

IDENTIFICATION

-----

PRODUCT CODE:           MAINDEC-11-DZRXA-E-0  
PRODUCT NAME:           RX11 SYSTEM RELIABILITY TEST  
DATE:                    APRIL 1976  
MAINTAINER:             DIAGNOSTIC ENGINEERING  
AUTHOR:                  DAVID L. ADAMS

COPYRIGHT (C) 1975,1976  
DIGITAL EQUIPMENT CORPORTION, MAYNARD, MASS.

THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO THESE LICENSE TERMS. TITLE TO AND OWNERSHIP OF THE SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.

THE INFOMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATIUN.

DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFRWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DEC.

## TABLE OF CONTENTS

-----

1.0	GENERAL PROGRAM INFORMATION
1.1	ABSTRACT
1.2	SYSTEM REQUIREMENTS
1.2.1	HARDWARE
1.2.2	SOFTWARE
2.0	OPERATING INSTRUCTIONS
2.0.1	OUTLINE OF OPERATING PROCEDURE
2.1	LOADING PROCEDURE
2.2	STARTING ADDRESSES
2.3	OPERATOR ACTION BEFORE STARTING PROGRAM
2.3.1	DEVICE ADDRESS SELECTION
2.3.2	NON-STANDARD DISKETTE ADDRESS SELECTION
2.3.3	SOFTWARE SWITCH REGISTER (LOC. 176)
2.3.4	TEST PARAMETER SELECTION ("DTESTP" LOC. 1212)
2.4	OPERATOR ACTION TO RUN THE PROGRAM
2.4.1	STARTING THE PROGRAM
2.4.2	OPERATING CONDITIONS
2.4.3	ACT11 & XXDP HOOKS
2.5	PROGRAM OPTIONS
2.5.1	PSEUDO SCOPE LOOP
2.5.2	DISKETTE COMPATABILITY
2.5.3	RX11 TO RX8 COMPATABILITY
2.6	RUN TIME
3.0	ERROR DETECTION
3.1	ERROR DEFINITIONS
3.2	DEFINITIVE ERROR CODES
3.3	UNEXPECTED OR MISSING ERROR CONDITIONS
3.4	POWER FAILURE
3.5	PROGRAM HUNG
4.0	ERROR REPORTING
5.0	HALTS

1.0 GENERAL PROGRAM INFORMATION

1.1 ABSTRACT

THE RX11 SYSTEM RELIABILITY PROGRAM CONSISTS OF SELECTABLE TESTS THAT CHECK THE OPERATION OF THE RX11 SYSTEM, BY WRITING, READING AND VERIFYING VARIOUS DATA PATTERNS, UNDER VARIOUS (SELECTABLE) HEAD MOVEMENTS. IT CAN TRANSFER DATA AND CHECK FOR ERRORS OVER THE ENTIRE DISKETTE, ALL TRACKS AND SECTORS, OR BETWEEN SEPARATELY SELECTABLE TRACK AND SECTOR ADDRESS LIMITS.

AS WELL AS TRANSFERRING DATA THE PROGRAM ACCUMULATES STATISTICAL DATA ON THE FOLLOWING:

- A. COMPLETED PASSES OF THE PROGRAM
- B. NUMBER OF RESTARTS
- C. NUMBER OF SECTORS WRITTEN/READ
- D. RECOVERABLE AND UNRECOVERABLE FAULTS AS FOLLOWS:

- 1. PARITY ERRORS
- 2. ERROR FLAG ERRORS
- 3. INTERRUPT ERRORS
- 4. SEEK ERRORS
- 5. DATA CRC ERRORS
- 6. CRC NO DATA ERRORS
- 7. DATA NO CRC ERRORS
- 8. DELETED DATA MARK ERRORS
- 9. WRITE ERRORS
- 10. READ ERRORS

- E. TOTAL OF EACH ERROR CODE DETECTED
- F. TOTAL OF TIMES EACH TRACK WAS ACCESSED
- G. TOTAL OF TIMES HEAD MOVED TO A TRACK.
- H. TOTAL OF ERRORS DETECTED PER TRACK PER DRIVE.

THE ABOVE INFORMATION IS REPORTED BY RUNNING THE ERROR DUMP PROGRAM.

1.2 SYSTEM REQUIREMENTS

1.2.1 HARDWARE REQUIREMENTS

THE FOLLOWING EQUIPMENT IS REQUIRED.

- A. PDP-11 SERIES OF COMPUTER WITH MIN 8K MEMORY
- B. RX11 FLOPPY DISK SYSTEM INCLUDING DUAL OR SINGLE DRIVE RX01 AND PDP-11 INTERFACE. (SEE SECTION 2.3 FOR SELECTION OF REGISTER ADDRESSES AND VECTOR ADDRESS)  
NOTE: A DISKETTE MUST BE INCLUDED WITH EACH DRIVE TESTED.
- C. CONSOLE TELEPRINTER

1.2.1 SOFTWARE REQUIREMENTS

THIS PROGRAM ASSUMES THAT THE RX11 INTERFACE DIAGNOSTIC ( MAINDEC = 11 - DZRXB-\* ) HAS BEEN SUCCESSFULLY RUN ON THIS SYSTEM.

## 2.0 OPERATING INSTRUCTIONS

### 2.0.1 OUTLINE OF OPERATING PROCEDURE

THE STANDARD OPERATING PROCEDURE FOR THE RX11 RELIABILITY TEST (TO RUN ALL TESTS ON BOTH DRIVES WITH NO OPERATOR INTERVENTION VIA THE SWITCH REGISTER) IS AS FOLLOWS:

- A. LOAD THE PROGRAM INTO MEMORY
  - 1. IF IT'S BEING LOADED FROM A DISKETTE, REPLACE THE "LIBRARY DISKETTE" WITH A "SCRATCH DISKETTE"
- B. START THE PROGRAM RUNNING AT LOCATION 200
- C. THE PROGRAM WILL TYPE OUT THE FOLLOWING:
  - 1. PROGRAM NAME AND REVISION
  - 2. RX11 REGISTER AND VECTOR ADDRESSES
  - 3. UNITS BEING TESTED
  - 4. "P"ATTERN, "T"EST, AND "S"EQUENCEIT THEN STARTS RUNNING UNDER THOSE CONDITIONS
- D. IF THERE ARE NO ERRORS, AT THE END OF THE COMPLETED PASS A "D" AND "BELL" IS TYPED, AND THE PROGRAM WILL CONTINUE ON FOR ANOTHER PASS.
- E. HALT THE PROGRAM AS FOLLOWS:
  - 1. IF THERE IS A HARDWARE SWITCH REGISTER, PUT SW 14 UP TO HALT AT THE END OF PASS.
  - 2. IF THERE IS NO SWITCH REGISTER, OR TO HALT AT ANY TIME, HALT THE PROCESSOR.

### 2.1 LOADING PROCEDURE

LOAD THE PROGRAM INTO MEMORY USING STANDARD PROCEDURE FOR BINARY PAPER TAPES.

MAKE SURE THAT THE TOTAL SYSTEM IS READY FOR OPERATION, DISKETTE INSERTED CORRECTLY, DOORS CLOSED ON DRIVES TO BE TESTED, ETC.

### 2.2 STARTING ADDRESSES

THE PROGRAM HAS THREE (3) STARTING LOCATIONS.

#### 2.2.1 INITIAL START (LOC. 200)

THIS STARTING LOCATION INITIALIZES THE PROGRAM AS FOLLOWS:

- A. CLEARS ALL ERROR LOGS
- B. RESETS COUNTERS AND CONSTANTS
- C. TYPES OUT RX11 REGISTER AND VECTOR ADDRESSES BEING USED.
- D. TYPES THE DRIVE AND TEST SELECTION IN "DTESTP"
- E. INITIATES THE COLLECTION OF STATISTICAL DATA PERTAINING TO THE OPERATION OF THE RX11 SYSTEM.

2.2.2 RESTART (LOC. 202)

THIS STARTING LOCATION DIRECTS THE PROGRAM TO CONTINUE RUNNING USING DRIVE AND TEST SELECTIONS SPECIFIED IN THE PREVIOUS INITIAL START. IT ALSO CONTINUES TO ACCUMULATE STATISTICAL INFORMATION, AMENDING IT TO THE DATA ALREADY COLLECTED PRIOR TO THIS RESTART.

2.2.3 ERROR DUMP (LOC. 204)

THIS STARTING ADDRESS DIRECTS THE PROGRAM TO PRINT OUT ON THE CONSOLE TELEPRINTER ALL THE DATA ACCUMULATED FROM THE LAST "INITIAL START" TO THE TIME OF THE PRINTOUT. TYPING OUT THIS INFORMATION DOES NOT DESTROY IT, SO A RESTART CAN BE INITIATED AND INFORMATION WILL CONTINUE TO BE COLLECTED.

2.3 OPERATOR ACTION BEFORE STARTING PROGRAM

2.3.1 DEVICE ADDRESS SELECTION

LIKE MOST OPTIONS ON THE PDP11 THE RX11 INTERFACE CARD HAS JUMPERABLE REGISTER AND VECTOR ADDRESSES. THIS ALLOWS FOR DEVICES WITH THE SAME STANDARD ADDRESSES TO BE JUMPERED TO AN OTHER ADDRESS SO THEY WILL RUN WITHOUT CONFLICT.

THE PROGRAM MUST KNOW WHAT ADDRESSES ARE BEING USED, AS IT IS THROUGH THESE REGISTERS AND VECTOR THAT ALL COMMUNICATION BETWEEN THE PDP11 AND RX11 IS HANDLED.

IF THE RX11 SYSTEM UNDER TEST IS JUMPERED FOR REGISTER ADDRESS OTHER THAN STANDARD, WHICH IS RXCS=177170 AND RXDB=177172. PLACE IN THE MEMORY LOCATION CALLED "RXCS" (LOC. 1206) ITS NEW ADDRESS, AND IN LOCATION "RXDB" (LOC. 1210) ITS NEW ADDRESS. IF THERE AS A NONSTANDARD INTERRUPT VECTOR ADDRESS (STANDARD IS LOC 264) THEN PLACE IN MEMORY LOCATION CALLED "INTVEC" (LOC. 1204) ITS NEW ADDRESS.

IF THESE THREE MEMORY LOCATIONS DO NOT CONTAIN THE ADDRESSES THAT THE INTERFACE BOARD IS WIRED TO THE PROGRAM WILL REPORT "TEST HUNG" AND HALT, OR HALT AT THE VECTOR ADDRESS THAT IS JUMPERED IN. (NOTE: THE VECTOR ADDRESSES CAN NOT BE JUMPERED FOR LOCATIONS 200 THROUGH 220 AS THESE ARE USED BY THE PROGRAM)

2.3.2 NON-STANDARD DISKETTE ADDRESS SELECTION

IF IT IS DESIREABLE TO TEST THE DISKETTES BETWEEN TRACK AND SECTOR ADDRESS LIMITS OTHER THAN BETWEEN THE NORMAL OUTER DIAMETER (OD) AND INNER DIAMETER (ID) TRACK ADDRESSES, AND/OR MINIMUM (FIRST) AND MAXIMUM (LAST) SECTOR ADDRESS. THIS IS DONE BY THE OPERATOR MAKING CHANGES TO TWO (2) MEMORY LOCATIONS BEFORE THE PROGRAM IS STARTED. ONE LOCATION IS CALLED OD WHICH CONTAINS TWO BYTES ONE FOR OD AND THE OTHER ID TRACK ADDRESSES. THE OTHER LOCATION IS CALLED FIRST AND IT TOO CONTAINS TWO BYTES ONE FOR FIRST THE OTHER FOR LAST SECTOR ADDRESS. (IF THESE TWO LOCATIONS ARE LEFT CLEARED THE MAX AND MIN VALUES FOR THE ADDRESSES ARE ASSUMED.)

A. DEFINITIONS:

OD = ADDRESS OF TRACK AT OUTER DIAMETER (MIN VALUE=0)  
ID = ADDRESS OF TRACK AT INNER DIAMETER (MAX VALUE=114 [OCTAL])  
FIRST = ADDRESS OF FIRST SECTOR OF TRACK (MIN VALUE=1)  
LAST = ADDRESS OF LAST SECTOR OF TRACK (MAX VALUE=32 [OCTAL])

B. LOCATIONS:

TRACKS LOC. 1200 BITS	14-----8	6-----0
	ID	OD
SECTORS LOC. 1202 BITS	12-----8	4-----0
	LAST	FIRST

C. RESTRICTIONS:

THE CONTENTS OF "OD" MUST BE LESS THAN OR EQUAL TO THE CONTENTS OF "ID"  
THE CONTENTS OF FIRST MUST BE LESS THAN OR EQUAL TO THE CONTENTS OF "LAST"

IF THESE LOCATIONS ARE CHANGED TO NEW LIMITS, THEN THE PROGRAM WILL ACCESS ONLY THOSE ADDRESSES INCLUSIVE OF AND BETWEEN THESE LIMITS.

2.3.3 SOFTWARE SWITCH REGISTER (LOC. 176)

FOR THE PDP 11 PROCESSORS THAT DO NOT HAVE A HARDWARE SWITCH REGISTER, OR IF THE OPERATOR WISHES TO SELECT THE SOFTWARE SWITCH REGISTER, BY PUTTING ALL THE SWITCHES UP TO A "1", (THIS MUST BE DONE EACH TIME THE PROGRAM IS STARTED AT LOCATION 200, OTHERWISE THE PROGRAM WILL USE THE HARDWARE SWR.) LOCATION 176 IS ASSIGNED AS THE SWITCH REGISTER. BITS SET TO A "1" IN THIS LOCATION HAVE THE SAME FUNCTION AS THE CORRESPONDING SWITCH IN THE HARDWARE SWITCH REGISTER. ALL REFERENCES TO THE SWR ARE INDIRECT AND THE PROGRAM ASSIGNS THE CORRECT ADDRESS OF THE SWR AT "INITIAL START". SEE SECTION 2.4.2 FOR THE SELECTION OF OPERATING CONDITIONS.

TO CHANGE THE SOFTWARE SWR, WHILE THE PROGRAM IS RUNNING, TYPE "CONTROL G". EACH TIME THE SWR IS TO BE TESTED THE PROGRAM WILL CHECK TO SEE IF THE SOFTWARE SWR IS SELECTED, AND THE PROGRAM IS NOT RUNNING IN AUTO MODE OF RXDP/ACT11. IF BOTH CONDITIONS EXIST THEN THE PROGRAM CHECKS FOR THE CTRL G IN THE KEYBOARD BUFFER. IF THE CTRL G IS THERE THE CONTENTS OF THE SOFTWARE SWR ARE PRINTED AND A "NEW =" IS ASKED FOR. THE OPERATOR MAY NOW TYPE IN THE NEW SWITCH REGISTER CONTENTS, TERMINATED BY A CARRIAGE RETURN (CR), OR IF HE DOESN'T WANT TO CHANGE THE SWR, JUST TERMINATE WITH THE (CR). NOTE SEE THE CHARACTER RESTRICTIONS BELOW.

WHEN THE PROGRAM DETECTS THE (CR) IT WILL REPLACE THE CONTENTS OF THE SOFTWARE SWR, IF A NEW ONE HAS BEEN TYPED IN, AND RETURN TO THE FLOW OF THE PROGRAM.

NOTE: CHARACTER RESTRICTIONS FOR CHANGING THE SOFTWARE SWR.

1. ONLY OCTAL NUMBERS 0 - 7 ARE ACCEPTED. ANY OTHER CHARACTER TYPED WILL BE PRINTED AS A ? AND THE WHOLE SWR MUST BE RETYPED.
2. TO WIPE OUT A "NEW" CONTENTS JUST TYPED IN, TYPE CTRL U. NOW A NEW CONTENTS CAN BE RETYPED.
3. ONLY 6 OCTAL CHARACTERS WILL BE PUT INTO THE SWR. IF MORE THAN 6 CHARACTERS ARE TYPED IN ONLY THE LAST 6 WILL BE PUT INTO THE SWR.

2,3,4 TEST PARAMETER SELECTION ("DTESTP" LOC. 1212)

THE DRIVE AND TEST SELECTION MUST BE DONE BEFORE THE PROGRAM STARTS. "DTESTP" (LOCATION 1212) IS WHERE THE BITS ARE SET TO TELL THE PROGRAM WHAT DRIVES ARE WANTED AND WHAT TEST TO RUN, AS INDICATED BELOW. WHEN THE PROGRAM STARTS IT WILL TYPE OUT THE TEST CONDITIONS IT IS RUNNING UNDER.

BIT 15 (1) SELECT DRIVE UNIT 1  
BIT 14 (1) SELECT DRIVE UNIT 0

THEN SET THE TEST CONDITIONS IN BITS 8 THROUGH 0 AS SHOWN BELOW.

"DTESTP" BITS 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0  
U1 U0 NOT USED U P P P T T T S S S

BIT 9 IF BIT 9 IS ON, ALL WRITE/READ FUNCTIONS WILL BE IN THE DELETED DATA MODE.  
BIT 8,7,6 SELECTS A DATA PATTERN TO BE USED.  
BIT 5,4,3 SELECTS A TEST TO BE PERFORMED.  
BIT 2,1,0 SELECTS A HEAD MOVEMENT SEQUENCE.

THE SELECTIONS ARE DEFINED AS FOLLOWS:

P = DATA PATTERN SELECTION

0 DEFAULT TO 7  
1 ZEROS  
2 ONES  
3 FLOATING ZERO  
4 FLOATING ONE  
5 125  
6 314  
7 RANDOM

T = FUNCTIONAL TESTS

0 DEFAULT TO 7  
1 WRITE ONLY  
2 WRITE/READ  
3 WRITE/READ CHECK  
4 READ CHECK ONLY  
5 READ ONLY (CRC CHECK)  
6 WRITE/READ CHECK ON ALTERNATING DRIVES \*  
7 WRITE/READ/READ CHECK \*\*

\* NOTE: TEST 6 WRITES THEN READ CHECKS ANY SELECTED DATA PATTERN USING ANY TRACK SEQUENCE, BUT ONE TRACK AT A TIME. FIRST ON UNIT 0 THEN UNIT 1. WHEN BOTH UNITS HAVE ACCESSED THAT TRACK, IT GOES BACK TO UNIT 0 FOR THE NEXT TRACK, ETC.



\*\* NOTE: THE FIRST HALF OF TEST 7 FORCES THE TRACK SEQUENCE TO INCREMENT UP THROUGH ALL TRACKS DOING WRITE/READ CHECK FUNCTIONS. THIS VERIFIES THAT ALL TRACKS ARE ACCESSABLE. THE SECOND HALF OF THE PASS WILL USE THE SEQUENCE SELECTED BY THE OPERATOR AS INDICATED BELOW, AND ONLY READ CHECK THE DATA JUST WRITTEN. THIS VERIFIES THAT THE DATA CAN BE READ FROM A TRACK AFTER THE HEAD HAS BEEN MOVED AWAY FROM AND BACK TO THAT TRACK. AT THE COMPLETION OF THE PASS THE DELETED DATA BIT IN TEST CONDITIONS IS COMPLEMENTED AND THE NEXT PASS WILL BE RUN UNDER THIS NEW CONDITION.

S = TRACK SEQUENCING

0	DEFAULT TO 7
1	INCREMENT
2	DECREMENT
3	INCREMENT/DECREMENT
4	BOUNCE
5	DECREASING BOUNCE
6	STROBE
7	RANDOM

IF NO BITS ARE SET (ZEROED "DTESTP:") THEN THE PROGRAM WILL SELECT ALL DRIVE UNITS THAT ARE READY AND DEFAULT TO TEST CONDITIONS AS INDICATED.

THE PROGRAM NEXT PRINTS THE REGISTER AND VECTOR ADDRESSES IT WILL USE IN COMMUNICATING WITH THE RX11 SYSTEM, AS EXPLAINED IN SECTION 2.3.1 THE OPERATOR MUST VERIFY THAT THESE ADDRESSES ARE THE ONES JUMPERED ON THE RX11 INTERFACE BOARD.

2.4 OPERATOR ACTION TO RUN THE PROGRAM

2.4.1 STARTING THE PROGRAM

SET THE DESIRED STARTING ADDRESS INTO THE SWITCH REGISTER, DEPENDING UPON THE TYPE OF CONSOLE AVAILABLE, LOAD ADDRESS, AND PRESS START.

THE PROGRAM WILL TYPE ITS "MAINDEC" NUMBER AND REVISION, AND DEPENDING UPON THE STARTING ADDRESS DO THE FOLLOWING:

SA200 - THE PROGRAM WILL TYPE DRIVE AND TEST PARAMETERS, THE TWO REGISTER ADDRESSES, AND VECTOR ADDRESS. IT WILL THEN BEGIN FUNCTIONAL TESTING, AND INFORMATION COLLECTION.

SA202 - THE PROGRAM WILL CONFIGURE TO CONDITIONS SET IN PREVIOUS "INITIAL START", PRINT OUT THESE CONDITIONS AND CONTINUE FUNCTIONAL TESTING AND DATA COLLECTING. THE ONLY OPERATOR ACTION REQUIRED IS THE DYNAMIC SELECTION OF OPERATING CONDITIONS IN THE SWITCH REGISTER AS REQUIRED.

SA204 - THIS PROGRAM WILL REPORT VIA THE TELEPRINTER ALL DATA COLLECTED. NO OTHER OPERATOR ACTION IS REQUIRED.

## 2.4.2 OPERATING CONDITIONS

THE PROGRAM CHECKS FOR OPERATING CONDITIONS AT VARIOUS POINTS WHILE RUNNING. IF THERE IS A HARDWARE SWR THESE CONDITIONS CAN BE CHANGED AND SET WHILE THE PROGRAM IS RUNNING. IF THE SOFTWARE SWR IS IN USE, THEN THESE CONDITIONS MUST BE SET IN LOCATION 176 BEFORE THE PROGRAM STARTS.

SW15 = HALT ON ERROR  
SW14 = HALT AT END OF PASS  
SW13 = DON'T PRINT ERROR MESSAGE  
SW12 = TYPE ONLY 10 DATA ERRORS  
SW11 = NO RETRY ON ERROR. LOG HARD ERROR  
SW08 = NO RECALIBRATION ON SEEK ERRORS  
SW15-SW0 (1) = SELECT SOFTWARE SWITCH REGISTER

NOTE: IF THERE IS A HARDWARE SWITCH REGISTER, AND THE OPERATOR WANTS THE SOFTWARE SWITCH REGISTER. PUT ALL SWITCHES UP (1) BEFORE STARTING THE PROGRAM AT THE INITIAL START ADDRESS.

THE PROGRAM WILL PRINT A "DRIVE(S)" SELECTED CONFIRMATION MESSAGE THAT IT WAS SUCCESSFUL IN FINDING AT LEAST ONE DRIVE READY (DRY) CONDITION ON AN OPERATOR SELECTED DRIVE OR EITHER OR BOTH IF NO SELECTION WAS MADE. THE DRIVES CONFIRMED AS BEING SELECTED BY THE PROGRAM MAY DIFFER FROM THAT SELECTED BY THE OPERATOR IF THE PROGRAM DETECTED A DRIVE TO BE NOT READY.

IF HOWEVER THERE ARE NO DRIVES IN THE DRIVE READY CONDITION THE PROGRAM WILL TYPE "NO DRIVES READY" AND HALT, AS IT CAN'T FUNCTION WITH NO DRIVES READY. WHEN THE REASON FOR ALL DRIVES TO BE NOT READY IS FOUND AND CORRECTED, THE OPERATOR MAY PRESS CONTINUE AND THE PROGRAM WILL GO BACK TO "INITIAL START" OR HE MAY RELOAD THE STARTING ADDRESS HIMSELF.

AS WELL AS "DRIVE(S)" SELECTED THE PROGRAM WILL TYPE OUT THE TEST PARAMETERS SELECTED OR DEFAULTED TO. IF NON-STANDARD TRACK AND/OR SECTOR ADDRESS LIMITS WERE SELECTED (SEE SECTION 2.3.2 OF THIS DOCUMENT) THESE LIMITS WILL ALSO BE PRINTED OUT FOR VERIFICATION BY THE OPERATOR.

## 2.4.3 ACT11 & XXDP HOOKS

THE PROGRAM HAS THE NECESSARY LOCATIONS SET UP FOR OPERATION UNDER ACT11 AND XXDP OPERATION. THE PROGRAM LOOKS AT THE LOADING MEDIA LOCATION AND IF IT CONTAINS THE NUMBER 10, INDICATING THE FLOPPY DISK LOADED THE PROGRAM, WILL TYPE THE FOLLOWING PROMPT MESSAGE & WAIT FOR A USER RESPONSE:

"CAUTION - IF YOU DESIRE TO TEST UNIT 0  
REPLACE LOAD MEDIUM WITH A SCRATCH DISKETTE  
THEN PRESS CONTINUE"

THUS, IF THE OPERATOR WISHES TO TEST DRIVE UNIT 0, AFTER LOADING THE PROGRAM, HE REMOVES THE LIBRARY DISKETTE FROM DRIVE 0, INSERTS A SCRATCH DISKETTE INTO DRIVE 0, AND PRESSES THE 'CONTINUE' SWITCH

## 2.5 PROGRAM OPTIONS

THERE ARE A COUPLE OF WAYS THE PROGRAM CAN BE SET UP TO RUN FOR OPTIONAL TESTING.

### 2.5.1 PSEUDO - SCOPE LOOP

BY SETTING THE ADDRESS LIMITS IN OD AND OR FIRST TO ONE OF TWO TRACKS AND OR SECTORS IT IS POSSIBLE TO PRODUCE A FAIRLY TIGHT PSEUDO - SCOPE LOOP. BY RUNNING ONLY A TEST THAT DOES WRITE ONLY OR READ ONLY ITS POSSIBLE TO SCOPE SPECIFIC FUNCTIONS, I.E. FILL BUFFER, WRITE, READ, EMPTY BUFFER.

### 2.5.2 DISKETTE COMPATABILITY

TO CHECK FOR DATA TRANSFER COMPATABILITY BETWEEN DRIVE UNITS USE THE FOLLOWING PROCEDURE. (IT IS ASSUMED THE SYSTEM HAS DUAL DRIVE RX01, IF NOT YOU MAY TEST COMPATABILITY BETWEEN DIFFERENT RX11 SYSTEMS BY RUNNING THE SAME TESTS ON BOTH.)

THIS TEST INSURES PROPER HEAD ALIGNMENT.

- A. HAVE DISKETTES IN BOTH DRIVES AND DRIVES READY.
- B. CLEAR THE UD/ID AND FIRST/LAST MEMORY LOCATION, FOR TOTAL DISKETTE TRANSFERS.
- C. LOAD THE INITIAL START ADDRESS (LOC 200) AND START THE PROGRAM RUNNING.
- D. SET LOCATION "DTESTP" FOR "DRIVE AND TEST CONDITIONS" ON BOTH DRIVES, ANY "P"ATTERN OF DATA, "S"EQUENCE #1 (INCRIMENT TRACKS, TO INSURE ALL HEAD POSITIONS ARE ALIGNED) AND "T"EST 3 (WRITE/READ CHECK). THIS WILL WRITE THEN READ AND VERIFY THE DATA ON BOTH DISKETTES.
- E. ALLOW THE PROGRAM TO RUN FOR AT LEAST 1 COMPLETE PASS.

NOTE: IF RANDOM PATTERN (0 OR 7) IS SELECTED YOU MUST HALT AT THE END OF THE FIRST PASS, AS ADDITIONAL PASSES CHANGES THE DATA AND YOU WILL GET DATA ERRORS WHEN YOU TRY TO REREAD.

TO HALT AT THE END OF PASS PUT SW14 UP (1) WHEN "OPERATING CONDITIONS" ARE REQUESTED BY THE PROGRAM.

- F. AFTER COMPLETION OF THE PASS, PROGRAM HALTED. SWAP DISKETTES, AND AS YOU ONLY WANT TO READ VERIFY THE DATA, START THE PROGRAM AGAIN AT LOC 200 (INITIAL START).
- G. WHEN REQUESTED SELECT THE SAME DRIVES, PATTERN, AND SEQUENCE, BUT SELECT TEST 4 (READ CHECK ONLY).
- H. ALLOW THE PROGRAM TO RUN AS LONG AS YOU WISH TO VERIFY THAT DATA WRITTEN AND CHECKED ON ONE DRIVE CAN BE READ AND VERIFIED ON THE OTHER DRIVE.

2.5.3 RX11 TO RX8 COMPATABILITY

TO WRITE A DISKETTE ON THE RX11 AND READ/VERIFY THE SAME DATA ON A RX8 REQUIRES THE FOLLOWING:

A. SET UP LOCATION "DTESTP" (1212) WITH ONE OF THE FOLLOWING DATA PATTERNS

P=1 (0'S)  
P=2 (1'S)  
P=5 (125) \*  
P=6 (314) \*

\* NOTE: IF ONE OF THESE PATTERNS IS SELECTED A MODIFICATION TO THE PROGRAM MUST BE MADE TO ALLOW THIS DATA TO BE READ ON A RX8 SYSTEM. THE MODIFICATION IS IN THE "PAT125" ROUTEEN, THE TWO (2) LOCATIONS CONTAINING THE "COMB DATABYTE" INSTRUCTION MUST BE CHANGED TO TWO (2) NOP'S (OCTAL 000240)

B. SET IN "DTESTP" TEST 3 WHICH WILL WRITE AND READ CHECK THE DATA THEREBY VERIFYING THE QUALITY OF THE DATA PRIOR TO THE DISKETTE SWAP.

SET THE SEQUENCE TO 1 TO INSURE THAT ALL TRACKS HAVE BEEN WRITTEN ON.

C. HALT THE PROGRAM AT THE COMPLETION OF ONE PASS.

D. THE DISKETTE IS NOW READY TO BE READ ON THE RX8 SYSTEM. SEE MD-8-DIRX8-\* FOR THE PROCEDURE TO READ THIS DISKETTE.

TO READ A DISKETTE THAT WAS WRITTEN ON A RX8 SYSTEM SET UP AS FOLLOWS:

E. SET IN "DTESTP" THE DATA PATTERN THAT WAS USED BY THE RX8 SYSTEM. (SEE NOTE UNDER SECTION A.)

F. SET "DTESTP" FOR TEST 4 (READ CHECK ONLY) AND SEQUENCE 1 TO CHECK ALL TRACKS. START THE PROGRAM, IT SHOULD RUN WITHOUT DATA ERRORS.

## 2.6

### RUN TIME

RUN TIME PER PASS DEPENDS UPON NUMBER OF A FUNCTIONS IN THE TEST SELECTED AND THE TRACK SEQUENCE.

EXAMPLES OF RUN TIMES FOLLOW.

(THESE TIMES ARE FOR 1 DRIVE AND COMPLETE DISKETTE, MAXIMUM ID/OD, FIRST/LAST LIMITS. IF OPERATING ON 2 DRIVES DOUBLE THE TIME.) THESE TIMES ARE FOR A PDP 11/05 PROCESSOR AND MAY CHANGE SLIGHTLY FOR A FASTER PROCESSOR.

TEST SELECTION	P	T	S	TIME/PASS
	7	6	5	3 MIN. 30 SEC.
	7	7	4	2 MIN. 47 SEC.
	7	7	7	2 MIN. 23 SEC.
	6	3	4	1 MIN. 46 SEC.
	7	3	1	1 MIN. 30 SEC.
	7	1	1	51 SEC.

NOTE: DUE TO THE SLOW SPEED OF THE LSI 11 PROCESSOR, THE RUN TIME IS ABOUT DOUBLE THAT LISTED ABOVE. TO SPEED UP THE RUNNING OF THIS PROGRAM IN THE LSI 11 YOU CAN CHANGE THE INTERLEAVE FACTOR, OF THE SECTORS, USED IN THE PROGRAM. TO DO THIS CHANGE THE CONTENTS OF LOCATION "THREE" FROM OCTAL 3 TO OCTAL 5.

## 3.0 ERROR DETECTION

### 3.1 PROGRAM DEFINITIONS

ON MOST ERRORS THE PROGRAM WILL TYPE OUT THE CONTENTS OF "STATUS A" AND "STATUS B".

STATUS A IS THE CONTENTS OF THE RXES (ERROR AND STATUS REGISTER) AT THE TIME THE ERROR IS DETECTED. IT SHOWS THE CRC, PAR, ETC. ERRORS.

STATUS B IS THE "DEFINITIVE ERROR CODES" THAT THE RX01 DETECTED, THAT MAY HAVE CAUSED THE ERROR CONDITION. THESE ERROR CODES ARE DEFINED IN SECTION 3.2.

THE PROGRAM HAS DEFINED THE FOLLOWING AS ERRORS.

#### 3.1.1 WRITE ERROR

A WRITE ERROR IS A RETRIED READ ERROR IF THE DATA BEING READ IS OF UNKNOWN QUALITY (THE DATA READ IS BEING READ FOR THE FIRST TIME AFTER A WRITE FUNCTION)

#### 3.1.2 READ (CRC) ERROR

A READ ERROR IS A RETRIED READ ERROR WHERE THE QUALITY OF THE DATA BEING READ IS KNOWN GOOD. (THE DATA HAS BEEN READ CORRECTLY SOME TIME PREVIOUSLY.)

- 3.1.3 CRC AND DATA ERROR
- 3.1.4 NO CRC ERROR BUT DATA ERROR
- 3.1.5 CRC ERROR BUT NO DATA ERROR

THE ABOVE THREE ERRORS ARE DETECTED WHEN THE PROGRAM IS VERIFYING THE DATA READ OFF THE DISKETTE AGAINST THE DATA THAT SHOULD HAVE BEEN READ.

UPON A NON-COMPARISON THE PROGRAM TYPES OUT THE "BYTE" NUMBER IN THE SECTOR, THE DATA READ FROM THE DISKETTE "BAD" AND THE EXPECTED DATA "GOOD".

BYTE#      BAD      GOOD  
 (THE DATA PATTERNS ARE FORMATTED AS SHOWN)

0            (TRACK ADDRESS; BITS 6 - 0)  
 1            (SECTOR ADDRESS; BITS 4 - 0)

BYTES 2 THROUGH 125 CONTAIN THE SELECTED "P"ATTERN.

126          (THE SUM OF ALL BYTES 0 - 125)  
 127          (THE NEGATIVE OF 2 TIMES BYTE 126)

THE PROGRAM DETECTS A CHECKSUM ERROR BY SUMMING ALL THE DATA READ FROM THE DISKETTE AND COMPARING THAT SUM TO 0.

AT THE END OF THE DATA ERROR TYPEOUT THE PROGRAM TYPES OUT IF THE CHECK SUM ACCUMULATED IS "GOOD"OR HAD "ERRORS". IF BYTES 0 OR 1 HAVE DATA ERRORS THE OPERATOR MUST CHECK THE RESULTS OF THE CHECK SUM. IF IT IS ALSU BAD, THEN THERE WAS A TRUE DATA ERROR. IF THE CEHCK SUM IS GOOD, THEN IT MIGHT BE THAT THE HEAD IS NOT OVER THE TRACK EXPECTED, AND THERE IS A POSITIONING ERROR.

- 3.1.6 SEEK ERROR

A SEEK ERROR IS DEFINED AS NOT A CRC AND NOT A PARITY ERROR. A PROGRAMED RECALIBRATE IS ISSUED TO TRY TO CORRECT EACH SEEK ERROR.

- 3.1.7 PARITY ERROR

A PARITY ERROR RESULTS FROM AN INCORRECT TRANSFER OF A COMMAND WORD FROM THE RX11 INTERFACE TO THE RX01 MICRO-PROCESSOR CONTROLLER.

### 3.2 DEFINITIVE ERROR CODES

THE RX01 MICROCONTROLLER HAS DEFINED THE ERROR CODES AND MEANINGS WHICH ARE AVAILABLE TO THE PROGRAM BY ISSUING COMMAND #7 "READ THE B-CODE STATUS REGISTER".

A DEFINITIVE ERROR CODE REPRESENTS [WHERE] WITHIN A FUNCTION AN ERROR WAS DETECTED.

THE FOLLOWING ARE THE DEFINITIVE ERROR CODES AND MEANINGS:

- 00 - NO ERROR
- 10 - DRIVE 0 FAILED TO SEE HOME FROM INITIALIZE
- 20 - DRIVE 1 FAILED TO SEE HOME FROM INITIALIZE
- 30 - HOME FOUND WHEN STEPPING OUT 10 TRACKS FROM INIT
- 40 - TRIED TO ACCESS A TRACK GREATER THAN 17(DECIMAL)
- 50 - HOME WAS FOUND BEFORE DESIRED TRACK
- 60 - SELF DIAGNOSTIC ERROR
- 70 - DESIRED SECTOR NOT FOUND AFTER SAMPLING 52 HEADERS
- 100 - WRITE PROTECT ERROR
- 110 - MORE THAN 40US AND NO SEP CLOCK DETECTED
- 120 - A PREAMBLE COULD NOT BE FOUND
- 130 - PREAMBLE FOUND BUT NO ID MARK FOUND IN TIME
- 140 - CRC ERROR ON SUPPUSIPLY GOOD HEADER
- 150 - GOOD HEADER (NO CRC ERROR) BUT TRACK COMPARE ERROR
- 160 - IDAM NOT FOUND IN TIME
- 170 - DATA AM NOT FOUND IN TIME
- 200 - DATA CRC ERROR
- 210 - ALL PARITY ERRORS

### 3.3 UNEXPECTED OR MISSING ERROR CONDITIONS

#### 3.3.1 MISSING DD MARK

AN ERROR WHEN THE PROGRAM WROTE DELETED DATA INFORMATION BUT NO DELETED DATA MARK WAS DETECTED WHEN THE DATA WAS READ.

#### 3.3.2 UNEXPECTED DD MARK

AN ERROR WHEN A DELETED DATA MARK IS DETECTED BUT NO DELETED DATA WAS WRITTEN.

### 3.3.3 NO INTERRUPT ON DONE

THE INTERRUPT ENABLE BIT WAS SET AND THE DONE FLAG WAS SET BUT NO INTERRUPT OCCURRED.

### 3.3.4 UNKNOWN INTERRUPT

IF AN INTERRUPT OCCURS FROM ANY OTHER DEVICE WHILE THIS PROGRAM IS RUNNING IT WILL HALT AT THE INTERRUPT VECTOR LOCATION.

IF AN INTERRUPT OCCURS ON INTERRUPT VECTOR LOCATION 264 (RX11), AND THERE IS NO ERROR, DONE, OR ERROR STATUS CONDITION SET, THEN IT WILL BE TAGGED AS AN UNKNOWN INTERRUPT.

### 3.4 POWER FAILURE

THE PROGRAM TESTS FOR TWO TYPES OF POWER FAILURE, TOTAL SYSTEM POWER LOSS AND RX11 POWER LOSS RESULTING IN A RECALIBRATION OF THE DRIVES.

THE TOTAL SYSTEM POWER FAILURE IS DETECTED BY THE "SYSMAC" SUBROUTINE .SPOWER. WHEN THE POWER IS DETECTED TO BE GOING DOWN, THE REGISTERS ARE SAVED. WHEN THE POWER COMES BACK UP THE REGISTERS ARE RESTORED AND THE MESSAGE "POWER" IS PRINTED. THE PROGRAM THEN AUTOMATICALLY DOES A RESTART.

LOSS OF POWER IN THE RX11 CAUSES A RECALIBRATION OF ALL DRIVES. WHEN THIS HAPPENS, THE "INIT DONE" BIT IS SET IN THE THE RXES REGISTER ALONG WITH THE NORMAL "DONE" FLAG. AT EACH INTERRUPT THE PROGRAM TESTS FOR INIT DONE. IF IT IS FOUND TRUE, THE FUNCTION WAS NOT COMPLETED AND A POWER LOSS MUST HAVE BEEN DETECTED. WHEN THIS HAPPENS THE PROGRAM TYPES OUT "RX11 POWER" AND DOES AN AUTOMATIC RESTART.

IF THERE ARE REPEATED, NOTHING ELSE HAPPENS BUT, RX11 POWER MESSAGES THE INIT DONE FLAG MIGHT BE STUCK ON.

### 3.5 PROGRAM HUNG

THERE ARE MANY PLACES WHERE THE PROGRAM MUST WAIT FOR AN OPERATION TO BE COMPLETED IN THE RX01. THESE ARE WAITING FOR THE "DONE" FLAG TO INDICATE A FUNCTION IS COMPLETED, OR WAITING FOR A TRANSFER REQUEST "TR" FLAG TO SEND OR RECEIVE THE NEXT BYTE OF INFORMATION. IF THE RX11 DOES NOT COME BACK WITH EITHER OR BOTH OF THESE FLAGS THE PROGRAM WOULD HANG UP.

TO INHIBIT THIS FROM HAPPENING THERE ARE TWO SUBROUTINES USED TO CHECK FOR THESE TWO FLAGS.

THE "DONECK" LOOKS FOR THE DONE FLAG AND IF IT IS NOT SEEN WITHIN A SPECIFIC TIME THE "TEST HUNG" MESSAGE IS PRINTED AND THE PROGRAM HALTS. IN REGISTER 3 (177703) IS THE RETURN ADDRESS OF THE TEST THAT IS WAITING FOR THE DONE FLAG.



"TRCK" LOOKS FOR THE "TR" FLAG. IF IT IS NOT SEEN WITHIN A SPECIFIC TIME IT ALSO PRINTS THE "TEST HUNG" MESSAGE AND HALTS. IN THE SP (177706) IS THE RETURN ADDRESS OF THE TEST WAITING FOR THE TR FLAG.

THE WAITING TIME FOR THE TWO FLAGS DEPENDS UPON THE HOST PROCESSOR AS INDICATED BELOW:

"DONE" WAIT	11/45 BIPOL 14 SEC. 11/05 CORE 62 SEC.
"TR" WAIT	11/45 BIPOL .44 SEC. 11/05 CORE 2 SEC.

4,0 ERROR REPORTING

ALL ERRORS DETECTED WILL BE REPORTED IF SW13 = 0 FOR OPERATING CONDITIONS. IF SW12 = 1 THEN ONLY 10 DATA ERRORS FOR 1 SECTOR WILL BE REPORTED, AND A TOTAL OF DATA ERRORS FOR THAT SECTOR WILL BE REPORTED AT THE END OF THE SECTOR.

THE END OF PASS INDICATOR TYPED, ALSO INDICATES WEITHER ANY ERRORS OCCURED DJRRING THAT PASS. IF THERE WERE NO ERRORS THEN A "\*" IS PRINTED. IF THERE WERE ERRORS THEN A "-" IS PRINTED. THE TOTAL ACCUMULATED ERRORS AND SYSTEMS OPERATION IS REPORTED BY THE "ERROR DUMP" PROGRAM.

5,0 HALTS

THERE ARE VARIOUS HALT LOCATIONS THROUGHOUT THE PROGRAM. SOME ARE THE RESULTS OF "HARD" ERRORS OTHERS ARE WAITING FOR INTERVENTION. THEY ARE LISTED BELOW:

HALT #	TYPE	DEFINITION
HLT1:	NO ERRUK	;CAUTION - LOAD MEDIUM ON UNIT 0
HLT3:	ERRUR	;ERRORS FOUND ON RECAL FUNCTIONS
HLT4:	ERRUR	;HARD PARITY ERRORS
HLT5:	ERRUR	;NO DRIVES READY
HLT6:	ERRUR	;SW15 SET ON PARITY ERROR
HLT7:	ERRUR	;SW15 SET ON SEEK ERRUR
HLT10:	ERRUR	;SW15 SET ON CRC GENERATOR ERROR (NO DATES ERROR)
HLT11:	ERRUR	;HARD CRC GENERATOR ERRUR (NO DATA ERRORS)
HLT12:	ERRUR	;SW15 SET ON DATA CRC ERRUR
HLT13:	ERRUR	;SW15 SET ON MISSING DETECTED DATA ERROR
HLT14:	ERRUR	;SW15 SET ON DATA NO CRC ERRUR
HLT15:	ERRUR	;SW15 SET ON MISSING INTERRUPT AT DONE
HLT16:	NO ERRUR	;HALT AT END OF PASS
HLT17:	NO ERRUR	;HALT AT END OF ERRUR DUMP
HLT20:	ERRUR	;PROGRAM HUNG WAITING FOR DONE
HLT21:	ERRUR	;PROGRAM HUNG WAITING FOR TR FLAG

MAINDEC=11=DZRXA=E  
DZR XAE,P11

MACY11 27(732) 20-SEP-76 15:46 PAGE 2

- 1
- 2
- 3
- 4
- 5

.NLIST CND,MD,MC  
.LIST ME  
.ENABL ABS,AMA  
.MCALL .HEADER,.EQUAT, .SETUP, .\$TYPE, .\$STYPOCT, .\$READ

6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56

.MCALL ,SPWER,STARS,,SDB2D,,SSB2U,,SSAVE,,STRAP,GETSWR

```
.TITLE MAINDEC-11-0ZRXA-E
; *COPYRIGHT (C) MARCH 21, 1976
; *DIGITAL EQUIPMENT CORP.
; *MAYNARD, MASS. 01754
; *
; *PROGRAM BY DAVID L ADAMS
; *
; *THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
; *PACKAGE (MAINDEC-11-UZQAC-C0), MAR 21, 1976.
; *
STN=1
$SWR=160000    ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYP0UT
```

```
;COPYRIGHT (C) 1975, 1976
;THIS SOFTWARE IS FURNISHED UNDER A LICENSE FOR USE ONLY
;ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH
;THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS
;SOFTWARE, ON ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED
;OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
;EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO
;THESE LICENSE TERMS. TITLE TO OWNERSHIP OF THE
;SOFTWARE SHALL AT ALL TIMES REMAIN IN DEC.
;
;THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE
;WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT
;BY DIGITAL EQUIPMENT CORPORATION.
;
;DEC ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY
;OF ITS SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY
;DEC.
```

.SHTTL BASIC DEFINITIONS

```
; *INITIAL ADDRESS OF THE STACK POINTER *** 1200 ***
STACK= 1200
.EQUIV EMT, ERMOR    ;;BASIC DEFINITION OF ERROR CALL
.EQUIV IUT, SCUPE    ;;BASIC DEFINITION OF SCOPE CALL

; *MISCELLANEOUS DEFINITIONS
HT= 11    ;;CODE FOR HORIZONTAL TAB
LF= 12    ;;CODE FOR LINE FEED
CR= 15    ;;CODE FOR CARRIAGE RETURN
CHLF= 200  ;;CODE FOR CARRIAGE RETURN-LINE FEED
PS= 17776  ;;PROCESSOR STATUS WORD
.EQUIV PS, PSW
```

```
57 177774 STKLMT= 177774 ;;STACK LIMIT REGISTER
58 177772 PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
59 177570 DSWR= 177570 ;;HARDWARE SWITCH REGISTER
60 177570 DUISP= 177570 ;;HARDWARE DISPLAY REGISTER

; *GENERAL PURPOSE REGISTER DEFINITIONS
63 000000 R0= X0 ;;GENERAL REGISTER
64 000010 R1= X1 ;;GENERAL REGISTER
65 000020 R2= X2 ;;GENERAL REGISTER
66 000030 R3= X3 ;;GENERAL REGISTER
67 000040 R4= X4 ;;GENERAL REGISTER
68 000050 R5= X5 ;;GENERAL REGISTER
69 000060 R6= X6 ;;GENERAL REGISTER
70 000070 R7= X7 ;;GENERAL REGISTER
71 .EQUIV R6, SP ;;STACK POINTER
72 .EQUIV R7, PC ;;PROGRAM COUNTER

; *PRIORITY LEVEL DEFINITIONS
75 000000 PR0= 0 ;;PRIORITY LEVEL 0
76 000040 PR1= 40 ;;PRIORITY LEVEL 1
77 000100 PR2= 100 ;;PRIORITY LEVEL 2
78 000140 PR3= 140 ;;PRIORITY LEVEL 3
79 000200 PR4= 200 ;;PRIORITY LEVEL 4
80 000240 PR5= 240 ;;PRIORITY LEVEL 5
81 000300 PR6= 300 ;;PRIORITY LEVEL 6
82 000340 PR7= 340 ;;PRIORITY LEVEL 7

; *"SWITCH REGISTER" SWITCH DEFINITIONS
85 100000 SW15= 100000
86 040000 SW14= 40000
87 200000 SW13= 20000
88 010000 SW12= 10000
89 040000 SW11= 4000
90 002000 SW10= 2000
91 001000 SW09= 1000
92 000400 SW08= 400
93 000200 SW07= 200
94 000100 SW06= 100
95 000040 SW05= 40
96 000020 SW04= 20
97 000010 SW03= 10
98 000004 SW02= 4
99 000002 SW01= 2
100 000001 SW00= 1
101 .EQUIV SW09, SW9
102 .EQUIV SW08, SW8
103 .EQUIV SW07, SW7
104 .EQUIV SW06, SW6
105 .EQUIV SW05, SW5
106 .EQUIV SW04, SW4
107 .EQUIV SW03, SW3
108 .EQUIV SW02, SW2
109 .EQUIV SW01, SW1
110 .EQUIV SW00, SW0

; *DATA BIT DEFINITIONS (BIT00 TO BIT15)
```

```

113      100000      BIT15= 100000
114      040000      BIT14= 400000
115      020000      BIT13= 200000
116      010000      BIT12= 100000
117      004000      BIT11= 4000
118      002000      BIT10= 2000
119      001000      BIT09= 1000
120      000400      BIT08= 400
121      000200      BIT07= 200
122      000100      BIT06= 100
123      000040      BIT05= 40
124      000020      BIT04= 20
125      000010      BIT03= 10
126      000004      BIT02= 4
127      000002      BIT01= 2
128      000001      BIT00= 1
129      .EQUIV BIT09,BIT9
130      .EQUIV BIT08,BIT8
131      .EQUIV BIT07,BIT7
132      .EQUIV BIT06,BIT6
133      .EQUIV BIT05,BIT5
134      .EQUIV BIT04,BIT4
135      .EQUIV BIT03,BIT3
136      .EQUIV BIT02,BIT2
137      .EQUIV BIT01,BIT1
138      .EQUIV BIT00,BIT0

140      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
141      ERVVEC= 4      ;TIME OUT AND OTHER ERRORS
142      RESVEC= 10     ;RESERVED AND ILLEGAL INSTRUCTIONS
143      TBIVVEC=14     ;"T" BIT
144      TRTVEC= 14     ;TRACE TRAP
145      BPTVEC= 14     ;BREAKPOINT TRAP (BPT)
146      IOTVEC= 20     ;INPUT/OUTPUT TRAP (IOT) **SCOPE**
147      PWRVEC= 24     ;POWER FAIL
148      EMTVEC= 30     ;EMULATOR TRAP (EMT) **ERROR**
149      TRAPVEC=34     ;"TRAP" TRAP
150      TKVVEC= 60     ;TTY KEYBOARD VECTOR
151      TPVVEC= 64     ;TTY PRINTER VECTOR
152      PIQVEC=240     ;PROGRAM INTERRUPT REQUEST VECTOR

153
154      ;SPECIAL EQUATES
155
156      RD0STAT =13
157      RD1STAT =33
158      RUER = 17
159      DONEBIT =40
160      FBIE =101
161      EBIE =103
162      WRTIE =105
163      RDIE =107
164      WTD0IE =115
165      RECAL =40001
166      PR4 =200
167
168

```

```

169      000340      PR7 =340
170      000000      OPEN =0
171
172      000000      .#0      .WORD 0,0
173      000000 000000 000000
174
175      000024      .#24      SPWHDN
176      013764      340
177      000026 000340
178
179      000034      .#34      STRAP      ;ADDRESS OF TRAP SERVICE
180      013716      340
181      000036 000340
182
183      000046      .#46      LOGICAL      ;ACT11 EOP HOOK
184      011130
185
186      000052      .#52      .WORD 0
187      000052 000000
188
189      000174      .#174     DISPREG: 0
190      000174 000000      SWREG: 0
191      000176 000000
192
193      ;*****
194
195      ;STARTING ADDRESSES
196      ;
197      ;INITIAL START =200 /CLEARS ALL ERROR LOGS,RESETS COUNTERS,AND ALLOWS FOR
198      ; /SELECTION OF DRIVES AND TEST CONDITIONS
199      ;
200      ;RESTART =202 /USES PREVIOUS INITIAL START DRIVES AND TEST
201      ; /SELECTION AND CONTINUES TO ACCUMULATE STATISTICAL DATA
202      ;
203      ;ERROR REPORT =204 /PRINTS OUT ALL RUN AND ERROR CONDITIONS
204      ; /ACUMULATED OVER THE RUN OF THE PROGRAM.
205      ;
206      ;
207
208
209      .#200
210      000200 000402      HR 13
211      000202 000403      HR 25
212      000204 000404      HR 35
213      000206 000137 001220 15$ JMP SA200 ;OPERATOR SELECTED CONDITIONS
214      000212 000137 002700 25$ JMP RESTART ;RESTART PROGRAM WITH PVIOUS CONDITIONS
215      000216 000137 021062 35$ JMP ERDUMP ;STATISTICAL ERROR PRINT OUT
216
217
218      ;*****
219
220      ;THE FOLLOWING LOCATIONS "OD","ID","FIRST",AND "LAST"
221      ;MAY BE CHANGED BY THE OPERATOR MANUALLY HOWEVER FOLLOWING THESE RESTRICTIONS.
222      ;(IF THESE LOCATIONS ARE LEFT CLEARED,MAX AND MIN VALUES ARE ASSUMED)
223      ;
224      ; 1. DEFINITIONS:

```

```

225 ; OD=ADDRESS OF TRACK AT OUTER DIAMETER (MIN VALUE=0)
226 ;
227 ; ID=ADDRESS OF TRACK AT INNER DIAMETER (MAX VALUE=114 (OCTAL))
228 ; FIRST=ADDRESS OF FIRST SECTOR OF TRACK (MIN VALUE=1)
229 ; LAST=ADDRESS OF LAST SECTOR OF TRACK (MAX VALUE=32 (OCTAL))
230 ;
231 ; 2. LOCATIONS:
232 ; TRACKS LOC, 1200 BITS 14-----8 6-----0
233 ; ID OD
234 ;
235 ; SECTORS LOC, 1202 BITS 12-----8 4-----0
236 ; LAST FIRST
237 ;
238 ; 3. RESTRICTIONS:
239 ; THE CONTENTS OF "OD" MUST BE <= THE CONTENTS OF "ID"
240 ; THE CONTENTS OF FIRST MUST BE <= THE CONTENTS OF "LAST"
241 ;
242 ;
243 ;
244 001200 001200 OD: =1200 0
245 000000 IU=OD+1 0
246 001202 000000 FIRST: 0
247 001203 LAST=FIRST+1 0
248 ;
249 ;
250 ;/*****
251 ;
252 ;
253 ;THE NEXT WORD IS THE LOCATION OF THE "INTERRUPT VECTOR" ADDRESS
254 ;IF THE HARDWARE IS JUMPED FOR OTHER THAN STANDARD (264) VECTOR
255 ;ADDRESS, THEN LOAD INTO THIS LOCATION THE NEW VECTOR ADDRESS.
256 ;
257 ;IF THIS ADDRESS IS INCORRECT (DOES NOT MATCH THE HARDWARE) THE
258 ;PROGRAM WILL HALT AT THE VECTOR ADDRESS THE HARDWARE IS JUMPED
259 ;TO, AS ALL OTHER VECTOR ADDRESSES WILL HAVE HALTS IN THEM.
260 ;
261 ;*****NOTE*****
262 ;
263 ;THE INTERRUPT VECTOR ADDRESS CAN NOT BE SET FOR ADDRESSES 200
264 ;THRU 243, AS THESE ADDRESS ARE USED BY THE PROGRAM
265 ;
266 ;*****
267 ;
268 001204 000264 INTVEC: 264
269 ;
270 ;/*****
271 ;THE FOLLOWING TWO WORDS CONTAIN THE DEVILE CODES FOR
272 ;THE RX11 INTERFACE REGISTERS
273 ; RXCS = COMMAND STATUS REGISTER (USED AT VARIOUS TIMES FOR:)
274 ; RXDB = DATA BUFFER REGISTER (USED AT VARIOUS TIMES FOR:)
275 ; (RXTA = TRACK ADDRESS)
276 ; (RXSA = SECTOR ADDRESS)
277 ; (RXES = ERROR STATUS)
278 ;IF THE RX11 SYSTEM UNDER TEST IS JUMPED FOR REGISTER
279 ;ADDRESSES OTHER THAN STANDARD: RXCS = 177170
280 ; RXDB = 177172
  
```

```

281 ;PLACE IN THESE TWO WORDS THE CORRESPONDING NEW ADDRESSES FOR
282 ;THE REGISTERS. THE PROGRAM WILL TYPE OUT THE REGISTER
283 ;ADDRESSES FOR VERIFICATION BY THE OPERATOR. IF THE ADDRESSES
284 ;ARE INCORRECT (I.E. = DON'T MATCH THE HARDWARE) AND THE PROGRAM
285 ;IS RUN WITH THE INCORRECT ADDRESSES, A "MUNG" CONDITION WILL BE
286 ;PREPONTED AND THE PROGRAM WILL "HALT" AS THE RX11 WILL NOT BE
287 ;ABLE TO RESPOND TO COMMANDS.
288 ;
289 ;/*****
290 001206 177170 RXCS: 177170
291 001210 177172 RXDB: 177172
292 ;
293 ;BIT ASSIGNMENT IN THE RXCS REGISTER.
294 ;
295 ;KEY: R = READ ONLY BIT
296 ; W = WRITE ONLY BIT
297 ;
298 ; 15 = R = ERROR FLAG
299 ; 14 = W = INITIALIZE (RECALIBRATE)
300 ; 13 =
301 ; 10 = NOT USED
302 ; 8 =
303 ; 7 = R = TRANSFER REQUEST (TR) FLAG
304 ; 6 = R/W = INTERRUPT ENABLE
305 ; 5 = R = DONE FLAG
306 ; 4 = W = UNIT SELECT
307 ; 3 = W = FUNCTION
308 ; 2 = W = FUNCTION
309 ; 1 = W = FUNCTION
310 ; 0 = W = GO
311 ;
312 ;FUNCTION CODES:
313 ;
314 ; 0 + GO = FILL BUFFER
315 ; 2 + GO = EMPTY BUFFER
316 ; 4 + GO = WRITE SECTOR
317 ; 6 + GO = READ SECTOR
318 ; 10 (NOT USED)
319 ; 12 + GO = READ STATUS "A"
320 ; 14 + GO = WRITE DELETED DATA
321 ; 16 + GO = READ STATUS "B" (CODES)
322 ;
323 ;
324 ;THE FOLLOWING BIT ASSIGNMENTS REPRESENTES THE STATUS AT THE END OF A
325 ;FUNCTION (EXCEPT FUNCTION "READ STATUS B") DISPLAYED IN THE
326 ;RX DATA BUFFER (RXDB).
327 ;
328 ;
329 ; 15 =
330 ; 10 = NOT USED
331 ; 8 =
332 ; 7 = SELECTED DRIVE READY *
333 ; 6 = DELETED DATA
334 ; 5 =
335 ; 4 =
336 ; 3 = WRITE PROTECT ERROR (WHEN AVAILABLE)
337 ; 2 = INITIALIZE DONE **
  
```

```

337          /      1 = PARITY ERROR
338          /      0 = CRC ERROR
339          /
340          / *  VISIBLE ONLY IF THE FUNCTION WAS A #12 "READ STATUS A"
341          /
342          / **  VISIBLE ONLY AFTER AN INITIALIZE [KEY] OR [PROGRAMMED] WAS ISSUED.
343
344 001212 000000      DTESTP1      .WORD 0      ;TEST SELECTION WORD
345 001214 177570      SWR:          .WORD USWR      ;ADDRESS OF SWITCH REGISTER
346 001216 177570      DISPLAY:      .WORD DDISP     ; ADDRESS OF DISPLAY REGISTER
347
348          / *****
349
350          ;START OF OPERATOR SELECTABLE TEST,DATA PATTERNS,AND DRIVE UNIT CONDITIONS
351          ;SET THE TEST CONDITIONS WANTED (OR LEAVE 0) IN LOCATION CALLED
352          ;"DTESTP" (LOC. 1212), BEFORE STARTING THE PROGRAM, WHEN THE PROGRAM
353          ;STARTS IT WILL TYPE OUT THE TEST CONDITIONS IT IS OPERATING UNDER.
354          /
355
356          ;SWITCH REGISTER BITS
357          /
358          / 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
359          / U1 U0      NOT USED      D P P P T T T S S S
360          /
361          / U1  SELECT DRIVE UNIT 1
362          / U0  SELECT DRIVE UNIT 0
363          / IF NEITHER DRIVE IS SPECIFIED,PROGRAM WILL SELECT ALL DRIVES
364          / THAT ARE READY
365          /
366          / D =   DELETED DATA FUNCTIONS
367          /
368          / IF THIS SWITCH IS ON ALL READ AND WRITE FUNCTIONS WILL BE IN THE
369          / DELETED DATA MODE
370          /
371          /
372          / P =   DATA PATTERN SELECTION
373          /      0   DO PATTERN 7 (RANDOM)
374          /      1   ZEROS
375          /      2   ONES
376          /      3   FLOATING ZERO
377          /      4   FLOATING ONE
378          /      5   125
379          /      6   314
380          /      7   RANDOM
381          /
382          / T =   FUNCTIONAL TESTS
383          /      0   DO TEST 7 (WRITE/READ/HEAD CHECK)
384          /      1   WRITE ONLY
385          /      2   WRITE/READ
386          /      3   WRITE/READ CHECK
387          /      4   READ CHECK ONLY
388          /      5   READ ONLY
389          /      6   WRITE/READ CHECK ON ALTERNATE DRIVES
390          /      7   WRITE/READ/READ CHECK *
391
392          / * NOTE: THE FIRST HALF OF TEST 7 FORCES THE TRACK SEQUENCE
  
```

```

393          ;TO INCREMENT UP THROUGH ALL THE TRACKS DOING WRITE / READ CHECK FUNCTIONS.
394          ;THIS VERIFIES THAT ALL THE TRACKS ARE ACCESSABLE.
395          ;THE SECOND HALF OF THE PASS WILL USE THE SEQUENCE SELECTED BY THE
396          ;OPERATOR AS INDICATED BELOW,AND ONLY READ CHECK THE DATA JUST WRITTEN.
397          ;THIS VERIFIES THAT THE DATA CAN BE READ FROM A TRACK,AFTER THE HEAD HAS BEEN
398          ;MOVED AWAY FROM AND BACK TO THAT TRACK. AT THE COMPLETION OF THE PASS
399          ;THE DELETED DATA HIT IN TEST CONDITIONS IS COMPLEMENTED AND THE NEXT
400          ;PASS WILL BE RUN UNDER THIS NEW CONDITION.
401          /
402          / S =   TRACK SEQUENCING
403          /      0   DO SEQUENCE 7 (RANDOM)
404          /      1   INCREMENT
405          /      2   DECREMENT
406          /      3   INCREMENT/DECREMENT
407          /      4   BOUNCE
408          /      5   DECREASING BOUNCE
409          /      6   STROBE
410          /      7   RANDOM
411          /
412          / *****
413
414          ;SET THE OPERATING CONDITIONS IN THE SWITCH REGISTER (HARDWARE).
415          ;FOR THE SOFTWARE SWITCH REGISTER (LOC. 176) BEFORE STARTING THE PROGRAM.
416          /
417          ;SWITCHES DO THE FOLLOWING WHEN SET TO "1".
418          /
419          /
420          / SW 15  #HALT ON ERROR
421          / SW 14  #HALT AT END OF TEST
422          / SW 13  #DON'T PRINT ERROR MESSAGE
423          / SW 12  #TYPE ONLY 10 DATA ERRORS
424          / SW 11  #NO RETRY ON ERROR, LOG HARD ERROR
425          /
426          / SW 8   #NU RECALIBRATION ON SEEK ERRORS
427          /
428          / SW15 - SW0  #SELECT SOFTWARE SWITCH REGISTER
429          /
430          / *****
431
432          ;
433          ;
434          ;
435 001220 000005      SA200:      RESET          ;INITIALIZE THE RX01
436 001222 012737 177570 001214      MOV #177570,SWR      ;RESET TO HARDWARE SWR.
437 001230 012706 001200      MOV #STACK,SP
438 001234 012746 000340      MOV #PH7,-(SP)
439 001240 012746 001246      MOV #2S,-(SP)
440 001244 000002      RTI
441 001246 104400 016554      251      TYPE ,MREV      ;PRINT NAME AND REVISION
442 001252 012737 001272 000004      MOV #33,4      ;SET TIME OUT VECTOR
443 001260 022777 177777 177726      CMP #177777,#SWR  ;IS SOFTWARE SWR SELECTED
444 001266 001402      BEQ 4S      ;YES, INSERT IT'S ADDRESS
445 001270 000423      BR 5S      ;BRANCH IF NO TIMEOUT TRAP OCCURS
446 001272 022626      CMP (SP)+,(SP)+      ;RESTORE STACK AFTER TRAP
447 001274 012737 000176 001214      MOV #SWREG,SWR   ;POINT TO SOFTWARE SWITCH REG.
448 001302 012737 000174 001216      MOV #DISPREG,DISPLAY ;POINT TO SOFTWARE DISP. REG.
  
```

```

449 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
450 001510 005737 000042 TST #42 //ARE WE RUNNING UNDER XXOP/ACT?
451 001314 001006 BNE 64$ //BRANCH IF YES
452 001316 025727 001214 000176 CMP SWR,#SWREG //SOFTWARE SWITCH REG SELECTED?
453 001324 001005 BNE 65$ //BRANCH IF NO
454 001326 104404 GTSWR //GET SOFT-SWR SETTINGS
455 001330 000403 BR 65$
456 001332 112737 000001 013714 64$: MOV# #1,$AUTOB //SET AUTO-MODE INDICATOR
457 001340 65$:
458 001340 012737 000006 000004 55: MOV #6,4 //RESTORE ERROR VECTOR
459 001346 013703 001204 MOV INTVEC,R5 //SET INTERRUPT VECTOR LOCATIONS
460 001352 012723 007250 MOV #INTSERV,(R3)+
461 001356 012713 000344 MOV #340,(R3)
462 001362 012737 001234 MOV #001234,RAN1 //INITIALIZE CONSTANTS IN RANDOM NUMBER GENERATOR
463 001370 012737 000765 012210 MOV #000765,RAN2
464 001376 012705 017024 MOV #PARLOG,R5
465 001402 005025 1$: CLR (R5)+ //SET UP TO CLEAR ALL COUNTERS
466 001404 022705 021044 CMP #PASCNTR+2,R5 //OF ERRORS,ACCESSES,AND PASSES
467 001410 001374 BNE 1$ //HAVE ALL COUNTERS BEEN CLEARED
468 001412 012705 017004 START1: MOV #RURETRY,R5 //SET ALL RETRY COUNTERS TO 10
469 001416 012725 000012 5$: MOV #10,(R5)+
470 001422 022705 017024 CMP #SRETRY+2,R5
471 001426 001373 BNE 5$
472 001430 104400 015167 TYPE ,MRXCS //TYPE OUT DEVICE REGISTER ADDRESSES
473 001434 013746 001206 MOV RXCS,-(SP) //SAVE RXCS FOR TYPEOUT
474 001440 104402 TYPDS //GO TYPE--OCTAL ASCII
475 001442 006 .BYTE 6 //TYPE 6 DIGITS
476 001443 000 .BYTE 0 //SUPPRESS LEADING ZEROS
477 001444 104400 015177 TYPE ,MRX0B
478 001450 013746 001210 MOV RX0B,-(SP) //SAVE RX0B FOR TYPEOUT
479 001454 104402 TYPDS //GO TYPE--OCTAL ASCII
480 001456 006 .BYTE 6 //TYPE 6 DIGITS
481 001457 000 .BYTE 0 //SUPPRESS LEADING ZEROS
482 001460 104400 015150 TYPE ,MINTVEC //AND VECTOR ADDRESS
483 001464 013746 001204 MOV INTVEC,-(SP) //SAVE INTVEC FOR TYPEOUT
484 001470 104402 TYPDS //GO TYPE--OCTAL ASCII
485 001472 003 .BYTE 3 //TYPE 3 DIGIT(S)
486 001473 000 .BYTE 0 //SUPPRESS LEADING ZEROS
487 001474 004737 007122 JSR PC,DONECK //WAIT FOR DONE BIT AFTER RECAL
488 001500 032777 000004 177502 HIT #BIT2,#RX0B //IS INIT DONE SET
489 001506 001002 BNE 1$ //YES SKIP NEXT INSTRUCTION
490 001510 104400 015535 TYPE ,MINIT1 //NO,PRINT NG INIT DONE ERROR
491 001514 005777 177466 1$: TST #RXCS //WAS THERE AN ERROR ON START RECAL
492 001520 100002 BPL TESTOR //IF NOT CONTINUE
493 001522 004737 002554 JSR PC,HOML //YES,DO RECAL AGAIN
494 001526 012737 000012 017014 TESTOR: MOV #10,#PRETRY //SET THE PARITY RETRY COUNTER
495 001534 005037 010704 CLR UNITSEL //CLEAR UNIT SELECTION WORD
496 001540 105777 177444 TSTB #RX0B //IS DRY BIT SET FOR UNIT 0 AFTER RECAL
497 001544 100003 BPL 2$ //IF NOT SKIP NEXT INSTRUCTION
498 001546 052737 000200 010704 BIS #BIT7,UNITSEL //YES,SET UNIT 0 READY BIT
499 001554 032737 140000 001212 2$: HIT #140000,UTESTP //WERE ANY DRIVES SPECIFIED
500 001562 001445 BEQ NOSEL //NO,SEE IF UNIT 1 IS READY
501 001564 005737 001212 TST UTESTP //WAS UNIT 1 SELECTED
502 001570 100044 BPL OPCOND //NO,IT MUST BE UNIT 0
503 001572 004737 001614 JSR PC,DRY1 //SEE IF UNIT 1 IS READY
504 001576 032737 000000 001212 HIT #40000,DTESTP //WAS UNIT 0 SELECTED

```

```

505 001604 001036 BNE OPCOND //YES,CONTINUE
506 001606 105037 010704 CLRB UNITSEL //NO,CLEAR UNIT 0 READY BIT
507 001612 000433 BR OPCOND
508
509 001614 012777 000033 177364 DRY1: MOV #RD1STAT,#RXCS //HEAD STATUS OF UNIT 1
510 001622 004737 007122 JSR PC,DONECK //WAIT FOR DONE BIT
511 001626 132777 000002 177354 HIT# #2,#RX0B //IS PARITY ERROR BIT SET
512 001634 001411 BEQ 2$ //NO,CONTINUE
513 001636 004737 002046 JSR PC,STATEN //YES,GO HANDLE PARITY ERROR
514 001642 000764 HR DRY1 //REISSUE FUNCTION
515 001644 032777 000004 177336 HIT #BIT2,#RX0B //INIT DONE SHOULD NOT BE SET
516 001652 001402 BEQ 2$
517 001654 104400 015567 TYPE ,MINIT2 //INIT DONE WAS SET PRINT ERROR
518 001660 105777 177324 2$: TSTB #RX0B //IS UNIT 1 DRY BIT SET
519 001664 100003 BPL 3$ //IF NOT SKIP NEXT INSTRUCTION
520 001666 052737 100000 010704 BIS #BIT15,UNITSEL //YES,SET UNIT 1 READY BIT
521 001674 000207 3$: RTS PC
522
523 001676 004737 001614 NOSEL: JSR PC,DRY1 //TEST UNIT 1 FOR DRY SET
524 001702 105737 010704 OPCOND: TSTB UNITSEL //IS UNIT 0 SELECTED
525 001706 100016 BPL 1$ //NO
526
527 //TEST ACT11 LOAD MEDIA INDICATOR
528
529 001710 123727 000041 000010 CMPB #41,#10 //DOES LOCATION 41 CONTAIN THE NUMBER 10?
530 001716 001012 BNE 1$ //BRANCH IF NOT
531 001720 005737 000042 TST #42 //CHECK FOR RXOP OPERATION
532 001724 001404 BEQ 2$
533 001726 042737 000200 010704 BIC #200,UNITSEL //IN CHAIN MODE, DESELECT UNIT 0
534 001734 000403 BR 1$ //AND DO NOT HALT
535 001736 104400 014643 2$: TYPE ,D0LOAD //INFORM USER TO REMOVE LOAD MEDIUM
536 //FROM UNIT 0 AND REPLACE WITH
537 //A 'SCRATCH' DISKETTE IF HE
538 //WISHES TO TEST UNIT 0
539 001744 000241 CLC //CLEAR C BIT FOR ROR'S THAT FOLLOW
540 001746 013737 001212 011370 1$: MOV UTESTP,SEQUEN //PUT INITIAL CONDITIONS IN SEQUENCE
541 001754 042737 177770 011370 BIC #17770,SEQUEN //CLEAR ALL BUT SEQUENCE BITS
542 001762 013737 002140 MOV UTESTP,TEST //PUT INITIAL CONDITIONS IN TEST
543 001770 042737 177707 002140 BIC #17770,TEST //CLEAR ALL BUT TEST BITS
544 001776 006037 002140 ROR TEST //RIGHT JUSTIFY TEST BITS
545 002002 006037 002140 ROR TEST
546 002006 006037 002140 ROR TEST
547 002012 013737 001212 010420 MOV UTESTP,PAT //PUT INITIAL CONDITIONS IN PATTERN
548 002020 006137 010420 ROL PAT //RIGHT JUSTIFY PATTERN BITS
549 002024 006137 010420 ROL PAT
550 002030 000337 010420 SWAB PAT
551 002034 042737 177770 010420 BIC #17770,PAT //CLEAR ALL BUT PATTERN BITS
552 002042 000137 002722 JMP XSTART
553
554 002046 005237 017024 STATER: INC PARLOG //INC PARITY ERROR COUNTER
555 002052 104400 015167 TYPE ,MRXCS //PRINT THE RXCS CONTENTS
556 002056 017746 177124 MOV #RXCS,-(SP) //SAVE #RXCS FOR TYPEOUT
557 002062 104402 TYPDS //GO TYPE--OCTAL ASCII
558 002064 003 .BYTE 3 //TYPE 3 DIGIT(S)
559 002065 000 .BYTE 0 //SUPPRESS LEADING ZEROS
560 002066 005337 017014 DEC P2METHY //HAVE 10 ERRORS OCCURED

```

561	002072	001405				BEQ 15		YES,LOG HARD ERROR
562	002074	104400	016017			TYPE ,MPAR		NO,TYPE PARITY ERROR
563	002100	104400	015052			TYPE ,MCRLF		
564	002104	000207				RTS PC		RETURN
565	002106	005237	017026	15:		INC HPARLOG		INC HARD PARITY ERROR LOG
566	002112	104400	015755			TYPE ,MUNREC		PRINT UNRECOVERABLE PARITY ERROR
567	002116	104400	016017			TYPE ,MPAR		
568	002122	104400	015052			TYPE ,MCRLF		
569	002126	104400	015702			TYPE ,MHALT4		
570	002132	000000		HLT4:		HALT		HALT BECAUSE OF HARD PARITY ERROR
571	002134	000137	001220			JMP SA200		IF CONTINUE IS PRESSED GO TO START
572	002140	000000		TEST:		0		
573								
574	002142	005002				CLR W2		MESSAGE FLAG
575	002144	005737	001200			TST 00		TEST FOR NO SELECTION OF TRACKS
576	002150	001441				BEQ 15		IT WILL BE STANDARD LIMITS
577	002152	123727	001201	000114		CMPB 10,#114		TEST FOR MAXIMUM ID LIMIT
578	002160	101112				BMI 145		
579	002162	001431				BEQ 25		
580	002164	123737	001200	001201	35:	CMPB 00,10		00 MUST BE EQUAL OR LESS THAN 10
581	002172	101105				BMI 145		
582	002174	004737	002432			JSR PC,TSTMSG		HAS INITIAL MESSAGE BEEN TYPED
583	002200	104400	015510			TYPE ,MOD		TYPE BOTH 10 AND 00 LIMITS
584	002204	113737	001200	002216		MOVW 00,75		
585	002212	004537	014446			JSR W5,SGLDEC		TYPE SINGLE DECIMAL WORD
586	002216	000000			75:	OPEN		
587	002220	104400	015047			TYPE ,UPLSP		
588	002224	104400	015514			TYPE ,MID		
589	002230	113737	001201	002242		MOVW 10,105		
590	002236	004537	014446			JSR W5,SGLDEC		
591	002242	000000			105:	OPEN		
592	002244	000403				HR 15		
593	002246	005737	001200	25:		TSTW 00		IS 00 = 0
594	002252	001344				HNE 55		NO,GO PRINT IT
595	002254	005737	001202	15:		TST F15T		TEST FOR NO SELECTION OF SECTORS
596	002260	001441				BEQ 155		IT WILL BE STANDARD LIMITS
597	002262	123727	001203	000032		CMPB LAST,#32		TEST FOR NORMAL LAST SECTOR LIMIT
598	002270	101053				BMI 155		
599	002272	001440				BEQ 45		
600	002274	105737	001202	55:		TSTB FIRST		THERE IS NO 0 SECTOR
601	002300	001447				BEQ 155		
602	002302	123737	001202	001203		CMPB FIRST,LAST		FIRST MUST BE EQUAL TO OR LESS
603	002310	101043				BMI 155		THAN LAST
604	002312	004737	002432			JSR PC,TSTMSG		
605	002316	104400	015047			TYPE ,UPLSP		
606	002322	104400	015520			TYPE ,MFKST		TYPE BOTH FIRST AND LAST LIMITS
607	002326	113737	001202	002340		MOVW FIRST,115		
608	002334	004537	014446			JSR W5,SGLDEC		
609	002340	000000			115:	OPEN		
610	002342	104400	015047			TYPE ,UPLSP		
611	002346	104400	015527			TYPE ,MLAST		
612	002352	113737	001203	002364		MOVW LAST,125		
613	002360	004537	014446			JSR W5,SGLDEC		
614	002364	000000			125:	OPEN		
615	002366	104400	015240			TYPE ,DBLLF		
616	002372	000404				BR 155		

617	002374	123727	001202	000001	45:	CMPB FIRST,#1		IS FIRST SET TO 1
618	002402	001334				HNE 55		NO,GO PRINT IT
619	002404	000207			135:	RTS PC		
620								
621	002406	104400	016363	145:		TYPE ,0D2BIG		PRINT TRACK LIMITS OUT OF RANGE
622	002412	005037	001200			CLR 00		
623	002416	000716				BR 15		
624	002420	104400	016455	155:		TYPE ,S2BIG		PRINT SECTOR LIMITS OUT OF RANGE
625	002424	005037	001202			CLR F15T		
626	002430	000765				BR 155		
627								
628	002432	005702			TSTMSG:	TST W2		TEST MESSAGE FLAG
629	002434	001005				HNE 15		
630	002436	104400	015446			TYPE ,MNONSTD		MESSAGE HEADING HAS NOT BEEN TYPED
631	002442	104400	015052			TYPE ,MCRLF		
632	002446	005202				INC W2		SET THE MESSAGE FLAG
633	002450	000207			15:	RTS PC		
634								
635	002452	104400	015052		ICOND:	TYPE ,MCRLF		
636	002456	104400	015417			TYPE ,MICON		TYPE INITIAL CONDITIONS
637	002462	105737	010704			TSTB UNITSEL		TEST FOR DRIVE SELECTION
638	002466	100002				HPL 15		
639	002470	104400	015106			TYPE ,MUNIT0		TYPE UNIT 0 SELECTED
640	002474	005737	010704	15:		TST UNITSEL		
641	002500	100002				HPL 25		
642	002502	104400	015116			TYPE ,MUNIT1		TYPE UNIT 1 SELECTED
643	002506	104400	015040	25:		TYPE ,TAB		
644	002512	004737	010012			JSR PC,TYPSEL		TYPE TEST SELECTIONS
645	002516	104400	015052			TYPE ,MCRLF		
646	002522	005737	010704			TST UNITSEL		WERE ANY DRIVES READY
647	002526	001005				BNE CONT12		YES,CONTINUE
648	002530	104400	015016			TYPE ,MNDRY		NO,TYPE NO DRIVES MESSAGE
649	002534	000000		HLT5:		HALT		NO DRIVES CAN'T CONTINUE
650	002536	000137	001220			JMP SA200		IF CONTINUE IS PRESSED RESTART
651	002542	004737	002142	CONT12:		JSR PC,TSTLIMITS		TYPE NONSTANDARD TRK AND SEC LIMITS
652	002546	000207				RTS PC		
653								
654	002550	104400		HOME:		CKSWR		
655	002552	032777	000400	176434		BIT #SW8,#SWR		TEST THE NO RECAL SWITCH
656	002560	001046				BNE RTN		RETURN IF THE SWITCH IS SET
657	002562	012737	000012	017014		MOV #10,,P2RETRY		SET UP THE PARITY RETRY COUNTER
658	002570	012737	000012	017010		MOV #10,,UDRETRY		USE THE UD RETRY COUNTER FOR RECAL RETRIES
659	002576	012777	040001	176402	25:	MOV #RECAL,#RXCS		ISSUE RECAL FUNCTION
660	002604	004737	007122			JSR PC,DONECK		WAIT FOR DONE FLAG
661	002610	032777	000002	176372		BIT #2,#RX08		WAS THERE A PARITY ERROR
662	002616	001403				BEQ 15		NO,CHECK FOR ANY OTHER ERROR
663	002620	004737	002046			JSR PC,STATER		YES,GO REPORT IT
664	002624	000764				BR 25		RETRY RECAL
665	002626	005777	176354	15:		TST #RXCS		IS THE ERROR FLAG SET
666	002632	100016				HPL XHOME		IF NOT RETURN
667	002634	004737	010200			JSR PC,RDCCODE		YES,PRINT STATUS REGISTERS
668	002640	123727	010072	000040		CMPB BSTAT,#40		IS THE B CODE LESS THAN CODE 40
669								(RECAL CODES ARE 10,20,AND 30)
670	002646	002010				BGE XHOME		IF NOT RETURN
671	002650	005337	017010			DEC 00RETRY		HAVE 10 ERRORS OCCURED
672	002654	001350				RNE 25		NO,RETRY

```
673 002656 104400 015713 TYPE ,MHALT3 ;YES,CAN'T CONTINUE AS RX11 WILL NOT RECAL
674 002662 000000 HALT ;
675 002664 000137 001220 HLT3: JMP SA200 ;IF CONT. SWITCH IS PRESSED GO TO START
676 002670 012737 000001 011362 XHOME: MOV #1,PRESTRK ;RESET THE PRESENT TRACK TO TRACK 1
677 002676 000207 RTN: RTS PC ;RETURN
678
679 ;*****
680 ;DECODE TEST BITS OF INITIAL CONDITIONS
681 ;
682 ;BITS 5,4,AND 3 OF THE INITIAL SWR SELECTED THE TEST WHICH
683 ;IS TO BE PERFORMED, THEY ARE AS FOLLOWS:
684 ;
685 ; BITS TESTS
686 ; 5 4 3
687 ; 0 0 0 NO TEST SELECTED (DEFAULT TO TEST 7)
688 ; 0 0 1 WRITE ONLY
689 ; 0 1 0 WRITE FOLLOWED BY READ
690 ; 0 1 1 WRITE FOLLOWED BY READ AND VERIFY
691 ; 1 0 0 READ AND VERIFY ONLY
692 ; 1 0 1 READ ONLY (CRC CHECK)
693 ; 1 1 0 READ AND VERIFY DELETED DATA
694 ; 1 1 1 WRITE FOLLOWED BY READ FOLLOWED BY ANOTHER
695 ; READ AND VERIFY
696
697 ;*****
698
699
700 002700 012706 001200 RESTART: MOV #STACK,SP ;RESET THE STACK POINTER
701 002704 012746 000340 MOV #PR7,=(SP) ;AND INTERRUPT LEVEL
702 002710 012746 002716 MOV #18,=(SP)
703 002714 000002 RTI
704 002716 005237 021440 IS: INC R6CNTNR ;INC RESTART COUNTER
705 002722 004737 002452 XSTART: JSR PC,ICOND ;TYPE OUT INITIAL CONDITIONS
706 002726 005037 011142 TESTSEL: CLR CHARCT ;CLEAR EOP CHARACTER COUNTER
707 002732 012737 000012 017004 MOV #10,,R0RETRY ;SET UP READ AND WRITE RETRY COUNTERS
708 002740 012737 000012 017006 MOV #10,,W0RETRY
709 002746 005037 000124 CLR ERWRT ;CLEAR WRITE CAUSED BY ERROR FLAG
710 002752 005037 000412 CLR R0AFTWT ;CLEAR READ AFTER WRITE FLAG
711 002756 142737 000377 003014 BICB #377,##R0RTEST ;CLEAR OUT BRANCH OFFSET
712 002764 005737 002140 TST TEST ;IF NO TEST SPECIFIED FORCE TEST 7
713 002770 001003 BNE 15
714 002772 012737 000007 002140 MOV #7,TEST
715 003000 013704 000140 15: MOV TEST,R4
716 003004 005304 DEC R4 ;ADJUST TEST BITS FOR CORRECT
717 003006 006304 ASL R4 ;BRANCH OFFSET
718 003010 150437 003014 BRANCH OFFSET
719 003014 000777 BISH R4,##R0RTEST ;INSERT OFFSET TO BRANCH INST.
720 003016 000137 003052 BR0RTEST: JMP WRONLY ;WRITE ONLY FUNCTION
721 003022 000137 000410 JMP WRTHD ;WRITE THEN READ FUNCTION
722 003026 000137 005456 JMP WTHDCK ;WRITE THEN READ VERIFY
723 003032 000137 006444 JMP W0DHWK ;READ VERIFY
724 003036 000137 006504 JMP W0UNLY ;READ ONLY FUNCTION
725 003042 000137 006544 JMP DRVSWP ;WRITE,AND READ VERIFY ON ALTERNATING DRIVES
726 003046 000137 006630 JMP TEST7 ;WRITE,READ,FOLLOWED BY READ VERIFY
727
728
```

```
729 ;*****
730 ;THIS IS A WRITE ONLY FUNCTION USING DATA PATTERN AND
731 ;TRACK SEQUENCE SPECIFIED BY INITIAL SWR SETTINGS
732 ; T = 1
733
734
735
736 003052 004737 010312 WRONLY: JSR PC,GETPATTERN ;SET UP SOFTWARE DATA BUFFER
737 003056 004737 011144 XWRONLY: JSR PC,INITTRACKS ;SET UP ID,OD,AND TRACK COUNTER
738 003062 004737 010614 JSR PC,GETUNIT ;SET UP DRIVE UNIT SELECTION
739 003066 004737 011254 15: JSR PC,GETTRACK ;PICK UP NEXT TRACK
740 003072 004737 003112 JSR PC,WRITE ;DO THE WRITE FUNCTION
741 003076 005337 011356 DEC TRCNTM ;TEST TRACK COUNTER
742 003102 001371 BNE 15
743 003104 004737 010732 JSR PC,STOP ;CHECK FOR LAST DRIVE AND EOP
744 003110 000762 BR XWRONLY ;NEXT PASS
745
746 003112 004737 012216 WRITE: JSR PC,INITSECTOR ;SET UP FIRST, LAST, AND SECTOR COUNTER
747 003116 004737 012316 XWRITE: JSR PC,GETSECTOR ;PICK UP NEXT SECTOR
748 003122 012737 000012 017022 MOV #10,,SRETRY
749 003130 012737 000012 017016 ONEWRT: MOV #10,,PRETRY ;SET RETRY COUNTER
750 003136 004737 005656 FILLBUF: JSR PC,ADJSUM ;ADJUST DATA BUFFER AND CHECK SUM FOR ADDRESSES
751 003142 012746 003372 MOV #FILLDONE,=(SP) ;PUT GOOD RETURN ON STACK
752 003146 012746 003214 MOV #FILLER,=(SP) ;PUT ERROR RETURN ON STACK
753 003152 005037 006434 CLR BYTECNTR
754 003156 012746 000200 MOV #PH4,=(SP)
755 003162 012746 003170 MOV #18,=(SP)
756 003166 000002 RTI
757 003170 012777 000101 176010 15: MOV #FNIE,0RXCS ;EXECUTE FILLBUFFER COMMAND
758 003176 004737 007014 FILLFLAG: JSR PC,TRCK ;TEST FOR TRANSFER REQUEST FLAG
759 003202 112077 176002 XFRBYTE: MOV0 (#0),0RXDB ;TRANSFER DATA BYTE
760 003206 005237 006434 INC BYTECNTR
761 003212 000771 BR FILLFLAG ;WAIT FOR NEXT TR FLAG
762
763 003214 005726 FILLER: TST (SP)+ ;REMOVE THE DONE RETURN FROM THE STACK
764 003216 012737 015366 003264 MOV #MFIL,PTYP1+2 ;PUT ADDR OF FILLBUF MESSAGE IN PAR ERR TYP0UT 1
765 003224 012737 015366 003354 MOV #MFIL,PTYP2+2 ;PUT ADDR OF FILLBUF MESSAGE IN PAR ERR TYP0UT 2
766 003232 012737 003136 003334 MOV #FILLBUF,PCONT+2 ;IF NOT HARD ERR RETURN THROUGH PCONT TO FILLBUF
767 003240 000137 003244 JMP PARTEST ;PRINT OUT PAR ERR AND TEST CONDITIONS FOR RETRY
768
769
770 003244 005237 017024 PARTEST: INC PARLOG ;INCREMENT PARITY ERROR
771 003250 104405 CKSWR
772 003252 032777 020000 175734 BIT #SW13,0SWR ;TEST DON'T PRINT ERROR SWITCH
773 003260 001006 BNE CONT4
774 003262 104400 000000 PTPY1: TYPE ,OPEN ;PRINT THE PARITY ERROR MESSAGE
775 003266 104400 016017 TYPE ,MPAR
776 003272 104400 015052 TYPE ,MCRLF
777 003276 104405 CKSWR
778 003300 005777 175710 TST 0SWR ;TEST HALT ON ERROR SWITCH
779 003304 100001 HPL CONT13
780 003306 000000 HLT6: HALT ;HALT ON ERROR
781 003310 032777 004000 175676 CONT13: BIT #SW11,0SWR ;TEST NO RETRY SWITCH
782 003316 001007 BNE CONT5 ;IF SET LOG HARD ERROR
783 003320 005337 017016 DEC PRETRY ;DECREMENT RETRY COUNTER
784 003324 005737 017016 TST PRETRY ;HAVE 10 ERRORS OCCURED
```



```

785 003330 001402
786 003332 000137 003136 PCONT: BEQ CONTS ;IF CLEARED LOG HARD ERROR
787 003336 005237 017026 CONTS: INC HPARLOG ;RETURN TO RETRY TEST THROUGH HERE
788 003342 104400 015240 TYPE ,DUBLF ;INC,HARD PARITY ERROR COUNTER
789 003346 104400 015755 TYPE ,MUNREC ;TYPE HARD PARITY ERROR
790 003352 104400 000000 PTP2: TYPE ,OPEN
791 003356 104400 016017 TYPE ,MPAR
792 003362 104400 015052 TYPE ,MCRLF
793 003366 000137 010074 JMP DELUNIT ;GO DELETE THE UNIT,CAN NOT CONTINUE
794
795 ;SWITCH 9 OF INITIAL TEST CONDITIONS SWITCH SETTINGS IS THE DELETED DATA
796 ;FUNCTION INDICATOR, WHEN THIS BIT IS SET ALL WRITE / READ FUNCTIONS WILL
797 ;SET AND CHECK THE DELETED DATA BIT ON THE DISKETTE.
798 ;)*****
799
800 003372 012737 000012 017016 FILLDONE: MOV #10,,PRETRY ;SET UP RETRY COUNTER
801 003400 012746 003462 REWRITE: MOV #WTDONE,-(SP) ;SET GOOD RETURN ON STACK
802 003404 012746 003510 MOV #WRTER,-(SP) ;SET ERROR RETURN ON STACK
803 003410 112737 000105 004120 MOVB #WRTIE,FUNCTION ;SET FUNCTION WORD TO WRITE
804 003416 032737 001000 001212 BIT #BIT9,UTESTP ;TEST FOR WRITE DELETED DATA
805 003424 001403 BEQ IS
806 003426 112737 000115 004120 MOVB #WTDODIE,FUNCTION
807 003434 002737 000001 021030 1S: ADD #1,WTCNTH ;INC TOTAL WRITE FUNCTIONS COUNTER
808 003442 005537 021032 ADC WTCNTH+2 ;DOUBLE PRECISION COUNTER
809 003446 004737 004022 JSR PC,COMMWORD ;TRANSFER COMMAND TO DRIVE
810 003452 004737 007122 JSR PC,DONECK ;TEST FOR DONE FLAG
811 003456 000137 007204 JMP NOINTER ;NO INTERRUPT ERROR
812
813 003462 005737 004124 WRTDONE: TST ERWRT ;IS THIS A REWRITE FROM A DATA ERROR
814 003466 001004 BNE IS ;YES GO REREAD THIS SECTOR
815 003470 005337 012306 DEC SECCNTR ;NO,TEST SECTOR COUNTER
816 003474 001003 RTS ES ;NOT LAST SECTOR GO TO NEXT ONE
817 003476 000207 1S: JMP PC
818 003500 000137 004246 2S: JMP WEREAD ;REREAD THE SECTOR
819 003504 000137 003110
820
821 003510 005726 WRTER: TST (SP)+ ;REMOVE THE DONE RETURN FROM THE STACK
822 003512 032737 000002 010070 BIT #BIT1,ASTAT ;IS THIS A PARITY ERROR
823 003520 001413 BEQ WRTSEK ;NO, IT MUST BE A SEEK ERROR
824
825 003522 012737 016010 003264 ;PARITY ERROR DURING A WRITE FUNCTION
826 003530 012737 016010 003354 MOV #WWRITE,PTYP1+2 ;PUT ADDR OF WRITE MESSAGE IN PAR ER TYP0UT 1
827 003536 012737 005400 003334 MOV #WWRITE,PTYP2+2 ;PUT ADDR OF WRITE MESSAGE IN PAR ER TYP0UT 2
828 003544 000137 003244 JMP PARTEST ;IF NOT HARD ERR RETURN THROUGH PCONT TO REWRITE
829
830 ;SEEK ERROR DURING A WRITE FUNCTION
831 003550 012737 003130 003674 WRTSEK: MOV #ONEWRT,SEKRTY+2 ;SETUP FOR WRT RETRY ON SEEK ERROR
832 ;(AFTER A RECAL. THE CONTENTS OF SECTOR 1,
833 ;TRACK 1 ARE LOADED INTO THE SECTOR BUFFER.
834 ;TO REWRITE THE CORRECT DATA THE PROGRAM
835 ;MUST REFILL THE SECTOR BUFFER.
836 003556 012737 016010 003630 MOV #WWRITE,STYP1+2 ;PUT ADDR OF WRITE MESSAGE IN SEEK ER TYP0UT 1
837 003564 012737 016010 003722 MOV #WWRITE,STYP2+2 ;PUT ADDR OF WRITE MESSAGE IN SEEK ER TYP0UT 2
838 003572 004737 003576 JSR PC,SEEKER ;RECORD SEEK ERROR
839
840 003576 012703 017036 SEEKER: MOV #ZSEKLOG,R3 ;SETUP INC INSTRUCTION FOR UNIT 0 LOG

```

```

841 003602 004737 021044 JSR PC,U1LOG ;TEST FOR UNIT 1 LOG
842 003606 005213 INC (R5) ;INCREMENT SEEK ERROR LOG
843 003610 004737 003770 JSR PC,TRKERR ;INC ERROR PER TRACK COUNTERS
844 003614 104405 CASKH
845 003616 032777 020000 175370 BIT #SW13,#SWR ;CHECK DON'T PRINT ERROR SWITCH
846 003624 001004 BNE SWHLT1
847 003626 104400 016010 STYP1: TYPE ,MWRITE ;PRINT WRITE (READ) SEEK ERROR
848 003632 004737 003736 JSR PC,SEKTP
849 003636 104405 CASKH
850 003640 005777 175350 TST #SWR ;TEST THE HALT ON ERROR SWITCH
851 003644 100001 HPL CONT14
852 003646 000000 HALT
853 003650 032777 004000 175336 HLT1: CONT14: ;HALT ON THE ERROR
854 003656 001007 ;CHECK THE NO RETRY SWITCH
855 003660 005337 017022 RNE HARDSK ;IF SET LOG HARD SEEK ERROR
856 003664 001404 DEC SRETRY ;HAVE 10 ERRORS BEEN LOGED
857 003666 004737 002550 BEQ HARDSK ;YES LOG HARD ERROR
858 003672 000137 003130 JSR PC,HARD ;RECALIBRATE DRIVES ON SEEK ERROR
859 003676 012703 017076 JMP UNEWRT ;NO,RETRY WRITE COMMAND (READ COMAND)
860 003702 004737 021044 MOV #ZSEKLOG,R3 ;SET INC. INST. FOR UNIT 0 ERR LOG
861 003706 005213 JSR PC,U1LOG ;TEST FOR UNIT 1 ERROR LOG
862 003710 104400 015240 INC (R5) ;HARD SEEK ERROR
863 003714 104400 015755 TYPE ,DUBLF
864 003720 104400 016010 TYPE ,MUNREC ;TYPE UNRECOVERABLE SEEK ERROR
865 003724 004737 003736 TYPE ,MWRITE ;ON WRITE (READ) COMMAND
866 003730 062706 000002 JSR PC,SEKTP
867 003734 000207 ADD #2,SP
868 RTS PC ;REMOVE SEEK ERROR FROM STACK POINTER
869 ;RETURN TO NEXT SECTOR BY DONE RETURN ON STACK
869 003736 104400 015775 SEKTP: TYPE ,MSEEK ;TYPE SEEK ERROR
870 003742 104400 014567 TYPE ,MPRES ;TYPE ADDRESS OF TRACK MOVED FROM
871 003746 013737 011362 003760 MOV #PRESTRK,1S
872 003754 004537 014446 JSR #5,SGLDEC
873 003760 000000 1S: OPEN
874 003762 104400 015052 TYPE ,MCRLF
875 003766 000207 RTS PC
876
877 003770 013705 011360 TRKERR: MOV #TARGET,R5 ;SET UP TO INC ERROR PER TRACK COUNTER
878 003774 006305 ASL R5 ;ADJUST FOR EVEN ADDRESS
879 003776 062705 020344 ADD #U0TRK,R5 ;ADDIN ADDRESS OF UNIT 0 LOG
880 004002 032737 000020 010704 BIT #BIT4,UNITSEL ;CHECK THE UNIT SELECTION BIT
881 004010 001402 BEQ IS ;IF CLEARED UNIT 0 IS ACTIVE
882 004012 062705 020576 ADD #U1TRK,R5 ;ADJUST FOR UNIT 1 LOG
883 004016 005215 INC (R5) ;INC THE CORRECT COUNTER
884 004020 000207 RTS PC
885
886 004022 013705 011360 COMMWORD: MOV #TARGET,R5 ;GET TRACK NUMBER
887 004026 006305 ASL R5 ;MULTIPLY BY 4 TO DOUBLE PRECISION
888 004030 006305 ASL R5 ;INTERLEAVE COUNTER LOCATIONS
889 004032 062705 017174 ADD #TKACC,R5 ;ADD ON ADDRESS OF TRACK ACCESS COUNTER
890 004036 062715 000001 ADD #1,(R5) ;INCREMENT THE COUNTER
891 004042 005565 000002 ADC #2,(R5) ;ADD CARRY TO HIGH ORDER WORD
892 004046 153737 010704 004120 B1S0 UNITSEL,FUNCTION ;SET UNIT SELECTION BIT IN COMMAND WORD
893 004054 013777 004120 175124 MOV #FUNCTION,#RXC5 ;SEND OUT COMMAND TO DRIVE
894 004062 004737 007014 JSR PC,TRCK ;WAIT FOR TR FLAG
895 004066 113777 012310 175114 MOVB #TSECTOR,#R5 ;SEND OUT TARGET SECTOR
896 004074 004737 007014 JSR PC,TRCK ;WAIT FOR TR FLAG

```

```

897 004100 113777 011360 175102      MOVB TARGET,0X0B      ;SEND OUT TARGET TEACK
898 004106 005046                      CLR -(SP)             ;FLOWER INTERRUPT LEVEL TO ALLOW AN INTERRUPT
899 004110 012746 004116              MOV #15,-(SP)
900 004114 000002                      RTI
901 004116 000207                      IS:                  RTS PC
902
903 004120 000000                      FUNCTION:            0
904 004122 000000                      NDAFTWT:            0
905 004124 000000                      ERWRT:              0
906 004126 000000                      DATAK:             0
907
908 ;*****
909
910 ;THIS IS A WRITE ALL SECTORS FOLLOWED BY A READ ALL SECTORS
911 ;WITH DATA PATTERNS AND HEAD SEQUENCING SET BY INITIAL SWR
912 ; T = 2
913
914 004130 005137 004122      WRTRD:              COM NDAFTWT      ;SET READ AFTER WRITE FLAG
915 004134 004737 010312      XWRTRD:            JSR PC,GETPATTERN
916 004140 004737 011144      JSR PC,INITTACKS
917 004144 004737 010614      JSR PC,GETUNIT
918 004150 004737 011254      IS:                 JSR PC,GETTRACK
919 004154 004737 003112      JSR PC,WRITE
920 004160 004737 004200      JSR PC,READ
921 004164 005337 011356      DEC TRKCNTR
922 004170 001367                      BNE IS
923 004172 004737 010732      JSR PC,STOP
924 004176 000760      BR XWRTRD
925
926 ;*****
927
928 ;READ DATA FROM THE DISKETTE
929
930
931 004200 004737 012216      HEAD:              JSR PC,INITSECTOR
932 004204 004737 012316      XHEAD:             JSR PC,GETSECTOR
933 004210 012737 000012 017010      MOV #10,,RDRETRY   ;SET UP RETRY COUNTERS FOR DELETED DATA
934 004216 012737 000012 017012      MOV #10,,DATARETRY ;FUATA ERROR
935 004224 012737 000012 017016      MOV #10,,PRETRY    ;PARITY
936 004232 012737 000012 017022      MOV #10,,SRETRY    ;SEEK
937 004240 012737 000012 017020      MOV #10,,CRETRY    ;AND CRC ERRORS
938 004246 005037 004126      REREAD:            CLR UATAK
939 004252 012746 004316      CLR #RDDONE,-(SP)  ;CLEAR CRC DATA CHECK FLAG
940 004256 012746 004460      MOV #RDERN,-(SP)  ;SET GOOD RETURN UN STACK
941 004262 112737 000107 004120      MOV #RDIE,FUNCTION ;SET READ ERROR RETURN UN STACK
942 004270 062737 000001 021034      ADD #1,RDNCNTR
943 004276 005337 021036      ADC RDNCNTR+2     ;INC TOTAL READ FUNCTIONS COUNTER
944 004302 004737 004022      JSR PC,COMMWORD   ;DOUBLE PRECISION COUNTER
945 004306 004737 007122      JSR PC,DDONECK
946 004312 000137 007204      JMP NDIRTER
947
948
949
950 004316 022737 000012 017004      RDDONE:            CMP #10,,RDRETRY   ;IS READ RETRY EQUAL TO 10
951 004324 001420      BEQ CONT1          ;YES,NO ERRORS OCCURED
952 004326 012703 017052      MOV #ZMOLUG,R3    ;SET INC. INST. FOR UNIT 0 ERROR LOG
  
```

```

953 004332 004737 021044      JSR PC,U1LOG       ;TEST FOR UNIT 1 ERROR LOG
954 004336 005213      INC (R3)           ;INC RECOVERABLE READ LOG
955 004340 012737 000012 017004      MOV #10,,RDRETRY   ;RESET READ RETRY COUNTER
956 004346 104400 015240      TYPE ,DBLLF
957 004352 104400 015627      TYPE ,MREK
958 004356 104400 015674      TYPE ,MREAD
959 004362 104400 015052      TYPE ,MCRLF
960 004366 022737 000012 017006      CONT1:             CMP #10,,WRETRY    ;IS WRITE RETRY EQUAL TO 10
961 004374 001420      BEQ CONT2          ;YES,NO ERRORS OCCURED
962 004376 012703 017056      MOV #ZWRLOG,R3    ;SET INC.INST.FOR UNIT 0 EROR LOG
963 004402 004737 021044      JSR PC,U1LOG
964 004406 005213      INC (R3)           ;INC RECOVERABLE WRITE LOG
965 004410 012737 000012 017006      MOV #10,,WRETRY    ;RESET THE WRITE RETRY COUNTER
966 004416 104400 015240      TYPE ,DBLLF
967 004422 104400 015627      TYPE ,MREK
968 004426 104400 015610      TYPE ,MWRITE
969 004432 104400 015052      TYPE ,MCRLF
970 004436 004737 005204      CONT2:             JSR PC,DDCHK
971 004442 005701      TST R1
972 004444 100001      BPL NEXTRD
973 004446 000207      RTS PC
974 004450 005337 012306      NEXTRD:            DEC SECCNTR
975 004454 001253      BNE XREAD
976 004456 000207      RTS PC
977
978 004460 005726      RDERR:             TST (SP)+
979 004462 032737 000002 010070      BIT #BIT1,ASTAT   ;REMOVE THE DONE RETURN FROM THE STACK
980 004470 001413      BEQ IS
981 ;PARITY ERROR DURING A READ FUNCTION
982 004472 012737 015674 003264      MOV #MHEAD,PTYP1+2 ;PUT ADDR OF READ MESSAGE IN PAR ERR TYPEOUT 1
983 004500 012737 015674 003354      MOV #MHEAD,PTYP2+2 ;PUT ADDR OF READ MESSAGE IN PAR ER TYPEOUT 2
984 004506 012737 004246 003334      MOV #RREAD,PCONT+2 ;IF HARD ERR RETURN THROUGH PCONT TO REREAD
985 004514 000137 003244      JMP PARTEST
986 004520 032737 000001 010070      IS:                 BIT #BIT0,ASTAT   ;IS THIS A CRC ERROR
987 004526 001014      BNE CRER          ;YES GO TEST AND LOG IT
988
989 ;SEEK ERROR DURING A READ FUNCTION
990 004530 012737 004246 003674      MOV #RREAD,SEKRTY+2 ;SET SEEK CONTINUE FOR READ RETRY
991 004536 012737 015674 003630      MOV #MHEAD,STYP1+2 ;SET ADDR OF READ MESSAGE IN SEEK ER TYPEOUT 1
992 004544 012737 015674 003722      MOV #MHEAD,STYP2+2 ;SET ADDR OF READ MESSAGE IN SEEK ER TYPEOUT 2
993 004552 004737 003576      JSR PC,SEEKER
994 004556 000734      BR NEXTRD
995 ;CRC ERROR DETECTED WHILE READING
996 004560 004737 003770      CRCER:             JSR PC,TRKERR
997 004564 005701      TST R1
998 004566 100061      BPL DATACRC
999 004570 005237 004126      INC UATAK
1000 004574 004737 005536      JSR PC,EMPBUFF
1001 004600 005737 006442      TST ERNCTR
1002 004604 001052      BNE DATACRC
1003 004606 012703 017046      MOV #ZCRCBAD,R3   ;YES,HEREAD AND/OR REWRITE THE DATA
1004 004612 004737 021044      JSR PC,U1LOG
1005 004616 005213      INC (R3)           ;TEST FOR UNIT 1 ERROR LOGS
1006 004620 104405      C#SWK
1007 004622 032777 020000 174364      BIT #SW13,SWR
1008 004630 001004      BNE IS
1009 004632 104400 015644      TYPE ,MBADCRC     ;TYPE CRC GENERATOR ERROR
  
```

1009	004636	104400	015052			TYPE ,MCRLF	
1010	004642	104405		2S:		CASWN	
1011	004644	005777	174344			TST #SWR	IF TEST HALT ON ERROR SWITCH
1012	004650	100001				BPL CONT15	
1013	004652	000000		HLT10:		HALT	IF HALT ON ERROR
1014	004654	032777	004000	174332	CONT15:	BIT #SW11,#SWR	IF CHECK NO RETRY SWITCH
1015	004662	001005				BNE 55	IF SET LOG HARD ERROR
1016	004664	005337	017020			DEC CRETRY	IF HAVE 10 ERRORS BEEN LOGED
1017	004670	001402				BEQ 35	IF YES, LOG HARD ERROR
1018	004672	000137	004246			JMP REREAD	IF NO, GO REREAD DATA
1019	004676	012703	017106		3S:	MOV #ZHCRCBAD,R3	IF SET INC, INST, FOR UNIT 0 ERROR LOG
1020	004702	004737	021044			JSR PC,U1LOG	IF TEST FOR UNIT 1 ERROR LOGS
1021	004706	005213				INC (R3)	
1022	004710	104400	015240			TYPE ,DBLLF	
1023	004714	104400	015644			TYPE ,MBADRC	IF TYPE HARD CRC GENERATOR ERROR
1024	004720	104400	015052			TYPE ,MCRLF	
1025	004724	104400	015724			TYPE ,MHALT11	IF AND HALT AS THE CRC GENERATOR DOESN'T WORK
1026	004730	000000		HLT11:		HALT	
1027							
1028	004732	012703	017042			DATA CRC ERROR REREAD AND/OR REWRITE TO GET GOOD DATA	
1029	004736	004737	021044		DATACRC:	MOV #ZCRCLOG,R3	IF SET INC, INST, FOR UNIT 0 ERROR LOG
1030	004742	005213				JSR PC,U1LOG	IF TEST FOR UNIT 1 ERROR LOGS
1031	004744	104405				INC (R3)	IF TRUE DATA CRC ERROR
1032	004746	032777	020000	174240		CASWN	
1033	004754	001004				BIT #SW13,#SWR	IF TEST DON'T PRINT ERROR SWITCH
1034	004756	104400	015736			BNE 45	
1035	004762	104400	015052			TYPE ,MCRC	IF TYPE DATA CRC ERROR
1036	004766	104405				TYPE ,MCRLF	
1037	004770	005777	174220		4S:	CASWN	
1038	004774	100001				TST #SWR	IF TEST HALT ON ERROR SWITCH
1039	004776	000000		HLT12:		BPL CONT16	
1040	005000	032777	004000	174206	CONT16:	HALT	IF HALT ON ERROR
1041	005006	001005				BIT #SW11,#SWR	IF TEST NO RETRY SWITCH
1042	005010	005337	017004			BNE 55	IF SET LOG HARD ERROR
1043	005014	001402				DEC RORETRY	IF HAVE 10 ERRORS OCCURRED
1044	005016	000137	004246			BEQ 55	IF YES LOG HARD ERROR OR REWRITE DATA
1045	005022	012703	017102		5S:	JMP REREAD	IF NO, GO REREAD THIS SECTOR
1046	005026	004737	021044			MOV #ZHCRCLOG,R3	IF SET INC, INST, FOR UNIT 0 ERROR LOG
1047	005032	005213				JSR PC,U1LOG	IF TEST FOR UNIT 1 ERROR LOGS
1048	005034	012737	000012	017004		INC (R3)	IF INC HARD CRC ERROR LOG
1049	005042	005737	004122			MOV #10, RORETRY	IF RESET READ RETRY COUNTER
1050	005046	001021				TST WDATA	IF IS THIS A HEAD AFTER WRITE FUNCTION
1051	005050	012703	017112			BNE CONT6	IF YES, GO REWRITE IT
1052	005054	004737	021044			MOV #ZHRDLOG,R3	IF NO, SET INC, INST, FOR UNIT 0 ERROR LOG
1053	005060	005213				JSR PC,U1LOG	IF TEST FOR UNIT 1 ERROR LOGS
1054	005062	104400	015240			INC (R3)	IF READ ONLY, HARD READ ERROR
1055	005066	104400	015755			TYPE ,DBLLF	
1056	005072	104400	015674			TYPE ,MUNREC	IF TYPE UNRECOVERABLE READ ERROR
1057	005076	104400	015052			TYPE ,MREAD	
1058	005102	062706	000002		CONT7:	TYPE ,MCRLF	
1059	005106	000137	004450			AUD #2,SP	IF REMOVE HEAD DONE ADDRESS FROM STACK
1060	005112	104405				JMP NEXTRD	IF READ NEXT SECTOR CAN'T READ THIS ONE
1061	005114	032777	004000	174072	CONT6:	CASWN	
1062	005122	001011				BIT #SW11,#SWR	IF TEST NO RETRY SWITCH
1063	005124	005337	017006			BNE 75	IF SET LOG HARD ERROR
1064	005130	001006				DEC WRETRY	IF HAVE 10 REWRITES BEEN DONE
						BEQ 75	IF YES, LOG HARD WRITE ERROR

1065	005132	005137	004124			COM ERWRT	IF NO, SET THE WRITE CAUSED BY ERROR FLAG
1066	005136	062706	000002			ADD #2,SP	IF REMOVE ADDRESS FROM STACK
1067	005142	000137	003130			JMP UNWRT	IF REWRITE SECTOR THEN REREAD TO CHECK FOR ERROR
1068	005146	012703	017116		7S:	MOV #ZHWRTLOG,R3	IF SET INC, INST, FOR UNIT 0 ERROR LOG
1069	005152	004737	021044			JSR PC,U1LOG	IF TEST FOR UNIT 1 ERROR LOGS
1070	005156	005213				INC (R3)	IF HARD WRITE ERROR
1071	005160	012737	000012	017006		MOV #10, WRETRY	IF RESET WRITE RETRY COUNTER
1072	005166	104400	015240			TYPE ,DBLLF	
1073	005172	104400	015755			TYPE ,MUNREC	IF TYPE UNRECOVERABLE WRITE ERROR
1074	005176	104400	016010			TYPE ,MWRITE	
1075	005202	000735				BR CONT7	
1076							
1077							
1078							
1079	005204	032737	001000	001212	DDCHK:	HIT #BIT9,DTESTP	IF TEST BIT 9 AS TO DELETED DATA TRANSFER
1080	005212	001472				BEQ CONT10	
1081	005214	132737	000100	010070		BIT# #BIT6,ASTAT	IF THIS IS A DELETED DATA FUNCTION
1082	005222	001114				BNE RETURN	IF DD BIT SHOULD BE SET
1083	005224	012703	017066			MOV #ZDDMIS,R3	IF SET INC INST FOR UNIT 0 ERROR LOG
1084	005230	004737	021044			JSR PC,U1LOG	IF TEST FOR UNIT 1 ERROR LOGS
1085	005234	005213				INC (R3)	IF INC MISSING DELETED DATA LOG
1086	005236	004737	005770			JSR PC,TRKERR	IF INC ERROR PER TRACK COUNTER
1087	005242	104405				CASWN	
1088	005244	032777	020000	173742		BIT #SW13,#SWR	IF TEST DON'T PRINT ERROR SWITCH
1089	005252	001011				BNE CONT11	
1090	005254	104400	014506			TYPE ,MDDMIS	IF TYPE MISSING DELETED DATA BIT
1091	005260	052737	000400	010704	DDERR:	HIS #BIT8,UNITSEL	IF SET HARD ERROR FLAG
1092	005266	004737	007716			JSR PC,TYPADR	IF TYPE ADDRESS OF ERROR
1093	005272	104400	015052			TYPE ,MCRLF	
1094	005276	104405				CASWN	
1095	005300	005777	175710		CONT11:	TST #SWR	IF TEST HALT ON ERROR SWITCH
1096	005304	100001				BPL CONT17	
1097	005306	000000		HLT13:		HALT	IF HALT ON DELETED DATA ERROR
1098	005310	032777	004000	173676	CONT17:	BIT #SW11,#SWR	IF TEST NO RETRY SWITCH
1099	005316	001005				BNE 45	IF SET LOG HARD ERROR
1100	005320	005337	017010			DEC UNRETRY	IF HAVE 10 ERRORS BEEN LOGED
1101	005324	001402				BEQ 45	IF YES LOG HARD ERROR
1102	005326	000137	004246			JMP REREAD	IF NO, REREAD SECTOR
1103	005332	012703	017126		4S:	MOV #ZDDOLOG,R3	IF SET INC INST, FOR UNIT 0 ERROR LOG
1104	005336	004737	021044			JSR PC,U1LOG	IF TEST FOR UNIT 1 ERROR LOGS
1105	005342	005213				INC (R3)	
1106	005344	104400	015240			TYPE ,DBLLF	
1107	005350	104400	015755			TYPE ,MUNREC	IF TYPE UNRECOVERABLE DELETED DATA ERROR
1108	005354	104400	015615			TYPE ,MODER	
1109	005360	104400	015052			TYPE ,MCRLF	
1110	005364	004737	007716			JSR PC,TYPADR	
1111	005370	104400	015052			TYPE ,MCRLF	
1112	005374	000137	004450			JMP NEXTRD	IF READ NEXT SECTOR
1113	005400	032737	000100	010070	CONT10:	BIT #BIT6,ASTAT	IF THIS IS NOT A DELETED DATA TRANSFER
1114	005406	001422				BEQ RETURN	
1115	005410	012703	017072			MOV #ZUNDDO,R3	IF SET INC INST, FOR UNIT 0 ERROR LOG
1116	005414	004737	021044			JSR PC,U1LOG	IF TEST FOR UNIT 1 ERROR LOGS
1117	005420	005213				INC (R3)	IF UNEXPECTED DD BIT SET
1118	005422	052737	000400	010704		HIS #BIT8,UNITSEL	IF SET HARD ERROR FLAG
1119	005430	004737	005770			JSR PC,TRKERR	IF INC ERROR PER TRACK COUNTER
1120	005434	104405				CASWN	

```

1121 005436 032777 020000 173550 BIT #SW13,#SWR ;TEST DON'T PRINT ERROR SWITCH
1122 005444 001314 BNE CONT11
1123 005446 104400 014462 TYPE ,MUNXUD ;TYPE UNEXPECTED DELETED DATA HIT
1124 005452 000702 BR UDERR
1125 005454 000207 RETURN: RTS PC
1126
1127
1128
1129
1130
1131 ;*****
1132 ;WRITE ALL SECTORS,READ AND VERIFY ALL SECTORS
1133 ; T = 3
1134 005456 005137 004122 WTRDCK: COM WDAFTWT ;SET READ AFTER WRITE FLAG
1135 005462 004737 010312 JSR PC,GETPATTERN
1136 005466 004737 011144 XWTRDCK: JSR PC,INITTRACKS
1137 005472 004737 010614 JSR PC,GETUNIT
1138 005476 004737 011254 1S: JSR PC,GETTRACK
1139 005502 004737 003112 JSR PC,WRITE
1140 005506 004737 005526 JSR PC,READCHK
1141 005512 005337 011356 DEC TRKNCTR
1142 005516 001367 HNE 1S
1143 005520 004737 JSR PC,STOP
1144 005524 000760 BR XWTRDCK
1145
1146
1147 ;*****
1148 ;READ A SECTOR,EMPTY THE SECTOR BUFFER AND VERIFY
1149 ;THE DATA READ AGAINST CORE DATA BUFFER
1150
1151
1152 005526 052701 100000 READCHK: BIS #BIT15,R1 ;SET READ ONE SECTOR FLAG
1153 005532 004737 004200 JSR PC,READ ;GO READ ONE SECTOR
1154 005536 005737 012306 EMPBUFF: TST SECCNTR ;IF CLEARED NO SECTOR WAS FOUND
1155 005542 001002 HNE 2S
1156 005544 000137 006432 JMP EXIT ;GO TO NEXT TRACK CAN'T READ THIS ONE
1157 005550 005037 006434 2S: CLR BYTECNTR ;CLEAR THE BYTE AND ERROR COUNTERS
1158 005554 005037 006442 CLR ERKNCTR
1159 005560 052701 000200 BIS #BIT7,R1 ;R1 BIT 7 IS USED AS FIRST ERROR FLAG
1160 005564 004737 005656 JSR PC,ADJSUM ;ADJUST DATA AND CK SUM FOR ADDRESSES
1161 005570 005037 005746 CLR CKSUM ;SET UP FOR CHECK SUM ACCUMULATION
1162 005574 012746 006212 MOV #EMPDONE,-(SP) ;SET UP RETURN ADDRESSES
1163 005578 012746 005750 MOV #EMPER,-(SP)
1164 005604 005046 CLR -(SP) ;LOWER INTERRUPT LEVEL
1165 005606 012746 005614 MOV #1S,-(SP)
1166 005612 000002 RTI
1167 005614 012777 000103 173364 1S: MOV #EMIE,#RXCS ;LOAD EMPTY BUFFER FUNCTION
1168 005622 004737 007014 EMPFLAG: JSR PC,TRCK ;TEST FOR TR FLAG
1169
1170 005626 117737 173356 006436 CKBYTE: MOV# #RX00,BAUBYTE ;SAVE BYTE FROM DISKETTE
1171 005634 063737 006436 005746 ADD BAUBYTE,CKSUM ;ACCUMULATE CHECK SUM
1172 005642 123720 006436 CMP# BAUBYTE,(R0)+ ;COMPARE AGAINST GOOD BYTE
1173 005646 001054 BNE DATAER ;IF NOT EQUAL GO TO DATAER
1174 005650 005237 006434 INC BYTECNTR
1175 005654 000762 BR EMPFLAG
1176

```

```

1177 005656 113737 011360 016604 ADJSUM: MOV# TARGET,BUFA0R ;SET FIRST AND SECOND BYTES WITH ADDRESSES
1178 005664 113737 012310 016605 MOV# TSECTOR,BUFA0R+1
1179 005672 013737 010612 005746 MOV SUM,CKSUM ;GET THE PATTERN SUM
1180 005700 063737 011360 005746 ADD TARGET,CKSUM ;ADD TRACK ADDRESS TO CHECK SUM
1181 005706 063737 012310 005746 ADD TSECTOR,CKSUM ;ADD SECTOR ADDRESS TO CHECK SUM
1182 005714 113737 005746 017002 MOV# CKSUM,BUFA0R+176 ;INSERT CHECK SUM TO DATA BUFFER
1183 005722 106337 005746 ASLE CKSUM ;GENERATE NEGATIVE CHECK SUM
1184 005726 105437 005746 NEGB CKSUM
1185 005732 113737 005746 017003 MOV# CKSUM,BUFA0R+177 ;INSERT NEG,SUM INTO DATA BUFFER
1186 005740 012700 016604 MOV #BUFA0R,R0 ;SET ADDRESS OF BYTE IN R0
1187 005744 000207 RTS PC ;RETURN
1188
1189 005746 000000 CKSUM: 0
1190
1191
1192 005750 005726 EMPER:
1193 005752 012737 015402 003264 TST (SP)+ ;REMOVE THE ODNE RETURN FROM THE STACK
1194 005760 012737 015402 003354 MOV #EMPTY,PTYP1+2 ;PUT ADDR OF EMPTYBUF MESSAGE IN PAR ER TYP0UT 1
1195 005766 012737 005536 003334 MOV #EMPTY,PTYP2+2 ;PUT ADDR OF EMPTYBUF MESSAGE IN PAR ER TYP0UT 2
1196 005774 000137 003244 MOV #EMPBUFF,PCONT+2 ;NOT HARD ERR RETURN THROUGH PCONT TO EMPYBUF
1197 006000 052737 000400 010704 DATAER: JMP PARTEST ;REPORT PARITY ERROR
1198 006006 005237 006442 BIS #BIT0,UNITSEL ;SET THE MAD ERROR FLAG
1199 006012 104405 INC ERKNCTR ;INC THE BYTE ERROR COUNTER
1200 006014 332777 020000 173172 CKSUM:
1201 006022 001062 BIT #SW13,#SWR ;TEST PRINT ERROR SW IN SWR
1202 006024 032777 010000 173162 BNE NOBRTYP ;DON'T PRINT THE ERROR
1203 006032 001404 BIT #SW12,#SWR ;TEST PRINT 10 ERRORS SWITCH
1204 006034 023727 006442 000012 HED 1S ;PRINT ALL ERRORS
1205 006042 003052 CMP ERKNCTR,#10. ;HAVE 10 ERRORS BEEN TYPED
1206 006044 005737 004126 BGT NOBRTYP ;YES,DON'T PRINT ANY MORE
1207 006050 001403 TST DATACK ;WAS THIS A READ CHECK DUE TO CRC ERROR
1208 006052 105701 REO 2S ;NO,REPORT DATA ERRORS
1209 006054 100017 TST# R1 ;YES,TEST FIRST ERROR FLAG
1210 006056 000412 BPL TYPERR ;TYPE THE DATA CRC ERROR
1211 006060 105701 BR 3S
1212 006062 130014 2S: TST# R1 ;TEST FIRST ERROR FLAG
1213 006064 104400 015052 BPL TYPERR
1214 006070 104400 014527 TYPE ,MCRFLF ;PRINT ERROR HEADER
1215 006074 104400 015052 TYPE ,MDEKHDR
1216 006100 004737 007716 TYPE ,MCRFLF ;PRINT TRACK AND SECTOR LOCATIONS
1217 006104 104400 014616 JSR PC,TYPADR ;SET UP COLMUN HEADINGS
1218 006110 042701 000200 BIT #BIT7,R1 ;CLEAR FIRST ERROR FLAG
1219 006114 013737 006434 006126 TYPERR: MOV BYTECNTR,1S ;PRINT BYTE NUMBER
1220 006122 004537 014446 JSR #5,SGLDEC
1221 006126 000000 1S: OPEN
1222 006130 104400 015047 TYPE ,DBLSP
1223 006134 013746 006436 MOV #BADBYTE,-(SP) ;PRINT BYTE READ FROM DISKETTE
1224 006140 104400 TPOS
1225 006142 000003 ,WORD 3
1226 006144 104400 015047 TYPE ,DBLSP
1227 006150 114037 006440 MOV# -(R0),GOODBYTE ;GET GOOD BYTE
1228 006154 005200 INC R0 ;RETURN R0 TO NEXT BYTE IN BUFFER
1229 006156 013746 006440 MOV GOODBYTE,-(SP)
1230 006162 104403 TYPON ;PRINT GOOD DATA
1231 006164 104400 015052 TYPE ,MCRFLF
1232 006170 104405 NOERTYP: CKSWK

```

```

1233 006172 005777 173016          TST #SWR          ITEST HALT ON ERROR SWITCH
1234 006176 100001          BPL CONT20
1235 006200 000000          HALT
1236 006202 005237          INC BYTECNTR
1237 006206 000137 005022          JMP EMPFLAG
1238
1239 006212 005737 004126          EMPDONE:        TST DATACK          ;WAS THIS READCHECK CAUSED BY CRC ERROR
1240 006216 001401          BEQ 15          ;NO,CONTINUE
1241 006220 000207          RTS PC          ;YES,RETURN TO CRC ERROR HANDLER
1242 006222 005737 006442          15:            TST ER CNTR        ;WAS THERE ERRORS
1243 006226 001471          BEQ CONT3       ;NO ERRORS
1244 006230 104405          CKS#W
1245 006232 032777 020000 172754          BIT #SW13,#SWR    ;YES,TEST DON'T PRINT SWITCH
1246 006240 001024          BNE 25          ;DON'T PRINT THE ERROR
1247 006242 104400 015333          TYPE ,MERCCT     ;PRINT THE TOTAL DATA ERROR COUNT
1248 006246 013737 006442 006260          MOV ER CNTR,35
1249 006250 004537 014446          JSR #5,SGLDEC
1250 006250 000000          OPEN
1251 006262 104400 016327          TYPE ,MSUM       ;INDICATE IF CHECK SUM WAS GOOD OR HAD ERRORS
1252 006266 105737 005746          TSTB CKSUM
1253 006272 001403          BEQ 45
1254 006274 104400 016067          TYPE ,MERS
1255 006300 000402          BR 55
1256 006302 104400 016322          45:            TYPE ,MGOOD
1257 006306 104400 015052          55:            TYPE ,MCRLF
1258 006312 012703 017062          25:            MOV #ZDATALOG,R3
1259 006316 004737 021044          JSR PC,U1LOG     ;SET INC,INST,FOR UNIT 0 ERROR LOG
1260 006322 005213          INC (R3)         ;TEST FOR UNIT 1 ERROR LOGS
1261 006324 004737 003770          JSR PC,TRKERR   ;INC DATA ERROR LOG
1262 006330 104405          CKS#W           ;INC ERROR PER TRACK COUNTER
1263 006332 032777 004000 172654          BIT #SW11,#SWR   ;TEST NO RETRY SWITCH
1264 006340 001007          BNE 65          ;IF SET LOG HARD ERROR
1265 006342 005337 017012          DEC DATARETRY   ;HAVE 10 ERRORS BEEN LOGED
1266 006346 001404          BEQ 65
1267 006350 004737 004246          JSR PC,REHEAD   ;NO,GO REREAD THE DATA
1268 006354 000137 005536          JMP EMPBUFF     ;GO RECHECK THE DATA
1269 006360 012703 017122          MOV #ZDATALOG,R3 ;YES,SET INC,INST,FOR UNIT 0 ERROR LOG
1270 006364 004737 021044          JSR PC,U1LOG     ;TEST FOR UNIT 1 ERROR LOGS
1271 006370 005213          INC (R3)         ;INC HARD DATA NO STATUS ERROR
1272 006372 104400 015240          TYPE ,URLLF
1273 006376 104400 015755          TYPE ,MUNREC    ;TYPE UNRECOVERABLE DATA NO
1274 006402 104400 014527          TYPE ,MOERHND   ;STATUS ERROR
1275 006406 104400 015052          TYPE ,MCRLF
1276 006412 005337 012306          CONT3:         DEC SECCNTR
1277 006416 001405          BEQ EXIT1
1278 006420 004737 004204          JSR PC,XREAD    ;READ THE NEXT SECTOR
1279 006424 000137 005536          JMP EMPBUFF
1280 006430 005001          CLR R1          ;CLEAR THE ONE READ FLAG
1281 006432 000207          EXIT1:         RTS PC
1282
1283 006434 000000          BYTECNTR:      0
1284 006436 000000          HADBYTE:       0
1285 006440 000000          GOODBYTE:     0
1286 006442 000000          ER CNTR:       0
1287
1288 ;*****

```

```

1289
1290 ;READ AND VERIFY ALL SECTORS WRITTEN BY
1291 ;PREVIOUS WRITE FUNCTION
1292
1293 ; T = 4
1294
1295 006444 004737 010312          RDCHK:         JSR PC,GETPATTERN
1296 006450 004737 011144          XRDKCHK:      JSR PC,INITTRACKS
1297 006454 004737 010614          JSR PC,GETUNIT
1298 006460 004737 011254          15:            JSR PC,GETTRACK
1299 006464 004737 005526          JSR PC,READCHK
1300 006470 005337 011356          DEC TRKCNTR
1301 006474 001371          HNE 15
1302 006476 004737 010732          JSR PC,STOP
1303 006502 000762          BR XRDKCHK
1304
1305
1306 ;*****
1307
1308 ;DO A READ ONLY FUNCTION ON ALL SECTORS
1309 ;THIS DOES NOT VERIFY THE DATA, ONLY TESTS FOR CRC ERRORS
1310 ; T = 5
1311
1312 006504 004737 010312          RDONLY:       JSR PC,GETPATTERN
1313 006510 004737 011144          XRONLY:      JSR PC,INITTRACKS
1314 006514 004737 010614          JSR PC,GETUNIT
1315 006520 004737 011254          15:            JSR PC,GETTRACK
1316 006524 004737 004200          JSR PC,READ
1317 006530 005337 011356          DEC TRKCNTR
1318 006534 001371          HNE 15
1319 006536 004737 010732          JSR PC,STOP
1320 006542 000762          BR XRONLY
1321
1322
1323 ;*****
1324
1325 ;WRITE AND READ CHECK ANY DATA PATTERN,AND ANY TRACK SEQUENCE
1326 ;BUT ALTERNATE BETWEEN DRIVE UNITS ON EACH TRACK SELECTED.
1327 ; T = 6
1328
1329
1330 006544 005137 004122          DRVSWP:      COM WDAFTWT     ;SET THE READ AFTER WRITE FLAG
1331 006550 004737 010312          JSR PC,GETPATTERN
1332 006554 004737 011144          25:            JSR PC,INITTRACKS
1333 006560 004737 010614          15:            JSR PC,GETUNIT     ;SET UP FOR UNIT 0
1334 006564 004737 011254          JSR PC,GETTRACK ;GET THE TRACK TO BE ACCESSED
1335 006570 004737 003112          JSR PC,WRITE
1336 006574 004737 005526          JSR PC,READCHK
1337 006600 004737 010614          JSR PC,GETUNIT ;NOW GO TO UNIT 1 ON THE SAME TRACK
1338 006604 004737 003112          JSR PC,WRITE
1339 006610 004737 005526          JSR PC,READCHK
1340 006614 005337 011356          DEC TRKCNTR
1341 006620 001357          BNE 15
1342 006622 004737 010732          JSR PC,STOP
1343 006626 000752          BR 25
1344

```

```

1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358 006630 004737 010312 TEST7: JSR PC,GETPATTERN
1359 006634 013737 011370 007012 MOV SEQUEN,SEQSAV ;SAVE THE SELECTED SEQUENCE
1360 006642 012737 000001 011370 TEST7X1: MOV #1,SEQUEN ;FORCE INCREMENT SEQUENCE
1361 006650 005137 004122 COM #0AFTWT ;SET READ AFTER WRITE FLAG FOR ERROR TESTING
1362 006654 004737 011144 JSR PC,INITTRACKS
1363 006660 004737 010614 JSR PC,GETUNIT
1364 006664 004737 011254 1S: JSR PC,GETTRACK
1365 006670 004737 005112 JSR PC,WRITE
1366 006674 004737 005526 JSR PC,READCHK
1367 006700 005337 011356 DEC TRKCNTR ;IS THE FIRST HALF OF THE PASS DONE
1368 006704 001367 BNE 1S ;NO,CONTINUE WRITING AND READING
1369 006706 005037 004122 CLR #0AFTWT ;YES,NOW DO A READ ONLY
1370 006712 013737 007012 011370 MOV SEQUAV,SEQUEN ;USING SELECTED SEQUENCE
1371 006720 004737 011144 JSR PC,INITTRACKS
1372 006724 004737 011254 2S: JSR PC,GETTRACK
1373 006730 004737 005526 JSR PC,READCHK
1374 006734 005337 011356 DEC TRKCNTR ;IS THIS HALF OF THE PASS DONE
1375 006740 001371 BNE 2S ;NO,CONTINUE READ CHECKING
1376 006742 004737 010752 JSR PC,STUP ;YES,TEST FOR END OF PASS CONDITIONS
1377 006746 032737 040100 010704 BIT #40100,UNITSEL ;HAVE ALL SELECTED UNITS BEEN USED
1378 006754 001332 BNE TEST7X ;IF A USED BIT IS STILL SET GO TO NEXT DRIVE
1379 006756 012737 001200 MOV #1200,SP ;RESET STACK ADDRESS FOR ACT11 EOP
1380 006762 032737 001000 001212 HIT #BIT9,UTESTP ;NEXT PASS,COMPLEMENT D O BIT
1381 006770 001404 BEQ 3S
1382 006772 042737 001000 001212 MIC #BIT9,UTESTP ;IT WAS ON CLEAR IT
1383 007000 007020 MP TEST7X
1384 007002 052737 001000 001212 3S: HIS #BIT9,UTESTP ;IT WAS OFF SET IT
1385 007010 007014 HR TEST7X
1386
1387 007012 000000 SEQSAV: 0
1388
1389
1390
1391
1392 007014 005037 007200 TRCA: CLR HCNTR1 ;CLEAR HUNG COUNTER
1393 007020 012746 000540 MOV #PH7,-(SP) ;RAISE INTERRUPT LEVEL TO INHIBIT INTERRUPTS
1394 007024 012746 007032 MOV #3S,-(SP)
1395 007030 000002 RTI
1396 007032 105777 172150 3S: TST #MXCS ;TEST FOR TR FLAG
1397 007036 100001 HPL 1S ;SKIP NEXT INST,IF NOT SET
1398 007040 000207 RTS PC ;RETURN TO TRANSFER DATA
1399 007042 023727 006434 000200 1S: CMP BYTECNTR,#128. ;HAVE 128 BYTES BEEN TRANSFERED
1400 007050 001403 BEQ 2S ;ALLOW AN INTERRUPT IF EQUAL
  
```

```

1401 007052 005237 007200 INC HCNTR1 ;IF THE HUNG COUNTER OVERFLOWS EITHER
1402 ;THERE WAS NO TR FLAG OR NO INTERRUPT OCCURED
1403 ;AFTER 128 BYTES TRANSFERED
1404 007056 001365 BNE 1S ;NO OVERFLOW RETEST FOR TR FLAG
1405 007060 132777 000040 172120 2S: BITD #DUNEMIT,0RXCS ;TEST FOR A DONE FLAG WHICH WOULD INDICATE A
1406 ;POSSIBLE ERROR CONDITION EXISTS WHILE
1407 ;TRANSFERRING COMMANDS OR DATA
1408 007066 001407 BEQ 4S ;IF DONE NOT SET,TYPE HUNG
1409 007070 005037 006434 CLR BYTECNTR ;CLEAR BYTE COUNTER SO IT WON'T INTERFERE
1410 ;WITH TESTING FOR DONE.
1411 007074 005726 TST (SP)+ ;REMOVE THIS RETURN ADDRESS FROM STACK
1412 007076 005046 CLR -(SP) ;LOWER INTERRUPT LEVEL TO ACCEPT AN INTERRUPT
1413 007100 012746 007106 MOV #4S,-(SP) ;THE INTERRUPT WILL RETURN THROUGH PREVIOUSLY
1414 007104 000002 RTI ;SET ADDRESSES
1415 007106 005237 007200 4S: INC HCNTR1 ;ALLOW SOME TIME FOR FLAG TO SET
1416 007112 001362 BNE 2S
1417 007114 104400 015432 TYPE ,MHUNG
1418 007120 000000 HALT ;PROGRAM HUNG, SP (177706) CONTAINS RETURN
1419 ;ADDRESS OF WAITING LOOP
1420
1421
1422 007122 012603 DUNECK: ;TEST FOR DONE FLAG AND PROGRAM HUNG
1423 MOV (SP)+,R3 ;TAKE THE RETURN ADDRESS OF THE STACK INCASE
1424 ;THE PROGRAM IS EXPECTING AN INTERRUPT AT DONE
1425 007130 012737 177740 007202 CLR HCNTR1 ;CLEAR THE LOWER HUNG COUNTER
1426 007136 032777 000040 172042 2S: MOV #177740,HCNTR2 ;SET UPPER COUNTER FOR 31 FULL LOOPS
1427 007144 001402 BIT #DUNEMIT,0RXCS ;TEST FOR DONE FLAG
1428 007146 010346 BEQ 1S ;IF NOT SET SKIP RETURN
1429 007150 000207 MOV R3,-(SP) ;PUT RETURN ADDRESS BACK ON THE STACK
1430 007152 002737 000001 007200 1S: RTS PC ;RETURN
1431 007160 005537 007202 ADD #1,HCNTR1 ;WHEN THE DOUBLE WORD COUNTER OVERFLOWS
1432 007164 005737 007202 ADC HCNTR2 ;THE PROGRAM MUST BE HUNG
1433 007170 001362 TST HCNTR2
1434 007172 104400 015432 BNE 2S ;IF NO OVERFLOW RETEST FOR DONE
1435 007176 000000 TYPE ,MHUNG
1436 HALT ;PROGRAM HUNG,CONTENTS OF R3 (177703) HOLDES
1437 ;ADDRESS OF WAITING TEST.
1438
1439 007200 000000 HCNTR1: 0
1440 007202 177740 HCNTR2: 177740
1441
1442
1443 ;AN INTERRUPT DID NOT OCCURE AT A FUNCTION DONE FLAG.
1444
1445 007204 005237 017034 NUNINTER: INC INTER ;INC INTERRUPT ERROR COUNTER
1446 007210 104405 CKSWH
1447 007212 032777 020000 171774 BIT #SW13,0SWR ;TEST DON'T PRINT ERROR SWITCH
1448 007220 001004 BNE 1S
1449 007222 104400 015260 TYPE ,MINTER ;TYPE NO INTERRUPT ON DONE ERROR
1450 007226 104400 015052 TYPE ,MCRLF
1451 007232 104405 1S: CKSWH
1452 007234 005777 TST #SWR ;TEST HALT ON ERROR SWITCH
1453 007240 100001 RPL CONT21
1454 007242 000000 HALT ;HALT ON ERROR
1455 007244 004737 007250 CONT21: JSR PC,INTSERV ;JSR TO INTSERV AS IF IT WAS AN INTERRUPT
1456
  
```



```

1569 010044 001 ,BYTE 1 //TYPE 1 DIGIT(S)
1570 010045 000 ,BYTE 0 //SUPPRESS LEADING ZEROS
1571 010046 104400 015047 TYPE ,DBLSP
1572 010052 104400 015254 TYPE ,MSEQ //TYPE SEQUENCE CODE
1573 010056 013746 011370 SEQUEN,=(SP) //SAVE SEQUEN FOR TYPEOUT
1574 010062 104402 TYPOS //GO TYPE--OCTAL ASCII
1575 010064 001 ,BYTE 1 //TYPE 1 DIGIT(S)
1576 010065 000 ,BYTE 0 //SUPPRESS LEADING ZEROS
1577 010066 000207 HTS PC
1578
1579 010070 000000 ASTAT: 0
1580 010072 000000 BSTAT: 0
1581
1582 010074 132737 000020 004120 DELUNIT: HITB #BIT4,FUNCTION //TEST FOR PRESENTLY USED UNIT
1583 010102 001016 BNE JS
1584 010104 104400 015106 TYPE ,MUNIT0 //UNIT 0 HAS BEEN DELETED FROM USE
1585 010110 104400 015126 TYPE ,MDELET
1586 010114 104400 015240 TYPE ,DBLLF
1587 010120 042737 000200 010704 HIC #BIT7,UNITSEL //CLEAR UNIT 0 SELECTION BIT
1588 010126 005737 010704 TST UNITSEL //WAS UNIT 1 SELECTED FOR USE
1589 010132 100020 BPL 15 //UNIT 1 NOT SELECTED GO DUMP ERROR REPORT
1590 010134 000137 002726 2S: JMP TESTSEL //CONTINUE ON OTHER UNIT AT BEGINING OF TEST
1591 010140 104400 015116 3S: TYPE ,MUNIT1 //UNIT 1 HAS BEEN DELETED FROM USE
1592 010144 104400 015126 TYPE ,MDELET
1593 010150 104400 015240 TYPE ,DBLLF
1594 010154 042737 100000 010704 HIC #BIT15,UNITSEL //CLEAR UNIT 1 SELECTION BIT
1595 010162 105737 010704 TST0 UNITSEL //WAS UNIT 0 SELECTED FOR USE
1596 010166 100762 BMI 25 //YES CONTINUE ON UNIT 0
1597 010170 005137 022514 COM SHORTRPT //SET SHORT REPORT FLAG
1598 010174 000137 021062 1S: JMP ENDUMP //CAN'T CONTINUE TYPE OUT STATISTICAL REPORT
1599
1600
1601 010200 117737 171004 010070 HOCODE: MOV0 #HX0B,ASTAT //SAVE THE A STATUS
1602 010206 012777 000017 170772 2S: MOV #RDER,#RXCS //READ THE B STATUS REGISTER
1603 010214 004737 007122 JSR PC,DONECK //WAIT FOR DONE FLAG
1604 010220 032777 000002 170762 BIT #2,#RX0B //WAS THERE A PARITY ERROR
1605 010226 001003 HEQ 15 //NO,CONTINUE
1606 010230 004737 002046 JSR PC,STATEN //YES,REPORT THE PARITY ERROR
1607 010234 000764 B# 25 //RETRY READING STATUS B
1608 010236 117737 170746 010072 1S: MOV0 #HX0B,BSTAT //SAVE THE B STATUS CODES
1609 010244 104400 015052 TYPE ,MCRLF
1610 010250 104400 015210 TYPCODE: TYPE ,MASTAT //TYPE THE CONTENTS OF THE TWO STATUS REGISTERS
1611 010254 013746 010070 MOV ASTAT,=(SP) //SAVE ASTAT FOR TYPEOUT
1612 010260 104402 TYPOS //GO TYPE--OCTAL ASCII
1613 010262 005 ,BYTE 3 //TYPE 3 DIGIT(S)
1614 010263 000 ,BYTE 0 //SUPPRESS LEADING ZEROS
1615 010264 104400 015040 TYPE ,TAB
1616 010270 104400 015224 TYPE ,M0STAT
1617 010274 013746 010072 MOV BSTAT,=(SP) //SAVE BSTAT FOR TYPEOUT
1618 010300 104402 TYPOS //GO TYPE--OCTAL ASCII
1619 010302 005 ,BYTE 3 //TYPE 3 DIGIT(S)
1620 010303 000 ,BYTE 0 //SUPPRESS LEADING ZEROS
1621 010304 104400 015052 TYPE ,MCRLF
1622 010310 000207 HTS PC
1623
1624

```

```

1625 //*****
1626
1627 //DECODES THE DATA PATTERN SELECTED IN THE SWR
1628 //
1629 //BITS 6,7,AND 8 OF THE INITIAL SWR SELECTED THE DATA PATTERN
1630 //TO BE USED AS FOLLOWS:
1631 //
1632 // BITS DATA PATTERN
1633 // 8 7 6
1634 // 0 0 0 NO PATTERN SPECIFIED (FORCE RANDOM DATA)
1635 // 0 0 1 ALL ZEROS
1636 // 0 1 0 ALL ONES
1637 // 0 1 1 FLOATING ZERO
1638 // 1 0 0 FLOATING ONE
1639 // 1 0 1 ALTERNATING BITS
1640 // 1 1 0 ALTERNATING PAIRS OF BITS
1641 // 1 1 1 RANDOM
1642 //
1643 //NOTE: ALL DATA PATTERNS WILL BE MODIFIED SO THE FIRST BYTE WILL
1644 //CONTAIN THE TRACK ADDRESS, THE SECOND BYTE WILL CONTAIN THE UNIT
1645 //NUMBER AND SECTOR ADDRESS IN WHICH THE DATA IS WRITTEN, THE LAST
1646 //TWO BYTES CONTAIN THE CHECK SUM NUMBERS.
1647 //*****
1648
1649
1650
1651 010312 142737 000377 010360 GETPATTERN: HIC #377,#BRONPAT //CLEAR BRANCH OFFSET
1652 010320 005037 010612 CLR SUM //SET UP FOR ACCUMULATION OF CHECK SUM
1653 010324 005737 010420 TST PAT //IF NO PATTERN SPECIFIED FORCE PATTERN 7
1654 010330 001003 BNE 15
1655 010332 012737 000007 010420 1S: MOV #7,PAT //GET PATTERN BITS
1656 010340 013704 010420 MOV PAT,R4 //ADJUST FOR CORRECT OFFSET
1657 010344 005304 UEC #4
1658 010346 006304 ASL #4
1659 010350 150437 010360 HIB #R4,#BRONPAT //INSERT OFFSET
1660 010354 012704 010606 MOV #BUFADR+2,R4 //SET UP ADDRESS OF FIRST BYTE
1661 010360 000777 BR . //BRANCH BY OFFSET SELECTED
1662 010362 000137 010422 JMP UATA0 //000 DATA BYTE
1663 010366 000137 010434 JMP UATA1 //377 DATA BYTE
1664 010372 000137 010444 JMP FLUAT0 //FLOAT A 0 THROUGH ALL 1'S
1665 010376 000137 010506 JMP FLUAT1 //FLOAT A 1 THROUGH ALL 0'S
1666 010402 000137 010514 JMP PAT125 //125/052 DATA WORD
1667 010406 000137 010534 JMP PAT314 //314/063 DATA WORD
1668 010412 000137 010544 JMP RANDATA //RANDOM DATA BYTE
1669
1670
1671 010416 000000 DATABYTE: 0
1672 010420 000000 PAT: 0
1673
1674
1675 //*****
1676
1677 //LOAD SOFTWARE BUFFER WITH ALL ZEROS
1678 // P = 1
1679
1680 010422 005037 010416 DATA0: CLR DATABYTE

```



```

1681 010426 004737 010564 PATGEN: JSR PC,LOAD ;GO LOAD THE DATA BUFFER
1682 010432 000775 BR PATGEN
1683
1684 ;*****
1685
1686 ;LOAD SOFTWARE BUFFER WITH ALL ONES
1687 ; P = 2
1688
1689
1690 010434 112737 000377 010416 DATA1: MOVW #577,DATABYTE
1691 010442 000771 BR PATGEN
1692
1693 ;*****
1694
1695 ;FLOAT A 0 THROUGH ONES IN SOFTWARE BUFFER
1696 ; P = 3
1697
1698
1699 010444 112737 000376 010416 FLDAT0: MOVW #576,DATABYTE ;SET UP A ONES FIELD
1700 010452 000261 XPATGEN: SEC ;SET THE C BIT TO ROTATE THROUGH THE DATA
1701 010454 012702 000000 1S: MOV #0,R2 ;CLR R2 (CAN'T USE "CLR" AS IT CLEARS "C" BIT)
1702 010460 103001 BCC ZS ;BR IF THE "C" BIT IS CLEARED
1703 010462 005202 INC R2 ;SET R2 IF NOT
1704 010464 004737 010564 2S: JSR PC,LOAD ;GO LOAD THE DATA BUFFER
1705 010470 000241 CLC
1706 010472 005702 TST R2 ;IS R2 NONZERO
1707 010474 001401 BEQ JS
1708 010476 000261 SEC ;YES, SET THE "C" BIT
1709 010500 106137 010416 3S: RDL DATABYTE
1710 010504 000763 BR 1S
1711
1712 ;*****
1713
1714 ;FLOAT A 1 THROUGH ALL ZEROS IN SOFTWARE BUFFER
1715 ; P = 4
1716
1717 010506 005037 010416 FLDAT1: CLR DATABYTE
1718 010512 000757 BR XPATGEN
1719
1720 ;*****
1721
1722 ;LOAD SOFTWARE BUFFER WITH ALTERNATING 1 AND 0 FOR
1723 ;ONE BYTE AND THE COMPLIMENT INTO THE NEXT
1724 ; P = 5
1725
1726 010514 112737 000125 010416 PAT125: MOVW #125,DATABYTE
1727 010522 004737 010564 XXPATGEN: JSR PC,LOAD
1728 010526 105137 010416 CMB DATABYTE ; * (SEE NOTE BELOW)
1729 010532 000773 BR XXPATGEN
1730
1731 ; * NOTE: TO MAKE PATTERN 5 AND 6 COMPATABLE WITH THE RXB PATTERNS
1732 ;NOP THIS INSTRUCTION. (CHANGE THESE 2 LOCATIONS TO #00240)
1733 ;THIS CHANGE IS FOR INTERPROCESSOR RX** COMPATABILITY TESTING ONLY.
1734
1735 ;*****
1736

```

```

1737 ;LOAD SOFTWARE BUFFER WITH ALTERNATING PAIRS OF 1 AND 0 AND
1738 ;COMPLIMENT INTO THE NEXT
1739 ; P = 6
1740
1741 010534 112737 000314 010416 PAT314: MOVW #514,DATABYTE
1742 010542 000767 BR XXPATGEN
1743
1744 ;*****
1745
1746 ;LOAD SOFTWARE BUFFER WITH RANDOM DATA PATTERN
1747 ; P = 0 OR 7
1748
1749 010544 004737 012122 RANDATA: JSR PC,RANGEN ;GET RANDOM NUMBER
1750 010550 113737 012214 010416 MOVW RANUM,DATABYTE
1751 010556 004737 010564 JSR PC,LOAD
1752 010562 000770 BR RANDATA
1753
1754 010564 063737 010416 010612 LOAD: ADD DATABYTE,SUM ;ACCUMULATE THE PATTERN CHECK SUM
1755 010572 113724 010416 FLDW DATABYTE,(R4)+ ;LOAD THE DATA BUFFER
1756 010576 022794 017002 CMP #0UFA0H+176,R4 ;HAVE 124 BYTES BEEN GENERATED
1757 010602 001401 BEQ 1S ;IF YES,RETURN TO TEST
1758 010604 000207 RTS PC ;IF NO,RETURN TO PATTERN GENERATOR
1759 010606 005726 1S: TST (SP)+ ;TAKE PATTERN RETURN ADDRESS OF STACK
1760 010610 000207 RTS PC ;RETURN TO TEST
1761
1762 010612 000000 SUM: 0
1763
1764 ;*****
1765
1766 ;TEST FOR SELECTED UNITS,DRIVE READY,AND USED CONDITIONS
1767 ;ALSO CONTAINS A "HAD ERROR" FLAG TO BE TESTED AT EOP.
1768 ;THE BITS IN UNITSEL ARE USED AS FOLLOWS
1769 ;
1770 ;BIT15 =UNIT 1 SELECTED VIA SWR
1771 ;BIT14 =UNIT 1 USED BIT
1772 ;BIT8 =THIS PASS HAD AN ERROR
1773 ;BIT7 =UNIT 0 SELECTED VIA SWR
1774 ;BIT6 =UNIT 0 USED BIT
1775 ;BIT4 =UNIT SELECTION FOR FUNCTION WORD
1776
1777 ;*****
1778
1779 010614 032737 000100 010704 GETUNIT: BIT #BIT6,UNITSEL ;WAS UNIT 0 JUST USED
1780 010622 001012 BNE 1S ;UNIT 0 USED CHECK UNIT 1
1781 010624 105737 010704 TSTM UNITSEL ;WAS UNIT 0 SELECTED
1782 010630 100007 RPL 1S ;NO GO TO UNIT 1
1783 010632 042737 000020 010704 RLC #40020,UNITSEL ;CLEAR UNIT 1 USED BIT AND FUNCTION UNIT BIT
1784 010640 052737 000100 010704 BIT #BIT6,UNITSEL ;SET UNIT 0 USED BIT
1785 010646 000207 RTS PC
1786 010650 005737 010704 1S: TST UNITSEL ;WAS UNIT 1 SELECTED
1787 010654 100012 RPL ZS ;NO RETURN
1788 010656 032737 040000 010704 BIT #BIT14,UNITSEL ;WAS UNIT 1 BEEN USED
1789 010664 001006 BNE ZS ;YES RETURN
1790 010666 042737 000100 010704 BIC #BIT6,UNITSEL ;CLEAR UNIT 0 USED BIT
1791 010674 052737 040020 010704 BIS #40020,UNITSEL ;SET UNIT 1 USED BIT AND FUNCTION UNIT BIT
1792 010702 00020F 2S: RTS PC

```

```

1793
1794
1795
1796 010704 000000          UNITSSEL:      0
1797
1798          ;TEST THAT ALL UNITS HAVE BEEN ACCESSED
1799
1800 010706 005737 010704      DONE:          TST UNITSSEL          ;IS UNIT 1 SELECTED
1801 010712 100000              BPL 15                ;NO RETURN
1802 010714 032737 040000 010704  BIT #BIT14,UNITSSEL  ;YES HAS IT BEEN USED
1803 010722 001000              HNE 15                ;YES RETURN
1804 010724 062706 000002      15:          ADD #2,SP            ;BYPASS NOT DONE RETURE ON STACK
1805 010730 000207
1806
1807
1808 010732 004737 010706      STOP:          JSR PC,DONE          ;HAVE ALL DRIVES BEEN USED
1809 010736 005737 007200              CLR MCNTR1
1810 010742 005237 021042              INC PASCNTR          ;KEEP TOTAL NUMBER OF PASSES
1811 010746 005737 011142              TST CHARCT          ;HAVE 72 EOP INDICATORS BEEN TYPED
1812 010752 001005              HNE 55
1813 010754 104400 015052              TYPE ,MCRLF          ;YES DO A CRLF
1814 010760 012737 177670 011142      35:          MOV #-/2,,CHARCT    ;RESET CHARACTER COUNTER
1815 010766 005237 011142              INC CHARCT
1816 010772 032737 000400 010704      BIT #BIT0,UNITSSEL  ;TEST HAD ERROR FLAG
1817 011000 001405              BEQ 25
1818 011002 104400 015057              TYPE ,MEREUP         ;HAD ERROR PRINT ?
1819 011006 000404              BR 15
1820 011010 104400 015055              25:          TYPE ,MEOP
1821 011014 104400 015061              TYPE ,MABELL
1822 011020 104405
1823 011022 032777 040000 170164      15:          HIT #SW14,#SWR      ;TEST EOP HALT SWITCH
1824 011030 001417
1825 011032 104400 015052              TYPE ,MCRLF
1826 011036 104400 010142              35:          TYPE ,MPASS          ;AT HALT UN PASS PRINT NUMBER OF PASSES COMPLETE
1827 011042 013737 021042 011054      MOV PASCNTR,65
1828 011050 004537 014446              JSR R5,SGLDEC
1829 011054 000000
1830 011056 104400 015052              65:          UPEN
1831 011062 000000
1832 011064 005037 011142              HLT16:         HALT
1833 011070 042737 040500 010704  CONT22:        CLR CHARCT          ;THIS CAUSES A CRLF AFTER HALT AT E.O.P.
1834 011076 022737 000007 010420      BR 15            ;CLEAR USED BITS, AND "HAD ERROR" BIT
1835 011104 001002
1836 011106 004737 010312              CMP #7,PAT          ;IF RANDOM DATA GET NEW DATA FOR BUFFER
1837 011112 005237 007200              BNE 25
1838 011116 001375
1839 011120 013725 000042              JSR PC,GETPATTERN  ;WAIT FOR THE EOP INDICATOR TO BE PRINTED
1840 011124 001405
1841 011126 000005
1842 011130 004715              25:          INC MCNTR1          ;ACT11 END OF PASS HOOKS
1843 011132 000240
1844 011134 000240
1845 011136 000240
1846 011140 000207
1847
1848 011142 000000          LOGICAL:       JSR PC,(R5)
1849
1850
1851
1852
1853
1854
1855 011144 042737 100200 001200      HERE:          RTS PC
1856 011152 005737 001200
1857 011156 001005
1858 011160 112737 000114 001201      CHARCT:       0
1859 011166 105037 001200
1860 011172 113737 001200 011360      15:          MOV# 0D,TARGET
1861 011200 005037 011366
1862 011204 113737 001201 011366      CLR X1D
1863 011212 005037 011364
1864 011216 113737 001200 011364      MOV# 0D,X1D
1865 011224 013737 011366 011356      MOV X1D,TRKCNTR    ;SET UP NUMBER OF TRACK MOVEMENTS
1866 011232 163737 011364 011356      SUB X0D,TRKCNTR
1867 011240 005237 011356
1868 011244 052737 100200 001200      INC TRKCNTR
1869 011252 000207      B15 #100200,0D     ;SET FIRST TIME BITS IN ID,0D
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892 011254 113737 011360 011362      ;*****
1893 011262 142737 000377 011320      ;INITIALIZE TRACK SEQUENCE
1894 011270 005737 011370
1895 011274 001005
1896 011276 012737 000007 011370      INITTRACK:      B15 #100200,0D     ;CLEAR FIRST USED BITS
1897 011304 013704 011370              TST 0D              ;TEST CONTENTS OF ID,0D FOR 0
1898 011310 005304              BNE 15              ;ID,0D SPECIFIED USE THEM
1899 011312 006304              MOV# #114,ID        ;NONE SPECIFIED SET ID TO MAXIMUM
1900 011314 150437 011320      15:          CLR# 0D              ;SET 0D TO MINIMUM
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
  
```

```

1849
1850
1851
1852
1853
1854
1855 011144 042737 100200 001200      ;*****
1856 011152 005737 001200          ;INITIALIZE TRACK SEQUENCE
1857 011156 001005
1858 011160 112737 000114 001201
1859 011166 105037 001200
1860 011172 113737 001200 011360      15:          MOV# 0D,TARGET
1861 011200 005037 011366
1862 011204 113737 001201 011366      CLR X1D
1863 011212 005037 011364
1864 011216 113737 001200 011364      MOV# 0D,X1D
1865 011224 013737 011366 011356      MOV X1D,TRKCNTR    ;SET UP NUMBER OF TRACK MOVEMENTS
1866 011232 163737 011364 011356      SUB X0D,TRKCNTR
1867 011240 005237 011356
1868 011244 052737 100200 001200      INC TRKCNTR
1869 011252 000207      B15 #100200,0D     ;SET FIRST TIME BITS IN ID,0D
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892 011254 113737 011360 011362      ;*****
1893 011262 142737 000377 011320      ;TEST FOR HEAD SEQUENCE SELECTED BY INITIAL SWR SETTING
1894 011270 005737 011370
1895 011274 001005
1896 011276 012737 000007 011370
1897 011304 013704 011370
1898 011310 005304
1899 011312 006304
1900 011314 150437 011320
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
  
```

```

1905 011326 000137 011426 JMP SEQ2 ;DECREMENT
1906 011332 000137 011460 JMP SEQ3 ;INCREMENT/DECREMENT
1907 011336 000137 011524 JMP SEQ4 ;BOUNCE ID TO OD
1908 011342 000137 011610 JMP SEQ5 ;DECREASING BOUNCE
1909 011346 000137 011742 JMP SEQ6 ;STROBE
1910 011352 000137 012022 JMP SEQ7 ;RANDDM
1911
1912 011356 000000 TRKCNTR: 0
1913 011360 000000 TARGET: 0
1914 011362 000000 PRESTRK: 0
1915 011364 000000 X00: 0
1916 011366 000000 X10: 0
1917 011370 000000 SEQUEN: 0
1918
1919 ;*****
1920 ;INCREMENT FROM OD+1 TO ID AND RETURN TO OD
1921 ; S = 1
1922
1923
1924 011372 042737 100200 001200 SEQ1: BIC #100200,OD ;CLEAR FIRST TIME BITS
1925 011400 123737 011366 011362 CMPB X10,PRESTRK ;PRESENT TRACK EQUAL TO ID
1926 011406 001004 BNE IS ;NO GET NEW TRACK
1927 011410 113737 001200 011360 MOVB OD,TARGET ;YES RETURN TO OD
1928 011416 000461 BR NEWTRK ;NEWTRK INCREMENTS THE TRACK MOVED TO COUNTER
1929
1930 011420 005237 011360 IS: INC TARGET ;AND RETURNS TO NEXT TRACK
1931 011424 002456 BR NEWTRK ;ADD 1 TO TARGET TRACK
1932
1933 ;*****
1934 ;DECREMENT FROM ID TO OD
1935 ; S = 2
1936
1937
1938 011426 105737 001201 SEQ2: TSTB ID ;FIRST TIME BIT SET
1939 011432 100007 BPL IS ;NO GET NEXT TRACK
1940 011434 042737 100200 001200 BIC #100200,OD ;YES CLEAR FIRST TIME BITS
1941 011442 113737 011366 011360 MOVB X10,TARGET ;GO TO ID
1942 011450 000444 BR NEWTRK
1943 011452 005337 011360 IS: DEC TARGET ;MOVE TO NEXT TRACK
1944 011456 000441 BR NEWTRK
1945
1946 ;*****
1947 ;INCREMENT THEN DECREMENT TRACKS
1948 ; S = 3
1949
1950
1951 011460 105737 001200 SEQ3: TSTB OD ;CHECK FIRST TIME BIT
1952 011464 100007 BPL IS ;NOT FIRST TIME THROUGH
1953 011466 042737 100200 001200 BIC #100200,OD ;CLEAR FIRST TIME BITS
1954 011474 005337 011356 DEC TRKCNTR
1955 011500 006337 011356 ASL TRKCNTR ;RESET TRACK COUNTER TO DOUBLE THE COUNT
1956 011504 013700 011366 MOV X10,R0
1957 011510 163700 011564 SUB X00,R0 ;GET DIFFERENCE BETWEEN ID AND OD
1958 011514 163700 011356 SUB TRKCNTR,R0 ;IS TRACK COUNTER HALF DONE
1959 011520 002342 HGF SEQ2 ;YES GO DECREMENT TRACKS
1960 011522 000723 BR SEQ1 ;NO CONTINUE INCREMENTING TRACKS

```

```

1961
1962 ;*****
1963 ;BOUNCE BETWEEN ID AND OD ONLY
1964 ; S = 4
1965
1966
1967
1968 011524 042737 100200 001200 SEQ4: BIC #100200,OD ;CLEAR THE FIRST TIME BITS
1969 011532 123737 011362 001200 CMPB PRESTRK,OD ;ID IT JUST DO OD
1970 011540 001404 BEQ IDNEXT ;YES
1971 011542 113737 001200 011360 MOVB OD,TARGET ;NO GO TO OD TRACK
1972 011550 000404 BR NEWTRK
1973 011552 113737 001201 011360 IDNEXT: MOVB ID,TARGET ;GO DO ID TRACK
1974 011560 000400 BR NEWTRK
1975
1976 ;*****
1977 ;INCREMENT THE "HEAD MOVED TO" COUNTERS AND RETURN
1978 ;THROUGH HERE FROM ALL TRACK SEQUENCES
1979
1980
1981 011562 013705 011360 NEWTRK: MOV TARGET,R5 ;GET TRACK ADDRESS
1982 011566 006305 ASL R5 ;MULTIPLY BY 4 TO DOUBLE PRECISION
1983 011570 006305 ASL R5 ;INTERLEAVE COUNTER ADDRESSES
1984 011572 062705 017660 ADD #HMOVE,R5 ;ADD ON ADDRESS OF HEAD MOVE COUNTER
1985 011576 062715 000001 ADD #1,(R5) ;INC COUNTER
1986 011602 005565 000002 ADC 2(R5) ;ADD CARRY TO HIGH ORDER WORD
1987 011606 000207 RTS PC ;RETURN TO ACCESS NEXT TRACK
1988
1989 ;*****
1990 ;BOUNCE BETWEEN DECREASING ID AND INCREASING OD
1991 ; S = 5
1992
1993
1994
1995 011610 105737 001201 SEQ5: TSTB ID ;TEST FIRST TIME BIT OF ID
1996 011614 100011 BPL IS
1997 011616 005237 011356 INC TRKCNTR ;ADJUST FOR ONE EXTRA MOVEMENT TO TRK 06
1998 011622 142737 000200 001201 BICB #200,ID ;CLEAR FIRST TIME BIT OF ID
1999 011630 113737 011366 011360 MOVB X10,TARGET ;MOVE TO ID
2000 011636 000751 BR NEWTRK
2001 011640 105737 001200 IS: TSTB OD ;TEST FIRST TIME BIT OF OD
2002 011644 100007 BPL 2S
2003 011646 142737 000200 001200 BICB #200,OD ;CLEAR FIRST TIME BIT OF OD
2004 011654 113737 011364 011360 MOVB X00,TARGET ;MOVE TO OD
2005 011662 000737 BR NEWTRK
2006 011664 132737 000001 011356 2S: BITB #1,TRKCNTR ;TEST COUNTER FOR ODD OR EVEN
2007 011672 001006 BNE 3S ;ODD MOVE TO NEW ID
2008 011674 005237 011364 INC X00 ;EVEN ADD 1 TO OD
2009 011700 113737 011364 011360 MOVB X00,TARGET
2010 011706 000405 BR LASTTRK ;SEE IF LAST TRACK IS BEING ACCESSED
2011 011710 005337 011366 3S: DEC X10 ;SUBTRACT 1 FROM ID
2012 011714 113737 011366 011360 MOVB X10,TARGET
2013 011722 123727 011356 000001 LASTTRK: CMPB TRKCNTR,#1
2014 011730 001003 BNE XRETURN
2015 011732 113737 001200 011360 MOVB OD,TARGET ;THIS IS LAST TRACK RETURN TO OD
2016 011740 000710 BR NEWTRK

```

```

2017
2018
2019
2020
2021
2022
2023 011742 105737 001200 SEQ6: TSTB OD ;CHECK FIRST TIME BIT
2024 011746 100007 MPL IS ;NOT FIRST TIME
2025 011750 042737 100200 001200 BIC #100200,OD ;WAS FIRST TIME,CLEAR THE BITS
2026 011756 005337 011356 UEC TRKNTR ;RESET COUNTER TO DOUBLE THE NUMBER OF TRACKS
2027 011762 006337 011356 ASL TRKNTR
2028 011766 123737 011362 001200 1S: CMPB PRESTRK,OD ;WAS OD JUST ACCESSED
2029 011774 001006 BNE UDNEXT ;NO GO TO OD
2030 011776 113737 011366 011360 MOVH XID,TARGET ;DECREASING ID IS NEXT TRACK
2031 012004 005337 011366 DEC XID
2032 012010 000664 HR NEWTRK
2033 012012 113737 001200 011360 DDNEXT: MOVH OD,TARGET ;OD IS NEXT TRACK
2034 012020 000664 HR NEWTRK
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044 012022 042737 100200 001200 SEQ7: BIC #100200,OD ;CLEAR THE FIRST TIME BITS
2045 012030 004737 012122 JSR PC,RANGEN ;GET A RANDOM NUMBER
2046 012034 042737 177600 012214 BIC #177600,RANUM ;CLEAR ALL BUT LOW 7 BITS
2047 012042 123737 012214 011366 IDCOMP: CMPB RANUM,XID ;IS RANUM LARGER THAN ID ADDRESS
2048 012050 003404 BLE UDCOMP ;NO,RANUM IS LESS OR EQUAL TO ID
2049 012052 163737 011366 012214 SUB XID,RANUM ;YES,SUBTRACT ID FROM IT
2050 012060 000770 BR IDCOMP ;SEE IF RANUM IS NOW OK
2051 012062 123737 012214 011364 UDCOMP: CMPB RANUM,XOD ;IS RANUM SMALLER THAN UD ADDRESS
2052 012070 002004 HGE PRESCHK ;NO,RANUM IS GREATER OR EQUAL TO UD
2053 012072 063737 011364 012214 ADD XOD,RANUM ;YES,ADD UD TO RANUM
2054 012100 000770 BR UDCOMP ;SEE IF RANUM IS NOW OK
2055 012102 123737 012214 011362 PRESCHK: CMPB RANUM,PRESSTRK ;IF RANUM EQUALS PRESENT TRACK
2056 012110 001744 SEQ7 ;GET ANOTHER RANDOM NUMBER
2057 012112 013737 012214 011360 MOV RANUM,TARGET ;RANUM OK PUT IT IN TARGET TRACK
2058 012120 000664 HR NEWTRK
2059
2060 012122 012700 000001 RANGEN: MOV #1,R0
2061 012126 063700 012210 ADD RAN1,R0
2062 012132 063700 012212 ADD RAN2,R0
2063 012136 042700 170000 BIC #170000,R0
2064 012142 000241 CLC
2065 012144 006100 ROL R0
2066 012146 006100 ROL R0
2067 012150 012037 012210 MOV R0,RAN1
2068 012154 005000 CLR R0
2069 012156 013700 012212 MOV RAN2,R0
2070 012162 006000 ROR R0
2071 012164 006000 ROR R0
2072 012166 063700 012210 ADD RAN1,R0

```

```

2073 012172 042700 170000 BIC #170000,R0
2074 012176 010037 012212 MOV R0,RAN2
2075 012202 010037 012214 MOV R0,RANUM
2076 012206 000207 RTS PC
2077
2078 012210 000000 RAN1: 0
2079 012212 000000 RAN2: 0
2080 012214 000000 RANUM: 0
2081
2082
2083
2084
2085
2086 012216 005737 001202 INITSECTOR: TST FIRST ;TEST FIRST AND LAST FOR 0
2087 012222 001005 BNE IS ;SECTORS SPECIFIED USE THEM
2088 012224 005237 001202 JNC FIRST ;NONE SPECIFIED SET FIRST TO 1
2089 012230 112737 000032 001203 MOVB #32,LAST ;SET LAST TO MAXIMUM
2090 012236 113737 001203 012306 1S: MOVB LAST,SECCNTR ;SET UP SECTOR COUNTER
2091 012244 163737 001202 012306 SUB FIRST,SECCNTR
2092 012252 005237 012306 INC SECCNTR
2093 012256 105037 012307 CLRB SECCNTR+1
2094 012262 113737 001202 012310 MOVB FIRST,TSECTOR ;PUT FIRST SECTOR IN TARGET SECTOR
2095 012270 163737 012314 012310 SUB THREE,TSECTOR ;SUB 3 FROM TSECTOR AS FIRST TIME THROUGH
2096
2097 012276 012737 000001 012312 MOV #1,INTLEAV ;IT GETS ADDED BACK ON.
2098 012304 000207 RTS PC ;SET INTERLEAVE OFFSET
2099
2100 012306 000000 SECCNTR: 0
2101 012310 000000 TSECTOR: 0
2102 012312 000000 INTLEAV: 0
2103 012314 000003 THREE: 3
2104
2105 012316 063737 012314 012310 GETSECTOR: ADD THREE,TSECTOR ;ADD 3 FOR INTERLEAVING
2106 012324 123737 001203 012310 CMPB LAST,TSECTOR
2107 012332 002010 BGE IS ;NEW SECTOR IS WITHIN LIMITS
2108 012334 113737 001202 012310 MOVB FIRST,TSECTOR ;RESET TARGET SECTOR TO INTERLEAVE
2109 012342 063737 012312 012310 ADD INTLEAV,TSECTOR ;ADD ON INTERLEAVE OFFSET VALUE
2110 012350 005237 012312 INC INTLEAV ;UP DATE THE OFFSET VALUE
2111 012354 000207 1S: RTS PC
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
;SBTTL TYPE ROUTINE
;*****
;ROUTINE TO TYPE ASCII MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
;NOTE1: SNULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
;NOTE2: SFILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
;NOTE3: SFILLC CONTAINS THE CHARACTER TO FILL AFTER.
;#
;#CALL#
;#1) USING A THAP INSTRUCTION
;# TYPE ,MESADR ;MESADR IS FIRST ADDRESS OF AN ASCII STRING
;#OR
;# TYPE
;# MESADR

```

```

2129          1*
2130
2131 012356 105737 012605 STYPE: TSTB STPFLG      ;;IS THERE A TERMINAL?
2132 012362 100002          BPL 15          ;;BR IF YES
2133 012364 000000          HALT          ;;HALT HERE IF NO TERMINAL
2134 012366 000407          BR 35          ;;LEAVE
2135 012370 010046          15: MOV #0,-(SP)      ;;SAVE R0
2136 012372 017600 000002 MOV #2(SP),R0      ;;GET ADDRESS OF ASCII STRING
2137 012376 112046          25: MOVH (R0)+,-(SP)      ;;PUSH CHARACTER TO BE TYPED ONTO STACK
2138 012400 001005          BNE 45          ;;BR IF IT ISN'T THE TERMINATOR
2139 012402 005726          TST (SP)+          ;;IF TERMINATOR POP IT OFF THE STACK
2140 012404 012600 000002 MOV (SP)+,R0      ;;RESTORE R0
2141 012406 062716 000002 35: ADD #2,(SP)      ;;ADJUST RETURN PC
2142 012412 000002          RTI          ;;RETURN
2143 012414 122716 000011 45: CMPB #HT,(SP)      ;;BRANCH IF <HT>
2144 012420 001430          BEQ 85          ;;
2145 012422 122716 000200 CMPB #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
2146 012426 001006          BNE 55          ;;
2147 012430 005726          TST (SP)+          ;;POP <CR><LF> EQUIV
2148 012432 104400          TYPE          ;;TYPE A CR AND LF
2149 012434 012607          SCRLF
2150 012436 105037 012572 CLRB SCHARCNT      ;;CLEAR CHARACTER COUNT
2151 012442 000755          BR 25          ;;GET NEXT CHARACTER
2152 012444 004737 012526 55: JSR PC,STYPEC      ;;GO TYPE THIS CHARACTER
2153 012450 123726 012604 65: CMPB #FILLC,(SP)+  ;;IS IT TIME FOR FILLER CHARS.?
2154 012454 001350          BNE 25          ;;IF NO GO GET NEXT CHAR.
2155 012456 013746 012602 MOV #NULL,-(SP)      ;;GET # OF FILLER CHARS. NEEDED
2156
2157 012462 105366 000001 75: DECB 1(SP)          ;;DOES A NULL NEED TO BE TYPED?
2158 012466 002770          BLT 65          ;;BR IF NO--GO POP THE NULL OFF OF STACK
2159 012470 004737 012526 JSR PC,STYPEC      ;;GO TYPE A NULL
2160 012474 105337 012572 DECB SCHARCNT      ;;DO NOT COUNT AS A COUNT
2161 012500 000770          BR 75          ;;LOOP
2162
2163          ;HORIZONTAL TAB PROCESSOR
2164
2165 012502 112716 000040 85: MOVB #',(SP)      ;;REPLACE TAB WITH SPACE
2166 012506 004737 012526 95: JSR PC,STYPEC      ;;TYPE A SPACE
2167 012512 132737 000007 012572 R1TB #7,SCHARCNT      ;;BRANCH IF NOT AT
2168 012520 001372          BNE 95          ;;TAB STOP
2169 012522 005726          TST (SP)+          ;;POP SPACE OFF STACK
2170 012524 000724          BR 25          ;;GET NEXT CHARACTER
2171 012526 105777 000044 STYPEC: TSTB #STPB STYPEC      ;;WAIT UNTIL PRINTER IS READY
2172 012532 100375          BPL STYPEC
2173 012534 116677 000002 MOVB 2(SP),#STPB      ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2174 012542 122766 000015 000002 CMPB #CR,(SP)      ;;IS CHARACTER A CARRIAGE RETURN?
2175 012550 001003          BNE 15          ;;BRANCH IF NO
2176 012552 105037 012572 CLRB SCHARCNT      ;;YES--CLEAR CHARACTER COUNT
2177 012556 000406          BR STYPEX      ;;EXIT
2178 012560 122766 000012 15: CMPB #LF,(SP)      ;;IS CHARACTER A LINE FEED?
2179 012566 001402          BEQ STYPEX      ;;BRANCH IF YES
2180 012570 105227          INCB (PC)+          ;;COUNT THE CHARACTER
2181 012572 000000          SCHARCNT:=WORD 0      ;;CHARACTER COUNT STORAGE
2182 012574 000207          STYPEX: RTS PC
2183
2184 012576 177564          STPB: .WORD 177564      ;;TTY PRINTER STATUS REG. ADDRESS

```

```

2185 012600 177566 STPB: .WORD 177566      ;;TTY PRINTER BUFFER REG. ADDRESS
2186 012602 000          #NULL: .BYTE 0      ;;CONTAINS NULL CHARACTER FOR FILLS
2187 012603 002          #FILLS: .BYTE 2      ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
2188 012604 012          #FILLC: .BYTE 12      ;;INSERT FILL CHARS. AFTER A "LINE FEED"
2189 012605 000          STPFLG: .BYTE 0      ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
2190 012606 077          #QUES: .ASCII "?"      ;;QUESTION MARK
2191 012607 015          #CRLF: .ASCII <15>      ;;CARRIAGE RETURN
2192 012610 000012 #LF: .ASCII <12>      ;;LINEFEED
2193          #SBTTL BINARY TO OCTAL (ASCII) AND TYPE
2194
2195          ;*****
2196          ;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
2197          ;OCTAL (ASCII) NUMBER AND TYPE IT.
2198          ;*STYPOS--ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
2199          ;*CALL:
2200          ;* MOV NUM,-(SP)      ;;NUMBER TO BE TYPED
2201          ;* TYPOS      ;;CALL FOR TYPEOUT
2202          ;* .BYTE N          ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
2203          ;* .BYTE M          ;;M=1 OR 0
2204          ;*          ;;1=TYPE LEADING ZEROS
2205          ;*          ;;0=SUPPRESS LEADING ZEROS
2206          ;*
2207          ;*STYPON--ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
2208          ;*STYPOS OR STYPC
2209          ;*CALL:
2210          ;* MOV NUM,-(SP)      ;;NUMBER TO BE TYPED
2211          ;* TYPON      ;;CALL FOR TYPEOUT
2212          ;*
2213          ;*STYPOC--ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
2214          ;*CALL:
2215          ;* MOV NUM,-(SP)      ;;NUMBER TO BE TYPED
2216          ;* TYPOC      ;;CALL FOR TYPEOUT
2217
2218 012612 017646 000000 STYPOS: MOV #,(SP),-(SP)      ;;PICKUP THE MODE
2219 012616 116637 000001 013035 MOVB 1(SP),#0FILL      ;;LOAD ZERO FILL SWITCH
2220 012624 112637 013037 MOVB (SP)+,#0MODE+1      ;;NUMBER OF DIGITS TO TYPE
2221 012630 062716 000002 ADD #2,(SP)      ;;ADJUST RETURN ADDRESS
2222 012634 000406          BR STYPON
2223 012636 112737 000001 013035 STYPOC: MOVB #1,#0FILL      ;;SET THE ZERO FILL SWITCH
2224 012644 112737 000006 013037 MOVB #6,#0MODE+1      ;;SET FOR SIX(6) DIGITS
2225 012652 112737 000005 013034 STYPON: MOVB #5,#0CNT      ;;SET THE ITERATION COUNT
2226 012660 010346          MOV #3,-(SP)      ;;SAVE R3
2227 012662 010446          MOV #4,-(SP)      ;;SAVE R4
2228 012664 010546          MOV #5,-(SP)      ;;SAVE R5
2229 012666 113704 013037 MOVB #0MODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
2230 012672 005404          NEG R4
2231 012674 062704 000006 ADD #6,R4          ;;SUBTRACT IT FOR MAX. ALLOWED
2232 012700 110437 013036 MOVB #0MODE          ;;SAVE IT FOR USE
2233 012704 113704 013035 MOVB #0FILL,R4      ;;GET THE ZERO FILL SWITCH
2234 012710 016605 000012 MOV 12(SP),R5      ;;PICKUP THE INPUT NUMBER
2235 012714 005003          CLR R3          ;;CLEAR THE OUTPUT WORD
2236 012716 006105 15: ROL R5          ;;ROTATE MSB INTO "C"
2237 012720 000404          BR 35          ;;GO DO MSB
2238 012722 006105 25: ROL R5          ;;FORM THIS DIGIT
2239 012724 006105          ROL R5
2240 012726 006105          ROL R5

```

```

2241 012730 010503      MOV      R5,R5
2242 012732 000103      ROL      R3
2243 012734 105337 015036  35:  DECB   S0MODE  ;;GET LSB OF THIS DIGIT
2244 012740 100016      BPL      75      ;;TYPE THIS DIGIT?
2245 012742 042703 177770  8:  BIC     #177770,R3  ;;BR IF NO
2246 012746 001002      BNE     #4      ;;GET MID OF JUNK
2247 012750 005704      TST     #4      ;;TEST FOR 0
2248 012752 001403      BEQ     #5      ;;SUPPRESS THIS 0?
2249 012754 005204      INC     #4      ;;BR IF YES
2250 012756 052703 000060  45:  BIC     #'0,R3  ;;DON'T SUPPRESS ANYMORE 0'S
2251 012762 052703 000040  56:  BIC     #' ,R3  ;;MAKE THIS DIGIT ASCII
2252 012766 110337 013032      MOV     R3,#8    ;;MAKE ASCII IF NOT ALREADY
2253 012772 104400 013032      TYPE   #8      ;;SAVE FOR TYPING
2254 012776 105337 013034  75:  DECB   S0CNT    ;;GO TYPE THIS DIGIT
2255 013002 003347      SGT     #1      ;;COUNT BY 1
2256 013004 002402      BLT     #5      ;;BR IF MORE TO DO
2257 013006 005204      INC     #4      ;;BR IF DONE
2258 013010 000744      BR      #23     ;;INSURE LAST DIGIT ISN'T A BLANK
2259 013012 012605      MOV     (SP)+,R5  ;;GO DO THE LAST DIGIT
2260 013014 012604      MOV     (SP)+,R4  ;;RESTORE R5
2261 013016 012603      MOV     (SP)+,R3  ;;RESTORE R4
2262 013020 016666 000002 000004  MOV     2(SP),4(SP)  ;;RESTORE R3
2263 013026 012616      MOV     (SP)+,(SP)  ;;SET THE STACK FOR RETURNING
2264 013030 000002      RTI
2265 013032 0000      RTI      ;;RETURN
2266 013033 0000      .BYTE 0    ;;STORAGE FOR ASCII DIGIT
2267 013034 0000      S0CNT: .BYTE 0  ;;TERMINATOR FOR TYPE ROUTINE
2268 013035 0000      S0FILL: .BYTE 0  ;;OCTAL DIGIT COUNTER
2269 013036 000000      S0MODE: .WORD 0  ;;ZERO FILL SWITCH
2270      .SBTTL SAVE AND RESTORE R0-R5 ROUTINES
2271
2272      ;;*****
2273      ;;SAVE R0-R5
2274      ;;CALL:
2275      ;; SAVREG
2276      ;;UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
2277      ;;
2278      ;;TOP---(+16)
2279      ;; +2---(+16)
2280      ;; +4---R5
2281      ;; +6---R4
2282      ;; +8---R3
2283      ;;+10---R2
2284      ;;+12---R1
2285      ;;+14---R0
2286
2287      $SAVREG:
2288      MOV     R0,=(SP)  ;;PUSH R0 ON STACK
2289      MOV     R1,=(SP)  ;;PUSH R1 ON STACK
2290      MOV     R2,=(SP)  ;;PUSH R2 ON STACK
2291      MOV     R3,=(SP)  ;;PUSH R3 ON STACK
2292      MOV     R4,=(SP)  ;;PUSH R4 ON STACK
2293      MOV     R5,=(SP)  ;;PUSH R5 ON STACK
2294      MOV     22(SP),-(SP)  ;;SAVE PS OF MAIN FLOW
2295      MOV     22(SP),-(SP)  ;;SAVE PC OF MAIN FLOW
2296      MOV     22(SP),-(SP)  ;;SAVE PS OF CALL
  
```

```

2297 013070 016666 000022      MOV     22(SP),-(SP)  ;;SAVE PC OF CALL
2298 013074 200002      RTI
2299
2300      ;;RESTORE R0-R5
2301      ;;CALL:
2302      ;; RESWEG
2303      $RESREG:
2304      MOV     (SP)+,22(SP)  ;;RESTORE PC OF CALL
2305      MOV     (SP)+,22(SP)  ;;RESTORE PS OF CALL
2306      MOV     (SP)+,22(SP)  ;;RESTORE PC OF MAIN FLOW
2307      MOV     (SP)+,22(SP)  ;;RESTORE PS OF MAIN FLOW
2308      MOV     (SP)+,R5
2309      MOV     (SP)+,R4
2310      MOV     (SP)+,R3
2311      MOV     (SP)+,R2
2312      MOV     (SP)+,R1
2313      MOV     (SP)+,R0
2314      RTI
2315      .SBTTL TTY INPUT ROUTINE
2316
2317      ;;*****
2318      STKS: .WORD 177560  ;;TTY KBD STATUS
2319      STKB: .WORD 177562  ;;TTY KBD BUFFER
2320      .ENABL LSB
2321
2322      ;;*****
2323      ;;SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
2324      ;;ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
2325      ;;SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
2326      ;;WHEN OPERATING IN TTY FLAG MODE.
2327      SCKSWR: CMP     #SWREG,SWR  ;;IS THE SOFT-SWR SELECTED?
2328      BNE     #155  ;;BRANCH IF NO
2329      TSTB   #STKS  ;;CHAR THERE?
2330      BPL     #155  ;;IF NO, DON'T WAIT AROUND
2331      MOV     #STKB,=(SP)  ;;SAVE THE CHAR
2332      BIC     #'C177,(SP)  ;;STRIP-OFF THE ASCII
2333      CMP     #7,(SP)+  ;;IS IT A CONTROL G?
2334      BNE     #155  ;;NO, RETURN TO USER
2335      CMPEB #SAUTOB,#1  ;;ARE WE RUNNING IN AUTO-MODE?
2336      BEQ     #155  ;;BRANCH IF YES
2337
2338      TYPE   #SCNTLG  ;;ECHO THE CONTROL-G (^G)
2339      SGTSWR: TYPE   #MSWR  ;;TYPE CURRENT CONTENTS
2340      MOV     SWREG,=(SP)  ;;SAVE SWREG FOR TYPEOUT
2341      TYPOC  ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2342      TYPE   #MNEW  ;;PROMPT FOR NEW SWR
2343      CLR    -(SP)  ;;CLEAR COUNTER
2344      CLR    -(SP)  ;;THE NEW SWR
2345      TSTB   #STKS  ;;CHAR THERE?
2346      BPL     #75  ;;IF NOT TRY AGAIN
2347
2348      MOV     #STKB,=(SP)  ;;PICK UP CHAR
2349      BIC     #'C177,(SP)  ;;MAKE IT 7-BIT ASCII
2350
2351
2352
  
```

```

2353 013250 021627 000025 95: CMP (SP),#25 //IS IT A CONTROL-U?
2354 013254 001005 BNE 105 //BRANCH IF NOT
2355 013256 104400 015660 TYPE ,SCNTLU //YES, ECHO CONTROL-U ("U")
2356 013262 062706 000006 205: ADD #6,SP //IGNORE PREVIOUS INPUT
2357 013266 000757 BR 195 //LET'S TRY IT AGAIN
2358
2359
2360 013270 021627 000015 105: CMP (SP),#15 //IS IT A <CR>?
2361 013274 001022 BNE 105 //BRANCH IF NO
2362 013276 005766 000004 TST 4(SP) //YES, IS IT THE FIRST CHAR?
2363 013302 001423 BEQ 115 //BRANCH IF YES
2364 013304 016677 000002 165702 MOV 2(SP),#SWR //SAVE NEW SWR
2365 013312 062706 000006 115: ADD #6,SP //CLEAR UP STACK
2366 013316 104400 012607 145: TYPE ,SCRLF //ECHO <CR> AND <LF>
2367 013322 123727 015715 000001 CMPB $INTAG,#1 //RE-ENABLE TTY KBD INTERRUPTS?
2368 013330 001005 BNE 155 //BRANCH IF NOT
2369 013332 012777 000100 177574 MOV #100,0$TKS //RE-ENABLE TTY KBD INTERRUPTS
2370 013340 000002 155: RTI //RETURN
2371 013342 004737 012526 165: JSR PC,$TYPEC //ECHO CHAR
2372 013346 021627 000060 CMP (SP),#0 //CHAR < 0?
2373 013352 002420 BLT 105 //BRANCH IF YES
2374 013354 021627 000067 CMP (SP),#07 //CHAR > 7?
2375 013360 003015 BGT 105 //BRANCH IF YES
2376 013362 042726 000060 BIC #60,(SP)+ //STRIP-OFF ASCII
2377 013366 005766 000002 TST 2(SP) //IS THIS THE FIRST CHAR
2378 013372 001403 BEQ 175 //BRANCH IF YES
2379 013374 006316 ASL (SP) //NO, SHIFT PRESENT
2380 013376 006316 ASL (SP) // CHAR OVER TO MAKE
2381 013400 006316 ASL (SP) // ROOM FOR NEW ONE.
2382 013402 005266 000002 175: INC 2(SP) //KEEP COUNT OF CHAR
2383 013406 056616 177776 BIS -2(SP),(SP) //SET IN NEW CHAR
2384 013412 000707 BR 75 //GET THE NEXT ONE
2385 013414 104400 012606 185: TYPE ,SQUES //TYPE ?<CR><LF>
2386 013420 000720 BR 205 //SIMULATE CONTROL-U
2387
2388
2389
2390
2391 //*****
2392 //THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
2393 //CALL:
2394 // RDCHR //INPUT A SINGLE CHARACTER FROM THE TTY
2395 // RETURN HERE //CHARACTER IS ON THE STACK
2396 // //WITH PARITY BIT STRIPPED OFF
2397 //
2398 013422 011646 SRDCHR: MOV (SP),-(SP) //PUSH DOWN THE PC
2399 013424 016666 000004 000002 MOV 4(SP),2(SP) //SAVE THE PS
2400 013432 105777 177476 15: TSTB 0$TKS //WAIT FOR
2401 013436 10375 BPL 15 //A CHARACTER
2402 013440 117766 177472 000004 MOVB 0$TKB,4(SP) //READ THE TTY
2403 013446 042766 177600 000004 BIC #~C177,4(SP) //GET RID OF JUNK IF ANY
2404 013454 026627 000004 000023 CMP 4(SP),#23 //IS IT A CONTROL-S?
2405 013462 001013 BNE 55 //BRANCH IF NO
2406 013464 105777 177444 25: TSTB 0$TKS //WAIT FOR A CHARACTER
2407 013470 10375 BPL 25 //LOOP UNTIL ITS THERE
2408 013472 117766 177440 MOVB 0$TKB,-(SP) //GET CHARACTER

```

```

2409 013476 042716 177600 BIC #~C177,(SP) //MAKE IT 7-BIT ASCII
2410 013502 022627 000021 CMP (SP)+,#21 //IS IT A CONTROL-Q?
2411 013506 001366 BNE 25 //IF NOT DISCARD IT
2412 013510 000750 BR 15 //YES, RESUME
2413 013512 026627 000004 000140 35: CMP 4(SP),#140 //IS IT UPPER CASE?
2414 013520 002407 BLT 45 //BRANCH IF YES
2415 013522 026627 000004 000175 35: CMP 4(SP),#175 //IS IT A SPECIAL CHAR?
2416 013530 003003 BGT 45 //BRANCH IF YES
2417 013532 042766 000040 000004 45: BIC #40,4(SP) //MAKE IT UPPER CASE
2418 013540 000002 RTI //GO BACK TO USER
2419
2420 //*****
2421 //THIS ROUTINE WILL INPUT A STRING FROM THE TTY
2422 //CALL:
2423 // RDLIN //INPUT A STRING FROM THE TTY
2424 // RETURN HERE //ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
2425 // //TERMINATOR WILL BE A BYTE OF ALL 0'S
2426 013542 010346 SRDLIN: MOV R3,-(SP) //SAVE R3
2427 013544 012703 013650 15: MOV $TTYIN,R3 //GET ADDRESS
2428 013550 022703 013660 25: CMP $TTYIN+8,R3 //BUFFER FULL?
2429 013554 101405 BLOS 45 //BR IF YES
2430 013556 104406 RDCHR //GO READ ONE CHARACTER FROM THE TTY
2431 013560 112613 MOVB (SP)+,(R3) //GET CHARACTER
2432 013562 122713 000177 105: CMPB #177,(R3) //IS IT A RUBOUT
2433 013566 001003 BNE 35 //SKIP IF NOT
2434 013570 104400 012606 45: TYPE ,SQUES //TYPE A "?"
2435 013574 000763 BR 15 //CLEAR THE BUFFER AND LOOP
2436 013576 111337 013646 35: MOVB (R3),95 //ECHO THE CHARACTER
2437 013602 104400 013646 TYPE ,95
2438 013606 122723 000015 CMPB #15,(R3)+ //CHECK FOR RETURN
2439 013612 001356 BNE 25 //LOOP IF NOT RETURN
2440 013614 105063 177777 CLRB -1(R3) //CLEAR RETURN (THE 15)
2441 013620 104400 012610 TYPE ,3LF //TYPE A LINE FEED
2442 013624 012603 MOV (SP)+,R3 //RESTORE R3
2443 013626 011646 MOV (SP),-(SP) //ADJUST THE STACK AND PUT ADDRESS OF THE
2444 013630 016666 000004 000002 MOV 4(SP),2(SP) // FIRST ASCII CHARACTER ON IT
2445 013636 012766 013650 000004 MOV $TTYIN,4(SP)
2446 013644 000002 RTI //RETURN
2447 013646 000 95: .BYTE 0 //STORAGE FOR ASCII CHAR. TO TYPE
2448 013647 000 .BYTE 0 //TERMINATOR
2449 013650 000010 $TTYIN: .BLKB 8 //RESERVE 8 BYTES FOR TTY INPUT
2450 013660 052536 005015 000 SCNTLU: .ASCIZ /"U<15><12> //CONTROL "U"
2451 013665 136 006507 000012 SCNTLG: .ASCIZ /"G<15><12> //CONTROL "G"
2452 013672 005015 053523 020122 $MSWR: .ASCIZ <15><12>/$WR = /
2453 013700 020075 000
2454 013703 040 047040 053505 $MNEW: .ASCIZ / NEW = /
2455 013710 036440 000040
2456 013714 000 $AUTOB: .BYTE 0 //AUTO MODE FLAG
2457 013715 000 $INTAG: .BYTE 0 //INTERRUPT MODE FLAG
2458
2459
2460 //*****
2461 //THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
2462 //AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
2463 //OF THE DESIRED ROUTINE, THEN USING THE ADDRESS OBTAINED IT WILL
2464 //GO TO THAT ROUTINE.

```

```

2465
2466 013716 010046 STRAP: MOV R0,=(SP) ;;SAVE R0
2467 013720 016600 000002 MOV 2(SP),R0 ;;GET TRAP ADDRESS
2468 013724 005740 TST -(R0) ;;BACKUP BY 2
2469 013726 111000 MOV# (R0),R0 ;;GET RIGHT BYTE OF TRAP
2470 013730 006300 ASL R0 ;;POSITION FOR INDEXING
2471 013732 016000 015740 MOV STRPAD(R0),R0 ;;INDEX TO TABLE
2472 013736 000200 RTS R0 ;;GO TO ROUTINE
2473
2474 .SBTTL TRAP TABLE
2475
2476 ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
2477 ;*BY THE "TRAP" INSTRUCTION.
2478
2479 ;
2480 ; ROUTINE
2481 ; -----
2482 013740 STRPAD:
2483 TYPE ;;CALL=TYPE TRAP+0(104400) TTY TYPEDOUT ROUTINE
2484 STYPOC ;;CALL=STYPOC TRAP+1(104401) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
2485 STYPOS ;;CALL=STYPOS TRAP+2(104402) TYPE OCTAL NUMBER (NO LEADING ZEROS)
2486 STYPON ;;CALL=STYPON TRAP+3(104403) TYPE OCTAL NUMBER (AS PER LAST CALL)
2487
2488 013750 SGTSWR ;;CALL=SGTSWR TRAP+4(104404) GET SOFT-SWR SETTING
2489
2490 013752 SCKSWR ;;CALL=SCKSWR TRAP+5(104405) TEST FOR CHANGE IN SOFT-SWR
2491 013754 SKDCMR ;;CALL=SKDCMR TRAP+6(104406) TTY TYPEIN CHARACTER ROUTINE
2492 013756 SKDLIN ;;CALL=SKDLIN TRAP+7(104407) TTY TYPEIN STRING ROUTINE
2493 013760 SSAVEG ;;CALL=SSAVEG TRAP+10(104410) SAVE R0-R5 ROUTINE
2494 013762 SRESREG ;;CALL=SRESREG TRAP+11(104411) RESTORE R0-R5 ROUTINE
2495
2496 .SBTTL POWER DOWN AND UP ROUTINES
2497
2498 ;*****
2499 ;POWER DOWN ROUTINE
2500 SPWRDN: MOV #SILLUP,#PWRVEC ;;SET FOR FAST UP
2501 MOV #300,#PWRVEC+2 ;;PRI0:7
2502 MOV R0,=(SP) ;;PUSH R0 ON STACK
2503 MOV R1,=(SP) ;;PUSH R1 ON STACK
2504 MOV R2,=(SP) ;;PUSH R2 ON STACK
2505 MOV R3,=(SP) ;;PUSH R3 ON STACK
2506 MOV R4,=(SP) ;;PUSH R4 ON STACK
2507 MOV R5,=(SP) ;;PUSH R5 ON STACK
2508 MOV #SWR,=(SP) ;;PUSH #SWR ON STACK
2509 SP,SSAVR6 ;;SAVE SP
2510 MOV #SPWRUP,#PWRVEC ;;SET UP VECTOR
2511 HALT
2512 BR =-2 ;;HANG UP
2513
2514 ;*****
2515 ;POWER UP ROUTINE
2516 SPWRUP: MOV #SILLUP,#PWRVEC ;;SET FOR FAST DOWN
2517 MOV SSAVEG,SP ;;GET SP
2518 CLR SSAVEG ;;WAIT LOOP FOR THE TTY
2519 INC SSAVEG ;;WAIT FOR THE INC
2520 BNE 15 ;;OF WORD
2521 MOV (SP)+,#SWR ;;POP STACK INTO #SWR
2522 MOV (SP)+,R5 ;;POP STACK INTO R5

```

```

2521 014070 012604 MOV (SP)+,R4 ;;POP STACK INTO R4
2522 014072 012603 MOV (SP)+,R3 ;;POP STACK INTO R3
2523 014074 012602 MOV (SP)+,R2 ;;POP STACK INTO R2
2524 014076 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
2525 014100 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
2526 014102 012737 015764 000024 MOV #SPWRDN,#PWRVEC ;;SET UP THE POWER DOWN VECTOR
2527 014110 012737 000340 000026 MOV #300,#PWRVEC+2 ;;PRI0:7
2528 014116 104400 TYPE ;;REPORT THE POWER FAILURE
2529 014120 014136 SPWRMG: ,WORD SPOWER ;;POWER FAIL MESSAGE POINTER
2530 014122 012716 MOV (PC)+,(SP) ;;RESTART AT RESTART
2531 014124 002700 SPWRAD: ,WORD RESTART ;;RESTART ADDRESS
2532 014126 000002 RTI
2533 014130 000000 SILLUP: HALT ;;THE POWER UP SEQUENCE WAS STARTED
2534 014132 000776 BR =-2 ;; BEFORE THE POWER DOWN WAS COMPLETE
2535 014134 000000 SSAVEG: 0 ;;PUT THE SP HERE
2536 014136 005015 047520 042527 SPOWER: ,ASCIZ <15><12>"POWER"
2537 014144 000122 ,EVEN
2538 .SBTTL SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE
2539
2540 ;*****
2541 ;*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
2542 ;*UNSIGNED DECIMAL ASCIZ NUMBER.
2543 ;*CALL
2544 ;* MOV NUMBER,=(SP) ;;PUT BINARY NUMBER ON THE STACK
2545 ;* JSR PC,#SUB2D ;;CALL
2546 ;* RETURN ;;ADDRESS OF THE 1ST ASCIZ CHAR,IS ON THE STACK
2547
2548
2549
2550 014146 016637 000002 014176 SUB2D: MOV 2(SP),15 ;;SAVE BINARY NUMBER
2551 014154 012746 014176 MOV #15,=(SP) ;;SET POINTER
2552 014160 004737 014202 JSR PC,#SUB2D ;;CALL DOUBLE LENGTH CONVERT
2553 014164 062716 000005 ADD #5,(SP) ;;ONLY ALLOW FIVE CHARACTERS
2554 014170 012666 000002 MOV (SP)+,2(SP) ;;PICKUP POINTER
2555 014174 000207 RTS PC ;;RETURN
2556 014176 000000 000000 15: ,WORD 0,0
2557 .SBTTL DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
2558
2559 ;*****
2560 ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
2561 ;*DECIMAL (ASCII) NUMBER, THE SIGN OF THE BINARY NUMBER MUST BE
2562 ;*POSITIVE.
2563 ;*CALL
2564 ;* MOV #PNTR,=(SP) ;;POINTER TO LOW WORD OF BINARY NUMBER
2565 ;* JSR PC,#SUB2D ;;CALL
2566 ;* RETURN ;;THE FIRST ADDRESS OF ASCII
2567 ;* IS ON THE STACK
2568
2569
2570 014202 104410 SUB2D: SAVREG ;;SAVE REGISTERS
2571 014204 016602 000002 MOV 2(SP),R2 ;;PICKUP THE DATA POINTER
2572 014210 012700 014362 MOV #SDECVL,R0 ;;GET ADDRESS OF "SDECVL" STRING
2573 014214 010066 000002 MOV R0,2(SP) ;;PUT ADDRESS OF ASCIZ STRING ON STACK
2574 014220 012201 MOV (R2)+,R1 ;;PICKUP THE BINARY NUMBER
2575 014222 012202 MOV (R2)+,R2
2576 014224 012737 000012 014300 MOV #10,,45 ;;SET UP TO DO 10 CONVERSIONS

```



```

2577 014232 012704 014312      MOV      #STNPNR,R4      ;ADDRESS OF TEN POWER
2578 014236 012705 014314      MOV      #STNPNR+2,R5
2579 014242 005003      15: CLR      R3          ;CLEAR PARTIAL
2580 014244 161401      23: SUB      (R4),R1     ;SUBTRACT TEN POWER
2581 014246 005602      SBC      R2
2582 014250 161502      SUB      (R5),R2
2583 014252 002402      BLT     35             ;JBR IF TEN POWER TO LARGE
2584 014254 005203      INC     M3            ;JADD 1 TO PARTIAL
2585 014256 000772      BR      25           ;JLOOP
2586 014260 062401      35: ADD      (R4)+,R1    ;RESTORE SUBTRACTED VALUE
2587 014262 005502      ADC     R2
2588 014264 062402      ADD      (R4)+,R2
2589 014266 022525      CMP     (R5)+,(R5)+   ;MOVE TO NEXT TEN POWER
2590 014270 052703 000060      BIS     #'0,M5        ;CHANGE PARTIAL TO ASCII
2591 014274 110320      MOVB   R3,(M0)+      ;SAVE IT
2592 014276 005327      DEC     (PC)+        ;IDONE?
2593 014300 000000      45: .WORD   0
2594 014302 001357      RNE     13           ;JBR IF NO
2595 014304 105020      CLRB   (M0)+        ;TERMINATOR
2596 014306 104411      RESREG                    ;RESTORE REGISTERS
2597 014310 000207      RTS     PC          ;RETURN
2598 014312 145000      STNPNR: 145000      ;1.0E09
2599 014314 035632      35632
2600 014316 160400      160400      ;1.0E08
2601 014320 002765      2765
2602 014322 113200      113200      ;1.0E07
2603 014324 000230      230
2604 014326 041100      041100      ;1.0E06
2605 014330 000017      17
2606 014332 103240      103240      ;1.0E05
2607 014334 000001      1
2608 014336 023420      23420       ;1.0E04
2609 014340 000000      0
2610 014342 001750      1750        ;1.0E03
2611 014344 000000      0
2612 014346 000144      144         ;1.0E02
2613 014350 000000      0
2614 014352 000012      12          ;1.0E01
2615 014354 000000      0
2616 014356 000001      1           ;1.0E00
2617 014360 000000      0
2618 014362 000014      SUBCVL: .BLKB 12.    ;RESERVE STORAGE FOR ASCII STRING
2619
2620 ;*****
2621 ;
2622 ;TYPE NUMERICAL ASCII STRING,RIGHT JUSTIFIED
2623 ;REPLACING LEADING ZEROS WITH SPACES.
2624 ;
2625 ;FIRST ADDRESS OF ASCII STRING MUST BE ON TOP OF THE STACK
2626
2627 014376 010046      RTJUST: MOV R0,-(SP)    ;SAVE R0
2628 014400 016600 000004      MOV 4(SP),R0      ;PICK UP ADDRESS OF ASCII STRING
2629 014404 010037 014436      MOV R0,35         ;SAVE ADDRESS FOR TYPE OUT
2630 014410 105710      15: TST  (R0)      ;IS THIS THE TERMINATOR
2631 014412 001406      BEQ  25           ;IF YES TYPE IT OUT
2632 014414 122710 000060      CMPB #'0,(R0)    ;IS IT A ZERO
  
```

```

2633 014420 001005      RNE  45             ;IF NO GO PRINT IT
2634 014422 112720 000040      MOVB #'',(R0)+    ;IF YES REPLACE IT WITH A SPACE
2635 014426 000770      BR  13             ;TEST NEXT CHAR.
2636 014430 112740 000060      25: MOVB #'0,-(R0)  ;STRING OFF ALL ZEROS,PUT BACK THE LAST ONE
2637 014434 104400      45: TYPE                    ;TYPE THE STRING
2638 014436 000000      35: OPEN
2639 014440 012600      MOV (SP)+,R0      ;RESTORE R0
2640 014442 012616      MOV (SP)+,(SP)   ;RESTORE THE STACK
2641 014444 000207      RTS PC          ;RETURN
2642
2643 ;TYPES 16 BIT WORD IN DECIMAL
2644
2645 014446 012546      SGLDEC: MOV (R5)+,-(SP) ;PUT NUMBER TO BE TYPED ON STACK
2646 014450 004737 014146      JSR PC,#$$SH2D   ;CONVERT NUMBER TO DECIMAL
2647 014454 004737 014376      JSR PC,RTJUST    ;TYPE THE DECIMAL NUMBER
2648 014460 000205      RTS R5
2649
2650
2651 014462 047125 054105 042520      MUXDD: .ASCII "UNEXPECTED D U MARK"
2652 014470 052103 042105 042040
2653 014476 042040 046440 051101
2654 014504 000113
2655
2656 014506 020104 020104 040515      MDDMIS: .ASCII "D U MARK MISSING"
2657 014514 045522 046440 051511
2658 014522 044523 043516 000
2659
2660
2661 014527 104 052101 026101      MDERHDR: .ASCII "DATA, NO STATUS ERROR"
2662 014534 047040 020117 052123
2663 014542 052101 051525 042440
2664 014550 051122 051117 000
2665
2666 014555 040 047117 052040      MTRK: .ASCII " ON TRACK"
2667 014562 040522 045503 000
2668
2669
2670 014567 040 043040 047522      MPRES: .ASCII " FROM TRACK"
2671 014574 020115 051124 041501
2672 014602 000113
2673
2674 014604 027440 051440 041505      MSECT: .ASCII " / SECTOR"
2675 014612 047524 000122
2676
2677 014616 005015 041040 052131      MCOLMUN: .ASCII "<15><12>" BYTE BAD GOOD"<15><12>"
2678 014624 020105 041040 042101
2679 014632 020040 047507 042117
2680 014640 005015 000
2681
2682 014643 015 041412 052501      00LOAD: .ASCII "<15><12>"CAUTION - IF YOU DESIRE TO TEST UNIT 0"
2683 014650 044524 047117 026440
2684 014656 044440 020106 047531
2685 014664 020125 042504 044523
2686 014672 042522 052040 020117
2687 014700 042524 052123 052440
2688 014706 044516 020124 060
  
```

2689	014713	015	051012	050105		.ASCII <15><12>"REPLACE LOAD MEDIUM WITH A SCRATCH DISKETTE"
2690	014720	040514	042503	046040		
2691	014726	040517	020104	042515		
2692	014734	044504	046525	053440		
2693	014742	052111	020110	020101		
2694	014750	041523	040522	041524		
2695	014756	020110	044504	045523		
2696	014764	052105	042524			
2697	014770	040505	044124	047105		.ASCII <15><12>"THEN PRESS CONTINUE"
2698	014776	050040	042522	051523		
2699	015004	041440	047117	044524		
2700	015012	052516	000105			
2701						
2702	015016	047516	042040	044522	MNDRY:	.ASCII "NO DRIVES READY"<15><12>
2703	015024	042526	020123	042522		
2704	015032	042101	006531	000012		
2705						
2706	015040	020040	020040	020040	TAB:	.ASCII <40><40><40><40><40><40>
2707	015046	000				
2708						
2709	015047	040	040040		DBLSP:	.ASCII <40><40>
2710						
2711	015052	005015	000		MCRLF:	.ASCII <15><12>
2712						
2713	015055	104	000		MEOP:	.ASCII "0"
2714						
2715	015057	055	000		MEREOP:	.ASCII "-"
2716						
2717	015061	007	000		MAHELL:	.ASCII <07>
2718						
2719	015063	105	051122	051117	MERHEADER:	.ASCII "ERROR CONDITIONS "
2720	015070	041440	047117	044504		
2721	015076	044524	047117	020123		
2722	015104	000040				
2723						
2724	015106	047125	052111	030040	MUNIT0:	.ASCII "UNIT 0 "
2725	015114	000040				
2726						
2727	015116	047125	052111	030440	MUNIT1:	.ASCII "UNIT 1 "
2728	015124	000040				
2729						
2730	015126	040510	020123	042502	MDELET:	.ASCII "HAS BEEN DELETED."
2731	015134	047105	042040	046105		
2732	015142	052105	042105	000056		
2733						
2734	015150	044440	052110	053040	MINTVEC:	.ASCII "INT VECTOR = "
2735	015156	041505	047524	020122		
2736	015164	020075	000			
2737						
2738	015167	122	041530	020123	MRXCS:	.ASCII "RXCS = "
2739	015174	020075	000			
2740						
2741	015177	040	054122	041104	MRXOR:	.ASCII "RXOR = "
2742	015204	036440	000040			
2743						
2744	015210	052123	052101	051525	MASTAT:	.ASCII "STATUS A = "

2745	015216	040440	036440	000040		
2746						
2747	015224	052123	052101	051525	MSTAT:	.ASCII "STATUS B = "
2748	015232	041040	036440	000040		
2749						
2750	015240	005015	000012		DBLLF:	.ASCII <15><12><12>
2751						
2752	015244	036520	000040		MPAT:	.ASCII "P = "
2753						
2754	015250	036524	000040		MTEST:	.ASCII "T = "
2755						
2756	015254	036523	000040		MSEQ:	.ASCII "S = "
2757						
2758	015260	047516	044440	052116	MINTER:	.ASCII "NO INTERRUPT AT DONE ERROR"
2759	015266	051105	050125	020124		
2760	015274	052101	042040	047117		
2761	015302	020105	051105	047522		
2762	015310	000122				
2763						
2764	015312	047125	047113	053517	MUNKINT:	.ASCII "UNKNOWN INTERRUPT"
2765	015320	020116	047111	042524		
2766	015326	052522	052120	000		
2767						
2768	015333	124	052117	046101	MERCT:	.ASCII "TOTAL READ CHECK ERRORS = "
2769	015340	051040	040505	020104		
2770	015346	044103	041505	020113		
2771	015354	051105	047522	051522		
2772	015362	036440	000040			
2773						
2774	015366	044506	046114	052502	MFIL:	.ASCII "FILLBUFFER "
2775	015374	043106	051105	000040		
2776						
2777	015402	046505	052120	041131	MEMPTY:	.ASCII "EMPTYBUFFER "
2778	015410	043125	042506	020122		
2779	015416	000				
2780						
2781	015417	104	044522	042526	MICON:	.ASCII "DRIVE(S) "
2782	015424	051450	020051	000040		
2783						
2784	015432	042524	052123	044040	MHUNG:	.ASCII "TEST HUNG"<15><12>
2785	015440	047125	006507	000012		
2786						
2787	015446	047516	051516	040524	MNONSTD:	.ASCII "NONSTANDARD TRACK / SECTOR LIMITS"
2788	015454	042116	051101	020104		
2789	015462	051124	041501	020113		
2790	015470	020057	042523	052103		
2791	015476	051117	046040	046511		
2792	015504	052111	000123			
2793						
2794	015510	042117	000075		MOD:	.ASCII "OD="
2795						
2796	015514	042111	000075		MID:	.ASCII "ID="
2797						
2798	015520	044506	051522	036524	MFIRST:	.ASCII "FIRST="
2799	015526	000				
2800						

2801	015527	114	051501	036524	MLAST:	.ASCIZ "LAST="
2802	015534	000				
2803						
2804	015535	111	044516	027124	MINIT1:	.ASCIZ "INIT,DONE NOT SET ERROR"<15><12>
2805	015542	047504	042516	047040		
2806	015550	052117	051440	052105		
2807	015556	042440	051122	051117		
2808	015564	005015	000			
2809						
2810	015567	111	044516	027124	MINIT2:	.ASCIZ "INIT,DONE SET ERROR" <15><12>
2811	015574	047504	042516	051440		
2812	015602	052105	042440	051122		
2813	015610	051117	005015	000		
2814						
2815	015615	104	042040	042440	MODER:	.ASCIZ "D D ERROR"
2816	015622	051122	051117	000		
2817						
2818	015627	122	041505	053117	MREC:	.ASCIZ "RECOVERABLE "
2819	015634	051105	041101	042514		
2820	015642	000040				
2821						
2822	015644	051103	020103	051105	MBADCR:	.ASCIZ "CRC ERROR NO DATA ERROR"
2823	015652	047522	020122	047516		
2824	015660	042040	052101	020101		
2825	015666	051105	047522	000122		
2826						
2827	015674	042522	042101	000040	MREAD:	.ASCIZ "READ "
2828						
2829	015702	040510	052114	032040	MHALT4:	.ASCIZ "HALT 4" <15><12>
2830	015710	005015	000			
2831						
2832	015713	110	046101	020124	MHALT3:	.ASCIZ "HALT 3" <15><12>
2833	015720	006463	000012			
2834						
2835	015724	040510	052114	030440	MHALT11:	.ASCIZ "HALT 11" <15><12>
2836	015732	006461	000012			
2837						
2838	015736	040504	040524	041440	MCRC:	.ASCIZ "DATA CRC ERROR"
2839	015744	041522	042440	051122		
2840	015752	051117	000			
2841						
2842						
2843	015755	040	047125	042522	MUNREC:	.ASCIZ " UNRECOVERABLE "
2844	015762	047503	042526	040522		
2845	015770	046102	020105	000		
2846						
2847	015775	123	042505	020113	MSEEK:	.ASCIZ "SEEK ERROR"
2848	016002	051105	047522	000122		
2849						
2850	016010	051127	052111	020105	MWRITE:	.ASCIZ "WRITE "
2851	016016	000				
2852						
2853	016017	120	051101	052111	MPAR:	.ASCIZ "PARITY ERROR"
2854	016024	020131	051105	047522		
2855	016032	000122				
2856						

2857	016034	051105	047522	020122	MNOFLAG:	.ASCIZ "ERROR FLAG ERROR"
2858	016042	046106	043501	042440		
2859	016050	051122	051117	000		
2860						
2861	016055	122	030530	020061	MDUMP:	.ASCIZ "RX11 DUMP"
2862	016062	052504	050115	000		
2863						
2864	016067	105	051122	051117	MEHS:	.ASCIZ "ERRORS"
2865	016074	000123				
2866						
2867	016076	044123	051117	020124	MSHORT:	.ASCIZ "SHORT DUMP"
2868	016104	052504	050115	000		
2869						
2870	016111	122	051505	040524	MRESTART:	.ASCIZ "RESTARTS = "
2871	016116	052122	020123	020075		
2872	016124	000				
2873						
2874	016125	120	051501	020123	MNOPAS:	.ASCIZ "PASS ABORTED"
2875	016132	041101	051117	042524		
2876	016140	000104				
2877						
2878	016142	040520	051523	051505	MPASS:	.ASCIZ "PASSES = "
2879	016150	036440	000040			
2880						
2881	016154	051127	052111	042524	MWRTE:	.ASCIZ "WRITTEN "
2882	016162	020116	000			
2883						
2884	016165	057	000040		MSLASH:	.ASCIZ "/"
2885						
2886	016170	000040			SPACE:	.ASCIZ "<40>
2887						
2888	016172	051105	047522	051522	MCODES:	.ASCIZ "ERRORS PER ERROR CODES"<15><12>
2889	016200	050040	051105	042440		
2890	016206	051122	051117	041440		
2891	016214	042117	051505	005015		
2892	016222	000				
2893						
2894	016223	124	040522	045503	MFKLOG:	.ASCIZ "TRACK ACCESSED MOVED TO UNIT 0 UNIT 1 ERRORS"<15><12>
2895	016230	020040	041501	042503		
2896	016236	051523	042105	020040		
2897	016244	046440	053117	042105		
2898	016252	052040	020117	020040		
2899	016260	052440	044516	020124		
2900	016266	020060	052440	044516		
2901	016274	020124	020061	051105		
2902	016302	047522	051522	005015		
2903	016310	000				
2904						
2905	016311	050	047516	042516	MNONE:	.ASCIZ "(NONE)"<15><12>
2906	016316	006451	000012			
2907						
2908	016322	047507	042117	000	MGOOD:	.ASCIZ "GOOD"
2909						
2910	016327	040	041440	042510	MSUM:	.ASCIZ " CHECK SUM "
2911	016334	045503	051440	046525		
2912	016342	000040				

```

2913
2914 016344 005015 054122 030461 MRX11: .ASCIZ <15><12>"RX11 / RXV11"
2915 016352 027440 051040 053150
2916 016360 030461 000
2917
2918 016363 015 052012 040522 002BIG: .ASCII <15><12> "TRACK LIMITS SELECTED OUT OF RANGE,"
2919 016370 045503 046040 046511
2920 016376 052111 020123 042523
2921 016404 042514 052103 042105
2922 016412 047440 052125 047440
2923 016420 020106 040522 043516
2924 016426 026105
2925 016430 051525 047111 020107 .ASCIZ "USING OD=0, ID=114"<15><12>
2926 016436 042117 030475 020054
2927 016444 042111 030475 032061
2928 016452 005015 000
2929
2930 016455 015 051412 041505 S2BIG: .ASCII <15><12> "SECTOR LIMITS SELECTED OUT OF RANGE,"
2931 016462 047524 020122 044514
2932 016470 044515 051524 051440
2933 016476 046105 041505 042524
2934 016504 020104 052517 020124
2935 016512 043117 051040 047101
2936 016520 042507 054
2937 016523 125 044523 043516 .ASCIZ "USING FIRST=1, LAST=32"<15><12>
2938 016530 043040 051111 052123
2939 016536 030475 020054 040514
2940 016544 052123 031475 006462
2941 016552 000012
2942
2943 016554 005015 040515 047111 MREV: .ASCIZ <15><12> "MAINDEC-11-DZRXA-E" <15><12>
2944 016562 042504 026503 030461
2945 016570 042055 051132 040530
2946 016576 042455 005015 000
2947
2948 016604 .EVEN
2949
2950 ;*****
2951
2952 ;THE FOLLOWING LOCATIONS ARE USED FOR DATA STORAGE,RETRY COUNTERS
2953 ;ACCESS COUNTERS ETC.
2954
2955 016604 000200 HUFADR: .BLKW 200
2956
2957 ;RETRY COUNTERS
2958 017004 000012 HURETRY: 10. ;FOR SETUP PURPOSE RURETRY MUST BE FIRST ON RETRY LIST
2959 017006 000012 WURETRY: 10.
2960 017010 000012 DURETRY: 10.
2961 017012 000012 DATARETRY: 10.
2962 017014 000012 P2URETRY: 10.
2963 017016 000012 PRETRY: 10.
2964 017020 000012 CRETRY: 10.
2965 017022 000012 SURETRY: 10. ;FOR SETUP PURPOSE SURETRY MUST BE LAST ON RETRY LIST
2966
2967
2968 ;GENERAL ERROR COUNTERS
  
```

```

2969 017024 000000 PARLOG: 0 ;FOR SETUP PURPOSE PARLOG MUST BE FIRST ON COUNTERS LIST
2970 017026 000000 MPANLOG: 0
2971 017030 000000 NOERLOG: 0
2972 017032 000000 UKNINT: 0
2973 017034 000000 INTER: 0
2974
2975 017036 000000 ;DRIVE RELATED ERROR COUNTERS
2976 017040 000000 ZSEKLOG: 0 ;"Z" LOGS ARE FOR DRIVE UNIT "Z"ERO
2977 017042 000000 SEKLOG: 0 ;NON"Z" LOGS ARE FOR DRIVE UNIT 1
2978 017044 000000 ZCRCLOG: 0
2979 017046 000000 CRCLOG: 0 ; * * * NOTE: * * *
2980 017050 000000 ZCRCBAD: 0 ;ERROR CODES MUST NOT BE CHANGED
2981 017052 000000 CCBAD: 0 ;FROM THIS ORDER, ERROR DUMP
2982 017054 000000 ZRDLOG: 0 ;ASSUMES TAGS OF LOGS ARE IN ORDER SHOWN.
2983 017056 000000 RDLUG: 0
2984 017060 000000 ZWRTLOG: 0
2985 017062 000000 WRTLOG: 0
2986 017064 000000 ZUATALOG: 0
2987 017066 000000 UATALOG: 0
2988 017070 000000 ZDUMIS: 0
2989 017072 000000 DUMIS: 0
2990 017074 000000 ZUNXDD: 0
2991 017076 000000 UNXDD: 0
2992 017100 000000 ZHSEKLOG: 0
2993 017102 000000 HSEKLOG: 0
2994 017104 000000 ZHCRCLOG: 0
2995 017106 000000 HCRCLOG: 0
2996 017110 000000 ZHCRCBAD: 0
2997 017112 000000 HCRCBAD: 0
2998 017114 000000 ZHRDLOG: 0
2999 017116 000000 HRDLOG: 0
3000 017120 000000 ZHWRTLOG: 0
3001 017122 000000 HWRTLOG: 0
3002 017124 000000 ZHDATALOG: 0
3003 017126 000000 HDATALOG: 0
3004 017130 000000 HDLOG: 0
3005
3006 ;ERROR CODES
3007 017132 000021 ERCODE: .BLKW 17,
3008
3009 ;THE FOLLOWING 2 BLOCKS OF WORDS ARE TRACK ACCESS COUNTER AND
3010 ;HEAD MOVED TO TRACK COUNTERS, THEY ARE DOUBLE PRECISION
3011 ;COUNTERS IN THE FOLLOWING FORMATT:
3012 ;
3013 ;LOC.X LOW ORDER WORD TRACK 0
3014 ;LOC.X+2 HIGH ORDER WORD TRACK 0
3015 ;LOC.X+4 LOW ORDER WORD TRACK 1
3016 ;LOC.X+6 HIGH ORDER WORD TRACK 1
3017 ;ETC.
3018
3019 ;TOTAL ACCESS / TRACK
3020 017174 000232 TRACC: .BLKW 232
3021
3022 ;TOTAL HEAD MOVEMENT TO TRACK
3023 017660 000232 HOMOVE: .BLKW 232
3024
  
```

```

3025                                     ;THE FOLLOWING 2 BLOCKS OF COUNTERS ARE SINGLE PRECISION
3026
3027                                     ;ERROR PER TRACK ON UNIT 0
3028 020344 000115      U0TRK:      .BLKW 115
3029
3030                                     ;ERROR PER TRACK ON UNIT 1
3031 020576 000115      U1TRK:      .BLKW 115
3032
3033                                     ;TOTAL WRITE AND READ FUNCTIONS DOUBLE PRECISION COUNTERS
3034 021030 000000 000000  WTCNTR:  .WORD 0,0
3035 021034 000000 000000  RDCNTR:  .WORD 0,0
3036
3037                                     ;RESTART AND PASSES COMPLETED COUNTERS
3038 021040 000000          RESTCNTR:  0
3039 021042 000000          PASCNTR:  0      ;FOR SETUP PURPOSE PASCNTR MUST BE PLACED LAST
3040
3041 021044 032737 000020 010704  U1LOG:  BIT #BIT4,UNITSEL      ;TEST FOR UNIT IN OPERATION
3042 021052 001402          BEQ 1$      ;IF BIT 4 IS 0 RETURN AND INC. UNIT 0 COUNTERS
3043 021054 062703 000002          ADD #2,R3      ;ADJUST THE CONTENTS OF R3 FOR
3044                                     ;THE ADDRESS OF UNIT 1 ERROR LOG COUNTERS.
3045 021060 000207          1$:      RTS PC
3046
3047                                     ;STATISTICAL ERROR REPORT ROUTEEN
3048                                     ;PRINTS ALL ERROR LOGS AND OPERATING CONDITIONS
3049
3050 ;/*****
3051
3052 021062 012706 001200  ERDUMP:  MOV #STACK,SP
3053 021066 104400 015240          TYPE ,DBLLF
3054 021072 005737 022514          TST SHURTRPT      ;IS SHORT REPORT REQUESTED
3055 021076 001405          BEQ 1$      ;NO,PRINT LONG REPORT
3056 021100 104400 015040          TYPE ,TAB
3057 021104 104400 016076          TYPE ,M$HUNKT      ;YES,TYPE SHORT REPOR HEADER
3058 021110 000404          BR 2$
3059 021112 104400 015040          1$:  TYPE ,TAB
3060 021116 104400 016055          TYPE ,MDUMP
3061 021122 104400 016554          TYPE ,MREV      ;TYPE LONG REPORT HEADER
3062 021126 004737 002452          JSR PC,ICOND      ;PRINT NAME AND REVISION OF PROGRAM
3063 021132 005737 021040          TST WESTCNTR      ;TYPE OUT SELECTED DRIVES AND TESTS
3064 021136 001412          BEQ 3$      ;HAVE THERE BEEN ANY RESTARTS
3065 021140 104400 016111          TYPE ,MRESTART      ;NO
3066 021144 013737 021040 021156  MOV WESTCNTR,12$      ;YES,PRINT OUT HOW MANY
3067 021152 004537 014446          JSR #5,SGLDEC
3068 021156 000000          OPEN
3069 021160 104400 015240          TYPE ,DBLLF
3070 021164 005737 021042          TST PASCNTR      ;HAVE ANY PASSES BEEN COMPLETED
3071 021170 001005          BNE 4$      ;YES,PRINT OUT HOW MANY
3072 021172 104400 016125          TYPE ,MNPAS
3073 021176 104400 015240          TYPE ,DBLLF      ;NO,TYPE PASS ABORTED
3074 021202 000412          BR 5$
3075 021204 104400 016142          TYPE ,MPASS      ;TYPE OUT NUMBER OF PASSES
3076 021210 013737 021042 021222  MOV PASCNTR,7$
3077 021216 004537 014446          JSR #5,SGLDEC
3078 021222 000000          OPEN
3079 021224 104400 015052          TYPE ,MCRLF
3080 021230 005737 022514          TST SHURTRPT      ;IS THIS A SHORT REPORT

```

```

3081 021234 001030          BNE AERRS      ;YES,DON'T PRINT WRITE/READ TOTALS
3082
3083                                     ;PRINTS TOTAL NUMBER OF SECTORS WRITTEN AND/OR
3084                                     ;READ FOR THE DURATION OF RUN TIME.
3085 021236 104400 016154          TYPE ,MWRTEEN      ;PRINT WRT/RD MESSAGE
3086 021242 104400 016165          TYPE ,MSLASH
3087 021246 104400 015674          TYPE ,MREAD
3088 021252 104400 015047          TYPE ,DBLSP
3089 021256 012746 021030          MOV #WTCNTR,-(SP)      ;ADDRESS OF TOTAL WRITES ON STACK
3090 021262 004737 014202          JSR PC,##SUB2D      ;TYPE TOTAL WRITES
3091 021266 004737 014376          JSR PC,RTJUST
3092 021272 104400 016165          TYPE ,MSLASH
3093 021276 012746 021034          MOV #RDCNTR,-(SP)      ;ADDRESS OF TOTAL READS ON STACK
3094 021302 004737 014202          JSR PC,##SUB2D      ;TYPE TOTAL READS
3095 021306 004737 014376          JSR PC,RTJUST
3096 021312 104400 015240          TYPE ,DBLLF
3097
3098                                     ;TYPE OUT MICRO CPU RELATED ERRORS
3099 021316 104400 015052          AERRS:  TYPE ,MCRLF
3100 021322 005737 017024          TST #PARLOG      ;WERE THERE ANY PARITY ERRORS
3101 021326 001435          BEQ 1$
3102 021330 013737 017024 021342  MOV #PARLOG,20$
3103 021336 004537 014446          JSR #5,SGLDEC      ;YES,TYPE OUT THE NUMBER
3104 021342 000000          OPEN
3105 021344 104400 015047          20$:  TYPE ,DBLSP
3106 021350 104400 016017          TYPE ,M$PAR
3107 021354 104400 015052          TYPE ,MCRLF
3108 021360 005737 017026          TST #H$PARLOG      ;WERE THERE ANY HARD PARITY ERRORS
3109 021364 001416          BEQ 1$
3110 021366 013737 017026 021400  MOV #H$PARLOG,21$
3111 021374 004537 014446          JSR #5,SGLDEC      ;YES,TYPE OUT THE COUNT
3112 021400 000000          OPEN
3113 021402 104400 015047          21$:  TYPE ,DBLSP
3114 021406 104400 015755          TYPE ,MUNREC
3115 021412 104400 016017          TYPE ,M$PAR
3116 021416 104400 015052          TYPE ,MCRLF
3117 021422 005737 017030          TST #OERRLOG      ;ANY STATUS ERROR FLAG ERRORS
3118 021426 001414          BEQ 1$      ;IF NONE CHECK NEXT ERROR
3119 021430 013737 017030 021442  MOV #OERRLOG,24$
3120 021436 004537 014446          JSR #5,SGLDEC      ;YES,TYPE OUT COUNT
3121 021442 000000          OPEN
3122 021444 104400 015047          24$:  TYPE ,DBLSP
3123 021450 104400 016034          TYPE ,MNOFLAG
3124 021454 104400 015052          TYPE ,MCRLF
3125 021460 005737 017034          25$:  TST INTER
3126 021464 001414          BEQ 4$
3127 021466 013737 017034 021500  MOV INTER,25$
3128 021474 004537 014446          JSR #5,SGLDEC      ;IF NONE CHECK NEXT ERROR GROUP
3129 021500 000000          OPEN
3130 021502 104400 015047          25$:  TYPE ,DBLSP
3131 021506 104400 015260          TYPE ,MINTER
3132 021512 104400 015052          TYPE ,MCRLF
3133
3134                                     ;TYPES DRIVE RELATED ERRORS
3135 021516 104400 015240          4$:   TYPE ,DBLLF
3136 021522 104400 015106          TYPE ,MUNIT0      ;TYPE OUT COLUMN HEADINGS
3137 021526 104400 015116          TYPE ,MUNIT1

```

3137	021532	104400	016170	TYPE ,SPACE	
3138	021536	104400	016067	TYPE ,MERS	
3139	021542	104400	015052	TYPE ,MCRLF	
3140	021546	005002		CLR R2	;;R2 USED AS AN ERROR PRINTED FLAG
3141	021550	013746	022516	MOV ZERO,=(SP)	;;PUT ADDR OF ZERO MARK ON STACK
3142	021554	012746	014462	MOV #MUNXDD,=(SP)	;;PUT ADDRESSES OF ALL RECOVERABLE
3143	021560	012746	014506	MOV #MDDHIS,=(SP)	ERROR MESSAGES ON THE STACK
3144	021564	012746	014527	MOV #MDEHNR,=(SP)	
3145	021570	012746	016010	MOV #MWRITE,=(SP)	
3146	021574	012746	015674	MOV #MREAD,=(SP)	
3147	021600	012746	015644	MOV #MBADCM,=(SP)	
3148	021604	012746	015736	MOV #MCRC,=(SP)	
3149	021610	012746	015775	MOV #MSEK,=(SP)	
3150	021614	012701	017036	MOV #ZSEKLOG,R1	;;PUT FIRST ERROR COUNTER IN REGISTER
3151	021620	005716		TST (SP)	;;GET ADDRESS OF MESSAGE
3152	021622	001436		REQ 35	;;IF 0 FINISHED SOFT ERRS DO HARD ERRS
3153	021624	005721		TST (R1)+	;;IS DRIVE 0 COUNTER CLEAR
3154	021626	001005		BNE 65	;;NO,GO PRINT ERROR COUNTER
3155	021630	005711		TST (R1)	;;IS DRIVE 1 COUNTER CLEAR
3156	021632	001005		BNE 65	;;NO,GO PRINT ERROR COUNTER
3157	021634	005721		TST (R1)+	;;ADJUST R1 FOR NEXT UNIT 0 COUNTER
3158	021636	005726		TST (SP)+	;;ADJUST STACK FOR NEXT ADDRESS
3159	021640	000767		HR 75	;;GO TEST NEXT PAIR OF COUNTERS
3160	021642	005202		INC R2	;;MAKE ERROR PRINTED FLAG NON ZERO
3161	021644	014137	021654	MOV =(R1),505	;;TYPE CONTENTS OF UNIT 0 COUNTER
3162	021650	004537	014446	JSR R5,SGLDEC	
3163	021654	000000		OPEN	
3164	021656	104400	015047	TYPE ,DBLSP	
3165	021662	005721		TST (R1)+	;;ADJUST R1 FOR UNIT 1 COUNTER
3166	021664	012137	021674	MOV (R1)+,515	;;TYPE COUNTER,GET ADDR OF NEXT IN R1
3167	021670	004537	014446	JSR R5,SGLDEC	
3168	021674	000000		OPEN	
3169	021676	104400	015047	TYPE ,DBLSP	
3170	021702	012637	021710	MUV (SP)+,135	
3171	021706	104400		TYPE	;;TYPE ERROR MESSAGE FROM STACK
3172	021710	000000		OPEN	
3173	021712	104400	015052	TYPE ,MCRLF	
3174	021716	000744		HR 75	;;GET NEXT COUNTER
3175	021720	104400	015052	TYPE ,MCRLF	
3176	021724	013746	022516	MOV ZERO,=(SP)	;;PUT ADDR OF ZERO MARK ON THE STACK
3177	021732	012746	015015	MOV #MDEHNR,=(SP)	;;PUT ADDRESS OF ALL HARD ERROR
3178	021734	012746	014527	MOV #MDEHNR,=(SP)	MESSAGES ON THE STACK
3179	021740	012746	016010	MOV #MWRITE,=(SP)	
3180	021744	012746	015674	MOV #MREAD,=(SP)	
3181	021750	012746	015644	MOV #MBADCM,=(SP)	
3182	021754	012746	015736	MOV #MCRC,=(SP)	
3183	021760	012746	015775	MOV #MSEK,=(SP)	
3184	021764	012701	017076	MOV #ZSEKLOG,R1	;;PUT FIRST HARD ERROR COUNTER IN R1
3185	021770	005716		TST (SP)	;;GET ADDRESS OF MESSAGE FROM STACK
3186	021772	001437		REQ MERS	;;IF 0 FINISHED ERROR LOGS
3187	021774	005721		TST (R1)+	;;IS DRIVE 0 COUNTER CLEAR
3188	021776	001005		BNE 115	;;NO,PRINT ERROR COUNTER AND MESSAGE
3189	022000	005711		TST (R1)	;;IS DRIVE 1 COUNTER CLEAR
3190	022002	001005		BNE 115	;;NO,GO PRINT THE ERROR
3191	022004	005721		TST (R1)+	;;ADJUST R1 FOR NEXT UNIT 0 LOG
3192	022006	005726		TST (SP)+	;;ADJUST STACK FOR NEXT ADDRESS

3193	022010	000767		HR 125	
3194	022012	014137	022022	MOV =(R1),525	;;TYPE CONTENTS OF UNIT 0 COUNTER
3195	022016	004537	014446	JSR R5,SGLDEC	
3196	022022	000000		OPEN	
3197	022024	104400	015047	TYPE ,DBLSP	
3198	022030	005721		TST (R1)+	;;ADJUST R1 FOR UNIT 1 COUNTER
3199	022032	012137	022042	MUV (R1)+,535	;;TYPE UNIT 1 LOG
3200	022036	004537	014446	JSR R5,SGLDEC	
3201	022042	000000		OPEN	
3202	022044	104400	015047	TYPE ,DBLSP	
3203	022050	104400	015755	TYPE ,MUNHEC	;;TYPE UNRECOVERABLE
3204	022054	012637	022062	MUV (SP)+,145	
3205	022060	104400		TYPE	;;ERROR MESSAGE FROM STACK
3206	022062	000000		OPEN	
3207	022064	104400	015052	TYPE ,MCRLF	
3208	022070	000737		HR 125	;;GET NEXT LOG
3209				;;PRINTS ERRORS PER ERROR CODES	
3210	022072	005702		TST R2	;;WAS AN ERROR PRINTED IN THE LAST GROUP
3211	022074	001004		BNE 55	;;IF R2 IS NONZERO BRANCH
3212	022076	104400	015040	TYPE ,TAB	
3213	022102	104400	016311	TYPE ,MNONE	;;TYPE NONE
3214	022106	005002		CLR R2	;;CLEAR ERROR PRINTED FLAG AGAIN
3215	022110	104400	015240	TYPE ,DBLLF	
3216	022114	104400	016172	TYPE ,MCODES	;;TYPE ERROR PER ERROR CODE
3217	022120	012700	000001	MUV #1,R0	;;SET UP CODE COUNTER
3218	022124	012701	017132	MOV #ENCODE,R1	;;GET ADDR OF FIRST ERROR CODE
3219	022130	020027	000022	CMP R0,#22	;;IS THE LAST CODE PRINTED
3220	022134	001001		BNE 55	;;NO,TEST THE NEXT ONE
3221	022136	000427		HR TYPTRK	;;YES,GO PRINT TRACK INFORMATION
3222	022140	005721		TST (R1)+	;;TEST FOR 0 WORD,GET NEXT ADDRESS
3223	022142	001002		BNE 15	;;IF NOT 0 TYPE COUNT AND CODE NUMBER
3224	022144	005200		INC R0	;;INC CODE COUNTER
3225	022146	000770		HR 25	
3226	022150	014137	022160	MOV =(R1),45	;;PUT CONTENTS OF COUNTER IN OPEN
3227	022154	004537	014446	JSR R5,SGLDEC	;;FOR TYPE OUT
3228	022160	000000		OPEN	
3229	022162	104400	015040	TYPE ,TAB	
3230					
3231	022166	010002		MOV R0,R2	;;PUT CODE NUMBER IN REGISTER TO
3232	022170	006302		ASL R2	;;TYPE IT OUT IN THE FROM OF ERROR CODE
3233	022172	006302		ASL R2	
3234	022174	006302		ASL R2	
3235	022176	010246		MUV R2,=(SP)	;;SAVE R2 FOR TYPEOUT
3236	022200	104400		;;TYPE	;;GO TYPE--OCTAL ASCII
3237	022202	004		.BYTE 4	;;TYPE 4 DIGIT(S)
3238	022203	001		.BYTE 1	;;TYPE LEADING ZEROS
3239	022204	104400	015052	TYPE ,MCRLF	
3240	022210	005200		INC R0	;;SET UP FOR NEXT COUNTER
3241	022212	005721		TST (R1)+	;;GET NEXT ERROR COUNTER
3242	022214	000745		HR 25	
3243				;;PRINTS TRACK ACCESS,HEAD MOVEMENT,AND ERRORS PER TRACK	
3244	022216	005702		TST R2	;;WAS AN ERROR PRINTED IN LAST GROUP
3245	022220	001004		BNE 55	;;IF PRINTED ERROR FLAG NONZERO BRANCH
3246	022222	104400	015040	TYPE ,TAB	
3247	022226	104400	016311	TYPE ,MNONE	;;TYPE NONE
3248	022232	104400	015240	TYPE ,DBLLF	

3249	022236	005737	022514	TST SHORTRPT	IF SHORT REPORT DON'T TYPE TRACK ACCESS OR ERRO
3250	022242	001121		BNE 65	
3251	022244	104400	016223	TYPE ,MTKLOG	ITYPE FORMAT OF ERROR PRINT OUT
3252	022250	005037	022330	CLR 25	ITRACK ADDRESS COUNTER
3253	022254	012704	017170	MOV #TKACC-4,R4	ISSET UP ADDRESS OF TRACK ACCESSED
3254	022260	012703	017654	MOV #HDMOVE-4,R3	IADDRESS OF HEAD MOVEMENT COUNTER
3255	022264	012702	020344	MOV #U1TRK,R2	IUNIT 0 ERROR PER TRACK COUNTER
3256	022270	012701	020576	MOV #U1TRK,R1	IUNIT 1 ERORR PER TRACK COUNTER
3257	022274	062704	000004	ADD #4,R4	IADJUST FOR ADDRESS OF NEXT COUNTER
3258	022300	005714		TST (R4)	IHAS THIS TRACK BEEN ACCESSED
3259	022302	001010		BNE 105	IYES,PRINT OUT THE COUNTERS
3260	022304	005764	000002	TST 2(M4)	ITEST UPPER WORD OF ACCESS COUNTER
3261	022310	001005		BNE 105	
3262	022312	062703	000004	ADD #4,R3	IThis TRACK HAS NOT BEEN USED, ADJUST
3263	022316	005722		TST (R2)+	IREGISTERS TO POINT TO NEXT TRACK COUNTERS
3264	022320	005721		TST (R1)+	
3265	022322	000463		BR 125	ITEST FOR LAST TRACK
3266	022324	004537	014446	JSR #5,SGLUEC	ITYPE TRACK NUMBER
3267	022330	000000		OPEN	IThis LOCATION USED AS TRACK ADDR COUNTER
3268	022332	104400	016170	TYPE ,SPACE	
3269	022336	010446		MOV #4,-(SP)	ITYPE TRACK ACCESSED COUNT
3270	022340	004737	014202	JSR PC,##SDB20	
3271	022344	004737	014376	JSR PC,RTJUST	
3272	022350	104400	016170	TYPE ,SPACE	
3273	022354	062703	000004	ADD #4,R5	
3274	022360	010346		MOV #3,-(SP)	ITYPE HEAD MOVED TO COUNT
3275	022362	004737	014202	JSR PC,##SDB20	
3276	022366	004737	014376	JSR PC,RTJUST	
3277	022372	104400	015047	TYPE ,DBLSP	
3278	022376	104400	016170	TYPE ,SPACE	
3279	022402	005712		TST (R2)	IFARE THERE ANY UNIT 0 ERRORS
3280	022404	001421		BEQ 205	IFNO,CHECK UNIT 1
3281	022406	012237	022416	MOV (R2)+,55	ITYPE UNIT 0 ERRORS
3282	022412	004537	014446	JSR #5,SGLUEC	
3283	022416	000000		OPEN	
3284	022420	005711		TST (R1)	IFARE THERE ANY UNIT 1 ERRORS
3285	022422	001420		BEQ 75	IFNO,GO ADJUST POINTERS
3286	022424	104400	016170	TYPE ,SPACE	
3287	022430	104400	015047	TYPE ,DBLSP	
3288	022434	012137	022444	MOV (R1)+,45	ITYPE UNIT 1 ERRORS
3289	022440	004537	014446	JSR #5,SGLUEC	
3290	022444	000000		OPEN	
3291	022446	000407		BR 115	IFGO TO NEXT TRACK
3292					
3293	022450	005722		TST (R2)+	IFNO ERRORS,ADJUST REG,FOR NEXT COUNTER
3294	022452	005711		TST (R1)	IFARE THERE ANY UNIT 1 ERRORS
3295	022454	001403		BEQ 75	IFNO,GO TO NEXT TRACK
3296	022456	104400	015040	TYPE ,TAB	IFYES,TYPE TAB FOR PLACEMENT
3297	022462	000762		BR 225	IFGO TYPE THE NUMBER
3298	022464	005721		TST (R1)+	IFNO ERRORS,ADJUST REG. FOR NEXT COUNTER
3299	022466	104400	015052	TYPE ,MCRLF	
3300	022472	005237	022330	INC 25	
3301	022476	023727	022330	CMP 25,#115	IFHAVE ALL TRACK ERRORS BEEN TYPED
3302	022504	001273		BNE 15	IFNO,TYPE THE NEXT ONE
3303	022506	005037	022514	CLR SHORTRPT	IFSET UP FOR LONG REPORT
3304	022512	000000		HLT 17:	IFYES DONE

3305					
3306	022514	000000		SHORTRPT:	0
3307	022516	000000		ZEND:	0
3308					
3309	000001				.END















.SRAND 1#  
 .SRDUE 1#  
 .SRDOC 1#  
 .SREAD 1# 5# 2315  
 .SR2AZ 1#  
 .SSAVE 1# 6# 2270  
 .SSP20 1# 6# 2539  
 .SSB20 1#  
 .SSCUP 1#  
 .SSIZE 1#  
 .SSUPR 1#  
 .STRAP 1# 6# 2458  
 .STYPB 1#  
 .STYPD 1#  
 .STYPE 1# 5# 2114  
 .STYPO 1# 5# 2193  
 .S4CCA 1#  
 .1170 1#

ADC	808	891	943	1431	1986	2587											
ADD	807	866	879	882	889	890	942	1058	1066	1171	1180	1181	1430	1476	1499		
	1537	1754	1804	1904	1985	2053	2061	2062	2072	2105	2109	2141	2221	2231	2356		
ASL	2365	2553	2586	2588	3043	3257	3262	3273									
	717	878	887	888	1658	1899	1955	1982	1983	2027	2379	2380	2381	2470	3232		
ASLB	3233	3234															
BCC	1183																
BEQ	1702																
	444	503	512	516	532	561	576	579	596	599	601	662	785	805	823		
	856	881	951	961	980	1017	1043	1064	1080	1101	1114	1203	1207	1240	1243		
	1253	1266	1277	1381	1400	1408	1427	1463	1490	1495	1529	1605	1707	1757	1817		
	1824	1840	1970	2056	2144	2179	2248	2336	2363	2378	2631	3042	3055	3064	3101		
	3109	3118	3126	3152	3186	3280	3285	3295									
BGE	670	1959	2052	2107													
BGT	1205	2255	2375	2416													
BHI	578	581	598	603													
BIC	533	541	543	551	1218	1382	1551	1587	1594	1783	1790	1833	1855	1924	1940		
	1953	1968	2025	2044	2046	2063	2073	2245	2332	2349	2376	2403	2409	2417			
BICB	711	1651	1893	1998	2003												
BIS	498	520	1091	1118	1152	1159	1197	1384	1485	1784	1791	1868	2250	2251	2383		
	2590																
BISB	718	892	1659	1900													
BIT	488	499	504	515	655	661	772	781	804	822	845	853	880	979	986		
	1026	1014	1032	1040	1061	1079	1088	1098	1113	1121	1200	1202	1245	1263	1377		
	1380	1426	1445	1462	1465	1471	1480	1489	1504	1521	1528	1555	1604	1779	1788		
	1802	1816	1823	3041													
BITB	511	1081	1405	1467	1582	2006	2167										
BLE	2048																
BLOS	2429																
BLT	2158	2256	2373	2414	2583												
BMI	1461	1533	1596														
BNE	451	453	467	471	489	505	530	594	618	629	647	656	672	713	742		
	773	782	814	816	846	854	922	975	987	1001	1007	1015	1033	1041	1050		
	1062	1082	1089	1099	1122	1142	1155	1173	1201	1246	1264	1301	1318	1341	1368		
	1375	1378	1404	1416	1433	1446	1466	1468	1472	1481	1505	1522	1556	1583	1654		
	1780	1789	1803	1812	1835	1838	1857	1895	1926	2007	2014	2029	2087	2138	2146		
	2154	2168	2175	2246	2328	2334	2354	2361	2368	2405	2411	2433	2439	2518	2594		
	2633	3071	3081	3154	3156	3188	3190	3211	3220	3223	3245	3250	3259	3261	3322		
BPL	492	497	502	519	525	638	641	666	779	851	972	997	1012	1038	1096		
	1209	1212	1234	1397	1451	1589	1782	1787	1801	1939	1952	1996	2002	2024	2132		
	2172	2244	2330	2346	2401	2407											
BR	210	211	212	445	455	507	514	534	592	616	623	626	664	719	744		
	761	924	993	1075	1124	1144	1175	1210	1255	1303	1320	1343	1383	1385	1492		
	1525	1527	1531	1558	1607	1661	1682	1691	1710	1718	1729	1742	1752	1819	1903		
	1928	1931	1942	1944	1960	1972	1974	2000	2005	2010	2016	2032	2034	2050	2054		
	2058	2134	2151	2161	2170	2177	2222	2237	2258	2357	2384	2386	2412	2435	2510		
	2534	2585	2635	3058	3074	3159	3174	3193	3208	3221	3225	3242	3265	3291	3297		
CLC	539	1705	2064														
CLR	465	495	574	622	625	706	709	710	753	898	938	1157	1158	1161	1164		
	1280	1369	1392	1409	1412	1424	1652	1680	1717	1809	1832	1861	1863	2068	2235		
	2343	2344	2516	2579	3140	3214	3252	3303									
CLRB	506	1859	2093	2150	2176	2440	2595										
CMP	443	446	452	466	470	950	960	1204	1399	1756	1834	2327	2333	2353	2360		
	2372	2374	2404	2410	2413	2415	2428	2589	3219	3301							
CMPB	529	577	580	597	602	617	668	1172	1925	1969	2013	2028	2047	2051	2055		
	2106	2143	2145	2153	2174	2178	2335	2367	2432	2438	2632						

COM	914	1065	1134	1330	1561	1597									
COMB	1728														
DEC	560	671	716	741	783	815	855	921	974	1016	1042	1063	1100	1141	1265
DEC8	1276	1300	1517	1340	1367	1374	1657	1898	1943	1954	2011	2026	2031	2592	
EMT	2157	2160	2243	2254											
HALT	47														
INC	174	538	570	649	674	780	852	1013	1026	1039	1097	1235	1418	1435	1452
INC	1831	2133	2509	2533	3304										
INC	554	565	632	704	760	770	787	842	861	883	954	964	998	1004	1021
INC	1033	1047	1053	1070	1085	1105	1117	1174	1198	1228	1236	1260	1271	1401	1415
INC	1443	1469	1478	1502	1703	1810	1815	1837	1867	1930	1997	2008	2088	2092	2110
INC8	2249	2257	2302	2517	2584	3160	3224	3240	3300						
IOT	2180														
JMP	213	214	215	552	571	650	675	720	721	722	723	724	725	726	767
JMP	786	793	811	818	819	828	858	946	985	1018	1044	1059	1067	1102	1112
JMP	1156	1195	1237	1268	1279	1464	1542	1590	1598	1662	1663	1664	1665	1666	1667
JSR	1668	1904	1905	1906	1907	1908	1909	1910							
JSR	487	493	503	510	513	523	582	585	590	604	608	613	644	651	660
JSR	663	667	705	736	737	738	739	740	743	746	747	750	758	809	810
JSR	838	841	843	848	857	860	865	872	894	896	915	916	917	918	919
JSR	920	923	931	932	944	945	955	963	970	992	995	999	1003	1020	1029
JSR	1046	1052	1069	1084	1086	1092	1104	1110	1116	1119	1135	1136	1137	1138	1139
JSR	1140	1143	1153	1160	1166	1216	1220	1249	1259	1261	1267	1270	1278	1295	1296
JSR	1297	1298	1299	1302	1312	1313	1314	1315	1316	1319	1331	1332	1333	1334	1335
JSR	1336	1337	1338	1339	1342	1358	1362	1363	1364	1365	1366	1371	1372	1373	1376
JSR	1453	1488	1491	1510	1510	1520	1524	1530	1536	1547	1552	1603	1606	1681	1704
JSR	1727	1749	1751	1808	1828	1836	1842	2045	2152	2159	2166	2371	2552	2646	2647
JSR	3062	3067	3077	3090	3091	3094	3095	3103	3111	3120	3128	3162	3167	3195	3200
JSR	3227	3266	3270	3271	3275	3276	3282	3289							
MOV	436	437	438	439	442	447	448	458	459	460	461	462	463	464	468
MOV	469	473	478	483	494	509	540	547	556	657	658	659	676	702	702
MOV	701	702	707	708	714	715	748	749	751	752	754	755	757	764	765
MOV	766	800	801	802	825	826	827	831	836	837	840	859	871	877	886
MOV	893	899	933	934	935	936	937	939	940	952	955	962	965	982	983
MOV	984	989	990	991	1002	1019	1028	1045	1048	1051	1068	1071	1083	1103	1115
MOV	1162	1163	1165	1167	1179	1186	1192	1193	1194	1219	1223	1229	1248	1258	1269
MOV	1359	1360	1370	1379	1393	1394	1413	1422	1425	1428	1486	1487	1496	1509	1514
MOV	1523	1526	1546	1550	1561	1567	1573	1602	1611	1617	1655	1656	1660	1701	1814
MOV	1827	1839	1865	1896	1897	1956	1961	2057	2067	2069	2074	2075	2097	2135	2136
MOV	2136	2140	2155	2218	2226	2227	2228	2234	2241	2259	2260	2261	2262	2263	2288
MOV	2289	2290	2291	2292	2293	2294	2295	2296	2297	2304	2305	2306	2307	2308	2309
MOV	2310	2311	2312	2313	2340	2364	2369	2399	2426	2427	2442	2443	2444	2445	2446
MOV	2466	2467	2471	2498	2499	2500	2501	2502	2503	2504	2505	2506	2507	2508	2514
MOV	2515	2519	2520	2521	2522	2523	2524	2525	2526	2527	2530	2530	2531	2534	2571
MOV	2572	2573	2574	2575	2576	2577	2578	2627	2628	2629	2639	2640	2645	3052	3066
MOV	3076	3089	3093	3102	3110	3119	3127	3141	3142	3143	3144	3145	3146	3147	3148
MOV	3149	3150	3161	3166	3170	3176	3177	3178	3180	3181	3182	3183	3184	3185	3186
MOV	3199	3204	3217	3218	3226	3231	3235	3254	3255	3256	3269	3274	3281	3288	3289
MOV8	456	584	589	607	612	759	803	806	895	897	941	1170	1177	1182	1185
MOV8	1185	1227	1459	1493	1601	1608	1690	1699	1726	1741	1750	1755	1858	1860	1862
MOV8	1864	1892	1927	1941	1971	1973	1999	2009	2012	2015	2030	2033	2069	2090	2094
MOV8	2094	2108	2137	2165	2173	2219	2220	2223	2224	2225	2229	2232	2233	2252	2331
MOV8	2348	2402	2408	2431	2436	2469	2591	2634	2636						
NEG	2230														
NEG8	1180														

NOP	1843	1844	1845												
RESET	435	1841													
ROL	548	549	2065	2066	2236	2238	2239	2240	2242						
ROLB	1709														
ROR	544	545	546	1497	1498	2070	2071								
RTI	440	703	756	900	1166	1395	1414	1475	2142	2264	2298	2314	2370	2418	2446
RTS	2532														
RTS	521	564	619	633	652	677	817	867	875	884	901	973	976	1125	1187
RTS	1241	1281	1398	1429	1477	1538	1577	1622	1758	1760	1785	1792	1805	1846	1869
RTS	1987	2076	2098	2111	2182	2072	2555	2597	2641	2648	3045				
SBC	2501														
SEC	1700	1708													
SUB	1866	1957	1958	2049	2091	2095	2500	2582							
SWAB	550														
TRAP	2474	2483	2484	2485	2487	2489	2490	2491	2492	2493					
TST	450	491	501	531	575	595	620	640	646	665	712	763	778	784	813
TST	821	850	971	978	996	1000	1011	1037	1049	1095	1154	1191	1206	1233	1239
TST	1242	1411	1432	1450	1460	1494	1508	1653	1706	1759	1786	1800	1811	1856	1894
TST	2086	2139	2147	2169	2247	2362	2377	2468	3054	3063	3070	3060	3100	3108	3117
TST	3125	3151	3153	3155	3157	3158	3165	3185	3187	3189	3191	3192	3198	3210	3222
TST8	3241	3244	3249	3258	3260	3263	3264	3279	3284	3293	3294	3298			
TST8	496	518	524	593	600	637	1208	1211	1252	1396	1532	1595	1761	1938	1951
TST8	1995	2001	2023	2131	2171	2329	2345	2400	2406	2630					
.ASCII	2190	2191	2682	2689	2918	2930									
.ASCII	2192	2450	2451	2452	2454	2536	2651	2656	2661	2666	2670	2674	2677	2697	2702
.ASCII	2706	2709	2711	2713	2715	2717	2719	2724	2727	2730	2734	2738	2741	2744	2747
.ASCII	2750	2752	2754	2756	2758	2764	2768	2774	2777	2781	2784	2787	2794	2796	2798
.ASCII	2801	2804	2810	2815	2818	2822	2827	2829	2832	2835	2838	2843	2847	2850	2853
.ASCII	2857	2861	2864	2867	2870	2874	2878	2881	2884	2886	2888	2894	2905	2908	2910
.ASCII	2914	2925	2937	2943											
.BLKB	2449	2618	2955												
.BLKB	3007	3020	3023	3028	3031										
.BYTE	475	476	480	481	485	486	558	559	516	1517	1517	1563	1564	1569	1575
.BYTE	1576	1613	1614	1619	1620	2186	2187	2188	2189	2265	2266	2267	2268	2447	2448
.BYTE	2456	2457	3237	3238											
.DSABL	2387														
.ENABL	1	4	2320												
.END	3309														
.ENDC	15	47	139	153	195	208	219	251	271	289	349	414	432	452	458
.ENDC	476	477	481	482	486	487	559	560	680	699	731	799	909	927	1130
.ENDC	1148	1289	1307	1325	1347	1389	1456	1517	1518	1564	1565	1570	1571	1576	1577
.ENDC	1614	1615	1620	1621	1626	1648	1676	1685	1694	1713	1721	1736	1		

	2456	2460	2466	2470	2474	2483	2484	2485	2486	2487	2489	2490	2491	2492	2493
	2494	2496	2506	2507	2512	2519	2520	2520	2530	2532	2536	2541	2559	2620	2950
	3050	3237	3230												
.IFF	47	195	219	251	271	289	344	414	432	475	476	480	481	486	559
	680	699	731	749	909	927	1130	1148	1289	1307	1325	1347	1389	1456	1517
	1564	1570	1576	1614	1620	1626	1640	1676	1685	1694	1713	1721	1736	1745	1765
	1778	1851	1874	1891	1920	1934	1947	1964	1977	1991	2019	2039	2083	2117	2196
	2273	2318	2323	2391	2393	2398	2414	2420	2429	2433	2450	2461	2467	2497	2513
	2530	2542	2560	2621	2451	3051	3230	3259							
.IFT	2393	2398													
.IFTF	2338	2391	2394												
.IIF	10	15	20	21	474	479	484	557	1515	1562	1568	1574	1612	1618	2184
	2185	2186	2187	2188	2189	2190	2191	2192	2193	2318	2319	2341	2442	2450	2456
	2457	2458	2482	2483	2484	2485	2487	2489	2490	2491	2492	2493	3236		
.IRP	208	2280	2308	2500	2506	2514	2520								
.LIST	1	3	153	174	208	2419	2474	2482	2483	2484	2485	2486	2467	2488	2489
.MACRO	2490	2491	2492	2493	2494										
.MCALL	1	2474													
.NLIST	5	6	153												
	1	2	153	174	208	2419	2474	2482	2483	2484	2485	2486	2487	2488	2489
.REPT	2490	2491	2492	2493	2494										
.SBTTL	174														
.TITLE	43	449	2114	2193	2270	2315	2458	2474	2494	2539	2557				
.WORD	173	187	344	345	346	1225	2161	2184	2185	2269	2318	2319	2529	2531	2556
	2593	3034	3035												

ERRORS DETECTED: 0  
 DEFAULT GLOBALS GENERATED: 0

\*.DZRKA,SEQ/SOL/CRF=SYSMAC,SML,DZRKA,P11  
 RUN-TIME: 42 44 6 SECONDS  
 RUN-TIME RATIO: 375/954.0  
 CORE USED: 34K (67 PAGES)