RM05/3/2 RM05/3/2 FORMATTER AH-F919A-MC
CZRMLAO FICHE 1 OF 1

d i g i t a l

```
.REM    %
```

IDENTIFICATION

PRODUCT CODE: AC-F918A-MC

PRODUCT NAME: CZRMLA0 RM05/3/2 FORMATTER

DATE CREATED: APRIL 1980

MAINTAINER: CX DIAGNOSTIC ENGINEERING

AUTHOR: MIKE LEAVITT

CONTENTS

1. ABSTRACT

THE RM05/3/2 FORMATTER PROGRAM PROVIDES THE FACILITIES TO FORMAT, TO CHECK THE HEADER AND DATA FIELDS OR VERIFY THE HEADER FIELD OF EACH DATA BLOCK ON THE 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. ALSO, WRITES THE DATA FIELD WITH SELECTED DATA PATTERN.
THE PROGRAM THEN VERIFIES THE WRITTEN DATA BLOCKS VIA EXECUTING THE "WRITE CHECK HEADER AND DATA" COMMAND.
THE PROGRAM THEN PERFORMS A VERIFY OF SECTOR ZERO, ON EACH OF THE TRACKS SELECTED FOR FORMATTING. THE VERIFY IS ACCOMPLISHED BY EXECUTING A "READ HEADER AND DATA" COMMAND AND COMPARING THE TWO HEADER WORDS TO AN EXPECTED VALUE.

IN THE VERIFY OPERATION, THE PROGRAM READS THE HEADER OF SECTOR 0 THRU 29.(18 BIT MODE) OR 31. (16 BIT MODE) ON EACH CYLINDER AND TRACK TO BE VERIFIED, THEN COMPARES THE TWO HEADER WORDS TO MAKE SURE THEY ARE CORRECT.

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
KW11-L OR KW11-P CLOCK
PROGRAM LOADING DEVICE
TERMINAL
RH11 OR RH70 CONTROLLER
1 TO 8 DISK DRIVES (ANY COMBINATION OF RM05'S, RM03'S OR RM02'S)

2.2 PRELIMINARY PROGRAMS

THERE ARE NO PREQUISITE PROGRAMS, PROVIDING THE RM SUBSYSTEM IS KNOWN TO BE OPERATIONAL. IF THE STATE OF THE RM SUBSYSTEM IS UNKNOWN, THE FOLLOWING PROGRAMS SHOULD BE RUN PRIOR TO USING THE FORMATTER PROGRAM.

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

RM05/3/2 FUNCTIONAL TEST, PART 1

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.(18 BIT MODE) SECTORS FOR A TRACK IN ALL OPERATIONS.

4. STARTING PROCEDURES

4.1 STARTING ADDRESSES

THE PROGRAM IS STARTED FROM LOCATION 200 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 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 OR 204

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 (F, C OR V):'

ENTER THE APPROPRIATE CODE: 'F' FOR FORMAT & QUICK VERIFY, 'C' FOR CHECK OR 'V' FOR VERIFY OPERATION. IF A CARRIAGE RETURN IS ENTERED IN RESPONSE TO THE REQUEST, THE PROGRAM WILL ASSUME 'FORMAT & QUICK 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 DECIMAL 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 (0, 1 OR 2):'

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'

OR

'STARTING VERIFY ON DRIVE N'

NOTE: IF A NEW PACK IS UNFORMATTED, THE PROGRAM ALSO ASKS YOU TO ENTER TWO SERIAL NUMBERS. (EXCEPT IF OPERATING IN VERIFY ONLY MODE) THESE TWO SERIAL NUMBER MUST BE NON-ZERO DECIMAL NUMBERS.

10. THE OPERATOR CAN DETERMINE WHERE THE DRIVE IS DURING THE OPERATION BY TYPING A 'CONTROL O'. THE PROGRAM WILL TYPE THE FOLLOWING MESSAGE:

'PRESENT ADDRESS IS: CXXX TXX'

11. TO CHANGE EITHER THE ADDRESSING MODE OR THE OPERATION MODE, THE PROGRAM MUST BE STARTED FROM LOCATION 200 OR 204 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. IF THE RH11 - RH70 IS NOT AT THE DEFAULT ADDRESS, THE PROGRAM MUST BE STARTED FROM 204 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. 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 RM 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. 'RH 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 RH 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. 'RH DIDN'T RESPOND TO ADDRESSING' - THE PROGRAM ACCESSED THE RH 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. 'TRACK/SECTOR FIELD IN HEADER IS NOT CORRECT' - THE TRACK AND SECTOR FIELD FROM A HEADER READ DURING THE VERIFICATION IS NOT CORRECT.

20. 'WRITE CHECK ERROR' - THE RH REPORTED A WRITE CHECK ERROR AND NO DRIVE ERRORS WERE SET.

7. MISCELLANEOUS

7.1 FORMAT TIME

IT TAKES APPROX. 10 MINUTES TO FORMAT AN ENTIRE RM03/2 DISK PACK IN 'FORMAT' MODE. THE 'CHECK' MODE TAKES APPROX. 30 MINUTES FOR AN ENTIRE RM03/2 DISK PACK.

IT TAKES APPROX. 30 MINUTES TO FORMAT AN ENTIRE RM05 DISK PACK IN 'FORMAT' MODE. THE 'CHECK' MODE TAKES APPROX. 90 MINUTES FOR AN ENTIRE RM05 DISK PACK.

THE 'VERIFY' ONLY MODE TAKES APPROX. 7 MINUTES FOR AN RM03/2 AND APPROX. 20 MINUTES FOR RM05.

7.2 HALTING THE PROGRAM

THE OPERATOR SHOULD NOT HALT THE PROGRAM VIA THE CONSOLE DURING A FORMAT OPERATION. HALTING THE PROGRAM MAY LEAVE A SECTOR INCORRECTLY FORMATTED. TO TERMINATE THE FORMAT, TYPE A 'CONTROL C'.

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 (QUICK VERIFY)

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.

8.4 HEADER VERIFICATION (VERIFY ONLY)

THE HEADER VERIFICATION OPERATION PERFORMS A READ HEADER AND DATA TO AN ENTIRE TRACK AT A TIME, THEN COMPARES THE EXPECTED HEADER WORDS WITH RECIEVED HEADER WORDS. THE CYLINDER ADDRESS FIELD FROM THE HEADER IS COMPARED TO THE EXPECTED CYLINDER, IF THE CYLINDER ADDRESSES DO NOT COMPARE, AN ERROR MESSAGE IS REPORTED. NEXT, THE TRACK/SECTOR FIELD FROM THE HEADER IS COMPARED TO THE EXPECTED TRACK/SECTOR, IF THE TRACK/SECTOR ADDRESSES DO NOT COMPARE, AN ERROR IS REPORTED. VERIFICATION OF THE HEADERS STARTS FROM SECTOR 0 ON THE BEGINNING CYLINDER AND TRACK, AND CONTINUES UNTIL COMPARING SECTOR 29.(18 BIT MODE) OR 31. (16 BIT MODE) ON ENDING CYLINDER AND TRACK. THIS CHECK IS PERFORMED TO CONFIRM THAT A DISK HAS PREVIOUSLY BEEN FORMATTED

AND ALL THE HEADERS ARE IN GOOD CONDITION.

9. BAD SPOT FILE

SECTORS 0 THRU 31.,ON CYLINDER 822., LAST TRACK 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
NUMBER AND CONSISTS OF MAXIMUM 10 OCTAL DIGITS.
THE FIRST WORD OF THE SERIAL NUMBER CONTAINS THE 5
LEAST SIGNIFICANT OCTAL DIGITS,WHILE THE SECOND
WORD CONTAINS THE 5 MOST SIGNIFICANT DIGITS.

THE THIRD WORD IS ALWAYS ZERO.

THE FORTH WORD DEFINES THE PACK IS A DATA PACK ,IF IT IS 0.

THE FOLLOWING 252 WORDS DEFINE THE BAD SECTOR ADDRESS;TWO
WORDS ARE USED TO DEFINE A SIGNLE BAD SECTOR, THE FIRST
WORD SPECIFIES THE CYLINDER ADDRESS, THE SECOND WORD
SPEDIFIES THE TRACK (HIGH BYTE) AND SECTOR (LOW BATE) ADDRESSES.
ALL UNUSED LOCATIONS ARE RECORDED WITH ONES.

THE SECTORS ON THE LAST TRACK ARE FURTHER DIVIDED INTO FOUR
GROUPS AS FOLLOWS:

(1) SECTORS 0, 2, 4, 6 AND 8. CONTAIN THE MANUFACTURES DEFINED BAD
SECTORS FOR 16 BIT MODE.

(2) SECTORS 1, 3, 5, 7 AND 9. CONTAIN THE MANUFACTURES DEFINED BAD
SECTORS FOR 18 BIT MODE.

(3) SECTORS 10., 12. THRU 30. CONTAIN THE USER DEFINED BAD SECTORS
FOR 16 BIT MODE.

(4) SECTORS 11., 13. THRU 31. CONTAIN THE USER DEFINED BAD SECTORS
FOR 18 BIT MODE.

THE SECTORS IN EACH GROUP ARE IDENTICAL AND SECTORS 0, 1, 10. AND 11.
ARE USED AS REFERENCE SECTORS BY THE PROGRAM.

THE PROGRAM PROVIDES A FEW UTILITY ROUTINES TO HANDLE THE BAD
SECTOR FILE.
TO ACCESS THESE ROUTINES START THE PROGRAM AT 204.

BEFORE THE PROGRAM FORMATS THE PACK,THE PROGRAM TYPES THE FOLLOWING
MESSAGE TO ALLOW THE OPERATOR TO ACCESS THE ROUTINES
BY ENTERING THE SELECTED FUNCTION NUMBER.

```
KEYIN U.S.R. BAD SECTORS.............................0
PRINT M.F.G. BAD SECTORS.............................1
```

```
                        PRINT U.S.R. BAD SECTORS.....................2
                        INITIALIZE U.S.R. BAD SECTOR FILE............3
                        PRINT A SECTOR OF THE LAST TRACK.............4
                        PRINT THIS MESSAGE...........................5
                        EXIT.....................................<CR>
                        ENTER UTILITY FUNCTION:


                        THE FIRST ROUTINE ALLOWS THE USER DEFINED BAD SECTORS
                        TO BE ENTERED MANUALLY BY SPECIFING THE CYLINDER ADDRESS, TRACK
                        ADDRESS AND SECTOR ADDRESS IN DECIMAL VALUE.  TO EXIT FROM
                        THE ROUTINE JUST ENTER A PERIOD(.) FOLLOWED BY A <CR>.

                        EXAMPLE:
                                CYLINDER  -1 / 200
                                TRACK  -1 / 4
                                SECTOR  -1 / 23
                                CYLINDER  -1 / .<CR>

                        THE SECOND AND THIRD ROUTINES TYPE THE MANUFACTURE AND USER
                        DEFINED BAD SECTORS ON THE CONSOLE IN OCTAL NUMBER.
                        THE FIRST TWO WORDS ON THE LISTING ARE THE SERIAL NUMBERS FOLLOWED
                        BY TWO WORDS OF ALL 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 TYPE A "CARRIAGE RETURN" <CR>.


                10.1    RH/RM DRIVER
                        --------------------

                        THIS DOCUMENT IS THE USER'S GUIDE FOR THE RH/RM 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
                                  =0            OFFLINE, DRIVE
                                                IS NOT AN RM05/3/2, OR
                                                NONEXISTENT DRIVE
                                  <0            UNSAFE
```

```
                                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
                                          5             RM02
                                          7             RM05
                                         -1             NOT AN RM05/3/2

                                THE 'RMINIT' ROUTINE WILL DO A READIN PRESET AND WILL SET FMT16.

                        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 FMT16, ECT, AND HCI
                                        .BYTE   0               ;(2) COMMAND
                                        .BYTE   0               ;(3) PSEL AND A17 AND A16
                                        .WORD   0               ;(4) WORD COUNT (MUST BE NEG.)
                                        .WORD   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 RH/RM
                                                                ;REGISTERS ON AN ERROR. IF LEFT
                                                                ;ZERO REGISTERS ARE NOT SAVED.
                                        .WORD   0               ;(16) STATUS/ERROR INDICATOR
                                                                ;BIT15=1=>ERROR OCCURRED
                                                                ;BIT07=1=>DONE
                                                                ;BIT14-BIT09 AND BIT06-BIT03
                                                                ;INDICATE TYPE OF ERROR

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

        FMTDPB: .BYTE   5               ;DRIVE #5
                .BYTE   0               ;
                .BYTE   161             ;WRITE COMMAND
                .BYTE   0               ;
                .WORD   -1000.          ;WORD COUNT
                .WORD   WRTBUF          ;BUFFER ADDRESS
                .BYTE   3               ;SECTOR
                .BYTE   5               ;TRACK
                .WORD   400             ;CYLINDER
                .WORD   ERRTB5          ;ERROR TABLE
                .WORD   0               ;STATUS/ERROR INDICATOR

                ALTERNATE DPB SETUP

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

10.5    RH/RM 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
                                        WRT CHK HDR & DATA      153                 D
                                        WRITE DATA              161                 D
                                        WRITE HEADER & DATA     163                 D
                                        READ DATA               171                 D
                                        READ HEADER & DATA      173                 D


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

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. (RH/RM REGISTERS WERE NOT SAVED)
(2) =>		REQUEST QUEUE HAS BEEN EMPTIED. THE DRIVER ISSUED A "DRIVE CLEAR" TO THE DRIVE. NOTE: ALL RH/RM REGISTERS ARE SAVED AS PER DPB+14 BEFORE THE "DRIVE CLEAR".
(3) =>		REQUEST QUEUE HAS BEEN EMPTIED. THE DRIVER ISSUED A MASSBUS INIT. ALL RH/RM 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= RMCS1'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)

                %
```

```
                        ;PROGRAM REVISION #001

                        .TITLE  CZRMLA0 RM05/3/2 FORMATTER
                        ;*COPYRIGHT (C) 1980
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY MIKE LEAVITT
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C4), 1980.
                        ;*
                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH                  USE
                        ;*      ------          --------------------
                        ;*        15            HALT ON ERROR
                        ;*        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  BASIC DEFINITIONS
                        ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                  STACK   = 1100
104000                  ERROR   = EMT           ;;BASIC DEFINITION OF ERROR CALL
000004                  SCOPE   = IOT           ;;BASIC DEFINITION OF SCOPE CALL

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

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

                        ;*PRIORITY LEVEL DEFINITIONS
000000                  PR0     = 0             ;;PRIORITY LEVEL 0
```

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

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

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

```
                000040          BIT5=BIT05
                000020          BIT4=BIT04
                000010          BIT3=BIT03
                000004          BIT2=BIT02
                000002          BIT1=BIT01
                000001          BIT0=BIT00

                                ;*BASIC "CPU" TRAP VECTOR ADDRESSES
                000004          ERRVEC  = 4             ;;TIME OUT AND OTHER ERRORS
                000010          RESVEC  = 10            ;;RESERVED AND ILLEGAL INSTRUCTIONS
                000014          TBITVEC = 14            ;;"T" BIT
                000014          TRTVEC  = 14            ;;TRACE TRAP
                000014          BPTVEC  = 14            ;;BREAKPOINT TRAP (BPT)
                000020          IOTVEC  = 20            ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                000024          PWRVEC  = 24            ;;POWER FAIL
                000030          EMTVEC  = 30            ;;EMULATOR TRAP (EMT) **ERROR**
                000034          TRAPVEC = 34            ;;"TRAP" TRAP
                000060          TKVEC   = 60            ;;TTY KEYBOARD VECTOR
                000064          TPVEC   = 64            ;;TTY PRINTER VECTOR
                000240          PIRQVEC = 240           ;;PROGRAM INTERRUPT REQUEST VECTOR
     79
     80                         .SBTTL  RH11/RH70 REGISTERS
     81
     82                         ;CONTROL AND STATUS REGISTER 1 (RMCS1)
     83
     84         000100          IE      = 100                   ;INTERRUPT ENABLE (BIT #6)
     85         000200          RDY     = 200                   ;READY (BIT #7)
     86         000400          A16     = 400                   ;HIGH ORDER BUS ADDRESS BIT (BIT #8)
     87         001000          A17     = 1000                  ;HIGH ORDER BUS ADDRESS BIT (BIT #9)
     88         002000          PSEL    = 2000                  ;PORT SELECT (BIT #10)
     89         020000          MCPE    = 20000                 ;MASSBUSS PARITY ERROR (BIT #13)
     90         040000          TRE     = 40000                 ;TRANSFER ERROR (BIT #14)
     91                         ;SC     = 100000                ;SPECIAL CONDITION (BIT #15)
     92
     93                         ;WORD COUNT REGISTER (RMWC)
     94                         ;(EACH BIT IS CALLED BY BIT NUMBER)
     95
     96                         ;BUS ADDRESS REGISTER (RMBA)
     97                         ;(EACH BIT IS CALLED BY BIT NUMBER)
     98
     99                         ;CONTROL AND STATUS REGISTER 2 (RMCS2)
    100
    101         000001          US1     = 1                     ;UNIT SELECT (BIT #0)
    102         000002          US2     = 2                     ;UNIT SELECT (BIT #1)
    103         000004          US4     = 4                     ;UNIT SELECT (BIT #2)
    104         000010          BAI     = 10                    ;BUS ADDRESS INCREMENT INHIBIT (BIT #3)
    105         000020          PAT     = 20                    ;MASSBUS PARITY TEST (BIT #4)
    106         000040          CLR     = 40                    ;CLEAR (BIT #5)
    107         000100          IR      = 100                   ;INPUT READY (BIT #6)
    108         000200          OR      = 200                   ;OUTPUT READY (BIT #7)
    109         000400          MPE     = 400                   ;MASS BUS PARITY ERROR (BIT #8)
    110         001000          MXF     = 1000                  ;MISSED TRANSFER ERROR (BIT #9)
    111         002000          PGE     = 2000                  ;PROGRAM ERROR (BIT #10)
    112         004000          NEM     = 4000                  ;NON EXISTENT MEMORY (BIT #11)
    113         010000          NED     = 10000                 ;NON EXISTENT DRIVE (BIT #12)
    114         020000          UPE     = 20000                 ;UNIBUS PARITY ERROR (BIT #13)
    115         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  RM REGISTERS

                        ;CONTROL AND STATUS 1 REGISTER. (#00)

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

                        ;DRIVE STATUS REGISTER (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)

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

```
                        DTSY    = 200                   ;MAINTAINABILITY SYNC DETECTED (BIT #7)

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

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

                        ;SERIAL NUMBER REGISTER (RMSN) (#14)
```

```
                        ;(EACH IS CALLED BY BIT NUMBER)
                        ;RM MAINTENANCE  REGISTER #02 (RMMR2) (#15)
040000                  SKI     = 40000                 ;SEEK INCOMPLETE (BIT #14)
                        ;ERROR REGISTER #02 (RMER2)
100000                  BSE     = 100000                ;BAD SECTOR ERROR (BIT #15)
                        ;ECC POSITION REGISTER (RMEC1) (#16)
                        ;(EACH BIT IS CALLED BY BIT NUMBER)
                        ;ECC PATTERN REGISTER (RMEC2) (#17)
                        ;(EACH BIT IS CALLED BY BIT NUMBER)
                        ;SOME 1ST HEADER WORD DEFINITIONS
100000                  MF      = 100000                ;MANUFACTURES BAD SECTOR FLAG (BIT #15)
040000                  UF      = 40000                 ;USERS BAD SECTOR FLAG (BIT #14)
010000                  FMT     = 10000                 ;FORMAT 16 BIT MODE (BIT #12)
                        .SBTTL  RM DRIVER COMMANDS
000101                  RNOP    = 101                   ;NO OPERATION
000105                  SEEK    = 105                   ;SEEK
000107                  RECAL   = 107                   ;RECALIBRATE
000111                  DRVCLR  = 111                   ;DRIVE CLEAR
000113                  RELSE   = 113                   ;RELEASE
000115                  OFFSET  = 115                   ;OFFSET
000117                  RTC     = 117                   ;RETURN TO CENTER LINE
000121                  READIN  = 121                   ;READ IN PRESET
000123                  ACK     = 123                   ;PACK ACKNOWLEDGE
000131                  SEARCH  = 131                   ;SEARCH
000141                  GETREG  = 141                   ;GET REGISTERS
000143                  SETFMT  = 143                   ;SET FORMAT (& ECI OR HCI)
000145                  SELDRV  = 145                   ;SELECT DRIVE
000151                  WCKD    = 151                   ;WRITE CHECK DATA
000153                  WCKHD   = 153                   ;WRITE CHECK HEADER & DATA
000161                  WRTDAT  = 161                   ;WRITE DATA
000163                  WRTHD   = 163                   ;WRITE HEADER & DATA
000171                  RDDAT   = 171                   ;READ DATA
000173                  RDHD    = 173                   ;READ HEADER & DATA
176700                  ABASE   = 176700
000254                  AVECT1  = 254
                        .SBTTL  TRAP CATCHER
000000                          .=0
                        ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                        ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                        ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
000174                          .=174
000174  000000  DISPREG: .WORD  0                       ;;SOFTWARE DISPLAY REGISTER
000176  000000  SWREG:   .WORD  0                       ;;SOFTWARE SWITCH REGISTER
```

```
                                .SBTTL  STARTING ADDRESS(ES)

     000200  000137  007014            JMP     @#START         ;;JUMP TO STARTING ADDRESS OF PROGRAM
280
281  000204  000137  006776            JMP     @#BEGIN         ;CHANGE THE RH ADDRESS
282
283                                .SBTTL  ACT11 HOOKS

                                ;;******************************************************************
                                ;HOOKS REQUIRED BY ACT11
             000210                     $SVPC=.                 ;SAVE PC
             000046                     .=46
     000046  016772                     $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
             000052                     .=52
     000052  020000                     .WORD   20000           ;;2)SET LOC.52 TO 20000
             000210                     .=$SVPC                 ;; RESTORE PC
284
285          001100                .=1100
286                                .SBTTL  APT PARAMETER BLOCK

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

     001100                     $APTHD:
     001100  000000             $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
     001102  001210             $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 ADDED UNIT
     001112  000032                     .WORD   $ETEND-$MAIL/2  ;;LENGTH MAILBOX-ETABLE(WORDS)
287          001114             TAB.XY=.                ;CMTAGSTARTING ADDRESS
288
```

```
                                .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          $TMP0:  .WORD   0               ;;USER DEFINED
001176  000000          $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001200  207     377 377 $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001204  077             $QUES:  .ASCII  /?/             ;;QUESTION MARK
001205  015             $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001206  012     000     $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;**********************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

                                ;;**********************************************************************
                                .EVEN
001210                  $MAIL:                          ;;APT MAILBOX
001210  000000          $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
001212  000000          $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
001214  000000          $TESTN: .WORD   ATESTN          ;;TEST NUMBER
001216  000000          $PASS:  .WORD   APASS           ;;PASS COUNT
001220  000000          $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
001222  000000          $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
001224  000000          $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
001226  000000          $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
```

```
001230                          $ETABLE:                ;;APT ENVIRONMENT TABLE
001230      000                 $ENV:    .BYTE  AENV    ;;ENVIRONMENT BYTE
001231      000                 $ENVM:   .BYTE  AENVM   ;;ENVIRONMENT MODE BITS
001232   000000                 $SWREG:  .WORD  ASWREG  ;;APT SWITCH REGISTER
001234   000000                 $USWR:   .WORD  AUSWR   ;;USER SWITCHES
001236   000000                 $CPUOP:  .WORD  ACPUOP  ;;CPU TYPE,OPTIONS
                                ;*                      BITS 15-11=CPU TYPE
                                ;*                              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                ;*                              11/70=06,PDQ=07,Q=10
                                ;*                      BIT 10=REAL TIME CLOCK
                                ;*                      BIT  9=FLOATING POINT PROCESSOR
                                ;*                      BIT  8=MEMORY MANAGEMENT
001240      000                 $MAMS1:  .BYTE  AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
001241      000                 $MTYP1:  .BYTE  AMTYP1  ;;MEM. TYPE,BLK#1
                                ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
                                ;*                              900 NSEC CORE=001
                                ;*                              300 NSEC BIPOLAR=002
                                ;*                              500 NSEC MOS=003
001242   000000                 $MADR1:  .WORD  AMADR1  ;;HIGH ADDRESS,BLK#1
                                ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
001244      000                 $MAMS2:  .BYTE  AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
001245      000                 $MTYP2:  .BYTE  AMTYP2  ;;MEM.TYPE,BLK#2
001246   000000                 $MADR2:  .WORD  AMADR2  ;;MEM.LAST ADDRESS,BLK#2
001250      000                 $MAMS3:  .BYTE  AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
001251      000                 $MTYP3:  .BYTE  AMTYP3  ;;MEM.TYPE,BLK#3
001252   000000                 $MADR3:  .WORD  AMADR3  ;;MEM.LAST ADDRESS,BLK#3
001254      000                 $MAMS4:  .BYTE  AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
001255      000                 $MTYP4:  .BYTE  AMTYP4  ;;MEM.TYPE,BLK#4
001256   000000                 $MADR4:  .WORD  AMADR4  ;;MEM.LAST ADDRESS,BLK#4
001260   000254                 $VECT1:  .WORD  AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001262   000000                 $VECT2:  .WORD  AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001264   176700                 $BASE:   .WORD  ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001266   000000                 $DEVM:   .WORD  ADEVM   ;;DEVICE MAP
001270   000000                 $CDW1:   .WORD  ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
001272   000000                 $CDW2:   .WORD  ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
001274                          $ETEND:
                                .MEXIT
```

```
 0                                  .SBTTL  USER DEFINED TAGS

 001274  176700                     $RMADR: .WORD   176700  ;RH/RM UNIBUS ADDRESS
 001276  000254                     $RMVEC: .WORD   254     ;RH INTERRUPT VECTOR
 001300  172540                     $LKCSR: .WORD   172540  ;ADDRESS OF KW11-P CSR
 001302  172542                     $LKCSB: .WORD   172542  ;ADDRESS OF KW11-P COUNTER BUFFER
 001304  000104  000106             $LPVEC: .WORD   104,106 ;ADDRESS OF KW11-P VECTOR
 001310  177546                     $LKS:   .WORD   177546  ;ADDRESS OF KW11-L CONTROL REGISTER
 001312  000100  000102             $LLVEC: .WORD   100,102 ;ADDRESS OF KW11-L VECTOR

         001222                     DRIVE   =$UNIT          ;CONTAINS DRIVE NUMBER SELECTED
                                                            ;SAME PARAMETER USED IN APT
 001316  000000                     SOFSW:  .WORD   0       ;CONTENTS ARE FOR SOFTWARE DECISIONS
 001320  000000                     MODE:   .WORD   0       ;FORMAT = -1, CHECK = 0 AND VERIFY ONLY = 1
 001322  000000                     SNGSEC: .WORD   0       ;READ MULTI-SECTOR = 0, READ SINGLE SECTOR = 1
 001324  000000                     LSTRK:  .WORD   0       ;CONTAINS THE LAST TRACK OF THE UNIT UNDER
                                                            ;TEST. RM02/3 = 4., RM05 = 18.
 001326  000000                     ENDCYL: .WORD   0       ;ENDING CYLINDER
 001330  000000                     BEGCYL: .WORD   0       ;STARTING CYLINDER
 001332  000000                     ENDTRK: .WORD   0       ;ENDING TRACK
 001334  000000                     BEGTRK: .WORD   0       ;STARTING TRACK
 001336  000000                     PATSEL: .WORD   0       ;CONTAINS PATTERN SELECTED
 001340  000000                     PATA:   .WORD   0       ;1ST WORD OF PATTERN
 001342  000000                     PATB:   .WORD   0       ;2ND WORD OF PATTERN
 001344  000000                     RETRY:  .WORD   0       ;MAINTAINS # OF WRITE RETRIES MADE
 001346  000000                     SAVSEC: .WORD   0       ;CONTAINS LAST BAD SECTOR ON FORMAT
 001350  000000                     SAVWC:  .WORD   0       ;CONTAINS WC FOR REMAINING SECTORS ON ERROR
 001352  000000                     WC:     .WORD   0       ;30 OR 32 SECTOR TRACK SIZE (IN WORDS)
 001354  000000                     MWC:    .WORD   0       ;2'S COMPLEMENT OF 'WC'
 001356  000000                     HEDERR: .WORD   0       ;POSITIONING ERROR DURING FORMAT INDICATOR
 001360  000000                     SEC30:  .WORD   0       ;30 OR 32 SECTOR MODE INDICATOR
                                                            ;0 = 30 SECTOR MODE
                                                            ;1'S = 32 SECTOR MODE
 001362  000000                     MAXSEC: .WORD   0       ;MAXIMUM SECTOR ADDRESS (FOR EITHER 30 OR 32 SECTOR
                                                            ;FORMAT)
 001364  000000                     DS.CYL: .WORD   0       ;ADDRESS OF CURRENT CYLINDER
 001366  000000                     DS.TRK: .WORD   0       ;ADDRESS OF CURRENT TRACK
 001370  000000                     DDRIVE: .WORD   0       ;DRIVE NUMBER OF 'DRIVER' ERROR MESSAGES
 001372  000000                     ATTN:   .WORD   0       ;ATTENTION REGISTER IMAGE FOR 'DRIVER'
                                                            ;ERROR MESSAGES
 001374  000000                     CNTLC:  .WORD   0       ;ADDRESS OF '^C' RETURN
 001376  000001                     CHGADR: .WORD   1       ;'CHANGE RH/RM ADDRESS' FLAG = 1
 001400  000000                     WRAP:   .WORD   0       ;WRAP AROUND FLAG,FORMAT FROM HIGH TO LOW
                                                            ;CYLINDER NUMBER
 001402  000000                     DEVMP:  .WORD   0       ;DEVICE MAP FOR APT
 001404  000000                     PACK:   .WORD   0       ;PACK >0, BAD SPOT FILE AVAILABLE FROM PACK
 001406  000000                     EDIT:   .WORD   0       ;EDIT THE BAD SPOT FILE ,IF EDIT >0
 001410  000000                     HEADX:  .WORD   0       ;HEAD BIT INDICATOR
                                                            ;BIT15!BIT14  GOOD SECTOR
                                                            ;BIT15 ALONE, USER DEFINED BAD SECTOR
                                                            ;BIT14 ALONE, MFG DEFINED BAD SECTOR
 001412  000000                     XXDP:   .WORD   0       ;THE LOW BYTE CONTAINS THE DRIVE NUMBER FROM WHICH
                                                            ;THE PROGRAM WAS LOADED. THE HIGH BYTE CONTAINS THE
                                                            ;'XXDP' DEVICE CODE THE RM05/3/2.

 001414                             BAD16:  .BLKW   256.    ;MFG 16 BIT BAD SPOT TABLE
 002414  177777                             .WORD   -1
```

```
002416  177777                          .WORD   -1

002420                          BAD18:  .BLKW   256.    ;MFG 18 BIT BAD SPOT TABLE
003420  177777                          .WORD   -1
003422  177777                          .WORD   -1

003424                          USTAB:  .BLKW   256.    ;USER BAD SPOT TABLE
004424  177777                          .WORD   -1
004426  177777                          .WORD   -1

004430                          USSAV:  .BLKW   256.    ;USER BAD SPOT TABLE SAVE FROM PACK
005430  177777                          .WORD   -1
005432  177777                          .WORD   -1

005434                          SECCU:  .BLKW   32.     ;32 WORDS MAX TO STORE BAD SECTORS
005534  177777                          .WORD   -1
005536  000000                  NEXT:   .WORD   0       ;FLAG, IF NEXT=-1,ALL GOOD SPOT ON
                                                        ;THE CURRENT TRACK ARE FORMATTED

                                ;DATA/PARAMETER BLOCK - USED FOR ALL DRIVE OPERATION

005540      000                 FMTDPB: .BYTE   0       ;DRIVE NUMBER
005541      000                         .BYTE   0       ;OFFSET VALUE OR FMT16, ECI, AND HCI
005542      000                         .BYTE   0       ;COMMAND
005543      000                         .BYTE   0       ;PSEL AND A17 AND A16
005544  000000                          .WORD   0       ;WORD COUNT (NEG)
005546  000000                          .WORD   0       ;BUFFER ADDRESS
005550      000                         .BYTE   0       ;SECTOR ADDRESS
005551      000                         .BYTE   0       ;TRACK ADDRESS
005552  000000                          .WORD   0       ;CYLINDER ADDRESS
005554  005560                          .WORD   RM.REG  ;ERROR TABLE POINTER
005556  000000                          .WORD   0       ;STATUS-ERROR INDICATOR
                                                        ;BIT 15 = 1: ERROR OCCURRED
                                                        ;BIT 07 = 1: DONE
                                                        ;BIT 14-10 AND BIT 06-03
                                                        ;INDICATE TYPE OF ERROR

                                ;RH/RM REGISTERS STORED HERE AFTER AN OPERATION

005560  000000                  RM.REG: .WORD   0       ;RMCS1
005562  000000                          .WORD   0       ;RMWC
005564  000000                          .WORD   0       ;RMBA
005566  000000                          .WORD   0       ;RMDA
005570  000000                          .WORD   0       ;RMCS2
005572  000000                          .WORD   0       ;RMDS
005574  000000                          .WORD   0       ;RMER1
005576  000000                          .WORD   0       ;RMAS
005600  000000                          .WORD   0       ;RMLA
005602  000000                          .WORD   0       ;RMDB
005604  000000                          .WORD   0       ;RMMR1
005606  000000                          .WORD   0       ;RMDT
005610  000000                          .WORD   0       ;RMSN
005612  000000                          .WORD   0       ;RMOF
005614  000000                          .WORD   0       ;RMCA
005616  000000                          .WORD   0       ;RMDC
005620  000000                          .WORD   0       ;RMER2
005622  000000                          .WORD   0       ;RMMR2
```

```
005624  000000                          .WORD   0       ;RMEC1
005626  000000                          .WORD   0       ;RMEC2

005630     000                  FMTX:   .BYTE   0       ;SPECIAL DPB TO WRITE HEADER OF BAD SPOT
005631     000                          .BYTE   0
005632     000                          .BYTE   0
005633     000                          .BYTE   0
005634  177776                          .WORD   -2      ;ONLY TWO WORDS FOR HEADER
005636  045764                          .WORD   RBUF    ;BUFFER ADDRESS
005640     000                          .BYTE   0       ;SECTOR
005641     000                          .BYTE   0       ;TRACK
005642  000000                          .WORD   0       ;CYLINDER ADDRESS
005644  000000                          .WORD   0       ;ERROR TABLE
                                                        ;DONT''SAVE ANYTHING
005646  000000                          .WORD   0       ;STATUS WORD

                             ;PARAMETER LIMIT POINTER TABLE

005650  005776  001467  001330  TABLE:  PAR1,823.,BEGCYL
005656  006011  000005  001334          PAR2,5.,BEGTRK
005664  006024  001467  001326          PAR3,823.,ENDCYL
005672  006035  000005  001332          PAR4,5.,ENDTRK,0

005702  006046  001467  001364  TABLE2: PAR5,823.,DS.CYL
005710  006057  000005  001366          PAR6,5.,DS.TRK
005716  006070  000040  001346          PAR7,32.,SAVSEC,0

005726  006046  001467  001364  TABLE3: PAR5,823.,DS.CYL
005734  006057  000005  001366          PAR6,5.,DS.TRK
005742  006070  000036  001346          PAR7,30.,SAVSEC,0

005752  006046  001467  001364  TABLE4: PAR5,823.,DS.CYL
005760  006057  000005  001366          PAR6,5,DS.TRK
005766  006101  047301  001346          PAR8,20161.,SAVSEC,0

                             ;ASCII MESSAGES FOR ADDRESS PARAMETERS

005776     123     124     101  PAR1:   .ASCIZ  @START CYL @
006011     123     124     101  PAR2:   .ASCIZ  @START TRK @
006024     105     116     104  PAR3:   .ASCIZ  @END CYL @
006035     105     116     104  PAR4:   .ASCIZ  @END TRK @

006046     103     131     114  PAR5:   .ASCIZ  /CYLINDER/
006057     124     122     101  PAR6:   .ASCIZ  /TRACK   /
006070     123     105     103  PAR7:   .ASCIZ  /SECTOR  /
006101     102     131     124  PAR8:   .ASCIZ  /BYTE POSITION /
                                .EVEN

006120  022116  022414  022502  TABCAL: .WORD   FUNCT2,FUNCT3,FUNCT4,FUNCT6,FUNCT7

                             ;SECTOR BUFFER ADDRESS TABLE

006132  045774                  ADRTBL: .WORD   BUFP             ;ADDRESS OF SECTOR 0
006134  047000                          .WORD   BUFP+<516.*1.>   ;ADDRESS OF SECTOR 1.
006136  050004                          .WORD   BUFP+<516.*2.>   ;ADDRESS OF SECTOR 2.
006140  051010                          .WORD   BUFP+<516.*3.>   ;ADDRESS OF SECTOR 3.
006142  052014                          .WORD   BUFP+<516.*4.>   ;ADDRESS OF SECTOR 4.
```

```
006144  053020                  .WORD   BUFP+<516.*5.>  ;ADDRESS OF SECTOR 5.
006146  054024                  .WORD   BUFP+<516.*6.>  ;ADDRESS OF SECTOR 6.
006150  055030                  .WORD   BUFP+<516.*7.>  ;ADDRESS OF SECTOR 7.
006152  056034                  .WORD   BUFP+<516.*8.>  ;ADDRESS OF SECTOR 8.
006154  057040                  .WORD   BUFP+<516.*9.>  ;ADDRESS OF SECTOR 9.
006156  060044                  .WORD   BUFP+<516.*10.> ;ADDRESS OF SECTOR 10.
006160  061050                  .WORD   BUFP+<516.*11.> ;ADDRESS OF SECTOR 11.
006162  062054                  .WORD   BUFP+<516.*12.> ;ADDRESS OF SECTOR 12.
006164  063060                  .WORD   BUFP+<516.*13.> ;ADDRESS OF SECTOR 13.
006166  064064                  .WORD   BUFP+<516.*14.> ;ADDRESS OF SECTOR 14.
006170  065070                  .WORD   BUFP+<516.*15.> ;ADDRESS OF SECTOR 15.
006172  066074                  .WORD   BUFP+<516.*16.> ;ADDRESS OF SECTOR 16.
006174  067100                  .WORD   BUFP+<516.*17.> ;ADDRESS OF SECTOR 17.
006176  070104                  .WORD   BUFP+<516.*18.> ;ADDRESS OF SECTOR 18.
006200  071110                  .WORD   BUFP+<516.*19.> ;ADDRESS OF SECTOR 19.
006202  072114                  .WORD   BUFP+<516.*20.> ;ADDRESS OF SECTOR 20.
006204  073120                  .WORD   BUFP+<516.*21.> ;ADDRESS OF SECTOR 21.
006206  074124                  .WORD   BUFP+<516.*22.> ;ADDRESS OF SECTOR 22.
006210  075130                  .WORD   BUFP+<516.*23.> ;ADDRESS OF SECTOR 23.
006212  076134                  .WORD   BUFP+<516.*24.> ;ADDRESS OF SECTOR 24.
006214  077140                  .WORD   BUFP+<516.*25.> ;ADDRESS OF SECTOR 25.
006216  100144                  .WORD   BUFP+<516.*26.> ;ADDRESS OF SECTOR 26.
006220  101150                  .WORD   BUFP+<516.*27.> ;ADDRESS OF SECTOR 27.
006222  102154                  .WORD   BUFP+<516.*28.> ;ADDRESS OF SECTOR 28.
006224  103160                  .WORD   BUFP+<516.*29.> ;ADDRESS OF SECTOR 29.
006226  104164                  .WORD   BUFP+<516.*30.> ;ADDRESS OF SECTOR 30.
006230  105170                  .WORD   BUFP+<516.*31.> ;ADDRESS OF SECTOR 31.

                        ;REMAINING WORD COUNT TABLE

006232  000402          WCTBL:  .WORD   258.            ;REMAINING WORD COUNT AFTER SECTOR 0
006234  001004                  .WORD   258.+<258.*1.>  ;REMAINING WORD COUNT AFTER SECTOR 1.
006236  001406                  .WORD   258.+<258.*2.>  ;REMAINING WORD COUNT AFTER SECTOR 2.
006240  002010                  .WORD   258.+<258.*3.>  ;REMAINING WORD COUNT AFTER SECTOR 3.
006242  002412                  .WORD   258.+<258.*4.>  ;REMAINING WORD COUNT AFTER SECTOR 4.
006244  003014                  .WORD   258.+<258.*5.>  ;REMAINING WORD COUNT AFTER SECTOR 5.
006246  003416                  .WORD   258.+<258.*6.>  ;REMAINING WORD COUNT AFTER SECTOR 6.
006250  004020                  .WORD   258.+<258.*7.>  ;REMAINING WORD COUNT AFTER SECTOR 7.
006252  004422                  .WORD   258.+<258.*8.>  ;REMAINING WORD COUNT AFTER SECTOR 8.
006254  005024                  .WORD   258.+<258.*9.>  ;REMAINING WORD COUNT AFTER SECTOR 9.
006256  005426                  .WORD   258.+<258.*10.> ;REMAINING WORD COUNT AFTER SECTOR 10.
006260  006030                  .WORD   258.+<258.*11.> ;REMAINING WORD COUNT AFTER SECTOR 11.
006262  006432                  .WORD   258.+<258.*12.> ;REMAINING WORD COUNT AFTER SECTOR 12.
006264  007034                  .WORD   258.+<258.*13.> ;REMAINING WORD COUNT AFTER SECTOR 13.
006266  007436                  .WORD   258.+<258.*14.> ;REMAINING WORD COUNT AFTER SECTOR 14.
006270  010040                  .WORD   258.+<258.*15.> ;REMAINING WORD COUNT AFTER SECTOR 15.
006272  010442                  .WORD   258.+<258.*16.> ;REMAINING WORD COUNT AFTER SECTOR 16.
006274  011044                  .WORD   258.+<258.*17.> ;REMAINING WORD COUNT AFTER SECTOR 17.
006276  011446                  .WORD   258.+<258.*18.> ;REMAINING WORD COUNT AFTER SECTOR 18.
006300  012050                  .WORD   258.+<258.*19.> ;REMAINING WORD COUNT AFTER SECTOR 19.
006302  012452                  .WORD   258.+<258.*20.> ;REMAINING WORD COUNT AFTER SECTOR 20.
006304  013054                  .WORD   258.+<258.*21.> ;REMAINING WORD COUNT AFTER SECTOR 21.
006306  013456                  .WORD   258.+<258.*22.> ;REMAINING WORD COUNT AFTER SECTOR 22.
006310  014060                  .WORD   258.+<258.*23.> ;REMAINING WORD COUNT AFTER SECTOR 23.
006312  014462                  .WORD   258.+<258.*24.> ;REMAINING WORD COUNT AFTER SECTOR 24.
006314  015064                  .WORD   258.+<258.*25.> ;REMAINING WORD COUNT AFTER SECTOR 25.
006316  015466                  .WORD   258.+<258.*26.> ;REMAINING WORD COUNT AFTER SECTOR 26.
```

```
006320  016070                          .WORD   258.+<258.*27.> ;REMAINING WORD COUNT AFTER SECTOR 27.
006322  016472                          .WORD   258.+<258.*28.> ;REMAINING WORD COUNT AFTER SECTOR 28.
006324  017074                          .WORD   258.+<258.*29.> ;REMAINING WORD COUNT AFTER SECTOR 29.
006326  017476                          .WORD   258.+<258.*30.> ;REMAINING WORD COUNT AFTER SECTOR 30.
006330  020100                          .WORD   258.+<258.*31.> ;REMAINING WORD COUNT AFTER SECTOR 31.

006332  001166                  BYTE16: .WORD   630.
006334  002354                          .WORD   630.+<630.*1.>
006336  003542                          .WORD   630.+<630.*2.>
006340  004730                          .WORD   630.+<630.*3.>
006342  006116                          .WORD   630.+<630.*4.>
006344  007304                          .WORD   630.+<630.*5.>
006346  010472                          .WORD   630.+<630.*6.>
006350  011660                          .WORD   630.+<630.*7.>
006352  013046                          .WORD   630.+<630.*8.>
006354  014234                          .WORD   630.+<630.*9.>
006356  015422                          .WORD   630.+<630.*10.>
006360  016610                          .WORD   630.+<630.*11.>
006362  017776                          .WORD   630.+<630.*12.>
006364  021164                          .WORD   630.+<630.*13.>
006366  022352                          .WORD   630.+<630.*14.>
006370  023540                          .WORD   630.+<630.*15.>
006372  024726                          .WORD   630.+<630.*16.>
006374  026114                          .WORD   630.+<630.*17.>
006376  027302                          .WORD   630.+<630.*18.>
006400  030470                          .WORD   630.+<630.*19.>
006402  031656                          .WORD   630.+<630.*20.>
006404  033044                          .WORD   630.+<630.*21.>
006406  034232                          .WORD   630.+<630.*22.>
006410  035420                          .WORD   630.+<630.*23.>
006412  036606                          .WORD   630.+<630.*24.>
006414  037774                          .WORD   630.+<630.*25.>
006416  041162                          .WORD   630.+<630.*26.>
006420  042350                          .WORD   630.+<630.*27.>
006422  043536                          .WORD   630.+<630.*28.>
006424  044724                          .WORD   630.+<630.*29.>
006426  046112                          .WORD   630.+<630.*30.>
006430  047300                          .WORD   630.+<630.*31.>

006432  001240                  BYTE18: .WORD   672.
006434  002500                          .WORD   672.+<672.*1.>
006436  003740                          .WORD   672.+<672.*2.>
006440  005200                          .WORD   672.+<672.*3.>
006442  006440                          .WORD   672.+<672.*4.>
006444  007700                          .WORD   672.+<672.*5.>
006446  011140                          .WORD   672.+<672.*6.>
006450  012400                          .WORD   672.+<672.*7.>
006452  013640                          .WORD   672.+<672.*8.>
006454  015100                          .WORD   672.+<672.*9.>
006456  016340                          .WORD   672.+<672.*10.>
006460  017600                          .WORD   672.+<672.*11.>
006462  021040                          .WORD   672.+<672.*12.>
006464  022300                          .WORD   672.+<672.*13.>
006466  023540                          .WORD   672.+<672.*14.>
006470  025000                          .WORD   672.+<672.*15.>
006472  026240                          .WORD   672.+<672.*16.>
006474  027500                          .WORD   672.+<672.*17.>
```

```
                                                                H  3
006476  030740                          .WORD   672.+<672.*18.>
006500  032200                          .WORD   672.+<672.*19.>
006502  033440                          .WORD   672.+<672.*20.>
006504  034700                          .WORD   672.+<672.*21.>
006506  036140                          .WORD   672.+<672.*22.>
006510  037400                          .WORD   672.+<672.*23.>
006512  040640                          .WORD   672.+<672.*24.>
006514  042100                          .WORD   672.+<672.*25.>
006516  043340                          .WORD   672.+<672.*26.>
006520  044600                          .WORD   672.+<672.*27.>
006522  046040                          .WORD   672.+<672.*28.>
006524  047300                          .WORD   672.+<672.*29.>
                                .EVEN
```

```
      0                         .SBTTL  ERROR POINTER TABLE

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

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


    006526                      $ERRTB:
  1
  2                             ;ERROR 1
  3
  4 006526  042226                      EM1             ;RH/RM INTERRUPT OCCURRED (RMAS=0)
  5 006530  043472                      DH1
  6 006532  045030                      DT1
  7 006534  045402                      DF1
  8
  9                             ;ERROR 2
 10
 11 006536  042270                      EM2             ;UNEXPECTED ATTENTION OCCURRED
 12 006540  043540                      DH2
 13 006542  045046                      DT2
 14 006544  045406                      DF2
 15
 16                             ;ERROR 3
 17
 18 006546  042326                      EM3             ;MASSBUS PARITY ERROR (MCPE=1)
 19 006550  043612                      DH3
 20 006552  045062                      DT3
 21 006554  045412                      DF3
 22
 23                             ;ERROR 4
 24
 25 006556  042364                      EM4             ;MASSBUS PARITY ERROR (PAR=1)
 26 006560  043663                      DH4
 27 006562  045076                      DT4
 28 006564  045416                      DF4
 29
 30                             ;ERROR 5
 31
 32 006566  000000                      0               ;UNUSED
 33 006570  000000                      0
 34 006572  000000                      0
 35 006574  000000                      0
 36
 37                             ;ERROR 6
 38
 39 006576  042455                      EM6             ;RH/RM DIDN'T RESPOND TO ADDRESSING
 40 006600  043746                      DH6
 41 006602  045114                      DT6
 42 006604  045422                      DF6
```

```
                                ;ERROR 7
006606  042520                          EM7             ;TRACK/SECTOR FIELD IN HEADER IS NOT CORRECT
006610  044554                          DH23
006612  045116                          DT7
006614  045512                          DF25

                                ;ERROR 10
006616  042572                          EM10            ;DRIVE OFFLINE
006620  043761                          DH10
006622  045126                          DT10
006624  045426                          DF10

                                ;ERROR 11
006626  042610                          EM11            ;PERSISTENT DRIVE UNSAFE ERROR
006630  043761                          DH10
006632  045126                          DT10
006634  045426                          DF10

                                ;ERROR 12
006636  042646                          EM12            ;UNCORRECTABLE MASSBUS PARITY ERROR
006640  043761                          DH10
006642  045126                          DT10
006644  045426                          DF10

                                ;ERROR 13
006646  042711                          EM13            ;SOFTWARE TIMEOUT
006650  043761                          DH10
006652  045126                          DT10
006654  045426                          DF10

                                ;ERROR 14
006656  042732                          EM14            ;DRIVE UNSAFE ERROR
006660  043761                          DH10
006662  045126                          DT10
006664  045426                          DF10

                                ;ERROR 15
006666  042755                          EM15            :CONTROLLER/DRIVE ERROR DURING WRITE
006670  043761                          DH10
006672  045126                          DT10
006674  045426                          DF10

                                ;ERROR 16
006676  043021                          EM16            ;CONTROLLER/DRIVE ERROR DURING WRITE CHECK
006700  043761                          DH10
006702  045126                          DT10
006704  045426                          DF10
```

```
                                ;ERROR 17
                                        EM17            ;RETRIES NOT SUCESSFUL - SECTOR NOT ACCEPTABLE
006706  043073                          DH17
006710  044210                          DT17
006712  045200                          DF17
006714  045442

                                ;ERROR 20
006716  043151                          EM20            ;DATA ERROR DURING WRITE CHECK
006720  044257                          DH20
006722  045212                          DT20
006724  045446                          DF20

                                ;ERROR 21
006726  043207                          EM21            ;CONTROLLER/DRIVE ERROR VERIFYING HEADERS
006730  043761                          DH10
006732  045126                          DT10
006734  045426                          DF10

                                ;ERROR 22
006736  043260                          EM22            ;'HCE' OR 'HCRC' ERROR VERIFYING HEADERS
006740  043761                          DH10
006742  045126                          DT10
006744  045426                          DF10

                                ;ERROR 23
006746  043330                          EM23            ;CYLINDER FIELD IN HEADER IS NOT CORRECT
006750  044554                          DH23
006752  045274                          DT23
006754  045512                          DF25

                                ;ERROR 24
006756  043400                          EM24            ;WRITE CHECK ERROR
006760  044613                          DH24
006762  045304                          DT24
006764  045472                          DF24

                                ;ERROR 25
006766  043422                          EM25            ;ILLEGAL BAD SECTOR
006770  044766                          DH25
006772  045372                          DT25
006774  045512                          DF25
```

```
                                        .SBTTL  PROGRAM START

006776  012737  000001  001376  BEGIN:  MOV     #1,CHGADR       ;ENABLE CHANGE OF 'RH/RM BUS ADDRESS'
007004  012737  177777  001270          MOV     #-1,$CDW1       ;ENABLE USE OF UTILITY ROUTINES
007012  000402                          BR      BEGIN2

007014  005037  001270          START:  CLR     $CDW1           ;DISABLE USE OF UTILITY ROUTINES

007020                          BEGIN2:
                                ;;**************************************************************************
007020  000240                  TST1:   NOP
007022  012737  000001  001214          MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
007030  005227  000000                  INC     #0              ;TTY LOOP, WAIT FOR INCREMENT
007034  001375                          BNE     .-4             ;OF WORD
007036  000005                          RESET                   ;CLEAR THE WORLD

                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
007040  012706  001114                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
007044  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
007046  022706  001154                  CMP     #SWR,R6         ;;DONE?
007052  001374                          BNE     .-6             ;;LOOP BACK IF NO
007054  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
007060  012737  023620  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
007066  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
007074  012737  027602  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
007102  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
007110  012737  027030  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
007116  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
007124  005037  001176                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
007130  112737  000001  001131          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
007136  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
007142  012737  007176  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
007150  012737  177570  001154          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
007156  012737  177570  001156          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
007164  022777  177777  171762          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
007172  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
007174  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
007176  012716  007204          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
007202  000002                          RTI
007204  012737  000176  001154  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
007212  012737  000174  001156          MOV     #DISPREG,DISPLAY
007220  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

007224  005037  001216                  CLR     $PASS           ;;CLEAR PASS COUNT
007230  132737  000200  001231          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
007236  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
007240  012737  001232  001154          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
007246                          67$:
                                .SBTTL  TYPE PROGRAM NAME
                                ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
007246  005227  177777                  INC     #-1             ;;FIRST TIME?
007252  001043                          BNE     68$             ;;BRANCH IF NO
```

```
        007254  104401  007322                  TYPE    ,69$            ;;TYPE ASCIZ STRING
                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
        007260  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
        007264  001012                          BNE     70$             ;;BRANCH IF YES
        007266  123727  001230  000001          CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
        007274  001406                          BEQ     70$             ;;BRANCH IF YES
        007276  023727  001154  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
        007304  001005                          BNE     71$             ;;BRANCH IF NO
        007306  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
        007310  000403                          BR      71$
        007312  112737  000001  001150  70$:   MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
        007320                          71$:
        007320  000420                          BR      68$             ;;GET OVER THE ASCIZ
                                        ;;69$:  .ASCIZ  <CRLF>@CZRMLA0 - RM05/3/2 FORMATTER@<CRLF>
        007362                          68$:
     16 007362  005227  177777                  INC     #-1             ;FIRST TIME THRU ?
     17 007366  001024                          BNE     1$              ;BR IF NO
     18 007370  104401  007376                  TYPE    ,72$            ;;TYPE ASCIZ STRING
        007374  000421                          BR      1$              ;;GET OVER THE ASCIZ
                                        ;;72$:  .ASCIZ  <CRLF>@PROGRAM REQUIRES 20K OF MEMORY@<CRLF>
        007440                          1$:
     22
     23                                 ;THE FOLLOWING FINDS OUT THE PROGRAM CONTROL MODE:
     24                                 ;PAPER TAPE (MANUAL), ACT11, XXDP CHAIN OR DUMP
     25
     26 007440  005037  001412                  CLR     XXDP            ;CLEAR 'XXDP' LOAD DEVICE STORAGE
     27 007444  122737  000016  000041          CMPB    #16,@#41        ;LOADED FROM AN RM05/3/2 ?
     28 007452  001160                          BNE     4$              ;BR IF NOT
     29 007454  013737  000040  001412          MOV     @#40,XXDP       ;GET DEVICE INDICATOR AND NUMBER
     30 007462  122737  000007  001412          CMPB    #7,XXDP         ;IS IT A VALID NUMBER ?
     31 007470  103002                          BHIS    2$              ;YES
     32 007472  105037  001412                  CLRB    XXDP            ;NO, DEFAULT TO DRIVE 0
     33 007476  005737  000042          2$:     TST     @#42            ;CHAIN MODE OR ACT11 AUTO ACCEPT ?
     34 007502  001425                          BEQ     3$              ;BR IF NEITHER
     35 007504  104401  007512                  TYPE    ,74$            ;;TYPE ASCIZ STRING
        007510  000412                          BR      73$             ;;GET OVER THE ASCIZ
                                        ;;74$:  .ASCIZ  <CRLF>/NOT TESTING DRIVE /
        007536                          73$:
     36 007536  005046                          CLR     -(SP)           ;CLEAR WORD ON STACK
     37 007540  113716  001412                  MOVB    XXDP,(SP)       ;GET DRIVE ADDRESS
     38 007544  104403                          TYPOS                   ;TYPE THE ADDRESS
     39 007546     001                          .BYTE   1               ;ONLY 1 CHARACTER
     40 007547     000                          .BYTE   0               ;SUPRESS LEADING ZEROS
     41 007550  104401  001205                  TYPE    ,$CRLF          ;CR-LF
     42 007554  000517                          BR      4$              ;GET NUMBER OF DRIVES
     43
     44 007556  005227  177777          3$:     INC     #-1             ;FIRST TIME THRU HERE ?
     45 007562  001114                          BNE     4$              ;NO
     46 007564  104401  007572                  TYPE    ,76$            ;;TYPE ASCIZ STRING
        007570  000410                          BR      75$             ;;GET OVER THE ASCIZ
                                        ;;76$:  .ASCIZ  <CRLF>/TO TEST DRIVE /
        007612                          75$:
     47 007612  005046                          CLR     -(SP)           ;CLEAR WORD ON STACK
     48 007614  113716  001412                  MOVB    XXDP,(SP)       ;GET DRIVE ADDRESS
     49 007620  104403                          TYPOS                   ;TYPE DRIVE ADDRESS
     50 007622     001                          .BYTE   1               ;ONLY 1 CHARACTER
     51 007623     000                          .BYTE   0               ;SUPRESS LEADING ZEROS
```

```
                                                TYPE    ,78$            ;;TYPE ASCIZ STRING
                                                BR      77$             ;;GET OVER THE ASCIZ
                                        ;;78$:  .ASCIZ  /, HALT PROGRAM, REMOVE RRDP PACK AND REPLACE IT/<CRLF>
                                        77$:
                                                TYPE    ,79$            ;;TYPE ASCIZ STRING
                                                BR      4$              ;;GET OVER THE ASCIZ
                                        ;;79$:  .ASCIZ  /WITH A WORK PACK, CLEAR LOCATION 40 AND RESTART PROGRAM./<CRLF>
                                        4$:
                                                MOV     #START,CNTLC    ;'CONTROL C' ADDRESS
                                                JSR     PC,$TKINT       ;TURN ON THE KEYBOARD INTERRUPT

                                                TSTB    $AUTOB          ;RUNNING AUTO MODE ?
                                                BNE     5$              ;BR IF YES
                                                JSR     PC,BUSADR       ;CHECK THE RH/RM ADDRESS
                                                MOV     $RMADR,RMADR    ;RH/RM ADDRESS
                                                MOV     $RMVEC,RMVEC    ;RH/RM VECTOR ADDRESS
                                        5$:
                                                TSTB    $ENV            ;RUN UNDER APT MODE?
                                                BEQ     SETVEC          ;NO, DO NOT LOAD FROM E-TABLE
                                                TSTB    $VECT1          ;UUT INTERRUPT VECTOR
                                                BEQ     6$              ;DO NOT CHANGE IF = 0
                                                MOVB    $VECT1,$RMVEC   ;NEW VALUE
                                                MOV     $RMVEC,RMVEC    ;RH/RM VECTOR
                                        6$:     TST     $BASE           ;UUT BASE REGISTER ADDRESS
                                                BEQ     SETVEC          ;DO NOT CHANGE IF = 0
                                                MOV     $BASE,$RMADR    ;NEW BASE ADDRESS
                                                MOV     $RMADR,RMADR    ;RH/RM ADDRESS


                                        ;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 RM DRIVER
                                                MOV     #-1,SAVEFG      ;SET THE SAVE REGISTERS FLAG
                                                INC     #-1             ;FIRST TIME THRU HERE ?
                                                BEQ     1$              ;BR IF NO
                                                BIT     #SW02,@SWR      ;TYPEOUT THE DRIVE STATUS TABLE ?
                                                BNE     12$             ;BR IF NO
                                        1$:     MOV     #PR7,PS         ;DON'T ALLOW RM INTERRUPT
                                                CLR     DEVMP           ;CLEAR DEVICE MAP
                                                CLR     R4              ;DRIVE TABLE POINTER
                                                TYPE    ,UNSTAT         ;TYPE 'UNIT STATUS'
                                        2$:     TYPE    ,$CRLF          ;CR-LF
                                                MOV     R4,-(SP)        ;;SAVE R4 FOR TYPEOUT
                                                                        ;;TYPE DRIVE NUMBER
                                                TYPOS                   ;;GO TYPE--OCTAL ASCII
                                                .BYTE   2               ;;TYPE 2 DIGIT(S)
                                                .BYTE   0               ;;SUPPRESS LEADING ZEROS
                                                TYPE    ,BLNKS4         ;TYPE 4 SPACES
                                                TSTB    DRVSTA(R4)      ;CHECK DRIVE'S STATUS
                                                BMI     5$              ;BR IF UNSAFE
                                                BNE     6$              ;BR IF ONLINE

                                                TSTB    DRVTYP(R4)      ;SEE IF OFFLINE OR NONEXISTENT
                                                BEQ     3$              ;BR IF NONEXISTENT
                                                BPL     4$              ;BR IF OFFLINE
                                                TYPE    ,NOTRM          ;DRIVE NOT AN RM05/3/2
```

```
                                        BR      11$             ;CHECK NEXT DRIVE

010246  104401  036270        3$:     TYPE    ,NOTPRS         ;DRIVE NOT PRESENT
010252  000452                        BR      11$             ;CHECK NEXT DRIVE

010254  104401  036226        4$:     TYPE    ,UNTOFF         ;DRIVE OFFLINE
010260  000421                        BR      8$              ;PRINT DRIVE TYPE

010262  104401  036305        5$:     TYPE    ,NOTSAF         ;DRIVE UNSAFE
010266  000416                        BR      8$              ;PRINT DRIVE TYPE

010270  005737  001412        6$:     TST     XXDP            ;LOADED FROM THIS DEVICE ?
010274  001406                        BEQ     7$              ;BR IF NO
010276  123704  001412                CMPB    XXDP,R4         ;LOADED FROM THIS DRIVE ?
010302  001003                        BNE     7$              ;BR IF NO
010304  104401  036315                TYPE    ,LODEV          ;DRIVE IS LOAD DEVICE
010310  000433                        BR      11$

010312  104401  036237        7$:     TYPE    ,UNTON          ;DRIVE ONLINE
010316  156437  030022  001402        BISB    ATABIT(R4),DEVMP        ;SET DEVICE MAP
010324  104401  036425        8$:     TYPE    ,BLNKS2         ;TYPE 2 SPACES
010330  005000                        CLR     R0
010332  116400  027716                MOVB    DRVTYP(R4),R0   ;GET DRIVE TYPE
010336  012737  036342  010376        MOV     #$RM03,10$      ;ASSUME ADDRESS OF RM03 MESSAGE
010344  122700  000004                CMPB    #4,R0           ;I  DEVICE AN RM03 ?
010350  001411                        BEQ     9$              ;BR IF YES
010352  012737  036335  010376        MOV     #$RM02,10$      ;ADDRESS OF RM02 MESSAGE
010360  122700  000005                CMPB    #5,R0           ;IS DEVICE AN RM02 ?
010364  001403                        BEQ     9$              ;BR IF YES
010366  012737  036347  010376        MOV     #$RM05,10$      ;ADDRESS OF RM05 MESSAGE

010374  104401                9$:     TYPE                    ;TYPE THE DRIVE TYPE MESSAGE
010376  000000                10$:    .WORD   0               ;MESSAGE ADDRESS HERE

010400  005204                11$:    INC     R4              ;INCREMENT DRIVE NUMBER/TABLE POINTER
010402  020427  000010                CMP     R4,#8.          ;FINISHED ?
010406  001275                        BNE     2$              ;BR IF NOT
010410  104401  001205        12$:    TYPE    ,$CRLF          ;CR-LF

010414  105737  001150                TSTB    $AUTOB          ;RUNNING UNDER AUTO MODE ?
010420  001566                        BEQ     M1              ;BR IF NO

                              ;PROGRAM IS RUNNING IN AUTO MODE - SEE IF SIZING IS ALLOWED
010422                        XSIZE:
010422  132737  000200  001231        BITB    #BIT7,$ENVM     ;APT AUTO SIZING ALLOWED ?
010430  001162                        BNE     M1              ;BR IF NO

010432  005737  001266                TST     $DEVM           ;DEVICE MAP AVAILABLE FROM APT MAIL BOX
010436  001406                        BEQ     1$              ;BR IF NONE
010440  013737  001266  001402        MOV     $DEVM,DEVMP     ;LOAD IT FROM MAIL BOX
010446  042737  177400  001402        BIC     #177400,DEVMP   ;ALLOW MAX 8 DRIVES
010454  005737  001402        1$:     TST     DEVMP           ;CHECK DEVICE MAP
010460  001002                        BNE     M5FX            ;BR IF DRIVES ASSIGNED
010462  000137  016342                JMP     $EOP            ;OTHERWISE,EXIT

010466  012737  000000  001334  M5FX: MOV     #0,BEGTRK       ;ALWAYS STARTS FROM 0 TRACK
010474  012737  000000  001330        MOV     #0,BEGCYL       ;ALWAYS STARTS FROM 0 CYLINDER
```

```
                                                ;LAST CYLINDER
                                                ;SEEK FROM LO TO HI CYLINDER ADDRESS
                                                ;DON'T ALLOW UPDATE OF BAD SECTOR FILE
                                                ;SET FORMAT AND VERIFY MODE
                                                ;APT PARAMETER = 0
                                                ;YES,DEFAULT TO -1
                                                ;NO, SET TO CHECK MODE
                                                ;SET 32 SECTOR
                                                ;APT PARAMETER =0
                                                ;NO,30 SECTOR
                                                ;COUNT OF WORDS
                                                        ;WORD COUNT
                                                ;MAX 32 SECTOR

                                                ;COUNT OF WORDS
                                                        ;WORD COUNT
                                                ;30 SECTOR
                                                ;MAX 30 SECTOR
                                                ;CHECK MODE ?
                                                ;NO
                                                ;INITIATE PATTERN
                                                ;INITIATE PATTERN
                                                ;WORSE CASE
                                                ;START WITH DRIVE NUMBER 0
                                                        ;IS THIS DRIVE IN DEVICE MAP ?
                                                ;BR IF YES
                                                ;INCREMENT DRIVE #
                                                ;ALL DONE WITH MAP ?
                                                ;BR IF NO
                                                ;DONE !!

                                                ;LOAD R1 INTO DRIVE
                                                        ;CLEAR THE BIT FROM DEVICE MAP
                                                ;IS DRIVE ON LINE ?
                                                ;BR IF NO
                                                ;LOADED FROM THIS DEVICE ?
                                                ;BR IF NO
                                                ;LOADED FROM THIS DRIVE ?
                                                ;BR IF YES

                                                ;SETUP TRACK PARAMETER FOR AN RM03/2
                                                ;IS DEVICE AN RM05 ?
                                                ;BR IF NO
                                                ;SETUP TRACK PARAMETER FOR AN RM05
                                                ;SETUP LAST TRACK AND
                                                ;END TRACK FOR AUTO MODE.
                                                ;START TO FORMAT OR CHECK

                                                ;END OF PASS FOR ALL DRIVES ON $DEVM


                                                ;SEE WHICH MODE THE PROGRAM IS TO BE RUN IN.
                                                ;MODES ARE: 'FORMAT & VERIFY', 'CHECK FORMAT' OR 'VERIFY HEADER FORMAT'

                                                ;SET MODE TO 'CHECK FORMAT'
```

```
                                                    ;TYPE 'PROGRAM MODE'
                                                    ;READ THE KEYBOARD
                                                    ;GET CHARACTER INPUT
                                                    ;'CR' ENTERED (DEFAULT)
                                                    ;BR IF YES
                                                    ;TYPE CHARACTER
                                                    ;'CHECK' ?
                                                    ;BR IF YES
                                                    ;'FORMAT' ?
                                                    ;BR IF YES
                                                    ;'VERIFY' ONLY ?
                                                    ;BR IF YES
                                                    ;NO CORRECT ENTRY
                                                    ;TRY AGAIN
                                                    ;TYPE REST OF 'CHECK'
                                                    ;GET STARTING ADDRESS

                                                    ;TYPE 'VERIFY'
                                                    ;SET VERIFY ONLY MODE

                                                    ;TYPE 'F'
                                                    ;TYPE 'FORMAT & VERIFY'
                                                    ;SET MODE TO FORMAT & VERIFY

;FIND OUT IF FORMAT IS TO BE IN 30 OR 32 SECTOR MODE

                                                    ;CR-LF
                                                    ;TYPE FORMAT MODE REQUEST
                                                    ;READ KEYBOARD
                                                    ;GET CHARACTER ENTRY
                                                    ;CARRIAGE RETURN ?
                                                    ;BR IF YES
                                                    ;TYPE CHARACTER
                                                    ;IS ENTRY A 'Y' ?
                                                    ;BR IF IT IS
                                                    ;IS ENTRY A 'N' ?
                                                    ;BR IF IT IS
                                                    ;TYPE '?'
                                                    ;TRY AGAIN

                                                    ;TYPE 'Y'
                                                    ;TYPEOUT MODE SELECTED
                                                    ;32 SECTOR INDICATOR
                                                    ;TRACK SIZE IN 32 SECTOR MODE
                                                    ;2'S COMPLEMENT WORD COUNT
                                                    ;MAX SECTOR ADDRESS IN 32 SECTOR MODE
                                                    ;CONTINUE

                                                    ;TYPE OUT 30 SECTOR MODE SELECTED
                                                    ;30 SECTOR INDICATOR
                                                    ;TRACK SIZE IN 30 SECTOR MODE
                                                    ;2'S COMPLEMENT WORD COUNT
                                                    ;MAX SECTOR ADDRESS IN 30 SECTOR MODE
```

```
                                011306  012737  000000  001334  M1B:    MOV     #0,BEGTRK       ;CLEAR STARTING TRACK ADDRESS
                                011314  012737  000000  001330          MOV     #0,BEGCYL       ;CLEAR BEGINNING CYLINDER ADDRESS
                                011322  005037  001400                  CLR     WRAP            ;SEEK FROM LO TO HI CYLINDER ADDRESS
                                011326  012737  000002  001336          MOV     #2,PATSEL       ;SETUP FOR WORST CASE PATTERN
                                011334  112737  177777  005540          MOVB    #-1,FMTDPB      ;SETUP INVALID DRIVE NUMBER

                                                        ;GO FIND OUT WHAT DRIVE

                                011342  012737  020262  001374  M0:     MOV     #C.ENTR,CNTLC   ;CONTROL C ABORT ENTRANCE
                                011350  004737  025576                  JSR     PC,$TKINT       ;INITIALIZE THE TTY KEYBOARD
                                011354  005037  001126                  CLR     $ERTTL          ;CLEAR THE ERROR ACCUMULATOR
                                011360  104401  036372                  TYPE    ,MUNIT          ;ASK FOR DRIVE NUMBER
                                011364  104410                          RDCHR                   ;READ THE KEYBOARD
                                011366  012637  001174                  MOV     (SP)+,$TMPO     ;GET CHARACTER TYPED
                                011372  023727  001174  000015          CMP     $TMPO,#CR       ;CARRIAGE RETURN ?
                                011400  001420                          BEQ     1$              ;BR IF YES
                                011402  023727  001174  000056          CMP     $TMPO,#'.       ;PERIOD ?
                                011410  001414                          BEQ     1$              ;BR IF YES
                                011412  104401  001174                  TYPE    ,$TMPO          ;TYPE CHARACTER
                                011416  012701  001174                  MOV     #$TMPO,R1       ;GET ADDRESS OF ENTRY
                                011422  004537  023410                  JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
                                011426  011452                          2$                      ;ILLEGAL CHARACTER
                                011430  011442                          1$                      ;CARRIAGE RETURN
                                011432  011452                          2$                      ;" "
                                011434  011442                          1$                      ;"."
                                011436  011460                          3$                      ;DIGIT 0-7
                                011440  011452                          2$                      ;DIGIT 8-9
                                011442  005002                  1$:     CLR     R2              ;SELECT DRIVE ZERO
                                011444  104401  036403                  TYPE    ,ZERO           ;TYPE UNIT NUMBER 0
                                011450  000403                          BR      3$              ;GO SEE IF DRIVE ZERO IS THERE

                                011452  104401  001204          2$:     TYPE    ,$QUES          ;TYPE '?'
                                011456  000731                          BR      M0              ;ASK FOR DRIVE NUMBER AGAIN
                                011460  010237  001222          3$:     MOV     R2,DRIVE        ;SAVE DRIVE NUMBER

                                011464  005737  001412                  TST     XXDP            ;LOADED FROM THIS DEVICE ?
                                011470  001406                          BEQ     4$              ;BR IF NO
                                011472  123702  001412                  CMPB    XXDP,R2         ;WAS THIS THE DRIVE ?
                                011476  001003                          BNE     4$              ;BR IF NO
                                011500  104401  036574                  TYPE    ,MLODEV         ;DRIVE IS LOAD DEVICE
                                011504  000716                          BR      M0

                                011506  004737  020016          4$:     JSR     PC,ST.CLK       ;START THE CLOCK
                                011512  004737  030052                  JSR     PC,RMINIT       ;GO SEE WHAT DRIVES ARE AVAILABLE
                                011516  012737  177777  027774          MOV     #-1,SAVEFG      ;SAVE THE REGISTERS
                                011524  012737  177777  027776          MOV     #-1,SEEKFG      ;SET 'NO OPTIMIZATION' FLAG
                                011532  005037  177776                  CLR     PS              ;PRIORITY = 0
                                011536  105762  027706                  TSTB    DRVSTA(R2)      ;LOOK AT DRIVE STATUS
                                011542  003015                          BGT     7$              ;BR IF ONLINE
                                011544  001403                          BEQ     5$              ;BR IF DRIVE NOT AVAILABLE
                                011546  104401  036554                  TYPE    ,MUSDR          ;'DRIVE UNSAFE'
                                011552  000673                          BR      M0              ;GO GET DRIVE NUMBER AGAIN

                                011554  105762  027716          5$:     TSTB    DRVTYP(R2)      ;A DRIVE PRESENT?
                                011560  001003                          BNE     6$              ;BR IF SO
                                011562  104401  036500                  TYPE    ,MDRNP          ;TYPE 'DRIVE NOT PRESENT'
                                011566  000665                          BR      M0              ;GO GET DRIVE NUMBER AGAIN
```

```
                                                6$:     TYPE    ,MER11          ;'DRIVE NOT AVAILABLE'
                                                        BR      M0              ;GO GET DRIVE NUMBER AGAIN

                                                7$:     MOVB    DRIVE,FMTDPB    ;SETUP DRIVE ADDRESS
                                                        MOV     #0,BEGCYL       ;CLEAR STARTING CYLINDER ADDRESS
                                                        MOV     #822.,ENDCYL    ;SETUP END CYLINDER
                                                        MOV     #0,BEGTRK       ;CLEAR STARTING TRACK ADDRESS
                                                        MOV     #4,R0           ;GET LAST TRACK FOR AN RM03/2
                                                        CMPB    #7,DRVTYP(R2)   ;IS DEVICE AN RM05 ?
                                                        BNE     8$              ;BR IF NO
                                                        MOV     #18.,R0         ;GET LAST TRACK FOR AN RM05
                                                8$:     MOV     R0,LSTRK        ;SETUP LAST TRACK AND
                                                        MOV     R0,ENDTRK       ;END TRACK PARAMETERS
                                                        INC     R0              ;ADJUST TRACK LIMIT
                                                        MOV     #10,R1          ;GET TRACK OFFSET IN PARAMETER TABLES
                                                        MOV     R0,TABLE(R1)    ;SETUP TRACK PARAMETER LIMITS IN "TABLE"
                                                        MOV     R0,TABLE2(R1)   ;SETUP TRACK PARAMETER LIMITS IN "TABLE2"
                                                        MOV     R0,TABLE3(R1)   ;SETUP TRACK PARAMETER LIMITS IN "TABLE3"
                                                        MOV     R0,TABLE4(R1)   ;SETUP TRACK PARAMETER LIMITS IN "TABLE4"
                                                        ADD     #14,R1          ;GET OFFSET FOR ENDTRK LIMIT IN "TABLE"
                                                        MOV     R0,TABLE(R1)    ;SETUP TRACK PARAMETER LIMIT
                                                        TYPE    ,$CRLF          ;CR-LF

                                        ;GET ADDRESS LIMITS

                                                M2:     TYPE    ,ENTADR         ;'ENTER ADDRESS LIMITS'
                                                        JSR     R4,PARENT       ;ENTER THE ADDRESS LIMITS
                                                        TABLE                   ;TABLE PARAMETERS
                                                        CLR     WRAP            ;SEEK  FROM LO TO HI CYLINDER ADDRESS
                                                        CMP     ENDCYL,BEGCYL   ;SEE IF ENDING CYL EQ TO GT THAN BEGINNING
                                                        BEQ     1$              ;BR IF ON THE SAME CYLINDER
                                                        BLO     3$              ;BR IF LESS
                                                        BR      M4              ;TO CHECK DRIVE

                                                1$:     CMP     ENDTRK,BEGTRK   ;SEE IF ENDING TRACK EQ OR GT THAN BEGINNING
                                                        BHIS    M4              ;BR IF YES
                                                2$:     TYPE    ,MADRER         ;INVALID ADDRESS ENTERED
                                                        BR      M2              ;TRY AGAIN

                                                3$:     MOV     #-1,WRAP        ;ALLOW SEEK FROM HI TO LO CYLINDER ADDRESS
                                                        BR      M4

                                        ;GO GET DATA PATTERN FOR FORMAT

                                                M4:     TST     MODE            ;IS IT CHECK MODE ?
                                                        BGT     M5              ;NO, VERIFY ONLY MODE
                                                        BNE     M4A             ;NO, FORMAT & VERIFY MODE
                                                        MOV     #133331,PATA    ;CHECK MODE USE WORST CASE DATA
                                                        MOV     #133331,PATB    ;INITIAL THE DATA PATTERN -SHIFT 3 BITS
                                                        BR      M5              ;START TO FORMAT AND CHECK

                                                M4A:    TYPE    ,MSELP          ;GO TYPE 'SELECT PATTERN'
                                                        RDCHR                   ;READ THE KEYBOARD
                                                        MOV     (SP)+,$TMP0     ;CHARCTER ENTRY
                                                        CMP     $TMP0,#CR       ;CARRIAGE RETURN ?
                                                        BEQ     1$              ;BR IF YES
```

```
380 012050  023727  001174  000056          CMP     $TMPO,#'.       ;PERIOD ?
381 012056  001414                          BEQ     1$              ;BR IF YES
382 012060  104401  001174                  TYPE    ,$TMPO          ;TYPE CHARACTER
383 012064  012701  001174                  MOV     #$TMPO,R1       ;ENTRY ADDRESS
384 012070  004537  023410                  JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
    012074  012130                          3$                      ;ILLEGAL CHARACTER
    012076  012110                          1$                      ;CARRIAGE RETURN
    012100  012130                          3$                      ;' '
    012102  012110                          1$                      ;'.'
    012104  012122                          2$                      ;DIGIT 0-7
    012106  012130                          3$                      ;DIGIT 8-9
385 012110  104401  036421          1$:     TYPE    ,TWO            ;TYPE '2'
386 012114  012702  000002                  MOV     #2,R2           ;WORST CASE PATTERN
387 012120  000406                          BR      4$              ;GO SAVE PATTERN
388
389 012122  020227  000002          2$:     CMP     R2,#2           ;IS # LARGER THAN 2
390 012126  101403                          BLOS    4$              ;BR IF NOT
391 012130  104401  001204          3$:     TYPE    ,$QUES          ;TYPE '?'
392 012134  000721                          BR      M4              ;RETYPE LINE
393
394 012136  104401  001205          4$:     TYPE    ,$CRLF          ;CR-LF
395 012142  010237  001336                  MOV     R2,PATSEL       ;SAVE PATTERN SELECTED
396
```

```
                                        .SBTTL  START OF FORMAT
                                        ;
                                        ;GO TYPE 'STARTING FORMAT ON DRIVE N'
012146                                  M5:
012146  012737  020302  000060                 MOV     #O.ENTR,@#TKVEC ;VECTOR FOR CONTROL-O
012154  005037  177776                        CLR     PS              ;PRIORITY = 0
                                        ;
012160  005037  001406                        CLR     EDIT            ;DON'T ALLOW UPDATE OF BAD SECTOR FILE
012164  005737  001320                        TST     MODE            ;'FORMAT', 'CHECK' OR VERIFY MODE ?
012170  003017                                BGT     3$              ;BR IF 'VERIFY ONLY' MODE
012172  001403                                BEQ     1$              ;BR IF 'CHECK' MODE
012174  104401  037452                        TYPE    ,MSFOR          ;TYPE 'STARTING FORMAT ON DRIVE N'
012200  000402                                BR      2$
                                        ;
012202  104401  037505                1$:     TYPE    ,MSCHK          ;TYPE 'STARTING CHECK ON DRIVE N'
012206                                  2$:
012206  013746  001222                        MOV     DRIVE,-(SP)     ;;SAVE DRIVE FOR TYPEOUT
                                                                        ;;TYPE DRIVE NUMBER
012212  104403                                TYPOS                   ;;GO TYPE--OCTAL ASCII
012214     001                                .BYTE   1               ;;TYPE 1 DIGIT(S)
012215     000                                .BYTE   0               ;;SUPPRESS LEADING ZEROS
012216  104401  001205                        TYPE    ,$CRLF          ;CR-LF
012222  012737  177777  001406                MOV     #-1,EDIT        ;UPDATE BAD SECTOR FILE AT END OF PASS
012230                                  3$:
                                        ;
                                        ;THIS CODE RETRIEVES THE BAD SPOT FILE FROM THE PACK (IF FORMATTED PACK)
                                        ;SET UP DPB BLOCK READ CYL 822., LAST TRACK.
                                        ;
012230  005037  001344                RETBAD: CLR     RETRY           ;CLEAR THE RETRY FLAG
012234  012737  177777  001404                MOV     #-1,PACK        ;RESET THE MFG AVAILABLE FLAG
012242  013737  001222  005540                MOV     DRIVE,FMTDPB    ;LOAD DRIVE #
012250  012737  001466  005552                MOV     #822.,FMTDPB+12 ;CYL 822
012256  113737  001324  005551                MOVB    LSTRK,FMTDPB+11 ;LAST TRACK
012264  105037  005550                        CLRB    FMTDPB+10       ;SECTOR 0
012270  012737  045774  005546                MOV     #BUFP,FMTDPB+6  ;BUFFER ADDRESS
012276  012737  157700  005544                MOV     #-<32.*258.>,FMTDPB+4   ;WORD COUNT FULL TRACK INCLUDING HEAD
012304  112737  000020  005541                MOVB    #20,FMTDPB+1    ;FMT SET
012312  112737  000143  005542                MOVB    #SETFMT,FMTDPB+2        ;SET 16 BIT MODE
012320  004037  030622                1$:     JSR     R0,RM05         ;SET THE FORMAT BIT
012324  005540                                FMTDPB
012326  000774                                BR      1$
012330  005737  005556                2$:     TST     FMTDPB+16       ;THE COMMAND IS DONE ?
012334  001775                                BEQ     2$              ;BR IF NOT
                                        ;
012336  012737  012336  001124                MOV     #.,$LPERR       ;LOOP ON ERROR ADDRESS
012344  012706  001100                        MOV     #STACK,SP       ;RESET STACK POINT
012350  112737  000173  005542                MOVB    #RDHD,FMTDPB+2  ;RELOAD THE READ HEAD AND DATA COMMAND
012356  004037  030622                3$:     JSR     R0,RM05         ;READ THE LAST TRACK
012362  005540                                FMTDPB
012364  000774                                BR      3$              ;BR IF QUEUE FAIL
012366  005737  005556                4$:     TST     FMTDPB+16       ;ALL DONE
012372  001775                                BEQ     4$              ;BR IF NOT
012374  100007                                BPL     6$              ;BR IF NO ERROR
                                        ;
                                        ;       ESCAPE  5$
                                        ;       JSR     PC,ERINDX       ;SEE WHICH ERROR
                                        ;       ERROR   10
```

```
                                        ;       ERROR   11
                                        ;       ERROR   12
                                        ;       ERROR   13
                                        ;       ERROR   14
                                        ;       ERROR   21              ;
                                        ;5$:    JSR     PC,LOP.CK       ;CHECK LOOP ON ERROR
012376  022737  000003  001344                  CMP     #3,RETRY        ;DONT' TRY OVER 3 TIMES
012404  103451                                  BLO     8$              ;
012406  005237  001344                          INC     RETRY
012412  000761                                  BR      3$              ;TRY AGAIN

012414  005037  001176          6$:     CLR     $ESCAPE         ;
012420  005037  001344                  CLR     RETRY
012424  005737  005556                  TST     FMTDPB+16       ;ANY ERROR ?
012430  100437                          BMI     8$              ;BR IF ANY
012432  022737  151466  045774          CMP     #151466,BUFP    ;ON THE LAST CYLINDER ?
012440  001033                          BNE     8$              ;BR IF NOT
012442  013700  001324                  MOV     LSTRK,R0        ;GET LAST TRACK
012446  000300                          SWAB    R0              ;MOVE TRACK TO HI BYTE AND SECTOR TO LO BYTE
012450  020037  045776                  CMP     R0,BUFP+2       ;ON LAST TRACK AND SECTOR 0
012454  001025                          BNE     8$              ;BR IF NOT
012456  005737  046000                  TST     BUFP+4          ;SERIAL NUMBER SHOULD NOT 0
012462  001003                          BNE     7$              ;BR IF NOT 0
012464  005737  046002                  TST     BUFP+6          ;SECOND WORD OF SERIAL NUMBER
012470  001417                          BEQ     8$              ;BR IF ZERO
012472  005737  046004          7$:     TST     BUFP+10         ;THE THIRD WORD MUST BE 0
012476  001014                          BNE     8$              ;BR IF NOT
012500  022737  177777  046006          CMP     #-1,BUFP+12     ;AN ALIGNMENT PACK ?
012506  001002                          BNE     .+6             ;BR IF NOT
012510  000137  017112                  JMP     REJEC1
012514  005737  046006                  TST     BUFP+12         ;A DATA PACK ?
012520  001003                          BNE     8$              ;BR IF NOT IDENTIFIED
012522  012737  000400  001404          MOV     #400,PACK       ;SET MFG BAD SPOT FILE AVAILABLE FLAG
012530  000240                  8$:     NOP                     ;DONE

                                ;THIS CODE INITIALIZES THE BAD16,BAD18,USTAB,USSAV TABLES
                                ;
012532  012706  001100          TABINT: MOV     #STACK,SP       ;INITIAL STACK POINT
012536  005046                          CLR     -(SP)           ;TERMINATOR
012540  012746  004430                  MOV     #USSAV,-(SP)    ;USSAV TABLE ADDRESS
012544  012746  002420                  MOV     #BAD18,-(SP)    ;MFG 18 BIT FILE ADDRESS
012550  012746  001414                  MOV     #BAD16,-(SP)    ;MFG 16 BIT FILE ADDRESS
012554  012703  003424                  MOV     #USTAB,R3       ;USER BAD SPOT FILE ADDRESS
012560  012704  000004          1$:     MOV     #4,R4           ;FIRST 4 WORDS SET TO 0 TEMPERARY
012564  005023                  2$:     CLR     (R3)+
012566  005304                          DEC     R4
012570  001375                          BNE     2$              ;BR IF NOT DONE
012572  012704  000374                  MOV     #252.,R4        ;SET -1 TO ALL THE REST LOCATIONS
012576  012705  177777                  MOV     #-1,R5
012602  010523                  3$:     MOV     R5,(R3)+
012604  005304                          DEC     R4              ;DECREMENT WORD COUNT
012606  001375                          BNE     3$              ;BR IF NOT DONE
012610  012603                          MOV     (SP)+,R3        ;NEXT TABLE
012612  001362                          BNE     1$              ;LOOPING BACK IF NOT TERMINATOR


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

```
110
111 012614  005737  001404          ;TABLD:  TST     PACK            ;BAD SPOT FILE AVAILABLE FROM PACK ?
112 012620  003451                   BLE     6$              ;BR IF NOT AVALIABLE
113 012622  012702  046000           MOV     #BUFP+4,R2      ;LOAD BAD16 TABLE FIRST
114 012626  012703  001414           MOV     #BAD16,R3       ;TABLE ADDRESS
115 012632  012704  000400           MOV     #256.,R4        ;WORD COUNT
116 012636  012223                   1$:     MOV     (R2)+,(R3)+     ;LOAD THE BAD16 TABLE
117 012640  005304                   DEC     R4              ;BR IF NOT DONE
118 012642  001375                   BNE     1$
119 012644  012702  047004           MOV     #BUFP+520.,R2   ;258 X 2 + 2 X 2 + BUFP
120 012650  012703  002420           MOV     #BAD18,R3       ;TABLE ADDRESS
121 012654  012704  000400           MOV     #256.,R4        ;WORD COUNT
122 012660  012223                   2$:     MOV     (R2)+,(R3)+     ;LOAD THE BAD18 TABLE
123 012662  005304                   DEC     R4              ;DECREMENT WORD COUNT
124 012664  001375                   BNE     2$              ;BR IF NOT DONE
125 012666  012702  061054           MOV     #BUFP+4+<11.*516.>,R2   ;SECTOR 11.
126 012672  005737  001360           TST     SEC30           ;FORMAT IN 16 BIT MODE
127 012676  100402                   BMI     3$              ;BR IF IT IS
128 012700  012702  060050           MOV     #BUFP+4+<10.*516.>,R2   ;SECTOR 10.
129 012704  012703  004430           3$:     MOV     #USSAV,R3       ;LOAD THE USSAV TABLE
130 012710  012704  000400           MOV     #256.,R4        ;WORD COUNT
131 012714  012223                   4$:     MOV     (R2)+,(R3)+     ;
132 012716  005304                   DEC     R4
133 012720  001375                   BNE     4$
134 012722  062702  000004           ADD     #4,R2           ;ACCESSES THE USTAB BUFFER ADD
135 012726  012703  003424           MOV     #USTAB,R3       ;LOAD THE USTAB TABLE
136 012732  012704  000400           MOV     #256.,R4        ;WORD COUNT
137 012736  012223                   5$:     MOV     (R2)+,(R3)+     ;LOAD THE TABLE
138 012740  005304                   DEC     R4              ;DECREMENT THE COUNT
139 012742  001375                   BNE     5$              ;BR IF NOT ALL DONE
140 012744  000240                   6$:     NOP             ;EXIT
141
142                                 ;THIS CODE ASK O.P TO ENTER A SERIAL NUMBER FOR UNFORMATTED PACK
143                                 ;
144 012746  005737  001320          SERNL:  TST     MODE            ;VERIFY ONLY MODE ?
145 012752  003162                   BGT     6$              ;BR IF YES
146 012754  005737  001404           TST     PACK            ;BAD SPOT FILE AVAILABLE ?
147 012760  003124                   BGT     4$              ;BR IF AVAILABLE
148 012762  105737  001150           TSTB    $AUTOB          ;RUNNING UNDER AUTO MODE ?
149 012766  001144                   BNE     5$              ;BR IF YES
150
151 012770  004737  025576           JSR     PC,$TKINT       ;INITIALTHE KEYBOARD
152 012774  104401  001205           TYPE    ,$CRLF          ;TYPE CRLF
153 013000  104401  040022           TYPE    ,MESG1          ;''ENTER TWO WORDS FOR SN''
154 013004  104401  040101           1$:     TYPE    ,SN1            ;
155 013010  012702  000012           MOV     #10.,R2         ;MAXMUM 10 OCTAL DIGITS ALLOWED
156 013014  005037  001414           CLR     BAD16           ;CLEAR THE LSW OF THE SERIAL NUMBER
157 013020  005037  001416           CLR     BAD16+2         ;CLEAR THE MSW OF THE SERIAL NUMBER
158 013024  104411                   RDLIN                   ;READ IN THE STRING
159 013026  012601                   MOV     (SP)+,R1        ;ADDRESS OF THE READ IN STRING
160 013030  105711                   2$:     TSTB    (R1)            ;TERMINATOR OF THE INPUT STRING
161 013032  001471                   BEQ     3$              ;BR IF IT IS
162 013034  121127  000060           CMPB    (R1),#'0        ;LESS THAN ASCIZ 0 ?
163 013040  103761                   BLO     1$              ;ENTER AGAIN IF IT IS
164 013042  121127  000067           CMPB    (R1),#'7        ;HIGH THAN ASCIX 7 ?
165 013046  101356                   BHI     1$              ;ENTER AGAIN IF SO
166 013050  013746  001414           MOV     BAD16,-(SP)     ;SAVE THE LSW
```

```
167 013054  042737  100000  001414          BIC     #BIT15,BAD16    ;CLEAR THE SIGN BIT OF LSW
168 013062  006137  001414                  ROL     BAD16           ;ROTATE THE LSW FIVE TIMES
169 013066  006137  001414                  ROL     BAD16           ;TO MOVE THE MOST SIGN DIGIT
170 013072  006137  001414                  ROL     BAD16           ;TO BIT 0 TO BIT 2
171 013076  006137  001414                  ROL     BAD16           ;
172 013102  006137  001414                  ROL     BAD16           ;
173 013106  042737  177770  001414          BIC     #177770,BAD16   ;LEFT ON THE MS DIGIT OF THE LSW
174 013114  006337  001416                  ASL     BAD16+2         ;SHIFT THE MSW  LEFT ONE OCTAL DIGIT
175 013120  100731                          BMI     1$              ;ENTER AGAIN IF SIGN BIT SET
176 013122  006337  001416                  ASL     BAD16+2         ;
177 013126  100726                          BMI     1$              ;
178 013130  006337  001416                  ASL     BAD16+2         ;
179 013134  100723                          BMI     1$              ;
180 013136  063737  001414  001416          ADD     BAD16,BAD16+2   ;APPENDING THE DIGITS INTO MSW
181 013144  012637  001414                  MOV     (SP)+,BAD16     ;RESTORE THE LSW
182 013150  006337  001414                  ASL     BAD16           ;SHIFT THE LSW LEFT ONE OCTAL DIGIT
183 013154  006337  001414                  ASL     BAD16           ;
184 013160  006337  001414                  ASL     BAD16           ;
185 013164  042737  100000  001414          BIC     #BIT15,BAD16    ;CLEAR THE SIGN BIT
186 013172  111103                          MOVB    (R1),R3         ;LOAD THE CURRENT READ IN DIGITS
187 013174  042703  177770                  BIC     #177770,R3      ;LEFT ONLY ONE OCTAL DIGIT
188 013200  060337  001414                  ADD     R3,BAD16        ;APPEND THE READ IN CHARACTER TO LSW
189 013204  005201                          INC     R1              ;ADJUST THE READ IN STRING ADDRESS
190 013206  005302                          DEC     R2              ;DECREMENT ONE DIGIT COUNT
191 013210  001307                          BNE     2$              ;BR IF NOT DONE
192 013212  105711                          TSTB    (R1)            ;IF 10 DIGITS HAVE BEEN ENTERED
193 013214  001273                          BNE     1$              ;MUST BE FOLLOWED BY <CR>
194 013216  005737  001414          3$:     TST     BAD16           ;SERIAL NUMBER MUST BE NOT 0
195 013222  001003                          BNE     4$              ;BR IF NOT 0
196 013224  005737  001416                  TST     BAD16+2         ;
197 013230  001665                          BEQ     1$              ;ENTER AGAIN IF ZERO
198 013232  013737  001414  002420  4$:     MOV     BAD16,BAD18     ;LOAD ALL TABLE WITH THE SERIAL NUMBER,
199 013240  013737  001414  003424          MOV     BAD16,USTAB
200 013246  013737  001414  004430          MOV     BAD16,USSAV
201 013254  013737  001416  003426          MOV     BAD16+2,USTAB+2
202 013262  013737  001416  004432          MOV     BAD16+2,USSAV+2
203 013270  013737  001416  002422          MOV     BAD16+2,BAD18+2
204 013276  000410                          BR      6$              ;EXIT
205
206 013300  005237  004430          5$:     INC     USSAV           ;CODE FOR AUTO MODE APT,ACT,ETC.
207 013304  005237  001414                  INC     BAD16
208 013310  005237  002420                  INC     BAD18
209 013314  005237  003424                  INC     USTAB
210
211 013320  000240                  6$:     NOP                     ;DONE
212
213                                 ;THIS CODING TO PROVIDE UTILITY ROUTINE ENTRIES
214
215 013322  000240                  CALIN:  NOP
216 013324  004737  025576                  JSR     PC,$TKINT       ;INITIALIZE THE KEYBOARD
217 013330  105737  001150                  TSTB    $AUTOB          ;AUTO BYTE SET ?
218 013334  001052                          BNE     4$              ;IF SO, BRANCH
219 013336  005737  001270                  TST     $CDW1           ;ALLOW ANY UTILITY ROUTINE ?
220 013342  001447                          BEQ     4$              ;BR IF NOT
221
222 013344  104401  040415          1$:     TYPE    ,MSGBLK         ;ENTER MESSAGE BLOCK
223 013350  104401  041204          2$:     TYPE    ,MUTLTY         ;TYPE UTILITY MESSAGE
```

```
224 013354  104410                        RDCHR                   ;READ IN THE ROUTINE NUMBER
225 013356  012637  001174                MOV     (SP)+,$TMP0     ;ADDRESS OF INPUT BUFFER
226 013362  022737  000060  001174        CMP     #'0,$TMP0       ;INPUT NUMBER LESS THAN 0 ?
227 013370  101020                        BHI     3$              ;BR IF SO
228 013372  122737  000064  001174        CMPB    #'4,$TMP0       ;INPUT NUMBER LARGER THAN 4 ?
229 013400  103414                        BLO     3$              ;BR IF SO
230 013402  104401  001174                TYPE    ,$TMP0          ;TYPE CHARACTER
231 013406  104401  001205                TYPE    ,$CRLF          ;CR-LF
232 013412  013701  001174                MOV     $TMP0,R1        ;GET NUMBER
233 013416  042701  177760                BIC     #177760,R1      ;LEFT OF THREE BITS
234 013422  006301                        ASL     R1              ;WORD INDEX
235 013424  004771  006120                JSR     PC,@TABCAL(R1)  ;CALL THE DESIRED UTILITY ROUTINE
236 013430  000747                        BR      2$              ;LOOP BACK TO THE SAME POINT
237 013432                          3$:
238 013432  104401  001205                TYPE    ,$CRLF          ;CR-LF
239 013436  022737  000065  001174        CMP     #'5,$TMP0       ;TYPE WHOLE MESSAGE AGAIN ?
240 013444  001737                        BEQ     1$              ;BR IF YES
241 013446  005037  001364                CLR     DS.CYL          ;CLEAR ALL PARAMETERS
242 013452  005037  001366                CLR     DS.TRK
243 013456  005037  001346                CLR     SAVSEC
244 013462                          4$:
245
```

```
                                ST1:
                                ;;*****************************************************************
                                ;THIS CODE SETS UP THE SELECTED DATA PATTERN TO BE WRITTEN ON EACH
                                ;SECTOR IMAGE
                                
013462 000240                   SELPAT: NOP
013464 012737 020302 000060             MOV     #0.ENTR,@#TKVEC ;VECTOR FOR CONTROL-O
013472 005037 177776                    CLR     PS              ;PRIORITY = 0

013476 005737 001320            1$:     TST     MODE            ;CHECK MODE ?
013502 001432                           BEQ     3$              ;YES
013504 002402                           BLT     2$              ;NO, FORMAT & VERIFY
013506 000137 015550                    JMP     HDREAD          ;NO, VERIFY ONLY

013512 005037 001340            2$:     CLR     PATA            ;CLEAR DATA PATTERN A
013516 005037 001342                    CLR     PATB            ;CLEAR DATA PATTERN B
013522 005737 001336                    TST     PATSEL          ;SEE IF PATTERN OF ONES
013526 001420                           BEQ     3$              ;BR IF SO
013530 012737 177777 001340             MOV     #177777,PATA    ;PATA = AB
013536 012737 177777 001342             MOV     #177777,PATB    ;PATB=CD
013544 022737 000001 001336             CMP     #1,PATSEL       ;SEE IF PATTERN OF ZEROS
013552 001406                           BEQ     3$              ;BR IF SO
013554 012737 066666 001340             MOV     #066666,PATA    ;SET UP WORST CASE
013562 012737 066666 001342             MOV     #066666,PATB    ;SET UP WORST CASE
013570 013703 001352            3$:     MOV     WC,R3           ;SET UP COUNTER
013574 012700 045774                    MOV     #BUFP,R0        ;SET UP MEMORY POINTER
013600 013701 001340            4$:     MOV     PATA,R1         ;SET UP PATTERN IN R1
013604 013702 001342                    MOV     PATB,R2         ;SET UP PATTERN IN R2
013610 010120                   5$:     MOV     R1,(R0)+        ;MOV 1ST PAT INTO MEM
013612 010220                           MOV     R2,(R0)+        ;MOV 2ND PAT INTO MEM
013614 005303                           DEC     R3              ;KEEP COUNT
013616 005303                           DEC     R3              ;KEEP COUNT
013620 001373                           BNE     5$              ;DO IT AGAIN IF R3 NOT 0

                                ;THIS CODE SETS UP FOR THE ACTUAL FORMAT

013622 004737 017352            WRTRK:  JSR     PC,SETTBL       ;GO SET UP DRIVER TABLE
013626 004737 017556                    JSR     PC,SETHDR       ;GO INITIALIZE HEADERS IN THE SECTOR BUFFER IN CORE

013632 112737 000020 005541             MOVB    #20,FMTDPB+1    ;LOAD FMT16 BIT
013640 005737 001360                    TST     SEC30           ;18 BIT MODE ?
013644 001002                           BNE     1$              ;BR IF NOT
013646 105037 005541                    CLRB    FMTDPB+1        ;CLEAR THE FMT16 BIT
013652 112737 000143 005542     1$:     MOVB    #SETFMT,FMTDPB+2  ;'LOAD FORMAT' COMMAND
013660 004037 030622            2$:     JSR     R0,RM05         ;START THE COMMAND
013664 005540                           FMTDPB                  ;DPB ADDRESS
013666 000774                           BR      2$              ;QUEUE FULL RETURN
013670 005737 005556            3$:     TST     FMTDPB+16       ;LOOK FOR DONE
013674 001775                           BEQ     3$              ;LOOP UNTIL FINISHED

                                ;SET UP SECCU TABLE: BAD SPOT ON CURRENT TRACK ,RETRIEVED FROM PACK
                                ;
013676 023727 005552 001466     WRTRKX: CMP     FMTDPB+12,#822. ;LAST CYLINDER ?
013704 103406                           BLO     1$              ;BR IF NO
013706 123737 005551 001324             CMPB    FMTDPB+11,LSTRK ;LAST TRACK OF LAST CYLINDER ?
013714 103402                           BLO     1$              ;BR IF NO
```

```
                                                                N 4
     58 013716  000137  015536                    JMP     WRTRKZ          ;DON'T DESTROY LAST TRACK ON CYL 822.
     59
     60 013722  012703  005434          1$:       MOV     #SECCU,R3       ;INITILIZE THE SECCU TABLE
     61 013726  012704  000040                    MOV     #32.,R4         ;TOTAL 32 WORD MAX
     62 013732  012702  177777                    MOV     #-1,R2          ;LOAD ALL LOCATION TO -1
     63 013736  010223                    2$:     MOV     R2,(R3)+        ;
     64 013740  005304                            DEC     R4
     65 013742  001375                            BNE     2$              ;BR IF NOT DONE
     66                                     ;
     67                                     ;LOAD THE BAD SPOT INTO SECCU TABLE IF ANY
     68                                     ;
     69 013744  005737  001404          WRTRKY: TST     PACK            ;BAD SPOT FILE AVAILABLE AT ALL
     70 013750  003461                            BLE     6$              ;BR IF NOT
     71 013752  005046                            CLR     -(SP)           ;CLEAR UP THE DUMMY RETURN ADDRESS
     72 013754  012746  003434                    MOV     #USTAB+8.,-(SP) ;USER BAD SECTOR TABLE
     73 013760  012702  001414                    MOV     #BAD16,R2       ;R2 POINTER OF MFG BAD SPOT TABLE
     74 013764  005737  001360                    TST     SEC30           ;IF 18 BIT MODE CHANGE POINTER
     75 013770  100402                            BMI     1$              ;BR IF 16 BIT MODE
     76 013772  012702  002420                    MOV     #BAD18,R2       ;
     77 013776  010201                    1$:     MOV     R2,R1           ;R1: LAST ADDRESS OF BAD SPOT TABLE
     78 014000  062701  000776                    ADD     #510.,R1        ;
     79 014004  012703  005434                    MOV     #SECCU,R3       ;TABLE ADDRESS
     80 014010  012704  000076                    MOV     #62.,R4         ;LAST ADDRESS OF THE TABLE
     81 014014  060304                            ADD     R3,R4           ;
     82 014016  062702  000010                    ADD     #10,R2          ;BAD SPOT STARTS FROM THE 5TH WORD OF THE FILE
     83 014022  023712  005552          2$:       CMP     FMTDPB+12,(R2)  ;BAD SPOT ON THIS CYLINDER ?
     84 014026  001007                            BNE     3$              ;BR IF NOT
     85 014030  123762  005551  000003            CMPB    FMTDPB+11,3(R2) ;ALSO, ON THE SAME TRK ?
     86 014036  001003                            BNE     3$              ;BR IF NOT
     87 014040  116205  000002                    MOVB    2(R2),R5        ;BAD SECTOR
     88 014044  010523                            MOV     R5,(R3)+        ;WORD STORAGE
     89 014046  062702  000004          3$:       ADD     #4,R2           ;ADVANCE TO NEXT SET
     90 014052  020102                            CMP     R1,R2           ;ALL SPOT ARE CHECKED ?
     91 014054  103412                            BLO     5$              ;BR IF IT IS
     92 014056  020403                            CMP     R4,R3           ;END OF SECCU TABLE ?
     93 014060  103401                            BLO     4$              ;REJECT THE PACK
     94 014062  000757                            BR      2$              ;LOOPING BACK
     95
     96 014064  004737  020160          4$:       JSR     PC,RESTRT       ;IMPROPER MFG INFORMATION
     97 014070  012737  177777  001404            MOV     #-1,PACK        ;RESET MFG FILE AVAILABLE FLAG
     98 014076  000137  012532                    JMP     TABINT          ;AND START AGAIN
     99
    100 014102  011601                    5$:     MOV     (SP),R1         ;NEXT TABLE
    101 014104  062701  000766                    ADD     #502.,R1        ;ENDING OF THE TABLE
    102 014110  012602                            MOV     (SP)+,R2        ;BEGINNING OF THE TABLE
    103 014112  001343                            BNE     2$              ;BR IF NOT DUMMY ADD
    104 014114  000240                    6$:     NOP                     ;DONE
    105
    106                                    ;THIS CODE SORTS THE BAD SECTOR TABLE "SECCU" IN ASCENDING ORDER
    107                                    ;AND DELETES THE DUPLICATED ONES
    108                                    ;
    109 014116  012702  005434          SORT1:  MOV     #SECCU,R2       ;TOP OF THE TABLE
    110 014122  012703  005532                  MOV     #SECCU+62.,R3   ;BOTTOM OF THE TABLE
    111 014126  012704  000037                  MOV     #31.,R4         ;TOTAL 32 ELEMENT,SORT 31 TIMES
    112 014132  022712  177777                  CMP     #-1,(R2)        ;IS THE TABLE EMPTY ?
    113 014136  001451                          BEQ     6$              ;BR IF IT IS, QUICK EXIT
    114 014140  021262  000002          1$:     CMP     (R2),2(R2)      ;N TH ELEMENT : N+1 TH ELEMENT
```

```
115 014144  103430                    BLO     4$              ;BR IF SMALLER
116 014146  001406                    BEQ     2$              ;DELET N+1 TH ELEMENT, IF SAME
117                                                           ;OTHERWISE, N > N+1
118 014150  011246                    MOV     (R2),-(SP)      ;STORE N ELEMENT
119 014152  016212  000002            MOV     2(R2),(R2)      ;SWITCH N+1 ELEMENT TO N ELEMENT
120 014156  012662  000002            MOV     (SP)+,2(R2)     ;SWITCH N ELEMENT TO N+1 ELEMENT
121 014162  000421                    BR      4$              ;ADJUST TOP POINTER
122 014164  022712  177777    2$:     CMP     #-1,(R2)        ;TERMINATOR ?
123 014170  001416                    BEQ     4$              ;BR IF IT IS
124 014172  010246                    MOV     R2,-(SP)        ;SAVE TOP POINTER
125 014174  062702  000002            ADD     #2,R2           ;DELET 2(R2)
126 014200  016222  000002    3$:     MOV     2(R2),(R2)+     ;
127 014204  022702  005532            CMP     #SECCU+62.,R2   ;BOTTOM OF THE TABLE ?
128 014210  101373                    BHI     3$              ;BR IF NOT
129 014212  012737  177777  005532    MOV     #-1,SECCU+62.   ;LOAD -1 TO TABLE BOTTOM
130 014220  012602                    MOV     (SP)+,R2        ;RESTORE THE TOP POINTER
131 014222  005304                    DEC     R4              ;DECREMENT ONE ELEMENT COUNT
132 014224  001416                    BEQ     6$              ;BR IF ALL DONE
133 014226  062702  000002    4$:     ADD     #2,R2           ;INCREMENT THE TOP POINTER
134 014232  020302                    CMP     R3,R2           ;REACH THE BOTTOM ?
135 014234  001341                    BNE     1$              ;BR IF NOT
136 014236  005304            5$:     DEC     R4              ;DECREMENT ONE SORT COUNT
137 014240  001410                    BEQ     6$              ;BR IF ALL DONE
138 014242  012702  005434            MOV     #SECCU,R2       ;RESET TOP POINTER
139 014246  162703  000002            SUB     #2,R3           ;UPDATE BOTTOM POINTER
140 014252  022703  005434            CMP     #SECCU,R3       ;EXIT IF TOP = BOTTOM
141 014256  001401                    BEQ     6$              ;
142 014260  000727                    BR      1$              ;SORT THE REST ELEMENTS
143 014262  000240            6$:     NOP                     ;DONE
144 014264  005037  005536            CLR     NEXT            ;FLAG,START OF A NEW TRACK
145
146 014270  005037  001316    WRTRK1: CLR     SOFSW           ;CLEAR ERROR COUNTER
147 014274  005037  001344            CLR     RETRY           ;ZERO THE RETRY COUNTER
148 014300  105037  005550            CLRB    FMTDPB+10       ;RESTORE SECTOR
149 014304  013737  001354  005544    MOV     MWC,FMTDPB+4    ;RESTORE WC
150 014312  012737  045774  005546    MOV     #BUFP,FMTDPB+6  ;RESTORE BA
151
152                                   ;THIS CODE TO SET UP WORD COUNT AND BUFFER ADDRESS
153
154 014320  022737  177777  005536    WRTRKA: CMP     #-1,NEXT        ;END OF OPERATION ?
155 014326  001003                    BNE     1$              ;BR IF NOT
156 014330  104401  040373            TYPE    ,MHALT
157 014334  000000                    HALT                    ;OTHERWISE SOMETHING WRONG *******
158
159 014336  113737  005536  005550    1$:     MOVB    NEXT,FMTDPB+10  ;STARTING SECTOR
160 014344  012701  005434            MOV     #SECCU,R1       ;R1 POINTS TO THE BAD SECTOR TABLE
161 014350  022711  177777    2$:     CMP     #-1,(R1)        ;END OF TABLE ?
162 014354  001513                    BEQ     WRTRKD          ;BR IF IT IS
163 014356  032711  100000            BIT     #BIT15,(R1)     ;THE BAD SECTOR HAS ALREADY ACCESSED ?
164 014362  001403                    BEQ     3$              ;BR IF NOT
165 014364  062701  000002            ADD     #2,R1           ;ADJUST THE POINTER
166 014370  000767                    BR      2$              ;LOOPING BACK
167 014372  123711  005550    3$:     CMPB    FMTDPB+10,(R1)  ;BAD SPOT = STARTING SECTOR ?
168 014376  001417                    BEQ     WRTRKB          ;BR IF IT IS
169 014400  103403                    BLO     4$              ;BR IF START SEC < BAD SPOT
170 014402  104401  040373            TYPE    ,MHALT
171 014406  000000                    HALT                    ;OTHERWISE , SOMETHING WRONG
```

```
                                                ;UPDATE NEXT STARTING ADDRESS
014410  111102          4$:     MOVB    (R1),R2
014412  023702  001362          CMP     MAXSEC,R2       ;NEXT BAD SPOT IS THE LAST SECTOR?
014416  001463                  BEQ     WRTRKF          ;BR IF LAST SECTOR
014420  103471                  BLO     WRTRKD          ;BR IF LESS
                        ;       BLOS    WRTRKD          ;BR IF IT IS
014422  005202                  INC     R2              ;NEXT = BAD + 1
014424  010237  005536          MOV     R2,NEXT         ;NEXT STARTING ADDRESS
014430  052721  100000          BIS     #BIT15,(R1)+    ;MARK THE ACCESSED SECTOR
014434  000432                  BR      WRTRKC          ;TO SET UP WORD COUNT

014436  105237  005550  WRTRKB: INCB    FMTDPB+10       ;INCREMENT THE STARTING SECTOR
014442  052721  100000          BIS     #BIT15,(R1)+    ;MARK THE ACCESSED SECTOR
014446  123711  005550          CMPB    FMTDPB+10,(R1)  ;STILL ON ANOTHER BAD SPOT ?
014452  001771                  BEQ     WRTRKB          ;LOOP AGAIN
014454  123737  005550  001362  CMPB    FMTDPB+10,MAXSEC        ;IF START SECTOR > MAXSEC
014462  101402                  BLOS    1$              ;IMPROPER MFG INFORMATION
                        ;       JSR     PC,RESTRT       ;START AGAIN AND REBUILD FMG FILE
                        ;       MOV     #-1,PACK        ;RESET THE MFG FILE AVAILABLE FLAG
                        ;       JMP     TABINT          ;RESTART POINT
014464  000137  015450          JMP     HSET            ;EXIT

014470  022711  177777  1$:     CMP     #-1,(R1)        ;TO SEE IF NEXT BAD SPOT IN TABLE
014474  001443                  BEQ     WRTRKD          ;BR IF NONE
014476  111102                  MOVB    (R1),R2         ;UPDATE THE NEXT
014500  023702  001362          CMP     MAXSEC,R2       ;BAD SECTOR IS THE LAST SECTOR ?
                        ;       BLOS    WRTRKD          ;BR IF IT IS
014504  001430                  BEQ     WRTRKF
014506  103436                  BLO     WRTRKD          ;BR IF LESS
014510  005202                  INC     R2              ;BAD SEC +1
014512  010237  005536          MOV     R2,NEXT         ;NEXT STARTING ADDRESS
014516  052711  100000          BIS     #BIT15,(R1)     ;MARK THE ACCESSED SECTOR

014522  013702  005536  WRTRKC: MOV     NEXT,R2         ;CALCULATE THE WORD COUNT
014526  162702  000002          SUB     #2,R2           ;ENDSECTOR = NEXT - 2
014532  006302                  ASL     R2              ;WORD INDEX
014534  016203  006232          MOV     WCTBL(R2),R3    ; WORD CTR FOR SECTOR 0 TO ENDING SECTOR
014540  113702  005550          MOVB    FMTDPB+10,R2    ;STARTING SECTOR
014544  006302                  ASL     R2              ;WORD INDEX
014546  166203  006232          SUB     WCTBL(R2),R3    ;WORD COUNT FOR SECTOR 0 TO STARTING SECTOR
014552  062703  000402          ADD     #258.,R3                ;ADJUST ONE SECTOR
014556  005403                  NEG     R3              ;GET THE NEAGTIVE WORD COUNT
014560  010337  005544          MOV     R3,FMTDPB+4     ;LOAD THE WORD CTR INTO DPB
014564  000432                  BR      WRTRKE          ;LOAD THE STARTING ADDRESS

014566  012737  177777  005536  WRTRKF: MOV     #-1,NEXT        ;ALL DONE
014574  052711  100000          BIS     #BIT15,(R1)     ;MASK THE SECTOR
014600  005302                  DEC     R2              ;PATCH FOR THE LAST SECTOR
014602  000405                  BR      WRTRKH          ;PATCH

014604  012737  177777  005536  WRTRKD: MOV     #-1,NEXT        ;NO MORE BAD SPOT
014612  013702  001362          MOV     MAXSEC,R2       ;LAST SECTOR

014616  006302          WRTRKH: ASL     R2              ;WORD INDEX
014620  016203  006232          MOV     WCTBL(R2),R3    ;WORD CTR FOR THE ENDING SECTOR
014624  113702  005550          MOVB    FMTDPB+10,R2    ;STARTING ADDRESS
014630  006302                  ASL     R2              ;WORD INDEX
```

```
                                                        SUB     WCTBL(R2),R3    ;WORD CTR FOR THE STARTING SECTOR
                                                        ADD     #258.,R3        ;ADJUST ONE SECTOR
                                                        NEG     R3              ;NEGATIVE WORD COUNT
                                                        MOV     R3,FMTDPB+4     ;LOAD THE WORD COUNT INTO DPB
                                                        NOP                             ;DONE

                                                WRTRKE: MOVB    FMTDPB+10,R2    ;STARTING ADDRESS
                                                        ASL     R2              ;LOCATE THE BUFFER ADDRESS
                                                        MOV     ADRTBL(R2),FMTDPB+6     ;BUFFER ADDRESS

                                                WRTRK2: MOVB    #WRTHD,FMTDPB+2  ;SET WRITE HEADER & DATA COMMAND IN TBL
                                                        MOV     #.,$LPERR       ;SET UP LOOP ON ERROR ADDRESS
                                                        MOV     #STACK,SP       ;LOAD STACK POINT
                                                1$:     JSR     R0,RM05         ;GO FORMAT A TRACK
                                                        FMTDPB                  ;ADRS OF PARAMETERS - TBL
                                                        BR      1$              ;WAIT FOR QUEUE IF FULL
                                                2$:     TST     FMTDPB+16       ;WAIT FOR COMMAND TO COMPLETE
                                                        BEQ     2$              ;BR IF NOT DONE
                                                        BPL     4$              ;BR IF NO ERROR
                                                        MOV     #3$,$ESCAPE     ;;ESCAPE TO 3$ ON ERROR
                                                        JSR     PC,ERINDX       ;SEE WHICH ERROR
                                                        EMT     10
                                                        EMT     11
                                                        EMT     12
                                                        EMT     13
                                                        EMT     14
                                                        EMT     15
                                                3$:     JSR     PC,LOP.CK       ;LOOP ON THE ERROR ?
                                                        CMP     #3,RETRY        ;ERROR RETRY LIMIT ?
                                                        BEQ     4$              ;BR IF REACHED
                                                        INC     RETRY           ;COUNT THE ERROR
                                                        BR      1$              ;TRY AGAIN
                                                4$:     CLR     $ESCAPE         ;CLEAR ERROR ESCAPE ADDRESS
                                                        CLR     RETRY           ;CLEAR THE RETRY COUNTER

                                                ;CHECK THE TRACK JUST WRITTEN

                                                CKTRK:  MOVB    #WCKHD,FMTDPB+2 ;SET WRITE CHECK HEADER & DATA COMMAND IN TBL
                                                        MOV     #.,$LPERR       ;SETUP LOOP ON ERROR ADDRESS
                                                        MOV     #STACK,SP       ;LOAD STACK POINTER
                                                1$:     JSR     R0,RM05         ;GO CHECK THE TRACK JUST FORMATTED
                                                        FMTDPB                  ;ADRS OF PARAMETERS - TBL
                                                        BR      1$              ;WAIT FOR QUEUE IF FULL
                                                2$:     TST     FMTDPB+16       ;WAIT FOR COMMAND TO COMPLETE
                                                        BEQ     2$              ;BR IF NOT DONE
                                                        BPL     9$              ;BR IF NO ERROR
                                                        MOVB    RM.REG+RMDA,SAVSEC  ;GET THE SECTOR ADDRESS
                                                        BNE     3$              ;BR IF NOT SECTOR 0
                                                        MOV     MAXSEC,SAVSEC   ;RESTORE TO LAST SECTOR +1
                                                        INC     SAVSEC          ;INCREMENT TO LAST SECTOR +1
                                                3$:     DEC     SAVSEC          ;ADJUST SECTOR TO THE ONE THAT FAILED
                                                        MOV     #11$,$ESCAPE    ;;ESCAPE TO 11$ ON ERROR
                                                        JSR     PC,ERINDX       ;SEE WHICH ERROR
                                                        EMT     10
                                                        EMT     11
                                                        EMT     12
                                                        EMT     13
```

```
285 015116  104014                      EMT     14
286 015120  032737  040000  005570      BIT     #WCE,RM.REG+RMCS2  ;WRITE CHECK ERROR ?
287 015126  001005                      BNE     4$                 ;BR IF SET
288 015130  033727  005574              BIT     RM.REG+RMER1,(PC)+  ;CHECK FOR DATA ERRORS
289 015134  130620                      .WORD   DCK!OPI!DTE!HCRC!HCE!FER  ;DATA ERROR BITS
290 015136  001001                      BNE     4$                 ;BR IF SET
291 015140  104016                      EMT     16
292 015142  005737  001316      4$:     TST     SOFSW              ;RETRYING THE SECTOR ?
293 015146  001413                      BEQ     5$                 ;BR IF NOT
294 015150  012737  015422  001176      MOV     #12$,$ESCAPE       ;;ESCAPE TO 12$ ON ERROR
295 015156  004737  020570              JSR     PC,RECORD          ;RECORD THE BAD SPOT
296 015162  012737  100000  001410      MOV     #BIT15,HEADX       ;RESET THE USER BIT
297 015170  004737  021416              JSR     PC,RESET           ;RESET THE HEADER BITS
298 015174  104017                      EMT     17
299 015176  005037  001176      5$:     CLR     $ESCAPE            ;RESET ESCAPE ADDRESS
300 015202  032737  040000  005572      BIT     #ERR,RM.REG+RMDS   ;DATA ERROR ?
301 015210  001002                      BNE     6$                 ;BR IF IT IS
302 015212  104024                      EMT     24
303 015214  000401                      BR      7$                 ;CONTINUE
304 015216                  6$:
    015216  104020                      EMT     20
305 015220  013701  001346      7$:     MOV     SAVSEC,R1          ;FAILING SECTOR ADDRESS
306 015224  113737  001346  005550      MOVB    SAVSEC,FMTDPB+10   ;SECTOR ADDRESS
307 015232  006301                      ASL     R1                 ;SETUP INDEX TO ADDRESS WORDS
308 015234  016137  006132  005546      MOV     ADRTBL(R1),FMTDPB+6  ;BUFFER ADDRESS FOR FAILING SECTOR
309 015242  013701  001362              MOV     MAXSEC,R1          ;R1 ENDING SECTOR #
310 015246  005737  005536              TST     NEXT               ;WHOLE TRACK DONE ?
311 015252  100404                      BMI     8$                 ;BR IF IT IS
312 015254  013701  005536              MOV     NEXT,R1            ;LOAD CURRENT ENDING SECTOR
313 015260  162701  000002              SUB     #2,R1              ;ADJUST TWO SECTORS
314 015264  006301          8$:         ASL     R1                 ;WORD INDEX
315 015266  016102  006232              MOV     WCTBL(R1),R2       ;WORDCTR FOR ENDING SECTOR
316 015272  113701  005550              MOVB    FMTDPB+10,R1       ;STARTING ADDRESS
317 015276  006301                      ASL     R1                 ;WORD COUNT
318 015300  166102  006232              SUB     WCTBL(R1),R2       ;WORD CTR FOR STARTING SECTOR
319 015304  005402                      NEG     R2
320 015306  010237  001350              MOV     R2,SAVWC           ;SAVE THE WORD COUNT
321 015312  012737  177376  005544      MOV     #-258.,FMTDPB+4    ;WORD COUNT FOR 1 SECTOR
322 015320  005237  001316              INC     SOFSW              ;INDICATE THAT A RETRY IS IN PROGRESS
323 015324  000137  014666              JMP     WRTRK2             ;REFORMAT ERROR SECTOR
324 015330  005737  001316      9$:     TST     SOFSW              ;RETRY IN PROGRESS ?
325 015334  001432                      BEQ     12$                ;BR IF NOT
326 015336  022737  000002  001316      CMP     #2,SOFSW           ;SEE IF LAST RETRY
327 015344  001403                      BEQ     10$                ;BR IF IT IS
328 015346  005237  001316              INC     SOFSW              ;INCREMENT RETRY COUNT
329 015352  000625                      BR      1$                 ;READ AGAIN
330 015354  123737  001362  005550  10$:  CMPB  MAXSEC,FMTDPB+10   ;SEE IF LAST SECTOR ON THE TRACK
331 015362  001417                      BEQ     12$                ;BR IF IT IS
332 015364  004737  017530              JSR     PC,SCAWC           ;SETUP TO CHECK REMAINING SECTORS ON THE TRACK
333 015370  005037  001316              CLR     SOFSW              ;CLEAR RETRY COUNTER
334 015374  000614                      BR      1$                 ;FINISH CHECKING THE TRACK
335 015376  004737  017304      11$:    JSR     PC,LOP.CK          ;CHECK FOR LOOP ON ERROR
336 015402  022737  000003  001344      CMP     #3,RETRY           ;ERROR RETRY REACHED ?
337 015410  001404                      BEQ     12$                ;BR IF IT IS
338 015412  005237  001344              INC     RETRY              ;COUNT THE RETRY
339 015416  000137  015026              JMP     1$                 ;DO THE WRITE CHECK AGAIN
340 015422  005037  001176      12$:    CLR     $ESCAPE            ;CLEAR THE ERROR RETURN ESCAPE ADDRESS
```

```
                                                        ;CLEAR THE RETRY COUNTER
015426  005037  001344                    CLR     RETRY
015432  005037  001316                    CLR     SOFSW           ;CLEAR THE SOFTWARE RETRY COUNT
015436  005737  005536                    TST     NEXT            ;WHOLE TRACK DONE ?
015442  100402                            BMI     .+6             ;BR IF IT IS
015444  000137  014320                    JMP     WRTRKA

015450  032777  000002  163476  HSET:     BIT     #BIT1,@SWR      ;LOOP ON CURRENT TRACK ?
015456  001402                            BEQ     1$              ;BR IF NOT
015460  000137  013676                    JMP     WRTRKX          ;START AGAIN ON THE SAME TRACK
                                                                  ;RESET THE HEADER WORD OF ALL MFG
                                                                  ;DEFINED BAD SECTOR
015464  012737  040000  001410  1$:       MOV     #BIT14,HEADX    ;RESET THE MFG BIT OF THE 1ST HEADER WORD
015472  012704  005434                    MOV     #SECCU,R4       ;TABLE FOR BAD SECTOR ON CURRENT TRACK
015476  022714  177777          2$:       CMP     #-1,(R4)        ;END OF TABLE ?
015502  001414                            BEQ     4$              ;BR IF SO
015504  032714  040000                    BIT     #BIT14,(R4)     ;HAS THE SECTOR BE ASSCESSED ?
015510  001006                            BNE     3$              ;BR IF SO
015512  052714  040000                    BIS     #BIT14,(R4)     ;MASK THE SECTOR TABLE
015516  111437  001346                    MOVB    (R4),SAVSEC     ;SECTOR ADDRESS
015522  004737  021416                    JSR     PC,RESET        ;RESET THE HEADER
015526  062704  000002          3$:       ADD     #2,R4           ;INCREMENT THE TABLE POINTER
015532  000761                            BR      2$              ;LOOPING BACK
015534  000240                  4$:       NOP                     ;ALL DONE

015536  004737  017426          WRTRKZ:   JSR     PC,TRKTST       ;GET UPDATED TRACK AND CYLINDER ADDRESSES
015542  000402                            BR      HDREAD          ;RETURN HERE IF DONE
                                                                  ;AND DO QUICK CHECK OF DISK
015544  000137  013676                    JMP     WRTRKX          ;CONTINUE WITH THE FORMAT

                                ;THIS CODE MAKES SURE EACH CYLINDER FORMATTED CONTAINS THE
                                ;PROPER CYLINDER ADDRESS. THE PROGRAM IS LOOKING FOR
                                ;POSSIBLE POSITIONER ERRORS THAT MAY HAVE OCCURRED DURING THE FORMAT.

015550  005737  001320          HDREAD:   TST     MODE            ;CHECK MODE ?
015554  001022                            BNE     1$              ;NO
015556  022737  066666  001340            CMP     #066666,PATA    ;PATTERN HAS BEEN ROTATED TO WORST CASE ?
015564  001002                            BNE     .+6
015566  000137  016342                    JMP     $EOP            ;EXIT

015572  000241                            CLC                     ;CLEAR THE CARRY BIT
015574  006037  001340                    ROR     PATA            ;ROTATE PATTERN
015600  103003                            BCC     .+10            ;SET SIGN BIT IF CARRY=1
015602  052737  100000  001340            BIS     #BIT15,PATA     ;FROM BIT 0
015610  013737  001340  001342            MOV     PATA,PATB       ;SET BOTH PATTERN
015616  000137  013462                    JMP     ST1             ;CONTINUE THE VERIFY OPERATION

015622  005737  001320          1$:       TST     MODE            ;'FORMAT', 'CHECK' OR VERIFY MODE ?
015626  003003                            BGT     2$              ;BR IF 'VERIFY ONLY' MODE
015630  104401  037537                    TYPE    ,MSQVER         ;TYPE 'STARTING QUICK VERIFICATION'
015634  000402                            BR      3$
015636  104401  037600          2$:       TYPE    ,MSVER          ;TYPE 'STARTING VERIFICATION'
015642                            3$:
015642  013746  001222                    MOV     DRIVE,-(SP)     ;;SAVE DRIVE FOR TYPEOUT
                                                                  ;;TYPE DRIVE NUMBER
015646  104403                            TYPOS                   ;;GO TYPE--OCTAL ASCII
015650     001                            .BYTE   1               ;;TYPE 1 DIGIT(S)
015651     000                            .BYTE   0               ;;SUPPRESS LEADING ZEROS
```

```
393 015652  104401  001205                      TYPE    ,$CRLF          ;CR-LF
394 015656  005037  001356                      CLR     HEDERR          ;CLEAR HEADER CHECK ERROR INDICATOR
395 015662  004737  017352                      JSR     PC,SETTBL       ;GO SET UP DRIVER TABLE
396 015666  004737  017556                      JSR     PC,SETHDR       ;GO SET UP HEADERS IN CORE
397 015672  012700  045774            4$:       MOV     #BUFP,R0        ;GET STARTING ADDRESS OF EXPECTED HEADERS
398 015676  012701  106174                      MOV     #BUFV,R1        ;GET STARTING ADDRESS OF RECIEVED HEADERS
399 015702  010137  005546                      MOV     R1,FMTDPB+6     ;SET UP BUFFER ADDRESS
400 015706  112737  000173  005542              MOVB    #RDHD,FMTDPB+2  ;SET UP READ HEADER & DATA COMMAND
401 015714  013737  001354  005544              MOV     MWC,FMTDPB+4    ;SETUP WORD FOR VERIFY ONLY MODE
402 015722  005037  001322                      CLR     SNGSEC          ;SETUP MULTI-SECTOR READ FLAG
403 015726  005737  001320                      TST     MODE            ;VERIFY MODE ONLY ?
404 015732  003005                             BGT     6$              ;BR IF YES
405 015734  012737  177776  005544    5$:       MOV     #-2,FMTDPB+4    ;SETUP WORD COUNT FOR QUICK VERIFY
406 015742  005237  001322                      INC     SNGSEC          ;SETUP SINGLE SECTOR READ
407
408 015746  012737  015746  001124    6$:       MOV     #.,$LPERR       ;SETUP LOOP ON ERROR ADDRESS
409 015754  004037  030622            7$:       JSR     R0,RM05         ;GO READ HEADER
410 015760  005540                             FMTDPB                  ;ADRS OF PARAMETER TBL
411 015762  000774                             BR      7$              ;WAIT IF QUEUE IS FULL
412 015764  005737  005556            8$:       TST     FMTDPB+16       ;WAIT FOR COMMAND TO COMPLETE
413 015770  001775                             BEQ     8$              ;BR IF NOT DONE
414 015772  100044                             BPL     11$             ;BR IF NOT ERROR
415
416 015774  012737  016170  001176              MOV     #13$,$ESCAPE    ;;ESCAPE TO 13$ ON ERROR
417 016002  004737  017204                      JSR     PC,ERINDX       ;SEE WHICH ERROR
418 016006  104010                             EMT     10              ;DRIVE OFFLINE
419 016010  104011                             EMT     11              ;PERSISTENT DRIVE UNSAFE ERROR
420 016012  104012                             EMT     12              ;UNCORRECTABLE MASSBUS PARITY ERROR
421 016014  104013                             EMT     13              ;SOFTWARE TIMEOUT
422 016016  104014                             EMT     14              ;DRIVE UNSAFE ERROR
423
424 016020  032737  040000  005572              BIT     #ERR,RM.REG+12  ;ANY DRIVE ERRORS ?
425 016026  001426                             BEQ     11$             ;BR IF NO
426 016030  022737  000200  005574              CMP     #HCE,RM.REG+14  ;IS IT A HEADER COMPARE ERROR ?
427 016036  001411                             BEQ     9$              ;BR IF YES
428 016040  032737  000400  005574              BIT     #HCRC,RM.REG+14 ;IS IT A HEADER CRC ERROR ?
429 016046  001005                             BNE     9$              ;BR IF YES
430 016050  022737  100000  005622              CMP     #BSE,RM.REG+42  ;BAD SECTOR ERROR ?
431 016056  001406                             BEQ     10$             ;BR IF YES
432 016060  104021                             EMT     21
433 016062  005037  001176            9$:       CLR     $ESCAPE         ;CLEAR THE ERROR ESCAPE ADDRESS
434 016066  104022                             EMT     22
435 016070  004737  017304                      JSR     PC,LOP.CK       ;CHECK FOR LOOP ON ERROR
436
437 016074  000240                   10$:      NOP
438 016076  005737  001322                      TST     SNGSEC          ;ALREADY READING SINGLE SECTORS ?
439 016102  001714                             BEQ     5$              ;BR IF NO
440 016104  052711  140000            11$:      BIS     #MF!UF,(R1)     ;SET MFG AND USER SECTOR GOOD BITS
441 016110  011037  045770                      MOV     (R0),RBUF+4     ;GET EXPECTED CYLINDER HEADER
442 016114  011137  045764                      MOV     (R1),RBUF       ;GET RECIEVED CYLINDER HEADER
443 016120  023737  045764  045770              CMP     RBUF,RBUF+4     ;SEE IF CYL READ EQUALS CYL EXPECTED
444 016126  001402                             BEQ     12$             ;BR IF CYL CORRECT
445 016130  104023                             EMT     23
446 016132  000416                             BR      13$
447
448 016134  005737  001320            12$:      TST     MODE            ;VERIFY ONLY MODE ?
449 016140  003451                             BLE     17$             ;BR IF NO
```

```
                                                                H 5
 450 016142  016037  000002  045772          MOV     2(R0),RBUF+6    ;GET EXPECTED TRACK/SECTOR HEADER
 451 016150  016137  000002  045766          MOV     2(R1),RBUF+2    ;GET RECIEVED TRACK/SECTOR HEADER
 452 016156  023737  045766  045772          CMP     RBUF+2,RBUF+6   ;IS TRACK AND SECTOR CORRECT ?
 453 016164  001405                          BEQ     15$             ;BR IF YES
 454 016166  104007                          EMT     7
 455
 456 016170  004737  017304          13$:    JSR     PC,LOP.CK       ;CHECK FOR LOOP ON ERROR
 457 016174  000137  016342          14$:    JMP     $EOP            ;END OF VERIFY
 458
 459 016200  000240                  15$:    NOP
 460 016202  123737  005550 ,001362          CMPB    FMTDPB+10,MAXSEC        ;DONE WITH ALL SECTORS ?
 461 016210  001412                          BEQ     16$             ;BR IF YES
 462 016212  105237  005550                  INCB    FMTDPB+10       ;UPDATE SECTOR ADDRESS
 463 016216  062700  001004                  ADD     #516.,R0        ;ADJUST POINTER TO NEXT EXPECTED HEADER
 464 016222  005737  001322                  TST     SNGSEC          ;READING ONE SECTOR AT A TIME ?
 465 016226  001252                          BNE     7$              ;BR IF YES
 466 016230  062701  001004                  ADD     #516.,R1        ;ADJUST POINTER TO NEXT RECIEVED HEADER
 467 016234  000723                          BR      11$             ;GO COMPARE NEXT SECTOR
 468 016236  105037  005550          16$:    CLRB    FMTDPB+10       ;START AT SECTOR 0
 469 016242  105237  005551                  INCB    FMTDPB+11       ;INCREMENT TRACK NUMBER
 470 016246  123737  005551  001332          CMPB    FMTDPB+11,ENDTRK        ;WAS IT LAST TRACK ?
 471 016254  101003                          BHI     17$             ;BR IF YES
 472 016256  004737  017756                  JSR     PC,UPDTRK       ;UPDATE TRACK ADDRESS IN BUFFER
 473 016262  000603                          BR      4$              ;GO READ NEXT TRACK
 474
 475 016264  113737  001334  005551  17$:    MOVB    BEGTRK,FMTDPB+11        ;GET BEGINNING TRACK ADDRESS
 476 016272  005737  001400                  TST     WRAP            ;SEEKING FROM HI TO LO CYLINDERS ?
 477 016276  001407                          BEQ     18$             ;BR IF NO
 478 016300  005337  005552                  DEC     FMTDPB+12       ;DECREMENT CYLINDER ADDRESS
 479 016304  023737  005552  001326          CMP     FMTDPB+12,ENDCYL        ;DONE WITH VERIFY ?
 480 016312  002730                          BLT     14$             ;BR IF YES
 481 016314  000406                          BR      19$
 482 016316  005237  005552          18$:    INC     FMTDPB+12       ;INCREMENT CYLINDER ADDRESS BEING CHECKED
 483 016322  023737  005552  001326          CMP     FMTDPB+12,ENDCYL        ;DONE WITH VERIFY ?
 484 016330  101321                          BHI     14$             ;BR IF YES
 485 016332  004737  017644          19$:    JSR     PC,UPDCYL       ;UPDATE CYLINDER ADDRESS IN BUFFER
 486 016336  000137  015672                  JMP     4$              ;GO READ NEXT CYLINDER
 487
 567
 568                                .SBTTL  END OF PASS ROUTINE

                                    ;;*******************************************************************
                                    ;*INCREMENT THE PASS NUMBER ($PASS)
                                    ;*IF THERES A MONITOR GO TO IT
                                    ;*IF THERE ISN'T JUMP TO DONE

     016342                         $EOP:
     016342  005737  001406          1$:     TST     EDIT            ;UPDATE LAST TRACK ?
     016346  001537                          BEQ     8$              ;BR IF NO
     016350  005037  001344                  CLR     RETRY           ;CLEAR RETRY COUNT
     016354  004537  020710                  JSR     R5,SORT2        ;SORT THE USTAB IN ASCENDING FORM
     016360  003434                          USTAB+8.                ;TABLE ENTRY POINT
     016362  004737  021050                  JSR     PC,TEXT         ;SET UP THE BUFFER
     016366  112737  000020  005541          MOVB    #20,FMTDPB+1    ;16 BIT MODE
     016374  112737  000143  005542          MOVB    #SETFMT,FMTDPB+2        ;SET FORMAT COMMAND
     016402  004037  030622          2$:     JSR     R0,RM05         ;CALL THE DRIVER
     016406  005540                          FMTDPB
```

```
016410  000774                          BR      2$                      ;LOOP IF QUEQU FAILS
016412  005737  005556          3$:     TST     FMTDPB+16               ;ALL DONE ?
016416  001775                          BEQ     3$                      ;BR IF NOT

016420  012737  001466  005552          MOV     #822.,FMTDPB+12 ;CYLINDER ADDRESS
016426  113737  001324  005551          MOVB    LSTRK,FMTDPB+11 ;TRACK NUMBER
016434  105037  005550                  CLRB    FMTDPB+10               ;SECTOR 0
016440  012737  157700  005544          MOV     #-<258.*32.>,FMTDPB+4   ;WORD COUNT
016446  012737  045774  005546          MOV     #BUFP,FMTDPB+6  ;BUFFER ADDRESS
016454  112737  000163  005542          MOVB    #WRTHD,FMTDPB+2 ;WRITE HEAD AND DATA COMMAND
016462  000240                  4$:     NOP
016464  004037  030622                  JSR     R0,RM05                 ;CALL THE DRIVER
016470  005540                          FMTDPB
016472  000773                          BR      4$                      ;LOOP IF QUEUE FAILS
016474  005737  005556          5$:     TST     FMTDPB+16               ;ALL DONE ?
016500  001775                          BEQ     5$                      ;BR IF NOT
016502  100061                          BPL     8$                      ;EXIT IF NO ERROR

016504  012737  016532  001176          MOV     #6$,$ESCAPE             ;ERROR ESCAPE ADDRESS
016512  004737  017204                  JSR     PC,ERINDX               ;ERROR INDICATOR RT.
016516  104010                          EMT     10
016520  104011                          EMT     11
016522  104012                          EMT     12
016524  104013                          EMT     13
016526  104014                          EMT     14
016530  104015                          EMT     15
016532  000240                  6$:     NOP
016534  005237  001344                  INC     RETRY                   ;
016540  022737  000003  001344          CMP     #3,RETRY                ;OVER 3 TIMES ?
016546  101345                          BHI     4$                      ;BR IF IT IS
016550  000137  017162                  JMP     REJCT3                  ;WRITE THE LAST TRACK FAILS THREE TIMES
016554  122737  000037  005550          CMPB    #31.,FMTDPB+10  ;LAST SECTOR ?
016562  003431                          BLE     8$                      ;BR IF ALL SET
016564  113737  005566  001346          MOVB    RM.REG+RMDA,SAVSEC      ;FOUND THE FALLING SPOT
016572  001003                          BNE     7$                      ;BR IF NOT 0
016574  012737  000037  001346          MOV     #31.,SAVSEC     ;THE LAST SECTOR IS A BAD SECTOR
016602  113737  001346  005550  7$:     MOVB    SAVSEC,FMTDPB+10        ;LOAD THE NEW STARTING SECTOR
016610  113701  001346                  MOVB    SAVSEC,R1               ;LOAD THE WORD CTR AND BUFFER
016614  006301                          ASL     R1                      ;WORD INDEX
016616  016137  006132  005546          MOV     ADRTBL(R1),FMTDPB+6     ;BUFFER ADDRESS
016624  012737  157700  005544          MOV     #-<258.*32.>,FMTDPB+4   ;WORD COUNT
016632  066137  006232  005544          ADD     WCTBL(R1),FMTDPB+4      ;WORD COUNT
016640  005037  001344                  CLR     RETRY                   ;RESET THE RETRY FLAG
016644  000706                          BR      4$                      ;BR BACK

016646  005737  001320          8$:     TST     MODE                    ;FORMAT, CHECK OR VERIFY MODE
016652  003007                          BGT     10$                     ;BR IF VERIFY ONLY MODE
016654  001403                          BEQ     9$                      ;BR IF CHECK
016656  104401  037633                  TYPE    ,MFCMPT                 ;FORMAT COMPLETE
016662  000405                          BR      11$
016664  104401  037656          9$:     TYPE    ,MCCMPT                 ;CHECK COMPLETE
016670  000402                          BR      11$
016672  104401  037700          10$:    TYPE    ,MVCMPT                 ;VERIFY COMPLETE
016676  104401  037723          11$:    TYPE    ,NUMERR                 ;TOTAL ERRORS
016702  013746  001126                  MOV     $ERTTL,-(SP)            ;;SAVE $ERTTL FOR TYPEOUT
016706  104405                          TYPDS                           ;;GO TYPE--DECIMAL ASCII WITH SIGN
016710  005037  001126                  CLR     $ERTTL                  ;CLEAR THE ERROR ACCUMULATOR
```

```
        016714  104401  037753                  TYPE    ,STARS
        016720  005737  001402                  TST     DEVMP           ;ALL ASSIGNED DEVICE DONE ?
        016724  001030                          BNE     DONE            ;TO FORMAT NEXT DRIVE
        016726  005337  001216                  DEC     $PASS           ;RESTORE THE PASS COUNT
        016732  000240                  12$:    NOP
        016734  005237  001216                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
        016740  042737  100000  001216          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
        016746  005327                          DEC     (PC)+           ;;LOOP?
        016750  000001                  $EOPCT: .WORD   1
        016752  003013                          BGT     $DOAGN          ;;YES
        016754  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
        016756  000001                  $ENDCT: .WORD   1
        016760  016750                          $EOPCT
        016762  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
        016766  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
        016770  000005                          RESET                   ;;CLEAR THE WORLD
        016772  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
        016774  000240                          NOP                     ;;SAVE ROOM
        016776  000240                          NOP                     ;;FOR
        017000  000240                          NOP                     ;;ACT11
        017002                          $DOAGN:
        017002  000137                          JMP     @(PC)+          ;;RETURN
        017004  017006                  $RTNAD: .WORD   DONE
   569
   570  017006  105737  001150          DONE:   TSTB    $AUTOB          ;RUNNING IN AUTO MODE ?
   571  017012  001405                          BEQ     13$             ;NO
   572  017014  005737  001402                  TST     DEVMP           ;ARE ALL DRIVES DONE ?
   573  017020  001427                          BEQ     16$             ;BR IF YES
   574  017022  000137  010466                  JMP     M5FX            ;NEXT DRIVE FROM APT DEVICE MAP
   575
   576  017026  032777  000001  162120  13$:    BIT     #SW0,@SWR       ;SEE IF SWR 0 SET
   577  017034  001002                          BNE     14$             ;BR IF SET
   578  017036  000137  011342                  JMP     M0              ;ASK FOR NEXT DRIVE
   579
   580  017042  012706  001100          14$:    MOV     #STACK,SP       ;RESET STACK POINTER
   581  017046  005737  001320                  TST     MODE            ;CHECK MODE ?
   582  017052  001402                          BEQ     15$             ;YES
   583  017054  000137  013462                  JMP     ST1             ;DO THE FORMAT OR CHECK AGAIN
   584
   585  017060  012737  133331  001340  15$:    MOV     #133331,PATA    ;INITIATE PATTERN
   586  017066  012737  133331  001342          MOV     #133331,PATB    ;INITIATE PATTERN
   587  017074  000137  013462                  JMP     ST1             ;LOOP ON CHECK
   588
   589  017100  005237  001216          16$:    INC     $PASS           ;GARBAGE CODE FOR APT
   590  017104  005237  001214          17$:    INC     $TESTN
   591  017110  000775                          BR      17$
   592
   593  017112  104401  001205          REJEC1: TYPE    ,$CRLF
   594  017116  104401  040354                  TYPE    ,MESG4
   595  017122  104401  040373                  TYPE    ,MHALT
   596  017126  000000                          HALT
   597  017130  000137  007020                  JMP     BEGIN2
   598
   599  017134  104401  001205          REJCT2: TYPE    ,$CRLF
   600  017140  113737  005551  001366          MOVB    FMTDPB+11,DS.TRK  ;TRACK NUMBER FOR ERROR LOG
   601  017146  104025                          EMT     25
   602  017150  104401  040373                  TYPE    ,MHALT
```

```
                                        HALT
                                        JMP     BEGIN2

                                REJCT3: TYPE    ,$CRLF
                                        TYPE    ,MESG17
                                        TYPE    ,MHALT
                                        HALT
                                        JMP     BEGIN2

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

                                .SBTTL  SUPPORT SUBROUTINES

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

                                ;THIS ROUTINE DETERMINES THE ERROR TYPE

                                ERINDX: MOV     R1,-(SP)        ;STORE R1
                                        CLR     R1              ;CLEAR R1
                                        BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
                                        .WORD   BIT14!BIT13!BIT2!BIT1  ;DRIVE OFFLINE
                                        BNE     5$              ;BR IF OFFLINE
                                        BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
                                        .WORD   BIT12           ;DRIVE PERSISTENTLY UNSAFE
                                        BNE     4$              ;BR IF UNSAFE
                                        BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
                                        .WORD   BIT11!BIT10     ;UNCORRECTABLE MASSBUS PARITY ERROR ?
                                        BNE     3$              ;BR IF PARITY ERROR
                                        BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
                                        .WORD   BIT09!BIT08     ;SOFTWARE TIMEOUT ?
                                        BNE     2$              ;BR IF YES
                                        BIT     FMTDPB+16,(PC)+ ;CHECK ERROR TYPE
                                        .WORD   BIT04           ;DRIVE UNSAFE ERROR ?
                                        BNE     1$              ;BR IF YES
                                        INC     R1              ;INCREMENT THE RETURN INDEX
                                1$:     INC     R1              ;INCREMENT THE RETURN INDEX
                                2$:     INC     R1              ;INCREMENT THE RETURN INDEX
                                3$:     INC     R1              ;INCREMENT THE RETURN INDEX
                                4$:     INC     R1              ;INCREMENT THE RETURN INDEX
                                5$:     ASL     R1              ;DOUBLE THE INCREMENT
                                        ADD     R1,2(SP)        ;DEVELOP THE RETURN ADDRESS
                                        MOV     (SP)+,R1        ;RESTORE R1
                                        RTS     PC              ;RETURN

                                ;ROUTINE TO CHECK FOR LOOP ON ERROR

                                LOP.CK: BIT     #SW09,@SWR      ;LOOP ON ERROR ?
                                        BEQ     1$              ;BR IF NOT
                                        JMP     @$LPERR         ;GO TO THE LOOP ON ERROR ADDRESS

                                1$:     CLR     $ESCAPE         ;CLEAR THE ERROR ESCAPE ADDRESS
                                        BIT     FMTDPB+16,(PC)+ ;CHECK FOR 'FATAL' ERROR
                                        .WORD   BIT14!BIT13!BIT12!BIT10!BIT01  ;'FATAL' ERROR BITS
                                        BNE     2$              ;BR IF NOT
                                        BIT     #WLE,RM.REG+RMER1  ;WRITE LOCK ERROR ?
                                        BEQ     3$              ;BR IF NOT
                                2$:     JMP     $EOP            ;TERMINATE THE FORMAT
```

```
                                ;THIS ROUTINE SETS UP THE INPUT TABLE FOR THE DRIVER ROUTINE

017352  113737  001222  005540  SETTBL: MOVB    DRIVE,FMTDPB    ;SET UP DRIVE #
017360  105037  005543          CLRB    FMTDPB+3        ;CLEAR HIGH ORDER ADRS BITS
017364  013737  001354  005544  MOV     MWC,FMTDPB+4    ;LOAD UP WORD COUNT
017372  012737  045774  005546  MOV     #BUFP,FMTDPB+6  ;LOAD UP CURRENT ADRS
017400  105037  005550          CLRB    FMTDPB+10       ;SET SECTOR TO ZERO
017404  113737  001334  005551  MOVB    BEGTRK,FMTDPB+11  ;SET UP STARTING TRK ADRS
017412  013737  001330  005552  MOV     BEGCYL,FMTDPB+12  ;SET UP STARTING CYL
017420  005037  005556          CLR     FMTDPB+16       ;CLEAR RM STATUS
017424  000207                  RTS     PC              ;RETURN FROM SETUP

                                ;THIS ROUTINE CONTROLS THE DISK ADDRESSING AND TOTAL TRK COUNT
                                ;IT IS ENTERED AFTER EVERY TRK OPERATION

017426                          TRKTST:
017426  105237  005551                  INCB    FMTDPB+11       ;INCREMENT TRACK NUMBER
017432  123737  005551  001332          CMPB    FMTDPB+11,ENDTRK        ;LAST TRACK TO BE FORMATTED ON THIS CYLINDER ?
017440  101003                  BHI     1$              ;BR IF YES
017442  004737  017756                  JSR     PC,UPDTRK       ;UPDATE TRACK ADDRESS IN BUFFER
017446  000425                  BR      4$              ;EXIT

017450  113737  001334  005551  1$:     MOVB    BEGTRK,FMTDPB+11        ;GET BEGINNING TRACK ADDRESS
017456  005737  001400          TST     WRAP            ;SEEKING FROM HI TO LO CYLINDER ?
017462  001407                  BEQ     2$              ;BR IF NO
017464  005337  005552          DEC     FMTDPB+12       ;DECREMENT CYLINDER ADDRESS
017470  023737  005552  001326          CMP     FMTDPB+12,ENDCYL        ;DONE WITH FORMATTING ?
017476  002413                  BLT     5$              ;BR IF YES
017500  000406                  BR      3$
017502  005237  005552  2$:     INC     FMTDPB+12       ;INCREMENT CYLINDER ADDRESS
017506  023737  005552  001326          CMP     FMTDPB+12,ENDCYL        ;DONE WITH FORMATTING ?
017514  101004                  BHI     5$              ;BR IF YES
017516  004737  017644  3$:     JSR     PC,UPDCYL       ;UPDATE CYLINDER ADDRESS IN BUFFER

017522  062716  000002  4$:     ADD     #2,(SP)         ;ADD TWO TO RETURN ADRS (NOT DONE)
017526  000207          5$:     RTS     PC              ;RETURN, DONE FORMAT !!

                                ;THIS ROUTINE SETS UP WC & BA FOR REMAINING SECTORS
                                ;AFTER A WRITE CHECK ERROR

017530  013737  001350  005544  SCAWC:  MOV     SAVWC,FMTDPB+4  ;SET UP WC FOR REMAINING SECTORS
017536  062737  001004  005546          ADD     #516.,FMTDPB+6  ;SET UP BA FOR REMAINING SECTORS
017544  005237  005550          INC     FMTDPB+10       ;ADVANCE TO NEXT SECTOR IN TBL
017550  005037  001316          CLR     SOFSW           ;RESET RETRY COUNTER
017554  000207                  RTS     PC              ;RETURN TO COMPLETE TRK FORMAT


                                ;THIS ROUTINE SETS UP THE CYLINDER ADRS, FORMAT BIT, TRACK AND
                                ;SECTOR ADRS IN MEMORY WITH THE STARTING CYL - TRK INFORMATION

017556  013701  001362  SETHDR: MOV     MAXSEC,R1       ;SET UP SECTOR COUNT
017562  012737  000001  001214          MOV     #1,$TESTN       ;GARBAGE CODE FOR APT
017570  012700  045774          MOV     #BUFP,R0        ;SET UP HEADER POINTER IN R0
017574  013702  001330          MOV     BEGCYL,R2       ;PUT STARTING CYL # IN R2
```

```
                                        BIS     #MF!UF,R2       ;SET MFG AND USR GOOD BITS (DEC STANDARD 144)
                                        MOV     BEGTRK,R3       ;PUT STARTING TRK # IN R3
                                        SWAB    R3              ;JUSTIFY TRACK ADRS
                                        TST     SEC30           ;SEE IF 30 OR 32 SECTOR MODE
                                        BEQ     1$              ;BR IF 30 SECTOR MODE
                                        BIS     #FMT,R2         ;SET THE 32 SECTOR FORMAT BIT
                                1$:     MOV     R2,(R0)+        ;WRITE IN HEADER AREA OF CORE THE CYL ADRS
                                        MOV     R3,(R0)+        ;WRITE IN HEADER AREA OF CORE THE TRK ADRS
                                        ADD     #512.,R0        ;SET UP FOR NEXT HEADER
                                        INC     R3              ;UPDATE SECTOR ADRS FOR NEXT HEADER
                                        DEC     R1              ;MAINTAIN COUNT OF SECTORS
                                        BGE     1$              ;BR IF NOT LAST SECTOR
                                        RTS     PC              ;EXIT - HEADERS ARE LOADED INTO CORE

                                ;THIS ROUTINE UPDATES THE CYLINDER ADRS OF THE HEADER WORDS
                                ;AND FILLS DATA BUFFER FIELD IN CORE

                                UPDCYL:
                                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
                                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
                                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
                                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
                                        MOV     MAXSEC,R1       ;SET UP SECTOR COUNT
                                        MOV     #BUFP,R0        ;SET UP HEADER POINTER IN R0
                                        MOV     BEGTRK,R3       ;GET BEGINNING TRACK
                                        SWAB    R3              ;STARTING SECTOR = 0
                                1$:     TST     WRAP            ;SEEKING FROM HI TO LO CYLINDERS ?
                                        BEQ     2$              ;BR IF NO
                                        SUB     #2,(R0)         ;DECREMENT CYLINDER NUMBER
                                2$:     INC     (R0)+           ;INCREMENT FOR NEXT CYLINDER
                                        MOV     R3,(R0)+        ;WRITE TRACK AND SECTOR HEADER
                                        MOV     #512.,R2        ;DATA FIELD LENGTH
                                3$:     MOV     PATA,(R0)+      ;FILL BUFFER
                                        MOV     PATB,(R0)+      ;FILL BUFFER
                                        SUB     #4,R2           ;END OF DATA FIELD ?
                                        BNE     3$              ;NO
                                        INC     R3              ;INCREMENT TO NEXT SECTOR
                                        DEC     R1              ;COUNT SECTORS
                                        BGE     1$              ;BR IF NOT LAST SECTOR
                                        MOV     (SP)+,R3        ;;POP STACK INTO R3
                                        MOV     (SP)+,R2        ;;POP STACK INTO R2
                                        MOV     (SP)+,R1        ;;POP STACK INTO R1
                                        MOV     (SP)+,R0        ;;POP STACK INTO R0
                                        INC     $TESTN          ;INCREMENT TST NUMBER FOR EACH CYLINDER CHANGE
                                                                ;GARBADE CODE FOR APT
                                        RTS     PC              ;EXIT

                                ;THIS ROUTINE UPDATES THE TRACK ADRS OF THE HEADER WORDS IN CORE

                                UPDTRK:
                                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
                                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
                                        MOV     MAXSEC,R1       ;SET UP SECTOR COUNT
                                        MOV     #BUFP,R0        ;SET UP HEADER POINTER IN R0
                                        TST     (R0)+           ;POINT HEADER POINTER TO TRK - SEC ADRS
                                1$:     ADD     #400,(R0)       ;INDEX TRK ADRS
                                        ADD     #516.,R0        ;SET UP FOR NEXT HEADER
```

```
                020004  005301                  DEC     R1              ;COUNT SECTORS
                020006  002372                  BGE     1$              ;BR IF NOT LAST SECTOR
                020010  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
                020012  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
                020014  000207                  RTS     PC              ;EXIT

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

                020016  012737  020074  000004  ST.CLK: MOV    #STCLK1,@#ERRVEC  ;SET UP VECTOR FOR P CLK
                020024  005037  000006          CLR     @#ERRVEC+2      ;NEW PSW
                020030  005777  161244          TST     @$LKCSR         ;CHECK FOR KW11-P
                020034  013746  001304          MOV     $LPVEC,-(SP)    ;VECTOR ADDRESS
                020040  012776  020250  000000  MOV     #CLOCK,@(SP)    ;SET UP KW11-P VECTOR
                020046  062716  000002          ADD     #2,(SP)         ;POINT TO PSW
                020052  012736  000300          MOV     #PR6,@(SP)+     ;PSW - PRI 6
                020056  012777  177777  161216  MOV     #-1,@$LKCSB     ;LOAD COUNTER BUFFER
                020064  012777  000135  161206  MOV     #135,@$LKCSR    ;SET CLK - CNT UP
                020072  000426                  BR      STCLK3
                020074  062706  000004  STCLK1: ADD     #4,SP           ;RESTORE THE STACK POINTER
                020100  012737  020144  000004  MOV     #STCLK2,@#ERRVEC        ;CHANGE ERROR VECTOR
                020106  005777  161176          TST     @$LKS           ;LOOK FOR KW11-L
                020112  013746  001312          MOV     $LLVEC,-(SP)    ;KW11-L VECTOR ADDRESS
                020116  012776  020250  000000  MOV     #CLOCK,@(SP)    ;SET UP KW11-L VECTOR
                020124  062716  000002          ADD     #2,(SP)         ;INCREMENT VECTOR ADDRESS
                020130  012736  000300          MOV     #PR6,@(SP)+     ;PSW - PRI 6
                020134  012777  000100  161146  MOV     #100,@$LKS      ;SET KW11-L INTERRUPT ENABLE
                020142  000402                  BR      STCLK3
                020144  062706  000004  STCLK2: ADD     #4,SP           ;RESTORE THE STACK POINTER
                020150  012737  000006  000004  STCLK3: MOV    #6,@#ERRVEC     ;RESTORE THE ERROR VECTOR
                020156  000207                  RTS     PC

                                        ;THIS CODE PRINT THE ABORT MESSAGE AND LET O.P. RESTART

                020160  105737  001150          RESTRT: TSTB    $AUTOB          ;RUN UNDER APT ?
                020164  001024                  BNE     1$              ;BR IF SO
                020166  004737  025576          JSR     PC,$TKINT       ;ENABLE TO READ
                020172  104401  040212          TYPE    ,MESG2          ;IMPROPER MFG INFORMATION
                020176  104410                  RDCHR
                020200  012637  001174          MOV     (SP)+,$TMPO     ;GET CHARACTER
                020204  023727  001174  000015  CMP     $TMPO,#CR       ;CARRIAGE RETURN ?
                020212  001411                  BEQ     1$              ;BR IF YES
                020214  123727  001174  000131  CMPB    $TMPO,#'Y       ;ENTER Y ?
                020222  001405                  BEQ     1$              ;BR IF YES
                020224  104401  040373          TYPE    ,MHALT
                020230  000000                  HALT
                020232  000137  007020          JMP     BEGIN2          ;JUMP TO BEGIN2

                020236  104401  036417  1$:     TYPE    ,Y              ;TYPE "Y"
                020242  104401  001205          TYPE    ,$CRLF          ;CR-LF
                020246  000207                  RTS     PC              ;EXIT

                                        ;THIS CODE SERVICES A CLOCK INTERRUPT EVERY 16MS

                020250  012746  000020  CLOCK:  MOV     #16.,-(SP)      ;PUT MILLISECONDS ON THE STACK
                020254  004737  034272          JSR     PC,RMTMR        ;GO REPORT TIME
                020260  000002                  RTI                     ;RETURN AND CONTINUE
```

```
                                ;ROUTINE TO INTERCEPT 'CONTROL C' TYPED DURING PARAMETER ENTRY TIME

020262  012706  001100          C.ENTR: MOV     #STACK,SP       ;INITIALIZE THE STACK
020266  005737  027746          1$:     TST     TRNSWT          ;ALL ACTIVITY STOPPED ?
020272  001375                          BNE     1$              ;BR IF NOT
020274  000005                          RESET                   ;CLEAR THE BUS
020276  000137  011306                  JMP     M1B             ;START AGAIN WITH DRIVE SELECTION

                                ;ROUTINE TO INTERCEPT 'CONTROL O' TYPED WHILE PROGRAM IS FORMATTING
                                ;THE DISK PACK. THE CURRENT DISK ADDRESS IS TYPED TO THE TERMINAL
                                ;DURING THIS ROUTINE IN THE FOLLOWING FORMAT.
                                ;
                                ;PRESENT ADDRESS IS: CXXX TXXX  ;WHERE 'C' IS THE CYLINDER ADDRESS
                                ;                               ;AND 'T' IS THE TRACK ADDRESS.(IN DECIMAL)

020302  117746  160654          O.ENTR: MOVB    @$TKB,-(SP)     ;READ FROM TTY BUFFER
020306  042716  177600                  BIC     #^C177,(SP)     ;STRIP PARITY
020312  021627  000003                  CMP     (SP),#3         ;CONTROL-C ?
020316  001005                          BNE     1$              ;BR IF NO
020320  104401  026767                  TYPE    ,$CNTLC         ;TYPE A CONTROL-C (^C)
020324  005726                          TST     (SP)+           ;RESTORE STACK
020326  000177  161042                  JMP     @CNTLC          ;JUMP TO CONTROL-C RESTART
020332                          1$:
020332  022627  000017                  CMP     (SP)+,#17       ;CONTROL-O ?
020336  001406                          BEQ     2$              ;YES
020340  005777  160616                  TST     @$TKB           ;CLEAR DONE BIT
020344  012777  000100  160606          MOV     #100,@$TKS      ;ENABLE TTY INTERRUPT
020352  000002                          RTI
020354                          2$:

020354  005037  177776          TYPADR: CLR     PS              ;SET PROCESSOR TO PRIORITY 0
020360  012737  020262  001374          MOV     #C.ENTR,CNTLC   ;CHANGE 'CONTROL C' RETURN ADDRESS
020366  104401  037775                  TYPE    ,ADDR1S         ;'PRESENT ADDRESS IS: '
020372  104401  036411                  TYPE    ,C              ;'C'
020376  013746  005552                  MOV     FMTDPB+12,-(SP) ;PUT THE CYLINDER ADDRESS ON THE STACK
020402  004737  027256                  JSR     PC,$SB2D        ;CONVERT IT TO DECIMAL
020406  004737  027216                  JSR     PC,$SUPRS       ;TYPE IT
020412  104401  036425                  TYPE    ,BLNKS2         ;TYPE 2 SPACES
020416  104401  036415                  TYPE    ,T              ;'T'
020422  005046                          CLR     -(SP)           ;CLEAR THE STACK
020424  113716  005551                  MOVB    FMTDPB+11,(SP)  ;PUT THE TRACK ADDRESS ON THE STACK
020430  004737  027256                  JSR     PC,$SB2D        ;CONVERT IT TO DECIMAL
020434  004737  027216                  JSR     PC,$SUPRS       ;TYPE IT
020440  104401  001205                  TYPE    ,$CRLF          ;CR-LF
020444  012737  020302  000060          MOV     #O.ENTR,@#TKVEC ;RESTORE ENTRANCE FOR CONTROL-O
020452  000002                          RTI                     ;RETURN

                                ;PARAMETER ENTRY ROUTINE
                                ;CALL
                                ;       JSR     R4,PARENT       ;THE CALLING SEQ
                                ;       TABLE                   ;PARAMETERS TABLE
                                ;

020454  010346                  PARENT: MOV     R3,-(SP)        ;SAVE R3
                                ;       MOV     #TABLE,R3       ;PARAMETER TABLE ADDRESS
020456  011403                          MOV     (R4),R3         ;PARAMETER TABLE ADDRESS
020460  012337  020470          1$:     MOV     (R3)+,2$        ;ADDRESS OF PARAMETER NAME
```

```
                                        BEQ     7$              ;BR IF AT END OF TABLE
                                        TYPE                    ;TYPE THE PARAMETER NAME
                                2$:     .WORD   0               ;ADDRESS OF PARAMETER NAME TEXT
                                        MOV     (R3)+,R2        ;MAXIMUM PARAMETER VALUE
                                        MOV     (R3)+,R5        ;ADDRESS OF PARAMETER
                                        MOV     (R5),-(SP)      ;CURRENT VALUE OF PARAMETER
                                        TYPDS                   ;TYPE THE CURRENT VALUE OF THE PARAMETER
                                        TYPE    ,SLASH          ;' / '
                                        RDLIN                   ;READ THE KEYBOARD
                                        MOV     (SP)+,R1        ;INPUT ASCII STRING ADDRESS
                                        JSR     R5,CK.DIG       ;CHECK THE DIGIT(S)
                                        1$                      ;CARRIAGE RETURN ONLY ENTERED
                                        7$                      ;5$ ONLY ENTERED
                                        4$                      ;ILLEGAL INPUT
                                        3$                      ;TERMINATED WITH A CARRIAGE RETURN
                                        4$                      ;TERMINATED WITH A ''.''
                                        5$                      ;TERMINATED WITH A ''.''
                                3$:     MOV     R2,(R5)         ;MOVE NEW VALUE TO PARAMETER LOCATION
                                        BR      1$              ;GET MORE PARAMETERS
                                4$:     TYPE    ,BADENT         ;'BAD ENTRY'
                                        SUB     #6,R3           ;DECREMENT THE TABLE POINTER
                                        BR      1$              ;TRY AGAIN
                                5$:     MOV     R2,(R5)         ;NEW VALUE
                                        BR      7$              ;EXIT
                                ;6$:    MOV     #TABLE,R3       ;RELOAD THE PARAMETER TABLE ADDRESS
                                6$:     MOV     (R4),R3         ;RELOAD THE PARAMETER TABLE ADDRESS
                                        BR      1$              ;TRY AGAIN
                                7$:     MOV     (SP)+,R3        ;RESTORE R3
                                        ADD     #2,R4   ;ADJUST THE RETURN ADDRESS
                                        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

                                RECORD: MOV     #USTAB+10,R1    ;TABLE ENTRY IN R1
                                1$:     CMP     #-1,(R1)        ;AN OPENING IN THE TABLE
                                        BEQ     3$              ;BR IF IT IS
                                        CMP     FMTDPB+12,(R1) ;CYLINDER NUMBER MATCHES ?
                                        BNE     2$              ;BR IF NOT
                                        CMPB    FMTDPB+11,3(R1) ;TRK NUMBER MATCHES ?
                                        BNE     2$              ;BR IF NOT
                                        CMPB    FMTDPB+10,2(R1) ;SECTOR NUMBER MATCHES ?
                                        BEQ     4$              ;BR TO EXIT,IF THE SPOT HAS RECORDED
                                2$:     ADD     #4,R1           ;INCREMENT 4 BYTES
                                        CMP     #USTAB+512.,R1  ;END OF TABLE ?
                                        BHI     1$              ;BR IF NOT
                                        BR      5$              ;THROW AWAY THE PACK
                                3$:     MOV     FMTDPB+12,(R1)  ;LOAD THE CYLINDER #
                                        MOVB    FMTDPB+11,3(R1) ;LOAD THE TRK NUMBER
                                        MOVB    SAVSEC,2(R1)    ;LOAD THE SECTOR NUMBER
                                4$:     RTS     PC              ;EXIT
                                5$:     TYPE    ,$CRLF
                                        TYPE    ,MESG3          ;THROW AWAY THE PACK
                                        TYPE    ,MHALT
                                        HALT
```

```
    929 020704  000137  007020                  JMP     BEGIN2          ;RESTART IF DESIRED ?
    930
    931                                 ;THIS ROUTINE SORT THE USTAB IN ASCENDING ORDER
    932                                 ;CALL SEQ       JSR     'PC,SORT2
    933
    934 020710                          SORT2:
    935                                 ;       MOV     #USTAB+8.,R1    ;TOP POINTER
    936 020710  011501                          MOV     (R5),R1         ;TOP POINTER
    937 020712  012702  000175                  MOV     #125.,R2        ;TOTAL 126 ELEMENTS
    938 020716  010103                          MOV     R1,R3           ;LOCATE THE BOTTOM POINTER
    939 020720  062703  000764                  ADD     #500.,R3        ;254-4) X 2
    940                                 ;       MOV     #USTAB+508.,R3  ;THE LAST ELEMENT
    941 020724  022711  177777                  CMP     #-1,(R1)        ;THE TABLE IS EMPTY ?
    942 020730  001444                          BEQ     4$              ;QUICK EXIT
    943 020732  021161  000004          1$:     CMP     (R1),4(R1)      ;COMPARE THE CYLINDER NUMBER
    944 020736  101013                          BHI     2$              ;SWITCH THE TWO ELEMENT
    945 020740  103426                          BLO     3$              ;INCREMENT POINTER
    946 020742  126161  000003  000007          CMPB    3(R1),7(R1)     ;COPMARE THE TRK NUMBER
    947 020750  101006                          BHI     2$              ;SWITCH ELEMENT
    948 020752  103421                          BLO     3$              ;INCREMENT POINTER
    949 020754  126161  000002  000006          CMPB    2(R1),6(R1)     ;COMPARE THE SECTOR NUMBER
    950 020762  101001                          BHI     2$              ;SWITCH ELEMENT
    951 020764  000414                          BR      3$              ;INCREMENT POINTER
    952 020766  011146                  2$:     MOV     (R1),-(SP)      ;SAVE THE N BLOCK
    953 020770  016146  000002                  MOV     2(R1),-(SP)     ;INTO STACK
    954 020774  016111  000004                  MOV     4(R1),(R1)      ;SWITCH THE N+1 BLOCK TO N BLOCK
    955 021000  016161  000006  000002          MOV     6(R1),2(R1)     ;
    956 021006  012661  000006                  MOV     (SP)+,6(R1)     ;LOAD THE N+1 BLOCK
    957 021012  012661  000004                  MOV     (SP)+,4(R1)     ;
    958 021016  062701  000004          3$:     ADD     #4,R1           ;UPDATE THE TOP POINTER
    959 021022  020103                          CMP     R1,R3           ;REACH THE BOTTOM ?
    960 021024  001342                          BNE     1$              ;BR IF NOT
    961 021026  005302                          DEC     R2              ;DECREMENT ONE SORT COUNT
    962 021030  001404                          BEQ     4$              ;BR IF ALL DONE
    963 021032  162703  000004                  SUB     #4,R3           ;ASJUST THE BOTTOM POINTER
    964                                 ;       MOV     #USTAB+8.,R1    ;RESET TOP POINTER
    965 021036  011501                          MOV     (R5),R1         ;RESET THE TOP POINTER
    966 021040  000734                          BR      1$              ;BR IF NOT ALL DONE
    967 021042                          4$:
    968 021042  062705  000002                  ADD     #2,R5           ;ADJUST RETURN ADDRESS
    969 021046  000205                          RTS     R5              ;EXIT
    970                                 ;       RTS     PC              ;EXIT
    971
    972                                 ;THIS ROUTINE TEXT THE LAST TRACK   CYL 822, LAST TRACK
    973                                 ;SECTORS 0,2,4,6,8      16 BIT MFG
    974                                 ;SECTORS 1,3,5,7,9      18 BIT MFG
    975                                 ;SECTORS 10,12,14,16.....30,  16 BIT USER
    976                                 ;SECTORS 11,13,15,17.....31         18 BIT USER
    977
    978 021050  012700  045774          TEXT:   MOV     #BUFP,R0        ;BUFFER ADDRESS
    979 021054  012701  020100                  MOV     #<258.*32.>,R1  ;TOTAL WORD COUNT
    980 021060  012702  177777                  MOV     #-1,R2          ;SET ALL LOCATIONS TO -1
    981 021064  010220                  1$:     MOV     R2,(R0)+        ;FULL THE BUFFER
    982 021066  005301                          DEC     R1              ;ALL DONE ?
    983 021070  001375                          BNE     1$              ;LOOP, IF NOT
    984 021072  013746  001214                  MOV     $TESTN,-(SP)    ;SAVE TESTN,BEGCYL,BEGTRK
    985 021076  013746  001330                  MOV     BEGCYL,-(SP)    ;
```

```
 986 021102  013746  001334                    MOV     BEGTRK,-(SP)    ;
 987 021106  013746  001362                    MOV     MAXSEC,-(SP)    ;SAVE FLAG MAXSEC AND SEC30
 988 021112  013746  001360                    MOV     SEC30,-(SP)     ;
 989 021116  012737  000037  001362            MOV     #37,MAXSEC      ;SET MAX 31 SECTORS
 990 021124  012737  177777  001360            MOV     #-1,SEC30       ;ALWAYS IN 16 BIT MODE
 991 021132  012737  001466  001330            MOV     #822.,BEGCYL    ;LOAD LAST CYLINDER
 992 021140  013737  001324  001334            MOV     LSTRK,BEGTRK    ;LAST TRACK
 993 021146  004737  017556                    JSR     PC,SETHDR       ;SET UP THE HEAD FIELD IN THE BUFFER
 994 021152  012637  001360                    MOV     (SP)+,SEC30     ;RESTORE SEC30 AND MAXSEC
 995 021156  012637  001362                    MOV     (SP)+,MAXSEC    ;
 996 021162  012637  001334                    MOV     (SP)+,BEGTRK    ;
 997 021166  012637  001330                    MOV     (SP)+,BEGCYL    ;
 998 021172  012637  001214                    MOV     (SP)+,$TESTN    ;
 999 021176  012701  001414                    MOV     #BAD16,R1       ;LOAD SERIAL NUMBER TO EVERY SECTOR
1000 021202  012702  046000                    MOV     #BUFP+4,R2      ;BUFFER LOCATION + 4
1001 021206  012703  000040                    MOV     #32.,R3         ;TOTAL 32 SECTORS
1002 021212  011112                2$:        MOV     (R1),(R2)       ;1 ST SN #
1003 021214  016162  000002  000002            MOV     2(R1),2(R2)     ;2 ND SN #
1004 021222  005062  000004                    CLR     4(R2)           ;THIRD WORD = 0
1005 021226  005062  000006                    CLR     6(R2)           ;ID = DATA PACK
1006 021232  062702  001004                    ADD     #516.,R2        ;ADVANCE TO NEXT SECTOR
1007 021236  005303                            DEC     R3              ;DECREMENT ONE SECTOR COUNT
1008 021240  001364                            BNE     2$              ;BR IF NOT DONE
1009 021242  005046                            CLR     -(SP)           ;LOAD TABLE INTO SECTORS
1010 021244  005046                            CLR     -(SP)                           ;DUMMY PAIRS
1011 021246  005046                            CLR     -(SP)
1012 021250  012746  001414                    MOV     #BAD16,-(SP)    ;TABLE ADDRESS
1013 021254  012746  046000                    MOV     #BUFP+4,-(SP)   ;BUFFER ADDRESS
1014 021260  012746  000005                    MOV     #5,-(SP)        ;SECTOR COUNT
1015 021264  012746  002420                    MOV     #BAD18,-(SP)    ;TABLE ADDRESS
1016 021270  012746  047004                    MOV     #BUFP+4+516.,-(SP) ;BUFFER ADDRESS
1017 021274  012746  000005                    MOV     #5,-(SP)        ;SECTOR COUNT
1018 021300  012746  003424                    MOV     #USTAB,-(SP)    ;USER DEFINED TABLE
1019 021304  012746  060050                    MOV     #BUFP+4+<516.*10.>,-(SP)        ;BUFFER ADDRESS
1020 021310  005737  001360                    TST     SEC30
1021 021314  100402                            BMI     3$              ;BR IF IN 16 BIT MODE
1022 021316  012716  061054                    MOV     #BUFP+4+<516.*11.>,(SP) :RESET THE BUFFER ADDRESS
1023 021322  012746  000013        3$:         MOV     #11.,-(SP)      ;SECTOR COUNT
1024 021326  012701  004430                    MOV     #USSAV,R1       ;TABLE ADDRESS IN R1
1025 021332  012702  061054                    MOV     #BUFP+4+<516.*11.>,R2   ;BUFFER ADDRESS IN R2
1026 021336  005737  001360                    TST     SEC30           ;BR IF IN 16 BIT MODE
1027 021342  100402                            BMI     4$              ;
1028 021344  012702  060050                    MOV     #BUFP+4+<516.*10.>,R2   ;BUFFER ADDRESS FOR 18 BIT MODE
1029 021350  012703  000013        4$:         MOV     #11.,R3         ;SECTOR COUNT IN R3
1030 021354  010105                            MOV     R1,R5           ;TEMPOR STORAGE
1031 021356  012704  000400        5$:         MOV     #256.,R4        ;WORD COUNT = DATA FIELD OF ONE SECTOR
1032 021362  012122                6$:         MOV     (R1)+,(R2)+     ;LOAD TABLE INTO SECTOR DATA FIELD
1033 021364  005304                            DEC     R4              ;ALL DONE ?
1034 021366  001375                            BNE     6$              ;BR IF NOT
1035 021370  010501                            MOV     R5,R1           ;RELOAD THE TABLE ADDRESS
1036 021372  062702  001010                    ADD     #520.,R2        ;SKIP ONE SECTOR
1037 021376  005303                            DEC     R3              ;DECREMENT ONE SECTOR COUNT
1038 021400  001366                            BNE     5$              ;BR IF NOT DONE
1039 021402  012603                            MOV     (SP)+,R3        ;RETRIEVE NEXT SET OF SECTOR #
1040 021404  012602                            MOV     (SP)+,R2        ;BUFFER ADDRESS
1041 021406  012601                            MOV     (SP)+,R1        ;TABLE ADDRESS
1042 021410  010105                            MOV     R1,R5           ;TEMPOR STORAGE
```

```
1043 021412  001361                        BNE     5$              ;BR IF NOT DUMMY PAIR
1044 021414  000207                        RTS     PC              ;EXIT
1045
1046                                ;THIS ROUTINE RESET:
1047                                ;RESET THE BIT 14/15 OF THE 1 ST HEADER WORD OF THE BAD SPOT
1048                                ;CALLING SEQ;           JSR     PC,RESET
1049
1050 021416  013737  005552  045764 RESET:  MOV     FMTDPB+12,RBUF  ;CYLINDER ADDRESS
1051 021424  001006                        BNE     1$              ;BR IF NOT ON TRACK 0
1052 021426  122737  000001  005551        CMPB    #1,FMTDPB+11    ;BR IF NOT CYL0,TRK 0 OR 1
1053 021434  103402                        BLO     1$              ;BR IF NOT
1054 021436  000137  017134                JMP     REJCT2          ;OTHERWISE, ILLEGAL BAD SECTORS
1055 021442  000240                1$:     NOP                     ;
1056 021444  113737  005551  045767        MOVB    FMTDPB+11,RBUF+3        ;TRACK ADDRESS
1057 021452  113737  001346  045766        MOVB    SAVSEC,RBUF+2   ;SECTOR ADDRESS
1058 021460  053737  001410  045764        BIS     HEADX,RBUF      ;SET MFG BIT,RESET USER BIT
1059 021466  113737  001222  005630        MOVB    DRIVE,FMTX      ;SETUP THE DPB FOR OPERATION
1060 021474  012737  177776  005634        MOV     #-2,FMTX+4      ;WORD COUNT =2
1061 021502  112737  000163  005632        MOVB    #WRTHD,FMTX+2   ;WRITE HEAD AND DATA COMMAND
1062 021510  012737  045764  005636        MOV     #RBUF,FMTX+6    ;BUFFER ADDRESS
1063 021516  113737  001346  005640        MOVB    SAVSEC,FMTX+10  ;SECTOR ADDRESS
1064 021524  113737  005551  005641        MOVB    FMTDPB+11,FMTX+11 ;TRACK ADDRESS
1065 021532  013737  005552  005642        MOV     FMTDPB+12,FMTX+12 ;CYLINDER ADDRESS
1066 021540  052737  010000  045764        BIS     #FMT,RBUF       ;ASSUM 16 BIT MODE
1067 021546  005737  001360                TST     SEC30           ;IN 16 BIT MODE ?
1068 021552  100403                        BMI     2$              ;BR IF IT IS
1069 021554  042737  010000  045764        BIC     #FMT,RBUF       ;RESET THE FMT BIT
1070 021562  004037  030622        2$:     JSR     R0,RM05         ;CALL THE DRIVER
1071 021566  005630                        FMTX                    ;DPB ADDRESS
1072 021570  000774                        BR      2$              ;BR IF QUEUE FAILS
1073 021572  005737  005646        3$:     TST     FMTX+16         ; DONE ?
1074 021576  001775                        BEQ     3$              ;BR IF NOT
1075 021600  000207                        RTS     PC              ;EXIT
1076
1077                                ;THE ROUTINE INSERT:
1078                                ;INSERT A BAD SECTOR ADDRESS INTO ONE OF THE USTAB,USSAV,BAD16,BAD18 TABLES
1079                                ;CALLING SEQ
1080                                ;               JSR     R5,INSERT
1081                                ;               TABLE + 8.
1082
1083 021602  010446                INSERT: MOV     R4,-(SP)        ;SAVE THE R4
1084 021604  012746  000176                MOV     #126.,-(SP)     ;ENTRIES OF THE TABLE
1085 021610  011504                        MOV     (R5),R4         ;TABLE ADDRESS + 8 BYTES
1086 021612  022714  177777        1$:     CMP     #-1,(R4)        ;AN OPEN ENTRY ?
1087 021616  001420                        BEQ     3$              ;BR IF SO
1088 021620  023714  001364                CMP     DS.CYL,(R4)     ;CHECK IF THE BAD SECTOR HAS RECORDED
1089 021624  001010                        BNE     2$              ;BR IF NOT
1090 021626  123764  001366  000003        CMPB    DS.TRK,3(R4)    ;CYL AND TRK MATCH ?
1091 021634  001004                        BNE     2$              ;BR IF NOT
1092 021636  123764  001346  000002        CMPB    SAVSEC,2(R4)    ;CYL ,TRK AND SEC MATCH ?
1093 021644  001415                        BEQ     4$              ;BR IF SO
1094 021646  005316                2$:     DEC     (SP)            ;DECREMENT THE ENTRY COUNT
1095 021650  001413                        BEQ     4$              ;BR IF EXHAUST
1096 021652  062704  000004                ADD     #4,R4           ;LOCATE THE NEXT ENTRY
1097 021656  000755                        BR      1$              ;BR BACK
1098 021660  013714  001364        3$:     MOV     DS.CYL,(R4)     ;LOAD INTO TABLE
1099 021664  113764  001366  000003        MOVB    DS.TRK,3(R4)    ;LOAD THE TRACK NUMBER
```

```
1100 021672  113764  001346  000002          MOVB    SAVSEC,2(R4)    ;LOAD THE SECTOR NUMBER
1101 021700  062706  000002          4$:     ADD     #2,SP           ;CLEAR UP THE STACK
1102 021704  012604                          MOV     (SP)+,R4        ;RESTORE R4
1103 021706  062705  000002                  ADD     #2,R5           ;ASJUST THE RETURN ADDRESS
1104 021712  000205                          RTS     R5              ;EXIT
1105
1106 021714  104401  041236          FUNCT1: TYPE    ,MESG5
1107 021720  104401  001205                  TYPE    ,$CRLF          ;CR-LF
1108 021724  012737  177777  001364  1$:     MOV     #-1,DS.CYL      ;RESET THE INPUT VALUES
1109 021732  012737  177777  001366          MOV     #-1,DS.TRK      ;
1110 021740  012737  177777  001346          MOV     #-1,SAVSEC      ;
1111 021746  004437  020454                  JSR     R4,PARENT       ;ENTER THE VALUE FROM THE KEYBOARD
1112 021752  005702                          TABLE2
1113
1114 021754  005737  001364                  TST     DS.CYL          ;UPDATE VALUES ?
1115 021760  100412                          BMI     2$              ;BR IF NOT
1116 021762  005737  001366                  TST     DS.TRK          ;
1117 021766  100407                          BMI     2$              ;
1118 021770  005737  001346                  TST     SAVSEC          ;
1119 021774  100404                          BMI     2$              ;
1120 021776  004537  021602                  JSR     R5,INSERT       ;INSERT INTO THE BAD16 TABLE
1121 022002  001424                          BAD16+8.                ;
1122 022004  000747                          BR      1$              ;NEXT SET
1123 022006  004537  020710          2$:     JSR     R5,SORT2        ;SORT THE TABLE
1124 022012  001424                          BAD16+8.
1125
1126 022014  104401  041321                  TYPE    ,MESG6
1127 022020  104401  001205                  TYPE    ,$CRLF          ;CR-LF
1128 022024  012737  177777  001364  3$:     MOV     #-1,DS.CYL      ;RESET THE INPUT VALUES
1129 022032  012737  177777  001366          MOV     #-1,DS.TRK      ;
1130 022040  012737  177777  001346          MOV     #-1,SAVSEC      ;
1131 022046  004437  020454                  JSR     R4,PARENT       ;INPUT VALUES FROM THE KEYBOARD
1132 022052  005726                          TABLE3                  ;18 BIT TABLE
1133
1134 022054  005737  001364                  TST     DS.CYL          ;UPDATED VALUES ?
1135 022060  100412                          BMI     4$              ;BR IF NOT
1136 022062  005737  001366                  TST     DS.TRK          ;
1137 022066  100407                          BMI     4$              ;
1138 022070  005737  001346                  TST     SAVSEC          ;
1139 022074  100404                          BMI     4$              ;
1140 022076  004537  021602                  JSR     R5,INSERT       ;INSERT THE BAD SECTOR INTO TABLE BAD18
1141 022102  002430                          BAD18+8.
1142 022104  000747                          BR      3$              ;NEXT SET
1143 022106  004537  020710          4$:     JSR     R5,SORT2        ;SORT THE BAD18 TABLE
1144 022112  002430                          BAD18+8.
1145 022114  000207                          RTS     PC              ;EXIT FROM THE CALLING OF TABCAL
1146
1147 022116  012737  003434  022272  FUNCT2: MOV     #USTAB+8.,3$    ;SETUP TABLES FOR 16 BITS MODE
1148 022124  012737  003434  022302          MOV     #USTAB+8.,5$    ;
1149 022132  012737  004440  022372          MOV     #USSAV+8.,7$    ;
1150 022140  012737  004440  022402          MOV     #USSAV+8.,9$    ;
1151 022146  005737  001360                  TST     SEC30           ;IN 16 BIT MODE ?
1152 022152  100414                          BMI     1$              ;BR IS SO,OTHERWISE RELOAD THE TABLES
1153 022154  012737  003434  022372          MOV     #USTAB+8.,7$    ;SET UP TABLES FOR 18 BITS MODE
1154 022162  012737  003434  022402          MOV     #USTAB+8.,9$    ;
1155 022170  012737  004440  022272          MOV     #USSAV+8.,3$    ;
1156 022176  012737  004440  022302          MOV     #USSAV+8.,5$    ;
```

```
1157
1158 022204  104401  041404          1$:     TYPE    ,MESG7          ;MESSAGE FOR ENTER 16 BITS USR TABLE
1159 022210  104401  001205                  TYPE    ,$CRLF          ;CR-LF
1160 022214  012737  177777  001364  2$:     MOV     #-1,DS.CYL      ;RESET THE INPUT VALUES
1161 022222  012737  177777  001366          MOV     #-1,DS.TRK      ;
1162 022230  012737  177777  001346          MOV     #-1,SAVSEC      ;
1163 022236  004437  020454                  JSR     R4,PARENT       ;ENTER BAD SECTOR ADDRESS FROM THE KEYBOARD
1164 022242  005702                          TABLE2
1165 022244  005737  001364                  TST     DS.CYL          ;UPDATED INPUT VALUE ?
1166 022250  100412                          BMI     4$              ;BR IF NOT
1167 022252  005737  001366                  TST     DS.TRK          ;
1168 022256  100407                          BMI     4$              ;
1169 022260  005737  001346                  TST     SAVSEC          ;
1170 022264  100404                          BMI     4$              ;
1171 022266  004537  021602                  JSR     R5,INSERT       ;INSERT THE BAD SECTOR INTO TABLE
1172 022272  000000          3$:     .WORD   0               ;TABLE ADDRESS
1173 022274  000747                  BR      2$              ;ENTER NEXT BAD SECTOR ADDRESS
1174 022276  004537  020710          4$:     JSR     R5,SORT2        ;SORT THE TABLE
1175 022302  000000          5$:     .WORD   0               ;TABLE ADDRESS
1176
1177 022304  104401  041467                  TYPE    ,MESG8          ;MESSAGE FOR 18 BIT USR TABLE
1178 022310  104401  001205                  TYPE    ,$CRLF          ;CR-LF
1179 022314  012737  177777  001364  6$:     MOV     #-1,DS.CYL      ;RESET THE INPUT VALUES
1180 022322  012737  177777  001366          MOV     #-1,DS.TRK      ;
1181 022330  012737  177777  001346          MOV     #-1,SAVSEC      ;
1182 022336  004437  020454                  JSR     R4,PARENT       ;ENTER BAD SECTOR FORM KEYBOARD
1183 022342  005726                          TABLE3                  ;
1184
1185 022344  005737  001364                  TST     DS.CYL          ;UPDATED INPUT VALUE ?
1186 022350  100412                          BMI     8$              ;BR IF NOT
1187 022352  005737  001366                  TST     DS.TRK          ;BR IF NOT UPDATE
1188 022356  100407                          BMI     8$              ;
1189 022360  005737  001346                  TST     SAVSEC          ;
1190 022364  100404                          BMI     8$              ;BR IF NOT UPDATED
1191 022366  004537  021602                  JSR     R5,INSERT       ;INSERT THE BAD SECTOR INTO TABLE
1192 022372  000000          7$:     .WORD   0               ;
1193 022374  000747                  BR      6$              ;BR TO ENTER NEXT SECTOR
1194 022376  004537  020710          8$:     JSR     R5,SORT2        ;SORT THE TABLE
1195 022402  000000          9$:     .WORD   0               ;TABLE ADDRESS
1196 022404  012737  177777  001406          MOV     #-1,EDIT        ;UPDATE BAD SECTOR FILE AT END OF PASS
1197 022412  000207                          RTS     PC              ;EXIT FORM THE TABCAL CALLING
1198
1199 022414  010146          FUNCT3: MOV     R1,-(SP)        ;SAVE R1
1200 022416  012701  001414                  MOV     #BAD16,R1       ;ADDRESS OF THE TABLE
1201 022422  104401  041552                  TYPE    ,MESG9          ;MESG TO TYPE THE 16 BIT MFG
1202 022426  104401  001205          1$:     TYPE    ,$CRLF          ;CR-LF FIRST
1203 022432  012146                          MOV     (R1)+,-(SP)     ;CALLING SEQ FOR TYPOC RT.
1204 022434  100402                          BMI     2$              ;BR IF END OF TABLE
1205 022436  104402                          TYPOC                   ;TYPE THE VALUE
1206 022440  000772                          BR      1$              ;LOOPING BACK
1207 022442  062706  000002          2$:     ADD     #2,SP           ;ADJUST THE RETURN ADDRESS
1208 022446  104401  041622                  TYPE    ,MESG10         ;MESG TO TYPE 18 BIT MFG
1209 022452  012701  002420                  MOV     #BAD18,R1       ;TABLE ADDRESS
1210 022456  104401  001205          3$:     TYPE    ,$CRLF          ;
1211 022462  012146                          MOV     (R1)+,-(SP)     ;CALLING SEQ FOR TYPOC RT.
1212 022464  100402                          BMI     4$              ;BR IF END OF TABLE
1213 022466  104402                          TYPOC                   ;TYPE THE VALUE IN OCTAL
```

```
1214 022470  000772                    BR      3$              ;LOOPING BACK
1215 022472  062706  000002      4$:    ADD     #2,SP           ;ADJUST THE RETURN ADDRESS
1216 022476  012601              5$:    MOV     (SP)+,R1        ;RESTORE THE R1
1217 022500  000207                     RTS     PC              ;EXIT FROM THE CALLING OF TABCAL
1218
1219 022502  010146              FUNCT4: MOV     R1,-(SP)        ;SAVE R1
1220 022504  010246                      MOV     R2,-(SP)        ;SAVE R2
1221 022506  012701  003424              MOV     #USTAB,R1       ;SET 16 BIT MODE TABLE
1222 022512  012702  004430              MOV     #USSAV,R2
1223 022516  005737  001360              TST     SEC30           ;16 BIT MODE ?
1224 022522  100404                      BMI     1$              ;BR IF SO
1225 022524  012702  003424              MOV     #USTAB,R2       ;OTHERWISE,RELOAD THE TABLE ADDRESS
1226 022530  012701  004430              MOV     #USSAV,R1       ;
1227 022534  104401  041672      1$:     TYPE    ,MESG11         ;MESG FOR 16 BIT
1228 022540  104401  001205      2$:     TYPE    ,$CRLF          ;
1229 022544  012146                      MOV     (R1)+,-(SP)     ;
1230 022546  100402                      BMI     3$              ;BR IF END OF TABLE
1231 022550  104402                      TYPOC                   ;
1232 022552  000772                      BR      2$              ;
1233 022554  062706  000002      3$:     ADD     #2,SP           ;ADJUST STACK POINTER
1234 022560  104401  041742              TYPE    ,MESG12         ;MESSAGE FOR 18 BIT MODE
1235 022564  104401  001205      4$:     TYPE    ,$CRLF          ;CR-LF
1236 022570  012246                      MOV     (R2)+,-(SP)     ;
1237 022572  100402                      BMI     5$              ;BR IF END OF TABLE
1238 022574  104402                      TYPOC
1239 022576  000772                      BR      4$              ;NEXT SECTOR ADDRESS
1240 022600  062706  000002      5$:     ADD     #2,SP           ;ADJUST THE STACK POINT
1241 022604  012602                      MOV     (SP)+,R2        ;RESTORE R2,R1
1242 022606  012601                      MOV     (SP)+,R1        ;
1243 022610  000207                      RTS     PC              ;EXIT FROM THE CALLING OF TABCAL
1244
1245 022612  010146              FUNCT5: MOV     R1,-(SP)        ;SAVE R1,R2
1246 022614  010246                      MOV     R2,-(SP)
1247 022616  005037  001420              CLR     BAD16+4.
1248 022622  005037  001422              CLR     BAD16+6.
1249 022626  012701  001424              MOV     #BAD16+8.,R1    ;START AT THE 5TH WORD
1250 022632  012702  000374              MOV     #252.,R2        ;WORD CTR
1251 022636  012721  177777      1$:     MOV     #-1,(R1)+       ;RESET TO -1
1252 022642  005302                      DEC     R2              ;DECREMENT WORD CTR
1253 022644  001374                      BNE     1$              ;BR IF NOT DONE
1254 022646  005037  002424              CLR     BAD18+4.
1255 022652  005037  002426              CLR     BAD18+6.
1256 022656  012701  002430              MOV     #BAD18+8.,R1    ;START AT THE 5TH WORD
1257 022662  012702  000374              MOV     #252.,R2        ;WORD CTR
1258 022666  012721  177777      2$:     MOV     #-1,(R1)+       ;RESET TO -1
1259 022672  005302                      DEC     R2              ;DECREMENT WORD CTR
1260 022674  001374                      BNE     2$              ;BR IF NOT DONE
1261 022676  012602                      MOV     (SP)+,R2        ;RESTORE R2,R1
1262 022700  012601                      MOV     (SP)+,R1
1263 022702  104401  042041              TYPE    ,MESG14         ;DONE MESAGE
1264 022706  000207                      RTS     PC              ;EXIT FROM THE TABCAL CALLING
1265
1266 022710  010146              FUNCT6: MOV     R1,-(SP)        ;STORE R1,R2
1267 022712  010246                      MOV     R2,-(SP)
1268 022714  005037  003430              CLR     USTAB+4
1269 022720  005037  003432              CLR     USTAB+6
1270 022724  012701  003434              MOV     #USTAB+8.,R1
```

```
1271 022730  012702  000374                MOV     #252.,R2        ;WORD CTR
1272 022734  012721  177777          1$:   MOV     #-1,(R1)+       ;RESET TO -1
1273 022740  005302                        DEC     R2              ;DECREMENT WORD CTR
1274 022742  001374                        BNE     1$              ;BR IF NOT DONE
1275 022744  005037  004434                CLR     USSAV+4
1276 022750  005037  004436                CLR     USSAV+6
1277 022754  012701  004440                MOV     #USSAV+8.,R1    ;TABLE ADDRESS
1278 022760  012702  000374                MOV     #252.,R2
1279 022764  012721  177777          2$:   MOV     #-1,(R1)+       ;RESET TO -1
1280 022770  005302                        DEC     R2              ;DECREMENT WORD CTR
1281 022772  001374                        BNE     2$              ;BR IF NOT DONE
1282 022774  012602                        MOV     (SP)+,R2        ;RESTORE R1,R2
1283 022776  012601                        MOV     (SP)+,R1
1284 023000  104401  042041                TYPE    ,MESG14         ;DONE MESSAGE
1285 023004  012737  177777  001406        MOV     #-1,EDIT        ;UPDATE BAD SECTOR FILE AT END OF PASS
1286 023012  000207                        RTS     PC              ;EXIT FORM THE TABCAL CALLING
1287
1288 023014  104401  042012          FUNCT7: TYPE    ,MESG13         ;MESSAGE ENTER SECTOR #
1289 023020  104411                  1$:   RDLIN                   ;READ THE SECTOR NUMBER
1290 023022  012601                        MOV     (SP)+,R1        ;ADDRESS OF READ-IN BUFFER
1291 023024  012702  000040                MOV     #32.,R2         ;MAX VALUE
1292 023030  004537  023462                JSR     R5,CK.DIG       ;CHECK THE READ-IN VALUE
1293 023034  023050                        2$
1294 023036  023050                        2$
1295 023040  023020                        1$
1296 023042  023050                        2$
1297 023044  023050                        2$
1298 023046  023050                        2$
1299 023050  006302                  2$:   ASL     R2              ;FOUND THE WORD INDEX
1300 023052  016201  006132                MOV     ADRTBL(R2),R1   ;ADDRESS OF THE BUFFER
1301 023056  012702  000402                MOV     #258.,R2        ;WC FOR WHOLE SECTOR INCLUDING HEADER
1302 023062  104401  001205          3$:   TYPE    ,$CRLF          ;CR-LF
1303 023066  012705  000010                MOV     #10,R5          ;8 WORDS ON A LINE
1304 023072  012146                  4$:   MOV     (R1)+,-(SP)     ;CONTENTS OF THE BUFFER
1305 023074  104402                        TYPOC                   ;
1306 023076  104401  036425                TYPE    ,BLNKS2         ;TYPE 2 SPACES
1307 023102  005302                        DEC     R2              ;DECREMENT WORD CTR
1308 023104  001403                        BEQ     5$              ;BR IF  FINISH
1309 023106  005305                        DEC     R5              ;START ANOTHER LINE ?
1310 023110  001370                        BNE     4$              ;BR IF NOT
1311 023112  000763                        BR      3$              ;BR IF SO
1312 023114  104401  001205          5$:   TYPE    ,$CRLF          ;CR-LF
1313 023120  000207                        RTS     PC              ;EXIT FROM TABCAL CALLING
1314
1315 023122  104401  042051          FUNCT8: TYPE    ,MESG15
1316 023126  012737  177777  001364  1$:   MOV     #-1,DS.CYL      ;RESET ALL INPUT VALUES
1317 023134  012737  177777  001366        MOV     #-1,DS.TRK      ;TRACK NUMBER
1318 023142  012737  047301  001346        MOV     #20161.,SAVSEC  ;BYTE NUMBER
1319 023150  004437  020454                JSR     R4,PARENT       ;ENTER THE VALUE FROM KEYBOARD
1320 023154  005752                        TABLE4
1321
1322 023156  005737  001364                TST     DS.CYL          ;UPDATED ?
1323 023162  100455                        BMI     6$              ;EXIT IF NOT
1324 023164  005737  001366                TST     DS.TRK          ;BR IF NOT UPDATED
1325 023170  100452                        BMI     6$              ;
1326 023172  005737  001346                TST     SAVSEC          ;BR TO EXIT IF NEG
1327 023176  100447                        BMI     6$              ;EXIT
```

```
1328 023200  022737  047301  001346          CMP     #20161.,SAVSEC  ;TOO LARGE ?
1329 023206  003443                          BLE     6$              ;BR IF SO,EXIT
1330 023210  010446                          MOV     R4,-(SP)        ;SAVE R4
1331 023212  013746  001346                  MOV     SAVSEC,-(SP)    ;SAVE SAVSEC
1332 023216  005004                          CLR     R4              ;INDEX STARTS FROM 0
1333 023220  023764  001346  006332  2$:     CMP     SAVSEC,BYTE16(R4)       ;LOCATE THE SECTOR ADDRESS
1334 023226  101403                          BLOS    3$              ;BR IF FOUND
1335 023230  062704  000002                  ADD     #2,R4           ;TRY NEXT SECTOR
1336 023234  000771                          BR      2$              ;LOOPING
1337 023236  006204                  3$:     ASR     R4              ;SECTOR ADDRESS
1338 023240  010437  001346                  MOV     R4,SAVSEC       ;16 BIT MODE BAD SECTOR ADD
1339 023244  004537  021602                  JSR     R5,INSERT       ;INSERT INTO TABLE
1340 023250  001424                          BAD16+8.
1341 023252  012637  001346                  MOV     (SP)+,SAVSEC    ;LOCATE SECTOR FOR 18 BIT
1342 023256  005004                          CLR     R4              ;RESET INDEX VALUE
1343 023260  023764  001346  006432  4$:     CMP     SAVSEC,BYTE18(R4)       ;USE TABLE 18 BIT
1344 023266  101403                          BLOS    5$              ;BR IF FOUND
1345 023270  062704  000002                  ADD     #2,R4           ;TRY NEXT SECTOR IF NOT FOUND
1346 023274  000771                          BR      4$              ;LOOPIN BACK
1347 023276  006204                  5$:     ASR     R4              ;SECTOR ADDRESS
1348 023300  010437  001346                  MOV     R4,SAVSEC       ;
1349 023304  004537  021602                  JSR     R5,INSERT       ;INSET INTO TABLE
1350 023310  002430                          BAD18+8.
1351 023312  012604                          MOV     (SP)+,R4        ;RESTORE R4
1352 023314  000704                          BR      1$              ;LOOPING BACK
1353 023316  004537  020710          6$:     JSR     R5,SORT2        ;SORT BOTH TABLE ANYWAY
1354 023322  001424                          BAD16+8.
1355 023324  004537  020710                  JSR     R5,SORT2        ;
1356 023330  002430                          BAD18+8.
1357 023332  000207                          RTS     PC              ;RETURN TO SERVICE LOOP
1358
1359                                 ;THIS ROUTINE IS USED TO CHECK IF AN
1360                                 ;ASCII CHARACTER IS A DIGIT BETWEEN 0 AND 7.
1361                                 ;CALL
1362                                 ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
1363                                 ;       JSR     R5,CK.OCT       ;CHECK THE CHARACTER
1364                                 ;       RETURN1                 ;CHARACTER IS NOT BETWEEN 0-7
1365                                 ;       RETURN2                 ;CHARACTER IS IN R2 AS A
1366                                 ;                               ;OCTAL DIGIT
1367
1368 023334  121127  000060          CK.OCT: CMPB    (R1),#'0        ;LESS THAN ZERO?
1369 023340  103407                          BLO     1$              ;YES
1370 023342  121127  000067                  CMPB    (R1),#'7        ;GREATER THAN SEVEN?
1371 023346  101004                          BHI     1$              ;YES
1372 023350  111102                          MOVB    (R1),R2         ;GET THE CHARACTER
1373 023352  042702  177770                  BIC     #^C7,R2         ;STRIP AWAY THE ASCII
1374 023356  005725                          TST     (R5)+           ;ADJUST FOR RETURN
1375 023360  000205                  1$:     RTS     R5              ;RETURN
1376
1377                                 ;THIS ROUTINE IS USED TO CHECK AN ASCII CHARACTER
1378                                 ;AND DETERMINE IF IT IS A DIGIT BETWEEN 0 AND 9.
1379                                 ;CALL
1380                                 ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
1381                                 ;       JSR     R5,CK.DEC       ;CHECK THE CHARACTER
1382                                 ;       RETURN1                 ;NOT BETWEEN 0 AND 9
1383                                 ;       RETURN2                 ;BETWEEN 0 AND 9
1384                                 ;                               ;R2 = DIGIT
```

```
                                ;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

023410  105711         CK.CHR: TSTB    (R1)            ;''CARRIAGE RETURN''?
023412  001417                 BEQ     3$              ;YES
023414  121127  000054         CMPB    (R1),#',        ;''COMMA''?
023420  001413                 BEQ     2$              ;YES
023422  121127  000056         CMPB    (R1),#'.        ;''PERIOD''?
023426  001407                 BEQ     1$              ;YES
023430  004537  023362         JSR     R5,CK.DEC       ;''DIGIT''?
023434  000410                 BR      4$              ;NO
023436  004537  023334         JSR     R5,CK.OCT       ;OCTAL ?
023442  005725                 TST     (R5)+           ;DIGIT BETWEEN 8-9
023444  005725                 TST     (R5)+           ;DIGIT BETWEEN 0-7
023446  005725         1$:     TST     (R5)+           ;PERIOD
023450  005725         2$:     TST     (R5)+           ;COMMA
023452  005725         3$:     TST     (R5)+           ;CARRIAGE RETURN
023454  005201                 INC     R1              ;MOVE POINTER TO NEXT CHARACTER
023456  011505         4$:     MOV     (R5),R5         ;UNKNOWN CHARACTER
023460  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

023462  010446         CK.DIG: MOV     R4,-(SP)        ;SAVE R4
023464  010346                 MOV     R3,-(SP)        ;SAVE R3
023466  010246                 MOV     R2,-(SP)        ;SAVE THE MAX. SIZE ON THE STACK
```

```
1442 023470  005002                    CLR     R2              ;START WITH 0
1443 023472  005003                    CLR     R3
1444 023474  005004                    CLR     R4
1445 023476  004537  023410            JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
     023502  023576                    6$                      ;ILLEGAL CHARACTER
     023504  023604                    9$                      ;CARRIAGE RETURN
     023506  023576                    6$                      :''
     023510  023600                    7$                      :''
     023512  023516                    1$                      ;DIGIT 0-7
     023514  023516                    1$                      ;DIGIT 8-9
1446 023516  062705  000004      1$:   ADD     #4,R5           ;STEP RETURN POINTER PAST ''CR'' & ''PERIOD'' RETURNS
1447 023522  006303                2$:   ASL     R3              ;INPUT NUMBER *2
1448 023524  010346                      MOV     R3,-(SP)        ;SAVE *2
1449 023526  006303                      ASL     R3              ;*4
1450 023530  006303                      ASL     R3              ;*8
1451 023532  062603                      ADD     (SP)+,R3        ;(*2)+(*8) = *10
1452 023534  060203                      ADD     R2,R3           ;UPDATE THE INPUT NUMBER
1453 023536  004537  023410            JSR     R5,CK.CHR       ;CHECK ONE CHARACTER
     023542  023602                    8$                      ;ILLEGAL CHARACTER
     023544  023566                    5$                      ;CARRIAGE RETURN
     023546  023564                    4$                      :''
     023550  023556                    3$                      :''
     023552  023522                    2$                      ;DIGIT 0-7
     023554  023522                    2$                      ;DIGIT 8-9
1454 023556  105711                3$:   TSTB    (R1)            ;DOES A ''CR'' FOLLOW THE ''PERIOD''
1455 023560  001010                      BNE     8$              ;BR IF NOT
1456 023562  005724                      TST     (R4)+           ;INCREMENT THE RETURN
1457 023564  005724                4$:   TST     (R4)+           ;INCREMENT THE RETURN
1458 023566  005724                5$:   TST     (R4)+           ;INCREMENT THE RETURN
1459 023570  020316                      CMP     R3,(SP)         ;CHECK THE MAGNITUDE OF THE NUMBER
1460 023572  002004                      BGE     9$              ;BR IF ENTERED NUMBER TOO LARGE
1461 023574  000402                      BR      8$              ;BYPASS INCREMENT
1462 023576  005725                6$:   TST     (R5)+           ;INCREMENT RETURN PAST INVALID RETURN
1463 023600  005725                7$:   TST     (R5)+           ;INCREMENT RETURN
1464 023602  060405                8$:   ADD     R4,R5           ;SETUP RETURN POINTER
1465 023604  010302                9$:   MOV     R3,R2           ;ENTERED VALUE
1466 023606  005726                      TST     (SP)+           ;CLEAN MAX. SIZE OFF OF STACK
1467 023610  012603                      MOV     (SP)+,R3        ;RESTORE R3
1468 023612  012604                      MOV     (SP)+,R4        ;RESTORE R4
1469 023614  011505                      MOV     (R5),R5         ;GET RETURN ADDRESS
1470 023616  000205                      RTS     R5              ;RETURN
1471
1472                                .SBTTL  MACRO ROUTINES
1473
1485
1486                                .SBTTL  ERROR HANDLER ROUTINE

                                    ;;**********************************************************************
                                    ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                    ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                    ;*AND GO TO TYPERR ON ERROR
                                    ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                    ;*SW15=1         HALT ON ERROR
                                    ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                    ;*SW10=1         BELL ON ERROR
                                    ;*SW09=1         LOOP ON ERROR
                                    ;*CALL
```

```
                                ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

023620                          $ERROR:
023620  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
023622  010137  001370                  MOV     R1,DDRIVE       ;DRIVE ADDRESS IF DRIVE ERROR CALL
023626  010337  001372                  MOV     R3,ATTN         ;ATTENTION REGISTER CONTENTS
023632  013737  005552  001364          MOV     FMTDPB+12,DS.CYL  ;CURRENT CYLINDER ADDRESS
023640  113737  005551  001366          MOVB    FMTDPB+11,DS.TRK  ;REQUESTED TRACK ADDRESS
023646  005737  005564                  TST     RM.REG+RMBA     ;NON-ZERO BUFFER ADDRESS ?
023652  001406                          BEQ     7$              ;BR IF NO BUFFER ADDRESS
023654  013746  005564                  MOV     RM.REG+RMBA,-(SP)  ;BUFFER ADDRESS
023660  162716  000002                  SUB     #2,(SP)         ;DECREMENT THE ADDRESS
023664  013637  001140                  MOV     @(SP)+,$GDDAT   ;GET THE BUFFER WORD WHICH DIDN'T COMPARE
023670  105237  001117          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
023674  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
023676  013777  001116  155252          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
023704  032777  002000  155242          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
023712  001402                          BEQ     1$              ;;NO - SKIP
023714  104401  001200                  TYPE    ,$BELL          ;;RING BELL
023720  005237  001126          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
023724  011637  001132                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
023730  162737  000002  001132          SUB     #2,$ERRPC
023736  117737  155170  001130          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
023744  032777  020000  155202          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
023752  001004                          BNE     20$             ;;SKIP TYPEOUTS
023754  004737  024054                  JSR     PC,TYPERR       ;;GO TO USER ERROR ROUTINE
023760  104401  001205                  TYPE    ,$CRLF
023764                          20$:
023764  122737  000001  001230          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
023772  001007                          BNE     2$              ;;NO,SKIP APT ERROR REPORT
023774  113737  001130  024006          MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
024002  004737  024656                  JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
024006     000                  21$:    .BYTE   0
024007     000                          .BYTE   0
024010  000777                  22$:    BR      22$             ;;APT ERROR LOOP
024012  005777  155136          2$:     TST     @SWR            ;;HALT ON ERROR
024016  100002                          BPL     3$              ;;SKIP IF CONTINUE
024020  000000                          HALT                    ;;HALT ON ERROR!
024022  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
024024  032777  001000  155122  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
024032  001402                          BEQ     4$              ;;BR IF NO
024034  013716  001124                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
024040  005737  001176          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
024044  001402                          BEQ     5$              ;;BR IF NONE
024046  013716  001176                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
024052                          5$:
024052  000002                          RTI                     ;;RETURN
1487
1488                                    }
1489                            ;THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE
1490                            ;WHICH ERROR IS TO BE REPORTED, IT THEN OBTAINS, FROM THE "ERROR
1491                            ;TABLE" ($ERRTB), AND REPORTS THE APPROPIATE INFORMATION
1492                            ;CONCERNING THE ERROR.
1493
1494 024054  104412              TYPERR: SAVREG                  ;SAVE R0-R5
1495 024056  005000                      CLR     R0              ;CLEAR R0 FOR ERROR NUMBER
1496 024060  113700  001130              MOVB    $ITEMB,R0       ;ERROR NUMBER
```

```
1497 024064  005300                    DEC     R0              ;FORM INDEX FOR ERROR TABLE
1498 024066  006300                    ASL     R0
1499 024070  006300                    ASL     R0
1500 024072  006300                    ASL     R0
1501 024074  062700  006526      1$:   ADD     #$ERRTB,R0      ;FORM ADDRESS
1502 024100  012037  024114            MOV     (R0)+,2$        ;GET ERROR MESSAGE (EM) POINTER
1503 024104  001404                    BEQ     3$              ;BR IF THERE ISN'T ONE
1504 024106  104401  001205            TYPE    ,$CRLF          ;''CARRIAGE RETURN - LINE FEED
1505 024112  104401                    TYPE
1506 024114  000000      2$:   .WORD   0               ;''EM'' POINTER GOES HERE
1507 024116  012037  024132      3$:   MOV     (R0)+,4$        ;PICK UP DATA HEADER (DH) POINTER
1508 024122  001404                    BEQ     5$              ;BR IF NONE
1509 024124  104401  001205            TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
1510 024130  104401                    TYPE
1511 024132  000000      4$:   .WORD   0               ;''DH'' POINTER GOES HERE
1512 024134  012001      5$:   MOV     (R0)+,R1        ;PICKUP DATA TABLE (DT) POINTER
1513 024136  001460                    BEQ     15$             ;BR IF NONE
1514 024140  005005                    CLR     R5              ;SET INDENT SWITCH
1515 024142  012000                    MOV     (R0)+,R0        ;DATA FORMAT (DF) POINTER
1516 024144  012002                    MOV     (R0)+,R2        ;NUMBER OF DH'S TO TYPE
1517 024146  001451                    BEQ     14$             ;BR IF DH NUMBER IS 0
1518 024150  005105                    COM     R5              ;NO INDENT
1519 024152  104401  001205            TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
1520 024156  112003      6$:   MOVB    (R0)+,R3        ;NUMBER OF DATA WORDS TO TYPE
1521 024160  112004                    MOVB    (R0)+,R4        ;AND HOW TO TYPE THEM
1522 024162  006004      7$:   ROR     R4              ;OCTAL OR DECIMAL?
1523 024164  103403                    BCS     8$              ;DECIMAL
1524 024166  013146                    MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
     024170  104402                    TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
1525 024172  000402                    BR      9$
1526 024174              8$:
     024174  013146                    MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
     024176  104405                    TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
1527 024200  005303      9$:   DEC     R3              ;MORE NUMBERS TO TYPE?
1528 024202  001403                    BEQ     10$             ;NO
1529 024204  104401  036425            TYPE    ,BLNKS2         ;TYPE 2 SPACES
1530 024210  000764                    BR      7$              ;LOOP
1531 024212  005302      10$:  DEC     R2              ;MORE DH'S?
1532 024214  003431                    BLE     15$             ;NO
1533 024216  104401  001205            TYPE    ,$CRLF          ;YES--START A NEW LINE
1534 024222  005760  000002            TST     2(R0)           ;ONLY A 'DH' IN THIS REQUEST ?
1535 024226  001404                    BEQ     11$             ;BR IF YES - BYPASS THE INDENT
1536 024230  005105                    COM     R5              ;INDENT?
1537 024232  001002                    BNE     11$             ;NO
1538 024234  104401  036425            TYPE    ,BLNKS2         ;TYPE 2 SPACES
1539 024240  012037  024246      11$:  MOV     (R0)+,12$       ;GET NEXT DH
1540 024244  104401                    TYPE                    ;AND TYPE IT
1541 024246  000000      12$:  .WORD   0               ;DH POINTER GOES HERE
1542 024250  005710                    TST     (R0)            ;TYPE A 'DT' ?
1543 024252  001003                    BNE     13$             ;BR IF A 'DT'
1544 024254  062700  000004            ADD     #4,R0           ;INCREMENT THE 'DF' POINTER
1545 024260  000754                    BR      10$             ;SEE IF END OF 'DF' BLOCK
1546 024262  104401  001205      13$:  TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
1547 024266  005705                    TST     R5              ;INDENT?
1548 024270  001332                    BNE     6$              ;NO
1549 024272  104401  036425      14$:  TYPE    ,BLNKS2         ;TYPE 2 SPACES
1550 024276  000727                    BR      6$              ;LOOP
```

```
1551 024300  104413                15$:    RESREG                  ;RESTORE R0-R5
1552 024302  000207                        RTS     PC              ;RETURN
1553
1554                               .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
                                   ;*

024304  105737  001173             $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
024310  100002                            BPL     1$              ;;BR IF YES
024312  000000                            HALT                    ;;HALT HERE IF NO TERMINAL
024314  000430                            BR      3$              ;;LEAVE
024316  010046                    1$:     MOV     R0,-(SP)        ;;SAVE R0
024320  017600  000002                    MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
024324  122737  000001  001230            CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
024332  001011                            BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
024334  132737  000100  001231            BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
024342  001405                            BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
024344  010037  024354                    MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
024350  004737  024646                    JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
024354  000000                    61$:    .WORD   0               ;;MESSAGE ADDRESS
024356  132737  000040  001231    62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
024364  001003                            BNE     60$             ;;YES,SKIP TYPE OUT
024366  112046                    2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
024370  001005                            BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
024372  005726                            TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
024374  012600                    60$:    MOV     (SP)+,R0        ;;RESTORE R0
024376  062716  000002            3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
024402  000002                            RTI                     ;;RETURN
024404  122716  000011            4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
024410  001430                            BEQ     8$
024412  122716  000200                    CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
024416  001006                            BNE     5$
024420  005726                            TST     (SP)+           ;;POP  <CR><LF> EQUIV
024422  104401                            TYPE                    ;;TYPE A CR AND LF
024424  001205                            $CRLF
024426  105037  024634                    CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
024432  000755                            BR      2$              ;;GET NEXT CHARACTER
024434  004737  024516            5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
024440  123726  001172            6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
024444  001350                            BNE     2$              ;;IF NO GO GET NEXT CHAR.
024446  013746  001170                    MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                  ;;AND THE NULL CHAR.
024452  105366  000001            7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
024456  002770                            BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
```

```
024460  004737  024516                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
024464  105337  024634                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
024470  000770                          BR      7$              ;;LOOP

                                ;HORIZONTAL TAB PROCESSOR

024472  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
024476  004737  024516          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
024502  132737  000007  024634          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
024510  001372                          BNE     9$              ;;TAB STOP
024512  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
024514  000724                          BR      2$              ;;GET NEXT CHARACTER
024516                          $TYPEC:
024516  105777  154436                  TSTB    @$TKS           ;;CHAR IN KYBD BUFFER?
024522  100022                          BPL     10$             ;;BR IF NOT
024524  017746  154432                  MOV     @$TKB,-(SP)     ;;GET CHAR
024530  042716  177600                  BIC     #177600,(SP)    ;;STRIP EXTRANEOUS BITS
024534  122716  000023                  CMPB    #$XOFF,(SP)     ;;WAS CHAR XOFF
024540  001012                          BNE     102$            ;;BR IF NOT
024542                          101$:
024542  105777  154412                  TSTB    @$TKS           ;;WAIT FOR CHAR
024546  100375                          BPL     101$
024550  117716  154406                  MOVB    @$TKB,(SP)      ;;GET CHAR
024554  042716  177600                  BIC     #177600,(SP)    ;;STRIP IT
024560  122716  000021                  CMPB    #$XON,(SP)      ;;WAS IT XON?
024564  001366                          BNE     101$            ;;BR IF NOT
024566                          102$:
024566  005726                          TST     (SP)+           ;;FIX STACK
024570                          10$:
024570  105777  154370                  TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
024574  100375                          BPL     10$
024576  116677  000002  154362          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
024604  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
024612  001003                          BNE     1$              ;;BRANCH IF NO
024614  105037  024634                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
024620  000406                          BR      $TYPEX          ;;EXIT
024622  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
024630  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
024632  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
024634  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
024636  000207                  $TYPEX: RTS     PC

1555
1556                            .SBTTL  APT COMMUNICATIONS ROUTINE

                                ;;**********************************************************************
024640  112737  000001  025104  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
024646  112737  000001  025102  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
024654  000403                          BR      $ATYC
024656  112737  000001  025104  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
024664                          $ATYC:
024664  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
024666  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
024670  105737  025102                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
024674  001450                          BEQ     5$              ;;IF NOT:  BR
024676  122737  000001  001230          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
024704  001031                          BNE     3$              ;;IF NOT:  BR
```

```
024706  132737  000100  001231          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
024714  001425                          BEQ     3$              ;;IF NOT:  BR
024716  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
024722  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
024730  005737  001210          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
024734  001375                          BNE     1$              ;;IF NOT:  WAIT
024736  010037  001224                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
024742  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
024744  001376                          BNE     2$
024746  163700  001224                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
024752  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
024754  010037  001226                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
024760  012737  000004  001210          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
024766  000413                          BR      5$
024770  017637  000004  025014  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
024776  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
025004  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
025010  004737  024304                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
025014  000000                  4$:     .WORD   0
025016                          5$:
025016  105737  025104          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
025022  001416                          BEQ     12$             ;;IF NOT:  BR
025024  005737  001230                  TST     $ENV            ;;RUNNING UNDER APT?
025030  001413                          BEQ     12$             ;;IF NOT:  BR
025032  005737  001210          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
025036  001375                          BNE     11$             ;;IF NOT:  WAIT
025040  017637  000004  001212          MOV     @4(SP),$FATAL   ;;GET ERROR #
025046  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
025054  005237  001210                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
025060  105037  025104          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
025064  105037  025103                  CLRB    $LFLG           ;;CLEAR LOG FLAG
025070  105037  025102                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
025074  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
025076  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
025100  000207                          RTS     PC              ;;RETURN
025102     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
025103     000          $LFLG:  .BYTE   0               ;;LOG FLAG
025104     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
                                .EVEN
            000200      APTSIZE = 200
            000001      APTENV  = 001
            000100      APTSPOOL= 100
            000040      APTCSUP = 040
1557
1558                            .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

025106  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
025112  116637  000001  025331          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
025120  112637  025333                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
025124  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
025130  000406                          BR      $TYPON
025132  112737  000001  025331  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
025140  112737  000006  025333          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
025146  112737  000005  025330  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
025154  010346                          MOV     R3,-(SP)        ;;SAVE R3
025156  010446                          MOV     R4,-(SP)        ;;SAVE R4
025160  010546                          MOV     R5,-(SP)        ;;SAVE R5
025162  113704  025333                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
025166  005404                          NEG     R4
025170  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
025174  110437  025332                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
025200  113704  025331                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
025204  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
025210  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
025212  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
025214  000404                          BR      3$              ;;GO DO MSB
025216  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
025220  006105                          ROL     R5
025222  006105                          ROL     R5
025224  010503                          MOV     R5,R3
025226  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
025230  105337  025332                  DECB    $OMODE          ;;TYPE THIS DIGIT?
025234  100016                          BPL     7$              ;;BR IF NO
025236  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
025242  001002                          BNE     4$              ;;TEST FOR 0
025244  005704                          TST     R4              ;;SUPPRESS THIS 0?
025246  001403                          BEQ     5$              ;;BR IF YES
025250  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
025252  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
025256  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
025262  110337  025326                  MOVB    R3,8$           ;;SAVE FOR TYPING
025266  104401  025326                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
025272  105337  025330          7$:     DECB    $OCNT           ;;COUNT BY 1
025276  003347                          BGT     2$              ;;BR IF MORE TO DO
025300  002402                          BLT     6$              ;;BR IF DONE
025302  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
025304  000744                          BR      2$              ;;GO DO THE LAST DIGIT
025306  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
025310  012604                          MOV     (SP)+,R4        ;;RESTORE R4
025312  012603                          MOV     (SP)+,R3        ;;RESTORE R3
025314  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
```

```
025322  012616                          MOV     (SP)+,(SP)
025324  000002                          RTI                     ;;RETURN
025326     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
025327     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
025330     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
025331     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
025332  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
1559
1560                            .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

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

025334                          $TYPDS:
025334  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
025336  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
025340  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
025342  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
025344  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
025346  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
025352  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
025356  100004                          BPL     1$              ;;BR IF INPUT IS POS.
025360  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
025362  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
025370  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
025372  012703  025550                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
025376  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
025402  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
025404  016001  025540                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
025410  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
025412  002402                          BLT     4$              ;;BR IF DONE
025414  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
025416  000774                          BR      3$
025420  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
025422  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
025424  001002                          BNE     5$              ;;FALL THROUGH IF 0
025426  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
025430  100407                          BMI     7$              ;;BR IF YES
025432  106316                  5$:     ASLB    (SP)            ;;MSD?
025434  103003                          BCC     6$              ;;BR IF NO
025436  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
025444  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
025450  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
025454  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
025456  005720                          TST     (R0)+           ;;JUST INCREMENTING
025460  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
025464  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
025466  003002                          BGT     8$              ;;GO TO EXIT
025470  010502                          MOV     R5,R2           ;;GET THE LSD
025472  000764                          BR      6$              ;;GO CHANGE TO ASCII
```

```
025474  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
025476  100003                          BPL     9$              ;;BR IF NO
025500  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
025506  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
025510  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
025512  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
025514  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
025516  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
025520  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
025522  104401  025550                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
025526  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
025534  012616                          MOV     (SP)+,(SP)
025536  000002                          RTI                     ;;RETURN TO USER
025540  023420                  $DTBL:  10000.
025542  001750                          1000.
025544  000144                          100.
025546  000012                          10.
025550                          $DBLK:  .BLKW   4
1561
1562                                    .SBTTL  TTY INPUT ROUTINE

                                        ;;**********************************************************************
                                        .ENABL  LSB
025560  000000                  $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
025562  000000                  $TKQIN: .WORD   0               ;;INPUT POINTER
025564  000000                  $TKQOUT: .WORD  0               ;;OUTPUT POINTER
025566                          $TKQSRT: .BLKB  7               ;;TTY KEYBOARD QUEUE
        025575                  $TKQEND=.
                                .EVEN

                                ;*TK INITIALIZE ROUTINE
                                ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
                                ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
                                ;
                                ;*CALL:
                                ;*      JSR     PC,$TKINT
                                ;*      RETURN
                                ;
025576  005037  025560          $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
025602  012737  025566  025562          MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
025610  013737  025562  025564          MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
025616  012737  025646  000060          MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
025624  012737  000200  000062          MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
025632  005777  153324                  TST     @$TKB           ;;CLEAR DONE FLAG
025636  012777  000100  153314          MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
025644  000207                          RTS     PC              ;;RETURN TO CALLER

                                ;*TK SERVICE ROUTINE
                                ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
                                ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
                                ;*IT IN THE QUEUE.
                                ;*IF THE CHARACTER IS A "CONTROL-C" (^C) $TKINT IS CALLED AND
                                ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (@CNTLC)
                                ;
025646  117746  153310          $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
025652  042716  177600                  BIC     #^C177,(SP)     ;;STRIP THE JUNK
025656  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL C?
```

```
025662  001007                          BNE     1$              ;;BRANCH IF NO
025664  104401  026767                  TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
025670  004737  025576                  JSR     PC,$TKINT       ;;INIT THE KEYBOARD
025674  005726                          TST     (SP)+           ;;CLEAN UP STACK
025676  000177  153472                  JMP     @CNTLC          ;;CONTROL C RESTART
025702  021627  000007          1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
025706  001004                          BNE     2$              ;;BRANCH IF NO
025710  022737  000176  001154          CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
025716  001500                          BEQ     6$              ;;GO TO SWR CHANGE

025720                          2$:
025720  022737  000007  025560          CMP     #7,$TKCNT       ;;IS THE QUEUE FULL?
025726  001004                          BNE     3$              ;;BRANCH IF NO
025730  104401  001200                  TYPE    ,$BELL          ;;RING THE TTY BELL
025734  005726                          TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
025736  000451                          BR      5$              ;;EXIT
025740  021627  000023          3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
025744  001021                          BNE     32$             ;;BRANCH IF NO
025746  005077  153206                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
025752  005726                          TST     (SP)+           ;;CLEAN CHAR OFF STACK
025754  105777  153200          31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
025760  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
025762  117746  153174                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
025766  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
025772  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
025776  001366                          BNE     31$             ;;BRANCH IF NO
026000  012777  000100  153152          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
026006  000002                          RTI                     ;;RETURN
026010  005237  025560          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
026014  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
026020  002405                          BLT     4$              ;;BRANCH IF YES
026022  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
026026  003002                          BGT     4$              ;;BRANCH IF YES
026030  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
026034  112677  177522          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
026040  005237  025562                  INC     $TKQIN          ;;UPDATE THE POINTER
026044  023727  025562  025575          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
026052  001003                          BNE     5$              ;;BRANCH IF NO
026054  012737  025566  025562          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
026062  000002                  5$:     RTI                     ;;RETURN

                                ;;**********************************************************************
                                ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
                                ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
                                ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
026064  022737  000176  001154  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
026072  001124                          BNE     15$             ;;EXIT IF NOT
026074  105777  153060                  TSTB    @$TKS           ;;IS A CHAR WAITING?
026100  100121                          BPL     15$             ;;IF NOT, EXIT
026102  117746  153054                  MOVB    @$TKB,-(SP)     ;;YES
026106  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
026112  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
026116  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
                                                                ;;AND EXIT

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

```
                                ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
                                ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
                                ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
026120  123727  001150  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
026126  001674                          BEQ     2$              ;;BRANCH IF YES
026130  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
026132  004737  025576                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
026136  005077  153016                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
026142  112737  000001  001151          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR

026150  104401  027001                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
026154  104401  027006  $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
026160  013746  000176                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
026164  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
026166  104401  027017                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
026172  005046          19$:    CLR     -(SP)           ;;CLEAR COUNTER
026174  005046                          CLR     -(SP)           ;;THE NEW SWR
026176  105777  152756          7$:     TSTB    @$TKS           ;;CHAR THERE?
026202  100375                          BPL     7$              ;;IF NOT TRY AGAIN

026204  117746  152752                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
026210  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII

026214  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL-C?
026220  001015                          BNE     9$              ;;BRANCH IF NOT
026222  104401  026767                  TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
026226  062706  000006                  ADD     #6,SP           ;;CLEAN UP STACK
026232  123727  001151  000001          CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
026240  001003                          BNE     8$              ;;BRANCH IF NO
026242  012777  000100  152710          MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
026250  000177  153120          8$:     JMP     @CNTLC          ;;CONTROL-C RESTART


026254  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
026260  001005                          BNE     10$             ;;BRANCH IF NOT
026262  104401  026774                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
026266  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
026272  000737                          BR      19$             ;;LET'S TRY IT AGAIN


026274  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
026300  001022                          BNE     16$             ;;BRANCH IF NO
026302  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
026306  001403                          BEQ     11$             ;;BRANCH IF YES
026310  016677  000002  152636          MOV     2(SP),@SWR      ;;SAVE NEW SWR
026316  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
026322  104401  001205          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
026326  123727  001151  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
026334  001003                          BNE     15$             ;;BRANCH IF NOT
026336  012777  000100  152614          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
026344  000002          15$:    RTI                     ;;RETURN
026346  004737  024516          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
026352  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
026356  002420                          BLT     18$             ;;BRANCH IF YES
026360  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
026364  003015                          BGT     18$             ;;BRANCH IF YES
026366  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
```

```
026372  005766  000002          TST     2(SP)           ;;IS THIS THE FIRST CHAR
026376  001403                  BEQ     17$             ;;BRANCH IF YES
026400  006316                  ASL     (SP)            ;;NO, SHIFT PRESENT
026402  006316                  ASL     (SP)            ;;   CHAR OVER TO MAKE
026404  006316                  ASL     (SP)            ;;   ROOM FOR NEW ONE.
026406  005266  000002  17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
026412  056616  177776          BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
026416  000667                  BR      7$              ;;GET THE NEXT ONE
026420  104401  001204  18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
026424  000720                  BR      20$             ;;SIMULATE CONTROL-U
                        .DSABL  LSB

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

026426  011646  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
026430  016666  000004  000002  MOV     4(SP),2(SP)     ;;THE PS
026436  005066  000004          CLR     4(SP)           ;;GET READY FOR A CHARACTER
026442  005046                  CLR     -(SP)           ;;PUT NEW PS ON STACK
026444  012746  026452          MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
026450  000002                  RTI                     ;;POP NEW PC AND PS
026452                  64$:
026452  005737  025560  1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
026456  001775                  BEQ     1$
026460  005337  025560          DEC     $TKCNT          ;;DECREMENT THE COUNTER
026464  117766  177074  000004  MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
026472  005237  025564          INC     $TKQOUT         ;;UPDATE THE POINTER
026476  023727  025564  025575  CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
026504  001003                  BNE     2$              ;;BRANCH IF NO
026506  012737  025566  025564  MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
026514  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

026516  010346  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
026520  005046          CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
026522  012703  026752  1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
026526  022703  026767  2$:     CMP     #$TTYIN+15,R3   ;;BUFFER FULL?
026532  101456                  BLOS    4$              ;;BR IF YES
026534  104410                  RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
026536  112613                  MOVB    (SP)+,(R3)      ;;GET CHARACTER
026540  122713  000177  10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
026544  001022                  BNE     5$              ;;BR IF NO
026546  005716                  TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
026550  001007                  BNE     6$              ;;BR IF NO
026552  112737  000134  026750  MOVB    #'\,9$          ;;TYPE A BACK SLASH
026560  104401  026750          TYPE    ,9$
```

```
026564  012716  177777          MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
026570  005303          6$:     DEC     R3              ;;BACKUP BY ONE
026572  020327  026752          CMP     R3,#$TTYIN      ;;STACK EMPTY?
026576  103434                  BLO     4$              ;;BR IF YES
026600  111337  026750          MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
026604  104401  026750          TYPE    ,9$             ;;GO TYPE
026610  000746                  BR      2$              ;;GO READ ANOTHER CHAR.
026612  005716          5$:     TST     (SP)            ;;RUBOUT KEY SET?
026614  001406                  BEQ     7$              ;;BR IF NO
026616  112737  000134  026750  MOVB    #'\,9$          ;;TYPE A BACK SLASH
026624  104401  026750          TYPE    ,9$
026630  005016                  CLR     (SP)            ;;CLEAR THE RUBOUT KEY
026632  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
026636  001003                  BNE     8$              ;;BR IF NO
026640  104401  026774          TYPE    ,$CNTLU         ;;TYPE A CONTROL ''U''
026644  000726                  BR      1$              ;;GO START OVER
026646  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A ''^R''?
026652  001011                  BNE     3$              ;;BRANCH IF NO
026654  105013                  CLRB    (R3)            ;;CLEAR THE CHARACTER
026656  104401  001205          TYPE    ,$CRLF          ;;TYPE A ''CR'' & ''LF''
026662  104401  026752          TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
026666  000717                  BR      2$              ;;GO PICKUP ANOTHER CHACTER
026670  104401  001204          4$:     TYPE    ,$QUES          ;;TYPE A '?'
026674  000712                  BR      1$              ;;CLEAR THE BUFFER AND LOOP
026676  111337  026750          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
026702  104401  026750          TYPE    ,9$
026706  122723  000015          CMPB    #15,(R3)+       ;;CHECK FOR RETURN
026712  001305                  BNE     2$              ;;LOOP IF NOT RETURN
026714  105063  177777          CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
026720  104401  001206          TYPE    ,$LF            ;;TYPE A LINE FEED
026724  005726                  TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
026726  012603                  MOV     (SP)+,R3        ;;RESTORE R3
026730  011646                  MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
026732  016666  000004  000002  MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
026740  012766  026752  000004  MOV     #$TTYIN,4(SP)
026746  000002                  RTI                     ;;RETURN
026750     000          !9$:    .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
026751     000          ;       .BYTE   0               ;;TERMINATOR
026752                  $TTYIN: .BLKB   15              ;;RESERVE 15 BYTES FOR TTY INPUT
026767     136     103     015  $CNTLC: .ASCIZ  /^C/<15><12>    ;;CONTROL ''C''
026774     136     125     015  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL ''U''
027001     136     107     015  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL ''G''
027006     015     012     123  $MSWR:  .ASCIZ  <15><12>/SWR = /
027017     040     040     116  $MNEW:  .ASCIZ  /  NEW = /
1563
1564                            .SBTTL  POWER DOWN AND UP ROUTINES

                                ;;**********************************************************************
                                ;POWER DOWN ROUTINE
027030  012737  027174  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
027036  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
027044  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
027046  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
027050  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
027052  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
027054  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
027056  010546                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
```

```
027060  017746  152070                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
027064  010637  027200                  MOV     SP,$SAVR6       ;;SAVE SP
027070  012737  027102  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
027076  000000                          HALT
027100  000776                          BR      .-2             ;;HANG UP

                        ;;**********************************************************************
                        ;POWER UP ROUTINE
027102  012737  027174  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
027110  013706  027200                  MOV     $SAVR6,SP       ;;GET SP
027114  005037  027200                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
027120  005237  027200          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
027124  001375                          BNE     1$              ;;OF  WORD
027126  012677  152022                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
027132  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
027134  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
027136  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
027140  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
027142  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
027144  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
027146  012737  027030  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
027154  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
027162  104401                          TYPE                    ;;REPORT THE POWER FAILURE
027164  027202          $PWRMG: .WORD   $POWER                  ;;POWER FAIL MESSAGE POINTER
027166  012716                          MOV     (PC)+,(SP)      ;;RESTART AT BEGIN2
027170  007020          $PWRAD: .WORD   BEGIN2                  ;;RESTART ADDRESS
027172  000002                          RTI
027174  000000          $ILLUP: HALT                            ;;THE POWER UP SEQUENCE WAS STARTED
027176  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
027200  000000          $SAVR6: 0                               ;;PUT THE SP HERE
1565 027202     200     042     120    $POWER: .ASCIZ  <CRLF>/''POWER UP''/
1566
1567                                    .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


027216  010046          $SUPRS: MOV     R0,-(SP)        ;;SAVE R0
027220  016600  000004          MOV     4(SP),R0        ;;PICKUP THE POINTER
027224  105710          1$:     TSTB    (R0)            ;;TERMINATEOR?
027226  001403                  BEQ     2$              ;;BR IF YES
027230  122720  000060          CMPB    #'0,(R0)+       ;;IS THIS AN ASCII ''0'' ?
027234  001773                  BEQ     1$              ;;BR IF YES
027236  005300          2$:     DEC     R0              ;;BACKUP BY ''1''
027240  010037  027246          MOV     R0,3$           ;;SAVE FOR TYPING
027244  104401                  TYPE                    ;;GO TYPE
027246  000000          3$:     .WORD   0               ;;ASCIZ POINTER GOES HERE
027250  012600                  MOV     (SP)+,R0        ;;RESTORE R0
027252  012616                  MOV     (SP)+,(SP)      ;;RESTORE THE STACK
027254  000207                  RTS     PC              ;;RETURN
1568
1569                                    .SBTTL  SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE
```

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

027256  016637  000002  027306  $SB2D:  MOV     2(SP),1$        ;;SAVE BINARY NUMBER
027264  012746  027306          MOV     #1$,-(SP)       ;;SET POINTER
027270  004737  027312          JSR     PC,@#$DB2D      ;;CALL DOUBLE LENGTH CONVERT
027274  062716  000005          ADD     #5,(SP)         ;;ONLY ALLOW FIVE CHARACTERS
027300  012666  000002          MOV     (SP)+,2(SP)     ;;PICKUP POINTER
027304  000207                  RTS     PC              ;;RETURN
027306  000000  000000  1$:     .WORD   0,0
1570
1571                            .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


027312  104412          $DB2D:  SAVREG                  ;;SAVE REGISTERS
027314  016602  000002          MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
027320  012700  027472          MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
027324  010066  000002          MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
027330  012201                  MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
027332  012202                  MOV     (R2)+,R2
027334  012737  000012  027410  MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
027342  012704  027422          MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
027346  012705  027424          MOV     #$TNPWR+2,R5
027352  005003          1$:     CLR     R3              ;;CLEAR PARTIAL
027354  161401          2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
027356  005602                  SBC     R2
027360  161502                  SUB     (R5),R2
027362  002402                  BLT     3$              ;;BR IF TEN POWER TO LARGE
027364  005203                  INC     R3              ;;ADD 1 TO PARTIAL
027366  000772                  BR      2$              ;;LOOP
027370  062401          3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
027372  005502                  ADC     R2
027374  062402                  ADD     (R4)+,R2
027376  022525                  CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
027400  052703  000060          BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
027404  110320                  MOVB    R3,(R0)+        ;;SAVE IT
027406  005327                  DEC     (PC)+           ;;DONE?
027410  000000          4$:     .WORD   0
027412  001357                  BNE     1$              ;;BR IF NO
027414  105020                  CLRB    (R0)+           ;;TERMINATOR
```

```
027416  104413                  RESREG                  ;;RESTORE REGISTERS
027420  000207                  RTS     PC              ;;RETURN
027422  145000          $TNPWR: 145000                  ;;1.0E09
027424  035632                  35632
027426  160400                  160400                  ;;1.0E08
027430  002765                  2765
027432  113200                  113200                  ;;1.0E07
027434  000230                  230
027436  041100                  041100                  ;;1.0E06
027440  000017                  17
027442  103240                  103240                  ;;1.0E05
027444  000001                  1
027446  023420                  23420                   ;;1.0E04
027450  000000                  0
027452  001750                  1750                    ;;1.0E03
027454  000000                  0
027456  000144                  144                     ;;1.0E02
027460  000000                  0
027462  000012                  12                      ;;1.0E01
027464  000000                  0
027466  000001                  1                       ;;1.0E00
027470  000000                  0
027472                  $DECVL: .BLKB   12.             ;;RESERVE STORAGE FOR ASCIZ STRING
1572
1573                            .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES

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

027506                          $SAVREG:
027506  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
027510  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
027512  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
027514  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
027516  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
027520  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
027522  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
027526  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
027532  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
027536  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
027542  000002                          RTI

                                ;*RESTORE R0-R5
                                ;*CALL:
                                ;*      RESREG
```

```
027544
027544  012666  000022          $RESREG:
027550  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
027554  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
027560  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
027564  012605                          MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
027566  012604                          MOV     (SP)+,R5        ;;POP STACK INTO R5
027570  012603                          MOV     (SP)+,R4        ;;POP STACK INTO R4
027572  012602                          MOV     (SP)+,R3        ;;POP STACK INTO R3
027574  012601                          MOV     (SP)+,R2        ;;POP STACK INTO R2
027576  012600                          MOV     (SP)+,R1        ;;POP STACK INTO R1
027600  000002                          MOV     (SP)+,R0        ;;POP STACK INTO R0
                                        RTI
1574
1575                            .SBTTL  TRAP DECODER

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

027602  010046          $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
027604  016600  000002          MOV     2(SP),R0        ;;GET TRAP ADDRESS
027610  005740                  TST     -(R0)           ;;BACKUP BY 2
027612  111000                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
027614  006300                  ASL     R0              ;;POSITION FOR INDEXING
027616  016000  027636          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
027622  000200                  RTS     R0              ;;GO TO ROUTINE


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

027624  011646          $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
027626  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
027634  000002                  RTI                     ;;RESTORE THE PSW

                                .SBTTL  TRAP TABLE

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

                                ;       ROUTINE
                                ;       -------
027636  027624          $TRPAD: .WORD   $TRAP2
027640  024304                  $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
027642  025132                  $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
027644  025106                  $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
027646  025146                  $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
027650  025334                  $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

027652  026154                  $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

027654  026064                  $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
027656  026426                  $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
027660  026516                  $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
027662  027506                  $SAVREG ;;CALL=SAVREG   TRAP+12(104412) SAVE R0-R5 ROUTINE
027664  027544                  $RESREG ;;CALL=RESREG   TRAP+13(104413) RESTORE R0-R5 ROUTINE
```

```
1576
1583
```

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

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

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

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

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

027666  000000  000000  000000  RMERRS: .WORD   0,0,0,0

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

027676      000                 DRVACT: .BYTE   0               ;DRIVE 0
027677      000                         .BYTE   0               ;DRIVE 1
027700      000                         .BYTE   0               ;DRIVE 2
027701      000                         .BYTE   0               ;DRIVE 3
027702      000                         .BYTE   0               ;DRIVE 4
027703      000                         .BYTE   0               ;DRIVE 5
027704      000                         .BYTE   0               ;DRIVE 6
027705      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

027706      000                 DRVSTA: .BYTE   0               ;DRIVE 0
027707      000                         .BYTE   0               ;DRIVE 1
027710      000                         .BYTE   0               ;DRIVE 2
027711      000                         .BYTE   0               ;DRIVE 3
027712      000                         .BYTE   0               ;DRIVE 4
027713      000                         .BYTE   0               ;DRIVE 5
027714      000                         .BYTE   0               ;DRIVE 6
027715      000                         .BYTE   0               ;DRIVE 7

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

```
                        
027716  000             DRVTYP: .BYTE   0               ;DRIVE 0
027717  000                     .BYTE   0               ;DRIVE 1
027720  000                     .BYTE   0               ;DRIVE 2
027721  000                     .BYTE   0               ;DRIVE 3
027722  000                     .BYTE   0               ;DRIVE 4
027723  000                     .BYTE   0               ;DRIVE 5
027724  000                     .BYTE   0               ;DRIVE 6
027725  000                     .BYTE   0               ;DRIVE 7

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

027726  000             DPINT:  .BYTE   0               ;DRIVE 0
027727  000                     .BYTE   0               ;DRIVE 1
027730  000                     .BYTE   0               ;DRIVE 2
027731  000                     .BYTE   0               ;DRIVE 3
027732  000                     .BYTE   0               ;DRIVE 4
027733  000                     .BYTE   0               ;DRIVE 5
027734  000                     .BYTE   0               ;DRIVE 6
027735  000                     .BYTE   0               ;DRIVE 7

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

027736  000             DPRQS:  .BYTE   0               ;DRIVE 0
027737  000                     .BYTE   0               ;DRIVE 1
027740  000                     .BYTE   0               ;DRIVE 2
027741  000                     .BYTE   0               ;DRIVE 3
027742  000                     .BYTE   0               ;DRIVE 4
027743  000                     .BYTE   0               ;DRIVE 5
027744  000                     .BYTE   0               ;DRIVE 6
027745  000                     .BYTE   0               ;DRIVE 7

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

027746  000000          TRNSWT: .WORD   0

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

027750  000000          SRCHWT: .WORD   0

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

027752  000             ACTDRV: .BYTE   0

                        ;SOFTWARE TIMER ROUTINE ACTIVE FLAG (ACTSTR=1 BYTE)
```

```
                                        ;ACTSTR=0 IF SOFTWARE TIMER ROUTINE IS INACTIVE
                                        ;ACTSTR>0 IF SOFTWARE TIMER ROUTINE IS ACTIVE

027753  000                     ACTSTR: .BYTE   0

                                ;UNLOAD FLAG (ULDFLG=8 BYTES)
                                        ;ULDFLG=0 IF NO UNLOAD COMMAND
                                        ;ULDFLG>0 IF UNLOAD COMMAND IN PROGRESS
                                        ;ULDFLG<0 IF UNLOAD COMMAND IN WAIT QUEUE

027754  000                     ULDFLG: .BYTE   0               ;DRIVE 0
027755  000                             .BYTE   0               ;DRIVE 1
027756  000                             .BYTE   0               ;DRIVE 2
027757  000                             .BYTE   0               ;DRIVE 3
027760  000                             .BYTE   0               ;DRIVE 4
027761  000                             .BYTE   0               ;DRIVE 5
027762  000                             .BYTE   0               ;DRIVE 6
027763  000                             .BYTE   0               ;DRIVE 7

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

027764  000                     LACNT:  .BYTE   0               ;DRIVE 0
027765  000                             .BYTE   0               ;DRIVE 1
027766  000                             .BYTE   0               ;DRIVE 2
027767  000                             .BYTE   0               ;DRIVE 3
027770  000                             .BYTE   0               ;DRIVE 4
027771  000                             .BYTE   0               ;DRIVE 5
027772  000                             .BYTE   0               ;DRIVE 6
027773  000                             .BYTE   0               ;DRIVE 7

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

027774  000000                  SAVEFG: .WORD   0

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

027776  177777                  SEEKFG: .WORD   -1

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

030000  177777                  TIMER:  .WORD   -1               ;DRIVE 0
030002  177777                          .WORD   -1              ;DRIVE 1
030004  177777                          .WORD   -1              ;DRIVE 2
030006  177777                          .WORD   -1              ;DRIVE 3
030010  177777                          .WORD   -1              ;DRIVE 4
030012  177777                          .WORD   -1              ;DRIVE 5
030014  177777                          .WORD   -1              ;DRIVE 6
```

```
030016  177777                          .WORD   -1              ;DRIVE 7
141
142                                     ;DATA TRANSFER UNDERWAY INDICATOR (DTUW=1 WORD)
143                                             ;DTUW<0 IF NO DATA TRANSFER UNDERWAY
144                                             ;DTUW=+N (WHERE N=0 TO 7) IMPLIES DATA TRANSFER UNDERWAY ON DRIVE N
145
146 030020  177777                  DTUW:   .WORD   -1
147
148                                     ;ATTENTION BITS TABLE (ATABIT=8 BYTES)
149                                             ;THIS TABLE CONTAINS THE CORRESPONDING BIT TO EACH DRIVES
150                                             ;ATTENTION BIT
151
152 030022      001                 ATABIT: .BYTE   1       ;DRIVE 0
153 030023      002                         .BYTE   2       ;DRIVE 1
154 030024      004                         .BYTE   4       ;DRIVE 2
155 030025      010                         .BYTE   10      ;DRIVE 3
156 030026      020                         .BYTE   20      ;DRIVE 4
157 030027      040                         .BYTE   40      ;DRIVE 5
158 030030      100                         .BYTE   100     ;DRIVE 6
159 030031      200                         .BYTE   200     ;DRIVE 7
160
161                                     ;FSRM TO RH11/RH70 "MASSBUS CONTROL BUS PARITY ERRORS" (MCPE) ALLOWED BEFORE
162                                     ;CALLING IT FATAL (MCPEMX=1 WORD)
163
164 030032  000003                  MCPEMX: .WORD   3
165
166                                     ;STORAGE FOR RMADR (THE FIRST ADDRESS (776700) OF THE RH/RM),
167                                     ;RMVEC (THE VECTOR ADDRESS (254)), AND RMVEC+2 (THE BR LEVEL (5)).
168
169 030034  176700                  RMADR:  .WORD   176700
170 030036  000254  000240          RMVEC:  .WORD   254,5*32.
171
172                                     ;MAXIMUM NUMBER OF LOOK AHEADS ALLOWED IS 4 (MXLACT=1 WORD)
173 030042  000004                  MXLACT: .WORD   4
174
175                                     ;MAXIMUM DELTA DELAY IS 8 SECTORS (MXDLTA=1 WORD)
176 030044  001000                  MXDLTA: .WORD   8.*64.
177
178                                     ;MINIMUM DELTA DELAY IS 2 SECTORS (MNDLTA=1 WORD)
179 030046  000200                  MNDLTA: .WORD   2*64.
180
181                                     ;MAXIMUM SEARCH FOR I/O WINDOW IS 5 SECTORS (MXWNDW=1 WORD)
182 030050  000005                  MXWNDW: .WORD   5
183
184                                     ;DEFINITIONS OF THE RH/RM ADDRESS INDEXES
185
186         000000                      RMCS1=0                 ;CONTROL AND STATUS REGISTER #1 (DRIVE REG. 00)
187         000002                      RMWC=2                  ;WORD COUNT REGISTER (NOT A DRIVE REG)
188         000004                      RMBA=4                  ;UNIBUS ADDRESS REGISTER (NOT A DRIVE REG)
189         000006                      RMDA=6                  ;DESIRED SECTOR/TRACK ADDRESS REGISTER (DRIVE REG. 05)
190         000010                      RMCS2=10                ;CONTROL AND STATUS REGISTER #2 (NOT A DRIVE REG)
191         000012                      RMDS=12                 ;DRIVE STATUS REGISTER (DRIVE REG 01)
192         000014                      RMER1=14                ;ERROR REGISTER #1 (DRIVE REG. 02)
193         000016                      RMAS=16                 ;ATTENTION SUMMARY PSEUDO REGISTER (DRIVE REG. 04)
194         000020                      RMLA=20                 ;LOOK AHEAD REGISTER (DRIVE REG. 07)
195         000022                      RMDB=22                 ;DATA BUFFER REGISTER (NOT A DRIVE REG.)
196         000024                      RMMR1=24                ;MAINTAINABILITY REGISTER (DRIVE REG. 03)
```

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

                                .SBTTL  RH/RM DRIVER INITIALIZATION CODE

                                ;THIS ROUTINE WILL DETERMINE WHICH RM DRIVES ARE
                                ;AVAILABLE FOR TESTING AND SET THE DRVSTA INDICATOR
                                ;TO THE PROPER STATE FOR EACH DRIVE.
                                ;NOTE: THIS ROUTINE CALLS DRVINT
                                ;
                                ;CALL
                                ;
                                ;       JSR     PC,RMINIT
                                ;       RETURN
                                ;
                                ;NOTE: THE 'P' OR 'L' CLOCK MUST BE STARTED
                                ;
030052  104412                  RMINIT: SAVREG                  ;SAVE R0 - R5
030054  013746  177776                  MOV     PS,-(SP)        ;SAVE THE PRESENT PROCESSOR STATUS
030060  012737  000240  177776          MOV     #<5*32.>,PS     ;CHANGE THE PRIORITY TO 5
030066  004737  035740                  JSR     PC,CLRQUE       ;CLEAR ALL REQUEST QUEUES
030072  012701  027666                  MOV     #RMERRS,R1      ;FIRST ADDRESS TO BE CLEARED
030076  012702  027776                  MOV     #SEEKFG,R2      ;LAST ADDRESS TO BE CLEARED
030102  005021                  1$:     CLR     (R1)+           ;CLEAR
030104  020102                          CMP     R1,R2           ;ARE WE DONE?
030106  103775                          BLO     1$              ;BR IF NO
030110  012702  030020                  MOV     #DTUW,R2        ;LAST ADDRESS
030114  012721  177777          2$:     MOV     #-1,(R1)+       ;INITIALIZE
030120  020102                          CMP     R1,R2           ;DONE?
030122  101774                          BLOS    2$              ;LOOP IF NO
030124  005037  027706                  CLR     DRVSTA          ;SET ALL DRIVES TO OFFLINE
030130  005037  027710                  CLR     DRVSTA+2
030134  005037  027712                  CLR     DRVSTA+4
030140  005037  027714                  CLR     DRVSTA+6
030144  013703  030036                  MOV     RMVEC,R3        ;SETUP THE RH/RM VECTOR
030150  012723  032706                  MOV     #ISR,(R3)+
030154  013713  030040                  MOV     RMVEC+2,(R3)
030160  013704  030034                  MOV     RMADR,R4        ;FIRST ADDRESS OF RH/RM
030164  012764  000040  000010          MOV     #CLR,RMCS2(R4)  ;MASSBUS INIT
030172  005001                          CLR     R1              ;START WITH DRIVE 0
030174  004037  030264          3$:     JSR     R0,DRVINT       ;INIT THE DRIVE
030200  000401                          BR      4$              ;'DVA' NOT SET OR PARITY ERROR
030202  000402                          BR      5$              ;NORMAL RETURN
030204  105061  027706          4$:     CLRB    DRVSTA(R1)      ;SET DRIVE STATUS TO OFFLINE
030210  005201                  5$:     INC     R1              ;GO TO NEXT DRIVE
030212  042701  177770                  BIC     #^C7,R1         ;MASK OUT UNUSED BITS
030216  001366                          BNE     3$              ;BR IF MORE DRIVES TO GO
030220  012701  000007                  MOV     #7,R1           ;START WITH DRIVE 7
030224  005037  177776                  CLR     PS              ;CLEAR THE PROCESSOR STATUS
030230  105761  027726          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 RM05/3/2. IF IT IS, A "READ-IN PRESET" IS ISSUED AND FMT16
                ;IS SET TO A "1". THEN MOL, DPR, DRY, AND VV ARE CHECKED TO
                ;INSURE THEY ARE ALL ON A "1". AND DEPENDING ON THEIR STATE,
                ;DRVSTA IS SET TO THE PROPER CONDITION.
                ;CALL
                ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
                ;       MOV     RMADR,R4        ;UNIBUS ADDRESS OF RH/RM (RMCS1)
                ;       JSR     R0,DRVINT       ;CALLED BY A JSR
                ;       RETURN1                 ;ERROR OCCURRED (PARITY)
                ;       RETURN2                 ;NORMAL RETURN
                ;

                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
                        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 PORT RM02 ?
                        BEQ     2$              ;BR IF SO
                        CMP     #24025,R5       ;DUAL PORT RM02 ?
                        BEQ     2$              ;BR IF SO
                        MOVB    #7,DRVTYP(R1)   ;SET RM05 INDICATOR
                        CMP     #20027,R5       ;SINGLE PORT RM05 ?
                        BEQ     2$              ;BR IF YES
                        CMP     #24027,R5       ;DUAL PORT RM05 ?
                        BEQ     2$              ;BR IF YES
```

```
                                                                K 8
    311 030446  112761  177777  027716          MOVB    #-1,DRVTYP(R1)  ;SET INDICATOR TO 'OTHER'
    312 030454  000446                          BR      6$              ;EXIT
    313
    314 030456  012746  000121          2$:     MOV     #121,-(SP)      ;DO A ''READ-IN PRESET''
    315 030462  004037  035064                  JSR     R0,WRT.RM
    316 030466  000000                          RMCS1
    317 030470  030616                          8$
    318 030472  012746  010000                  MOV     #BIT12,-(SP)    ;SET FMT16=1
    319 030476  004037  035064                  JSR     R0,WRT.RM
    320 030502  000032                          RMOF
    321 030504  030616                          8$
    322 030506  004037  034704                  JSR     R0,RD.RM        ;READ RMDS
    323 030512  000012                          RMDS
    324 030514  030616                          8$
    325 030516  012605                          MOV     (SP)+,R5        ;AND SAVE IT IN R5
    326 030520  100015                          BPL     4$              ;BR IF ATA=0
    327 030522  116164  030022  000016          MOVB    ATABIT(R1),RMAS(R4)     ;CLEAR ATTENTION BIT
    328 030530  004037  034704                  JSR     R0,RD.RM        ;FIND OUT WHY ATA=1
    329 030534  000014                          RMER1
    330 030536  030616                          8$
    331 030540  006126                          ROL     (SP)+           ;IS IT UNSAFE?
    332 030542  100004                          BPL     4$              ;BR IF NOT
    333 030544  112761  177777  027706          MOVB    #-1,DRVSTA(R1)  ;SET UNSAFE INDICATOR
    334 030552  000407                          BR      6$              ;EXIT
    335
    336 030554  005105                  4$:     COM     R5              ;CHECK MOL, DPR, DRY, AND VV
    337 030556  042705  167077                  BIC     #^C<BIT12!BIT08!BIT07!BIT06>,R5
    338 030562  001003                          BNE     6$              ;BR IF MOL, DPR, DRY, OR VV IS CLEAR
    339 030564  112761  000001  027706          MOVB    #1,DRVSTA(R1)   ;SET DRIVE STATUS TO ONLINE
    340 030572  005720                  6$:     TST     (R0)+           ;STEP OVER THE ERROR RETURN
    341 030574  000410                          BR      8$              ;EXIT
    342 030576  006301                  7$:     ASL     R1              ;CHANGE INDEX TO ADDRESS WORDS
    343 030600  012761  060000  030000          MOV     #60000,TIMER(R1)        ;START 2 SEC TIMER
    344 030606  006201                          ASR     R1              ;RESTORE R1
    345 030610  112761  177777  027726          MOVB    #-1,DPINT(R1)   ;SET PORT INITIALIZE INIDICATOR
    346 030616  012605                  8$:     MOV     (SP)+,R5        ;RESTORE R5
    347 030620  000200                          RTS     R0              ;EXIT
    348
    349                                 ;REQUEST PRE-PROCESSOR-HANDLES SUBSYSTEM REQUEST
    350                                 ;
    351                                 ;CALL
    352                                 ;
    353                                 ;       JSR     R0,RM05         ;CALL THE RM05 DRIVER
    354                                 ;       PNTADR                  ;ADDRESS OF POINTER OF DRIVES PARAMETER BLOCK
    355                                 ;       RETURN1                 ;RETURN HERE IF QUEUE IS FULL
    356                                 ;       RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE OR THERE
    357                                                                 ;IS AN ERROR CONDITION
    358
    359 030622  013746  177776          RM05:   MOV     PS,-(SP)        ;SAVE THE CALLING STATUS
    360 030626  013737  030040  177776          MOV     RMVEC+2,PS      ;DON'T ALLOW ANY RM INTERRUPTS
    361 030634  112737  000001  027752          MOVB    #1,ACTDRV       ;SET ''ACTIVE DRIVER'' FLAG
    362 030642  104412                          SAVREG                  ;SAVE R0 - R5
    363 030644  011002                          MOV     (R0),R2         ;PICKUP THE DRIVE PARAMETER BLOCK POINTER
    364 030646  005062  000016                  CLR     16(R2)          ;CLEAR THE STATUS/ERROR INDICATOR
    365 030652  111201                          MOVB    (R2),R1         ;PICKUP THE DRIVE NUMBER
    366 030654  013704  030034                  MOV     RMADR,R4        ;UNIBUS ADDRESS OF RMCS1
    367 030660  105761  027706                  TSTB    DRVSTA(R1)      ;CHECK DRIVES STATUS
```

```
                                                BGT     1$              ;BR IF ONLINE
                                                TSTB    ULDFLG(R1)      ;UNLOAD COMMAND IN QUEUE?
                                                BNE     3$              ;BR IF YES
                                                TSTB    DPINT(R1)       ;TRYING TO INIT THE DRIVE
                                                BNE     5$              ;BR IF YES
                                                JSR     R0,DRVINT       ;GO INIT. THE DRIVE
                                                BR      4$              ;ERROR RETURN
                                                TSTB    DRVSTA(R1)      ;IS DRIVE STATUS ONLINE?
                                                BLE     6$              ;BR IF NOT
                                        1$:     TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
                                                BNE     5$              ;BR IF YES
                                                MOV     R1,RMCS2(R4)    ;SELECT THE DRIVE
                                                JSR     R0,DRVQUE       ;PUT THIS REQUEST IN QUEUE
                                                BR      9$              ;QUEUE IS FULL
                                                CMPB    #103,2(R2)      ;IS THIS REQ. FOR AN UNLOAD?
                                                BNE     2$              ;BR IF NO
                                                MOVB    #-1,ULDFLG(R1)  ;SET THE "UNLOAD IN QUEUE" FLAG
                                        2$:     TSTB    DRVACT(R1)      ;IS THIS DRIVE ACTIVE?
                                                BNE     8$              ;BR IF YES
                                                JSR     PC,OPT          ;CALL THE OPTIMIZER
                                                BR      8$
                                        3$:     MOV     #BIT15!BIT13,16(R2)     ;SET THE "UNLOAD IN QUEUE" ERROR FLAG
                                                BR      8$              ;EXIT
                                        4$:     JSR     PC,C17          ;GO HANDLE THE PARITY ERROR
                                                BR      8$
                                        5$:     JSR     R0,DRVQUE       ;PUT REQUEST IN QUEUE
                                                BR      9$              ;QUEUE IS FULL
                                                BIT     #BIT06,(R4)     ;IE BIT SET ?
                                                BNE     8$              ;YES
                                                JSR     PC,SET.IE       ;SET THE INTERRUPT
                                                BR      8$              ;RETURN
                                        6$:     TSTB    DRVSTA(R1)      ;SEE IF DRIVE OFFLINE OR UNSAFE
                                                BLT     7$              ;BR IF UNSAFE
                                                MOV     #BIT15!BIT14,16(R2)     ;SET OFFLINE ERROR INDICATOR
                                                TSTB    DRVTYP(R1)      ;SEE IF OFFLINE OR NONEXISTENT
                                                BNE     8$              ;BR IF OFFLINE
                                                MOV     #BIT15!BIT01,16(R2)     ;REPORT DRIVE NONEXISTENT
                                                BR      8$              ;GO TO EXIT
                                        7$:     MOV     #BIT15!BIT12,16(R2)     ;DRIVE IS UNSAFE
                                        8$:     RESREG                  ;RESTORE R0 - R5
                                                TST     (R0)+           ;SETUP FOR NORMAL RETURN
                                                BR      10$             ;FINISH UP, THEN EXIT
                                        9$:     RESREG                  ;RESTORE R0 - R5
                                        10$:    TST     (R0)+           ;CORRECT THE RETURN ADDRESS
                                                CLRB    ACTDRV          ;CLEAR "ACTIVE DRIVER" FLAG
                                                MOV     (SP)+,PS        ;RETURN "PS" TO USER LEVEL
                                                RTS     R0              ;RETURN TO CALLER

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

```
                                                CLRB    DPRQS(R1)           ;RESET THE PORT REQ FLAG ****
                                                JSR     PC,GETREQ           ;GET "DPB" POINTER OF REQUEST
                                                TST     R2                  ;IS THERE A REQUEST IN QUEUE?
                                                BEQ     7$                  ;NO--BR TO EXIT
                                                MOV     R1,RMCS2(R4)        ;LOAD THE DRIVE ADDRESS *******
                                                MOV     #111,RMCS1(R4)      ;CLEAR THE DRIVE
                                                BIT     #BIT11,RMCS1(R4)            ;DVA SET ?
                                                BEQ     5$                  ;TO PROT REQUEST ,IF NOT
                                        10$:    TSTB    DRVSTA(R1)          ;IS DRIVE ONLINE?
                                                BGT     1$                  ;YES
                                                JSR     PC,POPQUE           ;NO--REMOVE REQUEST FROM QUEUE
                                                MOV     #BIT15!BIT14,16(R2)     ;SET OFFLINE STATUS/ERROR INDICATOR
                                                TSTB    DRVSTA(R1)          ;IS DRIVE UNSAFE ?
                                                BPL     8$                  ;BR TO EXIT IF NOT
                                                MOV     #BIT15!BIT12,16(R2)     ;SET UNSAFE STATUS/ERROR INDICATOR
                                                BR      8$                  ;BR TO EXIT
                                        1$:
                                        ;       MOV     #111,-(SP)          ;LOAD COMMAND ONTO THE STACK
                                        ;       JSR     R0,WRT.RM           ;LOAD THE REGISTER
                                        ;       RMCS1                       ;REGISTER INCREMENT
                                        ;       6$                          ;ERROR RETURN ADDRESS
                                        ;       BIT     #BIT11,(R4)         ;DRIVE AVAILABLE ?
                                        ;       BEQ     9$                  ;BR IF NOT
                                                CMPB    #150,2(R2)          ;IS THE REQUEST FOR I/O?
                                                BLT     2$                  ;YES
                                                JSR     PC,CI4              ;CALL THE COMMAND INITIATOR
                                                BR      8$                  ;BR TO EXIT
                                        2$:     TST     DTUW                ;DATA TRANSFER UNDERWAY?
                                                BGE     4$                  ;YES--GO START A SEARCH
                                                TST     SEEKFG              ;DO IMPLIED SEEKS?
                                                BMI     3$                  ;YES
                                                JSR     R0,LA               ;NO--DO LOOK AHEAD
                                                BR      8$                  ;RETURN HERE ON A PARITY ERROR
                                                BR      4$                  ;GO START A SEARCH
                                        3$:     JSR     PC,CI1              ;START A DATA TRANSFER
                                                BR      8$
                                        4$:     JSR     PC,CI3              ;START A SEARCH
                                                BR      8$                  ;GO TO THE EXIT
                                        5$:     MOVB    #-1,DPRQS(R1)       ;SET PORT REQUEST INDICATOR
                                                MOV     R1,R3               ;SET UP TO ADDRESS WORDS
                                                ASL     R3                  ;CONVERT TO WORD INDEX
                                                MOV     #60000,TIMER(R3)            ;START 10 SEC TIMER
                                                BR      7$                  ;EXIT
                                        6$:     JSR     PC,CI7              ;PROCESS THE PARITY ERROR
                                        7$:     BIT     #BIT06,(R4)         ;SEE IF 'IE' ALREADY SET
                                                BNE     8$                  ;BR IF SET
                                                JSR     PC,SET.IE           ;SET "IE" WITHOUT A "TRE"
                                        8$:     MOV     (SP)+,PS            ;RESTORE PROC. STATUS
                                                RESREG                      ;RESTORE R0 - R5
                                                RTS     PC

                                        ;COMMAND INITIATOR
                                        ;
                                        ;CALL
                                        ;       MOV     #DRVNUM,R1          ;DRIVE NUMBER
                                        ;       MOV     #DPB,R2             ;ADDRESS OF DPB
                                        ;       JSR     PC,CI?              ;CI?= CI1,CI3, OR CI4
```

```
                                        ;                               ;WHERE:
                                        ;                               ;CI1=DATA TRANSFER
                                        ;                               ;CI2=SEARCH REQUESTED BY DATA XFER
                                        ;                               ;CI4=NOT DATA TRANSFER
031352  004737  036134              CI1:  JSR     PC,POPQUE               ;REMOVE REQUEST FROM "DRIVES WAIT" QUEUE
031356  010237  027746                    MOV     R2,TRNSWT               ;PUT REQ. IN TRANSFER WAIT QUEUE
031362  010203                          MOV     R2,R3                   ;DPB ADDRESS TO R3
031364  013704  030034                  MOV     RMADR,R4                ;RMCS1 ADDRESS
031370  010164  000010                  MOV     R1,RMCS2(R4)            ;SELECT DRIVE
031374  062703  000004                  ADD     #4,R3                   ;DESIRED WORD COUNT
031400  062704  000002                  ADD     #2,R4                   ;RMWC ADDRESS
031404  012324                          MOV     (R3)+,(R4)+             ;LOAD WORD COUNT
031406  012324                          MOV     (R3)+,(R4)+             ;LOAD BUFFER ADDRESS
031410  012346                          MOV     (R3)+,-(SP)             ;LOAD SECTOR AND TRACK
031412  004037  035064                  JSR     R0,WRT.RM               ;CALL THE LOAD(WRITE) ROUTINE
031416  000006                          RMDA                            ;INDEX OF REGISTER TO LOAD
031420  032174                          CI7                             ;ERROR RETURN ADDRESS
031422  012346                          MOV     (R3)+,-(SP)             ;LOAD CYLINDER ADDRESS
031424  004037  035064                  JSR     R0,WRT.RM
031430  000034                          RMDC
031432  032174                          CI7
031434  016246  000002                  MOV     2(R2),-(SP)             ;LOAD "COMMAND+GO", "A17&A16", AND "PSEL"
031440  004037  035064                  JSR     R0,WRT.RM
031444  000000                          RMCS1
031446  032174                          CI7
031450  010137  030020                  MOV     R1,DTUW                 ;SET "DATA TRANSFER UNDERWAY"
031454  000137  032136                  JMP     CI5
031460  013704  030034              CI3:  MOV     RMADR,R4                ;RMCS1 ADDRESS
031464  010164  000010                  MOV     R1,RMCS2(R4)            ;SELECT DRIVE
031470  016246  000012                  MOV     12(R2),-(SP)            ;DESIRED CYLINDER ADDRESS
031474  004037  035064                  JSR     R0,WRT.RM
031500  000034                          RMDC
031502  032174                          CI7
031504  116203  000010                  MOVB    10(R2),R3               ;PICKUP SECTOR ADDRESS
                                        SUB     MXWNDW,R3               ;BACKUP BY MAX. SEARCH FOR I/O WINDOW
                                        BGE     1$
                                        ADD     #32.,R3
031510  010346                    1$:   MOV     R3,-(SP)                ;COMBINE THE ADJUSTED SECTOR WITH
031512  042716  177740                  BIC     #177740,(SP)            ;CHOP OF ALL EXENTED BITS ,IF ANY
031516  116266  000011  000001          MOVB    11(R2),1(SP)            ;THE DESIRED TRACK
031524  004037  035064                  JSR     R0,WRT.RM               ;LOAD DESIRED TRACK & SECTOR
031530  000006                          RMDA
031532  032174                          CI7
031534  012746  000131                  MOV     #131,-(SP)              ;START A SEARCH
031540  004037  035064                  JSR     R0,WRT.RM
031544  000000                          RMCS1
031546  032174                          CI7
031550  156137  030022  027750          BISB    ATABIT(R1),SRCHWT       ;SET "SEARCH WAIT" KEY
031556  000567                          BR      CI5
031560  013704  030034              CI4:  MOV     RMADR,R4                ;RMCS1 ADDRESS
031564  010164  000010                  MOV     R1,RMCS2(R4)            ;SELECT DRIVE
031570  116203  000002                  MOVB    2(R2),R3                ;PICKUP THE REQUESTED COMMAND
031574  122703  000131                  CMPB    #131,R3                 ;IS IT A SEARCH COMMAND?
031600  001007                          BNE     1$                      ;BR IF NO
031602  016246  000010                  MOV     10(R2),-(SP)            ;LOAD DESIRED TRACK & SECTOR
031606  004037  035064                  JSR     R0,WRT.RM
```

```
                                                RMDA
                                                CI7
                                                BR      2$              ;GO LOAD CYLINDER
000105                          1$:             CMPB    #105,R3         ;IS IT A SEEK COMMAND
                                                BNE     3$              ;BR IF NO
000012                          2$:             MOV     12(R2),-(SP)    ;LOAD DESIRED CYLINDER
035064                                          JSR     R0,WRT.RM
                                                RMDC
                                                CI7
                                                BR      CI6
000115                          3$:             CMPB    #115,R3         ;IS IT AN "OFFSET" COMMAND?
                                                BNE     4$              ;BR IF NO
034704                                          JSR     R0,RD.RM        ;MERGE THE OFFSET VALUE INTO RMOF
                                                RMOF                    ;BUT DON'T CHANGE THE UPPER
                                                CI7
000001                                          MOVB    1(R2),(SP)      ;BYTE WHEN LOADING THE
035064                                          JSR     R0,WRT.RM       ;REGISTER (RMOF)
                                                RMOF
                                                CI7
                                                BR      CI6             ;GO START THE COMMAND
000107                          4$:             CMPB    #107,R3         ;IS IT A "RECALIBRATE" COMMAND?
                                                BEQ     CI6             ;BR IF YES
000117                                          CMPB    #117,R3         ;IS IT A RETURN TO CENTER?
                                                BEQ     CI6             ;BR IF YES
000103                                          CMPB    #103,R3         ;IS IT AN "UNLOAD" COMMAND?
                                                BNE     5$              ;BR IF NO
000001  027676                                  MOVB    #1,DRVACT(R1)   ;SET THE DRIVE ACTIVE INDICATOR
027706                                          CLRB    DRVSTA(R1)      ;PUT DRIVE STATUS TO OFFLINE
000001  027754                                  MOVB    #1,ULDFLG(R1)   ;SET "UNLOAD IN PROGRESS" FLAG
                                                MOV     R3,-(SP)        ;START THE "UNLOAD" COMMAND
035064                                          JSR     R0,WRT.RM
                                                RMCS1
                                                CI7
                                                RTS     PC              ;RETURN TO USER
000143                          5$:             CMPB    #143,R3         ;IS IT A "SET FORMAT" COMMAND?
                                                BNE     6$              ;BR IF NO
034704                                          JSR     R0,RD.RM        ;READ THE OFFSET REGISTER
                                                RMOF
                                                CI7
000001  000001                                  MOVB    1(R2),1(SP)     ;COMBINE "FMT16","ECI", AND "HCI"
035064                                          JSR     R0,WRT.RM       ;LOAD "FMT16", "ECI", AND/OR "HCI".
                                                RMOF
                                                CI7
                                                BR      12$
000141                          6$:             CMPB    #141,R3         ;IS IT A "GET REGISTER" COMMAND?
                                                BNE     10$             ;BR IF NO
000006                          7$:             MOV     6(R2),R3        ;POINTS TO 1ST ADDRESS OF WHERE
                                                                        ;TO PUT THE REGISTER(S)
000010  032044                                  MOVB    10(R2),9$       ;INIT. THE INDEX FOR THE FIRST REG.
000011                                          MOVB    11(R2),R5       ;INDEX OF LAST REG. TO MOVE
034704                          8$:             JSR     R0,RD.RM        ;READ RH/RM REGISTER
000000                          9$:             RMCS1                   ;INDEX OF REG. TO READ
                                                CI7
                                                MOV     (SP)+,(R3)+     ;GET THE CONTENTS OF RH/RM REG.
032044                                          CMP     9$,R5           ;LAST REG. BEEN READ?
                                                BEQ     12$             ;GET OUT IF YES
000002  032044                                  ADD     #2,9$           ;INCREASE THE INDEX BY 2
```

```
                                                BR      8$                      ;LOOP--MORE TO READ
032066  000764
032070  122703  000145          10$:    CMPB    #145,R3                 ;IS IT A "SELECT DRIVE" COMMAND?
032074  001405                          BEQ     12$                     ;BR IF YES
032076  010346                  11$:    MOV     R3,-(SP)                ;LOAD THE COMMAND
032100  004037  035064                  JSR     R0,WRT.RM
032104  000000                          RMCS1
032106  032174                          CI7
032110  004737  036134          12$:    JSR     PC,POPQUE               ;REMOVE REQ. FROM QUEUE
032114  052762  000200  000016          BIS     #BIT07,16(R2)           ;SET THE "DONE" BIT
032122  005737  027774                  TST     SAVEFG                  ;SAVE THE RH/RM REGISTERS?
032126  100002                          BPL     13$                     ;BR IF NO
032130  004737  035256                  JSR     PC,SVRH70               ;YES--GO SAVE THE REGISTERS
032134  000207                  13$:    RTS     PC                      ;RETURN TO USER
032136  006301                  CI5:    ASL     R1
032140  012761  060000  030000          MOV     #60000,TIMER(R1)                ;SET A ONE SECOND TIMER
032146  006201                          ASR     R1
032150  112761  000001  027676          MOVB    #1,DRVACT(R1)           ;SET THE DRIVE ACTIVE
032156  000207                          RTS     PC                      ;RETURN TO THE USER
032160  010346                  CI6:    MOV     R3,-(SP)                ;LOAD THE COMMAND
032162  004037  035064                  JSR     R0,WRT.RM
032166  000000                          RMCS1
032170  032174                          CI7
032172  000761                          BR      CI5
032174  032764  010000  000010  CI7:    BIT     #BIT12,RMCS2(R4)                ;DRIVE NON-EXISTENT ?
                                        ;
                                        BNE     CI8                     ;BR IF YES
032202  005702                  1$:     TST     R2                      ;ANYTHING IN QUEUE ?
                                        ;       BEQ     CI7B                    ;BR IF NOT
032204  001001                          BNE     2$                      ;BR IF QUEUE IS THERE
032206  000207                          RTS     PC                      ;OTHERWISE EXIT
032210  012762  104000  000016  2$:     MOV     #BIT15!BIT11,16(R2)     ;SET "PARITY" ERROR INDICATOR
                                        ;       JSR     PC,SVRH70               ;GO SAVE THE RH/RM REGISTERS
032216  012746  000111          CI7B:   MOV     #111,-(SP)              ;DO A "DRIVE CLEAR"
032222  004037  035064                  JSR     R0,WRT.RM
032226  000000                          RMCS1
032230  032274                          CI8
032232  004737  036016          2$:     JSR     PC,EMPTYQ               ;EMPTY THE QUEUE
032236  105061  027736                  CLRB    DPRQS(R1)               ;CLEAR THE PORT REQUEST FLAG
032242  105061  027754                  CLRB    ULDFLG(R1)              ;CLEAR THE UNLOAD IN QUEUE FLAG
032246  105061  027676                  CLRB    DRVACT(R1)              ;DRIVE IS IDLE
032252  020237  027746                  CMP     R2,TRNSWT               ;IF THIS DRIVE HAD AN I/O REQUEST
                                        ;       CMP     R1,DTUW                 ;IF THIS DRIVE HAD AN I/O REQUEST
032256  001005                          BNE     1$                      ;IN PROGRESS CLEAR ALL OF THE FLAGS
032260  005037  027746                  CLR     TRNSWT
032264  012737  177777  030020          MOV     #-1,DTUW
032272  000207                  1$:     RTS     PC
032274  104412                  CI8:    SAVREG                          ;SAVE R0 - R5
032276  032764  010000  000010          BIT     #BIT12,RMCS2(R4)                ;IS 'NED' SET ?
                                        ;       BNE     1$                      ;BR IF YES
032304  005001                          CLR     R1
032306  005003                          CLR     R3
032310  105761  027676          1$:     TSTB    DRVACT(R1)              ;DRIVE ACTIVE?
                                        ;       BEQ     5$                      ;BR IF NO
032314  001003                          BNE     22$                     ;BR IF IN ACTIVE
032316  105761  027736                  TSTB    DPRQS(R1)               ;PORT REQUEST
032322  001443                          BEQ     5$                      ;BR IF NOT
032324  013702  027746          22$:    MOV     TRNSWT,R2               ;GET THE "TRANSFER WAIT" QUEUE
032330  020137  030020                  CMP     R1,DTUW                 ;DID THIS DRIVE HAVE AN I/O IN PROGRESS?
```

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

                                        ;LOOK AHEAD ROUTINE
                                        ;
                                        ;CALL
                                        ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
                                        ;       MOV     #DPB,R2         ;POINT TO DPB
                                        ;       JSR     R0,LA           ;GO CHECK THE WINDOW
                                        ;       RETURN1                 ;ERROR RETURN
                                        ;       RETURN2                 ;START A SEARCH
                                        ;       RETURN3                 ;START A DATA TRANSFER

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

```
                        116203  000010                  MOVB    10(R2),R3       ;GET DESIRED SECTOR ADDRESS AND
                        000303                          SWAB    R3              ;MULT. BY 64--ALIGN WITH
                        006203                          ASR     R3              ;LOOK AHEAD REGISTER
                        006203                          ASR     R3
                        012737  000340  177776          MOV     #340,PS         ;PRIORITY LEVEL "7"
                        004037  034704          6$:     JSR     R0,RD.RM        ;READ LOOK AHEAD REGISTER
                        000020                          RMLA
                        032700                          4$
                        021664  000020                  CMP     (SP),RMLA(R4)   ;CORRECT LA NUMBER ?
                        001402                          BEQ     7$              ;YES
                        005726                          TST     (SP)+           ;NO,CLEAR STACK
                        000415                          BR      3$
                        162603                  7$:     SUB     (SP)+,R3        ;CALCULATE THE DELTA
                        002002                          BGE     1$
                        062703  004000                  ADD     #<32.*64.>,R3   ;MAKE THE DELTA POSITIVE
                        023703  030044          1$:     CMP     MXDLTA,R3       ;CHECK THE DELTA TO SEE
                        002406                          BLT     3$              ;IF IT IS WITHIN THE
                        023703  030046                  CMP     MNDLTA,R3       ;WINDOW---IF YES, ZERO
                        002003                          BGE     3$              ;THE LOOK AHEAD COUNT
                        105061  027764          2$:     CLRB    LACNT(R1)       ;AND TAKE THE I/O EXIT
                        005720                          TST     (R0)+
                        005720                  3$:     TST     (R0)+           ;ADJUST THE RETURN ADDRESS
                        000402                          BR      5$              ;EXIT
                        004737  032174          4$:     JSR     PC,C17          ;PROCESS THE ERROR
                        000200                  5$:     RTS     R0              ;RETURN

                                                ;INTERRUPT SERVICE ROUTINE

                        112737  000001  027752  ISR:     MOVB    #1,ACTDRV       ;SET "ACTIVE DRIVER" FLAG
                        104412                          SAVREG                  ;SAVE R0 - R5
                        013704  030034                  MOV     RMADR,R4        ;ADDRESS OF RHSCS1
                        013701  030020                  MOV     DTUW,R1         ;GET "DATA TRANSFER UNDERWAY" INDICATOR
                        002403                          BLT     1$              ;BR IF NO DATA TRANSFER UNDERWAY
                        004737  032752                  JSR     PC,TD           ;CALL TRANSFER DONE
                        000402                          BR      2$              ;EXIT
                        004737  033122          1$:     JSR     PC,SC           ;CALL SPECIAL CONDITIONS
                        104413                  2$:     RESREG                  ;RESTORE R0 - R5
                        105037  027752                  CLRB    ACTDRV          ;CLEAR "ACTIVE DRIVER" FLAG
                        000002                          RTI                     ;RETURN

                                                ;TRANSFER DONE ROUTINE

                        105061  027676          TD:     CLRB    DRVACT(R1)      ;SET DRIVE ACTIVE INDICATOR TO IDLE
                        012737  177777  030020          MOV     #-1,DTUW        ;NO DATA TRANSFERS UNDERWAY
                        006301                          ASL     R1
                        012761  177777  030000          MOV     #-1,TIMER(R1)   ;CANCEL TIMEOUT
                        006201                          ASR     R1
                        013702  027746                  MOV     TRNSWT,R2       ;GET "DPB" ADDRESS FROM THE
                        005037  027746                  CLR     TRNSWT          ;TRANSFER WAIT QUEUE--CLEAR QUEUE
                        052762  000200  000016          BIS     #BIT07,16(R2)   ;SET DONE
                        010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT THE DRIVE
                        004037  034704                  JSR     R0,RD.RM        ;TRANSFER ERROR(TRE=1)?
                        000000                          RMCS1
                        032174                          C17
                        006126                          ROL     (SP)+
                        100417                          BMI     3$              ;BR IF YES
                        005737  027774                  TST     SAVEFG          ;SAVE THE RH/RM REGISTERS?
```

```
                                        BPL     1$              ;BR IF NO
                                        JSR     PC,SVRH70       ;YES--SAVE THE REGISTERS
                                1$:
                                        JSR     PC,GETREQ       ;GET DPB POINTER
                                        TST     R2              ;ENTRY FOR DRIVE ?
                                        BEQ     2$              ;BR IF NOT
                                        JSR     PC,OPT          ;CALL OPTIMIZER
                                        BR      SC              ;CHECK OTHER DRIVES
                                ;THE RELEASE DRIVE COMMAND IS FORECD TO ENTER FOR DUAL PORT OPERATION
                                2$:     MOV     #113,(R4)       ;RELEASE THE DRIVE
                                        BR      SC              ;CHECK FOR OTHER DRIVES
                                3$:     BIS     #BIT15!BIT06,16(R2)     ;SET DATA ERROR FLAG
                                        JSR     PC,EMPTYQ       ;EMPTY THE "DRIVE'S WAIT" QUEUE
                                        JSR     PC,SVRH70       ;SAVE THE RH/RM REGISTERS
                                        MOV     #40111,(R4)     ;ISSUE A "DRIVE CLEAR"
                                        MOV     #113,(R4)       ;ISSUE A RELEASE TO THE DRIVE
                                        BR      SC              ;CHECK FOR OTHER DRIVES


                                ;SPECIAL CONDITION ROUTINE

                                SC:     MOVB    RMAS(R4),R3     ;READ "RMAS"
                                        BNE     2$              ;BR IF ANY 'ATA' BITS SET
                                        JSR     R0,RD.RM        ;READ CONTROL AND STATUS REGISTER
                                        RMCS1
                                        CI8
                                ;       1$                      ;EXIT IF FAIL TO READ
                                        ROLB    (SP)+           ;IS "IE"=1?
                                        BMI     1$              ;YES, NO DRIVES TO CHECK
                                        JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
                                        EMT     1               ;REPORT AN ILLEGAL INTERRUPT
                                        JSR     PC,SET.IE       ;SET INTERRUPT ENABLE
                                1$:     RTS     PC              ;RETURN
                                2$:     CLR     -(SP)           ;PROCESS ALL DRIVES THAT HAVE
                                        MOVB    R3,(SP)         ;AN "ATA"=1
                                        MOV     #1,R3
                                        CLR     R1
                                SC3:    BIT     R3,(SP)         ;ATA=1?
                                        BNE     SC5             ;YES
                                SC4:    INC     R1              ;MOVE TO THE NEXT DRIVE
                                        ASLB    R3
                                        BNE     SC3             ;BR IF MORE TO CHECK?
                                        TST     (SP)+           ;CLEAN OFF THE STACK
                                        RTS     PC              ;RETURN TO USER
                                SC5:    TSTB    DPINT(R1)       ;INITIALIZING THE DRIVE ?
                                        BEQ     1$              ;BR IF NOT
                                        JMP     SC13            ;PROCESS THE DRIVE
                                1$:     TSTB    DPRQS(R1)       ;PORT REQUEST OUTSTANDING ?
                                        BEQ     2$              ;BR IF NOT
                                        JMP     SC13            ;START THE OUTSTANDING COMMAND
                                2$:     TSTB    DRVSTA(R1)      ;CHECK THE DRIVE STATUS
                                        BGT     5$              ;BR IF ONLINE
                                        TSTB    ULDFLG(R1)      ;UNLOAD IN PROGRESS?
                                        BLE     5$              ;BR IF NOT
                                        JSR     PC,GETREQ       ;GET DPB POINTER
                                        JSR     PC,SVRH70       ;SAVE THE RH/RM REGISTERS
```

```
                                                JSR     PC,SC12         ;SAVE RMDS, RMER1, RMER2, AND RMMR2
                                                                        ;ALSO DO A DRIVE INIT (DRVINT)
                                                TSTB    DRVSTA(R1)      ;DID DRIVE COME ONLINE?
                                                BLE     6$              ;NO
                                                BIT     #BIT14,RMERRS   ;WAS THERE AN ERROR?
                                                BNE     3$              ;BR IF ERROR
                                                JMP     SC11            ;NO ERROR
                                        ;
                                        3$:     MOV     RMERRS+2,R5     ;YES -- PICKUP RMER1 AND
                                                BR      SC6A            ;GO PROCESS THE ERROR
                                        5$:     TSTB    DRVACT(R1)      ;DRIVE ACTIVE WITH COMMAND OR ERROR RECOVERY ?
                                                BNE     SC6             ;BR IF EITHER
                                                JSR     PC,SC12         ;SAVE RMDS, RMER1, RMER2, AND RMMR2
                                                                        ;ALSO DO A DRVINT
                                        6$:     TSTB    DPINT(R1)       ;TRYING TO INIT THE DRIVE ?
                                                BNE     SC4             ;BR IF YES, CHECK ON MORE DRIVES
                                                TSTB    DRVSTA(R1)      ;CHECK ON DRIVE'S STATUS
                                                BMI     7$              ;BR IF UNSAFE
                                                BIT     #BIT13,RMERRS+4 ;ADDRESS PLUG CHANGED ?
                                                BNE     8$              ;BR IF YES
                                        ;       MOV     #113,-(SP)      ;RELEASE COMMAND
                                                MOV     #111,-(SP)      ;DRIVE CLEAR
                                                JSR     R0,WRT.RM       ;WRITE THE COMMAND INTO RMCS1
                                                RMCS1                   ;REGISTER INDEX
                                                SC8                     ;PARITY EXIT ADDRESS
                                        7$:     MOV     (SP),R5         ;PICKUP (RMAS) BEFORE THE ERROR CALL
                                                JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
                                                EMT     2               ;REPORT THE UNEXPECTED ATTENTION
                                                BR      SC4             ;GO CHECK FOR MORE ATA'S
                                        8$:
                                                JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
                                                EMT     5               ;REPORT THE ADDRESS PLUG CHANGE
                                                BR      SC4             ;CHECK FOR MORE DRIVES
                                        SC6:    ASL     R1              ;SETUP TO ADDRESS WORDS
                                                MOV     #-1,TIMER(R1)   ;STOP THE TIMER
                                                ASR     R1              ;RESTORE THE DRIVE ADDRESS
                                                JSR     PC,GETREQ       ;GET THE DPB POINTER FROM THE QUEUE
                                                MOV     R1,RMCS2(R4)    ;SELECT DRIVE
                                                JMP     SC11            ;PROCESS THE SEARCH
                                                JSR     R0,RD.RM        ;READ THE RM'S STATUS REG.
                                                RMDS
                                                SC8
                                                MOV     (SP),R5         ;AND PUT IT IN R5
                                                ROL     (SP)+           ;WAS THERE AN ERROR?
                                                BMI     1$              ;BR IF ERROR
                                                TSTB    DRVACT(R1)      ;CHECK DRIVE'S STATE
                                                BGT     SC11            ;BR IF DRIVE ACTIVE WITH ORDER
                                                BIS     #BIT15!BIT07!BIT03,16(R2)       ;INFORM USER OF ERROR RECOVER COMPLETION
                                                BR      SC7
                                        1$:     JSR     R0,RD.RM        ;READ ERROR REGISTER #1
                                                RMER1
                                                SC8
                                                MOV     (SP)+,R5        ;AND SAVE IT IN R5
                                                JSR     PC,SVRH70       ;SAVE RH/RM REGISTERS
                                                MOV     #111,-(SP)      ;ISSUE A DRIVE CLEAR
                                                JSR     R0,WRT.RM
                                                RMCS1
                                                SC8
```

```
                                SC6A:   ROL     R5              ;WAS "UNSAFE" CONDITION =1?
                                        BMI     1$              ;BR IF YES
                                        TST     R2              ;ANYTHING IN QUEUE ?
                                        BEQ     SC7             ;BR IF NOT
                                        BIS     #BIT15!BIT07!BIT05,16(R2)       ;INFORM USER OF ERROR
                                        BR      SC7
                                1$:     JSR     R0,RD.RM        ;READ DRIVE STATUS REG. #1
                                        RMDS
                                        SC8
                                        MOV     (SP),R5         ;SAVE RMDS IN R5
                                        ROL     (SP)+           ;"ERR"=1?
                                        BPL     2$              ;BR IF NO--UNSAFE CLEARED
                                        MOVB    #-1,DRVSTA(R1)  ;DRIVE IS UNSAFE
                                        JSR     PC,SVRH70       ;SAVE RH/RM REGISTERS
                                        BIS     #BIT15!BIT12,16(R2)     ;INFORM USER OF UNSAFE ERROR
                                        BR      SC7
                                2$:     BIT     #BIT12,R5       ;"MOL" = 1 ?
                                        BNE     3$              ;BR IF YES
                                        MOVB    #-1,DRVACT(R1)  ;ACTIVE ERROR RECOVER
                                        MOVB    #1,DRVSTA(R1)   ;ONLINE
                                        ASL     R1
                                        MOV     #30000.,TIMER(R1)       ;START 30 SECOND TIMER
                                        ASR     R1
                                        JMP     SC4
                                3$:     BIS     #BIT15!BIT07!BIT04,16(R2)       ;INFORM USER OF ERROR
                                SC7:    CLRB    DRVACT(R1)      ;DRIVE IS IDLE
                                ;       JSR     PC,EMPTYQ       ;DUMP THE QUEUE
                                        JSR     PC,POPQUE       ;REMOVE THE QUEUE
                                        TSTB    ULDFLG(R1)      ;UNLOAD IN RMOGRESS OR QUEUE?
                                        BGT     1$              ;BR IF NOT
                                        CLRB    ULDFLG(R1)      ;CLEAR UNLOAD FLAG
                                1$:     MOVB    ATABIT(R1),RMAS(R4)     ;CLEAR ATTENTION BIT
                                        TSTB    DRVSTA(R1)      ;IS THE DRIVE UNSAFE ?
                                        BMI     2$              ;BR IF IT IS
                                ;       MOV     #113,-(SP)      ;RELEASE COMMAND
                                        MOV     #111,-(SP)      ;DRIVE CLEAR COMMAND
                                        JSR     R0,WRT.RM       ;WRITE THE COMMAND INTO RPCS1
                                        RMCS1                   ;REGISTER INDEX
                                        SC8                     ;PARITY EXIT ADDRESS
                                2$:     JMP     SC4             ;CHECK FOR MORE DRIVES
                                SC8:    TSTB    DRVACT(R1)      ;IS DRIVE IDLE?
                                        BEQ     1$              ;YES
                                        JSR     PC,GETREQ       ;GET DPB POINTER
                                        JSR     PC,CI7          ;PROCESS THE PARITY ERROR
                                        BR      2$              ;CONTINUE
                                1$:
                                ;       JSR     PC,CI7          ;PROCESS THE PARITY ERROR
                                        JSR     PC,CI7B         ;PROCESS THE UNCORRECTABLE PARITY ERROR
                                2$:     JMP     SC4             ;CHECK MORE DRIVES
                                SC11:   TSTB    ULDFLG(R1)      ;"UNLOAD IN PROGRESS"?
                                        BLE     1$              ;BR IF NO
                                        CLRB    ULDFLG(R1)      ;CLEAR UNLOAD FLAG
                                1$:     CLRB    DRVACT(R1)      ;SET DRIVE IDLE
                                        BITB    ATABIT(R1),SRCHWT       ;DOING A SEARCH OPERATION FOR
                                                                ;AN I/O COMMAND?
                                        BNE     2$              ;BR IF YES
                                        JSR     PC,POPQUE       ;REMOVE REQUEST FROM QUEUE
```

```
 954 034006  052762  000200  000016          BIS     #BIT07,16(R2)   ;SET "DONE" BIT
 955 034014  005737  027774                  TST     SAVEFG          ;SAVE THE REGISTERS?
 956 034020  100002                          BPL     2$              ;BR IF NO
 957 034022  004737  035256                  JSR     PC,SVRH70       ;YES--SAVE ALL OF THE RH/RM REG'S
 958 034026  116164  030022  000016  2$:     MOVB    ATABIT(R1),RMAS(R4)     ;CLEAR ATTENTION BIT
 959 034034  146137  030022  027750          BICB    ATABIT(R1),SRCHWT       ;CLEAR IMPLIED SEEK SET
 960 034042  006301                          ASL     R1              ;WORD INDEX
 961 034044  012761  177777  030000          MOV     #-1,TIMER(R1)   ;STOP CLOCK
 962 034052  006201                          ASR     R1              ;RESTORE R1
 963 034054  004737  031114                  JSR     PC,OPT          ;START A REQUEST
 964 034060  000137  033176                  JMP     SC4             ;CHECK FOR MORE DRIVES
 965 034064  010164  000010          SC12:   MOV     R1,RMCS2(R4)    ;SELECT DRIVE
 966 034070  016437  000012  027666          MOV     RMDS(R4),RMERRS ;SAVE THE FOUR REGISTERS THAT
 967 034076  016437  000014  027670          MOV     RMER1(R4),RMERRS+2      ;WILL TELL US SOMETHING
 968 034104  016437  000042  027672          MOV     RMER2(R4),RMERRS+4
 969 034112  016437  000040  027674          MOV     RMMR2(R4),RMERRS+6
 970 034120  004037  030264                  JSR     R0,DRVINT       ;INIT. THE STATE OF THE DRIVE
 971 034124  000401                          BR      1$              ;TAKE ERROR EXIT
 972 034126  000207                          RTS     PC              ;RETURN
 973 034130  005726                  1$:     TST     (SP)+           ;POP PC OFF OF THE STACK
 974 034132  000674                          BR      SC8             ;PROCESS THE PARITY ERROR
 975 034134  006301                  SC13:   ASL     R1              ;SETUP TO ADDRESS WORDS
 976 034136  012761  177777  030000          MOV     #-1,TIMER(R1)   ;STOP THE TIMER
 977 034144  006201                          ASR     R1              ;
 978 034146  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT THE DRIVE
 979 034152  116164  030022  000016          MOVB    ATABIT(R1),RMAS(R4)     ;CLEAR THE ATTENTION BIT
 980 034160  105761  027726          1$:     TSTB    DPINT(R1)       ;INITIALIZING THE DRIVE ?
 981 034164  001424                          BEQ     2$              ;BR IF NOT
 982 034166  105061  027726                  CLRB    DPINT(R1)       ;CLEAR THE INIT INDICATOR
 983 034172  004037  030264                  JSR     R0,DRVINT       ;GO INIT THE DRIVE
 984 034176  000240                          NOP                     ;DUMMY PARITY ERROR RETURN
 985 034200  105761  027706                  TSTB    DRVSTA(R1)      ;DRIVE ONLINE ?
 986 034204  003014                          BGT     2$              ;BR IF YES -- START ORDER
 987 034206  005702                          TST     R2              ;QUEUE ENTRY FOR THE DRIVE
 988 034210  001426                          BEQ     3$              ;BR IF NOT
 989 034212  004737  036112                  JSR     PC,GETREQ       ;GET DPB ADDRESS
 990 034216  052762  140000  000016          BIS     #BIT15!BIT14,16(R2)     ;INFORM USER THAT DRIVE OFFLINE
 991 034224  004737  035256                  JSR     PC,SVRH70       ;SAVE THE REGISTERS
 992                             ;           JSR     PC,EMPTYQ       ;EMPTY THE REQUEST QUEUE
 993 034230  004737  036134                  JSR     PC,POPQUE       ;REMOVE THE QUEUE
 994 034234  000414                          BR      3$
 995 034236  032764  004000  000000  2$:     BIT     #BIT11,RMCS1(R4)        ;DVA SET ?
 996 034244  001006                          BNE     4$              ;SET THEN CALL OPT
 997 034246  006301                          ASL     R1
 998 034250  012761  060000  030000          MOV     #60000,TIMER(R1)
 999 034256  006201                          ASR     R1
1000 034260  000402                          BR      3$
1001 034262  004737  031114          4$:     JSR     PC,OPT          ;START THE PENDING REQUEST
1002 034266  000137  033176          3$:     JMP     SC4             ;PROCESS OTHER DRIVES
1003
1004                                 ;RM TIMER ROUTINE
1005                                 ;CALL
1006                                 ;       MOV     #TIME,-(SP)     ;ELASPED TIME IN MILLISECONDS ON THE STACK
1007                                 ;       JSR     PC,RMTMR        ;CALL RM05 TIME ROUTINE
1008
1009 034272  005737  027752          RMTMR:  TST     ACTDRV          ;CHECK "ACTDRV & ACTSTR"
1010 034276  001027                          BNE     4$              ;IF NON ZERO EXIT
```

```
1011 034300  112737  000001  027753          MOVB    #1,ACTSTR       ;SET "ACTSTR"
1012 034306  104412                          SAVREG                  ;SAVE R0 - R5
1013 034310  005001                          CLR     R1              ;START WITH DRIVE 0
1014 034312  005003                          CLR     R3
1015 034314  005763  030000          1$:     TST     TIMER(R3)       ;IS THE TIMER RUNNING?
1016 034320  002406                          BLT     2$              ;BR IF NO
1017 034322  166663  000002  030000          SUB     2(SP),TIMER(R3) ;COUNT THE INTERVAL
1018 034330  003002                          BGT     2$              ;BR IF NO SOFTWARE TIMEOUT
1019 034332  004737  034362                  JSR     PC,STO          ;CALL SOFTWARE TIMEOUT ROUTINE
1020 034336  005201                  2$:     INC     R1              ;MOVE TO NEXT DRIVE
1021 034340  005723                          TST     (R3)+
1022 034342  022701  000010                  CMP     #8.,R1          ;OUT OF DRIVES?
1023 034346  003362                          BGT     1$              ;BR IF NO
1024 034350  104413                  3$:     RESREG                  ;RESTORE R0 - R5
1025 034352  105037  027753                  CLRB    ACTSTR          ;ZERO ACTIVE SOFTWARE TIMEOUT ROUTINE FLAG
1026 034356  012616                  4$:     MOV     (SP)+,(SP)      ;ADJUST THE STACK
1027 034360  000207                          RTS     PC              ;RETURN
1028
1029                                 ;SOFTWARE TIMEOUT ROUTINE
1030                                 ;
1031                                 ;NOTE: THIS ROUTINE MUST BE ENTERED AT PRIORITY 6
1032                                 ;       OR GREATER
1033                                 ;
1034                                 ;CALL:  STO
1035                                 ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
1036                                 ;       JSR     PC,STO          ;CALL
1037                                 ;       RETURN
1038
1039 034362  010146                  STO:    MOV     R1,-(SP)        ;SAVE R1
1040 034364  010246                          MOV     R2,-(SP)        ;SAVE R2
1041 034366  010346                          MOV     R3,-(SP)        ;SAVE R3
1042 034370  010446                          MOV     R4,-(SP)        ;SAVE R4
1043 034372  013704  030034                  MOV     RMADR,R4        ;GET ADDRESS OF "RMCS1"
1044 034376  010164  000010                  MOV     R1,RMCS2(R4)    ;SELECT THE DRIVE
1045 034402  004037  034704                  JSR     R0,RD.RM        ;READ "DRIVE STATUS REG"
1046 034406  000012                          RMDS
1047                                 ;       STO5
1048 034410  034672                          STO9
1049 034412  105726                          TSTB    (SP)+           ;IS "DRY"=1?
1050 034414  100436                          BMI     STO2            ;BR IF YES
1051 034416  105761  027726          STO1:   TSTB    DPINT(R1)       ;TRYING TO INTIALIZE THE DRIVE ?
1052 034422  001033                          BNE     STO2            ;BR IF YES
1053 034424  105761  027736                  TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
1054 034430  001030                          BNE     STO2            ;BR IF YES
1055 034432  013702  027746                  MOV     TRNSWT,R2       ;PICKUP TRANSFER WAIT QUEUE
1056 034436  020137  030020                  CMP     R1,DTUW         ;TRANSFER UNDERWAY ON THIS DRIVE?
1057 034442  001404                          BEQ     1$              ;BR IF YES
1058 034444  000137  034672                  JMP     STO9            ;IF NOT DON'T BOTHER DRIVES
1059 034450  004737  036112                  JSR     PC,GETREQ       ;GET DPB ADDRESS
1060 034454  052762  101000  000016  1$:     BIS     #BIT15!BIT09,16(R2)     ;SET THE ERROR FLAGS
1061 034462  004737  035256                  JSR     PC,SVRH70       ;SAVE RH/RM REGISTERS
1062                                 ;       MOV     #CLR,RMCS2(R4)  ;"INIT" THE MASS BUS
1063 034466  105061  027676                  CLRB    DRVACT(R1)      ;DRIVE IS IDLE
1064 034472  105061  027754                  CLRB    ULDFLG(R1)      ;CLEAR THE UNLOAD FLAG
1065 034476  005037  027746                  CLR     TRNSWT          ;CLEAR DPB ADDRESS
1066 034502  012737  177777  030020          MOV     #-1,DTUW        ;CLEAR THE TRANSFER DRIVE #
1067 034510  000470                          BR      STO9            ;DON'T BOTHER OTHER DRIVES
```

```
1068 034512  116405  000016          ST02:   MOVB    RMAS(R4),R5     ;READ ATTENTION REG
1069 034516  136105  030022                  BITB    ATABIT(R1),R5   ;IS ATTENTION FOR THIS DRIVE UP ?
1070 034522  001007                          BNE     ST03            ;YES
1071 034524  105761  027726                  TSTB    DPINT(R1)       ;TRYING TO INTIALIZE THE DRIVE ?
1072 034530  001021                          BNE     ST06            ;BR IF YES - DRIVE NOT ONLINE
1073 034532  105761  027736                  TSTB    DPRQS(R1)       ;OUTSTANDING PORT REQUEST FOR THE DRIVE ?
1074 034536  001035                          BNE     ST07            ;BR IF YES - NO RESPONSE TO REQUEST
1075 034540  000454                          BR      ST09            ;OTHER WISE EXIT
1076 034542  105761  027726          ST03:   TSTB    DPINT(R1)       ;INITIALIZING THE DRIVE ?
1077 034546  001003                          BNE     1$              ;BR IF INIT PENDING
1078 034550  105761  027736                  TSTB    DPRQS(R1)       ;PORT REQUEST PENDING ?
1079 034554  001446                          BEQ     ST09            ;BR IF NOT
1080 034556  012763  177777  030000  1$:     MOV     #-1,TIMER(R3)   ;STOP THE TIMER
1081 034564  000442                          BR      ST09            ;EXIT
1082 034566  004737  032274          ST05:   JSR     PC,CI8          ;GO HANDLE THE PARITY ERROR
1083 034572  000437                          BR      ST09
1084 034574  105061  027726          ST06:   CLRB    DPINT(R1)       ;CLEAR THE INITIALIZE INDICATOR
1085 034600  105061  027706                  CLRB    DRVSTA(R1)      ;SET UNIT OFFLINE
1086 034604  012763  177777  030000          MOV     #-1,TIMER(R3)   ;STOP THE TIMER
1087 034612  004737  036112                  JSR     PC,GETREQ       ;GET THE DPB ADDRESS
1088 034616  005702                          TST     R2              ;REQUEST IN QUEUE ?
1089 034620  001424                          BEQ     ST09            ;BR IF NOT
1090 034622  052762  140000  000016          BIS     #BIT15!BIT14,16(R2)     ;INFORM THE USER DRIVE NOT AVAILABLE
1091 034630  000414                          BR      ST08            ;FINISH
1092 034632  012763  177777  030000  ST07:   MOV     #-1,TIMER(R3)   ;STOP THE TIMER
1093 034640  105061  027736                  CLRB    DPRQS(R1)       ;CLEAR PORT REQUEST INDICATOR
1094 034644  004737  036112                  JSR     PC,GETREQ       ;GET DPB ADDRESS
1095 034650  005702                          TST     R2              ;QUEUE ENTRY FOR DRIVE ?
1096 034652  001407                          BEQ     ST09            ;BR IF NONE
1097 034654  012762  100004  000016          MOV     #BIT15!BIT2,16(R2)      ;INFORM USER OF PORT REQUEST ERROR
1098 034662  004737  036016          ST08:   JSR     PC,EMPTYQ       ;CLEAR THE QUEUE FOR THE DRIVE
1099 034666  004737  035256                  JSR     PC,SVRH70       ;SAVE THE REGISTERS
1100 034672  012604                  ST09:   MOV     (SP)+,R4        ;RESTORE R4
1101 034674  012603                          MOV     (SP)+,R3        ;RESTORE R3
1102 034676  012602                          MOV     (SP)+,R2        ;RESTORE R2
1103 034700  012601                          MOV     (SP)+,R1        ;RESTORE R1
1104 034702  000207                          RTS     PC              ;RETURN
1105
1106                                 ;ROUTINE TO READ A RH/RM REGISTER
1107                                 ;
1108                                 ;CALL
1109                                 ;       JSR     R0,RD.RM        ;GO READ A REGISTER
1110                                 ;       INDEX                   ;REG. INDEX FROM BASE
1111                                 ;       ERRADR                  ;ERROR ADDRESS--PROCESS ERROR STARTING
1112                                 ;                               ;AT THIS ADDRESS
1113                                 ;       RETURN                  ;CONTENTS OF REG. IS ON THE STACK
1114                                 ;
1115 034704  013737  030032  035052  RD.RM:   MOV     MCPEMX,RD.RM2   ;MAX. RETRYS ALLOWED
1116 034712  011646                          MOV     (SP),-(SP)      ;SAVE R0 FOR RETURN
1117 034714  013737  030034  034730          MOV     RMADR,RD.ADR    ;FORM THE DESIRED ADDRESS
1118 034722  062037  034730                  ADD     (R0)+,RD.ADR    ;USING THE BASE AND THE INDEX
1119 034726  013727                  RD.RM1: MOV     @(PC)+,(PC)+    ;READ THE DESIRED REGISTER OF THE RM DRIVE
1120 034730  000000                  RD.ADR: .WORD   0               ;ADDRESS IS FORMED HERE
1121 034732  000000                  RD.WRD: .WORD   0               ;REG. CONTENTS PUT HERE
1122 034734  013766  034732  000002          MOV     RD.WRD,2(SP)    ;RETURN IT TO THE USER
1123 034742  013746  030034                  MOV     RMADR,-(SP)     ;PUT THE ADDRESS ON THE STACK
1124 034746  062716  000010                  ADD     #RMCS2,(SP)     ;FORM THE ADDRESS OF RMCS2
```

```
1125 034752 032736 010000                  BIT     #BIT12,@(SP)+   ;CHECK THE 'NED' BIT
1126 034756 001037                         BNE     RD.RM3          ;BR IF DRIVE NON-EXISTENT
1127 034760 017746 173050                  MOV     @RMADR,-(SP)    ;READ RMCS1
1128 034764 032716 020000                  BIT     #BIT13,(SP)     ;DID MCPE SET?
1129 034770 001002                         BNE     1$              ;BR IF YES
1130 034772 022620                         CMP     (SP)+,(R0)+     ;ADJUST FOR RETURN
1131 034774 000432                         BR      RD.RM4          ;EXIT
1132 034776                         1$:
     034776 004037 036202                  JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
     035002 104003                         EMT     3               ;REPORT "MCPE" ERROR
1133 035004 005737 030020                  TST     DTUW            ;DATA TRANSFER UNDERWAY?
1134 035010 100405                         BMI     2$              ;NO
1135 035012 032716 040000                  BIT     #BIT14,(SP)     ;"TRE" = 1 ?
1136 035016 001402                         BEQ     2$              ;NO
1137 035020 005726                         TST     (SP)+           ;YES--CLEAN OFF THE STACK AND
1138 035022 000415                         BR      RD.RM3          ;TAKE THE FATAL ERROR EXIT
1139 035024 052716 040000           2$:    BIS     #BIT14,(SP)     ;CLEAR "MCPE" BY SENDING A "1" TO "TRE"
1140 035030 000316                         SWAB    (SP)            ;POSITION BEFORE WRITING
1141 035032 013737 030034 035046           MOV     RMADR,3$        ;FORM ADDRESS OF HIGH BYTE
1142 035040 005237 035046                  INC     3$
1143 035044 112637                         MOVB    (SP)+,@(PC)+    ;WRITE THE HIGH BYTE OF RMCS1
1144 035046 000000                  3$:    .WORD   0               ;ADDRESS STORAGE
1145 035050 005327                         DEC     (PC)+           ;EXCEEDED MAX. RETRYS
1146 035052 000003                  RD.RM2: .WORD  3
1147 035054 002324                         BGE     RD.RM1          ;BR IF NO
1148 035056 011000                  RD.RM3: MOV    (R0),R0         ;FATAL ERROR EXIT
1149 035060 012616                         MOV     (SP)+,(SP)
1150 035062 000200                  RD.RM4: RTS    R0
1151
1152                                       ;ROUTINE TO WRITE A REGISTER
1153                                       ;
1154                                       ;CALL
1155                                       ;       MOV     DATA,-(SP)      ;DATA TO BE LOADED ON THE STACK
1156                                       ;       JSR     R0,WRT.RM       ;CALL THE ROUTINE TO LOAD(WRITE) THE REG.
1157                                       ;       INDEX                   ;INDEX OF THE REGISTER TO BE LOADED
1158                                       ;       ERRADR                  ;ADDRESS TO RETURN TO ON AN ERROR
1159                                       ;       RETURN                  ;ERROR FREE RETURN
1160
1161 035064 013737 030032 035242    WRT.RM: MOV    MCPEMX,WRT.R2   ;MAX RETRYS ALLOWED
1162 035072 016637 000002 035152           MOV     2(SP),WRT.WD    ;SAVE THE WORD TO WRITE
1163 035100 012616                         MOV     (SP)+,(SP)      ;ADJUST THE STACK
1164 035102 012037 035154                  MOV     (R0)+,WRT.AD    ;GET INDEX OF REGISTER TO BE WRITTEN
1165 035106 001015                         BNE     1$              ;BR IF NOT RMCS1
1166 035110 122737 000150 035152           CMPB    #150,WRT.WD     ;IS THE COMMAND FOR DATA TRANSFERS?
1167 035116 002411                         BLT     1$              ;YES--DON'T GET THE OLD A16 & A17, & PSEL
1168 035120 004037 034704                  JSR     R0,RD.RM        ;NO---COMBINE A16&A17, & PSEL WITH
1169 035124 000000                         RMCS1                   ;THE COMMAND BEFORE SENDING IT TO
1170 035126 035246                         WRT.R3                  ;THE RH/RM
1171 035130 000316                         SWAB    (SP)
1172 035132 042716 177770                  BIC     #^C7,(SP)
1173 035136 112637 035153                  MOVB    (SP)+,WRT.WD+1
1174 035142 063737 030034 035154    1$:    ADD     RMADR,WRT.AD    ;FORM THE ADDRESS OF THE DISK REG.
1175 035150 012737                  WRT.R1: MOV    (PC)+,@(PC)+    ;LOAD THE DESIRED REG.
1176 035152 000000                  WRT.WD: .WORD  0               ;WORD TO WRITE GOES HERE
1177 035154 000000                  WRT.AD: .WORD  0               ;ADDRESS IS FORMED HERE
1178 035156 013746 030034                  MOV     RMADR,-(SP)     ;PUT THE ADDRESS ON THE STACK
1179 035162 062716 000010                  ADD     #RMCS2,(SP)     ;FORM THE ADDRESS OF RMCS2
```

```
1180 035166  032736  010000                    BIT     #BIT12,@(SP)+   ;CHECK THE 'NED' BIT
1181 035172  001025                            BNE     WRT.R3          ;BR IF DRIVE NON-EXISTENT
1182 035174  004037  034704                    JSR     R0,RD.RM        ;CHECK FOR PARITY ERROR ON WRITE
1183 035200  000014                            RMER1
1184 035202  035246                            WRT.R3
1185 035204  032726  000010                    BIT     #BIT03,(SP)+
1186 035210  001420                            BEQ     WRT.R4          ;BR IF ''PAR=0''
1187 035212  016037  177776  035224            MOV     -2(R0),1$       ;PICKUP THE INDEX
1188 035220  004037  034704                    JSR     R0,RD.RM        ;READ THE REG.
1189 035224  000000                  1$:       .WORD   0               ;REG. INDEX
1190 035226  035246                            WRT.R3                  ;RETURN TO THIS ADDRESS ON ERROR
1191 035230  004037  036202                    JSR     R0,ES.SAV       ;SAVE THE ADDRESS IN '$ESCAPE'
     035234  104004                            EMT     4               ;REPORT THE PARITY ON WRITE ERROR
1192 035236  005726                            TST     (SP)+           ;CLEAR OFF THE STACK
1193 035240  005327                            DEC     (PC)+           ;DECREMENT THE ERROR COUNT
1194 035242  000003                  WRT.R2:   .WORD   3               ;RETRY COUNTER
1195 035244  002341                            BGE     WRT.R1          ;TRY AGAIN IF NOT FINISHED
1196 035246  011000                  WRT.R3:   MOV     (R0),R0         ;TAKE THE ''PARITY ON WRITE'' ERROR EXIT
1197 035250  000401                            BR      WRT.R5          ;EXIT
1198 035252  005720                  WRT.R4:   TST     (R0)+           ;ADJUST FOR ERROR FREE EXIT
1199 035254  000200                  WRT.R5:   RTS     R0
1200
1201                                           ;ROUTINE TO SAVE THE RH/RM REGISTERS AS PER DPB+14
1202                                           ;
1203                                           ;CALL
1204                                           ;       MOV     #DPBNUM,R2      ;DPB POINTER TO R2
1205                                           ;       JSR     PC,SVRH70       ;SAVE THE DRIVES REG'S
1206
1207 035256  104412                  SVRH70:   SAVREG                  ;SAVE R0 - R5
1208 035260  005702                            TST     R2              ;QUEUE ENTRY FOR THE DRIVE ?
1209 035262  001442                            BEQ     6$              ;BR IF NONE
1210 035264  013704  030034                    MOV     RMADR,R4
1211 035270  111264  000010                    MOVB    (R2),RMCS2(R4)  ;SELECT DRIVE
1212 035274  016203  000014                    MOV     14(R2),R3       ;GET THE ERROR TABLE POINTER
1213 035300  001433                            BEQ     6$              ;EXIT IF NO ADDRESS
1214 035302  005037  035336                    CLR     3$              ;COUNTER & POINTER
1215 035306  023727  035336  000022  1$:       CMP     3$,#RMDB        ;REACHED THE BUFFER REGISTER ?
1216 035314  001006                            BNE     2$              ;BR IF NOT
1217 035316  032764  000200  000010            BIT     #BIT07,RMCS2(R4)        ;'OR' SET ?
1218 035324  001002                            BNE     2$              ;BR IF SET
1219 035326  005023                            CLR     (R3)+           ;STORE RMDB AS ZEROES
1220 035330  000405                            BR      4$              ;CONTINUE
1221 035332  004037  034704            2$:     JSR     R0,RD.RM        ;READ THE SELECTED REGISTER
1222 035336  000000                    3$:     .WORD   0               ;REGISTER INDEX
1223 035340  035364                            5$                      ;ERROR RETURN ADDRESS
1224 035342  012623                            MOV     (SP)+,(R3)+     ;STORE THE REGISTER CONTENTS
1225 035344  023727  035336  000046  4$:       CMP     3$,#RMEC2       ;REACHED THE END ?
1226 035352  001406                            BEQ     6$              ;BR IF YES
1227 035354  062737  000002  035336            ADD     #2,3$           ;INCREMENT THE REGISTER INDEX
1228 035362  000751                            BR      1$              ;CONTINUE READING THE REGISTERS
1229 035364  004737  032174            5$:     JSR     PC,CI7          ;PROCESS THE UNCORRECTABLE PARITY ERROR
1230 035370  104413                    6$:     RESREG                  ;RESTORE R0 - R5
1231 035372  000207                            RTS     PC              ;RETURN
1232
1233                                           ;ROUTINE TO SET THE INTERRUPT WITHOUT GETTING A ''TRE''
1234                                           ;CALL
1235                                           ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER TO R1
```

```
                                        ;       JSR     PC,SET.IE       ;SET "IE"
                                        ;       RETURN
                                        SET.IE: MOV     R4,-(SP)        ;SAVE R4
035374  010446                                 MOV     RMADR,R4        ;PICKUP ADDRESS OF RMCS1
035376  013704  030034                         MOV     R1,RMCS2(R4)    ;SELECT DRIVE
035402  010164  000010                         MOV     (R4),-(SP)      ;READ RMCS1
035406  011446                                 BIS     #BIT14,(SP)     ;SET THE "TRE" BIT OF THE WORD READ
035410  052716  040000                         SWAB    (SP)            ;ADJUST FOR DATO
035414  000316                                 MOVB    #BIT06,(R4)     ;SET "IE"
035416  112714  000100                         BIT     #BIT12,RMCS2(R4)        ;IS "NED"=1?
035422  032764  010000  000010                 BNE     1$              ;YES--CLEAR "TRE"
035430  001002                                 TST     (SP)+           ;CLEAN OFF THE STACK
035432  005726                                 BR      2$
035434  000402                                 MOVB    (SP)+,1(R4)     ;CLEAR "TRE"
035436  112664  000001                 1$:     MOV     (SP)+,R4        ;RESTORE R4
035442  012604                         2$:     RTS     PC              ;RETURN TO CALLER
035444  000207
                                        ;QUEUE COUNT
035446      000                         QCNT:   .BYTE   0               ;DRIVE 0
035447      000                                 .BYTE   0               ;DRIVE 1
035450      000                                 .BYTE   0               ;DRIVE 2
035451      000                                 .BYTE   0               ;DRIVE 3
035452      000                                 .BYTE   0               ;DRIVE 4
035453      000                                 .BYTE   0               ;DRIVE 5
035454      000                                 .BYTE   0               ;DRIVE 6
035455      000                                 .BYTE   0               ;DRIVE 7

                                        ;QUEUE INPUT POINTERS

035456  035540                          QINPT:  .WORD   QDRV0           ;DRIVE 0
035460  035560                                  .WORD   QDRV1           ;DRIVE 1
035462  035600                                  .WORD   QDRV2           ;DRIVE 2
035464  035620                                  .WORD   QDRV3           ;DRIVE 3
035466  035640                                  .WORD   QDRV4           ;DRIVE 4
035470  035660                                  .WORD   QDRV5           ;DRIVE 5
035472  035700                                  .WORD   QDRV6           ;DRIVE 6
035474  035720                                  .WORD   QDRV7           ;DRIVE 7

                                        ;QUEUE OUTPUT POINTERS

035476  035540                          QOUTPT: .WORD   QDRV0           ;DRIVE 0
035500  035560                                  .WORD   QDRV1           ;DRIVE 1
035502  035600                                  .WORD   QDRV2           ;DRIVE 2
035504  035620                                  .WORD   QDRV3           ;DRIVE 3
035506  035640                                  .WORD   QDRV4           ;DRIVE 4
035510  035660                                  .WORD   QDRV5           ;DRIVE 5
035512  035700                                  .WORD   QDRV6           ;DRIVE 6
035514  035720                                  .WORD   QDRV7           ;DRIVE 7

035516  035540                          QSTART: .WORD   QDRV0           ;DRIVE 0 START ADDRESS
035520  035560                          QSTOP:  .WORD   QDRV1           ;DRIVE 0 STOP ADDRESS & DRIVE 1 START ADDRESS
035522  035600                                  .WORD   QDRV2           ;STOP DRIVE 1--START DRIVE 2
035524  035620                                  .WORD   QDRV3           ;STOP DRIVE 2--START DRIVE 3
035526  035640                                  .WORD   QDRV4           ;STOP DRIVE 3--START DRIVE 4
035530  035660                                  .WORD   QDRV5           ;STOP DRIVE 4--START DRIVE 5
035532  035700                                  .WORD   QDRV6           ;STOP DRIVE 5--START DRIVE 6
```

```
1293 035534  035720                        .WORD   QDRV7           ;STOP DRIVE 6--START DRIVE 7
1294 035536  035740                        .WORD   QTERM           ;STOP DRIVE 7
1295
1296                                ;DRIVE REQUEST QUEUES
1297
1298 035540                         QDRV0:  .BLKW   10
1299 035560                         QDRV1:  .BLKW   10
1300 035600                         QDRV2:  .BLKW   10
1301 035620                         QDRV3:  .BLKW   10
1302 035640                         QDRV4:  .BLKW   10
1303 035660                         QDRV5:  .BLKW   10
1304 035700                         QDRV6:  .BLKW   10
1305 035720                         QDRV7:  .BLKW   10
1306         035740                 QTERM=.
1307
1308                                ;ROUTINE TO CLEAR ALL OF THE REQUEST QUEUES
1309                                ;
1310                                ;CALL
1311                                ;       JSR     PC,CLRQUE
1312
1313 035740  104412                 CLRQUE: SAVREG                  ;SAVE R0 - R5
1314 035742  012702  035446                 MOV     #QCNT,R2        ;ZERO THE QUEUE COUNTS
1315 035746  005022                         CLR     (R2)+           ;DRIVES 0 & 1
1316 035750  005022                         CLR     (R2)+           ;DRIVES 2 & 3
1317 035752  005022                         CLR     (R2)+           ;DRIVES 4 & 5
1318 035754  005022                         CLR     (R2)+           ;DRIVES 6 & 7
1319 035756  012703  000010                 MOV     #8.,R3          ;MOVE THE STARTING
1320 035762  012701  035516                 MOV     #QSTART,R1      ;ADDRESS OF THE QUEUE INTO
1321 035766  012122                 1$:     MOV     (R1)+,(R2)+     ;THE QUEUE INPUT POINTER
1322 035770  005303                         DEC     R3
1323 035772  001375                         BNE     1$
1324 035774  012703  000010                 MOV     #8.,R3          ;MOVE THE STARTING ADDRESS
1325 036000  012701  035516                 MOV     #QSTART,R1      ;OF THE QUEUE INTO THE
1326 036004  012122                 2$:     MOV     (R1)+,(R2)+     ;QUEUE OUTPUT POINTER
1327 036006  005303                         DEC     R3
1328 036010  001375                         BNE     2$
1329 036012  104413                         RESREG                  ;RESTORE R0 - R5
1330 036014  000207                         RTS     PC
1331
1332                                ;EMPTY THE QUEUE SPECIFIED BY R1
1333                                ;
1334                                ;CALL
1335                                ;       MOV     DRVNUM,R1       ;DRIVE NUMBER TO R1
1336                                ;       JSR     PC,EMPTYQ
1337
1338 036016  105061  035446         EMPTYQ: CLRB    QCNT(R1)        ;CLEAR NUMBER OF ITEMS IN QUEUE
1339 036022  006301                         ASL     R1
1340 036024  016161  035456  035476         MOV     QINPT(R1),QOUTPT(R1)    ;SET OUTPUT QUEUE POINTER=INPUT POINTER
1341 036032  006201                         ASR     R1
1342 036034  000207                         RTS     PC
1343
1344                                ;ROUTINE TO PUT A REQUEST IN QUEUE
1345                                ;
1346                                ;CALL
1347                                ;       MOV     #DRVNUM,R1      ;DRIVE NUMBER
1348                                ;       MOV     #DPB,R2         ;ADDRESS OF PARAMETER BLOCK
1349                                ;       JSR     R0,DRVQUE       ;GO PUT REQUEST IN QUEUE
```

```
1350                                    ;       RETURN1                 ;RETURN HERE IF QUEUE IS FULL
1351                                    ;       RETURN2                 ;RETURN HERE IF REQUEST IS IN QUEUE
1352
1353 036036  122761  000010  035446  DRVQUE: CMPB    #10,QCNT(R1)        ;IS QUEUE FULL?
1354 036044  001421                          BEQ     2$                  ;BR IF YES-TAKE RETURN1
1355 036046  105261  035446                  INCB    QCNT(R1)            ;INCREMENT QUEUE COUNT
1356 036052  006301                          ASL     R1
1357 036054  010271  035456                  MOV     R2,@QINPT(R1)       ;PUT THIS REQUEST IN QUEUE
1358 036060  062761  000002  035456          ADD     #2,QINPT(R1)        ;UPDATE THE QUEUE POINTER
1359 036066  026161  035456  035520          CMP     QINPT(R1),QSTOP(R1)     ;TIME TO RESET THE POINTER
1360 036074  001003                          BNE     1$                  ;BR IF NO
1361 036076  016161  035516  035456          MOV     QSTART(R1),QINPT(R1)    ;YES--RESET POINTER
1362 036104  006201                  1$:     ASR     R1
1363 036106  005720                          TST     (R0)+               ;TAKE RETURN 2
1364 036110  000200                  2$:     RTS     R0                  ;RETURN TO USER
1365
1366                                    ;ROUTINE TO GET THE "DPB" ADDRESS OF NEXT REQUEST IN QUEUE
1367                                    ;
1368                                    ;CALL
1369                                    ;       MOV     #DRVNUM,R1          ;DRIVE NUMBER TO R1
1370                                    ;       JSR     PC,GETREQ           ;GO GET THE REQUEST
1371                                    ;       RETURN                      ;R2="DPB" ADDRESS OF THE REQUEST
1372                                    ;                                   ;R2=0 IF NO REQUEST IN QUEUE
1373
1374 036112  005002                  GETREQ: CLR     R2
1375 036114  105761  035446                  TSTB    QCNT(R1)            ;IS THERE ANY REQUEST IN QUEUE?
1376 036120  001404                          BEQ     2$                  ;NO
1377 036122  006301                  1$:     ASL     R1
1378 036124  017102  035476                  MOV     @QOUTPT(R1),R2      ;PICKUP "DPB" POINTER FOR THIS DRIVE
1379 036130  006201                          ASR     R1
1380 036132  000207                  2$:     RTS     PC                  ;RETURN TO USER
1381
1382                                    ;ROUTINE TO "POP" THE REQUEST FROM QUEUE
1383                                    ;
1384                                    ;CALL
1385                                    ;       MOV     #DRVNUM,R1          ;DRIVE NUMBER TO R1
1386                                    ;       JSR     PC,POPQUE           ;CALL TO REMOVE REQUEST
1387                                    ;       RETURN                      ;R2=ADDRESS OF DPB REMOVED
1388
1389 036134  105361  035446          POPQUE: DECB    QCNT(R1)            ;DECREMENT QUEUE COUNT
1390 036140  006301                          ASL     R1
1391 036142  017102  035476                  MOV     @QOUTPT(R1),R2      ;GET THE "DPB" POINTER
1392 036146  005071  035476                  CLR     @QOUTPT(R1)         ;REMOVE DPB ADDRESS FROM THE QUEUE
1393 036152  062761  000002  035476          ADD     #2,QOUTPT(R1)       ;UPDATE THE QUEUE POINTER
1394 036160  026161  035476  035520          CMP     QOUTPT(R1),QSTOP(R1)    ;TIME TO RESET THE POINTER?
1395 036166  001003                          BNE     1$                  ;NO--BR TO EXIT
1396 036170  016161  035516  035476          MOV     QSTART(R1),QOUTPT(R1)   ;YES--RESET THE POINTER
1397 036176  006201                  1$:     ASR     R1
1398 036200  000207                          RTS     PC                  ;RETURN TO USER
1399
1401                                    ;ROUTINE TO SAVE THE CONTENTS OF '$ESCAPE' WHEN THE DRIVER
1402                                    ;REPORTS AN ERROR DIRECTLY.
1403                                    ;
1404                                    ;CALL
1405                                    ;       JSR     R0,ES.SAV
1406                                    ;       ERROR   N                   ;;THE ERROR CALL
1407                                    ;       RETURN                      ;THE RETURN IS PAST THE ERROR CALL
```

```
1408
1409 036202  012037  036216          ES.SAV: MOV     (R0)+,1$        ;GET THE ERROR CALL
1410 036206  013746  001176                  MOV     $ESCAPE,-(SP)   ;SAVE THE ADDRESS IN '$ESCAPE'
1411 036212  005037  001176                  CLR     $ESCAPE         ;CLEAR THE ESCAPE RETURN
1412 036216  000000                  1$:     .WORD   0               ;THE ERROR CALL IS MOVED HERE
1413 036220  012637  001176                  MOV     (SP)+,$ESCAPE   ;RESTORE THE ESCAPE ADDRESS
1414 036224  000200                          RTS     R0              ;RETURN
1416
```

```
                                .SBTTL  TELETYPE MESSAGES
                        
036226  040     117     106     UNTOFF: .ASCIZ  / OFFLINE/
036237  040     117     116     UNTON:  .ASCIZ  / ONLINE/
036247  040     116     117     NOTRM:  .ASCIZ  @ NOT AN RM05/3/2@
036270  040     116     117     NOTPRS: .ASCIZ  / NOT PRESENT/
036305  040     125     116     NOTSAF: .ASCIZ  / UNSAFE/
036315  040     111     123     LODEV:  .ASCIZ  / IS LOAD DEVICE/
036335  122     115     060     $RM02:  .ASCIZ  /RM02/
036342  122     115     060     $RM03:  .ASCIZ  /RM03/
036347  122     115     060     $RM05:  .ASCIZ  /RM05/
036354  200     125     116     UNSTAT: .ASCIZ  <CRLF>/UNIT STATUS:/
036372  200     104     122     MUNIT:  .ASCIZ  <CRLF>/DRIVE: /
036403  060     000             ZERO:   .ASCIZ  /0/
036405  040     057     040     SLASH:  .ASCIZ  @ / @
036411  103     000             C:      .ASCIZ  /C/
036413  106     000             F:      .ASCIZ  /F/
036415  124     000             T:      .ASCIZ  /T/
036417  131     000             Y:      .ASCIZ  /Y/
036421  062     000             TWO:    .ASCIZ  /2/
036423  040                     BLNKS4: .ASCII  / /
036424  040                     BLNKS3: .ASCII  / /
036425  040                     BLNKS2: .ASCII  / /
036426  040     000             BLNKS1: .ASCIZ  / /
036430  105     116     124     ENTADR: .ASCIZ  /ENTER ADDRESS LIMITS:/<CRLF>
036457  040     077     104     MOFFLN: .ASCIZ  / ?DRIVE OFFLINE/<CRLF>
036500  040     077     104     MDRNP:  .ASCIZ  / ?DRIVE NOT PRESENT/<CRLF>
036525  040     077     104     MER11:  .ASCIZ  / ?DRIVE NOT AVAILABLE/<CRLF>
036554  040     077     104     MUSDR:  .ASCIZ  / ?DRIVE UNSAFE/<CRLF>
036574  040     077     104     MLODEV: .ASCIZ  / ?DRIVE IS LOAD DEVICE/<CRLF>
036624  040     123     105     MSELD:  .ASCIZ  / SELECTED/
036636  200     120     122     MMODE:  .ASCIZ  <CRLF>/PROGRAM MODE (F, C OR V): /
036672  200     106     117     MFORMT: .ASCIZ  <CRLF>/FORMAT & VERIFY/
036713  200     106     117     MCHECK: .ASCIZ  <CRLF>/FORMAT & VERIFY WITH ROTATING DATA PATTERN/
036767  200     126     105     MVERFY: .ASCIZ  <CRLF>/VERIFY ONLY/
037004  200     117     120     MSIZE:  .ASCIZ  <CRLF>/OPERATE IN 32 SECTOR MODE (Y OR N) ? /
037053  200     117     120     MSEC30: .ASCIZ  <CRLF>/OPERATION WILL BE IN 30 SECTOR (18 BIT) MODE/<CRLF>
037132  200     117     120     MSEC32: .ASCIZ  <CRLF>/OPERATION WILL BE IN 32 SECTOR (16 BIT) MODE/<CRLF>
037211  111     116     126     BADENT: .ASCIZ  /INVALID ENTRY/<CRLF>
037230  105     116     104     MADRER: .ASCII  /ENDING DISK ADRS MUST BE EQUAL TO OR GREATER/<CRLF>
037305  124     110     101             .ASCIZ  /THAN STARTING ADRS/<CRLF>
037331  200     123     105     MSELP:  .ASCII  <CRLF>/SELECT DATA PATTERN/<CRLF>
037356  040     050     060             .ASCII  / (0) ZERO'S /<CRLF>
037373  040     050     061             .ASCII  / (1) ONE'S/<CRLF>
037406  040     050     062             .ASCII  / (2) WORST CASE/<CRLF>
037426  105     116     124             .ASCIZ  /ENTER (0, 1 OR 2): /
037452  200     123     124     MSFOR:  .ASCIZ  <CRLF>/STARTING FORMAT ON DRIVE /
037505  200     123     124     MSCHK:  .ASCIZ  <CRLF>/STARTING CHECK ON DRIVE /
037537  200     123     124     MSQVER: .ASCIZ  <CRLF>/STARTING QUICK VERIFY ON DRIVE /
037600  200     123     124     MSVER:  .ASCIZ  <CRLF>/STARTING VERIFY ON DRIVE /
037633  200     106     117     MFCMPT: .ASCIZ  <CRLF>/FORMAT COMPLETE, /
037656  200     103     110     MCCMPT: .ASCIZ  <CRLF>/CHECK COMPLETE, /
037700  200     126     105     MVCMPT: .ASCIZ  <CRLF>/VERIFY COMPLETE, /
037723  124     117     124     NUMERR: .ASCIZ  /TOTAL ERRORS DETECTED: /
037753  200     052     052     STARS:  .ASCIZ  <CRLF>/**************/<CRLF>
037775  120     122     105     ADDRIS: .ASCIZ  /PRESENT ADDRESS IS: /
040022  105     116     124     MESG1:  .ASCIZ  /ENTER THE SERIAL NUMBER (MAX. 10 OCTAL DIGITS)/
```

```
 58 040101  200  123  116  SN1:     .ASCIZ  <CRLF>/SN # : /
 59 040112  200  117  126  MESG3:   .ASCII  <CRLF>/OVER 126. BAD SECTORS HAVE BEEN DETECTED/<CRLF>
 60 040164  120  101  103           .ASCIZ  /PACK NOT ACCEPTABLE !/
 61 040212  200  111  116  MESG2:   .ASCII  <CRLF>/INCORRECT MANUFACTURES DEFINED BAD SECTOR INFORMATION/
 62 040300  200  012  111           .ASCIZ  <CRLF><LF>/INITIALIZE THE BAD SECTOR FILE (Y OR N) ?/
 63 040354  200  101  114  MESG4:   .ASCII  <CRLF>/ALIGNMENT PACK/
 64 040373  200  120  122  MHALT:   .ASCIZ  <CRLF>/PROGRAM HALTED !/
 65 040415  200  123  105  MSGBLK:  .ASCII <CRLF>/SELECT ONE OF THE FOLLOWING FUNCTIONS/<CRLF>
 66 040464  113  105  131           .ASCII /KEYIN U.S.R. BAD SECTORS .....................0/<CRLF>
 67 040544  120  122  111           .ASCII /PRINT M.F.G. BAD SECTORS.....................1/<CRLF>
 68 040624  120  122  111           .ASCII /PRINT U.S.R. BAD SECTORS.....................2/<CRLF>
 69 040704  111  116  111           .ASCII /INITIALIZE U.S.R. BAD SECTOR FILE............3/<CRLF>
 70 040764  120  122  111           .ASCII /PRINT A SECTOR OF THE LAST TRACK.............4/<CRLF>
 71 041044  120  122  111           .ASCII /PRINT THIS MESSAGE...........................5/<CRLF>
 72 041124  105  130  111           .ASCIZ /EXIT .....................................<CR>/
 73 041204  200  105  116  MUTLTY:  .ASCIZ <CRLF>/ENTER UTILITY FUNCTION: /
 74 041236  200  105  116  MESG5:   .ASCIZ <CRLF>/ENTER 16 BIT M.F.G. BAD SECTOR ADDRESS IN DECIMAL/
 75 041321  200  105  116  MESG6:   .ASCIZ <CRLF>/ENTER 18 BIT M.F.G. BAD SECTOR ADDRESS IN DECIMAL/
 76 041404  200  105  116  MESG7:   .ASCIZ <CRLF>/ENTER 16 BIT U.S.R. BAD SECTOR ADDRESS IN DECIMAL/
 77 041467  200  105  116  MESG8:   .ASCIZ <CRLF>/ENTER 18 BIT U.S.R. BAD SECTOR ADDRESS IN DECIMAL/
 78 041552  200  061  066  MESG9:   .ASCIZ <CRLF>/16 BIT M.F.G. BAD SECTOR FILE IN OCTAL/
 79 041622  200  061  070  MESG10:  .ASCIZ <CRLF>/18 BIT M.F.G. BAD SECTOR FILE IN OCTAL/
 80 041672  200  061  066  MESG11:  .ASCIZ <CRLF>/16 BIT U.S.R. BAD SECTOR FILE IN OCTAL/
 81 041742  200  061  070  MESG12:  .ASCIZ <CRLF>/18 BIT U.S.R. BAD SECTOR FILE IN OCTAL/
 82 042012  105  116  124  MESG13:  .ASCIZ /ENTER SECTOR NUMBER : /
 83 042041  104  117  116  MESG14:  .ASCIZ /DONE !/<CRLF>
 84 042051  200  105  116  MESG15:  .ASCIZ <CRLF>/ENTER M.F.G. BAD SECTOR ADDRESS IN DECIMAL/<CRLF>
 85                        ;MESG16:.ASCIZ /ILLEGAL BAD SECTOR, PACK NOT ACCEPTABLE !/<CRLF>
 86 042126  127  122  111  MESG17:  .ASCII /WRITE BAD SECTOR FILE NOT SUCCESSFUL,/<CRLF>
 87 042174  120  101  103           .ASCIZ /PACK IS NOT ACCEPTABLE !/<CRLF>
 88
 89                                .EVEN
 90
```

```
                                .SBTTL  ERROR MESSAGES

042226  122  110  057  EM1:    .ASCIZ  @RH/RM INTERRUPT OCCURRED (RMAS=0)@
042270  125  116  105  EM2:    .ASCIZ  /UNEXPECTED ATTENTION OCCURRED/
042326  115  101  123  EM3:    .ASCIZ  /MASSBUS PARITY ERROR (MCPE=1)/
042364  115  101  123  EM4:    .ASCIZ  /MASSBUS PARITY ERROR (PAR=1)/
042421  101  104  104  EM5:    .ASCIZ  /ADDRESS PLUG CHANGE BIT SET/
042455  122  110  057  EM6:    .ASCIZ  @RH/RM DIDN'T RESPOND TO ADDRESSING@
042520  124  122  101  EM7:    .ASCIZ  @TRACK/SECTOR FIELD IN HEADER IS INCORRECT@
042572  104  122  111  EM10:   .ASCIZ  /DRIVE OFFLINE/
042610  120  105  122  EM11:   .ASCIZ  /PERSISTENT DRIVE UNSAFE ERROR/
042646  125  116  103  EM12:   .ASCIZ  /UNCORRECTABLE MASSBUS PARITY ERROR/
042711  123  117  106  EM13:   .ASCIZ  /SOFTWARE TIMEOUT/
042732  104  122  111  EM14:   .ASCIZ  /DRIVE UNSAFE ERROR/
042755  103  117  116  EM15:   .ASCIZ  @CONTROLLER/DRIVE ERROR DURING WRITE@
043021  103  117  116  EM16:   .ASCIZ  @CONTROLLER/DRIVE ERROR DURING WRITE CHECK@
043073  122  105  124  EM17:   .ASCIZ  @RETRIES NOT SUCESSFUL - SECTOR NOT ACCEPTABLE@
043151  104  101  124  EM20:   .ASCIZ  @DATA ERROR DURING WRITE CHECK@
043207  103  117  116  EM21:   .ASCIZ  @CONTROLLER/DRIVE ERROR VERIFYING HEADERS@
043260  047  110  103  EM22:   .ASCIZ  @'HCE' OR 'HCRC' ERROR VERIFYING HEADERS@
043330  103  131  114  EM23:   .ASCIZ  @CYLINDER FIELD IN HEADER IS NOT CORRECT@
043400  127  122  111  EM24:   .ASCIZ  @WRITE CHECK ERROR@
043422  111  114  114  EM25:   .ASCIZ /ILLEGAL BAD SECTOR, PACK NOT ACCEPTABLE/
```

```
043472  122  115  101  DH1:    .ASCIZ  /RMAS   RMCS1   RMER1   RMER2   RMDS    RMDC    RMDA/
043540  104  122  111  DH2:    .ASCIZ  /DRIVE   RMDS   RMER1   RMER2   RMAS    RMCS1/
043612  104  122  111  DH3:    .ASCIZ  /DRIVE   REG ADR  DATA  RMCS1   RMER1   RMER2/
043663  104  122  111  DH4:    .ASCIZ  /DRIVE   REG ADR    GOOD     BAD   RMCS1 RMER1   RMER2/
043746  122  110  040  DH6:    .ASCIZ  /RH ADDRESS/
043761  104  122  111  DH10:   .ASCIZ  /DRIVE   ERR PC  RMCS1   RMCS2   RMDS    RMER1   RMER2/
044046  122  115  105  DH10A:  .ASCIZ  /RMEC1  RMEC2   RMWC    RMBA    RMDA    RMAS    RMLA/
044132  122  115  104  DH10B:  .ASCIZ  /RMDB    RMMR1  RMDT    RMSN    RMOF  CYLINDER   TRACK/
044210  104  122  111  DH17:   .ASCIZ  /DRIVE   ERR PC CYLINDER TRACK   SECTOR/
044257  104  122  111  DH20:   .ASCIZ  /DRIVE   ERR PC CYLINDER TRACK   SECTOR/
044326  122  115  103  DH20A:  .ASCIZ  /RMCS1   RMCS2   RMDS   RMER1   RMER2   RMMR2   RMEC1   RMEC2/
044423  122  115  127  DH20B:  .ASCIZ  /RMWC    RMBA    RMDA    RMAS    RMLA    RMDB    RMMR1    RMDT/
044521  122  115  123  DH20C:  .ASCIZ  /RMSN   RMOF  CYLINDER  TRACK/
044554  104  122  111  DH23:   .ASCIZ  /DRIVE   ERR PC  EXPCTD  RECVED/
044613  040  040  040  DH24:   .ASCII  /                                        MEMORY  DISK/<CR><LF>
044701  104  122  111          .ASCIZ  /DRIVE   ERR PC  CYLNDR  TRACK   SECTOR  DATA    DATA/
044766  104  122  111  DH25:   .ASCIZ  /DRIVE   CYLNDR  TRACK   SECTOR  /

                       .EVEN
```

```
    1 045030  001372  005560  005574  DT1:    .WORD   ATTN,RM.REG,RM.REG+14,RM.REG+42,RM.REG+12,RM.REG+34,RM.REG+6
    2 045046  001370  027666  027670  DT2:    .WORD   DDRIVE,RMERRS,RMERRS+2,RMERRS+4,ATTN,RMCS1
    3 045062  001370  034730  034732  DT3:    .WORD   DDRIVE,RD.ADR,RD.WRD,RM.REG,RM.REG+14,RM.REG+42
    4 045076  001370  035154  035152  DT4:    .WORD   DDRIVE,WRT.AD,WRT.WD,RD.WRD,RM.REG,RM.REG+14,RM.REG+42
    5 045114  001274          DT6:    .WORD   $RMADR
    6 045116  001222  001132  045772  DT7:    .WORD   DRIVE,$ERRPC,RBUF+6,RBUF+2
    7 045126  001222  001132  005560  DT10:   .WORD   DRIVE,$ERRPC,RM.REG,RM.REG+10,RM.REG+12,RM.REG+14,RM.REG+42
    8 045144  005624  005626  005562          .WORD   RM.REG+44,RM.REG+46,RM.REG+2,RM.REG+4,RM.REG+6,RM.REG+16,RM.REG+20
    9 045162  005602  005604  005606          .WORD   RM.REG+22,RM.REG+24,RM.REG+26,RM.REG+30,RM.REG+32,DS.CYL,DS.TRK
   10 045200  001222  001132  001364  DT17:   .WORD   DRIVE,$ERRPC,DS.CYL,DS.TRK,SAVSEC
   11 045212  001222  001132  001364  DT20:   .WORD   DRIVE,$ERRPC,DS.CYL,DS.TRK,SAVSEC
   12 045224  005560  005570  005572          .WORD   RM.REG,RM.REG+10,RM.REG+12,RM.REG+14,RM.REG+42,RM.REG+40,RM.REG+44,RM.REG+46
   13 045244  005562  005564  005566          .WORD   RM.REG+2,RM.REG+4,RM.REG+6,RM.REG+16,RM.REG+20,RM.REG+22,RM.REG+24,RM.REG+26
   14 045264  005610  005612  001364          .WORD   RM.REG+30,RM.REG+32,DS.CYL,DS.TRK
   15 045274  001222  001132  045770  DT23:   .WORD   DRIVE,$ERRPC,RBUF+4,RBUF
   16 045304  001222  001132  001364  DT24:   .WORD   DRIVE,$ERRPC,DS.CYL,DS.TRK,SAVSEC,$GDDAT,RM.REG+22
   17 045322  005560  005570  005572          .WORD   RM.REG,RM.REG+10,RM.REG+12,RM.REG+14,RM.REG+42,RM.REG+40,RM.REG+44,RM.REG+46
   18 045342  005562  005564  005566          .WORD   RM.REG+2,RM.REG+4,RM.REG+6,RM.REG+16,RM.REG+20,RM.REG+22,RM.REG+24,RM.REG+26
   19 045362  005610  005612  001364          .WORD   RM.REG+30,RM.REG+32,DS.CYL,DS.TRK
   20
   21 045372  001222  005552  001366  DT25:   .WORD   DRIVE,FMTDPB+12,DS.TRK,SAVSEC
   22
```

```
 1 045402  000001            DF1:    .WORD   1
 2 045404     007    000             .BYTE   7,0
 3 045406  000001            DF2:    .WORD   1
 4 045410     006    000             .BYTE   6,0
 5 045412  000001            DF3:    .WORD   1
 6 045414     006    000             .BYTE   6,0
 7 045416  000001            DF4:    .WORD   1
 8 045420     007    000             .BYTE   7,0
 9 045422  000001            DF6:    .WORD   1
10 045424     001    000             .BYTE   1,0
11 045426  000003            DF10:   .WORD   3
12 045430     007    000             .BYTE   7.,0
13 045432  044046                    .WORD   DH10A
14 045434     007    000             .BYTE   7,0
15 045436  044132                    .WORD   DH10B
16 045440     007    140             .BYTE   7,140
17 045442  000001            DF17:   .WORD   1
18 045444     005    034             .BYTE   5,34
19 045446  000004            DF20:   .WORD   4
20 045450     005    034             .BYTE   5,34
21 045452  044326                    .WORD   DH20A
22 045454     010    000             .BYTE   8.,0
23 045456  044423                    .WORD   DH20B
24 045460     010    000             .BYTE   8.,0
25 045462  044521                    .WORD   DH20C
26 045464     004    014             .BYTE   4,14
27 045466  000001            DF23:   .WORD   1
28 045470     005    000             .BYTE   5,0
29 045472  000004            DF24:   .WORD   4
30 045474     007    034             .BYTE   7,34
31 045476  044326                    .WORD   DH20A
32 045500     010    000             .BYTE   8.,0
33 045502  044423                    .WORD   DH20B
34 045504     010    000             .BYTE   8.,0
35 045506  044521                    .WORD   DH20C
36 045510     004    014             .BYTE   4,14
37 045512  000001            DF25:   .WORD   1
38 045514     004    000             .BYTE   4,0
39
```

```
                                .SBTTL  BUSADR - GET BUS ADDRESS AND VECTOR ADDRESS FOR RH/RM

                                ;THIS ROUTINE IS USED TO INSURE THE BUS ADDRESS
                                ;OF THE RH/RM IS SETUP FOR THE PROPER ADDRESS.
                                ;IT WILL ALSO READ THE ADDRESS FROM THE TTY IF
                                ;REQUIRED.
                                ;NOTE: THIS ROUTINE DESTROYS R0-R4
                                ;CALL
                                ;
                                ;       JSR     PC,BUSADR
                                ;       RETURN

045516  005737  001376  BUSADR: TST     CHGADR          ;INPUT FROM TTY REQUESTED?
045522  001436                   BEQ     3$              ;NO
045524  005037  001376           CLR     CHGADR          ;DISABLE CHANGE OF RH/RM ADDRESS
045530  104401  001205           TYPE    ,$CRLF          ;CR-LF
045534  012700  001274  1$:      MOV     #$RMADR,R0      ;FIRST ADDRESS
045540  104401  045636           TYPE    ,MRMCS1         ;"RMCS1="
045544  011046                   MOV     (R0),-(SP)      ;PRESENT RMCS1 ADDRESS
045546  104402                   TYPOC                   ;TYPE IT
045550  104401  036425           TYPE    ,BLNKS2         ;TYPE 2 SPACES
045554  104411                   RDLIN                   ;GET THE ENTRY
045556  012601                   MOV     (SP)+,R1        ;ADDRESS OF ASCII TEXT
045560  004537  045654           JSR     R5,CK.NUM       ;ENTER AND STORE NEW ADDRESS
045564  000763                   BR      1$              ;ERROR RET
045566  012700  001276  2$:      MOV     #$RMVEC,R0      ;CHECK VERC
045572  104401  045645           TYPE    ,MRMVEC         ;"RMVEC="
045576  011046                   MOV     (R0),-(SP)      ;PRESENT RH/RM VECTOR ADDRESS ON THE STACK
045600  104402                   TYPOC                   ;TYPE IT
045602  104401  036425           TYPE    ,BLNKS2         ;TYPE 2 SPACES
045606  104411                   RDLIN                   ;READ THE ENTRY
045610  012601                   MOV     (SP)+,R1        ;ASCII TEXT ADDRESS
045612  004537  045654           JSR     R5,CK.NUM       ;ENTER AND STORE NEW ADDRESS
045616  000763                   BR      2$              ;ERROR RET
045620  012700  001274  3$:      MOV     #$RMADR,R0      ;FIRST ADDRESS OF NEW PARAMETERS
045624  012701  030034           MOV     #RMADR,R1       ;FIRST ADDRESS OF WHERE TO PUT THEM
045630  012021                   MOV     (R0)+,(R1)+     ;BUS ADDRESS
045632  012021                   MOV     (R0)+,(R1)+     ;VECTOR ADDRESS
045634  000207                   RTS     PC              ;RETURN

045636    122     115     103  MRMCS1: .ASCIZ  @RMCS1=@
045645    122     115     126  MRMVEC: .ASCIZ  @RMVEC=@

                                .SBTTL  CK.NUM - CHECK NUMBER (OCTAL)
                                ;THIS ROUTINE CHECKS AN ASCIZ STRING FOR LEGAL CHARACTERS
                                ;AND FORMS AN OCTAL NUMBER IN R2
                                ;CALL:
                                ;       MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
                                ;       R0=ADDRESS TO STORE THE NUMBER
                                ;       JSR     R5,CK.NUM
                                ;       RET1    ERROR RETURN
                                ;       RET2    NORMAL RET


045654  010246  CK.NUM: MOV     R2,-(SP)        ;SAVE R2
045656  010346                   MOV     R3,-(SP)                ;SAVE R3
```

```
045660  010446                  MOV     R4,-(SP)        ;SAVE R4
045662  012703  000006          MOV     #6,R3           ;MAX OCTAL DIGITS IN THE NUMBER
045666  005002                  CLR     R2              ;FINAL OCTAL VALUE
045670  112104          1$:     MOVB    (R1)+,R4        ;GET CURRENT POINTED BYTE
045672  001424                  BEQ     3$              ;BR IF TERMINATOR DETECTED
045674  120427  000060          CMPB    R4,#'0          ;SMALLER THAN ASCII-0 ?
045700  103425                  BLO     5$              ;YES,ERROR EXIT
045702  120427  000067          CMPB    R4,#'7          ;LARGER THAN ASCII-7 ?
045706  101022                  BHI     5$              ;YES,ERROR EXIT
045710  006302                  ASL     R2              ;SHIFT LEFT
045712  103420                  BCS     5$              ;
045714  006302                  ASL     R2              ;ONE
045716  103416                  BCS     5$              ;
045720  006302                  ASL     R2              ;OCTAL DIGIT
045722  103414                  BCS     5$              ;ERROR IF CARRY BIT SET
045724  042704  177770          BIC     #177770,R4      ;CHOP OFF HIGHER BITS
045730  060402                  ADD     R4,R2           ;APPENDING CURRENT DIGIT
045732  005303                  DEC     R3              ;DECREMENT BYTE COUNT
045734  001401                  BEQ     2$              ;BR IF LAST DIGIT
045736  000754                  BR      1$              ;LOOPING BACK
045740  112104          2$:     MOVB    (R1)+,R4        ;CHECK TERMINATOR
045742  001004                  BNE     5$              ;BR IF NOT FOUND
045744  005702          3$:     TST     R2              ;FINAL VALUE = 0 ?
045746  001401                  BEQ     4$              ;YES
045750  010210                  MOV     R2,(R0)         ;REPLACE THE ORIGINAL VALUE
045752  005725          4$:     TST     (R5)+           ;ADJUST FOR NORMAL RET
045754  012604          5$:     MOV     (SP)+,R4        ;RESTORE 4
045756  012603                  MOV     (SP)+,R3        ;RESTORE R3
045760  012602                  MOV     (SP)+,R2        ;RESTORE R2
045762  000205                  RTS     R5              ;EXIT
88
```

```
                                        .SBTTL  READ/WRITE BUFFERS
                                        ;BUFFER STARTS HERE
045764  000000  000000  000000  RBUF:   .WORD   0,0,0,0          ;BUFFER FOR HEADER CHECK
045774                        BUFP:   .BLKW   <258.*32.>        ;FORMAT AND CHECK BUFFER STARTS HERE AND
                                                                ;WILL OVERLAY ALL REMAINING CODE IN PROGRAM
106174                        BUFV:   .BLKW   <258.*32.>        ;BUFFER FOR VERIFY ONLY MODE

        000200                  .END    200
```

SYMBOL TABLE

```
ABASE = 176700        AT0    = 000001        BYTE16  006332        DH6     043746        EM14    042732
ACDW1 = 000000        AT1    = 000002        BYTE18  006432        DISPLA  001156        EM15    042755
ACDW2 = 000000        AT2    = 000004        C       036411        DISPRE  000174        EM16    043021
ACK   = 000123        AT3    = 000010        CALIN   013322        DLT    = 100000       EM17    043073
ACPUOP= 000000        AT4    = 000020        CHGADR  001376        DMD    = 000001       EM2     042270
ACTDRV  027752        AT5    = 000040        CI1     031352        DONE    017006        EM20    043151
ACTSTR  027753        AT6    = 000100        CI3     031460        DPINT   027726        EM21    043207
ADDRIS  037775        AT7    = 000200        CI4     031560        DPR    = 000400       EM22    043260
ADDW0 = 000000        AUNIT  = 000000        CI5     032136        DPRQS   027736        EM23    043330
ADDW1 = 000000        AUSWR  = 000000        CI6     032160        DRIVE  = 001222       EM24    043400
ADDW10= 000000        AVECT1= 000254         CI7     032174        DRQ    = 004000       EM25    043422
ADDW11= 000000        AVECT2= 000000         CI7B    032216        DRVACT  027676        EM3     042326
ADDW12= 000000        A16    = 000400        CI8     032274        DRVCLR= 000111        EM4     042364
ADDW13= 000000        A17    = 001000        CKSWR  = 104407       DRVINT  030264        EM5     042421
ADDW14= 000000        BADENT  037211         CKTRK   015006        DRVQUE  036036        EM6     042455
ADDW15= 000000        BAD16   001414         CK.CHR  023410        DRVSTA  027706        EM7     042520
ADDW2 = 000000        BAD18   002420         CK.DEC  023362        DRVTYP  027716        ENDCYL  001326
ADDW3 = 000000        BAI    = 000010        CK.DIG  023462        DRY    = 000200       ENDTRK  001332
ADDW4 = 000000        BEGCYL  001330         CK.NUM  045654        DSWR   = 177570       ENTADR  036430
ADDW5 = 000000        BEGIN   006776         CK.OCT  023334        DS.CYL  001364        ERINDX  017204
ADDW6 = 000000        BEGIN2  007020         CLOCK   020250        DS.TRK  001366        ERR    = 040000
ADDW7 = 000000        BEGTRK  001334         CLR    = 000040       DTE    = 010000       ERROR = 104000
ADDW8 = 000000        BIT0   = 000001        CLRQUE  035740        DTSY   = 000200       ERRVEC= 000004
ADDW9 = 000000        BIT00  = 000001        CNTLC   001374        DTUW    030020        ES.SAV  036202
ADEVCT= 000000        BIT01  = 000002        CR     = 000015       DT00   = 000001       F       036413
ADEVM = 000000        BIT02  = 000004        CRLF   = 000200       DT01   = 000002       FER    = 000020
ADRTBL  006132        BIT03  = 000010        C.ENTR  020262        DT02   = 000004       FMT    = 010000
AENV  = 000000        BIT04  = 000020        DCK    = 100000       DT03   = 000010       FMTDPB  005540
AENVM = 000000        BIT05  = 000040        DDISP  = 177570       DT04   = 000020       FMTX    005630
AFATAL= 000000        BIT06  = 000100        DDRIVE  001370        DT05   = 000040       FMT16  = 010000
AMADR1= 000000        BIT07  = 000200        DEVMP   001402        DT06   = 000100       FUNCT1  021714
AMADR2= 000000        BIT08  = 000400        DF1     045402        DT07   = 000200       FUNCT2  022116
AMADR3= 000000        BIT09  = 001000        DF10    045426        DT08   = 000400       FUNCT3  022414
AMADR4= 000000        BIT1   = 000002        DF17    045442        DT1     045030        FUNCT4  022502
AMAMS1= 000000        BIT10  = 002000        DF2     045406        DT10    045126        FUNCT5  022612
AMAMS2= 000000        BIT11  = 004000        DF20    045446        DT17    045200        FUNCT6  022710
AMAMS3= 000000        BIT12  = 010000        DF23    045466        DT2     045046        FUNCT7  023014
AMAMS4= 000000        BIT13  = 020000        DF24    045472        DT20    045212        FUNCT8  023122
AMSGAD= 000000        BIT14  = 040000        DF25    045512        DT23    045274        F1     = 000002
AMSGLG= 000000        BIT15  = 100000        DF3     045412        DT24    045304        F2     = 000004
AMSGTY= 000000        BIT2   = 000004        DF4     045416        DT25    045372        F3     = 000010
AMTYP1= 000000        BIT3   = 000010        DF6     045422        DT3     045062        F4     = 000020
AMTYP2= 000000        BIT4   = 000020        DH1     043472        DT4     045076        F5     = 000040
AMTYP3= 000000        BIT5   = 000040        DH10    043761        DT6     045114        GETREG= 000141
AMTYP4= 000000        BIT6   = 000100        DH10A   044046        DT7     045116        GETREQ  036112
AOE   = 001000        BIT7   = 000200        DH10B   044132        DULP    030266        GO     = 000001
APASS = 000000        BIT8   = 000400        DH17    044210        DVA    = 004000       GTSWR  = 104406
APRIOR= 000000        BIT9   = 001000        DH2     043540        ECH    = 000100       HCE    = 000200
APTCSU= 000040        BLNKS1  036426         DH20    044257        ECI    = 004000       HCI    = 002000
APTENV= 000001        BLNKS2  036425         DH20A   044326        EDIT    001406        HCRC   = 000400
APTSIZ= 000200        BLNKS3  036424         DH20B   044423        EMPTYQ  036016        HDREAD  015550
APTSPO= 000100        BLNKS4  036423         DH20C   044521        EMTVEC= 000030        HEADX   001410
ASWREG= 000000        BPTVEC= 000014         DH23    044554        EM1     042226        HEDERR  001356
ATA   = 100000        BSE    = 100000        DH24    044613        EM10    042572        HSET    015450
ATABIT  030022        BUFP    045774         DH25    044766        EM11    042610        HT     = 000011
ATESTN= 000000        BUFV    106174         DH3     043612        EM12    042646        IAE    = 002000
ATTN    001372        BUSADR  045516         DH4     043663        EM13    042711        IE     = 000100
```

```
ILF   = 000001
ILR   = 000002
INSERT  021602
IOTVEC= 000020
IR    = 000100
ISR     032706
LA      032532
LACNT   027764
LF    = 000012
LODEV   036315
LOP.CK  017304
LST   = 002000
LSTRK   001324
MADRER  037230
MAXSEC  001362
MCCMPT  037656
MCHECK  036713
MCLK  = 000002
MCPE  = 020000
MCPEMX  030032
MDRNP   036500
MER11   036525
MESG1   040022
MESG10  041622
MESG11  041672
MESG12  041742
MESG13  042012
MESG14  042041
MESG15  042051
MESG17  042126
MESG2   040212
MESG3   040112
MESG4   040354
MESG5   041236
MESG6   041321
MESG7   041404
MESG8   041467
MESG9   041552
MF    = 100000
MFCMPT  037633
MFORMT  036672
MHALT   040373
MINX  = 000004
MLODEV  036574
MMODE   036636
MNDLTA  030046
MODE    001320
MOFFLN  036457
MOH   = 020000
MOL   = 010000
MPE   = 000400
MRD   = 000020
MRMCS1  045636
MRMVEC  045645
MSCHK   037505
MSEC30  037053
MSEC32  037132
MSELD   036624
MSELP   037331
MSFOR   037452
MSGBLK  040415
MSIZE   037004
MSQVER  037537
MSTCK = 000010
MSVER   037600
MUNIT   036372
MUSDR   036554
MUTLTY  041204
MVCMPT  037700
MVERFY  036767
MWC     001354
MWR   = 000040
MXDLTA  030044
MXF   = 001000
MXLACT  030042
MXWNDW  030050
M0      011342
M1      010776
M1A     011132
M1B     011306
M2      011720
M4      012000
M4A     012026
M5      012146
M5FX    010466
NBA   = 100000
NED   = 010000
NEM   = 004000
NEXT    005536
NOTPRS  036270
NOTRM   036247
NOTSAF  036305
NUMERR  037723
OFDIR = 000001
OFFSET= 000115
OPI   = 020000
OPT     031114
OR    = 000200
O.ENTR  020302
PACK    001404
PAR   = 000010
PARENT  020454
PAR1    005776
PAR2    006011
PAR3    006024
PAR4    006035
PAR5    006046
PAR6    006057
PAR7    006070
PAR8    006101
PAT   = 000020
PATA    001340
PATB    001342
PATSEL  001336
PGE   = 002000
PGM   = 001000
PIP   = 020000
PIRQ  = 177772
PIRQVE= 000240
POPQUE  036134
PR0   = 000000
PR1   = 000040
PR2   = 000100
PR3   = 000140
PR4   = 000200
PR5   = 000240
PR6   = 000300
PR7   = 000340
PS    = 177776
PSEL  = 002000
PSW   = 177776
PWRVEC= 000024
QCNT    035446
QDRV0   035540
QDRV1   035560
QDRV2   035600
QDRV3   035620
QDRV4   035640
QDRV5   035660
QDRV6   035700
QDRV7   035720
QINPT   035456
QOUTPT  035476
QSTART  035516
QSTOP   035520
QTERM = 035740
RBUF    045764
RDCHR = 104410
RDDAT = 000171
RDHD  = 000173
RDLIN = 104411
RDY   = 000200
RD.ADR  034730
RD.RM   034704
RD.RM1  034726
RD.RM2  035052
RD.RM3  035056
RD.RM4  035062
RD.WRD  034732
READIN= 000121
RECAL = 000107
RECORD  020570
REJCT2  017134
REJCT3  017162
REJEC1  017112
RELSE = 000113
RESET   021416
RESREG= 104413
RESTRT  020160
RESVEC= 000010
RETBAD  012230
RETRY   001344
RMADR   030034
RMAS  = 000016
RMBA  = 000004
RMCS1 = 000000
RMCS2 = 000010
RMDA  = 000006
RMDB  = 000022
RMDC  = 000034
RMDS  = 000012
RMDT  = 000026
RMEC1 = 000044
RMEC2 = 000046
RMERRS  027666
RMER1 = 000014
RMER2 = 000042
RMHR  = 000036
RMINIT  030052
RMLA  = 000020
RMMR1 = 000024
RMMR2 = 000040
RMOF  = 000032
RMR   = 000004
RMSN  = 000030
RMTMR   034272
RMVEC   030036
RMWC  = 000002
RM.REG  005560
RM05    030622
RNOP  = 000101
RTC   = 000117
R6    =%000006
R7    =%000007
SAVEFG  027774
SAVREG= 104412
SAVSEC  001346
SAVWC   001350
SC      033122
SCAWC   017530
SCOPE = 000004
SC01  = 000100
SC02  = 000200
SC04  = 000400
SC10  = 001000
SC11    033754
SC12    034064
SC13    034134
SC20  = 002000
SC3     033172
SC4     033176
SC5     033210
SC6     033404
SC6A    033520
SC7     033646
SC8     033724
SEARCH= 000131
SECCU   005434
SEC30   001360
SEEK  = 000105
SEEKFG  027776
SELDRV= 000145
SELPAT  013462
SERNL   012746
SETFMT= 000143
SETHDR  017556
SETTBL  017352
SETVEC  010126
SET.IE  035374
SKI   = 040000
SLASH   036405
SNGSEC  001322
SN1     040101
SOFSW   001316
SORT1   014116
SORT2   020710
SRCHWT  027750
STACK = 001100
STARS   037753
START   007014
STCLK1  020074
STCLK2  020144
STCLK3  020150
STKLMT= 177774
ST0     034362
ST01    034416
ST02    034512
ST03    034542
ST05    034566
ST06    034574
ST07    034632
ST08    034662
ST09    034672
ST.CLK  020016
ST1     013462
SVRH70  035256
SWR     001154
SWREG   000176
SW0   = 000001
SW00  = 000001
SW01  = 000002
SW02  = 000004
SW03  = 000010
SW04  = 000020
SW05  = 000040
SW06  = 000100
SW07  = 000200
SW08  = 000400
SW09  = 001000
SW1   = 000002
SW10  = 002000
SW11  = 004000
SW12  = 010000
SW13  = 020000
SW14  = 040000
```

SYMBOL TABLE

```
SW15   = 100000        UPDTRK  017756        $BDADR  001136        $LF     001206        $RTNAD  017004
SW2    = 000004        UPE     = 020000      $BDDAT  001142        $LFLG   025103        $SAVRE  027506
SW3    = 000010        USSAV   004430        $BELL   001200        $LKCSB  001302        $SAVR6  027200
SW4    = 000020        USTAB   003424        $CDW1   001270        $LKCSR  001300        $SB2D   027256
SW5    = 000040        US1     = 000001      $CDW2   001272        $LKS    001310        $SETUP= 000116
SW6    = 000100        US2     = 000002      $CHARC  024634        $LLVEC  001312        $STUP  = 177777
SW7    = 000200        US4     = 000004      $CKSWR  026064        $LPADR  001122        $SUPRS  027216
SW8    = 000400        VV      = 000100      $CMTAG  001114        $LPERR  001124        $SVPC  = 000210
SW9    = 001000        WC      001352        $CM3    = 000000      $LPVEC  001304        $SWR   = 123000
T        036415        WCE     = 040000      $CM4    = 000001      $MADR1  001242        $SWREG  001232
TABCAL   006120        WCF     = 000040      $CNTLC  026767        $MADR2  001246        $TESTN  001214
TABINT   012532        WCKD    = 000151      $CNTLG  027001        $MADR3  001252        $TKB    001162
TABLD    012614        WCKHD   = 000153      $CNTLU  026774        $MADR4  001256        $TKCNT  025560
TABLE    005650        WCTBL   006232        $CPUOP  001236        $MAIL   001210        $TKINT  025576
TABLE2   005702        WLE     = 004000      $CRLF   001205        $MAMS1  001240        $TKQEN= 025575
TABLE3   005726        WRAP    001400        $DBLK   025550        $MAMS2  001244        $TKQIN  025562
TABLE4   005752        WRL     = 004000      $DB2D   027312        $MAMS3  001250        $TKQOU  025564
TAB.XY=  001114        WRTDAT= 000161        $DECVL  027472        $MAMS4  001254        $TKQSR  025566
TAP    = 040000        WRTHD   = 000163      $DEVCT  001220        $MBADR  001102        $TKS    001160
TBITVE=  000014        WRTRK   013622        $DEVM   001266        $MFLG   025102        $TKSRV  025646
TD       032752        WRTRKA  014320        $DOAGN  017002        $MNEW   027017        $TMPO   001174
TEXT     021050        WRTRKB  014436        $DTBL   025540        $MSGAD  001224        $TN    = 000002
TIMER    030000        WRTRKC  014522        $ENDAD  016772        $MSGLG  001226        $TNPWR  027422
TKVEC  = 000060        WRTRKD  014604        $ENDCT  016756        $MSGTY  001210        $TPB    001166
TPVEC  = 000064        WRTRKE  014652        $ENV    001230        $MSWR   027006        $TPFLG  001173
TRAPVE=  000034        WRTRKF  014566        $ENVM   001231        $MTYP1  001241        $TPS    001164
TRE    = 040000        WRTRKH  014616        $EOP    016342        $MTYP2  001245        $TRAP   027602
TRKTST   017426        WRTRKX  013676        $EOPCT  016750        $MTYP3  001251        $TRAP2  027624
TRK1   = 004000        WRTRKY  013744        $ERFLG  001117        $MTYP4  001255        $TRP   = 000014
TRK10  = 040000        WRTRKZ  015536        $ERMAX  001131        $NULL   001170        $TRPAD  027636
TRK2   = 010000        WRTRK1  014270        $ERROR  023620        $NWTST= 000000        $TSTM   001104
TRK20  = 100000        WRTRK2  014666        $ERRPC  001132        $OCNT   025330        $TSTNM  001116
TRK4   = 020000        WRT.AD  035154        $ERRTB  006526        $OMODE  025332        $TTYIN  026752
TRNSWT   027746        WRT.RM  035064        $ERTTL  001126        $PASS   001216        $TYPDS  025334
TRTVEC=  000014        WRT.R1  035150        $ESCAP  001176        $PASTM  001106        $TYPE   024304
TST1     007020        WRT.R2  035242        $ETABL  001230        $POWER  027202        $TYPEC  024516
TWO      036421        WRT.R3  035246        $ETEND  001274        $PWRAD  027170        $TYPEX  024636
TYPADR   020354        WRT.R4  035252        $FATAL  001212        $PWRDN  027030        $TYPOC  025132
TYPDS  = 104405        WRT.R5  035254        $FFLG   025104        $PWRMG  027164        $TYPON  025146
TYPE   = 104401        WRT.WD  035152        $FILLC  001172        $PWRUP  027102        $TYPOS  025106
TYPERR   024054        XSIZE   010422        $FILLS  001171        $QUES   001204        $UNIT   001222
TYPOC  = 104402        XXDP    001412        $GDADR  001134        $RDCHR  026426        $UNITM  001110
TYPON  = 104404        Y       036417        $GDDAT  001140        $RDLIN  026516        $USWR   001234
TYPOS  = 104403        ZERO    036403        $GET42  016762        $RDSZ  = 000015        $VECT1  001260
UF     = 040000        $APTHD  001100        $GTSWR  026154        $RESRE  027544        $VECT2  001262
ULDFLG   027754        $ATYC   024664        $HD    = 000000       $RMADR  001274        $XOFF  = 000023
UNS    = 040000        $ATY1   024640        $HIBTS  001100        $RMVEC  001276        $XON   = 000021
UNSTAT   036354        $ATY3   024646        $ICNT   001120        $RM02   036335        $$GET4= 000000
UNTOFF   036226        $ATY4   024656        $ILLUP  027174        $RM03   036342        $OFILL  025331
UNTON    036237        $AUTOB  001150        $INTAG  001151        $RM05   036347        .$X    = 001100
UPDCYL   017644        $BASE   001264        $ITEMB  001130

. ABS.   146374      000
         000000      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  45296 WORDS  ( 177 PAGES)
```

DYNAMIC MEMORY AVAILABLE FOR 69 PAGES

SEQ 0133

CZRMLA.BIC,CZRMLA/C=CZRMLA.DOC,CZRMLA,SYSMAC/M

```
$$GET4  10-568   10-568#
$OFILL  10-?58   10-?58#  10-?58*  10-?58*
$40CAT  10->86
$APTHD   4-286    4-286#
$ASTAT  10-?56   10-?56
$ATY1   10-?56#
$ATY3   10-?54   10-?56#
$ATY4   10->86   10-?56#
$ATYC   10-?56   10-?56#
$AUTOB   5-0#     8-15*    8-60     8-141    9-148    9-217    10-570   10-797   10-?62   10-?62   10-?62
$BASE    5-0#     8-72     8-74
$BDADR   5-0#
$BDDAT   5-0#
$BELL    5-0#    10->86   10->86   10->86   10-?62   10-?62   10-?62
$CDW1    5-0#     8-4*     8-7*     8-163    9-219
$CDW2    5-0#     8-167
$CHARC  10-?54   10-?54#  10-?54*  10-?54*  10-?54*
$CKSWR  10-?62#  10-?75   10-?75
$CM3     5-0      5-0#
$CM4     5-0      5-0      5-0#     5-0#
$CMTAG   5-0#     8-14     8-14     8-14     8-14
$CNTLC  10-840   10-?62   10-?62   10-?62   10-?62   10-?62#
$CNTLG  10-?62   10-?62#
$CNTLU  10-?62   10-?62   10-?62#
$CPUOP   5-0#
$CRLF    5-0#     8-41     8-91     8-139    8-244    8-345    8-394    9-17     9-152    9-231    9-238    10-393   10-593   10-599
        10-606   10-812   10-864   10-925   10-;07   10-;27   10-;59   10-;78   10-<02   10-<10   10-<28   10-<35   10-=02   10-=12
        10->86   10->86   10->86   10-?04   10-?09   10-?19   10-?33   10-?46   10-?54   10-?54   10-?54   10-?62   10-?62   10-?62
        10-?62   17-16
$DB2D   10-?69   10-?71#
$DBLK   10-?60   10-?60   10-?60#
$DECVL  10-?71   10-?71#
$DEVCT   5-0#
$DEVM    5-0#     8-149    8-151
$DOAGN  10-568   10-568   10-568#
$DTBL   10-?60   10-?60#
$ENDAD   4-283   10-568#
$ENDCT  10-568#
$ENV     5-0#     8-15     8-66     10->86   10-?54   10-?56   10-?56
$ENVM    5-0#     8-14     8-146    10-?54   10-?54   10-?56
$EOP     8-155    8-208    10-378   10-457   10-568#  10-659
$EOPCT  10-568   10-568#
$ERFLG   5-0#    10->86   10->86   10->86*
$ERMAX   5-0#     8-14*
$ERROR   8-14    10->86#
$ERRPC   5-0#    10->86   10->86   10->86   10->86*  10->86*  15-6     15-7     15-10    15-11    15-15    15-16
$ERRTB   7-0#    10-?01
$ERTTL   5-0#     8-283*  10-568   10-568*  10->86   10->86   10->86*
$ESCAP   5-0#     8-14*    9-64*    10-248*  10-261*  10-279*  10-294*  10-299*  10-340*  10-416*  10-433*  10-568*  10-653*  10->86
        10->86   10->86   10->86   11-817   11-868   11-870   11-;32   11-;91   11->00   11->10   11->11*  11->13*
$ETABL   5-0#
$ETEND   4-286    5-0#
$FATAL   5-0#    10-?56*
$FFLG   10-?56   10-?56#  10-?56*  10-?56*  10-?56*
$FILLC   5-0#    10-?54   10-?54   10-?54
$FILLS   5-0#    10-?54   10-?54
```

```
$GDADR    5-0#
$GDDAT    5-0#     10->86*  15-16
$GET42   10-568#
$GTSWR   10-?62#   10-?75   10-?75
$HD       4-72      4-72     4-72
$HIBTS    4-286#
$ICNT     5-0#
$ILLUP   10-?64    10-?64   10-?64#
$INTAG    5-0#     10-?62   10-?62   10-?62   10-?62   10-?62*
$ITEMB    5-0#     10->86   10->86   10->86   10->86*  10->96
$LF       5-0#     10->86   10->86   10-?54   10-?54   10-?62   10-?62   10-?62
$LFLG    10-?56#   10-?56*
$LKCSB    6-0#     10-779*
$LKCSR    6-0#     10-774   10-780*
$LKS      6-0#     10-784   10-789*
$LLVEC    6-0#     10-785
$LPADR    5-0#
$LPERR    5-0#      9-40*   10-240*  10-267*  10-408*  10-651   10->86
$LPVEC    6-0#     10-775
$MADR1    5-0#
$MADR2    5-0#
$MADR3    5-0#
$MADR4    5-0#
$MAIL     4-286     4-286    5-0#      8-9      8-14     8-15     10->86   10-?54
$MAMS1    5-0#                                                     :
$MAMS2    5-0#
$MAMS3    5-0#
$MAMS4    5-0#
$MBADR    4-286#
$MFLG    10-?56    10-?56#  10-?56*  10-?56*
$MNEW    10-?62    10-?62#
$MSGAD    5-0#     10-?56   10-?56*
$MSGLG    5-0#     10-?56*
$MSGTY    5-0#     10-?56   10-?56   10-?56*  10-?56*
$MSWR    10-?62    10-?62#
$MTYP1    5-0#
$MTYP2    5-0#
$MTYP3    5-0#
$MTYP4    5-0#
$NULL     5-0#     10-?54   10-?54   10-?54
$NWTST    8-9#
$OCNT    10-?58#   10-?58*  10-?58*
$OMODE   10-?58    10-?58#  10-?58*  10-?58*  10-?58*  10-?58*
$PASS     5-0#      8-14*   10-568   10-568   10-568*  10-568*  10-568*  10-589*
$PASTM    4-286#
$POWER   10-?64    10-?65#
$PWRAD   10-?64#
$PWRDN    8-14     10-?64   10-?64#
$PWRMG   10-?64#
$PWRUP   10-?64    10-?64#
$QUES     5-0#      8-228    8-255    8-298    8-391    10->86   10->86   10-?54   10-?54   10-?62   10-?62   10-?62   10-?62
$R2A     10-?75
$RDCHR   10-?62#   10-?75   10-?75
$RDDEC   10-?75
$RDLIN   10-?62#   10-?75   10-?75
$RDOCT   10-?75
```

```
$RESRE  10-?73#  10-?75
$RM02    8-128   12-9#
$RM03    8-125   12-10#
$RM05    8-131   12-11#
$RMADR   6-0#     8-63     8-74*    8-75     15-5     17-17    17-35
$RMVEC   6-0#     8-64     8-70*    8-71     17-26
$RTNAD  10-568#
$SAVR6  10-?64   10-?64#  10-?64*  10-?64*  10-?64*
$SAVRE  10-?73#  10-?75   10-?75
$SB2D   10-856   10-862   10-?69#
$SETUP   4-277    4-277    4-277    4-277    4-277#   4-277#   4-277#   4-277#   4-277#   8-14     8-14     8-14     8-14     8-14
         8-14     8-14     8-14     8-14     8-14     8-14     8-14     8-15     8-15     8-15     10-568   10-568   10-->86  10-->86
        10-->86  10-->86  10-?62   10-?62   10-?62   10-?62   10-?62
$STUP    4-277    4-277    4-277    4-277    4-277#   4-277#   4-277#   4-277#   4-277#   4-277#   4-277#   4-277#   4-277#
$SUPRS  10-857   10-863   10-?67#
$SVPC    4-283    4-283#
$SWR     4-61#    4-72     4-73     4-73     4-73     4-73     4-73     4-73     4-73     4-73     5-0      5-0      5-0      8-9
         8-14    10-568   10-568   10-568   10-568   10-568   10-->86  10-->86  10-->86  10-->86  10-->86  10-->86  10-->86  10-->86
        10-->86  10-->86  10-->86  10-?64
$SWREG   5-0#     8-14
$TESTN   5-0#     8-9*    10-590*  10-714*  10-753*  10-984   10-998*
$TKB     5-0#    10-836   10-846   10-?54   10-?54   10-?62   10-?62   10-?62   10-?62   10-?62   10-?62   10-?62
$TKCNT  10-?62   10-?62   10-?62#  10-?62*  10-?62*  10-?62*
$TKINT   8-58     8-282    9-151    9-216   10-799   10-?62   10-?62   10-?62#
$TKQEN  10-?62   10-?62   10-?62#
$TKQIN  10-?62   10-?62   10-?62#  10-?62*  10-?62*  10-?62*  10-?62*
$TKQOU  10-?62   10-?62   10-?62#  10-?62*  10-?62*  10-?62*
$TKQSR  10-?62   10-?62   10-?62   10-?62#
$TKS     5-0#    10-847*  10-?54   10-?54   10-?62   10-?62   10-?62   10-?62   10-?62   10-?62*  10-?62*  10-?62*  10-?62*  10-?62*
        10-?62*
$TKSRV  10-?62   10-?62#
$TMPO    5-0#     8-218*   8-219    8-221    8-222    8-224    8-226    8-247*   8-248    8-250    8-251    8-253    8-286*   8-287
         8-289    8-291    8-292    8-377*   8-378    8-380    8-382    8-383    9-225*   9-226    9-228    9-230    9-232    9-239
        10-802*  10-803   10-805
$TN      4-62#    4-72     8-9      8-9      8-9      8-9#
$TNPWR  10-?71   10-?71   10-?71#
$TPB     5-0#    10-?54   10-?54   10-?54*
$TPFLG   5-0#    10-?54   10-?54   10-?54
$TPS     5-0#    10-?54   10-?54   10-?54
$TRAP    8-14    10-?75#
$TRAP2  10-?75   10-?75#
$TRP    10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75
        10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75
        10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75
        10-?75   10-?75   10-?75#  10-?75#  10-?75#  10-?75#  10-?75#  10-?75#  10-?75#  10-?75#  10-?75#  10-?75#  10-?75#  10-?75#
$TRPAD  10-?75   10-?75#
$TSTM    4-286#
$TSTNM   5-0#    10-->86  10-->86  10-->86
$TTYIN  10-?62   10-?62   10-?62   10-?62   10-?62   10-?62#
$TYPBN  10-?75
$TYPDS  10-?60#  10-?75   10-?75
$TYPE   10-?54#  10-?56   10-?75   10-?75
$TYPEC  10-?54   10-?54   10-?54   10-?54#  10-?62
$TYPEX  10-?54   10-?54   10-?54#
$TYPOC  10-?58#  10-?75   10-?75
$TYPON  10-?58   10-?58#  10-?75
```

AOE 4-157#

L 11

SEQ 0141

```
APASS     5-0       5-0
APRIOR    5-0
APTCSU   10-?54    10-?56#
APTENV   10->86    10-?54    10-?56    10-?56#
APTSIZ    8-14     10-?56#
APTSPO   10-?54    10-?56    10-?56#
ASWREG    5-0       5-0
AT0       4-177#
AT1       4-178#
AT2       4-179#
AT3       4-180#
AT4       4-181#
AT5       4-182#
AT6       4-183#
AT7       4-184#
ATA       4-144#
ATABIT    8-121     8-184     8-192     11-152#   11-327    11-424    11-530    11-928    11-950    11-958    11-959    11-979    11-:69
ATESTN    5-0       5-0
ATTN      6-0#     10->86*   15-1      15-2
AUNIT     5-0       5-0
AUSWR     5-0       5-0
AVECT1    4-275#    5-0       5-0
AVECT2    5-0       5-0
BAD16     6-0#      9-94      9-114     9-156*    9-157*    9-166     9-167*    9-168*    9-169*    9-170*    9-171*    9-172*    9-173*    9-174*
          9-176*    9-178*    9-180     9-180*    9-181*    9-182*    9-183*    9-184*    9-185*    9-188*    9-194     9-196     9-198     9-199
          9-200     9-201     9-202     9-203     9-207*    10-73     10-999    10-:12    10-;21    10-;24    10-<00    10-<47*   10-<48*   10-<49
         10-=40    10-=54
BAD18     6-0#      9-93      9-120     9-198*    9-203*    9-208*    10-76     10-:15    10-;41    10-;44    10-<09    10-<54*   10-<55*   10-<56
         10-=50    10-=56
BADENT   10-891    12-39#
BAI       4-104#
BEGCYL    6-0       6-0#      8-158*    8-274*    8-329*    8-353     10-671    10-716    10-985    10-991*   10-997*
BEGIN     4-281     8-3#
BEGIN2    8-5       8-9#      10-597    10-604    10-610    10-809    10-929    10-?64
BEGTRK    6-0       6-0#      8-157*    8-273*    8-331*    8-358     10-475    10-670    10-685    10-718    10-737    10-986    10-992*   10-996*
BIT0      4-78#
BIT00     4-78      4-78#
BIT01     4-78      4-78#     10-655    11-404    11-659
BIT02     4-78      4-78#
BIT03     4-78      4-78#     11-886    11-;85
BIT04     4-78      4-78#     10-635    11-921
BIT05     4-78      4-78#     11-901
BIT06     4-78      4-78#     11-337    11-395    11-469    11-782    11-<45
BIT07     4-78      4-78#     11-337    11-604    11-759    11-886    11-901    11-921    11-954    11-<17
BIT08     4-78      4-78#     10-632    11-337
BIT09     4-78      4-78#     10-632    10->86    11-:60
BIT1      4-78#    10-347     10-623
BIT10     4-78#    10-629     10-655    10->86    11-661
BIT11     4-78#    10-629     11-290    11-431    11-625    11-995
BIT12     4-78#    10-626     10-655    11-284    11-318    11-337    11-406    11-439    11-619    11-642    11-657    11-671    11-911    11-913
          11-;25    11-;80    11-<46
BIT13     4-78#    10-623     10-655    10->86    11-389    11-860    11-;28
BIT14     4-78#    10-352     10-356    10-358    10-623    10-655    11-401    11-436    11-847    11-990    11-:90    11-;35    11-;39    11-<43
BIT15     4-78#     9-167      9-185    10-163    10-180    10-184    10-203    10-218    10-296    10-383    11-389    11-401    11-404    11-406
          11-436    11-439    11-625    11-659    11-661    11-782    11-886    11-901    11-911    11-921    11-990    11-:60    11-:90    11-:97
BIT2      4-78#    10-623     11-:97
```

```
DT06      4-197#
DT07      4-198#
DT08      4-199#
DT1       7-6       15-1#
DT10      7-55      7-62      7-69      7-76      7-83      7-90      7-97      7-118     7-125     15-7#
DT17      7-104     15-10#
DT2       7-13      15-2#
DT20      7-111     15-11#
DT23      7-132     15-15#
DT24      7-139     15-16#
DT25      7-146     15-21#
DT3       7-20      15-3#
DT4       7-27      15-4#
DT6       7-41      15-5#
DT7       7-48      15-6#
DTE       4-160#    10-289
DTSY      4-173#
DTUW      11-146#   11-230    11-452    11-508*   11-639*   11-652    11-666    11-668*   11-677*   11-741    11-753*   11-:56    11-:66*   11-;33
DULP      11-279#   11-291
DVA       4-131#
ECH       4-154#
ECI       4-223#
EDIT      6-0#      8-161*    9-8*      9-18*     10-568    10-;96*   10-<85*
EM1       7-4       13-3#
EM10      7-53      13-10#
EM11      7-60      13-11#
EM12      7-67      13-12#
EM13      7-74      13-13#
EM14      7-81      13-14#
EM15      7-88      13-15#
EM16      7-95      13-16#
EM17      7-102     13-17#
EM2       7-11      13-4#
EM20      7-109     13-18#
EM21      7-116     13-19#
EM22      7-123     13-20#
EM23      7-130     13-21#
EM24      7-137     13-22#
EM25      7-144     13-23#
EM3       7-18      13-5#
EM4       7-25      13-6#
EM5       13-7#
EM6       7-39      13-8#
EM7       7-46      13-9#
EMPTYQ    11-631    11-682    11-783    11-:98    11-=38#
EMTVEC    4-78#     8-14*     8-14*
ENDCYL    6-0       6-0#      8-159*    8-330*    8-353     10-479    10-483    10-689    10-693
ENDTRK    6-0       6-0#      8-205*    8-337*    8-358     10-470    10-680
ENTADR    8-349     12-25#
ERINDX    10-249    10-280    10-417    10-568    10-620#
ERR       4-143#    10-300    10-424
ERROR     4-78#
ERRVEC    4-78#     8-14      8-14*     8-14*     10-772*   10-773*   10-783*   10-792*
ES.SAV    11-817    11-868    11-870    11-;32    11-;91    11->09#
F         8-237     12-17#
F1        4-126#
```

MSIZE 8-245 12-36#

K 12

SEQ 0153

```
MSQVER  10-389   12-49#
MSTCK    4-170#
MSVER   10-391   12-50#
MUNIT    8-284   12-13#
MUSDR    8-317   12-29#
MUTLTY   9-223   12-73#
MVCMPT  10-568   12-53#
MVERFY   8-233   12-35#
MWC      6-0#     8-170*   8-175*   8-262*   8-269*  10-149   10-401   10-667
MWR      4-172#
MXDLTA  11-176#  11-725
MXF      4-110#
MXLACT  11-173#  11-708
MXWNDW  11-182#
NBA      4-203#
NED      4-113#
NEM      4-112#
NEXT     6-0#    10-144*  10-154   10-159   10-179*  10-202*  10-205   10-217*  10-222*  10-310   10-312   10-343
NOTPRS   8-104   12-6#
NOTRM    8-101   12-5#
NOTSAF   8-110   12-7#
NUMERR  10-568   12-54#
O.ENTR   9-5     10-8     10-836#  10-865
OFDIR    4-221#
OFFSET   4-259#
OPI      4-161#  10-289
OPT     11-387   11-422#  11-777   11-963   11-:01
OR       4-108#
PACK     6-0#     9-25*    9-85*    9-111    9-146   10-69    10-97*
PAR      4-151#
PAR1     6-0      6-0#
PAR2     6-0      6-0#
PAR3     6-0      6-0#
PAR4     6-0      6-0#
PAR5     6-0      6-0      6-0      6-0#
PAR6     6-0      6-0      6-0      6-0#
PAR7     6-0      6-0      6-0#
PAR8     6-0      6-0#
PARENT   8-350   10-874#  10-;11   10-;31   10-;63   10-;82   10-=19
PAT      4-105#
PATA     6-0#     8-180*   8-371*  10-16*   10-20*   10-24*   10-28    10-376   10-381*  10-383*  10-384   10-585*  10-745
PATB     6-0#     8-181*   8-372*  10-17*   10-21*   10-25*   10-29    10-384*  10-586*  10-746
PATSEL   6-0#     8-182*   8-276*   8-395*  10-18    10-22
PGE      4-111#
PGM      4-138#
PIP      4-142#
PIRQ     4-78#
PIRQVE   4-78#
POPQUE  11-435   11-487   11-603   11-924   11-953   11-993   11-=89#
PR0      4-78#
PR1      4-78#
PR2      4-78#
PR3      4-78#
PR4      4-78#
PR5      4-78#
PR6      4-78#   10-778   10-788
```

PR7 4-78# 8-87

M 12

SEQ 0155

```
PS        4-78     4-78#    8-87*    8-214*   8-313*   9-6*     10-9*    10-851*  11-222   11-223*  11-252*  11-260*  11-359   11-360*
         11-413*  11-423   11-472*  11-714*
PSEL      4-88#
PSW       4-78#
PWRVEC    4-78#    8-14*    8-14*    10-?64*  10-?64*  10-?64*  10-?64*  10-?64*  10-?64*
QCNT     11-<55#  11-=14   11-=38*  11-=53   11-=55*  11-=75   11-=89*
QDRV0    11-<66   11-<77   11-<86   11-<98#
QDRV1    11-<67   11-<78   11-<87   11-<99#
QDRV2    11-<68   11-<79   11-<88   11-=00#
QDRV3    11-<69   11-<80   11-<89   11-=01#
QDRV4    11-<70   11-<81   11-<90   11-=02#
QDRV5    11-<71   11-<82   11-<91   11-=03#
QDRV6    11-<72   11-<83   11-<92   11-=04#
QDRV7    11-<73   11-<84   11-<93   11-=05#
QINPT    11-<66#  11-=40   11-=57*  11-=58*  11-=59   11-=61*
QOUTPT   11-<77#  11-=40*  11-=78   11-=91   11-=92*  11-=93*  11-=94   11-=96*
QSTART   11-<86#  11-=20   11-=25   11-=61   11-=96
QSTOP    11-<87#  11-=59   11-=94
QTERM    11-<94   11-=06#
R6        4-78#    8-14     8-14*    8-14*
R7        4-78#
RBUF      6-0     10-441*  10-442*  10-443   10-443   10-450*  10-451*  10-452   10-452   10-:50*  10-:56*  10-:57*  10-:58*  10-:62
         10-:66*  10-:69*  15-6     15-6     15-15    15-15    18-5#
RD.ADR   11-;17*  11-;18*  11-;20#  15-3
RD.RM    11-292   11-322   11-328   11-551   11-575   11-589   11-701   11-715   11-761   11-811   11-878   11-888   11-903   11-:45
         11-;15#  11-;68   11-;82   11-;88   11-<21
RD.RM1   11-;19#  11-;47
RD.RM2   11-;15*  11-;46#
RD.RM3   11-;26   11-;38   11-;48#
RD.RM4   11-;31   11-;50#
RD.WRD   11-;21#  11-;22   15-3     15-4
RDCHR     8-217    8-246    8-285    8-376    9-224    10-801   10-?62   10-?75#
RDDAT     4-271#
RDHD      4-272#   9-42     10-400
RDLIN     9-158    10-886   10-<89   10-?75#  17-22    17-31
RDY       4-85#
READIN    4-261#
RECAL     4-256#
RECORD   10-295    10-908#
REJCT2   10-599#   10-:54
REJCT3   10-568    10-606#
REJECT    9-82     10-593#
RELSE     4-258#
RESET    10-297    10-360   10-:50#
RESREG   10-?51    10-?71   10-?75#  11-261   11-407   11-410   11-473   11-686   11-746   11-:24   11-<30   11-=29
RESTRT   10-96     10-797#
RESVEC    4-78#
RETBAD    9-24#
RETRY     6-0#     9-24*    9-59     9-61*    9-65*    10-147*  10-257   10-259*  10-262*  10-336   10-338*  10-341*  10-568   10-568*
         10-568*  10-568*
RM.REG    6-0      6-0#     10-274   10-286   10-288   10-300   10-424   10-426   10-428   10-430   10-568   10-657   10->86   10->86
          15-1     15-1     15-1     15-1     15-1     15-1     15-3     15-3     15-3     15-4     15-4     15-4     15-7     15-7
          15-7     15-7     15-7     15-8     15-8     15-8     15-8     15-8     15-8     15-8     15-9     15-9     15-9     15-9
          15-9     15-12    15-12    15-12    15-12    15-12    15-12    15-12    15-12    15-13    15-13    15-13    15-13    15-13
          15-13    15-13    15-13    15-14    15-14    15-16    15-17    15-17    15-17    15-17    15-17    15-17    15-17    15-17
          15-18    15-18    15-18    15-18    15-18    15-18    15-18    15-18    15-19    15-19
```

RM05 9-34 9-43 10-46 10-242 10-268 10-409 10-568 B 13 10-568 10-:70 11-359#

SEQ 0157

```
RMADR    8-63*   8-75*   11-169#  11-241   11-366   11-490   11-510   11-532   11-699   11-740   11-:43   11-;17   11-;23   11-;27
         11-;41  11-;74   11-;78   11-<10   11-<40   17-36
RMAS     11-193# 11-327*  11-809   11-928*  11-958*  11-979*  11-:68
RMBA     10->86  10->86   11-188#
RMCS1    11-186# 11-283*  11-290   11-316   11-430*  11-431   11-506   11-528   11-570   11-590   11-601   11-616   11-629   11-762
         11-812  11-865   11-895   11-934   11-995   11-;69   15-2
RMCS2    10-286  11-190#  11-242*  11-282*  11-284   11-379*  11-429*  11-491*  11-511*  11-533*  11-619   11-642   11-657   11-671
         11-680* 11-700*  11-760*  11-876*  11-965*  11-978*  11-:44*  11-;24   11-;79   11-<11*  11-<17   11-<41*  11-<46
RMDA     10-274  10-568   11-189#  11-498   11-524   11-539
RMDB     11-195# 11-<15
RMDC     11-200# 11-502   11-514   11-546
RMDS     10-300  11-191#  11-323   11-702   11-879   11-904   11-966   11-:46
RMDT     11-197# 11-293
RMEC1    11-204#
RMEC2    11-205# 11-<25
RMER1    10-288  10-657   11-192#  11-329   11-889   11-967   11-;83
RMER2    11-203# 11-968
RMERRS   11-22#  11-225   11-847   11-850   11-860   11-966*  11-967*  11-968*  11-969*  15-2     15-2     15-2
RMHR     11-201#
RMINIT    8-81    8-310   11-221#
RMLA     11-194# 11-716   11-718
RMMR1    11-196#
RMMR2    11-202# 11-969
RMOF     11-199# 11-320   11-552   11-556   11-576   11-580
RMR       4-150#
RMSN     11-198#
RMTMR    10-818  11-:09#
RMVEC     8-64*   8-71*   11-170#  11-238   11-240   11-360
RMWC     11-187#
RNOP      4-254#
RTC       4-260#
SAVEFG    8-82*   8-311*  11-124#  11-605   11-766   11-955
SAVREG   10->94  10-?71   10-?75#  11-221   11-362   11-422   11-641   11-739   11-:12   11-<07   11-=13
SAVSEC    6-0     6-0      6-0      6-0#     9-243*  10-274*  10-276*  10-277*  10-278*  10-305   10-306   10-359*  10-568   10-568
         10-568* 10-568*  10-923   10-:57   10-:63   10-:92   10-;00   10-;10*  10-;18   10-;30*  10-;38   10-;62*  10-;69   10-;81*
         10-;89  10-=18*  10-=26   10-=28   10-=31   10-=33   10-=38*  10-=41*  10-=43   10-=48*  15-10    15-11    15-16    15-21
SAVWC     6-0#   10-320*  10-703
SC       11-745  11-778   11-781   11-787   11-809#
SC01      4-207#
SC02      4-208#
SC04      4-209#
SC10      4-210#
SC11     11-877  11-885   11-946#
SC12     11-843  11-854   11-965#
SC13     11-833  11-836   11-975#
SC20      4-211#
SC3      11-824# 11-828
SC4      11-826# 11-857   11-869   11-871   11-920   11-936   11-945   11-964   11-:02
SC5      11-825  11-831#
SC6      11-853  11-872#
SC6A     11-851  11-897#
SC7      11-887  11-900   11-902   11-912   11-922#
SC8      11-866  11-880   11-890   11-896   11-905   11-935   11-937#  11-974
SCAWC    10-332  10-703#
SCOPE     4-78#
SEARCH    4-263#
```

SEC30 6-0# 8-166* 8-176* 8-260* 8-267* 9-126 10-42 D 13 10-74 10-720 10-988 10-990* 10-994* 10-:20 10-:26
SEQ 0159

```
TYPOS    8-38    8-49    8-92    9-16    10-392  10-?75#
UF       4-249#  10-440  10-717
ULDFLG   11-105# 11-281* 11-369  11-384* 11-567* 11-633* 11-670* 11-839  11-925  11-927* 11-946  11-948* 11-:64*
UNS      4-162#
UNSTAT   8-90    12-12#
UNTOFF   8-107   12-3#
UNTON    8-120   12-4#
UPDCYL   10-485  10-695  10-734#
UPDTRK   10-472  10-682  10-759#
UPE      4-114#
US1      4-101#
US2      4-102#
US4      4-103#
USSAV    6-0#    9-92    9-129   9-200*  9-202*  9-206*  10-:24  10-;49  10-;50  10-;55  10-;56  10-<22  10-<26  10-<75*
         10-<76* 10-<77
USTAB    6-0#    9-95    9-135   9-199*  9-201*  9-209*  10-72   10-568  10-908  10-918  10-:18  10-;47  10-;48  10-;53
         10-;54  10-<21  10-<25  10-<68* 10-<69* 10-<70
VV       4-135#
WC       6-0#    8-169*  8-174*  8-261*  8-268*  10-26
WCE      4-115#  10-286
WCF      4-153#
WCHKX    11-769  11-790
WCKD     4-267#
WCKHD    4-268#  10-266
WCTBL    6-0#    10-208  10-211  10-226  10-229  10-315  10-318  10-568
WLE      4-159#  10-657
WRAP     6-0#    8-160*  8-275*  8-352*  8-363*  10-476  10-686  10-739
WRL      4-140#
WRT.AD   11-;64* 11-;74* 11-;77# 15-4
WRT.R1   11-;75# 11-;95
WRT.R2   11-;61* 11-;94#
WRT.R3   11-;70  11-;81  11-;84  11-;90  11-;96#
WRT.R4   11-;86  11-;98#
WRT.R5   11-;97  11-;99#
WRT.RM   11-315  11-319  11-497  11-501  11-505  11-513  11-523  11-527  11-538  11-545  11-555  11-569  11-579  11-600
         11-615  11-628  11-864  11-894  11-933  11-;61#
WRT.WD   11-;62* 11-;66  11-;73* 11-;76# 15-4
WRTDAT   4-269#
WRTHD    4-270#  10-239  10-568  10-:61
WRTRK    10-38#
WRTRK1   10-146#
WRTRK2   10-239# 10-323
WRTRKA   10-154# 10-345
WRTRKB   10-168  10-183# 10-186
WRTRKC   10-181  10-205#
WRTRKD   10-162  10-176  10-195  10-200  10-222#
WRTRKE   10-215  10-235#
WRTRKF   10-175  10-199  10-217#
WRTRKH   10-220  10-225#
WRTRKX   10-54#  10-349  10-368
WRTRKY   10-69#
WRTRKZ   10-58   10-365#
XSIZE    8-145#
XXDP     6-0#    8-26*   8-29*   8-30    8-32*   8-37    8-48    8-113   8-115   8-195   8-197   8-302   8-304
Y        8-258   10-811  12-19#
ZERO     8-295   12-14#
```

```
$$CMRE    4-481#
$$CMTM    4-481#   5-0
$$ESCA    4-78#
$$NEWT    4-78#    8-9
$$SET    10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75#
$$SETM    8-14     8-14#
$$SKIP    4-78#
.$ACT1    4-68#    4-283
.$APTB    4-67#    5-0      5-0#
.$APTH    4-67#    4-286
.$APTY    4-67#   10-?56
.$CATC    4-64#    4-279
.$CMTA    4-65#    4-481
.$DB2D    4-65#   10-?71
.$EOP     4-65#   10-568
.$ERRO    4-65#   10->86
.$ERRT    4-65#
.$POWE    4-66#   10-?64
.$READ    4-65#   10-?62
.$SAVE    4-64#   10-?73
.$SB2D    4-64#   10-?69
.$SUPR    4-64#   10-?67
.$TRAP    4-64#   10-?75
.$TYPD    4-65#   10-?60
.$TYPE    4-65#   10-?54
.$TYPO    4-65#   10-?58
.EQUAT    4-64#    4-78
.HEADE    4-64#    4-72
.SETUP    4-64#    4-277
.SWRHI    4-64#    4-73
.SWRLO    4-73#    4-74     4-75     4-76
CKCHR     4-16#    8-293    8-384    10->45   10->53
CKDIG     4-26#   10-888
CKNUM     4-39#
COMMEN    4-78#
ENDCOM    4-78#
ENDTYP   10-488#  10-568
ERRCAL   10-?77#  11-817   11-868   11-870   11-;32   11-;91
ERRENT   10->74#  10->86
ERROR     4-8#    10-250   10-251   10-252   10-253   10-254   10-255   10-281   10-282   10-283   10-284   10-285   10-291   10-298
         10-302   10-304   10-418   10-419   10-420   10-421   10-422   10-432   10-434   10-445   10-454   10-568   10-568   10-568
         10-568   10-568   10-568   10-601
ESCAPE    4-78#   10-248   10-279   10-294   10-416
GETPRI    4-78#
GETSWR    4-68#    4-78#    8-15     8-15#
LOOP      4-52#   10-267
MORETA   4-289#   5-0
MULT      4-78#
NEWTST    4-78#    8-9
POP       4-78#   10-752   10-767   10-?56   10-?56   10-?60   10-?64   10-?64   10-?73
PUSH      4-78#   10-734   10-759   10-?56   10-?56   10-?56   10-?60   10-?64   10-?64   10-?73
REPORT    4-78#
SETPRI    4-78#   10-?62
SETTRA   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75   10-?75#
SETUP     4-78#    8-14
SKIP      4-78#
```