RP04/05/06

FORMATTER
CZRJBCO

AH-9187C-MC

FICHE 1 OF 1

AUG 1978
digital
MADE IN USA

```
                                .REM @
```

I D E N T I F I C A T I O N

PRODUCT CODE: AC-9185C-MC

PRODUCT NAME: CZRJBC0 RP04/5/6 FORMATTER PROGRAM

PRODUCT DATE: 25-MARCH-1978

MAINTAINER: DIAGNOSTIC ENGINEERING

AUTHOR: C. HESS

CONTENTS

```
                        1.      ABSTRACT
                                --------


                                THE RP04/5/6 FORMATTER PROGRAM FORMATS THE DISK PACK AND PERFORMS
                                A CURSORY CHECK OF THE PACK'S SURFACE.  THE PROGRAM ALLOWS THE
                                OPERATOR TO SPECIFY ADDRESS LIMITS, PATTERNS, AND EITHER 16 BIT OR
                                18 BIT FORMAT MODE.  THE PROGRAM VERIFIES EACH TRACK WRITTEN
                                AS WELL AS VERIFYING THE FORMAT OPERATION.




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

                        2.1     EQUIPMENT

                                PDP-11 PROCESSOR
                                8K MEMORY
                                TELETYPE
                                PROGRAM LOAD DEVICE
                                KW11-L OR KW11-P CLOCK
                                RH11 OR RH70 WITH 1 - 8 RP04, RP05, RP06 DISK DRIVES

                        2.2     PRELIMINARY PROGRAMS

                                RP04/5/6 DISKLESS CONTROLLER TEST
                                        PART 1 (MAINDEC-11-DZRJG)
                                        PART 2 (MAINDEC-11-DZRJH)

                                RP04/5/6 FUNCTIONAL CONTROLLER TEST
                                        PART 1 (MAINDEC-11-DZRJI)
                                        PART 2 (MAINDEC-11-DZRJJ)



                        3.      LOADING PROCEDURES
                                ------------------

                                THE PROGRAM MAY BE LOADED FROM PAPER TAPE USING THE ABSOLUTE
                                LOADER OR IT MAY BE LOADED FROM THE APPROPRIATE 'XXDP' MEDIA
                                USING THE ASSOCIATED LOADER.  THE PROGRAM MAY NOT BE INCLUDED
                                IN AN 'XXDP' CHAIN.



                        4.      STARTING PROCEDURES
                                -------------------

                        4.1     STARTING ADDRESSES

                                THE PROGRAM IS STARTED FROM LOCATION 200(8) IF THE ADDRESS OF THE
                                RH11 OR RH70 WILL NOT BE CHANGED
```

THE PROGRAM IS STARTED FROM LOCATION 204(8) IF THE ADDRESS OF
THE RH11 OR RH70 IS TO BE CHANGED FROM THE PRELOADED VALUE.
(SEE SECTION 4.3)

4.2 OPERATION ACTION

1. LOAD THE PROGRAM INTO MEMORY (SEE SECTION 3).

2. LOAD THE STARTING ADDRESS - 200(8) OR 204(8).

3. SET THE SWITCHES AS REQUIRED AND PRESS 'START'.

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

4. THE PROGRAM WILL THEN TYPE THE FOLLOWING MESSAGE:

       'PROGRAM MODE (C OR F):'

   ENTER THE APPROPRIATE CODE: 'C' FOR 'CHECK' OPERATION OR 'F'
   FOR 'FORMAT & VERIFY'. IF A 'CARRIAGE RETURN' IS ENTERED IN
   RESPONSE TO THE REQUEST, THE PROGRAM WILL ASSUME 'FORMAT &
   VERIFY'.

5. THE PROGRAM WILL THEN ASK FOR THE FORMATTING MODE:

       'OPERATE IN 22 SECTOR (16 BIT) MODE (Y OR N)'

   ENTER THE APPROPRIATE CHARACTER: 'Y' FOR 16 BIT MODE OR 'N' FOR
   18 BIT MODE. IF A 'CARRIAGE RETURN' IS ENTERED IN RESPONSE TO
   THIS REQUEST, THE PROGRAM WILL ASSUME 16 BIT MODE.

6. THE PROGRAM WILL THEN ASK FOR A DRIVE:

       'DRIVE: '

   ENTER THE ADDRESS OF THE DRIVE TO BE FORMATTED. A 'PERIOD' OR
   'CARRIAGE RETURN' ENTRY WILL SELECT DRIVE 0. IF THE DRIVE SELECTED
   IS NOT AVAILABLE, THE PROGRAM WILL TYPE AN ERROR MESSAGE AND
   RETURN TO THE DRIVE ADDRESS REQUEST.

7. AFTER THE OPERATOR HAS SELECTED A DRIVE, THE PROGRAM WILL ASK
   FOR ADDRESS LIMITS FOR THE SELECTED DRIVE:

       'ENTER ADDRESS LIMITS: '

   THE PREVIOUSLY SELECTED OR DEFAULT VALUES FOR BEGINNING CYLINDER
   AND TRACK AND FOR ENDING CYLINDER AND TRACK WILL BE TYPED OUT.
   IF A 'CARRIAGE RETURN' IS TYPED AS A RESPONSE, THE PRESENT VALUE
   WILL BE USED; IF A 'PERIOD' IS TYPED, THE PROGRAM WILL BYPASS
   THE REMAINING ENTRIES AND WILL USE THEIR PRESENT VALUES. NOTE
   THAT THE CYLINDER AND TRACK VALUES ARE D E C I M A L NUMBERS.
   THE ADDRESS SPECIFIED BY THE BEGINNING CYLINDER AND TRACK MUST
   BE LESS THAN THE ADDRESS SPECIFIED BY THE ENDING CYLINDER AND
   TRACK ADDRESS.

7. THE PROGRAM WILL THEN ASK FOR THE DATA PATTERN:

```
'SELECT DATA PATTERN
  (0) ZERO'S
  (1) ONES
  (2) WORST CASE:'
```

ENTER THE CODE FOR THE REQUIRED PATTERN. 'CARRIAGE RETURN' OR 'PERIOD' ENTRIES WILL CAUSE THE PROGRAM TO USE THE 'WORST CASE' PATTERN.

THE 'WORST CASE' PATTERN IS THE FOLLOWING OCTAL SEQUENCE REPEATED THROUGH THE DATA AREA OF THE SECTOR:

```
165555
133333
```

8. THE PROGRAM WILL THEN TYPE:

```
'STARTING FORMAT ON DRIVE N'
```

OR

```
'STARTING CHECK ON DRIVE N'
```

9. THE OPERATOR CAN DETERMINE WHERE THE DRIVE IS DURING THE FORMAT OR CHECK OPERATION BY TYPING A 'CONTROL C'. THE PROGRAM WILL TYPE THE FOLLOWING MESSAGE:

```
'PRESENT ADDRESS IS: CXXX  TXX'
```

IF A 'CONTROL C' IS TYPED WHILE THE PROGRAM IS TYPING THE CURRENT ADDRESS, THE PROGRAM WILL ABORT THE FORMAT (OR CHECK) OPERATION AND RETURN TO THE 'DRIVE' REQUEST.

10. IF THE OPERATOR DOES NOT SELECT A DIFFERENT DRIVE WHEN THE FORMAT OR CHECK OPERATION HAS COMPLETED, THE PROGRAM WILL NOT ALTER THE ADDRESS LIMITS SPECIFIED AT THE START OF THE PREVIOUS OPERATION. IF THE FORMAT OR CHECK OPERATION IS TERMINATED BY A 'CONTROL C' OR IF A DIFFERENT DRIVE IS SELECTED, THE PROGRAM WILL RESET THE ADDRESS LIMITS TO THE VALUES APPROPRIATE FOR THE DRIVE TYPE.

11. TO CHANGE EITHER THE ADDRESSING MODE OR THE OPERATION MODE, THE PROGRAM MUST BE STARTED FROM LOCATION 200(8) OR 204(8) AGAIN.

4.3 RH11 - RH70 UNIBUS ADDRESS

THE PROGRAM ASSUMES THAT THE RH11 OR RH70 ADDRESSES START AT 176700 AND THAT THE VECTOR ADDRESS IS 254. THESE ADDRESSES MAY BE CHANGED WHEN THE PROGRAM IS STARTED FROM LOCATION 204(8). IF THE RH11 - RH70 IS NOT AT THE DEFAULT ADDRESS, THE PROGRAM MUST BE STARTED FROM 204(8) INITIALLY AS THE PROGRAM GOES THROUGH THE ADDRESS CHANGE ROUTINE AT INITIAL START ONLY. ENTER THE RH11 RH70 ADDRESS IN RESPONSE TO THE REQUEST FROM THE PROGRAM.

4.4 OTHER UNIBUS ADDRESSES

LOC	TAG	CONTENTS	FUNCTION
1144	$TKS	177560	TTY KEYBOARD STATUS REGISTER
1146	$TKB	177562	TTY KEYBOARD BUFFER REGISTER
1150	$TPS	177564	TTY PRINTER STATUS REGISTER
1152	$TPB	177566	TTY PRINTER BUFFER REGISTER
1176	$LKCSR	172540	KW11-P CONTROL REGISTER
1200	$LKCSB	172542	KW11-P COUNTER REGISTER
1202	$LPVEC	104	KW11-P VECTOR ADDRESS
1206	$LKS	177546	KW11-L CONTROL REGISTER
1210	$LKV	100	;ADDRESS OF KW11-L VECTOR

5. SWITCH REGISTER SETTINGS

SW<15>=1...HALT ON ERROR
SW<13>=1...INHIBIT ERROR TYPEOUTS
SW<10>=1...BELL ON ERROR
SW<09>=1...LOOP ON ERROR
SW<07>=1...PRINT SOFT ERROR REPORTS AS THEY OCCUR
SW<02>=1...DON'T DISPLAY SYSTEM STATUS AFTER INITIAL START
SW<01>=1...LOOP ON THE CURRENT TRACK
SW<00>=1...LOOP THE PROGRAM ON THE SELECTED DRIVE

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 RP04/5/6 INTERRUPT. THE 'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

'SWR = NNNNNN NEW ='

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

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

6. ERROR MESSAGES

1. 'RH11 INTERRUPT OCCURRED (RPAS=0) - AN INTERRUPT OCCURRED, BUT NOTHING ON THE MASSBUS IS INDICATING AN ATTENTION.

2. 'UNEXPECTED ATTENTION OCCURRED' - THE INDICATED DRIVE INTERRUPTED, BUT NO INTERRUPT WAS EXPECTED FROM THE INDICATED DRIVE.

3. 'MASSBUS PARITY ERROR (MCPE=1)' - A CONTROL BUS PARITY ERROR WAS DETECTED BY THE RH11 WHEN THE INDICATED REGISTER WAS READ.

4. 'MASSBUS PARITY ERROR (PAR=1)' - A CONTROL BUS PARITY ERROR OCCURRED WHEN THE INDICATED REGISTER WAS WRITTEN.

5. 'ADDRESS PLUG CHANGE BIT SET' - THE PROGRAM FOUND THE 'OPE' BIT SET FOR THE INDICATED DRIVE.

6. 'RH11 DIDN'T RESPOND TO ADDRESSING' - THE PROGRAM ACCESSED THE RH11 AT THE INDICATED ADDRESS AND RECEIVED NO RESPONSE.

7. 'DRIVE OFFLINE' - THE INDICATED DRIVE HAS GONE OFFLINE

8. 'PERSISTENT DRIVE UNSAFE ERROR' - THE INDICATED DRIVE HAS BECOME UNSAFE AND THE CONDITION CANNOT BE CLEARED BY ISSUING 'DRIVE CLEAR' INSTRUCTIONS.

9. 'UNCORRECTABLE MASSBUS PARITY ERROR' - THE PROGRAM ATTEMPTED TO PERFORM AN OPERATION AND DETECTED 3 SUCESSIVE MASSBUS PARITY ERRORS OR THE PROGRAM ATTEMPTED TO CLEAR A 'PAR' ERROR IN THE DRIVE AND A PARITY ERROR OCCURRED.

10. 'SOFTWARE TIMEOUT' - THE OPERATION FAILED TO COMPLETE WITHIN 1 SECOND.

11. 'DRIVE UNSAFE ERROR' - A NON-PERSISTENT UNSAFE ERROR OCCURRED DURING THE OPERATION.

12. 'CONTROLLER/DRIVE ERROR DURING WRITE' - THE INDICATED NON-DATA ERROR WAS DETECTED DURING A FORMAT OPERATION.

13. 'CONTROLLER/DRIVE ERROR DURING WRITE CHECK' - A NON-DATA ERROR OCCURRED DURING THE WRITE CHECK.

14. 'DATA ERROR DURING WRITE CHECK' - A DATA RELATED ERROR OCCURRED DURING A WRITE CHECK OPERATION. A DATA ERROR IS CONSIDERED TO BE A 'DCK' ERROR.

15. 'RETRIES NOT SUCESSFUL - SECTOR NOT ACCEPTABLE' - THE INDICATED SECTOR FAILED DURING RETRY FOLLOWING A DATA ERROR.

16. 'CONTROLLER/DRIVE ERROR VERIFYING HEADERS' - AN ERROR OCCURRED DURING THE VERIFICATION PASS AFTER FORMATTING.

17. ''HCE' ERROR VERIFYING HEADERS' - A HEADER ERROR OCCURRED DURING

THE VERIFICATION PASS AFTER FORMATTING.

18. 'CYLINDER FIELD IN HEADER IS NOT CORRECT' - THE CYLINDER FIELD FROM A HEADER READ DURING THE VERIFICATION PASS IS NOT CORRECT.

19. 'WRITE CHECK ERROR' - THE RH11 REPORTED A WRITE CHECK ERROR AND NO DRIVE ERRORS WERE SET.

20. 'HARDWARE ERROR DURING WRITE CHECK' - AN ERROR THAT WAS NEITHER A DATA ERROR, WRITE CHECK ERROR, NOR CONTROLLER ERROR OCCURRED DURING A WRITE CHECK. THESE ERRORS COULD BE ONE OF THE FOLLOWING: 'OPI', 'DTE', 'HCRC', 'HCE', OR 'FER'.

7. MISCELLANEOUS

7.1 FORMAT TIME

IT TAKES APPROXIMATELY 8 MINUTES TO FORMAT AN ENTIRE RP04/5 PACK AND APPROXIMATELY 16 MINUTES TO FORMAT AN RP06 PACK. THE 'CHECK' MODE TIME FOR AN ENTIRE PACK IS 4 MINUTES FOR RP04/5'S AND 8 MINUTES FOR RP06'S.

7.2 HALTING THE PROGRAM

THE OPERATOR SHOULD NOT HALT THE PROGRAM DURING A FORMAT OPERATION. HALTING THE PROGRAM MAY LEAVE A SECTOR INCORRECTLY FORMATTED. TO TERMINATE THE FORMAT, TYPE A 'CONTROL C' AND WHILE THE PROGRAM IS TYPING THE ADDRESS, TYPE ANOTHER 'CONTROL C'; THIS SEQUENCE RETURNS THE PROGRAM TO THE DRIVE ADDRESS ENTRY ROUTINE.

7.3 SURFACE VERIFICATION

THE FORMATTER PROGRAM IS NOT INTENDED TO BE USED TO PERFORM DISK PACK VERIFICATION. IF THE PROGRAM REPORTS A SECTOR AS BEING 'NOT ACCEPTABLE', THIS MAY IN FACT INDICATE A BAD SURFACE; HOWEVER, SECTORS WHICH 'PASSED' MAY OR MAY NOT BE GOOD.

8. PROGRAM DESCRIPTION

8.1 FORMAT OPERATION

THE PROGRAM FORMATS THE PACK ONE TRACK AT A TIME BETWEEN THE LIMITS SPECIFIED BY THE DRIVE.

THE FORMAT OPERATION CONSISTS OF A WRITE HEADER AND DATA COMMAND FOR THE ENTIRE TRACK FOLLOWED BY A WRITE CHECK HEADER AND DATA COMMAND. IF AN ERROR OCCURS DURING THE WRITE, THE PROGRAM WILL RETRY THE WRITE BEFORE CONTINUING. IF A DATA RELATED ERROR IS DETECTED

DURING THE WRITE CHECK (A DATA ERROR IS DEFINED AS ONE OF
THE FOLLOWING ERRORS: 'DCK', 'OPI', 'DTE', 'HCRC', 'HCE',
OR 'FER'), THE SECTOR IN ERROR WILL BE REWRITTEN. THE PROGRAM WILL
CHECK THE SECTOR TWICE; IF A DATA ERROR IS DETECTED IN THE SECTOR
DURING EITHER OF THE WRITE CHECKS, THE PROGRAM WILL DECLARE
THE SECTOR AS BEING 'UNACCEPTABLE'. FOLLOWING THIS SEQUENCE,
THE REMAINDER OF THE TRACK IS CHECKED. IF DATA ERRORS ARE ENCOUNTERED
IN ANY OF THE REMAINING SECTORS, EACH SECTOR WILL BE HANDLED AS
DESCRIBED ABOVE.

IF A NON-DATA RELATED ERROR OCCURS DURING THE WRITE CHECK, THE
PROGRAM WILL RETRY THE COMMAND FOR THE ENTIRE TRACK BEFORE PROCEEDING
TO THE NEXT TRACK.

IT SHOULD BE NOTED THAT THE FORMATTER PROGRAM FORMATS THE PACK ONLY
AND IS NOT DIRECTLY VERIFYING THE SURFACE OF THE PACK. ERRORS
THAT ARE ENCOUNTERED MAY BE THE RESULT OF BAD AREAS ON THE PACK
BUT, AS THE PROGRAM IS NOT DESIGNED TO PERFORM AN EXHAUSTIVE CHECK
OF THE DISK PACK SURFACE, THE PROGRAM CANNOT MAKE THIS CONCLUSION.
IN GENERAL, HOWEVER, THE OPERATOR CAN ASSUME THAT SECTORS WHICH
THE PROGRAM CALLS 'UNACCEPTABLE' INDICATE BAD AREAS OF THE PACK;
UNFORTUNATELY, SECTORS WHICH 'PASS' CANNOT BE ASSUMED TO BE GOOD.

DURING THE FORMAT OPERATION, THE PROGRAM FILLS THE DATA FIELD
WITH THE PATTERN SELECTED BY THE OPERATOR. THE KEYWORDS IN THE
HEADER ARE ALWAYS SET TO ZERO.

8.2 CHECK OPERATION

THE CHECK OPERATION IS IDENTICAL TO THE WRITE CHECK PORTION OF
THE FORMAT OPERATION OPERATION DESCRIBED IN SECTION 8.1 EXCEPT
THAT THE PROGRAM WILL NOT RE-WRITE ERROR SECTORS.

8.3 POSITIONER VERIFICATION

AFTER THE PROGRAM COMPLETES THE FORMAT OPERATION, THE
POSITIONER IS RETURNED TO THE STARTING CYLINDER AND THE HEADER
FROM SECTOR 0 ON THE STARTING TRACK IS READ. THE CYLINDER ADDRESS
FIELD FROM THE HEADER IS COMPARED TO THE REQUESTED CYLINDER; IF
THE CYLINDER ADDRESSES DO NOT COMPARE, AN ERROR MESSAGE IS TYPED.
THE PROGRAM CHECKS THE CYLINDER ADDRESS FIELD FROM THE HEADER
OF SECTOR 0 ON THE BEGINNING TRACK OF EACH CYLINDER FORMATTED.
THIS CHECK IS PERFORMED TO CONFIRM THAT THE DISK'S POSITIONER
ADVANCED PROPERLY DURING THE FORMAT OPERATION.

9. PROGRAM LISTING

@

```
.TITLE  CZRJBC0, RP04/5/6 FMTR
;*COPYRIGHT (C) 1976,1978
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
```

```
                                ;*PROGRAM BY C. HESS
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                                ;*

                                .SBTTL  OPERATIONAL SWITCH SETTINGS
                                ;*
                                ;*      SWITCH                  USE
                                ;*      ------          --------------------
                                ;*        15            HALT ON ERROR
                                ;*        13            INHIBIT ERROR TYPEOUTS
                                ;*        10            BELL ON ERROR
                                ;*         9            LOOP ON ERROR
                                ;*         2            DON'T DISPLAY SYSTEM STATUS AFTER INITIAL START
                                ;*         1            LOOP ON THE CURRENT TRACK
                                ;*         0            LOOP THE PROGRAM ON THE SELECTED DRIVE

                                .SBTTL  TRAP CATCHER

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

                                .SBTTL  ACT11 HOOKS

                                ;;**********************************************************************
                                ;HOOKS REQUIRED BY ACT11
        000200                          $SVPC=.                 ;SAVE PC
        000046                          .=46
000046  005324                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
        000052                          .=52
000052  020000                          .WORD   20000           ;;2)SET LOC.52 TO 20000
        000200                          .=$SVPC                 ;; RESTORE PC

                                .SBTTL  STARTING ADDRESS = 200
        000200                          .=200
000200  000137  002110                  JMP     BEGIN1          ;NORMAL STARTING ADDRESS

                                .SBTTL  STARTING ADDRESS TO CHANGE THE RH11 ADDRESS = 204
000204  000137  002100                  JMP     BEGIN           ;CHANGE THE RH11 ADDRESS

                                .SBTTL  BASIC DEFINITIONS

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

                                ;*MISCELLANEOUS DEFINITIONS
        000011                  HT=     11              ;;CODE FOR HORIZONTAL TAB
        000012                  LF=     12              ;;CODE FOR LINE FEED
```

```
529          000015                  CR=     15              ;;CODE FOR CARRIAGE RETURN
530          000200                  CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
531          177776                  PS=     177776          ;;PROCESSOR STATUS WORD
532                                  .EQUIV  PS,PSW
533          177774                  STKLMT= 177774          ;;STACK LIMIT REGISTER
534          177772                  PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
535          177570                  DSWR=   177570          ;;HARDWARE SWITCH REGISTER
536          177570                  DDISP=  177570          ;;HARDWARE DISPLAY REGISTER
537
538                                  ;*GENERAL PURPOSE REGISTER DEFINITIONS
539          000000                  R0=     %0              ;;GENERAL REGISTER
540          000001                  R1=     %1              ;;GENERAL REGISTER
541          000002                  R2=     %2              ;;GENERAL REGISTER
542          000003                  R3=     %3              ;;GENERAL REGISTER
543          000004                  R4=     %4              ;;GENERAL REGISTER
544          000005                  R5=     %5              ;;GENERAL REGISTER
545          000006                  R6=     %6              ;;GENERAL REGISTER
546          000007                  R7=     %7              ;;GENERAL REGISTER
547          000006                  SP=     %6              ;;STACK POINTER
548          000007                  PC=     %7              ;;PROGRAM COUNTER
549
550                                  ;*PRIORITY LEVEL DEFINITIONS
551          000000                  PR0=    0               ;;PRIORITY LEVEL 0
552          000040                  PR1=    40              ;;PRIORITY LEVEL 1
553          000100                  PR2=    100             ;;PRIORITY LEVEL 2
554          000140                  PR3=    140             ;;PRIORITY LEVEL 3
555          000200                  PR4=    200             ;;PRIORITY LEVEL 4
556          000240                  PR5=    240             ;;PRIORITY LEVEL 5
557          000300                  PR6=    300             ;;PRIORITY LEVEL 6
558          000340                  PR7=    340             ;;PRIORITY LEVEL 7
559
560                                  ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
561          100000                  SW15=   100000
562          040000                  SW14=   40000
563          020000                  SW13=   20000
564          010000                  SW12=   10000
565          004000                  SW11=   4000
566          002000                  SW10=   2000
567          001000                  SW09=   1000
568          000400                  SW08=   400
569          000200                  SW07=   200
570          000100                  SW06=   100
571          000040                  SW05=   40
572          000020                  SW04=   20
573          000010                  SW03=   10
574          000004                  SW02=   4
575          000002                  SW01=   2
576          000001                  SW00=   1
577                                  .EQUIV  SW09,SW9
578                                  .EQUIV  SW08,SW8
579                                  .EQUIV  SW07,SW7
580                                  .EQUIV  SW06,SW6
581                                  .EQUIV  SW05,SW5
582                                  .EQUIV  SW04,SW4
583                                  .EQUIV  SW03,SW3
584                                  .EQUIV  SW02,SW2
```

```
                                .EQUIV  SW01,SW1
                                .EQUIV  SW00,SW0

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

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

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

                                .SBTTL  RH11 REGISTERS

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

                                ;CONTROL AND STATUS REGISTER 1 (RPCS1)

    000100                      IE=     100                     ;INTERRUPT ENABLE (BIT #6)
    000200                      RDY=    200                     ;READY (BIT #7)
    000400                      A16=    400                     ;HIGH ORDER BUS ADDRESS BIT (BIT #8)
```

```
                                ;HIGH ORDER BUS ADDRESS BIT (BIT #9)
                                ;PORT SELECT (BIT #10)
                                ;MASSBUSS PARITY ERROR (BIT #13)
                                ;TRANSFER ERROR (BIT #14)
                                ;SPECIAL CONDITION (BIT #15)

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

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

                                ;CONTROL AND STATUS REGISTER 2 (RPCS2)

                                ;UNIT SELECT (BIT #0)
                                ;UNIT SELECT (BIT #1)
                                ;UNIT SELECT (BIT #2)
                                ;BUS ADDRESS INCREMENT INHIBIT (BIT #3)
                                ;MASSBUS PARITY TEST (BIT #4)
                                ;CLEAR (BIT #5)
                                ;INPUT READY (BIT #6)
                                ;OUTPUT READY (BIT #7)
                                ;MASS BUS PARITY ERROR (BIT #8)
                                ;MISSED TRANSFER ERROR (BIT #9)
                                ;PROGRAM ERROR (BIT #10)
                                ;NON EXISTENT MEMORY (BIT #11)
                                ;NON EXISTENT DRIVE (BIT #12)
                                ;UNIBUS PARITY ERROR (BIT #13)
                                ;WRITE CHECK ERROR (BIT #14)
                                ;DATA LATE (BIT #15)

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


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

                                .SBTTL  RP04/5/6 REGISTERS

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

                                ;CONTROL AND STATUS 1 REGISTER. (#00)

                                ;GO BIT (BIT #0)
                                ;FUNCTION CODE BIT #1
                                ;FUNCTION CODE BIT #2
                                ;FUNCTION CODE BIT #3
                                ;FUNCTION CODE BIT #4
                                ;FUNCTION CODE BIT #5
                                ;DEVICE AVAILABLE (BIT #11)

                                ;DRIVE STATUS REGISTER (RPDS1) (#01)

                                ;DF5=   1                       DRIVE FORWARD 5''/SEC.  (BIT #0)
                                ;DFF20= 2                       ;DRIVE FORWARD 20''/SEC.  (BIT #1)
                                ;DIGB=  4                       ;DRIVE TO INNER GUARD BAND (BIT #2)
```

```
                                ;GO REVERSE (BIT #3)
                                ;DIFFERENCE LESS THAN 64 (BIT #4)
                                ;DIFFERENCE EQUALS 1 (BIT #5)
                                ;VOLUME VALID (BIT #6)
                                ;DRIVE READY (BIT #7)
                                ;DRIVE PRESENT (BIT #8)
                                ;PROGRAMABLE (BIT #9)
                                ;LAST SECTOR TRANSFERRED (BIT #10)
                                ;WRITE LOCK (BIT #11)
                                ;MEDIUM ON-LINE (BIT #12)
                                ;POSITIONING OPERATION IN PROGRESS (BIT #13)
                                ;COMPOSITE ERROR (BIT #14)
                                ;ATTENTION ACTIVE (BIT #15)

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

                                ;ILLEGAL FUNCTION (BIT #0)
                                ;ILLEGAL REGISTER (BIT #1)
                                ;REGISTER MODIFICATION REFUSED (BIT #2)
                                ;PARITY ERROR (BIT #3)
                                ;FORMAT ERROR (BIT #4)
                                ;WRITE CLOCK FAIL (BIT #5)
                                ;ECC HARD ERROR (BIT #6)
                                ;HEADER COMPARE ERROR (BIT #7)
                                ;HEADER CRC ERROR (BIT #8)
                                ;ADDRESS OVERFLOW ERROR (BIT #9)
                                ;INVALID ADDRESS ERROR (BIT #10)
                                ;WRITE LOCK ERROR (BIT #11)
                                ;DRIVE TIMING ERROR (BIT #12)
                                ;OPERATION INCOMPLETE (BIT #13)
                                ;DRIVE UNSAFE (BIT #14)
                                ;DATA CHECK ERROR (BIT 15)

                        ;MAINTAINABILITY REGISTER (RPMR)(#03)

                                ;DIAGINOSTIC MODE (BIT #0)
                                ;MAINTAINABILITY CLOCK (BIT #1)
                                ;MAINTAINABILITY INDEX (BIT #2)
                                ;MAINTAINABILITY SECTOR CLOCK (BIT #3)
                                ;MAINTAINABILITY READ (BIT #4)
                                ;MAINTAINABILITY WRITE (BIT #5)
                                ;MAINTAINABILITY SYNC DETECTED (BIT #7)

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

                                ;DEVICE 0 (BIT #0)
                                ;DEVICE 1 (BIT #1)
                                ;DEVICE 2 (BIT #2)
                                ;DEVICE 3 (BIT #3)
                                ;DEVICE 4 (BIT #4)
                                ;DEVICE 5 (BIT #5)
                                ;DEVICE 6 (BIT #6)
                                ;DEVICE 7 (BIT #7)

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

```
753
754                                     ;DRIVE TYPE REGISTER (RPDT) (#06)
755
756             000001                  DT00=   1               ;DRIVE TYPE NUMBER BIT 1
757             000002                  DT01=   2               ;DRIVE TYPE NUMBER BIT 2
758             000004                  DT02=   4               ;DRIVE TYPE NUMBER BIT 3
759             000010                  DT03=   10              ;DRIVE TYPE NUMBER BIT 4
760             000020                  DT04=   20              ;DRIVE TYPE NUMBER BIT 5
761             000040                  DT05=   40              ;DRIVE TYPE NUMBER BIT 6
762             000100                  DT06=   100             ;DRIVE TYPE NUMBER BIT 7
763             000200                  DT07=   200             ;DRIVE TYPE NUMBER BIT 8
764             000400                  DT08=   400             ;DRIVE TYPE NUMBER BIT 9
765             004000                  DRQ=    4000            ;DRIVE REQUEST REQUIRED (BIT #11)
766             020000                  MOH=    20000           ;MOVING HEAD (BIT #13)
767             040000                  TAP=    40000           ;TAPE DRIVE (BIT #14)
768             100000                  NBA=    100000          ;NOT BLOCK ADDRESSED (BIT #15)
769
770                                     ;LOOK-AHEAD REGISTER (RPLA) (#07)
771
772             000001                  EXT1=   1               ;EXTENSION 1 (BIT #0)
773             000002                  EXT2=   2               ;EXTENSION 2 (BIT #1)
774             000004                  EXT4=   4               ;EXTENSION 3 (BIT #2)
775             000010                  EXT10=  10              ;EXTENSION 4 (BIT #3)
776             000020                  EXT20=  20              ;EXTENSION 5 (BIT #4)
777             000040                  EXT40=  40              ;EXTENSION 6 (BIT #5)
778             000100                  SC1=    100             ;SECTOR COUNT FIELD 0 (BIT #6)
779             000200                  SC2=    200             ;SECTOR COUNT FIELD 1 (BIT #7)
780                                     ;SC4=   400             ;SECTOR COUNT FIELD 2 (BIT #8)
781             001000                  SC10=   1000            ;SECTOR COUNT FIELD 3 (BIT #9)
782             002000                  SC20=   2000            ;SECTOR COUNT FIELD 4 (BIT #10)
783             004000                  TRK1=   4000            ;TRACK FIELD 1 (BIT #11)
784             010000                  TRK2=   10000           ;TRACK FIELD 2 (BIT #12)
785             020000                  TRK4=   20000           ;TRACK FIELD 3 (BIT #13)
786             040000                  TRK10=  40000           ;TRACK FIELD 4 (BIT #14)
787             100000                  TRK20=  100000          ;TRACK FIELD 5 (BIT #15)
788
789                                     ;RP04 ERROR REGISTER #2 (RPER2) (#10)
790
791             000001                  WCU=    1               ;WRITE CURRENT UNSAFE (BIT #0)
792             000002                  CSF=    2               ;CURRENT SINK FAILURE (BIT #1)
793             000004                  WSU=    4               ;WRITE SELECT UNSAFE (BIT #2)
794             000010                  CSU=    10              ;CURRENT SWITCH UNSAFE (BIT #3)
795             000020                  MSE=    20              ;MOTOR SEQUENCE ERROR (BIT #4)
796             000040                  TDF=    40              ;TRANSITIONS DETECTOR FAILURE (BIT #5)
797             000100                  TUF=    100             ;TRANSITIONS UNSAFE (BIT #6)
798             000200                  FEN=    200             ;FAILSAFE ENABLED (BIT #7)
799             000400                  WRU=    400             ;WRITE READY UNSAFE (BIT #8)
800             001000                  MHS=    1000            ;MULTIPLE HEAD SELECT (BIT #9)
801             002000                  NHS=    2000            ;NO HEAD SELECTION (BIT #10)
802             004000                  IXE=    4000            ;INDEX ERROR (BIT #11)
803             010000                  VU30=   10000           ;30VOLT UNSAFE (BIT #12)
804             020000                  PLU=    20000           ;PLO UNSAFE (BIT #13)
805             100000                  ACU=    100000          ;AC UNSAFE (BIT #15)
806
807                                     ;RP05/6 ERROR REGISTER #02 (RPER2) (#10)
808
```

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

                        ;OFFSET REGISTER (RPOF) (#11)

                        OF25=   1               ;OFFSET 25 MICRO INCHES (BIT #0)
                        OF50=   2               ;OFFSET 50 MICRO INCHES (BIT #1)
                        OF100=  4               ;OFFSET 100 MICRO INCHES (BIT #2)
                        OF200=  10              ;OFFSET 200 MICRO INCHES (BIT #3)
                        OF400=  20              ;OFFSET 400 MICRO INCHES (BIT #4)
                        OF800=  40              ;OFFSET 800 MICRO INCHES (BIT #5)
                        OFREV=  200             ;OFFSET NEGATIVE (REVERSE) (BIT #5)
                        HCI=    2000            ;HEADER COMPARE INHIBIT (BIT #10)
                        ECI=    4000            ;ERROR CORRECTION CODE INHIBIT (BIT #11)
                        FMT22=  10000           ;FORMAT BIT (BIT #12

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

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

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

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

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

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

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

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

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

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

                                .SBTTL  RP04/5/6 DRIVER COMMANDS

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

000101                          RNOP    =       101             ;NO OPERATION
000103                          UNLOAD  =       103             ;UNLOAD
000105                          SEEK    =       105             ;SEEK
000107                          RECAL   =       107             ;RECALIBRATE
000111                          DRVCLR  =       111             ;DRIVE CLEAR
000113                          RELSE   =       113             ;RELEASE
000115                          OFFSET  =       115             ;OFFSET
000117                          RTC     =       117             ;RETURN TO CENTER LINE
000121                          READIN  =       121             ;READ IN PRESET
000123                          ACK     =       123             ;PACK ACKNOWLEDGE
000131                          SEARCH  =       131             ;SEARCH
000141                          GETREG  =       141             ;GET REGISTERS
000143                          SETFMT  =       143             ;SET FORMAT (& ECI OR HCI)
000145                          SELDRV  =       145             ;SELECT DRIVE
000151                          WCKD    =       151             ;WRITE CHECK DATA
000153                          WCKHD   =       153             ;WRITE CHECK HEADER & DATA
000161                          WRTDAT  =       161             ;WRITE DATA
000163                          WRTHD   =       163             ;WRITE HEADER & DATA
000171                          RDDAT   =       171             ;READ DATA
000173                          RDHD    =       173             ;READ HEADER & DATA
```

```
                                .SBTTL  COMMON TAGS

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

            001100                      .=1100
001100                          $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                  $PASS:  .WORD   0                ;;CONTAINS PASS COUNT
001102     000                  $TSTNM: .BYTE   0                ;;CONTAINS THE TEST NUMBER
001103     000                  $ERFLG: .BYTE   0                ;;CONTAINS ERROR FLAG
001104  000000                  $ICNT:  .WORD   0                ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000                  $LPADR: .WORD   0                ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000                  $LPERR: .WORD   0                ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000                  $ERTTL: .WORD   0                ;;CONTAINS TOTAL ERRORS DETECTED
001114     000                  $ITEMB: .BYTE   0                ;;CONTAINS ITEM CONTROL BYTE
001115     001                  $ERMAX: .BYTE   1                ;;CONTAINS MAX. ERRORS PER TEST
001116  000000                  $ERRPC: .WORD   0                ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000                  $GDADR: .WORD   0                ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000                  $BDADR: .WORD   0                ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000                  $GDDAT: .WORD   0                ;;CONTAINS 'GOOD' DATA
001126  000000                  $BDDAT: .WORD   0                ;;CONTAINS 'BAD' DATA
001130  000000                          .WORD   0                ;;RESERVED--NOT TO BE USED
001132  000000                          .WORD   0
001134     000                  $AUTOB: .BYTE   0                ;;AUTOMATIC MODE INDICATOR
001135     000                  $INTAG: .BYTE   0                ;;INTERRUPT MODE INDICATOR
001136  000000                          .WORD   0
001140  177570                  SWR:    .WORD   DSWR             ;;ADDRESS OF SWITCH REGISTER
001142  177570                  DISPLAY: .WORD  DDISP            ;;ADDRESS OF DISPLAY REGISTER
001144  177560                  $TKS:   177560                   ;;TTY KBD STATUS
001146  177562                  $TKB:   177562                   ;;TTY KBD BUFFER
001150  177564                  $TPS:   177564                   ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566                  $TPB:   177566                   ;;TTY PRINTER BUFFER REG. ADDRESS
001154     000                  $NULL:  .BYTE   0                ;;CONTAINS NULL CHARACTER FOR FILLS
001155     002                  $FILLS: .BYTE   2                ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156     012                  $FILLC: .BYTE   12               ;;INSERT FILL CHARS. AFTER A 'LINE FEED''
001157     000                  $TPFLG: .BYTE   0                ;;'TERMINAL AVAILABLE'' FLAG (BIT<07>=0=YES)
001160  000000                  $ESCAPE:0                        ;;ESCAPE ON ERROR ADDRESS
001162  177607  000377          $BELL:  .ASCIZ  <207><377><377>  ;;CODE FOR BELL
001166     077                  $QUES:  .ASCII  /?/              ;;QUESTION MARK
001167     015                  $CRLF:  .ASCII  <15>             ;;CARRIAGE RETURN
001170  000012                  $LF:    .ASCIZ  <12>             ;;LINE FEED
                                ;;**********************************************************************
            000015              CR      =       15
            000012              LF      =       12
001172  176700                  $RPADR: .WORD   176700           ;RH11/RP04/5/6 UNIBUS ADDRESS
001174  000254                  $RPVEC: .WORD   254              ;RH11 INTERRUPT VECTOR
001176  172540                  $LKCSR: .WORD   172540           ;ADDRESS OF KW11-P CSR
001200  172542                  $LKCSB: .WORD   172542           ;ADDRESS OF KW11-P COUNTER BUFFER
001202  000104  000106          $LPVEC: .WORD   104,106          ;ADDRESS OF KW11-P VECTOR
001206  177546                  $LKS:   .WORD   177546           ;ADDRESS OF KW11-L CONTROL REGISTER
001210  000100  000102          $LLVEC: .WORD   100,102          ;ADDRESS OF KW11-L VECTOR
001214  000000                  DRIVE:  .WORD   0                ;CONTAINS DRIVE NUMBER SELECTED
001216  000000                  SOFSW:  .WORD   0                ;CONTENTS ARE FOR SOFTWARE DECISIONS
001220  000000                  MODE:   .WORD   0                ;'FORMAT & VERIFY' OR 'CHECK' MODE INDICATOR
001222  000632                  ENDCYL: .WORD   410.             ;ENDING CYLINDER
```

COMMON TAGS SEQ 0019

```
                                BEGCYL: .WORD   0               ;STARTING CYLINDER
                                ENDTRK: .WORD   18.             ;ENDING TRACK
                                BEGTRK: .WORD   0               ;STARTING TRACK
                                TTRKS:  .WORD   0               ;TOTAL # OF TRACKS TO BE FORMATTED
                                TTRKSC: .WORD   0               ;TOTAL # OF TRACKS COUNTER
                                TRKCNT: .WORD   0               ;COUNTS TRKS FROM 0-18 PER CYL
                                PATSEL: .WORD   0               ;CONTAINS PATTERN SELECTED
                                PATA:   .WORD   0               ;1ST WORD OF PATTERN
                                PATB:   .WORD   0               ;2ND WORD OF PATTERN
                                CYLCK:  .WORD   0               ;STORE CYLINDER ADDRESS FOR FORMAT VERIFICATION
                                RETRY:  .WORD   0               ;MAINTAINS # OF WRITE RETRIES MADE
                                SAVSEC: .WORD   0               ;CONTAINS LAST BAD SECTOR ON FORMAT
                                SAVWC:  .WORD   0               ;CONTAINS WC FOR REMAINING SECTORS ON ERROR
                                WC:     .WORD   0               ;20 OR 22 SECTOR TRACK SIZE (IN WORDS)
                                MWC:    .WORD   0               ;2'S COMPLEMENT OF 'WC'
                                HEDERR: .WORD   0               ;POSITIONING ERROR DURING FORMAT INDICATOR
                                SEC20:  .WORD   0               ;20 OR 22 SECTOR MODE INDICATOR
                                                                ;0 = 20 SECTOR MODE
                                                                ;1'S = 22 SECTOR MODE
                                MAXSEC: .WORD   0               ;MAXIMUM SECTOR ADDRESS (FOR EITHER 20 OR 22 SECTOR
                                                                ;FORMAT)
                                DS.CYL: .WORD   0               ;ADDRESS OF CURRENT CYLINDER
                                DS.TRK: .WORD   0               ;ADDRESS OF CURRENT TRACK
                                DDRIVE: .WORD   0               ;DRIVE NUMBER OF 'DRIVER' ERROR MESSAGES
                                ATTN:   .WORD   0               ;ATTENTION REGISTER IMAGE FOR 'DRIVER'
                                                                ;ERROR MESSAGES
                                CNTLC:  .WORD   0               ;ADDRESS OF '^C' RETURN
                                CHGADR: .WORD   0               ;'CHANGE RH11 ADDRESS' FLAG
                                EFLG:   .WORD   0               ;'ERROR DURING WRITE CHECK' FLAG

                                ;RH11/RP04/5/6 REGISTERS STORED HERE AFTER AN OPERATION

                                RP.REG: .WORD   0               ;RPCS1
                                        .WORD   0               ;RPWC
                                        .WORD   0               ;RPBA
                                        .WORD   0               ;RPDA
                                        .WORD   0               ;RPCS2
                                        .WORD   0               ;RPDS1
                                        .WORD   0               ;RPER1
                                        .WORD   0               ;RPAS
                                        .WORD   0               ;RPLA
                                        .WORD   0               ;RPDB
                                        .WORD   0               ;RPMR
                                        .WORD   0               ;RPDT
                                        .WORD   0               ;RPSN
                                        .WORD   0               ;RPOF
                                        .WORD   0               ;RPCA
                                        .WORD   0               ;RPCC
                                        .WORD   0               ;RPER2
                                        .WORD   0               ;RPER3
                                        .WORD   0               ;RPEC1
                                        .WORD   0               ;RPEC2


                                ;DATA/PARAMETER BLOCK - USED FOR ALL DRIVE OPERATION
```

```
001356     000                        FMTDPB: .BYTE   0               ;DRIVE NUMBER
001357     000                                .BYTE   0               ;OFFSET VALUE OR FMT22,ECI, AND HCI
001360     000                                .BYTE   0               ;COMMAND
001361     000                                .BYTE   0               ;PSEL AND A17 AND A16
001362  000000                                .WORD   0               ;WORD COUNT (NEG)
001364  000000                                .WORD   0               ;BUFFER ADDRESS
001366     000                                .BYTE   0               ;SECTOR ADDRESS
001367     000                                .BYTE   0               ;TRACK ADDRESS
001370  000000                                .WORD   0               ;CYLINDER ADDRESS
001372  001306                                .WORD   RP.REG          ;ERROR TABLE POINTER
001374  000000                                .WORD   0               ;STATUS-ERROR INDICATOR
                                                                      ;BIT 15 = 1: ERROR OCCURRED
                                                                      ;BIT 07 = 1: DONE
                                                                      ;BIT 14-10 AND BIT 06-03
                                                                      ;INDICATE TYPE OF ERROR

                                      ;PARAMETER POINTER TABLE

001376  001430  000000  001224        TABLE:  PAR1,0,BEGCYL
001404  001443  000022  001230                PAR2,18.,BEGTRK
001412  001456  000000  001222                PAR3,0,ENDCYL
001420  001467  000022  001226                PAR4,18.,ENDTRK,0
001426  000000

                                      ;ASCII MESSAGES FOR ADDRESS PARAMETERS

001430  052123  051101  020124        PAR1:   .ASCIZ  @START CYL @
001436  054503  020114     000
001443     123  040524  052122        PAR2:   .ASCIZ  @START TRK @
001450  052040  045522  000040
001456  047105  020104  054503        PAR3:   .ASCIZ  @END CYL @
001464  020114     000
001467     105  042116  052040        PAR4:   .ASCIZ  @END TRK @
001474  045522  000040

                                      ;SECTOR BUFFER ADDRESS TABLE

001500  025324                        ADRTBL: .WORD   BUFP            ;ADDRESS OF SECTOR 0
001502  026334                                .WORD   BUFP+<520.*1.>  ;ADDRESS OF SECTOR 1
001504  027344                                .WORD   BUFP+<520.*2.>  ;ADDRESS OF SECTOR 2
001506  030354                                .WORD   BUFP+<520.*3.>  ;ADDRESS OF SECTOR 3
001510  031364                                .WORD   BUFP+<520.*4.>  ;ADDRESS OF SECTOR 4
001512  032374                                .WORD   BUFP+<520.*5.>  ;ADDRESS OF SECTOR 5
001514  033404                                .WORD   BUFP+<520.*6.>  ;ADDRESS OF SECTOR 6
001516  034414                                .WORD   BUFP+<520.*7.>  ;ADDRESS OF SECTOR 7
001520  035424                                .WORD   BUFP+<520.*8.>  ;ADDRESS OF SECTOR 8
001522  036434                                .WORD   BUFP+<520.*9.>  ;ADDRESS OF SECTOR 9
001524  037444                                .WORD   BUFP+<520.*10.> ;ADDRESS OF SECTOR 10
001526  040454                                .WORD   BUFP+<520.*11.> ;ADDRESS OF SECTOR 11
001530  041464                                .WORD   BUFP+<520.*12.> ;ADDRESS OF SECTOR 12
001532  042474                                .WORD   BUFP+<520.*13.> ;ADDRESS OF SECTOR 13
001534  043504                                .WORD   BUFP+<520.*14.> ;ADDRESS OF SECTOR 14
001536  044514                                .WORD   BUFP+<520.*15.> ;ADDRESS OF SECTOR 15
001540  045524                                .WORD   BUFP+<520.*16.> ;ADDRESS OF SECTOR 16
001542  046534                                .WORD   BUFP+<520.*17.> ;ADDRESS OF SECTOR 17
```

```
 1067  001544  047544                          .WORD   BUFP+<520.*18.> ;ADDRESS OF SECTOR 18
 1068  001546  050554                          .WORD   BUFP+<520.*19.> ;ADDRESS OF SECTOR 19
 1069  001550  051564                          .WORD   BUFP+<520.*20.> ;ADDRESS OF SECTOR 20
 1070  001552  052574                          .WORD   BUFP+<520.*21.> ;ADDRESS OF SECTOR 21
 1071
 1072                                  ;REMAINING WORD COUNT TABLE
 1073
 1074  001554  000404                  WCTBL:  .WORD   260.             ;REMAINING WORD COUNT AFTER SECTOR 0
 1075  001556  001010                          .WORD   260.+<260.*1.>   ;REMAINING WORD COUNT AFTER SECTOR 1
 1076  001560  001414                          .WORD   260.+<260.*2.>   ;REMAINING WORD COUNT AFTER SECTOR 2
 1077  001562  002020                          .WORD   260.+<260.*3.>   ;REMAINING WORD COUNT AFTER SECTOR 3
 1078  001564  002424                          .WORD   260.+<260.*4.>   ;REMAINING WORD COUNT AFTER SECTOR 4
 1079  001566  003030                          .WORD   260.+<260.*5.>   ;REMAINING WORD COUNT AFTER SECTOR 5
 1080  001570  003434                          .WORD   260.+<260.*6.>   ;REMAINING WORD COUNT AFTER SECTOR 6
 1081  001572  004040                          .WORD   260.+<260.*7.>   ;REMAINING WORD COUNT AFTER SECTOR 7
 1082  001574  004444                          .WORD   260.+<260.*8.>   ;REMAINING WORD COUNT AFTER SECTOR 8
 1083  001576  005050                          .WORD   260.+<260.*9.>   ;REMAINING WORD COUNT AFTER SECTOR 9
 1084  001600  005454                          .WORD   260.+<260.*10.>  ;REMAINING WORD COUNT AFTER SECTOR 10
 1085  001602  006060                          .WORD   260.+<260.*11.>  ;REMAINING WORD COUNT AFTER SECTOR 11
 1086  001604  006464                          .WORD   260.+<260.*12.>  ;REMAINING WORD COUNT AFTER SECTOR 12
 1087  001606  007070                          .WORD   260.+<260.*13.>  ;REMAINING WORD COUNT AFTER SECTOR 13
 1088  001610  007474                          .WORD   260.+<260.*14.>  ;REMAINING WORD COUNT AFTER SECTOR 14
 1089  001612  010100                          .WORD   260.+<260.*15.>  ;REMAINING WORD COUNT AFTER SECTOR 15
 1090  001614  010504                          .WORD   260.+<260.*16.>  ;REMAINING WORD COUNT AFTER SECTOR 16
 1091  001616  011110                          .WORD   260.+<260.*17.>  ;REMAINING WORD COUNT AFTER SECTOR 17
 1092  001620  011514                          .WORD   260.+<260.*18.>  ;REMAINING WORD COUNT AFTER SECTOR 18
 1093  001622  012120                          .WORD   260.+<260.*19.>  ;REMAINING WORD COUNT AFTER SECTOR 19
 1094  001624  012524                          .WORD   260.+<260.*20.>  ;REMAINING WORD COUNT AFTER SECTOR 20
 1095  001626  013130                          .WORD   260.+<260.*21.>  ;REMAINING WORD COUNT AFTER SECTOR 21
 1096
 1097                                  .EVEN
 1098
```

```
 1099                                   .SBTTL  ERROR POINTER TABLE
 1100
 1101                                   ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
 1102                                   ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
 1103                                   ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
 1104                                   ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
 1105                                   ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
 1106
 1107                                   ;*      EM              ;;POINTS TO THE ERROR MESSAGE
 1108                                   ;*      DH              ;;POINTS TO THE DATA HEADER
 1109                                   ;*      DT              ;;POINTS TO THE DATA
 1110                                   ;*      DF              ;;POINTS TO THE DATA FORMAT
 1111
 1112
 1113  001630                           $ERRTB:
 1114                                   ;ERROR 1
 1115
 1116  001630  022242                           EM1                     ;RH11 INTERRUPT OCCURRED (RPAS=0)
 1117  001632  023424                           DH1
 1118  001634  024676                           DT1
 1119  001636  025210                           DF1
 1120
 1121                                   ;ERROR 2
 1122
 1123  001640  022303                           EM2                     ;UNEXPECTED ATTENTION OCCURRED
 1124  001642  023431                           DH2
 1125  001644  024700                           DT2
 1126  001646  025214                           DF2
 1127
 1128                                   ;ERROR 3
 1129
 1130  001650  022341                           EM3                     ;MASSBUS PARITY ERROR (MCPE=1)
 1131  001652  023506                           DH3
 1132  001654  024714                           DT3
 1133  001656  025220                           DF3
 1134
 1135                                   ;ERROR 4
 1136
 1137  001660  022377                           EM4                     ;MASSBUS PARITY ERROR (PAR=1)
 1138  001662  023534                           DH4
 1139  001664  024722                           DT4
 1140  001666  025224                           DF4
 1141
 1142                                   ;ERROR 5
 1143
 1144  001670  022434                           EM5                     ;ADDRESS PLUG CHANGE BIT SET
 1145  001672  023431                           DH2
 1146  001674  024700                           DT2
 1147  001676  025214                           DF2
 1148
 1149                                   ;ERROR 6
 1150
 1151  001700  022470                           EM6                     ;RH11 DIDN'T RESPOND TO ADDRESSING
 1152  001702  023573                           DH6
 1153  001704  024732                           DT6
 1154  001706  025210                           DF1
```

```
1155
1156                                    ;ERROR 7
1157
1158  001710  000000                            0                       ;UNUSED
1159  001712  000000                            0
1160  001714  000000                            0
1161  001716  000000                            0
1162
1163                                    ;ERROR 10
1164
1165  001720  022532                            EM10                    ;DRIVE OFFLINE
1166  001722  023606                            DH10
1167  001724  024734                            DT10
1168  001726  025230                            DF10
1169
1170                                    ;ERROR 11
1171
1172  001730  022550                            EM11                    ;PERSISTENT DRIVE UNSAFE ERROR
1173  001732  023606                            DH10
1174  001734  024734                            DT10
1175  001736  025230                            DF10
1176
1177                                    ;ERROR 12
1178
1179  001740  022606                            EM12                    ;UNCORRECTABLE MASSBUS PARITY ERROR
1180  001742  023606                            DH10
1181  001744  024734                            DT10
1182  001746  025230                            DF10
1183
1184                                    ;ERROR 13
1185
1186  001750  022651                            EM13                    ;SOFTWARE TIMEOUT
1187  001752  023606                            DH10
1188  001754  024734                            DT10
1189  001756  025230                            DF10
1190
1191                                    ;ERROR 14
1192
1193  001760  022672                            EM14                    ;DRIVE UNSAFE ERROR
1194  001762  023606                            DH10
1195  001764  024734                            DT10
1196  001766  025230                            DF10
1197
1198                                    ;ERROR 15
1199
1200  001770  022715                            EM15                    ;CONTROLLER/DRIVE ERROR DURING WRITE
1201  001772  023606                            DH10
1202  001774  024734                            DT10
1203  001776  025230                            DF10
1204
1205                                    ;ERROR 16
1206
1207  002000  022761                            EM16                    ;CONTROLLER/DRIVE ERROR DURING WRITE CHECK
1208  002002  023606                            DH10
1209  002004  024734                            DT10
1210  002006  025230                            DF10
```

```
1211
1212                                    ;ERROR 17
1213
1214  002010  023033                            EM17                    ;RETRIES NOT SUCESSFUL - SECTOR NOT ACCEPTABLE
1215  002012  024056                            DH17
1216  002014  025010                            DT17
1217  002016  025244                            DF17
1218
1219                                    ;ERROR 20
1220
1221  002020  023111                            EM20                    ;DATA ERROR DURING WRITE CHECK
1222  002022  024125                            DH20
1223  002024  025022                            DT20
1224  002026  025250                            DF20
1225
1226                                    ;ERROR 21
1227
1228  002030  023147                            EM21                    ;CONTROLLER/DRIVE ERROR VERIFYING HEADERS
1229  002032  023606                            DH10
1230  002034  024734                            DT10
1231  002036  025230                            DF10
1232
1233                                    ;ERROR 22
1234
1235  002040  023220                            EM22                    ;'HCE' ERROR VERIFYING HEADERS
1236  002042  023606                            DH10
1237  002044  024734                            DT10
1238  002046  025230                            DF10
1239
1240                                    ;ERROR 23
1241
1242  002050  023267                            EM23                    ;CYLINDER FIELD IN HEADER IS NOT CORRECT
1243  002052  024424                            DH23
1244  002054  025104                            DT23
1245  002056  025270                            DF23
1246
1247                                    ;ERROR 24
1248
1249  002060  023337                            EM24                    ;WRITE CHECK ERROR
1250  002062  024522                            DH24
1251  002064  025122                            DT24
1252  002066  025274                            DF24
1253
1254                                    ;ERROR 25
1255
1256  002070  023361                            EM25                    ;HARDWARE ERROR DURING WRITE CHECK
1257  002072  024125                            DH20
1258  002074  025022                            DT20
1259  002076  025250                            DF20
1260
1261                                    ;;**********************************************************************
1262
1263                                    .SBTTL  MAIN PROGRAM
```

```
 1264
 1265                                   ;;*****************************************************************
 1266
 1267  002100  012737  177777  001302  BEGIN:  MOV     #-1,CHGADR      ;SET CHANGE 'RH11 BUS ADDRESS' INDICATOR
```

```
 1268  002106  000402                          BR      BEGIN2          ;START THE PROGRAM
 1269  002110  005037  001302          BEGIN1: CLR     CHGADR          ;CLEAR THE 'CHANGE RH11 ADDRESS' INDICATOR
 1270  002114  000005                  BEGIN2: RESET                   ;CLEAR THE BUS
 1271  002116  005037  001304                  CLR     EFLG            ;CLEAR ERROR DURING WRITE CHECK FLAG
 1272                                  .SBTTL  INITIALIZE THE COMMON TAGS
 1273                                  ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
 1274  002122  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
 1275  002126  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
 1276  002130  022706  001140                  CMP     #SWR,R6 ;;DONE?
 1277  002134  001374                          BNE     .-6             ;;LOOP BACK IF NO
 1278  002136  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
 1279                                  ;;INITIALIZE A FEW VECTORS
 1280  002142  012737  006766  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
 1281  002150  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
 1282  002156  012737  012144  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
 1283  002164  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
 1284  002172  005037  001160                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
 1285  002176  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
 1286                                  ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
 1287                                  ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
 1288  002204  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
 1289  002210  012737  002244  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
 1290  002216  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
 1291  002224  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
 1292  002232  022777  177777  176700          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
 1293  002240  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
 1294                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
 1295  002242  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
 1296  002244  012716  002252          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
 1297  002250  000002                          RTI
 1298  002252  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
 1299  002260  012737  000174  001142          MOV     #DISPREG,DISPLAY
 1300  002266  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
 1301
 1302  002272  000005                          RESET                   ;CLEAR WORLD
 1303  002274  012737  000240  000036          MOV     #PR5,@#TRAPVEC+2  ;CHANGE TRAP PRIORITY BACK TO 5
 1304  002302  012737  000240  000032          MOV     #PR5,@#EMTVEC+2 ;CHANGE EMT PRIORITY BACK TO 5
 1305  002310  012737  002110  001300          MOV     #BEGIN1,CNTLC   ;'CONTROL C' ADDRESS
```

INITIALIZE THE COMMON TAGS SEQ 0027

```
1306  002316  005227  177777                  INC     #-1             ;FIRST START ?
1307  002322  001010                          BNE     1$              ;BR IF NOT
1308  002324  104401  025324                  TYPE    ,TITLE          ;ADRS OF 'TITLE' MESSAGE
1309  002330  122737  000011  000041          CMPB    #11,41          ;LOADED FROM AN RP04/5/6 ?
1310  002336  001002                          BNE     1$              ;BR IF NOT
1311  002340  104401  025401                  TYPE    ,LOADRV         ;INSTRUCT OPERATOR TO REMOVE 'XXDP'
1312                                                                  ;PACK FROM DRIVE 0
1313  002344  004737  010334          1$:     JSR     PC,$TKINT       ;TURN ON THE KEYBOARD INTERRUPT
1314                                  .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1315  002350  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
1316  002354  001006                          BNE     67$             ;;BRANCH IF YES
1317  002356  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
1318  002364  001005                          BNE     68$             ;;BRANCH IF NO
1319  002366  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
1320  002370  000403                          BR      68$
1321  002372  112737  000001  001134  67$:     MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
1322  002400                          68$:
1323  002400  005227  177777                  INC     #-1             ;CHECK FOR FIRST START
1324  002404  001010                          BNE     SETVEC          ;BR IF NOT
1325  002406  004737  025626                  JSR     PC,BUSADR       ;CHECK THE RH11 ADDRESS
1326  002412  013737  001172  012376          MOV     $RPADR,RPADR    ;RH11 ADDRESS
1327  002420  013737  001174  012400          MOV     $RPVEC,RPVEC    ;RH11 VECTOR ADDRESS
1328
1329                                  ;DISPLAY DRIVE STATUS AND SET UP THE OTHER UNITS THE
1330                                  ;       PROGRAM WILL USE
1331
1332  002426  004737  006072          SETVEC: JSR     PC,ST.CLK       ;START THE CLOCK
1333  002432  004737  012414                  JSR     PC,RPINIT       ;INITIALIZE THE RP04/5/6 DRIVER
1334  002436  012737  177777  012336          MOV     #-1,SAVEFG      ;SET THE SAVE REGISTERS FLAG
1335  002444  005227  177777                  INC     #-1             ;SEE IF FIRST START
1336  002450  001404                          BEQ     11$             ;BR IF YES
1337  002452  032777  000004  176460          BIT     #SW02,@SWR      ;TYPEOUT THE DRIVE STATUS TABLE ?
1338  002460  001076                          BNE     10$             ;BR IF NOT
1339  002462  012737  000340  177776  11$:     MOV     #PR7,PS         ;SET PRIORITY TO 7
1340  002470  005004                          CLR     R4              ;DRIVE TABLE POINTER
1341  002472  104401  001167                  TYPE    ,$CRLF          ;CR-LF
1342  002476  104401  020770                  TYPE    ,SYSTAT         ;TYPE STATUS HEADING
1343  002502                          1$:
1344  002502  010446                          MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
1345                                                                  ;;TYPE DRIVE NUMBER
1346  002504  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1347  002506     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
1348  002507     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
1349  002510  104401  021033                  TYPE    ,LIN4SP         ;SPACES
1350  002514  105764  012250                  TSTB    DRVSTA(R4)      ;CHECK DRIVE'S STATUS
1351  002520  100416                          BMI     4$              ;BR IF UNSAFE
1352  002522  001020                          BNE     5$              ;BR IF ONLINE
1353  002524  105764  012260                  TSTB    DRVTYP(R4)      ;SEE IF OFFLINE OR NONEXISTENT
1354  002530  001404                          BEQ     2$              ;BR IF NONEXISTENT
1355  002532  100006                          BPL     3$              ;BR IF OFFLINE
1356  002534  104401  020703                  TYPE    ,NOTRP          ;DRIVE NOT AN RP04/5/6
1357  002540  000440                          BR      9$              ;CHECK NEXT DRIVE
1358  002542  104401  020724          2$:     TYPE    ,NOTPRS         ;DRIVE NOT PRESENT
1359  002546  000435                          BR      9$              ;CHECK NEXT DRIVE
1360  002550  104401  020662          3$:     TYPE    ,UNTOFF         ;DRIVE OFFLINE
1361  002554  000405                          BR      6$              ;PRINT DRIVE TYPE
```

GET VALUE FOR SOFTWARE SWITCH REGISTER SEQ 0028

```
1362  002556  104401  020741          4$:     TYPE    ,NOTSAF             ;DRIVE UNSAFE
1363  002562  000402                          BR      6$                  ;PRINT DRIVE TYPE
1364  002564  104401  020673          5$:     TYPE    ,UNTON              ;DRIVE ONLINE
1365  002570  104401  021035          6$:     TYPE    ,LINSP              ;SPACES
1366  002574  012737  020751  002640          MOV     #RP04B,8$           ;ADDRESS OF RP04 MESSAGE
1367  002602  132764  000001  012260          BITB    #BIT00,DRVTYP(R4)   ;RP04 ?
1368  002610  001012                          BNE     7$                  ;BR IF YES
1369  002612  012737  020756  002640          MOV     #RP05,8$            ;ADDRESS OF RP05 MESSAGE
1370  002620  132764  000002  012260          BITB    #BIT01,DRVTYP(R4)   ;RP05 ?
1371  002626  001003                          BNE     7$                  ;BR IF YES
1372  002630  012737  020763  002640          MOV     #RP06,8$            ;ADDRESS OF RP06 MESSAGE
1373  002636  104401                  7$:     TYPE                        ;TYPE THE DRIVE TYPE MESSAGE
1374  002640  000000                  8$:     .WORD   0                   ;MESSAGE ADDRESS HERE
1375  002642  104401  001167          9$:     TYPE    ,$CRLF              ;CR-LF
1376  002646  005204                          INC     R4                  ;INCREMENT DRIVE NUMBER/TABLE POINTER
1377  002650  020427  000010                  CMP     R4,#8.              ;FINISHED ?
1378  002654  001312                          BNE     1$                  ;BR IF NOT
1379  002656  104401  001167          10$:    TYPE    ,$CRLF              ;CR-LF
1380
1381
1382                                  ;SEE WHICH MODE THE PROGRAM IS TO BE RUN IN.
1383                                  ;MODES ARE: 'FORMAT & VERIFY' OR 'CHECK FORMAT'
1384
1385  002662  005037  001220          M1:     CLR     MODE                ;SET MODE TO 'CHECK FORMAT'
1386  002666  104401  021251                  TYPE    ,MMODE              ;TYPE 'PROGRAM MODE'
1387  002672  104411                          RDLIN                       ;READ THE KEYBOARD
1388  002674  012601                          MOV     (SP)+,R1            ;GET ADDRESS OF INPUT
1389  002676  105711                          TSTB    (R1)                ;'CR' ENTERED (DEFAULT)
1390  002700  001414                          BEQ     2$                  ;BR IF YES
1391  002702  122711  000103                  CMPB    #'C,(R1)            ;'CHECK' ?
1392  002706  001406                          BEQ     1$                  ;BR IF YES
1393  002710  122711  000106                  CMPB    #'F,(R1)            ;'FORMAT' ?
1394  002714  001411                          BEQ     3$                  ;BR IF YES
1395  002716  104401  001166                  TYPE    ,$QUES              ;NO CORRECT ENTRY
1396  002722  000757                          BR      M1                  ;TRY AGAIN
1397  002724  104401  021324          1$:     TYPE    ,MHECK              ;TYPE REST OF 'CHECK'
1398  002730  000410                          BR      M1A                 ;GET STARTING ADDRESS
1399  002732  104401  021303          2$:     TYPE    ,MFORMT             ;TYPE DEFAULT MESSAGE
1400  002736  000402                          BR      4$                  ;SET UP MODE
1401  002740  104401  021337          3$:     TYPE    ,MORMAT             ;TYPE REST OF 'FORMAT'
1402  002744  012737  177777  001220  4$:     MOV     #-1,MODE            ;SET MODE TO 'FORMAT & VERIFY'
1403
1404                                  ;FIND OUT IF FORMAT IS TO BE IN 20 OR 22 SECTOR MODE
1405
1406  002752  104401  021357          M1A:    TYPE    ,MSIZE              ;TYPE FORMAT MODE REQUEST
1407  002756  104411                          RDLIN                       ;READ THE KEYBOARD
1408  002760  012601                          MOV     (SP)+,R1            ;ADDRESS OF ENTRY
1409  002762  122711  000131                  CMPB    #'Y,(R1)            ;IS ENTRY A 'Y' ?
1410  002766  001410                          BEQ     1$                  ;BR IF IT IS
1411  002770  122711  000116                  CMPB    #'N,(R1)            ;IS ENTRY A 'N' ?
1412  002774  001424                          BEQ     2$                  ;BR IF IT IS
1413  002776  105711                          TSTB    (R1)                ;DEFAULT MODE ?
1414  003000  001403                          BEQ     1$                  ;BR IF IT IS
1415  003002  104401  001166                  TYPE    ,$QUES              ;TYPE '?'
1416  003006  000761                          BR      M1A                 ;TRY AGAIN
1417  003010  104401  021430          1$:     TYPE    ,MSEC22             ;TYPEOUT MODE SELECTED
```

GET VALUE FOR SOFTWARE SWITCH REGISTER SEQ 0029

```
1418  003014  012737  013130  001256          MOV     #<260.*22.>,WC  ;TRACK SIZE IN 22 SECTOR MODE
1419  003022  012737  164650  001260          MOV     #-<260.*22.>,MWC  ;2'S COMPLEMENT WORD COUNT
1420  003030  012737  177777  001264          MOV     #-1,SEC20       ;22 SECTOR INDICATOR
1421  003036  012737  000025  001266          MOV     #21.,MAXSEC     ;MAX SECTOR ADDRESS IN 22 SECTOR MODE
1422  003044  000415                          BR      M1B             ;CONTINUE
1423  003046  104401  021507          2$:     TYPE    ,MSEC20         ;TYPE OUT 20 SECTOR MODE SELECTED
1424  003052  012737  012120  001256          MOV     #<260.*20.>,WC  ;TRACK SIZE IN 20 SECTOR MODE
1425  003060  012737  165660  001260          MOV     #-<260.*20.>,MWC  ;2'S COMPLEMENT WORD COUNT
1426  003066  005037  001264                  CLR     SEC20           ;20 SECTOR INDICATOR
1427  003072  012737  000023  001266          MOV     #19.,MAXSEC     ;MAX SECTOR ADDRESS IN 20 SECTOR MODE
1428  003100  005037  001230          M1B:    CLR     BEGTRK          ;CLEAR STARTING TRACK ADDRESS
1429  003104  005037  001224                  CLR     BEGCYL          ;CLEAR BEGINNING CYLINDER ADDRESS
1430  003110  012737  000022  001226          MOV     #18.,ENDTRK     ;SETUP END TRACK ADDRESS
1431  003116  012737  000002  001240          MOV     #2,PATSEL       ;SETUP FOR WORST CASE PATTERN
1432  003124  112737  177777  001356          MOVB    #-1,FMTDPB      ;SETUP INVALID DRIVE NUMBER
1433  003132  000400                          BR      M0              ;BRANCH TO START
1434
1435                                  ;GO FIND OUT WHAT DRIVE
1436
1437  003134  012737  006246  001300  M0:     MOV     #0ENTER,CNTLC   ;CONTROL C ABORT ENTRANCE
1438  003142  005037  001112                  CLR     $ERTTL          ;CLEAR THE ERROR ACCUMULATOR
1439  003146  004737  010334                  JSR     PC,$TKINT       ;INITIALIZE THE TTY KEYBOARD
1440  003152  104401  021007                  TYPE    ,MUNIT          ;ASK FOR DRIVE NUMBER
1441  003156  104411                          RDLIN                   ;READ THE KEYBOARD
1442  003160  012601                          MOV     (SP)+,R1        ;ADDRESS OF ENTRY
1443  003162  004537  006556                  JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
1444  003166  003214                          3$                      ;ILLEGAL CHARACTER
1445  003170  003202                          2$                      ;CARRIAGE RETURN
1446  003172  003214                          3$                      ;''
1447  003174  003202                          2$                      ;,,
1448  003176  003222                          4$                      ;DIGIT 0-7
1449  003200  003214                          3$                      ;DIGIT 8-9
1450  003202  005037  001214          2$:     CLR     DRIVE           ;SELECT DRIVE ZERO
1451  003206  104401  021031                  TYPE    ,MDRVD          ;GO TYPE DEFAULT DRIVE NUMBER
1452  003212  000413                          BR      5$              ;GO SEE IF DRIVE ZERO IS THERE
1453  003214  104401  001166          3$:     TYPE    ,$QUES          ;TYPE '?'
1454  003220  000745                          BR      M0              ;ASK FOR DRIVE NUMBER AGAIN
1455  003222  010237  001214          4$:     MOV     R2,DRIVE        ;SAVE DRIVE NUMBER
1456  003226  005702                          TST     R2              ;SEE IF DRIVE 0
1457  003230  001004                          BNE     5$              ;BR IF NOT
1458  003232  122737  000011  000041          CMPB    #11,41          ;PROGRAM LOADED FROM AN RP04/5/6 ?
1459  003240  001431                          BEQ     7$              ;BR IF IT WAS, CAN'T FORMAT THE DRIVE
1460  003242  004737  006072          5$:     JSR     PC,ST.CLK       ;START THE CLOCK
1461  003246  004737  012414                  JSR     PC,RPINIT       ;GO SEE WHAT DRIVES ARE AVAILABLE
1462  003252  012737  177777  012336          MOV     #-1,SAVEFG      ;SAVE THE REGISTERS
1463  003260  012737  177777  012340          MOV     #-1,SEEKFG      ;SET 'NO OPTIMIZATION' FLAG
1464  003266  005037  177776                  CLR     PS              ;SET PRIORITY BACK TO ZERO
1465  003272  105762  012250                  TSTB    DRVSTA(R2)      ;LOOK AT DRIVE STATUS
1466  003276  003015                          BGT     8$              ;BRANCH IF ONLINE
1467  003300  001403                          BEQ     6$              ;BR IF DRIVE NOT AVAILABLE
1468  003302  104401  021217                  TYPE    ,MUSDR          ;'DRIVE UNSAFE'
1469  003306  000712                          BR      M0              ;GO GET DRIVE NUMBER AGAIN
1470  003310  105762  012260          6$:     TSTB    DRVTYP(R2)      ;A DRIVE PRESENT?
1471  003314  001003                          BNE     7$              ;BR IF SO
1472  003316  104401  021112                  TYPE    ,MDRNP          ;TYPE 'DRIVE NOT PRESENT'
1473  003322  000704                          BR      M0              ;GO GET DRIVE NUMBER AGAIN
```

```
                        GET VALUE FOR SOFTWARE SWITCH REGISTER                                  SEQ 0030

 1474  003324  104401  021137          7$:     TYPE    ,MER11          ;'DRIVE NOT AVAILABLE'
 1475  003330  000701                  BR      M0              ;GO GET DRIVE NUMBER AGAIN
 1476  003332  123737  001214  001356  8$:     CMPB    DRIVE,FMTDPB    ;SAME DRIVE AS LAST TIME ?
 1477  003340  001430                  BEQ     M2              ;BR IF IT IS
 1478  003342  113737  001214  001356          MOVB    DRIVE,FMTDPB    ;SETUP DRIVE ADDRESS
 1479  003350  012737  000632  001222          MOV     #410.,ENDCYL    ;SETUP FOR RP04/5
 1480  003356  132762  000003  012260          BITB    #BIT00!BIT01,DRVTYP(R2)  ;SEE IF DRIVE RP04/5
 1481  003364  001003                  BNE     9$              ;BR IF EITHER
 1482  003366  012737  001456  001222          MOV     #814.,ENDCYL    ;SETUP ENDING CYLINDER FOR RP06
 1483  003374  005037  001230          9$:     CLR     BEGTRK          ;CLEAR STARTING TRACK ADDRESS
 1484  003400  005037  001224                  CLR     BEGCYL          ;CLEAR STARTING CYLINDER ADDRESS
 1485  003404  013737  001222  001400          MOV     ENDCYL,TABLE+2  ;ENTRY LIMIT FOR BEGINNING CYLINDER
 1486  003412  013737  001222  001414          MOV     ENDCYL,TABLE+16 ;ENTRY LIMIT FOR END CYLINDER
 1487  003420  000400                  BR      M2              ;GET ADDRESS LIMITS FROM THE OPERATOR
 1488
 1489                                  ;GET ADDRESS LIMITS
 1490
 1491  003422  104401  021040          M2:     TYPE    ,ENTADR         ;'ENTER ADDRESS LIMITS'
 1492  003426  004737  006366                  JSR     PC,PARENT       ;GET THE ADDRESS LIMITS
 1493  003432  023737  001222  001224          CMP     ENDCYL,BEGCYL   ;SEE IF ENDING CYLINDER EQ TO GT THAN BEGINNING
 1494  003440  101010                  BHI     M4              ;BR IF HIGHER
 1495  003442  103404                  BLO     3$              ;BR IF LESS
 1496  003444  023737  001226  001230  2$:     CMP     ENDTRK,BEGTRK   ;SEE IF ENDING TRACK EQ OR GT THAN BEGINNING
 1497  003452  103003                  BHIS    M4              ;BR IF YES
 1498  003454  104401  021606          3$:     TYPE    ,MADRER         ;INVALID ADDRESS ENTERED
 1499  003460  000760                  BR      M2              ;TRY AGAIN
 1500
 1501                                  ;GO GET DATA PATTERN FOR FORMAT
 1502
 1503  003462  104401  021710          M4:     TYPE    ,MSELP          ;GO TYPE 'SELECT PATTERN'
 1504  003466  104411                  RDLIN                   ;READ THE KEYBOARD
 1505  003470  012601                  MOV     (SP)+,R1        ;ENTRY ADDRESS
 1506  003472  004537  006556          JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
 1507  003476  003532                  3$                      ;ILLEGAL CHARACTER
 1508  003500  003512                  1$                      ;CARRIAGE RETURN
 1509  003502  003532                  3$                      ;"."
 1510  003504  003512                  1$                      ;","
 1511  003506  003524                  2$                      ;DIGIT 0-7
 1512  003510  003532                  3$                      ;DIGIT 8-9
 1513  003512  104401  022010          1$:     TYPE    ,MPATD          ;TYPE DEFAULT PATTERN
 1514  003516  012702  000002                  MOV     #2,R2           ;WORST CASE PATTERN
 1515  003522  000406                  BR      4$              ;GO SAVE PATTERN
 1516  003524  020227  000002          2$:     CMP     R2,#2           ;IS # LARGER THAN 2
 1517  003530  101403                  BLOS    4$              ;BRANCH IF NOT
 1518  003532  104401  001166          3$:     TYPE    ,$QUES          ;TYPE '?'
 1519  003536  000751                  BR      M4              ;RETYPE LINE
 1520  003540  010237  001240          4$:     MOV     R2,PATSEL       ;SAVE PATTERN SELECTED
 1521
 1522                                  ;GO TYPE 'STARTING FORMAT ON DRIVE N'
 1523
 1524  003544  012737  006266  001300  M5:     MOV     #TYPADR,CNTLC   ;CHANGE ^C ENTRANCE
 1525  003552  005737  001220                  TST     MODE            ;'FORMAT' OR 'CHECK' MODE ?
 1526  003556  001403                  BEQ     1$              ;BR IF 'CHECK' MODE
 1527  003560  104401  022023                  TYPE    ,MSFOU          ;TYPE 'STARTING FORMAT ON DRIVE N'
 1528  003564  000402                  BR      2$
 1529  003566  104401  022060          1$:     TYPE    ,MSCHK          ;TYPE 'STARTING CHECK ON DRIVE N'
```

```
1530  003572                                2$:
1531  003572  013746  001214                      MOV     DRIVE,-(SP)     ;;SAVE DRIVE FOR TYPEOUT
1532                                                                        ;;TYPE DRIVE NUMBER
1533  003576  104403                              TYPOS                   ;;GO TYPE--OCTAL ASCII
1534  003600     001                              .BYTE   1               ;;TYPE 1 DIGIT(S)
1535  003601     000                              .BYTE   0               ;;SUPPRESS LEADING ZEROS
1536  003602  104401  001167                      TYPE    ,$CRLF          ;CR-LF
1537
1538                                        ;SETUP TOTAL TRACK COUNT FOR FORMAT
1539
1540  003606  023737  001222  001224  CKADRS: CMP     ENDCYL,BEGCYL   ;STARTING AND ENDING CYLINDERS THE SAME ?
1541  003614  001011                          BNE     1$              ;BRANCH IF BEGCYL NOT EQUAL TO ENDCYL
1542  003616  013737  001226  001232          MOV     ENDTRK,TTRKS    ;END TRACK ADDRESS
1543  003624  163737  001230  001232          SUB     BEGTRK,TTRKS    ;SUBTRACT THE STARTING TRACK ADDRESS
1544  003632  005237  001232                  INC     TTRKS           ;MAKE THE NUMBER OF TRACKS INCLUSIVE
1545  003636  000427                          BR      SETPAT          ;SETUP THE DATA PATTERN
1546  003640  013702  001222          1$:     MOV     ENDCYL,R2       ;ENDING CYLINDER
1547  003644  163702  001224                  SUB     BEGCYL,R2       ;SUBTRACT THE STARTING CYLINDER
1548  003650  013737  001226  001232          MOV     ENDTRK,TTRKS    ;CALCULATE THE RESIDUAL TRACKS
1549  003656  163737  001230  001232          SUB     BEGTRK,TTRKS    ;SUBTRACT THE STARTING TRACK
1550  003664  100004                          BPL     2$              ;BR IF ENDING TRACK GREATER
1551  003666  062737  000023  001232          ADD     #19.,TTRKS      ;CORRECT THE VALUE
1552  003674  005302                          DEC     R2              ;COUNT THE PARTIAL TRACK
1553  003676  005237  001232          2$:     INC     TTRKS           ;MAKE THE NUMBER INCLUSIVE
1554  003702  005302                  3$:     DEC     R2              ;DECREMENT THE CYLINDER COUNT
1555  003704  100404                          BMI     SETPAT          ;BR IF DONE
1556  003706  062737  000023  001232          ADD     #19.,TTRKS      ;ADD CYLINDER WORTH OF TRACKS
1557  003714  000772                          BR      3$              ;CONTINUE
1558
1559                                        ;THIS CODE SETS UP THE SELECTED DATA PATTERN TO BE WRITTEN ON EACH SECTOR IMAGE
1560
1561  003716  005037  001242          SETPAT: CLR     PATA            ;CLEAR DATA PATTERN A
1562  003722  005037  001244                  CLR     PATB            ;CLEAR DATA PATTERN B
1563  003726  005737  001240                  TST     PATSEL          ;SEE IF PATTERN OF ONES
1564  003732  001416                          BEQ     1$              ;BR IF SO
1565  003734  005137  001242                  COM     PATA            ;SET PATA TO ONES
1566  003740  005137  001244                  COM     PATB            ;SET PATB TO ONES
1567  003744  022737  000001  001240          CMP     #1,PATSEL       ;SEE IF PATTERN OF ZEROS
1568  003752  001406                          BEQ     1$              ;BRANCH IF SO
1569  003754  012737  165555  001242          MOV     #165555,PATA    ;SET UP WORST CASE
1570  003762  012737  133333  001244          MOV     #133333,PATB    ;SET UP WORST CASE
1571  003770  013703  001256          1$:     MOV     WC,R3           ;SET UP COUNTER
1572  003774  012700  025324                  MOV     #BUFP,R0        ;SET UP MEMORY POINTER
1573  004000  013701  001242                  MOV     PATA,R1         ;SET UP PATTERN IN R1
1574  004004  013702  001244                  MOV     PATB,R2         ;SET UP PATTERN IN R2
1575  004010  010120                  2$:     MOV     R1,(R0)+        ;MOV 1ST PAT INTO MEM
1576  004012  010220                          MOV     R2,(R0)+        ;MOV 2ND PAT INTO MEM
1577  004014  005303                          DEC     R3              ;KEEP COUNT
1578  004016  005303                          DEC     R3              ;KEEP COUNT
1579  004020  001373                          BNE     2$              ;DO IT AGAIN IF R3 NOT 0
1580
1581                                        ;THIS CODE SETS UP FOR THE ACTUAL FORMAT
1582
1583  004022  004737  005532          WRTRK:  JSR     PC,SETTBL       ;GO SET UP DRIVER TABLE
1584  004026  004737  005732                  JSR     PC,SETHDR       ;GO INITIALIZE HEADERS IN THE SECTOR BUFFER IN CORE
1585  004032  112737  000020  001357          MOVB    #20,FMTDPB+1    ;LOAD FMT22 BIT
```

```
1586  004040  005737  001264                  TST     SEC20           ;18 BIT MODE ?
1587  004044  001002                          BNE     1$              ;BR IF NOT
1588  004046  105037  001357                  CLRB    FMTDPB+1        ;CLEAR THE FMT22 BIT
1589  004052  112737  000143  001360  1$:     MOVB    #SETFMT,FMTDPB+2  ;'LOAD FORMAT' COMMAND
1590  004060  004037  013164          2$:     JSR     R0,RP04         ;START THE COMMAND
1591  004064  001356                          FMTDPB                  ;DPB ADDRESS
1592  004066  000774                          BR      2$              ;QUEUE FULL RETURN
1593  004070  005737  001374          3$:     TST     FMTDPB+16       ;LOOK FOR DONE
1594  004074  001775                          BEQ     3$              ;LOOP UNTIL FINISHED
1595  004076  005037  001216          WRTRK1: CLR     SOFSW           ;CLEAR ERROR COUNTER
1596  004102  005037  001250                  CLR     RETRY           ;ZERO THE RETRY COUNTER
1597  004106  005037  001304                  CLR     EFLG            ;CLEAR ERROR DURING WRITE CHECK FLAG
1598  004112  105037  001366                  CLRB    FMTDPB+10       ;RESTORE SECTOR
1599  004116  013737  001260  001362          MOV     MWC,FMTDPB+4    ;RESTORE WC
1600  004124  012737  025324  001364          MOV     #BUFP,FMTDPB+6  ;RESTORE BA
1601  004132  005737  001220                  TST     MODE            ;'FORMAT' OR 'CHECK' MODE ?
1602  004136  001450                          BEQ     CKTRK           ;BR IF 'CHECK' MODE
1603  004140  112737  000163  001360  WRTRK2: MOVB    #WRTHD,FMTDPB+2  ;SET WRITE HEADER & DATA COMMAND IN TBL
1604  004146  012737  004146  001110          MOV     #.,$LPERR       ;SETUP LOOP ON ERROR ADDRESS
1605  004154  012706  001100                  MOV     #STACK,SP       ;LOAD STACK POINTER
1606  004160  004037  013164          1$:     JSR     R0,RP04         ;GO FORMAT A TRACK
1607  004164  001356                          FMTDPB                  ;ADRS OF PARAMETERS - TBL
1608  004166  000774                          BR      1$              ;WAIT FOR QUEUE IF FULL
1609  004170  005737  001374          2$:     TST     FMTDPB+16       ;WAIT FOR COMMAND TO COMPLETE
1610  004174  001775                          BEQ     2$              ;BRANCH IF NOT DONE
1611  004176  100024                          BPL     4$              ;BRANCH IF NO ERROR
1612  004200  012737  004226  001160          MOV     #3$,$ESCAPE     ;;ESCAPE TO 3$ ON ERROR
1613  004206  004737  005364                  JSR     PC,ERINDX       ;SEE WHICH ERROR
1614  004212  104010                          ERROR   10              ;DRIVE OFFLINE
1615  004214  104011                          ERROR   11              ;PERSISTENT DRIVE UNSAFE ERROR
1616  004216  104012                          ERROR   12              ;UNCORRECTABLE MASSBUS PARITY ERROR
1617  004220  104013                          ERROR   13              ;SOFTWARE TIMEOUT
1618  004222  104014                          ERROR   14              ;DRIVE UNSAFE ERROR
1619  004224  104015                          ERROR   15              ;DRIVE/CONTROLLER ERROR DURING WRITE
1620  004226  004737  005464          3$:     JSR     PC,LOP.CK       ;LOOP ON THE ERROR ?
1621  004232  022737  000003  001250          CMP     #3,RETRY        ;ERROR RETRY LIMIT ?
1622  004240  001403                          BEQ     4$              ;BR IF REACHED
1623  004242  005237  001250                  INC     RETRY           ;COUNT THE ERROR
1624  004246  000744                          BR      1$              ;TRY AGAIN
1625  004250  005037  001160          4$:     CLR     $ESCAPE         ;CLEAR ERROR ESCAPE ADDRESS
1626  004254  005037  001250                  CLR     RETRY           ;CLEAR THE RETRY COUNTER
1627
1628                                  ;CHECK THE TRACK JUST WRITTEN
1629
1630  004260  112737  000153  001360  CKTRK:  MOVB    #WCKHD,FMTDPB+2 ;SET WRITE CHECK HEADER & DATA COMMAND IN TBL
1631  004266  012737  004266  001110          MOV     #.,$LPERR       ;SETUP LOOP ON ERROR ADDRESS
1632  004274  012706  001100                  MOV     #STACK,SP       ;LOAD STACK POINTER
1633  004300  004037  013164          1$:     JSR     R0,RP04         ;GO CHECK THE TRACK JUST FORMATTED
1634  004304  001356                          FMTDPB                  ;ADRS OF PARAMETERS - TBL
1635  004306  000774                          BR      1$              ;WAIT FOR QUEUE IF FULL
1636  004310  005737  001374          2$:     TST     FMTDPB+16       ;WAIT FOR COMMAND TO COMPLETE
1637  004314  001775                          BEQ     2$              ;BRANCH IF NOT DONE
1638  004316  100147                          BPL     8$              ;BRANCH IF NO ERROR
1639  004320  113737  001314  001252          MOVB    RP.REG+RPDA,SAVSEC  ;GET THE SECTOR ADDRESS
1640  004326  001005                          BNE     3$              ;BRANCH IF NOT SECTOR 0
1641  004330  013737  001266  001252          MOV     MAXSEC ,AVSEC   ;RESTORE TO LAST SECTOR +1
```

```
1642  004336  005237  001252                  INC     SAVSEC          ;INCREMENT TO LAST SECTOR +1
1643  004342  005337  001252          3$:     DEC     SAVSEC          ;ADJUST SECTOR TO THE ONE THAT FAILED
1644  004346  012737  004712  001160          MOV     #10$,$ESCAPE    ;;ESCAPE TO 10$ ON ERROR
1645  004354  004737  005364                  JSR     PC,ERINDX       ;SEE WHICH ERROR
1646  004360  104010                          ERROR   10              ;DRIVE OFFLINE
1647  004362  104011                          ERROR   11              ;PERSISTENT DRIVE UNSAFE ERROR
1648  004364  104012                          ERROR   12              ;UNCORRECTABLE MASSBUS PARITY ERROR
1649  004366  104013                          ERROR   13              ;SOFTWARE TIMEOUT
1650  004370  104014                          ERROR   14              ;DRIVE UNSAFE ERROR
1651  004372  032737  040000  001316          BIT     #WCE,RP.REG+RPCS2  ;WRITE CHECK ERROR ?
1652  004400  001005                          BNE     4$              ;BR IF SET
1653  004402  033727  001322                  BIT     RP.REG+RPER1,(PC)+  ;CHECK FOR DATA ERRORS
1654  004406  130620                          .WORD   DCK!OPI!DTE!HCRC!HCE!FER  ;DATA ERROR BITS
1655  004410  001001                          BNE     4$              ;BR IF SET
1656  004412  104016                          ERROR   16              ;CONTROLLER/DRIVE ERROR DURING WRITE CHECK
1657  004414  005737  001220          4$:     TST     MODE            ;FORMAT OR CHECK MODE ?
1658  004420  001416                          BEQ     5$              ;BR IF CHECK
1659  004422  005737  001216                  TST     SOFSW           ;RETRYING THE SECTOR ?
1660  004426  001413                          BEQ     5$              ;BR IF NOT
1661  004430  012737  004736  001160          MOV     #11$,$ESCAPE    ;;ESCAPE TO 11$ ON ERROR
1662  004436  005737  001304                  TST     EFLG            ;ERROR DURING WRITE CHECK REPORTED ?
1663  004442  001402                          BEQ     21$             ;NO,  INCREMENT ERROR COUNT
1664  004444  005337  001112                  DEC     $ERTTL          ;YES, DECREMENT REPORTED ERROR COUNT
1665  004450  005037  001304          21$:    CLR     EFLG            ;CLEAR ERROR DURING WRITE CHECK FLAG
1666  004454  104017                          ERROR   17              ;SECTOR NOT ACCEPTABLE
1667  004456  005037  001160          5$:     CLR     $ESCAPE         ;CLEAR THE ERROR ESCAPE ADDRESS
1668  004462  005737  001322                  TST     RP.REG+RPER1    ;DATA CHECK ERROR ?
1669  004466  100011                          BPL     23$             ;NO, CHECK OTHER ERRORS
1670  004470  032777  000200  174442          BIT     #BIT7,@SWR      ;INHIBIT SOFT ERROR REPORTS ?
1671  004476  001426                          BEQ     7$              ;YES, SKIP ERROR REPORT
1672  004500  012737  000001  001304          MOV     #1,EFLG         ;SET ERROR DURING WRITE CHECK FLAG
1673  004506  104020                          ERROR   20              ;DATA ERROR DURING WRITE CHECK
1674  004510  000421                          BR      7$              ;CONTINUE
1675  004512  032737  040000  001316  23$:    BIT     #WCE,RP.REG+RPCS2       ;WRITE CHECK ERROR ?
1676  004520  001411                          BEQ     24$             ;NO, MUST BE HARDWARE ERROR
1677  004522  032777  000200  174410          BIT     #BIT7,@SWR      ;INHIBIT SOFT ERROR REPORTS ?
1678  004530  001411                          BEQ     7$              ;YES, SKIP ERROR REPORT
1679  004532  012737  000001  001304          MOV     #1,EFLG         ;SET ERROR DURING WRITE CHECK FLAG
1680  004540  104024                          ERROR   24              ;WRITE CHECK ERROR
1681  004542  000404                          BR      7$              ;CONTINUE
1682  004544  012737  000001  001304  24$:    MOV     #1,EFLG         ;SET ERROR DURING WRITE CHECK FLAG
1683  004552  104025                          ERROR   25              ;HARDWARE ERROR DURING WRITE CHECK
1684  004554  013701  001252          7$:     MOV     SAVSEC,R1       ;FAILING SECTOR ADDRESS
1685  004560  113737  001252  001366          MOVB    SAVSEC,FMTDPB+10  ;SECTOR ADDRESS
1686  004566  006301                          ASL     R1              ;SETUP INDEX TO ADDRESS WORDS
1687  004570  016137  001500  001364          MOV     ADRTBL(R1),FMTDPB+6  ;BUFFER ADDRESS FOR FAILING SECTOR
1688  004576  013746  001260                  MOV     MWC,-(SP)       ;GET MAXIMUM WORD COUNT VALUE (2'S COMP)
1689  004602  066116  001554                  ADD     WCTBL(R1),(SP)  ;ADD WORD COUNT THROUGH FAILING SECTOR
1690  004606  012637  001254                  MOV     (SP)+,SAVWC     ;STORE WORD COUNT FOR REMAINDER OF TRACK
1691  004612  005737  001220                  TST     MODE            ;FORMAT OR CHECK MODE ?
1692  004616  001421                          BEQ     9$              ;BR IF CHECK
1693  004620  012737  177374  001362          MOV     #-260.,FMTDPB+4 ;WORD COUNT FOR 1 SECTOR
1694  004626  005237  001216                  INC     SOFSW           ;INDICATE THAT A RETRY IS IN PROGRESS
1695  004632  000137  004140                  JMP     WRTRK2          ;REFORMAT ERROR SECTOR
1696  004636  005737  001216          8$:     TST     SOFSW           ;RETRY IN PROGRESS ?
1697  004642  001435                          BEQ     11$             ;BR IF NOT
```

```
 1698  004644  022737  000002  001216          CMP     #2,SOFSW        ;SEE IF LAST RETRY
 1699  004652  001403                          BEQ     9$              ;BR IF IT IS
 1700  004654  005237  001216                  INC     SOFSW           ;INCREMENT RETRY COUNT
 1701  004660  000607                          BR      1$              ;READ AGAIN
 1702  004662  123737  001266  001366  9$:     CMPB    MAXSEC,FMTDPB+10  ;SEE IF LAST SECTOR ON THE TRACK
 1703  004670  001422                          BEQ     11$             ;BR IF IT IS
 1704  004672  004737  005704                  JSR     PC,SCAWC        ;SETUP TO CHECK REMAINING SECTORS ON THE TRACK
 1705  004676  005037  001216                  CLR     SOFSW           ;CLEAR RETRY COUNTER
 1706  004702  005037  001304                  CLR     EFLG            ;CLEAR ERROR DURING WRITE CHECK FLAG
 1707  004706  000137  004300                  JMP     1$              ;FINISH CHECKING THE TRACK
 1708  004712  004737  005464          10$:    JSR     PC,LOP.CK       ;CHECK FOR LOOP ON ERROR
 1709  004716  022737  000003  001250          CMP     #3,RETRY        ;ERROR RETRY REACHED ?
 1710  004724  001404                          BEQ     11$             ;BR IF IT IS
 1711  004726  005237  001250                  INC     RETRY           ;COUNT THE RETRY
 1712  004732  000137  004300                  JMP     1$              ;DO THE WRITE CHECK AGAIN
 1713  004736  005037  001160          11$:    CLR     $ESCAPE         ;CLEAR THE ERROR RETURN ESCAPE ADDRESS
 1714  004742  005037  001250                  CLR     RETRY           ;CLEAR THE RETRY COUNTER
 1715  004746  005037  001304                  CLR     EFLG            ;CLEAR ERROR DURING WRITE CHECK FLAG
 1716  004752  032777  000002  174160          BIT     #BIT1,@SWR      ;LOOP ON CURRENT TRACK ?
 1717  004760  001402                          BEQ     12$             ;BR IF NOT
 1718  004762  000137  004076                  JMP     WRTRK1          ;START AGAIN ON THE SAME TRACK
 1719  004766  004737  005622          12$:    JSR     PC,TRKTST       ;GET UPDATED TRACK AND CYLINDER ADDRESSES
 1720  004772  000402                          BR      HDREAD          ;RETURN HERE IF DONE - DO QUICK CHECK OF DISK
 1721  004774  000137  004076                  JMP     WRTRK1          ;CONTINUE WITH THE FORMAT
 1722
 1723                                  ;THIS CODE MAKES SURE EACH CYLINDER FORMATTED CONTAINS THE
 1724                                  ;PROPER CYLINDER ADDRESS. THE PROGRAM IS LOOKING FOR
 1725                                  ;POSSIBLE POSITIONER ERRORS THAT MAY HAVE OCCURRED DURING THE FORMAT.
 1726
 1727  005000  005037  001262          HDREAD: CLR     HEDERR          ;CLEAR HEADER CHECK ERROR INDICATOR
 1728  005004  004737  005532                  JSR     PC,SETTBL       ;GO SET UP DRIVER TABLE
 1729  005010  004737  005732                  JSR     PC,SETHDR       ;GO SET UP HEADERS IN CORE
 1730  005014  013737  001224  001246          MOV     BEGCYL,CYLCK    ;USE 'BEGCYL' FOR FORMAT VERIFICATION
 1731  005022  012737  177774  001362          MOV     #-4,FMTDPB+4    ;SET UP WORD COUNT
 1732  005030  012737  025314  001364          MOV     #RBUF,FMTDPB+6  ;SET UP BUFFER ADRS
 1733  005036  005037  001366                  CLR     FMTDPB+10       ;CLEAR THE SECTOR & TRACK ADDRESS FIELD
 1734  005042  013737  001224  001370          MOV     BEGCYL,FMTDPB+12  ;SETUP THE CYLINDER FIELD
 1735  005050  112737  000173  001360          MOVB    #RDHD,FMTDPB+2  ;SET UP READ HEADER & DATA COMMAND
 1736  005056  012737  005056  001110          MOV     #.,$LPERR       ;SETUP LOOP ON ERROR ADDRESS
 1737  005064  012706  001100                  MOV     #STACK,SP       ;LOAD STACK POINTER
 1738  005070  004037  013164          1$:     JSR     R0,RP04         ;GO READ HEADER
```

```
 1739  005074  001356                          FMTDPB                  ;ADRS OF PARAMETER TBL
 1740  005076  000774                          BR      1$              ;WAIT IF QUEUE IS FULL
 1741  005100  005737  001374          2$:     TST     FMTDPB+16       ;WAIT FOR COMMAND TO COMPLETE
 1742  005104  001775                          BEQ     2$              ;BRANCH IF NOT DONE
 1743  005106  100024                          BPL     4$              ;BR IF NOT ERROR
 1744  005110  012737  005172  001160          MOV     #5$,$ESCAPE     ;;ESCAPE TO 5$ ON ERROR
 1745  005116  004737  005364                  JSR     PC,ERINDX       ;SEE WHICH ERROR
 1746  005122  104010                          ERROR   10              ;DRIVE OFFLINE
 1747  005124  104011                          ERROR   11              ;PERSISTENT DRIVE UNSAFE ERROR
 1748  005126  104012                          ERROR   12              ;UNCORRECTABLE MASSBUS PARITY ERROR
 1749  005130  104013                          ERROR   13              ;SOFTWARE TIMEOUT
 1750  005132  104014                          ERROR   14              ;DRIVE UNSAFE ERROR
 1751  005134  032737  177577  001374          BIT     #^C<HCE>,FMTDPB+16  ;IS ONLY ERROR A HEADER COMPARE ERROR ?
 1752  005142  001401                          BEQ     3$              ;BR IF YES
 1753  005144  104021                          ERROR   21              ;CONTROLLER/DRIVE ERROR VERIFYING HEADERS
 1754  005146  005037  001160          3$:     CLR     $ESCAPE         ;CLEAR THE ERROR ESCAPE ADDRESS
 1755  005152  104022                          ERROR   22              ;HEADER COMPARE ERROR VERIFYING HEADERS
 1756  005154  004737  005464                  JSR     PC,LOP.CK       ;CHECK FOR LOOP ON ERROR
 1757  005160  023737  025314  025324  4$:     CMP     RBUF,BUFP       ;SEE IF CYL READ EQUALS CYL EXPECTED
 1758  005166  001403                          BEQ     6$              ;BR IF CYL CORRECT
 1759  005170  104023                          ERROR   23              ;CYLINDER NOT CORRECT
 1760  005172  004737  005464          5$:     JSR     PC,LOP.CK       ;CHECK FOR LOOP ON ERROR
 1761  005176  023737  001222  001246  6$:     CMP     ENDCYL,CYLCK    ;SEE IF LAST CYLINDER
 1762  005204  001002                          BNE     7$              ;BR IF NOT FINISHED
 1763  005206  000137  005230                  JMP     $EOP            ;END OF FORMAT
 1764  005212  005237  001246          7$:     INC     CYLCK           ;INCREMENT CYLINDER ADDRESS BEING CHECKED
 1765  005216  004737  006012                  JSR     PC,UPDACY       ;SET UP FOR NEXT CYL
 1766  005222  005237  001370                  INC     FMTDPB+12       ;ADVANCE TO NEXT CYL #
 1767  005226  000720                          BR      1$              ;GO READ NEXT HEADER
 1768
 1769                                  .SBTTL  END OF PASS ROUTINE
 1770
 1771                                  ;;*****************************************************************
 1772                                  ;*INCREMENT THE PASS NUMBER ($PASS)
 1773                                  ;*IF THERES A MONITOR GO TO IT
 1774                                  ;*IF THERE ISN'T JUMP TO DONE
 1775
 1776  005230                          $EOP:
 1777  005230  005737  001220                  TST     MODE            ;'FORMAT' OR 'CHECK' MODE ?
 1778  005234  001403                          BEQ     3$              ;BR IF 'CHECK'
 1779  005236  104401  022114                  TYPE    ,MFCMPT         ;'FORMAT COMPLETE'
 1780  005242  000402                          BR      4$              ;FINISH THE END MESSAGE
 1781  005244  104401  022141          3$:     TYPE    ,MCCMPT         ;'CHECK COMPLETE'
 1782  005250  104401  022165          4$:     TYPE    ,NUMERR         ;'TOTAL ERRORS DETECTED: '
 1783  005254  013746  001112                  MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
 1784  005260  104405                          .TYPDS                  ;;GO TYPE--DECIMAL ASCII WITH SIGN
 1785  005262  104401  001167                  TYPE    ,$CRLF          ;CR-LF
 1786  005266  005237  001100                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
 1787  005272  042737  100000  001100          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
 1788  005300  005327                          DEC     (PC)+           ;;LOOP?
 1789  005302  000001          $EOPCT: .WORD   1
 1790  005304  003013                          BGT     $DOAGN          ;;YES
 1791  005306  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
 1792  005310  000001          $ENDCT: .WORD   1
 1793  005312  005302                          $EOPCT
 1794  005314  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
```

```
 1795  005320  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
 1796  005322  000005                          RESET                   ;;CLEAR THE WORLD
 1797  005324  004710                $ENDAD:   JSR     PC,(R0)         ;;GO TO MONITOR
 1798  005326  000240                          NOP                     ;;SAVE ROOM
 1799  005330  000240                          NOP                     ;;FOR
 1800  005332  000240                          NOP                     ;;ACT11
 1801  005334                        $DOAGN:
 1802  005334  000137                          JMP     @(PC)+          ;;RETURN
 1803  005336  005340                $RTNAD:   .WORD   DONE
 1804  005340  032777  000001  173572 DONE:    BIT     #SW0,@SWR       ;SEE IF SWR 0 SET
 1805  005346  001002                          BNE     1$              ;BR IF SET
 1806  005350  000137  003134                  JMP     M0              ;ASK FOR NEXT DRIVE
 1807  005354  012706  001100          1$:     MOV     #STACK,SP       ;RESET STACK POINTER
 1808  005360  000137  003544                  JMP     M5              ;DO THE FORMAT OR CHECK AGAIN
 1809
 1810                                ;;**************************************************************************
 1811
 1812                                .SBTTL  SUPPO   SUBROUTINES
 1813
 1814                                ;;**************************************************************************
 1815
 1816                                ;THIS ROUTINE DETERMINES THE ERROR TYPE
 1817
 1818  005364  010146                ERINDX:   MOV     R1,-(SP)        ;STORE R1
 1819  005366  005001                          CLR     R1              ;CLEAR R1
 1820  005370  033727  001374                  BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
 1821  005374  060006                          .WORD   BIT14!BIT13!BIT2!BIT1  ;DRIVE OFFLINE
 1822  005376  001025                          BNE     5$              ;BR IF OFFLINE
 1823  005400  033727  001374                  BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
 1824  005404  010000                          .WORD   BIT12           ;DRIVE PERSISTENTLY UNSAFE
 1825  005406  001020                          BNE     4$              ;BR IF UNSAFE
 1826  005410  033727  001374                  BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
 1827  005414  006000                          .WORD   BIT11!BIT10     ;UNCORRECTABLE MASSBUS PARITY ERROR ?
 1828  005416  001013                          BNE     3$              ;BR IF PARITY ERROR
 1829  005420  033727  001374                  BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
 1830  005424  001400                          .WORD   BIT09!BIT08     ;SOFTWARE TIMEOUT ?
 1831  005426  001006                          BNE     2$              ;BR IF YES
 1832  005430  033727  001374                  BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
 1833  005434  000020                          .WORD   BIT04           ;DRIVE UNSAFE ERROR ?
 1834  005436  001001                          BNE     1$              ;BR IF YES
 1835  005440  005201                          INC     R1              ;INCREMENT THE RETURN INDEX
 1836  005442  005201                  1$:     INC     R1              ;INCREMENT THE RETURN INDEX
 1837  005444  005201                  2$:     INC     R1              ;INCREMENT THE RETURN INDEX
 1838  005446  005201                  3$:     INC     R1              ;INCREMENT THE RETURN INDEX
 1839  005450  005201                  4$:     INC     R1              ;INCREMENT THE RETURN INDEX
 1840  005452  006301                  5$:     ASL     R1              ;DOUBLE THE INCREMENT
 1841  005454  060166  000002                  ADD     R1,2(SP)        ;DEVELOP THE RETURN ADDRESS
 1842  005460  012601                          MOV     (SP)+,R1        ;RESTORE R1
 1843  005462  000207                          RTS     PC              ;RETURN
 1844
 1845                                ;ROUTINE TO CHECK FOR LOOP ON ERROR
 1846
 1847  005464  032777  001000  173446 LOP.CK:  BIT     #SW09,@SWR      ;LOOP ON ERROR ?
 1848  005472  001402                          BEQ     1$              ;BR IF NOT
 1849  005474  000177  173410                  JMP     @$LPERR         ;GO TO THE LOOP ON ERROR ADDRESS
 1850  005500  005037  001160          1$:     CLR     $ESCAPE         ;CLEAR THE ERROR ESCAPE ADDRESS
```

```
 1851  005504  033727  001374          BIT     FMTDPB+16,(PC)+  ;CHECK FOR 'FATAL' ERROR
 1852  005510  072002                  .WORD   BIT14!BIT13!BIT12!BIT10!BIT01  ;'FATAL' ERROR BITS
 1853  005512  001004                  BNE     2$               ;BR IF NOT
 1854  005514  032737  004000  001322  BIT     #WLE,RP.REG+RPER1  ;WRITE LOCK ERROR ?
 1855  005522  001402                  BEQ     3$               ;BR IF NOT
 1856  005524  000137  005230  2$:     JMP     $EOP             ;TERMINATE THE FORMAT
 1857  005530  000207          3$:     RTS     PC               ;RETURN
 1858
 1859                          ;THIS ROUTINE SETS UP THE INPUT TABLE FOR THE DRIVER ROUTINE
 1860
 1861  005532  113737  001214  001356  SETTBL: MOVB    DRIVE,FMTDPB     ;SET UP DRIVE #
 1862  005540  105037  001361          CLRB    FMTDPB+3         ;CLEAR HIGH ORDER ADRS BITS
 1863  005544  013737  001260  001362  MOV     MWC,FMTDPB+4     ;LOAD UP WORD COUNT
 1864  005552  012737  025324  001364  MOV     #BUFP,FMTDPB+6   ;LOAD UP CURRENT ADRS
 1865  005560  105037  001366          CLRB    FMTDPB+10        ;SET SECTOR TO ZERO
 1866  005564  113737  001230  001367  MOVB    BEGTRK,FMTDPB+11  ;SET UP STARTING TRK ADRS
 1867  005572  013737  001224  001370  MOV     BEGCYL,FMTDPB+12  ;SET UP STARTING CYL
 1868  005600  005037  001374          CLR     FMTDPB+16        ;CLEAR RP04 STATUS
 1869  005604  013737  001230  001236  MOV     BEGTRK,TRKCNT    ;SET UP PARTIAL CYL TRACK COUNT
 1870  005612  013737  001232  001234  MOV     TTRKS,TTRKSC     ;SET UP TOTAL TRACKS COUNTER
 1871  005620  000207                  RTS     PC               ;RETURN FROM SETUP
 1872
 1873                          ;THIS ROUTINE CONTROLS THE DISK ADDRESSING AND TOTAL TRK COUNT
 1874                          ;IT IS ENTERED AFTER EVERY TRK OPERATION
 1875
 1876  005622  005337  001234  TRKTST: DEC     TTRKSC           ;SEE IF LAST TRACK
 1877  005626  001425          BEQ     3$               ;BRANCH IF LAST TRACK
 1878  005630  022737  000022  001236  CMP     #18.,TRKCNT      ;IS THIS THE LAST TRACK IN THE CYLINDER ?
 1879  005636  001407          BEQ     1$               ;BRANCH IF SO
 1880  005640  005237  001236          INC     TRKCNT           ;COUNT UP TRACK WITHIN CYLINDER
 1881  005644  105237  001367          INCB    FMTDPB+11        ;INCREMENT TRACK NUMBER
 1882  005650  004737  006042          JSR     PC,UPDATK        ;UPDATE TRACK ADDRESS IN BUFFER
 1883  005654  000410                  BR      2$               ;EXIT
 1884  005656  005037  001236  1$:     CLR     TRKCNT           ;CLEAR TRACK COUNT FOR NEXT CYLINDER
 1885  005662  105037  001367          CLRB    FMTDPB+11        ;RESET TRACK ADDRESS TO 0
 1886  005666  005237  001370          INC     FMTDPB+12        ;UPDATE CYLINDER NUMBER
 1887  005672  004737  006012          JSR     PC,UPDACY        ;UPDATE CYLINDER NUMBER IN BUFFER
 1888  005676  062716  000002  2$:     ADD     #2,(SP)          ;ADD TWO TO RETURN ADRS
 1889  005702  000207          3$:     RTS     PC               ;RETURN
 1890
 1891                          ;THIS ROUTINE SETS UP WC & BA FOR REMAINING SECTORS
 1892                          ;AFTER A WRITE CHECK ERROR
 1893
 1894  005704  013737  001254  001362  SCAWC:  MOV     SAVWC,FMTDPB+4  ;SET UP WC FOR REMAINING SECTORS
 1895  005712  062737  001010  001364  ADD     #520.,FMTDPB+6  ;SET UP BA FOR REMAINING SECTORS
 1896  005720  005237  001366          INC     FMTDPB+10        ;ADVANCE TO NEXT SECTOR IN TBL
 1897  005724  005037  001216          CLR     SOFSW            ;RESET RETRY COUNTER
 1898  005730  000207                  RTS     PC               ;RETURN TO COMPLETE TRK FORMAT
 1899
 1900
 1901                          ;THIS ROUTINE SETS UP THE CYLINDER ADRS, FORMAT BIT, TRACK AND
 1902                          ;SECTOR ADRS IN MEMORY WITH THE STARTING CYL - TRK INFORMATION
 1903
 1904  005732  013701  001266  SETHDR: MOV     MAXSEC,R1        ;SET UP SECTOR COUNT
 1905  005736  012700  025324          MOV     #BUFP,R0         ;SET UP HEADER POINTER IN R0
 1906  005742  013702  001224          MOV     BEGCYL,R2        ;PUT STARTING CYL # IN R2
```

```
 1907  005746  013703  001230              MOV     BEGTRK,R3       ;PUT STARTING TRK # IN R3
 1908  005752  000303                      SWAB    R3              ;JUSTIFY TRACK ADRS
 1909  005754  005737  001264              TST     SEC20           ;SEE IF 20 OR 22 SECTOR MODE
 1910  005760  001402                      BEQ     1$              ;BR IF 20 SECTOR MODE
 1911  005762  052702  010000              BIS     #10000,R2       ;SET THE 22 SECTOR FORMAT BIT
 1912  005766  010220              1$:     MOV     R2,(R0)+        ;WRITE IN HEADER AREA OF CORE THE CYL ADRS
 1913  005770  010320                      MOV     R3,(R0)+        ;WRITE IN HEADER AREA OF CORE THE TRK ADRS
 1914  005772  005020                      CLR     (R0)+           ;CLR 1ST KEYWORD
 1915  005774  005010                      CLR     (R0)            ;CLR 2ND KEYWORD
 1916  005776  062700  001002              ADD     #514.,R0        ;SET UP FOR NEXT HEADER
 1917  006002  005203                      INC     R3              ;UPDATE SECTOR ADRS FOR NEXT HEADER
 1918  006004  005301                      DEC     R1              ;MAINTAIN COUNT OF SECTORS
 1919  006006  002367                      BGE     1$              ;BRANCH IF NOT LAST SECTOR
 1920  006010  000207                      RTS     PC              ;EXIT - HEADERS ARE LOADED INTO CORE
 1921
 1922                          ;THIS ROUTINE UPDATES THE CYLINDER ADRS OF THE HEADER WORDS IN CORE
 1923
 1924  006012  013701  001266  UPDACY:    MOV     MAXSEC,R1       ;SET UP SECTOR COUNT
 1925  006016  012700  025324              MOV     #BUFP,R0        ;SET UP HEADER POINTER IN R0
 1926  006022  005220              1$:     INC     (R0)+           ;INCREMENT FOR NEXT CYLINDER
 1927  006024  042710  177400              BIC     #177400,(R0)    ;RESET TRK ADRS TO 0
 1928  006030  062700  001006              ADD     #518.,R0        ;SET UP FOR NEXT HEADER
 1929  006034  005301                      DEC     R1              ;COUNT SECTORS
 1930  006036  002371                      BGE     1$              ;BRANCH IF NOT LAST SECTOR
 1931  006040  000207                      RTS     PC              ;EXIT
 1932
 1933                          ;THIS ROUTINE UPDATES THE TRACK ADRS OF THE HEADER WORDS IN CORE
 1934
 1935  006042  013701  001266  UPDATK:    MOV     MAXSEC,R1       ;SET UP SECTOR COUNT
 1936  006046  012700  025324              MOV     #BUFP,R0        ;SET UP HEADER POINTER IN R0
 1937  006052  005720                      TST     (R0)+           ;POINT HEADER POINTER TO TRK - SEC ADRS
 1938  006054  062710  000400              ADD     #400,(R0)       ;INDEX TRK ADRS
 1939  006060  062700  001010              ADD     #520.,R0        ;SET UP FOR NEXT HEADER
 1940  006064  005301                      DEC     R1              ;COUNT SECTORS
 1941  006066  002372                      BGE     1$              ;BRANCH IF NOT LAST SECTOR
 1942  006070  000207                      RTS     PC              ;EXIT
 1943
 1944                          ;ROUTINE TO CHECK FOR KW11-L OR KW11-P CLOCKS
 1945
 1946  006072  012737  006150  000004  ST.CLK:   MOV     #STCLK1,@#ERRVEC  ;SET UP VECTOR FOR P CLK
 1947  006100  005037  000006              CLR     @#ERRVEC+2      ;NEW PSW
 1948  006104  005777  173066              TST     @$LKCSR         ;CHECK FOR KW11-P
 1949  006110  013746  001202              MOV     $LPVEC,-(SP)    ;VECTOR ADDRESS
 1950  006114  012776  006234  000000      MOV     #CLOCK,@(SP)    ;SET UP KW11-P VECTOR
 1951  006122  062716  000002              ADD     #2,(SP)         ;POINT TO PSW
 1952  006126  012736  000300              MOV     #PR6,@(SP)+     ;PSW - PRI 6
 1953  006132  012777  177777  173040      MOV     #-1,@$LKCSB     ;LOAD COUNTER BUFFER
 1954  006140  012777  000135  173030      MOV     #135,@$LKCSR    ;SET CLK - CNT UP
 1955  006146  000426                      BR      STCLK3
 1956  006150  062706  000004  STCLK1:    ADD     #4,SP           ;RESTORE THE STACK POINTER
 1957  006154  012737  006220  000004      MOV     #STCLK2,@#ERRVEC        ;CHANGE ERROR VECTOR
 1958  006162  005777  173020              TST     @$LKS           ;LOOK FOR KW11-L
 1959  006166  013746  001210              MOV     $LLVEC,-(SP)    ;KW11-L VECTOR ADDRESS
 1960  006172  012776  006234  000000      MOV     #CLOCK,@(SP)    ;SET UP KW11-L VECTOR
 1961  006200  062716  000002              ADD     #2,(SP)         ;INCREMENT VECTOR ADDRESS
 1962  006204  012736  000300              MOV     #PR6,@(SP)+     ;PSW - PRI 6
```

```
 1963  006210  012777  000100  172770          MOV     #1G0,@$LKS      ;SET KW11-L INTERRUPT ENABLE
 1964  006216  000402                          BR      STCLK3
 1965  006220  062706  000004          STCLK2: ADD     #4,SP           ;RESTORE THE STACK POINTER
 1966  006224  012737  000006  000004  STCLK3: MOV     #6,@#ERRVEC     ;RESTORE THE ERROR VECTOR
 1967  006232  000207                          RTS     PC
 1968
 1969                                  ;THIS CODE SERVICES A CLOCK INTERRUPT EVERY 16MS
 1970
 1971  006234  012746  000020          CLOCK:  MOV     #16.,-(SP)      ;PUT MILLISECONDS ON THE STACK
 1972  006240  004737  016616                  JSR     PC,RPTMR        ;GO REPORT TIME
 1973  006244  000002                          RTI                     ;RETURN AND CONTINUE
 1974
 1975                                  ;ROUTINE TO INTERCEPT 'CONTROL C' TYPED DURING PARAMETER ENTRY TIME
 1976
 1977  006246  012706  001100          OENTER: MOV     #STACK,SP       ;INITIALIZE THE STACK
 1978  006252  005737  012310          1$:     TST     TRNSWT          ;ALL ACTIVITY STOPPED ?
 1979  006256  001375                          BNE     1$              ;BR IF NOT
 1980  006260  000005                          RESET                   ;CLEAR THE BUS
 1981  006262  000137  003100                  JMP     M1B             ;START AGAIN WITH DRIVE SELECTION
 1982
 1983                                  ;ROUTINE TO TYPE THE PRESENT DISK ADDRESS.  ENTERED BY TYPING 'CONTROL C'
 1984
 1985  006266  005037  177776          TYPADR: CLR     PSW             ;SET PROCESSOR TO PRIORITY 0
 1986  006272  012737  006246  001300          MOV     #OENTER,CNTLC   ;CHANGE 'CONTROL C' RETURN ADDRESS
 1987  006300  104401  022215                  TYPE    ,ADDRIS         ;'PRESENT ADDRESS IS: '
 1988  006304  104401  021025                  TYPE    ,C              ;'C'
 1989  006310  013746  001370                  MOV     FMTDPB+12,-(SP) ;PUT THE CYLINDER ADDRESS ON THE STACK
 1990  006314  004737  011620                  JSR     PC,$SB2D        ;CONVERT IT TO DECIMAL
 1991  006320  004737  011560                  JSR     PC,$SUPRS       ;TYPE IT
 1992  006324  104401  021035                  TYPE    ,LINSP          ;SPACES
 1993  006330  104401  021027                  TYPE    ,T              ;'T'
 1994  006334  005046                          CLR     -(SP)           ;CLEAR THE STACK
 1995  006336  113716  001367                  MOVB    FMTDPB+11,(SP)  ;PUT THE TRACK ADDRESS ON THE STACK
 1996  006342  004737  011620                  JSR     PC,$SB2D        ;CONVERT IT TO DECIMAL
 1997  006346  004737  011560                  JSR     PC,$SUPRS       ;TYPE IT
 1998  006352  104401  001167                  TYPE    ,$CRLF          ;CR-LF
 1999  006356  012737  006266  001300          MOV     #TYPADR,CNTLC   ;RESTORE ENTRANCE TO THIS ROUTINE
 2000  006364  000002                          RTI                     ;RETURN
 2001
 2002                                  ;PARAMETER ENTRY ROUTINE
 2003                                  ;CALL
 2004                                  ;       MOV     #ADR,R3         ;PARAMETER TABLE ADDRESS
 2005                                  ;       JSR     PC,PARENT       ;GET THE PARAMETERS
 2006
 2007  006366  010346                  PARENT: MOV     R3,-(SP)        ;SAVE R3
 2008  006370  012703  001376                  MOV     #TABLE,R3       ;PARAMETER TABLE ADDRESS
 2009  006374  012337  006404          1$:     MOV     (R3)+,3$        ;ADDRESS OF PARAMETER NAME
 2010  006400  001436                          BEQ     9$              ;BR IF AT END OF TABLE
 2011  006402  104401                          TYPE                    ;TYPE THE PARAMETER NAME
 2012  006404  000000          3$:     .WORD   0                       ;ADDRESS OF PARAMETER NAME TEXT
 2013  006406  012302                          MOV     (R3)+,R2        ;MAXIMUM PARAMETER VALUE
 2014  006410  012305                          MOV     (R3)+,R5        ;ADDRESS OF PARAMETER
 2015  006412  011546                          MOV     (R5),-(SP)      ;CURRENT VALUE OF PARAMETER
 2016  006414  104405                          TYPDS                   ;TYPE THE CURRENT VALUE OF THE PARAMETER
 2017  006416  104401  021021                  TYPE    ,SLASH          ;' / '
 2018  006422  104411                          RDLIN                   ;READ THE KEYBOARD
```

```
                                                MOV     (SP)+,R1        ;INPUT ASCII STRING ADDRESS
                                                JSR     R5,CK.DIG       ;CHECK THE DIGIT(S)
                                                1$                      ;CARRIAGE RETURN ONLY ENTERED
                                                9$                      ;PERIOD ONLY ENTERED
                                                6$                      ;ILLEGAL INPUT
                                                5$                      ;TERMINATED WITH A CARRIAGE RETURN
                                                6$                      ;TERMINATED WITH A ','
                                                7$                      ;TERMINATED WITH A ':'
                                        5$:     MOV     R2,(R5)         ;MOVE NEW VALUE TO PARAMETER LOCATION
                                                BR      1$              ;GET MORE PARAMETERS
                                        6$:     TYPE    ,BADENT         ;'BAD ENTRY'
                                                SUB     #6,R3           ;DECREMENT THE TABLE POINTER
                                                BR      1$              ;TRY AGAIN
                                        7$:     MOV     R2,(R5)         ;NEW VALUE
                                                BR      9$              ;EXIT
                                        8$:     MOV     #TABLE,R3       ;RELOAD THE PARAMETER TABLE ADDRESS
                                                BR      1$              ;TRY AGAIN
                                        9$:     MOV     (SP)+,R3        ;RESTORE R3
                                                RTS     PC              ;RETURN


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

                                        CK.OCT: CMPB    (R1),#'0        ;LESS THAN ZERO?
                                                BLO     1$              ;YES -- BRANCH
                                                CMPB    (R1),#'7        ;GREATER THAN SEVEN?
                                                BHI     1$              ;YES -- BRANCH
                                                MOVB    (R1),R2         ;GET THE CHARACTER
                                                BIC     #^C7,R2         ;STRIP AWAY THE ASCII
                                                TST     (R5)+           ;ADJUST FOR RETURN
                                        1$:     RTS     R5              ;RETURN

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

                                        CK.DEC: CMPB    (R1),#'0        ;LESS THAN ZERO?
                                                BLO     1$              ;YES -- BRANCH
                                                CMPB    (R1),#'9        ;GREATER THAN NINE?
                                                BHI     1$              ;YES -- BRANCH
                                                MOVB    (R1),R2         ;GET THE CHARACTER
                                                BIC     #'0,R2          ;STRIP AWAY THE ASCII
                                                TST     (R5)+           ;ADJUST FOR RETURN
                                        1$:     RTS     R5              ;RETURN
```

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

006556  105711         CK.CHR: TSTB    (R1)            ;''CARRIAGE RETURN''?
006560  001417                 BEQ     3$              ;YES -- BRANCH
006562  121127  000054         CMPB    (R1),#',        ;''COMMA''?
006566  001413                 BEQ     2$              ;YES -- BRANCH
006570  121127  000056         CMPB    (R1),#'.        ;''PERIOD''?
006574  001407                 BEQ     1$              ;YES -- BRANCH
006576  004537  006530         JSR     R5,CK.DEC       ;''DIGIT''?
006602  000410                 BR      4$              ;NO -- BRANCH
006604  004537  006502         JSR     R5,CK.OCT       ;OCTAL ?
006610  005725                 TST     (R5)+           ;DIGIT BETWEEN 8-9
006612  005725                 TST     (R5)+           ;DIGIT BETWEEN 0-7
006614  005725         1$:     TST     (R5)+           ;PERIOD
006616  005725         2$:     TST     (R5)+           ;COMMA
006620  005725         3$:     TST     (R5)+           ;CARRIAGE RETURN
006622  005201                 INC     R1              ;MOVE POINTER TO NEXT CHARACTER
006624  011505         4$:     MOV     (R5),R5         ;UNKNOWN CHARACTER
006626  000205                 RTS     R5              ;RETURN

                                ;THIS ROUTINE CHECKS AN ASCII STRING FOR LEGAL
                                ;CHARACTERS AND FORMS A DECIMAL VALUE BINARY NUMBER IN R2.
                                ;CALL
                                ;       MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
                                ;       MOV     #NUM,R2         ;MAX. MAGNITUDE OF INPUT NUMBER
                                ;       JSR     R5,CK.DIG       ;CHECK DIGITS
                                ;       RETURN  ADR1            ;''CR'' ONLY ENTERED -- R2=0
                                ;       RETURN  ADR2            ;''PERIOD'' ONLY ENTERED -- R2=0
                                ;       RETURN  ADR3            ;ILLEGAL CHARACTER OR INPUT TOO LARGE -- R2=?
                                ;       RETURN  ADR4            ;''CR'' -- R2 = NUMBER
                                ;       RETURN  ADR5            ;''COMMA'' -- R2 = NUMBER
                                ;       RETURN  ADR6            ;''PERIOD'' -- R2 = NUMBER

006630  010446         CK.DIG: MOV     R4,-(SP)        ;SAVE R4
006632  010346                 MOV     R3,-(SP)        ;SAVE R3
006634  010246                 MOV     R2,-(SP)        ;SAVE THE MAX. SIZE ON THE STACK
006636  005002                 CLR     R2              ;START WITH 0
006640  005003                 CLR     R3
006642  005004                 CLR     R4
006644  004537  006556         JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
006650  006744                 6$                      ;ILLEGAL CHARACTER
006652  006752                 9$                      ;CARRIAGE RETURN
006654  006744                 6$                      ;''-''
006656  006746                 7$                      ;''.''
```

```
 2131  006660  006664                          1$                      ;DIGIT 0-7
 2132  006662  006664                          1$                      ;DIGIT 8-9
 2133  006664  062705  000004          1$:     ADD     #4,R5           ;STEP RETURN POINTER PAST ''CR'' & 'PERIOD'' RETURNS
 2134  006670  006303                  2$:     ASL     R3              ;INPUT NUMBER *2
 2135  006672  010346                          MOV     R3,-(SP)        ;SAVE *2
 2136  006674  006303                          ASL     R3              ;*4
 2137  006676  006303                          ASL     R3              ;*8
 2138  006700  062603                          ADD     (SP)+,R3        ;(*2)+(*8) = *10
 2139  006702  060203                          ADD     R2,R3           ;UPDATE THE INPUT NUMBER
 2140  006704  004537  006556                  JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
 2141  006710  006750                          8$                      ;ILLEGAL CHARACTER
 2142  006712  006734                          5$                      ;CARRIAGE RETURN
 2143  006714  006732                          4$                      ;''
 2144  006716  006724                          3$                      ;''
 2145  006720  006670                          2$                      ;DIGIT 0-7
 2146  006722  006670                          2$                      ;DIGIT 8-9
 2147  006724  105711                  3$:     TSTB    (R1)            ;DOES A ''CR'' FOLLOW THE 'PERIOD''
 2148  006726  001010                          BNE     8$              ;BR IF NOT
 2149  006730  005724                          TST     (R4)+           ;INCREMENT THE RETURN
 2150  006732  005724                  4$:     TST     (R4)+           ;INCREMENT THE RETURN
 2151  006734  005724                  5$:     TST     (R4)+           ;INCREMENT THE RETURN
 2152  006736  020316                          CMP     R3,(SP)         ;CHECK THE MAGNITUDE OF THE NUMBER
 2153  006740  101004                          BHI     9$              ;BR IF ENTERED NUMBER TOO LARGE
 2154  006742  000402                          BR      8$              ;BYPASS INCREMENT
 2155  006744  005725                  6$:     TST     (R5)+           ;INCREMENT RETURN PAST INVALID RETURN
 2156  006746  005725                  7$.     TST     (R5)+           ;INCREMENT RETURN
 2157  006750  060405                  8$:     ADD     R4,R5           ;SETUP RETURN POINTER
 2158  006752  010302                  9$:     MOV     R3,R2           ;ENTERED VALUE
 2159  006754  005726                          TST     (SP)+           ;CLEAN MAX. SIZE OFF OF STACK
 2160  006756  012603                          MOV     (SP)+,R3        ;RESTORE R3
 2161  006760  012604                          MOV     (SP)+,R4        ;RESTORE R4
 2162  006762  011505                          MOV     (R5),R5         ;GET RETURN ADDRESS
 2163  006764  000205                          RTS     R5              ;RETURN
 2164
 2165                                  .SBTTL  MACRO ROUTINES
 2166
 2167
 2168                                  .SBTTL  ERROR HANDLER ROUTINE
 2169
 2170                                  ;;**********************************************************************
 2171                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
 2172                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
 2173                                  ;*AND GO TO TYPERR ON ERROR
 2174                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 2175                                  ;*SW15=1         HALT ON ERROR
 2176                                  ;*SW13=1         INHIBIT ERROR TYPEOUTS
 2177                                  ;*SW10=1         BELL ON ERROR
 2178                                  ;*SW09=1         LOOP ON ERROR
 2179                                  ;*CALL
 2180                                  ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
 2181
 2182  006766                          $ERROR:
 2183  006766  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
 2184  006770  010137  001274                  MOV     R1,DDRIVE       ;DRIVE ADDRESS IF DRIVE ERROR CALL
 2185  006774  010337  001276                  MOV     R3,ATTN         ;ATTENTION REGISTER CONTENTS
 2186  007000  013737  001370  001270          MOV     FMTDPB+12,DS.CYL  ;CURRENT CYLINDER ADDRESS
```

```
 2187  007006  113737  001367  001272          MOVB    FMTDPB+11,DS.TRK  ;REQUESTED TRACK ADDRESS
 2188  007014  005737  001312                  TST     RP.REG+RPBA       ;NON-ZERO BUFFER ADDRESS ?
 2189  007020  001406                          BEQ     7$                ;BR IF NO BUFFER ADDRESS
 2190  007022  013746  001312                  MOV     RP.REG+RPBA,-(SP) ;BUFFER ADDRESS
 2191  007026  162716  000002                  SUB     #2,(SP)           ;DECREMENT THE ADDRESS
 2192  007032  013637  001124                  MOV     @(SP)+,$GDDAT     ;GET THE BUFFER WORD WHICH DIDN'T COMPARE
 2193  007036  105237  001103          7$:     INCB    $ERFLG            ;;SET THE ERROR FLAG
 2194  007042  001775                          BEQ     7$                ;;DON'T LET THE FLAG GO TO ZERO
 2195  007044  013777  001102  172070          MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER AND ERROR FLAG
 2196  007052  032777  002000  172060          BIT     #BIT10,@SWR       ;;BELL ON ERROR?
 2197  007060  001402                          BEQ     1$                ;;NO - SKIP
 2198  007062  104401  001162                  TYPE    ,$BELL            ;;RING BELL
 2199  007066  005237  001112          1$:     INC     $ERTTL            ;;COUNT THE NUMBER OF ERRORS
 2200  007072  011637  001116                  MOV     (SP),$ERRPC       ;;GET ADDRESS OF ERROR INSTRUCTION
 2201  007076  162737  000002  001116          SUB     #2,$ERRPC
 2202  007104  117737  172006  001114          MOVB    @$ERRPC,$ITEMB    ;;STRIP AND SAVE THE ERROR ITEM CODE
 2203  007112  032777  020000  172020          BIT     #BIT13,@SWR       ;;SKIP TYPEOUT IF SET
 2204  007120  001004                          BNE     20$               ;;SKIP TYPEOUTS
 2205  007122  004737  007174                  JSR     PC,TYPERR         ;;GO TO USER ERROR ROUTINE
 2206  007126  104401  001167                  TYPE    ,$CRLF
 2207  007132                          20$:
 2208  007132  005777  172002          2$:     TST     @SWR              ;;HALT ON ERROR
 2209  007136  100002                          BPL     3$                ;;SKIP IF CONTINUE
 2210  007140  000000                          HALT                      ;;HALT ON ERROR!
 2211  007142  104407                          CKSWR                     ;;TEST FOR CHANGE IN SOFT-SWR
 2212  007144  032777  001000  171766  3$:     BIT     #BIT09,@SWR       ;;LOOP ON ERROR SWITCH SET?
 2213  007152  001402                          BEQ     4$                ;;BR IF NO
 2214  007154  013716  001110                  MOV     $LPERR,(SP)       ;;FUDGE RETURN FOR LOOPING
 2215  007160  005737  001160          4$:     TST     $ESCAPE           ;;CHECK FOR AN ESCAPE ADDRESS
 2216  007164  001402                          BEQ     5$                ;;BR IF NONE
 2217  007166  013716  001160                  MOV     $ESCAPE,(SP)      ;;FUDGE RETURN ADDRESS FOR ESCAPE
 2218  007172                          5$:
 2219  007172  000002                          RTI                       ;;RETURN
 2220
 2221
 2222                                  ;THIS ROUTINE USES THE ''ITEM CONTROL BYTE'' ($ITEMB) TO DETERMINE
 2223                                  ;WHICH ERROR IS TO BE REPORTED, IT THEN OBTAINS, FROM THE 'ERROR
 2224                                  ;TABLE'' ($ERRTB), AND REPORTS THE APPROPIATE INFORMATION
 2225                                  ;CONCERNING THE ERROR.
 2226
 2227  007174  104412                  TYPERR: SAVREG                    ;SAVE R0-R5
 2228  007176  005000                          CLR     R0                ;CLEAR R0 FOR ERROR NUMBER
 2229  007200  113700  001114                  MOVB    $ITEMB,R0         ;ERROR NUMBER
 2230  007204  005300                          DEC     R0                ;FORM INDEX FOR ERROR TABLE
 2231  007206  006300                          ASL     R0
 2232  007210  006300                          ASL     R0
 2233  007212  006300                          ASL     R0
 2234  007214  062700  001630          1$:     ADD     #$ERRTB,R0        ;FORM ADDRESS
 2235  007220  012037  007234                  MOV     (R0)+,2$          ;GET ERROR MESSAGE (EM) POINTER
 2236  007224  001404                          BEQ     3$                ;BRANCH IF THERE ISN'T ONE
 2237  007226  104401  001167                  TYPE    ,$CRLF            ;''CARRIAGE RETURN - LINE FEED
 2238  007232  104401                          TYPE
 2239  007234  000000          2$:     .WORD   0                         ;''EM'' POINTER GOES HERE
 2240  007236  012037  007252          3$:     MOV     (R0)+,4$          ;PICK UP DATA HEADER (DH) POINTER
 2241  007242  001404                          BEQ     5$                ;BRANCH IF NONE
 2242  007244  104401  001167                  TYPE    ,$CRLF            ;CARRIAGE RETURN-LINE FEED
```

```
                                        TYPE
                                4$:     .WORD   0               ;'DH'' POINTER GOES HERE
                                5$:     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    ,LINSP          ;YES--TYPE SEPERATORS
                                        BR      11$             ;LOOP
                                14$:    DEC     R2              ;MORE DH'S?
                                        BLE     20$             ;NO--BRANCH
                                        TYPE    ,$CRLF          ;YES--START A NEW LINE
                                        TST     2(R0)           ;ONLY A 'DH' IN THIS REQUEST ?
                                        BEQ     15$             ;BR IF YES - BYPASS THE INDENT
                                        COM     R5              ;INDENT?
                                        BNE     15$             ;NO--BRANCH
                                        TYPE    ,LINSP          ;YES--TYPE SPACES
                                15$:    MOV     (R0)+,16$       ;GET NEXT DH
                                        TYPE                    ;AND TYPE IT
                                16$:    .WORD   0               ;DH POINTER GOES HERE
                                        TST     (R0)            ;TYPE A 'DT' ?
                                        BNE     21$             ;BR IF A 'DT'
                                        ADD     #4,R0           ;INCREMENT THE 'DF' POINTER
                                        BR      14$             ;SEE IF END OF 'DF' BLOCK
                                21$:    TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
                                        TST     R5              ;INDENT?
                                        BNE     10$             ;NO--BRANCH
                                17$:    TYPE    ,LINSP          ;YES--TYPE SPACES
                                        BR      10$             ;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.
                                ;*
```

```
 2299                                  .*CALL:
 2300                                  ;*1) USING A TRAP INSTRUCTION
 2301                                  ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
 2302                                  ;*OR
 2303                                  ;*      TYPE
 2304                                  ;*      MESADR
 2305                                  ;*
 2306
 2307  007424  105737  001157         $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
 2308  007430  100002                         BPL     1$              ;;BR IF YES
 2309  007432  000000                         HALT                    ;;HALT HERE IF NO TERMINAL
 2310  007434  000407                         BR      3$              ;;LEAVE
 2311  007436  010046                 1$:     MOV     R0,-(SP)        ;;SAVE R0
 2312  007440  017600  000002                 MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
 2313  007444  112046                 2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
 2314  007446  001005                         BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
 2315  007450  005726                         TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
 2316  007452  012600                 60$:    MOV     (SP)+,R0        ;;RESTORE R0
 2317  007454  062716  000002         3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
 2318  007460  000002                         RTI                     ;;RETURN
 2319  007462  122716  000011         4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
 2320  007466  001430                         BEQ     8$
 2321  007470  122716  000200                 CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
 2322  007474  001006                         BNE     5$
 2323  007476  005726                         TST     (SP)+           ;;POP  <CR><LF> EQUIV
 2324  007500  104401                         TYPE                    ;;TYPE A CR AND LF
 2325  007502  001167                         $CRLF
 2326  007504  105037  007640                 CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
 2327  007510  000755                         BR      2$              ;;GET NEXT CHARACTER
 2328  007512  004737  007574         5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
 2329  007516  123726  001156         6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
 2330  007522  001350                         BNE     2$              ;;IF NO GO GET NEXT CHAR.
 2331  007524  013746  001154                 MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
 2332                                                                 ;;AND THE NULL CHAR.
 2333  007530  105366  000001         7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
 2334  007534  002770                         BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
 2335  007536  004737  007574                 JSR     PC,$TYPEC       ;;GO TYPE A NULL
 2336  007542  105337  007640                 DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
 2337  007546  000770                         BR      7$              ;;LOOP
 2338
 2339                                  ;HORIZONTAL TAB PROCESSOR
 2340
 2341  007550  112716  000040         8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
 2342  007554  004737  007574         9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
 2343  007560  132737  000007  007640         BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
 2344  007566  001372                         BNE     9$              ;;TAB STOP
 2345  007570  005726                         TST     (SP)+           ;;POP SPACE OFF STACK
 2346  007572  000724                         BR      2$              ;;GET NEXT CHARACTER
 2347  007574  105777  171350         $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
 2348  007600  100375                         BPL     $TYPEC
 2349  007602  116677  000002  171342         MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
 2350  007610  122766  000015  000002         CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
 2351  007616  001003                         BNE     1$              ;;BRANCH IF NO
 2352  007620  105037  007640                 CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
 2353  007624  000406                         BR      $TYPEX          ;;EXIT
 2354  007626  122766  000012  000002 1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
```

```
                                        BEQ     $TYPEX          ;;BRANCH IF YES
                                        INCB    (PC)+           ;;COUNT THE CHARACTER
                                $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
                                $TYPEX: RTS     PC


                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

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

007644  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
007650  116637  000001  010067          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
007656  112637  010071                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
007662  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
007666  000406                          BR      $TYPON
007670  112737  000001  010067  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
007676  112737  000006  010071          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
007704  112737  000005  010066  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
007712  010346                          MOV     R3,-(SP)        ;;SAVE R3
007714  010446                          MOV     R4,-(SP)        ;;SAVE R4
007716  010546                          MOV     R5,-(SP)        ;;SAVE R5
007720  113704  010071                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
007724  005404                          NEG     R4
007726  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
007732  110437  010070                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
007736  113704  010067                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
007742  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
007746  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
007750  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
007752  000404                          BR      3$              ;;GO DO MSB
007754  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
007756  006105                          ROL     R5
007760  006105                          ROL     R5
007762  010503                          MOV     R5,R3
007764  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
```

```
 2411  007766  105337  010070                 DECB    $OMODE          ;;TYPE THIS DIGIT?
 2412  007772  100016                         BPL     7$              ;;BR IF NO
 2413  007774  042703  177770                 BIC     #177770,R3      ;;GET RID OF JUNK
 2414  010000  001002                         BNE     4$              ;;TEST FOR 0
 2415  010002  005704                         TST     R4              ;;SUPPRESS THIS 0?
 2416  010004  001403                         BEQ     5$              ;;BR IF YES
 2417  010006  005204                 4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
 2418  010010  052703  000060                 BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
 2419  010014  052703  000040         5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
 2420  010020  110337  010064                 MOVB    R3,8$           ;;SAVE FOR TYPING
 2421  010024  104401  010064                 TYPE    ,8$             ;;GO TYPE THIS DIGIT
 2422  010030  105337  010066         7$:     DECB    $OCNT           ;;COUNT BY 1
 2423  010034  003347                         BGT     2$              ;;BR IF MORE TO DO
 2424  010036  002402                         BLT     6$              ;;BR IF DONE
 2425  010040  005204                         INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
 2426  010042  000744                         BR      2$              ;;GO DO THE LAST DIGIT
 2427  010044  012605                 6$:     MOV     (SP)+,R5        ;;RESTORE R5
 2428  010046  012604                         MOV     (SP)+,R4        ;;RESTORE R4
 2429  010050  012603                         MOV     (SP)+,R3        ;;RESTORE R3
 2430  010052  016666  000002  000004         MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
 2431  010060  012616                         MOV     (SP)+,(SP)
 2432  010062  000002                         RTI                     ;;RETURN
 2433  010064     000                 8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
 2434  010065     000                         .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
 2435  010066     000                 $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
 2436  010067     000                 $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
 2437  010070  000000                 $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
 2438
 2439                                 .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
 2440
 2441                                 ;;**************************************************************
 2442                                 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
 2443                                 ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
 2444                                 ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
 2445                                 ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
 2446                                 ;*REPLACED WITH SPACES.
 2447                                 ;*CALL:
 2448                                 ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
 2449                                 ;*      TYPDS                   ;;GO TO THE ROUTINE
 2450
 2451  010072                         $TYPDS:
 2452  010072  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 2453  010074  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 2454  010076  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 2455  010100  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 2456  010102  010546                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 2457  010104  012746  020200                 MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
 2458  010110  016605  000020                 MOV     20(SP),R5       ;;GET THE INPUT NUMBER
 2459  010114  100004                         BPL     1$              ;;BR IF INPUT IS POS.
 2460  010116  005405                         NEG     R5              ;;MAKE THE BINARY NUMBER POS.
 2461  010120  112766  000055  000001         MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
 2462  010126  005000                 1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
 2463  010130  012703  010306                 MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
 2464  010134  112723  000040                 MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
 2465  010140  005002                 2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
 2466  010142  016001  010276                 MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
```

```
2467  010146  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
2468  010150  002402                          BLT     4$              ;;BR IF DONE
2469  010152  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
2470  010154  000774                          BR      3$
2471  010156  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
2472  010160  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
2473  010162  001002                          BNE     5$              ;;FALL THROUGH IF 0
2474  010164  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
2475  010166  100407                          BMI     7$              ;;BR IF YES
2476  010170  106316                  5$:     ASLB    (SP)            ;;MSD?
2477  010172  103003                          BCC     6$              ;;BR IF NO
2478  010174  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
2479  010202  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
2480  010206  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
2481  010212  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
2482  010214  005720                          TST     (R0)+           ;;JUST INCREMENTING
2483  010216  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
2484  010222  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
2485  010224  003002                          BGT     8$              ;;GO TO EXIT
2486  010226  010502                          MOV     R5,R2           ;;GET THE LSD
2487  010230  000764                          BR      6$              ;;GO CHANGE TO ASCII
2488  010232  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
2489  010234  100003                          BPL     9$              ;;BR IF NO
2490  010236  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
2491  010244  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
2492  010246  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
2493  010250  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
2494  010252  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
2495  010254  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2496  010256  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
2497  010260  104401  010306                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
2498  010264  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
2499  010272  012616                          MOV     (SP)+,(SP)
2500  010274  000002                          RTI                     ;;RETURN TO USER
2501  010276  023420                  $DTBL:  10000.
2502  010300  001750                          1000.
2503  010302  000144                          100.
2504  010304  000012                          10.
2505  010306  000004                  $DBLK:  .BLKW   4
2506
2507                                  .SBTTL  TTY INPUT ROUTINE
2508
2509                                  ;;**********************************************************************
2510                                  .ENABL  LSB
2511  010316  000000                  $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
2512  010320  000000                  $TKQIN: .WORD   0               ;;INPUT POINTER
2513  010322  000000                  $TKQOUT: .WORD  0               ;;OUTPUT POINTER
2514  010324  000007                  $TKQSRT: .BLKB  7               ;;TTY KEYBOARD QUEUE
2515          010333                  $TKQEND=.
2516          010334                  .EVEN
2517
2518                                  ;*TK INITIALIZE ROUTINE
2519                                  ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
2520                                  ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
2521                                  ;
2522                                  ;*CALL:
```

```
 2523                                   ;*      JSR     PC,$TKINT
 2524                                   ;*      RETURN
 2525                                   ;
 2526  010334  005037  010316          $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
 2527  010340  012737  010324  010320          MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
 2528  010346  013737  010320  010322          MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
 2529  010354  012737  010404  000060          MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
 2530  010362  012737  000200  000062          MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
 2531  010370  005777  170552                  TST     @$TKB           ;;CLEAR DONE FLAG
 2532  010374  012777  000100  170542          MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
 2533  010402  000207                          RTS     PC              ;;RETURN TO CALLER
 2534
 2535                                   ;*TK SERVICE ROUTINE
 2536                                   ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
 2537                                   ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
 2538                                   ;*IT IN THE QUEUE.
 2539                                   ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
 2540                                   ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (@CNTLC)
 2541                                   ;
 2542  010404  117746  170536          $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
 2543  010410  042716  177600                  BIC     #^C177,(SP)     ;;STRIP THE JUNK
 2544  010414  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL C?
 2545  010420  001007                          BNE     1$              ;;BRANCH IF NO
 2546  010422  104401  011517                  TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
 2547  010426  004737  010334                  JSR     PC,$TKINT       ;;INIT THE KEYBOAPD
 2548  010432  005726                          TST     (SP)+           ;;CLEAN UP STACK
 2549  010434  000177  170640                  JMP     @CNTLC          ;;CONTROL C RESTART
 2550  010440  021627  000007          1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
 2551  010444  001004                          BNE     2$              ;;BRANCH IF NO
 2552  010446  022737  000176  001140          CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
 2553  010454  001500                          BEQ     6$              ;;GO TO SWR CHANGE
 2554
 2555  010456                          2$:
 2556  010456  022737  000007  010316          CMP     #7,$TKCNT       ;;IS THE QUEUE FULL?
 2557  010464  001004                          BNE     3$              ;;BRANCH IF NO
 2558  010466  104401  001162                  TYPE    ,$BELL          ;;RING THE TTY BELL
 2559  010472  005726                          TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
 2560  010474  000451                          BR      5$              ;;EXIT
 2561  010476  021627  000023          3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
 2562  010502  001021                          BNE     32$             ;;BRANCH IF NO
 2563  010504  005077  170434                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
 2564  010510  005726                          TST     (SP)+           ;;CLEAN CHAR OFF STACK
 2565  010512  105777  170426          31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
 2566  010516  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
 2567  010520  117746  170422                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
 2568  010524  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
 2569  010530  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
 2570  010534  001366                          BNE     31$             ;;BRANCH IF NO
 2571  010536  012777  000100  170400          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
 2572  010544  000002                          RTI                     ;;RETURN
 2573  010546  005237  010316          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
 2574  010552  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
 2575  010556  002405                          BLT     4$              ;;BRANCH IF YES
 2576  010560  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
 2577  010564  003002                          BGT     4$              ;;BRANCH IF YES
 2578  010566  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
```

```
 2579  010572  112677  177522          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
 2580  010576  005237  010320                  INC     $TKQIN          ;;UPDATE THE POINTER
 2581  010602  023727  010320  010333          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
 2582  010610  001003                          BNE     5$              ;;BRANCH IF NO
 2583  010612  012737  010324  010320          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
 2584  010620  000002                  5$:     RTI                     ;;RETURN
 2585
 2586                                          ;;**********************************************************************
 2587                                          ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
 2588                                          ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
 2589                                          ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
 2590                                          ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
 2591  010622  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
 2592  010630  001124                          BNE     15$             ;;EXIT IF NOT
 2593  010632  105777  170306                  TSTB    @$TKS           ;;IS A CHAR WAITING?
 2594  010636  100121                          BPL     15$             ;;IF NOT, EXIT
 2595  010640  117746  170302                  MOVB    @$TKB,-(SP)     ;;YES
 2596  010644  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
 2597  010650  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
 2598  010654  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
 2599                                                                  ;;AND EXIT
 2600
 2601                                          ;;**********************************************************************
 2602                                          ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
 2603                                          ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
 2604                                          ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
 2605  010656  123727  001134  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
 2606  010664  001674                          BEQ     2$              ;;BRANCH IF YES
 2607  010666  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
 2608  010670  004737  010334                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
 2609  010674  005077  170244                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
 2610  010700  112737  000001  001135          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
 2611
 2612  010706  104401  011531                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
 2613  010712  104401  011536          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
 2614  010716  013746  000176                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
 2615  010722  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 2616  010724  104401  011547                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
 2617  010730  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
 2618  010732  005046                          CLR     -(SP)           ;;THE NEW SWR
 2619  010734  105777  170204          7$:     TSTB    @$TKS           ;;CHAR THERE?
 2620  010740  100375                          BPL     7$              ;;IF NOT TRY AGAIN
 2621
 2622  010742  117746  170200                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
 2623  010746  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
 2624
 2625  010752  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL-C?
 2626  010756  001015                          BNE     9$              ;;BRANCH IF NOT
 2627  010760  104401  011517                  TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
 2628  010764  062706  000006                  ADD     #6,SP           ;;CLEAN UP STACK
 2629  010770  123727  001135  000001          CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
 2630  010776  001003                          BNE     8$              ;;BRANCH IF NO
 2631  011000  012777  000100  170136          MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
 2632  011006  000177  170266          8$:     JMP     @CNTLC          ;;CONTROL-C RESTART
 2633
 2634
```

```
 2635  011012  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
 2636  011016  001005                          BNE     10$             ;;BRANCH IF NOT
 2637  011020  104401  011524                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
 2638  011024  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
 2639  011030  000737                          BR      1^$             ;;LET'S TRY IT AGAIN
 2640
 2641
 2642  011032  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
 2643  011036  001022                          BNE     16$             ;;BRANCH IF NO
 2644  011040  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
 2645  011044  001403                          BEQ     11$             ;;BRANCH IF YES
 2646  011046  016677  000002  170064          MOV     2(SP),@SWR      ;;SAVE NEW SWR
 2647  011054  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
 2648  011060  104401  001167          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
 2649  011064  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
 2650  011072  001003                          BNE     15$             ;;BRANCH IF NOT
 2651  011074  012777  000100  170042          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
 2652  011102  000002          15$:    RTI                             ;;RETURN
 2653  011104  004737  007574          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
 2654  011110  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
 2655  011114  002420                          BLT     18$             ;;BRANCH IF YES
 2656  011116  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
 2657  011122  003015                          BGT     18$             ;;BRANCH IF YES
 2658  011124  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
 2659  011130  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
 2660  011134  001403                          BEQ     17$             ;;BRANCH IF YES
 2661  011136  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
 2662  011140  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
 2663  011142  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
 2664  011144  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
 2665  011150  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
 2666  011154  000667                          BR      7$              ;;GET THE NEXT ONE
 2667  011156  104401  001166          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
 2668  011162  000720                          BR      20$             ;;SIMULATE CONTROL-U
 2669                                  .DSABL  LSB
 2670
 2671
 2672                                  ;;**************************************************************************
 2673                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
 2674                                  ;*CALL:
 2675                                  ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
 2676                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
 2677                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
 2678                                  ;
 2679
 2680  011164  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
 2681  011166  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE PS
 2682  011174  005066  000004                  CLR     4(SP)           ;;GET READY FOR A CHARACTER
 2683  011200  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
 2684  011202  012746  011210                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
 2685  011206  000002                          RTI                     ;;POP NEW PC AND PS
 2686  011210                          64$:
 2687  011210  005737  010316          1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
 2688  011214  001775                          BEQ     1$
 2689  011216  005337  010316                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
 2690  011222  117766  177074  000004          MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
```

```
 2691  011230  005237  010322                  INC     $TKQOUT          ;;UPDATE THE POINTER
 2692  011234  023727  010322  010333          CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
 2693  011242  001003                          BNE     2$               ;;BRANCH IF NO
 2694  011244  012737  010324  010322          MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
 2695  011252  000002                  2$:     RTI                      ;;RETURN
 2696                                  ;;*************************************************************
 2697                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
 2698                                  ;*CALL:
 2699                                  ;*      RDLIN                    ;;INPUT A STRING FROM THE TTY
 2700                                  ;*      RETURN HERE              ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
 2701                                  ;*                               ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
 2702
 2703  011254  010346                  $RDLIN: MOV     R3,-(SP)         ;;SAVE R3
 2704  011256  005046                          CLR     -(SP)            ;;CLEAR THE RUBOUT KEY
 2705  011260  012703  011510          1$:     MOV     #$TTYIN,R3       ;;GET ADDRESS
 2706  011264  022703  011517          2$:     CMP     #$TTYIN+7,R3     ;;BUFFER FULL?
 2707  011270  101456                          BLOS    4$               ;;BR IF YES
 2708  011272  104410                          RDCHR                    ;;GO READ ONE CHARACTER FROM THE TTY
 2709  011274  112613                          MOVB    (SP)+,(R3)       ;;GET CHARACTER
 2710  011276  122713  000177          10$:    CMPB    #177,(R3)        ;;IS IT A RUBOUT
 2711  011302  001022                          BNE     5$               ;;BR IF NO
 2712  011304  005716                          TST     (SP)             ;;IS THIS THE FIRST RUBOUT?
 2713  011306  001007                          BNE     6$               ;;BR IF NO
 2714  011310  112737  000134  011506          MOVB    #'\,9$           ;;TYPE A BACK SLASH
 2715  011316  104401  011506                  TYPE    ,9$
 2716  011322  012716  177777                  MOV     #-1,(SP)         ;;SET THE RUBOUT KEY
 2717  011326  005303                  6$:     DEC     R3               ;;BACKUP BY ONE
 2718  011330  020327  011510                  CMP     R3,#$TTYIN       ;;STACK EMPTY?
 2719  011334  103434                          BLO     4$               ;;BR IF YES
 2720  011336  111337  011506                  MOVB    (R3),9$          ;;SETUP TO TYPEOUT THE DELETED CHAR.
 2721  011342  104401  011506                  TYPE    ,9$              ;;GO TYPE
 2722  011346  000746                          BR      2$               ;;GO READ ANOTHER CHAR.
 2723  011350  005716                  5$:     TST     (SP)             ;;RUBOUT KEY SET?
 2724  011352  001406                          BEQ     7$               ;;BR IF NO
 2725  011354  112737  000134  011506          MOVB    #'\,9$           ;;TYPE A BACK SLASH
 2726  011362  104401  011506                  TYPE    ,9$
 2727  011366  005016                          CLR     (SP)             ;;CLEAR THE RUBOUT KEY
 2728  011370  122713  000025          7$:     CMPB    #25,(R3)         ;;IS CHARACTER A CTRL U?
 2729  011374  001003                          BNE     8$               ;;BR IF NO
 2730  011376  104401  011524                  TYPE    ,$CNTLU          ;;TYPE A CONTROL 'U''
 2731  011402  000726                          BR      1$               ;;GO START OVER
 2732  011404  122713  000022          8$:     CMPB    #22,(R3)         ;;IS CHARACTER A ''^R''?
 2733  011410  001011                          BNE     3$               ;;BRANCH IF NO
 2734  011412  105013                          CLRB    (R3)             ;;CLEAR THE CHARACTER
 2735  011414  104401  001167                  TYPE    ,$CRLF           ;;TYPE A ''CR'' & 'LF''
 2736  011420  104401  011510                  TYPE    ,$TTYIN          ;;TYPE THE INPUT STRING
 2737  011424  000717                          BR      2$               ;;GO PICKUP ANOTHER CHACTER
 2738  011426  104401  001166          4$:     TYPE    ,$QUES           ;;TYPE A '?'
 2739  011432  000712                          BR      1$               ;;CLEAR THE BUFFER AND LOOP
 2740  011434  111337  011506          3$:     MOVB    (R3),9$          ;;ECHO THE CHARACTER
 2741  011440  104401  011506                  TYPE    ,9$
 2742  011444  122723  000015                  CMPB    #15,(R3)+        ;;CHECK FOR RETURN
 2743  011450  001305                          BNE     2$               ;;LOOP IF NOT RETURN
 2744  011452  105063  177777                  CLRB    -1(R3)           ;;CLEAR RETURN (THE 15)
 2745  011456  104401  001170                  TYPE    ,$LF             ;;TYPE A LINE FEED
 2746  011462  005726                          TST     (SP)+            ;;CLEAN RUBOUT KEY FROM THE STACK
```

```
 2747  011464  012603                          MOV     (SP)+,R3        ;;RESTORE R3
 2748  011466  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
 2749  011470  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
 2750  011476  012766  011510  000004          MOV     #$TTYIN,4(SP)
 2751  011504  000002                          RTI                     ;;RETURN
 2752  011506     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
 2753  011507     000                          .BYTE   0               ;;TERMINATOR
 2754  011510  000007                  $TTYIN: .BLKB   7               ;;RESERVE 7 BYTES FOR TTY INPUT
 2755  011517     136  006503  000012  $CNTLC: .ASCIZ  /^C/<15><12>    ;;CONTROL ''C''
 2756  011524  052536  005015     000  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL 'U''
 2757  011531     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL 'G''
 2758  011536  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
 2759  011544  020075     000
 2760  011547     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
 2761  011554  036440  000040
 2762
 2763                                  .SBTTL  TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS
 2764
 2765                                  ;;**************************************************************
 2766                                  ;*THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
 2767                                  ;*LEADING NUMBERS.
 2768                                  ;*CALL
 2769                                  ;*      MOV     #NUMADR,-(SP)   ;;FIRST ADDRESS OF ASCIZ STRING
 2770                                  ;*      JSR     PC,@#$SUPRS
 2771
 2772
 2773  011560  010046                  $SUPRS: MOV     R0,-(SP)        ;;SAVE R0
 2774  011562  016600  000004                  MOV     4(SP),R0        ;;PICKUP THE POINTER
 2775  011566  105710                  1$:     TSTB    (R0)            ;;TERMINATEOR?
 2776  011570  001403                          BEQ     2$              ;;BR IF YES
 2777  011572  122720  000060                  CMPB    #'0,(R0)+       ;;IS THIS AN ASCII '0' ?
 2778  011576  001773                          BEQ     1$              ;;BR IF YES
 2779  011600  005300                  2$:     DEC     R0              ;;BACKUP BY '1'
 2780  011602  010037  011610                  MOV     R0,3$           ;;SAVE FOR TYPING
 2781  011606  104401                          TYPE                    ;;GO TYPE
 2782  011610  000000                  3$:     .WORD   0               ;;ASCIZ POINTER GOES HERE
 2783  011612  012600                          MOV     (SP)+,R0        ;;RESTORE R0
 2784  011614  012616                          MOV     (SP)+,(SP)      ;;RESTORE THE STACK
 2785  011616  000207                          RTS     PC              ;;RETURN
 2786
 2787                                  .SBTTL  SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE
 2788
 2789                                  ;;**************************************************************
 2790                                  ;*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
 2791                                  ;*UNSIGNED DECIMAL ASCIZ NUMBER.
 2792                                  ;*CALL
 2793                                  ;*      MOV     NUMBER,-(SP)    ;;PUT BINARY NUMBER ON THE STACK
 2794                                  ;*      JSR     PC,@#$SB2D      ;;CALL
 2795                                  ;*      RETURN                  ;;ADDRESS OF THE 1ST ASCIZ CHAR.IS ON THE STACK
 2796
 2797
 2798  011620  016637  000002  011650  $SB2D:  MOV     2(SP),1$        ;;SAVE BINARY NUMBER
 2799  011626  012746  011650                  MOV     #1$,-(SP)       ;;SET POINTER
 2800  011632  004737  011654                  JSR     PC,@#$DB2D      ;;CALL DOUBLE LENGTH CONVERT
 2801  011636  062716  000005                  ADD     #5,(SP)         ;;ONLY ALLOW FIVE CHARACTERS
 2802  011642  012666  000002                  MOV     (SP)+,2(SP)     ;;PICKUP POINTER
```

```
 2803  011646  000207                          RTS     PC              ;;RETURN
 2804  011650  000000  000000          1$:     .WORD   0,0
 2805
 2806                                  .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
 2807
 2808                                  ;;*****************************************************************
 2809                                  ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
 2810                                  ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
 2811                                  ;*POSITIVE.
 2812                                  ;*CALL
 2813                                  ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
 2814                                  ;*      JSR     PC,@#$DB2D
 2815                                  ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
 2816                                                                  ;;IS ON THE STACK
 2817
 2818
 2819  011654  104412                  $DB2D:  SAVREG                  ;;SAVE REGISTERS
 2820  011656  016602  000002                  MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
 2821  011662  012700  012034                  MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
 2822  011666  010066  000002                  MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
 2823  011672  012201                          MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
 2824  011674  012202                          MOV     (R2)+,R2
 2825  011676  012737  000012 011752          MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
 2826  011704  012704  011764                  MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
 2827  011710  012705  011766                  MOV     #$TNPWR+2,R5
 2828  011714  005003                  1$:     CLR     R3              ;;CLEAR PARTIAL
 2829  011716  161401                  2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
 2830  011720  005602                          SBC     R2
 2831  011722  161502                          SUB     (R5),R2
 2832  011724  002402                          BLT     3$              ;;BR IF TEN POWER TO LARGE
 2833  011726  005203                          INC     R3              ;;ADD 1 TO PARTIAL
 2834  011730  000772                          BR      2$              ;;LOOP
 2835  011732  062401                  3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
 2836  011734  005502                          ADC     R2
 2837  011736  062402                          ADD     (R4)+,R2
 2838  011740  022525                          CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
 2839  011742  052703  000060                  BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
 2840  011746  110320                          MOVB    R3,(R0)+        ;;SAVE IT
 2841  011750  005327                          DEC     (PC)+           ;;DONE?
 2842  011752  000000                  4$:     .WORD   0
 2843  011754  001357                          BNE     1$              ;;BR IF NO
 2844  011756  105020                          CLRB    (R0)+           ;;TERMINATOR
 2845  011760  104413                          RESREG                  ;;RESTORE REGISTERS
 2846  011762  000207                          RTS     PC              ;;RETURN
 2847  011764  145000                  $TNPWR: 145000                  ;;1.0E09
 2848  011766  035632                          35632
 2849  011770  160400                          160400                  ;;1.0E08
 2850  011772  002765                          2765
 2851  011774  113200                          113200                  ;;1.0E07
 2852  011776  000230                          230
 2853  012000  041100                          041100                  ;;1.0E06
 2854  012002  000017                          17
 2855  012004  103240                          103240                  ;;1.0E05
 2856  012006  000001                          1
 2857  012010  023420                          23420                   ;;1.0E04
 2858  012012  000000                          0
```

```
 2859  012014  001750                          1750                    ;;1.0E03
 2860  012016  000000                          0
 2861  012020  000144                          144                     ;;1.0E02
 2862  012022  000000                          0
 2863  012024  000012                          12                      ;;1.0E01
 2864  012026  000000                          0
 2865  012030  000001                          1                       ;;1.0E00
 2866  012032  000000                          0
 2867  012034  000014                $DECVL: .BLKB   12.               ;;RESERVE STORAGE FOR ASCIZ STRING
 2868
 2869                                .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
 2870
 2871                                ;;*****************************************************************
 2872                                ;*SAVE R0-R5
 2873                                ;*CALL:
 2874                                ;*      SAVREG
 2875                                ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
 2876                                ;*
 2877                                ;*TOP---(+16)
 2878                                ;* +2---(+18)
 2879                                ;* +4---R5
 2880                                ;* +6---R4
 2881                                ;* +8---R3
 2882                                ;*+10---R2
 2883                                ;*+12---R1
 2884                                ;*+14---R0
 2885
 2886  012050                        $SAVREG:
 2887  012050  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 2888  012052  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 2889  012054  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 2890  012056  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 2891  012060  010446                         MOV     R4,-(SP)        ;;PUSH R4 ON STACK
 2892  012062  010546                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 2893  012064  016646  000022                 MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
 2894  012070  016646  000022                 MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
 2895  012074  016646  000022                 MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
 2896  012100  016646  000022                 MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
 2897  012104  000002                         RTI
 2898
 2899                                ;*RESTORE R0-R5
 2900                                ;*CALL:
 2901                                ;*      RESREG
 2902  012106                        $RESREG:
 2903  012106  012666  000022                 MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
 2904  012112  012666  000022                 MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
 2905  012116  012666  000022                 MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
 2906  012122  012666  000022                 MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
 2907  012126  012605                         MOV     (SP)+,R5        ;;POP STACK INTO R5
 2908  012130  012604                         MOV     (SP)+,R4        ;;POP STACK INTO R4
 2909  012132  012603                         MOV     (SP)+,R3        ;;POP STACK INTO R3
 2910  012134  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
 2911  012136  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
 2912  012140  012600                         MOV     (SP)+,R0        ;;POP STACK INTO R0
 2913  012142  000002                         RTI
 2914
```

```
                                .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.

012144  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
012146  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
012152  005740                          TST     -(R0)           ;;BACKUP BY 2
012154  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
012156  006300                          ASL     R0              ;;POSITION FOR INDEXING
012160  016000  012200                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
012164  000200                          RTS     R0              ;;GO TO ROUTINE

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

012166  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
012170  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
012176  000002                          RTI                     ;;RESTORE THE PSW

                                .SBTTL  TRAP TABLE

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

                                ;       ROUTINE
                                ;       -------
012200  012166                  $TRPAD: .WORD   $TRAP2
012202  007424                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
012204  007670                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
012206  007644                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
012210  007704                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
012212  010072                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

012214  010712                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

012216  010622                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
012220  011164                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
012222  011254                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
012224  012050                          $SAVREG ;;CALL=SAVREG   TRAP+12(104412) SAVE R0-R5 ROUTINE
012226  012106                          $RESREG ;;CALL=RESREG   TRAP+13(104413) RESTORE R0-R5 ROUTINE


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

                                ;COPYRIGHT (C) 1976
                                ;DIGITAL EQUIPMENT CORP.
                                ;MAYNARD, MA 01754
                                ;AUTHOR(S): JIM LACEY/CHUCK HESS

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

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

012230  000000  000000  000000  RPERRS: .WORD   0,0,0,0
012236  000000

                                        ;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

012240      000                 DRVACT: .BYTE   0               ;DRIVE 0
012241      000                         .BYTE   0               ;DRIVE 1
012242      000                         .BYTE   0               ;DRIVE 2
012243      000                         .BYTE   0               ;DRIVE 3
012244      000                         .BYTE   0               ;DRIVE 4
012245      000                         .BYTE   0               ;DRIVE 5
012246      000                         .BYTE   0               ;DRIVE 6
012247      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

012250      000                 DRVSTA: .BYTE   0               ;DRIVE 0
012251      000                         .BYTE   0               ;DRIVE 1
012252      000                         .BYTE   0               ;DRIVE 2
012253      000                         .BYTE   0               ;DRIVE 3
012254      000                         .BYTE   0               ;DRIVE 4
012255      000                         .BYTE   0               ;DRIVE 5
012256      000                         .BYTE   0               ;DRIVE 6
012257      000                         .BYTE   0               ;DRIVE 7

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

012260      000                 DRVTYP: .BYTE   0               ;DRIVE 0
012261      000                         .BYTE   0               ;DRIVE 1
012262      000                         .BYTE   0               ;DRIVE 2
012263      000                         .BYTE   0               ;DRIVE 3
012264      000                         .BYTE   0               ;DRIVE 4
012265      000                         .BYTE   0               ;DRIVE 5
012266      000                         .BYTE   0               ;DRIVE 6
012267      000                         .BYTE   0               ;DRIVE 7

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

```
3027
3028  C  270     000                 DPINT:  .BYTE   0               ;DRIVE 0
3029  012271     000                         .BYTE   0               ;DRIVE 1
3030  012272     000                         .BYTE   0               ;DRIVE 2
3031  012273     000                         .BYTE   0               ;DRIVE 3
3032  012274     000                         .BYTE   0               ;DRIVE 4
3033  012275     000                         .BYTE   0               ;DRIVE 5
3034  012276     000                         .BYTE   0               ;DRIVE 6
3035  012277     000                         .BYTE   0               ;DRIVE 7
3036
3037                                 ;TABLE OF PENDING DUAL PORT REQUESTS
3038                                         ;DPRQS=0 IF THAT A DUAL PORT REQUEST IS NOT PENDING FOR THAT DRIVE
3039                                         ;DPRQS<0 IF THAT A DUAL PORT REQUEST IS PENDING FOR THAT DRIVE
3040
3041  012300     000                 DPRQS:  .BYTE   0               ;DRIVE 0
3042  012301     000                         .BYTE   0               ;DRIVE 1
3043  012302     000                         .BYTE   0               ;DRIVE 2
3044  012303     000                         .BYTE   0               ;DRIVE 3
3045  012304     000                         .BYTE   0               ;DRIVE 4
3046  012305     000                         .BYTE   0               ;DRIVE 5
3047  012306     000                         .BYTE   0               ;DRIVE 6
3048  012307     000                         .BYTE   0               ;DRIVE 7
3049
3050                                 ;TRANSFER WAIT FLAG (TRNSWT=1 WORD)
3051                                         ;THIS IS A ONE WORD QUEUE. IT WILL CONTAIN THE ADDRESS OF
3052                                         ;''DPB'' OF THE I/O OPERATION.
3053
3054  012310  000000                 TRNSWT: .WORD   0
3055
3056                                 ;SEARCH WAIT KEYS (SRCHWT=1 WORD)
3057                                         ;THIS IS A ONE WORD QUEUE THAT WILL CONTAIN A KEY FOR EACH OF
3058                                         ;THE DRIVES THAT ARE PERFORMING A SEARCH COMMAND FOR THE I/O
3059                                         ;REQUEST THAT IS AT THE TOP OF THEIR REQUEST QUEUE.
3060                                         ;EACH DRIVE IS ASSIGNED ONE BIT, STARTING AT BIT00 FOR DRIVE 0.
3061
3062  012312  000000                 SRCHWT: .WORD   0
3063
3064                                 ;RP04/5/6 DRIVER ACTIVE FLAG (ACTDRV=1 BYTE)
3065                                         ;ACTDRV=0 IF DRIVER IS INACTIVE
3066                                         ;ACTDRV>0 IF DRIVER IS ACTIVE
3067
3068  012314     000                 ACTDRV: .BYTE   0
3069
3070                                 ;SOFTWARE TIMER ROUTINE ACTIVE FLAG (ACTSTR=1 BYTE)
3071                                         ;ACTSTR=0 IF SOFTWARE TIMER ROUTINE IS INACTIVE
3072                                         ;ACTSTR>0 IF SOFTWARE TIMER ROUTINE IS ACTIVE
3073
3074  012315     000                 ACTSTR: .BYTE   0
3075
3076                                 ;UNLOAD FLAG (ULDFLG=8 BYTES)
3077                                         ;ULDFLG=0 IF NO UNLOAD COMMAND
3078                                         ;ULDFLG>0 IF UNLOAD COMMAND IN PROGRESS
3079                                         ;ULDFLG<0 IF UNLOAD COMMAND IN WAIT QUEUE
3080
3081  012316     000                 ULDFLG: .BYTE   0               ;DRIVE 0
3082  012317     000                         .BYTE   0               ;DRIVE 1
```

```
 3083  012320  000                     .BYTE   0               ;DRIVE 2
 3084  012321  000                     .BYTE   0               ;DRIVE 3
 3085  012322  000                     .BYTE   0               ;DRIVE 4
 3086  012323  000                     .BYTE   0               ;DRIVE 5
 3087  012324  000                     .BYTE   0               ;DRIVE 6
 3088  012325  000                     .BYTE   0               ;DRIVE 7
 3089
 3090                                ;LOOK AHEAD COUNT (LACNT=8 BYTES)
 3091                                        ;LACNT WILL INDICATE THE NUMBER OF LOOK AHEADS PERFORMED
 3092
 3093  012326  000                   LACNT:  .BYTE   0               ;DRIVE 0
 3094  012327  000                           .BYTE   0               ;DRIVE 1
 3095  012330  000                           .BYTE   0               ;DRIVE 2
 3096  012331  000                           .BYTE   0               ;DRIVE 3
 3097  012332  000                           .BYTE   0               ;DRIVE 4
 3098  012333  000                           .BYTE   0               ;DRIVE 5
 3099  012334  000                           .BYTE   0               ;DRIVE 6
 3100  012335  000                           .BYTE   0               ;DRIVE 7
 3101
 3102                                ;SAVE REGISTERS FLAG (SAVEFG =1 WORD)
 3103                                        ;SAVEFG <0 IF SAVE THE RH11/RP04/5/6 REGISTERS WHEN THE
 3104                                        ;OPERATION IS COMPLETED AS PER (DPB+14).
 3105                                        ;SAVEFG=0 IF SAVE THE RH11/RP04/5/6 REGISTERS, AS PER
 3106                                        ;(DPB+14), AFTER AN ERROR.
 3107
 3108  012336  000000                SAVEFG: .WORD   0
 3109
 3110                                ;SEEK FLAG (SEEKFG=1 WORD)
 3111                                        ;SEEKFG=0 IF WHEN THE DISK ADDRESS ISN'T IN THE WINDOW
 3112                                        ;FOR A DATA TRANSFER START A SEARCH COMMAND
 3113                                        ;SEEKFG<0 IF DATA TRANSFER WILL DO IMPLIED SEEKS,
 3114                                        ;DISREGARD THE WINDOW
 3115
 3116  012340  000000                SEEKFG: .WORD   0
 3117
 3118                                ;TIMEOUT TABLE (TIMER=8 WORDS)
 3119                                        ;THIS TABLE CONTAINS THE TIME ALLOWED FOR AN OPERATION
 3120
 3121  012342  177777                TIMER:  .WORD   -1      ;DRIVE 0
 3122  012344  177777                        .WORD   -1      ;DRIVE 1
 3123  012346  177777                        .WORD   -1      ;DRIVE 2
 3124  012350  177777                        .WORD   -1      ;DRIVE 3
 3125  012352  177777                        .WORD   -1      ;DRIVE 4
 3126  012354  177777                        .WORD   -1      ;DRIVE 5
 3127  012356  177777                        .WORD   -1      ;DRIVE 6
 3128  012360  177777                        .WORD   -1      ;DRIVE 7
 3129
 3130                                ;DATA TRANSFER UNDERWAY INDICATOR (DTUW=1 WORD)
 3131                                        ;DTUW<0 IF NO DATA TRANSFER UNDERWAY
 3132                                        ;DTUW=+N (WHERE N=0 TO 7) IMPLIES DATA TRANSFER UNDERWAY ON DRIVE N
 3133
 3134  012362  177777                DTUW:   .WORD   -1
 3135
 3136                                ;ATTENTION BITS TABLE (ATABIT=8 BYTES)
 3137                                        ;THIS TABLE CONTAINS THE CORRESPONDING BIT TO EACH DRIVES
 3138                                        ;ATTENTION BIT
```

```
                                ATABIT: .BYTE   1       ;DRIVE 0
                                        .BYTE   2       ;DRIVE 1
                                        .BYTE   4       ;DRIVE 2
                                        .BYTE   10      ;DRIVE 3
                                        .BYTE   20      ;DRIVE 4
                                        .BYTE   40      ;DRIVE 5
                                        .BYTE   100     ;DRIVE 6
                                        .BYTE   200     ;DRIVE 7

                                ;RP04/5/6 TO RH11 'MASSBUS CONTROL BUS PARITY ERRORS'' (MCPE) ALLOWED BEFORE
                                ;CALLING IT FATAL (MCPEMX=1 WORD)

012374  000003                 MCPEMX: .WORD   3

                                ;STORAGE FOR RPADR (THE FIRST ADDRESS (776700) OF THE RH11/RP04/5/6),
                                ;RPVEC (THE VECTOR ADDRESS (254)), AND RPVEC+2 (THE BR LEVEL (5)).

012376  176700                  RPADR:  .WORD   176700
012400  000254  000240          RPVEC:  .WORD   254,5*32.

                                ;MAXIMUM NUMBER OF LOOK AHEADS ALLOWED IS 4 (MXLACT=1 WORD)

012404  000004                  MXLACT: .WORD   4
                                ;MAXIMUM DELTA DELAY IS 8 SECTORS (MXDLTA=1 WORD)

012406  001000                  MXDLTA: .WORD   8.*64.
                                ;MINIMUM DELTA DELAY IS 2 SECTORS (MNDLTA=1 WORD)

012410  000200                  MNDLTA: .WORD   2*64.
                                ;MAXIMUM SEARCH FOR I/O WINDOW IS 5 SECTORS (MXWNDW=1 WORD)

012412  000005                  MXWNDW: .WORD   5

                                ;DEFINITIONS OF THE RH11/RP04/5/6 ADDRESS INDEXES

        000000                  RPCS1=0                 ;CONTROL AND STATUS REGISTER #1 (DRIVE REG. 00)
        000002                  RPWC=2                  ;WORD COUNT REGISTER (NOT A DRIVE REG)
        000004                  RPBA=4                  ;UNIBUS ADDRESS REGISTER (NOT A DRIVE REG)
        000006                  RPDA=6                  ;DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG. 05)
        000010                  RPCS2=10                ;CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG)
        000012                  RPDS1=12                ;DRIVE STATUS REGISTER (DRIVE REG 01)
        000014                  RPER1=14                ;ERROR REGISTER #1 (DRIVE REG. 02)
        000016                  RPAS=16                 ;ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG. 04)
        000020                  RPLA=20                 ;LOOK AHEAD REGISTER (DRIVE REG. 07)
        000022                  RPDB=22                 ;DATA BUFFER REGISTER (NOT A DRIVE REG.)
        000024                  RPMR=24                 ;MAINTAINABILITY REGISTER (DRIVE REG. 03)
        000026                  RPDT=26                 ;DRIVE TYPE REGISTER (DRIVE REG. 06)
        000030                  RPSN=30                 ;SERIAL NUMBER REGISTER (DRIVE REG. 10)
        000032                  RPOF=32                 ;OFFSET REGISTER (DRIVE REG. 11)
        000034                  RPCA=34                 ;DESIRED CYLINDER ADDRESS REGISTER (DRIVE REG. 12)
        000036                  RPCC=36                 ;CURRENT CYLINDER ADDRESS REGISTER (DRIVE REG. 13)
        000040                  RPER2=40                ;ERROR REGISTER #2 (DRIVE REG. 14)
        000042                  RPER3=42                ;ERROR REGISTER #3 (DRIVE REG. 15)
        000044                  RPEC1=44                ;ECC POSITION REGISTER (DRIVE REG. 16)
        000046                  RPEC2=46                ;ECC PATTERN REGISTER (DRIVE REG.  17)
```

```
                                        ;RH11/RP04/5/6 DRIVER INITIALIZATION CODE
                                                ;THIS ROUTINE WILL DETERMINE WHICH RP04/5/6 DRIVES ARE
                                                ;AVAILABLE FOR TESTING AND SET THE DRVSTA INDICATOR
                                                ;TO THE PROPER STATE FOR EACH DRIVE.
                                                ;NOTE: THIS ROUTINE CALLS DRVINT
                                        ;
                                        ;CALL
                                        ;
                                        ;       JSR     PC,RPINIT
                                        ;       RETURN
                                        ;
                                        ;NOTE: THE 'P' OR 'L' CLOCK MUST BE STARTED
                                        ;
012414  104412                  RPINIT: SAVREG                          ;SAVE R0 - R5
012416  013746  177776                  MOV     @#PS,-(SP)              ;SAVE THE PRESENT PROCESSOR STATUS
012422  012737  000240  177776          MOV     #<5*32.>,@#PS           ;CHANGE THE PRIORITY TO 5
012430  004737  020400                  JSR     PC,CLRQUE               ;CLEAR ALL REQUEST QUEUES
012434  012701  012230                  MOV     #RPERRS,R1              ;FIRST ADDRESS TO BE CLEARED
012440  012702  012340                  MOV     #SEEKFG,R2              ;LAST ADDRESS TO BE CLEARED
012444  005021                  1$:     CLR     (R1)+                   ;CLEAR
012446  020102                          CMP     R1,R2                   ;ARE WE DONE?
012450  101775                          BLOS    1$                      ;BRANCH IF NO
012452  012702  012362                  MOV     #DTUW,R2                ;LAST ADDRESS
012456  012721  177777          2$:     MOV     #-1,(R1)+               ;INITIALIZE
012462  020102                          CMP     R1,R2                   ;DONE?
012464  101774                          BLOS    2$                      ;LOOP IF NO
012466  005037  012250                  CLR     DRVSTA                  ;SET ALL DRIVES TO OFFLINE
012472  005037  012252                  CLR     DRVSTA+2
012476  005037  012254                  CLR     DRVSTA+4
012502  005037  012256                  CLR     DRVSTA+6
012506  013703  012400                  MOV     RPVEC,R3                ;SETUP THE RH11/RP04/5/6 VECTOR
012512  012723  015260                  MOV     #ISR,(R3)+
012516  013713  012402                  MOV     RPVEC+2,(R3)
012522  013704  012376                  MOV     RPADR,R4                ;FIRST ADDRESS OF RH11/RP04
012526  012764  000040  000010          MOV     #BIT05,RPCS2(R4)        ;MASSBUS INIT
012534  005001                          CLR     R1                      ;START WITH DRIVE 0
012536  004037  012626          3$:     JSR     R0,DRVINT               ;INIT THE DRIVE
012542  000401                          BR      4$                      ;'DVA' NOT SET OR PARITY ERROR
012544  000402                          BR      5$                      ;NORMAL RETURN
012546  105061  012250          4$:     CLRB    DRVSTA(R1)              ;SET DRIVE STATUS TO OFFLINE
012552  005201                  5$:     INC     R1                      ;GO TO NEXT DRIVE
012554  042701  177770                  BIC     #^C7,R1                 ;MASK OUT UNUSED BITS
012560  001366                          BNE     3$                      ;BR IF MORE DRIVES TO GO
012562  012701  000007                  MOV     #7,R1                   ;START WITH DRIVE 7
012566  005037  177776                  CLR     @#PS                    ;CLEAR THE PROCESSOR STATUS
012572  105761  012270          6$:     TSTB    DPINT(R1)               ;WAITING FOR DRIVE TO SWITCH PORTS ?
012576  001405                          BEQ     8$                      ;BR NOT WAITING
012600  004737  020034                  JSR     PC,SET.IE               ;SET INTERRUPT
012604  105761  012270          7$:     TSTB    DPINT(R1)               ;DRIVE SWITCHED PORTS ?
012610  001375                          BNE     7$                      ;BR IF NOT
012612  005301                  8$:     DEC     R1                      ;GO TO THE NEXT DRIVE
012614  100366                          BPL     6$                      ;CHECK NEXT DRIVE
012616  012637  177776                  MOV     (SP)+,@#PS              ;RESTORE THE PROCESSOR STATUS
012622  104413                          RESREG                          ;RESTORE R0 - R5
012624  000207                          RTS     PC                      ;BYE-BYE
```

```
3251
3252                                    ;DRIVE INITILIZATION ROUTINE
3253                                            ;THIS ROUTINE DETERMINES IF A DRIVE EXIST AND IF IT IS
3254                                            ;AN RP04/5/6. IF IT IS, A 'READ-IN PRESET' IS ISSUED AND FMT22
3255                                            ;IS SET TO A '1'. THEN MOL, DPR, DRY, AND VV ARE CHECKED TO
3256                                            ;INSURE THEY ARE ALL ON A '1'. AND DEPENDING ON THEIR STATE,
3257                                            ;DRVSTA IS SET TO THE PROPER CONDITION.
3258                                    ;CALL
3259                                    ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
3260                                    ;       MOV     RPADR,R4        ;UNIBUS ADDRESS OF RH11/RP04/5/6 (RPCS1)
3261                                    ;       JSR     R0,DRVINT       ;CALLED BY A JSR
3262                                    ;       RETURN1                 ;ERROR OCCURRED (PARITY)
3263                                    ;       RETURN2                 ;NORMAL RETURN
3264                                    ;
3265
3266  012626  010546                    DRVINT: MOV     R5,-(SP)        ;SAVE R5
3267  012630  105061  012250                    CLRB    DRVSTA(R1)      ;START DRIVE STATUS AS OFFLINE
3268  012634  105061  012260                    CLRB    DRVTYP(R1)      ;CLEAR THE DRIVE TYPE INDICATOR
3269  012640  105061  012316                    CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD FLAG
3270  012644  010164  000010                    MOV     R1,RPCS2(R4)    ;SELECT A DRIVE
3271  012650  112764  000111  000000          MOVB    #111,RPCS1(R4)  ;DO A DRIVE CLEAR COMMAND (& SEIZE DRIVE)
3272  012656  032764  010000  000010          BIT     #BIT12,RPCS2(R4) ;NONEXISTENT DRIVE?
3273  012664  001403                          BEQ     1$              ;NO---BRANCH
3274  012666  004737  020034                  JSR     PC,SET.IE       ;GO SET "IE" WITHOUT A "TRE"
3275  012672  000520                          BR      6$              ;LEAVE THIS ROUTINE
3276  012674  105061  012250          1$:     CLRB    DRVSTA(R1)      ;SET DRIVE STATUS TO OFFLINE
3277  012700  032764  004000  000000          BIT     #BIT11,RPCS1(R4)  ;SEE IF DRIVE AVAILABLE
3278  012706  001514                          BEQ     7$              ;BR IF DRIVE NOT AVAILABLE
3279  012710  004037  017344                  JSR     R0,RD.RP        ;READ THE DRIVE TYPE REG.
3280  012714  000026                          RPDT
3281  012716  013160                          8$                      ;ERROR RETURN ADDRESS
3282  012720  012605                          MOV     (SP)+,R5        ;PUT DRIVE TYPE IN R5
3283  012722  112761  000001  012260          MOVB    #1,DRVTYP(R1)   ;SET RP04 INDICATOR
3284  012730  022705  020020                  CMP     #20020,R5       ;IS IT A SINGLE PORT RP04?
3285  012734  001431                          BEQ     2$              ;BRANCH IF YES
3286  012736  022705  024020                  CMP     #24020,R5       ;IS IT A DUAL PORT RP04?
3287  012742  001426                          BEQ     2$              ;BR IF YES
3288  012744  112761  000002  012260          MOVB    #2,DRVTYP(R1)   ;SET RP05 INDICATOR
3289  012752  022705  020021                  CMP     #20021,R5       ;SINGLE PORT RP05 ?
3290  012756  001420                          BEQ     2$              ;BR IF YES
3291  012760  022705  024021                  CMP     #24021,R5       ;DUAL PORT RP05 ?
3292  012764  001415                          BEQ     2$              ;BR IF YES
3293  012766  112761  000004  012260          MOVB    #4,DRVTYP(R1)   ;SET RP06 INDICATOR
3294  012774  022705  020022                  CMP     #20022,R5       ;SINGLE PORT RP06 ?
3295  013000  001407                          BEQ     2$              ;BR IF YES
3296  013002  022705  024022                  CMP     #24022,R5       ;DUAL PORT RP06 ?
3297  013006  001404                          BEQ     2$              ;BR IF YES
3298  013010  112761  177777  012260          MOVB    #-1,DRVTYP(R1)  ;SET INDICATOR TO 'OTHER'
3299  013016  000446                          BR      6$              ;EXIT
3300  013020  012746  000121          2$:     MOV     #121,-(SP)      ;DO A "READ-IN PRESET"
3301  013024  004037  017524                  JSR     R0,WRT.RP
3302  013030  000000                          RPCS1
3303  013032  013160                          8$
3304  013034  012746  010000                  MOV     #BIT12,-(SP)    ;SET FMT22=1
3305  013040  004037  017524                  JSR     R0,WRT.RP
3306  013044  000032                          RPOF
```

```
 3307  013046  013160                          8$
 3308  013050  004037  017344                  JSR     R0,RD.RP        ;READ RPDS1
 3309  013054  000012                          RPDS1
 3310  013056  013160                          8$
 3311  013060  012605                          MOV     (SP)+,R5        ;AND SAVE IT IN R5
 3312  013062  100015                          BPL     4$              ;BRANCH IF ATA=0
 3313  013064  116164  012364  000016          MOVB    ATABIT(R1),RPAS(R4) ;CLEAR ATTENTION BIT
 3314  013072  004037  017344                  JSR     R0,RD.RP        ;FIND OUT WHY ATA=1
 3315  013076  000014                          RPER1
 3316  013100  013160                          8$
 3317  013102  006126                          ROL     (SP)+           ;IS IT UNSAFE?
 3318  013104  100004                          BPL     4$              ;BR IF NOT
 3319  013106  112761  177777  012250          MOVB    #-1,DRVSTA(R1)  ;SET UNSAFE INDICATOR
 3320  013114  000407                          BR      6$              ;EXIT
 3321  013116  005105                  4$:     COM     R5              ;CHECK MOL, DPR, DRY, AND VV
 3322  013120  042705  167077                  BIC     #^C<BIT12!BIT08!BIT07!BIT06>,R5
 3323  013124  001003                          BNE     6$              ;BRANCH IF MOL, DPR, DRY, OR VV IS CLEAR
 3324  013126  112761  000001  012250          MOVB    #1,DRVSTA(R1)   ;SET DRIVE STATUS TO ONLINE
 3325  013134  005720                  6$:     TST     (R0)+           ;STEP OVER THE ERROR RETURN
 3326  013136  000410                          BR      8$              ;EXIT
 3327  013140  006301                  7$:     ASL     R1              ;CHANGE INDEX TO ADDRESS WORDS
 3328  013142  012761  003720  012342          MOV     #2000.,TIMER(R1)  ;START 2 SEC TIMER
 3329  013150  006201                          ASR     R1              ;RESTORE R1
 3330  013152  105161  012270                  COMB    DPINT(R1)       ;SET PORT INITIALIZE INIDICATOR
 3331  013156  005720                          TST     (R0)+
 3332  013160  012605                  8$:     MOV     (SP)+,R5        ;RESTORE R5
 3333  013162  000200                          RTS     R0              ;EXIT
 3334
 3335                                  ;REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST
 3336                                  ;
 3337                                  ;CALL
 3338                                  ;
 3339                                  ;       JSR     R0,@#RP04       ;CALL THE RP04/5/6 DRIVER
 3340                                  ;       PNTADR                  ;ADDRESS OF POINTER OF DRIVES PARAMETER BLOCK
 3341                                  ;       RETURN1                 ;RETURN HERE IF QUEUE IS FULL
 3342                                  ;       RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE OR THERE
 3343                                                                  ;IS AN ERROR CONDITION
 3344
 3345  013164  013746  177776          RP04:   MOV     @#PS,-(SP)      ;SAVE THE CALLING STATUS
 3346  013170  013737  012402  177776          MOV     RPVEC+2,@#PS    ;DON'T ALLOW ANY RP04/5/6 INTERRUPTS
 3347  013176  112737  000001  012314          MOVB    #1,ACTDRV       ;SET "ACTIVE DRIVER" FLAG
 3348  013204  104412                          SAVREG                  ;SAVE R0 - R5
 3349  013206  011002                          MOV     (R0),R2         ;PICKUP THE DRIVE PARAMETER BLOCK POINTER
 3350  013210  005062  000016                  CLR     16(R2)          ;CLEAR THE STATUS/ERROR INDICATOR
 3351  013214  111201                          MOVB    (R2),R1         ;PICKUP THE DRIVE NUMBER
 3352  013216  013704  012376                  MOV     RPADR,R4        ;UNIBUS ADDRESS OF RPCS1
 3353  013222  105761  012250                  TSTB    DRVSTA(R1)      ;CHECK DRIVES STATUS
 3354  013226  003014                          BGT     1$              ;BRANCH IF ONLINE
 3355  013230  105761  012316                  TSTB    ULDFLG(R1)      ;UNLOAD COMMAND IN QUEUE?
 3356  013234  001036                          BNE     3$              ;BRANCH IF YES
 3357  013236  105761  012270                  TSTB    DPINT(R1)       ;TRYING TO INIT THE DRIVE
 3358  013242  001042                          BNE     5$              ;BR IF YES
 3359  013244  004037  012626                  JSR     R0,DRVINT       ;GO INIT. THE DRIVE
 3360  013250  000434                          BR      4$              ;ERROR RETURN
 3361  013252  105761  012250                  TSTB    DRVSTA(R1)      ;IS DRIVE STATUS ONLINE?
 3362  013256  003445                          BLE     6$              ;BR IF NOT
```

```
 3363  013260  105761  012300          1$:     TSTB    DPRQS(R1)         ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
 3364  013264  001031                          BNE     5$                ;BR IF YES
 3365  013266  010164  000010                  MOV     R1,RPCS2(R4)      ;SELECT THE DRIVE
 3366  013272  004037  020476                  JSR     R0,DRVQUE         ;PUT THIS REQUEST IN QUEUE
 3367  013276  000460                          BR      9$                ;QUEUE IS FULL
 3368  013300  122762  000103  000002          CMPB    #103,2(R2)        ;IS THIS REQ. FOR AN UNLOAD?
 3369  013306  001003                          BNE     2$                ;BR IF NO
 3370  013310  112761  177777  012316          MOVB    #-1,ULDFLG(R1)    ;SET THE 'UNLOAD IN QUEUE'' FLAG
 3371  013316  105761  012240          2$:     TSTB    DRVACT(R1)        ;IS THIS DRIVE ACTIVE?
 3372  013322  001043                          BNE     8$                ;BR IF YES
 3373  013324  004737  013456                  JSR     PC,OPT            ;CALL THE OPTIMIZER
 3374  013330  000440                          BR      8$
 3375  013332  012762  120000  000016  3$:     MOV     #BIT15!BIT13,16(R2) ;SET THE 'UNLOAD IN QUEUE'' ERROR FLAG
 3376  013340  000434                          BR      8$                ;EXIT
 3377  013342  004737  014566          4$:     JSR     PC,CI7            ;GO HANDLE THE PARITY ERROR
 3378  013346  000431                          BR      8$
 3379  013350  004037  020476          5$:     JSR     R0,DRVQUE         ;PUT REQUEST IN QUEUE
 3380  013354  000431                          BR      9$                ;QUEUE IS FULL
 3381  013356  032714  000100                  BIT     #BIT06,(R4)       ;IS 'IE' SET ALREADY ?
 3382  013362  001023                          BNE     8$                ;BR IF IT IS
 3383  013364  004737  020034                  JSR     PC,SET.IE         ;SET INTERRUPT
 3384  013370  000420                          BR      8$                ;RETURN, REQUEST IN QUEUE
 3385  013372  105761  012250          6$:     TSTB    DRVSTA(R1)        ;SEE IF DRIVE OFFLINE OR UNSAFE
 3386  013376  002412                          BLT     7$                ;BR IF UNSAFE
 3387  013400  012762  140000  000016          MOV     #BIT15!BIT14,16(R2)  ;SET OFFLINE ERROR INDICATOR
 3388  013406  105761  012260                  TSTB    DRVTYP(R1)        ;SEE IF OFFLINE OR NONEXISTENT
 3389  013412  001007                          BNE     8$                ;BR IF OFFLINE
 3390  013414  012762  100002  000016          MOV     #BIT15!BIT01,16(R2)  ;REPORT DRIVE NONEXISTENT
 3391  013422  000403                          BR      8$                ;GO TO EXIT
 3392  013424  012762  110000  000016  7$:     MOV     #BIT15!BIT12,16(R2)  ;DRIVE IS UNSAFE
 3393  013432  104413          8$:     RESREG                            ;RESTORE R0 - R5
 3394  013434  005720                  TST     (R0)+                     ;SETUP FOR NORMAL RETURN
 3395  013436  000401                  BR      10$                       ;FINISH UP, THEN EXIT
 3396  013440  104413          9$:     RESREG                            ;RESTORE R0 - R5
 3397  013442  005720          10$:    TST     (R0)+                     ;CORRECT THE RETURN ADDRESS
 3398  013444  105037  012314                  CLRB    ACTDRV            ;CLEAR ''ACTIVE DRIVER'' FLAG
 3399  013450  012637  177776                  MOV     (SP)+,@#PS        ;RETURN 'PS'' TO USER LEVEL
 3400  013454  000200                          RTS     R0                ;RETURN TO CALLER
 3401
 3402                                  ;OPTIMIZER-CALLED FOR A PARTICULAR DRIVE
 3403                                  ;
 3404                                  ;CALL
 3405                                  ;       MOV     #DRVNUM,R1        ;DRIVE NUMBER TO R1
 3406                                  ;       JSR     PC,OPT            ;SETUP A COMMAND
 3407                                  ;
 3408  013456  104412          OPT:    SAVREG                            ;SAVE R0 - R5
 3409  013460  013746  177776                  MOV     @#PS,-(SP)        ;SAVE PROC. STATUS
 3410  013464  146137  012364  012312          BICB    ATABIT(R1),SRCHWT         ;CLEAR 'SEARCH WAIT'' KEY
 3411  013472  004737  020552                  JSR     PC,GETREQ         ;GET ''DPB'' POINTER OF REQUEST
 3412  013476  005702                          TST     R2                ;IS THERE A REQUEST IN QUEUE?
 3413  013500  001505                          BEQ     7$                ;NO--BRANCH TO EXIT
 3414  013502  032764  004000  000000          BIT     #BIT11,RPCS1(R4)          ;IS DVA SET?
 3415  013510  001407                          BEQ     10$               ;BRANCH IF NOT
 3416  013512  032764  000100  000012          BIT     #BIT6,RPDS1(R4)   ;IS VV SET ?
 3417  013520  001003                          BNE     10$               ;BR IF IT IS
 3418  013522  004037  012626          9$:     JSR     R0,DRVINT         ;SEE IF DRIVE STILL ONLINE ?
```

```
3419  013526  000470                          BR      6$                ;PARITY OR 'DVA' NOT SET
3420  013530  105761  012250          10$:    TSTB    DRVSTA(R1)        ;IS DRIVE ONLINE?
3421  013534  003014                          BGT     1$                ;YES--BRANCH
3422  013536  004737  020574                  JSR     PC,POPQUE         ;NO--REMOVE REQUEST FROM QUEUE
3423  013542  012762  140000  000016          MOV     #BIT15!BIT14,16(R2)  ;SET OFFLINE STATUS/ERROR INDICATOR
3424  013550  105761  012250                  TSTB    DRVSTA(R1)        ;IS DRIVE UNSAFE ?
3425  013554  100064                          BPL     8$                ;BR TO EXIT IF NOT
3426  013556  012762  110000  000016          MOV     #BIT15!BIT12,16(R2)  ;SET UNSAFE STATUS/ERROR INDICATOR
3427  013564  000460                          BR      8$                ;BRANCH TO EXIT
3428  013566  012746  000111          1$:     MOV     #111,-(SP)        ;LOAD COMMAND ONTO THE STACK
3429  013572  004037  017524                  JSR     R0,WRT.RP         ;LOAD THE REGISTER
3430  013576  000000                          RPCS1                     ;REGISTER INCREMENT
3431  013600  013710                          6$                        ;ERROR RETURN ADDRESS
3432  013602  032714  004000                  BIT     #BIT11,(R4)       ;DRIVE AVAILABLE ?
3433  013606  001427                          BEQ     5$                ;BR IF NOT
3434  013610  122762  000150  000002          CMPB    #150,2(R2)        ;IS THE REQUEST FOR I/O?
3435  013616  002403                          BLT     2$                ;YES--BRANCH
3436  013620  004737  014152                  JSR     PC,CI4            ;CALL THE COMMAND INITIATOR
3437  013624  000440                          BR      8$                ;BRANCH TO EXIT
3438  013626  005737  012362          2$:     TST     DTUW              ;DATA TRANSFER UNDERWAY?
3439  013632  002012                          BGE     4$                ;YES--GO START A SEARCH
3440  013634  005737  012340                  TST     SEEKFG            ;DO IMPLIED SEEKS?
3441  013640  100404                          BMI     3$                ;YES---BRANCH
3442  013642  004037  015122                  JSR     R0,LA             ;NO--DO LOOK AHEAD
3443  013646  000427                          BR      8$                ;RETURN HERE ON A PARITY ERROR
3444  013650  000403                          BR      4$                ;GO START A SEARCH
3445  013652  004737  013736          3$:     JSR     PC,CI1            ;START A DATA TRANSFER
3446  013656  000423                          BR      8$
3447  013660  004737  014044          4$:     JSR     PC,CI3            ;START A SEARCH
3448  013664  000420                          BR      8$                ;GO TO THE EXIT
3449  013666  112761  177777  012300  5$:     MOVB    #-1,DPRQS(R1)     ;SET PORT REQUEST INDICATOR
3450  013674  010103                          MOV     R1,R3             ;SET UP TO ADDRESS WORDS
3451  013676  006303                          ASL     R3                ;CONVERT TO WORD INDEX
3452  013700  012763  023420  012342          MOV     #10000.,TIMER(R3)  ;START 10 SEC TIMER
3453  013706  000402                          BR      7$                ;EXIT
3454  013710  004737  014566          6$:     JSR     PC,CI7            ;PROCESS THE PARITY ERROR
3455  013714  032714  000100          7$:     BIT     #BIT06,(R4)       ;SEE IF 'IE' ALREADY SET
3456  013720  001002                          BNE     8$                ;BR IF SET
3457  013722  004737  020034                  JSR     PC,SET.IE         ;SET ''IE'' WITHOUT A 'TRE''
3458  013726  012637  177776          8$:     MOV     (SP)+,@#PS        ;RESTORE PROC. STATUS
3459  013732  104413                          RESREG                    ;RESTORE R0 - R5
3460  013734  000207                          RTS     PC
3461
3462                                  ;COMMAND INITIATOR
3463                                  ;
3464                                  ;CALL
3465                                  ;       MOV     #DRVNUM,R1        ;DRIVE NUMBER
3466                                  ;       MOV     #DPB,R2           ;ADDRESS OF DPB
3467                                  ;       JSR     PC,CI?            ;CI?= CI1,CI3, OR CI4
3468                                  ;                                 ;WHERE:
3469                                  ;                                 ;CI1=DATA TRANSFER
3470                                  ;                                 ;CI2=SEARCH REQUESTED BY DATA XFER
3471                                  ;                                 ;CI4=NOT DATA TRANSFER
3472
3473  013736  004737  020574          CI1:    JSR     PC,POPQUE         ;REMOVE REQUEST FROM ''DRIVES WAIT'' QUEUE
3474  013742  010237  012310                  MOV     R2,TRNSWT         ;PUT REQ. IN TRANSFER WAIT QUEUE
```

```
3475  013746  010203                          MOV     R2,R3           ;DPB ADDRESS TO R3
3476  013750  013704  012376                  MOV     RPADR,R4        ;RPCS1 ADDRESS
3477  013754  010164  000010                  MOV     R1,RPCS2(R4)    ;SELECT DRIVE
3478  013760  062703  000004                  ADD     #4,R3           ;DESIRED WORD COUNT
3479  013764  062704  000002                  ADD     #2,R4           ;RPWC ADDRESS
3480  013770  012324                          MOV     (R3)+,(R4)+     ;LOAD WORD COUNT
3481  013772  012324                          MOV     (R3)+,(R4)+     ;LOAD BUFFER ADDRESS
3482  013774  012346                          MOV     (R3)+,-(SP)     ;LOAD SECTOR AND TRACK
3483  013776  004037  017524                  JSR     R0,WRT.RP       ;CALL THE LOAD(WRITE) ROUTINE
3484  014002  000006                          RPDA                    ;INDEX OF REGISTER TO LOAD
3485  014004  014566                          CI7                     ;ERROR RETURN ADDRESS
3486  014006  012346                          MOV     (R3)+,-(SP)     ;LOAD CYLINDER ADDRESS
3487  014010  004037  017524                  JSR     R0,WRT.RP
3488  014014  000034                          RPCA
3489  014016  014566                          CI7
3490  014020  016246  000002                  MOV     2(R2),-(SP)     ;LOAD ''COMMAND+GO'', ''A17&A16'', AND 'PSEL''
3491  014024  004037  017524                  JSR     R0,WRT.RP
3492  014030  000000                          RPCS1
3493  014032  014566                          CI7
3494  014034  010137  012362                  MOV     R1,DTUW         ;SET 'DATA TRANSFER UNDERWAY''
3495  014040  000137  014530                  JMP     CI5
3496  014044  013704  012376          CI3:    MOV     RPADR,R4        ;RPCS1 ADDRESS
3497  014050  010164  000010                  MOV     R1,RPCS2(R4)    ;SELECT DRIVE
3498  014054  016246  000012                  MOV     12(R2),-(SP)    ;DESIRED CYLINDFR ADDRESS
3499  014060  004037  017524                  JSR     R0,WRT.RP
3500  014064  000034                          RPCA
3501  014066  014566                          CI7
3502  014070  116203  000010                  MOVB    10(R2),R3       ;PICKUP SECTOR ADDRESS
3503  014074  163703  012412                  SUB     MXWNDW,R3       ;BACKUP BY MAX. SEARCH FOR I/O WINDOW
3504  014100  002002                          BGE     1$
3505  014102  062703  000026                  ADD     #22.,R3
3506  014106  010346                  1$:     MOV     R3,-(SP)        ;COMBINE THE ADJUSTED SECTOR WITH
3507  014110  116266  000011  000001          MOVB    11(R2),1(SP)    ;THE DESIRED TRACK
3508  014116  004037  017524                  JSR     R0,WRT.RP       ;LOAD DESIRED TRACK & SECTOR
3509  014122  000006                          RPDA
3510  014124  014566                          CI7
3511  014126  012746  000131                  MOV     #131,-(SP)      ;START A SEARCH
3512  014132  004037  017524                  JSR     R0,WRT.RP
3513  014136  000000                          RPCS1
3514  014140  014566                          CI7
3515  014142  156137  012364  012312          BISB    ATABIT(R1),SRCHWT       ;SET ''SEARCH WAIT'' KEY
3516  014150  000567                          BR      CI5
3517  014152  013704  012376          CI4:    MOV     RPADR,R4        ;RPCS1 ADDRESS
3518  014156  010164  000010                  MOV     R1,RPCS2(R4)    ;SELECT DRIVE
3519  014162  116203  000002                  MOVB    2(R2),R3        ;PICKUP THE REQUESTED COMMAND
3520  014166  122703  000131                  CMPB    #131,R3         ;IS IT A SEARCH COMMAND?
3521  014172  001007                          BNE     1$              ;BRANCH IF NO
3522  014174  016246  000010                  MOV     10(R2),-(SP)    ;LOAD DESIRED TRACK & SECTOR
3523  014200  004037  017524                  JSR     R0,WRT.RP
3524  014204  000006                          RPDA
3525  014206  014566                          CI7
3526  014210  000403                          BR      2$              ;GO LOAD CYLINDER
3527  014212  122703  000105          1$:     CMPB    #105,R3         ;IS IT A SEEK COMMAND
3528  014216  001007                          BNE     3$              ;BRANCH IF NO
3529  014220  016246  000012          2$:     MOV     12(R2),-(SP)    ;LOAD DESIRED CYLINDER
3530  014224  004037  017524                  JSR     R0,WRT.RP
```

```
                                        RPCA
                                        CI7
                                        BR      CI6
                        000115  3$:     CMPB    #115,R3         ;IS IT AN ''OFFSET'' COMMAND?
                                        BNE     4$              ;BR IF NO
                        017344          JSR     R0,RD.RP        ;MERGE THE OFFSET VALUE INTO RPOF
                                        RPOF                    ;BUT DON'T CHANGE THE UPPER
                                        CI7
                        000001          MOVB    1(R2),(SP)      ;BYTE WHEN LOADING THE
                        017524          JSR     R0,WRT.RP       ;REGISTER (RPOF)
                                        RPOF
                                        CI7
                                        BR      CI6             ;GO START THE COMMAND
                        000107  4$:     CMPB    #107,R3         ;IS IT A ''RECALIBRATE'' COMMAND?
                                        BEQ     CI6             ;BRANCH IF YES
                        000117          CMPB    #117,R3         ;IS IT A RETURN TO CENTER?
                                        BEQ     CI6             ;BRANCH IF YES
                        000103          CMPB    #103,R3         ;IS IT AN ''UNLOAD'' COMMAND?
                                        BNE     5$              ;BRANCH IF NO
                        012240          MOVB    #1,DRVACT(R1)   ;SET THE DRIVE ACTIVE INDICATOR
                        012250          CLRB    DRVSTA(R1)      ;PUT DRIVE STATUS TO OFFLINE
                        012316          MOVB    #1,ULDFLG(R1)   ;SET ''UNLOAD IN PROGRESS'' FLAG
                                        MOV     R3,-(SP)        ;START THE ''UNLOAD'' COMMAND
                        017524          JSR     R0,WRT.RP
                                        RPCS1
                                        CI7
                                        RTS     PC              ;RETURN TO USER
                        000143  5$:     CMPB    #143,R3         ;IS IT A ''SET FORMAT'' COMMAND?
                                        BNE     6$              ;BRANCH IF NO
                        017344          JSR     R0,RD.RP        ;READ THE OFFSET REGISTER
                                        RPOF
                                        CI7
                        000001  000001  MOVB    1(R2),1(SP)     ;COMBINE ''FMT22'',''ECI'', AND ''HCI''
                        017524          JSR     R0,WRT.RP       ;LOAD ''FMT22'', ''ECI'', AND/OR ''HCI''.
                                        RPOF
                                        CI7
                                        BR      12$
                        000141  6$:     CMPB    #141,R3         ;IS IT A ''GET REGISTER'' COMMAND?
                                        BNE     10$             ;BRANCH IF NO
                        000006  7$:     MOV     6(R2),R3        ;POINTS TO 1ST ADDRESS OF WHERE
                                                                ;TO PUT THE REGISTER(S)
                        000010  014436  MOVB    10(R2),9$       ;INIT. THE INDEX FOR THE FIRST REG.
                        000011          MOVB    11(R2),R5       ;INDEX OF LAST REG. TO MOVE
                        017344  8$:     JSR     R0,RD.RP        ;READ RP04/5/6 REGISTER
                                9$:     RPCS1                   ;INDEX OF REG. TO READ
                                        CI7
                                        MOV     (SP)+,(R3)+     ;GET THE CONTENTS OF RH11/RP04/5/6 REG.
                        014436          CMP     9$,R5           ;LAST REG. BEEN READ?
                                        BEQ     12$             ;GET OUT IF YES
                        000002  014436  ADD     #2,9$           ;INCREASE THE INDEX BY 2
                                        BR      8$              ;LOOP--MORE TO READ
                        000145  10$:    CMPB    #145,R3         ;IS IT A ''SELECT DRIVE'' COMMAND?
                                        BEQ     12$             ;BRANCH IF YES
                                11$:    MOV     R3,-(SP)        ;LOAD THE COMMAND
                        017524          JSR     R0,WRT.RP
                                        RPCS1
```

```
                                                CI7
                        020574          12$:    JSR     PC,POPQUE       ;REMOVE REQ. FROM QUEUE
                        000200  000016          BIS     #BIT07,16(R2)   ;SET THE 'DONE'' BIT
                        012336                  TST     SAVEFG          ;SAVE THE RH11/RP04/5/6 REGISTERS?
                                                BPL     13$             ;BRANCH IF NO
                        017716                  JSR     PC,SVRH11       ;YES--GO SAVE THE REGISTERS
                                        13$:    RTS     PC              ;RETURN TO USER
                                        CI5:    ASL     R1
                        001750  012342          MOV     #1000.,TIMER(R1)        ;SET A ONE SECOND TIMER
                                                ASR     R1
                        000001  012240          MOVB    #1,DRVACT(R1)   ;SET THE DRIVE ACTIVE
                                                RTS     PC              ;RETURN TO THE USER
                                        CI6:    MOV     R3,-(SP)        ;LOAD THE COMMAND
                        017524                  JSR     R0,WRT.RP
                                                RPCS1
                                                CI7
                                                BR      CI5
                        010000  000010  CI7:    BIT     #BIT12,RPCS2(R4)  ;DRIVE NON-EXISTENT ?
                                                BNE     CI8             ;BR IF YES
                                        1$:     TST     R2              ;ANYTHING IN QUEUE ?
                                                BEQ     CI7B            ;BR IF NOT
                        104000  000016          MOV     #BIT15!BIT11,16(R2)     ;SET 'PARITY'' ERROR INDICATOR
                        017716                  JSR     PC,SVRH11       ;GO SAVE THE RH11/RP04/5/6 REGISTERS
                        000111          CI7B:   MOV     #111,-(SP)      ;DO A 'DRIVE CLEAR''
                        017524                  JSR     R0,WRT.RP
                                                RPCS1
                                                CI8
                        020456                  JSR     PC,EMPTYQ       ;EMPTY THE QUEUE
                        012316                  CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD IN QUEUE FLAG
                        012240                  CLRB    DRVACT(R1)      ;DRIVE IS IDLE
                        012362                  CMP     R1,DTUW         ;IF THIS DRIVE HAD AN I/O REQUEST
                                                BNE     1$              ;IN PROGRESS CLEAR ALL OF THE FLAGS
                        012310                  CLR     TRNSWT
                        177777  012362          MOV     #-1,DTUW
                                        1$:     RTS     PC
                                        CI8:    SAVREG                  ;SAVE R0 - R5
                        010000  000010          BIT     #BIT12,RPCS2(R4)  ;IS 'NED' SET ?
                                                BNE     1$              ;BR IF YES
                                                CLR     R1
                                                CLR     R3
                        012240          1$:     TSTB    DRVACT(R1)      ;DRIVE ACTIVE?
                                                BEQ     5$              ;BRANCH IF NO
                        012310                  MOV     TRNSWT,R2       ;GET THE 'TRANSFER WAIT'' QUEUE
                        012362                  CMP     R1,DTUW         ;DID THIS DRIVE HAVE AN I/O IN PROGRESS?
                                                BEQ     2$              ;BRANCH IF YES
                        020552                  JSR     PC,GETREQ       ;GET THE DPB POINTER
                                        2$:     TST     R2              ;QUEUE ENTRY FOR DRIVE ?
                                                BEQ     4$              ;BR IF NOT
                        010000  000010          BIT     #BIT12,RPCS2(R4)  ;'NED' SET ?
                                                BEQ     3$              ;BR IF NOT
                        100002  000016          MOV     #BIT15!BIT01,16(R2)  ;SET 'DRIVE NON-EXISTENT' INDICATOR
                                                BR      4$              ;CONTINUE
                        102000  000016  3$:     MOV     #BIT15!BIT10,16(R2) ;SET 'NON-CLEARABLE PARITY'' ERROR INDICATOR
                        017716                  JSR     PC,SVRH11       ;SAVE RH11/RP04/5/6 REGISTERS
                        177777  012342  4$:     MOV     #-1,TIMER(R3)   ;STOP THE TIMER
                        012240                  CLRB    DRVACT(R1)      ;SET 'DRIVE ACTIVE'' TO IDLE
```

```
3643  015000  020137  012362                  CMP     R1,DTUW         ;IS THIS DRIVE SETUP FOR A TRANSFER
3644  015004  001005                          BNE     5$              ;BR IF NOT
3645  015006  012737  177777  012362          MOV     #-1,DTUW        ;RESET THE INDICATOR
3646  015014  005037  012310                  CLR     TRNSWT          ;CLEAR THE TRANSFER QUEUE
3647  015020  105061  012316          5$:     CLRB    ULDFLG(R1)      ;CLEAR UNLOAD FLAG
3648  015024  032764  010000  000010          BIT     #BIT12,RPCS2(R4)  ;'NED' SET ?
3649  015032  001021                          BNE     6$              ;BR IF YES
3650  015034  005201                          INC     R1              ;MOVE TO THE NEXT DRIVE
3651  015036  062703  000002                  ADD     #2,R3
3652  015042  042701  177770                  BIC     #^C7,R1
3653  015046  001316                          BNE     1$              ;BRANCH IF MORE DRIVES
3654  015050  012737  177777  012362          MOV     #-1,DTUW        ;NO DATA TRANSFERS UNDERWAY
3655  015056  005037  012310                  CLR     TRNSWT          ;CLEAR THE 'TRANSFER WAIT' QUEUE
3656  015062  004737  020400                  JSR     PC,CLRQUE       ;CLEAR ALL OF THE REQUEST QUEUES
3657  015066  012764  000040  000010          MOV     #BIT05,RPCS2(R4) ;DO A MASSBUS INIT.
3658  015074  000406                          BR      7$              ;CONTINUE
3659  015076  004737  020456          6$:     JSR     PC,EMPTYQ       ;CLEAR THE DRIVE'S QUEUE
3660  015102  105061  012250                  CLRB    DRVSTA(R1)      ;SET DRIVE TO OFFLINE
3661  015106  105061  012260                  CLRB    DRVTYP(R1)      ;CLEAR THE DRIVE TYPE INDICATOR
3662  015112  004737  020034          7$:     JSR     PC,SET.IE       ;SET ''IE'' WITHOUT ''TRE''
3663  015116  104413                          RESREG                  ;RESTORE R0 - R5
3664  015120  000207                          RTS     PC              ;RETURN
3665
3666                                  ;LOOK AHEAD ROUTINE
3667                                  ;
3668                                  ;CALL
3669                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
3670                                  ;       MOV     #DPB,R2         ;POINT TO DPB
3671                                  ;       JSR     R0,LA           ;GO CHECK THE WINDOW
3672                                  ;       RETURN1                 ;ERROR RETURN
3673                                  ;       RETURN2                 ;START A SEARCH
3674                                  ;       RETURN3                 ;START A DATA TRANSFER
3675
3676  015122  013704  012376          LA:     MOV     RPADR,R4        ;GET RPCS1'S ADDRESS
3677  015126  010164  000010                  MOV     R1,RPCS2(R4)    ;SELECT DRIVE
3678  015132  004037  017344                  JSR     R0,RD.RP        ;READ CURRENT CYLINDER
3679  015136  000036                          RPCC
3680  015140  015252                          4$                      ;ERROR RETURN ADDRESS
3681  015142  022662  000012                  CMP     (SP)+,12(R2)    ;IS CURRENT CYLINDER=DESIRED
3682                                                                  ;CYLINDER?
3683  015146  001037                          BNE     3$              ;EXIT IF NO
3684  015150  105261  012326                  INCB    LACNT(R1)       ;INCREMENT THE LOOK AHEAD COUNT
3685  015154  126137  012326  012404          CMPB    LACNT(R1),MXLACT ;EXCEED MAX?
3686  015162  003026                          BGT     2$              ;BRANCH IF YES
3687  015164  116203  000010                  MOVB    10(R2),R3       ;GET DESIRED SECTOR ADDRESS AND
3688  015170  000303                          SWAB    R3              ;MULT. BY 64--ALIGN WITH
3689  015172  006203                          ASR     R3              ;LOOK AHEAD REGISTER
3690  015174  006203                          ASR     R3
3691  015176  012737  000340  177776          MOV     #340,@#PS       ;PRIORITY LEVEL '7''
3692  015204  004037  017344                  JSR     R0,RD.RP        ;READ LOOK AHEAD REGISTER
3693  015210  000020                          RPLA
3694  015212  015252                          4$
3695  015214  162603                          SUB     (SP)+,R3        ;CALCULATE THE DEI TA
3696  015216  002002                          BGE     1$
3697  015220  062703  002600                  ADD     #<22.*64.>,R3   ;MAKE THE DELTA POSITIVE
3698  015224  023703  012406          1$.     CMP     MXDLTA,R3       ;CHECK THE DELTA TO SEE
```

```
 3699  015230  002406                          BLT     3$              ;IF IT IS WITHIN THE
 3700  015232  023703  012410                  CMP     MNDLTA,R3       ;WINDOW---IF YES, ZERO
 3701  015236  002003                          BGE     3$              ;THE LOOK AHEAD COUNT
 3702  015240  105061  012326          2$:     CLRB    LACNT(R1)       ;AND TAKE THE I/O EXIT
 3703  015244  005720                          TST     (R0)+
 3704  015246  005720                  3$:     TST     (R0)+           ;ADJUST THE RETURN ADDRESS
 3705  015250  000402                          BR      5$              ;EXIT
 3706  015252  004737  014566          4$:     JSR     PC,CI7          ;PROCESS THE ERROR
 3707  015256  000200                  5$:     RTS     R0              ;RETURN
 3708
 3709                                  ;INTERRUPT SERVICE ROUTINE
 3710
 3711  015260  112737  000001  012314  ISR:    MOVB    #1,ACTDRV       ;SET ''ACTIVE DRIVER'' FLAG
 3712  015266  104412                          SAVREG                  ;SAVE R0 - R5
 3713  015270  013704  012376                  MOV     RPADR,R4        ;ADDRESS OF RHSCS1
 3714  015274  013701  012362                  MOV     DTUW,R1         ;GET ''DATA TRANSFER UNDERWAY'' INDICATOR
 3715  015300  002403                          BLT     1$              ;BRANCH IF NO DATA TRANSFER UNDERWAY
 3716  015302  004737  015324                  JSR     PC,TD           ;CALL TRANSFER DONE
 3717  015306  000402                          BR      2$              ;EXIT
 3718  015310  004737  015464          1$:     JSR     PC,SC           ;CALL SPECIAL CONDITIONS
 3719  015314  104413                  2$:     RESREG                  ;RESTORE R0 - R5
 3720  015316  105037  012314                  CLRB    ACTDRV          ;CLEAR ''ACTIVE DRIVER'' FLAG
 3721  015322  000002                          RTI                     ;RETURN
 3722
 3723                                  ;TRANSFER DONE ROUTINE
 3724
 3725  015324  105061  012240          TD:     CLRB    DRVACT(R1)      ;SET DRIVE ACTIVE INDICATOR TO IDLE
 3726  015330  012737  177777  012362          MOV     #-1,DTUW        ;NO DATA TRANSFERS UNDERWAY
 3727  015336  006301                          ASL     R1
 3728  015340  012761  177777  012342          MOV     #-1,TIMER(R1)   ;CANCEL TIMEOUT
 3729  015346  006201                          ASR     R1
 3730  015350  013702  012310                  MOV     TRNSWT,R2       ;GET ''DPB'' ADDRESS FROM THE
 3731  015354  005037  012310                  CLR     TRNSWT          ;TRANSFER WAIT QUEUE--CLEAR QUEUE
 3732  015360  052762  000200  000016          BIS     #BIT07,16(R2)   ;SET DONE
 3733  015366  010164  000010                  MOV     R1,RPCS2(R4)    ;SELECT THE DRIVE
 3734  015372  004037  017344                  JSR     R0,RD.RP        ;TRANSFER ERROR(TRE=1)?
 3735  015376  000000                          RPCS1
 3736  015400  014566                          CI7
 3737  015402  006126                          ROL     (SP)+
 3738  015404  100413                          BMI     3$              ;BR IF YES
 3739  015406  005737  012336                  TST     SAVEFG          ;SAVE THE RH11/RP04/5/6 REGISTERS?
 3740  015412  100002                          BPL     1$              ;BRANCH IF NO
 3741  015414  004737  017716                  JSR     PC,SVRH11       ;YES--SAVE THE REGISTERS
 3742  015420  004737  013456          1$:     JSR     PC,OPT          ;CALL OPTIMIZER
 3743  015424  000417                          BR      SC              ;CHECK OTHER DRIVES
 3744  015426  012714  000113          2$:     MOV     #113,(R4)       ;RELEASE THE DRIVE
 3745  015432  000414                          BR      SC              ;CHECK FOR OTHER DRIVES
 3746  015434  052762  100100  000016  3$:     BIS     #BIT15!BIT06,16(R2) ;SET DATA ERROR FLAG
 3747  015442  004737  020456                  JSR     PC,EMPTYQ       ;EMPTY THE ''DRIVE'S WAIT'' QUEUE
 3748  015446  004737  017716                  JSR     PC,SVRH11       ;SAVE THE RH11/RP04/5/6 REGISTERS
 3749  015452  012714  040111                  MOV     #40111,(R4)     ;ISSUE A ''DRIVE CLEAR''
 3750  015456  012714  000113                  MOV     #113,(R4)       ;ISSUE A RELEASE TO THE DRIVE
 3751  015462  000400                          BR      SC              ;CHECK FOR OTHER DRIVES
 3752
 3753                                  ;SPECIAL CONDITION ROUTINE
 3754
```

```
 3755  015464  116403  000016          SC:     MOVB    RPAS(R4),R3     ;READ 'RPAS'
 3756  015470  001014                          BNE     2$              ;BRANCH IF ANY 'ATA' BITS SET
 3757  015472  004037  017344                  JSR     R0,RD.RP        ;READ CONTROL AND STATUS REGISTER
 3758  015476  000000                          RPCS1
 3759  015500  014666                          CI8
 3760  015502  106126                          ROLB    (SP)+           ;IS ''IE''=1?
 3761  015504  100405                          BMI     1$              ;YES, NO DRIVES TO CHECK
 3762  015506  004037  020636                  JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
 3763  015512  104001                          ERROR   1               ;REPORT AN ILLEGAL INTERRUPT
 3764  015514  004737  020034                  JSR     PC,SET.IE       ;SET INTERRUPT ENABLE
 3765  015520  000207                  1$:     RTS     PC              ;RETURN
 3766  015522  005046                  2$:     CLR     -(SP)           ;PROCESS ALL DRIVES THAT HAVE
 3767  015524  110316                          MOVB    R3,(SP)         ;AN ''ATA''=1
 3768  015526  012703  000001                  MOV     #1,R3
 3769  015532  005001                          CLR     R1
 3770  015534  030316                  SC3:    BIT     R3,(SP)         ;ATA=1?
 3771  015536  001005                          BNE     SC5             ;YES--BRANCH
 3772  015540  005201                  SC4:    INC     R1              ;MOVE TO THE NEXT DRIVE
 3773  015542  106303                          ASLB    R3
 3774  015544  001373                          BNE     SC3             ;BRANCH IF MORE TO CHECK?
 3775  015546  005726                          TST     (SP)+           ;CLEAN OFF THE STACK
 3776  015550  000207                          RTS     PC              ;RETURN TO USER
 3777  015552  105761  012270          SC5:    TSTB    DPINT(R1)       ;INITIALIZING THE DRIVE ?
 3778  015556  001402                          BEQ     1$              ;BR IF NOT
 3779  015560  000137  016456                  JMP     SC13            ;PROCESS THE DRIVE
 3780  015564  105761  012300          1$:     TSTB    DPRQS(R1)       ;PORT REQUEST OUTSTANDING ?
 3781  015570  001402                          BEQ     2$              ;BR IF NOT
 3782  015572  000137  016456                  JMP     SC13            ;START THE OUTSTANDING COMMAND
 3783  015576  105761  012250          2$:     TSTB    DRVSTA(R1)      ;CHECK THE DRIVE STATUS
 3784  015602  003025                          BGT     5$              ;BRANCH IF ONLINE
 3785  015604  105761  012316                  TSTB    ULDFLG(R1)      ;UNLOAD IN PROGRESS?
 3786  015610  003422                          BLE     5$              ;BRANCH IF NOT
 3787  015612  004737  020552                  JSR     PC,GETREQ       ;GET DPB POINTER
 3788  015616  004737  017716                  JSR     PC,SVRH11       ;SAVE THE RH11/RP04/5/6 REGISTERS
 3789  015622  004737  016406                  JSR     PC,SC12         ;SAVE RPDS1, RPER1, RPER2, AND RPER3
 3790                                                                  ;ALSO DO A DRIVE INIT (DRVINT)
 3791  015626  105761  012250                  TSTB    DRVSTA(R1)      ;DID DRIVE COME ONLINE?
 3792  015632  003416                          BLE     6$              ;NO---BRANCH
 3793  015634  032737  040000  012230          BIT     #BIT14,RPERRS   ;WAS THERE AN ERROR?
 3794  015642  001002                          BNE     3$              ;BR IF ERROR
 3795  015644  000137  016316                  JMP     SC11            ;NO ERROR
 3796  015650  013705  012232          3$:     MOV     RPERRS+2,R5       'ES -- PICKUP RPER1 AND
 3797  015654  000502                          BR      SC6A            ;GO PROCESS THE ERROR
 3798  015656  105761  012240          5$:     TSTB    DRVACT(R1)      ;DRIVE ACTIVE WITH COMMAND OR ERROR RECOVERY ?
 3799  015662  001033                          BNE     SC6             ;BR IF EITHER
 3800  015664  004737  016406                  JSR     PC,SC12         ;SAVE RPDS1, RPER1, RPER2, AND RPER3
 3801                                                                  ;ALSO DO A DRVINT
 3802  015670  105761  012270          6$:     TSTB    DPINT(R1)       ;TRYING TO INIT THE DRIVE ?
 3803  015674  001321                          BNE     SC4             ;BR IF YES, CHECK ON MORE DRIVES
 3804  015676  105761  012250                  TSTB    DRVSTA(R1)      ;CHECK ON DRIVE'S STATUS
 3805  015702  100412                          BMI     7$              ;BR IF UNSAFE
 3806  015704  032737  020000  012236          BIT     #BIT13,RPERRS+6 ;ADDRESS PLUG CHANGED ?
 3807  015712  001013                          BNE     8$              ;BR IF YES
 3808  015714  012746  000113                  MOV     #113,-(SP)      ;RELEASE COMMAND
 3809  015720  004037  017524                  JSR     R0,WRT.RP       ;WRITE THE COMMAND INTO RPCS1
 3810  015724  000000                          RPCS1                   ;REGISTER INDEX
```

```
3811  015726  016266                        SC8                     ;PARITY EXIT ADDRESS
3812  015730  011605                7$:     MOV     (SP),R5         ;PICKUP (RPAS) BEFORE THE ERROR CALL
3813  015732  004037  020636                JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
3814  015736  104002                        ERROR   2               ;REPORT THE UNEXPECTED ATTENTION
3815  015740  000677                        BR      SC4             ;GO CHECK FOR MORE ATA'S
3816  015742                        8$:
3817  015742  004037  020636                JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
3818  015746  104005                        ERROR   5               ;REPORT THE ADDRESS PLUG CHANGE
3819  015750  000673                        BR      SC4             ;CHECK FOR MORE DRIVES
3820  015752  006301                SC6:    ASL     R1              ;SETUP TO ADDRESS WORDS
3821  015754  012761  177777  012342        MOV     #-1,TIMER(R1)   ;STOP THE TIMER
3822  015762  006201                        ASR     R1              ;RESTORE THE DRIVE ADDRESS
3823  015764  004737  020552                JSR     PC,GETREQ       ;GET THE DPB POINTER FROM THE QUEUE
3824  015770  010164  000010                MOV     R1,RPCS2(R4)    ;SELECT DRIVE
3825  015774  004037  017344                JSR     R0,RD.RP        ;READ THE RP04'S STATUS REG.
3826  016000  000012                        RPDS1
3827  016002  016266                        SC8
3828  016004  011605                        MOV     (SP),R5         ;AND PUT IT IN R5
3829  016006  006126                        ROL     (SP)+           ;WAS THERE AN ERROR?
3830  016010  100407                        BMI     1$              ;BR IF ERROR
3831  016012  105761  012240                TSTB    DRVACT(R1)      ;CHECK DRIVE'S STATE
3832  016016  003137                        BGT     SC11            ;BR IF DRIVE ACTIVE WITH ORDER
3833  016020  052762  100210  000016        BIS     #BIT15!BIT07!BIT03,16(R2)  ;INFORM USER OF ERROR RECOVER COMPLETION
3834  016026  000470                        BR      SC7
3835  016030  004037  017344          1$:   JSR     R0,RD.RP        ;READ ERROR REGISTER #1
3836  016034  000014                        RPER1
3837  016036  016266                        SC8
3838  016040  012605                        MOV     (SP)+,R5        ;AND SAVE IT IN R5
3839  016042  004737  017716                JSR     PC,SVRH11       ;SAVE RH11/RP04/5/6 REGISTERS
3840  016046  012746  000111                MOV     #111,-(SP)      ;ISSUE A DRIVE CLEAR
3841  016052  004037  017524                JSR     R0,WRT.RP
3842  016056  000000                        RPCS1
3843  016060  016266                        SC8
3844  016062  006105                SC6A:   ROL     R5              ;WAS 'UNSAFE'' CONDITION =1?
3845  016064  100406                        BMI     1$              ;BRANCH IF YES
3846  016066  005702                        TST     R2              ;ANYTHING IN QUEUE ?
3847  016070  001447                        BEQ     SC7             ;BR IF NOT
3848  016072  052762  100240  000016        BIS     #BIT15!BIT07!BIT05,16(R2)       ;INFORM USER OF ERROR
3849  016100  000443                        BR      SC7
3850  016102  004037  017344          1$:   JSR     R0,RD.RP        ;READ DRIVE STATUS REG. #1
3851  016106  000012                        RPDS1
3852  016110  016266                        SC8
3853  016112  011605                        MOV     (SP),R5         ;SAVE RPDS1 IN R5
3854  016114  006126                        ROL     (SP)+           ;'ERR'=1?
3855  016116  100011                        BPL     2$              ;BR IF NO--UNSAFE CLEARED
3856  016120  112761  177777  012250        MOVB    #-1,DRVSTA(R1)  ;DRIVE IS UNSAFE
3857  016126  004737  017716                JSR     PC,SVRH11       ;SAVE RH11/RP04/5/6 REGISTERS
3858  016132  052762  110000  000016        BIS     #BIT15!BIT12,16(R2)     ;INFORM USER OF UNSAFE ERROR
3859  016140  000423                        BR      SC7
3860  016142  032705  010000          2$:   BIT     #BIT12,R5       ;'MOL'' = 1 ?
3861  016146  001015                        BNE     3$              ;BR IF YES
3862  016150  112761  177777  012240        MOVB    #-1,DRVACT(R1)  ;ACTIVE ERROR RECOVER
3863  016156  112761  000001  012250        MOVB    #1,DRVSTA(R1)   ;ONLINE
3864  016164  006301                        ASL     R1
3865  016166  012761  072460  012342        MOV     #30000.,TIMER(R1) ;START 30 SECOND TIMER
3866  016174  006201                        ASR     R1
```

```
3867  016176  000137  015540                  JMP     SC4
3868  016202  052762  100220  000016  3$:     BIS     #BIT15!BIT07!BIT04,16(R2)       ;INFORM USER OF ERROR
3869  016210  105061  012240          SC7:    CLRB    DRVACT(R1)              ;DRIVE IS IDLE
3870  016214  004737  020456                  JSR     PC,EMPTYQ               ;DUMP THE QUEUE
3871  016220  105761  012316                  TSTB    ULDFLG(R1)              ;UNLOAD IN PROGRESS OR QUEUE?
3872  016224  003002                          BGT     1$                      ;BR IF NOT
3873  016226  105061  012316                  CLRB    ULDFLG(R1)              ;CLEAR UNLOAD FLAG
3874  016232  116164  012364  000016  1$:     MOVB    ATABIT(R1),RPAS(R4)  ;CLEAR ATTENTION BIT
3875  016240  105761  012250                  TSTB    DRVSTA(R1)              ;IS THE DRIVE UNSAFE ?
3876  016244  100406                          BMI     2$                      ;BR IF IT IS
3877  016246  012746  000113                  MOV     #113,-(SP)              ;RELEASE COMMAND
3878  016252  004037  017524                  JSR     R0,WRT.RP               ;WRITE THE COMMAND INTO RPCS1
3879  016256  000000                          RPCS1                           ;REGISTER INDEX
3880  016260  016266                          SC8                             ;PARITY EXIT ADDRESS
3881  016262  000137  015540          2$:     JMP     SC4                     ;CHECK FOR MORE DRIVES
3882  016266  105761  012240          SC8:    TSTB    DRVACT(R1)              ;IS DRIVE IDLE?
3883  016272  001405                          BEQ     1$                      ;YES--BRANCH
3884  016274  004737  020552                  JSR     PC,GETREQ               ;GET DPB POINTER
3885  016300  004737  014566                  JSR     PC,CI7                  ;PROCESS THE PARITY ERROR
3886  016304  000402                          BR      2$                      ;CONTINUE
3887  016306  004737  014614          1$:     JSR     PC,CI7B                 ;PROCESS THE UNCORRECTABLE PARITY ERROR
3888  016312  000137  015540          2$:     JMP     SC4                     ;CHECK MORE DRIVES
3889  016316  105761  012316          SC11:   TSTB    ULDFLG(R1)              ;''UNLOAD IN PROGRESS''?
3890  016322  003402                          BLE     1$                      ;BRANCH IF NO
3891  016324  105061  012316                  CLRB    ULDFLG(R1)              ;CLEAR UNLOAD FLAG
3892  016330  105061  012240          1$:     CLRB    DRVACT(R1)              ;SET DRIVE IDLE
3893  016334  136137  012364  012312          BITB    ATABIT(R1),SRCHWT   ;DOING A SEARCH OPERATION FOR
3894                                                                          ;AN I/O COMMAND?
3895  016342  001012                          BNE     2$                      ;BRANCH IF YES
3896  016344  004737  020574                  JSR     PC,POPQUE               ;REMOVE REQUEST FROM QUEUE
3897  016350  052762  000200  000016          BIS     #BIT07,16(R2)           ;SET ''DONE'' BIT
3898  016356  005737  012336                  TST     SAVEFG                  ;SAVE THE REGISTERS?
3899  016362  100002                          BPL     2$                      ;BRANCH IF NO
3900  016364  004737  017716                  JSR     PC,SVRH11               ;YES--SAVE ALL OF THE RH11/RP04/5/6 REG'S
3901  016370  116164  012364  00001o  2$:     MOVB    ATABIT(R1),RPAS(R4)  ;CLEAR ATTENTION BIT
3902  016376  004737  013456                  JSR     PC,OPT                  ;START A REQUEST
3903  016402  000137  015540                  JMP     SC4                     ;CHECK FOR MORE DRIVES
3904  016406  010164  000010          SC12:   MOV     R1,RPCS2(R4)            ;SELECT DRIVE
3905  016412  016437  000012  012230          MOV     RPDS1(R4),RPERRS        ;SAVE THE FOUR REGISTERS THAT
3906  016420  016437  000014  012232          MOV     RPER1(R4),RPERRS+2      ;WILL TELL US SOMETHING
3907  016426  016437  000040  012234          MOV     RPER2(R4),RPERRS+4
3908  016434  016437  000042  012236          MOV     RPER3(R4),RPERRS+6
3909  016442  004037  012626                  JSR     R0,DRVINT               ;INIT. THE STATE OF THE DRIVE
3910  016446  000401                          BR      1$                      ;TAKE ERROR EXIT
3911  016450  000207                          RTS     PC                      ;RETURN
3912  016452  005726                  1$:     TST     (SP)+                   ;POP PC OFF OF THE STACK
3913  016454  000704                          BR      SC8                     ;PROCESS THE PARITY ERROR
3914  016456  006301                  SC13:   ASL     R1                      ;SETUP TO ADDRESS WORDS
3915  016460  012761  177777  012342          MOV     #-1,TIMER(R1)           ;STOP THE TIMER
3916  016466  006201                          ASR     R1                      ;
3917  016470  010164  000010                  MOV     R1,RPCS2(R4)            ;SELECT THE DRIVE
3918  016474  116164  012364  000016          MOVB    ATABIT(R1),RPAS(R4)  ;CLEAR THE ATTENTION BIT
3919  016502  032714  004000                  BIT     #BIT11,(R4)             ;DRIVE AVAILABLE ?
3920  016506  001006                          BNE     1$                      ;BR IF AVAILABLE
3921  016510  006301                          ASL     R1
3922
```

```
3923  016512  012761  023420  012342          MOV     #10000.,TIMER(R1)  ;START 10 SEC TIMER AGAIN
3924
3925  016520  006201                          ASR     R1
3926  016522  000433                          BR      3$                 ;EXIT
3927  016524  105761  012270          1$:     TSTB    DPINT(R1)          ;INITIALIZING THE DRIVE ?
3928  016530  001424                          BEQ     2$                 ;BR IF NOT
3929  016532  105061  012270                  CLRB    DPINT(R1)          ;CLEAR THE INIT INDICATOR
3930  016536  004037  012626                  JSR     R0,DRVINT          ;GO INIT THE DRIVE
3931  016542  000240                          NOP                        ;DUMMY PARITY ERROR RETURN
3932  016544  105761  012250                  TSTB    DRVSTA(R1)         ;DRIVE ONLINE ?
3933  016550  003014                          BGT     2$                 ;BR IF YES -- START ORDER
3934  016552  005702                          TST     R2                 ;QUEUE ENTRY FOR THE DRIVE
3935  016554  001416                          BEQ     3$                 ;BR IF NOT
3936  016556  004737  020552                  JSR     PC,GETREQ          ;GET DPB ADDRESS
3937  016562  052762  140000  000016          BIS     #BIT15!BIT14,16(R2)  ;INFORM USER THAT DRIVE OFFLINE
3938  016570  004737  017716                  JSR     PC,SVRH11          ;SAVE THE REGISTERS
3939  016574  004737  020456                  JSR     PC,EMPTYQ          ;EMPTY THE REQUEST QUEUE
3940  016600  000404                          BR      3$
3941  016602  105061  012300          2$:     CLRB    DPRQS(R1)          ;CLEAR THE PORT REQUEST INDICATOR
3942  016606  004737  013456                  JSR     PC,OPT             ;START THE PENDING REQUEST
3943  016612  000137  015540          3$:     JMP     SC4                ;PROCESS OTHER DRIVES
3944
3945                                  ;RP04/5/6 TIMER ROUTINE
3946                                  ;CALL
3947                                  ;       MOV     #TIME,-(SP)        ;ELASPED TIME IN MILLISECONDS ON THE STACK
3948                                  ;       JSR     PC,RPTMR           ;CALL RP04/5/6 TIME ROUTINE
3949
3950  016616  005737  012314          RPTMR:  TST     ACTDRV             ;CHECK "ACTDRV & ACTSTR"
3951  016622  001030                          BNE     4$                 ;IF NON ZERO EXIT
3952  016624  112737  000001  012315          MOVB    #1,ACTSTR          ;SET "ACTSTR"
3953  016632  104412                          SAVREG                     ;SAVE R0 - R5
3954  016634  005001                          CLR     R1                 ;START WITH DRIVE 0
3955  016636  005003                          CLR     R3
3956  016640  005763  012342          1$:     TST     TIMER(R3)          ;IS THE TIMER RUNNING?
3957  016644  002407                          BLT     2$                 ;BRANCH IF NO
3958  016646  166663  000002  012342          SUB     2(SP),TIMER(R3)    ;COUNT THE INTERVAL
3959  016654  003003                          BGT     2$                 ;BR IF NO SOFTWARE TIMEOUT
3960  016656  004737  016710                  JSR     PC,STO             ;CALL SOFTWARE TIMEOUT ROUTINE
3961  016662  000405                          BR      3$                 ;GO TO THE EXIT
3962  016664  005201                  2$:     INC     R1                 ;MOVE TO NEXT DRIVE
3963  016666  005723                          TST     (R3)+
3964  016670  022701  000010                  CMP     #8.,R1             ;OUT OF DRIVES?
3965  016674  003361                          BGT     1$                 ;BRANCH IF NO
3966  016676  104413                  3$:     RESREG                     ;RESTORE R0 - R5
3967  016700  105037  012315                  CLRB    ACTSTR             ;ZERO ACTIVE SOFTWARE TIMEOUT ROUTINE FLAG
3968  016704  012616                  4$:     MOV     (SP)+,(SP)         ;ADJUST THE STACK
3969  016706  000207                          RTS     PC                 ;RETURN
3970
3971                                  ;SOFTWARE TIMEOUT ROUTINE
3972                                  ;
3973                                  ;NOTE: THIS ROUTINE MUST BE ENTERED AT PRIORITY 6
3974                                  ;       OR GREATER
3975                                  ;
3976                                  ;CALL:  STO
3977                                  ;       MOV     #DRVNUM,R1         ;DRIVE NUMBER
3978                                  ;       JSR     PC,STO             ;CALL
```

```
                                        ;       RETURN
                                        STO:    MOV     R1,-(SP)        ;SAVE R1
                                                MOV     R3,-(SP)        ;SAVE R3
                                                MOV     RPADR,R4        ;GET ADDRESS OF 'RPCS1'
                                                MOV     R1,RPCS2(R4)    ;SELECT THE DRIVE
                                                JSR     R0,RD.RP        ;READ 'DRIVE STATUS REG'
                                                RPDS1
                                                STO5
                                                TSTB    (SP)+           ;IS 'DRY'=1?
                                                BMI     STO2            ;BR IF YES
                                        STO1:   TSTB    DPINT(R1)       ;TRYING TO INTIALIZE THE DRIVE ?
                                                BNE     STO2            ;BR IF YES
                                                TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
                                                BNE     STO2            ;BR IF YES
                                                MOV     TRNSWT,R2       ;PICKUP TRANSFER WAIT QUEUE
                                                CMP     R1,DTUW         ;TRANSFER UNDERWAY ON THIS DRIVE?
                                                BEQ     1$              ;BRANCH IF YES
                                                JSR     PC,GETREQ       ;GET DPB ADDRESS
                                        1$:     BIS     #BIT15!BIT09,16(R2) ;SET THE ERROR FLAGS
                                                JSR     PC,SVRH11       ;SAVE RH11/RP04/5/6 REGISTERS
                                                MOV     #BIT05,RPCS2(R4) ;"INIT" THE MASS BUS
                                                CLRB    DRVACT(R1)      ;DRIVE IS IDLE
                                                CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD FLAG
                                                CLR     R1              ;START WITH DRIVE 0
                                                CLR     R3
                                        2$:     JSR     R0,DRVINT       ;INIT. THIS DRIVE
                                                BR      STO5            ;PARITY ERROR RETURN
                                                TSTB    DRVACT(R1)      ;DRIVE IDLE BEFORE THE INIT.?
                                                BEQ     4$              ;YES--BRANCH
                                                MOV     TRNSWT,R2       ;GET TRANSFER WAIT QUEUE
                                                CMP     DTUW,R1         ;WAS THERE I/O ON THIS DRIVE?
                                                BEQ     3$              ;YES--BRANCH
                                                JSR     PC,GETREQ       ;GET THE DPB POINTER FROM QUEUE
                                        3$:     BIS     #BIT15!BIT08,16(R2) ;INFORM USER OF INIT.
                                                CLRB    DRVACT(R1)      ;SET DRIVE ACTIVE TO IDLE
                                        4$:     CLRB    ULDFLG(R1)      ;NO UNLOAD
                                                MOV     #-1,TIMER(R3)   ;STOP THE TIMER
                                                TST     (R3)+           ;UPDATE THE INDEX
                                                INC     R1              ;INCREMENT THE DRIVE NUMBER
                                                CMP     #8.,R1          ;LAST DRIVE BEEN CHECKED?
                                                BGT     2$              ;NO--LOOP
                                                MOV     #-1,DTUW        ;NO DATA TRANSFERS UNDERWAY
                                                CLR     TRNSWT          ;CLEAR TRANSFER WAIT QUEUE
                                                JSR     PC,CLRQUE       ;CLEAR ALL REQUEST QUEUES
                                                BR      STO9            ;EXIT
                                        STO2:   MOVB    RPAS(R4),R5     ;READ ATTENTION REG
                                                BITB    ATABIT(R1),R5   ;IS ATTENTION FOR THIS DRIVE UP ?
                                                BNE     STO3            ;YES--BRANCH
                                                TSTB    DPINT(R1)       ;TRYING TO INTIALIZE THE DRIVE ?
                                                BNE     STO6            ;BR IF YES - DRIVE NOT ONLINE
                                                TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
                                                BNE     STO7            ;BR IF YES - NO RESPONSE TO REQUEST
                                                CMP     R1,DTUW         ;DATA TRANSFER UNDERWAY FOR THIS DRIVE
                                                BNE     STO1            ;BR IF NO
                                                JSR     R0,RD.RP        ;YES--CHECK 'RDY'
```

```
 4035  017176  000000                          RPCS1
 4036  017200  017232                          STO5
 4037  017202  105726                          TSTB    (SP)+
 4038  017204  100255                          BPL     STO1            ;BR IF 'RDY'=0
 4039  017206  105761  012270          STO3:   TSTB    DPINT(R1)       ;INITIALIZING THE DRIVE ?
 4040  017212  001003                          BNE     1$              ;BR IF INIT PENDING
 4041  017214  105761  012300                  TSTB    DPRQS(R1)       ;PORT REQUEST PENDING ?
 4042  017220  001446                          BEQ     STO9            ;BR IF NOT
 4043  017222  012763  177777  012342  1$:     MOV     #-1,TIMER(R3)   ;STOP THE TIMER
 4044  017230  000442                          BR      STO9            ;EXIT
 4045  017232  004737  014666          STO5:   JSR     PC,CI8          ;GO HANDLE THE PARITY ERROR
 4046  017236  000437                          BR      STO9
 4047  017240  105061  012270          STO6:   CLRB    DPINT(R1)       ;CLEAR THE INITIALIZE INDICATOR
 4048  017244  105061  012250                  CLRB    DRVSTA(R1)      ;SET UNIT OFFLINE
 4049  017250  012763  177777  012342          MOV     #-1,TIMER(R3)  ;STOP THE TIMER
 4050  017256  004737  020552                  JSR     PC,GETREQ       ;GET THE DPB ADDRESS
 4051  017262  005702                          TST     R2              ;REQUEST IN QUEUE ?
 4052  017264  001424                          BEQ     STO9            ;BR IF NOT
 4053  017266  052762  140000  000016          BIS     #BIT15!BIT14,16(R2)  ;INFORM THE USER DRIVE NOT AVAILABLE
 4054  017274  000414                          BR      STO8            ;FINISH
 4055  017276  012763  177777  012342  STO7:   MOV     #-1,TIMER(R3)   ;STOP THE TIMER
 4056  017304  105061  012300                  CLRB    DPRQS(R1)       ;CLEAR PORT REQUEST INDICATOR
 4057  017310  004737  020552                  JSR     PC,GETREQ       ;GET DPB ADDRESS
 4058  017314  005702                          TST     R2              ;QUEUE ENTRY FOR DRIVE ?
 4059  017316  001407                          BEQ     STO9            ;BR IF NONE
 4060  017320  012762  100004  000016          MOV     #BIT15!BIT2,16(R2)  ;INFORM USER OF PORT REQUEST ERROR
 4061  017326  004737  020456          STO8:   JSR     PC,EMPTYQ       ;CLEAR THE QUEUE FOR THE DRIVE
 4062  017332  004737  017716                  JSR     PC,SVRH11       ;SAVE THE REGISTERS
 4063  017336  012603                  STO9:   MOV     (SP)+,R3        ;RESTORE R3
 4064  017340  012601                          MOV     (SP)+,R1        ;RESTORE R1
 4065  017342  000207                          RTS     PC              ;RETURN
 4066
 4067                                  ;ROUTINE TO READ A RH11/RP04/5/6 REGISTER
 4068                                  ;
 4069                                  ;CALL
 4070                                  ;       JSR     R0,RD.RP        ;GO READ A REGISTER
 4071                                  ;       INDEX                   ;REG. INDEX FROM BASE
 4072                                  ;       ERRADR                  ;ERROR ADDRESS--PROCESS ERROR STARTING
 4073                                  ;                               ;AT THIS ADDRESS
 4074                                  ;       RETURN                  ;CONTENTS OF REG. IS ON THE STACK
 4075                                  ;
 4076  017344  013737  012374  017512  RD.RP:   MOV     MCPEMX,RD.RP2   ;MAX. RETRYS ALLOWED
 4077  017352  011646                          MOV     (SP),-(SP)      ;SAVE R0 FOR RETURN
 4078  017354  013737  012376  017370          MOV     RPADR,RD.ADR    ;FORM THE DESIRED ADDRESS
 4079  017362  062037  017370                  ADD     (R0)+,RD.ADR    ;USING THE BASE AND THE INDEX
 4080  017366  013727                  RD.RP1: MOV     @(PC)+,(PC)+    ;READ THE DESIRED REGISTER OF THE RP04
 4081  017370  000000                  RD.ADR: .WORD   0               ;ADDRESS IS FORMED HERE
 4082  017372  000000                  RD.WRD: .WORD   0               ;REG. CONTENTS PUT HERE
 4083  017374  013766  017372  000002          MOV     RD.WRD,2(SP)    ;RETURN IT TO THE USER
 4084  017402  013746  012376                  MOV     RPADR,-(SP)     ;PUT THE ADDRESS ON THE STACK
 4085  017406  062716  000010                  ADD     #RPCS2,(SP)     ;FORM THE ADDRESS OF RPCS2
 4086  017412  032736  010000                  BIT     #BIT12,@(SP)+   ;CHECK THE 'NED' BIT
 4087  017416  001037                          BNE     RD.RP3          ;BR IF DRIVE NON-EXISTENT
 4088  017420  017746  172752                  MOV     @RPADR,-(SP)    ;READ RPCS1
 4089  017424  032716  020000                  BIT     #BIT13,(SP)     ;DID MCPE SET?
 4090  017430  001002                          BNE     1$              ;BRANCH IF YES
```

```
4091  017432  022620                          CMP     (SP)+,(R0)+     ;ADJUST FOR RETURN
4092  017434  000432                          BR      RD.RP4          ;EXIT
4093  017436                          1$:
4094  017436  004037  020636                  JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
4095  017442  104003                          ERROR   3               ;REPORT 'MCPE' ERROR
4096  017444  005737  012362                  TST     DTUW            ;DATA TRANSFER UNDERWAY?
4097  017450  100405                          BMI     2$              ;NO--BRANCH
4098  017452  032716  040000                  BIT     #BIT14,(SP)     ;NO--'TRE'=1?
4099  017456  001402                          BEQ     2$              ;NO--BRANCH
4100  017460  005726                          TST     (SP)+           ;YES--CLEAN OFF THE STACK AND
4101  017462  000415                          BR      RD.RP3          ;TAKE THE FATAL ERROR EXIT
4102  017464  052716  040000          2$:     BIS     #BIT14,(SP)     ;CLEAR 'MCPE' BY SENDING A '1' TO 'TRE'
4103  017470  000316                          SWAB    (SP)            ;POSITION BEFORE WRITING
4104  017472  013737  012376  017506          MOV     RPADR,3$        ;FORM ADDRESS OF HIGH BYTE
4105  017500  005237  017506                  INC     3$
4106  017504  112637                          MOVB    (SP)+,@(PC)+    ;WRITE THE HIGH BYTE OF RPCS1
4107  017506  000000                  3$:     .WORD   0               ;ADDRESS STORAGE
4108  017510  005327                          DEC     (PC)+           ;EXCEEDED MAX. RETRYS
4109  017512  000003                  RD.RP2: .WORD   3
4110  017514  002324                          BGE     RD.RP1          ;BRANCH IF NO
4111  017516  011000                  RD.RP3: MOV     (R0),R0         ;FATAL ERROR EXIT
4112  017520  012616                          MOV     (SP)+,(SP)
4113  017522  000200                  RD.RP4: RTS     R0
4114
4115                                  ;ROUTINE TO WRITE A REGISTER
4116                                  ;
4117                                  ;CALL
4118                                  ;       MOV     DATA,-(SP)      ;DATA TO BE LOADED ON THE STACK
4119                                  ;       JSR     R0,WRT.RP       ;CALL THE ROUTINE TO LOAD(WRITE) THE REG.
4120                                  ;       INDEX                   ;INDEX OF THE REGISTER TO BE LOADED
4121                                  ;       ERRADR                  ;ADDRESS TO RETURN TO ON AN ERROR
4122                                  ;       RETURN                  ;ERROR FREE RETURN
4123
4124  017524  013737  012374  017700  WRT.RP: MOV     MCPEMX,WRT.R2   ;MAX RETRYS ALLOWED
4125  017532  016637  000002  017612          MOV     2(SP),WRT.WD    ;SAVE THE WORD TO WRITE
4126  017540  012616                          MOV     (SP)+,(SP)      ;ADJUST THE STACK
4127  017542  012037  017614                  MOV     (R0)+,WRT.AD    ;GET INDEX OF REGISTER TO BE WRITTEN
4128  017546  001015                          BNE     1$              ;BRANCH IF NOT RPCS1
4129  017550  122737  000150  017612          CMPB    #150,WRT.WD     ;IS THE COMMAND FOR DATA TRANSFERS?
4130  017556  002411                          BLT     1$              ;YES--DON'T GET THE OLD A16 & A17, & PSEL
4131  017560  004037  017344                  JSR     R0,RD.RP        ;NO---COMBINE A16&A17, & PSEL WITH
4132  017564  000000                          RPCS1                   ;THE COMMAND BEFORE SENDING IT TO
4133  017566  017706                          WRT.R3                  ;THE RH11/RP04
4134  017570  000316                          SWAB    (SP)
4135  017572  042716  177770                  BIC     #^C7,(SP)
4136  017576  112637  017613                  MOVB    (SP)+,WRT.WD+1
4137  017602  063737  012376  017614  1$:     ADD     RPADR,WRT.AD    ;FORM THE ADDRESS OF THE DISK REG.
4138  017610  012737                  WRT.R1: MOV     (PC)+,@(PC)+    ;LOAD THE DESIRED REG.
4139  017612  000000                  WRT.WD: .WORD   0               ;WORD TO WRITE GOES HERE
4140  017614  000000                  WRT.AD: .WORD   0               ;ADDRESS IS FORMED HERE
4141  017616  013746  012376                  MOV     RPADR,-(SP)     ;PUT THE ADDRESS ON THE STACK
4142  017622  062716  000010                  ADD     #RPCS2,(SP)     ;FORM THE ADDRESS OF RPCS2
4143  017626  032736  010000                  BIT     #BIT12,@(SP)+   ;CHECK THE 'NED' BIT
4144  017632  001025                          BNE     WRT.R3          ;BR IF DRIVE NON-EXISTENT
4145  017634  004037  017344                  JSR     R0,RD.RP        ;CHECK FOR PARITY ERROR ON WRITE
4146  017640  000014                          RPER1
```

```
 4147  017642  017706                          WRT.R3
 4148  017644  032726  000010                  BIT     #BIT03,(SP)+
 4149  017650  001420                          BEQ     WRT.R4          ;BRANCH IF 'PAR=0''
 4150  017652  016037  177776  017664          MOV     -2(R0),1$       ;PICKUP THE INDEX
 4151  017660  004037  017344                  JSR     R0,RD.RP        ;READ THE REG.
 4152  017664  000000                  1$:     .WORD   0               ;REG. INDEX
 4153  017666  017706                          WRT.R3                  ;RETURN TO THIS ADDRESS ON ERROR
 4154  017670  004037  020636                  JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
 4155  017674  104004                          ERROR   4               ;REPORT THE PARITY ON WRITE ERROR
 4156  017676  005327                          DEC     (PC)+           ;DECREMENT THE ERROR COUNT
 4157  017700  000003                  WRT.R2: .WORD   3               ;RETRY COUNTER
 4158  017702  002342                          BGE     WRT.R1          ;TRY AGAIN IF NOT FINISHED
 4159  017704  005726                          TST     (SP)+           ;CLEAN OFF THE STACK
 4160  017706  011000                  WRT.R3: MOV     (R0),R0         ;TAKE THE 'PARITY ON WRITE'' ERROR EXIT
 4161  017710  000401                          BR      WRT.R5          ;EXIT
 4162  017712  005720                  WRT.R4: TST     (R0)+           ;ADJUST FOR ERROR FREE EXIT
 4163  017714  000200                  WRT.R5: RTS     R0
 4164
 4165                                  ;ROUTINE TO SAVE THE RH11/RP04/5/6 REGISTERS AS PER DPB+14
 4166                                  ;
 4167                                  ;CALL
 4168                                  ;       MOV     #DPBNUM,R2      ;DPB POINTER TO R2
 4169                                  ;       JSR     PC,SVRH11       ;SAVE THE DRIVES REG'S
 4170
 4171  017716  104412                  SVRH11: SAVREG                  ;SAVE R0 - R5
 4172  017720  005702                          TST     R2              ;QUEUE ENTRY FOR THE DRIVE ?
 4173  017722  001430                          BEQ     4$              ;BR IF NONE
 4174  017724  013704  012376                  MOV     RPADR,R4
 4175  017730  111264  000010                  MOVB    (R2),RPCS2(R4)  ;SELECT DRIVE
 4176  017734  016203  000014                  MOV     14(R2),R3       ;GET THE ERROR TABLE POINTER
 4177  017740  001433                          BEQ     6$              ;EXIT IF NO ADDRESS
 4178  017742  005037  017776                  CLR     3$              ;COUNTER & POINTER
 4179  017746  023727  017776  000022  1$:     CMP     3$,#RPDB        ;REACHED THE BUFFER REGISTER ?
 4180  017754  001006                          BNE     2$              ;BR IF NOT
 4181  017756  032764  000200  000010          BIT     #BIT07,RPCS2(R4)  ;'OR' SET ?
 4182  017764  001002                          BNE     2$              ;BR IF SET
 4183  017766  005023                          CLR     (R3)+           ;STORE RPDB AS ZEROES
 4184  017770  000405                          BR      4$              ;CONTINUE
 4185  017772  004037  017344          2$:     JSR     R0,RD.RP        ;READ THE SELECTED REGISTER
 4186  017776  000000                  3$:     .WORD   0               ;REGISTER INDEX
 4187  020000  020024                          5$                      ;ERROR RETURN ADDRESS
 4188  020002  012623                          MOV     (SP)+,(R3)+     ;STORE THE REGISTER CONTENTS
 4189  020004  023727  017776  000046  4$:     CMP     3$,#RPEC2       ;REACHED THE END ?
 4190  020012  001406                          BEQ     6$              ;BR IF YES
 4191  020014  062737  000002  017776          ADD     #2,3$           ;INCREMENT THE REGISTER INDEX
 4192  020022  000751                          BR      1$              ;CONTINUE READING THE REGISTERS
 4193  020024  004737  014566          5$:     JSR     PC,C17          ;PROCESS THE UNCORRECTABLE PARITY ERROR
 4194  020030  104413                  6$:     RESREG                  ;RESTORE R0 - R5
 4195  020032  000207                          RTS     PC              ;RETURN
 4196
 4197                                  ;ROUTINE TO SET THE INTERRUPT WITHOUT GETTING A 'TRE''
 4198                                  ;CALL
 4199                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
 4200                                  ;       JSR     PC,SET.IE       ;SET ''IE''
 4201                                  ;       RETURN
 4202
```

```
 4203  020034  010446                  SET.IE: MOV     R4,-(SP)          ;SAVE R4
 4204  020036  013704  012376                  MOV     RPADR,R4          ;PICKUP ADDRESS OF RPCS1
 4205  020042  010164  000010                  MOV     R1,RPCS2(R4)      ;SELECT DRIVE
 4206  020046  011446                          MOV     (R4),-(SP)        ;READ RPCS1
 4207  020050  052716  040000                  BIS     #BIT14,(SP)       ;SET THE 'TRE' BIT OF THE WORD READ
 4208  020054  000316                          SWAB    (SP)              ;ADJUST FOR DATO
 4209  020056  112714  000100                  MOVB    #BIT06,(R4)       ;SET 'IE'
 4210  020062  032764  010000  000010          BIT     #BIT12,RPCS2(R4)  ;IS 'NED'=1?
 4211  020070  001002                          BNE     1$                ;YES--CLEAR 'TRE'
 4212  020072  005726                          TST     (SP)+             ;CLEAN OFF THE STACK
 4213  020074  000402                          BR      2$
 4214  020076  112664  000001          1$:     MOVB    (SP)+,1(R4)       ;CLEAR 'TRE'
 4215  020102  012604                  2$:     MOV     (SP)+,R4          ;RESTORE R4
 4216  020104  000207                          RTS     PC                ;RETURN TO CALLER
 4217
 4218                                  ;QUEUE COUNT
 4219  020106      000                 QCNT:   .BYTE   0                 ;DRIVE 0
 4220  020107      000                         .BYTE   0                 ;DRIVE 1
 4221  020110      000                         .BYTE   0                 ;DRIVE 2
 4222  020111      000                         .BYTE   0                 ;DRIVE 3
 4223  020112      000                         .BYTE   0                 ;DRIVE 4
 4224  020113      000                         .BYTE   0                 ;DRIVE 5
 4225  020114      000                         .BYTE   0                 ;DRIVE 6
 4226  020115      000                         .BYTE   0                 ;DRIVE 7
 4227
 4228                                  ;QUEUE INPUT POINTERS
 4229
 4230  020116  020200                  QINPT:  .WORD   QDRV0              ;DRIVE 0
 4231  020120  020220                          .WORD   QDRV1             ;DRIVE 1
 4232  020122  020240                          .WORD   QDRV2             ;DRIVE 2
 4233  020124  020260                          .WORD   QDRV3             ;DRIVE 3
 4234  020126  020300                          .WORD   QDRV4             ;DRIVE 4
 4235  020130  020320                          .WORD   QDRV5             ;DRIVE 5
 4236  020132  020340                          .WORD   QDRV6             ;DRIVE 6
 4237  020134  020360                          .WORD   QDRV7             ;DRIVE 7
 4238
 4239                                  ;QUEUE OUTPUT POINTERS
 4240
 4241  020136  020200                  QOUTPT: .WORD   QDRV0              ;DRIVE 0
 4242  020140  020220                          .WORD   QDRV1             ;DRIVE 1
 4243  020142  020240                          .WORD   QDRV2             ;DRIVE 2
 4244  020144  020260                          .WORD   QDRV3             ;DRIVE 3
 4245  020146  020300                          .WORD   QDRV4             ;DRIVE 4
 4246  020150  020320                          .WORD   QDRV5             ;DRIVE 5
 4247  020152  020340                          .WORD   QDRV6             ;DRIVE 6
 4248  020154  020360                          .WORD   QDRV7             ;DRIVE 7
 4249
 4250  020156  020200                  QSTART: .WORD   QDRV0              ;DRIVE 0 START ADDRESS
 4251  020160  020220                  QSTOP:  .WORD   QDRV1              ;DRIVE 0 STOP ADDRESS & DRIVE 1 START ADDRESS
 4252  020162  020240                          .WORD   QDRV2             ;STOP DRIVE 1--START DRIVE 2
 4253  020164  020260                          .WORD   QDRV3             ;STOP DRIVE 2--START DRIVE 3
 4254  020166  020300                          .WORD   QDRV4             ;STOP DRIVE 3--START DRIVE 4
 4255  020170  020320                          .WORD   QDRV5             ;STOP DRIVE 4--START DRIVE 5
 4256  020172  020340                          .WORD   QDRV6             ;STOP DRIVE 5--START DRIVE 6
 4257  020174  020360                          .WORD   QDRV7             ;STOP DRIVE 6--START DRIVE 7
 4258  020176  020400                          .WORD   QTERM             ;STOP DRIVE 7
```

```
 4259
 4260                                   ;DRIVE REQUEST QUEUES
 4261
 4262  020200  000010                   QDRV0:  .BLKW   10
 4263  020220  000010                   QDRV1:  .BLKW   10
 4264  020240  000010                   QDRV2:  .BLKW   10
 4265  020260  000010                   QDRV3:  .BLKW   10
 4266  020300  000010                   QDRV4:  .BLKW   10
 4267  020320  000010                   QDRV5:  .BLKW   10
 4268  020340  000010                   QDRV6:  .BLKW   10
 4269  020360  000010                   QDRV7:  .BLKW   10
 4270          020400                   QTERM=.
 4271
 4272                                   ;ROUTINE TO CLEAR ALL OF THE REQUEST QUEUES
 4273                                   ;
 4274                                   ;CALL
 4275                                   ;       JSR     PC,CLRQUE
 4276
 4277  020400  104412                   CLRQUE: SAVREG                  ;SAVE R0 - R5
 4278  020402  012702  020106                   MOV     #QCNT,R2        ;ZERO THE QUEUE COUNTS
 4279  020406  005022                           CLR     (R2)+           ;DRIVES 0 & 1
 4280  020410  005022                           CLR     (R2)+           ;DRIVES 2 & 3
 4281  020412  005022                           CLR     (R2)+           ;DRIVES 4 & 5
 4282  020414  005022                           CLR     (R2)+           ;DRIVES 6 & 7
 4283  020416  012703  000010                   MOV     #8.,R3          ;MOVE THE STARTING
 4284  020422  012701  020156                   MOV     #QSTART,R1      ;ADDRESS OF THE QUEUE INTO
 4285  020426  012122                   1$:     MOV     (R1)+,(R2)+     ;THE QUEUE INPUT POINTER
 4286  020430  005303                           DEC     R3
 4287  020432  001375                           BNE     1$
 4288  020434  012703  000010                   MOV     #8.,R3          ;MOVE THE STARTING ADDRESS
 4289  020440  012701  020156                   MOV     #QSTART,R1      ;OF THE QUEUE INTO THE
 4290  020444  012122                   2$:     MOV     (R1)+,(R2)+     ;QUEUE OUTPUT POINTER
 4291  020446  005303                           DEC     R3
 4292  020450  001375                           BNE     2$
 4293  020452  104413                           RESREG                  ;RESTORE R0 - R5
 4294  020454  000207                           RTS     PC
 4295
 4296                                   ;EMPTY THE QUEUE SPECIFIED BY R1
 4297                                   ;
 4298                                   ;CALL
 4299                                   ;       MOV     DRVNUM,R1       ;DRIVE NUMBER TO R1
 4300                                   ;       JSR     PC,EMPTYQ
 4301
 4302  020456  105061  020106           EMPTYQ: CLRB    QCNT(R1)        ;CLEAR NUMBER OF ITEMS IN QUEUE
 4303  020462  006301                           ASL     R1
 4304  020464  016161  020116  020136           MOV     QINPT(R1),QOUTPT(R1) ;SET OUTPUT QUEUE POINTER-INPUT POINTER
 4305  020472  006201                           ASR     R1
 4306  020474  000207                           RTS     PC
 4307
 4308                                   ;ROUTINE TO PUT A REQUEST IN QUEUE
 4309                                   ;
 4310                                   ;CALL
 4311                                   ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
 4312                                   ;       MOV     #DPB,R2         ;ADDRESS OF PARAMETER BLOCK
 4313                                   ;       JSR     R0,DRVQUE       ;GO PUT REQUEST IN QUEUE
 4314                                   ;       RETURN1                 ;RETURN HERE IF QUEUE IS FULL
```

```
                                        ;       RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE
                                        ;
020476  122761  000010  020106  DRVQUE: CMPB    #10,QCNT(R1)            ;IS QUEUE FULL?
020504  001421                          BEQ     2$                      ;BR IF YES-TAKE RETURN1
020506  105261  020106                  INCB    QCNT(R1)                ;INCREMENT QUEUE COUNT
020512  006301                          ASL     R1
020514  010271  020116                  MOV     R2,@QINPT(R1)           ;PUT THIS REQUEST IN QUEUE
020520  062761  000002  020116          ADD     #2,QINPT(R1)            ;UPDATE THE QUEUE POINTER
020526  026161  020116  020160          CMP     QINPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER
020534  001003                          BNE     1$                      ;BRANCH IF NO
020536  016161  020156  020116          MOV     QSTART(R1),QINPT(R1) ;YES--RESET POINTER
020544  006201                  1$:     ASR     R1
020546  005720                          TST     (R0)+                   ;TAKE RETURN 2
020550  000200                  2$:     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
020552  005002                  GETREQ: CLR     R2
020554  105761  020106                  TSTB    QCNT(R1)                ;IS THERE ANY REQUEST IN QUEUE?
020560  001404                          BEQ     2$                      ;NO---BRANCH
020562  006301                  1$:     ASL     R1
020564  017102  020136                  MOV     @QOUTPT(R1),R2  ;PICKUP 'DPB' POINTER FOR THIS DRIVE
020570  006201                          ASR     R1
020572  000207                  2$:     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
020574  105361  020106          POPQUE: DECB    QCNT(R1)                ;DECREMENT QUEUE COUNT
020600  006301                          ASL     R1
020602  017102  020136                  MOV     @QOUTPT(R1),R2  ;GET THE 'DPB' POINTER
020606  062761  000002  020136          ADD     #2,QOUTPT(R1)   ;UPDATE THE QUEUE POINTER
020614  026161  020136  020160          CMP     QOUTPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER?
020622  001003                          BNE     1$                      ;NO--BRANCH TO EXIT
020624  016161  020156  020136          MOV     QSTART(R1),QOUTPT(R1) ;YES--RESET THE POINTER
020632  006201                  1$:     ASR     R1
020634  000207                          RTS     PC                      ;RETURN TO USER
                                        ;ROUTINE TO SAVE THE CONTENTS OF '$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
```

```
                                ES.SAV: MOV     (R0)+,1$        ;GET THE ERROR CALL
                                        MOV     $ESCAPE,-(SP)   ;SAVE THE ADDRESS IN '$ESCAPE'
                                        CLR     $ESCAPE         ;CLEAR THE ESCAPE RETURN
                                1$:     .WORD   0               ;THE ERROR CALL IS MOVED HERE
                                        MOV     (SP)+,$ESCAPE   ;RESTORE THE ESCAPE ADDRESS
                                        RTS     R0              ;RETURN

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

                                .SBTTL  TELETYPE MESSAGES

020662 047440 043106 044514     UNTOFF: .ASCIZ  / OFFLINE/
020670 042516    000
020673    040 047117 044514     UNTON:  .ASCIZ  / ONLINE/
020700 042516    000
020703    040 047516 020124     NOTRP:  .ASCIZ  @ NOT AN RP04/5/6@
020710 047101 051040 030120
020716 027464 027465 000066
020724 047040 052117 050040     NOTPRS: .ASCIZ  / NOT PRESENT/
020732 042522 042523 052116
020740    000
020741    040 047125 040523     NOTSAF: .ASCIZ  / UNSAFE/
020746 042506    000
020751    122 030120 000064     RP04B:  .ASCIZ  /RP04/
020756 050122 032460    000     RP05:   .ASCIZ  /RP05/
020763    122 030120 000066     RP06:   .ASCIZ  /RP06/
020770 047125 052111 051440     SYSTAT: .ASCIZ  /UNIT STATUS/<CR><LF><LF>
020776 040524 052524 006523
021004 005012    000
021007    015 042012 044522     MUNIT:  .ASCIZ  <CR><LF>/DRIVE: /
021014 042526 020072    000
021021    040 020057    000     SLASH:  .ASCIZ  @ / @
021025    103    000            C:      .ASCIZ  /C/
021027    124    000            T:      .ASCIZ  /T/
021031    060    000            MDRVD:  .ASCIZ  /0/
021033    040    040            LIN4SP: .ASCII  /  /
021035    040 000040            LINSP:  .ASCIZ  /  /
021040 047105 042524 020122     ENTADR: .ASCIZ  /ENTER ADDRESS LIMITS:/<CR><LF>
021046 042101 051104 051505
021054 020123 044514 044515
021062 051524 006472 000012
021070 020040 051104 053111     MOFFLN: .ASCIZ  /  DRIVE OFFLINE/<CR><LF>
021076 020105 043117 046106
021104 047111 006505 000012
021112 042040 044522 042526     MDRNP:  .ASCIZ  / DRIVE NOT PRESENT/<CR><LF>
021120 047040 052117 050040
021126 042522 042523 052116
021134 005015    000
021137    040 051104 053111     MER11:  .ASCIZ  / DRIVE NOT AVAILABLE/<CR><LF>
021144 020105 047516 020124
021152 053101 044501 040514
021160 046102 006505 000012
021166 042040 044522 042526     MNRP04: .ASCIZ  @ DRIVE NOT AN RP04/5/6@<CR><LF>
021174 047040 052117 040440
021202 020116 050122 032060
```

```
 4427  021210  032457  033057  005015
 4428  021216     000
 4429  021217     040  051104  053111  MUSDR:  .ASCIZ  / DRIVE UNSAFE/<CR><LF>
 4430  021224  020105  047125  040523
 4431  021232  042506  005015     000
 4432  021237     040  042523  042514  MSELD:  .ASCIZ  / SELECTED/
 4433  021244  052103  042105     000
 4434  021251     015  050012  047522  MMODE:  .ASCIZ  <CR><LF>/PROGRAM MODE (C OR F): /
 4435  021256  051107  046501  046440
 4436  021264  042117  020105  041450
 4437  021272  047440  020122  024506
 4438  021300  020072     000
 4439  021303     040  047506  046522  MFORMT: .ASCIZ  / FORMAT & VERIFY/
 4440  021310  052101  023040  053040
 4441  021316  051105  043111  000131
 4442  021324  044103  041505  020113  MHECK:  .ASCIZ  /CHECK ONLY/
 4443  021332  047117  054514     000
 4444  021337     106  051117  040515  MORMAT: .ASCIZ  /FORMAT & VERIFY/
 4445  021344  020124  020046  042526
 4446  021352  044522  054506     000
 4447  021357     015  005012  050117  MSIZE:  .ASCIZ  <CR><LF><LF>/OPERATE IN 22 SECTOR MODE (Y OR N) ? /
 4448  021364  051105  052101  020105
 4449  021372  047111  031040  020062
 4450  021400  042523  052103  051117
 4451  021406  046440  042117  020105
 4452  021414  054450  047440  020122
 4453  021422  024516  037440  000040
 4454  021430  050117  051105  052101  MSEC22: .ASCIZ  /OPERATION WILL BE IN 22 SECTOR (16 BIT) MODE/<CR><LF>
 4455  021436  047511  020116  044527
 4456  021444  046114  041040  020105
 4457  021452  047111  031040  020062
 4458  021460  042523  052103  051117
 4459  021466  024040  033061  041040
 4460  021474  052111  020051  047515
 4461  021502  042504  005015     000
 4462  021507     117  042520  040522  MSEC20: .ASCIZ  /OPERATION WILL BE IN 20 SECTOR (18 BIT) MODE/<CR><LF>
 4463  021514  044524  047117  053440
 4464  021522  046111  020114  042502
 4465  021530  044440  020116  030062
 4466  021536  051440  041505  047524
 4467  021544  020122  030450  020070
 4468  021552  044502  024524  046440
 4469  021560  042117  006505  000012
 4470  021566  047111  040526  044514  BADENT: .ASCIZ  /INVALID ENTRY/<CR><LF>
 4471  021574  020104  047105  051124
 4472  021602  006531  000012
 4473  021606  047105  044504  043516  MADRER: .ASCII  /ENDING DSK ADRS MUST BE EQUAL TO OR GREATER/<CR><LF>
 4474  021614  042040  045523  040440
 4475  021622  051104  020123  052515
 4476  021630  052123  041040  020105
 4477  021636  050505  040525  020114
 4478  021644  047524  047440  020122
 4479  021652  051107  040505  042524
 4480  021660  006522     012
 4481  021663     124  040510  020116          .ASCIZ  /THAN STARTING ADRS/<CR><LF>
 4482  021670  052123  051101  044524
```

```
 4483  021676  043516  040440  051104
 4484  021704  006523  000012
 4485  021710  042523  042514  052103  MSELP:  .ASCII  /SELECT DATA PATTERN/<CR><LF>
 4486  021716  042040  052101  020101
 4487  021724  040520  052124  051105
 4488  021732  006516     012
 4489  021735     040  030050  020051          .ASCII  / (0) ZERO'S/<CR><LF>
 4490  021742  042532  047522  051447
 4491  021750  005015
 4492  021752  024040  024461  047440          .ASCII  / (1) ONE'S/<CR><LF>
 4493  021760  042516  051447  005015
 4494  021766  024040  024462  053440          .ASCIZ  / (2) WORST CASE: /
 4495  021774  051117  052123  041440
 4496  022002  051501  035105  000040
 4497
 4498  022010  047527  051522  020124  ^PATD:  .ASCIZ  /WORST CASE/
 4499  022016  040503  042523     000
 4500  022023     015  005012  052123  MSFOU:  .ASCIZ  <CR><LF><LF>/STARTING FORMAT ON DRIVE /
 4501  022030  051101  044524  043516
 4502  022036  043040  051117  040515
 4503  022044  020124  047117  042040
 4504  022052  044522  042526  000040
 4505  022060  005015  051412  040524  MSCHK:  .ASCIZ  <CR><LF><LF>/STARTING CHECK ON DRIVE /
 4506  022066  052122  047111  020107
 4507  022074  044103  041505  020113
 4508  022102  047117  042040  044522
 4509  022110  042526  000040
 4510  022114  005015  043012  051117  MFCMPT: .ASCIZ  <CR><LF><LF>/FORMAT COMPLETE, /
 4511  022122  040515  020124  047503
 4512  022130  050115  042514  042524
 4513  022136  020054     000
 4514  022141     015  005012  044103  MCCMPT: .ASCIZ  <CR><LF><LF>/CHECK COMPLETE, /
 4515  022146  041505  020113  047503
 4516  022154  050115  042514  042524
 4517  022162  020054     000
 4518  022165     124  052117  046101  NUMERR: .ASCIZ  /TOTAL ERRORS DETECTED: /
 4519  022172  042440  051122  051117
 4520  022200  020123  042504  042524
 4521  022206  052103  042105  020072
 4522  022214     000
 4523  022215     120  042522  042523  ADDRIS: .ASCIZ  /PRESENT ADDRESS IS: /
 4524  022222  052116  040440  042104
 4525  022230  042522  051523  044440
 4526  022236  035123  000040
 4527
 4528                                   ;;*****************************************************************
 4529
 4530                                   .SBTTL  ERROR MESSAGES
 4531
 4532                                   ;;*****************************************************************
 4533
 4534  022242  044122  030461  044440  EM1:    .ASCIZ  /RH11 INTERRUPT OCCURRED (RPAS=0)/
 4535  022250  052116  051105  052522
 4536  022256  052120  047440  041503
 4537  022264  051125  042522  020104
 4538  022272  051050  040520  036523
```

```
 4539  022300  024460     000
 4540  022303     125  042516  050130  EM2:    .ASCIZ  /UNEXPECTED ATTENTION OCCURRED/
 4541  022310  041505  042524  020104
 4542  022316  052101  042524  052116
 4543  022324  047511  020116  041517
 4544  022332  052503  051122  042105
 4545  022340     000
 4546  022341     115  051501  041123  EM3:    .ASCIZ  /MASSBUS PARITY ERROR (MCPE=1)/
 4547  022346  051525  050040  051101
 4548  022354  052111  020131  051105
 4549  022362  047522  020122  046450
 4550  022370  050103  036505  024461
 4551  022376     000
 4552  022377     115  051501  041123  EM4:    .ASCIZ  /MASSBUS PARITY ERROR (PAR=1)/
 4553  022404  051525  050040  051101
 4554  022412  052111  020131  051105
 4555  022420  047522  020122  050050
 4556  022426  051101  030475  000051
 4557  022434  042101  051104  051505  EM5:    .ASCIZ  /ADDRESS PLUG CHANGE BIT SET/
 4558  022442  020123  046120  043525
 4559  022450  041440  040510  043516
 4560  022456  020105  044502  020124
 4561  022464  042523  000124
 4562  022470  044122  030461  042040  EM6:    .ASCIZ  /RH11 DIDN'T RESPOND TO ADDRESSING/
 4563  022476  042111  023516  020124
 4564  022504  042522  050123  047117
 4565  022512  020104  047524  040440
 4566  022520  042104  042522  051523
 4567  022526  047111  000107
 4568  022532  051104  053111  020105  EM10:   .ASCIZ  /DRIVE OFFLINE/
 4569  022540  043117  046106  047111
 4570  022546  000105
 4571  022550  042520  051522  051511  EM11:   .ASCIZ  /PERSISTENT DRIVE UNSAFE ERROR/
 4572  022556  042524  052116  042040
 4573  022564  044522  042526  052440
 4574  022572  051516  043101  020105
 4575  022600  051105  047522  000122
 4576  022606  047125  047503  051122  EM12:   .ASCIZ  /UNCORRECTABLE MASSBUS PARITY ERROR/
 4577  022614  041505  040524  046102
 4578  022622  020105  040515  051523
 4579  022630  052502  020123  040520
 4580  022636  044522  054524  042440
 4581  022644  051122  051117     000
 4582  022651     123  043117  053524  EM13:   .ASCIZ  /SOFTWARE TIMEOUT/
 4583  022656  051101  020105  044524
 4584  022664  042515  052517  000124
 4585  022672  051104  053111  020105  EM14:   .ASCIZ  /DRIVE UNSAFE ERROR/
 4586  022700  047125  040523  042506
 4587  022706  042440  051122  051117
 4588  022714     000
 4589  022715     103  047117  051124  EM15:   .ASCIZ  @CONTROLLER/DRIVE ERROR DURING WRITE@
 4590  022722  046117  042514  027522
 4591  022730  051104  053111  020105
 4592  022736  051105  047522  020122
 4593  022744  052504  044522  043516
 4594  022752  053440  044522  042524
```

```
 4595  022760     000
 4596  022761     103  047117  051124  EM16:   .ASCIZ  @CONTROLLER/DRIVE ERROR DURING WRITE CHECK@
 4597  022766  046117  042514  027522
 4598  022774  051104  053111  020105
 4599  023002  051105  047522  020122
 4600  023010  052504  044522  043516
 4601  023016  053440  044522  042524
 4602  023024  041440  042510  045503
 4603  023032     000
 4604  023033     122  052105  044522  EM17:   .ASCIZ  @RETRIES NOT SUCESSFUL - SECTOR NOT ACCEPTABLE@
 4605  023040  051505  047040  052117
 4606  023046  051440  041525  051505
 4607  023054  043123  046125  026440
 4608  023062  051440  041505  047524
 4609  023070  020122  047516  020124
 4610  023076  041501  042503  052120
 4611  023104  041101  042514     000
 4612  023111     104  052101  020101  EM20:   .ASCIZ  @DATA ERROR DURING WRITE CHECK@
 4613  023116  051105  047522  020122
 4614  023124  052504  044522  043516
 4615  023132  053440  044522  042524
 4616  023140  041440  042510  045503
 4617  023146     000
 4618  023147     103  047117  051124  EM21:   .ASCIZ  @CONTROLLER/DRIVE ERROR VERIFYING HEADERS@
 4619  023154  046117  042514  027522
 4620  023162  051104  053111  020105
 4621  023170  051105  047522  020122
 4622  023176  042526  044522  054506
 4623  023204  047111  020107  042510
 4624  023212  042101  051105  000123
 4625  023220  042510  042101  051105  EM22:   .ASCIZ  @HEADER COMPARE ERROR VERIFYING HEADERS@
 4626  023226  041440  046517  040520
 4627  023234  042522  042440  051122
 4628  023242  051117  053040  051105
 4629  023250  043111  044531  043516
 4630  023256  044040  040505  042504
 4631  023264  051522     000
 4632  023267     103  046131  047111  EM23:   .ASCIZ  @CYLINDER FIELD IN HEADER IS NOT CORRECT@
 4633  023274  042504  020122  044506
 4634  023302  046105  020104  047111
 4635  023310  044040  040505  042504
 4636  023316  020122  051511  047040
 4637  023324  052117  041440  051117
 4638  023332  042522  052103     000
 4639  023337     127  044522  042524  EM24:   .ASCIZ  @WRITE CHECK ERROR@
 4640  023344  041440  042510  045503
 4641  023352  042440  051122  051117
 4642  023360     000
 4643  023361     110  051101  053504  EM25:   .ASCIZ  @HARDWARE ERROR DURING WRITE CHECK@
 4644  023366  051101  020105  051105
 4645  023374  047522  020122  052504
 4646  023402  044522  043516  053440
 4647  023410  044522  042524  041440
 4648  023416  042510  045503     000
 4649          023424                          .EVEN
 4650
```

```
 4651                                 ;;*****************************************************************
 4652
 4653  023424  050122  051501     000  DH1:    .ASCIZ  /RPAS/
 4654  023431     104  044522  042526  DH2:    .ASCIZ  /DRIVE   RPDS1   RPER1   RPER2   RPER3   RPAS/
 4655  023436  020040  051040  042120
 4656  023444  030523  020040  051040
 4657  023452  042520  030522  020040
 4658  023460  051040  042520  031122
 4659  023466  020040  051040  042520
 4660  023474  031522  020040  051040
 4661  023502  040520  000123
 4662  023506  051104  053111  020105  DH3:    .ASCIZ  /DRIVE   REG ADR  DATA/
 4663  023514  020040  042522  020107
 4664  023522  042101  020122  042040
 4665  023530  052101  000101
 4666  023534  051104  053111  020105  DH4:    .ASCIZ  /DRIVE   REG ADR    GOOD    BAD/
 4667  023542  020040  042522  020107
 4668  023550  042101  020122  020040
 4669  023556  043440  047517  020104
 4670  023564  020040  041040  042101
 4671  023572     000
 4672  023573     122  020110  042101  DH6:    .ASCIZ  /RH ADDRESS/
 4673  023600  051104  051505  000123
 4674  023606  051104  053111  020105  DH10:   .ASCIZ  /DRIVE   ERR PC  RPCS1   RPCS2   RPDS1   RPER1   RPER2   RPER3/
 4675  023614  020040  051105  020122
 4676  023622  041520  020040  050122
 4677  023630  051503  020061  020040
 4678  023636  050122  051503  020062
 4679  023644  020040  050122  051504
 4680  023652  020061  020040  050122
 4681  023660  051105  020061  020040
 4682  023666  050122  051105  020062
 4683  023674  020040  050122  051105
 4684  023702  000063
 4685  023704  050122  041505  020061  DH10A:  .ASCIZ  /RPEC1   RPEC2   RPWC    RPBA    RPDA    RPAS    RPLA/
 4686  023712  020040  050122  041505
 4687  023720  020062  020040  050122
 4688  023726  041527  020040  020040
 4689  023734  050122  040502  020040
 4690  023742  020040  050122  040504
 4691  023750  020040  020040  050122
 4692  023756  051501  020040  020040
 4693  023764  050122  040514     000
 4694  023771     122  042120  020102  DH10B:  .ASCIZ  /RPDB    RPMR    RPDT    RPSN    RPOF    RPCA    RPCC/
 4695  023776  020040  051040  046520
 4696  024004  020122  020040  051040
 4697  024012  042120  020124  020040
 4698  024020  051040  051520  020116
 4699  024026  020040  051040  047520
 4700  024034  020106  020040  051040
 4701  024042  041520  020101  020040
 4702  024050  051040  041520  000103
 4703  024056  051104  053111  020105  DH17:   .ASCIZ  /DRIVE   ERR PC CYLINDER TRACK   SECTOR/
 4704  024064  020040  051105  020122
 4705  024072  041520  041440  046131
 4706  024100  047111  042504  020122
```

```
 4707  024106  051124  041501  020113
 4708  024114  020040  042523  052103
 4709  024122  051117     000
 4710  024125     104  044522  042526  DH20:   .ASCIZ  /DRIVE   ERR PC CYLINDER TRACK   SECTOR/
 4711  024132  020040  042440  051122
 4712  024140  050040  020103  054503
 4713  024146  044514  042116  051105
 4714  024154  052040  040522  045503
 4715  024162  020040  051440  041505
 4716  024170  047524  000122
 4717  024174  050122  051503  020061  DH20A:  .ASCIZ  /RPCS1   RPCS2   RPDS1   RPER1   RPER2   RPER3   RPEC1   RPEC2/
 4718  024202  020040  050122  051503
 4719  024210  020062  020040  050122
 4720  024216  051504  020061  020040
 4721  024224  050122  051105  020061
 4722  024232  020040  050122  051105
 4723  024240  020062  020040  050122
 4724  024246  051105  020063  020040
 4725  024254  050122  041505  020061
 4726  024262  020040  050122  041505
 4727  024270  000062
 4728  024272  050122  041527  020040  DH20B:  .ASCIZ  /RPWC    RPBA    RPDA    RPAS    RPLA    RPDB    RPMR    RPDT/
 4729  024300  020040  050122  040502
 4730  024306  020040  020040  050122
 4731  024314  040504  020040  020040
 4732  024322  050122  051501  020040
 4733  024330  020040  050122  040514
 4734  024336  020040  020040  050122
 4735  024344  041104  020040  020040
 4736  024352  050122  051115  020040
 4737  024360  020040  050122  052104
 4738  024366     000
 4739  024367     122  051520  020116  DH20C:  .ASCIZ  /RPSN    RPOF    RPCA    RPCC/
 4740  024374  020040  051040  047520
 4741  024402  020106  020040  051040
 4742  024410  041520  020101  020040
 4743  024416  051040  041520  000103
 4744  024424  020040  020040  020040  DH23:   .ASCII  /                EXPT'D/<CR><LF>
 4745  024432  020040  020040  020040
 4746  024440  020040  020040  054105
 4747  024446  052120  042047  005015
 4748  024454  051104  053111  020105          .ASCIZ  /DRIVE   ERR PC  CYLNDR  ACTUAL HEADER/
 4749  024462  020040  051105  020122
 4750  024470  041520  020040  054503
 4751  024476  047114  051104  020040
 4752  024504  041501  052524  046101
 4753  024512  044040  040505  042504
 4754  024520  000122
 4755  024522  020040  020040  020040  DH24:   .ASCII  /                                        MEMORY  DISK/<CR><LF>
 4756  024530  020040  020040  020040
 4757  024536  020040  020040  020040
 4758  024544  020040  020040  020040
 4759  024552  020040  020040  020040
 4760  024560  020040  020040  020040
 4761  024566  020040  020040  042515
 4762  024574  047515  054522  020040
```

```
 4763  024602  044504  045523  005015
 4764  024610  051104  053111  020105          .ASCIZ  /DRIVE   ERR PC  CYLNDR  TRACK   SECTOR  DATA    DATA/
 4765  024616  020040  051105  020122
 4766  024624  041520  020040  054503
 4767  024632  047114  051104  020040
 4768  024640  051124  041501  020113
 4769  024646  020040  042523  052103
 4770  024654  051117  020040  040504
 4771  024662  040524  020040  020040
 4772  024670  040504  040524     000
 4773
 4774
 4775  024676  001276                  DT1:    .WORD   ATTN
 4776  024700  001274  012230  012232  DT2:    .WORD   DDRIVE,RPERRS,RPERRS+2,RPERRS+4,RPERRS+6,ATTN
 4777  024706  012234  012236  001276
 4778  024714  001274  017370  017372  DT3:    .WORD   DDRIVE,RD.ADR,RD.WRD
 4779  024722  001274  017614  017612  DT4:    .WORD   DDRIVE,WRT.AD,WRT.WD,RD.WRD
 4780  024730  017372
 4781  024732  001172                  DT6:    .WORD   $RPADR
 4782  024734  001214  001116  001306  DT10:   .WORD   DRIVE,$ERRPC,RP.REG,RP.REG+10,RP.REG+12,RP.REG+14,RP.REG+40,RP.REG+42
 4783  024742  001316  001320  001322
 4784  024750  001346  001350
 4785  024754  001352  001354  001310          .WORD   RP.REG+44,RP.REG+46,RP.REG+2,RP.REG+4,RP.REG+6,RP.REG+16,RP.REG+20
 4786  024762  001312  001314  001324
 4787  024770  001326
 4788  024772  001330  001332  001334          .WORD   RP.REG+22,RP.REG+24,RP.REG+26,RP.REG+30,RP.REG+32,RP.REG+34,RP.REG+36
 4789  025000  001336  001340  001342
 4790  025006  001344
 4791  025010  001214  001116  001270  DT17:   .WORD   DRIVE,$ERRPC,DS.CYL,DS.TRK,SAVSEC
 4792  025016  001272  001252
 4793  025022  001214  001116  001270  DT20:   .WORD   DRIVE,$ERRPC,DS.CYL,DS.TRK,SAVSEC
 4794  025030  001272  001252
 4795  025034  001306  001316  001320          .WORD   RP.REG,RP.REG+10,RP.REG+12,RP.REG+14,RP.REG+40,RP.REG+42,RP.REG+44,RP.RE
 4796  025042  001322  001346  001350
 4797  025050  001352  001354
 4798  025054  001310  001312  001314          .WORD   RP.REG+2,RP.REG+4,RP.REG+6,RP.REG+16,RP.REG+20,RP.REG+22,RP.REG+24,RP.RE
 4799  025062  001324  001326  001330
 4800  025070  001332  001334
 4801  025074  001336  001340  001342          .WORD   RP.REG+30,RP.REG+32,RP.REG+34,RP.REG+36
 4802  025102  001344
 4803  025104  001214  001116  025324  DT23:   .WORD   DRIVE,$ERRPC,BUFP,RBUF,RBUF+2,RBUF+4,RBUF+6
 4804  025112  025314  025316  025320
 4805  025120  025322
 4806  025122  001214  001116  001270  DT24:   .WORD   DRIVE,$ERRPC,DS.CYL,DS.TRK,SAVSEC,$GDDAT,RP.REG+22
 4807  025130  001272  001252  001124
 4808  025136  001330
 4809  025140  001306  001316  001320          .WORD   RP.REG,RP.REG+10,RP.REG+12,RP.REG+14,RP.REG+40,RP.REG+42,RP.REG+44,RP.RE
 4810  025146  001322  001346  001350
 4811  025154  001352  001354
 4812  025160  001310  001312  001314          .WORD   RP.REG+2,RP.REG+4,RP.REG+6,RP.REG+16,RP.REG+20,RP.REG+22,RP.REG+24,RP.RE
 4813  025166  001324  001326  001330
 4814  025174  001332  001334
 4815  025200  001336  001340  001342          .WORD   RP.REG+30,RP.REG+32,RP.REG+34,RP.REG+36
 4816  025206  001344
 4817
 4818  025210  000001                  DF1:    .WORD   1
```

```
 4819  025212     001      000                    .BYTE   1,0
 4820  025214  000001                   DF2:     .WORD   1
 4821  025216     006      000                    .BYTE   6,0
 4822  025220  000001                   DF3:     .WORD   1
 4823  025222     003      000                    .BYTE   3,0
 4824  025224  000001                   DF4:     .WORD   1
 4825  025226     004      000                    .BYTE   4,0
 4826  025230  000003                   DF10:    .WORD   3
 4827  025232     010      000                    .BYTE   8.,0
 4828  025234  023704                             .WORD   DH10A
 4829  025236     007      000                    .BYTE   7,0
 4830  025240  023771                             .WORD   DH10B
 4831  025242     007      000                    .BYTE   7,0
 4832  025244  000001                   DF17:    .WORD   1
 4833  025246     005      034                    .BYTE   5,34
 4834  025250  000004                   DF20:    .WORD   4
 4835  025252     005      034                    .BYTE   5,34
 4836  025254  024174                             .WORD   DH20A
 4837  025256     010      000                    .BYTE   8.,0
 4838  025260  024272                             .WORD   DH20B
 4839  025262     010      000                    .BYTE   8.,0
 4840  025264  024367                             .WORD   DH20C
 4841  025266     004      000                    .BYTE   4,0
 4842  025270  000001                   DF23:    .WORD   1
 4843  025272     007      000                    .BYTE   7,0
 4844  025274  000004                   DF24:    .WORD   4
 4845  025276     007      034                    .BYTE   7,34
 4846  025300  024174                             .WORD   DH20A
 4847  025302     010      000                    .BYTE   8.,0
 4848  025304  024272                             .WORD   DH20B
 4849  025306     010      000                    .BYTE   8.,0
 4850  025310  024367                             .WORD   DH20C
 4851  025312     004      000                    .BYTE   4,0
 4852
 4853                                ;;*****************************************************************
 4854
 4855                                ;BUFFER STARTS HERE
 4856
 4857                                ;;*****************************************************************
 4858
 4859  025314  000000  000000  000000  RBUF:    .WORD   0,0,0,0         ;BUFFER FOR HEADER CHECK
 4860  025322  000000
 4861
 4862  025324                          BUFP:                   ;FORMAT AND CHECK BUFFER STARTS HERE
 4863
 4864  025324  005015  055012  026532  TITLE:   .ASCII  <CR><LF><LF>/ZZ-CZRJB-C/<CR><LF>
 4865  025332  055103  045122  026502
 4866  025340  006503     012
 4867  025343     122  030120  027464           .ASCIZ  @RP04/5/6 FORMATTER PROGRAM@<CR><LF><LF>
 4868  025350  027465  020066  047506
 4869  025356  046522  052101  042524
 4870  025364  020122  051120  043517
 4871  025372  040522  006515  005012
 4872  025400     000
 4873  025401     015  052012  020117  LOADRV: .ASCII  <CR><LF>/TO 'FORMAT' OR 'CHECK' DRIVE 0, REPLACE THE 'XXDP'/<CR><LF>
 4874  025406  043047  051117  040515
```

```
 4875  025414  023524  047440  020122
 4876  025422  041447  042510  045503
 4877  025430  020047  051104  053111
 4878  025436  020105  026060  051040
 4879  025444  050105  040514  042503
 4880  025452  052040  042510  023440
 4881  025460  054130  050104  006447
 4882  025466     012
 4883  025467     120  041501  020113          .ASCII  /PACK ON DRIVE 0 WITH ANOTHER PACK, CLEAR MEMORY LOCATION 40,/<CR><LF>
 4884  025474  047117  042040  044522
 4885  025502  042526  030040  053440
 4886  025510  052111  020110  047101
 4887  025516  052117  042510  020122
 4888  025524  040520  045503  020054
 4889  025532  046103  040505  020122
 4890  025540  042515  047515  054522
 4891  025546  046040  041517  052101
 4892  025554  047511  020116  030064
 4893  025562  006454     012
 4894  025565     101  042116  051040          .ASCIZ  /AND RESTART THE PROGRAM/<CR><LF>
 4895  025572  051505  040524  052122
 4896  025600  052040  042510  050040
 4897  025606  047522  051107  046501
 4898  025614  005015     000
 4899
 4900
 4901                                  .SBTTL  BUSADR - GET BUS ADDRESS AND VECTOR ADDRESS FOR RH11
 4902                                  ;THIS ROUTINE IS USED TO INSURE THE BUS ADDRESS
 4903                                  ;OF THE RH11 IS SETUP FOR THE PROPER ADDRESS.
 4904                                  ;IT WILL ALSO READ THE ADDRESS FROM THE TTY IF
 4905                                  ;REQUIRED.
 4906                                  ;NOTE: THIS ROUTINE DESTROYS R0-R4
 4907                                  ;CALL
 4908                                  ;
 4909                                  ;       JSR     PC,BUSADR
 4910                                  ;       RETURN
 4911
 4912  025620  177700                  HIAD:   .WORD 177700
 4913  025622  000776                  HIVEC:  .WORD 776
 4914  025624  000000                  LIMIT:  .WORD
 4915  025626  005737  001302          BUSADR: TST     CHGADR          ;INPUT FROM TTY REQUESTED?
 4916  025632  001460                          BEQ     7$              ;NO--BRANCH
 4917  025634  005037  001302                  CLR     CHGADR          ;YES--CLEAR THE REQUEST FLAG
 4918  025640  012700  001172          1$:     MOV     #$RPADR,R0      ;FIRST ADDRESS
 4919  025644  104401  026046                  TYPE    ,MRPCS1         ;'RPCS1='
 4920  025650  012046                          MOV     (R0)+,-(SP)     ;PRESENT RPCS1 ADDRESS
 4921  025652  104402                          TYPOC                   ;TYPE IT
 4922  025654  104401  021035                  TYPE    ,LINSP          ;2 SPACES
 4923  025660  104411                          RDLIN                   ;GET THE ENTRY
 4924  025662  012601                          MOV     (SP)+,R1        ;ADDRESS OF ASCII TEXT
 4925  025664  013737  025620  025624          MOV     HIAD,LIMIT      ;THIS IS THE ADDRSS HIGH LIMIT
 4926  025672  004537  026070                  JSR     R5,CK.NUM       ;CHECK THE NUMBER
 4927  025676  025716                          3$                      ;CARRIAGE RETURN ONLY ENTERED
 4928  025700  025774                          7$                      ;PERIOD ONLY ENTERED
 4929  025702  025640                          1$                      ;ILLEGAL INPUT
 4930  025704  025712                          2$                      ;TERMINATED WITH A CARRIAGE RETURN
```

```
 4931  025706  025640                        1$                      ;TERMINATED WITH A ''.''
 4932  025710  025770                        4$                      ;TERMINATED WITH A ''.''
 4933  025712  010260  177776          2$:   MOV     R2,-2(R0)         ;SAVE NEW RPCS1
 4934  025716  104401  026057          3$:   TYPE    ,MRHVEC           ;''RHVEC=''
 4935  025722  012700  001174                MOV     #$RPVEC,R0        ;FIRST VECTOR
 4936  025726  012046                        MOV     (R0)+,-(SP)       ;PRESENT RH11 VECTOR ADDRESS ON THE STACK
 4937  025730  104402                        TYPOC                     ;TYPE IT
 4938  025732  104401  021035                TYPE    ,LINSP            ;2 SPACES
 4939  025736  104411                        RDLIN                     ;READ THE ENTRY
 4940  025740  012601                        MOV     (SP)+,R1          ;ASCII TEXT ADDRESS
 4941  025742  013737  025622  025624        MOV     HIVEC,LIMIT       ;VECTOR LIMIT
 4942  025750  004537  026070                JSR     R5,CK.NUM         ;CHECK THE NUMBER
 4943  025754  025774                        7$                        ;CARRIAGE RETURN ONLY ENTERED
 4944  025756  025774                        7$                        ;PERIOD ONLY ENTERED
 4945  025760  025716                        3$                        ;ILLEGAL INPUT
 4946  025762  025770                        4$                        ;TERMINATED WITH A CARRIAGE RETURN
 4947  025764  025716                        3$                        ;TERMINATED WITH A ''.''
 4948  025766  025770                        4$                        ;TERMINATED WITH A ''.''
 4949  025770  010260  177776          4$:   MOV     R2,-2(R0)         ;SAVE INPUT
 4950  025774  013701  000004          7$:   MOV     ERRVEC,R1         ;SAVE THE ERROR VECTOR
 4951  026000  012737  026034  000004        MOV     #8$,ERRVEC        ;SETUP FOR TRAP
 4952  026006  005777  153160                TST     @$RPADR           ;CHECK FOR RH11
 4953  026012  010137  000004                MOV     R1,ERRVEC         ;RESTORE ERROR VECTOR
 4954  026016  012700  001172                MOV     #$RPADR,R0        ;FIRST ADDRESS OF NEW PARAMETERS
 4955  026022  012701  012376                MOV     #RPADR,R1         ;FIRST ADDRESS OF WHERE TO PUT THEM
 4956  026026  012021                        MOV     (R0)+,(R1)+       ;BUS ADDRESS
 4957  026030  012021                        MOV     (R0)+,(R1)+       ;VECTOR ADDRESS
 4958  026032  000207                        RTS     PC                ;RETURN
 4959  026034  010137  000004          8$:   MOV     R1,ERRVEC         ;RESTORE ERROR VECTOR
 4960  026040  022626                        CMP     (SP)+,(SP)+       ;CLEAN OFF THE STACK
 4961  026042  104006                        ERROR   6                 ;REPORT THE ERROR
 4962  026044  000675                        BR      1$                ;ASK FOR BUS ADDRESS
 4963
 4964  026046  050122  051503  020061  MRPCS1: .ASCIZ  @RPCS1 = @
 4965  026054  020075     000
 4966  026057     122  053110  041505  MRHVEC: .ASCIZ  @RHVEC = @
 4967  026064  036440  000040
 4968
 4969                                  .SBTTL  CK.NUM - CHECK NUMBER (OCTAL)
 4970                                  ;THIS ROUTINE CHECKS AN ASCIZ STRING FOR LEGAL CHARACTERS
 4971                                  ;AND FORMS AN OCTAL NUMBER IN R2
 4972                                  ;CALL:
 4973                                  ;       MOV     #ADR,R1           ;ADDRESS OF ASCIZ STRING
 4974                                  ;       MOV     #NUM,R2           ;MAX SIZE OF INPUT NUMBER
 4975                                  ;       JSR     R5,CK.NUM         ;GO FORM THE NUMBER
 4976                                  ;       RETURN  ADR1              ;''CR'' ONLY ENTERED -- R2 = 0
 4977                                  ;       RETURN  ADR2              ;''PERIOD'' ONLY ENTERED -- R2 = 0
 4978                                  ;       RETURN  ADR3              ;ILLEGAL CHARACTER IN THE INPUT STRING
 4979                                  ;       RETURN  ADR4              ;''CR'' ENTERED -- R2 = NUMBER
 4980                                  ;       RETURN  ADR5              ;''COMMA'' -- R2 = NUMBER
 4981                                  ;       RETURN  ADR6              ;''PERIOD'' -- R2 = NUMBER
 4982
 4983  026070  010446                  CK.NUM: MOV     R4,-(SP)          ;SAVE R4
 4984  026072  010346                          MOV     R3,-(SP)          ;SAVE R3
 4985  026074  010246                          MOV     R2,-(SP)          ;SAVE R2
 4986  026076  005004                          CLR     R4                ;RETURN POINTER
```

```
 4987  026100  005003                        CLR     R3              ;START NUMBER AT ZERO
 4988  026102  005002                        CLR     R2              ;STORE RESULT
 4989  026104  004537  006556                JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
 4990  026110  026210                        6$                      ;ILLEGAL CHARACTER
 4991  026112  026214                        8$                      ;CARRIAGE RETURN
 4992  026114  026210                        6$                      ;''
 4993  026116  026212                        7$                      ;'.'
 4994  026120  026124                        1$                      ;DIGIT 0-7
 4995  026122  026210                        6$                      ;DIGIT 8-9
 4996  026124  062705  000004        1$:     ADD     #4,R5           ;INCREMENT RETURN PAST ''CR'' AND 'PERIOD'' ONLY RETURNS
 4997  026130  006303                2$:     ASL     R3              ;FOR THE OCTAL NUMBER IN R3
 4998  026132  103426                        BCS     6$              ;DON'T LET IT GET TO BIG
 4999  026134  006303                        ASL     R3
 5000  026136  103424                        BCS     6$
 5001  026140  006303                        ASL     R3
 5002  026142  103422                        BCS     6$
 5003  026144  060203                        ADD     R2,R3
 5004  026146  004537  006556                JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
 5005  026152  026214                        8$                      ;ILLEGAL CHARACTER
 5006  026154  026176                        5$                      ;CARRIAGE RETURN
 5007  026156  026174                        4$                      ;''
 5008  026160  026166                        3$                      ;'.'
 5009  026162  026130                        2$                      ;DIGIT 0-7
 5010  026164  026214                        8$                      ;DIGIT 8-9
 5011  026166  105711                3$:     TSTB    (R1)            ;DOES A ''CR'' FOLLOW THE 'PERIOD''
 5012  026170  001011                        BNE     8$              ;BR IF NOT
 5013  026172  005724                        TST     (R4)+           ;INCREMENT THE RETURN
 5014  026174  005724                4$:     TST     (R4)+           ;INCREMENT THE RETURN INDEX
 5015  026176  005724                5$:     TST     (R4)+           ;INCREMENT THE RETURN INDEX
 5016  026200  023703  025624                CMP     LIMIT,R3        ;INPUT VALUE TOO LARGE ?
 5017  026204  101003                        BHI     8$              ;BR IF IT IS
 5018  026206  000401                        BR      7$              ;BR IF NOT
 5019  026210  005725                6$:     TST     (R5)+           ;INCREMENT THE RETURN ADDRESS
 5020  026212  005725                7$:     TST     (R5)+           ;INCREMENT THE RETURN ADDRESS
 5021  026214  060405                8$:     ADD     R4,R5           ;SETUP FOR PROPER RETURN
 5022  026216  010302                        MOV     R3,R2           ;LOAD ENTERED VALUE
 5023  026220  005726                        TST     (SP)+           ;CLEAN OFF THE STACK
 5024  026222  012603                        MOV     (SP)+,R3        ;RESTORE R3
 5025  026224  012604                        MOV     (SP)+,R4        ;RESTORE R4
 5026  026226  011505                        MOV     (R5),R5         ;GET RETURN ADDRESS
 5027  026230  000205                        RTS     R5              ;RETURN
 5028
 5029
 5030
 5031          000001                .END
```

```
ABS    = 000200            816#
ACK    = 000123            886#
ACL    = 000040            850#    859#
ACTDRV  012314            3068#   3347*   3398*   3711*   3720*   3950
ACTSTR  012315            3074#   3952*   3967*
ACU    = 100000            805#
ADDRIS  022215            1987    4523#
ADRTBL  001500            1049#   1687
AOE    = 001000            722#
ATA    = 100000            709#
ATABIT  012364            3140#   3313    3410    3515    3874    3893    3901    3918    4026
ATTN    001276             979#   2185*   4775    4776
AT0    = 000001            742#
AT1    = 000002            743#
AT2    = 000004            744#
AT3    = 000010            745#
AT4    = 000020            746#
AT5    = 000040            747#
AT6    = 000100            748#
AT7    = 000200            749#
A16    = 000400            640#
A17    = 001000            641#
BADENT  021566            2029    4470#
BAI    = 000010            658#
BEGCYL  001224             955#   1030    1429*   1484*   1493    1540    1547    1730    1734    1867    1906
BEGIN   002100             517    1267#
BEGIN1  002110             514    1269#   1305
BEGIN2  002114            1268    1270#
BEGTRK  001230             957#   1031    1428*   1483*   1496    1543    1549    1866    1869    1907
BIT0   = 000001            614#
BIT00  = 000001            604#    614    1367    1480
BIT01  = 000002            603#    613    1370    1480    1852    3390    3637
BIT02  = 000004            602#    612
BIT03  = 000010            601#    611    3833    4148
BIT04  = 000020            600#    610    1833    3868
BIT05  = 000040            599#    609    3230    3657    3848    4000
BIT06  = 000100            598#    608    3322    3381    3455    3746    4209
BIT07  = 000200            597#    607    3322    3589    3732    3833    3848    3868    3897    4181
BIT08  = 000400            596#    606    1830    3322    4013
BIT09  = 001000            595#    605    1830    2212    3998
BIT1   = 000002            613#   1716    1821
BIT10  = 002000            594#   1827    1852    2196    3639
BIT11  = 004000            593#   1827    3277    3414    3432    3608    3919
BIT12  = 010000            592#   1824    1852    3272    3304    3322    3392    3426    3604    3623    3635    3648    3858
                          3860    4086    4143    4210
BIT13  = 020000            591#   1821    1852    2203    3375    3806    4089
BIT14  = 040000            590#   1821    1852    3387    3423    3793    3937    4053    4098    4102    4207
BIT15  = 100000            589#   3375    3387    3390    3392    3423    3426    3608    3637    3639    3746    3833    3848
                          3858    3868    3937    3998    4013    4053    4060
BIT2   = 000004            612#   1821    4060
BIT3   = 000010            611#
BIT4   = 000020            610#
BIT5   = 000040            609#
BIT6   = 000100            608#   3416
BIT7   = 000200            607#   1670    1677
BIT8   = 000400            606#
```

```
BIT9  = 001000   605#
BPTVEC= 000014   621#
BUFP    025324  1049  1050  1051  1052  1053  1054  1055  1056  1057  1058  1059  1060  1061
                1062  1063  1064  1065  1066  1067  1068  1069  1070  1572  1600  1757  1864
                1905  1925  1936  4803  4862#
BUSADR  025626  1325  4915#
C       021025  1988  4404#
CHGADR  001302   982#  1267* 1269* 4915  4917*
CI1     013736  3445  3473#
CI3     014044  3447  3496#
CI4     014152  3436  3517#
CI5     014530  3495  3516  3594#  3603
CI6     014552  3533  3543  3545  3547  3599#
CI7     014566  3377  3454  3485  3489  3493  3501  3510  3514  3525  3532  3538  3542  3556
                3562  3566  3576  3587  3602  3604#  3706  3736  3885  4193
CI7B    014614  3607  3610#  3887
CI8     014666  3605  3613  3622#  3759  4045
CKADRS  003606  1540#
CKSWR = 104407  2183  2211  2954#
CKTRK   004260  1602  1630#
CK.CHR  006556  1443  1506  2089#  2126  2140  4989  5004
CK.DEC  006530  2067#  2095
CK.DIG  006630  2020  2120#
CK.NUM  026070  4926  4942  4983#
CK.OCT  006502  2049#  2097
CLOCK   006234  1950  1960  1971#
CLR   = 000040   660#
CLRQUE  020400  3212  3656  4023  4277#
CNTLC   001300   981#  1305* 1437* 1524* 1986* 1999* 2549  2632
CR    = 000015   529#   942#  2350  2360  4398  4401  4409  4413  4416  4420  4424  4429  4434
                4447  4454  4462  4470  4473  4481  4485  4489  4492  4500  4505  4510  4514
                4744  4755  4864  4867  4873  4883  4894
CRLF  = 000200   530#  2321  2360
CSF   = 000002   792#   810#
CSU   = 000010   794#   812#
CYLCK   001246   964#  1730* 1761  1764*
DCK   = 100000   728#  1654
DCL   = 000100   851#   860#
DCU   = 000001   857#
DDISP = 177570   536#   927  1291
DDRIVE  001274   978#  2184* 4776  4778  4779
DE1   = 000040   699#
DFF20 = 000002   695#
DF1     025210  1119  1154  4818#
DF10    025230  1168  1175  1182  1189  1196  1203  1210  1231  1238  4826#
DF17    025244  1217  4832#
DF2     025214  1126  1147  4820#
DF20    025250  1224  1259  4834#
DF23    025270  1245  4842#
DF24    025274  1252  4844#
DF3     025220  1133  4822#
DF4     025224  1140  4824#
DH1     023424  1117  4653#
DH10    023606  1166  1173  1180  1187  1194  1201  1208  1229  1236  4674#
DH10A   023704  4685#  4828
DH10B   023771  4694#  4830
```

```
DH17    024056          1215    4703#
DH2     023431          1124    1145    4654#
DH20    024125          1222    1257    4710#
DH20A   024174          4717#   4836    4846
DH20B   024272          4728#   4838    4848
DH20C   024367          4739#   4840    4850
DH23    024424          1243    4744#
DH24    024522          1250    4755#
DH3     023506          1131    4662#
DH4     023534          1138    4666#
DH6     023573          1152    4672#
DIGB  = 000004           696#
DISPLA  001142           927#   1291*   1299*   2195*
DISPRE  000174           498#   1299
DLT   = 100000           670#
DL64  = 000020           698#
DMD   = 000001           732#
DONE    005340          1803    1804#
DPINT   012270          3028#   3241    3244    3330*   3357    3777    3802    3927    3929*   3990    4028    4039    4047*
DPR   = 000400           702#
DPRQS   012300          3041#   3363    3449*   3780    3941*   3992    4030    4041    4056*
DRIVE   001214           951#   1450*   1455*   1476    1478    1531    1861    4782    4791    4793    4803    4806
DRQ   = 004000           765#
DRVACT  012240          2985#   3371    3550*   3597*   3616*   3627    3642*   3725*   3798    3831    3862*   3869*   3882
                        3892*   4001*   4007    4014*
DRVCLR= 000111           881#
DRVINT  012626          3232    3266#   3359    3418    3909    3930    4005
DRVQUE  020476          3366    3379    4317#
DRVSTA  012250          1350    1465    2999#   3222*   3223*   3224*   3225*   3235*   3267*   3276*   3319*   3324*   3353
                        3361    3385    3420    3424    3551*   3660*   3783    3791    3804    3856*   3863*   3875    3932
                        4048*
DRVTYP  012260          1353    1367    1370    1470    1480    3015#   3268*   3283*   3288*   3293*   3298*   3388    3661*
DRY   = 000200           701#
DSWR  = 177570           535#    926    1290
DS.CYL  001270           976#   2186*   4791    4793    4806
DS.TRK  001272           977#   2187*   4791    4793    4806
DTE   = 010000           725#   1654
DTSY  = 000200           738#
DTUW    012362          3134#   3218    3438    3494*   3617    3620*   3630    3643    3645*   3654*   3714    3726*   3995
                        4010    4021*   4032    4096
DT00  = 000001           756#
DT01  = 000002           757#
DT02  = 000004           758#
DT03  = 000010           759#
DT04  = 000020           760#
DT05  = 000040           761#
DT06  = 000100           762#
DT07  = 000200           763#
DT08  = 000400           764#
DT1     024676          1118    4775#
DT10    024734          1167    1174    1181    1188    1195    1202    1209    1230    1237    4782#
DT17    025010          1216    4791#
DT2     024700          1125    1146    4776#
DT20    025022          1223    1258    4793#
DT23    025104          1244    4803#
DT24    025122          1251    4806#
```

```
DT3      024714            1132    4778#
DT4      024722            1139    4779#
DT6      024732            1153    4781#
DVA    = 004000             690#
ECH    = 000100             719#
ECI    = 004000             833#
EFLG     001304             983#  1271*  1597*  1662   1665*  1672*  1679*  1682*  1706*  1715*
EMPTYQ   020456            3614    3659   3747   3870   3939   4061   4302#
EMTVEC=  000030             624#  1280*  1281*  1304*
EM1      022242            1116    4534#
EM10     022532            1165    4568#
EM11     022550            1172    4571#
EM12     022606            1179    4576#
EM13     022651            1186    4582#
EM14     022672            1193    4585#
EM15     022715            1200    4589#
EM16     022761            1207    4596#
EM17     023033            1214    4604#
EM2      022303            1123    4540#
EM20     023111            1221    4612#
EM21     023147            1228    4618#
EM22     023220            1235    4625#
EM23     023267            1242    4632#
EM24     023337            1249    4639#
EM25     023361            1256    4643#
EM3      022341            1130    4546#
EM4      022377            1137    4552#
EM5      022434            1144    4557#
EM6      022470            1151    4562#
ENDCYL   001222             954#  1032   1479*  1482*  1485   1486   1493   1540   1546   1761
ENDTRK   001226             956#  1033   1430*  1496   1542   1548
ENTADR   021040            1491    4409#
ERINDX   005364            1613    1645   1745   1818#
ERR    = 040000             708#
ERRVEC=  000004             617#  1288   1289*  1300*  1946*  1947*  1957*  1966*  4950   4951*  4953*  4959*
ES.SAV   020636            3762    3813   3817   4094   4154   4371#
EXT1   = 000001             772#
EXT10  = 000010             775#
EXT2   = 000002             773#
EXT20  = 000020             776#
EXT4   = 000004             774#
EXT40  = 000040             777#
FEN    = 000200             798#
FER    = 000020             717#  1654
FMTDPB   001356            1011#  1432*  1476   1478*  1585*  1588*  1589*  1591   1593   1598*  1599*  1600*  1603*
                           1607   1609   1630*  1634   1636   1685*  1687*  1693*  1702   1731*  1732*  1733*  1734*
                           1735*  1739   1741   1751   1766*  1820   1823   1826   1829   1832   1851   1861*  1862*
                           1863*  1864*  1865*  1866*  1867*  1868*  1881*  1885*  1886*  1894*  1895*  1896*  1989
                           1995   2186   2187
FMT22  = 010000             834#
F1     = 000002             685#
F2     = 000004             686#
F3     = 000010             687#
F4     = 000020             688#
F5     = 000040             689#
GETREG=  000141             888#
```

```
GETREQ  020552          3411    3632    3787    3823    3884    3936    3997    4012    4050    4057    4338#
GNS   = ****** U         497    2946    2947    2948    2949    2950    2952    2954    2955    2956    2957    2958
GO    = 000001           684#
GRV   = 000010           697#
GTSWR = 104406          1319    2952#
HCE   = 000200           720#    1654    1751
HCI   = 002000           832#
HCRC  = 000400           721#    1654
HDPEAD  005000          1720    1727#
HEDERR  001262           970#    1727*
HIAD    025620          4912#    4925
HIVEC   025622          4913#    4941
HT    = 000011           527#    2319    2360
IAE   = 002000           723#
IE    = 000100           638#
ILF   = 000001           713#
ILR   = 000002           714#
IOTVEC= 000020           622#
IR    = 000100           661#
ISR     015260          3227    3711#
IXE   = 004000           802#    820#
LA      015122          3442    3676#
LACNT   012326          3093#   3684*   3685    3702*
LF    = 000012           528#    943#    2354    2360    4398    4401    4409    4413    4416    4420    4424    4429    4434
                        4447    4454    4462    4470    4473    4481    4485    4489    4492    4500    4505    4510    4514
                        4744    4755    4864    4867    4873    4883    4894
LIMIT   025624          4914#   4925*   4941*   5016
LINSP   021035          1365    1992    2265    2274    2285    4408#   4922    4938
LIN4SP  021033          1349    4407#
LOADRV  025401          1311    4873#
LOP.CK  005464          1620    1708    1756    1760    1847#
LST   = 002000           704#
MADRER  021606          1498    4473#
MAXSEC  001266           974#    1421*   1427*   1641    1702    1904    1924    1935
MCCMPT  022141          1781    4514#
MCLK  = 000002           733#
MCPE  = 020000           643#
MCPEMX  012374          3152#   4076    4124
MDRNP   021112          1472    4416#
MDRVD   021031          1451    4406#
MER11   021137          1474    4420#
MFCMPT  022114          1779    4510#
MFORMT  021303          1399    4439#
MHECK   021324          1397    4442#
MHS   = 001000           800#    818#
MINX  = 000004           734#
MMODE   021251          1386    4434#
MNDLTA  012410          3168#   3700
MNRP04  021166          4424#
MODE    001220           953#   1385*   1402*   1525    1601    1657    1691    1777
MOFFLN  021070          4413#
MOH   = 020000           766#
MOL   = 010000           706#
MORMAT  021337          1401    4444#
MPATD   022010          1513    4498#
MPE   = 000400           663#
```

```
MRD    = 000020            736#
MRHVEC  026057            4934    4966#
MRPCS1  026046            4919    4964#
MSCHK   022060            1529    4505#
MSE    = 000020            795#
MSEC20  021507            1423    4462#
MSEC22  021430            1417    4454#
MSELD   021237            4432#
MSELP   021710            1503    4485#
MSFOU   022023            1527    4500#
MSIZE   021357            1406    4447#
MSTCK  = 000010            735#
MUNIT   021007            1440    4401#
MUSDR   021217            1468    4429#
MWC     001260             969#   1419*   1425*   1599    1688    1863
MWR    = 000040            737#
MXDLTA  012406            3165#   3698
MXF    = 001000            664#
MXLACT  012404            3162#   3685
MXWNDW  012412            3171#   3503
M0      003134            1433    1437#   1454    1469    1473    1475    1806
M1      002662            1385#   1396
M1A     002752            1398    1406#   1416
M1B     003100            1422    1428#   1981
M2      003422            1477    1487    1491#   1499
M4      003462            1494    1497    1503#   1519
M5      003544            1524#   1808
NBA    = 100000            768#
NED    = 010000            667#
NEM    = 004000            666#
NHS    = 002000            801#    819#
NOTPRS  020724            1358    4390#
NOTRP   020703            1356    4387#
NOTSAF  020741            1362    4393#
NUMERR  022165            1782    4518#
OCYL   = 100000            853#    863#
OENTER  006246            1437    1977#   1986
OFFSET= 000115             883#
OFREV  = 000200            831#
OF100  = 000004            827#
OF200  = 000010            828#
OF25   = 000001            825#
OF400  = 000020            829#
OF50   = 000002            826#
OF800  = 000040            830#
OPE    = 020000            861#
OPI    = 020000            726#   1654
OPT     013456            3373    3408#   3742    3902    3942
OR     = 000200            662#
PAR    = 000010            716#
PARENT  006366            1492    2007#
PAR1    001430            1030    1038#
PAR2    001443            1031    1040#
PAR3    001456            1032    1042#
PAR4    001467            1033    1044#
PAT    = 000020            659#
```

```
PATA    001242            962#   1561*  1565*  1569*  1573
PATB    001244            963#   1562*  1566*  1570*  1574
PATSEL  001240            961#   1431*  1520*  1563   1567
PGE   = 002000            665#
PGM   = 001000            703#
PIP   = 020000            707#
PIRQ  = 177772            534#
PIRQVE= 000240            628#
PLU   = 020000            804#    821#
POPQUE  020574           3422    3473   3588   3896   4353#
PR0   = 000000            551#
PR1   = 000040            552#
PR2   = 000100            553#
PR3   = 000140            554#
PR4   = 000200            555#
PR5   = 000240            556#   1303   1304
PR6   = 000300            557#   1952   1962
PR7   = 000340            558#   1339
PS    = 177776            531#    532   1339*  1464*  3210   3211*  3240*  3248*  3345   3346*  3399*  3409   3458*
                         3691*
PSEL  = 002000            642#
PSU   = 000001            847#
PSW   = 177776            532#   1985*
PWRVEC= 000024            623#
QCNT    020106           4219#   4278   4302*  4317   4319*  4339   4353*
QDRV0   020200           4230    4241   4250   4262#
QDRV1   020220           4231    4242   4251   4263#
QDRV2   020240           4232    4243   4252   4264#
QDRV3   020260           4233    4244   4253   4265#
QDRV4   020300           4234    4245   4254   4266#
QDRV5   020320           4235    4246   4255   4267#
QDRV6   020340           4236    4247   4256   4268#
QDRV7   020360           4237    4248   4257   4269#
QINPT   020116           4230#   4304   4321*  4322*  4323   4325*
QOUTPT  020136           4241#   4304*  4342   4355   4356*  4357   4359*
QSTART  020156           4250#   4284   4289   4325   4359
QSTOP   020160           4251#   4323   4357
QTERM = 020400           4258    4270#
RAW   = 000020            813#
RBUF    025314           1732    1757   4803   4859#
RDCHR = 104410           2708    2955#
RDDAT = 000171            895#
RDHD  = 000173            896#   1735
RDLIN = 104411           1387    1407   1441   1504   2018   2956#  4923   4939
RDY   = 000200            639#
RD.ADR  017370           4078*   4079*  4081#  4778
RD.RP   017344           3279    3308   3314   3536   3560   3574   3678   3692   3734   3757   3825   3835   3850
                         3985    4034   4076#  4131   4145   4151   4185
RD.RP1  017366           4080#   4110
RD.RP2  017512           4076*   4109#
RD.RP3  017516           4087    4101   4111#
RD.RP4  017522           4092    4113#
RD.WRD  017372           4082#   4083   4778   4779
READIN= 000121            885#
RECAL = 000107            880#
RELSE = 000113            882#
```

```
RESREG= 104413          2287    2845    2958#   3249    3393    3396    3459    3663    3719    3966    4194    4293
RESVEC= 000010           618#
RETRY   001250           965#   1596*   1621    1623*   1626*   1709    1711*   1714*
RMR   = 000004           715#
RNOP  = 000101           877#
RPADR   012376          1326*   3157#   3229    3352    3476    3496    3517    3676    3713    3983    4078    4084    4088
                        4104    4137    4141    4174    4204    4955
RPAS  = 000016          3182#   3313*   3755    3874*   3901*   3918*   4025
RPBA  = 000004          2188    2190    3177#
RPCA  = 000034          3189#   3488    3500    3531
RPCC  = 000036          3190#   3679
RPCS1 = 000000          3175#   3271*   3277    3302    3414    3430    3492    3513    3555    3575    3586    3601    3612
                        3735    3758    3810    3842    3879    4035    4132
RPCS2 = 000010          1651    1675    3179#   3230*   3270*   3272    3365*   3477*   3497*   3518*   3604    3623    3635
                        3648    3657*   3677*   3733*   3824*   3904*   3917*   3984*   4000*   4085    4142    4175*   4181
                        4205*   4210
RPDA  = 000006          1639    3178#   3484    3509    3524
RPDB  = 000022          3184#   4179
RPDS1 = 000012          3180#   3309    3416    3826    3851    3905    3986
RPDT  = 000026          3186#   3280
RPEC1 = 000044          3193#
RPEC2 = 000046          3194#   4189
RPERRS  012230          2977#   3213    3793    3796    3806    3905*   3906*   3907*   3908*   4776
RPER1 = 000014          1653    1668    1854    3181#   3315    3836    3906    4146
RPER2 = 000040          3191#   3907
RPER3 = 000042          3192#   3908
RPINIT  012414          1333    1461    3209#
RPLA  = 000020          3183#   3693
RPMR  = 000024          3185#
RPOF  = 000032          3188#   3306    3537    3541    3561    3565
RPSN  = 000030          3187#
RPTMR   016616          1972    3950#
RPVEC   012400          1327*   3158#   3226    3228    3346
RPWC  = 000002          3176#
RP.REG  001306           987#   1020    1639    1651    1653    1668    1675    1854    2188    2190    4782    4785    4788
                        4795    4798    4801    4806    4809    4812    4815
RP04    013164          1590    1606    1633    1738    3345#
RP04B   020751          1366    4395#
RP05    020756          1369    4396#
RP06    020763          1372    4397#
RTC   = 000117           884#
SAVEFG  012336          1334*   1462*   3108#   3590    3739    3898
SAVREG= 104412          2227    2819    2957#   3209    3348    3408    3622    3712    3953    4171    4277
SAVSEC  001252           966#   1639*   1641*   1642*   1643*   1684    1685    4791    4793    4806
SAVWC   001254           967#   1690*   1894
SC      015464          3718    3743    3745    3751    3755#
SCAWC   005704          1704    1894#
SC1   = 000100           778#
SC10  = 001000           781#
SC11    016316          3795    3832    3889#
SC12    016406          3789    3800    3904#
SC13    016456          3779    3782    3914#
SC2   = 000200           779#
SC20  = 002000           782#
SC3     015534          3770#   3774
SC4     015540          3772#   3803    3815    3819    3867    3881    3888    3903    3943
```

```
SC5     015552            3771    3777#
SC6     015752            3799    3820#
SC6A    016062            3797    3844#
SC7     016210            3834    3847    3849    3859    3869#
SC8     016266            3811    3827    3837    3843    3852    3880    3882#   3913
SEARCH= 000131             887#
SEC20   001264             971#   1420*   1426*   1586    1909
SEEK  = 000105             879#
SEEKFG  012340            1463*   3116#   3214    3440
SELDRV= 000145             890#
SETFMT= 000143             889#   1589
SETHDR  005732            1584    1729    1904#
SETPAT  003716            1545    1555    1561#
SETTBL  005532            1583    1728    1861#
SETVEC  002426            1324    1332#
SET.IE  020034            3243    3274    3383    3457    3662    3764    4203#
SKI   = 040000             852#    862#
SLASH   021021            2017    4403#
SOFSW   001216             952#   1595*   1659    1694*   1696    1698    1700*   1705*   1897*
SRCHWT  012312            3062#   3410*   3515*   3893
STACK = 001100             522#   1278    1605    1632    1737    1807    1977
STCLK1  006150            1946    1956#
STCLK2  006220            1957    1965#
STCLK3  006224            1955    1964    1966#
STKLMT= 177774             533#
STO     016710            3960    3981#
STO1    016740            3990#   4033    4038
STO2    017136            3989    3991    3993    4025#
STO3    017206            4027    4039#
STO5    017232            3987    4006    4036    4045#
STO6    017240            4029    4047#
STO7    017276            4031    4055#
STO8    017326            4054    4061#
STO9    017336            4024    4042    4044    4046    4052    4059    4063#
ST.CLK  006072            1332    1460    1946#
SVRH11  017716            3592    3609    3640    3741    3748    3788    3839    3857    3900    3938    3999    4062    4171#
SWR     001140             926#   1276    1290*   1292    1298*   1317    1337    1670    1677    1716    1804    1847    2196
                          2203    2208    2212    2552    2591    2646*
SWREG   000176             499#   1298    1317    2552    2591    2614
SW0   = 000001             586#   1804
SW00  = 000001             576#    586
SW01  = 000002             575#    585
SW02  = 000004             574#    584    1337
SW03  = 000010             573#    583
SW04  = 000020             572#    582
SW05  = 000040             571#    581
SW06  = 000100             570#    580
SW07  = 000200             569#    579
SW08  = 000400             568#    578
SW09  = 001000             567#    577    1847
SW1   = 000002             585#
SW10  = 002000             566#
SW11  = 004000             565#
SW12  = 010000             564#
SW13  = 020000             563#
SW14  = 040000             562#
```

```
SW15  = 100000          561#
SW2   = 000004          584#
SW3   = 000010          583#
SW4   = 000020          582#
SW5   = 000040          581#
SW6   = 000100          580#
SW7   = 000200          579#
SW8   = 000400          578#
SW9   = 001000          577#
SYSTAT  020770         1342   4398#
T       021027         1993   4405#
TABLE   001376         1030#  1485*  1486*  2008   2034
TAP   = 040000          767#
TBITVE= 000014          619#
TD      015324         3716   3725#
TDF   = 000040          796#   814#
TIMER   012342         3121#  3328*  3452*  3595*  3641*  3728*  3821*  3865*  3915*  3923*  3956   3958*  4016*
                       4043*  4049*  4055*
TITLE   025324         1308   4864#
TKVEC = 000060          626#  2529*  2530*
TPVEC = 000064          627#
TRAPVE= 000034          625#  1282*  1283*  1303*
TRE   = 040000          644#
TRKCNT  001236          960#  1869*  1878   1880*  1884*
TRKTST  005622         1719   1876#
TRK1  = 004000          783#
TRK10 = 040000          786#
TRK2  = 010000          784#
TRK20 = 100000          787#
TRK4  = 020000          785#
TRNSWT  012310         1978   3054#  3474*  3619*  3629   3646*  3655*  3730   3731*  3994   4009   4022*
TRTVEC= 000014          620#
TTRKS   001232          958#  1542*  1543*  1544*  1548*  1549*  1551*  1553*  1556*  1870
TTRKSC  001234          959#  1870*  1876*
TUF   = 000100          797#   815#
TYPADR  006266         1524   1985#  1999
TYPDS = 104405         1784   2016   2262   2950#
TYPE  = 104401         1308   1311   1341   1342   1349   1356   1358   1360   1362   1364   1365   1373   1375
                       1379   1386   1395   1397   1399   1401   1406   1415   1417   1423   1440   1451   1453
                       1468   1472   1474   1491   1498   1503   1513   1518   1527   1529   1536   1779   1781
                       1782   1785   1987   1988   1992   1993   1998   2011   2017   2029   2198   2206   2237
                       2238   2242   2243   2252   2265   2269   2274   2276   2282   2285   2324   2421   2497
                       2546   2558   2612   2613   2616   2627   2637   2648   2667   2715   2721   2726   2730
                       2735   2736   2738   2741   2745   2781   2946#  4919   4922   4934   4938
TYPERR  007174         2205   2227#
TYPOC = 104402         2258   2615   2947#  4921   4937
TYPON = 104404         2949#
TYPOS = 104403         1346   1533   2948#
ULDFLG  012316         3081#  3269*  3355   3370*  3552*  3615*  3647*  3785   3871   3873*  3889   3891*  4002*
                       4015*
UNLOAD= 000103          878#
UNS   = 040000          727#
UNTOFF  020662         1360   4383#
UNTON   020673         1364   4385#
UPDACY  006012         1765   1887   1924#
UPDATK  006042         1882   1935#
```

```
UPE    = 020000            668#
US1    = 000001            655#
US2    = 000002            656#
US4    = 000004            657#
UWR    = 000010            849#
VUF    = 000002            848#
VU30   = 010000            803#
VV     = 000100            700#
WAO    = 000002            858#
WC       001256            968#  1418*  1424*  1571
WCE    = 040000            669#  1651   1675
WCF    = 000040            718#
WCKD   = 000151            891#
WCKHD  = 000153            892#  1630
WCTBL    001554           1074#  1689
WCU    = 000001            791#   809#
WLE    = 004000            724#  1854
WRL    = 004000            705#
WRTDAT= 000161             893#
WRTHD  = 000163            894#  1603
WRTRK    004022           1583#
WRTRK1   004076           1595#  1718   1721
WRTRK2   004140           1603#  1695
WRT.AD   017614           4127*  4137*  4140#  4779
WRT.RP   017524           3301   3305   3429   3483   3487   3491   3499   3508   3512   3523   3530   3540   3554
                          3564   3585   3600   3611   3809   3841   3878   4124#
WRT.R1   017610           4138#  4158
WRT.R2   017700           4124*  4157#
WRT.R3   017706           4133   4144   4147   4153   4160#
WRT.R4   017712           4149   4162#
WRT.R5   017714           4161   4163#
WRT.WD   017612           4125*  4129   4136*  4139#  4779
WRU    = 000400            799#   817#
WSU    = 000004            793#   811#
$AUTOB   001134            923#  1321*  2605   2762
$BDADR   001122            918#
$BDDAT   001126            920#
$BELL    001162            937#  2198   2220   2558   2755
$CHARC   007640           2326*  2336*  2343   2352*  2357#
$CKSWR   010622           2591#  2954
$CMTAG   001100            906#  1273   1274   1284
$CM3   = 000000            936#
$CNTLC   011517           2546   2627   2755#
$CNTLG   011531           2612   2757#
$CNTLU   011524           2637   2730   2756#
$CRLF    001167            939#  1341   1375   1379   1536   1785   1998   2206   2220   2237   2242   2252   2269
                          2282   2325   2360   2648   2735   2755
$DBLK    010306           2463   2497   2505#
$DB2D    011654           2800   2819#
$DECVL   012034           2821   2867#
$DOAGN   005334           1790   1795   1801#
$DTBL    010276           2466   2501#
$ENDAD   005324            507   1797#
$ENDCT   005310           1792#
$EOP     005230           1763   1776#  1856
$EOPCT   005302           1789#  1793
```

```
$ERFLG  001103            909#  2193*  2220
$ERMAX  001115            915#  1285*
$ERROR  006766           1280   2182#
$ERRPC  001116            916#  2200*  2201*  2202   2220   4782   4791   4793   4803   4806
$ERRTB  001630           1113#  2234
$ERTTL  001112            913#  1438*  1664*  1783   2199*  2220
$ESCAP  001160            936#  1284*  1612*  1625*  1644*  1661*  1667*  1713*  1744*  1754*  1850*  2215  2217
                         2220   3762   3813   3817   4094   4154   4363   4372   4373*  4375*
$FILLC  001156            934#  2329   2360
$FILLS  001155            933#  2360
$GDADR  001120            917#
$GDDAT  001124            919#  2192*  4806
$GET42  005314           1794#
$GTSWR  010712           2613#  2952
$HD   = 000000            478
$ICNT   001104            910#
$INTAG  001135            924#  2610*  2629   2649   2762
$ITEMB  001114            914#  2202*  2220   2229
$LF     001170            940#  2220   2360   2745   2755
$LKCSB  001200            947#  1953*
$LKCSR  001176            946#  1948   1954*
$LKS    001206            949#  1958   1963*
$LLVEC  001210            950#  1959
$LPADR  001106            911#
$LPERR  001110            912#  1604*  1631*  1736*  1849   2214
$LPVEC  001202            948#  1949
$MAIL = ****** U          1302   1317   2208   2313
$MNEW   011547           2616   2760#
$MSWR   011536           2613   2758#
$NULL   001154            932#  2331   2360
$OCNT   010066           2393*  2422*  2435#
$OMODE  010070           2388*  2392*  2397   2400*  2411*  2437#
$PASS   001100            907#  1786*  1787*  1804
$QUES   001166            938#  1395   1415   1453   1518   2220   2360   2667   2738   2755
$RDCHR  011164           2680#  2955
$RDDEC= ****** U         2957
$RDLIN  011254           2703#  2956
$RDOCT= ****** U         2957
$RDSZ = 000007           2696#
$RESRE  012106           2902#  2958
$RPADR  001172            944#  1326   4781   4918   4952   4954
$RPVEC  001174            945#  1327   4935
$RTNAD  005336           1803#
$R2A  = ****** U         2959
$SAVRE  012050           2886#  2957
$SB2D   011620           1990   1996   2798#
$SETUP= 000106           1267#  1279   1280   1282   1284   1286   1314   1786   2183   2211   2219   2550   2555
                         2556   2586   2762
$STUP = 177777           1267#
$SUPRS  011560           1991   1997   2773#
$SVPC = 000200            505#   510
$SWR  = 123000            468#   478    483    484    485    486    487    936    937   1284   1773   1786   1796
                         1802   1804   2174   2175   2176   2177   2178   2196   2203   2208   2212   2220
$TKB    001146            929#  2510   2531   2542   2567   2595   2622
$TKCNT  010316           2511#  2526*  2556   2573*  2687   2689*
$TKINT  010334           1313   1439   2526#  2547   2608
```

```
$TKQEN= 010333          2515#   2581    2692
$TKQIN  010320          2512#   2527*   2528    2579*   2580*   2581    2583*
$TKQOU  010322          2513#   2528*   2690    2691*   2692    2694*
$TKQSR  010324          2514#   2527    2583    2694
$TKS    001144           928#   2510    2532*   2563*   2565    2571*   2593    2609*   2619    2631*   2651*
$TKSRV  010404          2529    2542#
$TN   = 000000           468#    478
$TNPWR  011764          2826    2827    2847#
$TPB    001152           931#   2349*   2360
$TPFLG  001157           935#   2307    2360
$TPS    001150           930#   2347    2360
$TRAP   012144          1282    2923#
$TRAP2  012166          2934#   2945
$TRP  = 000014          2938#   2947#   2948#   2949#   2950#   2951#   2952    2953#   2954    2955#   2956#   2957#   2958#
                        2959#
$TRPAD  012200          2928    2945#
$TSTNM  001102           908#   2195    2220
$TTYIN  011510          2705    2706    2718    2736    2750    2754#
$TYPBN= ****** U        2951
$TYPDS  010072          2451#   2950
$TYPE   007424          2307#   2938    2946
$TYPEC  007574          2328    2335    2342    2347#   2348    2653
$TYPEX  007642          2353    2355    2358#
$TYPOC  007670          2391#   2947
$TYPON  007704          2390    2393#   2949
$TYPOS  007644          2386#   2948
$$GET4= 000000          1796#
$OFILL  010067          2387*   2391*   2401    2436#
$40CAT= ****** U        2205
.     - 026232           493#    497#    505     506#    508#    510#    513#    905#    941     1277    1604    1631    1736
                        1804    2220    2360    2505#   2510    2514#   2515    2516#   2754#   2755    2762    2867#   4262#
                        4263#   4264#   4265#   4266#   4267#   4268#   4269#   4270    4649#   4774#   4900#
```

```
CKCHR     468#  1443  1506  2126  2140  4989  5004
CKDIG     468#  2020
CKNUM     468#  4926  4942
COMMEN      1#   629#
ENDCOM      1#   629#
ENDTYP   1769#  1777
ERRCAL   2960#  3762  3813  3816  4093  4154
ERRENT   2167#  2184
ERROR     523#  1614  1615  1616  1617  1618  1619  1646  1647  1648  1649  1650  1656  1666  1673
         1680  1683  1746  1747  1748  1749  1750  1753  1755  1759  3763  3814  3818  4095  4155
         4961
ESCAPE      1#   629#  1612  1644  1661  1744
GETPRI      1#   629#
GETSWR      1#   468#   629#  1314
LOOP      468#  1604  1631  1736
MORETA    899#   942
MULT        1#   629#
NEWTST      1#   629#
POP         1#   629#  2492  2907
PUSH        1#   629#  2451  2887
REPORT      1#   629#
RP04.D      1#  2960
SCOPE     524#
SETPRI      1#   629#  2683
SETTRA   2938#  2947  2948  2949  2950  2952  2954  2955  2956  2957  2958
SETUP       1#   629#  1272
SKIP        1#   629#
SLASH       1#   629#
SPACE     629#
STARS       1#   503   629#   630   634   676   680   871   875   901   941  1261  1265  1771  1810
         1814  2170  2292  2363  2441  2509  2586  2601  2672  2696  2765  2789  2808  2871  2917
         2969  4379  4528  4532  4651  4853  4857
SWRSU       1#   629#  1286#
TRMTRP   2938#
TYPBIN      1#   629#
TYPDEC      1#   629#  1783  2260
TYPNAM      1#   629#
TYPNUM      1#   629#
TYPOCS      1#   629#  1343  1530
TYPOCT      1#   629#  2257  2614
TYPTXT      1#   629#
$$CMRE    899#
$$CMTM    899#
$$ESCA      1#   629#
$$NEWT      1#   629#
$$SET    2938#  2947  2948  2949  2950  2952  2954  2955  2956  2957  2958
$$SKIP      1#   629#
.EQUAT      1#   468#   519
.HEADE      1#   468#
.KT11       1#
.SETUP      1#   468#  1267
.SWRHI      1#   468#   479
.SWRLO    487#   488    489
.$ACT1      1#   468#   501
.$APTB      1#
.$APTH      1#
```

```
.$APTY      1#
.$ASTA      1#
.$CATC      1#     468#     49'
.$CMTA      1#     468#     899
.$DB2D      1#     468#    2806
.$DB2O      1#
.$DIV       1#
.$EOP       1#     468#    1769
.$ERRO      1#     468#    2168
.$ERRT      1#     468#
.$MULT      1#
.$POWE      1#
.$RAND      1#
.$RDDE      1#
.$RDOC      1#
.$READ      1#     468#    2507
.$R2AZ      1#
.$SAVE      1#     468#    2869
.$SB2D      1#     468#    2787
.$SB2O      1#
.$SCOP      1#
.$SIZE      1#
.$SUPR      1#     468#    2763
.$TRAP      1#     468#    2915
.$TYPB      1#
.$TYPD      1#     468#    2439
.$TYPE      1#     468#    2290
.$TYPO      1#     468#    2361
.$40CA      1#
.1170       1#

. ABS.  026232     000

 ERRORS DETECTED:  0

 CZRJBC.BIN,CZRJBC.LST/CRF/SOL/NL:TOC=CZRJBC.SML,RP0456.011,CZRJBC.P11
 RUN-TIME: 18 24 1 SECONDS
 RUN-TIME RATIO: 198/44=4.4
 CORE USED:  46K  (91 PAGES)
```