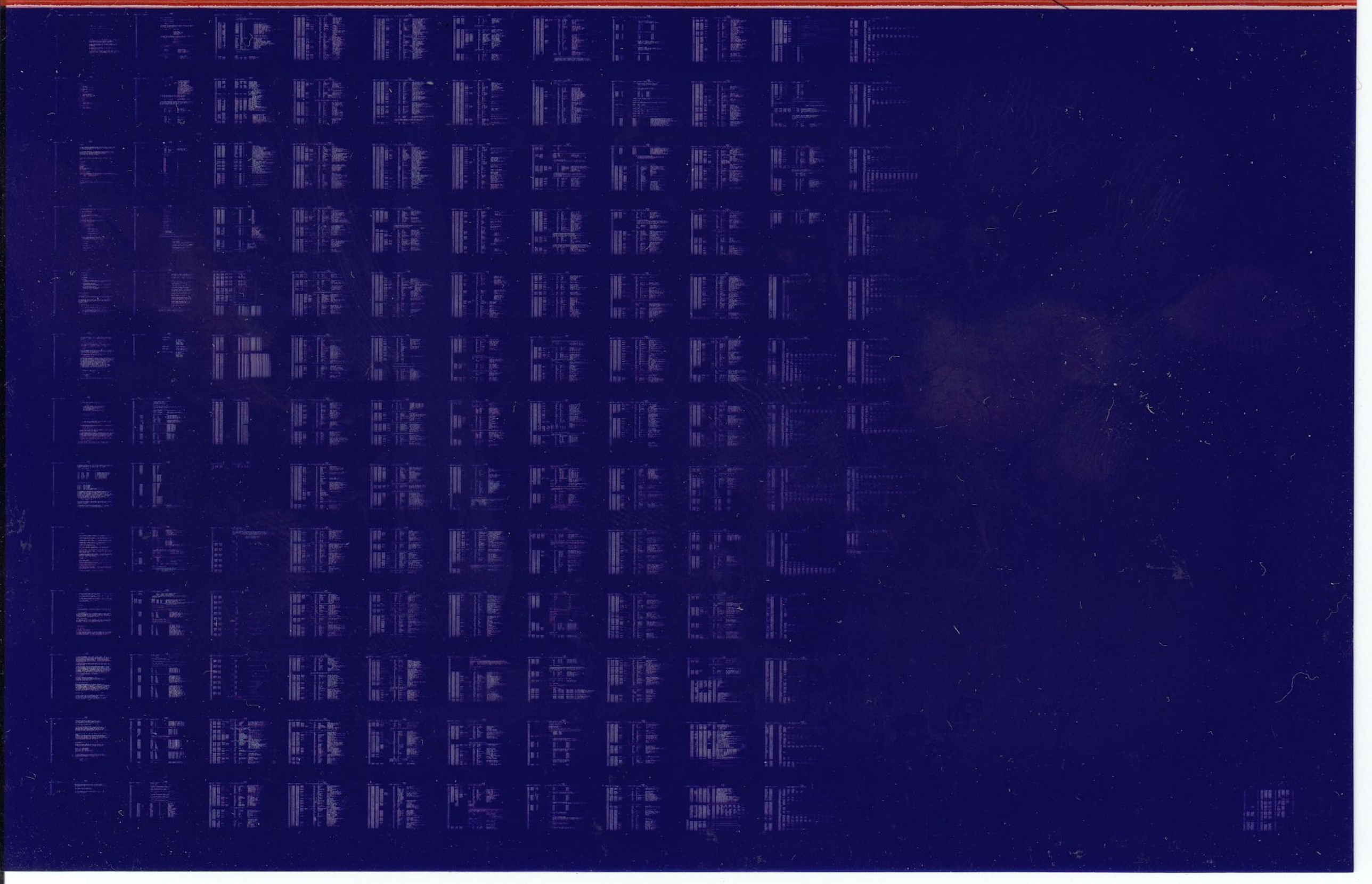
RM03,02

FORMATTER
CZRMAC0

AH-9254C-MC

FICHE 1 OF 1

JAN 1978
digital
MADE IN USA



.REM %

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

PRODUCT CODE: AC-9252C-MC

PRODUCT NAME: CZRMACO RM03/RM02 FORMATTER

DATE CREATED: AUGUST 1977

MAINTAINER: DIAGNOSTIC ENGINEERING

AUTHOR: C. HESS/C. CHEN

CONTENTS

1. ABSTRACT

2. REQUIREMENTS

    2.1 EQUIPMENT
    2.2 PRELIMINARY PROGRAMS

3. LOADING PROCEDURES
    3.1 PAPER TAPE AND XXDP
    3.2 APT
    3.3 APT ETABLE DEFINITIONS

4. STARTING PROCEDURES

    4.1 STARTING ADDRESSES
    4.2 OPERATOR ACTION
    4.3 RH11 - RH70 UNIBUS ADDRESS
    4.4 OTHER UNIBUS ADDRESSES

5. SWITCH REGISTER SETTINGS

6. ERROR MESSAGES

7. MISCELLANEOUS

    7.1 FORMAT TIMES
    7.2 HALTING THE PROGRAM
    7.3 SURFACE VERIFICATION

8. PROGRAM DESCRIPTION

    8.1 FORMAT OPERATION
    8.2 CHECK OPERATION
    8.3 POSITIONER VERIFICATION

9. BAD SPOT FILE

10. RM03 SOFTWARE DRIVER DOCUMENT

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


                                THE RM03 FORMATTER PROGRAM PROVIDES THE FACILITIES TO FORMAT OR
                                TO CHECK THE HEADER AND DATA FIELDS OF EACH DATA BLOCK ON THE DISC PACK.
                                EACH HEADER FIELD SPECIFIES THE ADDRESS OF ITS ASSOCIATED DATA
                                BLOCK ON THE DISK PACK.


                                IN THE FORMAT OPERATION, THE PROGRAM WRITES THE HEADER OF EACH
                                DATA BLOCK WITH A CYLINDER NUMBER,TRACK NUMBER AND SECTOR NUMBER;
                                WRITES THE DATA FIELD WITH SELECTED DATA PATTERN.
                                THE PROGRAM THEN VERIFIES THE WRITTEN DATA BLOCKS VIA EXECUTING THE
                                "WRITE CHECK HEAD AND DATA" COMMAND.



                                IN THE CHECK OPERATION,THE PROGRAM REPEATS THE FORMAT OPERATION
                                THREE TIMES WITH DATA PATTERN BEING ROTATED ONE BIT AT EACH
                                PASS.


                        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 RM03 DISK DRIVES
                                DISK PACK(S)

                        2.2     PRELIMINARY PROGRAMS

                                THERE ARE NO PREQUISITE PROGRAMS PROVIDING THE RM03 SUBSYSTEM IS
                                KNOWN TO BE OPERATIONAL.  IF THE STATE OF THE RM03 SUBSYSTEM IS UNKNOWN,
                                THE FOLLOWING PROGRAMS SHOULD BE RUN PRIC TO USING THE FORMATTER PROGRAM
                                        RM03    DISKLESS DIAGNOSTIC
                                        RM03    FUNCTIONAL TEST


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

                        3.1     PAPER TAPE AND XXDP

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


                        3.2     APT
```

```
                        THIS PROGRAM IS APT COMPATIBLE TO THE EXTENT THAT APT HOOKS WILL
                        BE IN THE PROGRAM AND WILL WORK THRU THE 'OPTION INTERFACE'.

                        FOR OTHER INTERFACES,APT MAY ONLY LOAD AND START THE PROGRAM.
                        I.E. LOAD AND DUMP MODE.

                        AUTOMATIC MODE (MONITOR)

                        1.  THE INPUT DIALOGUE IS BYPASSED.
                        2.  THE BUSS ADDRESS AND CONTROLLER INTERRUPT VECTOR IS
                            DEFAULTED.
                        DUMP MODE: INPUT DIALOGUE AFTER PROGRAM STARTS

                3.3     APT ETABLE DEFINITIONS

                        THE FOLLOWING DEFINITIONS ARE VALID FOR SPECIFYING APT
                        ENVIRONMENTAL TABLE (ETABLE) ENTRIES,VIA RUNNING
                        THE APT UTILITY PROGRAM "TSP":

                        1.    SOFTWARE ENVIRONMENT:

                              = 1 IF APT SCRIPT MODE
                              = 0 IF STANDLONE MODE

                        2.    ENVIRONMENT MODE:

                              BIT 7 = 1  ETABLE DOES SIZING
                                    = 0  PROGRAM DOES SIZING

                              BIT 6 = 1  SPOOL MESSAGES TO APT IF SCRIPT MODE
                                    = 0  DON'T SPOOL TO APT

                              BIT 5 = 1  SUPPRESS CONSOLE OUTPUT
                                    = 0  ALLOW CONSOLE OUTPUT

                              BIT 4 TO BIT 0 ARE NOT USED

                        3.    SWITCH 1 (SOFTWARE SWITCH REGISTER)
                              IF ENVIRONMENT MODE BIT 7 (SIZING BIT ) IS SET TO 1,
                              THE SOFTWARE SWITCH REGISTER WILL BE USED, INSTEAD
                              OF THE HARDWARE CONSOLE SWITCH REGISTER.

                        4.    SWITCH 2 (USER SWITCH REGISTER)
                                 NOT USED

                        5.    CPU OPTIONS
                                 NOT USED

                        6.    MEMORY TYPES 1-4 AND MAX MEMORY ADDRESSES
                                 NOT USED

                        7.    INTERRUPT VECTOR 1:
                                 USED WHEN ENVIRONMENT MODE BIT 7 = 1;DEFAULT = 254

                        8.    BUS PRIORITY 1:
                                 NOT USED.
```

9. INTERRUPT VECTOR 2:
NOT USED

10. BUS PRIORITY 2:
NOT USED

11. BASE ADDRESS:
USED WHEN ENVIRONMENT MODE BIT 7 = 1;DEFAULT = 176700

12. DEVICE MAP:
USED WHEN ENVIRONMENT MODE BIT 7 = 1. EACH BIT SET TO 1 IN BITS 0 TO 7 WILL SELECT THE CORRESPONDING DRIVE TO BE TESTED. BITS 8-15 ARE NOT USED.

13. CONTROLLER DESCRIPTOR WORDS:
USED WHEN ENVIRONMENT MODE BIT 7 = 1. ANY NONE ZERO NUMBER SPECIFIES THAT PROGRAM RUNS IN CHECK MODE.

14. CONTROLLER DESCRIPTOR WORDS:
USED WHEN ENVIRONMENT MODE BIT 7 = 1. ANY NONE ZERO NUMBER SPECIFIES THAT PROGRAM SELECTS 30 SECTORS FOR A TRACK IN ALL OPERATIONS.

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 FROM ITS DEFAULT VALUE OF 176700.

THE PROGRAM IS STARTED FROM LOCATION 204(8) IF THE ADDRESS OF THE RH11 OR RH70 IS TO BE CHANGED FROM THE PRELOADED VALUE. NOTE THAT STARTING ADDRESS 204 IS VALID ONLY ONCE AFTER THE PROGRAM IS LOADED.
(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 32 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 TRACK MUST BE
LESS THAN THE ENDING TRACK ADDRESS IF THESE TWO
ADDRESSES ARE ON THE SAME CYLINDER. ON THE OTHER HAND,
THE ENDING CYLINDER ADDRESS MAY BE LESS THAN THE STARTING
CYLINDER ADDRESS. IN THIS CASE ,THE PROGRAM WILL FORMAT
OR VERIFY THE DISK PACK FROM THE STARTING CYLINDER TO
CYLINDER 822 AND THEN FROM CYLINDER 0 TO THE ENDING CYLINDER.

8. THE PROGRAM WILL THEN ASK FOR THE DATA PATTERN:
    (IN CASE OF FORMAT MODE)

```
'SELECT DATA PATTERN
  (0) ZERO'S
  (1) ONE'S
  (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:

066666

NOTE: IN THE CHECK OPERATION, THE ABOVE MENTIONED MESSAGE IS BYPASSED. THE DATA PATTERN IS INITIATED TO 133331 AND IS ROTATED ONE BIT TO THE RIGHT AT EACH PASS OF THE CHECK OPERATION. AFTER THE CHECK OPERATION IS DONE, THE DATA PATTERN WILL BE RESTORED TO 066666.

9. THE PROGRAM WILL THEN TYPE:

'STARTING FORMAT ON DRIVE N'

OR

'STARTING CHECK ON DRIVE N'

IF A NEW PACK IS UNDER FORMATTED, THE PROGRAM ALSO ASKS TO ENTER TWO SERIAL NUMBERS. THESE TWO SERIAL NUMBER MUST BE NON-ZERO DECIMAL NUMBERS.

10. THE OPERATOR CAN DETERMINE WHERE THE DRIVE IS DURING THE FORMAT OR CHECK OPERATION BY TYPING A 'CONTROL O'. 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.

11. 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.

12. 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
1160	$TKS	177560	TTY KEYBOARD STATUS REGISTER
1162	$TKB	177562	TTY KEYBOARD BUFFER REGISTER
1164	$TPS	177564	TTY PRINTER STATUS REGISTER
1166	$TPB	177566	TTY PRINTER BUFFER REGISTER
1276	$LKCSR	172540	KW11-P CONTROL REGISTER
1300	$LKCSB	172542	KW11-P COUNTER REGISTER
1302	$LPVEC	104	KW11-P VECTOR ADDRESS
1304	$LKS	177546	KW11-L CONTROL REGISTER
1306	$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<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 RM03 INTERRUPT. THE
'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE
TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

'SWR = NNNNNN   NEW ='

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

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

BE FOLLOWED.

6. ERROR MESSAGES

1. 'RH11 INTERRUPT OCCURRED (RMAS=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 ONE OF THE FOLLOWING ERRORS: 'DCK', 'OPI', 'DTE', 'HCRC', 'HCE', OR 'FER'.

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.

7. MISCELLANEOUS

7.1 FORMAT TIME

IT TAKES APPROXIMATELY 10 MINUTES TO FORMAT AN ENTIRE RM03 PACK. THE 'CHECK' MODE TIME FOR AN ENTIRE RM03 PACK IS 24 MINUTES.

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 O' 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 OPERATOR.

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 'NONRECOVERABLE'. 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.

8.2 CHECK OPERATION

THE CHECK OPERATION IS IDENTICAL TO THE
THE FORMAT OPERATION DESCRIBED IN SECTION 8.1,
EXCEPT THAT IN EACH CHECK OPERATION THE FORMAT OPERATION
IS REPEATED THREE TIMES WITH DATA PATTERN BEING ROTATED
ONE BIT POSITION AT EACH PASS.

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. BAD SPOT FILE

SECTORS 0 THROUGH 31,ON CYLINDER 822., TRACK 4 ARE ALWAYS FORMATTED
IN 16 BIT MODE. THEY CONTAIN THE BAD SECTOR ADDRESS ON THE
PACK FOR BOTH 30 AND 32 SECTOR OPERATIONS.
THE DATA FIELD OF EACH SECTOR OF THE BAD SPOT FILE IS DESCRIPTED BELOW:
THE FIRST TWO WORDS IN THE DATA FIELD SPECIFY THE SERIAL
NUMBER OF THE PACK. THE SERIAL NUMBER MUST BE A NON-ZERO

```
598                     NUMBER AND CONSISTS OF MAXIMUM 10 OCTAL DIGITS.
599                     THE FIRST WORD OF THE SERIAL NUMBER CONTAINS THE 5
600                     LEST SIGNIFICANT OCTAL DIGITS,WHILE THE SECOND
601                     WORD CONTAINS THE 5 MOST SIGNIFICANT DIGITS.
602
603                     THE THIRD WORD IS ALWAYS ZERO.
604
605                     THE FORTH WORD DEFINES THE PACK IS A DATA PACK ,IF IT IS 0.
606
607                     THE FOLLOWING 252 WORDS DEFINE THE BAD SECTOR ADDRESS;TWO
608                     WORDS ARE USED TO DEFINE A SIGNLE BAD SECTOR, THE FIRST
609                     WORD SPECIFIES THE CYLINDER ADDRESS, THE SECOND WORD
610                     SPEDIFIES THE TRACK (HIGH BYTE) AND SECTOR (LOW BATE) ADDRESSES.
611                     ALL UNUSED LOCATIONS ARE RECORDED WITH  ONES.
612
613
614
615                     THE SECTORS ON THE LAST TRACK ARE FURTHER DIVIDED INTO FOUR
616                     GROUPS.
617                     SECTORS 0,2,4,6 AND 8 CONTAIN THE MANUFACTURE DEFINED BAD SECTORS
618                     FOR 16 BIT MODE;SECTORS 1,3,5,7 AND 9  FOR 18 BIT MODE.
619                     SECTORS 10,12...THROUGH 30 CONTAIN THE USER DEFINED BAD SECTORS
620                     FOR 16 BIT MODE;SECTORS 11,13..THROUGH 31  FOR 18 BIT MODE.
621                     THE SECTORS IN EACH GROUP ARE IDENTICAL AND SECTORS 0,1,10AND 11 ARE
622                      USED AS REFERENCE SECTORS BY THE PROGRAM.
623
624
625
626                     THE PROGRAM PROVIDES A FEW UTILITY ROUTINES TO HANDLE THE BAD
627                     SECTOR FILE.
628
629                     BEFORE THE PROGRAM FORMATS THE PACK,THE PROGRAM TYPES THE FOLLOWING
630                     MESSAGE TO ALLOW THE OPERATOR TO ACCESS THE ROUTINES
631                     BY ENTERING THE SELECTED FUNCTION NUMBER FOLLOWED BY <CR>.
632
633
634                     KEYIN  U.S.R. BAD SECTORS.....................0
635                     PRINT M.F.G. BAD SECTORS.....................1
636                     PRINT U.S.R. BAD SECTORS.....................2
637                     INITIALIZE U.S.R. BAD SECTOR FILE............3
638                     PRINT A SECTOR OF THE LAST TRACK.............4
639                     EXIT.........................................<CR>
640
641
642
643                     THE FIRST ROUTINE ALLOWS THE USER DEFINED BAD SECTORS
644                     TO BE ENTERED MANUALLY BY SPECIFING THE CYLINDER ADDRESS, TRACK
645                     ADDRESS AND SECTOR ADDRESS IN DECIMAL VALUE.  TO EXIT FROM
646                     THE ROUTINE JUST ENTER A . FOLLOWED BY <CR>.
647
648                     EXAMPLE:
649                             CYLINDER  -1/200
650                             TRACK  -1/4
651                             SECTOR  -1/23
652                             CYLINDER  -1/.<CR>
653
```

```
                        THE SECOND AND THIRD ROUTINES TYPE THE MANUFACTURE AND USER
                        DEFINED BAD SECTORS ON THE CONSOLE IN OCTAL NUMBER.
                        THE FIRST FOUR WORDS ON THE LISTING ARE THE SERIAL NUMBER FOLLOWED
                        BY TWO 0 'S.

                        THE FOURTH  ROUTINE RESETS ALL
                        USER DEFINED BAD SECTOR ADDRESSES TO -1.


                        THE FIFTH ROUTINE ALLOWS A WHOLE SECTOR OF THE BAD SPOT FILE
                        RETRIEVED FROM THE PACK BEING FORMATTED TO  BE
                        TYPED ON THE CONSOLE.


                        TO EXIT THE ROUTINE JUST ENTER .<CR>   .
```

10.1 RH70(11)/RM03 DRIVER

THIS DOCUMENT IS THE USER'S GUIDE FOR THE RH70/RM03 DRIVER.

10.2 TO INITIALIZE THE DRIVER:

```
        JSR     PC,RMINIT
        RETURN
```

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

DRVSTA	DRIVE STATE
>0	ONLINE RM03
=0	OFFLINE RM03, DRIVE IS NOT AN RM03, OR NONEXISTENT DRIVE
<0	UNSAFE RM03

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

DRVTYP	CONDITION
0	NONEXISTENT DRIVE
4	RM03
-1	NOT AN RM03

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

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

CALL:

```
        JSR     R0,RM03         ;MAKE THE CALL
        PNTDPB                  ;ADDRESS OF DPB*
        RETURN1                 ;RETURN IF QUEUE IS FULL
        RETURN2                 ;RETURN IF REQUEST IS IN
                                ;QUEUE OR THERE IS AN
                                ;ERROR CONDITION
```

*DPB (DATA PARAMETER BLOCK)

```
PNTDPB: .BYTE   0               ;(0) DRIVE NUMBER
        .BYTE   0               ;(1) OFFSET VALUE OR FMT22, ECT, AND HCI
        .BYTE   0               ;(2) COMMAND
        .BYTE   0               ;(3) PSEL AND A17 AND A16
        .WORD   0               ;(4) WORD COUNT (MUST BE NEG.)
```

```
                        .WORD   0               ;(6) BUFFER ADDRESS OR
                                                ;REGISTER TABLE POINTER
                        .BYTE   0               ;(10) SECTOR ADDRESS OR
                                                ;FIRST REG. INDEX
                        .BYTE   0               ;(11) TRACK ADDRESS OR
                                                ;LAST REG. INDEX
                        .WORD   0               ;(12) CYLINDER ADDRESS
                        .WORD   0               ;(14) ERROR TABLE POINTER
                                                ;POINTS TO THE FIRST OF TWENTY
                                                ;LOCATIONS OF WHERE THE DRIVER
                                                ;IS TO STORE THE RH70/RM03
                                                ;REGISTERS ON AN ERROR. IF LEFT
                                                ;ZERO REGISTERS ARE NOT SAVED.
                        .WORD   0               ;(16) STATUS/ERROR INDICATOR
                                                ;BIT15=1=>ERROR OCCURRED
                                                ;BIT07=1=>DONE
                                                ;BIT14-BIT09 AND BIT06-BIT03
                                                ;INDICATE TYPE OF ERROR

10.4    THE DRIVER PROVIDES A SOFTWARE TIMEOUT CAPABILITY.
TO UTILIZE THIS CAPABILITY YOU MUST SUPPLY THE "RM TIMER" ROUTINE
WITH THE ELAPSED TIME IN THE FOLLOWING MANNER:

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

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

10.4.1  EXAMPLE - WRITE 1000. WORDS

        1$:     JSR     R0,RM03         ;CALL THE DRIVER
                WRTDPB                  ;DPB ADDRESS
                BR      1$              ;WAIT FOR QUEUE IF FULL
        2$:     TST     WRTDPB+16       ;WAIT FOR COMMAND TO COMPLETE
                BEQ     2$
                BMI     ERROR1          ;ERROR OCCURRED
                .
                .
                .

        WRTDPB: .BYTE   5               ;DRIVE #5
                .BYTE   0               ;
                .BYTE   161             ;WRITE COMMAND
                .BYTE   0               ;
                .WORD   -1000.          ;WORD COUNT
                .WORD   WRTBUF          ;BUFFER ADDRESS
                .BYTE   3               ;SECTOR
                .BYTE   5               ;TRACK
                .WORD   400             ;CYLINDER
                .WORD   ERRTBS          ;ERROR TABLE
```

```
                        .WORD   0               ;STATUS/ERROR INDICATOR

                        ALTERNATE DPB SETUP

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

10.5 RH70/RM03 REGISTERS

MNEMONIC	INDEX
RMCS1	0
RMWC	2
RMBA	4
RMDA	6
RMCS2	10
RMDS	12
RMER1	14
RMAS	16
RMLA	20
RMDB	22
RMMR1	24
RMDT	26
RMSN	30
RMOF	32
RMDC	34
RMHR	36
RMMR2	40
RMER2	42
RMEC1	44
RMEC2	46

10.6 COMMANDS PERFORMED BY THE DRIVER

COMMAND	CODE	COMMAND TYPE
NO OPERATION	101	N
UNLOAD	103	N
SEEK	105	P
RECALIRATE	107	P
DRIVE CLEAR	111	N
RELEASE	113	N
OFFSET	115	P

RETURN TO CENTER	117	P
READIN PRESET	121	N
PACK ACKNOWLEDGE	123	N
SEARCH	131	P
GET REGISTER(S)	141	S
SET FORMAT	143	S
SELECT DRIVE	145	S
WRITE CHECK DATA	151	D
WRITE CHECK HEADER AND DATA	153	D
WRITE DATA	161	D
WRITE HEADER & DATA	163	D
READ DATA	171	D
READ HEADER & DATA	173	D

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

10.7 DPB STATUS/ERROR INDICATOR WORD

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

BIT NO.	MEANING IF ON A "1"
15	ERROR OCCURRED DONE (BIT07=0); BITS 14-10 SPECIFIES TYPE DONE (BIT07=1); BITS 06-03 SPECIFIES TYPE
14(1)	USER MADE A REQUEST FOR A FUNCTION TO BE PERFORMED ON AN OFFLINE OR UNSAFE DRIVE
13(1)	USER MADE A REQUEST FOR A FUNCTION TO BE PERFORMED ON A DRIVE THAT HAS AN UNLOAD REQUEST IN QUEUE.
12(2)	PERSISTENT UNSAFE CONDITION EXIST.
11(2)	UNCORRECTABLE PARITY ERROR OCCURRED

10(2)(4)	FATAL PARITY ERROR. A MASSBUS CLEAR WAS PERFORMED, ALL QUEUES WERE EMPTIED, AND ALL DRVACT'S SET TO THE IDLE STATE
9(3)(4)	SOFTWARE TIMEOUT OCCURRED ON THIS DRIVE
8(4)	SOFTWARE TIMEOUT OCCURRED ON ANOTHER DRIVE
7	DONE
6(2)	ERROR OCCURRED DURING AN I/O OPERATION
5(2)	ERROR OCCURRED DURING AN OPERATION OTHER THAN I/O.
4(2)	CORRECTABLE UNSAFE CONDITION OCCURRED
3(2)	DRIVE ERROR OCCURRED THAT CAUSED AN AUTOMATIC "RECALIBRATE" SEQUENCE
2	PORT REQUEST TIMEOUT. THE DRIVER REQUESTED THE DRIVE BUT THE OPPOSITE PORT DID NOT RELEASE THE DRIVE WITHIN 20 SECONDS.
1	NON-EXISTENT DRIVE REQUESTED. USER MADE A REQUEST FOR A NON-EXISTENT DRIVE.
(1) =>	REQUEST WASN'T PUT IN QUEUE. (RH70/RM03 REGISTERS WERE NOT SAVED)
(2) =>	REQUEST QUEUE HAS BEEN EMPTIED. THE DRIVER ISSUED A "DRIVE CLEAR" TO THE DRIVE. NOTE: ALL RH70/RM03 REGISTERS ARE SAVED AS PER DPB+14 BEFORE THE "DRIVE CLEAR".
(3) =>	REQUEST QUEUE HAS BEEN EMPTIED. THE DRIVER ISSUED A MASSBUS INIT. ALL RH70/RM03 REGISTERS FOR THE DRIVE WERE SAVED AS PER DPB+14 BEFORE THE INIT.
(4) =>	A "RECALIBRATE" SHOULD BE ISSUED BEFORE ANY OTHER COMMAND.

10.8 ERROR CALLS MADE BY THE DRIVER.

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

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

N	TYPE	DATA AVAILABLE

```
                                1       RH70 INTERRUPT          *R4= PMCS1'S ADDRESS
                                        OCCURRED (RHAS=0)

                                2       UNEXPECTED ATTENTION     R1= DRIVE NUMBER
                                        OCCURRED                 R3= ATA BIT
                                                                *R4= RMCS1'S ADDRESS
                                                                 R5= (RMAS)
                                                                 RMERRS  =RMDS
                                                                 RMERRS+2=RMER1
                                                                 RMERRS+4=RMER2
                                                                 RMERRS+6=RMMR2

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

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

                                5       ADDRESS PLUG CHANGE      R1= DRIVE NUMBER
                                        BIT SET ('OPE' ERROR)    R3= ATA BIT
                                                                *R4= RMCS1'S ADDRESS
                                                                 R5= (RMAS)
                                                                 RMERRS  =RMDS
                                                                 RMERRS+2=RMER1
                                                                 RMERRS+4=RMER2
                                                                 RMERRS+6=RMMR2

                                * THIS IS THE ACTUAL UNIBUS ADDRESS (176700)

                                %
```

```
                        .TITLE  CZRMACO RM03/2 FORMATTER
                        ;*COPYRIGHT (C) 1977
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY C. HESS/C. CHEN
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                        ;*
                        .SBTTL  BASIC DEFINITIONS

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

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

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

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

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
```

```
1037            004000                  SW11=   4000
1038            002000                  SW10=   2000
1039            001000                  SW09=   1000
1040            000400                  SW08=   400
1041            000200                  SW07=   200
1042            000100                  SW06=   100
1043            000040                  SW05=   40
1044            000020                  SW04=   20
1045            000010                  SW03=   10
1046            000004                  SW02=   4
1047            000002                  SW01=   2
1048            000001                  SW00=   1
1049                                    .EQUIV  SW09,SW9
1050                                    .EQUIV  SW08,SW8
1051                                    .EQUIV  SW07,SW7
1052                                    .EQUIV  SW06,SW6
1053                                    .EQUIV  SW05,SW5
1054                                    .EQUIV  SW04,SW4
1055                                    .EQUIV  SW03,SW3
1056                                    .EQUIV  SW02,SW2
1057                                    .EQUIV  SW01,SW1
1058                                    .EQUIV  SW00,SW0
1059
1060                                    ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
1061            100000                  BIT15=  100000
1062            040000                  BIT14=  40000
1063            020000                  BIT13=  20000
1064            010000                  BIT12=  10000
1065            004000                  BIT11=  4000
1066            002000                  BIT10=  2000
1067            001000                  BIT09=  1000
1068            000400                  BIT08=  400
1069            000200                  BIT07=  200
1070            000100                  BIT06=  100
1071            000040                  BIT05=  40
1072            000020                  BIT04=  20
1073            000010                  BIT03=  10
1074            000004                  BIT02=  4
1075            000002                  BIT01=  2
1076            000001                  BIT00=  1
1077                                    .EQUIV  BIT09,BIT9
1078                                    .EQUIV  BIT08,BIT8
1079                                    .EQUIV  BIT07,BIT7
1080                                    .EQUIV  BIT06,BIT6
1081                                    .EQUIV  BIT05,BIT5
1082                                    .EQUIV  BIT04,BIT4
1083                                    .EQUIV  BIT03,BIT3
1084                                    .EQUIV  BIT02,BIT2
1085                                    .EQUIV  BIT01,BIT1
1086                                    .EQUIV  BIT00,BIT0
1087
1088                                    ;*BASIC "CPU" TRAP VECTOR ADDRESSES
1089            000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
1090            000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
1091            000014                  TBITVEC=14              ;;"T" BIT
1092            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
                    176700                  ABASE=176700
                    000254                  AVECT1=254

                                            .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  015742                              $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
                    000200                  .=200
000200  000137  007014                              JMP     BEGIN1  ;NORMAT STARTING ADDRESS
000204  000137  006776                              JMP     BEGIN   ;CHANGE THE RH11 ADDRESS

                    001100                  .=1100
                                            .SBTTL  APT PARAMETER BLOCK

                                            ;;**************************************************************
                                            ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                            ;;**************************************************************
                    001100                          .$X=.   ;;SAVE CURRENT LOCATION
                    000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024  000200                              200     ;;FOR APT START UP
```

```
                                .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
                               $APTHDR ;;POINT TO APT HEADER BLOCK
                                .=.$X   ;;RESET LOCATION COUNTER
                              ;;****************************************************************
                              ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                              ;INTERFACE SPEC.

001100                        $APTHD:
001100  000000                $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001102  001206                $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001104  000310                $TSTM:  .WORD   200.    ;;RUN TIM OF LONGEST TEST
001106  000310                $PASTM: .WORD   200.    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001110  000310                $UNITM: .WORD   200.    ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001112  000032                        .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
001114                        TAB.XY  =       .       ;CMTAGSTARTING ADDRESS

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

                              .SBTTL  RH11 REGISTERS

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

                              ;CONTROL AND STATUS REGISTER 1 (RMCS1)

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

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

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

                              ;CONTROL AND STATUS REGISTER 2 (RMCS2)

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

```
                        DLT=    100000                  ;DATA LATE (BIT #15)

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


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

                        .SBTTL  RM03 REGISTERS

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

                        ;CONTROL AND STATUS 1 REGISTER. (#00)

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

                        ;DRIVE STATUS REGISTER (RMDS1) (#01)

000100                  VV=     100                     ;VOLUME VALID (BIT #6)
000200                  DRY=    200                     ;DRIVE READY (BIT #7)
000400                  DPR=    400                     ;DRIVE PRESENT (BIT #8)
001000                  PGM=    1000                    ;PROGRAMABLE (BIT #9)
002000                  LST=    2000                    ;LAST SECTOR TRANSFERRED (BIT #10)
004000                  WRL=    4000                    ;WRITE LOCK (BIT #11)
010000                  MOL=    10000                   ;MEDIUM ON-LINE (BIT #12)
020000                  PIP=    20000                   ;POSITIONING OPERATION IN PROGRESS (BIT #13)
040000                  ERR=    40000                   ;COMPOSITE ERROR (BIT #14)
100000                  ATA=    100000                  ;ATTENTION ACTIVE (BIT #15)

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

000001                  ILF=    1                       ;ILLEGAL FUNCTION (BIT #0)
000002                  ILR=    2                       ;ILLEGAL REGISTER (BIT #1)
000004                  RMR=    4                       ;REGISTER MODIFICATION REFUSED (BIT #2)
000010                  PAR=    10                      ;PARITY ERROR (BIT #3)
000020                  FER=    20                      ;FORMAT ERROR (BIT #4)
000040                  WCF=    40                      ;WRITE CLOCK FAIL (BIT #5)
000100                  ECH=    100                     ;ECC HARD ERROR (BIT #6)
000200                  HCE=    200                     ;HEADER COMPARE ERROR (BIT #7)
000400                  HCRC=   400                     ;HEADER CRC ERROR (BIT #8)
001000                  AOE=    1000                    ;ADDRESS OVERFLOW ERROR (BIT #9)
002000                  IAE=    2000                    ;INVALID ADDRESS ERROR (BIT #10)
004000                  WLE=    4000                    ;WRITE LOCK ERROR (BIT #11)
010000                  DTE=    10000                   ;DRIVE TIMING ERROR (BIT #12)
020000                  OPI=    20000                   ;OPERATION INCOMPLETE (BIT #13)
040000                  UNS=    40000                   ;DRIVE UNSAFE (BIT #14)
100000                  DCK=    100000                  ;DATA CHECK ERROR (BIT 15)

                        ;MAINTAINABILITY REGISTER (RMMR1)(#03)
```

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

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

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

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

                ;DRIVE TYPE REGISTER (RMDT) (#06)

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

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

000100          SC01=   100     ;SECTOR COUNT FIELD 0 (BIT #6)
000200          SC02=   200     ;SECTOR COUNT FIELD 1 (BIT #7)
000400          SC04=   400     ;SECTOR COUNT FIELD 2 (BIT #8)
001000          SC10=   1000    ;SECTOR COUNT FIELD 3 (BIT #9)
002000          SC20=   2000    ;SECTOR COUNT FIELD 4 (BIT #10)
004000          TRK1=   4000    ;TRACK FIELD 1 (BIT #11)
010000          TRK2=   10000   ;TRACK FIELD 2 (BIT #12)
020000          TRK4=   20000   ;TRACK FIELD 3 (BIT #13)
040000          TRK10=  40000   ;TRACK FIELD 4 (BIT #14)
100000          TRK20=  100000  ;TRACK FIELD 5 (BIT #15)

                ;OFFSET REGISTER (RMOF) (#11)

000001          OFDIR=  1       ;OFFSET DIRECTION FOR RM03
002000          HCI=    2000    ;HEADER COMPARE INHIBIT (BIT #10)
004000          ECI=    4000    ;ERROR CORRECTION CODE INHIBIT (BIT #11)
```

```
1317            010000                  FMT16=  10000                   ;FORMAT BIT (BIT #12
1318
1319                                    ;DESIRED CYLINDER ADDRESS (RMDC) (#12)
1320                                    ;(EACH BIT IS CALLED BY BIT NUMBER)
1321
1322                                    ;SERIAL NUMBER REGISTER (RMSN) (#14)
1323                                    ;(EACH IS CALLED BY BIT NUMBER)
1324
1325                                    ;RM03 MAINTENANCE  REGISTER #02 (RMMR2) (#15)
1326
1327            040000                  SKI=    40000                   ;SEEK INCOMPLETE (BIT #14)
1328
1329                                    ;ECC POSITION REGISTER (RMEC1) (#16)
1330                                    ;(EACH BIT IS CALLED BY BIT NUMBER)
1331
1332                                    ;ECC PATTERN REGISTER (RMEC2) (#17)
1333                                    ;(EACH BIT IS CALLED BY BIT NUMBER)
1334
1335                                    ;;*****************************************************************
1336
1337                                    .SBTTL  RM03 DRIVER COMMANDS
1338
1339                                    ;;*****************************************************************
1340
1341            000101                  RNOP    =       101             ;NO OPERATION
1342            000105                  SEEK    =       105             ;SEEK
1343            000107                  RECAL   =       107             ;RECALIBRATE
1344            000111                  DRVCLR  =       111             ;DRIVE CLEAR
1345            000113                  RELSE   =       113             ;RELEASE
1346            000115                  OFFSET  =       115             ;OFFSET
1347            000117                  RTC     =       117             ;RETURN TO CENTER LINE
1348            000121                  READIN  =       121             ;READ IN PRESET
1349            000123                  ACK     =       123             ;PACK ACKNOWLEDGE
1350            000131                  SEARCH  =       131             ;SEARCH
1351            000141                  GETREG  =       141             ;GET REGISTERS
1352            000143                  SETFMT  =       143             ;SET FORMAT (& ECI OR HCI)
1353            000145                  SELDRV  =       145             ;SELECT DRIVE
1354            000151                  WCKD    =       151             ;WRITE CHECK DATA
1355            000153                  WCKHD   =       153             ;WRITE CHECK HEADER & DATA
1356            000161                  WRTDAT  =       161             ;WRITE DATA
1357            000163                  WRTHD   =       163             ;WRITE HEADER & DATA
1358            000171                  RDDAT   =       171             ;READ DATA
1359            000173                  RDHD    =       173             ;READ HEADER & DATA
1360
1361
```

```
                                .SBTTL  COMMON TAGS

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

001114                                  .=TAB.XY
001114                          $CMTAG:                         ;;START OF COMMON TAGS
001114  000000                          .WORD   0
001116  000                     $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001117  000                     $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001120  000000                  $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001122  000000                  $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001124  000000                  $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001126  000000                  $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001130  000                     $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001131  001                     $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001132  000000                  $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001134  000000                  $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001136  000000                  $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001140  000000                  $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001142  000000                  $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001144  000000                          .WORD   0               ;;RESERVED--NOT TO BE USED
001146  000000                          .WORD   0
001150  000                     $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001151  000                     $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001152  000000                          .WORD   0
001154  177570                  SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001156  177570                  DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001160  177560                  $TKS:   177560                  ;;TTY KBD STATUS
001162  177562                  $TKB:   177562                  ;;TTY KBD BUFFER
001164  177564                  $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001166  177566                  $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001170  000                     $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001171  002                     $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001172  012                     $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001173  000                     $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001174  000000                  $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001176  177607  000377          $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001202  077                     $QUES:  .ASCII  /?/             ;;QUESTION MARK
001203  015                     $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001204  000012                  $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;**********************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

                                ;;**********************************************************************
                                .NLIST ME
                                ;
                                ;

                                .EVEN
001206                          $MAIL:                          ;;APT MAILBOX
001206  000000                  $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
001210  000000                  $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
001212  000000                  $TESTN: .WORD   ATESTN          ;;TEST NUMBER
001214  000000                  $PASS:  .WORD   APASS           ;;PASS COUNT
```

```
1418  001216  000000                  $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
1419  001220  000000                  $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
1420  001222  000000                  $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
1421  001224  000000                  $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
1422  001226                          $ETABLE:                ;;APT ENVIRONMENT TABLE
1423  001226     000                  $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
1424  001227     000                  $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
1425  001230  000000                  $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
1426  001232  000000                  $USWR:  .WORD   AUSWR   ;;USER SWITCHES
1427  001234  000000                  $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
1428                                                          BITS 15-11=CPU TYPE
1429                                                          :*
1430                                                          :*
1431                                                          BIT 10=REAL TIME CLOCK
1432                                                          BIT  9=FLOATING POINT PROCESSOR
1433                                                          BIT  8=MEMORY MANAGEMENT
1434  001236     000                  $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
1435  001237     000                  $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
1436                                                          MEM.TYPE BYTE   --  (HIGH BYTE)
1437                                                          :*
1438                                                          :*
1439                                                          :*
1440  001240  000000                  $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
1441                                                          MEM.LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
1442  001242     000                  $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
1443  001243     000                  $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
1444  001244  000000                  $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
1445  001246     000                  $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
1446  001247     000                  $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
1447  001250  000000                  $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
1448  001252     000                  $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
1449  001253     000                  $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
1450  001254  000000                  $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
1451  001256  000254                  $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
1452  001260  000000                  $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
1453  001262  176700                  $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
1454  001264  000000                  $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
1455  001266  000000                  $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
1456  001270  000000                  $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
1457  001272                          $ETEND:
1458                                  .MEXIT
1459          000015                  CR      =       15
1460          000012                  LF      =       12
1461  001272  176700                  $RMADR: .WORD   176700          ;RH11/RM03 UNIBUS ADDRESS
1462  001274  000254                  $RMVEC: .WORD   254             ;RH11 INTERRUPT VECTOR
1463  001276  172540                  $LKCSR: .WORD   172540          ;ADDRESS OF KW11-P CSR
1464  001300  172542                  $LKCSB: .WORD   172542          ;ADDRESS OF KW11-P COUNTER BUFFER
1465  001302  000104  000106         $LPVEC: .WORD   104,106         ;ADDRESS OF KW11-P VECTOR
1466  001306  177546                  $LKS:   .WORD   177546          ;ADDRESS OF KW11-L CONTROL REGISTER
1467  001310  000100  000102         $LLVEC: .WORD   100,102         ;ADDRESS OF KW11-L VECTOR
1468          001220                  DRIVE   =$UNIT                  ;CONTAINS DRIVE NUMBER SELECTED
1469                                                                  ;SAME PARAMETER USED IN APT
1470  001314  000000                  SOFSW:  .WORD   0               ;CONTENTS ARE FOR SOFTWARE DECISIONS
1471  001316  000000                  MODE:   .WORD   0               ;'FORMAT & VERIFY' OR 'CHECK' MODE INDICATOR
1472  001320  001466                  ENDCYL: .WORD   822.            ;ENDING CYLINDER
1473  001322  000000                  BEGCYL: .WORD   0               ;STARTING CYLINDER
```

```
1474  001324  000004                ENDTRK: .WORD   4.              ;ENDING TRACK
1475  001326  000000                BEGTRK: .WORD   0               ;STARTING TRACK
1476  001330  000000                TTRKS:  .WORD   0               ;TOTAL # OF TRACKS TO BE FORMATTED
1477  001332  000000                TTRKSC: .WORD   0               ;TOTAL # OF TRACKS COUNTER
1478  001334  000000                TRKCNT: .WORD   0               ;COUNTS TRKS FROM 0-5 PER CYL
1479  001336  000000                PATSEL: .WORD   0               ;CONTAINS PATTERN SELECTED
1480  001340  000000                PATA:   .WORD   0               ;1ST WORD OF PATTERN
1481  001342  000000                PATB:   .WORD   0               ;2ND WORD OF PATTERN
1482  001344  000000                CYLCK:  .WORD   0               ;STORE CYLINDER ADDRESS FOR FORMAT VERIFICATION
1483  001346  000000                RETRY:  .WORD   0               ;MAINTAINS # OF WRITE RETRIES MADE
1484  001350  000000                SAVSEC: .WORD   0               ;CONTAINS LAST BAD SECTOR ON FORMAT
1485  001352  000000                SAVWC:  .WORD   0               ;CONTAINS WC FOR REMAINING SECTORS ON ERROR
1486  001354  000000                WC:     .WORD   0               ;30 OR 32 SECTOR TRACK SIZE (IN WORDS)
1487  001356  000000                MWC:    .WORD   0               ;2'S COMPLEMENT OF 'WC'
1488  001360  000000                HEDERR: .WORD   0               ;POSITIONING ERROR DURING FORMAT INDICATOR
1489  001362  000000                SEC30:  .WORD   0               ;30 OR 32 SECTOR MODE INDICATOR
1490                                                                ;0 = 30 SECTOR MODE
1491                                                                ;1'S = 32 SECTOR MODE
1492  001364  000000                MAXSEC: .WORD   0               ;MAXIMUM SECTOR ADDRESS (FOR EITHER 30 OR 32 SECTOR
1493                                                                ;FORMAT)
1494  001366  000000                DS.CYL: .WORD   0               ;ADDRESS OF CURRENT CYLINDER
1495  001370  000000                DS.TRK: .WORD   0               ;ADDRESS OF CURRENT TRACK
1496  001372  000000                DDRIVE: .WORD   0               ;DRIVE NUMBER OF 'DRIVER' ERROR MESSAGES
1497  001374  000000                ATTN:   .WORD   0               ;ATTENTION REGISTER IMAGE FOR 'DRIVER'
1498                                                                ;ERROR MESSAGES
1499  001376  000000                CNTLC:  .WORD   0               ;ADDRESS OF '↑C' RETURN
1500  001400  000000                CHGADR: .WORD   0               ;'CHANGE RH11 ADDRESS' FLAG
1501  001402  000000                WRAP:   .WORD   0       ;WRAP AROUND FLAG,FORMAT FROM HIGH TO LOW
1502                                                                ;CYLINDER NUMBER
1503  001404  000000                DEVMP:  .WORD   0               ;DEVICE MAP FOR APT
1504  001406  000000                PACK:   .WORD   0       ;PACK>0, BAD SPOT FILE AVAILABLE FROM PACK
1505  001410  000000                EDIT:   .WORD   0               ;EDIT THE BAD SPOT FILE ,IF EDIT>0
1506  001412  000000                HEADX:  .WORD   0               ;HEAD BIT INDICATOR
1507                                                                ;BIT15!BIT14  GOOD SECTOR
1508                                                                ;BIT15 ALONE, USER DEFINED BAD SECTOR
1509                                                                ;BIT14 ALONE, MFG DEFINED BAD SECTOR
1510  001414  000400                BAD16:  .BLKW   256.            ;MFG 16 BIT BAD SPOT TABLE
1511  002414  177777                        .WORD   -1
1512  002416  177777                        .WORD   -1
1513  002420  000400                BAD.8:  .BLKW   256.            ;MFG 18 BIT BAD SPOT TABLE
1514  003420  177777                        .WORD   -1
1515  003422  177777                        .WORD   -1
1516  003424  000400                USTAB:  .BLKW   256.            ;USER BAD SPOT TABLE
1517  004424  177777                        .WORD   -1
1518  004426  177777                        .WORD   -1
1519  004430  000400                USSAV:  .BLKW   256.            ;USER BAD SPOT TABLE SAVE FROM PACK
1520  005430  177777                        .WORD   -1
1521  005432  177777                        .WORD   -1
1522  005434  000040                SECCU:  .BLKW   32.     ;32 WORDS MAX TO STORE BAD SECTORS
1523  005534  177777                        .WORD   -1
1524  005536  000000                NEXT:   .WORD   0               ;FLAG, IF NEXT=-1,ALL GOOD SPOT ON
1525                                                                ;THE CURRENT TRACK ARE FORMATTED
1526
1527                                ;RH11/RM03 REGISTERS STORED HERE AFTER AN OPERATION
1528
1529  005540  000000                RM.REG: .WORD   0               ;RMCS1
```

```
1530  005542  000000                  .WORD   0               ;RMWC
1531  005544  000000                  .WORD   0               ;RMBA
1532  005546  000000                  .WORD   0               ;RMDA
1533  005550  000000                  .WORD   0               ;RMCS2
1534  005552  000000                  .WORD   0               ;RMDS
1535  005554  000000                  .WORD   0               ;RMER1
1536  005556  000000                  .WORD   0               ;RMAS
1537  005560  000000                  .WORD   0               ;RMLA
1538  005562  000000                  .WORD   0               ;RMDB
1539  005564  000000                  .WORD   0               ;RMMR1
1540  005566  000000                  .WORD   0               ;RMDT
1541  005570  000000                  .WORD   0               ;RMSN
1542  005572  000000                  .WORD   0               ;RMOF
1543  005574  000000                  .WORD   0               ;RMCA
1544  005576  000000                  .WORD   0               ;RMDC
1545  005600  000000                  .WORD   0               ;RMER2
1546  005602  000000                  .WORD   0               ;RMMR2
1547  005604  000000                  .WORD   0               ;RMEC1
1548  005606  000000                  .WORD   0               ;RMEC2
1549
1550
1551                          ;DATA/PARAMETER BLOCK - USED FOR ALL DRIVE OPERATION
1552
1553  005610      000         FMTDPB: .BYTE   0               ;DRIVE NUMBER
1554  005611      000                 .BYTE   0               ;OFFSET VALUE OR FMT22,ECI, AND HCI
1555  005612      000                 .BYTE   0               ;COMMAND
1556  005613      000                 .BYTE   0               ;PSEL AND A17 AND A16
1557  005614  000000                  .WORD   0               ;WORD COUNT (NEG)
1558  005616  000000                  .WORD   0               ;BUFFER ADDRESS
1559  005620      000                 .BYTE   0               ;SECTOR ADDRESS
1560  005621      000                 .BYTE   0               ;TRACK ADDRESS
1561  005622  000000                  .WORD   0               ;CYLINDER ADDRESS
1562  005624  005540                  .WORD   RM.REG          ;ERROR TABLE POINTER
1563  005626  000000                  .WORD   0               ;STATUS-ERROR INDICATOR
1564                                                          ;BIT 15 = 1: ERROR OCCURRED
1565                                                          ;BIT 07 = 1: DONE
1566                                                          ;BIT 14-10 AND BIT 06-03
1567                                                          ;INDICATE TYPE OF ERROR
1568
1569
1570  00563C      000         FMTX:   .BYTE   0       ;SPECIAL DPB TO WRITE HEADER OF BAD SPOT
1571  005631      000                 .BYTE   0
1572  005632      000                 .BYTE   0
1573  005633      000                 .BYTE   0
1574  005634  177776                  .WORD   -2      ;ONLY TWO WORDS FOR HEADER
1575  005636  043736                  .WORD   RBUF    ;BUFFER ADDRESS
1576  005640      000                 .BYTE   0       ;SECTOR
1577  005641      000                 .BYTE   0       ;TRACK
1578  005642  000000                  .WORD   0       ;CYLINDER ADDRESS
1579  005644  000000                  .WORD   0       ;ERROR TABLE
1580                                                  ;DONT"SAVE ANYTHING
1581  005646  000000                  .WORD   0       ;STATUS WORD
1582                          ;PARAMETER POINTER TABLE
1583
1584  005650  005776  001467  001322  TABLE:  PAR1,823.,BEGCYL
1585  005656  006011  000005  001326          PAR2,5.,BEGTRK
```

```
1586  005664  006024  001467  001320          PAR3,823.,ENDCYL
1587  005672  006035  000005  001324          PAR4,5.,ENDTRK,0
1588  005700  000000
1589
1590  005702  006046  001467  001366  TABLE2: PAR5,823.,DS.CYL
1591  005710  006057  000005  001370          PAR6,5.,DS.TRK
1592  005716  006070  000040  001350          PAR7,32.,SAVSEC,0
1593  005724  000000
1594
1595  005726  006046  001467  001366  TABLE3: PAR5,823.,DS.CYL
1596  005734  006057  000005  001370          PAR6,5.,DS.TRK
1597  005742  006070  000036  001350          PAR7,30.,SAVSEC,0
1598  005750  000000
1599  005752  006046  001467  001366  TABLE4: PAR5,823.,DS.CYL
1600  005760  006057  000005  001370          PAR6,5,DS.TRK
1601  005766  006101  047301  001350          PAR8,20161.,SAVSEC,0
1602  005774  000000
1603                                  ;ASCII MESSAGES FOR ADDRESS PARAMETERS
1604
1605  005776  052123  051101  020124  PAR1:   .ASCIZ  @START CYL @
1606  006004  054503  020114    000
1607  006011     123  040524  052122  PAR2:   .ASCIZ  @START TRK @
1608  006016  052040  045522  0000 J
1609  006024  047105  020104  054503  PAR3:   .ASCIZ  @END CYL @
1610  006032  020114    000
1611  006035     105  042116  052040  PAR4:   .ASCIZ  @END TRK @
1612  006042  045522  000040
1613
1614  006046  054503  044514  042116  PAR5:   .ASCIZ  /CYLINDER/
1615  006054  051105    000
1616  006057     124  040522  045503  PAR6:   .ASCIZ  /TRACK   /
1617  006064  020040  000040
1618  006070  042523  052103  051117  PAR7:   .ASCIZ  /SECTOR  /
1619  006076  020040    000
1620  006101     102  052131  020105  PAR8:   .ASCIZ  /BYTE POSITION /
1621  006106  047520  044523  044524
1622  006114  047117  000040
1623                                  .EVEN
1624  006120  021044  021324  021412  TABCAL: .WORD   FUNCT2,FUNCT3,FUNCT4,FUNCT6,FUNCT7
1625  006126  021620  021716
1626                                  ;SECTOR BUFFER ADDRESS TABLE
1627
1628  006132  043746                  ADRTBL: .WORD   BUFP                ;ADDRESS OF SECTOR 0
1629  006134  044752                          .WORD   BUFP+<516.*1.>      ;ADDRESS OF SECTOR 1
1630  006136  045756                          .WORD   BUFP+<516.*2.>      ;ADDRESS OF SECTOR 2
1631  006140  046762                          .WORD   BUFP+<516.*3.>      ;ADDRESS OF SECTOR 3
1632  006142  047766                          .WORD   BUFP+<516.*4.>      ;ADDRESS OF SECTOR 4
1633  006144  050772                          .WORD   BUFP+<516.*5.>      ;ADDRESS OF SECTOR 5
1634  006146  051776                          .WORD   BUFP+<516.*6.>      ;ADDRESS OF SECTOR 6
1635  006150  053002                          .WORD   BUFP+<516.*7.>      ;ADDRESS OF SECTOR 7
1636  006152  054006                          .WORD   BUFP+<516.*8.>      ;ADDRESS OF SECTOR 8
1637  006154  055012                          .WORD   BUFP+<516.*9.>      ;ADDRESS OF SECTOR 9
1638  006156  056016                          .WORD   BUFP+<516.*10.>     ;ADDRESS OF SECTOR 10
1639  006160  057022                          .WORD   BUFP+<516.*11.>     ;ADDRESS OF SECTOR 11
1640  006162  060026                          .WORD   BUFP+<516.*12.>     ;ADDRESS OF SECTOR 12
1641  006164  061032                          .WORD   BUFP+<516.*13.>     ;ADDRESS OF SECTOR 13
```

```
1642  006166  062036                  .WORD   BUFP+<516.*14.>  ;ADDRESS OF SECTOR 14
1643  006170  063042                  .WORD   BUFP+<516.*15.>  ;ADDRESS OF SECTOR 15
1644  006172  064046                  .WOPD   BUFP+<516.*16.>  ;ADDRESS OF SECTOR 16
1645  006174  065052                  .WORD   BUFP+<516.*17.>  ;ADDRESS OF SECTOR 17
1646  006176  066056                  .WORD   BUFP+<516.*18.>  ;ADDRESS OF SECTOR 18
1647  006200  067062                  .WORD   BUFP+<516.*19.>  ;ADDRESS OF SECTOR 19
1648  006202  070066                  .WORD   BUFP+<516.*20.>  ;ADDRESS OF SECTOR 20
1649  006204  071072                  .WORD   BUFP+<516.*21.>  ;ADDRESS OF SECTOR 21
1650  006206  072076                  .WORD   BUFP+<516.*22.>  ;ADDRESS OF SECTOR 22
1651  006210  073102                  .WORD   BUFP+<516.*23.>  ;ADDRESS OF SECTOR 23
1652  006212  074106                  .WORD   BUFP+<516.*24.>  ;ADDRESS OF SECTOR 24
1653  006214  075112                  .WORD   BUFP+<516.*25.>  ;ADDRESS OF SECTOR 25
1654  006216  076116                  .WORD   BUFP+<516.*26.>  ;ADDRESS OF SECTOR 26
1655  006220  077122                  .WORD   BUFP+<516.*27.>  ;ADDRESS OF SECTOR 27
1656  006222  100126                  .WORD   BUFP+<516.*28.>  ;ADDRESS OF SECTOR 28
1657  006224  101132                  .WORD   BUFP+<516.*29.>  ;ADDRESS OF SECTOR 29
1658  006226  102136                  .WORD   BUFP+<516.*30.>  ;ADDRESS OF SECTOR 30
1659  006230  103142                  .WORD   BUFP+<516.*31.>  ;ADDRESS OF SECTOR 31
1660
1661                          ;REMAINING WORD COUNT TABLE
1662
1663  006232  000402          WCTBL:  .WORD   258.             ;REMAINING WORD COUNT AFTER SECTOR 0
1664  006234  001004                  .WORD   258.+<258.*1.>   ;REMAINING WORD COUNT AFTER SECTOR 1
1665  006236  001406                  .WORD   258.+<258.*2.>   ;REMAINING WORD COUNT AFTER SECTOR 2
1666  006240  002010                  .WORD   258.+<258.*3.>   ;REMAINING WORD COUNT AFTER SECTOR 3
1667  006242  002412                  .WORD   258.+<258.*4.>   ;REMAINING WORD COUNT AFTER SECTOR 4
1668  006244  003014                  .WORD   258.+<258.*5.>   ;REMAINING WORD COUNT AFTER SECTOR 5
1669  006246  003416                  .WORD   258.+<258.*6.>   ;REMAINING WORD COUNT AFTER SECTOR 6
1670  006250  004020                  .WORD   258.+<258.*7.>   ;REMAINING WORD COUNT AFTER SECTOR 7
1671  006252  004422                  .WORD   258.+<258.*8.>   ;REMAINING WORD COUNT AFTER SECTOR 8
1672  006254  005024                  .WORD   258.+<258.*9.>   ;REMAINING WORD COUNT AFTER SECTOR 9
1673  006256  005426                  .WORD   258.+<258.*10.>  ;REMAINING WORD COUNT AFTER SECTOR 10
1674  006260  006030                  .WORD   258.+<258.*11.>  ;REMAINING WORD COUNT AFTER SECTOR 11
1675  006262  006432                  .WORD   258.+<258.*12.>  ;REMAINING WORD COUNT AFTER SECTOR 12
1676  006264  007034                  .WORD   258.+<258.*13.>  ;REMAINING WORD COUNT AFTER SECTOR 13
1677  006266  007436                  .WORD   258.+<258.*14.>  ;REMAINING WORD COUNT AFTER SECTOR 14
1678  006270  010040                  .WORD   258.+<258.*15.>  ;REMAINING WORD COUNT AFTER SECTOR 15
1679  006272  010442                  .WORD   258.+<258.*16.>  ;REMAINING WORD COUNT AFTER SECTOR 16
1680  006274  011044                  .WORD   258.+<258.*17.>  ;REMAINING WORD COUNT AFTER SECTOR 17
1681  006276  011446                  .WORD   258.+<258.*18.>  ;REMAINING WORD COUNT AFTER SECTOR 18
1682  006300  012050                  .WORD   258.+<258.*19.>  ;REMAINING WORD COUNT AFTER SECTOR 19
1683  006302  012452                  .WORD   258.+<258.*20.>  ;REMAINING WORD COUNT AFTER SECTOR 20
1684  006304  013054                  .WORD   258.+<258.*21.>  ;REMAINING WORD COUNT AFTER SECTOR 21
1685  006306  013456                  .WORD   258.+<258.*22.>  ;REMAINING WORD COUNT AFTER SECTOR 22
1686  006310  014060                  .WORD   258.+<258.*23.>  ;REMAINING WORD COUNT AFTER SECTOR 23
1687  006312  014462                  .WORD   258.+<258.*24.>  ;REMAINING WORD COUNT AFTER SECTOR 24
1688  006314  015064                  .WORD   258.+<258.*25.>  ;REMAINING WORD COUNT AFTER SECTOR 25
1689  006316  015466                  .WORD   258.+<258.*26.>  ;REMAINING WORD COUNT AFTER SECTOR 26
1690  006320  016070                  .WORD   258.+<258.*27.>  ;REMAINING WORD COUNT AFTER SECTOR 27
1691  006322  016472                  .WORD   258.+<258.*28.>  ;REMAINING WORD COUNT AFTER SECTOR 28
1692  006324  017074                  .WORD   258.+<258.*29.>  ;REMAINING WORD COUNT AFTER SECTOR 29
1693  006326  017476                  .WORD   258.+<258.*30.>  ;REMAINING WORD COUNT AFTER SECTOR 30
1694  006330  020100                  .WORD   258.+<258.*31.>  ;REMAINING WORD COUNT AFTER SECTOR 31
1695  006332  001166          BYTE16: .WORD   630.
1696  006334  002354                  .WORD   630.+<630.*1.>
1697  006336  003542                  .WORD   630.+<630.*2.>
```

```
1698  006340  004730                          .WORD   630.+<630.*3.>
1699  006342  006116                          .WORD   630.+<630.*4.>
1700  006344  007304                          .WORD   630.+<630.*5.>
1701  006346  010472                          .WORD   630.+<630.*6.>
1702  006350  011660                          .WORD   630.+<630.*7.>
1703  006352  013046                          .WORD   630.+<630.*8.>
1704  006354  014234                          .WORD   630.+<630.*9.>
1705  006356  015422                          .WORD   630.+<630.*10.>
1706  006360  016610                          .WORD   630.+<630.*11.>
1707  006362  017776                          .WORD   630.+<630.*12.>
1708  006364  021164                          .WORD   630.+<630.*13.>
1709  006366  022352                          .WORD   630.+<630.*14.>
1710  006370  023540                          .WORD   630.+<630.*15.>
1711  006372  024726                          .WORD   630.+<630.*16.>
1712  006374  026114                          .WORD   630.+<630.*17.>
1713  006376  027302                          .WORD   630.+<630.*18.>
1714  006400  030470                          .WORD   630.+<630.*19.>
1715  006402  031656                          .WORD   630.+<630.*20.>
1716  006404  033044                          .WORD   630.+<630.*21.>
1717  006406  034232                          .WORD   630.+<630.*22.>
1718  006410  035420                          .WORD   630.+<630.*23.>
1719  006412  036606                          .WORD   630.+<630.*24.>
1720  006414  037774                          .WORD   630.+<630.*25.>
1721  006416  041162                          .WORD   630.+<630.*26.>
1722  006420  042350                          .WORD   630.+<630.*27.>
1723  006422  043536                          .WORD   630.+<630.*28.>
1724  006424  044724                          .WORD   630.+<630.*29.>
1725  006426  046112                          .WORD   630.+<630.*30.>
1726  006430  047300                          .WORD   630.+<630.*31.>
1727  006432  001240          BYTE18:         .WORD   672.
1728  006434  002500                          .WORD   672.+<672.*1.>
1729  006436  003740                          .WORD   672.+<672.*2.>
1730  006440  005200                          .WORD   672.+<672.*3.>
1731  006442  006440                          .WORD   672.+<672.*4.>
1732  006444  007700                          .WORD   672.+<672.*5.>
1733  006446  011140                          .WORD   672.+<672.*6.>
1734  006450  012400                          .WORD   672.+<672.*7.>
1735  006452  013640                          .WORD   672.+<672.*8.>
1736  006454  015100                          .WORD   672.+<672.*9.>
1737  006456  016340                          .WORD   672.+<672.*10.>
1738  006460  017600                          .WORD   672.+<672.*11.>
1739  006462  021040                          .WORD   672.+<672.*12.>
1740  006464  022300                          .WORD   672.+<672.*13.>
1741  006466  023540                          .WORD   672.+<672.*14.>
1742  006470  025000                          .WORD   672.+<672.*15.>
1743  006472  026240                          .WORD   672.+<672.*16.>
1744  006474  027500                          .WORD   672.+<672.*17.>
1745  006476  030740                          .WORD   672.+<672.*18.>
1746  006500  032200                          .WORD   672.+<672.*19.>
1747  006502  033440                          .WORD   672.+<672.*20.>
1748  006504  034700                          .WORD   672.+<672.*21.>
1749  006506  036140                          .WORD   672.+<672.*22.>
1750  006510  037400                          .WORD   672.+<672.*23.>
1751  006512  040640                          .WORD   672.+<672.*24.>
1752  006514  042100                          .WORD   672.+<672.*25.>
1753  006516  043340                          .WORD   672.+<672.*26.>
```

```
1754  006520  044600                          .WORD   672.+<672.*27.>
1755  006522  046040                          .WORD   672.+<672.*28.>
1756  006524  047300                          .WORD   672.+<672.*29.>
1757
1758                                  .EVEN
1759
```

```
                        .SBTTL  ERROR POINTER TABLE

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

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


006526                  $ERRTB:
                        ;ERROR 1

006526  040474                 EM1                     ;RH11 INTERRUPT OCCURRED (RMAS=0)
006530  041662                 DH1
006532  043256                 DT1
006534  043622                 DF1

                        ;ERROR 2

006536  040535                 EM2                     ;UNEXPECTED ATTENTION OCCURRED
006540  041730                 DH2
006542  043274                 DT2
006544  043626                 DF2

                        ;ERROR 3

006546  040573                 EM3                     ;MASSBUS PARITY ERROR (MCPE=1)
006550  042002                 DH3
006552  043310                 DT3
006554  043632                 DF3

                        ;ERROR 4

006556  040631                 EM4                     ;MASSBUS PARITY ERROR (PAR=1)
006560  042053                 DH4
006562  043324                 DT4
006564  043636                 DF4

                        ;ERROR 5

006566  000000                 0                       ;UNUSED
006570  000000                 0
006572  000000                 0
006574  000000                 0

                        ;ERROR 6

006576  040722                 EM6                     ;RH11 DIDN'T RESPOND TO ADDRESSING
006600  042136                 DH6
006602  043342                 DT6
006604  043642                 DF6
```

```
1816
1817                                    ;ERROR 7
1818
1819  006606  000000                            0                       ;UNUSED
1820  006610  000000                            0
1821  006612  000000                            0
1822  006614  000000                            0
1823
1824                                    ;ERROR 10
1825
1826  006616  040764                            EM10                    ;DRIVE OFFLINE
1827  006620  042151                            DH10
1828  006622  043344                            DT10
1829  006624  043646                            DF10
1830
1831                                    ;ERROR 11
1832
1833  006626  041002                            EM11                    ;PERSISTENT DRIVE UNSAFE ERROR
1834  006630  042151                            DH10
1835  006632  043344                            DT10
1836  006634  043646                            DF10
1837
1838                                    ;ERROR 12
1839
1840  006636  041040                            EM12                    ;UNCORRECTABLE MASSBUS PARITY ERROR
1841  006640  042151                            DH10
1842  006642  043344                            DT10
1843  006644  043646                            DF10
1844
1845                                    ;ERROR 13
1846
1847  006646  041103                            EM13                    ;SOFTWARE TIMEOUT
1848  006650  042151                            DH10
1849  006652  043344                            DT10
1850  006654  043646                            DF10
1851
1852                                    ;ERROR 14
1853
1854  006656  041124                            EM14                    ;DRIVE UNSAFE ERROR
1855  006660  042151                            DH10
1856  006662  043344                            DT10
1857  006664  043646                            DF10
1858
1859                                    ;ERROR 15
1860
1861  006666  041147                            EM15                    ;CONTROLLER/DRIVE ERROR DURING WRITE
1862  006670  042151                            DH10
1863  006672  043344                            DT10
1864  006674  043646                            DF10
1865
1866                                    ;ERROR 16
1867
1868  006676  041213                            EM16                    ;CONTROLLER/DRIVE ERROR DURING WRITE CHECK
1869  006700  042151                            DH10
1870  006702  043344                            DT10
1871  006704  043646                            DF10
```

```
1872
1873                                    ;ERROR 17
1874
1875  006706  041265                            EM17                    ;RETRIES NOT SUCESSFUL - SECTOR NOT ACCEPTABLE
1876  006710  042400                            DH17
1877  006712  043416                            DT17
1878  006714  043662                            DF17
1879
1880                                    ;ERROR 20
1881
1882  006716  041343                            EM20                    ;DATA ERROR DURING WRITE CHECK
1883  006720  042447                            DH20
1884  006722  043430                            DT20
1885  006724  043666                            DF20
1886
1887                                    ;ERROR 21
1888
1889  006726  041401                            EM21                    ;CONTROLLER/DRIVE ERROR VERIFYING HEADERS
1890  006730  042151                            DH10
1891  006732  043344                            DT10
1892  006734  043646                            DF10
1893
1894                                    ;ERROR 22
1895
1896  006736  041452                            EM22                    ;'HCE' ERROR VERIFYING HEADERS
1897  006740  042151                            DH10
1898  006742  043344                            DT10
1899  006744  043646                            DF10
1900
1901                                    ;ERROR 23
1902
1903  006746  041521                            EM23                    ;CYLINDER FIELD IN HEADER IS NOT CORRECT
1904  006750  042744                            DH23
1905  006752  043512                            DT23
1906  006754  043706                            DF23
1907
1908                                    ;ERROR 24
1909
1910  006756  041571                            EM24                    ;WRITE CHECK ERROR
1911  006760  043042                            DH24
1912  006762  043524                            DT24
1913  006764  043712                            DF24
1914
1915                                    ;ERROR 25
1916
1917  006766  041613                            EM25                    ;ILLEGAL BAD SECTOR
1918  006770  043215                            DH25
1919  006772  043612                            DT25
1920  006774  043732                            DF25
1921
1922                                    ;;*****************************************************************
1923
1924                                    .SBTTL  MAIN PROGRAM
1925
1926                                    ;;*****************************************************************
1927
```

```
1928  006776  012737  177777  001400  BEGIN:  MOV     #-1,CHGADR      ;SET CHANGE 'RH70 BUS ADDRESS' INDICATOR
1929  007004  012737  177777  001266          MOV     #-1,$CDW1       ;FLAG TO USE UTILITY ROUTINES
1930  007012  000410                          BR      BEGIN2          ;START THE PROGRAM
1931  007014  005037  001400          BEGIN1: CLR     CHGADR          ;CLEAR THE 'CHANGE RH70 ADDRESS' INDICATOR
1932  007020  005037  001266                  CLR     $CDW1           ;REST UTILITY ROUTINE FLAG
1933                                          ;;*****************************************************************
1934  007024  000240                  TST1:   NOP
1935  007026  012737  000001  001212          MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
1936  007034  000005          BEGIN2: RESET           ;CLEAR THE BUS
1937                                  .SBTTL  INITIALIZE THE COMMON TAGS
1938                                  ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
1939  007036  012706  001114          MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
1940  007042  005026                  CLR     (R6)+           ;;CLEAR MEMORY LOCATION
1941  007044  022706  001154          CMP     #SWR,R6 ;;DONE?
1942  007050  001374                  BNE     .-6             ;;LOOP BACK IF NO
1943  007052  012706  001100          MOV     #STACK,SP       ;;SETUP THE STACK POINTER
1944                                  ;;INITIALIZE A FEW VECTORS
1945  007056  012737  022516  000030  MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
1946  007064  012737  000340  000032  MOV     #340,@#EMTVEC+2 ;;LEVEL 7
1947  007072  012737  026412  000034  MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1948  007100  012737  000340  000036  MOV     #340,@#TRAPVEC+2;LEVEL 7
1949  007106  012737  025654  000024  MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
1950  007114  012737  000340  000026  MOV     #340,@#PWRVEC+2 ;;LEVEL 7
1951  007122  005037  001174          CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
1952  007126  112737  000001  001131  MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
1953                                  ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1954                                  ;;EQUAL TO A "-1"  SETUP FOR A SOFTWARE SWITCH REGISTER.
1955  007134  013746  000004          MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
1956  007140  012737  007174  000004  MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
1957  007146  012737  177570  001154  MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
1958  007154  012737  177570  001156  MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
1959  007162  022777  177777  171764  CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
1960  007170  001012                  BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1961                                                          ;;AND  THE HARDWARE SWR IS NOT = -1
1962  007172  000403                  BR      65$             ;;BRANCH IF NO TIMEOUT
1963  007174  012716  007202  64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
1964  007200  000002                  RTI
1965  007202  012737  000176  001154  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
1966  007210  012737  000174  001156          MOV     #DISPREG,DISPLAY
1967  007216  012637  000004  66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
1968
1969  007222  005037  001214          CLR     $PASS           ;;CLEAR PASS COUNT
1970  007226  132737  000200  001227  BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
1971  007234  001403                  BEQ     67$             ;;YES,USE NON-APT SWITCH
1972  007236  012737  001230  001154  MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
1973  007244          67$:
1974  007244  000005                  RESET                   ;CLEAR WORLD
1975  007246  012737  000240  000036  MOV     #PRS,@#TRAPVEC+2  ;CHANGE TRAP PRIORITY BACK TO 5
1976  007254  012737  000240  000032  MOV     #PRS,@#EMTVEC+2 ;CHANGE EMT PRIORITY BACK TO 5
1977  007262  012737  007014  001376  MOV     #BEGIN1,CNTLC   ;'CONTROL C' ADDRESS
1978  007270  005227  177777          INC     #-1             ;FIRST START ?
1979  007274  001011                  BNE     :$              ;BR IF NOT
1980  007276  104401  043746          TYPE    ,$TITLE         ;ADRS OF 'TITLE' MESSAGE
1981  007302  122737  000077  000004.  CMPB    @#4,#'?         ;LOAD FROM RM03/RM02 ?
1982  007310  001003                  BNE     .$              ;BRANCH IF NOT
1983  007312  104401  044046          TYPE    ,$CHANGE        ;TYPE MESSAGE TO CHANGE PACK
```

```
                                                    HALT                    ;THEN WAIT FOR RESTART
                                        1$:         JSR     PC,$TKINT       ;TURN ON THE KEYBOARD INTERRUPT
                                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
                                                    TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
                                                    BNE     68$             ;;BRANCH IF YES
                                                    CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
                                                    BEQ     68$             ;;BRANCH IF YES
                                                    CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
                                                    BNE     69$             ;;BRANCH IF NO
                                                    GTSWR                   ;;GET SOFT-SWR SETTINGS
                                                    BR      69$
                                        68$:        MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
                                        69$:
                                                    INC     #-1             ;CHECK FOR FIRST START
                                                    BNE     SETVEC          ;BR IF NOT
                                                    JSR     PC,BUSADR       ;CHECK THE RH11 ADDRESS
                                                    MOV     $RMADR,RMADR    ;RH11 ADDRESS
                                                    MOV     $RMVEC,RMVEC    ;RH11 VECTOR ADDRESS
                                                    TSTB    $ENV            ;RUN UNDER APT MODE?
                                                    BEQ     SETVEC          ;NO,DONOT LOAD FROM E-TABLE
                                                    TSTB    $VECT1          ;UUT INTERRUPT VECTOR
                                                    BEQ     .+10            ;NOT CHANGE IF = 0
                                                    MOVB    $VECT1,$RMVEC   ;NEWVALUE
                                                    TST     $BASE           ;UUT BASE REGISTER ADDRESS
                                                    BEQ     .+10            ;NOT CHANGE IF = 0
                                                    MOV     $BASE,$RMADR    ;NEW BASE ADDRESS
                                                    MOV     $RMADR,RMADR    ;RH11/70 ADDRESS
                                                    MOV     $RMVEC,RMVEC    ;RH11/70 VECTOR

                                        ;DISPLAY DRIVE STATUS AND SET UP THE OTHER UNITS THE
                                        ;       PROGRAM WILL USE

                                        SETVEC: JSR     PC,ST.CLK       ;START THE CLOCK
                                                JSR     PC,RMINIT       ;INITIALIZE THE RM03 DRIVER
                                                MOV     #-1,SAVEFG      ;SET THE SAVE REGISTERS FLAG
                                                INC     #-1             ;SEE IF FIRST START
                                                BEQ     11$             ;BR IF YES
                                                BIT     #SW02,@SWR      ;TYPEOUT THE DRIVE STATUS TABLE ?
                                                BNE     10$             ;BR IF NOT
                                        11$:    MOV     #PR7,PS         ;SET PRIORITY TO 7
                                                CLR     R4              ;DRIVE TABLE POINTER
                                                TYPE    ,$CRLF          ;CR-LF
                                                TYPE    ,SYSTAT         ;TYPE STATUS HEADING
                                        1$:
                                                MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
                                                                        ;;TYPE DRIVE NUMBER
                                                TYPOS                   ;;GO TYPE--OCTAL ASCII
                                                .BYTE   2               ;;TYPE 2 DIGIT(S)
                                                .BYTE   0               ;;SUPPRESS LEADING ZEROS
                                                TYPE    ,LIN4SP         ;SPACES
                                                TSTB    DRVSTA(R4)      ;CHECK DRIVE'S STATUS
                                                BMI     4$              ;BR IF UNSAFE
                                                BNE     5$              ;BR IF ONLINE
                                                TSTB    DRVTYP(R4)      ;SEE IF OFFLINE OR NONEXISTENT
                                                BEQ     2$              ;BR IF NONEXISTENT
                                                BPL     3$              ;BR IF OFFLINE
```

```
2040  007572  104401  035033                    TYPE    ,NOTRM          ;DRIVE NOT AN RM03
2041  007576  000443                            BR      9$              ;CHECK NEXT DRIVE
2042  007600  104401  035055            2$:     TYPE    ,NOTPRS         ;DRIVE NOT PRESENT
2043  007604  000440                            BR      9$              ;CHECK NEXT DRIVE
2044  007606  104401  035012            3$:     TYPE    ,UNTOFF         ;DRIVE OFFLINE
2045  007612  000410                            BR      6$              ;PRINT DRIVE TYPE
2046  007614  104401  035072            4$:     TYPE    ,NOTSAF         ;DRIVE UNSAFE
2047  007620  000405                            BR      6$              ;PRINT DRIVE TYPE
2048  007622  104401  035023            5$:     TYPE    ,UNTON          ;DRIVE ONLINE
2049  007626  156437  026632  001404            BISB    @ABIT(R4),DEVMP         ;SET DEVICE MAP
2050  007634  104401  035166            6$:     TYPE    ,LINSP          ;SPACES
2051  007640  012737  035102  007704            MOV     #RM03B,8$       ;ADDRESS OF RP04 MESSAGE
2052  007646  132764  000001  026526            BITB    #BIT00,DRVTYP(R4)   ;RM03 ?
2053  007654  001012                            BNE     7$              ;BR IF YES
2054  007656  012737  035107  007704            MOV     #RP05,8$        ;ADDRESS OF RP05 MESSAGE
2055  007664  132764  000002  026526            BITB    #BIT01,DRVTYP(R4)   ;RP05 ?
2056  007672  001003                            BNE     7$              ;BR IF YES
2057  007674  012737  035114  007704            MOV     #RM03A,8$       ;ADDRESS OF RM03 MESSAGE
2058  007702  104401                    7$:     TYPE                    ;TYPE THE DRIVE TYPE MESSAGE
2059  007704  000000                    8$:     .WORD   0               ;MESSAGE ADDRESS HERE
2060  007706  104401  001203            9$:     TYPE    ,$CRLF          ;CR-LF
2061  007712  005204                            INC     R4              ;INCREMENT DRIVE NUMBER/TABLE POINTER
2062  007714  020427  000010                    CMP     R4,#8.          ;FINISHED ?
2063  007720  001307                            BNE     1$              ;BR IF NOT
2064  007722  104401  001203            10$:    TYPE    ,$CRLF          ;CR-LF
2065                                    ;
2066  007726  105737  001150            APT:    TSTB    $AUTOB          ;RUN UNDER APT ?
2067  007732  001544                            BEQ     M1              ;NO,DON', BOTHER
2068  007734  005037  001410                    CLR     EDIT    ;CLEAR THE TEXT LAST TRACK FLAG
2069  007740  005737  001264                    TST     $DEVM   ;DEVICE MAP AVAILABLE FROM APT MAIL BOX
2070  007744  001406                            BEQ     1$              ;BRANCH IF NONE
2071  007746  013737  001264  001404            MOV     $DEVM,DEVMP     ;LOAD IT FORM MAIL BOX
2072  007754  042737  177400  001404            BIC     #177400,DEVMP   ;ALLOW MAX 8 DRIVES
2073  007762  005737  001404            1$:     TST     DEVMP           ;CHECK DEVICE MAP
2074  007766  001002                            BNE     M5FX            ;BRANCH IF DRIVES ASSIGNED
2075  007770  000137  015324                    JMP     $EOP            ;OTHERWISE,EXIT
2076  007774  005037  001326            M5FX:   CLR     BEGTRK          ;ALWAY STARTS FROM 0 TRK
2077  010000  005037  001322                    CLR     BEGCYL          ;ALWAYS STARTS FROM 0 CYLINDER
2078  010004  012737  001466  001320            MOV     #822.,ENDCYL    ;LAST CYLINDER
2079  010012  012737  000004  001324            MOV     #4,ENDTRK       ;LAST TRACK
2080  010020  005037  001402                    CLR     WRAP            ;CLEAR WRAP AROUND FLAG
2081  010024  012737  177777  001316            MOV     #-1,MODE        ;SET ORMAT AND VERIFY MODE
2082  010032  005737  001266                    TST     $CDW1           ;APT PARAMETER = 0
2083  010036  001402                            BEQ     .+6             ;YES,DEFAULT TO -1
2084  010040  005037  001316                    CLR     MODE            ;NO,SET TO CHECK MODE
2085  010044  012737  177777  001362            MOV     #-1,SEC30       ;SET 32 SECTOR
2086  010052  005737  001270                    TST     $CDW2           ;APT PARAMETER =0
2087  010056  001012                            BNE     1$              ;NO,30 SECTOR
2088  010060  012737  020100  001354            MOV     #<258.*32.>,WC  ;COUNT OF WORDS
2089  010066  012737  157700  001356            MOV     #-<258.*32.>,MWC        ;WORD COUNT
2090  010074  012737  000037  001364            MOV     #31.,MAXSEC     ;MAX 32 SECTOR
2091  010102  000413                            BR      2$
2092  010104  012737  017074  001354  1$:       MOV     #<258.*30.>,WC  ;COUNT OF WORDS
2093  010112  012737  160704  001356            MOV     #-<258.*30.>,MWC        ;WORD COUNT
2094  010120  005037  001362                    CLR     SEC30           ;30 SECTOR
2095  010124  012737  000035  001364            MOV     #29.,MAXSEC     ;MAX 30 SECTOR
```

```
2096  010132  005737  001316          2$:     TST     MODE            ;CHECK MODE ?
2097  010136  001006                          BNE     .+16            ;NO
2098  010140  012737  133331  001340          MOV     #133331,PATA    ;INITIATE PATTERN
2099  010146  012737  133331  001342          MOV     #133331,PATB    ;INITIATE PATTERN
2100  010154  012737  000002  001336          MOV     #2,PATSEL       ;WOPSE CASE
2101  010162  012700  000001                  MOV     #1,R0           ;UUT STARTS AT DRIVE 0
2102  010166  005001                          CLR     R1              ;DRIVE NUMBER
2103  010170  030037  001404          3$:     BIT     R0,DEVMP        ;BIT OF DEVICE MAP SET ?
2104  010174  001005                          BNE     4$      ;FIND THE DRIVE # FROM DEVICE MAP
2105  010176  000241                          CLC
2106  010200  006100                          ROL     R0      ;NEXT DRIVE
2107  010202  003416                          BLE     6$      ;BRANCH IF ALL DRIVES ARE DONE
2108  010204  005201                          INC     R1              ;INCREMENT DRIVE #
2109  010206  000770                          BR      3$      ;LOOP
2110  010210  010137  001220          4$:     MOV     R1,DRIVE        ;LOAD R1 INTO DRIVE
2111  010214  040037  001404                  BIC     R0,DEVMP        ;CLEAR THE BIT FROM DEVICE MAP
2112  010220  105761  026516                  TSTB    DRVSTA(R1)      ;DRIVE ON LINE ?
2113  010224  003003                          BGT     5$      ;BRANCH IF DRIVE ON LINE
2114  010226  005237  001216                  INC     $DEVCT  ;INCREMENT DEVICE COUNT TO CHANGE MAIL BOX
2115  010232  000756                          BR      3$      ;LOOP BACK
2116  010234  000137  011160          5$:     JMP     M5      ;START TO FORMAT OR CHECK
2117  010240  000137  015324          6$:     JMP     $EOP    ;END OF PASS FOR ALL DRIVES ON $DEVM
2118
2119
2120                                  ;SEE WHICH MODE THE PROGRAM IS TO BE RUN IN.
2121                                  ;MODES ARE: 'FORMAT & VERIFY' OR 'CHECK FORMAT'
2122
2123  010244  005037  001316          M1:     CLR     MODE            ;SET MODE TO 'CHECK FORMAT'
2124  010250  104401  035376                  TYPE    ,MMODE          ;TYPE 'PROGRAM MODE'
2125  010254  104411                          RDLIN                   ;READ THE KEYBOARD
2126  010256  012601                          MOV     (SP)+,R1        ;GET ADDRESS OF INPUT
2127  010260  105711                          TSTB    (R1)            ;'CR' ENTERED (DEFAULT)
2128  010262  001414                          BEQ     2$              ;BR IF YES
2129  010264  122711  000103                  CMPB    #'C,(R1)        ;'CHECK' ?
2130  010270  001406                          BEQ     1$              ;BR IF YES
2131  010272  122711  000106                  CMPB    #'F,(R1)        ;'FORMAT' ?
2132  010276  001411                          BEQ     3$              ;BR IF YES
2133  010300  104401  001202                  TYPE    ,$QUES          ;NO CORRECT ENTRY
2134  010304  000757                          BR      M1              ;TRY AGAIN
2135  010306  104401  035451          1$:     TYPE    ,MHECK          ;TYPE REST OF 'CHECK'
2136  010312  000410                          BR      M1A             ;GET STARTING ADDRESS
2137  010314  104401  035430          2$:     TYPE    ,MFORMT         ;TYPE DEFAULT MESSAGE
2138  010320  000402                          BR      4$              ;SET UP MODE
2139  010322  104401  035464          3$:     TYPE    ,MORMAT         ;TYPE REST OF 'FORMAT'
2140  010326  012737  177777  001316  4$:     MOV     #-1,MODE        ;SET MODE TO 'FORMAT & VERIFY'
2141
2142                                  ;FIND OUT IF FORMAT IS TO BE IN 30 OR 32 SECTOR MODE
2143
2144  010334  104401  035504          M1A:    TYPE    ,MSIZE          ;TYPE FORMAT MODE REQUEST
2145  010340  104411                          RDLIN                   ;READ THE KEYBOARD
2146  010342  012601                          MOV     (SP)+,R1        ;ADDRESS OF ENTRY
2147  010344  122711  000131                  CMPB    #'Y,(R1)        ;IS ENTRY A 'Y' ?
2148  010350  001410                          BEQ     1$              ;BR IF IT IS
2149  010352  122711  000116                  CMPB    #'N,(R1)        ;IS ENTRY A 'N' ?
2150  010356  001424                          BEQ     2$              ;BR IF IT IS
2151  010360  105711                          TSTB    (R1)            ;DEFAULT MODE ?
```

```
2152  010362  001403                        BEQ     1$              ;BR IF IT IS
2153  010364  104401  001202                TYPE    ,$QUES          ;TYPE '?'
2154  010370  000761                        BR      M1A             ;TRY AGAIN
2155  010372  104401  035555        1$:     TYPE    ,MSEC22         ;TYPEOUT MODE SELECTED
2156  010376  012737  020100  001354        MOV     #<258.*32.>,WC  ;TRACK SIZE IN 32 SECTOR MODE
2157  010404  012737  157700  001356        MOV     #-<258.*32.>,MWC  ;2'S COMPLEMENT WORD COUNT
2158  010412  012737  177777  001362        MOV     #-1,SEC30       ;32 SECTOR INDICATOR
2159  010420  012737  000037  001364        MOV     #31.,MAXSEC     ;MAX SECTOR ADDRESS IN 32 SECTOR MODE
2160  010426  000415                        BR      M1B             ;CONTINUE
2161  010430  104401  035634        2$:     TYPE    ,MSEC20         ;TYPE OUT 30 SECTOR MODE SELECTED
2162  010434  012737  017074  001354        MOV     #<258.*30.>,WC  ;TRACK SIZE IN 30 SECTOR MODE
2163  010442  012737  160704  001356        MOV     #-<258.*30.>,MWC  ;2'S COMPLEMENT WORD COUNT
2164  010450  005037  001362                CLR     SEC30           ;30 SECTOR INDICATOR
2165  010454  012737  000035  001364        MOV     #29.,MAXSEC     ;MAX SECTOR ADDRESS IN 30 SECTOR MODE
2166  010462  005037  001326        M1B:    CLR     BEGTRK          ;CLEAR STARTING TRACK ADDRESS
2167  010466  005037  001322                CLR     BEGCYL          ;CLEAR BEGINNING CYLINDER ADDRESS
2168  010472  005037  001402                CLR     WRAP            ;CLEAR WRAP AROUND FLAG
2169  010476  012737  000004  001324        MOV     #4.,ENDTRK      ;SETUP END TRACK ADDRESS
2170  010504  012737  000002  001336        MOV     #2,PATSEL       ;SETUP FOR WORST CASE PATTERN
2171  010512  112737  177777  005610        MOVB    #-1,FMTDPB      ;SETUP INVALID DRIVE NUMBER
2172  010520  000400                        BR      M0              ;BRANCH TO START
2173
2174                                        ;GO FIND OUT WHAT DRIVE
2175
2176  010522  012737  017240  001376  M0:    MOV     #QENTER,CNTLC   ;CONTROL C ABORT ENTRANCE
2177  010530  005037  001126                CLR     $ERTTL          ;CLEAR THE ERROR ACCUMULATOR
2178  010534  004737  024422                JSR     PC,$TKINT       ;INITIALIZE THE TTY KEYBOARD
2179  010540  104401  035140                TYPE    ,MUNIT          ;ASK FOR DRIVE NUMBER
2180  010544  104411                        RDLIN                   ;READ THE KEYBOARD
2181  010546  012601                        MOV     (SP)+,R1        ;ADDRESS OF ENTRY
2182  010550  004537  022306                JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
2183  010554  010600                        3$                      ;ILLEGAL CHARACTER
2184  010556  010570                        2$                      ;CARRIAGE RETURN
2185  010560  010600                        3$                      ;".
2186  010562  010570                        2$                      ;.'.
2187  010564  010606                        4$                      ;DIGIT 0-7
2188  010566  010600                        3$                      ;DIGIT 8-9
2189  010570  005002                2$:     CLR     R2              ;SELECT DRIVE ZERO
2190  010572  104401  035162                TYPE    ,MDRVD          ;GO TYPE DEFAULT DRIVE NUMBER
2191  010576  000403                        BR      4$              ;GO SEE IF DRIVE ZERO IS THERE
2192  010600  104401  001202        3$:     TYPE    ,$QUES          ;TYPE '?'
2193  010604  000746                        BR      M0              ;ASK FOR DRIVE NUMBER AGAIN
2194  010606  010237  001220        4$:     MOV     R2,DRIVE        ;SAVE DRIVE NUMBER
2195  010612  005702                        TST     R2              ;SEE IF DRIVE 0
2196  010614  001004                        BNE     5$              ;BR IF NOT
2197  010616  122737  000011  000041        CMPB    #11,41          ;PROGRAM LOADED FROM AN RM03 ?
2198  010624  001431                        BEQ     7$              ;BR IF IT WAS, CAN'T FORMAT THE DRIVE
2199  010626  004737  017020        5$:     JSR     PC,ST.CLK       ;START THE CLOCK
2200  010632  004737  026662                JSR     PC,RMINIT       ;GO SEE WHAT DRIVES ARE AVAILABLE
2201  010636  012737  177777  026604        MOV     #-1,SAVEFG      ;SAVE THE REGISTERS
2202  010644  012737  177777  026606        MOV     #-1,SEEKFG      ;SET 'NO OPTIMIZATION' FLAG
2203  010652  005037  177776                CLR     PS              ;SET PRIORITY BACK TO ZERO
2204  010656  105762  026516                TSTB    DRVSTA(R2)      ;LOOK AT DRIVE STATUS
2205  010662  003015                        BGT     8$              ;BRANCH IF ONLINE
2206  010664  001403                        BEQ     6$              ;BR IF DRIVE NOT AVAILABLE
2207  010666  104401  035344                TYPE    ,MUSDR          ;'DRIVE UNSAFE'
```

```
2208  010672  000713                    BR      M0              ;GO GET DRIVE NUMBER AGAIN
2209  010674  105762  026526    6$:     TSTB    DRVTYP(R2)      ;A DRIVE PRESENT?
2210  010700  001003                    BNE     7$              ;BR IF SO
2211  010702  104401  035243            TYPE    ,MDRNP          ;TYPE 'DRIVE NOT PRESENT'
2212  010706  000705                    BR      M0              ;GO GET DRIVE NUMBER AGAIN
2213  010710  104401  035270    7$:     TYPE    ,MER11          ;'DRIVE NOT AVAILABLE'
2214  010714  000702                    BR      M0              ;GO GET DRIVE NUMBER AGAIN
2215  010716  123737  001220  005610  8$:     CMPB    DRIVE,FMTDPB    ;SAME DRIVE AS LAST TIME ?
2216  010724  001422                    BEQ     M2              ;BR IF IT IS
2217  010726  113737  001220  005610          MOVB    DRIVE,FMTDPB    ;SETUP DRIVE ADDRESS
2218  010734  012737  001466  001320          MOV     #822.,ENDCYL    ;SETUP FOR RM03
2219  010742  132762  000003  026526          BITB    #BIT00!BIT01,DRVTYP(R2)  ;SEE IF DRIVE RM03
2220  010750  001003                    BNE     9$              ;BR IF EITHER
2221  010752  012737  001466  001320          MOV     #822.,ENDCYL    ;SETUP ENDING CYLINDER FOR RM03
2222  010760  005037  001326    9$:     CLR     BEGTRK          ;CLEAR STARTING TRACK ADDRESS
2223  010764  005037  001322            CLR     BEGCYL          ;CLEAR STARTING CYLINDER ADDRESS
2224  010770  000400                    BR      M2              ;GET ADDRESS LIMITS FROM THE OPERATOR
2225
2226                                    ;GET ADDRESS LIMITS
2227
2228  010772  104401  035171    M2:     TYPE    ,ENTADR         ;'ENTER ADDRESS LIMITS'
2229                            ;       JSR     PC,PARENT       ;GET THE ADDRESS LIMITS
2230  010776  004437  017416            JSR     R4,PARENT       ;ENTER THE ADDRESS LIMITS
2231  011002  005650                    TABLE                   ;TABLE PARAMETERS
2232  011004  005037  001402            CLR     WRAP            ;CLEAR WRAP FLAG
2233  011010  023737  001320  001322    CMP     ENDCYL,BEGCYL   ;SEE IF ENDING CYLINDER EQ TO GT THAN BEGINNING
2234  011016  001402                    BEQ     2$              ;BR IF ON THE SAME CYLINDER
2235  011020  103410                    BLO     4$              ;BR IF LESS
2236  011022  000413                    BR      M4              ;TO CHECK DRIVE
2237  011024  023737  001324  001326  2$:     CMP     ENDTRK,BEGTRK   ;SEE IF ENDING TRACK EQ OR GT THAN BEGINNING
2238  011032  103007                    BHIS    M4              ;BR IF YES
2239  011034  104401  035733    3$:     TYPE    ,MADRER         ;INVALID ADDRESS ENTERED
2240  011040  000754                    BR      M2              ;TRY AGAIN
2241  011042  012737  177777  001402  4$:     MOV     #-1,WRAP        ;SET WRAP AROUND FLAG
2242  011050  000400                    BR      M4
2243
2244                                    ;GO GET DATA PATTERN FOR FORMAT
2245
2246  011052  005737  001316    M4:     TST     MODE            ;VERIFY MODE ?
2247  011056  001007                    BNE     M4A             ;NO
2248  011060  012737  133331  001340          MOV     #133331,PATA    ;VERIFY MODE USE WOREST CASE DATA
2249  011066  012737  133331  001342          MOV     #133331,PATB    ;INITIAL THE DATA PATTERN -SHIFT 3 BITS
2250  011074  000431                    BR      M5              ;START TO FORMAT AND CHECK
2251  011076  104401  036035    M4A:    TYPE    ,MSELP          ;GO TYPE 'SELECT PATTERN'
2252  011102  104411                    RDLIN                   ;READ THE KEYBOARD
2253  011104  012601                    MOV     (SP)+,R1        ;ENTRY ADDRESS
2254  011106  004537  022306            JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
2255  011112  011146                    3$                      ;ILLEGAL CHARACTER
2256  011114  011126                    1$                      ;CARRIAGE RETURN
2257  011116  011146                    3$                      ;" "
2258  011120  011126                    1$                      ;"'"
2259  011122  011140                    2$                      ;DIGIT 0-7
2260  011124  011146                    3$                      ;DIGIT 8-9
2261  011126  104401  036165    1$:     TYPE    ,MPATD          ;TYPE DEFAULT PATTERN
2262  011132  012702  000002            MOV     #2,R2           ;WORST CASE PATTERN
2263  011136  000406                    BR      4$              ;GO SAVE PATTERN
```

```
                        ;GET VALUE FOR SOFTWARE SWITCH REGISTER

2264  011140  020227  000002        2$:     CMP     R2,#2           ;IS # LARGER THAN 2
2265  011144  101403                        BLOS    4$              ;BRANCH IF NOT
2266  011146  104401  001202        3$:     TYPE    ,$QUES          ;TYPE '?'
2267  011152  000737                        BR      M4              ;RETYPE LINE
2268  011154  010237  001336        4$:     MOV     R2,PATSEL       ;SAVE PATTERN SELECTED
2269
2270                                ;GO TYPE 'STARTING FORMAT ON DRIVE N'
2271
2272  011160  012737  017260  000060  M5:   MOV     #NEWSVC,@#TKVEC ;VECTOR FOR CONTROL-O
2273  011166  005037  177776              CLR     @#PSW           ;LEVEL 0
2274  011172  005737  001316              TST     MODE            ;'FORMAT' OR 'CHECK' MODE ?
2275  011176  001403                        BEQ     1$              ;BR IF 'CHECK' MODE
2276  011200  104401  036200              TYPE    ,MSFOU          ;TYPE 'STARTING FORMAT ON DRIVE N'
2277  011204  000402                        BR      2$
2278  011206  104401  036235        1$:     TYPE    ,MSCHK          ;TYPE 'STARTING CHECK ON DRIVE N'
2279  011212                        2$:
2280  011212  013746  001220              MOV     DRIVE,-(SP)     ;;SAVE DRIVE FOR TYPEOUT
2281                                                                ;;TYPE DRIVE NUMBER
2282  011216  104403                        TYPOS                   ;;GO TYPE--OCTAL ASCII
2283  011220     001                        .BYTE   1               ;;TYPE 1 DIGIT(S)
2284  011221     000                        .BYTE   0               ;;SUPPRESS LEADING ZEROS
2285  011222  104401  001203              TYPE    ,$CRLF          ;CR-LF
2286
2287                                ;THIS CODE RETRIEVES THE BAD SPOT FILE FROM THEPACK (IF FORMATTED PACK)
2288                                ;SET UP DPB BLOCK READ CYL 822. TRK 4
2289
2290  011226  005037  001346        RETBAD: CLR     RETRY           ;CLEAR THE RETRY FLAG
2291  011232  013737  001220  005610        MOV     DRIVE,FMTDPB    ;LOAD DRIVE #
2292  011240  012737  157700  005614        MOV     #-(32.*258.),FMTDPB+4 ;WORD COUNT FULL TRACK INCLUDING HEAD
2293  011246  012737  043746  005616        MOV     #BUFP,FMTDPB+6  ;BUFFER ADDRESS
2294  011254  105037  005620               CLRB    FMTDPB+10       ;SECTOR 0
2295  011260  112737  000004  005621        MOVB    #4,FMTDPB+11    ;TRACK 4
2296  011266  012737  001466  005622        MOV     #822.,FMTDPB+12 ;CYL 822
2297  011274  112737  000173  005612        MOVB    #RDHD,FMTDPB+2  ;READ HEAD AND DATA COMMAND
2298  011302  012737  177777  001406        MOV     #-1,PACK        ;RESET THE MFG AVAILABLE FLAG
2299  011310  112737  000143  005612        MOVB    #SETFMT,FMTDPB+2  ;SET 16 BIT MODE
2300  011316  112737  000020  005611        MOVB    #20,FMTDPB+1    ;FMT SET
2301  011324  004037  027410        1$:     JSR     R0,RM03         ;SET THE FORMAT BIT
2302  011330  005610                        FMTDPB
2303  011332  000774                        BR      1$
2304  011334  005737  005626        2$:     TST     FMTDPB+16       ;THE COMMAND IS DONE ?
2305  011340  001775                        BEQ     2$              ;BRANCH IF NOT
2306  011342  112737  000173  005612        MOVB    #RDHD,FMTDPB+2  ;RELOAD THE READ HEAD AND DATA COMMAND
2307  011350  012737  011350  001124        MOV     #.,$LPERR       ;LOOP ON ERROR ADDRESS
2308  011356  012706  001100               MOV     #STACK,SP       ;RESET STACK POINT
2309  011362  004037  027410        3$:     JSR     R0,RM03         ;READ THE LAST TRACK
2310  011366  005610                        FMTDPB
2311  011370  000774                        BR      3$              ;BRANCH IF QUEUE FAIL
2312  011372  005737  005626        4$:     TST     FMTDPB+16       ;ALL DONE
2313  011376  001775                        BEQ     4$              ;BRANCH IF NOT
2314  011400  100024                        BPL     6$              ;BRANCH IF NO ERROR
2315  011402  012737  011430  001174        MOV     #5$,$ESCAPE     ;;ESCAPE TO 5$ ON ERROR
2316  011410  004737  016144               JSR     PC,ERINDX       ;SEE WHICH ERROR
2317  011414  104010                        ERROR   10
2318  011416  104011                        ERROR   11
2319  011420  104012                        ERROR   12
```

```
                                        ERROR   13
                                        ERROR   14
                                        ERROR   21
                                5$:     JSR     PC,LOP.CK       ;CHECK LOOP ON ERROR
001346                                  CMP     #3,RETRY        ;DONT' TRY OVER 3 TIMES
                                        BLO     8$              ;
001346                                  INC     RETRY
                                        BR      3$              ;TRY AGAIN
001174                          6$:     CLR     $ESCAPE         ;
001346                                  CLR     RETRY
005626                                  TST     FMTDPB+16       ;ANY ERROR ?
                                        BMI     8$              ;BRANCH IF ANY
043746                                  BIT     #BIT15!BIT14,@#BUFP ;HEADER INFORMATION  OK ?
                                        BEQ     8$              ;BRANCH IF NOT
043746                                  BIT     #BIT12,@#BUFP   ;FMT BET MUST BE SET
                                        BEQ     8$              ;BRANCH IF NOT SET
043746                                  CMP     #151466,@#BUFP  ;ON THE LAST CYLINDER ?
                                        BNE     8$              ;BRANCH IF NOT
043750                                  CMP     #2000,@#BUFP+2  ;ON TRACK 4 AND SECTOR 0
                                        BNE     8$              ;BRANCH IF NOT
043752                                  TST     @#BUFP+4        ;SERIAL NUMBER SHOULD NOT 0
                                        BNE     7$              ;BRANCH IF NOT 0
043754                                  TST     @#BUFP+6        ;SECOND WORD OF SERIAL NUMBER
                                        BEQ     8$              ;BRANCH IF ZERO
043756                          7$:     TST     @#BUFP+10       ;THE THIRD WORD MUST BE 0
                                        BNE     8$              ;BRANCH IF NOT
043760                                  CMP     #-1,@#BUFP+12   ;AN ALIGNMENT PACK ?
                                        BNE     .+6             ;BRANCH IF NOT
016066                                  JMP     REJEC1
043760                                  TST     @#BUFP+12       ;A DATA PACK ?
                                        BNE     8$              ;BRANCH IF NOT IDITENTIFIED
001406                                  MOV     #400,PACK       ;SET MFG BAD SPOT FILE AVAILABLE FLAG
                                8$:     NOP                     ;DONE
                                ;THIS CODE INITIALIZES THE BAD16,BAD18,USTAB,USSAV TABLES
                                ;

001100                  TABINT: MOV     #STACK,SP       ;INITIAL STACK POINT
                                CLR     -(SP)           ;TERMINATOR
004430                          MOV     #USSAV,-(SP)    ;USSAV TABLE ADDRESS
002420                          MOV     #BAD18,-(SP)    ;MFG 18 BIT FILE ADDRESS
001414                          MOV     #BAD16,-(SP)    ;MFG 16 BIT FILE ADDRESS
003424                          MOV     #USTAB,R3       ;USER BAD SPOT FILE ADDRESS
000004                  1$:     MOV     #4,R4           ;FIRST 4 WORDS SET TO 0 TEMPERARY
                        2$:     CLR     (R3)+
                                DEC     R4
                                BNE     2$              ;BRANCH IF NOT DONE
000374                          MOV     #252.,R4        ;SET -1 TO ALL THE REST LOCATIONS
177777                          MOV     #-1,R5
                        3$:     MOV     R5,(R3)+
                                DEC     R4              ;DECREMENT WORD COUNT
                                BNE     3$              ;BRANCH IF NOT DONE
                                MOV     (SP)+,R3        ;NEXT TABLE
                                BNE     1$              ;LOOPING BACK IF NOT TERMINATOR


                        ;THIS CODE LOADS TABLE BAD16,BAD18,USSAV,FROM PACK (IF FORMATTED PACK
```

```
2376
2377  011666  005737  001406          ;TABLD:  TST     PACK            ;BAD SPOT FILE AVAILABLE FROM PACK ?
2378                                  ;        BLE     5$              ;BRANCH IF NOT
2379  011672  003451                  ;        BLE     6$              ;BRANCH IF NOT AVALIABLE
2380  011674  012702  043752                  MOV     #BUFP+4,R2      ;LOAD BAD16 TABLE FIRST
2381  011700  012703  001414                  MOV     #BAD16,R3       ;TABLE ADDRESS
2382  011704  012704  000400                  MOV     #256.,R4        ;WORD COUNT
2383  011710  012223          1$:     MOV     (R2)+,(R3)+     ;LOAD THE WHOLE SECTOR
2384  011712  005304                  DEC     R4              ;BRANCH IF NOT DONE
2385  011714  001375                  BNE     1$
2386  011716  012702  044756                  MOV     #BUFP+520.,R2   ;258 X 2 + 2 X 2 + BUFP
2387  011722  012703  002420                  MOV     #BAD18,R3       ;TABLE ADDRESS
2388  011726  012704  000400                  MOV     #256.,R4        ;WORD COUNT
2389  011732  012223          2$:     MOV     (R2)+,(R3)+     ;LOAD INTO TABLE
2390  011734  005304                  DEC     R4              ;DECREMENT WORD COUNT
2391  011736  001375                  BNE     2$              ;BRANCH IF NOT DONE
2392  011740  012702  057026                  MOV     #BUFP+4+<11.*516.> ,R2 ;SECTOR 11
2393  011744  005737  001362                  TST     SEC30           ;FORMAT IN 16 BIT MODE
2394  011750  100402                  BMI     3$              ;BRANCH IF IT IS
2395  011752  012702  056022                  MOV     #BUFP+4+<10.*516.> ,R2 ;ECTOR 10
2396  011756  012703  004430          3$:     MOV     #USSAV,R3       ;LOAD THE TABLE
2397  011762  012704  000400                  MOV     #256.,R4        ;WORD COUNT
2398  011766  012223          4$:     MOV     (R2)+,(R3)+     ;
2399  011770  005304                  DEC     R4
2400  011772  001375                  BNE     4$
2401  011774  062702  000004                  ADD     #4,R2           ;ACCESSES THE USTAB BUFFER ADD
2402  012000  012703  003424                  MOV     #USTAB,R3       ;TABLE ADDRESS
2403  012004  012704  000400                  MOV     #256.,R4        ;WORD COUNT
2404  012010  012223          5$:     MOV     (R2)+,(R3)+     ;LOAD THE TABLE
2405  012012  005304                  DEC     R4              ;DECREMENT THE COUNT
2406  012014  001375                  BNE     5$              ;BRANCH IF NOT ALL DONE
2407  012016  000240          6$:     NOP                     ;EXIT
2408                                  ;5$:     NOP             ;DONE
2409
2410                                  ;THIS CODE ASK O.P TO ENTER A SERIAL NUMBER FOR UN FORMATTED PACK
2411                                  ;
2412                                  ;
2413
2414  012020  005737  001406          SERNL:  TST     PACK            ;BAD SPOT FILE AVAILABLE ?
2415  012024  003124                  BGT     4$              ;BRANCH IF AVAILABLE
2416  012026  105737  001150                  TSTB    $AUTOB          ;RUN UNDER APT ?
2417  012032  001144                  BNE     5$              ;BRANCH IF IT IS
2418  012034  004737  024422                  JSR     PC,$TKINT       ;INITIALTHE KEYBOARD
2419  012040  104401  001203                  TYPE    ,$CRLF          ;TYPE CRLF
2420  012044  104401  036462                  TYPE    ,MESG1          ;"ENTEP TWO WORDS FOR SN"
2421  012050  104401  036550          1$:     TYPE    ,SN1            ;
2422  012054  012702  000012                  MOV     #10.,R2         ;MAXMUM 10 OCTAL DIGITS ALLOWED
2423  012060  005037  001414                  CLR     BAD16           ;CLEAR THE LSW OF THE SERIAL NUMBER
2424  012064  005037  001416                  CLR     BAD16+2         ;CLEAR THE MSW OF THE SERIAL NUMBER
2425  012070  104411                  RDLIN                   ;READ IN THE STRING
2426  012072  012601                  MOV     (SP)+,R1        ;ADDRESS OF THE READ IN STRING
2427  012074  105711          2$:     TSTB    (R1)            ;TERMINATOR OF THE INPUT STRING
2428  012076  001471                  BEQ     3$              ;BRANCH IF IT IS
2429  012100  121127  000060                  CMPB    (R1),#'0        ;LESS THAN ASCIZ 0 ?
2430  012104  103761                  BLO     1$              ;ENTER AGAIN IF IT IS
2431  012106  121127  000067                  CMPB    (R1),#'7        ;HIGH THAN ASCIX 7 ?
```

```
2432  012112  101356                          BHI     1$              ;ENTER AGAIN IF SO
2433  012114  013746  001414                  MOV     BAD16,-(SP)     ;SAVE THE LSW
2434  012120  042737  100000  001414          BIC     #BIT15,BAD16    ;CLEAR THE SIGN BIT OF LSW
2435  012126  006137  001414                  ROL     BAD16           ;ROTATE THE LSW FIVE TIMES
2436  012132  006137  001414                  ROL     BAD16           ;TO MOVE THE MOST SIGN DIGIT
2437  012136  006137  001414                  ROL     BAD16           ;TO BIT 0 TO BIT 2
2438  012142  006137  001414                  ROL     BAD16           ;
2439  012146  006137  001414                  ROL     BAD16           ;
2440  012152  042737  177770  001414          BIC     #177770,BAD16   ;LEFT ON THE MS DIGIT OF THE LSW
2441  012160  006337  001416                  ASL     BAD16+2         ;SHIFT THE MSW  LEFT ONE OCTAL DIGIT
2442  012164  100731                          BMI     1$              ;ENTER AGAIN IF SIGN BIT SET
2443  012166  006337  001416                  ASL     BAD16+2         ;
2444  012172  100726                          BMI     1$              ;
2445  012174  006337  001416                  ASL     BAD16+2         ;
2446  012200  100723                          BMI     1$              ;
2447  012202  063737  001414  001416          ADD     BAD16,BAD16+2   ;APPENDING THE DIGITS INTO MSW
2448  012210  012637  001414                  MOV     (SP)+,BAD16     ;RESTORE THE LSW
2449  012214  006337  001414                  ASL     BAD16           ;SHIFT THE LSW LEFT ONE OCTAL DIGIT
2450  012220  006337  001414                  ASL     BAD16           ;
2451  012224  006337  001414                  ASL     BAD16           ;
2452  012230  042737  100000  001414          BIC     #BIT15,BAD16    ;CLEAR THE SIGN BIT
2453  012236  111103                          MOVB    (R1),R3         ;LOAD THE CURRENT READ IN DIGITS
2454  012240  042703  177770                  BIC     #177770,R3      ;LEFT ONLY ONE OCTAL DIGIT
2455  012244  060337  001414                  ADD     R3,BAD16        ;APPEND THE READ IN CHARACTER TO LSW
2456  012250  005201                          INC     R1              ;ADJUST THE READ IN STRING ADDRESS
2457  012252  005302                          DEC     R2              ;DECREMENT ONE DIGIT COUNT
2458  012254  001307                          BNE     2$              ;BRANCH IF NOT DONE
2459  012256  105711                          TSTB    (R1)            ;IF 10 DIGITS HAVE BEEN ENTERED
2460  012260  001273                          BNE     1$              ;MUST BE FOLLOWED BY <CR>
2461  012262  005737  001414          3$:     TST     BAD16           ;SERIAL NUMBER MUST BE NOT 0
2462  012266  001003                          BNE     4$              ;BRANCH IF NOT 0
2463  012270  005737  001416                  TST     BAD16+2         ;
2464  012274  001665                          BEQ     1$              ;ENTER AGAIN IF ZERO
2465  012276  013737  001414  002420  4$:     MOV     BAD16,BAD18             ;LOAD ALL TABLE WITH THE SERIAL NUMBER
2466  012304  013737  001414  003424          MOV     BAD16,USTAB             ;
2467  012312  013737  001414  004430          MOV     BAD16,USSAV             ;
2468  012320  013737  001416  003426          MOV     BAD16+2,USTAB+2 ;
2469  012326  013737  001416  004432          MOV     BAD16+2,USSAV+2 ;
2470  012334  013737  001416  002422          MOV     BAD16+2,BAD18+2 ;
2471  012342  000410                          BR      6$              ;EXIT
2472  012344  005237  004430          5$:     INC     USSAV           ;CODE FOR AUTO MODE APT,ACT,ETC.
2473  012350  005237  001414                  INC     BAD16
2474  012354  005237  002420                  INC     BAD18
2475  012360  005237  003424                  INC     USTAB
2476  012364  000240                  6$:     NOP                     ;DONE
2477                                  ;       MOV     #NEWSVC,@#^KVEC ;NEW INTERRUPT VECTOR FOR KEYBOARD
2478                                  ;       CLR     @#PSW
2479                                  ;THIS CODING PROVID UTILITY ROUTINE ENTRIES
2480
2481  012366  000240                  CALIN:  NOP
2482  012370  004737  024422                  JSR     PC,$TKINT       ;INITIALIZE THE KEYBOARD
2483  012374  105737  001150                  TSTB    $AUTOB  ;AUTO BYTE SET ?
2484  012400  001033                          BNE     CKADRS          ;IF SO, BRANCH
2485  012402  005737  001266                  TST     $CDW1           ;ALLOW ANY UTILITY ROUTINE ?
2486  012406  001430                          BEQ     CKADRS          ;BRANCH IF NOT
2487  012410  104401  037111          1$:     TYPE    .MSGBLK         ;ENTER MESSAGE BLOCK
```

```
2488  012414  104411                          RDLIN                   ;READ IN THE ROUTINE NUMBER
2489  012416  012601                          MOV     (SP)+,R1        ;ADDRESS OF INPUT BUFFER
2490  012420  122711  000060                  CMPB    #'0,(R1)        ;INPUT NUMBER LESS THAN 0 ?
2491  012424  101012                          BHI     2$              ;BRANCH IF SO
2492  012426  122711  000064                  CMPB    #'4,(R1)        ;INPUT NUMBER LARGER THAN 4 ?
2493  012432  103407                          BLO     2$              ;BRANCH IF SO
2494  012434  042711  177760                  BIC     #177760,(R1)    ;CLEAR OF THREE BITS
2495  012440  011101                          MOV     (R1),R1         ;FOUND THE INDEX VALUE
2496  012442  006301                          ASL     R1              ;WORD INDEX
2497  012444  004771  006120                  JSR     PC,@TABCAL(R1)  ;CALL THE DESIRED UTILITY ROUTINE
2498  012450  000757                          BR      1$              ;LOOP BACK TO THE SAME POINT
2499  012452  000240                  2$:     NOP                     ;EXIT
2500  012454  005037  001366                  CLR     DS.CYL          ;CLEAR ALL PARAMETERS
2501  012460  005037  001370                  CLR     DS.TRK          ;
2502  012464  005037  001350                  CLR     SAVSEC          ;
2503                                  ;SETUP TOTAL TRACK COUNT FOR FORMAT
2504
2505  012470  023737  001320  001322  CKADRS: CMP     ENDCYL,BEGCYL   ;STARTING AND ENDING CYLINDERS THE SAME ?
2506  012476  001011                          BNE     5$              ;BRANCH IF BEGCYL NOT EQUAL TO ENDCYL
2507  012500  013737  001324  001330          MOV     ENDTRK,TTRKS    ;END TRACK ADDRESS
2508  012506  163737  001326  001330          SUB     BEGTRK,TTRKS    ;SUBTRACT THE STARTING TRACK ADDRESS
2509  012514  005237  001330                  INC     TTRKS           ;MAKE THE NUMBER OF TRACKS INCLUSIVE
2510  012520  000450                          BR      SETPAT          ;SETUP THE DATA PATTERN
2511  012522  005737  001402          5$:     TST     WRAP            ;WRAP AROUND FLAG SET ?
2512  012526  001407                          BEQ     1$              ;NO
2513  012530  012702  001466                  MOV     #822.,R2        ;INITIAL COUNTER TO 822
2514  012534  163702  001322                  SUB     BEGCYL,R2       ;FIRST PART OF CYL #
2515  012540  063702  001320                  ADD     ENDCYL,R2       ;SECOND PART OF CYL #
2516  012544  000405                          BR      4$
2517  012546  013702  001320          1$:     MOV     ENDCYL,R2       ;ENDING CYLINDER
2518  012552  163702  001322                  SUB     BEGCYL,R2       ;SUBTRACT THE STARTING CYLINDER
2519  012556  005302                          DEC     R2              ;EXCLUSIVE LAST CYLINDER
2520  012560  012737  000005  001330  4$:     MOV     #5,TTRKS        ;INITATE COUNTER
2521  012566  163737  001326  001330          SUB     BEGTRK,TTRKS    ;# OF TRACK ON STARTING CYLINDER
2522  012574  063737  001324  001330          ADD     ENDTRK,TTRKS    ;# OF TRACK ON ENDING CYLINDER
2523  012602  005237  001330                  INC     TTRKS           ;INCLUSIVE ONE TRACK
2524  012606  005302                  3$:     DEC     R2              ;DECREMENT THE CYLINDER COUNT
2525  012610  100404                          BMI     6$              ;BR IF DONE
2526  012612  062737  000005  001330          ADD     #5.,TTRKS       ;ADD CYLINDER WORTH OF TRACKS
2527  012620  000772                          BR      3$              ;CONTINUE
2528  012622  005037  001410          6$:     CLR     EDIT            ;RESET EDIT FLAG
2529  012626  023727  001330  010011          CMP     TTRKS,#(821.*5) ;ALL TRACKS ARE FORMATTED ?
2530  012634  103402                          BLO     SETPAT          ;BRANCH IF NOT
2531  012636  005237  001410                  INC     EDIT            ;SET EDIT FLAG
2532
2533                                  ;THIS CODE SETS UP THE SELECTED DATA PATTERN TO BE WRITTEN ON EACH SECTOR IMAGE
2534
2535  012642  005737  001316          SETPAT: TST     MODE            ;VERIFY MODE ?
2536  012646  001427                          BEQ     1$              ;YES
2537  012650  005037  001340          4$:     CLR     PATA            ;CLEAR DATA PATTERN A
2538  012654  005037  001342                  CLR     PATB            ;CLEAR DATA PATTERN B
2539  012660  005737  001336                  TST     PATSEL          ;SEE IF PATTERN OF ONES
2540  012664  001420                          BEQ     1$              ;BR IF SO
2541  012666  012737  177777  001340          MOV     #177777,PATA    ;PATA = AB
2542  012674  012737  177777  001342          MOV     #177777,PATB    ;PATB=CD
2543  012702  022737  000001  001336          CMP     #1,PATSEL       ;SEE IF PATTERN OF ZEROS
```

```
                                        ;GET VALUE FOR SOFTWARE SWITCH REGISTER

2544  012710  001406                            BEQ     1$              ;BRANCH IF SO
2545  012712  012737  066666  001340            MOV     #066666,PATA    ;SET UP WORST CASE
2546  012720  012737  066666  001342            MOV     #066666,PATB    ;SET UP WORST CASE
2547  012726  013703  001354          1$:       MOV     WC,R3           ;SET UP COUNTER
2548  012732  012700  043746                    MOV     #BUFP,R0        ;SET UP MEMORY POINTER
2549  012736  013701  001340          3$:       MOV     PATA,R1         ;SET UP PATTERN IN R1
2550  012742  013702  001342                    MOV     PATB,R2         ;SET UP PATTERN IN R2
2551  012746  010120                    2$:     MOV     R1,(R0)+        ;MOV 1ST PAT INTO MEM
2552  012750  010220                            MOV     R2,(R0)+        ;MOV 2ND PAT INTO MEM
2553  012752  005303                            DEC     R3              ;KEEP COUNT
2554  012754  005303                            DEC     R3              ;KEEP COUNT
2555  012756  001373                            BNE     2$              ;DO IT AGAIN IF R3 NOT 0
2556                                  ;THIS CODE SETS UP FOR THE ACTUAL FORMAT
2557
2558  012760  012737  017260  000060            MOV     #NEWSVC,@#TKVEC ;NEW INTERRUPT VECTOR
2559  012766  005037  177776                    CLR     @#PSW           ;CLEAR THE PS
2560  012772  004737  016312          WRTRK:    JSR     PC,SETTBL       ;GO SET UP DRIVER TABLE
2561  012776  004737  016624                    JSR     PC,SETHDR       ;GO INITIALIZE HEADERS IN THE SECTOR BUFFER IN CORE
2562  013002  112737  000020  005611            MOVB    #20,FMTDPB+1    ;LOAD FMT22 BIT
2563  013010  005737  001362                    TST     SEC30           ;18 BIT MODE ?
2564  013014  001002                            BNE     1$              ;BR IF NOT
2565  013016  105037  005611                    CLRB    FMTDPB+1        ;CLEAR THE FMT22 BIT
2566  013022  112737  000143  005612  1$:       MOVB    #SETFMT,FMTDPB+2  ;'LOAD FORMAT' COMMAND
2567  013030  004037  027410          2$:       JSR     R0,RM03         ;START THE COMMAND
2568  013034  005610                            FMTDPB                  ;DPB ADDRESS
2569  013036  000774                            BR      2$              ;QUEUE FULL RETURN
2570  013040  005737  005626          3$:       TST     FMTDPB+16       ;LOOK FOR DONE
2571  013044  001775                            BEQ     3$              ;LOOP UNTIL FINISHED
2572
2573                                  ;SET UP SECCU TABLE: BAD SPOT ON CURRENT TRACK ,RETRIEVED FROM PACK
2574                                  ;
2575  013046  012703  005434          WRTRKX:   MOV     #SECCU,R3       ;INITILIZE THE SECCU TABLE
2576  013052  012704  000040                    MOV     #32.,R4         ;TOTAL 32 WORD MAX
2577  013056  012702  177777                    MOV     #-1,R2          ;LOAD ALL LOCATION TO -1
2578  013062  010223                    1$:     MOV     R2,(R3)+        ;
2579  013064  005304                            DEC     R4
2580  013066  001375                            BNE     1$              ;BRANCH IF NOT DONE
2581                                  ;
2582                                  ;LOAD THE BAD SPOT INTO SECCU TABLE IF ANY
2583                                  ;
2584  013070  005737  001406          WRTRKY:   TST     PACK            ;BAD SPOT FILE AVAILABLE AT ALL
2585  013074  003461                            BLE     6$              ;BRANCH IF NOT
2586  013076  005046                            CLR     -(SP)           ;CLEAR UP THE DUMMY RETURN ADDRESS
2587  013100  012746  003434                    MOV     #USTAB+8.,-(SP) ;USER BAD SECTOR TABLE
2588  013104  012702  001414                    MOV     #BAD16,R2       ;R2 POINTER OF MFG BAD SPOT TABLE
2589  013110  005737  001362                    TST     SEC30           ;IF 18 BIT MODE CHANGE POINTER
2590  013114  100402                            BMI     1$              ;BRANCH IF 16 BIT MODE
2591  013116  012702  002420                    MOV     #BAD18,R2       ;
2592  013122  010201                    1$:     MOV     R2,R1           ;R1: LAST ADDRESS OF BAD SPOT TABLE
2593  013124  062701  000776                    ADD     #510.,R1        ;
2594  013130  012703  005434                    MOV     #SECCU,R3       ;TABLE ADDRESS
2595  013134  012704  000076                    MOV     #62.,R4         ;LAST ADDRESS OF THE TABLE
2596  013140  060304                            ADD     R3,R4           ;
2597  013142  062702  000010                    ADD     #10,R2          ;BAD SPOT STARTS FROM THE 5TH WORD OF THE FILE
2598  013146  023712  005622          2$:       CMP     FMTDPB+12,(R2)  ;BAD SPOT ON THIS CYLINDER ?
2599  013152  001007                            BNE     3$              ;BRANCH IF NOT
```

```
2600  013154  123762  005621  000003          CMPB    FMTDPB+11,3(R2) ;ALSO, ON THE SAME TRK ?
2601  013162  001003                          BNE     3$              ;BRANCH IF NOT
2602  013164  116205  000002                  MOVB    2(R2),R5        ;BAD SECTOR
2603  013170  010523                          MOV     R5,(R3)+        ;WORD STORAGE
2604  013172  062702  000004          3$:     ADD     #4,R2           ;ADVANCE TO NEXT SE-
2605  013176  020102                          CMP     R1,R2           ;ALL SPOT ARE CHECKED ?
2606  013200  103412                          BLO     5$              ;BRANCH IF IT IS
2607  013202  020403                          CMP     R4,R3           ;END OF SECCU TABLE ?
2608  013204  103401                          BLO     4$              ;REJECT THE PACK
2609  013206  000757                          BR      2$              ;LOOPING BACK
2610  013210  004737  017162          4$:     JSR     PC,RE>TRT               ;IMPORPER MFG INFORMATION
2611  013214  012737  177777  001406          MOV     #-1,PACK                ;RESET MFG FILE AVAILABLE FLAG
2612  013222  000137  011604                  JMP     TABINT                  ;AND START AGAIN
2613  013226  011601                  5$:     MOV     (SP),R1         ;NEXT TABLE
2614  013230  062701  000766                  ADD     #502.,R1        ;ENDING OF THE TABLE
2615  013234  012602                          MOV     (SP)+,R2        ;BEGINNING OF THE TABLE
2616  013236  001343                          BNE     2$              ;BRANCH IF NOT DUMMY ADD
2617  013240  000240                  6$:     NOP                     ;DONE
2618
2619                                  ;THIS CODE SORTS THE BAD SECTOR TABLE "SECCU" IN ASCENGING ORDER
2620                                  ;AND DELETES THE DUPLICATED ONES
2621                                  ;
2622  013242  012702  005434          SORT1:  MOV     #SECCU,R2       ;TOP OF THE TABLE
2623  013246  012703  005532                  MOV     #SECCU+62.,R3   ;BOTTOM OF THE TABLE
2624  013252  012704  000037                  MOV     #31.,R4 ;TOTAL 32 ELEMENT,SORT 31 TIMES
2625  013256  022712  177777                  CMP     #-1,(R2)        ;IS THE TABLE EMPTY ?
2626  013262  001451                          BEQ     6$              ;BRANCH IF IT IS, QUICK EXIT
2627  013264  021262  000002          1$:     CMP     (R2),2(R2)      ;N TH ELEMENT : N+1 TH ELEMENT
2628  013270  103430                          BLO     4$              ;BRANCH IF SMALLER
2629  013272  001406                          BEQ     2$              ;DELET N+1 TH ELEMENT, IF SAME
2630                                                                  ;OTHERWISE, N > N+1
2631  013274  011246                          MOV     (R2),-(SP)      ;STORE N ELEMENT
2632  013276  016212  000002                  MOV     2(R2),(R2)      ;SWITCH N+1 ELEMENT TO N ELEMENT
2633  013302  012662  000002                  MOV     (SP)+,2(R2)     ;SWITCH N ELEMENT TO N+1 ELEMENT
2634  013306  000421                          BR      4$              ;ADJUST TOP POINTER
2635  013310  022712  177777          2$:     CMP     #-1,(R2)        ;TERMINATOR ?
2636  013314  001416                          BEQ     4$              ;BRANCH IF IT IS
2637  013316  010246                          MOV     R2,-(SP)        ;SAVE TOP POINTER
2638  013320  062702  000002                  ADD     #2,R2           ;DELET 2(R2)
2639  013324  016222  000002          3$:     MOV     2(R2),(R2)+     ;
2640  013330  022702  005532                  CMP     #SECCU+62.,R2   ;BOTTOM OF THE TABLE ?
2641  013334  101373                          BHI     3$              ;BRANCH IF NOT
2642  013336  012737  177777  005532          MOV     #-1,SECCU+62.   ;LOAD -1 TO TABLE BOTTOM
2643  013344  012602                          MOV     (SP)+,R2        ;RESTORE THE TOP POINTER
2644  013346  005304                          DEC     R4              ;DECREMENT ONE ELEMENT COUNT
2645  013350  001416                          BEQ     6$              ;BRANCH IF ALL DONE
2646  013352  062702  000002          4$:     ADD     #2,R2           ;INCREMENT THE TOP POINTER
2647  013356  020302                          CMP     R3,R2           ;REACH THE BOTTOM ?
2648  013360  001341                          BNE     1$              ;BRANCH IF NOT
2649  013362  005304                  5$:     DEC     R4              ;DECREMENT ONE SORT COUNT
2650  013364  001410                          BEQ     6$              ;BRANCH IF ALL DONE
2651  013366  012702  005434                  MOV     #SECCU,R2       ;RESET TOP POINTER
2652  013372  162703  000002                  SUB     #2,R3           ;UPDATE BOTTOM POINTER
2653  013376  022703  005434                  CMP     #SECCU,R3       ;EXIT IF TOP = BOTTOM
2654  013402  001401                          BEQ     6$              ;
2655  013404  000727                          BR      1$              ;SORT THE REST ELEMENTS
```

```
                        ;GET VALUE FOR SOFTWARE SWITCH REGISTER
2656  013406  000240          6$:     NOP                     ;DONE
2657  013410  005037  005536          CLR     NEXT            ;FLAG START OF A NEW TRACK
2658  013414  005037  001314  WRTRK1: CLR     SOFSW           ;CLEAR ERROR COUNTER
2659  013420  005037  001346          CLR     RETRY           ;ZERO THE RETRY COUNTER
2660  013424  105037  005620          CLRB    FMTDPB+10       ;RESTORE SECTOR
2661  013430  013737  001356  005614  MOV     MWC,FMTDPB+4    ;RESTORE WC
2662  013436  012737  043746  005616  MOV     #BUFP,FMTDPB+6  ;RESTORE BA
2663                          ;DELET THE FOLLOWING TWO LINES FOR NEW VERIFY OPERATION
2664                          ;       TST     MODE            ;'FORMAT' OR 'CHECK' MODE ?
2665                          ;       BEQ     CKTRK           ;BR IF 'CHECK' MODE
2666                          ;
2667                          ;THIS CODE TO SET UP WORD COUNT AND BUFFER ADDRESS
2668                          ;
2669  013444  022737  177777  005536  WRTRKA: CMP     #-1,NEXT        ;END OF OPERATION ?
2670  013452  001001                  BNE     1$              ;BRANCH IF NOT
2671  013454  000000                  HALT                    ;OTHERWISE SOMETHING WRONG *******
2672  013456  113737  005536  005620  1$:     MOVB    NEXT,FMTDPB+10  ;STARTING SECTOR
2673  013464  012701  005434          MOV     #SECCU,R1       ;R1 POINTS TO THE BAD SECTOR TABLE
2674  013470  022711  177777          2$:     CMP     #-1,(R1)        ;END OF TABLE ?
2675  013474  001511                  BEQ     WRTRKD          ;BRANCH IF IT IS
2676  013476  032711  100000          BIT     #BIT15,(R1)     ;THE BAD SECTOR HAS ALREADY ACCESSED ?
2677  013502  001403                  BEQ     3$              ;BRANCH IF NOT
2678  013504  062701  000002          ADD     #2,R1           ;ADJUST THE POINTER
2679  013510  000767                  BR      2$              ;LOOPING BACK
2680  013512  123711  005620          3$:     CMPB    FMTDPB+10,(R1)  ;BAD SPOT = STARTING SECTOR ?
2681  013516  001415                  BEQ     WRTRKB          ;BRANCH   IF IT I
2682  013520  103401                  BLO     4$              ;BRANCH IF START SEC < BAD SPOT
2683  013522  000000                  HALT                    ;OTHERWISE   SOMETHING WRONG
2684  013524  111102                  4$:     MOVB    (R1),R2         ;UPDATE NEXT STARTING ADDRESS
2685  013526  023702  001364          CMP     MAXSEC,R2       ;NEXT BAD SPOT IS THE LAST SECTOR?
2686  013532  001463                  BEQ     WRTRKF          ;BRANCH IF LAST SECTOR *******
2687  013534  103471                  BLO     WRTRKD          ;BRANCH IF LESS ******
2688                          ;       BLOS    WRTRKD          ;BRANCH IF IT IS
2689  013536  005202                  INC     R2              ;NEXT = BAU + 1
2690  013540  010237  005536          MOV     R2,NEXT         ;NEXT STARTING ADDRESS
2691  013544  052721  100000          BIS     #BIT15,(R1)+    ;MARK THE ACCESSED SECTOR
2692  013550  000432                  BR      WRTRKC          ;TO SET UP WORD COUNT
2693  013552  105237  005620  WRTRKB: INCB    FMTDPB+10       ;INCREMENT THE STARTING SECTOR
2694  013556  052721  100000          BIS     #BIT15,(R1)+    ;MARK THE ACCESSED SECTOR
2695  013562  123711  005620          CMPB    FMTDPB+10,(R1)  ;STILL ON ANOTHER BAD SPOT ?
2696  013566  001771                  BEQ     WRTRKB                  ;LOOP AGAIN
2697  013570  123737  005620  001364  CMPB    FMTDPB+10,MAXSEC        ;IF START SECTOR > MAXSEC
2698  013576  101402                  BLOS    1$              ;IMPORPER MFG INDORMATION
2699                          ;       JSR     PC,RESTRT       ;START AGAIN AND REBUILD FMG FILE
2700                          ;       MOV     #-1,PACK        ;RESET THE MFG FILE AVAILABLE FLAG
2701                          ;       JMP     TABINT          ;RESTART POINT
2702  013600  000137  014564          JMP     HSET            ;EXIT
2703  013604  022711  177777  1$:     CMP     #-1,(R1)        ;TO SEE IF NEXT BAD SPOT IN TABLE
2704  013610  001443                  BEQ     WRTRKD          ;BRANCH IF NONE
2705  013612  111102                  MOVB    (R1),R2         ;UPDATE THE NEXT
2706  013614  023702  001364          CMP     MAXSEC,R2       ;BAD SECTOR IS THE LAST SECTOR ?
2707                          ;       BLOS    WRTRKD          ;BRANCH IF IT IS
2708  013620  001430                  BEQ     WRTRKF          ;*********
2709  013622  103436                  BLO     WRTRKD          ;BRANCH IF LESS ***
2710  013624  005202                  INC     R2              ;BAD SEC +1
2711  013626  010237  005536          MOV     R2,NEXT         ;NEXT STARTING ADDRESS
```

```
2712  013632  052711  100000              BIS     #BIT15,(R1)       ;MARK THE ACCESSED SECTOR
2713  013636  013702  005536    WRTRKC:   MOV     NEXT,R2           ;CALCULATE THE WORD COUNT
2714  013642  162702  000002              SUB     #2,R2             ;ENDSECTOR = NEXT - 2
2715  013646  006302                      ASL     R2                ;WORD INDEX
2716  013650  016203  006232              MOV     WCTBL(R2),R3      ; WORD CTR FOR SECTOR 0 TO ENDING SECTOR
2717  013654  113702  005620              MOVB    FMTDPB+10,R2      ;STARTING SECTOR
2718  013660  006302                      ASL     R2                ;WORD INDEX
2719  013662  166203  006232              SUB     WCTBL(R2),R3      ;WORD COUNT FOR SECTOR 0 TO STARTING SECTOR
2720  013666  062703  000402              ADD     #258.,R3                  ;ADJUST ONE SECTOR
2721  013672  005403                      NEG     R3                ;GET THE NEAGTIVE WORD COUNT
2722  013674  010337  005614              MOV     R3,FMTDPB+4       ;LOAD THE WORD CTR INTO DPB
2723  013700  000432                      BR      WRTRKE            ;LOAD THE STARTING ADDRESS
2724  013702  012737  177777  005536  WRTRKF:   MOV     #-1,NEXT          ;ALL DONE
2725  013710  052711  100000              BIS     #BIT15,(R1)       ;MASK THE SECTOR
2726  013714  005302                      DEC     R2                ;PATCH FOR THE LAST SECTOR
2727  013716  000405                      BR      WRTRKH            ;PATCH
2728  013720  012737  177777  005536  WRTRKD:   MOV     #-1,NEXT          ;NO MORE BAD SPOT
2729  013726  013702  001364              MOV     MAXSEC,R2         ;LAST SECTOR
2730  013732  006302                WRTRKH:   ASL     R2                ;WORD INDEX
2731  013734  016203  006232              MOV     WCTBL(R2),R3      ;WORD CTR FOR THE ENDING SECTOR
2732  013740  113702  005620              MOVB    FMTDPB+10,R2      ;STARTING ADDRESS
2733  013744  006302                      ASL     R2                ;WORD INDEX
2734  013746  166203  006232              SUB     WCTBL(R2),R3      ;WORD CTR FOR THE STARTING SECTOR
2735  013752  062703  000402              ADD     #258.,R3          ;ADJUST ONE SECTOR
2736  013756  005403                      NEG     R3                ;NEGATIVE WORD COUNT
2737  013760  010337  005614              MOV     R3,FMTDPB+4       ;LOAD THE WORD COUNT INTO DPB
2738  013764  000240                      NOP                               ;DONE
2739  013766  113702  005620        WRTRKE:   MOVB    FMTDPB+10,R2      ;STARTING ADDRESS
2740  013772  006302                      ASL     R2                ;LOCATE THE BUFFER ADDRESS
2741  013774  016237  006132  005616      MOV     ADRTBL(R2),FMTDPB+6       ;BUFFER ADDRESS
2742  014002  112737  000163  005612  WRTRK2:   MOVB    #WRTHD,FMTDPB+2   ;SET WRITE HEADER & DATA COMMAND IN TBL
2743  014010  012737  014010  001124      MOV     #.,$LPERR         ;SET UP LOOP ON ERROR ADDRESS
2744  014016  012706  001100              MOV     #STACK,SP         ;LOAD STACK POINT
2745  014022  004037  027410        1$:       JSR     R0,RM03           ;GO FORMAT A TRACK
2746  014026  005610                      FMTDPB                    ;ADRS OF PARAMETERS - TBL
2747  014030  000774                      BR      1$                ;WAIT FOR QUEUE IF FULL
2748  014032  005737  005626        2$:       TST     FMTDPB+16         ;WAIT FOR COMMAND TO COMPLETE
2749  014036  001775                      BEQ     2$                ;BRANCH IF NOT DONE
2750  014040  100024                      BPL     4$                ;BRANCH IF NO ERROR
2751  014042  012737  014070  001174      MOV     #3$,$ESCAPE       ;;ESCAPE TO 3$ ON ERROR
2752  014050  004737  016144              JSR     PC,ERINDX         ;SEE WHICH ERROR
2753  014054  104010                      ERROR   10                ;DRIVE OFFLINE
2754  014056  104011                      ERROR   11                ;PERSISTENT DRIVE UNSAFE ERROR
2755  014060  104012                      ERROR   12                ;UNCORRECTABLE MASSBUS PARITY ERROR
2756  014062  104013                      ERROR   13                ;SOFTWARE TIMEOUT
2757  014064  104014                      ERROR   14                ;DRIVE UNSAFE ERROR
2758  014066  104015                      ERROR   15                ;DRIVE/CONTROLLER ERROR DURING WRITE
2759  014070  004737  016244        3$:       JSR     PC,LOP.CK         ;LOOP ON THE ERROR ?
2760  014074  022737  000003  001346      CMP     #3,RETRY          ;ERROR RETRY LIMIT ?
2761  014102  001403                      BEQ     4$                ;BR IF REACHED
2762  014104  005237  001346              INC     RETRY             ;COUNT THE ERROR
2763  014110  000744                      BR      1$                ;TRY AGAIN
2764  014112  005037  001174        4$:       CLR     $ESCAPE           ;CLEAR ERROR ESCAPE ADDRESS
2765  014116  005037  001346              CLR     RETRY             ;CLEAR THE RETRY COUNTER
2766
2767                                  ;CHECK THE TRACK JUST WRITTEN
```

```
2768
2769  014122  112737  000153  005612  CKTRK:  MOVB    #WCKHD,FMTDPB+2 ;SET WRITE CHECK HEADER & DATA COMMAND IN TBL
2770  014130  012737  014130  001124          MOV     #.,$LPERR       ;SETUP LOOP ON ERROR ADDRESS
2771  014136  012706  001100                  MOV     #STACK,SP       ;LOAD STACK POINTER
2772  014142  004037  027410          1$:     JSR     R0,RM03         ;GO CHECK THE TRACK JUST FORMATTED
2773  014146  005610                          FMTDPB                  ;ADRS OF PARAMETERS - TBL
2774  014150  000774                          BR      1$              ;WAIT FOR QUEUE IF FULL
2775  014152  005737  005626          2$:     TST     FMTDPB+16       ;WAIT FOR COMMAND TO COMPLETE
2776  014156  001775                          BEQ     2$              ;BRANCH IF NOT DONE
2777  014160  100131                          BPL     8$              ;BRANCH IF NO ERROR
2778  014162  113737  005546  001350          MOVB    RM.REG+RMDA,SAVSEC  ;GET THE SECTOR ADDRESS
2779  014170  001005                          BNE     3$              ;BRANCH IF NOT SECTOR 0
2780  014172  013737  001364  001350          MOV     MAXSEC,SAVSEC   ;RESTORE TO LAST SECTOR +1
2781  014200  005237  001350                  INC     SAVSEC          ;INCREMENT TO LAST SECTOR +1
2782  014204  005337  001350          3$:     DEC     SAVSEC          ;ADJUST SECTOR TO THE ONE THAT FAILED
2783  014210  012737  014512  001174          MOV     #10$,$ESCAPE    ;ESCAPE TO 10$ ON ERROR
2784  014216  004737  016144                  JSR     PC,ERINDX       ;SEE WHICH ERROR
2785  014222  104010                          ERROR   10              ;DRIVE OFFLINE
2786  014224  104011                          ERROR   11              ;PERSISTENT DRIVE UNSAFE ERROR
2787  014226  104012                          ERROR   12              ;UNCORRECTABLE MASSBUS PARITY ERROR
2788  014230  104013                          ERROR   13              ;SOFTWARE TIMEOUT
2789  014232  104014                          ERROR   14              ;DRIVE UNSAFE ERROR
2790  014234  032737  040000  005550          BIT     #WCE,RM.REG+RMCS2   ;WRITE CHECK ERROR ?
2791  014242  001005                          BNE     4$              ;BR IF SET
2792  014244  033727  005554                  BIT     RM.REG+RMER1,(PC)+  ;CHECK FOR DATA ERRORS
2793  014250  130620                          .WORD   DCK!OPI!DTE!HCRC!HCE!FER  ;DATA ERROR BITS
2794  014252  001001                          BNE     4$              ;BR IF SET
2795  014254  104016                          ERROR   16              ;CONTROLLER/DRIVE ERROR DURING WRITE CHECK
2796  014256  005737  001314          4$:     TST     SOFSW           ;RETRYING THE SECTOR ?
2797  014262  001413                          BEQ     5$              ;BR IF NOT
2798  014264  012737  014536  001174          MOV     #11$,$ESCAPE    ;ESCAPE TO 11$ ON ERROR
2799  014272  004737  017532                  JSR     PC,RECORD       ;RECORD THE BAD SPOT
2800  014276  012737  100000  001412          MOV     #BIT15,HEADX    ;RESET THE USER BIT
2801  014304  004737  020354                  JSR     PC,RESET        ;RESET THE HEADER BITS
2802  014310  104017                          ERROR   17              ;SECTOR NOT ACCEPTABLE
2803  014312  005037  001174          5$:     CLR     $ESCAPE         ;RESET ESCAPE ADDRESS
2804  014316  032737  040000  005552          BIT     #ERR,RM.REG+RMDS    ;DATA ERROR ?
2805  014324  001002                          BNE     6$              ;BR IF IT IS
2806  014326  104024                          ERROR   24              ;WRITE CHECK ERROR
2807  014330  000401                          BR      7$              ;CONTINUE
2808  014332  104020                  6$:     ERROR   20              ;DATA ERROR DURING WRITE CHECK
2809  014334  013701  001350          7$:     MOV     SAVSEC,R1       ;FAILING SECTOR ADDRESS
2810  014340  113737  001350  005620          MOVB    SAVSEC,FMTDPB+10    ;SECTOR ADDRESS
2811  014346  006301                          ASL     R1              ;SETUP INDEX TO ADDRESS WORDS
2812  014350  016137  006132  005616          MOV     ADRTBL(R1),FMTDPB+6 ;BUFFER ADDRESS FOR FAILING SECTOR
2813  014356  013701  001364                  MOV     MAXSEC,R1       ;R1 ENDING SECTOR #
2814  014362  005737  005536                  TST     NEXT            ;WHOLE TRACK DONE ?
2815  014366  100404                          BMI     13$             ;BRANCH IF IT IS
2816  014370  013701  005536                  MOV     NEXT,R1         ;LOAD CURRENT ENDING SECTOR
2817  014374  162701  000002                  SUB     #2,R1           ;ADJUST TWO SECTORS
2818  014400  006301                  13$:    ASL     R1              ;WORD INDEX
2819  014402  016102  006232                  MOV     WCTBL(R1),R2    ;WORDCTR FOR ENDING SECTOR
2820  014406  113701  005620                  MOVB    FMTDPB+10,R1    ;STARTING ADDRESS
2821  014412  006301                          ASL     R1              ;WORD COUNT
2822  014414  166102  006232                  SUB     WCTBL(R1),R2    ;WORD CTR FOR STARTING SECTOR
2823  014420  005402                          NEG     R2
```

```
2824  014422  010237  001352              MOV     R2,SAVWC        ;SAVE THE WORD COUNT
2825  014426  012737  177376  005614      MOV     #-258.,FMTDPB+4 ;WORD COUNT FOR 1 SECTOR
2826  014434  005237  001314              INC     SOFSW           ;INDICATE THAT A RETRY IS IN PROGRESS
2827  014440  000137  014002              JMP     WRTRK2          ;REFORMAT ERROR SECTOR
2828  014444  005737  001314      8$:     TST     SOFSW           ;RETRY IN PROGRESS ?
2829  014450  001432                      BEQ     11$             ;BR IF NOT
2830  014452  022737  000002  001314      CMP     #2,SOFSW        ;SEE IF LAST RETRY
2831  014460  001403                      BEQ     9$              ;BR IF IT IS
2832  014462  005237  001314              INC     SOFSW           ;INCREMENT RETRY COUNT
2833  014466  000625                      BR      1$              ;READ AGAIN
2834  014470  123737  001364  005620  9$: CMPB    MAXSEC,FMTDPB+10  ;SEE IF LAST SECTOR ON THE TRACK
2835  014476  001417                      BEQ     11$             ;BR IF IT IS
2836  014500  004737  016576              JSR     PC,SCAWC        ;SETUP TO CHECK REMAINING SECTORS ON THE TRACK
2837  014504  005037  001314              CLR     SOFSW           ;CLEAR RETRY COUNTER
2838  014510  000614                      BR      1$              ;FINISH CHECKING THE TRACK
2839  014512  004737  016244      10$:    JSR     PC,LOP.CK       ;CHECK FOR LOOP ON ERROR
2840  014516  022737  000003  001346      CMP     #3,RETRY        ;ERROR RETRY REACHED ?
2841  014524  001404                      BEQ     11$             ;BR IF IT IS
2842  014526  005237  001346              INC     RETRY           ;COUNT THE RETRY
2843  014532  000137  014142              JMP     1$              ;DO THE WRITE CHECK AGAIN
2844  014536  005037  001174      11$:    CLR     $ESCAPE         ;CLEAR THE ERROR RETURN ESCAPE ADDRESS
2845  014542  005037  001346              CLR     RETRY           ;CLEAR THE RETRY COUNTER
2846  014546  005037  001314              CLR     SOFSW           ;CLEAR THE SOFTWARE RETRY COUNT
2847  014552  005737  005536              TST     NEXT            ;WHOLE TRACK DONE ?
2848  014556  100402                      BMI     .+6             ;BRANCH IF IT IS
2849  014560  000137  013444              JMP     WRTRKA
2850  014564  032777  000002  164362  HSET: BIT     #BIT1,@SWR      ;LOOP ON CURRENT TRACK ?
2851  014572  001402                      BEQ     12$             ;BR IF NOT
2852  014574  000137  013046              JMP     WRTRKX          ;START AGAIN ON THE SAME TRACK
2853                                                              ;RESET THE HEADER WORD OF ALL MFG
2854                                                              ;DEFINED BAD SECTOR
2855  014600  012737  040000  001412  12$: MOV     #BIT14,HEADX    ;RESET THE MFG BIT OF THE 1ST HEADER WORD
2856  014606  012704  005434              MOV     #SECCU,R4       ;TABLE FOR BAD SECTOR ON CURRENT TRACK
2857  014612  022714  177777      14$:    CMP     #-1,(R4)        ;END OF TABLE ?
2858  014616  001414                      BEQ     16$             ;BRANCH IF SO
2859  014620  032714  040000              BIT     #BIT14,(R4)     ;HAS THE SECTOR BE ASSCESSED ?
2860  014624  001006                      BNE     15$             ;BRANCH IF SO
2861  014626  052714  040000              BIS     #BIT14,(R4)     ;MASK THE SECTOR TABLE
2862  014632  111437  001350              MOVB    (R4),SAVSEC     ;SECTOR ADDRESS
2863  014636  004737  020354              JSR     PC,RESET        ;RESET THE HEADER
2864  014642  062704  000002      15$:    ADD     #2,R4           ;INCREMENT THE TABLE POINTER
2865  014646  000761                      BR      14$             ;LOOPING BACK
2866  014650  000240              16$:    NOP                     ;ALL DONE
2867
2868  014652  004737  016402      WRTRKZ: JSR     PC,TRKTST       ;GET UPDATED TRACK AND CYLINDER ADDRESSES
2869  014656  000402                      BR      HDREAD          ;RETURN HERE IF DONE - DO QUICK CHECK OF DISK
2870  014660  000137  013046              JMP     WRTRKX          ;CONTINUE WITH THE FORMAT
2871
2872                                      ;THIS CODE MAKES SURE EACH CYLINDER FORMATTED CONTAINS THE
2873                                      ;PROPER CYLINDER ADDRESS. THE PROGRAM IS LOOKING FOR
2874                                      ;POSSIBLE POSITIONER ERRORS THAT MAY HAVE OCCURRED DURING THE FORMAT.
2875
2876  014664  005737  001316      HDREAD: TST     MODE            ;VERIFY MODE ?
2877  014670  001022                      BNE     9$              ;NO
2878  014672  022737  066666  001340      CMP     #066666,PATA    ;PATTERN HAS BEEN ROTATED TO WORST CASE ?
2879  014700  001002                      BNE     .+6
```

```
2880  014702  000137  015324                JMP     $EOP            ;EXIT
2881  014706  000241                       CLC                     ;CLEAR THE CARRY BIT
2882  014710  006037  001340                ROR     PATA            ;ROTATE PATTERN
2883  014714  103003                       BCC     .+10            ;SET SIGN BIT IF CARRY=1
2884  014716  052737  100000  001340        BIS     #BIT15,PATA     ;FROM BIT 0
2885  014724  013737  001340  001342        MOV     PATA,PATB       ;SET BOTH PATTERN
2886  014732  000137  012470                JMP     CKADRS          ;CONTINUE THE VERIFY OPERATION
2887  014736  005037  001360        9$:     CLR     HEDERR          ;CLEAR HEADER CHECK ERROR INDICATOR
2888  014742  004737  016312                JSR     PC,SETTBL       ;GO SET UP DRIVER TABLE
2889  014746  004737  016624                JSR     PC,SETHDR       ;GO SET UP HEADERS IN CORE
2890  014752  013737  001322  001344        MOV     BEGCYL,CYLCK    ;USE 'BEGCYL' FOR FORMAT VERIFICATION
2891  014760  012737  177776  005614        MOV     #-2,FMTDPB+4    ;SET UP WORD COUNT
2892  014766  012737  043736  005616        MOV     #RBUF,FMTDPB+6  ;SET UP BUFFER ADRS
2893  014774  005037  005620                CLR     FMTDPB+10       ;CLEAR THE SECTOR & TRACK ADDRESS FIELD
2894  015000  013737  001322  005622        MOV     BEGCYL,FMTDPB+12  ;SETUP THE CYLINDER FIELD
2895  015006  112737  000173  005612        MOVB    #RDHD,FMTDPB+2  ;SET UP READ HEADER & DATA COMMAND
2896  015014  012737  015014  001124        MOV     #.,$LPERR       ;SETUP LOOP ON ERROR ADDRESS
2897  015022  012706  001100                MOV     #STACK,SP       ;LOAD STACK POINTER
2898  015026  004037  027410        1$:     JSR     R0,RM03         ;GO READ HEADER
2899  015032  005610                       FMTDPB                  ;ADRS OF PARAMETER TBL
2900  015034  000774                       BR      1$              ;WAIT IF QUEUE IS FULL
2901  015036  005737  005626        2$:     TST     FMTDPB+16       ;WAIT FOR COMMAND TO COMPLETE
2902  015042  001775                       BEQ     2$              ;BRANCH IF NOT DONE
2903  015044  100033                       BPL     4$              ;BR IF NOT ERROR
2904  015046  012737  015154  001174        MOV     #5$,$ESCAPE     ;ESCAPE TO 5$ ON ERROR
2905  015054  004737  016144                JSR     PC,ERINDX       ;SEE WHICH ERROR
2906  015060  104010                       ERROR   10              ;DRIVE OFFLINE
2907  015062  104011                       ERROR   11              ;PERSISTENT DRIVE UNSAFE ERROR
2908  015064  104012                       ERROR   12              ;UNCORRECTABLE MASSBUS PARITY ERROR
2909  015066  104013                       ERROR   13              ;SOFTWARE TIMEOUT
2910  015070  104014                       ERROR   14              ;DRIVE UNSAFE ERROR
2911  015072  032737  177577  005626        BIT     #^C<HCE>,FMTDPB+16  ;IS ONLY ERROR A HEADER COMPARE ERROR ?
2912  015100  001410                       BEQ     3$              ;BR IF YES
2913  015102  005737  005554                TST     RM.REG+14       ;ANY DRIVE ERROR
2914  015106  001412                       BEQ     4$              ;BRANCH IF NONE
2915  015110  032737  100000  005602        BIT     #BIT15,RM.REG+42  ;BAD SPOT ERROR ?
2916  015116  001006                       BNE     4$              ;BRANCH IF IT IS
2917  015120  104021                       ERROR   21              ;CONTROLLER/DRIVE ERROR VERIFYING HEADERS
2918  015122  005037  001174        3$:     CLR     $ESCAPE         ;CLEAR THE ERROR ESCAPE ADDRESS
2919  015126  104022                       ERROR   22              ;HEADER COMPARE ERROR VERIFYING HEADERS
2920  015130  004737  016244                JSR     PC,LOP.CK       ;CHECK FOR LOOP ON ERROR
2921  015134  052737  140000  043736  4$:     BIS     #BIT15!BIT14,RBUF  ;SET BIT15 AND BIT 14
2922  015142  023737  043736  043746        CMP     RBUF,BUFP       ;SEE IF CYL READ EQUALS CYL EXPECTED
2923  015150  001403                       BEQ     6$              ;BR IF CYL CORRECT
2924  015152  104023                       ERROR   23              ;CYLINDER NOT CORRECT
2925  015154  004737  016244        5$:     JSR     PC,LOP.CK       ;CHECK FOR LOOP ON ERROR
2926  015160  023737  001320  001344  6$:     CMP     ENDCYL,CYLCK    ;SEE IF LAST CYLINDER
2927  015166  001002                       BNE     7$              ;BR IF NOT FINISHED
2928  015170  000137  015324                JMP     $EOP            ;END OF FORMAT
2929  015174  005237  001344        7$:     INC     CYLCK           ;INCREMENT CYLINDER ADDRESS BEING CHECKED
2930  015200  022737  001466  001344        CMP     #622.,CYLCK     ;LAST CYLINDER ?
2931  015206  003041                       BGT     8$              ;NO
2932  015210  122737  000004  005621        CMPB    #4,FMTDPB+11    ;LAST TRACK ?
2933  015216  003035                       BGT     8$              ;NO, THEN BRANCH
2934  015220  005237  001344                INC     CYLCK           ;INCREMENT CYLINDER ADDRESS
2935  015224  023737  001320  001344        CMP     ENDCYL,CYLCK    ;LAST CYLINDER ?
```

```
2936  015232  001752                  BEQ     6$                      ;EXIT IF IT IS
2937  015234  005037  001344          CLR     CYLCK                   ;YES LAST CYLINDER
2938  015240  005037  005622          CLR     FMTDPB+12               ;RESER CYLINDER # TO 0
2939  015244  005037  005620          CLR     FMTDPB+10               ;RESET TRACK AND SECTOR #
2940  015250  005037  005626          CLR     FMTDPB+16               ;RESET CYLINDER #
2941  015254  013746  001322          MOV     BEGCYL,-(SP)    ;SAVE BEGCYL
2942  015260  013746  001326          MOV     BEGTRK,-(SP)    ;SAVE BEGTRK
2943  015264  005037  001326          CLR     BEGTRK
2944  015270  005037  001322          CLR     BEGCYL
2945  015274  004737  016624          JSR     PC,SETHDR               ;SET UP BUFFER
2946  015300  012637  001326          MOV     (SP)+,BEGTRK    ;RESTORE BEGTRK
2947  015304  012637  001322          MOV     (SP)+,BEGCYL    ;RESTORE BEGCYL
2948  015310  000646                  BR      1$
2949  015312  004737  016712  8$:     JSR     PC,UPDACY               ;SET UP FOR NEXT CYL
2950  015316  005237  005622          INC     FMTDPB+12               ;ADVANCE TO NEXT CYL #
2951  015322  000641                  BR      1$                      ;GO READ NEXT HEADER
2952
2953                                  .SBTTL  END OF PASS ROUTINE
2954
2955                                  ;;*****************************************************************
2956                                  ;*INCREMENT THE PASS NUMBER ($PASS)
2957                                  ;*IF THERES A MONITOR GO TO IT
2958                                  ;*IF THERE ISN'T JUMP TO DONE
2959
2960  015324                          $EOP:
2961  015324  005737  001410  5$:     TST     EDIT                    ;TEST LAST TRACK ?
2962  015330  001536                  BEQ     11$                     ;BRANCH IF NOT
2963  015332  005037  001346          CLR     RETRY                   ;CLEAR RETRY COUNT
2964                          ;       JSR     PC,SORT2                ;SORT THE USTAB
2965  015336  004537  017646          JSR     R5,SORT2                ;SORT THE USTAB IN ASCENDING FORM
2966  015342  003434                  USTAB+8.                        ;TABLE ENTRY POINT
2967  015344  004737  020006          JSR     PC,TEXT                 ;SET UP THE BUFFER
2968  015350  112737  000020  005611  MOVB    #20,FMTDPB+1            ;16 BIT MODE
2969  015356  112737  000143  005612  MOVB    #SETFMT,FMTDPB+2                ;SET FORMAT COMMAND
2970  015364  004037  027410  6$:     JSR     R0,RM03                 ;CALL THE DRIVER
2971  015370  005610                  FMTDPB
2972  015372  000774                  BR      6$                      ;LOOP IF QUEQU FAILS
2973  015374  005737  005626  7$:     TST     FMTDPB+16               ;ALL DONE ?
2974  015400  001775                  BEQ     7$                      ;BRANCH IF NOT
2975  015402  012737  001466  005622  MOV     #822.,FMTDPB+12 ;CYLINDER ADDRESS
2976  015410  112737  000004  005621  MOVB    #4,FMTDPB+11    ;TRACK NUMBER
2977  015416  105037  005620          CLRB    FMTDPB+10               ;SECTOR 0
2978  015422  012737  157700  005614  MOV     #-<32.*258.>,FMTDPB+4 ;WORD COUNT
2979  015430  012737  043746  005616  MOV     #BUFP,FMTDPB+6  ;BUFFER ADDRESS
2980  015436  112737  000163  005612  MOVB    #WRTHD,FMTDPB+2 ;WRITE HEAD AND DATA COMMAND
2981  015444  004037  027410  8$:     JSR     R0,RM03                 ;CALL THE DRIVER
2982  015450  005610                  FMTDPB
2983  015452  000774                  BR      8$                      ;LOOP IF QUEUE FAILS
2984  015454  005737  005626  9$:     TST     FMTDPB+16               ;ALL DONE ?
2985  015460  001775                  BEQ     9$                      ;BRANCH IF NOT
2986  015462  100061                  BPL     11$                     ;EXIT IF NO ERROR
2987  015464  012737  015512  001174  MOV     #22$,$ESCAPE    ;ERROR ESCAPE ADDRESS
2988  015472  004737  016144          JSR     PC,ERINDX               ;ERROR INDICATOR RT.
2989  015476  104010                  ERROR   10
2990  015500  104011                  ERROR   11
2991  015502  104012                  ERROR   12
```

```
2992  015504  104013                          ERROR   13
2993  015506  104014                          ERROR   14
2994  015510  104015                          ERROR   15
2995  015512  000240                  22$:    NOP
2996  015514  005237  001346                  INC     RETRY                   ;
2997  015520  022737  000003  001346          CMP     #3,RETRY                ;OVER 3 TIMES ?
2998  015526  101346                          BHI     8$                      ;BRANCH IF IT IS
2999  015530  000137  016126                  JMP     REJCT3                  ;WRITE THE LAST TRACK FAILS THREE TIMES
3000  015534  122737  000037  005620          CMPB    #31.,FMTDPB+10          ;LAST SECTOR ?
3001  015542  003431                          BLE     11$                     ;BRANCH IF ALL SET
3002  015544  113737  005546  001350          MOVB    RM.REG+RMDA,SAVSEC      ;FOUND THE FALLING SPOT
3003  015552  001003                          BNE     10$                     ;BRANCH IF NOT 0
3004  015554  012737  000037  001350          MOV     #31.,SAVSEC             ;THE LAST SECTOR IS A BAD SECTOR
3005  015562  113737  001350  005620  10$:    MOVB    SAVSEC,FMTDPB+10                ;LOAD THE NEW STARTING SECTOR
3006  015570  113701  001350                  MOVB    SAVSEC,R1                       ;COOK THE WORD CTR AND BUFFER
3007  015574  006301                          ASL     R1                      ;WORD INDEX
3008  015576  016137  006132  005616          MOV     ADRTBL(R1),FMTDPB+6     ;BUFFER ADDRESS
3009  015604  012737  157700  005614          MOV     #-(258.*32.),FMTDPB+4   ;WORD COUNT
3010  015612  066137  006232  005614          ADD     WCTBL(R1),FMTDPB+4      ;WORD COUNT
3011  015620  005037  001346                  CLR     RETRY                           ;RESET THE RETRY FLAG
3012  015624  000707                          BR      8$                              ;BRANCH BACK
3013  015626  005237  001216          11$:    INC     $DEVCT                  ;INCREMENT THE DEVICE COUNT
3014  015632  005737  001316                  TST     MODE                    ;FORMAT OR CHECK MODE
3015  015636  001403                          BEQ     12$                     ;BR IF CHECK
3016  015640  104401  036271                  TYPE    ,MFCMPT                 ;FORMAT COMPLETE
3017  015644  000402                          BR      13$                     ;FINISH THE MESSAGE
3018  015646  104401  036316          12$:    TYPE    ,MCCMPT                 ;CHECK COMPLETE
3019  015652  104401  036342          13$:    TYPE    ,NUMERR                 ;TOTAL ERRORS
3020  015656  013746  001126                  MOV     $ERTTL,-(SP)            ;;SAVE $ERTTL FOR TYPEOUT
3021  015662  104405                          TYPDS                           ;;GO TYPE--DECIMAL ASCII WITH SIGN
3022  015664  104401  001203                  TYPE    ,$CRLF
3023  015670  005737  001404                  TST     DEVMP                   ;ALL ASSIGNED DEVICE DONE ?
3024  015674  001402                          BEQ     14$                     ;BRANCH IF ALL DONE
3025  015676  005337  001214                  DEC     $PASS                   ;RESTORE THE PASS COUNT
3026  015702  000240                  14$:    NOP                             ;
3027
3028
3029  015704  005237  001214                  INC     $PASS                   ;;INCREMENT THE PASS NUMBER
3030  015710  042737  100000  001214          BIC     #100000,$PASS           ;;DON'T ALLOW A NEG. NUMBER
3031  015716  005327                          DEC     (PC)+                   ;;LOOP?
3032  015720  000001                  $EOPCT: .WORD   1
3033  015722  003013                          BGT     $DOAGN                  ;;YES
3034  015724  012737                          MOV     (PC)+,@(PC)+            ;;RESTORE COUNTER
3035  015726  000001                  $ENDCT: .WORD   1
3036  015730  015720                          $EOPCT
3037  015732  013700  000042          $GET42: MOV     @#42,R0                 ;;GET MONITOR ADDRESS
3038  015736  001405                          BEQ     $DOAGN                  ;;BRANCH IF NO MONITOR
3039  015740  000005                          RESET                           ;;CLEAR THE WORLD
3040  015742  004710                  $ENDAD: JSR     PC,(R0)                 ;;GO TO MONITOR
3041  015744  000240                          NOP                             ;;SAVE ROOM
3042  015746  000240                          NOP                             ;;FOR
3043  015750  000240                          NOP                             ;;ACT11
3044  015752                          $DOAGN:
3045  015752  000137                          JMP     @(PC)+                  ;;RETURN
3046  015754  015756                  $RTNAD: .WORD   DONE
3047  015756  105737  001150          DONE:   TSTB    $AUTOB                  ;RUN UNDER APT
```

```
3048  015762  001405                        BEQ     2$              ;NO
3049  015764  005737  001404                TST     DEVMP           ;ALL DRIVES ARE DONE ?
3050  015770  001427                        BEQ     4$              ;BRANCH IF ALL DONE
3051  015772  000137  007774                JMP     MSFX            ;NEXT DRIVE FROM APT DEVICE MAP
3052  015776  032777  000001  163150  2$:   BIT     #SW0,@SWR       ;SEE IF SWR 0 SET
3053  016004  001002                        BNE     1$              ;BR IF SET
3054  016006  000137  010522                JMP     M0              ;ASK FOR NEXT DRIVE
3055  016012  012706  001100          1$:   MOV     #STACK,SP       ;RESET STACK POINTER
3056  016016  005737  001316                TST     MODE            ;VERIFY MODE ?
3057  016022  001402                        BEQ     3$              ;YES
3058  016024  000137  012470                JMP     CKADRS          ;DO THE FORMAT OR CHECK AGAIN
3059  016030  012737  133331  001340  3$:   MOV     #133331,PATA    ;INITIATE PATTERN
3060  016036  012737  133331  001342        MOV     #133331,PATB    ;INITIATE PATTERN
3061  016044  000137  012470                JMP     CKADRS          ;LOOP ON CHECK
3062  016050  005237  001214          4$:   INC     $PASS           ;GARBAGE CODE FOR APT
3063  016054  005237  001212          5$:   INC     $TESTN
3064  016060  005237  001216                INC     $DEVCT
3065  016064  000773                        BR      5$
3066  016066  104401  001203          REJEC1: TYPE    ,$CRLF
3067  016072  104401  037050                TYPE    ,MESG4
3068  016076  000000                        HALT
3069  016100  000137  000200                JMP     @#200
3070  016104  104401  001203          REJCT2: TYPE    ,$CRLF
3071  016110  113737  005621  001370        MOVB    FMTDPB+11,DS.TRK  ;TRACK NUMBER FOR ERROR LOG
3072  016116  104025                        ERROR   25              ;REPORT ILLEGAL BAD SECTOR
3073  016120  000000                        HALT
3074  016122  000137  000200                JMP     @#200
3075  016126  104401  001203          REJCT3: TYPE    ,$CRLF
3076  016132  104401  040375                TYPE    ,MESG17
3077  016136  000000                        HALT
3078  016140  000137  000200                JMP     @#200
3079
3080                                  ;;*****************************************************************
3081
3082                                  .SBTTL  SUPPORT SUBROUTINES
3083
3084                                  ;;*****************************************************************
3085
3086                                  ;THIS ROUTINE DETERMINES THE ERROR TYPE
3087
3088  016144  010146                  ERINDX: MOV     R1,-(SP)        ;STORE R1
3089  016146  005001                          CLR     R1              ;CLEAR R1
3090  016150  033727  005626                  BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
3091  016154  060006                          .WORD   BIT14!BIT13!BIT2!BIT1  ;DRIVE OFFLINE
3092  016156  001025                          BNE     5$              ;BR IF OFFLINE
3093  016160  033727  005626                  BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
3094  016164  010000                          .WORD   BIT12           ;DRIVE PERSISTENTLY UNSAFE
3095  016166  001020                          BNE     4$              ;BR IF UNSAFE
3096  016170  033727  005626                  BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
3097  016174  006000                          .WORD   BIT11!BIT10     ;UNCORRECTABLE MASSBUS PARITY ERROR ?
3098  016176  001013                          BNE     3$              ;BR IF PARITY ERROR
3099  016200  033727  005626                  BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
3100  016204  001400                          .WORD   BIT09!BIT08     ;SOFTWARE TIMEOUT ?
3101  016206  001006                          BNE     2$              ;BR IF YES
3102  016210  033727  005626                  BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
3103  016214  000020                          .WORD   BIT04           ;DRIVE UNSAFE ERROR ?
```

```
3104  016216  001004                        BNE     1$              ;BR IF YES
3105  016220  005201                        INC     R1              ;INCREMENT THE RETURN INDEX
3106  016222  005201              1$:       INC     R1              ;INCREMENT THE RETURN INDEX
3107  016224  005201              2$:       INC     R1              ;INCREMENT THE RETURN INDEX
3108  016226  005201              3$:       INC     R1              ;INCREMENT THE RETURN INDEX
3109  016230  005201              4$:       INC     R1              ;INCREMENT THE RETURN INDEX
3110  016232  006301              5$:       ASL     R1              ;DOUBLE THE INCREMENT
3111  016234  060166  000002                ADD     R1,2(SP)        ;DEVELOP THE RETURN ADDRESS
3112  016240  012601                        MOV     (SP)+,R1        ;RESTORE R1
3113  016242  000207                        RTS     PC              ;RETURN
3114
3115                                  ;ROUTINE TO CHECK FOR LOOP ON ERROR
3116
3117  016244  032777  001000  162702  LOP.CK: BIT     #SW09,@SWR      ;LOOP ON ERROR ?
3118  016252  001402                        BEQ     1$              ;BR IF NOT
3119  016254  000177  162644                JMP     @$LPERR         ;GO TO THE LOOP ON ERROR ADDRESS
3120  016260  005037  001174          1$:   CLR     $ESCAPE         ;CLEAR THE ERROR ESCAPE ADDRESS
3121  016264  033727  005626                BIT     FMTDPB+16,(PC)+ ;CHECK FOR 'FATAL' ERROR
3122  016270  072002                        .WORD   BIT14!BIT13!BIT12!BIT10!BIT01  ;'FATAL' ERROR BITS
3123  016272  001004                        BNE     2$              ;BR IF NOT
3124  016274  032737  004000  005554        BIT     #WLE,RM.REG+RMER!  ;WRITE LOCK ERROR ?
3125  016302  001402                        BEQ     3$              ;BR IF NOT
3126  016304  000137  015324          2$:   JMP     $EOP            ;TERMINATE THE FORMAT
3127  016310  000207                3$:   RTS     PC              ;RETURN
3128
3129                                  ;THIS ROUTINE SETS UP THE INPUT TABLE FOR THE DRIVER ROUTINE
3130
3131  016312  113737  001220  005610  SETTBL: MOVB    DRIVE,FMTDPB    ;SET UP DRIVE #
3132  016320  105037  005613                CLRB    FMTDPB+3        ;CLEAR HIGH ORDER ADRS BITS
3133  016324  013737  001356  005614        MOV     MWC,FMTDPB+4    ;LOAD UP WORD COUNT
3134  016332  012737  043746  005616        MOV     #BUFP,FMTDPB+6  ;LOAD UP CURRENT ADRS
3135  016340  105037  005620                CLRB    FMTDPB+10       ;SET SECTOR TO ZERO
3136  016344  113737  001326  005621        MOVB    BEGTRK,FMTDPB+11  ;SET UP STARTING TRK ADRS
3137  016352  013737  001322  005622        MOV     BEGCYL,FMTDPB+12  ;SET UP STARTING CYL
3138  016360  005037  005626                CLR     FMTDPB+16       ;CLEAR RM03 STATUS
3139  016364  013737  001326  001334        MOV     BEGTRK,TRKCNT   ;SET UP PARTIAL CYL TRACK COUNT
3140  016372  013737  001330  001332        MOV     TTRKS,TTRKSC    ;SET UP TOTAL TRACKS COUNTER
3141  016400  000207                        RTS     PC              ;RETURN FROM SETUP
3142
3143                                  ;THIS ROUTINE CONTROLS THE DISK ADDRESSING AND TOTAL TRK COUNT
3144                                  ;IT IS ENTERED AFTER EVERY TRK OPERATION
3145
3146  016402  005337  001332          TRKTST: DEC     TTRKSC          ;SEE IF LAST TRACK
3147  016406  001472                        BEQ     3$              ;BRANCH IF LAST TRACK
3148  016410  022737  000004  001334        CMP     #4.,TRKCNT      ;IS THIS THE LAST TRACK IN THE CYLINDER ?
3149  016416  001423                        BEQ     1$              ;BRANCH IF SO
3150  016420  005237  001334                INC     TRKCNT          ;COUNT UP TRACK WITHIN CYLINDER
3151  016424  105237  005621                INCB    FMTDPB+11       ;INCREMENT TRACK NUMBER
3152  016430  022737  001466  005622        CMP     #822.,FMTDPB+12 ;LAST CYLINDER AND LAST TRACK ?
3153  016436  003010                        BGT     5$              ;NO
3154  016440  122737  000004  005621        CMPB    #4,FMTDPB+11    ;LAST TRACK ?
3155  016446  003004                        BGT     5$              ;NO
3156  016450  005337  001332                DEC     TTRKSC          ;DECREMENT ONE TRACK
3157  016454  001447                        BEQ     3$              ;BRANCH IF LAST TRACK
3158  016456  000403                        BR      1$              ;DON'T DESTORY THE LAST TRACK OF CYL822
3159  016460  004737  016770          5$:   JSR     PC,UPDATK       ;UPDATE TRACK ADDRESS IN BUFFER
```

```
3160  016464  000441                          BR      2$              ;EXIT
3161  016466  005037  001334          1$:     CLR     TRKCNT          ;CLEAR TRACK COUNT FOR NEXT CYLINDER
3162  016472  105037  005621                  CLRB    FMTDPB+11       ;RESET TRACK ADDRESS TO 0
3163  016476  005237  005622                  INC     FMTDPB+12       ;UPDATE CYLINDER NUMBER
3164  016502  022737  001467  005622          CMP     #823.,FMTDPB+12 ;LAST CYLINDER REACHED ?
3165  016510  003025                          BGT     4$              ;NO
3166  016512  013746  001322                  MOV     BEGCYL,-(SP)    ;SAVE BEGCYL
3167  016516  013746  001326                  MOV     BEGTRK,-(SP)    ;SAVE BEGTRK
3168  016522  005037  001322                  CLR     BEGCYL          ;RESET THE STARTING CYL
3169  016526  005037  001326                  CLR     BEGTRK          ;RESET THE START TRACK
3170  016532  004737  016624                  JSR     PC,SETHDR       ;INITIATE HEAD WORDS
3171  016536  005037  005620                  CLR     FMTDPB+10       ;CLEAR TRACK AND SECTOR
3172  016542  005037  005622                  CLR     FMTDPB+12       ;CLEAR CYLINDER #
3173  016546  005037  005626                  CLR     FMTDPB+16       ;CLEAR STATUS
3174  016552  012637  001326                  MOV     (SP)+,BEGTRK    ;RESTORE BEGTRK
3175  016556  012637  001322                  MOV     (SP)+,BEGCYL    ;RESTORE BEGCYL
3176  016562  000402                          BR      2$
3177  016564  004737  016712          4$:     JSR     PC,UPDACY       ;UPDATE CYLINDER NUMBER IN BUFFER
3178  016570  062716  000002          2$:     ADD     #2,(SP)         ;ADD TWO TO RETURN ADRS
3179  016574  000207                  3$:     RTS     PC              ;RETURN
3180
3181                                  ;THIS ROUTINE SETS UP WC & BA FOR REMAINING SECTORS
3182                                  ;AFTER A WRITE CHECK ERROR
3183
3184  016576  013737  001352  005614  SCAWC:  MOV     SAVWC,FMTDPB+4  ;SET UP WC FOR REMAINING SECTORS
3185  016604  062737  001004  005616          ADD     #516.,FMTDPB+6  ;SET UP BA FOR REMAINING SECTORS
3186  016612  005237  005620                  INC     FMTDPB+10       ;ADVANCE TO NEXT SECTOR IN TBL
3187  016616  005037  001314                  CLR     SOFSW           ;RESET RETRY COUNTER
3188  016622  000207                          RTS     PC              ;RETURN TO COMPLETE TRK FORMAT
3189
3190
3191                                  ;THIS ROUTINE SETS UP THE CYLINDER ADRS, FORMAT BIT, TRACK AND
3192                                  ;SECTOR ADRS IN MEMORY WITH THE STARTING CYL - TRK INFORMATION
3193
3194  016624  013701  001364  SETHDR: MOV     MAXSEC,R1       ;SET UP SECTOR COUNT
3195  016630  012737  000001  001212          MOV     #1,$TESTN       ;GARBAGE CODE FOR APT
3196  016636  012700  043746                  MOV     #BUFP,R0        ;SET UP HEADER POINTER IN R0
3197  016642  013702  001322                  MOV     BEGCYL,R2       ;PUT STARTING CYL # IN R2
3198  016646  052702  140000                  BIS     #140000,R2      ;BIT 15,BIT 14 ARE SET FOR 144 SPC 3/14/77
3199  016652  013703  001326                  MOV     BEGTRK,R3       ;PUT STARTING TRK # IN R3
3200  016656  000303                          SWAB    R3              ;JUSTIFY TRACK ADRS
3201  016660  005737  001362                  TST     SEC30           ;SEE IF 30 OR 32 SECTOR MODE
3202  016664  001402                          BEQ     1$              ;BR IF 30 SECTOR MODE
3203  016666  052702  010000                  BIS     #10000,R2       ;SET THE 32 SECTOR FORMAT BIT
3204  016672  010220                  1$:     MOV     R2,(R0)+        ;WRITE IN HEADER AREA OF CORE THE CYL ADRS
3205  016674  010320                          MOV     R3,(R0)+        ;WRITE IN HEADER AREA OF CORE THE TRK ADRS
3206  016676  062700  001000                  ADD     #512.,R0        ;SET UP FOR NEXT HEADER
3207  016702  005203                          INC     R3              ;UPDATE SECTOR ADRS FOR NEXT HEADER
3208  016704  005301                          DEC     R1              ;MAINTAIN COUNT OF SECTORS
3209  016706  002371                          BGE     1$              ;BRANCH IF NOT LAST SECTOR
3210  016710  000207                          RTS     PC              ;EXIT - HEADERS ARE LOADED INTO CORE
3211
3212                                  ;THIS ROUTINE UPDATES THE CYLINDER ADRS OF THE HEADER WORDS IN CORE
3213
3214  016712  013701  001364  UPDACY: MOV     MAXSEC,R1       ;SET UP SECTOR COUNT
3215  016716  012700  043746                  MOV     #BUFP,R0        ;SET UP HEADER POINTER IN R0
```

```
3216  016722  010246                        MOV     R2,-(SP)        ;SAVE R2
3217  016724  005220                1$:     INC     (R0)+           ;INCREMENT FOR NEXT CYLINDER
3218  016726  042720  177400                BIC     #177400,(R0)+   ;RESET TRK ADRS TO 0
3219  016732  012702  001000                MOV     #512.,R2        ;DATA FIELD LENGTH
3220  016736  013720  001340        2$:     MOV     PATA,(R0)+      ;FILL BUFFER
3221  016742  013720  001342                MOV     PATB,(R0)+      ;FILL BUFFER
3222  016746  162702  000004                SUB     #4,R2           ;END OF DATA FIELD ?
3223  016752  001371                        BNE     2$              ;NO
3224  016754  005301                        DEC     R1              ;COUNT SECTORS
3225  016756  002362                        BGE     1$              ;BRANCH IF NOT LAST SECTOR
3226  016760  012602                        MOV     (SP)+,R2        ;RESETORE THE R2
3227  016762  005237  001212                INC     $TESTN          ;INCREMENT TEST NUMBER FOR EACH CYLINDER CHANGE
3228                                                                ;GARBADE CODE FOR APT
3229  016766  000207                        RTS     PC              ;EXIT
3230
3231                                ;THIS ROUTINE UPDATES THE TRACK ADRS OF THE HEADER WORDS IN CORE
3232
3233  016770  013701  001364        UPDATK: MOV     MAXSEC,R1       ;SET UP SECTOR COUNT
3234  016774  012700  043746                MOV     #BUFP,R0        ;SET UP HEADER POINTER IN R0
3235  017000  005720                        TST     (R0)+           ;POINT HEADER POINTER TO TRK - SEC ADRS
3236  017002  062710  000400        1$:     ADD     #400,(R0)       ;INDEX TRK ADRS
3237  017006  062700  001004                ADD     #516.,R0        ;SET UP FOR NEXT HEADER
3238  017012  005301                        DEC     R1              ;COUNT SECTORS
3239  017014  002372                        BGE     1$              ;BRANCH IF NOT LAST SECTOR
3240  017016  000207                        RTS     PC              ;EXIT
3241
3242                                ;ROUTINE TO CHECK FOR KW11-L OR KW11-P CLOCKS
3243
3244  017020  012737  017076  000004  ST.CLK: MOV     #STCLK1,@#ERRVEC  ;SET UP VECTOR FOR P CLK
3245  017026  005037  000006                CLR     @#ERRVEC+2      ;NEW PSW
3246  017032  005777  162240                TST     @$LKCSR         ;CHECK FOR KW11-P
3247  017036  013746  001302                MOV     $LPVEC,-(SP)    ;VECTOR ADDRESS
3248  017042  012776  017226  000000        MOV     #CLOCK,@(SP)    ;SET UP KW11-P VECTOR
3249  017050  062716  000002                ADD     #2,(SP)         ;POINT TO PSW
3250  017054  012736  000300                MOV     #PR6,@(SP)+     ;PSW - PRI 6
3251  017060  012777  177777  162212        MOV     #-1,@$LKCSB     ;LOAD COUNTER BUFFER
3252  017066  012777  000135  162202        MOV     #135,@$LKCSR    ;SET CLK - CNT UP
3253  017074  000423                        BR      STCLK3
3254  017076  062706  000004        STCLK1: ADD     #4,SP           ;RESTORE THE STACK POINTER
3255  017102  012737  017146  000004        MOV     #STCLK2,@#ERRVEC        ;CHANGE ERROR VECTOR
3256  017110  005777  162172                TST     @$LKS           ;LOOK FOR KW11-L
3257  017114  013746  001310                MOV     $LLVEC,-(SP)    ;KW11-L VECTOR ADDRESS
3258  017120  012776  017226  000000        MOV     #CLOCK,@(SP)    ;SET UP KW11-L VECTOR
3259  017126  062716  000002                ADD     #2,(SP)         ;INCREMENT VECTOR ADDRESS
3260  017132  012736  000300                MOV     #PR6,@(SP)+     ;PSW - PRI 6
3261  017136  012777  000100  162142        MOV     #100,@$LKS      ;SET KW11-L INTERRUPT ENABLE
3262  017144  000402                        BR      STCLK3
3263  017146  062706  000004        STCLK2: ADD     #4,SP           ;RESTORE THE STACK POINTER
3264  017152  012737  000006  000004  STCLK3: MOV     #6,@#ERRVEC     ;RESTORE THE ERROR VECTOR
3265  017160  000207                        RTS     PC
3266
3267                                ;THIS CODE PRINT THE ABORT MESSAGE AND LET O.P. RESTART
3268                                ;
3269  017162  105737  001150        RESTRT: TSTB    $AUTOB          ;RUN UNDER APT ?
3270  017166  001016                        BNE     1$              ;BRANCH IF SO
3271  017170  004737  024422                JSR     PC,$TKINT       ;ENABLE TO READ
```

```
3272  017174  104401  036667          TYPE    ,MESG2          ;IMPORPER MFG INFORMATION
3273  017200  104411                  RDLIN
3274  017202  012601                  MOV     (SP)+,R1        ;LOCATE THE READ IN LINE
3275  017204  105711                  TSTB    (R1)            ;TERMINATOR BY <CR>
3276  017206  001406                  BEQ     1$              ;BRANCH IF SO
3277  017210  121127  000131          CMPB    (R1),#'Y        ;ENTER Y ?
3278  017214  001403                  BEQ     1$              ;BRANCH IF SO
3279  017216  000000                  HALT                    ;OTHERWISE HALT
3280  017220  000137  000200          JMP     @#200           ;JUMP TO 200 IF HIT START SWITCH
3281  017224  000207          1$:     RTS     PC              ;EXIT
3282
3283                          ;THIS CODE SERVICES A CLOCK INTERRUPT EVERY 16MS
3284
3285  017226  012746  000020  CLOCK:  MOV     #16.,-(SP)      ;PUT MILLISECONDS ON THE STACK
3286  017232  004737  033056          JSR     PC,RPTMR        ;GO REPORT TIME
3287  017236  000002                  RTI                     ;RETURN AND CONTINUE
3288
3289                          ;ROUTINE TO INTERCEPT 'CONTROL C' TYPED DURING PARAMETER ENTRY TIME
3290
3291  017240  012706  001100  OENTER: MOV     #STACK,SP       ;INITIALIZE THE STACK
3292  017244  005737  026556  1$:     TST     TRNSWT          ;ALL ACTIVITY STOPPED ?
3293  017250  001375                  BNE     1$              ;BR IF NOT
3294  017252  000005                  RESET                   ;CLEAR THE BUS
3295  017254  000137  010462          JMP     M1B             ;START AGAIN WITH DRIVE SELECTION
3296
3297                          ;ROUTINE TO TYPE THE PRESENT DISK ADDRESS.  ENTERED BY TYPING 'CONTROL C'
3298
3299  017260  117746  161676  NEWSVC: MOVB    @$TKB,-(SP) ;READ FROM TTY BUFFER
3300  017264  042716  177600          BIC     #^C177,(SP)     ;STRIP PARITY
3301  017270  022627  000017          CMP     (SP)+,#15.      ;CONTROL-O ?
3302  017274  001406                  BEQ     1$              ;YES
3303  017276  005777  161660          TST     @$TKB           ;CLEAR DONE BIT
3304  017302  012777  000100  161650  MOV     #100,@$TKS      ;ENABLE TTY INTERRUPT
3305  017310  000002                  RTI
3306  017312  004737  024422  1$:     JSR     PC,$TKINT       ;INITIAL VECTOR
3307  017316  005037  177776  TYPADR: CLR     PSW             ;SET PROCESSOR TO PRIORITY 0
3308  017322  012737  017240  001376  MOV     #OENTER,CNTLC   ;CHANGE 'CONTROL C' RETURN ADDRESS
3309  017330  104401  036372          TYPE    ,ADDRIS         ;'PRESENT ADDRESS IS: '
3310  017334  104401  035156          TYPE    ,C              ;'C'
3311  017340  013746  005622          MOV     FMTDPB+12,-(SP) ;PUT THE CYLINDER ADDRESS ON THE STACK
3312  017344  004737  026066          JSR     PC,$B2D         ;CONVERT IT TO DECIMAL
3313  017350  004737  026026          JSR     PC,$SUPRS       ;TYPE IT
3314  017354  104401  035166          TYPE    ,LINSP          ;SPACES
3315  017360  104401  035160          TYPE    ,T              ;'T'
3316  017364  005046                  CLR     -(SP)           ;CLEAR THE STACK
3317  017366  113716  005621          MOVB    FMTDPB+11,(SP)  ;PUT THE TRACK ADDRESS ON THE STACK
3318  017372  004737  026066          JSR     PC,$B2D         ;CONVERT IT TO DECIMAL
3319  017376  004737  026026          JSR     PC,$SUPRS       ;TYPE IT
3320  017402  104401  001203          TYPE    ,$CRLF          ;CR-LF
3321  017406  012737  017260  000060  MOV     #NEWSVC,@#TKVEC ;RESTORE ENTRANCE TO THIS ROUTINE
3322  017414  000002                  RTI                     ;RETURN
3323
3324                          ;PARAMETER ENTRY ROUTINE
3325                          ;CALL
3326                          ;       JSR     R4,PARENT       ;THE CALLING SEQ
3327                          ;       TABLE                   ;PARAMETERS TABLE
```

```
                                        ;
                                        ;
017416  010346                  PARENT: MOV     R3,-(SP)        ;SAVE R3
                                        MOV     #TABLE,R3       ;PARAMETER TABLE ADDRESS
017420  011403                  ;       MOV     (R4),R3         ;PARAMETER TABLE ADDRESS
017422  012337  017432          1$:     MOV     (R3)+,3$        ;ADDRESS OF PARAMETER NAME
017426  001435                          BEQ     9$              ;BR IF AT END OF TABLE
017430  104401                          TYPE                    ;TYPE THE PARAMETER NAME
017432  000000                  3$:     .WORD   0               ;ADDRESS OF PARAMETER NAME TEXT
017434  012302                          MOV     (R3)+,R2        ;MAXIMUM PARAMETER VALUE
017436  012305                          MOV     (R3)+,R5        ;ADDRESS OF PARAMETER
017440  011546                          MOV     (R5),-(SP)      ;CURRENT VALUE OF PARAMETER
017442  104405                          TYPDS                   ;TYPE THE CURRENT VALUE OF THE PARAMETER
017444  104401  035152                  TYPE    ,SLASH          ;' /'
017450  104411                          RDLIN                   ;READ THE KEYBOARD
017452  012601                          MOV     (SP)+,R1        ;INPUT ASCII STRING ADDRESS
017454  004537  022360                  JSR     R5,CK.DIG       ;CHECK THE DIGIT(S)
017460  017422                          1$                      ;CARRIAGE RETURN ONLY ENTERED
017462  017522                          9$                      ;PERIOD ONLY ENTERED
017464  017500                          6$                      ;ILLEGAL INPUT
017466  017474                          5$                      ;TERMINATED WITH A CARRIAGE RETURN
017470  017500                          6$                      ;TERMINATED WITH A ","
017472  017512                          7$                      ;TERMINATED WITH A "."
017474  010215                  5$:     MOV     R2,(R5)         ;MOVE NEW VALUE TO PARAMETER LOCATION
017476  000755                          BR      1$              ;GET MORE PARAMETERS
017500  104401  035713          6$:     TYPE    ,BADENT         ;'BAD ENTRY'
017504  162703  000006                  SUB     #6,R3           ;DECREMENT THE TABLE POINTER
017510  000744                          BR      1$              ;TRY AGAIN
017512  010215                  7$:     MOV     R2,(R5)         ;NEW VALUE
017514  000402                          BR      9$              ;EXIT
                                ;8$:    MOV     #TABLE,R3       ;RELOAD THE PARAMETER TABLE ADDRESS
017516  011403                  8$:     MOV     (R4),R3         ;RELOAD THE PARAMETER TABLE ADDRESS
017520  000740                          BR      1$              ;TRY AGAIN
017522  012603                  9$:     MOV     (SP)+,R3        ;RESTORE R3
017524  062704  000002                  ADD     #2,R4   ;ADJUST THE RETURN ADDRESS
017530  000204                          RTS     R4              ;RETURN FROM THE R4
                                ;       RTS     PC              ;RETURN
                                ;THIS ROUTINE LOAD THE BAD SECTOR INTO USTAB TABLE
                                ;CALL SEQ  JSR  PC,RECORD
                                ;ALL REGISTER USED ARE DESTORIED
                                
017532  012701  003434          RECORD: MOV     #USTAB+10,R1    ;TABLE ENTRY IN R1
017536  022711  177777          1$:     CMP     #-1,(R1)        ;AN OPENING IN THE TABLE
017542  001421                          BEQ     3$              ;BRANCH IF IT IS
017544  023711  005622                  CMP     FMTDPB+12,(R1)  ;CYLINDER NUMBER MATCHES ?
017550  001010                          BNE     2$              ;BRANCH IF NOT
017552  123761  005621  000003          CMPB    FMTDPB+11,3(R1) ;TRK NUMBER MATCHES ?
017560  001004                          BNE     2$              ;BRANCH IF NOT
017562  123761  005620  000002          CMPB    FMTDPB+10,2(R1) ;SECTOR NUMBER MATCHES ?
017570  001416                          BEQ     4$              ;BRANCH TO EXIT IF THE SPOT HAS RECOREDED
017572  062701  000004          2$:     ADD     #4,R1           ;INCREMENT 4 BYTES
017576  022701  004424                  CMP     #USTAB+512.,R1  ;END OF TABLE ?
017602  101355                          BHI     1$              ;BRANCH IF NOT
017604  000411                          BR      5$              ;THROW AWAY THE PACK
017606  013711  005622          3$:     MOV     FMTDPB+12,(R1)  ;LOAD THE CYLINDER #
017612  113761  005621  000003          MOVB    FMTDPB+11,3(R1) ;LOAD THE TRK NUMBER
```

```
3384  017620  113761  001350  000002          MOVB    SAVSEC,2(R1)    ;LOAD THE SECTOR NUMBER
3385  017626  000207                  4$:     RTS     PC              ;EXIT
3386  017630  104401  001203          5$:     TYPE    ,$CRLF
3387  017634  104401  036566                  TYPE    ,MESG3          ;THROW AWAY THE PACK
3388  017640  000000                          HALT
3389  017642  000137  000200                  JMP     @#200           ;RESTART IF DESIRED ?
3390
3391                                  ;THIS ROUTINE SORT THE USTAB IN ASCENDING ORDER
3392                                  ;CALL SEQ       JSR     PC,SORT2
3393
3394  017646                          SORT2:
3395                                  ;       MOV     #USTAB+8.,R1    ;TOP POINTER
3396  017646  011501                          MOV     (R5),R1         ;TOP POINTER
3397  017650  012702  000175                  MOV     #125.,R2        ;TOTAL 126 ELEMENTS
3398  017654  010103                          MOV     R1,R3           ;LOCATE THE BOTTOM POINTER
3399  017656  062703  000764                  ADD     #500.,R3        ;254-4) X 2
3400                                  ;       MOV     #USTAB+508.,R3  ;THE LAST ELEMENT
3401  017662  022711  177777                  CMP     #-1,(R1)        ;THE TABLE IS EMPTY ?
3402  017666  001444                          BEQ     4$              ;QUICK EXIT
3403  017670  021161  000004          1$:     CMP     (R1),4(R1)      ;COMPARE THE CYLINDER NUMBER
3404  017674  101013                          BHI     2$              ;SWITCH THE TWO ELEMENT
3405  017676  103426                          BLO     3$              ;INCREMENT POINTER
3406  017700  126161  000003  000007          CMPB    3(R1),7(R1)     ;COPMARE THE TRK NUMBER
3407  017706  101006                          BHI     2$              ;SWITCH ELEMENT
3408  017710  103421                          BLO     3$              ;INCREMENT POINTER
3409  017712  126161  000002  000006          CMPB    2(R1),6(R1)     ;COMPARE THE SECTOR NUMBER
3410  017720  101001                          BHI     2$              ;SWITCH ELEMENT
3411  017722  000414                          BR      3$              ;INCREMENT POINTER
3412  017724  011146                  2$:     MOV     (R1),-(SP)      ;SAVE THE N TH BLOCK
3413  017726  016146  000002                  MOV     2(R1),-(SP)     ;INTO STACK
3414  017732  016111  000004                  MOV     4(R1),(R1)      ;SWITCH THE N+1 BLOCK TO N BLOCK
3415  017736  016161  000006  000002          MOV     6(R1),2(R1)     ;
3416  017744  012661  000006                  MOV     (SP)+,6(R1)     ;LOAD THE N+1 BLOCK
3417  017750  012661  000004                  MOV     (SP)+,4(R1)     ;
3418  017754  062701  000004          3$:     ADD     #4,R1           ;UPDATE THE TOP POINTER
3419  017760  020103                          CMP     R1,R3           ;REACH THE BOTTOM ?
3420  017762  001342                          BNE     1$              ;BRANCH IF NOT
3421  017764  005302                          DEC     R2              ;DECREMENT ONE SORT COUNT
3422  017766  001404                          BEQ     4$              ;BRANCH IF ALL DONE
3423  017770  162703  000004                  SUB     #4,R3           ;ASJUST THE BOTTOM POINTER
3424                                  ;       MOV     #USTAB+8.,R1    ;RESET TOP POINTER
3425  017774  011501                          MOV     (R5),R1         ;RESET THE TOP POINTER
3426  017776  000734                          BR      1$              ;BRANCH IF NOT ALL DONE
3427  020000                          4$:
3428  020000  062705  000002                  ADD     #2,R5           ;ADJUST RETURN ADDRESS
3429  020004  000205                          RTS     R5              ;EXIT
3430                                  ;       RTS     PC              ;EXIT
3431                                  ;THIS ROUTINE TEXT THE LAST TRACK   CYL 822, TRK 4
3432                                  ;SECTORS 0,2,4,6,8       16 BIT MFG
3433                                  ;SECTORS 1,3,5,7,9       18 BIT MFG
3434                                  ;SECTORS 10,12,14,16,.....30,  16 BIT USER
3435                                  ;SECTORS 11,13,15,17,.....31          18 BIT USER
3436
3437  020006  012700  043746          TEXT:   MOV     #BUFP,R0        ;BUFFER ADDRESS
3438  020012  012701  020100                  MOV     #<258.*32.>,R1  ;TOTAL WORD COUNT
3439  020016  012702  177777                  MOV     #-1,R2          ;SET ALL LOCATIONS TO -1
```

```
3440  020022  010220                  1$:     MOV     R2,(R0)+        ;FULL THE BUFFER
3441  020024  005301                          DEC     R1              ;ALL DONE ?
3442  020026  001375                          BNE     1$              ;LOOP, IF NOT
3443  020030  013746  001212                  MOV     $TESTN,-(SP)    ;SAVE TESTN,BEGCYL,BEGTRK
3444  020034  013746  001322                  MOV     BEGCYL,-(SP)    ;
3445  020040  013746  001326                  MOV     BEGTRK,-(SP)    ;
3446  020044  013746  001364                  MOV     MAXSEC,-(SP)    ;SAVE FLAG MAXSEC AND SEC30
3447  020050  013746  001362                  MOV     SEC30,-(SP)     ;
3448  020054  012737  000037  001364          MOV     #37,MAXSEC      ;SET MAX 31 SECTORS
3449  020062  012737  177777  001362          MOV     #-1,SEC30       ;ALWAYS IN 16 BIT MODE
3450  020070  012737  001466  001322          MOV     #822.,BEGCYL    ;LOAD LAST CYLINDER
3451  020076  012737  000004  001326          MOV     #4,BEGTRK       ;LAST TRACK
3452  020104  004737  016624                  JSR     PC,SETHDR       ;SET UP THE HEAD FIELD IN THE BUFFER
3453  020110  012637  001362                  MOV     (SP)+,SEC30     ;RESTORE SEC30 AND MAXSEC
3454  020114  012637  001364                  MOV     (SP)+,MAXSEC    ;
3455  020120  012637  001326                  MOV     (SP)+,BEGTRK    ;
3456  020124  012637  001322                  MOV     (SP)+,BEGCYL    ;
3457  020130  012637  001212                  MOV     (SP)+,$TESTN    ;
3458  020134  012701  001414                  MOV     #BAD16,R1       ;LOAD SERIAL NUMBER TO EVERY SECTOR
3459  020140  012702  043752                  MOV     #BUFP+4,R2      ;BUFFER LOCATION + 4
3460  020144  012703  000040                  MOV     #32.,R3         ;TOTAL 32 SECTORS
3461  020150  011112                  2$:     MOV     (R1),(R2)       ;1 ST SN #
3462  020152  016162  000002  000002          MOV     2(R1),2(R2)     ;2 ND SN #
3463  020160  005062  000004                  CLR     4(R2)           ;THIRD WORD = 0
3464  020164  005062  000006                  CLR     6(R2)           ;ID = DATA PACK
3465  020170  062702  001004                  ADD     #516.,R2        ;ADVANCE TO NEXT SECTOR
3466  020174  005303                          DEC     R3              ;DECREMENT ONE SECTOR COUNT
3467  020176  001364                          BNE     2$              ;BRANCH IF NOT DONE
3468  020200  005046                          CLR     -(SP)           ;LOAD TABLE INTO SECTORS
3469  020202  005046                          CLR     -(SP)                           ;DUMMY PAIRS
3470  020204  005046                          CLR     -(SP)
3471  020206  012746  001414                  MOV     #BAD16,-(SP)    ;TABLE ADDRESS
3472  020212  012746  043752                  MOV     #BUFP+4,-(SP)   ;BUFFER ADDRESS
3473  020216  012746  000005                  MOV     #5,-(SP)        ;SECTOR COUNT
3474  020222  012746  002420                  MOV     #BAD18,-(SP)    ;TABLE ADDRESS
3475  020226  012746  044756                  MOV     #BUFP+4+516.,-(SP) ;BUFFER ADDRESS
3476  020232  012746  000005                  MOV     #5,-(SP)        ;SECTOR COUNT
3477  020236  012746  003424                  MOV     #USTAB,-(SP)    ;USER DEFINED TABLE
3478  020242  012746  056022                  MOV     #BUFP+4+<516.*10.>,-(SP)        ;BUFFER ADDRESS
3479  020246  005737  001362                  TST     SEC30
3480  020252  100402                          BMI     3$              ;BRANCH IF IN 16 BIT MODE
3481  020254  012716  057026                  MOV     #BUFP+4+<516.*11.>,(SP) ;RESET THE BUFFER ADDRESS
3482  020260  012746  000013          3$:     MOV     #11.,-(SP)      ;SECTOR COUNT
3483  020264  012701  004430                  MOV     #USSAV,R1       ;TABLE ADDRESS IN R1
3484  020270  012702  057026                  MOV     #BUFP+4+<516.*11.>,R2   ;BUFFER ADDRESS IN R2
3485  020274  005737  001362                  TST     SEC30           ;BRANCH IF IN 16 BIT MODE
3486  020300  100402                          BMI     4$              ;
3487  020302  012702  056022                  MOV     #BUFP+4+<516.*10.>,R2   ;BUFFER ADDRESS FOR 18 BIT MODE
3488  020306  012703  000013          4$:     MOV     #11.,R3         ;SECTOR COUNT IN R3
3489  020312  010105                          MOV     R1,R5           ;TEMPOR STORAGE
3490  020314  012704  000400          5$:     MOV     #256.,R4        ;WORD COUNT = DATA FIELD OF ONE SECTOR
3491  020320  012122                  6$:     MOV     (R1)+,(R2)+     ;LOAD TABLE INTO SECTOR DATA FIELD
3492  020322  005304                          DEC     R4              ;ALL DONE ?
3493  020324  001375                          BNE     6$              ;BRANCH IF NOT
3494  020326  010501                          MOV     R5,R1           ;RELOAD THE TABLE ADDRESS
3495  020330  062702  001010                  ADD     #520.,R2        ;SKIP ONE SECTOR
```

```
3496  020334  005303                          DEC     R3              ;DECREMENT ONE SECTOR COUNT
3497  020336  001366                          BNE     5$              ;BRANCH IF NOT DONE
3498  020340  012603                          MOV     (SP)+,R3        ;RETRIEVE NEXT SET OF SECTOR *
3499  020342  012602                          MOV     (SP)+,R2        ;BUFFER ADDRESS
3500  020344  012601                          MOV     (SP)+,R1        ;TABLE ADDRESS
3501  020346  010105                          MOV     R1,R5           ;TEMPOR STORAGE
3502  020350  001361                          BNE     5$              ;BRANCH IF NOT DUMMY PAIR
3503  020352  000207                          RTS     PC              ;EXIT
3504
3505                                  ;THIS ROUTINE RESET:
3506                                  ;RESET THE BIT 14/15 OF THE 1 ST HEADER WORD OF THE BAD SPOT
3507                                  ;CALLING SEQ;           JSR     PC,RESET
3508                                  ;
3509
3510  020354  013737  005622  043736  RESET:  MOV     FMTDPB+12,RBUF  ;CYLINDER ADDRESS
3511  020362  001006                          BNE     3$              ;BRANCH IF NOT ON TRACK 0
3512  020364  122737  000001  005621          CMPB    #1,FMTDPB+11    ;BRANCH IF NOT CYL0,TRK 0 OR 1
3513  020372  103402                          BLO     3$              ;BRANCH IF NOT
3514  020374  000137  016104                  JMP     REJCT2          ;OTHERWISE, ILLEGAL BAD SECTORS
3515  020400  000240                  3$:     NOP                     ;
3516  020402  113737  005621  043741          MOVB    FMTDPB+11,RBUF+3        ;TRACK ADDRESS
3517  020410  113737  001350  043740          MOVB    SAVSEC,RBUF+2   ;SECTOR ADDRESS
3518  020416  053737  001412  043736          BIS     HEADX,RBUF      ;SET MFG BIT,RESET USER BIT
3519  020424  113737  001220  005630          MOVB    DRIVE,FMTX      ;SETUP THE DPB FOR OPERATION
3520  020432  012737  177776  005634          MOV     #-2,FMTX+4      ;WORD COUNT =2
3521  020440  112737  000163  005632          MOVB    #WRTHD,FMTX+2   ;WRITE HEAD AND DATA COMMAND
3522  020446  012737  043736  005636          MOV     #RBUF,FMTX+6    ;BUFFER ADDRESS
3523  020454  113737  001350  005640          MOVB    SAVSEC,FMTX+10  ;SECTOR ADDRESS
3524  020462  113737  005621  005641          MOVB    FMTDPB+11,FMTX+11 ;TRACK ADDRESS
3525  020470  013737  005622  005642          MOV     FMTDPB+12,FMTX+12 ;CYLINDER ADDRESS
3526  020476  052737  010000  043736          BIS     #10000,RBUF     ;ASSUM 16 BIT MODE
3527  020504  005737  001362                  TST     SEC30           ;IN 16 BIT MODE ?
3528  020510  100403                          BMI     1$              ;BRANCH IF IT IS
3529  020512  042737  010000  043736          BIC     #10000,RBUF     ;RESET THE FMT BIT
3530  020520  004037  027410          1$:     JSR     R0,RM03         ;CALL THE DRIVER
3531  020524  005630                          FMTX                    ;DPB ADDRESS
3532  020526  000774                          BR      1$              ;BRANCH IF QUEUE FAILS
3533  020530  005737  005646          2$:     TST     FMTX+16         ; DONE ?
3534  020534  001775                          BEQ     2$              ;BRANCH IF NOT
3535  020536  000207                          RTS     PC              ;EXIT
3536                                  ;THE ROUTINE INSERT:
3537                                  ;INSERT A BAD SECTOR ADDRESS INTO ONE OF THE USTAB,USSAV,BAD16,BAD18 TABLES
3538                                  ;CALLING SEQ
3539                                  ;               JSR     R5,INSERT
3540                                  ;               TABLE + 8.
3541                                  ;
3542  020540  010446                  INSERT: MOV     R4,-(SP)        ;SAVE THE R4
3543  020542  012746  000176                  MOV     #126.,-(SP)     ;ENTRIES OF THE TABLE
3544  020546  011504                          MOV     (R5),R4         ;TABLE ADDRESS + 8 BYTES
3545  020550  022714  177777          1$:     CMP     #-1,(R4)        ;AN OPEN ENTRY ?
3546  020554  001420                          BEQ     3$              ;BRANCH IF SO
3547  020556  023714  001366                  CMP     DS.CYL,(R4)     ;CHECK IF THE BAD SECTOR HAS RECORDED
3548  020562  001010                          BNE     2$              ;BRANCH IF NOT
3549  020564  123764  001370  000003          CMPB    DS.TRK,3(R4)    ;CYL AND TRK MATCH ?
3550  020572  001004                          BNE     2$              ;BRANCH IF NOT
3551  020574  123764  001350  000002          CMPB    SAVSEC,2(R4)    ;CYL ,TRK AND SEC MATCH ?
```

```
3552  020602  001415                          BEQ     4$              ;BRANCH IF SO
3553  020604  005316                  2$:     DEC     (SP)            ;DECREMENT THE ENTRY COUNT
3554  020606  001413                          BEQ     4$              ;BRANCH IF EXHAUST
3555  020610  062704  000004                  ADD     #4,R4           ;LOCATE THE NEXT ENTRY
3556  020614  000755                          BR      1$              ;BRANCH BACK
3557  020616  013714  001366          3$:     MOV     DS.CYL,(R4)     ;LOAD INTO TABLE
3558  020622  113764  001370  000003          MOVB    DS.TRK,3(R4)    ;LOAD THE TRACK NUMBER
3559  020630  113764  001350  000002          MOVB    SAVSEC,2(R4)    ;LOAD THE SECTOR NUMBER
3560  020636  062706  000002          4$:     ADD     #2,SP           ;CLEAR UP THE STACK
3561  020642  012604                          MOV     (SP)+,R4        ;RESTORE R4
3562  020644  062705  000002                  ADD     #2,R5           ;ASJUST THE RETURN ADDRESS
3563  020650  000205                          RTS     R5              ;EXIT
3564  020652  104401  037631  FUNCT1:  TYPE    ,MESG5
3565  020656  012737  177777  001366  1$:     MOV     #-1,DS.CYL      ;RESET THE INPUT VALUES
3566  020664  012737  177777  001370          MOV     #-1,DS.TRK      ;
3567  020672  012737  177777  001350          MOV     #-1,SAVSEC      ;
3568  020700  004437  017416                  JSR     R4,PARENT       ;ENTER THE VALUE FROM THE KEYBOARD
3569  020704  005702                          TABLE2
3570  020706  005737  001366                  TST     DS.CYL          ;UPDATE VALUES ?
3571  020712  100412                          BMI     2$              ;BRANCH IF NOT
3572  020714  005737  001370                  TST     DS.TRK          ;
3573  020720  100407                          BMI     2$              ;
3574  020722  005737  001350                  TST     SAVSEC          ;
3575  020726  100404                          BMI     2$              ;
3576  020730  004537  020540                  JSR     R5,INSERT       ;INSERT INTO THE BAD16 TABLE
3577  020734  001424                          BAD16+8.                ;
3578  020736  000747                          '       1$              ;NEXT SET
3579  020740  004537  017646          2$:     J_      R5,SORT2        ;SORT THE TABLE
3580  020744  001424                          BAD16+8.
3581  020746  104401  037702                  TYPE    ,MESG6
3582  020752  012737  177777  001366  3$:     MOV     #-1,DS.CYL      ;RESET THE INPUT VALUES
3583  020760  012737  177777  001370          MOV     #-1,DS.TRK      ;
3584  020766  012737  177777  001350          MOV     #-1,SAVSEC      ;
3585  020774  004437  017416                  JSR     R4,PARENT       ;INPUT VALUES FROM THE KEYBOARD
3586  021000  005726                          TABLE3                  ;1d BIT TABLE
3587  021002  005737  001366                  TST     DS.CYL          ;UPDATED VALUES ?
3588  021006  100412                          BMI     4$              ;BRANCH IF NOT
3589  021010  005737  001370                  TST     DS.TRK          ;
3590  021014  100407                          BMI     4$              ;
3591  021016  005737  001350                  TST     SAVSEC          ;
3592  021022  100404                          BMI     4$              ;
3593  021024  004537  020540                  JSR     R5,INSERT       ;INSERT THE BAD SECTOR INTO TABLE BAD18
3594  021030  002430                          BAD18+8.
3595  021032  000747                          BR      3$              ;NEXT SET
3596  021034  004537  017646          4$:     JSR     R5,SORT2        ;SORT THE BAD18 TABLE
3597  021040  002430                          BAD18+8.
3598  021042  000207                          RTS     PC              ;EXIT FROM THE CALLING OF TABCAL
3599  021044  012737  003434  021214  FUNCT2: MOV     #USTAB+8.,3$    ;SETUP TABLES FOR 16 BITS MODE
3600  021052  012737  003434  021224          MOV     #USTAB+8.,5$    ;
3601  021060  012737  004440  021310          MOV     #USSAV+8.,7$    ;
3602  021066  012737  004440  021320          MOV     #USSAV+8.,9$    ;
3603  021074  005737  001362                  TST     SEC30           ;IN 16 BIT MODE ?
3604  021100  100414                          BMI     1$              ;BRANCH IS SO,OTHERWISE RELOAD THE TABLES
3605  021102  012737  003434  021310          MOV     #USTAB+8.,7$    ;SET UP TABLES FOR 18 BITS MODE
3606  021110  012737  003434  021320          MOV     #USTAB+8.,9$    ;
3607  021116  012737  004440  021214          MOV     #USSAV+8.,3$    ;
```

```
3608  021124  012737  004440  021224          MOV     #USSAV+8.,5$    ;
3609  021132  104401  037753          1$:     TYPE    ,MESG7          ;MESSAGE FOR ENTER 16 BITS USR TABLE
3610  021136  012737  177777  001366  2$:     MOV     #-1,DS.CYL      ;RESET THE INPUT VALUES
3611  021144  012737  177777  001370          MOV     #-1,DS.TRK      ;
3612  021152  012737  177777  001350          MOV     #-1,SAVSEC      ;
3613  021160  004437  017416                  JSR     R4,PARENT       ;ENTER BAD SECTOR ADDRESS FROM THE KEYBOARD
3614  021164  005702                          TABLE2                  ;
3615  021166  005737  001366                  TST     DS.CYL          ;UPDATED INPUT VALUE ?
3616  021172  100412                          BMI     4$              ;BRANCH IF NOT
3617  021174  005737  001370                  TST     DS.TRK          ;
3618  021200  100407                          BMI     4$              ;
3619  021202  005737  001350                  TST     SAVSEC          ;
3620  021206  100404                          BMI     4$              ;
3621  021210  004537  020540                  JSR     R5,INSERT       ;INSERT THE BAD SECTOR INTO TABLE
3622  021214  000000          3$:             .WORD   0               ;TABLE ADDRESS
3623  021216  000747                          BR      2$              ;ENTER NEXT BAD SECTOR ADDRESS
3624  021220  004537  017646          4$:     JSR     R5,SORT2        ;SORT THE TABLE
3625  021224  000000          5$:             .WORD   0               ;TABLE ADDRESS
3626  021226  104401  040024                  TYPE    ,MESG8          ;MESSAGE FOR 18 BIT USR TABLE
3627  021232  012737  177777  001366  6$:     MOV     #-1,DS.CYL      ;RESET THE INPUT VALUES
3628  021240  012737  177777  001370          MOV     #-1,DS.TRK      ;
3629  021246  012737  177777  001350          MOV     #-1,SAVSEC      ;
3630  021254  004437  017416                  JSR     R4,PARENT       ;ENTER BAD SECTOR FORM KEYBOARD
3631  021260  005726                          TABLE3                  ;
3632  021262  005737  001366                  TST     DS.CYL          ;UPDATED INPUT VALUE ?
3633  021266  100412                          BMI     8$              ;BRANCH IF NOT
3634  021270  005737  001370                  TST     DS.TRK          ;BRANCH IF NOT UPDATE
3635  021274  100407                          BMI     8$              ;
3636  021276  005737  001350                  TST     SAVSEC          ;
3637  021302  100404                          BMI     8$              ;BRANCH IF NOT UPDATED
3638  021304  004537  020540                  JSR     R5,INSERT       ;INSERT THE BAD SECTOR INTO TABLE
3639  021310  000000          7$:             .WORD   0               ;
3640  021312  000747                          BR      6$              ;BRANCH TO ENTER NEXT SECTOR
3641  021314  004537  017646          8$:     JSR     R5,SORT2        ;SORT THE TABLE
3642  021320  000000          9$:             .WORD   0               ;TABLE ADDRESS
3643  021322  000207                          RTS     PC              ;EXIT FORM THE TABCAL CALLING
3644  021324  010146          FUNCT3:         MOV     R1,-(SP)        ;SAVE R1
3645  021326  012701  001414                  MOV     #BAD16,R1       ;ADDRESS OF THE TABLE
3646  021332  104401  040075                  TYPE    ,MESG9          ;MESG TO TYPE THE 16 BIT MFG
3647  021336  104401  001203          1$:     TYPE    ,$CRLF          ;CR-LF FIRST
3648  021342  012146                          MOV     (R1)+,-(SP)     ;CALLING SEQ FOR TYPOC RT.
3649  021344  100402                          BMI     2$              ;BRANCH IF END OF TABLE
3650  021346  104402                          TYPOC                   ;TYPE THE VALUE
3651  021350  000772                          BR      1$              ;LOOPING BACK
3652  021352  062706  000002          2$:     ADD     #2,SP           ;ADJUST THE RETURN ADDRESS
3653  021356  104401  040135                  TYPE    ,MESG10         ;MESG TO TYPE 18 BIT MFG
3654  021362  012701  002420                  MOV     #BAD18,R1       ;TABLE ADDRESS
3655  021366  104401  001203          3$:     TYPE    ,$CRLF          ;
3656  021372  012146                          MOV     (R1)+,-(SP)     ;CALLING SEQ FOR TYPOC RT.
3657  021374  100402                          BMI     4$              ;BRANCH IF END OF TABLE
3658  021376  104402                          TYPOC                   ;TYPE THE VALUE IN OCTAL
3659  021400  000772                          BR      3$              ;LOOPING BACK
3660  021402  062706  000002          4$:     ADD     #2,SP           ;ADJUST THE RETURN ADDRESS
3661  021406  012601          5$:             MOV     (SP)+,R1        ;RESTORE THE R1
3662  021410  000207                          RTS     PC              ;EXIT FROM THE CALLING OF TABCAL
3663  021412  010146          FUNCT4:         MOV     R1,-(SP)        ;SAVE R1
```

```
3664  021414  010246                  MOV     R2,-(SP)                ;SAVE R2
3665  021416  012701  003424          MOV     #USTAB,R1               ;SET 16 BIT MODE TABLE
3666  021422  012702  004430          MOV     #USSAV,R2               ;
3667  021426  005737  001362          TST     SEC30                   ;16 BIT MODE ?
3668  021432  100404                  BMI     1$                      ;BRANCH IF SO
3669  021434  012702  003424          MOV     #USTAB,R2               ;OTHERWISE,RELOAD THE TABLE ADDRESS
3670  021440  012701  004430          MOV     #USSAV,R1               ;
3671  021444  104401  040175  1$:     TYPE    ,MESG11                 ;MESG FOR 16 BIT
3672  021450  104401  001203  2$:     TYPE    ,$CRLF
3673  021454  012146                  MOV     (R1)+,-(SP)             ;
3674  021456  100402                  BMI     3$                      ;BRANCH IF END OF TABLE
3675  021460  104402                  TYPOC                           ;
3676  021462  000772                  BR      2$                      ;
3677  021464  062706  000002  3$:     ADD     #2,SP                   ;ADJUST STACK POINTER
3678  021470  104401  040235          TYPE    ,MESG12                 ;MESSAGE FOR 18 BIT MODE
3679  021474  104401  001203  4$:     TYPE    ,$CRLF                  ;CR-LF
3680  021500  012246                  MOV     (R2)+,-(SP)             ;
3681  021502  100402                  BMI     5$                      ;BRANCH IF END OF TABLE
3682  021504  104402                  TYPOC
3683  021506  000772                  BR      4$                      ;NEXT SECTOR ADDRESS
3684  021510  062706  000002  5$:     ADD     #2,SP                   ;ADJUST THE STACK POINT
3685  021514  012602                  MOV     (SP)+,R2                ;RESTORE R2,R1
3686  021516  012601                  MOV     (SP)+,R1                ;
3687  021520  000207                  RTS     PC                      ;EXIT FROM THE CALLING OF TABCAL
3688  021522  010146  FUNCT5:  MOV     R1,-(SP)                ;SAVE R1,R2
3689  021524  010246                  MOV     R2,-(SP)                ;
3690  021526  005037  001420          CLR     BAD16+4.
3691  021532  005037  001422          CLR     BAD16+6.
3692  021536  012701  001424          MOV     #BAD16+8.,R1            ;START AT THE 5TH WORD
3693  021542  012702  000374          MOV     #252.,R2                ;WORD CTR
3694  021546  012721  177777  1$:     MOV     #-1,(R1)+               ;RESET TO -1
3695  021552  005302                  DEC     R2                      ;DECREMENT WORD CTR
3696  021554  001374                  BNE     1$                      ;BRANCH IF NOT DONE
3697  021556  005037  002424          CLR     BAD18+4.
3698  021562  005037  002426          CLR     BAD18+6.
3699  021566  012701  002430          MOV     #BAD18+8.,R1            ;START AT THE 5TH WORD
3700  021572  012702  000374          MOV     #252.,R2                ;WORD CTR
3701  021576  012721  177777  2$:     MOV     #-1,(R1)+               ;RESET TO -1
3702  021602  005302                  DEC     R2                      ;DECREMENT WORD CTR
3703  021604  001374                  BNE     2$                      ;BRANCH IF NOT DONE
3704  021606  012602                  MOV     (SP)+,R2                ;RESTORE R2,R1
3705  021610  012601                  MOV     (SP)+,R1                ;
3706  021612  104401  040324          TYPE    ,MESG14                 ;DONE MESAGE
3707  021616  000207                  RTS     PC                      ;EXIT FROM THE TABCAL CALLING
3708  021620  010146  FUNCT6:  MOV     R1,-(SP)                ;STORE R1,R2
3709  021622  010246                  MOV     R2,-(SP)                ;
3710  021624  005037  003430          CLR     USTAB+4
3711  021630  005037  003432          CLR     USTAB+6
3712  021634  012701  003434          MOV     #USTAB+8.,R1
3713  021640  012702  000374          MOV     #252.,R2                ;WORD CTR
3714  021644  012721  177777  1$:     MOV     #-1,(R1)+               ;RESET TO -1
3715  021650  005302                  DEC     R2                      ;DECREMENT WORD CTR
3716  021652  001374                  BNE     1$                      ;BRANCH IF NOT DONE
3717  021654  005037  004434          CLR     USSAV+4
3718  021660  005037  004436          CLR     USSAV+6
3719  021664  012701  004440          MOV     #USSAV+8.,R1            ;TABLE ADDRESS
```

```
3720  021670  012702  000374            MOV     #252.,R2                ;
3721  021674  012721  177777      2$:   MOV     #-1,(R1)+               ;RESET TO -1
3722  021700  005302                    DEC     R2                      ;DECREMENT WORD CTR
3723  021702  001374                    BNE     2$                      ;BRANCH IF NOT DONE
3724  021704  012602                    MOV     (SP)+,R2                ;RESTORE R1,R2
3725  021706  012601                    MOV     (SP)+,R1                ;
3726  021710  104401  040324            TYPE    ,MESG14                 ;DONE MESSAGE
3727  021714  000207                    RTS     PC                      ;EXIT FORM THE TABCAL CALLING
3728  021716  104401  040275  FUNCT7:  TYPE    ,MESG13         ;MESSAGE ENTER SECTOR #
3729  021722  104411            1$:   RDLIN                           ;READ THE SECTOR NUMBER
3730  021724  012601                    MOV     (SP)+,R1                ;ADDRESS OF READ-IN BUFFER
3731  021726  012702  000040            MOV     #32.,R2                 ;MAX VALUE
3732  021732  004537  022360            JSR     R5,CK.DIG               ;CHECK THE READ-IN VALUE
3733  021736  021752                    2$
3734  021740  021752                    2$
3735  021742  021722                    1$
3736  021744  021752                    2$
3737  021746  021752                    2$
3738  021750  021752                    2$
3739  021752  006302            2$:   ASL     R2                      ;FOUND THE WORD INDEX
3740  021754  016201  006132            MOV     ADRTBL(R2),R1           ;ADDRESS OF THE BUFFER
3741  021760  012702  000402            MOV     #258.,R2                ;WORD CTR FOR WHOLE SECTORINCLUDING HEAD
3742  021764  104401  001203      3$:   TYPE    ,$CRLF                  ;CR-LF
3743  021770  012705  000010            MOV     #10,R5                  ;8 WORDS ON A LINE
3744  021774  012146            4$:   MOV     (R1)+,-(SP)             ;CONTENTS OF THE BUFFER
3745  021776  104402                    TYPOC                           ;
3746  022000  104401  035166            TYPE    ,_INSP                  ;TYPE THE SPACES
3747  022004  005302                    DEC     R2                      ;DECREMENT WORD CTR
3748  022006  001403                    BEQ     5$                      ;BRANCH IF  FINISH
3749  022010  005305                    DEC     R5                      ;START ANOTHER LINE ?
3750  022012  001370                    BNE     4$                      ;BRANCH IF NOT
3751  022014  000763                    BR      3$                      ;BRANCH IF SO
3752  022016  000207            5$:   RTS     PC                      ;EXIT FROM TABCAL CALLING
3753  022020  104401  040333  FUNCT8:  TYPE    ,MESG15
3754  022024  012737  177777  001366  1$:   MOV     #-1,DS.CYL      ;RESET ALL INPUT VALUES
3755  022032  012737  177777  001370        MOV     #-1,DS.TRK      ;TRACK NUMBER
3756  022040  012737  047301  001350        MOV     #20161.,SAVSEC  ;BYTE NUMBER
3757  022046  004437  017416                JSR     R4,PARENT       ;ENTER THE VALUE FROM KEYBOARD
3758  022052  005752                    TABLE4
3759  022054  005737  001366            TST     DS.CYL                  ;UPDATED ?
3760  022060  100455                    BMI     6$                      ;EXIT IF NOT
3761  022062  005737  001370            TST     DS.TRK                  ;BRANCH IF NOT UPDATED
3762  022066  100452                    BMI     6$                      ;
3763  022070  005737  001350            TST     SAVSEC                  ;BRANCH TO EXIT IF NEG
3764  022074  100447                    BMI     6$                      ;EXIT
3765  022076  022737  047301  001350    CMP     #20161.,SAVSEC          ;TOO LARGE ?
3766  022104  003443                    BLE     6$                      ;BRANCH IF SO,EXIT
3767  022106  010446                    MOV     R4,-(SP)                ;SAVE R4
3768  022110  013746  001350            MOV     SAVSEC,-(SP)            ;SAVE SAVSEC
3769  022114  005004                    CLR     R4                      ;INDEX STARTS FROM 0
3770  022116  023764  001350  006332  2$:   CMP     SAVSEC,BYTE16(R4)  ;LOCATE THE SECTOR ADDRESS
3771  022124  101403                    BLOS    3$                      ;BRANCH IF FOUND
3772  022126  062704  000002            ADD     #2,R4                   ;TRY NEXT SECTOR
3773  022132  000771                    BR      2$                      ;LOOPING
3774  022134  006204            3$:   ASR     R4                      ;SECTOR ADDRESS
3775  022136  010437  001350            MOV     R4,SAVSEC               ;16 BIT MODE BAD SECTOR ADD
```

```
3776  022142  004537  020540            JSR     R5,INSERT       ;INSERT INTO TABLE
3777  022146  001424                    BAD16+8.
3778  022150  012637  001350            MOV     (SP)+,SAVSEC    ;LOCATE SECTOR FOR 18 BIT
3779  022154  005004                    CLR     R4              ;RESET INDEX VALUE
3780  022156  023764  001350  006432  4$:   CMP     SAVSEC,BYTE18(R4) ;USE TABLE 18 BIT
3781  022164  101403                    BLOS    5$              ;BRANCH IF FOUND
3782  022166  062704  000002            ADD     #2,R4           ;TRY NEXT SECTOR IF NOT FOUND
3783  022172  000771                    BR      4$              ;LOOPIN BACK
3784  022174  006204              5$:   ASR     R4              ;SECTOR ADDRESS
3785  022176  010437  001350            MOV     R4,SAVSEC       ;
3786  022202  004537  020540            JSR     R5,INSERT       ;INSET INTO TABLE
3787  022206  002430                    BAD18+8.
3788  022210  012604                    MOV     (SP)+,R4        ;RESTORE R4
3789  022212  000704                    BR      1$              ;LOOPING BACK
3790  022214  004537  017646        6$: JSR     R5,SORT2        ;SORT BOTH TABLE ANYWAY
3791  022220  001424                    BAD16+8.
3792  022222  004537  017646            JSR     R5,SORT2        ;
3793  022226  002430                    BAD18+8.
3794  022230  000207                    RTS     PC              ;RETURN TO SERVICE LOOP
3795
3796
3797                                ;**************************
3798
3799                                ;THIS ROUTINE IS USED TO CHECK IF AN
3800                                ;ASCII CHARACTER IS A DIGIT BETWEEN 0 AND 7.
3801                                ;CALL
3802                                ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
3803                                ;       JSR     R5,CK.OCT       ;CHECK THE CHARACTER
3804                                ;       RETURN1                 ;CHARACTER IS NOT BETWEEN 0-7
3805                                ;       RETURN2                 ;CHARACTER IS IN R2 AS A
3806                                ;                               ;OCTAL DIGIT
3807
3808  022232  121127  000060      CK.OCT: CMPB    (R1),#'0        ;LESS THAN ZERO?
3809  022236  103407                    BLO     1$              ;YES -- BRANCH
3810  022240  121127  000067            CMPB    (R1),#'7        ;GREATER THAN SEVEN?
3811  022244  101004                    BHI     1$              ;YES -- BRANCH
3812  022246  111102                    MOVB    (R1),R2         ;GET THE CHARACTER
3813  022250  042702  177770            BIC     #^C7,R2         ;STRIP AWAY THE ASCII
3814  022254  005725                    TST     (R5)+           ;ADJUST FOR RETURN
3815  022256  000205              1$:   RTS     R5              ;RETURN
3816
3817                                ;THIS ROUTINE IS USED TO CHECK AN ASCII CHARACTER
3818                                ;AND DETERMINE IF IT IS A DIGIT BETWEEN 0 AND 9.
3819                                ;CALL
3820                                ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
3821                                ;       JSR     R5,CK.DEC       ;CHECK THE CHARACTER
3822                                ;       RETURN1                 ;NOT BETWEEN 0 AND 9
3823                                ;       RETURN2                 ;BETWEEN 0 AND 9
3824                                ;                               ;R2 = DIGIT
3825
3826  022260  121127  000060      CK.DEC: CMPB    (R1),#'0        ;LESS THAN ZERO?
3827  022264  103407                    BLO     1$              ;YES -- BRANCH
3828  022266  121127  000071            CMPB    (R1),#'9        ;GREATER THAN NINE?
3829  022272  101004                    BHI     1$              ;YES -- BRANCH
3830  022274  111102                    MOVB    (R1),R2         ;GET THE CHARACTER
3831  022276  042702  000060            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

022306  105711                 CK.CHR: TSTB    (R1)            ;"CARRIAGE RETURN"?
022310  001417                         BEQ     3$              ;YES -- BRANCH
022312  121127  000054                 CMPB    (R1),#',        ;"COMMA"?
022316  001413                         BEQ     2$              ;YES -- BRANCH
022320  121127  000056                 CMPB    (R1),#'.        ;"PERIOD"?
022324  001407                         BEQ     1$              ;YES -- BRANCH
022326  004537  022260                 JSR     R5,CK.DEC       ;"DIGIT"?
022332  000410                         BR      4$              ;NO -- BRANCH
022334  004537  022232                 JSR     R5,CK.OCT       ;OCTAL ?
022340  005725                         TST     (R5)+           ;DIGIT BETWEEN 8-9
022342  005725                         TST     (R5)+           ;DIGIT BETWEEN 0-7
022344  005725                 1$:     TST     (R5)+           ;PERIOD
022346  005725                 2$:     TST     (R5)+           ;COMMA
022350  005725                 3$:     TST     (R5)+           ;CARRIAGE RETURN
022352  005201                         INC     R1              ;MOVE POINTER TO NEXT CHARACTER
022354  011505                 4$:     MOV     (R5),R5         ;UNKNOWN CHARACTER
022356  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

022360  010446                 CK.DIG: MOV     R4,-(SP)        ;SAVE R4
022362  010346                         MOV     R3,-(SP)        ;SAVE R3
022364  010246                         MOV     R2,-(SP)        ;SAVE THE MAX. SIZE ON THE STACK
022366  005002                         CLR     R2              ;START WITH 0
022370  005003                         CLR     R3
022372  005004                         CLR     R4
022374  004537  022306                 JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
022400  022474                         6$                      ;ILLEGAL CHARACTER
022402  022502                         9$                      ;CARRIAGE RETURN
```

```
3888  022404  022474                        6$                      ;",",
3889  022406  022476                        7$                      ;"-","
3890  022410  022414                        1$                      ;DIGIT 0-7
3891  022412  022414                        1$                      ;DIGIT 8-9
3892  022414  062705  000004          1$:   ADD     #4,R5             ;STEP RETURN POINTER PAST "CR" & "PERIOD" RETURNS
3893  022420  006303                  2$:   ASL     R3                ;INPUT NUMBER *2
3894  022422  010346                        MOV     R3,-(SP)          ;SAVE *2
3895  022424  006303                        ASL     R3                ;*4
3896  022426  006303                        ASL     R3                ;*8
3897  022430  062603                        ADD     (SP)+,R3          ;(*2)+(*8) = *10
3898  022432  060203                        ADD     R2,R3             ;UPDATE THE INPUT NUMBER
3899  022434  004537  022306                JSR     R5,CK.CHR         ;CHECK ONE CHARACTER
3900  022440  022500                        8$                        ;ILLEGAL CHARACTER
3901  022442  022464                        5$                        ;CARRIAGE RETURN
3902  022444  022462                        4$                        ;"."
3903  022446  022454                        3$                        ;"-"."
3904  022450  022420                        2$                        ;DIGIT 0-7
3905  022452  022420                        2$                        ;DIGIT 8-9
3906  022454  105711                  3$:   TSTB    (R1)              ;DOES A "CR" FOLLOW THE "PERIOD"
3907  022456  001010                        BNE     8$                ;BR IF NOT
3908  022460  005724                        TST     (R4)+             ;INCREMENT THE RETURN
3909  022462  005724                  4$:   TST     (R4)+             ;INCREMENT THE RETURN
3910  022464  005724                  5$:   TST     (R4)+             ;INCREMENT THE RETURN
3911  022466  020316                        CMP     R3,(SP)           ;CHECK THE MAGNITUDE OF THE NUMBER
3912  022470  002004                        BGE     9$                ;BR IF ENTERED NUMBER TOO LARGE
3913  022472  000402                        BR      8$                ;BYPASS INCREMENT
3914  022474  005725                  6$:   TST     (R5)+             ;INCREMENT RETURN PAST INVALID RETURN
3915  022476  005725                  7$:   TST     (R5)+             ;INCREMENT RETURN
3916  022500  060405                  8$:   ADD     R4,R5             ;SETUP RETURN POINTER
3917  022502  010302                  9$:   MOV     R3,R2             ;ENTERED VALUE
3918  022504  005726                        TST     (SP)+             ;CLEAN MAX. SIZE OFF OF STACK
3919  022506  012603                        MOV     (SP)+,R3          ;RESTORE R3
3920  022510  012604                        MOV     (SP)+,R4          ;RESTORE R4
3921  022512  011505                        MOV     (R5),R5           ;GET RETURN ADDRESS
3922  022514  000205                        RTS     R5                ;RETURN
3923
3924                                  .SBTTL  MACRO ROUTINES
3925
3926
3927                                  .SBTTL  ERROR HANDLER ROUTINE
3928
3929                                  ;;*****************************************************************
3930                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
3931                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
3932                                  ;*AND GO TO TYPERR ON ERROR
3933                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
3934                                  ;*SW15=1         HALT ON ERROR
3935                                  ;*SW13=1         INHIBIT ERROR TYPEOUTS
3936                                  ;*SW10=1         BELL ON ERROR
3937                                  ;*SW09=1         LOOP ON ERROR
3938                                  ;*CALL
3939                                  ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
3940
3941  022516                          $ERROR:
3942  022516  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
3943  022520  010137  001372                  MOV     R1,DDRIVE       ;DRIVE ADDRESS IF DRIVE ERROR CALL
```

```
3944  022524  010337  001374                    MOV     R3,ATTN          ;ATTENTION REGISTER CONTENTS
3945  022530  013737  005622  001366            MOV     FMTDPB+12,DS.CYL  ;CURRENT CYLINDER ADDRESS
3946  022536  113737  005621  001370            MOVB    FMTDPB+11,DS.TRK  ;REQUESTED TRACK ADDRESS
3947  022544  005737  005544                    TST     RM.REG+RMBA      ;NON-ZERO BUFFER ADDRESS ?
3948  022550  001406                            BEQ     7$               ;BR IF NO BUFFER ADDRESS
3949  022552  013746  005544                    MOV     RM.REG+RMBA,-(SP)  ;BUFFER ADDRESS
3950  022556  162716  000002                    SUB     #2,(SP)          ;DECREMENT THE ADDRESS
3951  022562  013637  001140                    MOV     @(SP)+,$GDDAT    ;GET THE BUFFER WORD WHICH DIDN'T COMPARE
3952  022566  105237  001117            7$:     INCB    $ERFLG           ;;SET THE ERROR FLAG
3953  022572  001775                            BEQ     7$               ;;DON'T LET THE FLAG GO TO ZERO
3954  022574  013777  001116  156354            MOV     $TSTNM,@DISPLAY  ;;DISPLAY TEST NUMBER AND ERROR FLAG
3955  022602  032777  002000  156344            BIT     #BIT10,@SWR      ;;BELL ON ERROR?
3956  022610  001402                            BEQ     1$               ;;NO - SKIP
3957  022612  104401  001176                    TYPE    ,$BELL           ;;RING BELL
3958  022616  005237  001126            1$:     INC     $ERTTL           ;;COUNT THE NUMBER OF ERRORS
3959  022622  011637  001132                    MOV     (SP),$ERRPC      ;;GET ADDRESS OF ERROR INSTRUCTION
3960  022626  162737  000002  001132            SUB     #2,$ERRPC
3961  022634  117737  156272  001130            MOVB    @$ERRPC,$ITEMB   ;;STRIP AND SAVE THE ERROR ITEM CODE
3962  022642  032777  020000  156304            BIT     #BIT13,@SWR      ;;SKIP TYPEOUT IF SET
3963  022650  001004                            BNE     20$              ;;SKIP TYPEOUTS
3964  022652  004737  022752                    JSR     PC,TYPERR        ;;GO TO USER ERROR ROUTINE
3965  022656  104401  001203                    TYPE    ,$CRLF
3966  022662                            20$:
3967  022662  122737  000001  001226            CMPB    #APTENV,$ENV     ;;RUNNING IN APT MODE
3968  022670  001007                            BNE     2$               ;;NO,SKIP APT ERROR REPORT
3969  022672  113737  001130  022704            MOVB    $ITEMB,21$       ;;SET ITEM NUMBER AS ERROR NUMBER
3970  022700  004737  023502                    JSR     PC,$ATY4         ;;REPORT FATAL ERROR TO APT
3971  022704     000            21$:     .BYTE   0
3972  022705     000                     .BYTE   0
3973  022706  000777            22$:     BR      22$              ;;APT ERROR LOOP
3974  022710  005777  156240            2$:     TST     @SWR             ;;HALT ON ERROR
3975  022714  100002                            BPL     3$               ;;SKIP IF CONTINUE
3976  022716  000000                            HALT                     ;;HALT ON ERROR!
3977  022720  104407                            CKSWR                    ;;TEST FOR CHANGE IN SOFT-SWR
3978  022722  032777  001000  156224    3$:     BIT     #BIT09,@SWR      ;;LOOP ON ERROR SWITCH SET?
3979  022730  001402                            BEQ     4$               ;;BR IF NO
3980  022732  013716  001124                    MOV     $_PERR,(SP)      ;;FUDGE RETURN FOR LOOPING
3981  022736  005737  001174            4$:     TST     $ESCAPE          ;;CHECK FOR AN ESCAPE ADDRESS
3982  022742  001402                            BEQ     5$               ;;BR IF NONE
3983  022744  013716  001174                    MOV     $ESCAPE,(SP)     ;;FUDGE RETURN ADDRESS FOR ESCAPE
3984  022750                            5$:
3985  022750  000002                            RTI                      ;;RETURN
3986
3987
3988                                    ;THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE
3989                                    ;WHICH ERROR IS TO BE REPORTED, IT THEN OBTAINS, FROM THE "ERROR
3990                                    ;TABLE" ($ERRTB), AND REPORTS THE APPROPIATE INFORMATION
3991                                    ;CONCERNING THE ERROR.
3992
3993  022752  104412            TYPERR: SAVREG                           ;SAVE R0-R5
3994  022754  005000                    CLR     R0                       ;CLEAR R0 FOR ERROR NUMBER
3995  022756  113700  001130            MOVB    $ITEMB,R0                ;ERROR NUMBER
3996  022762  005300                    DEC     R0                       ;FORM INDEX FOR ERROR TABLE
3997  022764  006300                    ASL     R0
3998  022766  006300                    ASL     R0
3999  022770  006300                    ASL     R0
```

```
4000  022772  062700  006526          1$:     ADD     #$ERRTB,R0      ;FORM ADDRESS
4001  022776  012037  023012                  MOV     (R0)+,2$        ;GET ERROR MESSAGE (EM) POINTER
4002  023002  001404                          BEQ     3$              ;BRANCH IF THERE ISN'T ONE
4003  023004  104401  001203                  TYPE    ,$CRLF          ;"CARRIAGE RETURN - LINE FEED
4004  023010  104401                          TYPE
4005  023012  000000                  2$:     .WORD   0               ;"EM" POINTER GOES HERE
4006  023014  012037  023030          3$:     MOV     (R0)+,4$        ;PICK UP DATA HEADER (DH) POINTER
4007  023020  001404                          BEQ     5$              ;BRANCH IF NONE
4008  023022  104401  001203                  TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
4009  023026  104401                          TYPE
4010  023030  000000                  4$:     .WORD   0               ;"DH" POINTER GOES HERE
4011  023032  012001                  5$:     MOV     (R0)+,R1        ;PICKUP DATA TABLE (DT) POINTER
4012  023034  001460                          BEQ     20$             ;BRANCH IF NONE
4013  023036  005005                          CLR     R5              ;SET INDENT SWITCH
4014  023040  012000                          MOV     (R0)+,R0        ;DATA FORMAT (DF) POINTER
4015  023042  012002                          MOV     (R0)+,R2        ;NUMBER OF DH'S TO TYPE
4016  023044  001451                          BEQ     17$             ;BRANCH IF DH NUMBER IS 0
4017  023046  005105                          COM     R5              ;NO INDENT
4018  023050  104401  001203                  TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
4019  023054  112003                  10$:    MOVB    (R0)+,R3        ;NUMBER OF DATA WORDS TO TYPE
4020  023056  112004                          MOVB    (R0)+,R4        ;AND HOW TO TYPE THEM
4021  023060  006004                  11$:    ROR     R4              ;OCTAL OR DECIMAL?
4022  023062  103403                          BCS     12$             ;DECIMAL--BRANCH
4023  023064  013146                          MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
4024  023066  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4025  023070  000402                          BR      13$
4026  023072                          12$:
4027  023072  013146                          MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
4028  023074  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
4029  023076  005303                  13$:    DEC     R3              ;MORE NUMBERS TO TYPE?
4030  023100  001403                          BEQ     14$             ;NO--BRANCH
4031  023102  104401  035166                  TYPE    ,LINSP          ;YES--TYPE SEPERATORS
4032  023106  000764                          BR      11$             ;LOOP
4033  023110  005302                  14$:    DEC     R2              ;MORE DH'S?
4034  023112  003431                          BLE     20$             ;NO--BRANCH
4035  023114  104401  001203                  TYPE    ,$CRLF          ;YES--START A NEW LINE
4036  023120  005760  000002                  TST     2(R0)           ;ONLY A 'DH' IN THIS REQUEST ?
4037  023124  001404                          BEQ     15$             ;BR IF YES - BYPASS THE INDENT
4038  023126  005105                          COM     R5              ;INDENT?
4039  023130  001002                          BNE     15$             ;NO--BRANCH
4040  023132  104401  035166                  TYPE    ,LINSP          ;YES--TYPE SPACES
4041  023136  012037  023144          15$:    MOV     (R0)+,16$       ;GET NEXT DH
4042  023142  104401                          TYPE                    ;AND TYPE IT
4043  023144  000000                  16$:    .WORD   0               ;DH POINTER GOES HERE
4044  023146  005710                          TST     (R0)            ;TYPE A 'DT' ?
4045  023150  001003                          BNE     21$             ;BR IF A 'DT'
4046  023152  062700  000004                  ADD     #4,R0           ;INCREMENT THE 'DF' POINTER
4047  023156  000754                          BR      14$             ;SEE IF END OF 'DF' BLOCK
4048  023160  104401  001203          21$:    TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
4049  023164  005705                          TST     R5              ;INDENT?
4050  023166  001332                          BNE     10$             ;NO--BRANCH
4051  023170  104401  035166          17$:    TYPE    ,LINSP          ;YES--TYPE SPACES
4052  023174  000727                          BR      10$             ;LOOP
4053  023176  104413                  20$:    RESREG                  ;RESTORE R0-R5
4054  023200  000207                          RTS     PC              ;RETURN
4055
```

```
                                        .SBTTL  TYPE ROUTINE

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

023202  105737  001173          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
023206  100002                          BPL     1$              ;;BR IF YES
023210  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
023212  000430                          BR      3$              ;;LEAVE
023214  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
023216  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
023222  122737  000001  001226          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
023230  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
023232  132737  000100  001227          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
023240  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
023242  010037  023252                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
023246  004737  023472                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
023252  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
023254  132737  000040  001227  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
023262  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
023264  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
023266  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
023270  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
023272  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
023274  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
023300  000002                          RTI                     ;;RETURN
023302  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
023306  001430                          BEQ     8$
023310  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
023314  001006                          BNE     5$
023316  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
023320  104401                          TYPE                    ;;TYPE A CR AND LF
023322  001203                          $CRLF
023324  105037  023460                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
023330  000755                          BR      2$              ;;GET NEXT CHARACTER
023332  004737  023414          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
023336  123726  001172          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
023342  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
023344  013746  001170                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
023350  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
023354  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
023356  004737  023414                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
023362  105337  023460                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
```

```
4112  023366  000770                        BR      7$              ;;LOOP
4113
4114                                  ;HORIZONTAL TAB PROCESSOR
4115
4116  023370  112716  000040          8$:   MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
4117  023374  004737  023414          9$:   JSR     PC,$TYPEC       ;;TYPE A SPACE
4118  023400  132737  000007  023460        BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
4119  023406  001372                        BNE     9$              ;;TAB STOP
4120  023410  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
4121  023412  000724                        BR      2$              ;;GET NEXT CHARACTER
4122  023414  105777  155544        $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
4123  023420  100375                        BPL     $TYPEC
4124  023422  116677  000002  155536        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
4125  023430  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
4126  023436  001003                        BNE     1$              ;;BRANCH IF NO
4127  023440  105037  023460                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
4128  023444  000406                        BR      $TYPEX          ;;EXIT
4129  023446  122766  000012  000002  1$:   CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
4130  023454  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
4131  023456  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
4132  023460  000000        $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
4133  023462  000207        $TYPEX: RTS     PC
4134
4135
4136
4137                                  .SBTTL  APT COMMUNICATIONS ROUTINE
4138
4139                                  ;;**************************************************************
4140  023464  112737  000001  023730  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
4141  023472  112737  000001  023726  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
4142  023500  000403                        BR      $ATYC
4143  023502  112737  000001  023730  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
4144  023510                        $ATYC:
4145  023510  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
4146  023512  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
4147  023514  105737  023726                TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
4148  023520  001450                        BEQ     5$              ;;IF NOT:  BR
4149  023522  122737  000001  001226        CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
4150  023530  001031                        BNE     3$              ;;IF NOT:  BR
4151  023532  132737  000100  001227        BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
4152  023540  001425                        BEQ     3$              ;;IF NOT:  BR
4153  023542  017600  000004                MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
4154  023546  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
4155  023554  005737  001206          1$:   TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
4156  023560  001375                        BNE     1$              ;;IF NOT:  WAIT
4157  023562  010037  001222                MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
4158  023566  105720                  2$:   TSTB    (R0)+           ;;FIND END OF MESSAGE
4159  023570  001376                        BNE     2$
4160  023572  163700  001222                SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
4161  023576  006200                        ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
4162  023600  010037  001224                MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
4163  023604  012737  000004  001206        MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
4164  023612  000413                        BR      5$
4165  023614  017637  000004  023640  3$:   MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
4166  023622  062766  000002  000004        ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
4167  023630  013746  177776                MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
```

```
                                        JSR     PC,$TYPE        ;;CALL TYPE MACRO
                                4$:     .WORD   0
                                5$:
                                10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
                                        BEQ     12$             ;;IF NOT.  BR
                                        TST     $ENV            ;;RUNNING UNDER APT?
                                        BEQ     12$             ;;IF NOT.  BR
                                11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
                                        BNE     11$             ;;IF NOT.  WAIT
                                        MOV     @4(SP),$FATAL   ;;GET ERROR #
                                        ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
                                        INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
                                12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
                                        CLRB    $LFLG           ;;CLEAR LOG FLAG
                                        CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
                                        MOV     (SP)+,R1        ;;POP STACK INTO R1
                                        MOV     (SP)+,R0        ;;POP STACK INTO R0
                                        RTS     PC              ;;RETURN
                                $MFLG:  .BYTE   0               ;;MESSG. FLAG
                                $LFLG:  .BYTE   0               ;;LOG FLAG
                                $FFLG:  .BYTE   0               ;;FATAL FLAG
                                        .EVEN
                                APTSIZE=200
                                APTENV=001
                                APTSPOOL=100
                                APTCSUP=040

                                .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

                                $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
                                        MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
                                        MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
                                        ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
```

```
4224  023754  000406                          BR      $TYPON
4225  023756  112737  000001  024155  $TYPOC. MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
4226  023764  112737  000006  024157          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
4227  023772  112737  000005  024154  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
4228  024000  010346                          MOV     R3,-(SP)        ;;SAVE R3
4229  024002  010446                          MOV     R4,-(SP)        ;;SAVE R4
4230  024004  010546                          MOV     R5,-(SP)        ;;SAVE R5
4231  024006  113704  024157                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
4232  024012  005404                          NEG     R4
4233  024014  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
4234  024020  110437  024156                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
4235  024024  113704  024155                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
4236  024030  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
4237  024034  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
4238  024036  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
4239  024040  000404                          BR      3$              ;;GO DO MSB
4240  024042  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
4241  024044  006105                          ROL     R5
4242  024046  006105                          ROL     R5
4243  024050  010503                          MOV     R5,R3
4244  024052  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
4245  024054  105337  024156                  DECB    $OMODE          ;;TYPE THIS DIGIT?
4246  024060  100016                          BPL     7$              ;;BR IF NO
4247  024062  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
4248  024066  001002                          BNE     4$              ;;TEST FOR 0
4249  024070  005704                          TST     R4              ;;SUPPRESS THIS 0?
4250  024072  001403                          BEQ     5$              ;;BR IF YES
4251  024074  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
4252  024076  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
4253  024102  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
4254  024106  110337  024152                  MOVB    R3,8$           ;;SAVE FOR TYPING
4255  024112  104401  024152                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
4256  024116  105337  024154          7$:     DECB    $OCNT           ;;COUNT BY 1
4257  024122  003347                          BGT     2$              ;;BR IF MORE TO DO
4258  024124  002402                          BLT     6$              ;;BR IF DONE
4259  024126  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
4260  024130  000744                          BR      2$              ;;GO DO THE LAST DIGIT
4261  024132  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
4262  024134  012604                          MOV     (SP)+,R4        ;;RESTORE R4
4263  024136  012603                          MOV     (SP)+,R3        ;;RESTORE R3
4264  024140  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
4265  024146  012616                          MOV     (SP)+,(SP)
4266  024150  000002                          RTI                     ;;RETURN
4267  024152     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
4268  024153     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
4269  024154     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
4270  024155     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
4271  024156  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
4272
4273                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
4274
4275                                  ;;*****************************************************************
4276                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
4277                                  ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
4278                                  ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
4279                                  ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
```

```
                                        ;*REPLACED WITH SPACES.
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                                        ;*      TYPDS                   ;;GO TO THE ROUTINE
                                        $TYPDS:
024160                                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
024160  010046                                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
024162  010146                                 MOV     R2,-(SP)        ;;PUSH R2 ON STACK
024164  010246                                 MOV     R3,-(SP)        ;;PUSH R3 ON STACK
024166  010346                                 MOV     R5,-(SP)        ;;PUSH R5 ON STACK
024170  010546                                 MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
024172  012746  020200                        MOV     20(SP),R5       ;;GET THE INPUT NUMBER
024176  016605  000020                        BPL     1$              ;;BR IF INPUT IS POS.
024202  100004                                NEG     R5              ;;MAKE THE BINARY NUMBER POS.
024204  005405                                MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
024206  112766  000055  000001                CLR     R0              ;;ZERO THE CONSTANTS INDEX
024214  005000                        1$:     MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
024216  012703  024374                        MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
024222  112723  000040                CLR     R2              ;;CLEAR THE BCD NUMBER
024226  005002                        2$:     MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
024230  016001  024364                        SUB     R1,R5           ;;FORM THIS BCD DIGIT
024234  160105                        3$:     BLT     4$              ;;BR IF DONE
024236  002402                                INC     R2              ;;INCREASE THE BCD DIGIT BY 1
024240  005202                                BR      3$
024242  000774                        4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
024244  060105                                TST     R2              ;;CHECK IF BCD DIGIT=0
024246  005702                                BNE     5$              ;;FALL THROUGH IF 0
024250  001002                                TSTB    (SP)            ;;STILL DOING LEADING 0'S?
024252  105716                                BMI     7$              ;;BR IF YES
024254  100407                        5$:     ASLB    (SP)            ;;MSD?
024256  106316                                BCC     6$              ;;BR IF NO
024260  103003                                MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
024262  116663  000001  177777        6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
024270  052702  000060                7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
024274  052702  000040                        MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
024300  110223                                TST     (R0)+           ;;JUST INCREMENTING
024302  005720                                CMP     R0,#10          ;;CHECK THE TABLE INDEX
024304  020027  000010                        BLT     2$              ;;GO DO THE NEXT DIGIT
024310  002746                                BGT     8$              ;;GO TO EXIT
024312  003002                                MOV     R5,R2           ;;GET THE LSD
024314  010502                                BR      6$              ;;GO CHANGE TO ASCII
024316  000764                        8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
024320  105726                                BPL     9$              ;;BR IF NO
024322  100003                                MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
024324  116663  177777  177776                CLRB    (R3)            ;;SET THE TERMINATOR
024332  105013                        9$:     MOV     (SP)+,R5        ;;POP STACK INTO R5
024334  012605                                MOV     (SP)+,R3        ;;POP STACK INTO R3
024336  012603                                MOV     (SP)+,R2        ;;POP STACK INTO R2
024340  012602                                MOV     (SP)+,R1        ;;POP STACK INTO R1
024342  012601                                MOV     (SP)+,R0        ;;POP STACK INTO R0
024344  012600                                TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
024346  104401  024374                        MOV     2(SP),4(SP)     ;;ADJUST THE STACK
024352  016666  000002  000004                MOV     (SP)+,(SP)
024360  012616                                RTI                     ;;RETURN TO USER
024362  000002
024364  023420                        $DTBL:  10000.
```

```
4336  024366  001750                        1000.
4337  024370  000144                        100.
4338  024372  000012                        10.
4339  024374  000004              $DBLK:    .BLKW   4
4340
4341                                        .SBTTL  TTY INPUT ROUTINE
4342
4343
4344                                        ;;*****************************************************************
4345  024404  000000                        .ENABL  LSB
4346  024406  000000              $TKCNT:   .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
4347  024410  000000              $TKQIN:   .WORD   0               ;;INPUT POINTER
4348  024412  000007              $TKQOUT:  .WORD   0               ;;OUTPUT POINTER
4349          024421              $TKQSRT:  .BLKB   7               ;;TTY KEYBOARD QUEUE
4350          024422              $TKQEND=.
4351                                        .EVEN
4352
4353                                        ;*TK INITIALIZE ROUTINE
4354                                        ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
4355                                        ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
4356                                        ;
4357                                        ;*CALL:
4358                                        ;*      JSR     PC,$TKINT
4359                                        ;*      RETURN
4360                                        ;
4361  024422  005037  024404        $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
4362  024426  012737  024412  024406         MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
4363  024434  013737  024406  024410         MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
4364  024442  012737  024472  000060         MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
4365  024450  012737  000200  000062         MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
4366  024456  005777  154500                 TST     @$TKB           ;;CLEAR DONE FLAG
4367  024462  012777  000100  154470         MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
4368  024470  000207                         RTS     PC              ;;RETURN TO CALLER
4369
4370                                        ;*TK SERVICE ROUTINE
4371                                        ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
4372                                        ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
4373                                        ;*IT IN THE QUEUE.
4374                                        ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
4375                                        ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (@CNTLC)
4376  024472  117746  154464        $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
4377  024476  042716  177600                 BIC     #^C177,(SP)     ;;STRIP THE JUNK
4378  024502  021627  000003                 CMP     (SP),#3         ;;IS IT A CONTROL C?
4379  024506  001007                         BNE     1$              ;;BRANCH IF NO
4380  024510  104401  025613                 TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
4381  024514  004737  024422                 JSR     PC,$TKINT       ;;INIT THE KEYBOARD
4382  024520  005726                         TST     (SP)+           ;;CLEAN UP STACK
4383  024522  000177  154650                 JMP     @CNTLC          ;;CONTROL C RESTART
4384  024526  021627  000007         1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
4385  024532  001004                         BNE     2$              ;;BRANCH IF NO
4386  024534  022737  000176  001154         CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
4387  024542  001500                         BEQ     6$              ;;GO TO SWR CHANGE
4388
4389  024544                         2$:
4390  024544  022737  000007  024404         CMP     #7,$TKCNT       ;;IS THE QUEUE FULL?
4391  024552  001004                         BNE     3$              ;;BRANCH IF NO
```

```
 4392  024554  104401  001176            TYPE    $BELL           ;;RING THE TTY BELL
 4393  024560  005726                    TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
 4394  024562  000451                    BR      5$              ;;EXIT
 4395  024564  021627  000023    3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
 4396  024570  001021                    BNE     32$             ;;BRANCH IF NO
 4397  024572  005077  154362            CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
 4398  024576  005726                    TST     (SP)+           ;;CLEAN CHAR OFF STACK
 4399  024600  105777  154354    31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
 4400  024604  100375                    BPL     31$             ;;LOOP UNTIL ITS THERE
 4401  024606  117746  154350            MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
 4402  024612  042716  177600            BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
 4403  024616  022627  000021            CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
 4404  024622  001366                    BNE     31$             ;;BRANCH IF NO
 4405  024624  012777  000100  154326    MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
 4406  024632  000002                    RTI                     ;;RETURN
 4407  024634  005237  024404    32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
 4408  024640  021627  000140            CMP     (SP),#140       ;;IS IT UPPER CASE?
 4409  024644  002405                    BLT     4$              ;;BRANCH IF YES
 4410  024646  021627  000175            CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
 4411  024652  003002                    BGT     4$              ;;BRANCH IF YES
 4412  024654  042716  000040            BIC     #40,(SP)        ;;MAKE IT UPPER CASE
 4413  024660  112677  177522    4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
 4414  024664  005237  024406            INC     $TKQIN          ;;UPDATE THE POINTER
 4415  024670  023727  024406  024421    CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
 4416  024676  001003                    BNE     5$              ;;BRANCH IF NO
 4417  024700  012737  024412  024406    MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
 4418  024706  000002            5$:     RTI                     ;;RETURN
 4419
 4420                                    ;;**************************************************************
 4421                                    ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
 4422                                    ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
 4423                                    ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
 4424                                    ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
 4425  024710  022737  000176  001154  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
 4426  024716  001124                    BNE     15$             ;;EXIT IF NOT
 4427  024720  105777  154234            TSTB    @$TKS           ;;IS A CHAR WAITING?
 4428  024724  100121                    BPL     15$             ;;IF NOT, EXIT
 4429  024726  117746  154230            MOVB    @$TKB,-(SP)     ;;YES
 4430  024732  042716  177600            BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
 4431  024736  021627  000007            CMP     (SP),#7         ;;IS IT A CONTROL-G?
 4432  024742  001300                    BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
 4433                                    ;;AND EXIT
 4434
 4435                                    ;;**************************************************************
 4436                                    ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
 4437                                    ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
 4438                                    ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
 4439  024744  123727  001150  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
 4440  024752  001674                    BEQ     2$              ;;BRANCH IF YES
 4441  024754  005726                    TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
 4442  024756  004737  024422            JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
 4443  024762  005077  154172            CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
 4444  024766  112737  000001  001151    MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
 4445
 4446  024774  104401  025625            TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
 4447  025000  104401  025632  $GTSWR:  TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
```

```
4448  025004  013746  000176                    MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
4449  025010  104402                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
4450  025012  104401  025643                    TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
4451  025016  005046                  19$:      CLR     -(SP)           ;;CLEAR COUNTER
4452  025020  005046                            CLR     -(SP)           ;;THE NEW SWR
4453  025022  105777  154132          7$:       TSTB    @$TKS           ;;CHAR THERE?
4454  025026  100375                            BPL     7$              ;;IF NOT TRY AGAIN
4455
4456  025030  117746  154126                    MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
4457  025034  042716  177600                    BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
4458
4459  025040  021627  000003                    CMP     (SP),#3         ;;IS IT A CONTROL-C?
4460  025044  001015                            BNE     9$              ;;BRANCH IF NOT
4461  025046  104401  025613                    TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
4462  025052  062706  000006                    ADD     #6,SP           ;;CLEAN UP STACK
4463  025056  123727  001151  000001            CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
4464  025064  001003                            BNE     8$              ;;BRANCH IF NO
4465  025066  012777  000100  154064            MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
4466  025074  000177  154276          8$:       JMP     @CNTLC          ;;CONTROL-C RESTART
4467
4468
4469  025100  021627  000025          9$:       CMP     (SP),#25        ;;IS IT A CONTROL-U?
4470  025104  001005                            BNE     10$             ;;BRANCH IF NOT
4471  025106  104401  025620                    TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
4472  025112  062706  000006          20$:      ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
4473  025116  000737                            BR      19$             ;;LET'S TRY IT AGAIN
4474
4475
4476  025120  021627  000015          10$:      CMP     (SP),#15        ;;IS IT A <CR>?
4477  025124  001022                            BNE     16$             ;;BRANCH IF NO
4478  025126  005766  000004                    TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
4479  025132  001403                            BEQ     11$             ;;BRANCH IF YES
4480  025134  016677  000002  154012            MOV     2(SP),@SWR      ;;SAVE NEW SWR
4481  025142  062706  000006          11$:      ADD     #6,SP           ;;CLEAR UP STACK
4482  025146  104401  001203          14$:      TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
4483  025152  123727  001151  000001            CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
4484  025160  001003                            BNE     15$             ;;BRANCH IF NOT
4485  025162  012777  000100  153770            MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
4486  025170  000002                  15$:      RTI                     ;;RETURN
4487  025172  004737  023414          16$:      JSR     PC,$TYPEC       ;;ECHO CHAR
4488  025176  021627  000060                    CMP     (SP),#60        ;;CHAR < 0?
4489  025202  002420                            BLT     18$             ;;BRANCH IF YES
4490  025204  021627  000067                    CMP     (SP),#67        ;;CHAR > 7?
4491  025210  003015                            BGT     18$             ;;BRANCH IF YES
4492  025212  042726  000060                    BIC     #60,(SP)+       ;;STRIP-OFF ASCII
4493  025216  005766  000002                    TST     2(SP)           ;;IS THIS THE FIRST CHAR
4494  025222  001403                            BEQ     17$             ;;BRANCH IF YES
4495  025224  006316                            ASL     (SP)            ;;NO, SHIFT PRESENT
4496  025226  006316                            ASL     (SP)            ;;    CHAR OVER TO MAKE
4497  025230  006316                            ASL     (SP)            ;;    ROOM FOR NEW ONE.
4498  025232  005266  000002          17$.      INC     2(SP)           ;;KEEP COUNT OF CHAR
4499  025236  056616  177776                    BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
4500  025242  000667                            BR      7$              ;;GET THE NEXT ONE
4501  025244  104401  001202          18$:      TYPE    ,$QUES          ;;TYPE ?<CR><LF>
4502  025250  000720                            BR      20$             ;;SIMULATE CONTROL-U
4503                                  .DSABL    LSB
```

```
                                ;;*****************************************************************
                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*CALL:
                                ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
                                ;
025252  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
025254  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE PS
025262  005066  000004                  CLR     4(SP)           ;;GET READY FOR A CHARACTER
025266  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
025270  012746  025276                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
025274  000002                          RTI                     ;;POP NEW PC AND PS
025276                          64$:
025276  005737  024404          1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
025302  001775                          BEQ     1$
025304  005337  024404                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
025310  117766  177074  000004          MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
025316  005237  024410                  INC     $TKQOUT         ;;UPDATE THE POINTER
025322  023727  024410  024421          CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
025330  001003                          BNE     2$              ;;BRANCH IF NO
025332  012737  024412  024410          MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
025340  000002                  2$:     RTI                     ;;RETURN
                                ;;*****************************************************************
                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;*CALL:
                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
                                ;
025342  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
025344  005046                          CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
025346  012703  025576          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
025352  022703  025613          2$:     CMP     #$TTYIN+15,R3   ;;BUFFER FULL?
025356  101456                          BLOS    4$              ;;BR IF YES
025360  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
025362  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
025364  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
025370  001022                          BNE     5$              ;;BR IF NO
025372  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
025374  001007                          BNE     6$              ;;BR IF NO
025376  112737  000134  025574          MOVB    #'\,9$          ;;TYPE A BACK SLASH
025404  104401  025574                  TYPE    ,9$
025410  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
025414  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
025416  020327  025576                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
025422  103434                          BLO     4$              ;;BR IF YES
025424  111337  025574                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
025430  104401  025574                  TYPE    ,9$             ;;GO TYPE
025434  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
025436  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
025440  001406                          BEQ     7$              ;;BR IF NO
025442  112737  000134  025574          MOVB    #' ,9$          ;;TYPE A BACK SLASH
```

```
4560  025450  104401  025574                    TYPE    ,9$
4561  025454  005016                            CLR     (SP)            ;;CLEAR THE RUBOUT KEY
4562  025456  122713  000025          7$:       CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
4563  025462  001003                            BNE     8$              ;;BR IF NO
4564  025464  104401  025620                    TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
4565  025470  000726                            BR      1$              ;;GO START OVER
4566  025472  122713  000022          8$:       CMPB    #22,(R3)        ;;IS CHARACTER A "↑R"?
4567  025476  001011                            BNE     3$              ;;BRANCH IF NO
4568  025500  105013                            CLRB    (R3)            ;;CLEAR THE CHARACTER
4569  025502  104401  001203                    TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
4570  025506  104401  025576                    TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
4571  025512  000717                            BR      2$              ;;GO PICKUP ANOTHER CHACTER
4572  025514  104401  001202          4$:       TYPE    ,$QUES          ;;TYPE A '?'
4573  025520  000712                            BR      1$              ;;CLEAR THE BUFFER AND LOOP
4574  025522  111337  025574          3$:       MOVB    (R3),9$         ;;ECHO THE CHARACTER
4575  025526  104401  025574                    TYPE    ,9$
4576  025532  122723  000015                    CMPB    #15,(R3)+       ;;CHECK FOR RETURN
4577  025536  001305                            BNE     2$              ;;LOOP IF NOT RETURN
4578  025540  105063  177777                    CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
4579  025544  104401  001204                    TYPE    ,$LF            ;;TYPE A LINE FEED
4580  025550  005726                            TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
4581  025552  012603                            MOV     (SP)+,R3        ;;RESTORE R3
4582  025554  011646                            MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
4583  025556  016666  000004  000002            MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
4584  025564  012766  025576  000004            MOV     #$TTYIN,4(SP)
4585  025572  000002                            RTI                     ;;RETURN
4586  025574     000          9$:       .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
4587  025575     000                    .BYTE   0                       ;;TERMINATOR
4588  025576  000015          $TTYIN:   .BLKB   15                      ;;RESERVE 15 BYTES FOR TTY INPUT
4589  025613     136  006503  000012    $CNTLC:   .ASCIZ  /↑C/<15><12>  ;;CONTROL "C"
4590  025620  052536  005015     000    $CNTLU:   .ASCIZ  /↑U/<15><12>  ;;CONTROL "U"
4591  025625     136  006507  000012    $CNTLG:   .ASCIZ  /↑G/<15><12>  ;;CONTROL "G"
4592  025632  005015  053523  020122    $MSWR:    .ASCIZ  <15><12>/SWR = /
4593  025640  020075     000
4594  025643     040  047040  053505    $MNEW:    .ASCIZ  /  NEW = /
4595  025650  036440  000040
4596
4597                                     .SBTTL  POWER DOWN AND UP ROUTINES
4598
4599                                     ;;*****************************************************************
4600                                     ;POWER DOWN ROUTINE
4601  025654  012737  026020  000024    $PWRDN:   MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
4602  025662  012737  000340  000026              MOV     #340,@#PWRVEC+2 ;;PRIO:7
4603  025670  010046                            MOV     R0,-(SP)        ;;PUSH R0 ON STACK
4604  025672  010146                            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
4605  025674  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
4606  025676  010346                            MOV     R3,-(SP)        ;;PUSH R3 ON STACK
4607  025700  010446                            MOV     R4,-(SP)        ;;PUSH R4 ON STACK
4608  025702  010546                            MOV     R5,-(SP)        ;;PUSH R5 ON STACK
4609  025704  017746  153244                    MOV     @$SWR,-(SP)     ;;PUSH @$SWR ON STACK
4610  025710  010637  026024                    MOV     SP,$SAVR6       ;;SAVE SP
4611  025714  012737  025726  000024            MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
4612  025722  000000                            HALT
4613  025724  000776                            BR      .-2             ;;HANG UP
4614
4615                                     ;*****************************************************************
```

```
                                        ;POWER UP ROUTINE
026726  012737  026020  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
026734  013706  026024                  MOV     $SAVR6,SP        ;;GET SP
026740  005037  026024                  CLR     $SAVR6           ;;WAIT LOOP FOR THE TTY
026744  005237  026024          1$:     INC     $SAVR6           ;;WAIT FOR THE INC
026750  001375                  1$      BNE     1$               ;;OF  WORD
026752  012677  153176                  MOV     (SP)+,@SWR       ;;POP STACK INTO @SWR
026756  012605                          MOV     (SP)+,R5         ;;POP STACK INTO R5
026760  012604                          MOV     (SP)+,R4         ;;POP STACK INTO R4
026762  012603                          MOV     (SP)+,R3         ;;POP STACK INTO R3
026764  012602                          MOV     (SP)+,R2         ;;POP STACK INTO R2
026766  012601                          MOV     (SP)+,R1         ;;POP STACK INTO R1
026770  012600                          MOV     (SP)+,R0         ;;POP STACK INTO R0
026772  012737  025654  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
026000  012737  000340  000026          MOV     #340,@#PWRVEC+2  ;;PRIO:7
026006  104401                          TYPE                     ;;REPORT THE POWER FAILURE
026010  036417                  $PWRMG: .WORD   PMSG             ;;POWER FAIL MESSAGE POINTER
026012  012716                          MOV     (PC)+,(SP)       ;;RESTART AT BEGIN2
026014  007034                  $PWRAD: .WORD   BEGIN2           ;;RESTART ADDRESS
026016  000002                          RTI
026020  000000                  $ILLUP: HALT                     ;;THE POWER UP SEQUENCE WAS STARTED
026022  000776                          BR      .-2              ;; BEFORE THE POWER DOWN WAS COMPLETE
026024  000000                  $SAVR6: 0                        ;;PUT THE SP HERE

                                        .SBTTL  TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS

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

026026  010046                  $SUPRS: MOV     R0,-(SP)         ;;SAVE R0
026030  016600  000004                  MOV     4(SP),R0         ;;PICKUP THE POINTER
026034  105710                  1$:     TSTB    (R0)             ;;TERMINATEOR?
026036  001403                          BEQ     2$               ;;BR IF YES
026040  122720  000060                  CMPB    #'0,(R0)+        ;;IS THIS AN ASCII "0" ?
026044  001773                          BEQ     1$               ;;BR IF YES
026046  005300                  2$:     DEC     R0               ;;BACKUP BY "1"
026050  010037  026056                  MOV     R0,3$            ;;SAVE FOR TYPING
026054  104401                          TYPE                     ;;GO TYPE
026056  000000                  3$:     .WORD   0                ;;ASCIZ POINTER GOES HERE
026060  012600                          MOV     (SP)+,R0         ;;RESTORE R0
026062  012616                          MOV     (SP)+,(SP)       ;;RESTORE THE STACK
026064  000207                          RTS     PC               ;;RETURN

                                        .SBTTL  SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE

                                        ;;**************************************************************
                                        ;*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
                                        ;*UNSIGNED DECIMAL ASCIZ NUMBER.
                                        ;*CALL
                                        ;*      MOV     NUMBER,-(SP)    ;;PUT BINARY NUMBER ON THE STACK
                                        ;*      JSR     PC,@#$S2D       ;;CALL
```

```
                                        ;*      RETURN                  ;;ADDRESS OF THE 1ST ASCIZ CHAR.IS ON THE STACK
                                        ;
                                        ;
026066  016637  000002  026116 $SB2D:   MOV     2(SP),1$                ;;SAVE BINARY NUMBER
026074  012746  026116                  MOV     #1$,-(SP)               ;;SET POINTER
026100  004737  026122                  JSR     PC,@#$DB2D              ;;CALL DOUBLE LENGTH CONVERT
026104  062716  000005                  ADD     #5,(SP)                 ;;ONLY ALLOW FIVE CHARACTERS
026110  012666  000002                  MOV     (SP)+,2(SP)             ;;PICKUP POINTER
026114  000207                          RTS     PC                      ;;RETURN
026116  000000  000000          1$:     .WORD   0,0
                                        ;
                                        .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
                                        ;
                                        ;;*****************************************************************
                                        ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
                                        ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
                                        ;*POSITIVE.
                                        ;*CALL
                                        ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
                                        ;*      JSR     PC,@#$DB2D
                                        ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
                                        ;;IS ON THE STACK
                                        ;
                                        ;
026122  104412          $DB2D:  SAVREG                                  ;;SAVE REGISTERS
026124  016602  000002                  MOV     2(SP),R2                ;;PICKUP THE DATA POINTER
026130  012700  026302                  MOV     #$DECVL,R0              ;;GET ADDRESS OF "$DECVL" STRING
026134  010066  000002                  MOV     R0,2(SP)                ;;PUT ADDRESS OF ASCIZ STRING ON STACK
026140  012201                          MOV     (R2)+,R1                ;;PICKUP THE BINARY NUMBER
026142  012202                          MOV     (R2)+,R2
026144  012737  000012  026220          MOV     #10.,4$                 ;;SET UP TO DO 10 CONVERSIONS
026152  012704  026232                  MOV     #$TNPWR,R4              ;;ADDRESS OF TEN POWER
026156  012705  026234                  MOV     #$TNPWR+2,R5
026162  005003                  1$:     CLR     R3                      ;;CLEAR PARTIAL
026164  161401                  2$:     SUB     (R4),R1                 ;;SUBTRACT TEN POWER
026166  005602                          SBC     R2
026170  161502                          SUB     (R5),R2
026172  002402                          BLT     3$                      ;;BR IF TEN POWER TO LARGE
026174  005203                          INC     R3                      ;;ADD 1 TO PARTIAL
026176  000772                          BR      2$                      ;;LOOP
026200  062401                  3$:     ADD     (R4)+,R1                ;;RESTORE SUBTRACTED VALUE
026202  005502                          ADC     R2
026204  062402                          ADD     (R4)+,R2
026206  022525                          CMP     (R5)+,(R5)+             ;;MOVE TO NEXT TEN POWER
026210  052703  000060                  BIS     #'0,R3                  ;;CHANGE PARTIAL TO ASCII
026214  110320                          MOVB    R3,(R0)+                ;;SAVE IT
026216  005327                          DEC     (PC)+                   ;;DONE?
026220  000000                  4$:     .WORD   0
026222  001357                          BNE     1$                      ;;BR IF NO
026224  105020                          CLRB    (R0)+                   ;;TERMINATOR
026226  104413                          RESREG                          ;;RESTORE REGISTERS
026230  000207                          RTS     PC                      ;;RETURN
026232  145000          $TNPWR: 145000                                  ;;1.0E09
026234  035632                  35632
026236  160400                  160400                                  ;;1.0E08
026240  002765                  2765
```

```
4728  026242  113200                          113200                  ;;1.0E07
4729  026244  000230                          230
4730  026246  041100                          041100                  ;;1.0E06
4731  026250  000017                          17
4732  026252  103240                          103240                  ;;1.0E05
4733  026254  000001                          1
4734  026256  023420                          23420                   ;;1.0E04
4735  026260  000000                          0
4736  026262  001750                          1750                    ;;1.0E03
4737  026264  000000                          0
4738  026266  000144                          144                     ;;1.0E02
4739  026270  000000                          0
4740  026272  000012                          12                      ;;1.0E01
4741  026274  000000                          0
4742  026276  000001                          1                       ;;1.0E00
4743  026300  000000                          0
4744  026302  000014                $DECVL: .BLKB   12.               ;;RESERVE STORAGE FOR ASCIZ STRING
4745
4746                                .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
4747
4748                                ;;*****************************************************************
4749                                ;*SAVE R0-R5
4750                                ;*CALL:
4751                                ;*      SAVREG
4752                                ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
4753                                ;*
4754                                ;*TOP---(+16)
4755                                ;* +2---(+18)
4756                                ;* +4---R5
4757                                ;* +6---R4
4758                                ;* +8---R3
4759                                ;*+10---R2
4760                                ;*+12---R1
4761                                ;*+14---R0
4762
4763  026316                        $SAVREG:
4764  026316  010046                        MOV     R0,-(SP)          ;;PUSH R0 ON STACK
4765  026320  010146                        MOV     R1,-(SP)          ;;PUSH R1 ON STACK
4766  026322  010246                        MOV     R2,-(SP)          ;;PUSH R2 ON STACK
4767  026324  010346                        MOV     R3,-(SP)          ;;PUSH R3 ON STACK
4768  026326  010446                        MOV     R4,-(SP)          ;;PUSH R4 ON STACK
4769  026330  010546                        MOV     R5,-(SP)          ;;PUSH R5 ON STACK
4770  026332  016646  000022                MOV     22(SP),-(SP)      ;;SAVE PS OF MAIN FLOW
4771  026336  016646  000022                MOV     22(SP),-(SP)      ;;SAVE PC OF MAIN FLOW
4772  026342  016646  000022                MOV     22(SP),-(SP)      ;;SAVE PS OF CALL
4773  026346  016646  000022                MOV     22(SP),-(SP)      ;;SAVE PC OF CALL
4774  026352  000002                        RTI
4775
4776                                ;*RESTORE R0-R5
4777                                ;*CALL:
4778                                ;*      RESREG
4779  026354                        $RESREG:
4780  026354  012666  000022                MOV     (SP)+,22(SP)      ;;RESTORE PC OF CALL
4781  026360  012666  000022                MOV     (SP)+,22(SP)      ;;RESTORE PS OF CALL
4782  026364  012666  000022                MOV     (SP)+,22(SP)      ;;RESTORE PC OF MAIN FLOW
4783  026370  012666  000022                MOV     (SP)+,22(SP)      ;;RESTORE PS OF MAIN FLOW
```

```
4784  026374  012605                  MOV     (SP)+,R5        ;;POP STACK INTO R5
4785  026376  012604                  MOV     (SP)+,R4        ;;POP STACK INTO R4
4786  026400  012603                  MOV     (SP)+,R3        ;;POP STACK INTO R3
4787  026402  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
4788  026404  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
4789  026406  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
4790  026410  000002                  RTI
4791
4792                                  .SBTTL  TRAP DECODER
4793
4794                                  ;+*********************************************************
4795                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
4796                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
4797                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
4798                                  ;*GO TO THAT ROUTINE.
4799
4800  026412  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
4801  026414  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
4802  026420  005740                  TST     -(R0)           ;;BACKUP BY 2
4803  026422  111000                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
4804  026424  006300                  ASL     R0              ;;POSITION FOR INDEXING
4805  026426  016000  026446          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
4806  026432  000200                  RTS     R0              ;;GO TO ROUTINE
4807
4808
4809                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
4810
4811  026434  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
4812  026436  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
4813  026444  000002                  RTI                     ;;RESTORE THE PSW
4814
4815                                  .SBTTL  TRAP TABLE
4816
4817                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
4818                                  ;*BY THE "TRAP" INSTRUCTION.
4819
4820                                  ;       ROUTINE
4821                                  ;       -------
4822  026446  026434                  $TRPAD: .WORD   $TRAP2
4823  026450  023202                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
4824  026452  023756                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
4825  026454  023732                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
4826  026456  023772                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
4827  026460  024160                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
4828
4829  026462  025000                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
4830
4831  026464  024710                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
4832  026466  025252                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
4833  026470  025342                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
4834  026472  026316                          $SAVREG ;;CALL=SAVREG   TRAP+12(104412) SAVE R0-R5 ROUTINE
4835  026474  026354                          $RESREG ;;CALL=RESREG   TRAP+13(104413) RESTORE R0-R5 ROUTINE
4836
4837
4838
4839                                  .SBTTL  SINGLE/DUAL PORT RH70/RM03 DRIVER (REV 5.1)-24-AUG-77
```

```
                                        ;NEW DRIVE TYPE ID FOR RM02 **********
                                        ; 10-AUG-77 ***********

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

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

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

026476  000000  000000  000000  RMERRS: .WORD   0,0,0,0
026504  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

026506      000                 DRVACT: .BYTE   0               ;DRIVE 0
026507      000                         .BYTE   0               ;DRIVE 1
026510      000                         .BYTE   0               ;DRIVE 2
026511      000                         .BYTE   0               ;DRIVE 3
026512      000                         .BYTE   0               ;DRIVE 4
026513      000                         .BYTE   0               ;DRIVE 5
026514      000                         .BYTE   0               ;DRIVE 6
026515      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

026516      000                 DRVSTA: .BYTE   0               ;DRIVE 0
026517      000                         .BYTE   0               ;DRIVE 1
026520      000                         .BYTE   0               ;DRIVE 2
026521      000                         .BYTE   0               ;DRIVE 3
026522      000                         .BYTE   0               ;DRIVE 4
026523      000                         .BYTE   0               ;DRIVE 5
026524      000                         .BYTE   0               ;DRIVE 6
026525      000                         .BYTE   0               ;DRIVE 7

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

026526      000                 DRVTYP: .BYTE   0               ;DRIVE 0
026527      000                         .BYTE   0               ;DRIVE 1
026530      000                         .BYTE   0               ;DRIVE 2
```

```
4896  026531  000                       .BYTE   0               ;DRIVE 3
4897  026532  000                       .BYTE   0               ;DRIVE 4
4898  026533  000                       .BYTE   0               ;DRIVE 5
4899  026534  000                       .BYTE   0               ;DRIVE 6
4900  026535  000                       .BYTE   0               ;DRIVE 7
4901
4902                              ;TABLE OF DUAL PORT INITIALIZATION INDICATORS
4903                                      ;DPINT=0 IF INITIALIZATION IS NOT ACTIVE ON THE DRIVE
4904                                      ;DPINT<0 IF INITIALIZATION IS IN PROGRESS
4905
4906  026536  000                   DPINT:  .BYTE   0               ;DRIVE 0
4907  026537  000                           .BYTE   0               ;DRIVE 1
4908  026540  000                           .BYTE   0               ;DRIVE 2
4909  026541  000                           .BYTE   0               ;DRIVE 3
4910  026542  000                           .BYTE   0               ;DRIVE 4
4911  026543  000                           .BYTE   0               ;DRIVE 5
4912  026544  000                           .BYTE   0               ;DRIVE 6
4913  026545  000                           .BYTE   0               ;DRIVE 7
4914
4915                              ;TABLE OF PENDING DUAL PORT REQUESTS
4916                                      ;DPRQS=0 IF THAT A DUAL PORT REQUEST IS NOT PENDING FOR THAT DRIVE
4917                                      ;DPRQS<0 IF THAT A DUAL PORT REQUEST IS PENDING FOR THAT DRIVE
4918
4919  026546  000                   DPRQS:  .BYTE   0               ;DRIVE 0
4920  026547  000                           .BYTE   0               ;DRIVE 1
4921  026550  000                           .BYTE   0               ;DRIVE 2
4922  026551  000                           .BYTE   0               ;DRIVE 3
4923  026552  000                           .BYTE   0               ;DRIVE 4
4924  026553  000                           .BYTE   0               ;DRIVE 5
4925  026554  000                           .BYTE   0               ;DRIVE 6
4926  026555  000                           .BYTE   0               ;DRIVE 7
4927
4928                              ;TRANSFER WAIT FLAG (TRNSWT=1 WORD)
4929                                      ;THIS IS A ONE WORD QUEUE  IT WILL CONTAIN THE ADDRESS OF
4930                                      ;"DPB" OF THE I/O OPERATION.
4931
4932  026556  000000                TRNSWT: .WORD   0
4933
4934                              ;SEARCH WAIT KEYS (SRCHWT=1 WORD)
4935                                      ;THIS IS A ONE WORD QUEUE THAT WILL CONTAIN A KEY FOR EACH OF
4936                                      ;THE DRIVES THAT ARE PERFORMING A SEARCH COMMAND FOR THE I/O
4937                                      ;REQUEST THAT IS AT THE TOP OF THEIR REQUEST QUEUE.
4938                                      ;EACH DRIVE IS ASSIGNED ONE BIT, STARTING AT BIT00 FOR DRIVE 0.
4939
4940  026560  000000                SRCHWT: .WORD   0
4941
4942                              ;RM03 DRIVER ACTIVE FLAG (ACTDRV=1 BYTE)
4943                                      ;ACTDRV=0 IF DRIVER IS INACTIVE
4944                                      ;ACTDRV>0 IF DRIVER IS ACTIVE
4945
4946  026562  000                   ACTDRV: .BYTE   0
4947
4948                              ;SOFTWARE TIMER ROUTINE ACTIVE FLAG (ACTSTR=1 BYTE)
4949                                      ;ACTSTR=0 IF SOFTWARE TIMER ROUTINE IS INACTIVE
4950                                      ;ACTSTR>0 IF SOFTWARE TIMER ROUTINE IS ACTIVE
4951
```

```
4952  026563  000                 ACTSTR: .BYTE   0
4953
4954                              ,UNLOAD FLAG (ULDFLG=8 BYTES)
4955                                      ;ULDFLG=0 IF NO UNLOAD COMMAND
4956                                      ;ULDFLG>0 IF UNLOAD COMMAND IN PROGRESS
4957                                      ;ULDFLG<0 IF UNLOAD COMMAND IN WAIT QUEJE
4958
4959  026564  000                 ULDFLG: .BYTE   0               ;DRIVE 0
4960  026565  000                         .BYTE   0               ;DRIVE 1
4961  026566  000                         .BYTE   0               ;DRIVE 2
4962  026567  000                          BYTE   0               ;DRIVE 3
4963  026570  000                         .BYTE   0               ;DRIVE 4
4964  026571  000                         .BYTE   0               ;DRIVE 5
4965  026572  000                         .BYTE   0               ;DRIVE 6
4966  026573  000                         .BYTE   0               ;DRIVE 7
4967
4968                              ;LOOK AHEAD COUNT (LACNT=8 BYTES)
4969                                      ;LACNT WILL INDICATE THE NUMBER OF LOOK AHEADS PERFORMED
4970
4971  026574  000                 LACNT:   BYTE   0               ;DRIVE 0
4972  026575  000                         .BYTE   0               ;DRIVE 1
4973  026576  000                         .BYTE   0               ;DRIVE 2
4974  026577  000                         .BYTE   0               ;DRIVE 3
4975  026600  000                         .BYTE   0               ;DRIVE 4
4976  026601  000                         .BYTE   0               ;DRIVE 5
4977  026602  000                         .BYTE   0               ;DRIVE 6
4978  026603  000                         .BYTE   0               ;DRIVE 7
4979
4980                              ;SAVE REGISTERS FLAG (SAVEFG =1 WORD)
4981                                      ;SAVEFG <0 IF SAVE THE RH70/RM03 REGISTERS WHEN THE
4982                                      ;OPERATION IS COMPLETED AS PER (DPB+14).
4983                                      ;SAVEFG=0 IF SAVE THE RH70/RM03 REGISTERS, AS PER
4984                                      ;(DPB+14), AFTER AN ERROR.
4985
4986  026604  000000              SAVEFG: .WORD   0
4987
4988                              ;SEEK FLAG (SEEKFG=1 WORD)
4989                                      ;SEEKFG=0 IF WHEN THE DISK ADDRESS ISN'T IN THE WINDOW
4990                                      ;FOR A DATA TRANSFER START A SEARCH COMMAND
4991                                      ;SEEKFG<0 IF DATA TRANSFER WILL DO IMPLIED SEEKS,
4992                                      ;DISREGARD THE WINDOW
4993
4994  026606  177777              SEEKFG: .WORD   -1
4995
4996                              ;TIMEOUT TABLE (TIMER=8 WORDS)
4997                                      ;THIS TABLE CONTAINS THE TIME ALLOWED FOR AN OPERATION
4998
4999  026610  177777              TIMER:  .WORD   -1      ;DRIVE 0
5000  026612  177777                      .WORD   -1      ;DRIVE 1
5001  026614  177777                      .WORD   -1      ;DRIVE 2
5002  026616  177777                      .WORD   -1      ;DRIVE 3
5003  026620  177777                      .WORD   -1      ;DRIVE 4
5004  026622  177777                      .WORD   -1      ;DRIVE 5
5005  026624  177777                      .WORD   -1      ;DRIVE 6
5006  026626  177777                      .WORD   -1      ;DRIVE 7
5007
```

```
                                ;DATA TRANSFER UNDERWAY INDICATOR (DTUW=1 WORD)
                                        ;DTUW<0 IF NO DATA TRANSFER UNDERWAY
                                        ;DTUW=+N (WHERE N=0 TO 7) IMPLIES DATA TRANSFER UNDERWAY ON DRIVE N

026630  177777                  DTUW:   .WORD   -1

                                ;ATTENTION BITS TABLE (ATABIT=8 BYTES)
                                        ;THIS TABLE CONTAINS THE CORRESPONDING BIT TO EACH DRIVES
                                        ;ATTENTION BIT

026632      001                 ATABIT: .BYTE   1       ;DRIVE 0
026633      002                         .BYTE   2       ;DRIVE 1
026634      004                         .BYTE   4       ;DRIVE 2
026635      010                         .BYTE   10      ;DRIVE 3
026636      020                         .BYTE   20      ;DRIVE 4
026637      040                         .BYTE   40      ;DRIVE 5
026640      100                         .BYTE   100     ;DRIVE 6
026641      200                         .BYTE   200     ;DRIVE 7

                                ;FSRM03 TO RH70 "MASSBUS CONTROL BUS PARITY ERRORS" (MCPE) ALLOWED BEFORE
                                ;CALLING IT FATAL (MCPEMX=1 WORD)

026642  000003                  MCPEMX: .WORD   3

                                ;STORAGE FOR RMADR (THE FIRST ADDRESS (776700) OF THE RH70/RM03),
                                ;RMVEC (THE VECTOR ADDRESS (254)), AND RMVEC+2 (THE BR LEVEL (5)).

026644  176700                  RMADR:  .WORD   176700
026646  000254  000240          RMVEC:  .WORD   254,5*32.

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

026652  000004                  MXLACT: .WORD   4
                                ;MAXIMUM DELTA DELAY IS 8 SECTORS (MXDLTA=1 WORD)

026654  001000                  MXDLTA: .WORD   8.*64.
                                ;MINIMUM DELTA DELAY IS 2 SECTORS (MNDLTA=1 WORD)

026656  000200                  MNDLTA: .WORD   2*64.
                                ;MAXIMUM SEARCH FOR I/O WINDOW IS 5 SECTORS (MXWNDW=1 WORD)

026660  000005                  MXWNDW: .WORD   5

                                ;DEFINITIONS OF THE RH70/RM03 ADDRESS INDEXES

000000                          RMCS1=0                 ;CONTROL AND STATUS REGISTER #1 (DRIVE REG. 00)
000002                          RMWC=2                  ;WORD COUNT REGISTER (NOT A DRIVE REG)
000004                          RMBA=4                  ;UNIBUS ADDRESS REGISTER (NOT A DRIVE REG)
000006                          RMDA=6                  ;DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG. 05)
000010                          RMCS2=10                ;CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG)
000012                          RMDS=12                 ;DRIVE STATUS REGISTER (DRIVE REG 01)
000014                          RMER1=14                ;ERROR REGISTER #1 (DRIVE REG. 02)
000016                          RMAS=16                 ;ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG. 04)
000020                          RMLA=20                 ;LOOK AHEAD REGISTER (DRIVE REG. 07)
000022                          RMDB=22                 ;DATA BUFFER REGISTER (NOT A DRIVE REG.)
000024                          RMMR1=24                ;MAINTAINABILITY REGISTER (DRIVE REG. 03)
```

```
                    000026                  RMDT=26                         ;DRIVE TYPE REGISTER (DRIVE REG. 06)
                    000030                  RMSN=30                         ;SERIAL NUMBER REGISTER (DRIVE REG. 10)
                    000032                  RMOF=32                         ;OFFSET REGISTER (DRIVE REG. 11)
                    000034                  RMDC=34                         ;DESIRED CYLINDER ADDRESS REGISTER (DRIVE REG. 12)
                    000036                  RMHR=36                         ;DUMMY  ADDRESS REGISTER (DRIVE REG. 13)
                    000040                  RMMR2=40                        ;MAINTENANCE REGISTER #2
                    000042                  RMER2=42                        ;ERROR REGISTER #2 (DRIVE REG. 15)
                    000044                  RMEC1=44                        ;ECC POSITION REGISTER (DRIVE REG. 16)
                    000046                  RMEC2=46                        ;ECC PATTERN REGISTER (DRIVE REG   17)

                                            ;RH70/RM03 DRIVER INITIALIZATION CODE
                                                    ;THIS ROUTINE WILL DETERMINE WHICH RM03 DRIVES ARE
                                                    ;AVAILABLE FOR TESTING AND SET THE DRVSTA INDICATOR
                                                    ;TO THE PROPER STATE FOR EACH DRIVE.
                                                    ;NOTE: THIS ROUTINE CALLS DRVINT
                                                    ;
                                            ;CALL
                                                    ;
                                                    ;       JSR     PC,RMINIT
                                                    ;       RETURN
                                                    ;
                                            ;NOTE: THE 'P' OR 'L' CLOCK MUST BE STARTED
                                                    ;
026662  104412          RMINIT: SAVREG                  ;SAVE R0 - R5
026664  013746  177776          MOV     @#PS,-(SP)      ;SAVE THE PRESENT PROCESSOR STATUS
026670  012737  000240  177776  MOV     #(5*32.),@#PS   ;CHANGE THE PRIORITY TO 5
026676  004737  034524          JSR     PC,CLRQUE       ;CLEAR ALL REQUEST QUEUES
026702  012701  026476          MOV     #RMERRS,R1      ;FIRST ADDRESS TO BE CLEARED
026706  012702  026606          MOV     #SEEKFG,R2      ;LAST ADDRESS TO BE CLEARED
026712  005021          1$:     CLR     (R1)+           ;CLEAR
026714  020102                  CMP     R1,R2           ;ARE WE DONE?
026716  103775                  BLO     1$              ;BRANCH IF NO
026720  012702  026630          MOV     #DTUW,R2        ;LAST ADDRESS
026724  012721  177777  2$:     MOV     #-1,(R1)+       ;INITIALIZE
026730  020102                  CMP     R1,R2           ;DONE?
026732  101774                  BLOS    2$              ;LOOP IF NO
026734  005037  026516          CLR     DRVSTA          ;SET ALL DRIVES TO OFFLINE
026740  005037  026520          CLR     DRVSTA+2
026744  005037  026522          CLR     DRVSTA+4
026750  005037  026524          CLR     DRVSTA+6
026754  013703  026646          MOV     RMVEC,R3        ;SETUP THE RH70/RM03 VECTOR
026760  012723  031474          MOV     #ISR,(R3)+
026764  013713  026650          MOV     RMVEC+2,(R3)
026770  013704  026644          MOV     RMADR,R4        ;FIRST ADDRESS OF RH70/RM03
026774  012764  000040  000010  MOV     #BIT05,RMCS2(R4)  ;MASSBUS INIT
027002  005001                  CLR     R1              ;START WITH DRIVE 0
027004  004037  027074  3$:     JSR     R0,DRVINT       ;INIT THE DRIVE
027010  000401                  BR      4$              ;'DVA' NOT SET OR PARITY ERROR
027012  000402                  BR      5$              ;NORMAL RETURN
027014  105061  026516  4$:     CLRB    DRVSTA(R1)      ;SET DRIVE STATUS TO OFFLINE
027020  005201          5$:     INC     R1              ;GO TO NEXT DRIVE
027022  042701  177770          BIC     #^C7,R1         ;MASK OUT UNUSED BITS
027026  001366                  BNE     3$              ;BR IF MORE DRIVES TO GO
027030  012701  000007          MOV     #7,R1           ;START WITH DRIVE 7
027034  005037  177776          CLR     @#PS            ;CLEAR THE PROCESSOR STATUS
027040  105761  026536  6$:     TSTB    DPINT(R1)       ;WAITING FOR DRIVE TO SWITCH PORTS ?
```

```
                                        BEQ     8$                      ;BR NOT WAITING
                                        JSR     PC,SET.IE               ;SET INTERRUPT
                                7$:     TSTB    DPINT(R1)               ;DRIVE SWITCHED PORTS ?
                                        BNE     7$                      ;BR IF NOT
                                8$:     DEC     R1                      ;GO TO THE NEXT DRIVE
                                        BPL     6$                      ;CHECK NEXT DRIVE
                                        MOV     (SP)+,@#PS              ;RESTORE THE PROCESSOR STATUS
                                        RESREG                          ;RESTORE R0 - R5
                                        RTS     PC                      ;BYE-BYE

                                ;DRIVE INITILIZATION ROUTINE
                                        ;THIS ROUTINE DETERMINES IF A DRIVE EXIST AND IF IT IS
                                        ;AN RM03. IF IT IS, A "READ-IN PRESET" IS ISSUED AND FMT22
                                        ;IS SET TO A "1". THEN MOL, DPR, DRY, AND VV ARE CHECKED TO
                                        ;INSURE THEY ARE ALL ON A "1". AND DEPENDING ON THEIR STATE,
                                        ;DRVSTA IS SET TO THE PROPER CONDITION.
                                ;CALL
                                ;       MOV     #DRVNUM,R1              ;DRIVE NUMBER TO R1
                                ;       MOV     RMADR,R4                ;UNIBUS ADDRESS OF RH70/RM03 (RMCS1)
                                ;       JSR     R0,DRVINT               ;CALLED BY A JSR
                                ;       RETURN1                         ;ERROR OCCURRED (PARITY)
                                ;       RETURN2                         ;NORMAL RETURN
                                ;

                                DRVINT: MOV     R5,-(SP)                ;SAVE R5
                                DULP:   CLRB    DRVSTA(R1)              ;START DRIVE STATUS AS OFFLINE
                                        CLRB    DRVTYP(R1)              ;CLEAR THE DRIVE TYPE INDICATOR
                                        CLRB    ULDFLG(R1)              ;CLEAR THE UNLOAD FLAG
                                        MOV     R1,RMCS2(R4)            ;SELECT A DRIVE
                                        MOVB    #111,RMCS1(R4)          ;DO A DRIVE CLEAR COMMAND (& SEIZE DRIVE)
                                        BIT     #BIT12,RMCS2(R4)        ;NONEXISTENT DRIVE?
                                        BEQ     1$                      ;NO---BRANCH
                                        JSR     PC,SET.IE               ;GO SET "IE" WITHOUT A "TRE"
                                        BR      6$                      ;LEAVE THIS ROUTINE
                                1$:     CLRB    DRVSTA(R1)              ;SET DRIVE STATUS TO OFFLINE
                                        BIT     #BIT11,RMCS1(R4)        ;SEE IF DRIVE AVAILABLE
                                        BEQ     DULP                    ;BR IF DRIVE NOT AVAILABLE
                                        JSR     R0,RD.RM                ;READ THE DRIVE TYPE REG.
                                        RMDT
                                        8$                              ;ERROR RETURN ADDRESS
                                        MOV     (SP)+,R5                ;PUT DRIVE TYPE IN R5
                                        MOVB    #4,DRVTYP(R1)           ;SET RM03 INDICATOR
                                        CMP     #20024,R5               ;SINGLE PORT RM03 ?
                                        BEQ     2$                      ;BR IF YES
                                        CMP     #24024,R5               ;DUAL PORT RM03 ?
                                        BEQ     2$                      ;BR IF YES
                                        MOVB    #5,DRVTYP(R1)           ;SET RM02 INDICATOR
                                        CMP     #20025,R5               ;SINGLE PPRT RM02 ?
                                        BEQ     2$                      ;BRANCH IF SO
                                        CMP     #24025,R5               ;DUAL PORT RM02 ?
                                        BEQ     2$                      ;BRANCH IF SO
                                        MOVB    #-1,DRVTYP(R1)          ;SET INDICATOR TO 'OTHER'
                                        BR      6$                      ;EXIT
                                2$:     MOV     #121,-(SP)              ;DO A "READ-IN PRESET"
                                        JSR     R0,WR.RM
                                        RMCS1
```

```
5176  027256  027404                          8$
5177  027260  012746  010000                  MOV     #BIT12,-(SP)    ;SET FMT22=1
5178  027264  004037  033650                  JSR     R0,WRT.RM
5179  027270  000032                          RMOF
5180  027272  027404                          8$
5181  027274  004037  033470                  JSR     R0,RD.RM        ;READ RMDS
5182  027300  000012                          RMDS
5183  027302  027404                          8$
5184  027304  012605                          MOV     (SP)+,R5        ;AND SAVE IT IN R5
5185  027306  100015                          BPL     4$              ;BRANCH IF ATA=0
5186  027310  116164  026632  000016          MOVB    ATABIT(R1),RMAS(R4) ;CLEAR ATTENTION BIT
5187  027316  004037  033470                  JSR     R0,RD.RM        ;FIND OUT WHY ATA=1
5188  027322  000014                          RMER1
5189  027324  027404                          8$
5190  027326  006126                          ROL     (SP)+           ;IS IT UNSAFE?
5191  027330  100004                          BPL     4$              ;BR IF NOT
5192  027332  112761  177777  026516          MOVB    #-1,DRVSTA(R1)  ;SET UNSAFE INDICATOR
5193  027340  000407                          BR      6$              ;EXIT
5194  027342  005105                  4$:     COM     R5              ;CHECK MOL, DPR, DRY, AND VV
5195  027344  042705  167077                  BIC     #^C<BIT12!BIT08!BIT07!BIT06>,R5
5196  027350  001003                          BNE     6$              ;BRANCH IF MOL, DPR, DRY, OR VV IS CLEAR
5197  027352  112761  000001  026516          MOVB    #1,DRVSTA(R1)   ;SET DRIVE STATUS TO ONLINE
5198  027360  005720                  6$:     TST     (R0)+           ;STEP OVER THE ERROR RETURN
5199  027362  000410                          BR      8$              ;EXIT
5200  027364  006301                  7$:     ASL     R1              ;CHANGE INDEX TO ADDRESS WORDS
5201  027366  012761  060000  026610          MOV     #60000,TIMER(R1)  ;START 2 SEC TIMER
5202  027374  006201                          ASR     R1              ;RESTORE R1
5203  027376  112761  177777  026536          MOVB    #-1,DFINT(R1)   ;SET PORT INITIALIZE INIDICATOR
5204  027404  012605                  8$:     MOV     (SP)+,R5        ;RESTOPE R5
5205  027406  000200                          RTS     R0              ;EXIT
5206
5207                                  ;REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST
5208                                  ;
5209                                  ;CALL
5210                                  ;
5211                                  ;       JSR     R0,@#RM03       ;CALL THE RM03 DRIVER
5212                                  ;       PNTADR                  ;ADDRESS OF POINTER OF DRIVES PARAMETER BLOCK
5213                                  ;       RETURN1                 ;RETURN HERE IF QUEUE IS FULL
5214                                  ;       RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE OR THERE
5215                                  ;                               ;IS AN ERROR CONDITION
5216
5217  027410  013746  177776  RM03:   MOV     @#PS,-(SP)      ;SAVE THE CALLING STATUS
5218  027414  013737  026650  177776  MOV     RMVEC+2,@#PS    ;DON'T ALLOW ANY RM03 INTERRUPTS
5219  027422  112737  000001  026562  MOVB    #1,ACTDRV       ;SET "ACTIVE DRIVER" FLAG
5220  027430  104412                  SAVREG                  ;SAVE R0 - R5
5221  027432  011002                  MOV     (R0),R2         ;PICKUP THE DRIVE PARAMETER BLOCK POINTER
5222  027434  005062  000016          CLR     16(R2)          ;CLEAR THE STATUS/ERROR INDICATOR
5223  027440  111201                  MOVB    (R2),R1         ;PICKUP THE DRIVE NUMBER
5224  027442  013704  026644          MOV     RMADR,R4        ;UNIBUS ADDRESS OF RMCS1
5225  027446  105761  026516          TSTB    DRVSTA(R1)      ;CHECK DRIVES STATUS
5226  027452  003014                  BGT     1$              ;BRANCH IF ONLINE
5227  027454  105761  026564          TSTB    ULDFLG(R1)      ;UNLOAD COMMAND IN QUEUE?
5228  027460  001036                  BNE     3$              ;BRANCH IF YES
5229  027462  105761  026536          TSTB    DPINT(R1)       ;TRYING TO INIT THE DRIVE
5230  027466  001042                  BNE     5$              ;BR IF YES
5231  027470  004037  027074          JSR     R0,DRVINT       ;GO INIT  THE DRIVE
```

```
5232  027474  000434                        BR      4$              ;ERROR RETURN
5233  027476  105761  026516                TSTB    DRVSTA(R1)      ;IS DRIVE STATUS ONLINE?
5234  027502  003445                        BLE     6$              ;BR IF NOT
5235  027504  105761  026546          1$:   TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
5236  027510  001031                        BNE     5$              ;BR IF YES
5237  027512  010164  000010                MOV     R1,RMCS2(R4)    ;SELECT THE DRIVE
5238  027516  004037  034622                JSR     R0,DRVQUE       ;PUT THIS REQUEST IN QUEUE
5239  027522  000460                        BR      9$              ;QUEUE IS FULL
5240  027524  122762  000103  000002        CMPB    #103,2(R2)      ;IS THIS REQ. FOR AN UNLOAD?
5241  027532  001003                        BNE     2$              ;BR IF NO
5242  027534  112761  177777  026564        MOVB    #-1,ULDFLG(R1)  ;SET THE "UNLOAD IN QUEUE" FLAG
5243  027542  105761  026506          2$:   TSTB    DRVACT(R1)      ;IS THIS DRIVE ACTIVE?
5244  027546  001043                        BNE     8$              ;BR IF YES
5245  027550  004737  027702                JSR     PC,OPT          ;CALL THE OPTIMIZER
5246  027554  000440                        BR      8$
5247  027556  012762  120000  000016  3$:   MOV     #BIT15!BIT13,16(R2) ;SET THE "UNLOAD IN QUEUE" ERROR FLAG
5248  027564  000434                        BR      8$              ;EXIT
5249  027566  004737  030762          4$:   JSR     PC,CI?          ;GO HANDLE THE PARITY ERROR
5250  027572  000431                        BR      8$
5251  027574  004037  034622          5$:   JSR     R0,DRVQUE       ;PUT REQUEST IN QUEUE
5252  027600  000431                        BR      9$              ;QUEUE IS FULL
5253  027602  032714  000100                BIT     #BIT06,(R4)     ;IE BIT SET ?
5254  027606  001023                        BNE     8$              ;YES
5255  027610  004737  034160                JSR     PC,SET.IE       ;SET THE INTERRUPT
5256  027614  000420                        BR      8$              ;RETURN
5257  027616  105761  026516          6$:   TSTB    DRVSTA(R1)      ;SEE IF DRIVE OFFLINE OR UNSAFE
5258  027622  002412                        BLT     7$              ;BR IF UNSAFE
5259  027624  012762  140000  000016        MOV     #BIT15!BIT14,16(R2)  ;SET OFFLINE ERROR INDICATOR
5260  027632  105761  026526                TSTB    DRVTYP(R1)      ;SEE IF OFFLINE OR NONEXISTENT
5261  027636  001007                        BNE     8$              ;BR IF OFFLINE
5262  027640  012762  100002  000016        MOV     #BIT15!BIT01,16(R2)  ;REPORT DRIVE NONEXISTENT
5263  027646  000403                        BR      8$              ;GO TO EXIT
5264  027650  012762  110000  000016  7$:   MOV     #BIT15!BIT12,16(R2)  ;DRIVE IS UNSAFE
5265  027656  104413                  8$:   RESREG                  ;RESTORE R0 - R5
5266  027660  005720                        TST     (R0)+           ;SETUP FOR NORMAL RETURN
5267  027662  000401                        BR      10$             ;FINISH UP, THEN EXIT
5268  027664  104413                  9$:   RESREG                  ;RESTORE R0 - R5
5269  027666  005720                  10$:  TST     (R0)+           ;CORRECT THE RETURN ADDRESS
5270  027670  105037  026562                CLRB    ACTDRV          ;CLEAR "ACTIVE DRIVER" FLAG
5271  027674  012637  177776                MOV     (SP)+,@#PS      ;RETURN "PS" TO USER LEVEL
5272  027700  000200                        RTS     R0              ;RETURN TO CALLER
5273
5274                                  ;OPTIMIZER-CALLED FOR A PARTICULAR DRIVE
5275                                  ;
5276                                  ;CALL
5277                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
5278                                  ;       JSR     PC,OPT          ;SETUP A COMMAND
5279                                  ;
5280  027702  104412                  OPT:   SAVREG                  ;SAVE R0 - R5
5281  027704  013746  177776                MOV     @#PS,-(SP)      ;SAVE PROC. STATUS
5282  027710  146137  026632  026560        BICB    ATABIT(R1),SRCHW        ;CLEAR LA SEACH FLAG
5283  027716  105061  026546                CLRB    DPRQS(R1)       ;RESET THE PORT REQ FLAG ****
5284  027722  004737  034676                JSR     PC,GETREQ       ;GET "DPB" POINTER OF REQUEST
5285  027726  005702                        TST     R2              ;IS THERE A REQUEST IN QUEUE?
5286  027730  001472                        BEQ     7$              ;NO- BRANCH TO EXIT
5287  027732  010164  000010                MOV     R1,RMCS2(R4)    ;LOAD THE DRIVE ADDRESS *******
```

```
5288  027736  012764  000111  000000          MOV     #111,RMCS1(R4)  ;CLEAR THE DRIVE
5289  027744  032764  004000  000000          BIT     #BIT11,RMCS1(R4)        ;DVA SET ?
5290  027752  001446                          BEQ     5$              ;IF NOT, REQUEST
5291  027754  105761  026516          10$:    TSTB    DRVSTA(R1)      ;IS DRIVE ONLINE?
5292  027760  003014                          BGT     1$              ;YES--BRANCH
5293  027762  004737  034720                  JSR     PC,POPQUE       ;NO--REMOVE REQUEST FROM QUEUE
5294  027766  012762  140000  000016          MOV     #BIT15!BIT14,16(R2)  ;SET OFFLINE STATUS/ERROR INDICATOR
5295  027774  105761  026516                  TSTB    DRVSTA(R1)      ;IS DRIVE UNSAFE ?
5296  030000  100053                          BPL     8$              ;BR TO EXIT IF NOT
5297  030002  012762  110000  000016          MOV     #BIT15!BIT12,16(R2)  ;SET UNSAFE STATUS/ERROR INDICATOR
5298  030010  000447                          BR      8$              ;BRANCH TO EXIT
5299  030012                          .$:
5300                                  :       MOV     #111,-(SP)      ;LOAD COMMAND ONTO THE STACK
5301                                  :       JSR     R0,WRT.RM       ;LOAD THE REGISTER
5302                                  :       RMCS1                   ;REGISTER INCREMENT
5303                                  :       6$                      ;ERROR RETURN ADDRESS
5304                                  :       BIT     #BIT11,(R4)     ;DRIVE AVAILABLE ?
5305                                  :       BEQ     9$              ;BR IF NOT
5306  030012  122762  000150  000002          CMPB    #150,2(R2)      ;IS THE REQUEST FOR I/O?
5307  030020  002403                          BLT     2$              ;YES--BRANCH
5308  030022  004737  030346                  JSR     PC,CI4          ;CALL THE COMMAND INITIATOR
5309  030026  000440                          BR      8$              ;BRANCH TO EXIT
5310  030030  005737  026630          2$:     TST     DTUW            ;DATA TRANSFER UNDERWAY?
5311  030034  002012                          BGE     4$              ;YES--GO START A SEARCH
5312  030036  005737  026606                  TST     SEEKFG          ;DO IMPLIED SEEKS?
5313  030042  100404                          BMI     3$              ;YES---BRANCH
5314  030044  004037  031320                  JSR     R0,LA           ;NO--DO LOOK AHEAD
5315  030050  000427                          BR      8$              ;RETURN HERE ON A PARITY ERROR
5316  030052  000403                          BR      4$              ;GO START A SEARCH
5317  030054  004737  030140          3$:     JSR     PC,CI1          ;START A DATA TRANSFER
5318  030060  000423                          BR      8$
5319  030062  004737  030246          4$:     JSR     PC,CI3          ;START A SEARCH
5320  030066  000420                          BR      8$              ;GO TO THE EXIT
5321  030070  112761  177777  026546  5$:     MOVB    #-1,DPRQS(R1)   ;SET PORT REQUEST INDICATOR
5322  030076  010103                          MOV     R1,R3           ;SET UP TO ADDRESS WORDS
5323  030100  006303                          ASL     R3              ;CONVERT TO WORD INDEX
5324  030102  012763  060000  026610          MOV     #60000.,TIMER(R3)  ;START 10 SEC TIMER
5325  030110  000402                          BR      7$              ;EXIT
5326  030112  004737  030762          6$:     JSR     PC,CI7          ;PROCESS THE PARITY ERROR
5327  030116  032714  000100          7$:     BIT     #BIT06,(R4)     ;SEE IF 'IE' ALREADY SET
5328  030122  001002                          BNE     8$              ;BR IF SET
5329  030124  004737  034160                  JSR     PC,SET.IE       ;SET "IE" WITHOUT A "TRE"
5330  030130  012637  177776          8$:     MOV     (SP)+,@#PS      ;RESTORE PROC. STATUS
5331  030134  104413                          RESREG                  ;RESTORE R0 - R5
5332  030136  000207                          RTS     PC
5333
5334                                  ;COMMAND INITIATOR
5335                                  ;
5336                                  ;CALL
5337                                  :       MOV     #DRVNUM,R1      ;DRIVE NUMBER
5338                                  :       MOV     #DPB,R2         ;ADDRESS OF DPB
5339                                  :       JSR     PC,CI?          ;CI?= CI1,CI3, OR CI4
5340                                  :                               ;WHERE:
5341                                  :                               ;CI1=DATA TRANSFER
5342                                  :                               ;CI2=SEARCH REQUESTED BY DATA XFER
5343                                  :                               ;CI4=NOT DATA TRANSFER
```

```
5344
5345  030140  004737  034720          CI1:    JSR     PC,POPQUE       ;REMOVE REQUEST FROM "DRIVES WAIT" QUEUE
5346  030144  010237  026556                  MOV     R2,TRNSWT       ;PUT REQ. IN TRANSFER WAIT QUEUE
5347  030150  010203                          MOV     R2,R3           ;DPB ADDRESS TO R3
5348  030152  013704  026644                  MOV     RMADR,R4        ;RMCS1 ADDRESS
5349  030156  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT DRIVE
5350  030162  062703  000004                  ADD     #4,R3           ;DESIRED WORD COUNT
5351  030166  062704  000002                  ADD     #2,R4           ;RMWC ADDRESS
5352  030172  012324                          MOV     (R3)+,(R4)+     ;LOAD WORD COUNT
5353  030174  012324                          MOV     (R3)+,(R4)+     ;LOAD BUFFER ADDRESS
5354  030176  012346                          MOV     (R3)+,-(SP)     ;LOAD SECTOR AND TRACK
5355  030200  004037  033650                  JSR     R0,WRT.RM       ;CALL THE LOAD(WRITE) ROUTINE
5356  030204  000006                          RMDA                    ;INDEX OF REGISTER TO LOAD
5357  030206  030762                          CI7                     ;ERROR RETURN ADDRESS
5358  030210  012346                          MOV     (R3)+,-(SP)     ;LOAD CYLINDER ADDRESS
5359  030212  004037  033650                  JSR     R0,WRT.RM
5360  030216  000034                          RMDC
5361  030220  030762                          CI7
5362  030222  016246  000002                  MOV     2(R2),-(SP)     ;LOAD "COMMAND+GO", "A17&A16", AND "PSEL"
5363  030226  004037  033650                  JSR     R0,WRT.RM
5364  030232  000000                          RMCS1
5365  030234  030762                          CI7
5366  030236  010137  026630                  MOV     R1,DTUW         ;SET "DATA TRANSFER UNDERWAY"
5367  030242  000137  030724                  JMP     CI5
5368  030246  013704  026644          CI3:    MOV     RMADR,R4        ;RMCS1 ADDRESS
5369  030252  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT DRIVE
5370  030256  016246  000012                  MOV     12(R2),-(SP)    ;DESIRED CYLINDER ADDRESS
5371  030262  004037  033650                  JSR     R0,WRT.RM
5372  030266  000034                          RMDC
5373  030270  030762                          CI7
5374  030272  116203  000010                  MOVB    10(R2),R3       ;PICKUP SECTOR ADDRESS
5375                                  ;       SUB     MXWNDW,R3       ;BACKUP BY MAX. SEARCH FOR I/O WINDOW
5376                                  ;       BGE     1$
5377                                  ;       ADD     #32.,R3
5378  030276  010346                  1$:     MOV     R3,-(SP)        ;COMBINE THE ADJUSTED SECTOR WITH
5379  030300  042716  177740                  BIC     #177740,(SP)    ;CHOP OF ALL EXENTED BITS ,IF ANY
5380  030304  116266  000011  000001          MOVB    11(R2),1(SP)    ;THE DESIRED TRACK
5381  030312  004037  033650                  JSR     R0,WRT.RM       ;LOAD DESIRED TRACK & SECTOR
5382  030316  000006                          RMDA
5383  030320  030762                          CI7
5384  030322  012746  000131                  MOV     #131,-(SP)      ;START A SEARCH
5385  030326  004037  033650                  JSR     R0,WRT.RM
5386  030332  000000                          RMCS1
5387  030334  030762                          CI7
5388  030336  156137  026632  026560          BISB    ATABIT(R1),SRCHWT       ;SET "SEARCH WAIT" KEY
5389  030344  000567                          BR      CI5
5390  030346  013704  026644          CI4:    MOV     RMADR,R4        ;RMCS1 ADDRESS
5391  030352  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT DRIVE
5392  030356  116203  000002                  MOVB    2(R2),R3        ;PICKUP THE REQUESTED COMMAND
5393  030362  122703  000131                  CMPB    #131,R3         ;IS IT A SEARCH COMMAND?
5394  030366  001007                          BNE     1$              ;BRANCH IF NO
5395  030370  016246  000010                  MOV     10(R2),-(SP)    ;LOAD DESIRED TRACK & SECTOR
5396  030374  004037  033650                  JSR     R0,WRT.RM
5397  030400  000006                          RMDA
5398  030402  030762                          CI7
5399  030404  000403                          BR      2$              ;GO LOAD CYLINDER
```

```
5400  030406  122703  000105          1$:   CMPB   #105,R3          ;IS IT A SEEK COMMAND
5401  030412  001007                        BNE    3$               ;BRANCH IF NO
5402  030414  016246  000012          2$:   MOV    12(R2),-(SP)     ;LOAD DESIRED CYLINDER
5403  030420  004037  033650                JSR    R0,WRT.RM
5404  030424  000034                        RMDC
5405  030426  030762                        CI7
5406  030430  000546                        BR     CI6
5407  030432  122703  000115          3$:   CMPB   #115,R3          ;IS IT AN "OFFSET" COMMAND?
5408  030436  001013                        BNE    4$               ;BR IF NO
5409  030440  004037  033470                JSR    R0,RD.RM         ;MERGE THE OFFSET VALUE INTO RMOF
5410  030444  000032                        RMOF                    ;BUT DON'T CHANGE THE UPPER
5411  030446  030762                        CI7
5412  030450  116216  000001                MOVB   1(R2),(SP)       ;BYTE WHEN LOADING THE
5413  030454  004037  033650                JSR    R0,WRT.RM        ;REGISTER (RMOF)
5414  030460  000032                        RMOF
5415  030462  030762                        CI7
5416  030464  000530                        BR     CI6              ;GO START THE COMMAND
5417  030466  122703  000107          4$:   CMPB   #107,R3          ;IS IT A "RECALIBRATE" COMMAND?
5418  030472  001525                        BEQ    CI6              ;BRANCH IF YES
5419  030474  122703  000117                CMPB   #117,R3          ;IS IT A RETURN TO CENTER?
5420  030500  001522                        BEQ    CI6              ;BRANCH IF YES
5421  030502  122703  000103                CMPB   #103,R3          ;IS IT AN "UNLOAD" COMMAND?
5422  030506  001016                        BNE    5$               ;BRANCH IF NO
5423  030510  112761  000001  026506        MOVB   #1,DRVACT(R1)    ;SET THE DRIVE ACTIVE INDICATOR
5424  030516  105061  026516                CLRB   DRVSTA(R1)       ;PUT DRIVE STATUS TO OFFLINE
5425  030522  112761  000001  026564        MOVB   #1,ULDFLG(R1)    ;SET "UNLOAD IN PROGPESS" FLAG
5426  030530  010346                        MOV    R3,-(SP)         ;START THE "UNLOAD" COMMAND
5427  030532  004037  033650                JSR    R0,WRT.RM
5428  030536  000000                        RMCS1
5429  030540  030762                        CI7
5430  030542  000207                        RTS    PC               ;RETURN TO USER
5431  030544  122703  000143          5$:   CMPB   #143,R3          ;IS IT A "SET FORMAT" COMMAND?
5432  030550  001014                        BNE    6$               ;BRANCH IF NO
5433  030552  004037  033470                JSR    R0,RD.RM         ;READ THE OFFSET REGISTER
5434  030556  000032                        RMOF
5435  030560  030762                        CI7
5436  030562  116266  000001  000001        MOVB   1(R2),1(SP)      ;COMBINE "FMT22","ECI", AND "HCI"
5437  030570  004037  033650                JSR    R0,WRT.RM        ;LOAD "FMT22", "ECI", AND/OR "HCI".
5438  030574  000032                        RMOF
5439  030576  030762                        CI7
5440  030600  000436                        BR     12$
5441  030602  122703  000141          6$:   CMPB   #141,R3          ;IS IT A "GET REGISTER" COMMAND?
5442  030606  001023                        BNE    10$              ;BRANCH IF NO
5443  030610  016203  000006          7$:   MOV    6(R2),R3         ;POINTS TO 1ST ADDRESS OF WHERE
5444                                                                ;TO PUT THE REGISTER(S)
5445  030614  116237  000010  030632        MOVB   10(R2),9$        ;INIT. THE INDEX FOR THE FIRST REG.
5446  030622  116205  000011                MOVB   11(R2),R5        ;INDEX OF LAST REG. TO MOVE
5447  030626  004037  033470          8$:   JSR    R0,RD.RM         ;READ RH70/RM03 REGISTER
5448  030632  000000          9$:   RMCS1                           ;INDEX OF REG. TO READ
5449  030634  030762                        CI7
5450  030636  012623                        MOV    (SP)+,(R3)+      ;GET THE CONTENTS OF RH70//RM03 REG.
5451  030640  023705  030632                CMP    9$,R5            ;LAST REG. BEEN READ?
5452  030644  001414                        BEQ    12$              ;GET OUT IF YES
5453  030646  062737  000002  030632        ADD    #2,9$            ;INCREASE THE INDEX BY 2
5454  030654  000764                        BR     8$               ;LOOP--MORE TO READ
5455  030656  122703  000145          10$:  CMPB   #145,R3          ;IS IT A "SELECT DRIVE" COMMAND?
```

```
5456  030662  001405                        BEQ     12$                ;BRANCH IF YES
5457  030664  010346                 11$:   MOV     R3,-(SP)           ;LOAD THE COMMAND
5458  030666  004037  033650                JSR     R0,WRT.RM
5459  030672  000000                        RMCS1
5460  030674  030762                        CI7
5461  030676  004737  034720          12$:  JSR     PC,POPQUE          ;REMOVE REQ. FROM QUEUE
5462  030702  052762  000200  000016        BIS     #BIT07,16(R2)      ;SET THE "DONE" BIT
5463  030710  005737  026604                TST     SAVEFG             ;SAVE THE RH70/RM03 REGISTERS?
5464  030714  100002                        BPL     13$                ;BRANCH IF NO
5465  030716  004737  034042                JSR     PC,SVRH70          ;YES--GO SAVE THE REGISTERS
5466  030722  000207                 13$:   RTS     PC                 ;RETURN TO USER
5467  030724  006301                 CI5:   ASL     R1
5468  030726  012761  060000  026610        MOV     #60000,TIMER(R1)           ;SET A ONE SECOND TIMER
5469  030734  006201                        ASR     R1
5470  030736  112761  000001  026506        MOVB    #1,DRVACT(R1)      ;SET THE DRIVE ACTIVE
5471  030744  000207                        RTS     PC                 ;RETURN TO THE USER
5472  030746  010346                 CI6:   MOV     R3,-(SP)           ;LOAD THE COMMAND
5473  030750  004037  033650                JSR     R0,WRT.RM
5474  030754  000000                        RMCS1
5475  030756  030762                        CI7
5476  030760  000761                        BR      CI5
5477  030762  032764  010000  000010  CI7:  BIT     #BIT12,RMCS2(R4)   ;DRIVE NON-EXISTENT ?
5478                                 ;
5479  030770  005702                 1$:    TST     R2                 ;ANYTHING IN QUEUE ?
5480                                 ;      BEQ     CI7B               ;BR IF NOT
5481  030772  001001                        BNE     2$                 ;BRANCH IF QUEUE IS THERE
5482  030774  000207                        RTS     PC                 ;OTHERWISE EXIT
5483  030776  012762  104000  000016  2$:   MOV     #BIT15!BIT11,16(R2)        ;SET "PARITY" ERROR INDICATOR
5484                                 ;      JSR     PC,SVRH70          ;GO SAVE THE RH70/RM03 REGISTERS
5485  031004  012746  000111          CI7B: MOV     #111,-(SP)         ;DO A "DRIVE CLEAR"
5486  031010  004037  033650                JSR     R0,WRT.RM
5487  031014  000000                        RMCS1
5488  031016  031062                        CI8
5489  031020  004737  034602          2$:   JSR     PC,EMPTYQ          ;EMPTY THE QUEUE
5490  031024  105061  026546                CLRB    DPRQS(R1)          ;CLEAR THE PORT REQUEST FLAG
5491  031030  105061  026564                CLRB    ULDFLG(R1)         ;CLEAR THE UNLOAD 1.! QUEUE FLAG
5492  031034  105061  026506                CLRB    DRVACT(R1)         ;DRIVE IS IDLE
5493  031040  020237  026556                CMP     R2,TRNSWT          ;IF THIS DRIVE HAD AN I/O REQUEST
5494                                 ;      CMP     R1,DTUW            ;IF THIS DRIVE HAD AN I/O REQUEST
5495  031044  001005                        BNE     1$                 ;IN PROGRESS CLEAR ALL OF THE FLAGS
5496  031046  005037  026556                CLR     TRNSWT
5497  031052  012737  177777  026630        MOV     #-1,DTUW
5498  031060  000207                 1$:    RTS     PC
5499  031062  104412                 CI8:   SAVREG                     ;SAVE R0 - R5
5500  031064  032764  010000  000010        BIT     #BIT12,RMCS2(R4)   ;IS 'NED' SET ?
5501                                 ;      BNE     1$                 ;BR IF YES
5502  031072  005001                        CLR     R1
5503  031074  005003                        CLR     R3
5504  031076  105761  026506          1$:   TSTB    DRVACT(R1)         ;DRIVE ACTIVE?
5505                                 ;      BEQ     5$                 ;BRANCH IF NO
5506  031102  001003                        BNE     22$                ;BRANCH IF IN ACTIVE
5507  031104  105761  026546                TSTB    DPRQS(R1)          ;PORT REQUEST
5508  031110  001443                        BEQ     5$                 ;BRANCH IF NOT
5509  031112  013702  026556          22$:  MOV     TRNSWT,R2          ;GET THE "TRANSFER WAIT" QUEUE
5510  031116  020137  026630                CMP     R1,DTUW            ;DID THIS DRIVE HAVE AN I/O IN PROGRESS?
5511  031122  001402                        BEQ     2$                 ;BRANCH IF YES
```

```
5512  031124  004737  034676            2$:   JSR     PC,GETREQ       ;GET THE DPB POINTER
5513  031130  005702                          TST     R2              ;QUEUE ENTRY FOR DRIVE ?
5514  031132  001413                          BEQ     4$              ;BR IF NOT
5515  031134  032764  010000  000010          BIT     #BIT12,RMCS2(R4)  ;'NED' SET ?
5516  031142  001404                          BEQ     3$              ;BR IF NOT
5517  031144  012762  100002  000016          MOV     #BIT15!BIT01,16(R2)  ;SET 'DRIVE NON-EXISTENT' INDICATOR
5518  031152  000403                          BR      4$              ;CONTINUE
5519  031154  012762  102000  000016    3$:   MOV     #BIT15!BIT10,16(R2) ;SET "NON-CLEARABLE PARITY" ERROR INDICATOR
5520                                          ;
5521  031162  012763  177777  026610    4$:   MOV     #-1,TIMER(R3)   ;STOP THE TIMER
5522  031170  105061  026506                  CLRB    DRVACT(R1)      ;SET "DRIVE ACTIVE" TO IDLE
5523  031174  105061  026546                  CLRB    DPRQS(R1)       ;CLEAR PORT REQUEST FLAG
5524  031200  020137  026630                  CMP     R1,DTUW         ;IS THIS DRIVE SETUP FOR A TRANSFER
5525  031204  001005                          BNE     5$              ;BR IF NOT
5526  031206  012737  177777  026630          MOV     #-1,DTUW        ;RESET THE INDICATOR
5527  031214  005037  026556                  CLR     TRN$WT          ;CLEAR THE TRANSFER QUEUE
5528  031220  105061  026564            5$:   CLRB    ULDFLG(R1)      ;CLEAR UNLOAD FLAG
5529  031224  032764  010000  000010          BIT     #BIT12,RMCS2(R4)  ;'NED' SET ?
5530                                          ;
5531  031232  005201                          BNE     6$              ;BR IF YES
5532  031234  062703  000002                  INC     R1              ;MOVE TO THE NEXT DRIVE
5533  031240  042701  177770                  ADD     #2,R3
5534  031244  001314                          BIC     #^C7,R1
5535  031246  012737  177777  026630          BNE     1$              ;BRANCH IF MORE DRIVES
5536  031254  005037  026556                  MOV     #-1,DTUW        ;NO DATA TRANSFERS UNDERWAY
5537  031260  004737  034524                  CLR     TRN$WT          ;CLEAR THE 'TRANSFER WAIT' QUEUE
5538  031264  012764  000040  000010          JSR     PC,CLRQUE       ;CLEAR ALL OF THE REQUEST QUEUES
5539  031272  000406                          MOV     #BIT05,RMCS2(R4) ;DO A MASSBUS INIT.
5540  031274  004737  034602            6$:   BR      7$              ;CONTINUE
5541  031300  105061  026516                  JSR     PC,EMPTYQ       ;CLEAR THE DRIVE'S QUEUE
5542  031304  105061  026526                  CLRB    DRVSTA(R1)      ;SET DRIVE TO OFFLINE
5543  031310  004737  034160            7$:   CLRB    DRVTYP(R1)      ;CLEAR THE DRIVE TYPE INDICATOR
5544  031314  104413                          JSR     PC,SET.IE       ;SET "IE" WITHOUT "TRE"
5545  031316  000207                          RESREG                  ;RESTORE R0 - R5
5546                                          RTS     PC              ;RETURN
5547                                          ;LOOK AHEAD ROUTINE
5548                                          ;
5549                                          ;CALL
5550                                          ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
5551                                          ;       MOV     #DPB,R2         ;POINT TO DPB
5552                                          ;       JSR     R0,LA           ;GO CHECK THE WINDOW
5553                                          ;       RETURN1                 ;ERROR RETURN
5554                                          ;       RETURN2                 ;START A SEARCH
5555                                          ;       RETURN3                 ;START A DATA TRANSFER
5556
5557  031320  013704  026644            LA:   MOV     RMADR,R4        ;GET RMCS1'S ADDRESS
5558  031324  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT DRIVE
5559  031330  004037  033470                  JSR     R0,RD.RM        ;READ DRIVE STATUS
5560  031334  000012                          RMDS
5561  031336  031466                          4$                      ;ERROR RETURN ADDRESS
5562  031340  042716  157577                  BIC     #^C020200,(SP)  ;ON CYLINDER ?
5563  031344  022726  000200                  CMP     #200,(SP)+      ;PIP=0,DRY=1?
5564  031350  001044                          BNE     3$              ;NO
5565  031352  105261  026574                  INCB    LACNT(R1)       ;INCREMENT THE LOOK AHEAD COUNT
5566  031356  126137  026574  026652          CMPB    LACNT(R1),MXLACT ;EXCEED MAX?
5567  031364  003033                          BGT     2$              ;BRANCH IF YES
```

```
5568  031366  116203  000010                      MOVB    10(R2),R3       ;GET DESIRED SECTOR ADDRESS AND
5569  031372  000303                              SWAB    R3              ;MULT. BY 64--ALIGN WITH
5570  031374  006203                              ASR     R3              ;LOOK AHEAD REGISTER
5571  031376  006203                              ASR     R3
5572  031400  012737  000340  177776              MOV     #340,@#PS       ;PRIORITY LEVEL "7"
5573  031406  004037  033470              6$:     JSR     R0,RD.RM        ;READ LOOK AHEAD REGISTER
5574  031412  000020                              RMLA
5575  031414  031466                              4$
5576  031416  021664  000020                      CMP     (SP),RMLA(R4)   ;CORRECT LA NUMBER ?
5577  031422  001402                              BEQ     7$              ;YES
5578  031424  005726                              TST     (SP)+           ;NO,CLEAR STACK
5579  031426  000415                              BR      3$
5580  031430  162603                      7$:     SUB     (SP)+,R3        ;CALCULATE THE DELTA
5581  031432  002002                              BGE     1$
5582  031434  062703  004000                      ADD     #<32.*64.>,R3   ;MAKE THE DELTA POSITIVE
5583  031440  023703  026654              1$:     CMP     MXDLTA,R3       ;CHECK THE DELTA TO SEE
5584  031444  002406                              BLT     3$              ;IF IT IS WITHIN THE
5585  031446  023703  026656                      CMP     MNDLTA,R3       ;WINDOW---IF YES, ZERO
5586  031452  002003                              BGE     3$              ;THE LOOK AHEAD COUNT
5587  031454  105061  026574              2$:     CLRB    LACNT(R1)       ;AND TAKE THE I/O EXIT
5588  031460  005720                              TST     (R0)+
5589  031462  005720                      3$:     TST     (R0)+           ;ADJUST THE RETURN ADDRESS
5590  031464  000402                              BR      5$              ;EXIT
5591  031466  004737  030762              4$:     JSR     PC,CI7          ;PROCESS THE ERROR
5592  031472  000200                      5$:     RTS     R0              ;RETURN
5593
5594                                      ;INTERRUPT SERVICE ROUTINE
5595
5596  031474  112737  000001  026562      ISR:    MOVB    #1,ACTDRV       ;SET "ACTIVE DRIVER" FLAG
5597  031502  104412                              SAVREG                  ;SAVE R0 - R5
5598  031504  013704  026644                      MOV     RMADR,R4        ;ADDRESS OF RHSCS1
5599  031510  013701  026630                      MOV     DTUW,R1         ;GET "DATA TRANSFER UNDERWAY" INDICATOR
5600  031514  002403                              BLT     1$              ;BRANCH IF NO DATA TRANSFER UNDERWAY
5601  031516  004737  031540                      JSR     PC,TD           ;CALL TRANSFER DONE
5602  031522  000402                              BR      2$              ;EXIT
5603  031524  004737  031710              1$:     JSR     PC,SC           ;CALL SPECIAL CONDITIONS
5604  031530  104413                      2$:     RESREG                  ;RESTORE R0 - R5
5605  031532  105037  026562                      CLRB    ACTDRV          ;CLEAR "ACTIVE DRIVER" FLAG
5606  031536  000002                              RTI                     ;RETURN
5607
5608                                      ;TRANSFER DONE ROUTINE
5609
5610  031540  105061  026506              TD:     CLRB    DRVACT(R1)      ;SET DRIVE ACTIVE INDICATOR TO IDLE
5611  031544  012737  177777  026630              MOV     #-1,DTUW        ;NO DATA TRANSFERS UNDERWAY
5612  031552  006301                              ASL     R1
5613  031554  012761  177777  026610              MOV     #-1,TIMER(R1)   ;CANCEL TIMEOUT
5614  031562  006201                              ASR     R1
5615  031564  013702  026556                      MOV     TRNSWT,R2       ;GET "DPB" ADDRESS FROM THE
5616  031570  005037  026556                      CLR     TRNSWT          ;TRANSFER WAIT QUEUE--CLEAR QUEUE
5617  031574  052762  000200  000016              BIS     #BIT07,16(R2)   ;SET DONE
5618  031602  010164  000010                      MOV     R1,RMCS2(R4)    ;SELECT THE DRIVE
5619  031606  004037  033470                      JSR     R0,RD.RM        ;TRANSFER ERROR(TRE=1)?
5620  031612  000000                              RMCS1
5621  031614  030762                              CI7
5622  031616  006126                              ROL     (SP)+
5623  031620  100417                              BMI     3$              ;BR IF YES
```

```
5624  031622  005737  026604              TST     SAVEFG          ;SAVE THE RH70/RM03 REGISTERS?
5625  031626  100002                      BPL     1$              ;BRANCH IF NO
5626  031630  004737  034042              JSR     PC,SVRH70       ;YES--SAVE THE REGISTERS
5627  031634                          1$:
5628  031634  004737  034676              JSR     PC,GETREQ       ;GET DPB POINTER
5629  031640  005702                      TST     R2              ;ENTRY FOR DRIVE ?
5630  031642  001403                      BEQ     2$              ;BR IF NOT
5631  031644  004737  027702              JSR     PC,OPT          ;CALL OPTIMIZER
5632  031650  000417                      BR      SC              ;CHECK OTHER DRIVES
5633                          ;THE RELEASE DRIVE COMMAND IS FORCED TO ENTER FOR DUAL PORT OPERATION
5634  031652  012714  000113          2$: MOV     #113,(R4)       ;RELEASE THE DRIVE
5635  031656  000414                      BR      SC              ;CHECK FOR OTHER DRIVES
5636  031660  052762  100100  000016  3$: BIS     #BIT15!BIT06,16(R2) ;SET DATA ERROR FLAG
5637  031666  004737  034602              JSR     PC,EMPTYQ       ;EMPTY THE "DRIVE'S WAIT" QUEUE
5638  031672  004737  034042              JSR     PC,SVRH70       ;SAVE THE RH70/RM03 REGISTERS
5639  031676  012714  040111              MOV     #40111,(R4)     ;ISSUE A "DRIVE CLEAR"
5640  031702  012714  000113              MOV     #113,(R4)       ;ISSUE A RELEASE TO THE DRIVE
5641  031706  000400                      BR      SC              ;CHECK FOR OTHER DRIVES
5642
5643
5644
5645                          ;SPECIAL CONDITION ROUTINE
5646
5647  031710  116403  000016          SC: MOVB    RMAS(R4),R3     ;READ "RMAS"
5648  031714  001014                      BNE     2$              ;BRANCH IF ANY 'ATA' BITS SET
5649  031716  004037  033470              JSR     R0,RD.RM        ;READ CONTROL AND STATUS REGISTER
5650  031722  000000                      RMCS1
5651                          ;           CIB
5652  031724  031744                      1$                      ;EXIT IF FAIL TO READ
5653  031726  106126                      ROLB    (SP)+           ;IS "IE"=1?
5654  031730  100405                      BMI     1$              ;YES, NO DRIVES TO CHECK
5655  031732  004037  034766              JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
5656  031736  104001                      ERROR   1               ;REPORT AN ILLEGAL INTERRUPT
5657  031740  004737  034160              JSR     PC,SET.IE       ;SET INTERRUPT ENABLE
5658  031744  000207                  1$: RTS     PC              ;RETURN
5659  031746  005046                  2$: CLR     -(SP)           ;PROCESS ALL DRIVES THAT HAVE
5660  031750  110316                      MOVB    R3,(SP)         ;AN "ATA"=1
5661  031752  012703  000001              MOV     #1,R3
5662  031756  005001                      CLR     R1
5663  031760  030316              SC3:    BIT     R3,(SP)         ;ATA=1?
5664  031762  001005                      BNE     SC5             ;YES--BRANCH
5665  031764  005201              SC4:    INC     R1              ;MOVE TO THE NEXT DRIVE
5666  031766  106303                      ASLB    R3
5667  031770  001373                      BNE     SC3             ;BRANCH IF MORE TO CHECK?
5668  031772  005726                      TST     (SP)+           ;CLEAN OFF THE STACK
5669  031774  000207                      RTS     PC              ;RETURN TO USER
5670  031776  105761  026536          SC5: TSTB   DPINT(R1)       ;INITIALIZING THE DRIVE ?
5671  032002  001402                      BEQ     1$              ;BR IF NOT
5672  032004  000137  032720              JMP     SC13            ;PROCESS THE DRIVE
5673  032010  105761  026546          1$: TSTB    DPRQS(R1)       ;PORT REQUEST OUTSTANDING ?
5674  032014  001402                      BEQ     2$              ;BR IF NOT
5675  032016  000137  032720              JMP     SC13            ;START THE OUTSTANDING COMMAND
5676  032022  105761  026516          2$: TSTB    DRVSTA(R1)      ;CHECK THE DRIVE STATUS
5677  032026  003025                      BGT     5$              ;BRANCH IF ONLINE
5678  032030  105761  026564              TSTB    ULDFLG(R1)      ;UNLOAD IN PROGRESS?
5679  032034  003422                      BLE     5$              ;BRANCH IF NOT
```

```
5680  032036  004737  034676                      JSR     PC,GETREQ       ;GET DPB POINTER
5681  032042  004737  034042                      JSR     PC,SVRH70       ;SAVE THE RH70/RM03 REGISTERS
5682  032046  004737  032656                      JSR     PC,SC12         ;SAVE RMDS, RMER1, RMER2, AND RMMR2
5683                                                                        ;ALSO DO A DRIVE INIT (DRVINT)
5684  032052  105761  026516                      TSTB    DRVSTA(R1)      ;DID DRIVE COME ONLINE?
5685  032056  003416                              BLE     6$              ;NO---BRANCH
5686  032060  032737  040000  026476              BIT     #BIT14,RMERRS   ;WAS THERE AN ERROR?
5687  032066  001002                              BNE     3$              ;BR IF ERROR
5688  032070  000137  032546                      JMP     SC11            ;NO ERROR
5689  032074  013705  026500          3$:         MOV     RMERRS+2,R5     ;YES -- PICKUP RMER1 AND
5690  032100  000504                              BR      SC6A            ;GO PROCESS THE ERROR
5691  032102  105761  026506          5$:         TSTB    DRVACT(R1)      ;DRIVE ACTIVE WITH COMMAND OR ERROR RECOVERY ?
5692  032106  001033                              BNE     SC6             ;BR IF EITHER
5693  032110  004737  032656                      JSR     PC,SC12         ;SAVE RMDS, RMER1, RMER2, AND RMMR2
5694                                                                        ;ALSO DO A DRVINT
5695  032114  105761  026536          6$:         TSTB    DPINT(R1)       ;TRYING TO INIT THE DRIVE ?
5696  032120  001321                              BNE     SC4             ;BR IF YES, CHECK ON MORE DRIVES
5697  032122  105761  026516                      TSTB    DRVSTA(R1)      ;CHECK ON DRIVE'S STATUS
5698  032126  100413                              BMI     7$              ;BR IF UNSAFE
5699  032130  032737  020000  026502              BIT     #BIT13,RMERRS+4 ;ADDRESS PLUG CHANGED ?
5700  032136  001013                              BNE     8$              ;BR IF YES
5701                                          :   MOV     #113,-(SP)      ;RELEASE COMMAND
5702  032140  012746  000111                      MOV     #111,-(SP)      ;DRIVE CLEAR
5703  032144  004037  033650                      JSR     R0,WRT.RM       ;WRITE THE COMMAND INTO RMCS1
5704  032150  000000                              RMCS1                   ;REGISTER INDEX
5705  032152  032516                              SCB                     ;PARITY EXIT ADDRESS
5706  032154  011605          7$.                 MOV     (SP),R5         ;PICKUP (RMAS) BEFORE THE ERROR CALL
5707  032156  004037  034766                      JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
5708  032162  104002                              ERROR   2               ;REPORT THE UNEXPECTED ATTENTION
5709  032164  000677                              BR      SC4             ;GO CHECK FOR MORE ATA'S
5710  032166                                  8$:
5711  032166  004037  034766                      JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
5712  032172  104005                              ERROR   5               ;REPORT THE ADDRESS PLUG CHANGE
5713  032174  000673                              BR      SC4             ;CHECK FOR MORE DRIVES
5714  032176  006301                  SC6:        ASL     R1              ;SETUP TO ADDRESS WORDS
5715  032200  012761  177777  026610              MOV     #-1,TIMER(R1)   ;STOP THE TIMER
5716  032206  006201                              ASR     R1              ;RESTORE THE DRIVE ADDRESS
5717  032210  004737  034676                      JSR     PC,GETREQ       ;GET THE DPB POINTER FROM THE QUEUE
5718  032214  010164  000010                      MOV     R1,RMCS2(R4)    ;SELECT DRIVE
5719  032220  000137  032546                      JMP     SC11            ;PROCESS THE SEARCH
5720  032224  004037  033470                      JSR     R0,RD.RM        ;READ THE RM03'S STATUS REG.
5721  032230  000012                              RMDS
5722  032232  032516                              SCB
5723  032234  011605                              MOV     (SP),R5         ;AND PUT IT IN R5
5724  032236  006126                              ROL     (SP)+           ;WAS THERE AN ERROR?
5725  032240  100407                              BMI     1$              ;BR IF ERROR
5726  032242  105761  026506                      TSTB    DRVACT(R1)      ;CHECK DRIVE'S STATE
5727  032246  003137                              BGT     SC11            ;BR IF DRIVE ACTIVE WITH ORDER
5728  032250  052762  100210  000016              BIS     #BIT15!BIT07!BIT03,16(R2)  ;INFORM USER OF ERROR RECOVER COMPLETION
5729  032256  000470                              BR      SC7
5730  032260  004037  033470          1$:         JSR     R0,RD.RM        ;READ ERROR REGISTER #1
5731  032264  000014                              RMER1
5732  032266  032516                              SCB
5733  032270  012605                              MOV     (SP)+,R5        ;AND SAVE IT IN R5
5734  032272  004737  034042                      JSR     PC,SVRH70       ;SAVE RH70/RM03 REGISTERS
5735  032276  012746  000111                      MOV     #111,-(SP)      ;ISSUE A DRIVE CLEAR
```

```
5736  032302  004037  033650                  JSR     R0,WRT.RM
5737  032306  000000                          RMCS1
5738  032310  032516                          SCB
5739  032312  006105                  SC6A:   ROL     R5              ;WAS "UNSAFE" CONDITION =1?
5740  032314  100406                          BMI     1$              ;BRANCH IF YES
5741  032316  005702                          TST     R2              ;ANYTHING IN QUEUE ?
5742  032320  001447                          BEQ     SC7             ;BR IF NOT
5743  032322  052762  100240  000016          BIS     #BIT15!BIT07!BIT05,16(R2)       ;INFORM USER OF ERROR
5744  032330  000443                          BR      SC7
5745  032332  004037  033470          1$:     JSR     R0,RD.RM        ;READ DRIVE STATUS REG. #1
5746  032336  000012                          RMDS
5747  032340  032516                          SCB
5748  032342  011605                          MOV     (SP),R5         ;SAVE RMDS IN R5
5749  032344  006126                          ROL     (SP)+           ;"ERR"=1?
5750  032346  100011                          BPL     2$              ;BR IF NO--UNSAFE CLEARED
5751  032350  112761  177777  026516          MOVB    #-1,DRVSTA(R1)  ;DRIVE IS UNSAFE
5752  032356  004737  034042                  JSR     PC,SVRH70       ;SAVE RH70/RM03 REGISTERS
5753  032362  052762  110000  000016          BIS     #BIT15!BIT12,16(R2)     ;INFORM USER OF UNSAFE ERROR
5754  032370  000423                          BR      SC7
5755  032372  032705  010000          2$:     BIT     #BIT12,R5       ;"MOL" = 1 ?
5756  032376  001015                          BNE     3$              ;BR IF YES
5757  032400  112761  177777  026506          MOVB    #-1,DRVACT(R1)  ;ACTIVE ERROR RECOVER
5758  032406  112761  000001  026516          MOVB    #1,DRVSTA(R1)   ;ONLINE
5759  032414  006301                          ASL     R1
5760  032416  012761  072460  026610          MOV     #30000.,TIMER(R1) ;START 30 SECOND TIMER
5761  032424  006201                          ASR     R1
5762  032426  000137  031764                  JMP     SC4
5763  032432  052762  100220  000016  3$:     BIS     #BIT15!BIT07!BIT04,16(R2)       ;INFORM USER OF ERROR
5764  032440  105061  026506          SC7:    CLRB    DRVACT(R1)      ;DRIVE IS IDLE
5765                                  ;       JSR     PC,EMPTYQ       ;DUMP THE QUEUE
5766  032444  004737  034720                  JSR     PC,POPQUE       ;REMOVE THE QUEUE
5767  032450  105761  026564                  TSTB    ULDFLG(R1)      ;UNLOAD IN RMOGRESS OR QUEUE?
5768  032454  003002                          BGT     1$              ;BR IF NOT
5769  032456  105061  026564                  CLRB    ULDFLG(R1)      ;CLEAR UNLOAD FLAG
5770  032462  116164  026632  000016  1$:     MOVB    ATABIT(R1),RMAS(R4)  ;CLEAR ATTENTION BIT
5771  032470  105761  026516                  TSTB    DRVSTA(R1)      ;IS THE DRIVE UNSAFE ?
5772  032474  100406                          BMI     2$              ;BR IF IT IS
5773                                  ,       MOV     #113,-(SP)      ;RELEASE COMMAND
5774  032476  012746  000111                  MOV     #111,-(SP)      ;DRIVE CLEAR COMMAND
5775  032502  004037  033650                  JSR     R0,WRT.RM       ;WRITE THE COMMAND INTO RFCS1
5776  032506  000000                          RMCS1                   ;REGISTER INDEX
5777  032510  032516                          SCB                     ;PARITY EXIT ADDRESS
5778  032512  000137  031764          2$:     JMP     SC4             ;CHECK FOR MORE DRIVES
5779  032516  105761  026506          SC8:    TSTB    DRVACT(R1)      ;IS DRIVE IDLE?
5780  032522  001405                          BEQ     1$              ;YES--BRANCH
5781  032524  004737  034676                  JSR     PC,GETREQ       ;GET DPB POINTER
5782  032530  004737  030762                  JSR     PC,CI7          ;PROCESS THE PARITY ERROR
5783  032534  000402                          BR      2$              ;CONTINUE
5784  032536                          1$:
5785                                  ;       JSR     PC,CI7          ;PROCESS THE PARITY ERROR
5786  032536  004737  031004                  JSR     PC,CI7B         ;PROCESS THE UNCORRECTABLE PARITY ERROR
5787  032542  000137  031764          2$:     JMP     SC4             ;CHECK MORE DRIVES
5788  032546  105761  026564          SC11:   TSTB    ULDFLG(R1)      ;"UNLOAD IN PROGRESS"?
5789  032552  003402                          BLE     1$              ;BRANCH IF NO
5790  032554  105061  026564                  CLRB    ULDFLG(R1)      ;CLEAR UNLOAD FLAG
5791  032560  105061  026506          1$:     CLRB    DRVACT(R1)      ;SET DRIVE IDLE
```

```
5792  032564  136137  026632  026560          BITB    ATABIT(R1),SRCHWT  ;DOING A SEARCH OPERATION FOR
5793                                                                     ;AN I/O COMMAND?
5794  032572  001012                          BNE     2$                 ;BRANCH IF YES
5795  032574  004737  034720                  JSR     PC,POPQUE          ;REMOVE REQUEST FROM QUEUE
5796  032600  052762  000200  000016          BIS     #BIT07,16(R2)      ;SET "DONE" BIT
5797  032606  005737  026604                  TST     SAVEFG             ;SAVE THE REGISTERS?
5798  032612  100002                          BPL     2$                 ;BRANCH IF NO
5799  032614  004737  034042                  JSR     PC,SVRH70          ;YES--SAVE ALL OF THE RH70/RM03 REG'S
5800  032620  116164  026632  000016  2$:     MOVB    ATABIT(R1),RMAS(R4)  ;CLEAR ATTENTION BIT
5801  032626  146137  026632  026560          BICB    ATABIT(R1),SRCHWT    ;CLEAR IMPLIED SEEK SET
5802  032634  006301                          ASL     R1                 ;WORD INDEX
5803  032636  012761  177777  026610          MOV     #-1,TIMER(R1)      ;STOP CLOCK
5804  032644  006201                          ASR     R1                 ;RESTORE R1
5805  032646  004737  027702                  JSR     PC,OPT             ;START A REQUEST
5806  032652  000137  031764                  JMP     SC4                ;CHECK FOR MORE DRIVES
5807  032656  010164  000010          SC12:   MOV     R1,RMCS2(R4)       ;SELECT DRIVE
5808  032662  016437  000012  026476          MOV     RMDS(R4),RMERRS    ;SAVE THE FOUR REGISTERS THAT
5809  032670  016437  000014  026500          MOV     RMER1(R4),RMERRS+2     ;WILL TELL US SOMETHING
5810  032676  016437  000042  026502          MOV     RMER2(R4),RMERRS+4
5811  032704  016437  000040  026504          MOV     RMMR2(R4),RMERRS+6
5812                                  ;       JSR     R0,DRVINT          ;INIT. THE STATE OF THE DRIVE
5813                                  ;       BR      1$                 ;TAKE ERROR EXIT
5814  032712  000207                          RTS     PC                 ;RETURN
5815  032714  005726                  1$:     TST     (SP)+              ;POP PC OFF OF THE STACK
5816  032716  000677                          BR      SC8                ;PROCESS THE PARITY ERROR
5817  032720  006301                  SC13:   ASL     R1                 ;SETUP TO ADDRESS WORDS
5818  032722  012761  177777  026610          MOV     #-1,TIMER(R1)      ;STOP THE TIMER
5819  032730  006201                          ASR     R1                 ;
5820  032732  010164  000010                  MOV     R1,RMCS2(R4)       ;SELECT THE DRIVE
5821  032736  116164  026632  000016          MOVB    ATABIT(R1),RMAS(R4)  ;CLEAR THE ATTENTION BIT
5822  032744  105761  026536          1$:     TSTB    DPINT(R1)          ;INITIALIZING THE DRIVE ?
5823  032750  001424                          BEQ     2$                 ;BR IF NOT
5824  032752  105061  026536                  CLRB    DPINT(R1)          ;CLEAR THE INIT INDICATOR
5825  032756  004037  027074                  JSR     R0,DRVINT          ;GO INIT THE DRIVE
5826  032762  000240                          NOP                        ;DUMMY PARITY ERROR RETURN
5827  032764  105761  026516                  TSTB    DRVSTA(R1)         ;DRIVE ONLINE ?
5828  032770  003014                          BGT     2$                 ;BR IF YES -- START ORDER
5829  032772  005702                          TST     R2                 ;QUEUE ENTRY FOR THE DRIVE
5830  032774  001426                          BEQ     3$                 ;BR IF NOT
5831  032776  004737  034676                  JSR     PC,GETREQ          ;GET DPB ADDRESS
5832  033002  052762  140000  000016          BIS     #BIT15!BIT14,16(R2)  ;INFORM USER THAT DRIVE OFFLINE
5833  033010  004737  034042                  JSR     PC,SVRH70          ;SAVE THE REGISTERS
5834                                  ;       JSR     PC,EMPTYQ          ;EMPTY THE REQUEST QUEUE
5835  033014  004737  034720                  JSR     PC,POPQUE          ;REMOVE THE QUEUE
5836  033020  000414                          BR      3$
5837  033022  032764  004000  000000  2$:     BIT     #BIT11,RMCS1(R4)       ;DVA SET ?
5838  033030  001006                          BNE     4$                 ;SET THEN CALL OPT
5839  033032  006301                          ASL     R1
5840  033034  012761  060000  026610          MOV     #60000,TIMER(R1)
5841  033042  006201                          ASR     R1
5842  033044  000402                          BR      3$
5843  033046  004737  027702          4$:     JSR     PC,OPT             ;START THE PENDING REQUEST
5844  033052  000137  031764          3$:     JMP     SC4                ;PROCESS OTHER DRIVES
5845
5846                                  ;/RM03 TIMER ROUTINE
5847                                  ;CALL
```

```
5848                                            ;       MOV     #TIME,-(SP)     ;ELASPED TIME IN MILLISECONDS ON THE STACK
5849                                            ;       JSR     PC,RPTMR        ;CALL RM03 TIME ROUTINE
5850
5851  033056  005737  026562          RPTMR:  TST     ACTDRV          ;CHECK "ACTDRV & ACTSTR"
5852  033062  001027                          BNE     4$              ;IF NON ZERO EXIT
5853  033064  112737  000001  026563        MOVB    #1,ACTSTR       ;SET "ACTSTR"
5854  033072  104412                          SAVREG                  ;SAVE R0 - R5
5855  033074  005001                          CLR     R1              ;START WITH DRIVE 0
5856  033076  005003                          CLR     R3
5857  033100  005763  026610          1$:     TST     TIMER(R3)       ;IS THE TIMER RUNNING?
5858  033104  002406                          BLT     2$              ;BRANCH IF NO
5859  033106  166663  000002  026610        SUB     2(SP),TIMER(R3) ;COUNT THE INTERVAL
5860  033114  003002                          BGT     2$              ;BR IF NO SOFTWARE TIMEOUT
5861  033116  004737  033146                JSR     PC,STO          ;CALL SOFTWARE TIMEOUT ROUTINE
5862  033122  005201                  2$:     INC     R1              ;MOVE TO NEXT DRIVE
5863  033124  005723                          TST     (R3)+
5864  033126  022701  000010                  CMP     #8.,R1          ;OUT OF DRIVES?
5865  033132  003362                          BGT     1$              ;BRANCH IF NO
5866  033134  104413                  3$:     RESREG                  ;RESTORE R0 - R5
5867  033136  105037  026563                CLRB    ACTSTR          ;ZERO ACTIVE SOFTWARE TIMEOUT ROUTINE FLAG
5868  033142  012616                  4$:     MOV     (SP)+,(SP)      ;ADJUST THE STACK
5869  033144  000207                          RTS     PC              ;RETURN
5870
5871                                            ;SOFTWARE TIMEOUT ROUTINE
5872                                            ;
5873                                            ;NOTE: THIS ROUTINE MUST BE ENTERED AT PRIORITY 6
5874                                            ;       OR GREATER
5875                                            ;
5876                                            ;CALL:  STO
5877                                            ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
5878                                            ;       JSR     PC,STO          ;CALL
5879                                            ;       RETURN
5880
5881  033146  010146                  STO:    MOV     R1,-(SP)        ;SAVE R1
5882  033150  010246                          MOV     R2,-(SP)        ;SAVE R2
5883  033152  010346                          MOV     R3,-(SP)        ;SAVE R3
5884  033154  010446                          MOV     R4,-(SP)        ;SAVE R4
5885  033156  013704  026644                  MOV     RMADR,R4        ;GET ADDRESS OF "RMCS1"
5886  033162  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT THE DRIVE
5887  033166  004037  033470                  JSR     R0,RD.RM        ;READ "DRIVE STATUS REG"
5888  033172  000012                          RMDS
5889                                  ;       STOS
5890  033174  033456                          STO9
5891  033176  105726                          TSTB    (SP)+           ;IS "DRY"=1?
5892  033200  100436                          BMI     STO2            ;BR IF YES
5893  033202  105761  026536          STO1:   TSTB    DPINT(R1)       ;TRYING TO INTIALIZE THE DRIVE ?
5894  033206  001033                          BNE     STO2            ;BR IF YES
5895  033210  105761  026546                TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
5896  033214  001030                          BNE     STO2            ;BR IF YES
5897  033216  013702  026556                  MOV     TRNSWT,R2       ;PICKUP TRANSFER WAIT QUEUE
5898  033222  020137  026630                  CMP     R1,DTUW         ;TRANSFER UNDERWAY ON THIS DRIVE?
5899  033226  001404                          BEQ     1$              ;BRANCH IF YES
5900  033230  000137  033456                  JMP     STO9            ;IF NOT DON'T BOTHER DRIVES
5901  033234  004737  034676                  JSR     PC,GETREQ       ;GET DPB ADDRESS
5902  033240  052762  101000  000016  1$:     BIS     #BIT15!BIT09,16(R2) ;SET THE ERROR FLAGS
5903  033246  004737  034042                  JSR     PC,SVRH70       ;SAVE RH70/RM03 REGISTERS
```

```
5904                                        ;       MOV     #BIT05,RMCS2(R4) ;"INIT" THE MASS BUS
5905  033252  105061  026506                        CLRB    DRVACT(R1)       ;DRIVE IS IDLE
5906  033256  105061  026564                        CLRB    ULDFLG(R1)       ;CLEAR THE UNLOAD FLAG
5907  033262  005037  026556                        CLR     TRNSWT           ;CLEAR DPB ADDRESS
5908  033266  012737  177777  026630                MOV     #-1,DTUW         ;CLEAR THE TRANSFER DRIVE #
5909  033274  000470                                BR      ST09             ;DON'T BOTHER OTHER DRIVES
5910  033276  116405  000016          ST02:         MOVB    RMAS(R4),R5      ;READ ATTENTION REG
5911  033302  136105  026632                        BITB    ATABIT(R1),R5    ;IS ATTENTION FOR THIS DRIVE UP ?
5912  033306  001007                                BNE     ST03             ;YES--BRANCH
5913  033310  105761  026536                        TSTB    DPINT(R1)        ;TRYING TO INTIALIZE THE DRIVE ?
5914  033314  001021                                BNE     ST06             ;BR IF YES - DRIVE NOT ONLINE
5915  033316  105761  026546                        TSTB    DPRQS(R1)        ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
5916  033322  001035                                BNE     ST07             ;BR IF YES - NO RESPONSE TO REQUEST
5917  033324  000454                                BR      ST09             ;OTHER WISE EXIT
5918  033326  105761  026536          ST03:         TSTB    DPINT(R1)        ;INITIALIZING THE DRIVE ?
5919  033332  001003                                BNE     1$               ;BR IF INIT PENDING
5920  033334  105761  026546                        TSTB    DPRQS(R1)        ;PORT REQUEST PENDING ?
5921  033340  001446                                BEQ     ST09             ;BR IF NOT
5922  033342  012763  177777  026610    1$:         MOV     #-1,TIMER(R3)    ;STOP THE TIMER
5923  033350  000442                                BR      ST09             ;EXIT
5924  033352  004737  031062          ST05:         JSR     PC,CI8           ;GO HANDLE THE PARITY ERROR
5925  033356  000437                                BR      ST09
5926  033360  105061  026536          ST06:         CLRB    DPINT(R1)        ;CLEAR THE INITIALIZE INDICATOR
5927  033364  105061  026516                        CLRB    DRVSTA(R1)       ;SET UNIT OFFLINE
5928  033370  012763  177777  026610                MOV     #-1,TIMER(R3)  ;STOP THE TIMER
5929  033376  004737  034676                        JSR     PC,GETREQ        ;GET THE DPB ADDRESS
5930  033402  005702                                TST     R2               ;REQUEST IN QUEUE ?
5931  033404  001424                                BEQ     ST09             ;BR IF NOT
5932  033406  052762  140000  000016                BIS     #BIT15!BIT14,16(R2)  ;INFORM THE USER DRIVE NOT AVAILABLE
5933  033414  000414                                BR      ST08             ;FINISH
5934  033416  012763  177777  026610    ST07:       MOV     #-1,TIMER(R3)    ;STOP THE TIMER
5935  033424  105061  026546                        CLRB    DPRQS(R1)        ;CLEAR PORT REQUEST INDICATOR
5936  033430  004737  034676                        JSR     PC,GETREQ        ;GET DPB ADDRESS
5937  033434  005702                                TST     R2               ;QUEUE ENTRY FOR DRIVE ?
5938  033436  001407                                BEQ     ST09             ;BR IF NONE
5939  033440  012762  100004  000016                MOV     #BIT15!BIT2,16(R2)  ;INFORM USER OF PORT REQUEST ERROR
5940  033446  004737  034602          ST08:         JSR     PC,EMPTYQ        ;CLEAR THE QUEUE FOR THE DRIVE
5941  033452  004737  034042                        JSR     PC,SVRH70        ;SAVE THE REGISTERS
5942  033456  012604                    ST09:       MOV     (SP)+,R4         ;RESTORE R4
5943  033460  012603                                MOV     (SP)+,R3         ;RESTORE R3
5944  033462  012602                                MOV     (SP)+,R2         ;RESTORE R2
5945  033464  012601                                MOV     (SP)+,R1         ;RESTORE R1
5946  033466  000207                                RTS     PC               ;RETURN
5947
5948                                        ;ROUTINE TO READ A RH70/RM03 REGISTER
5949                                        ;
5950                                        ;CALL
5951                                        ;       JSR     R0,RD.RM         ;GO READ A REGISTER
5952                                        ;       INDEX                    ;REG. INDEX FROM BASE
5953                                        ;       ERRADR                   ;ERROR ADDRESS--PROCESS ERROR STARTING
5954                                        ;                                ;AT THIS ADDRESS
5955                                        ;       RETURN                   ;CONTENTS OF REG. IS ON THE STACK
5956                                        ;
5957  033470  013737  026642  033636  RD.RM:       MOV     MCPEMX,RD.RM2    ;MAX. RETRYS ALLOWED
5958  033476  011646                              MOV     (SP),-(SP)       ;SAVE R0 FOR RETURN
5959  033500  013737  026644  033514              MOV     RMADR,RD.ADR     ;FORM THE DESIRED ADDRESS
```

```
5960  033506  062037  033514            ADD     (R0)+,RD.ADR    ;USING THE BASE AND THE INDEX
5961  033512  013727                RD.RM1: MOV     @(PC)+,(PC)+    ;READ THE DESIRED REGISTER OF THE RM03
5962  033514  000000                RD.ADR: .WORD   0               ;ADDRESS IS FORMED HERE
5963  033516  000000                RD.WRD: .WORD   0               ;REG. CONTENTS PUT HERE
5964  033520  013766  033516  000002        MOV     RD.WRD,2(SP)    ;RETURN IT TO THE USER
5965  033526  013746  026644            MOV     RMADR,-(SP)     ;PUT THE ADDRESS ON THE STACK
5966  033532  062716  000010            ADD     #RMCS2,(SP)     ;FORM THE ADDRESS OF RMCS2
5967  033536  032736  010000            BIT     #BIT12,@(SP)+   ;CHECK THE 'NED' BIT
5968  033542  001037                    BNE     RD.RM3          ;BR IF DRIVE NON-EXISTENT
5969  033544  017746  173074            MOV     @RMADR,-(SP)    ;READ RMCS1
5970  033550  032716  020000            BIT     #BIT13,(SP)     ;DID MCPE SET?
5971  033554  001002                    BNE     1$              ;BRANCH IF YES
5972  033556  022620                    CMP     (SP)+,(R0)+     ;ADJUST FOR RETURN
5973  033560  000432                    BR      RD.RM4          ;EXIT
5974  033562                        1$:
5975  033562  004037  034766            JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
5976  033566  104003                    ERROR   3               ;REPORT "MCPE" ERROR
5977  033570  005737  026630            TST     DTUW            ;DATA TRANSFER UNDERWAY?
5978  033574  100405                    BMI     2$              ;NO--BRANCH
5979  033576  032716  040000            BIT     #BIT14,(SP)     ;NO--"TRE"=1?
5980  033602  001402                    BEQ     2$              ;NO--BRANCH
5981  033604  005726                    TST     (SP)+           ;YES--CLEAN OFF THE STACK AND
5982  033606  000415                    BR      RD.RM3          ;TAKE THE FATAL ERROR EXIT
5983  033610  052716  040000        2$: BIS     #BIT14,(SP)     ;CLEAR "MCPE" BY SENDING A "1" TO "TRE"
5984  033614  000316                    SWAB    (SP)            ;POSITION BEFORE WRITING
5985  033616  013737  026644  033632    MOV     RMADR,3$        ;FORM ADDRESS OF HIGH BYTE
5986  033624  005237  033632            INC     3$
5987  033630  112637                    MOVB    (SP)+,@(PC)+    ;WRITE THE HIGH BYTE OF RMCS1
5988  033632  000000                3$: .WORD   0               ;ADDRESS STORAGE
5989  033634  005327                    DEC     (PC)+           ;EXCEEDED MAX. RETRYS
5990  033636  000003                RD.RM2: .WORD   3
5991  033640  002324                    BGE     RD.RM1          ;BRANCH IF NO
5992  033642  011000                RD.RM3: MOV     (R0),R0         ;FATAL ERROR EXIT
5993  033644  012616                    MOV     (SP)+,(SP)
5994  033646  000200                RD.RM4: RTS     R0
5995
5996                                ;ROUTINE TO WRITE A REGISTER
5997                                ;
5998                                ;CALL
5999                                ;       MOV     DATA,-(SP)      ;DATA TO BE LOADED ON THE STACK
6000                                ;       JSR     R0,WRT.RM       ;CALL THE ROUTINE TO LOAD(WRITE) THE REG.
6001                                ;       INDEX                   ;INDEX OF THE REGISTER TO BE LOADED
6002                                ;       ERRADR                  ;ADDRESS TO RETURN TO ON AN ERROR
6003                                ;       RETJRN                  ;ERROR FREE RETURN
6004
6005  033650  013737  026642  034026  WRT.RM: MOV     MCPEMX,WRT.R2   ;MAX RETRYS ALLOWED
6006  033656  016637  000002  033736        MOV     2(SP),WRT.WD    ;SAVE THE WORD TO WRITE
6007  033664  012616                    MOV     (SP)+,(SP)      ;ADJUST THE STACK
6008  033666  012037  033740            MOV     (R0)+,WRT.AD    ;GET INDEX OF REGISTER TO BE WRITTEN
6009  033672  001015                    BNE     1$              ;BRANCH IF NOT RMCS1
6010  033674  122737  000.50  033736    CMPB    #150,WRT.WD     ;IS THE COMMAND FOR DATA TRANSFERS?
6011  033702  002411                    BLT     1$              ;YES--DON'T GET THE OLD A16 & A17, & PSEL
6012  033704  004037  033470            JSR     R0,RD 7M        ;NO---COMBINE A16&A17, & PSEL WITH
6013  033710  000000                    RMCS1                   ;THE COMMAND BEFORE SENDING IT TO
6014  033712  034032                    WRT.R3                  ;THE RH70/RM03
6015  033714  000316                    SWAB    (SP)
```

```
6016  033716  042716  177770                    BIC     #^C7,(SP)
6017  033722  112637  033737                    MOVB    (SP)+,WRT.WD+1
6018  033726  063737  026644  033740  1$:       ADD     RMADR,WRT.AD      ;FORM THE ADDRESS OF THE DISK REG.
6019  033734  012737                  WRT.R1:   MOV     (PC)+,@(PC)+      ;LOAD THE DESIRED REG.
6020  033736  000000                  WRT.WD:   .WORD   0                 ;WORD TO WRITE GOES HERE
6021  033740  000000                  WRT.AD:   .WORD   0                 ;ADDRESS IS FORMED H RE
6022  033742  013746  026644                    MOV     RMADR,-(SP)       ;PUT THE ADDRESS ON THE STACK
6023  033746  062716  000010                    ADD     #RMCS2,(SP)       ;FORM THE ADDRESS OF RMCS2
6024  033752  032736  010000                    BIT     #BIT12,@(SP)+     ;CHECK THE 'NED' BIT
6025  033756  001025                            BNE     WRT.R3            ;BR IF DRIVE NON-EXISTENT
6026  033760  004037  033470                    JSR     R0,RD.RM          ;CHECK FOR PARITY ERROR ON WRITE
6027  033764  000014                            RMER1
6028  033766  034032                            WRT.R3
6029  033770  032726  000010                    BIT     #BIT03,(SP)+
6030  033774  001420                            BEQ     WRT.R4            ;BRANCH IF "PAR=0"
6031  033776  016037  177776  034010            MOV     -2(R0),1$         ;PICKUP THE INDEX
6032  034004  004037  033470                    JSR     R0,RD.RM          ;READ THE REG.
6033  034010  000000                  1$:       .WORD   0                 ;REG. INDEX
6034  034012  034032                            WRT.R3                    ;RETURN TO THIS ADDRESS ON ERROR
6035  034014  004037  034766                    JSR     R0,ES.SAV         ;SAVE THE ADDRESS IN '$ESCAPE'
6036  034020  104004                            ERROR   4                 ;REPORT THE PARITY ON WRITE ERROR
6037  034022  005726                            TST     (SP)+             ;CLEAR OFF THE STACK
6038  034024  005327                            DEC     (PC)+             ;DECREMENT THE ERROR COUNT
6039  034026  000003                  WRT.R2:   .WORD   3                 ;RETRY COUNTER
6040  034030  002341                            BGE     WRT.R1            ;TRY AGAIN IF NOT FINISHED
6041  034032  011000                  WRT.R3:   MOV     (R0),R0           ;TAKE THE "PARITY ON WRITE" ERROR EXIT
6042  034034  000401                            BR      WRT.R5            ;EXIT
6043  034036  005720                  WRT.R4:   TST     (R0)+             ;ADJUST FOR ERROR FREE EXIT
6044  034040  000200                  WRT.R5:   RTS     R0
6045
6046                                            ;ROUTINE TO SAVE THE RH70/RP04/5/RM03 REGISTERS AS PER DPB+14
6047                                            ;
6048                                            ;CALL
6049                                            ;       MOV     #DPBNUM,R2        ;DPB POINTER TO R2
6050                                            ;       JSR     PC,SVRH70         ;SAVE THE DRIVES REG'S
6051
6052  034042  104412                  SVRH70:   SAVREG                    ;SAVE R0 - R5
6053  034044  005702                            TST     R2                ;QUEUE ENTRY FOR THE DRIVE ?
6054  034046  001442                            BEQ     6$                ;BR IF NONE
6055  034050  013704  026644                    MOV     RMADR,R4
6056  034054  111264  000010                    MOVB    (R2),RMCS2(R4)    ;SELECT DRIVE
6057  034060  016203  000014                    MOV     14(R2),R3         ;GET THE ERROR TABLE POINTER
6058  034064  001433                            BEQ     6$                ;EXIT IF NO ADDRESS
6059  034066  005037  034122                    CLR     3$                ;COUNTER & POINTER
6060  034072  023727  034122  000022  1$:       CMP     3$,#RMDB          ;REACHED THE BUFFER REGISTER ?
6061  034100  001006                            BNE     2$                ;BR IF NOT
6062  034102  032764  000200  000010            BIT     #BIT07,RMCS2(R4)  ;'OR' SET ?
6063  034110  001002                            BNE     2$                ;BR IF SET
6064  034112  005023                            CLR     (R3)+             ;STORE RMDB AS ZEROES
6065  034114  000405                            BR      4$                ;CONTINUE
6066  034116  004037  033470          2$:       JSR     R0,RD.RM          ;READ THE SELECTED REGISTER
6067  034122  000000                  3$:       .WORD   0                 ;REGISTER INDEX
6068  034124  034150                            5$                        ;ERROR RETURN ADDRESS
6069  034126  012623                            MOV     (SP)+,(R3)+       ;STORE THE REGISTER CONTENTS
6070  034130  023727  034122  000046  4$:       CMP     3$,#RMEC2         ;REACHED THE END ?
6071  034136  001406                            BEQ     6$                ;BR IF YES
```

```
 6072  034140  062737  000002  034122          ADD     #2,3$           ;INCREMENT THE REGISTER INDEX
 6073  034146  000751                          BR      1$              ;CONTINUE READING THE REGISTERS
 6074  034150  004737  030762          5$:     JSR     PC,CI7          ;PROCESS THE UNCORRECTABLE PARITY ERROR
 6075  034154  104413                  6$:     RESREG                  ;RESTORE R0 - R5
 6076  034156  000207                          RTS     PC              ;RETURN
 6077
 6078                                  ;ROUTINE TO SET THE INTERRUPT WITHOUT GETTING A "TRE"
 6079                                  ;CALL
 6080                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
 6081                                  ;       JSR     PC,SET.IE       ;SET "IE"
 6082                                  ;       RETURN
 6083
 6084  034160  010446                  SET.IE: MOV     R4,-(SP)        ;SAVE R4
 6085  034162  013704  026644                  MOV     RMADR,R4        ;PICKUP ADDRESS OF RMCS1
 6086  034166  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT DRIVE
 6087  034172  011446                          MOV     (R4),-(SP       ;READ RMCS1
 6088  034174  052716  040000                  BIS     #BIT14,(SP)     ;SET THE "TRE" BIT OF THE WORD READ
 6089  034200  000316                          SWAB    (SP)            ;ADJUST FOR DATO
 6090  034202  112714  000100                  MOVB    #BIT06,(R4)     ;SET "IE"
 6091  034206  032764  010000  000010          BIT     #BIT12,RMCS2(R4) ;IS "NED"=1?
 6092  034214  001002                          BNE     1$              ;YES--CLEAR "TRE"
 6093  034216  005726                          TST     (SP)+           ;CLEAN OFF THE STACK
 6094  034220  000402                          BR      2$
 6095  034222  112664  000001          1$:     MOVB    (SP)+,1(R4)     ;CLEAR "TRE"
 6096  034226  012604                  2$:     MOV     (SP)+,R4        ;RESTORE R4
 6097  034230  000207                          RTS     PC              ;RETURN TO CALLER
 6098
 6099                                  ;QUEUE COUNT
 6100  034232      000                 QCNT:    .BYTE   0               ;DRIVE 0
 6101  034233      000                          BYTE    0               ;DRIVE 1
 6102  034234      000                          .BYTE   0               ;DRIVE 2
 6103  034235      000                          .BYTE   0               ;DRIVE 3
 6104  034236      000                          .BYTE   0               ;DRIVE 4
 6105  034237      000                          .BYTE   0               ;DRIVE 5
 6106  034240      000                          .BYTE   0               ;DRIVE 6
 6107  034241      000                          .BYTE   0               ;DRIVE 7
 6108
 6109                                  ;QUEUE INPUT POINTERS
 6110
 6111  034242  034324                  QINPT:   .WORD   QDRV0           ;DRIVE 0
 6112  034244  034344                           .WORD   QDRV1           ;DRIVE 1
 6113  034246  034364                           .WORD   QDRV2           ;DRIVE 2
 6114  034250  034404                           .WORD   QDRV3           ;DRIVE 3
 6115  034252  034424                           .WORD   QDRV4           ;DRIVE 4
 6116  034254  034444                           .WORD   QDRV5           ;DRIVE 5
 6117  034256  034464                           .WORD   QDRV6           ;DRIVE 6
 6118  034260  034504                           .WORD   QDRV7           ;DRIVE 7
 6119
 6120                                  ;QUEUE OUTPUT POINTERS
 6121
 6122  034262  034324                  QOUTPT:  .WORD   QDRV0           ;DRIVE 0
 6123  034264  034344                           WORD    QDRV1           ;DRIVE 1
 6124  034266  034364                           .WORD   QDRV2           ;DRIVE 2
 6125  034270  034404                           .WORD   QDRV3           ;DRIVE 3
 6126  034272  034424                           .WORD   QDRV4           ;DRIVE 4
 6127  034274  034444                           .WORD   QDRV5           ;DRIVE 5
```

```
6128  034276  034464                        .WORD   QDRV6           ;DRIVE 6
6129  034300  034504                        .WORD   QDRV7           ;DRIVE 7
6130
6131  034302  034324                QSTART: .WORD   QDRV0           ;DRIVE 0 START ADDRESS
6132  034304  034344                QSTOP:  .WORD   QDRV1           ;DRIVE 0 STOP ADDRESS & DRIVE 1 START ADDRESS
6133  034306  034364                        .WORD   QDRV2           ;STOP DRIVE 1--START DRIVE 2
6134  034310  034404                        .WORD   QDRV3           ;STOP DRIVE 2--START DRIVE 3
6135  034312  034424                        .WORD   QDRV4           ;STOP DRIVE 3--START DRIVE 4
6136  034314  034444                        .WORD   QDRV5           ;STOP DRIVE 4--START DRIVE 5
6137  034316  034464                        .WORD   QDRV6           ;STOP DRIVE 5--START DRIVE 6
6138  034320  034504                        .WORD   QDRV7           ;STOP DRIVE 6--START DRIVE 7
6139  034322  034524                        .WORD   QTERM           ;STOP DRIVE 7
6140
6141                                ;DRIVE REQUEST QUEUES
6142
6143  034324  000010                QDRV0:  .BLKW   10
6144  034344  000010                QDRV1:  .BLKW   10
6145  034364  000010                QDRV2:  .BLKW   10
6146  034404  000010                QDRV3:  .BLKW   10
6147  034424  000010                QDRV4:  .BLKW   10
6148  034444  000010                QDRV5:  .BLKW   10
6149  034464  000010                QDRV6:  .BLKW   10
6150  034504  000010                QDRV7:  .BLKW   10
6151          034524                QTERM=.
6152
6153                                ;ROUTINE TO CLEAR ALL OF THE REQUEST QUEUES
6154                                ;
6155                                ;CALL
6156                                ;       JSR     PC,CLRQUE
6157
6158  034524  104413                CLRQUE: SAVREG                  ;SAVE R0 - R5
6159  034526  012702  034232                MOV     #QCNT,R2        ;ZERO THE QUEUE COUNTS
6160  034532  005022                        CLR     (R2)+           ;DRIVES 0 & 1
6161  034534  005022                        CLR     (R2)+           ;DRIVES 2 & 3
6162  034536  005022                        CLR     (R2)+           ;DRIVES 4 & 5
6163  034540  005022                        CLR     (R2)+           ;DRIVES 6 & 7
6164  034542  012703  000010                MOV     #8.,R3          ;MOVE THE STARTING
6165  034546  012701  034302                MOV     #QSTART,R1      ;ADDRESS OF THE QUEUE INTO
6166  034552  012122                1$:     MOV     (R1)+,(R2)+     ;THE QUEUE INPUT POINTER
6167  034554  005303                        DEC     R3
6168  034556  001375                        BNE     1$
6169  034560  012703  000010                MOV     #8.,R3          ;MOVE THE STARTING ADDRESS
6170  034564  012701  034302                MOV     #QSTART,R1      ;OF THE QUEUE INTO THE
6171  034570  012122                2$:     MOV     (R1)+,(R2)+     ;QUEUE OUTPUT POINTER
6172  034572  005303                        DEC     R3
6173  034574  001375                        BNE     2$
6174  034576  104413                        RESREG                  ;RESTORE R0 - R5
6175  034600  000207                        RTS     PC
6176
6177                                ;EMPTY THE QUEUE SPECIFIED BY R1
6178                                ;
6179                                ;CALL
6180                                ;       MOV     DRVNUM,R1       ;DRIVE NUMBER TO R1
6181                                ;       JSR     PC,EMPTYQ
6182
6183  034602  105061  034232        EMPTYQ: CLRB    QCNT(R1)        ;CLEAR NUMBER OF ITEMS IN QUEUE
```

```
6184  034606  006301                          ASL     R1
6185  034610  016161  034242  034262          MOV     QINPT(R1),QOUTPT(R1) ;SET OUTPUT QUEUE POINTER=INPUT POINTER
6186  034616  006201                          ASR     R1
6187  034620  000207                          RTS     PC
6188
6189                                  ;ROUTINE TO PUT A REQUEST IN QUEUE
6190                                  ;
6191                                  ;CALL
6192                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
6193                                  ;       MOV     #DPB,R2         ;ADDRESS OF PARAMETER BLOCK
6194                                  ;       JSR     R0,DRVQUE       ;GO PUT REQUEST IN QUEUE
6195                                  ;       RETURN1                 ;RETURN HERE IF QUEUE IS FULL
6196                                  ;       RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE
6197
6198  034622  122761  000010  034232  DRVQUE: CMPB    #10,QCNT(R1)    ;IS QUEUE FULL?
6199  034630  001421                          BEQ     2$              ;BR IF YES-TAKE RETURN1
6200  034632  105261  034232                  INCB    QCNT(R1)        ;INCREMENT QUEUE COUNT
6201  034636  006301                          ASL     R1
6202  034640  010271  034242                  MOV     R2,@QINPT(R1)   ;PUT THIS REQUEST IN QUEUE
6203  034644  062761  000002  034242          ADD     #2,QINPT(R1)    ;UPDATE THE QUEUE POINTER
6204  034652  026161  034242  034304          CMP     QINPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER
6205  034660  001003                          BNE     1$              ;BRANCH IF NO
6206  034662  016161  034302  034242          MOV     QSTART(R1),QINPT(R1) ;YES--RESET POINTER
6207  034670  006201                  1$:     ASR     R1
6208  034672  005720                          TST     (R0)+           ;TAKE RETURN 2
6209  034674  000200                  2$:     RTS     R0              ;RETURN TO USER
6210
6211                                  ;ROUTINE TO GET THE "DPB" ADDRESS OF NEXT REQUEST IN QUEUE
6212                                  ;
6213                                  ;CALL
6214                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
6215                                  ;       JSR     PC,GETREQ       ;GO GET THE REQUEST
5216                                  ;       RETURN                  ;R2="DPB" ADDRESS OF THE REQUEST
6217                                  ;                               ;R2=0 IF NO REQUEST IN QUEUE
6218
6219  034676  005002                  GETREQ: CLR     R2
6220  034700  105761  034232                  TSTB    QCNT(R1)        ;IS THERE ANY REQUEST IN QUEUE?
6221  034704  001404                          BEQ     2$              ;NO---BRANCH
6222  034706  006301                  1$:     ASL     R1
6223  034710  017102  034262                  MOV     @QOUTPT(R1),R2  ;PICKUP "DPB" POINTER FOR THIS DRIVE
6224  034714  006201                          ASR     R1
6225  034716  000207                  2$:     RTS     PC              ;RETURN TO USER
6226
6227                                  ;ROUTINE TO "POP" THE REQUEST FROM QUEUE
6228                                  ;
6229                                  ;CALL
6230                                  ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
6231                                  ;       JSR     PC,POPQUE       ;CALL TO REMOVE REQUEST
6232                                  ;       RETURN                  ;R2=ADDRESS OF DPB REMOVED
6233
6234  034720  105361  034232          POPQUE: DECB    QCNT(R1)        ;DECREMENT QUEUE COUNT
6235  034724  006301                          ASL     R1
6236  034726  017102  034262                  MOV     @QOUTPT(R1),R2  ;GET THE "DPB" POINTER
6237  034732  005071  034262                  CLR     @QOUTPT(R1)     ;REMOVE DPB ADDRESS FROM THE QUEUE
6238  034736  062761  000002  034262          ADD     #2,QOUTPT(R1)   ;UPDATE THE QUEUE POINTER
6239  034744  026161  034262  034304          CMP     QOUTPT(R1),QSTOP(R1) ;TIME TO RESET THE POINTER?
```

```
006240  034752  001003                          BNE     1$                      ;NO--BRANCH TO EXIT
006241  034754  016161  034302  034262          MOV     QSTART(R1),QOUTPT(R1)   ;YES--RESET THE POINTER
006242  034762  006201                  1$:     ASR     R1
006243  034764  000207                          RTS     PC                      ;RETURN TO USER
006244
006245                                  ;ROUTINE TO SAVE THE CONTENTS OF '$ESCAPE' WHEN THE DRIVER
006246                                  ;REPORTS AN ERROR DIRECTLY.
006247                                  ;
006248                                  ;CALL
006249                                  ;       JSR     R0,ES.SAV
006250                                  ;       ERROR   N                       ;;THE ERROR CALL
006251                                  ;       RETURN                          ;THE RETURN IS PAST THE ERROR CALL
006252
006253  034766  012037  035002          ES.SAV: MOV     (R0)+,1$                ;GET THE ERROR CALL
006254  034772  013746  001174                  MOV     $ESCAPE,-(SP)           ;SAVE THE ADDRESS IN '$ESCAPE'
006255  034776  005037  001174                  CLR     $ESCAPE                 ;CLEAR THE ESCAPE RETURN
006256  035002  000000                  1$:     .WORD   0                       ;THE ERROR CALL IS MOVED HERE
006257  035004  012637  001174                  MOV     (SP)+,$ESCAPE           ;RESTORE THE ESCAPE ADDRESS
006258  035010  000200                          RTS     R0                      ;RETURN
006259
006260
006261                                  ;;*****************************************************************
006262                                  .NLIST  BEX

                                        .SBTTL  TELETYPE MESSAGES

035012  047440  043106  044514  UNTOFF: .ASCIZ  / OFFLINE/
035023     040  047117  044514  UNTON:  .ASCIZ  / ONLINE/
035033     040  047516  020124  NOTRM:  .ASCIZ  @ NOT AN RM03/RM02@
035055     040  047516  020124  NOTPRS: .ASCIZ  / NOT PRESENT/
035072  052440  051516  043101  NOTSAF: .ASCIZ  / UNSAFE/
035102  046522  031060     000  RM03B:  .ASCIZ  /RM02/
035107     122  030120  000065  RPO5:   .ASCIZ  /RP05/
035114  046522  031460     000  RM03A:  .ASCIZ  /RM03/
035121     125  044516  020124  SYSTAT: .ASCIZ  /UNIT STATUS/<CR><LF><LF>
035140  005015  051104  053111  MUNIT:  .ASCIZ  <CR><LF>/DRIVE: /
035152  027440  000040          SLASH:  .ASCIZ  @ / @
035156  000103                  C:      .ASCIZ  /C/
035160  000124                  T:      .ASCIZ  /T/
035162  000060                  MDRVD:  .ASCIZ  /0/
035164  020040                  LIN4SP: .ASCII  /   /
035166  020040     000          LINSP:  .ASCIZ  /  /
035171     105  052116  051105  ENTADR: .ASCIZ  /ENTER ADDRESS LIMITS:/<CR><LF>
035221     040  042040  044522  MOFFLN: .ASCIZ  /  DRIVE OFFLINE/<CR><LF>
035243     040  051104  053111  MDRNP:  .ASCIZ  / DRIVE NOT PRESENT/<CR><LF>
035270  042040  044522  042526  MER11:  .ASCIZ  / DRIVE NOT AVAILABLE/<CR><LF>
035317     040  051104  053111  MNRM03: .ASCIZ  @ DRIVE NOT AN RM03@<CR><LF>
035344  042040  044522  042526  MUSDR:  .ASCIZ  / DRIVE UNSAFE/<CR><LF>
035364  051440  046105  041505  MSELD:  .ASCIZ  / SELECTED/
035376  005015  051120  043517  MMODE:  .ASCIZ  <CR><LF>/PROGRAM MODE (C OR F): /
035430  043040  051117  040515  MFORMT. .ASCIZ  / FORMAT & VERIFY/
035451     103  042510  045503  MHECK:  .ASCIZ  /CHECK ONLY/
035464  047506  046522  052101  MORMAT:  .ASCIZ  /FORMAT & VERIFY/
035504  005015  047412  042520  MSIZE:  .ASCIZ  <CR><LF><LF>/OPERATE IN 32 SECTOR MODE (Y OR N) ? /
035555     117  042520  040522  MSEC22: .ASCIZ  /OPERATION WILL BE IN 32 SECTOR (16 BIT) MODE/<CR><LF>
035634  050117  051105  052101  MSEC20: .ASCIZ  /OPERATION WILL BE IN 30 SECTOR (18 BIT) MODE <CR><LF>
```

```
035713     111  053116  046101  BADENT: .ASCIZ  /INVALID ENTRY/<CR><LF>
035733     105  042116  047111  MADRER: .ASCII  /ENDING DSK ADRS MUST BE EQUAL TO OR GREATER/<CR><LF>
036010  044124  047101  051440          .ASCIZ  /THAN STARTING ADRS/<CR><LF>
036035     123  046105  041505  MSELP:  .ASCII  /SELECT DATA PATTERN (BY ENTERING 0,1 OR 2)/<CR><LF>
036111     040  030050  020051          .ASCII  / (0) ZERO'S /<CR><LF>
036127     040  030450  020051          .ASCII  / (1) ONE'S/<CR><LF>
036143     040  031050  020051          .ASCIZ  / (2) WORST CASE: /

036165     127  051117  052123  MPATD:  .ASCIZ  /WORST CASE/
036200  005015  051412  040524  MSFOU:  .ASCIZ  <CR><LF><LF>/STARTING FORMAT ON DRIVE /
036235     015  005012  052123  MSCHK:  .ASCIZ  <CR><LF><LF>/STARTING CHECK ON DRIVE /
036271     015  005012  047506  MFCMPT: .ASCIZ  <CR><LF><LF>/FORMAT COMPLETE, /
036316  005015  041412  042510  MCCMPT: .ASCIZ  <CR><LF><LF>/CHECK COMPLETE, /
036342  047524  040524  020114  NUMERR: .ASCIZ  /TOTAL ERRORS DETECTED: /
036372  051120  051505  047105  ADDRIS: .ASCIZ  /PRESENT ADDRESS IS: /
036417     015  050012  053517  PMSG:   .ASCIZ <CR><LF>/POWER ON ,PROGRAM STARTS AT 200/<LF>
036462  047105  042524  020122  MESG1:  .ASCIZ  /ENTER THE SERIAL NUMBER (MAXIMUM 10 OCTAL DIGITS)  <CR><LF><LF>
036550  005015  051440  020116  SN1:    .ASCIZ  <CR><LF>/ SN #    :  /
036566  005015  053117  051105  MESG3:  .ASCII  <CR><LF>/OVER 126 BAD SECTORS HAVE BEEN DETECTED/<CR><LF>
036641     120  041501  020113          .ASCIZ  /PACK NOT ACCEPTABLE !/
036667     015  044412  041516  MESG2:  .ASCII  <CR><LF>/INCORRECT MANUFACTURE DEFINED BAD SECTOR/
036741     040  047111  047506          .ASCII  / INFORMATION/<CR><LF>
036757     015  044412  044516          .ASCII  <CR><LF>/INITIALIZE THE BAD SECTOR FILE ?/
037021     050  047101  053523          .ASCIZ  /(ANSWER:  Y OR N <CR>)/
037050  005015  046101  043511  MESG4:  .ASCIZ  <CR><LF>/ALIGNMENT PACK, PROGRAM HALT !/
037111     015  051412  046105  MSGBLK: .ASCII <CR><LF>/SELECT ONE OF THE FOLLOWING FUNCTIONS/<CR><LF>
037162  042513  044531  020116          .ASCII  /KEYIN U.S.R. BAD SECTORS .....0/<CR><LF>
037243     120  044522  052116          .ASCII  /PRINT M.F.G. BAD SECTORS.....1/<CR><LF>
037324  051120  047111  020124          .ASCII  /PRINT U.S.R. BAD SECTORS.....2/<CR><LF>
037405     111  044516  044524          .ASCII  /INITIALIZE U.S.R. BAD SECTOR FILE.....3/<CR><LF>
037466  051120  047111  020124          .ASCII  /PRINT A SECTOR OF THE LAST TRACK.....4/<CR><LF>
037547     105  044530  020124          .ASCIZ  /EXIT.....<CR>/<CR><LF>
037631     105  052116  051105  MESG5:  .ASCIZ  /ENTER 16 BIT M.F.G. BAD SECTOR ADDRESS/<CR><LF>
037702  047105  042524  020122  MESG6:  .ASCIZ  /ENTER 18 BIT M.F.G. BAD SECTOR ADDRESS/<CR><LF>
037753     105  052116  051105  MESG7:  .ASCIZ  /ENTER 16 BIT U.S.R. BAD SECTOR ADDRESS/<CR><LF>
040024  047105  042524  020122  MESG8:  .ASCIZ  /ENTER 18 BIT U.S.R. BAD SECTOR ADDRESS/<CR><LF>
040075     061  020066  044502  MESG9:  .ASCIZ  /16 BIT M.F.G. BAD SECTOR FILE/<CR><LF>
040135     061  020070  044502  MESG10: .ASCIZ  /18 BIT M F.G. BAD SECTOR FILE/<CR><LF>
040175     061  020066  044502  MESG11: .ASCIZ  /16 BIT U.S.R. BAD SECTOR FILE/<CR><LF>
040235     061  020070  044502  MESG12: .ASCIZ  /18 BIT U.S.R. BAD SECTOR FILE/<CR><LF>
040275     105  052116  051105  MESG13: .ASCIZ  /ENTER SECTOR NUMBER : /
040324  005015  047504  042516  MESG14: .ASCIZ   <CR><LF>/DONE/
040333     105  052116  051105  MESG15: .ASCIZ  /ENTER M.F.G. BAD SECTOR ADDRESS/<CR><LF>
                                ;MESG16:        .ASCIZ  /ILLEGAL BAD SECTOR,PACK NOT ACCEPTABLE/<CR><LF>
040375     127  044522  042524  MESG17: .ASCII  /WRITE BAD SECTOR FILE NOT SUCCESSFUL/<CR><LF>
040443     120  041501  020113          .ASCIZ  /PACK IS NOT ACCEPTABLE/<CR><LF>

                                .EVEN

(1)                             ;;*****************************************************************

                                .SBTTL  ERROR MESSAGES

(1)                             ;;*****************************************************************

040474  044122  030461  044440  EM1:    .ASCIZ  /RH11 INTERRUPT OCCURRED (RMAS=0)/
```

```
040535     125  042516  050130  EM2:    .ASCIZ  /UNEXPECTED ATTENTION OCCURRED/
040573     115  051501  041123  EM3:    .ASCIZ  /MASSBUS PARITY ERROR (MCPE=1)/
040631     115  051501  041123  EM4:    .ASCIZ  /MASSBUS PARITY ERROR (PAR=1)/
040666  042101  051104  051505  EM5:    .ASCIZ  /ADDRESS PLUG CHANGE BIT SET/
040722  044122  030461  042040  EM6:    .ASCIZ  /RH11 DIDN'T RESPOND TO ADDRESSING/
040764  051104  053111  020105  EM10:   .ASCIZ  /DRIVE OFFLINE/
041002  042520  051522  051511  EM11:   .ASCIZ  /PERSISTENT DRIVE UNSAFE ERROR/
041040  047125  047503  051122  EM12:   .ASCIZ  /UNCORRECTABLE MASSBUS PARITY ERROR/
041103     123  043117  053524  EM13:   .ASCIZ  /SOFTWARE TIMEOUT/
041124  051104  053111  020105  EM14:   .ASCIZ  /DRIVE UNSAFE ERROR/
041147     103  047117  051124  EM15:   .ASCIZ  @CONTROLLER/DRIVE ERROR DURING WRITE@
041213     103  047117  051124  EM16:   .ASCIZ  @CONTROLLER/DRIVE ERROR DURING WRITE CHECK@
041265     122  052105  044522  EM17:   .ASCIZ  @RETRIES NOT SUCESSFUL - SECTOR NOT ACCEPTABLE@
041343     104  052101  020101  EM20:   .ASCIZ  @DATA ERROR DURING WRITE CHECK@
041401     103  047117  051124  EM21:   .ASCIZ  @CONTROLLER/DRIVE ERROR VERIFYING HEADERS@
041452  042510  042101  051105  EM22:   .ASCIZ  @HEADER COMPARE ERROR VERIFYING HEADERS@
041521     103  046131  047111  EM23:   .ASCIZ  @CYLINDER FIELD IN HEADER IS NOT CORRECT@
041571     127  044522  042524  EM24:   .ASCIZ  @WRITE CHECK ERROR@
041613     111  046114  043505  EM25:   .ASCIZ /ILLEGAL BAD SECTOR,PACK NOT ACCEPTABLE/

(1)                             ;;*****************************************************.********

041662  046522  051501  051011  DH1:    .ASCIZ  /RMAS   RMCS1   RMER1   RMER2   RMDS    RMDC    RMDA/
041730  051104  053111  020105  DH2:    .ASCIZ  /DRIVE   RMDS   RMER1   RMER2   RMAS    RMCS1/
042002  051104  053111  020105  DH3:    .ASCIZ  /DRIVE   REG ADR  DATA  RMCS1   RMER1   RMER2/
042053     104  044522  042526  DH4:    .ASCIZ  /DRIVE   REG ADR    GOOD    BAD   RMCS1 RMER1   RMER2/
042136  044122  040440  042104  DH6:    .ASCIZ  /RH ADDRESS/
042151     104  044522  042526  DH10:   .ASCIZ  /DRIVE   ERR PC  RMCS1   RMCS2   RMDS    RMER1   RMER2/
042236  046522  041505  020061  DH10A:  .ASCIZ  /RMEC1   RMEC2   RMWC    RMBA    RMDA    RMAS    RMLA/
042323     122  042115  004502  DH10B:  .ASCIZ  /RMDB   RMMR1   RMDT    RMSN    RMOF  CYLINDER  TRACK/
042400  051104  053111  020105  DH17:   .ASCIZ  /DRIVE   ERR PC CYLINDER TRACK   SECTOR/
042447     104  044522  042526  DH20:   .ASCIZ  /DRIVE   ERR PC CYLINDER TRACK   SECTOR/
042516  046522  051503  020061  DH20A:  .ASCIZ  /RMCS1   RMCS2   RMDS    RMER1   RMER2   RMMR2   RMEC1   RMEC2/
042613     122  053515  020103  DH20B:  .ASCIZ  /RMWC    RMBA    RMDA    RMAS    RMLA    RMDB    RMMR1   RMDT/
042711     122  051515  004516  DH20C:  .ASCIZ  /RMSN   RMOF  CYLINDER  TRACK/
042744  020040  020040  020040  DH23:   .ASCII  /                       EXPT'D/<CR><LF>
042774  051104  053111  020105          .ASCIZ  /DRIVE   ERR PC  CYLNDR  ACTUAL HEADER/
043042  020040  020040  020040  DH24:   .ASCII  /                                       MEMORY  DISK/<CR><LF>
043130  051104  053111  020105          .ASCIZ  /DRIVE   ERR PC  CYLNDR  TRACK   SECTOR  DATA    DATA/
043215     104  044522  042526  DH25:   .ASCIZ /DRIVE   CYLNDR  TRACK   SECTOR  /
                                .LIST   BEX
6263
6264
6265  043256  001374  005540  005554  DT1:    .WORD   ATTN,RM.REG,RM.REG+14,RM.REG+42,RM.REG+12,RM.REG+34,RM.REG+6
6266  043264  005602  005552  005574
6267  043272  005546
6268  043274  001372  026476  026500  DT2:    .WORD   DDRIVE,RMERRS,RMERRS+2,RMERRS+4,ATTN,RMCS1
6269  043302  026502  001374  000000
6270  043310  001372  033514  033516  DT3:    .WORD   DDRIVE,RD.ADR,RD.WRD,RM.REG,RM.REG+14,RM.REG+42
6271  043316  005540  005554  005602
6272  043324  001372  033740  033736  DT4:    .WORD   DDRIVE,WRT.AD,WRT.WD,RD.WRD,RM.REG,RM.REG+14,RM.REG+42
6273  043332  033516  005540  005554
6274  043340  005602
6275  043342  001272                  DT6:    .WORD   $RMADR
6276  043344  001220  001132  005540  DT10:   .WORD   DRIVE,$ERRPC,RM.REG,RM.REG+10,RM.REG+12,RM.REG+14,RM.REG+42
6277  043352  005550  005552  005554
```

```
6278  043360  005602
6279  043362  005604  005606  005542          .WORD   RM.REG+44,RM.REG+46,RM.REG+2,RM.REG+4,RM.REG+6,RM.REG+16,RM.REG+20
6280  043370  005544  005546  005556
6281  043376  005560
6282  043400  005562  005564  005566          .WORD   RM.REG+22,RM.REG+24,RM.REG+26,RM.REG+30,RM.REG+32,DS.CYL,DS.TRK
6283  043406  005570  005572  001366
6284  043414  001370
6285  043416  001220  001132  001366  DT17:   .WORD   DRIVE,$ERRPC,DS.CYL,DS.TRK,SAVSEC
6286  043424  001370  001350
6287  043430  001220  001132  001366  DT20:   .WORD   DRIVE,$ERRPC,DS.CYL,DS.TRK,SAVSEC
6288  043436  001370  001350
6289  043442  005540  005550  005552          .WORD   RM.REG,RM.REG+10,RM.REG+12,RM.REG+14,RM.REG+42,RM.REG+40,RM.REG+44,RM.RE
6290  043450  005554  005602  005600
6291  043456  005604  005606
6292  043462  005542  005544  005546          .WORD   RM.REG+2,RM.REG+4,RM.REG+6,RM.REG+16,RM.REG+20,RM.REG+22,RM.REG+24,RM.RE
6293  043470  005556  005560  005562
6294  043476  005564  005566
6295  043502  005570  005572  001366          .WORD   RM.REG+30,RM.REG+32,DS.CYL,DS.TRK
6296  043510  001370
6297  043512  001220  001132  043746  DT23:   .WORD   DRIVE,$ERRPC,BUFP,RBUF,RBUF+2
6298  043520  043736  043740
6299  043524  001220  001132  001366  DT24:   .WORD   DRIVE,$ERRPC,DS.CYL,DS TRK,SAVSEC,$GDDAT,RM.REG+22
6300  043532  001370  001350  001140
6301  043540  005562
6302  043542  005540  005550  005552          .WORD   RM.REG,RM.REG+10,RM.REG+12,RM.REG+14,RM.REG+42,RM.REG+40,RM.REG+44,RM.RE
6303  043550  005554  005602  005600
6304  043556  005604  005606
6305  043562  005542  005544  005546          .WORD   RM.REG+2,RM.REG+4,RM.REG+6,RM.REG+16,RM.REG+20,RM.REG+22,RM.REG+24,RM.RE
6306  043570  005556  005560  005562
6307  043576  005564  005566
6308  043602  005570  005572  001366          .WORD   RM.REG+30,RM.REG+32,DS.CYL,DS.TRK
6309  043610  001370
6310
6311  043612  001220  005622  001370  DT25:   .WORD   DRIVE,FMTDPB+12,DS.TRK,SAVSEC
6312  043620  001350
6313  043622  000001                  DF1:    .WORD   1
6314  043624      007       000               .BYTE   7,0
6315  043626  000001                  DF2:    .WORD   1
6316  043630      006       000               .BYTE   6,0
6317  043632  000001                  DF3:    .WORD   1
6318  043634      006       000               .BYTE   6,0
6319  043636  000001                  DF4:    .WORD   1
6320  043640      007       000               .BYTE   7,0
6321  043642  000001                  DF6:    .WORD   1
6322  043644      001       000               .BYTE   1,0
6323  043646  000003                  DF10:   .WORD   3
6324  043650      007       000               .BYTE   7,0
6325  043652  042236                        .WORD   DH10A
6326  043654      007       000               .BYTE   7,0
6327  043656  042323                        .WORD   DH10B
6328  043660      007       140               .BYTE   7,140
6329  043662  000001                  DF17:   .WORD   1
6330  043664      005       034               .BYTE   5,34
6331  043666  000004                  DF20:   .WORD   4
6332  043670      005       034               .BYTE   5,34
6333  043672  042516                        .WORD   DH20A
```

```
6334  043674  010     000             .BYTE   8.,0
6335  043676  042613                  .WORD   DH20B
6336  043700  010     000             .BYTE   8.,0
6337  043702  042711                  .WORD   DH20C
6338  043704  004     014             .BYTE   4,14
6339  043706  000001          DF23:   .WORD   1
6340  043710  005     000             .BYTE   5,0
6341  043712  000004          DF24:   .WORD   4
6342  043714  007     034             .BYTE   7,34
6343  043716  042516                  .WORD   DH20A
6344  043720  010     000             .BYTE   8.,0
6345  043722  042613                  .WORD   DH20B
6346  043724  010     000             .BYTE   8.,0
6347  043726  042711                  .WORD   DH20C
6348  043730  004     014             .BYTE   4,14
6349
6350  043732  000001          DF25:   .WORD   1
6351  043734  004     000             .BYTE   4,0
6352                          ;;*****************************************************************
6353
6354                          ;BUFFER STARTS HERE
6355
6356                          ;;*****************************************************************
6357
6358  043736  000000  000000  000000  RBUF:   .WORD   0,0,0,0          ;BUFFER FOR HEADER CHECK
6359  043744  000000
6360
6361  043746                  BUFP:                   ;FORMAT AND CHECK BUFFER STARTS HERE
6362                                                  ;AND WILL OVERLAY ALL REMAINING CODE IN PROGRAM
6363
6364                          .NLIST  BEX
      043746  005015  041412  051132  TITLE:  .ASCII  <CR><LF><LF>/CZRMACO/<CR><LF>
      043762  046522  031460  051057          .ASCII  @RM03/RM02 FORMATTER @<CR><LF><LF>
      044011     120  047522  051107          .ASCIZ  /PROGRAM NEEDS 20 K MEMORY/<CR><LF><LF>
      044046  005015  047524  023440  LOADRV: .ASCII  <CR><LF>/TO 'FORMAT' OR 'CHECK' DRIVE 0, REPLACE THE 'XXDP'/<CR><LF>
      044134  040520  045503  047440          .ASCII  /PACK ON DRIVE 0 WITH ANOTHER PACK, CLEAR MEMORY LOCATION 40,//<CR><LF>
      044232  047101  020104  042522          .ASCIZ  /AND RESTART THE PROGRAM//<CR><LF>
                              BEX             .LIST
6365
6366
6367                          .SBTTL  BUSADR - GET BUS ADDRESS AND VECTOR ADDRESS FOR RH11
6368                          ;THIS ROUTINE IS USED TO INSURE THE BUS ADDRESS
6369                          ;OF THE RH11 IS SETUP FOR THE PROPER ADDRESS.
6370                          ;IT WILL ALSO READ THE ADDRESS FROM THE TTY IF
6371                          ;REQUIRED.
6372                          ;NOTE: THIS ROUTINE DESTROYS R0-R4
6373                          ;CALL
6374                          ;
6375                          ;       JSR     PC,BUSADR
6376                          ;       RETURN
6377
6378  044264  005737  001400  BUSADR: TST     CHGADR          ;INPUT FROM TTY REQUESTED?
6379  044270  001434                  BEQ     3$              ;NO--BRANCH
6380  044272  005037  001400          CLR     CHGADR          ;YES--CLEAR THE REQUEST FLAG
6381  044276  012700  001272  1$:     MOV     #$RMADR,R0      ;FIRST ADDRESS
6382  044302  104401  044400          TYPE    ,MRMCS1         ;"RMCS1="
```

```
6383  044306  011046                          MOV     (R0),-(SP)      ;PRESENT RMCS1 ADDRESS
6384  044310  104402                          TYPOC                   ;TYPE IT
6385  044312  104401  035166                  TYPE    ,LINSP          ;2 SPACES
6386  044316  104411                          RDLIN                   ;GET THE ENTRY
6387  044320  012601                          MOV     (SP)+,R1        ;ADDRESS OF ASCII TEXT
6388  044322  004537  044422                  JSR     R5,CK.NUM       ;ENTER AND STORE NEW ADDRESS
6389  044326  000763                          BR      1$              ;ERROR RET
6390  044330  012700  001274          2$:     MOV     #$RMVEC,R0      ;CHECK VERC
6391  044334  104401  044411                  TYPE    ,MRHVEC         ;"RHVEC="
6392  044340  011046                          MOV     (R0),-(SP)      ;PRESENT RH11 VECTOR ADDRESS ON THE STACK
6393  044342  104402                          TYPOC                   ;TYPE IT
6394  044344  104401  035166                  TYPE    ,LINSP          ;2 SPACES
6395  044350  104411                          RDLIN                   ;READ THE ENTRY
6396  044352  012601                          MOV     (SP)+,R1        ;ASCII TEXT ADDRESS
6397  044354  004537  044422                  JSR     R5,CK.NUM       ;ENTER AND STORE NEW ADDRESS
6398  044360  000763                          BR      2$              ;ERROR RE
6399  044362  012700  001272          3$:     MOV     #$RMADR,R0      ;FIRST ADDRESS OF NEW PARAMETERS
6400  044366  012701  026644                  MOV     #RMADR,R1       ;FIRST ADDRESS OF WHERE TO PUT THEM
6401  044372  012021                          MOV     (R0)+,(R1)+     ;BUS ADDRESS
6402  044374  012021                          MOV     (R0)+,(R1)+     ;VECTOR ADDRESS
6403  044376  000207                          RTS     PC              ;RETURN
6404
6405  044400  046522  051503  020061  MRMCS1: .ASCIZ  @RMCS1 = @
6406  044406  020075     000
6407  044411     122  053110  041505  MRHVEC: .ASCIZ  @RHVEC = @
6408  044416  036440  000040
6409
6410                                  .SBTTL  CK.NUM - CHECK NUMBER (OCTAL)
6411                                  ;THIS ROUTINE CHECKS AN ASCIZ STRING FOR LEGAL CHARACTERS
6412                                  ;AND FORMS AN OCTAL NUMBER IN R2
6413                                  ;CALL:
6414                                  ;       MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
6415                                  ;       R0=ADDRESS TO STORE THE NUMBER
6416                                  ;       JSR     R5,CK.NUM
6417                                  ;       RET1    ERROR RETURN
6418                                  ;       RET2    NORMAL RET
6419
6420
6421
6422  044422  010246          CK.NUM: MOV     R2,-(SP)        ;SAVE R2
6423  044424  010346                  MOV     R3,-(SP)                ;SAVE R3
6424  044426  010446                  MOV     R4,-(SP)        ;SAVE R4
6425  044430  012703  000006          MOV     #6,R3           ;MAX OCTAL DIGITS IN THE NUMBER
6426  044434  005002                  CLR     R2              ;FINAL OCTAL VALUE
6427  044436  112104          1$:     MOVB    (R1)+,R4        ;GET CURRENT POINTED BYTE
6428  044440  001424                  BEQ     3$              ;BRANCH IF TERMINATOR DETECTED
6429  044442  120427  000060          CMPB    R4,#'0          ;SMALLER THAN ASCII-0 ?
6430  044446  103425                  BLO     5$              ;YES, ERROR EXIT
6431  044450  120427  000067          CMPB    R4,#'7          ;LARGER THAN ASCII-7 ?
6432  044454  101022                  BHI     5$              ;YES, ERROR EXIT
6433  044456  006302                  ASL     R2              ;SHIFT LEFT
6434  044460  103420                  BCS     5$              ;
6435  044462  006302                  ASL     R2              ;ONE
6436  044464  103416                  BCS     5$              ;
6437  044466  006302                  ASL     R2              ;OCTAL DIGIT
6438  044470  103414                  BCS     5$              ;ERROR IF CARRY BIT SET
```

```
6439  044472  042704  177770          BIC     #177770,R4      ;CHOP OFF HIGHER BITS
6440  044476  060402                  ADD     R4,R2           ;APPENDING CURRENT DIGIT
6441  044500  005303                  DEC     R3              ;DECREMENT BYTE COUNT
6442  044502  001401                  BEQ     2$              ;BRANCH IF LAST DIGIT
6443  044504  000754                  BR      1$              ;LOOPING BACK
6444  044506  112104          2$:     MOVB    (R1)+,R4        ;CHECK TERMINATOR
6445  044510  001004                  BNE     5$              ;BRANCH IF NOT FOUND
6446  044512  005702          3$:     TST     R2              ;FINAL VALUE = 0 ?
6447  044514  001401                  BEQ     4$              ;YES
6448  044516  010210                  MOV     R2,(R0)         ;REPLACE THE ORIGINAL VALUE
6449  044520  005725          4$:     TST     (R5)+           ;ADJUST FOR NORMAL RET
6450  044522  012604          5$:     MOV     (SP)+,R4        ;RESTORE R4
6451  044524  012603                  MOV     (SP)+,R3        ;RESTORE R3
6452  044526  012602                  MOV     (SP)+,R2        ;RESTORE R2
6453  044530  000205                  RTS     R5              ;EXIT
6454
6455
6456
6457
6458
6459
6460          000001          .END
```

```
ABASE = 176700           1101#   1412    1453
ACDW1 = 000000           1412    1455
ACDW2 = 000000           1412    1456
ACK   = 000123           1349#
ACPUOP= 000000           1412    1427
ACTDRV  026562           4946#   5219*   5270*   5596*   5605*   5851
ACTSTR  026563           4952#   5853*   5867*
ADDRIS  036372           3309    6262#
ADDW0 = 000000           1412
ADDW1 = 000000           1412
ADDW10= 000000           1412
ADDW11= 000000           1412
ADDW12= 000000           1412
ADDW13= 000000           1412
ADDW14= 000000           1412
ADDW15= 000000           1412
ADDW2 = 000000           1412
ADDW3 = 000000           1412
ADDW4 = 000000           1412
ADDW5 = 000000           1412
ADDW6 = 000000           1412
ADDW7 = 000000           1412
ADDW8 = 000000           1412
ADDW9 = 000000           1412
ADEVCT= 000000           1412    1418
ADEVM = 000000           1412    1454
ADRTBL  006132           1628#   2741    2812    3008    3740
AENV  = 000000           1412    1423
AENVM = 000000           1412    1424
AFATAL= 000000           1412    1415
AMADR1= 000000           1412    1440
AMADR2= 000000           1412    1444
AMADR3= 000000           1412    1447
AMADR4= 000000           1412    1450
AMAMS1= 000000           1412    1434
AMAMS2= 000000           1412    1442
AMAMS3= 000000           1412    1445
AMAMS4= 000000           1412    1448
AMSGAD= 000000           1412    1420
AMSGLG= 000000           1412    1421
AMSGTY= 000000           1412    1414
AMTYP1= 000000           1412    1435
AMTYP2= 000000           1412    1443
AMTYP3= 000000           1412    1446
AMTYP4= 000000           1412    1449
AOE   = 001000           1251#
APASS = 000000           1412    1417
APRIOR= 000000           1412
APT     007726           2066#
APTCSU= 000040           4086    4193#
APTENV= 000001           3967    4079    4149    4191#
APTSIZ= 000200           1970    4190#
APTSPD= 000100           4081    4151    4192#
ASWREG= 000000           1412    1425
ATA   = 100000           1238#
ATABIT  026632           2049    5018#   5186    5282    5388    5770    5792    5800    5801    5821    5911
```

```
ATESTN= 000000  1412    1416
ATTN    001374  1497#   3944*   6265    6268
ATO     = 000001  1271#
AT1     = 000002  1272#
AT2     = 000004  1273#
AT3     = 000010  1274#
AT4     = 000020  1275#
AT5     = 000040  1276#
AT6     = 000100  1277#
AT7     = 000200  1278#
AUNIT   = 000000  1412    1419
ALSWR   = 000000  1412    1426
AVECT1= 000254  1102#   1412    1451
AVECT2= 000000  1412    1452
A16     = 000400  1175#
A17     = 001000  1176#
BADENT  003713  3353    6262#
BAD16   001414  1510#   2360    2381    2423*   2424*   2433    2434*   2435*   2436*   2437*   2438*   2439*   2440*
                2441*   2443*   2445*   2447*   2448*   2449*   2450*   2451*   2452*   2455*   2461    2463    2465
                2466    2467    2468    2469    2470    3473*   2588    3458    3471    3577    3580    3645    3690*
                3691*   3692    3777    3791
BAD18   002420  1513#   2359    2387    2465*   2470*   2474*   2591    3474    3594    3597    3654    3697*   3698*
                3699    3787    3793
BAI     = 000010  1193#
BEGCYL  001322  1473#   1584    2077*   2167*   2223*   2233    2505    2514    2518    2890    2894    2941    2944*
                2947*   3137    3166    3168*   3175*   3197    3444    3450*   3456*
BEGIN   006776  1138    1928#
BEGIN1  007014  1137    1931#   1977
BEGIN2  007034  1930    1936#   4634
BEGTRK  001326  1475#   1585    2076*   2166*   2222*   2237    2508    2521    2942    2943*   2946*   3136    3139
                3167    3169*   3174*   3199    3445    3451*   3455*
BIT0    = 000001  1086#
BIT00   = 000001  1076#   1086    2052    2219
BIT01   = 000002  1075#   1085    2055    2219    3122    5262    5517
BIT02   = 000004  1074#   1084
BIT03   = 000010  1073#   1083    5728    6029
BIT04   = 000020  1072#   1082    3103    5763
BIT05   = 000040  1071#   1081    5108    5538    5743
BIT06   = 000100  1070#   1080    5195    5253    5327    5636    6090
BIT07   = 000200  1069#   1079    5195    5462    5617    5728    5743    5763    5796    6062
BIT08   = 000400  1068#   1078    3100    5195
BIT09   = 001000  1067#   1077    3100    3978    5902
BIT1    = 000002  1085#   2850    3091
BIT10   = 002000  1066#   3097    3122    3955    5519
BIT11   = 004000  1065#   3097    5155    5289    5483    5837
BIT12   = 010000  1064#   2334    3094    3122    5150    5177    5195    5264    5297    5477    5500    5515    5529
                5753    5755    5967    6024    6091
BIT13   = 020000  1063#   3091    3122    3962    5247    5699    5970
BIT14   = 040000  1062#   2332    2855    2859    2861    2921    3091    3122    5259    5294    5686    5832    5932
                5979    5983    6088
BIT15   = 100000  1061#   2332    2434    2452    2676    2691    2694    2712    2725    2800    2884    2915    2921
                5247    5259    5262    5264    5294    5297    5483    5517    5519    5636    5728    5743    5753
                5763    5832    5902    5932    5939
BIT2    = 000004  1084#   3091    5939
BIT3    = 000010  1083#
BIT4    = 000020  1082#
```

```
BIT5  = 000040  1081#
BIT6  = 000100  1080#
BIT7  = 000200  1079#
BIT8  = 000400  1078#
BIT9  = 001000  1077#
BPTVEC= 000014  1093#
BUFP    043746  1628  1629  1630  1631  1632  1633  1634  1635  1636  1637  1638  1639  1640
                1641  1642  1643  1644  1645  1646  1647  1648  1649  1650  1651  1652  1653
                1654  1655  1656  1657  1658  1659  2293  2332  2334  2336  2338  2340  2342
                2344  2346  2349  2380  2386  2392  2395  2548  2662  2922  2979  3134  3196
                3215  3234  3437  3459  3472  3475  3478  3481  3484  3487  6297  6361#
BUSADR  044264  1999  6378#
BYTE16  00063?  1695#  3770
BYTE18  006432  1727#  3780
C       035156  3310  6262#
CALIN   012366  2481#
CHGADR  001400  1500#  1928*  1931*  6378  6380*
CI1     030140  5317  5345#
CI3     030246  5319  5368#
CI4     030346  5308  5390#
CI5     030724  5367  5389  5467#  5476
CI6     030746  5406  5416  5418  5423  5472#
CI7     030762  5249  5326  5357  5361  5365  5373  5383  5387  5398  5405  5411  5415  5429
                5435  5439  5449  5460  5475  5477#  5591  5621  5792  6074
CI7B    031004  5485#  5786
CI8     031062  5488  5499#  5924
CKADRS  012470  2484  2486  2505#  2886  3058  3061
CKSWR = 104407  3942  3977  4831#
CKTRK   014122  2769#
CK.CHR  022306  2182  2254  3848#  3855  3899
CK.DEC  022260  3826#  3854
CK.DIG  022360  3344  3732  3879#
CK.NUM  044422  6388  6397  6422#
CK.OCT  022232  3808#  3856
CLOCK   017226  3248  3258  3285#
CLR   = 000040  1195#
C.RGUC  034524  5090  5537  6158#
CNTLC   001376  1499#  1977*  2176*  3308*  4383  4466
CR    = 000015  1001#  1459#  4125  4135  6262  6364
CRLF  = 000200  1002#  4096  4135
CYLCK   001344  1482#  2890*  2926  2929*  2930  2934*  2935  2937*
DCK   = 100000  1257#  2793
DDISP = 177570  1008#  1390  1958
DDRIVE  001372  1496#  3943*  6268  6270  6272
DEVMP   001404  1503#  2049*  2071*  2072*  2073  2103  2111*  3023  3049
DF1     043622  1780  6313#
DF10    043646  1829  1836  1843  1850  1857  1864  1871  1892  1899  6323#
DF17    043662  1878  6329#
DF2     043626  1787  6315#
DF20    043666  1385  6331#
DF23    043706  1906  6339#
DF24    043712  1913  6341#
DF25    043732  1920  6350#
DF3     043632  1794  6317#
DF4     043636  1801  6319#
DF8     043642  1815  6321#
```

```
DH1     041662          1778    6262#
DH10    042151          1827    1834    1841    1848    1855    1862    1869    1890    1897    6262#
DH10A   042236          6262#   6325
DH10B   042323          6262#   6327
DH17    042400          1876    6262#
DH2     041730          1785    6262#
DH20    042447          1883    6262#
DH20A   042516          6262#   6333    6343
DH20B   042613          6262#   6335    6345
DH20C   042711          6262#   6337    6347
DH23    042744          1904    6262#
DH24    043042          1911    6262#
DH25    043215          1918    6262#
DH3     042002          1792    6262#
DH4     042053          1799    6262#
DH6     042136          1813    6262#
DISPLA  001156          1390#   1958*   1966*   3954*
DISPRE  000174          1123#   1966
DLT   = 100000          1205#
DMD   = 000001          1261#
DONE    015756          3046    3047#
DPINT   026536          4906#   5119    5122    5203*   5229    5670    5695    5822    5824*   5893    5913    5918    5926*
DPR   = 000400          1231#
DPRQS   026546          4919#   5235    5283*   5321*   5490*   5507    5523*   5673    5895    5915    5920    5935*
DRIVE = 001220          1468#   2110*   2194*   2215    2217    2280    2291    3131    3519    6276    6285    6287    6297
                        6299    6311
DRQ   = 004000          1294#
DRVACT  026506          4864#   5243    5423*   5470*   5492*   5504    5522*   5610*   5691    5726    5757*   5764*   5779
                        5791*   5905*
DRVCLR= 000111          1344#
DRVINT  027074          5110    5144#   5231    5825
DRVQUE  034622          5238    5251    6198#
DRVSTA  026516          2034    2112    2204    4878#   5100*   5101*   5102*   5103*   5113*   5145*   5154*   5192*   5197*
                        5225    5233    5257    5291    5295    5424*   5541*   5676    5684    5697    5751*   5758*   5771
                        5827    5927*
DRVTYP  026526          2037    2052    2055    2209    2219    4893#   5146*   5161*   5166*   5171*   5260    5542*
DRY   = 000200          1230#
DSWR  = 177570          1007#   1389    1957
DS.CYL  001366          1494#   1590    1595    1599    2500*   3547    3557    3565*   3570    3582*   3587    3610*   3615
                        3627*   3632    3754*   3759    3945*   6282    6285    6287    6295    6299    6308
DS.TRK  001370          1495#   1591    1596    1600    2501*   3071*   3549    3558    3566*   3572    3583*   3589    3611*
                        3617    3628*   3634    3755*   3761    3946*   6282    6285    6287    6295    6299    6303    6311
DTE   = 010000          1254#   2793
DTSY  = 000200          1267#
DTUW    026630          5012#   5096    5310    5366*   5497*   5510    5524    5526*   5535*   5599    5611*   5898    5908*
                        5977
DT00  = 000001          1285#
DT01  = 000002          1286#
DT02  = 000004          1287#
DT03  = 000010          1288#
DT04  = 000020          1289#
DT05  = 000040          1290#
DT06  = 000100          1291#
DT07  = 000200          1292#
DT08  = 000400          1293#
DT1     043256          1779    6265#
```

```
DT10     043344            1828    1835    1842    1849    1856    1863    1870    1891    1898    6276#
DT17     043416            1877    6285#
DT2      043274            1786    6268#
DT20     043430            1884    6287#
DT23     043512            1905    6297#
DT24     043524            1912    6299#
DT25     043612            1919    6311#
DT3      043310            1793    6270#
DT4      043324            1800    6272#
DT6      043342            1814    6275#
DULP     027076            5145#   5156
DVA    = 004000            1225#
ECH    = 000100            1248#
ECI    = 004000            1316#
EDIT     001410            1505#   2068*   2528*   2531*   2961
EMPTYQ   034602            5489    5540    5637    5940    6183#
EMTVEC= 000030            1096#   1945*   1946*   1976*
EM1      040474            1777    6262#
EM10     040764            1826    6262#
EM11     041002            1833    6262#
EM12     041040            1840    6262#
EM13     041103            1847    6262#
EM14     041124            1854    6262#
EM15     041147            1861    6262#
EM16     041213            1868    6262#
EM17     041265            1875    6262#
EM2      040535            1784    6262#
EM20     041343            1882    6262#
EM21     041401            1889    6262#
EM22     041452            1896    6262#
EM23     041521            1903    6262#
EM24     041571            1910    6262#
EM25     041613            1917    6262#
EM3      040573            1791    6262#
EM4      040631            1798    6262#
EM5      040666            6262#
EM6      040722            1812    6262#
ENDCYL   001320            1472#   1586    2078*   2218*   2221*   2233    2505    2515    2517    2926    2935
ENDTRK   001324            1474#   1587    2079*   2169*   2237    2507    2522
ENTADR   035171            2228    6262#
ERINDX   016144            2316    2752    2784    2905    2988    3088#
ERR    = 040000            1237#   2804
ERRVEC= 000004            1089#   1955    1956*   1967*   3244*   3245*   3255*   3264*
ES.SAV   034766            5655    5707    5711    5975    6035    6253#
FER    = 000020            1246#   2793
FMTDPB   005610            1553#   2171*   2215    2217*   2291*   2292*   2293*   2294*   2295*   2296*   2297*   2299*   2300*
                           2302    2304    2306*   2310    2312    2330    2562*   2565*   2566*   2568    2570    2598    2600
                           2660*   2661*   2662*   2672*   2680    2693*   2695    2697    2717    2722*   2732    2737*   2739
                           2741*   2742*   2746    2748    2769*   2773    2775    2810*   2812*   2820    2825*   2834    2891*
                           2892*   2893*   2894*   2895*   2899    2901    2911    2932    2938*   2939*   2940*   2950*   2968*
                           2969*   2971    2973    2975*   2976*   2977*   2978*   2979*   2980*   2982    2984    3000    3005*
                           3008*   3009*   3010*   3071    3090    3093    3096    3099    3102    3121    3131*   3132*   3133*
                           3134*   3135*   3136*   3137*   3138*   3151*   3152    3154    3162*   3163*   3164    3171*   3172*
                           3173*   3184*   3185*   3186*   3311    3317    3372    3374    3376    3382    3383    3510    3512
                           3516    3524    3525    3945    3946    6311
FMTA     005630            1570#   3519*   3520*   3521*   3522*   3523*   3524*   3525*   3531    3533
```

```
FMT16 = 010000          1317#
FUNCT1  020652          3564#
FUNCT2  021044          1624    3599#
FUNCT3  021324          1624    3644#
FUNCT4  021412          1624    3663#
FUNCT5  021522          3688#
FUNCT6  021620          1624    3708#
FUNCT7  021716          1624    3728#
FUNCT8  022020          3753#
F1      = 000002        1220#
F2      = 000004        1221#
F3      = 000010        1222#
F4      = 000020        1223#
F5      = 000040        1224#
GETREG= 000141          1351#
GETREQ  034676          5284    5512    5628    5680    5717    5781    5831    5901    5929    5936    6219#
GNS     = ****** U      1122    4823    4824    4825    4826    4827    4829    4831    4832    4833    4834    4835
GO      = 000001        1219#
GTSWR = 104406          1993    4829#
HCE     = 000200        1249#   2793    2911
HCI     = 002000        1315#
HCRC    = 000400        1250#   2793
HDREAD  014664          2869    2876#
HEADX   001412          1506#   2800*   2855*   3518
HEDERR  001360          1488#   2887*
HSET    014564          2702    2850#
HT      = 000011         999#   4094    4135
IAE     = 002000        1252#
IE      = 000100        1173#
ILF     = 000001        1242#
ILR     = 000002        1243#
INSERT  020540          3542#   3576    3593    3621    3638    3776    3786
IOTVEC= 000020          1094#
IR      = 000100        1196#
ISR     031474          5105    5596#
LA      031320          5314    5557#
LACNT   026574          4971#   5565*   5566    5587*
LF      = 000012        1000#   1460#   4129    4135    6262    6364
LINSP   035166          2050    3314    3746    4031    4040    4051    6262#   6385    6394
LIN4SP  035164          2033    6262#
LOADRV  044046          1983    6364#
LOP.CK  016244          2323    2759    2839    2920    2925    31.7#
LST     = 002000        1233#
MADRER  035733          2239    6262#
MAXSEC  001364          1492#   2090*   2095*   2159*   2165*   2685    2697    2706    2729    2780    2813    2834    3194
                        3214    3233    3446    3448*   3454*
MCCMPT  036316          3018    6262#
MCLK    = 000002        1262#
MCPE    = 020000        1178#
MCPEMX  026642          5030#   5957    6005
MDRNP   035243          2211    6262#
MDRVD   035162          2190    6262#
MER11   035270          2213    6262#
MESG1   036462          2420    6262#
MESG10  040135          3653    6262#
MESG11  040175          3671    6262#
```

MESG12	040235	3678	6262#								
MESG13	040275	3728	6262#								
MESG14	040324	3706	3726	6262#							
MESG15	040333	3753	6262#								
MESG17	040375	3076	6262#								
MESG2	036667	3272	6262#								
MESG3	036566	3387	6262#								
MESG4	037050	3067	6262#								
MESG5	037631	3564	6262#								
MESG6	037702	3581	6262#								
MESG7	037753	3609	6262#								
MESG8	040024	3626	6262#								
MESG9	040075	3646	6262#								
MFCMPT	036271	3016	6262#								
MFORMT	035430	2137	6262#								
MHECK	035451	2135	6262#								
MINX =	000004	1263#									
MMODE	035376	2124	6262#								
MNDLTA	026656	5046#	5585								
MNRM03	035317	6262#									
MODE	001316	1471#	2081*	2084*	2096	2123#*	2140*	2246	2274	2535	2876	3014	3056
MOFFLN	035221	6262#									
MOH =	020000	1295#									
MOL =	010000	1235#									
MORMAT	035464	2139	6262#								
MPATD	036165	2261	6262#								
MPE =	000400	1198#									
MRD =	000020	1265#									
MRHVEC	044411	6391	6407#								
MRMCS1	044400	6382	6405#								
MSCHK	036235	2278	6262#								
MSEC20	035634	2161	6262#								
MSEC22	035555	2155	6262#								
MSELD	035364	6262#									
MSELP	036035	2251	6262#								
MSFOU	036200	2276	6262#								
MSGBLK	037111	2487	6262#								
MSIZE	035504	2144	6262#								
MSTCK =	000010	1264#									
MUNIT	035140	2179	6262#								
MUSDR	035344	2207	6262#								
MWC	001356	1487#	2089*	2093*	2157*	2163*	2661	3133			
MWR =	000040	1266#									
MXDLTA	026654	5043#	5583								
MXF =	001000	1199#									
MXLACT	026652	5040#	5566								
MXWNDW	026660	5049#									
M0	010522	2172	2176#	2193	2208	2212	2214	3054			
M1	010244	2067	2123#	2134							
M1A	010334	2136	2144#	2154							
M1B	010462	2160	2166#	3295							
M2	010772	2216	2224	2228#	2240						
M4	011052	2236	2238	2242	2246#	2267					
M4A	011076	2247	2251#								
M5	011160	2116	2250	2272#							
MSF/	007774	2074	2076#	3051							

```
NBA     = 100000            1297#
NED     = 010000            1202#
NEM     = 004000            1201#
NEWSVC  017260              2272    2558    3299#   3321
NEXT    005536              1524#   2657*   2669    2672    2690*   2711*   2713    2724*   2728*   2814    2816    2847
NOTPRS  035055              2042    6262#
NOTRM   035033              2040    6262#
NOTSAF  035072              2046    6262#
NUMERR  036342              3019    6262#
OENTER  017240              2176    3291#   3308
OFDIR   = 000001            1314#
OFFSET= 000115              1346#
OPI     = 020000            1255#   2793
OPT     007702              5245    5280#   5631    5805    5843
OR      = 000200            1197#
PACK    001406              1504#   2298*   2351*   2377    2414    2584    2611*
PAR     = 000010            1245#
PARENT  017416              2230    3330#   3568    3585    3613    3630    3757
PAR1    005776              1584    1605#
PAR2    006011              1585    1607#
PAR3    006024              1586    1609#
PAR4    006035              1587    1611#
PAR5    006046              1590    1595    1599    1614#
PAR6    006057              1591    1596    1600    1616#
PAR7    006070              1592    1597    1618#
PAR8    006101              1601    1620#
PAT     = 000020            1194#
PATA    001340              1480#   2098*   2248*   2537*   2541*   2545*   2549    2378    2732*   2884*   2885    3059*   3220
PATB    001342              1481#   2099*   2249*   2538*   2542*   2546*   2550    2385*   3060*   3221
PATSEL  001336              1479#   2100*   2170*   2268*   2539    2543
PGE     = 002000            1200#
PGM     = 001000            1232#
PIP     = 020000            1236#
PIRQ    = 177772            1006#
PIRQVE= 000240              1100#
PMSG    036417              4632    6262#
POPQUE  034720              5293    5345    5461    5766    5795    5835    6234#
PRO     = 000000            1023#
PR1     = 000040            1024#
PR2     = 000100            1025#
PR3     = 000140            1026#
PR4     = 000200            1027#
PR5     = 000240            1028#   1975    1976
PR6     = 000300            1029#   3250    3260
PR7     = 000340            1030#   2023
PS      =  77776            1003#   1004    2023*   2203*   5088    5089*   5118*   5126*   5217    5218*   5271*   5281    5330*
                            5572#
PSEL    = 002000            1177#
PSW     = 177776            1004#   2273*   2559*   3307*
PWRVEC= 000024              1095#   1949*   1950*   4601*   4602*   4611*   4617*   4629*   4630*
QCNT    034232              6100#   6159    6183*   6198    6200*   6220    6234*
QDRV0   034324              6111    6122    6131    6143#
QDRV1   034344              6112    6123    6132    6144#
QDRV2   034364              6113    6124    6133    6145#
QDRV3   034404              6114    6125    6134    6146#
QDRV4   034424              6115    6126    6135    6147#
```

```
QDRV5   034444            6116    6127    6136    6148#
QDRV5   034464            6117    6128    6137    6149#
QDRV7   034504            6118    6129    6138    6150#
QINPT   034242            6111#   6185    6202#   6203#   6204    6206#
QOUTPT  034262            6122#   6185#   6223    6236    6237#   6238#   6239    6241#
QSTART  034302            6131#   6165    6170    6206    6241
QSTOP   034304            6132#   6204    6239
QTERM = 034524            6139    6151#
RBUF    043736            1575    2892    2921#   2922    3510#   3516#   3517#   3518#   3522    3526#   3529#   6297    6358#
RDCHR = 104410            4542    4832#
RDDAT = 000171            1358#
RDHD  = 000173            1359#   2297    2306    2895
RDLIN = 104411            2125    2145    2180    2252    2425    2488    3273    3342    3729    4833#   6386    6395
RDY   = 000200            1174#
RD.ADR  033514            5959#   5960#   5962#   6270
RD.RM   033470            5157    5181    5187    5409    5433    5447    5559    5573    5619    5649    5720    5730    5745
                          5887    5957#   6012    6026    6032    6066
RD.RM1  033512            5961#   5991
RD.RM2  033636            5957#   5990#
RD.RM3  033642            5968    5982    5992#
RD.RM4  033646            5973    5994#
RD.WRD  033516            5963#   5964    6270    6272
READIN= 000121            1348#
RECAL = 000107            1343#
RECORD  017532            2799    3369#
REJCT2  016104            3070#   3514
REJCT3  016126            2999    3075#
REJEC1  016066            2348    3066#
RELSE = 000113            1345#
RESET   020354            2801    2863    3510#
RESREG= 104413            4053    4722    4835#   5127    5265    5268    5331    5544    5604    5866    6075    6174
RESTRT  017162            2510    3269#
RESVEC= 000010            1090#
RETBAD  011226            2290#
RETRY   001346            1483#   2290#   2324    2326#   2329#   2659#   2760    2762#   2765#   2840    2842#   2845#   2963#
                          2996#   2997    3011#
RMADR   026644            2000#   2010#   5035#   5107    5224    5348    5368    5390    5557    5598    5885    5959    5965
                          5969    5985    6018    6022    6055    6085    6400
RMAS  = 000016            5060#   5186#   5647    5770#   5800#   5821#   5910
RMBA  = 000004            3947    3949    5055#
RMCS1 = 000000            5053#   5149#   5155    5175    5288#   5289    5364    5386    5428    5448    5459    5474    5487
                          5620    5650    5704    5737    5776    5837    6013    6268
RMCS2 = 000010            2790    5057#   5108#   5148#   5150    5237#   5287#   5349#   5369#   5391#   5477    5500    5515
                          5529    5538#   5558#   5618#   5718#   5807#   5820#   5886#   5966    6023    6056#   6062    6086#
                          6091
RMDA  = 000006            2778    3002    5056#   5356    5382    5397
RMDB  = 000022            5062#   6060
RMDC  = 000034            5067#   5360    5372    5404
RMDS  = 000012            2804    5058#   5182    5560    5721    5746    5808    5888
RMDT  = 000026            5064#   5158
RMEC1 = 000044            5071#
RMEC2 = 000046            5072#   6070
RMERRS  026476            4856#   5091    5686    5689    5699    5808#   5809#   5810#   5811#   6268
RMER1 = 0000.4            2792    3124    5059#   5188    5731    5809    6027
RMER2 = 000042            5070#   5810
RMHR  = 000036            5068#
```

```
RMINIT  026662          2017    2200    5087#
RMLA    = 000020        5061#   5574    5576
RMMR1   = 000024        5063#
RMMR2   = 000040        5069#   5811
RMOF    = 000032        5066#   5179    5410    5414    5434    5438
RMR     = 000004        1244#
RMSN    = 000030        5065#
RMVEC   026646          2001#   2011#   5036#   5104    5106    5218
RMWC    = 000002        5054#
RM.REG  005540          1529#   1562    2778    2790    2792    2804    2913    2915    3002    3114    3947    3949    6265
                        6270    6272    6276    6279    6282    6289    6292    6295    6299    6302    6305    6308
RM03    027410          2301    2309    2567    2745    2772    2898    2970    2981    3530    5217#
RM03A   035114          2057    6262#
RM03B   035102          2051    6262#
RNOP    = 000101        1341#
RPTMR   033056          3286    5851#
RPOS    035107          2054    6262#
RTC     = 000117        1347#
SAVEFG  026604          2018#   2201#   4986#   5463    5624    5797
SAVREG= 104412          3993    4696    4834#   5087    5220    5280    5499    5597    5854    6052    6158
SAVSEC  001350          1484#   1592    1597    1601    2502#   2778#   2780#   2781#   2782#   2809    2810    2862#   3002#
                        3004#   3005    3006    3384    3517    3523    3551    3559    3567#   3574    3584#   3591    3612#
                        3619    3629#   3636    3756#   3763    3765    3768    3770    3775#   3778#   3780    3785#   6285
                        6287    6299    6311
SAVWC   001352          1485#   2824#   3184
SC      031710          5603    5632    5635    5641    5647#
SCAWC   016576          2836    3184#
SC01    = 000100        1301#
SC02    = 000200        1302#
SC04    = 000400        1303#
SC10    = 001000        1304#
SC11    032546          5688    5719    5727    5788#
SC12    032656          5682    5693    5807#
SC13    032720          5672    5675    5817#
SC20    = 002000        1305#
SC3     031760          5663#   5667
SC4     031764          5665#   5696    5709    5713    5762    5778    5787    5806    5844
SC5     031776          5664    5670#
SC6     032176          5692    5714#
SC6A    032312          5690    5739#
SC7     032440          5729    5742    5744    5754    5764#
SC8     032516          5705    5722    5732    5738    5747    5777    5779#   5816
SEARCH= 000131          1350#
SECCU   005434          1522#   2575    2594    2622    2623    2640    2642#   2651    2653    2673    2856
SEC30   001362          1489#   2085#   2094#   2158#   2164#   2393    2563    2589    3201    3447    3449#   3453#   3479
                        3485    3527    3603    3667
SEEK    = 000105        1342#
SEEKFG  026606          2202#   4994#   5092    5312
SELDRV= 000145          1353#
SERNL   012020          2414#
SETFMT= 000143          1352#   2299    2566    2969
SETHDR  016624          2561    2889    2945    3170    3194#   3452
SETPAT  012642          2510    2530    2535#
SETTBL  016312          2560    2888    3131#
SETVEC  007464          1998    2003    2016#
SET.IE  034160          5121    5152    5255    5329    5543    5657    6084#
```

```
SKI     = 040000          1327#
SLASH     035152          3341    6262#
SN1       036550          2421    6262#
SOFSW     001314          1470#   2658#   2796    2826#   2828    2830    2832#   2837#   2846#   3187#
SORT1     013242          2622#
SORT2     017646          2965    3394#   3579    3596    3624    3641    3790    3792
SPCHWT    026560          4940#   5282#   5388#   5792    5801#
STACK   = 001100           994#   1943    2308    2356    2744    2771    2897    3055    3291
STCLK1    017076          3244    3254#
STCLK2    017146          3255    3263#
STCLK3    017152          3253    3262    3264#
STKLMT= 177774            1005#
STO       033146          5861    5881#
STO1      033202          5893#
STO2      033276          5892    5894    5896    5910#
STO3      033326          5912    5918#
STO5      033352          5924#
STO6      033360          5914    5926#
STO7      033416          5916    5934#
STO8      033446          5933    5940#
STO9      033456          5890    5900    5909    5917    5921    5923    5925    5931    5938    5942#
ST.CLK    017020          2016    2199    3244#
SVRH70    034042          5465    5626    5638    5681    5734    5752    5799    5833    5903    5941    6056#
SWR       001154          1389#   1941    1957#   1959    1965#   1972#   1991    2021    2850    3052    3117    3955    3962
                          3974    3978    4386    4425    4480#   4609    4622#
SWREG     000176          1124#   1965    1991    4386    4425    4448
SW0     = 000001          1058#   3052
SW00    = 000001          1048#   1058
SW01    = 000002          1047#   1057
SW02    = 000004          1046#   1056    2021
SW03    = 000010          1045#   1055
SW04    = 000020          1044#   1054
SW05    = 000040          1043#   1053
SW06    = 000100          1042#   1052
SW07    = 000200          1041#   1051
SW08    = 000400          1040#   1050
SW09    = 001000          1039#   1049    3117
SW1     = 000002          1057#
SW10    = 002000          1038#
SW11    = 004000          1037#
SW12    = 010000          1036#
SW13    = 020000          1035#
SW14    = 040000          1034#
SW15    = 100000          1033#
SW2     = 000004          1056#
SW3     = 000010          1055#
SW4     = 000020          1054#
SW5     = 000040          1053#
SW6     = 000100          1052#
SW7     = 000200          1051#
SW8     = 000400          1050#
SW9     = 001000          1049#
SYSTAT    035121          2026    6262#
T         035160          3315    6262#
TABCAL    006120          1624#   2497
TABINT    011604          2356#   2612
```

```
TABLD   011666  2377#
TABLE   005650  1584#  2231
TABLE2  005702  1590#  3569  3614
TABLE3  005726  1595#  3586  3631
TABLE4  005752  1599#  3758
TAB.XY= 001114  1163#  1368
TAP     = 040000  1296#
TBITVE= 000014  1091#
TD      031540  5601  5610#
TEXT    020006  2967  3437#
TIMER   026610  4999#  5201#  5324#  5468#  5521#  5613#  5715#  5760#  5803#  5818#  5840#  5857  5859#
                5922#  5928#  5934#
TITLE   043746  1980  6364#
TKVEC   = 000060  1098#  2272#  2558#  3321#  4363#  4364#
TPVEC   = 000064  1099#
TRAPVF= 000034  1097#  1947#  1948#  1975#
TRE     = 040000  1179#
TRKCNT  001334  1478#  3139#  3148  3150#  3161#
TRKTST  016402  2868  3146#
TRK1    = 004000  1306#
TRK10   = 040000  1309#
TRK2    = 010000  1307#
TRK20   = 100000  1310#
TRK4    = 020000  1308#
TRNSWT  026556  3292  4932#  5346#  5493  5496#  5509  5527#  5536#  5615  5616#  5897  5907#
TRTVEC= 000014  1092#
TST1    007024  1934#
TTRKS   001330  1476#  2507#  2508#  2509#  2520#  2521#  2522#  2523#  2526#  2529  3140
TTRKSC  001332  1477#  3140#  3146#  3156#
TYPADR  017316  3307#
TYPDS   = 104405  3021  3340  4028  4827#
TYPE    = 104401  1980  1983  2025  2026  2033  2040  2042  2044  2046  2048  2050  2058  2060
                  2064  2124  2133  2135  2137  2139  2144  2153  2155  2161  2179  2190  2192
                  2207  2211  2213  2228  2239  2251  2261  2266  2276  2278  2285  2419  2420
                  2421  2487  3016  3018  3019  3022  3066  3067  3070  3075  3078  3272  3309
                  3310  3314  3315  3320  3335  3341  3353  3386  3387  3564  3581  3609  3626
                  3646  3647  3653  3655  3671  3672  3678  3679  3706  3726  3728  3742  3746
                  3753  3957  3965  4003  4004  4008  4009  4018  4031  4035  4040  4042  4048
                  4051  4059  4255  4331  4380  4392  4446  4447  4450  4461  4471  4482  4501
                  4549  4555  4560  4564  4569  4570  4572  4575  4579  4631  4658  4823#  6382
                  6385  6391  6394
TYPERR  022752  3964  3993#
TYPOC   = 104402  3650  3658  3675  3682  3745  4024  4449  4824#  6384  6393
TYPON   = 104404  4826#
TYPOS   = 104403  2030  2282  4825#
ULOFLG  026564  4959#  5147#  5227  5242#  5425#  5491#  5528#  5678  5767  5769#  5788  5790#  5906#
UNS     = 040000  1256#
UNTOFF  035012  2044  6262#
UNTON   035023  2048  6262#
UPDACY  016712  2949  3177  3214#
UPDATM  016770  3159  3233#
UPE     = 020000  1203#
USSAV   004400  1519#  2358  2396  2467#  2469#  2472#  3483  3601  3602  3607  3608  3666  3670
                3717#  3718#  3719
USTAB   003424  1516#  2361  2402  2466#  2468#  2475#  2587  2966  3369  3379  3477  3599  3600
                3605  3606  3665  3669  3710#  3711#  3712
```

```
US1     = 000001            1190#
US2     = 000002            1191#
US4     = 000004            1192#
VV      = 000100            1229#
WC        001354            1486#   2088*   2092*   2156*   2162*   2547
WCE     = 040000            1204#   2790
WCF     = 000040            1247#
WCKD    = 000151            1354#
WCKHD   = 000153            1355#   2769
WCTBL     006232            1663#   2716    2719    2731    2734    2819    2822    3010
WLE     = 004000            1253#   3124
WRAP      001402            1501#   2080*   2168*   2232*   2241*   2511
WRL     = 004000            1234#
WRTDAT  = 000161            1356#
WRTHD   = 000163            1357#   2742    2980    3521
WRTRK     012772            2560#
WRTRKA    013444            2669#   2849
WRTRKB    013552            2681    2693#   2696
WRTRKC    013636            2692    2713#
WRTRKD    013720            2675    2687    2704    2709    2728#
WRTRKE    013765            2723    2739#
WRTRKF    013702            2686    2708    2724#
WRTRKH    013732            2727    2730#
WRTRKX    013046            2575#   2852    2870
WRTRKY    013070            2584#
WRTRKZ    014652            2868#
WRTRK1    013414            2658#
WRTRK2    014002            2742#   2827
WRT.AD    033740            6008*   6018*   6021#   6272
WRT.RM    033650            5174    5178    5355    5359    5363    5371    5381    5385    5396    5403    5413    5427    5437
                            5458    5473    5486    5703    5736    5775    6005#
WRT.R1    033734            6019#   6040
WRT.R2    034026            6005*   6039#
WRT.R3    034032            6014    6025    6028    6034    6041#
WRT.R4    034036            6030    6043#
WRT.R5    034040            6042    6044#
WRT.WD    033736            6006*   6010    6017*   6020#   6272
$APTHD    001100            1150    1156#
$ASTAT=   ******  U         4171    4186
$ATYC     023510            4142    4144#
$ATY1     023464            4140#
$ATY3     023472            4084    4141#
$ATY4     023502            3970    4143#
$AUTOB    001150            1386#   1995*   2066    2416    2483    3047    3269    4439    4596
$BASE     001262            1453#   2007    2009
$BLHDR    001136            1381#
$BDDAT    001142            1383#
$BELL     001176            1400#   3957    3986    4392    4589
$CDW1     001266            1455#   1929*   1932*   2082    2485
$CDW2     001270            1456#   2086
$CHARC    023460            4101#   4111*   4118    4127*   4132#
$CKSWP    024710            4425#   4831
$CMTAG    001114            1369#   1938    1939    1951
$CM3    = 000000            1399#
$CNTLC    025613            4380    4461    4589#
$CNTLG    025625            4446    4591#
```

```
$CNTLU  025620  4471  4564  4590#
$CPUOP  001234  142?#
$CRLF   001203  1402#  2025  2060  2064  2285  2419  3022  3066  3070  3075  3320  3386  3647
                3655  3672  3679  3742  3965  3986  4003  4008  4018  4035  4048  4100  4135
                4482  4569  4589
$DBLK   024374  4297  433.  4339#
$DB2D   026122  4677  46?o#
$DECVL  026302  4698  4744#
$DEVCT  001216  1418#  2114*  3013*  3064*
$DEVM   001264  1454#  2069  2071
$DOAGN  015752  3033  3038  3044#
$DTBL   024364  4300  4335#
$ENDAD  015742  1132  3040#
$ENDCT  015726  3035#
$ENV    001226  1423#  1989  2002  3967  4079  4149  4173
$ENVM   00.227  1424#  1970  4081  4086  4151
$EOP    015324  2075  2117  2880  2928  2960#  3126
$EOPCT  015720  3032#  3036
$ERFLG  001117  1372#  3952*  3986
$ERMAX  001131  1378#  1952*
$ERROR  022516  1945  3941#
$ERRPC  001132  1379#  3959*  3960*  3961  3986  6276  6285  6287  6297  6299
$ERRTB  006526  1774#  4000
$ERTTL  001126  1376#  2177*  3020  3958*  3986
$ESCAP  001174  1399#  1951*  2315*  2328*  2751*  2764*  2783*  2798*  2803*  2844*  2904*  2918*  2987*
                3120*  3981  3983  3986  5655  5707  5711  5975  6035  6245  6254  6255*  6257*
$ETABL  001226  1422#
$ETEND  001272  1162  1457#
$FATAL  001210  1415#  4177*
$FFLG   023730  4140*  4143*  4171  4180*  4188#
$FILLC  001172  .397#  4104  4135
$FILLS  001171  1396#  413?
$GDADR  001134  1380#
$GDDAT  001140  1382#  3951*  6299
$GET42  015732  3037#
$GTSWR  025000  4447#  4829
$HD     =  000000  991
$HIBTC  001100  1157#
$ICNT   001120  1373#
$ILLUP  026020  4601  4617  4636#
$INTAG  001151  1387#  4444*  4463  4483  4596
$ITEMB  001130  1377#  3961*  3969  3986  3995
$LF     001204  1403#  3986  4135  4579  4589
$LFLG   023727  4181*  4187#
$LKCSB  001300  1464#  3251*
$LKCSR  001276  1463#  3246  3252*
$L?S    001306  1466#  3256  3261*
$LLVEC  00.310  1467#  3257
$LPADR  001122  1374#
$LPERR  001124  1375#  2307*  2743*  2770*  2896*  3119  3980
$LPVEC  001302  1465#  3247
$MADR1  001240  1440#
$MADR2  001244  1444#
$MADR3  001250  .447#
$MADR4  001254  1450#
$MAIL   001206  1156  1162  1413#  1935  1969  1989  3967  4079
```

```
$MAMS1  001236            1434#
$MAMS2  001242            1442#
$MAMS3  001246            1445#
$MAMS4  001252            1448#
$UIBADR 001102            1158#
$MFLG   023726            4141*   4147    4182*   4186#
$MNEW   025643            4450    4594#
$MSGAD  001222            1420#   4157*   4160
$MSGLG  001224            1421#   4162*
$MSGTY  001206            1414#   4155    4163*   4175    4179*
$MSWR   025632            4447    4592#
$MTYP1  001237            1435#
$MTYP2  001243            1443#
$MTYP3  001247            1446#
$MTYP4  001253            1449#
$NULL   001170            1395#   4106    4135
$NWTST= 000000            1933#
$OCNT   024154            4227*   4256*   4269#
$OMODE  024156            4222*   4226*   4231    4234*   4245*   4271#
$PASS   001214            1417#   1969*   3025*   3029*   3030*   3047    3062#
$PASTM  001106            1160#
$PWRAD  026014            4634#
$PWRDN  025654            1943    4601#   4629
$PWRMG  026010            1632#
$PWRUP  025726            4611    4617#
$QUES   001202            1401#   2133    2153    2192    2266    3986    4135    4501    4572    4589
$RDCHR  025252            4514#   4832
$RDDEC= ******  U         4834
$RDLIN  025342            4537#   4833
$RDOCT= ******  U         4834
$RDSZ = 000015            4530#
$RESRE  026354            4779#   4835
$RMADR  001272            1461#   2000    2009*   2010    6275    6381    6399
$RMVEC  001274            1462#   2001    2006*   2011    6390
$RTNAD  015754            3046#
$R2A  = ******  U         4836
$SAVRE  026316            4763#   4834
$SAVR6  026024            4610*   4618    4619*   4620*   4638#
$SB2D   026066            3312    3318    4675#
$SETUP= 000116            1928#   1944    1945    1947    1949    1951    1953    1986    3029    3942    3977    3985    4384
                          4385    4390    4420    4596
$STUP = 177777            1926#
$SUPPS  026026            3313    3319    4650#
$SVPC = 000200            1130#   1135
$SWR  = 123000             981#    991    1108    1109    1110    1111    1112    1393    1400    1935    1951    2957    3029
                          3039    3045    3047    3933    3934    3935    3936    3937    3955    3962    3974    3978    3986
                          4635
$SWREG  001230            1425#   1972
$TESTN  001212            1416#   1935*   3063*   3195*   3227*   3443    3457*
$TKB    001162            1392#   3299    3303    4344    4365    4376    4401    4429    4456
$TKCNT  024404            4345#   4360*   4390    4407*   4521    4523#
$TKINT  024422            1985    2178    2418    2482    3271    3306    4360#   4381    4442
$TKQEN= 024421            4349#   4415    4526
$TPGIN  024406            4346#   4361*   4362    4413*   4414*   4415    4417*
$TPGOU  024410            4347#   4362*   4525    4526*   4526    4528*
$TPUSR  024412            4348#   4361    4417    4525
```

```
$TKS     001160          1391#  3304#  4344   4366#  4397#  4399   4405#  4427   4443#  4453   4465#  4485#
$TKSRV   024472          4363   4376#
$TN    = 000002           981#   991   1933   1935#
$TNPWR   026232          4703   4704   4724#
$TPB     001166          1394#  4124#  4135
$TPFLG   001173          1398#  4073   4135
$TPS     001164          1393#  4122   4135
$TRAP    026412          1947   4800#
$TRAP2   026434          4811#  4822
$TRP   = 000014          4815#  4824#  4825#  4826#  4827#  4828#  4829   4830#  4831   4832#  4833#  4834#  4835#
                         4836#
$TRPAD   026446          4805   4822#
$TSTM    001104          1159#
$TSTNM   001116          1371#  3954   3986
$TTYIN   025576          4539   4540   4552   4570   4584   4588#
$TYPBN=  ******  U       4828
$TYPOS   024160          4285#  4827
$TYPE    023202          4073#  4168   4815   4823
$TYPEC   023414          4103   4110   4117   4122#  4123   4487
$TYPEX   023462          4128   4130   4133#
$TYPOC   023756          4225#  4824
$TYPON   023772          4224   4227#  4826
$TYPOS   023732          4220#  4825
$UNIT    001220          1419#  1468
$UNITM   001110          1161#
$JSWR    001232          1426#
$VECT1   001256          1451#  2004   2006
$VECT2   001260          1452#
$$GET4=  000000          3039#
$OFILL   024155          4221#  4225#  4235   4270#
$40CAT=  ******  U       3964
.      = 044732          1118#  1122#  1130   1131#  1133#  1135#  1136#  1140#  1146   1147#  1149#  1151#  1163
                         1368#  1404   1510#  1513#  1516#  1519#  1522#  1942   2005   2008   2083   2097   2307
                         2347   2743   2770   2848   2879   2883   2896   3047   3986   4135   4189#  4339#  4344
                         4348#  4349   4350#  4588#  4589   4596   4613   4637   4744#  6143#  6144#  6145#  6146#
                         6147#  6148#  6149#  6150#  6151
.$ASTA=  ******  U       4141   4144
.$,    = 001100          1146#  1151
```

```
CKCHR     981#  2182  2254  3885  3899
CKCIG     981#  3344
CKNUM     981#
COMMEN   1101#
ENDCOM   1101#
ENDTYP   2953#  2961
ERRCAL   4837#  5655  5707  5710  5974  6035
ERRENT   3926#  3943
ERROR     995#  2317  2318  2319  2320  2321  2322  2753  2754  2755  2756  2757  2758  2785  2786
         2787   2788  2789  2795  2802  2806  2808  2906  2907  2908  2909  2910  2917  2919  2924
         2989   2990  2991  2992  2993  2994  3072  5656  5708  5712  5976  6036
ESCAPE   1101#  2315  2751  2783  2798  2904
GETPRI   1101#
GETSWR    981#  1101#  1986
LOOP      981#  2770  2896
MORETA   1362#  1459
MULT     1101#
NEWTST   1101#  1933
POP      1101#  4183  4184  4326  4622  4623  4784
PUSH     1101#  4144  4146  4167  4285  4603  4609  4764
REPORT   1101#
RM03.D      1#  4837
SCOPE     996#
SETPRI   1101#  4517
SETTRA   4815#  4824  4825  4826  4827  4829  4831  4832  4833  4834  4835
SETUP    1101#  1937
SKIP     1101#
SLASH    1101#
SPACE    1101#
STARS    1101#  1128  1143  1145  1152  1165  1169  1211  1215  1335  1339  1364  1404  1407  1922
         1926   1933  2955  3080  3084  3929  4058  4139  4197  4275  4343  4420  4435  4506  4530
         4599   4615  4642  4666  4685  4748  4794  4848  6261  6262  6352  6356
SWPSU    1101#  1953#
TRMTRP   4815#
TYPBIN   1101#
TYPDEC   1101#  3020  4026
TYPNAM   1101#
TYPNUM   1101#
TYPOCS   1101#  2027  2279
TYPOCT   1101#  4023  4448
TYPTXT   1101#
$$CMRE   1362#
$$CMTM   1362#
$$SCA    1101#
$$NEWT   1101#  1933
$$SET    4815#  4824  4825  4826  4827  4829  4831  4832  4833  4834  4835
$$SETM   1969#
$$SKIP   1101#
.EQUAT    981#   991
.HEADE    981#
.SETUP    981#  1928
.SWRHI    981#  1104
.SWRLO   1112#  1113  1114
.$ACTI    981#  1126
.$APTB    981#  1405#
.$APTH    981#  1141
```

```
.$APTY    981#   4137
.$CATC    981#   1116
.$CMTA    981#   1362
.$DB2D    981#   4683
.$EOP     981#   2953
.$ERRO    981#   3927
.$ERRT    981#
.$POWE    981#   4597
.$READ    981#   4341
.$SAVE    981#   4746
.$SB2D    981#   4664
.$SUPR    981#   4640
.$TRAP    981#   4792
.$TYPD    981#   4273
.$TYPE    981#   4056
.$TYPO    981#   4195


. ABS.  044532     000

 ERRORS DETECTED:  0

 RM03:CZRMAC,RM03:CZRMAC.SEQ/NL:MD:MC:CND:TOC/LI:M_/DOC/SOL/CRF=RM03:RMDRV5.P11,PM03:CZRMAC.P11
 RUN-TIME:  29 19 1 SECONDS
 RUN-TIME RATIO:  439/50=8.6
 CORE USED:  39K  (77 PAGES)

 DOCUMENT PAGES:  139
```