RK11/RK05

PERFORMANCE EXERCISER

MD-11-DZRKH-E

EP-DZRKH-E-DL-A

FICHE 1 OF 1

NOV 1976

digital

MADE IN USA

.REM %

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZRKH-E
PRODUCT NAME: RK11/RK05 PERFORMANCE EXERCISER
DATE CREATED: JAN 10,1975
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: JIM KAPADIA
REVISIONS: TOM SAWYER, GEORGE GALLANT

TABLE OF CONTENTS

1.0 ABSTRACT

THE RK11/RK05 PERFORMANCE EXERCISER IS A HIGH LEVEL EXERCISER PROGRAM AIMED AT SIMULATING A RK11/RK05 SYSTEM ENVIRONMENT AND CHECKING FOR ERRORS THAT ARISE IN SUCH AN ENVIRONMENT (INTERACTION,, POLLING, ETC). IT ALSO PROVIDES A MEANS OF EVALUATING A SYSTEM THROUGH ITS ERROR LOGGING AND DATA-TRANSFER LOGGING FACILITIES.

AT THE BEGINNING OF THE PROGRAM THERE IS A SERIES OF TESTS SPECIFICALLY AIMED AT DETECTING AND ANALYZING FAILURES ASSOCIATED WITH BOUNDARY CONDITION TRANSFERS.

THE LATTER PART (AND THE MORE SIGNIFICANT ONE) CONSISTS OF THE EXERCISER.

2.0 REQUIREMENTS

2.1 EQUIPMENT

A. PDP11 WITH CONSOLE TELTYPE
B. 8K OF MEMORY - 12K FOR CHAIN MODE
C. RK11 CONTROLLER
D. RK05 DRIVES WITH PLATTERS

2.2 PRELIMINARY PROGRAMS

SINCE THIS IS A HIGH-LEVEL EXERCISER PROGRAM THE CONTROLLER AND THE DRIVE SHOULD BE FREE OF BASIC FAULTS. IT IS POSSIABLE TO HANG THE PROGRAM IF THE HEAD POSITIONING LOGIC IS FAULTY ON DUAL DENSITY DRIVES. THUS THE FOLLOWING PROGRAMS SHOULD BE RUN BEFORE ATTEMPTING TO USE THIS PROGRAM.

A. RK11 BASIC LOGIC TESTS (I AND II)
B. RK11/RK05 DYNAMIC TESTS
C. RK05 UTILITY PACKAGE (IF NEEDED)

2.3 EXECUTION TIME

THIS VARIES FROM 30 TO 90 MINUTES FOR A PASS. IT SHOULD BE NOTED THAT THIS IS AN EXERCISER LEVEL PROGRAM AND SHOULD BE PREFERRABLY RUN FOR A LONG PERIOD OF TIME.

3.0 STARTING ADDRESS

200 - ALL SWITCHES DOWN. SEE SEC. 6.0 FOR SWITCHES.

210 - RESTART ADDRESS. THE RESTART ADDRESS PROVIDES THE USER WITH AN ABILITY TO GO STRAIGHT TO EXERCISER PART OF THE PROGRAM (SKIPPING TESTS 1-7). THERE IS A SWITCH OPTION (SW 4) WHICH ALLOWS THE USER TO INHIBIT THE REWRITE OF RANDOM PATTERNS ON ALL DRIVES, ON RESTART. SEE SEC- 6.9.

4.0 PROGRAM CONTROL MODES AND OPERATOR ACTION

PAPER TAPE LOADING
RKDP DUMP MODE
RKDP CHAIN MODE
ACT11

4.1 PAPER TAPE LOADING

4.1.1 LOAD PROGRAM INTO MEMORY USING STANDARD PROCEDURE FOR ABSOLUTE TAPES.

4.1.2 MAKE SURE THAT THE DRIVES TO BE CHECKED ARE LOADED WITH DISKS AND ARE IN 'RUN'. 'WRT ENABLE' THEM. CHECK THAT 'WRT PROT' LIGHT ON THESE DRIVES IS OFF. PUT DRIVES THAT ARE NOT TO BE TESTED ON 'LOAD'.

4.1.3 LOAD ADDRESS 200

4.1.4 SET SWITCHES IF DESIRED (SEE SEC 6.0) AND PRESS START

4.1.5 THE PROGRAM IDENTIFIES ITSELF

MAINDEC-11-DZRKH-E RK11/RK05 PERFORMANCE EXERCISER

THEN IT PROCEEDS TO TEST THE DRIVES.

4.2 RKDP DUMP MODE

4.2.1 THE PROGRAM IS LOADED BY THE RKDP MONITOR.

4.2.2 SET SA=200. SELECT ANY SWITCHES YOU WANT AND PRESS START.

4.2.3 THE PROGRAM IDENTIFIES ITSELF AND PRINTS OUT:

REPLACE DR0 RKDP-PAK AND TYPE CR WHEN DONE IF NOT DR0 ON LOAD

IN RESPONSE TO THIS, REPLACE THE RKDP PAK ON DRIVE 0. IF YOU WANT DRIVE 0 TESTED. THEN TYPE CARRIAGE RETURN. IF YOU DO NOT WANT TO TEST DRIVE 0 (OR REPLACE RKDP PAK), PUT DRIVE 0 ON 'LOAD', 'WRITE PROTECT' AND THEN TYPE CARRIAGE RETURN.

4.3 RKDP CHAIN MODE

THE PROGRAM IS CHAIN LOADED FROM RKDP PAK ON DRIVE 0. AFTER IDENTIFYING ITSELF, THE FOLLOWING MESSAGE APPEARS:

DR0 NOT TESTED.

DRIVE 0 WILL NOT BE TESTED SINCE THE RKDP PAK IS ON THAT DRIVE.

```
                                        4.4   ACT11 MODE
```

THE PROGRAM IS LOADED BY THE ACT11 MONITOR. AFTER IDENTIFYING ITSELF, ASCERTAINS THE NUMBER OF DRIVES PRESENT AND PROCEEDS TO TEST EACH OF THEM AS BEFORE.

5.0 DRIVE SELECTION

PUT ALL THE DRIVES THAT ARE TO BE EXERCISED AND TESTED ON 'RUN', WRITE ENABLE THEM. MAKE SURE THAT THE 'WRT PROT' IS OFF. THE PROGRAM RECOGNIZES THAT THESE DRIVES ARE ON LINE AND PROCEEDS TO TEST THEM. DUAL DENSITY DRIVES WILL HAVE THE LETTER F TYPED AFTER THE DRIVE NUMBER.

IF A FATAL ERROR OCCURS ON A DRIVE WHILE THE PROGRAM IS RUNNING THE DRIVE IS AUTOMATICALLY DESELECTED ('DSELCT') AND DROPPED FROM THE DRIVE SELECTION LIST.

6.0 SWITCH OPTIONS

IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I.E. AN 11/34) THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH REGISTER IS NOT PRESENT AND WILL USE A 'SOFTWARE' SWITCH REGISTER. THE 'SOFTWARE' SWITCH REGISTER IS LOCATED AT LOCATION 176 (8). THE SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD ROUTINE WHICH IS CALLED BY TYPING A 'CONTROL G'. THE PROGRAM WILL RECOGNIZE THE 'CONTROL G' WHENEVER THE PROGRAM ENTERS THE SCOPE ROUTINE OR BEGINS A NEW TEST. 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.

SW<15>=1	HALT ON ERROR
SW<13>=1	INHIBIT ERROR PRINTOUTS
SW<12>=1	TYPE OUT THE ERROR HISTORY
SW<11>=1	DUMP OUT ALL RK11 REGISTERS
SW<10>=1	RING BELL ON ERROR
SW<09>=1	LOOP ON SPECIFIC ERROR
SW<08>=1	DUMP OUT TRANSFER DATA AND ERROR STATISTICS
SW<06>=1	SELECT BUS ADDRESS LIMITS FOR DATA TRANSFERS
SW<05>=1	HALT BEFORE DOING THE NEXT SET OF COMMANDS
SW<04>=1	DO NOT REWRITE THE DISKS ON 210 RETSART
SW<03>=1	TYPE OUT ELAPSED TIME AT ERROR
SW<02>=1	DROP DRIVE AFTER MAXIMUM ERRORS

```
  259                                          SW(01)=1    TYPE SERIAL NUMBER OF ERRORING DRIVE
```

6.1 SW<15>

THE PROGRAM HALTS ON ENCOUNTERING AN ERROR, AFTER TYPING OUT THE ERROR MESSAGE AND PERTINENT INFORMATION. PRESSING "CONTINUE" RESTORES NORMAL OPERATION OF THE PROGRAM.

6.2 SW<13>

THIS SWITCH INHIBITS ALL ERROR MESSAGES. NORMALLY USED WHEN LOOPING ON ERROR (SW 9).

6.3 SW<12>

IF THIS SWITCH IS SET WHEN AN ERROR OCCURS, INFORMATION ABOUT THE HISTORY OF THAT ERROR IS TYPED OUT.

THE FUNCTION THAT WAS BEING PERFORMED ON THE RK11 IS TYPED OUT. THE FUNCTION COULD BE EITHER A READ, WRITE, WRITE CHECK, READ CHECK. BESIDES THESE NORMAL FUNCTIONS, IT COULD BE A CONTROL RESET, DRIVE RESET OR POSITIONING OF THE HEADS (SEEKING). FOR THE FOUR FUNCTIONS THE INITIAL DISK ADDRESS, BUS ADDRESS AND WORD COUNT (2'S COMPLEMENT) ARE ALSO GIVEN. FOR DRIVE RESET AND POSITIONING THE DRIVE NUMBER OR WHICH THE OPERATION WAS BEING PERFORMED IS GIVEN.

SIMILAR INFORMATION IS TYPED OUT ABOUT THE FUNCTION THAT WAS DONE JUST BEFORE THE ONE GIVING THE ERROR.

6.4 SW<11>

IF THIS SWITCH IS SET WHEN AN ERROR OCCURS, THE CONTENTS OF ALL RK11 REGISTERS ARE TYPED OUT.

6.5 SW<09>

THIS SWITCH PROVIDES THE TIGHTEST POSSIBLE SCOPE LOOP. LOOPING IS DONE WHEN AN ERROR OCCURS. NOTE THAT THERE ARE TWO CLASSES OF ERRORS AND HENCE TWO CLASSES OF ERROR LOOPS, REFER TO SEC 8.0 FOR THE DIFFERNECE IN THE ERROR LOOPS PROVIDED BY SW 9.

6.6 SW<08>

WHEN THIS SWITCH IS SET, THE ERROR AND TRANSFER DATA STATISTICS WHICH HAVE BEEN COLLECTED UNTIL THAT TIME, ARE TYPED OUT.

THE TRANSFER DATA STATISTICS GIVE THE NUMBER OF WORDS WRITTEN AND READ ON EACH DRIVE THAT IS PRESENT. IT SHOULD BE NOTED THAT READ CHECK AND WRITE CHECK ARE CONSIDERED TO BE ESSENTIALLY READ OPERATIONS.

THE ERROR STATISTICS GIVE THE NUMBER OF ERRORS THAT HAVE OCCURRED (IF ANY) IN THE FOLLOWING CATEGORIES, ON THE DRIVES

316 THAT ARE PRESENT:

```
                                        CHECK SUM ERROR
                                        WRITE CHECK ERROR
                                        DATA COMPARISON ERROR
                                        HARD ERROR
                                        SEEK ERROR
                                        SEEK INCOMPLETE

                                    ABORTS - WHEN AN ERROR OCCURS THE FUNCTION IS RETRIED TWICE.  IF
                                    STILL  THE  ERROR PERSISTS THE FUNCTION IS ABORTED AND THE ABORT
                                    COUNT IS INCREMENTED FOR THAT DRIVE.

                            6.7     SW<06>

                                    THIS SWITCH ENABLES THE USER TO SELECT THE LIMITS OF THE  MEMORY
                                    BUS  ADDRESSES  BETWEEN  WHICH  THE DATA TRANSFERS WILL BE DONE.
                                    NORMALLY THE TRANSFERS ARE DONE BETWEEN THE LOWER LIMIT (BASEBA)
                                    AND  THE  HIGHER  LIMIT  (MAXBA).  THESE TWO LIMITS ARE NORMALLY
                                    SELECTED BY THE PROGRAM AND USE THE  MAXIMUM  AVAILABLE  MEMORY.
                                    IF  THE  USER WANTS TO DO DATA TRANSFERS BETWEEN SELECTED MEMORY
                                    ADDRESSES (EX:  BETWEEN 12K AND 16K) THEN THIS SWITCH SHOULD  BE
                                    SET  AT  THE  STARTING  OF  THE  PROGRAM.  THE FOLLOWING MESSAGE
                                    APPEARS:

                                    TYPE OCTAL BUS ADDRESS FOR DATA XFER, BETWEEN XXXXXX AND YYYYYY

                                        LO LIMIT?
                                        HI LIMIT?

                                    IN RESPONSE THE USER  SHOULD  TYPE  IN  ANY  TWO  BUS  ADDRESSES
                                    (OCTAL)  BETWEEN  XXXXXX  AND  YYYYYY.   IF  THE  USER  TYPES IN
                                    ANYTHING OUT OF THE X AND Y RANGE THE QUESTION IS ASKED AGAIN.

                                    THIS SWITCH COULD BE QUITE USEFUL  IN  DETERMINING  WHETHER  THE
                                    PROBLEM IS WITHIN THE RK11 OR OUTSIDE (IN MEMORY).  NORMALLY, IF
                                    THE PROBLEM IS WITHIN THE RK11, ERRORS  WILL  KEEP  ON  OCCURING
                                    REGARDLESS  OF  WHERE  IN  THE  MEMORY DATA TRANSFERS ARE TAKING
                                    PLACE.  ON THE OTHER HAND IF THE PROBLEM IS MEMORY RELATED,  THE
                                    ERRORS  WILL  TEND  TO  DISAPPEAR  FOR DATA TRANSFERS TO CERTAIN
                                    MEMORY BLOCKS AND WOULD REAPPEAR FOR OTHER ONES.

                            6.8     SW<05>

                                    THIS SWITCH PROVIDES THE USER A CAPABILITY TO HALT  THE  PROGRAM
                                    AT  A  KNOWN  POINT.   THE HALT IS DONE AFTER THE CURRENT SET OF
                                    EIGHT COMMANDS IN THE QUEUE HAVE BEEN EXECUTED.  THE  "HALT"  IS
                                    LOCATED  AT  THE BEGINNING OF THE'GENBRQ' ROUTINE, JUST BEFORE A
                                    SET OF 8 NEW COMMANDS IS GENERATED.  AFTER  THE  PROGRAM  HALTS,
                                    THE  EXECUTION  CAN  BE  RESUMED  BY  PRESSING  CONTINUE, OR THE
                                    PROGRAM CAN BE STARTED BACK AT 200 OR RESTARTED AT 210.

                            6.9     SW<04>

                                    THIS SWITCH PROVIDES THE USER WITH AN ABILITY TO SKIP  THE  TIME
```

```
373                                     CONSUMING REWRITE OF ALL THE DISKS WHEN THE PROGRAM IS RESTARTED
374                                     AT 210.  THIS SWITCH  CAN  BE  USED  ONLY  WHEN  RESTARTING  THE
```

PROGRAM AT 210 WITH SW 4 SET. ON RESTARTING THE PROGRAM AT 210,
THE INITIAL BOUNDARY CONDITION TESTS (TST1-TST7) ARE SKIPPED.
IF SWITCH 4 IS SET, THE REWRITE OF ALL THE DISKS (WHICH WOULD
HAVE BEEN NORMALLY DONE) IS ALSO SKIPPED. THE USER IS CAUTIONED
TO USE THIS SWITCH CAREFULLY.   THE DISKS SHOULD HAVE BEEN
WRITTEN WITH RANDOM PATTERNS AT LEAST ONCE BEFORE RESTARTING THE
PROGRAM AT 210.   IT SHOULD BE NOTED THAT TESTS 1-7 WRITE ON
CYLINDERS 0,1. ON RESTART, THE STATISTICS COLLECTED SO FAR ARE
SAVED.

6.10 SW<03>

THIS SWITCH ALLOWS THE TYPEOUT OF THE ELAPSED TIME AT WHICH
ERROR OCCURRED.   THE TIMING STARTS AT THE BEGINNING OF THE
EXERCISER PROGRAM. THIS SWITCH SHOULD NOT BE SET IF KW11L LINE
CLOCK IS NOT AVAILABLE ON THE SYSTEM.

7.0 EXERCISER PROGRAM

THE EXERCISER PROGRAM ATTEMPTS TO SIMULATE A DISK OPERATING SYSTEM ENVIRONMENT BY DOING RANDOM EVENTS (FUNCTIONS) USING RANDOMLY SELECTED PARAMETERS (DISK ADDRESS, BUS ADDRESS, WORD COUNT,ETC). AN ATTEMPT IS MADE TO DETECT INTER-ACTION PROBLEMS, OVERLAPPING SEEK PROBLEMS, ETC. FOR EXAMPLE, OVER 500 MILLION BITS ARE TRANSFERRED PER HOUR ON A TYPICAL RK11/RK05 SYSTEM (BASED ON 2 DRIVES, PDP11/50, 28K SYSTEM).

EIGHT JOBS OR COMMANDS ARE GENERATED AT A TIME (GEN8RQ) AND PUT IN A QUEUE TO BE PROCESSED. THE ALGORITHM WORKS AS FOLLOWS. COMMANDS IN THE QUEUE ARE PREPOSITIONED (HEADS) BY PREFORMING OVERLAPPING SEEKS. WHILE SOME OF THE DRIVES ARE BEING POSITIONED, THE LAST AVAILABLE (AND EXECUTABLE) COMMAND IS PERFORMED. THUS WHILE SOME DRIVES ARE BUSY POSITIONING THEIR HEADS, SOME DRIVE IS PERFORMING A FUNCTION (DATA TRANSFER,ETC). AS SOON AS THE CONTROLLER IS FREE , A CHECK IS MADE TO SEE IF THERE IS ANY DRIVE WHICH HAS ALREADY POSITIONED ITS HEAD. IF ONE IS FOUND THE COMMAND IS EXECUTED ON THAT DRIVE AND THE CONTROLLER AGAIN BECOMES BUSY. IF NO POSITIONED COMMAND IS FOUND, A CHECK IS MADE TO SEE IF THERE IS A COMMAND THAT IS TO BE POSITIONED. IF YES, IT IS POSITIONED AND THE LAST AVAILABLE COMMAND IS EXECUTED. IF IT IS FOUND THAT NO DRIVE NEEDS TO BE POSITIONED (THIS COULD HAPPEN IF THERE IS ONLY ONE COMMAND LEFT IN THE QUEUE OR THE REMAINING COMMANDS IN THE QUEUE ARE TO BE PERFORMED ON THE SAME DRIVE), THEN THE COMMANDS IS/ ARE EXECUTED.

THE ABOVE ALGORITHM HELPS SIMULATE A REAL ENVIRONMENT, AT THE SAME TIME MAXIMISING THE RATE OF DATA TRANSFERS. THE EXERCISER PROGRAM GIVES AN ELABORATE ERROR DETECTION CAPABILITY. THE STATE OF THE PROGRAM IS CONTINUOSLY TRACKED BY SOFTWARE KEYS, FLAGS, ETC. THESE FLAGS AND KEYS HAVE BEEN EXPLAINED IN DETAIL AT THE BEGINING OF THE LISTINGS, WHERE THEY ARE DEFINED. ON DUAL DENSITY DRIVES, ONLY ONE LOGICAL DRIVE IS SELECTED DURING EACH QUEUE BUILD. THIS INSURES THAT OVERLAPPED SEEKS WILL NOT INTEFER WITH THE HEAD POSITIONING LOGIC.

THE PARAMETERS USED FOR DOING THE COMMANDS ARE SELECTED RANDOMLY USING A RANDOM GENERATOR . THE FUNCTION TO BE PERFORMED IS SELECTED RANDOMLY FROM ONE OF THE FOUR: WRITE, READ, WRITE CHECK, OR READ CHECK. THE DRIVE NUMBER IS SELECTED FROM THE AVAILABLE DRIVES. THE DISK ADDRESS IS SELECTED OVER THE ENTIRE RANGE AND THE WORD COUNT AND BUS ADDRESS ARE SELECTED RANDOMLY IN SUCH A WAY THAT A NON-EXISTENT MEMORY ERROR OR OVERRUN CONDITION DOES NOT OCCUR.

RANDOM DATA BLOCKS ARE WRITTEN ON THE DISK. THE FIRST WORD OF EACH SECTOR BLOCK IS A NUMBER (2'S COMPLEMENT) INDICATING THE TOTAL NUMBER OF WORDS WRITTEN IN THAT SECTOR. THE REST OF THE WORDS IN THE BLOCK ARE GENERATED USING THE DISK ADDRESS (OF THAT SECTOR) AS THE RANDOM SEED NUMBER.

8.0 LOOPING CAPABILITIES:

SWITCH 9 GIVES LOOPING CAPABILITIES, ON ERROR. THERE ARE TWO CLASSES OF ERRORS:

A. ERRORS OCCURING IN THE NON-EXERCISER PART OF THE PROGRAM (ERROR NUMBERS UNDER 100 IN THE ERROR ITEMS TABLE)

B. ERRORS OCCURING IN THE EXERCISER PART OF THE PROGRAM (ERROR NUMBERS STARTING FROM 100 AND UP IN THE ERROR ITEMS TABLE)

C. NON-EXERCISER SCOPE LOOPS: IN THIS CASE, THE PROGRAM LOOPS ON A SPECIFIC ERROR GIVING A NARROW SCOPE LOOP. THIS SCOPE LOOP IS SIMILIAR TO THE ONE PROVIDED IN THE RK11 BASIC LOGIC TEST AND DYNAMIC TEST, WHICH THE USER MIGHT BE FAMILIAR WITH.

D. EXERCISER SCOPE LOOPS: WHEN AN ERROR OCCURS (AFTER TYPING OUT THE ERROR MESSAGE) CONTROL IS TRANSFERRED TO THE BEGINNING OF THE COMMAND-QUEUE. THE COMMANDS FROM THE FIRST COMMAND ONWARDS, ARE EXECUTED AGAIN TILL THE POINT OF ERROR. THIS LOOPING PROVIDES THE USER WITH A CAPABILITY TO RECREATE A SET OF EVENTS THAT LED TO THE ERROR.

9.0 TRANSFER DATA LOGGING

IN THIS PROGRAM, WHENEVER A DATA TRANSFER TAKES PLACE IT IS LOGGED WHETHER IT IS READ, READ CHECK, WRITE OR WRITE CHECK. SEPERATE COUNTS ARE KEPT FOR DATA TRANSFERS TAKING PLACE ON EACH DRIVE IN THE SYSTEM. AT any GIVEN TIME THE USER CAN GET THESE TRANSFER STATISTICS BY SETTING SWITCH 8 TO 1 (SEE SEC.6.6). THIS IS HELPFUL FOR EVALUATING A SYSTEM.

10.0 ERROR LOGGING

THROUGHOUT THE EXERCISER PROGRAM, WHEN AN ERROR OCCURS IT IS LOGGED. THE FOLLOWING CLASSES OF ERRORS ARE LOGGED FOR EACH DRIVE IN THE SYSTEM:

- CHECK SUM ERROR
- WRITE CHECK ERROR
- DATA COMPARISON ERROR
- HARD ERRORS
- SEEK ERROR
- SEEK INCOMPLETE ERROR
- ABORTS

THE ERROR STATISTICS CAN BE OBTAINED BY PUTTING SWITCH 8 TO 1. THE ERROR STATISTICS CAN BE USED IN CONJUNCTION WITH DATA TRANSFER STATISTICS TO GIVE AN IDEA OF THE SYSTEM PERFORMANCE (NUMBER OF WORDS TRANSFERRED PER ERROR, CSE FREQUENCY, RECOVERABLE VERSUS NON-RECOVERABLE ERRORS ETC.).

11.0 ERROR REPORTING AND RECOVERY

WHENEVER AN ERROR OCCURS IT IS REPORTED ALONG WITH RELEVANT INFORMATION. THE RK11 REGISTERS REPORTED IN THE ERROR MESSAGES REPRESENT THE CONTENTS AT THE TIME OF ERROR. EACH ERROR MESSAGE CONTAINS A 'PC' NUMBER, THIS IS THE PC LOCATION IN THE PROGRAM WHERE THE ERROR CALL IS LOCATED. THE USER IS ADVISED TO REFERENCE THIS LOCATION IN THE LISTINGS, IN CASE MORE INFORMATION ABOUT THE ERROR IS DESIRED.

SOME (SYSTEM) ERRORS REFER TO SOFTWARE FLAGS AND KEYS WHICH ARE USED TO MONITOR THE ONGOING ACTIVITIES ON THE SYSTEM. THESE FLAGS ARE EXPLAINED AT THE BEGINING OF THE LISTINGS AND SOULD BE REFERRED TO, IF THE NEED ARISES.

IF A FATAL ERROR CONDITION IS DETECTED (LIKE DRIVE UNSAFE, WRITE PROTECT SET, DRIVE READY CLEAR, ETC.) THE DRIVE IS REMOVED FROM THE DRIVE SELECTION TABLE AND DROPPED FROM FURTHER TESTING. A MESSAGE IS GIVEN INDICATING DROPPING OF THAT DRIVE. FOR FURTHER INFORMATION, REFER TO THE 'CHKDRV' AND 'DSELCT' ROUTINES IN THE LISTINGS.

RECOVERABLE ERRORS ARE RETRIED THREE TIMES. IF THE ERROR CONDITION FAILS TO CORRECT OR A IF A DIFFERENT ERROR OCCURS THE FUNCTION IS ABORTED. MESSAGES ARE PRINTED ONLY ONCE FOR EACH ERROR. AFTER EIGHT ABORTS ARE RECORDED ON A DRIVE THE DRIVE IS DROPPED. DUAL DENSITY DRIVES ARE ALWAYS DROPPED IN PAIRS.

12.0 SUBROUTINES AND HANDLERS

THERE ARE TWO WAYS IN WHICH MOST OF THE SUEROUTINES USED IN THIS PROGRAM ARE CALLED:

1. THROUGH THE NORMAL JSR CALL

```
JSR     REG,SUBROUTINE
```

2. THROGH THE 'TRAP' INSTRUCTION. THE TRAP INSTRUCTION WITH ITS LOWER BYTE ENCODED SERVES AS A CALL FOR SOME ROUTINES. WHEN THE 'TRAP' IS EXECUTED A TRAP OCCURS TO THE TRAP VECTOR AND THE TRAP DECODER IS ENTERED. THE TRAP DECODER ($TRAP) WILL PICK UP THE LOWER BYTE OF THE 'TRAP' INSTRUCTION AND USE IT TO INDEX THROUGH THE TRAP TABLE ($TRAPAD) FOR THE STARTING ADDRESS OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL GO TO THE DESIRED ROUTINE.

3. $SCOPE - THE SCOPE HANDLER

THE SCOPE HANDLER IS ENTERED THROUGH THE EXECUTION OF THE 'IOT' INSTRUCTION. IT KEEPS TRACK OF VARIOUS POINTERS, FLAGS AND DECIDES IF LOOPING IS TO BE DONE ON ERROR (SW 9). IT SHOULD BE NOTED THAT THIS HANDLER IS USED MOSTLY IN THE NON-EXERCISER PART OF THE PROGRAM.

4. $ERROR - ERROR HANDLER ROUTINE

THE ERROR HANDLER IS ENTERED THROUGH THE EXECUTION OF THE 'EMT' INSTRUCTION. THE LOWER BYTE OF THE EMT INSTRUCTION IS ENCODED TO GIVE AN IDENTIFIER TO THE ERROR CALL. THUS 'ERROR 1' IS 104001, ETC. THE ERROR ROUTINE DECIDES IF ANY ACTION IS TO BE TAKEN DEPENDING ON THE SWITCH SETTING (LIKE, HALT ON ERROR, INHIBIT ERROR TYPEOUT, ETC.).

MOST OF THE SUBROUTINES RESIDE IN THE LATTER PART OF THE PROGRAM. THE USER CAN REFER TO THEM TROUGH THE CROSS REFERENCE TABLE AT THE END OF THE LISTINGS OR TABLE OF CONTENTS AT THE BEGINING.
%

```
00160
        .TITLE  MAINDEC-11-DZRKH-E
        ;*COPYRIGHT (C) 1973
        ;*DIGITAL EQUIPMENT CORP.
        ;*MAYNARD, MASS. 01754
        ;*
        ;*PROGRAM BY JIM KAPADIA
        ;*
        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
        ;*PACKAGE (MAINDEC-11-DZQAC-CO),MAR 21, 1976.
        ;*
00180   ;*REVISED BY GEORGE GALLANT,TOM SAWYER   -  MARCH 1976
        .SBTTL  OPERATIONAL SWITCH SETTINGS
```

```
                                        ;*
                                        ;*      SWITCH          USE
                                        ;*      ------          ------------------
                                        ;*        15            HALT ON ERROR
                                        ;*        14            LOOP ON TEST
                                        ;*        12            TYPE OUT ERROR HISTORY
                                        ;*        10            BELL ON ERROR
                                        ;*         9            LOOP ON ERROR
                                        ;*         8            TYPE OUT ERROR AND TRANSFER DATA STATISTICS
                                        ;*         6            SELECT BUS ADDRESS LIMITS FOR DISK DATA TRANSFERS
                                        ;*         5            HALT BEFORE DOING NEXT SET OF COMMANDS(GENBRQ)
                                        ;*         4            DO NOT REWRITE THE DISKS ON RESTART AT 210
                                        ;*         3            TYPE OUT ELAPSED TIME AT ERROR
                                        ;*         2            DROP DRIVE AFTER MAXM ERORS ON THIS DRIVE
                                        ;*         1            TYPE SERIAL NUMBER OF ERRORING DRIVE
                                00260   ;*        11            DUMP OUT ALL RK11 REGISTERS ON ERROR
                                00270
                                00280
                                00290   ;*      YOU ARE ADVISED TO READ THE DOCUMENT FOR THIS PROGRAM.
```

```
                                00480
                                00490
                                        .SBTTL  BASIC DEFINITIONS

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

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

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

```
                        SW03=   10
                        SW02=   4
                        SW01=   2
                        SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

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

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

```
                                00510   KWLVEC=100              ;KW11L CLOCK VECTOR
                                00520
                                00530   .EQUIV  BIT15,ERR
                                00540   .EQUIV  BIT14,HE
                                00550   .EQUIV  BIT13,SCP
                                00560
                                00570
                                00580   .EQUIV  BIT12,DPL
                                00590   .EQUIV  BIT10,DRU
                                00600   .EQUIV  BIT09,SIN
                                00610   .EQUIV  BIT07,DRY
                                00620   .EQUIV  BIT06,RWS
                                00630   .EQUIV  BIT05,WPS
                                00640
                                00650
                                00660
                                00670   .EQUIV  BIT12,SKE
                                00680   .EQUIV  BIT01,CSE
                                00690   .EQUIV  BIT00,WCE
                                00700
                                        .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  003470                JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
                                00720
000210                                  .=210                   ;RESTART ADDRESS, IF RESTART IS
000210  105237  001753          00740   INCB    FRSTRT          ;DONE AT 210, THE BOUNDARY CONDITION
000214  000137  003470          00750   JMP     @#START         ;TESTS (TST1-7) ARE SKIPPED. IF SW 4
                                00760                           ;IS SET THEN THE DISKS ARE NOT REWRITTEN
                                00770                           ;(WRDSK) WITH RANDOM PATTERNS. NORAMALLY
                                00780                           ;ALL THE DISKS PRESENT ARE COMPLETELY
                                00790                           ;WRITTEN WITH RANDOM PATTERNS, AT THE
                                00800                           ;BEGINING OF THE
                                00810                           ;EXERCISER PART OF THE PRCGRAM.
                                00820
                                        .SBTTL  ACT11 HOOKS

                                ;;**************************************************************
                                ;HOOKS REQUIRED BY ACT11
000220                                  $SVPC=.                 ;SAVE PC
000046                                  .=46
000046  022376                        $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
000052                                  .=52
000052  000000                        .WORD   0               ;;2)SET LOC.52 TO ZERO
000220                                  .=$SVPC                 ;; RESTORE PC
                                00840
                                00850   ;KT11 REGISTER DEFINITIONS
                                        .SBTTL  MEMORY MANAGEMENT DEFINITIONS
```

```
                                        ;*KT11 VECTOR ADDRESS

000250                                  MMVEC=  250

                                        ;*KT11 STATUS REGISTER ADDRESSES

177572                                  SR0=    177572
177574                                  SR1=    177574
177576                                  SR2=    177576
172516                                  SR3=    172516

                                        ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS

172300                                  KIPDR0= 172300
172302                                  KIPDR1= 172302
172304                                  KIPDR2= 172304
172306                                  KIPDR3= 172306
172310                                  KIPDR4= 172310
172312                                  KIPDR5= 172312
172314                                  KIPDR6= 172314
172316                                  KIPDR7= 172316

                                        ;*KERNEL "D" PAGE DESCRIPTOR REGISTERS

172320                                  KDPDR0= 172320
172322                                  KDPDR1= 172322
172324                                  KDPDR2= 172324
172326                                  KDPDR3= 172326
172330                                  KDPDR4= 172330
172332                                  KDPDR5= 172332
172334                                  KDPDR6= 172334
172336                                  KDPDR7= 172336

                                        ;*KERNEL "I" PAGE ADDRESS REGISTERS

172340                                  KIPAR0= 172340
172342                                  KIPAR1= 172342
172344                                  KIPAR2= 172344
172346                                  KIPAR3= 172346
172350                                  KIPAR4= 172350
172352                                  KIPAR5= 172352
172354                                  KIPAR6= 172354
172356                                  KIPAR7= 172356

                                        ;*KERNEL "D" PAGE ADDRESS REGISTERS

172360                                  KDPAR0= 172360
172362                                  KDPAR1= 172362
172364                                  KDPAR2= 172364
172366                                  KDPAR3= 172366
172370                                  KDPAR4= 172370
172372                                  KDPAR5= 172372
172374                                  KDPAR6= 172374
172376                                  KDPAR7= 172376

                        00870
```

```
                                .SBTTL  COMMON TAGS

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

                        001100          .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000          $PASS:  .WORD   0               ;;CONTAINS PASS COUNT
001102  000             $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103  000             $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000          $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000          $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000          $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000          $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114  000             $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115  001             $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000          $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000          $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000          $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000          $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000          $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                  .WORD   0
001134  000             $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135  000             $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                  .WORD   0
001140  177570          SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570          DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560          $TKS:   177560                  ;;TTY KBD STATUS
001146  177562          $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564          $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566          $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154  000             $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155  002             $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156  012             $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157  000             $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  000000          $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
                                                        ;;WHICH  ($REG0) WAS OBTAINED
001162  000000          $REG0:  .WORD   0               ;;CONTAINS (($REGAD)+0)
001164  000000          $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
001166  000000          $REG2:  .WORD   0               ;;CONTAINS (($REGAD)+4)
001170  000000          $REG3:  .WORD   0               ;;CONTAINS (($REGAD)+6)
001172  000000          $REG4:  .WORD   0               ;;CONTAINS (($REGAD)+10)
001174  000000          $REG5:  .WORD   0               ;;CONTAINS (($REGAD)+12)
001176  000000          $REG6:  .WORD   0               ;;CONTAINS (($REGAD)+14)
001200  000000          $REG7:  .WORD   0               ;;CONTAINS (($REGAD)+16)
001202  000000          $REG10: .WORD   0               ;;CONTAINS (($REGAD)+20)
001204  000000          $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001206  177607  000377  $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001212  077             $QUES:  .ASCII  /?/             ;;QUESTION MARK
001213  015             $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001214  000012          $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;**********************************************************************
```

```
                                        .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


001216                                  $ERRTB:
                                00890   ;*THERE ARE TWO CLASSES OF ERRORS:
                                00900   ;*1. ERRORS IN EXERCISER PART OF THE PROGRAM - ERROR NUMBERS BELOW 100
                                00910   ;*2. ERRORS IN THE NON-EXERCISER PART OF THE PROGRAM - ERROR NUMBERS EQUAL
                                00920   ;*TO AND GREATER THAN 100.
                                00930   ;*THE DOCUMENT CONTAINS MORE INFORMATION ON THESE.
                                00940   ;*THE FOLLOWING ERRORS OCCUR IN THE EXERCISER PART OF THE PROGRAM.
                                00950
                                00960
                                00970   ;ITEM   1
                                00980
001216  027224                  00990           EM1     ;ERROR ON WRITE
001220  031322                  01000           DH1     ;PC     RKCS    RKER    RKDS    RKDA
001222  032014                  01010           DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
001224  000000                  01020           0
                                01030
                                01040   ;ITEM   2
                                01050
001226  027242                  01060           EM2     ;ATTEMPT TO INITIATE FUNCTION ON 'BUSY' DRIVE
001230  031370                  01070           DH2     ;PC     DRIVE
001232  032030                  01080           DT2     ;$ERRPC $REG0
001234  000000                  01090           0
                                01100
                                01110   ;ITEM   3
                                01120
001236  027315                  01130           EM3     ;CONTROL READY NOT SET
001240  031322                  01140           DH1     ;PC     RKCS    RKER    RKDS    RKDA
001242  032014                  01150           DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
001244  000000                  01160           0
                                01170
                                01180   ;ITEM   4
                                01190
001246  027340                  01200           EM4     ;R/W/S READY NOT SET
001250  031322                  01210           DH1     ;PC     RKCS    RKER    RKDS    RKDA
001252  032014                  01220           DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
001254  000000                  01230           0
                                01240
                                01250   ;ITEM   5
                                01260
001256  027363                  01270           EM5     ;CONTROL READY NOT SET AFTER FIRST INTERRUPT ON ISSUING SEEK
001260  031322                  01280           DH1     ;PC     RKCS    RKER    RKDS    RKDA
001262  032014                  01290           DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
```

```
  951  001264  000000                    01300            0
  952                                    01310
  953                                    01320
  954                                    01330    ;ITEM   6
  955                                    01340
  956  001266  027450                    01350            EM6     ;WRONG BITS IN RKCS, EXPECT SEEK
  957  001270  031322                    01360            DH1     ;PC     RKCS    RKER    RKDS    RKDA
  958  001272  032014                    01370            DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
  959  001274  000000                    01380            0
  960                                    01390
  961                                    01400    ;ITEM   7
  962                                    01410
  963  001276  027507                    01420            EM7     ;'BUSY' FLAG CLEAR ON INTERRUPTING DRIVE
  964  001300  031370                    01430            DH2     ;PC     DRIVE
  965  001302  032030                    01440            DT2     ;$ERRPC $REG0
  966  001304  000000                    01450            0
  967                                    01460
  968                                    01470    ;ITEM   10
  969                                    01480
  970  001306  027554                    01490            EM10    ;'POSITIONING' FLAG FOR INTERRUPTING DRIVE CLEAR
  971  001310  031370                    01500            DH2     ;PC     DRIVE
  972  001312  032030                    01510            DT2     ;$ERRPC $REG0
  973  001314  000000                    01520            0
  974                                    01530
  975                                    01540    ;ITEM   11
  976                                    01550
  977  001316  027631                    01560            EM11    ;'ERR'OR SET AFTER FIRST INTERRUPT ON ISSUING SEEK
  978  001320  031322                    01570            DH1     ;PC     RKCS    RKER    RKDS    RKDA
  979  001322  032014                    01580            DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
  980  001324  000000                    01590            0
  981                                    01600
  982                                    01610    ;ITEM   12
  983                                    01620
  984  001326  027711                    01630            EM12    ;SCP SET AFTER FIRST INTERRUPT ON ISSUING SEEK
  985  001330  031322                    01640            DH1     ;PC     RKCS    RKER    RKDS    RKDA
  986  001332  032014                    01650            DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
  987  001334  000000                    01660            0
  988                                    01670
  989                                    01680    ;ITEM   13
  990                                    01690
  991  001336  027763                    01700            EM13    ;CONTROL READY NOT SET AFTER SEEK DONE INTERRUPT
  992  001340  031322                    01710            DH1     ;PC     RKCS    RKER    RKDS    RKDA
  993  001342  032014                    01720            DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
  994  001344  000000                    01730            0
  995                                    01740
  996                                    01750    ;ITEM   14
  997                                    01760
  998  001346  030036                    01770            EM14    ;INTERRUPTING DRIVE (SEEK DONE) WAS NOT 'BUSY'
  999  001350  031322                    01780            DH1     ;PC     RKCS    RKER    RKDS    RKDA
 1000  001352  032014                    01790            DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
 1001  001354  000000                    01800            0
 1002                                    01810
 1003                                    01820    ;ITEM   15
 1004                                    01830
 1005  001356  030111                    01840            EM15    ;R/W/S READY NOT SET FOR INTERRUPTING DRIVE (SEEK DONE)
 1006  001360  031322                    01850            DH1     ;PC     RKCS    RKER    RKDS    RKDA
```

```
1007  001362  032014                  01860            DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
1008  001364  000000                  01870            0
1009                                  01880
1010                                  01890    ;ITEM   16
1011                                  01900
1012  001366  030175                  01910            EM16    ;'SIN' ERROR
1013  001370  031322                  01920            DH1     ;PC     RKCS    RKER    RKDS    RKDA
1014  001372  032014                  01930            DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
1015  001374  000000                  01940            0
1016                                  01950
1017                                  01960    ;ITEM   17
1018                                  01970
1019  001376  030206                  01980            EM17    ;'ERR'OR ON DOING SEEK
1020  001400  031322                  01990            DH1     ;PC     RKCS    RKER    RKDS    RKDA
1021  001402  032014                  02000            DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
1022  001404  000000                  02010            0
1023                                  02020
1024                                  02030    ;ITEM   20
1025                                  02040
1026  001406  030234                  02050            EM20    ;SCP DID NOT SET AFTER SEEK WAS DONE
1027  001410  031322                  02060            DH1     ;PC     RKCS    RKER    RKDS    RKDA
1028  001412  032014                  02070            DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
1029  001414  000000                  02080            0
1030                                  02090
1031                                  02100    ;ITEM   21
1032                                  02110
1033  001416  030300                  02120            EM21    ;SOFT ERROR
1034  001420  031405                  02130            DH21    ;$ERRPC RKCS    RKER    RKDS    RKDA:   DRV#
1035                                  02140                    ;CYL    SUR     SEC
1036  001422  032036                  02150            DT21    ;$ERRPC $REG0   $REG1   $REG2           $REG3
1037                                  02160                    ;$REG4  $REG5   $REG6
1038  001424  000000                  02170            0
1039                                  02180
1040                                  02190    ;ITEM   22
1041                                  02200
1042  001426  000000                  02210            0
1043  001430  031405                  02220            DH21    ;$ERRPC RKCS    RKER    RKDS    RKDA:   DRV#
1044                                  02230                    ;CYL    SUR     SEC
1045  001432  032036                  02240            DT21    ;$ERRPC $REG0   $REG1   $REG2           $REG3
1046                                  02250                    ;$REG4  $REG5   $REG6
1047  001434  000000                  02260            0
1048                                  02270
1049                                  02280    ;ITEM   23
1050                                  02290
1051  001436  030312                  02300            EM23    ;DATA (COMPARISON) ERROR
1052  001440  031502                  02310            DH23    ;PC     RKBA    EXPCT   RECVD   RKDA
1053  001442  032014                  02320            DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
1054  001444  000000                  02330            0
1055                                  02340
1056                                  02350    ;ITEM   24
1057                                  02360
1058  001446  030341                  02370            EM24    ;CONTROL READY CLEAR ON  INTERRUPT AFTER RK FUNCTION
1059  001450  031322                  02380            DH1     ;PC     RKCS    RKER    RKDS    RKDA
1060  001452  032014                  02390            DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
1061  001454  000000                  02400            0
1062                                  02410
```

```
1063                                    02420   ;ITEM   25
1064                                    02430
1065    001456  000060                  02440           0
1066    001460  031547                  02450           DH25    ;PC     RKCS    RKER    RKDS    RKDA    DRIVE #
1067    001462  032060                  02460           DT25    ;$ERRPC $REG0   $REG1   $REG2   $REG3   $REG4
1068    001464  000000                  02470           0
1069                                    02480
1070                                    02490   ;ITEM   26
1071                                    02500
1072    001466  030415                  02510           EM26    ;STUCK IN LOOP, 8 Q-COMMANDS SHOULD BE DONE BY NOW
1073    001470  031322                  02520           DH1     ;PC     RKCS    RKER    RKDS    RKDA
1074    001472  032014                  02530           DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
1075    001474  000000                  02540           0
1076                                    02550
1077                                    02560   ;ITEM   27
1078                                    02570
1079    001476  030470                  02580           EM27    :ATTEMPT TO DO WRITE CHECK BEFORE WRITE
1080    001500  031624                  02590           DH27    ;PC     KEY     FUNCTION CODE
1081    001502  032076                  02600           DT103   ;PC     $REG0   $REG1
1082    001504  000000                  02610           0
1083                                    02620
1084                                    02630   ;ITEM   30
1085                                    02640
1086    001506  030531                  02650           EM30    ;ATTEMPT TO REEXECUTE A COMMAND IN PROGRESS OR ALREADY FINISHED
1087    001510  031705                  02660           DH30    ;PC     KEY
1088    001512  032030                  02670           DT2     ;$ERRPC $REG0
1089    001514  000000                  02680           0
1090                                    02690
1091                                    02700   ;ITEM   31
1092                                    02710
1093    001516  030624                  02720           EM31    ;'FUNCTION IN PROGRESS' FLAG FOR INTERRUPTING DRIVE IS NOT SET
1094    001520  031370                  02730           DH2     ;PC     DRIVE
1095    001522  032030                  02740           DT2     ;$ERRPC $REG0
1096    001524  000000                  02750           0
1097                                    02760
1098                                    02770   ;ITEM   32
1099                                    02780
1100    001526  030715                  02790           EM32    ;UNEXPECTED DRIVE INTERRUPTED
1101    001530  031657                  02800           DH103   ;PC     EXPCT   RECVD
1102    001532  032076                  02810           DT103   ;$ERRPC $REG0   $REG1
1103    001534  000000                  02820           0
1104                                    02830
1105                                    02840   ;ITEM   33
1106                                    02850
1107    001536  030747                  02860           EM33    ;WRONG FUNCTION CODE IN RKCS AFTER INTERRUPT
1108    001540  031657                  02870           DH103   ;PC     EXPCT   RECVD
1109    001542  032076                  02880           DT103   ;$ERRPC $REG0   $REG1
1110    001544  000000                  02890           0
1111                                    02900
1112                                    02910   ;ITEM   34
1113                                    02920
1114    001546  031024                  02930           EM34    ;DRIVE READY CLEAR
1115    001550  031322                  02940           DH1     ;PC     RKCS    RKER    RKDS    RKDA
1116    001552  032014                  02950           DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
1117    001554  000000                  02960           0
1118                                    02970
```

```
1119
1120
1121  001556  031043                    02980   ;ITEM   35
1122  001560  031322                    02990
1123  001562  032014                    03000           EM35    ;DRIVE POWER LOW
1124  001564  000000                    03010           DH1     ;PC     RKCS    RKER    RKDS    RKDA
1125                                     03020           DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
1126                                     03030           0
1127                                     03040
1128  001566  031061                    03050   ;ITEM   36
1129  001570  031322                    03060
1130  001572  032014                    03070           EM36    ;DRIVE UNSAFE
1131  001574  000000                    03080           DH1     ;PC     RKCS    RKER    RKDS    RKDA
1132                                     03090           DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
1133                                     03100           0
1134                                     03110
1135  001576  031075                    03120   ;ITEM   37
1136  001600  031322                    03130
1137  001602  032014                    03140           EM37    ;WPS SET
1138  001604  000000                    03150           DH1     ;PC     RKCS    RKER    RKDS    RKDA
1139                                     03160           DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
1140                                     03170           0
1141                                     03180
1142                                     03190   ;*
1143                                     03200   ;*THE FOLLOWING ERRORS OCCUR IN THE NON-EXERCISER PART OF THE PROGRAM.
1144                                     03210   ;*
1145                                     03220
1146  001606  030312                    03230   ;ITEM   100
1147  001610  031502                    03240
1148  001612  032014                    03250           EM23    ;DATA (COMPARISON) ERROR
1149  001614  000000                    03260           DH23    ;PC     RKBA    EXPCT   RECVD   RKDA
1150                                     03270           DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
1151                                     03280           0
1152                                     03290
1153  001616  031105                    03300   ;ITEM   101
1154  001620  031322                    03310
1155  001622  032014                    03320           EM101   ;INTERRUPT DID NOT OCCUR AFTER WRITE
1156  001624  000000                    03330           DH1     ;PC     RKCS    RKER    RKDS    RKDA
1157                                     03340           DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
1158                                     03350           0
1159                                     03360
1160  001626  031145                    03370   ;ITEM   102
1161  001630  031322                    03380
1162  001632  032014                    03390           EM102   ;'ERR'OR SET
1163  001634  000000                    03400           DH1     ;PC     RKCS    RKER    RKDS    RKDA
1164                                     03410           DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
1165                                     03420           0
1166                                     03430
1167  001636  031161                    03440   ;ITEM   103
1168  001640  031657                    03450
1169  001642  032076                    03460           EM103   ;RKDA   INCREMENTED WRONGLY
1170  001644  000000                    03470           DH103   ;PC     EXPCT   RECVD
1171                                     03480           DT103   ;$ERRPC $REG0   $REG1
1172                                     03490           0
1173                                     03500
1174  001646  031207                    03510   ;ITEM   104
                                         03520
                                         03530           EM104   ;RKBA INCREMENTED WRONGLY
```

```
 1175  001650  031657                  03540            DH103   ;PC     EXPCT   RECVD
 1176  001652  032076                  03550            DT103   ;$ERRPC $REG0   $REG1
 1177  001654  000000                  03560            0
 1178                                  03570
 1179                                  03580    ;ITEM   105
 1180                                  03590
 1181  001656  031235                  03600            EM105   ;RKWC DID NOT OVERFLOW TO 0
 1182  001660  031721                  03610            DH105   ;PC     RKDA    RKWC
 1183  001662  032076                  03620            DT103   ;$ERRPC $REG0   $REG1
 1184  001664  000000                  03630            0
 1185                                  03640
 1186                                  03650    ;ITEM   106
 1187                                  03660
 1188  001666  031265                  03670            EM106   ;MEX BITS INCORRECT
 1189  001670  031322                  03680            DH1     ;PC     RKCS    RKER    RKDS    RKDA
 1190  001672  032014                  03690            DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
 1191  001674  000000                  03700            0
 1192                                  03710
 1193                                  03720    ;ITEM   107
 1194                                  03730
 1195  001676  030312                  03740            EM23    ;DATA (COMPARISON) ERROR ON READ
 1196  001700  031657                  03750            DH103   ;PC     EXPCT   RECVD
 1197  001702  032076                  03760            DT103   ;$ERRPC $REG0   $REG1
 1198  001704  000000                  03770            0
 1199                                  03780
 1200                                  03790    ;ITEM   110
 1201                                  03800
 1202  001706  031304                  03810            EM110   ;WRITE CHECK ERROR
 1203  001710  031746                  03820            DH110   ;PC     RKCS    RKER    RKBA    RKDA
 1204  001712  032014                  03830            DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
 1205  001714  000000                  03840            0
```

```
1206  001716  177400                  03860  RKDS:   .WORD   177400
1207  001720  177402                  03870  RKER:   .WORD   177402
1208  001722  177404                  03880  RKCS:   .WORD   177404
1209  001724  177406                  03890  RKWC:   .WORD   177406
1210  001726  177410                  03900  RKBA:   .WORD   177410
1211  001730  177412                  03910  RKDA:   .WORD   177412
1212  001732  177416                  03920  RKDB:   .WORD   177416
1213  001734  177546                  03930  KWLS:   .WORD   177546  ;STATUS REGISTER FOR KW11L
1214                                  03940
1215  001736  000372                  03950  PCNTR:  .WORD   250.
1216                                  03960
1217  001740  000220                  03970  RKVEC:  .WORD   220      ;NORMAL RK11 INTERRUPT VECTOR ADDRESS
1218  001742  000222                  03980  RKSTAT: .WORD   222      ;PSW TO BE USED ON INTERRUPT
1219                                  03990
1220  001744  000240                  04000  PPRLVL: .WORD   240      ;PROGRAM PRIORITY LEVEL=5. PRIORITY LEVEL
1221                                  04010                           ;AT WHICH THE PROGRAM OPERATES CAN BE CHANGED
1222                                  04020                           ;BY ALTERING THIS LOCATION.
1223  001746  000340                  04030  KWPLVL: .WORD   340      ;PRIORITY LEVEL OF THE KW11L CLOCK SERVICE
1224                                  04040                           ;ROUTINE.
1225                                  04050
1226  001750  177777                  04060  SRDRV:  .WORD   177777  ;'SRDRV' CONTAINS THE DRIVE NO WHOOSE SERIAL
1227                                  04070                           ;NO IS TO BE TYPED OUT WHEN AN ERROR OCCURS,
1228                                  04080                           ;IF SW 1 IS SET. WHEN (SRDRV)=-1 SERIAL NO
1229                                  04090                           ;IS NOT TYPPED OUT, BECAUSE THE ERROR WAS NOT
1230                                  04100                           ;POSITIVELY ATTRIBUTABLE TO A SPECIFIC DRIVE.
1231                                  04110
1232                                  04120
1233  001752     000                  04130  FTITLE: .BYTE   0
1234  001753     000                  04140  FRSTRT: .BYTE   0        ;FLAG FOR RESTART AT 210
```

```
                                        04160   ;THIS TABLE CONTAINS (IN ASCENDING ORDER) THE DRIVE NUMBERS THAT ARE
                                        04170   ;PRESENT. THUS IF 3 DRIVES 0,1,2 ARE PRESENT: PDR WILL CONTAIN  PDR1 WILL
                                        04180   ;CONTAIN 1 AND PDR2 WILL CONTAIN 2.
                                        04190
001754  000010                          04200   PDR:    .BLKB   10
                                        04210
001764  000000                          04220   DRVPRS: .WORD   0       ;CONTAINS TOTAL NUMBER OF DRIVES PRESENT
                                        04230
                                        04240   ;THE FOLLOWING LOCATIONS CONTAIN SERIAL NUMBERS CORRESPONDING TO EACH
                                        04250   ;DRIVE. THE SERIAL NUMBERS ARE KEYED IN BY THE USER, WHEN THE PROGRAM
                                        04260   ;IS STARTED WITH SWITCH 1 SET TO 1. THIS FEATURE IS NORMALLY USED IN
                                        04270   ;PRODUCTION ENVIRONMENT.
                                        04280
001766  000010                          04290   SRNO:   .BLKW   10      ;SERIAL NO'S FOR DRIVES 0-7
                                        04300
                                        04310
                                        04320   ;THE FOLLOWING 8 KEYS ARE FOR THE 8 COMMANDS IN THE QUEUE, TO BE
                                        04330   ;EXECUTED ON DIFFERENT DRIVES. EACH KEY IS ASSOCIATED WITH AN EXECUTABLE
                                        04340   ;COMMAND ON THE RK11. VARIOUS BITS OF THE KEY DESCRIBE A COMMAND
                                        04350   ;AS INDICATED BELOW
                                        04360
                                        04370   ;<0-2>  DRIVE NUMBER ON WHICH THE COMMAND IS TO BE EXECUTED
                                        04380   ;<4>    INDICATES THAT THE HEADS ARE BEING/OR HAVE BEEN
                                        04390   ;       POSITIONED ON THE DRIVE
                                        04400   ;<5>    INDICATES A 'WRT CHK' SHOULD BE DONE FOLLOWING THE 'WRITE'
                                        04410   ;<6>    INDICATES A WRITE CHECK FUNCTION HAS BEEN INITIATED
                                        04420   ;<7>    INDICATES THAT A FUNCTION IS IN PROGRESS (IT IS NOT SET
                                        04430   ;       WHEN POSITIONING IS BEING DONE ON A DRIVE)
                                        04440   ;<8-10> INDICATES THE POSITION OF THIS KEY IN THE 8-KEY TABLE
                                        04450   ;       (POSITIONS BEING 0,1,2,3,4,5,6,7)
                                        04460   ;<11>   INDICATES THAT FUNCTION CORRESPONDING TO THIS KEY HAS
                                        04470   ;       BEEN ABORTED
                                        04480   ;<12>   INDICATES HIGH PRIORITY FOR THE COMMAND (NORMALLY
                                        04490   ;       SET AFTER AN ERROR OCCURED ON THE COMMAND)
                                        04500   ;<14>   INDICATES THAT THE COMMAND CORRESPONDING TO THIS KEY HAS BEEN
                                        04510   ;       ABORTED BECAUSE THE DRIVE WAS DESELECTED (DSELECT)
                                        04520   ;<15>   INDICATES THAT THE COMMAND HAS BEEN COMPLETED
                                        04530   ;       (ALSO SET WHEN COMMAND IS ABORTED AFTER RETRIES)
                                        04540
                                        04550
002006  000010                          04560   KEY:    .BLKW   10      ;KEY FOR THE  COMMANDS IN QUEUE
                                        04570
                                        04580
                                        04590   ;THE PARAMETERS TO BE USED FOR EACH COMMAND IN THE QUEUE
                                        04600   ;ARE STORED IN A TABLE STARTING AT 'CMND'. BITS <8-10>
                                        04610   ;OF THE COMMAND KEYS (KEY, KEY2, ---KEY8) ARE USED TO POINT
                                        04620   ;TO THE RIGHT SET OF PARAMETERS.
                                        04630
                                        04640
                                        04650   ;       WORD 1 CONTAINS RKDA TO BE USED
                                        04660   ;       WORD 2 CONTAINS RKCS (FUNCTION BITS ONLY)
                                        04670   ;       WORD 3 CONTAINS RKWC (WORD COUNT 2'S COMP)
                                        04680   ;       WODR 4 CONTAINS RKBA
                                        04690
002026  000040                          04700   CMND:   .BLKW   40      ;STORAGE TABLE
```

```
 1290                                   04720
 1291                                   04730
 1292                                   04740   ;THESE ARE BUSY FLAGS FOR THE DRIVES. IF A DRIVE IS BUSY PERFORMING
 1293                                   04750   ;ANY FUNCTION (INCLUDING POSITIONING) THEN BIT 7 OF THE FLAG FOR THAT
 1294                                   04760   ;DRIVE IS SET. BITS 0-3 CONTAIN THE OFFSET TO KEY # WHICH MADE THE DRIVE
 1295                                   04770   ;BUSY. EX: DRIVE #3 WAS MADE TO DO A WRITE BY COMMAND
 1296                                   04780   ;KEY5, HENCE 'BUSY3' WILL CONTAIN 210. NOTE THAT 10 IS THE
 1297                                   04790   ;OFFSET FOR KEY5 (TAKING KEY AS BASE).  KEY #= OFFSET(0-3)/2 + 1
 1298                                   04800
 1299  002126  000010                  04810   BUSY:   .BLKB   10      ;BUSY FLAGS FOR DRIVES 0-7
 1300                                   04820
 1301                                   04830
 1302                                   04840   ;THESE FLAGS WHEN SET INDICATE THAT A DRIVE IS BEING
 1303                                   04850   ;POSITIONED OR HAS ALREADY BEEN POSITIONED.
 1304                                   04860
 1305  002136  000010                  04870   POS:    .BLKB   10      ;DRIVE 0 POSITIONED
 1306                                   04880
 1307                                   04890
 1308                                   04900
 1309                                   04910   ;RETRY COUNTS FOR A PARTICULAR FUNCTION ON A DRIVE THE FUNCTION IS ABORTED
 1310                                   04920   ;ON A DRIVE WHEN THE RETRY COUNT REACHES 3.
 1311                                   04930
 1312  002146  000010                  04940   RETRY:  .BLKB   10      ;DRIVES 0-7 RERTY COUNTS
```

```
1313  002156  000000                  04960  WCFLG:  .WORD  0       ;IF BIT 15 IS SET WRITE CHK IS TO BE DONE
1314                                  04970                         ;FOLLOWING THE WRITE. BITS 0-3 CONTAIN THE
1315                                  04980                         ;OFFSET TO KEY# (FROM BASE=KEY)
1316                                  04990
1317  002160  000000                  05000  QSCNT:  .WORD  0       ;THIS IS A COUNT FOR KEEPING TRACK OF THE TIME
1318                                  05010                         ;TAKEN BY ALL THE 8 COMMANDS IN THE QUEUE.
1319                                  05020                         ;IF THIS COUNTS DOWN TO 0 AN ERROR IS REPORTED
1320                                  05030
1321                                  05040
1322  002162  000000                  05050  PRSFNC: .WORD  0       ;COTAINS INFO ABOUT THE PRESENT COMMAND
1323                                  05060                         ;BEING PERFORMED ON THE RK11
1324  002164  000000                  05070  PSTFNC: .WORD  0       ;CONTAINS INFO ABOUT THE COMMAND PERFORMED
1325                                  05080                         ;BEFORE THE 'PRSCMND'
1326                                  05090
1327                                  05100
1328  002166  000000                  05110  CICNT:  .WORD  0       ;THIS IS A COUNT-TIMER USED FOR KEEPING TRACK
1329  002170  000000                  05120  CICNT1: .WORD  0       ;OF THE TIME TAKEN BY ANY FUNCTION TO BE
1330                                  05130                         ;COMPLETED. IF THE COUNT GOES TO 0 AN ERROR IS REPORTED.
1331                                  05140
1332  002172  000000                  05150  TIMER:  .WORD  0
1333  002174  000000                  05160  ERCODE: .WORD  0
1334  002176  000000                  05170  DRVPTR: .WORD  0
1335  002200  000000                  05180  DRVCNT: .WORD  0
1336                                  05190
1337                                  05200
1338  002202  000000                  05210  QDRV:   .WORD  0       ;TEMPORARY REGISTERS USED BY 'GEN8RQ'
1339  002204  000000                  05220  QCYL:   .WORD  0       ;ROUTINE TO STORE VARIOUS PARAMETERS
1340  002206  000000                  05230  QSUR:   .WORD  0       ;OF A COMMAND AS THEY ARE GENERATED.
1341  002210  000000                  05240  QSEC:   .WORD  0
1342  002212  000000                  05250  QFNC:   .WORD  0
1343  002214  000000                  05260  QBUSAD: .WORD  0
1344  002216  000000                  05270  QWRCNT: .WORD  0
1345                                  05280
1346                                  05290
1347                                  05300  ;THIS TABLE CONTAINS VARIOUS MAPPING FACTORS TO BE USED
1348                                  05310  ;FOR GENERATING RANDOM PARAMETERS FROM RANDOM NUMBERS
1349                                  05320
1350  002220  000000                  05330  DRMAP:  .WORD  0       ;MAPPING FACTOR FOR GENERATING RANDOM DRIVE NUMBER
1351  002222  000000                  05340  CYLMAP: .WORD  0       ;MAPPING FACTOR FOR CYLINDER
1352  002224  000000                  05350  SECMAP: .WORD  0       ;MAPPING FACTOR FOR SECTOR
1353  002226  000000                  05360  FNMAP:  .WORD  0       ;MAPPING FACTOR FOR FUNCTION
1354  002230  000000                  05370  BAMAP:  .WORD  0       ;MAPPING FACTOR FOR BUS ADDRESS
1355  002232  000000                  05380  WCMAP:  .WORD  0       ;MAPPING FACTOR FOR WORD COUNT
1356                                  05390
1357                                  05400
1358                                  05410  ;THESE TWO FLAGS CORRESPOND TO THE 2 INTERRUPT HANDLERS (RK11) USED
1359                                  05420  ;IN THIS PROGRAM. WHEN THE INTERRUPT HANDLER IS ENTERED THE FLAG IS
1360                                  05430  ;CLEARED OR SET.
1361                                  05440
1362  002234     000                  05450  INTFLG: .BYTE  0       ;FOR 'INTHND ', CLEARED ON ENTERING HANDLER
1363  002235     000                  05460  INT1FL: .BYTE  0       ;FOR 'INT1SK','SET ON ENTERING HANDLER
1364                                  05470
1365  002236  000000                  05480  SAVKEY: .WORD  0
1366  002240  000000                  05490  ECOUNT: .WORD  0
```

```
                                        05510
                                        05520
                                        05530   ;THIS TABLE CONTAINS COUNTS FOR THE NUMBER OF OF ERRORS OCCURING ON A
                                        05540   ;DRIVE (NOTE: ONLY THOSE ERRORS WHICH ARE POSITIVELY ATTRIBUTABLE TO A
                                        05550   ;SPECIFIC DRIVE). THE COUNT  KEPT ONLY IF SWITCH 2 IS SET, WHEN THE COUNT
                                        05560   ;REACHES THE MAXIMUM ALLOWABLE (USUALLY 3) THE DRIVE IS DROPPED FROM
                                        05570   ;TESTING AND IS TAKEN OUT OF THE DRIVE SELECTION TABLE.
                                        05580
002242  000010                          05590   ERDRV:  .BLKB   10      ;COUNT FOR DRIVES 0-7
                                        05600
002252  000000                          05610   KWHR:   .WORD   0       ;COUNTS HOURS (2'S COMPLEMENT)
002254  000000                          05620   KWMIN:  .WORD   0       ;COUNTS MINUTES (2'S COMPLEMENT)
002256  000000                          05630   KWSEC:  .WORD   0       ;COUNTS SECONDS (2'S COMPLEMENT)
002260  000000                          05640   KWCOUNT: .WORD  0       ;COUNTS CPS FROM KW11L (2'S COMPLMNT)
                                        05650
                                        05660   ;THIS TABLE CONTAINS COUNTS FOR HARD ERRORS ON A PARTICULAR DRIVE.
                                        05670   ;EX HECN2 WILL CONTAIN THE TOTAL NUMBER OF HARD ERRORS THAT OCCURED ON
                                        05680   ;DRIVE 2
                                        05690
002262  000010                          05700   HECN:   .BLKW   10      ;DRIVE 0-7 HARD ERROR COUNTS
                                        05710
                                        05720   ;THIS TABLE CONTAINS COUNTS FOR SEEK ERRORS
                                        05730   ;ON A PARTICULAR DRIVE.
                                        05740
002302  000010                          05750   SKECN:  .BLKW   10      ;DRIVE 0-7 SEEK ERROR COUNTS
                                        05760
                                        05770
                                        05780   ;THIS TABLE CONTAINS COUNTS FOR SIN ERRORS ON A
                                        05790   ;PARTICULAR DRIVE
                                        05800
002322  000010                          05810   SINCN:  .BLKB   10      ;DRIVE 0-7 SIN COUNTS
                                        05820
                                        05830   ;THIS TABLE CONTAINS COUNTS FOR WRITE CHECK ERRORS
                                        05840   ;THAT OCCURED ON A PARTICULAR DRIVE
                                        05850
002332  000010                          05860   WCECN:  .BLKW   10      ;WCE COUNT FOR DRIVES 0-7
                                        05870
                                        05880
                                        05890   ;THIS TABLE CONTAINS COUNTS FOR CHECK SUM ERROR THAT
                                        05900   ;OCCURED ON A PARTICULAR DRIVE
                                        05910
002352  000010                          05920   CSECN:  .BLKW   10      ;CSE COUNT FOR DRIVES 0-7
                                        05930
                                        05940
                                        05950   ;THIS TABLE CONTAINS COUNT OF NUMBER OF FUNCTIONS
                                        05960   ;THAT WERE ABORTED ON A PARTICULAR DRIVE. A
                                        05970   ;FUNCTION IS ABORTED ONLY AFTER DOING RETRIES
                                        05980
002372  000010                          05990   ABORT:  .BLKW   10      ;ABORT COUNT FOR DRIVES 0-7
                                        06000
                                        06010   ;COUNTS FOR NUMBER OF DATA ERRORS THAT OCCURED ON INDIVIDUAL DRIVES.
                                        06020
002412  000010                          06030   DATER:  .BLKW   10      ;DRIVES 0-7
```

```
1420  002432  000000                  06050  NWRTL:  .WORD   0       ;LO WORD: OF THE 2 WORD COUNT-GIVING TOTAL
1421  002434  000000                  06060  NWRTH:  .WORD   0       ;HI WORD:# OF WORDS WRITTEN ON DRIVE 0
1422  002436  000016                  06070          .BLKW   14.     ;FOR REST OF DRIVES 1-7
1423                                  06080
1424                                  06090
1425  002472  000000                  06100  NRDL:   .WORD   0       ;LO WORD: 2 WORD COUNT GIVING TOTAL
1426  002474  000000                  06110  NRDH:   .WORD   0       ;HI WORD: # OF WORDS READ ON DRIVE 0
1427  002476  000016                  06120          .BLKW   14.     ;FOR DRIVES 1-7
1428                                  06130
1429  002532  002026                  06140  PCMND:  .WORD   CMND    ;POINTERS TO PARAMETERS FOR COMMANDS IN QUEUE
1430  002534  002036                  06150          .WORD   CMND+10 ;POINTER TO SECOND COMMAND
1431  002536  002046                  06160          .WORD   CMND+20 ;POINTER TO THIRD COMMAND
1432  002540  002056                  06170          .WORD   CMND+30 ;POINTER TO FOURTH COMMAND
1433  002542  002066                  06180          .WORD   CMND+40 ;POINTER TO FIFTH COMMAND
1434  002544  002076                  06190          .WORD   CMND+50 ;POINTER TO SIXTH COMMAND
1435  002546  002106                  06200          .WORD   CMND+60 ;POINTER TO SEVENTH COMMAND
1436  002550  002116                  06210          .WORD   CMND+70 ;POINTER TO EIGHTH COMMAND
1437                                  06220
1438                                  06230
1439  002552  000000                  06240  BASEBA: .WORD   0       ;CONTAINS THE LOWEST BUS ADDRESS STARTING WHICH DATA TRA
1440                                  06250                          ;CAN BE DONE
1441  002554  000000                  06260  MAXBA:  .WORD   0       ;CONTAINS THE HIGHEST BUS ADDRESS TO WHICH DATA TRANSFER
1442                                  06270                          ;CAN BE DONE.
1443  002556  000000                  06280  REPCNT: .WORD   0       ;CONTAINS THE REPETITION COUNT- THE NUMBER
1444                                  06290                          ;OF TIMES Q REQUESTS WILL BE GENERATED. WHEN THIS
1445                                  06300                          ;COUNT GOES TO 0, IT MEANS AN END OF PASS. HOWEVER
1446                                  06310                          ;NOTE THAT THERE IS NO TRUE END OF PASS, IN THIS KIND
1447                                  06320                          ;OF EXERCISER PROGRAM. THE EXERCISER RESUMES FROM
1448                                  06330                          ;THE POINT IT LEFT OFF, AFTER TYPING OUT THE END IF
1449                                  06340                          ;PASS MESSAGE.
1450                                  06350
1451                                  06360
1452  002560  000                     06370  RKDPDU: .BYTE   0       ;FLAG SET WHEN PROGRAM OPERATING IN RKDP DUMP MODE
1453  002561  000                     06380  RKDPCH: .BYTE   0       ;FLAG SET WHEN OPERATING IN RKDP CHAIN MODE
1454  002562  000                     06390  PPTP:   .BYTE   0       ;FLAG SET WHEN PROGRAM IS PAPER TAPE LOADED
1455  002563  000                     06400  ACT11:  .BYTE 0         ;FLAG SET WHEN OPERATING UNDER ACT11
1456                                  06410          .EVEN
```

```
                                                06430   ;ASCII MESSAGES
                                                06440
002564  005015  045523  000105  06450   MSG1:   .ASCIZ  <15><12>/SKE/
002572  005015  041527  000105  06460   MSG2:   .ASCIZ<15><12>/WCE/
002600  005015  051503  000105  06470   MSG3:   .ASCIZ<15><12>  /CSE/
002606  005015  040510  042122  06480   MSG4:   .ASCIZ  <15><12>/HARD EROR/
002614  042440  047522  000122
002622  047440  020116  047504  06490   MSG5:   .ASCIZ/ ON DOING /
002630  047111  020107     000
002635     127  044522  042524  0650C   MSG6:   .ASCIZ  /WRITE/
002642     000
002643     122  040505  000104  06510   MSG7:   .ASCIZ  /READ/
002650  051127  020124  044103  06520   MSG8:   .ASCIZ  /WRT CHK/
002656  000113
002660  042122  041440  045510  06530   MSG9:   .ASCIZ  /RD CHK/
002666     000
002667     015  040412  047502  06540   MSG10:  .ASCIZ  <15><12>/ABORTED/<15><12>
002674  052122  042105  005015
002702     000
002703     123  042505  000113  06550   MSG11:  .ASCIZ  /SEEK/
002710  005015  041520  000075  06560   MSG12:  .ASCIZ  <15><12>/PC=/
002716  044120  051531  041040  06570   MSG13:  .ASCIZ  /PHYS BA=/
002724  036501     000
002727     015  047012  020117  06580   MSG14:  .ASCIZ  <15><12>/NO DRVS PRSNT/
002734  051104  051526  050040
002742  051522  052116     000
002747     015  042012  053122  06590   MSG15:  .ASCIZ  <15><12>/DRVE # DIDN'T INTERUPT AFTER /
002754  020105  020043  044504
002762  047104  052047  044440
002770  052116  051105  050125
002776  020124  043101  042524
003004  020122     000
003007     015  045412  054505  06600   MSG16:  .ASCIZ  <15><12>/KEY-8   BUSY-7/
003014  034055  020040  041040
003022  051525  026531  000067
003030  051440  020122  047516  06610   MSG17:  .ASCIZ  / SR NO/
003036     000
003037     072     000          06620   MSG18:  .ASCIZ  /:/
003041     015  020012  051104  06630   MSG19:  .ASCIZ  <15><12>/ DROPPED DRIVE # /
003046  050117  042520  020104
003054  051104  053111  020105
003062  020043     000
003065     015  042012  044522  06640   MSG20:  .ASCIZ  <15><12>/DRIVE/
003072  042526     000
003075     015  051012  050105  06650   MSG21:  .ASCIZ  <15><12>/REPLACE DRO RKDP-PAK $ TYPE CR, IF NOT PUT DRO ON LOAD/
003102  040514  042503  042040
003110  030122  051040  042113
003116  026520  040520  020113
003124  020044  054524  042520
003132  041440  026122  044440
003140  020106  047516  020124
003146  052520  020124  051104
003154  020060  047117  046040
003162  040517  000104
003166  005015  051104  020060  06660   MSG23:  .ASCIZ  <15><12>/DRO NOT TESTED/
003174  047516  020124  042524
```

```
1513  003202  052123  042105     000
1514  003207     054  000040                06670   MSG24:  .ASCIZ  . /
1515  003212  000106                        06680   MSG25:  .ASCIZ  /F/
1516  003214  005015  051104  053111        06690   MSG26:  .ASCIZ  <15><12>/DRIVE  WRDS WRITN  WRDS READ    CSE    WCE DATERR     H
1517  003222  020105  053440  042122
1518  003230  020123  051127  052111
1519  003236  020116  053440  042122
1520  003244  020123  042522  042101
1521  003252  020040  020040  051503
1522  003260  020105  020040  053440
1523  003266  042503  042040  052101
1524  003274  051105  020122  020040
1525  003302  020040  042510  020040
1526  003310  020040  045523  020105
1527  003316  040440  047502  052122
1528  003324  020040  020040  044523
1529  003332  000116
1530  003334  005015  047125  041101        06700   MSG27:  .ASCIZ  <15><12>/UNABLE TO CLEAR ERROR AFTER THREE TRIES/
1531  003342  042514  052040  020117
1532  003350  046103  040505  020122
1533  003356  051105  047522  020122
1534  003364  043101  042524  020122
1535  003372  044124  042522  020105
1536  003400  051124  042511  000123
1537  003406  005015  051105  047522        06710   MSG28:  .ASCIZ  <15><12>/ERROR CONDITION CLEARED ON RETRY # /
1538  003414  020122  047503  042116
1539  003422  052111  047511  020116
1540  003430  046103  040505  042522
1541  003436  020104  047117  051040
1542  003444  052105  054522  021440
1543  003452  000040
1544                                        06720
1545  003454     040                        06730   BLNKS3: .ASCII  / /
1546  003455     040                        06740   BLNKS2: .ASCII  / /
1547  003456  000040                        06750   BLNKS1: .ASCIZ  / /
1548                                        06760           .EVEN
```

```
1549                                                    06780   ;IF POWER FAILED, ON RETURN OF POWER ENTER HERE.
1550                                                    06790
1551  003460  004737  022260                            06800   PFSTRT: JSR     PC,WATIME       ;WAIT SOME TIME
1552  003464  105237  001753                            06810           INCB    FRSTRT          ;INDICATE THAT THE STATISTICS HAVE
1553                                                    06820                                   ;TO BE SAVED, ON RETRN FROM PWR FAIL.
1554                                                    06830
1555                                                    06840
1556                                                    06860
1557                                                    06870
1558  003470                                            06880   START:
1559                                                            .SBTTL  INITIALIZE THE COMMON TAGS
1560                                                            ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
1561  003470  012706  001100                                           MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
1562  003474  005026                                            CLR     (R6)+           ;;CLEAR MEMORY LOCATION
1563  003476  022706  001140                                    CMP     #SWR,R6 ;;DONE?
1564  003502  001374                                           BNE     .-6             ;;LOOP BACK IF NO
1565  003504  012706  001100                                   MOV     #STACK,SP       ;;SETUP THE STACK POINTER
1566                                                    ;;INITIALIZE A FEW VECTORS
1567  003510  012737  026612  000020                           MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
1568  003516  012737  000340  000022                           MOV     #340,@#IOTVEC+2 ;;LEVEL 7
1569  003524  012737  026756  000034                           MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1570  003532  012737  000340  000036                           MOV     #340,@#TRAPVEC+2;LEVEL 7
1571  003540  012737  027042  000024                           MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
1572  003546  012737  000340  000026                           MOV     #340,@#PWRVEC+2 ;;LEVEL 7
1573  003554  012737  003554  001106                           MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1574  003562  012737  003562  001110                           MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
1575                                                    ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1576                                                    ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
1577  003570  013746  000004                           MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
1578  003574  012737  003630  000004                   MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
1579  003602  012737  177570  001140                   MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
1580  003610  012737  177570  001142                   MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
1581  003616  022777  177777  175314                   CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
1582  003624  001012                                   BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1583                                                                            ;;AND  THE HARDWARE SWR IS NOT = -1
1584  003626  000403                                   BR      65$             ;;BRANCH IF NO TIMEOUT
1585  003630  012716  003636                   64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
1586  003634  000002                                   RTI
1587  003636  012737  000176  001140           65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
1588  003644  012737  000174  001142                   MOV     #DISPREG,DISPLAY
1589  003652  012637  000004                   66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
1590
1591                                                    .SBTTL  TYPE PROGRAM NAME
1592                                                    ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
1593  003656  005227  177777                           INC     #-1             ;;FIRST TIME?
1594  003662  001052                                   BNE     67$             ;;BRANCH IF NO
1595  003664  104400  003722                           TYPE    ,68$            ;;TYPE ASCIZ STRING
1596                                                    .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1597  003670  005737  000042                           TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
1598  003674  001006                                   BNE     69$             ;;BRANCH IF YES
1599  003676  023727  001140  000176                   CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
1600  003704  001005                                   BNE     70$             ;;BRANCH IF NO
1601  003706  104405                                   GTSWR                   ;;GET SOFT-SWR SETTINGS
1602  003710  000403                                   BR      70$
1603  003712  112737  000001  001134           69$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
1604  003720                                   70$:
```

```
 1605  003720  000433                                          BR      67$                ;;GET OVER THE ASCIZ
 1606                                              ;;68$:  .ASCIZ  <CRLF>*MAINDEC-11-DZRKH-E RK11/RK05 PERFORMANCE EXERCISER*<CRLF>
 1607  004010                                      67$:
 1608  004010  012737  025634  000030  06900               MOV     #$ERROR,@#EMTVEC           ;EMT VECTOR FOR ERROR ROUTINE
 1609  004016  013737  001744  000032  06910               MOV     PPRLVL,@#EMTVEC+2          ;LEVEL 5
 1610  004024  012737  022202  000100  06920               MOV     #KWSRVE,@#KWLVEC           ;KW11L CLOCK SERVICE
 1611  004032  013737  001746  000102  06930               MOV     KWPLVL,@#KWLVEC+2          ;LEVEL 7
 1612                                    06940
```

```
 1613                                          06960   ;THE FOLLOWING CODE FINDS OUT THE PROGRAM CONTROL MODE:
 1614                                          06970   ;PAPER TAPE (MANUAL), ACT11, RKDP CHAIN OR DUMP
 1615                                          06980
 1616                                          06990
 1617  004040  005737  000042                 07000           TST     @#42            ;IS LOC 42 0?
 1618  004044  001010                         07010           BNE     1$              ;YES, BRANCH
 1619  004046  123727  000041  000002         07020           CMPB    @#41,#2         ;DOES BYTE 41 CONTAIN 2?
 1620  004054  001413                         07030           BEQ     2$              ;YES, IT IS RKDP DUMP MODE
 1621                                          07040                                   ;NO, PROGRAM LOADED BY PAPER TP
 1622  004056  112737  000001  002562         07050           MOVB    #1,PPTP         ;SET FLAG INDICATING PPR TPE
 1623  004064  000430                         07060           BR      ST2
 1624  004066  123727  000041  000002         07070   1$:     CMPB    @#41,#2         ;DOES BYTE 41 CONTAIN 2?
 1625  004074  001413                         07080           BEQ     3$              ;YES, RKDP CHAIN MODE
 1626  004076  105737  000041                 07090           TSTB    @#41            ;BYTE41 0?
 1627  004102  001416                         07100           BEQ     4$              ;ACT 11
 1628                                          07110
 1629  004104  104400  003075                 07120   2$:     TYPE    .MSG21
 1630  004110  104407                         07130           RDCHR
 1631                                          07140
 1632  004112  005726                         07150           TST     (R6)+           ;POP UP STACK
 1633  004114  112737  000001  002560         07160           MOVB    #1,RKDPDU       ;SET FLAG INDICATING RKDP DUMP MODE
 1634  004122  000411                         07170           BR      ST2
 1635                                          07180
 1636  004124  104400  003166                 07190   3$:     TYPE    ,MSG23
 1637  004130  112737  000001  002561         07200           MOVB    #1,RKDPCH       ;SET FLAG INDICATING RKDP CHAIN MODE
 1638  004136  000403                         07210           BR      ST2
 1639                                          07220
 1640  004140  112737  000001  002563         07230   4$:     MOVB    #1,ACT11        ;SET FLAG INDICATING ACT11 MODE
```

```
 1641  004146  104415                    07250  S12:    CON.RESET
 1642  004150  005037  001764            07260          CLR     DRVPRS          ;FIND WHICH DRIVE #'S ARE PRESENT
 1643  004154  005000                    07270          CLR     R0
 1644  004156  005002                    07280          CLR     R2
 1645  004160  005037  001754            07290          CLR     PDR             ;CLEAR DRIVES PRESENT TABLE
 1646  004164  005037  001756            07300          CLR     PDR+2
 1647  004170  005037  001760            07310          CLR     PDR+4
 1648  004174  005037  001762            07320          CLR     PDR+6
 1649  004200  012701  001754            07330          MOV     #PDR,R1
 1650  004204  012703  002006            07340          MOV     #KEY,R3
 1651  004210  105737  002561            07350          TSTB    RKDPCH          ;DONT USE DRIVE ZERO IN CHAIN MODE
 1652  004214  001016                    07360          BNE     3$
 1653  004216  010277  175506            07370  1$:     MOV     R2,@RKDA        ;SELECT A DRIVE
 1654  004222  032777  000200  175466    07380          BIT     #200,@RKCS      ;IS IT IN SYSTEM?
 1655  004230  001410                    07390          BEQ     3$              ;NO
 1656                                    07400
 1657  004232  012777  000015  175462    07410          MOV     #15,@RKCS       ;PERFORM A DRIVE RESET
 1658  004240  104416                    07420          CON.RDY
 1659  004242  110021                    07430  2$:     MOVB    R0,(R1)+        ;STORE THE DRIVE NUMBER
 1660  004244  010223                    07440          MOV     R2,(R3)+        ;STORE ADDRESS IN KEY TABLE
 1661  004246  005237  001764            07450          INC     DRVPRS          ;BUMP THE NUMBER OF DRIVES COUNTER
 1662                                    07460
 1663  004252  062702  020000            07470  3$:     ADD     #20000,R2       ;NEXT DRIVE ADDRESS
 1664  004256  005200                    07480          INC     R0              ;NEXT DRIVE NUMBER
 1665                                    07490
 1666  004260  022700  000010            07500          CMP     #8.,R0          ;DONE ALL DRIVES???
 1667  004264  001354                    07510          BNE     1$              ;LOOP TILL DONE
 1668                                    07520
 1669  004266  013703  001764            07530          MOV     DRVPRS,R3       ;FIND WHICH DRIVES ARE TYPE F
 1670  004272  012701  001754            07540          MOV     #PDR,R1
 1671  004276  005000                    07550          CLR     R0
 1672  004300  005002                    07560          CLR     R2
 1673  004302  104400  003065            07570          TYPE    ,MSG20
 1674                                    07580
 1675  004306  005703                    07590          TST     R3              ;ANY DRIVES TO DO
 1676                                    07600
 1677  004310  111102                    07610  4$:     MOVB    (R1),R2         ;GET DRIVE NUMBER
 1678  004312  010200                    07620          MOV     R2,R0
 1679  004314  002472                    07630          BLT     9$
 1680                                    07640
 1681  004316  104400  003207            07650          TYPE    ,MSG24
 1682                                    07660
 1683  004322  010246                    07670          MOV     R2,-(SP)        ;TYPE THE DRIVE NUMBER
 1684  004324  104402                    07680          TYPOS
 1685  004326     001                    07690          .BYTE   1
 1686  004327     000                    07700          .BYTE   0
 1687                                    07710
 1688  004330  000241                    07720          CLC                     ;MOVE DRIVE NUMBER TO BITS 15,14,13
 1689  004332  006002                    07730          ROR     R2              ;BIT0 TO CARRY
 1690  004334  006002                    07740          ROR     R2              ;BIT0 TO BIT15
 1691  004336  006002                    07750          ROR     R2              ;BIT0 TO BIT14
 1692  004340  006002                    07760          ROR     R2              ;BIT0 TO BIT13
 1693  004342  042702  017777            07770          BIC     #17777,R2       ;CLEAR ANY EXTRANEOUS BITS
 1694                                    07780
 1695  004346  010237  002202            07790          MOV     R2,@DRV
 1696  004352  104417                    07800          DRV.RESET               ;RESET THE DRIVE VIA @DRV
```

```
1697                                                    07810
1698  004354  032702  020000                            07820          BIT     #20000,R2       ;EVEN DRIVE NUMBER???
1699  004360  001003                                    07830          BNE     5$              ;NO - CLEAR BIT
1700                                                    07840
1701  004362  052702  020000                            07850          BIS     #20000,R2       ;MAKE IT AN ODD DRIVE
1702  004366  000402                                    07860          BR      6$
1703                                                    07870
1704  004370  042702  020000                            07880  5$:     BIC     #20000,R2       ;MAKE IT AN EVEN DRIVE
1705                                                    07890
1706  004374  010277  175330                            07900  6$:     MOV     R2,@RKDA        ;SELECT THE NEW DRIVE
1707  004400  032777  000200  175310                    07910          BIT     #200,@RKDS      ;MAKE SURE DRIVE IS IN SYSTEM
1708  004406  001420                                    07920          BEQ     8$              ;IF NOT, SKIP THIS TEST
1709                                                    07930
1710  004410  012777  000011  175304                    07940          MOV     #11,@RKCS       ;START A SEEK TO CYL 0
1711  004416  104416                                    07950          CON.RDY                 ;WAIT FOR CONTROLLER
1712  004420  032777  000100  175270                    07960          BIT     #100,@RKDS      ;IS IT IN MOTION???
1713  004426  001010                                    07970          BNE     8$              ;NO - J TYPE DRIVE
1714                                                    07980
1715  004430  152711  000200                            07990          BISB    #200,(R1)       ;YES - SET THE F TYPE BIT
1716  004434  104400  003212                            08000          TYPE    ,MSG25
1717                                                    08010
1718  004440  032777  000100  175250                    08020  7$:     BIT     #100,@RKDS      ;WAIT FOR HEADS TO STOP
1719  004446  001774                                    08030          BEQ     7$
1720                                                    08040
1721  004450  105737  001753                            08050  8$:     TSTB    $RSTRT
1722  004454  001012                                    08060          BNE     9$
1723                                                    08070
1724  004456  032777  000002  174454                    08080          BIT     #SW1,@SWR       ;SERIAL NO. SW SET?
1725  004464  001406                                    08090          BEQ     9$              ;NO
1726                                                    08100
1727  004466  104400  003030                            08110          TYPE    ,MSG17          ;TYPE "SR  NO"
1728  004472  104412                                    08120          RDDEC                   ;READ FROM TTY INPUT
1729  004474  006300                                    08130          ASL     R0              ;SAVE SERIAL NO FOR THE DRIVE
1730  004476  012660  001766                            08140          MOV     (SP)+,SRNO(R0)
1731                                                    08150
1732  004502  005201                                    08160  9$:     INC     R1
1733  004504  005303                                    08170          DEC     R3
1734  004506  003300                                    08180          BGT     4$
1735  004510  104400  001213                            08190          TYPE    ,$CRLF
```

```
1736                                    08210   ;'MAXBA' IS THE HIGHEST BUS ADDRESS (MEMORY) TO WHICH DATA TRANSFERS CAN
1737                                    08220   ;BE DONE BY THE PROGRAM.
1738                                    08230   ;'MAXBA' IS FIGURED USING THE FOLLOWING ALGORITHM:
1739                                    08240
1740                                    08250   ;1.  IF KT11 IS NOT PRESENT,
1741                                    08260   ;A.  AND THE PROGRAM IS RUN UNDER XXDP, THEN THE TOP 1.5 K IS RESERVED
1742                                    08270   ;AND THE  'MAXBA' IS COMPUTED ($LSTAD-6000).
1743                                    08280   ;B.  AND THE PROGRAM IS NOT RUNNING UNDER XXDP, THEN THE TOP 320 WORDS
1744                                    08290   ;ARE RESERVED FOR 'MOM',LOADER,ETC. AND THE 'MAXBA' IS COMPUTED ($LSTAD-500).
1745                                    08300
1746                                    08310   ;2. IF KT11 IS PRESENT,
1747                                    08320   ;A.  AND MORE THAN 28K MEMORY IS PRESENT, THEN THE MAXIMUM BUS ADDRESS
1748                                    08330   ;IS 147776 (OCTAL).
1749                                    08340   ;B.  AND LESS THAN 28K IS PRESENT, THEN THE TOP 2K IS RESERVED FOR RKDP
1750                                    08350   ;MONITOR AND 'MAXBA' IS COMPUTED.
1751                                    08360   ;FIGURE OUT THE AVAILABLE MEMORY AND 'MAXBA'
1752                                    08370
1753  004514  004737  017132              08380   ST4:    JSR     PC,$SIZE        ;GO SIZE THE MEMORY
1754  004520  012702  002552              08390           MOV     #BASEBA,R2      ;INITIALIZE POINTERS
1755  004524  012703  002554              08400           MOV     #MAXBA,R3
1756  004530  005737  017216              08410           TST     $KT11           ;KT11 AVAILABLE?
1757  004534  100022                      08420           BPL     4$              ;NO
1758  004536  013700  017464              08430           MOV     $LSTBK,R0       ;GET THE LAST BANK OF MEMORY
1759  004542  020027  001540              08440           CMP     R0,#1540        ;28K OR MORE?
1760  004546  002012                      08450           BGE     3$              ;YES
1761                                    08460
1762  004550  162700  000040              08470           SUB     #40,R0          ;BACK UP 2 K'S (RKDP MONITOR, ETC.)
1763  004554  012701  177772              08480           MOV     #-6,R1          ;AND FORM THE MAXIMUM BUS ADDRESS
1764                                    08490                                   ;FOR DATA TRANSFER
1765  004560  006300                      08500   1$:     ASL     R0
1766  004562  005201                      08510           INC     R1
1767  004564  001375                      08520           BNE     1$
1768  004566  162700  000002              08530           SUB     #2,R0
1769  004572  000415                      08540   2$:     BR      6$
1770                                    08550
1771  004574  012713  147776              08560   3$:     MOV     #147776,(R3)    ;FOR 28K OR MORE, THIS IS THE 'MAXBA'
1772  004600  000413                      08570           BR      7$
1773                                    08580
1774  004602  013700  017462              08590   4$:     MOV     $LSTAD,R0       ;KT11 NOT PRESENT, GET THE LAST
1775                                    08600                                   ;AVAILABLE ADDRESS
1776                                    08610
1777  004606  005737  002560              08620   5$:     TST     RKDPDU          ;RKDP DUMP OR CHAIN MODE?
1778  004612  001003                      08630           BNE     8$              ;YES
1779  004614  162700  000500              08640           SUB     #500,R0         ;NO, SAVE THE LAST 320 WORDS
1780  004620  000402                      08650           BR      6$
1781  004622  162700  006000              08660   8$:     SUB     #6000,R0        ;SAVE THE LAST 1.5K OF MEMORY (RKDP
1782                                    08670                                   ;MONITOR, ETC.)
1783  004626  010013                      08680   6$:     MOV     R0,(R3)         ;SAVE THE MAXIMUM BUS ADDRESS(MAXBA) TO
1784                                    08690                                   ;WHICH DATA TRANSFER CAN BE DONE SAFELY
1785  004630  012712  032106              08700   7$:     MOV     #PGEND,(R2)     ;'BASEBA'
1786  004634  032777  000100  174276      08710           BIT     #SW06,@SWR
1787  004642  001510                      08720           BEQ     ST3
1788  004644  104400  004652                              TYPE    ,65$            ;;TYPE ASCIZ STRING
1789  004650  000432                                      BR      64$             ;;GET OVER THE ASCIZ
1790                                          ;;65$:  .ASCIZ  <15><12>/TYPE OCTAL BUS ADDRESSES FOR DATA XFER, BETWEEN
1791  004736                                  64$:
```

```
1792  004736  011246                  08740          MOV     (R2),-(SP)      ;'BASEBA'
1793  004740  104401                  08750          TYPOC
1794  004742  104400  004750                         TYPE    ,67$            ;;TYPE ASCIZ STRING
1795  004746  000402                                 BR      66$             ;;GET OVER THE ASCIZ
1796                                          ;;67$:  .ASCIZ  / 8 /
1797  004754                                  66$:
1798  004754  011346                  08770          MOV     (R3),-(SP)      ;'MAXBA'
1799  004756  104401                  08780          TYPOC
1800  004760                          08790  9$:
1801  004760  104400  004766                         TYPE    ,69$            ;;TYPE ASCIZ STRING
1802  004764  000407                                 BR      68$             ;;GET OVER THE ASCIZ
1803                                          ;;69$:  .ASCIZ  <15><12>/LO LIMIT? /
1804  005004                                  68$:
1805  005004  104411                  08800          RDOCT
1806  005006  012600                  08810          MOV     (SP)+,R0
1807  005010  020012                  08820          CMP     R0,(R2)         ;CORRECT LO LIMIT?
1808  005012  103762                  08830          BLO     9$
1809  005014  020013                  08840          CMP     R0,(R3)         ;CORRECT LO LIMIT?
1810  005016  103360                  08850          BHIS    9$
1811  005020  010012                  08860          MOV     R0,(R2)         ;'BASEBA'
1812  005022                          08870  10$:
1813  005022  104400  005030                         TYPE    ,71$            ;;TYPE ASCIZ STRING
1814  005026  000407                                 BR      70$             ;;GET OVER THE ASCIZ
1815                                          ;;71$:  .ASCIZ  <15><12>/HI LIMIT? /
1816  005046                                  70$:
1817  005046  104411                  08880          RDOCT
1818  005050  012600                  08890          MOV     (SP)+,R0
1819  005052  020013                  08900          CMP     R0,(R3)         ;CORRECT HI LIMIT?
1820  005054  101362                  08910          BHI     10$
1821  005056  020012                  08920          CMP     R0,(R2)         ;CORRECT LO LIMIT?
1822  005060  101760                  08930          BLOS    10$
1823  005062  010013                  08940          MOV     R0,(R3)         ;'MAXBA'
1824                                  08950
1825  005064  023727  002554  037476  08960  ST3:    CMP     MAXBA,#37476    ;8K MEMORY - CLOBBER XXDPT
1826  005072  002003                  08970          BGE     1$
1827  005074  012737  037476  002554  08980          MOV     #37476,MAXBA    ;BUT SAVE LOADER
1828  005102  105737  001753          08990  1$:     TSTB    FRSTRT          ;PROGRAM RESTARTED AT 210?
1829  005106  001402                  09000          BEQ     BCTST           ;NO
1830  005110  000137  007534          09010          JMP     EXRCSR          ;YES, SKIP TEST 1 TO 7
```

```
 1831                                          09030   ;THIS IS THE BEGINING OF THE CONSTRAINED TESTS AIMED AT CHECKING THE
 1832                                          09040   ;DIFFERENT BOUNDARY CONDITIONS OF RK11/RK05
 1833                                          09050
 1834                                          09060   ;FIND OUT THE DRIVE NUMBER TO BE TESTED.
 1835  005114  012737  002006  002176         09070   BCTST:  MOV     #KEY,DRVPTR     ;INITIALIZE PTR TO DRV#
 1836  005122  013737  001764  002200         09080           MOV     DRVPRS,DRVCNT   ;NUMBER OF DRIVES PRESENT
 1837  005130  017737  175042  002202         09090   NXTDRV: MOV     @DRVPTR,GDRV    ;SAVE DRIVE #(BITS 15-13)
 1838  005136  062737  000002  002176         09100           ADD     #2,DRVPTR       ;INCRMENT PTR TO NXT DRV#
 1839  005144  005337  002200                 09110           DEC     DRVCNT          ;DONE ALL DRIVES?
 1840  005150  100002                         09120           BPL     TST1            ;NO, GO TEST THIS DRIVE
 1841  005152  000137  007534                 09130           JMP     EXRCSR          ;ALL DONE, GO TO EXERCISER PART
```

```
1842                                        ;;*****************************************************************
1843                                        ;*TEST 1        PERFORM WRITE OF 401 WORDS (1 SECTOR + 1 WORDS)
1844                                        ;THIS TEST PERFORMS A WRITE OF 401 WORDS (1 SECTOR + 1WORD) AND
1845                                        ;CHECKS IF RKDA,RKBA,RKWC INCREMENTED CORRECTLY.WRITING IS DONE
1846                                        ;ON CYLINDER 0, SURFACE 0, SECTORS 0,1 AND 10,11. IT SHOULD BE
1847                                        ;NOTED THAT THIS IS A BOUNDARY CONDITION TRANSFER. THE VALIDITY
1848                                        ;OF THE TRANSFER IS CHECKED IN THE NEXT TEST.
1849                                        ;DATA PATTERN WRITTEN IS 111111.
1850                                        ;;*****************************************************************
1851  005156  000004                       TST1:   SCOPE
1852                                09240
1853  005160  013701  002202       09250           MOV     @DRV,R1         ;GET RKDA
1854  005164  010102               09260           MOV     R1,R2           ;SAVE RKDA
1855  005166  062702  000002       09270           ADD     #2,R2           ;EXPCTD RKDA AFTER WRITE IS DONE
1856                                09280
1857  005172  012737  005200  001110  09290         MOV     #1$,$LPERR      ;RETURN ADDRESS FOR LUPING
1858                                09300
1859  005200  104415               09310   1$:     CON.RESET
1860  005202  104417               09320           DRV.RESET
1861  005204  104415               09330           CON.RESET               ;CLEAR MASK BITS IN POLLING LOGIC
1862  005206  012737  111111  032106  09340   2$:     MOV     #111111,@DBUF   ;PATTERN TO BE WRITTEN
1863  005214  012703  000401       09350           MOV     #401,R3         ;WORD COUNT FOR WRITE
1864  005220  004737  006046       09360           JSR     PC,@DOWRITE     ;GO DO WRITE
1865                                09370
1866  005224  104101               09380           ERROR   !01             ;INTERRUPT DID NOT OCCUR AFTER WRITE
1867  005226  004737  017604       09390           JSR     PC,CHKCS        ;CHECK ERROR BIT IN RKCS
1868  005232  104102               09400           ERROR   102             ;ERROR BIT IN RKCS SET ON DOING WRITE
1869  005234  004737  017620       09410           JSR     PC,CHKDA        ;CHECK IF RKDA INCREMENTED RIGHT
1870  005240  104103               09420           ERROR   103             ;RKDA DID NOT INCREMENT RIGHT AFTER
1871                                09430                                   ;A WRITE OF 401 WORDS.
1872  005242  004737  017722       09440           JSR     PC,CHKWC        ;CHECK IF RKWC OVERFLOWED TO 0
1873  005246  104105               09450           ERROR   105             ;RKWC DID NOT OVERFLOW TO 0 AFTER
1874                                09460                                   ;A WRITE OF 401 WORDS.
1875  005250  032701  000010       09470           BIT     #10,R1          ;SECTORS 10,11 WRITTEN?
1876  005254  001005               09480           BNE     TST2            ;YES
1877  005256  062701  000012       09490           ADD     #12,R1          ;RKDA TO BE USED NEXT (SEC 10)
1878  005262  062702  000016       09500           ADD     #16,R2          ;EXPCTD RKDA AFTER WRITE IS DONE
1879  005266  000747               09510           BR      2$              ;GO WRITE SECS 10,11
```

```
                                        ;;*****************************************************************
                                        ;*TEST 2        READ & CHECK THAT 401 WORD WRITE WAS DONE CORRECTLY
                                        ;THIS TEST PERFORMS A READ OF THE 401 WORDS WRITTEN IN THE
                                        ;PREVIOUS TEST AND CHECKS THAT THEY WERE CORRECTLY READ.MOREOVER
                                        ;IT CHECKS THAT ONLY ONE NON-ZERO WORD (401TH) WAS WRITTEN IN THE
                                        ;SECOND SECTOR AND THE REST OF THE WORDS ARE ALL ZEROS.
                                        ;;*****************************************************************
005270  000004                  TST2:   SCOPE
                        09600
005272  013701  002202          09610           MOV     QDRV,R1         ;GET DRIVE #
005276  012737  005304  001110  09620           MOV     #1$,$LPERR      ;ADDRESS FOR LUPING ON EROR
005304  104415                  09630   1$:     CON.RESET
005306  104417                  09640           DRV.RESET
005310  104415                  09650           CON.RESET
                                09660
                                09670                                   ;CLEAN UP THE DATA BUFFER
005312  004737  006176          09680           JSR     PC,CLEANBUF     ;INTO WHICH READ WILL
                                09690                                   ;BE DONE
005316  012703  001000          09700           MOV     #1000,R3        ;WORD COUNT
                                09710
005322  004737  006166          09720           JSR     PC,DOREAD       ;GO DO A READ OF 2 SECTORS
                                09730                                   ;FROM DISK ADDRESS GIVEN IN R1
005326  104101                  09740           ERROR   101             ;INTERRUPT DID NOT OCCUR AFTER
                                09750                                   ;READ OF 401 WORDS WAS DONE.
005330  004737  017604          09760           JSR     PC,CHKCS        ;CHECK IF EROR BIT IN RKCS SET?
005334  104102                  09770           ERROR   102             ;EROR BIT IN RKCS SET ON DOING A
                                09780                                   ;READ OF 401 WORDS.
005336  012704  032106          09790           MOV     #DBUF,R4        ;STARTING BUS ADDRESS, INTO WHICH READ
005342  010402                  09800           MOV     R4,R2           ;WAS DONE
005344  004737  017642          09810           JSR     PC,CHKBA        ;CHECK IF RKBA INCREMENTED RIGHT
005350  104104                  09820           ERROR   104             ;RKBA DID NOT INCREMENT RIGHT AFTER READ
                                09830                                   ;OF 401 WORDS.
005352  012705  177764          09840           MOV     #-14,R5         ;ALLOW 12 ERRORS, AT THE MOST
005356  022712  111111          09850   2$:     CMP     #111111,(R2)    ;CORRECT DATA READ?
005362  001410                  09860           BEQ     3$              ;YES
005364  012737  111111  001164  09870           MOV     #111111,$REG1   ;GET EXPCTD DATA WORD
005372  004737  005462          09880           JSR     PC,ERINF1       ;GET ERROR INFORMATION
005376  104100                  09890           ERROR   100             ;DATA ERROR OCCURRED WHEN A
                                09900                                   ;READ OF 401 WORDS WAS DONE
                                09910                                   ;THE DISK ADDRESS FROM WHERE
                                09920                                   ;THE DATA WAS READ INCORRECTLY
                                09930                                   ;IS GIVEN IN THE ERROR MESSAGE
                                09940
005400  005205                  09950           INC     R5              ;REPORT 12 ERORS AT MOST
005402  001421                  09960           BEQ     6$
005404  005722                  09970   3$:     TST     (R2)+           ;INCREMENT POINTER
005406  020227  033110          09980           CMP     R2,#DBUF+1002   ;CHECKED ALL 401 WORDS?
005412  001361                  09990           BNE     2$
                                10000
005414  005712                  10010   4$:     TST     (R2)            ;CHECK THAT REST OF 377 WORDS
005416  001407                  10020           BEQ     5$              ;ARE ALL 0'S
005420  005037  001164          10030           CLR     $REG1           ;GET EXPCTD DATA WORD (0)
005424  004737  005462          10040           JSR     PC,ERINF1       ;GET ERROR INFO
005430  104100                  10050           ERROR   100             ;DATA ERROR.  IN A PREVIOUS
                                10060                                   ;TEST A WRITE OF 401 WORDS
                                10070                                   ;(1 SECTOR + 1 WORD) WAS DONE
```

```
 1936                                   10080                                  ;NOW THESE 2 SECTORS WERE
 1937                                   10090                                  ;READ IN THE SECOND SECTOR
 1938                                   10100                                  ;THE FIRST WORD IS A NON-ZERO
 1939                                   10110                                  ;WORD (WHICH WAS WRITTEN BEFORE)
 1940                                   10120                                  ;THE REST OF 377 WORD
 1941                                   10130                                  ;SHOULD BE ALL ZEROS, IF
 1942                                   10140                                  ;THE WRITE WAS DONE CORRECTLY
 1943                                   10150                                  ;(& READ IS DONE CORRECTLY)
 1944                                   10160
 1945  005432  005205                   10170          INC     R5              ;REPORT 12 ERORS AT MOST
 1946  005434  001404                   10180          BEQ     6$
 1947  005436  005722                   10190    5$:   TST     (R2)+           ;ALL WORDS CHECKED?
 1948  005440  020227  034106           10200          CMP     R2,#DBUF+2000
 1949  005444  001363                   10210          BNE     4$              ;IF NOT GO BAK
 1950                                   10220
 1951  005446  032701  000010           10230    6$:   BIT     #10,R1          ;WERE SECTORS 10,11 READ
 1952  005452  001030                   10240          BNE     TST3            ;YES
 1953  005454  062701  000012           10250          ADD     #12,R1          ;FROM NEW RKDA, SEC 10
 1954  005460  000711                   10260          BR      1$              ;GO BACK AND READ FROM SECS 10,11
 1955                                   10270
 1956                                   10280    ;ERINF1
 1957                                   10290    ;AT THE TIME OF ENTRY:
 1958                                   10300    ;R2 CONTAINS ERRORING BUS ADDRESS (WHERE DATA ERROR OCCURRED).
 1959                                   10310    ;(R2) CONTAINS BAD DATA THAT WAS READ BACK FROM DISK.
 1960                                   10320    ;R1 CONTAINS DISK ADDRESS WHERE READ BEGAN.
 1961                                   10330
 1962  005462  010237  001162           10340    ERINF1: MOV     R2,$REG0        ;GET BUS ADDRESS OF DATA ERROR
 1963  005466  011237  001166           10350            MOV     (R2),$REG2      ;GET BAD DATA WORD (READ)
 1964  005472  010146                   10360            MOV     R1,-(SP)
 1965  005474  020227  033104           10370            CMP     R2,#DBUF+776    ;FIGURE OUT THE DISK ADDRESS
 1966  005500  003001                   10380            BGT     1$              ;WHERE DATA ERROR OCCURRED
 1967  005502  005316                   10390            DEC     (SP)
 1968  005504  005216                   10400    1$:   INC     (SP)
 1969  005506  032716  000010           10410          BIT     #10,(SP)
 1970  005512  001405                   10420          BEQ     2$
 1971  005514  032716  000004           10430          BIT     #4,(SP)
 1972  005520  001402                   10440          BEQ     2$
 1973  005522  062716  000004           10450          ADD     #4,(SP)
 1974  005526  012637  001170           10460    2$:   MOV     (SP)+,$REG3
 1975  005532  000207                   10470          RTS     PC
```

```
                                        ;;*****************************************************************
                                        ;*TEST 3        PERFORM WRITE OF 12 SECTORS + 1 WORD
                                                ;THIS TEST CHECKS FOR ANOTHER BOUNDARY CONDITION.  IT
                                                ;PERFORMS A WRITE OF 12 SECTORS + 1 WORD.  RKDA,RKBA,
                                                ;RKWC ARE CHECKED TO SEE IF THEY ARE INCREMENTED CORRECTLY.
                                                ;VALIDITY OF THE DATA WRITTEN IS CHECKED IN THE NEXT
                                                ;TEST.  DATA IS WRITTEN ON SECTORS 0-11, SURFACE 0
                                                ;CYLINDER 0 (6001TH WORD ON SECTOR 0, SURFACE 1, CYL 0).
                                                ;ALSO ON SECTORS 0-11, SURFACE 1 (6001TH WORD ON SECTOR
                                                ;0, CYL 1)
                                        ;;*****************************************************************
005534  000004                        TST3:   SCOPE
005536  013701  002202          010600        MOV     @DRV,R1         ;GET DRIVE #
005542  012737  005562  001110  010610        MOV     #1$,$LPERR      ;LUP ON EROR TO '1$'
005550  010102                  010620        MOV     R1,R2
005552  062702  000021          010630        ADD     #21,R2
005556  012703  006001          010640        MOV     #6001,R3
005562  104415                  010650  1$:   CON.RESET
005564  104417                  010660        DRV.RESET
005566  104415                  010670        CON.RESET
                                010680
                                010690
005570  012737  044444  032106  010700        MOV     #44444,DBUF     ;PATTERN TO BE WRITTEN
                                010710
005576  004737  006046          010720        JSR     PC,DOWRITE      ;GO DO WRITE
                                010730
005602  104101                  010740        ERROR   101             ;INTERUPT DID NOT OCCUR ON
                                010750                                ;COMPLETION OF WRITE
005604  004737  017604          010760        JSR     PC,CHKCS        ;CHECK IF EROR BIT IN RKCS SET
005610  104102                  010770        ERROR   102             ;EROR BIT IN RKCS SET ON DOING WRITE
005612  004737  017620          010780        JSR     PC,CHKDA        ;CHECK IF RKDA INCREMENTED RIGHT
005616  104103                  010790        ERROR   103             ;RKDA DID NOT INCREMENT CORRECTLY
                                010800                                ;AFTER A WRITE OF 6001 (OCTAL) WORDS
                                010810                                ;(12 SECTORS + 1)
005620  004737  017722          010820        JSR     PC,CHKWC        ;CHECK IF RKWC OVERFLOWED TO 0
005624  104105                  010830        ERROR   105             ;RKWC DID NOT OVERFLOW TO 0
005626  032701  000020          010840        BIT     #20,R1          ;WRITTEN ON SURFACE 1?
005632  001006                  010850        BNE     TST4            ;YES
005634  010201                  010860        MOV     R2,R1
005636  062702  000020          010870        ADD     #20,R2          ;SURFACE 1
005642  012703  005401          010880        MOV     #5401,R3        ;WORD COUNT
005646  000745                  010890        BR      1$              ;GO WRITE SURFACE 1
```

```
                                        ;;*****************************************************************
                                        ;*TEST 4        READ & CHECK THAT 6001 WORD WRITE WAS DONE CORRCTLY
                                                ;THIS TEST CHECKS THAT THE 6001-WORD WRITE THAT WAS DONE IN THE
                                                ;PREVIOUS TEST WAS CORRECT, ESPECIALLY THE LAST 401 WORDS. THE
                                                ;FIRST WORD OF THE SECTOR( IN WHICH THE 6001TH WORD) IS WRITTEN
                                                ;IS THE ONLY NON-ZERO WORD IN THAT SECTOR, THE REST 277 WORDS ARE
                                                ;ALL ZEROS, IF THE WRITING WAS DONE CORRECTLY.
                                        ;;*****************************************************************
005650  000004                        TST4:   SCOPE
005652  013701  002202          10990         MOV     @DRV,R1         ;GET DRIVE #
005656  062701  000013          11000         ADD     #13,R1          ;DISK ADDRESS FROM WHERE READ IS DONE
005662  012737  005670  001110  11010         MOV     #1$,$LPERR
005670  104415                  11020  1$:    CON.RESET
005672  104417                  11030         DRV.RESET
005674  104415                  11040         CON.RESET
                                11050
005676  004737  006176          11060         JSR     PC,CLEANBUF     ;CLEAN UP THE BUFFER INTO WHICH
                                11070                                 ;READ WILL BE DONE
                                11080                                 ;SET UP RKDA
                                11090                                 ;SECTOR 11, SURFACE 0
005702  012703  001000          11100         MOV     #1000,R3        ;WORD COUNT
                                11110
005706  004737  006166          11120         JSR     PC,DOREAD       ;GO READ 1000 WORDS (2 SECS)
005712  104101                  11130         ERROR   101             ;INTERRUPT DID NOT OCCUR AFTER
                                11140                                 ;COMPLETION OF READ
005714  004737  017604          11150         JSR     PC,CHKCS        ;CHECK IF EROR BIT IN RKCS SET
005720  104102                  11160         ERROR   102             ;EROR (RKCS) SET ON DOING READ
005722  012704  032106          11170         MOV     #DBUF,R4        ;STARTING BA OF DATA BUFER
005726  010402                  11180         MOV     R4,R2
005730  004737  017642          11190         JSP     PC,CHKBA        ;RKBA INCREMENTED CORRECTLY?
005734  104104                  11200         ERROR   104             ;RKBA DID NOT INCREMENT CORRECTLY
005736  012705  177764          11210         MOV     #-14,R5
005742  022712  044444          11220  2$:    CMP     #44444,(R2)     ;DATA WORD OK?
005746  001410                  11230         BEQ     3$
005750  012737  044444  001164  11240         MOV     #44444,$REG1    ;NO, GET EXPCTD DATA WORD
005756  004737  005462          11250         JSR     PC,ERINF1       ;GET, OTHER ERROR INFO
005762  104100                  11260         ERROR   100             ;DATA ERROR.  A WRITE OF 6001
                                11270                                 ;WORDS (12 SECS + 1 WORD) WAS DONE
                                11280                                 ;IN A PREVIOUS TEST.  THE LAST TWO
                                11290                                 ;SECTORS (LAST 401 WORDS) WERE READ
                                11300                                 ;BACK.  THIS ERROR INDICATES THAT
                                11310                                 ;SEC #11 (LAST BUT ONE SECTOR) GAVE
                                11320                                 ;BAD DATA WORDS
005764  005205                  11330         INC     R5              ;REPORT 12 ERORS AT MOST
005766  001421                  11340         BEQ     6$
005770  005722                  11350  3$:    TST     (R2)+           ;INCREMENT POINTER TO BA
005772  020227  033110          11360         CMP     R2,#DBUF+1002   ;CHECKED 401 WORDS?
005776  001361                  11370         BNE     2$
006000  005712                  11380  4$:    TST     (R2)            ;CHECK THAT THE REMAINING 377
006002  001407                  11390         BEQ     5$              ;WORDS OF THE LAST SECTOR (SEC #0)
006004  005037  001164          11400         CLR     $REG1           ;WERE READ BACK AS 0'S
006010  004737  005462          11410         JSR     PC,ERINF1
006014  104100                  11420         ERROR   100             ;DATA ERROR.  IF WRITE WAS DONE CORRECTLY
                                11430                                 ;IN THE PREVIOUS TEST, THE LAST SECTOR
                                11440                                 ;OF THE DATA BLOCK (12 SECS + 1 WORD)
                                11450                                 ;SHOULD CONTAIN ONLY 1 (FIRST) WORD
```

```
2074                                    11460                                   ;AS NON-ZERO, THE REST 377 SHOULD BE
2075                                    11470                                   ;ALL 0'S.  THIS ERROR INDICATES THAT
2076                                    11480                                   ;THE SOME OF 377 WORDS
2077                                    11490                                   ;WERE NOT CORRECT
2078  006016  005205                    11500           INC     R5              ;REPORT 12 ERORS AT MOST
2079  006020  001404                    11510           BEQ     6$
2080  006022  005722                    11520   5$:     TST     (R2)+           ;INCREMENT POINTER
2081  006024  020227  034106            11530           CMP     R2,#DBUF+2000   ;CHECKED ALL WORDS?
2082  006030  001363                    11540           BNE     4$              ;NO
2083                                    11550
2084  006032  032701  000020            11560   6$:     BIT     #20,R1          ;DONE CHECKING FOR SURFACE 1?
2085  006036  001070                    11570           BNE     .ST5            ;YES
2086  006040  062701  000020            11580           ADD     #20,R1          ;SO SET UP FOR SURFACE 1
2087  006044  000711                    11590           BR      1$              ;GO BACK & READ SURFACE 1
```

```
2088                                        11610   ;DOWRITE
2089                                        11620   ;THIS ROUTINE PERFORMS A WRITE ON A DISK.AT THE TIME OF ENTRY, R1 CONTAINS
2090                                        11630   ;DISK ADDRESS (RKDA) WHERE WRITE IS TO BE DONE, R3 CONTAINS THE WORD COUNT
2091                                        11640   ;(RKWC), 'DBUF' CONTAINS THE DATA TO BE WRITTEN. NOTE IBA BIT IS SET.
2092                                        11650
2093                                        11660   ;WRITE IS DONE IN INTERRUPT MODE. IF THE INTERRUPT DOES NOT OCCUR WITHIN
2094                                        11670   ;A CERTAIN TIME, RETURN IS MADE TO THE ERROR MESSAGE FOLLOWING THE 'JSR'
2095                                        11680   ;CALL. IF THE INTERRUPT OCCURS, RETURN ADDRESS IS ADJUSTED TO SKIP OVER
2096                                        11690   ;THE ERROR MESSAGE.
2097                                        11700
2098  006046  012777  004002  173646        11710   DOWRITE: MOV    #4002,@RKCS     ;WRITE, IBA
2099  006054  010177  173650                11720   DOXFER: MOV     R1,@RKDA        ;ADDRESS THE DRIVE
2100  006060  010377  173640                11730           MOV     R3,@RKWC        ;XFER THIS # OF WORDS
2101  006064  005477  173634                11740           NEG     @RKWC
2102  006070  012777  032106  173630        11750           MOV     #DBUF,@RKBA     ;USE THIS BUS ADDRESS
2103  006076  012777  006156  173634        11760           MOV     #3$,@RKVEC      ;SET UP INTERRUPT VECTOR
2104  006104  005046                        11770           CLR     -(SP)           ;NEW PSW
2105  006106  012746  006114                11780           MOV     #1$,-(SP)       ;SET NEW PC TO STACK **********
2106  006112  000002                        11790           RTI
2107  006114  052777  000101  173600        11800   1$:     BIS     #101,@RKCS      ;SET IDE, GO (WRITE,IBA/ READ)
2108  006122  005037  002166                11810           CLR     CICNT
2109  006126  012737  177760  002170        11820           MOV     #-20,CICNT1
2110  006134  005237  002166                11830   2$:     INC     CICNT           ;WAIT  FOR INTERRUPT
2111  006140  001375                        11840           BNE     .-4
2112                                        11850
2113  006142  005237  002170                11860           INC     CICNT1
2114  006146  001372                        11870           BNE     2$
2115                                        11880
2116  006150  004737  021462                11890           JSR     PC,GT4RG        ;TIMED OUT, INTERRUPT DID NOT OCCUR
2117  006154  000207                        11900           RTS     PC              ;RETURN TO THE EROR MEASGE
2118                                        11910
2119  006156  022626                        11920   3$:     CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
2120  006160  062716  000002                11930           ADD     #2,(SP)         ;ADJUST RETURN ADDRESS TO SKIP OVER
2121  006164  000207                        11940           RTS     PC              ;EROR MESAGE ON RETURN
```

```
 2122                                          11960   ;THIS ROUTINE PERFORMS A READ ON THE DISK.AT THE TIME OF ENTRY R1 CONTAINS
 2123                                          11970   ;THE DISK ADDRESS FROM WHERE THE READ IS TO BE DONE. R3 CONTAINS THE WORD
 2124                                          11980   ;COUNT (RKWC). READ WILL BE DONE INTO DATA BUFFER AT 'DBUF'.
 2125                                          11990
 2126                                          12000   ;READ IS DONE IN INTERRUPT MODE. IF THE INTERRUPT DOES NOT OCCUR WITHIN
 2127                                          12010   ;A CERTAIN TIME, RETURN IS MADE TO THE ERROR MESSAGE FOLLOWING THE 'JSR'
 2128                                          12020   ;CALL. IF THE INTERRUPT OCCURS AS EXPECTED, RETURN ADDRESS IS ADJUSTED
 2129                                          12030   ;TO SKIP OVER THE ERROR MESSAGE.
 2130                                          12040
 2131  006166  012777  000004  173526         12050   DOREAD: MOV     #4,@RKCS        ;READ
 2132  006174  000727                         12060           BR      DOXFER
 2133                                          12070
 2134                                          12080
 2135                                          12090   ;CLEANBUF
 2136                                          12100   ;CLEANS OUT THE DATA BUFFER (ALL WORDS WRITTEN TO 177777) INTO WHICH THE
 2137                                          12110   ;READ FROM THE DISK WILL BE DONE.  DATA BUFFER STARTS AT 'DBUF'AND IS
 2138                                          12120   ;1000 (OCTAL) WORDS LONG.
 2139                                          12130
 2140  006176  012702  177000                 12140   CLEANBUF:MOV    #-1000,R2       ;SET COUNT
 2141  006202  012705  032106                 12150           MOV     #DBUF,R5        ;INITIALIZE BA
 2142  006206  012725  022222                 12160   1$:     MOV     #22222,(R5)+
 2143  006212  005202                         12170           INC     R2              ;DONE ALL WORDS?
 2144                                          12180                                   ;BUFFER STARTING AT (PHYSICAL) BUS
 2145                                          12190                                   ;ADDRESS 177000 (177000-200776)
 2146  006214  001374                         12200           BNE     1$
 2147  006216  000207                         12210           RTS     PC              ;YES RETURN
```

```
                                        ;;*****************************************************************
                                        ;*TEST 5         CHECK DATA TRANSFER AROUND 32K BOUNDARY
                                                ;*THIS TEST PERFORMES A WRITE OF 2 SECTORS ON THE DISK FROM MEMORY
                                                ;*LOCATIONS AROUND THE 32K BOUNDARY.  SECTORS 0,1, CYL 0, SURFACE
                                                ;*0 ARE WRITTEN.  PHYSICAL BUS ADDRESSES FOR THE DATA BUFFER:
                                                ;* 177000 TO 200776 I.E. (32K-256) TO (32K+255)

                                                ;*CHECKING IS DONE TO SEE IF MEX BITS, RKBA,RKDA,RKWC INCREMENTED
                                                ;*CORRECTLY. THEN DATA BUFFER IS CLEARED OUT AND A READ IS DONE
                                                ;*INTO IT.  A CHECK IS MADE TO SEE IF THE CORRECT DATA WAS RECIEVED.
                                                ;*ONLY 12 DATA ERRORS ARE REPORTED.
                                        ;;*****************************************************************
006220  000004                        TST5:   SCOPE
006222  023727  017464  002000  12350          CMP     $LSTBK,#2000    ;33K OR MORE OF MEMORY?
006230  103002                       12360          BHIS    1$              ;YES
006232  000137  006602               12370          JMP     TST6            ;IF NOT, DONT DO THIS TEST
006236  012737  001600  172352       12380  1$:     MOV     #1600,@#KIPAR5  ;MAP 28-32K THRU PAR 5
006244  012737  002000  172354       12390          MOV     #2000,@#KIPAR6  ;MAP 32-36K THRU PAR 6
006252  012737  000001  177572       12400          MOV     #1,@#SR0        ;TURN ON MEMORY MANAGEMENT
                                     12410
                                     12420  ;SET UP DATA BUFFER (1000 OCTAL WORDS LONG) FOR WRITING TWO SECTORS
                                     12430  ;ON THE DISK.  THE TRANSFER IS DONE AROUND THE 32K BOUNDARY FROM
                                     12440  ;BUS ADDRESS (PHYSICAL) 177000 TO 200776, (32K-256) TO (32+256)
                                     12450
                                     12460  ;DATA IN THE BUFFER IS A COUNT PATTERN STARTING FROM 1 FOR THE FIRST
                                     12470  ;WORD TO 1000 FOR THE LAST WORD.
                                     12480  ;PHYSICAL BUFFER ADDRESS:  177000 TO  200776 (32K-256 TO 32K+256)
006260  012737  006266  00:110       12490          MOV     #2$,$LPERR      ;LUP TO 2$ ON EROR (SW 9)
006266  104415                       12500  2$:     CON.RESET
006270  104417                       12510          DRV.RESET
                                     12520
006272  012700  000001               12530          MOV     #1,R0           ;INITIALIZE DATA PATTERN TO BE
                                     12540                                  ;WRITTEN
006276  012701  137000               12550          MOV     #137000,R1      ;BA TO START PHYSICAL ADDRESS=177000
006302  010021                       12560  3$:     MOV     R0,(R1)+        ;WRITE COUNT PATTERN (1-1000)
006304  005200                       12570          INC     R0              ;INTO DATA BUFFER (PHYS ADDRES
006306  020027  001001               12580          CMP     R0,#1001        ;177000 TO 200776)
006312  001373                       12590          BNE     3$
006314  013777  002202  173406       12600          MOV     @DRV,@RKDA      ;SUR 0, SEC 0, CYL 0
006322  012777  177000  173374       12610          MOV     #-1000,@RKWC    ;WORD COUNT =2 SECS
006330  012777  177000  173370       12620          MOV     #177500,@RKBA   ;BUS ADDRESS.
006336  012777  000003  173356       12630          MOV     #3,@RKCS        ;WRITE,GO
006344  104415                       12640          CON.RDY                 ;WAIT FOR CNTROL RDY
                                     12650
006346  004737  017604               12660          JSR     PC,CHKCS        ;ANY EROR IN RKCS?
006352  104102                       12670          ERROR   102             ;'ERR' SET IN RKCS, ON DOING A
                                     12680                                  ;WRITE OF SECTORS (0,1) FROM
                                     12690                                  ;(PHYSICAL) BUS ADDRESSS 177000
                                     12700                                  ;(177000 TO 200776)
006354  004737  017676               12710          JSR     PC,CHKMEX
                                     12720                                  ;CHECK THAT RKBA OVERFLOWED INTO
                                     12730                                  ;EXTENDED MEM. BIT (0) OF RKCS (BIT 4)
                                     12740                                  ;EX MEM BIT 0 SET?
                                     12750                                  ;GET RKCS,ER,DS,DA
006360  104106                       12760          ERROR   106             ;MEX BITS INCORRECT, BIT 4 OF RKCS
                                     12770                                  ;(MEX BIT 0) SHOULD HAVE SET
```

```
2204                                        12780                       ;AFTER RKBA OVERFLOWED ON DOING
2205                                        12790                       ;A TRANSFER OF 2 SECTORS FROM BUS
2206                                        12800                       ;ADDRESS (PHYSICAL) STARTING AT 177000
2207  006362  012703  001000                12810           MOV     #1000,R3    ;WORD COUNT
2208  006366  012704  177000                12820           MOV     #177000,R4  ;STARTING BUS ADDRESS
2209  006372  004737  017642                12830           JSR     PC,CHKBA    ;CHECK IF RKBA INCREMENTED CORRECTLY
2210  006376  104104                       12840           ERROR   104         ;;RKBA DID NOT INCREMENT CORRECTLY
2211                                        12850                       ;AFTER WRITE IF 2 SECTORS FROM A DATA
2212                                        12860                       ;BUFFER STARTING AT (PHYSICAL) BUS
2213                                        12870                       ;ADDRESS (177000-200776)
2214  006400  013702  002202                12880           MOV     @DRV,R2     ;GET EXPECTED
2215  006404  062702  000002                12890           ADD     #2,R2       ;DISK ADDRESS
2216  006410  004737  017620                12900           JSR     PC,CHKDA    ;CHECK IF RKDA INCREMENTED CORRECTLY
2217  006414  104103                       12910           ERROR   103         ;RKDA INCREMENTED WRONGLY AFTER
2218                                        12920                       ;A WROTE OF 2 SECTOR (0,1) FROM
2219                                        12930                       ;DATA BUFFER STARTING AT BUS
2220                                        12940                       ;ADDRESS (PHYSICAL) 177000.
2221                                        12950
2222  006416  004737  017722                12960           JSR     PC,CHKWC    ;CHECK IF RKWC OVERFLOWED CORRECTLY
2223  006422  104105                       12970           ERROR   105         ;RKWC DID NOT OVERFLOW TO 0 ON
2224                                        12980                       ;DOING A WRITE OF 2 SECTORS FROM
2225                                        12990                       ;BA = 177000
2226                                        13000
2227                                        13010                       ;NOW, READ IS DONE OF THE DATA
2228                                        13020                       ;THAT WAS WRITTEN TO SEE IF IT
2229                                        13030                       ;CAN BE READ CORRECTLY
2230                                        13040
2231  006424  012737  006432  001110        13050           MOV     #4$,$LPERR  ;LUP TO 4$ ON EROR (SW 9)
2232  006432  104415                       13060   4$:     CON.RESET
2233  006434  104417                       13070           DRV.RESET
2234                                        13080
2235  006436  012701  137000                13090           MOV     #137000,R1  ;CLEAR THE 1000-WORD DATA
2236  006442  005021                       13100   5$:     CLR     (R1)+       ;BUFFER (PA 177000 TO 200776)
2237  006444  020127  141000                13110           CMP     R1,#141000  ;ALL DONE?
2238  006450  001374                       13120           BNE     5$
2239                                        13130
2240                                        13140                       ;NOW, READ BACK INTO THE
2241                                        13150                       ;SAME BUFFER 2 SECTORS
2242                                        13160                       ;WRITTEN PREVIOUSLY
2243                                        13170
2244  006452  013777  002202  173250        13180           MOV     @DRV,@RKDA  ;ADDRESS THE DRIVE CYL 0, SEC 0, 0
2245  006460  012777  177000  173236        13190           MOV     #-1000,@RKWC ;READ 2 SECTORS
2246  006466  012777  177000  173232        13200           MOV     #177000,@RKBA ;INTO THIS BUS ADDRESS
2247  006474  012777  000005  173220        13210           MOV     #5,@RKCS    ;READ, GO
2248                                        13220
2249  006502  104416                       13230           CON.RDY             ;WAIT FOR CNTROL RDY
2250                                        13240
2251  006504  004737  017604                13250           JSR     PC,CHKCS    ;ANY ERROR IN RKCS?
2252  006510  104102                       13260           ERROR   102         ;ERROR BIT SET IN RKCS ON DOING
2253                                        13270                       ;A READ OF 2 SECTORS (0,1), CYL 0,
2254                                        13280                       ;SUR 0, INTO DATA BUFFER STARTING
2255                                        13290                       ;AT BUS ADDRESS 1770000, NOTE AFTER
2256                                        13300                       ;1777776, RKBA WILL OVERFLOW (0)
2257                                        13310                       ;INTO MEX BITS (BIT 4) OF RKCS.
2258                                        13320                       ;IF THE ENTIRE TRANSFER (1000 WORD)
2259                                        13330                       ;WAS DONE RKBA WILL CONTAIN 1000
```

```
2260                                                13340                                   ;AND BIT 4 OF RKCS (MEX) WILL BE SET.
2261                                                13350
2262  006512  012705  177764                      13360  6$:      MOV     #-14,R5         ;REPORT ONLY 12 ERRORS.
2263  006516  012702  137000                      13370           MOV     #137000,R2      ;STARTING ADDRESS OF BUFFER
2264                                                13380                                   ;(PA=177000)
2265  006522  012701  000001                      13390           MOV     #1,R1           ;INITIALIZE DATA PATTERN
2266                                                13400
2267  006526  020112                             13410  7$:      CMP     R1,(R2)         ;CORRECT DATA WORD RECVD?
2268  006530  001414                             13420           BEQ     8$
2269  006532  005705                             13430           TST     R5              ;REPORT THIS ERROR?
2270  006534  001420                             13440           BEQ     9$
2271  006536  005205                             13450           INC     R5              ;COUNT ERORS
2272                                                13460
2273  006540  010137  001162                      13470           MOV     R1,$REG0        ;GET EXPCTED DATA WORD
2274                                                13480
2275  006544  011237  001164                      13490           MOV     (R2),$REG1      ;GET DATA WORD RECVD
2276  006550  104107                             13500           ERROR   107             ;DATA COMPARISON ERROR ON DOING A
2277                                                13510                                   ;READ OF 2 SECTORS (0,1, CYL 0, SURFACE 0)
2278                                                13520                                   ;INTO DATA BUFFER (PHYSICAL ADDRESS
2279                                                13530                                   ;177000 TO 200776)
2280                                                13540
2281  006552  104420  002716                      13550           TYPMSG  ,MSG13          ;TYPE 'PHYSICAL BUS ADDRESS'
2282  006556  004737  017466                      13560           JSR     PC,TYPDBO       ;TYPE THE 6 DIGIT PHYSICAL BUS ADDRESS
2283                                                13570                                   ;WHERE THE DATA ERROR OCCURRED.
2284                                                13580
2285  006562  062702  000002                      13590  8$:      ADD     #2,R2           ;INCREMENT POINTER TO BA
2286  006566  005201                             13600           INC     R1
2287  006570  022701  001001                      13610           CMP     #1001,R1        ;CHECKED THE ENTIRE BUFFER?
2288  006574  001354                             13620           BNE     7$
2289                                                13630
2290                                                13640                                   ;**OFF
2291  006576  005037  177572                      13650  9$:      CLR     @#SR0           ;TURN OFF MEMORY MANAGEMENT
```

```
                                                ;;*****************************************************************
                                                ;*TEST 6         CHECK DATA TRANSFER FROM 28K TO 32K
                                                        ;*T.    TEST DOES A WRITE OF 4K WORDS FROM A BUFFER LOCATED AT 28K
                                                        ;*TH... THE DATA IS READ BACK INTO THE SAME BUFFER(28K-32K) AND IS
                                                        ;*CHECKED TO SEE IF IT IS CORRECT.  NOTE THAT THE BUFFER IS FILLED
                                                        ;*WITH ALL 1'S BEFORE DOING THE READ.

                                                        ;*THE WRITE IS DONE STARTING AT CYLINDER 0, SECTOR 0, SURFACE 0.
                                                ;;*****************************************************************
006602  000004                                  TST6:   SCOPE
                                        13760
006604  023727  017464  001740          13770           CMP     $LSTBK,#1740    ;32K OR MORE OF MEMORY?
006612  103002                          13780           BHIS    1$              ;YES
006614  000137  007136                  13790           JMP     TST7            ;IF NOT, DONT DO THIS TEST
006620  012737  001600  172352          13800   1$:     MOV     #1600,@#KIPAR5  ;MAP 28-32K THRU PAR 5
006626  012737  000001  177572          13810           MOV     #1,@#SR0        ;TURN ON MEM MANAGEMENT
                                        13820
                                        13830   ;WRITE A COUNT PATTERN (1-10000) INTO DATA BUFFER (PHYSICAL ADDRESS
                                        13840   ;160000-177776).  THIS DATA BUFFER WILL BE USED FOR WRITING ON THE DISK.
                                        13850
006634  012737  006642  001110          13860           MOV     #2$,$LPERR      ;LUP TO 2$ ON EROR
006642  104415                          13870   2$:     CON.RESET
006644  104417                          13880           DRV.RESET
006646  012700  000001                  13890           MOV     #1,R0           ;INITIALIZE DATA PATTERN TO BE WRITTEN
006652  012701  120000                  13900           MOV     #120000,R1      ;INITIALIZE BA (PA=160000) 28K
006656  010021                          13910   3$:     MOV     R0,(R1)+        ;WRITE COUNT PATTERNS (1-10000)
006660  005200                          13920           INC     R0              ;INTO DATA BUFFER (PA 160000 TO
006662  020027  010001                  13930           CMP     R0,#10001       ;177776, 28K-32K)
006666  001373                          13940           BNE     3$
                                        13950
006670  013777  002202  173032          13960           MOV     @DRV,@RKDA      ;WRITE ON SEC 0, CYL 0, SUR1
006676  012777  170000  173020          13970           MOV     #-10000,@RKWC   ;4K WORDS
006704  012777  160000  173014          13980           MOV     #160000,@RKBA   ;FROM THIS BUS ADDRESS
006712  012777  000003  173002          13990           MOV     #3,@RKCS        ;WRITE, GO
                                        14000
006720  104416                          14010           CON.RDY                 ;WAIT FOR CONTROL READY
                                        14020
006722  004737  017604                  14030           JSR     PC,CHKCS        ;ANY ERROR IN RKCS?
006726  104102                          14040           ERROR   102             ;'ERR' BIT SET IN RKCS ON DOING
                                        14050                                   ;A WRITE OF 4K WORDS FROM 160000
                                        14060                                   ;(28K-32K).  DISK WRITE BEGAN ON
                                        14070                                   ;SEC 0, CYL 0, SUR 0.  IF ALL 4K
                                        14080                                   ;WORDS WERE WRITTEN RKBA SHOULD OVERFLOW
                                        14090                                   ;AND CONTAIN 0.  BIT 4 OF RKCS
                                        14100                                   ;(MEX BIT) SHOULD BE 1
                                        14110
006730  004737  017676                  14120           JSR     PC,CHKMEX       ;CHECK IF RKBA OVERFLOWED AND MEX
                                        14130                                   ;BITS (4) IN RKCS WAS SET.
006734  104106                          14140           ERROR   106             ;;MEX BIT 4 NOT SET AFTER OVERLFOW OF
                                        14150                                   ;RKBA.  WRITE OF 4K WORDS (28K-32K) WAS DONE.
                                        14160
                                        14170                                   ;RETURN HERE IF NO ERROR
006736  012703  010000                  14180           MOV     #10000,R3       ;WORD COUNT
006742  012704  160000                  14190           MOV     #160000,R4      ;STARTING BUS ADDRESS
006746  004737  017642                  14200           JSR     PC,CHKBA        ;CHECK IF RKBA INCREMENTED CORRECTLY
006752  104104                          14210           ERROR   104             ;RKBA DID NOT OVERFLOW TO AFTER A WRITE IF 4K
```

```
2348                                                14220                                   ;WORDS (160000 TO 177776)
2349                                                14230
2350  006754  004737  017722                       14240           JSR     PC,CHKWC        ;CHECK RKWC OVERFLOWED CORRECTLY
2351  006760  104105                               14250           ERROR   105             ;RKWC DID NOT OVEFLOW TO 0
2352                                                14260                                   ;AFTER A WRITE OF 4K WORD (160000
2353                                                14270                                   ;TO 177776)
2354                                                14280
2355                                                14290
2356  006762  012737  006770  001110                14300           MOV     #4$,$LPERR      ;RETURN ADDRESS FOR LUPING
2357  006770  104415                               14310   4$:     CON.RESET
2358  006772  104417                               14320           DRV.RESET
2359                                                14330
2360  006774  012701  120000                       14340           MOV     #120000,R1      ;CLEAR THE DATA BUFFER BEFORE
2361  007000  005021                               14350   5$:     CLR     (R1)+           ;DOING A READ INTO IT
2362  007002  022701  140000                       14360           CMP     #140000,R1
2363  007006  001374                               14370           BNE     5$
2364                                                14380
2365  007010  013777  002202  172712                14390           MOV     @DRV,@RKDA      ;READ FROM SEC 0, SUR 0, CYL 0
2366  007016  012777  170000  172700                14400           MOV     #-10000,@RKWC   ;4K WORDS
2367  007024  012777  160000  172674                14410           MOV     #160000,@RKBA   ;INTO THIS BUS ADDRESS
2368                                                14420
2369  007032  012777  000005  172662                14430           MOV     #5,@RKCS        ;READ, GO
2370  007040  104416                               14440           CON.RDY                 ;WAIT FOR CNTROL RDY
2371                                                14450
2372  007042  004737  017604                       14460           JSR     PC,CHKCS        ;ANY ERROR IN RKCS?
2373  007046  104102                               14470           ERROR   102             ;ERROR BIT SET IN RKCS ON DOING
2374                                                14480                                   ;A READ OF 4K WORDS INTO MEMORY
2375                                                14490                                   ;(160000-177776)
2376  007050  012705  177764                       14500           MOV     #-14,R5         ;REPORT 12 ERRORS AT MOST
2377  007054  012702  120000                       14510           MOV     #120000,R2      ;STARTING ADDRESS OF BUFFER
2378                                                14520                                   ;(PA=160000)
2379  007060  012701  000001                       14530           MOV     #1,R1           ;INITIALIZE DATA PATTERN
2380                                                14540
2381  007064  020112                               14550   6$:     CMP     R1,(R2)         ;CORRECT DATA WORD?
2382  007066  001413                               14560           BEQ     7$
2383  007070  010137  001162                       14570           MOV     R1,$REG0        ;GET EXPCTD DATA WORD
2384  007074  011237  001164                       14580           MOV     (R2),$REG1      ;GET DATA WORD RECVD
2385  007100  104107                               14590           ERROR   107             ;DATA ERROR ON DOING A READ OF
2386                                                14600                                   ;4K WORDS STARTING FROM SEC 0,
2387                                                14610                                   ;CYL 0, SUR 0 INTO MEMORY (160000-
2388                                                14620                                   ;177776)
2389  007102  104420  002716                       14630           TYPMSG  ,MSG13          ;TYPE 'PHYSICAL BUS ADDRESS'
2390  007106  004737  017466                       14640           JSR     PC,TYPDBO       ;TYPE OUT THE PHYSICAL BUS ADDRESS
2391                                                14650                                   ;WHERE DATA
2392  007112  005205                               14660           INC     R5              ;ERROR OCCURRED.  R2 CONTAINS
2393  007114  001406                               14670           BEQ     8$              ;THE VIRTUAL ADDRESS
2394                                                14680
2395  007116  062702  000002                       14690   7$:     ADD     #2,R2           ;POINTER TO NEXT BA
2396  007122  005201                               14700           INC     R1              ;NEXT PATTERN
2397  007124  022701  001000                       14710           CMP     #1000,R1        ;ALL DONE?
2398  007130  001355                               14720           BNE     6$              ;NO
2399                                                14730
2400  007132  005037  177572                       14740   8$:     CLR     @#SR0           ;TURN OFF MEM MANAGEMENT
```

```
                                        ;;*****************************************************************
                                        ;*TEST 7        PERFORM THE LARGEST POSSIBLE DATA TRANSFER
                                        ;THIS TEST PERFORMS THE LARGEST DATA TRANSFER POSSIBLE
                                        ;WITHIN AVAILABLE MEMORY.

                                        ;1) IF KT11 IS PRESENT A WRITE OF 16K WORDS IS DONE ON
                                        ;THE DISK (PROVIDED 28K OR MORE IS PRESENT).

                                        ;2) IF KT11 IS NOT PRESENT THEN ALL BUT TOP 1.5K OF MEMORY WILL BE USED
                                        ;FOR WRITING (RKDP MONITOR).

                                        ;ALL DATA TRANSFERS ARE DONE STARTING AT BA 'PGEND'.
                                        ;ALL WRITES ARE DONE BEGINNING AT SECTOR 0, CYLINDER 0,
                                        ;SURFACE 0.

                                        ;AFTER DOING THE 'WRITE' A 'WRITE CHECK' IS DONE
                                        ;TO SEE IF THE WRITE WAS DONE CORRECTLY.  ANY WRITE
                                        ;CHECK ERROR IS REPORTED.
                                        ;;*****************************************************************
007136  000004                  14940   TST7:   SCOPE
007140  005737  017216          14950           TST     $KT11           ;MEMORY MANAGEMENT PRESENT?
007144  100004                  14960           BPL     1$              ;NO
                                14970                                   ;YES
007146  023727  017464  001540  14980           CMP     $LSTBK,#1540    ;28K OR MORE?
007154  002034                  14990           BGE     XFR16K          ;YES
                                15000
007156  013703  002554          15010   1$:     MOV     MAXBA,R3        ;FIGURE OUT THE MAXIMUM
007162  162703  032106          15020           SUB     #PGEND,R3       ;XFER THAT CAN BE DONE
007166  000241                  15030           CLC
007170  006003                  15040           ROR     R3              ;FROM 'PGEND' TO THE
                                15050                                   ;'MAXBA'
007172  005001                  15060           CLR     R1
007174  010346                  15070           MOV     R3,-(SP)        ;PUT DIVIDEND ON STACK (LO)
007176  005046                  15080           CLR     -(SP)           ;(HIGH PART)
007200  012746  000400          15090           MOV     #400,-(SP)      ;DIVISOR
007204  004737  024640          15100           JSR     PC,@#$DIV       ;GO TO DIVIDE ROUTINE
007210  005726                  15110           TST     (SP)+           ;POP OFF REMAINDER FROM STACK
007212  001401                  15120           BEQ     2$
007214  005201                  15130           INC     R1              ;GET # OF SECTORS REQUIRED TO
                                15140                                   ;DO WRITE OF # OF WORDS
007216  062601                  15150   2$:     ADD     (SP)+,R1        ;(CONTAINED IN R0). R1 CONTAINS
                                15160                                   ;# OF SECTORS
007220  005002                  15170           CLR     R2              ;FORM THE EXPECTED DISK
007222  162701  000014          15180   3$:     SUB     #14,R1          ;ADDRESS - AFTER THE WRITE
007226  100403                  15190           BMI     4$              ;IS DONE.
007230  062702  000020          15200           ADD     #20,R2
007234  000772                  15210           BR      3$
007236  062701  000014          15220   4$:     ADD     #14,R1          ;R2 CONTAINS EXPCTD RKDA
007242  050102                  15230           BIS     R1,R2           ;AFTER WRITE IS DONE
007244  000404                  15240           BR      XFR
                                15250
007246  012703  040000          15260   XFR16K: MOV     #40000,R3       ;# OF WORDS = 16K
007252  012702  000124          15270           MOV     #124,R2         ;RKDA SHOULD INCREMENT TO
                                15280                                   ;THIS AFTER A TRANSFER OF 16K
                                15290                                   ;WORDS STARTING AT DISK
                                15300                                   ;ADDRESS = 0 (CYL 2,SUR 1,SEC 4)
```

```
2457                                                15310
2458  007256  053702  002202                        15320  XFR:    BIS     @DRV,R2          ;ADD THE DRIVE # TO
2459                                                15330                                   ;EXPCTD RKDA AFTER XFER
2460  007262  012737  007270  001110                15340          MOV     #1$,$LPERR       ;LUP BAK TO '1$' ON ERROR
2461                                                15350                                   ;(SW 9)
2462  007270  104415                                15360  1$:     CON.RESET
2463  007272  104417                                15370          DRV.RESET
2464  007274  013777  002202  172426                15380          MOV     @DRV,@RKDA       ;ADDRESS THE DRIVE, CYL 0,SUR 0,
2465                                                15390                                   ;SEC 0
2466  007302  010377  172416                        15400          MOV     R3,@RKWC
2467  007306  005477  172412                        15410          NEG     @RKWC            ;WORD COUNT (IF MORE THAN
2468                                                15420                                   ;28K IS AVAILABLE A TRANSFER
2469                                                15430                                   ;OF 20K WILL BE DONE, IF
2470                                                15440                                   ;LESS THAN 28K, THE
2471                                                15450                                   ;LARGEST TRANSFER WITHIN THE
2472                                                15460                                   ;AVAILABLE MEMORY WILL
2473                                                15470                                   ;BE DONE. IN BOTH
2474                                                15480                                   ;CASES, DATA TRANSFER
2475                                                15490                                   ;WILL BE DONE STARTING
2476                                                15500                                   ;AT 'PGEND'
2477  007312  012777  032106  172406                15510          MOV     #PGEND,@RKBA     ;START WRITE FROM HERE
2478  007320  012777  000003  172374                15520          MOV     #3,@RKCS         ;WRITE, GO
2479                                                15530
2480  007326  104416                                15540          CON.RDY                  ;WAIT FOR CONTROL READY
2481                                                15550
2482  007330  004737  017604                        15560          JSR     PC,CHKCS         ;CHCK RKCS FOR ERROR?
2483  007334  104102                                15570          ERROR   102              ;ERROR BIT SET IN RKCS ON
2484                                                15580                                   ;DOING A LARGE 'WRITE'
2485                                                15590
2486  007336  004737  017620                        15600          JSR     PC,CHKDA         ;CHECK IF RKDA INCREMENTED CORRECTLY?
2487  007342  104103                                15610          ERROR   103              ;RKDA DID NOT INCREMENT
2488                                                15620                                   ;CORRECTLY
2489                                                15630
2490  007344  012704  032106                        15640          MOV     #PGEND,R4
2491  007350  004737  017642                        15650          JSR     PC,CHKBA         ;CHECK THAT RKBA INCREMENTED
2492                                                15660                                   ;CORRECTLY?
2493  007354  104104                                15670          ERROR   104              ;RKBA DID NOT INCREMENT
2494                                                15680                                   ;CORRECTLY
2495                                                15690
2496  007356  004737  017722                        15700          JSR     PC,CHKWC         ;CHECK THAT RKWC OVERFLOWED
2497  007362  104105                                15710          ERROR   105              ;RKWC DID NOT OVERFLOW TO
2498                                                15720                                   ;ZERO AFTER A WRITE
2499                                                15730
2500  007364  012737  007372  001110                15740          MOV     #2$,$LPERR       ;LUP BAK TO '2$' ON EROR (SW 9)
2501  007372  104415                                15750  2$:     CON.RESET
2502  007374  104417                                15760          DRV.RESET
2503  007376  013777  002202  172324                15770  3$:     MOV     @DRV,@RKDA       ;ADDRESS THE DRIVE, CYL 0,SEC 0,SUR 0
2504  007404  010377  172314                        15780          MOV     R3,@RKWC
2505  007410  005477  172310                        15790          NEG     @RKWC
2506  007414  012777  032106  172304                15800          MOV     #PGEND,@RKBA     ;STARTING BA
2507                                                15810
2508  007422  012777  000407  172272                15820  4$:     MOV     #407,@RKCS       ;WRITE CHECK, SSE, GO
2509                                                15830
2510  007430  104416                                15840          CON.RDY                  ;WAIT FOR CONTROL RDY
2511                                                15850
2512  007432  004737  017604                        15860          JSR     PC,CHKCS         ;ANY ERROR IN RKCS
```

```
2513  007436  104102                  15870            ERROR   102
2514                                  15880
2515  007440  032777  040000  172254  15890            BIT     #BIT14,@RKCS    ;SKIP CHECKING FOR WCE IF HE SET
2516  007446  001030                  15900            BNE     7$
2517  007450  032777  000001  172242  15910    5$:     BIT     #WCE,@RKER      ;WRITE CHECK ERROR?
2518  007456  001406                  15920            BEQ     6$              ;NO
2519                                  15930
2520  007460  004737  021462          15940            JSR     PC,GT4RG        ;GET RKCS,ER,DS,DA
2521  007464  017737  172236  001166  15950            MOV     @RKBA,$REG2
2522  007472  104110                  15960            ERROR   110             ;WRITE CHECK ERROR. RKDA GIVES THE DISK ADDRESS
2523                                  15970                                    ;WHERE WCE OCCURRED. (NOTE THE SECTOR IN
2524                                  15980                                    ;ERROR IS OBTAINED BY BACKING OFF 1 SECTOR).
2525                                  15990                                    ;NOTE THAT THE DATA BUFFER WHICH WAS WRITE
2526                                  16000                                    ;CHECKED IS NOT ON A SECTOR BOUNDARY
2527                                  16010                                    ;(LIKE 256, 512 WORD ETC.), BUT IS SOME
2528                                  16020                                    ;SECTORS & A FRACTION (LIKE 300 WORDS ETC.)
2529                                  16030
2530  007474  005777  172224          16040    6$:     TST     @RKWC           ;WAS THE ENTIRE DATA BUFFER
2531  007500  001413                  16050            BEQ     7$              ;WRITE CHECKED?
2532                                  16060
2533  007502  017705  172216          16070            MOV     @RKWC,R5
2534  007506  042705  177760          16080            BIC     #177760,R5      ;FORM THE RKBA AND RKWC
2535  007512  006305                  16090            ASL     R5
2536  007514  160577  172206          16100            SUB     R5,@RKBA        ;TO BE USED FOR DOING
2537                                  16110                                    ;WRITE-CHECK IF THE REST
2538  007520  042777  000017  172176  16120            BIC     #17,@RKWC       ;OF THE DATA BUFFER
2539                                  16130
2540  007526  000735                  16140            BR      4$              ;GO DO WRT-CHK FOR
2541                                  16150                                    ;REST OF THE BUFFER
2542                                  16160
2543                                  16170    ;GO CHECK THE REST OF THE DRIVES.
2544                                  16180
2545  007530  000137  005130          16190    7$:     JMP     NXTDRV
```

```
                                                16210   .SBTTL  EXERCISER PROGRAM
                                                16220
                                                16230   ;BEGINING OF THE EXERCISER PART OF THE PROGRAM.
                                                16240   ;IF THE PROGRAM WAS RESTARTED AT 210, THEN THE STATISTICS COLLECTED
                                                16250   ;SO FAR WILL NOT BE CLEARED.
                                                16260
                                                16270
007534  104415                                  16280   EXRCSR: CON.RESET
007536  012700  002006                          16290           MOV     #KEY,R0         ;CLEAR UP THE LOCATIONS FROM
007542  005020                                  16300   1$:     CLR     (R0)+           ;'KEY' TO 'KWHR' (EXCLUDED)
007544  020027  002252                          16310           CMP     R0,#KWHR
007550  001374                                  16320           BNE     1$
007552  105737  001753                          16330           TSTB    FRSTRT          ;RESTARTED AT 210?
007556  001045                                  16340           BNE     3$              ;YES, SAVE THE STATISTICS COLLECTED
                                                16350                                   ;UPTILL NOW, DONT CLEAR THEM
                                                16360
007560  005020                                  16370   2$:     CLR     (R0)+           ;CLEAR LOCATIONS FROM 'HECN'
007562  020027  002532                          16380           CMP     R0,#PCMND       ;TO 'PCMND' (EXCLUDED)
007566  001374                                  16390           BNE     2$
                                                16400
007570  012700  002254                          16410           MOV     #KWMIN,R0       ;INITIALIZE COUNTS FOR MINS,
007574  012701  177704                          16420           MOV     #-60.,R1        ;SECS, KW11
007600  010120                                  16430           MOV     R1,(R0)+
007602  010120                                  16440           MOV     R1,(R0)+
007604  010120                                  16450           MOV     R1,(R0)+
                                                16460
007606  012700  025356                          16470           MOV     #RSDRVL,R0      ;INITIALIZE PTR TO RANDOM NO. SEEDS
007612  012720  123456                          16480           MOV     #123456,(R0)+   ;USED BY THE RANDOM NUMBER GENERATOR
007616  012720  176543                          16490           MOV     #176543,(R0)+   ;SET UP RANDOM NUMBER SEEDS TO BE
007622  012720  001201                          16500           MOV     #1201,(R0)+
007626  012720  062465                          16510           MOV     #62465,(R0)+
007632  012720  176105                          16520           MOV     #176105,(R0)+
007636  012720  174532                          16530           MOV     #174532,(R0)+
007642  012720  157650                          16540           MOV     #157650,(R0)+
007646  012720  030753                          16550           MOV     #30753,(R0)+
007652  012720  131547                          16560           MOV     #131547,(R0)+
007656  012720  032070                          16570           MOV     #32070,(R0)+
007662  012720  123456                          16580           MOV     #123456,(R0)+
007666  012720  176543                          16590           MOV     #176543,(R0)+
                                                16600
                                                16610
007672  013737  001736  002556                  16620   3$:     MOV     PCNTR,REPCNT    ;REPETITION COUNT. WHEN THIS COUNT GOES
                                                16630                                   ;TO 0, IT'S CONSIDERED TO BE AN END OF PASS.
                                                16640                                   ;THE NEXT PASS WILL START WILL START RIGHT
                                                16650                                   ;FROM WHERE THE EXERCISER LEFT OFF.
                                                16660
007700  004737  022304                          16670           JSR     PC,CHDPRS       ;CHECK IF ANY DRIVES ARE PRESENT
                                                16680                                   ;IF NONE, GO TO $EOP, END OF PASS
                                                16690
                                                16700
                                                16710

007704  012746  177777                                  MOV     #177777,-(SP)   ;PUT LOW DIVIDEND ON STACK
007710  005046                                          CLR     -(SP)           ;CLEAR HIGH DIVIDEND AND PUSH
                                                                                ;IT ON STACK
007712  013746  001764                                  MOV     DRVPRS,-(SP)    ;PUSH DIVISOR ON STACK
```

```
 2602  007716  004737  024640                    JSR     PC,@#$DIV       ;GO TO THE 'DIVIDE' SUBROUTINE
 2603  007722  005726                            TST     (SP)+           ;DISCARD THE REMAINDER. QUOTIENT IS
 2604                                                                    ;NOW ON TOP OF THE STACK
 2605  007724  012637  002220          16730     MOV     (SP)+,DRMAP     ;GET MAPPING FACTOR FOR DRIVES
 2606  007730  012737  000504  002222  16740     MOV     #504,CYLMAP     ;GET MAPPING FACTOR FOR CYLINDERS
 2607  007736  012737  012527  002224  16750     MOV     #5463.,SECMAP   ;GET MAPPING FACTOR FOR SECTORS
 2608  007744  012737  031463  002226  16760     MOV     #13107.,FNMAP   ;GET MAPPING FACTOR FOR FUNCTION
 2609                                    16770
 2610  007752  013700  002554          16780     MOV     MAXBA,R0        ;COMPUTE THE MAXIMUM ALLOWABLE
 2611  007756  163700  002552          16790     SUB     BASEBA,R0       ;WORD COUNT FOR DATA TRANSFERS
 2612  007762  000241                    16800     CLC
 2613  007764  006000                    16810     ROR     R0              ;CONVERT TO WORDS
 2614
 2615  007766  012746  177777                    MOV     #177777,-(SP)   ;PUT LOW DIVIDEND ON STACK
 2616  007772  005046                            CLR     -(SP)           ;CLEAR HIGH DIVIDEND AND PUSH
 2617                                                                    ;IT ON STACK
 2618  007774  010046                            MOV     R0,-(SP)        ;PUSH DIVISOR ON STACK
 2619  007776  004737  024640                    JSR     PC,@#$DIV       ;GO TO THE 'DIVIDE' SUBROUTINE
 2620  010002  005726                            TST     (SP)+           ;DISCARD THE REMAINDER. QUOTIENT IS
 2621                                                                    ;NOW ON TOP OF THE STACK
 2622  010004  005216                    16830     INC     (SP)
 2623  010006  012637  002230          16840     MOV     (SP)+,BAMAP     ;SAVE THE MAPPING FACTOR FOR BUS ADDRESS
```

```
2624                                                    16860   ;THE ENTIRE DISK (ALL DRIVES) IS WRITTEN WITH RANDOM PATTERNS. THE FIRST
2625                                                    16870   ;WORD OF EACH SECTOR GIVES THE NUMBER OF WORDS (2'S COMPLEMENT) WRITTEN IN
2626                                                    16880   ;THAT SECTORS. REST OF THE DATA WORDS FOR THE SECTOR ARE GENERATED USING
2627                                                    16890   ;THE ABSOLUTE DISK ADDRESS AS THE RANDOM SEED.
2628                                                    16900   ;IF THE PROGRAM WAS RESTARTED AT 210 THEN CHECK IF SW 4 IS SET. IF IT IS
2629                                                    16910   ;THEN DO NOT REWRITE THE DISK WITH RANDOM PATTERNS. IF SW 4 IS NOT SET THEN
2630                                                    16920   ;ALL THE DISKS (PRESENT) ARE WRITTEN WITH RANDOM PATTERNS.
2631                                                    16930
2632  010012  105737  001753                          16940   WRDSK:  TSTB    FRSTRT          ;RESTARTED AT 210?
2633  010016  001407                                   16950           BEQ     1$              ;NO
2634  010020  105037  001753                          16960           CLRB    FRSTRT          ;YES, CLEAR THE RESTART FLAG
2635  010024  032777  000020  171106                  16970           BIT     #SW4,@SWR       ;SW 4 SET?
2636  010032  001401                                   16980           BEQ     1$              ;NO
2637  010034  000537                                   16990           BR      BEGEX1          ;YES, DONT REWRITE ALL DISKS WITH
2638                                                    17000                                   ;RANDOM PATTERNS
2639  010036  012737  010054  001110                  17010   1$:     MOV     #2$,$LPERR      ;LUP TO '2$' ON EROR, SW 9 SET!
2640  010044  012702  001754                          17020           MOV     #PDR,R2         ;POINTER TO DRIVE #'S TABLE
2641  010050  013703  001764                          17030           MOV     DRVPRS,R3       ;# OF DRIVES PRESENT
2642  010054  012700  011410                          17040   2$:     MOV     #4872.,R0       ;NUMBER OF SECTORS PER DISK
2643  010060  112201                                   17050           MOVB    (R2)+,R1        ;GET THE FIRST AVAILABLE DRIVE
2644  010062  042701  177770                          17060           BIC     #177770,R1      ;POSITION THE BITS (15,14,13)
2645  010066  010137  002202                          17070           MOV     R1,@DRV
2646  010072  000241                                   17080           CLC
2647  010074  006001                                   17090           ROR     R1
2648  010076  006001                                   17100           ROR     R1
2649  010100  006001                                   17110           ROR     R1
2650  010102  006001                                   17120           ROR     R1
2651  010104  010177  171620                          17130           MOV     R1,@RKDA        ;BASE DISK ADDRESS
2652                                                    17140
2653  010110  013704  002554                          17150           MOV     MAXBA,R4        ;CALCULATE MAXIMUM BUFFER
2654  010114  163704  002552                          17160           SUB     BASEBA,R4
2655  010120  006204                                   17170           ASR     R4              ;CONVERT TO WORDS
2656  010122  042704  000377                          17180           BIC     #377,R4         ;KEEP ONLY WHOLE BLOCKS
2657  010126  000304                                   17190           SWAB    R4
2658  010130  020427  000014                          17200           CMP     R4,#12.         ;MAX OF 12 SECTORS
2659  010134  003402                                   17210           BLE     3$
2660  010136  012704  000014                          17220           MOV     #12.,R4
2661                                                    17230
2662  010142  010401                                   17240   3$:     MOV     R4,R1
2663  010144  020400                                   17250           CMP     R4,R0
2664  010146  003401                                   17260           BLE     4$
2665  010150  010001                                   17270           MOV     R0,R1
2666  010152  000301                                   17280   4$:     SWAB    R1
2667  010154  005401                                   17290           NEG     R1
2668  010156  010137  010172                          17300           MOV     R1,5$
2669  010162  010177  171536                          17310           MOV     R1,@RKWC
2670  010166  004537  016350                          17320           JSR     R5,GENBUF       ;GENERATE RANDOM DATA BUFER
2671  010172  000000                                   17330   5$:     .WORD   0
2672  010174  032106                                   17340           .WORD   DBUF            ;STARTING ADDRESS OF DATA BUFER
2673                                                    17350
2674  010176  012777  032106  171522                  17360           MOV     #DBUF,@RKBA     ;FROM THIS BUS ADDRESS
2675  010204  012777  000003  171510                  17370           MOV     #3,@RKCS        ;WRITE, GO
2676                                                    17380
2677  010212  104416                                   17390           CON.RDY                 ;WAIT FOR CONTROL RDY
2678  010214  004737  017604                          17400           JSR     PC,CHKCS        ;ANY ERROR?
2679  010220  104001                                   17410           ERROR   1               ;AN ERROR OCCURED WHILE DOING WRITE
```

```
 2680                                          17420                                  ;ON THE DISK. YOU ARE ADVISED TO USE
 2681                                          17430                                  ;BASIC AND DYNAMIC TESTS.
 2682                                          17440
 2683  010222  032777  000400  170710          17450          BIT     #BIT8,@SWR      ;TYPE CURRENT STATUS
 2684  010230  001435                          17460          BEQ     6$
 2685  010232  032700  000377                  17470          BIT     #377,R0
 2686  010236  001032                          17480          BNE     6$
 2687  010240  104400  010246                                 TYPE    ,65$            ;;TYPE ASCIZ STRING
 2688  010244  000421                                        BR      64$             ;;GET OVER THE ASCIZ
 2689                                                 ;;65$:  .ASCIZ  <15><12>/WRITTING RANDOM PATTERN, DRIVE /
 2690  010310                                                 64$:
 2691  010310  013746  002202                  17500          MOV     @DRV,-(SP)
 2692  010314  104402                          17510          TYPOS
 2693  010316     001                          17520          .BYTE   1
 2694  010317     000                          17530          .BYTE   0
 2695                                          17540
 2696  010320  104400  001213                  17550          TYPE    ,$CRLF
 2697  010324  160400                          17560  6$:     SUB     R4,R0           ;DECREMENT SECTOR COUNT
 2698  010326  003305                          17570          BGT     3$
 2699                                          17580
 2700  010330  005303                          17590          DEC     R3              ;MORE DRIVES TO DO?
 2701  010332  001250                          17600          BNE     2$
 2702                                          17610
 2703                                          17620  ;IF SW 3 IS SET, ENABLE THE LINE CLOCK AND INITIALIZE COUNTS.
 2704                                          17630
 2705  010334  032777  000010  170576          17640  BEGEX1: BIT     #SW3,@SWR       ;KW11L CLOCK PRESENT?
 2706  010342  001403                          17650          BEQ     BEGNEX          ;IF NOT, SKIP. NOTE:IF KW11L IS
 2707                                          17660                                  ;NOT PRESENT SW3 SHOULD NOT BE SET
 2708                                          17670                                  ;OTHERWISE, A TIMEOUT WILL OCCUR.
 2709  010344  052777  000100  171362          17680          BIS     #BIT6,@KWLS     ;ENABLE KW11L CLOCK
 2710                                          17690                                  ;SERVICED AT PRIORITY 7 (KWSRVE)
```

```
2711                                            17710   ;THE PROGRAM IS GOING TO LOOP BA   TO THIS POINT AT THE END OF A PASS.
2712                                            17720
2713  010352  104415                            17730   BEGNEX: CON.RESET               ;CLEAR EVERYTHING IN DRIVES
2714  010354  012706  001100                    17740           MOV     #STACK,SP       ;RESET STACK
2715  010360  005737  001764                    17750           TST     DRVPRS          ;ANY DRIVES LEFT IN SYSTEM?
2716  010364  001402                            17760           BEQ     QMNGER
2717  010366  004737  014174                    17770           JSR     PC,GEN8RQ       ;GO GENERATE 8 COMMANDS FOR THE Q
2718                                            17780
2719                                            17790
2720                                            17800
2721                                            17810   ;CHECK IF THERE IS ANY UNFINISHED COMMAND IN THE Q, WAITING TO BE EXECUTED.
2722                                            17820   ;IF THERE IS NONE, GO TO THE 'GEN8RQ' AND GENERATE 8 REQUESTS (COMMANDS),
2723                                            17830   ;AND PUT THEM IN THE Q. IF THERE IS AN UNFINISHED COMMAND, FIND OUT IF
2724                                            17840   ;THERE IS A PRIORITY COMMAND TO BE EXECUTED IMMEDIATELY.
2725                                            17850
2726  010372  013746  001744                    17860   QMNGER: MOV     PPRLVL,-(SP)    ;NEW PSW , RAISE PRIORITY
2727  010376  012746  010404                    17870           MOV     #1$,-(SP)       ;RETURN PC ***************
2728  010402  000002                            17880           RTI
2729                                            17890
2730  010404  005737  001764                    17900   1$:     TST     DRVPRS          ;ANY DRIVES IN SYSTEM?
2731  010410  001040                            17910           BNE     2$
2732  010412  104400  010420                                    TYPE    ,65$            ;;TYPE ASCIZ STRING
2733  010416  000432                                            BR      64$             ;;GET OVER THE ASCIZ
2734                                                  ;;65$:    .ASCIZ  <15><12>/HALTING - PRESS CONTINUE TO RESTART AT '200'/<15><12><1
2735  010504                                          64$:
2736  010504  000000                            17930           HALT
2737  010506  000137  003470                    17940           JMP     START
2738                                            17950
2739  010512  012700  002006                    17960   2$:     MOV     #KEY,R0
2740  010516  005720                            17970   3$:     TST     (R0)+           ;ANY UNFINISHED COMMAND
2741  010520  100006                            17980           BPL     4$              ;IN Q
2742  010522  020027  002026                    17990           CMP     R0,#KEY+20
2743  010526  001373                            18000           BNE     3$
2744  010530  004737  014174                    18010           JSR     PC,GEN8RQ       ;IF NOT, GO GENERATE 8 MORE COMMANDS
2745  010534  000460                            18020           BR      CHFAFN
2746                                            18030
2747                                            18040
2748  010536  012700  002006                    18050   4$:     MOV     #KEY,R0
2749  010542  032710  010000                    18060   5$:     BIT     #BIT12,(R0)     ;ANY HIGH PRIORITY COMMAND IN Q
2750  010546  001404                            18070           BEQ     6$
2751  010550  005710                            18080           TST     (R0)            ;FINISHED?
2752  010552  100402                            18090           BMI     6$              ;YES
2753  010554  105710                            18100           TSTB    (R0)            ;IN EXECUTION?
2754  010556  100165                            18110           BPL     PRICMND         ;NO, GO PROCESS HIGH PRIORITY
2755                                            18120
2756  010560  005720                            18130   6$:     TST     (R0)+           ;CHECK THE ENTIRE Q
2757  010562  020027  002026                    18140           CMP     R0,#KEY+20
2758  010566  001365                            18150           BNE     5$
2759                                            18160
2760                                            18170
2761                                            18180   ;FIND OUT IF THERE IS A WRITE CHECK FUNCTION TO BE DONE IMMEDIATELY AFTER
2762                                            18190   ;A WRITE, THAT WAS DONE PREVIOUSLY.
2763                                            18200
2764  010570  005737  002156                    18210           TST     WCFLG           ;IS WRITE CHECK TO FOLLOW
2765  010574  100040                            18220           BPL     CHFAFN          ;WRITE? IF NOT, BRANCH
2766                                            18230                                   ;YES
```

```
 2767  010576  013700  002156          18240          MOV     WCFLG,R0        ;GET WC FLAG
 2768  010602  042700  177400          18250          BIC     #177400,R0      ;LOWER BYTE CONTAINS KEY#X2 (OFFSET FROM
 2769                                  18260                                  ;KEY), OF THE WRITE FUNCTION
 2770  010606  016001  002532          18270          MOV     PCMND(R0),R1    ;GET POINTER
 2771  010612  062700  002006          18280          ADD     #KEY,R0         ;FROM ADDRESS OF THE KEY
 2772                                  18290                                  ;WHICH WAS USED FOR PREVIOUS
 2773                                  18300                                  ;WRITE
 2774  010616  022761  000002  000002  18310          CMP     #2,2(R1)        ;CHECK THAT THE FUNCTION
 2775  010624  001413                  18320          BEQ     7$              ;WAS A WRITE
 2776  010626  011037  001162          18330          MOV     (R0),$REG0      ;GET KEY CONTENTS
 2777  010632  016137  000002  001164  18340          MOV     2(R1),$REG1     ;GET THE FUNCTION INDICATED BY THIS KEY
 2778  010640  104027                  18350          ERROR   27              ;IF NOT, ERROR
 2779                                  18360                                  ;SEFORE DOING A WRITE CHECK A WRITE IS
 2780                                  18370                                  ;ALWAYS DONE. OCCURANCE OF THIS ERROR
 2781                                  18380                                  ;INDICATES THAT SOMEHOW AN ATTEMPT IS BEING
 2782                                  18390                                  ;MADE TO DO WRITE CHECK BEFORE A WRITE IS
 2783                                  18400                                  ;PERFORMED. THE KEY IN ERROR MESSAGE WAS
 2784                                  18410                                  ;THE ONE WHICH GAVE RISE TO THIS ATTEMPT.
 2785                                  18420                                  ;THE FUNCTION CODE IS THE FUNCTION ASSOSCIATED
 2786                                  18430                                  ;WITH THAT KEY
 2787  010642  005037  002156          18440          CLR     WCFLG           ;ABORT THIS WRITE CHECK
 2788  010646  052710  100000          18450          BIS     #BIT15,(R0)
 2789  010652  000647                  18460          BR      QMNGER
 2790  010654  012761  000006  000002  18470  7$:     MOV     #6,2(R1)        ;SET UP BITS FOR WRT CHK
 2791                                  18480                                  ;NOW
 2792  010662  042710  174340          18490          BIC     #174340,(R0)    ;CLEAR OUT THE UNNECESSARY
 2793                                  18500                                  ;FLAGS FROM THE KEY
 2794  010666  052710  000100          18510          BIS     #BIT6,(R0)      ;SET FLAG FOR WRITE CHECK, FOR
 2795                                  18520                                  ;THIS KEY
 2796  010672  000137  011332          18530          JMP     EXCUT
 2797                                  18540
 2798                                  18550                                  ;NO HIGH PRIORITY COMMAND IN Q
```

```
 2799                                                 18570  ;NO HIGH PRIORITY COMMAND WAS FOUND IN THE Q. FIND OUT THE FIPST AVAILABLE
 2800                                                 18580  ;COMMAND IN THE Q, FOR EXECUTION.
 2801  010676  005000                                 18590  CHFAFN: CLR     R0              ;WAIT FOR ANY IMMEDIATE INTERUPT
 2802  010700  005046                                 18600          CLR     -(SP)           ;NEW PSW
 2803  010702  012746  010710                         18610          MOV     #1$,-(SP)       ;RETURN FOR RTI
 2804  010706  000002                                 18620          RTI
 2805                                                 18630
 2806                                                 18640
 2807  010710  105200                                 18650  1$:     INCB    R0
 2808  010712  001376                                 18660          BNE     .-2
 2809  010714  013746  001744                         18670          MOV     PPRLVL,-(SP)    ;NEW PSW, LOCK OUT CPU
 2810  010720  012746  010726                         18680          MOV     #2$,-(SP)       ;RETURN FOR RTI *************
 2811  010724  000002                                 18690          RTI
 2812                                                 18700
 2813                                                 18710
 2814  010726  012700  002006                         18720  2$:     MOV     #KEY,R0
 2815  010732  005710                                 18730  CH1:    TST     (R0)            ;UNFINISHED COMMAND?
 2816  010734  100436                                 18740          BMI     CH2
 2817  010736  105710                                 18750          TSTB    (R0)            ;IN PROGRESS?
 2818  010740  100434                                 18760          BMI     CH2
 2819  010742  011001                                 18770          MOV     (R0),R1
 2820  010744  042701  177770                         18780          BIC     #177770,R1
 2821  010750  105761  002126                         18790          TSTB    BUSY(R1)        ;IS THIS DRIVE BUSY?
 2822  010754  100426                                 18800          BMI     CH2
 2823  010756  105761  002136                         18810          TSTB    POS(R1)         ;HAS THIS DRIVE BEEN POSITIONED
 2824                                                 18820                                  ;FOR ANY OTHER COMMAND.  IF YES,
 2825                                                 18830                                  ;SKIP.  IF NOT, PROCEED
 2826  010762  001023                                 18840          BNE     CH2
 2827                                                 18850
 2828                                                 18860                                  ;CHECK IF THERE IS AY OTHER COMMAND
 2829  010764  010002                                 18870          MOV     R0,R2           ;ON A DRIVE THAT IS NOT THE SAME
 2830                                                 18880                                  ;AS THE PREVIOUS DRIVE.  THIS COMMAND
 2831  010766  000414                                 18890          BR      2$              ;SHOULD NOT BE IN PROGRESS
 2832                                                 18900                                  ;AND SHOULD NOT HAVE BEEN COMPLETED
 2833  010770  005712                                 18910  1$:     TST     (R2)            ;UNFINISHED COMMAND?
 2834  010772  100412                                 18920          BMI     2$
 2835  010774  105712                                 18930          TSTB    (R2)            ;IN PROGRESS?
 2836  010776  100410                                 18940          BMI     2$
 2837  011000  011203                                 18950          MOV     (R2),R3
 2838  011002  042703  177770                         18960          BIC     #177770,R3
 2839  011006  105763  002126                         18970          TSTB    BUSY(R3)        ;IS THE DRIVE BUSY?
 2840  011012  100402                                 18980          BMI     2$
 2841  011014  020301                                 18990          CMP     R3,R1           ;IS THIS COMMAND ON THE SAME DRIVE?
 2842  011016  001447                                 19000          BEQ     POSTION         ;IF YES, GO AND POSITION THE COMMAND
 2843                                                 19010                                  ;POINTED TO BY R0, (BECAUSE THERE IS
 2844                                                 19020                                  ;ONE MORE COMMAND THAT CAN BE PERFORMED
 2845                                                 19030                                  ;ON THE SAME DRIVE)
 2846  011020  005722                                 19040  2$:     TST     (R2)+           ;CHECK REST OF THE Q
 2847  011022  020227  002026                         19050          CMP     R2,#KEY+20
 2848  011026  001360                                 19060          BNE     1$
 2849                                                 19070                                  ;IF THERE WAS NO EXECUTABLE COMMAND
 2850  011030  000470                                 19080          BR      EXNSK           ;ON ANY OTHER DRIVE, THEN EXECUTE
 2851                                                 19090                                  ;COMMAND POINTED TO BY R0
 2852                                                 19100
 2853  011032  005720                                 19110  CH2:    TST     (R0)+           ;CHECK ENTIRE Q
 2854  011034  020027  002026                         19120          CMP     R0,#KEY+20
```

```
2855  011040  001334                    19130          BNE     CH1
2856                                    19140
2857                                    19150
2858                                    19160  ;NO (UNPOSITIONED) EXECUTABLE COMMAND WAS FOUND IN THE Q. CHECK IF THERE
2859                                    19170  ;IS ANY POSITIONED COMMAND IN THE Q, WAITING TO BE EXECUTED.
2860                                    19180
2861  011042  105777  170654            19190          TSTB    @RKCS           ;IF THE CONTROLLER IS BUSY GO TO
2862  011046  100402                    19200          BMI     1$              ;STATUS AND WAIT FOR INTERRUPTS
2863  011050  000137  021066            19210          JMP     STATUS          ;IF THE CONTROLLER IS NOT BUSY FIND
2864                                    19220
2865  011054  012700  002006            19230  1$:     MOV     #KEY,R0         ;ANY POSITIONED COMMAND AND EXECUTE
2866  011060  032710  000020            19240  2$:     BIT     #BIT4,(R0)
2867  011064  001414                    19250          BEQ     3$              ;IT
2868  011066  005710                    19260          TST     (R0)            ;MAKE SURE COMMAND IS POSITIONED
2869  011070  100412                    19270          BMI     3$
2870  011072  105710                    19280          TSTB    (R0)            ;COMMAND IS UNFINISHED,
2871  011074  100410                    19290          BMI     3$              ;IT IS NOT IN PROGRESS,
2872  011076  011001                    19300          MOV     (R0),R1         ;THE DRIVE IS NOT IN BUSY
2873  011100  042701  177770            19310          BIC     #177770,R1
2874  011104  105761  002126            19320          TSTB    BUSY(R1)
2875  011110  100402                    19330          BMI     3$
2876  011112  000137  011332            19340          JMP     EXCUT           ;GO EXECUTE THE COMMAND IF THE
2877                                    19350
2878  011116  005720                    19360  3$:     TST     (R0)+           ;ABOVE CONDITIONS ARE SATISFIED
2879  011120  020027  002026            19370          CMP     R0,#KEY+20              ;CHECK ALL COMMANDS IN Q
2880  011124  001355                    19380          BNE     2$
2881  011126  000137  021066            19390          JMP     STATUS
2882                                    19400
2883                                    19410
2884                                    19420
2885                                    19430  ;ENTER HERE IF THERE IS A PRIORITY COMMAND TO BE EXECUTED.
2886                                    19440
2887                                    19450
2888  011132  000137  011332            19460  PRICMN: JMP     EXCUT           ;GO EXECUTE NON-SEEK COMMAND.
2889                                    19470
2890                                    19480  ;ENTER THIS IF A COMMAND IS TO BE PREPOSITIONED (BEFORE EXECUTING A
2891                                    19490  ;FUNCTION)
2892                                    19500
2893  011136  032710  000020            19510  POSTION: BIT    #BIT4,(R0)      ;ALREADY POSITIONED?
2894  011142  001333                    19520          BNE     CH2             ;YES, GO BACK AND CHECK IF REST OF THE
2895                                    19530                                  ;COMMANDS IN THE Q ARE TO BE POSITIONED
2896                                    19540
2897  011144  004737  011704            19550          JSR     PC,POSK         ;GO CHECK, & SET UP FLAGS
2898  011150  004737  020042            19560          JSR     PC,POSCMND      ;SAVE INFO ABOUT PRESENT & PAST COMAND
2899  011154  012777  000111  170540    19570          MOV     #111,@RKCS      ;POSITION THE COMMAND
2900  011162  005046                    19580          CLR     -(SP)           ;NEW PSW, DROP PRIORITY
2901  011164  012746  011172            19590          MOV     #1$,-(SP)       ;RETURN FOR RTI
2902  011170  000002                    19600          RTI
2903  011172                            19610  1$:
2904  011172  105737  002235            19620          TSTB    INTIFL
2905  011176  001775                    19630          BEQ     .-4             ;WAIT FOR INTERRUPT
2906  011200  012777  012776  170532    19640          MOV     #INTHND,@RKVEC  ;RE-ESTABLISH RK INTERRUPT VECTOR
2907  011206  000137  010676            19650          JMP     CHFAFN          ;GO BACK AND CHECK IF ANY COMMANDS TO BE
2908                                    19660                                  ;POSITIONED OR EXECUTED
2909                                    19670
2910                                    19680
```

```
 2911                                                 19690                                  ;IS THERE ANY POSITIONED COMMAND IN
 2912                                                 19700                                  ;THE Q.  FIND OUT? IF THERE IS
 2913                                                 19710                                  ;ONE EXECUTE THE POSITIONED COMMAND,
 2914                                                 19720                                  ;BUT FIRST POSITION THE COMMAND (KEY)
 2915                                                 19730                                  ;POINTED TO BY R0
 2916                                                 19740
 2917  011212  012701  002006                       19750  EXNSK:  MOV     #KEY,R1
 2918  011216  032711  000020                       19760  1$:     BIT     #BIT4,(R1)      ;HEADS POSITIONED?
 2919  011222  001412                               19770          BEQ     2$
 2920  011224  005711                               19780          TST     (R1)            ;FUNCTION COMPLETED?
 2921  011226  100410                               19790          BMI     2$              ;YES, BRANCH
 2922  011230  105711                               19800          TSTB    (R1)            ;FUNCTION IN PROGRESS?
 2923  011232  100406                               19810          BMI     2$
 2924  011234  011102                               19820          MOV     (R1),R2
 2925  011236  042702  177770                       19830          BIC     #177770,R2
 2926  011242  105762  002126                       19840          TSTB    BUSY(R2)        ;DRIVE BUSY?
 2927  011246  100005                               19850          BPL     3$
 2928                                                 19860
 2929  011250  005721                               19870  2$:     TST     (R1)+           ;CHECK ALL COMMANDS IN Q
 2930  011252  020127  002026                       19880          CMP     R1,#KEY+20
 2931  011256  001357                               19890          BNE     1$
 2932                                                 19900
 2933  011260  000424                               19910          BR      EXCUT           ;NO POSITIONED COMMAND WAS
 2934                                                 19920                                  ;FOUND IN THE Q. SO EXECUTE
 2935                                                 19930                                  ;COMMAND POINTED TO BY R0
 2936                                                 19940
 2937                                                 19950
 2938                                                 19960                                  ;AN ALREADY POSITIONED COMMAND HAS
 2939                                                 19970                                  ;BEEN FOUND.
 2940  011262  010137  002236                       19980  3$:     MOV     R1,SAVKEY       ;SAVE POINTER TO THE COMMAND(KEY)
 2941                                                 19990                                  ;WHICH IS ALREADY POSITIONED
 2942  011266  004737  011704                       20000          JSR     PC,POSK         ;GO AND POSITION THE COMMAND(KEY)
 2943                                                 20010                                  ;POINTED TO BY R0
 2944  011272  004737  020042                       20020          JSR     PC,POSCMND      ;SAVE INFO ABOUT PAST & PRESENT COMAND
 2945  011276  012777  000111  170416               20030          MOV     #111,@RKCS      ;SEEK, IDE, GO
 2946  011304  005046                               20040          CLR     -(SP)           ;ALLOW FIRST INTERRUPT
 2947  011306  012746  011314                       20050          MOV     #4$,-(SP)       ;RETURN FOR RTI **********
 2948  011312  000002                               20060          RTI
 2949                                                 20070
 2950                                                 20080
 2951  011314  105737  002235                       20090  4$:     TSTB    INT1FL          ;DID INTERRUPT OCCUR?
 2952  011320  012777  012776  170412               20100          MOV     #INTHND,@RKVEC  ;REESTABLISH INTERRUPT ADDRESS
 2953  011326  013700  002236                       20110          MOV     SAVKEY,R0       ;RESTORE THE SAVED POINTER TO KEY
```

```
2954
2955
2956
2957  011332  011001                20130                           ;EXECUTE THE COMMAND POINTED TO BY R0
2958  011334  042701  177770        20140                           ;R0 CONTAINS THE POINTER TO THE
2959  011340  005710                20150                           ;COMMAND KEY
2960  011342  100004                20160  EXCUT:  MOV     (R0),R1   ;GET DRIVE NO
2961  011344  011037  001162        20170          BIC     #177770,R1
2962  011350  104030                20180          TST     (R0)      ;THIS COMMAND UNFINISHED?
2963                                20190          BPL     1$
2964                                20200          MOV     (R0),$REG0  ;GET KEY
2965                                20210          ERROR   30        ;IF NOT, ERROR
2966  011352  000512                20220                            ;AN ATTEMPT WAS MADE TO REEXECUTE A COMMAND
2967  011354  105710                20230                            ;THAT HAS ALREADY BEEN EXECUTED BEFORE.
2968  011356  100004                20240                            ;(ON DRIVE NO. GIVEN IN ERROR MESSAGE)
2969  011360  011037  001162        20250          BR      ABRT1
2970  011364  104030                20260  1$:     TSTB    (R0)      ;IS IT IN PROGRESS?
2971                                20270          BPL     3$
2972  011366  000504                20280          MOV     (R0),$REG0  ;GET KEY
2973                                20290          ERROR   30        ;IF YES, ERROR
2974                                20300
2975                                20310          BR      ABRT1     ;AN ATTEMPT WAS MADE TO REEXECUTE A COMMAND
2976  011370  105761  002126        20320                            ;THAT IS IN PROGRESS (ON DRIVE NO. GIVEN
2977  011374  100004                20330                            ;IN EROR MESAGE)
2978  011376  010137  001162        20340
2979  011402  104002                20350  3$:     TSTB    BUSY(R1)  ;IS THE DRIVE BUSY?
2980  011404  000470                20360          BPL     4$
2981                                20370          MOV     R1,$REG0  ;GET DRIVE #
2982                                20380          ERROR   2         ;'BUSY' FLAG FOR THE DRIVE (CONTAINED
2983                                20390          BR      ABRT      ;IN R1) IS SET, INDICATING THAT THE DRIVE
2984                                20400                            ;IS 'BUSY' AND ENGAGED IN AN ACTIVITY,
2985                                20410                            ;STILL AN ATTEMPT WAS MADE TO INITIATE
2986  011406  105777  170310        20420                            ;A FUNCTION ON THIS DRIVE.  THIS IS AN
2987  011412  100404                20430                            ;UNEXCEPTED ERROR CONDITION.
2988  011414  004737  021462        20440
2989  011420  104003                20450  4$:     TSTB    @RKCS     ;CONTROL READY SET?
2990  011422  000461                20460          BMI     5$        ;YES
2991                                20470          JSR     PC,GT4RG  ;GET RKCS, ER, DS, DA
2992                                20480          ERROR   3         ;CONTROL READY IS NOT SET.  THIS IS AN
2993  011424  011002                20490          BR      ABRT      ;UNEXPECTED ERROR CONDITION AT THIS
2994  011426  000302                20500                            ;POINT OF EXECUTION, CONTROL SHOULD
2995  011430  042702  177770        20510                            ;BE NORMALLY READY.
2996  011434  006302                20520  5$:     MOV     (R0),R2
2997  011436  010204                20530          SWAB    R2
2998  011440  016205  002532        20540          BIC     #177770,R2  ;FORM THE ADDRESS OF THE
2999                                20550          ASL     R2          ;PARAMETER LIST FOR THIS
3000  011444  011577  170260        20560          MOV     R2,R4       ;COMMAND
3001                                20570          MOV     PCMND(R2),R5
3002  011450  032777  000100  170240  20580
3003  011456  001006                20590          MOV     (R5),@RKDA  ;GET THE FIRST ITEM FROM LIST
3004  011460  010137  001750        20600                              ;DISK ADDRES
3005  011464  004737  021462        20610          BIT     #RWS,@RKDS  ;R/W/S RDY SET?
3006  011470  104004                20620          BNE     6$          ;YES
3007  011472  000435                20630          MOV     R1,SRDRV    ;GET DRIVE #, FOR TYPING SERIAL #
3008                                20640          JSR     PC,GT4RG    ;GET RKCS, ER, DS, DA
3009                                20650          ERROR   4           ;R/W/S RDY IS NOT SET FOR THE DRIVE.
                                    20660          BR      ABRT        ;A (NON-SEEK) FUNCTION WAS SUPPOSED
                                    20670                              ;TO BE EXECUTED ON THIS DRIVE.  THIS IS
                                    20680                              ;AN UNEXPECTED ERROR CONDITIONS AT THIS
```

```
                                                                20690                                   ;POINT OF EXECUTION R/W/S RDY SHOULD
                                                                20700                                   ;BE NORMALLY SET.
3012  011474  016503  000002                                    20710  6$:      MOV     2(R5),R3        ;GET FUNCTION TO BE DONE
3013                                                            20720
3014  011500  122703  000002                                    20730           CMPB    #2,R3           ;IS IT A WRITE FUNCTION?
3015  011504  001437                                           20740           BEQ     WRFNC           ;YES, IT IS WRITE
3016                                                            20750                                   ;IT'S NOT A WRITE FUNCTION
3017                                                            20760
3018  011506  016503  000002                                    20770  NWRFNC:  MOV     2(R5),R3        ;GET FUNCTION TO BE DONE
3019  011512  052703  000501                                    20780           BIS     #501,R3         ;SET SSE,IDE,GO BITS
3020  011516  016577  000004  170200                            20790           MOV     4(R5),@RKWC     ;GET WORD COUNT
3021  011524  016577  000006  170174                            20800           MOV     6(R5),@RKBA     ;GET BUS ADDRESS
3022                                                            20810
3023  011532  004737  020064                                    20820           JSR     PC,FNCMND       ;SAVE INFO ABOUT PAST & PRESENT COMAND
3024                                                            20830
3025  011536  010377  170160                                    20840           MOV     R3,@RKCS        ;SET SSE,IDE, FUNCTION, GO
3026                                                            20850
3027  011542  052710  000200                                    20860           BIS     #BIT7,(R0)      ;SET FLAG INDICATING FUNCTION
3028  011546  052704  000200                                    20870           BIS     #BIT7,R4        ;IN PROGRESS
3029                                                            20880                                   ;R4 CONTAINS OFFSET TO THE COMMAND KEY
3030                                                            20890                                   ;(FROM 'KEY') BITS 0-3, BIT 7 IS SET
3031  011552  110461  002126                                    20900           MOVB    R4,BUSY(R1)     ;SET FLAG INDICATING DRIVE BUSY
3032  011556  110437  002234                                    20910           MOVB    R4,INTFLG       ;SET FLAG FOR INTERUPT USE
3033  011562  000137  021066                                    20920           JMP     STATUS
3034                                                            20930
3035  011566  004737  014122                                    20940  ABRT:    JSR     PC,DRVABT       ;ABORT THE FUNCTION
3036  011572  004737  016204                                    20950           JSR     PC,CHKDRV       ;GO CHECK, DRIVE ERRORS
3037  011576  000240                                           20960           NOP
3038  011600  000137  010372                                    20970  ABRT1:   JMP     @MNGER
```

```
 3039                                                  20990                                  ;THE FUNCTION TO BE EXECUTED IS A
 3040                                                  21000                                  ;WRITE FUNCTION
 3041  011604  016537  000004  011624                 21010  WRFNC:  MOV     4(R5),1$        ;GET # OF WORDS TO BE WRITTEN
 3042  011612  016537  000006  011626                 21020          MOV     6(R5),2$        ;GET STARTING BA OF BUFFER
 3043  011620  004537  016350                         21030          JSR     R5,GENBUF       ;GO GENERATE BUFFER
 3044  011624  000000                                 21040  1$:     .WORD   0               ;SAVE # OF WORDS
 3045  011626  000000                                 21050  2$:     .WORD   0               ;SAVE STARTING BUS ADDRESS OF BUFFER
 3046  011630  052703  000101                         21060          BIS     #101,R3         ;SET IDE AND GO BIT
 3047  011634  013777  011624  170062                 21070          MOV     1$,@RKWC        ;SET WORD COUNT
 3048  011642  013777  011626  170056                 21080          MOV     2$,@RKBA        ;SET BUS ADDRESS
 3049  011650  004737  020064                         21090          JSR     PC,FNCMND       ;SAVE INFO ABOUT PAST & PRESENT COMAND
 3050                                                 21100
 3051  011654  010377  170042                         21110          MOV     R3,@RKCS        ;ISSUE WRITE,IDE,GO
 3052  011660  052710  000200                         21120          BIS     #BIT7,(R0)      ;SET FLAG INDICATING FUNCTION IN
 3053  011664  052704  000200                         21130          BIS     #BIT7,R4        ;PROGRESS
 3054                                                 21140
 3055  011670  110461  002126                         21150          MOVB    R4,BUSY(R1)     ;SET FLAG INDICATING DRIVE BUSY
 3056  011674  110437  002234                         21160          MOVB    R4,INTFLG       ;SET FLAG FOR INTERUPT USE
 3057                                                 21170                                  ;R4 CONTAINS OFFSET TO THE COMMAND KEY
 3058                                                 21180                                  ;(FROM 'KEY') BITS 0-3, BIT 7 IS SET
 3059  011700  000137  021066                         21190          JMP     STATUS
```

```
 3060  011704  011001                  21210  PUSK:   MOV     (R0),R1         ;INITIAL CHECKING PRIOR TO DOING
 3061  011706  042701  177770          21220          BIC     #177770,R1      ;POSITIONING COMMAND POINTED TO BY R0
 3062  011712  105761  002126          21230          TSTB    BUSY(R1)        ;DRIVE BUSY?
 3063  011716  100004                  21240          BPL     1$
 3064  011720  010137  001162          21250          MOV     R1,$REG0        ;GET DRIVE # FOR WHICH 'BUSY' IS SET
 3065  011724  104002                  21260          ERROR   2               ;'BUSY' FLAG FOR THE DRIVE (CONTAINED
 3066  011726  000470                  21270          BR      7$              ;IN R1) IS SET, INDICATING THAT THE DRIVE
 3067                                  21280                                  ;IS 'BUSY' AND ENGAGED IN AN ACTIVITY,
 3068                                  21290                                  ;STILL AN ATTEMPT WAS MADE TO INITIATE
 3069                                  21300                                  ;POSITIONINGN ON THIS DRIVE.  THIS IS AN
 3070                                  21310                                  ;UNEXCEPTED ERROR CONDITION.
 3071  011730  105777  167766          21320  1$:     TSTB    @RKCS           ;CONTROL READY SET?
 3072  011734  100404                  21330          BMI     2$              ;YES
 3073  011736  004737  021462          21340          JSR     PC,GT4RG        ;GET RKCS, ER, DS, DA
 3074  011742  104003                  21350          ERROR   3               ;CONTROL READY IS NOT SET.  THIS IS AN
 3075  011744  000454                  21360          BR      6$              ;UNEXPECTED ERROR CONDITION AT THIS
 3076                                  21370                                  ;POINT OF EXECUTION, CONTROL SHOULD
 3077                                  21380                                  ;BE NORMALLY READY.
 3078  011746  011002                  21390  2$:     MOV     (R0),R2
 3079  011750  000302                  21400          SWAB    R2
 3080  011752  042702  177770          21410          BIC     #177770,R2
 3081  011756  006302                  21420          ASL     R2
 3082  011760  016203  002532          21430          MOV     PCMND(R2),R3    ;STARTING WORD OF COMMAND TABLE
 3083  011764  011377  167740          21440          MOV     (R3),@RKDA
 3084  011770  032777  000100  167720  21450          BIT     #RWS,@RKDS      ;R/W/S RDY SET?
 3085  011776  001006                  21460          BNE     3$              ;YES
 3086  012000  010137  001750          21470          MOV     R1,SRDRV        ;GET DRIVE #, FOR TYPING SERIAL #
 3087  012004  004737  021462          21480          JSR     PC,GT4RG        ;GET RKCS, ER, DS, DA
 3088  012010  104004                  21490          ERROR   4               ;R/W/S RDY IS NOT SET FOR THE DRIVE.  A
 3089  012012  000431                  21500          BR      6$              ;(POSITIONING) SEEK WAS SUPPOSED TO BE
 3090                                  21510                                  ;BE EXECUTED ON THIS DRIVE.  THIS IS
 3091                                  21520                                  ;AN UNEXPECTED ERROR CONDITION.  AT THIS
 3092                                  21530                                  ;POINT OF EXECUTION R/W/S RDY SHOULD
 3093                                  21540                                  ;BE NORMALLY SET.
 3094  012014  110261  002126          21550  3$:     MOVB    R2,BUSY(R1)     ;SET FLAG INDICATING DRIVE BUSY
 3095  012020  152761  000200  002126  21560          BISB    #BIT7,BUSY(R1)
 3096  012026  032710  000010          21570          BIT     #BIT3,(R0)      ;IS .HIS A SEEK COMMAND
 3097  012032  001006                  21580          BNE     4$
 3098  012034  112761  000377  002136  21590          MOVB    #377,POS(R1)    ;SET FLAG INDICATING, THIS DRIVE POSITIONS
 3099  012042  052710  000020          21600          BIS     #BIT4,(R0)      ;SET FLAG INDICATING THIS COMMAND
 3100  012046  000402                  21610          BR      5$
 3101  012050  052710  000200          21620  4$:     BIS     #BIT7,(R0)      ;POSITIONED.  SET FLAG INDICATING
 3102                                  21630                                  ;FUNCTION IN PROGRESS
 3103  012054  012777  012116  167656  21640  5$:     MOV     #INT1SK,@RKVEC  ;SET UP RK11 VECTOR
 3104  012062  013777  001744  167652  21650          MOV     PPRLVL,@RKSTAT  ;PSW ON INTERRUPT
 3105  012070  105037  002235          21660          CLRB    INT1FL          ;CLEAR FLAG INDICATING - INTERRUPT
 3106  012074  000207                  21670          RTS     PC              ;OCCURRED
 3107                                  21680
 3108  012076  004737  014122          21690  6$:     JSR     PC,DRVABT       ;ABORT THE FUNCTION
 3109  012102  004737  016204          21700          JSR     PC,CHKDRV       ;GO CHECK, DRIVE ERRORS
 3110  012106  000240                  21710          NOP
 3111  012110  005726                  21720  7$:     TST     (SP)+           ;POP THE RETURN ADDRESS
 3112  012112  000137  010372          21730          JMP     @MNGER
```

```
3113                                                    21750   ;AFTER ISSUING A SEEK FOR POSITIONING, AN INTERRUPT IS ALLLOWED TO TAKE
3114                                                    21760   ;*PLACE. THIS HANDLER SERVICES THE FIRST INTERRUPT, WHICH OCCURS AS A RESULT
3115                                                    21770   ;OF THE SEEK BEING ACCEPTED.
3116                                                    21780
3117  012116  105237  002235                          21790   INTISK: INCB    INTIFL          ;INDICATE THAT INTERRUPT TOOK PLACE
3118  012122  053766  001744  000002                  21800           BIS     PPRLVL,2(SP)    ;CPU PRIORITY TO 5 ON RETURN FROM
3119                                                    21810                                   ;INTERRUPT
3120                                                    21820
3121  012130  004737  017764                          21830           JSR     PC,CHKCRDY      ;CONTROL READY SET?
3122  012134  104005                                  21840           ERROR   5               ;CONTROL READY NOT SET AFTER THE FIRST
3123                                                    21850                                   ;INTERRUPT ON INITIATING SEEK
3124                                                    21860
3125  012136  032777  020000  167556                  21870           BIT     #SCP,@RKCS      ;SCP BIT SET?
3126  012144  001403                                  21880           BEQ     1$
3127  012146  004737  021652                          21890           JSR     PC,RG4SDR       ;GET RKCS, ER, DS, DA AND DRIVE # FOR
3128                                                    21900                                   ;TYPING SERIAL DRIVE #
3129  012152  104012                                  21910           ERROR   12              ;SCP SET AFTER FIRST INTERRUPT ON
3130                                                    21920                                   ;ACCEPTING SEEK.  A SEEK WAS INITIATED,
3131                                                    21930                                   ;AS A RESULT OF WHICH (THIS) FIRST
3132                                                    21940                                   ;INTERRUPT OCCURRED, BUT SCP SHOULD
3133                                                    21950                                   ;NOT BE SET AFTER THE FIRST INTERRUPT.
3134                                                    21960                                   ;(IT GETS SET ONLY AFTER THE SEEK
3135                                                    21970                                   ;IS DONE).
3136                                                    21980                                   ;THIS IS THE FIRST INTERRUPT
3137                                                    21990                                   ;AFTER ISSUING SEEK
3138  012154  017700  167542                          22000   1$:     MOV     @RKCS,R0
3139  012160  042700  177760                          22010           BIC     #177760,R0      ;CHECK THERE IS A SEEK FUNCTION
3140  012164  022700  000010                          22020           CMP     #10,R0          ;IN RKCS
3141  012170  001403                                  22030           BEQ     2$
3142  012172  004737  021462                          22040           JSR     PC,GT4RG        ;GET RKCS,ER,DS,DA
3143  012176  104006                                  22050           ERROR   6               ;RKCS DOES NOT CONTAIN 'SEEK' FUNCTION.
3144                                                    22060                                   ;A SEEK WAS INITIATED AS A RESULT OF WHICH
3145                                                    22070                                   ;(THIS) FIRST INTERRUPT OCCURRED.  RKCS
3146                                                    22080                                   ;SHOULD CONTAIN THE SEEK FUNCTION BITS,
3147                                                    22090                                   ;BUT IT DID NOT
3148                                                    22100
3149  012200  017700  167524                          22110   2$:     MOV     @RKDA,R0        ;GET THE DRIVE # FROM RKDA
3150  012204  042700  017777                          22120           BIC     #17777,R0
3151  012210  000241                                  22130           CLC
3152  012212  006100                                  22140           ROL     R0
3153  012214  006100                                  22150           ROL     R0
3154  012216  006100                                  22160           ROL     R0
3155  012220  006100                                  22170           ROL     R0
3156  012222  010001                                  22180           MOV     R0,R1
3157  012224  105761  002126                          22190           TSTB    BUSY(R1)        ;CHECK THIS DRIVE NO. WAS BUSY
3158  012230  100403                                  22200           BMI     3$              ;OK, IF IT WAS
3159  012232  010137  001162                          22210           MOV     R1,$REG0        ;GET DRIVE #(FROM RKDA)
3160  012236  104007                                  22220           ERROR   7               ;'BUSY' FLAG FOR THE DRIVE WAS NOT SET.
3161                                                    22230                                   ;THE DRIVE # (IN RKDA) WHICH CAUSED (?)
3162                                                    22240                                   ;THIS (FIRST) INTERRUPT SHOULD HAVE BEEN
3163                                                    22250                                   ;'BUSY' (SOFTWARE FLAG).  NOTE WHENEVER
3164                                                    22260                                   ;ANY OPERATION IS INITIATED ON ANY DRIVE
3165                                                    22270                                   ;'BUSY' FLAG FOR THAT DRIVE IS SET.
3166  012240  116100  002126                          22280   3$:     MOVB    BUSY(R1),R0
3167  012244  042700  177600                          22290           BIC     #177600,R0      ;FORM THE ADDRESS OF
3168  012250  062700  002006                          22300           ADD     #KEY,R0         ;THIS KEY
```

```
3169
3170  012254  032710  000020          22310
3171  012260  001003                  22320          BIT     #BIT4,(R0)      ;IS THE KEY ACTIVE FOR 'POSITIONING'?
3172  012262  010137  001162          22330          BNE     4$
3173  012266  104010                  22340          MOV     R1,$REG0        ;GET DRIVE NUMBER
3174                                  22350          ERROR   10              ;POSITIONING FLAG (IN THE KEY) IS NOT
3175                                  22360                                  ;SET FOR THE INTERRUPTING DRIVE (GIVEN
3176                                  22370                                  ;IN EROR MESAGE)
3177  012270  105761  002136          22380
3178  012274  001003                  22390  4$:     TSTB    POS(R1)         ;IS THE 'POSITIONING' FLAG SET?
3179  012276  010137  001162          22400          BNE     5$
3180  012302  104010                  22410          MOV     R1,$REG0        ;GET DRIVE # FROM RKDA
3181                                  22420          ERROR   10              ;THIS INTERRUPT IS SUPPOSED TO BE AS
3182                                  22430                                  ;A RESULT OF INITIATING A (POSITIONING)
3183                                  22440                                  ;SEEK.  WHENEVER A SEEK IS INITIATED
3184                                  22450                                  ;TO POSITION THE HEADS THE POSITIONING
3185                                  22460                                  ;FLAG (POS#) IS SET. OCCURANCE OF THIS
3186                                  22470                                  ;ERROR INDICATED THAT THE DRIVE #
3187                                  22480                                  ;(FROM RKDA)GIVING THIS INTERRUPT DID
3188                                  22490                                  ;NOT HAVE ITS POSITIONING FLAG SET.
3189  012304  005777  167412          22500
3190  012310  100044                  22510  5$:     TST     @RKCS           ;ANY ERROR?
3191                                  22520          BPL     8$              ;NO - RETURN
3192  012312  032737  000040  002174  22530
3193  012320  001012                  22540          BIT     #BIT5,ERCODE
3194  012322  042737  000040  002174  22550          BNE     6$              ;SAME ERROR
3195  012330  001026                  22560          BIC     #BIT5,ERCODE
3196  012332  052737  000040  002174  22570          BNE     7$
3197                                  22580          BIS     #BIT5,ERCODE
3198  012340  004737  021652          22590
3199                                  22600          JSR     PC,RG4SDR       ;GET RKCS, ER, DS, DA AND DRIVE # FOR
3200  012344  104011                  22610                                  ;TYPING SERIAL DRIVE #
3201                                  22620          ERROR   11              ;BIT 15, 'ERR' SET IN RKKCS AFTER FIRST
3202                                  22630                                  ;INTERRUPT - WHICH OCCURRED AFTER A
3203                                  22640                                  ;POSITIONING SEEK WAS INITIATED.  RKER
3204  012346  042710  000020          22650                                  ;WILL CONTAIN ERROR BIT/S
3205  012352  105061  002126          22660  6$:     BIC     #BIT4,(R0)      ;CLEAR 'POSITIONING' BIT
3206  012356  105061  002136          22670          CLRB    BUSY(R1)        ;CLEAR 'BUSY' FLAG
3207                                  22680          CLRB    POS(R1)         ;CLEAR 'POSITIONING' FLAG
3208  012362  004737  015454          22690
3209  012366  052710  010000          22700          JSR     PC,CLRERR       ;CLEAR THE ERROR
3210  012372  105261  002146          22710          BIS     #BIT12,(R0)     ;INDICATE HIGH PRIORITY ON
3211  012376  126127  002146  000003  22720          INCB    RETRY(R1)       ;INCREMENT RETRY COUNT
3212  012404  003406                  22730          CMPB    RETRY(R1),#3    ;DONE RETRIES?
3213                                  22740          BLE     8$              ;NO
3214  012406  004737  014122          22750
3215  012412  005061  002146          22760  7$:     JSR     PC,DRVABT       ;ABORT THE FUNCTION
3216  012416  005037  002174          22770          CLR     RETRY(R1)
3217                                  22780          CLR     ERCODE
3218  012422  000002                  22790
                                      22800  8$:     RTI
```

```
3219                                    22820                                   ;THIS ROUTINE IS ENTERED UPON COMPLETION
3220                                    22830                                   ;OF A SEEK OPERATION UNDER INTERRUPT MODE
3221                                    22840
3222  012424  004737  017764            22850   SKCMP:  JSR     PC,CHKCRDY      ;CONTROL READY SET?
3223  012430  104013                    22860           ERROR   13              ;CONTROL RDY NOT SET, AFTER A SEEK
3224                                    22870                                   ;DONE INTERRUPT!
3225                                    22880
3226  012432  017746  167260            22890   1$:     MOV     @RKDS,-(SP)     ;GET DRIVE # THAT INTERRUPTED
3227  012436  004737  021620            22900           JSR     PC,CROTLF       ;ROTATE BITS 15,14,13 TO POSITION
3228                                    22910                                   ;0,1,2
3229  012442  012601                    22920           MOV     (SP)+,R1        ;GET THE ROTATED BITS (DRIVE ID BITS)
3230  012444  105761  002126            22930           TSTB    BUSY(R1)        ;CHECK THAT DRIVE WAS BUSY
3231  012450  100403                    22940           BMI     2$
3232  012452  004737  021462            22950           JSR     PC,GT4RG
3233  012456  104014                    22960           ERROR   14              ;'BUSY' FLAG FOR THE INTERRUPTING DRIVE
3234                                    22970                                   ;(RKDS) WAS NOT SET.  DRIVE # (15,14,13 - RKDS)
3235                                    22980                                   ;INTERRUPTED AFTER SEEK WAS COMPLETE, BUT
3236                                    22990                                   ;'BUSY' FLAG CORRSPONDING TO THAT DRIVE
3237                                    23000                                   ;WAS CLEAR.  IF THE CORRECT DRIVE WAS
3238                                    23010                                   ;INTERRUPTING 'BUSY' FLAG SHOULD HAVE
3239                                    23020                                   ;BEEN SET.
3240  012460  116100  002126            23030   2$:     MOVB    BUSY(R1),R0     ;FORM THE ADDRESS OF THE
3241  012464  042700  177600            23040           BIC     #177600,R0      ;KEY
3242  012470  062700  002006            23050           ADD     #KEY,R0
3243  012474  032710  000020            23060           BIT     #BIT4,(R0)      ;CHECK THAT POSITION BIT WAS SET
3244  012500  001003                    23070           BNE     3$
3245  012502  010137  001162            23080           MOV     R1,$REGO        ;GET DRIVE NUMBER
3246  012506  104010                    23090           ERROR   10              ;POSITIONING FLAG (IN KEY) IS NOT SET
3247                                    23100                                   ;SET FOR INTERRUPTING DRIVE (GIVEN IN EROR
3248                                    23110                                   ;MESAGE)
3249  012510  105761  002136            23120   3$:     TSTB    POS(R1)         ;CHECK POSITIONING FLAG WAS SET
3250  012514  001003                    23130           BNE     4$
3251  012516  010137  001162            23140           MOV     R1,$REGO
3252  012522  104010                    23150           ERROR   10              ;'POSITIONING' FLAG WAS CLEAR.  WHEN
3253                                    23160                                   ;DOING A (POSITIONING) SEEK, THE FLAG
3254                                    23170                                   ;'POS#' IS SET TO INDICATE THAT DRIVE
3255                                    23180                                   ;IS POSITIONING HEADS (SEEKING).  THIS
3256                                    23190                                   ;ERROR INDICATES THAT FOR THE INTERRUPTING
3257                                    23200                                   ;DRIVE (RKDS) POSITIONING FLAG WAS
3258                                    23210                                   ;NOT SET.
3259  012524  004737  016204            23220   4$:     JSR     PC,CHKDRV       ;GO, CHECK FOR DRV ERORS (DPL,DRU,DRY,WPS)
3260  012530  000501                    23230           BR      PFTERR          ;IF THERE IS A DRV EROR, RETURN HERE
3261                                    23240                                   ;THE DRIVE HAS BEEN DESELECTED, ALL
3262                                    23250                                   ;FURTHER COMANDS ON THAT DRIVE ABORTED
3263                                    23260                                   ;RETURRN HERE IF NO DRIVE ERRORS
3264  012532  017777  167160  167170    23270           MOV     @RKDS,@RKDA     ;GET THE DRIVE # FROM DRIVE ID BITS
3265                                    23280                                   ;IN RKDS AND ADDRESS THE DRIVE
3266  012540  032777  000100  167150    23290           BIT     #RWS,@RKDS      ;R/W/S RDY SET?
3267  012546  001005                    23300           BNE     5$              ;YES
3268  012550  004737  021462            23310           JSR     PC,GT4RG        ;GET RKCS,ER,DS,DA
3269  012554  010137  001750            23320           MOV     R1,SRDRV        ;GET DRIVE #, FOR TYPING SERIAL #
3270  012560  104015                    23330           ERROR   15              ;R/W/S RDY NOT SET FOR INTERRUPTING
3271                                    23340                                   ;DRIVE (RKDS), AFTER SEEK WAS COMPLETED.
3272  012562  032777  001000  167126    23350   5$:     BIT     #SIN,@RKDS      ;'SIN' ERROR?
3273  012570  001007                    23360           BNE     PSINER          ;YES, BRANCH
3274                                    23370
```

```
  3275  012572  032777  040000  167122  23380          BIT     #HE,@RKCS       ;ANY HARD ERROR?
  3276  012600  001060                  23390          BNE     PORKER          ;YES
  3277                                  23400                                  ;NO ERRORS DETECTED ON POSITIONING
  3278  012602  105061  002126          23410          CLRB    BUSY(R1)        ;CLEAR BUSY FLAG FOR THIS DRIVE
  3279  012606  000207                  23420          RTS     PC
```

```
3280
3281
3282  012610  004737  021462
3283  012614  010137  001750
3284  012620  104016
3285  012622  042710  000020
3286  012626  105061  002136
3287  012632  105061  002126
3288  012636  004737  015620
3289
3290  012642  004737  015732
3291
3292  012646  000432
3293
3294
3295  012650  105261  002146
3296  012654  105061  002136
3297  012660  105061  002126
3298  012664  122761  000003  002146
3299  012672  001403
3300  012674  052710  010000
3301  012700  000207
3302
3303  012702  052710  104000
3304  012706  105061  002146
3305  012712  010102
3306  012714  006302
3307  012716  005262  002372
3308  012722  042710  010000
3309  012726  104420  002667
3310  012732  000207
3311
3312  012734  042710  010000
3313  012740  000207
3314
3315  012742  004737  021462
3316  012746  010137  001750
3317  012752  104017
3318
3319
3320  012754  004737  015454
3321  012760  042710  000020
3322  012764  105061  002136
3323  012770  105061  002126
3324  012774  000725

23440                                   ;THERE WAS A 'SIN' ERROR ON
23450                                   ;DOING POSITIONING (SEEK)
23460  PSINER:  JSR     PC,GT4RG
23470           MOV     R1,SRDRV        ;GET DRIVE #, FOR TYPING SERIAL #
23480           ERROR   16              ;'SIN' ON DOING (POSITIONING) SEEK
23490           BIC     #BIT4,(R0)      ;CLEAR 'POSITIONING' BIT
23500           CLRB    POS(R1)         ;CLEAR POSITTIONING FLAG
23510           CLRB    BUSY(R1)        ;CLEAR BUSY FLAG FOR THIS DRIVE
23520           JSR     PC,CLRSIN       ;CLEAR 'SIN' ERROR
23530
23540           JSR     PC,SINCNT       ;KEEP COUNT OF 'SIN' ERRORS. IF MORE
23550                                   ;THAN 'MAX' DESELECT THE DRIVE
23560           BR      PFTERR          ;RETUN HERE IF COUNT=MAX
23570
23580                                   ;RETURN HERE IF COUNT LESS THAN MAX
23590  PSRTRY:  INCB    RETRY(R1)       ;INCRMNT REETRY COUNT
23600           CLRB    POS(R1)         ;CLEAR POSITION FLAG
23610           CLRB    BUSY(R1)        ;CLEAR BUSY FLAG
23620           CMPB    #3,RETRY(R1)    ;DONE ALL RETRIES?
23630           BEQ     1$              ;YES
23640           BIS     #BIT12,(R0)     ;SET HI PRIORITY ON RETRY
23650           RTS     PC              ;RETURN
23660
23670  1$:      BIS     #104000,(R0)    ;INDICATE COMMAND ABORTED
23680           CLRB    RETRY(R1)       ;CLEAR RETRY COUNT FOR FUTURE USE
23690           MOV     R1,R2
23700           ASL     R2
23710           INC     ABORT(R2)       ;INCREMENT ABORT COUNT
23720           BIC     #BIT12,(R0)     ;CLR HI PRIORITY  BIT
23730           TYPMSG  ,MSG10          ;PRINT ABORT MESSAGE
23740           RTS     PC
23750
23760  PFTERR:  BIC     #BIT12,(R0)     ;CLEAR HI PRIORITY BIT IF SET
23770           RTS     PC
23780
23790  PORKER:  JSR     PC,GTIRG        ;IF ANY ERROR BIT IN RKCS SET
23800           MOV     R1,SRDRV        ;GET DRIVE #, FOR TYPING SERIAL #
23810           ERROR   17              ;ON DOING POSITIONING (SEEK)
23820                                   ;ENTER HERE
23830
23840           JSR     PC,CLRERR       ;GO CLEAR THE HARD ERROR
23850           BIC     #BIT4,(R0)      ;CLEAR POSITIONING BIT IN KEY
23860           CLRB    POS(R1)         ;CLEAR POSITIONING FLAG FOR THIS DRIVE
23870           CLRB    BUSY(R1)        ;CLEAR BUSY FLAG FOR THIS DRIVE
23880           BR      PSRTRY
```

```
 3325
 3326
 3327
 3328  012776  105037  002234                23900                          ;ENTER THIS ROUTINE WHEN AN INTERRUPT
 3329                                         23910                          ;OCCURS. NOTE THERE IS ONE OTHER
 3330  013002  013766  001744  000002         23920                          ;INTERRUPT ROUTINE- 'INTISK'
 3331  013010  004737  017764                23930  INTHND: CLRB    INTFLG  ;CLEAR FLAG TO INDICATE THAT
 3332  013014  104024                         23940                          ;AN INTERRUPT OCCURED.
 3333                                         23950          MOV     PPRLVL,2(SP) ;CPU PRIORITY TO 5 ON RTI
 3334  013016  032777  020000  166676         23960          JSR     PC,CHKCRDY ;CONTROL READY SET?
 3335  013024  001403                         23970          ERROR   24      ;CONTROL RDY CLEAR AFTER INTRUPT ON COMPLETION
 3336  013026  004737  012424                23980                          ;OF FUNCTION, IT SHOULD BE SET
 3337  013032  000002                         23990  1$:     BIT     #SCP,@RKCS ;SCP SET? SEARCH COMPLETE?
 3338  013034  017746  166670                24000          BEQ     2$
 3339                                         24010          JSR     PC,SKCMP ;GO TO SEEK COMPLETE ROUTINE
 3340                                         24020          RTI
 3341  013040  004737  021620                24030  2$:     MOV     @RKDA,-(SP) ;GET THE DRIVE # TO WHICH
 3342                                         24040                          ;THE COMMAND WAS ISSUED UNDER
 3343  013044  012605                         24050                          ;INTERRUPT MODE
 3344  013046  105765  002126                24060          JSR     PC,CROTLF ;ROTATE BITS 15,14,13 TO POSITION
 3345  013052  100403                         24070                          ;0,1,2
 3346  013054  010537  001162                24080          MOV     (SP)+,R5 ;POP OFF THE DRIVE # FROM THE STACK
 3347  013060  104007                         24090          TSTB    BUSY(R5) ;CHECK THAT DRIVE WAS BUSY
 3348                                         24100          BMI     3$
 3349                                         24110          MOV     R5,$REGC
 3350                                         24120          ERROR   7       ;'BUSY' FLAG FOR THE INTERUPTING DRIVE
 3351                                         24130                          ;WAS NOT SET.  WHENEVER ANY ACTIVITY
 3352                                         24140                          ;IS INITIATED ON A DRIVE, THE (SOFTWARE)
 3353                                         24150                          ;'BUSY' FLAG IS SET TO INDICATE THAT
 3354                                         24160                          ;DRIVE IS BUSY DOING SOMETHING.
 3355  013062  105065  002136                24170                          ;OCCURANCE OF THIS ERROR INDICATES
 3356  013066  116500  002126                24180                          ;THAT THE 'BUSY' FLAG FOR THE INTERRUPTING
 3357  013072  042700  177760                24190                          ;DRIVE (RKDA) IS CLEAR!
 3358  013076  062700  002006                24200  3$:     CLRB    POS(R5) ;CLEAR THE POSITION FLAG
 3359  013102  032710  000010                24210          MOVB    BUSY(R5),R0 ;GET THE ADDRESS OF THE
 3360  013106  001401                         24220          BIC     #177760,R0 ;COMMAND KEY
 3361  013110  104020                         24230          ADD     #KEY,R0
 3362                                         24240          BIT     #BIT3,(R0) ;CHECK IT'S NOT A SEEK FUNCTION
 3363                                         24250          BEQ     4$
 3364                                         24260          ERROR   20      ;(SOFTWARE) FLAG FOR THE INTERRUPTING
 3365                                         24270                          ;DRIVE (RKDA) INDICATES THAT A SEEK
 3366                                         24280                          ;FUNCTION WAS BEING EXECUTED ON THE
 3367                                         24290                          ;DRIVE.  IF THAT'S THE CASE, SCP BIT SHOULD
 3368                                         24300                          ;HAVE SET ON SEEK DONE INTRUPT, BUT IT
 3369  013112  105710                         24310                          ;WAS NOT.  NOTE, HERE THERE IS A CHANGE
 3370  013114  100403                         24320                          ;OF MULTIPLE ERRORS OR ERROR
 3371  013116  010537  001162                24330                          ;MISINTERPRETATION
 3372  013122  104031                         24340  4$:     TSTB    (R0)    ;CHECK THAT FUNCTION IN
 3373                                         24350          BMI     5$      ;PROGRESS BIT WAS SET
 3374                                         24360          MOV     R5,$REGD ;GET DRIVE NUMBER
 3375                                         24370          ERROR   31      ;IF NOT, ERROR
 3376                                         24380                          ;'FUNCTION IN PROGRESS' FLAG (IN THE KEY)
 3377  013124  011002                         24390                          ;WAS NOT SET FOR THE INTERRUPTING DRIVE.
 3378  013126  042702  177770                24400                          ;IF THIS DRIVE WAS BUSY DOING THE
 3379  013132  020502                         24410                          ;FUNCTION, THE ABOVE FLAG SHOULD BE SET.
 3380  013134  001405                         24420  5$:     MOV     (R0),R2 ;CHECK THAT THE INTERRUPTING
                                              24430          BIC     #177770,R2 ;DRIVE (IN RKDA) IS THE SAME AS
                                              24440          CMP     R5,R2   ;THE DRIVE IN THE KEY
                                              24450          BEQ     6$
```

```
3381  013136  010537  001162          24460          MOV     R5,$REGO        ;GET EXPCTD DRIVE NO.
3382  013142  010237  001164          24470          MOV     R2,$REG1        ;GET DRIVE NO. RECVD
3383  013146  104032                  24480          ERROR   32              ;UNEXPECTED DRIVE NO. INTERRUPTED
3384  013150  006302                  24490  6$:     ASL     R2
3385  013152  011003                  24500          MOV     (R0),R3
3386  013154  010037  002236          24510          MOV     R0,SAVKEY
3387  013160  000303                  24520          SWAB    R3
3388  013162  042703  177770          24530          BIC     #177770,R3      ;GET POSITION OF THE PARAMETER
3389  013166  006303                  24540          ASL     R3              ;LIST FROM THE KEY
3390  013170  017704  166526          24550          MOV     @RKCS,R4
3391  013174  042704  177761          24560          BIC     #177761,R4      ;CLEAR ALL BUT FUNCTION CODE
3392  013200  016305  002532          24570          MOV     PCMND(R3),R5    ;CHECK THAT THE FUNCTION IN
3393                                  24580                                  ;RKCS AND THE KEY ARE SAME.
3394  013204  126504  000002          24590          CMPB    2(R5),R4
3395                                  24600
3396  013210  001406                  24610          BEQ     7$              ;IF NOT, ERROR
3397  013212  016537  000002  001162  24620          MOV     2(R5),$REGO     ;GET EXPCTD FUNCTION CODE IN RKCS
3398  013220  010437  001164          24630          MOV     R4,$REG1        ;GET CODE RECVD
3399  013224  104033                  24640          ERROR   33              ;FUNCTION CODE THAT IS IN RKCS AFTER THIS
3400                                  24650                                  ;INTERRUPT OCCURED IS NOT THE EXPECTED ONE
3401  013226  042710  000200          24660  7$:     BIC     #BIT7,(R0)      ;CLEAR 'FUNCTION IN PROGRESS' BIT
3402  013232  142761  000200  002126  24670          BICB    #BIT7,BUSY(R1)  ;CLEAR BUSY FLAG FOR THIS DRIVE
3403  013240  004737  016204          24680          JSR     PC,CHKDRV       ;CHECK FOR DRIVE ERRORS
3404  013244  000002                  24690          RTI                     ;IF THERE WAS ANY DRIVE EROR
3405                                  24700                                  ;DESELECT THE DRIVE AND EXIT
3406                                  24710                                  ;IF NO ERROR, RETURN HERE
3407  013246  032777  001000  166442  24720          BIT     #SIN,@RKDS      ;SIN ERROR?
3408  013254  001426                  24730          BEQ     10$
3409  013256  004737  021652          24740          JSR     PC,RG4SDR       ;GET RKCS, ER, DS, DA AND DRIVE # FOR
3410                                  24750                                  ;TYPING SERIAL DRIVE #
3411  013262  104016                  24760          ERROR   16              ;SIN ERROR
3412  013264  004737  015620          24770          JSR     PC,CLRSIN
3413  013270  004737  015732          24780          JSR     PC,SINCNT       ;HELP COUNT OF SIN'S. IF MORE
3414                                  24790                                  ;THAN MAXM ALLOWABLE, DESELECT THE DRIVE
3415  013274  000002                  24800          RTI                     ;RETURN HERE IF COUNT=MAX
3416                                  24810                                  ;RETURN HERE IF LESS THAN MAX
3417  013276  105261  002146          24820          INCB    RETRY(R1)
3418  013302  122761  000003  002146  24830          CMPB    #3,RETRY(R1)
3419  013310  001405                  24840          BEQ     8$
3420  013312  052710  010000          24850          BIS     #BIT12,(R0)
3421  013316  042710  000020          24860          BIC     #BIT4,(R0)
3422  013322  000002                  24870          RTI
3423                                  24880
3424  013324  004737  014122          24890  8$:     JSR     PC,DRVABT       ;ABORT THIS FUNCTION
3425  013330  000002                  24900  9$:     RTI
3426                                  24910
3427  013332  032777  040000  166362  24920  10$:    BIT     #HE,@RKCS       ;HARD ERROR?
3428  013340  001076                  24930          BNE     HRDERR
3429  013342  032777  100000  166352  24940          BIT     #ERR,@RKCS      ;SOFT ERROR?
3430  013350  001002                  24950          BNE     SFTERR          ;YES
3431  013352  000137  013774          24960          JMP     NOEROR          ;NO
```

```
3432                                                24980                                   ;THERE WAS A SOFT ERROR
3433                                                24990
3434                                                25000
3435  013356  032777  000001  166334              25010  SFTERR:  BIT     #WCE,@RKER      ;WCE?
3436  013364  001417                              25020           BEQ     1$
3437                                                25030
3438  013366  032737  000001  002174              25040           BIT     #BIT0,ERCODE    ;ALREADY WORKING ON A WC ERROR?
3439  013374  001054                              25050           BNE     3$              ;YES - IGNORE THIS ONE
3440  013376  042737  000001  002174              25060           BIC     #BIT0,ERCODE    ;ANY OTHER ERROR?
3441  013404  001052                              25070           BNE     4$              ;YES - ABORT THIS FUNCTION
3442  013406  052737  000001  002174              25080           BIS     #BIT0,ERCODE    ;SET THE WC ERROR BIT
3443                                                25090
3444  013414  005262  002332                      25100           INC     WCECN(R2)       ;INCREMENT WCE COUNT FOR
3445  013420  104420  002572                      25110           TYPMSG  ,MSG2           ;THIS DRIVE
3446                                                25120
3447  013424  032777  000002  166266              25130  1$:      BIT     #CSE,@RKER      ;CSE?
3448  013432  001417                              25140           BEQ     2$
3449                                                25150
3450  013434  032737  000002  002174              25160           BIT     #BIT1,ERCODE    ;ALREADY WORKING ON A CS ERROR?
3451  013442  001031                              25170           BNE     3$              ;YES - IGNORE THIS ONE
3452  013444  042737  000002  002174              25180           BIC     #BIT1,ERCODE    ;ANY OTHER ERROR?
3453  013452  001027                              25190           BNE     4$              ;YES - ABORT THIS FUNCTION
3454  013454  052737  000002  002174              25200           BIS     #BIT1,ERCODE    ;SET THE CS ERROR BIT
3455                                                25210
3456  013462  005262  002352                      25220           INC     CSECN(R2)       ;INCREMENT CSE COUNT FOR THIS DRIVE
3457  013466  104420  002600                      25230           TYPMSG  ,MSG3
3458  013472  104420  002622                      25240  2$:      TYPMSG  ,MSG5
3459  013476  004737  021366                      25250           JSR     PC,TYPFN        ;GO TYPE OUT THE FUNCTION THAT GAVE ERROR
3460  013502  004737  021514                      25260           JSR     PC,GETINF
3461  013506  004737  021656                      25270           JSR     PC,GTSDRV       ;SAVE DRIVE #, FOR TYPING SERIAL #
3462  013512  104021                              25280           ERROR   21              ;SOFT ERROR ON PERFORMING A DATA
3463                                                25290                                   ;TRANSFER RKDA PRINTED OUT IN ERROR
3464                                                25300                                   ;MESSAGE IS BROKEN DOWN INTO DRVE #,
3465                                                25310                                   ;CYL # SEC #, SUR #
3466  013514  022704  000004                      25320           CMP     #4,R4           ;WAS IT A READ FUNCTION?
3467  013520  001002                              25330           BNE     3$
3468  013522  004737  016554                      25340           JSR     PC,DATCHK       ;GO CHECK THE DATA READ
3469                                                25350
3470  013526  000137  013700                      25360  3$:      JMP     CMNERR
3471  013532  000137  013736                      25370  4$:      JMP     CMNABT
```

```
3472                                                25390                                   ;IF THERE WAS A HARD
3473                                                25400                                   ;ERROR DURING THE DATA
3474                                                25410                                   ;TRANSFER ENTER HERE
3475                                                25420
3476  013536  032777  010000  166154                25430  HRDERR:  BIT     #SKE,@RKER      ;SKE?
3477  013544  001421                                25440           BEQ     1$
3478                                                25450
3479  013546  032737  000004  002174                25460           BIT     #BIT2,ERCODE    ;ALREADY WORKING ON A SK ERROR?
3480  013554  001051                                25470           BNE     CMNERR          ;YES - IGNORE THIS ONE
3481  013556  042737  000004  002174                25480           BIC     #BIT2,ERCODE    ;ANY OTHER ERROR?
3482  013564  001064                                25490           BNE     CMNABT          ;YES - ABORT THIS FUNCTION
3483  013566  052737  000004  002174                25500           BIS     #BIT2,ERCODE    ;SET THE SK ERROR BIT
3484                                                25510
3485  013574  005262  002302                        25520           INC     SKECN(R2)       ;INCREMENT COUNT FOR SKE
3486  013600  005262  002262                        25530           INC     HECN(R2)        ;ON THIS DRIVE
3487                                                25540
3488  013604  104420  002564                        25550           TYPMSG  ,MSG1
3489                                                25560
3490  013610  032777  167740  166102                25570  1$:      BIT     #167740,@RKER   ;ANY HE BESIDES SKE?
3491  013616  001417                                25580           BEQ     2$
3492                                                25590
3493  013620  032737  000010  002174                25600           BIT     #BIT3,ERCODE    ;ALREADY WORKING ON A HE ERROR?
3494  013626  001024                                25610           BNE     CMNERR          ;YES - IGNORE THIS ONE
3495  013630  042737  000010  002174                25620           BIC     #BIT3,ERCODE    ;ANY OTHER ERROR?
3496  013636  001037                                25630           BNE     CMNABT          ;YES - ABORT THIS FUNCTION
3497  013640  052737  000010  002174                25640           BIS     #BIT3,ERCODE    ;SET THE HE ERROR BIT
3498                                                25650
3499  013646  005262  002262                        25660           INC     HECN(R2)        ;INCREMENT HE COUNT FOR
3500  013652  104420  002606                        25670           TYPMSG  ,MSG4           ;THIS DRIVE
3501                                                25680
3502  013656  104420  002622                        25690  2$:      TYPMSG  ,MSG5           ;PRINT 'ON DOING'
3503  013662  004737  021366                        25700           JSR     PC,TYPFN        ;GO PRINT OUT THE FUNCTION
3504                                                25710                                   ;WHICH GAVE THIS ERROR
3505                                                25720
3506  013666  004737  021514                        25730           JSR     PC,GETINF
3507  013672  004737  021656                        25740           JSR     PC,GTSDRV       ;SAVE DRIVE #, FOR TYPING SERIAL #
3508  013676  104022                                25750           ERROR   22              ;HARD ERROR ON DOING DATA XFER
3509                                                25760
3510  013700  105261  002146                        25770  CMNERR:  INCB    RETRY(R1)       ;INCREMENT RETRY COUNT
3511  013704  122761  000003  002146                25780           CMPB    #3,RETRY(R1)    ;3 TRIES IN ALL?
3512  013712  001411                                25790           BEQ     CMNABT          ;YES, ABORT THIS FUNCTION
3513                                                25800
3514  013714  052777  010000  166314                25810           BIS     #BIT12,@SAVKEY  ;INDICATE HIGH PRIORITY ON RETRY
3515  013722  042777  000020  166306                25820           BIC     #BIT4,@SAVKEY   ;CLEAR 'POSITIONING HEADS', IF SET
3516  013730  004737  015454                        25830           JSR     PC,CLRERR       ;CLEAR THIS ERROR
3517  013734  000002                                25840           RTI
3518                                                25850
3519  013736  005037  002174                        25860  CMNABT:  CLR     ERCODE          ;CLEAR ERROR CODE
3520  013742  104420  003334                        25870           TYPMSG  ,MSG27
3521  013746  004737  014122                        25880           JSR     PC,DRVABT       ;ABORT THE FUNCTION
3522  013752  032777  010000  165740                25890           BIT     #SKE,@RKER      ;SEEK ERROR?
3523  013760  001404                                25900           BEQ     3$              ;NO
3524  013762  010137  002202                        25910           MOV     R1,@DRV         ;SET DRIVE NUMBER
3525  013766  104415                                25920           CON.RESET               ;RESET THE CONTROLLER
3526  013770  104417                                25930           DRV.RESET               ;RESET THE DRIVE
3527                                                25940
```

```
  3528  013772  000002                25950   3$:     RTI                     ;THIS DATA TRANSFER
```

```
3529                                                    25970                                   ;IF THERE WAS NO HARD OR
3530                                                    25980                                   ;SOFT ERROR ON DATA TRANSFER
3531                                                    25990                                   ;ENTER HERE
3532  013774  105761  002146                          26000   NOEROR: TSTB    RETRY(R1)
3533  014000  001406                                  26010           BEQ     1$
3534  014002  104420  003406                          26020           TYPMSG  ,MSG28
3535  014006  116146  002146                          26030           MOVB    RETRY(R1),-(SP)
3536  014012  104402                                  26040           TYPOS
3537  014014     002                                  26050           .BYTE   2
3538  014015     000                                  26060           .BYTE   0
3539                                                    26070
3540  014016  005037  002174                          26080   1$:     CLR     ERCODE
3541  014022  105061  002146                          26090           CLRB    RETRY(R1)
3542  014026  042777  010000  166202                  26100           BIC     #BIT12,@SAVKEY  ;CLEAR PRIORITY BIT IF SET
3543  014034  004737  020500                          26110           JSR     PC,STATSTC      ;GO, COLLECT STATISTICS ON THIS
3544                                                    26120                                   ;DATA TRANSFER
3545                                                    26130
3546  014040  022704  000004                          26140           CMP     #4,R4           ;WAS IT A 'READ' FUNCTION?
3547  014044  001002                                  26150           BNE     2$
3548  014046  004737  016554                          26160           JSR     PC,DATCHK       ;IF IT WAS 'READ', CHECK THE DATA
3549                                                    26170
3550                                                    26180
3551                                                    26190
3552  014052  022704  000002                          26200   2$:     CMP     #2,R4           ;WAS IT A 'WRITE' FUNCTION?
3553  014056  001011                                  26210           BNE     3$
3554  014060  032777  000040  166150                  26220           BIT     #BIT5,@SAVKEY   ;IS 'WRT CHK' TO FOLLOW THIS
3555  014066  001412                                  26230           BEQ     4$              ;'WRT' FUNCTION?
3556  014070  052703  100000                          26240           BIS     #BIT15,R3       ;SET FLAG BIT TO INDICATE
3557                                                    26250                                   ;THAT WRITE CHECK SHOULD
3558                                                    26260                                   ;FOLLOW THIS WRITE
3559  014074  010337  002156                          26270           MOV     R3,WCFLG        ;SET UP WC FLAG TO INDICATE
3560                                                    26280                                   ;THE ABOVE. THE LOWER BYTE
3561                                                    26290                                   ;CONTAINS THE POINTER TO THE
3562                                                    26300                                   ;COMMAND LIST, WHICH WILL
3563                                                    26310                                   ;BE USED, FOR DOING WRITE CHECK
3564  014100  000407                                  26320           BR      5$              ;IF WRITE CHECK IS TO BE DONE, DONT
3565                                                    26330                                   ;SET BIT 15 OF THE KEY TILL WRT CHK
3566                                                    26340                                   ;IS DONE
3567                                                    26350
3568  014102  022704  000006                          26360   3$:     CMP     #6,R4           ;WRT CHK FUNCTION?
3569  014106  001002                                  26370           BNE     4$
3570  014110  005037  002156                          26380           CLR     WCFLG           ;CLEAR WR CHK FLAG
3571                                                    26390
3572  014114  052710  100000                          26400   4$:     BIS     #BIT15,(R0)     ;SET FLAG TO INDICATE THIS
3573  014120  000002                                  26410   5$:     RTI                     ;FUNCTION IS COMPLETED
```

```
 3574                                          26430                                  ;ABORT THE FUNCTION ON DRIVE POINTED TO BY R1
 3575                                          26440                                  ;CLEAR ASSOCIATED FLAGS
 3576                                          26450                                  ;DROP THE DRIVE IF MORE THAN 8. ABORTS
 3577                                          26460
 3578  014122  010246                         26470  DRVABT:  MOV     R2,-(SP)       ;SAVE R2
 3579  014124  105061  002146                 26480           CLRB    RETRY(R1)
 3580  014130  052710  104000                 26490           BIS     #104000,(R0)   ;INDICATE THAT FUNCTION IS ABORTED
 3581  014134  042710  010000                 26500           BIC     #BIT12,(R0)    ;CLEAR HIGH PRIORITY BIT IF SET
 3582  014140  010102                         26510           MOV     R1,R2
 3583  014142  006302                         26520           ASL     R2
 3584  014144  005262  002372                 26530           INC     ABORT(R2)
 3585  014150  026227  002372  000010         26540           CMP     ABORT(R2),#10
 3586  014156  003402                         26550           BLE     1$
 3587  014160  000137  015760                 26560           JMP     DSELCT         ;DROP THE DRIVE
 3588  014164  012602                         26570  1$:      MOV     (SP)+,R2       ;RESTORE R2
 3589  014166  104400  002667                 26580           TYPE    ,MSG10         ;TYPE "ABORTED"
 3590  014172  000207                         26590           RTS     PC             ;RETURN
```

```
 3591                                                  26610   ;THIS ROUTINE GENERATES THE 8 COMMAND REQUESTS AND PUT THEM IN THE QUEUE. THE
 3592                                                  26620   ;FOLLOWING PARAMETERS ARE GENERATED RANDOMLY:
 3593                                                  26630   ;1.  DRIVE NUMBER ON WHICH THE COMMAND WILL BE PERFORMED.
 3594                                                  26640   ;2.  FUNCTION TO BE PERFORMED.
 3595                                                  26650   ;3.  DISK ADDRESS - CYLINDER, SURFACE, SECTOR.
 3596                                                  26660   ;4.  STARTING BUS ADDRESS.
 3597                                                  26670   ;5.  WORD COUNT FOR DATA TRANSFER.
 3598                                                  26680
 3599                                                  26690   ;THE QUEUE IS LOCATED AT 'CMND' (TO 'CMND8').
 3600                                                  26700
 3601  014174  104406                                 26710   GEN8RQ: CKSWR
 3602  014176  005337  002556                         26720           DEC     REPCNT          ;DECREMENT REPETITION COUNT
 3603  014202  001026                                 26730           BNE     1$              ;CONTINUE IF NOT 0
 3604  014204  013737  001736  002556                 26740           MOV     PCNTR,REPCNT    ;REESTABLISH REPETITION COUNT FOR EXERCISER
 3605  014212  104400  014220                         26750           TYPE    ,65$            ;;TYPE ASCIZ STRING
 3606  014216  000410                                 26760           BR      64$             ;;GET OVER THE ASCIZ
 3607                                                  26770   ;;65$:  .ASCIZ  <15><12><12><12>/END OF PASS/
 3608  014240                                         26780   64$:
 3609  014240  013746  001100                         26790           MOV     $PASS,-(SP)
 3610  014244  005216                                 26800           INC     (SP)
 3611  014246  104404                                 26810           TYPDS
 3612  014250  004737  020610                         26820           JSR     PC,REPSTAT
 3613  014254  000137  022332                         26830           JMP     $EOP
 3614                                                  26840
 3615  014260  032777  000400  164652                 26850   1$:     BIT     #SW8,@SWR
 3616  014266  001406                                 26860           BEQ     2$
 3617  014270  104400  001213                         26870           TYPE    ,$CRLF
 3618  014274  104400  001213                         26880           TYPE    ,$CRLF
 3619  014300  004737  020610                         26890           JSR     PC,REPSTAT
 3620  014304  032777  000040  164626                 26900   2$:     BIT     #SW5,@SWR       ;HALT?
 3621  014312  001401                                 26910           BEQ     3$              ;NO
 3622  014314  000000                                 26920           HALT                    ;YES, PRESSING CONTINUE RESUMES PROG EXECUTION
 3623                                                  26930
 3624  014316  004737  022304                         26940   3$:     JSR     PC,CHDPRS       ;CHECK IF ANY DRIVES PRESENT
 3625  014322  012700  002006                         26950   4$:     MOV     #KEY,R0
 3626  014326  010004                                 26960           MOV     R0,R4
 3627  014330  005001                                 26970           CLR     R1
 3628  014332  010120                                 26980   5$:     MOV     R1,(R0)+        ;CLEAR THE 8 COMMAND KEYS
 3629  014334  062701  000400                         26990           ADD     #400,R1         ;BITS 8,9,10 INDICATE THEIR
 3630  014340  022701  004000                         27000           CMP     #4000,R1        ;POSITION 0,1,2,3,4,5,6,7
 3631  014344  001372                                 27010           BNE     5$
 3632  014346  012700  002026                         27020           MOV     #CMND,R0        ;CLEAR THE PARAMETER TABLE
 3633  014352  010005                                 27030           MOV     R0,R5
 3634  014354  012701  177740                         27040           MOV     #-40,R1         ;FOR THE 8 COMMANDS IN Q
 3635  014360  005020                                 27050   6$:     CLR     (R0)+           ;(8X4) WORDS IN ALL
 3636  014362  005201                                 27060           INC     R1
 3637  014364  001375                                 27070           BNE     6$
 3638                                                  27080
 3639  014366  004737  015436                         27090           JSR     PC,CLRFLGS      ;CLEAR THE FLAGS PERTAINING TO THE 8
 3640                                                  27100                                   ;COMMANDS IN THE Q
 3641                                                  27110                                   ;R4 CONTAINS 'KEY'
 3642                                                  27120                                   ;R5 CONTAINS 'CMND'
 3643  014372  022737  000001  001764                 27130   GEN1:   CMP     #1,DRVPRS       ;ONLY 1 DRV PRESENT?
 3644  014400  001002                                 27140           BNE     22$             ;NO
 3645  014402  005000                                 27150           CLR     R0              ;YES
 3646  014404  000420                                 27160           BR      3$
```

```
 3647  014406  004737  025224          27140  22$:   JSR     PC,$RAND        ;GENERATE A RANDOM NUMBER
 3648                                  27150                                  ;GET A RANDOM DRIVE NUMBER
 3649                                  27160                                  ;FROM THE AVAILABLE DRIVES
 3650  014412  025356                  27170          RSDRVL
 3651
 3652  014414  013746  025360                         MOV     RSDRVH,-(SP)    ;PUT LOW DIVIDEND ON STACK
 3653  014420  005046                                 CLR     -(SP)           ;CLEAR HIGH DIVIDEND AND PUSH
 3654                                                                         ;IT ON STACK
 3655  014422  013746  002220                         MOV     DRMAP,-(SP)     ;PUSH DIVISOR ON STACK
 3656  014426  004737  024640                         JSR     PC,@#$DIV       ;GO TO THE 'DIVIDE' SUBROUTINE
 3657  014432  005726                                 TST     (SP)+           ;DISCARD THE REMAINDER. QUOTIENT IS
 3658                                                                         ;NOW ON TOP OF THE STACK
 3659  014434  012600                  27190          MOV     (SP)+,R0
 3660  014436  020037  001764          27200          CMP     R0,DRVPRS       ;MAKE SURE CORRECT MAPPING IS DONE
 3661  014442  001001                  27210          BNE     3$
 3662  014444  005300                  27220          DEC     R0              ;'QDRVE' CONTAINS RANDOM DRIVE NO.
 3663  014446  005037  002202          27230  3$:     CLR     QDRV
 3664  014452  116037  001754  002202  27240          MOVB    PDP(R0),QDRV
 3665  014460  105737  002202          27250          TSTB    QDRV            ;TEST IF TYPE F DRIVE
 3666  014464  100016                  27260          BPL     32$             ;NOT IF POSITIVE
 3667                                  27270
 3668  014466  142737  000200  002202  27280          BICB    #200,QDRV       ;CLEAR THE FLAG BIT
 3669  014474  132737  000001  002202  27290          BITB    #1,QDRV         ;ODD OR EVEN DRIVE ADDRESS
 3670  014502  001404                  27300          BEQ     31$
 3671                                  27310
 3672  014504  005737  015372          27320          TST     ODDEVN          ;MUST HAVE BEEN AN ODD ONE
 3673  014510  001730                  27330          BEQ     GEN1            ;INSURE THAT ODDS WHAT WE WANT
 3674  014512  000403                  27340          BR      32$
 3675                                  27350
 3676  014514  005737  015372          27360  31$:    TST     ODDEVN          ;MUST HAVE BEEN AN EVEN ONE
 3677  014520  001332                  27370          BNE     22$             ;NO GOOD - TRY AGAIN
 3678                                  27380
 3679  014522  004737  025224          27390  32$:    JSR     PC,$RAND        ;GENERATE A RANDOM NUMBER
 3680  014526  025362                  27400          RSFUNL
 3681
 3682  014530  013746  025364                         MOV     RSFUNH,-(SP)    ;PUT LOW DIVIDEND ON STACK
 3683  014534  005046                                 CLR     -(SP)           ;CLEAR HIGH DIVIDEND AND PUSH
 3684                                                                         ;IT ON STACK
 3685  014536  013746  002226                         MOV     FNMAP,-(SP)     ;PUSH DIVISOR ON STACK
 3686  014542  004737  024640                         JSR     PC,@#$DIV       ;GO TO THE 'DIVIDE' SUBROUTINE
 3687  014546  005726                                 TST     (SP)+           ;DISCARD THE REMAINDER. QUOTIENT IS
 3688                                                                         ;NOW ON TOP OF THE STACK
 3689  014550  021627  000003          27420          CMP     (SP),#3         ;MAKE SURE CORRECT FUNCTION IS SELCTD
 3690  014554  001001                  27430          BNE     20$
 3691  014556  005316                  27440          DEC     (SP)
 3692  014560  012637  002212          27450  20$:    MOV     (SP)+,QFNC      ;THE FUNCTION THAT CAN BE PERFORMED
 3693                                  27460
 3694  014564  004737  025224          27470          JSR     PC,$RAND        ;GENERATE A RANDOM NUMBER
 3695  014570  025366                  27480          RSCYLL
 3696
 3697  014572  013746  025370                         MOV     RSCYLH,-(SP)    ;PUT LOW DIVIDEND ON STACK
 3698  014576  005046                                 CLR     -(SP)           ;CLEAR HIGH DIVIDEND AND PUSH
 3699                                                                         ;IT ON STACK
 3700  014600  013746  002222                         MOV     CYLMAP,-(SP)    ;PUSH DIVISOR ON STACK
 3701  014604  004737  024640                         JSR     PC,@#$DIV       ;GO TO THE 'DIVIDE' SUBROUTINE
 3702  014610  005726                                 TST     (SP)+           ;DISCARD THE REMAINDER. QUOTIENT IS
```

```
3703                                                                    ;NOW ON TOP OF THE STACK
3704  014612  012637  002204          27500          MOV    (SP)+,QCYL  ;(FROM 0-312)
3705  014616  022737  000313  002204  27510          CMP    #313,QCYL
3706  014624  001002                  27520          BNE    4$
3707  014626  005337  002204          27530          DEC    QCYL        ;'QCYL' CONTAINS RANDOM CYLINDER NO.
3708                                  27540
3709  014632                          27550    4$:
3710
3711  014632  013746  025370                         MOV    RSCYLH,-(SP)  ;PUT LOW DIVIDEND ON STACK
3712  014636  005046                                 CLR    -(SP)         ;CLEAR HIGH DIVIDEND AND PUSH
3713                                                                       ;IT ON STACK
3714  014640  013746  002224                         MOV    SECMAP,-(SP)  ;PUSH DIVISOR ON STACK
3715  014644  004737  024640                         JSR    PC,@#$DIV     ;GO TO THE 'DIVIDE' SUBROUTINE
3716  014650  005726                                 TST    (SP)+         ;DISCARD THE REMAINDER. QUOTIENT IS
3717                                                                       ;NOW ON TOP OF THE STACK
3718  014652  012637  002210          27560          MOV    (SP)+,QSEC    ;(BETWEEN 0-13/8)
3719  014656  022737  000014  002210  27570          CMP    #14,QSEC
3720  014664  001002                  27580          BNE    5$
3721  014666  005337  002210          27590          DEC    QSEC          ;'QSEC' CONTAINS RANDOM SEC #
3722                                  27600
3723  014672  022737  077777  025370  27610    5$:   CMP    #77777,RSCYLH ;SELECT SURFACE # RANDOMLY
3724  014700  101005                  27620          BHI    6$
3725  014702  012737  000020  002206  27630          MOV    #BIT4,QSUR    ;SURFACE 1
3726                                  27640                                ;R1 CONTAINS THE MAXM WORD COUNT BASED ON
3727                                  27650                                ;AVAILABLE DISK SPACE.
3728                                  27660                                ;R2 CONTAINS THE MAXM WORD COUNT BASED ON
3729                                  27670                                ;AVAILABLE MEMORY SPACE.
3730  014710  005001                  27680          CLR    R1
3731  014712  000404                  27690          BR     7$
3732                                  27700
3733  014714  005037  002206          27710    6$:   CLR    QSUR          ;SURFACE 0
3734  014720  012701  000014          27720          MOV    #14,R1        ;14 SECTORS ON SUR 0
3735                                  27730
3736                                  27740    ;ASSUMING THAT ENOUGH MEMORY IS AVAILABLE THE MAXIMUM TRANSFER THAT CAN
3737                                  27750    ;TAKE PLACE IS 20K WORDS. IF THE RANDOM CYLINDER # > OR = TO 307 AND THE
3738                                  27760    ;SURFACE IS 1, CHANCES ARE THAT THE NUMBER OF WORDS TO BE TRANSFERRED MAY
3739                                  27770    ;BE GREATER THAN THE SPACE AVAILABLE ON THE DISK.  IN THAT CASE, THE WORDS
3740                                  27780    ;COUNT IS TO BE SELECTED IN SUCH AWAY THAT THIS OVERFLOW DOES NOT OCCUR.
3741                                  27790
3742  014724  023727  002204  000306  27800    7$:   CMP    QCYL,#306     ;IS THE RANDOM CYL # GREATER THAN 306?
3743  014732  002003                  27810          BGE    9$
3744  014734  012701  177777          27820    8$:   MOV    #177777,R1    ;IF YES, MAKE SURE THAT THE
37-5  014740  000431                  27830          BR     21$           ;WORD COUNT IS SELECTED PROPERLY
3746                                  27840                                ;COMPUTE MAXM WORD COUNT BASED ON
3747                                  27850                                ;AVAILABLE DISK SPACE
3748  014742  012700  000014          27860    9$:   MOV    #14,R0        ;COMPUTE # OF SECTORS AVAILABLE ON
3749  014746  163700  002210          27870          SUB    QSEC,R0       ;CYLINDERS SELECTED
3750  014752  060001                  27880          ADD    R0,R1
3751  014754  012703  000312          27890          MOV    #312,R3       ;COMPUTE # OF SECTORS AVAILABLE
3752  014760  163703  002204          27900          SUB    QCYL,R3       ;BEYOND THE CYLINDER SELECTED
3753  014764  012746  000030                         MOV    #30,-(SP)            ;;PUT THE MULTIPLER ON THE STACK
3754  014770  010346                                 MOV    R3,-(SP)             ;;PUT THE MULTIPLICAND ON THE STACK
3755  014772  004737  024526                         JSR    PC,@#$MULT    ;;CALL THE MULTPLY ROUTINE
3756  014776  012616                                 MOV    (SP)+,(SP)    ;;DISREGARD THE MSB'S
3757  015000  012603                                 MOV    (SP)+,R3             ;;GET THE LSB'S OF THE PRODUCT
3758  015002  060103                  27920          ADD    R1,R3         ;COMPUTE TOTAL # OF SECTORS AVAILABLE
```

```
3759  015004  012746  000400                          MOV     #400,-(SP)          ;;PUT THE MULTIPLER ON THE STACK
3760  015010  010346                                  MOV     R3,-(SP)            ;;PUT THE MULTIPLICAND ON THE STACK
3761  015012  004737  024526                          JSR     PC,@#$MULT          ;;CALL THE MULTPLY ROUTINE
3762  015016  012616                                  MOV     (SP)+,(SP)          ;;DISREGARD THE MSB'S
3763  015020  012603                                  MOV     (SP)+,R3            ;;GET THE LSB'S OF THE PRODUCT
3764  015022  010301                        27940     MOV     R3,R1               ;COMPUTE TOTAL # OF WORDS-SPACE
3765                                         27950                                 ;AVAILABLE ON DISK FROM THE SELECTED
3766                                         27960                                 ;CYL #
3767                                         27970                                 ;COMPUTE MAXM WORD COUNT BASED ON
3768                                         27980                                 ;AVAILABLE MEMORY SPACE.
3769  015024  004737  025224                27990  21$:     JSR     PC,$RAND            ;GENERATE RANDOM NUMBER
3770  015030  025372                        28000             RSBAL
3771
3772  015032  013746  025374                          MOV     RSBAH,-(SP)         ;PUT LOW DIVIDEND ON STACK
3773  015036  005046                                  CLR     -(SP)               ;CLEAR HIGH DIVIDEND AND PUSH
3774                                                                              ;IT ON STACK
3775  015040  013746  002230                          MOV     BAMAP,-(SP)         ;PUSH DIVISOR ON STACK
3776  015044  004737  024640                          JSR     PC,@#$DIV           ;GO TO THE 'DIVIDE' SUBROUTINE
3777  015050  005726                                  TST     (SP)+               ;DISCARD THE REMAINDER. QUOTIENT IS
3778                                                                              ;NOW ON TOP OF THE STACK
3779  015052  012637  002214                28020     MOV     (SP)+,QBUSAD        ;BE USED
3780  015056  006337  002214                28030     ASL     QBUSAD
3781  015062  063737  002552  002214        28040     ADD     BASEBA,QBUSAD       ;FORM THE RANDOM BUS-ADDRESS
3782                                         28050                                 ;BY ADDING RANDOM OFFSET TO
3783                                         28060                                 ;THE BASE BUS-ADDRESS
3784  015070  013703  002554                28070     MOV     MAXBA,R3            ;COMPUTE # OF WORDS THAT
3785  015074  163703  002214                28080     SUB     QBUSAD,R3           ;CAN BE TRANSFERRED, USING THE
3786  015100  000241                        28090     CLC
3787  015102  006003                        28100     ROR     R3                  ;ABOVE GENERATED BUS-ADDRESS WITHOUT
3788  015104  010302                        28110     MOV     R3,R2               ;CAUSING A NXM
3789                                         28120
3790  015106  020201                        28130  10$:     CMP     R2,R1               ;SELECT SMALLER OF THE TWO
3791  015110  103401                        28140     BLO     11$                 ;WORD-COUNTS THAT WILL BE
3792                                         28150                                 ;USED FOR GENERATING A RANDOM
3793  015112  010103                        28160     MOV     R1,R3               ;WORD COUNT)
3794                                         28170
3795                                         28180                                 ;R3 CONTAINS THE MAXM WORD COUNT
3796                                         28190                                 ;POSSIBLE. COMPUTE THE WORD COUNT
3797                                         28200                                 ;MAPPING FACTOR FROM THIS.
3798  015114                                 28210  11$:
3799
3800  015114  012746  177777                          MOV     #177777,-(SP)       ;PUT LOW DIVIDEND ON STACK
3801  015120  005046                                  CLR     -(SP)               ;CLEAR HIGH DIVIDEND AND PUSH
3802                                                                              ;IT ON STACK
3803  015122  010346                                  MOV     R3,-(SP)            ;PUSH DIVISOR ON STACK
3804  015124  004737  024640                          JSR     PC,@#$DIV           ;GO TO THE 'DIVIDE' SUBROUTINE
3805  015130  005726                                  TST     (SP)+               ;DISCARD THE REMAINDER. QUOTIENT IS
3806                                                                              ;NOW ON TOP OF THE STACK
3807  015132  005216                        28220     INC     (SP)
3808  015134  012637  002232                28230     MOV     (SP)+,WCMAP         ;WORD COUNT MAPPING FACTOR
3809                                         28240
3810  015140  004737  025224                28250     JSR     PC,$RAND            ;GENERATE A RANDOM NUMBER
3811  015144  025376                        28260     RSWCL
3812
3813  015146  013746  025400                          MOV     RSWCH,-(SP)         ;PUT LOW DIVIDEND ON STACK
3814  015152  005046                                  CLR     -(SP)               ;CLEAR HIGH DIVIDEND AND PUSH
```

```
                                                                ;IT ON STACK
015154 013746 002232                            MOV     WCMAP,-(SP)     ;PUSH DIVISOR ON STACK
015160 004737 024640                            JSR     PC,@#$DIV       ;GO TO THE 'DIVIDE' SUBROUTINE
015164 005726                                   TST     (SP)+           ;DISCARD THE REMAINDER. QUOTIENT IS
                                                                ;NOW ON TOP OF THE STACK
015166 012637 002216            28280           MOV     (SP)+,QWRCNT    ;'QWRCNT' CONTAINS THE RANDOM
                                28290                           ;WORD COUNT THAT WILL BE USED
015172 005737 002216            28300           TST     QWRCNT          ;MAKE SURE THE WORD COUNT IS
015176 003004                   28310           BGT     12$             ;NOT 0
015200 005437 002216            28320           NEG     QWRCNT          ;TAKE CARE OF ZERO AND NEG NMBRS
015204 005237 002216            28330           INC     QWRCNT
015210 113700 002202            28340   12$:    MOVB    QDRV,R0
015214 000241                   28350           CLC
015216 006000                   28360           ROR     R0
015220 006000                   28370           ROR     R0              ;POSITION THE DRIVE NUMBER IN
015222 006000                   28380           ROR     R0              ;BITS 15,14,13
015224 006000                   28390           ROR     R0
015226 013701 002204            28400           MOV     QCYL,R1
015232 000301                   28410           SWAB    R1
015234 000241                   28420           CLC
015236 006001                   28430           ROR     R1              ;POSITION THE CYLINDER NUMBER
015240 006001                   28440           ROR     R1              ;IN BITS 12-5
015242 006001                   28450           ROR     R1
015244 050100                   28460           BIS     R1,R0
015246 053700 002210            28470           BIS     QSEC,R0         ;R0 CONTAINS THE COMPLETE DISK
015252 053700 002206            28480           BIS     QSUR,R0         ;ADDRESS
015256 010025                   28490           MOV     R0,(R5)+        ;INSERT RKDA IN THE PARAMETER TABLE
                                28500                           ;(FOR THE 8 COMMANDS)
                                28510                           ;WHICH FUNCTION?
                                28520                           ;0-READ CHECK, 1-READ, 2-WRITE
015260 022737 000001 002212     28530           CMP     #1,QFNC
015266 001412                   28540           BEQ     2$              ;READ
015270 003014                   28550           BGT     14$             ;READ CHECK
015272 012725 000002            28560   1$:     MOV     #2,(R5)+        ;WRITE FUNCTION CODE
015276 023727 025400 077777     28570           CMP     RSWCH,#77777    ;SHOULD WRITE CHECK BE DONE
015304 101010                   28580           BHI     15$             ;AFTER THE WRITE?
015306 052714 000040            28590           BIS     #BIT5,(R4)      ;SET FLAG IN KEY TO INDICATE
                                28600                           ;THAT WRITE CHECK SHOULD BE
                                28610                           ;DONE FOLLOWING THE WRITE
015312 000405                   28620           BR      15$
                                28630
015314 012725 000004            28640   2$:     MOV     #4,(R5)+        ;READ FUNCTION CODE
015320 000402                   28650           BR      15$
                                28660
015322 012725 000012            28670   14$:    MOV     #12,(R5)+       ;READ CHECK FUNCTION CODE
                                28680
015326 013715 002216            28690   15$:    MOV     QWRCNT,(R5)     ;INSERT THE WORD COUNT (RKWC)
015332 005425                   28700           NEG     (R5)+           ;(2'S COMPLEMENT)
015334 013725 002214            28710           MOV     QBUSAD,(R5)+    ;INSERT THE BUS ADDRESS (RKBA) FOR
                                28720                           ;THIS COMMAND IN THE PARAMETER
                                28730                           ;TABLE
015340 053724 002202            28740   16$:    BIS     QDRV,(R4)+      ;SET THE DRIVE NUMBER INSIDE
                                28750                           ;THE KEY FOR THIS COMMAND
015344 020427 002026            28760           CMP     R4,#KEY+20      ;GENERATED 8 COMMANDS IN
                                28770                           ;THE QUEUE?
015350 001402                   28780           BEQ     17$
```

```
 3871  015352  000137  014372          28790            JMP     GEN1            ;IF NOT, GO BACK AND GENERATE
 3872                                  28800                                    ;THE NEXT COMMAND AND THE
 3873                                  28810                                    ;PARAMETERS (RKWC, BA, DA)
 3874  015356  005237  015372          28820    17$:    INC     ODDEVN          ;CHANGE FROM ODD/EVEN TO EVEN/ODD
 3875  015362  042737  177776  015372  28830            BIC     #177776,ODDEVN  ;KEEP ONLY ONE BIT
 3876  015370  000207                  28840            RTS     PC              ;ALL 8 COMMANDS HAVE BEEN
 3877                                  28850                                    ;GENERATED IN THE TASK-QUEUE
 3878  015372  000000                  28860    ODDEVN: 0
```

```
 3879                                          28880   ;ENTER THIS CODE IF SW 9 HAS BEEN SET AND LOOPING HAS TO BE DONE ON ERROR
 3880                                          28890   ;CONTROL IS TRANFERRED TO THIS, ON RETURN FROM THE ERROR HANDLER 'SERROR'.
 3881                                          28900
 3882                                          28910
 3883  015274  004737  015454                 28920   EXCRLUP: JSR    PC,CLRERR       ;WAIT FOR OTHER DRIVES TO GET DONE
 3884                                          28930                                   ;THEN ISSUE A CONTROL RESET
 3885  015400  010146                         28940            MOV    R1,-(SP)
 3886  015402  012701  002006                 28950            MOV    #KEY,R1         ;CLEAR OUT THE COMMAND-KEYS
 3887  015406  042721  114220                 28960   1$:      BIC    #!14220,(R1)+
 3888  015412  020127  002026                 28970            CMP    R1,#KEY+20
 3889  015416  001373                         28980            BNE    1$
 3890  015420  012601                         28990            MOV    (SP)+,R1
 3891  015422  004737  015436                 29000            JSR    PC,CLRFLGS      ;CLEAR OUT THE VARIOUS FLAGS PERTAINING
 3892                                          29010                                   ;TO THE 8 COMMANDS IN THE Q
 3893  015426  012706  001100                 29020            MOV    #STACK,SP       ;REESTABLISH STACK POINTER
 3894  015432  000137  010372                 29030            JMP    @MNGER          ;START OVER AGAIN, PROCESS THE COMMANDS
 3895                                          29040                                   ;IN THE Q AGAIN. NOTE THAT ON LOOPING
 3896                                          29050                                   ;(ON EROR, WITH SW 9) AN ATTEMPT IS MADE
 3897                                          29060                                   ;TO RECREATE THE SET OF EVENTS WHICH LED TO
 3898                                          29070                                   ;ERROR.
 3899                                          29080
 3900                                          29090
 3901                                          29100
 3902                                          29110
 3903                                          29120   ;CLRFLGS
 3904                                          29130   ;THIS ROUTINE CLEARS OUT THE VARIOUS FLAGS USED FOR THE 8 COMMANDS IN THE QUEUE.
 3905                                          29140
 3906  015436  012700  002126                 29150   CLRFLGS: MOV    #BUSY,R0        ;CLEAR THE 8 BUSY FLAGS
 3907  015442  005020                         29160   1$:      CLR    (R0)+
 3908  015444  020027  002162                 29170            CMP    R0,#QSCNT+2     ;ALL DONE?
 3909  015450  001374                         29180            BNE    1$              ;NO
 3910  015452  000207                         29190            RTS    PC
```

```
 3911                                                  29210  ;CLRERR
 3912                                                  29220  ;THIS ROUTINE IS ENTERED WHEN A HARD ERROR HAS OCCURRED AND IT HAS TO BE
 3913                                                  29230  ;CLEARED.  THE DRIVES THAT HAVE BEEN BUSY ARE CHECKED TO SEE IF THEIR R/W/S
 3914                                                  29240  ;RDY BIT HAS SET.  WHEN R/W/S IS SET, CHECKING IS DONE FOR ANY ERROR. IF A
 3915                                                  29250  ;ERROR OCCURED IT IS REPORTED, IF NOT, APPROPRIATE FLAGS ARE SET AND
 3916                                                  29260  ;CLEARED FOR THAT DRIVE. AFTER ABOVE IS DONE FOR ALL DRIVES THAT HAVE BEEN
 3917                                                  29270  ;SEEKING, A CONTROL RESET IS ISSUED TO CLEAR THE HARD ERROR.
 3918                                                  29280
 3919  015454  010446                                 29290  CLRERR: MOV     R4,-(SP)        ;SAVE R4, R4 ON THE STACK
 3920  015456  010546                                 29300          MOV     R5,-(SP)
 3921  015460  005005                                 29310          CLR     R5
 3922  015462  005077  164242                         29320          CLR     @RKDA
 3923                                                  29330
 3924  015466  105765  002126                         29340  1$:     TSTB    BUSY(R5)        ;WAS THIS DRIVE BUSY SEEKING?
 3925  015472  100035                                 29350          BPL     4$              ;NO
 3926  015474  005037  002172                         29360          CLR     TIMER
 3927  015500  032777  000100  164210                 29370  2$:     BIT     #RWS,@RKDS      ;R/W/S SET?
 3928  015506  001015                                 29380          BNE     3$              ;YES
 3929  015510  005237  002172                         29390          INC     TIMER           ;KEEP TIME
 3930  015514  001371                                 29400          BNE     2$              ;WAIT FOR R/W/S RDY
 3931  015516  004737  021652                         29410          JSR     PC,RG4SDR       ;GET RKCS, ER, DS, DA AND DRIVE # FOR
 3932                                                  29420                                  ;TYPING SERIAL DRIVE #
 3933  015522  104004                                 29430          ERROR   4               ;R/W/S READY DID NOT SET
 3934                                                  29440                                  ;FOR THIS DRIVE, WAITED LONG ENOUGH.
 3935  015524  032777  001000  164164                 29450          BIT     #SIN,@RKDS      ;SIN ERROR ON THIS DRIVE?
 3936  015532  001403                                 29460          BEQ     3$
 3937  015534  004737  021652                         29470    .     JSR     PC,RG4SDR       ;GET RKCS, ER, DS, DA AND DRIVE # FOR
 3938                                                  29480                                  ;TYPING SERIAL DRIVE #
 3939  015540  104016                                 29490          ERROR   16              ;SIN OCCURED ON THIS DRIVE
 3940                                                  29500
 3941  015542  116504  002126                         29510  3$:     MOVB    BUSY(R5),R4     ;FORM THE ADDRESS OF THE
 3942  015546  042704  177760                         29520          BIC     #177760,R4      ;KEY WHICH MADE THIS DRIVE
 3943  015552  062704  002036                         29530          ADD     #KEY,R4 ;BUSY
 3944                                                  29540
 3945  015556  042714  010000                         29550          BIC     #10000,(R4)     ;CLEAR HIGH PRIORITY BIT, IF SET
 3946  015562  105065  002126                         29560          CLRB    BUSY(R5)        ;MAKE THIS DRIVE FREE, AVAILABLE
 3947                                                  29570
 3948                                                  29580
 3949  015566  062777  020000  164134                 29590  4$:     ADD     #20000,@RKDA    ;ADDRESS THE NEXT POSSIBLE DRIVE
 3950  015574  005205                                 29600          INC     R5              ;INCREMENT COUNT
 3951  015576  022705  000010                         29610          CMP     #10,R5          ;ALL DONE
 3952  015602  001331                                 29620          BNE     1$              ;NO
 3953                                                  29630
 3954  015604  004737  020032                         29640          JSR     PC,CRCMND       ;SAVE INFO ABOUT THE PAST & PRESENT COMAND
 3955  015610  104415                                 29650          CON.RESET
 3956  015612  012605                                 29660          MOV     (SP)+,R5        ;RESTORE R4,R5
 3957  015614  012604                                 29670          MOV     (SP)+,R4
 3958  015616  000207                                 29680          RTS     PC              ;RETURN
```

```
                                                29700   ;CLRSIN
                                                29710   ;THIS ROUTINE IS ENTERED WHEN THERE IS A 'SIN' ERROR.  AT TIME OF ENTRY
                                                29720   ;RKDA CONTAINS THE DRIVE # THAT GAVE 'SIN'.  A DRIVE RESET IS DONE ON THAT
                                                29730   ;DRIVE.  AFTER IT IS DONE, ROUTINE 'CLRHE' IS ENTERED, TO WAIT FOR THE
                                                29740   ;OTHER DRIVES THAT HAVE BEEN DOING SEEKS.  WHEN ALL THE DRIVES GIVE
                                                29750   ;'R/W/S RDY', A CONTROL RESET IS DONE, RETURN IS MADE BACK TO THIS
                                                29760   ;ROUTINE-'CLRSIN'- AND FINALLY CONTROL IS TRANSFERRED BACK TO THE MAIN
                                                29770   ;PROGRAM.
                                                29780
3968  015620  017737  164104  002216            29790   CLRSIN: MOV     @RKDA,@WRCNT    ;SAVE DISK ADDRESS
3969  015626  004737  015454                    29800           JSR     PC,CLRERR       ;GO, WAIT FOR OTHER DRIVES TO COMPLETE
3970                                            29810                                   ;THEIR SEEKS(IF THEY ARE DOING ANY)
3971                                            29820                                   ;THEN DO CON.RESET TO CLR THE ERROR.
3972  015632  013777  002216  164070            29830           MOV     @WRCNT,@RKDA    ;ADDRESS THE DRIVE AGAIN
3973  015640  004737  020006                    29840           JSR     PC,DRCMND       ;SAVE INFO ABOUT THE PAST & PRESENT COMAND
3974  015644  012777  000015  164050            29850           MOV     #15,@RKCS       ;DO DRIVE RESET ON THE
3975                                            29860                                   ;DRIVE
3976                                            29870                                   ;INDICATED IN RKDA
3977  015652  104416                            29880           CON.RDY
3978  015654  005037  002172                    29890           CLR     TIMER
3979  015660  032777  000100  164030            29900   1$:     BIT     #RWS,@RKDS      ;WAIT FOR R/W/S RDY TO SET
3980  015666  001015                            29910           BNE     2$
3981  015670  005237  002172                    29920           INC     TIMER
3982  015674  001371                            29930           BNE     1$
3983  015676  004737  021652                    29940           JSR     PC,RG4SDR       ;GET RKCS, ER, DS, DA AND DRIVE # FOR
3984                                            29950                                   ;TYPING SERIAL DRIVE #
3985  015702  104004                            29960           ERROR   4               ;R/W/S RDY DID NOT SET AFTER
3986                                            29970                                   ;DOING DRIVE RESET, TIMED OUT
3987  015704  032777  001000  164004            29980           BIT     #SIN,@RKDS
3988  015712  001403                            29990           BEQ     2$
3989  015714  004737  021652                    30000           JSR     PC,RG4SDR       ;GET RKCS, ER, DS, DA AND DRIVE # FOR
3990                                            30010                                   ;TYPING SERIAL DRIVE #
3991  015720  104016                            30020           ERROR   16              ;A DRIVE RESET WAS DONE ON THIS DRIVE
3992                                            30030                                   ;TO CLEAR 'SIN', BUT 'SIN' DID NOT GET
3993                                            30040                                   ;CLEARED
3994                                            30050
3995  015722  004737  020032                    30060   2$:     JSR     PC,CRCMND       ;SAVE INFO ABOUT THIS COMMAND
3996  015726  104415                            30070           CON.RESET               ;DO IT TO CLEAR OUT MASK F/FS
3997  015730  000207                            30080           RTS     PC              ;EXIT FROM THIS ROUTINE
```

```
                                                30100   ;SINCNT
                                                30110   ;THIS ROUTINE IS ENTERED WHEN A 'SIN' ERROR OCCURS.  THE 'SIN' COUNT FOR
                                                30120   ;THE DRIVE GIVING 'SIN' IS INCREMENTED.  IF MORE THAN 5 'SIN' ERRORS
                                                30130   ;OCCURRED THE DRIVE IS DESELECTED.  AT THE TIME OF ENTRY R1 CONTAINS THE
                                                30140   ;DRIVE NUMBER THAT GAVE 'SIN' ERROR.
                                                30150   ;CALL:  JSR     PC,SINCNT
                                                30160   ;       -----           RETURN HERE IF SIN COUNT (MAXIMUM ALLOWABLE)
                                                30170   ;                       WAS EXCEEDED.
                                                30180   ;       -----           RETURN HERE IF TOTAL SIN COUNT LESS THAN MAXIMUM
                                                30190   ;                       ALLOWABLE.
                                                30200
                                                30210
4010  015732  105261  002322                  30220   SINCNT: INCB    SINCN(R1)       ;INCREMENT 'SIN' COUNT FOR THIS DRIVE
4011  015736  122761  000005  002322           30230           CMPB    #5,SINCN(R1)    ;5 ERRORS OCCURRED?
4012  015744  101403                           30240           BLOS    1$              ;YES
4013  015746  062716  000002                   30250           ADD     #2,(SP)         ;ADJUST PC FOR RETURN TO THE RIGHT POINT
4014  015752  000207                           30260           RTS     PC              ;RETURN
4015                                           30270
4016  015754  000137  015760                   30280   1$:     JMP     DSELCT          ;5 ERRORS OCCURRED, GO DESELECT
```

```
 4017                                          30300   ;DSELCT
 4018                                          30310   ;THIS ROUTINE IS ENTERED WHEN A DRIVE IS TO BE DESELECTED (TAKEN
 4019                                          30320   ;OUT OF SELECTION LIST), BECAUSE THE FATAL ERRORS ON THAT DRIVE
 4020                                          30330   ;HAS REACHED A MAXIMUM COUNT.  R1 CONTAINS THE DRIVE NUMBER THAT
 4021                                          30340   ;THAT IS TO BE DESELECTED. THE DRIVE IS DESELECTED IF 1. TOTAL SIN COUNT
 4022                                          30350   ;FOR THAT DRIVE REACHES THE MAXIMUM ALLOWABLE  2.IF  A FATAL ERROR
 4023                                          30360   ;LIKE DRIVE UNSAFE, DRIVE POWER LOW OCCURS.  3. IF  WPS GETS SET, OR DRY
 4024                                          30370   ;IS CLEAR.
 4025                                          30380
 4026                                          30390
 4027                                          30400
 4028  015760  012705  001753                 30410   DSELCT: MOV     #PDR-1,R5
 4029  015764  005205                          30420   1$:     INC     R5              ;LOCATE THE DRIVE (TO BE
 4030  015766  111503                          30430           MOVB    (R5),R3         ;DESELECTED) IN THE TABLE
 4031  015770  042703  177600                 30440           BIC     #177600,R3      ;DROP THE F FLAG
 4032  015774  020301                          30450           CMP     R3,R1           ;IS THIS THE ONE
 4033  015776  001404                          30460           BEQ     2$              ;CONTAINING AVAILABLE DRIVES
 4034  016000  022705  001764                 30470           CMP     #PDR+10,R5
 4035  016004  001367                          30480           BNE     1$
 4036                                          30490
 4037  016006  000474                          30500           BR      11$             ;DRVE# WAS NOT FOUND IN TABLE, EXIT
 4038                                          30510
 4039  016010  105715                          30520   2$:     TSTB    (R5)            ;IS THIS AN F TYPE DRIVE?
 4040  016012  100001                          30530           BPL     5$
 4041  016014  005202                          30540           INC     R2
 4042  016016  020527  001764                 30550   5$:     CMP     R5,#PDR+10      ;IS THIS DRIVE # THE LAST ENTRY IN TABLE?
 4043  016022  001406                          30560           BEQ     4$              ;YES
 4044  016024  010504                          30570           MOV     R5,R4
 4045  016026  005205                          30580           INC     R5              ;IF NOT, TAKE OUT THIS DRIVE # FROM
 4046  016030  112524                          30590   3$:     MOVB    (R5)+,(R4)+     ;THE MIDDLE AND PUSH UP THE
 4047  016032  022704  001763                 30600           CMP     #PDR+7,R4       ;REST OF THE ENTRIES
 4048  016036  001374                          30610           BNE     3$
 4049  016040  105065  177777                 30620   4$:     CLRB    -1(R5)          ;CLEAR LAST ENTRY IN TABLE
 4050                                          30630
 4051                                          30640                                   ;THE DRIVE # TYPED OUT WAS DESELECTED
 4052                                          30650                                   ;BECAUSE ERROR COUNT EXCEEDED THE
 4053                                          30660                                   ;MAXIMUM ALLOWABLE
 4054  016044  104400  003041                 30670           TYPE    ,MSG19          ;TYPE "DRVE DROPPED"
 4055  016050  010146                          30680           MOV     R1,-(SP)        ;PUSH DRIVE NUMBER ON STACK
 4056  016052  104402                          30690           TYPOS                   ;TYPE IT ON THE TERMINAL
 4057  016054     001                          30700           .BYTE   1
 4058  016055     000                          30710           .BYTE   0
 4059  016056  004737  026374                 30720           JSR     PC,SNOTYP       ;GO TYPE OUT SERIAL NO OF THE DRIVE,
 4060                                          30730                                   ;IF SW 1 IS SET.
 4061  016062  005337  001764                 30740           DEC     DRVPRS          ;DECREMENT THE TOTAL NUMBER OF
 4062                                          30750                                   ;DRIVES PRESENT
 4063  016066  004737  022304                 30760           JSR     PC,CHDPRS       ;CHECK IF ANY DRIVES PRESENT
 4064                                          30770                                   ;IF NONE GOT TO END OF PASS, $EOP
 4065
 4066  016072  012746  177777                                 MOV     #177777,-(SP)   ;PUT LOW DIVIDEND ON STACK
 4067  016076  005046                                          CLR     -(SP)           ;CLEAR HIGH DIVIDEND AND PUSH
 4068                                                                                  ;IT ON STACK
 4069  016100  013746  001764                                 MOV     DRVPRS,-(SP)    ;PUSH DIVISOR ON STACK
 4070  016104  004737  024640                                 JSR     PC,@#$DIV       ;GO TO THE 'DIVIDE' SUBROUTINE
 4071  016110  005726                                          TST     (SP)+           ;DISCARD THE REMAINDER. QUOTIENT IS
 4072                                                                                  ;NOW ON TOP OF THE STACK
```

```
 4073  016112  012637  002220          30790            MOV     (SP)+,DRMAP     ;TO BE USED FOR GENERATING RANDOM
 4074                                  30800                                    ;DRIVE NUMBERS.
 4075  016116  012704  002006          30810            MOV     #KEY,R4         ;DESELECT THE COMMANDS
 4076  016122  011405                  30820    6$:     MOV     (R4),R5         ;IN THE 'COMMAND Q' CORRESPONDING
 4077  016124  042705  177770          30830            BIC     #177770,R5      ;TO THE DESELECTED DRIVE
 4078  016130  020105                  30840            CMP     R1,R5
 4079  016132  001002                  30850            BNE     7$
 4080  016134  052714  104000          30860            BIS     #104000,(R4)    ;INDICATE COMMAND DESELECTED
 4081  016140  005724                  30870    7$:     TST     (R4)+           ;(AND COMPLETED)
 4082  016142  022704  002026          30880            CMP     #KEY+20,R4
 4083  016146  001365                  30890            BNE     6$
 4084                                  30900
 4085  016150  005702                  30910    8$:     TST     R2              ;F TYPE DRIVE?
 4086  016152  001412                  30920            BEQ     11$             ;YES - JUST EXIT
 4087  016154  032701  000001          30930            BIT     #1,R1           ;ODD OR EVEN DRIVE NUMBER
 4088  016160  001403                  30940            BEQ     9$
 4089  016162  042701  000001          30950            BIC     #1,R1
 4090  016166  000402                  30960            BR      10$
 4091  016170  052701  000001          30970    9$:     BIS     #1,R1
 4092  016174  000137  015760          30980   10$:     JMP     DSELCT          ;DROP CORRESPONDING DRIVE
 4093                                  30990
 4094                                  31000
 4095  016200  000137  010352          31010   11$:     JMP     BEGNEX          ;GO RESTART EXERSISOR PART OF TEST
```

```
 4096                                          31030   ;CHKDRV
 4097                                          31040   ;THIS ROUTINE CHECKS FOR FATAL ERRORS OF THE DRIVE LIKE DPL, DRV
 4098                                          31050   ;WPS.  IF ANY ONE OF THESE ERRORS OCCUR THE DRIVE IS DESELECTED
 4099                                          31060   ;AND NO MORE FUNCTIONS WILL BE PERFORMED ON THAT DRIVE.  R1
 4100                                          31070   ;CONTAINS THE DRIVE NUMBER TO BE CHECKED.
 4101                                          31080
 4102                                          31090   ;*NOTE 1: IN THE ERROR MESSAGE WHERE RKDA IS PRINTED OUT, IT GIVES
 4103                                          31100   ;*ONLY THE DRIVE NUMBER (NOT CYLINDER, SUFACE AND SECTOR).
 4104                                          31110
 4105                                          31120   ;CALL:  JSR     PC,CHKDRV
 4106                                          31130   ;       ----- RETURN HERE IF ANY FATAL ERROR OCCURED
 4107                                          31140   ;       ----- RETURN HERE IF THERE WAS NO FATAL ERROR
 4108                                          31150
 4109  016204  017746  163520                 31160   CHKDRV: MOV     @RKDA,-(SP)     ;SAVE RKDA
 4110  016210  010146                         31170           MOV     R1,-(SP)         GET DRIVE #
 4111  016212  000241                         31180           CLC
 4112  016214  006016                         31190           ROR     (SP)
 4113  016216  006016                         31200           ROR     (SP)
 4114  016220  006016                         31210           ROR     (SP)
 4115  016222  006016                         31220           ROR     (SP)
 4116  016224  012677  163500                 31230           MOV     (SP)+,@RKDA     ;ADDRESS THE DRIVE TO BE CHECKED
 4117  016230  032777  010000  163460         31240           BIT     #DPL,@RKDS      ;DRIVE POWER LOW?
 4118  016236  001403                         31250           BEQ     1$
 4119  016240  004737  021652                 31260           JSR     PC,RG4SDR       ;GET RKCS, ER, DS, DA AND DRIVE #
 4120                                          31270                                   ;FOR TYPING SERIAL NUMBER
 4121  016244  104035                         31280           ERROR   35              ;DRIVE POWER LO, *NOTE 1 ABOVE
 4122  016246  032777  002000  163442         31290   1$:     BIT     #DRU,@RKDS      ;DRIVE
 4123  016254  001403                         31300           BEQ     2$
 4124  016256  004737  021652                 31310           JSR     PC,RG4SDR       ;GET RKCS, ER, DS, DA AND DRIVE #
 4125                                          31320                                   ;FOR TYPING SERIAL NUMBER
 4126  016262  104036                         31330           ERROR   36              ;DRIVE UNSAFE BIT IS SET
 4127                                          31340                                   ;*NOTE 1 ABOVE
 4128  016264  032777  000040  163424         31350   2$:     BIT     #WPS,@RKDS      ;WRITE PROTECT SET?
 4129  016272  001403                         31360           BEQ     3$
 4130  016274  004737  021652                 31370           JSR     PC,RG4SDR       ;GET RKCS, ER, DS, DA AND DRIVE #
 4131                                          31380                                   ;FOR TYPING SERIAL NUMBER
 4132  016300  104037                         31390           ERROR   37              ;WPS SET, CHECK WRTE PROTECT SWTCH ON DRIVE
 4133                                          31400                                   ;*NOTE 1 ABOVE
 4134  016302  032777  000200  163406         31410   3$:     BIT     #DRY,@RKDS      ;DRIVE READY CLEAR?
 4135  016310  001004                         31420           BNE     4$
 4136  016312  004737  021652                 31430           JSR     PC,RG4SDR       ;GET RKCS, ER, DS, DA AND DRIVE #
 4137                                          31440                                   ;FOR TYPING SERIAL NUMBER
 4138  016316  104034                         31450           ERROR   34              ;DRIVE READY CLEAR, SHOULD BE SET
 4139                                          31460                                   ;*NOTE 1 ABOVE
 4140  016320  000411                         31470           BR      5$
 4141                                          31480
 4142  016322  032777  012040  163366         31490   4$:     BIT     #12040,@RKDS    ;ANY ERROR?
 4143  016330  001005                         31500           BNE     5$              ;YES
 4144  016332  012677  163372                 31510           MOV     (SP)+,@RKDA     ;RESTORE RKDA
 4145  016336  062716  000002                 31520           ADD     #2,(SP)         ;ADJUST RETURN ADDRESS
 4146  016342  000207                         31530           RTS     PC
 4147                                          31540
 4148  016344  000137  015760                 31550   5$:     JMP     DSELCT
```

```
 4149                                                31570   ;GENBUF
 4150                                                31580   ;THIS ROUTINE GENERRAES A BUFFER FULL OF RANDOM DATA WORDS. THIS BUFFER
 4151                                                31590   ;IS THEN USED TO WRITE DATA ON THE DISK.  AT THE TIME OF ENTRY, RKDA
 4152                                                31600   ;CONTAINS THE DISK ADDRESS WHERE WRITE WILL BE DONE. SEED WORDS USED FOR
 4153                                                31610   ;THE RANDOM NUMBERS ARE:
 4154                                                31620   ;       1) ABSOLUTE DISK ADDRESS (DRIVE #, CYL #, SEC #, SUR #) - $HINUM
 4155                                                31630   ;       2) COMPLEMENT OF THE ABOVE WORD
 4156                                                31640   ;CALL:  JSR     R5,GENBUF
 4157                                                31650   ;       X               ;X IS THE WORD COUNT (2'S COMPLEMENT)
 4158                                                31660   ;       Y               ;Y IS THE STARTING ADDRESS OF THE
 4159                                                31670                           ;MEMORY BUFFER.
 4160                                                31680
 4161  016350  104413                                31690   GENBUF: SAVREG                          ;SAVE REGISTERS
 4162  016352  016504  000002                        31700           MOV     2(R5),R4                ;GET STARTING ADDRESS OF BUFFER
 4163  016356  011503                                31710           MOV     (R5),R3                 ;GET WORD COUNT (# OF WORDS TO
 4164                                                31720                                           ;BE GENERATED)
 4165                                                31730
 4166  016360  017702  163344                        31740           MOV     @RKDA,R2
 4167  016364  010237  025404                        31750   1$:     MOV     R2,RSDTH                ;GET THIS RANDOM SEED
 4168  016370  010237  025402                        31760           MOV     R2,RSDTL
 4169  016374  005137  025402                        31770           COM     RSDTL                   ;GET LO RANDOM SEED
 4170                                                31780
 4171  016400  022703  177400                        31790           CMP     #-400,R3                ;IF THE BUFFER IS MORE THAN
 4172  016404  003003                                31800           BGT     2$                      ;ONE SECTOR (400 WORDS) LONG,
 4173  016406  010305                                31810           MOV     R3,R5                   ;GENERATE THE BUFFER IN SUCH
 4174  016410  005003                                31820           CLR     R3                      ;A WAY THAT EACH SECTOR
 4175  016412  000404                                31830           BR      3$                      ;BEGINS WITH RANDOM DATA
 4176                                                31840
 4177  016414  062703  000400                        31850   2$:     ADD     #400,R3                 ;WORDS GENERATED USING THAT
 4178  016420  012705  177400                        31860           MOV     #-400,R5                ;SECTOR ADDRESS AS THE RANDOM
 4179                                                31870                                           ;SEED
 4180  016424  010524                                31880   3$:     MOV     R5,(R4)+                ;FIRST WORD OF EVERY SECTOR IS
 4181                                                31890                                           ;A WORD COUNT (2'S COMP) INDICATING  #
 4182                                                31900                                           ;OF WORDS ACTUA _Y WRITTEN IN THAT SECTOR
 4183  016426  005205                                31910           INC     R5                      ;ALL DONE?
 4184  016430  001427                                31920           BEQ     8$
 4185                                                31930
 4186  016432  004737  025224                        31940   4$:     JSR     PC,$RAND                ;GENERATE DATA WORDS
 4187  016436  025402                                31950           RSDTL
 4188  016440  012737  000002  016550                31960           MOV     #2,RCNT
 4189  016446  013737  025404  016552                31970           MOV     RSDTH,RNUM
 4190  016454  000406                                31980           BR      6$
 4191  016456  005337  016550                        31990   5$:     DEC     RCNT
 4192  016462  001763                                32000           BEQ     4$
 4193  016464  013737  025402  016552                32010           MOV     RSDTL,RNUM
 4194  016472  005737  016552                        32020   6$:     TST     RNUM
 4195  016476  001767                                32030           BEQ     5$
 4196  016500  013724  016552                        32040           MOV     RNUM,(R4)+              ;FILL THE BUFFER. DON'T USE
 4197  016504  005205                                32050           INC     R5                      ;ALL DONE?
 4198  016506  001351                                32060           BNE     4$                      ;NO
 4199                                                32070
 4200  016510  005703                                32080   8$:     TST     R3                      ;ANY MORE DATA WORDS (FOR
 4201  016512  001412                                32090           BEQ     10$                     ;REST OF THE SECTORS)?
 4202                                                32100                                           ;YES
 4203  016514  010246                                32110           MOV     R2,-(SP)
 4204  016516  042716  177760                        32120           BIC     #177760,(SP)            ;(ABSOLUTE DISK ADDRESS & ITS
```

```
 4205  016522  022726  000013          32130            CMP     #13,(SP)+       ;COMPLEMENT) TO USE FOR
 4206  016526  001002                  32140            BNE     9$              ;GENERATING DATA WORDS
 4207  016530  062702  000004          32150            ADD     #4,R2           ;OF THE NEXT BLOCK
 4208                                  32160
 4209  016534  005202                  32170    9$:     INC     R2              ;HI RANDOM SEED
 4210  016536  000712                  32180            BR      1$
 4211                                  32190
 4212  016540  104414                  32200    10$:    RESREG                  ;RESTORE REGISTERS
 4213  016542  062705  000004          32210            ADD     #4,R5           ;ADJUST RETURN ADDRESS
 4214  016546  000205                  32220            RTS     R5              ;RETURN
 4215                                  32230
 4216  016550  000000                  32240    RCNT:   0
 4217  016552  000000                  32250    RNUM:   0
```

```
4218                                                    32270   ;DATCHK
4219                                                    32280   ;THIS ROUTINE IS ENTERED WHEN THE DATA THAT WAS READ FROM THE DISK IS TO
4220                                                    32290   ;BE CHECKED.  AT THE TIME OF ENTRY R3 CONTAINS THE OFFSET OF  POINTER TO
4221                                                    32300   ;THE ADDRESS OF THE PARAMETER LIST (RKCS,DA,WC,BA). DATA IS CHECKED IN
4222                                                    32310   ;BLOCKS OF 1 SECTOR (400 WORDS).  EACH BLOCK IS GENERATED USING THE SECTOR
4223                                                    32320   ;ADDRESS (AND ITS COMPLEMENT) AS RANDOM SEEDS.  WHEN A DATA MISCOMPARISON
4224                                                    32330   ;OCCURS THE BUS ADDRESS, EXPECTED AND RECEIVED DATA ARE REPORTED.
4225                                                    32340
4226  016554  104413                                   32350   DATCHK: SAVREG                          ;SAVE R0-R5
4227  016556  016303  002532                           32360           MOV     PCMND(R3),R3            ;GET ADDRESS OF THE PARAMETER
4228                                                    32370                                           ;TABLE
4229  016562  016304  000004                           32380           MOV     4(R3),R4                ;GET WORD COUNT (2'S COMP)
4230  016566  016305  000006                           32390           MOV     6(R3),R5                ;GET BUS ADDRESS
4231  016572  011301                                   32400           MOV     (R3),R1                 ;GET DISK ADDRESS
4232                                                    32410
4233  016574  010146                                   32420           MOV     R1,-(SP)
4234  016576  004737  021620                           32430           JSR     PC,CROTLF               ;ROTATE BITS 15,14,13 TO
4235                                                    32440                                           ;0,1,2
4236  016602  012602                                   32450           MOV     (SP)+,R2                ;POP OFF DRIVE # FROM THE STACK
4237  016604  006302                                   32460           ASL     R2
4238  016606  062702  002412                           32470           ADD     #DATER,R2               ;FORM THE ADDRESS OF DATA ERROR COUNT
4239                                                    32480                                           ;FOR THIS DRIVE
4240  016612  012737  177764  002240                   32490           MOV     #-14,ECOUNT
4241  016620  011327  025404                           32500           MOV     (R3),RSDTH              ;CREATE RANDOM SEEDS TO
4242  016624  013737  025404  025402                   32510           MOV     RSDTH,RSDTL             ;BE USED FOR CHECKING DATA
4243  016632  005137  025402                           32520           COM     RSDTL
4244                                                    32530
4245  016636  022704  177400                           32540   1$:     CMP     #-400,R4                ;DATA IS CHECKED IN 1 SECTOR
4246  016642  003003                                   32550           BGT     2$                      ;BLOCKS. EACH SECTOR IS GENERATED
4247  016644  010403                                   32560           MOV     R4,R3                   ;USING THAT SECTOR ADDRESS
4248  016646  005004                                   32570           CLR     R4                      ;AS THE RANDOM SEED
4249  016650  000404                                   32580           BR      3$
4250                                                    32590
4251  016652  062704  000400                           32600   2$:     ADD     #400,R4
4252  016656  012703  177400                           32610           MOV     #-400,R3
4253  016662  012500                                   32620   3$:     MOV     (R5)+,R0                ;SAVE THE FIRST WORD OF THE SECTOR.
4254                                                    32630                                           ;FIRST WORD OF EVERY SECTOR IS A COUNT
4255                                                    32640                                           ;(2'S COMP) INDICATING # OF WORDS ACTUALY
4256                                                    32650                                           ;WRITTEN IN THAT SECTOR
4257  016664  005200                                   32660           INC     R0                      ;INCREMENT COUNT OF # OF WORDS  (WRITEN)
4258                                                    32670                                           ;IN THE SECTOR
4259  016666  005203                                   32680           INC     R3                      ;INCRMENT COUNT OF DATA WORDS TO BE CHECKED
4260  016670  001465                                   32690           BEQ     14$                     ;BRANCH, IF DONE
4261                                                    32700
4262  016672  004737  025224                           32710   4$:     JSR     PC,$RAND                ;GENERATE RANDOM DATA WORD
4263  016676  025402                                   32720           RSDTL
4264  016700  012737  000002  016550                   32730           MOV     #2,RCNT
4265  016706  013737  025404  016552                   32740           MOV     RSDTH,RNUM
4266  016714  000406                                   32750           BR      10$
4267  016716  005337  016550                           32760   9$:     DEC     RCNT
4268  016722  001763                                   32770           BEQ     4$
4269  016724  013737  025402  016552                   32780           MOV     RSDTL,RNUM
4270  016732  005737  016552                           32790   10$:    TST     RNUM
4271  016736  001767                                   32800           BEQ     9$
4272                                                    32810
4273  016740  005700                                   32820           TST     R0
```

;

```
 4274  016742  001401                    32830          BEQ     5$
 4275  016744  005200                    32840          INC     R0
 4276                                    32850
 4277  016746  023715  016552            32860  5$:     CMP     RNUM,(R5)       ;EXPCTD WORD = RECVD WORD?
 4278  016752  001431                    32870          BEQ     8$              ;YES
 4279                                    32880
 4280  016754  005700                    32890          TST     R0
 4281  016756  001005                    32900          BNE     6$
 4282  016760  005715                    32910          TST     (R5)
 4283  016762  001425                    32920          BEQ     8$
 4284  016764  005037  001164            32930          CLR     $REG1
 4285  016770  000403                    32940          BR      7$
 4286                                    32950
 4287  016772  013737  016552  001164    32960  6$:     MOV     RNUM,$REG1      ;SAVE EXPCTD DATA WORD
 4288  017000  005212                    32970  7$:     INC     (R2)            ;INCRMNT DATA EROR COUNT FOR THIS DRVE
 4289  017002  005737  002240            32980          TST     ECOUNT          ;STORE ONLY 12 (DEC) DATA ERRORS
 4290  017006  001413                    32990          BEQ     8$              ;IF MORE EXIT
 4291  017010  010537  001162            33000          MOV     R5,$REG0        ;SAVE ERROR BUS ADDRESS
 4292  017014  011537  001166            33010          MOV     (R5),$REG2      ;SAVE ERROR DATA WORD
 4293  017020  010137  001170            33020          MOV     R1,$REG3
 4294  017024  004737  021656            33030          JSR     PC,GTSDRV       ;SAVE DRIVE #, FOR TYPING SERIAL #
 4295  017030  104023                    33040          ERROR   23              ;DATA (COMPARISON) ERROR ON DOING
 4296                                    33050                                  ;READ FROM DISK NORMALLY ONLY 12 DATA
 4297                                    33060                                  ;ERRORS WILL BE REPORTED.  THROUGH
 4298                                    33070                                  ;CHECKING WILL BE DONE, ERRORS
 4299                                    33080                                  ;EXCEEDING 12 WON'T BE REPORTED.  IF
 4300                                    33090                                  ;YOU WANT MORE, CHANGE 'ECOUNT', TO
 4301                                    33100                                  ;WHATEVER # OF ERRORS YOU WANT REPORTED
 4302  017032  005237  002240            33110          INC     ECOUNT
 4303                                    33120
 4304  017036  005725                    33130  8$:     TST     (R5)+
 4305  017040  005203                    33140          INC     R3              ;DONE CHECKING?
 4306  017042  001313                    33150          BNE     4$
 4307                                    33160
 4308  017044  005704                    33170  14$:    TST     R4              ;ANY MORE SECTOR BLOCKS
 4309  017046  001427                    33180          BEQ     17$             ;TO CHECK? IF NOT, EXIT
 4310                                    33190
 4311  017050  010146                    33200          MOV     R1,-(SP)
 4312                                    33210                                  ;GET THE NEW RANDOM SEEDS
 4313  017052  042716  177760            33220          BIC     #177760,(SP)    ;(ABSOLUTE DISK ADDRESS & ITS COMPLEMENT)
 4314  017056  022726  000013            33230          CMP     #13,(SP)+       ;TO USE FOR GENERATING DATA WORDS
 4315  017062  001002                    33240          BNE     15$             ;OF THE NEXT BLOCK
 4316  017064  062701  000004            33250          ADD     #4,R1
 4317                                    33260
 4318  017070  005201                    33270  15$:    INC     R1
 4319  017072  032777  000002  162620    33280          BIT     #CSE,@RKER      ;IF THERE WAS A CSE THEN CHECK
 4320  017100  001403                    33290          BEQ     16$             ;ONLY THOSE SECTORS THAT WERE READ
 4321  017102  020177  162622            33300          CMP     R1,@RKDA
 4322  017106  001407                    33310          BEQ     17$
 4323  017110  010137  025404            33320  16$:    MOV     R1,RSDTH
 4324  017114  010137  025402            33330          MOV     R1,RSDTL
 4325  017120  005137  025402            33340          COM     RSDTL
 4326  017124  000644                    33350          BR      1$
 4327  017126  104414                    33360  17$:    RESREG                  ;RESTORE R0-R5
 4328  017130  000207                    33370          RTS     PC
```

```
                                .SBTTL  ROUTINE TO SIZE MEMORY

                                ;;**********************************************************************
                                ;*CALL:
                                ;*      JSR     PC,$SIZE
                                ;*      RETURN
                                ;*$LSTAD WILL CONTAIN:
                                ;*      WITH KT11 OPTION        -- LAST VIRTUAL ADDRESS OF THE LAST BANK
                                ;*      WITHOUT KT11 OPTION     -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
                                ;*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
                                ;*$KT11 IS THE MEMORY MANAGEMENT KEY
                                ;*BIT07 = 0 DON'T USE MEMORY MANAGEMENT
                                ;*      MUST BE SETUP BEFORE THE CALL
                                ;*BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
                                ;*      DETERMINED BY ROUTINE

017132  010046          $SIZE:  MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
017134  010146                  MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
017136  010246                  MOV     R2,-(SP)        ;;SAVE R2 ON THE STACK
017140  010346                  MOV     R3,-(SP)        ;;SAVE R3 ON THE STACK
017142  013746  000004          MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
017146  013746  000006          MOV     @#ERRVEC+2,-(SP)
017152  010600                  MOV     SP,R0           ;;SAVE THE STACK POINTER
                                ;;SET THE ERRVEC PS TO THE PRESENT PS
017154  013746  000034          MOV     @#TRAPVEC,-(SP) ;;SAVE CURRENT TRAP VECTOR
017160  012737  017170  000034  MOV     #64$,@#TRAPVEC  ;;SETUP NEW TRAP VECTOT
017166  104400                  TRAP                    ;;PUSH OLD PSW AND PC ON STACK
017170  016637  000002  000006  64$:    MOV     2(SP),@#ERRVEC+2        ;;SAVE PSW IN @#ERRVEC+2
017176  012716  017204          MOV     #65$,(SP)       ;;REPLACE OLD PC WITH NEW
017202  000002                  RTI                     ;;RESTORE PSW
017204  012637  000034  65$:    MOV     (SP)+,@#TRAPVEC ;;RESTORE OLD TRAP  VECTOR
017210  012701  003776          MOV     #3776,R1        ;;SETUP ADDRESS
017214  105727                  TSTB    (PC)+           ;;USE MEMORY MANAGEMENT?
017216  000200          $KT11:  .WORD   200             ;;SET TO USE MEMORY MANAGEMENT
017220  100062                  BPL     $CORE           ;;BR IF NO
017222  012737  017360  000004  MOV     #$KTNEX,@#ERRVEC ;;SET FOR TIMEOUT
017230  005737  177572          TST     @#SR0           ;;KT11 ARE YOU THERE?
017234  052737  100000  017216  BIS     #100000,$KT11   ;;YES--SET KT11 KEY
017242  005046                  CLR     -(SP)           ;;INITIALIZE FOR "PAR" LOADING
017244  012702  172340          MOV     #KIPAR0,R2      ;;ADDRESS OF FIRST "PAR"
017250  012703  000010          MOV     #^D8,R3         ;;LOAD EIGHT "PAR.'S" AND EIGHT "PDR.'S"
017254  012762  077406  177740  1$:     MOV     #77406,-40(R2)  ;;PDR = 4K, UP, READ/WRITE
017262  011622                  MOV     (SP),(R2)+      ;;LOAD "PAR"
017264  062716  000200          ADD     #200,(SP)       ;;UPDATE FOR NEXT "PAR"
017270  077307                  SOB     R3,1$           ;;LOOP UNTIL ALL EIGHT ARE LOADED
017272  012742  177600          MOV     #177600,-(R2)   ;;SETUP KIPAR7 FOR I/O
017276  005042                  CLR     -(R2)           ;;SETUP KIPAR6 FOR TESTING
017300  012737  017316  000004  MOV     #2$,@#ERRVEC    ;;CATCH TIMEOUT IF NO SR3
017306  012737  000020  172516  MOV     #20,@#SR3       ;;ENABLE 22 BIT MODE
017314  000401                  BR      3$              ;;THIS PDP-11 HAS A SR3 REGISTER
017316  022626          2$:     CMP     (SP)+,(SP)+     ;;CLEAN OFF THE STACK--NO SR3
017320  005237  177572  3$:     INC     @#SR0           ;;TURN ON MEMORY MANAGEMENT
017324  012737  017350  000004  MOV     #$KTOUT,@#ERRVEC ;;SET FOR TIME OUT
017332  005737  143776  4$:     TST     @#143776        ;;TRAP ON NON-EX-MEM
017336  062712  000040          ADD     #40,(R2)        ;;MAKE A 1K STEP
017342  023712  172356          CMP     @#KIPAR7,(R2)   ;;LAST ONE?
```

```
 4385  017346  101371                          BHI     4$              ;;NO--TRY IT
 4386  017350  011202                 $KTOUT:  MOV     (R2),R2         ;;GET LAST BANK+1
 4387  017352  005037  177572                  CLR     @#SR0           ;;TURN OFF MEMORY MANAGEMENT
 4388  017356  000421                          BR      $SIZEX
 4389  017360  042737  100000  017216  $KTNEX:  BIC     #100000,$KT11   ;;KT11 NON-EXISTENT
 4390  017366  012737  017416  000004  $CORE:   MOV     #$CROUT,@#ERRVEC ;;SET FOR TIMEOUT
 4391  017374  005002                          CLR     R2              ;;SET UP BANK
 4392  017376  062701  004000          1$:     ADD     #4000,R1        ;;INCREMENT BY 1K
 4393  017402  062702  000040                  ADD     #40,R2          ;;1K STEP
 4394  017406  005711                          TST     (R1)            ;;TRAP ON TIME OUT
 4395  017410  022701  177776                  CMP     #177776,R1      ;;LAST ONE
 4396  017414  001370                          BNE     1$              ;;NO--TRY AGAIN
 4397  017416  162701  004000  $CROUT:  SUB     #4000,R1
 4398  017422  162702  000040  $SIZEX:  SUB     #40,R2          ;;DROP BACK
 4399  017426  010006                          MOV     R0,SP           ;;RESTORE THE STACK
 4400  017430  012637  000006                  MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
 4401  017434  012637  000004                  MOV     (SP)+,@#ERRVEC
 4402  017440  010137  017462                  MOV     R1,$LSTAD       ;;LAST ADDRESS
 4403  017444  010237  017464                  MOV     R2,$LSTBK       ;;LAST BANK
 4404  017450  012603                          MOV     (SP)+,R3        ;;RESTORE R3
 4405  017452  012602                          MOV     (SP)+,R2        ;;RESTORE R2
 4406  017454  012601                          MOV     (SP)+,R1        ;;RESTORE R1
 4407  017456  012600                          MOV     (SP)+,R0        ;;RESTORE R0
 4408  017460  000207                          RTS     PC
 4409  017462  000000                 $LSTAD:  .WORD   0               ;;CONTAINS THE LAST ADDRESS
 4410  017464  000000                 $LSTBK:  .WORD   0               ;;CONTAINS THE LAST BANK
```

```
4411                                                33410   ;TYPDBO
4412                                                33420   ;THIS ROUTINE CONVERTS A VIRTLAL ADDRESS TO PHYSICAL ADDRESS AND TYPES
4413                                                33430   ;OUT THE 6 DIGIT PHYSICAL ADDRESS. R2 CONTAINS VIRTUAL ADDRESS AT THE TIME
4414                                                33440   ;OF ENTRY.
4415                                                33450   ;TYPE OUT IS INHIBITED IF SW 13 IS SET.
4416                                                33460
4417  017466  032777  020000  161444                33470   TYPDBO: BIT     #SW13,@SWR      ;INHIBIT TYPECLT?
4418  017474  001042                                33480           BNE     2$              ;YES
4419  017476  010346                                33490           MOV     R3,-(SP)
4420  017500  010446                                33500           MOV     R4,-(SP)
4421  017502  010546                                33510           MOV     R5,-(SP)
4422                                                33520
4423  017504  010246                                33530           MOV     R2,-(SP)        ;PUSH VA ON STACK
4424  017506  004737  021620                        33540           JSR     PC,CROTLF       ;ROTATE BITS 15,14,13 INTO 2,1,0
4425  017512  012603                                33550           MOV     (SP)+,R3        ;FORM OFFSET TO BE USED
4426  017514  006303                                33560           ASL     R3              ;FOR KIPAR
4427                                                33570
4428  017516  016304  172340                        33580           MOV     KIPAR0(R3),R4   ;GET THE BASE PAGE ADDRESS FROM
4429                                                33590                                   ;KIPAR
4430  017522  005003                                33600           CLR     R3              ;ROTATE LEFT 6 TIMES (MULTIPLY
4431  017524  012705  177772                        33610           MOV     #-6,R5          ;BY 100 OCTAL) TO GET THE
4432  017530  006104                                33620   1$:     ROL     R4              ;BASE BUS ADDRESS (PHYSICAL)
4433  017532  005205                                33630           INC     R5              ;R3 CONTAINS MSB-2 BITS
4434  017534  001375                                33640           BNE     1$
4435                                                33650
4436                                                33660
4437  017536  010246                                33670           MOV     R2,-(SP)        ;STRIP OFF TOP 3 BITS FROM  VA &
4438                                                33680
4439  017540  042716  160000                        33690           BIC     #160000,(SP)    ;GET THE OFFSET INSIDE THE PAGE
4440  017544  062604                                33700           ADD     (SP)+,R4        ;FORM THE ENTIRE PHYSICAL
4441  017546  005503                                33710           ADC     R3              ;ADDRESS. R4 CONTAINS LOWER 16 BITS
4442                                                33720                                   ;R3 CONTAINS TOP 2 BITS
4443  017550  010437  001162                        33730           MOV     R4,$REG0        ;SAVE LOWER 16 BITS OF PA
4444  017554  010337  001164                        33740           MOV     R3,$REG1        ;SAVE TOP 2 BITS OF PA
4445  017560  012746  001162                        33750           MOV     #$REG0,-(SP)    ;PUSH POINTER TO PA ON STACK
4446  017564  004737  024112                        33760           JSR     PC,@#$OB20      ;CONVERT THE 18 BIT BINARY
4447                                                33770                                   ;ADDRESSS TO OCTAL ASICZ NUMBERS ON RETURN
4448                                                33780                                   ;POINTER TO THE FIRST ASCIZ CHARACTERS
4449                                                33790                                   ;IS ON STACK
4450  017570  004737  024442                        33800           JSR     PC,@#SUPRS      ;TYPE OUT THE OCTAL 6 DIGIT
4451                                                33810                                   ;PHYSICAL ADDRESS.
4452                                                33820
4453  017574  012605                                33830           MOV     (SP)+,R5
4454  017576  012604                                33840           MOV     (SP)+,R4
4455  017600  012603                                33850           MOV     (SP)+,R3
4456                                                33860
4457  017602  000207                                33870   2$:     RTS     PC
```

```
 4458                                                33890   ;CHKCS
 4459                                                33900   ;THIS ROUTINE CHECKS IF BIT 15 OF RKCS WAS SET. IF IT WAS RETURN IS MADE TO
 4460                                                33910   ;THE ERROR MESSAGE FOLLOWING THE JSP CALL. IF NOT, THE ERROR MADE TO SKIP
 4461                                                33920   ;OVER THE ERROR MESSAGE.
 4462                                                33930
 4463  017604  005777  162112                        33940   CHKCS:  TST     @RKCS           ;BIT 15 SET?
 4464  017610  100073                                33950           BPL     COMRET          ;NO
 4465  017612  004737  021462                        33960           JSR     PC,GT4RG        ;YES, GET RKCS, ER, DS, DA
 4466  017616  000207                                33970           RTS     PC              ;RETURN TO THE ERROR MESSAGE
 4467                                                33980
 4468                                                33990   ;CHKDA
 4469                                                34000   ;THIS ROUTINE CHECKS IF RKDA INCREMENTED CORRECTLY. IF NOT, RETURN IS MADE
 4470                                                34010   ;TO THE ERROR MESSAGE FOLLOWING THE JSR CALL. IF YES, RETURN IS MADE TO
 4471                                                34020   ;SKIP OVER THE ERROR MESSAGE.
 4472                                                34030   ;AT THE TIME OF ENTRY, R2 CONTAINS THE EXPECTED RKDA.
 4473                                                34040
 4474  017620  020277  162104                        34050   CHKDA:  CMP     R2,@RKDA        ;DID RKDA INCREMENT CORRECTLY?
 4475  017624  001465                                34060           BEQ     COMRET          ;YES
 4476  017626  010237  001162                        34070           MOV     R2,$REG0        ;GET EXPCTD RKDA
 4477  017632  017737  162072  001164                34080           MOV     @RKDA,$REG1     ;GET RKDA RECVD
 4478  017640  000207                                34090           RTS     PC              ;RETURN TO THE ERROR MESSAGE
 4479                                                34100
 4480                                                34110   ;CHKBA
 4481                                                34120   ;THIS ROUTINE CHECKS IF RKBA INCREMENTED CORRECTLY. IF NOT, RETURN IS MADE
 4482                                                34130   ;TO THE ERROR MESSAGE FOLLOWING THE JSR CALL. IF YES, RETURN IS MADE TO
 4483                                                34140   ;SKIP OVER THE ERROR MESSAGE.
 4484                                                34150   ;AT THE TIME OF ENTRY, R3 CONTAINS THE WORD COUNT (# OF WORDS TRANSFERRED)
 4485                                                34160   ;R4 CONTAINS THE BUS ADDRESS WHERE THE TRANSFER STARTED.
 4486                                                34170
 4487  017642  000241                                34180   CHKBA:  CLC
 4488  017644  006103                                34190           ROL     R3              ;FORM THE EXPCTD BUS ADDRESS
 4489  017646  060304                                34200           ADD     R3,R4
 4490  017650  000241                                34210           CLC
 4491  017652  006003                                34220           ROR     R3
 4492  017654  020477  162046                        34230           CMP     R4,@RKBA        ;DID RKBA INCREMENT CORRECTLY?
 4493  017660  001447                                34240           BEQ     COMRET          ;YES
 4494  017662  010437  001162                        34250           MOV     R4,$REG0        ;GET EXPCTD RKBA
 4495  017666  000207                                34260           RTS     PC              ;RETURN TO THE EROR MESAGE
 4496                                                34270
 4497  017670  017737  162032  001164                34280           MOV     @RKBA,$REG1     ;GET RKBA RECVD
 4498                                                34290   ;CHKMEX
 4499                                                34300   ;THIS ROUTINE CHECKS THAT RKBA OVERFLOWED AND MEX BIT WAS SETIN RKCS (BIT 4)
 4500                                                34310   ;IF RKBA OVERFLOWED CORRECTLY, THE RETURN ADDRESS IS ADJUSTED TO SKIP THE
 4501                                                34320   ;ERROR MESSAGE ON RETURN.
 4502                                                34330
 4503  017676  017746  162020                        34340   CHKMEX: MOV     @RKCS,-(SP)     ;GET RKCS
 4504  017702  042716  177717                        34350           BIC     #177717,(SP)    ;GET MEX BITS 4,5
 4505  017706  022726  000020                        34360           CMP     #BIT4,(SP)+     ;CHECK BIT 4 SET?
 4506  017712  001432                                34370           BEQ     COMRET          ;YES, OK
 4507  017714  004737  021462                        34380           JSR     PC,GT4RG        ;SAVE RKCS,ER,DS,DA
 4508  017720  000207                                34390           RTS     PC              ;RETURN
```

```
 4509                                                34410   ;CHKWC
 4510                                                34420   ;THIS ROUTINE CHECKS IF RKWC OVERFLOWED CORRECTLY AFTER A DATA TRANSFER
 4511                                                34430   ;IF IT DID NOT, RETURN IS MADE TO THE ERROR MESSAGE. IF IT DID, RETURN IS
 4512                                                34440   ;MADE TO SKIP OVER THE ERROR MESSAGE.
 4513                                                34450
 4514  017722  005777  161776                       34460   CHKWC:  TST     @RKWC           ;RKWC OVERFLOWED?
 4515  017726  001424                               34470           BEQ     COMRET          ;YES
 4516  017730  017737  161774  001162               34480           MOV     @RKDA,$REG0
 4517  017736  017737  161762  001164               34490           MOV     @RKWC,$REG1
 4518  017744  000207                               34500           RTS     PC              ;RETURN TO THE EROR MESAGE
 4519                                                34510
 4520                                                34520
 4521                                                34530
 4522                                                34540   ;CHKRWS
 4523                                                34550   ;THIS ROUTINE CHECKS IF R/W/S RDY BIT IN RKDS IS SET. IF IT IS NOT SET RETURN
 4524                                                34560   ;IS MADE TO THE ERROR MESSAGE FOLLOWING THE JSR CALL. IF IT IS, THE RETURN
 4525                                                34570   ;ADDRESS IS ADJUSTED TO SKIP OVER THE ERROR MESSAGE ON RERTURN.
 4526                                                34580
 4527  017746  032777  000100  161742               34590   CHKRWS: BIT     #RWS,@RKDS      ;RWS RDY SET?
 4528  017754  001011                               34600           BNE     COMRET          ;YES
 4529  017756  004737  021462                       34610           JSR     PC,GT4RG        ;GET RKCS, ER, DS, DA
 4530  017762  000207                               34620           RTS     PC              ;RETURN TO THE ERROR MESSAGE
 4531                                                34630
 4532                                                34640
 4533                                                34650   ;CHKCRDY
 4534                                                34660   ;THIS ROUTINE CHECKS IF CONTROL READY BIT IN RKCS IS SET. IF IT IS NOT,
 4535                                                34670   ;RETURN IS MADE TO THE ERROR MESSAGE FOLLOWING THE JSR CALL. IF IT IS,
 4536                                                34680   ;RETURN ADDRESS IS ADJUSTED TO SKIP OVER THE ERROR MESSAGE.
 4537                                                34690
 4538  017764  105777  161732                       34700   CHKCRDY: TSTB   @RKCS           ;CONTROL READY SET?
 4539  017770  100403                               34710           BMI     COMRET          ;YES
 4540  017772  004737  021462                       34720           JSR     PC,GT4RG        ;GET RKCS, ER, DS, DA
 4541  017776  000207                               34730           RTS     PC              ;RETURN TO THE EROR MESAGE
 4542                                                34740
 4543  020000  062716  000002                       34750   COMRET: ADD     #2,(SP)         ;ADJUST RETURN ADDRESS TO SKIP OVER MESAGE
 4544  020004  000207                               34760           RTS     PC
```

```
 4545                                   34780   ;THIS ROUTINE KEEPS A HISTORY OF THE COOMANDS THAT ARE BEING EXECUTED
 4546                                   34790   ;ON THE RK11. 'PRSFNC' CONTAINS INFORMATION ABOUT THE PRESENT COMMAND
 4547                                   34800   ;WHICH IS ABOUT TO BE INITIATED. 'PSTFNC' CONTAINS INFORMATION ABOUT THE
 4548                                   34810   ;COMMAND THAT WAS EXECUTED BEFORE THIS NEW ONE. THERE ARE MULTIPLE POINTS
 4549                                   34820   ;OF ENTRY DEPENDING ON THE TYPE OF COMMAND BEING PRESENTLY INITIATED:
 4550                                   34830
 4551                                   34840   ;DRCMND - ENTERED WHEN A DRIVE RESET IS BEING INITIATED. DRIVE # IS SAVED
 4552                                   34850   ;IN BITS 0-2 OF 'PRSCMND' AND BIT 8 IS SET.
 4553                                   34860
 4554                                   34870   ;CRCMND - ENTERED WHEN A CONTROL RESET IS BEING INITIATED. BIT 14 OF
 4555                                   34880   ;'PRSCMND' IS SET.
 4556                                   34890
 4557                                   34900   ;POSCMND - ENTERED WHEN A POSITIONING SEEK IS BEING INITIATED. BITS 0-2
 4558                                   34910   ;CONTAIN THE DRIVE NUMBER ON WHICH THE POSITIONING SEEK WAS DONE. ALSO
 4559                                   34920   ;BIT 7 IS SET.
 4560                                   34930
 4561                                   34940   ;IN ALL ABOVE CASES BIT 15 OF 'PRSCMND' IS SET.
 4562                                   34950
 4563                                   34960   ;FNCMND - ENTERED WHEN A COMMAND OTHER THAN ANY ONE OF THE ABOVE IS BEING
 4564                                   34970   ;INITIATED (EX: READ, WRITE, ETC).
 4565                                   34980   ;THE OFFSET TO THE COMMAND KEY (BASE=KEY) IS SAVED IN BITS 0-3 OF 'PSRCMND'.
 4566                                   34990
 4567                                   35000   ;IT SHOULD BE NOTED THAT CONTENTS OF 'PRSFNC' ARE PUSHED INTO 'PSTFNC'
 4568                                   35010   ;AND SAVED, BEFORE PUTTING INFO ABOUT THE PRESENT COMMAND IN 'PRSFNC'.
 4569                                   35020
 4570                                   35030   ;R0 CONTAINS ADDRESS OF THE COMMAND KEY, AT THE TIME OF ENTRY.
 4571                                   35040
 4572  020006  012737  100400  002212  35050   DRCMND: MOV     #BIT15+BIT8,QFNC
 4573  020014  017746  161710          35060           MOV     @RKDA,-(SP)             ;SAVE DRIVE #
 4574  020020  004737  021620          35070           JSR     PC,CRDTLF
 4575  020024  052637  002212          35080           BIS     (SP)+,QFNC
 4576  020030  000424                  35090           BR      P2
 4577                                   35100
 4578  020032  012737  140000  002212  35110   CRCMND: MOV     #BIT15+BIT14,QFNC
 4579  020040  000420                  35120           BR      P2
 4580                                   35130
 4581  020042  011037  002212          35140   POSCMND:        MOV     (R0),QFNC
 4582  020046  042737  177770  002212  35150           BIC     #177770,QFNC            ;GET DRIVE NO.
 4583  020054  052737  100200  002212  35160           BIS     #BIT15+BIT7,QFNC
 4584  020062  000407                  35170           BR      P2
 4585                                   35180
 4586  020064  005037  002212          35190   FNCMND: CLR     QFNC
 4587  020070  010046                  35200   P1:     MOV     R0,-(SP)
 4588  020072  162716  002006          35210           SUB     #KEY,(SP)
 4589  020076  052637  002212          35220           BIS     (SP)+,QFNC
 4590                                   35230
 4591  020102  013737  002162  002164  35240   P2:     MOV     PRSFNC,PSTFNC
 4592  020110  013737  002212  002162  35250           MOV     QFNC,PRSFNC
 4593  020116  000207                  35260           RTS     PC
```

```
 4594                                                35280   ;HISTRY
 4595                                                35290   ;THIS ROUTINE TYPES OUT INFORMATION ABOUT THE FUNCTION THAT WAS
 4596                                                35300   ;BEING PERFORMED ON THE RK AT THE TIME OF ERROR AND THE FUNCTION
 4597                                                35310   ;THAT WAS PERFORMED JUST BEFORE THAT FUNCTION (WHICH LED TO
 4598                                                35320   ;THE ERROR). THIS ROUTINE IS CALLED WHEN AN ERROR OCCURS AND SW 12
 4599                                                35330   ;IS SET.
 4600                                                35340
 4601  020120  010046                               35350   HISTRY: MOV     R0,-(SP)
 4602  020122  010146                               35360           MOV     R1,-(SP)
 4603                                                35370
 4604  020124  012700  002162                       35380           MOV     #PRSFNC,R0
 4605  020130  104400  020440                       35390           TYPE    ,MH1
 4606  020134  104400  020464                       35400           TYPE    ,MH3
 4607  020140  104400  020454                       35410           TYPE    ,MH2
 4608  020144  005710                               35420   6$:     TST     (R0)
 4609  020146  100053                               35430           BPL     3$              ;READ, READ CHECK, WRITE, WRITE CHECK, SEEK
 4610  020150  105710                               35440           TSTB    (R0)
 4611  020152  100427                               35450           BMI     2$              ;POSITIONING (SEEK)
 4612  020154  032710  040000                       35460           BIT     #BIT14,(R0)
 4613  020160  001014                               35470           BNE     1$              ;CONTROL RESET
 4614                                                35480
 4615  020162  104400  020170                               TYPE    ,65$            ;;TYPE ASCIZ STRING
 4616  020166  000410                               BR      64$             ;;GET OVER THE ASCIZ
 4617                                        ;;65$:  .ASCIZ  /RESET ON DRV /
 4618  020210                                64$:
 4619  020210  000425                       35500           BR      7$
 4620                                        35510
 4621  020212                               35520   1$:
 4622  020212  104400  020220                               TYPE    ,67$            ;;TYPE ASCIZ STRING
 4623  020216  000404                               BR      66$             ;;GET OVER THE ASCIZ
 4624                                        ;;67$:  .ASCIZ  /CRESET/
 4625  020230                                66$:
 4626  020230  000463                       35530           BR      4$
 4627                                        35540
 4628  020232                               35550   2$:
 4629  020232  104400  020240                               TYPE    ,69$            ;;TYPE ASCIZ STRING
 4630  020236  000412                               BR      68$             ;;GET OVER THE ASCIZ
 4631                                        ;;69$:  .ASCIZ  /POSITICNING DRIVE /
 4632  020264                                68$:
 4633  020264  011046                       35560   7$:     MOV     (R0),-(SP)
 4634  020266  042716  177770               35570           BIC     #177770,(SP)    ;TYPE DRIVE NO.
 4635  020272  104401                       35580           TYPOC
 4636  020274  000441                       35590           BR      4$
 4637                                        35600
 4638  020276  011001                       35610   3$:     MOV     (R0),R1
 4639  020300  016101  002532               35620           MOV     PCMND(R1),R1    ;GO TYPE OUT THE FUNCTION
 4640  020304  016104  000002               35630           MOV     2(R1),R4        ;BEING PERFORMED
 4641  020310  004737  021366               35640           JSR     PC,TYPFN
 4642  020314  104400  020322                               TYPE    ,71$            ;;TYPE ASCIZ STRING
 4643  020320  000403                               BR      70$             ;;GET OVER THE ASCIZ
 4644                                        ;;71$:  .ASCIZ  <15><12>/DA=/
 4645  020330                                70$:
 4646  020330  011146                       35660           MOV     (R1),-(SP)      ;TYPE OUT DISK ADDRESS
 4647  020332  104401                       35670           TYPOC
 4648  020334  104400  020342                               TYPE    ,73$            ;;TYPE ASCIZ STRING
 4649  020340  000403                               BR      72$             ;;GET OVER THE ASCIZ
```

```
                                                        ;;73$:  .ASCIZ  /  BA=/
                                                        72$:
020350                                                  
020350  016146  000006                      35690         MOV     6(R1),-(SP)
020354  104401                              35700         TYPOC
020356  104400  020364                                    TYPE    ,75$            ;;TYPE ASCIZ STRING
020362  000403                                          BR      74$             ;;GET OVER THE ASCIZ
                                                        ;;75$:  .ASCIZ  /  WC=/
020372                                                  74$:
020372  016146  000004                      35720         MOV     4(R1),-(SP)
020376  104401                              35730         TYPOC
                                            35740
020400  020027  002164                      35750  4$:    CMP     R0,#PSTFNC
020404  001410                              35760         BEQ     5$
020406  005720                              35770         TST     (R0)+
020410  104400  020440                      35780         TYPE    ,MH1
020414  104400  020467                      35790         TYPE    ,MH4
020420  104400  020454                      35800         TYPE    ,MH2
020424  000647                              35810         BR      6$
020426  104400  001213                      35820  5$:    TYPE    ,$CRLF
020432  012601                              35830         MOV     (SP)+,R1
020434  012600                              35840         MOV     (SP)+,R0
020436  000207                              35850         RTS     PC
020440  005015  052506  041516              35860  MH1:   .ASCIZ  <15><12>/FUNCTION /
020446  044524  047117  000040
020454  042440  051122  051117              35870  MH2:   .ASCIZ  / ERROR /
020462  000040
020464  052101     000                      35880  MH3:   .ASCIZ  /AT/
020467     120  044522  051117              35890  MH4:   .ASCIZ  /PRIOR TO/
020474  052040  000117
                                            35900         .EVEN
```

```
 4680                                          35920   ;STATSTC
 4681                                          35930   ;AT THE TIME OF ENTRY R1 CONTAINS THE DRIVE NUMBER FOR WHICH THE STATISTIC
 4682                                          35940   ;IS TO BE OBTAINED.  R5 CONTAINS THE POINTER TO THE PARAMETER TABLE FOR
 4683                                          35950   ;THE COMMAND EXECUTED ON THE ABOVE DRIVE. R4 CONTAINS THE FUNCTION CODE
 4684                                          35960   ;(WRITE, READ, ETC) FOR WHICH STATISTICS  ARE TO BE TAKEN.
 4685                                          35970
 4686  020500  010046                          35980   STATSTC:MOV     R0,-(SP)        ;PUSH R0, R2, R3 ONTO THE
 4687  020502  010246                          35990           MOV     R2,-(SP)        ;STACK
 4688  020504  010346                          36000           MOV     R3,-(SP)
 4689                                          36010
 4690  020506  005002                          36020           CLR     R2
 4691  020510  005701                          36030           TST     R1              ;DRIVE 0?
 4692  020512  001404                          36040           BEQ     2$              ;FORM THE OFFSET FOR THE
 4693  020514  062702  000004                  36050   1$:     ADD     #4,R2           ;'WORDS XFERRED COUNTS'-
 4694  020520  005301                          36060           DEC     R1              ;NWRTL, NRDL
 4695  020522  001374                          36070           BNE     1$
 4696  020524  016500  000004                  36080   2$:     MOV     4(R5),R0        ;GET WORD COUNT (RKWC) FROM
 4697  020530  005400                          36090           NEG     R0              ;THE PARAMETER TABLE
 4698                                          36100
 4699  020532  005777  161164                  36110           TST     @RKCS           ;ANY ERROR DURING THE XFER?
 4700  020536  100004                          36120           BPL     3$
 4701                                          36130                                   ;YES,
 4702  020540  017703  161160                  36140           MOV     @RKWC,R3        ;GET THE # OF WORDS THAT
 4703  020544  005403                          36150           NEG     R3              ;WERE ACTUALLY X-FERRED
 4704  020546  160300                          36160           SUB     R3,R0
 4705                                          36170
 4706  020550  022704  000002                  36180   3$:     CMP     #2,R4           ;WRITE FUNCTION?
 4707  020554  001005                          36190           BNE     5$
 4708                                          36200                                   ;YES, ADD THE # OF WORDS
 4709  020556  060062  002432                  36210   4$:     ADD     R0,NWRTL(R2)    ;XFERRED (WRITE)
 4710  020562  005562  002434                  36220           ADC     NWRTH(R2)       ;NOTE IT'S 2-WORD COUNT LO, HI
 4711  020566  000404                          36230           BR      6$
 4712                                          36240
 4713  020570  060062  002472                  36250   5$:     ADD     R0,NRDL(R2)     ;ADD THE # OF WORDS READ
 4714                                          36260                                   ;NOTE THAT WRT CHK,
 4715                                          36270                                   ;READ CHK ARE ALSO CONS-
 4716                                          36280                                   ;IDERED TO BE 'READ'
 4717  020574  005562  002474                  36290           ADC     NRDH(R2)        ;CARRY OVER TO THE HI WORD
 4718                                          36300
 4719  020600  012603                          36310   6$:     MOV     (SP)+,R3        ;POP R3,R2,R0 FROM THE STACK
 4720  020602  012602                          36320           MOV     (SP)+,R2
 4721  020604  012600                          36330           MOV     (SP)+,R0
 4722  020606  000207                          36340           RTS     PC
```

```
4723
4724
4725
4726  020610  004737  026272          36360  ;REPSTAT
4727  020614  104400  003214          36370  ;THIS ROUTINE REPORTS ERROR STATISTICS AND DATA-TRANSFER STATISTICS.
4728  020620  013700  001764          36380
4729  020624  012701  001754          36390  REPSTA: JSR     PC,TIMTYP               ;TYPE CURRENT TIME
4730                                  36400          TYPE    ,MSG26
4731  020630  104400  001213          36410          MOV     DRVPRS,R0
4732  020634  112102                  36420          MOV     #PDR,R1
4733  020636  042702  177770          36430
4734  020642  010246                  36440  1$:     TYPE    ,$CRLF
4735  020644  104402                  36450          MOVB    (R1)+,R2
4736  020646     003                  36460          BIC     #177770,R2
4737  020647     000                  36470          MOV     R2,-(SP)
4738  020650  104400  003454          36480          TYPOS
4739                                  36490          .BYTE   3
4740  020654  005004                  36500          .BYTE   0
4741  020656  010203                  36510          TYPE    ,BLNKS3
4742  020660  001404                  36520
4743  020662  062704  000004          36530          CLR     R4
4744  020666  005303                  36540          MOV     R2,R3
4745  020670  001374                  36550          BEQ     3$
4746                                  36560  2$:     ADD     #4,R4
4747  020672  104400  003456          36570          DEC     R3
4748  020676  010446                  36580          BNE     2$
4749  020700  062716  002432          36590
4750  020704  004737  024232          36600  3$:     TYPE    ,BLNKS1
4751  020710  004737  024442          36610          MOV     R4,-(SP)
4752  020714  104400  003456          36620          ADD     #NWRTL,(SP)
4753  020720  010446                  36630          JSR     PC,$DB2D
4754  020722  062716  002472          36640          JSR     PC,SUPRS
4755  020726  004737  024232          36650          TYPE    ,BLNKS1
4756  020732  004737  024442          36660          MOV     R4,-(SP)
4757                                  36670          ADD     #NRDL,(SP)
4758  020736  006302                  36680          JSR     PC,$DB2D
4759                                  36690          JSR     PC,SUPRS
4760  020740  104400  003456          36700
4761  020744  016246  002352          36710          ASL     R2
4762  020750  104404                  36720
4763                                  36730          TYPE    ,BLNKS1
4764  020752  104400  003456          36740          MOV     CSECN(R2),-(SP)
4765  020756  016246  002332          36750          TYPDS
4766  020762  104404                  36760
4767                                  36770          TYPE    ,BLNKS1
4768  020754  104400  003456          36780          MOV     WCECN(R2),-(SP)
4769  020770  016246  002412          36790          TYPDS
4770  020774  042716  100000          36800
4771  021000  104404                  36810          TYPE    ,BLNKS1
4772                                  36820          MOV     DATER(R2),-(SP)
4773  021002  104400  003456          36830          BIC     #100000,(SP)    ;DONT TYPE A NEGATIVE NO.
4774  021006  016246  002262          36840          TYPDS
4775  021012  104404                  36850
4776                                  36860          TYPE    ,BLNKS1
4777  021014  104400  003456          36870          MOV     HECN(R2),-(SP)
4778  021020  016246  002302          36880          TYPDS
                                      36890
                                      36900          TYPE    ,BLNKS1
                                      36910          MOV     SKECN(R2),-(SP)
```

```
 4779  021024  104404                 36920          TYPDS
 4780                                  36930
 4781                                  36940
 4782  021026  104400  003456         36950          TYPE    ,BLNKS1
 4783  021032  016246  002372         36960          MOV     ABORT(R2),-(SP)
 4784  021036  104404                 36970          TYPDS
 4785                                  36980
 4786  021040  006202                 36990          ASR     R2
 4787  021042  104400  003456         37000          TYPE    ,BLNKS1
 4788  021046  116246  002322         37010          MOVB    SINCN(R2),-(SP)
 4789  021052  104404                 37020          TYPDS
 4790                                  37030
 4791  021054  005300                 37040          DEC     R0
 4792  021056  001264                 37050          BNE     1$
 4793  021060  104400  001213         37060          TYPE    ,$CRLF
 4794  021064  000207                 37070          RTS     PC
```

```
 4795                                          37090   ;STATUS
 4796                                          37100   ;THIS ROUTINE IS NORMALLY ENTERED WHEN THE PROGRAM IS WAITING FOR THE
 4797                                          37110   ;CONTROLLER TO FINISH WHAT IT IS DOING. THERE ARE TWO DOUBLE PRECISION
 4798                                          37120   ;COUNTS KEPT IN THIS ROUTINE.
 4799                                          37130   ;CICNT,CICNT1
 4800                                          37140   ;THIS COUNT KEEPS TRACK OF HOW LONG THE CONTROLLER HAS BEEN BUSY AFTER
 4801                                          37150   ;A COMMAND WAS INITIATED. THE CONTROLLER SHOULD FINISH WHATEVER IT IS DOING
 4802                                          37160   ;BEFORE THIS COUNT EXPIRES. IF IT DOES NOT, THERE IS AN ERROR CONDITION AND
 4803                                          37170   ;IT IS SO REPORTED.
 4804                                          37180
 4805                                          37190   ;@SCNT
 4806                                          37200   ;THIS COUNT IS INITIALIZED EVERY TIME 8 COMMANDS ARE GENERATED. THE COUNT
 4807                                          37210   ;IS INCREMENTED EVERY TIME THIS ROUTINE IS ENTERED. ALL THE 8 COMMANDS
 4808                                          37220   ;SHOULD BE DONE BEFORE THIS COUNT EXPIRES. IF THEY DO NOT AN ERROR CONDITION
 4809                                          37230   ;IS REPORTED. THIS COUNT HAS BEEN KEPT PRIMARILY TO INSURE THAT THE PROGRAM
 4810                                          37240   ;DOES NOT GET CAUGHT IN AN INDEFINITE LOOP, BECAUSE OF AN ERROR CONDITION.
 4811                                          37250
 4812  021066  005046                        37260   STATUS: CLR     -(SP)           ;DROP PRIORITY AND WAIT FOR INT
 4813  021070  012746  021076                37270           MOV     #1$,-(SP)       ;RETURN FOR RTI
 4814  021074  000002                        37280           RTI
 4815                                          37290                                   ;NOTE THAT THE INTERRUPTS ARE ALLOWED ONLY
 4816                                          37300                                   ;AT CERTAIN PLACES IN THE PROGRAM, BECAUSE
 4817                                          37310                                   ;IT MAKES TROUBLESHOOTING OF FALIURES EASY.
 4818                                          37320                                   ;OTHER PLACES WHERE INTERRUPTS ARE ALLOWED
 4819                                          37330                                   ;TO TAKE PLACE:
 4820                                          37340                                   ;'CHFAFN',  FIRST INTERRUPT AFTER ISSUING
 4821                                          37350                                   ;A SEEK FUNCTION.
 4822                                          37360
 4823  021076  005237  002160                37370   1$:     INC     @SCNT
 4824  021102  001456                        37380           BEQ     QEROR
 4825  021104  105777  160612                37390           TSTB    @RKCS
 4826  021110  100514                        37400           BMI     CNOBSY
 4827                                          37410
 4828  021112  005037  002166                37420           CLR     CICNT
 4829  021116  012737  177761  002170        37430           MOV     #-17,CICNT1
 4830                                          37440
 4831  021124  105737  002234                37450   CBSY:   TSTB    INTFLG          ;FOR A NON-SEEK COMAND:
 4832                                          37460                                   ;INTFLG- BIT 7 IS SET, BITS 0-3 CONTAIN
 4833                                          37470                                   ;OFFSET TO THE COMMAND KEY (FROM KEY),
 4834                                          37480                                   ;FOR WHICH THIS INTERUPT IS EXPCTD.
 4835                                          37490                                   ;WHEN THE INTERUPT OCCURS & 'INTHND' IS
 4836                                          37500                                   ;ENTERED 'INTFLG' IS CLEARED.
 4837  021130  001507                        37510           BEQ     SEXIT
 4838  021132  005237  002166                37520           INC     CICNT
 4839  021136  001372                        37530           BNE     CBSY
 4840  021140  005237  002170                37540           INC     CICNT1
 4841  021144  001367                        37550           BNE     CBSY
 4842                                          37560
 4843                                          37570                                   ;TIMED OUT WHILE WAITING FOR THE INTRUPT.
 4844                                          37580                                   ;ONE OF THE COMMANDS DID NOT INTERRUPT
 4845  021146  113700  002234                37590   NIEROR: MOVB    INTFLG,R0
 4846  021152  042700  177760                37600           BIC     #177760,R0
 4847  021156  010003                        37610           MOV     R0,R3
 4848  021160  062700  002006                37620           ADD     #KEY,R0
 4849  021164  011037  001172                37630           MOV     (R0),$REG4
 4850  021170  042737  177770  001172        37640           BIC     #177770,$REG4
```

```
 4851  021176  013737  001172  001750  37650           MOV     $REG4,SRDRV     ;GET DRIVE #, FOR TYPING SERIAL #
 4852                                  37660
 4853  021204  104420  002747          37670           TYPMSG  ,MSG15          ;PRINT 'DRVE # DIDN'T INTRUPT AFTER'
 4854  021210  016305  002532          37680           MOV     PCMND(R3),R5
 4855  021214  016504  000002          37690           MOV     2(R5),R4
 4856  021220  004737  021366          37700           JSR     PC,TYPFN
 4857  021224  004737  021462          37710           JSR     PC,GT4RG
 4858  021230  104025                  37720           ERROR   25              ;COMMAND TYPED OUT IN EROR MESAGE DID
 4859                                  37730                                   ;NOT INTERUPT ON COMPLETION.
 4860  021232  052710  104000          37740           BIS     #BIT15+BIT11,(R0)       ;INDICATE THAT FUNCTION IS ABCRTED
 4861  021236  000444                  37750           BR      SEXIT
 4862                                  37760
 4863                                  37770
 4864  021240  005037  002160          37780   QEROR:  CLR     QSCNT           ;REESTABLISH COUNT
 4865  021244  004737  021462          37790           JSR     PC,GT4RG
 4866  021250  104026                  37800           ERROR   26              ;ALL 8 COMMANDS SHOULD BE DONE BY NOW, TIMED
 4867                                  37810                                   ;OUT. THE PROGRAM IS WAITING FOR ONE OF THE
 4868                                  37820                                   ;COMMANDS IN THE Q TO BE FINISHED AND THIS
 4869                                  37830                                   ;DID NOT HAPPEN OR FOR SOME OTHER REASON THE
 4870                                  37840                                   ;'FINISHED' FLAG (BIT 15) OF ONE OF THE 8
 4871                                  37850                                   ;COMMAND KEYS WAS NOT SET. VARIOUS FLAGS 'POS'(-
 4872                                  37860                                   ;'BUSY'(-7), 'KEY'(-8) CONTAIN INFORMATION
 4873                                  37870                                   ;ABOUT THE STATUS OF THE SYSTEM.
 4874                                  37880
 4875  021252  032777  020000  157660  37890           BIT     #SW13,@SWR      ;INHIBIT TYPEOUT?
 4876  021260  001024                  37900           BNE     2$              ;YES
 4877  021262  104400  003007          37910           TYPE,   MSG16
 4878  021266  012700  002006          37920           MOV     #KEY,R0
 4879  021272  012701  002126          37930           MOV     #BUSY,R1
 4880  021276  012702  177770          37940           MOV     #-10,R2
 4881  021302  104400  001213          37950   1$:     TYPE    ,$CRLF
 4882  021306  012046                  37960           MOV     (R0)+,-(SP)     ;TYPE OUT CONTENTS OF ALL KEYS
 4883  021310  104401                  37970           TYPOC                   ;KEY-KEY8
 4884  021312  104400  003454          37980           TYPE    ,BLNKS3
 4885  021316  005046                  37990           CLR     -(SP)
 4886  021320  112116                  38000           MOVB    (R1)+,-(SP)     ;TYPE OUT CONTENTS OF ALL BUSY FLAGS
 4887  021322  104402                  38010           TYPOS                   ;BUSY-BUSY7
 4888  021324     003                  38020           .BYTE   3
 4889  021325     000                  38030           .BYTE   0
 4890  021326  005202                  38040           INC     R2              ;DONE?
 4891  021330  001364                  38050           BNE     1$              ;NO
 4892                                  38060
 4893  021332  004737  015454          38070   2$:     JSR     PC,CLRERR       ;MAKE SURE THERE IS NO HEAD MOVEMENT ON
 4894                                  38080                                   ;ANY DRIVE & THEN DO CONTROL RESET
 4895  021336  000137  010352          38090           JMP     BEGNEX          ;GO, BAK AND CONTINUE
 4896                                  38100
 4897  021342  005004                  38110   CNOBSY: CLR     R4
 4898  021344  005204                  38120           INC     R4
 4899  021346  001376                  38130           BNE     .-2
 4900  021350  013746  001744          38140   SEXIT:  MOV     PPRLVL,-(SP)
 4901  021354  012746  021362          38150           MOV     #RTIPC7,-(SP)   ;RETURN FOR RTI ***********
 4902  021360  000002                  38160           RTI
 4903                                  38170
 4904  021362  000137  010372          38180   RTIPC7: JMP     QMNGER
```

```
                                                38200   ;TYPFN
                                                38210   ;ROUTINE TO TYPE OUT THE FUNCTION (READ,WRITE,ETC) ;R4 CONTAINS THE
                                                38220   ;FUNCTION CODE AT THE TIME OF ENTRY.
                                                38230   ;SW 13, IF SET INHIBITS TYPEOUT.
                                                38240
4910  021366  032777  020000  157544            38250   TYPFN:  BIT     #SW13,@SWR      ;INHIBIT TYPEOUT?
4911  021374  001031                            38260           BNE     5$              ;YES
4912  021376  020427  000002                    38270           CMP     R4,#2           ;WRITE?
4913  021402  001002                            38280           BNE     1$
4914  021404  104400  002635                    38290           TYPE    ,MSG6
4915  021410  022704  000004                    38300   1$:     CMP     #4,R4           ;READ?
4916  021414  001002                            38310           BNE     2$
4917  021416  104400  002643                    38320           TYPE    ,MSG7
4918  021422  022704  000012                    38330   2$:     CMP     #12,R4          ;READ CHECK?
4919  021426  001002                            38340           BNE     3$
4920  021430  104400  002660                    38350           TYPE    ,MSG9
4921  021434  022704  000006                    38360   3$:     CMP     #6,R4           ;WRITE CHECK?
4922  021440  001002                            38370           BNE     4$
4923  021442  104400  002650                    38380           TYPE    ,MSG8
4924  021446  022704  000010                    38390   4$:     CMP     #10,R4          ;SEEK?
4925  021452  001002                            38400           BNE     5$
4926  021454  104400  002703                    38410           TYPE    ,MSG11
4927  021460  000207                            38420   5$:     RTS     PC
```

```
 4928                                                38440   ;GT4RG
 4929                                                38450   ;GET CONTENTS OF RKCS, RKER, RKDS, RKDAA
 4930                                                38460
 4931  021462  017737  160242  001170               38470   GT4RG:  MOV     @RKDA,$REG3
 4932  021470  017737  160226  001162               38480   GT3RG:  MOV     @RKCS,$REG0
 4933  021476  017737  160216  001164               38490           MOV     @RKER,$REG1
 4934  021504  017737  160206  001166               38500           MOV     @RKDS,$REG2
 4935  021512  000207                               38510           RTS     PC
 4936                                                38520
 4937                                                38530
 4938                                                38540   ;GETINF
 4939                                                38550   ;THIS ROUTINE GETS CONTENTS OF RKCE, RKER, RKDS. THEN IT BREAKS DOWN THE
 4940                                                38560   ;CONTENTS OF RKDA INTO ITS COMPONENT: CYLINDER, SECTOR, SURFACE AND DRIVE
 4941                                                38570   ;NUMBER.
 4942                                                38580
 4943  021514  004737  021470                       38590   GETINF: JSR     PC,GT3RG
 4944  021520  010046                               38600           MOV     R0,-(SP)
 4945  021522  010146                               38610           MOV     R1,-(SP)
 4946  021524  010246                               38620           MOV     R2,-(SP)
 4947  021526  012700  001200                       38630           MOV     #$REG6+2,R0
 4948  021532  017701  160172                       38640           MOV     @RKDA,R1
 4949  021536  010102                               38650           MOV     R1,R2
 4950  021540  042702  177760                       38660           BIC     #177760,R2
 4951  021544  010240                               38670           MOV     R2,-(R0)
 4952  021546  006201                               38680           ASR     R1
 4953  021550  006201                               38690           ASR     R1
 4954  021552  006201                               38700           ASR     R1
 4955  021554  006201                               38710           ASR     R1
 4956  021556  010102                               38720           MOV     R1,R2
 4957  021560  042702  177776                       38730           BIC     #177776,R2
 4958  021564  010240                               38740           MOV     R2,-(R0)
 4959  021566  006201                               38750           ASR     R1
 4960  021570  010102                               38760           MOV     R1,R2
 4961  021572  042702  177400                       38770           BIC     #177400,R2
 4962  021576  010240                               38780           MOV     R2,-(R0)
 4963  021600  000301                               38790           SWAB    R1
 4964  021602  042701  177770                       38800           BIC     #177770,R1
 4965  021606  010140                               38810           MOV     R1,-(R0)
 4966  021610  012602                               38820           MOV     (SP)+,R2
 4967  021612  012601                               38830           MOV     (SP)+,R1
 4968  021614  012600                               38840           MOV     (SP)+,R0
 4969  021616  000207                               38850           RTS     PC
```

```
 4970                                           38870  ;CROTLF
 4971                                           38880  ;CALL:  MOV     #NO,-(SP)       ;PUSH NO. TO BE ROTATED ON STACK
 4972                                           38890  ;       JSR     PC,CROTLF
 4973                                           38900  ;THIS ROUTINE ROTATES BITS 15, 14, 13 OF A WORD INTO BITS 2, 1, 0. THE
 4974                                           38910  ;REST OF THE BITS OF THE ROTATED WORD ARE CLEARED.
 4975                                           38920
 4976                                           38930
 4977  021620  042766  017777  000002          38940  CROTLF: BIC     #17777,2(SP)
 4978  021626  000241                           38950          CLC
 4979  021630  006166  000002                  38960          ROL     2(SP)
 4980  021634  006166  000002                  38970          ROL     2(SP)
 4981  021640  006166  000002                  38980          ROL     2(SP)
 4982  021644  006166  000002                  38990          ROL     2(SP)
 4983  021650  000207                           39000          RTS     PC
 4984                                           39010
 4985                                           39020
 4986                                           39030  ;RG4SDRV
 4987                                           39040  ;CALL:  JSR     PC,RG4SDRV
 4988                                           39050  ;THIS ROUTINE GETS THE CONTENTS OF RKDS, RKER, RKCS, RKDA. THEN
 4989                                           39060  ;IT SAVES THE DRIVE NUMBER FROM RKDA IN 'SRDRV'.
 4990  021652  004737  021462                   39070  RG4SDR: JSR     PC,GT4RG        ;GET RKCS, ER, DS, DA
 4991                                           39080
 4992                                           39090
 4993                                           39100  ;GTSDRV
 4994                                           39110  ;CALL:  JSR     PC,GTSDRV
 4995                                           39120  ;THIS ROUTINE EXTRACTS THE DRIVE # FROM RKDA (BITS 15,14,13) AND SAVES
 4996                                           39130  ;IT IN "SRDRV" (BITS 0,1,2)
 4997                                           39140
 4998  021656  017746  160046                   39150  GTSDRV: MOV     @RKDA,-(SP)     ;GET BITS 15,14,13 FROM RKDA
 4999  021662  004737  021620                   39160          JSR     PC,CROTLF
 5000  021666  012637  001750                   39170          MOV     (SP)+,SRDRV     ;SAVE THE DRIVE #
 5001  021672  000207                           39180          RTS     PC
```

```
                                                39200           .SBTTL  DRV.RESET - DRIVE RESET POUTINE
                                                39210   ;DRV.RESET - DRIVE RESET ROUTINE
                                                39220   ;IF R/W/S RDY DOES NOT SET WITHIN A CERTAIN TIME OF DOING DRIVE RESET
                                                39230   ;AN ERROR IS REPORTED.
                                                39240
 5007  021674  005037  022004                  39250   DR.RST: CLR     TIMOUT
 5008  021700  013777  002202  160022          39260           MOV     @DRV,@RKDA
 5009  021706  012777  000015  160006          39270           MOV     #15,@RKCS
 5010  021714  104416                          39280           CON.RDY
 5011  021716  032777  000100  157772          39290   1$:     BIT     #100,@RKDS      ;DID R/W/S RDY SET?
 5012  021724  001026                          39300           BNE     2$              ;YES
 5013  021726  012746  177760                  39310           MOV     #-20,-(SP)      ;NO, WAIT FOR R/W/S
 5014  021732  005216                          39320           INC     (SP)
 5015  021734  001376                          39330           BNE     .-2
 5016  021736  005726                          39340           TST     (SP)+
 5017  021740  005237  022004                  39350           INC     TIMOUT
 5018  021744  001364                          39360           BNE     1$
 5019  021746  032777  020000  157164          39370           BIT     #SW13,@SWR      ;INHIBIT TYPEOUT?
 5020  021754  001012                          39380           BNE     2$              ;YES
 5021  021756  104400  001213                  39390           TYPE    ,$CRLF          ;TIMED OUT, R/W/S RDY DID NOT SET
 5022  021762  104400  027340                  39400           TYPE    ,EMH            ;REPORT ERROR
 5023  021766  104400  002710                  39410           TYPE    ,MSG12
 5024  021772  011646                          39420           MOV     (SP),-(SP)
 5025  021774  162716  000002                  39430           SUB     #2,(SP)
 5026  022000  104401                          39440           TYPOC
 5027  022002  000002                          39450   2$:     RTI
 5028  022004  000000                          39460   TIMOUT: 0
```

```
 5029                                          39480          .SBTTL  CON.RESET - CONTROL RESET ROUTINE
 5030                                          39490  ;CON.RESET
 5031                                          39500  ;CONTROL RESET ROUTINE
 5032                                          39510  ;CON.RDY
 5033                                          39520  ;CONTROL READY ROUTINE
 5034                                          39530
 5035  022006  012777  000001  157736         39540  CN.RST: MOV     #1,@RKCS
 5036  022014  005037  002172                 39550  CN.RDY: CLR     TIMER
 5037  022020  105777  157676                 39560  1$:     TSTB    @RKCS           ;DID CONTROL RDY SET?
 5038  022024  100451                         39570          BMI     2$              ;YES
 5039  022026  012746  177750                 39580          MOV     #-30.,-(SP)     ;WAIT FOR CNTRL RDY
 5040  022032  005216                         39590          INC     (SP)
 5041  022034  001376                         39600          BNE     .-2
 5042  022036  005726                         39610          TST     (SP)+
 5043  022040  005237  002172                 39620          INC     TIMER
 5044  022044  001365                         39630          BNE     1$
 5045  022046  032777  020000  157064         39640          BIT     #SW13,@SWR      ;INHIBIT TYPEOUT?
 5046  022054  001035                         39650          BNE     2$              ;YES
 5047  022056  104400  002710                 39660          TYPE    ,MSG12          ;CNTRL RDY DID NOT SET, REPORT ERROR
 5048  022062  011646                         39670          MOV     (SP),-(SP)
 5049  022064  162716  000002                 39680          SUB     #2,(SP)
 5050  022070  104401                         39690          TYPOC
 5051  022072  104400  022100                                 TYPE    ,65$            ;;TYPE ASCIZ STRING
 5052  022076  000421                                        BR      64$             ;;GET OVER THE ASCIZ
 5053                                        ;;65$:  .ASCIZ  <15><12>/CONTROLLER NOT READY -   RKCS=/
 5054  022142                                 64$:
 5055  022142  017746  157554                 39710          MOV     @RKCS,-(SP)
 5056  022146  104401                         39720          TYPOC
 5057  022150  000002                         39730  2$:     RTI
```

```
5058                                    39750           .SBTTL  TYPMSG - TYPE MESSAGE ROUTINE (SW13)
5059                                    39760   ;TYPMSG
5060                                    39770   ;THIS ROUTINE IS USED FOR MESSAGE TYPEOUTS. IF SW 13 IS SET THE TYPEOUT
5061                                    39780   ;IS SKIPPED.
5062                                    39790   ;CALL:  TYPMSG
5063                                    39800   ;       POINTER                 ;POINTER TO THE ASCII MESSAGE STRING
5064                                    39810
5065  022152  032777  020000  156760    39820   TY.MSG: BIT     #SW13,@SWR      ;INHIBIT TYPEOUT?
5066  022160  001005                    39830           BNE     2$              ;YES
5067  022162  017637  000000  022172    39840           MOV     @(SP),1$        ;GET POINTER TO ASCII STRING
5068  022170  104400                    39850           TYPE
5069  022172  000000                    39860   1$:     .WORD   0
5070                                    39870
5071  022174  062716  000002            39880   2$:     ADD     #2,(SP)         ;ADJUST RETURN ADDRESS TO SKIP OVER POINTER
5072  022200  000002                    39890           RTI
```

```
                                                39910           .SBTTL  KWSRVE - KW11L CLOCK SERVICE ROUTINE
                                                39920   ;THIS ROUTINE SERVICES THE INTERRUPT FROM THE KW11L LINE CLOCK
                                                39930   ;AND KEEPS TRACK OF ELAPSED TIME.
                                                39940   ;KWCOUNT- CONTAINS CYCLES (PER SECOND) (2'S COMPLEMENT)
                                                39950   ;KWSEC- CONTAINS SECONDS (2'S COMPLEMENT)
                                                39960   ;KWMIN- CONTAINS MINUTES (2'S COMPLEMENT)
                                                39970   ;KWHR- CONTAINS HOURS (2'S COMPLEMENT)
                                                39980
5081  022202  005237  002260                    39990   KWSRVE: INC     KWCOUNT         ;COUNT 60 CPS
5082  022206  001401                            40000           BEQ     1$              ;OVERFLOWED?
5083  022210  000002                            40010           RTI
5084  022212  012737  177704  002260            40020   1$:     MOV     #-60.,KWCOUNT   ;RESET 60 CPS COUNT
5085  022220  005237  002256                    40030           INC     KWSEC           ;COUNT SECONDS
5086  022224  001401                            40040           BEQ     2$              ;OVERFLOWED?
5087  022226  000002                            40050           RTI                     ;RETURN
5088  022230  012737  177704  002256            40060   2$:     MOV     #-60.,KWSEC     ;RESET "SECONDS" COUNT
5089  022236  005237  002254                    40070           INC     KWMIN           ;COUNT MINUTES
5090  022242  001005                            40080           BNE     3$              ;OVERFLOWED?
5091  022244  012737  177704  00225-            40090           MOV     #-60.,KWMIN     ;RESET "MINUTES" COUNT
5092  022252  005237  002252                    40100           INC     KWHR            ;COUNT HOURS
5093                                            40110
5094  022256  000002                            40120   3$:     RTI                     ;RETURN
5095                                            40130
5096                                            40140
5097                                            40150   ;WATIME
5098                                            40160   ;ROUTINE PROVIDES SOME WAITING TIME.
5099  022260  005046                            40170   WATIME: CLR     -(SP)
5100  022262  011616                            40180   1$:     MOV     (SP),(SP)               ;WASTE SOME TIME
5101  022264  011616                            40190           MOV     (SP),(SP)
5102  022266  011616                            40200           MOV     (SP),(SP)
5103  022270  011616                            40210           MOV     (SP),(SP)
5104  022272  062716  000001                    40220           ADD     #1,(SP)
5105  022276  001371                            40230           BNE     1$
5106  022300  005026                            40240           CLR     (SP)+
5107  022302  000207                            40250           RTS     PC
```

```
 5108                                   40270   ;CHPDRS
 5109                                   40280   ;THIS ROUTINE CHECKS IF THERE ANY DRIVES PRESENT (ON LINE).  IF THERE
 5110                                   40290   ;ARE, A RETURN IS MADE. IF THERE ARE NONE PRESENT, A MESSAGE IS PRINTED OUT.
 5111                                   40300   ;THE STACK POINTER IS RE-INITIATED TO 1100 AND CONTROL IS TRANSFERRED
 5112                                   40310   ;TO THE END OF PASS ROUTINE, $EOP. BEFORE PASSING CONTROL TO $EOP, SOME
 5113                                   40320   ;TIME IS KILLED (WATIME).  THIS IS DONE TO KEEP THE NUMBER OF MESSAGES
 5114                                   40330   ;(END OF PASS #X) TO A SMALL AMOUNT.
 5115                                   40340
 5116  022304  005737  001764           40350   CHDPRS: TST     DRVPRS          ;ANY DRIVES PRESENT?
 5117  022310  001401                   40360           BEQ     1$              ;NO
 5118  022312  000207                   40370           RTS     PC              ;YES, EXIT
 5119  022314  104400  002727           40380   1$:     TYPE    ,MSG14          ;NO, GIVE A MESSAGE
 5120  022320  004737  022260           40390           JSR     PC,WATIME       ;KILL SOME TIME
 5121  022324  012706  001100           40400           MOV     #STACK,SP       ;REINITIALIZE STACK
 5122  022330  000400                   40410           BR      $EOP            ;GO TO END OF PASS ROUTINE
 5123                                   40420
```

```
                                        .SBTTL  END OF PASS ROUTINE

                                        ;;**********************************************************************
                                        ;*INCREMENT THE PASS NUMBER ($PASS)
                                        ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO QMNGER

022332                                  $EOP:
022332  000004                                SCOPE
022334  005037  001102                      CLR     $TSTNM          ;;ZERO THE TEST NUMBER
022340  005237  001100                      INC     $PASS           ;;INCREMENT THE PASS NUMBER
022344  042737  100000  001100              BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
022352  005327                              DEC     (PC)+           ;;LOOP?
022354  000001                  $EOPCT:     .WORD   1
022356  003013                              BGT     $DOAGN          ;;YES
022360  012737                              MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
022362  000001                  $ENDCT:     .WORD   1
022364  022354                              $EOPCT
022366  013700  000042          $GET42:     MOV     @#42,R0         ;;GET MONITOR ADDRESS
022372  001405                              BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
022374  000005                              RESET                   ;;CLEAR THE WORLD
022376  004710                  $ENDAD:     JSR     PC,(R0)         ;;GO TO MONITOR
022400  000240                              NOP                     ;;SAVE ROOM
022402  000240                              NOP                     ;;FOR
022404  000240                              NOP                     ;;ACT11
022406                          $DOAGN:
022406  000137                              JMP     @(PC)+          ;;RETURN
022410  010372                  $RTNAD:     .WORD   QMNGER
```

```
5153                                            .SBTTL  TTY INPUT ROUTINE
5154
5155                                            ;;**********************************************************************
5156                                            .ENABL  LSB
5157
5158                                            ;;**********************************************************************
5159                                            ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
5160                                            ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
5161                                            ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
5162                                            ;*WHEN OPERATING IN TTY FLAG MODE.
5163  022412  022737  000176  001140    $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
5164  022420  001074                            BNE     15$             ;;BRANCH IF NO
5165  022422  105777  156516                    TSTB    @$TKS           ;;CHAR THERE?
5166  022426  100071                            BPL     15$             ;;IF NO, DON'T WAIT AROUND
5167  022430  117746  156512                    MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
5168  022434  042716  177600                    BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
5169  022440  022726  000007                    CMP     #7,(SP)+        ;;IS IT A CONTROL G?
5170  022444  001062                            BNE     15$             ;;NO, RETURN TO USER
5171  022446  123727  001134  000001            CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
5172  022454  001456                            BEQ     15$             ;;BRANCH IF YES
5173
5174  022456  104400  023137                    TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
5175  022462  104400  023144            $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
5176  022466  013746  000176                    MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
5177  022472  104401                            TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
5178  022474  104400  023155                    TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
5179  022500  005046                    19$:    CLR     -(SP)           ;;CLEAR COUNTER
5180  022502  005046                            CLR     -(SP)           ;;THE NEW SWR
5181  022504  105777  156434            7$:     TSTB    @$TKS           ;;CHAR THERE?
5182  022510  100375                            BPL     7$              ;;IF NOT TRY AGAIN
5183
5184  022512  117746  156430                    MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
5185  022516  042716  177600                    BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
5186
5187
5188
5189  022522  021627  000025            9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
5190  022526  001005                            BNE     10$             ;;BRANCH IF NOT
5191  022530  104400  023132                    TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
5192  022534  062706  000006            20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
5193  022540  000757                            BR      19$             ;;LET'S TRY IT AGAIN
5194
5195
5196  022542  021627  000015            10$:    CMP     (SP),#15        ;;IS IT A <CR>?
5197  022546  001022                            BNE     16$             ;;BRANCH IF NO
5198  022550  005766  000004                    TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
5199  022554  001403                            BEQ     11$             ;;BRANCH IF YES
5200  022556  016677  000002  156354            MOV     2(SP),@SWR      ;;SAVE NEW SWR
5201  022564  062706  000006            11$:    ADD     #6,SP           ;;CLEAR UP STACK
5202  022570  104400  001213            14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
5203  022574  123727  001135  000001            CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
5204  022602  001003                            BNE     15$             ;;BRANCH IF NOT
5205  022604  012777  000100  156332            MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
5206  022612  000002            15$:    RTI                             ;;RETURN
5207  022614  004737  024042            16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
5208  022620  021627  000060                    CMP     (SP),#60        ;;CHAR < 0?
```

```
                                        BLT     18$             ;;BRANCH IF YES
                                        CMP     (SP),#67        ;;CHAR > 7?
                                        BGT     18$             ;;BRANCH IF YES
                                        BIC     #60,(SP)+       ;;STRIP-OFF ASCII
                                        TST     2(SP)           ;;IS THIS THE FIRST CHAR
                                        BEQ     17$             ;;BRANCH IF YES
                                        ASL     (SP)            ;;NO, SHIFT PRESENT
                                        ASL     (SP)            ;;   CHAR OVER TO MAKE
                                        ASL     (SP)            ;;   ROOM FOR NEW ONE.
                                17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
                                        BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
                                        BR      7$              ;;GET THE NEXT ONE
                                18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
                                        BR      20$             ;;SIMULATE CONTROL-U
                                .DSABL  LSB
                                ;;*****************************************************************
                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*CALL:
                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
                                ;
                                $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
                                        MOV     4(SP),2(SP)     ;;SAVE THE PS
                                1$:     TSTB    @$TKS           ;;WAIT FOR
                                        BPL     1$              ;;A CHARACTER
                                        MOVB    @$TKB,4(SP)     ;;READ THE TTY
                                        BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
                                        CMP     4(SP),#23       ;;IS IT A CONTROL-S?
                                        BNE     3$              ;;BRANCH IF NO
                                2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
                                        BPL     2$              ;;LOOP UNTIL ITS THERE
                                        MOVB    @$TKB,-(SP)     ;;GET CHARACTER
                                        BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
                                        CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
                                        BNE     2$              ;;IF NOT DISCARD IT
                                        BR      1$              ;;YES, RESUME
                                3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
                                        BLT     4$              ;;BRANCH IF YES
                                        CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
                                        BGT     4$              ;;BRANCH IF YES
                                        BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
                                4$:     RTI                     ;;GO BACK TO USER
                                ;;*****************************************************************
                                ;*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 STAC
                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

                                $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
                                1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
                                2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
```

```
5265  023026  101405                        BLOS    4$              ;;BR IF YES
5266  023030  104407                        RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
5267  023032  112613                        MOVB    (SP)+,(R3)      ;;GET CHARACTER
5268  023034  122713  000177          10$:  CMPB    #177,(R3)       ;;IS IT A RUBOUT
5269  023040  001003                        BNE     3$              ;;SKIP IF NOT
5270  023042  104400  001212          4$:   TYPE    ,$QUES          ;;TYPE A '?'
5271  023046  000763                        BR      1$              ;;CLEAR THE BUFFER AND LOOP
5272  023050  111337  023120          3$:   MOVB    (R3),9$         ;;ECHO THE CHARACTER
5273  023054  104400  023120                TYPE    ,9$
5274  023060  122723  000015                CMPB    #15,(R3)+       ;;CHECK FOR RETURN
5275  023064  001356                        BNE     2$              ;;LOOP IF NOT RETURN
5276  023066  105063  177777                CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
5277  023072  104400  001214                TYPE    ,$LF            ;;TYPE A LINE FEED
5278  023076  012603                        MOV     (SP)+,R3        ;;RESTORE R3
5279  023100  011646                        MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
5280  023102  016666  000004  000002        MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
5281  023110  012766  023122  000004        MOV     #$TTYIN,4(SP)
5282  023116  000002                        RTI                     ;;RETURN
5283  023120     000          9$:   .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
5284  023121     000                .BYTE   0                       ;;TERMINATOR
5285  023122  000010          $TTYIN: .BLKB   8.                    ;;RESERVE 8 BYTES FOR TTY INPUT
5286  023132  052536  005015     000  $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
5287  023137     136  006507  000012  $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
5288  023144  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
5289  023152  020075     000
5290  023155     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
5291  023162  036440  000040
```

```
                                        .SBTTL  READ AN OCTAL NUMBER FROM THE TTY

                                        ;;**********************************************************************
                                        ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
                                        ;*CHANGE IT TO BINARY.
                                        ;*CALL:
                                        ;*      RDOCT                   ;;READ AN OCTAL NUMBER
                                        ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
                                        ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT

023166  011646                 $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
023170  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
023176  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
023200  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
023202  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
023204  104410                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
023206  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
023210  005001                          CLR     R1              ;;CLEAR DATA WORD
023212  005002                          CLR     R2
023214  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
023216  001412                          BEQ     3$              ;;IF ZERO GET OUT
023220  006301                          ASL     R1              ;;*2
023222  006102                          ROL     R2
023224  006301                          ASL     R1              ;;*4
023226  006102                          ROL     R2
023230  006301                          ASL     R1              ;;*8
023232  006102                          ROL     R2
023234  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
023240  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
023242  000764                          BR      2$              ;;LOOP
023244  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
023246  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
023252  010237  023266                  MOV     R2,$HIOCT
023256  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
023260  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
023262  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
023264  000002                          RTI                     ;;RETURN
023266  000000                  $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
```

```
                                        .SBTTL  READ A DECIMAL NUMBER FROM THE TTY

                                        ;;**************************************************************
                                        ;*THIS ROUTINE WILL READ A DECIMAL (ASCII) NUMBER FROM THE TTY AND
                                        ;*CHANGE IT TO BINARY. IF TOO MANY CHARACTERS OR ANY ILLEGAL CHARACTERS
                                        ;*ARE READ A "?" FOLLOWED BY A CARRIAGE RETURN-LINE FEED WILL BE TYPED.
                                        ;*THE COMPLETE NUMBER MUST BE RETYPED. THE INPUT IS TERMINATED BY THE
                                        ;*USER TYPING A CARRIAGE RETURN. THE RANGE OF THE INPUT NUMBER IS
                                        ;*POSITIVE 32767 TO NEGATIVE 32768.
                                        ;*CALL:
                                        ;*      RDDEC                   ;;READ A DECIMAL NUMBER
                                        ;*      RETURN HERE             ;;NUMBER IS ON TOP OF THE STACK
                                        ;
023270  011646          $RDDEC: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR
023272  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE INPUT NUMBER
023300  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
023302  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
023304  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
023306  104410          1$:     RDLIN                   ;;READ AN ASCIZ LINE
023310  012600                  MOV     (SP)+,R0        ;;ADDRESS OF 1ST CHAR.
023312  010037  023436          MOV     R0,6$           ;;SAVE INCASE OF BAD INPUT
023316  005046                  CLR     -(SP)           ;;CLEAR DATA WORD
023320  005002                  CLR     R2              ;;SIGN SET POSITIVE
023322  122710  000055          CMPB    #'-,(R0)        ;;SEE IF A MINUS SIGN WAS TYPED
023326  001001                  BNE     2$              ;;BR IF NO MINUS SIGN
023330  112002                  MOVB    (R0)+,R2        ;;SAVE FOR LATER USE
023332  112001          2$:     MOVB    (R0)+,R1        ;;PICKUP THIS CHARACTER
023334  001424                  BEQ     3$              ;;GET OUT IF ZERO
023336  122701  000060          CMPB    #'0,R1          ;;MAKE SURE THIS CHARACTER
023342  003032                  BGT     5$              ;;IS A DIGIT BETWEEN 0 & 9
023344  122701  000071          CMPB    #'9,R1
023350  002427                  BLT     5$
023352  032716  170000          BIT     #^C7777,(SP)    ;;DON'T LET NUMBER GET TO BIG
023356  001024                  BNE     5$              ;;BR IF NUMBER WOULD OVERFLOW
023360  006316                  ASL     (SP)            ;;*2
023362  011646                  MOV     (SP),-(SP)      ;;SAVE FOR LATER
023364  006316                  ASL     (SP)            ;;*4
023366  006316                  ASL     (SP)            ;;*8.
023370  062616                  ADD     (SP)+,(SP)      ;;*10.
023372  102416                  BVS     5$              ;;OVERFLOW ISN'T ALLOWED
023374  162701  000060          SUB     #'0,R1          ;;STRIP AWAY THE ASCII JUNK
023400  060116                  ADD     R1,(SP)         ;;ADD IN THIS DIGIT
023402  102412                  BVS     5$              ;;OVERFLOW ISN'T ALLOWED
023404  000752                  BR      2$              ;;LOOP
023406  005702          3$:     TST     R2              ;;CHECK IF NUMBER IS NEG
023410  001401                  BEQ     4$              ;;BR IF NO
023412  005416                  NEG     (SP)            ;;YES--NEGATE THE NUMBER
023414  012666  000012  4$:     MOV     (SP)+,12(SP)    ;;SAVE THE RESULT
023420  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
023422  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
023424  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
023426  000002                  RTI                     ;;RETURN

023430  005726          5$:     TST     (SP)+           ;;CLEAN PARTIAL NUMBER FROM STACK
023432  105010                  CLRB    (R0)            ;;SET A TERMINATOR
```

```
 5386  023434  104400                          TYPE                    ;;TYPE THE INPUT UP TO BAD CHAR.
 5387  023436  000000                  6$:     .WORD   0               ;;POINTER GOES HERE
 5388  023440  104400  001212                  TYPE    ,$QUES          ;;"?" "CR" &"LF"
 5389  023444  000720                          BR      1$              ;;TRY AGAIN
```

```
                                        .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
                                
023446                          $TYPDS:
023446  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
023450  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
023452  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
023454  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
023456  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
023460  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
023464  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
023470  100004                          BPL     1$              ;;BR IF INPUT IS POS.
023472  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
023474  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
023502  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
023504  012703  023662                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
023510  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
023514  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
023516  016001  023652                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
023522  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
023524  002402                          BLT     4$              ;;BR IF DONE
023526  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
023530  000774                          BR      3$
023532  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
023534  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
023536  001002                          BNE     5$              ;;FALL THROUGH IF 0
023540  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
023542  100407                          BMI     7$              ;;BR IF YES
023544  106316                  5$:     ASLB    (SP)            ;;MSD?
023546  103003                          BCC     6$              ;;BR IF NO
023550  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
023556  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
023562  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
023566  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
023570  005720                          TST     (R0)+           ;;JUST INCREMENTING
023572  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
023576  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
023600  003002                          BGT     8$              ;;GO TO EXIT
023602  010502                          MOV     R5,R2           ;;GET THE LSD
023604  000764                          BR      6$              ;;GO CHANGE TO ASCII
023606  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
023610  100003                          BPL     9$              ;;BR IF NO
023612  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
023620  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
023622  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
023624  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
023626  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
```

```
5446  023630  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
5447  023632  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
5448  023634  104400  023662                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
5449  023640  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
5450  023646  012616                          MOV     (SP)+,(SP)
5451  023650  000002                          RTI                     ;;RETURN TO USER
5452  023652  023420                  $DTBL:  10000.
5453  023654  001750                          1000.
5454  023656  000144                          100.
5455  023660  000012                          10.
5456  023662  000004                  $DBLK:  .BLKW   4
```

```
                                        .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
                                        ;*

023672  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
023676  100002                          BPL     1$              ;;BR IF YES
023700  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
023702  000407                          BR      3$              ;;LEAVE
023704  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
023706  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
023712  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
023714  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
023716  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
023720  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
023722  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
023726  000002                          RTI                     ;;RETURN
023730  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
023734  001430                          BEQ     8$
023736  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
023742  001006                          BNE     5$
023744  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
023746  104400                          TYPE                    ;;TYPE A CR AND LF
023750  001213                          $CRLF
023752  105037  024106                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
023756  000755                          BR      2$              ;;GET NEXT CHARACTER
023760  004737  024042          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
023764  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
023770  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
023772  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
023776  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
024002  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
024004  004737  024042                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
024010  105337  024106                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
024014  000770                          BR      7$              ;;LOOP

                                        ;HORIZONTAL TAB PROCESSOR

024016  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
024022  004737  024042          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
024026  132737  000007  024106          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
024034  001372                          BNE     9$              ;;TAB STOP
024036  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
```

```
024040  000724                              BR      2$              ;;GET NEXT CHARACTER
024042  105777  155102              $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
024046  100375                              BPL     $TYPEC
024050  116677  000002  155074              MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
024056  122766  000015  000002              CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
024064  001003                              BNE     1$              ;;BRANCH IF NO
024066  105037  024106                      CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
024072  000406                              BR      $TYPEX          ;;EXIT
024074  122766  000012  000002      1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
024102  001402                              BEQ     $TYPEX          ;;BRANCH IF YES
024104  105227                              INCB    (PC)+           ;;COUNT THE CHARACTER
024106  000000                      $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
024110  000207                      $TYPEX: RTS     PC
```

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

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

024112  104413                  $DB20:  SAVREG                  ;;SAVE ALL REGISTERS
024114  016601  000002                  MOV     2(SP),R1        ;;PICKUP THE POINTER TO LOW WORD
024120  012705  024231                  MOV     #$OCTVL+13.,R5  ;;POINTER TO DATA TABLE
024124  012704  000014                  MOV     #12.,R4         ;;DO ELEVEN CHARACTERS
024130  012703  177770                  MOV     #^C7,R3         ;;MASK
024134  012100                          MOV     (R1)+,R0        ;;LOWER WORD
024136  012101                          MOV     (R1)+,R1        ;;HIGH WORD
024140  005002                          CLR     R2              ;;TERMINATOR
024142  110245                  1$:     MOVB    R2,-(R5)        ;;PUT CHARACTER IN DATA TABLE
024144  010002                          MOV     R0,R2           ;;GET THIS DIGIT
024146  005304                          DEC     R4              ;;COUNT THIS CHARACTER
024150  003007                          BGT     3$              ;;BR IF NOT THE LAST DIGIT
024152  001405                          BEQ     2$              ;;BR IF IT IS THE LAST DIGIT
024154  005205                          INC     R5              ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
024156  010566  000002                  MOV     R5,2(SP)        ;;ASCIZ CHAR. & PUT IT ON THE STACK
024162  104414                          RESREG                  ;;RESTORE ALL REGISTERS
024164  000207                          RTS     PC              ;;RETURN TO USER
024166  006203                  2$:     ASR     R3              ;;POSITION THE MASK FOR THE LAST DIGIT
024170  006001                  3$:     ROR     R1              ;;POSITION THE BINARY NUMBER FOR
024172  006000                          ROR     R0              ;;      THE NEXT OCTAL DIGIT
024174  006001                          ROR     R1
024176  006000                          ROR     R0
024200  006001                          ROR     R1
024202  006000                          ROR     R0
024204  040302                          BIC     R3,R2           ;;MASK OUT ALL JUNK
024206  062702  000060                  ADD     #'0,R2          ;;MAKE THIS CHAR. ASCII
024212  000753                          BR      1$              ;;GO PUT IT IN THE DATA TABLE
024214  000016                  $OCTVL: .BLKB   14.             ;;RESERVE DATA TABLE
```

```
                                        .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
                                        ;;
                                        ;;
024232  104413                  $DB2D:  SAVREG                  ;;SAVE REGISTERS
024234  016602  000002                  MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
024240  012700  024412                  MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
024244  010066  000002                  MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
024250  012201                          MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
024252  012202                          MOV     (R2)+,R2
024254  012737  000012  024330          MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
024262  012704  024342                  MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
024266  012705  024344                  MOV     #$TNPWR+2,R5
024272  005003                  1$:     CLR     R3              ;;CLEAR PARTIAL
024274  161401                  2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
024276  005602                          SBC     R2
024300  161502                          SUB     (R5),R2
024302  002402                          BLT     3$              ;;BR IF TEN POWER TO LARGE
024304  005203                          INC     R3              ;;ADD 1 TO PARTIAL
024306  000772                          BR      2$              ;;LOOP
024310  062401                  3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
024312  005502                          ADC     R2
024314  062402                          ADD     (R4)+,R2
024316  022525                          CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
024320  052703  000060                  BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
024324  110320                          MOVB    R3,(R0)+        ;;SAVE IT
024326  005327                          DEC     (PC)+           ;;DONE?
024330  000000                  4$:     .WORD   0
024332  001357                          BNE     1$              ;;BR IF NO
024334  105020                          CLRB    (R0)+           ;;TERMINATOR
024336  104414                          RESREG                  ;;RESTORE REGISTERS
024340  000207                          RTS     PC              ;;RETURN
024342  145000                  $TNPWR: 145000                  ;;1.0E09
024344  035632                          35632
024346  160400                          160400                  ;;1.0E08
024350  002765                          2765
024352  113200                          113200                  ;;1.0E07
024354  000230                          230
024356  041100                          041100                  ;;1.0E06
024360  000017                          17
024362  103240                          103240                  ;;1.0E05
024364  000001                          1
024366  023420                          23420                   ;;1.0E04
024370  000000                          0
024372  001750                          1750                    ;;1.0E03
024374  000000                          0
024376  000144                          144                     ;;1.0E02
```

```
 5622  024400  000000                                  0
 5623  024402  000012                                  12                  ;;1.0E01
 5624  024404  000000                                  0
 5625  024406  000001                                  1                   ;;1.0E00
 5626  024410  000000                                  0
 5627  024412  000014                        $DECVL:   .BLKB   12.         ;;RESERVE STORAGE FOR ASCIZ STRING
```

```
5628                                    40600           .SBTTL  SUPRS - TYPE NUMERICAL ASCIZ STRING, REPLACE LEADING 0'S BY BLAN
5629                                    40610           .SBTTL  SUPRSL - TYPE NUMERICAL ASCIZ STRING, LEFT JUSTIFY
5630                                    40620           ;NOT FROM SYSMAC
5631                                    40630
5632  024426  010046                    40640   SUPRSL: MOV     R0,-(SP)        ;SAVE R0
5633  024430  005037  024516            40650           CLR     SUP2
5634  024434  016600  000004            40660           MOV     4(SP),R0
5635  024440  000405                    40670           BR      SUP1
5636                                    40680
5637  024442  010046                    40690   SUPRS:  MOV     R0,-(SP)        ;SAVE R0
5638  024444  016600  000004            40700           MOV     4(SP),R0        ;PICKUP THE POINTER
5639  024450  010037  024516            40710           MOV     R0,SUP2         ;SAVE FOR TYPING
5640  024454                            40720   SUP1:
5641  024454  105710                    40730   1$:     TSTB    (R0)            ;TERMINATOR?
5642  024456  001406                    40740           BEQ     2$              ;BR IF YES
5643  024460  122710  000060            40750           CMPB    #'0,(R0)        ;IS THIS AN ASCII "0"?
5644  024464  001006                    40760           BNE     4$              ;NO
5645  024466  112720  000040            40770           MOVB    #40,(R0)+       ;REPLACE IT WITH "BLANK"
5646  024472  000770                    40780           BR      1$
5647  024474  005300                    40790   2$:     DEC     R0              ;BACKUP BY 1
5648  024476  112710  000060            40800           MOVB    #'0,(R0)        ;ASCII "0"
5649  024502  005737  024516            40810   4$:     TST     SUP2            ;LEFT JUSTIFY?
5650  024506  001002                    40820           BNE     5$              ;NO
5651  024510  010037  024516            40830           MOV     R0,SUP2         ;YES
5652  024514  104400                    40840   5$:     TYPE                    ;GC TYPE
5653  024516  000000                    40850   SUP2:   .WORD   0
5654  024520  012600                    40860           MOV     (SP)+,R0        ;RESTORE R0
5655  024522  012616                    40870           MOV     (SP)+,(SP)      ;RESTORE THE STACK
5656  024524  000207                    40880           RTS     PC              ;RETURN
```

```
                                        .SBTTL  INTEGER MULTIPLY ROUTINE

                                        ;;*****************************************************************
                                        ;*CALL
                                        ;*      MOV     MULTIPLER,-(SP)
                                        ;*      MOV     MULTIPLICAND,-(SP)
                                        ;*      JSR     PC,@#$MULT
                                        ;*      RETURN  ;;PRODUCT IS ON THE STACK
                                        ;*
                                        ;*      STACK   PRODUCT
                                        ;*      -----   -------
                                        ;*      TOP     LSB'S
                                        ;*      +2      MSB'S

024526                                  $MULT:
024526  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
024530  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
024532  010246                                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
024534  005046                                  CLR     -(SP)           ;;CLEAR THE SIGN KEY
024536  016601  000012                          MOV     12(SP),R1       ;;GET THE MULTIPLICAND
024542  100002                                  BPL     1$              ;;BR IF PLUS
024544  005216                                  INC     (SP)            ;;SET THE SIGN KEY
024546  005401                                  NEG     R1              ;;MAKE THE MULTIPLICAND POSTIVE
024550  016602  000014                  1$:     MOV     14(SP),R2       ;;GET THE MULTIPLIER
024554  100002                                  BPL     2$              ;;BR IF PLUS
024556  005316                                  DEC     (SP)            ;;UPDATE THE SIGN KEY
024560  005402                                  NEG     R2              ;;MAKE THE MULTIPLIER POSTIVE
024562  012746  000021          2$:     MOV     #17.,-(SP)      ;;SET THE LOOP COUNT
024566  005000                                  CLR     R0              ;;SETUP FOR THE MULTIPLY LOOP
024570  103001                          3$:     BCC     4$              ;;DON'T ADD IF MULTIPLICAND = 0
024572  060200                                  ADD     R2,R0
024574  006000                          4$:     ROR     R0              ;;POSITION THE PARITIAL PRODUCT AND
024576  006001                                  ROR     R1              ;;THE MULTIPLICAND
024600  005316                                  DEC     (SP)            ;;HAS ALL BITS OF THE MULTIPLICAND BEEN DONE?
024602  001372                                  BNE     3$              ;;BR IF NO
024604  022616                                  CMP     (SP)+,(SP)      ;;SHOULD PRODUCT BE NEGATIVE?
024606  001403                                  BEQ     5$              ;;GO TO EXIT IF NO
024610  005400                                  NEG     R0              ;;YES--SO MAKE IT SO
024612  005401                                  NEG     R1
024614  005600                                  SBC     R0
024616  005726                          5$:     TST     (SP)+           ;;CLEAR SIGN INFO. OFF OF STACK
024620  010066  000012                          MOV     R0,12(SP)       ;;PUT THE PRODUCT ON THE STACK (MSB'S)
024624  010166  000010                          MOV     R1,10(SP)       ;;LSB'S
024630  012602                                  MOV     (SP)+,R2        ;;POP STACK INTO R2
024632  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
024634  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
024636  000207                                  RTS     PC
```

```
 5704                                           40920   .SBTTL  INTEGER DIVIDE ROUTINE
 5705                                           40930
 5706                                           40940   ;*CALL:
 5707                                           40950   ;*      MOV     LOW DIVIDEND,-(SP)      ;THE HIGH DIVIDEND MUST BE < 1/2
 5708                                           40960   ;*      MOV     HIGH DIVIDEND,-(SP)     ;       AS LARGE AS THE DIVISOR
 5709                                           40970   ;*      MOV     DIVISOR,-(SP)
 5710                                           40980   ;*      JSR     PC,$DIV
 5711                                           40990   ;*      RETURN                          ;QUOTIENT & REMAINDER ARE ON THE STACK
 5712                                           41000   ;*      "V"=0   IMPLIES NO ERROR
 5713                                           41010   ;*      "V"=1   IMPLIES ERROR OCCURRED
 5714                                           41020   ;*              "C"=0   DIVIDE OVERFLOW OCCURRED
 5715                                           41030   ;*              "C"=1   ATTEMPTED TO DIVIDE BY ZERO
 5716                                           41040   ;*
 5717                                           41050   ;*
 5718                                           41060   ;*      STACK   NO ERROR        OVERFLOW        DIVIDE BY ZERO
 5719                                           41070   ;*      -----   --------        --------        -------------
 5720                                           41080   ;*      TOP     REMAINDER       ALL ZEROS       ALL ONES
 5721                                           41090   ;*      +2      QUOTIENT        ALL ZEROS       ALL ONES
 5722                                           41100
 5723  024640  013746  000034                  41110   $DIV:   MOV     34,-(SP)        ;SAVE CURRENT TRAP VECTOR
 5724  024644  012737  024654  000034          41120           MOV     #1$,34          ;SET UP TRAP VECTOR
 5725  024652  104400                          41130           TRAP
 5726  024654  012716  024676                  41140   1$:     MOV     #2$,(SP)        ;REPLACE NEW PC
 5727  024660  016637  000004  000034          41150           MOV     4(SP),34        ;RESTORE OLD TRAP VECT
 5728  024666  016666  000002  000004          41160           MOV     2(SP),4(SP)     ;SAVE PSW
 5729  024674  000002                          41170           RTI                     ;RESTORE PSW
 5730                                          41180
 5731  024676  042716  000017                  41190   2$:     BIC     #17,(SP)        ;STRIP AWAY CONDITION CODES
 5732  024702  010046                          41200           MOV     R0,-(SP)        ;PUSH R0 ON STACK
 5733  024704  010146                          41210           MOV     R1,-(SP)        ;PUSH R1 ON STACK
 5734  024706  010246                          41220           MOV     R2,-(SP)        ;PUSH R2 ON STACK
 5735  024710  010346                          41230           MOV     R3,-(SP)        ;PUSH R3 ON STACK
 5736  024712  005046                          41240           CLR     -(SP)           ;SAVE A PLACE FOR SIGNS
 5737  024714  012746  000021                  41250           MOV     #17.,-(SP)      ;SETUP THE ITERATION COUNTER
 5738  024720  016601  000024                  41260           MOV     24(SP),R1       ;PICKUP THE DIVIDEND
 5739  024724  016600  000022                  41270           MOV     22(SP),R0
 5740  024730  100005                          41280           BPL     3$              ;CHECK THE SIGN
 5741  024732  105366  000003                  41290           DECB    3(SP)           ;KEEP TRACK OF THE SIGN
 5742  024736  005400                          41300           NEG     R0              ;AND NEGATE THE ORIGINAL
 5743  024740  005401                          41310           NEG     R1              ;NUMBER
 5744  024742  005600                          41320           SBC     R0
 5745  024744  016602  000020                  41330   3$:     MOV     20(SP),R2       ;PICKUP THE DIVISOR
 5746  024750  022702  000001                  41340           CMP     #1,R2           ;IF THE DIVISOR IS 1 SKIP THE REST
 5747  024754  001463                          41350           BEQ     13$             ;YES
 5748  024756  005702                          41360           TST     R2
 5749  024760  002407                          41370           BLT     4$              ;CHECK THE SIGN
 5750  024762  003011                          41380           BGT     5$              ;DIVISOR OF 0 IS A NO-NO
 5751  024764  052766  000003  000014          41390           BIS     #3,14(SP)       ;SET "V" & "C"
 5752  024772  012700  177777                  41400           MOV     #-1,R0          ;SET REMAINDER TO ALL ONES
 5753  024776  000424                          41410           BR      9$              ;EXIT
 5754  025000  005266  000002                  41420   4$:     INC     2(SP)           ;KEEP TRACK OF DIVISORS SIGN
 5755  025004  000401                          41430           BR      6$
 5756  025006  005402                          41440   5$:     NEG     R2              ;NEGATE THE ORIGINAL NUMBER
 5757  025010  000241                          41450   6$:     CLC                     ;CLEAR "C"
 5758  025012  000405                          41460           BR      8$              ;START FORMING QUOTIENT
 5759  025014  006100                          41470   7$:     ROL     R0              ;POSITION MSB'S
```

```
 5760  025016  010003                 41480            MOV     R0,R3           ;COPY
 5761  025020  060203                 41490            ADD     R2,R3           ;COMPARE DIVIDEND & DIVISOR
 5762  025022  103001                 41500            BCC     8$              ;BR IF DIVIDEND > DIVISOR
 5763  025024  010300                 41510            MOV     R3,R0           ;REMAINDER AFTER THIS LOOP
 5764  025026  006101                 41520    8$:     ROL     R1              ;QUOTIENT BIT ENTERS HERE
 5765  025030  005316                 41530            DEC     (SP)            ;DONE?
 5766  025032  001370                 41540            BNE     7$              ;BR IF NO
 5767  025034  005701                 41550            TST     R1              ;OVERFLOW?
 5768  025036  100005                 41560            BPL     10$             ;BR IF NO
 5769  025040  052766  000002  000014 41570            BIS     #2,14(SP)       ;SET "V" IN RETURN STATUS WORD
 5770  025046  005000                 41580            CLR     R0              ;SET REMAINDER TO ALL ZEROS
 5771  025050  010001                 41590    9$:     MOV     R0,R1           ;COPY REMAINDER INTO QUOTIENT
 5772  025052  005726                 41600    10$:    TST     (SP)+           ;CLEAR COUNTER FROM STACK
 5773  025054  005716                 41610            TST     (SP)            ;REMAINDER SIGN CORRECTION NEEDED?
 5774  025056  002004                 41620            BGE     11$             ;BR IF NO
 5775  025060  005400                 41630            NEG     R0              ;NEGATE REMAINDER
 5776  025062  105066  000001         41640            CLRB    1(SP)           ;CLEAR SIGN
 5777  025066  005316                 41650            DEC     (SP)            ;BUT DON'T FORGET QUOTIENT
 5778  025070  005726                 41660    11$:    TST     (SP)+           ;QUOTIENT SIGN CORRECTION NEEDED?
 5779  025072  001401                 41670            BEQ     12$             ;BR IF NO
 5780  025074  005401                 41680            NEG     R1              ;NEGATE QUOTIENT
 5781  025076  010166  000020         41690    12$:    MOV     R1,20(SP)       ;RETURN QUOTIENT AND
 5782  025102  010066  000016         41700            MOV     R0,16(SP)       ;REMAINDER TO USER
 5783  025106  012603                 41710            MOV     (SP)+,R3        ;POP STACK INTO R3
 5784  025110  012602                 41720            MOV     (SP)+,R2        ;POP STACK INTO R2
 5785  025112  012601                 41730            MOV     (SP)+,R1        ;POP STACK INTO R1
 5786  025114  012600                 41740            MOV     (SP)+,R0        ;POP STACK INTO R0
 5787  025116  012666  000002         41750            MOV     (SP)+,2(SP)     ;SETUP TO RETURN CONDITION CODES
 5788  025122  000002                 41760            RTI                     ;RETURN
 5789  025124  022626                 41770    13$:    CMP     (SP)+,(SP)+     ;POP THE STACK
 5790  025126  000763                 41780            BR      12$
```

```
                                        .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

025130                                  $SAVREG:
025130  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
025132  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
025134  010246                                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
025136  010346                                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
025140  010446                                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
025142  010546                                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
025144  016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
025150  016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
025154  016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
025160  016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
025164  000002                                  RTI

                                        ;*RESTORE R0-R5
                                        ;*CALL:
                                        ;*      RESREG
025166                                  $RESREG:
025166  012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
025172  012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
025176  012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
025202  012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
025206  012605                                  MOV     (SP)+,R5        ;;POP STACK INTO R5
025210  012604                                  MOV     (SP)+,R4        ;;POP STACK INTO R4
025212  012603                                  MOV     (SP)+,R3        ;;POP STACK INTO R3
025214  012602                                  MOV     (SP)+,R2        ;;POP STACK INTO R2
025216  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
025220  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
025222  000002                                  RTI
```

```
 5836                                          41820  .SBTTL  RANDOM NUMBER GENERATOR ROUTINE
 5837                                          41830  ;*CALL:
 5838                                          41840  ;*      JSR     PC,$RAND        ;CALL THE ROUTINE
 5839                                          41850  ;*      RETURN                  ;RETURN HERE THE RANDOM
 5840                                          41860  ;*                              ;NUMBER WILL BE IN
 5841                                          41870  ;*                              ;$HINUM,$LONUM
 5842  025224                                  41880  $RAND:
 5843  025224  010046                         41890          MOV     R0,-(SP)        ;PUSH R0 ON STACK
 5844  025226  010146                         41900          MOV     R1,-(SP)        ;PUSH R1 ON STACK
 5845  025230  010246                         41910          MOV     R2,-(SP)        ;PUSH R2 ON STACK
 5846  025232  010346                         41920          MOV     R3,-(SP)        ;PUSH R3 ON STACK
 5847  025234  010446                         41930          MOV     R4,-(SP)        ;PUSH R4 ON STACK
 5848  025236  017604  000012                 41940          MOV     12(SP),R4       ;GET POINTER TO THE SAVED SEEDS
 5849                                          41950                                  ;FOR GENERATING THIS RANDOM NUMBER
 5850  025242  011400                         41960          MOV     (R4),R0         ;GET LO NUMBER SEED
 5851  025244  016401  000002                 41970          MOV     2(R4),R1        ;GET HIGH NUMBER SEED
 5852  025250  012703  177771                 41980          MOV     #-7,R3          ;SET SHIFT COUNT
 5853  025254  005002                         41990          CLR     R2              ;ZERO R2
 5854  025256  006300                         42000  1$:     ASL     R0              ;SHIFT R0 LEFT AND
 5855  025260  006101                         42010          ROL     R1              ;ROTATE CARRY INTO R1 AND
 5856  025262  006102                         42020          ROL     R2              ;ROTATE CARRY INTO R2
 5857  025264  005203                         42030          INC     R3              ;CHECK FOR DONE
 5858  025266  001373                         42040          BNE     1$              ;CONTINUE SHIFT LOOP
 5859  025270  061400                         42050          ADD     (R4),R0         ;ADD NUMBER TO MAKE X 129
 5860  025272  005501                         42060          ADC     R1              ;PROPOGATE CARRY
 5861  025274  066401  000002                 42070          ADD     2(R4),R1        ;ADD NUMBER TO MAKE X 129
 5862  025300  005502                         42080          ADC     R2              ;PROPOGATE CARRY
 5863  025302  062700  001057                 42090          ADD     #1057,R0        ;ADD LOW CONSTANT
 5864  025306  005501                         42100          ADC     R1              ;PROPOGATE CARRY
 5865  025310  005502                         42110          ADC     R2              ;PROPOGATE CARRY
 5866  025312  062701  047401                 42120          ADD     #47401,R1       ;ADD HIGH CONSTANT
 5867  025316  005502                         42130          ADC     R2              ;PROPOGATE CARRY
 5868  025320  062702  000006                 42140          ADD     #6,R2           ;ADD HIGHEST CONSTART
 5869  025324  060200                         42150          ADD     R2,R0           ;REPRIME R0 WITH HIGHEST DIGIT
 5870  025326  005501                         42160          ADC     R1              ;PROPOGATE CARRY
 5871  025330  010014                         42170          MOV     R0,(R4)         ;SAVE R0-$LONUM (FOR USE NXT TIME)
 5872  025332  010164  000002                 42180          MOV     R1,2(R4)        ;SAVE R1-$HINUM (FOR USE NXT TIME)
 5873  025336  012604                         42190          MOV     (SP)+,R4
 5874  025340  012603                         42200          MOV     (SP)+,R3        ;POP STACK INTO R3
 5875  025342  012602                         42210          MOV     (SP)+,R2        ;POP STACK INTO R2
 5876  025344  012601                         42220          MOV     (SP)+,R1        ;POP STACK INTO R1
 5877  025346  012600                         42230          MOV     (SP)+,R0        ;POP STACK INTO R0
 5878  025350  062716  000002                 42240          ADD     #2,(SP)         ;ADJUST SP FOR CORRECT RETURN
 5879  025354  000207                         42250          RTS     PC              ;RETURN
 5880  025356  123456                         42260  RSDRVL: 123456                  ;RANDOM SEED FOR DRIVE SELECTION (LO)
 5881  025360  176543                         42270  RSDRVH: 176543                  ;  "    "      (HI)
 5882  025362  001201                         42280  RSFUNL: 1201                    ;RANDOM SEED FOR FUNCTION
 5883  025364  062465                         42290  RSFUNH: 62465                   ;  "    "      (HI)
 5884  025366  176105                         42300  RSCYLL: 176105                  ;RANDOM SEED FOR CYLINDER (LO)
 5885  025370  174532                         42310  RSCYLH: 174532                  ;  "    "      (HI)
 5886  025372  157650                         42320  RSBAL:  157650                  ;RANDOM SEED FOR BUS ↑ADDRESS (LO)
 5887  025374  030753                         42330  RSBAH:  30753                   ;  "    "      (HI)
 5888  025376  131547                         42340  RSWCL:  131547                  ;RANDOM SEED FOR WORD COUNT (LO)
 5889  025400  032070                         42350  RSWCH:  32070                   ;  "    "      (HI)
 5890  025402  123456                         42360  RSDTL:  123456                  ;RANDOM SEED FOR DATA (LO)
 5891  025404  176543                         42370  RSDTH:  176543                  ;  "    "      (HI)
```

```
                                        .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

005917  025406  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
005918  025412  116637  000001  025631          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
005919  025420  112637  025633                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
005920  025424  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
005921  025430  000406                          BR      $TYPON
005922  025432  112737  000001  025631  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
005923  025440  112737  000006  025633          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
005924  025446  112737  000005  025630  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
005925  025454  010346                          MOV     R3,-(SP)        ;;SAVE R3
005926  025456  010446                          MOV     R4,-(SP)        ;;SAVE R4
005927  025460  010546                          MOV     R5,-(SP)        ;;SAVE R5
005928  025462  113704  025633                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
005929  025466  005404                          NEG     R4
005930  025470  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
005931  025474  110437  025632                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
005932  025500  113704  025631                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
005933  025504  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
005934  025510  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
005935  025512  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
005936  025514  000404                          BR      3$              ;;GO DO MSB
005937  025516  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
005938  025520  006105                          ROL     R5
005939  025522  006105                          ROL     R5
005940  025524  010503                          MOV     R5,R3
005941  025526  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
005942  025530  105337  025632                  DECB    $OMODE          ;;TYPE THIS DIGIT?
005943  025534  100016                          BPL     7$              ;;BR IF NO
005944  025536  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
005945  025542  001002                          BNE     4$              ;;TEST FOR 0
005946  025544  005704                          TST     R4              ;;SUPPRESS THIS 0?
005947  025546  001403                          BEQ     5$              ;;BR IF YES
```

```
 5948  025550  005204                          4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
 5949  025552  052703  000060                          BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
 5950  025556  052703  000040                  5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
 5951  025562  110337  025626                          MOVB    R3,8$           ;;SAVE FOR TYPING
 5952  025566  104400  025626                          TYPE    ,8$             ;;GO TYPE THIS DIGIT
 5953  025572  105337  025630                  7$:     DECB    $OCNT           ;;COUNT BY 1
 5954  025576  003347                                  BGT     2$              ;;BR IF MORE TO DO
 5955  025600  002402                                  BLT     6$              ;;BR IF DONE
 5956  025602  005204                                  INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
 5957  025604  000744                                  BR      2$              ;;GO DO THE LAST DIGIT
 5958  025606  012605                          6$:     MOV     (SP)+,R5        ;;RESTORE R5
 5959  025610  012604                                  MOV     (SP)+,R4        ;;RESTORE R4
 5960  025612  012603                                  MOV     (SP)+,R3        ;;RESTORE R3
 5961  025614  016666  000002  000004                  MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
 5962  025622  012616                                  MOV     (SP)+,(SP)
 5963  025624  000002                                  RTI                     ;;RETURN
 5964  025626  000                             8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
 5965  025627  000                                     .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
 5966  025630  000                             $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
 5967  025631  000                             $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
 5968  025632  000000                          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
 5969                          042400
```

```
 5970                                          42420   .SBTTL  ERROR HANDLER ROUTINE
 5971                                          42430
 5972                                          42440   ;*SW15=1        HALT ON ERROR
 5973                                          42450   ;*SW13=1        INHIBIT ERROR TYPEOUTS
 5974                                          42460   ;*SW12=1        TYPE OUT THE ERROR HISTORY, THE FUNCTION THAT
 5975                                          42470   ;               WAS BEING PERFORMED ON RK AT THE TIME OF ERROR AND
 5976                                          42480   ;               THE FUNCTION PERFORMED PRIOR TO THAT.
 5977                                          42490   ;*NOTE THIS SWITCH OPTION (12) IS MEANINGFUL ONLY FOR ERRORS OCCURING IN THE
 5978                                          42500   ;*EXERCISER PART OF THE PROGRAM.
 5979                                          42510   ;*SW11=1        DUMP OUT ALL RK REGISTERS
 5980                                          42520   ;*SW10=1        BELL ON ERROR
 5981                                          42530   ;*SW09=1        LOOP ON ERROR
 5982                                          42540   ;*SW03=1        TYPE OUT TIME AT WHICH ERROR OCCURED
 5983                                          42550   ;*SW02=1        DROP THE DRIVE AFTER MAXM ERORS ON THIS DRIVE
 5984                                          42560   ;*SW01=1        TYPE OUT THE SERIAL NUMBER OF THE ERRORING DRIVE
 5985                                          42570   ;*GO TO $ERRTYP ON ERROR
 5986                                          42580
 5987                                          42590
 5988  025634  105237  001103                 42600   $ERROR: INCB    $ERFLG          ;SET THE ERROR FLAG
 5989  025640  001775                         42610           BEQ     $ERROR          ;DON'T LET THE FLAG GO TO ZERO
 5990  025642  013737  001102  001142         42620           MOV     $TSTNM,@#DISPLAY;DISPLAY TEST NUMBER AND ERROR FLAG
 5991  025650  032777  002000  153262         42630           BIT     #SW10,@SWR      ;BELL ON ERROR?
 5992  025656  001402                         42640           BEQ     1$              ;NO - SKIP
 5993  025660  104460  001206                 42650           TYPE    ,$BELL          ;RING BELL
 5994  025664  005237  001112                 42660   1$:     INC     $ERTTL          ;COUNT THE NUMBER OF ERRORS
 5995                                          42670
 5996  025670  032777  000004  153242         42680           BIT     #SW2,@SWR       ;COUNT # OF ERORS & DROP DRVE?
 5997  025676  001415                         42690           BEQ     3$              ;NO
 5998                                          42700                                   ;YES
 5999  025700  010146                         42710           MOV     R1,-(SP)        ;SAVE R1
 6000  025702  013701  001750                 42720           MOV     $RDRV,R1        ;GET ERRORING DRIVE #
 6001  025706  100410                         42730           BMI     2$              ;IF (SRDRV)= -1, SKIP(BECAUSE THE
 6002                                          42740                                   ;EROR WAS NOT ATTRIBUTABLE TO ANY
 6003                                          42750                                   ;SPECIFIC DRIVE)
 6004  025710  105261  002242                 42760           INCB    ERDRV(R1)       ;COUNT # OF ERORS ON THIS DRIVE
 6005  025714  126127  002242  000003         42770           CMPB    ERDRV(R1),#3    ;# OF ERORS GREATER THAN ALLOWABLE?
 6006  025722  101402                         42780           BLOS    2$              ;NO
 6007  025724  000137  015760                 42790           JMP     DSELCT          ;DROP THE DRIVE
 6008                                          42800
 6009  025730  012601                         42810   2$:     MOV     (SP)+,R1        ;RESTORE R1
 6010                                          42820
 6011  025732  011637  001116                 42830   3$:     MOV     (SP),$ERRPC     ;GET ADDRESS OF ERROR INSTRUCTION
 6012  025736  162737  000002  001116         42840           SUB     #2,$ERRPC
 6013  025744  005046                         42850           CLR     -(SP)
 6014  025746  117716  153144                 42860           MOVB    @$ERRPC,(SP)    ;STRIP AND SAVE THE ERROR ITEM CODE
 6015  025752  121627  000100                 42870           CMPB    (SP),#100       ;FORM THE CO4$ECT ITEMB IF THIS IS AN
 6016  025756  002402                         42880           BLT     4$              ;EROR MESAGE EQUAL OR ABOVE 100.
 6017  025760  162716  000040                 42890           SUB     #40,(SP)        ;NOTE THERE R 2 CLASSES OF ERRORS:
 6018                                          42900                                   ;1) $ITEMB'S BELOW 100
 6019                                          42910                                   ;2) $ITEMB'S ABOVE 100
 6020                                          42920                                   ;SUBTRACTION FACTOR HAS TOBE SUCH THAT
 6021                                          42930                                   ;THE CO4$ECT OFFSET IS SELECTED. THIS
 6022                                          42940                                   ;FACTOR WILL CHANGE IF THE TOTAL # OF
 6023                                          42950                                   ;ERROR MESSAGES IN CLASS 1 CHANGES.
 6024                                          42960                                   ;#=100 -LAST ITEM IN # CLASS 1 - 1
 6025  025764  012637  001114                 42970   4$:     MOV     (SP)+,$ITEMB
```

```
 6026  025770  032777  020000  153142  42980          BIT     #SW13,@SWR      ;SKIP TYPEOUT IF SET
 6027  025776  001012                  42990          BNE     5$              ;SKIP TYPEOUTS
 6028  026000  004737  026104          43000          JSR     PC,@#$ERRTYP    ;GO TO USER ERROR ROUTINE
 6029  026004  104400  001213          43010          TYPE    ,$CRLF
 6030                                   43020
 6031  026010  032777  010000  153122  43030          BIT     #SW12,@SWR      ;TYPE ERROR HISTORY?
 6032  026016  001402                  43040          BEQ     5$              ;NO
 6033  026020  004737  020120          43050          JSR     PC,HISTRY       ;YES
 6034                                   43060
 6035  026024  005777  153110          43070  5$:     TST     @SWR            ;HALT ON ERROR
 6036  026030  100001                  43080          BPL     6$              ;SKIP IF CONTINUE
 6037  026032  000000                  43090          HALT                    ;HALT ON ERROR!
 6038  026034  032777  001000  153076  43100  6$:     BIT     #SW09,@SWR      ;LOOP ON ERROR SWITCH SET?
 6039  026042  001411                  43110          BEQ     7$              ;BR IF NO
 6040  026044  123727  001114  000040  43120          CMPB    $ITEMB,#40      ;THERE R 37 ERROR MESSAGES IN CLASS 1
 6041  026052  103011                  43130          BHIS    8$
 6042  026054  013746  001744          43140          MOV     PPRLVL,-(SP)    ;LOCK OUT ALL INTERUPTS ON RETURN
 6043                                   43150                                  ;FROM THIS EROR HANDLER, IF THE EROR
 6044                                   43160                                  ;IS IN EXERCISER & LOOPING IS TO
 6045                                   43170                                  ;BE DONE
 6046  026060  005726                  43180          TST     (SP)+
 6047  026062  012716  015374          43190          MOV     #EXCRLUP,(SP)   ;IF THIS ERROR CALL WAS FROM EXERCISER
 6048                                   43200                                  ;PART OF THE PROGRAM, GO TO 'EXCRLUP'
 6049                                   43210                                  ;OTHERWISE RETURN THRU '$LUPERR'
 6050                                   43220
 6051  026066  012737  177777  001750  43230  7$:     MOV     #-1,SRDRV       ;RESET SERIAL NO FLAG
 6052                                   43240                                  ;IF (SRDRV)=-1, THEN THE SERIAL
 6053                                   43250                                  ;NO OF THE DRIVE WILL NOT BE
 6054                                   43260                                  ;TYPED OUT.
 6055                                   43270                                  ;OTHERWISE, SERIAL NO FOR THE DRIVE
 6056                                   43280                                  ;# IN 'SRDRV' WILL BE TYPED.
 6057  026074  000002                  43290          RTI
 6058  026076  013716  001110          43300  8$:     MOV     $LPERR,(SP)     ;FUDGE RETURN FOR LOOPING
 6059  026102  000771                  43310          BR      7$              ;RETURN
```

```
 6060                                          43330   .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
 6061                                          43340
 6062                                          43350   ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
 6063                                          43360   ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
 6064                                          43370   ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
 6065                                          43380
 6066                                          43390
 6067  026104  104400  001213                 43400   $ERRTYP:TYPE    ,$CRLF          ;"CARRIAGE RETURN" & "LINE FEED"
 6068  026110  010046                         43410           MOV     R0,-(SP)        ;SAVE R0
 6069  026112  005000                         43420           CLR     R0              ;PICKUP THE ITEM INDEX
 6070  026114  153700  001114                 43430           BISB    @#$ITEMB,R0
 6071  026120  001004                         43440           BNE     1$              ;IF ITEM NUMBER IS ZERO, JUST
 6072  026122  013746  001116                 43450           MOV     $ERRPC,-(SP)    ;TYPE THE PC OF THE ERROR
 6073  026126  104401                         43460           TYPOC
 6074  026130  000435                         43470           BR      5$              ;GET OUT
 6075  026132  005300                         43480   1$:     DEC     R0              ;ADJUST THE INDEX SO THAT IT WILL
 6076  026134  006300                         43490           ASL     R0              ;WORK FOR THE ERROR TABLE
 6077  026136  006300                         43500           ASL     R0
 6078  026140  006300                         43510           ASL     R0
 6079  026142  062700  001216                 43520           ADD     #$ERRTB,R0      ;FORM TABLE POINTER
 6080  026146  012037  026156                 43530           MOV     (R0)+,2$        ;PICKUP "ERROR MESSAGE" POINTER
 6081  026152  001414                         43540           BEQ     4$              ;SKIP TYPEOUT IF NO POINTER
 6082  026154  104400                         43550           TYPE                    ;TYPE THE "ERROR MESSAGE"
 6083  026156  000000                         43560   2$:     .WORD   0               ;"ERROR MESSAGE" POINTER GOES HERE
 6084  026160  123727  001114  000040         43570           CMPB    $ITEMB,#40      ;SKIP TIME & SERIAL # FOR
 6085                                          43580                                   ;NON-EXERCISER ERORS
 6086  026166  103004                         43590           BHIS    3$
 6087  026170  004737  026272                 43600           JSR     PC,TIMTYP       ;TYPE OUT TIME IF SW 3 IS SET
 6088                                          43610
 6089  026174  004737  026374                 43620           JSR     PC,SNOTYP       ;GO TYPE OUT THE SERIAL # OF THE
 6090                                          43630                                   ;ERRORING DRIVE, IF SW 1 IS SET.
 6091                                          43640
 6092  026200  104400  001213                 43650   3$:     TYPE    ,$CRLF          ;"CARRIAGE RETURN" & "LINE FEED"
 6093  026204  032777  004000  152726         43660   4$:     BIT     #SW11,@SWR      ;DUMP OUT RK REGISTERS?
 6094  026212  001404                         43670           BEQ     5$
 6095  026214  004737  026440                 43680           JSR     PC,DMPREG       ;GO TYPE OUT RK REGISTERS
 6096  026220  104400  001213                 43690           TYPE    ,$CRLF
 6097                                          43700
 6098  026224  012037  026234                 43710   5$:     MOV     (R0)+,6$        ;PICKUP "DATA HEADER" POINTER
 6099  026230  001404                         43720           BEQ     7$              ;SKIP TYPEOUT IF 0
 6100  026232  104400                         43730           TYPE                    ;TYPE THE "DATA HEADER"
 6101  026234  000000                         43740   6$:     .WORD   0               ;"DATA HEADER" POINTER GOES HERE
 6102  026236  104400  001213                 43750           TYPE    ,$CRLF          ;"CARRIAGE RETURN" & "LINE FEED"
 6103  026242  011000                         43760   7$:     MOV     (R0),R0         ;PICKUP "DATA TABLE" POINTER
 6104  026244  001406                         43770           BEQ     9$
 6105                                          43780
 6106                                          43790                                   ;TYPE AN OCTAL NUMBER
 6107  026246  013046                         43800   8$:     MOV     @(R0)+,-(SP)    ;SAVE @(R0)+ FOR TYPEOUT
 6108  026250  104401                         43810           TYPOC                   ;GO TYPE--OCTAL ASCII(ALL DIGITS)
 6109  026252  104400  003455                 43820           TYPE    ,BLNKS2         ;TYPE TWO(2) SPACES
 6110  026256  005710                         43830           TST     (R0)            ;IS THERE ANOTHER NUMBER?
 6111  026260  001372                         43840           BNE     8$              ;BR IF YES
 6112                                          43850
 6113  026262  012600                         43860   9$:     MOV     (SP)+,R0        ;RESTORE R0
 6114  026264  104400  001213                 43870           TYPE    ,$CRLF          ;"CARRIAGE RETURN" & "LINE FEED"
 6115  026270  000207                         43880           RTS     PC              ;RETURN
```

```
 6116                                          43900   ;TIMTYP
 6117                                          43910   ;THIS ROUTINE TYPES THE TIME IN HOURS:MIN:SECS IF SW 3 IS SET
 6118                                          43920   ;SW 3 SHOULD NOT BE SET IF KW11L IS NOT PRESENT.
 6119                                          43930
 6120  026272  032777  000010  152640         43940   TIMTYP: BIT     #SW3,@SWR       ;IS SW 3 SET?
 6121  026300  001432                         43950           BEQ     4$              ;IF NOT SKIP TYPING TIME
 6122  026302  104400  003454                 43960           TYPE    ,BLNKS3
 6123  026306  104413                         43970           SAVREG                  ;SAVE R0-R4
 6124  026310  012700  002252                 43980           MOV     #KWHR,R0        ;INITIALIZE POINTER
 6125  026314  012001                         43990           MOV     (R0)+,R1
 6126  026316  000404                         44000           BR      2$
 6127  026320  012001                         44010   1$:     MOV     (R0)+,R1        ;TYPE OUT
 6128  026322  001402                         44020           BEQ     2$
 6129  026324  062701  000074                 44030           ADD     #60.,R1
 6130  026330  010137  026370                 44040   2$:     MOV     R1,5$
 6131  026334  012746  026370                 44050           MOV     #5$,-(SP)       ;HOURS:MINS:SECS
 6132  026340  004737  024232                 44060           JSR     PC,@#$DB2D      ;CONVERT TO ASCIZ STRING
 6133  026344  004737  024426                 44070           JSR     PC,@#SUPRSL     ;GO TYPE
 6134  026350  020027  002260                 44080           CMP     R0,#KWSEC+2     ;ALL DONE?
 6135  026354  001403                         44090           BEQ     3$
 6136  026356  104400  003037                 44100           TYPE    ,MSG18
 6137  026362  000756                         44110           BR      1$
 6138  026364  104414                         44120   3$:     RESREG                  ;RESTORE R0-R4
 6139  026366  000207                         44130   4$:     RTS     PC              ;RETURN
 6140  026370  000000  000000                 44140   5$:     .WORD   0,0
```

```
6141                                            44160   ;SNOTYP
6142                                            44170   ;THIS ROUTINE TYPES OUT THE SERIAL NUMBER OF THE ERRORING DRIVE. IF SW 1
6143                                            44180   ;IS SET. NOTE THAT THE SERIAL NUMBER IS TYPED OUT ONLY WHEN THE DRIVE
6144                                            44190   ;CAN BE IDENTIFIED POSITIVELY, AS THE ONE WHICH GAVE THE ERROR. IF THE
6145                                            44200   ;ERROR CANNOT BE ATTRIBUTED TO ANY SPECIFIC DRIVE ( (SRDRV)= -1, THEN
6146                                            44210   ;THE SERIAL NUMBER IS NOT TYPED OUT.
6147                                            44220
6148  026374  032777  000002  152536            44230   SNOTYP: BIT     #SW1,@SWR       ;TYPE OUT SERIAL #?
6149  026402  001415                            44240           BEQ     2$              ;NO
6150  026404  010146                            44250           MOV     R1,-(SP)        ;SAVE R1
6151  026406  013701  001750                    44260           MOV     SRDRV,R1        ;GET ERRORING DRIVE #
6152  026412  006301                            44270           ASL     R1              ;IF (SRDRV)= -1, SKIP (BECAUSE
6153  026414  100407                            44280           BMI     1$              ;THE ERROR WAS NOT ATTRIBUTABLE
6154                                            44290                                   ;TO A SPECIFIC DRIVE)
6155  026416  104400  001213                    44300           TYPE    ,$CRLF
6156  026422  104400  003030                    44310           TYPE    ,MSG17          ;TYPE "SR. NO:"
6157  026426  016146  001766                    44320           MOV     SRNO(R1),-(SP)  ;GET THE SERIAL #
6158  026432  104404                            44330           TYPDS                   ;TYPE IT OUT (DECIMAL)
6159                                            44340
6160  026434  012601                            44350   1$:     MOV     (SP)+,R1        ;RESTORE R1
6161  026436  000207                            44360   2$:     RTS     PC              ;RETURN
6162                                            44370
6163                                            44380
6164                                            44390
6165                                            44400   ;DMPREG
6166                                            44410   ;THIS ROUTINE DUMPS OUT ALL RK11 REGISTERS WHEN SW 11 IS SET AND AN ERROR OCCURS
6167                                            44420
6168  026440                                    44430   DMPREG:
6169  026440  104400  026446                            TYPE    ,65$            ;;TYPE ASCIZ STRING
6170  026444  000441                                    BR      64$             ;;GET OVER THE ASCIZ
6171                                                ;;65$:  .ASCIZ  <15><12>/  PC     RKDS    RKER    RKCS    RKWC    RKBA    RKDA
6172  026550                                        64$:
6173  026550  013746  001116                    44440           MOV     $ERRPC,-(SP)
6174  026554  104401                            44450           TYPOC
6175  026556  104400  003455                    44460           TYPE    ,BLNKS2
6176  026562  010046                            44470           MOV     R0,-(SP)
6177  026564  012700  001716                    44480           MOV     #RKDS,R0
6178  026570  013046                            44490   1$:     MOV     @(R0)+,-(SP)
6179  026572  104401                            44500           TYPOC
6180  026574  104400  003455                    44510           TYPE    ,BLNKS2
6181  026600  020027  001732                    44520           CMP     R0,#RKDB
6182  026604  003771                            44530           BLE     1$
6183  026606  012600                            44540           MOV     (SP)+,R0
6184  026610  000207                            44550           RTS     PC
```

```
                                        .SBTTL  SCOPE HANDLER ROUTINE

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

026612                                  $SCOPE:
026612  104406                                CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
026614  032777  040000  152316          1$:   BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
026622  001047                                BNE     $OVER           ;;YES IF SW14=1
                                        ;*****START OF CODE FOR THE XOR TESTER*****
026624  000416                          $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                        ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
026626  013746  000004                        MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
026632  012737  026652  000004                MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
026640  005737  177060                        TST     @#177060        ;;TIME OUT ON XOR?
026644  012637  000004                        MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
026650  000421                                BR      $SVLAD          ;;GO TO THE NEXT TEST
026652  022626                          5$:   CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
026654  012637  000004                        MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
026660  000407                                BR      7$              ;;LOOP ON THE PRESENT TEST
026662                                  6$:;*****END OF CODE FOR THE XOR TESTER*****
026662  105737  001103                  2$:   TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
026666  001412                                BEQ     $SVLAD          ;;BR IF NO
026670  032777  001000  152242                BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
026676  001404                                BEQ     4$              ;;BR IF NO
026700  013737  001110  001106          7$:   MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
026706  000415                                BR      $OVER
026710  105037  001103                  4$:   CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
026714  105237  001102                  $SVLAD: INCB  $TSTNM          ;;COUNT TEST NUMBERS
026720  011637  001106                        MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
026724  011637  001110                        MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
026730  005037  001204                        CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
026734  112737  000001  001115                MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
026742  013777  001102  152172          $OVER: MOV    $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
026750  013716  001106                        MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
026754  000002                                RTI                     ;;FIXES PS
```

```
                                        .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.

026756  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
026760  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
026764  005740                          TST     -(R0)           ;;BACKUP BY 2
026766  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
026770  006300                          ASL     R0              ;;POSITION FOR INDEXING
026772  016000  027000                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
026776  000200                          RTS     R0              ;;GO TO ROUTINE

                                        .SBTTL  TRAP TABLE

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

                                        ;       ROUTINE
                                        ;       -------
027000                          $TRPAD:
027000  023672                          $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
027002  025432                          $TYPOC  ;;CALL=TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING
027004  025406                          $TYPOS  ;;CALL=TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZE
027006  025446                          $TYPON  ;;CALL=TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST C
027010  023446                          $TYPDS  ;;CALL=TYPDS    TRAP+4(104404)  TYPE DECIMAL NUMBER (WITH SIGN)

027012  022462                          $GTSWR  ;;CALL=GTSWR    TRAP+5(104405)  GET SOFT-SWR SETTING

027014  022412                          $CKSWR  ;;CALL=CKSWR    TRAP+6(104406)  TEST FOR CHANGE IN SOFT-SWR
027016  022674                          $RDCHR  ;;CALL=RDCHR    TRAP+7(104407)  TTY TYPEIN CHARACTER ROUTINE
027020  023014                          $RDLIN  ;;CALL=RDLIN    TRAP+10(104410) TTY TYPEIN STRING ROUTINE
027022  023166                          $RDOCT  ;;CALL=RDOCT    TRAP+11(104411) READ AN OCTAL NUMBER FROM TTY
027024  023270                          $RDDEC  ;;CALL=RDDEC    TRAP+12(104412) READ A DECIMAL NUMBER FROM TTY
027026  025130                          $SAVREG ;;CALL=SAVREG   TRAP+13(104413) SAVE R0-R5 ROUTINE
027030  025166                          $RESREG ;;CALL=RESREG   TRAP+14(104414) RESTORE R0-R5 ROUTINE
027032  022006                          CN.RST  ;;CALL=CON.RESET        TRAP+15(104415) CONTROL RESET ROUTINE
027034  022014                          CN.RDY  ;;CALL=CON.RDY  TRAP+16(104416) WAIT FOR CONTROL READY
027036  021674                          DR.RST  ;;CALL=DRV.RESET        TRAP+17(104417) DRIVE RESET ROUTINE
027040  022152                          TY.MSG  ;;CALL=TYPMSG   TRAP+20(104420) TYPE MESSAGE ROUTINE, SW13
                        44640
```

```
                                        .SBTTL  POWER DOWN AND UP ROUTINES

                                        ;;**********************************************************************
                                        ;POWER DOWN ROUTINE
027042  012737  027206  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
027050  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
027056  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
027060  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
027062  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
027064  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
027066  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
027070  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
027072  017746  152042                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
027076  010637  027212                  MOV     SP,$SAVR6       ;;SAVE SP
027102  012737  027114  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
027110  000000                          HALT
027112  000776                          BR      .-2             ;;HANG UP

                                        ;;**********************************************************************
                                        ;POWER UP ROUTINE
027114  012737  027206  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
027122  013706  027212                  MOV     $SAVR6,SP       ;;GET SP
027126  005037  027212                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
027132  005237  027212          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
027136  001375                          BNE     1$              ;;OF  WORD
027140  012677  151774                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
027144  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
027146  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
027150  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
027152  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
027154  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
027156  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
027160  012737  027042  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
027166  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
027174  104400                          TYPE                    ;REPORT THE POWER FAILURE
027176  027214          $PWRMG: .WORD   $POWER                  ;;POWER FAIL MESSAGE POINTER
027200  012716                  MOV     (PC)+,(SP)              ;;RESTART AT PFSTRT
027202  003460          $PWRAD: .WORD   PFSTRT                  ;;RESTART ADDRESS
027204  000002                  RTI
027206  000000          $ILLUP: HALT                            ;;THE POWER UP SEQUENCE WAS STARTED
027210  000776                  BR      .-2                     ;; BEFORE THE POWER DOWN WAS COMPLETE
027212  000000          $SAVR6: 0                               ;;PUT THE SP HERE
027214  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
027222  000122
                                        .EVEN
```

```
                                                44680   ;ERROR MESSAGES
                                                44690
027224  051105  051117  047440  44700   EM1:    .ASCIZ  /EROR ON WRITE/
027232  020116  051127  052111
027240  000105
027242  052101  046505  052120  44710   EM2:    .ASCIZ  /ATEMPT TO INITIATE FUNCTION ON 'BUSY' DRVE/
027250  052040  020117  047111
027256  052111  040511  042524
027264  043040  047125  052103
027272  047511  020116  047117
027300  023440  052502  054523
027306  020047  051104  042526
027314     000
027315     103  052116  047522  44720   EM3:    .ASCIZ  /CNTROL RDY NOT SET/
027322  020114  042122  020131
027330  047516  020124  042523
027336  000124
027340  051057  053457  051457  44730   EM4:    .ASCIZ  "/R/W/S RDY NOT SET"
027346  051040  054504  047040
027354  052117  051440  052105
027362     000
027363     103  052116  047522  44740   EM5:    .ASCIZ  /CNTROL RDY NOT SET AFTER 1ST INTRUPT ON ISSUING SEEK/
027370  020114  042122  020131
027376  047516  020124  042523
027404  020124  043101  042524
027412  020122  051461  020124
027420  047111  051124  050125
027426  020124  047117  044440
027434  051523  044525  043516
027442  051440  042505  000113
027450  051127  047117  020107  44750   EM6:    .ASCIZ  /WRONG BITS IN RKCS, EXPCT SEEK/
027456  044502  051524  044440
027464  020116  045522  051503
027472  020054  054105  041520
027500  020124  042523  045505
027506     000
027507     047  052502  054523  44760   EM7:    .ASCIZ  /'BUSY' FLAG CLEAR ON INTRUPTING DRVE/
027514  020047  046106  043501
027522  041440  042514  051101
027530  047440  020116  047111
027536  051124  050125  044524
027544  043516  042040  053122
027552  000105
027554  050047  051517  052111  44770   EM10:   .ASCIZ  /'POSITIONING' FLAG FOR INTRUPTING DRVE CLEAR/
027562  047511  044516  043516
027570  020047  046106  043501
027576  043040  051117  044440
027604  052116  052522  052120
027612  047111  020107  051104
027620  042526  041440  042514
027626  051101     000
027631     047  051105  023522  44780   EM11:   .ASCIZ  /'ERR'OR SET AFTER 1ST INTERRUPT ON ISSUING SEEK/
027636  051117  051440  052105
027644  040440  052106  051105
027652  030440  052123  044440
027660  052116  051105  052522
```

```
6373  027666  052120  047440  020116
6374  027674  051511  052523  047111
6375  027702  020107  042523  045505
6376  027710     000
6377  027711     123  050103  051440  44790   EM12:   .ASCIZ  /SCP SET AFTER 1ST INTRUPT ON ISSUING SEEK/
6378  027716  052105  040440  052106
6379  027724  051105  030440  052123
6380  027732  044440  052116  052522
6381  027740  052120  047440  020116
6382  027746  051511  052523  047111
6383  027754  020107  042523  045505
6384  027762     000
6385  027763     103  052116  047522  44800   EM13:   .ASCIZ  /CNTROL RDY NOT SET AFTER SEEK DONE INTRUPT/
6386  027770  020114  042122  020131
6387  027776  047516  020124  042523
6388  030004  020124  043101  042524
6389  030012  020122  042523  045505
6390  030020  042040  047117  020105
6391  030026  047111  051124  050125
6392  030034  000124
6393  030036  047111  051124  050125  44810   EM14:   .ASCIZ  /INTRUPTING DRVE (SEEK DONE) WAS NOT 'BUSY'/
6394  030044  [illegible]524  043516  042040
6395  030052  053122  020105  051450
6396  030060  042505  020113  047504
6397  030066  042516  020051  040527
6398  030074  020123  047516  020124
6399  030102  041047  051525  023531
6400  030110     000
6401  030111     122  053457  051457  44820   EM15:   .ASCIZ  "R/W/S READY NOT SET FOR INTRUPTING DRVE (SEEK DONE)"
6402  030116  051040  040505  054504
6403  030124  047040  052117  051440
6404  030132  052105  043040  051117
6405  030140  044440  052116  052522
6406  030146  052120  047111  020107
6407  030154  051104  042526  024040
6408  030162  042523  045505  042040
6409  030170  047117  024505     000
6410  030175     123  047111  042440  44830   EM16:   .ASCIZ  /SIN EROR/
6411  030202  047522  000122
6412  030206  042447  051122  047447  44840   EM17:   .ASCIZ  /'ERR'OR ON DOING SEEK/
6413  030214  020122  047117  042040
6414  030222  044517  043516  051440
6415  030230  042505  000113
6416  030234  041523  020120  044504  44850   EM20:   .ASCIZ  /SCP DID NOT SET AFTER SEEK WAS DONE/
6417  030242  020104  047516  020124
6418  030250  042523  020124  043101
6419  030256  042524  020122  042523
6420  030264  045505  053440  051501
6421  030272  042040  047117  000105
6422  030300  047523  052106  042440  44860   EM21:   .ASCIZ  /SOFT EROR/
6423  030306  047522  000122
6424  030312  040504  040524  024040  44870   EM23:   .ASCIZ  /DATA (COMPARISON) EROR/
6425  030320  047503  050115  051101
6426  030326  051511  047117  020051
6427  030334  051105  051117     000
6428  030341     103  052116  047522  44880   EM24:   .ASCIZ  /CNTROL RDY CLR ON INTRUPT AFTER RK FUNCTION/
```

```
 6429  030346  020114  042122  020131
 6430  030354  046103  020122  047117
 6431  030362  044440  052116  052522
 6432  030370  052120  040440  052106
 6433  030376  051105  051040  020113
 6434  030404  052506  041516  044524
 6435  030412  047117     000
 6436  030415     123  052524  045523  44890   EM26:   .ASCIZ  /STUCK IN LOOP,8 COMANDS SHLDBE DONE BY NOW/
 6437  030422  044440  020116  047514
 6438  030430  050117  034054  041440
 6439  030436  046517  047101  051504
 6440  030444  051440  046110  041104
 6441  030452  020105  047504  042516
 6442  030460  041040  020131  047516
 6443  030466  000127
 6444  030470  052101  050115  020124  44900   EM27:   .ASCIZ  /ATMPT TO DO WRITE BEFORE WRT CHK/
 6445  030476  047524  042040  020117
 6446  030504  051127  052111  020105
 6447  030512  042502  047506  042522
 6448  030520  053440  052122  041440
 6449  030526  045510     000
 6450  030531     101  046524  052120  44910   EM30:   .ASCIZ  /ATMPT TO REEXECUTE COMMAND-IN PROGRESS OR ALREADY FINISHED/
 6451  030536  052040  020117  042522
 6452  030544  054105  041505  052125
 6453  030552  020105  047503  046515
 6454  030560  047101  026504  047111
 6455  030566  050040  047522  051107
 6456  030574  051505  020123  051117
 6457  030602  040440  051114  040505
 6458  030610  054504  043040  047111
 6459  030616  051511  042510  000104
 6460  030624  043047  047125  052103  44920   EM31:   .ASCIZ  /'FUNCTION IN PROGRES' FLG FOR INTRUPTING DRIVE ISN'T SET/
 6461  030632  047511  020116  047111
 6462  030640  050040  047522  051107
 6463  030646  051505  020047  046106
 6464  030654  020107  047506  020122
 6465  030662  047111  051124  050125
 6466  030670  044524  043516  042040
 6467  030676  044522  042526  044440
 6468  030704  047123  052047  051440
 6469  030712  052105     000
 6470  030715     125  042516  050130  44930   EM32:   .ASCIZ  /UNEXPCTED DRIVE INTRUPTED/
 6471  030722  052103  042105  042040
 6472  030730  044522  042526  044440
 6473  030736  052116  052522  052120
 6474  030744  042105     000
 6475  030747     125  042516  050130  44940   EM33:   .ASCIZ  /UNEXPCTD FUNCTION CODE IN RKCS AFTER INTRUPT/
 6476  030754  052103  020104  052506
 6477  030762  041516  044524  047117
 6478  030770  041440  042117  020105
 6479  030776  047111  051040  041513
 6480  031004  020123  043101  042524
 6481  031012  020122  047111  051124
 6482  031020  050125  000124
 6483  031024  051104  042526  051040  44950   EM34:   .ASCIZ  /DRVE RDY CLEAR/
 6484  031032  054504  041440  042514
```

```
6485  031040  051101  000
6486  031043  104  053122  020105  44960   EM35:   .ASCIZ  /DRVE POWER LO/
6487  031050  047523  042527  020122
6488  031056  047514  000
6489  031061  104  053122  020105  44970   EM36:   .ASCIZ  /DRVE UNSAFE/
6490  031066  047125  040523  042506
6491  031074  000
6492  031075  127  051520  051440  44980   EM37:   .ASCIZ  /WPS SET/
6493  031102  052105  000
6494  031105  111  052116  051105  44990   EM101:  .ASCIZ  /INTERUPT DIDN'T OCUR AFTER WRTE/
6495  031112  050125  020124  044504
6496  031120  047104  052047  047440
6497  031126  052503  020122  043101
6498  031134  042524  020122  051127
6499  031142  042524  000
6500  031145  047  051105  023522  45000   EM102:  .ASCIZ  /'ERR'OR SET/
6501  031152  051117  051440  052105
6502  031160  000
6503  031161  122  042113  020101  45010   EM103:  .ASCIZ  /RKDA INCRMENTED WRONG/
6504  031166  047111  051103  042515
6505  031174  052116  042105  053440
6506  031202  047522  043516  000
6507  031207  122  041113  020101  45020   EM104:  .ASCIZ  /RKBA INCRMENTED WRONG/
6508  031214  047111  051103  042515
6509  031222  052116  042105  053440
6510  031230  047522  043516  000
6511  031235  122  053513  020103  45030   EM105:  .ASCIZ  /RKWC DIDN'T OVRFLO TO 0/
6512  031242  044504  047104  052047
6513  031250  047440  051126  046106
6514  031256  020117  047524  030040
6515  031264  000
6516  031265  115  054105  041040  45040   EM106:  .ASCIZ  /MEX BITS WRONG/
6517  031272  052111  020123  051127
6518  031300  047117  000107
6519  031304  051127  042524  041440  45050   EM110:  .ASCIZ  /WRTE CHK EROR/
6520  031312  045510  042440  047522
6521  031320  000122
```

```
                                                45070   ;DATA HEADERS
031322  020040  041520  020040                45080   DH1:    .ASCIZ  / PC    RKCS    RKER    RKDS    RKDA/
031330  020040  051040  041513
031336  020123  020040  051040
031344  042513  020122  020040
031352  051040  042113  020123
031360  020040  051040  042113
031366  000101
                                                45090
031370  020040  041520  020040                45100   DH2:    .ASCIZ  / PC    DRV#/
031376  020040  051104  021526
031404      000
                                                45110
031405      040 050040  020103                45120   DH21:   .ASCIZ  / PC    RKCS    RKER    RKDS    DRIVE   CYL     SUR     SEC/
031412  020040  020040  045522
031420  051503  020040  020040
031426  045522  051105  020040
031434  020040  045522  051504
031442  020040  042040  044522
031450  042526  020040  020040
031456  054503  020114  020040
031464  020040  052523  020122
031472  020040  020040  042523
031500  000103
                                                45130
031502  020040  041520  020040                45140   DH23:   .ASCIZ  / PC    RKBA    EXPCT   RECVD   RKDA/
031510  020040  045522  040502
031516  020040  020040  054105
031524  041520  020124  020040
031532  042522  053103  020104
031540  020040  045522  040504
031546      000
031547      040 050040  020103                45150   DH25:   .ASCIZ  / PC    RKCS    RKER    RKDS    RKDA    DRVE#/
031554  020040  020040  045522
031562  051503  020040  051040
031570  042513  020122  020040
031576  051040  042113  020123
031604  020040  051040  042113
031612  020101  020040  051104
031620  042526  000043
                                                45160
031624  020040  041520  020040                45170   DH27:   .ASCIZ  / PC    KEY     FNCTN CODE/
031632  020040  045440  054505
031640  020040  020040  047106
031646  052103  020116  047503
031654  042504      000
031657      040 050040  020103                45180   DH103:  .ASCIZ  / PC    EXPCT   RECVD/
031664  020040  042440  050130
031672  052103  020040  051040
031700  041505  042126      000
                                                45190
031705      040 050040  020103                45200   DH30:   .ASCIZ  / PC    KEY/
031712  020040  045440  054505
031720      000
031721      040 050040  020103                45210   DH105:  .ASCIZ  / PC    RKDA    RKWC/
031726  020040  020040  045522
```

```
  6578  031734  040504  020040  051040
  6579  031742  053513  000103
  6580  031746  020040  041520  020040  45220   DH110:  .ASCIZ  / PC     RKCS    RVER    RKBA    PKDA/
  6581  031754  020040  051040  041513
  6582  031762  020123  020040  051040
  6583  031770  042513  020122  020040
  6584  031776  051040  041113  020101
  6585  032004  020040  051040  042113
  6586  032012  000101
  6587                                  45230
  6588                                  45240
  6589                                  45250           .EVEN
  6590                                  45260
  6591  032014  001116  001162  001164  45270   DT1:    .WORD   $ERRPC,$REG0,$REG1,$REG2,$REG3,0
  6592  032022  001166  001170  000000
  6593                                  45280
  6594  032030  001116  001162  000000  45290   DT2:    .WORD   $ERRPC,$REG0,0
  6595                                  45300
  6596  032036  001116  001162  001164  45310   DT21:   .WORD   $ERRPC,$REG0,$REG1,$REG2,$REG3,$REG4,$REG5,$REG6,0
  6597  032044  001166  001170  001172
  6598  032052  001174  001176  000000
  6599  032060  001116  001162  001164  45320   DT25:   .WORD   $ERRPC,$REG0,$REG1,$REG2,$REG3,$REG4,0
  6600  032066  001166  001170  001172
  6601  032074  000000
  6602  032076  001116  001162  001164  45330   DT103:  .WORD   $ERRPC,$REG0,$REG1,0
  6603  032104  000000
  6604                                  45340
  6605                                  45350   ;THIS IS THE DATA BUFFER USED TO WRITE THE RANDOM PATTERNS ON THE
  6606                                  45360   ;DISK AT THE BEGINING. 400 (OCTAL) WORDS ARE WRITTEN AT A TIME, THUS
  6607                                  45370   ;THIS BUFFER IS 400/8 WORDS LONG.
  6608                                  45380
  6609  032106                          45390   DBUF:
  6610  032106  000240                 45400   PGEND:  NOP
  6611           000001                 45410           .END
```

```
ABORT   002372          1415#  3307*  3584*  3585   4783
ABRT    011566          2980   2990   3007   3035#
ABRT1   011600          2966   2972   3038#
ACT11   002563          1455#  1640*
BAMAP   002230          1354#  2623*  3775
BASEBA  002552          1439#  1754   2611   2654   3781
BCTST   005114          1829   1835#
BEGEX1  010334          2637   2705#
BEGNEX  010352          2706   2713#  4095   4895
BIT0  = 000001           714#  3438   3440   3442
BIT00 = 000001           704#   714    747
BIT01 = 000002           703#   713    746
BIT02 = 000004           702#   712
BIT03 = 000010           701#   711
BIT04 = 000020           700#   710
BIT05 = 000040           699#   709    741
BIT06 = 000100           698#   708    740
BIT07 = 000200           697#   707    739
BIT08 = 000400           696#   706
BIT09 = 001000           695#   705    738   6215
BIT1  = 000002           713#  3450   3452   3454
BIT10 = 002000           694#   737
BIT11 = 004000           693#  4860
BIT12 = 010000           692#   736    745   2749   3209   3300   3308   3312   3420   3514   3542   3581
BIT13 = 020000           691#   733
BIT14 = 040000           690#   732   2515   4578   4612   6199
BIT15 = 100000           689#   731   2788   3556   3572   4572   4578   4583   4860
BIT2  = 000004           712#  3479   3481   3483
BIT3  = 000010           711#  3096   3359   3493   3495   3497
BIT4  = 000020           710#  2866   2893   2918   3099   3170   3204   3243   3285   3321   3421   3515   3725
                        4505
BIT5  = 000040           709#  3192   3194   3196   3554   3851
BIT6  = 000100           708#  2709   2794
BIT7  = 000200           707#  3027   3028   3052   3053   3095   3101   3401   3402   4583
BIT8  = 000400           706#  2683   4572
BIT9  = 001000           705#
BLNKS1  003456          1547#  4747   4752   4760   4764   4768   4773   4777   4782   4787
BLNKS2  003455          1546#  6109   6175   6180
BLNKS3  003454          1545#  4738   4884   6122
BPTVEC= 000014           721#
BUSY    002126          1299#  2821   2839   2874   2926   2976   3031*  3055*  3062   3094*  3095*  3157   3166
                        3205*  3230   3240   3278*  3287*  3297*  3323*  3344   3356   3402*  3906   3924   3941
                        3946*  4879
CBSY    021124          4831#  4839   4841
CHDPRS  022304          2592   3624   4063   5116#
CHFAFN  010676          2745   2765   2801#  2907
CHKBA   017642          1909   2047   2209   2346   2491   4487#
CHKCRD  017764          3121   3222   3331   4538#
CHKCS   017604          1867   1904   2004   2043   2192   2251   2329   2372   2482   2512   2678   4463#
CHKDA   017620          1869   2006   2216   2486   4474#
CHKDRV  016204          3036   3109   3259   3403   4109#
CHKMEX  017676          2197   2338   4503#
CHKRWS  017746          4527#
CHKWC   017722          1872   2010   2222   2350   2496   4514#
CH1     010732          2815#  2855
CH2     011032          2816   2818   2822   2826   2853#  2894
```

F13


```
CICNT   002166          1328#  2108*  2110*  4828*  4838*
CICNT1  002170          1329#  2109*  2113*  4829*  4840*
CKSWR = 104406          3601   6198   6260#
CLEANB  006176          1896   2034   2140#
CLRERR  015454          3208   3320   3516   3883   3919#  3969   4893
CLRFLG  015436          3639   3891   3906#
CLRSIN  015620          3288   3412   3968#
CMNABT  013736          3471   3482   3496   3512   3519#
CMND    002026          1289#  1429   1430   1431   1432   1433   1434   1435   1436   3632
CMNERR  013700          3470   3480   3494   3510#
CNOBSY  021342          4826   4897#
CN.RDY  022014          5036#  6268
CN.RST  022006          5035#  6267
COMRET  020000          4464   4475   4493   4506   4515   4528   4539   4543#
CON.RD= 104416          1658   1711   2190   2249   2327   2370   2480   2510   2677   3977   5010   6268#
CON.RE= 104415          1641   1859   1861   1891   1893   1993   1995   2030   2032   2176   2232   2313   2357
                        2462   2501   2553   2713   3525   3955   3996   6267#
CR    = 000015           629#  5517   5527
CRCMND  020032          3954   3995   4578#
CRLF  = 000200           630#  1607   5488   5527
CROTLF  021620          3227   3341   4234   4424   4574   4977#  4999
CSE   = 000002           746#  3447   4319
CSECN   002352          1408#  3456*  4761
CYLMAP  002222          1351#  2606*  3700
DATCHK  016554          3468   3548   4226#
DATER   002412          1419#  4238   4769
DBUF    032106          1862*  1907   1926   1948   1965   1998*  2045   2064   2081   2102   2141   2672   2674
                        6609#
DDISP = 177570           636#   869   1580
DH1     031322           921    935    942    949    957    978    985    992    999   1006   1013   1020   1027
                        1059   1073   1115   1122   1129   1136   1154   1161   1189   6523#
DH103   031657          1101   1108   1168   1175   1196   6568#
DH105   031721          1182   6576#
DH110   031746          1203   6580#
DH2     031370           928    964    971   1094   6531#
DH21    031405          1034   1043   6535#
DH23    031502          1052   1147   6547#
DH25    031547          1066   6554#
DH27    031624          1080   6563#
DH30    031705          1087   6573#
DISPLA  001142           869#  1580*  1588*  5990*  6225*
DISPRE  000174           756#  1588
DMPREG  026440          6095   6168#
DOREAD  006166          1900   2040   2131#
DOWRIT  006046          1864   2000   2098#
DOXFER  006054          2099#  2132
DPL   = 010000           736#  4117
DRCMND  020006          3973   4572#
DRMAP   002220          1350#  2605*  3655   4073*
DRU   = 002000           737#  4122
DRVABT  014122          3035   3108   3214   3424   3521   3578#
DRVCNT  002200          1335#  1836*  1839*
DRVPRS  001764          1241#  1642*  1661*  1669   1836   2601   2641   2715   2730   3643   3660   4061*  4069
                        4728   5116
DRVPTR  002176          1334#  1835*  1837   1838*
DRV.RE= 104417          1696   1860   1892   1994   2031   2177   2233   2314   2358   2463   2502   3526   6269#
```

```
DRY    = 000200            739*  4134
DR.RST   021674           5007*  6269
DSELCT   015760            3587   4016   4028*  4092   4148   6007
DSWR   = 177570             635*   868   1579
DT1      032014             922    936    943    950    958    979    986    993   1000   1007   1014   1021   1028
                           1053   1060   1074   1116   1123   1130   1137   1148   1155   1162   1190   1204   6591*
DT103    032076            1081   1102   1109   1169   1176   1183   1197   6602*
DT2      032030             929    965    972   1088   1095   6594*
DT21     032036            1036   1045   6596*
DT25     032060            1067   6599*
ECOUNT   002240            1366*  4240*  4289   4302*
EMTVEC=  000030             724*  1608*  1609*
EM1      027224             920   6319*
EM10     027554             970   6360*
EM101    031105            1153   6494*
EM102    031145            1160   6500*
EM103    031161            1167   6503*
EM104    031207            1174   6507*
EM105    031235            1181   6511*
EM106    031265            1188   6516*
EM11     027631             977   6368*
EM110    031304            1202   6519*
EM12     027711             984   6377*
EM13     027763             991   6385*
EM14     030036             998   6393*
EM15     030111            1005   6401*
EM16     030175            1012   6410*
EM17     030206            1019   6412*
EM2      027242             927   6322*
EM20     030234            1026   6416*
EM21     030300            1033   6422*
EM23     030312            1051   1146   1195   6424*
EM24     030341            1058   6428*
EM26     030415            1072   6436*
EM27     030470            1079   6444*
EM3      027315             934   6330*
EM30     030531            1086   6450*
EM31     030624            1093   6460*
EM32     030715            1100   6470*
EM33     030747            1107   6475*
EM34     031024            1114   6483*
EM35     031043            1121   6486*
EM36     031061            1128   6489*
EM37     031075            1135   6492*
EM4      027340             941   5022   6334*
EM5      027363             948   6338*
EM6      027450             956   6347*
EM7      027507             963   6353*
ERCODE   002174            1333*  3192   3194*  3196*  3216*  3438   3440*  3442*  3450   3452*  3454*  3479   3481*
                           3483*  3493   3495*  3497*  3519*  3540*
ERDRV    002242            1375*  6004*  6005
ERINF1   005462            1916   1932   1962*  2053   2069
ERR    = 100000             731*  3429
ERRVEC=  000004             717*  1577   1578*  1589*  4349   4350   4356*  4364*  4376*  4381*  4390*  4400*  4401*
                           6204   6205*  6207*  6210*
EXCRLU   015374            3883*  6047
```

```
EXCUT   011332          2796    2876    2888    2933    2957#
EXNSK   011212          2850    2917#
EXRCSR  007534          1830    1841    2553#
FNCMND  020064          3023    3049    4586#
FNMAP   002226          1353#   2608*   3685
FRSTRT  001753           762*   1234#   1552*   1721    1828    2558    2632    2634*
FTITLE  001752          1233#
GENBUF  016350          2670    3043    4161#
GEN1    014372          3643#   3673    3871
GENBRQ  014174          2717    2744    3601#
GETINF  021514          3460    3506    4943#
GNS     = ****** U       755    1606    1790    1796    1803    1815    2689    2734    3607    4617    4624    4631    4644
                        4650    4656    5053    6171    6252    6253    6254    6255    6256    6258    6260    6261    6262
                        6263    6264    6265    6266    6267    6268    6269    6270
GTSDRV  021656          3461    3507    4294    4998#
GTSWR   = 104405        1601    6258#
GT3RG   021470          4532#   4943
GT4RG   021462          2116    2520    2988    3005    3073    3087    3142    3232    3268    3282    3315    4465    4507
                        4529    4540    4857    4865    4931#   4990
HE      = 040000         732#   3275    3427
HECN    002262          1386#   3486*   3499*   4774
HISTRY  020120          4601#   6033
HRDERR  013536          3428    3476#
HT      = 000011         627#   5486    5527
INTFLG  002234          1362#   3032*   3056*   3328*   4831    4845
INTHND  012776          2906    2952    3328#
INT1FL  002235          1363#   2904    2951    3105*   3117*
INT1SK  012116          3103    3117#
IOTVEC= 000020           722#   1567*   1568*
KDPAR0= 172360           831#
KDPAR1= 172362           832#
KDPAR2= 172364           833#
KDPAR3= 172366           834#
KDPAR4= 172370           835#
KDPAR5= 172372           836#
KDPAR6= 172374           837#
KDPAR7= 172376           838#
KDPDR0= 172320           809#
KDPDR1= 172322           810#
KDPDR2= 172324           811#
KDPDR3= 172326           812#
KDPDR4= 172330           813#
KDPDR5= 172332           814#
KDPDR6= 172334           815#
KDPDR7= 172336           816#
KEY     002006          1275#   1650    1835    2554    2739    2742    2748    2757    2771    2814    2847    2854    2865
                        2879    2917    2930    3168    3242    3358    3625    3868    3886    3888    3943    4075    4082
                        4588    4848    4878
KIPAR0= 172340           820#   4335    4368    4428
KIPAR1= 172342           821#
KIPAR2= 172344           822#
KIPAR3= 172346           823#
KIPAR4= 172350           824#
KIPAR5= 172352           825#   2164*   2306*
KIPAR6= 172354           826#   2165*
KIPAR7= 172356           827#   4384
```

```
KIPDR0= 172300            798#
KIPDR1= 172302            799#
KIPDR2= 172304            800#
KIPDR3= 172306            801#
KIPDR4= 172310            802#
KIPDR5= 172312            803#
KIPDR6= 172314            804#
KIPDR7= 172316            805#
KWCOUN  002260           1380#   5081#   5084#
KWHR    002252           1377#   2556    5092#   6124
KWLS    001734           1213#   2709#
KWLVEC= 000100            729#   1610#   1611#
KWMIN   002254           1378#   2566    5089#   5091#
KWPLVL  001746           1223#   1611
KWSEC   002256           1379#   5085#   5088#   6134
KWSRVE  022202           1610    5081#
LF    = 000012            628#   5521    5527
MAXBA   002554           1441#   1755    1825    1827#   2427    2610    2653    3784
MH1     020440           4605    4664    4672#
MH2     020454           4607    4666    4674#
MH3     020464           4606    4676#
MH4     020467           4665    4677#
MMVEC = 000250            787#
MSG1    002564           1459#   3488
MSG10   002667           1473#   3309    3589
MSG11   002703           1476#   4926
MSG12   002710           1477#   5023    5047
MSG13   002716           1478#   2281    2389
MSG14   002727           1480#   5119
MSG15   002747           1483#   4853
MSG16   003007           1489#   4877
MSG17   003030           1492#   1727    6156
MSG18   003037           1494#   6136
MSG19   003041           1495#   4054
MSG2    002572           1460#   3445
MSG20   003065           1499#   1673
MSG21   003075           1501#   1629
MSG23   003166           1511#   1636
MSG24   003207           1514#   1681
MSG25   003212           1515#   1716
MSG26   003214           1516#   4727
MSG27   003334           1530#   3520
MSG28   003406           1537#   3534
MSG3    002600           1461#   3457
MSG4    002606           1462#   3500
MSG5    002622           1464#   3458    3502
MSG6    002635           1466#   4914
MSG7    002643           1468#   4917
MSG8    002650           1469#   4923
MSG9    002660           1471#   4920
NIEROR  021146           4845#
NOEROR  013774           3431    3532#
NRDH    002474           1426#   4717*
NRDL    002472           1425#   4713*   4754
NWRFNC  011506           3018#
NWRTH   002434           1421#   4710*
```

```
NWRTL   002432          1420#   4709*   4749
NXTDRV  005130          1837#   2545
ODDEVN  015372          3672    3676    3874*   3875*   3878#
PC     =%000007          648#   1551*   1753*   1864*   1867*   1869*   1872*   1896*   1900*   1904*   1909*   1916*   1932*
                        1975*   2000*   2004*   2006*   2010*   2034*   2040*   2043*   2047*   2053*   2069*   2116*   2117*
                        2121*   2147*   2192*   2197*   2209*   2216*   2222*   2251*   2282*   2329*   2338*   2346*   2350*
                        2372*   2390*   2436*   2482*   2486*   2491*   2496*   2512*   2520*   2592*   2602*   2619*   2678*
                        2717*   2744*   2897*   2898*   2942*   2944*   2988*   3005*   3023*   3035*   3036*   3049*   3073*
                        3087*   3106*   3108*   3109*   3121*   3127*   3142*   3198*   3208*   3214*   3222*   3227*   3232*
                        3259*   3268*   3279*   3282*   3288*   3290*   3301*   3310*   3313*   3315*   3320*   3331*   3336*
                        3341*   3403*   3409*   3412*   3413*   3424*   3459*   3460*   3461*   3468*   3503*   3506*   3507*
                        3516*   3521*   3543*   3548*   3590*   3612*   3619*   3624*   3639*   3647*   3656*   3679*   3686*
                        3694*   3701*   3715*   3755*   3761*   3769*   3776*   3804*   3810*   3817*   3876*   3883*   3891*
                        3910*   3931*   3937*   3954*   3958*   3969*   3973*   3983*   3989*   3995*   3997*   4014*   4059*
                        4063*   4070*   4119*   4124*   4130*   4136*   4146*   4186*   4234*   4262*   4294*   4328*   4361
                        4408*   4424*   4446*   4450*   4457*   4465*   4466*   4478*   4495*   4507*   4508*   4518*   4529*
                        4530*   4540*   4541*   4544*   4574*   4593*   4641*   4671*   4722*   4726*   4750*   4751*   4755*
                        4756*   4794*   4856*   4857*   4865*   4893*   4927*   4935*   4943*   4969*   4983*   4990*   4999*
                        5001*   5107*   5118*   5120*   5137*   5140*   5146*   5151    5207*   5495*   5502*   5509*   5523*
                        5525*   5554*   5601*   5606*   5656*   5703*   5879*   6028*   6033*   6087*   6089*   6095*   6115*
                        6132*   6133*   6139*   6161*   6184*   6308
PCMND   002532          1429#   2563    2770    2998    3082    3392    4227    4639    4854
PCNTR   001736          1215#   2587    3604
PDR     001754          1239#   1645*   1646*   1647*   1648*   1649    1670    2640    3664    4028    4034    4042    4047
                        4729
PFSTRT  003460          1551#   6309
PFTERR  012734          3260    3292    3312#
PGEND   032106          1785    2428    2477    2490    2506    6610#
PIRQ   = 177772          634#
PIRQVE= 000240           728#
PORKER  012742          3276    3315#
POS     002136          1305#   2823    3098*   3177    3206*   3249    3286*   3296*   3322*   3355*
POSCMN  020042          2898    2944    4581#
POSK    011704          2897    2942    3060#
POSTIO  011136          2842    2893#
PPRLVL  001744          1220#   1609    2726    2809    3104    3118    3330    4900    6042
PPTP    002562          1454#   1622*
PRICMN  011132          2754    2888#
PRSFNC  002162          1322#   4591    4592*   4604
PR0    = 000000          651#
PR1    = 000040          652#
PR2    = 000100          653#
PR3    = 000140          654#
PR4    = 000200          655#
PR5    = 000240          656#
PR6    = 000300          657#
PR7    = 000340          658#
PS     = 177776          631#    632
PSINER  012610          3273    3282#
PSRTRY  012650          3295#   3324
PSTFNC  002164          1324#   4591*   4661
PSW    = 177776          632#
PWRVEC= 000024           723#   1571*   1572*   6276*   6277*   6286*   6292*   6304*   6305*
P1      020070          4587#
P2      020102          4576    4579    4584    4591#
QBUSAD  002214          1343#   3779*   3780*   3781*   3785    3863
```

```
GCYL    002204          1339#  3704*  3705   3707*  3742   3752   3832
QDRV    002202          1338#  1695*  1837*  1853   1889   1988   2027   2186   2214   2244   2322   2365   2458
                        2464   2503   2645*  2691   3524*  3663*  3664*  3665   3668*  3669   3826   3866   5008
QEROR   021240          4824   4864#
QFNC    002212          1342#  3692*  3845   4572*  4575*  4578*  4581*  4582*  4583*  4586*  4589*  4592
QMNGER  010372          2716   2726#  2789   3038   3112   3894   4904   5152
QSCNT   002160          1317#  3908   4823*  4864*
QSEC    002210          1341#  3718*  3719   3721*  3749   3839
QSUR    002206          1340#  3725*  3733*  3840
QWRCNT  002216          1344#  3820*  3822   3824*  3825*  3861   3968*  3972
RCNT    016550          4188*  4191*  4216#  4264*  4267*
RDCHR = 104407          1630   5266   6261#
RDDEC = 104412          1728   6264#
RDLIN = 104410          5307   5349   6262#
RDOCT = 104411          1805   1817   6263#
REPCNT  002556          1443#  2587*  3602*  3604*
REPSTA  020610          3612   3619   4726#
RESREG= 104414          4212   4327   5553   5605   6138   6266#
RESVEC= 000010           718#
RETRY   002146          1312#  3210*  3211   3215*  3295*  3298   3304*  3417*  3419   3510*  3511   3532   3535
                        3541*  3579*
RG4SDR  021652          3127   3198   3409   3931   3937   3983   3989   4119   4124   4130   4136   4990#
RKBA    001726          1210#  2102*  2188*  2246*  2324*  2367*  2477*  2506*  2521   2536*  2674*  3021*  3048*
                        4492   4497
RKCS    001722          1208#  1657*  1710*  2098*  2107*  2131*  2189*  2247*  2325*  2369*  2478*  2508*  2515
                        2675*  2861   2899*  2945*  2986   3025*  3051*  3071   3125   3138   3189   3275   3334
                        3390   3427   3429   3974*  4463   4503   4538   4699   4825   4932   5009*  5035*  5037
                        5055
RKDA    001730          1211#  1653*  1706*  2099*  2186*  2244*  2322*  2365*  2464*  2503*  2651*  3000*  3083*
                        3149   3264*  3338   3922*  3949*  3968   3972*  4109   4116*  4144*  4166   4321   4474
                        4477   4516   4573   4931   4948   4998   5008*
RKDB    001732          1212#  6181
RKDPCH  002561          1453#  1637*  1651
RKDPDU  002560          1452#  1633*  1777
RKDS    001716          1206#  1654   1707   1712   1718   3002   3084   3226   3264   3266   3272   3407   3927
                        3935   3979   3987   4117   4122   4128   4134   4142   4527   4934   5011   6177
RKER    001720          1207#  2517   3435   3447   3476   3490   3522   4319   4933
RKSTAT  001742          1218#  3104*
RKVEC   001740          1217#  2103*  2906*  2952*  3103*
RKWC    001724          1209#  2100*  2101*  2187*  2245*  2323*  2366*  2466*  2467*  2504*  2505*  2530   2533
                        2538*  2669*  3020*  3047*  4514   4517   4702
RNUM    016552          4189*  4193*  4194   4196   4217#  4265*  4269*  4270   4277   4287
RSBAH   025374          3772   5887#
RSBAL   025372          3770   5886#
RSCYLH  025370          3697   3711   3723   5885#
RSCYLL  025366          3695   5884#
RSDRVH  025360          3652   5881#
RSDRVL  025356          2572   3650   5880#
RSDTH   025404          4167*  4189   4241*  4242   4265   4323*  5891#
RSDTL   025402          4168*  4169*  4187   4193   4242*  4243*  4263   4269   4324*  4325*  5890#
RSFUNH  025364          3682   5883#
RSFUNL  025362          3680   5882#
RSWCH   025400          3813   3849   5889#
RSWCL   025376          3811   5888#
RTIPC7  021362          4901   4904#
RWS   = 000100           740#  3002   3084   3266   3927   3979   4527
```

R0 =%000000
639# 1643* 1659 1664* 1666 1671* 1678* 1729* 1730* 1758* 1759 1762* 1765*
1768* 1774* 1779* 1781* 1783 1806* 1807 1809 1811 1818* 1819 1821 1823
2179* 2182 2183* 2184 2315* 2317 2318* 2319 2554* 2555* 2556 2562* 2563
2566* 2568* 2569* 2570* 2572* 2573* 2574* 2575* 2576* 2577* 2578* 2579* 2580*
2581* 2582* 2583* 2584* 2610* 2611* 2613* 2618 2642* 2663 2665 2685 2697*
2739* 2740 2742 2748* 2749 2751 2753 2756 2757 2757* 2768* 2770 2771*
2776 2788* 2792* 2794* 2801* 2807* 2814* 2815 2817 2819 2829 2853 2854
2865* 2866 2868 2870 2872 2878 2879 2893 2953* 2957 2959 2961 2967
2969 2993 3027* 3052* 3060 3078 3096 3099* 3101* 3138* 3139* 3140 3149*
3150* 3152* 3153* 3154* 3155* 3156 3165* 3167* 3168* 3170 3204* 3209* 3240*
3241* 3242* 3243 3285* 3300* 3303* 3308* 3312* 3321* 3356* 3357* 3358* 3359
3369 3377 3385 3386 3401* 3420* 3421* 3572* 3580* 3581* 3625* 3626 3628*
3632* 3633 3635* 3645* 3659* 3660 3662* 3664 3748* 3749* 3750 3826* 3928*
3829* 3830* 3831* 3838* 3839* 3840* 3841 3906* 3907* 3908 4253* 4257* 4273
4275* 4280 4345 4351* 4399 4407* 4581 4587 4601 4604* 4608 4610 4612
4633 4638 4661 4663 4670* 4686 4696* 4697* 4704* 4709 4713 4721* 4728*
4791* 4845* 4846* 4847 4848* 4849 4860* 4878* 4882 4944 4947* 4951* 4958*
4962* 4965* 4968* 5143* 5146 5304 5308* 5311 5327* 5346 5350* 5351 5354
5356 5357 5381* 5385* 5403 5413* 5417 5433 5434 5447* 5478 5479* 5480
5483* 5543* 5547 5557* 5559* 5561* 5581* 5582 5600* 5604* 5632 5634* 5637
5638* 5639 5641 5643 5645* 5647* 5648* 5651 5654* 5672 5685* 5687* 5688*
5694* 5696* 5698 5702* 5732 5739* 5742* 5744* 5752* 5759* 5760 5763* 5770*
5771 5775* 5782 5786* 5809 5834* 5843 5850* 5854* 5859* 5863* 5869* 5871
5877* 6068 6069* 6070* 6075* 6076* 6077* 6078* 6079* 6080 6098 6103* 6107
6110 6113* 6124* 6125 6127 6134 6176 6177* 6178 6181 6183* 6236 6237*
6238 6239* 6240* 6241* 6242* 6278 6303*

R1 =%000001
640# 1649* 1659* 1670* 1677 1715* 1732* 1763* 1766* 1853* 1854 1875 1877*
1889* 1951 1953* 1964 1988* 1990 2012 2014* 2027* 2028* 2084 2086* 2099
2181* 2182* 2235* 2236* 2237 2265* 2267 2273 2286* 2287 2316* 2317* 2360*
2361* 2362 2379* 2381 2383 2396* 2397 2432* 2439* 2441* 2444* 2448* 2449
2567* 2568 2569 2570 2643* 2644* 2645 2647* 2648* 2649* 2650* 2651 2662*
2665* 2666* 2667* 2668 2669 2770* 2774 2777 2790* 2819* 2820* 2821 2823
2841 2872* 2873* 2874 2917* 2918 2920 2922 2924 2929 2930 2940 2957*
2958* 2976 2978 3004 3031* 3055* 3060* 3061* 3062 3064 3086 3094* 3095*
3098* 3156* 3157 3159 3166 3172 3177 3179 3205* 3206* 3210* 3211 3215*
3229* 3230 3240 3245 3249 3251 3269 3278* 3283 3286* 3287* 3295* 3296*
3297* 3298 3304* 3305 3316 3322* 3323* 3402* 3417* 3418 3510* 3511 3524
3532 3535 3541* 3579* 3582 3627* 3628 3629* 3630 3634* 3636* 3730* 3734*
3744* 3750* 3758 3764* 3790 3793 3832* 3833* 3835* 3836* 3837* 3838 3885
3886* 3887* 3888 3890* 4010* 4011 4032 4055 4078 4087 4089* 4091* 4110
4231* 4233 4293 4311 4316* 4318* 4321 4323 4324 4346 4360* 4392* 4394
4395 4397* 4402 4406* 4602 4638* 4639* 4640 4646 4652 4658 4669* 4691
4694* 4729* 4732 4879* 4886 4945 4948* 4949 4952* 4953* 4954* 4955* 4956
4959* 4960 4963* 4964* 4965 4967* 5305 5309* 5313* 5315* 5317* 5320* 5323
5326* 5347 5357* 5359 5361 5371* 5372 5380* 5404 5417* 5418 5422 5446*
5539* 5543 5544* 5556* 5558* 5560* 5583* 5589* 5595* 5673 5676* 5679* 5689*
5695* 5699 5701* 5733 5738* 5743* 5764* 5767 5771* 5790* 5781 5785* 5810
5833* 5844 5851* 5855* 5860* 5861* 5864* 5866* 5870* 5872 5876* 5999 6000*
6004* 6005 6009* 6125* 6127* 6129* 6130 6150 6151* 6152* 6157 6160* 6279
6302*

R2 =%000002
641# 1644* 1653 1660 1663* 1672* 1677* 1678 1683 1689* 1690* 1691* 1692*
1693* 1695 1698 1701* 1704* 1706 1754* 1785* 1792 1807 1811* 1821 1854*
1855* 1878* 1908* 1913 1925 1926 1929 1947 1948 1962 1963 1965 1990*
1991* 2014 2015* 2046* 2050 2063 2064 2066 2080 2081 2140* 2143* 2214*
2215* 2263* 2267 2275 2285* 2377* 2381 2384 2395* 2443* 2446* 2449* 2453*
2458* 2640* 2643 2829* 2833 2835 2837 2846 2847 2924* 2925* 2926 2993*

```
                    2994*  2995*  2996*  2997   2998   3078*  3079*  3080*  3081*  3082   3094   3305*  3306*
                    3307*  3377*  3378*  3379   3382   3384*  3444*  3456*  3485*  3486*  3499*  3578   3582*
                    3583*  3584*  3585   3588*  3788*  3790   4041*  4085   4166*  4167   4168   4203   4207*
                    4209*  4236*  4237*  4238*  4288*  4347   4368*  4370*  4371*  4374*  4375*  4383*  4384
                    4386*  4391*  4393*  4398*  4403   4405*  4423   4437   4474   4476   4687   4690*  4693*
                    4709*  4710*  4713*  4717*  4720*  4732*  4733*  4734   4741   4758*  4761   4765   4769
                    4774   4778   4783   4786*  4788   4880*  4890*  4946   4949*  4950*  4951   4956*  4957*
                    4958   4960*  4961*  4962   4966*  5306   5310*  5314*  5316*  5318*  5324   5325*  5348
                    5353*  5356*  5375   5379*  5405   5416*  5420*  5423   5430*  5431*  5432   5437*  5445*
                    5545*  5546   5547*  5562*  5563*  5580*  5583   5584*  5590*  5591*  5596*  5597*  5674
                    5680*  5683*  5687   5700*  5734   5745*  5746   5748   5756*  5761   5784*  5811   5832*
                    5845   5853*  5856*  5862*  5865*  5867*  5868*  5869   5875*  6280   6301*
R3      =%000003     642*  1650*  1660*  1669*  1675   1733*  1755*  1771*  1783*  1798   1809   1819   1823*
                    1863*  1898*  1992*  2016*  2038*  2100   2207*  2344*  2427*  2428*  2430*  2433   2452*
                    2466   2504   2641*  2700*  2837*  2838*  2839   2841   3012*  3014   3018*  3019*  3025
                    3046*  3051   3082*  3083   3385*  3387*  3388*  3389*  3392   3556*  3559   3751*  3752*
                    3754   3756   3757*  3758*  3760   3762   3763*  3764   3784*  3785*  3787*  3788   3793*
                    3803   4030*  4031*  4032   4163*  4171   4173   4174*  4177*  4200   4227*  4229   4230
                    4231   4241   4247*  4252*  4259*  4305*  4348   4369*  4373*  4404*  4419   4425*  4426*
                    4428   4430*  4441*  4444   4455*  4488*  4489   4491*  4688   4702*  4703*  4704   4719*
                    4741*  4744*  4847*  4854   5262   5263*  5264   5267*  5268   5272   5274   5276*  5278*
                    5406   5414*  5415*  5429*  5432*  5441*  5442*  5444*  5542*  5555*  5562   5588*  5593*
                    5599*  5600   5735   5760*  5761*  5763   5783*  5812   5831*  5846   5852*  5857*  5874*
                    5925   5934*  5940*  5941*  5944*  5949*  5950*  5951   5960*  6281   6300*
R4      =%000004     643*  1907*  1908   2045*  2046   2208*  2345*  2490*  2653*  2654*  2655*  2656*  2657*
                    2658   2660*  2662   2663   2697   2997*  3028*  3031   3032   3053*  3055   3056   3390*
                    3391*  3394   3398   3466   3546   3552   3568   3626*  3851*  3866*  3863   3919   3941*
                    3942*  3943*  3945*  3957*  4044*  4046*  4047   4075*  4076   4080*  4081   4082   4162*
                    4180*  4196*  4229*  4245   4247   4248*  4251*  4308   4420   4428*  4432*  4440*  4443
                    4454*  4489*  4492   4494   4640*  4706   4740*  4743*  4748   4753   4855*  4897*  4898*
                    4912   4915   4918   4921   4924   5541*  5548*  5586*  5589   5595   5597   5813   5830*
                    5847   5848*  5850   5851   5859   5861   5871*  5872*  5873*  5926   5928*  5929*  5930*
                    5931   5932*  5946   5948*  5956*  5959*  6282   6299*
R5      =%000005     644*  1912*  1923*  1945*  2049*  2061*  2078*  2141*  2142*  2262*  2269   2271*  2376*
                    2392*  2533*  2534*  2535*  2536   2670*  2998*  3000   3012   3018   3020   3021   3041
                    3042   3043*  3343*  3344   3346   3355*  3356   3371   3379   3381   3392*  3394   3397
                    3633*  3841*  3848*  3856*  3859*  3861*  3862*  3863*  3920   3921*  3924   3941   3946*
                    3950*  3951   3956*  4028*  4029*  4030   4034   4039   4042   4044   4045*  4046   4049*
                    4076*  4077*  4078   4162   4163   4173*  4178*  4180   4183*  4197*  4213*  4214*  4230*
                    4253   4277   4282   4291   4292   4304   4421   4431*  4433*  4453*  4696   4854*  4855
                    5407   5409*  5411*  5418*  5422*  5437   5443*  5540*  5546*  5551*  5552   5587*  5591
                    5598   5814   5829*  5927   5933*  5935*  5937*  5938*  5939*  5940   5958*  6283   6298*
R6      =%000006     645*   647   1561*  1562*  1563   1632
R7      =%000007     646*   648
SAVKEY  002236      1365*  2940*  2953   3386*  3514*  3515*  3542*  3554
SAVREG= 104413      4161   4226   5538   5579   6123   6265*
SCP    = 020000      733*  3125   3334
SECMAP  002224      1352*  2607*  3714
SEXIT   021350      4837   4861   4900*
SFTERR  013356      3430   3435*
SIN    = 001000      738*  3272   3407   3935   3987
SINCN   002322      1397*  4010*  4011   4788
SINCNT  015732      3290   3413   4010*
SKCMP   012424      3222*  3336
SKE    = 010000      745*  3476   3522
SKECN   002302      1391*  3485*  4778
```

```
SNOTYP  026374          4059    6089    6148#
SP     =.000006          647#   1565*   1577*   1585*   1589    1683*   1730    1792*   1798*   1806    1818    1964*   1967*
                        1968*   1969    1971    1973*   1974    2104*   2105*   2119    2120*   2433*   2434*   2435*   2437
                        2441    2598*   2599*   2601*   2603    2605    2615*   2616*   2618*   2620    2622*   2623    2691*
                        2714*   2726*   2727*   2802*   2803*   2809*   2810*   2900*   2901*   2946*   2947*   3111    3118*
                        3226*   3229    3330*   3338*   3343    3535*   3578*   3588    3609*   3610*   3652*   3653*   3655*
                        3657    3659    3682*   3683*   3685*   3687    3689    3691*   3692    3697*   3698*   3700*   3702
                        3704    3711*   3712*   3714*   3716    3718    3753*   3754*   3756*   3757    3759*   3760*   3762*
                        3763    3772*   3773*   3775*   3777    3779    3800*   3801*   3803*   3805    3807*   3808    3813*
                        3814*   3816*   3818    3820    3885*   3890    3893*   3919*   3920*   3956    3957    4013*   4055*
                        4066*   4067*   4069*   4071    4073    4109*   4110*   4112*   4113*   4114*   4115*   4116    4144
                        4145*   4203*   4204*   4205    4233*   4236    4311*   4313*   4314    4345*   4346*   4347*   4348*
                        4349*   4350*   4351    4353*   4356    4357*   4359    4367*   4371    4372*   4379    4399*   4400
                        4401    4404    4405    4406    4407    4419*   4420*   4421*   4423*   4425    4437*   4439*   4440
                        4445*   4453    4454    4455    4503*   4504*   4505    4543*   4573*   4575    4587*   4588*   4589
                        4601*   4602*   4633*   4634*   4646*   4652*   4658*   4669    4670    4686*   4687*   4688*   4719
                        4720    4721    4734*   4748*   4749*   4753*   4754*   4761*   4765*   4769*   4770*   4774*   4778*
                        4783*   4788*   4812*   4813*   4882*   4885*   4886*   4900*   4901*   4944*   4945*   4946*   4966
                        4967    4968    4977*   4979*   4980*   4981*   4982*   4998*   5000    5013*   5014*   5016    5024*
                        5025*   5039*   5040*   5042    5048*   5049*   5055*   5067    5071*   5099*   5100*   5101*   5102*
                        5103*   5104*   5106*   5121*   5167*   5168*   5169    5176*   5179*   5180*   5184*   5185*   5189
                        5192*   5196    5198    5200    5201*   5208    5210    5212*   5213    5215*   5216*   5217*   5218*
                        5219*   5234*   5235*   5238*   5239*   5240    5244*   5245*   5246    5249    5251    5253*   5262*
                        5267    5278    5279*   5280*   5281*   5302*   5303*   5304*   5305*   5306*   5308    5311*   5319*
                        5320    5322    5323*   5325    5326    5327    5344*   5345*   5346*   5347*   5348*   5350    5352*
                        5363    5365*   5366*   5367*   5368*   5369*   5372*   5377*   5378*   5379    5380    5381    5384
                        5403*   5404*   5405*   5406*   5407*   5408*   5409    5412*   5425    5427*   5429    5439    5441
                        5443    5444    5445    5446    5447    5449*   5450*   5478*   5479    5480*   5482    5483    5484*
                        5486    5488    5490    5496    5498*   5500*   5508*   5512    5516    5517    5521    5539    5552*
                        5580    5582*   5632*   5634    5637*   5638    5654    5655*   5672*   5673*   5674*   5675*   5676
                        5678*   5680    5682*   5684*   5690*   5692    5697    5698*   5699*   5700    5701    5702    5723*
                        5726*   5727    5728*   5731*   5732*   5733*   5734*   5735*   5736*   5737*   5738    5739    5741*
                        5745    5751*   5754*   5765*   5769*   5772    5773    5776*   5777*   5778    5781*   5782*   5783
                        5784    5785    5786    5787*   5789    5809*   5810*   5811*   5812*   5813*   5814*   5815*   5816*
                        5817*   5818*   5825*   5826*   5827*   5828*   5829    5830    5831    5832    5833    5834    5843*
                        5844*   5845*   5846*   5847*   5848    5873    5874    5875    5876    5877    5878*   5917*   5918
                        5919    5920*   5925*   5926*   5927*   5933    5958    5959    5960    5961*   5962*   5999*   6009
                        6011    6013*   6014*   6015    6017*   6025    6042*   6046    6047*   6058*   6068*   6072*   6107*
                        6113    6131*   6150*   6157*   6160    6173*   6176*   6178*   6183    6204*   6207    6209    6210
                        6221    6222    6226*   6236*   6237    6278*   6279*   6280*   6281*   6282*   6283*   6284*   6285
                        6293*   6297    6298    6299    6300    6301    6302    6303    6308*

SRDRV   001750          1226#   3004*   3086*   3269*   3283*   3316*   4851*   5000*   6000    6051*   6151
SRNO    001756          1248#   1730*   6157
SR0    = 177572          791#   2166*   2291*   2307*   2400*   4365    4380*   4387*
SR1    = 177574          792#
SR2    = 177576          793#
SR3    = 172516          794#   4377*
STACK  = 001100          622#   1565    2714    3893    5121
START   003470           759     763    1558#   2737
STATST  020500          3543    4686#
STATUS  021066          2863    2881    3033    3059    4812#
STKLMT= 177774           633#
ST2     004146          1623    1634    1638    1641#
ST3     005064          1787    1825#
ST4     004514          1753#
SUPRS   024442          4450    4751    4756    5637#
```

```
SUPRSL  024426          5632#  6133
SUP1    024454          5635   5640#
SUP2    024516          5633*  5639*  5649   5651*  5653#
SR      001140           868#  1563   1579*  1591   1587*  1599   1724   1786   2635   2683   2705   3615   3620
                        4417   4875   4910   5019   5045   5065   5163   5200*  5991   5996   6026   6031   6035
                        6039   6093   6120   6148   6199   6215   6284   6297#
SREG    000176           757#  1587   1599   5163   5176
SW      = 000001         686#
SW00    = 000001         676#   686
SW01    = 000002         675#   685
SW02    = 000004         674#   684
SW03    = 000010         673#   683
SW04    = 000020         672#   682
SW05    = 000040         671#   681
SW06    = 000100         670#   680    1786
SW07    = 000200         669#   679
SW08    = 000400         668#   678
SW09    = 001000         667#   677    6038
SW1     = 000002         685#  1724    6148
SW10    = 002000         666#  5991
SW11    = 004000         665#  6093
SW12    = 010000         664#  6031
SW13    = 020000         663#  4417   4875   4910   5019   5045   5065   6026
SW14    = 040000         662#
SW15    = 100000         661#
SW2     = 000004         684#  5996
SW3     = 000010         683#  2705   6120
SW4     = 000020         682#  2635
SW5     = 000040         681#  3620
SW6     = 000100         680#
SW7     = 000200         679#
SW8     = 000400         678#  3615
SW9     = 001000         677#
TBITVE= 000014           719#
TIMER   002172          1332#  3926*  3929*  3978*  3981*  5036*  5043*
TIMOUT  022004          5007*  5017*  5028#
TIMTYP  026272          4726   6087   6120#
TKVEC   = 000060         726#
TPVEC   = 000064         727#
TRAPVE= 000034           725#  1569*  1570*  4353   4354*  4359*
TRTVEC= 000014           720#
TST1    005156          1840   1851#
TST2    005270          1876   1887#
TST3    005534          1952   1987#
TST4    005650          2013   2026#
TST5    006220          2085   2160#
TST6    006602          2163   2301#
TST7    007136          2305   2420#
TYPDBO  017466          2282   2390   4417#
TYPOS   = 104404        3611   4762   4766   4771   4775   4779   4784   4789   6158   6256#
TYPE    = 104400        1595   1629   1636   1673   1681   1716   1727   1735   1788   1794   1801   1813   2687
                        2696   2732   3589   3605   3617   3618   4054   4605   4606   4607   4615   4622   4629
                        4642   4648   4654   4664   4665   4666   4668   4727   4731   4739   4747   4752   4760
                        4764   4768   4773   4777   4782   4787   4793   4877   4881   4894   4914   4917   4920
                        4923   4926   5021   5022   5023   5047   5051   5068   5119   5144   5175   5178   5191
                        5202   5221   5270   5273   5277   5386   5388   5448   5491   5652   5952   5993   6029
```

```
                        6067    6082    6092    6096    6100    6102    6109    6114    6122    6136    6155    6156    6169
                        6175    6180    6252#   6306
TYPFN   021366          3459    3503    4641    4856    4910#
TYPMSG= 104420          2281    2389    3309    3445    3457    3458    3488    3500    3502    3520    3534    4853    6270#
TYPOC = 104401          1793    1799    4635    4647    4653    4659    4883    5026    5050    5056    5177    6073    6138
                        6174    6179    6253#
TYPON = 104403          6255#
TYPOS = 104402          1684    2692    3536    4056    4735    4887    6254#
TY.MSG  022152          5065#   6270
WATIME  022260          1551    5099#   5120
WCE   = 000001           747#   2517    3435
WCECN   002332          1402#   3444#   4765
WCFLG   002156          1313#   2764    2767    2787#   3559#   3570#
WCMAP   002232          1355#   3808#   3816
WPS   = 000040           741#   4128
WRDSK   010012          2632#
WRFNC   011604          3015    3041#
XFR     007256          2450    2458#
XFR16K  007246          2425    2452#
$AUTOB  001134           865#   1603#   5171    5292
$BDADR  001122           860#
$BDDAT  001126           862#
$BELL   001206           890#   F993
$CHARC  024106          5493#   5503#   5510    5519#   5524#
$CKSUM  022412          5163#   6260
$CMTAG  001100           848#   1560    1561    1569
$CM1  = 000011           880#    881#    882#    883#    884#    885#    886#    887#    888#    889#
$CM2  = 000022           880#    881#    882#    883#    884#    885#    886#    887#    888#    889#
$CM3  = 000011           878#    880
$CNTLG  023137          5174    5287#
$CNTLU  023132          5191    5286#
$CORE   017366          4363    4390#
$CRLF   001213           892#   1735    2696    3617    3618    4668    4731    4793    4881    5021    5202    5286    539_
                        5492    5527    6023    6067    6092    6096    6102    6114    6155
$CROUT  017416          4390    43F7#
$DBLK   023662          5414    5418    5456#
$DB2D   024232          4750    4735    5579#   6132
$DB2O   024112          4446    5538#
$DECVL  024412          5581    5627#
$DIV    024640          2436    2602    2619    3656    3686    3701    3715    3776    3804    3817    4070    5723#
$DOAGN  022406          5139    5144    5150#
$DTBL   023652          5417    5452#
$ENDAD  022376           777    5146#
$ENDCT  022362          5141#
$EOP    022332          3613    5122    5132#
$EOPCT  022354          5138#   5142
$ERFLG  001103           851#   5988#   6190    6213    6219#   6228
$ERMAX  001115           857#   6224#   6228
$ERROR  025634          1608    5988#   5989
$ERRPC  001116           858#   6011#   6012#   6014    6072    6173    6591    6594    6596    6599    6602
$ERRTB  001216           909#   6079
$ERRTY  026104          6028    6067#
$ERTTL  001112           855#   5994#
$ESCAP  001204           889#   6223#
$FILLC  001156           876#   5496    5527
$FILLS  001155           875#   5527
```

```
$GDADR  001120           859#
$GDDAT  001124           861#
$GET42  022366          5143#
$GTSWR  022462          5175#  6259
$HD   = 000000           596
$HIOCT  023266          5324*  5329#
$ICNT   001104           852#
$ILLUP  027206          6276   6292   6311#
$INTAG  001135           866#  5203   5292
$ITEMB  001114           856#  6025*  6040   6070   6084
$KTNEX  017360          4364   4389#
$KTOUT  017350          4381   4386#
$KT11   017216          1756   2421   4362#  4366*  4389*
$LF     001214           893#  5277   5286   5390   5527
$LPADR  001106           853#  1573*  6217*  6221*  6226   6228
$LPERR  001110           854#  1574*  1857*  1890*  1989*  2029*  2175*  2231*  2312*  2356*  2460*  2500*  2639*
                        6058   6217   6222*  6228
$LSTAD  017462          1774   4402*  4409#
$LSTBK  017464          1758   2161   2303   2424   4403*  4410#
$MAIL = ****** L        1591   1599   5480   6221
$MNEM   023155          5178   5290#
$MSWR   023144          5175   5288#
$MUL  = 000003          3756#  3758   3762#  3764
$MULT   024526          3755   3761   5671#
$NULL   001154           874#  5498   5527
$NWTST= 000001          1842#  1844   1880#  1882   1976#  1978   2018#  2020   2148#  2150   2292#  2294   2401#
                        2403
$OCNT   025630          5924*  5953*  5966#
$OCTVL  024214          5540   5565#
$OMODE  025632          5919*  5923*  5928   5931*  5942*  5968#
$OVER   026742          6200   6218   6225#
$PASS   001100           849#  3609   5135*  5136*  5153
$POWER  027214          6307   6314#
$PWRAD  027202          6309#
$PWRDN  027042          1571   6276#  6304
$PWRMG  027176          6307#
$PWRUP  027114          6286   6292#
$QUES   001212           891#  5221   5270   5286   5388   5390   5527
$RAND   025224          3647   3679   3694   3769   3810   4186   4262   5842#
$ROCHR  022674          5234#  6261
$RODEC  023270          5344#  6264
$ROLIN  023014          5262#  6262
$ROOCT  023166          5302#  6263
$ROSZ = 000010          5255#
$REGAD  001160           878#
$REG0   001162           880#  1962*  2273*  2383*  2776*  2961*  2969*  2978*  3064*  3159*  3172*  3179*  3245*
                        3251*  3346*  3371*  3381*  3397*  4291*  4443*  4445   4476*  4494*  4516*  4932*  6591
                        6594   6596   6599   6602
$REG1   001164           881#  1915*  1931*  2052*  2068*  2275*  2384*  2777*  3382*  3398*  4284*  4287*  4444*
                        4477*  4497*  4517*  4933*  6591   6596   6599   6602
$REG10  001202           888#
$REG2   001166           882#  1963*  2521*  4292*  4934*  6591   6596   6599
$REG3   001170           883#  1974*  4293*  4931*  6591   6596   6599
$REG4   001172           884#  4849*  4850*  4851   6596   6599
$REG5   001174           885#  6596
$REG6   001176           886#  4947   6596
```

```
$REG7   001200          887#
$RESRE  025166         5824#  6266
$RTNAD  022410         5152#
$R2A  = ****** U       6267
$SAVRE  025130         5808#  6265
$SAVR6  027212         6285*  6293   6294*  6295*  6313#
$SCOPE  026612         1567   6197#
$SETUP= 000115         1556#  1566   1567   1569   1571   1573   1595   1596   5134   5158   5292   6198
$SIZE   017132         1753   4345#
$SIZEX  017422         4388   4398#
$STUP = 177777         1556#
$SVLAD  026714         6208   6214   6220#
$SVPC = 000220          775#   780
$SWR  = 143000          585#   596    601    602    603    604    605    606    889    890   1573   1574   1852
                       1888   1988   2027   2161   2302   2421   5129   5135   5145   5151   5153   6191   6192
                       6193   6194   6199   6211   6213   6214   6215   6220   6222   6225   6228   6310
$SWRMK= 000000         6194
$TKB    001146          871#  5156   5167   5184   5238   5244
$TKS    001144          870#  5156   5165   5181   5205*  5236   5242
$TN   = 000010          585#   596   1842   1852#  1880   1888#  1976   1988#  2018   2027#  2148   2161#  2292
                       2302#  2401   2421#
$TNPWR  024342         5586   5587   5607#
$TPB    001152          873#  5516*  5527
$TPFLG  001157          877#  5474   5527
$TPS    001150          872#  5514   5527
$TRAP   026756         1569   6236#
$TRP  = 000021         6244#  6253#  6254#  6255#  6256#  6257#  6258   6259#  6260   6261#  6262#  6263#  6264#
                       6265#  6266#  6267#  6268#  6269#  6270#  6271#
$TRPAD  027000         6241   6251#
$TSTNM  001102          850#  5134*  5990   6190   6220*  6225   6228
$TTYIN  023122         5263   5264   5281   5285#
$TYPBN= ****** U       6257
$TYPDS  023446         5402#  6236
$TYPE   023672         5474#  6244   6252
$TYPEC  024042         5207   5495   5502   5509   5514#  5515
$TYPEX  024110         5520   5522   5525#
$TYPOC  025432         5922#  6253
$TYPON  025446         5921   5924#  6255
$TYPOS  025406         5917#  6254
$XTSTR  026624         6202#
$$GET4= 000000         5145#
$DFILL  025631         5918#  5922*  5932   5967#
$40CAT= ****** U       6199
.     = 032110          751#   755#   761#   775    776#   778#   780#   847#   894   1239#  1248#  1275#  1289#
                       1299#  1305#  1312#  1375#  1386#  1391#  1397#  1402#  1408#  1415#  1419#  1422#  1427#
                       1564   1573   1574   1607#  1791#  1804#  1816#  2111   2735#  2808   2905   4618#  4625#
                       4632#  4899   5015   5041   5054#  5153   5156   5285#  5286   5292   5390   5456#  5527
                       5565#  5627#  6228   6288   6312
```

F14

CROSS REFERENCE TABLE -- MACRO NAMES

```
COMENT  1842#  1844   1880#  1882   1976#  1978   2018#  2020   2148#  2150   2292#  2294   2401#  2403
COMMEN     1#   729#
ENDCOM     1#   729#
ERROR    623#  1866   1868   1870   1873   1902   1905   1910   1917   1933   2002   2005   2007   2011   2041
        2044   2048   2054   2070   2193   2202   2210   2217   2223   2252   2276   2330   2340   2347   2351
        2373   2385   2483   2487   2493   2497   2513   2522   2679   2778   2962   2970   2979   2989   3006
        3065   3074   3088   3122   3129   3143   3160   3173   3180   3200   3223   3233   3246   3252   3270
        3284   3317   3332   3347   3361   3372   3383   3399   3411   3462   3508   3933   3939   3985   3991
        4121   4126   4132   4138   4295   4858   486C
ESCAPE     1#   729#
GETPRI     1#   729#  4353
GETSWR     1#   585#   729#  1596#
MULT       1#   729#  3753   3759
NEWTST     1#   729#  1842   1880   1976   2018   2148   2292   2401
POP        1#   729#  5325   5379   5443   5700   5829   6297   6298
PUSH       1#   729#  5304   5346   5402   5671   5809   6278   6284
REPORT     1#   729#
SCOPE    624#  1851   1887   1987   2026   2160   2301   2420   5133
SDIVDE   617#  2597   2614   3651   3681   3696   3709   3771   3798   3812   4065
SETPRI     1#   729#
SETTRA  6244#  6253   6254   6255   6256   6258   6260   6261   6262   6263   6264   6265   6266   6267   6268
        6269   6270
SETUP      1#   729#  1558
SKIP       1#   729#
SLASH      1#   729#
SPACE    729#
STARS      1#   729#   773    843    894   1842   1850   1880   1886   1976   1986   2018   2025   2148   2159
        2292   2300   2401   2419   4331   5126   5155   5158   5226   5255   5294   5332   5392   5459   5529
        5568   5659   5793   5894   6187   6230   6274   6290
SWRSU      1#   585#   729#  1575#
TRMTRP  6244#
TYPBIN     1#   729#
TYPDEC     1#   729#
TYPNAM     1#   585#   729#  1591
TYPNUM     1#   729#
TYPOCS     1#   729#
TYPOCT     1#   729#  5176
TYPTXT     1#   729#  1788   1794   1800   1812   2687   2732   3605   4615   4621   4628   4642   4648   4654
        5051   6168
$$CMRE   841#   880    881    882    883    884    885    886    887    888
$$CMTM   841#
$$ESCA     1#   729#
$$NEXT     1#   729#  1842   1880   1976   2018   2148   2292   2401
$$SET   6244#  6253   6254   6255   6256   6258   6260   6261   6262   6263   6264   6265   6266   6267   6268
        6269   6270
$$SKIP     1#   729#
.EQUAT     1#   585#   619
.HEADE     1#   585#   586
.KT11      1#   585#   783
.SETUP     1#   585#  1556
.SWRHI     1#   585#   597
.SWRLO   607#   608    609    610    611    612
.$ACT1     1#   585#   771
.$APTB     1#
.$APTH     1#
.$APTY     1#
```

```
.$ASTA      1#
.$CATC      1#    585#    749
.$CMTA      1#    585#    841
.$DB2D      1#    585#   5566
.$DB2O      1#    585#   5527
.$DIV       1#    585#
.$EOP       1#    585#   5124
.$ERRO      1#
.$ERRT      1#
.$MULT      1#    585#   5657
.$POWE      1#    585#   6272
.$RAND      1#    585#
.$RDDE      1#    585#   5330
.$RDOC      1#    585#   5292
.$READ      1#    585#   5153
.$R2AZ      1#
.$SAVE      1#    585#   5791
.$SB2D      1#
.$SB2O      1#
.$SCOP      1#    585#   6185
.$SIZE      1#    585#   4329
.$SUPR      1#
.$TRAP      1#    585#   6228
.$TYPB      1#
.$TYPD      1#    585#   5390
.$TYPE      1#    585#   5457
.$TYPO      1#    585#   5892
.$40CA      1#
.1170       1#
```

	5015	5018	5020	5041	5044	5046	5066	5090	5105	5164	5170	5190	5197	5204	5241
	5247	5269	5275	5355	5364	5424	5481	5489	5497	5511	5518	5603	5644	5650	5691
	5766	5858	5945	6027	6071	6111	6200	6296							
BPL	1757	1840	2422	2741	2754	2765	2927	2960	2968	2977	3063	3190	3666	3925	4040
	4363	4464	4609	4700	5166	5182	5237	5243	5410	5440	5475	5515	5677	5681	5740
	5766	5943	6036												
BR	1584	1602	1605	1623	1634	1638	1702	1769	1772	1780	1789	1795	1802	1814	1879
	1954	2017	2087	2132	2447	2450	2540	2637	2688	2733	2745	2789	2831	2850	2933
	2966	2972	2980	2990	3007	3066	3075	3089	3100	3260	3292	3324	3564	3606	3646
	3674	3731	3745	3856	3957	4037	4090	4140	4175	4190	4210	4249	4266	4285	4326
	4378	4388	4576	4579	4584	4616	4619	4623	4626	4630	4636	4643	4649	4655	4667
	4711	4861	5052	5122	5193	5220	5222	5248	5271	5321	5374	5389	5421	5438	5477
	5494	5504	5513	5520	5564	5594	5635	5646	5753	5755	5758	5790	5921	5936	5957
	6059	6074	6126	6137	6170	6202	6208	6211	6218	6288	6312				
BVS	5370	5373													
CLC	1688	2429	2612	2646	3151	3786	3827	3834	4111	4487	4490	4978	5757		
CLR	1562	1642	1643	1644	1645	1646	1647	1648	1671	1672	1931	2068	2104	2108	2236
	2291	2361	2400	2432	2434	2443	2555	2562	2599	2616	2787	2801	2802	2900	2946
	3215	3216	3519	3540	3570	3627	3635	3645	3653	3663	3683	3698	3712	3730	3733
	3773	3801	3814	3907	3921	3922	3926	3978	4067	4174	4248	4284	4307	4375	4387
	4391	4430	4586	4690	4740	4812	4828	4864	4885	4897	5007	5036	5099	5106	5134
	5179	5180	5309	5310	5352	5353	5413	5416	5545	5588	5633	5675	5685	5736	5770
	5853	5934	6013	6069	6223	6294									
CLRB	2634	3105	3205	3206	3278	3286	3287	3296	3297	3304	3322	3323	3328	3355	3541
	3579	3946	4049	5276	5385	5442	5493	5519	5604	5776	6219				
CMP	1563	1581	1599	1666	1759	1807	1809	1819	1821	1825	1913	1926	1948	1965	2050
	2064	2081	2119	2161	2184	2237	2267	2287	2303	2319	2362	2381	2397	2424	2556
	2563	2658	2663	2742	2757	2774	2841	2847	2854	2879	2930	3140	3379	3466	3546
	3552	3568	3585	3630	3643	3660	3689	3705	3719	3723	3742	3790	3845	3849	3868
	3888	3908	3951	4032	4034	4042	4047	4078	4082	4171	4205	4245	4277	4314	4321
	4379	4384	4395	4474	4492	4505	4661	4706	4912	4915	4918	4921	4924	5163	5169
	5189	5196	5208	5210	5240	5246	5249	5251	5264	5434	5598	5692	5746	5789	6134
	6181	6209													
CMPB	1619	1624	3014	3211	3298	3394	3418	3511	4011	5171	5203	5268	5274	5354	5359
	5361	5486	5488	5496	5517	5521	5643	6005	6015	6040	6084				
COM	4169	4243	4325												
DEC	1733	1839	1967	2700	3602	3852	3691	3707	3721	4061	4191	4267	4694	4744	4791
	5137	5548	5601	5647	5682	5690	5765	5777	6075						
DECB	5500	5503	5741	5942	5953										
EMT	623														
HALT	755	2736	3622	5476	6037	6287	6311								
INC	1593	1661	1664	1732	1766	1923	1945	1968	2061	2078	2110	2113	2143	2183	2271
	2286	2318	2392	2396	2439	2622	3307	3444	3456	3485	3486	3499	3584	3610	3636
	3807	3825	3874	3929	3950	3981	4029	4041	4045	4183	4197	4209	4257	4259	4275
	4288	4302	4305	4318	4380	4433	4823	4838	4840	4890	4898	5014	5017	5040	5043
	5081	5085	5089	5092	5135	5218	5420	5551	5593	5678	5754	5857	5948	5956	5994
	6295														
INCB	762	1552	2807	3117	3210	3295	3417	3510	4010	5523	5988	6004	6220		
IOT	624														
JMP	759	763	1830	1841	2163	2305	2545	2737	2796	2863	2876	2881	2888	2907	3033
	3038	3059	3112	3431	3470	3471	3587	3613	3871	3894	4016	4092	4095	4148	4895
	4904	5151	6007												
JSR	1551	1753	1864	1867	1869	1872	1896	1900	1904	1909	1916	1932	2000	2004	2006
	2010	2034	2040	2043	2047	2053	2069	2116	2192	2197	2209	2216	2222	2251	2282
	2329	2338	2346	2350	2372	2390	2436	2482	2486	2491	2496	2512	2520	2592	2602
	2619	2670	2678	2717	2744	2897	2898	2942	2944	2988	3005	3023	3035	3036	3043

```
        3049  3073  3087  3108  3109  3121  3127  3142  3198  3208  3214  3222  3227  3232  3259
        3268  3282  3288  3290  3315  3320  3331  3336  3341  3403  3409  3412  3413  3424  3459
        3460  3461  3468  3503  3506  3507  3516  3521  3543  3548  3612  3619  3624  3639  3647
        3656  3679  3686  3694  3701  3715  3755  3761  3769  3776  3804  3810  3817  3883  3891
        3921  3937  3954  3969  3973  3983  3989  3995  4059  4063  4070  4119  4124  4130  4136
        4186  4234  4262  4294  4424  4446  4450  4465  4507  4529  4540  4574  4641  4726  4750
        4751  4755  4756  4856  4857  4865  4893  4943  4990  4999  5120  5146  5207  5495  5502
        5509  6028  6033  6087  6089  6095  6132  6133
MOV     1561  1565  1567  1568  1569  1570  1571  1572  1573  1574  1577  1578  1579  1580  1585
        1587  1588  1589  1608  1609  1610  1611  1649  1650  1653  1657  1660  1669  1670  1678
        1683  1695  1706  1710  1730  1754  1755  1758  1763  1771  1774  1783  1785  1792  1798
        1806  1811  1818  1823  1827  1835  1836  1837  1853  1854  1857  1862  1863  1889  1890
        1898  1907  1908  1912  1915  1962  1963  1964  1974  1988  1989  1990  1992  1998  2014
        2016  2027  2029  2038  2045  2046  2049  2052  2098  2099  2100  2102  2103  2105  2109
        2131  2140  2141  2142  2164  2165  2166  2175  2179  2181  2182  2186  2187  2188  2189
        2207  2208  2214  2231  2235  2244  2245  2246  2247  2262  2263  2265  2273  2275  2306
        2307  2312  2315  2316  2317  2322  2323  2324  2325  2344  2345  2356  2360  2365  2366
        2367  2369  2376  2377  2379  2383  2384  2427  2433  2435  2452  2453  2460  2464  2466
        2477  2478  2490  2500  2503  2504  2506  2508  2521  2533  2554  2566  2567  2568  2569
        2570  2572  2573  2574  2575  2576  2577  2578  2579  2580  2581  2582  2583  2584  2587
        2598  2601  2605  2606  2607  2608  2610  2615  2618  2623  2639  2640  2641  2642  2645
        2651  2653  2660  2662  2665  2668  2669  2674  2675  2691  2714  2726  2727  2739  2748
        2767  2770  2776  2777  2790  2803  2809  2810  2814  2819  2829  2837  2865  2872  2899
        2901  2906  2917  2924  2940  2945  2947  2952  2953  2957  2961  2969  2978  2993  2997
        2998  3000  3004  3012  3018  3020  3021  3025  3041  3042  3047  3048  3051  3060  3064
        3078  3082  3083  3086  3103  3104  3138  3149  3156  3159  3172  3179  3226  3229  3245
        3251  3264  3269  3283  3305  3316  3330  3338  3343  3346  3371  3377  3381  3382  3385
        3386  3390  3392  3397  3398  3524  3559  3578  3582  3588  3604  3609  3625  3626  3628
        3632  3633  3634  3652  3655  3659  3682  3685  3692  3697  3700  3704  3711  3714  3718
        3725  3734  3744  3748  3751  3753  3754  3756  3757  3759  3760  3762  3763  3764  3772
        3775  3779  3784  3788  3793  3800  3803  3808  3813  3816  3820  3832  3841  3848  3856
        3859  3861  3863  3885  3886  3890  3893  3906  3919  3920  3956  3957  3968  3972  3974
        4028  4044  4055  4066  4069  4073  4075  4076  4109  4110  4116  4144  4162  4163  4166
        4167  4168  4173  4178  4180  4188  4189  4193  4196  4203  4227  4229  4230  4231  4233
        4236  4240  4241  4242  4247  4252  4253  4264  4265  4269  4287  4291  4292  4293  4311
        4323  4324  4345  4346  4347  4348  4349  4350  4351  4353  4354  4356  4357  4359  4360
        4364  4368  4369  4370  4371  4374  4376  4377  4381  4386  4390  4399  4400  4401  4402
        4403  4404  4405  4406  4407  4419  4420  4421  4423  4425  4428  4431  4437  4443  4444
        4445  4453  4454  4455  4476  4477  4494  4497  4503  4516  4517  4572  4573  4578  4581
        4587  4591  4592  4601  4602  4604  4633  4638  4639  4640  4646  4652  4658  4669  4670
        4686  4687  4688  4696  4702  4719  4720  4721  4728  4729  4734  4741  4748  4753  4761
        4765  4769  4774  4778  4783  4813  4829  4847  4849  4851  4854  4855  4878  4879  4880
        4882  4900  4901  4931  4932  4933  4934  4944  4945  4946  4947  4948  4949  4951  4956
        4958  4960  4962  4965  4966  4967  4968  4998  5000  5008  5009  5013  5024  5035  5039
        5048  5055  5067  5084  5088  5091  5100  5101  5102  5103  5121  5140  5143  5176  5200
        5205  5234  5235  5262  5263  5278  5279  5280  5281  5302  5303  5304  5305  5306  5308
        5323  5324  5325  5326  5327  5344  5345  5346  5347  5348  5350  5351  5366  5378  5379
        5380  5381  5403  5404  5405  5406  5407  5408  5409  5414  5417  5437  5443  5444  5445
        5446  5447  5449  5450  5478  5479  5483  5498  5539  5540  5541  5542  5543  5544  5547
        5552  5580  5581  5582  5583  5584  5585  5586  5587  5632  5634  5637  5638  5639  5651
        5654  5655  5672  5673  5674  5676  5680  5684  5698  5699  5700  5701  5702  5723  5724
        5726  5727  5728  5732  5733  5734  5735  5737  5738  5739  5745  5752  5760  5763  5771
        5781  5782  5783  5784  5785  5786  5787  5809  5810  5811  5812  5813  5814  5815  5816
        5817  5818  5825  5826  5827  5828  5829  5830  5831  5832  5833  5834  5843  5844  5845
        5846  5847  5848  5850  5851  5852  5871  5872  5873  5874  5875  5876  5877  5917  5925
        5926  5927  5933  5940  5958  5959  5960  5961  5962  5990  5999  6000  6009  6011  6025
```

```
        6042  6047  6051  6058  6068  6072  6080  6098  6103  6107  6113  6124  6125  6127  6130
        6131  6150  6151  6157  6160  6173  6176  6177  6178  6183  6204  6205  6207  6210  6217
        6221  6222  6225  6226  6236  6237  6241  6276  6277  6278  6279  6280  6281  6282  6283
        6284  6285  6286  6292  6293  6297  6298  6299  6300  6301  6302  6303  6304  6305  6308
MOVB    1603  1622  1633  1637  1640  1659  1677  2643  3031  3032  3055  3056  3094  3098  3166
        3240  3356  3535  3664  3826  3941  4030  4046  4732  4788  4845  4696  5167  5184  5238
        5244  5267  5272  5311  5356  5357  5412  5415  5429  5432  5441  5480  5508  5516  5546
        5600  5645  5648  5918  5919  5922  5923  5924  5928  5931  5932  5951  6014  6224  6239
NEG     2101  2467  2505  2667  3824  3862  4697  4703  5377  5411  5679  5683  5694  5695  5742
        5743  5756  5775  5780  5929
NOP     3037  3110  5147  5148  5149  6610
RESET   5145
ROL     3152  3153  3154  3155  4432  4488  4979  4980  4981  4982  5314  5316  5318  5759  5764
        5855  5856  5935  5937  5938  5939  5941
ROR     1689  1690  1691  1692  2430  2613  2647  2648  2649  2650  3787  3828  3829  3830  3831
        3835  3836  3837  4112  4113  4114  4115  4491  5556  5557  5558  5559  5560  5561  5688
        5689
RTI     1586  2106  2728  2804  2811  2902  2948  3218  3337  3404  3415  3422  3425  3517  3528
        3573  4358  4814  4902  5027  5057  5072  5083  5087  5094  5206  5254  5282  5328  5382
        5451  5485  5729  5788  5819  5835  5963  6057  6227  6310
RTS     1975  2117  2121  2147  3106  3279  3301  3310  3313  3590  3876  3910  3958  3997  4014
        4146  4214  4328  4408  4457  4466  4478  4495  4508  4518  4530  4541  4544  4593  4671
        4722  4794  4927  4935  4969  4983  5001  5107  5118  5525  5554  5606  5656  5703  5879
        6115  6139  6161  6184  6242
SBC     5590  5696  5744
SOB     4373
SUB     1762  1768  1779  1781  2428  2444  2536  2611  2654  2697  3749  3752  3785  4397  4398
        4588  4704  5025  5049  5371  5418  5589  5591  6012  6017
SWAB    2657  2666  2994  3079  3387  3833  4963
TRAP    4355  5725  6244  6253  6254  6255  6256  6258  6260  6261  6262  6263  6264  6265  6266
        6267  6268  6269  6270
TST     1597  1617  1632  1675  1756  1777  1925  1929  1947  2063  2066  2080  2269  2421  2437
        2530  2603  2620  2715  2730  2740  2751  2756  2764  2815  2833  2846  2853  2868  2878
        2920  2929  2959  3111  3189  3657  3672  3676  3687  3702  3716  3777  3805  3818  3822
        4071  4081  4085  4194  4200  4270  4273  4280  4282  4289  4304  4308  4365  4382  4394
        4463  4514  4608  4663  4691  4699  5016  5042  5116  5198  5213  5322  5375  5384  5423
        5433  5482  5490  5512  5649  5697  5748  5767  5772  5773  5778  5946  6035  6046  6110
        6206  6238
TSTB    1626  1651  1721  1828  2558  2632  2753  2817  2821  2823  2835  2839  2861  2870  2874
        2904  2922  2926  2951  2967  2976  2986  3062  3071  3157  3177  3230  3249  3344  3369
        3532  3665  3924  4039  4361  4538  4610  4825  4831  5037  5165  5181  5236  5242  5425
        5439  5474  5514  5641  6213
.ASCII   891   892  1545  1546
.ASCIZ   890   893  1459  1460  1461  1462  1464  1466  1468  1469  1471  1473  1476  1477  1478
        1480  1483  1489  1492  1494  1495  1499  1501  1511  1514  1515  1516  1530  1537  1547
        1607  1791  1797  1804  1816  2690  2735  3608  4618  4625  4632  4645  4651  4657  4672
        4674  4676  4677  5054  5286  5287  5288  5290  6172  6314  6319  6322  6330  6334  6338
        6347  6353  6360  6368  6377  6385  6393  6401  6410  6412  6416  6422  6424  6428  6436
        6444  6450  6460  6470  6475  6483  6486  6489  6492  6494  6500  6503  6507  6511  6516
        6519  6523  6531  6535  6547  6554  6563  6568  6573  6576  6580
.BLKB    1239  1299  1305  1312  1375  1397  5285  5565  5627
.BLKW    1248  1275  1289  1386  1391  1402  1408  1415  1419  1422  1427  5456
.BYTE     850   851   856   857   865   866   874   875   876   877  1233  1234  1362  1363  1452
        1453  1454  1455  1685  1686  2693  2694  3537  3538  4057  4058  4736  4737  4888  4889
        5283  5284  5964  5965  5966  5967
.DSABL  5223
```

```
.ENABL      1   585  5156
.END     6611
.ENDC     591   604   605   606   607   623   715   729   760   774   778   780   795   817   839
          844   848   850   878   889   890   891   895  1556  1565  1566  1569  1571  1573  1575
         1591  1595  1599  1605  1607  1791  1797  1804  1816  1843  1844  1850  1851  1852  1881
         1882  1886  1887  1888  1977  1978  1986  1987  1988  2019  2020  2025  2026  2027  2149
         2150  2159  2160  2161  2293  2294  2300  2301  2302  2402  2403  2419  2420  2421  2690
         2735  3608  3757  3758  3763  3764  4332  4353  4357  4367  4411  4618  4625  4632  4645
         4651  4657  5054  5127  5129  5131  5134  5139  5142  5143  5145  5151  5153  5156  5157
         5159  5187  5223  5227  5255  5256  5263  5265  5268  5270  5286  5292  5295  5297  5330
         5333  5393  5460  5480  5530  5569  5660  5794  5895  6172  6188  6191  6194  6199  6201
         6212  6213  6215  6220  6221  6225  6228  6231  6237  6240  6252  6253  6254  6255  6256
         6257  6258  6259  6260  6261  6262  6263  6264  6265  6266  6267  6268  6269  6270  6275
         6284  6285  6291  6297  6298  6308  6310  6317
.EQUIV    623   624   632   647   648   677   678   679   680   681   682   683   684   685   686
          705   706   707   708   709   710   711   712   713   714   731   732   733   736   737
          738   739   740   741   745   746   747
.EVEN    1456  1548  1607  1791  1797  1804  1816  2690  2735  3608  4618  4625  4632  4645  4651
         4657  4679  5054  6172  6316  6589
.IF       587   603   604   605   606   621   687   715   758   773   776   778   795   806   828
          843   847   849   878   889   890   894   895  1556  1560  1565  1567  1569  1571  1573
         1591  1594  1595  1596  1599  1606  1790  1796  1803  1815  1842  1844  1850  1852  1880
         1882  1886  1888  1976  1978  1986  1988  2018  2020  2025  2027  2148  2150  2159  2161
         2292  2294  2300  2302  2401  2403  2419  2421  2689  2734  3607  3756  3758  3762  3764
         4331  4335  4353  4356  4367  4617  4624  4631  4644  4650  4656  5053  5126  5127  5128
         5129  5130  5131  5133  5138  5141  5143  5145  5151  5153  5155  5157  5158  5159  5187
         5226  5227  5255  5263  5264  5268  5269  5285  5286  5292  5294  5297  5309  5332  5392
         5459  5480  5529  5568  5659  5793  5894  6171  6187  6190  6194  6199  6211  6213  6214
         6215  6220  6221  6222  6227  6228  6230  6236  6240  6244  6253  6254  6255  6256  6257
         6258  6260  6261  6262  6263  6264  6265  6266  6267  6268  6269  6270  6274  6284  6285
         6290  6297  6298  6306  6308  6310  6314
.IFF       603   605   606   621   774   778   780   844   847   849   878   895  1565  1594  1595
          1843  1844  1851  1852  1881  1882  1887  1888  1977  1978  1987  1988  2019  2020  2026
          2027  2149  2150  2160  2161  2293  2294  2301  2302  2402  2403  2420  2421  3758  3764
          4332  4344  4356  4399  5127  5130  5134  5138  5141  5153  5156  5159  5227  5229  5234
          5255  5256  5265  5269  5286  5295  5333  5393  5460  5530  5569  5660  5794  5895  6188
          6212  6213  6214  6228  6231  6237  6275  6291  6308
.IFT      1607  1791  1797  1804  1816  2690  2735  3608  3758  3764  4347  4360  4403  4410  4618
          4625  4632  4645  4651  4657  5054  5229  5234  5313  5329  5330  6172  6220
.IFTF     1607  1791  1797  1804  1816  2690  2735  3608  3757  3763  4344  4349  4399  4406  4618
          4625  4632  4645  4651  4657  5054  5174  5227  5230  5309  5313  5329  6172  6219
.IIF       586   591   596   601   602   603   604   607   608   609   610   611   612   613   755
          894  1566  1569  1573  1574  1595  5134  5135  5153  5156  5177  5278  5286  5292  5390
         5527  6191  6192  6193  6194  6198  6220  6225  6228  6252  6253  6254  6255  6256  6258
         6260  6261  6262  6263  6264  6265  6266  6267  6268  6269  6270
.IRP      1556  1842  1880  1976  2018  2148  2292  2401  5304  5325  5346  5379  5403  5443  5672
         5700  5809  5829  6278  6284  6297  6298
.LIST        1   585   617   729   755   878   880   881   882   883   884   885   886   887   888
          889  1556  1575  1595  1596  1607  1791  1797  1804  1816  1842  1852  1880  1888  1976
         1988  2018  2027  2148  2161  2292  2302  2401  2421  2690  2735  3608  3756  3762  4618
         4625  4632  4645  4651  4657  5054  5134  5145  5255  6172  6194  6244  6252  6253  6254
         6255  6256  6257  6258  6259  6260  6261  6262  6263  6264  6265  6266  6267  6268  6269
         6270  6271
.MACRO       1   607   617   841  1842  1880  1976  2018  2148  2292  2401  6244
.MCALL     585   729  1575  1596
.NLIST       1   585   617   729   755   878   880   881   882   883   884   885   386   887   888
```

```
             889   1556   1575   1595   1596   1607   1791   1797   1804   1816   1842   1852   1880   1888   1976
            1988   2018   2027   2148   2161   2292   2302   2401   2421   2690   2735   3608   3756   3762   4618
            4625   4632   4645   4651   4657   5054   5134   5145   5255   6172   6194   6244   6252   6253   6254
            6255   6256   6257   6258   6259   6260   6261   6262   6263   6264   6265   6266   6267   6269   6269
            6270   6271
.NTYPE      3756   3762
.PAGE        617    841    895   1206   1235   1290   1313   1367   1420   1457   1549   1613   1641   1736   1831
            1842   1880   1976   2018   2088   2122   2148   2292   2401   2546   2624   2711   2799   2954   3039
            3060   3113   3219   3280   3325   3432   3472   3529   3574   3591   3879   3911   3959   3998   4017
            4096   4149   4218   4329   4411   4458   4509   4545   4594   4680   4723   4795   4905   4928   4970
            5002   5029   5058   5073   5108   5124   5153   5292   5330   5390   5457   5527   5566   5628   5657
            5704   5791   5836   5892   5970   6060   6116   6141   6185   6228   6272   6317   6522
.REM            1
.REPT         755    880
.SBTTL        597    619    749    758    771    783    841    895   1559   1591   1596   1842   1880   1976   2018
            2148   2292   2401   2546   4329   5002   5029   5058   5073   5124   5153   5292   5330   5390   5457
            5527   5566   5628   5629   5657   5704   5791   5836   5892   5970   6060   6185   6228   6244   6272
.TITLE        586
.WORD         755    756    757    779    845    852    853    854    855    858    859    860    861    862    863
             864    867    868    869    878    880    881    882    883    884    885    886    887    888   1206
            1207   1208   1209   1210   1211   1212   1213   1215   1217   1218   1220   1223   1226   1241   1313
            1317   1322   1324   1328   1329   1332   1333   1334   1335   1338   1339   1340   1341   1342   1343
            1344   1350   1351   1352   1353   1354   1355   1365   1366   1377   1378   1379   1380   1420   1421
            1425   1426   1429   1430   1431   1432   1433   1434   1435   1436   1439   1441   1443   2671   2672
            3044   3045   4362   4409   4410   5069   5138   5141   5152   5329   5387   5524   5602   5653   5968
            6083   6101   6140   6307   6309   6591   6594   6596   6599   6602
```

```
ERRORS DETECTED:  0
DEFAULT GLOBALS GENERATED:  0

*,DZRKHE.SEQ/SOL/CRF/PAGNUM/NL:TOC=SYSMAC.CO,DZRKHE.P11
RUN-TIME:  46 71 10 SECONDS
RUN-TIME RATIO:  241/128=1.8
CORE USED:  34K  (67 PAGES)
```

Spooler runtime 25 Seconds, 105 KCS, 5% disk reads, 4 disk writes, 181 pages