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 DIAGNOSTIC 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	OD:	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.

2.3.2 NON-STANDARD DISKETTE ADDRESS SELECTION

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 ID	6----0 OD
SECTORS LOCATION 1202	BITS	12----8 LAST	4----0 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.

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

2.3.4.1 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 "DIESTP" , 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.

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

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 Ø ARE NOT WRITTEN TO A 1.

THE PROGRAM WRITES THE RXCS = Ø

NO INTERRUPTS SHOULD OCCUR

THE RXCS SHOULD REMAIN UNCHANGED = 40 (DONE)

THE RXDB SHOULD = Ø

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 Ø)

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 RXØ1 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 RXØ1 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 RXØ1 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

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 TO 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 TO 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 RX01 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.
```

D. 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
                                        .MCALL .HEADER,.EQUAT,.SETUP,.STYPE,.STYPOCT,.STRAP,TYPOCS
                                        .MCALL SETPRI,.SPOWER,STARS,.SSB2D,.SDB2D,.SAVE,COMMENT
                                        .MCALL  ENDCOMMENT,.SREAD,GETSWR


                                .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-C0),MAR 21, 1976.
                                ;*
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
                                ,EQUIV  R6,SP           ;;STACK POINTER
                                ,EQUIV  R7,PC           ;;PROGRAM COUNTER

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

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



```
000004                          SW02=   4
000002                          SW01=   2
000001                          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
                                        RDER    =17             ; READ B CODE
                                        DONEBIT =40
                                        FBIE    =101            ; IE+FILL BUFFER
                                        EBIE    =103            ; IE+EMPTY BUFFER
                                        WRTIE   =105            ; IE+WRITE SECTOR
                                        RDIE    =107            ; IE+READ SECTOR
                                        WTDDIE  =115            ; IE+WRITE DD SECTOR
                                        RECAL   =40001
                                        OPEN    =0

                                        .=0
000000  000000  000000                          .WORD 0,0

                                        .=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

                                        .=20
000020  006476                          XSCOPE
000022  000340                          PR7
000024  015116                          $PWRDN
000026  000340                          PR7
000030  006232                          XERROR
000032  000340                          PR7
000034  015046                          $TRAP                   ;ADDRESS OF TRAP SERVICE
000036  000340                          PR7

                                        .=46
000046  002452                          LOGICAL                 ;ACT 11 EOP HOOKS


                                        .=52
000052  000000                          .WORD 0

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


                                        .=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)

                        001200          .=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  - EPROR
                        ;               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  104400  016704                  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  104404                          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 READY
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  104400  016173                  TYPE ,MUNIT1
001550  104400  016203                  TYPE ,MONLY
001554  000407                          BR XSA202                       ;AND DO NOT HALT
001556  104400  017226          5$:     TYPE,   D0LOAD                  ;INFORM USER TO REMOVE LOAD MEDIUM
                                                                        ;FROM UNIT 0 AND REPLACE WITH
                                                                        ;A 'SCRATCH' DISKETTE IF HE
                                                                        ;WISHES TO TEST UNIT 0
001562  000000                          HALT                            ;WAIT FOR USER RESPONSE
001564  000403                          BR XSA202
001566  052737  100000  012724  3$:     BIS #BIT15,UNITSEL              ;SET SELECTED BIT
001574  042737  100200  001200  XSA202: BIC     #100200,OD             ;CLEAR 1ST TIME BITS FOR BOTH DRIVES
001602  104400  015614                  TYPE, MDTESTP
001606  013746  001212                  MOV     DTESTP,-(SP)    ;;SAVE DTESTP FOR TYPEOUT
001612  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
001614     006                          .BYTE   6               ;;TYPE 6 DIGIT(S)
001615     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
001616  104400  016120                  TYPE ,MCRLF
001622  005737  001200          LIMITS: TST OD
001626  001005                          BNE TRKLMT
001630  005037  013136                  CLR SEQUEN
001634  104400  016417                  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  104400  016445                  TYPE ,MOD
001666  113746  001200                  MOVB OD,-(SP)
001672  104402                          TYPOS
001674     003                          .BYTE  3
001675     000                          .BYTE  0
001676  104400  016451                  TYPE ,MID
001702  113746  001201                  MOVB ID,-(SP)
001706  104402                          TYPOS
001710     003                          .BYTE  3
001711     000                          .BYTE  0
001712  000405                          BR SECLMT
001714  104400  017053          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
```

```
   510  001732  001003                          BNE 1$
   511  001734  112737  000001  001202          MOVB #1,FIRST
   512  001742  123727  001203  000032  1$:     CMPB LAST,#32
   513  001750  101023                          BHI 2$
   514  001752  123737  001202  001203          CMPB FIRST,LAST
   515  001760  101017                          BHI 2$
   516  001762  104400  016457          3$:     TYPE ,MFIRST
   517  001766  113746  001202                  MOVB FIRST,-(SP)
   518  001772  104402                          TYPOS
   519  001774     003                          ,BYTE  3
   520  001775     000                          ,BYTE  0
   521  001776  104400  016470                  TYPE ,MLAST
   522  002002  113746  001203                  MOVB LAST,-(SP)
   523  002006  104402                          TYPOS
   524  002010     003                          ,BYTE  3
   525  002011     000                          ,BYTE  0
   526  002012  104400  016120                  TYPE ,MCRLF
   527  002016  000406                          BR PRETEST
   528  002020  104400  017140          2$:     TYPE,   S2BIG                   ;TYPE MSG. INDICATING THAT
   529                                                                          ;SECTOR RANGE SELECTED WAS
   530                                                                          ;INVALID AND DEFAULTING TO A
   531  002024  012737  015001  001202          MOV #15001,FIRST                ; 1ST SECTOR VALUE OF 1 AND
   532  002032  000753                          BR 3$                           ;A LAST SECTOR VALUE OF 32
```



```
   533                                          ,SBTTL PRETEST - INITIALIZE (KEY) PART I
   534                                  ;
   535                                  ;       "KEY" INITIALIZE SHOULD HAVE (SET) THE DONE FLAG BECAUSE
   536                                  ;       ANY (INIT) OF THE RX01 MICROCONTROLLER IS AN IMPLIED (READ SECTOR)
   537                                  ;       OF TRACK 1 SECTOR 1 (FOR SYSTEMS PROGRAMMING BOOTSTRAP APPLICATIONS).
   538                                  ;
   539                                  ;       THEREFORE, ANY ERROR (EXCEPT PARITY) THAT MAY OCCUR FROM A NORMAL
   540                                  ;       "READ SECTOR" COMMAND MAY OCCUR HERE CAUSING THE ERROR FLAG TO SET, AND
   541                                  ;       DISPLAYING THE ERROR STATUS WITHIN THE TRANSFER REGISTER AT "DONE".
   542                                  ;
   543                                  ;       THE TRANSFER REQUEST FLAG SHOULD BE CLEARED.
   544                                  ;
   545                                  ;NOTE:  SCOPE LOOPING IS NOT OFFERED BECAUSE THE "INIT" FUNCTION
   546                                  ;       WHICH PRODUCED THE ERROR HAS NOT YET BEEN VERIFIED.
   547
   548  002034  042737  000400  012724  PRETEST: BIC #BIT8,UNITSEL
   549  002042  005037  006472                  CLR ERRORS
   550  002046  000004                          SCOPE                           ; REFRESHES FAST FOR ERROR TYPEOUT
   551  002050  013737  006556  002532          MOV PCSCOPE,FAST                ; FOR ERROR TYPEOUT INFORMATION
   552  002056  012700  000040                  MOV #40, R0                     ; PROGRAM EXPECTS DONE FLAG
   553  002062  017701  177120          3$:     MOV @ RXCS, R1                  ; ACTUAL CONTENTS OF RXCS
   554  002066  020100                          CMP R1, R0
   555  002070  001407                          BEQ 1$                          ; OK
   556  002072  062737  000001  006756          ADD #1,HANGER                   ;WAIT FOR THE INIT. FUNCTION TO
   557  002100  005537  006760                  ADC HANGPL                      ;FINISH BEFORE CALLING IT AN ERROR
   558  002104  001366                          BNE 3$
   559
   560                                  ; (R0) = 40 ; (R1) = ACTUAL RXCS ; (R2) = N/A
   561
   562  002106  104000                          ERROR                           ; RXCS NOT = 40 (DONE)
   563

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

                ;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 00              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
        ;
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

653
654
655                                     ;////////////////////////////////////////////////////////////////////////////
656                                     ;  STATUS A [RXDB] AT "DONE"
657                                     ;       7       6       -       -       3       2       1       0               /
658                                     ;
659                                     ;       SEL                                     WRITE   INIT    PAR     CRC             /
660                                     ;       DRIVE   DD                              PROTECT [DONE]                  /
661                                     ;       RDY                                                     (N/A)           /
662                                     ;
663                                     ;////////////////////////////////////////////////////////////////////////////
664
665
666  002110  012737  177740  006760  1$:     MOV #177740,HANGPL                      ;RESET HANG COUNTER
667  002116  012700  000204                  MOV #204,R0                             ;EXPECT INIT DONE AND UNIT 0 RDY
668  002122  017702  177062                  MOV @RXDB,R2
669  002126  010201                          MOV R2,R1
670  002130  042701  000100                  BIC #BIT6,R1                            ;CLEAR DELETED DATA BIT
671  002134  105737  012724                  TSTB UNITSEL                            ;WAS UNIT 0 SELECTED
672  002140  100404                          BMI 2$
673  002142  042701  000200                  BIC #BIT7,R1                            ;UNIT 0 WAS NOT SELECTED
674  002146  042700  000200                  BIC #BIT7,R0                            ;CLEAR THE DRIVE 0 RDY BITS
675  002152  020100                  2$:     CMP R1, R0
676  002154  001544                          BEQ REBEGIN
677
678                                     ; (R0) = 4 IF UNIT 0 IS NOT SELECTED, OR 204 IF UNIT 0 IS SELECTED
679                                     ; (R1) = ACTUAL RXDB MINUS DELETED DATA BIT#6
680                                     ; (R2) = ACTUAL RXDB
681
682  002156  104000                          ERROR                                   ; RXDB NOT = 4, OR 204
683
684

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

                ;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 RX01                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
        ;
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

754  002160  000542                        BR      REBEGIN
755  002162  004737  002556        MORETESTS: JSR PC, LOCKUP
756  002166  005723                  TONOTHERE: TST (P3)+                    ;ADJUST R3 FOR NEXT TEST ADDRESS
```



```
757  002170  016337  002214  006556  FIRSTTEST: MOV TESTS(R3), PCSCOPE       ; EQUIVALENT TO " SCOPE "
758  002176  013737  006556  002532            MOV PCSCOPE,FAST              ;SAVE THE FIRST ADDRESS OF THE TEST
759  002204  012706  001200                    MOV #STACK,SP
760  002210  000173  002214                    JMP @TESTS(R3)
761  002214  002166  002620  002740  TESTS:    TONOTHERE, T1, T2, T3, T4, T5, T6, T7
762  002222  003262  003420  003564
763  002230  004010  004142
764  002234  004140  004226  004216            T10, T11, T12, T13, T14, T15, T16, T17
765  002242  004272  004416  004610
766  002250  004750  005216
767  002254  005462  005506  005672            T20, T21, T22, T23, T24, T25, T26, NOMORETESTS
768  002262  005750  006016  006050
769  002270  006062  002274
770
771                                     ; TEST 1 - RXCS TEST PART I / INTERRUPT TEST PART I
772
773                                     ; TEST 2 - INTERRUPT TEST PART II / VECTOR ADDRESS VERIFICATION
774
775                                     ; * TEST 3 - INTERRUPT TEST PART III / PRIORITY LEVEL VERIFICATION PART I
776
777                                     ; * TEST 4 - INTERRUPT TEST PART IV / PRIORITY VERIFICATION PART II
778
779                                     ; TEST 5 - INIT [PROGRAMMED] / RST
780
781                                     ; TEST 6 - FILL BUFFER TRANSFER LENGTH VERIFICATION
782
783                                     ; TEST 7 - EMPTY BUFFER TRANSFER LENGTH AND CONTENT VERIFICATION PART I
784
785                                     ; TEST 10 - EMPTY BUFFER TRANSFER LENGTH AND CONTENT VERIFICATION PART II
786
787                                     ; TEST 11 - FILL/EMPTY BUFFER ALL 0'S
788
789                                     ; TEST 12 - FILL/EMPTY BUFFER ALL 1'S
790
791                                     ; TEST 13 - DRIVE READY VERIFICATION FOR SELECTED DRIVES
792
793                                     ; TEST 14 - ERROR FLAG AND B - CODE VERIFICATION PART I
794
795                                     ; TEST 15 - ERROR FLAG AND B - CODE VERIFICATION PART II
796                                     ;       /DELETED DATA BIT SETS
797
798                                     ;TEST 16 - ERROR FLAG AND B - CODE VERIFICATION PART III
799                                     ;       /DELETED DATA BIT CLEARS
800
801                                     ;TEST 17 - ILLEGAL TRACK ERROR AND B - CODE VERIFICATION
802
803                                     ;TEST 20 - SEEK VERIFICATION VIA READ FUNCTION
804
805                                     ;TEST 21 - WRITE TEST
806
807                                     ;TEST 22 - INITIALIZE IMPLIED READ
808
809                                     ;TEST 23 - READ TEST
810
811                                     ;TEST 24 - DATA TRANSFER & VERIFICATION
812
```

```
                                ;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  104400  016120          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  104400  002534  4$:     TYPE, MX
002362  104400  006470          TYPE ,MABELL
002366  005237  002530  2$:     INC PASS
002372  102775                  BVS 2$
002374  104405                  CKSWR
002376  032777  040000  176612  BIT #SW14,@SWR                  ; AC SW 14 = 1 TO HALT AT END OF PASS
002404  001413                  BEQ 6$
002406  104400  016120          TYPE ,MCRLF
002412  104400  006725          TYPE ,MPASS
002416  013737  002530  002430  MOV PASS,5$
002424  004537  015600          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  104400  002536  1$:     TYPE ,MILTST
002522  000137  001232          JMP SA200
```

```
  877  002526  000000                  CCOUNT: 0
  878  002530  000000                  PASS:   0
  879  002532  000000                  FAST:   0
  880
  881  002534  000103                  MX:     .ASCIZ "C"
  882
  883  002536  046111  042514  040507  MILTST: .ASCIZ "ILLEGAL TEST"<15><12>
  884  002544  020114  042524  052123
  885  002552  005015      000
  886
  887          002556                          .EVEN
  888
```



```
  889                                  ; DATA SW 10 = 1 TO HALT AT END OF TEST
  890
  891  002556  104405                  LOCKUP: CKSWR
  892  002560  032777  002000  176430          BIT #SW10,@SWR
  893  002566  001401                          BEQ 1$
  894  002570  000000                          HALT
  895
  896                                  ; DATA SW 12 = 1 TO LOCK SCOPE LOOP ON TEST OK OR NOT
  897
  898  002572  032777  010000  176416  1$:     BIT #SW12,@SWR                  ;1S LOOP ON TEST SWITCH SET
  899  002600  001403                          BEQ 2$                          ;IF NOT SET GO ON TO NEXT TEST
  900  002602  062716  000002                  ADD #2,@SP                      ;IF SET RETURN TO FIRSTTEST
  901  002606  000207                          RTS PC
  902  002610  042737  040100  012724  2$:     BIC #40100,UNITSEL              ;CLEAR UNIT USED BITS
  903  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
;D00(1) H                STUCK HIGH                      E22
;RX RUN                  STUCK LOW                       E18
;BUS D00                 STUCK LOW                       E7
;
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

002642  104412                  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
;
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

002660  104412                  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
        ;
        ;/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*
        ;/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*
        ;/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*

1064  002732  000004                5$:     SCOPE
1065  002734  000137  004246                JMP CEXIT                   ;END OF TEST 1
```



```
                                        .SBTTL 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) = 1 DOES INDEED WRITE IT TO A 1, THEREFORE BECAUSE DONE = 1
                                ; AN INTERRUPT SHOULD OCCUR (THE PDP PRIORITY IS 0 )
                                
002740  013702  001204          T2:     MOV KRXVEC, R2
002744  010246                          MOV R2,-(SP)                ;SAVE INTERRUPT VECTOR FOR
                                                                    ;ERROR REPORT
002746  012722  003016                  MOV #1$, (R2)+              ; RX01 VECTOR ADDRESS
002752  012722  000340                  MOV #PR7, (R2)+
002756  012602                          MOV (SP)+,R2                ;RESTORE INTERRUPT VECTOR FOR
                                                                    ;ERROR REPORT
002760  005046                          CLR -(SP)                   ; PDP PRIORITY <ON>
002762  012746  002770                  MOV #6$,-(SP)
002766  000002                          RTI
002770  052777  000100  176210  6$:     BIS #BIT6, @ RXCS           ; SET RX01 INTERRUPT ENABLE
002776  000240                          NOP
                                
        ;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;
        ;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;
        ;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;/*\;
                
                ;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
        ;
        ;/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*
        ;/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*
```

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

  1124  003000  000240                      NOP
  1125  003002  012746  000340              MOV #PR7,-(SP)              ; RESET PDP PRIORITY <7> OFF
  1126  003006  012746  003014              MOV #7$,-(SP)
  1127  003012  000002                      RTI
  1128
  1129                                      ; (R0) = N/A ; (R1) = N/A ; (R2) = N/A
  1130
  1131  003014  104000              7$:     ERROR                       ; NO RX01 INTERRUPT OCCURRED
  1132

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

                ;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                       E38
        ;
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

  1167
  1168                                      ; RETURN TO HERE WITH THE PDP PRIORITY = 7 IF AN RX01 INTERRUPT
  1169
  1170  003016  005737  003032      1$:     TST CATERR                  ;DID MORE THAN ONE INTERRUPT
  1171                                                                  ;OCCUR DUE TO BOTH 'DONE' & 'IE'
  1172                                                                  ;BITS SET BUT 'RQST INTR' FLOP
  1173                                                                  ;NEVER GETTING CLEARED?
  1174  003022  001004                      BNE     DEATH              ;BRANCH IF YES
  1175  003024  105237  003032              INCB    CATERR             ;OTHERWISE, SET FLAG TO INDICATE
  1176                                                                  ;ONE INTERRUPT OCCURRED & GO ON
  1177  003030  000402                      BR      OK2GO
```



```
  1178  003032  000000              CATERR: .WORD   0                   ;THIS IS THE INTERRUPT INDICATOR.
  1179                                                                  ;1ST TIME THRU THIS TEST THE IN-
  1180                                                                  ;DICATOR SHOULD ALWAYS BE ZERO;
  1181                                                                  ;IF NON-ZERO A MULTIPLE OF
  1182                                                                  ;INTERRUPTS HAS OCCURRED DUE TO
  1183                                                                  ;THE AFOREMENTIONED CONDITION
  1184  003034  000000              DEATH:  HALT                        ;NO SENSE GOING ON - YOU ARE
  1185                                                                  ;LOCKED IN THE INTERRUPT STATE!
  1186                                                                  ;PRESS THE 'CONT' SWITH (IF
  1187                                                                  ;PRESENT) TO GO ON  AT YOUR OWN
  1188                                                                  ;RISK
  1189  003036  104412              OK2GO:  SUBSCOPE                    ;EVERYTHING APPEARS TO BE OK
  1190  003040  005037  003032              CLR     CATERR             ;CLEAR INTERRUPT INDICATOR FOR
  1191                                                                  ;NEXT POSSIBLE PASS THRU THIS TEST
  1192
  1193                                      ; THE RXCS SHOULD = 140 (INTERRUPT ENABLE+DONE)
  1194
  1195  003044  012700  000140              MOV #140, R0
  1196  003050  020077  176132              CMP R0, @ RXCS
  1197  003054  001403                      BEQ 2$
  1198  003056  017701  176124              MOV @ RXCS, R1
  1199
  1200                                      ; (R0) = 140 ; (R1) = ACTUAL RXCS ; (R2) = N/A
  1201
  1202  003062  104000                      ERROR                       ; THE RXCS NOT = 140
  1203

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

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

  1228  003064  104412              2$:     SUBSCOPE
  1229
  1230
  1231                                      ; TEST 2 - CONT'D
  1232
  1233                                      ; THE PURPOSE OF THIS TEST IS TO VERIFY THAT BIT 6 OF THE RXCS (INTERRUPT ENABLE)
```

```
1234                                    ; CAN BE CLEARED AFTER IT WAS KNOWN TO BE SET
1235
1236  003066  013702  001204                  MOV KRXVEC, R2
1237  003072  010246                         MOV     R2,-(SP)                ;SAVE INTERRUPT VECTOR FOR
1238                                                                          ;ERROR REPORT
1239  003074  012722  003164                  MOV #4$, (R2)+                  ; RX11 VECTOR ADDRESS
1240  003100  012722  000340                  MOV #PR7, (R2)+
1241  003104  012602                         MOV     (SP)+,R2                ;RESTORE INTERRUPT VECTOR FOR
1242                                                                          ;ERROR REPORT
1243  003106  042777  000100  176072         BIC #BIT6, @ RXCS               ; CLEAR THE RX11 INTERRUPT ENABLE BIT
1244
1245                                    ; THE RXCS SHOULD = 40 (DONE)
1246
1247  003114  012700  000040                  MOV #40, R0
1248  003120  020077  176062                  CMP R0, @ RXCS
1249  003124  001403                         BEQ 3$
1250  003126  017701  176054                  MOV @ RXCS, R1
1251
1252                                    ; (R0) = 40 ; (P1) = ACTUAL RXCS ; (R2) = N/A
1253
1254  003132  104000                          ERROR                           ; RXCS NOT = 40
1255  003134  104412                  3$:    SUBSCOPE
1256                                    ; NO RX11 INTERRUPTS SHOULD OCCUR (YET)
1257
1258  003136  005046                          CLR -(SP)                       ; PDP PRIORITY <ON>
1259  003140  012746  003146                  MOV #10$,-(SP)
1260  003144  000002                         RTI
1261  003146  000240                  10$:   NOP
1262  003150  000240                         NOP
1263  003152  012746  000340                  MOV #PR7,-(SP)                  ; PDP PRIORLITY <OFF> 7
1264  003156  012746  003166                  MOV #11$,-(SP)
1265  003162  000002                         RTI
1266
1267                                    ; RETURN TO HERE WITH THE PDP PRIORITY = 7 IF AN UNEXPECTED RX11 INTERRUPT
1268                                    ; WHILE CLEARING THE RX11 INTERRUPT ENABLE BIT 6
1269
1270                                    ; (R0) = N/A ; (R1) = N/A ; (R2) = N/A
1271
1272  003164  104000                  4$:    ERROR                           ; UNEXPECTED RX11 INTERRUPT
1273  003166  104412                  11$:   SUBSCOPE
1274
1275                                    ; AN RX11 INTERRUPT SHOULD OCCUR (NOW)
1276
1277  003170  013702  001204                  MOV KRXVEC, R2
1278  003174  010246                         MOV     R2,-(SP)                ;SAVE INTERRUPT VECTOR FOR
1279                                                                          ;ERROR REPORT
1280  003176  012722  003246                  MOV #5$, (R2)+                  ; RX11 VECTOR ADDRESS
1281  003202  012722  000340                  MOV #PR7, (R2)+
1282  003206  012602                         MOV     (SP)+,R2                ;RESTORE INTERRUPT VECTOR FOR
1283                                                                          ;ERROR REPORT
1284  003210  005046                          CLR -(SP)                       ; PDP PRIORITY <ON>
1285  003212  012746  003220                  MOV #12$,-(SP)
1286  003216  000002                         RTI
1287  003220  052777  000100  175760  12$:   BIS #BIT6, @ RXCS               ; SET RX11 INTERRUPT ENABLE BIT
1288  003226  000240                         NOP
1289  003230  000240                         NOP
```



```
1290  003232  012746  000340                  MOV #PR7,-(SP)
1291  003236  012746  003244                  MOV #13$,-(SP)
1292  003242  000002                         RTI
1293
1294                                    ; (R0) = N/A ; (R1) = N/A ; (R2) = N/A
1295
1296  003244  104000                  13$:   ERROR                           ; NO RX11 INTERRUPT OCCURRED
1297
1298                                    ; RETURN TO HERE WITH THE PDP PRIORITY = 7 IF AN RX01 INTERRUPT
1299
1300  003246  000004                  5$:    SCOPE
1301  003250  042777  000100  175730         BIC #BIT6, @ RXCS               ; CLEAR THE RX11 INTERRUPT ENABLE
1302
1303  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
```



```
003414  000137  004246  4$:     JMP CEXIT                       ;END OF TEST 3

                                        .SBTTL TEST 4 - INTERRUPT TEST PART IV / PRIORITY VERIFICATION PART II

                                ; THE PURPOSE OF THIS TEST IS TO VERIFY THE PRIORITY OF THE RX11 INTERRUPT REQUEST LINE
                                ; THE PROGRAM SETS THE PDP PRIORITY = THE DEVICE LEVEL, (NORMALLY 5 OR THE CONTENTS OF L
                                ;NO RX01 INTERRUPTS SHOULD OCCUR
                                ; IF AN INTERRUPT DOES OCCUR THEN THE PRIORITY LEVEL OF THE RX11 IS (NOT)
                                ; = THE NORMAL DEVICE LEVEL OF 5, OR WHATEVER IS THE VALUE IN LOCATION 'BRLEV:'
                                ; BUT MAYBE SOME VALUE GREATER THAN THE CONTENTS OF LOC. 'BRLEV:'
                                ; NOTE: IF THERE IS NO HARDWARE "PSW" THIS TEST WILL BE SKIPPED.

003420  005001          T4:     CLR     R1                      ;INDICATOR TO CPU PRIORITY ROUTINE
                                                                ;TO DROP CPU PRIORITY 1 LEVEL
                                                                ;LESS THAN THE DEVICE LEVEL
003422  013702  001204          MOV KRXVEC, R2
003426  010246                  MOV     R2,-(SP)                ;SAVE INTERRUPT VECTOR FOR
                                                                ;ERROR REPORT
003430  012722  003546          MOV #1$, (R2)+                  ; RX01 VECTOR ADDRESS
003434  012722  000340          MOV #PR7, (R2)+
003440  012602                  MOV     (SP)+,R2                ;RESTORE INTERRUPT VECTOR FOR
                                                                ;ERROR REPORT
003442  052701  000200          BIS     #BIT7,R1                ;SET INDICATOR TO CPU PRIORITY
                                                                ;ROUTINE TO SET CPU PRIORITY LEVEL
                                                                ;TO THE SAME LEVEL AS THE DEVICE
003446  013746  000004          MOV     4,-(SP)                 ;SAVE 'BUSERR' TIMEOUT 'PC'
003452  012737  003470  000004  MOV     #3$,4                   ;SET TIMEOUT VECTOR
003460  012737  000240  177776  MOV     #PR5,PSW                ;SET LEVEL TO 5 IF 'PSW' EXISTS
003466  000404                  BR      4$                      ;GO ON TO RESET VECTOR & DO TEST
003470  022626          3$:     CMP     (SP)+,(SP)+             ;CORRECT STACK FROM BUS TIMEOUT
003472  012637  000004          MOV     (SP)+,4                 ;RESTORE TIMEOUT VECTOR TO 'BUSERR'
003476  000430                  BR      5$                      ;NO HARDWARE PSW - SKIP THIS TEST
003500  012637  000004  4$:     MOV     (SP)+,4                 ;RESET TIMEOUT VECTOR TO BUSERR
003504  004737  006200          JSR     PC,CPUPRI               ;CALCULATE CPU PRIORITY LEVEL TO
                                                                ;BE THE SAME AS THE DEVICE LEVEL
                                                                ;I,E, - SAME AS CONTENTS OF LOC.
                                                                ;'BRLEV'
003510  010046                  MOV     R0,-(SP)        ;;PUT NEW PS ON STACK
003512  012746  003520          MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
003516  000002                  RTI                     ;;POP NEW PC AND PS
003520                  64$:
003520  052777  000100  175460  BIS     #BIT6,@RXCS             ;SET RX01 INTERRUPT ENABLE
003526  000240                  NOP
003530  000240                  NOP
003532  013746  000340          MOV     PR7,-(SP)       ;;PUT NEW PS ON STACK
003536  012746  003544          MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
003542  000002                  RTI                     ;;POP NEW PC AND PS
003544                  65$:
003544  000401                  BR 2$

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

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

003546  104000          1$:     ERROR                           ;PRIORITY LEVEL NOT = TO CONTENTS
                                                                ;OF LOCATION 'BRLEV:' (NORMALLY 5)
```

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



```
                                                .SBTTL TEST 5 - INIT [PROGRAMMED] / RST

                                        ; THE PURPOSE OF THIS TEST IS TO VERIFY THAT SETTING THE RXCS BIT 14
                                        ; CAUSES AN RX01 PROGRAMMED SUBSYSTEM INITIALIZE

                                        ; THE RXCS SHOULD = 40 (DONE)

                                        ; THE RXDB SHOULD = 4, OR 104, OR 204, OR 304

003564  052777  040000  175414  T5:     BIS #BIT14, @ RXCS              ; RX01 PROGRAMMED INITIALIZE
003572  004737  006574          1$:     JSR PC, SDN                     ; WAIT FOR THE DONE BIT
003576  000775                          BR 1$
003600  012700  000040                  MOV #40, R0                     ; PROGRAM EXPECTS RXCS = 40 (DONE)
003604  017701  175376                  MOV @ RXCS, R1                  ; ACTUAL RXCS
003610  020100                          CMP R1, R0
003612  001401                          BEQ 2$

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

003614  104000                          ERROR                           ; RXCS NOT = 40

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

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

003616  104412                  2$:     SUBSCOPE

                                        ;/////////////////////////////////////////////////////////////////////////////
                                        ;
                                        ;       7       6       -       -       3       2       1       0       /
                                        ;
                                        ;       SEL                             WRITE   INIT    PAR             /
                                        ;       DRIVE   DD                      PROTECT [DONE]          CRC     /
```

```
                                ;       RDY                                             (N/A)                   /
                                ;
                                ;/////////////////////////////////////////////////////////////////////////

                                ; THE RXDB SHOULD = 4, 104, IF TESTING UNIT 1
                                ; OR 204, OR 304 IF TESTING UNIT 0 (SEL. DRIVE RDY. BIT SET)

003620  012700  000204                  MOV #204,R0
003624  017702  175360                  MOV @RXDB,R2
003630  010201                          MOV R2,R1
003632  042701  000100                  BIC #BIT6,R1                    ;CLEAR DELETED DATA BIT
003636  105737  012724                  TSTB UNITSEL                    ;WAS UNIT 0 SELECTED
003642  100404                          BMI 3$
003644  042701  000200                  BIC #BIT7,R1                    ;UNIT 0 WAS NOT SELECTED
003650  042700  000200                  BIC #BIT7,R0                    ;CLEAR UNIT 0 RDY BITS
003654  020100                  3$:     CMP R1, R0
003656  001401                          BEQ 4$

                                ; (R0) = 4, OR 204
                                ; (R1) = ACTUAL RXDB MINUS DELETED DATA BIT#6
                                ; (R2) = ACTUAL RXDB

003660  104000                          ERROR                           ; RXDB NOT = 4, OR 104, OR 204, OR 304


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

        ;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 D05                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
;
;/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*
;/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*
;/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*

003662  104412                  4$:     SUBSCOPE
```



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

                                ; THE PURPOSE OF THIS TEST IS TO VERIFY THE RST COMMAND #12+GO
                                ; AND THAT THE DONE BIT IS CLEARED AFTER THE FUNCTION IS ISSUED.

003664  012777  000013  175314          MOV #13, @ RXCS                 ; RST COMMAND
003672  032777  000040  175306          BIT #BIT5, @ RXCS               ; TEST DONE BIT (SHOULD = 0)
003700  001404                          BEQ 5$
003702  017701  175300                  MOV @ RXCS, R1
003706  005000                          CLR R0

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

003710  104000                          ERROR                           ; DONE BIT NOT = 0
003712  104412                  5$:     SUBSCOPE
003714  004737  006574          9$:     JSR PC, SDN                     ; WAIT FOR DONE FLAG
003720  000775                          BR 9$
003722  012700  000040                  MOV #40, R0                     ; PROGRAM EXPECTS RXCS = 40 (DONE)
003726  017701  175254                  MOV @ RXCS, R1                  ; ACTUAL RXCS
003732  020100                          CMP R1, R0
003734  001401                          BEQ 10$

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

003736  104000                          ERROR                           ; RXCS NOT = 40


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

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

003740  104412                  10$:    SUBSCOPE

                                ; THE RXDB SHOULD = 200 (IF DRIVE 0 IS READY), OR 0 IF UNIT 0 IS NOT SELECTED
                                ; MAYBE 300 (IF DRIVE 0 IS READY AND SECTOR 1 WAS WRITTEN WITH DELETED DATA)
```

```

003742  017702  175242                  MOV @ RXDB, R2                  ; ACTUAL RXDB
003746  010201                          MOV R2, R1
003750  042701  000100                  BIC #BIT6, R1                   ; CLEAR N/A DELETED DATA BIT
003754  012700  000200                  MOV #200,R0                     ;EXPECT UNIT 0 RDY SET
003760  105737  012724                  TSTB UNITSEL
003764  100403                          BMI 11$
003766  042701  000200                  BIC #BIT7,R1                    ; UNIT 0 NOT SELECTED
003772  005000                          CLR R0                          ; DISREGUARD RDY BITS
003774  020100                  11$:    CMP R1, R0
003776  001401                          BEQ 12$

                                ; (R0) = 0 OR 200
                                ; (R1) = ACTUAL RXDB MINUS DELETED DATA BIT#6
                                ; (R2) = ACTUAL RXDB

004000  104000                          ERROR                           ; RXDB NOT = 200, OR NOT = 0
004002  000004                  12$:    SCOPE
004004  000137  004246                  JMP CEXIT                       ;END OF TEST 5
```



```
                                        .SBTTL TEST 6 - FILL BUFFER TRANSFER LENGTH VERIFICATION

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

                                ; 128 BYTE TRANSFERS SHOULD OCCUR

004010  012737  000006  012324  T6:     MOV #6, PAT                     ; COUNT PATTERN
004016  004737  004026                  JSR PC, T6FILL
004022  000137  004246                  JMP CEXIT                       ;END OF TEST 6

                                ; EXECUTE THE FOLLOWING " MOV #1, @ RXCS " INSTEAD OF " INC @ RXCS " FOR LOOPS

004026  005002                  T6FILL: CLR R2
004030  012777  000001  175150          MOV #1, @ RXCS                  ;FILL BUFFER FUNCTION [0 + GO BIT]
004036  004737  012260                  JSR PC, GETPATTERN
004042  012704  017404                  MOV #BUFADR, R4
004046  004737  004064          1$:     JSR PC, FBEB                    ; SUBROUTINE TO FILL/EMPTY BUFFER
004052  000207                          RTS PC                          ; EXIT SUBROUTINE T6FILL

004054  112477  175130                  MOVB (R4)+, @ RXDB              ; FILL THE SECTOR BUFFER
004060  005202                          INC R2                          ;INC R2 FOR BYTE COUNT
004062  000771                          BR 1$

                                ; SUBROUTINE TO FILL AND EMPTY THE SECTOR BUFFER

004064  004737  006560          FBEB:   JSR PC, STR                     ; TEST FOR TRANSFER REQUEST
004070  000403                          BR 1$
004072  062716  000002                  ADD #2, @ SP                    ; ADJUST FOR EXIT FROM THIS SUBROUTINE
004076  000207                          RTS PC                          ; EXIT TO SERVICE TRANSFER REQUEST

004100  004737  006574          1$:     JSR PC, SDN
004104  000767                          BR FBEB                         ; NOT TRANSFER REQUEST OR DONE
004106  005777  175074                  TST @ RXCS                      ;TEST FOR ERROR FLAG
004112  100003                          BPL 3$
004114  017701  175070                  MOV @RXDB,R1

                                ; (R0) = N/A ; (R1) = RXDB (STATUS A) ; (R2) = ACTUAL # OF TRANSFERS

004120  104000                          ERROR                           ; UNEXPECTED RX11 ERROR FLAG


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

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

1672  004122  104412                  36:     SUBSCOPE
1673
1674                                  ; 128 BYTES SHOULD HAVE BEEN TRANSFERRED TO OR FROM THE SECTOR BUFFER
1675
1676  004124  022702  000200                  CMP #200,R2
1677  004130  001401                          BEQ 2$
1678
1679                                  ; (R0) = N/A ; (R1) = N/A ; (R2) = ACTUAL # OF TRANSFERS
1680
1681  004132  104000                          ERROR                           ; INCORRECT TRANSFER LENGTH
1682

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

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

1706  004134  000004                  2$:     SCOPE
1707
1708  004136  000207                          RTS PC                          ; EXIT SUBROUTINE FBEB
```



```
1709                                            .SBTTL TEST 10 - EMPTY BUFFER XFER LENGTH AND CONTENT VERIFICATION PART II
1710
1711                                  ; THE PURPOSE OF THIS TEST IS TO VERIFY THAT THE PREVIOUS EMPTY BUFFER TEST
1712                                  ; DID NOT EMPTY AND DESTROY THE CONTENTS OF THE SECTOR BUFFER
1713                                  ;NOTE: TEST 6 MUST BE RUN BEFORE THIS TEST.
1714
1715  004140  000240                  T10:    NOP                             ; NOP FOR T10 " FAST " DEFINITION
1716
1717                                          .SBTTL TEST 7 - EMPTY BUFFER XFER LENGTH AND CONTENT VERIFICATION PART I
1718
1719                                  ; THE PURPOSE OF THIS TEST IS TO VERIFY THE TRANSFER LENGTH OF THE FOUNCTION
1720                                  ; "EMPTY BUFFER" AND ALSO TO VERIFY THE CONTENTS OF THE SECTOR BUFFER
1721                                  ; NOTE TEST 6 MUST BE RUN BEFORE THIS TEST
1722
1723  004142  004737  004152          T7:     JSR PC, T7EMPTY
1724  004146  000137  004246                  JMP CEXIT                       ;END OF TEST 7 OR 10
1725  004152  005002                  T7EMPTY: CLR R2
1726  004154  012777  000003  175024          MOV #3, @ RXCS                  ; ISSUE THE COMAND TO EMPTY BUFFER
1727  004162  012704  017404                  MOV #BUFADR, R4
1728  004166  004737  004064          2$:     JSR PC, FBEB                    ; SUBROUTINE TO EMPTY BUFFER
1729  004172  000207                          RTS PC                          ; EXIT SUBROUTINE T7EMPTY
1730
1731                                  ; EMPTY BUFFER AND VERIFY THE CONTENTS
1732
1733  004174  112400                          MOVB (R4)+, R0                  ; EXPECTED CONTENTS OF SECTOR BUFFER
1734  004176  117701  175006                  MOVB @ RXDB, R1                 ; ACTUAL CONTENTS OF SECTOR BUFFER
1735  004202  005202                          INC R2                          ; ACTUAL # TRANSFERS TO THIS ERROR
1736  004204  020001                          CMP R0, R1
1737  004206  001401                          BEQ 1$
1738
1739                                  ; IF AN ERROR OCCURS:
1740
1741                                  ; (R0) = EXPECTED CONTENTS OF SECTOR BUFFER
1742                                  ; (R1) = ACTUAL CONTENTS FROM SECTOR BUFFER
1743                                  ; (R2) = TOTAL # OF ACTUAL TRANSFERS
1744
1745  004210  104000                          ERROR                           ; DATA " TO " SB NOT = DATA " FROM "
1746

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

                ;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                    ----------
;
;/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*
;/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*
;/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*/;\*

004212  000004                  1$:     SCOPE
004214  000764                          BR 2$

                                        .SBTTL TEST 12 - FILL/EMPTY BUFFER ALL 1'S

                                        ; PROGRAMMING NOTE; THE FOLLOWING " MOV " INSTEAD OF " INC " FOR LOOPS

004216  012737  000001  012324  T12:    MOV #1, PAT
004224  000402                          BR .+6

                                        .SBTTL TEST 11 - FILL/EMPTY BUFFER ALL 0'S

004226  005037  012324          T11:    CLR PAT
004232  004737  004026                  JSR PC, T6FILL
004236  004737  004152                  JSR PC, T7EMPTY
004242  000137  004246                  JMP CEXIT               ;END OF TEST 11 OR 12
```



```
004246  012737  000103  002534  CEXIT:  MOV #'C,MX              ;EOP INDICATOR FOR END OF CONTROL TESTS
004254  000137  002162                  JMP MORETESTS

004260  012737  000104  002534  DEXIT:  MOV #'D,MX              ;EOP INDICATOR FOR END OF DRIVE TESTS
004266  000137  002162                  JMP MORETESTS

                                        .SBTTL TEST 13 DRIVE READY VERIFICATION

                                        ;THIS TEST VERIFIES THAT DRIVE READY WILL SET FOR ALL SELECTED DRIVES.

                                        ;THIS SECTION OF TEST FOR UNIT 0 DRY NO A READ STATUS A FUNCTION

004272  105737  012724          T13:    TSTB UNITSEL            ;WAS UNIT 0 SELECTED
004276  100020                          BPL 2$                  ;NO, GO TEST UNIT 1
004300  012777  000013  174700          MOV #13,@RXCS           ;READ STATUS A UNIT 0
004306  004737  006574          1$:     JSR PC,SDN              ;WAIT FOR DONE FLAG
004312  000775                          BR 1$
004314  012700  000200                  MOV #200,R0             ;EXPECT DRIVE READY TO BE SET
004320  017702  174664                  MOV @RXDB,R2            ;ACTUAL RXDB
004324  010201                          MOV R2,R1
004326  042701  000100                  BIC #BIT6,R1            ;CLEAR DD BIT IF SET
004332  020001                          CMP R0,R1               ;WAS " DRY " SET AND INITDONE CLEARED
004334  001401                          BEQ 2$

                                        ;R0 = 200 ; P1 = RXDB MINUS DD BIT ;R2 = ACTUAL RXDB

004336  104000                          EPROR
004340  104412                  2$:     SUBSCOPE

                                        ;TEST FOR UNIT 1 DRIVE READY ON A READ STATUS A FUNCTION

004342  005737  012724                  TST UNITSEL             ;WAS UNIT 1 SELECTED
004346  100020                          BPL 3$                  ;NO, GO TO END OF TEST
004350  012777  000033  174630          MOV #33,@RXCS           ;READ STATUS A FOR DRIVE 1
004356  004737  006574          4$:     JSR PC,SDN              ;WAIT FOR DONE FLAG
004362  000775                          BR 4$
004364  012700  000200                  MOV #200,R0             ;EXPECT " DRY " TO BE SET
004370  017702  174614                  MOV @RXDB,R2
004374  010201                          MOV R2,R1
004376  042701  000100                  BIC #BIT6,R1            ;CLEAR DD BIT IF ANY
004402  020100                          CMP R1,R0               ;IS DRY SET AND INITDONE CLEARED
004404  001401                          BEQ 3$

                                        ;R0 = 200 ; R1 = RXDB MINUS DD BIT ; P2 = ACTUAL RXDB

004406  104000                          ERROR

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

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

1895  004410  000004                  3$:     SCOPE
1896  004412  000137  004260                  JMP DEXIT                       ;END OF TEST 13
```



```
1897                                            .SBTTL TEST 14 - ERROR FLAG AND B-CODE VERIFICATION PART 1
1898
1899                                    ;THE PURPOSE OF THIS TEST IS TO VERIFY THAT TRYING TO READ A NON-EXISTANT
1900                                    ;SECTOR WILL CAUSE AN ERROR, AND THE CORRECT ERROR CODE WILL BE PUT
1901                                    ;INTO THE RXDB WHEN THE B STATUS IS READ.
1902
1903                                    ;THIS SECTION INSURES THAT ONLY 2 TR FLAGS WERE REQUIRED TO TAKE THE
1904                                    ;DISKETTE ADDRESS, AND THAT AN ERROR DOES EXIST.
1905
1906  004416  005002                    T14:    CLR R2
1907  004420  005000                            CLR R0
1908  004422  105737  012724                    TSTB UNITSEL                    ;IS UNIT 0 SELECTED
1909  004426  100004                            BPL 10$
1910  004430  012777  000007  174550            MOV #7,@RXCS                    ;SET READ SECTOR FUNCTION AND GO
1911  004436  000403                            BR 11$
1912  004440  012777  000027  174540    10$:    MOV #27,@RXCS                   ;SEND READ FUNCTION TO UNIT 1
1913  004446  004737  005106            11$:    JSR PC,ILLADR                   ;SEND THE BAD SECTOR ADDRESS
1914  004452  000406                            BR 1$                           ;PREMATURE ERROR CONDITION
1915  004454  012700  100040                    MOV #100040,R0                  ;EXPECT ERROR AND DONE BITS
1916  004460  017701  174522                    MOV @RXCS,R1                    ;SAVE THE RXCS
1917  004464  020001                            CMP R0,R1
1918  004466  001401                            BEQ 2$
1919
1920                                    ;R0 = 100040 ; R1 = ACTUAL RXCS ; R2 = # OF TR FLAGS
1921
1922  004470  104000                    1$:     ERROR
1923

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

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

1948  004472  104412                    2$:     SUBSCOPE
1949
1950                                    ;T14 CONT. - THIS SECTION TESTS THAT NO PARITY OR CRC ERROR OCCURRED
1951                                    ;ON PREVIOUS FUNCTION.
1952
```

```
1953  004474  005000                          CLR R0                          ;STATUS A SHOULD BE CLEAR
1954  004476  017702  174506                  MOV @RXDB,R2
1955  004502  010201                          MOV R2,R1
1956  004504  042701  000100                  BIC #BIT6,R1
1957  004510  020001                          CMP R0,R1
1958  004512  001401                          BEQ 3$
1959
1960                                  ;R0 = 0 ; R1 = RXDB MINUS DD BIT ; R2 = ACTUAL RXDB
1961
1962  004514  104000                          ERROR
1963  004516  104412                  3$:     SUBSCOPE
1964
1965                                  ;T14 CONT. - THIS SECTION TESTS THAT NO ERROR CONDITIONS EXIST ON A
1966                                  ;READ STATUS B FUNCTION.
1967
1968  004520  012777  000017  174460          MOV #17,@RXCS                   ;SET READ STATUS B FUNCTION
1969  004526  004737  006574          4$:     JSR PC,SDN                      ;WAIT FOR DONE FLAG
1970  004532  000775                          BR 4$
1971  004534  005777  174446                  TST @RXCS                       ;IS THE ERROR BIT SET
1972  004540  100007                          BPL 5$                          ;NO, GO ON TO NEXT SECTION
1973  004542  012700  000040                  MOV #40,R0                      ;YES,SET UP FOR ERROR
1974  004546  017701  174434                  MOV @RXCS,R1
1975  004552  017702  174432                  MOV @RXDB,R2
1976
1977                                  ;R0 = 40 ; R1 = RXCS ; R2 = RXDB
1978
1979  004556  104000                          ERROR                           ;RXCS NOT = 40
1980  004560  104412                  5$:     SUBSCOPE
1981
1982                                  ;T14 CONT. - THIS SECTION TESTS THE B-CODE FOR " CAN'T FIND SECTOR " #70
1983
1984  004562  012700  000070                  MOV #70,R0
1985  004566  017701  174416                  MOV @RXDB,R1
1986  004572  005002                          CLR R2
1987  004574  020001                          CMP R0,R1                       ;IS THE B-CODE = 70
1988  004576  001401                          BEQ 6$
1989
1990                                  ;R0 = 70 ; R1 = ACTUAL B-CODE ; R2 = N/A
1991
1992  004600  104000                          ERROR                           ;RXDB NOT = 70
1993

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

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

2029  004602  000004                  6$:     SCOPE
2030  004604  000137  004260                  JMP DEXIT                       ;END OF TEST 14
2031
```

```
                                        .SBTTL TEST 15 - ERROR FLAG AND B-CODE VERIFICATION PART II

                                ;THIS TEST VERIFIES THAT TRYING TO WRITE,USING DELETED DATA MODE, ON A
                                ;NON-EXISTANT SECTOR WILL PRODUCE AN ERROR AND THE CORRECT B-CODE IS PRODUCED
                                ;THIS SECTION SENDS OUT AN ILLEGAL SECTOR ADDRESS AND EXPECTS AN ERROR
                                ; NOTE TEST 14 MUST BE RUN BEFORE THIS TEST

004610  005000          T15:    CLR R0
004612  005002                  CLR R2
004614  105737  012724          TSTB UNITSEL                    ;WAS UNIT 0 SELECTED
004620  100004                  BPL 10$
004622  012777  000015  174356  MOV #15,@RXCS                   ;SET WRITE DELETED DATA FUNCTION
004630  000403                  BR 11$
004632  012777  000035  174346  10$:    MOV #35,@RXCS           ;SEND WTR DD FUNCTION TO UNIT 1
004640  004737  005106          11$:    JSR PC,ILLADR           ;SEND THE ILLEGAL SECTOR ADDRESS
004644  000406                  BR 1$                           ;PREMATURE ERROR CONDITION
004646  012700  100040          MOV #100040,R0                  ;EXPECT ERROR AND DONE FLAGS
004652  017701  174330          MOV @RXCS,R1
004656  020001                  CMP R0,R1
004660  001401                  BEQ 2$

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

004662  104000          1$:     ERROR                           ;RXCS NOT = 100040
004664  104412          2$:     SUBSCOPE

                                ;T15 CONT. - THIS SECTION TESTS THAT THERE IS NO PARITY, CRC ERROR
                                ;AND THAT THE DELETED DATA BIT IS SET.

004666  005002                  CLR R2
004670  012700  000100          MOV #100,R0                     ;EXPECT DELETED DATA BIT TO BE SET
004674  017701  174310          MOV @RXDB,R1
004700  020001                  CMP R0,R1
004702  001401                  BEQ 3$

                                ; R0 = 100 ; R1 = ACTUAL RXDB ; R2 = N/A
004704  104000                  ERROR                           ;DELETED DATA NOT SET OR OTHER ERRORS
004706  104412          3$:     SUBSCOPE

                                ;T15 CONT. - THIS SECTION TESTS FOR THE B-CODE FOR ILLEGAL SECTOR.

004710  012777  000017  174270          MOV #17,@RXCS           ;SET READ STATUS B FUNCTION
004716  004737  006574          4$:     JSR PC,SDN              ;WAIT FOR DONE FLAG
004722  000775                  BR 4$
004724  012700  000070          MOV #70,R0
004730  017701  174254          MOV @RXDB,R1
004734  020001                  CMP R0,R1
004736  001401                  BEQ 5$

                                ; R0 = 70 ; R1 = ACTUAL B-CODE ; R2 = N/A
004740  104000                  ERROR                           ;RXDB NOT = 70
004742  000004          5$:     SCOPE
004744  000137  004260          JMP DEXIT                       ;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$:     ERROR
005024  104412          2$:     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  104412          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
```

```
2137
2138                                    ;GENERATE AN ILLEGAL SECTOR ADDRESS
2139
2140  005106  004737  006560            ILLADR: JSR PC,STR                      ;LOOK FOR A TR FLAG
2141  005112  000402                            BR 2$                           ;NO TR FLAG, IS DONE SET
2142  005114  005202                            INC R2                          ;TR FLAG COUNTER
2143  005116  000404                            BR 3$
2144  005120  004737  006574            2$:     JSR PC,SDN                      ;LOOK FOR DONE FLAG
2145  005124  000770                            BR ILLADR                       ;NOT DONE RECHECK TR FLAG
2146  005126  000430                            BR 1$                           ;DONE IS SET TOO EARLY GO TO ERROR
2147  005130  005077  174054            3$:     CLR @RXDB                       ;0 SECTOR ADDRESS (ILLEGAL)
2148  005134  004737  006560            7$:     JSR PC,STR                      ;LOOK FOR SECOND TR FLAG
2149  005140  000402                            BR 5$                           ;NOT TR, IS IT DONE
2150  005142  005202                            INC R2
2151  005144  000404                            BR 6$                           ;TR FLAG SEND TRACK ADDRESS
2152  005146  004737  006574            5$:     JSR PC,SDN                      ;LOOK FOR DONE FLAG
2153  005152  000770                            BR 7$                           ;NOT DONE, RECHECK TR FLAG
2154  005154  000415                            BR 1$                           ;DONE TOO SOON GO TO ERROR
2155  005156  005077  174026            6$:     CLR @RXDB                       ;SEND TRACK ADDRESS OF 0
2156  005162  004737  006560            11$:    JSR PC,STR                      ;ARE THERE ANY MORE TR FLAGS
2157  005166  000402                            BR 10$                          ;NO, LOOK FOR DONE
2158  005170  005202                            INC R2                          ;YES
2159  005172  000406                            BR 1$                           ;TOO MANY TR FLAGS OR MICROCONTROLLER
2160                                                                            ;DID NOT DETECT THE ERROR
2161  005174  004737  006574            10$:    JSR PC,SDN                      ;LOOK FOR DONE FLAG
2162  005200  000770                            BR 11$                          ;NOT DONE RETEST TR FLAG
2163  005202  062716  000002                    ADD #2,@SP
2164  005206  000207            4$:     RTS PC
2165  005210  017701  173772            1$:     MOV @RXCS,R1
2166  005214  000774                            BR 4$
```



```
2167
2168                                            .SBTTL TEST 17 - ILLEGAL TRACK ERROR VERIFICATION
2169
2170                                    ;THIS TEST VERIFIES THAT IF A TRACK ADDRESS LARGER THAN 114 (OCTAL) IS
2171                                    ;ACCESSED, AN ERROR CONDITION WILL EXIST, AND A B-CODE WILL = 40
2172
2173  005216  005002            T17:    CLR R2
2174  005220  005000                    CLR R0
2175  005222  012777  000007  173756    MOV #7,@RXCS                    ;SET READ FUNCTION
2176  005230  004737  006560    3$:     JSR PC,STR                      ;LOOK FOR TR FLAG
2177  005234  000401                    BR 1$                           ;NO TR FLAG CHECK DONE
2178  005236  000410                    BR 2$
2179  005240  004737  006574    1$:     JSR PC,SDN
2180  005244  000771                    BR 3$
2181  005246  017701  173734            MOV @RXCS,R1                    ;DONE OCCURRED TOO SOON SET UP FOR ERROR
2182  005252  017702  173732            MOV @RXDB,R2
2183  005256  000433                    BR 4$
2184  005260  012777  000001  173722    MOV #1,@RXDB                    ;SEND LEGAL SECTOR ADDRESS
2185  005266  004737  006560    5$:     JSR PC,STR                      ;LOOK FOR TR FLAG
2186  005272  000401                    BR 6$
2187  005274  000410                    BR 7$
2188  005276  004737  006574    6$:     JSR PC,SDN
2189  005302  000771                    BR 5$
2190  005304  017701  173676            MOV @RXCS,R1                    ;DONE SET TOO EARLY
2191  005310  017702  173674            MOV @RXDB,R2
2192  005314  000414                    BR 4$
2193  005316  012777  000115  173664    7$:     MOV #115,@RXDB          ;SEND ILLEGAL TRACK ADDRESS
2194  005324  004737  006574    10$:    JSR PC,SDN                      ;WAIT FOR DONE ON THE ERROR
2195  005330  000775                    BR 10$
2196  005332  012700  100040            MOV #100040,R0                  ;EXPECT ERROR AND DONE SET
2197  005336  017701  173644            MOV @RXCS,R1
2198  005342  020001                    CMP R0,R1
2199  005344  001401                    BEQ 11$
2200
2201                                    ;TWO ERROR CONDITIONS TO REPORT
2202                                    ;IF R0 = 0 THEN R1 = RXCS ;R2 = RXDB ON A DONE TOO SOON ERROR
2203                                    ;IF R0 = 100040 THEN R1 = ACTUAL RXCS ; R2 = N/A
2204
2205  005346  104000            4$:     ERROR                           ;DONE SET TOO SOON OR NO ERROR OCCURRED
2206

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

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

  2230  005350  104412                  11$:    SUBSCOPE
  2231
  2232                                  ;T17 CONT. - TEST THAT THERE WERE NO OTHER ERRORS THAN THE ILLEGAL TRACK ERROR EXPECTED
  2233                                  ;AND THAT THE DELETED DATA BIT WAS CLEARED BY TEST 16.
  2234
  2235  005352  005000                          CLR R0                          ;DD BIT CLEARED IN TEST 16 SO RXDB = 0
  2236  005354  005002                          CLR R2
  2237  005356  017701  173626                  MOV @RXDB,R1
  2238  005362  020001                          CMP R0,R1
  2239  005364  001401                          BEQ 12$
  2240
  2241                                  ; R0 = 0 ; R1 = ACTUAL RXDB ; R2 = N/A
  2242
  2243  005366  104000                          ERROR
  2244  005370  104412                  12$:    SUBSCOPE
  2245
  2246                                  ;T17 CONT. - VERIFIES THAT READ STATUS B HAS NO ERRORS
  2247
  2248  005372  012777  000017  173606          MOV #17,@RXCS                   ;SET READ STATUS B FUNCTION
  2249  005400  004737  006574          13$:    JSR PC,SDN                      ;WAIT FOR DONE FLAG
  2250  005404  000775                          BR 13$
  2251  005406  005777  173574                  TST @RXCS                       ;WAS THERE AN ERROR ON THIS FUNCTION
  2252  005412  100007                          BPL 14$                         ;NO
  2253  005414  012700  000040                  MOV #40,R0
  2254  005420  017701  173562                  MOV @RXCS,R1
  2255  005424  017702  173560                  MOV @RXDB,R2
  2256
  2257                                  ; R0 = 40 (DONE) ; R1 = ACTUAL RXCS ; R2 = ACTUAL RXDB
  2258
  2259  005430  104000                          ERROR                           ;THERE WAS AN ERROR NO READ STATUS B
  2260  005432  104412                  14$:    SUBSCOPE
  2261  005434  005002                          CLR R2
  2262  005436  012700  000040                  MOV #40,R0                      ;B-CODE FOR ILLEGAL TRACK ADDRESS
  2263  005442  017701  173542                  MOV @RXDB,R1
  2264  005446  020001                          CMP R0,R1
  2265  005450  001401                          BEQ 15$                         ;B-CODE IS CORRECT
  2266
  2267                                  ; R0 = 40 ; R1 = ACTUAL B-CODE ; R2 = N/A
  2268
  2269  005452  104000                          ERROR
  2270  005454  000004                  15$:    SCOPE
  2271  005456  000137  004260                  JMP DEXIT                       ;END OF TEST 17
```



```
  2272
  2273                                          .SBTTL TEST 20 - SEEK VERIFICATION VIA READ FUNCTION
  2274
  2275                                  ; THE PURPOSE OF THIS TEST IS TO DO A READ FUNCTION  ON ALL
  2276                                  ;SECTORS OF VARIOUS TRACKS ON THE DISKETTE, THIS WILL TEST FOR SEEK ERRORS
  2277                                  ;FOR THOSE TRACK POSITIONS, UNLESS OTHERWISE SELECTED (IN OD/ID) THE TRACKS
  2278                                  ;ACCESSED ARE 0 (OD), 52, 53 (BOTH SIDES OF THE WRITE CURRENT CHANGE), AND 114 (ID).
  2279
  2280
  2281  005462                  T20:
  2282

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

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

  2304  005462  013702  001204                  MOV KRXVEC,R2                   ;SET UP INTERRUPT ADDRESSES
  2305  005466  012722  011526                  MOV #INTSERV,(R2)+
  2306  005472  012722  000340                  MOV #PR7,(R2)+
  2307  005476  004737  006762                  JSR PC,RDONLY                   ;DO THE READ FUNCTION
  2308  005502  000137  004260                  JMP DEXIT                       ;END OF TEST 20
  2309
  2310
  2311                                          .SBTTL TEST 21 - WRITE TEST
  2312
  2313                                  ;THE PURPOSE OF THIS TEST IS TO WRITE ALL ONES ON SECTOR 1 TRACK 1,
  2314                                  ;AND VERIFY THAT THE DATA IN THE SECTOR BUFFER IS NOT MODIFIED.
  2315
  2316  005506  012737  000001  013126  T21:    MOV #1,TARGET
  2317  005514  012737  000001  013416          MOV #1,TSECTOR
  2318  005522  004737  012634                  JSR PC,GETUNIT
  2319  005526  012737  000001  012324          MOV #1,PAT
  2320  005534  004737  012260                  JSR PC,GETPATTERN
  2321  005540  004737  010766                  JSR PC,ADJSUM                   ;SET CHECK SUM VALUES
  2322  005544  005002                          CLR R2
  2323  005546  012777  000001  173432          MOV #1,@RXCS                    ;SET FILL BUFFER FUNCTION
  2324  005554  004737  004064          1$:     JSR PC,FBEB
  2325  005560  000404                          BR 2$
  2326  005562  112077  173422                  MOVB (R0)+,@RXDB
  2327  005566  005202                          INC R2
```

```
  2328  005570  000771                          BR 1$
  2329  005572  012737  000005  007702  2$:     MOV #5,FUNCTION                 ;SET WRITE FUNCTION
  2330  005600  004737  005610                  JSR PC,T21X                     ;GO ISSUE THE COMMAND
  2331  005604  000137  004260                  JMP DEXIT                       ;END OF TEST 21
```



```
  2332  005610  004737  007634          T21X:   JSR PC,COMMWORD
  2333  005614  004737  006574          3$:     JSR PC,SDN
  2334  005620  000775                          BR 3$
  2335  005622  005777  173360                  TST @RXCS                       ;IS THE ERROR BIT SET
  2336  005626  100013                          BPL 4$                          ;NO ERRORS
  2337  005630  017701  173354                  MOV @RXDB,R1                    ;SAVE STATUS A
  2338  005634  012777  000017  173344          MOV #17,@RXCS                   ;SET READ STATUS B FUNCTION
  2339  005642  004737  006574          5$:     JSR PC,SDN
  2340  005646  000775                          BR 5$
  2341  005650  017702  173334                  MOV @RXDB,R2                    ;SAVE STATUS B
  2342
  2343                                  ;R0 = ADDR OF LAST DATA BYTE ; R1 = STATUS A ; R2 = STATUS B
  2344
  2345  005654  104000                          ERROR
  2346

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

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

  2375  005656  104412                  4$:     SUBSCOPE
  2376  005660  005000                          CLR R0
  2377  005662  005001                          CLR R1
  2378  005664  004737  004152                  JSR PC,T7EMPTY                  ;EMPTY BUFFER AND CHECK CONTENTS
  2379  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
```



```
                                        .SBTTL TEST 24 - DATA TRANSFER AND VERIFICATION

                                ;THE PURPOSE OF THIS TEST IS TO WRITE THEN READ AND VERIFY DATA
                                ;ON ALL SECTORS OF THE SELECTED TRACKS. THE TEST ALTERNATES BETWEEN
                                ;DRIVES ON THE SELECTED TRACKS. DATA PATTERN IS A FLOATING 0.

006016  012737  000002  012324  T24:    MOV #2,PAT                      ;SET DATA PATTERN TO FLOATING 0
006024  013702  001204          T24X:   MOV KRXVEC,R2                   ;SET INTERRUPT ADDRESSES
006030  012722  011526                  MOV #INTSERV,(R2)+
006034  012712  000340                  MOV #PR7,(R2)
006040  004737  007012                  JSR PC,DRVSWP                   ;GO TRANSFER THE DATA
006044  000137  004260                  JMP DEXIT                       ;END OF TEST 24 OR 25

                                        .SBTTL TEST 25 - DATA TRANSFER AND VERIFICATION VIA DELETED DATA MODE

                                ;THIS TEST TRANSFERES DATA JUST LIKE TEST 24 EXCEPT IT USES THE
                                ;DELETED DATA FORMAT AND A DATA PATTERN OF FLOATING 1.

006050  012737  000003  012324  T25:    MOV #3,PAT                      ;SET DATA PATTERN TO FLOATING 1
006056  000137  006024                  JMP T24X                        ;GO TRANSFER THE DATA

                                        .SBTTL TEST 26 - HEAD "HOME" TEST

                                ;THIS TEST MOVES THE HEAD OUT TO TRACK 12 (OCTAL) AND THEN WRITES/READ CHECKS
                                ;ALL SECTORS (RANDOM DATA) ON EACH TRACK. THE TRACK SEQUENCE
                                ;IS DECREMENT;>ED BACK TO TRACK 0 (HOME). AFTER COMPLETING
                                ;DRIVE 0 IT SWITCHES OVER TO DRIVE 1 DOING THE SAME TEST.

006062  052737  000200  013136  T26:    BIS #BIT7,SEQUEN                ;SPECIAL DECREMENT SEQUENCE
006070  012737  000007  012324          MOV #7,PAT                      ;SELECT RANDOM DATA
006076  013702  001204                  MOV KRXVEC,R2
006102  012722  011526                  MOV #INTSERV,(R2)+
006106  012712  000340                  MOV #PR7,(R2)
006112  004737  007066                  JSR PC,WTRDCK
006116  042737  000200  013136          BIC #BIT7,SEQUEN
006124  000137  004260                  JMP DEXIT                       ;END OF TEST 26
```

```
                                        ;THE FOLLOWING SECTION OF CODE WILL ALLOW PROVIDING INFORMATION
                                        ;TO THE USER WHEN AN 'UNEXPECTED' BUS TIMEOUT TO LOCATION 4 OCCURS

006130  104400  016734        BUSERR: TYPE,   LOC4M                   ;TYPE MESSAGE INDICATING AN
                                                                      ;UNEXPECTED BUS TIMEOUT OCCURRED
006134  012646                        MOV     (SP)+,-(SP)     ;;SAVE (SP)+ FOR TYPEOUT
                                                                ;;SETUP TO TYPE PC WHERE TIMEOUT OCCURRED
006136  104402                        TYPOS                   ;;GO TYPE--OCTAL ASCII
006140      006                     .BYTE   6               ;;TYPE 6 DIGITS
006141      000                     .BYTE   0               ;;SUPPRESS LEADING ZEROS
006142  104400  017047              TYPE,   PCM                     ;TYPE MESSAGE '=PC'
006146  012716  002466              MOV     #REBEGIN,(SP)           ;SET RETURN 'PC' TO START THE
                                                                      ;PROGRAM OVER AGAIN
006152  000002                      RTI                             ;RETURN TO BEGINNING OF PROGRAM

                                        ;THE FOLLOWING SECTION OF CODE WILL ALLOW PROVIDING INFORMATION
                                        ;TO THE USER WHEN AN 'UNEXPECTED' RESERVED INSTRUCTION TRAP TO LOCATION
                                        ;10 OCCURS

006154  104400  017001        RESERR: TYPE,   LOC10M                  ;TYPE MESSAGE INDICATING AN
                                                                      ;UNEXPECTED RESERVED INSTRUCTION
                                                                      ;TRAP OCCURRED
006160  012646                        MOV     (SP)+,-(SP)     ;;SAVE (SP)+ FOR TYPEOUT
                                                                ;;SETUP TO TYPE PC WHERE RESERVED TRAP OCCURRED
006162  104402                        TYPOS                   ;;GO TYPE--OCTAL ASCII
006164      006                     .BYTE   6               ;;TYPE 6 DIGITS
006165      000                     .BYTE   0               ;;SUPPRESS LEADING ZEROS
006166  104400  017047              TYPE,   PCM                     ;TYPE MESSAGE '=PC'
006172  012716  002466              MOV     #REBEGIN,(SP)           ;SET RETURN 'PC' TO START THE
                                                                      ;PROGRAM OVER AGAIN
006176  000002                      RTI                             ;RETURN TO BEGINNING OF PROGRAM

                                        ;THIS ROUTINE WILL CALCULATE THE PRIORITY LEVEL FOR THE PROCESSOR
                                        ;BASED ON THE CURRENT PRIORITY LEVEL OF THE DEVICE (CONTENTS OF 'BRLEV:')

006200  013700  001214        CPUPRI: MOV     BRLEV,R0                ;GET THE PROPOSED RX11 DEVICE
                                                                      ;INTERRUPT PRIORITY LEVEL VALUE
006204  105701                        TSTB    R1                      ;IS CPU LEVEL TO BE THE SAME AS
                                                                      ;THE DEVICE LEVEL OR 1 LESS?
006206  100401                        BMI     1$                      ;BRANCH IF SAME AS!
006210  005300                        DEC     R0                      ;DROP DEVICE LEVEL PRIORITY
                                                                      ;BY 1 LEVEL FOR PSW
006212  006300                1$:     ASL     R0                      ;FORM BITS <7-5> FOR PSW
006214  006300                        ASL     R0                      ;       .
006216  006300                        ASL     R0                      ;       .
006220  006300                        ASL     R0                      ;       .
006222  006300                        ASL     R0                      ;       .
006224  042700  000037              BIC     #37,R0                  ;ENSURE THAT T,N,Z,V, & C BITS
                                                                      ;FOR THE PROCESSOR ARE CLEAR
006230  000207                      RTS     PC                      ;RETURN TO MAINLINE CODE
```



```
                                        .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  104405                        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  104400  015636                TYPE, MXEHEADER

006304  104400  016120        1$:     TYPE, MCRLF
006310  011604                        MOV @ SP, R4                    ; ERADR
006312  162704  000002                SUB #2, R4
006316  010446                        MOV R4, -(SP)
006320  104402                        TYPOS
006322      006                     .BYTE 6
006323      001                     .BYTE 1
006324  104400  016641                TYPE, SPACE
006330  013746  002532                MOV FAST, -(SP)                 ; FAST (FIRST ADDRESS OF SELECTED TEST)
006334  104403                        TYPON
006336  104400  016641                TYPE, SPACE
006342  013746  006556                MOV PCSCOPE, -(SP)              ; FAPT (FIRST ADDRESS OF PRESENT TEST)
006346  104403                        TYPON
006350  104400  016641                TYPE, SPACE
006354  010246                        MOV R2, -(SP)                   ; BLANK
006356  104403                        TYPON
006360  104400  016641                TYPE, SPACE
006364  010046                        MOV R0, -(SP)                   ; EXPECTED (GOOD) RESULT OF TEST
006366  104403                        TYPON
006370  104400  016641                TYPE, SPACE
006374  010146                        MOV R1, -(SP)                   ; ACTUAL (BAD) RESULT OF TEST
006376  104403                        TYPON
006400  104400  016641                TYPE, SPACE
006404  013737  002530  006416        MOV PASS,2$
006412  004537  015600                JSR R5,SGLDEC
006416  000000                2$:     OPEN
```

```
                                        ; DATA SW 0 = 0 TO RING BELL AT ERROR
                                        ;
006420  052737  000400  012724  NOPRINT: BIS #BIT8,UNITSEL          ;SET HARD ERROR FLAG
006426  004737  006450                  JSR PC,DING
                                        ;
                                        ; DATA SW 15 = 1 TO HALT AT ERROR
                                        ;
006432  104405                  1$:     CKSWR
006434  032777  100000  172554          BIT #SW15,@SWR
006442  001401                          BEQ 2$
006444  000000                          HALT
006446  000002                  2$:     RTI
                                        ;
006450  104405                  DING:   CKSWR
006452  032777  000001  172536          BIT #SW0,@SWR
006460  001002                          BNE 1$
006462  104400  006470                  TYPE ,MABELL
006466  000207                  1$:     RTS PC
                                        ;
006470  000007                  MABELL: .ASCIZ <07>                 ; DING - A - LING
                                        .EVEN
                                        ;
006472  000000                  ERRORS: 0
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  104405                  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 ; (P1) = 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,F15,E1,E19,E18,E22
        ;
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

                                        ;
006624  005037  006756                  CLR HANGER                      ;RESET THE HANG COUNTERS
006630  012737  177740  006760          MOV #177740,HANGPL
006636  104405                          CKSWR
006640  032777  020000  172350          BIT #SW13,@SWR
006646  001402                          BEQ 1$
006650  000137  002162                  JMP MORETESTS
006654  104400  006674          1$:     TYPE, MHUNGPC
                                        ;
                                        ; THE PC IS ALREADY ON THE STACK
                                        ;
006660  162716  000002                  SUB #2, @ SP
006664  104402                          TYPOS
006666     006                          .BYTE 6
006667     001                          .BYTE 1
006670  000137  002162                  JMP MORETESTS
```



```
                                        ;
006674  005015  042504  044526  MHUNGPC: .ASCIZ <15><12>"DEVICE TEST HUNG @ PC "
006702  042503  052040  051505
006710  020124  052510  043516
006716  040040  050040  020103
006724     000
006725     040  040520  051523  MPASS:  .ASCIZ " PASS ="
006732  036440     000
        006736                          .EVEN
                                        ;
006736  005037  006756          XSDN:   CLR HANGER
006742  012737  177740  006760          MOV #177740,HANGPL
006750  062716  000002                  ADD #2, @ SP    .               ; UPDATE FOR EXIT
006754  000207                          RTS PC
006756  000000                  HANGER: 0
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  016366  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  104405          PARTEST:        CKSWR
007254  032777  020000  171734          BIT #SW13,@SWR           ;TEST DON'T PRINT ERROR SWITCH
007262  001006                          BNE CONT4
007264  104400  000000  PTYP1:          TYPE ,OPEN               ;PRINT THE PARITY ERROR MESSAGE
007270  104400  016577                  TYPE ,MPAR
007274  104400  016120                  TYPE ,MCRLF
007300  104405          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$:     INC HANGER               ;WAIT FOR INTERRUPT
007422  001375                          BNE 3$
007424  000137  011462                  JMP NOINTER              ;NO INTERRUPT ERROR
```

```
2801  007430  005337  013414          WRTDONE:        DEC SECCNTR             ;TEST SECTOR COUNTER
2802  007434  001001                                  BNE 2$                  ;NOT LAST SECTOR GO TO NEXT ONE
2803  007436  000207                                  RTS PC
2804  007440  000137  007140          2$:             JMP XWRITE
2805
2806  007444  005726                  WRTER:          TST (SP)+               ;REMOVE THE DONE RETURN FROM THE STACK
2807  007446  032737  000002  012146                  BIT #BIT1,ASTAT         ;IS THIS A PARITY ERROR
2808  007454  001413                                  BEQ WRTSEK              ;NO, IT MUST BE A SEEK ERROR
2809                                  ;PARITY ERROR DURING A WRITE FUNCTION
2810  007456  012737  016570  007266                  MOV #MWRITE,PTYP1+2     ;PUT ADDR OF WRITE MESSAGE IN PAR ER TYPOUT 1
2811  007464  012737  007430  007330                  MOV #WRTDONE,PCONT+2    ;IF NO LOOP GO TO NEXT SECTOR
2812  007472  012737  007332  007324                  MOV #FILLDONE,PLOOP+2   ;IF LOOP RETURN THROUGH PLOOP TO REWRITE
2813  007500  000137  007252                          JMP PARTEST             ;GO INC LOG AND TEST FOR RETRY
2814
2815                                  ;SEEK ERROR DURING A WRITE FUNCTION
2816  007504  012737  007144  007604  WRTSEK:         MOV #FILLBUF,SEKRTY+2   ;SETUP FOR WRT RETRY ON SEEK ERROR
2817                                                                          ;(AFTER A RECAL, THE CONTENTS OF SECTOR 1,
2818                                                                          ;TRACK 1 ARE LOADED INTO THE SECTOR BUFFER,
2819                                                                          ;TO REWRITE THE CORRECT DATA THE PROGRAM
2820                                                                          ;MUST REFILL THE SECTOR BUFFER,
2821  007512  012737  016570  007542                  MOV #MWRITE,STYP1+2     ;PUT ADDR OF WRITE MESSAGE IN SEEK ER TYPEOUT 1
2822  007520  004737  007526                          JSR PC,SEEKER           ;RECORD SEEK ERROR
2823  007524  000741                                  BR WRTDONE              ;GO TO NEXT SECTOR CAN'T FIND THIS ONE
2824
2825  007526  104405                  SEEKER:         CKSWR
2826  007530  032777  020000  171460                  BIT #SW13,@SWR          ;CHECK DON'T PRINT ERROR SWITCH
2827  007536  001004                                  BNE SWHLT1
2828  007540  104400  016570          STYP1:          TYPE ,MWRITE            ;PRINT WRITE (READ) SEEK ERROR
2829  007544  004737  007606                          JSR PC,SEKTYP
2830  007550  104405                  SWHLT1:         CKSWR
2831  007552  005777  171440                          TST @SWR                ;TEST THE HALT ON ERROR SWITCH
2832  007556  100001                                  BPL CONT14
2833  007560  000000                  HLT7:           HALT                    ;HALT ON THE ERROR
2834  007562  004737  007706          CONT14:         JSR PC,HOME             ;RECALIBRATE ON SEEK ERRORS
2835  007566  104405                                  CKSWR
2836  007570  032777  004000  171420                  BIT #SW11,@SWR          ;CHECK THE LOOP ON ERROR SWITCH
2837  007576  001001                                  BNE SEKRTY              ;IF SET LOOP ON THE ERROR THROUGH SEEK RETRY,
2838  007600  000207                                  RTS PC
2839  007602  000137  007144          SEKRTY:         JMP FILLBUF             ;RETRY WRITE COMMAND (READ COMAND)
2840
2841  007606  104400  016555          SEKTYP:         TYPE ,MSEEK             ;TYPE SEEK ERROR
2842  007612  104400  016032                          TYPE ,MPRES             ;TYPE ADDRESS OF TRACK MOVED FROM
2843  007616  013746  013130                  MOV     PRESTRK,-(SP)   ;;SAVE PRESTRK FOR TYPEOUT
2844  007622  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
2845  007624     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
2846  007625     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
2847  007626  104400  016120                  TYPE ,MCRLF
2848  007632  000207                          RTS PC
2849
```



```
2850  007634  153737  012724  007702  COMMWORD:       BISB UNITSEL,FUNCTION   ;SET UNIT SELECTION BIT IN COMMAND WORD
2851  007642  013777  007702  171336                  MOV FUNCTION,@RXCS      ;SEND OUT COMMAND TO DRIVE
2852  007650  004737  006560          1$:             JSR PC,STR              ;WAIT FOR TR FLAG
2853  007654  000775                                  BR 1$
2854  007656  113777  013416  171324                  MOVB TSECTOR,@RXDB      ;SEND OUT TARGET SECTOR
2855  007664  004737  006560          2$:             JSR PC,STR              ;WAIT FOR TR FLAG
2856  007670  000775                                  BR 2$
2857  007672  113777  013126  171310                  MOVB TARGET,@RXDB       ;SEND OUT TARGET TRACK
2858  007700  000207                                  RTS PC
2859
2860  007702  000000                  FUNCTION:       0
2861  007704  000000                  DATACK:         0                       ;DATA CHECK ON CRC ERROR FLAG
2862
2863  007706  104405                  HOME:           CKSWR
2864  007710  032777  000400  171300                  BIT #SW8,@SWR           ;TEST NO RECAL SWITCH
2865  007716  001035                                  BNE RTN
2866  007720  012777  040001  171260                  MOV #RECAL,@RXCS        ;ISSUE RECAL FUNCTION
2867  007726  004737  006574          2$:             JSR PC,SDN
2868  007732  000775                                  BR 2$
2869  007734  005777  171246                          TST @RXCS               ;WAS THERE AN ERROR
2870  007740  100021                                  BPL XHOME               ;NO
2871  007742  104405                                  CKSWR
2872  007744  032777  020000  171244                  BIT #BIT13,@SWR         ;YES, SHOULD IT BE PRINTED
2873  007752  001002                                  BNE 1$                  ;NO
2874  007754  004737  012152                          JSR  PC,RDCODE
2875  007760  104405                  1$:             CKSWR
2876  007762  005777  171230                          TST @SWR                ;TEST HALT ON ERROR SWITCH
2877  007766  100001                                  BPL 3$
2878  007770  000000                                  HALT
2879  007772  032777  004000  171216  3$:             BIT #SW11,@SWR          ;TEST LOOP ON ERROR SWITCH
2880  010000  001342                                  BNE HOME
2881  010002  000207                                  RTS PC
2882  010004  012737  000001  013130  XHOME:          MOV #1,PRESTRK          ;SET THE PRESENT TRACK TO TRACK 1
2883  010012  000207                  RTN:            RTS PC
2884
```

```
                                        ;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  005462  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  016530  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  016530  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  104405                              CKSWR
010260  032777  020000  170730              BIT #SW13,@SWR          ;TEST DON'T PRINT SWITCH
010266  001004                              BNE 2$
010270  104400  016500                      TYPE ,MBADCRC           ;TYPE CRC GENERATOR ERROR
010274  104400  016120                      TYPE ,MCRLF
010300  104405                  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  104405                  DATACRC:      CKSWR
010332  032777  020000  170656              BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
010340  001004                              BNE 4$
010342  104400  016536                      TYPE ,MCRC              ;TYPE DATA CRC ERROR
010346  104400  016120                      TYPE ,MCRLF
010352  104405                  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
```

```
2970  010410  022737  006050  006556  DDCHK:    CMP #T25,PCSCOPE        ;IS THIS TEST 25
2971  010416  001041                    BNE CONT10
2972  010420  132737  000100  012146            BITB #BIT6,ASTAT        ;THIS IS TEST 25
2973  010426  001056                    BNE RETURN              ;DD BIT SHOULD BE SET
2974  010430  104405                    CKSWR
2975  010432  032777  020000  170556            BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
2976  010440  001013                    BNE CONT11
2977  010442  004737  010566                    JSR PC,ERMSG
2978  010446  104400  015747                    TYPE ,MDDMIS            ;TYPE MISSING DELETED DATA BIT
2979  010452  052737  000400  012724  DDERR:    BIS #BIT8,UNITSEL       ;SET HARD ERROR FLAG
2980  010460  004737  012050                    JSR PC,TYPADR           ;TYPE ADDRESS OF ERROR
2981  010464  104400  016120                    TYPE ,MCRLF
2982  010470  104405            CONT11:       CKSWR
2983  010472  005777  170520                    TST @SWR                ;TEST HALT ON ERROR SWITCH
2984  010476  100001                    BPL CONT17
2985  010500  000000            HLT13:        HALT                    ;HALT ON DELETED DATA ERROR
2986  010502  032777  004000  170506  CONT17:   BIT #SW11,@SWR          ;TEST LOOP ON ERROR
2987  010510  001402                    BEQ 4$
2988  010512  000137  010024                    JMP REREAD              ;LOOP ON TEST
2989  010516  000137  010132            4$:     JMP NEXTRD              ;READ NEXT SECTOR
2990  010522  032737  000100  012146  CONT10:   BIT #BIT6,ASTAT        ;THIS IS NOT A DELETED DATA TRANSFER
2991  010530  001415                    BEQ RETURN
2992  010532  052737  000400  012724            BIS #BIT8,UNITSEL       ;SET HARD ERROR FLAG
2993  010540  104405                    CKSWR
2994  010542  032777  020000  170446            BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
2995  010550  001347                    BNE CONT11
2996  010552  004737  010566                    JSR PC,ERMSG
2997  010556  104400  015721                    TYPE ,MUNXDD            ;TYPE UNEXPECTED DELETED DATA BIT
2998  010562  000733                    BR DDERR
2999  010564  000207            RETURN:       RTS PC
3000
3001
3002  010566  104400  016123  ERMSG:    TYPE ,MERHEADER
3003  010572  013746  006556            MOV     PCSCOPE,-(SP)   ;;SAVE PCSCOPE FOR TYPEOUT
3004  010576  104402                    TYPOS                   ;;GO TYPE--OCTAL ASCII
3005  010600     006                    .BYTE   6               ;;TYPE 6 DIGITS
3006  010601     000                    .BYTE   0               ;;SUPPRESS LEADING ZEROS
3007  010602  104400  006725                    TYPE ,MPASS
3008  010606  013737  002530  010620            MOV PASS,1$
3009  010614  004537  015600                    JSR R5,SGLDEC
3010  010620  000000            1$:           OPFN
3011  010622  104400  016120                    TYPE ,MCRLF
3012  010626  004737  006450                    JSR PC,DING
3013  010632  000207                    RTS PC
3014
3015
```



```
3016                                        .SBTTL READ AND VERIFY DATA
3017
3018                                        ;READ A SECTOR,EMPTY THE SECTOR BUFFER AND VERIFY
3019                                        ;THE DATA READ AGAINST CORE DATA BUFFER
3020
3021  010634  052701  100000  READCHK:  BIS #BIT15,R1           ;SET READ ONE SECTOR FLAG
3022  010640  004737  010014            JSR PC,READ             ;GO READ ONE SECTOR
3023  010644  005737  013414  EMPBUFF:  TST SECCNTR             ;IF CLEARED NO SECTORS WERE FOUND
3024  010650  001002                    BNE 1$
3025  010652  000137  011446            JMP EXIT                ;GO TO NEXT TRACK
3026  010656  005037  011452  1$:       CLR BYTECNTR            ;CLEAR THE BYTE AND ERROR COUNTERS
3027  010662  005037  011460            CLR ERCNTR
3028  010666  052701  000200            BIS #BIT7,R1            ;R1 BIT 7 IS USED AS FIRST ERROR FLAG
3029  010672  004737  010766            JSR PC,ADJSUM           ;ADJUST DATA AND CK SUM FOR ADDRESSES
3030  010676  005037  011056            CLR CKSUM               ;SET UP FOR CHECK SUM ACCUMULATION
3031  010702  012746  011306            MOV #EMPDONE,-(SP)      ;SET UP RETURN ADDRESSES
3032  010706  012746  011060            MOV #EMPER,-(SP)
3033  010712  005046            CLR     -(SP)                   ;LOWER 'CPU' LEVEL
3034  010714  012746  010722            MOV     #2$,-(SP)       ;SET RETURN 'PC'
3035  010720  000002            RTI                             ;GET 'CPU' LEVEL INTO 'PSW'
3036  010722  012777  000103  170256  2$:       MOV #EBIE,@RXCS         ;LOAD EMPTY BUFFER FUNCTION
3037  010730  105777  170252  EMPFLAG:  TSTB @RXCS              ;TEST FOR TR FLAG
3038  010734  100375                    BPL EMPFLAG
3039
3040  010736  117737  170246  011454  CKBYTE:   MOVB @RXDB,BADBYTE     ;SAVE BYTE FROM DISKETTE
3041  010744  063737  011454  011056            ADD BADBYTE,CKSUM       ;ACCUMULATE CHECK SUM
3042  010752  123720  011454                    CMPB BADBYTE,(R0)+      ;COMPARE AGAINST GOOD BYTE
3043  010756  001054                    BNE DATAER              ;IF NOT EQUAL GO TO DATAER
3044  010760  005237  011452                    INC BYTECNTR
3045  010764  000761                    BR EMPFLAG              ;GET NEXT BYTE
3046
3047  010766  113737  013126  017404  ADJSUM:   MOVB TARGET,BUFADR     ;SET FIRST AND SECOND BYTES WITH ADDRESSES
3048  010774  113737  013416  017405            MOVB TSECTOR,BUFADR+1
3049  011002  013737  012536  011056            MOV SUM,CKSUM           ;GET THE PATTERN SUM
3050  011010  063737  013126  011056            ADD TARGET,CKSUM        ;ADD TRACK ADDRESS TO CHECK SUM
3051  011016  063737  013416  011056            ADD TSECTOR,CKSUM       ;ADD SECTOR ADDRESS TO CHECK SUM
3052  011024  113737  011056  017602            MOVB CKSUM,BUFADR+176   ;INSERT CHECK SUM TO DATA BUFFER
3053  011032  106337  011056                    ASLB CKSUM              ;GENERATE NEGITIVE CHECK SUM
3054  011036  105437  011056                    NEGB CKSUM
3055  011042  113737  011056  017603            MOVB CKSUM,BUFADR+177   ;INSERT NEG,SUM INTO DATA BUFFER
3056  011050  012700  017404                    MOV #BUFADR,R0          ;SET ADDRESS OF BYTE IN R0
3057  011054  000207                    RTS PC                  ;RETURN
3058
3059  011056  000000            CKSUM:    0
3060
3061  011060  005726            EMPER:    TST (SP)+               ;REMOVE THE DONE RETURN FROM THE STACK
3062  011062  012737  016402  007266            MOV #MEMPTY,PTYP1+2     ;PUT ADDR OF EMPTYBUF MESSAGE IN PAR ER TYPOUT 1
3063  011070  012737  010644  007324            MOV #EMPBUFF,PLOOP+2    ;RETURN THROUGH HERE TO LOOP ON ERROR
3064  011076  012737  011430  007330            MOV #NXREAD,PCONT+2     ;IF NO LOOP ON ERROR GO TO NEXT SECTOR
3065  011104  000137  007252                    JMP PARTEST             ;REPORT PARITY ERROR
3066
```

```
3067  011110  052737  000400  012724  DATAER:   BIS #BIT8,UNITSEL     ;SET THE HAD ERROR FLAG
3068  011116  005237  011460            INC ERCNTR              ;INC THE BYTE ERROR COUNTER
3069  011122  104405                    CKSWR
3070  011124  032777  020000  170064    BIT #SW13,@SWR          ;TEST PRINT ERROR SW IN SWR
3071  011132  001054                    BNE NOERTYP             ;DON'T PRINT THE ERROR
3072  011134  032777  001000  170054    BIT #SW9,@SWR           ;TEST PRINT 10 ERRORS SWITCH
3073  011142  001004                    BNE 1$                  ;IF SET PRINT ALL ERRORS
3074  011144  023727  011460  000012    CMP ERCNTR,#10.         ;HAVE 10 ERRORS BEEN TYPED
3075  011152  003044                    BGT NOERTYP             ;YES,DON'T PRINT ANY MORE
3076  011154  105701            1$:     TSTB R1                 ;TEST FIRST ERROR FLAG
3077  011156  100014                    BPL TYPERR
3078  011160  004737  010566            JSR PC,ERMSG            ;PRINT ADDRESS OF TEST
3079  011164  104400  015772            TYPE ,MDERHDR           ;FIRST ERROR, PRINT ERROR HEADER
3080  011170  104400  016120            TYPE ,MCRLF
3081  011174  004737  012050            JSR PC,TYPADR           ;PRINT TRACK AND SECTOR LOCATIONS
3082  011200  104400  016061            TYPE ,MCOLMUN           ;SET UP COLMUN HEADINGS
3083  011204  042701  000200            BIC #BIT7,R1            ;CLEAR FIRST ERROR FLAG
3084  011210  013737  011452  011222  TYPERR:   MOV BYTECNTR,1$       ;PRINT BYTE NUMBER
3085  011216  004537  015600            JSR R5,SGLDEC
3086  011222  000000            1$:     OPEN
3087  011224  104400  016115            TYPE ,DBLSP
3088  011230  013746  011454            MOV BADBYTE,-(SP)       ;PRINT BYTE READ FROM DISKETTE
3089  011234  104402                    TYPOS
3090  011236  000003                    .WORD 3
3091  011240  104400  016115            TYPE ,DBLSP
3092  011244  114037  011456            MOVB -(R0),GOODBYTE     ;GET GOOD BYTE
3093  011250  005200                    INC R0                  ;RETURN R0 TO NEXT BYTE IN BUFFER
3094  011252  013746  011456            MOV GOODBYTE,-(SP)
3095  011256  104403                    TYPON                   ;PRINT GOOD DATA
3096  011260  104400  016120            TYPE ,MCRLF
3097  011264  104405            NOERTYP: CKSWR
3098  011266  005777  167724            TST @SWR                ;TEST HALT ON ERROR SWITCH
3099  011272  100001                    BPL CONT20
3100  011274  000000            HLT14:  HALT
3101  011276  005237  011452    CONT20: INC BYTECNTR
3102  011302  000137  010730            JMP EMPFLAG
3103
```



```
3104  011306  005737  007704    EMPDONE:  TST DATACK              ;WAS THIS READ CHECK CAUSED BY A CRC ERROR
3105  011312  001401                      BEQ 1$                  ;NO
3106  011314  000207                      RTS PC                  ;YES,RETURN TO CRC HANDLER
3107  011316  005737  011460      1$:     TST ERCNTR              ;WAS THERE ERRORS
3108  011322  001442                      BEQ NXREAD              ;NO ERRORS
3109  011324  104405                      CKSWR
3110  011326  032777  020000  167662      BIT #SW13,@SWR          ;YES,TEST DON'T PRINT SWITCH
3111  011334  001024                      BNE 2$                  ;DON'T PRINT THE ERROR
3112  011336  104400  016333              TYPE ,MERCT             ;PRINT THE TOTAL DATA ERROR COUNT
3113  011342  013737  011460  011354      MOV ERCNTR,3$
3114  011350  004537  015600              JSR R5,SGLDEC
3115  011354  000000            3$:       OPEN
3116  011356  104400  016650              TYPE ,MSUM              ;INDICATE IF CHECK SUM WAS GOOD OR HAD ERRORS
3117  011362  105737  011056              TSTB CKSUM
3118  011366  001403                      BEQ 4$
3119  011370  104400  016635              TYPE ,MBAD
3120  011374  000402                      BR 5$
3121  011376  104400  016643      4$:     TYPE,MGOOD
3122  011402  104400  016120      5$:     TYPE ,MCRLF
3123  011406  104405            2$:     CKSWR
3124  011410  032777  004000  167600      BIT #SW11,@SWR          ;TEST LOOP ON ERROR SWITCH
3125  011416  001404                      BEQ NXREAD              ;IF NOT SET GO TO NEXT SECTOR
3126  011420  004737  010024              JSR PC,REREAD           ;YES,GO REREAD THE DATA
3127  011424  000137  010644              JMP EMPBUFF             ;GO RECHECK THE DATA
3128  011430  005337  013414    NXREAD:   DEC SECCNTR
3129  011434  001404                      BEQ EXIT
3130  011436  004737  010020              JSR PC,XREAD            ;READ THE NEXT SECTOR
3131  011442  000137  010644              JMP EMPBUFF
3132  011446  005001            EXIT:     CLR R1                  ;CLEAR THE ONE READ FLAG
3133  011450  000207                      RTS PC
3134
3135  011452  000000            BYTECNTR: 0
3136  011454  000000            BADBYTE:  0
3137  011456  000000            GOODBYTE: 0
3138  011460  000000            ERCNTR:   0
3139
3140                            ;;*************************************************************
3141
3142                                    ;AN INTERRUPT DID NOT OCCURE AT A FUNCTION DONE FLAG.
3143
3144  011462  104405            NOINTER:  CKSWR
3145  011464  032777  020000  167524      BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
3146  011472  001006                      BNE 1$
3147  011474  004737  010566              JSR PC,ERMSG
3148  011500  104400  016256              TYPE ,MINTER            ;TYPE NO INTERRUPT ON DONE ERROR
3149  011504  104400  016120              TYPE ,MCRLF
3150  011510  104405            1$:       CKSWR
3151  011512  005777  167500              TST @SWR                ;TEST HALT ON ERROR SWITCH
3152  011516  100001                      BPL CONT21
3153  011520  000000            HLT15:    HALT                    ;HALT ON ERROR
3154  011522  004737  011526    CONT21:   JSR PC,INTSERV          ;JSR TO INTSERV AS IF IT WAS AN INTERRUPT
3155
```

```
                                        ,SBTTL  INTERRUPT SERVICE
                                
011526  117737  167456  012146  INTSERV:      MOVB @RXDB,ASTAT        ;SAVE THE ERROR AND STATUS WORD
011534  005777  167446                        TST @RXCS               ;TEST THE ERROR FLAG
011540  100444                                BMI RXERROR             ;THERE WAS AN ERROR GO REPORT IT
011542  032737  000004  012146                BIT #BIT2,ASTAT         ;IS INIT DONE SET
011550  001402                                BEQ 2$                  ;NO,CONTINUE
011552  000137  012006                        JMP RXPWR               ;YES,REPORT POWER FAILED AND RESTART
011556  032737  000003  012146  2$:           BIT #3,ASTAT            ;ARE PAR OR CRC BITS SET
011564  001021                                BNE 1$                  ;YES GO REPORT ERROR
011566  132777  000040  167412                BITB #DONEBIT,@RXCS     ;IS DONE SET
011574  001012                                BNE 3$                  ;IF SET RETURN TO TEST
011576  104405                                CKSWR
011600  032777  020000  167410                BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
011606  001004                                BNE 4$                  ;DON'T PRINT
011610  104400  016311                        TYPE ,MUKNINT           ;TYPE UNKNOWN INTERRUPT
011614  104400  016120                        TYPE ,MCRLF
011620  000002                  4$:           RTI                     ;RETURN FROM THE INTERRUPT
011622  062706  000006          3$:           ADD #6,SP               ;BYPASS INTERRUPT POINTERS ON STACK
011626  000207                                RTS PC                  ;RETURN TO PROGRAM
011630  104405                  1$:           CKSWR
011632  032777  020000  167356                BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
011640  001004                                BNE RXERROR
011642  104400  016614                        TYPE ,MNOFLAG           ;TYPE NO STATUS ERROR ERROR
011646  104400  016120                        TYPE ,MCRLF
011652  005237  006472          RXERROR:      INC ERRORS              ;AN ERROR INDICATOR
011656  001775                                BEQ RXERROR
011660  052737  000400  012724                BIS #BIT8,UNITSEL       ;SET HARD ERROR FLAG
011666  012777  000017  167312  2$:           MOV #RDER,@RXCS         ;GET THE ERROR CODE
011674  004737  006574          3$:           JSR PC,SDN              ;TEST FOR DONE FLAG
011700  000775                                BR 3$
011702  032777  000002  167300                BIT #2,@RXDB            ;WAS THERE A PARITY ERROR
011710  001403                                BEQ 1$                  ;NO,CONTINUE
011712  004737  012022                        JSR PC,PARTYP           ;YES,GO REPORT THE PARITY ERROR
011716  000763                                BR 2$                   ;REISSUE THE FUNCTION
011720  117737  167264  012150  1$:           MOVB @RXDB,BSTAT        ;SAVE THE ERROR CODE IN B STATUS
011726  104405                  NOPRNT:       CKSWR
011730  032777  020000  167260                BIT #SW13,@SWR          ;TEST PRINT ERROR SWITCH IN SWR
011736  001020                                BNE 2$
011740  104400  016120                        TYPE ,MCRLF
011744  004737  010566                        JSR PC,ERMSG            ;TYPE ERROR AND MESSAGES
011750  104400  016212                        TYPE ,MRXCS             ;TYPE COMMAND STATUS REGISTER
011754  013746  007702                MOV     FUNCTION,-(SP)  ;;SAVE FUNCTION FOR TYPEOUT
011760  104402                        TYPOS                   ;;GO TYPE--OCTAL ASCII
011762      006                       .BYTE   6               ;;TYPE 6 DIGIT(S)
011763      000                       .BYTE   0               ;;SUPPRESS LEADING ZEROS
011764  004737  012050                        JSR PC,TYPADR           ;TYPE ADDRESSES AND RUN CONDITIONS
011770  104400  016120                        TYPE ,MCRLF
011774  004737  012220                        JSR PC,TYPCODE          ;PRINT THE STATUS REGISTERS
012000  062706  000004          2$:           ADD #4,SP               ;MOVE ERROR RETURN TO TOP OF STACK
012004  000207                                RTS PC

012006  104400  016665          RXPWR:        TYPE ,MRX11             ;ONLY THE RX11 POWER HAS FAILED
012012  104400  015270                        TYPE ,SPOWER            ;PRINT POWER FAILED
012016  000137  001350                        JMP RESTART             ;GO TO RESTART
```



```
012022  104400  016212  PARTYP:         TYPE ,MRXCS
012026  017746  167154                MOV     @RXCS,-(SP)     ;;SAVE @RXCS FOR TYPEOUT
012032  104402                        TYPOS                   ;;GO TYPE--OCTAL ASCII
012034      006                       .BYTE   6               ;;TYPE 6 DIGIT(S)
012035      000                       .BYTE   0               ;;SUPPRESS LEADING ZEROS
012036  104400  016577                TYPE ,MPAR
012042  104400  016120                TYPE ,MCRLF
012046  000207                        RTS PC

012050  104400  016020  TYPADR:       TYPE ,MTRK              ;TYPE TRACK ADDRESS
012054  013746  013126                MOV     TARGET,-(SP)    ;;SAVE TARGET FOR TYPEOUT
012060  104402                        TYPOS                   ;;GO TYPE--OCTAL ASCII
012062      003                       .BYTE   3               ;;TYPE 3 DIGIT(S)
012063      000                       .BYTE   0               ;;SUPPRESS LEADING ZEROS
012064  104400  016047                TYPE ,MSECT             ;TYPE SECTOR ADDRESS
012070  013737  013416  012144        MOV TSECTOR,2$
012076  042737  177740  012144        BIC #177740,2$          ;CLEAR ALL BUT SECTOR ADDRESS
012104  013746  012144                MOV     2$,-(SP)        ;;SAVE 2$ FOR TYPEOUT
012110  104402                        TYPOS                   ;;GO TYPE--OCTAL ASCII
012112      002                       .BYTE   2               ;;TYPE 2 DIGIT(S)
012113      000                       .BYTE   0               ;;SUPPRESS LEADING ZEROS
012114  104400  016115                TYPE ,DBLSP
012120  032737  000020  012724        BIT #BIT4,UNITSEL       ;WHITCH DRIVE IS BEING USED
012126  001003                        BNE 1$
012130  104400  016163                TYPE ,MUNIT0            ;TYPE UNIT 0
012134  000402                        BR 4$
012136  104400  016173  1$:           TYPE ,MUNIT1            ;TYPE UNIT 1
012142  000207          4$:           RTS PC
012144  000000          2$:           OPEN

012146  000000  ASTAT:          0
012150  000000  BSTAT:          0

012152  117737  167032  012146  RDCODE:         MOVB @RXDB,ASTAT        ;SAVE THE A STATUS
012160  012777  000017  167020  2$:             MOV #RDER,@RXCS         ;READ THE B STATUS REGISTER
012166  004737  006574          3$:             JSR PC,SDN              ;WAIT FOR DONE FLAG
012172  000775                                  BR 3$
012174  032777  000002  167006                  BIT #2,@RXDB            ;WAS THERE A PARITY ERROR
012202  001403                                  BEQ 1$                  ;NO,CONTINUE
012204  004737  012022                          JSR PC,PARTYP           ;YES,REPORT THE PARITY ERROR
012210  000763                                  BR 2$                   ;RETRY READING STATUS B
012212  117737  166772  012150  1$:             MOVB @RXDB,BSTAT        ;SAVE THE B STATUS CODES
012220  104400  016222  TYPCODE:        TYPE ,MASTAT            ;TYPE THE CONTENTS OF THE TWO STATUS REGISTERS
012224  013746  012146                  MOV     ASTAT,-(SP)     ;;SAVE ASTAT FOR TYPEOUT
012230  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
012232      003                         .BYTE   3               ;;TYPE 3 DIGIT(S)
012233      000                         .BYTE   0               ;;SUPPRESS LEADING ZEROS
012234  104400  016106                  TYPE ,TAB
012240  104400  016236                  TYPE ,MBSTAT
012244  013746  012150                  MOV BSTAT,-(SP)
012250  104403                          TYPON
012252  104400  016120                  TYPE ,MCRLF
012256  000207                          RTS PC
```

```
                                        .SBTTL  PATTERN GENERATOR

                                        ;NOTE: ALL DATA PATTERNS WILL BE MODIFIED SO THE FIRST BYTE WILL
                                        ;CONTAIN THE TRACK ADDRESS, THE SECOND BYTE WILL CONTAIN THE UNIT
                                        ;NUMBER AND SECTOR ADDRESS IN WHICH THE DATA IS WRITTEN, THE MOST
                                        ;SIGNIFICANT BIT OF THIS SECOND BYTE INDICATES THE UNIT, UNIT 0
                                        ;IF "0" UNIT 1 IF "1", THE LAST TWO BYTES CONTAIN THE CHECK SUM.
                                ;;*****************************************************************

012260  012704  017404          GETPATTERN:   MOV #BUFADR,R4          ;SET ADDRESS OF FIRST DATA BYTE
012264  005037  012536                        CLR SUM                 ;SET UP FOR ACCUMULATION OF CHECK SUM
012270  013705  012324                        MOV PAT,R5              ;GET PATTERN BITS
012274  006305                                ASL R5
012276  000175  012302                        JMP @PATTERNS(R5)
012302  012326                  PATTERNS:     DATA0                   ;000 DATA BYTE
012304  012340                                DATA1                   ;377 DATA BYTE
012306  012350                                FLOAT0                  ;FLOAT A 0 THROUGH ALL 1'S
012310  012412                                FLOAT1                  ;FLOAT A 1 THROUGH ALL 0'S
012312  012420                                PAT125                  ;125/052 DATA WORD
012314  012440                                PAT314                  ;314/063 DATA WORD
012316  012450                                COUNT                   ;INCRUMENT DATA PATTERN
012320  012470                                RANDATA                 ;RANDOM DATA BYTE

012322  000000                  DATABYTE:     0
012324  000000                  PAT:          0


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

                                        ;LOAD SOFTWARE BUFFER WITH ALL ZEROS
                                        ;   PAT = 0

012326  005037  012322          DATA0:        CLR DATABYTE
012332  004737  012510          PATGEN:       JSR PC,LOAD             ;GO LOAD THE DATA BUFFER
012336  000775                                BR PATGEN

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

                                        ;LOAD SOFTWARE BUFFER WITH ALL ONES
                                        ;   PAT = 1

012340  112737  000377  012322  DATA1:        MOVB #377,DATABYTE
012346  000771                                BR PATGEN
```



```
                                        ;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$:   INC DATABYTE
012462  004737  012510                        JSR PC,LOAD
012466  000773                                BR 1$
```

```
3366                                    ;;*************************************************************
3367
3368                                            ;LOAD SOFTWARE BUFFER WITH RANDOM DATA PATTERN
3369                                            ;   PAT = 7
3370
3371  012470  004737  012540          RANDATA:        JSR PC,RANGEN           ;GET RANDOM NUMBER
3372  012474  113737  012632  012322                  MOVB RANUM,DATABYTE
3373  012502  004737  012510                          JSR PC,LOAD
3374  012506  000770                                  BR RANDATA
3375
3376  012510  063737  012322  012536  LOAD:           ADD DATABYTE,SUM        ;ACCUMULATE THE PATTERN CHECK SUM
3377  012516  113724  012322                          MOVB DATABYTE,(R4)+     ;LOAD THE DATA BUFFER
3378  012522  022704  017604                          CMP #BUFADR+200,R4      ;HAVE 128 BYTES BEEN GENERATED
3379  012526  001401                                  BEQ 1$                  ;IF YES,RETURN TO TEST
3380  012530  000207                                  RTS PC                  ;IF NO,RETURN TO PATTERN GENERATOR
3381  012532  005726                  1$:             TST (SP)+               ;TAKE PATTERN RETURN ADDRESS OF STACK
3382  012534  000207                                  RTS PC                  ;RETURN TO TEST
3383
3384  012536  000000                  SUM:            0
3385
3386  012540  012700  000001          RANGEN:         MOV #1,R0
3387  012544  063700  012626                          ADD RAN1,R0
3388  012550  063700  012630                          ADD RAN2,R0
3389  012554  042700  170000                          BIC #170000,R0
3390  012560  000241                                  CLC
3391  012562  006100                                  ROL R0
3392  012564  006100                                  ROL R0
3393  012566  010037  012626                          MOV R0,RAN1
3394  012572  005000                                  CLR R0
3395  012574  013700  012630                          MOV RAN2,R0
3396  012600  006000                                  ROR R0
3397  012602  006000                                  ROR R0
3398  012604  063700  012626                          ADD RAN1,R0
3399  012610  042700  170000                          BIC #170000,R0
3400  012614  010037  012630                          MOV R0,RAN2
3401  012620  010037  012632                          MOV R0,RANUM
3402  012624  000207                                  RTS PC
3403
3404  012626  001234                  RAN1:           001234
3405  012630  000765                  RAN2:           000765
3406  012632  000000                  RANUM:          0
3407
```



```
3408                                            .SBTTL  UNIT SELECTION
3409
3410                                            ;TEST FOR SELECTED UNITS,DRIVE READY,AND USED CONDITIONS
3411                                            ;ALSO CONTAINS A "HAD ERROR" FLAG TO BE TESTED AT EOP.
3412                                            ;THE BITS IN UNITSEL ARE USED AS FOLLOWS
3413                                            ;
3414                                            ;BIT15  =UNIT 1 SELECTED VIA SWR
3415                                            ;BIT14  =UNIT 1 USED BIT
3416                                            ;BIT8   =THIS PASS HAD AN ERROR
3417                                            ;BIT7   =UNIT 0 SELECTED VIA SWR
3418                                            ;BIT6   =UNIT 0 USED BIT
3419                                            ;BIT4   =UNIT SELECTION FOR FUNCTION WORD
3420
3421                                    ;;*************************************************************
3422
3423  012634  032737  000100  012724  GETUNIT:        BIT #BIT6,UNITSEL       ;WAS UNIT 0 JUST USED
3424  012642  001012                                  BNE 1$                  ;UNIT 0 USED CHECK UNIT 1
3425  012644  105737  012724                          TSTB UNITSEL            ;WAS UNIT 0 SELECTED
3426  012650  100007                                  BPL 1$                  ;NO GO TO UNIT 1
3427  012652  042737  040020  012724                  BIC #40020,UNITSEL      ;CLEAR UNIT 1 USED BIT AND FUNCTION UNIT BIT
3428  012660  052737  000100  012724                  BIS #BIT6,UNITSEL       ;SET UNIT 0 USED BIT
3429  012666  000207                                  RTS PC
3430  012670  005737  012724          1$:             TST UNITSEL             ;WAS UNIT 1 SELECTED
3431  012674  100012                                  BPL 2$                  ;NO RETURN
3432  012676  032737  040000  012724                  BIT #BIT14,UNITSEL      ;HAS UNIT 1 BEEN USED
3433  012704  001006                                  BNE 2$                  ;YES RETURN
3434  012706  042737  000100  012724                  BIC #BIT6,UNITSEL       ;CLEAR UNIT 0 USED BIT
3435  012714  052737  040020  012724                  BIS #40020,UNITSEL      ;SET UNIT 1 USED BIT AND FUNCTION UNIT BIT
3436  012722  000207                  2$:             RTS PC
3437
3438
3439
3440  012724  000000                  UNITSEL:        0
3441
3442                                            ;TEST THAT ALL UNITS HAVE BEEN ACCESSED
3443
3444  012726  005737  012724          DONE:           TST UNITSEL             ;IS UNIT 1 SELECTED
3445  012732  100006                                  BPL 1$                  ;NO RETURN
3446  012734  032737  040000  012724                  BIT #BIT14,UNITSEL      ;YES HAS IT BEEN USED
3447  012742  001002                                  BNE 1$                  ;YES RETURN
3448  012744  062716  000002                          ADD #2,@SP              ;BYPASS NOT DONE RETURE ON STACK
3449  012750  000207                  1$:             RTS PC
3450
```

```
                                        .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,TPKCNTR         ;SET TRACK COUNTER
013072  000770                                BR 1$
013074  012737  000004  013124  3$:           MOV #4,TPKCNTR          ;SET THE TRACK COUNTER
013102  000764                                BR 1$


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

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
```



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

                                        ;LIMITED SEQUENCE, ACCESS TRACKS 52 TO 53 TO 114 BACK TO 0
013140  005737  001200          LIMTRK:       TST OD                  ;TEST HIGH ORDER FIRST TIME BIT
013144  100007                                BPL 1$                  ;NOT SET, ON TRACK 52
013146  012737  000052  013126                MOV #52,TARGET          ;GO TO TRACK 52
013154  042737  100000  001200                BIC #BIT15,OD           ;CLEAR FIRST TIME BIT
013162  000207                                RTS PC
013164  105737  001200          1$:           TSTB OD                 ;TEST LOW ORDER FIRST TIME BIT
013170  100007                                BPL 2$                  ;NOT SET, ON TRACK 53
013172  012737  000053  013126                MOV #53,TARGET          ;GO TO TRACK 53
013200  042737  000200  001200                BIC #BIT7,OD
013206  000207                                RTS PC
013210  023727  013126  000114  2$:           CMP TARGET,#114         ;IS IT ON TRACK 114
013216  001404                                BEQ 3$                  ;YES,GO TO TRACK 0
013220  012737  000114  013126                MOV #114,TARGET         ;NO, GO TO TRACK 114
013226  000207                                RTS PC
013230  005037  013126          3$:           CLR TARGET              ;GO TO TRACK 0
013234  000207                                RTS PC

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

                                        ;INCREMENT FROM OD+1 TO ID AND RETURN TO OD
                                        ; USED WHEN TRACK LIMITS ARE SELECTED

013236  042737  100200  001200  SEQ1:          BIC #100200,OD          ;CLEAR FIRST TIME BITS
013244  123737  013134  013130                CMPB XID,PRESTRK        ;PRESENT TRACK EQUAL TO ID
013252  001004                                BNE 1$                  ;NO GET NEW TRACK
013254  113737  001200  013126                MOVB OD,TARGET          ;YES RETURN TO OD
013262  000207                                RTS PC
013264  005237  013126          1$:           INC TARGET              ;ADD 1 TO TARGET TRACK
013270  000207                                RTS PC

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

                                        ;DECREMENT FROM ID = 12 TO OD = 0
                                        ;USED IN TEST 26 ONLY

013272  005737  001200  SEQ2:          TST OD                  ;FIRST TIME BIT SET
013276  100007                                BPL 1$                  ;NO GET NEXT TRACK
013300  042737  100200  001200                BIC #100200,OD          ;YES CLEAR FIRST TIME BITS
013306  012737  000012  013126                MOV #12,TARGET          ;MOVE OUT 10 TRACKS
013314  000207                                RTS PC
013316  005337  013126          1$:           DEC TARGET              ;MOVE TO NEXT TRACK
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  104400                 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
```

```
 3632  013654  000724                          BR      2$              ;;GET NEXT CHARACTER
 3633  013656  105777  000044          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
 3634  013662  100375                          BPL     $TYPEC
 3635  013664  116677  000002  000036        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
 3636  013672  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
 3637  013700  001003                          BNE     1$              ;;BRANCH IF NO
 3638  013702  105037  013722                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
 3639  013706  000406                          BR      $TYPEX          ;;EXIT
 3640  013710  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
 3641  013716  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
 3642  013720  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
 3643  013722  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
 3644  013724  000207          $TYPEX: RTS     PC
 3645
 3646  013726  177564          $TPS:   .WORD   177564          ;;TTY PRINTER STATUS REG. ADDRESS
 3647  013730  177566          $TPB:   .WORD   177566          ;;TTY PRINTER BUFFER REG. ADDRESS
 3648  013732      000         $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
 3649  013733      002         $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
 3650  013734      012         $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
 3651  013735      000         $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
 3652  013736      077         $QUES:  .ASCII  "?"             ;;QUESTION MARK
 3653  013737      015         $CRLF:  .ASCII  <15>            ;;CARRAIGE RETURN
 3654  013740  000012          $LF:    .ASCIZ  <12>            ;;LINEFEED
```



```
 3655                                   .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
 3656
 3657                                   ;;*************************************************************
 3658                                   ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
 3659                                   ;*OCTAL (ASCII) NUMBER AND TYPE IT.
 3660                                   ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
 3661                                   ;*CALL:
 3662                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 3663                                   ;*      TYPOS                   ;;CALL FOR TYPEOUT
 3664                                   ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
 3665                                   ;*      .BYTE   M               ;;M=1 OR 0
 3666                                   ;*                                      ;;1=TYPE LEADING ZEROS
 3667                                   ;*                                      ;;0=SUPPRESS LEADING ZEROS
 3668                                   ;*
 3669                                   ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
 3670                                   ;*$TYPOS OR $TYPOC
 3671                                   ;*CALL:
 3672                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 3673                                   ;*      TYPON                   ;;CALL FOR TYPEOUT
 3674                                   ;*
 3675                                   ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
 3676                                   ;*CALL:
 3677                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 3678                                   ;*      TYPOC                   ;;CALL FOR TYPEOUT
 3679
 3680  013742  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
 3681  013746  116637  000001  014165        MOVB    1(SP),$0FILL    ;;LOAD ZERO FILL SWITCH
 3682  013754  112637  014167                MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
 3683  013760  062716  000002                ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
 3684  013764  000406                        BR      $TYPON
 3685  013766  112737  000001  014165  $TYPOC: MOVB    #1,$0FILL       ;;SET THE ZERO FILL SWITCH
 3686  013774  112737  000006  014167        MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
 3687  014002  112737  000005  014164  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
 3688  014010  010346                        MOV     R3,-(SP)        ;;SAVE R3
 3689  014012  010446                        MOV     R4,-(SP)        ;;SAVE R4
 3690  014014  010546                        MOV     R5,-(SP)        ;;SAVE R5
 3691  014016  113704  014167                MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
 3692  014022  005404                        NEG     R4
 3693  014024  062704  000006                ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
 3694  014030  110437  014166                MOVB    R4,$OMODE       ;;SAVE IT FOR USE
 3695  014034  113704  014165                MOVB    $0FILL,R4       ;;GET THE ZERO FILL SWITCH
 3696  014040  016605  000012                MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
 3697  014044  005003                        CLR     R3              ;;CLEAR THE OUTPUT WORD
 3698  014046  006105          1$:     ROL     R5              ;;ROTATE MSB INTO "C"
 3699  014050  000404                        BR      3$              ;;GO DO MSB
 3700  014052  006105          2$:     ROL     R5              ;;FORM THIS DIGIT
 3701  014054  006105                        ROL     R5
 3702  014056  006105                        ROL     R5
 3703  014060  010503                        MOV     R5,R3
 3704  014062  006103          3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
 3705  014064  105337  014166                DECB    $OMODE          ;;TYPE THIS DIGIT?
 3706  014070  100016                        BPL     7$              ;;BR IF NO
 3707  014072  042703  177770                BIC     #177770,R3      ;;GET RID OF JUNK
 3708  014076  001002                        BNE     4$              ;;TEST FOR 0
 3709  014100  005704                        TST     R4              ;;SUPPRESS THIS 0?
 3710  014102  001403                        BEQ     5$              ;;BR IF YES
```

```
3711  014104  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
3712  014106  052703  000060                   BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
3713  014112  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
3714  014116  110337  014162                  MOVB    R3,8$           ;;SAVE FOR TYPING
3715  014122  104400  014162                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
3716  014126  105337  014164          7$:     DECB    $OCNT           ;;COUNT BY 1
3717  014132  003347                  BGT     2$              ;;BR IF MORE TO DO
3718  014134  002402                  BLT     6$              ;;BR IF DONE
3719  014136  005204                  INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
3720  014140  000744                  BR      2$              ;;GO DO THE LAST DIGIT
3721  014142  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
3722  014144  012604                  MOV     (SP)+,R4        ;;RESTORE R4
3723  014146  012603                  MOV     (SP)+,R3        ;;RESTORE R3
3724  014150  016666  000002  000004         MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
3725  014156  012616                  MOV     (SP)+,(SP)
3726  014160  000002                  RTI                     ;;RETURN
3727  014162     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
3728  014163     000                  .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
3729  014164     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
3730  014165     000                  $0FILL: .BYTE   0               ;;ZERO FILL SWITCH
3731  014166  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```



```
3732                                    .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
3733
3734                                    ;;**************************************************************
3735                                    ;*SAVE R0-R5
3736                                    ;*CALL:
3737                                    ;*      SAVREG
3738                                    ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
3739                                    ;*
3740                                    ;*TOP---(+16)
3741                                    ;* +2---(+18)
3742                                    ;* +4---R5
3743                                    ;* +6---R4
3744                                    ;* +8---R3
3745                                    ;*+10---R2
3746                                    ;*+12---R1
3747                                    ;*+14---R0
3748
3749  014170                            $SAVREG:
3750  014170  010046                            MOV     R0,-(SP)        ;;PUSH R0 ON STACK
3751  014172  010146                            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
3752  014174  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
3753  014176  010346                            MOV     R3,-(SP)        ;;PUSH R3 ON STACK
3754  014200  010446                            MOV     R4,-(SP)        ;;PUSH R4 ON STACK
3755  014202  010546                            MOV     R5,-(SP)        ;;PUSH R5 ON STACK
3756  014204  016646  000022                    MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
3757  014210  016646  000022                    MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
3758  014214  016646  000022                    MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
3759  014220  016646  000022                    MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
3760  014224  000002                            RTI
3761
3762                                    ;*RESTORE R0-R5
3763                                    ;*CALL:
3764                                    ;*      RESREG
3765  014226                            $RESREG:
3766  014226  012666  000022                    MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
3767  014232  012666  000022                    MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
3768  014236  012666  000022                    MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
3769  014242  012666  000022                    MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
3770  014246  012605                            MOV     (SP)+,R5        ;;POP STACK INTO R5
3771  014250  012604                            MOV     (SP)+,R4        ;;POP STACK INTO R4
3772  014252  012603                            MOV     (SP)+,R3        ;;POP STACK INTO R3
3773  014254  012602                            MOV     (SP)+,R2        ;;POP STACK INTO R2
3774  014256  012601                            MOV     (SP)+,R1        ;;POP STACK INTO R1
3775  014260  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
3776  014262  000002                            RTI
```

```
                                        .SBTTL  TTY INPUT ROUTINE


                                        ;;*************************************************************
014264  177560                          STKS:   .WORD   177560          ;;TTY KBD STATUS
014266  177562                          STKB:   .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  104400  015015                          TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
014340  104400  015022                  $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
014344  013746  000176                          MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
014350  104401                                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
014352  104400  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  177672                          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  104400  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$:    ADD     #6,SP           ;;CLEAR UP STACK
014446  104400  013737                  14$:    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
```



```
014472  004737  013656                  16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
014476  021627  000060                          CMP     (SP),#60        ;;CHAR < 0?
014502  002420                                  BLT     18$             ;;BRANCH IF YES
014504  021627  000067                          CMP     (SP),#67        ;;CHAR > 7?
014510  003015                                  BGT     18$             ;;BRANCH IF YES
014512  042726  000060                          BIC     #60,(SP)+       ;;STRIP-OFF ASCII
014516  005766  000002                          TST     2(SP)           ;;IS THIS THE FIRST CHAR
014522  001403                                  BEQ     17$             ;;BRANCH IF YES
014524  006316                                  ASL     (SP)            ;;NO, SHIFT PRESENT
014526  006316                                  ASL     (SP)            ;;   CHAR OVER TO MAKE
014530  006316                                  ASL     (SP)            ;;   ROOM FOR NEW ONE.
014532  005266  000002                  17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
014536  056616  177776                          BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
014542  000707                                  BR      7$              ;;GET THE NEXT ONE
014544  104400  013736                  18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
014550  000720                                  BR      20$             ;;SIMULATE CONTROL-U
                                        .DSABL  LSB



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

                                        ;

014552  011646                          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
014554  016666  000004  000002                  MOV     4(SP),2(SP)     ;;SAVE THE PS
014562  105777  177476                  1$:     TSTB    @$TKS           ;;WAIT FOR
014566  100375                                  BPL     1$              ;;A CHARACTER
014570  117766  177472  000004                  MOVB    @$TKB,4(SP)     ;;READ THE TTY
014576  042766  177600  000004                  BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
014604  026627  000004  000023                  CMP     4(SP),#23       ;;IS IT A CONTROL-S?
014612  001013                                  BNE     3$              ;;BRANCH IF NO
014614  105777  177444                  2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
014620  100375                                  BPL     2$              ;;LOOP UNTIL ITS THERE
014622  117746  177440                          MOVB    @$TKB,-(SP)     ;;GET CHARACTER
014626  042716  177600                          BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
014632  022627  000021                          CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
014636  001366                                  BNE     2$              ;;IF NOT DISCARD IT
014640  000750                                  BR      1$              ;;YES, RESUME
014642  026627  000004  000140          3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
014650  002407                                  BLT     4$              ;;BRANCH IF YES
014652  026627  000004  000175                  CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
014660  003003                                  BGT     4$              ;;BRANCH IF YES
014662  042766  000040  000004                  BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
014670  000002                          4$:     RTI                     ;;GO BACK TO USER
                                        ;;*************************************************************
                                        ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                        ;*CALL:
                                        ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                        ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                        ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

014672  010346                          $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
```

```
 3889  014674  012703  015000          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
 3890  014700  022703  015010          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
 3891  014704  101405                          BLOS    4$              ;;BR IF YES
 3892  014706  104406                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
 3893  014710  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
 3894  014712  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
 3895  014716  001003                          BNE     3$              ;;SKIP IF NOT
 3896  014720  104400  013736          4$:     TYPE    ,$QUES          ;;TYPE A '?'
 3897  014724  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
 3898  014726  111337  014776          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
 3899  014732  104400  014776                  TYPE    ,9$
 3900  014736  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
 3901  014742  001356                          BNE     2$              ;;LOOP IF NOT RETURN
 3902  014744  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
 3903  014750  104400  013740                  TYPE    ,$LF            ;;TYPE A LINE FEED
 3904  014754  012603                          MOV     (SP)+,R3        ;;RESTORE R3
 3905  014756  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
 3906  014760  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
 3907  014766  012766  015000  000004          MOV     #$TTYIN,4(SP)
 3908  014774  000002                          RTI                     ;;RETURN
 3909  014776     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR, TO TYPE
 3910  014777     000                          .BYTE   0               ;;TERMINATOR
 3911  015000  000010                  $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
 3912  015010  052536  005015     000 $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
 3913  015015     136  006507  000012 $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
 3914  015022  005015  053523  020122 $MSWR:  .ASCIZ  <15><12>/SWR = /
 3915  015030  020075     000
 3916  015033     040  047040  053505 $MNEW:  .ASCIZ  /  NEW = /
 3917  015040  036440  000040
 3918  015044     000                  $AUTOB: .BYTE   0       ;;AUTO MODE FLAG
 3919  015045     000                  $INTAG: .BYTE   0       ;;INTERRUPT MODE FLAG
```



```
 3920                                  .SBTTL  TRAP DECODER
 3921
 3922                                  ;;**************************************************************
 3923                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
 3924                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
 3925                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
 3926                                  ;*GO TO THAT ROUTINE.
 3927
 3928  015046  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
 3929  015050  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
 3930  015054  005740                          TST     -(R0)           ;;BACKUP BY 2
 3931  015056  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
 3932  015060  006300                          ASL     R0              ;;POSITION FOR INDEXING
 3933  015062  016000  015070                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
 3934  015066  000200                          RTS     R0              ;;GO TO ROUTINE
 3935
 3936                                  .SBTTL  TRAP TABLE
 3937
 3938                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
 3939                                  ;*BY THE "TRAP" INSTRUCTION.
 3940
 3941                                  ;       ROUTINE
 3942                                  ;       -------
 3943  015070                  $TRPAD:
 3944  015070  013506                          $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
 3945  015072  013766                          $TYPOC  ;;CALL=TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
 3946  015074  013742                          $TYPOS  ;;CALL=TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
 3947  015076  014002                          $TYPON  ;;CALL=TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
 3948
 3949  015100  014340                          $GTSWR  ;;CALL=GTSWR    TRAP+4(104404)  GET SOFT-SWR SETTING
 3950
 3951  015102  014270                          $CKSWR  ;;CALL=CKSWR    TRAP+5(104405)  TEST FOR CHANGE IN SOFT-SWR
 3952  015104  014552                          $RDCHR  ;;CALL=RDCHR    TRAP+6(104406)  TTY TYPEIN CHARACTER ROUTINE
 3953  015106  014672                          $RDLIN  ;;CALL=RDLIN    TRAP+7(104407)  TTY TYPEIN STRING ROUTINE
 3954  015110  014170                          $SAVREG ;;CALL=SAVREG   TRAP+10(104410) SAVE R0-R5 ROUTINE
 3955  015112  014226                          $RESREG ;;CALL=RESREG   TRAP+11(104411) RESTORE R0-R5 ROUTINE
 3956  015114  006516                          XSUBSCOPE       ;;CALL=SUBSCOPE TRAP+12(104412)
```

```
                                        .SBTTL  POWER DOWN AND UP ROUTINES

                                        ;;**************************************************************
                                        ;POWER DOWN ROUTINE
015116  012737  015262  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
015124  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
015132  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
015134  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
015136  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
015140  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
015142  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
015144  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
015146  017746  164044                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
015152  010637  015266                  MOV     SP,$SAVR6       ;;SAVE SP
015156  012737  015170  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
015164  000000                          HALT
015166  000776                          BR      .-2             ;;HANG UP

                                        ;;**************************************************************
                                        ;POWER UP ROUTINE
015170  012737  015262  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
015176  013706  015266                  MOV     $SAVR6,SP       ;;GET SP
015202  005037  015266                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
015206  005237  015266          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
015212  001375                          BNE     1$              ;;OF  WORD
015214  012677  163776                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
015220  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
015222  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
015224  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
015226  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
015230  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
015232  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
015234  012737  015116  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
015242  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
015250  104400                          TYPE                    ;REPORT THE POWER FAILURE
015252  015270          $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
015254  012716                          MOV     (PC)+,(SP)      ;;RESTART AT RESTART
015256  001350          $PWRAD: .WORD   RESTART         ;;RESTART ADDRESS
015260  000002                          RTI
015262  000000          $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
015264  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
015266  000000          $SAVR6: 0                       ;;PUT THE SP HERE
015270  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
015276  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,@#$SB2D      ;;CALL
                                        ;*      RETURN                  ;;ADDRESS OF THE 1ST ASCIZ CHAR,IS ON THE STACK

015300  016637  000002  015330  $SB2D:  MOV     2(SP),1$        ;;SAVE BINARY NUMBER
015306  012746  015330                  MOV     #1$,-(SP)       ;;SET POINTER
015312  004737  015334                  JSR     PC,@#$DB2D      ;;CALL DOUBLE LENGTH CONVERT
015316  062716  000005                  ADD     #5,(SP)         ;;ONLY ALLOW FIVE CHARACTERS
015322  012666  000002                  MOV     (SP)+,2(SP)     ;;PICKUP POINTER
015326  000207                          RTS     PC              ;;RETURN
015330  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

015334  104410          $DB2D:  SAVREG                  ;;SAVE REGISTERS
015336  016602  000002                  MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
015342  012700  015514                  MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
015346  010066  000002                  MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
015352  012201                          MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
015354  012202                          MOV     (R2)+,R2
015356  012737  000012  015432          MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
015364  012704  015444                  MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
015370  012705  015446                  MOV     #$TNPWR+2,R5
015374  005003                  1$:     CLR     R3              ;;CLEAR PARTIAL
015376  161401                  2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
015400  005602                          SBC     R2
015402  161502                          SUB     (R5),R2
015404  002402                          BLT     3$              ;;BR IF TEN POWER TO LARGE
015406  005203                          INC     R3              ;;ADD 1 TO PARTIAL
015410  000772                          BR      2$              ;;LOOP
015412  062401                  3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
015414  005502                          ADC     R2
015416  062402                          ADD     (R4)+,R2
015420  022525                          CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
015422  052703  000060                  BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
015426  110320                          MOVB    R3,(R0)+        ;;SAVE IT
015430  005327                          DEC     (PC)+           ;;DONE?
015432  000000                  4$:     .WORD   0
015434  001357                          BNE     1$              ;;BR IF NO
```

```
4058  015436  105020                          CLRB    (R0)+           ;;TERMINATOR
4059  015440  104411                          RESREG                  ;;RESTORE REGISTERS
4060  015442  000207                          RTS     PC              ;;RETURN
4061  015444  145000                  $TNPWR: 145000                  ;;1.0E09
4062  015446  035632                          35632
4063  015450  160400                          160400                  ;;1.0E08
4064  015452  002765                          2765
4065  015454  113200                          113200                  ;;1.0E07
4066  015456  000230                          230
4067  015460  041100                          041100                  ;;1.0E06
4068  015462  000017                          17
4069  015464  103240                          103240                  ;;1.0E05
4070  015466  000001                          1
4071  015470  023420                          23420                   ;;1.0E04
4072  015472  000000                          0
4073  015474  001750                          1750                    ;;1.0E03
4074  015476  000000                          0
4075  015500  000144                          144                     ;;1.0E02
4076  015502  000000                          0
4077  015504  000012                          12                      ;;1.0E01
4078  015506  000000                          0
4079  015510  000001                          1                       ;;1.0E00
4080  015512  000000                          0
4081  015514  000014                  $DECVL: .BLKB   12.             ;;RESERVE STORAGE FOR ASCIZ STRING
4082
```



```
4083                                    ;;************************************************************
4084
4085                                    ;TYPE NUMERICAL ASCIZ STRING,RIGHT JUSTIFIED
4086                                    ;REPLACING LEADING ZEROS WITH SPACES.
4087                                    ;
4088                                    ;FIRST ADDRESS OF ASCIZ STRING MUST BE ON TOP OF THE STACK
4089
4090  015530  010046                  RTJUST:         MOV R0,-(SP)            ;SAVE R0
4091  015532  016600  000004                          MOV 4(SP),R0            ;PICK UP ADDRESS OF ASCIZ STRING
4092  015536  010037  015570                          MOV R0,3$               ;SAVE ADDRESS FOR TYPE OUT
4093  015542  105710                  1$:             TSTB (R0)               ;IS THIS THE TERMINATOR
4094  015544  001406                                  BEQ 2$                  ;IF YES TYPE IT OUT
4095  015546  122710  000060                        CMPB #'0,(R0)           ;IS IT A ZERO
4096  015552  001005                                  BNE 4$                  ;IF NO GO PRINT IT
4097  015554  112720  000040                        MOVB #' ,(R0)+          ;IF YES REPLACE IT WITH A SPACE
4098  015560  000770                                  BR 1$                   ;TEST NEXT CHAR.
4099  015562  112740  000060          2$:             MOVB #'0,-(R0)          ;STRING OFF ALL ZEROS,PUT BACK THE LAST ONE
4100  015566  104400                  4$:             TYPE                    ;TYPE THE STRING
4101  015570  000000                  3$:             OPEN
4102  015572  012600                                  MOV (SP)+,R0            ;RESTORE R0
4103  015574  012616                                  MOV (SP)+,(SP)          ;RESTORE THE STACK
4104  015576  000207                                  RTS PC                  ;RETURN
4105
4106                                    ;TYPES 16 BIT WORD IN DECIMAL
4107
4108  015600  012546                  SGLDEC:         MOV (R5)+,-(SP)         ;PUT NUMBER TO BE TYPED ON STACK
4109  015602  004737  015300                          JSR PC,@#$SB2D          ;CONVERT NUMBER TO DECIMAL
4110  015606  004737  015530                          JSR PC,RTJUST           ;TYPE THE DECIMAL NUMBER
4111  015612  000205                                  RTS R5
```

```
                                        .SBTTL  MESSAGES

015614  042524  052123  050040  MDTESTP:        .ASCIZ "TEST PARAMETERS: "
015622  051101  046501  052105
015630  051105  035123  000040

015636  005015  042412  040522  MXEHEADER:      .ASCIZ <15><12><12> "ERADR  FAST   FAPT          GOOD   BAD     PASS"
015644  051104  020040  040506
015652  052123  020040  043040
015660  050101  020124  020040
015666  020040  020040  020040
015674  043440  047517  020104
015702  020040  040502  020104
015710  020040  020040  040520
015716  051523     000
015721     125  042516  050130  MUNXDD:         .ASCIZ "UNEXPECTED D D MARK"<15><12>
015726  041505  042524  020104
015734  020104  020104  040515
015742  045522  005015     000

015747     104  042040  046440  MDDMIS:         .ASCIZ "D D MARK MISSING"<15><12>
015754  051101  020113  044515
015762  051523  047111  006507
015770  000012

015772  040504  040524  020054  MDERHDR:        .ASCIZ "DATA, NO STATUS ERROR"
016000  047516  051440  040524
016006  052524  020123  051105
016014  047522  000122

016020  047440  020116  051124  MTRK:           .ASCIZ " ON TRACK"
016026  041501  000113

016032  020040  051106  046517  MPRES:          .ASCIZ "  FROM TRACK"
016040  052040  040522  045503
016046     000

016047     040  020057  042523  MSECT:          .ASCIZ " / SECTOR"
016054  052103  051117     000

016061     015  020012  054502  MCOLMUN:        .ASCIZ <15><12>" BYTE  BAD  GOOD"<15><12>
016066  042524  020040  040502
016074  020104  043440  047517
016102  006504  000012

016106  020040  020040  020040  TAB:            .ASCIZ <40><40><40><40><40><40>
016114     000

016115     040  000040          DBLSP:          .ASCIZ <40><40>

016120  005015     000          MCRLF:          .ASCIZ <15><12>

016123     015  042412  051122  MERHEADER:      .ASCIZ <15><12>"ERROR CONDITIONS:  TEST PC = "
016130  051117  041440  047117
```



```
016136  044504  044524  047117
016144  035123  020040  042524
016152  052123  050040  020103
016160  020075     000

016163     125  044516  020124  MUNIT0:         .ASCIZ "UNIT 0 "
016170  220060     000

016173     125  044516  020124  MUNIT1:         .ASCIZ "UNIT 1 "
016200  020061     000

016203     117  046116  020131  MONLY:          .ASCIZ "ONLY  "
016210  000040

016212  054122  051503  036440  MRXCS:          .ASCIZ "RXCS = "
016220  000040

016222  052123  052101  051525  MASTAT:         .ASCIZ "STATUS A = "
016230  040440  036440  000040

016236  052123  052101  051525  MBSTAT:         .ASCIZ "STATUS B = "
016244  041040  036440  000040

016252  005015  000012          DBLLF:          .ASCIZ <15><12><12>

016256  047516  044440  052116  MINTER:         .ASCIZ "NO INTERRUPT AT DONE ERROR"
016264  051105  052522  052120
016272  040440  020124  047504
016300  042516  042440  051122
016306  051117     000

016311     125  045516  047516  MUKNINT:        .ASCIZ "UNKNOWN INTERRUPT"
016316  047127  044440  052116
016324  051105  052522  052120
016332     000

016333     124  052117  046101  MERCT:          .ASCIZ "TOTAL READ CHECK ERRORS = "
016340  051040  040505  020104
016346  044103  041505  020113
016354  051105  047522  051522
016362  036440  000040

016366  044506  046114  052502  MFIL:           .ASCIZ "FILLBUFFER "
016374  043106  051105  000040

016402  046505  052120  041131  MEMPTY:         .ASCIZ "EMPTYBUFFER "
016410  043125  042506  020122
016416     000

016417     040  051124  041501  MLIMTRK:        .ASCIZ " TRACKS 52,53,114,0  "
016424  051513  032440  026062
016432  031465  030454  032061
016440  030054  020040     000

016445     117  036504     000  MOD:            .ASCIZ "OD="
```

```
4224  016451  040  044440  036504  MID:          ,ASCIZ "  ID="
4225  016456  000
4226
4227  016457  040  043040  051111  MFIRST:       ,ASCIZ "  FIRST="
4228  016464  052123  000075
4229
4230  016470  020040  040514  052123  MLAST:      ,ASCIZ "  LAST="
4231  016476  000075
4232
4233  016500  051103  020103  051105  MBADCRC:    ,ASCIZ "CRC ERROR NO DATA ERROR"
4234  016506  047522  020122  047516
4235  016514  042040  052101  020101
4236  016522  051105  047522  000122
4237
4238  016530  042522  042101  000040  MREAD:      ,ASCIZ "READ "
4239
4240  016536  040504  040524  041440  MCRC:       ,ASCIZ "DATA CRC ERROR"
4241  016544  041522  042440  051122
4242  016552  051117  000
4243
4244  016555  123  042505  020113  MSEEK:        ,ASCIZ "SEEK ERROR"
4245  016562  051105  047522  000122
4246
4247  016570  051127  052111  020105  MWRITE:     ,ASCIZ "WRITE "
4248  016576  000
4249
4250  016577  120  051101  052111  MPAR:         ,ASCIZ "PARITY ERROR"
4251  016604  020131  051105  047522
4252  016612  000122
4253
4254  016614  051105  047522  020122  MNOFLAG:    ,ASCIZ "ERROR FLAG ERROR"
4255  016622  046106  043501  042440
4256  016630  051122  051117  000
4257
4258  016635  102  042101  000  MBAD:           ,ASCIZ "BAD"
4259
4260  016641  040  000  SPACE:                   ,ASCIZ <40>
4261
4262  016643  107  047517  000104  MGOOD:       ,ASCIZ "GOOD"
4263
4264  016650  020040  044103  041505  MSUM:      ,ASCIZ "  CHECK SUM "
4265  016656  020113  052523  020115
4266  016664  000
4267
4268  016665  015  051012  030530  MRX11:       ,ASCIZ <15><12>"RX11 / RXV11"
4269  016672  020061  020057  054122
4270  016700  030526  000061
4271
4272  016704  005015  046412  044501  MREV:      ,ASCIZ <15><12><12> "MAINDEC-11-DZRXB-E" <15><12>
4273  016712  042116  041505  030455
4274  016720  026461  055104  054122
4275  016726  026502  006505  000012
4276
4277  016734  005015  047125  054105  LOC4M:  ,ASCIZ  <15><12>"UNEXPECTED TRAP TO LOC. 4 OCCURRED"
4278  016742  042520  052103  042105
4279  016750  052040  040522  020120
```



```
4280  016756  047524  046040  041517
4281  016764  020056  020064  041517
4282  016772  052503  051122  042105
4283  017000  000
4284
4285  017001  015  052412  042516  LOC10M: ,ASCIZ  <15><12>"UNEXPECTED TRAP TO LOC. 10 OCCURRED"
4286  017006  050130  041505  042524
4287  017014  020104  051124  050101
4288  017022  052040  020117  047514
4289  017030  027103  030440  020060
4290  017036  041517  052503  051122
4291  017044  042105  000
4292
4293  017047  075  041520  000  PCM:     ,ASCIZ  "=PC"
4294
4295  017053  015  052012  040522  OD2BIG: ,ASCII  <15><12>"TRACK LIMITS SELECTED OUT OF RANGE"
4296  017060  045503  046040  046511
4297  017066  052111  020123  042523
4298  017074  042514  052103  042105
4299  017102  047440  052125  047440
4300  017110  020106  040522  043516
4301  017116  105
4302  017117  015  042012  043105          ,ASCIZ  <15><12>"DEFAULTING TO "
4303  017124  052501  052114  047111
4304  017132  020107  047524  000040
4305
4306  017140  005015  042523  052103  S2BIG:  ,ASCII  <15><12>"SECTOR LIMITS SELECTED OUT OF RANGE"
4307  017146  051117  046040  046511
4308  017154  052111  020123  042523
4309  017162  042514  052103  042105
4310  017170  047440  052125  047440
4311  017176  020106  040522  043516
4312  017204  105
4313  017205  015  042012  043105          ,ASCIZ  <15><12>"DEFAULTING TO "
4314  017212  052501  052114  047111
4315  017220  020107  047524  000040
4316
4317  017226  005015  040503  052125  D0LOAD: ,ASCII  <15><12>"CAUTION - IF YOU DESIRE TO TEST UNIT 0"
4318  017234  047511  020116  020055
4319  017242  043111  054440  052517
4320  017250  042040  051505  051111
4321  017256  020105  047524  052040
4322  017264  051505  020124  047125
4323  017272  052111  030040
4324  017276  005015  042522  046120          ,ASCII  <15><12>"REPLACE LOAD MEDIUM WITH A SCRATCH DISKETTE"
4325  017304  041501  020105  047514
4326  017312  042101  046440  042105
4327  017320  052511  020115  044527
4328  017326  044124  040440  051440
4329  017334  051103  052101  044103
4330  017342  042040  051511  042513
4331  017350  052124  105
4332  017353  015  052012  042510          ,ASCIZ  <15><12>"THEN PRESS CONTINUE"<15><12>
4333  017360  020116  051120  051505
4334  017366  020123  047503  052116
4335  017374  047111  042525  005015
```

```
  4336  017402     000
  4337
  4338           017404                          .EVEN
  4339
  4340                                           ;;*****************************************************************
  4341
  4342                                           ;THE FOLLOWING LOCATIONS ARE USED FOR DATA STORAGE,RETRY COUNTERS
  4343                                           ;ACCESS COUNTERS ETC,
  4344
  4345  017404  000200                  BUFADR:         .BLKB 200
  4346
  4347           000001                          .END
```



```
ADJSUM  010766          2321    2394    2412    2750    3029    3047#
ASTAT   012146          2807    2916    2923    2972    2990    3158*   3161    3164    3241#   3245*   3255
BADBYT  011454          3040*   3041    3042    3088    3136#
BIT0  = 000001           179#   2923
BIT00 = 000001           169#    179
BIT01 = 000002           168#    178
BIT02 = 000004           167#    177
BIT03 = 000010           166#    176
BIT04 = 000020           165#    175
BIT05 = 000040           164#    174
BIT06 = 000100           163#    173
BIT07 = 000200           162#    172
BIT08 = 000400           161#    171
BIT09 = 001000           160#    170
BIT1  = 000002           178#   2807    2916
BIT10 = 002000           159#
BIT11 = 004000           158#
BIT12 = 010000           157#
BIT13 = 020000           156#   2872
BIT14 = 040000           155#    449    1431    3432    3446
BIT15 = 100000           154#    454     470    3021    3466    3504
BIT2  = 000004           177#   3161
BIT3  = 000010           176#
BIT4  = 000020           175#   3233    3572
BIT5  = 000040           174#   1539    2636
BIT6  = 000100           173#    670     830    1082    1243    1287    1301    1341    1359    1401    1420    1488    1592
                        1831    1850    1956    2972    2990    3423    3428    3434
BIT7  = 000200           172#    673     674    1382    1491    1492    1596    2447    2453    3028    3083    3509    3574
BIT8  = 000400           171#    548     832     866    2528    2560    2979    2992    3067    3183
BIT9  = 001000           170#
BPTVEC= 000014           186#
BRLEV   001214           370#   2491
BSTAT   012150          3191*   3242#   3253*   3261
BUFADR  017404          1624    1727    3047*   3048*   3052*   3055*   3056    3276    3378    4345#
BUSERR  006130           214    2459#
BYTECN  011452          2753*   2761*   3026*   3044*   3084    3101*   3135#
CATERR  003032          1170    1175*   1178*   1190*
CCOUNT  002526           438*    835     838*    839     841*    877#   2527*
CEXIT   004246          1065    1303    1360    1421    1607    1617    1724    1810    1811#
CKBYTE  010736          3040#
CKSUM   011056          3030*   3041*   3049*   3050*   3051*   3052    3053*   3054*   3055    3059#   3117
CKSWR = 104405           846     891    2524    2565    2571    2621    2670    2770    2776    2825    2830    2835    2863
                        2871    2875    2938    2943    2954    2959    2974    2982    2993    3069    3097    3109    3123
                        3144    3150    3168    3176    3192    3951#
COMMWO  007634          2332    2792    2850#   2895
CONT10  010522          2971    2990#
CONT11  010470          2976    2982#   2995
CONT13  007312          2778    2780#
CONT14  007562          2832    2834#
CONT15  010312          2945    2947#
CONT16  010364          2961    2963#
CONT17  010502          2984    2986#
CONT20  011276          3099    3101#
CONT21  011522          3152    3154#
CONT4   007300          2772    2776#
COUNT   012450          3287    3362#
```

```
CPUPRI  006200          1334    1393    2491#
CR    = 000015            94#   3636    3646
CRCER   010234          2924    2932#
CRLF  = 000200            95#   3607    3646
DATABY  012322          3291#   3300*   3310*   3316*   3326*   3334*   3343*   3345*   3354*   3362*   3363*   3372*   3376
                        3377
DATACK  007704          2861#   2890*   2934*   3104
DATACR  010330          2933    2937    2954#
DATAER  011110          3043    3067#
DATA0   012326          3281    3300#
DATA1   012340          3282    3310#
DBLLF   016252          4191#
DBLSP   016115          3087    3091    3232    4162#
DDCHK   010410          2907    2970#
DDERR   010452          2979#   2998
DDISP = 177570           101*    378
DEATH   003034          1174    1184#
DEXIT   004260          1814#   1896    2030    2084    2136    2271    2308    2331    2399    2415    2428    2454
DING    006450          2561    2571#   3012
DISPLA  001220           378#    419*
DISPRE  000174           237#    419
DONE    012726          2743    3444#
DONEBI= 000040           201*    2796    2899    3166
DRVSWP  007012          2427    2717#
DSWR  = 177570           100*    377
DTESTP  001212           368*    445     449     452     473     867
D0LOAD  017226           464     4317#
EBIE  = 000103           203*    3036
EMPBUF  010644          2935    3023#   3063    3127    3131
EMPDON  011306          3031    3104#
EMPER   011060          3032    3061#
EMPFLA  010730          3037#   3038    3045    3102
EMTVEC= 000030           189*
EPCSCO  006474          2517*   2518*   2581#   2613
ERCNTR  011460          2936    3027*   3068*   3074    3107    3113    3138#
ERMSG   010566          2977    2996    3002#   3078    3147    3196
ERRORS  006472           549*   2519*   2580#   2586    2595*   2601    3181*
ERRVEC= 000004           182#
EXIT    011446          3025    3129    3132#
FAST    002532           551*    758*    879#   2540
FBEB    004064          1625    1634#   1640    1728    2324
FBIE  = 000101           202*    2757
FILLBU  007144          2750*   2767    2782    2816    2839
FILLDO  007332          2751    2785#   2812
FILLER  007222          2752    2764#
FILLFL  007202          2758*   2759    2762
FIRST   001202           301*    302     509     511*    514     517     531*    3547    3549*   3552    3555    3569
FIRSTT  002170           757*    872
FLOAT0  012350          3283    3316#
FLOAT1  012412          3284    3334#
FUNCTI  007702          2329*   2413*   2788*   2791*   2850*   2851    2860#   2894*   3198
GETPAT  012260          1623    2320    2393    2411    2717    2735    3276#
GETSEC  013422          2749    2889    3565#
GETTRA  013104          2706    2720    2738    3485#
GETUNI  012634          2318    2410    2705    2719    2723    2737    3423#
GNS   = ****** U        3944    3945    3946    3947    3949    3951    3952    3953    3954    3955    3956
```



```
GOODBY  011456          3092*   3094    3137#
GTSWR = 104404           425    3949#
HANGER  006756           440*    556*    831*    855*   2638*   2668*   2693*   2697#   2785*   2798*   2891*   2901*
HANGPL  006760           441*    557*    666*   2639*   2669*   2694*   2698#
HERE    002462           858     864#
HLT10   010310          2946#
HLT12   010362          2962#
HLT13   010500          2985#
HLT14   011274          3100#
HLT15   011520          3153#
HLT6    007310          2779#
HLT7    007560          2833#
HOME    007706          2834    2863#   2880
HT    = 000011            92#   3605    3646
HUNGUP  006622          2640    2647#
ID    = 001201           300#    486     488     496    3469
ILLADR  005106          1913    2046    2099    2140#   2145
INCERR  006244          2519#   2520
INITSE  013324          2748    2888    3547#
INITTR  012752          2704    2718    2736    3461#
INTLEA  013420          3558*   3563#   3570    3571*
INTSER  011526          2305    2425    2450    3154    3158#
IOTVEC= 000020           187#
KRXVEC  001204           304#    1018    1072    1236    1277    1319    1375    2304    2424    2449
LAST  = 001203           302#    512     514     522    3550*   3551    3567
LF    = 000012            93#   3640    3646
LIMITS  001622           478#    505
LIMTRK  013140          3487    3501#
LOAD    012510          3301    3321    3344    3364    3373    3376#
LOCKUP  002556           755     891#
LOC10M  017001          2475    4285#
LOC4M   016734          2459    4277#
LOGICA  002452           230     860#
MABELL  006470           843    2574    2577#
MASTAT  016222          3254    4185#
MBAD    016635          3119    4258#
MBADCR  016500          2941    4233#
MBSTAT  016236          3260    4188#
MCOLMU  016061          3082    4154#
MCRC    016536          2957    4240#
MCRLF   016120           477     526     837     849    2532    2775    2847    2942    2958    2981    3011    3080    3096
                        3122    3149    3172    3180    3195    3203    3217    3263    4164#
MDDMIS  015747          2978    4133#
MDERHD  015772          3079    4139#
MDTEST  015614           472    4115#
MEMPTY  016402          3062    4213#
MERCT   016333          3112    4204#
MERHEA  016123          3002    4166#
MFIL    016366          2765    4210#
MFIRST  016457           516    4227#
MGOOD   016643          3121    4262#
MHUNGP  006674          2674    2684#
MID     016451           495    4224#
MILTST  002536           874     883#
MINTER  016256          3148    4193#
MLAST   016470           521    4230#
```

```
MLIMTR  016417            481    4217#
MNOFLA  016614           3179    4254#
MOD     016445            490    4222#
MONLY   016203            462    4179#
MORETE  002162            755#   1812    1815    2673    2682
MPAR    016577           2774    3216    4250#
MPASS   006725            850    2689#   3007
MPRES   016032           2842    4147#
MREAD   016530           2919    2927    4238#
MREV    016704            411    4272#
MRXCS   016212           3197    3211    4182#
MRX11   016665           3208    4268#
MSECT   016047           3225    4151#
MSEEK   016555           2841    4244#
MSUM    016650           3116    4264#
MTRK    016020           3220    4144#
MUKNIN  016311           3171    4199#
MUNIT0  016163           3235    4173#
MUNIT1  016173            461    3237    4176#
MUNXDD  015721           2997    4128#
MWRITE  016570           2810    2821    2828    4247#
MX      002534            834*    842     881#   1811*   1814*
MXEHEA  015636           2530    4119#
NEXTRD  010132           2783    2906    2909    2911#   2921    2929    2949    2966    2989
NOERTY  011264           3071    3075    3097#
NOINTE  011462           2800    2903    3144#
NOMORE  002274            767     830#
NOPRIN  006420           2526    2560#
NOPRNT  011726           3192#
NOSCOP  006504           2595#   2623
NXREAD  011430           3064    3108    3125    3128#
OD      001200            299#    300     471*    478     488     491     504*   3463*   3464    3467    3471    3475*   3501
                         3504*   3506    3509*   3523*   3526    3536    3538*
OD2BIG  017053            501    4295#
OK2GO   003036           1177    1189#
OPEN  = 000000            206#    853    2557    2773    3010    3086    3115    3239    4101
PARTES  007252           2768    2770#   2813    2922    3065
PARTYP  012022           3189    3211#   3251
PASS    002530            439*    844*    851     878#   2555    3008
PAT     012324           1615*   1802*   1807*   2319*   2390*   2392*   2407*   2409*   2423*   2436*   2448*   3278    3292#
PATGEN  012332           3301#   3302    3311
PATTER  012302           3280    3281#
PAT125  012420           3285    3343#
PAT314  012440           3286    3354#
PC    =%000007            113#    755*    860*    901*    903*   1334*   1393*   1432*   1548*   1616*   1623*   1625*   1626*
                         1634*   1637*   1639*   1708*   1723*   1728*   1729*   1808*   1809*   1826*   1845*   1913*   1969*
                         2046*   2074*   2099*   2126*   2140*   2144*   2148*   2152*   2156*   2161*   2164*   2176*   2179*
                         2185*   2188*   2194*   2249*   2307*   2318*   2320*   2321*   2324*   2330*   2332*   2333*   2339*
                         2378*   2379*   2391*   2393*   2394*   2396*   2398*   2408*   2410*   2411*   2412*   2414*   2427*
                         2452*   2505*   2561*   2575*   2632*   2641*   2696*   2704*   2705*   2706*   2707*   2710*   2717*
                         2718*   2719*   2720*   2721*   2722*   2723*   2724*   2725*   2728*   2735*   2736*   2737*   2738*
                         2739*   2740*   2743*   2744*   2748*   2749*   2750*   2792*   2803*   2822*   2829*   2834*   2838*
                         2848*   2852*   2855*   2858*   2867*   2874*   2881*   2883*   2888*   2889*   2895*   2907*   2910*
                         2913*   2928*   2935*   2977*   2980*   2996*   2999*   3012*   3013*   3022*   3029*   3057*   3078*
                         3081*   3106*   3126*   3130*   3133*   3147*   3154*   3175*   3185*   3189*   3196*   3202*   3204*
                         3206*   3218*   3238*   3247*   3251*   3264*   3301*   3321*   3344*   3364*   3371*   3373*   3380*
```



```
                         3382*   3402*   3429*   3436*   3449*   3476*   3505*   3510*   3514*   3516*   3527*   3529*   3540*
                         3542*   3559*   3575*   3614*   3621*   3628*   3642*   3644*   3833*   3993    4015*   4018*   4055*
                         4060*   4104*   4109*   4110*
PCM     017047           2466    2483    4293#
PCONT   007326           2766*   2781    2783#   2811*   2921*   3064*
PCSCOP  006556            551     757*    758    2543    2596*   2624    2626#   2789    2905    2970    3003
PIRQ  = 177772             99#
PIRQVE= 000240            193#
PLOOP   007322           2767*   2782#   2812*   2920*   3063*
PRESTR  013130           2843    2882*   3485*   3493#   3524
PRETES  002034            527     548#
PR0   = 000000            116#
PR1   = 000040            117#
PR2   = 000100            118#
PR3   = 000140            119#
PR4   = 000200            120#   1328
PR5   = 000240            121#   1387
PR6   = 000300            122#
PR7   = 000340            123#    215     217     221     223     225     227     431    1022    1030    1076    1125    1240
                         1263    1281    1290    1323    1344    1379    1404    2306    2426    2451
PS    = 177776             96#     97
PSW   = 177776             97#   1328*   1387#
PTYP1   007264           2765*   2773#   2810*   2919*   3062*
PWRVEC= 000024            188#   3961*   3962*   3971*   3977*   3989*   3990*
RANDAT  012470           3288    3371#   3374
RANGEN  012540           3371    3386#
RANUM   012632           3372    3401*   3406#
RAN1    012626            436*   3387    3393*   3398    3404#
RAN2    012630            437*   3388    3395    3400*   3405#
RDCHR = 104406           3892    3952#
RDCODE  012152           2874    3245#
RDDONE  010110           2892    2905#
RDER  = 000017            200#   3184    3246
RDERR   010142           2893    2915#
RDIE  = 000107            205#   2894
RDLIN = 104407           3953#
RDONLY  006762           2307    2704#
READ    010014           2707    2888#   3022
READCH  010634           2722    2725    2740    3021#
REBEGI  002466            676     754     864     866#   2467    2484
RECAL = 040001            207#   2395    2866
REREAD  010024           2890*   2920    2926    2950    2967    2988    3126
RESERR  006154            216    2475#
RESREG= 104411           3955*   4059
RESTAR  001350            404     430*   3210    3994
RESVEC= 000010            183#
RETURN  010564           2973    2991    2999#
RTJUST  015530           4090*   4110
RTN     010012           2865    2883#
RTSPC   006572           2630    2632#
RXCS    001206            306#    434     553     830*    917*    919    1082*   1196    1198    1243*   1248    1250    1287*
                         1301*   1341*   1359*   1401*   1420*   1431*   1435   1538*   1539    1541    1551    1622*   1641
                         1726*   1825*   1844*   1910*   1912*   1916    1968*   1971    1974    2043*   2045*   2049    2073*
                         2096*   2098*   2102    2125*   2165    2175*   2181    2190    2197    2248*   2251    2254    2323*
                         2335    2338*   2395*   2629    2636    2757*   2758    2796    2851*   2866*   2869    2899    3036*
                         3037    3159    3166    3184*   3212    3246*
```

```
RXDB    001210   348#   434*   435*   668    973   1486   1590   1628*  1643   1734   1829   1848   1954
                1975   1985   2063   2077   2115   2129   2147*  2155*  2182   2184*  2191   2193*  2237
                2255   2263   2326*  2337   2341   2760*  2854*  2857*  3040   3158   3187   3191   3245
                3249   3253
RXERRO  011652  3160   3178   3181#  3182
RXPWR   012006  3163   3208#
R0      =%000000 104#   402*   408*   442    552*   554    667*   674*   675    918*   920    972*   974
                1195*  1196   1247*  1248   1337   1397   1434*  1436   1485*  1492*  1493   1542*  1550*
                1552   1593*  1597*  1598   1733*  1736   1828*  1832   1847*  1851   1907*  1915*  1917
                1953*  1957   1973*  1984*  1987   2039*  2048*  2050   2062*  2064   2076*  2078   2092*
                2101*  2103   2114*  2116   2128*  2130   2174*  2196*  2198   2235*  2238   2253*  2262*
                2264   2326   2376*  2491*  2496*  2498*  2499*  2500*  2501*  2502*  2503*  2549   2760
                3042   3056*  3092   3093*  3386*  3387*  3388*  3389*  3391*  3392*  3393   3394*  3395*
                3396*  3397*  3398*  3399*  3400   3401   3597   3598*  3599   3602*  3750   3775*  3928
                3929*  3930   3931*  3932*  3933*  3934*  3963   3988*  4035*  4036   4054*  4058*  4090
                4091*  4092   4093   4095   4097*  4099*  4102*
R1      =%000001 105#   553*   554    669*   670*   673*   675    919*   920    973*   974    1198*  1250*
                1316*  1372*  1382*  1435*  1436   1487*  1488*  1491*  1493   1541*  1551*  1552   1591*
                1592*  1596*  1598   1643*  1734*  1736   1830*  1831*  1832   1849*  1850*  1851   1916*
                1917   1955*  1956*  1957   1974*  1985*  1987   2049*  2050   2063*  2064   2077*  2078
                2102*  2103   2115*  2116   2129*  2130   2165*  2181*  2190*  2197*  2198   2237*  2238
                2254*  2263*  2264   2337*  2377*  2493   2552   2908   2932   3021*  3028*  3076   3083*
                3132*  3751   3774*  3964   3987*  4037*  4043*  4049*
R2      =%000002 106#   668*   669    1018*  1019   1021*  1022*  1023*  1072*  1073   1075*  1076*  1077*
                1236*  1237   1239*  1240*  1241*  1277*  1278   1280*  1281*  1282*  1319*  1320   1322*
                1323*  1324*  1375*  1376   1378*  1379*  1380*  1486*  1487   1590*  1591   1621*  1629*
                1676   1725*  1735*  1829*  1830   1848*  1849   1906*  1954*  1955   1975*  1986*  2040*
                2061*  2093*  2113*  2142*  2150*  2158*  2173*  2182*  2191*  2236*  2255*  2261*  2304*
                2305*  2306*  2322*  2327*  2341*  2424*  2425*  2426*  2449*  2450*  2451*  2546   3318*
                3320*  3323   3752   3773*  3965   3986*  4034*  4037   4038*  4044*  4045*  4050*  4051*
R3      =%000003 107#   756    757    760    867*   868*   869    871*   3688   3697*  37w3*  37w4*  37w7*
                3712*  3713*  3714   3723*  3753   3772*  3888   3889*  3890   3893*  3894   3898   3900
                3902*  3904*  3966   3985*  4042*  4047*  4053*  4054
R4      =%000004 108#   1624*  1628   1727*  1733   2533*  2534*  2535   3276*  3377*  3378   3689   3691*
                3692*  3693*  3694   3695*  3709   3711*  3719*  3722*  3754   3771*  3967   3984*  4040*
                4043   4049   4051
R5      =%000005 109#   852*   857*   860    2556*  3009*  3085*  3114*  3278*  3279*  3280   3690   3696*
                3698*  3700*  3701*  3702*  3703   3721*  3755   3770*  3968   3983*  4041*  4045   4052
                4108   4111*
R6      =%000006 110#   112
R7      =%000007 111#   113
SAVREG= 104410   3954*  4033
SA200   001232   244    408#   875
SA202   001222   245    402#
SCOPIN  006536   2567   2614   2621#
SDN     006574   1432   1548   1639   1826   1845   1969   2074   2126   2144   2152   2161   2179   2188
                2194   2249   2333   2339   2396   2636#  2867   3185   3247
SECCNT  013414   2801*  2911*  3023   3128*  3551*  3552*  3553*  3554*  3561#
SECLMT  001726   482    500    5w9#
SEEKER  007526   2822   2825#  2928
SEKRTY  007602   2816*  2837   2839#  2926*
SEKTYP  007606   2829   2841#
SEQUEN  013136   480*   2447*  2453*  3461   3466*  3486   3496#
SEQ1    013236   3488   3523#
SEQ2    013272   3489   3536#
SGLDEC  015600   852    2556   3009   3085   3114   4108#
```



```
SP      =%000006 112#   403*   410*   412*   417    429    431*   432*   473*   491*   496*   517*   522*
                759*   900*   1019*  1023   1025*  1026*  1030*  1031*  1073*  1077   1079*  1080*  1125*
                1126*  1237*  1241   1258*  1259*  1263*  1264*  1278*  1282   1284*  1285*  1290*  1291*
                1320*  1324   1326*  1330   1331   1333   1337*  1338*  1344*  1345*  1376*  1380   1385*
                1389   1390   1392   1397*  1398*  1404*  1405*  1636*  2163*  2461*  2467*  2478*  2484*
                2517   2533   2535*  2540*  2543*  2546*  2549*  2552*  2596   2613   2624*  2631*  2678*
                2695*  2751*  2752*  2754*  2755*  2764   2786*  2787*  2793*  2794*  2806   2843*  2892*
                2893*  2896*  2897*  2915   2965*  3003*  3031*  3032*  3033*  3034*  3061   3088*  3094*
                3174*  3198*  3205*  3212*  3221*  3228*  3255*  3261*  3381   3448*  3597*  3598   3599*
                3601   3602   3603*  3605   3607   3609   3615   3617*  3619*  3627*  3631   3635   3636
                3640   3680*  3681   3682   3683*  3688*  3689*  3690*  3696   3721   3722   3723   3724*
                3725*  3750*  3751*  3752*  3753*  3754*  3755*  3756*  3757*  3758*  3759*  3766*  3767*
                3768*  3769*  3770   3771   3772   3773   3774   3775   3793*  3794*  3795   3802*  3805*
                3806*  3810*  3811*  3815   3818*  3822   3824   3826   3827*  3834   3836   3838*  3839
                3841*  3842*  3843*  3844*  3845*  3860*  3861*  3864*  3865*  3866   3870*  3871*  3872
                3875   3877   3879*  3888*  3893   3904   3905*  3906*  3907*  3928*  3929   3963*  3964*
                3965*  3966*  3967*  3968*  3969*  3970   3978*  3982   3983   3984   3985   3986   3987
                3988   3993*  4013   4014*  4016*  4017*  4034   4036*  4090*  4091   4102   4103*  4108*
SPACE   016641   2539   2542   2545   2548   2551   2554   4260#
STACK = 001200    87#    297    403    410    759
STKLMT= 177774    98#
STR     006560   1634   2140   2148   2156   2176   2185   2629#  2852   2855
STYP1   007540   2821*  2828#  2927#
SUBSCO= 104412    957    1015   1189   1228   1255   1273   1470   1532   1547   1585   1672   1838   1948
                1963   1980   2056   2069   2109   2121   2230   2244   2260   2375   3956#
SUM     012536   3049   3277*  3376*  3384#
SWHLT1  007550   2827   2830#
SWR     001216    377#   409*   414    418*   423    847    892    898   2525   2566   2572   2622   2671
                2771   2777   2780   2826   2831   2836   2864   2872   2876   2879   2939   2944   2947
                2955   2960   2963   2975   2983   2986   2994   3070   3072   3098   3110   3124   3145
                3151   3169   3177   3193   3789   3826*  3969   3982*
SWREG   000176    238#   418    423   3789   3802
SW0     = 000001  151#   2572
SW00    = 000001  141#   151
SW01    = 000002  140#   150
SW02    = 000004  139#   149
SW03    = 000010  138#   148
SW04    = 000020  137#   147
SW05    = 000040  136#   146
SW06    = 000100  135#   145
SW07    = 000200  134#   144
SW08    = 000400  133#   143
SW09    = 001000  132#   142
SW1     = 000002  150#
SW10    = 002000  131#   892
SW11    = 004000  130#   2622   2780   2836   2879   2947   2963   2986   3124
SW12    = 010000  129#   898
SW13    = 020000  128#   2525   2671   2771   2826   2939   2955   2975   2994   3070   3110   3145   3169
                3177   3193
SW14    = 040000  127#   847
SW15    = 100000  126#   2566
SW2     = 000004  149#
SW3     = 000010  148#
SW4     = 000020  147#
SW5     = 000040  146#
SW6     = 000100  145#
```

```
SW7    = 000200            144#
SW8    = 000400            143#   2864
SW9    = 001000            142#   3072
S2BIG    017140            528    4306#
TAB      016106           3259    4159#
TARGET   013126           2316*   2857    3047    3050    3221    3467*   3485    3492*   3503*   3508*   3511    3513*   3515*
                          3526*   3528*   3539*   3541#
TBITVE=  000014            184#
TESTS    002214            757     760     761#
TKVEC  = 000060            191#
TPVEC  = 000064            192#
TRAPVE=  000034            190#
TRKCNT   013124           2708*   2726*   2741*   3472*   3473*   3474*   3477*   3479*   3491#
TRKLMT   001642            479     486#
TRTVEC=  000014            185#
TSECTO   013416           2317*   2854    3048    3051    3226    3555*   3556*   3562#   3565*   3566*   3567    3569*   3570*
                          3574*
TYPADR   012050           2980    3081    3202    3220#
TYPCOD   012220           3204    3254#
TYPE   = 104400            411     461     462     464     472     477     481     490     495     501     516     521     526
                           528     837     842     843     849     850     874    2459    2466    2475    2483    2530    2532
                          2539    2542    2545    2548    2551    2554    2574    2674    2773    2774    2775    2828    2841
                          2842    2847    2941    2942    2957    2958    2978    2981    2997    3002    3007    3011    3079
                          3080    3082    3087    3091    3096    3112    3116    3119    3121    3122    3148    3149    3171
                          3172    3179    3180    3195    3197    3203    3208    3209    3211    3216    3217    3220    3225
                          3232    3235    3237    3254    3259    3260    3263    3610    3715    3800    3801    3804    3817
                          3828    3847    3896    3899    3903    3944#   3991    4100
TYPERR   011210           3077    3084#
TYPOC  = 104401           3803    3945#
TYPON  = 104403           2541    2544    2547    2550    2553    3095    3262    3947#
TYPOS  = 104402            474     492     497     518     523    2463    2480    2536    2679    2844    3004    3089    3199
                          3213    3222    3229    3256    3946#
T0NOTH   002166            756#    761
T1       002620            761     917#
T10      004140            764    1715#
T11      004226            764    1807#
T12      004216            764    1802#
T13      004272            764    1823#
T14      004416            764    1906#
T15      004610            764    2039#
T16      004750            764    2092#
T17      005216            764    2173#
T2       002740            761    1072#
T20      005462            767    2281#   2905
T21      005506            767    2316#
T21X     005610           2330    2332#   2414
T22      005672            767    2388#
T23      005750            767    2407#
T24      006016            767    2423#
T24X     006024           2424#   2437
T25      006050            767    2436#   2789    2970
T26      006062            767    2447#
T3       003262            761    1316#
T4       003420            761    1372#
T5       003564            761    1431#
T6       004010            761    1615#
```



```
T6FILL   004026           1616    1621#   1808    2391    2408
T7       004142            761    1723#
T7EMPT   004152           1723    1725#   1809    2378    2398
UNITSE   012724            444*    447*    451*    454*    460*    470*    548*    671     832     866*    902*   1489    1594
                          1823    1842    1908    2041    2094    2388    2528   2560*   2850    2979*   2992*   3067*   3183*
                          3233    3423    3425    3427*   3428*   3430    3432    3434*   3435*   3440#   3444    3446    3572
WRITE    007134           2721    2724    2739    2748#
WRTDON   007430           2786    2801#   2811    2823
WRTER    007444           2787    2806#
WRTIE  = 000105            204#   2788
WRTSEK   007504           2808    2816#
WTDDIF=  000115            206#   2791
WTROCK   007066           2452    2735#
XERROR   006232            224    2517#
XFRBYT   007210           2760#
XHOME    010004           2870    2882#
XID      013134           3468*   3469*   3472    3495#   3524
XOD      013132           3470*   3471*   3473    3494#
XPATGE   012356           3317#   3335
XREAD    010020           2889#   2912    3130
XSA202   001574            443     457     469     471#
XSCOPE   006476            220   2586#
XSDN     006736           2637    2693#
XSUBSC   006516           2601#   3956
XWRITE   007140           2749#   2766    2804
XWTRDC   007072           2736#   2745
XXPATG   012426           3344#   3346    3355
$AUTOB   015044            427*   3797    3918#
$CHARC   013722           3612*   3622*   3629    3638*   3643#
$CKSWR   014270           3789#   3951
$CNTLG   015015           3800    3913#
$CNTLU   015010           3817    3912#
$CRLF    013737           3611    3653#   3828    3912
$DB2D    015334           4015    4033#
$DECVL   015514           4035    4081#
$FILLC   013734           3615    3650#
$FILLS   013733           3649#
$GTSWR   014340           3801#   3949
$HD    = 000003             21      22
$ILLUP   015262           3961    3977    3996#
$INTAG   015045           3829    3919#
$LF      013740           3654#   3903    3912
$MAIL  = ****** U          423    3599
$MNEW    015033           3804    3916#
$MSWR    015022           3801    3914#
$NULL    013732           3617    3648#
$OCNT    014164           3687*   3716*   3729#
$OMODE   014166           3682*   3686*   3691    3694*   3705*   3731#
$POWER   015270           3209    3992    3999#
$PWRAD   015256           3994#
$PWRDN   015116            222    3961#   3989
$PWRMG   015252           3992#
$PWRUP   015170           3971    3977#
$QUES    013736           3652#   3847    3896    3912
$RDCHR   014552           3860#   3952
$RDDEC=  ****** U         3954
```

```
$RDLIN  014672          3888#   3953
$RDOCT= ****** U        3954
$RDSZ = 000010          3881#
$RESRE  014226          3765#   3955
$R2A  = ****** U        3956
$SAVRE  014170          3749#   3954
$SAVR6  015266          3970#   3978    3979#   3980#   3998#
$SB2D   015300          4013#   4109
$SETUP= 000114           240#    420    3784    3918
$STUP = 177777           240#
$SWR  = 160000            21      22#   3995
$TKB    014266          3781#   3793    3810    3864    3870
$TKS    014264          3780#   3791    3807    3831#   3862    3868
$TN   = 000001            21#
$TNPWR  015444          4040    4041    4061#
$TPB    013730          3635#   3647#
$TPFLG  013735          3593    3651#
$TPS    013726          3633    3646#
$TRAP   015046           226    3928#
$TRP  = 000013          3936#   3945#   3946#   3947#   3948#   3949    3950#   3951    3952#   3953#   3954#   3955#   3956#
                        3957#
$TRPAD  015070          3933    3943#
$TIYIN  015000          3889    3890    3907    3911#
$TYPBN= ****** U        3948
$TYPDS= ****** U        3948
$TYPE   013506          3593#   3936    3944
$TYPEC  013656          3614    3621    3628    3633#   3634    3833
$TYPEX  013724          3639    3641    3644#
$TYPOC  013766          3685#   3945
$TYPON  014002          3684    3687#   3947
$TYPOS  013742          3680#   3946
$0FILL  014165          3681#   3685#   3695    3730#
.     = 017604           210#    212     213#    219#    229#    233#    236#    241#    297#    887#   1803   2691#   3646
                        3647    3648    3649    3650    3651    3652    3653    3654    3655    3780    3781    3911#   3912
                        3918    3919    3920    3973    3997    4081#   4338#   4345#
```



```
COMMEN      6#    194#    564     685     927     981    1041    1085    1133    1204    1443    1502    1559    1649    1683
         1747    1858    1924    1994    2207    2283    2347    2648
ENDCOM      7#    194#    649     750     953    1011    1060    1120    1163    1224    1466    1528    1581    1668    1702
         1791    1891    1944    2025    2226    2300    2371    2663
ERROR      88#     562     682     925     979    1039    1131    1202    1254    1272    1296    1351    1414    1441    1500
         1546    1557    1605    1647    1681    1745    1837    1856    1922    1962    1979    1992    2055    2068    2082
         2108    2120    2134    2205    2243    2259    2269    2345    2647
ESCAPE    194#
GETPRI    194#
GETSWR      7#    194#    420
MULT      194#
M7727     248#    618     731    1447    1772    1862    1998    2351
M7846     246#    563     684     926     980    1040    1084    1132    1203    1501    1558    1648    1682    1746    1923
         2206
NEWTST    194#
POP       194#   3770    3982    3983
PUSH      194#   3750    3963    3969
REPORT    194#
SCOPE      89#    550    1064    1300    1358    1419    1606    1706    1795    1895    2029    2083    2135    2270
SETPRI      6#    194#   1337    1344    1397    1404
SETTRA   3936#   3945    3946    3947    3949    3951    3952    3953    3954    3955    3956
SETUP     194#
SKIP      194#
SLASH     194#
SPACE     194#
STARS       6#    194#    251     252     253     264     265     266     271     272     273     292     293     294     385
          396    2509    2510    2514    2515    2712    2730    3140    3272    3295    3304    3329    3337    3348    3357
         3366    3421    3483    3498    3518    3531    3578    3657    3734    3779    3784    3852    3881    3922    3959
         3975    4004    4022    4083    4340
SWRSU     194#
TRMTRP   3936#
TYPBIN    194#
TYPDEC    194#
TYPNAM    194#
TYPNUM    194#
TYPOCS      5#    194#    473    2461    2478    2843    3003    3198    3212    3221    3228    3255
TYPOCT    194#   3802
TYPTXT    194#
$$ESCA    194#
$$NEWT    194#
$$SET    3936#   3945    3946    3947    3949    3951    3952    3953    3954    3955    3956
$$SKIP    194#
.EQUAT      5#     84
.HEADE      5#     11
.SETUP      5#    240
.$DB2D      6#   4020
.$POWE      6#   3957
.$READ      7#   3777
.$SAVE      6#   3732
.$SB2D      6#   4002
.$TRAP      5#   3920
.$TYPE      5#   3576
.$TYPO      5#   3655
```

```
ADC      557  2639  4050
ADD      435   556   900  1636  2163  2518  2631  2638  2695  2965  3041  3050  3051  3174  3205
        3376  3387  3388  3398  3448  3566  3570  3603  3683  3693  3818  3827  4016  4049  4051
ASL      871  2498  2499  2500  2501  2502  3279  3841  3842  3843  3932
ASLB    3053
BCC     3319
BEQ      415   450   459   555   676   833   848   858   893   899   921   975  1197  1249  1437
        1494  1540  1553  1599  1677  1737  1833  1852  1918  1958  1988  2051  2065  2079  2104
        2117  2131  2199  2239  2265  2520  2567  2614  2623  2640  2672  2781  2797  2808  2900
        2906  2917  2987  2991  3105  3108  3118  3125  3129  3162  3182  3188  3250  3324  3379
        3465  3487  3512  3573  3606  3641  3710  3798  3825  3840  4094
BGE     3568
BGT     3075  3717  3837  3878
BHI      487   489   513   515
BHIS     870
BIC      460   471   548   670   673   674   830   866   868   902  1243  1301  1359  1420  1488
        1491  1492  1592  1596  1831  1850  1956  2453  2503  3083  3227  3389  3399  3427  3434
        3463  3504  3509  3523  3538  3565  3707  3794  3811  3838  3865  3871  3879
BIS      447   451   454   470  1082  1287  1341  1382  1401  1431  2395  2447  2560  2979  2992
        3021  3028  3067  3183  3428  3435  3466  3475  3574  3712  3713  3845  4053
BISB    2850
BIT      445   449   832   847   892   898  1539  2525  2528  2566  2572  2622  2636  2671  2771
        2780  2796  2807  2826  2836  2864  2872  2879  2899  2916  2923  2939  2947  2955  2963
        2975  2986  2990  2994  3070  3072  3110  3124  3145  3161  3164  3169  3177  3187  3193
        3233  3249  3423  3432  3446  3572
BITB    2972  3166  3629
BLOS    3891
BLT     3620  3710  3835  3876  4046
BMI      672  1490  1595  2495  3160  3462  3488
BNE      422   424   443   446   457   479   510   558   836   840   856  1174  2526  2529  2573
        2587  2602  2637  2709  2727  2742  2772  2790  2799  2802  2827  2837  2865  2873  2880
        2902  2912  2924  2937  2940  2948  2956  2964  2971  2973  2976  2995  3024  3043  3071
        3073  3111  3146  3165  3167  3170  3178  3194  3234  3424  3433  3447  3525  3548  3600
        3608  3616  3630  3637  3708  3790  3796  3816  3823  3830  3867  3873  3895  3901  3951
        4057  4096
BPL      453  1642  1824  1843  1909  1972  2042  2095  2252  2336  2389  2630  2759  2778  2832
        2870  2877  2909  2933  2945  2961  2984  3038  3077  3099  3152  3426  3431  3445  3502
        3507  3537  3594  3634  3706  3792  3808  3863  3869
BR       242   243   404   416   426   448   463   469   482   500   505   527   532   754   872
        1177  1329  1332  1388  1391  1408  1433  1549  1630  1635  1640  1796  1803  1827  1846
        1911  1914  1970  2044  2047  2075  2097  2100  2127  2141  2143  2145  2146  2149  2151
        2153  2154  2157  2159  2162  2166  2177  2178  2180  2183  2186  2187  2189  2192  2195
        2250  2325  2328  2334  2340  2397  2745  2762  2823  2853  2856  2868  2929  2949  2998
        3045  3120  3186  3190  3236  3248  3252  3302  3311  3327  3335  3346  3355  3365  3374
        3478  3480  3489  3596  3613  3623  3632  3639  3684  3699  3720  3819  3846  3848  3874
        3897  3973  3997  4048  4098
BVS      845
CLC     3322  3390
CLR      408   438   439   440   444   480   504   549   831   841   917   972  1025  1079  1190
        1258  1284  1316  1372  1542  1597  1621  1725  1807  1906  1907  1953  1986  2039  2040
        2061  2092  2093  2113  2114  2147  2155  2173  2174  2235  2236  2261  2322  2376  2377
        2390  2407  2527  2595  2668  2693  2753  2754  2785  2793  2890  2891  2896  3026  3027
        3030  3033  3132  3277  3300  3334  3394  3468  3470  3515  3697  3805  3806  3979  4042
CLRB    3554  3612  3638  3902  4058
CMP      414   417   423   554   675   839   869   920   974  1196  1248  1330  1389  1436  1493
        1552  1598  1676  1736  1832  1851  1917  1957  1987  2050  2064  2078  2103  2116  2130
```



```
        2198  2238  2264  2613  2789  2905  2970  3074  3378  3511  3789  3795  3815  3822  3834
        3836  3866  3872  3875  3877  3890  4052
CMPB     455   486   488   512   514  3042  3524  3567  3605  3607  3615  3636  3640  3797  3829
        3894  3900  4095
COMB    3345
DEC     2496  2708  2726  2741  2801  2911  3128  3541  4055
DECB    3619  3622  3705  3716
EMT       88
HALT     212   468   854   894  1184  2568  2779  2833  2878  2946  2962  2985  3100  3153  3595
        3972  3996
INC      402   838   844   855  1629  1735  2142  2150  2158  2327  2392  2409  2519  2761  2798
        2901  2934  3044  3068  3093  3101  3181  3320  3363  3474  3528  3549  3553  3571  3711
        3719  3844  3980  4047
INCB    1175  3642
IOT       89
JMP      244   245   760   864   875  1065  1303  1360  1421  1607  1617  1724  1810  1812  1815
        1896  2030  2084  2136  2271  2308  2331  2399  2415  2428  2437  2454  2673  2682  2768
        2782  2783  2800  2804  2813  2839  2903  2922  2950  2966  2967  2988  2989  3025  3065
        3102  3127  3131  3163  3210  3280
JSR      755   852   860  1334  1393  1432  1548  1616  1623  1625  1634  1639  1723  1728  1808
        1809  1826  1845  1913  1969  2046  2074  2099  2126  2140  2144  2148  2152  2156  2161
        2176  2179  2185  2188  2194  2249  2307  2318  2320  2321  2324  2330  2332  2333  2339
        2378  2391  2393  2394  2396  2398  2408  2410  2411  2412  2414  2427  2452  2556  2561
        2704  2705  2706  2707  2717  2718  2719  2720  2721  2722  2723  2724  2725  2735  2736
        2737  2738  2739  2740  2743  2748  2749  2750  2792  2822  2829  2834  2852  2855  2867
        2874  2888  2889  2895  2907  2928  2935  2977  2980  2996  3009  3012  3022  3029  3078
        3081  3085  3114  3126  3130  3147  3154  3185  3189  3196  3202  3204  3247  3251  3301
        3321  3344  3364  3371  3373  3614  3621  3628  3833  4015  4109  4110
MOV      403   409   410   412   413   418   419   429   431   432   434   436   437   441   473
         531   551   552   553   666   667   668   669   757   758   759   834   851   857   867
         918   919   973  1018  1019  1021  1022  1023  1026  1030  1031  1072  1073  1075  1076
        1077  1080  1125  1126  1195  1198  1236  1237  1239  1240  1241  1247  1250  1259  1263
        1264  1277  1278  1280  1281  1282  1285  1290  1291  1319  1320  1322  1323  1324  1326
        1327  1328  1331  1333  1337  1338  1344  1345  1375  1376  1378  1379  1380  1385  1386
        1387  1390  1392  1397  1398  1404  1405  1434  1435  1485  1486  1487  1538  1541  1550
        1551  1590  1591  1593  1615  1622  1624  1643  1726  1727  1802  1811  1814  1825  1828
        1829  1830  1844  1847  1848  1849  1910  1912  1915  1916  1954  1955  1968  1973  1974
        1975  1984  1985  2043  2045  2048  2049  2062  2063  2073  2076  2077  2096  2098  2101
        2102  2115  2125  2128  2129  2165  2175  2181  2182  2184  2190  2191  2193  2196  2197
        2237  2248  2253  2254  2255  2262  2263  2304  2305  2306  2316  2317  2319  2323  2329
        2337  2338  2341  2413  2423  2424  2425  2426  2436  2448  2449  2450  2451  2461  2467
        2478  2484  2491  2517  2533  2535  2540  2543  2546  2549  2552  2555  2596  2624  2669
        2694  2751  2752  2755  2757  2765  2766  2767  2786  2787  2794  2810  2811  2812  2816
        2821  2843  2851  2866  2882  2892  2893  2897  2919  2920  2921  2926  2927  3003  3008
        3031  3032  3034  3036  3049  3056  3062  3063  3064  3084  3088  3094  3113  3184  3198
        3212  3221  3226  3228  3246  3255  3261  3276  3278  3318  3362  3386  3393  3395  3400
        3401  3472  3477  3479  3503  3508  3513  3539  3558  3597  3598  3602  3617  3680  3688
        3689  3690  3696  3703  3721  3722  3723  3724  3725  3750  3751  3752  3753  3754  3755
        3756  3757  3758  3759  3766  3767  3768  3769  3770  3771  3772  3773  3774  3775  3802
        3826  3831  3860  3861  3888  3889  3904  3905  3906  3907  3928  3929  3933  3961  3962
        3963  3964  3965  3966  3967  3968  3969  3970  3971  3977  3978  3982  3983  3984  3985
        3986  3987  3988  3989  3990  3993  4013  4014  4017  4034  4035  4036  4037  4038  4039
        4040  4041  4090  4091  4092  4102  4103  4108
MOVB     427   491   496   511   517   522  1628  1733  1734  2326  2760  2788  2791  2854  2857
        2894  3040  3047  3048  3052  3055  3092  3158  3191  3245  3253  3310  3316  3343  3354
        3372  3377  3467  3469  3471  3485  3526  3550  3551  3555  3569  3599  3627  3635  3681
```

```
        3682    3685    3686    3687    3691    3694    3695    3714    3793    3810    3864    3870    3893    3898    3931
        4054    4097    4099
NEG     3692
NEGB    3054
NOP      861     862     863    1028    1029    1083    1124    1261    1262    1288    1289    1342    1343    1402    1403
        1715
RESET    430     859
ROL     3391    3392    3698    3700    3701    3702    3704
ROLB    3326
ROR     3396    3397
RTI      433    1027    1032    1081    1127    1260    1265    1286    1292    1339    1346    1399    1406    2469    2486
        2569    2597    2603    2615    2625    2756    2795    2898    3035    3173    3604    3726    3760    3776    3832
        3880    3908    3995
RTS      901     903    1626    1637    1708    1729    2164    2379    2505    2575    2632    2641    2696    2710    2728
        2744    2803    2838    2848    2858    2881    2883    2910    2913    2999    3013    3057    3106    3133    3175
        3206    3218    3238    3264    3380    3382    3402    3429    3436    3449    3476    3505    3510    3514    3516
        3527    3529    3540    3542    3559    3575    3644    3934    4018    4060    4104    4111
SBC     4044
SEC     3317    3325
SUB     2534    2678    3473    3552    3556    4043    4045
TRAP    3936    3945    3946    3947    3949    3951    3952    3953    3954    3955    3956
TST      421     442     452     458     478     756     835    1170    1641    1842    1971    2251    2335    2586    2601
        2764    2777    2806    2831    2869    2876    2908    2915    2932    2936    2944    2960    2983    3023    3061
        3098    3104    3107    3151    3159    3323    3381    3430    3444    3464    3486    3501    3536    3547    3601
        3609    3631    3709    3824    3839    3930
TSTB     509     671    1489    1594    1823    1908    2041    2094    2388    2493    2629    2758    3037    3076    3117
        3425    3461    3506    3593    3633    3791    3807    3862    3868    4093
.ASCII  3652    3653    4295    4306    4317    4324
.ASCIZ   881     883    2577    2684    2689    3654    3912    3913    3914    3916    3999    4115    4119    4128    4133
        4139    4144    4147    4151    4154    4159    4162    4164    4166    4173    4176    4179    4182    4185    4188
        4191    4193    4199    4204    4210    4213    4217    4222    4224    4227    4230    4233    4238    4240    4244
        4247    4250    4254    4258    4260    4262    4264    4268    4272    4277    4285    4293    4302    4313    4332
.BLKB   3911    4081    4345
.BYTE    475     476     493     494     498     499     519     520     524     525    2464    2465    2481    2482    2537
        2538    2680    2681    2845    2846    3005    3006    3200    3201    3214    3215    3223    3224    3230    3231
        3257    3258    3648    3649    3650    3651    3727    3728    3729    3730    3909    3910    3918    3919
.DSABL  3849
.ENABL     4    3782
.END    4347
.ENDC     16      88     180     194     240     252     253     254     265     266     267     272     273     274     293
         294     295     386     397     423     429     476     477     568     652     689     753     931     956     985
        1014    1045    1063    1089    1123    1137    1166    1208    1227    1338    1345    1398    1405    1447    1469
        1506    1531    1563    1584    1653    1671    1687    1705    1751    1794    1862    1894    1928    1947    1998
        2028    2211    2229    2287    2303    2351    2374    2465    2466    2482    2483    2510    2511    2515    2516
        2652    2666    2713    2731    2846    2847    3006    3007    3141    3201    3202    3215    3216    3224    3225
        3231    3232    3258    3259    3273    3296    3305    3330    3338    3349    3358    3367    3422    3484    3499
        3519    3532    3579    3599    3658    3735    3780    3783    3785    3813    3849    3853    3881    3882    3889
        3891    3894    3896    3912    3920    3923    3929    3932    3944    3945    3946    3947    3948    3949    3950
        3951    3952    3953    3954    3955    3956    3960    3969    3970    3976    3982    3983    3993    3995    4002
        4005    4023    4084    4341
.EQUIV    88      89      97     112     113     142     143     144     145     146     147     148     149     150     151
         170     171     172     173     174     175     176     177     178     179
.EVEN     887    2578    2691    4001    4338
.IF        12      86     152     180     240     251     252     253     264     265     266     271     272     273     292
         293     294     385     396     420     423     475     476     565     649     686     750     928     953     982
        1011    1042    1060    1086    1120    1134    1163    1205    1224    1337    1344    1397    1404    1444    1466
```



```
        1503    1528    1560    1581    1650    1668    1684    1702    1748    1791    1859    1891    1925    1944    1995
        2025    2208    2226    2284    2300    2348    2371    2464    2465    2481    2482    2509    2510    2514    2515
        2649    2663    2712    2730    2845    2846    3005    3006    3140    3200    3201    3214    3215    3223    3224
        3230    3231    3257    3258    3272    3295    3304    3329    3337    3348    3357    3366    3421    3483    3498
        3518    3531    3578    3599    3657    3734    3779    3783    3784    3785    3813    3852    3853    3881    3889
        3890    3894    3895    3911    3912    3918    3922    3928    3932    3936    3945    3946    3947    3948    3949
        3951    3952    3953    3954    3955    3956    3959    3969    3970    3975    3982    3983    3991    3993    3995
        3999    4004    4022    4083    4340
.IFF      88     252     253     254     265     266     267     272     273     274     293     294     295     386     397
         476    1337    1344    1397    1404    2464    2465    2481    2482    2510    2511    2515    2516    2713    2731
        2846    3005    3006    3141    3201    3215    3224    3231    3258    3273    3296    3305    3330    3338    3349
        3358    3367    3422    3484    3499    3519    3532    3579    3658    3735    3780    3785    3853    3855    3860
        3881    3882    3891    3895    3912    3923    3929    3960    3976    3993    4005    4023    4084    4341
.IFT    3855    3860
.IFTF   3800    3853    3856
.IIF      11      16      21      22     474    2462    2479    2844    3004    3199    3213    3222    3229    3256    3646
        3647    3648    3649    3650    3651    3652    3653    3654    3655    3780    3781    3803    3904    3912    3918
        3919    3920    3944    3945    3946    3947    3949    3951    3952    3953    3954    3955    3956
.IRP     240    3750    3770    3963    3969    3982    3983
.LIST      3     194     212     240     564     653     685     754     927     957     981    1015    1041    1064    1085
        1124    1133    1167    1204    1228    1443    1470    1502    1532    1559    1585    1649    1672    1683    1706
        1747    1795    1858    1895    1924    1948    1994    2029    2207    2230    2283    2304    2347    2375    2648
        2667    3881    3936    3944    3945    3946    3947    3948    3949    3950    3951    3952    3953    3954    3955
        3956    3957
.MACRO    246     248    3936
.MCALL      5       6       7     194
.NLIST      2     194     212     240     564     653     685     754     927     957     981    1015    1041    1064    1085
        1124    1133    1167    1204    1228    1443    1470    1502    1532    1559    1585    1649    1672    1683    1706
        1747    1795    1858    1895    1924    1948    1994    2029    2207    2230    2283    2304    2347    2375    2648
        2667    3881    3936    3944    3945    3946    3947    3948    3949    3950    3951    3952    3953    3954    3955
        3956    3957
.PAGE      43      83    2507
.REPT     212     565     649     686     750     928     953     982    1011    1042    1060    1086    1120    1134    1163
        1205    1224    1444    1466    1503    1528    1560    1581    1650    1668    1684    1702    1748    1791    1859
        1891    1925    1944    1995    2025    2208    2226    2284    2300    2348    2371    2649    2663
.SBTTL     84     249     269     295     346     398     420     533     904    1066    1304    1362    1422    1608    1709
        1717    1798    1805    1817    1897    2032    2085    2168    2273    2311    2380    2402    2417    2431    2439
        2507    2582    2699    2746    2885    3016    3156    3265    3408    3451    3543    3576    3655    3732    3777
        3920    3936    3957    4002    4020    4112
.TITLE     11
.WORD     211     214     215     216     217     234     377     378    1178    3090    3643    3646    3647    3731    3780
        3781    3992    3994    4019    4056
```

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

*DZRXBE,DZRXBE/CRF/SOL=DZRXBE
RUN-TIME:  40 29 5 SECONDS
RUN-TIME RATIO:  141/76=1.8
CORE USED:  18K  (35 PAGES)
```