RX11

RX11 DIAGNOSTIC

MD-11-DZRXB-E

EP-DZRXB-E-DL-B

FICHE 1 OF 1

JUL 1976

digital

MADE IN USA

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZRXB-E-D
PRODUCT NAME:	RX11 INTERFACE DIAGNOSTIC
DATE:	APRIL 1976
MAINTAINER:	DIAGNOSTIC ENGINEERING
AUTHOR:	DAVID L. ADAMS

TABLE OF CONTENTS

1.0 GENERAL PROGRAM INFORMATION

1.1 ABSTRACT

THE RX11 INTERFACE DIAG'IOSTIC CONSISTS OF A SERIES OF SELECTABLE TESTS THAT MAY BE RUN INDIVIDUALLY, SEQUENCE THROUGH ALL TESTS, OR START AT A SELECTED TEST AND RUN THROUGH REMAINING TESTS, IN ORDER, THEN GO BACK TO THE SELECTED TEST.

THESE TESTS CHECK OUT THE BASIC FUNCTIONS OF THE RX11 INTERFACE SUCH AS:

A. DONE FLAG
B. INTERRUPT LEVEL / ADDRESS
C. PROGRAM INITIALIZE
D. READ STATUS REGISTERS
E. FILL / EMPTY BUFFER TRANSFER VERIFICATION
F. FILL / EMPTY BUFFER WITH DATA PATTERNS

IT IS NECESSARY TO INSURE THAT THESE FUNCTIONS WORK BEFORE A DATA RELIABILITY TEST IS RUN.

ANY ERRORS ARE REPORTED BY THE PROGRAM, AND IT IS POSSIBLE TO LOOP ON THE ERROR OR A PARTICULAR TEST FOR SCOPE TESTING.

1.2 SYSTEM REQUIREMENTS

1.2.1 HARDWARE REQUIREMENTS

THE FOLLOWING EQUIPMENT IS REQUIRED:

A. PDP-11 SERIES COMPUTER WITH MINIMUM OF 8K MEMORY
B. RX11 FLOPPY DISK SYSTEM, INCLUDING A SINGLE OR DUAL DRIVE RX01 AND A PDP-11 INTERFACE CARD [M7846].
   NOTE: A DISKETTE MUST BE INCLUDED WITH EACH DRIVE TESTED.
C. CONSOLE TELEPRINTER

1.2.2 SOFTWARE REQUIREMENTS

NO PREREQUISITE SOFTWARE

2.0 OPERATING INSTRUCTIONS

2.0.1 OUTLINE OF OPERATING PROCEDURE

THE STANDARD RUNNING PROCEDURE FOR THE DIAGNOSTIC (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 IS BEING LOADED FROM A DISKETTE REPLACE THE "LIBRARY" DISKETTE WITH A "SCRATCH" DISKETTE.

NOTE: IF THIS STEP IS FORGOTTEN AND THE PROGRAM WAS LOADED VIA RXDP (FLOPPY MONITOR) ON UNIT 0 WITH UNIT 0 SELECTED BY USER TO UNDERGO TESTING THE PROGRAM WILL FAILSAFE THE OPERATION AND PROMPT THE USER AS FOLLOWS: "CAUTION - IF YOU DESIRE TO TEST UNIT 0 REPLACE LOAD MEDIUM WITH A SCRATCH DISKETTE THEN PRESS CONTINUE"

CAUTION AGAIN, HOWEVER ----------
NOTE 1) WHEN RUNNING THIS PROGRAM ON A SMALL 11 (E.G. /04, LSI 11, ETC.) WHERE THERE IS NO CONSOLE SWITCH REGISTER IT IS IMPERATIVE TO REMEMBER THIS SETUP.

NOTE 2) BEFORE PROCEEDING TO STEP B. ENSURE THAT THE FOLLOWING MODIFIABLE LOCATIONS CONTAIN THE PARAMETERS YOU REQUIRE FOR TESTING. THE FOLLOWING TABLE DESCRIBES EACH LOCATION WITH RESPECT TO THE DEFAULT PARAMETERS WHICH WILL BE USED IF LEFT UNMODIFIED BY THE USER:

LOCATION	LABEL	CONTENTS	PROGRAM REACTION
1200	0D:	0	TRACKS 0,52,53,114(8)
1202	FIRST:	015001	SECTORS 1 THRU 32(8)
1204	KRXVEC:	264	ASSUMES PROPER DEVICE VECTOR
1206	RXCS:	177170	ASSUMES PROPER DEVICE STATUS REGISTER (CALCULATES 'RXDB' ADDRESS FROM)
1212	DTESTP:	0	TESTS BOTH UNITS AUTOMATICALLY SEQUENCES THRU ALL TESTS
1214	BRLEV:	5	ASSUMES PROPER DEVICE 'BR' LEVEL

REFERENCE SECTION 2 OF THIS DOCUMENT FOR A MORE THOROUGH DESCRIPTION OF EACH OF THESE ITEMS AND HOW TO MODIFY THESE LOCATIONS IF YOU DESIRE TO CHANGE THE ABOVE MENTIONED DEFAULT TESTING PARAMETERS.

B. START THE PROGRAM AT LOCATION 200

C. THE PROGRAM WILL TYPE OUT MAINDEC NUMBER,A TEST PARAMETER OF 0
(USE BOTH DRIVES AND RUN ALL TESTS). THEN TYPE
TRACKS TO BE ACCESSED AND SECTOR LIMITS.
THE PROGRAM IS NOW RUNNING ALL TESTS IN SEQUENCE.

D. IF THERE ARE NO ERRORS , AT THE END OF THE PASS
(APPROX. 50 SECONDS RUN TIME), A "D" WILL BE TYPED
AND IT WILL CONTINUE ON FOR ANOTHER PASS.

E. TO HALT THE TEST AT ANY TIME (AFTER OR BEFORE
COMPLETION OF A PASS) JUST HALT THE PROCESSOR.

F. AFTER COMPLETING A PASS OF THE DIAGNOSTIC, THE
RX11 RELIABILITY TEST MAY BE RUN.

G. THERE ARE TWO TYPES OF ERROR PRINT OUT FORMATS
1. TESTS PRETEST, 1 - 17, AND 21 - 23 USE THE
FORMAT SHOWN IN SECTION 3.1. THE IMPORTANT
ADDRESS THERE IS THE "ERADR" (ERROR ADDRESS)
GO TO THE LISTING AT THAT LOCATION TO
GET MORE INFORMATION ON THE ERROR CONDITION
2. TEST 20, AND 24 - 26 USE THE FORMATT SHOWN
IN SECTION 3.3. IN THIS CASE THE "TEST PC"
IS THE ADDRESS OF THE TEST BEING RUN WHEN
THE ERROR OCCURED. THEN THE VITAL INFORMATION
OF THE ERROR IS PRINTED (CONTENTS OF ALL
REGISTERS, ADDRESS OF WHERE ON THE DISKETTE
THE ERROR OCCURED, AND THE TYPE OF ERROR).

2.1 LOADING THE PROGRAM

LOAD THE PROGRAM INTO MEMORY USING THE STANDARD PROCEDURE FOR
BINARY PAPER TAPES.
MAKE SURE THE TOTAL SYSTEM IS READY FOR OPERATION. THE DISKETTES
INSERTED PROPERLY, DOORS CLOSED ON DRIVES TO BE TESTED ETC.

2.2 STARTING ADDRESSES

THE PROGRAM HAS TWO STARTING ADDRESS LOCATIONS AS FOLLOWS:

2.2.1 INITIAL START [LOC.200]

THIS STARTING ADDRESS TESTS FOR AND SELECTS THE HARDWARE, OR SOFTWARE SWITCH REGISTER, PRINTS MAINDEC NAME AND REVISION, THE TEST AND DRIVE SELECTION, AND TRACKS AND SECTORS BEING USED.

2.2.2 RESTART [LOC.202]

THIS STARTING ADDRESS DIRECTS THE PROGRAM TO CONTINUE RUNNING USING THE DRIVE AND TEST SELECTIONS SPECIFIED IN THE PREVIOUS INITIAL START.

2.3 OPERATOR ACTION BEFORE STARTING THE PROGRAM

2.3.1 DEVICE ADDRESS SELECTION

LIKE MOST OPTIONS ON THE PDP-11 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 REGISTER AND VECTOR ADDRESSES THAT ALL COMMUNICATION BETWEEN THE PDP-11 AND THE RX11 IS HANDLED.

IF THE RX11 SYSTEM UNDER TEST IS JUMPERED FOR REGISTER ADDRESSES OTHER THAN STANDARD, WHICH IS RXCS = 177170 AND RXDB = 177172 PLACE IN THE MEMORY LOCATION CALLED "RXCS" [LOC. 1206] ITS NEW ADDRESS. THE PROGRAM ASSUMES THE NEXT EVEN ADDRESS ABOVE THAT OF RXCS, WILL BE THE ADDRESS OF RXDB, SO SETTING THAT ADDRESS IS NOT NECESSARY. IF THERE IS A NONSTANDARD INTERRUPT VECTOR ADDRESS (STANDARD IS LOC. 264) THEN PLACE IN MEMORY LOCATION CALLED "KRXVEC" [LOC. 1204] ITS NEW ADDRESS.

IF EITHER OF THESE LOCATIONS IS LOADED WITH A WRONG ADDRESS, THE PROGRAM WILL GET UNPREDICTABLE ERRORS AND MAY HALT.

NOTE: THE PROGRAM EXPECTS THAT THE PRIORITY LEVEL JUMPERS ARE SET FOR A NORMAL 'BR' LEVEL OF 5 (CONTENTS OF PROGRAM LOCATION 'BRLEV:' IS SET TO 5). IF THE PRIORITY LEVEL JUMPERS ARE SET TO ANY OTHER LEVEL TESTS 3 & 4 WILL REPORT ERRORS, UNLESS PROGRAM LOCATION 'BRLEV:' HAS BEEN PATCHED TO CONTAIN THE RELEVENT 'BR' LEVEL BEFORE EXECUTING THE PROGRAM.

IF THIS IS BEING TESTED ON A LSI 11, TESTS 3 AND 4 WILL NOT BE RUN AS THE LSI 11 HAS ONLY 1 LEVEL OF INTERRUPT.

2.3.3 NON-STANDARD DISKETTE ADDRESS SELECTION

SEQ 0008

IF IT IS DESIRABLE TO TEST THE DISKETTE BETWEEN TRACK AND SECTOR ADDRESS LIMITS OTHER THAN THE PRESELECTED TRACK ADDRESSES, AND/OR MINIMUM (FIRST) AND MAXIMUM (LAST) SECTOR ADDRESSES. THIS IS DONE BY THE OPERATOR MAKING CHANGES TO TWO MEMORY LOCATIONS BEFORE THE PROGRAM IS STARTED. ONE LOCATION IS CALLED "OD" [LOC. 1200] WHICH CONTAINS THE TWO BYTES FOR INNER AND OUTER TRACK ADDRESSES. THE OTHER LOCATION IS CALLED "FIRST" AND IT CONTAINS THE TWO BYTES FOR THE FIRST AND LAST SECTOR ADDRESSES.

A. DEFINITIONS

OD = ADDRESS OF TRACK AT OUTER DIAMETER (MIN. 0)
ID = ADDRESS OF TRACK AT INNER DIAMETER (MAX. 114)
FIRST = ADDRESS OF FIRST SECTOR ON A TRACK (MIN. 1)
LAST = ADDRESS OF LAST SECTOR ON A TRACK (MAX. 32)

B. LOCATIONS

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

C. RESTRICTIONS

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

IF THESE LOCATIONS ARE CHANGED TO NEW LIMITS, THEN THE PROGRAM WILL ACCESS ONLY THOSE ADDRESSES INCLUSIVE OF AND BETWEEN THESE LIMITS. THE EXCEPTION TO THIS IS TEST 26 WHICH ALWAYS USES A SPECIAL TRACK SEQUENCE.

IF THE "OD" LOCATION IS CLEARED OR SET TO ANY ILLEGAL COMBINATION OF TRACKS, THE PROGRAM WILL CLEAR LOCATION "OD". THE TRACK SEQUENCE WILL THEN BE TRACKS 0, 52, 53, AND 114 (OCTAL) ONLY.

IF THE "FIRST" LOCATION IS CLEARED OR SET TO ANY ILLEGAL COMBINATION OF SECTOR ADDRESSE LIMITS THEN THE PROGRAM WILL SET "FIRST" TO 1 AND "LAST" TO 32 (OCTAL).

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, OTHER WISE 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 CORRISPONDING 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 DELECTION MUST BE MADE BEFORE THE PROGRAM STARTS.
LOCATION "DTESTP" (LOC. 1212) IS WHERE THE BITS ARE SET TO TELL THE
PROGRAM WHAT DRIVES ARE WANTED AND WHAT TESTS TO RUN AS INDICATED BELOW.
WHEN THE PROGRAM STARTS IT WILL PRINT OUT THE CONDITIONS UNDER WHICH IT IS RUNNING.

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

NOTE: IF NEITHER OF THE ABOVE BITS ARE SET TO A 1, THEN
THE PROGRAM EXPECTS BOTH DRIVES TO BE READY FOR OPERATION (POWER
ON, DISKETTES INSERTED, AND DOORS CLOSED).

THEN SET THE TEST SELECTION IN BITS 4,3,2,1,AND 0 AS FOLLOWS:

"DTESTP" BITS 15 14 13-----5 4 3 2 1 0
U1 U0 NOT USED TESTS

BITS 4	3	2	1	0	TESTS
0	0	0	0	0	(IF NO TEST SELECTED DEFAULTS TO TEST 1)
0	0	0	0	1	TEST 1
0	0	0	1	0	TEST 2
0	0	0	1	1	TEST 3
0	0	1	0	0	TEST 4
0	0	1	0	1	TEST 5
0	0	1	1	0	TEST 6
0	0	1	1	1	TEST 7
0	1	0	0	0	TEST 10
0	1	0	0	1	TEST 11
0	1	0	1	0	TEST 12
0	1	0	1	1	TEST 13
0	1	1	0	0	TEST 14
0	1	1	0	1	TEST 15
0	1	1	1	0	TEST 16
0	1	1	1	1	TEST 17
1	0	0	0	0	TEST 20
1	0	0	0	1	TEST 21
1	0	0	1	0	TEST 22
1	0	0	1	1	TEST 23
1	0	1	0	0	TEST 24
1	0	1	0	1	TEST 25
1	0	1	1	0	TEST 26

NOTE1: SELECTION OF TESTS 27 THROUGH 37 WILL CAUSE THE MESSAGE
"ILLEGAL TEST" TO BE PRINTED.

NOTE2: WHEN A SPECIFIED TEST IS SELECTED THE PROGRAM WILL START AT
THAT TEST AND THEN RUN THROUGH ALL THE FOLLOWING TESTS UNTIL
IT COMPLETES TEST 26, INDICATED BY THE EOP TYPE OUT.
THEN IT WILL GO BACK TO THE TEST SELECTED AND START THE
NEXT PASS. (IE. IF TEST 24 IS SELECTED THE PROGRAM WILL RN TEST 24,
25, AND 26, THEN GO BACK TO TEST 24.)

AN EXPANDED DEFINITION OF THE TESTS IS IN SECTION 2.5

2.3 4.: PREREQUISITE OF TESTS:

THE FOLLOWING TESTS MUST BE RUN IN ORDER, AS ONE TEST SETS UP FOR THE NEXT TEST.

TEST 6 BEFORE TESTS 7 AND TEST 10
TEST 14 BEFORE TEST 15 AND TEST 16
TEST 16 BEFORE TEST 17
TEST 21 BEFORE TEST 22 AND TEST 23

SEE SECTION 2.5 UNDER THE ABOVE TESTS FOR EXPLANATION

2.4 OPERATOR ACTION TO RUN THE PROGRAM

2.4.1 STARTING THE PROGRAM

DEPENDING UPON THE STARTING ADDRESS SELECTED THE PROGRAM WILL DO THE FOLLOWING:

SA200 (INITIAL START)

THE SELECTION OF HARDWARE OR SOFTWARE SWITCH REGISTER IS MADE THEN THE PROGRAM WILL TYPE ITS IDENTIFICATION NUMBER,THE TEST PARAMETERS SELECTED IN LOCATION "DTESTP" ,AND TRACKS AND SECTORS BEING TESTED. THE PROGRAM THEN PROCEEDS TO RUN UNDER THOSE CONDITIONS.

SA202 (RESTART)

THE PROGRAM WILL TYPE OUT THE TEST PARAMETERS SELECTED BY THE PREVIOUS INITIAL START, PRINTS THE DISKETTE ADDRESS LIMITS, AND STARTS RUNNING THE TESTS. THE ONLY OPERATOR ACTION REQUIRED IS TO SET THE OPERATING CONDITIONS AS DEFINED IN SECTION 2.4.2, AFTER DEPRESSING THE "LOAD ADRS" SWITCH AND BEFORE DEPRESSING THE START SWITCH.

2.4.2 OPERATING CONDITIONS

AFTER THE TEST SELECTION HAS BEEN MADE PRESS THE "CONT" SWITCH.
THE PROGRAM WILL THEN ASK FOR OPERATING CONDITIONS. SWITCHES
0 AND 8 THROUGH 15 ARE USED AS INDICATED BELOW. ONCE THEY ARE
SET UP AGAIN DEPRESS THE "CONT" SWITCH. THE PROGRAM IS NOW RUNNING
UNDER THE SELECTED CONDITIONS.

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.

SW15 (1) - HALT ON ERROR

THE PROGRAM HALTS ON DETECTING AN ERROR, AFTER PRINTING THE
ERROR MESSAGE. PRESSING "CONT" RESTORES THE NORMAL OPERATION OF
THE PROGRAM.

SW14 (1) - HALT AT END OF PASS

AT "END OF PASS" THE PROGRAM TYPES A BELL THEN AN EOP INDICATOR.

"D" MEANS NO ERRORS DURING THE PASS
"-" MEANS HAD ERRORS DURING THE PASS

IF SW14 IS SET THE PROGRAM WILL HALT,IF SW14 IS OFF THE PROGRAM
GOES BACK TO THE TEST SELECTED AND RECYCLES THROUGH TO THE LAST
TEST, AT WHICH TIME ANOTHER EOP INDICATOR IS PRINTED.
IF THE PROGRAM HALTS DUE TO SW14 THEN PRESS "CONT" WILL RESTORE
THE NORMAL FLOW OF THE PROGRAM. IF IT HALTS AT THE END OF A
PASS IT WILL TYPE OUT THE NUMBER OF PASSES COMPLETED.

SW13 (1) - INHIBIT ERROR TYPEOUT

AT THE DETECTION OF AN ERROR IF SW13 IS SET NO ERROR PRINT OUT WILL OCCUR.
IF SW13 IS OFF THE ERROR INFORMATION IS PRINTED AS DESCRIBED
IN SECTION 3.0 ERROR DETECTION

SW12 (1) - LOOP ON TEST

AT THE COMPLETION OF A TEST THE PROGRAM CHECKS SW12. IF SET THE
PROGRAM WILL GO BACK TO THE BEGINNING OF THAT TEST AND RERUN IT.
THIS PRODUCES A SCOPE LOOP ON A PARTICULAR TEST.THE
PROGRAM WILL STAY IN THIS TEST UNTIL;
A. HALT ON END OF TEST SWITCH IS SET
B. LOOP ON TEST SWITCH IS TURNED OFF
AT WHICH TIME THE PROGRAM WILL GO ON TO THE NEXT TEST.

NOTE: IF SW12 IS SET AND NO TEST SPECIFIED (0) THE PROGRAM
WILL LOOP ON TEST 1.

NOTE: TO LOOP ON A TEST THAT REQUIRES A PREVIOUS TEST TO BE RUN
FIRST (SECTION 2.3.4). SELECT THE PREREQUISITE TEST AND SET
THE "HALT AT END OF TEST" SWITCH. START THE PROGRAM AND
WHEN IT HALTS,SELECT THE DESIRED TEST AND SET THE "LOOP ON
TEST" SWITCH. THE PROGRAM WILL NOW STAY IN THAT TEST

SW11 1) - LOCK ON ERROR

IN SOME TESTS ERRORS CAN OCCUR IN SEVERAL PLACES THROUGH OUT THE TEST. WHEN THE ERROR HAS BEEN REPORTED THE PROGRAM SETS A PC FLAG TO INDICATE WHERE THE ERROR OCCURED. IF SW11 IS SET THE PROGRAM GOES BACK TO THE BEGINNING OF THE TEST RUNNING, AND GOES THROUGH THE TEST UNTIL:

A. IT FINDS A DIFFERENT ERROR IN AN EARLIER PART OF THE TEST IN WHICH CASE IT WILL LOCK ONTO THAT ERROR.

B. IT DETECTS THE PC FLAG INDICATING THIS IS WHERE THE ERROR OCCURED. IT THEN GOES BACK TO THE BEGINNING OF THE TEST AGAIN.

THIS LOOP WILL CONTINUE UNTIL HALT ON ERROR SWITCH IS SET OR THE LOCK ON ERROR SWITCH IS TURNED OFF.

SW10 (1) - HALT AT END OF TEST

WHEN SET IT WILL HALT THE PROGRAM AT THE END OF THE TEST PRESENTLY RUNNING.

SW 9 - LIMIT DATA ERROR PRINT OUTS

(0) - WHEN OFF ONLY THE FIRST 10 DATA BYTE ERRORS WILL BE PRINTED ON A READ CHECK TEST, FOR EACH SECTOR. ANY MORE ERRORS WILL BE TABULATED BUT NOT PRINTED. AN ERROR ON A DIFFERENT SECTOR WILL ALLOW 10 MORE DATA BYTE ERRORS TO BE PRINTED.

(1) - WHEN SET ALL DATA BYTE ERRORS FOR ALL SECTORS WILL BE PRINTED ON AN ERROR.

SW 8 (1) - INHIBIT RECALIBRATION

NO RECALIBRATION OF THE DRIVES WILL OCCUR UPON THE DETECTION OF A SEEK ERROR IF THIS SWITCH IS SET.

SW 0 (1) - INHIBIT BELL AT ERROR

IF SW0 IS OFF THE ERROR ROUTINE WILL RING THE TELEPRINTER BELL AT EACH ERROR DETECTED. WITH SW0 SET NO BELL WILL RING.

2.5 TEST DEFINITIONS

2.5.1 PRETEST - INITIALIZE [KEY] PART I

EACH TIME THE PROGRAM IS STARTED, BY EITHER STARTING ADDRESS, IT RUNS THROUGH A PRETEST.

KEY INITIALIZE SHOULD SET THE DONE FLAG BECAUSE ANY INITIALIZATION OF THE RX01 MICROPROCESSOR IS AN IMPLIED [READ SECTOR] OF TRACK 1 SECTOR 1. THEREFORE ANY ERROR,EXCEPT PARITY,THAT MAY OCCUR FROM A NORMAL [READ SECTOR] COMMAND MAY OCCUR DURING AN INITIALIZE, CAUSING THE ERROR FLAG TO SET.

PRETEST INSURES THAT:

A. DONE IS SET

B. ERROR FLAG IS CLEARED

C. TR FLAG IS CLEARED

D. INIT DONE IS SET

2.5.2 TEST 1 - RXCS TEST PART I / INTERRUPT TEST PART I
THE PURPOSE OF THIS TEST IS TO VERIFY THAT WRITING ALL RXCS WRITABLE BITS TO A 0 ARE NOT WRITTEN TO A 1. SEQ 0014

THE PROGRAM WRITES THE RXCS = 0

NO INTERRUPTS SHOULD OCCUR

THE RXCS SHOULD REMAIN UNCHANGED = 40 (DONE)

THE RXDB SHOULD = 0

2.5.3 TEST 2 - INTERRUPT TEST PART II / VECTOR ADDRESS VERIFICATION

THE PURPOSE OF THIS TEST IS TO VERIFY THAT WRITING THE RXCS INTERRUPT ENABLE BIT (BIT 6) TO A 1 , DOES INDEED WRITE IT TO A 1, THEREFORE BECAUSE DONE IS SET AN INTERRUPT SHOULD OCCUR (THE PDP 11 PRIORITY IS 0)

2.5.4 TEST 3 - INTERRUPT TEST PART III / PRIORITY LEVEL TEST PART I

THE PURPOSE OF THIS TEST IS TO VERIFY THE PRIORITY OF THE INTERRUPT REQUEST LINE. THE PROGRAM SETS THE PDP-11 PRIORITY TO 4

AN RX01 INTERRUPT SHOULD OCCUR ON PRIORITY LEVEL 5

IF NO INTERRUPT OCCURS THEN THE PRIORITY LEVEL OF THE RX11 IS NOT 5, BUT MAYBE LEVELS 4,3,2,OR 1

2.5.5 TEST 4 - INTERRUPT TEST PART IV / PRIORITY TEST PART II

THE PURPOSE OF THIS TEST IS TO VERIFY THE PRIORITY OF THE RX11 INTERRUPT REQUEST LINE. THE PROGRAM SETS THE PDP-11 PRIORITY TO 5.

NO INTERRUPT SHOULD OCCUR
IF AN INTERRUPT DOES OCCUR THEN THE PRIORITY LEVEL OF THE RX11 IS NOT LEVEL 5,BUT MAYBE LEVEL 6, OR 7.

2.5.6 TEST 5 - INIT [PROGRAMMED] B / READ STATUS

THE PURPOSE OF THIS TEST IS TO VERIFY THAT SETTING THE RX11 BIT 14 CAUSES A RX01 PROGRAMMED SUBSYSTEM INITIALIZE

THE RXCS SHOULD = 40 (DONE)
THE RXDB SHOULD = 4, OR 104, OR 204, OR 304

TEST 5 CONT'D - RXCS TEST PART II / RST

THE PURPOSE OF THIS TEST IS TO VERIFY THE READ STATUS COMMAND (FUNCTION #12)., AND THAT DONE BIT IS CLEARED BY THE FUNCTION.

2.5.7 TEST 6 - FILL BUFFER TRANSFER LENGTH TEST

THE PURPOSE OF THIS TEST IS TO VERIFY THE TRANSFER LENGTH OF THE FUNCTION "FILL BUFFER" OF THE RX01 MICROCONTROLLER

NOTE: THIS TEST LOADS THE SECTOR BUFFER FOR TEST 7 AND 10, AND MUST BE RUN PREVIOUS TO THEM.

2.5.8 TEST 7 - EMPTY BUFFER TRANSFER LENGTH AND CONTENT VERIFICATION PART I SEQ 0015

THE PURPOSE OF THIS TEST IS TO VERIFY THE TRANSFER LENGTH
OF THE FUNCTION "EMPTY BUFFER" AND TO VERIFY THE CONTENTS OF
THE SECTOR BUFFER.

2.5.9 TEST 10 - EMPTY BUFFER TRANSFER LENGTH AND CONTENT VERIFICATION PART II

THE PURPOSE OF THIS TEST IS TO VERIFY THE PREVIOUS EMPTY BUFFER
TEST DID NOT EMPTY AND DESTROY THE CONTENTS OF THE SECTOR BUFFER.

2.5.10 TEST 11 - FILL / EMPTY BUFFER WITH ALL 0'S

DURING THE EMPTY BUFFER FUNCTION THIS TEST VERIFIES THAT ALL
0'S ARE IN FACT IN THE SECTOR BUFFER.

2.5.11 TEST 12 - FILL / EMPTY BUFFER WITH ALL 1'S

DURING THE EMPTY BUFFER FUNCTION THIS TEST VERIFIES THAT ALL
1'S ARE IN FACT IN THE SECTOR BUFFER.

2.5.12 TEST 13 - DRIVE READY VERIFICATION

TESTS THAT THE DRIVE READY (RDY) BIT WILL SET FOR ALL SELECTED DRIVES.
THE RDY BIT WILL BE SET AFTER A READ STATUS FUNCTION DIRECTED TO
THE SELECTED DRIVE.

2.5.13 TEST 14 - ERROR FLAG AND B-CODE VERIFICATION PART I

THE PURPOSE OF THIS TEST IS TO VERIFY THAT TRYING TO READ A
NON-EXISTANT SECTOR WILL CAUSE AN ERROR AND THE CORRECT ERROR
CODE WILL BE PUT INTO THE RXDB WHEN THE STATUS B IS READ.
NOTE: THIS TEST CHECKS FOR PARITY ERROR ON THE READ STATUS B
FUNCTION, THE NEXT TWO TESTS (T15 & T16) DO NOT. THIS TEST MUST BE RUN
BEFORE TESTS 15 & 16.

2.5.14 TEST 15 - ERROR FLAG AND B-CODE VERIFICATION PART II

THIS TEST VERIFIES THAT TRYING TO WRITE DELETED DATA ON AN
ILLEGAL SECTOR WILL PRODUCE AN ERROR AND THE CORRECT B-CODE
IS PRODUCED. THE DELETED DATA BIT SHOULD BE SET AFTER THIS TEST.

2.5.15 TEST 16 - ERROR FLAG AND B-CODE VERIFICATION PART III

VERIFIES THAT A WRITE FUNCTION TO A NONEXISTANT SECTOR WILL
PRODUCE AN ERROR AND THE CORRECT B-CODE IS PRODUCED. THE DELETED
DATA BIT WILL ALSO BE CLEARED.
NOTE: TEST 16 MUST BE RUN BEFORE TEST 17 AS TEST 16 CLEARS THE
DELETED DATA BIT AND TEST 17 TESTS THAT IT IS CLEARED.

2.5.16 TEST 17 - ILLEGAL TRACK ERROR VERIFICATION

THIS TEST VERIFIES THAT IF A TRACK ADDRESS LARGER THAN 114(OCTAL) IS ACCESSED, AN ERROR CONDITION WILL OCCUR, AND THE B-CODE WILL = 40. IT ALSO EXPECTS THE DELETED DATA BIT TO BE CLEARED.

2.5.17 TEST 20 - SEEK VERIFICATION VIA READ FUNCTION

THIS TEST DOES A READ FUNCTION ON THE SELECTED TRACKS TESTING FOR SEEK ERRORS ON VARIOUS SECTIONS OF THE DISKETTE.

2.5.18 TEST 21 - WRITE TEST

THE PURPOSE OF THIS TEST IS TO WRITE ALL ONES ON SECTOR 1, TRACK 1, AND TO VERIFY THAT THE DATA IN THE SECTOR BUFFER IS NOT CHANGED.
NOTE: THIS TEST MUST BE RUN BEFORE TESTS 22 & 23 AS THEY CHECK FOR DATA WRITTEN ON TRACK 1 SECTOR 1.

2.5.19 TEST 22 - INITIALIZE IMPLIED READ

AFTER PREVIOUSLY WRITING DATA ON TRACK 1 SECTOR 1, THIS TEST CHANGES THE CONTENTS OF THE SECTOR BUFFER AND DOES A PROGRAMMED INITIALIZE. AT THE END OF AN INIT.(RECAL.) THE SECTOR BUFFER MUST CONTAIN THE DATA FROM TRACK 1 SECTOR 1.
NOTE: UNIT 0 MUST BE ON-LINE FOR THIS TEST TO WORK.

2.5.20 TEST 23 - READ TEST

THIS TEST VERIFIES THAT A READ FUNCTION DOES INFACT LOAD THE SECTOR BUFFER WITH DATA READ FROM THE SELECTED ADDRESS.

2.5.21 TEST 24 - DATA TRANSFER AND VERIFICATION

THE PURPOSE OF THIS TEST IS TO WRITE THEN READ AND CHECK DATA ON ALL SECTORS OF THE SELECTED TRACKS. THE TEST ALTERNATES BETWEEN DRIVES, IF BOTH DRIVES ARE SELECTED, BEFORE CHANGING TRACKS.
THE DATA PATTERN USED IS A FLOATING 0 PATTERN.

2.5.22 TEST 25 - DATA VERIFICATION VIA DELETED DATA MODE.

THIS TEST IS THE SAME AS TEST 24 EXCEPT IT CHECKS FOR DELETED DATA INDICATORS AND USES A DATA PATTERN OF FLOATING 1.

2.5.23 TEST 26 - HEAD "HOME" TEST

THIS TEST CHECKS FOR THE "HOME FOUND BEFORE THE DESIRED TRACK WAS REACHED" ERROR CODE. THE HEAD IS MOVED OUT 10 TRACKS THEN DECREMENTED BACK TO TRACK 0. IT TESTS ALL SELECTED DRIVES, AND USES A DATA PATTERN OF RANDOM DATA.

3.0 ERRORS

PRETEST AND TESTS 1 - 17, AND TESTS 21 - 23 HANDLE ERRORS AS INDICATED IN SECTION 3.1. FOR THE MOST PART THESE TESTS DO NOT RELY ON AN INTERRUPT TO INDICATE THE FUNCTION IS COMPLETED. WHEREAS THE OTHER TESTS (TESTS 20, AND 24 - 26) DO READ, WRITE AND READ CHECK FUNCTIONS OVER THE SELECTED TRACK,SECTORS, AND DRIVES. THESE REQUIRE THE INTERRUPT SERVICE AND ERROR DETECTION THAT WAS USED IN THE DATA RELIABILITY TEST. THIS IS DESCRIBED IN SECTION 3.3.
NOTE: IF LOOP ON ERROR SWITCH IS UP THEN THE PROGRAM WILL LOOP ON THE SHORTEST SET OF INSTRUCTIONS THAT WILL KEEP IT IN THE FAILING LOOP. OTHERWISE AFTER REPORTING THE ERROR THE PROGRAM WILL CONTINUE RUNNING THROUGH THE REMAINING ADDRESSES AND TESTS.

3.1 ERROR HEADING FOR TESTS 1 - 17, AND 21 - 23 PLUS PRETEST.

THE ERROR HEADING IS AS FOLLOWS:

ERADR FAST FAPT [BLANK] GOOD BAD

UNDER EACH COLUMN THE ERROR ROUTINE PRINTS PERTINENT INFORMATION.

ERADR = ERROR ADDRESS
ADDRESS OF THE ERROR TRAP INSTRUCTION WHERE THE ERROR WAS DETECTED.

FAST = FIRST ADDRESS OF SELECTED TEST
ADDRESS OF THE TEST SELECTED AND RUNNING

FAPT = FIRST ADDRESS OF PRESENT TEST
ADDRESS OF THE TEST OR SUBTEST PRESENTLY RUNNING, OR ADDRESS OF THE SCOPE LOOP.

[BLANK]
ADDITIONAL GENERAL INFORMATION SUPPLIED BY SOME TESTS ON AN ERROR.

GOOD = EXPECTED RESULTS OF THE TEST
TEST RESULTS OF WHAT SHOULD HAVE HAPPENED IF THERE WAS NO ERROR.

BAD = ACTUAL TEST RESULTS
THE DATA THAT WAS RECIEVED FROM THE RX01, THAT CAUSED THE ERROR.

PASS = NUMBER OF PASSES MADE UP TO THIS ERROR

3.2 ERROR OUTPUT PER TEST

THE FOLLOWING ARE THE TYPES OF PRINT OUTS UNDER THE COLUMNS [BLANK], GOOD, AND BAD FOR THE VARIOUS TESTS, USING THIS ERROR FORMAT.

TEST (SECTION)	[BLANK] (R2)	GOOD (R0)	BAD (R1)
PRETEST (1)	N/A	40	(RXCS)
PRETEST (2)	(RXCS) INCL.DD BIT	4 OR 204	(RXCS) NO DD BIT
TEST 1 (1)	N/A	40	(RXCS)
TEST 1 (2)	N/A	0	(RXCS)
TEST 1 (3)	(KRXVEC)	N/A	N/A
TEST 2 (1)	(KRXVEC)	N/A	N/A
TEST 2 (2)	(KRXVEC)	140	(RXCS)
TEST 2 (3)	(KRXVEC)	40	(RXCS)
TEST 2 (4)	(KRXVEC)	40	(RXCS)
TEST 2 (5)	(KRXVEC)	40	(RXCS)
TEST 3 (1)	(KRXVEC)	N/A	N/A
TEST 4 (1)	(KRXVEC)	N/A	N/A
TEST 5 (1)	N/A	40	(RXCS)
TEST 5 (2)	(RXDB) INCL. DD BIT	4 OR 204	(RXDB) NO DD BIT
TEST 5 (3)	N/A	0	(RXCS)
TEST 5 (4)	N/A	40	(RXCS)
TEST 5 (5)	(RXCS) INCL. DD BIT	200	(RXCS) NO DD BIT
TEST 6 (1)	NO. OF XFERS	N/A	N/A
TEST 7 (1)	NO. OF XFERS	EXPEC. DATA	ACTUAL DATA
TEST 10 (1)	NO. OF XFERS	EXPEC. DATA	ACTUAL DATA

TEST 11&12 (1)	[USES TEST 6&7 TC FILL / EMPTY BUFFER]		
TEST 13 (1)	(RXDB)	200	(RXDB) NO DD BIT
TEST 13 (2)	(RXDB)	200	(RXDB) NO DD BIT
TEST 14 (1)	NO. OF TR'S	100040	(RXCS)
TEST 14 (2)	(RXDB)	0	(RXDB) NO DD BIT
TEST 14 (3)	(RXDB)	40	(RXCS)
TEST 14 (4)	N/A	70	(RXDB) ERROR CODE
TEST 15 (1)	NO. OF TR'S	100040	(RXCS)
TEST 15 (2)	N/A	100	(RXDB)
TEST 15 (3)	N/A	70	(RXDB) ERROR CODE
TEST 16 (1)	NO. OF TR'S	100040	(RXCS)
TEST 16 (2)	N/A	0	(RXDB)
TEST 16 (3)	N/A	70	(RXDB) ERROR CODE
TEST 17 (1A)	(RXDB)	0	(RXCS)
TEST 17 (1B)	N/A	100040	(RXCS)
TEST 17 (2)	N/A	0	(RXDB)
TEST 17 (3)	(RXDB)	40	(RXCS)
TEST 17 (4)	N/A	40	(RXDB) ERROR CODE
TEST 21 (1)	(RXES) STATUS A	NO. OF BYTE	(RXDB) STATUS B
TEST 21 (2)	[USES TEST 7 TO EMPTY BUFFER]		
TEST 22	[USES TEST 6 & 7 TO FILL AND EMPTY BUFFER]		
TEST 23	[USES TEST 6 & 21 TC FILL AND CHECK BUFFER]		

3.3 ERROR HEADING FOR TESTS 20, 24 - 26

AS PREVIOUSLY STATED THESE TESTS ACCESS ALL THE SELECTED SECTORS, TRACKS, AND DRIVES, AND RELY ON THE INTERRUPT SERVICE ROUTINE TO INDICATE THAT A FUNCTION IS COMPLETED OR AN ERROR OCCURED. ALL ERRORS, WITH THE EXCEPTIONS WHERE NOTED, WILL TYPE AS ITS FIRST OR SECOND LINE OF THE MESSAGE "ERROR CONDITIONS TEST PC = XXXX PASS = X ".
THE TEST PC NUMBER IS THE STARTING ADDRESS OF THE TEST RUNNING, AND THE PASS NUMBER IS THE NUMBER OF PASSES MADE UP TO THE ERROR

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 WAS DETECTED. IT SHOWS THE CRC, PAR, ETC. ERRORS

STATUS B IS THE "DEFINITIVE ERROR CODES" THAT THE RXD1 DETECTED, THAT MAY HAVE CAUSED THE ERROR CONDITION. THESE ERROR CODES ARE DEFINED IN SECTION 3.3.4

THERE ARE THREE CATEGORIES OF ERRORS AS LISTED AND DESCRIBED BELOW.

3.3.1 NO ERROR FLAG ERRORS

THESE ARE ERRORS THAT CAN OCCUR BUT THE ERROR FLAG IN THE RXCS WILL NOT BE SET.

A. UNEXPECTED OR MISSING DELETED DATA BIT

THIS ERROR RESULTS WHEN THE PROGRAM EXPECTS AND DOESN'T SEE THE DD BIT ("D D MARK MISSING"), OR DOESN'T EXPECT AND FINDS THE DELETED DATA BIT SET ("UNEXPECTED D D MARK").
THE PROGRAM WILL TYPE OUT AT WHAT DISKETTE ADDRESS THIS OCCURED THEN CONTINUE TESTING.
NOTE: SEE SECTION 3.3.3 FOR OTHER CAUSES OF THIS ERROR.

B. DATA NO STATUS ERROR

THIS ERROR OCCURS DURING A READ CHECK WHEN THE DATA READ DOES NOT MATCH THE DATA IN THE MEMORY DATA BUFFER, AND THERE WAS NO CRC ERROR INDICATED. THIS MEANS THAT THE DATA WAS PROBABLY READ OFF THE DISKETTE CORRECTLY BUT THE TRANSFER BETWEEN THE SECTOR BUFFER AND THE RXDB IN THE RX11 PRODUCED BAD DATA.

THE ERROR MESSAGE WILL INCLUDE THE DISKETTE ADDRESS, "BYTE" NUMBER IN THE SECTOR, THE DATA READ FROM THE SECTOR BUFFER "BAD", AND THE EXPECTED DATA FROM THE MEMORY BUFFER "GOOD".

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

0 (TRACK ADDRESS; BITS 6 - 0)
1 (UNIT NUMBER BIT 7)
(SECTOR ADDRESS BITS 4 - 0)

BYTES 2 - 125 CONTAIN THE SELECTED DATA PATTERN.

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

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

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

IF SWITCH 9 IS DOWN THEN ONLY 10 DATA ERRORS WILL BE PRINTED, AND AT THE END OF THE SECTOR THE "TOTAL READ CHECK ERRORS =" WILL BE TYPED. IF SWITCH 9 IS UP THEN ALL THE DATA ERRORS FOR THAT SECTOR WILL BE TYPED OUT.

C. 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 "SYSMAC" SUBROUTINE ".$POWER". 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 RESTARTS.

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

C. UNKNOWN INTERRUPT

IF AN INTERRUPT OCCURS THROUGH THE RX11 INTERRUPT VECTOR ADDRESS AND NONE OF THE STATUS BITS ARE SET (DONE, ERROR, ETC.) THE PROGRAM WILL TYPE "UNKNOWN INTERRUPT" AND RETURN BACK TO THE PROGRAM TO CONTINUE THE FUNCTION.
THE ERROR HEADING IS NOT PRINTED.

E. NO INTERRUPT AT DONE

THE PROGRAM EXPECTS AN INTERRUPT AT DONE ON THE FUNCTIONS OF THESE TESTS. IF AN INTERRUPT DOES NOT OCCUR AT DONE TIME THEN THE PROGRAM WILL TYPE OUT "NO INTERRUPT AT DONE ERROR" THEN GO INTO THE INTERRUPT SERVICE ROUTINE AS IF AN INTERRUPT DID OCCUR. AT THIS POINT OTHER ERRORS MAY BE PRINTED IF ANY ARE DETECTED.

3.3.2 ERROR FLAG ERRORS

THESE ERRORS ARE DETECTED AS THE RESULTS OF THE ERROR BIT BEING SET IN THE RXCS AT AN INTERRUPT.

A. PARITY ERROR

A PARITY ERROR RESULTS FROM AN INCORRECT TRANSFER OF A COMMAND WORD FROM THE RX11 INTERFACE TO THE RX01 MICRO-PROCESSOR CONTROLLER. THE PROGRAM WILL TYPE OUT THE CONTENTS OF THE COMMAND STATUS REGISTER (RXCS) SHOWING THE FUNCTION THAT FAILED, THE ADDRESS OF THE ERROR, CONTENTS OF STATUS A (RXES) WITH THE PARITY BIT SET, CONTENTS OF STATUS B (RXDB) WITH THE DEFINITIVE ERROR CODE OF 210 SET. THEN A "READ, WRITE, FILL BUFFER OR EMPTY BUFFER PARITY ERROR" WILL BE PRINTED.
IF A PARITY ERROR OCCURS ON A "READ DEFINITIVE ERROR CODE" FUNCTION. THEN THE CONTENTS OF THE RXCS AND "PARITY ERROR" WILL BE TYPED OUT.

B. CRC ERRORS

ON ALL DATA TRANSFERS BETWEEN THE SECTOR BUFFER AND THE DISKETTE, A CRC WORD IS GENERATED AND CHECKED. IF AN ERROR IS DETECTED BY THE MICRO-PROCESSOR IN THIS CRC WORD THEN A CRC ERROR IS GENERATED.
THE PROGRAM AGAIN TYPES OUT THE CONTENTS OF THE REGISTERS (RXCS CONTAINS FUNCTION, STATUS A WITH "CRC ERR" BIT SET, STATUS B WITH AN ERROR CODE OF 200). THEN IF IT IS A READ ONLY FUNCTION, OR A READ CHECK FUNCTION AND THERE WERE DATA ERRORS IT WILL TYPE OUT "DATA CRC ERRORS" THEN PRINT THE BAD BYTES IF ANY. IF IT WAS A READ CHECK FUNCTION AND THERE WERE NO DATA ERRORS IT WILL PRINT "CRC ERROR NO DATA ERROR".

C. SEEK ERRORS

ANY ERROR THAT PRODUCES A DEFINITIVE ERROR CODE BUT DOES NOT SET AN ERROR BIT IN STATUS A (RXDB AT END OF FUNCTION) IS LABELED A SEEK ERROR. SEE SECTION 3.3.4 FOR ERROR CODES AND MEANINGS.
THE SAME INFORMATION IS PRINTED FOR THESE ERRORS AS IN PARITY, OR CRC ERRORS, EXCEPT IT STATES THAT IT IS A "WRITE OR READ SEEK ERROR".
IF SWITCH 8 IS DOWN THEN AT EACH SEEK ERROR FOUND THE PROGRAM DOES AN INITIALIZE OF THE RX01 SO IT WILL RECALIBRATE TO A KNOWN (HOME) POSITION. THE PROGRAM THEN GOES ON TO THE NEXT SECTOR OR TRACK AND CONTINUES TESTING, IF THE LOOP ON ERROR SWITCH IS OFF. (SEE SECTION 3.3.3 FOR ERRORS CAUSED BY PREVIOUS ERRORS.) IF THE LOOP ON ERROR SWITCH IS UP IT WILL RETRY THE FUNCTION AT THE SAME ADDRESS.
IF SWITCH 8 IS UP THEN NO "INITIALIZE" IS DONE AND THE PROGRAM LOOKS AT THE OTHER SWITCHES FOR OPERATING CONDITIONS.
SEEK ERRORS ALSO PRINT THE TRACK ADDRESS THAT THE HEAD MOVED FROM AT THE TIME OF THE ERROR.

D. ERROR FLAG ERROR

IF THE ERROR FLAG IS NOT SET IN THE RXCS AND AN ERROR BIT IS SET IN STATUS A OR AN ERROR CODE IS SET IN STATUS B THEN THERE WAS AN ERROR BUT THE ERROR FLAG WAS NOT SET. THE MESSAGE "ERROR FLAG ERROR" IS PRINTED THEN THE PROGRAM CONTINUES TO TYPE OUT THE TYPE OF ERROR.

3.3.3 ERRORS RESULTING FROM PREVIOUS ERRORS

IF THERE IS A "WRITE SEEK ERROR" THE PROGRAM WILL GO ON TO THE NEXT ADDRESS WITHOUT WRITING ON THE ADDRESS WHERE THE ERROR OCCURED. (UNLESS THE LOOP ON ERROR SWITCH 11 IS UP AND THE SEEK ERROR IS RECOVERED.) IF THE WRITE FUNCTION IS FOLLOWED BY A READ CHECK FUNCTION AND THE READ DOES NOT HAVE A SEEK ERROR AT THE SAME ADDRESS. THEN THERE MAY BE DATA ERRORS, OR UNEXPECTED OR MISSING DELETED DATA BIT ERRORS RESULTING FROM NO DATA BEING WRITTEN ON THAT ADDRESS BY THE PREVIOUS WRITE FUNCTION.

3.3.4 DEFINITIVE ERROR CODES

THE RX01 MICRO-PROCESSOR HAS DEFINED THE ERROR CODES AND MEANINGS
WHICH ARE AVAILABLE TO THE PROGRAM BY ISSUING COMMAND #7
"READ DEFINITIVE ERROR CODE"
THE FOLLOWING ARE THE CODES AND THEIR MEANINGS

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 FOR INIT.
40 - TRIED TO ACCESS A TRACK GREATER THEN 76
50 - HOME FOUND BEFORE DESIRED TRACK WAS REACHED
60 - SELF DIAGNOSTIC ERROR
70 - DESIRED SECTOR NOT FOUND AFTER SAMPLING 52 HEADERS
100 - WRITE PROTECT ERROR
110 - MORE THEN 40 US 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 A HEADER, NO ERROR FLAG
150 - GOOD HEADER (NO CRC ERROR) BUT TRACK COMPARE ERROR
160 - ID ADDRESS MARK NOT FOUND IN TIME
170 - DATA MARK NOT FOUND IN TIME
200 - DATA CRC ERROR
210 - PARITY ERRORS

3.4 PROGRAM HUNG

IF THERE IS NO RESPONSE FROM THE RX11 WHILE WAITING FOR THE
TRANSFER REQUEST (TR) FLAG OR THE DONE FLAG. THE
PROGRAM WILL TYPE "DEVICE TEST HUNG @ PC" (ONLY IF SW13 IS OFF)
AND THEN GO ON TO THE NEXT TEST, OR THE BEGINNING OF THE PRESENT
TEST.

4.0 HALTS

THE ONLY HALTS IN THE PROGRAM ARE THE SELECTABLE HALTS (EOP,EOT,
AT ERROR), THE ILLEGAL VECTOR HALTS, AND THE ILLEGAL TEST SELECTION HALT.

NOTE: ONE ADDITIONAL 'HALT' EXISTS IN THE PROGRAM.
IT OCCURS WHEN THE USER HAS LOADED HIS PROGRAM
VIA THE 'RXDP' MONITOR (ON UNIT 0) AND ALSO
REQUIRES TESTING OF UNIT 0. A PROMPT MESSAGE
IS TYPED REMINDING THE USER TO REPLACE HIS LOAD
MEDIUM WITH A SCRATCH DISKETTE BEFORE GOING ON.
THE PROGRAM WILL WAIT FOR THE 'CONTINUE' SWITCH
TO BE DEPRESSED.

5.0 FLOW CHARTS

```
                                        .NLIST CND,MD,MC
                                        .LIST ME
                                        .ENABL ABS,AMA



                        .TITLE  MAINDEC-11-DZRXB-E
                        ;*COPYRIGHT (C) MARCH 21,1976
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY D. ADAMS/B. BURGESS
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                        ;*
000001                  $TN=1
160000                  $SWR=160000     ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYPOUT



                                ;COPYRIGHT (C) 1975,1976
                                ;THIS SOFTWARE IS FURNISHED UNDER LICENCE 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 LICENCE TERMS. TITLE TO 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 CORPORATION.

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

```
                        ;MODIFIED TO REV. D BY B. BURGESS NOV. 10, 1975 AS FOLLOWS:
                        ;
                        ;A)     ADDED CAPABILITY OF VARIABLE DEVICE 'BR' LEVEL.  ALL RELEVANT TESTS
                        ;       CALCULATE 'CPU' LEVEL BASED ON CURRENT CONTENTS OF LOCATION 'BRLEV:'.
                        ;       DEFAULT 'BR' LEVEL, FOR THE DEVICE, SET BY THE PROGRAM IS 5.  ANY OTHER
                        ;       'BR' LEVEL ( E.G. 6 ) WOULD HAVE TO BE PATCHED INTO LOCATION 'BRLEV:'
                        ;       BEFORE RUNNING THE PROGRAM.
                        ;
                        ;B)     ADDED TWO (2) ROUTINES TO HANDLE 'UNEXPECTED' BUS TIMEOUT AND
                        ;       RESERVED INSTRUCTION TRAPS ( TRAPS TO VECTORS 4 & 10, RESPECTIVELY).
                        ;       BOTH ROUTINES WILL INDICATE WHICH TRAP OCCURRED, THE 'PC' LOCATION
                        ;       OF WHERE THE TRAP OCCURRED, AND ATTEMPT TO RESTART THE PROGRAM.
                        ;
                        ;C)     ADDED CODE TO FAILSAFE UNIT 0 UNDERGOING TESTING IF PROGRAM WAS
                        ;       LOADED VIA UNIT 0 USING 'RXDP' MONITOR AND USER STARTED RUNNING
                        ;       THE PROGRAM WITHOUT HAVING REPLACED HIS LOAD MEDIUM WITH A 'SCRATCH'
                        ;       DISKETTE.
                        ;
                        ;D)     ADDED MESSAGES TO INDICATE TO USER WHEN HE HAS SELECTED TRACK AND/OR
                        ;       SECTOR LIMITS 'OUT OF RANGE' AND CORRESPONDING DEFAULT LIMITS WHEN
                        ;       THIS CONDITION ARISES
                        ;
                        ;E)     MODIFIED TESTS 1 THRU 4 TO CORRECTLY PRINT OUT THE CONTENTS OF
                        ;       'KRXVEC' ( LOCATION HOLDING THE DEVICE VECTOR) AS 264 INSTEAD OF 270.
                        ;
                        ;F)     MODIFIED TEST 2 TO HANDLE A 'LOCKED IN INTERRUPT STATE' CONDITION
                        ;       ARISING WHEN 'INTERRUPT ENABLE' AND 'DONE' ARE BOTH QUALIFIED AND
                        ;       THE 'REQUEST INTERRUPT' FLOP NEVER GETS CLEARED.
                        ;
                        ;G)     ADDED EXTENSIVE MAINTENANCE INFORMATION BASED ON FAULT INSERTION
                        ;       RESULTS.  INFORMATION IS KEYED TO THE 'ERROR' REPORT WITHIN A
                        ;       TEST.  INFORMATION PROVIDED SHOULD BE SELF-EXPLANATORY BUT SHOULD
                        ;       NOT BE MISCONSTRUED AS BEING ALL ENCOMPASSING DUE TO HUMAN ERRORS
                        ;       IN STATISTICS GATHERING, INABILITY TO FAULT INSERT SOME CHIPS, AS
                        ;       WELL AS ONLY TWO (2) MODULES ABLE TO BE FAULT INSERTED I.E. -
                        ;       M7846 (UNIBUS INTERFACE) AND M7727 (READ/WRITE CONTROL).
                        ;
                        ;H.     ADDED FLOW CHARTS
                        ;
```

```
                        .SBTTL  BASIC DEFINITIONS

                        ;*INITIAL ADDRESS OF THE STACK POINTER *** 1200 ***
001200                  STACK=  1200
                        .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                        .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

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

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

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

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11=   4000
002000                  SW10=   2000
001000                  SW09=   1000
000400                  SW08=   400
000200                  SW07=   200
000100                  SW06=   100
000040                  SW05=   40
000020                  SW04=   20
000010                  SW03=   10
```

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

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

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

```
                                        ;SPECIAL EQUATES


                000017                  RDER    =17             ; READ B CODE
                000040                  DONEBIT =40
                000101                  FBIE    =101            ; IE+FILL BUFFER
                000103                  EBIE    =103            ; IE+EMPTY BUFFER
                000105                  WRTIE   =105            ; IE+WRITE SECTOR
                000107                  RDIE    =107            ; IE+READ SECTOR
                000115                  WTDDIE  =115            ; IE+WRITE DD SECTOR
                040001                  RECAL   =40001
                000000                  OPEN    =0

                000000                  .=0
000000  000000  000000                          .WORD 0,0

                000004                  .=4
000004  006130                  .WORD   BUSERR          ;UNEXPECTED TIMEOUT TRAP PC
000006  000340                  .WORD   PR7             ;UNEXPECTED TIMEOUT TRAP PS
000010  006154                  .WORD   RESERR          ;UNEXPECTED RESERVED INSTRUCTION TRAP PC
000012  000340                  .WORD   PR7             ;UNEXPECTED RESERVED INSTRUCTION TRAP PS

                000020                  .=20
000020  006476                  XSCOPE
000022  000340                  PR7
000024  015132                  $PWRDN
000026  000340                  PR7
000030  006232                  XERROR
000032  000340                  PR7
000034  015046                  $TRAP                   ;ADDRESS OF TRAP SERVICE
000036  000340                  PR7

                000046                  .=46
000046  002452                  LOGICAL                 ;ACT 11 EOP HOOKS

                000052                  .=52
000052  000000                  .WORD 0

                000174                  .=174
000174  000000                  DISPREG:        0
000176  000000                  SWREG:          0

                000200                  .=200
000200  000401                          BR 1$
000202  000402                          BR 2$
000204  000137  001232          1$:     JMP SA200               ;OPERATOR SELECTED CONDITIONS
000210  000137  001222          2$:     JMP SA202               ;RESTART PROGRAM WITH PREVIOUS PARAMETERS
```

```
                        .SBTTL TEST SELECTION VIA SWITCH REGISTER

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

                        ;SET TEST AND DRIVE SELECTION IN " DTESTP " LOCATION 1212
                        ;
                        ; BIT 15 = 1 - UNIT 1 SELECTED
                        ; BIT 14 = 1 - UNIT 0 SELECTED
                        ; BIT 15 & BIT 14 = 0 - BOTH DRIVES MUST BE READY
                        ;
                        ; BIT 4 - BIT 0 = OCTAL NUMBER OF DESIRED STARTING TEST
                        ; BIT 4 - BIT 0 = 0 -ALL TESTS WILL BE SEQUENCED THROUGH

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


                        .SBTTL OPERATIONAL SWITCH REGISTER POSITIONS

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

                ; SET OPERATING CONDITIONS IN THE SWITCH REGISTER (HARDWARE)
                ; OR SOFTWARE SWITCH REGISTER LOCATION 176

                ; 15 = 1 - HALT ON ERROR
                ; 14 = 1 - HALT AT END OF PASS
                ; 13 = 1 - INHIBIT ERROR TYPEOUT
                ; 12 = 1 - LOOP ON TEST
                ; 11 = 1 - LOCK ON ERROR
                ; 10 = 1 - HALT AT END OF TEST
                ;  9 = 1 - PRINT ALL DATA ERRORS
                ;  9 = 0 - PRINT ONLY FIRST 10 DATA ERRORS PER SECTOR
                ;  8 = 1 - INHIBIT RECALIBRATION ON SEEK ERRORS.
                ;
                ;  0 = 1 - INHIBIT <BELL> AT ERROR

                ; 15-0 = 1 - SELECT SOFTWARE SWITCH REGISTER

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

```
                        .SBTTL RXCS (RX COMMAND STATUS REGISTER)

                        .=STACK

001200  000000 OD:      0                       ;OD/ID = 0 UNLESS SPECIFIC TRACKS SELECTED.
        001201 ID=OD+1
001202  015001 FIRST:   015001                   ; FIRST = 1, LAST = 32
        001203 LAST=FIRST+1

001204  000264 KRXVEC: 264

001206  177170 RXCS:    177170

                        ; RXCS: STANDARD DEVICE ADDRESS = 177170

                        ; TOGGLE INTO PROGRAM LOCATION " RXCS " THE RX11 DEVICE ADDRESS IF NOT = 177170

                        ;KEY: R - READ ONLY BIT
                        ;     W - WRITE ONLY BIT

                        ;               15 - R  - ERROR
                        ;               14 - W  - INITIALIZE
                        ;               13 -
                        ;               12 -
                        ;               11 -    (BITS 13-8)
                        ;               10 -    (NOT USED)
                        ;                9 -
                        ;                8 -
                        ;                7 - R  - TRANSFER REQUEST
                        ;                6 - R/W- INTERRUPT ENABLE
                        ;                5 - R  - DONE
                        ;                4 - W  - UNIT SELECT
                        ;                3 - W  - FUNCTION
                        ;                2 - W  - FUNCTION
                        ;                1 - W  - FUNCTION
                        ;                0 - W  - GO !

                        ; FUNCTION

                        ; 3 2 1 0

                        ; - - - GO

                        ;   0 + GO -    FILL BUFFER
                        ;   2 + GO -    EMPTY BUFFER
                        ;   4 + GO -    WRITE SECTOR
                        ;   6 + GO -    READ SECTOR
                        ;   -
                        ;  12 + GO -    READ STATUS " A "
                        ;  14 + GO -    WRITE DELETED DATA
                        ;  16 + GO -    READ STATUS " B " (CODES)
```

```
                                .SBTTL RXDB (RX DATA BUFFER REGISTER)

001210  177172          RXDB:   177172

                        ; RXDB: STANDARD DEVICE ADRESS = 177172

                        ; THE FOLLOWING BIT IDENTIFICATION REPRESENTS THE STATUS AT THE END OF A FUNCTION
                        ; (BUT NOT FUNCTION # 16 TO READ STATUS " B ") DISPLAYED WITHIN THE RX-DATA BUFFER.

                        ;       (A)     7 - SELECTED DRIVE READY
                        ;               6 - DELETED DATA
                        ;               5 -
                        ;               4 -
                        ;               3 - WRITE PROTECT ERROR
                        ;       (B)     2 - INITIALIZE DONE
                        ;               1 - PARITY ERROR
                        ;               0 - CRC ERROR

                        ; (A) - VISIBLE ONLY IF THE FUNCTION WAS # 12 READ STATUS " A "
                        ;
                        ; (B) - INIT DONE VISIBLE IF AN INITIZLIAE [KEY] OR [PROGRAMMED] WAS ISSUED

001212  000000          DTESTP: 0

001214  000005          BRLEV:  5

                        ;BRLEV: STANDARD PRIORITY INTERRUPT LEVEL = 5

                        ;TOGGLE INTO PROGRAM LOCATION "BRLEV" THE RX11 INTERRUPT PRIORITY
                        ;LEVEL IF NOT = 5

001216  177570          SWR:            .WORD DSWR
001220  177570          DISPLAY:        .WORD DDISP

                        ; R0 - GOOD     /EXPECTED RESULT OF TEST
                        ; R1 - EAC      /ACTUAL RESULT OF TEST
                        ; R2 - BLANK    /
                        ; R3 - TEST Q

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

                        ;WORD "UNITSEL" HAS THE FOLLOWING BIT DEFFINITIONS
                        ;
                        ;BIT15 = 1 - UNIT 1 SELECTED FOR USE
                        ;BIT14 = 1 - UNIT 1 IN USE
                        ;BIT8  = 1 - THIS PASS HAD AN ERROR
                        ;BIT7  = 1 - UNIT 0 SELECTED FOR USE
                        ;BIT6  = 1 - UNIT 0 IN USE
                        ;BIT4  =     UNIT SELECTION BIT

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

```
                                        .SBTTL START AND RESTART ADDRESSES

                                ; THE STARTING ADDRESS WAS 202

001222  005200                  SA202:  INC R0
001224  012706  001200                  MOV #STACK,SP
001230  000447                          BR RESTART

                                ; THE STARTING ADDRESS WAS 200

001232  005000                  SA200:  CLR R0
001234  012737  177570  001216          MOV #177570,SWR                 ;RESET TO HARDWARE SWR.
001242  012706  001200                  MOV #STACK,SP
001246  104401  016720                  TYPE ,MREV                      ;PRINT THE NAME AND REVISION
001252  013746  000004                  MOV 4,-(SP)                     ;SAVE 'BUSERR' TIMEOUT 'PC'
001256  012737  001276  000004          MOV #1$,4                       ;SET UP TIMEOUT VECTOR
001264  022777  177777  177724          CMP #177777,@SWR                ;IS SOFTWARE SWR SELECTED
001272  001402                          BEQ 2$                          ;YES, INSERT IT'S ADDRESS
001274  000423                          BR 3$                           ;BR IF NO TIMEOUT TRAP OCCURS
001276  022626                  1$:     CMP (SP)+,(SP)+                 ;RESTORE THE STACK
001300  012737  000176  001216  2$:     MOV #SWREG,SWR                  ;POINT TO SOFTWARE SWITCH REGISTER
001306  012737  000174  001220          MOV #DISPREG,DISPLAY            ;POINT TO SOFTWARE DISPLAY REG.
                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
001314  005737  000042                  TST     @#42                    ;;ARE WE RUNNING UNDER XXDP/ACT?
001320  001006                          BNE     64$                     ;;BRANCH IF YES
001322  023727  001216  000176          CMP     SWR,#SWREG              ;;SOFTWARE SWITCH REG SELECTED?
001330  001005                          BNE     65$                     ;;BRANCH IF NO
001332  104405                          GTSWR                           ;;GET SOFT-SWR SETTINGS
001334  000403                          BR      65$
001336  112737  000001  015044  64$:     MOVB    #1,$AUTOB               ;;SET AUTO-MODE INDICATOR
001344                          65$:
001344  012637  000004          3$:     MOV (SP)+,4                             ;RESET TIMEOUT VECTOR TO 'BUSERR'
001350  000005                  RESTART:RESET                                   ;INITIALIZE THE RX11 SYSTEM
001352  012746  000340                  MOV #PR7,-(SP)
001356  012746  001364                  MOV #4$,-(SP)
001362  000002                          RTI                                     ;LOAD THE PSW
001364  013737  001206  001210  4$:     MOV RXCS, RXDB                          ;GET ADDRESS OF RXCS
001372  062737  000002  001210          ADD #2, RXDB                            ;SET UP ADDRESS OF RXDB
001400  012737  001234  012626          MOV #001234,RAN1                        ; INITIALIZE CONSTANTS OF
001406  012737  000765  012630          MOV #000765,RAN2                        ; RANDOM NUMBER GENERATOR
001414  005037  002526                  CLR CCOUNT
001420  005037  002530                  CLR PASS
001424  005037  006756                  CLR HANGER
001430  012737  177740  006760          MOV #177740,HANGPL
001436  005700                          TST R0
001440  001055                          BNE XSA202                              ; STARTING ADDRESS WAS 202
001442  005037  012724                  CLR UNITSEL
001446  032737  140000  001212          BIT #140000,DTESTP                      ;WERE ANY DRIVES SELECTED
001454  001004                          BNE 1$                                  ;YES GO SET THEIR BITS
001456  052737  100200  012724          BIS #100200,UNITSEL                     ;NO, BOTH UNITS MUST BE READ
001464  000415                          BR 2$
001466  032737  040000  001212  1$:     BIT #BIT14,DTESTP                       ;WAS UNIT 0 SELECTED
001474  001434                          BEQ 3$                                  ;NO, MUST BE UNIT 1
001476  052737  000200  012724          BIS #200,UNITSEL                        ;YES,SET SELECTED BIT
001504  005737  001212                  TST DTESTP                              ;WAS UNIT 1 SELECTED
001510  100003                          BPL 2$                                  ;NO
```

```
001512  052737  100000  012724          BIS #BIT15,UNITSEL              ;YES,SET THE SELECTED BIT
001520  123727  000041  000010  2$:     CMPB    41,#10                  ;WAS PROGRAM LOADED IN DUMP MODE
                                                                        ;VIA XXDP?
001526  001022                          BNE     XSA202                  ;BRANCH IF NOT
001530  005737  000042                  TST @#42                        ;CHECK FOR RXDP OPERATION
001534  001410                          BEQ 5$
001536  042737  000200  012724          BIC #200,UNITSEL                ;IN CHAIN MODE, DESELECT UNIT 0
001544  104401  016207                  TYPE ,MUNIT1
001550  104401  016217                  TYPE ,MONLY
001554  000407                          BR XSA202                       ;AND DO NOT HALT
001556  104401  017242          5$:     TYPE,   DOLOAD                  ;INFORM USER TO REMOVE LOAD MEDIUM
                                                                        ;FROM UNIT 0 AND REPLACE WITH
                                                                        ;A 'SCRATCH' DISKETTE IF HE
                                                                        ;WISHES TO TEST UNIT 0
001562  000000                          HALT                            ;WAIT FOR USER RESPONSE
001564  000403                          BR XSA202
001566  052737  100000  012724  3$:     BIS #BIT15,UNITSEL              ;SET SELECTED BIT
001574  042737  100200  001200  XSA202: BIC     #100200,OD              ;CLEAR 1ST TIME BITS FOR BOTH DRIVES
001602  104401  015630                  TYPE, MDTESTP
001606  013746  001212                  MOV     DTESTP,-(SP)    ;;SAVE DTESTP FOR TYPEOUT
001612  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
001614     006                          .BYTE   6               ;;TYPE 6 DIGIT(S)
001615     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
001616  104401  016134                  TYPE ,MCRLF
001622  005737  001200          LIMITS: TST OD
001626  001005                          BNE TRKLMT
001630  005037  013136                  CLR SEQUEN
001634  104401  016433                  TYPE ,MLIMTRK
001640  000432                          BR SECLMT

                                ; 0 <= OD <= ID <= 114

001642  123727  001201  000114  TRKLMT: CMPB ID,#114
001650  101021                          BHI 1$
001652  123737  001200  001201          CMPB OD,ID
001660  101015                          BHI 1$
001662  104401  016461                  TYPE ,MOD
001666  113746  001200                  MOVB OD,-(SP)
001672  104403                          TYPOS
001674     003                          .BYTE  3
001675     000                          .BYTE  0
001676  104401  016465                  TYPE ,MID
001702  113746  001201                  MOVB ID,-(SP)
001706  104403                          TYPOS
001710     003                          .BYTE  3
001711     000                          .BYTE  0
001712  000405                          BR SECLMT
001714  104401  017067          1$:     TYPE,   OD2BIG                  ;TYPE MSG. INDICATING ID OR OD
                                                                        ;TOO BIG & DEFAULTING TO TRACKS
                                                                        ;0, 52, 53, 114
001720  005037  001200                  CLR OD
001724  000736                          BR LIMITS

                                ; 1 <= FIRST <= LAST <= 32

001726  105737  001202          SECLMT: TSTB FIRST
```

```
                001003                          BNE 1$
                112737  000001  001202         MOVB #1,FIRST
                123727  001203  000032  1$:    CMPB LAST,#32
                101023                          BHI 2$
                123737  001202  001203         CMPB FIRST,LAST
                101017                          BHI 2$
                104401  016473          3$:     TYPE ,MFIRST
                113746  001202                  MOVB FIRST,-(SP)
                104403                          TYPOS
                   003                          .BYTE  3
                   000                          .BYTE  0
                104401  016504                  TYPE ,MLAST
                113746  001203                  MOVB LAST,-(SP)
                104403                          TYPOS
                   003                          .BYTE  3
                   000                          .BYTE  0
                104401  016134                  TYPE ,MCRLF
                000406                          BR PRETEST
                104401  017154          2$:     TYPE,   S2BIG                   ;TYPE MSG. INDICATING THAT
                                                                                ;SECTOR RANGE SELECTED WAS
                                                                                ;INVALID AND DEFAULTING TO A
                012737  015001  001202          MOV #15001,FIRST                ; 1ST SECTOR VALUE OF 1 AND
                000753                          BR 3$                           ;A LAST SECTOR VALUE OF 32
```

```
                        .SBTTL PRETEST - INITIALIZE [KEY] PART I
                ;
                ;       "KEY" INITIALIZE SHOULD HAVE [SET] THE DONE FLAG BECAUSE
                ;       ANY [INIT] OF THE RX01 MICROCONTROLLER IS AN IMPLIED [READ SECTOR]
                ;       OF TRACK 1 SECTOR 1 (FOR SYSTEMS PROGRAMMING BOOTSTRAP APPLICATIONS).
                ;
                ;       THEREFORE, ANY ERROR (EXCEPT PARITY) THAT MAY OCCUR FROM A NORMAL
                ;       "READ SECTOR" COMMAND MAY OCCUR HERE CAUSING THE ERROR FLAG TO SET, AND
                ;       DISPLAYING THE ERROR STATUS WITHIN THE TRANSFER REGISTER AT "DONE".
                ;
                ;       THE TRANSFER REQUEST FLAG SHOULD BE CLEARED.
                ;
                ;NOTE:  SCOPE LOOPING IS NOT OFFERED BECAUSE THE "INIT" FUNCTION
                ;       WHICH PRODUCED THE ERROR HAS NOT YET BEEN VERIFIED.
002034  042737  000400  012724  PRETEST: BIC #BIT8,UNITSEL
002042  005037  006472                  CLR ERRORS
002046  000004                        SCOPE                   ; REFRESHES FAST FOR ERROR TYPEOUT
002050  013737  006556  002532        MOV PCSCOPE,FAST        ; FOR ERROR TYPEOUT INFORMATION
002056  012700  000040                MOV #40, R0             ; PROGRAM EXPECTS DONE FLAG
002062  017701  177120          3$:   MOV @ RXCS, R1          ; ACTUAL CONTENTS OF RXCS
002066  020100                        CMP R1, R0
002070  001407                        BEQ 1$                  ; OK
002072  062737  000001  006756        ADD #1,HANGER           ;WAIT FOR THE INIT. FUNCTION TO
002100  005537  006760                ADC HANGPL              ;FINISH BEFORE CALLING IT AN ERROR
002104  001366                        BNE 3$

                                ; (R0) = 40 ; (R1) = ACTUAL RXCS ; (R2) = N/A

002106  104000                        ERROR                   ; RXCS NOT = 40 (DONE)


;/*\:/*\:/*\:/*\:/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/* :
;/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

        ;THE FOLLOWING IS A PRESENTATION OF POSSIBLE REASONS AS TO WHY
        ;THIS ERROR REPORT WAS GENERATED.  THE INFORMATION SHOWN IS
        ;BASED ON FAULT INSERTION RESULTS,  AND SHOULD PROVIDE LOGICAL
        ;AREAS TO CHECK FOR THE RELEVANT FAULT/S.
        ;
        ;IF THIS ERROR REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS
        ;ANALYZE THE FOLLOWING AREA/S:
;
;M7846 (UNIBUS INTERFACE)
;
;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
;


;       NOTE:  MAKE SURE THE DRIVES ARE CONNECTED CORECTLY,THE
;               DISKETTES INSERTED,AND THE DOORS OF THE SELECTED DRIVES
;               ARE CLOSED.  IF THE THESE CONDITIONS ARE NOT SET THERE
;               WILL BE AN ERROR AT THIS POINT.

;B TRANSFER REQUEST     STUCK HIGH                      E22
```

```
        ;DONE                   STUCK LOW                       E22,E36,E1
        ;RUN(0) H               STUCK LOW                       E22
        ;SELECT DO              STUCK LOW                       E18,E34,E17,E40,E11
        ;B DONE                 STUCK LOW                       E18,E15,E19,E41
        ;RUN(1) H               STUCK HIGH                      E18
        ;RX INIT                STUCK HIGH                      E32
        ;OUT                    STUCK HIGH                      E23,E4
        ;B INIT                 STUCK HIGH                      E36,E8
        ;BUS -> RXCS            STUCK HIGH/LOW                  E36,E40
        ;BUS INTR               STUCK HIGH                      E38
        ;BUS D02/D04            STUCK LOW                       E38
        ;RUN(1) H               STUCK LOW                       E37
        ;INT ENB(1) H           STUCK LOW                       E37
        ;TRANSFER REQUEST       STUCK HIGH                      E1
        ;INT ENB(1) H           STUCK HIGH                      E1
        ; ----------            STROBE DISABLED                 E1
        ; ----------            'A' INPUTS NOT SELECTED         E1
        ;CMD                    STUCK LOW                       E17,E21
        ;B DONE                 STUCK HIGH                      E15
        ;RUN(0) H               STUCK HIGH                      E15
        ;OUT                    STUCK LOW                       E24
        ;RX INIT                STUCK LOW                       E21,E11
        ;IN                     STUCK HIGH                      E21
        ;RX RUN                 STUCK LOW                       E14
        ; ----------            LOCKED IN 'PRESET' STATE        E2
        ; ----------            NO STROBE SIGNAL                E3
        ;BUS D15                STUCK LOW                       E9
        ;DATA                   STUCK HIGH                      E11
        ;

                ;IF THE FAULT CANNOT BE FOUND ON THE UNIBUS INTERFACE MODULE
                ;AND/OR THE FAULT IS NOT INHERENT TO THE UNIBUS INTERFACE MODULE
                ;M7846 THERE IS A POSSIBILITY OF ITS EXISTENCE ON THE READ/WRITE
                ;MODULE M7727.
;
;               NOTE:   ONLY APPROX. 30% OF THIS MODULE LENT ITSELF CONDUCIVE TO
;                       THE FAULT INSERTION PROCESS; ERGO, THE RESOLUTION FOR FAULT
;                       ANALYSIS OBTAINABLE BY THIS MODULE IS NOT VERY HIGH.
;                       HOWEVER, ANALYSIS OF THE FOLLOWING AREA/S, IF THIS ERROR
;                       REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS, SHOULD AT
;                       LEAST PLACE YOU WITHIN THE RELEVANT AREA ON THE MODULE.
;
;M7727 (READ/WRITE CONTROL)
;
;               SIGNAL NAME             REASON                  POSSIBLE CHIPS
;
; ----------                    PIN 15 NOT AT GROUND            E15
;SEL TRK 0                      STUCK LOW                       E15
;DK0 TRK 0                      STUCK LOW                       E15
;SEL DK0                        STUCK LOW                       E13
;DC LO                          STUCK LOW                       E13
;OUT                            STUCK HIGH                      E13
; ----------                    E18 CLOCK LOCKED HIGH/LOW       E14,E18
; ----------                    REPLACE E18                      ----------
;INIT                           STUCK LOW                       E18,E16
; ----------                    PIN 14 NOT +5V THRU 1K          E16
```

```
        ; ----------          'J,K' INPUTS TO E18 LOCKED      E17
        ;                       HIGH/LOW
        ; ----------          'J' INPUT TO E16 LOCKED HIGH    E17
        ;
        ;/: */: */: */:`*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:`*
        ;/: */: */: */: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/: */:\*
        ;/: */: */: */: */: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:`*/:\*/:\*/:\*/: *
```

```
                        ;////////////////////////////////////////////////////////////////////////
                        ; STATUS A [RXDB] AT "DONE"
                        ;     7       6       -       -       3       2       1       0         /
                        ;
                        ;     SEL                                   WRITE   INIT    PAR     CRC       /
                        ;     DRIVE   DD                            PROTECT [DONE]                    /
                        ;     RDY                                                   (N/A)             /
                        ;
                        ;////////////////////////////////////////////////////////////////////////

002110  012737  177740  006760  1$:     MOV #177740,HANGPL              ;RESET HANG COUNTER
002116  012700  000204                   MOV #204,R0                     ;EXPECT INIT DONE AND UNIT 0 RDY
002122  017702  177062                   MOV @RXDB,R2
002126  010201                           MOV R2,R1
002130  042701  000100                   BIC #BIT6,R1                    ;CLEAR DELETED DATA BIT
002134  105737  012724                   TSTB UNITSEL                    ;WAS UNIT 0 SELECTED
002140  100404                           BMI 2$
002142  042701  000200                   BIC #BIT7,R1                    ;UNIT 0 WAS NOT SELECTED
002146  042700  000200                   BIC #BIT7,R0                    ;CLEAR THE DRIVE 0 RDY BITS
002152  020100                   2$:     CMP R1,R0
002154  001544                           BEQ REBEGIN

                        ; (R0) = 4 IF UNIT 0 IS NOT SELECTED, OR 204 IF UNIT 0 IS SELECTED
                        ; (R1) = ACTUAL RXDB MINUS DELETED DATA BIT#6
                        ; (R2) = ACTUAL RXDB

002156  104000                           ERROR                           ; RXDB NOT = 4, OR 204
```

```
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

                ;THE FOLLOWING IS A PRESENTATION OF POSSIBLE REASONS AS TO WHY
                ;THIS ERROR REPORT WAS GENERATED.  THE INFORMATION SHOWN IS
                ;BASED ON FAULT INSERTION RESULTS,  AND SHOULD PROVIDE LOGICAL
                ;AREAS TO CHECK FOR THE RELEVANT FAULT/S.
                ;
                ;IF THIS ERROR REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS
                ;ANALYZE THE FOLLOWING AREA/S:
        ;
        ;M7846 (UNIBUS INTERFACE)
        ;
        ;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
```

```
        ;


        ;       NOTE:  MAKE SURE THE DRIVES ARE CONNECTED CORECTLY,THE
        ;               DISKETTES INSERTED,AND THE DOORS OF THE SELECTED DRIVES
        ;               ARE CLOSED. IF THE THESE CONDITIONS ARE NOT SET THERE
        ;               WILL BE AN ERROR AT THIS POINT.

        ;B SHIFT                STUCK LOW/HIGH                  E15,E19,E2
        ;B DONE                 STUCK HIGH                      E15
        ;TO RXD1                STUCK LOW/HIGH                  E15
        ;LOAD                   STUCK HIGH                      E17,E18
        ;SELECT 00              STUCK HIGH                      E17,E18
        ;SELECT 02              STUCK LOW                       E17,E18,E34
        ;IN                     STUCK LOW                       E17,E21
        ;CMD                    STUCK LOW/HIGH                  E21
        ;BUS D15                STUCK HIGH                      E40
        ;RX INIT                STUCK HIGH                      E32
        ;BUS D00 -> D03         STUCK HIGH                      E41,E7
        ;DATA                   STUCK LOW                       E18,E11
        ; ----------            INCORRECT SHIFT OUTPUTS         E5
        ; ----------            CAN'T SELECT 'B' INPUTS         E1
        ; ----------            CAN'T RESET                     E3
        ;B INIT                 STUCK LOW                       E34,E8
        ;RX BUSY                STUCK LOW                       E34,E22,E19
        ;B DONE                 STUCK LOW                       E19
        ;BUS D05                STUCK LOW                       E4
        ;B SER DATA             STUCK LOW/HIGH                  E9
        ;INT ENB(1) H           STUCK HIGH                      E37
        ;

                ;IF THE FAULT CANNOT BE FOUND ON THE UNIBUS INTERFACE MODULE
                ;AND/OR THE FAULT IS NOT INHERENT TO THE UNIBUS INTERFACE MODULE
                ;M7846 THERE IS A POSSIBILITY OF ITS EXISTENCE ON THE READ/WRITE
                ;MODULE M7727.
        ;
        ;       NOTE:   ONLY APPROX. 30% OF THIS MODULE LENT ITSELF CONDUCIVE TO
        ;               THE FAULT INSERTION PROCESS; ERGO, THE RESOLUTION FOR FAULT
        ;               ANALYSIS OBTAINABLE BY THIS MODULE IS NOT VERY HIGH.
        ;               HOWEVER, ANALYSIS OF THE FOLLOWING AREA/S, IF THIS ERROR
        ;               REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS, SHOULD AT
        ;               LEAST PLACE YOU WITHIN THE RELEVANT AREA ON THE MODULE.
        ;
        ;M7727 (READ/WRITE CONTROL)
        ;
        ;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
        ;
        ;SEL WT PROT            STUCK LOW                       E4,E6,E15
        ;
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

751  002160  000542                        BR      REBEGIN
752  002162  004737  002556        MORETESTS: JSR PC, LOCKUP
753  002166  005723                TONOTHERE: TST (R3)+                    ;ADJUST R3 FOR NEXT TEST ADDRESS
```

```
                002170  016337  002214  006556  FIRSTTEST: MOV TESTS(R3), PCSCOPE       ; EQUIVALENT TO " SCOPE "
                002176  013737  006556  002532          MOV PCSCOPE,FAST                ;SAVE THE FIRST ADDRESS OF THE TEST
                002204  012706  001200                  MOV #STACK,SP
                002210  000173  002214                  JMP @TESTS(R3)
                002214  002166  002620  002740  TESTS:  TONOTHERE, T1, T2, T3, T4, T5, T6, T7
                002222  003262  003420  003564
                002230  004010  004142
                002234  004140  004226  004216          T10, T11, T12, T13, T14, T15, T16, T17
                002242  004272  004416  004610
                002250  004750  005216
                002254  005462  005506  005672          T20, T21, T22, T23, T24, T25, T26, NOMORETESTS
                002262  005750  006016  006050
                002270  006062  002274
                                                ; TEST 1 - RXCS TEST PART I / INTERRUPT TEST PART I
                                                ; TEST 2 - INTERRUPT TEST PART II / VECTOR ADDRESS VERIFICATION
                                                ; * TEST 3 - INTERRUPT TEST PART III / PRIORITY LEVEL VERIFICATION PART I
                                                ; * TEST 4 - INTERRUPT TEST PART IV / PRIORITY VERIFICATION PART II
                                                ; TEST 5 - INIT [PROGRAMMED] / RST
                                                ; TEST 6 - FILL BUFFER TRANSFER LENGTH VERIFICATION
                                                ; TEST 7 - EMPTY BUFFER TRANSFER LENGTH AND CONTENT VERIFICATION PART I
                                                ; TEST 10 - EMPTY BUFFER TRANSFER LENGTH AND CONTENT VERIFICATION PART II
                                                ; TEST 11 - FILL/EMPTY BUFFER ALL 0'S
                                                ; TEST 12 - FILL/EMPTY BUFFER ALL 1'S
                                                ; TEST 13 - DRIVE READY VERIFICATION FOR SELECTED DRIVES
                                                ; TEST 14 - ERROR FLAG AND B - CODE VERIFICATION PART I
                                                ; TEST 15 - ERROR FLAG AND B - CODE VERIFICATION PART II
                                                ;       /DELETED DATA BIT SETS
                                                ;TEST 16 - ERROR FLAG AND B - CODE VERIFICATION PART III
                                                ;       /DELETED DATA BIT CLEARS
                                                ;TEST 17 - ILLEGAL TRACK ERROR AND B - CODE VERIFICATION
                                                ;TEST 20 - SEEK VERIFICATION VIA READ FUNCTION
                                                ;TEST 21 - WRITE TEST
                                                ;TEST 22 - INITIALIZE IMPLIED READ
                                                ;TEST 23 - READ TEST
                                                ;TEST 24 - DATA TRANSFER & VERIFICATION
```

```
                        ;TEST 25 - DATA TRANSFER & VERIFICATION VIA DELETED DATA MODE
                        ;TEST 26 - HEAD "HOME" TEST
                        ;THERE ARE NO MORE TESTS
                        ; * NOTE: ON PROCESSORS WITHOUT HARDWARE PROCESSOR STATUS WORDS (PSW)
                        ;       THESE TEST WILL NOT BE RUN.
```

```
                                        ;PRINT AN END OF PASS INDICATOR
                                        ;
                                        ; C - RX11/RX01 TEST PASS OK
                                        ; D - RX11/RX01 AND DRIVE TESTING OK
                                        ; - - AN ERROR OCCURRED (DURING C OR D)
                                        ;
                                        ; NOTE: IF BIT 8 OF UNITSEL IS A 1
                                        ; THEN AN ERROR HAS OCCURRED FOR THIS PASS
                                        ;
002274  042777  000100  176704  NOMORETESTS:  BIC   #BIT6,@RXCS     ;CLEAR 'IE' BIT BEFORE NEXT PASS
002302  005037  006756                  CLR HANGER
002306  032737  000400  012724          BIT #BIT8,UNITSEL           ;"C" OR "D" MEANS ERRORLESS PASS.
002314  001403                          BEQ 1$
002316  012737  000055  002534          MOV #'-, MX                 ; " - " MEANS UN-ERRORLESS PASS
002324  005737  002526          1$:     TST CCOUNT
002330  001002                          BNE 3$
002332  104401  016134                  TYPE, MCRLF
002336  005237  002526          3$:     INC CCOUNT
002342  022737  000110  002526          CMP #72., CCOUNT
002350  001002                          BNE 4$
002352  005037  002526                  CLR CCOUNT
002356  104401  002534          4$:     TYPE, MX
002362  104401  006470                  TYPE ,MABELL
002366  005237  002530          2$:     INC PASS
002372  102775                          BVS 2$
002374  104406                          CKSWR
002376  032777  040000  176612          BIT #SW14,@SWR              ; AC SW 14 = 1 TO HALT AT END OF PASS
002404  001413                          BEQ 6$
002406  104401  016134                  TYPE ,MCRLF
002412  104401  006725                  TYPE ,MPASS
002416  013737  002530  002430          MOV PASS,5$
002424  004537  015614                  JSR R5,SGLDEC
002430  000000          5$:     OPEN
002432  000000                  HALT
002434  005237  006756          6$:     INC HANGER                  ;WAIT FOR EOP INDICATOR TO BE PRINTED
002440  001375                          BNE 6$
002442  013705  000042                  MOV @#42,R5                 ;ACT 11 END OF PASS HOOKS
002446  001405                          BEQ HERE
002450  000005                          RESET
002452  004715          LOGICAL: JSR PC,(R5)
002454  000240                  NOP
002456  000240                  NOP
002460  000240                  NOP
002462  000137  002466          HERE:   JMP REBEGIN
                                        ;
002466  042737  000400  012724  REBEGIN: BIC #BIT8,UNITSEL          ;CLEAR HARD ERROR INDICATOR
002474  013703  001212                  MOV DTESTP, R3
002500  042703  177740                  BIC #177740, R3             ; R3 CONTAINS TEST # 0 TO 26
002504  020327  000027                  CMP R3,#27
002510  103002                          BHIS 1$
002512  006303                          ASL R3
002514  000625                          BR FIRSTTEST
                                        ;
002516  104401  002536          1$:     TYPE ,MILTST
002522  000137  001232                  JMP SA200
                                        ;
```

F04


```
874  002526  000000                    CCOUNT: 0
875  002530  000000                    PASS:   0
876  002532  000000                    FAST:   0
877
878  002534  000103                    MX:     .ASCIZ "C"
879
880  002536  046111  042514  040507   MILTST: .ASCIZ "ILLEGAL TEST"<15><12>
881  002544  020114  042524  052123
882  002552  005015     000
883
884          002556                            .EVEN
885
```

```
                                        ; DATA SW 10 = 1 TO HALT AT END OF TEST
                                        ;
002556  104406                  LOCKUP: CKSWR
002560  032777  002000  176430          BIT #SW10,@SWR
002566  001401                          BEQ 1$
002570  000000                          HALT
                                        ;
                                        ; DATA SW 12 = 1 TO LOCK SCOPE LOOP ON TEST OK OR NOT
                                        ;
002572  032777  010000  176416  1$:     BIT #SW12,@SWR          ;IS LOOP ON TEST SWITCH SET
002600  001403                          BEQ 2$                  ;IF NOT SET GO ON TO NEXT TEST
002602  062716  000002                  ADD #2,@SP              ;IF SET RETURN TO FIRSTTEST
002606  000207                          RTS PC
002610  042737  040100  012724  2$:     BIC #40100,JNITSEL      ;CLEAR UNIT USED BITS
002616  000207                          RTS PC
```

```
                        .SBTTL TEST 1 - RXCS TEST PART I / INTERRUPT TEST PART I

                ; THE PURPOSE OF THIS TEST IS TO VERIFY THAT WRITING ALL RXCS WRITABLE BITS TO A 0
                ; ARE [NOT] WRITTEN TO A 1

                ;THE PROGRAM WRITES THE RXCS = 0

                ;NO INTERRUPTS SHOULD OCCUR

                ;THE RXCS SHOULD REMAIN UNCHANGED = 40 (DONE)

                ;THE RXDB SHOULD = 0

002620  005077  176362  T1:   CLR @ RXCS                   ; WRITING RXCS TO 0, ALSO WRITES RXDB = 0
002624  012700  000040        MOV #40, R0
002630  017701  176352        MOV @ RXCS, R1
002634  020001                CMP R0, R1
002636  001401                BEQ 1$

                ; (R0) = 40 ; (R1) = ACTUAL RXCS ; (R2) = N/A

002640  104000                ERROR                       ; RXCS NOT = 40 (DONE)

        ;/* :/*\:/*\:/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

                ;THE FOLLOWING IS A PRESENTATION OF POSSIBLE REASONS AS TO WHY
                ;THIS ERROR REPORT WAS GENERATED.  THE INFORMATION SHOWN IS
                ;BASED ON FAULT INSERTION RESULTS,  AND SHOULD PROVIDE LOGICAL
                ;AREAS TO CHECK FOR THE RELEVANT FAULT/S.
                ;
                ;IF THIS ERROR REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS
                ;ANALYZE THE FOLLOWING AREA/S:
        ;
        ;M7846 (UNIBUS INTERFACE)
        ;
        ;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
        ;
        ;BUS D06                 STUCK HIGH                      E4
        ; ----------             CAN'T SEND DATA                 E41
        ;B TRANSFER REQUEST      STUCK HIGH                      E15
        ;RX INIT                 STUCK LOW                       E36,E11
        ;BUS D15                 STUCK LOW                       E14
        ;DOO(1) H                STUCK HIGH                      E22
        ;RX RUN                  STUCK LOW                       E18
        ;BUS DOO                 STUCK LOW                       E7
        ;
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/: */: *
        ;/:\*/: */: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/: */: */: *
        ;/: */: */: */: */:\*/: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/: */: */:\* : *

002642  104413  1$:   SUBSCOPE
```

```
                        ;////////////////////////////////////////////////////////////
                        ; RXDB AT "DONE"
                        ;       7       6       -       -       3       2       1       0
                        ;
                        ;       SEL                                     WRITE   INIT    PAR
                        ;       DRIVE   DD                              PROTECT [DONE]          CRC
                        ;       RDY                                                     (N/A)
                        ;
                        ;////////////////////////////////////////////////////////////

002644  005000                        CLR R0
002646  017701  176336                MOV @ RXDB, R1
002652  020100                        CMP R1, R0
002654  001401                        BEQ 3$

                        ; (R0) = 0 ; (R1) = ACTUAL RXDB ; (R2) = N/A

002656  104000                        ERROR                           ; RXDB NOT = 0

        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*  :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

                ;THE FOLLOWING IS A PRESENTATION OF POSSIBLE REASONS AS TO WHY
                ;THIS ERROR REPORT WAS GENERATED.  THE INFORMATION SHOWN IS
                ;BASED ON FAULT INSERTION RESULTS,  AND SHOULD PROVIDE LOGICAL
                ;AREAS TO CHECK FOR THE RELEVANT FAULT/S.
                ;
                ;IF THIS ERROR REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS
                ;ANALYZE THE FOLLOWING AREA/S:
        ;
        ;M7846 (UNIBUS INTERFACE)
        ;
        ;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
        ;
        ;BUS D01 -> D03          STUCK HIGH                      E7
        ;LOAD                    STUCK LOW                       E17,E18
        ; ----------             NO SACK                         E12
        ; ----------             CLOCK STUCK HIGH/LOW            E12
        ;IN                      STUCK LOW/HIGH                  E21
        ;RX BUSY                 STUCK HIGH                      E22
        ;OUT                     STUCK LOW                       E22
        ;BUS D00/D02/D03         STUCK HIGH                      E4
        ;BUS -> RXCS             STUCK LOW                       E40
        ;REG SELECT              STUCK HIGH                      E23
        ;B DONE                  STUCK HIGH                      E19
        ; ----------             'B2' INPUT STUCK HIGH           E1
        ;
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/: */:\* : */: */: * : */:\*/: */:\*/:\*/:\*/:\*/:\*/:\*/: */:\*/:\* : */:\* : *
        ;/:\*/: */:\*/: */: */: */:\*/: */:\*/:\*/:\*/:\*/:\*/: */:\*/:\*/: */:\*/:\*/:\*

:012    002660  104413                3$:     SUBSCOPE
```

```
                                ; INTERRUPT TEST PART I ; NO INTERRUPTS SHOULD OCCUR

002662  013702  001204                 MOV KRXVEC, R2
002666  010246                         MOV     R2,-(SP)        ;SAVE INTERRUPT VECTOR FOR
                                                                ;ERROR REPORT
002670  012722  002730                 MOV #4$, (R2)+          ; RX01 VECTOR ADDRESS
002674  012722  000340                 MOV #PR7, (R2)+
002700  012602                         MOV     (SP)+,R2        ;RESTORE INTERRUPT VECTOR FOR
                                                                ;ERROR REPORT
002702  005046                         CLR -(SP)               ; PDP PRIORITY (ON)
002704  012746  002712                 MOV #2$,-(SP)
002710  000002                         RTI
002712  000240                 2$:     NOP
002714  000240                         NOP
002716  012746  000340                 MOV #PR7,-(SP)          ; RESET PDP PRIORITY (7) OFF
002722  012746  002732                 MOV #5$,-(SP)
002726  000002                         RTI

                                ; RETURN TO HERE WITH PDP PRIORITY = 7 IF AN RX01 INTERRUPT


                                ; (R0) = N/A ; (R1) = N/A ; (R2) = N/A

002730  104000                 4$:     ERROR                   ; UNEXPECTED RX01 INTERRUPT

        ;/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/* :/*  :/*\:/*\:/*\:/*  :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/* :/*\:/*\:/*  :/*\:/*  :/*  :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

                ;THE FOLLOWING IS A PRESENTATION OF POSSIBLE REASONS AS TO WHY
                ;THIS ERROR REPORT WAS GENERATED.  THE INFORMATION SHOWN IS
                ;BASED ON FAULT INSERTION RESULTS,  AND SHOULD PROVIDE LOGICAL
                ;AREAS TO CHECK FOR THE RELEVANT FAULT/S.
                ;
                ;IF THIS ERROR REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS
                ;ANALYZE THE FOLLOWING AREA/S:
        ;
        ;M7846 (UNIBUS INTERFACE)
        ;
        ;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
        ;
        ;INT ENB                 STUCK HIGH                      E36
        ;
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

1061  002732  000004                 5$:     SCOPE
1062  002734  000137  004246                 JMP CEXIT                       ;END OF TEST 1
```

```
1063                                    .SBTTL TEST 2 - INTERRUPT TEST PART II / VECTOR ADDRESS VERIFICATION
1064
1065                                    ; THE PURPOSE OF THIS TEST IS TO VERIFY THAT WRITING THE RXCS INTERRUPT ENABLE BI
1066                                    ; (BIT 6) = 1 DOES INDEED WRITE IT TO A 1, THEREFORE BECAUSE DONE = 1
1067                                    ; AN INTERRUPT SHOULD OCCUR (THE PDP PRIORITY IS 0 )
1068
1069  002740  013702  001204           T2:     MOV KRXVEC, R2
1070  002744  010246                           MOV R2,-(SP)                    ;SAVE INTERRUPT VECTOR FOR
1071                                                                           ;ERROR REPORT
1072  002746  012722  003016                   MOV #1$, (R2)+                  ; RX01 VECTOR ADDRESS
1073  002752  012722  000340                   MOV #PR7, (R2)+
1074  002756  012602                           MOV (SP)+,R2                    ;RESTORE INTERRUPT VECTOR FOR
1075                                                                           ;ERROR REPORT
1076  002760  005046                           CLR -(SP)                       ; PDP PRIORITY (ON)
1077  002762  012746  002770                   MOV #6$,-(SP)
1078  002766  000002                           RTI
1079  002770  052777  000100  176210  6$:     BIS #BIT6, @ RXCS               ; SET RX01 INTERRUPT ENABLE
1080  002776  000240                           NOP
1081

        ;/*`:/*\:/*.:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*.:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*`:
        ;/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

                ;THE FOLLOWING IS A PRESENTATION OF POSSIBLE REASONS AS TO WHY
                ;THIS ERROR REPORT WAS GENERATED.  THE INFORMATION SHOWN IS
                ;BASED ON FAULT INSERTION RESULTS,  AND SHOULD PROVIDE LOGICAL
                ;AREAS TO CHECK FOR THE RELEVANT FAULT/S.
                ;
                ;IF THIS ERROR REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS
                ;ANALYZE THE FOLLOWING AREA/S:

        ;M7846 (UNIBUS INTERFACE)
        ;
        ;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
        ;
        ; ----------            SACK FLOP LOCKED CLEAR          E13,E31,E36
        ; ----------            BBSY FLOP CLOCK LOCKED HIGH     E13,E31,E36,E33
        ;BUS REQUEST            STUCK HIGH                      E31
        ;BUS INTR               STUCK HIGH                      E31,E35,E8,E38
        ; ----------            SACK FLOP CLOCK LOCKED HIGH     E31,E21,E9,E12
        ; ----------            SACK FLOP CLOCK LOCKED LOW      E31,E21
        ; ----------            BBSY FLOP CLOCK LOCKED LOW      E31,E33,E9
        ;BUS D02/D04/D05        STUCK HIGH                      E38
        ;BUS SACK               STUCK HIGH                      E32
        ;BG OUT                 STUCK HIGH                      E32,E25,E28
        ;BUS REQUEST            STUCK LOW                       E39
        ; ----------            JUMPER (N1) NOT IN               ----------
        ;BUS D07                STUCK HIGH                      E35,E8
        ; ----------            GRANT FLOP CAN'T BE PRESET      E25,E9,E28
        ;B SER DATA             STUCK LOW                       E9
        ; ----------            GRANT FLOP 'Q' OUTPUT           E28
        ;                          STUCK HIGH

        ;`:*`:*:*`:*`:*`:*`:*`:*:*`:*`:*`:*`:*`:*:*`:*`:*`:*`:*`:*:*
        ;:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*:*
```

```
        ;/:\*/:\*/:\*/:\*/: */:\*/:\*/: */:\*/:\*/:\*/:\*/:\*/: */:\*/:\*/:\*

  1121  003000  000240                          NOP
  1122  003002  012746  000340                  MOV #PR7,-(SP)              ; RESET PDP PRIORITY (7) OFF
  1123  003006  012746  003014                  MOV #7$,-(SP)
  1124  003012  000002                          RTI
  1125
  1126                                  ; (R0) = N/A ; (R1) = N/A ; (R2) = N/A
  1127
  1128  003014  104000                  7$:     ERROR                       ; NO RX01 INTERRUPT OCCURRED
  1129

        ; * :/* :/*\:/*\:/*\:/* :/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ; * :/*\:/*\:/*\:/*\:/*\:/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

                ;THE FOLLOWING IS A PRESENTATION OF POSSIBLE REASONS AS TO WHY
                ;THIS ERROR REPORT WAS GENERATED.  THE INFORMATION SHOWN IS
                ;BASED ON FAULT INSERTION RESULTS,  AND SHOULD PROVIDE LOGICAL
                ;AREAS TO CHECK FOR THE RELEVANT FAULT/S.
                ;
                ;IF THIS ERROR REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS
                ;ANALYZE THE FOLLOWING AREA/S:
        ;
        ;M7846 (UNIBUS INTERFACE)
        ;
        ;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
        ;
        ; ----------            BBSY FLOP LOCKED IN 'RESET'     E40,E39,E31
        ; ----------            INT ENB FLOP CLOCK LOCKED HIGH  E40
        ; ----------            INT ENB FLOP CLOCK LOCKED LOW   E40,E37
        ;RX INIT                STUCK HIGH                      E36
        ;BUS REQUEST            STUCK HIGH                      E36,E39,E32
        ; ----------            LOCKED IN INTERRUPT STATE       E36,E39
        ;BG OUT                 STUCK HIGH                      E33
        ;INT ENB(1) H           STUCK LOW                       E37,E1,E4
        ; ----------            GRANT FLOP 'Q' OUTPUT           E28
        ;                         STUCK LOW
        ;TRANSFER REQUEST       STUCK LOW                       E4
        ;BUS INTR               STUCK LOW                       E39
        ;
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/: */: */:\*/:\*/:\*/:\*
        ;/:\*/:\*/: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

  1164
  1165                                  ; RETURN TO HERE WITH THE PDP PRIORITY = 7 IF AN RX01 INTERRUPT
  1166
  1167  003016  005737  003032          1$:     TST CATERR                  ;DID MORE THAN ONE INTERRUPT
  1168                                                                      ;OCCUR DUE TO BOTH 'DONE' & 'IE'
  1169                                                                      ;BITS SET BUT 'RQST INTR' FLOP
  1170                                                                      ;NEVER GETTING CLEARED?
  1171  003022  001004                          BNE     DEATH              ;BRANCH IF YES
  1172  003024  105237  003032                  INCB    CATERR             ;OTHERWISE, SET FLAG TO INDICATE
  1173                                                                      ;ONE INTERRUPT OCCURRED & GO ON
  1174  003030  000402                          BR      OK2GO
```

```
1175  003032  000000                CATERR: .WORD   0                       ;THIS IS THE INTERRUPT INDICATOR.
1176                                                                        ;1ST TIME THRU THIS TEST THE IN-
1177                                                                        ;DICATOR SHOULD ALWAYS BE ZERO;
1178                                                                        ;IF NON-ZERO A MULTIPLE OF
1179                                                                        ;INTERRUPTS HAS OCCURRED DUE TO
1180                                                                        ;THE AFOREMENTIONED CONDITION
1181  003034  000000                DEATH:  HALT                            ;NO SENSE GOING ON - YOU ARE
1182                                                                        ;LOCKED IN THE INTERRUPT STATE!
1183                                                                        ;PRESS THE 'CONT' SWITH (IF
1184                                                                        ;PRESENT) TO GO ON  AT YOUR OWN
1185                                                                        ;RISK
1186  003036  104413                OK2GO:  SUBSCOPE                        ;EVERYTHING APPEARS TO BE OK
1187  003040  005037  003032                CLR     CATERR                  ;CLEAR INTERRUPT INDICATOR FOR
1188                                                                        ;NEXT POSSIBLE PASS THRU THIS TEST
1189
1190                                ; THE RXCS SHOULD = 140 (INTERRUPT ENABLE+DONE)
1191
1192  003044  012700  000140                MOV #140, R0
1193  003050  020077  176132                CMP R0, @ RXCS
1194  003054  001403                        BEQ 2$
1195  003056  017701  176124                MOV @ RXCS, R1
1196
1197                                ; (R0) = 140 ; (R1) = ACTUAL RXCS ; (R2) = N/A
1198
1199  003062  104000                        ERROR                           ; THE RXCS NOT = 140
1200

        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

                ;THE FOLLOWING IS A PRESENTATION OF POSSIBLE REASONS AS TO WHY
                ;THIS ERROR REPORT WAS GENERATED.  THE INFORMATION SHOWN IS
                ;BASED ON FAULT INSERTION RESULTS,  AND SHOULD PROVIDE LOGICAL
                ;AREAS TO CHECK FOR THE RELEVANT FAULT/S.
                ;
                ;IF THIS ERROR REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS
                ;ANALYZE THE FOLLOWING AREA/S:

        ;M7846 (UNIBUS INTERFACE)

        ;       SIGNAL NAME             REASON                  POSSIBLE CHIPS

        ;BUS D07                STUCK HIGH                      E4
        ;BUS D06                STUCK HIGH                      E1

        ;/:\*/:\*/: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/: */:\*/: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\* :\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

1225  003064  104413                2$:     SUBSCOPE
1226
1227
1228                                ; TEST 2 - CONT'D
1229
1230                                ; THE PURPOSE OF THIS TEST IS TO VERIFY THAT BIT 6 OF THE RXCS (INTERRUPT ENABLE.
```

```
1231                                    ; CAN BE CLEARED AFTER IT WAS KNOWN TO BE SET
1232
1233  003066  013702  001204                    MOV KRXVEC, R2
1234  003072  010246                            MOV     R2,-(SP)            ;SAVE INTERRUPT VECTOR FOR
1235                                                                        ;ERROR REPORT
1236  003074  012722  003164                    MOV #4$, (R2)+              ; RX11 VECTOR ADDRESS
1237  003100  012722  000340                    MOV #PR7, (R2)+
1238  003104  012602                            MOV     (SP)+,R2            ;RESTORE INTERRUPT VECTOR FOR
1239                                                                        ;ERROR REPORT
1240  003106  042777  000100  176072            BIC #BIT6, @ RXCS           ; CLEAR THE RX11 INTERRUPT ENABLE BIT
1241
1242                                    ; THE RXCS SHOULD = 40 (DONE)
1243
1244  003114  012700  000040                    MOV #40, R0
1245  003120  020077  176062                    CMP R0, @ RXCS
1246  003124  001403                            BEQ 3$
1247  003126  017701  176054                    MOV @ RXCS, R1
1248
1249                                    ; (R0) = 40 ; (R1) = ACTUAL RXCS ; (R2) = N/A
1250
1251  003132  104000                            ERROR                       ; RXCS NOT = 40
1252  003134  104413                    3$:     SUBSCOPE
1253                                    ; NO RX11 INTERRUPTS SHOULD OCCUR [YET]
1254
1255  003136  005046                            CLR -(SP)                   ; PDP PRIORITY (ON)
1256  003140  012746  003146                    MOV #10$,-(SP)
1257  003144  000002                            RTI
1258  003146  000240                    10$:    NOP
1259  003150  000240                            NOP
1260  003152  012746  000340                    MOV #PR7,-(SP)              ; PDP PRIORLITY (OFF) 7
1261  003156  012746  003166                    MOV #11$,-(SP)
1262  003162  000002                            RTI
1263
1264                                    ; RETURN TO HERE WITH THE PDP PRIORITY = 7 IF AN UNEXPECTED RX11 INTERRUPT
1265                                    ; WHILE CLEARING THE RX11 INTERRUPT ENABLE BIT 6
1266
1267                                    ; (R0) = N/A ; (R1) = N/A ; (R2) = N/A
1268
1269  003164  104000                    4$:     ERROR                       ; UNEXPECTED RX11 INTERRUPT
1270  003166  104413                    11$:    SUBSCOPE
1271
1272                                    ; AN RX11 INTERRUPT SHOULD OCCUR [NOW]
1273
1274  003170  013702  001204                    MOV KRXVEC, R2
1275  003174  010246                            MOV     R2,-(SP)            ;SAVE INTERRUPT VECTOR FOR
1276                                                                        ;ERROR REPORT
1277  003176  012722  003246                    MOV #5$, (R2)+              ; RX11 VECTOR ADDRESS
1278  003202  012722  000340                    MOV #PR7, (R2)+
1279  003206  012602                            MOV     (SP)+,R2            ;RESTORE INTERRUPT VECTOR FOR
1280                                                                        ;ERROR REPORT
1281  003210  005046                            CLR -(SP)                   ; PDP PRIORITY (ON)
1282  003212  012746  003220                    MOV #12$,-(SP)
1283  003216  000002                            RTI
1284  003220  052777  000100  175760    12$:    BIS #BIT6, @ RXCS           ; SET RX11 INTERRUPT ENABLE BIT
1285  003226  000240                            NOP
1286  003230  000240                            NOP
```

```
1287  003232  012746  000340              MOV #PR7,-(SP)
1288  003236  012746  003244              MOV #13$,-(SP)
1289  003242  000002                      RTI
1290
1291                                  ; (R0) = N/A ; (R1) = N/A ; (R2) = N/A
1292
1293  003244  104000                  13$:    ERROR                       ; NO RX11 INTERRUPT OCCURRED
1294
1295                                  ; RETURN TO HERE WITH THE PDP PRIORITY = 7 IF AN RX01 INTERRUPT
1296
1297  003246  000004                  5$:     SCOPE
1298  003250  042777  000100  175730        BIC #BIT6, @ RXCS           ; CLEAR THE RX11 INTERRUPT ENABLE
1299
1300  003256  000137  004246                  JMP CEXIT                   ;END OF TEST 2
```

```
                                        .SBTTL TEST 3 - INTERRUPT TEST PART III / PRIORITY LEVEL VERIFICATION PART I

                                ; THE PURPOSE OF THIS TEST IS TO VERIFY THE PRIORITY OF THE RX11 INTERRUPT REQUEST LINE
                                ; THE PROGRAM SETS THE PDP PRIORITY TO 1 LESS THAN THE DEVICE LEVEL
                                ;(DEVICE LEVEL SPECIFIED BY CONTENTS OF LOCATION 'BRLEV:' -- NORMALLY 5)
                                ; AN RX01 INTERRUPT SHOULD OCCUR
                                ; IF NO INTERRUPT OCCURS THEN THE PRIORITY LEVEL OF THE RX11 IS [NOT] = NORMAL
                                ;DEVICE LEVEL OF 5 OR THE DEVICE LEVEL AS SPECIFIED BY THE CONTENTS OF
                                ;LOCATION 'BRLEV:' WHICH MAY HAVE BEEN CHANGED BY THE USER BEFORE PROGRAM
                                ; EXECUTION, BUT MAYBE SOME VALUE LESS THAN THE CONTENTS OF LOCATION 'BRLEV:'.
                                ; NOTE: IF THERE IS NO HARDWARE "PSW" THIS TEST WILL BE SKIPPED.

003262  005001          T3:     CLR     R1                      ;INDICATOR TO CPU PRIORITY
                                                                ;ROUTINE TO DROP CPU PRIORITY
                                                                ;TO 1 LESS THAN DEVICE LEVEL
003264  013702  001204          MOV KRXVEC, R2
003270  010246                  MOV     R2,-(SP)                ;SAVE INTERRUPT VECTOR FOR
                                                                ;ERROR REPORT
003272  012722  003404          MOV #1$, (R2)+                  ; RX01 VECTOR ADDRESS
003276  012722  000340          MOV #PR7, (R2)+
003302  012602                  MOV     (SP)+,R2                ;RESTORE INTERRUPT VECTOR FOR
                                                                ;ERROR REPORT
003304  013746  000004          MOV     4,-(SP)                 ;SAVE 'BUSERR' TIMEOUT 'PC'
003310  012737  003326  000004  MOV     #2$,4                   ;SET TIMEOUT VECTOR
003316  012737  000200  177776  MOV     #PR4,PSW                ;SET LEVEL TO 4 IF 'PSW' EXISTS
003324  000404                  BR      3$                      ;GO TO RESET VECTOR 4 & DO TEST
003326  022626          2$:     CMP     (SP)+,(SP)+             ;CORRECT STACK FROM BUS TIMEOUT
003330  012637  000004          MOV     (SP)+,4                 ;RESTORE TIMEOUT VECTOR TO 'BUSERR'
003334  000427                  BR      4$                      ;NO HARDWARE PSW - SKIP THIS TEST
003336  012637  000004  3$:     MOV     (SP)+,4                 ;RESET TIMEOUT VECTOR TO 'BUSERR'
003342  004737  006200          JSR     PC,CPUPRI               ;CALCULATE PRIORITY LEVEL OF CPU
                                                                ;BASED ON CURRENT DEVICE PRIORITY
                                                                ;LEVEL RESIDING IN LOC. 'BRLEV'
003346  010046                  MOV     R0,-(SP)        ;;PUT NEW PS ON STACK
003350  012746  003356          MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
003354  000002                  RTI                     ;;POP NEW PC AND PS
003356                  64$:
003356  052777  000100  175622  BIS #BIT6, @ RXCS                ; SET THE RX01 INTERRUPT ENABLE
003364  000240                  NOP
003366  000240                  NOP
003370  013746  000340          MOV     PR7,-(SP)       ;;PUT NEW PS ON STACK
003374  012746  003402          MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
003400  000002                  RTI                     ;;POP NEW PC AND PS
003402                  65$:

                                ; (R0) = N/A ; (R1) = N/A ; (R2) = N/A

003402  104000                  ERROR                           ;PRIORITY LEVEL IS NOT = CONTENTS
                                                                ;OF 'BRLEV:' (NORMALLY 5) BUT
                                                                ;MAYBE SOME VALUE LESS THAN THE
                                                                ;THE CURRENT CONTENTS OF 'BRLEV:'

                                ; RETURN TO HERE WITH THE PDP PRIORITY = 7 IF AN RX01 INTERRUPT

003404  000004          1$:     SCOPE
003406  042777  000100  175572  BIC #BIT6, @ RXCS                ; CLEAR THE RX11 INTERRUPT ENABLE
```

```
1357  003414  000137  004246          4$:     JMP CEXIT                       ;END OF TEST 3
1358
1359                                          .SBTTL TEST 4 - INTERRUPT TEST PART IV / PRIORITY VERIFICATION PART II
1360
1361                                  ; THE PURPOSE OF THIS TEST IS TO VERIFY THE PRIORITY OF THE RX11 INTERRUPT REQUEST LINE
1362                                  ; THE PROGRAM SETS THE PDP PRIORITY = THE DEVICE LEVEL, (NORMALLY 5 OR THE CONTENTS OF _
1363                                  ;NO RX01 INTERRUPTS SHOULD OCCUR
1364                                  ; IF AN INTERRUPT DOES OCCUR THEN THE PRIORITY LEVEL OF THE RX11 IS [NOT]
1365                                  ; = THE NORMAL DEVICE LEVEL OF 5, OR WHATEVER IS THE VALUE IN LOCATION 'BRLEV:'
1366                                  ; BUT MAYBE SOME VALUE GREATER THAN THE CONTENTS OF LOC. 'BRLEV:'
1367                                  ; NOTE: IF THERE IS NO HARDWARE "PSW" THIS TEST WILL BE SKIPPED.
1368
1369  003420  005001          T4:     CLR     R1                              ;INDICATOR TO CPU PRIORITY ROUTINE
1370                                                                          ;TO DROP CPU PRIORITY 1 LEVEL
1371                                                                          ;LESS THAN THE DEVICE LEVEL
1372  003422  013702  001204                  MOV KRXVEC, R2
1373  003426  010246                          MOV     R2,-(SP)                ;SAVE INTERRUPT VECTOR FOR
1374                                                                          ;ERROR REPORT
1375  003430  012722  003546                  MOV #1$, (R2)+                  ; RX01 VECTOR ADDRESS
1376  003434  012722  000340                  MOV #PR7, (R2)+
1377  003440  012602                          MOV     (SP)+,R2                ;RESTORE INTERRUPT VECTOR FOR
1378                                                                          ;ERROR REPORT
1379  003442  052701  000200                  BIS     #BIT7,R1                ;SET INDICATOR TO CPU PRIORITY
1380                                                                          ;ROUTINE TO SET CPU PRIORITY LEVEL
1381                                                                          ;TO THE SAME LEVEL AS THE DEVICE
1382  003446  013746  000004                  MOV     4,-(SP)                 ;SAVE 'BUSERR' TIMEOUT 'PC'
1383  003452  012737  003470  000004          MOV     #3$,4                   ;SET TIMEOUT VECTOR
1384  003460  012737  000240  177776          MOV     #PR5,PSW                ;SET LEVEL TO 5 IF 'PSW' EXISTS
1385  003466  000404                          BR      4$                      ;GO ON TO RESET VECTOR & DO TEST
1386  003470  022626          3$:     CMP     (SP)+,(SP)+                     ;CORRECT STACK FROM BUS TIMEOUT
1387  003472  012637  000004                  MOV     (SP)+,4                 ;RESTORE TIMEOUT VECTOR TO 'BUSERR'
1388  003476  000430                          BR      5$                      ;NO HARDWARE PSW - SKIP THIS TEST
1389  003500  012637  000004          4$:     MOV     (SP)+,4                 ;RESET TIMEOUT VECTOR TO BUSERR
1390  003504  004737  006200                  JSR     PC,CPUPRI               ;CALCULATE CPU PRIORITY LEVEL TO
1391                                                                          ;BE THE SAME AS THE DEVICE LEVEL
1392                                                                          ;I.E. - SAME AS CONTENTS OF LOC.
1393                                                                          ;'BRLEV'
1394  003510  010046                          MOV     R0,-(SP)        ;;PUT NEW PS ON STACK
1395  003512  012746  003520                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
1396  003516  000002                          RTI                     ;;POP NEW PC AND PS
1397  003520                  64$:
1398  003520  052777  000100  175460          BIS     #BIT6,@RXCS             ;SET RX01 INTERRUPT ENABLE
1399  003526  000240                          NOP
1400  003530  000240                          NOP
1401  003532  013746  000340                  MOV     PR7,-(SP)       ;;PUT NEW PS ON STACK
1402  003536  012746  003544                  MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
1403  003542  000002                          RTI                     ;;POP NEW PC AND PS
1404  003544                  65$:
1405  003544  000401                          BR 2$
1406
1407                                  ; RETURN TO HERE WITH THE PDP PRIORITY = 7 IF AN RX01 INTERRUPT
1408
1409                                  ; (R0) = N/A ; (R1) = N/A ; (R2) = N/A
1410
1411  003546  104000          1$:     ERROR                                   ;PRIORITY LEVEL NOT = TO CONTENTS
1412                                                                          ;OF LOCATION 'BRLEV:' (NORMALLY 5)
```

```
1413                                                    ;BUT MAYBE SOME VALUE GREATER THAN
1414                                                    ;THE CONTENTS SPECIFIED BY LOC.
1415                                                    ;'BRLEV:'
1416  003550  000004                    2$:     SCOPE
1417  003552  042777  000100  175426            BIC #BIT6, @ RXCS       ; CLEAR THE RX01 INTERRUPT ENABLE
1418  003560  000137  004246            5$:     JMP CEXIT               ;END OF TEST 4
```

```
1419                                    .SBTTL TEST 5 - INIT [PROGRAMMED] / RST
1420
1421                                ; THE PURPOSE OF THIS TEST IS TO VERIFY THAT SETTING THE RXCS BIT 14
1422                                ; CAUSES AN RX01 PROGRAMMED SUBSYSTEM INITIALIZE
1423
1424                                ; THE RXCS SHOULD = 40 (DONE)
1425
1426                                ; THE RXDB SHOULD = 4, OR 104, OR 204, OR 304
1427
1428  003564  052777  040000  175414  T5:   BIS #BIT14, @ RXCS          ; RX01 PROGRAMMED INITIALIZE
1429  003572  004737  006574          1$:   JSR PC, SDN                 ; WAIT FOR THE DONE BIT
1430  003576  000775                        BR 1$
1431  003600  012700  000040                MOV #40, R0                 ; PROGRAM EXPECTS RXCS = 40 (DONE)
1432  003604  017701  175376                MOV @ RXCS, R1              ; ACTUAL RXCS
1433  003610  020100                        CMP R1, R0
1434  003612  001401                        BEQ 2$
1435
1436                                ; (R0) = 40 ; (R1) = ACTUAL RXCS ; (R2) = N/A
1437
1438  003614  104000                        ERROR                       ; RXCS NOT = 40
1439
```

```
;/*:/*\:/*:/*\:/*:/*\:/*:/*\:/*:/*\:/*:/*\:/*:/*\:/*:/*\:/*:/*\:/*:/*\:/*:
;/*\:/*:/*\:/*:/*\:/*:/*\:/*:/*\:/*:/*\:/*:/*\:/*:/*\:/*:/*\:/*:/*\:/*:/*\:
;/* :/*\:/*\:/*\:/*:/*\:/*\:/*\:/*\:/*\:.*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

        ;IF THE FAULT CANNOT BE FOUND ON THE UNIBUS INTERFACE MODULE
        ;AND/OR THE FAULT IS NOT INHERENT TO THE UNIBUS INTERFACE MODULE
        ;M7846 THERE IS A POSSIBILITY OF ITS EXISTENCE ON THE READ/WRITE
        ;MODULE M7727.
;
;       NOTE:   ONLY APPROX. 30% OF THIS MODULE LENT ITSELF CONDUCIVE TO
;               THE FAULT INSERTION PROCESS; ERGO, THE RESOLUTION FOR FAULT
;               ANALYSIS OBTAINABLE BY THIS MODULE IS NOT VERY HIGH.
;               HOWEVER, ANALYSIS OF THE FOLLOWING AREA/S, IF THIS ERROR
;               REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS, SHOULD AT
;               LEAST PLACE YOU WITHIN THE RELEVANT AREA ON THE MODULE.
;
;M7727 (READ/WRITE CONTROL)
;
;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
;
; ----------            'J' INPUT LOCKED LOW            E16
;
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
```

```
1467  003616  104413                   2$:   SUBSCOPE
1468
1469
1470
1471                                    ;       7       6       -       -       3       2       1       0
1472                                    ;
1473                                    ;       SEL                             WRITE   INIT    PAR
1474                                    ;       DRIVE   DD                      PROTECT [DONE]          CRC
```

```
1475                                    ;       RDY                                             (N/A)
1476                                    ;
1477                                    ;////////////////////////////////////////////////////////////////////////
1478
1479                                    ; THE RXDB SHOULD = 4, 104, IF TESTING UNIT 1
1480                                    ; OR 204, OR 304 IF TESTING UNIT 0 (SEL. DRIVE RDY. BIT SET.
1481
1482  003620  012700  000204                       MOV #204,R0
1483  003624  017702  175360                       MOV @RXDB,R2
1484  003630  010201                               MOV R2,R1
1485  003632  042701  000100                       BIC #BIT6,R1                    ;CLEAR DELETED DATA BIT
1486  003636  105737  012724                       TSTB UNITSEL                    ;WAS UNIT 0 SELECTED
1487  003642  100404                               BMI 3$
1488  003644  042701  000200                       BIC #BIT7,R1                    ;UNIT 0 WAS NOT SELECTED
1489  003650  042700  000200                       BIC #BIT7,R0                    ;CLEAR UNIT 0 RDY BITS
1490  003654  020100                    3$:        CMP R1, R0
1491  003656  001401                               BEQ 4$
1492
1493                                    ; (R0) = 4, OR 204
1494                                    ; (R1) = ACTUAL RXDB MINUS DELETED DATA BIT#6
1495                                    ; (R2) = ACTUAL RXDB
1496
1497  003660  104000                               ERROR                           ; RXDB NOT = 4, OR 104, OR 204, OR 304
1498
```

```
        ;/*:/*\:/*\:/*\:/* :/*\:/*:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*:/*:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*:/*:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

                ;THE FOLLOWING IS A PRESENTATION OF POSSIBLE REASONS AS TO WHY
                ;THIS ERROR REPORT WAS GENERATED.  THE INFORMATION SHOWN IS
                ;BASED ON FAULT INSERTION RESULTS,  AND SHOULD PROVIDE LOGICAL
                ;AREAS TO CHECK FOR THE RELEVANT FAULT/S.
                ;
                ;IF THIS ERROR REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS
                ;ANALYZE THE FOLLOWING AREA/S:
        ;
        ;M7846 (UNIBUS INTERFACE)
        ;
        ;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
        ;
        ;RX INIT                STUCK HIGH                      E11,E36
        ;IN                     STUCK HIGH                      E11
        ;BUS DO5                STUCK LOW                       E1
        ;RX DATA                STUCK LOW                       E14
        ; ----------            SACK FLOP CLOCK LOCKED LOW      E9
        ;B SER DATA             STUCK LOW                       E9
        ;DONE                   STUCK HIGH                      E22
        ; ----------            LOAD PULSE STUCK LOW            E3
        ;
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/: */: *
        ;/:\*/: */:\*/:\*/:\*/: */: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/: */: *
        ;/:\*/:\*/:\*/:\*/:\*/: */: */: */:\*/:\*/:\*/:\*/: * :\*/:\*/:\*/:\*/: * : *

1529  003662  104413                    4$:     SUBSCOPE
```

```
1530                                    ; TEST 5 CONT'D - RXCS TEST PART II / RST
1531
1532                                    ; THE PURPOSE OF THIS TEST IS TO VERIFY THE RST COMMAND #12+GO
1533                                    ; AND THAT THE DONE BIT IS CLEARED AFTER THE FUNCTION IS ISSUED.
1534
1535  003664  012777  000013  175314          MOV #13, @ RXCS                 ; RST COMMAND
1536  003672  032777  000040  175306          BIT #BIT5, @ RXCS               ; TEST DONE BIT (SHOULD = 0)
1537  003700  001404                          BEQ 5$
1538  003702  017701  175300                  MOV @ RXCS, R1
1539  003706  005000                          CLR R0
1540
1541                                    ; (R0) = 0 ; (R1) = ACTUAL RXCS ; (R2) = N/A
1542
1543  003710  104000                          ERROR                           ; DONE BIT NOT = 0
1544  003712  104413                  5$:     SUBSCOPE
1545  003714  004737  006574          9$:     JSR PC, SDN                     ; WAIT FOR DONE FLAG
1546  003720  000775                          BR 9$
1547  003722  012700  000040                  MOV #40, R0                     ; PROGRAM EXPECTS RXCS = 40 (DONE)
1548  003726  017701  175254                  MOV @ RXCS, R1                  ; ACTUAL RXCS
1549  003732  020100                          CMP R1, R0
1550  003734  001401                          BEQ 10$
1551
1552                                    ; (R0) = 40 ; (R1) = ACTUAL RXCS ; (R2) = N/A
1553
1554  003736  104000                          ERROR                           ; RXCS NOT = 40
1555

        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*  :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

                ;THE FOLLOWING IS A PRESENTATION OF POSSIBLE REASONS AS TO WHY
                ;THIS ERROR REPORT WAS GENERATED.  THE INFORMATION SHOWN IS
                ;BASED ON FAULT INSERTION RESULTS,  AND SHOULD PROVIDE LOGICAL
                ;AREAS TO CHECK FOR THE RELEVANT FAULT/S.
                ;
                ;IF THIS ERROR REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS
                ;ANALYZE THE FOLLOWING AREA/S:
        ;
        ;M7846 (UNIBUS INTERFACE)
        ;
        ;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
        ;
        ; ----------            PARITY FLOP CLOCK LOCKED HIGH   E2
        ; ----------            PARITY FLOP 'Q' OUTPUT          E2
        ;                         LOCKED HIGH
        ;RX DATA                STUCK LOW                       E6
        ;
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:  */:\*/:  */:  */:\*/:  */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:  */:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:  */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

1582  003740  104413          10$:    SUBSCOPE
1583
1584                                    ; THE RXDB SHOULD = 200 (IF DRIVE 0 IS READY), OR 0 IF UNIT 0 IS NOT SELECTED
1585                                    ; MAYBE 300 (IF DRIVE 0 IS READY AND SECTOR 1 WAS WRITTEN WITH DELETED DATA.
```

```
1586
1587  003742  017702  175242          MOV @ RXDB, R2                  ; ACTUAL RXDB
1588  003746  010201                  MOV R2, R1
1589  003750  042701  000100          BIC #BIT6, R1                   ; CLEAR N/A DELETED DATA BIT
1590  003754  012700  000200          MOV #200,R0                     ;EXPECT UNIT 0 RDY SET
1591  003760  105737  012724          TSTB UNITSEL
1592  003764  100403                  BMI 11$
1593  003766  042701  000200          BIC #BIT7,R1                    ; UNIT 0 NOT SELECTED
1594  003772  005000                  CLR R0                          ; DISREGUARD RDY BITS
1595  003774  020100          11$:    CMP R1, R0
1596  003776  001401                  BEQ 12$
1597
1598                          ; (R0) = 0 OR 200
1599                          ; (R1) = ACTUAL RXDB MINUS DELETED DATA BIT#6
1600                          ; (R2) = ACTUAL RXDB
1601
1602  004000  104000                  ERROR                           ; RXDB NOT = 200, OR NOT = 0
1603  004002  000004          12$:    SCOPE
1604  004004  000137  004246          JMP CEXIT                       ;END OF TEST 5
```

```
1605                                    .SBTTL TEST 6 - FILL BUFFER TRANSFER LENGTH VERIFICATION
1606
1607                                    ; THE PURPOSE OF THIS TEST IS TO VERIFY THE TRANSFER LENGTH OF THE FUNCTION
1608                                    ; "FILL BUFFER" OF THE RX01 MICROCONTROLLER
1609
1610                                    ; 128 BYTE TRANSFERS SHOULD OCCUR
1611
1612  004010  012737  000006  012324  T6:     MOV #6, PAT                     ; COUNT PATTERN
1613  004016  004737  004026                  JSR PC, T6FILL
1614  004022  000137  004246                  JMP CEXIT                       ;END OF TEST 6
1615
1616                                    ; EXECUTE THE FOLLOWING " MOV #1, @ RXCS " INSTEAD OF " INC @ RXCS " FOR LOOPS
1617
1618  004026  005002                  T6FILL: CLR R2
1619  004030  012777  000001  175150          MOV #1, @ RXCS                  ;FILL BUFFER FUNCTION [0 + GO BIT]
1620  004036  004737  012260                  JSR PC, GETPATTERN
1621  004042  012704  017420                  MOV #BUFADR, R4
1622  004046  004737  004064          1$:     JSR PC, FBEB                    ; SUBROUTINE TO FILL/EMPTY BUFFER
1623  004052  000207                          RTS PC                          ; EXIT SUBROUTINE T6FILL
1624
1625  004054  112477  175130                  MOVB (R4)+, @ RXDB              ; FILL THE SECTOR BUFFER
1626  004060  005202                          INC R2                          ;INC R2 FOR BYTE COUNT
1627  004062  000771                          BR 1$
1628
1629                                    ; SUBROUTINE TO FILL AND EMPTY THE SECTOR BUFFER
1630
1631  004064  004737  006560          FBEB:   JSR PC, STR                     ; TEST FOR TRANSFER REQUEST
1632  004070  000403                          BR 1$
1633  004072  062716  000002                  ADD #2, @ SP                    ; ADJUST FOR EXIT FROM THIS SUBROUTINE
1634  004076  000207                          RTS PC                          ; EXIT TO SERVICE TRANSFER REQUEST
1635
1636  004100  004737  006574          1$:     JSR PC, SDN
1637  004104  000767                          BR FBEB                         ; NOT TRANSFER REQUEST OR DONE
1638  004106  005777  175074                  TST @ RXCS                      ;TEST FOR ERROR FLAG
1639  004112  100003                          BPL 3$
1640  004114  017701  175070                  MOV @RXDB,R1
1641
1642                                    ; (R0) = N/A ; (R1) = RXDB (STATUS A) ; (R2) = ACTUAL # OF TRANSFERS
1643
1644  004120  104000                          ERROR                           ; UNEXPECTED RX11 ERROR FLAG
1645

        ; * :/*\:/* :/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        :/*\:/* :/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        :/* :/*.:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

                ;THE FOLLOWING IS A PRESENTATION OF POSSIBLE REASONS AS TO WHY
                ;THIS ERROR REPORT WAS GENERATED.  THE INFORMATION SHOWN IS
                ;BASED ON FAULT INSERTION RESULTS,  AND SHOULD PROVIDE LOGICAL
                ;AREAS TO CHECK FOR THE RELEVANT FAULT/S.
                ;
                ;IF THIS ERROR REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS
                ;ANALYZE THE FOLLOWING AREA/S:
        ;
        ;M7846  UNIBUS INTERFACE.
        ;
```

```
        ;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
        ;
        ; ----------            PARITY FLOP CAN'T BE CLEARED    E2
        ;
        ;/: */: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/: */: */:\*/:\*/:\*/:\*/: */: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

1669  004122  104413                  3$:     SUBSCOPE
1670
1671                                  ; 128 BYTES SHOULD HAVE BEEN TRANSFERRED TO OR FROM THE SECTOR BUFFER
1672
1673  004124  022702  000200                  CMP #200,R2
1674  004130  001401                          BEQ 2$
1675
1676                                  ; (R0) = N/A ; (R1) = N/A ; (R2) = ACTUAL # OF TRANSFERS
1677
1678  004132  104000                          ERROR                           ; INCORRECT TRANSFER LENGTH
1679

        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

                ;THE FOLLOWING IS A PRESENTATION OF POSSIBLE REASONS AS TO WHY
                ;THIS ERROR REPORT WAS GENERATED.  THE INFORMATION SHOWN IS
                ;BASED ON FAULT INSERTION RESULTS,  AND SHOULD PROVIDE LOGICAL
                ;AREAS TO CHECK FOR THE RELEVANT FAULT/S.
                ;
                ;IF THIS ERROR REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS
                ;ANALYZE THE FOLLOWING AREA/S:

        ;M7846 (UNIBUS INTERFACE)
        ;
        ;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
        ;
        ;RX DATA                STUCK LOW                       E19,E6
        ;
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

1703  004134  000004                  2$:     SCOPE
1704
1705  004136  000207                          RTS PC                          ; EXIT SUBROUTINE FBEB
```

```
1706                                    .SBTTL TEST 10 - EMPTY BUFFER XFER LENGTH AND CONTENT VERIFICATION PART II
1707
1708                                    ; THE PURPOSE OF THIS TEST IS TO VERIFY THAT THE PREVIOUS EMPTY BUFFER TEST
1709                                    ; DID NOT EMPTY AND DESTROY THE CONTENTS OF THE SECTOR BUFFER
1710                                    ;NOTE: TEST 6 MUST BE RUN BEFORE THIS TEST.
1711
1712  004140  000240                    T10:    NOP                             ; NOP FOR T10 " FAST " DEFINITION
1713
1714                                    .SBTTL TEST 7 - EMPTY BUFFER XFER LENGTH AND CONTENT VERIFICATION PART I
1715
1716                                    ; THE PURPOSE OF THIS TEST IS TO VERIFY THE TRANSFER LENGTH OF THE FOUNCTION
1717                                    ; "EMPTY BUFFER" AND ALSO TO VERIFY THE CONTENTS OF THE SECTOR BUFFER
1718                                    ; NOTE TEST 6 MUST BE RUN BEFORE THIS TEST
1719
1720  004142  004737  004152            T7:     JSR PC, T7EMPTY
1721  004146  000137  004246                    JMP CEXIT                       ;END OF TEST 7 OR 10
1722  004152  005002                    T7EMPTY: CLR R2
1723  004154  012777  000003  175024            MOV #3, @ RXCS                  ; ISSUE THE COMAND TO EMPTY BUFFER
1724  004162  012704  017420                    MOV #BUFADR, R4
1725  004166  004737  004064            2$:     JSR PC, FBEB                    ; SUBROUTINE TO EMPTY BUFFER
1726  004172  000207                            RTS PC                          ; EXIT SUBROUTINE T7EMPTY
1727
1728                                    ; EMPTY BUFFER AND VERIFY THE CONTENTS
1729
1730  004174  112400                            MOVB (R4)+, R0                  ; EXPECTED CONTENTS OF SECTUR BUFFER
1731  004176  117701  175006                    MOVB @ RXDB, R1                 ; ACTUAL CONTENTS OF SECTOR BUFFER
1732  004202  005202                            INC R2                          ; ACTUAL # TRANSFERS TO THIS ERROR
1733  004204  020001                            CMP R0, R1
1734  004206  001401                            BEQ 1$
1735
1736                                    ; IF AN ERROR OCCURS:
1737
1738                                    ; (R0) - EXPECTED CONTENTS OF SECTOR BUFFER
1739                                    ; (R1) - ACTUAL CONTENTS FROM SECTOR BUFFER
1740                                    ; (R2) - TOTAL # OF ACTUAL TRANSFERS
1741
1742  004210  104000                            ERROR                           ; DATA " TO " SB NOT = DATA " FROM "
1743

        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

                ;THE FOLLOWING IS A PRESENTATION OF POSSIBLE REASONS AS TO WHY
                ;THIS ERROR REPORT WAS GENERATED.  THE INFORMATION SHOWN IS
                ;BASED ON FAULT INSERTION RESULTS,  AND SHOULD PROVIDE LOGICAL
                ;AREAS TO CHECK FOR THE RELEVANT FAULT/S.
                ;
                ;IF THIS ERROR REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS
                ;ANALYZE THE FOLLOWING AREA/S:
        ;
        ;M7846 (UNIBUS INTERFACE)
        ;
        ;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
        ;
        ;RX RUN                  STUCK LOW                       E24
```

```
        ;LOAD            STUCK LOW                          E18
        ; ----------            BINARY COUNTER 'RESET'      E34
        ;                          LOCKED HIGH
        ;RX DATA                 STUCK HIGH                 E3
        ;BUS D04/D05             STUCK HIGH/LOW             E1,E4
        ;BUS D02                 STUCK HIGH                 E7
        ;

                ;IF THE FAULT CANNOT BE FOUND ON THE UNIBUS INTERFACE MODULE
                ;AND/OR THE FAULT IS NOT INHERENT TO THE UNIBUS INTERFACE MODULE
                ;M7846 THERE IS A POSSIBILITY OF ITS EXISTENCE ON THE READ/WRITE
                ;MODULE M7727.
        ;
        ;       NOTE:   ONLY APPROX. 30% OF THIS MODULE LENT ITSELF CONDUCIVE TO
        ;               THE FAULT INSERTION PROCESS; ERGO, THE RESOLUTION FOR FAULT
        ;               ANALYSIS OBTAINABLE BY THIS MODULE IS NOT VERY HIGH.
        ;               HOWEVER, ANALYSIS OF THE FOLLOWING AREA/S, IF THIS ERROR
        ;               REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS, SHOULD AT
        ;               LEAST PLACE YOU WITHIN THE RELEVANT AREA ON THE MODULE.
        ;
        ;M7727 (READ/WRITE CONTROL)
        ;
        ;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
        ;
        ; ----------            REPLACE E5                        ----------
        ;
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/: *
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

1792  004212  000004                  1$:     SCOPE
1793  004214  000764                          BR 2$
1794
1795                                          .SBTTL TEST 12 - FILL/EMPTY BUFFER ALL 1'S
1796
1797                                  ; PROGRAMMING NOTE: THE FOLLOWING " MOV " INSTEAD OF " INC " FOR LOOPS
1798
1799  004216  012737  000001  012324  T12:     MOV #1, PAT
1800  004224  000402                          BR .+6
1801
1802                                          .SBTTL TEST 11 - FILL/EMPTY BUFFER ALL 0'S
1803
1804  004226  005037  012324          T11:    CLR PAT
1805  004232  004737  004026                  JSR PC, T6FILL
1806  004236  004737  004152                  JSR PC, T7EMPTY
1807  004242  000137  004246                  JMP CEXIT                       ;END OF TEST 11 OR 12
```

```
1808  004246  012737  000103  002534  CEXIT:  MOV #'C,MX                      ;EOP INDICATOR FOR END OF CONTROL TESTS
1809  004254  000137  002162                  JMP MORETESTS
1810
1811  004260  012737  000104  002534  DEXIT:  MOV #'D,MX                      ;EOP INDICATOR FOR END OF DRIVE TESTS
1812  004266  000137  002162                  JMP MORETESTS
1813
1814                                          .SBTTL TEST 13 DRIVE READY VERIFICATION
1815
1816                                  ;THIS TEST VERIFIES THAT DRIVE READY WILL SET FOR ALL SELECTED DRIVES.
1817
1818                                  ;THIS SECTION OF TEST FOR UNIT 0 DRY NO A READ STATUS A FUNCTION
1819
1820  004272  105737  012724          T13:    TSTB UNITSEL                    ;WAS UNIT 0 SELECTED
1821  004276  100020                          BPL 2$                          ;NO, GO TEST UNIT 1
1822  004300  012777  000013  174700          MOV #13,@RXCS                   ;READ STATUS A UNIT 0
1823  004306  004737  006574          1$:     JSR PC,$DN                      ;WAIT FOR DONE FLAG
1824  004312  000775                          BR 1$
1825  004314  012700  000200                  MOV #200,R0                     ;EXPECT DRIVE READY TO BE SET
1826  004320  017702  174664                  MOV @RXDB,R2                    ;ACTUAL RXDB
1827  004324  010201                          MOV R2,R1
1828  004326  042701  000100                  BIC #BIT6,R1                    ;CLEAR DD BIT IF SET
1829  004332  020001                          CMP R0,R1                       ;WAS " DRY " SET AND INITDONE CLEARED
1830  004334  001401                          BEQ 2$
1831
1832                                  ;R0 = 200 ; R1 = RXDB MINUS DD BIT ;R2 = ACTUAL RXDB
1833
1834  004336  104000                          ERROR
1835  004340  104413          2$:     SUBSCOPE
1836
1837                                  ;TEST FOR UNIT 1 DRIVE READY ON A READ STATUS A FUNCTION
1838
1839  004342  005737  012724                  TST UNITSEL                     ;WAS UNIT 1 SELECTED
1840  004346  100020                          BPL 3$                          ;NO, GO TO END OF TEST
1841  004350  012777  000033  174630          MOV #33,@RXCS                   ;READ STATUS A FOR DRIVE 1
1842  004356  004737  006574          4$:     JSR PC,$DN                      ;WAIT FOR DONE FLAG
1843  004362  000775                          BR 4$
1844  004364  012700  000200                  MOV #200,R0                     ;EXPECT " DRY " TO BE SET
1845  004370  017702  174614                  MOV @RXDB,R2
1846  004374  010201                          MOV R2,R1
1847  004376  042701  000100                  BIC #BIT6,R1                    ;CLEAR DD BIT IF ANY
1848  004402  020100                          CMP R1,R0                       ;IS DRY SET AND INITDONE CLEARED
1849  004404  001401                          BEQ 3$
1850
1851                                  ;R0 = 200 ; R1 = RXDB MINUS DD BIT ; R2 = ACTUAL RXDB
1852
1853  004406  104000                          ERROR
1854

        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

                ;IF THE FAULT CANNOT BE FOUND ON THE UNIBUS INTERFACE MODULE
                ;AND/OR THE FAULT IS NOT INHERENT TO THE UNIBUS INTERFACE MODULE
                ;M7846 THERE IS A POSSIBILITY OF ITS EXISTENCE ON THE READ/WRITE
                ;MODULE M7727.
```

```
;
;       NOTE:   ONLY APPROX. 30% OF THIS MODULE LENT ITSELF CONDUCIVE TO
;               THE FAULT INSERTION PROCESS; ERGO, THE RESOLUTION FOR FAULT
;               ANALYSIS OBTAINABLE BY THIS MODULE IS NOT VERY HIGH.
;               HOWEVER, ANALYSIS OF THE FOLLOWING AREA/S, IF THIS ERROR
;               REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS, SHOULD AT
;               LEAST PLACE YOU WITHIN THE RELEVANT AREA ON THE MODULE.
;
;M7727 (READ/WRITE CONTROL)
;
;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
;


;       NOTE:  MAKE SURE THE DRIVES ARE CONNECTED CORECTLY,THE
;               DISKETTES INSERTED,AND THE DOORS OF THE SELECTED DRIVES
;               ARE CLOSED. IF THE THESE CONDITIONS ARE NOT SET THERE
;               WILL BE AN ERROR AT THIS POINT.

;SEL DK1                STUCK LOW                       E13,E14
;DK1 INDX               STUCK HIGH/LOW                  E15
;SEL INDX               STUCK HIGH/LOW                  E15
; ----------            'A2' INPUT STUCK LOW            E15

;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

1892  004410  000004                 3$:     SCOPE
1893  004412  000137  004260                 JMP DEXIT                       ;END OF TEST 13
```

```
                                            .SBTTL TEST 14 - ERROR FLAG AND B-CODE VERIFICATION PART I

                                    ;THE PURPOSE OF THIS TEST IS TO VERIFY THAT TRYING TO READ A NON-EXISTANT
                                    ;SECTOR WILL CAUSE AN ERROR, AND THE CORRECT ERROR CODE WILL BE PUT
                                    ;INTO THE RXDB WHEN THE B STATUS IS READ.

                                    ;THIS SECTION INSURES THAT ONLY 2 TR FLAGS WERE REQUIRED TO TAKE THE
                                    ;DISKETTE ADDRESS, AND THAT AN ERROR DOES EXIST.

004416  005002                      T14:    CLR R2
004420  005000                              CLR R0
004422  105737  012724                      TSTB UNITSEL                    ;IS UNIT 0 SELECTED
004426  100004                              BPL 10$
004430  012777  000007  174550              MOV #7,@RXCS                    ;SET READ SECTOR FUNCTION AND GO
004436  000403                              BR 11$
004440  012777  000027  174540      10$:    MOV #27,@RXCS                   ;SEND READ FUNCTION TO UNIT 1
004446  004737  005106              11$:    JSR PC,ILLADR                   ;SEND THE BAD SECTOR ADDRESS
004452  000406                              BR 1$                           ;PREMATURE ERROR CONDITION
004454  012700  100040                      MOV #100040,R0                  ;EXPECT ERROR AND DONE BITS
004460  017701  174522                      MOV @RXCS,R1                    ;SAVE THE RXCS
004464  020001                              CMP R0,R1
004466  001401                              BEQ 2$

                                    ;R0 = 100040 ; R1 = ACTUAL RXCS ; R2 = # OF TR FLAGS

004470  104000                      1$:     ERROR

;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

        ;THE FOLLOWING IS A PRESENTATION OF POSSIBLE REASONS AS TO WHY
        ;THIS ERROR REPORT WAS GENERATED.  THE INFORMATION SHOWN IS
        ;BASED ON FAULT INSERTION RESULTS,  AND SHOULD PROVIDE LOGICAL
        ;AREAS TO CHECK FOR THE RELEVANT FAULT/S.
        ;
        ;IF THIS ERROR REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS
        ;ANALYZE THE FOLLOWING AREA/S:
;
;M7846 (UNIBUS INTERFACE)
;
;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
;
;BUS D15                STUCK HIGH                      E14,E9,E40,E24
;RX RUN                 STUCK LOW                       E41
;
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

004472  104413                      2$:     SUBSCOPE

                                    ;T14 CONT. - THIS SECTION TESTS THAT NO PARITY OR CRC ERROR OCCURRED
                                    ;ON PREVIOUS FUNCTION.
```

```
1950  004474  005000                      CLR R0                          ;STATUS A SHOULD BE CLEAR
1951  004476  017702  174506              MOV @RXDB,R2
1952  004502  010201                      MOV R2,R1
1953  004504  042701  000100              BIC #BIT6,R1
1954  004510  020001                      CMP R0,R1
1955  004512  001401                      BEQ 3$
1956
1957                                  ;R0 = 0 ; R1 = RXDB MINUS DD BIT ; R2 = ACTUAL RXDB
1958
1959  004514  104000                      ERROR
1960  004516  104413              3$:     SUBSCOPE
1961
1962                                  ;T14 CONT. - THIS SECTION TESTS THAT NO ERROR CONDITIONS EXIST ON A
1963                                  ;READ STATUS B FUNCTION.
1964
1965  004520  012777  000017  174460      MOV #17,@RXCS                   ;SET READ STATUS B FUNCTION
1966  004526  004737  006574      4$:     JSR PC,$DN                      ;WAIT FOR DONE FLAG
1967  004532  000775                      BR 4$
1968  004534  005777  174446              TST @RXCS                       ;IS THE ERROR BIT SET
1969  004540  100007                      BPL 5$                          ;NO, GO ON TO NEXT SECTION
1970  004542  012700  000040              MOV #40,R0                      ;YES,SET UP FOR ERROR
1971  004546  017701  174434              MOV @RXCS,R1
1972  004552  017702  174432              MOV @RXDB,R2
1973
1974                                  ;R0 = 40 ; R1 = RXCS ; R2 = RXDB
1975
1976  004556  104000                      ERROR                           ;RXCS NOT = 40
1977  004560  104413              5$:     SUBSCOPE
1978
1979                                  ;T14 CONT. - THIS SECTION TESTS THE B-CODE FOR " CAN'T FIND SECTOR " #70
1980
1981  004562  012700  000070              MOV #70,R0
1982  004566  017701  174416              MOV @RXDB,R1
1983  004572  005002                      CLR R2
1984  004574  020001                      CMP R0,R1                       ;IS THE B-CODE = 70
1985  004576  001401                      BEQ 6$
1986
1987                                  ;R0 = 70 ; R1 = ACTUAL B-CODE ; R2 = N/A
1988
1989  004600  104000                      ERROR                           ;RXDB NOT = 70
1990

        ;/*\:/*\:/*\:/*\:/*\:/*\:/* :/*\:/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

                ;IF THE FAULT CANNOT BE FOUND ON THE UNIBUS INTERFACE MODULE
                ;AND/OR THE FAULT IS NOT INHERENT TO THE UNIBUS INTERFACE MODULE
                ;M7846 THERE IS A POSSIBILITY OF ITS EXISTENCE ON THE READ/WRITE
                ;MODULE M7727.
        ;
        ;       NOTE:   ONLY APPROX. 30% OF THIS MODULE LENT ITSELF CONDUCIVE TO
        ;               THE FAULT INSERTION PROCESS; ERGO, THE RESOLUTION FOR FAULT
        ;               ANALYSIS OBTAINABLE BY THIS MODULE IS NOT VERY HIGH.
        ;               HOWEVER, ANALYSIS OF THE FOLLOWING AREA/S, IF THIS ERROR
        ;               REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS, SHOULD AT
```

```
                LEAST PLACE YOU WITHIN THE RELEVANT AREA ON THE MODULE.

;M7727 (READ/WRITE CONTROL)

;       SIGNAL NAME             REASON                  POSSIBLE CHIPS

;INIT                   STUCK HIGH                      E16
;SEL TRK 0              STUCK HIGH                      E15
;DK1 TRK 0              STUCK HIGH                      E15
;SEL DK0                STUCK LOW                       E15,E14,E13
;WT GATE                STUCK HIGH                      E5
;SEL WT PROT            STUCK LOW                       E5
;RAW DATA               STUCK HIGH/LOW                  E14
;STEP                   STUCK LOW                       E14
; ----------            REPLACE E4                       ----------

;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

2026  004602  000004                  6$:     SCOPE
2027  004604  000137  004260                  JMP DEXIT                       ;END OF TEST 14
2028
```

```
2029                                    .SBTTL TEST 15 - ERROR FLAG AND B-CODE VERIFICATION PART II
2030
2031                                    ;THIS TEST VERIFIES THAT TRYING TO WRITE,USING DELETED DATA MODE, ON A
2032                                    ;NON-EXISTANT SECTOR WILL PRODUCE AN ERROR AND THE CORRECT B-CODE IS PRODUCED
2033                                    ;THIS SECTION SENDS OUT AN ILLEGAL SECTOR ADDRESS AND EXPECTS AN ERROR
2034                                    ; NOTE TEST 14 MUST BE RUN BEFORE THIS TEST
2035
2036  004610  005000                    T15:    CLR R0
2037  004612  005002                            CLR R2
2038  004614  105737  012724                    TSTB UNITSEL                    ;WAS UNIT 0 SELECTED
2039  004620  100004                            BPL 10$
2040  004622  012777  000015  174356            MOV #15,@RXCS                   ;SET WRITE DELETED DATA FUNCTION
2041  004630  000403                            BR 11$
2042  004632  012777  000035  174346  10$:     MOV #35,@RXCS                   ;SEND WTR DD FUNCTION TO UNIT 1
2043  004640  004737  005106          11$:     JSR PC,ILLADR                   ;SEND THE ILLEGAL SECTOR ADDRESS
2044  004644  000406                            BR 1$                           ;PREMATURE ERROR CONDITION
2045  004646  012700  100040                    MOV #100040,R0                  ;EXPECT ERROR AND DONE FLAGS
2046  004652  017701  174330                    MOV @RXCS,R1
2047  004656  020001                            CMP R0,R1
2048  004660  001401                            BEQ 2$
2049
2050                                    ;R0 = 100040 ;R1 = ACTUAL RXCS ; R2 = # OF TR FLAGS
2051
2052  004662  104000                    1$:     ERROR                           ;RXCS NOT = 100040
2053  004664  104413                    2$:     SUBSCOPE
2054
2055                                    ;T15 CONT. - THIS SECTION TESTS THAT THERE IS NO PARITY, CRC ERROR
2056                                    ;AND THAT THE DELETED DATA BIT IS SET.
2057
2058  004666  005002                            CLR R2
2059  004670  012700  000100                    MOV #100,R0                     ;EXPECT DELETED DATA BIT TO BE SET
2060  004674  017701  174310                    MOV @RXDB,R1
2061  004700  020001                            CMP R0,R1
2062  004702  001401                            BEQ 3$
2063
2064                                    ; R0 = 100 ; R1 = ACTUAL RXDB ; R2 = N/A
2065  004704  104000                            ERROR                           ;DELETED DATA NOT SET OR OTHER ERRORS
2066  004706  104413                    3$:     SUBSCOPE
2067
2068                                    ;T15 CONT. - THIS SECTION TESTS FOR THE B-CODE FOR ILLEGAL SECTOR.
2069
2070  004710  012777  000017  174270            MOV #17,@RXCS                   ;SET READ STATUS B FUNCTION
2071  004716  004737  006574          4$:     JSR PC,$DN                      ;WAIT FOR DONE FLAG
2072  004722  000775                            BR 4$
2073  004724  012700  000070                    MOV #70,R0
2074  004730  017701  174254                    MOV @RXDB,R1
2075  004734  020001                            CMP R0,R1
2076  004736  001401                            BEQ 5$
2077
2078                                    ; R0 = 70 ; R1 = ACTUAL B-CODE ; R2 = N/A
2079  004740  104000                            ERROR                           ;RXDB NOT = 70
2080  004742  000004                    5$:     SCOPE
2081  004744  000137  004260                    JMP @EXIT                       ;END OF TEST 15
```

```
                                        .SBTTL TEST 16 - ERROR FLAG AND B-CODE VERIFICATION PART III

                                ;THIS TEST VERIFIES THAT A WRITE FUNCTION TO A NON-EXISTANT SECTOR WILL
                                ;PRODUCE AN ERROR AND A B-CODE OF 70. THE DELETED DATA BIT SHOULD ALSO BE CLEARED
                                ;THIS SECTION TRANSFERS AN ILLEGAL SECTOR ADDRESS FOR A WRITE FUNCTION
                                ; NOTE TEST 14 MUST BE RUN BEFORE THIS TEST

004750  005000          T16:    CLR R0
004752  005002                  CLR R2
004754  105737  012724          TSTB UNITSEL                    ;WAS UNIT 0 SELECTED
004760  100004                  BPL 10$
004762  012777  000005  174216  MOV #5,@RXCS                    ;SET THE WRITE FUNCTION
004770  000403                  BR 11$
004772  012777  000025  174206  10$:    MOV #25,@RXCS           ;SEND WRITE FUNCTION TO UNIT 1
005000  004737  005106  11$:    JSR PC,ILLADR                   ;SEND THE ILLEGAL ADDRESS
005004  000406                  BR 1$                           ;PREMATURE ERROR CONDITION
005006  012700  100040          MOV #100040,R0                  ;EXPECT ERROR AND DONE BITS SET
005012  017701  174170          MOV @RXCS,R1
005016  020001                  CMP R0,R1
005020  001401                  BEQ 2$

                                ; R0 = 100040 , R1 = ACTUAL RXCS ; R2 = # OF TR FLAGS

005022  104000          1$:     ERROR
005024  104413          2$:     SUBSCOPE

                                ;T16 CONT. - TESTS FOR NO PARITY, CRC ERRORS, AND NO DELETED DATA BIT

005026  005002                  CLR R2
005030  005000                  CLR R0                          ;NO BITS SHOULD BE SET IN THE RXDB
005032  017701  174152          MOV @RXDB,R1
005036  020001                  CMP R0,R1
005040  001401                  BEQ 3$

                                ; R0 = 0 ; R1 = ACTUAL RXDB ; R2 = N/A
005042  104000                  ERROR                           ;SOME BIT IS SET IN THE RXDB
005044  104413          3$:     SUBSCOPE

                                ;T16 CONT. - TEST FOR CORRECT B-CODE FOR ILLEGAL SECTOR ADDRESS

005046  012777  000017  174132  MOV #17,@RXCS                   ;SET READ STATUS B FUNCTION
005054  004737  006574  4$:     JSR PC,SDN                      ;WAIT FOR DONE FLAG
005060  000775                  BR 4$
005062  012700  000070          MOV #70,R0
005066  017701  174116          MOV @RXDB,R1
005072  020001                  CMP R0,R1                       ;IS B-CODE = 70
005074  001401                  BEQ 5$                          ;YES, CONTINUE

                                ; R0 = 70 ; R1 = ACTUAL RXDB ; R2 = N/A
005076  104000                  ERROR
005100  000004          5$:     SCOPE
005102  000137  004260          JMP DEXIT                       ;END OF TEST 16
```

```
2134
2135                                    ;GENERATE AN ILLEGAL SECTOR ADDRESS
2136
2137  005106  004737  006560    ILLADR: JSR PC,STR              ;LOOK FOR A TR FLAG
2138  005112  000402                    BR 2$                   ;NO TR FLAG, IS DONE SET
2139  005114  005202                    INC R2                  ;TR FLAG COUNTER
2140  005116  000404                    BR 3$
2141  005120  004737  006574    2$:     JSR PC,SDN              ;LOOK FOR DONE FLAG
2142  005124  000770                    BR ILLADR               ;NOT DONE RECHECK TR FLAG
2143  005126  000430                    BR 1$                   ;DONE IS SET TOO EARLY GO TO ERROR
2144  005130  005077  174054    3$:     CLR @RXDB               ;0 SECTOR ADDRESS (ILLEGAL)
2145  005134  004737  006560    7$:     JSR PC,STR              ;LOOK FOR SECOND TR FLAG
2146  005140  000402                    BR 5$                   ;NOT TR, IS IT DONE
2147  005142  005202                    INC R2
2148  005144  000404                    BR 6$                   ;TR FLAG SEND TRACK ADDRESS
2149  005146  004737  006574    5$:     JSR PC,SDN              ;LOOK FOR DONE FLAG
2150  005152  000770                    BR 7$                   ;NOT DONE, RECHECK TR FLAG
2151  005154  000415                    BR 1$                   ;DONE TOO SOON GO TO ERROR
2152  005156  005077  174026    6$:     CLR @RXDB               ;SEND TRACK ADDRESS OF 0
2153  005162  004737  006560    11$:    JSR PC,STR              ;ARE THERE ANY MORE TR FLAGS
2154  005166  000402                    BR 10$                  ;NO, LOOK FOR DONE
2155  005170  005202                    INC R2                  ;YES
2156  005172  000406                    BR 1$                   ;TOO MANY TR FLAGS OR MICROCONTROLLER
2157                                                            ;DID NOT DETECT THE ERROR
2158  005174  004737  006574    10$:    JSR PC,SDN              ;LOOK FOR DONE FLAG
2159  005200  000770                    BR 11$                  ;NOT DONE RETEST TR FLAG
2160  005202  062716  000002          ADD #2,@SP
2161  005206  000207            4$:     RTS PC
2162  005210  017701  173772    1$:     MOV @RXCS,R1
2163  005214  000774                    BR 4$
```

```
2164
2165                                    .SBTTL TEST 17 - ILLEGAL TRACK ERROR VERIFICATION
2166
2167                                    ;THIS TEST VERIFIES THAT IF A TRACK ADDRESS LARGER THAN 114 (OCTAL) IS
2168                                    ;ACCESSED, AN ERROR CONDITION WILL EXIST, AND A B-CODE WILL = 40
2169
2170  005216  005002                    T17:    CLR R2
2171  005220  005000                            CLR R0
2172  005222  012777  000007  173756            MOV #7,@RXCS                    ;SET READ FUNCTION
2173  005230  004737  006560            3$:     JSR PC,STR                      ;LOOK FOR TR FLAG
2174  005234  000401                            BR 1$                           ;NO TR FLAG CHECK DONE
2175  005236  000410                            BR 2$
2176  005240  004737  006574            1$:     JSR PC,SDN
2177  005244  000771                            BR 3$
2178  005246  017701  173734                    MOV @RXCS,R1                    ;DONE OCCURRED TOO SOON SET UP FOR ERROR
2179  005252  017702  173732                    MOV @RXDB,R2
2180  005256  000433                            BR 4$
2181  005260  012777  000001  173722    2$:     MOV #1,@RXDB                    ;SEND LEGAL SECTOR ADDRESS
2182  005266  004737  006560            5$:     JSR PC,STR                      ;LOOK FOR TR FLAG
2183  005272  000401                            BR 6$
2184  005274  000410                            BR 7$
2185  005276  004737  006574            6$:     JSR PC,SDN
2186  005302  000771                            BR 5$
2187  005304  017701  173676                    MOV @RXCS,R1                    ;DONE SET TOO EARLY
2188  005310  017702  173674                    MOV @RXDB,R2
2189  005314  000414                            BR 4$
2190  005316  012777  000115  173664    7$:     MOV #115,@RXDB                  ;SEND ILLEGAL TRACK ADDRESS
2191  005324  004737  006574            10$:    JSR PC,SDN                      ;WAIT FOR DONE ON THE ERROR
2192  005330  000775                            BR 10$
2193  005332  012700  100040                    MOV #100040,R0                  ;EXPECT ERROR AND DONE SET
2194  005336  017701  173644                    MOV @RXCS,R1
2195  005342  020001                            CMP R0,R1
2196  005344  001401                            BEQ 11$
2197
2198                                    ;TWO ERROR CONDITIONS TO REPORT
2199                                    ;IF R0 = 0 THEN R1 = RXCS ;R2 = RXDB ON A DONE TOO SOON ERROR
2200                                    ;IF R0 = 100040 THEN R1 = ACTUAL RXCS ; R2 = N/A
2201
2202  005346  104000            4$:     ERROR                                   ;DONE SET TOO SOON OR NO ERROR OCCURRED
2203

        ;/*`:/*\:/*\:/*\:/*\:/*\:/*\:/*`:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/* :
        ;/*\:/*.:/*\:/*\:/*\:/*\:/*\:/*\:/*`:/*\:/*`:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

                ;THE FOLLOWING IS A PRESENTATION OF POSSIBLE REASONS AS TO WHY
                ;THIS ERROR REPORT WAS GENERATED.  THE INFORMATION SHOWN IS
                ;BASED ON FAULT INSERTION RESULTS,  AND SHOULD PROVIDE LOGICAL
                ;AREAS TO CHECK FOR THE RELEVANT FAULT/S.
                ;
                ;IF THIS ERROR REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS
                ;ANALYZE THE FOLLOWING AREA/S:
        ;
        ;M7846 (UNIBUS INTERFACE)
        ;
        ;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
```

```
                        ;
                        ; ----------            INT ENB FLOP CLOCK LOCKED HIGH  E40
                        ;
                        ;/: */:\*/:\*/: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
                        ;/:\*/: */:\*/:\*/: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
                        ;/: */: */:\*/: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/: */:\*

 2227  005350  104413                    11$:    SUBSCOPE
 2228
 2229                                    ;T17 CONT. - TEST THAT THERE WERE NO OTHER ERRORS THAN THE ILLEGAL TRACK ERROR EXPECTED
 2230                                    ;AND THAT THE DELETED DATA BIT WAS CLEARED BY TEST 16.
 2231
 2232  005352  005000                            CLR R0                          ;DD BIT CLEARED IN TEST 16 SO RXDB = 0
 2233  005354  005002                            CLR R2
 2234  005356  017701  173626                    MOV @RXDB,R1
 2235  005362  020001                            CMP R0,R1
 2236  005364  001401                            BEQ 12$
 2237
 2238                                    ; R0 = 0 ; R1 = ACTUAL RXDB ; R2 = N/A
 2239
 2240  005366  104000                            ERROR
 2241  005370  104413                    12$:    SUBSCOPE
 2242
 2243                                    ;T17 CONT. - VERIFIES THAT READ STATUS B HAS NO ERRORS
 2244
 2245  005372  012777  000017  173606            MOV #17,@RXCS                   ;SET READ STATUS B FUNCTION
 2246  005400  004737  006574            13$:    JSR PC,$DN                      ;WAIT FOR DONE FLAG
 2247  005404  000775                            BR 13$
 2248  005406  005777  173574                    TST @RXCS                       ;WAS THERE AN ERROR ON THIS FUNCTION
 2249  005412  100007                            BPL 14$                         ;NO
 2250  005414  012700  000040                    MOV #40,R0
 2251  005420  017701  173562                    MOV @RXCS,R1
 2252  005424  017702  173560                    MOV @RXDB,R2
 2253
 2254                                    ; R0 = 40 (DONE) ; R1 = ACTUAL RXCS ; R2 = ACTUAL RXDB
 2255
 2256  005430  104000                            ERROR                           ;THERE WAS AN ERROR NO READ STATUS B
 2257  005432  104413                    14$:    SUBSCOPE
 2258  005434  005002                            CLR R2
 2259  005436  012700  000040                    MOV #40,R0                      ;B-CODE FOR ILLEGAL TRACK ADDRESS
 2260  005442  017701  173542                    MOV @RXDB,R1
 2261  005446  020001                            CMP R0,R1
 2262  005450  001401                            BEQ 15$                         ;B-CODE IS CORRECT
 2263
 2264                                    ; R0 = 40 ; R1 = ACTUAL B-CODE ; R2 = N/A
 2265
 2266  005452  104000                            ERROR
 2267  005454  000004                    15$:    SCOPE
 2268  005456  000137  004260                    JMP @EXIT                       ;END OF TEST 17
```

```
2269
2270                                    .SBTTL TEST 20 - SEEK VERIFICATION VIA READ FUNCTION
2271
2272                                ; THE PURPOSE OF THIS TEST IS TO DO A READ FUNCTION  ON ALL
2273                                ;SECTORS OF VARIOUS TRACKS ON THE DISKETTE. THIS WILL TEST FOR SEEK ERRORS
2274                                ;FOR THOSE TRACK POSITIONS. UNLESS OTHERWISE SELECTED (IN OD/ID) THE TRACKS
2275                                ;ACCESSED ARE 0 (OD), 52, 53 (BOTH SIDES OF THE WRITE CURRENT CHANGE), AND 114 (ID).
2276
2277
2278  005462                        T20:
2279

        ;/* :/* :/*\:/*\:/*\:/*\:/* :/* :/*\:/*\:/*\:/*\:/*\:/*\:/* :/*\:/*\:/*\:
        :/* :/* :/* :/* :/*\:/*\:/* :/* :/*\:/*\:/*\:/*\:/*\:/*\:/* :/*\:/*\:/*\:
        : * :/* :/* :/* :/*\:/*\:/*`:/*`:/*\:/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

                ;
                ;IF THE DIAGNOSTIC GIVES AN ERROR REPORT WITH A FORMAT OF
                ;'TEST PC=' WHERE THE 'PC' IS WITHIN THE RANGE OF THIS TEST
                ;THEN THE POSSIBLE CHIPS VERSUS THE 'B' CODE (INTERPRETIVE
                ;ERROR CODE) PRINTED ARE AS FOLLOWS:
                ;
        ;       IF 'B' CODE WAS         POSSIBLE CHIPS
        ;
        ;               120             E5,E6
        ;               150             E13,E14,E16,E17
        ;               200             E5,E6
        ;
        ;/: */:\*/:\*/:\*/:\*/: */:\*/:`*/: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:`*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/: */:\*

2301  005462  013702  001204                MOV KRXVEC,R2                   ;SET UP INTERRUPT ADDRESSES
2302  005466  012722  011526                MOV #INTSERV,(R2)+
2303  005472  012722  000340                MOV #PR7,(R2)+
2304  005476  004737  006762                JSR PC,RDONLY                   ;DO THE READ FUNCTION
2305  005502  000137  004260                JMP DEXIT                       ;END OF TEST 20
2306
2307
2308                                    .SBTTL TEST 21 - WRITE TEST
2309
2310                                ;THE PURPOSE OF THIS TEST IS TO WRITE ALL ONES ON SECTOR 1 TRACK 1,
2311                                ;AND VERIFY THAT THE DATA IN THE SECTOR BUFFER IS NOT MODIFIED.
2312
2313  005506  012737  000001  013126  T21:   MOV #1,TARGET
2314  005514  012737  000001  013416        MOV #1,TSECTOR
2315  005522  004737  012634                JSR PC,GETUNIT
2316  005526  012737  000001  012324        MOV #1,PAT
2317  005534  004737  012260                JSR PC,GETPATTERN
2318  005540  004737  010766                JSR PC,ADJSUM                   ;SET CHECK SUM VALUES
2319  005544  005002                        CLR R2
2320  005546  012777  000001  173432        MOV #1,@RXCS                    ;SET FILL BUFFER FUNCTION
2321  005554  004737  004064          1$:   JSR PC,FBEB
2322  005560  000404                        BR 2$
2323  005562  112077  173422                MOVB (R0)+,@RXDB
2324  005566  005202                        INC R2
```

```
2325  005570  000771                    BR 1$
2326  005572  012737  000005  007702  2$:  MOV #5,FUNCTION          ;SET WRITE FUNCTION
2327  005600  004737  0056:0              JSR PC,T21X               ;GO ISSUE THE COMMAND
2328  005604  000137  0042:0              JMP DEXIT                 ;END OF TEST 21
```

```
 2329  005610  004737  007634          T21X:   JSR PC,COMMWORD
 2330  005614  004737  006574          3$:     JSR PC,SDN
 2331  005620  000775                          BR 3$
 2332  005622  005777  173360                  TST @RXCS                       ;IS THE ERROR BIT SET
 2333  005626  100013                          BPL 4$                          ;NO ERRORS
 2334  005630  017701  173354                  MOV @RXDB,R1                    ;SAVE STATUS A
 2335  005634  012777  000017  173344          MOV #17,@RXCS                   ;SET READ STATUS B FUNCTION
 2336  005642  004737  006574          5$:     JSR PC,SDN
 2337  005646  000775                          BR 5$
 2338  005650  017702  173334                  MOV @RXDB,R2                    ;SAVE STATUS B
 2339
 2340                                  ;R0 = ADDR OF LAST DATA BYTE ; R1 = STATUS A ; R2 = STATUS B
 2341
 2342  005654  104000                          ERROR
 2343

        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:

                ;IF THE FAULT CANNOT BE FOUND ON THE UNIBUS INTERFACE MODULE
                ;AND/OR THE FAULT IS NOT INHERENT TO THE UNIBUS INTERFACE MODULE
                ;M7846 THERE IS A POSSIBILITY OF ITS EXISTENCE ON THE READ/WRITE
                ;MODULE M7727.
        ;
        ;       NOTE:   ONLY APPROX. 30% OF THIS MODULE LENT ITSELF CONDUCIVE TO
        ;               THE FAULT INSERTION PROCESS; ERGO, THE RESOLUTION FOR FAULT
        ;               ANALYSIS OBTAINABLE BY THIS MODULE IS NOT VERY HIGH.
        ;               HOWEVER, ANALYSIS OF THE FOLLOWING AREA/S, IF THIS ERROR
        ;               REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS, SHOULD AT
        ;               LEAST PLACE YOU WITHIN THE RELEVANT AREA ON THE MODULE.
        ;
        ;M7727 (READ/WRITE CONTROL)
        ;
        ;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
        ;
        ;WT DATA                 STUCK LOW/HIGH                  E5
        ;WT GATE                 STUCK LOW                       E13
        ;
        ;*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

 2372  005656  104413          4$:     SUBSCOPE
 2373  005660  005000                  CLR R0
 2374  005662  005001                  CLR R1
 2375  005664  004737  004152          JSR PC,T7EMPTY                  ;EMPTY BUFFER AND CHECK CONTENTS
 2376  005670  000207                  RTS PC
```

```
                                        .SBTTL TEST 22 - INITIALIZE IMPLIED READ
                                ;AFTER PREVIOUSLY WRITING A PATTERN ON SECTOR 1 TRACK 1, THIS TEST
                                ;CHANGES THE CONTENTS OF THE SECTOR BUFFER AND DOES A PROGRAMMED INITIALIZE.
                                ;AFTER WHICH THE SECTOR BUFFER MUST AGAIN CONTAIN THE DATA PREVIOUSLY
                                ;WRITTEN ON THAT SECTOR AND TRACK.
                                ;NOTE:  THIS TEST WILL ONLY WORK IF UNIT 0 IS SELECTED AND ON LINE.

005672  105737  012724        T22:    TSTB UNITSEL                    ;IF UNIT 0 IS NOT SELECTED SKIP THIS TEST
005676  100022                        BPL 2$
005700  005037  012324                CLR PAT
005704  004737  004026                JSR PC,T6FILL                   ;LOAD THE SECTOR BUFFER WITH 0
005710  005237  012324                INC PAT                         ;RELOAD CORE BUFFER WITH 1'S
005714  004737  012260                JSR PC,GETPATTERN
005720  004737  010766                JSR PC,ADJSUM
005724  052777  040001  173254        BIS #RECAL,@RXCS                ;SET THE INIT. BIT
005732  004737  006574        1$:     JSR PC,SDN
005736  000775                        BR 1$
005740  004737  004152                JSR PC,T7EMPTY                  ;EMPTY THE SECTOR BUFFER AND CHECK IT.
005744  000137  004260        2$:     JMP DEXIT                       ;END OF TEST 22

                                        .SBTTL TEST 23 - READ TEST

                                ;THIS TEST VERIFIES THAT A READ FUNCTION DOES IN FACT LOAD THE SECTOR
                                ;BUFFER WITH DATA READ FROM THE SELECTED ADDRESS.

005750  005037  012324        T23:    CLR PAT
005754  004737  004026                JSR PC,T6FILL                   ;LOAD SECTOR BUFFER WITH 0'S
005760  005237  012324                INC PAT
005764  004737  012634                JSR PC,GETUNIT
005770  004737  012260                JSR PC,GETPATTERN               ;RELOAD CORE BUFFER WITH 1'S
005774  004737  010766                JSR PC,ADJSUM                   ;SET UP FOR CHECK SUM
006000  012737  000007  007702        MOV #7,FUNCTION                 ;SET READ FUNCTION AND GO
006006  004737  005610                JSR PC,T21X                     ;ISSUE COMMAND,WAIT FOR DONE,& TEST DATA
006012  000137  004260                JMP DEXIT                       ;END OF TEST 23
```

```
2413
2414                                    .SBTTL TEST 24 - DATA TRANSFER AND VERIFICATION
2415
2416                                ;THE PURPOSE OF THIS TEST IS TO WRITE THEN READ AND VERIFY DATA
2417                                ;ON ALL SECTORS OF THE SELECTED TRACKS. THE TEST ALTERNATES BETWEEN
2418                                ;DRIVES ON THE SELECTED TRACKS. DATA PATTERN IS A FLOATING 0.
2419
2420  006016  012737  000002  012324  T24:    MOV #2,PAT                      ;SET DATA PATTERN TO FLOATING 0
2421  006024  013702  001204          T24X:   MOV KRXVEC,R2                   ;SET INTERRUPT ADDRESSES
2422  006030  012722  011526                  MOV #INTSERV,(R2)+
2423  006034  012712  000340                  MOV #PR7,(R2)
2424  006040  004737  007012                  JSR PC,DRVSWP                   ;GO TRANSFER THE DATA
2425  006044  000137  004260                  JMP DEXIT                       ;END OF TEST 24 OR 25
2426
2427
2428                                    .SBTTL TEST 25 - DATA TRANSFER AND VERIFICATION VIA DELETED DATA MODE
2429
2430                                ;THIS TEST TRANSFERES DATA JUST LIKE TEST 24 EXCEPT IT USES THE
2431                                ;DELETED DATA FORMAT AND A DATA PATTERN OF FLOATING 1.
2432
2433  006050  012737  000003  012324  T25:    MOV #3,PAT                      ;SET DATA PATTERN TO FLOATING 1
2434  006056  000137  006024                  JMP T24X                        ;GO TRANSFER THE DATA
2435
2436                                    .SBTTL TEST 26 - HEAD "HOME" TEST
2437
2438                                ;THIS TEST MOVES THE HEAD OUT TO TRACK 12 (OCTAL) AND THEN WRITES/READ CHECKS
2439                                ;ALL SECTORS (RANDOM DATA) ON EACH TRACK. THE TRACK SEQUENCE
2440                                ;IS DECREMENT;>ED BACK TO TRACK 0 (HOME;. AFTER COMPLETING
2441                                ;DRIVE 0 IT SWITCHES OVER TO DRIVE 1 DOING THE SAME TEST.
2442
2443
2444  006062  052737  000200  013136  T26:    BIS #BIT7,SEQUEN                ;SPECIAL DECREMENT SEQUENCE
2445  006070  012737  000007  012324          MOV #7,PAT                      ;SELECT RANDOM DATA
2446  006076  013702  001204                  MOV KRXVEC,R2
2447  006102  012722  011526                  MOV #INTSERV,(R2)+
2448  006106  012712  000340                  MOV #PR7,(R2)
2449  006112  004737  007066                  JSR PC,WTRDCK
2450  006116  042737  000200  013136          BIC #BIT7,SEQUEN
2451  006124  000137  004260                  JMP DEXIT                       ;END OF TEST 26
```

```
2452
2453                                    ;THE FOLLOWING SECTION OF CODE WILL ALLOW PROVIDING INFORMATION
2454                                    ;TO THE USER WHEN AN 'UNEXPECTED' BUS TIMEOUT TO LOCATION 4 OCCURS
2455
2456  006130  104401  016750            BUSERR: TYPE,   LOC4M                   ;TYPE MESSAGE INDICATING AN
2457                                                                            ;UNEXPECTED BUS TIMEOUT OCCURRED
2458  006134  012646                            MOV     (SP)+,-(SP)     ;;SAVE (SP)+ FOR TYPEOUT
2459                                                                    ;;SETUP TO TYPE PC WHERE TIMEOUT OCCURRED
2460  006136  104403                            TYPOS                   ;;GO TYPE--OCTAL ASCII
2461  006140     006                           .BYTE   6               ;;TYPE 6 DIGITS
2462  006141     000                           .BYTE   0               ;;SUPPRESS LEADING ZEROS
2463  006142  104401  017063                    TYPE,   PCM                     ;TYPE MESSAGE '=PC'
2464  006146  012716  002466                    MOV     #REBEGIN,(SP)           ;SET RETURN 'PC' TO START THE
2465                                                                            ;PROGRAM OVER AGAIN
2466  006152  000002                            RTI                             ;RETURN TO BEGINNING OF PROGRAM
2467
2468                                    ;THE FOLLOWING SECTION OF CODE WILL ALLOW PROVIDING INFORMATION
2469                                    ;TO THE USER WHEN AN 'UNEXPECTED' RESERVED INSTRUCTION TRAP TO LOCATION
2470                                    ;10 OCCURS
2471
2472  006154  104401  017015            RESERR: TYPE,   LOC10M                  ;TYPE MESSAGE INDICATING AN
2473                                                                            ;UNEXPECTED RESERVED INSTRUCTION
2474                                                                            ;TRAP OCCURRED
2475  006160  012646                            MOV     (SP)+,-(SP)     ;;SAVE (SP)+ FOR TYPEOUT
2476                                                                    ;;SETUP TO TYPE PC WHERE RESERVED TRAP OCCURRED
2477  006162  104403                            TYPOS                   ;;GO TYPE--OCTAL ASCII
2478  006164     006                           .BYTE   6               ;;TYPE 6 DIGITS
2479  006165     000                           .BYTE   0               ;;SUPPRESS LEADING ZEROS
2480  006166  104401  017063                    TYPE,   PCM                     ;TYPE MESSAGE '=PC'
2481  006172  012716  002466                    MOV     #REBEGIN,(SP)           ;SET RETURN 'PC' TO START THE
2482                                                                            ;PROGRAM OVER AGAIN
2483  006176  000002                            RTI                             ;RETURN TO BEGINNING OF PROGRAM
2484
2485                                    ;THIS ROUTINE WILL CALCULATE THE PRIORITY LEVEL FOR THE PROCESSOR
2486                                    ;BASED ON THE CURRENT PRIORITY LEVEL OF THE DEVICE (CONTENTS OF 'BRLEV:')
2487
2488  006200  013700  001214            CPUPRI: MOV     BRLEV,R0                ;GET THE PROPOSED RX11 DEVICE
2489                                                                            ;INTERRUPT PRIORITY LEVEL VALUE
2490  006204  105701                            TSTB    R1                      ;IS CPU LEVEL TO BE THE SAME AS
2491                                                                            ;THE DEVICE LEVEL OR 1 LESS?
2492  006206  100401                            BMI     1$                      ;BRANCH IF SAME AS!
2493  006210  005300                            DEC     R0                      ;DROP DEVICE LEVEL PRIORITY
2494                                                                            ;BY 1 LEVEL FOR PSW
2495  006212  006300                    1$:     ASL     R0                      ;FORM BITS (7-5) FOR PSW
2496  006214  006300                            ASL     R0                      ;       .
2497  006216  006300                            ASL     R0                      ;       .
2498  006220  006300                            ASL     R0                      ;       .
2499  006222  006300                            ASL     R0                      ;       .
2500  006224  042700  000037                    BIC     #37,R0                  ;ENSURE THAT T,N,Z,V, & C BITS
2501                                                                            ;FOR THE PROCESSOR ARE CLEAR
2502  006230  000207                            RTS     PC                      ;RETURN TO MAINLINE CODE
2503
```

```
                        .SBTTL  " ERROR " TRAP SERVICE ROUTINE

                        ;;*****************************************************************
                        ;;*****************************************************************
                        ;                       " ERROR "
                        ;;*****************************************************************
                        ;;*****************************************************************
006232  011637  006474          XERROR: MOV @ SP, EPCSCOPE              ; RETURN ADDRESS FROM " ERROR"
006236  062737  000002  006474          ADD #2, EPCSCOPE                ; NOW (EPCSCOPE) = SUBSCOPE+2, OR SCOPE+2
006244  005237  006472          INCERRORS: INC ERRORS
006250  001775                          BEQ INCERRORS

                        ; DATA SW 13 = 0 TO PRINT APPROPRIATE ERROR MESSAGE

006252  104406                          CKSWR
006254  032777  020000  172734          BIT #SW13,@SWR
006262  001056                          BNE NOPRINT
006264  005037  002526                  CLR CCOUNT
006270  032737  000400  012724          BIT #BIT8,UNITSEL               ;WAS PREVIOUS ERROR REPORTED ON THIS PASS
006276  001002                          BNE 1$
006300  104401  015652                  TYPE, MXEHEADER

006304  104401  016134          1$:     TYPE, MCRLF
006310  011604                          MOV @ SP, R4                    ; ERADR
006312  162704  000002                  SUB #2, R4
006316  010446                          MOV R4, -(SP)
006320  104403                          TYPOS
006322     006                          .BYTE 6
006323     001                          .BYTE 1
006324  104401  016655                  TYPE, SPACE
006330  013746  002532                  MOV FAST, -(SP)                 ; FAST (FIRST ADDRESS OF SELECTED TEST)
006334  104404                          TYPON
006336  104401  016655                  TYPE, SPACE
006342  013746  006556                  MOV PCSCOPE, -(SP)              ; FAPT (FIRST ADDRESS OF PRESENT TEST)
006346  104404                          TYPON
006350  104401  016655                  TYPE, SPACE
006354  010246                          MOV R2, -(SP)                   ; BLANK
006356  104404                          TYPON
006360  104401  016655                  TYPE, SPACE
006364  010046                          MOV R0, -(SP)                   ; EXPECTED (GOOD) RESULT OF TEST
006366  104404                          TYPON
006370  104401  016655                  TYPE, SPACE
006374  010146                          MOV R1, -(SP)                   ; ACTUAL (BAD) RESULT OF TEST
006376  104404                          TYPON
006400  104401  016655                  TYPE, SPACE
006404  013737  002530  006416          MOV PASS,2$
006412  004537  015614                  JSR R5,SGLDEC
006416  000000                  2$:     OPEN
```

```
2555                                    ; DATA SW 0 = 0 TO RING BELL AT ERROR
2556
2557  006420  052737  000400  012724    NOPRINT: BIS #BIT8,UNITSEL          ;SET HARD ERROR FLAG
2558  006426  004737  006450                     JSR PC,DING
2559
2560                                    ; DATA SW 15 = 1 TO HALT AT ERROR
2561
2562  006432  104406                    1$:      CKSWR
2563  006434  032777  100000  172554              BIT #SW15,@SWR
2564  006442  001401                              BEQ 2$
2565  006444  000000                              HALT
2566  006446  000002                    2$:      RTI
2567
2568  006450  104406                    DING:    CKSWR
2569  006452  032777  000001  172536              BIT #SW0,@SWR
2570  006460  001002                              BNE 1$
2571  006462  104401  006470                      TYPE ,MABELL
2572  006466  000207                    1$:      RTS PC
2573
2574  006470  000007                    MABELL: .ASCIZ <07>                 ; DING - A - LING
2575                                            .EVEN
2576
2577  006472  000000                    ERRORS: 0
2578  006474  000000                    EPCSCOPE: 0
```

```
                        .SBTTL " SCOPE " TRAP SERVICE ROUTINE

                ;                       " SCOPE "

006476  005737  006472 XSCOPE: TST ERRORS
006502  001015          BNE SCOPING

                ; NO ERRORS HAVE BEEN DETECTED

                ; JUST SET (PCSCOPE) = FIRST ADDRESS OF THE SCOPE LOOP

                ; (IN CASE ERRORS ARE DETECTED LATER)

006504  005037  006472 NOSCOPE: CLR ERRORS
006510  011637  006556          MOV @ SP, PCSCOPE
006514  000002                  RTI

                ;                       " SUBSCOPE "

006516  005737  006472 XSUBSCOPE: TST ERRORS
006522  001001          BNE 1$
006524  000002          RTI                             ; NO ERRORS EXIST

                ; ERRORS DO EXIST

                ; IF THIS ERROR ADDRESS IS THE SAME ADDRESS WITHIN PROGRAM LOCATION " EPCSCOPE"

                ; THEN THIS IS A SCOPING LOOP

                ; IF NOT - THEN EXIT

006526  021637  006474 1$:     CMP @ SP, EPCSCOPE
006532  001401          BEQ SCOPING
006534  000002          RTI

                ; SW 11 = 1 TO LOCK ON SCOPING LOOP

                ; THIS IS A SCOPING LOOP

006536  104406  SCOPING: CKSWR
006540  032777  004000  172450  BIT #SW11,@SWR
006546  001756          BEQ NOSCOPE                     ;DO NOT LOOP ON ERROR
006550  013716  006556          MOV PCSCOPE, @ SP
006554  000002          RTI                             ; LOCK FOR SCOPE LOOP
006556  000000  PCSCOPE:       0
```

```
                                        ; WAIT FOR TRANSFER REQUEST FLAG TO SET
                                        ;
006560  105777  172422        STR:      TSTB @ RXCS
006564  100002                         BPL RTSPC
006566  062716  000002                 ADD #2, @ SP                    ; ADJUST FOR EXIT
006572  000207                 RTSPC:   RTS PC
                                        ;
                                        ; WAIT FOR THE DONE FLAG TO SET
                                        ;
006574  032777  000040  172404  SDN:     BIT #BIT5, @ RXCS               ; TEST THE DONE BIT # 5
006602  001055                         BNE XSDN
006604  062737  000001  006756         ADD #1,HANGER
006612  005537  006760                 ADC HANGPL
006616  001401                         BEQ HUNGUP
006620  000207                         RTS PC
                                        ;
                                        ; THE DEVICE TEST IS HUNG - DONE HAS NOT SET
                                        ;
                                        ; (R0) = N/A ; (R1) = N/A ; (R2) = N/A
                                        ;
006622  104000                 HUNGUP: ERROR

        ;/*\:/*\:/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/* :/*\:
        ;/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
        ;/* :/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:/*\:
                ;
                ;       IF THE DIAGNOSTIC HITS THIS ERROR REPORT AND THE 'HUNG @ PC'
                ;       LOCATION IS WITHIN ONE OF THE TESTS TABULATED BELOW THEN THE
                ;       POSSIBLE CHIPS ARE AS FOLLOWS:
                ;
        ;       HUNG @ TEST             POSSIBLE CHIPS
        ;
        ; TEST 5 ( PART 1 )             E7,E34,E4,E22,E15,E1,E11,E37
        ; TEST 5 ( PART 2 )             E7,E34,E2,E14,E6,E1,E37,E8
        ; TEST 6                        E4,E15,E1,E19,E18,E22
        ;
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/: */:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

                                        ;
006624  005037  006756                 CLR HANGER                      ;RESET THE HANG COUNTERS
006630  012737  177740  006760         MOV #177740,HANGPL
006636  104406                         CKSWR
006640  032777  020000  172350         BIT #SW13,@SWR
006646  001402                         BEQ 1$
006650  000137  002162                 JMP MORETESTS
006654  104401  006674         1$:     TYPE, MHUNGPC
                                        ;
                                        ; THE PC IS ALREADY ON THE STACK
                                        ;
006660  162716  000002                 SUB #2, @ SP
006664  104403                         TYPOS
006666     006                         .BYTE 6
006667     001                         .BYTE 1
006670  000137  002162                 JMP MORETESTS
```

```
2680
2681  006674  005015  042504  044526  MHUNGPC: .ASCIZ <15><12>"DEVICE TEST HUNG @ PC "
2682  006702  042503  052040  051505
2683  006710  020124  052510  043516
2684  006716  040040  050040  020103
2685  006724     000
2686  006725     040  040520  051523  MPASS:   .ASCIZ " PASS ="
2687  006732  036440     000
2688          006736                          .EVEN
2689
2690  006736  005037  006756          XSDN:    CLR HANGER
2691  006742  012737  177740  006760           MOV #177740,HANGPL
2692  006750  062716  000002                   ADD #2, @ SP                    ; UPDATE FOR EXIT
2693  006754  000207                           RTS PC
2694  006756  000000                  HANGER:  0
2695  006760  177740                  HANGPL:  177740
```

```
                                        .SBTTL DRIVE TEST SELECTION

                                        ;DO A READ ONLY FUNCTION ON ALL SECTORS.
                                        ;THIS DOES NOT VERIFY THE DATA, ONLY TESTS FOR CRC ERRORS.

006762  004737  012752        RDONLY:         JSR PC,INITTRACKS
006766  004737  012634                        JSR PC,GETUNIT
006772  004737  013104        1$:             JSR PC,GETTRACK
006776  004737  010014                        JSR PC,READ
007002  005337  013124                        DEC TRKCNTR
007006  001371                                BNE 1$
007010  000207                                RTS PC

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

                                        ;WRITE AND READ DATA ON SPECIFIED TRACK AND ALTERNATE
                                        ;DRIVES BEFORE GOING TO THE NEXT TRACK.

007012  004737  012260        DRVSWP:         JSR PC,GETPATTERN
007016  004737  012752                        JSR PC,INITTRACKS
007022  004737  012634        1$:             JSR PC,GETUNIT
007026  004737  013104                        JSR PC,GETTRACK
007032  004737  007134                        JSR PC,WRITE
007036  004737  010634                        JSR PC,READCHK
007042  004737  012634                        JSR PC,GETUNIT
007046  004737  007134                        JSR PC,WRITE
007052  004737  010634                        JSR PC,READCHK
007056  005337  013124                        DEC TRKCNTR
007062  001357                                BNE 1$
007064  000207                                RTS PC

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

                                        ;WRITE ALL SECTORS AND READ/VERIFY ALL SECTORS

007066  004737  012260        WTRDCK:         JSR PC,GETPATTERN
007072  004737  012752        XWTRDCK:        JSR PC,INITTRACKS
007076  004737  012634                        JSR PC,GETUNIT
007102  004737  013104        1$:             JSR PC,GETTRACK
007106  004737  007134                        JSR PC,WRITE
007112  004737  010634                        JSR PC,READCHK
007116  005337  013124                        DEC TRKCNTR
007122  001367                                BNE 1$
007124  004737  012726                        JSR PC,DONE             ;HAVE BOTH DRIVES BEEN TESTED
007130  000207                                RTS PC                  ;YES
007132  000757                                BR XWTRDCK              ;NO. GO TO OTHER UNIT
```

```
                                        .SBTTL  WRITE FUNCTION
                                
007134  004737  013324          WRITE:        JSR PC,INITSECTOR       ;SET UP FIRST,LAST,AND SECTOR COUNTER
007140  004737  013422          XWRITE:       JSR PC,GETSECTOR        ;PICK UP NEXT SECTOR
007144  004737  010766          FILLBUF:      JSR PC,ADJSUM           ;ADJUST DATA BUFFER AND CHECK SUM FOR ADDRESSES
007150  012746  007332                        MOV #FILLDONE,-(SP)     ;PUT GOOD RETURNON STACK
007154  012746  007222                        MOV #FILLER,-(SP)       ;PUT ERROR RETURN ON STACK
007160  005037  011452                        CLR BYTECNTR
007164  005046                        CLR     -(SP)                   ;LOWER 'CPU' LEVEL
007166  012746  007174                MOV     #1$,-(SP)               ;SET RETURN 'PC'
007172  000002                      RTI                             ;GET 'CPU' LEVEL INTO 'PSW'
007174  012777  000101  172004  1$:           MOV #FBIE,@RXCS         ;EXECUTE FILLBUFER COMMAND
007202  105777  172000          FILLFLAG:     TSTB @RXCS              ;TEST FOR TRANSFER REQUEST FLAG
007206  100375                              BPL FILLFLAG
007210  112077  171774          XFRBYTE:      MOVB (R0)+,@RXDB        ;TRANSFER DATA BYTE
007214  005237  011452                        INC BYTECNTR
007220  000770                              BR FILLFLAG             ;WAIT FOR NEXT TR FLAG

007222  005726                  FILLER:       TST (SP)+               ;REMOVE THE DONE RETURN FROM THE STACK
007224  012737  016402  007266              MOV #MFIL,PTYP1+2       ;PUT ADDR OF FILLBUF MESSAGE IN PAR ERR TYPOUT 1
007232  012737  007140  007330              MOV #XWRITE,PCONT+2     ;IF NO LOOP ON ERROR GO TO NEXT SECTOR
007240  012737  007144  007324              MOV #FILLBUF,PLOOP+2    ;IF LOOP ON ERROR RETURN THROUGH PLOOP
007246  000137  007252                      JMP PARTEST             ;PRINT OUT PAR ERR AND TEST CONDITIONS FOR RETRY

007252  104406                  PARTEST:      CKSWR
007254  032777  020000  171734              BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
007262  001006                              BNE CONT4
007264  104401  000000          PTYP1:        TYPE ,OPEN              ;PRINT THE PARITY ERROR MESSAGE
007270  104401  016613                      TYPE ,MPAR
007274  104401  016134                      TYPE ,MCRLF
007300  104406                  CONT4:        CKSWR
007302  005777  171710                      TST @SWR                ;TEST HALT ON ERROR SWITCH
007306  100001                              BPL CONT13
007310  000000                  HLT6:         HALT                    ;HALT ON ERROR
007312  032777  004000  171676  CONT13:      BIT #SW11,@SWR          ;TEST LOOP ON ERROR SWITCH
007320  001402                              BEQ PCONT               ;IF NOT SET GO TO NEXT SECTOR
007322  000137  007144          PLOOP:        JMP FILLBUF             ;RETURN TO LOOP ON TEST THROUGH HERE
007326  000137  010132          PCONT:        JMP NEXTRD              ;GO TO NEXT SECTOR THROUGH HERE

007332  005037  006756          FILLDONE:     CLR HANGER
007336  012746  007430                        MOV #WRTDONE,-(SP)      ;SET GOOD RETURN ON STACK
007342  012746  007444                        MOV #WRTER,-(SP)        ;SET ERROR RETURN ON STACK
007346  112737  000105  007702              MOVB #WRTIE,FUNCTION    ;SET FUNCTION WORD TO WRITE
007354  022737  006050  006556              CMP #T25,PCSCOPE        ;IS THIS THE DELETED DATA TEST
007362  001003                              BNE 1$
007364  112737  000115  007702              MOVB #WTDDIE,FUNCTION
007372  004737  007634          1$:           JSR PC,COMMWORD         ;TRANSFER COMMAND TO DRIVE
007376  005046                        CLR     -(SP)                   ;LOWER 'CPU' LEVEL
007400  012746  007406                MOV     #2$,-(SP)               ;SET RETURN 'PC'
007404  000002                      RTI                             ;GET 'CPU' LEVEL INTO 'PSW'
007406  032777  000040  171572  2$:           BIT #DONEBIT,@RXCS      ;WAIT FOR DONE
007414  001774                              BEQ 2$
007416  005237  006756          3$:           INC HANGER              ;WAIT FOR INTERRJPT
007422  001375                              BNE 3$
007424  000137  011462                      JMP NOINTER             ;NO INTERRUPT ERROR
```

```
2798  007430  005337  013414          WRTDONE:        DEC SECCNTR             ;TEST SECTOR COUNTER
2799  007434  001001                                  BNE 2$                  ;NOT LAST SECTOR GO TO NEXT ONE
2800  007436  000207                                  RTS PC
2801  007440  000137  007140          2$:             JMP XWRITE
2802
2803  007444  005726                  WRTER:          TST (SP)+               ;REMOVE THE DONE RETURN FROM THE STACK
2804  007446  032737  000002  012146                  BIT #BIT1,ASTAT         ;IS THIS A PARITY ERROR
2805  007454  001413                                  BEQ WRTSEK              ;NO, IT MUST BE A SEEK ERROR
2806                                ;PARITY ERROR DURING A WRITE FUNCTION
2807  007456  012737  016604  007266                  MOV #MWRITE,PTYP1+2     ;PUT ADDR OF WRITE MESSAGE IN PAR ER TYPOUT 1
2808  007464  012737  007430  007330                  MOV #WRTDONE,PCONT+2    ;IF NO LOOP GO TO NEXT SECTOR
2809  007472  012737  007332  007324                  MOV #FILLDONE,PLOOP+2   ;IF LOOP RETURN THROUGH PLOOP TO REWRITE
2810  007500  000137  007252                          JMP PARTEST             ;GO INC LOG AND TEST FOR RETRY
2811
2812                                ;SEEK ERROR DURING A WRITE FUNCTION
2813  007504  012737  007144  007604  WRTSEK:         MOV #FILLBUF,SEKRTY+2   ;SETUP FOR WRT RETRY ON SEEK ERROR
2814                                                                          ;(AFTER A RECAL. THE CONTENTS OF SECTOR 1,
2815                                                                          ;TRACK 1 ARE LOADED INTO THE SECTOR BUFFER.
2816                                                                          ;TO REWRITE THE CORRECT DATA THE PROGRAM
2817                                                                          ;MUST REFILL THE SECTOR BUFFER.
2818  007512  012737  016604  007542                  MOV #MWRITE,STYP1+2     ;PUT ADDR OF WRITE MESSAGE IN SEEK ER TYPEOUT 1
2819  007520  004737  007526                          JSR PC,SEEKER           ;RECORD SEEK ERROR
2820  007524  000741                                  BR WRTDONE              ;GO TO NEXT SECTOR CAN'T FIND THIS ONE
2821
2822  007526  104406                  SEEKER:         CKSWR
2823  007530  032777  020000  171460                  BIT #SW13,@SWR          ;CHECK DON'T PRINT ERROR SWITCH
2824  007536  001004                                  BNE SWHLT1
2825  007540  104401  016604          STYP1:          TYPE ,MWRITE            ;PRINT WRITE (READ) SEEK ERROR
2826  007544  004737  007606                          JSR PC,SEKTYP
2827  007550  104406                  SWHLT1:         CKSWR
2828  007552  005777  171440                          TST @SWR                ;TEST THE HALT ON ERROR SWITCH
2829  007556  100001                                  BPL CONT14
2830  007560  000000                  HLT7:           HALT                    ;HALT ON THE ERROR
2831  007562  004737  007706          CONT14:         JSR PC,HOME             ;RECALIBRATE ON SEEK ERRORS
2832  007566  104406                                  CKSWR
2833  007570  032777  004000  171420                  BIT #SW11,@SWR          ;CHECK THE LOOP ON ERROR SWITCH
2834  007576  001001                                  BNE SEKRTY              ;IF SET LOOP ON THE ERROR THROUGH SEEK RETRY.
2835  007600  000207                                  RTS PC
2836  007602  000137  007144          SEKRTY:         JMP FILLBUF             ;RETRY WRITE COMMAND (READ COMAND)
2837
2838  007606  104401  016571          SEKTYP:         TYPE ,MSEEK             ;TYPE SEEK ERROR
2839  007612  104401  016046                          TYPE ,MPRES             ;TYPE ADDRESS OF TRACK MOVED FROM
2840  007616  013746  013130                  MOV     PRESTRK,-(SP)   ;;SAVE PRESTRK FOR TYPEOUT
2841  007622  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
2842  007624     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
2843  007625     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
2844  007626  104401  016134                          TYPE ,MCRLF
2845  007632  000207                                  RTS PC
2846
```

```
2847  007634  153737  012724  007702  COMMWORD:       BISB UNITSEL,FUNCTION   ;SET UNIT SELECTION BIT IN COMMAND WORD
2848  007642  013777  007702  171336                  MOV FUNCTION,@RXCS      ;SEND OUT COMMAND TO DRIVE
2849  007650  004737  006560          1$:             JSR PC,STR              ;WAIT FOR TR FLAG
2850  007654  000775                                  BR 1$
2851  007656  113777  013416  171324                  MOVB TSECTOR,@RXDB      ;SEND OUT TARGET SECTOR
2852  007664  004737  006560          2$:             JSR PC,STR              ;WAIT FOR TR FLAG
2853  007670  000775                                  BR 2$
2854  007672  113777  013126  171310                  MOVB TARGET,@RXDB       ;SEND OUT TARGET TRACK
2855  007700  000207                                  RTS PC
2856
2857  007702  000000          FUNCTION:       0
2858  007704  000000          DATACK:         0                               ;DATA CHECK ON CRC ERROR FLAG
2859
2860  007706  104406          HOME:           CKSWR
2861  007710  032777  000400  171300                  BIT #SW8,@SWR           ;TEST NO RECAL SWITCH
2862  007716  001035                                  BNE RTN
2863  007720  012777  040001  171260                  MOV #RECAL,@RXCS        ;ISSUE RECAL FUNCTION
2864  007726  004737  006574          2$:             JSR PC,SDN
2865  007732  000775                                  BR 2$
2866  007734  005777  171246                        TST @RXCS               ;WAS THERE AN ERROR
2867  007740  100021                                  BPL XHOME               ;NO
2868  007742  104406                                  CKSWR
2869  007744  032777  020000  171244                  BIT #BIT13,@SWR         ;YES, SHOULD IT BE PRINTED
2870  007752  001002                                  BNE 1$                  ;NO
2871  007754  004737  012152                        JSR  PC,RDCODE
2872  007760  104406          1$:             CKSWR
2873  007762  005777  171230                        TST @SWR                ;TEST HALT ON ERROR SWITCH
2874  007766  100001                                  BPL 3$
2875  007770  000000                                  HALT
2876  007772  032777  004000  171216  3$:             BIT #SW11,@SWR          ;TEST LOOP ON ERROR SWITCH
2877  010000  001342                                  BNE HOME
2878  010002  000207                                  RTS PC
2879  010004  012737  000001  013130  XHOME:         MOV #1,PRESTRK          ;SET THE PRESENT TRACK TO TRACK 1
2880  010012  000207          RTN:            RTS PC
2881
```

```
                        .SBTTL  READ DATA FROM THE DISKETTE

010014  004737  013324        READ:       JSR PC,INITSECTOR
010020  004737  013422        XREAD:      JSR PC,GETSECTOR
010024  005037  007704        REREAD:     CLR DATACK              ;CLEAR CRC DATA CHECK FLAG
010030  005037  006756                    CLR HANGER
010034  012746  010110                    MOV #RDDONE,-(SP)       ;SET GOOD RETURN ON STACK
010040  012746  010142                    MOV #RDERR,-(SP)        ;SET READ ERROR RETURN ON STACK
010044  112737  000107  007702            MOVB #RDIE,FUNCTION
010052  004737  007634                    JSR PC,COMMWORD
010056  005046                    CLR     -(SP)                   ;LOWER 'CPU' LEVEL
010060  012746  010066            MOV     #1$,-(SP)               ;SET RETURN 'PC'
010064  000002                    RTI                             ;GET 'CPU' LEVEL INTO 'PSW'
010066  032777  000040  171112    1$:     BIT #DONEBIT,@RXCS      ;WAIT FOR DONE BIT
010074  001774                    BEQ 1$
010076  005237  006756            2$:     INC HANGER              ;WAIT FOR INTERRUPT
010102  001375                    BNE 2$
010104  000137  011462 -                  JMP NOINTER             ;NO INTERRUPT ON DONE

010110  022737  005452  006556    RDDONE:     CMP #T20,PCSCOPE    ;IS THIS THE READ ONLY TEST (T20)
010116  001405                    BEQ NEXTRD                      ;YES,DON'T CHECK FOR DELETED DATA
010120  004737  010410                    JSR PC,DDCHK            ;CHECK FOR DELETED DATA INDICATOR
010124  005701                    TST R1                          ;BIT 15 OF R1 IS READ 1 SECTOR FLAG
010126  100001                    BPL NEXTRD
010130  000207                    RTS PC                          ;IF SET,GO VERIFY DATA JUST READ
010132  005337  013414            NEXTRD:     DEC SECCNTR
010136  001330                    BNE XREAD
010140  000207                    RTS PC                          ;READ FUNCTION IS DONE

010142  005726                    RDERR:      TST (SP)+           ;REMOVE THE DONE RETURN FROM THE STACK
010144  032737  000002  012146              BIT #BIT1,ASTAT         ;IS THIS A PARITY ERROR
010152  001413                    BEQ 1$                          ;NO, SEE IF ITS A CRC ERROR
                        ;PARITY ERROR DURING A READ FUNCTION
010154  012737  016544  007266              MOV #MREAD,PTYP1+2      ;PUT ADDR OF READ MESSAGE IN PAR ERR TYPEOUT 1
010162  012737  010024  007324              MOV #REREAD,PLOOP+2     ;IF LOOP ON ERROR LOOP THROUGH PLOOP
010170  012737  010132  007330              MOV #NEXTRD,PCONT+2     ;IF NO LOOP GO TO NEXT READ
010176  000137  007252                      JMP PARTEST             ;RECORD PARITY ERROR AND RETRY FUNCTION
010202  032737  000001  012146  1$:         BIT #BIT0,ASTAT         ;IS THIS A CRC ERROR
010210  001011                    BNE CRCER                       ;YES GO TEST AND LOG IT
                        ;SEEK ERROR DURING A READ FUNCTION
010212  012737  010024  007604              MOV #REREAD,SEKRTY+2    ;SET SEEK CONTINUE FOR READ RETRY
010220  012737  016544  007542              MOV #MREAD,STYP1+2      ;SET ADDR OF READ MESSAGE IN SEEK ER TYPEOUT 1
010226  004737  007526                      JSR PC,SEEKER           ;RECORD SEEK ERROR
010232  000737                            BR NEXTRD               ;GO TO NEXT SECTOR,CAN'T READ THIS ONE
```

```
                                        ;CRC ERROR DETECTED WHILE READING
                                
010234  005701                  CRCER:  TST R1                  ;IF READ ONLY, REPORT DATA CRC ERROR
010236  100034                          BPL DATACRC
010240  005237  007704                  INC DATACK              ;SET DATA CHECK FLAG
010244  004737  010644                  JSR PC,EMPBUFF          ;CHECK FOR A DATA ERROR
010250  005737  011460                  TST ERCNTR              ;WAS THERE A DATA ERROR
010254  001025                          BNE DATACRC             ;YES, REPORT IT
010256  104406                          CKSWR
010260  032777  020000  170730          BIT #SW13,@SWR          ;TEST DON'T PRINT SWITCH
010266  001004                          BNE 2$
010270  104401  016514                  TYPE ,MBADCRC           ;TYPE CRC GENERATOR ERROR
010274  104401  016134                  TYPE ,MCRLF
010300  104406                  2$:     CKSWR
010302  005777  170710                  TST @SWR                ;TEST HALT ON ERROR SWITCH
010306  100001                          BPL CONT15
010310  000000                  HLT10:  HALT                    ;HALT ON ERROR
010312  032777  004000  170676  CONT15: BIT #SW11,@SWR          ;CHECK LOOP ON ERROR SWITCH
010320  001001                          BNE 3$
010322  000703                          BR NEXTRD               ;DON'T LOOP GO TO NEXT SECTOR
010324  000137  010024          3$:     JMP REREAD              ;LOOP ON TEST.
                                
                                        ;DATA CRC ERROR
                                
010330  104406                  DATACRC: CKSWR
010332  032777  020000  170656          BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
010340  001004                          BNE 4$
010342  104401  016552                  TYPE ,MCRC              ;TYPE DATA CRC ERROR
010346  104401  016134                  TYPE ,MCRLF
010352  104406                  4$:     CKSWR
010354  005777  170636                  TST @SWR                ;TEST HALT ON ERROR SWITCH
010360  100001                          BPL CONT16
010362  000000                  HLT12:  HALT                    ;HALT ON ERROR
010364  032777  004000  170624  CONT16: BIT #SW11,@SWR          ;TEST LOOP ON ERROR
010372  001004                          BNE 5$                  ;IF SET LOOP ON THE TEST
010374  062706  000002                  ADD #2,SP               ;REMOVE READ DONE ADDRESS FROM STACK
010400  000137  010132                  JMP NEXTRD              ;READ NEXT SECTOR CAN'T READ THIS ONE
010404  000137  010024          5$:     JMP REREAD              ;NO,GO REREAD THIS SECTOR
```

```
2967  010410  022737  006050  006556  DDCHK:      CMP #T25,PCSCOPE        ;IS THIS TEST 25
2968  010416  001041                              BNE CONT10
2969  010420  132737  000100  012146              BITB #BIT6,@STAT        ;THIS IS TEST 25
2970  010426  001056                              BNE RETURN              ;DD BIT SHOULD BE SET
2971  010430  104406                              CKSWR
2972  010432  032777  020000  170556              BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
2973  010440  001013                              BNE CONT11
2974  010442  004737  010566                      JSR PC,ERMSG
2975  010446  104401  015763                      TYPE ,MDDMIS            ;TYPE MISSING DELETED DATA BIT
2976  010452  052737  000400  012724  DDERR:      BIS #BIT3,UNITSEL       ;SET HARD ERROR FLAG
2977  010460  004737  012050                      JSR PC,TYPADR           ;TYPE ADDRESS OF ERROR
2978  010464  104401  016134                      TYPE ,MCRLF
2979  010470  104406                  CONT11:     CKSWR
2980  010472  005777  170520                      TST @SWR                ;TEST HALT ON ERROR SWITCH
2981  010476  100001                              BPL CONT17
2982  010500  000000                  HLT13:      HALT                    ;HALT ON DELETED DATA ERROR
2983  010502  032777  004000  170506  CONT17:     BIT #SW11,@SWR          ;TEST LOOP ON ERROR
2984  010510  001402                              BEQ 4$
2985  010512  000137  010024                      JMP REREAD              ;LOOP ON TEST
2986  010516  000137  010132          4$:         JMP NEXTRD              ;READ NEXT SECTOR
2987  010522  032737  000100  012146  CONT10:     BIT #BIT6,@STAT         ;THIS IS NOT A DELETED DATA TRANSFER
2988  010530  001415                              BEQ RETURN
2989  010532  052737  000400  012724              BIS #BIT8,UNITSEL       ;SET HARD ERROR FLAG
2990  010540  104406                              CKSWR
2991  010542  032777  020000  170446              BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
2992  010550  001347                              BNE CONT11
2993  010552  004737  010566                      JSR PC,ERMSG
2994  010556  104401  015735                      TYPE ,MUNXDD            ;TYPE UNEXPECTED DELETED DATA BIT
2995  010562  000733                              BR DDERR
2996  010564  000207                  RETURN:     RTS PC
2997
2998
2999  010566  104401  016137          ERMSG:      TYPE ,MERHEADER
3000  010572  013746  006556                  MOV     PCSCOPE,-(SP)   ;;SAVE PCSCOPE FOR TYPEOUT
3001  010576  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3002  010600     006                          .BYTE   6               ;;TYPE 6 DIGITS
3003  010601     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
3004  010602  104401  006725                  TYPE ,MPASS
3005  010606  013737  002530  010620          MOV PASS,1$
3006  010614  004537  015614                  JSR R5,SGLDEC
3007  010620  000000                  1$:     OPEN
3008  010622  104401  016134                  TYPE ,MCRLF
3009  010626  004737  006450                  JSR PC,DING
3010  010632  000207                          RTS PC
3011
3012
```

```
                                        .SBTTL READ AND VERIFY DATA

                                        ;READ A SECTOR,EMPTY THE SECTOR BUFFER AND VERIFY
                                        ;THE DATA READ AGAINST CORE DATA BUFFER

010634  052701  100000        READCHK:  BIS #BIT15,R1           ;SET READ ONE SECTOR FLAG
010640  004737  010014                  JSR PC,READ             ;GO READ ONE SECTOR
010644  005737  013414        EMPBUFF:  TST SECCNTR             ;IF CLEARED NO SECTORS WERE FOUND
010650  001002                          BNE 1$
010652  000137  011446                  JMP EXIT                ;GO TO NEXT TRACK
010656  005037  011452        1$:       CLR BYTECNTR            ;CLEAR THE BYTE AND ERROR COUNTERS
010662  005037  011460                  CLR ERCNTR
010666  052701  000200                  BIS #BIT7,R1            ;R1 BIT 7 IS USED AS FIRST ERROR FLAG
010672  004737  010766                  JSR PC,ADJSUM           ;ADJUST DATA AND CK SUM FOR ADDRESSES
010676  005037  011056                  CLR CKSUM               ;SET UP FOR CHECK SUM ACCUMULATION
010702  012746  011306                  MOV #EMPDONE,-(SP)      ;SET UP RETURN ADDRESSES
010706  012746  011060                  MOV #EMPER,-(SP)
010712  005046                  CLR     -(SP)                   ;LOWER 'CPU' LEVEL
010714  012746  010722          MOV     #2$,-(SP)               ;SET RETURN 'PC'
010720  000002                  RTI                             ;GET 'CPU' LEVEL INTO 'PSW'
010722  012777  000103  170256  2$:       MOV #EBIE,@RXCS         ;LOAD EMPTY BUFFER FUNCTION
010730  105777  170252        EMPFLAG:  TSTB @RXCS              ;TEST FOR TR FLAG
010734  100375                          BPL EMPFLAG

010736  117737  170246  011454  CKBYTE:   MOVB @RXDB,BADBYTE     ;SAVE BYTE FROM DISKETTE
010744  063737  011454  011056            ADD BADBYTE,CKSUM       ;ACCUMULATE CHECK SUM
010752  123720  011454                    CMPB BADBYTE,(R0)+      ;COMPARE AGAINST GOOD BYTE
010756  001054                          BNE DATAER              ;IF NOT EQUAL GO TO DATAER
010760  005237  011452                  INC BYTECNTR
010764  000761                          BR EMPFLAG              ;GET NEXT BYTE

010766  113737  013126  017420  ADJSUM:   MOVB TARGET,BUFADR      ;SET FIRST AND SECOND BYTES WITH ADDRESSES
010774  113737  013416  017421            MOVB TSECTOR,BUFADR+1
011002  013737  012536  011056            MOV SUM,CKSUM           ;GET THE PATTERN SUM
011010  063737  013126  011056            ADD TARGET,CKSUM        ;ADD TRACK ADDRESS TO CHECK SUM
011016  063737  013416  011056            ADD TSECTOR,CKSUM       ;ADD SECTOR ADDRESS TO CHECK SUM
011024  113737  011056  017616            MOVB CKSUM,BUFADR+176   ;INSERT CHECK SUM TO DATA BUFFER
011032  106337  011056                    ASLB CKSUM              ;GENERATE NEGITIVE CHECK SUM
011036  105437  011056                    NEGB CKSUM
011042  113737  011056  017617            MOVB CKSUM,BUFADR+177   ;INSERT NEG.SUM INTO DATA BUFFER
011050  012700  017420                    MOV #BUFADR,R0          ;SET ADDRESS OF BYTE IN R0
011054  000207                          RTS PC                  ;RETURN

011056  000000        CKSUM:    0

011060  005726        EMPER:    TST (SP)+               ;REMOVE THE DONE RETURN FROM THE STACK
011062  012737  016416  007266            MOV #MEMPTY,PTYP1+2     ;PUT ADDR OF EMPTYBUF MESSAGE IN PAR ER TYPOUT 1
011070  012737  010644  007324            MOV #EMPBUFF,PLOOP+2    ;RETURN THROUGH HERE TO LOOP ON ERROR
011076  012737  011430  007330            MOV #NXREAD,PCONT+2     ;IF NO LOOP ON ERROR GO TO NEXT SECTOR
011104  000137  007252                    JMP PARTEST             ;REPORT PARITY ERROR
```

```
3064  011110  052737  000400  012724  DATAER:   BIS #BIT8,UNITSEL      ;SET THE HAD ERROR FLAG
3065  011116  005237  011460                    INC ERCNTR            ;INC THE BYTE ERROR COUNTER
3066  011122  104406                            CKSWR
3067  011124  032777  020000  170064            BIT #SW13,@SWR        ;TEST PRINT ERROR SW IN SWR
3068  011132  001054                            BNE NOERTYP           ;DON'T PRINT THE ERROR
3069  011134  032777  001000  170054            BIT #SW9,@SWR         ;TEST PRINT 10 ERRORS SWITCH
3070  011142  001004                            BNE 1$                ;IF SET PRINT ALL ERRORS
3071  011144  023727  011460  000012            CMP ERCNTR,#10.       ;HAVE 10 ERRORS BEEN TYPED
3072  011152  003044                            BGT NOERTYP           ;YES,DON'T PRINT ANY MORE
3073  011154  105701                    1$:     TSTB R1               ;TEST FIRST ERROR FLAG
3074  011156  100014                            BPL TYPERR
3075  011160  004737  010566                    JSR PC,ERMSG          ;PRINT ADDRESS OF TEST
3076  011164  104401  016006                    TYPE ,MDERHDR         ;FIRST ERROR, PRINT ERROR HEADER
3077  011170  104401  016134                    TYPE ,MCRLF
3078  011174  004737  012050                    JSR PC,TYPADR         ;PRINT TRACK AND SECTOR LOCATIONS
3079  011200  104401  016075                    TYPE ,MCOLMUN         ;SET UP COLMUN HEADINGS
3080  011204  042701  000200                    BIC #BIT7,R1          ;CLEAR FIRST ERROR FLAG
3081  011210  013737  011452  011222  TYPERR:   MOV BYTECNTR,1$       ;PRINT BYTE NUMBER
3082  011216  004537  015614                    JSR R5,SGLDEC
3083  011222  000000                    1$:     OPEN
3084  011224  104401  016131                    TYPE ,DBLSP
3085  011230  013746  011454                    MOV BADBYTE,-(SP)      ;PRINT BYTE READ FROM DISKETTE
3086  011234  104403                            TYPOS
3087  011236  000003                            .WORD 3
3088  011240  104401  016131                    TYPE ,DBLSP
3089  011244  114037  011456                    MOVB -(R0),GOODBYTE    ;GET GOOD BYTE
3090  011250  005200                            INC R0                 ;RETURN R0 TO NEXT BYTE IN BUFFER
3091  011252  013746  011456                    MOV GOODBYTE,-(SP)
3092  011256  104404                            TYPON                  ;PRINT GOOD DATA
3093  011260  104401  016134                    TYPE ,MCRLF
3094  011264  104406            NOERTYP:        CKSWR
3095  011266  005777  167724                    TST @SWR               ;TEST HALT ON ERROR SWITCH
3096  011272  100001                            BPL CONT20
3097  011274  000000            HLT14:          HALT
3098  011276  005237  011452    CONT20:         INC BYTECNTR
3099  011302  000137  010730                    JMP EMPFLAG
3100
```

```
3101  011306  005737  007704          EMPDONE:        TST DATACK              ;WAS THIS READ CHECK CAUSED BY A CRC ERROR
3102  011312  001401                                  BEQ 1$                  ;NO
3103  011314  000207                                  RTS PC                  ;YES,RETURN TO CRC HANDLER
3104  011316  005737  011460          1$:             TST ERCNTR              ;WAS THERE ERRORS
3105  011322  001442                                  BEQ NXREAD              ;NO ERRORS
3106  011324  104406                                  CKSWR
3107  011326  032777  020000  167662                  BIT #SW13,@SWR          ;YES,TEST DON'T PRINT SWITCH
3108  011334  001024                                  BNE 2$                  ;DON'T PRINT THE ERROR
3109  011336  104401  016347                          TYPE ,MERCT             ;PRINT THE TOTAL DATA ERROR COUNT
3110  011342  013737  011460  011354                  MOV ERCNTR,3$
3111  011350  004537  015614                          JSR R5,SGLDEC
3112  011354  000000                  3$:             OPEN
3113  011356  104401  016664                          TYPE ,MSUM              ;INDICATE IF CHECK SUM WAS GOOD OR HAD ERRORS
3114  011362  105737  011056                          TSTB CKSUM
3115  011366  001403                                  BEQ 4$
3116  011370  104401  016651                          TYPE ,MBAD
3117  011374  000402                                  BR 5$
3118  011376  104401  016657          4$:             TYPE,MGOOD
3119  011402  104401  016134          5$:             TYPE ,MCRLF
3120  011406  104406                  2$:             CKSWR
3121  011410  032777  004000  167600                  BIT #SW11,@SWR          ;TEST LOOP ON ERROR SWITCH
3122  011416  001404                                  BEQ NXREAD              ;IF NOT SET GO TO NEXT SECTOR
3123  011420  004737  010024                          JSR PC,REREAD           ;YES,GO REREAD THE DATA
3124  011424  000137  010644                          JMP EMPBUFF             ;GO RECHECK THE DATA
3125  011430  005337  013414          NXREAD:         DEC SECCNTR
3126  011434  001404                                  BEQ EXIT
3127  011436  004737  010020                          JSR PC,XREAD            ;READ THE NEXT SECTOR
3128  011442  000137  010644                          JMP EMPBUFF
3129  011446  005001                  EXIT:           CLR R1                  ;CLEAR THE ONE READ FLAG
3130  011450  000207                                  RTS PC
3131
3132  011452  000000                  BYTECNTR:       0
3133  011454  000000                  BADBYTE:        0
3134  011456  000000                  GOODBYTE:       0
3135  011460  000000                  ERCNTR:         0
3136
3137                                  ;;**************************************************************
3136
3139                                          ;AN INTERRUPT DID NOT OCCURE AT A FUNCTION DONE FLAG.
3140
3141  011462  104406                  NOINTER:        CKSWR
3142  011464  032777  020000  167524                  BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
3143  011472  001006                                  BNE 1$
3144  011474  004737  010566                          JSR PC,ERMSG
3145  011500  104401  016272                          TYPE ,MINTER            ;TYPE NO INTERRUPT ON DONE ERROR
3146  011504  104401  016134                          TYPE ,MCRLF
3147  011510  104406                  1$:             CKSWR
3148  011512  005777  167500                          TST @SWR                ;TEST HALT ON ERROR SWITCH
3149  011516  100001                                  BPL CONT21
3150  011520  000000                  HLT15:          HALT                    ;HALT ON ERROR
3151  011522  004737  011526          CONT2.:         JSR PC,INTSERV          ;JSR TO INTSERV AS IF IT WAS AN INTERRUPT
3152
```

```
3153                                            .SBTTL  INTERRUPT SERVICE
3154
3155  011526  117737  167456  012146  INTSERV:  MOVB @RXDB,ASTAT        ;SAVE THE ERROR AND STATUS WORD
3156  011534  005777  167446                    TST @RXCS               ;TEST THE ERROR FLAG
3157  011540  100444                            BMI RXERROR             ;THERE WAS AN ERROR GO REPORT IT
3158  011542  032737  000004  012146            BIT #BIT2,ASTAT         ;IS INIT DONE SET
3159  011550  001402                            BEQ 2$                  ;NO,CONTINUE
3160  011552  000137  012006                    JMP RXPWR               ;YES,REPORT POWER FAILED AND RESTAR
3161  011556  032737  000003  012146  2$:       BIT #3,ASTAT            ;ARE PAR OR CRC BITS SET
3162  011564  001021                            BNE 1$                  ;YES GO REPORT ERROR
3163  011566  132777  000040  167412            BITB #DONEBIT,@RXCS     ;IS DONE SET
3164  011574  001012                            BNE 3$                  ;IF SET RETURN TO TEST
3165  011576  104406                            CKSWR
3166  011600  032777  020000  167410            BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
3167  011606  001004                            BNE 4$                  ;DON'T PRINT
3168  011610  104401  016325                    TYPE ,MUKNINT           ;TYPE UNKNOWN INTERRUPT
3169  011614  104401  016134                    TYPE ,MCRLF
3170  011620  000002          4$:               RTI                     ;RETURN FROM THE INTERRUPT
3171  011622  062706  000006          3$:       ADD #6,SP               ;BYPASS INTERRUPT POINTERS ON STACK
3172  011626  000207                            RTS PC                  ;RETURN TO PROGRAM
3173  011630  104406          1$:               CKSWR
3174  011632  032777  020000  167356            BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
3175  011640  001004                            BNE RXERROR
3176  011642  104401  016630                    TYPE ,MNOFLAG           ;TYPE NO STATUS ERROR ERROR
3177  011646  104401  016134                    TYPE ,MCRLF
3178  011652  005237  006472          RXERROR:  INC ERRORS              ;AN ERROR INDICATOR
3179  011656  001775                            BEQ RXERROR
3180  011660  052737  000400  012724            BIS #BIT8,UNITSEL       ;SET HARD ERROR FLAG
3181  011666  012777  000017  167312  2$:       MOV #RDER,@RXCS         ;GET THE ERROR CODE
3182  011674  004737  006574          3$:       JSR PC,SDN              ;TEST FOR DONE FLAG
3183  011700  000775                            BR 3$
3184  011702  032777  000002  167300            BIT #2,@RXDB            ;WAS THERE A PARITY ERROR
3185  011710  001403                            BEQ 1$                  ;NO,CONTINUE
3186  011712  004737  012022                    JSR PC,PARTYP           ;YES,GO REPORT THE PARITY ERROR
3187  011716  000763                            BR 2$                   ;REISSUE THE FUNCTION
3188  011720  117737  167264  012150  1$:       MOVB @RXDB,BSTAT        ;SAVE THE ERROR CODE IN B STATUS
3189  011726  104406          NOPRNT:           CKSWR
3190  011730  032777  020000  167260            BIT #SW13,@SWR          ;TEST PRINT ERROR SWITCH IN SWR
3191  011736  001020                            BNE 2$
3192  011740  104401  016134                    TYPE ,MCRLF
3193  011744  004737  010566                    JSR PC,ERMSG            ;TYPE ERROR AND MESSAGES
3194  011750  104401  016226                    TYPE ,MRXCS             ;TYPE COMMAND STATUS REGISTER
3195  011754  013746  007702                MOV FUNCTION,-(SP)  ;;SAVE FUNCTION FOR TYPEOUT
3196  011760  104403                        TYPOS               ;;GO TYPE--OCTAL ASCII
3197  011762     006                        .BYTE 6             ;;TYPE 6 DIGIT(S)
3198  011763     000                        .BYTE 0             ;;SUPPRESS LEADING ZEROS
3199  011764  004737  012050                    JSR PC,TYPADR           ;TYPE ADDRESSES AND RUN CONDITIONS
3200  011770  104401  016134                    TYPE ,MCRLF
3201  011774  004737  012220                    JSR PC,TYPCODE          ;PRINT THE STATUS REGISTERS
3202  012000  062706  000004          2$:       ADD #4,SP               ;MOVE ERROR RETURN TO TOP OF STACK
3203  012004  000207                            RTS PC
3204
3205  012006  104401  016701          RXPWR:    TYPE ,MRX11             ;ONLY THE RX11 POWER HAS FAILED
3206  012012  104401  015304                    TYPE ,$POWER            ;PRINT POWER FAILED
3207  012016  000137  001350                    JMP RESTART             ;GO TO RESTART
```

```
3208  012022  104401  016226          PARTYP:         TYPE ,MRXCS
3209  012026  017746  167154                  MOV     @RXCS,-(SP)     ;;SAVE @RXCS FOR TYPEOUT
3210  012032  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3211  012034     006                          .BYTE   6               ;;TYPE 6 DIGIT(S)
3212  012035     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
3213  012036  104401  016613                  TYPE ,MPAR
3214  012042  104401  016134                  TYPE ,MCRLF
3215  012046  000207                          RTS PC
3216
3217  012050  104401  016034          TYPADR:         TYPE ,MTRK              ;TYPE TRACK ADDRESS
3218  012054  013746  013126                  MOV     TARGET,-(SP)    ;;SAVE TARGET FOR TYPEOUT
3219  012060  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3220  012062     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
3221  012063     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
3222  012064  104401  016063                  TYPE ,MSECT             ;TYPE SECTOR ADDRESS
3223  012070  013737  013416  012144          MOV TSECTOR,2$
3224  012076  042737  177740  012144          BIC #177740,2$          ;CLEAR ALL BUT SECTOR ADDRESS
3225  012104  013746  012144                  MOV     2$,-(SP)        ;;SAVE 2$ FOR TYPEOUT
3226  012110  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3227  012112     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
3228  012113     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
3229  012114  104401  016131                  TYPE ,DBLSP
3230  012120  032737  000020  012724          BIT #BIT4,UNITSEL       ;WHITCH DRIVE IS BEING USED
3231  012126  001003                          BNE 1$
3232  012130  104401  016177                  TYPE ,MUNIT0            ;TYPE UNIT 0
3233  012134  000402                          BR 4$
3234  012136  104401  016207          1$:             TYPE ,MUNIT1            ;TYPE UNIT 1
3235  012142  000207                  4$:             RTS PC
3236  012144  000000                  2$:             OPEN
3237
3238  012146  000000                  ASTAT:          0
3239  012150  000000                  BSTAT:          0
3240
3241
3242  012152  117737  167032  012146  RDCODE:         MOVB @RXDB,ASTAT        ;SAVE THE A STATUS
3243  012160  012777  000017  167020  2$:             MOV #RDER,@RXCS         ;READ THE B STATUS REGISTER
3244  012166  004737  006574          3$:             JSR PC,SDN              ;WAIT FOR DONE FLAG
3245  012172  000775                          BR 3$
3246  012174  032777  000002  167006          BIT #2,@RXDB            ;WAS THERE A PARITY ERROR
3247  012202  001403                          BEQ 1$                  ;;NO,CONTINUE
3248  012204  004737  012022                  JSR PC,PARTYP           ;YES,REPORT THE PARITY ERROR
3249  012210  000763                          BR 2$                   ;RETRY READING STATUS B
3250  012212  117737  166772  012150  1$:             MOVB @RXDB,BSTAT        ;SAVE THE B STATUS CODES
3251  012220  104401  016236          TYPCODE:        TYPE ,MASTAT            ;TYPE THE CONTENTS OF THE TWO STATUS REGISTERS
3252  012224  013746  012146                  MOV     ASTAT,-(SP)     ;;SAVE ASTAT FOR TYPEOUT
3253  012230  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3254  012232     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
3255  012233     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
3256  012234  104401  016122                  TYPE ,TAB
3257  012240  104401  016252                  TYPE ,MBSTAT
3258  012244  013746  012150                  MOV BSTAT,-(SP)
3259  012250  104404                          TYPON
3260  012252  104401  016134                  TYPE ,MCRLF
3261  012256  000207                          RTS PC
```

```
3262                                    .SBTTL  PATTERN GENERATOR
3263
3264                                    ;NOTE: ALL DATA PATTERNS WILL BE MODIFIED SO THE FIRST BYTE WIL
3265                                    ;CONTAIN THE TRACK ADDRESS. THE SECOND BYTE WILL CONTAIN THE UNIT
3266                                    ;NUMBER AND SECTOR ADDRESS IN WHICH THE DATA IS WRITTEN. THE MOST
3267                                    ;SIGNIFICANT BIT OF THIS SECOND BYTE INDICATES THE UNIT. UNIT 0
3268                                    ;IF "0" UNIT 1 IF "1". THE LAST TWO BYTES CONTAIN THE CHECK SUM.
3269                                ;;**************************************************************
3270
3271
3272
3273  012260  012704  017420        GETPATTERN:     MOV #BUFADR,R4          ;SET ADDRESS OF FIRST DATA BYTE
3274  012264  005037  012536                        CLR SUM                 ;SET UP FOR ACCUMULATION OF CHECK SUM
3275  012270  013705  012324                        MOV PAT,R5              ;GET PATTERN BITS
3276  012274  006305                                ASL R5
3277  012276  000175  012302                        JMP @PATTERNS(R5)
3278  012302  012326                PATTERNS:       DATA0                   ;000 DATA BYTE
3279  012304  012340                                DATA1                   ;377 DATA BYTE
3280  012306  012350                                FLOAT0                  ;FLOAT A 0 THROUGH ALL 1'S
3281  012310  012412                                FLOAT1                  ;FLOAT A 1 THROUGH ALL 0'S
3282  012312  012420                                PAT125                  ;125/052 DATA WORD
3283  012314  012440                                PAT314                  ;314/063 DATA WORD
3284  012316  012450                                COUNT                   ;INCRUMENT DATA PATTERN
3285  012320  012470                                RANDATA                 ;RANDOM DATA BYTE
3286
3287
3288  012322  000000                DATABYTE:       0
3289  012324  000000                PAT:            0
3290
3291
3292                                ;;**************************************************************
3293
3294                                        ;LOAD SOFTWARE BUFFER WITH ALL ZEROS
3295                                        ;   PAT = 0
3296
3297  012326  005037  012322        DATA0:          CLR DATABYTE
3298  012332  004737  012510        PATGEN:         JSR PC,LOAD             ;GO LOAD THE DATA BUFFER
3299  012336  000775                                BR PATGEN
3300
3301                                ;;**************************************************************
3302
3303                                        ;LOAD SOFTWARE BUFFER WITH ALL ONES
3304                                        ;   PAT = 1
3305
3306
3307  012340  112737  000377  012322  DATA1:          MOVB #377,DATABYTE
3308  012346  000771                                BR PATGEN
3309
```

```
                                        ;FLOAT A 0 THROUGH ONES IN SOFTWARE BUFFER
                                        ;   PAT = 2
                                        ;
012350  112737  000376  012322  FLOAT0:       MOVB #376,DATABYTE      ;SET UP A ONES FIELD
012356  000261                  XPATGEN:      SEC                     ;SET THE C BIT TO ROTATE THROUGH THE DATA
012360  012702  000000          1$:           MOV #0,R2               ;CLR R2 (CAN'T USE "CLR" IT CLEARS "C" BIT.
012364  103001                                BCC 2$                  ;BR IF "C" BIT IS CLEARED
012366  005202                                INC R2                  ;SET R2 IF "C" BIT IS SET
012370  004737  012510          2$:           JSR PC,LOAD             ;GO LOAD THE DATA BUFFER
012374  000241                                CLC                     ;CLEAR THE "C" BIT
012376  005702                                TST R2                  ;IS R2 NONZERO
012400  001401                                BEQ 3$
012402  000261                                SEC                     ;YES, SET THE "C" BIT
012404  106137  012322          3$:           ROLB DATABYTE
012410  000763                                BR 1$
                                        ;
                                        ;;*****************************************************************
                                        ;
                                        ;FLOAT A 1 THROUGH ALL ZEROS IN SOFTWARE BUFFER
                                        ;   PAT = 3
                                        ;
012412  005037  012322          FLOAT1:       CLR DATABYTE
012416  000757                                BR XPATGEN
                                        ;
                                        ;;*****************************************************************
                                        ;
                                        ;LOAD SOFTWARE BUFFER WITH ALTERNATING 1 AND 0 FOR
                                        ;ONE BYTE AND THE COMPLIMENT INTO THE NEXT
                                        ;   PAT = 4
                                        ;
012420  112737  000125  012322  PAT125:       MOVB #125,DATABYTE
012426  004737  012510          XXPATGEN:     JSR PC,LOAD
012432  105137  012322                        COMB DATABYTE
012436  000773                                BR XXPATGEN
                                        ;
                                        ;;*****************************************************************
                                        ;
                                        ;LOAD SOFTWARE BUFFER WITH ALTERNATING PAIRS OF 1 AND 0 AND
                                        ;COMPLIMENT INTO THE NEXT
                                        ;   PAT = 5
                                        ;
012440  112737  000314  012322  PAT314:       MOVB #314,DATABYTE
012446  000767                                BR XXPATGEN
                                        ;
                                        ;;*****************************************************************
                                        ;
                                        ;LOAD SOFTWARE BUFFER WITH COUNT PATTERN
                                        ;   PAT = 6
                                        ;
012450  012737  000377  012322  COUNT:        MOV #377,DATABYTE
012456  005237  012322          1$:           INC DATABYTE
012462  004737  012510                        JSR PC,LOAD
012466  000773                                BR 1$
```

```
3363                                    ;;*****************************************************************
3364
3365                                            ;LOAD SOFTWARE BUFFER WITH RANDOM DATA PATTERN
3366                                            ;   PAT = 7
3367
3368  012470  004737  012540          RANDATA:        JSR PC,RANGEN           ;GET RANDOM NUMBER
3369  012474  113737  012632  012322                  MOVB RANUM,DATABYTE
3370  012502  004737  012510                          JSR PC,LOAD
3371  012506  000770                                  BR RANDATA
3372
3373  012510  063737  012322  012536  LOAD:          ADD DATABYTE,SUM        ;ACCUMULATE THE PATTERN CHECK SUM
3374  012516  113724  012322                          MOVB DATABYTE,(R4)+     ;LOAD THE DATA BUFFER
3375  012522  022704  017620                          CMP #BUFADR+200,R4      ;HAVE 128 BYTES BEEN GENERATED
3376  012526  001401                                  BEQ 1$                  ;IF YES,RETURN TO TEST
3377  012530  000207                                  RTS PC                  ;IF NO,RETURN TO PATTERN GENERATOR
3378  012532  005726                  1$:             TST (SP)+               ;TAKE PATTERN RETURN ADDRESS OF STACK
3379  012534  000207                                  RTS PC                  ;RETURN TO TEST
3380
3381  012536  000000                  SUM:            0
3382
3383  012540  012700  000001          RANGEN:         MOV #1,R0
3384  012544  063700  012626                          ADD RAN1,R0
3385  012550  063700  012630                          ADD RAN2,R0
3386  012554  042700  170000                          BIC #170000,R0
3387  012560  000241                                  CLC
3388  012562  006100                                  ROL R0
3389  012564  006100                                  ROL R0
3390  012566  010037  012626                          MOV R0,RAN1
3391  012572  005000                                  CLR R0
3392  012574  013700  012630                          MOV RAN2,R0
3393  012600  006000                                  ROR R0
3394  012602  006000                                  ROR R0
3395  012604  063700  012626                          ADD RAN1,R0
3396  012610  042700  170000                          BIC #170000,R0
3397  012614  010037  012630                          MOV R0,RAN2
3398  012620  010037  012632                          MOV R0,RANUM
3399  012624  000207                                  RTS PC
3400
3401  012626  001234                  RAN1:           001234
3402  012630  000765                  RAN2:           000765
3403  012632  000000                  RANUM:          0
3404
```

```
                                        .SBTTL  UNIT SELECTION

                                        ;TEST FOR SELECTED UNITS,DRIVE READY,AND USED CONDITIONS
                                        ;ALSO CONTAINS A "HAD ERROR" FLAG TO BE TESTED AT EOP.
                                        ;THE BITS IN UNITSEL ARE USED AS FOLLOWS
                                        ;
                                        ;BIT15  =UNIT 1 SELECTED VIA SWR
                                        ;BIT14  =UNIT 1 USED BIT
                                        ;BIT8   =THIS PASS HAD AN ERROR
                                        ;BIT7   =UNIT 0 SELECTED VIA SWR
                                        ;BIT6   =UNIT 0 USED BIT
                                        ;BIT4   =UNIT SELECTION FOR FUNCTION WORD

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

012634  032737  000100  012724  GETUNIT:      BIT #BIT6,JNITSEL       ;WAS UNIT 0 JUST USED
012642  001012                                BNE 1$                  ;UNIT 0 USED CHECK UNIT 1
012644  105737  012724                        TSTB UNITSEL            ;WAS UNIT 0 SELECTED
012650  100007                                BPL 1$                  ;NO GO TO UNIT 1
012652  042737  040020  012724                BIC #40020,JNITSEL      ;CLEAR UNIT 1 USED BIT AND FUNCTION UNIT BIT
012660  052737  000100  012724                BIS #BIT6,UNITSEL       ;SET UNIT 0 USED BIT
012666  000207                                RTS PC
012670  005737  012724          1$:           TST UNITSEL             ;WAS UNIT 1 SELECTED
012674  100012                                BPL 2$                  ;NO RETURN
012676  032737  040000  012724                BIT #BIT14,UNITSEL      ;HAS UNIT 1 BEEN USED
012704  001006                                BNE 2$                  ;YES RETURN
012706  042737  000100  012724                BIC #BIT6,UNITSEL       ;CLEAR UNIT 0 USED BIT
012714  052737  040020  012724                BIS #40020,UNITSEL      ;SET UNIT 1 USED BIT AND FUNCTION UNIT BIT
012722  000207                          2$:   RTS PC


012724  000000                        UNITSEL:    0

                                        ;TEST THAT ALL UNITS HAVE BEEN ACCESSED

012726  005737  012724          DONE:         TST UNITSEL             ;IS UNIT 1 SELECTED
012732  100006                                BPL 1$                  ;NO RETURN
012734  032737  040000  012724                BIT #BIT14,UNITSEL      ;YES HAS IT BEEN JSED
012742  001002                                BNE 1$                  ;YES RETURN
012744  062716  000002                        ADD #2,@SP              ;BYPASS NOT DONE RETURE ON STACK
012750  000207                          1$:   RTS PC
```

```
                                        .SBTTL  TRACK SEQUENCE SELECTION

                                        ;INITIALIZE TRACK SEQUENCE

                                    ;NOTE: IF WORD SEQUEN IS CLEARED THEN TRACK SEQUENCE IS FROM 0-52-53-114 ONLY
                                    ;IF BIT 15 OF SEQUEN IS "1" THEN TRACK SELECTION IS INC. BETWEEN SELECTED OD/ID LIMITS.
                                    ;IF BIT 7 IS "1" THEN TEST 25 DECREMENT SEQUENCE IS REQUIRED.


012752  105737  013136      INITTRACK:      TSTB SEQUEN             ;IS THIS TEST 26 SPECIAL SEQUENCE
012756  100442                              BMI 2$                  ;YES, DEC FROM TRACK 12 TO 0
012760  042737  100200  001200            BIC #100200,OD          ;CLEAR FIRST USED BITS
012766  005737  001200                      TST OD                  ;TEST CONTENTS OF ID,OD FOR 0
012772  001440                              BEQ 3$                  ;SEQUENCE WILL BE FROM "HOME"-52-53-114-0
012774  052737  100000  013136            BIS #BIT15,SEQUEN       ;LIMITS WERE SELECTED, INC FROM OD TO ID.
013002  113737  001200  013126            MOVB OD,TARGET          ;INIT OD AS PRESENT TRACK
013010  005037  013134                      CLR XID                 ;INIT WORKING ID AND OD LOCATIONS
013014  113737  001201  013134            MOVB ID,XID
013022  005037  013132                      CLR XOD
013026  113737  001200  013132            MOVB OD,XOD
013034  013737  013134  013124            MOV XID,TRKCNTR         ;SET UP NUMBER OF TRACK MOVEMENTS
013042  163737  013132  013124            SUB XOD,TRKCNTR
013050  005237  013124                      INC TRKCNTR
013054  052737  100200  001200  1$:       BIS #100200,OD          ;SET FIRST TIME BITS IN ID,OD
013062  000207                            RTS PC
013064  012737  000013  013124  2$:       MOV #13,TRKCNTR         ;SET TRACK COUNTER
013072  000770                            BR 1$
013074  012737  000004  013124  3$:       MOV #4,TRKCNTR          ;SET THE TRACK COUNTER
013102  000764                            BR 1$


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

013104  113737  013126  013130  GETTRACK:       MOVB TARGET,PRESTRK     ;RESET TO PRESENT TRACK
013112  005737  013136                          TST SEQUEN              ;IS THIS THE LIMITED SEQUENCE
013116  001410                                  BEQ LIMTRK              ;YES, DOING ONLY 0-52-53-114
013120  100446                                  BMI SEQ1                ;NO,SEQUENCE IS BETWEEN SELECTED LIMITS
013122  000463                                  BR SEQ2                 ;NO,THIS IS TEST 26 DEC SEQUENCE

013124  000000  TRKCNTR:        0
013126  000000  TARGET:         0
013130  000000  PRESTRK:        0
013132  000000  XOD:            0
013134  000000  XID:            0
013136  000000  SEQUEN:         0
```

```
3435                                    ;;*****************************************************************
3496
3497                                            ;LIMITED SEQUENCE, ACCESS TRACKS 52 TO 53 TO 114 BACK TO 0
3498  013140  005737  001200          LIMTRK:       TST 0D                  ;TEST HIGH ORDER FIRST TIME BIT
3499  013144  100007                               BPL 1$                  ;NOT SET, ON TRACK 52
3500  013146  012737  000052  013126               MOV #52,TARGET          ;GO TO TRACK 52
3501  013154  042737  100000  001200               BIC #BIT15,0D           ;CLEAR FIRST TIME BIT
3502  013162  000207                               RTS PC
3503  013164  105737  001200          1$:          TSTB 0D                 ;TEST LOW ORDER FIRST TIME BIT
3504  013170  100007                               BPL 2$                  ;NOT SET, ON TRACK 53
3505  013172  012737  000053  013126               MOV #53,TARGET          ;GO TO TRACK 53
3506  013200  042737  000200  001200               BIC #BIT7,0D
3507  013206  000207                               RTS PC
3508  013210  023727  013126  000114  2$:          CMP TARGET,#114         ;IS IT ON TRACK 114
3509  013216  001404                               BEQ 3$                  ;YES,GO TO TRACK 0
3510  013220  012737  000114  013126               MOV #114,TARGET         ;NO, GO TO TRACK 114
3511  013226  000207                               RTS PC
3512  013230  005037  013126          3$:          CLR TARGET              ;GO TO TRACK 0
3513  013234  000207                               RTS PC
3514
3515                                    ;;*****************************************************************
3516
3517                                            ;INCREMENT FROM 0D+1 TO ID AND RETURN TO 0D
3518                                            ; USED WHEN TRACK LIMITS ARE SELECTED
3519
3520  013236  042737  100200  001200  SEQ1:       BIC #100200,0D          ;CLEAR FIRST TIME BITS
3521  013244  123737  013134  013130               CMPB XID,PRESTRK        ;PRESENT TRACK EQUAL TO ID
3522  013252  001004                               BNE 1$                  ;NO GET NEW TRACK
3523  013254  113737  001200  013126               MOVB 0D,TARGET          ;YES RETURN TO 0D
3524  013262  000207                               RTS PC
3525  013264  005237  013126          1$:          INC TARGET              ;ADD 1 TO TARGET TRACK
3526  013270  000207                               RTS PC
3527
3528                                    ;;*****************************************************************
3529
3530                                            ;DECREMENT FROM ID = 12 TO 0D = 0
3531                                            ;USED IN TEST 26 ONLY
3532
3533  013272  005737  001200  SEQ2:       TST 0D                  ;FIRST TIME BIT SET
3534  013276  100007                               BPL 1$                  ;NO GET NEXT TRACK
3535  013300  042737  100200  001200               BIC #100200,0D          ;YES CLEAR FIRST TIME BITS
3536  013306  012737  000012  013126               MOV #12,TARGET          ;MOVE OUT 10 TRACKS
3537  013314  000207                               RTS PC
3538  013316  005337  013126          1$:          DEC TARGET              ;MOVE TO NEXT TRACK
3539  013322  000207                               RTS PC
```

```
                                        .SBTTL  SECTOR SELECTION
                                        ;SECTOR INITIALIZATION AND SELECTION
013324  005737  001202          INITSECTOR:   TST FIRST               ;TEST FIRST AND LAST FOR 0
013330  001005                                BNE 1$                  ;SECTORS SPECIFIED USE THEM
013332  005237  001202                        INC FIRST               ;NONE SPECIFIED SET FIRST TO 1
013336  112737  000032  001203                MOVB #32,LAST           ;SET LAST TO MAXIMUM
013344  113737  001203  013414  1$:           MOVB LAST,SECCNTR       ;SET UP SECTOR COUNTER
013352  163737  001202  013414                SUB FIRST,SECCNTR
013360  005237  013414                        INC SECCNTR
013364  105037  013415                        CLRB SECCNTR+1
013370  113737  001202  013416                MOVB FIRST,TSECTOR      ;PUT FIRST SECTOR IN TARGET SECTOR
013376  162737  000003  013416                SUB #3,TSECTOR          ;SUB 3 FROM TSECTOR AS FIRST TIME THROUGH
                                                                      ;IT GETS ADDED BACK ON.
013404  012737  000001  013420                MOV #1,INTLEAV          ;SET INTERLEAVE OFFSET
013412  000207                                RTS PC

013414  000000                  SECCNTR:      0
013416  000000                  TSECTOR:      0
013420  000000                  INTLEAV:      0

013422  042737  000200  013416  GETSECTOR:    BIC #200,TSECTOR        ;CLEAR THE UNIT BIT BEFORE TESTING
013430  062737  000003  013416                ADD #3,TSECTOR          ;ADD 3 FOR INTERLEAVING
013436  123737  001203  013416                CMPB LAST,TSECTOR
013444  002010                                BGE 1$                  ;NEW SECTOR IS WITHIN LIMITS
013446  113737  001202  013416                MOVB FIRST,TSECTOR      ;RESET TARGET SECTOR TO INTERLEAVE
013454  063737  013420  013416                ADD INTLEAV,TSECTOR     ;ADD ON INTERLEAVE OFFSET VALUE
013462  005237  013420                        INC INTLEAV             ;UP DATE THE OFFSET VALUE
013466  032737  000020  012724  1$:           BIT #BIT4,UNITSEL       ;IS THIS UNIT 0
013474  001403                                BEQ 2$
013476  052737  000200  013416                BIS #BIT7,TSECTOR       ;NO, SET UNIT IDENTIFIER IN TARGET SECTOR
013504  000207                  2$:           RTS PC
```

```
                                .SBTTL  TYPE ROUTINE

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

013506  105737  013735          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
013512  100002                          BPL     1$              ;;BR IF YES
013514  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
013516  000407                          BR      3$              ;;LEAVE
013520  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
013522  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
013526  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
013530  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
013532  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
013534  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
013536  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
013542  000002                          RTI                     ;;RETURN
013544  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
013550  001430                          BEQ     8$
013552  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
013556  001006                          BNE     5$
013560  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
013562  104401                          TYPE                    ;;TYPE A CR AND LF
013564  013737                          $CRLF
013566  105037  013722                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
013572  000755                          BR      2$              ;;GET NEXT CHARACTER
013574  004737  013656          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
013600  123726  013734          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
013604  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
013606  013746  013732                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
013612  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
013616  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
013620  004737  013656                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
013624  105337  013722                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
013630  000770                          BR      7$              ;;LOOP

                                ;HORIZONTAL TAB PROCESSOR

013632  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
013636  004737  013656          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
013642  132737  000007  013722          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
013650  001372                          BNE     9$              ;;TAB STOP
013652  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
```

```
3629  013654  000724                          BR      2$              ;;GET NEXT CHARACTER
3630  013656  105777  000044                $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
3631  013662  100375                          BPL     $TYPEC
3632  013664  116677  000002  000036        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
3633  013672  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
3634  013700  001003                          BNE     1$              ;;BRANCH IF NO
3635  013702  105037  013722                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
3636  013706  000406                          BR      $TYPEX          ;;EXIT
3637  013710  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
3638  013716  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
3639  013720  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
3640  013722  000000                $CHARCNT:.WORD   0               ;;CHARACTER COUNT STORAGE
3641  013724  000207                $TYPEX: RTS     PC
3642
3643  013726  177564                $TPS:   .WORD   177564          ;;TTY PRINTER STATUS REG. ADDRESS
3644  013730  177566                $TPB:   .WORD   177566          ;;TTY PRINTER BUFFER REG. ADDRESS
3645  013732     000                $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
3646  013733     002                $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
3647  013734     012                $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
3648  013735     000                $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
3649  013736     077                $QUES:  .ASCII  "?"             ;;QUESTION MARK
3650  013737     015                $CRLF:  .ASCII  <15>            ;;CARRAIGE RETURN
3651  013740  000012                $LF:    .ASCIZ  <12>            ;;LINEFEED
```

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

013742  017646  000000  $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
013746  116637  000001  014165  MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
013754  112637  014167          MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
013760  062716  000002          ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
013764  000406                  BR      $TYPON
013766  112737  000001  014165  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
013774  112737  000006  014167          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
014002  112737  000005  014164  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
014010  010346                  MOV     R3,-(SP)        ;;SAVE R3
014012  010446                  MOV     R4,-(SP)        ;;SAVE R4
014014  010546                  MOV     R5,-(SP)        ;;SAVE R5
014016  113704  014167          MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
014022  005404                  NEG     R4
014024  062704  000006          ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
014030  110437  014166          MOVB    R4,$OMODE       ;;SAVE IT FOR USE
014034  113704  014165          MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
014040  016605  000012          MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
014044  005003                  CLR     R3              ;;CLEAR THE OUTPUT WORD
014046  006105          1$:     ROL     R5              ;;ROTATE MSB INTO "C"
014050  000404                  BR      3$              ;;GO DO MSB
014052  006105          2$:     ROL     R5              ;;FORM THIS DIGIT
014054  006105                  ROL     R5
014056  006105                  ROL     R5
014060  010503                  MOV     R5,R3
014062  006103          3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
014064  105337  014166          DECB    $OMODE          ;;TYPE THIS DIGIT?
014070  100016                  BPL     7$              ;;BR IF NO
014072  042703  177770          BIC     #177770,R3      ;;GET RID OF JUNK
014076  001002                  BNE     4$              ;;TEST FOR 0
014100  005704                  TST     R4              ;;SUPPRESS THIS 0?
014102  001403                  BEQ     5$              ;;BR IF YES
```

```
 3708  014104  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
 3709  014106  052703  000060                   BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
 3710  014112  052703  000040            5$:    BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
 3711  014116  110337  014162                   MOVB    R3,8$           ;;SAVE FOR TYPING
 3712  014122  104401  014162                   TYPE    ,8$             ;;GO TYPE THIS DIGIT
 3713  014126  105337  014164            7$:    DECB    $OCNT           ;;COUNT BY 1
 3714  014132  003347                          BGT     2$              ;;BR IF MORE TO DO
 3715  014134  002402                          BLT     6$              ;;BR IF DONE
 3716  014136  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
 3717  014140  000744                          BR      2$              ;;GO DO THE LAST DIGIT
 3718  014142  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
 3719  014144  012604                          MOV     (SP)+,R4        ;;RESTORE R4
 3720  014146  012603                          MOV     (SP)+,R3        ;;RESTORE R3
 3721  014150  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
 3722  0   '56  012616                          MOV     (SP)+,(SP)
 3723  0.  .60  000002                          RTI                     ;;RETURN
 3724  014162     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
 3725  014163     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
 3726  014164     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
 3727  014165     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
 3728  014166  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
                        .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES

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

014170                  $SAVREG:
014170  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
014172  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
014174  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
014176  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
014200  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
014202  010546                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
014204  016646  000022          MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
014210  016646  000022          MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
014214  016646  000022          MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
014220  016646  000022          MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
014224  000002                  RTI

                        ;*RESTORE R0-R5
                        ;*CALL:
                        ;*      RESREG
014226                  $RESREG:
014226  012666  000022          MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
014232  012666  000022          MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
014236  012666  000022          MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
014242  012666  000022          MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
014246  012605                  MOV     (SP)+,R5        ;;POP STACK INTO R5
014250  012604                  MOV     (SP)+,R4        ;;POP STACK INTO R4
014252  012603                  MOV     (SP)+,R3        ;;POP STACK INTO R3
014254  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
014256  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
014260  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
014262  000002                  RTI
```

```
                                .SBTTL  TTY INPUT ROUTINE

                                ;;*****************************************************************
                                $TKS:   .WORD   177560          ;;TTY KBD STATUS
                                $TKB:   .WORD   177562          ;;TTY KBD BUFFER
                                .ENABL  LSB

                                ;;*****************************************************************
                                ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
                                ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
                                ;*WHEN OPERATING IN TTY FLAG MODE.
014270  022737  000176  001216  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED?
014276  001074                        BNE     15$             ;;BRANCH IF NO
014300  105777  177760                TSTB    @$TKS           ;;CHAR THERE?
014304  100071                        BPL     15$             ;;IF NO, DON'T WAIT AROUND
014306  117746  177754                MOVB    @$TKB,-(SP)     ;;SAVE THE CHAR
014312  042716  177600                BIC     #^C177,(SP)     ;;STRIP-OFF THE ASCII
014316  022726  000007                CMP     #7,(SP)+        ;;IS IT A CONTROL G?
014322  001062                        BNE     15$             ;;NO, RETURN TO USER
014324  123727  015044  000001        CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
014332  001456                        BEQ     15$             ;;BRANCH IF YES

014334  104401  015015                TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
014340  104401  015022        $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
014344  013746  000176                MOV     $WREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
014350  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS
014352  104401  015033                TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
014356  005046                19$:    CLR     -(SP)           ;;CLEAR COUNTER
014360  005046                        CLR     -(SP)           ;;THE NEW SWR
014362  105777  177676        7$:     TSTB    @$TKS           ;;CHAR THERE?
014366  100375                        BPL     7$              ;;IF NOT TRY AGAIN

014370  117746  177572                MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
014374  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII


014400  021627  000025        9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
014404  001005                        BNE     10$             ;;BRANCH IF NOT
014406  104401  015010                TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
014412  062706  000006        20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
014416  000757                        BR      19$             ;;LET'S TRY IT AGAIN


014420  021627  000015        10$:    CMP     (SP),#15        ;;IS IT A <CR>?
014424  001022                        BNE     16$             ;;BRANCH IF NO
014426  005766  000004                TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
014432  001403                        BEQ     11$             ;;BRANCH IF YES
014434  016677  000002  164554        MOV     2(SP),@SWR      ;;SAVE NEW SWR
014442  062706  000006        11$:    ADD     #6,SP           ;;CLEAR UP STACK
014446  104401  013737        14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
014452  123727  015045  000001        CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
014460  001003                        BNE     15$             ;;BRANCH IF NOT
014462  012777  000100  177574        MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
014470  000002                15$:    RTI                     ;;RETURN
```

```
3830  014472  004737  013656          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
3831  014476  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
3832  014502  002420                          BLT     18$             ;;BRANCH IF YES
3833  014504  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
3834  014510  003015                          BGT     18$             ;;BRANCH IF YES
3835  014512  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
3836  014516  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
3837  014522  001403                          BEQ     17$             ;;BRANCH IF YES
3838  014524  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
3839  014526  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
3840  014530  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
3841  014532  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
3842  014536  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
3843  014542  000707                          BR      7$              ;;GET THE NEXT ONE
3844  014544  104401  013735          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
3845  014550  000720                          BR      20$             ;;SIMULATE CONTROL-U
3846                                .DSABL  LSB
3847
3848
3849                                ;;*****************************************************************
3850                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
3851                                ;*CALL:
3852                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
3853                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
3854                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
3855                                ;
3856
3857  014552  011646                $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
3858  014554  016666  000004  000002        MOV     4(SP),2(SP)     ;;SAVE THE PS
3859  014562  105777  177476          1$:   TSTB    @$TKS           ;;WAIT FOR
3860  014566  100375                        BPL     1$              ;;A CHARACTER
3861  014570  117766  177472  000004        MOVB    @$TKB,4(SP)     ;;READ THE TTY
3862  014576  042766  177600  000004        BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
3863  014604  026627  000004  000023        CMP     4(SP),#23       ;;IS IT A CONTROL-S?
3864  014612  001013                        BNE     3$              ;;BRANCH IF NO
3865  014614  105777  177444          2$:   TSTB    @$TKS           ;;WAIT FOR A CHARACTER
3866  014620  100375                        BPL     2$              ;;LOOP UNTIL ITS THERE
3867  014622  117746  177440                MOVB    @$TKB,-(SP)     ;;GET CHARACTER
3868  014626  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
3869  014632  022627  000021                CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
3870  014636  001366                        BNE     2$              ;;IF NOT DISCARD IT
3871  014640  000750                        BR      1$              ;;YES, RESUME
3872  014642  026627  000004  000140  3$:   CMP     4(SP),#140      ;;IS IT UPPER CASE?
3873  014650  002407                        BLT     4$              ;;BRANCH IF YES
3874  014652  026627  000004  000175        CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
3875  014660  003003                        BGT     4$              ;;BRANCH IF YES
3876  014662  042766  000040  000004        BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
3877  014670  000002                4$:     RTI                     ;;GO BACK TO USER
3878                                ;;*****************************************************************
3879                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
3880                                ;*CALL:
3881                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
3882                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
3883                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
3884
3885  014672  010346                $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
```

```
3886  014674  012703  015000          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
3887  014700  022703  015010          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
3888  014704  101405                          BLOS    4$              ;;BR IF YES
3889  014706  104407                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
3890  014710  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
3891  014712  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBCUT
3892  014716  001003                          BNE     3$              ;;SKIP IF NOT
3893  014720  104401  013736          4$:     TYPE    ,$QUES          ;;TYPE A '?'
3894  014724  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
3895  014726  111337  014776          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
3896  014732  104401  014776                  TYPE    ,9$
3897  014736  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
3898  014742  001356                          BNE     2$              ;;LOOP IF NOT RETURN
3899  014744  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
3900  014750  104401  013740                  TYPE    ,$LF            ;;TYPE A LINE FEED
3901  014754  012603                          MOV     (SP)+,R3        ;;RESTORE R3
3902  014756  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
3903  014760  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
3904  014766  012766  015000  000004          MOV     #$TTYIN,4(SP)
3905  014774  000002                          RTI                     ;;RETURN
3906  014776     000          9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
3907  014777     000                  .BYTE   0               ;;TERMINATOR
3908  015000  000010          $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
3909  015010  052536  005015     000  $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
3910  015015     136  006507  000012  $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
3911  015022  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
3912  015030  020075     000
3913  015033     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
3914  015040  036440  000040
3915  015044     000          $AUTOB: .BYTE   0       ;;AUTO MODE FLAG
3916  015045     000          $INTAG: .BYTE   0       ;;INTERRUPT MODE FLAG
```

```
                                        .SBTTL  TRAP DECODER
                                
                                        ;;**********************  '**************************************
                                        ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
                                        ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
                                        ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
                                        ;*GO TO THAT ROUTINE.
                                
015046  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
015050  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
015054  005740                          TST     -(R0)           ;;BACKUP BY 2
015056  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
015060  006300                          ASL     R0              ;;POSITION FOR INDEXING
015062  016000  015102                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
015066  000200                          RTS     R0              ;;GO TO ROUTINE
                                
                                
                                        ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
                                
015070  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
015072  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
015100  000002                          RTI                     ;;RESTORE THE PSW
                                
                                        .SBTTL  TRAP TABLE
                                
                                        ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                        ;*BY THE "TRAP" INSTRUCTION.
                                
                                        ;       ROUTINE
                                        ;       -------
015102  015070                  $TRPAD: .WORD   $TRAP2
015104  013506                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
015106  013766                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
015110  013742                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
015112  014002                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
                                
015114  014340                          $GTSWR  ;;CALL=GTSWR    TRAP+5(104405)  GET SOFT-SWR SETTING
                                
015116  014270                          $CKSWR  ;;CALL=CKSWR    TRAP+6(104406)  TEST FOR CHANGE IN SOFT-SWR
015120  014552                          $RDCHR  ;;CALL=RDCHR    TRAP+7(104407)  TTY TYPEIN CHARACTER ROUTINE
015122  014672                          $RDLIN  ;;CALL=RDLIN    TRAP+10(104410) TTY TYPEIN STRING ROUTINE
015124  014170                          $SAVREG ;;CALL=SAVREG   TRAP+11(104411) SAVE R0-R5 ROUTINE
015126  014226                          $RESREG ;;CALL=RESREG   TRAP+12(104412) RESTORE R0-R5 ROUTINE
015130  006516                          XSUBSCOPE       ;;CALL=SUBSCOPE TRAP+13(104413)
```

```
                                .SBTTL  POWER DOWN AND UP ROUTINES

                                ;;*****************************************************************
                                ;POWER DOWN ROUTINE
015132  012737  015276  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
015140  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
015146  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
015150  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
015152  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
015154  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
015156  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
015160  010546                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
015162  017746  164030          MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
015166  010637  015302          MOV     SP,$SAVR6       ;;SAVE SP
015172  012737  015204  000024  MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
015200  000000                  HALT
015202  000776                  BR      .-2             ;;HANG UP

                                ;;*****************************************************************
                                ;POWER UP ROUTINE
015204  012737  015276  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
015212  013706  015302          MOV     $SAVR6,SP       ;;GET SP
015216  005037  015302          CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
015222  005237  015302  1$:     INC     $SAVR6          ;;WAIT FOR THE INC
015226  001375                  BNE     1$              ;;OF  WORD
015230  012677  163762          MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
015234  012605                  MOV     (SP)+,R5        ;;POP STACK INTO R5
015236  012604                  MOV     (SP)+,R4        ;;POP STACK INTO R4
015240  012603                  MOV     (SP)+,R3        ;;POP STACK INTO R3
015242  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
015244  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
015246  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
015250  012737  015132  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
015256  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
015264  104401                  TYPE                    ;REPORT THE POWER FAILURE
015266  015304          $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
015270  012716                  MOV     (PC)+,(SP)      ;;RESTART AT RESTART
015272  001350          $PWRAD: .WORD   RESTART         ;;RESTART ADDRESS
015274  000002                  RTI
015276  000000          $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
015300  000776                  BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
015302  000000          $SAVR6: 0                       ;;PUT THE SP HERE
015304  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
015312  000122
                                .EVEN
```

```
                                .SBTTL  SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE

                                ;;*****************************************************************
                                ;*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
                                ;*UNSIGNED DECIMAL ASCIZ NUMBER.
                                ;*CALL
                                ;*      MOV     NUMBER,-(SP)    ;;PUT BINARY NUMBER ON THE STACK
                                ;*      JSR     PC,@#$$B2D      ;;CALL
                                ;*      RETURN                  ;;ADDRESS OF THE 1ST ASCIZ CHAR.IS ON THE STACK

015314  016637  000002  015344 $$B2D:  MOV     2(SP),1$        ;;SAVE BINARY NUMBER
015322  012746  015344                 MOV     #1$,-(SP)       ;;SET POINTER
015326  004737  015350                 JSR     PC,@#$DB2D      ;;CALL DOUBLE LENGTH CONVERT
015332  062716  000005                 ADD     #5,(SP)         ;;ONLY ALLOW FIVE CHARACTERS
015336  012666  000002                 MOV     (SP)+,2(SP)     ;;PICKUP POINTER
015342  000207                         RTS     PC              ;;RETURN
015344  000000  000000         1$:     .WORD   0,0
                                .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE

                                ;;*****************************************************************
                                ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
                                ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
                                ;*POSITIVE.
                                ;*CALL
                                ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
                                ;*      JSR     PC,@#$DB2D
                                ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
                                                                ;;IS ON THE STACK


015350  104411                  $DB2D:  SAVREG                  ;;SAVE REGISTERS
015352  016602  000002                  MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
015356  012700  015530                  MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
015362  010066  000002                  MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
015366  012201                          MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
015370  012202                          MOV     (R2)+,R2
015372  012737  000012  015446          MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
015400  012704  015460                  MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
015404  012705  015462                  MOV     #$TNPWR+2,R5
015410  005003                  1$:     CLR     R3              ;;CLEAR PARTIAL
015412  161401                  2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
015414  005602                          SBC     R2
015416  161502                          SUB     (R5),R2
015420  002402                          BLT     3$              ;;BR IF TEN POWER TO LARGE
015422  005203                          INC     R3              ;;ADD 1 TO PARTIAL
015424  000772                          BR      2$              ;;LOOP
015426  062401                  3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
015430  005502                          ADC     R2
015432  062402                          ADD     (R4)+,R2
015434  022525                          CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
015436  052703  000060                  BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
015442  110320                          MOVB    R3,(R0)+        ;;SAVE IT
015444  005327                          DEC     (PC)+           ;;DONE?
015446  000000                  4$:     .WORD   0
015450  001357                          BNE     1$              ;;BR IF NO
```

```
 4062  015452  105020                    CLRB    (R0)+           ;;TERMINATOR
 4063  015454  104412                    RESREG                  ;;RESTORE REGISTERS
 4064  015456  000207                    RTS     PC              ;;RETURN
 4065  015460  145000          $TNPWR:   145000                  ;;1.0E09
 4066  015462  035632                    35632
 4067  015464  160400                    160400                  ;;1.0E08
 4068  015466  002765                    2765
 4069  015470  113200                    113200                  ;;1.0E07
 4070  015472  000230                    230
 4071  015474  041100                    041100                  ;;1.0E06
 4072  015476  000017                    17
 4073  015500  103240                    103240                  ;;1.0E05
 4074  015502  000001                    1
 4075  015504  023420                    23420                   ;;1.0E04
 4076  015506  000000                    0
 4077  015510  001750                    1750                    ;;1.0E03
 4078  015512  000000                    0
 4079  015514  000144                    144                     ;;1.0E02
 4080  015516  000000                    0
 4081  015520  000012                    12                      ;;1.0E01
 4082  015522  000000                    0
 4083  015524  000001                    1                       ;;1.0E00
 4084  015526  000000                    0
 4085  015530  000014          $DECVL:   .BLKB   12.             ;;RESERVE STORAGE FOR ASCIZ STRING
 4086
```

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

                                ;TYPE NUMERICAL ASCIZ STRING,RIGHT JUSTIFIED
                                ;REPLACING LEADING ZEROS WITH SPACES.
                                ;
                                ;FIRST ADDRESS OF ASCIZ STRING MUST BE ON TOP OF THE STACK

015544  010046          RTJUST: MOV R0,-(SP)            ;SAVE R0
015546  016600  000004          MOV 4(SP),R0            ;PICK UP ADDRESS OF ASCIZ STRING
015552  010037  015604          MOV R0,3$               ;SAVE ADDRESS FOR TYPE OUT
015556  105710          1$:     TSTB (R0)               ;IS THIS THE TERMINATOR
015560  001406                  BEQ 2$                  ;IF YES TYPE IT OUT
015562  122710  000060          CMPB #'0,(R0)           ;IS IT A ZERO
015566  001005                  BNE 4$                  ;IF NO GO PRINT IT
015570  112720  000040          MOVB #' ,(R0)+          ;IF YES REPLACE IT WITH A SPACE
015574  000770                  BR 1$                   ;TEST NEXT CHAR.
015576  112740  000060  2$:     MOVB #'0,-(R0)          ;STRING OFF ALL ZEROS,PUT BACK THE LAST ONE
015602  104401          4$:     TYPE                    ;TYPE THE STRING
015604  000000          3$:     OPEN
015606  012600                  MOV (SP)+,R0            ;RESTORE R0
015610  012616                  MOV (SP)+,(SP)          ;RESTORE THE STACK
015612  000207                  RTS PC                  ;RETURN

                                ;TYPES 16 BIT WORD IN DECIMAL

015614  012546          SGLDEC: MOV (R5)+,-(SP)         ;PUT NUMBER TO BE TYPED ON STACK
015616  004737  015314          JSR PC,@#$SB2D          ;CONVERT NUMBER TO DECIMAL
015622  004737  015544          JSR PC,RTJUST           ;TYPE THE DECIMAL NUMBER
015626  000205                  RTS R5
```

```
                                                .SBTTL  MESSAGES


015630  042524  052123  050040  MDTESTP:        .ASCIZ "TEST PARAMETERS: "
015636  051101  046501  052105
015644  051105  035123  000040

015652  005015  042412  040522  MXEHEADER:      .ASCIZ <15><12><12> "ERADR  FAST   FAPT          GOOD   BAD     PASS"
015660  051104  020040  040506
015666  052123  020040  043040
015674  050101  020124  020040
015702  020040  020040  020040
015710  043440  047517  020104
015716  020040  040502  020104
015724  020040  020040  040520
015732  051523     000
015735     125  042516  050130  MUNXDD:         .ASCIZ "UNEXPECTED D D MARK"<15><12>
015742  041505  042524  020104
015750  020104  020104  040515
015756  045522  005015     000

015763     104  042040  046440  MDDMIS:         .ASCIZ "D D MARK MISSING"<15><12>
015770  051101  020113  044515
015776  051523  047111  006507
016004  000012

016006  040504  040524  020054  MDERHDR:        .ASCIZ "DATA, NO STATUS ERROR"
016014  047516  051440  040524
016022  052524  020123  051105
016030  047522  000122

016034  047440  020116  051124  MTRK:           .ASCIZ " ON TRACK"
016042  041501  000113

016046  020040  051106  046517  MPRES:          .ASCIZ "  FROM TRACK"
016054  052040  040522  045503
016062     000

016063     040  020057  042523  MSECT:          .ASCIZ " / SECTOR"
016070  052103  051117     000

016075     015  020012  054502  MCOLMUN:        .ASCIZ <15><12>" BYTE  BAD  GOOD"<15><12>
016102  042524  020040  040502
016110  020104  043440  047517
016116  006504  000012

016122  020040  020040  020040  TAB:            .ASCIZ <40><40><40><40><40><40>
016130     000

016131     040  000040          DBLSP:          .ASCIZ <40><40>

016134  005015     000          MCRLF:          .ASCIZ <15><12>

016137     015  042412  051122  MERHEADER:      .ASCIZ <15><12>"ERROR CONDITIONS:  TEST PC = "
016144  051117  041440  047117
```

```
 4172  016152  044504  044524  047117
 4173  016160  035123  020040  042524
 4174  016166  052123  050040  020103
 4175  016174  020075     000
 4176
 4177  016177     125  044516  020124  MUNIT0:         .ASCIZ "UNIT 0 "
 4178  016204  020060     000
 4179
 4180  016207     125  044516  020124  MUNIT1:         .ASCIZ "UNIT 1 "
 4181  016214  020061     000
 4182
 4183  016217     117  046116  020131  MONLY:          .ASCIZ "ONLY  "
 4184  016224  000040
 4185
 4186  016226  054122  051503  036440  MRXCS:          .ASCIZ "RXCS = "
 4187  016234  000040
 4188
 4189  016236  052123  052101  051525  MASTAT:         .ASCIZ "STATUS A = "
 4190  016244  040440  036440  000040
 4191
 4192  016252  052123  052101  051525  MBSTAT:         .ASCIZ "STATUS B = "
 4193  016260  041040  036440  000040
 4194
 4195  016266  005015  000012          DBLLF:          .ASCIZ <15><12><12>
 4196
 4197  016272  047516  044440  052116  MINTER:         .ASCIZ "NO INTERRUPT AT DONE ERROR"
 4198  016300  051105  052522  052120
 4199  016306  040440  020124  047504
 4200  016314  042516  042440  051122
 4201  016322  051117     000
 4202
 4203  016325     125  045516  047516  MUKNINT:        .ASCIZ "UNKNOWN INTERRUPT"
 4204  016332  047127  044440  052116
 4205  016340  051105  052522  052120
 4206  016346     000
 4207
 4208  016347     124  052117  046101  MERCT:          .ASCIZ "TOTAL READ CHECK ERRORS = "
 4209  016354  051040  040505  020104
 4210  016362  044103  041505  020113
 4211  016370  051105  047522  051522
 4212  016376  036440  000040
 4213
 4214  016402  044506  046114  052502  MFIL:           .ASCIZ "FILLBUFFER "
 4215  016410  043106  051105  000040
 4216
 4217  016416  046505  052120  041131  MEMPTY:         .ASCIZ "EMPTYBUFFER "
 4218  016424  043125  042506  020122
 4219  016432     000
 4220
 4221  016433     040  051124  041501  MLIMTRK:        .ASCIZ " TRACKS 52,53,114,0  "
 4222  016440  051513  032440  026062
 4223  016446  031465  030454  032061
 4224  016454  030054  020040     000
 4225
 4226  016461     117  036504     000  MOD:            .ASCIZ "OD="
 4227
```

```
4228  016465  040  044440  036504  MID:          .ASCIZ "  ID="
4229  016472  000
4230
4231  016473  040  043040  051111  MFIRST:       .ASCIZ "  FIRST="
4232  016500  052123  000075
4233
4234  016504  020040  040514  052123  MLAST:      .ASCIZ "  LAST="
4235  016512  000075
4236
4237  016514  051103  020103  051105  MBADCRC:    .ASCIZ "CRC ERROR NO DATA ERROR"
4238  016522  047522  020122  047516
4239  016530  042040  052101  020101
4240  016536  051105  047522  000122
4241
4242  016544  042522  042101  000040  MREAD:      .ASCIZ "READ "
4243
4244  016552  040504  040524  041440  MCRC:       .ASCIZ "DATA CRC ERROR"
4245  016560  041522  042440  051122
4246  016566  051117  000
4247
4248  016571  123  042505  020113  MSEEK:        .ASCIZ "SEEK ERROR"
4249  016576  051105  047522  000122
4250
4251  016604  051127  052111  020105  MWRITE:     .ASCIZ "WRITE "
4252  016612  000
4253
4254  016613  120  051101  052111  MPAR:         .ASCIZ "PARITY ERROR"
4255  016620  020131  051105  047522
4256  016626  000122
4257
4258  016630  051105  047522  020122  MNOFLAG:    .ASCIZ "ERROR FLAG ERROR"
4259  016636  046106  043501  042440
4260  016644  051122  051117  000
4261
4262  016651  102  042101  000  MBAD:           .ASCIZ "BAD"
4263
4264  016655  040  000  SPACE:                   .ASCIZ <40>
4265
4266  016657  107  047517  000104  MGOOD:       .ASCIZ "GOOD"
4267
4268  016664  020040  044103  041505  MSUM:      .ASCIZ "  CHECK SUM "
4269  016672  020113  052523  020115
4270  016700  000
4271
4272  016701  015  051012  030530  MRX11:       .ASCIZ <15><12>"RX11 / RXV11"
4273  016706  020061  020057  054122
4274  016714  030526  000061
4275
4276  016720  005015  046412  044501  MREV:      .ASCIZ <15><12><12> "MAINDEC-11-DZRXB-E" <15><12>
4277  016726  042116  041505  030455
4278  016734  026461  055104  054122
4279  016742  026502  006505  000012
4280
4281  016750  005015  047125  054105  LOC4M:  .ASCIZ  <15><12>"UNEXPECTED TRAP TO LOC. 4 OCCURRED"
4282  016756  042520  052103  042105
4283  016764  052040  040522  020120
```

```
4284  016772  047524  046040  041517
4285  017000  020056  020064  041517
4286  017006  052503  051122  042105
4287  017014     000
4288
4289  017015     015  052412  042516  LOC10M: .ASCIZ  <15><12>"UNEXPECTED TRAP TO LOC. 10 OCCURRED"
4290  017022  050130  041505  042524
4291  017030  020104  051124  050101
4292  017036  052040  020117  047514
4293  017044  027103  030440  020060
4294  017052  041517  052503  051122
4295  017060  042105     000
4296
4297  017063     075  041520     000  PCM:    .ASCIZ  "=PC"
4298
4299  017067     015  052012  040522  OD2BIG: .ASCII  <15><12>"TRACK LIMITS SELECTED OUT OF RANGE"
4300  017074  045503  046040  046511
4301  017102  052111  020123  042523
4302  017110  042514  052103  042105
4303  017116  047440  052125  047440
4304  017124  020106  040522  043516
4305  017132     105
4306  017133     015  042012  043105          .ASCIZ  <15><12>"DEFAULTING TO "
4307  017140  052501  052114  047111
4308  017146  020107  047524  000040
4309
4310  017154  005015  042523  052103  S2BIG:  .ASCII  <15><12>"SECTOR LIMITS SELECTED OUT OF RANGE"
4311  017162  051117  046040  046511
4312  017170  052111  020123  042523
4313  017176  042514  052103  042105
4314  017204  047440  052125  047440
4315  017212  020106  040522  043516
4316  017220     105
4317  017221     015  042012  043105          .ASCIZ  <15><12>"DEFAULTING TO "
4318  017226  052501  052114  047111
4319  017234  020107  047524  000040
4320
4321  017242  005015  040503  052125  DOLOAD: .ASCII  <15><12>"CAUTION - IF YOU DESIRE TO TEST UNIT 0"
4322  017250  047511  020116  020055
4323  017256  043111  054440  052517
4324  017264  042040  051505  051111
4325  017272  020105  047524  052040
4326  017300  051505  020124  047125
4327  017306  052111  030040
4328  017312  005015  042522  046120          .ASCII  <15><12>"REPLACE LOAD MEDIUM WITH A SCRATCH DISKETTE"
4329  017320  041501  020105  047514
4330  017326  042101  046440  042105
4331  017334  052511  020115  044527
4332  017342  044124  040440  051440
4333  017350  051103  052101  044103
4334  017356  042040  051511  042513
4335  017364  052124     105
4336  017367     015  052012  042510          .ASCIZ  <15><12>"THEN PRESS CONTINUE"<15><12>
4337  017374  020116  051120  051505
4338  017402  020123  047503  052116
4339  017410  047111  042525  005015
```

F10


```
 4340  017416     000
 4341
 4342           017420                          .EVEN
 4343
 4344                                           ;;**************************************************************************
 4345
 4346                                           ;THE FOLLOWING LOCATIONS ARE USED FOR DATA STORAGE,RETRY COUNTERS
 4347                                           ;ACCESS COUNTERS ETC.
 4348
 4349  017420  000200                           BUFADR:         .BLKB 200
 4350
 4351           000001                          .END
```

```
ADJSUM  010766          2318    2391    2409    2747    3026    3044#
ASTAT   012146          2804    2913    2920    2969    2987    3155*   3158    3161    3238#   3242*   3252
BADBYT  011454          3037*   3038    3039    3085    3133#
BIT0  = 000001           176#   2920
BIT00 = 000001           166#    176
BIT01 = 000002           165#    175
BIT02 = 000004           164#    174
BIT03 = 000010           163#    173
BIT04 = 000020           162#    172
BIT05 = 000040           161#    171
BIT06 = 000100           160#    170
BIT07 = 000200           159#    169
BIT08 = 000400           158#    168
BIT09 = 001000           157#    167
BIT1  = 000002           175#   2804    2913
BIT10 = 002000           156#
BIT11 = 004000           155#
BIT12 = 010000           154#
BIT13 = 020000           153#   2869
BIT14 = 040000           152#    446    1428    3429    3443
BIT15 = 100000           151#    451     467    3018    3463    3501
BIT2  = 000004           174#   3158
BIT3  = 000010           173#
BIT4  = 000020           172#   3230    3569
BIT5  = 000040           171#   1536    2633
BIT6  = 000100           170#    667     827    1079    1240    1284    1298    1339    1356    1398    1417    1485    1589
                        1828    1847    1953    2969    2987    3420    3425    3431
BIT7  = 000200           169#    670     671    1379    1488    1489    1593    2444    2450    3025    3080    3506    3571
BIT8  = 000400           168#    545     829     863    2525    2557    2976    2989    3064    3180
BIT9  = 001000           167#
BPTVEC= 000014           183#
BRLEV   001214           367#   2488
BSTAT   012150          3188*   3239#   3250*   3258
BUFADR  017420          1621    1724    3044*   3045*   3049*   3052*   3053    3273    3375    4349#
BUSERR  006130           211    2456#
BYTECN  011452          2750*   2758*   3023*   3041*   3081    3098*   3132#
CATERR  003032          1167    1172*   1175#   1187*
CCOUNT  002526           435*    832     835*    836     838*    874#   2524*
CEXIT   004246          1062    1300    1357    1418    1604    1614    1721    1807    1808#
CKBYTE  010736          3037#
CKSUM   011056          3027*   3038*   3046*   3047*   3048*   3049    3050*   3051*   3052    3056#   3114
CKSWR = 104406           843     888    2521    2562    2568    2618    2667    2767    2773    2822    2827    2832    2960
                        2868    2872    2935    2940    2951    2956    2971    2979    2990    3066    3094    3106    3120
                        3141    3147    3165    3173    3189    3955#
CCMMWO  007634          2329    2789    2847#   2892
CONT10  010522          2968    2987#
CONT11  010470          2973    2979#   2992
CONT13  007312          2775    2777#
CONT14  007562          2829    2831#
CONT15  010312          2942    2944#
CONT16  010364          2958    2960#
CONT17  010502          2981    2983#
CONT20  011276          3096    3098#
CONT21  011522          3149    3151#
CONT4   007300          2769    2773#
CCUNT   012450          3284    3359#
```

```
CPUPRI  006200            1331    1390    2488#
CR    = 000015              91#   3633    3643
CRCER   010234            2921    2929#
CRLF  = 000200              92#   3604    3643
DATABY  012322            3288#   3297*   3307*   3313*   3323*   3331*   3340*   3342*   3351*   3359*   3360*   3369*   3373
                          3374
DATACK  007704            2858#   2887*   2931*   3101
DATACR  010330            2930    2934    2951#
DATAER  011110            3040    3064#
DATAO   012326            3278    3297#
DATAI   012340            3279    3307#
DBLLF   016266            4195#
DBLSP   016131            3084    3088    3229    4166#
DDCHK   010410            2904    2967#
DDERR   010452            2976#   2935
DDISP = 177570              98#    375
DEATH   003034            1171    1181#
DEXIT   004260            1811#   1893    2027    2081    2133    2268    2305    2328    2396    2412    2425    245:
DING    006450            2558    2568#   3009
DISPLA  001220             375#    416*
DISPRE  000174             234#    416
DONE    012726            2740    3441#
DONEBI= 000040             198#   2793    2896    3163
DRVSWP  007012            2424    2714#
DSWR  = 177570              97#    374
DTESTP  001212             365#    442     446     449     470     864
DULOAD  017242             461    4321#
EBIE  = 000103             200#   3033
EMPBUF  010644            2932    3020#   3060    3124    3128
EMPDON  011306            3028    3101#
EMPER   011060            3029    3058#
EMPFLA  010730            3034#   3035    3042    3099
EMTVEC= 000030             186#
EPCSCO  006474            2514#   2515#   2578#   2610
ERCNTR  011460            2933    3024#   3065#   3071    3104    3110    3135#
ERMSG   010566            2974    2993    2999#   3075    3144    3193
ERRORS  006472             546#   2516#   2577#   2583    2592#   2598    3178#
ERRVEC= 000004             179#
EXIT    011446            3022    3126    3129#
FAST    002532             548#    755#    876#   2537
FBEB    004064            1622    1631#   1637    1725    2321
FBIE  = 000101             199#   2754
FILLBU  007144            2747#   2764    2779    2813    2836
FILLDO  007332            2748    2782#   2809
FILLER  007222            2749    2761#
FILLFL  007202            2755#   2756    2759
FIRST   001202             298#    299     506     508#    511     514     528#    3544    3546#   3549    3552    3566
FIRSTT  002170             754#    869
FLOATO  012350            3280    3313#
FLOATI  012412            3281    3331#
FUNCTI  007702            2326#   2410#   2785#   2788#   2847#   2848    2857#   2891#   3195
GETPAT  012260            1620    2317    2390    2408    2714    2732    3273#
GETSEC  013422            2746    2886    3562#
GETTRA  013104            2703    2717    2735    3482#
GETUNI  012634            2315    2407    2702    2716    2720    2734    3420#
GMS   = ******            3948    3949    3950    3951    3953    3955    3956    3957    3958    3959    3960
```

```
GOODBY  011456            3089*   3091    3134#
GTSWR = 104405             422    3953#
HANGER  006756             437*    553*    828*    852*   2635*   2665*   2690*   2694#   2782*   2795*   2888*   2898*
HANGPL  006760             438*    554*    663*   2636*   2666*   2691*   2695#
HERE    002462             855     861#
HLT10   010310            2943#
HLT12   010362            2959#
HLT13   010500            2982#
HLT14   011274            3097#
HLT15   011520            3150#
HLT6    007310            2776#
HLT7    007560            2830#
HOME    007706            2831    2860#   2877
HT    = 000011              89#   3602    3643
HUNGUP  006622            2637    2644#
ID    = 001201             297#    483     485     493    3466
ILLADR  005106            1910    2043    2096    2137#   2142
INCERR  006244            2516#   2517
INITSE  013324            2745    2885    3544#
INITTR  012752            2701    2715    2733    3458#
INTLEA  013420            3555*   3560#   3567    3568*
INTSER  011526            2302    2422    2447    3151    3155#
IOTVEC= 000020             184#
KRXVEC  001204             301#    1015    1069    1233    1274    1316    1372    2301    2421    2446
LAST  = 001203             299#    509     511     519    3547*   3548    3564
LF    = 000012              90#   3637    3643
LIMITS  001622             475#    502
LIMTRK  013140            3484    3498#
LOAD    012510            3298    3318    3341    3361    3370    3373#
LOCKUP  002556             752     888#
LOC10M  017015            2472    4289#
LOC4M   016750            2456    4281#
LOGICA  002452             227     857#
MABELL  006470             940    2571    2574#
MASTAT  016236            3251    4189#
MBAD    016651            3116    4262#
MBADCR  016514            2938    4237#
MBSTAT  016252            3257    4192#
MCOLMU  016075            3079    4158#
MCRC    016552            2954    4244#
MCRLF   016134             474     523     834     846    2529    2772    2844    2939    2955    2978    3008    3077    3093
                          3119    3146    3169    3177    3192    3200    3214    3260    4168#
MDDMIS  015763            2975    4137#
MDERHD  016006            3076    4143#
MDTEST  015630             469    4119#
MEMPTY  016416            3059    4217#
MERCT   016347            3109    4208#
MERHEA  016137            2999    4170#
MFIL    016402            2762    4214#
MFIRST  016473             513    4231#
MGOOD   016657            3118    4266#
MHUNGP  006674            2671    2681#
MID     016465             492    4228#
MILTST  002536             871     880#
MINTER  016272            3145    4197#
MLAST   016504             518    4234#
```

```
MLIMTR  016433            479    4221#
MNOFLA  016630           3176    4258#
MOD     016461            487    4226#
MONLY   016217            459    4183#
MORETE  002162            752#   1809    1812    2670    2679
MPAR    016613           2771    3213    4254#
MPASS   006725            847    2686#   3004
MPRES   016046           2839    4151#
MREAD   016544           2916    2924    4242#
MREV    016720            408    4276#
MRXCS   016226           3194    3208    4186#
MRX11   016701           3205    4272#
MSECT   016063           3222    4155#
MSEEK   016571           2838    4248#
MSUM    016664           3113    4268#
MTRK    016034           3217    4148#
MUKNIN  016325           3168    4203#
MUNITO  016177           3232    4177#
MUNIT1  016207            458    3234    4180#
MUNXDD  015735           2994    4132#
MWRITE  016604           2807    2818    2825    4251#
MX      002534            831*    839     878#   1808*   1811*
MXEHEA  015652           2527    4123#
NEXTRD  010132           2780    2903    2906    2908#   2918    2926    2946    2963    2986
NOERTY  011264           3068    3072    3094#
NOINTE  011462           2797    2900    3141#
NOMORE  002274            764     827#
NOPRIN  006420           2523    2557#
NOPRNT  011726           3189#
NOSCOP  006504           2592#   2620
NXREAD  011430           3061    3105    3122    3125#
OD      001200            296#    297     468*    475     485     488     501*   3460*   3461    3464    3468    3472*   3498
                         3501*   3503    3506*   3520*   3523    3533   3535*
OD2BIG  017067            498    4299#
OK2GO   003036           1174    1186#
OPEN  = 000000            205#    850    2554    2770    3007    3083    3112    3236    4105
PARTES  007252           2765    2767#   2810    2919    3062
PARTYP  012022           3186    3208#   3248
PASS    002530            436*    841*    848     875#   2552    3005
PAT     012324           1612*   1799*   1804*   2316*   2387*   2389*   2404*   2406*   2420*   2433*   2445*   3275    3289#
PATGEN  012332           3298#   3299    3308
PATTER  012302           3277    3278#
PAT125  012420           3282    3340#
PAT314  012440           3283    3351#
PCM     017063           2463    2480    4297#
PCONT   007326           2763*   2778    2780#   2808*   2918*   3061*
PCSCOP  006556            548     754*    755    2540    2593*   2621    2623#   2786    2902    2967    3000
PIRQ  = 177772             96#
PIRQVE= 000240            190#
PLOOP   007322           2764*   2779#   2809*   2917*   3060*
PRESTR  013130           2840    2879*   3482*   3490#   3521
PRETES  002034            524     545#
PR0   = 000000            113#
PR1   = 000040            114#
PR2   = 000100            115#
PR3   = 000140            116#
```

```
PR4    = 000200            117#   1325
PR5    = 000240            118#   1384
PR6    = 000300            119#
PR7    = 000340            120#    212    214    218    220    222    224    428   1019   1027   1073   1122   1237
                          1260   1278   1287   1320   1341   1376   1401   2303   2423   2448
PS     = 177776             93#     94
PSW    = 177776             94#   1325*  1384*
PTYP1    007264           2762*  2770*  2807*  2916*  3059*
PWRVEC= 000024             185#  3965*  3966*  3975*  3981*  3993*  3994*
RANDAT   012470           3285   3368*  3371
RANGEN   012540           3368   3383*
RANUM    012632           3369   3398*  3403*
RAN1     012626            433*  3384   3390*  3395   3401*
RAN2     012630            434*  3385   3392   3397*  3402*
RDCHR  = 104407           3889   3956*
RDCODE   012152           2871   3242*
RDDONE   010110           2889   2902*
RDER   = 000017            197*  3181   3243
RDERR    010142           2890   2912*
RDIE   = 000107            202*  2891
RDLIN  = 104410           3957*
RDONLY   006762           2304   2701*
READ     010014           2704   2885*  3019
READCH   010634           2719   2722   2737   3018*
REBEGI   002466            673    751    861    863*  2464   2481
RECAL  = 040001            204*  2392   2863
REREAD   010024           2887*  2917   2923   2947   2964   2985   3123
RESERR   006154            213   2472*
RESREG= 104412            3959*  4063
RESTAR   001350            401    427*  3207   3998
RESVEC= 000010             180*
RETURN   010564           2970   2988   2996*
RTJUST   015544           4094*  4114
RTN      010012           2862   2880*
RTSPC    006572           2627   2629*
RXCS     001206            303#   431    550    827*  914*  916   1079*  1193   1195   1240*  1245   1247   1284*
                          1298*  1338*  1356*  1398*  1417*  1428*  1432   1535*  1536   1538   1548   1619*  1638
                          1723*  1822*  1841*  1907*  1909*  1913   1965*  1968   1971   2040*  2042*  2046   2070*
                          2093*  2095*  2099   2122*  2162   2172*  2178   2187   2194   2245*  2248   2251   2320*
                          2332   2335*  2392*  2626   2633   2754*  2755   2793   2848*  2863*  2866   2896   3033*
                          3034   3156   3163   3181*  3209   3243*
RXDB     001210            345#   431*   432*   665    970   1483   1587   1625*  1640   1731   1826   1845   1951
                          1972   1982   2060   2074   2112   2126   2144*  2152*  2179   2181*  2188   2190*  2234
                          2252   2260   2323*  2334   2338   2757*  2851*  2854*  3037   3155   3184   3198   3242
                          3246   3250
RXERRO   011652           3157   3175   3178#  3179
RXPWR    012006           3160   3205#
SAVREG= 104411            3958*  4037
SA200    001232            241    405*   872
SA202    001222            242    399#
SCOPIN   006536           2584   2611   2618#
SDN      006574           1429   1545   1636   1823   1842   1966   2071   2123   2141   2149   2158   2176   2185
                          2191   2246   2330   2336   2393   2633#  2864   3182   3244
SECCNT   013414           2798*  2908*  3020   3125*  3548*  3549*  3550*  3551*  3558#
SECLMT   001726            479    497    506#
SEEKER   007526           2819   2822#  2925
```

```
SEKRTY  007602          2813*  2834   2836*  2923*
SEKTYP  007606          2826   2838*
SEQUEN  013136           477*  2444*  2450*  3458   3463*  3483   3493*
SEQ1    013236          3485   3520*
SEQ2    013272          3486   3533*
SGLDEC  015614           849   2553   3006   3082   3111   4112*
SPACE   016655          2536   2539   2542   2545   2548   2551   4264*
STACK = 001200            84*   294    400    407    756
STKLMT= 177774            95*
STR     006560          1631   2137   2145   2153   2173   2182   2626*  2849   2852
STYP1   007540          2818*  2825*  2924*
SUBSCO= 104413           954   1012   1186   1225   1252   1270   1467   1529   1544   1582   1669   1835   1945
                        1960   1977   2053   2066   2106   2118   2227   2241   2257   2372   3960*
SUM     012536          3046   3274*  3373*  3381*
SWHLT1  007550          2824   2827*
SWR     001216           374*   406*   411    415*   420    844    889    895   2522   2563   2569   2619   2668
                        2768   2774   2777   2823   2828   2833   2861   2869   2873   2876   2936   2941   2944
                        2952   2957   2960   2972   2980   2983   2991   3067   3069   3095   3107   3121   3142
                        3148   3166   3174   3190   3786   3823*  3973   3986*
SWREG   000176           235*   415    420   3786   3799
SW0   = 000001           148*  2569
SW00  = 000001           138*   148
SW01  = 000002           137*   147
SW02  = 000004           136*   146
SW03  = 000010           135*   145
SW04  = 000020           134*   144
SW05  = 000040           133*   143
SW06  = 000100           132*   142
SW07  = 000200           131*   141
SW08  = 000400           130*   140
SW09  = 001000           129*   139
SW1   = 000002           147*
SW10  = 002000           128*   889
SW11  = 004000           127*  2619   2777   2833   2876   2944   2960   2983   3121
SW12  = 010000           126*   895
SW13  = 020000           125*  2522   2668   2768   2823   2936   2952   2972   2991   3067   3107   3142   3166
                        3174   3190
SW14  = 040000           124*   844
SW15  = 100000           123*  2563
SW2   = 000004           146*
SW3   = 000010           145*
SW4   = 000020           144*
SW5   = 000040           143*
SW6   = 000100           142*
SW7   = 000200           141*
SW8   = 000400           140*  2861
SW9   = 001000           139*  3069
S2BIG   017154           525   4310*
TAB     016122          3256   4163*
TARGET  013126          2313*  2854   3044   3047   3218   3464*  3482   3489*  3500*  3505*  3508   3510*  3512*
                        3523*  3525*  3536*  3538*
TBITVE= 000014           181*
TESTS   002214           754    757    758*
TKVEC = 000060           188*
TPVEC = 000064           189*
TRAPVE= 000034           187*
```

```
TRKCNT  013124  2705*  2723*  2738*  3469*  3470*  3471*  3474*  3476*  3488#
TRKLMT  001642   476    483#
TRTVEC= 000014   182#
TSECTO  013416  2314*  2851   3045   3048   3223   3552*  3553*  3559#  3562*  3563*  3564   3566*  3567*
                3571*
TYPADR  012050  2977   3078   3199   3217#
TYPCOD  012220  3201   3251#
TYPE  = 104401   408    458    459    461    469    474    478    487    492    498    513    518    523
                 525    834    839    840    846    847    871   2456   2463   2472   2480   2527   2529
                2536   2539   2542   2545   2548   2551   2571   2671   2770   2771   2772   2825   2838
                2839   2844   2938   2939   2954   2955   2975   2978   2994   2999   3004   3008   3076
                3077   3079   3084   3088   3093   3109   3113   3116   3118   3119   3145   3146   3168
                3169   3176   3177   3192   3194   3200   3205   3206   3208   3213   3214   3217   3222
                3229   3232   3234   3251   3256   3257   3260   3607   3712   3797   3798   3801   3814
                3825   3844   3893   3896   3900   3948#  3995   4104
TYPERR  011210  3074   3081#
TYPOC = 104402  3800   3949#
TYPON = 104404  2538   2541   2544   2547   2550   3092   3259   3951#
TYPOS = 104403   471    489    494    515    520   2460   2477   2533   2676   2841   3001   3086   3136
                3210   3219   3226   3253   3950#
TONOTH  002166   753#   758
T1      002620   758    914#
T10     004140   761   1712#
T11     004226   761   1804#
T12     004216   761   1799#
T13     004272   761   1820#
T14     004416   761   1903#
T15     004610   761   2036#
T16     004750   761   2089#
T17     005216   761   2170#
T2      002740   758   1069#
T20     005462   764   2278#  2902
T21     005506   764   2313#
T21X    005610  2327   2329#  2411
T22     005672   764   2385#
T23     005750   764   2404#
T24     006016   764   2420#
T24X    006024  2421#  2434
T25     006050   764   2433#  2786   2967
T26     006062   764   2444#
T3      003262   758   1313#
T4      003420   758   1369#
T5      003564   758   1428#
T6      004010   758   1612#
T6FILL  004026  1613   1618#  1805   2388   2405
T7      004142   758   1720#
T7EMPT  004152  1720   1722#  1806   2375   2395
UNITSE  012724   441*   444*   448*   451*   457*   467*   545*   668    829    863*   899*   1486   1591
                1820   1839   1905   2038   2091   2385   2525   2557*  2847   2976*  2989*  3064*  3180*
                3230   3420   3422   3424*  3425*  3427   3429   3431*  3432*  3437#  3441   3443   3569
WRITE   007134  2718   2721   2736   2745#
WRTDON  007430  2783   2798#  2808   2820
WRTER   007444  2784   2803#
WRTIE = 000105   201#  2785
WRTSEK  007504  2805   2813#
WTDCIE= 000115   203#  2788
```

```
WTRDCK  007066            2449    2732#
XERROR  006232             221    2514#
XFRBYT  007210            2757#
XHOME   010004            2867    2879#
XID     013134            3465*   3466*   3469    3492*   3521
XOD     013132            3467*   3468*   3470    3491#
XPATGE  012356            3314#   3332
XREAD   010020            2886#   2909    3127
XSA202  001574             440     454     460     466     468#
XSCOPE  006476             217    2583#
XSDN    006736            2634    2690#
XSUBSC  006516            2598#   3960
XWRITE  007140            2746#   2763    2801
XWTRDC  007072            2733#   2742
XXPATG  012426            3341#   3343    3352
$AUTOB  015044             424*   3794    3915#
$CHARC  013722            3609*   3619*   3626    3635*   3640#
$CKSWR  014270            3786#   3955
$CNTLG  015015            3797    3910#
$CNTLU  015010            3814    3909#
$CRLF   013737            3608    3650#   3825    3909
$DB2D   015350            4019    4037#
$DECVL  015530            4039    4085#
$FILLC  013734            3612    3647#
$FILLS  013733            3646#
$GTSWR  014340            3798#   3953
$HD   = 000003              18      19
$ILLUP  015276            3965    3981    4000#
$INTAG  015045            3826    3916#
$LF     013740            3651#   3900    3909
$MAIL = ****** U           420    3596
$MNEW   015033            3801    3913#
$MSWR   015022            3798    3911#
$NULL   013732            3614    3645#
$OCNT   014164            3684*   3713*   3726#
$OMODE  014166            3679*   3683*   3688    3691*   3702*   3728#
$POWER  015304            3206    3996    4003#
$PWRAD  015272            3998#
$PWRDN  015132             219    3965#   3993
$PWRMG  015266            3996#
$PWRUP  015204            3975    3981#
$QUES   013736            3649#   3844    3893    3909
$RDCHR  014552            3857#   3956
$RDDEC= ****** U          3958
$RDLIN  014672            3885#   3957
$RDOCT= ****** U          3958
$RDSZ = 000010            3878#
$RESRE  014226            3762#   3959
$R2A  = ****** U          3960
$SAVRE  014170            3746#   3958
$SAVR6  015302            3974*   3982    3983*   3984*   4002#
$SB2D   015314            4017#   4113
$SETUP= 000114             237#    417    3781    3915
$STUP = 177777             237#
$SWR  = 160000              18      19#   3999
$TKB    014266            3778#   3790    3807    3861    3867
```

```
$TKS     014264            3777*  3788   3804   3828*  3859   3865
$TN    = 000001              18*
$TNPWR   015460            4044   4045   4065*
$TPB     013730            3632*  3644*
$TPFLG   013735            3590   3648*
$TPS     013726            3630   3643*
$TRAP    015046             223   3925*
$TRAP2   015070            3936*  3947
$TRP   = 000014            3940*  3949*  3950*  3951*  3952*  3953   3954*  3955   3956*  3957*  3958*  3959*  3960*
                           3961*
$TRPAD   015102            3930   3947*
$TTYIN   015000            3886   3887   3904   3908*
$TYPBN=  ****** U          3952
$TYPDS=  ****** U          3952
$TYPE    013506            3590*  3940   3948
$TYPEC   013656            3611   3618   3625   3630*  3631   3830
$TYPEX   013724            3636   3638   3641*
$TYPOC   013766            3682*  3949
$TYPON   014002            3681   3684*  3951
$TYPOS   013742            3677*  3950
$OFILL   014165            3678*  3682*  3692   3727*
.      = 017620             207*   209    210*   216*   226*   230*   233*   238*   294*   884*   1800   2688*  3643
                           3644   3645   3646   3647   3648   3649   3650   3651   3652   3777   3778   3908*  3909
                           3915   3916   3917   3977   4001   4085*  4342*  4349*
```

```
COMMEN     1#     5#   191#   561    682    924    978   1038   1082   1130   1201   1440   1499   1556   1646
        1680   1744   1855   1921   1991   2204   2280   2344   2645
ENDCOM     1#     5#   191#   646    747    950   1008   1057   1117   1160   1221   1463   1525   1578   1665
        1699   1788   1888   1941   2022   2223   2297   2368   2660
ERROR     85#   559    679    922    976   1036   1128   1199   1251   1269   1293   1348   1411   1438   1497
        1543   1554   1602   1644   1678   1742   1834   1853   1919   1959   1976   1989   2052   2065   2079
        2105   2117   2131   2202   2240   2256   2266   2342   2644
ESCAPE     1#   191#
GETPRI     1#   191#
GETSWR     1#     5#   191#   417
MULT       1#   191#
M7727    245#   615    728   1444   1769   1859   1995   2348
M7846    243#   560    681    923    977   1037   1081   1129   1200   1498   1555   1645   1679   1743   1920
        2203
NEWTST     1#   191#
POP        1#   191#  3767   3986   3987
PUSH       1#   191#  3747   3967   3973
REPORT     1#   191#
SCOPE     86#   547   1061   1297   1355   1416   1603   1703   1792   1892   2026   2080   2132   2267
SETPRI     1#     5#   191#  1334   1341   1394   1401
SETTRA  3940#  3949   3950   3951   3953   3955   3956   3957   3958   3959   3960
SETUP      1#   191#
SKIP       1#   191#
SLASH      1#   191#
SPACE    191#
STARS      1#     5#   191#   248    249    250    261    262    263    268    269    270    289    290    291
         382    393   2506   2507   2511   2512   2709   2727   3137   3269   3292   3301   3326   3334   3345
        3354   3363   3418   3480   3495   3515   3528   3575   3654   3731   3776   3781   3849   3878   3919
        3963   3979   4008   4026   4087   4344
SWRSU      1#   191#
TRMTRP  3940#
TYPBIN     1#   191#
TYPDEC     1#   191#
TYPNAM     1#   191#
TYPNUM     1#   191#
TYPOCS     1#     5#   191#   470   2458   2475   2840   3000   3195   3209   3218   3225   3252
TYPOCT     1#   191#  3799
TYPTXT     1#   191#
$$ESCA     1#   191#
$$NEWT     1#   191#
$$SET   3940#  3949   3950   3951   3953   3955   3956   3957   3958   3959   3960
$$SKIP     1#   191#
.EQUAT     1#     5#    81
.HEADE     1#     5#     8
.KT11      1#
.SETUP     1#     5#   237
.SWRHI     1#
.$ACTI     1#
.$APTB     1#
.$APTH     1#
.$APTY     1#
.$ASTA     1#
.$CATC     1#
.$CMTA     1#
.$DB2C     1#     5#  4024
.$DB2C     1#
```

```
MAINDEC-11-DZRXB-E      MACY11 27(1006)  20-OCT-77  10:12  PAGE 111
DZRXBE.P11    27-FEB-76 00:00          CROSS REFERENCE TABLE -- MACRO NAMES                    SEQ 0133

.$DIV       1#
.$EOP       1#
.$ERRO      1#
.$ERRT      1#
.$MULT      1#
.$POWE      1#      5#   3961
.$RAND      1#
.$RDDE      1#
.$RDOC      1#
.$READ      1#      5#   3774
.$R2AZ      1#
.$SAVE      1#      5#   3729
.$SB2D      1#      5#   4006
 $SB2O      1#
.$SCOP      1#
.$SIZE      1#
.$SUPR      1#
.$TRAP      1#      5#   3917
.$TYPB      1#
.$TY-       1#
.$TYPE      1#      5#   3573
.$TYPO      1#      5#   3652
.$40CA      1#
.1170       1#


. ABS.  017620     000

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 ,DZRXBE.SRC/SOL/CRF=SYSMAC.SML,DZRXBE.P11
 RUN-TIME: 13 15 1 SECONDS
 RUN-TIME RATIO: 656/30=21.5
 CORE USED:  33K  (65 PAGES)
```