RX11

RX11 DIAGNOSTIC

MD-11-DZRXB-D

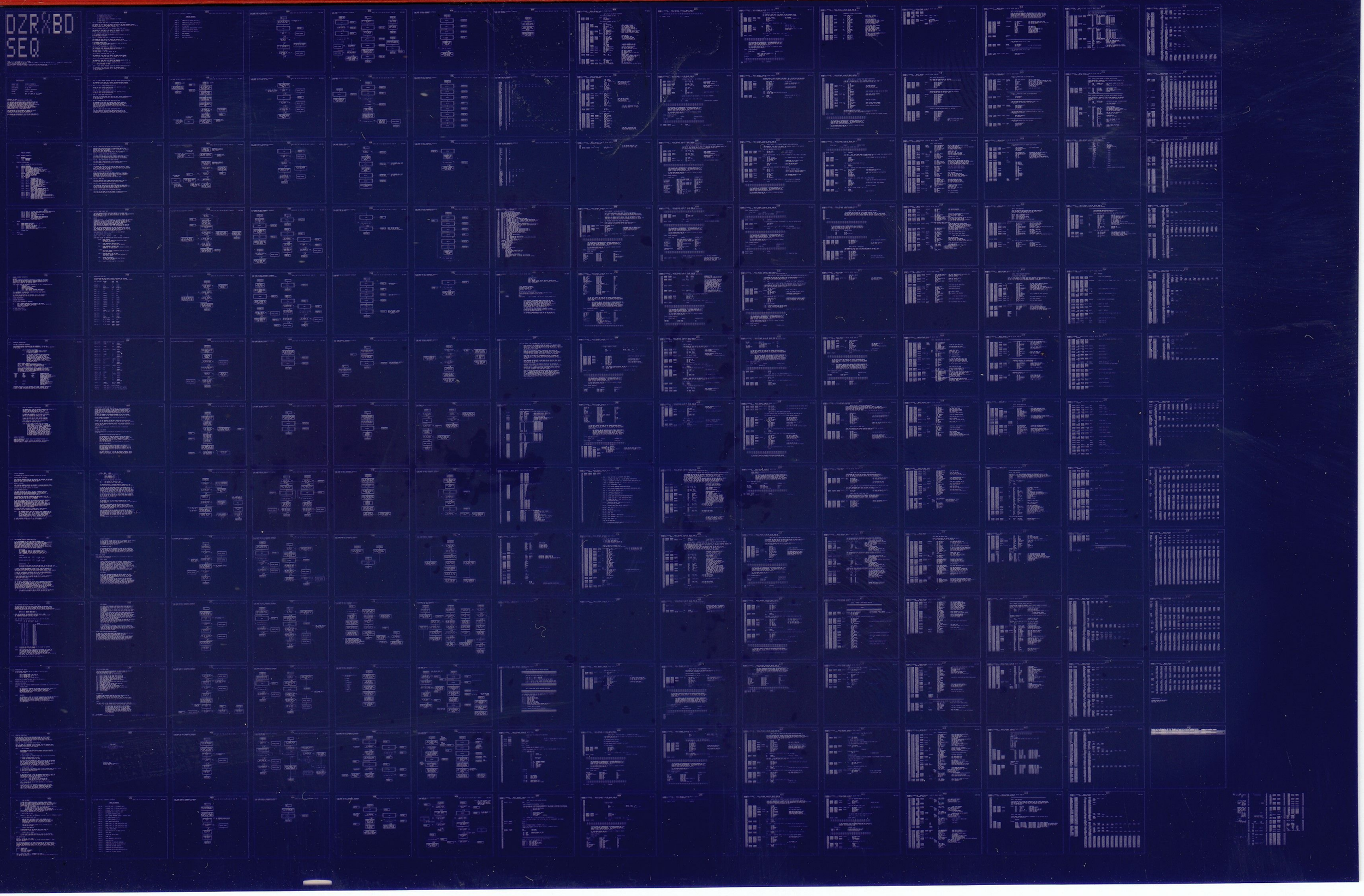
EP-DZRXB-D-DL-A

FEB 1976

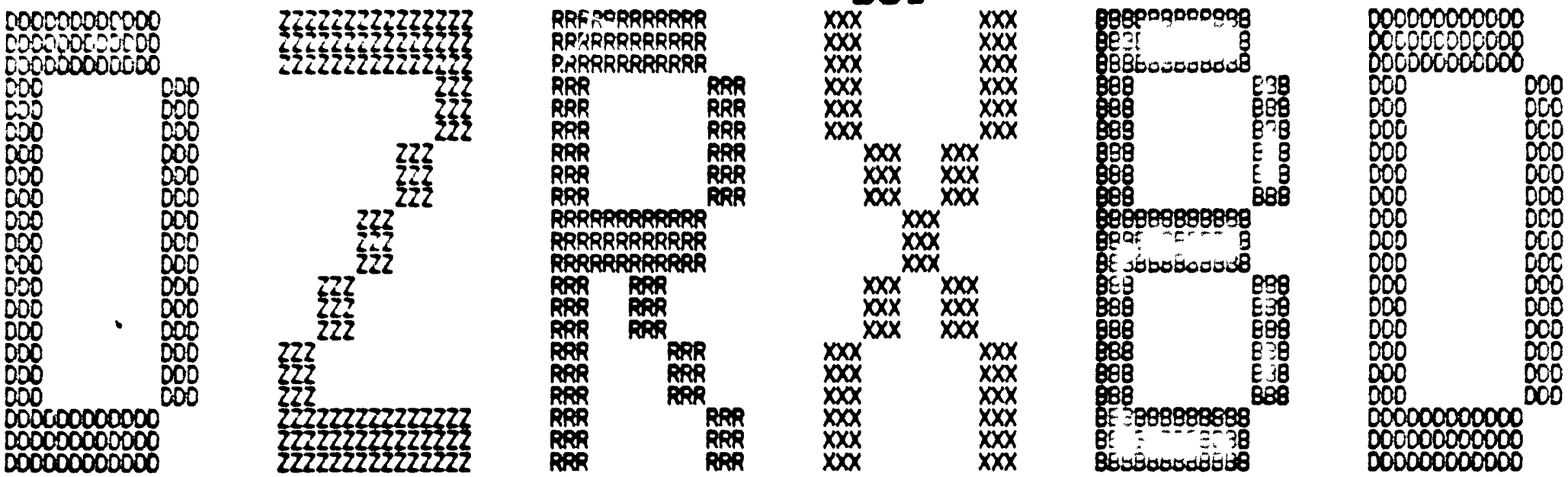
FICHE 1 OF 1

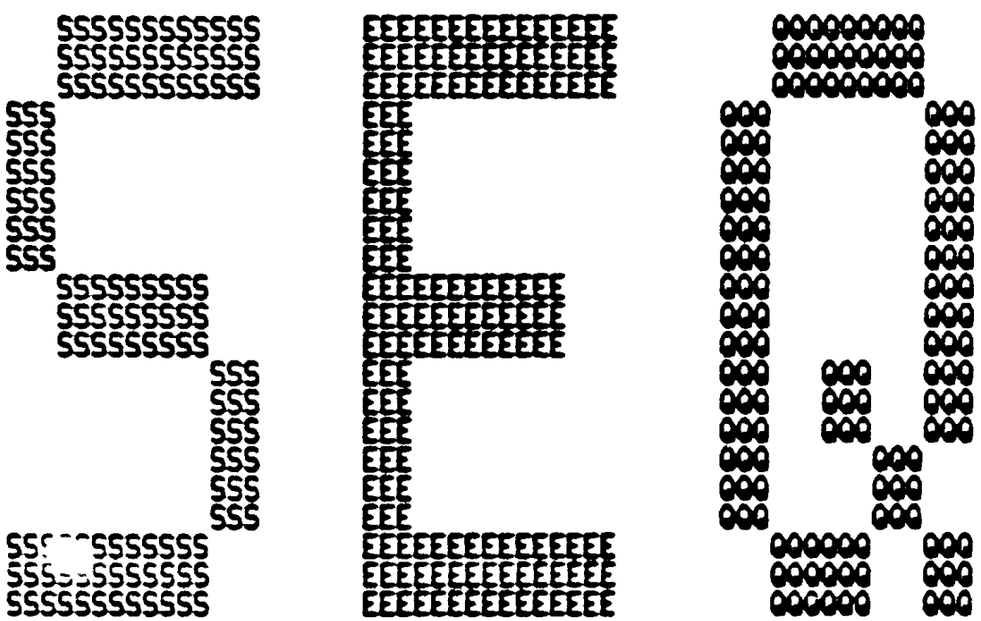
digital

Made In U.S.A.



B01





LPTSPL Version 101(2107) Running on MTA140
 START User WELLINGHAM,A[1,2] Job DZRXB0 Seq. 3592 Date 17-Jan-76 18:27:33 Monitor RV22SD KI10 SYS#514 *START*
Request created: 17-Jan-76 17:57:10
File: WHTG:DZRXB0.SEQ<055>[404,3722] Created: 09-Jan-76 16:12:28 Printed: 17-Jan-76 18:27:34
QUEUE Switches: /FILE:ASCII /COPIES:1 /SPACING:1 /LIMIT:410 /FORMS:NORMAL

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZRXB-D-D

PRODUCT NAME: RX11 INTERFACE DIAGNOSTIC

DATE CREATED: DEC 21,1975

MAINTAINER: DIAGNOSTIC ENGINEERING

AUTHOR: DAVID L. ADAMS

REVISED: SEPT. 12, 1975 BY D. L. ADAMS
NOV. 10, 1975 BY B. BURGESS

TABLE OF CONTENTS

1.0 GENERAL PROGRAM INFORMATION SEQ 0004

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].
C. CONSOLE TELEPRINTER

1.2.2 SOFTWARE REQUIREMENTS

NO PREREQUISITE SOFTWARE

2.0 OPERATING INSTRUCTIONS SEQ 0005

2.0.1 OUTLINE OF OPERATING PROCEDURE

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

A. LOAD THE PROGRAM INTO MEMORY
1. IF IT IS BEING LOADED FROM A DISKETTE REPLACE THE "LIBRARY" DISKETTE WITH A "SCRATCH" DISKETTE.

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

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

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

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

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

B. START THE PROGRAM AT LOCATION 200

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

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

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

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

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

2.1 LOADING THE PROGRAM

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

2.2 STARTING ADDRESSES

THE PROGRAM HAS TWO STARTING ADDRESS LOCATIONS AS FOLLOWS:

2.2.1 INITIAL START [LOC.200]

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

2.2.2 RESTART [LOC.202]

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

2.3 OPERATOR ACTION BEFORE STARTING THE PROGRAM

2.3.1 DEVICE ADDRESS SELECTION

LIKE MOST OPTIONS ON THE PDP-11 THE RX11 INTERFACE CARD HAS JUMPERABLE REGISTER AND VECTOR ADDRESSES. THIS ALLOWS FOR DEVICES WITH THE SAME STANDARD ADDRESSES TO BE JUMPERED TO AN OTHER ADDRESS SO THEY WILL RUN WITHOUT CONFLICT.

THE PROGRAM MUST KNOW WHAT ADDRESSES ARE BEING USED, AS IT IS THROUGH THESE REGISTER AND VECTOR ADDRESSES THAT ALL COMMUNICATION BETWEEN THE PDP-11 AND THE RX11 IS HANDLED.

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

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

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

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

2.3.2 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	6----0
		ID	OD
SECTORS LOCATION 1202	BITS	12----8	4----0
		LAST	FIRST

C. RESTRICTIONS

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

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

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

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

2.3.3 SOFTWARE SWITCH REGISTER (LOC. 176)

FOR THE PDP 11 PROCESSORS THAT DO NOT HAVE A HARDWARE SWITCH REGISTER OR IF THE OPERATOR WISHES TO SELECT THE SOFTWARE SWITCH REGISTER BY PUTTING ALL THE HARDWARE SWITCHES UP TO A "1", 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.

NOTE: THE LOCATION 176 MUST BE SET UP BEFORE THE START OF THE PROGRAM AS THERE IS NO DYNAMIC CHANGE OF THIS LOCATION AVAILABLE.

2.3.4 TEST PARAMETER SELECTION ("DTESTP" LOC. 1212) SEQ 0009

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
					(IF NO TEST SELECTED DEFAULTS TO TEST 1)
0	0	0	0	0	
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 SEL :CTED 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: SEQ 0010

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

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

SEE SECTION 2.5 UNDER THE ABOVE TESTS FOR EXPLANATION

2.4 OPERATOR ACTION TO RUN THE PROGRAM

2.4.1 STARTING THE PROGRAM

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

SA200 (INITIAL START)

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

SA202 (RESTART)

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

2.4.2 OPERATING CONDITIONS

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

SW15-SW0 (1) - SELECT SOFTWARE SWITCH REGISTER

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

SW15 (1) - HALT ON ERROR

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

SW14 (1) - HALT AT END OF PASS

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

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

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

SW13 (1) - INHIBIT ERROR TYPEOUT

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

SW12 (1) - LOOP ON TEST

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

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

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

SW11 (1) - LOCK ON ERROR

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

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

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

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

SW10 (1) - HALT AT END OF TEST

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

SW 9 - LIMIT DATA ERROR PRINT OUTS

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

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

SW 8 (1) - INHIBIT RECALIBRATION

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

SW 0 (1) - INHIBIT BELL AT ERROR

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

2.5 TEST DEFINITIONS

2.5.1 PRETEST - INITIALIZE (KEY) PART I

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

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

PRETEST INSURES THAT:

A. DONE IS SET

B. ERROR FLAG IS CLEARED

C. TR FLAG IS CLEARED

D. INIT DONE IS SET

2.5.2 TEST 1 - RXCS TEST PART I / INTERRUPT TEST PART I

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

THE PROGRAM WRITES THE RXCS = 0

NO INTERRUPTS SHOULD OCCUR

THE RXCS SHOULD REMAIN UNCHANGED = 40 (DONE)

THE RXDB SHOULD = 0

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

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

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

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

AN RXDI 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 RXDI 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 RXDI 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 RXD1, 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

G02

TEST 11&12 (1)	[USES TEST 6&7 TO FILL / EMPTY BUFFER]		SEQ 0018
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 SEQ 0019

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 RXDI 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 SS 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 RXD1 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, CONTEN , 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 SEQ 0023

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

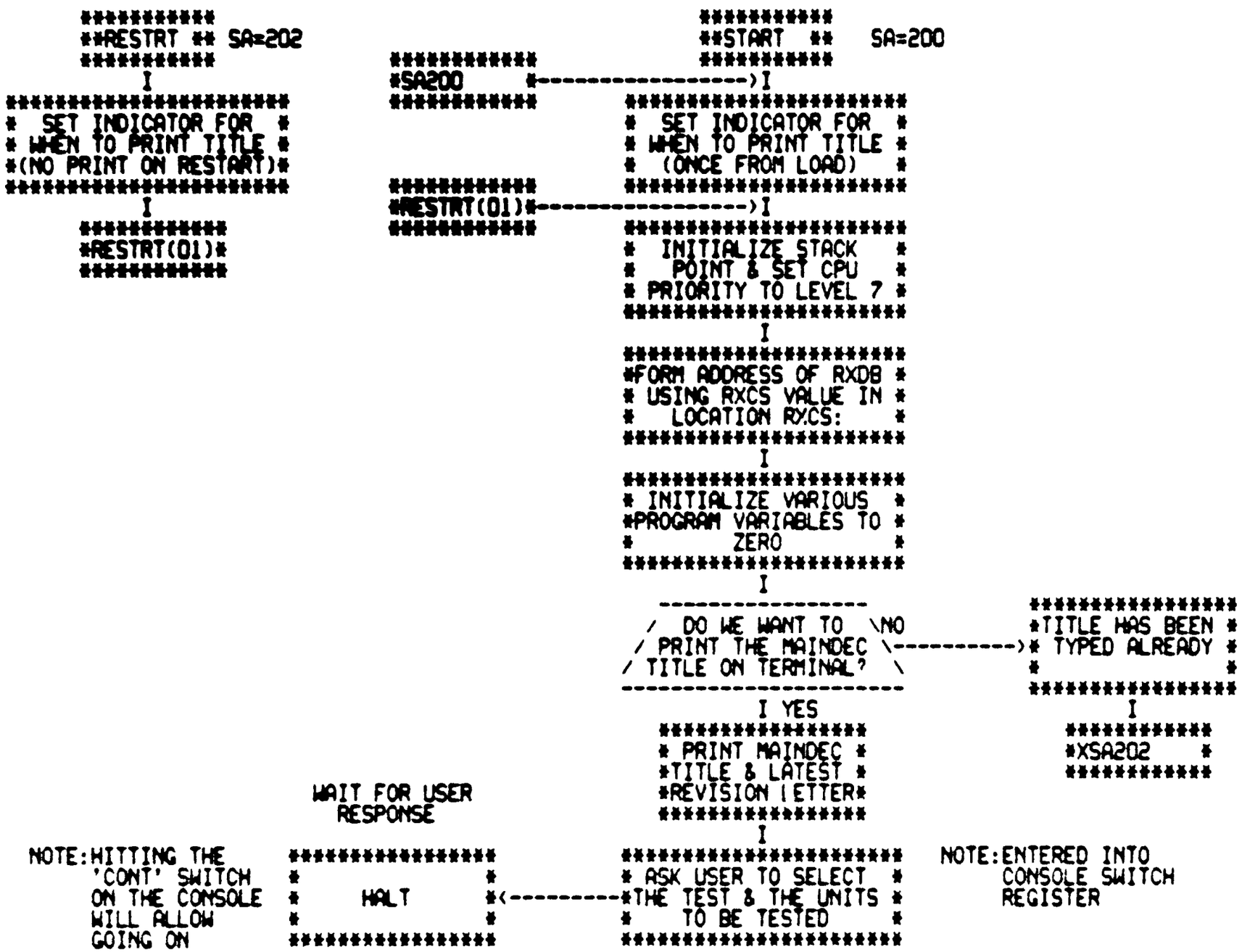
FLOW CHART
****

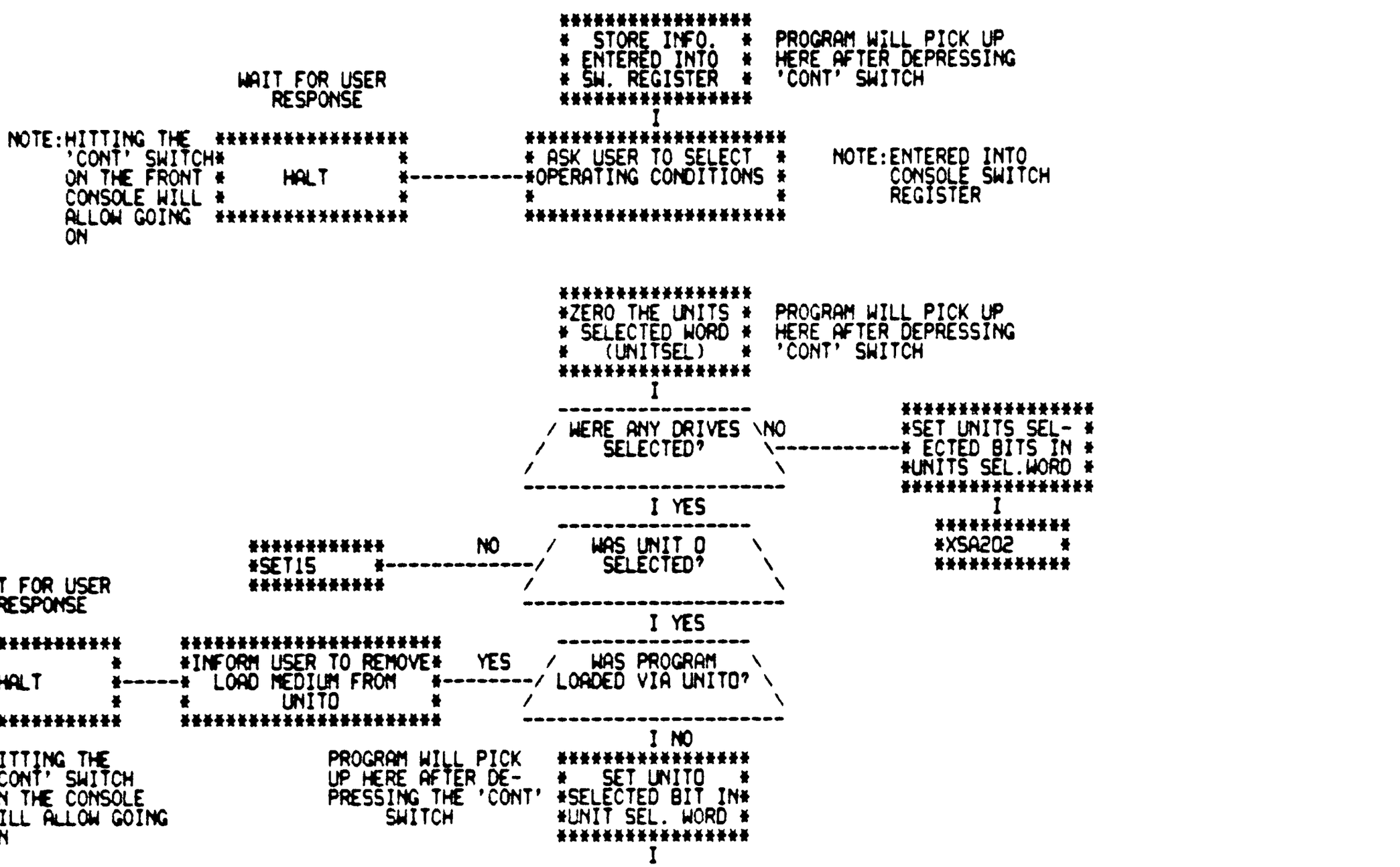
FLOW CHART FOR RX11 DIAGNOSTIC INTERFACE

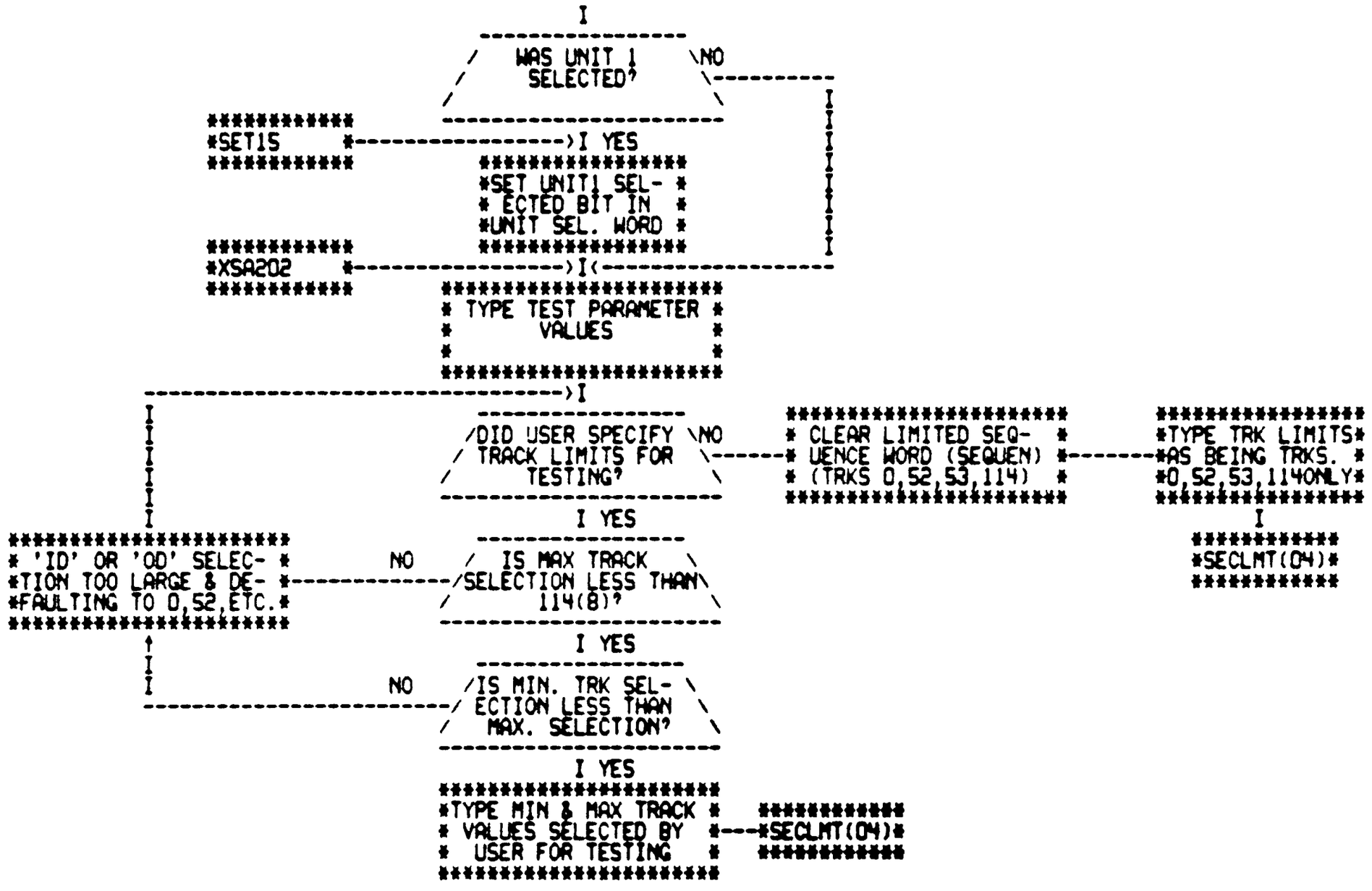
****

TABLE OF CONTENTS
****

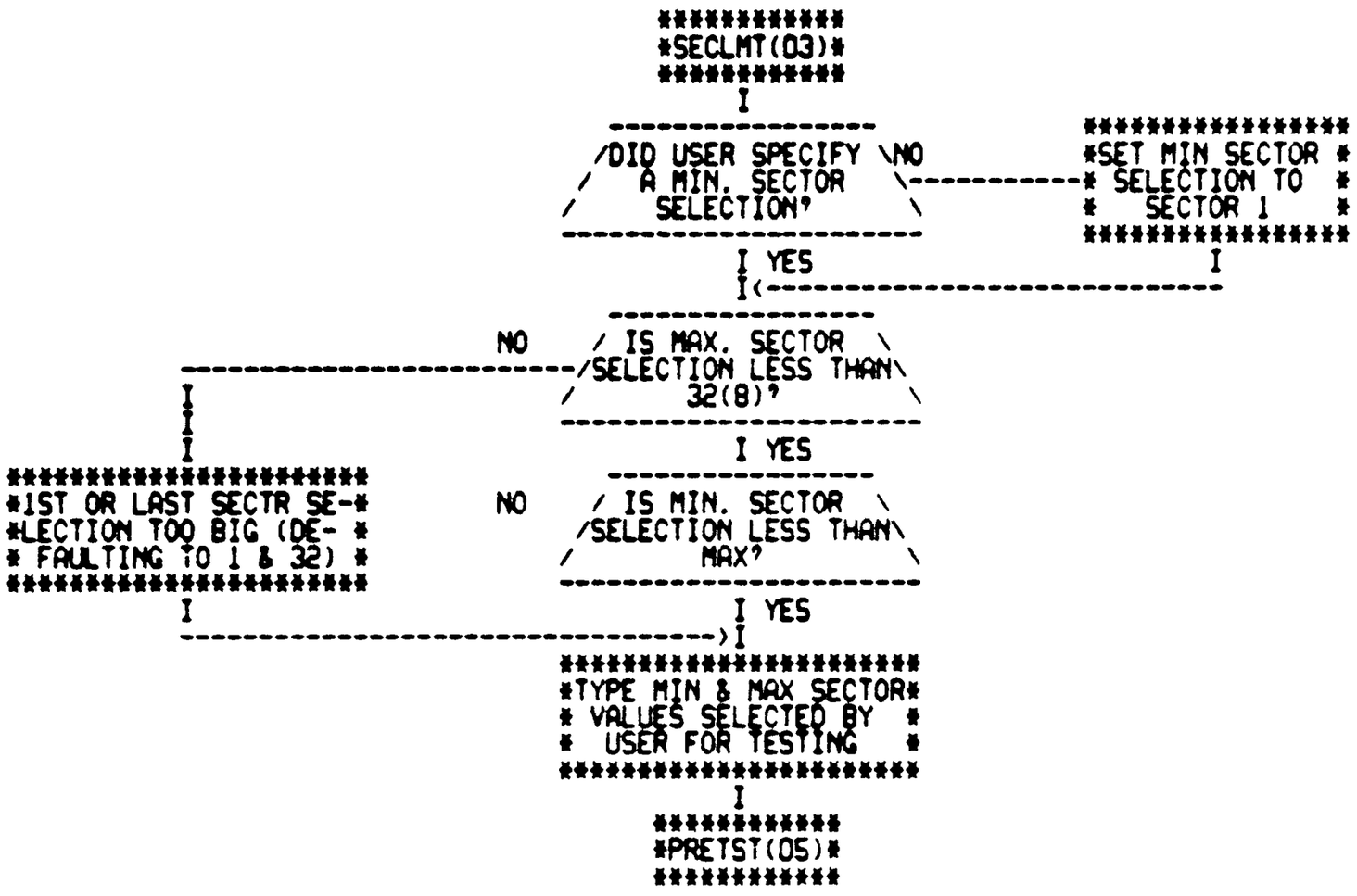
TABLE OF CONTENTS
****

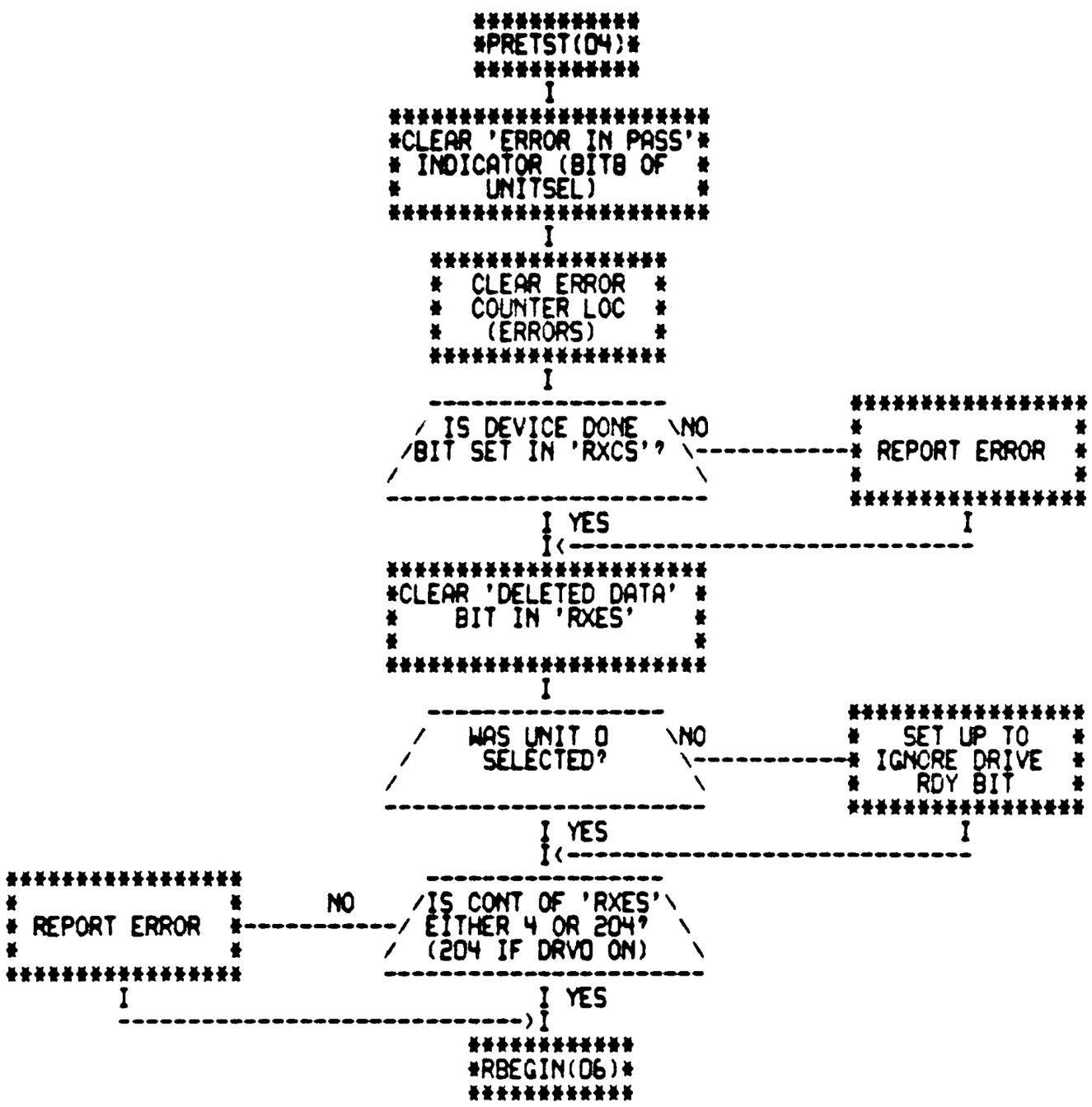

**RESTRT ** SA=202
#RESTRT(01)#
SET INDICATOR FOR WHEN TO PRINT TITLE (NO PRINT ON RESTART)
INITIALIZE STACK POINT & SET CPU PRIORITY TO LEVEL 7
FORM ADDRESS OF RXDB USING RXCS VALUE IN LOCATION RXCS:
INITIALIZE VARIOUS PROGRAM VARIABLES TO ZERO
DO WE WANT TO PRINT THE MAINDEC TITLE ON TERMINAL?
NO
YES
TITLE HAS BEEN TYPED ALREADY
#XSA202
#SA200
**START ** SA=200
#RESTRT(01)#
SET INDICATOR FOR WHEN TO PRINT TITLE (ONCE FROM LOAD)
PRINT MAINDEC TITLE & LATEST REVISION LETTER
ASK USER TO SELECT THE TEST & THE UNITS TO BE TESTED
WAIT FOR USER RESPONSE
HALT
NOTE: HITTING THE 'CONT' SWITCH ON THE CONSOLE WILL ALLOW GOING ON
NOTE: ENTERED INTO CONSOLE SWITCH REGISTER

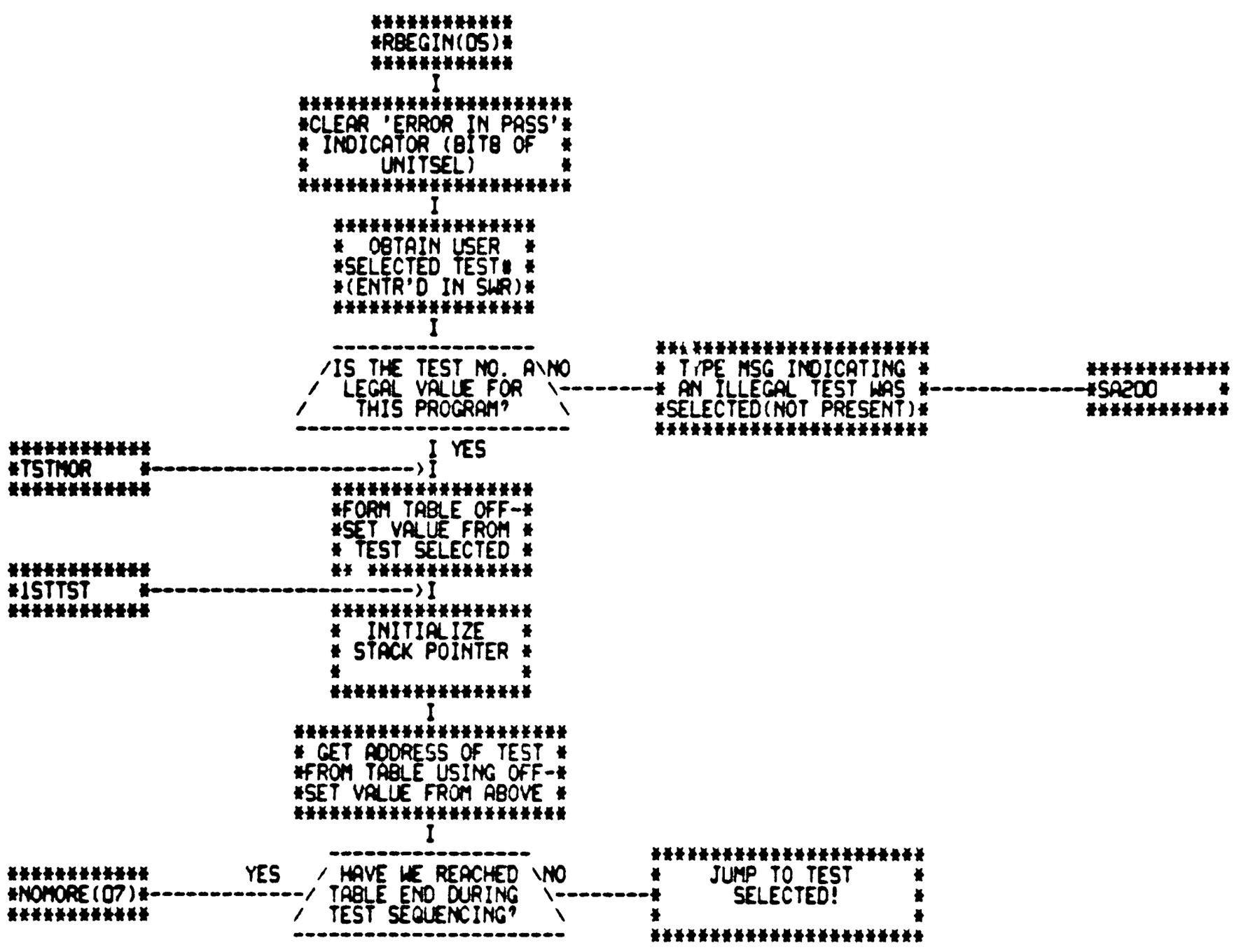

STORE INFO. ENTERED INTO SW. REGISTER
PROGRAM WILL PICK UP HERE AFTER DEPRESSING 'CONT' SWITCH
WAIT FOR USER RESPONSE
NOTE:HITTING THE 'CONT' SWITCH ON THE FRONT CONSOLE WILL ALLOW GOING ON
HALT
ASK USER TO SELECT OPERATING CONDITIONS
NOTE:ENTERED INTO CONSOLE SWITCH REGISTER
ZERO THE UNITS SELECTED WORD (UNITSEL)
PROGRAM WILL PICK UP HERE AFTER DEPRESSING 'CONT' SWITCH
WERE ANY DRIVES SELECTED?
NO
SET UNITS SEL- ECTED BITS IN UNITS SEL.WORD
XSA202
I YES
WAS UNIT 0 SELECTED?
NO
SET15
WAIT FOR USER RESPONSE
I YES
WAS PROGRAM LOADED VIA UNIT0?
YES
INFORM USER TO REMOVE LOAD MEDIUM FROM UNIT0
HALT
I NO
NOTE:HITTING THE 'CONT' SWITCH ON THE CONSOLE WILL ALLOW GOING ON
PROGRAM WILL PICK UP HERE AFTER DE- PRESSING THE 'CONT' SWITCH
SET UNIT0 SELECTED BIT IN UNIT SEL. WORD
I

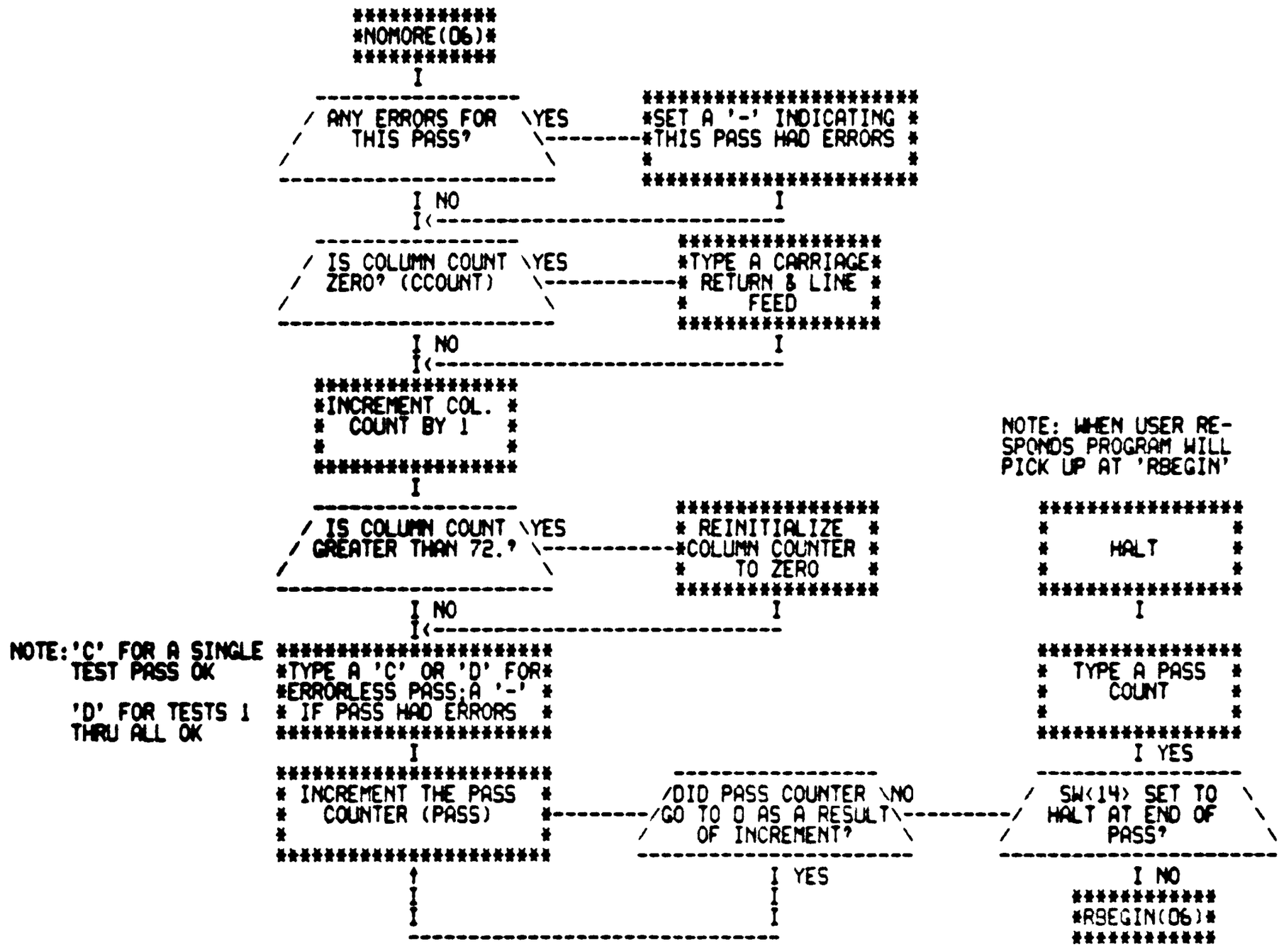

WAS UNIT 1 SELECTED?
NO
YES
*SET15
*XSA202
*SET UNIT1 SEL- ECTED BIT IN UNIT SEL. WORD
TYPE TEST PARAMETER VALUES
DID USER SPECIFY TRACK LIMITS FOR TESTING?
NO
YES
* CLEAR LIMITED SEQUENCE WORD (SEQUEN) (TRKS 0,52,53,114)
*TYPE TRK LIMITS AS BEING TRKS. 0,52,53,114ONLY
*SECLMT(04)
'ID' OR 'OD' SELECTION TOO LARGE & DEFAULTING TO 0,52,ETC.
IS MAX TRACK SELECTION LESS THAN 114(8)?
NO
YES
IS MIN. TRK SELECTION LESS THAN MAX. SELECTION?
NO
YES
TYPE MIN & MAX TRACK VALUES SELECTED BY USER FOR TESTING
*SECLMT(04)

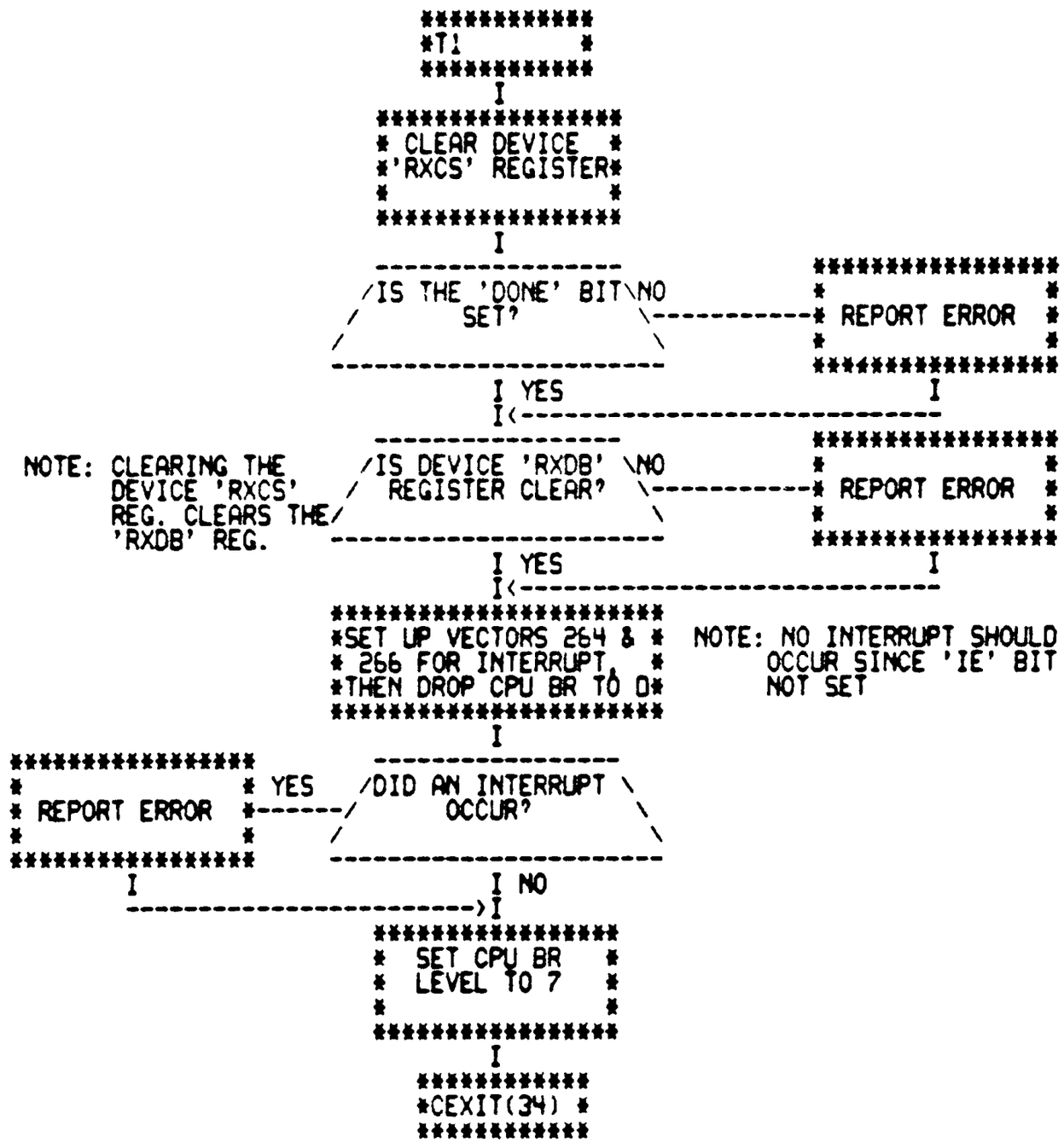
F03

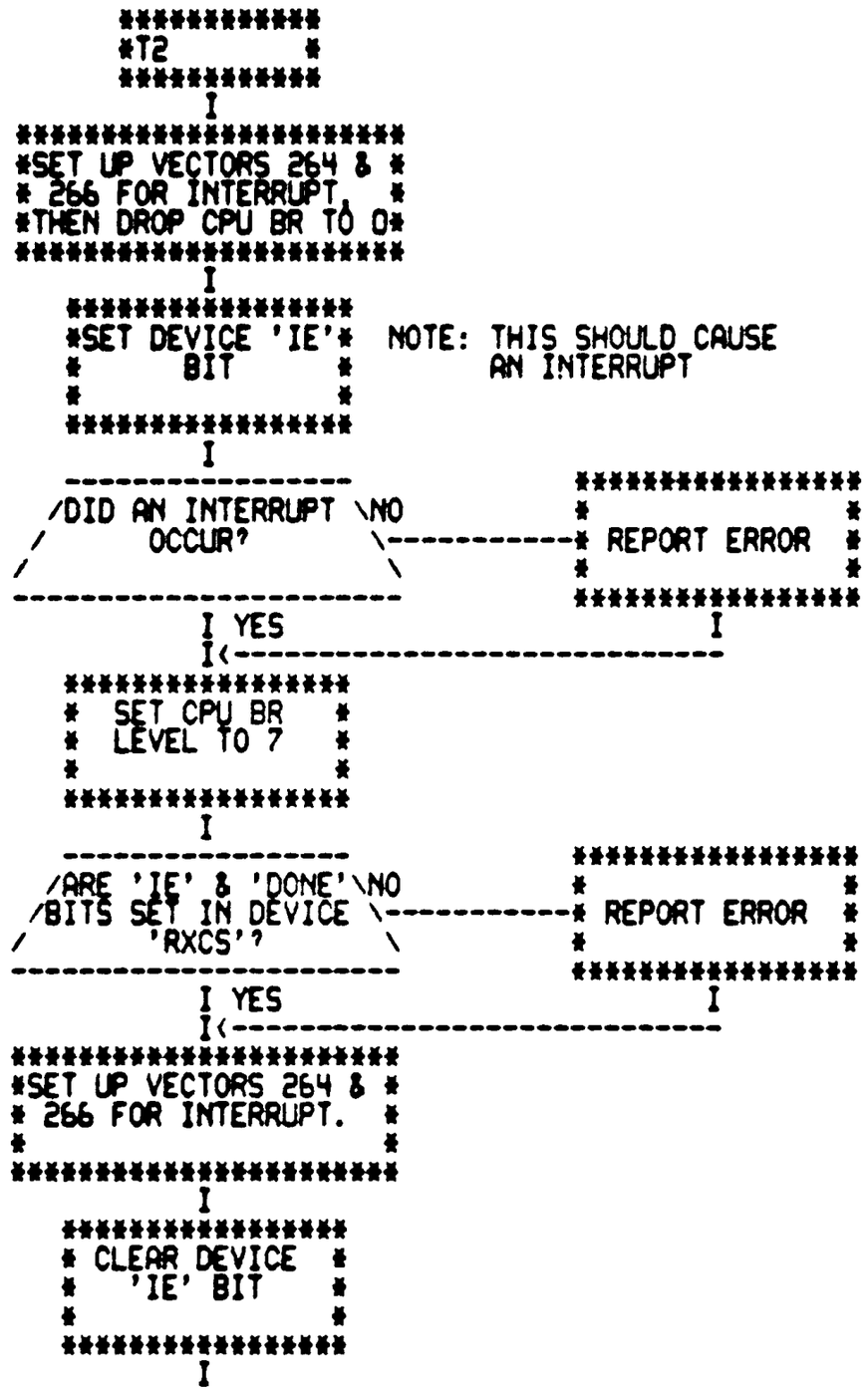

SECLMT(03)
/DID USER SPECIFY \NO
/ A MIN. SECTOR \
/ SELECTION? \
*SET MIN SECTOR *
* SELECTION TO *
* SECTOR 1 *
I YES
/ IS MAX. SECTOR \
NO /SELECTION LESS THAN\
/ 32(8)? \
I YES
/ IS MIN. SECTOR \
NO /SELECTION LESS THAN\
/ MAX? \
I YES
1ST OR LAST SECTR SE-
*LECTION TOO BIG (DE- *
* FAULTING TO 1 & 32) *
TYPE MIN & MAX SECTOR
* VALUES SELECTED BY *
* USER FOR TESTING *
PRETST(05)

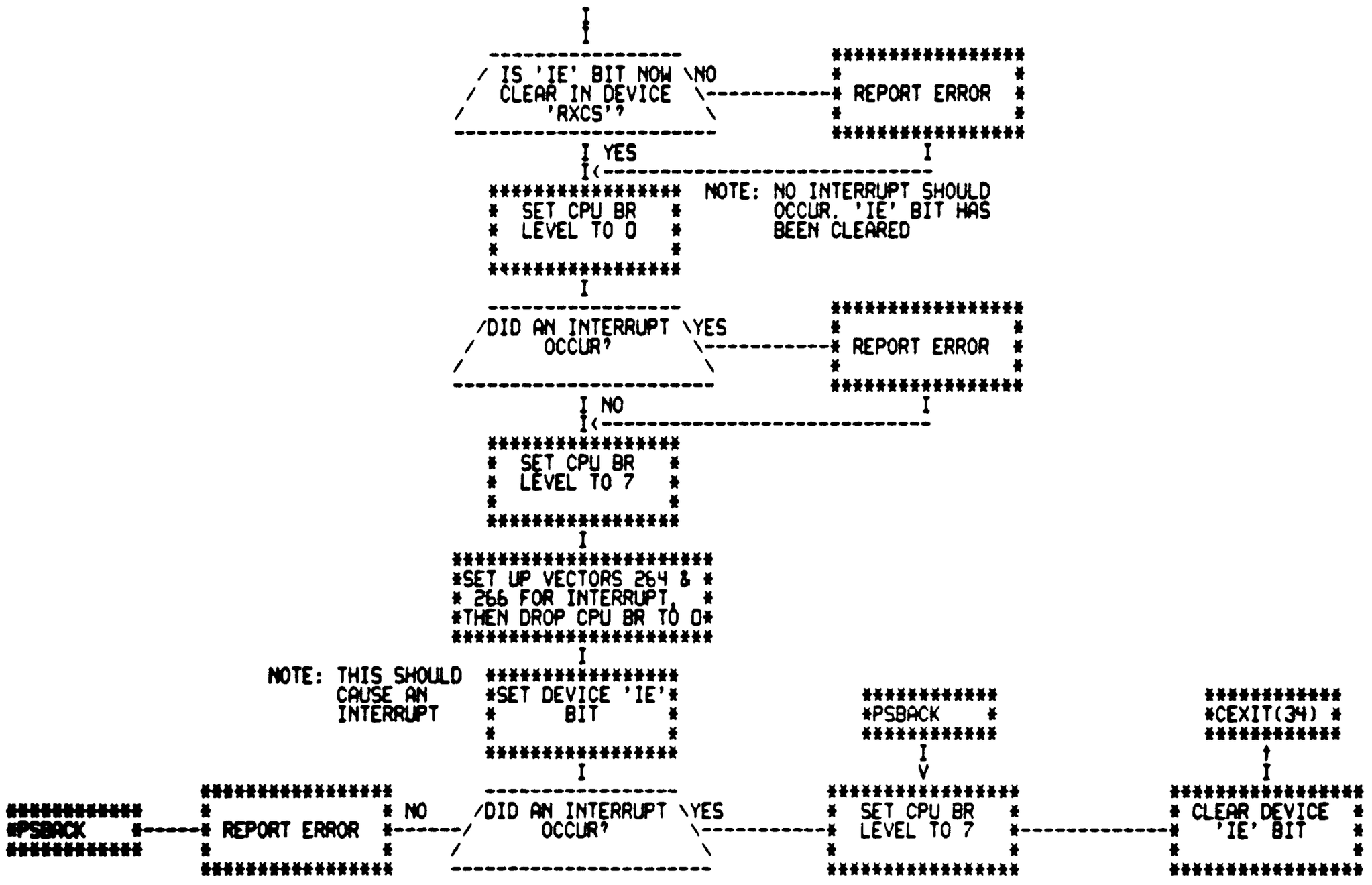

PRETST(04)
CLEAR 'ERROR IN PASS' INDICATOR (BIT8 OF UNITSEL)
CLEAR ERROR COUNTER LOC (ERRORS)
IS DEVICE DONE BIT SET IN 'RXCS'?
NO
REPORT ERROR
YES
CLEAR 'DELETED DATA' BIT IN 'RXES'
WAS UNIT 0 SELECTED?
NO
SET UP TO IGNORE DRIVE RDY BIT
YES
IS CONT OF 'RXES' EITHER 4 OR 204? (204 IF DRV0 ON)
NO
REPORT ERROR
YES
RBEGIN(06)


RBEGIN(05)
CLEAR 'ERROR IN PASS' INDICATOR (BIT8 OF UNITSEL)
OBTAIN USER SELECTED TEST# (ENTR'D IN SWR)
IS THE TEST NO. A LEGAL VALUE FOR THIS PROGRAM?
NO
TYPE MSG INDICATING AN ILLEGAL TEST WAS SELECTED(NOT PRESENT)
SA200
YES
TSTMOR
FORM TABLE OFF-SET VALUE FROM TEST SELECTED
1STTST
INITIALIZE STACK POINTER
GET ADDRESS OF TEST FROM TABLE USING OFF-SET VALUE FROM ABOVE
HAVE WE REACHED TABLE END DURING TEST SEQUENCING?
NO
YES
NOMORE(07)
JUMP TO TEST SELECTED!

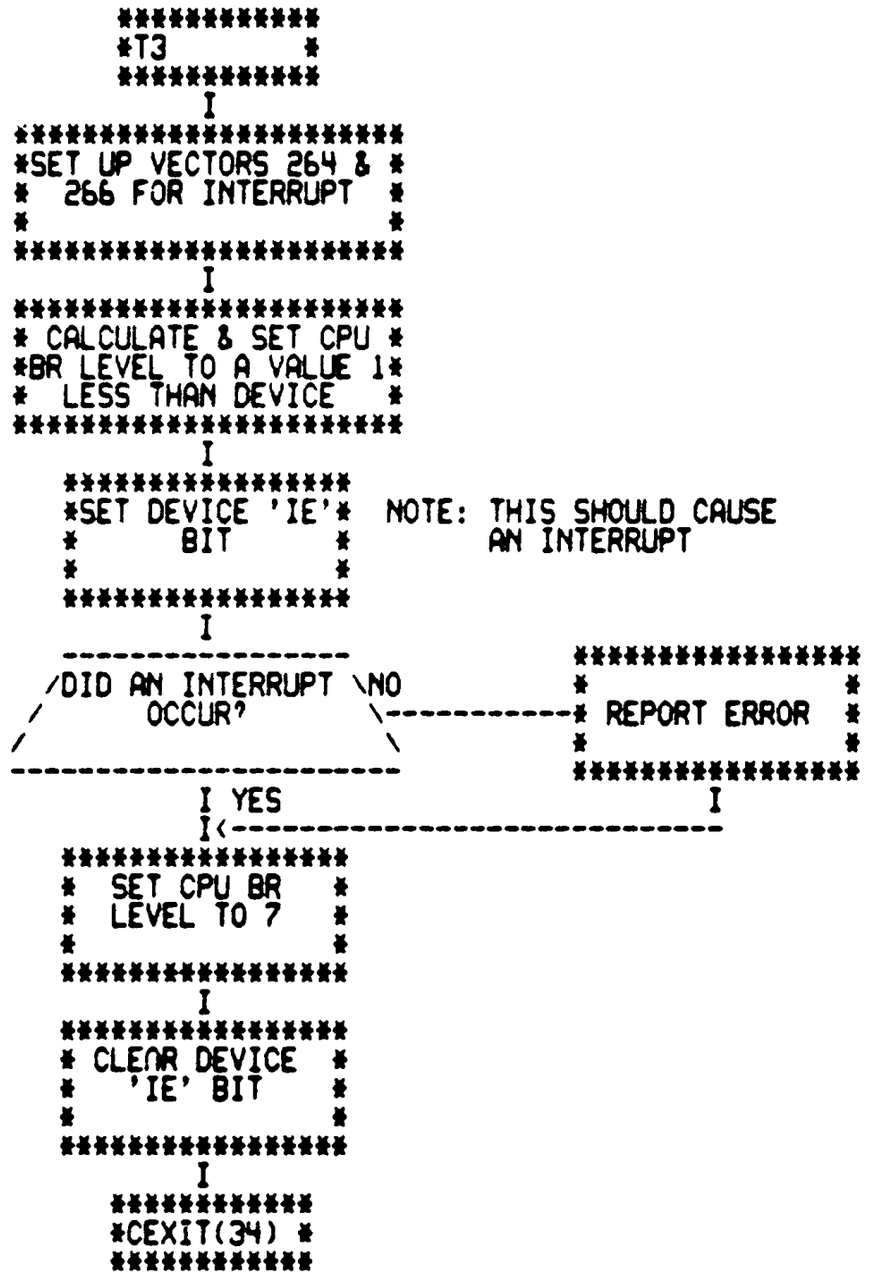

NOMORE(06)
ANY ERRORS FOR THIS PASS?
YES
SET A '-' INDICATING THIS PASS HAD ERRORS
NO
IS COLUMN COUNT ZERO? (CCOUNT)
YES
TYPE A CARRIAGE RETURN & LINE FEED
NO
INCREMENT COL. COUNT BY 1
IS COLUMN COUNT GREATER THAN 72.?
YES
REINITIALIZE COLUMN COUNTER TO ZERO
NO
TYPE A 'C' OR 'D' FOR ERRORLESS PASS;A '-' IF PASS HAD ERRORS
INCREMENT THE PASS COUNTER (PASS)
DID PASS COUNTER GO TO 0 AS A RESULT OF INCREMENT?
YES
HALT
TYPE A PASS COUNT
YES
SW(14) SET TO HALT AT END OF PASS?
NO
RBEGIN(06)
NOTE: WHEN USER RESPONDS PROGRAM WILL PICK UP AT 'RBEGIN'
NOTE:'C' FOR A SINGLE TEST PASS OK
'D' FOR TESTS 1 THRU ALL OK

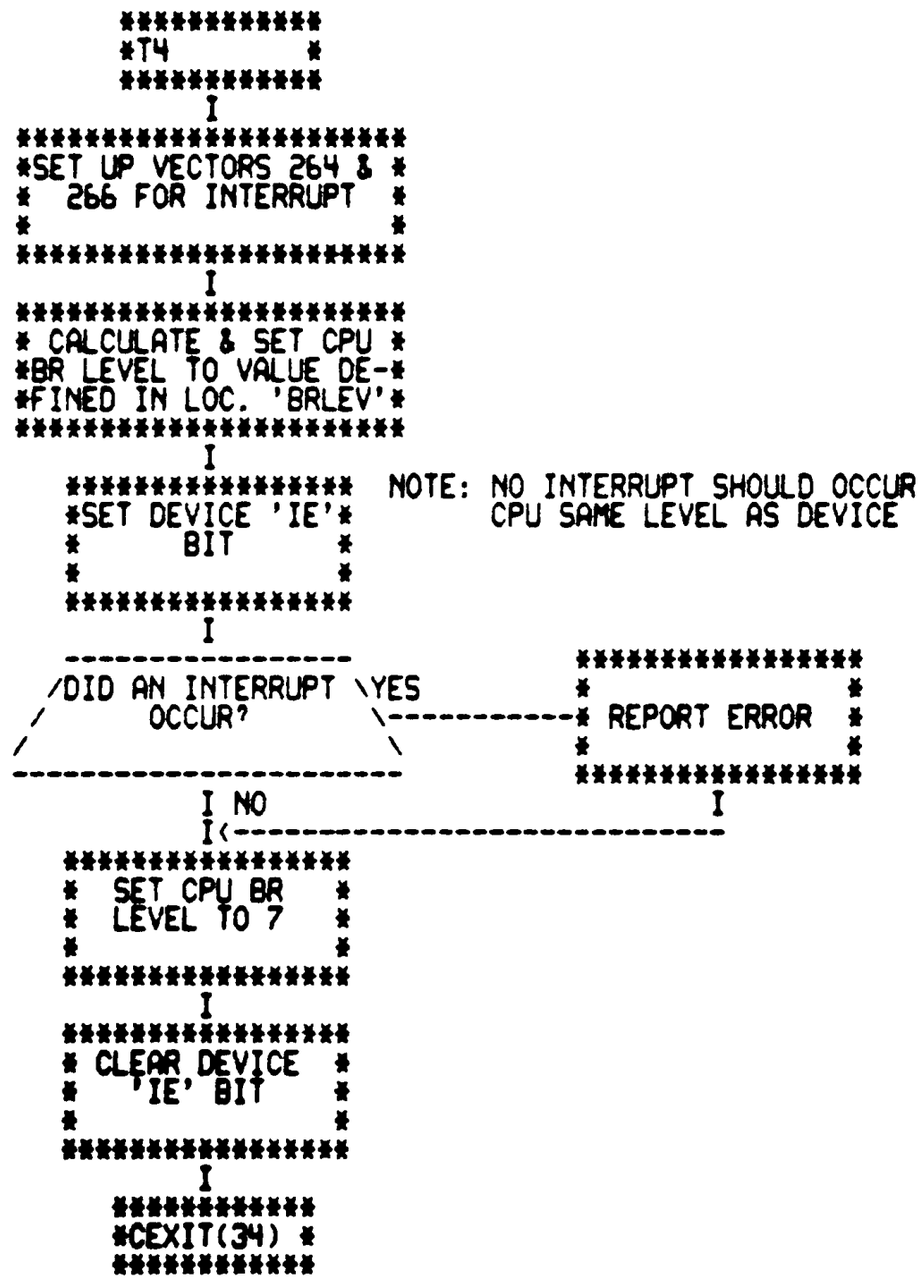

T1
CLEAR DEVICE 'RXCS' REGISTER
IS THE 'DONE' BIT SET?
NO
REPORT ERROR
YES
NOTE: CLEARING THE DEVICE 'RXCS' REG. CLEARS THE 'RXDB' REG.
IS DEVICE 'RXDB' REGISTER CLEAR?
NO
REPORT ERROR
YES
SET UP VECTORS 264 & 266 FOR INTERRUPT, THEN DROP CPU BR TO 0
NOTE: NO INTERRUPT SHOULD OCCUR SINCE 'IE' BIT NOT SET
DID AN INTERRUPT OCCUR?
YES
REPORT ERROR
NO
SET CPU BR LEVEL TO 7
CEXIT(34)

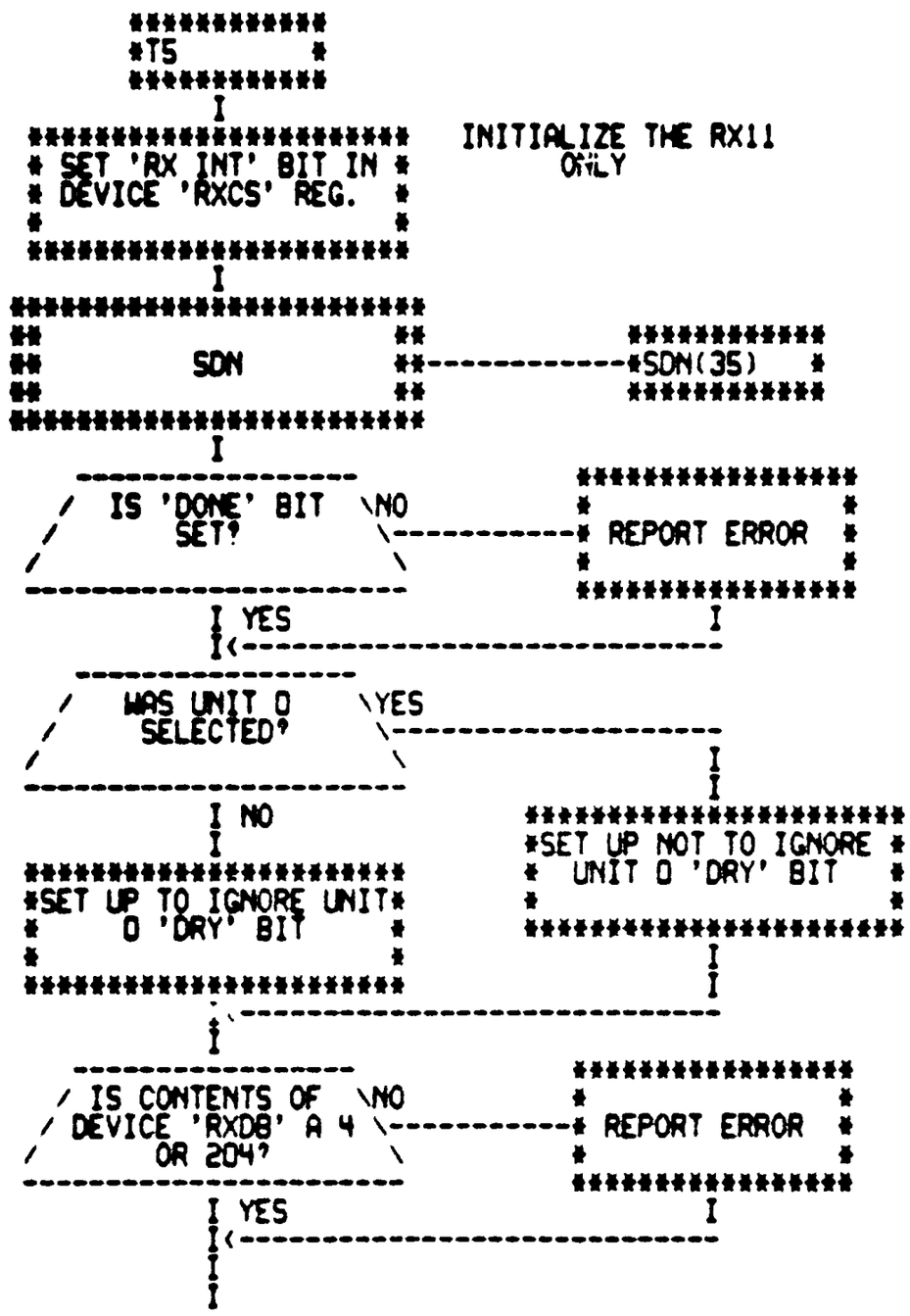

T2
SET UP VECTORS 264 & 266 FOR INTERRUPT, THEN DROP CPU BR TO 0
SET DEVICE 'IE' BIT
NOTE: THIS SHOULD CAUSE AN INTERRUPT
DID AN INTERRUPT OCCUR?
NO
REPORT ERROR
YES
SET CPU BR LEVEL TO 7
ARE 'IE' & 'DONE' BITS SET IN DEVICE 'RXCS'?
NO
REPORT ERROR
YES
SET UP VECTORS 264 & 266 FOR INTERRUPT.
CLEAR DEVICE 'IE' BIT

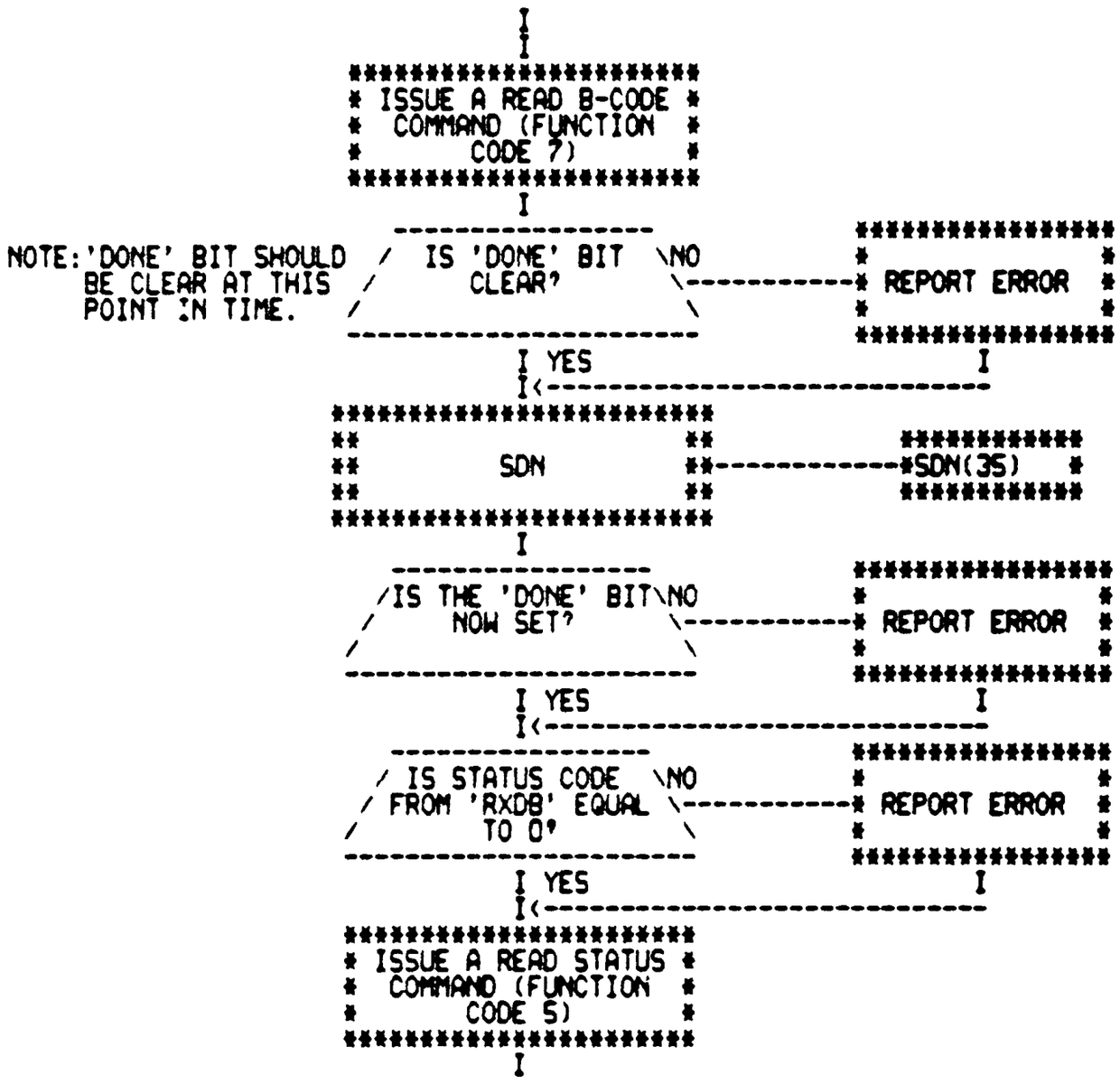

IS 'IE' BIT NOW CLEAR IN DEVICE 'RXCS'?
NO
REPORT ERROR
YES
NOTE: NO INTERRUPT SHOULD OCCUR. 'IE' BIT HAS BEEN CLEARED
SET CPU BR LEVEL TO 0
DID AN INTERRUPT OCCUR?
YES
REPORT ERROR
NO
SET CPU BR LEVEL TO 7
SET UP VECTORS 264 & 266 FOR INTERRUPT, THEN DROP CPU BR TO 0
NOTE: THIS SHOULD CAUSE AN INTERRUPT
SET DEVICE 'IE' BIT
DID AN INTERRUPT OCCUR?
NO
REPORT ERROR
#PSBACK
YES
SET CPU BR LEVEL TO 7
#PSBACK
CLEAR DEVICE 'IE' BIT
CEXIT(34)

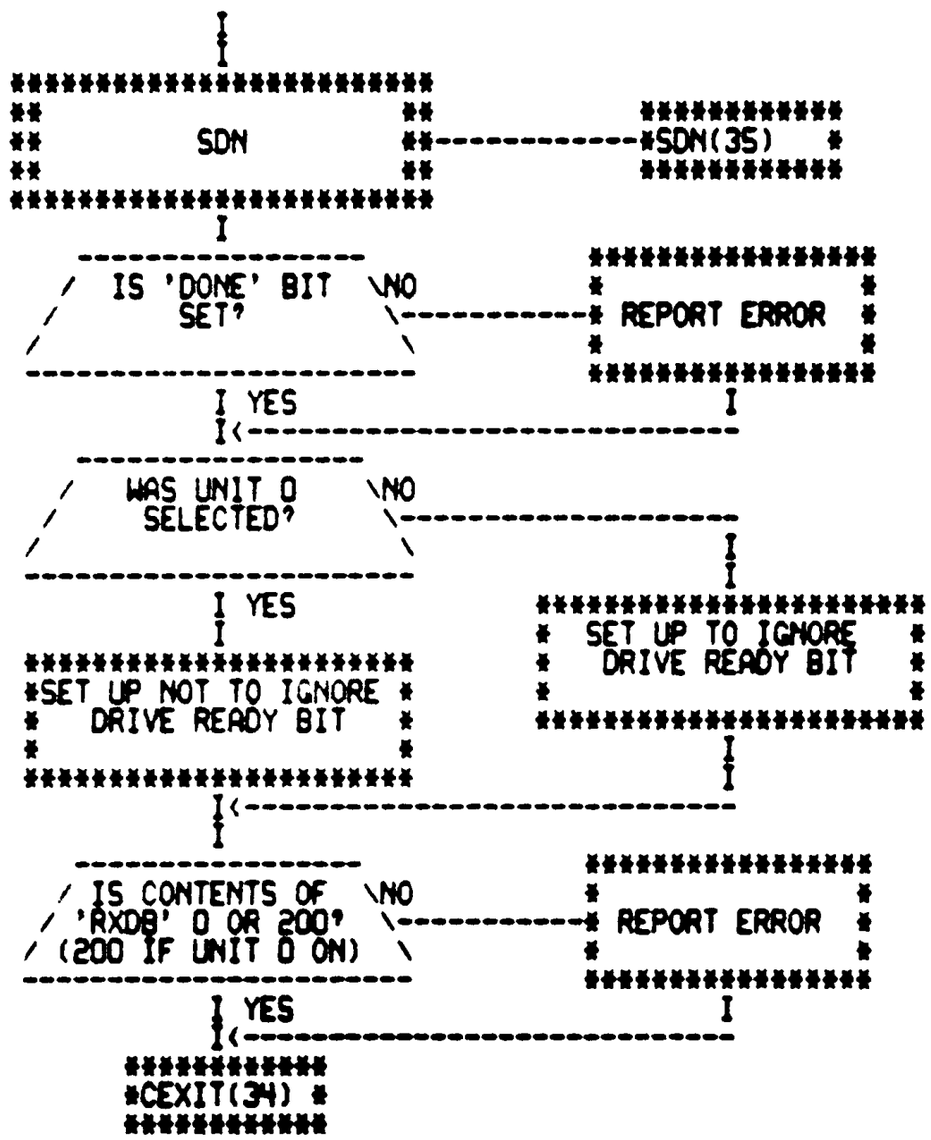
M03

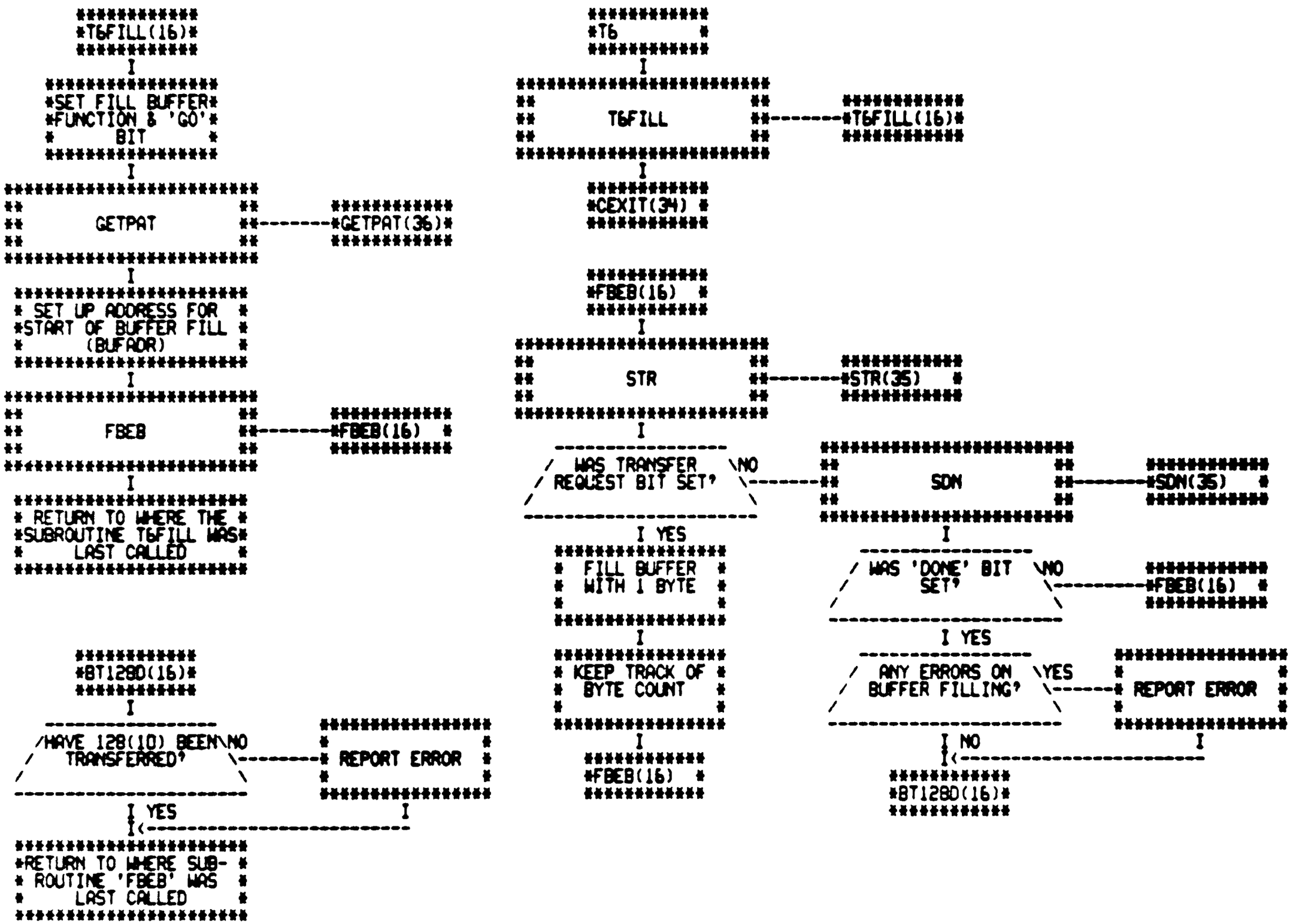

T3
SET UP VECTORS 264 & 266 FOR INTERRUPT
CALCULATE & SET CPU BR LEVEL TO A VALUE 1 LESS THAN DEVICE
SET DEVICE 'IE' BIT
NOTE: THIS SHOULD CAUSE AN INTERRUPT
DID AN INTERRUPT OCCUR?
NO
REPORT ERROR
YES
SET CPU BR LEVEL TO 7
CLEAR DEVICE 'IE' BIT
EXIT(34)


T4
SET UP VECTORS 264 & 266 FOR INTERRUPT
CALCULATE & SET CPU BR LEVEL TO VALUE DEFINED IN LOC. 'BRLEV'
SET DEVICE 'IE' BIT
NOTE: NO INTERRUPT SHOULD OCCUR CPU SAME LEVEL AS DEVICE
DID AN INTERRUPT OCCUR?
YES
REPORT ERROR
NO
SET CPU BR LEVEL TO 7
CLEAR DEVICE 'IE' BIT
CEXIT(34)

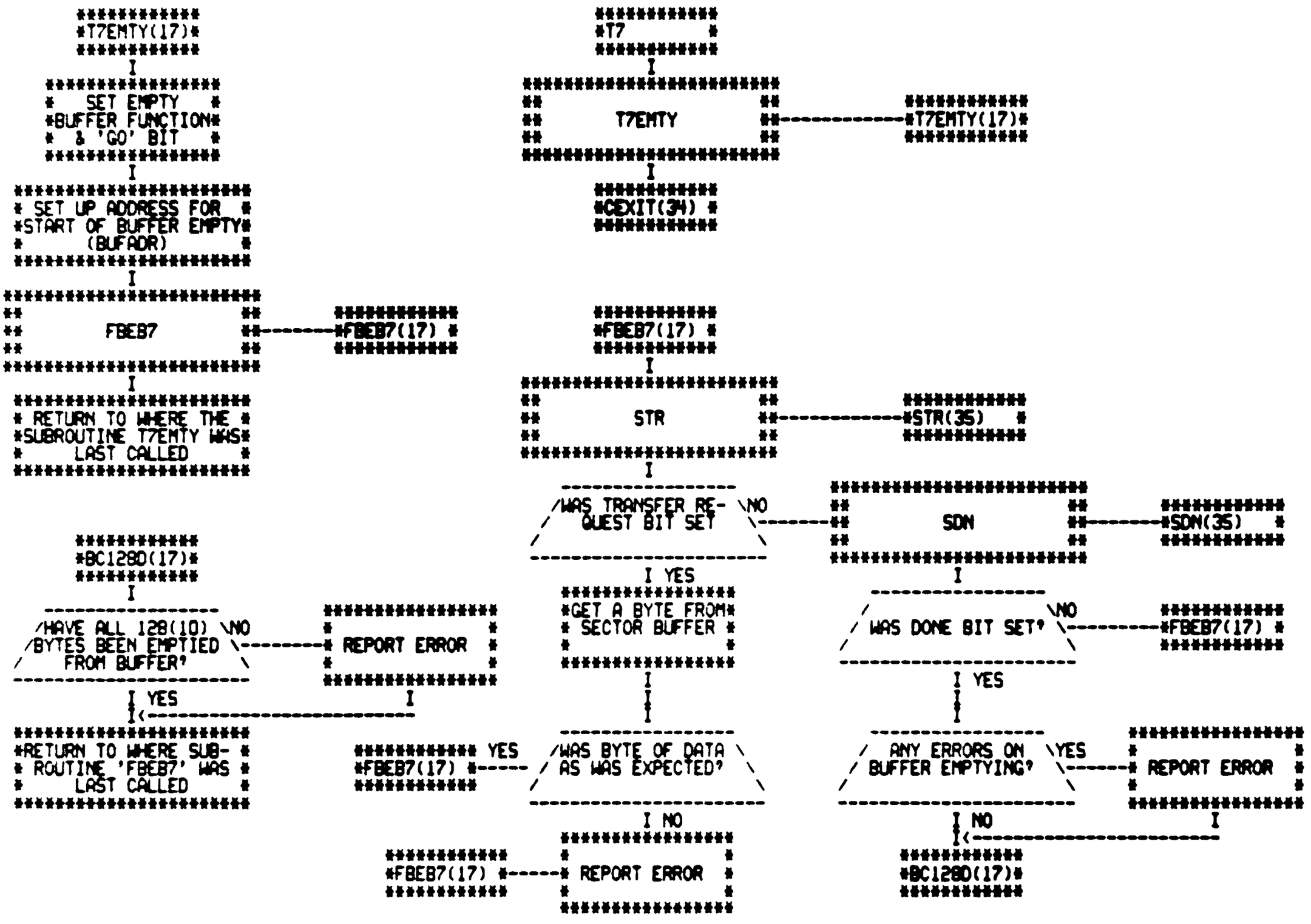

*T5 *
INITIALIZE THE RX11 ONLY
* SET 'RX INT' BIT IN * DEVICE 'RXCS' REG.
SDN
SDN(35)
IS 'DONE' BIT SET?
NO
REPORT ERROR
YES
WAS UNIT 0 SELECTED?
YES
NO
SET UP TO IGNORE UNIT 0 'DRY' BIT
*SET UP NOT TO IGNORE * UNIT 0 'DRY' BIT
IS CONTENTS OF DEVICE 'RXDB' A 4 OR 204?
NO
REPORT ERROR
YES

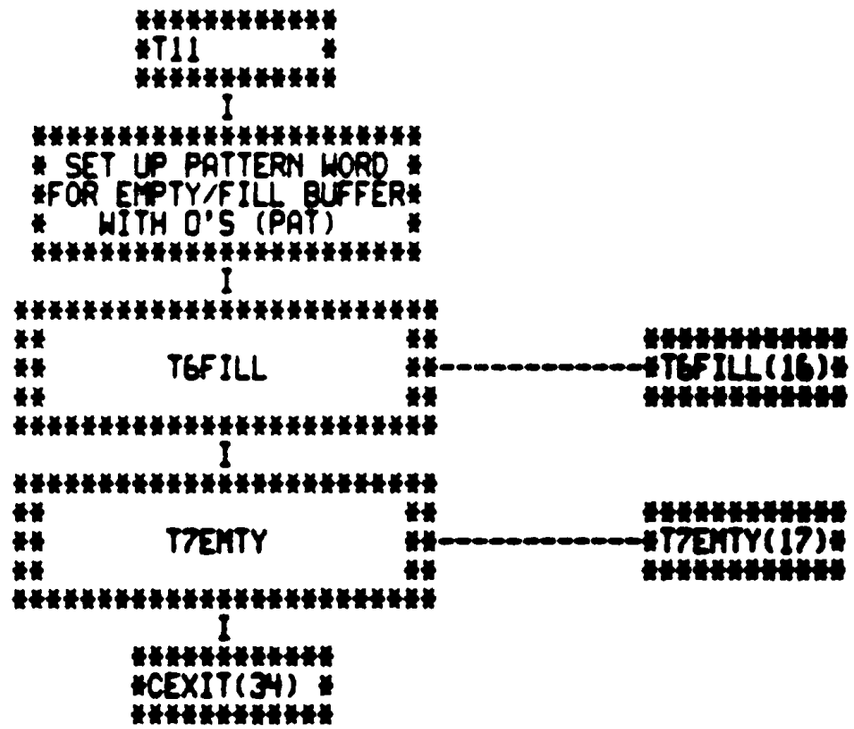

ISSUE A READ B-CODE COMMAND (FUNCTION CODE 7)
NOTE:'DONE' BIT SHOULD BE CLEAR AT THIS POINT IN TIME.
IS 'DONE' BIT CLEAR?
NO
REPORT ERROR
YES
SDN
SDN(35)
IS THE 'DONE' BIT NOW SET?
NO
REPORT ERROR
YES
IS STATUS CODE FROM 'RXDB' EQUAL TO 0?
NO
REPORT ERROR
YES
ISSUE A READ STATUS COMMAND (FUNCTION CODE 5)

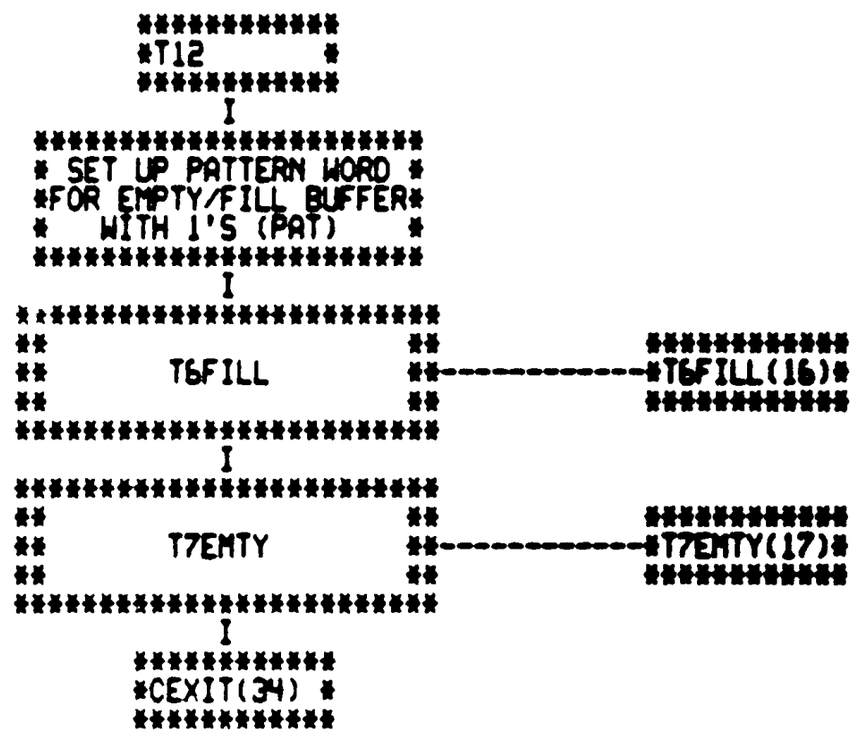

SDN
SDN(35)
IS 'DONE' BIT SET?
NO
YES
REPORT ERROR
WAS UNIT 0 SELECTED?
NO
YES
SET UP TO IGNORE DRIVE READY BIT
SET UP NOT TO IGNORE DRIVE READY BIT
IS CONTENTS OF 'RXDB' 0 OR 200? (200 IF UNIT 0 ON)
NO
YES
REPORT ERROR
CEXIT(34)

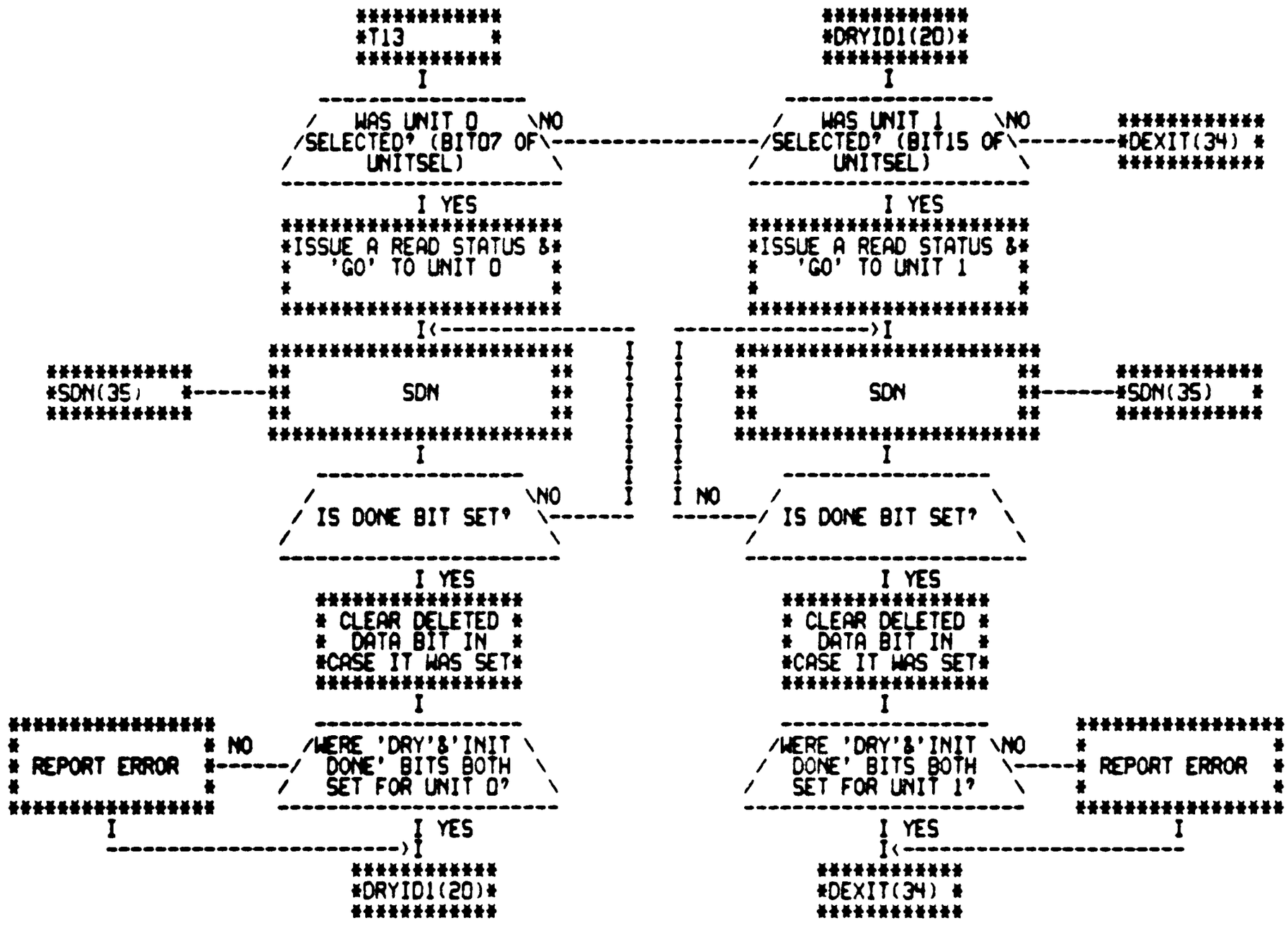

T6FILL(16)
SET FILL BUFFER FUNCTION & 'GO' BIT
GETPAT
GETPAT(36)
FBEB
FBEB(16)
SET UP ADDRESS FOR START OF BUFFER FILL (BUFADR)
RETURN TO WHERE THE SUBROUTINE T6FILL WAS LAST CALLED
BT128D(16)
HAVE 128(10) BEEN TRANSFERRED?
NO
YES
REPORT ERROR
RETURN TO WHERE SUB-ROUTINE 'FBEB' WAS LAST CALLED
T6
T6FILL
T6FILL(16)
CEXIT(34)
FBEB(16)
STR
STR(35)
WAS TRANSFER REQUEST BIT SET?
NO
YES
FILL BUFFER WITH 1 BYTE
KEEP TRACK OF BYTE COUNT
FBEB(16)
SDN
SDN(35)
WAS 'DONE' BIT SET?
NO
YES
FBEB(16)
ANY ERRORS ON BUFFER FILLING?
YES
NO
REPORT ERROR
BT128D(16)

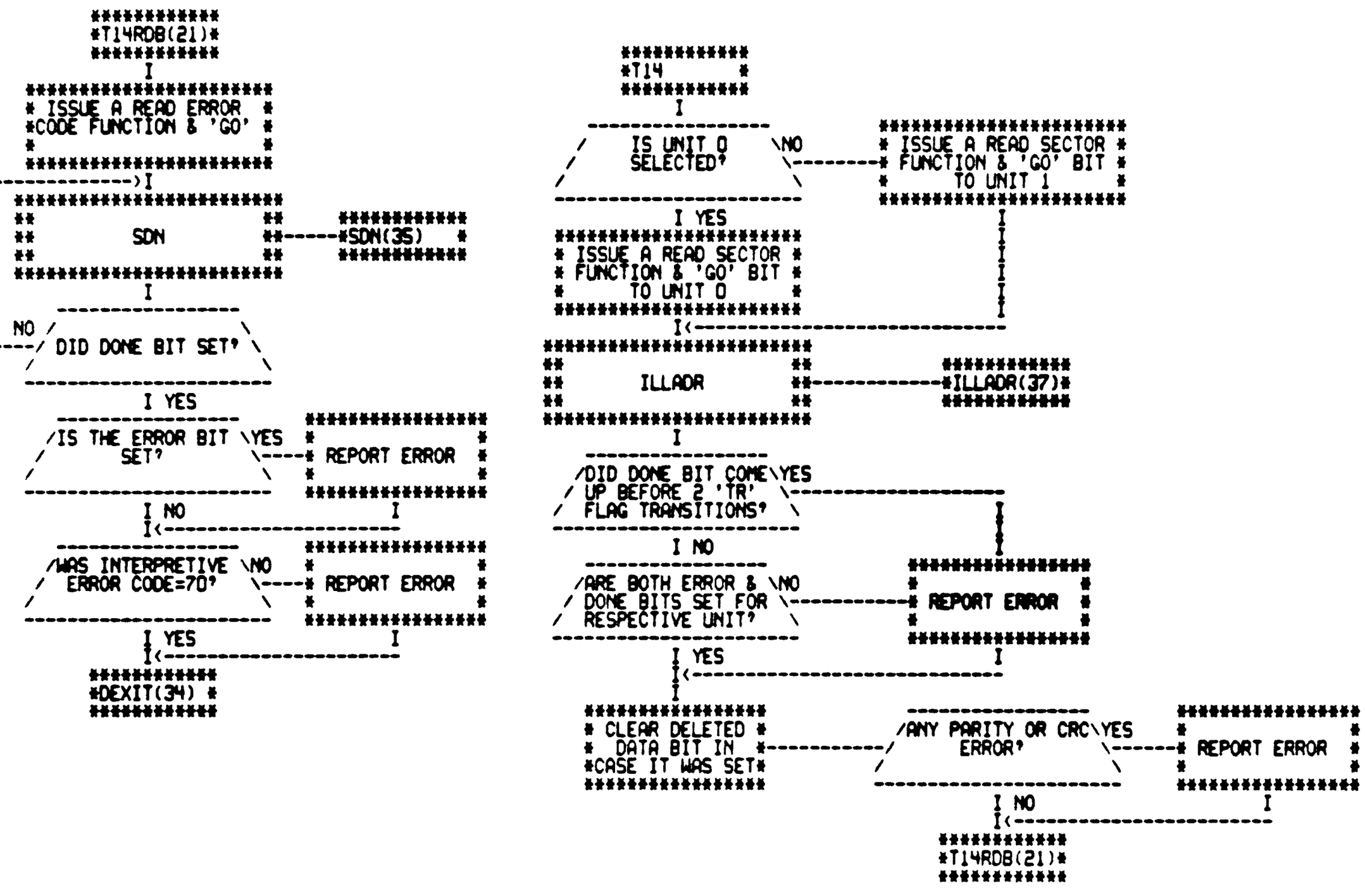
F04

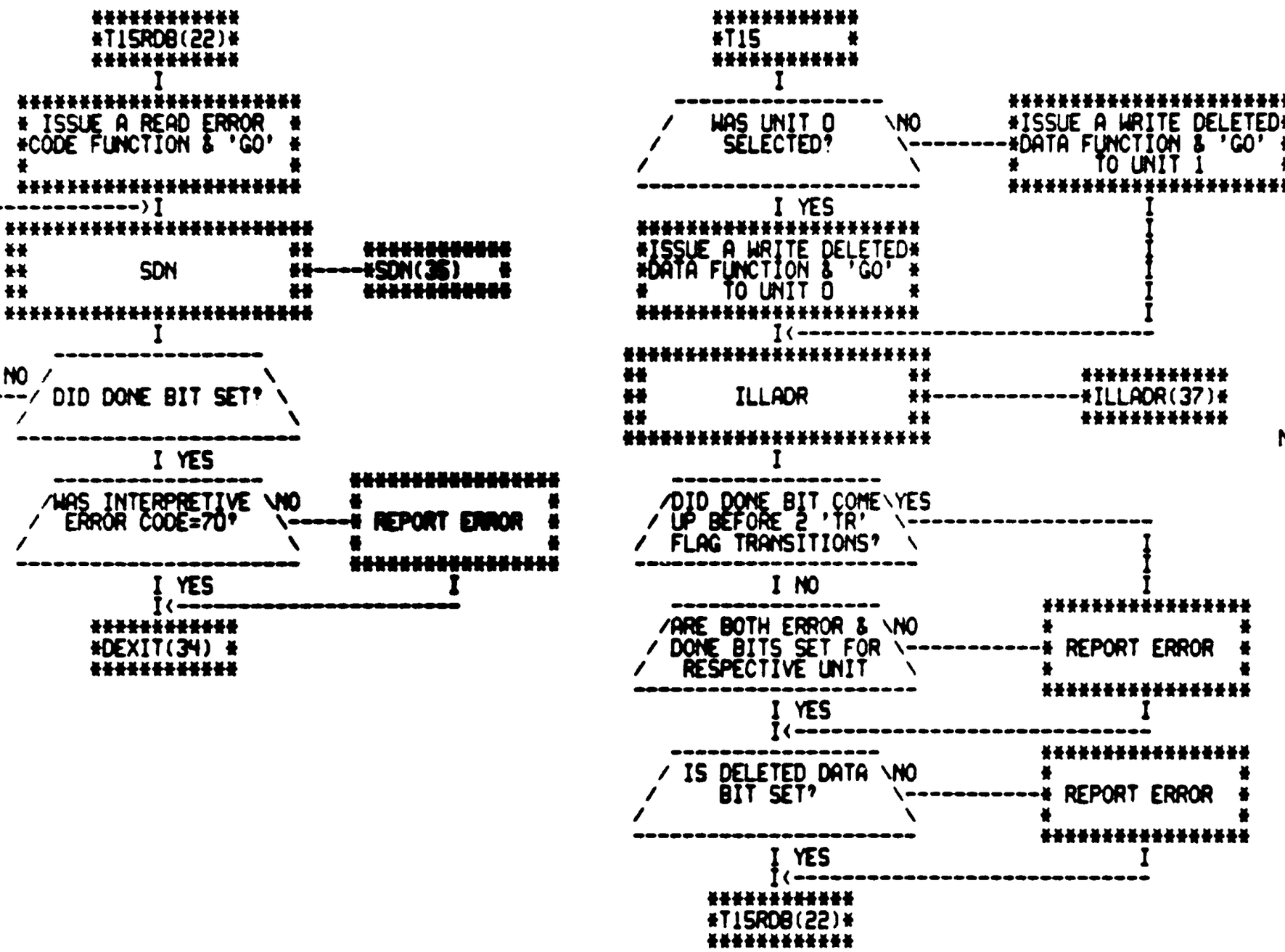

T7EMTY(17)
SET EMPTY BUFFER FUNCTION & 'GO' BIT
SET UP ADDRESS FOR START OF BUFFER EMPTY (BUFADR)
FBEB7
FBEB7(17)
RETURN TO WHERE THE SUBROUTINE T7EMTY WAS LAST CALLED
T7
T7EMTY
T7EMTY(17)
CEXIT(34)
FBEB7(17)
STR
STR(35)
WAS TRANSFER REQUEST BIT SET
NO
YES
SDN
SDN(35)
BC128D(17)
HAVE ALL 128(10) BYTES BEEN EMPTIED FROM BUFFER?
NO
REPORT ERROR
YES
RETURN TO WHERE SUBROUTINE 'FBEB7' WAS LAST CALLED
FBEB7(17)
YES
GET A BYTE FROM SECTOR BUFFER
WAS BYTE OF DATA AS WAS EXPECTED?
NO
REPORT ERROR
FBEB7(17)
WAS DONE BIT SET?
NO
FBEB7(17)
YES
ANY ERRORS ON BUFFER EMPTYING?
YES
REPORT ERROR
NO
BC128D(17)

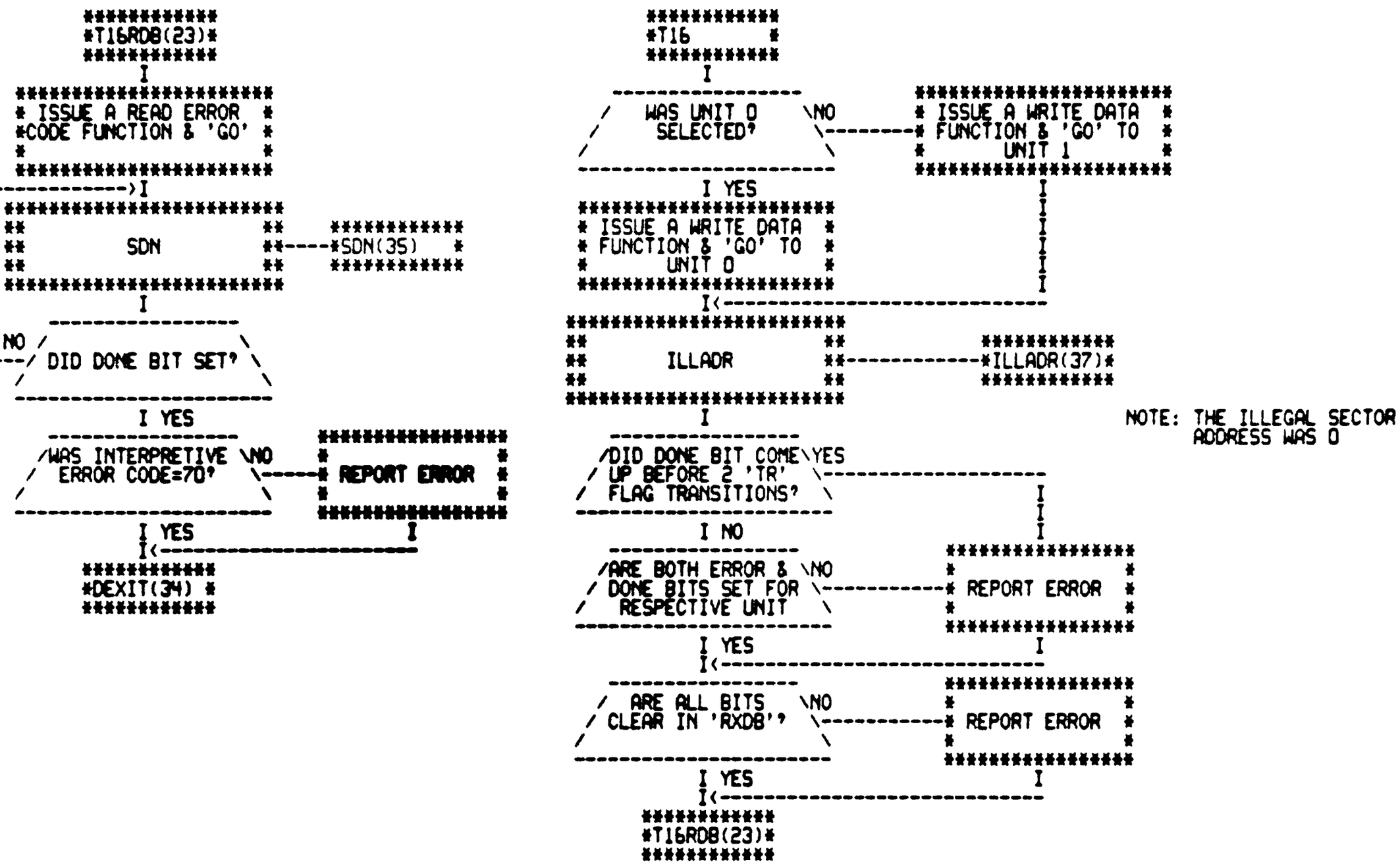

T11
SET UP PATTERN WORD FOR EMPTY/FILL BUFFER WITH 0'S (PAT)
T6FILL
T6FILL(16)
T7EMTY
T7EMTY(17)
CEXIT(34)

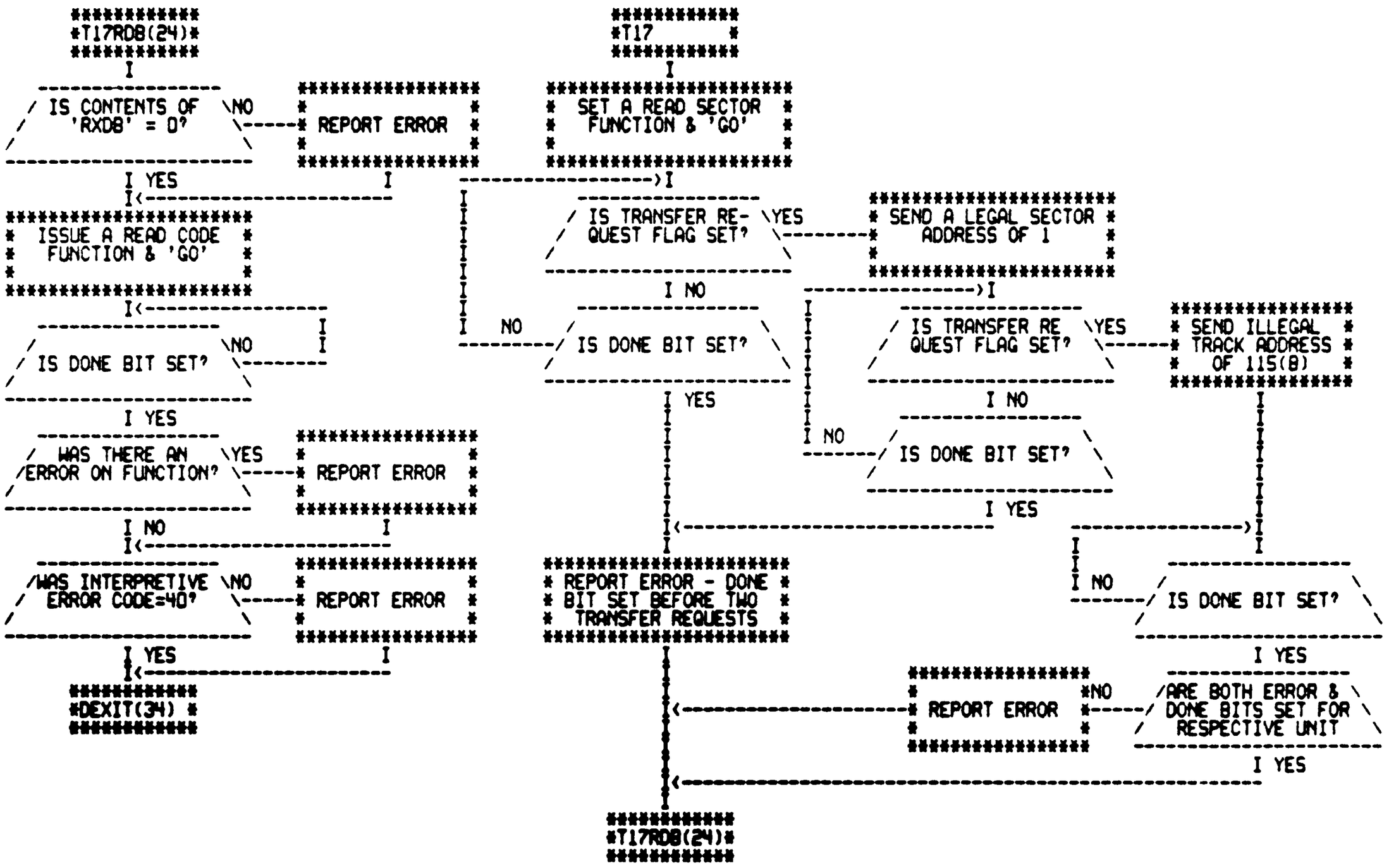

T12
SET UP PATTERN WORD FOR EMPTY/FILL BUFFER WITH 1'S (PAT)
T6FILL
T6FILL(16)
T7EMTY
T7EMTY(17)
CEXIT(34)

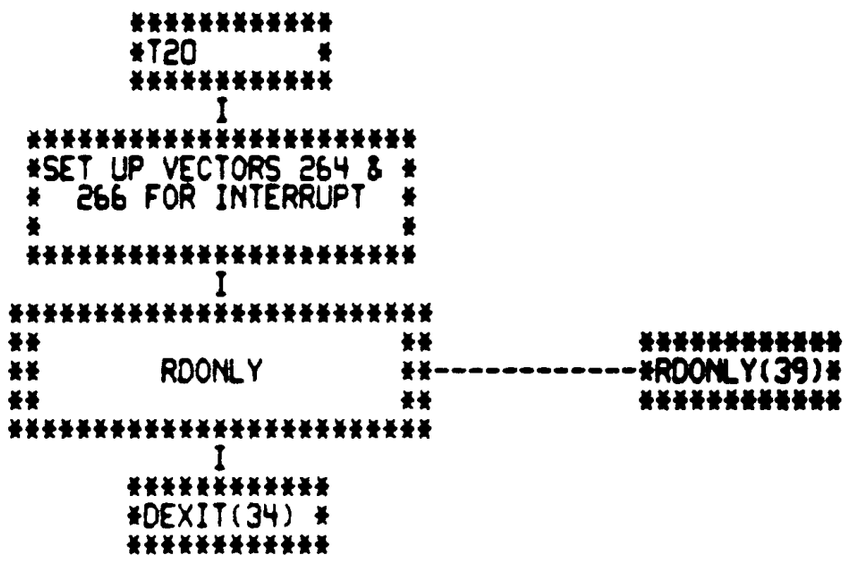

T13
WAS UNIT 0 SELECTED? (BIT07 OF UNITSEL)
NO
YES
ISSUE A READ STATUS & 'GO' TO UNIT 0
SDN
SDN(35)
IS DONE BIT SET?
NO
YES
CLEAR DELETED DATA BIT IN CASE IT WAS SET
WERE 'DRY'&'INIT DONE' BITS BOTH SET FOR UNIT 0?
NO
REPORT ERROR
YES
DRYID1(20)
DRYID1(20)
WAS UNIT 1 SELECTED? (BIT15 OF UNITSEL)
NO
DEXIT(34)
YES
ISSUE A READ STATUS & 'GO' TO UNIT 1
SDN
SDN(35)
NO
IS DONE BIT SET?
YES
CLEAR DELETED DATA BIT IN CASE IT WAS SET
WERE 'DRY'&'INIT DONE' BITS BOTH SET FOR UNIT 1?
NO
REPORT ERROR
YES
DEXIT(34)

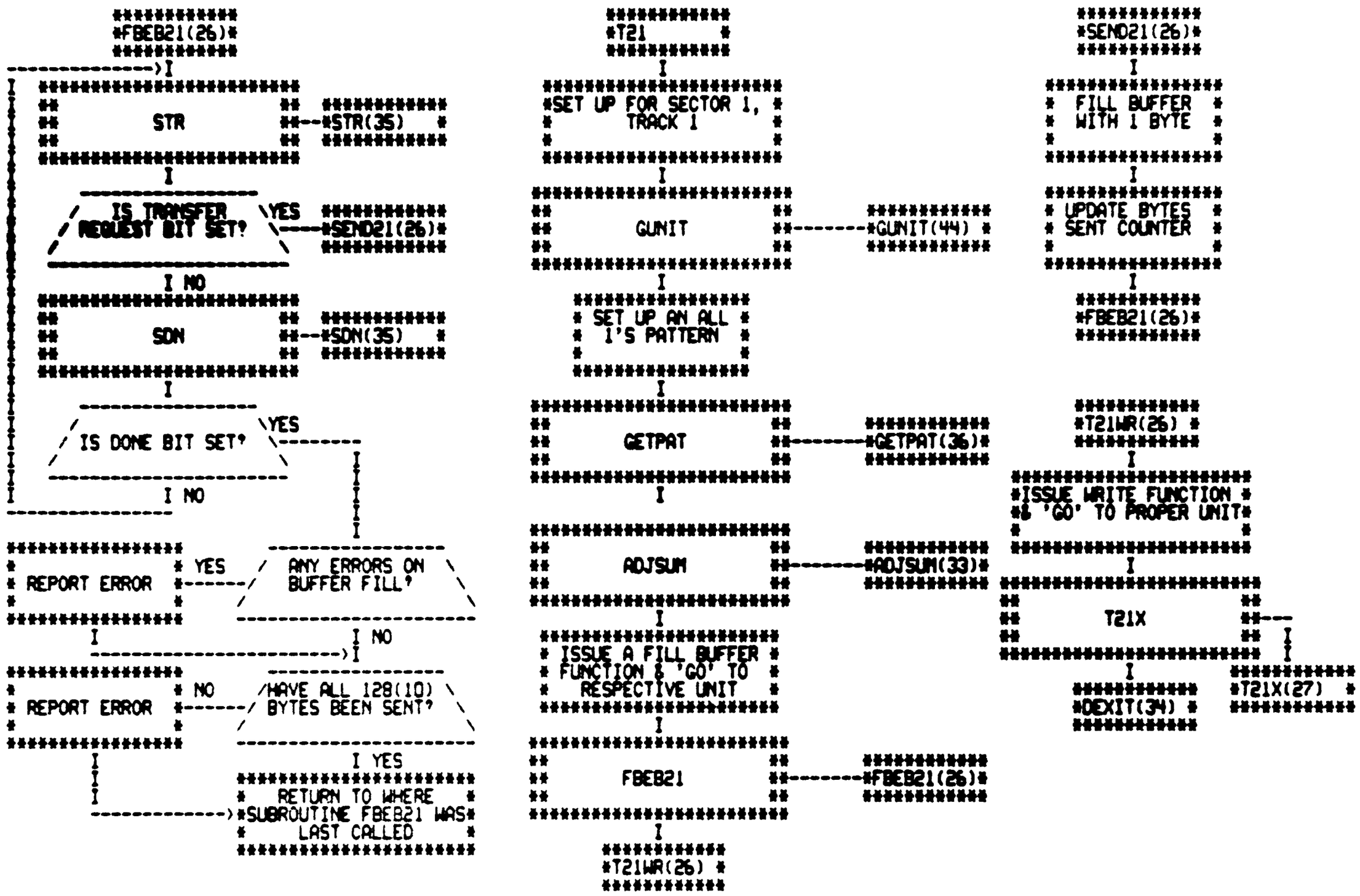

T14RDB(21)
ISSUE A READ ERROR CODE FUNCTION & 'GO'
SDN
SDN(35)
NO DID DONE BIT SET?
I YES
IS THE ERROR BIT SET?
YES
REPORT ERROR
I NO
WAS INTERPRETIVE ERROR CODE=70?
NO
REPORT ERROR
I YES
DEXIT(34)
T14
IS UNIT 0 SELECTED?
NO
ISSUE A READ SECTOR FUNCTION & 'GO' BIT TO UNIT 1
I YES
ISSUE A READ SECTOR FUNCTION & 'GO' BIT TO UNIT 0
ILLADR
ILLADR(37)
DID DONE BIT COME UP BEFORE 2 'TR' FLAG TRANSITIONS?
I NO
ARE BOTH ERROR & DONE BITS SET FOR RESPECTIVE UNIT?
NO
REPORT ERROR
I YES
CLEAR DELETED DATA BIT IN CASE IT WAS SET
ANY PARITY OR CRC ERROR?
YES
REPORT ERROR
I NO
T14RDB(21)

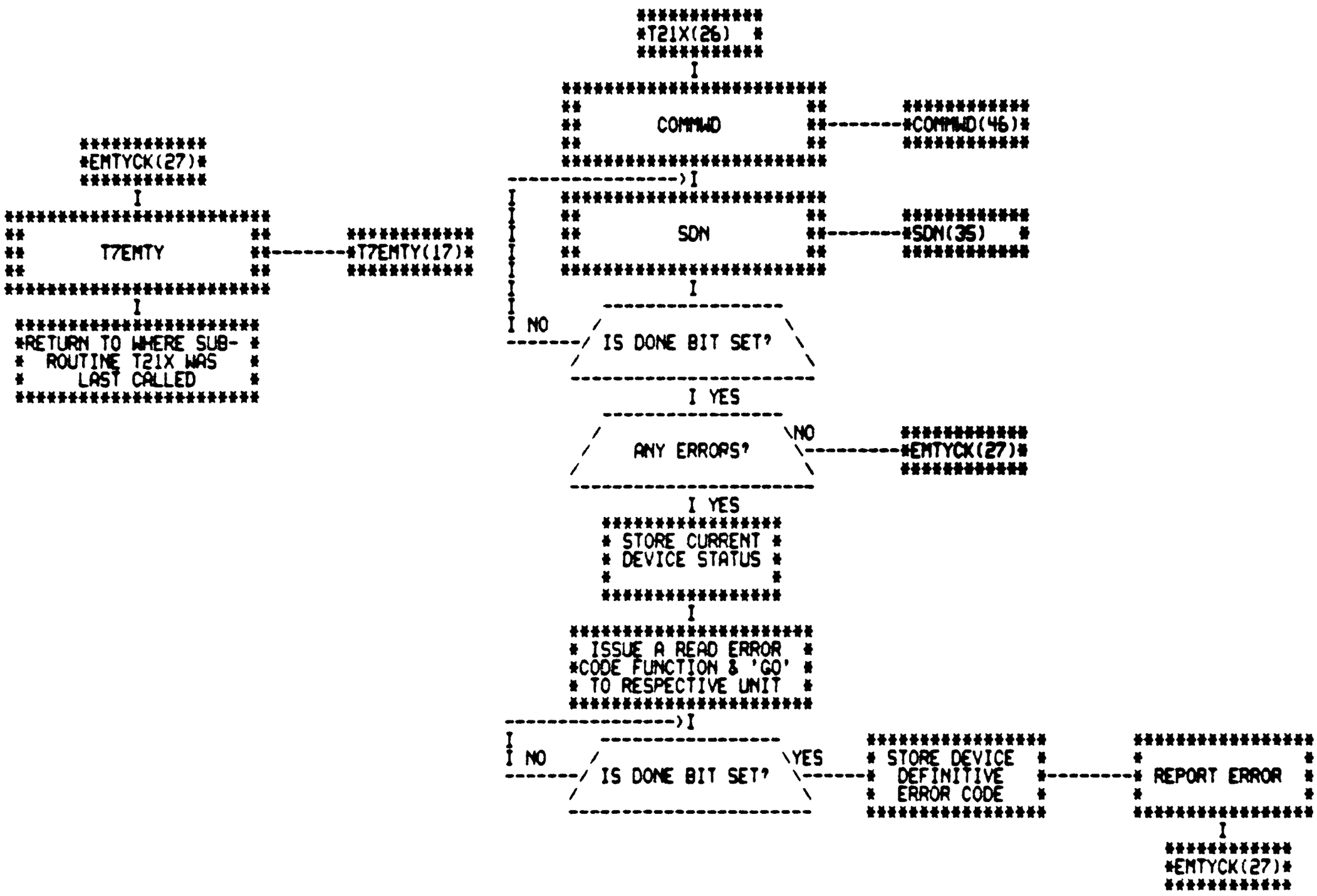

T15RDB(22)
ISSUE A READ ERROR CODE FUNCTION & 'GO'
SDN
SDN(36)
DID DONE BIT SET?
NO
YES
WAS INTERPRETIVE ERROR CODE=70?
NO
REPORT ERROR
YES
DEXIT(34)
T15
WAS UNIT 0 SELECTED?
NO
ISSUE A WRITE DELETED DATA FUNCTION & 'GO' TO UNIT 1
YES
ISSUE A WRITE DELETED DATA FUNCTION & 'GO' TO UNIT 0
ILLADR
ILLADR(37)
NOTE: THE ILLEGAL SECTOR ADDRESS WAS 0 BUT 'DO' BIT SHOULD STILL BE SET
DID DONE BIT COME UP BEFORE 2 'TR' FLAG TRANSITIONS?
YES
NO
ARE BOTH ERROR & DONE BITS SET FOR RESPECTIVE UNIT
NO
REPORT ERROR
YES
IS DELETED DATA BIT SET?
NO
REPORT ERROR
YES
T15RDB(22)

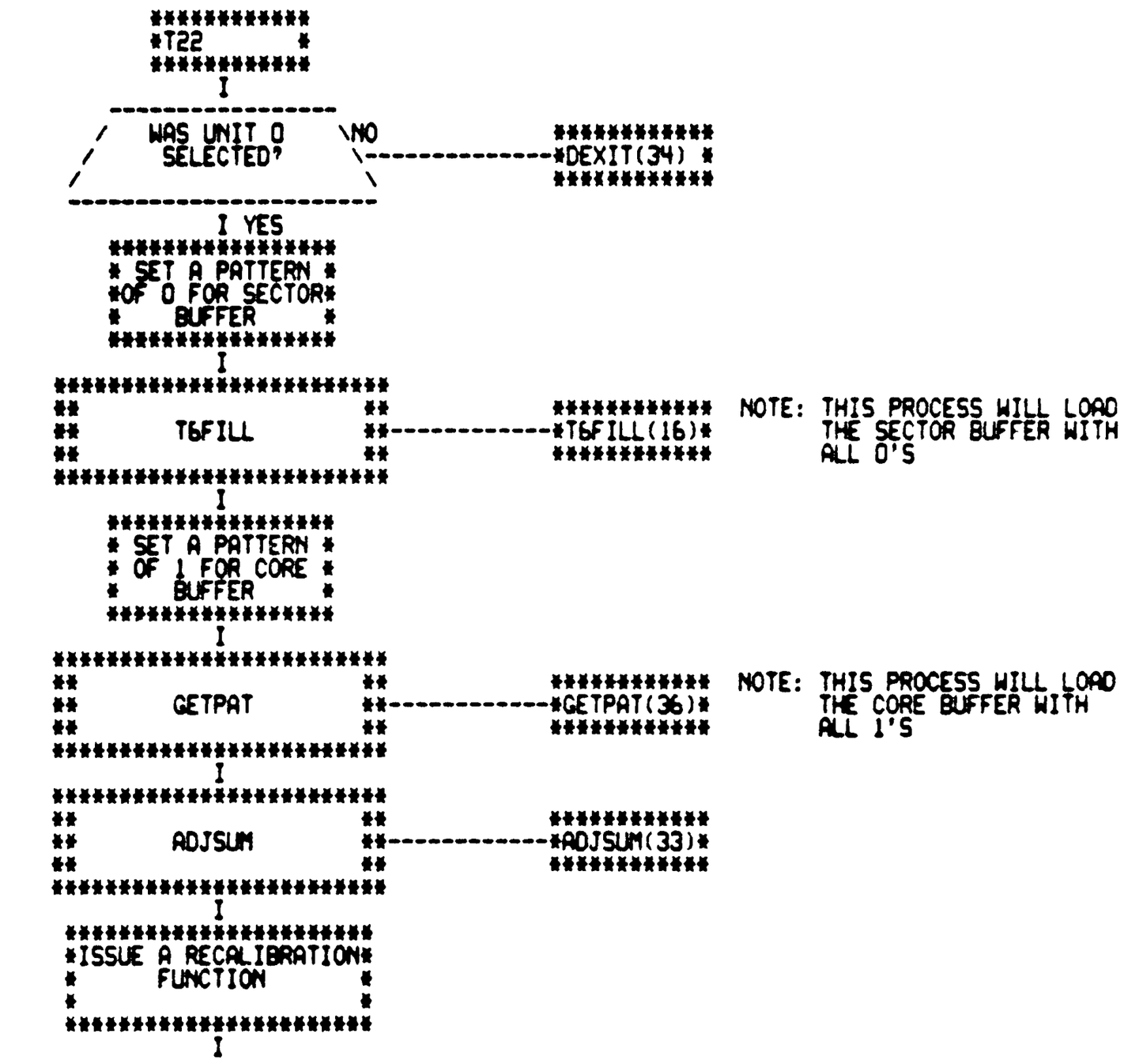

T16RDB(23)
ISSUE A READ ERROR CODE FUNCTION & 'GO'
SDN
SDN(35)
DID DONE BIT SET?
NO
YES
WAS INTERPRETIVE ERROR CODE=70?
NO
REPORT ERROR
YES
DEXIT(34)
T16
WAS UNIT 0 SELECTED?
NO
ISSUE A WRITE DATA FUNCTION & 'GO' TO UNIT 1
YES
ISSUE A WRITE DATA FUNCTION & 'GO' TO UNIT 0
ILLADR
ILLADR(37)
DID DONE BIT COME UP BEFORE 2 'TR' FLAG TRANSITIONS?
YES
NO
ARE BOTH ERROR & DONE BITS SET FOR RESPECTIVE UNIT
NO
REPORT ERROR
YES
ARE ALL BITS CLEAR IN 'RXDB'?
NO
REPORT ERROR
YES
T16RDB(23)
NOTE: THE ILLEGAL SECTOR ADDRESS WAS 0


#T17RDB(24)#
IS CONTENTS OF 'RXDB' = 0?
NO
REPORT ERROR
YES
ISSUE A READ CODE FUNCTION & 'GO'
IS DONE BIT SET?
NO
YES
WAS THERE AN ERROR ON FUNCTION?
YES
REPORT ERROR
NO
WAS INTERPRETIVE ERROR CODE=40?
NO
REPORT ERROR
YES
#DEXIT(34)#
#T17#
SET A READ SECTOR FUNCTION & 'GO'
IS TRANSFER REQUEST FLAG SET?
YES
SEND A LEGAL SECTOR ADDRESS OF 1
NO
IS DONE BIT SET?
NO
YES
IS TRANSFER REQUEST FLAG SET?
YES
SEND ILLEGAL TRACK ADDRESS OF 115(8)
NO
IS DONE BIT SET?
NO
YES
IS DONE BIT SET?
YES
ARE BOTH ERROR & DONE BITS SET FOR RESPECTIVE UNIT
YES
NO
REPORT ERROR
REPORT ERROR - DONE BIT SET BEFORE TWO TRANSFER REQUESTS
#T17RDB(24)#

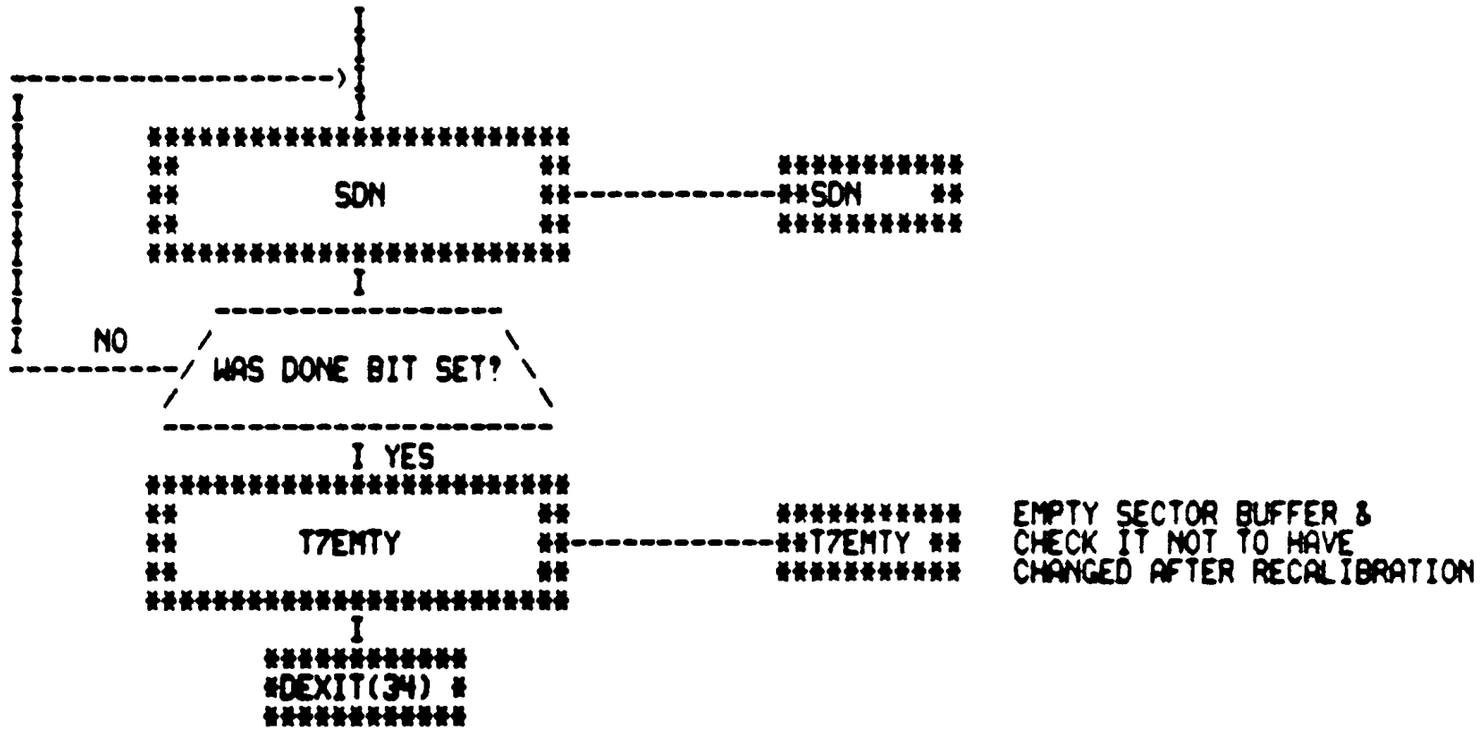

T20
SET UP VECTORS 264 & 266 FOR INTERRUPT
RDONLY
RDONLY(39)
DEXIT(34)

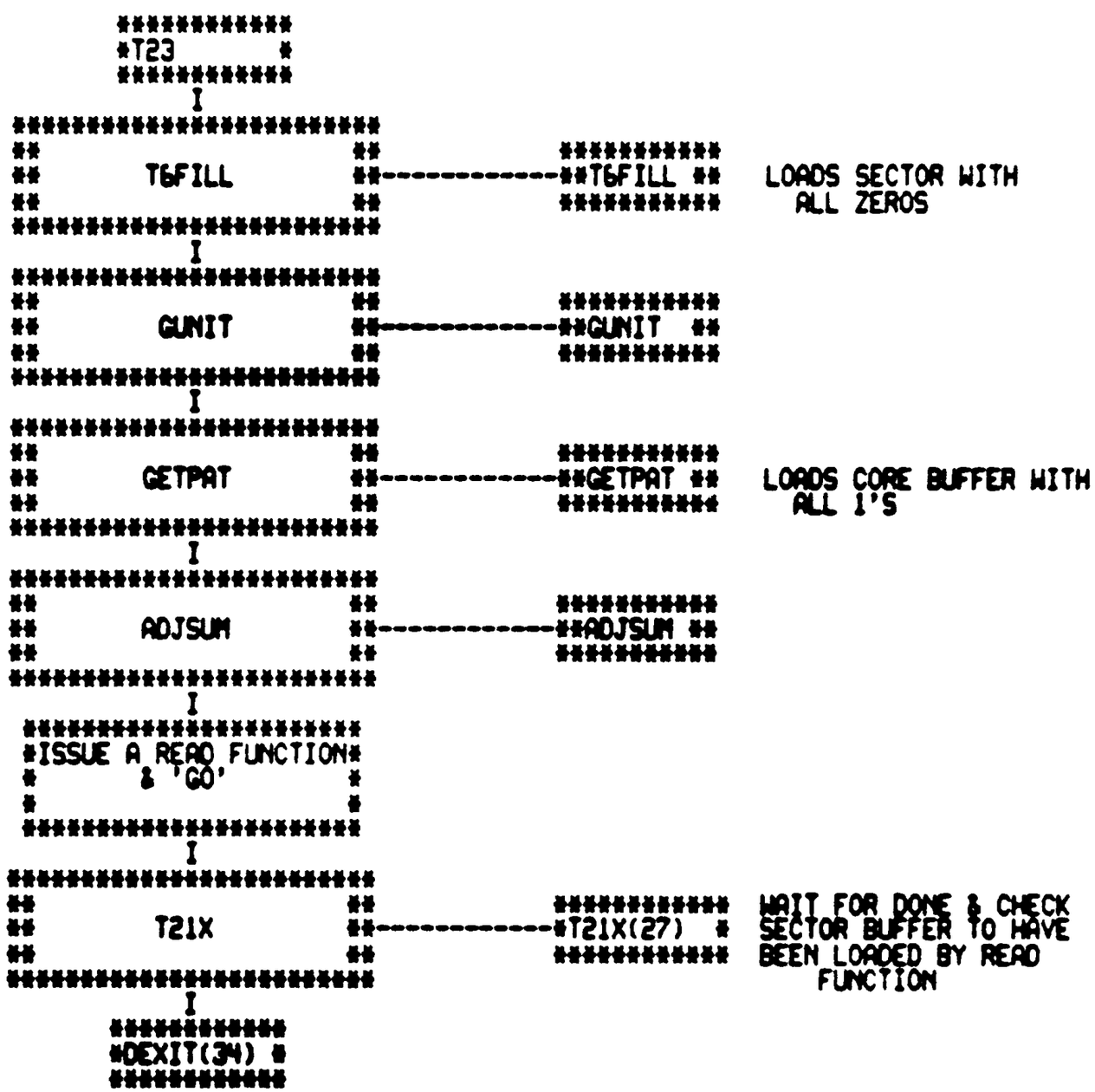

#FBEB21(26)#
IS TRANSFER REQUEST BIT SET?
YES
#SEND21(26)#
STR
#STR(35)#
I NO
SDN
#SDN(35)#
IS DONE BIT SET?
YES
I NO
REPORT ERROR
YES
ANY ERRORS ON BUFFER FILL?
I NO
REPORT ERROR
NO
HAVE ALL 128(10) BYTES BEEN SENT?
I YES
RETURN TO WHERE SUBROUTINE FBEB21 WAS LAST CALLED
#T21 #
SET UP FOR SECTOR 1, TRACK 1
GUNIT
#GUNIT(44) #
SET UP AN ALL 1'S PATTERN
GETPAT
#GETPAT(36)#
ADJSUM
#ADJSUM(33)#
ISSUE A FILL BUFFER FUNCTION & 'GO' TO RESPECTIVE UNIT
FBEB21
#FBEB21(26)#
#T21WR(26) #
#SEND21(26)#
FILL BUFFER WITH 1 BYTE
UPDATE BYTES SENT COUNTER
#FBEB21(26)#
#T21WR(26) #
#ISSUE WRITE FUNCTION & 'GO' TO PROPER UNIT#
T21X
#DEXIT(34) #
#T21X(27) #

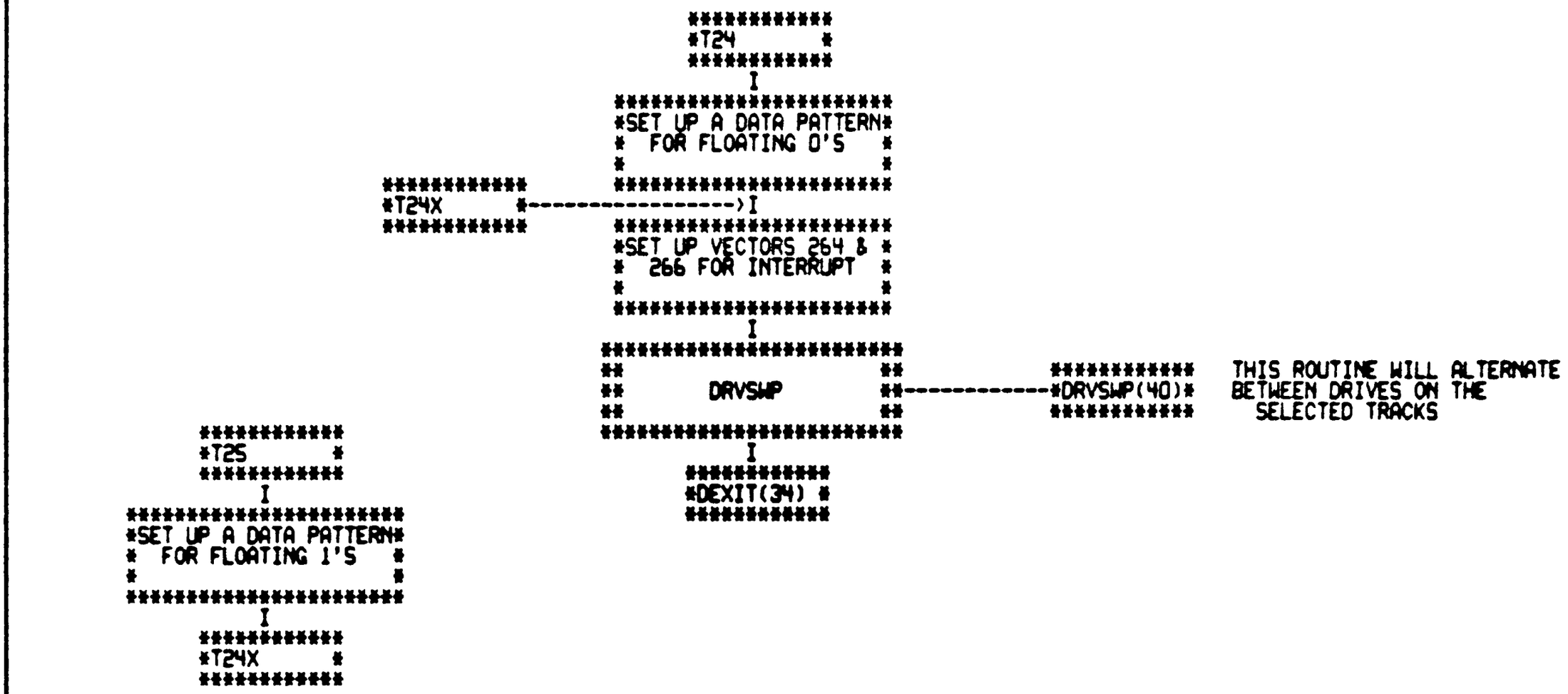

T21X(26)
COMMND
COMMND(46)
SDN
SDN(35)
IS DONE BIT SET?
NO
YES
ANY ERRORS?
NO
EMTYCK(27)
STORE CURRENT DEVICE STATUS
ISSUE A READ ERROR CODE FUNCTION & 'GO' TO RESPECTIVE UNIT
IS DONE BIT SET?
YES
NO
STORE DEVICE DEFINITIVE ERROR CODE
REPORT ERROR
EMTYCK(27)
T7EMTY
T7EMTY(17)
RETURN TO WHERE SUB-ROUTINE T21X WAS LAST CALLED
EMTYCK(27)

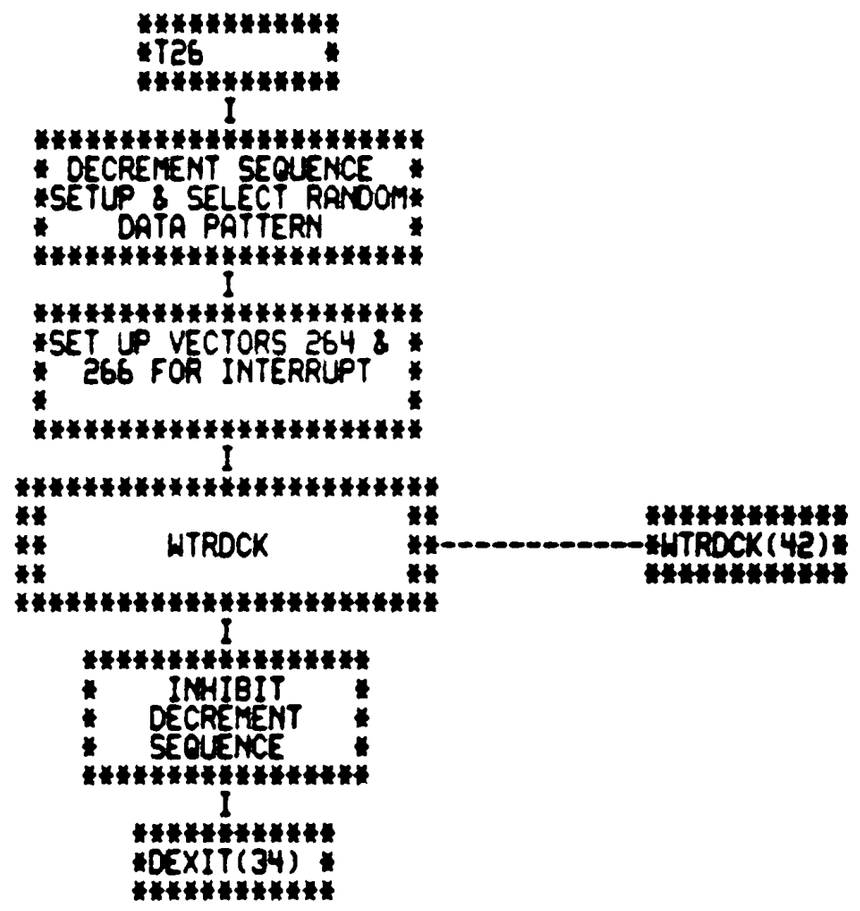

T22
WAS UNIT 0 SELECTED?
NO
DEXIT(34)
YES
SET A PATTERN OF 0 FOR SECTOR BUFFER
T6FILL
T6FILL(16)
NOTE: THIS PROCESS WILL LOAD THE SECTOR BUFFER WITH ALL 0'S
SET A PATTERN OF 1 FOR CORE BUFFER
GETPAT
GETPAT(36)
NOTE: THIS PROCESS WILL LOAD THE CORE BUFFER WITH ALL 1'S
ADJSUM
ADJSUM(33)
ISSUE A RECALIBRATION FUNCTION

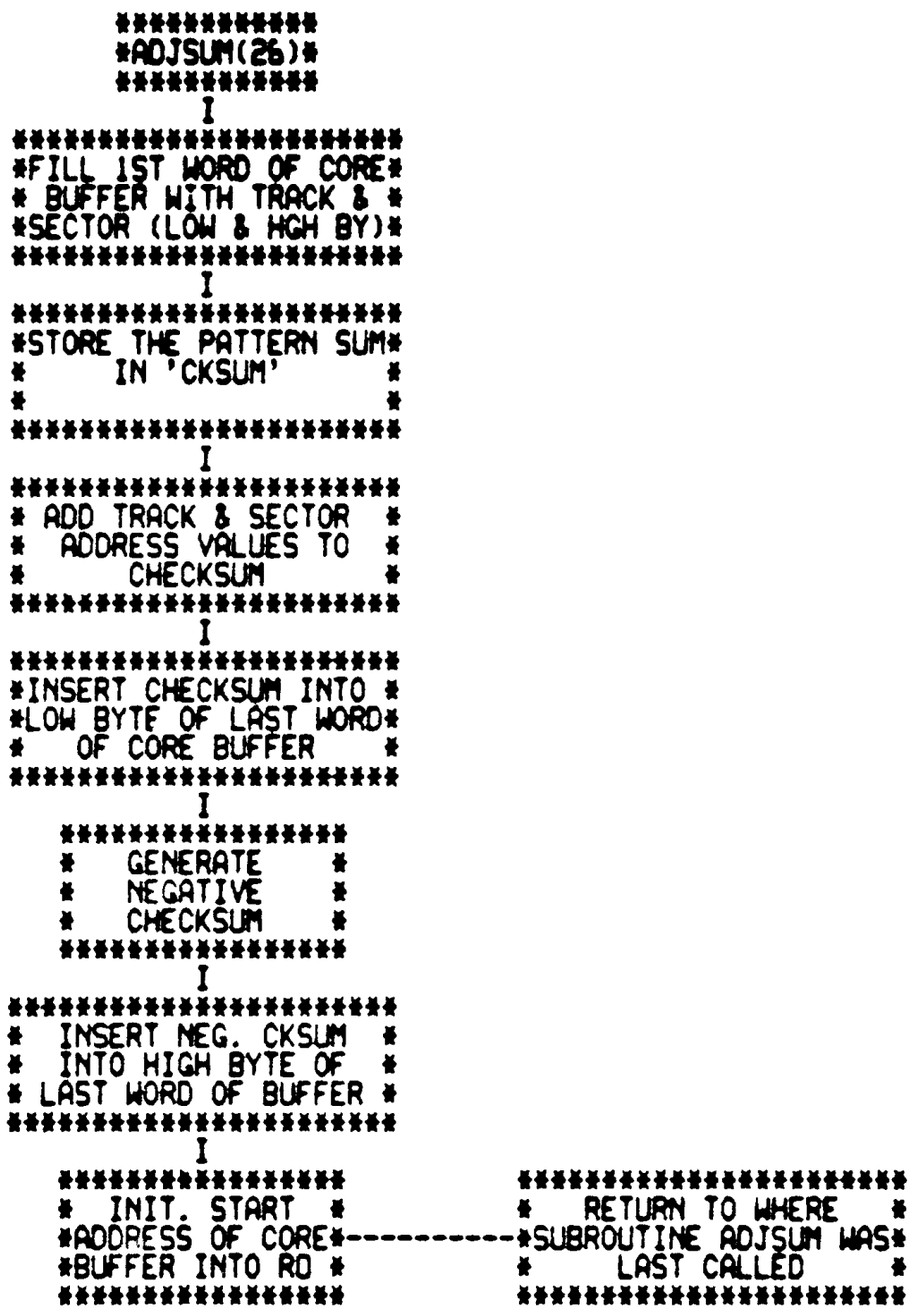
E05

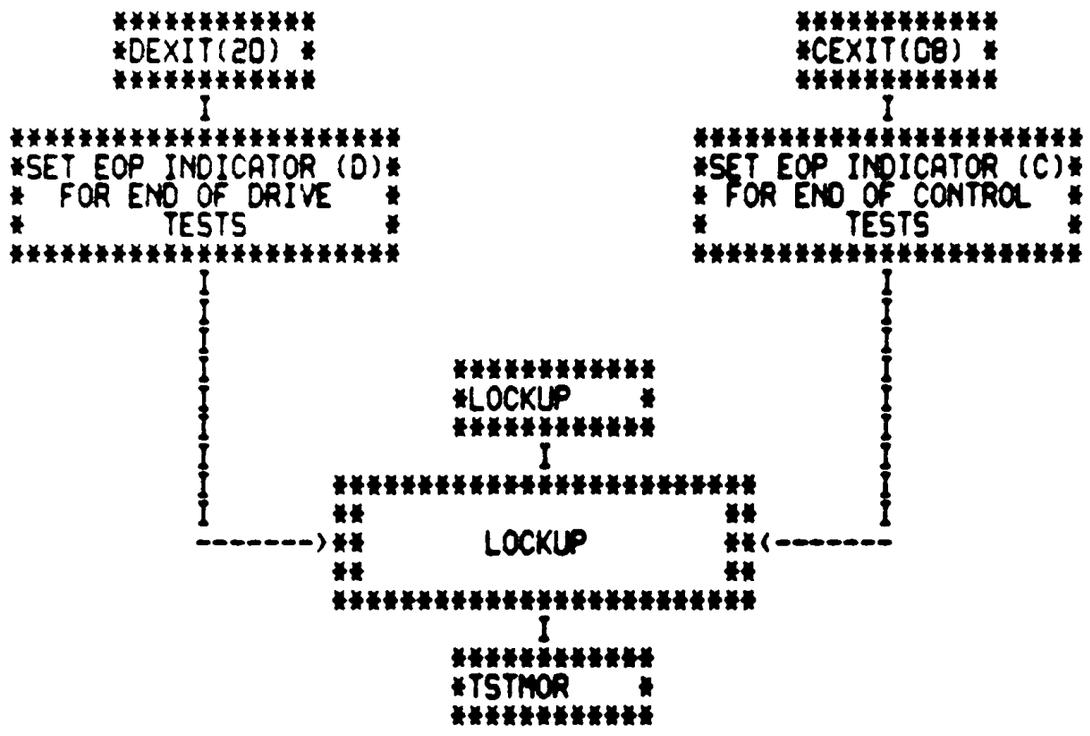

SDN
SDN
NO
WAS DONE BIT SET?
YES
T7EMTY
T7EMTY
EMPTY SECTOR BUFFER &
CHECK IT NOT TO HAVE
CHANGED AFTER RECALIBRATION
#EXIT(34) #

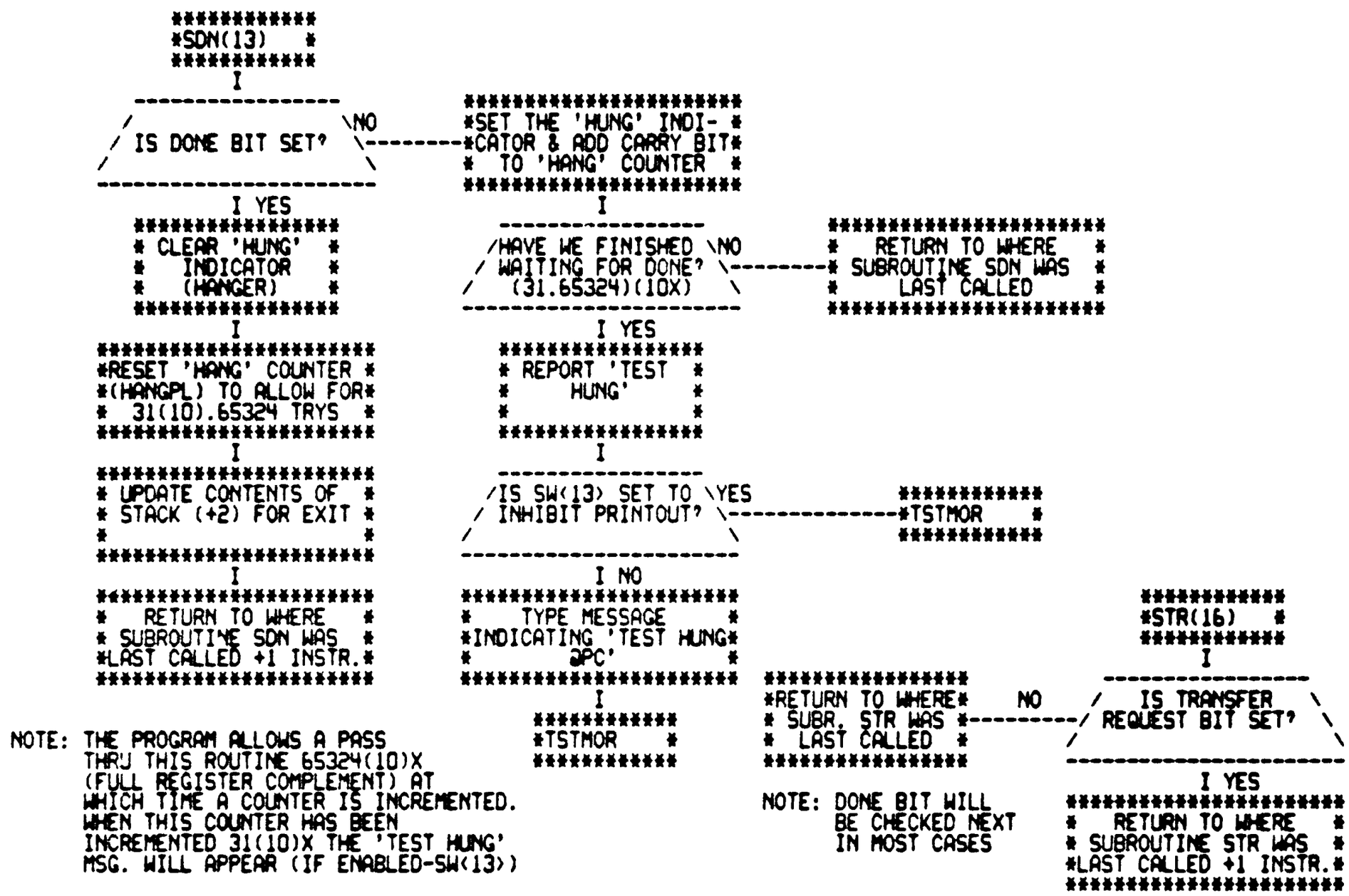

T23
T6FILL
T6FILL
LOADS SECTOR WITH ALL ZEROS
GUNIT
GUNIT
GETPAT
GETPAT
LOADS CORE BUFFER WITH ALL I'S
ADJSUM
ADJSUM
ISSUE A READ FUNCTION & 'GO'
T21X
T21X(27)
WAIT FOR DONE & CHECK SECTOR BUFFER TO HAVE BEEN LOADED BY READ FUNCTION
EXIT(34)


T24
SET UP A DATA PATTERN FOR FLOATING 0'S
T24X
SET UP VECTORS 264 & 266 FOR INTERRUPT
DRVSWP
DRVSWP(40)
THIS ROUTINE WILL ALTERNATE BETWEEN DRIVES ON THE SELECTED TRACKS
DEXIT(34)
T25
SET UP A DATA PATTERN FOR FLOATING 1'S
T24X


T26
DECREMENT SEQUENCE SETUP & SELECT RANDOM DATA PATTERN
SET UP VECTORS 264 & 266 FOR INTERRUPT
WTRDCK
WTRDCK(42)
INHIBIT DECREMENT SEQUENCE
DEXIT(34)


ADJSUM(26)
FILL 1ST WORD OF CORE BUFFER WITH TRACK & SECTOR (LOW & HGH BY)
STORE THE PATTERN SUM IN 'CKSUM'
ADD TRACK & SECTOR ADDRESS VALUES TO CHECKSUM
INSERT CHECKSUM INTO LOW BYTE OF LAST WORD OF CORE BUFFER
GENERATE NEGATIVE CHECKSUM
INSERT NEG. CKSUM INTO HIGH BYTE OF LAST WORD OF BUFFER
INIT. START ADDRESS OF CORE BUFFER INTO R0
RETURN TO WHERE SUBROUTINE ADJSUM WAS LAST CALLED


*DEXIT(20) *
SET EOP INDICATOR (D)
* FOR END OF DRIVE *
* TESTS *
*CEXIT(08) *
SET EOP INDICATOR (C)
* FOR END OF CONTROL *
* TESTS *
*LOCKUP *
LOCKUP
*TSTMOR *


*SDN(13) *
IS DONE BIT SET?
NO
YES
* CLEAR 'HUNG' INDICATOR (HANGER) *
*RESET 'HANG' COUNTER *(HANGPL) TO ALLOW FOR* 31(10).65324 TRYS *
* UPDATE CONTENTS OF STACK (+2) FOR EXIT *
* RETURN TO WHERE SUBROUTINE SDN WAS *LAST CALLED +1 INSTR.*
*SET THE 'HUNG' INDI- *CATOR & ADD CARRY BIT* TO 'HANG' COUNTER *
HAVE WE FINISHED WAITING FOR DONE? (31.65324)(10X)
NO
YES
* REPORT 'TEST HUNG' *
IS SW(13) SET TO INHIBIT PRINTOUT?
YES
NO
*TSTMOR *
* TYPE MESSAGE *INDICATING 'TEST HUNG' @PC' *
*TSTMOR *
* RETURN TO WHERE SUBROUTINE SDN WAS LAST CALLED *
RETURN TO WHERE SUBR. STR WAS * LAST CALLED *
NO
*STR(16) *
IS TRANSFER REQUEST BIT SET?
NO
YES
* RETURN TO WHERE SUBROUTINE STR WAS *LAST CALLED +1 INSTR.*
NOTE: THE PROGRAM ALLOWS A PASS THRU THIS ROUTINE 65324(10)X (FULL REGISTER COMPLEMENT) AT WHICH TIME A COUNTER IS INCREMENTED. WHEN THIS COUNTER HAS BEEN INCREMENTED 31(10)X THE 'TEST HUNG' MSG. WILL APPEAR (IF ENABLED-SW(13))
NOTE: DONE BIT WILL BE CHECKED NEXT IN MOST CASES

NOTE: PATTERN SELECTIONS ARE AS FOLLOWS:

A) ZEROS

B) ONES

C) FLOATING 0

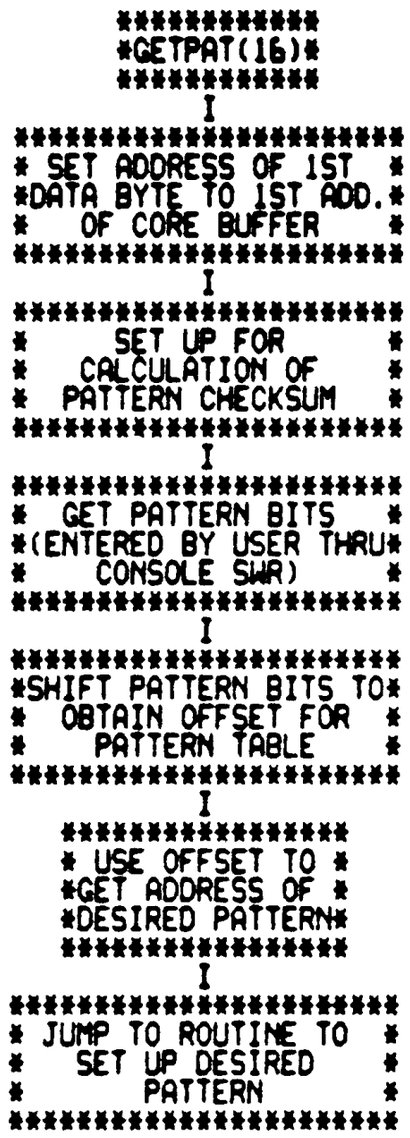
D) FLOATING 1

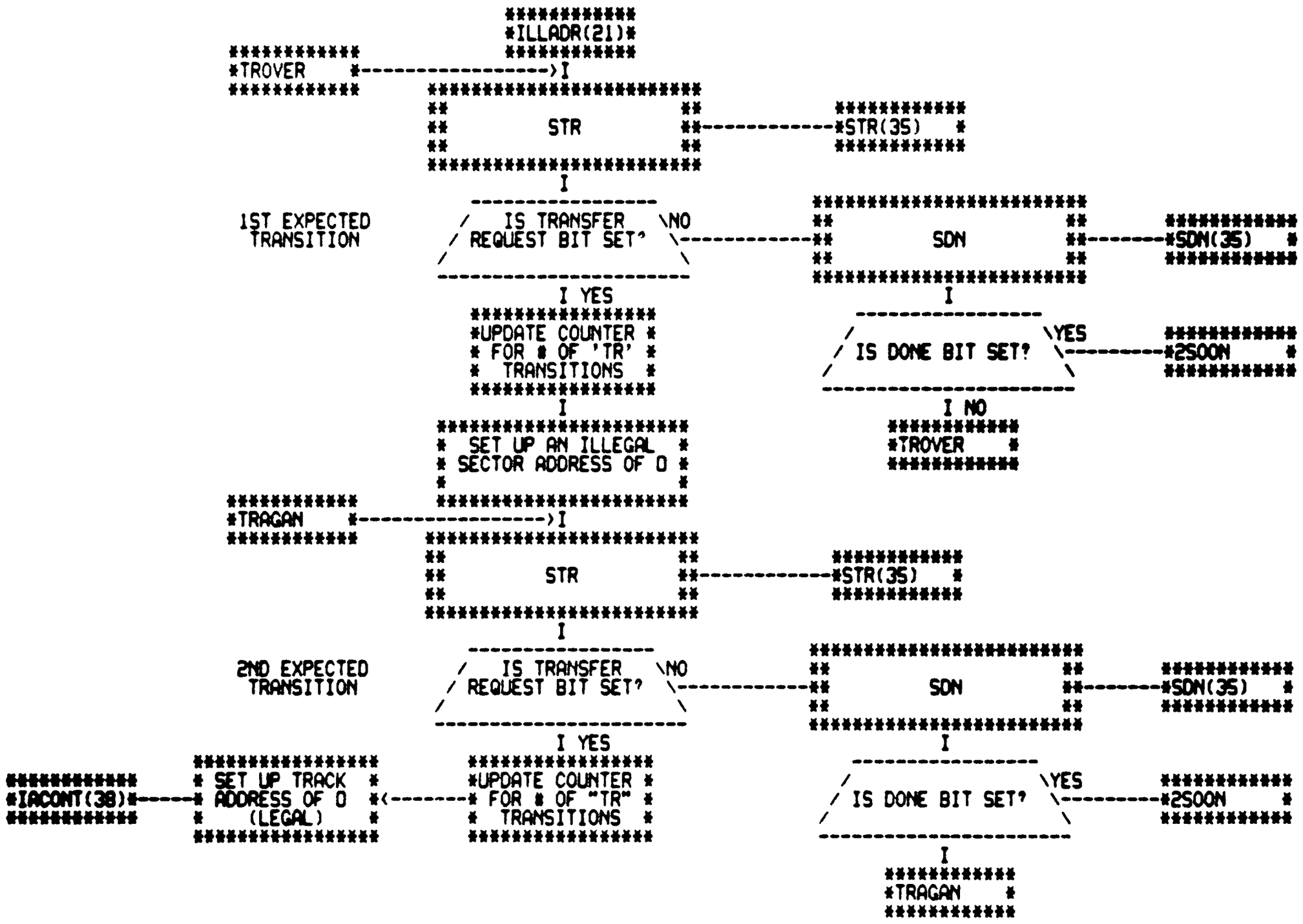
E) 125052

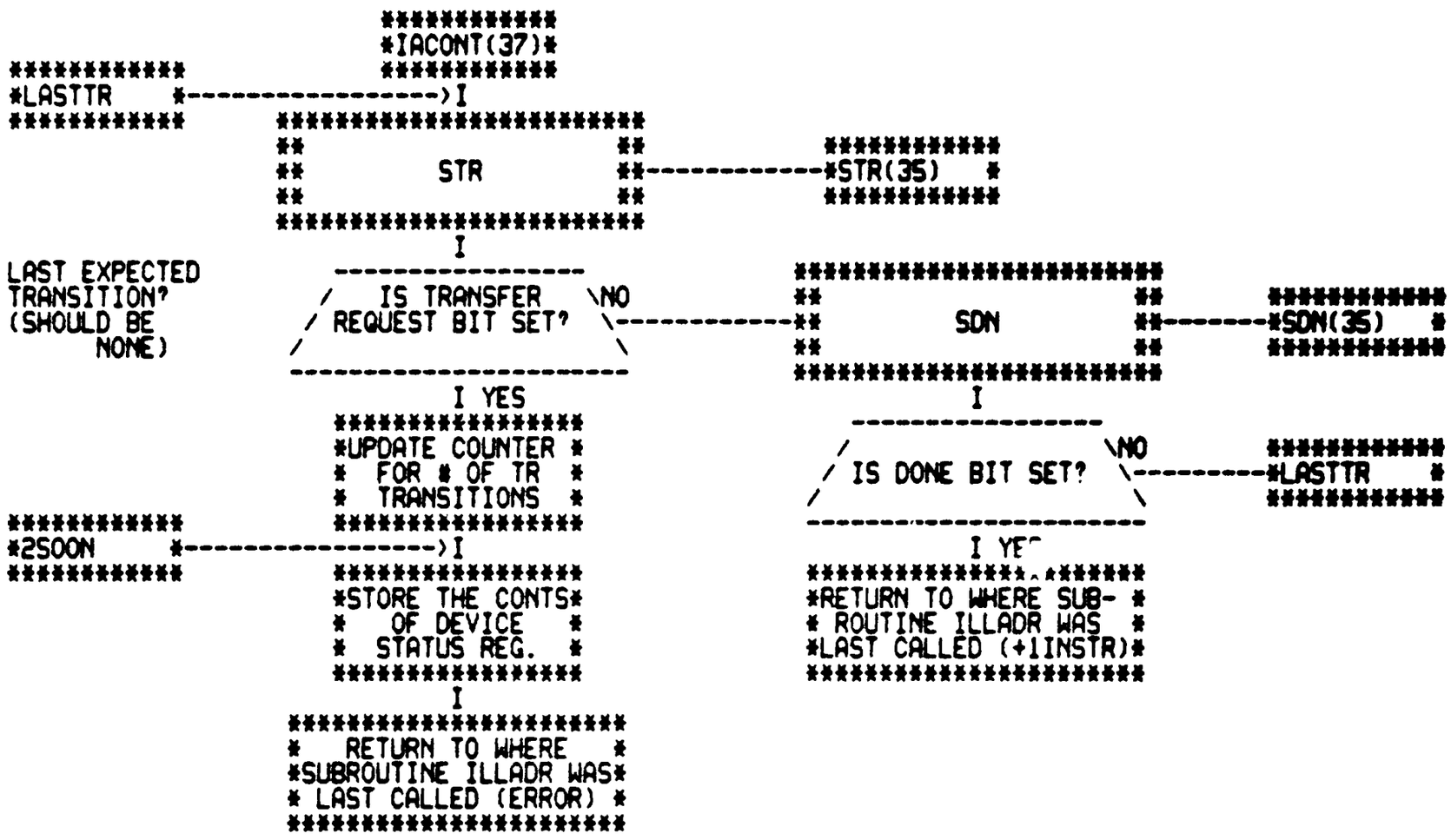
F) 314063

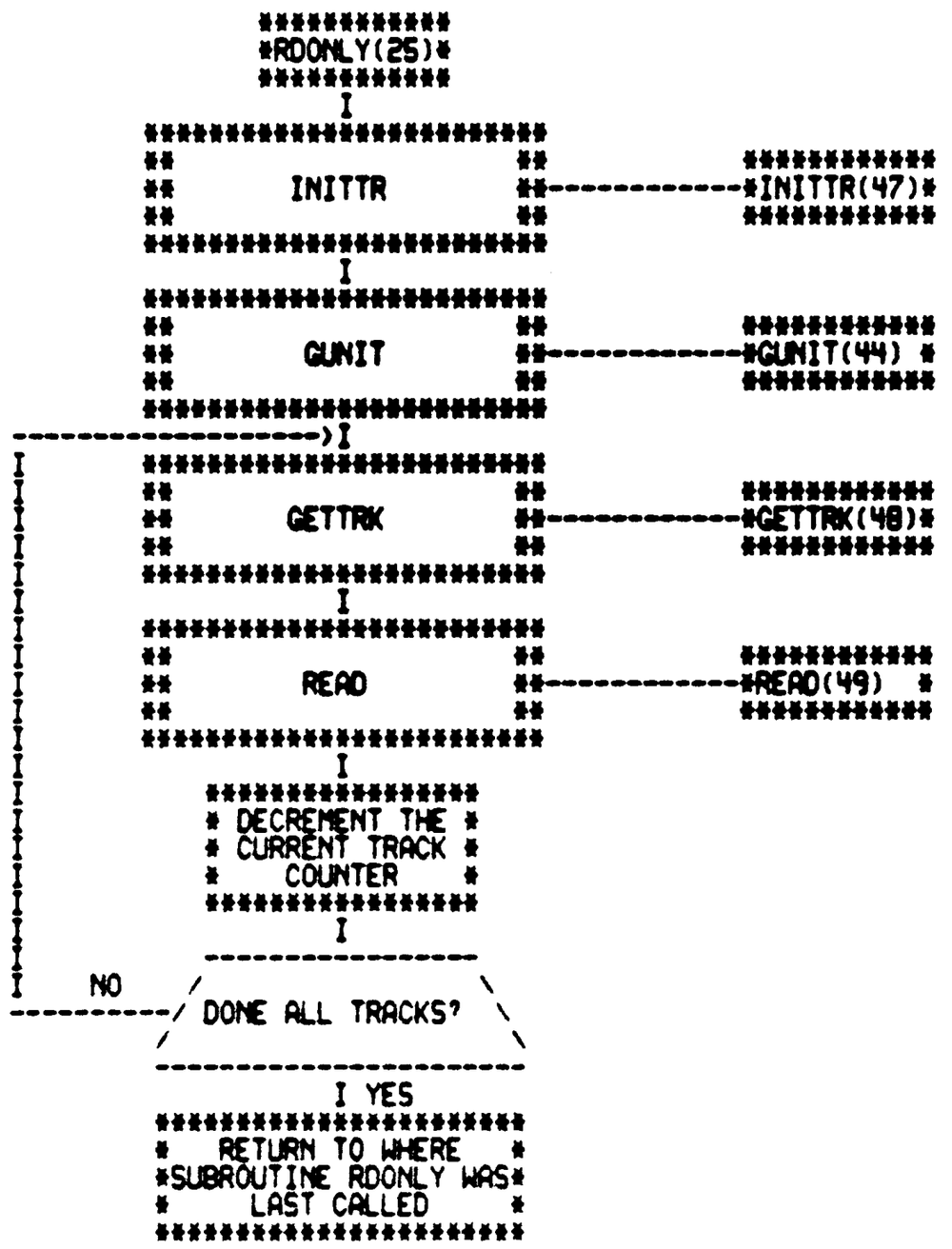
G) COUNT UP

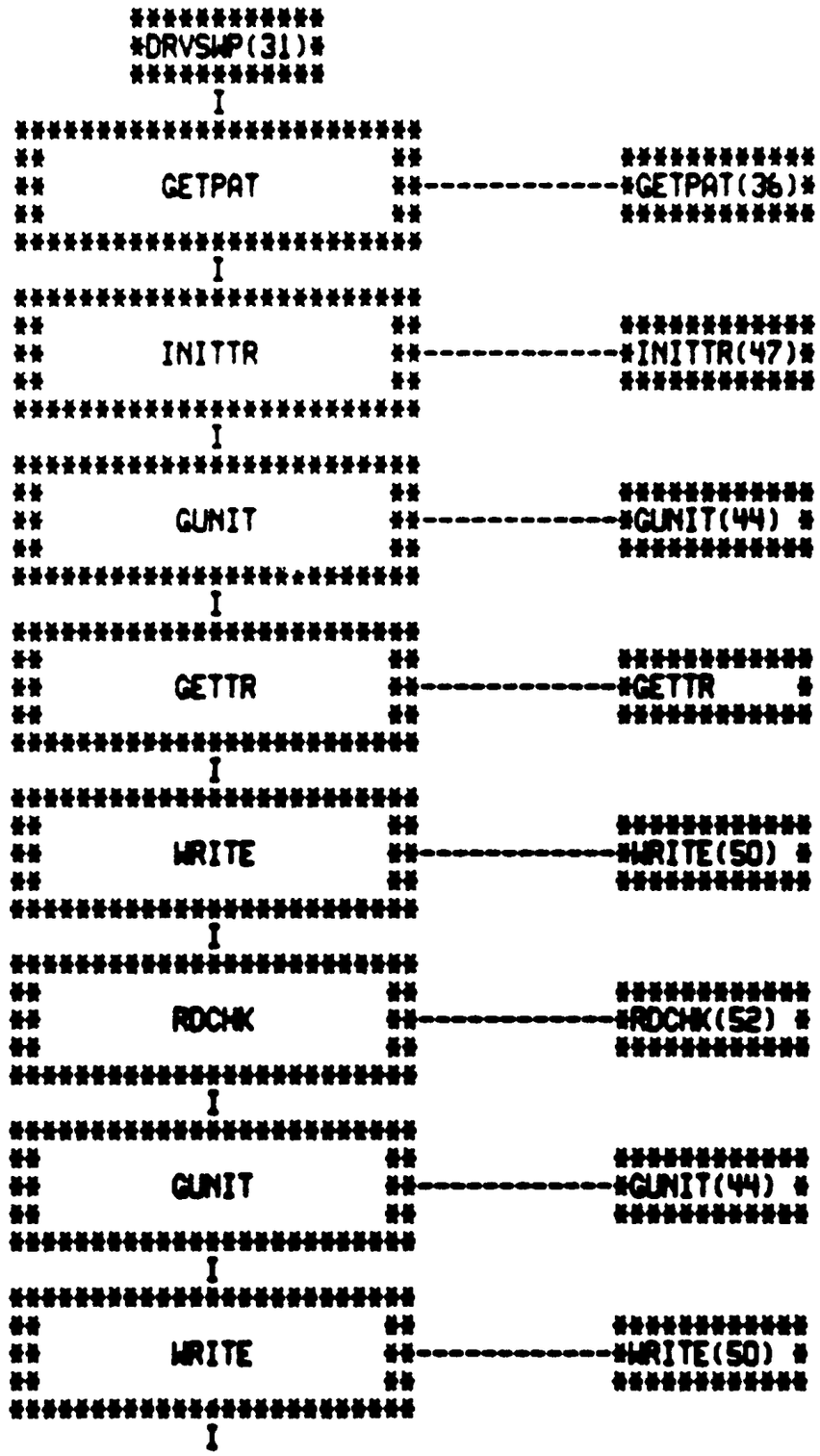
H) RANDOM

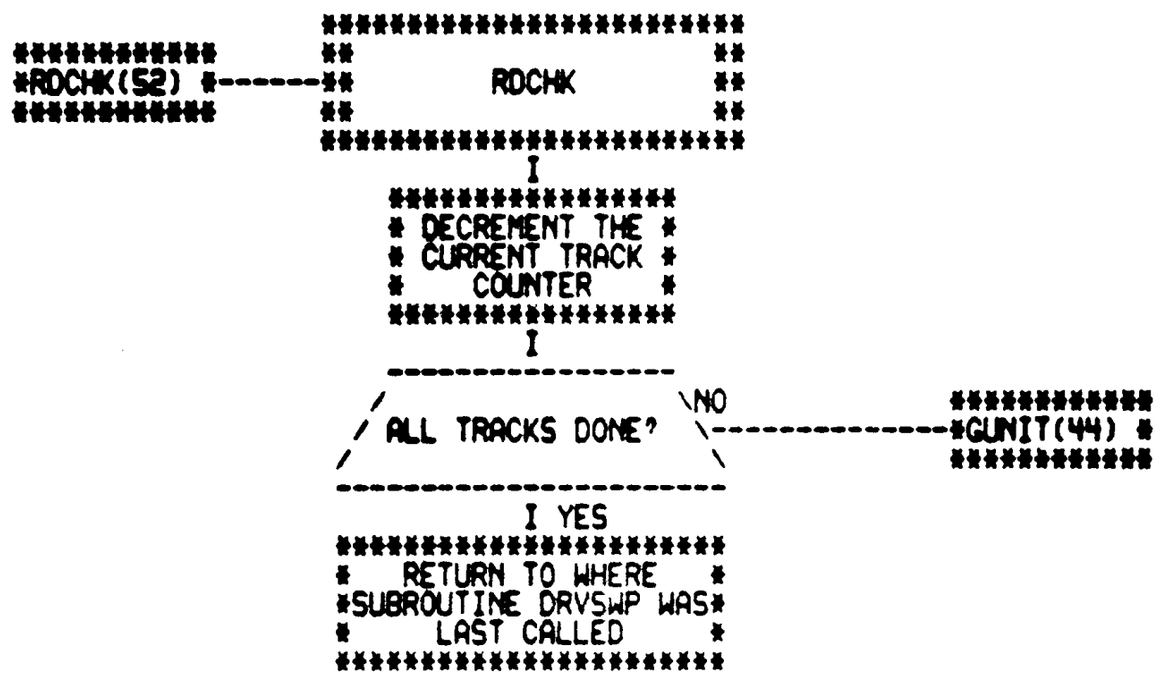


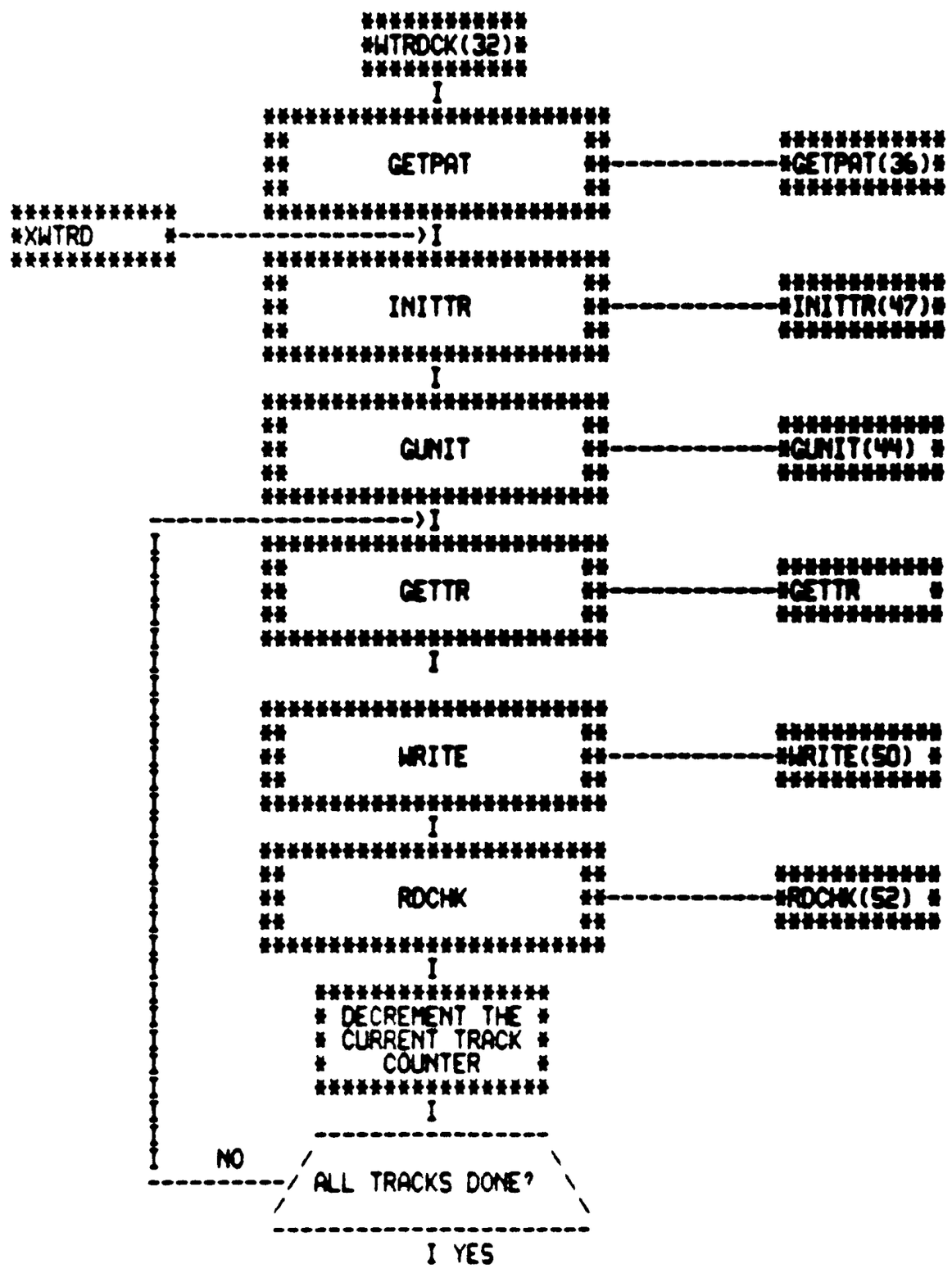

ILLADR(21)
*TROVER *
STR
*STR(35) *
IS TRANSFER REQUEST BIT SET?
NO
YES
1ST EXPECTED TRANSITION
SDN
*SDN(35) *
IS DONE BIT SET?
YES
NO
*2SOON *
*UPDATE COUNTER * * FOR # OF 'TR' * * TRANSITIONS *
* SET UP AN ILLEGAL * * SECTOR ADDRESS OF 0 *
*TROVER *
*TRAGAN *
STR
*STR(35) *
IS TRANSFER REQUEST BIT SET?
NO
YES
2ND EXPECTED TRANSITION
SDN
*SDN(35) *
IS DONE BIT SET?
YES
NO
*2SOON *
*UPDATE COUNTER * * FOR # OF "TR" * * TRANSITIONS *
* SET UP TRACK * * ADDRESS OF 0 * * (LEGAL) *
IACONT(38)
*TRAGAN *

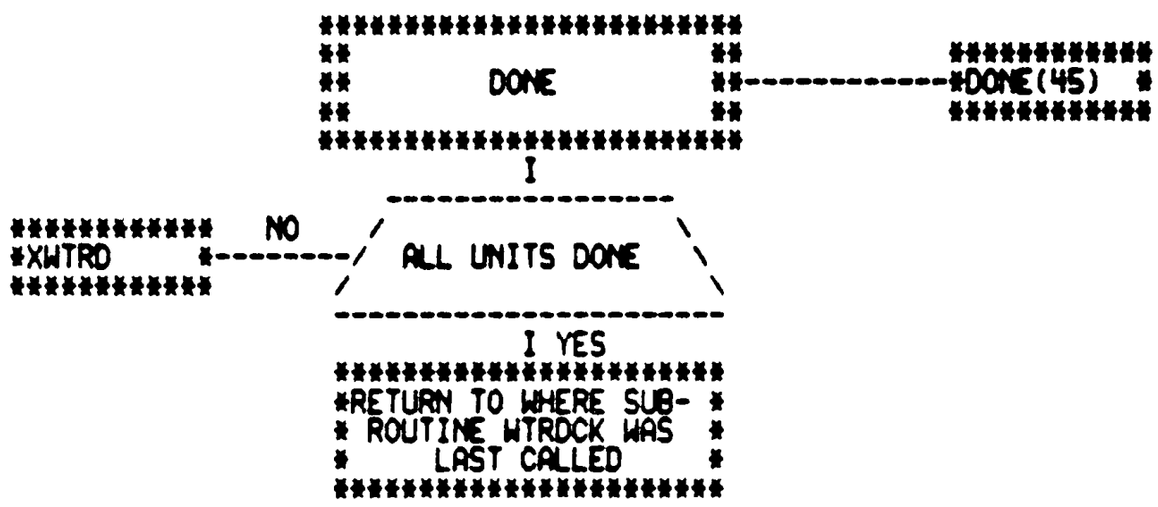

IACONT(37)
*LASTTR *
STR
*STR(35) *
LAST EXPECTED TRANSITION? (SHOULD BE NONE)
IS TRANSFER REQUEST BIT SET?
NO
SDN
*SDN(35) *
I YES
*UPDATE COUNTER * * FOR # OF TR * * TRANSITIONS *
IS DONE BIT SET?
NO
*LASTTR *
*2SOON *
STORE THE CONTS * OF DEVICE * * STATUS REG. *
I YE
*RETURN TO WHERE SUB- * * ROUTINE ILLADR WAS * *LAST CALLED (+1INSTR)*
* RETURN TO WHERE * *SUBROUTINE ILLADR WAS* * LAST CALLED (ERROR) *

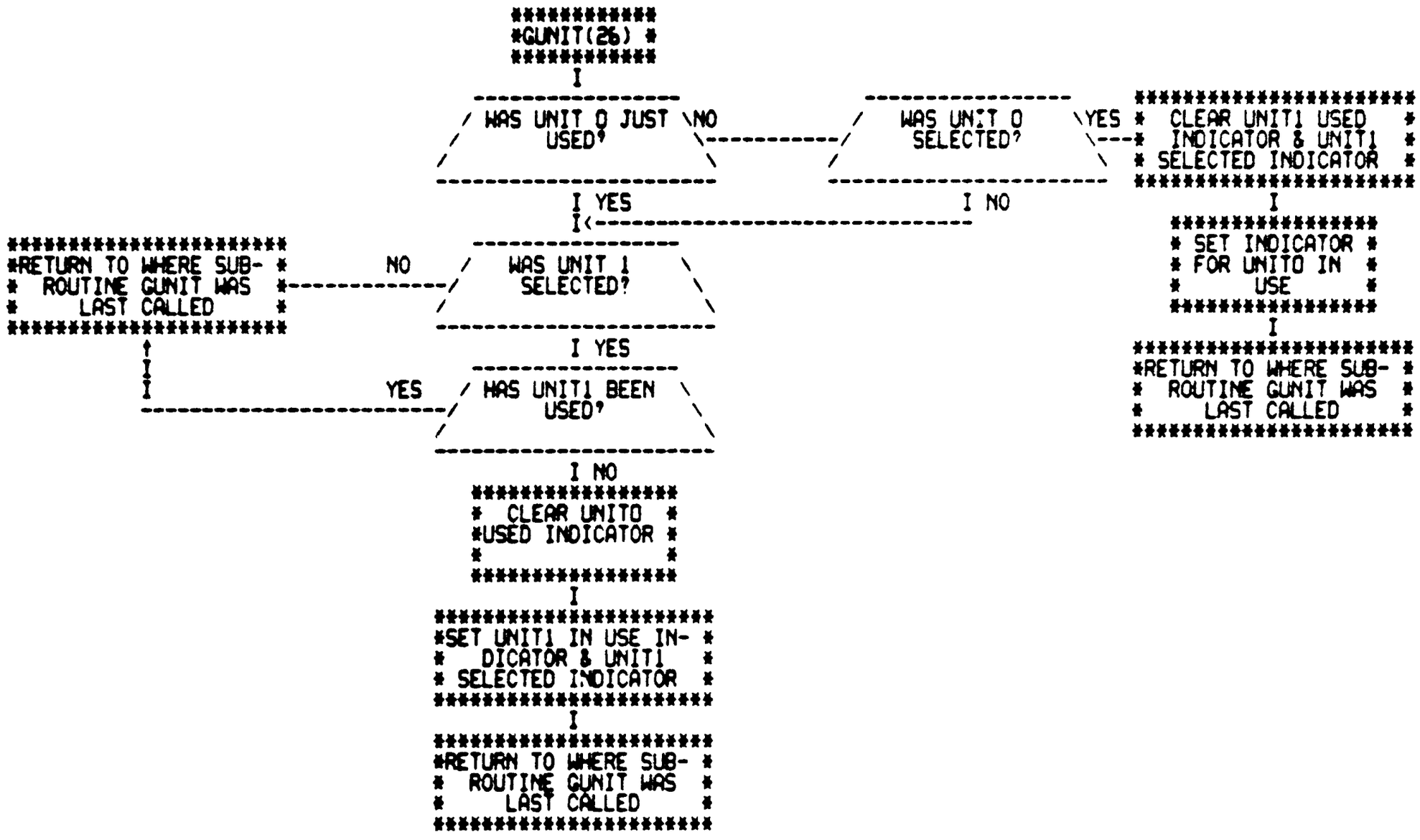

RDONLY(25)
INITTR
INITTR(47)
GUNIT
GUNIT(44)
GETTRK
GETTRK(48)
READ
READ(49)
DECREMENT THE CURRENT TRACK COUNTER
NO
DONE ALL TRACKS?
YES
RETURN TO WHERE SUBROUTINE RDONLY WAS LAST CALLED

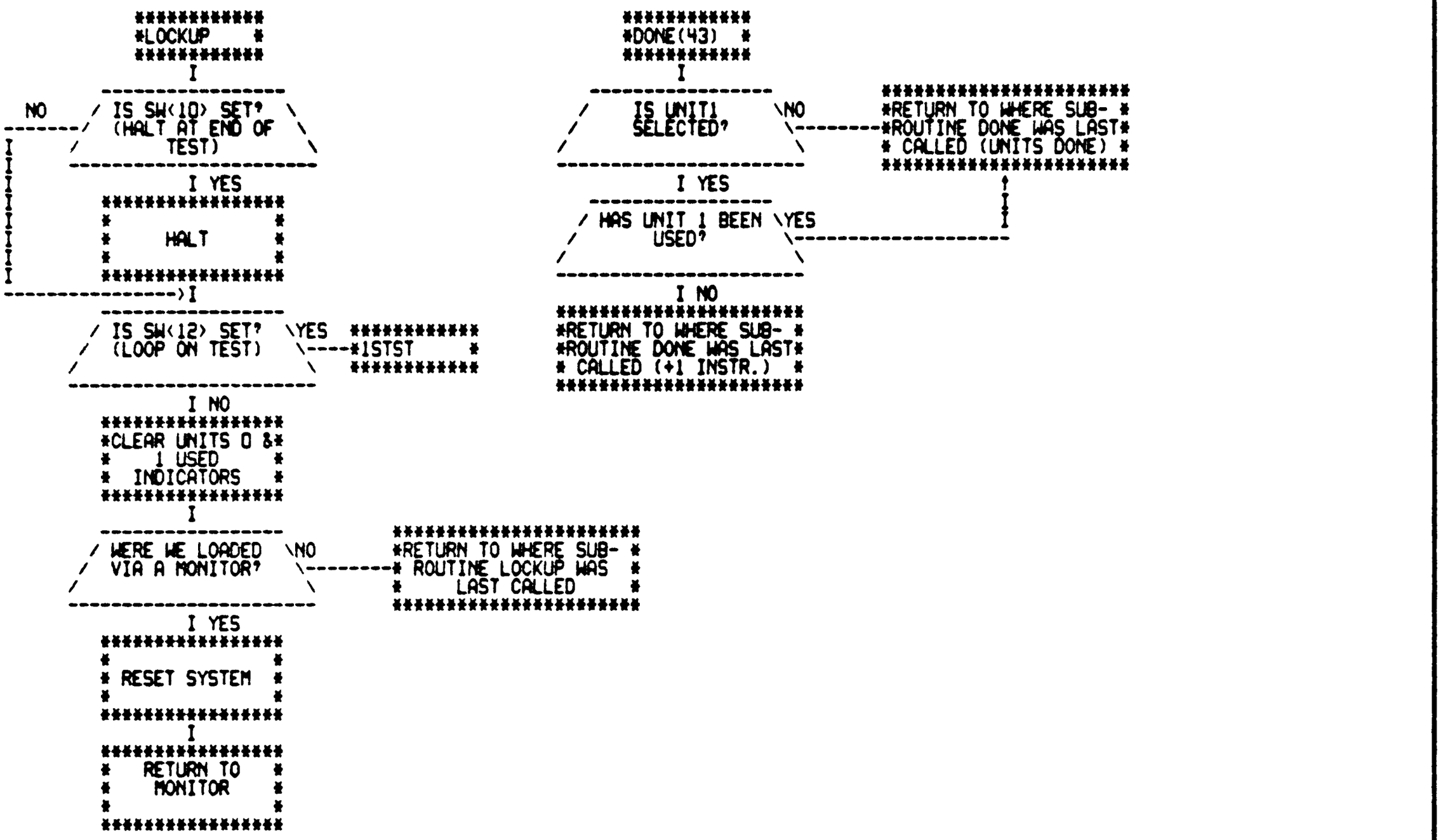

DRVSWP(31)
GETPAT
GETPAT(36)
INITTR
INITTR(47)
GUNIT
GUNIT(44)
GETTR
GETTR
WRITE
WRITE(50)
RDCHK
RDCHK(52)
GUNIT
GUNIT(44)
WRITE
WRITE(50)

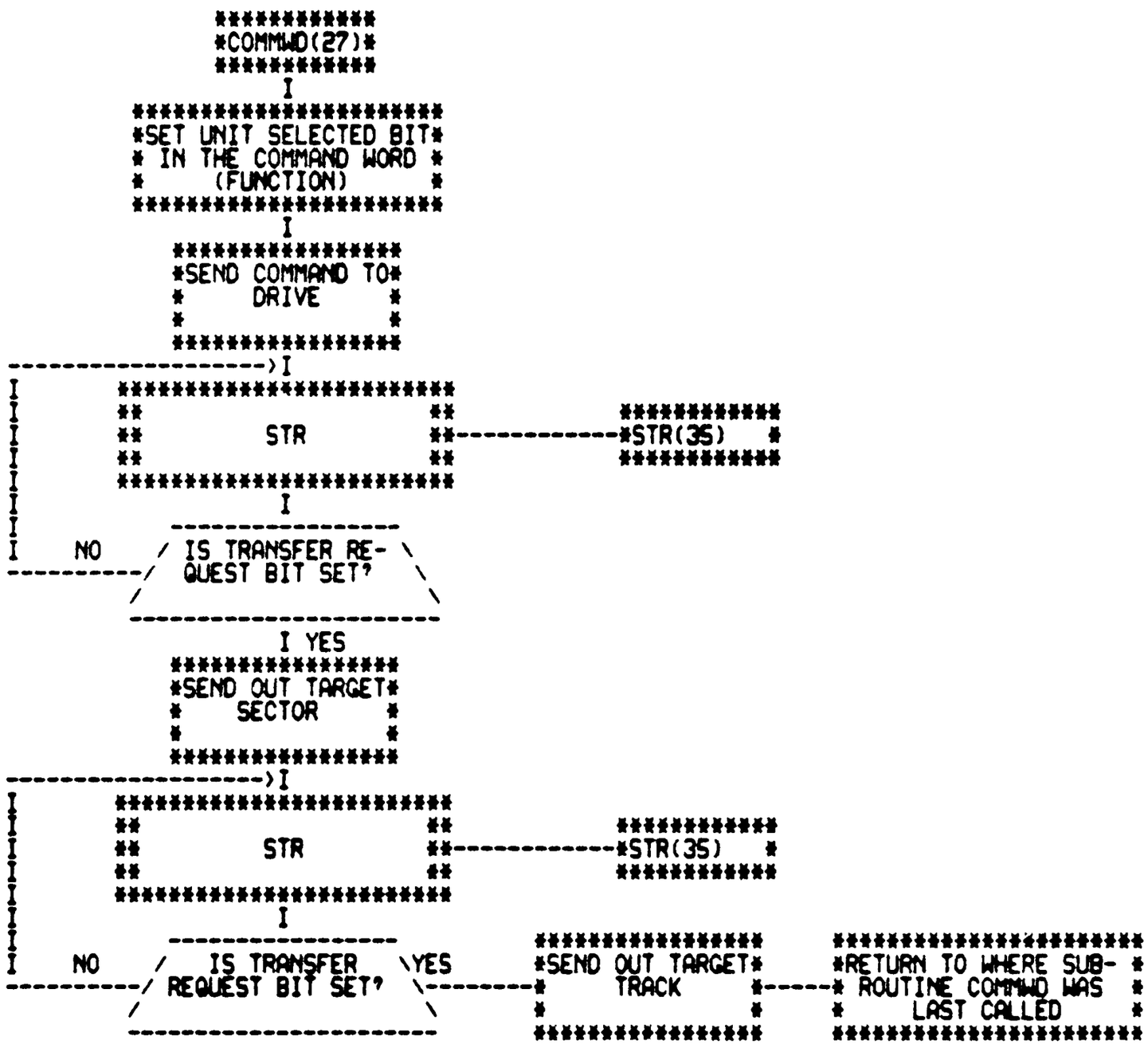

*RDCHK(52) *
RDCHK
DECREMENT THE CURRENT TRACK COUNTER
ALL TRACKS DONE?
NO
*GUNIT(44) *
YES
RETURN TO WHERE *SUBROUTINE DRVSWP WAS* LAST CALLED

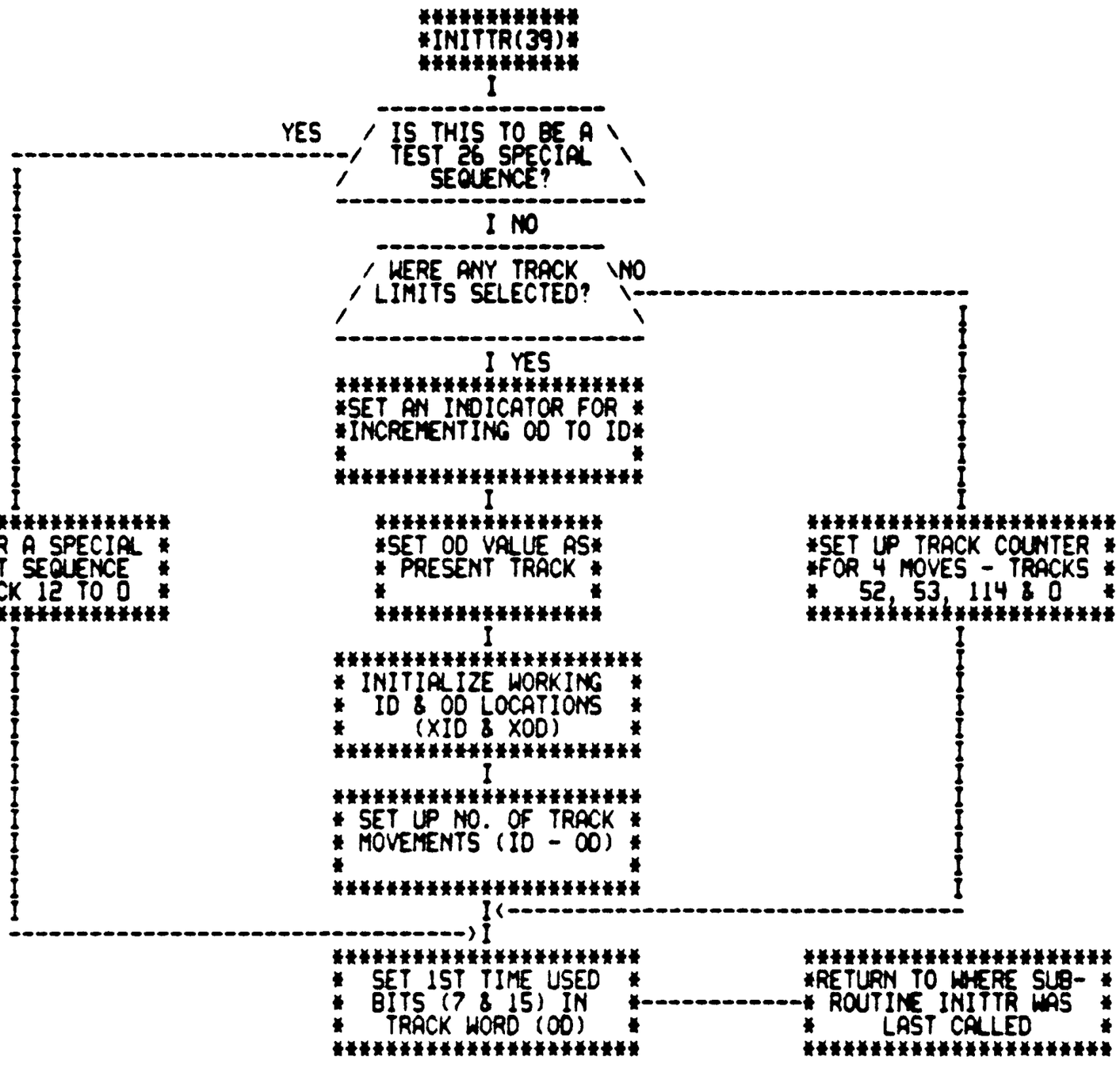

WTRDCK(32)
GETPAT
GETPAT(36)
XWTRD
INITTR
INITTR(47)
GUNIT
GUNIT(44)
GETTR
GETTR
WRITE
WRITE(50)
RDCHK
RDCHK(52)
DECREMENT THE CURRENT TRACK COUNTER
NO
ALL TRACKS DONE?
YES


DONE
DONE(45)
ALL UNITS DONE
NO
XWTRD
YES
RETURN TO WHERE SUB-
ROUTINE WTRDCK WAS
LAST CALLED

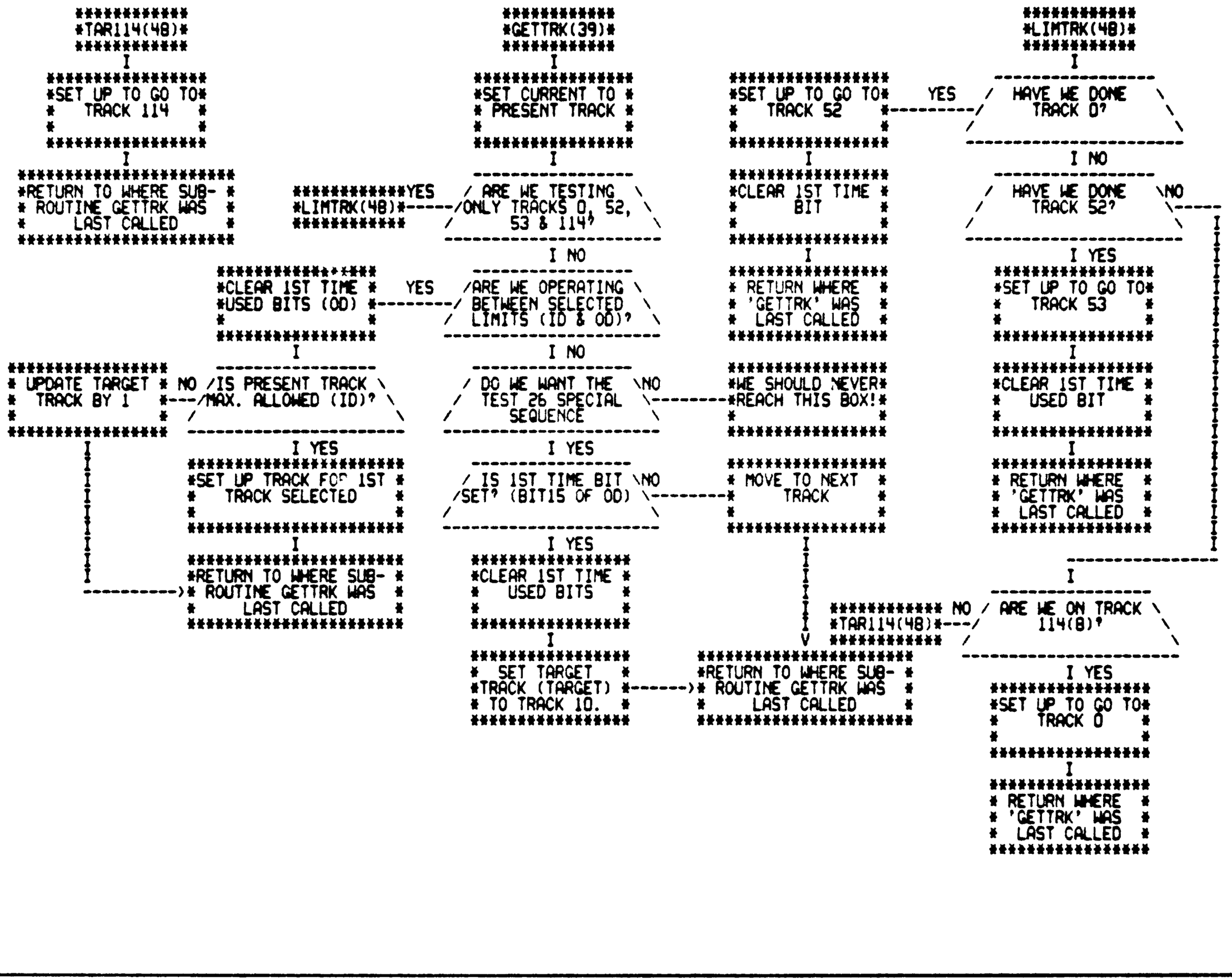

GUNIT(26)
WAS UNIT 0 JUST USED?
NO
WAS UNIT 0 SELECTED?
YES
CLEAR UNIT1 USED INDICATOR & UNIT1 SELECTED INDICATOR
YES
NO
SET INDICATOR FOR UNIT0 IN USE
RETURN TO WHERE SUB-ROUTINE GUNIT WAS LAST CALLED
NO
WAS UNIT 1 SELECTED?
YES
WAS UNIT1 BEEN USED?
YES
NO
CLEAR UNIT0 USED INDICATOR
SET UNIT1 IN USE INDICATOR & UNIT1 SELECTED INDICATOR
RETURN TO WHERE SUB-ROUTINE GUNIT WAS LAST CALLED


LOCKUP
IS SW(10) SET? (HALT AT END OF TEST)
NO
HALT
IS SW(12) SET? (LOOP ON TEST)
YES
1STST
CLEAR UNITS 0 & 1 USED INDICATORS
WERE WE LOADED VIA A MONITOR?
NO
RESET SYSTEM
RETURN TO MONITOR
RETURN TO WHERE SUB- ROUTINE LOCKUP WAS LAST CALLED
DONE(43)
IS UNIT1 SELECTED?
NO
RETURN TO WHERE SUB- ROUTINE DONE WAS LAST CALLED (UNITS DONE)
HAS UNIT 1 BEEN USED?
YES
RETURN TO WHERE SUB- ROUTINE DONE WAS LAST CALLED (+1 INSTR.)

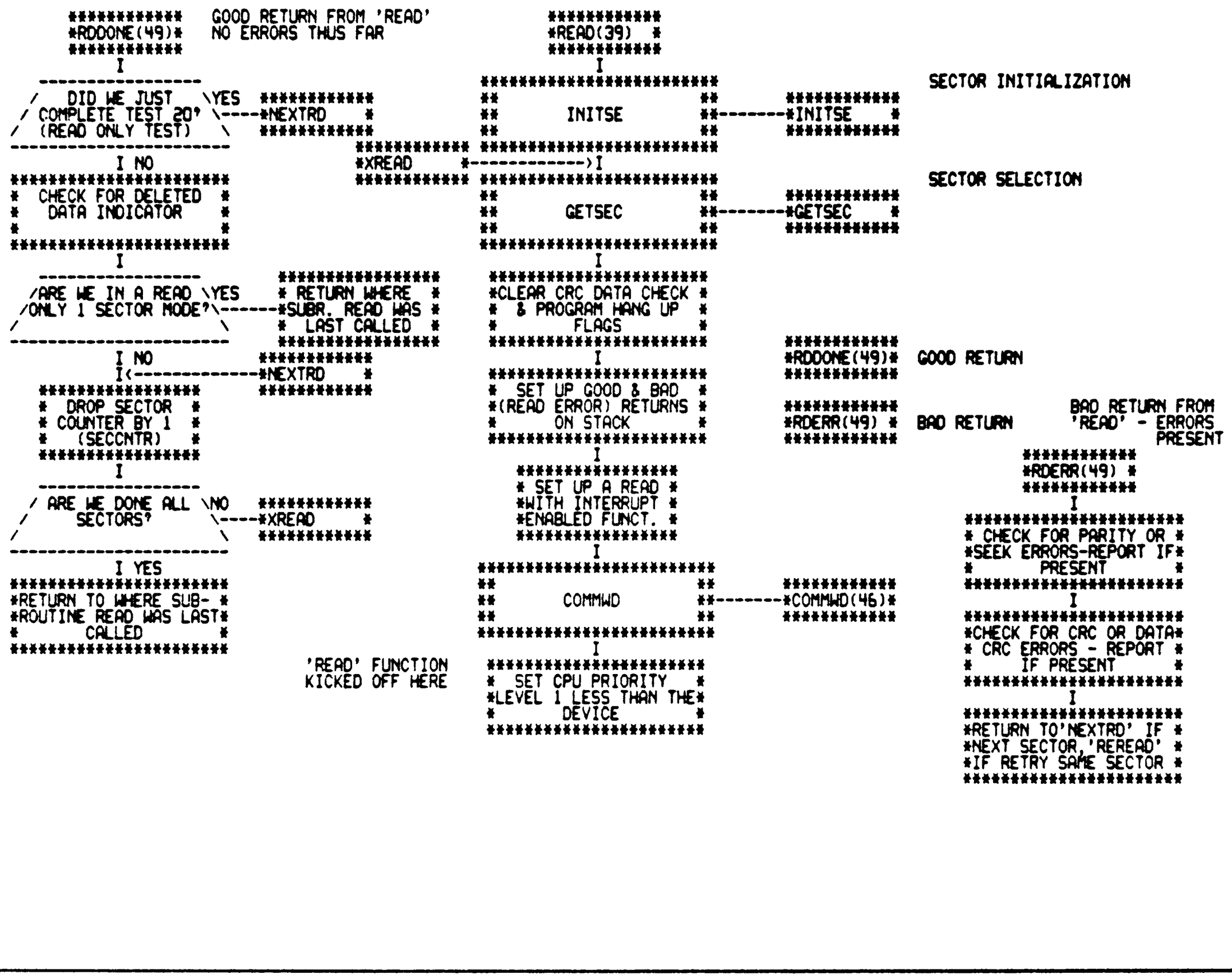

COMMWD(27)
SET UNIT SELECTED BIT IN THE COMMAND WORD (FUNCTION)
SEND COMMAND TO DRIVE
STR
STR(35)
IS TRANSFER REQUEST BIT SET?
NO
YES
SEND OUT TARGET SECTOR
STR
STR(35)
IS TRANSFER REQUEST BIT SET?
NO
YES
SEND OUT TARGET TRACK
RETURN TO WHERE SUB-ROUTINE COMMWD WAS LAST CALLED

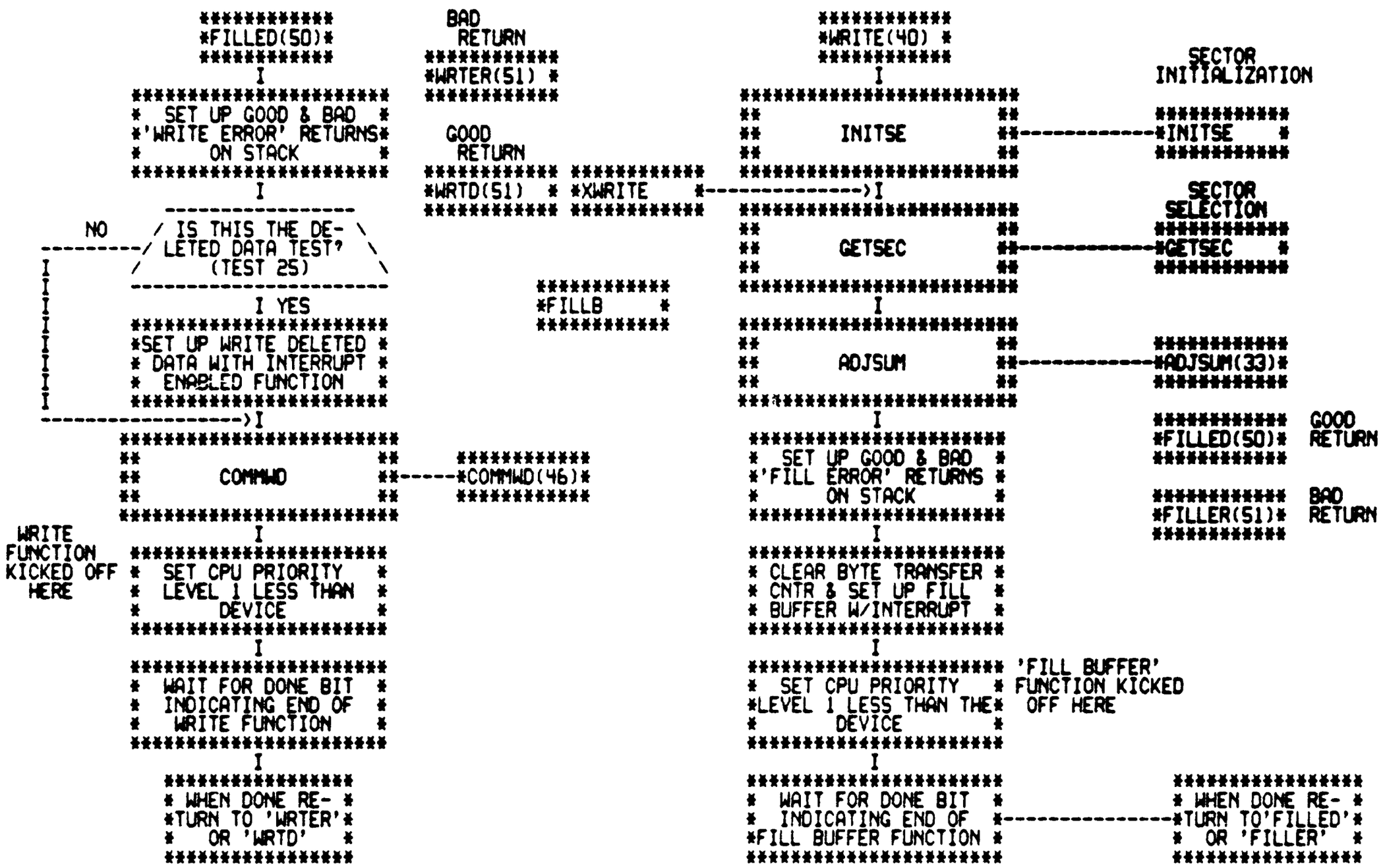

INITTR(39)
IS THIS TO BE A TEST 26 SPECIAL SEQUENCE?
YES
I NO
WERE ANY TRACK LIMITS SELECTED?
NO
I YES
SET AN INDICATOR FOR INCREMENTING OD TO ID
*SET OD VALUE AS PRESENT TRACK *
SET UP FOR A SPECIAL DECREMENT SEQUENCE FROM TRACK 12 TO 0
SET UP TRACK COUNTER FOR 4 MOVES - TRACKS 52, 53, 114 & 0
* INITIALIZE WORKING ID & OD LOCATIONS (XID & XOD) *
* SET UP NO. OF TRACK MOVEMENTS (ID - OD) *
*SET 1ST TIME USED BITS (7 & 15) IN TRACK WORD (OD) *
*RETURN TO WHERE SUB-ROUTINE INITTR WAS LAST CALLED *

K06

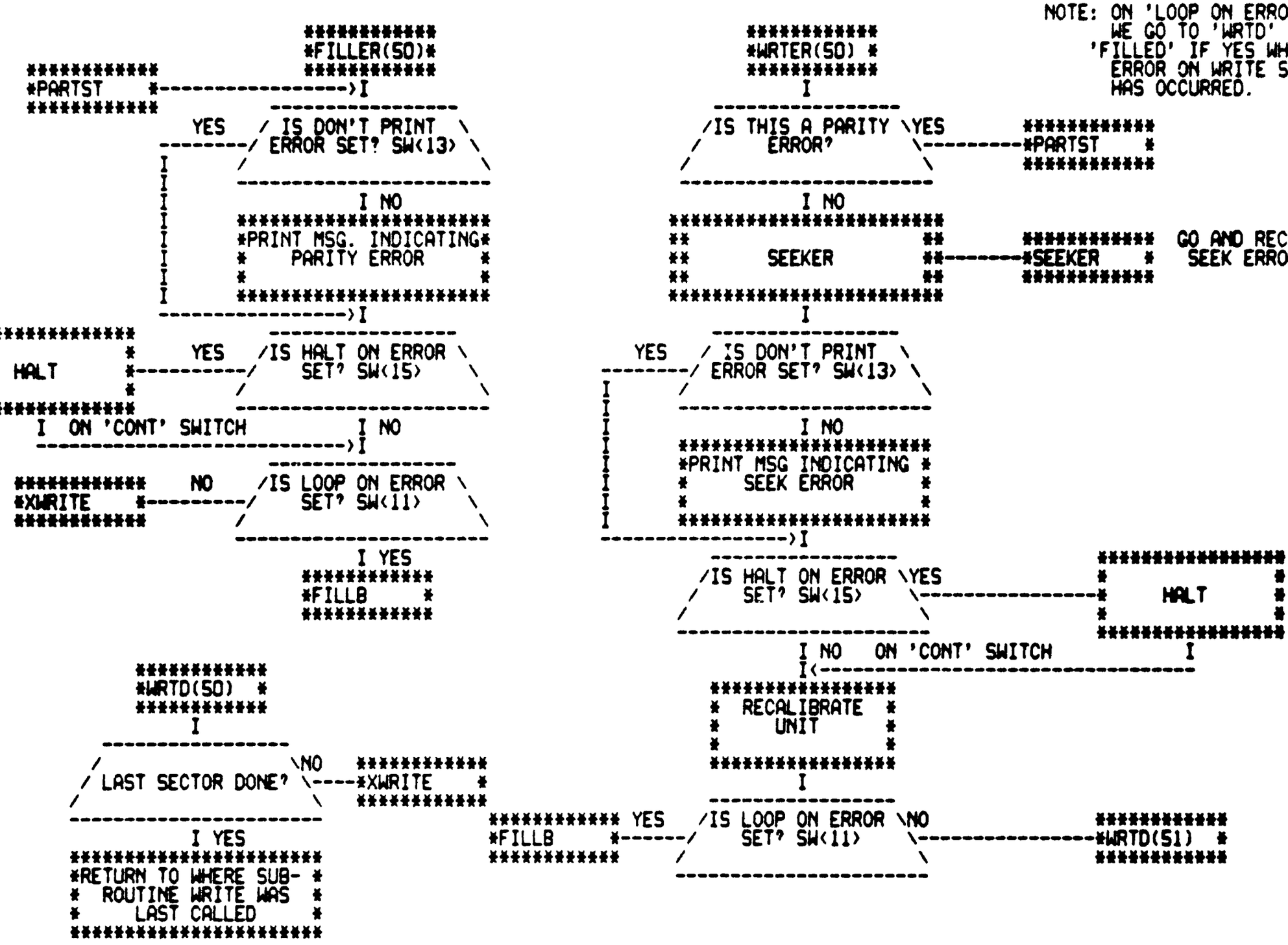

TAR114(48)
SET UP TO GO TO TRACK 114
RETURN TO WHERE SUB- ROUTINE GETTRK WAS LAST CALLED
GETTRK(39)
SET CURRENT TO PRESENT TRACK
ARE WE TESTING ONLY TRACKS 0, 52, 53 & 114?
YES
LIMTRK(48)
CLEAR 1ST TIME USED BITS (OD)
ARE WE OPERATING BETWEEN SELECTED LIMITS (ID & OD)?
UPDATE TARGET TRACK BY 1
IS PRESENT TRACK MAX. ALLOWED (ID)?
NO
SET UP TRACK FOR 1ST TRACK SELECTED
RETURN TO WHERE SUB- ROUTINE GETTRK WAS LAST CALLED
DO WE WANT THE TEST 26 SPECIAL SEQUENCE
IS 1ST TIME BIT SET? (BIT15 OF OD)
CLEAR 1ST TIME USED BITS
SET TARGET TRACK (TARGET) TO TRACK 10.
SET UP TO GO TO TRACK 52
CLEAR 1ST TIME BIT
RETURN WHERE 'GETTRK' WAS LAST CALLED
WE SHOULD NEVER REACH THIS BOX!
MOVE TO NEXT TRACK
ARE WE ON TRACK 114(8)?
HAVE WE DONE TRACK 0?
HAVE WE DONE TRACK 52?
SET UP TO GO TO TRACK 53
CLEAR 1ST TIME USED BIT
RETURN WHERE 'GETTRK' WAS LAST CALLED
SET UP TO GO TO TRACK 0
RETURN WHERE 'GETTRK' WAS LAST CALLED

READ SEQUENCE




FILLED(50)
SET UP GOOD & BAD 'WRITE ERROR' RETURNS ON STACK
IS THIS THE DELETED DATA TEST? (TEST 25)
NO
I YES
SET UP WRITE DELETED DATA WITH INTERRUPT ENABLED FUNCTION
COMMWD
COMMWD(46)
WRITE FUNCTION KICKED OFF HERE
SET CPU PRIORITY LEVEL 1 LESS THAN DEVICE
WAIT FOR DONE BIT INDICATING END OF WRITE FUNCTION
WHEN DONE RE-TURN TO 'WRTER' OR 'WRTD'
BAD RETURN
*WRTER(51) *
GOOD RETURN
*WRTD(51) *
*XWRITE *
*FILLB *
*WRITE(40) *
INITSE
GETSEC
ADJSUM
SET UP GOOD & BAD 'FILL ERROR' RETURNS ON STACK
CLEAR BYTE TRANSFER CNTR & SET UP FILL BUFFER W/INTERRUPT
SET CPU PRIORITY LEVEL 1 LESS THAN DEVICE
'FILL BUFFER' FUNCTION KICKED OFF HERE
WAIT FOR DONE BIT INDICATING END OF FILL BUFFER FUNCTION
WHEN DONE RE-TURN TO'FILLED' OR 'FILLER'
SECTOR INITIALIZATION
*INITSE *
SECTOR SELECTION
*GETSEC *
ADJSUM(33)
FILLED(50)
GOOD RETURN
FILLER(51)
BAD RETURN

FLOW CHART FOR RX11 DIAGNOSTIC INTERFACE
WRITE SEQUENCE




RDCHK(40)
READ A SECTOR
&
EMPTY SECTOR BUFFER
VERIFY DATA READ
AGAINST DATA IN CORE
BUFFER
EVENTUALLY RETURN TO
WHERE SUBR. RDCHK WAS
LAST CALLED

FLOW CHART CROSS REFERENCE LIST

FLOW CHART CROSS REFERENCE LIST

T14	21					
T14RDB	21	21				
T15	22					
T15RDB	22	22				
T16	23					
T16RDB	23	23				
T17	24					
T17RDB	24	24				
T2	09					
T20	25					
T21	26					
T21WR	26	26				
T21X	26	27	30			
T22	28					
T23	30					
T24	31					
T24X	31	31				
T25	31					
T26	32					
T3	11					
T4	12					
T5	13					
T6	16					
T6FILL	16	16	18	19	28	30
T7	17					
T7EMTY	17	17	18	19	27	29
TAR114	48	48				
TRAGAN	37	37				
TROVER	37	37				
TSTMOR	06	34	35	35		
WRITE	40	40	42	50		
WRTD	50	51	51			
WRTER	50	51				
WTROCK	32	42				
XREAD	49	49				
XSA202	01	02	03			
XWRITE	50	51	51			
XWTRD	42	43				
1STST	45					
1STTST	06					
2SOON	37	37	38			

E07

```
                                .NLIST CND,MD,MC
                                .LIST ME
                                .ENABL ABS,AMA
                                .MCALL .HEADER,.EQUAT,.STYPE,.STYPOCT,.STRAP,TYPOCS
                                .MCALL SETPRI,.SPOWER,STARS,.SSB2D,.SOB2D,.SSAVE,COMMENT
                                .MCALL  ENDCOMMENT


                        .TITLE  MAINDEC-11-DZRXB-D
                        ;*COPYRIGHT (C) DEC 21,1975
                        ;*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-B1),AUG 29,1975.
                        ;*
000001                  $TN=1
160000                  $SWR=160000     ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYPOUT


                                ;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
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             ;;PRICRITY 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
                
                
                
                
            000017                      RDER    =17             ; READ B CODE
            000040                      DONEBIT =40
            000101                      FBIE    =101            ; IE+FILL BUFFER
            000103                      EBIE    =103            ; IE+EMPTY BUFFER
            000105                      WRTIE   =105            ; IE+WRITE SECTOR
            000107                      RDIE    =107            ; IE+READ SECTOR
            000115                      WTDDIE  =115            ; IE+WRITE DD SECTOR
            040001                      RECAL   =40001
            000000                      OPEN    =0
                
            000000                      .=0
000000  000000  000000                          .WORD 0,0
                
            000004                      .=4
000004  005756                      .WORD   BUSERR          ;UNEXPECTED TIMEOUT TRAP PC
000006  000340                      .WORD   PR7             ;UNEXPECTED TIMEOUT TRAP PS
000010  006002                      .WORD   RESERR          ;UNEXPECTED RESERVED INSTRUCTION TRAP PC
000012  000340                      .WORD   PR7             ;UNEXPECTED RESERVED INSTRUCTION TRAP PS
                
            000020                      .=20
000020  006316                      XSCOPE
000022  000340                      PR7
000024  014044                      $PWRDN
000026  000340                      PR7
000030  006060                      XERROR
000032  000340                      PR7
000034  014004                      $TRAP                   ;ADDRESS OF TRAP SERVICE
000036  000340                      PR7
                
            000046                      .=46
000046  002434                      LOGICAL                 ;ACT 11 EOP HOOKS
                
                
            000052                      .=52
000052  000000                      .WORD 0
                
            000174                      .=174
000174  000000                      DISPREG:        0
000176  000000                      SWREG:          0
                
            000200                      .=200
000200  000401                              BR 1$
000202  000402                              BR 2$
000204  000137  001232          1$:         JMP $A200               ;OPERATOR SELECTED CONDITIONS
000210  000137  001222          2$:         JMP $A202               ;RESTART PROGRAM WITH PREVIOUS PARAMETERS
```

```
239
240
241
```

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

                       ; FUNCTION

                       ; 3 2 1 0

                       ; - - - GO

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

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

001210  177172                  RXDB:   177172

                                ; RXDB: STANDARD DEVICE ADRESS = 177172

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

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

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

001212  000000                  DTESTP: 0

001214  000005                  BRLEV:  5

                                ;BRLEV: STANDARD PRIORITY INTERRUPT LEVEL = 5

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

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

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

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

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

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

```
                                        .SBTTL START AND RESTART ADDRESSES


                                ; THE STARTING ADDRESS WAS 202

001222  005200                  SA202:  INC R0
001224  012706  001200                  MOV #STACK,SP
001230  000426                          BR RESTART

                                ; THE STARTING ADDRESS WAS 200

001232  005000                  SA200:  CLR R0
001234  012706  001200                  MOV #STACK,SP
001240  013746  000004                  MOV     4,-(SP)         ;SAVE 'BUSERR' TIMEOUT 'PC'
001244  012737  001264  000004          MOV #1$,4               ;SET UP TIMEOUT VECTOR
001252  022777  177777  177736          CMP #177777,@SWR        ;IS SOFTWARE SWR SELECTED
001260  001402                          BEQ 2$                  ;YES, INSERT IT'S ADDRESS
001262  000407                          BR 3$                   ;BR IF NO TIMEOUT TRAP OCCURS
001264  022626                  1$:     CMP (SP)+,(SP)+         ;RESTORE THE STACK
001266  012737  000176  001216  2$:     MOV #SWREG,SWR          ;POINT TO SOFTWARE SWITCH REGISTER
001274  012737  000174  001220          MOV #DISPREG,DISPLAY    ;POINT TO SOFTWARE DISPLAY REG.
001302  012637  000004          3$:     MOV (SP)+,4             ;RESET TIMEOUT VECTOR TO 'BUSERR'
001306  012746  000340          RESTART: MOV #PR7,-(SP)
001312  012746  001320                  MOV #4$,-(SP)
001316  000002                          RTI                     ;LOAD THE PSW
001320  013737  001206  001210  4$:     MOV RXCS, RXDB          ;GET ADDRESS OF RXCS
001326  062737  000002  001210          ADD #2, RXDB            ;SET UP ADDRESS OF RXDB
001334  012737  001234  012350          MOV #001234,RAN1        ; INITIALIZE CONSTANTS OF
001342  012737  000765  012352          MOV #000765,RAN2        ; RANDOM NUMBER GENERATOR
001350  005037  002336                  CLR CCOUNT
001354  005037  002340                  CLR PASS
001360  005037  006560                  CLR HANGER
001364  005700                          TST R0
001366  001040                          BNE XSA202              ; STARTING ADDRESS WAS 202
001370  104400  015605                  TYPE, MREV              ; PRESENT MAINDEC REVISION
001374  005037  012446                  CLR UNITSEL
001400  032737  140000  001212          BIT #140000,DTESTP      ;WERE ANY DRIVES SELECTED
001406  001004                          BNE 1$                  ;YES GO SET THEIR BITS
001410  052737  100200  012446          BIS #100200,UNITSEL     ;NO, BOTH UNITS MUST BE READY
001416  000424                          BR XSA202
001420  032737  040000  001212  1$:     BIT #BIT14,DTESTP       ;WAS UNIT 0 SELECTED
001426  001415                          BEQ 3$                  ;NO, MUST BE UNIT 1
001430  123727  000041  000010          CMPB    41,#10          ;WAS PROGRAM LOADED IN DUMP MODE
                                                                ;VIA XXDP?
001436  001003                          BNE     2$              ;BRANCH IF NOT
001440  104400  016125                  TYPE,   DOLOAD          ;INFORM USER TO REMOVE LOAD MEDIUM
                                                                ;FROM UNIT 0 AND REPLACE WITH
                                                                ;A 'SCRATCH' DISKETTE IF HE
                                                                ;WISHES TO TEST UNIT 0
001444  000000                          HALT                    ;WAIT FOR USER RESPONSE
001446  052737  000200  012446  2$:     BIS #200,UNITSEL        ;YES, SET SELECTED BIT
001454  005737  001212                  TST DTESTP              ;WAS UNIT 1 SELECTED
001460  100003                          BPL XSA202              ;NO
001462  052737  100000  012446  3$:     BIS #BIT15,UNITSEL      ;SET SELECTED BIT
001470  042737  100200  001200  XSA202: BIC     #100200,@0      ;CLEAR 1ST TIME BITS FOR BOTH DRIVES
```

```
                                        TYPE, MCRLF
                                        TYPE, MDTESTP
                                        MOV     DTESTP,-(SP)    ;;SAVE DTESTP FOR TYPEOUT
                                        TYPOS                   ;;GO TYPE--OCTAL ASCII
                                        .BYTE   6               ;;TYPE 6 DIGIT(S)
                                        .BYTE   0               ;;SUPPRESS LEADING ZEROS
                                        TYPE ,MCRLF
                                LIMITS: TST OD
                                        BNE TRKLMT
                                        CLR SEQUEN
                                        TYPE ,MLIMTRK
                                        BR SECLMT

                                ; 0 <= OD <= ID <= 114

                                TRKLMT: CMPB ID,#114
                                        BHI 1$
                                        CMPB OD,ID
                                        BHI 1$
                                        TYPE ,MOD
                                        MOV     OD,-(SP)        ;;SAVE OD FOR TYPEOUT
                                        TYPOS                   ;;GO TYPE--OCTAL ASCII
                                        .BYTE   3               ;;TYPE 3 DIGIT(S)
                                        .BYTE   0               ;;SUPPRESS LEADING ZEROS
                                        TYPE ,MID
                                        MOVB ID,-(SP)
                                        TYPON
                                        BR SECLMT
                                1$:     TYPE,   OO2BIG          ;TYPE MSG. INDICATING ID OR OD
                                                                ;TOO BIG & DEFAULTING TO TRACKS
                                                                ;0, 52, 53, 114
                                        CLR OD
                                        BR LIMITS

                                ; 1 <= FIRST <= LAST <= 32

                                SECLMT: TSTB FIRST
                                        BNE 1$
                                        MOVB #1,FIRST
                                1$:     CMPB LAST,#32
                                        BHI 2$
                                        CMPB FIRST,LAST
                                        BHI 2$
                                3$:     TYPE ,MFIRST
                                        MOVB FIRST,-(SP)
                                        TYPON
                                        TYPE ,MLAST
                                        MOVB LAST,-(SP)
                                        TYPON
                                        TYPE ,MCRLF
                                        BR PRETEST
                                2$:     TYPE,   52BIG           ;TYPE MSG. INDICATING THAT
                                                                ;SECTOR RANGE SELECTED WAS
                                                                ;INVALID AND DEFAULTING TO A
```

```
   499
   500                                                          ;A 1ST SECTOR VALUE OF 1 AND
   501                                                          ;A LAST SECTOR VALUE OF 32
   501  001716  012737  015001  001202          MOV #15001,FIRST  ;SET FIRST TO 1 LAST TO 32
   502  001724  000755                          BR 3$
```

```
                                        .SBTTL PRETEST - INITIALIZE (KEY) PART I
                                ;
                                ;       "KEY" INITIALIZE SHOULD HAVE (SET) THE DONE FLAG BECAUSE
                                ;       ANY (INIT) OF THE RX01 MICROCONTROLLER IS AN IMPLIED (READ SECTOR)
                                ;       OF TRACK 1 SECTOR 1 (FOR SYSTEMS PROGRAMMING BOOTSTRAP APPLICATIONS).
                                ;
                                ;       THEREFORE, ANY ERROR (EXCEPT PARITY) THAT MAY OCCUR FROM A NORMAL
                                ;       "READ SECTOR" COMMAND MAY OCCUR HERE CAUSING THE ERROR FLAG TO SET, AND
                                ;       DISPLAYING THE ERROR STATUS WITHIN THE TRANSFER REGISTER AT "DONE".
                                ;
                                ;       THE TRANSFER REQUEST FLAG SHOULD BE CLEARED.
                                ;
                                ;NOTE:  SCOPE LOOPING IS NOT OFFERED BECAUSE THE "INIT" FUNCTION
                                ;       WHICH PRODUCED THE ERROR HAS NOT YET BEEN VERIFIED.
                                ;
001726  042737  000400  012446  PRETEST: BIC #BIT8,UNITSEL
001734  005037  006312                  CLR ERRORS
001740  000004                          SCOPE                   ; REFRESHES FAST FOR ERROR TYPEOUT
001742  013737  006374  002342          MOV PCSCOPE,FAST        ; FOR ERROR TYPEOUT INFORMATION
001750  012700  000040                  MOV #40, R0             ; PROGRAM EXPECTS DONE FLAG
001754  017701  177226                  MOV @ RXCS, R1          ; ACTUAL CONTENTS OF RXCS
001760  020100                          CMP R1, R0
001762  001401                          BEQ 1$                  ; OK
                                ; (R0) = 40 ; (R1) = ACTUAL RXCS ; (R2) = N/A
001764  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

;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

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

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

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

001766  012700  000204        1$:     MOV #204,R0                             ;EXPECT INIT DONE AND UNIT 0 RDY
001772  017702  177212                MOV @RXDB,R2
001776  010201                       MOV R2,R1
002000  042701  000100                BIC #BIT6,R1                            ;CLEAR DELETED DATA BIT
002004  105737  012446                TSTB UNITSEL                            ;WAS UNIT 0 SELECTED
002010  100404                       BMI 2$
002012  042701  000200                BIC #BIT7,R1                            ;UNIT 0 WAS NOT SELECTED
002016  042700  000200                BIC #BIT7,R0                            ;CLEAR THE DRIVE 0 RDY BITS
002022  020100                2$:     CMP R1,R0
002024  001524                       BEQ REBEGIN

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

002026  104000                       ERROR                                   ; RXDB NOT = 4, OR 204


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

                ;THE FOLLOWING IS A PRESENTATION OF POSSIBLE REASONS AS TO WHY
                ;THIS ERROR REPORT WAS GENERATED.  THE INFORMATION SHOWN IS
                ;BASED ON FAULT INSERTION RESULTS,  AND SHOULD PROVIDE LOGICAL
                ;AREAS TO CHECK FOR THE RELEVANT FAULT/S.
                ;
                ;IF THIS ERROR REPORT WAS THE 1ST GIVEN IN A SERIES OF ERRORS
                ;ANALYZE THE FOLLOWING AREA/S:
        ;
        ;M7846 (UNIBUS INTERFACE)
        ;
        ;       SIGNAL NAME             REASON                  POSSIBLE CHIPS
        ;
        ;B SHIFT                 STUCK LOW/HIGH                  E15,E19,E2
        ;B DONE                  STUCK HIGH                      E15
        ;TO RXD1                 STUCK LOW/HIGH                  E15
        ;LOAD                    STUCK HIGH                      E17,E18
```

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

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

706  002030  000522                        BR      REBEGIN
707  002032  004737  002366        MORETESTS: JSR PC, LOCKUP
708  002036  005723                TONOTHERE: TST (R3)+                    ;ADJUST R3 FOR NEXT TEST ADDRESS
709  002040  016337  002064  006374 FIRSTTEST: MOV TESTS(R3), PCSCOPE       ; EQUIVALENT TO " SCOPE "
710  002046  013737  006374  002342          MOV PCSCOPE,FAST              ;SAVE THE FIRST ADDRESS OF THE TEST
711  002054  012706  001200                  MOV #STACK,SP
712  002060  000173  002064                  JMP @TESTS(R3)
713  002064  002036  002446  002566  TESTS:  TONOTHERE, T1, T2, T3, T4, T5, T6, T7
714  002072  003110  003246  003412
715  002100  003636  003770
716  002104  003766  004054  004044          T10, T11, T12, T13, T14, T15, T16, T17
717  002112  004120  004244  004436
718  002120  004576  005044
```

```
719  002124  005310  005334  005520          T20, T21, T22, T23, T24, T25, T26, NOMORETESTS
720  002132  005576  005644  005676
721  002140  005710  002144
722
723                                  ; TEST 1 - RXCS TEST PART I / INTERRUPT TEST PART I
724
725                                  ; TEST 2 - INTERRUPT TEST PART II / VECTOR ADDRESS VERIFICATION
726
727                                  ; * TEST 3 - INTERRUPT TEST PART III / PRIORITY LEVEL VERIFICATION PART I
728
729                                  ; * TEST 4 - INTERRUPT TEST PART IV / PRIORITY VERIFICATION PART II
730
731                                  ; TEST 5 - INIT (PROGRAMMED) / RST
732
733                                  ; TEST 6 - FILL BUFFER TRANSFER LENGTH VERIFICATION
734
735                                  ; TEST 7 - EMPTY BUFFER TRANSFER LENGTH AND CONTENT VERIFICATION PART I
736
737                                  ; TEST 10 - EMPTY BUFFER TRANSFER LENGTH AND CONTENT VERIFICATION PART II
738
739                                  ; TEST 11 - FILL/EMPTY BUFFER ALL 0'S
740
741                                  ; TEST 12 - FILL/EMPTY BUFFER ALL 1'S
742
743                                  ; TEST 13 - DRIVE READY VERIFICATION FOR SELECTED DRIVES
744
745                                  ; TEST 14 - ERROR FLAG AND B - CODE VERIFICATION PART I
746
747                                  ; TEST 15 - ERROR FLAG AND B - CODE VERIFICATION PART II
748                                  ;       /DELETED DATA BIT SETS
749
750                                  ;TEST 16 - ERROR FLAG AND B - CODE VERIFICATION PART III
751                                  ;       /DELETED DATA BIT CLEARS
752
753                                  ;TEST 17 - ILLEGAL TRACK ERROR AND B - CODE VERIFICATION
754
755                                  ;TEST 20 - SEEK VERIFICATION VIA READ FUNCTION
756
757                                  ;TEST 21 - WRITE TEST
758
759                                  ;TEST 22 - INITIALIZE IMPLIED READ
760
761                                  ;TEST 23 - READ TEST
762
763                                  ;TEST 24 - DATA TRANSFER & VERIFICATION
764
765                                  ;TEST 25 - DATA TRANSFER & VERIFICATION VIA DELETED DATA MODE
766
767                                  ;TEST 26 - HEAD "HOME" TEST
768
769                                  ;THERE ARE NO MORE TESTS
770
771                                  ; * NOTE: ON PROCESSORS WITHOUT HARDWARE PROCESSOR STATUS WORDS (PSW)
772                                  ;       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
                                    ;
002144  042777  000100  177034  NOMORETESTS:  BIC  #BIT6,@RXCS     ;CLEAR 'IE' BIT BEFORE NEXT PASS
002152  032737  000400  012446          BIT #BIT8,UNITSEL           ;"C" OR "D" MEANS ERRORLESS PASS.
002160  001403                          BEQ 1$
002162  012737  000055  002344          MOV #'-, MX                 ; " - " MEANS UN-ERRORLESS PASS
002170  005737  002336          1$:     TST CCOUNT
002174  001002                          BNE 3$
002176  104400  015030                  TYPE, MCRLF
002202  005237  002336          3$:     INC CCOUNT
002206  022737  000110  002336          CMP #72., CCOUNT
002214  001002                          BNE 4$
002216  005037  002336                  CLR CCOUNT
002222  104400  002344          4$:     TYPE, MX
002226  104400  006310                  TYPE ,MABELL
002232  005237  002340          2$:     INC PASS
002236  102775                          BVS 2$
002240  032777  040000  176750          BIT #SW14,@SWR              ; AC SW 14 = 1 TO HALT AT END OF PASS
002246  001413                          BEQ REBEGIN
002250  104400  015030                  TYPE ,MCRLF
002254  104400  006527                  TYPE ,MPASS
002260  013737  002340  002272          MOV PASS,5$
002266  004537  014504                  JSR R5,SGLDEC
002272  000000                  5$:     OPEN
002274  000000                          HALT
002276  042737  000400  012446  REBEGIN: BIC #BIT8,UNITSEL          ;CLEAR HARD ERROR INDICATOR
002304  013703  001212                  MOV DTESTP, R3
002310  042703  177740                  BIC #177740, R3             ; R3 CONTAINS TEST # 0 TO 26
002314  020327  000027                  CMP R3,#27
002320  103002                          BHIS 1$
002322  006303                          ASL R3
002324  000645                          BR FIRSTTEST
                                    ;
002326  104400  002346          1$:     TYPE ,MILTST
002332  000137  001232                  JMP SA200
                                    ;
002336  000000                  CCOUNT: 0
002340  000000                  PASS:   0
002342  000000                  FAST:   0
                                    ;
002344  000103                  MX:     .ASCIZ "C"
                                    ;
002346  046111  042514  040507  MILTST: .ASCIZ "ILLEGAL TEST"<15><12>
002354  020114  042524  052123
002362  005015     000
                                    ;
```

```
   827          002366                          .EVEN
```

```
                                        ; DATA SW 10 = 1 TO HALT AT END OF TEST
002366  032777  002000  176622  LOCKUP: BIT #SW10,@SWR
002374  001401                        BEQ 1$
002376  000000                        HALT
                                        ; DATA SW 12 = 1 TO LOCK SCOPE LOOP ON TEST OK OR NOT
002400  032777  010000  176610  1$:     BIT #SW12,@SWR          ;IS LOOP ON TEST SWITCH SET
002406  001403                        BEQ 2$                  ;IF NOT SET GO ON TO NEXT TEST
002410  062716  000002                ADD #2,@SP              ;IF SET RETURN TO FIRSTTEST
002414  000207                        RTS PC
002416  042737  040100  012446  2$:     BIC #40100,UNITSEL      ;CLEAR UNIT USED BITS
002424  013705  000042                MOV @#42,R5             ;ACT 11 END OF PASS HOOKS
002430  001405                        BEQ HERE
002432  000005                        RESET
002434  004715                LOGICAL: JSR PC,(R5)
002436  000240                        NOP
002440  000240                        NOP
002442  000240                        NOP
002444  000207                HERE:   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

002446  005077  176534          T1:     CLR @ RXCS                      ; WRITING RXCS TO 0, ALSO WRITES RXDB = 0
002452  012700  000040                  MOV #40, R0
002456  017701  176524                  MOV @ RXCS, R1
002462  020001                          CMP R0, R1
002464  001401                          BEQ 1$

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

002466  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
        ;
        ;/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#
        ;/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#
        ;/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#

903  002470  104406          1$:     SUBSCOPE
```

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


002472  005000                        CLR R0
002474  017701  176510                MOV @ RXDB, R1
002500  020100                        CMP R1, R0
002502  001401                        BEQ 3$

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

002504  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
;
;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
```

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

   961  002506  104406                    3$:     SUBSCOPE
```

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

002510  013702  001204                MOV KRXVEC, R2
002514  010246                        MOV     R2,-(SP)                ;SAVE INTERRUPT VECTOR FOR
                                                                        ;ERROR REPORT
002516  012722  002556                MOV #4$, (R2)+                  ; RX01 VECTOR ADDRESS
002522  012722  000340                MOV #PR7, (R2)+
002526  012602                        MOV     (SP)+,R2                ;RESTORE INTERRUPT VECTOR FOR
                                                                        ;ERROR REPORT
002530  005046                        CLR -(SP)                       ; PDP PRIORITY <ON>
002532  012746  002540                MOV #2$,-(SP)
002536  000002                        RTI
002540  000240                2$:     NOP
002542  000240                        NOP
002544  012746  000340                MOV #PR7,-(SP)                  ; RESET PDP PRIORITY <7> OFF
002550  012746  002560                MOV #5$,-(SP)
002554  000002                        RTI

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


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

002556  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

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

1010  002560  000004                5$:     SCOPE
1011  002562  000137  004074                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 )

002566  013702  001204          T2:     MOV KRXVEC, R2
002572  010246                          MOV     R2,-(SP)            ;SAVE INTERRUPT VECTOR FOR
                                                                    ;ERROR REPORT
002574  012722  002644                  MOV #1$, (R2)+              ; RX01 VECTOR ADDRESS
002600  012722  000340                  MOV #PR7, (R2)+
002604  012602                          MOV     (SP)+,R2            ;RESTORE INTERRUPT VECTOR FOR
                                                                    ;ERROR REPORT
002606  005046                          CLR -(SP)                   ; PDP PRIORITY (ON)
002610  012746  002616                  MOV #6$,-(SP)
002614  000002                          RTI
002616  052777  000100  176362  6$:     BIS #BIT6, @ RXCS           ; SET RX01 INTERRUPT ENABLE
002624  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
```

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

1070  002626  000240                          NOP
1071  002630  012746  000340                 MOV #PR7,-(SP)              ; RESET PDP PRIORITY (7) OFF
1072  002634  012746  002642                 MOV #7$,-(SP)
1073  002640  000002                          RTI
1074
1075                                  ; (R0) = N/A ; (R1) = N/A ; (R2) = N/A
1076
1077  002642  104000                  7$:     ERROR                       ; NO RX01 INTERRUPT OCCURRED
1078

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

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

1113
1114                                  ; RETURN TO HERE WITH THE PDP PRIORITY = 7 IF AN RX01 INTERRUPT
1115
1116  002644  005737  002660          1$:     TST CATERR                  ;DID MORE THAN ONE INTERRUPT
1117                                                                      ;OCCUR DUE TO BOTH 'DONE' & 'IE'
1118                                                                      ;BITS SET BUT 'RQST INTR' FLOP
1119                                                                      ;NEVER GETTING CLEARED?
```

```
1120  002650  001004                        BNE     DEATH           ;BRANCH IF YES
1121  002652  105237  002660                INCB    CATERR          ;OTHERWISE, SET FLAG TO INDICATE
1122                                                                  ;ONE INTERRUPT OCCURRED & GO ON
1123  002656  000402                        BR      OK2GO
1124  002660  000000              CATERR:   .WORD   0               ;THIS IS THE INTERRUPT INDICATOR.
1125                                                                  ;1ST TIME THRU THIS TEST THE IN-
1126                                                                  ;DICATOR SHOULD ALWAYS BE ZERO;
1127                                                                  ;IF NON-ZERO A MULTIPLE OF
1128                                                                  ;INTERRUPTS HAS OCCURRED DUE TO
1129                                                                  ;THE AFOREMENTIONED CONDITION
1130  002662  000000              DEATH:    HALT                    ;NO SENSE GOING ON - YOU ARE
1131                                                                  ;LOCKED IN THE INTERRUPT STATE!
1132                                                                  ;PRESS THE 'CONT' SWITH (IF
1133                                                                  ;PRESENT) TO GO ON  AT YOUR OWN
1134                                                                  ;RISK
1135  002664  104406              OK2GO:    SUBSCOPE                ;EVERYTHING APPEARS TO BE OK
1136  002666  005037  002660                CLR     CATERR          ;CLEAR INTERRUPT INDICATOR FOR
1137                                                                  ;NEXT POSSIBLE PASS THRU THIS TEST
1138
1139                                ; THE RXCS SHOULD = 140 (INTERRUPT ENABLE+DONE)
1140
1141  002672  012700  000140                MOV #140, R0
1142  002676  020077  176304                CMP R0, @ RXCS
1143  002702  001403                        BEQ 2$
1144  002704  017701  176276                MOV @ RXCS, R1
1145
1146                                ; (R0) = 140 ; (R1) = ACTUAL RXCS ; (R2) = N/A
1147
1148  002710  104000                        ERROR                   ; THE RXCS NOT = 140
1149

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

        ;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

;/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#
;/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#
;/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#/:\#
```

```
1174  002712  104406                  2$:     SUBSCOPE
1175
1176
1177                                  ; TEST 2 - CONT'D
1178
1179                                  ; THE PURPOSE OF THIS TEST IS TO VERIFY THAT BIT 6 OF THE RXCS (INTERRUPT ENABLE)
1180                                  ; CAN BE CLEARED AFTER IT WAS KNOWN TO BE SET
1181
1182  002714  013702  001204                MOV KRXVEC, R2
1183  002720  010246                        MOV     R2,-(SP)                ;SAVE INTERRUPT VECTOR FOR
1184                                                                         ;ERROR REPORT
1185  002722  012722  003012                MOV #4$, (R2)+                  ; RX11 VECTOR ADDRESS
1186  002726  012722  000340                MOV #PR7, (R2)+
1187  002732  012602                        MOV     (SP)+,R2                ;RESTORE INTERRUPT VECTOR FOR
1188                                                                         ;ERROR REPORT
1189  002734  042777  000100  176244        BIC #BIT6, @ RXCS               ; CLEAR THE RX11 INTERRUPT ENABLE BIT
1190
1191                                  ; THE RXCS SHOULD = 40 (DONE)
1192
1193  002742  012700  000040                MOV #40, R0
1194  002746  020077  176234                CMP R0, @ RXCS
1195  002752  001403                        BEQ 3$
1196  002754  017701  176226                MOV @ RXCS, R1
1197
1198                                  ; (R0) = 40 ; (R1) = ACTUAL RXCS ; (R2) = N/A
1199
1200  002760  104000                        ERROR                           ; RXCS NOT = 40
1201  002762  104406                  3$:     SUBSCOPE
1202                                  ; NO RX11 INTERRUPTS SHOULD OCCUR (YET)
1203
1204  002764  005046                        CLR -(SP)                       ; PDP PRIORITY (ON)
1205  002766  012746  002774                MOV #10$,-(SP)
1206  002772  000002                        RTI
1207  002774  000240                  10$:    NOP
1208  002776  000240                        NOP
1209  003000  012746  000340                MOV #PR7,-(SP)                  ; PDP PRIORLITY (OFF) 7
1210  003004  012746  003014                MOV #11$,-(SP)
1211  003010  000002                        RTI
1212
1213                                  ; RETURN TO HERE WITH THE PDP PRIORITY = 7 IF AN UNEXPECTED RX11 INTERRUPT
1214                                  ; WHILE CLEARING THE RX11 INTERRUPT ENABLE BIT 6
1215
1216                                  ; (R0) = N/A ; (R1) = N/A ; (R2) = N/A
1217
1218  003012  104000                  4$:     ERROR                           ; UNEXPECTED RX11 INTERRUPT
1219  003014  104406                  11$:    SUBSCOPE
1220
1221                                  ; AN RX11 INTERRUPT SHOULD OCCUR (NOW)
1222
1223  003016  013702  001204                MOV KRXVEC, R2
1224  003022  010246                        MOV     R2,-(SP)                ;SAVE INTERRUPT VECTOR FOR
1225                                                                         ;ERROR REPORT
1226  003024  012722  003074                MOV #5$, (R2)+                  ; RX11 VECTOR ADDRESS
1227  003030  012722  000340                MOV #PR7, (R2)+
```

H09


```
1228  003034  012602                          MOV     (SP)+,R2                ;RESTORE INTERRUPT VECTOR FOR
1229                                                                          ;ERROR REPORT
1230  003036  005046                          CLR -(SP)                       ; PDP PRIORITY <ON>
1231  003040  012746  003046                  MOV #12$,-(SP)
1232  003044  000002                          RTI
1233  003046  052777  000100  176132  12$:    BIS #BIT6, @ RXCS               ; SET RX11 INTERRUPT ENABLE BIT
1234  003054  000240                          NOP
1235  003056  000240                          NOP
1236  003060  012746  000340                  MOV #PR7,-(SP)
1237  003064  012746  003072                  MOV #13$,-(SP)
1238  003070  000002                          RTI
1239
1240                                  ; (R0) = N/A ; (R1) = N/A ; (R2) = N/A
1241
1242  003072  104000                  13$:    ERROR                           ; NO RX11 INTERRUPT OCCURRED
1243
1244                                  ; RETURN TO HERE WITH THE PDP PRIORITY = 7 IF AN RX01 INTERRUPT
1245
1246  003074  000004                  5$:     SCOPE
1247  003076  042777  000100  176102          BIC #BIT6, @ RXCS               ; CLEAR THE RX11 INTERRUPT ENABLE
1248
1249  003104  000137  004074                  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.

003110  005001          T3:     CLR     R1                      ;INDICATOR TO CPU PRIORITY
                                                                ;ROUTINE TO DROP CPU PRIORITY
                                                                ;TO 1 LESS THAN DEVICE LEVEL
003112  013702  001204          MOV KRXVEC, R2
003116  010246                  MOV     R2,-(SP)                ;SAVE INTERRUPT VECTOR FOR
                                                                ;ERROR REPORT
003120  012722  003232          MOV #1$, (R2)+                  ; RX01 VECTOR ADDRESS
003124  012722  000340          MOV #PR7, (R2)+
003130  012602                  MOV     (SP)+,R2                ;RESTORE INTERRUPT VECTOR FOR
                                                                ;ERROR REPORT
003132  013746  000004          MOV     4,-(SP)                 ;SAVE 'BUSERR' TIMEOUT 'PC'
003136  012737  003154  000004  MOV     #2$,4                   ;SET TIMEOUT VECTOR
003144  012737  000200  177776  MOV     #PR4,PSW                ;SET LEVEL TO 4 IF 'PSW' EXISTS
003152  000404                  BR      3$                      ;GO TO RESET VECTOR 4 & DO TEST
003154  022626          2$:     CMP     (SP)+,(SP)+             ;CORRECT STACK FROM BUS TIMEOUT
003156  012637  000004          MOV     (SP)+,4                 ;RESTORE TIMEOUT VECTOR TO 'BUSERR'
003162  000427                  BR      4$                      ;NO HARDWARE PSW - SKIP THIS TEST
003164  012637  000004  3$:     MOV     (SP)+,4                 ;RESET TIMEOUT VECTOR TO 'BUSERR'
003170  004737  006026          JSR     PC,CPUPRI               ;CALCULATE PRIORITY LEVEL OF CPU
                                                                ;BASED ON CURRENT DEVICE PRIORITY
                                                                ;LEVEL RESIDING IN LOC. 'BRLEV'
003174  010046                  MOV     R0,-(SP)        ;;PUT NEW PS ON STACK
003176  012746  003204          MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
003202  000002                  RTI                     ;; POP NEW PC AND PS
003204                  64$:
003204  052777  000100  175774  BIS #BIT6, @ RXCS                ; SET THE RX01 INTERRUPT ENABLE
003212  000240                  NOP
003214  000240                  NOP
003216  013746  000340          MOV     PR7,-(SP)       ;;PUT NEW PS ON STACK
003222  012746  003230          MOV     #65$,-(SP)      ;;PUT NEW PC ON  STACK
003226  000002                  RTI                     ;; POP NEW PC AND PS
003230                  65$:

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

003230  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
```

```
1304  003232  000004                    1$:     SCOPE
1305  003234  042777  000100  175744            BIC #BIT6, @ RXCS              ; CLEAR THE RX11 INTERRUPT ENABLE
1306  003242  000137  004074            4$:     JMP CEXIT                      ;END OF TEST 3
1307
1308                                            .SBTTL TEST 4 - INTERRUPT TEST PART IV / PRIORITY VERIFICATION PART II
1309
1310                                    ; THE PURPOSE OF THIS TEST IS TO VERIFY THE PRIORITY OF THE RX11 INTERRUPT REQUEST LINE
1311                                    ; THE PROGRAM SETS THE PDP PRIORITY = THE DEVICE LEVEL, (NORMALLY 5 OR THE CONTENTS OF L
1312                                    ;NO RX01 INTERRUPTS SHOULD OCCUR
1313                                    ; IF AN INTERRUPT DOES OCCUR THEN THE PRIORITY LEVEL OF THE RX11 IS [NOT]
1314                                    ; = THE NORMAL DEVICE LEVEL OF 5, OR WHATEVER IS THE VALUE IN LOCATION 'BRLEV:'
1315                                    ; BUT MAYBE SOME VALUE GREATER THAN THE CONTENTS OF LOC. 'BRLEV:'
1316                                    ; NOTE: IF THERE IS NO HARDWARE "PSW" THIS TEST WILL BE SKIPPED.
1317
1318  003246  005001                    T4:     CLR     R1                     ;INDICATOR TO CPU PRIORITY ROUTINE
1319                                                                           ;TO DROP CPU PRIORITY 1 LEVEL
1320                                                                           ;LESS THAN THE DEVICE LEVEL
1321  003250  013702  001204                    MOV KRXVEC, R2
1322  003254  010246                            MOV     R2,-(SP)               ;SAVE INTERRUPT VECTOR FOR
1323                                                                           ;ERROR REPORT
1324  003256  012722  003374                    MOV #1$, (R2)+                 ; RX01 VECTOR ADDRESS
1325  003262  012722  000340                    MOV #PR7, (R2)+
1326  003266  012602                            MOV     (SP)+,R2               ;RESTORE INTERRUPT VECTOR FOR
1327                                                                           ;ERROR REPORT
1328  003270  052701  000200                    BIS     #BIT7,R1               ;SET INDICATOR TO CPU PRIORITY
1329                                                                           ;ROUTINE TO SET CPU PRIORITY LEVEL
1330                                                                           ;TO THE SAME LEVEL AS THE DEVICE
1331  003274  013746  000004                    MOV     4,-(SP)                ;SAVE 'BUSERR' TIMEOUT 'PC'
1332  003300  012737  003316  000004            MOV     #3$,4                  ;SET TIMEOUT VECTOR
1333  003306  012737  000240  177776            MOV     #PR5,PSW               ;SET LEVEL TO 5 IF 'PSW' EXISTS
1334  003314  000404                            BR      4$                     ;GO ON TO RESET VECTOR & DO TEST
1335  003316  022626                    3$:     CMP     (SP)+,(SP)+            ;CORRECT STACK FROM BUS TIMEOUT
1336  003320  012637  000004                    MOV     (SP)+,4                ;RESTORE TIMEOUT VECTOR TO 'BUSERR'
1337  003324  000430                            BR      5$                     ;NO HARDWARE PSW - SKIP THIS TEST
1338  003326  012637  000004            4$:     MOV     (SP)+,4                ;RESET TIMEOUT VECTOR TO BUSERR
1339  003332  004737  006026                    JSR     PC,CPUPRI              ;CALCULATE CPU PRIORITY LEVEL TO
1340                                                                           ;BE THE SAME AS THE DEVICE LEVEL
1341                                                                           ;I.E. - SAME AS CONTENTS OF LOC.
1342                                                                           ;'BRLEV
1343  003336  010046                            MOV     R0 -(SP)       ;;PUT NEW PS ON STACK
1344  003340  012746  003346                    MOV     #6$,-(SP)      ;;PUT NEW PC ON  STACK
1345  003344  000002                            RTI                    ;; POP NEW PC AND PS
1346  003346                            64$:
1347  003346  052777  000100  175632            BIS     #BIT6,@RXCS            ;SET RX01 INTERRUPT ENABLE
1348  003354  000240                            NOP
1349  003356  000240                            NOP
1350  003360  013746  000340                    MOV     PR7,-(SP)      ;;PUT NEW PS ON STACK
1351  003364  012746  003372                    MOV     #65$,-(SP)     ;;PUT NEW PC ON  STACK
1352  003370  000002                            RTI                    ;; POP NEW PC AND PS
1353  003372                            65$:
1354  003372  000401                            BR 2$
1355
1356                                    ; RETURN TO HERE WITH THE PDP PRIORITY = 7 IF AN RX01 INTERRUPT
1357
```

```
1358                                    ; (R0) = N/A ; (R1) = N/A ; (R2) = N/A
1359
1360  003374  104000                    1$:     ERROR                           ;PRIORITY LEVEL NOT = TO CONTENTS
1361                                                                            ;OF LOCATION 'BRLEV:' (NORMALLY 5)
1362                                                                            ;BUT MAYBE SOME VALUE GREATER THAN
1363                                                                            ;THE CONTENTS SPECIFIED BY LOC.
1364                                                                            ;'BRLEV:'
1365  003376  000004                    2$:     SCOPE
1366  003400  042777  000100  175600            BIC #BIT6, @ RXCS               ; CLEAR THE RX01 INTERRUPT ENABLE
1367  003406  000137  004074            5$:     JMP CEXIT                       ;END OF TEST 4
```

```
1368                                            .SBTTL TEST 5 - INIT [PROGRAMMED] / RST
1369
1370                                    ; THE PURPOSE OF THIS TEST IS TO VERIFY THAT SETTING THE RXCS BIT 14
1371                                    ; CAUSES AN RX01 PROGRAMMED SUBSYSTEM INITIALIZE
1372
1373                                    ; THE RXCS SHOULD = 40 (DONE)
1374
1375                                    ; THE RXDB SHOULD = 4, OR 104, OR 204, OR 304
1376
1377  003412  052777  040000  175566  T5:      BIS #BIT14, @ RXCS              ; RX01 PROGRAMMED INITIALIZE
1378  003420  004737  006412          1$:      JSR PC, SDN                     ; WAIT FOR THE DONE BIT
1379  003424  000775                           BR 1$
1380  003426  012700  000040                   MOV #40, R0                     ; PROGRAM EXPECTS RXCS = 40 (DONE)
1381  003432  017701  175550                   MOV @ RXCS, R1                  ; ACTUAL RXCS
1382  003436  020100                           CMP R1, R0
1383  003440  001401                           BEQ 2$
1384
1385                                    ; (R0) = 40 ; (R1) = ACTUAL RXCS ; (R2) = N/A
1386
1387  003442  104000                           ERROR                           ; RXCS NOT = 40
1388

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

                ;IF THE FAULT CANNOT BE FOUND ON THE UNIBUS INTERFACE MODULE
                ;AND/OR THE FAULT IS NOT INHERENT TO THE UNIBJS 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
        ;
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

1416  003444  104406          2$:      SUBSCOPE
1417
1418                                    ;////////////////////////////////////////////////////////////////////////////////
1419                                    ;
1420                                    ;       7       6       -       -       3       2       1       0               /
1421                                    ;
```

```
1422                                    ;       SEL                                     WRITE   INIT    PAR             /
1423                                    ;       DRIVE   00                              PROTECT [DONE]          CRC     /
1424                                    ;       RDY                                                     (N/A)           /
1425                                    ;
1426                                    ;////////////////////////////////////////////////////////////////////////////////
1427
1428                                    ; THE RXDB SHOULD = 4, 104, IF TESTING UNIT 1
1429                                    ; OR 204, OR 304 IF TESTING UNIT 0 (SEL. DRIVE RDY. BIT SET)
1430
1431  003446  012700  000204                        MOV #204,R0
1432  003452  017702  175532                        MOV @RXDB,R2
1433  003456  010201                                MOV R2,R1
1434  003460  042701  000100                        BIC #BIT6,R1                    ;CLEAR DELETED DATA BIT
1435  003464  105737  012446                        TSTB UNITSEL                    ;WAS UNIT 0 SELECTED
1436  003470  100404                                BMI 3$
1437  003472  042701  000200                        BIC #BIT7,R1                    ;UNIT 0 WAS NOT SELECTED
1438  003476  042700  000200                        BIC #BIT7,R0                    ;CLEAR UNIT 0 RDY BITS
1439  003502  020100                        3$:     CMP R1, R0
1440  003504  001401                                BEQ 4$
1441
1442                                    ; (R0) = 4, OR 204
1443                                    ; (R1) = ACTUAL RXDB MINUS DELETED DATA BIT#6
1444                                    ; (R2) = ACTUAL RXDB
1445
1446  003506  104000                                ERROR                           ; RXDB NOT = 4, OR 104, OR 204, OR 304
1447

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

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

;M7846 (UNIBUS INTERFACE)

;       SIGNAL NAME             REASON                  POSSIBLE CHIPS

;RX INIT                 STUCK HIGH                      E11,E36
;IN                      STUCK HIGH                      E11
;BUS DO5                 STUCK LOW                       E1
;RX DATA                 STUCK LOW                       E14
; ----------             SACK FLOP CLOCK LOCKED LOW      E9
;B SER DATA              STUCK LOW                       E9
;DONE                    STUCK HIGH                      E22
; ----------             LOAD PULSE STUCK LOW            E3

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

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

  1478  003510  104406                  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.

003512  012777  000013  175466          MOV #13, @ RXCS                 ; RST COMMAND
003520  032777  000040  175460          BIT #BIT5, @ RXCS               ; TEST DONE BIT (SHOULD = 0)
003526  001404                          BEQ 5$
003530  017701  175452                  MOV @ RXCS, R1
003534  005000                          CLR R0

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

003536  104000                          ERROR                           ; DONE BIT NOT = 0
003540  104406                  5$:     SUBSCOPE
003542  004737  006412          9$:     JSR PC, SDN                     ; WAIT FOR DONE FLAG
003546  000775                          BR 9$
003550  012700  000040                  MOV #40, R0                     ; PROGRAM EXPECTS RXCS = 40 (DONE)
003554  017701  175426                  MOV @ RXCS, R1                  ; ACTUAL RXCS
003560  020100                          CMP R1, R0
003562  001401                          BEQ 10$

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

003564  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
        ;
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

1531    003566  104406          10$:    SUBSCOPE
1532
```

```
1533                                    ; THE RXDB SHOULD = 200 (IF DRIVE 0 IS READY), OR 0 IF UNIT 0 IS NOT SELECTED
1534                                    ; MAYBE 300 (IF DRIVE 0 IS READY AND SECTOR 1 WAS WRITTEN WITH DELETED DATA)
1535
1536  003570  017702  175414                  MOV @ RXDB, R2                  ; ACTUAL RXDB
1537  003574  010201                          MOV R2, R1
1538  003576  042701  000100                  BIC #BIT6, R1                   ; CLEAR N/A DELETED DATA BIT
1539  003602  012700  000200                  MOV #200,R0                     ;EXPECT UNIT 0 RDY SET
1540  003606  105737  012446                  TSTB UNITSEL
1541  003612  100403                          BMI 11$
1542  003614  042701  000200                  BIC #BIT7,R1                    ; UNIT 0 NOT SELECTED
1543  003620  005000                          CLR R0                          ; DISREGUARD RDY BITS
1544  003622  020100                  11$:    CMP R1, R0
1545  003624  001401                          BEQ 12$
1546
1547                                    ; (R0) = 0 OR 200
1548                                    ; (R1) = ACTUAL RXDB MINUS DELETED DATA BIT#6
1549                                    ; (R2) = ACTUAL RXDB
1550
1551  003626  104000                          ERROR                           ; RXDB NOT = 200, OR NOT = 0
1552  003630  000004                  12$:    SCOPE
1553  003632  000137  004074                  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

003636  012737  000006  012046  T6:     MOV #6, PAT                     ; COUNT PATTERN
003644  004737  003654                  JSR PC, T6FILL
003650  000137  004074                  JMP CEXIT                       ;END OF TEST 6

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

003654  005002                  T6FILL: CLR R2
003656  012777  000001  175322          MOV #1, @ RXCS                  ;FILL BUFFER FUNCTION [0 + GO BIT]
003664  004737  012002                  JSR PC, GETPATTERN
003670  012704  016300                  MOV #BUFADR, R4
003674  004737  003712          1$:     JSR PC, FBEB                    ; SUBROUTINE TO FILL/EMPTY BUFFER
003700  000207                          RTS PC                          ; EXIT SUBROUTINE T6FILL

003702  112477  175302                  MOVB (R4)+, @ RXDB              ; FILL THE SECTOR BUFFER
003706  005202                          INC R2                          ;INC R2 FOR BYTE COUNT
003710  000771                          BR 1$

                                ; SUBROUTINE TO FILL AND EMPTY THE SECTOR BUFFER

003712  004737  006376          FBEB:   JSR PC, STR                     ; TEST FOR TRANSFER REQUEST
003716  000403                          BR 1$
003720  062716  000002                  ADD #2, @ SP                    ; ADJUST FOR EXIT FROM THIS SUBROUTINE
003724  000207                          RTS PC                          ; EXIT TO SERVICE TRANSFER REQUEST

003726  004737  006412          1$:     JSR PC, SDN
003732  000767                          BR FBEB                         ; NOT TRANSFER REQUEST OR DONE
003734  005777  175246                  TST @ RXCS                      ;TEST FOR ERROR FLAG
003740  100003                          BPL 3$
003742  017701  175242                  MOV @RXDB,R1

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

003746  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
        ;
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*

1618  003750  104406                  3$:     SUBSCOPE
1619
1620                                  ; 128 BYTES SHOULD HAVE BEEN TRANSFERRED TO OR FROM THE SECTOR BUFFER
1621
1622  003752  022702  000200                  CMP #200,R2
1623  003756  001401                          BEQ 2$
1624
1625                                  ; (R0) = N/A ; (R1) = N/A ; (R2) = ACTUAL # OF TRANSFERS
1626
1627  003760  104000                          ERROR                           ; INCORRECT TRANSFER LENGTH
1628

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

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

1652  003762  000004                  2$:     SCOPE
1653
1654  003764  000207                          RTS PC                          ; EXIT SUBROUTINE FBEB
```

```
                                        .SBTTL TEST 10 - EMPTY BUFFER XFER LENGTH AND CONTENT VERIFICATION PART II

                                ; THE PURPOSE OF THIS TEST IS TO VERIFY THAT THE PREVIOUS EMPTY BUFFER TEST
                                ; DID NOT EMPTY AND DESTROY THE CONTENTS OF THE SECTOR BUFFER
                                ;NOTE: TEST 6 MUST BE RUN BEFORE THIS TEST.

003766  000240                  T10:    NOP                             ; NOP FOR T10 " FAST " DEFINITION

                                        .SBTTL TEST 7 - EMPTY BUFFER XFER LENGTH AND CONTENT VERIFICATION PART I

                                ; THE PURPOSE OF THIS TEST IS TO VERIFY THE TRANSFER LENGTH OF THE FOUNCTION
                                ; "EMPTY BUFFER" AND ALSO TO VERIFY THE CONTENTS OF THE SECTOR BUFFER
                                ; NOTE TEST 6 MUST BE RUN BEFORE THIS TEST

003770  004737  004000          T7:     JSR PC, T7EMPTY
003774  000137  004074                  JMP CEXIT                       ;END OF TEST 7 OR 10
004000  005002                  T7EMPTY: CLR R2
004002  012777  000003  175176          MOV #3, @ RXCS                  ; ISSUE THE COMAND TO EMPTY BUFFER
004010  012704  016300                  MOV #BUFADR, R4
004014  004737  003712          2$:     JSR PC, FBEB                    ; SUBROUTINE TO EMPTY BUFFER
004020  000207                          RTS PC                          ; EXIT SUBROUTINE T7EMPTY

                                ; EMPTY BUFFER AND VERIFY THE CONTENTS

004022  112400                          MOVB (R4)+, R0                  ; EXPECTED CONTENTS OF SECTUR BUFFER
004024  117701  175160                  MOVB @ RXDB, R1                 ; ACTUAL CONTENTS OF SECTOR BUFFER
004030  005202                          INC R2                          ; ACTUAL # TRANSFERS TO THIS ERROR
004032  020001                          CMP R0, R1
004034  001401                          BEQ 1$

                                ; IF AN ERROR OCCURS:

                                ; (R0) - EXPECTED CONTENTS OF SECTOR BUFFER
                                ; (R1) - ACTUAL CONTENTS FROM SECTOR BUFFER
                                ; (R2) - TOTAL # OF ACTUAL TRANSFERS

004036  104000                          ERROR                           ; DATA " TO " SB NOT = DATA " FROM "

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

        ;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:
;
;M7946 (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                      ----------
                        ;
                        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
                        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
                        ;/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*/:\*
  004040 000004                         1$:     SCOPE
  004042 000764                                 BR 2$

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

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

  004044 012737 000001 012046   T12:    MOV #1, PAT
  004052 000402                         BR .+6

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

  004054 005037 012046          T11:    CLR PAT
  004060 004737 003654                  JSR PC, T6FILL
  004064 004737 004000                  JSR PC, T7EMPTY
  004070 000137 004074                  JMP CEXIT                       ;END OF TEST 11 OR 12
```

```
1757  004074  012737  000103  002344  CEXIT:  MOV #'C,MX                      ;EOP INDICATOR FOR END OF CONTROL TESTS
1758  004102  000137  002032                  JMP MORETESTS
1759
1760  004106  012737  000104  002344  DEXIT:  MOV #'D,MX                      ;EOP INDICATOR FOR END OF DRIVE TESTS
1761  004114  000137  002032                  JMP MORETESTS
1762
1763                                          .SBTTL TEST 13 DRIVE READY VERIFICATION
1764
1765                                  ;THIS TEST VERIFIES THAT DRIVE READY WILL SET FOR ALL SELECTED DRIVES.
1766
1767                                  ;THIS SECTION OF TEST FOR UNIT 0 DRY NO A READ STATUS A FUNCTION
1768
1769  004120  105737  012446          T13:    TSTB UNITSEL                    ;WAS UNIT 0 SELECTED
1770  004124  100020                          BPL 2$                          ;NO, GO TEST UNIT 1
1771  004126  012777  000013  175052          MOV #13,@RXCS                   ;READ STATUS A UNIT 0
1772  004134  004737  006412          1$:     JSR PC,$DN                      ;WAIT FOR DONE FLAG
1773  004140  000775                          BR 1$
1774  004142  012700  000200                  MOV #200,R0                     ;EXPECT DRIVE READY TO BE SET
1775  004146  017702  175036                  MOV @RXDB,R2                    ;ACTUAL RXDB
1776  004152  010201                          MOV R2,R1
1777  004154  042701  000100                  BIC #BIT6,R1                    ;CLEAR DD BIT IF SET
1778  004160  020001                          CMP R0,R1                       ;WAS " DRY " SET AND INITDONE CLEARED
1779  004162  001401                          BEQ 2$
1780
1781                                  ;R0 = 200 ; R1 = RXDB MINUS DD BIT ;R2 = ACTUAL RXDB
1782
1783  004164  104000                          ERROR
1784  004166  104406          2$:     SUBSCOPE
1785
1786                                  ;TEST FOR UNIT 1 DRIVE READY ON A READ STATUS A FUNCTION
1787
1788  004170  005737  012446                  TST UNITSEL                     ;WAS UNIT 1 SELECTED
1789  004174  100020                          BPL 3$                          ;NO, GO TO END OF TEST
1790  004176  012777  000033  175002          MOV #33,@RXCS                   ;READ STATUS A FOR DRIVE 1
1791  004204  004737  006412          4$:     JSR PC,$DN                      ;WAIT FOR DONE FLAG
1792  004210  000775                          BR 4$
1793  004212  012700  000200                  MOV #200,R0                     ;EXPECT " DRY " TO BE SET
1794  004216  017702  174766                  MOV @RXDB,R2
1795  004222  010201                          MOV R2,R1
1796  004224  042701  000100                  BIC #BIT6,R1                    ;CLEAR DD BIT IF ANY
1797  004230  020100                          CMP R1,R0                       ;IS DRY SET AND INITDONE CLEARED
1798  004232  001401                          BEQ 3$
1799
1800                                  ;R0 = 200 ; R1 = RXDB MINUS DD BIT ; R2 = ACTUAL RXDB
1801
1802  004234  104000                          ERROR
1803
```

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

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

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

1834  004236  000004                 3$:     SCOPE
1835  004240  000137  004106                 JMP DEXIT                       ;END OF TEST 13
```

```
1836                                        .SBTTL TEST 14 - ERROR FLAG AND B-CODE VERIFICATION PART I
1837
1838                                ;THE PURPOSE OF THIS TEST IS TO VERIFY THAT TRYING TO READ A NON-EXISTANT
1839                                ;SECTOR WILL CAUSE AN ERROR, AND THE CORRECT ERROR CODE WILL BE PUT
1840                                ;INTO THE RXDB WHEN THE B STATUS IS READ.
1841
1842                                ;THIS SECTION INSURES THAT ONLY 2 TR FLAGS WERE REQUIRED TO TAKE THE
1843                                ;DISKETTE ADDRESS, AND THAT AN ERROR DOES EXIST.
1844
1845  004244  005002                T14:    CLR R2
1846  004246  005000                        CLR R0
1847  004250  105737  012446               TSTB UNITSEL                    ;IS UNIT 0 SELECTED
1848  004254  100004                        BPL 10$
1849  004256  012777  000007  174722       MOV #7,@RXCS                    ;SET READ SECTOR FUNCTION AND GO
1850  004264  000403                        BR 11$
1851  004266  012777  000027  174712  10$:  MOV #27,@RXCS                   ;SEND READ FUNCTION TO UNIT 1
1852  004274  004737  004734        11$:  JSR PC,ILLADR                   ;SEND THE BAD SECTOR ADDRESS
1853  004300  000406                        BR 1$                           ;PREMATURE ERROR CONDITION
1854  004302  012700  100040               MOV #100040,R0                  ;EXPECT ERROR AND DONE BITS
1855  004306  017701  174674               MOV @RXCS,R1                    ;SAVE THE RXCS
1856  004312  020001                        CMP R0,R1
1857  004314  001401                        BEQ 2$
1858
1859                                ;R0 = 100040 ; R1 = ACTUAL RXCS ; R2 = # OF TR FLAGS
1860
1861  004316  104000                1$:     ERROR
1862

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

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

1887  004320  104406                2$:     SUBSCOPE
1888
1889                                ;T14 CONT. - THIS SECTION TESTS THAT NO PARITY OR CRC ERROR OCCURRED
```

```
1890                                    ;ON PREVIOUS FUNCTION.
1891
1892  004322  005000                            CLR R0                          ;STATUS A SHOULD BE CLEAR
1893  004324  017702  174660                    MOV @RXDB,R2
1894  004330  010201                            MOV R2,R1
1895  004332  042701  000100                    BIC #BIT6,R1
1896  004336  020001                            CMP R0,R1
1897  004340  001401                            BEQ 3$
1898
1899                                    ;R0 = 0 ; R1 = RXDB MINUS DO BIT ; R2 = ACTUAL RXDB
1900
1901  004342  104000                            ERROR
1902  004344  104406                    3$:     SUBSCOPE
1903
1904                                    ;T14 CONT. - THIS SECTION TESTS THAT NO ERROR CONDITIONS EXIST ON A
1905                                    ;READ STATUS B FUNCTION.
1906
1907  004346  012777  000017  174632            MOV #17,@RXCS                   ;SET READ STATUS B FUNCTION
1908  004354  004737  006412            4$:     JSR PC,SDN                      ;WAIT FOR DONE FLAG
1909  004360  000775                            BR 4$
1910  004362  005777  174620                    TST @RXCS                       ;IS THE ERROR BIT SET
1911  004366  100007                            BPL 5$                          ;NO, GO ON TO NEXT SECTION
1912  004370  012700  000040                    MOV #40,R0                      ;YES,SET UP FOR ERROR
1913  004374  017701  174606                    MOV @RXCS,R1
1914  004400  017702  174604                    MOV @RXDB,R2
1915
1916                                    ;R0 = 40 ; R1 = RXCS ; R2 = RXDB
1917
1918  004404  104000                            ERROR                           ;RXCS NOT = 40
1919  004406  104406                    5$:     SUBSCOPE
1920
1921                                    ;T14 CONT. - THIS SECTION TESTS THE B-CODE FOR " CAN'T FIND SECTOR " #70
1922
1923  004410  012700  000070                    MOV #70,R0
1924  004414  017701  174570                    MOV @RXDB,R1
1925  004420  005002                            CLR R2
1926  004422  020001                            CMP R0,R1                       ;IS THE B-CODE = 70
1927  004424  001401                            BEQ 6$
1928
1929                                    ;R0 = 70 ; R1 = ACTUAL B-CODE ; R2 = N/A
1930
1931  004426  104000                            ERROR                           ;RXDB NOT = 70
1932

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

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

1968  004430  000004                  6$:     SCOPE
1969  004432  000137  004106                  JMP DEXIT                       ;END OF TEST 14
1970
```

```
                                        .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

004436  005000          T15:    CLR R0
004440  005002                  CLR R2
004442  105737  012446          TSTB UNITSEL                    ;WAS UNIT 0 SELECTED
004446  100004                  BPL 10$
004450  012777  000015  174530  MOV #15,@RXCS                   ;SET WRITE DELETED DATA FUNCTION
004456  000403                  BR 11$
004460  012777  000035  174520  10$:    MOV #35,@RXCS          ;SEND WTR DD FUNCTION TO UNIT 1
004466  004737  004734  11$:    JSR PC,ILLADR                   ;SEND THE ILLEGAL SECTOR ADDRESS
004472  000406                  BR 1$                           ;PREMATURE ERROR CONDITION
004474  012700  100040          MOV #100040,R0                  ;EXPECT ERROR AND DONE FLAGS
004500  017701  174502          MOV @RXCS,R1
004504  020001                  CMP R0,R1
004506  001401                  BEQ 2$

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

004510  104000          1$:     ERROR                           ;RXCS NOT = 100040
004512  104406          2$:     SUBSCOPE

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

004514  005002                  CLR R2
004516  012700  000100          MOV #100,R0                     ;EXPECT DELETED DATA BIT TO BE SET
004522  017701  174462          MOV @RXDB,R1
004526  020001                  CMP R0,R1
004530  001401                  BEQ 3$

                                ; R0 = 100 ; R1 = ACTUAL RXDB ; R2 = N/A
004532  104000                  ERROR                           ;DELETED DATA NOT SET OR OTHER ERRORS
004534  104406          3$:     SUBSCOPE

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

004536  012777  000017  174442  MOV #17,@RXCS                   ;SET READ STATUS B FUNCTION
004544  004737  006412  4$:     JSR PC,$DN                      ;WAIT FOR DONE FLAG
004550  000775                  BR 4$
004552  012700  000070          MOV #70,R0
004556  017701  174426          MOV @RXDB,R1
004562  020001                  CMP R0,R1
004564  001401                  BEQ 5$

                                ; R0 = 70 ; R1 = ACTUAL B-CODE ; R2 = N/A
004566  104000                  ERROR                           ;RXDB NOT = 70
004570  000004          5$:     SCOPE
004572  000137  004106          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

004576  005000          T16:    CLR R0
004600  005002                  CLR R2
004602  105737  012446          TSTB UNITSEL                    ;WAS UNIT 0 SELECTED
004606  100004                  BPL 10$
004610  012777  000005  174370  MOV #5,@RXCS                    ;SET THE WRITE FUNCTION
004616  000403                  BR 11$
004620  012777  000025  174360 10$:  MOV #25,@RXCS              ;SEND WRITE FUNCTION TO UNIT 1
004626  004737  004734         11$:  JSR PC,ILLADR              ;SEND THE ILLEGAL ADDRESS
004632  000406                  BR 1$                           ;PREMATURE ERROR CONDITION
004634  012700  100040          MOV #100040,R0                  ;EXPECT ERROR AND DONE BITS SET
004640  017701  174342          MOV @RXCS,R1
004644  020001                  CMP R0,R1
004646  001401                  BEQ 2$

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

004650  104000          1$:     ERROR
004652  104406          2$:     SUBSCOPE

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

004654  005002                  CLR R2
004656  005000                  CLR R0                          ;NO BITS SHOULD BE SET IN THE RXDB
004660  017701  174324          MOV @RXDB,R1
004664  020001                  CMP R0,R1
004666  001401                  BEQ 3$

                                ; R0 = 0 ; R1 = ACTUAL RXDB ; R2 = N/A
004670  104000                  ERROR                           ;SOME BIT IS SET IN THE RXDB
004672  104406          3$:     SUBSCOPE

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

004674  012777  000017  174304  MOV #17,@RXCS                   ;SET READ STATUS 8 FUNCTION
004702  004737  006412  4$:     JSR PC,SDN                      ;WAIT FOR DONE FLAG
004706  000775                  BR 4$
004710  012700  000070          MOV #70,R0
004714  017701  174270          MOV @RXDB,R1
004720  020001                  CMP R0,R1                       ;IS B-CODE = 70
004722  001401                  BEQ 5$                          ;YES, CONTINUE

                                ; R0 = 70 ; R1 = ACTUAL RXDB ; R2 = N/A
004724  104000                  ERROR
004726  000004          5$:     SCOPE
004730  000137  004106          JMP DEXIT                       ;END OF TEST 16
```

```
2076
2077                                    ;GENERATE AN ILLEGAL SECTOR ADDRESS
2078
2079  004734  004737  006376  ILLADR:   JSR PC,STR          ;LOOK FOR A TR FLAG
2080  004740  000402                    BR 2$               ;NO TR FLAG, IS DONE SET
2081  004742  005202                    INC R2              ;TR FLAG COUNTER
2082  004744  000404                    BR 3$
2083  004746  004737  006412  2$:       JSR PC,SDN          ;LOOK FOR DONE FLAG
2084  004752  000770                    BR ILLADR           ;NOT DONE RECHECK TR FLAG
2085  004754  000430                    BR 1$               ;DONE IS SET TOO EARLY GO TO ERROR
2086  004756  005077  174226  3$:       CLR @RXDB           ;0 SECTOR ADDRESS (ILLEGAL)
2087  004762  004737  006376  7$:       JSR PC,STR          ;LOOK FOR SECOND TR FLAG
2088  004766  000402                    BR 5$               ;NOT TR, IS IT DONE
2089  004770  005202                    INC R2
2090  004772  000404                    BR 6$               ;TR FLAG SEND TRACK ADDRESS
2091  004774  004737  006412  5$:       JSR PC,SDN          ;LOOK FOR DONE FLAG
2092  005000  000770                    BR 7$               ;NOT DONE, RECHECK TR FLAG
2093  005002  000415                    BR 1$               ;DONE TOO SOON GO TO ERROR
2094  005004  005077  174200  6$:       CLR @RXDB           ;SEND TRACK ADDRESS OF 0
2095  005010  004737  006376  11$:      JSR PC,STR          ;ARE THERE ANY MORE TR FLAGS
2096  005014  000402                    BR 10$              ;NO, LOOK FOR DONE
2097  005016  005202                    INC R2              ;YES
2098  005020  000406                    BR 1$               ;TOO MANY TR FLAGS OR MICROCONTROLLER
2099                                                        ;DID NOT DETECT THE ERROR
2100  005022  004737  006412  10$:      JSR PC,SDN          ;LOOK FOR DONE FLAG
2101  005026  000770                    BR 11$              ;NOT DONE RETEST TR FLAG
2102  005030  062716  000002            ADD #2,@SP
2103  005034  000207            4$:     RTS PC
2104  005036  017701  174144  1$:       MOV @RXCS,R1
2105  005042  000774                    BR 4$
```

```
2106
2107                                        .SBTTL TEST 17 - ILLEGAL TRACK ERROR VERIFICATION
2108
2109                                ;THIS TEST VER'FIES THAT IF A TRACK ADDRESS LARGER THAN 114 (OCTAL) IS
2110                                ;ACCESSED, AN ERROR CONDITION WILL EXIST, AND A B-CODE WILL = 40
2111
2112  005044  005002          T17:    CLR R2
2113  005046  005000                  CLR R0
2114  005050  012777  000007  174130  MOV #7,@RXCS                    ;SET READ FUNCTION
2115  005056  004737  006376    3$:    JSR PC,STR                      ;LOOK FOR TR FLAG
2116  005062  000401                  BR 1$                           ;NO TR FLAG CHECK DONE
2117  005064  000410                  BR 2$
2118  005066  004737  006412    1$:    JSR PC,SDN
2119  005072  000771                  BR 3$
2120  005074  017701  174106          MOV @RXCS,R1                    ;DONE OCCURRED TOO SOON SET UP FOR ERROR
2121  005100  017702  174104          MOV @RXDB,R2
2122  005104  000433                  BR 4$
2123  005106  012777  000001  174074  2$:    MOV #1,@RXDB              ;SEND LEGAL SECTOR ADDRESS
2124  005114  004737  006376    5$:    JSR PC,STR                      ;LOOK FOR TR FLAG
2125  005120  000401                  BR 6$
2126  005122  000410                  BR 7$
2127  005124  004737  006412    6$:    JSR PC,SDN
2128  005130  000771                  BR 5$
2129  005132  017701  174050          MOV @RXCS,R1                    ;DONE SET TOO EARLY
2130  005136  017702  174046          MOV @RXDB,R2
2131  005142  000414                  BR 4$
2132  005144  012777  000115  174036  7$:    MOV #115,@RXDB            ;SEND ILLEGAL TRACK ADDRESS
2133  005152  004737  006412    10$:   JSR PC,SDN                      ;WAIT FOR DONE ON THE ERROR
2134  005156  000775                  BR 10$
2135  005160  012700  100040          MOV #100040,R0                  ;EXPECT ERROR AND DONE SET
2136  005164  017701  174016          MOV @RXCS,R1
2137  005170  020001                  CMP R0,R1
2138  005172  001401                  BEQ 11$
2139
2140                                ;TWO ERROR CONDITIONS TO REPORT
2141                                ;IF R0 = 0 THEN R1 = RXCS ;R2 = RXDB ON A DONE TOO SOON ERROR
2142                                ;IF R0 = 100040 THEN R1 = ACTUAL RXCS ; R2 = N/A
2143
2144  005174  104000    4$:    ERROR                                   ;DONE SET TOO SOON OR NO ERROR OCCURRED
2145

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

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

005176  104406                  11$:    SUBSCOPE

                                ;T17 CONT. - TEST THAT THERE WERE NO OTHER ERRORS THAN THE ILLEGAL TRACK ERROR EXPECTED
                                ;AND THAT THE DELETED DATA BIT WAS CLEARED BY TEST 16.

005200  005000                          CLR R0                          ;DD BIT CLEARED IN TEST 16 SO RXDB = 0
005202  005002                          CLR R2
005204  017701  174000                  MOV @RXDB,R1
005210  020001                          CMP R0,R1
005212  001401                          BEQ 12$

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

005214  104000                          ERROR
005216  104406                  12$:    SUBSCOPE

                                ;T17 CONT. - VERIFIES THAT READ STATUS B HAS NO ERRORS

005220  012777  000017  173760          MOV #17,@RXCS                   ;SET READ STATUS B FUNCTION
005226  004737  006412          13$:    JSR PC,$DN                      ;WAIT FOR DONE FLAG
005232  000775                          BR 13$
005234  005777  173746                  TST @RXCS                       ;WAS THERE AN ERROR ON THIS FUNCTION
005240  100007                          BPL 14$                         ;NO
005242  012700  000040                  MOV #40,R0
005246  017701  173734                  MOV @RXCS,R1
005252  017702  173732                  MOV @RXDB,R2

                                ; R0 = 40 (DONE) ; R1 = ACTUAL RXCS ; R2 = ACTUAL RXDB

005256  104000                          ERROR                           ;THERE WAS AN ERROR NO READ STATUS B
005260  104406                  14$:    SUBSCOPE
005262  005002                          CLR R2
005264  012700  000040                  MOV #40,R0                      ;B-CODE FOR ILLEGAL TRACK ADDRESS
005270  017701  173714                  MOV @RXDB,R1
005274  020001                          CMP R0,R1
005276  001401                          BEQ 15$                         ;B-CODE IS CORRECT

                                ; R0 = 40 ; R1 = ACTUAL B-CODE ; R2 = N/A

005300  104000                          ERROR
005302  000004                  15$:    SCOPE
005304  000137  004106                  JMP DEXIT                       ;END OF TEST 17
```

```
                                      .SBTTL TEST 20 - SEEK VERIFICATION VIA READ FUNCTION

                              ; THE PURPOSE OF THIS TEST IS TO DO A READ FUNCTION  ON ALL
                              ;SECTORS OF VARIOUS TRACKS ON THE DISKETTE. THIS WILL TEST FOR SEEK ERRORS
                              ;FOR THOSE TRACK POSITIONS. UNLESS OTHERWISE SELECTED (IN OD/ID) THE TRACKS
                              ;ACCESSED ARE 0 (OD), 52, 53 (BOTH SIDES OF THE WRITE CURRENT CHANGE), AND 114 (ID).


005310                        T20:


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

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

005310  013702  001204                MOV KRXVEC,R2                           ;SET UP INTERRUPT ADDRESSES
005314  012722  011256                MOV #INTSERV,(R2)+
005320  012722  000340                MOV #PR7,(R2)+
005324  004737  006564                JSR PC,RDONLY                           ;DO THE READ FUNCTION
005330  000137  004106                JMP DEXIT                               ;END OF TEST 20


                                      .SBTTL TEST 21 - WRITE TEST

                              ;THE PURPOSE OF THIS TEST IS TO WRITE ALL ONES ON SECTOR 1 TRACK 1,
                              ;AND VERIFY THAT THE DATA IN THE SECTOR BUFFER IS NOT MODIFIED.

005334  012737  000001  012650  T21:  MOV #1,TARGET
005342  012737  000001  013140        MOV #1,TSECTOR
005350  004737  012356                JSR PC,GETUNIT
005354  012737  000001  012046        MOV #1,PAT
005362  004737  012002                JSR PC,GETPATTERN
005366  004737  010532                JSR PC,ADJSUM                           ;SET CHECK SUM VALUES
005372  005002                        CLR R2
005374  012777  000001  173604        MOV #1,@RXCS                            ;SET FILL BUFFER FUNCTION
005402  004737  003712          1$:   JSR PC,FBEB
005406  000404                        BR 2$
```

F11


```
  2265  005410  112077  173574                    MOVB (R0)+,@RXDB
  2266  005414  005202                            INC R2
  2267  005416  000771                            BR 1$
  2268  005420  012737  000005  007472  2$:       MOV #5,FUNCTION                 ;SET WRITE FUNCTION
  2269  005426  004737  005436                    JSR PC,T21X                     ;GO ISSUE THE COMMAND
  2270  005432  000137  004106                    JMP DEXIT                       ;END OF TEST 21
```

```
2271  005436  004737  007424          T21X:   JSR PC,COMMWORD
2272  005442  004737  006412          3$:     JSR PC,SDN
2273  005446  000775                          BR 3$
2274  005450  005777  173532                  TST @RXCS                       ;IS THE ERROR BIT SET
2275  005454  100013                          BPL 4$                          ;NO ERRORS
2276  005456  017701  173526                  MOV @RXDB,R1                    ;SAVE STATUS A
2277  005462  012777  000017  173516          MOV #17,@RXCS                   ;SET READ STATUS B FUNCTION
2278  005470  004737  006412          5$:     JSR PC,SDN
2279  005474  000775                          BR 5$
2280  005476  017702  173506                  MOV @RXDB,R2                    ;SAVE STATUS B
2281
2282                                  ;R0 = ADDR OF LAST DATA BYTE ; R1 = STATUS A ; R2 = STATUS B
2283
2284  005502  104000                          ERROR
2285

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

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

2314  005504  104406          4$:     SUBSCOPE
2315  005506  005000                  CLR R0
2316  005510  005001                  CLR R1
2317  005512  004737  004000          JSR PC,T7EMPTY                  ;EMPTY BUFFER AND CHECK CONTENTS
2318  005516  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.

005520  105737  012446          T22:    TSTB UNITSEL                    ;IF UNIT 0 IS NOT SELECTED SKIP THIS TEST
005524  100022                          BPL 2$
005526  005037  012046                  CLR PAT
005532  004737  003654                  JSR PC,T6FILL                           ;LOAD THE SECTOR BUFFER WITH 0
005536  005237  012046                  INC PAT                                 ;RELOAD CORE BUFFER WITH 1'S
005542  004737  012002                  JSR PC,GETPATTERN
005546  004737  010532                  JSR PC,ADJSUM
005552  052777  040001  173426          BIS #RECAL,@RXCS                        ;SET THE INIT. BIT
005560  004737  006412          1$:     JSR PC,SDN
005564  000775                          BR 1$
005566  004737  004000                  JSR PC,T7EMPTY                          ;EMPTY THE SECTOR BUFFER AND CHECK IT.
005572  000137  004106          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.

005576  005037  012046          T23:    CLR PAT
005602  004737  003654                  JSR PC,T6FILL                   ;LOAD SECTOR BUFFER WITH 0'S
005606  005237  012046                  INC PAT
005612  004737  012356                  JSR PC,GETUNIT
005616  004737  012002                  JSR PC,GETPATTERN               ;RELOAD CORE BUFFER WITH 1'S
005622  004737  010532                  JSR PC,ADJSUM                   ;SET UP FOR CHECK SUM
005626  012737  000007  007472          MOV #7,FUNCTION                 ;SET READ FUNCTION AND GO
005634  004737  005436                  JSR PC,T21X                     ;ISSUE COMMAND,WAIT FOR DONE,& TEST DATA
005640  000137  004106                  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.

005644  012737  000002  012046  T24:    MOV #2,PAT                      ;SET DATA PATTERN TO FLOATING 0
005652  013702  001204          T24X:   MOV KRXVEC,R2                   ;SET INTERRUPT ADDRESSES
005656  012722  011256                  MOV #INTSERV,(R2)+
005662  012712  000340                  MOV #PR7,(R2)
005666  004737  006614                  JSR PC,DRVSWP                   ;GO TRANSFER THE DATA
005672  000137  004106                  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.

005676  012737  000003  012046  T25:    MOV #3,PAT                      ;SET DATA PATTERN TO FLOATING 1
005704  000137  005652                  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.


005710  052737  000200  012660  T26:    BIS #BIT7,SEQUEN                ;SPECIAL DECREMENT SEQUENCE
005716  012737  000007  012046          MOV #7,PAT                      ;SELECT RANDOM DATA
005724  013702  001204                  MOV KRXVEC,R2
005730  012722  011256                  MOV #INTSERV,(R2)+
005734  012712  000340                  MOV #PR7,(R2)
005740  004737  006670                  JSR PC,WTRDCK
005744  042737  000200  012660          BIC #BIT7,SEQUEN
005752  000137  004106                  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

005756  104400  015633       BUSERR: TYPE,   LOC4M                   ;TYPE MESSAGE INDICATING AN
                                                                      ;UNEXPECTED BUS TIMEOUT OCCURRED
005762  012646                       MOV    (SP)+,-(SP)        ;;SAVE (SP)+ FOR TYPEOUT
                                                                ;;SETUP TO TYPE PC WHERE TIMEOUT OCCURRED
005764  104402                       TYPOS                     ;;GO TYPE--OCTAL ASCII
005766     006                       .BYTE  6                  ;;TYPE 6 DIGITS
005767     000                       .BYTE  0                  ;;SUPPRESS LEADING ZEROS
005770  104400  015746               TYPE,  PCM                      ;TYPE MESSAGE '=PC'
005774  012716  002276               MOV    #REBEGIN,(SP)            ;SET RETURN 'PC' TO START THE
                                                                      ;PROGRAM OVER AGAIN
006000  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

006002  104400  015700       RESERR: TYPE,   LOC10M                  ;TYPE MESSAGE INDICATING AN
                                                                      ;UNEXPECTED RESERVED INSTRUCTION
                                                                      ;TRAP OCCURRED
006006  012646                       MOV    (SP)+,-(SP)        ;;SAVE (SP)+ FOR TYPEOUT
                                                                ;;SETUP TO TYPE PC WHERE RESERVED TRAP OCCURRED
006010  104402                       TYPOS                     ;;GO TYPE--OCTAL ASCII
006012     006                       .BYTE  6                  ;;TYPE 6 DIGITS
006013     000                       .BYTE  0                  ;;SUPPRESS LEADING ZEROS
006014  104400  015746               TYPE,  PCM                      ;TYPE MESSAGE '=PC'
006020  012716  002276               MOV    #REBEGIN,(SP)            ;SET RETURN 'PC' TO START THE
                                                                      ;PROGRAM OVER AGAIN
006024  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:')

006026  013700  001214       CPUPRI: MOV    BRLEV,R0                 ;GET THE PROPOSED RX11 DEVICE
                                                                      ;INTERRUPT PRIORITY LEVEL VALUE
006032  105701                       TSTB   R1                       ;IS CPU LEVEL TO BE THE SAME AS
                                                                      ;THE DEVICE LEVEL OR 1 LESS?
006034  100401                       BMI    1$                       ;BRANCH IF SAME AS!
006036  005300                       DEC    R0                       ;DROP DEVICE LEVEL PRIORITY
                                                                      ;BY 1 LEVEL FOR PSW
006040  006300               1$:     ASL    R0                       ;FORM BITS <7-5> FOR PSW
006042  006300                       ASL    R0                       ;       .
006044  006300                       ASL    R0                       ;       .
006046  006300                       ASL    R0                       ;       .
006050  006300                       ASL    R0                       ;       .
006052  042700  000037               BIC    #37,R0                   ;ENSURE THAT T,N,Z,V, & C BITS
                                                                      ;FOR THE PROCESSOR ARE CLEAR
006056  000207                       RTS    PC                       ;RETURN TO MAINLINE CODE
```

```
                                        .SBTTL  " ERROR " TRAP SERVICE ROUTINE

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

                                        ;                       " ERROR "

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

006060  011637  006314          XERROR: MOV @ SP, EPCSCOPE              ; RETURN ADDRESS FROM " ERROR"
006064  062737  000002  006314          ADD #2, EPCSCOPE                ; NOW (EPCSCOPE) = SUBSCOPE+2, OR SCOPE+2
006072  005237  006312          INCERRORS: INC ERRORS
006076  001775                          BEQ INCERRORS

                                        ; DATA SW 13 = 0 TO PRINT APPROPRIATE ERROR MESSAGE

006100  032777  020000  173110          BIT #SW13,@SWR
006106  001056                          BNE NOPRINT
006110  005037  002336                  CLR CCOUNT
006114  032737  000400  012446          BIT #BIT8,UNITSEL               ;WAS PREVIOUS ERROR REPORTED ON THIS PASS
006122  001002                          BNE 1$
006124  104400  014546                  TYPE, MXEHEADER

006130  104400  015030          1$:     TYPE, MCRLF
006134  011604                          MOV @ SP, R4                    ; ERADR
006136  162704  000002                  SUB #2, R4
006142  010446                          MOV R4, -(SP)
006144  104402                          TYPOS
006146     006                          .BYTE 6
006147     001                          .BYTE 1
006150  104400  015542                  TYPE, SPACE
006154  013746  002342                  MOV FAST, -(SP)                 ; FAST (FIRST ADDRESS OF SELECTED TEST)
006160  104403                          TYPON
006162  104400  015542                  TYPE, SPACE
006166  013746  006374                  MOV PCSCOPE, -(SP)              ; FAPT (FIRST ADDRESS OF PRESENT TEST)
006172  104403                          TYPON
006174  104400  015542                  TYPE, SPACE
006200  010246                          MOV R2, -(SP)                   ; BLANK
006202  104403                          TYPON
006204  104400  015542                  TYPE, SPACE
006210  010046                          MOV R0, -(SP)                   ; EXPECTED (GOOD) RESULT OF TEST
006212  104403                          TYPON
006214  104400  015542                  TYPE, SPACE
006220  010146                          MOV R1, -(SP)                   ; ACTUAL (BAD) RESULT OF TEST
006222  104403                          TYPON
006224  104400  015542                  TYPE, SPACE
006230  013737  002340  006242          MOV PASS,2$
006236  004537  014504                  JSR R5,SGLDEC
006242  000000          2$:     OPEN
```

```
                                        ; DATA SW 0 = 0 TO RING BELL AT ERROR
                                        
006244  052737  000400  012446  NOPRINT: BIS #BIT8,UNITSEL              ;SET HARD ERROR FLAG
006252  004737  006272                   JSR PC,DING
                                        
                                        ; DATA SW 15 = 1 TO HALT AT ERROR
                                        
006256  032777  100000  172732  1$:      BIT #SW15,@SWR
006264  001401                           BEQ 2$
006266  000000                           HALT
006270  000002                  2$:      RTI
                                        
006272  032777  000001  172716  DING:    BIT #SW0,@SWR
006300  001002                           BNE 1$
006302  104400  006310                   TYPE ,MABELL
006306  000207                  1$:      RTS PC
                                        
006310  000007                  MABELL:  .ASCIZ <07>                     ; DING - A - LING
                                         .EVEN
                                        
006312  000000                  ERRORS:  0
006314  000000                  EPCSCOPE: 0
```

```
                                        .SBTTL " SCOPE " TRAP SERVICE ROUTINE

                                ;                       " SCOPE "

006316  005737  006312          XSCOPE: TST ERRORS
006322  001015                          BNE SCOPING

                                ; NO ERRORS HAVE BEEN DETECTED

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

                                ; (IN CASE ERRORS ARE DETECTED LATER)

006324  005037  006312          NOSCOPE: CLR ERRORS
006330  011637  006374                  MOV @ SP, PCSCOPE
006334  000002                          RTI

                                ;                       " SUBSCOPE "

006336  005737  006312          XSUBSCOPE: TST ERRORS
006342  001001                          BNE 1$
006344  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

006346  021637  006314          1$:     CMP @ SP, EPCSCOPE
006352  001401                          BEQ SCOPING
006354  000002                          RTI

                                ; SW 11 = 1 TO LOCK ON SCOPING LOOP

                                ; THIS IS A SCOPING LOOP

006356  032777  004000  172632  SCOPING: BIT #SW11,@SWR
006364  001757                          BEQ NOSCOPE                     ;DO NOT LOOP ON ERROR
006366  013716  006374                  MOV PCSCOPE, @ SP
006372  000002                          RTI                             ; LOCK FOR SCOPE LOOP
006374  000000                  PCSCOPE:        0
```

```
                                        ; WAIT FOR TRANSFER REQUEST FLAG TO SET
                                        ;
006376  105777  172604         STR:    TSTB @ RXCS
006402  100002                         BPL RTSPC
006404  062716  000002                 ADD #2, @ SP                    ; ADJUST FOR EXIT
006410  000207                 RTSPC:  RTS PC
                                        ;
                                        ; WAIT FOR THE DONE FLAG TO SET
                                        ;
006412  032777  000040  172566 SDN:     BIT #BIT5, @ RXCS               ; TEST THE DONE BIT # 5
006420  001047                         BNE XSDN
006422  062737  000001  006560         ADD #1,HANGER
006430  005537  006562                 ADC HANGPL
006434  001401                         BEQ HUNGUP
006436  000207                         RTS PC
                                        ;
                                        ; THE DEVICE TEST IS HUNG - DONE HAS NOT SET
                                        ;
                                        ; (R0) = N/A ; (R1) = N/A ; (R2) = N/A
                                        ;
006440  104000                 HUNGUP: ERROR

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

2602    006442  032777  020000  172546          BIT #SW13,@SWR
2603    006450  001402                          BEQ 1$
2604    006452  000137  002032                  JMP MORETESTS
2605    006456  104400  006476          1$:     TYPE, MHUNGPC
2606
2607                                    ; THE PC IS ALREADY ON THE STACK
2608
2609    006462  162716  000002                  SUB #2, @ SP
2610    006466  104402                          TYPOS
2611    006470     006                          .BYTE 6
2612    006471     001                          .BYTE 1
2613    006472  000137  002032                  JMP MORETESTS
2614
2615    006476  005015  042504  044526  MHUNGPC: .ASCIZ <15><12>"DEVICE TEST HUNG @ PC "
```

```
2616  006504  042503  052040  051505
2617  006512  020124  052510  043516
2618  006520  040040  050040  020103
2619  006526     000
2620  006527     040  040520  051523  MPASS:  .ASCIZ " PASS ="
2621  006534  036440     000
2622          006540                          .EVEN
2623
2624  006540  005037  006560          XSDN:   CLR HANGER
2625  006544  012737  177740  006562          MOV #177740,HANGPL
2626  006552  062716  000002                  ADD #2, @ SP                ; UPDATE FOR EXIT
2627  006556  000207                          RTS PC
2628  006560  000000                  HANGER: 0
2629  006562  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.

006564  004737  012474        RDONLY:       JSR PC,INITTRACKS
006570  004737  012356                      JSR PC,GETUNIT
006574  004737  012626        1$:           JSR PC,GETTRACK
006600  004737  007576                      JSR PC,READ
006604  005337  012646                      DEC TRKCNTR
006610  001371                              BNE 1$
006612  000207                              RTS PC

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

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

006614  004737  012002        DRVSWP:       JSR PC,GETPATTERN
006620  004737  012474                      JSR PC,INITTRACKS
006624  004737  012356        1$:           JSR PC,GETUNIT
006630  004737  012626                      JSR PC,GETTRACK
006634  004737  006736                      JSR PC,WRITE
006640  004737  010400                      JSR PC,READCHK
006644  004737  012356                      JSR PC,GETUNIT
006650  004737  006736                      JSR PC,WRITE
006654  004737  010400                      JSR PC,READCHK
006660  005337  012646                      DEC TRKCNTR
006664  001357                              BNE 1$
006666  000207                              RTS PC

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

                                        ;WRITE ALL SECTORS AND READ/VERIFY ALL SECTORS

006670  004737  012002        WTRDCK:       JSR PC,GETPATTERN
006674  004737  012474        XWTRDCK:      JSR PC,INITTRACKS
006700  004737  012356                      JSR PC,GETUNIT
006704  004737  012626        1$:           JSR PC,GETTRACK
006710  004737  006736                      JSR PC,WRITE
006714  004737  010400                      JSR PC,READCHK
006720  005337  012646                      DEC TRKCNTR
006724  001367                              BNE 1$
006726  004737  012450                      JSR PC,DONE             ;HAVE BOTH DRIVES BEEN TESTED
006732  000207                              RTS PC                  ;YES
006734  000757                              BR XWTRDCK              ;NO, GO TO OTHER UNIT
```

```
                                        .SBTTL  WRITE FUNCTION

006736 004737 013046          WRITE:    JSR PC,INITSECTOR     ;SET UP FIRST,LAST,AND SECTOR COUNTER
006742 004737 013144          XWRITE:   JSR PC,GETSECTOR      ;PICK UP NEXT SECTOR
006746 004737 010532          FILLBUF:  JSR PC,ADJSUM         ;ADJUST DATA BUFFER AND CHECK SUM FOR ADDRESSES
006752 012746 007130                    MOV #FILLDONE,-(SP)   ;PUT GOOD RETURNON STACK
006756 012746 007024                    MOV #FILLER,-(SP)     ;PUT ERROR RETURN ON STACK
006762 005037 011206                    CLR BYTECNTR
006766 005046                     CLR   -(SP)                 ;LOWER 'CPU' LEVEL
006770 012746 006776              MOV   #1$,-(SP)             ;SET RETURN 'PC'
006774 000002                     RTI                         ;GET 'CPU' LEVEL INTO 'PSW'
006776 012777 000101 172202 1$:         MOV #FBIE,@RXCS       ;EXECUTE FILLBUFER COMMAND
007004 105777 172176          FILLFLAG: TSTB @RXCS            ;TEST FOR TRANSFER REQUEST FLAG
007010 100375                           BPL FILLFLAG
007012 112077 172172          XFRBYTE:  MOVB (R0)+,@RXDB      ;TRANSFER DATA BYTE
007016 005237 011206                    INC BYTECNTR
007022 000770                           BR FILLFLAG           ;WAIT FOR NEXT TR FLAG

007024 005726                 FILLER:   TST (SP)+             ;REMOVE THE DONE RETURN FROM THE STACK
007026 012737 015267 007066             MOV #MFIL,PTYP1+2     ;PUT ADDR OF FILLBUF MESSAGE IN PAR ERR TYPOUT 1
007034 012737 006742 007126             MOV #XWRITE,PCONT+2   ;IF NO LOOP ON ERROR GO TO NEXT SECTOR
007042 012737 006746 007122             MOV #FILLBUF,PLOOP+2  ;IF LOOP ON ERROR RETURN THROUGH PLOOP
007050 000137 007054                    JMP PARTEST           ;PRINT OUT PAR ERR AND TEST CONDITIONS FOR RETRY

007054 032777 020000 172134 PARTEST:    BIT #SW13,@SWR        ;TEST DON'T PRINT ERROR SWITCH
007062 001006                           BNE CONT4
007064 104400 000000          PTYP1:    TYPE ,OPEN            ;PRINT THE PARITY ERROR MESSAGE
007070 104400 015500                    TYPE ,MPAR
007074 104400 015030                    TYPE ,MCRLF
007100 005777 172112          CONT4:    TST @SWR              ;TEST HALT ON ERROR SWITCH
007104 100001                           BPL CONT13
007106 000000                 HLT6:     HALT                  ;HALT ON ERROR
007110 032777 004000 172100 CONT13:     BIT #SW11,@SWR        ;TEST LOOP ON ERROR SWITCH
007116 001402                           BEQ PCONT             ;IF NOT SET GO TO NEXT SECTOR
007120 000137 006746          PLOOP:    JMP FILLBUF           ;RETURN TO LOOP ON TEST THROUGH HERE
007124 000137 007714          PCONT:    JMP NEXTRD            ;GO TO NEXT SECTOR THROUGH HERE

007130 005037 006560          FILLDONE: CLR HANGER
007134 012746 007226                    MOV #WRTDONE,-(SP)    ;SET GOOD RETURN ON STACK
007140 012746 007242                    MOV #WRTER,-(SP)      ;SET ERROR RETURN ON STACK
007144 112737 000105 007472             MOVB #WRTIE,FUNCTION  ;SET FUNCTION WORD TO WRITE
007152 022737 005676 006374             CMP #T25,PCSCOPE      ;IS THIS THE DELETED DATA TEST
007160 001003                           BNE 1$
007162 112737 000115 007472             MOVB #WTDOIE,FUNCTION
007170 004737 007424          1$:       JSR PC,COMMWORD       ;TRANSFER COMMAND TO DRIVE
007174 005046                     CLR   -(SP)                 ;LOWER 'CPU' LEVEL
007176 012746 007204              MOV   #2$,-(SP)             ;SET RETURN 'PC'
007202 000002                     RTI                         ;GET 'CPU' LEVEL INTO 'PSW'
007204 032777 000040 171774 2$:         BIT #DONEBIT,@RXCS    ;WAIT FOR DONE
007212 001774                           BEQ 2$
007214 005237 006560          3$:       INC HANGER            ;WAIT FOR INTERRUPT
007220 001375                           BNE 3$
007222 000137 011216                    JMP NOINTER           ;NO INTERRUPT ERROR
```

```
2730  007226  005337  013136          WRTDONE:        DEC SECCNTR             ;TEST SECTOR COUNTER
2731  007232  001001                                  BNE 2$                  ;NOT LAST SECTOR GO TO NEXT ONE
2732  007234  000207                                  RTS PC
2733  007236  000137  006742          2$:             JMP XWRITE
2734
2735  007242  005726                  WRTER:          TST (SP)+               ;REMOVE THE DONE RETURN FROM THE STACK
2736  007244  032737  000002  011670                  BIT #BIT1,ASTAT         ;IS THIS A PARITY ERROR
2737  007252  001413                                  BEQ WRTSEK              ;NO, IT MUST BE A SEEK ERROR
2738                                  ;PARITY ERROR DURING A WRITE FUNCTION
2739  007254  012737  015471  007066                  MOV #MWRITE,PTYP1+2     ;PUT ADDR OF WRITE MESSAGE IN PAR ER TYPOUT 1
2740  007262  012737  007226  007126                  MOV #WRTDONE,PCONT+2    ;IF NO LOOP GO TO NEXT SECTOR
2741  007270  012737  007130  007122                  MOV #FILLDONE,PLOOP+2   ;IF LOOP RETURN THROUGH PLOOP TO REWRITE
2742  007276  000137  007054                          JMP PARTEST             ;GO INC LOG AND TEST FOR RETRY
2743
2744                                  ;SEEK ERROR DURING A WRITE FUNCTION
2745  007302  012737  006746  007374  WRTSEK:         MOV #FILLBUF,SEKRTY+2   ;SETUP FOR WRT RETRY ON SEEK ERROR
2746                                                                          ;(AFTER A RECAL. THE CONTENTS OF SECTOR 1,
2747                                                                          ;TRACK 1 ARE LOADED INTO THE SECTOR BUFFER.
2748                                                                          ;TO REWRITE THE CORRECT DATA THE PROGRAM
2749                                                                          ;MUST REFILL THE SECTOR BUFFER.
2750  007310  012737  015471  007336                  MOV #MWRITE,STYP1+2     ;PUT ADDR OF WRITE MESSAGE IN SEEK ER TYPEOUT 1
2751  007316  004737  007324                          JSR PC,SEEKER           ;RECORD SEEK ERROR
2752  007322  000741                                  BR WRTDONE              ;GO TO NEXT SECTOR CAN'T FIND THIS ONE
2753
2754  007324  032777  020000  171664  SEEKER:         BIT #SW13,@SWR          ;CHECK DON'T PRINT ERROR SWITCH
2755  007332  001004                                  BNE SWHLT1
2756  007334  104400  015471          STYP1:          TYPE ,MWRITE            ;PRINT WRITE (READ) SEEK ERROR
2757  007340  004737  007376                          JSR PC,SEKTYP
2758  007344  005777  171646          SWHLT1:         TST @SWR                        ;TEST THE HALT ON ERROR SWITCH
2759  007350  100001                                  BPL CONT14
2760  007352  000000                  HLT7:           HALT                    ;HALT ON THE ERROR
2761  007354  004737  007476          CONT14:         JSR PC,HOME             ;RECALIBRATE ON SEEK ERRORS
2762  007360  032777  004000  171630                  BIT #SW11,@SWR          ;CHECK THE LOOP ON ERROR SWITCH
2763  007366  001001                                  BNE SEKRTY              ;IF SET LOOP ON THE ERROR THROUGH SEEK RETRY.
2764  007370  000207                                  RTS PC
2765  007372  000137  006746          SEKRTY:         JMP FILLBUF             ;RETRY WRITE COMMAND (READ COMAND)
2766
2767  007376  104400  015456          SEKTYP:         TYPE ,MSEEK             ;TYPE SEEK ERROR
2768  007402  104400  014742                          TYPE ,MADRES            ;TYPE ADDRESS OF TRACK MOVED FROM
2769  007406  013746  012652                  MOV     PRESTRK,-(SP)   ;;SAVE PRESTRK FOR TYPEOUT
2770  007412  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
2771  007414     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
2772  007415     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
2773  007416  104400  015030                          TYPE ,MCRLF
2774  007422  000207                                  RTS PC
2775
```

```
 2776  007424  153737  012446  007472  COMMWORD:   BISB UNITSEL,FUNCTION  ;SET UNIT SELECTION BIT IN COMMAND WORD
 2777  007432  013777  007472  171546              MOV FUNCTION,@RXCS     ;SEND OUT COMMAND TO DRIVE
 2778  007440  004737  006376          1$:         JSR PC,STR             ;WAIT FOR TR FLAG
 2779  007444  000775                              BR 1$
 2780  007446  113777  013140  171534              MOVB TSECTOR,@RXDB     ;SEND OUT TARGET SECTOR
 2781  007454  004737  006376          2$:         JSR PC,STR             ;WAIT FOR TR FLAG
 2782  007460  000775                              BR 2$
 2783  007462  113777  012650  171520              MOVB TARGET,@RXDB      ;SEND OUT TARGET TRACK
 2784  007470  000207                              RTS PC
 2785
 2786  007472  000000          FUNCTION:   0
 2787  007474  000000          DATACK:     0                              ;DATA CHECK ON CRC ERROR FLAG
 2788
 2789  007476  032777  000400  171512  HOME:       BIT #SWB,@SWR          ;TEST NO RECAL SWITCH
 2790  007504  001033                              BNE RTN
 2791  007506  012777  040001  171472              MOV #RECAL,@RXCS       ;ISSUE RECAL FUNCTION
 2792  007514  004737  006412          2$:         JSR PC,SDN
 2793  007520  000775                              BR 2$
 2794  007522  005777  171460                      TST @RXCS              ;WAS THERE AN ERROR
 2795  007526  100017                              BPL XHOME              ;NO
 2796  007530  032777  020000  171460              BIT #BIT13,@SWR        ;YES, SHOULD IT BE PRINTED
 2797  007536  001002                              BNE 1$                 ;NO
 2798  007540  004737  011674                      JSR  PC,RDCODE
 2799  007544  005777  171446          1$:         TST @SWR                       ;TEST HALT ON ERROR SWITCH
 2800  007550  100001                              BPL 3$
 2801  007552  000000                              HALT
 2802  007554  032777  004000  171434  3$:         BIT #SW11,@SWR         ;TEST LOOP ON ERROR SWITCH
 2803  007562  001345                              BNE HOME
 2804  007564  000207                              RTS PC
 2805  007566  012737  000001  012652  XHOME:      MOV #1,PRESTRK         ;SET THE PRESENT TRACK TO TRACK 1
 2806  007574  000207          RTN:        RTS PC
 2807
```

```
                                        .SBTTL  READ DATA FROM THE DISKETTE
                                
007576  004737  013046          READ:         JSR PC,INITSECTOR
007602  004737  013144          XREAD:        JSR PC,GETSECTOR
007606  005037  007474          REREAD:       CLR DATACK              ;CLEAR CRC DATA CHECK FLAG
007612  005037  006560                        CLR HANGER
007616  012746  007672                        MOV #RDDONE,-(SP)       ;SET GOOD RETURN ON STACK
007622  012746  007724                        MOV #RDERR,-(SP)        ;SET READ ERROR RETURN ON STACK
007626  112737  000107  007472                MOVB #RDIE,FUNCTION
007634  004737  007424                        JSR PC,COMMWORD
007640  005046                      CLR       -(SP)                   ;LOWER 'CPU' LEVEL
007642  012746  007650              MOV       #1$,-(SP)               ;SET RETURN 'PC'
007646  000002                      RTI                               ;GET 'CPU' LEVEL INTO 'PSW'
007650  032777  000040  171330      1$:       BIT #DONEBIT,@RXCS      ;WAIT FOR DONE BIT
007656  001774                              BEQ 1$
007660  005237  006560              2$:       INC HANGER              ;WAIT FOR INTERRUPT
007664  001375                              BNE 2$
007666  000137  011216                        JMP NOINTER             ;NO INTERRUPT ON DONE
                                
007672  022737  005310  006374    RDDONE:       CMP #T20,PCSCOPE        ;IS THIS THE READ ONLY TEST (T20)
007700  001405                              BEQ NEXTRD              ;YES,DON'T CHECK FOR DELETED DATA
007702  004737  010162                        JSR PC,DDCHK            ;CHECK FOR DELETED DATA INDICATOR
007706  005701                              TST R1                  ;BIT 15 OF R1 IS READ 1 SECTOR FLAG
007710  100001                              BPL NEXTRD
007712  000207                              RTS PC                  ;IF SET,GO VERIFY DATA JUST READ
007714  005337  013136          NEXTRD:       DEC SECCNTR
007720  001330                              BNE XREAD
007722  000207                              RTS PC                  ;READ FUNCTION IS DONE
                                
007724  005726                  RDERR:        TST (SP)+               ;REMOVE THE DONE RETURN FROM THE STACK
007726  032737  000002  011670              BIT #BIT1,ASTAT         ;IS THIS A PARITY ERROR
007734  001413                              BEQ 1$                  ;NO, SEE IF ITS A CRC ERROR
                                    ;PARITY ERROR DURING A READ FUNCTION
007736  012737  015431  007066              MOV #MREAD,PTYP1+2      ;PUT ADDR OF READ MESSAGE IN PAR ERR TYPEOUT 1
007744  012737  007606  007122              MOV #REREAD,PLOOP+2     ;IF LOOP ON ERROR LOOP THROUGH PLOOP
007752  012737  007714  007126              MOV #NEXTRD,PCONT+2     ;IF NO LOOP GO TO NEXT READ
007760  000137  007054                        JMP PARTEST             ;RECORD PARITY ERROR AND RETRY FUNCTION
007764  032737  000001  011670      1$:       BIT #BIT0,ASTAT         ;IS THIS A CRC ERROR
007772  001011                              BNE CRCER               ;YES GO TEST AND LOG IT
                                    ;SEEK ERROR DURING A READ FUNCTION
007774  012737  007606  007374              MOV #REREAD,SEKRTY+2    ;SET SEEK CONTINUE FOR READ RETRY
010002  012737  015431  007336              MOV #MREAD,STYP1+2      ;SET ADDR OF READ MESSAGE IN SEEK ER TYPEOUT 1
010010  004737  007324                        JSR PC,SEEKER           ;RECORD SEEK ERROR
010014  000737                              BR NEXTRD               ;GO TO NEXT SECTOR,CAN'T READ THIS ONE
```

```
                                        ;CRC ERROR DETECTED WHILE READING
                                
010016  005701                  CRCER:    TST R1                  ;IF READ ONLY, REPORT DATA CRC ERROR
010020  100032                            BPL DATACRC
010022  005237  007474                    INC DATACK              ;SET DATA CHECK FLAG
010026  004737  010410                    JSR PC,EMPBUFF          ;CHECK FOR A DATA ERROR
010032  005737  011214                    TST ERCNTR              ;WAS THERE A DATA ERROR
010036  001023                            BNE DATACRC             ;YES, REPORT IT
010040  032777  020000  171150            BIT #SW13,@SWR          ;TEST DON'T PRINT SWITCH
010046  001004                            BNE 2$
010050  104400  015401                    TYPE ,MBADCRC           ;TYPE CRC GENERATOR ERROR
010054  104400  015030                    TYPE ,MCRLF
010060  005777  171132            2$:     TST @SWR                ;TEST HALT ON ERROR SWITCH
010064  100001                            BPL CONT15
010066  000000                  HLT10:    HALT                    ;HALT ON ERROR
010070  032777  004000  171120  CONT15:   BIT #SW11,@SWR          ;CHECK LOOP ON ERROR SWITCH
010076  001001                            BNE 3$
010100  000705                            BR NEXTRD               ;DON'T LOOP GO TO NEXT SECTOR
010102  000137  007606            3$:     JMP REREAD              ;LOOP ON TEST.
                                
                                        ;DATA CRC ERROR
                                
010106  032777  020000  171102  DATACRC:  BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
010114  001004                            BNE 4$
010116  104400  015437                    TYPE ,MCRC              ;TYPE DATA CRC ERROR
010122  104400  015030                    TYPE ,MCRLF
010126  005777  171064            4$:     TST @SWR                ;TEST HALT ON ERROR SWITCH
010132  100001                            BPL CONT16
010134  000000                  HLT12:    HALT                    ;HALT ON ERROR
010136  032777  004000  171052  CONT16:   BIT #SW11,@SWR          ;TEST LOOP ON ERROR
010144  001004                            BNE 5$                  ;IF SET LOOP ON THE TEST
010146  062706  000002                    ADD #2,SP               ;REMOVE READ DONE ADDRESS FROM STACK
010152  000137  007714                    JMP NEXTRD              ;READ NEXT SECTOR CAN'T READ THIS ONE
010156  000137  007606            5$:     JMP REREAD              ;NO,GO REREAD THIS SECTOR
```

```
2899  010162  022737  005676  006374  DDCHK:      CMP #T25,PCSCOPE        ;IS THIS TEST 25
2890  010170  001037                              BNE CONT10
2891  010172  132737  000100  011670              BITB #BIT6,ASTAT        ;THIS IS TEST 25
2892  010200  001053                              BNE RETURN              ;DD BIT SHOULD BE SET
2893  010202  032777  020000  171006              BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
2894  010210  001013                              BNE CONT11
2895  010212  004737  010332                      JSR PC,ERMSG
2896  010216  104400  014657                      TYPE ,MDDMIS            ;TYPE MISSING DELETED DATA BIT
2897  010222  052737  000400  012446  DDERR:     BIS #BIT8,UNITSEL       ;SET HARD ERROR FLAG
2898  010230  004737  011572                      JSR PC,TYPADR           ;TYPE ADDRESS OF ERROR
2899  010234  104400  015030                      TYPE ,MCRLF
2900  010240  005777  170752          CONT11:     TST @SWR                ;TEST HALT ON ERROR SWITCH
2901  010244  100001                              BPL CONT17
2902  010246  000000                  HLT13:      HALT                    ;HALT ON DELETED DATA ERROR
2903  010250  032777  004000  170740  CONT17:     BIT #SW11,@SWR          ;TEST LOOP ON ERROR
2904  010256  001402                              BEQ 4$
2905  010260  000137  007606                      JMP REREAD              ;LOOP ON TEST
2906  010264  000137  007714          4$:         JMP NEXTRD              ;READ NEXT SECTOR
2907  010270  032737  000100  011670  CONT10:     BIT #BIT6,ASTAT         ;THIS IS NOT A DELETED DATA TRANSFER
2908  010276  001414                              BEQ RETURN
2909  010300  052737  000400  012446              BIS #BIT8,UNITSEL       ;SET HARD ERROR FLAG
2910  010306  032777  020000  170702              BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
2911  010314  001351                              BNE CONT11
2912  010316  004737  010332                      JSR PC,ERMSG
2913  010322  104400  014631                      TYPE ,MUNXDD            ;TYPE UNEXPECTED DELETED DATA BIT
2914  010326  000735                              BR DDERR
2915  010330  000207                  RETURN:     RTS PC
2916
2917
2918  010332  104400  015033          ERMSG:      TYPE ,MERHEADER
2919  010336  013746  006374                  MOV     PCSCOPE,-(SP)   ;;SAVE PCSCOPE FOR TYPEOUT
2920  010342  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
2921  010344     006                          .BYTE   6               ;;TYPE 6 DIGITS
2922  010345     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
2923  010346  104400  006527                      TYPE ,MPASS
2924  010352  013737  002340  010364              MOV PASS,1$
2925  010360  004537  014504                      JSR R5,SGLDEC
2926  010364  000000                  1$:         OPEN
2927  010366  104400  015030                      TYPE ,MCRLF
2928  010372  004737  006272                      JSR PC,DING
2929  010376  000207                              RTS PC
2930
2931
```

```
                                        .SBTTL READ AND VERIFY DATA
                                        ;
                                        ;READ A SECTOR,EMPTY THE SECTOR BUFFER AND VERIFY
                                        ;THE DATA READ AGAINST CORE DATA BUFFER
                                        ;
010400  052701  100000        READCHK:  BIS #BIT15,R1           ;SET READ ONE SECTOR FLAG
010404  004737  007576                  JSR PC,READ             ;GO READ ONE SECTOR
010410  005737  013136        EMPBUFF:  TST SECCNTR             ;IF CLEARED NO SECTORS WERE FOUND
010414  001002                          BNE 1$
010416  000137  011202                  JMP EXIT                ;GO TO NEXT TRACK
010422  005037  011206        1$:       CLR BYTECNTR            ;CLEAR THE BYTE AND ERROR COUNTERS
010426  005037  011214                  CLR ERCNTR
010432  052701  000200                  BIS #BIT7,R1            ;R1 BIT 7 IS USED AS FIRST ERROR FLAG
010436  004737  010532                  JSR PC,ADJSUM           ;ADJUST DATA AND CK SUM FOR ADDRESSES
010442  005037  010622                  CLR CKSUM               ;SET UP FOR CHECK SUM ACCUMULATION
010446  012746  011046                  MOV #EMPDONE,-(SP)      ;SET UP RETURN ADDRESSES
010452  012746  010624                  MOV #EMPER,-(SP)
010456  005046                  CLR     -(SP)                   ;LOWER 'CPU' LEVEL
010460  012746  010466          MOV     #2$,-(SP)               ;SET RETURN 'PC'
010464  000002                  RTI                             ;GET 'CPU' LEVEL INTO 'PSW'
010466  012777  000103  170512  2$:       MOV #EBIE,@RXCS         ;LOAD EMPTY BUFFER FUNCTION
010474  105777  170506        EMPFLAG:  TSTB @RXCS              ;TEST FOR TR FLAG
010500  100375                          BPL EMPFLAG
                                        ;
010502  117737  170502  011210  CKBYTE:  MOVB @RXDB,BADBYTE      ;SAVE BYTE FROM DISKETTE
010510  063737  011210  010622          ADD BADBYTE,CKSUM       ;ACCUMULATE CHECK SUM
010516  123720  011210                  CMPB BADBYTE,(R0)+      ;COMPARE AGAINST GOOD BYTE
010522  001054                          BNE DATAER              ;IF NOT EQUAL GO TO DATAER
010524  005237  011206                  INC BYTECNTR
010530  000761                          BR EMPFLAG              ;GET NEXT BYTE
                                        ;
010532  113737  012650  016300  ADJSUM:  MOVB TARGET,BUFADR     ;SET FIRST AND SECOND BYTES WITH ADDRESSES
010540  113737  013140  016301          MOVB TSECTOR,BUFADR+1
010546  013737  012260  010622          MOV SUM,CKSUM           ;GET THE PATTERN SUM
010554  063737  012650  010622          ADD TARGET,CKSUM        ;ADD TRACK ADDRESS TO CHECK SUM
010562  063737  013140  010622          ADD TSECTOR,CKSUM       ;ADD SECTOR ADDRESS TO CHECK SUM
010570  113737  010622  016476          MOVB CKSUM,BUFADR+176   ;INSERT CHECK SUM TO DATA BUFFER
010576  106337  010622                  ASLB CKSUM              ;GENERATE NEGITIVE CHECK SUM
010602  105437  010622                  NEGB CKSUM
010606  113737  010622  016477          MOVB CKSUM,BUFADR+177   ;INSERT NEG.SUM INTO DATA BUFFER
010614  012700  016300                  MOV #BUFADR,R0          ;SET ADDRESS OF BYTE IN R0
010620  000207                          RTS PC                  ;RETURN
                                        ;
010622  000000                  CKSUM:    0
                                        ;
010624  005726                  EMPER:    TST (SP)+             ;REMOVE THE DONE RETURN FROM THE STACK
010626  012737  015303  007066          MOV #MEMPTY,PTYP1+2     ;PUT ADDR OF EMPTYBUF MESSAGE IN PAR ER TYPOUT 1
010634  012737  010410  007122          MOV #EMPBUFF,PLOOP+2    ;RETURN THROUGH HERE TO LOOP ON ERROR
010642  012737  011164  007126          MOV #NXREAD,PCONT+2     ;IF NO LOOP ON ERROR GO TO NEXT SECTOR
010650  000137  007054                  JMP PARTEST             ;REPORT PARITY ERROR
                                        ;
```

```
 2983  010654  052737  000400  012446  DATAER:     BIS #BIT8,UNITSEL       ;SET THE HAD ERROR FLAG
 2984  010662  005237  011214                      INC ERCNTR              ;INC THE BYTE ERROR COUNTER
 2985  010666  032777  020000  170322              BIT #SW13,@SWR          ;TEST PRINT ERROR SW IN SWR
 2986  010674  001054                              BNE NOERTYP             ;DON'T PRINT THE ERROR
 2987  010676  032777  001000  170312              BIT #SW9,@SWR           ;TEST PRINT 10 ERRORS SWITCH
 2988  010704  001004                              BNE 1$                  ;IF SET PRINT ALL ERRORS
 2989  010706  023727  011214  000012              CMP ERCNTR,#10.         ;HAVE 10 ERRORS BEEN TYPED
 2990  010714  003044                              BGT NOERTYP             ;YES,DON'T PRINT ANY MORE
 2991  010716  105701                      1$:     TSTB R1                 ;TEST FIRST ERROR FLAG
 2992  010720  100014                              BPL TYPERR
 2993  010722  004737  010332                      JSR PC,ERMSG            ;PRINT ADDRESS OF TEST
 2994  010726  104400  014702                      TYPE ,MDERHDR           ;FIRST ERROR, PRINT ERROR HEADER
 2995  010732  104400  015030                      TYPE ,MCRLF
 2996  010736  004737  011572                      JSR PC,TYPADR           ;PRINT TRACK AND SECTOR LOCATIONS
 2997  010742  104400  014771                      TYPE ,MCOLMUN           ;SET UP COLMUN HEADINGS
 2998  010746  042701  000200                      BIC #BIT7,R1            ;CLEAR FIRST ERROR FLAG
 2999  010752  013737  011206  010764  TYPERR:     MOV BYTECNTR,1$         ;PRINT BYTE NUMBER
 3000  010760  004537  014504                      JSR R5,SGLDEC
 3001  010764  000000                      1$:     OPEN
 3002  010766  104400  015025                      TYPE ,DBLSP
 3003  010772  013746  011210                      MOV BADBYTE,-(SP)       ;PRINT BYTE READ FROM DISKETTE
 3004  010776  104402                              TYPOS
 3005  011000  000003                              .WORD 3
 3006  011002  104400  015025                      TYPE ,DBLSP
 3007  011006  114037  011212                      MOVB -(R0),GOODBYTE     ;GET GOOD BYTE
 3008  011012  005200                              INC R0                  ;RETURN R0 TO NEXT BYTE IN BUFFER
 3009  011014  013746  011212                      MOV GOODBYTE,-(SP)
 3010  011020  104403                              TYPON                   ;PRINT GOOD DATA
 3011  011022  104400  015030                      TYPE ,MCRLF
 3012  011026  005777  170164  NOERTYP:           TST @SW                 ;TEST HALT ON ERROR SWITCH
 3013  011032  100001                              BPL CONT20
 3014  011034  000000                      HLT14:  HALT
 3015  011036  005237  011206  CONT20:           INC BYTECNTR
 3016  011042  000137  010474                      JMP EMPFLAG
 3017
```

```
3018  011046  005737  007474          EMPDONE:    TST DATACK          ;WAS THIS READ CHECK CAUSED BY A CRC ERROR
3019  011052  001401                  BEQ 1$      ;NO
3020  011054  000207                  RTS PC      ;YES,RETURN TO CRC HANDLER
3021  011056  005737  011214          1$:         TST ERCNTR          ;WAS THERE ERRORS
3022  011062  001440                  BEQ NXREAD  ;NO ERRORS
3023  011064  032777  020000  170124  BIT #SW13,@SWR  ;YES,TEST DON'T PRINT SWITCH
3024  011072  001024                  BNE 2$      ;DON'T PRINT THE ERROR
3025  011074  104400  015234          TYPE ,MERCT ;PRINT THE TOTAL DATA ERROR COUNT
3026  011100  013737  011214  011112  MOV ERCNTR,3$
3027  011106  004537  014504          JSR R5,SGLDEC
3028  011112  000000                  3$:         OPEN
3029  011114  104400  015551          TYPE ,MSUM  ;INDICATE IF CHECK SUM WAS GOOD OR HAD ERRORS
3030  011120  105737  010622          TSTB CKSUM
3031  011124  001403                  BEQ 4$
3032  011126  104400  015536          TYPE ,MBAD
3033  011132  000402                  BR 5$
3034  011134  104400  015544          4$:         TYPE,MGOOD
3035  011140  104400  015030          5$:         TYPE ,MCRLF
3036  011144  032777  004000  170044  2$:         BIT #SW11,@SWR  ;TEST LOOP ON ERROR SWITCH
3037  011152  001404                  BEQ NXREAD  ;IF NOT SET GO TO NEXT SECTOR
3038  011154  004737  007606          JSR PC,REREAD  ;YES,GO REREAD THE DATA
3039  011160  000137  010410          JMP EMPBUFF ;GO RECHECK THE DATA
3040  011164  005337  013136          NXREAD:     DEC SECCNTR
3041  011170  001404                  BEQ EXIT
3042  011172  004737  007602          JSR PC,XREAD  ;READ THE NEXT SECTOR
3043  011176  000137  010410          JMP EMPBUFF
3044  011202  005001                  EXIT:       CLR R1      ;CLEAR THE ONE READ FLAG
3045  011204  000207                  RTS PC
3046
3047  011206  000000                  BYTECNTR:   0
3048  011210  000000                  BADBYTE:    0
3049  011212  000000                  GOODBYTE:   0
3050  011214  000000                  ERCNTR:     0
3051
3052                                  ;*****************************************************************
3053
3054                                          ;AN INTERRUPT DID NOT OCCURE AT A FUNCTION DONE FLAG.
3055
3056  011216  032777  020000  167772  NOINTER:   BIT #SW13,@SWR  ;TEST DON'T PRINT ERROR SWITCH
3057  011224  001006                  BNE 1$
3058  011226  004737  010332          JSR PC,ERMSG
3059  011232  104400  015157          TYPE ,MINTER  ;TYPE NO INTERRUPT ON DONE ERROR
3060  011236  104400  015030          TYPE ,MCRLF
3061  011242  005777  167750          1$:         TST @SWR    ;TEST HALT ON ERROR SWITCH
3062  011246  100001                  BPL CONT21
3063  011250  000000                  HLT15:      HALT        ;HALT ON ERROR
3064  011252  004737  011256          CONT21:     JSR PC,INTSERV  ;JSR TO INTSERV AS IF IT WAS AN INTERRUPT
3065
```

```
                                                .SBTTL  INTERRUPT SERVICE

011256  117737  167726  011670  INTSERV:        MOVB @RXDB,ASTAT        ;SAVE THE ERROR AND STATUS WORD
011264  005777  167716                          TST @RXCS               ;TEST THE ERROR FLAG
011270  100442                                  BMI RXERROR             ;THERE WAS AN ERROR GO REPORT IT
011272  032737  000004  011670                  BIT #BIT2,ASTAT         ;IS INIT DONE SET
011300  001402                                  BEQ 2$                  ;NO,CONTINUE
011302  000137  011530                          JMP RXPWR               ;YES,REPORT POWER FAILED AND RESTART
011306  032737  000003  011670  2$:             BIT #3,ASTAT            ;ARE PAR OR CRC BITS SET
011314  001020                                  BNE 1$                  ;YES GO REPORT ERROR
011316  132777  000040  167662                  BITB #DONEBIT,@RXCS     ;IS DONE SET
011324  001011                                  BNE 3$                  ;IF SET RETURN TO TEST
011326  032777  020000  167662                  BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
011334  001004                                  BNE 4$                  ;DON'T PRINT
011336  104400  015212                          TYPE ,MUKNINT           ;TYPE UNKNOWN INTERRUPT
011342  104400  015030                          TYPE ,MCRLF
011346  000002                  4$:             RTI                     ;RETURN FROM THE INTERRUPT
011350  062706  000006          3$:             ADD #6,SP               ;BYPASS INTERRUPT POINTERS ON STACK
011354  000207                                  RTS PC                  ;RETURN TO PROGRAM
011356  032777  020000  167632  1$:             BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
011364  001004                                  BNE RXERROR
011366  104400  015515                          TYPE ,MNOFLAG           ;TYPE NO STATUS FRROR ERROR
011372  104400  015030                          TYPE ,MCRLF
011376  005237  006312          RXERROR:        INC ERRORS              ;AN ERROR INDICATOR
011402  001775                                  BEQ RXERROR
011404  052737  000400  012446                  BIS #BIT8,UNITSEL       ;SET HARD ERROR FLAG
011412  012777  000017  167566  2$:             MOV #RDER,@RXCS         ;GET THE ERROR CODE
011420  004737  006412          3$:             JSR PC,SDN              ;TEST FOR DONE FLAG
011424  000775                                  BR 3$
011426  032777  000002  167554                  BIT #2,@RXDB            ;WAS THERE A PARITY ERROR
011434  001403                                  BEQ 1$                  ;NO,CONTINUE
011436  004737  011544                          JSR PC,PARTYP           ;YES,GO REPORT THE PARITY ERROR
011442  000763                                  BR 2$                   ;REISSUE THE FUNCTION
011444  117737  167540  011672  1$:             MOVB @RXDB,BSTAT        ;SAVE THE ERROR CODE IN B STATUS
011452  032777  020000  167536  NOPRNT:         BIT #SW13,@SWR          ;TEST PRINT ERROR SWITCH IN SWR
011460  001020                                  BNE 2$
011462  104400  015030                          TYPE ,MCRLF
011466  004737  010332                          JSR PC,ERMSG            ;TYPE ERROR AND MESSAGES
011472  104400  015113                          TYPE ,MRXCS             ;TYPE COMMAND STATUS REGISTER
011476  013746  007472                  MOV     FUNCTION,-(SP)  ;;SAVE FUNCTION FOR TYPEOUT
011502  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
011504     006                          .BYTE   6               ;;TYPE 6 DIGIT(S)
011505     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
011506  004737  011572                          JSR PC,TYPADR           ;TYPE ADDRESSES AND RUN CONDITIONS
011512  104400  015030                          TYPE ,MCRLF
011516  004737  011742                          JSR PC,TYPCODE          ;PRINT THE STATUS REGISTERS
011522  062706  000004          2$:             ADD #4,SP               ;MOVE ERROR RETURN TO TOP OF STACK
011526  000207                                  RTS PC

011530  104400  015566          RXPWR:          TYPE ,MRX11             ;ONLY THE RX11 POWER HAS FAILED
011534  104400  014200                          TYPE ,$POWER            ;PRINT POWER FAILED
011540  000137  001306                          JMP RESTART             ;GO TO RESTART
```

```
3118  011544  104400  015113          PARTYP:         TYPE ,MRXCS
3119  011550  017746  167432                  MOV     @RXCS,-(SP)     ;;SAVE @RXCS FOR TYPEOUT
3120  011554  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3121  011556     006                          .BYTE   6               ;;TYPE 6 DIGIT(S)
3122  011557     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
3123  011560  104400  015500                  TYPE ,MPAR
3124  011564  104400  015030                  TYPE ,MCRLF
3125  011570  000207                          RTS PC
3126
3127  011572  104400  014730          TYPADR:         TYPE ,MTRK              ;TYPE TRACK ADDRESS
3128  011576  013746  012650                  MOV     TARGET,-(SP)    ;;SAVE TARGET FOR TYPEOUT
3129  011602  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3130  011604     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
3131  011605     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
3132  011606  104400  014757                  TYPE ,MSECT             ;TYPE SECTOR ADDRESS
3133  011612  013737  013140  011666          MOV TSECTOR,2$
3134  011620  042737  177740  011666          BIC #177740,2$          ;CLEAR ALL BUT SECTOR ADDRESS
3135  011626  013746  011666                  MOV     2$,-(SP)        ;;SAVE 2$ FOR TYPEOUT
3136  011632  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3137  011634     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
3138  011635     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
3139  011636  104400  015025                  TYPE ,DBLSP
3140  011642  032737  000020  012446          BIT #BIT4,UNITSEL       ;WHITCH DRIVE IS BEING USED
3141  011650  001003                          BNE 1$
3142  011652  104400  015073                  TYPE ,MUNIT0            ;TYPE UNIT 0
3143  011656  000402                          BR 4$
3144  011660  104400  015103          1$:             TYPE ,MUNIT1            ;TYPE UNIT 1
3145  011664  000207                  4$:             RTS PC
3146  011666  000000                  2$:             OPEN
3147
3148  011670  000000                  ASTAT:          0
3149  011672  000000                  BSTAT:          0
3150
3151
3152  011674  117737  167310  011670  RDCODE:         MOVB @RXDB,ASTAT        ;SAVE THE A STATUS
3153  011702  012777  000017  167276  2$:             MOV #RDER,@RXCS         ;READ THE B STATUS REGISTER
3154  011710  004737  006412          3$:             JSR PC,SDN              ;WAIT FOR DONE FLAG
3155  011714  000775                          BR 3$
3156  011716  032777  000002  167264          BIT #2,@RXDB            ;WAS THERE A PARITY ERROR
3157  011724  001403                          BEQ 1$                  ;NO,CONTINUE
3158  011726  004737  011544                  JSR PC,PARTYP           ;YES,REPORT THE PARITY ERROR
3159  011732  000763                          BR 2$                   ;RETRY READING STATUS B
3160  011734  117737  167250  011672  1$:             MOVB @RXDB,BSTAT        ;SAVE THE B STATUS CODES
3161  011742  104400  015123          TYPCODE:        TYPE ,MASTAT            ;TYPE THE CONTENTS OF THE TWO STATUS REGISTERS
3162  011746  013746  011670                  MOV     ASTAT,-(SP)     ;;SAVE ASTAT FOR TYPEOUT
3163  011752  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3164  011754     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
3165  011755     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
3166  011756  104400  015016                  TYPE ,TAB
3167  011762  104400  015137                  TYPE ,MBSTAT
3168  011766  013746  011672                  MOV BSTAT,-(SP)
3169  011772  104403                          TYPON
3170  011774  104400  015030                  TYPE ,MCRLF
3171  012000  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.
                                ;***************************************************************

012002  012704  016300          GETPATTERN:     MOV #BUFADR,R4          ;SET ADDRESS OF FIRST DATA BYTE
012006  005037  012260                          CLR SUM                 ;SET UP FOR ACCUMULATION OF CHECK SUM
012012  013705  012046                          MOV PAT,R5              ;GET PATTERN BITS
012016  006305                                  ASL R5
012020  000175  012024                          JMP @PATTERNS(R5)
012024  012050                  PATTERNS:       DATA0                   ;000 DATA BYTE
012026  012062                                  DATA1                   ;377 DATA BYTE
012030  012072                                  FLOAT0                  ;FLOAT A 0 THROUGH ALL 1'S
012032  012134                                  FLOAT1                  ;FLOAT A 1 THROUGH ALL 0'S
012034  012142                                  PAT125                  ;125/052 DATA WORD
012036  012162                                  PAT314                  ;314/063 DATA WORD
012040  012172                                  COUNT                   ;INCRUMENT DATA PATTERN
012042  012212                                  RANDATA                 ;RANDOM DATA BYTE

012044  000000                  DATABYTE:       0
012046  000000                  PAT:            0

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

                                        ;LOAD SOFTWARE BUFFER WITH ALL ZEROS
                                        ;   PAT = 0

012050  005037  012044          DATA0:          CLR DATABYTE
012054  004737  012232          PATGEN:         JSR PC,LOAD             ;GO LOAD THE DATA BUFFER
012060  000775                                  BR PATGEN

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

                                        ;LOAD SOFTWARE BUFFER WITH ALL ONES
                                        ;   PAT = 1

012062  112737  000377  012044  DATA1:          MOVB #377,DATABYTE
012070  000771                                  BR PATGEN
```

```
                                        ;FLOAT A 0 THROUGH ONES IN SOFTWARE BUFFER
                                        ;   PAT = 2
                                        
012072  112737  000376  012044  FLOAT0:       MOVB #376,DATABYTE      ;SET UP A ONES FIELD
012100  000261                  XPATGEN:      SEC                     ;SET THE C BIT TO ROTATE THROUGH THE DATA
012102  012702  000000          1$:           MOV #0,R2               ;CLR R2 (CAN'T USE "CLR" IT CLEARS "C" BIT)
012106  103001                                BCC 2$                  ;BR IF "C" BIT IS CLEARED
012110  005202                                INC R2                  ;SET R2 IF "C" BIT IS SET
012112  004737  012232          2$:           JSR PC,LOAD             ;GO LOAD THE DATA BUFFER
012116  000241                                CLC                     ;CLEAR THE "C" BIT
012120  005702                                TST R2                  ;IS R2 NONZERO
012122  001401                                BEQ 3$
012124  000261                                SEC                     ;YES, SET THE "C" BIT
012126  106137  012044          3$:           ROLB DATABYTE
012132  000763                                BR 1$
                                        
                                        ;******************************************************************
                                        
                                        ;FLOAT A 1 THROUGH ALL ZEROS IN SOFTWARE BUFFER
                                        ;   PAT = 3
                                        
012134  005037  012044          FLOAT1:       CLR DATABYTE
012140  000757                                BR XPATGEN
                                        
                                        ;******************************************************************
                                        
                                        ;LOAD SOFTWARE BUFFER WITH ALTERNATING 1 AND 0 FOR
                                        ;ONE BYTE AND THE COMPLIMENT INTO THE NEXT
                                        ;   PAT = 4
                                        
012142  112737  000125  012044  PAT125:       MOVB #125,DATABYTE
012150  004737  012232          XXPATGEN:     JSR PC,LOAD
012154  105137  012044                        COMB DATABYTE
012160  000773                                BR XXPATGEN
                                        
                                        ;******************************************************************
                                        
                                        ;LOAD SOFTWARE BUFFER WITH ALTERNATING PAIRS OF 1 AND 0 AND
                                        ;COMPLIMENT INTO THE NEXT
                                        ;   PAT = 5
                                        
012162  112737  000314  012044  PAT314:       MOVB #314,DATABYTE
012170  000767                                BR XXPATGEN
                                        
                                        ;******************************************************************
                                        
                                        ;LOAD SOFTWARE BUFFER WITH COUNT PATTERN
                                        ;   PAT = 6
                                        
012172  012737  000377  012044  COUNT:       MOV #377,DATABYTE
012200  005237  012044          1$:           INC DATABYTE
012204  004737  012232                        JSR PC,LOAD
012210  000773                                BR 1$
```

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

                                        ;LOAD SOFTWARE BUFFER WITH RANDOM DATA PATTERN
                                        ;   PAT = 7

012212  004737  012262          RANDATA:      JSR PC,RANGEN           ;GET RANDOM NUMBER
012216  113737  012354  012044                MOVB RANUM,DATABYTE
012224  004737  012232                        JSR PC,LOAD
012230  000770                                BR RANDATA

012232  063737  012044  012260  LOAD:         ADD DATABYTE,SUM        ;ACCUMULATE THE PATTERN CHECK SUM
012240  113724  012044                        MOVB DATABYTE,(R4)+     ;LOAD THE DATA BUFFER
012244  022704  016500                        CMP #BUFADR+200,R4      ;HAVE 128 BYTES BEEN GENERATED
012250  001401                                BEQ 1$                  ;IF YES,RETURN TO TEST
012252  000207                                RTS PC                  ;IF NO,RETURN TO PATTERN GENERATOR
012254  005726                  1$:           TST (SP)+               ;TAKE PATTERN RETURN ADDRESS OF STACK
012256  000207                                RTS PC                  ;RETURN TO TEST

012260  000000                  SUM:          0

012262  012700  000001          RANGEN:       MOV #1,R0
012266  063700  012350                        ADD RAN1,R0
012272  063700  012352                        ADD RAN2,R0
012276  042700  170000                        BIC #170000,R0
012302  000241                                CLC
012304  006100                                ROL R0
012306  006100                                ROL R0
012310  010037  012350                        MOV R0,RAN1
012314  005000                                CLR R0
012316  013700  012352                        MOV RAN2,R0
012322  006000                                ROR R0
012324  006000                                ROR R0
012326  063700  012350                        ADD RAN1,R0
012332  042700  170000                        BIC #170000,R0
012336  010037  012352                        MOV R0,RAN2
012342  010037  012354                        MOV R0,RANUM
012346  000207                                RTS PC

012350  001234          RAN1:         001234
012352  000765          RAN2:         000765
012354  000000          RANUM:        0
```

```
                                        .SBTTL  UNIT SELECTION

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

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

012356  032737  000100  012446  GETUNIT:      BIT #BIT6,UNITSEL       ;WAS UNIT 0 JUST USED
012364  001012                                BNE 1$                  ;UNIT 0 USED CHECK UNIT 1
012366  105737  012446                        TSTB UNITSEL            ;WAS UNIT 0 SELECTED
012372  100007                                BPL 1$                  ;NO GO TO UNIT 1
012374  042737  040020  012446                BIC #40020,UNITSEL      ;CLEAR UNIT 1 USED BIT AND FUNCTION UNIT BIT
012402  052737  000100  012446                BIS #BIT6,UNITSEL       ;SET UNIT 0 USED BIT
012410  000207                                RTS PC
012412  005737  012446          1$:           TST UNITSEL             ;WAS UNIT 1 SELECTED
012416  100012                                BPL 2$                  ;NO RETURN
012420  032737  040000  012446                BIT #BIT14,UNITSEL      ;HAS UNIT 1 BEEN USED
012426  001006                                BNE 2$                  ;YES RETURN
012430  042737  000100  012446                BIC #BIT6,UNITSEL       ;CLEAR UNIT 0 USED BIT
012436  052737  040020  012446                BIS #40020,UNITSEL      ;SET UNIT 1 USED BIT AND FUNCTION UNIT BIT
012444  000207                          2$:   RTS PC


012446  000000                  UNITSEL:      0

                                        ;TEST THAT ALL UNITS HAVE BEEN ACCESSED

012450  005737  012446          DONE:         TST UNITSEL             ;IS UNIT 1 SELECTED
012454  100006                                BPL 1$                  ;NO RETURN
012456  032737  040000  012446                BIT #BIT14,UNITSEL      ;YES HAS IT BEEN USED
012464  001002                                BNE 1$                  ;YES RETURN
012466  062716  000002                        ADD #2,(SP)             ;BYPASS NOT DONE RETURE ON STACK
012472  000207                          1$:   RTS PC
```

```
                                        .SBTTL  TRACK SEQUENCE SELECTION

                                        ;INITIALIZE TRACK SEQUENCE

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


012474  105737  012660        INITTRACK:      TSTB SEQUEN             ;IS THIS TEST 26 SPECIAL SEQUENCE
012500  100442                                BMI 2$                  ;YES, DEC FROM TRACK 12 TO 0
012502  042737  100200  001200                BIC #100200,OD          ;CLEAR FIRST USED BITS
012510  005737  001200                        TST OD                  ;TEST CONTENTS OF ID,OD FOR 0
012514  001440                                BEQ 3$                  ;SEQUENCE WILL BE FROM "HOME"-52-53-114-0
012516  052737  100000  012660                BIS #BIT15,SEQUEN       ;LIMITS WERE SELECTED, INC FROM OD TO ID.
012524  113737  001200  012650                MOVB OD,TARGET          ;INIT OD AS PRESENT TRACK
012532  005037  012656                        CLR XID                 ;INIT WORKING ID AND OD LOCATIONS
012536  113737  001201  012656                MOVB ID,XID
012544  005037  012654                        CLR XOD
012550  113737  001200  012654                MOVB OD,XOD
012556  013737  012656  012646                MOV XID,TRKCNTR         ;SET UP NUMBER OF TRACK MOVEMENTS
012564  163737  012654  012646                SUB XOD,TRKCNTR
012572  005237  012646                        INC TRKCNTR
012576  052737  100200  001200  1$:           BIS #100200,OD          ;SET FIRST TIME BITS IN ID,OD
012604  000207                                RTS PC
012606  012737  000013  012646  2$:           MOV #13,TRKCNTR         ;SET TRACK COUNTER
012614  000770                                BR 1$
012616  012737  000004  012646  3$:           MOV #4,TRKCNTR          ;SET THE TRACK COUNTER
012624  000764                                BR 1$


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

012626  113737  012650  012652  GETTRACK:       MOVB TARGET,PRESTRK     ;RESET TO PRESENT TRACK
012634  005737  012660                        TST SEQUEN              ;IS THIS THE LIMITED SEQUENCE
012640  001410                                BEQ LIMTRK              ;YES, DOING ONLY 0-52-53-114
012642  100446                                BMI SEQ1                ;NO,SEQUENCE IS BETWEEN SELECTED LIMITS
012644  000463                                BR SEQ2                 ;NO,THIS IS TEST 26 DEC SEQUENCE

012646  000000                TRKCNTR:        0
012650  000000                TARGET:         0
012652  000000                PRESTRK:        0
012654  000000                XOD:            0
012656  000000                XID:            0
012660  000000                SEQUEN:         0
```

```
3405                                    ;*****************************************************************
3406
3407                                            ;LIMITED SEQUENCE, ACCESS TRACKS 52 TO 53 TO 114 BACK TO 0
3408  012662  005737  001200          LIMTRK:       TST 0D                  ;TEST HIGH ORDER FIRST TIME BIT
3409  012666  100007                               BPL 1$                  ;NOT SET, ON TRACK 52
3410  012670  012737  000052  012650               MOV #52,TARGET          ;GO TO TRACK 52
3411  012676  042737  100000  001200               BIC #BIT15,0D           ;CLEAR FIRST TIME BIT
3412  012704  000207                               RTS PC
3413  012706  105737  001200          1$:          TSTB 0D                 ;TEST LOW ORDER FIRST TIME BIT
3414  012712  100007                               BPL 2$                  ;NOT SET, ON TRACK 53
3415  012714  012737  000053  012650               MOV #53,TARGET          ;GO TO TRACK 53
3416  012722  042737  000200  001200               BIC #BIT7,0D
3417  012730  000207                               RTS PC
3418  012732  023727  012650  000114  2$:          CMP TARGET,#114         ;IS IT ON TRACK 114
3419  012740  001404                               BEQ 3$                  ;YES,GO TO TRACK 0
3420  012742  012737  000114  012650               MOV #114,TARGET         ;NO, GO TO TRACK 114
3421  012750  000207                               RTS PC
3422  012752  005037  012650          3$:          CLR TARGET              ;GO TO TRACK 0
3423  012756  000207                               RTS PC
3424
3425                                    ;*****************************************************************
3426
3427                                            ;INCREMENT FROM 0D+1 TO ID AND RETURN TO 0D
3428                                            ; USED WHEN TRACK LIMITS ARE SELECTED
3429
3430  012760  042737  100200  001200  SEQ1:       BIC #100200,0D          ;CLEAR FIRST TIME BITS
3431  012766  123737  012656  012652               CMPB XID,PRESTRK        ;PRESENT TRACK EQUAL TO ID
3432  012774  001004                               BNE 1$                  ;NO GET NEW TRACK
3433  012776  113737  001200  012650               MOVB 0D,TARGET          ;YES RETURN TO 0D
3434  013004  000207                               RTS PC
3435  013006  005237  012650          1$:          INC TARGET              ;ADD 1 TO TARGET TRACK
3436  013012  000207                               RTS PC
3437
3438                                    ;*****************************************************************
3439
3440                                            ;DECREMENT FROM ID = 12 TO 0D = 0
3441                                            ;USED IN TEST 26 ONLY
3442
3443  013014  005737  001200          SEQ2:        TST 0D                  ;FIRST TIME BIT SET
3444  013020  100007                               BPL 1$                  ;NO GET NEXT TRACK
3445  013022  042737  100200  001200               BIC #100200,0D          ;YES CLEAR FIRST TIME BITS
3446  013030  012737  000012  012650               MOV #12,TARGET          ;MOVE OUT 10 TRACKS
3447  013036  000207                               RTS PC
3448  013040  005337  012650          1$:          DEC TARGET              ;MOVE TO NEXT TRACK
3449  013044  000207                               RTS PC
```

```
                                        .SBTTL  SECTOR SELECTION

                                        ;SECTOR INITIALIZATION AND SELECTION

013046  005737  001202        INITSECTOR:     TST FIRST               ;TEST FIRST AND LAST FOR 0
013052  001005                                BNE 1$                  ;SECTORS SPECIFIED USE THEM
013054  005237  001202                        INC FIRST               ;NONE SPECIFIED SET FIRST TO 1
013060  112737  000032  001203                MOVB #32,LAST           ;SET LAST TO MAXIMUM
013066  113737  001203  013136  1$:           MOVB LAST,SECCNTR       ;SET UP SECTOR COUNTER
013074  163737  001202  013136                SUB FIRST,SECCNTR
013102  005237  013136                        INC SECCNTR
013106  105037  013137                        CLRB SECCNTR+1
013112  113737  001202  013140                MOVB FIRST,TSECTOR      ;PUT FIRST SECTOR IN TARGET SECTOR
013120  162737  000003  013140                SUB #3,TSECTOR          ;SUB 3 FROM TSECTOR AS FIRST TIME THROUGH
                                                                      ;IT GETS ADDED BACK ON.
013126  012737  000001  013142                MOV #1,INTLEAV          ;SET INTERLEAVE OFFSET
013134  000207                                RTS PC

013136  000000                SECCNTR:        0
013140  000000                TSECTOR:        0
013142  000000                INTLEAV:        0

013144  042737  000200  013140  GETSECTOR:    BIC #200,TSECTOR        ;CLEAR THE UNIT BIT BEFORE TESTING
013152  062737  000003  013140                ADD #3,TSECTOR          ;ADD 3 FOR INTERLEAVING
013160  123737  001203  013140                CMPB LAST,TSECTOR
013166  002010                                BGE 1$                  ;NEW SECTOR IS WITHIN LIMITS
013170  113737  001202  013140                MOVB FIRST,TSECTOR      ;RESET TARGET SECTOR TO INTERLEAVE
013176  063737  013142  013140                ADD INTLEAV,TSECTOR     ;ADD ON INTERLEAVE OFFSET VALUE
013204  005237  013142                        INC INTLEAV             ;UP DATE THE OFFSET VALUE
013210  032737  000020  012446  1$:           BIT #BIT4,UNITSEL       ;IS THIS UNIT 0
013216  001403                                BEQ 2$
013220  052737  000200  013140                BIS #BIT7,TSECTOR       ;NO, SET UNIT IDENTIFIER IN TARGET SECTOR
013226  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
                                ;*

013230  105737  013453          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
013234  100002                          BPL     1$              ;;BR IF YES
013236  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
013240  000407                          BR      3$              ;;LEAVE
013242  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
013244  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
013250  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
013252  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
013254  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
013256  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
013260  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
013264  000002                          RTI                     ;;RETURN
013266  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
013272  001426                          BEQ     8$
013274  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
013300  001004                          BNE     5$
013302  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
013304  104400                          TYPE                    ;;TYPE A CR AND LF
013306  013455                          $CRLF
013310  000757                          BR      2$              ;;GET NEXT CHARACTER
013312  004737  013374          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
013316  123726  013452          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
013322  001352                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
013324  013746  013450                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
013330  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
013334  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
013336  004737  013374                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
013342  105337  013440                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
013346  000770                          BR      7$              ;;LOOP

                                ;HORIZONTAL TAB PROCESSOR

013350  112716  000040          8$:     MOVB    #40,(SP)        ;;REPLACE TAB WITH SPACE
013354  004737  013374          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
013360  132737  000007  013440          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
```

```
 3537  013366  001372                        BNE     9$              ;;TAB STOP
 3538  013370  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
 3539  013372  000726                        BR      2$              ;;GET NEXT CHARACTER
 3540  013374  105777  000044          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
 3541  013400  100375                        BPL     $TYPEC
 3542  013402  116677  000002  000036        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
 3543  013410  122766  000015  000002        CMPB    #15,2(SP)       ;;BRANCH IF
 3544  013416  001003                        BNE     1$              ;;NOT <CR>
 3545  013420  105037  013440                CLRB    $CHARCNT        ;;
 3546  013424  000406                        BR      $TYPEX          ;;EXIT
 3547  013426    2766  000012  000002  1$:   CMPB    #12,2(SP)       ;;BRANCH IF
 3548  013434  002002                        BGE     $TYPEX          ;;<LF>
 3549  013436  105227                        INCB    (PC)+           ;;INC SPACE
 3550  013440  000000              $CHARCNT:.WORD   0               ;;COUNT
 3551  013442  000207              $TYPEX: RTS     PC
 3552                               ;;      EQUATES
 3553          000011              THT=11
 3554          000200              TCRLF=200
 3555
 3556  013444  177564              $TPS:   .WORD   177564          ;;TTY PRINTER STATUS REG. ADDRESS
 3557  013446  177566              $TPB:   .WORD   177566          ;;TTY PRINTER BUFFER REG. ADDRESS
 3558  013450     000              $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
 3559  013451     002              $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
 3560  013452     012              $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
 3561  013453     000              $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
 3562  013454     077              $QUES:  .ASCII  "?"             ;;QUESTION MARK
 3563  013455     015      000     $CRLF:  .ASCIZ  <15>            ;;CARRAIGE RETURN
 3564  013457     012      000     $LF:    .ASCIZ  <12>            ;;LINEFEED
 3565          013462              .EVEN
```

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

                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

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

013462  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
013466  116637  000001  013705          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
013474  112637  013707                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
013500  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
013504  000406                          BR      $TYPON
013506  112737  000001  013705  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
013514  112737  000006  013707          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
013522  112737  000005  013704  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
013530  010346                          MOV     R3,-(SP)        ;;SAVE R3
013532  010446                          MOV     R4,-(SP)        ;;SAVE R4
013534  010546                          MOV     R5,-(SP)        ;;SAVE R5
013536  113704  013707                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
013542  005404                          NEG     R4
013544  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
013550  110437  013706                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
013554  113704  013705                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
013560  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
013564  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
013566  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
013570  000404                          BR      3$              ;;GO DO MSB
013572  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
013574  006105                          ROL     R5
013576  006105                          ROL     R5
013600  010503                          MOV     R5,R3
013602  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
013604  105337  013706                  DECB    $OMODE          ;;TYPE THIS DIGIT?
013610  100016                          BPL     7$              ;;BR IF NO
013612  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
```

```
 3620  013616  001002                        BNE     4$              ;;TEST FOR 0
 3621  013620  005704                        TST     R4              ;;SUPPRESS THIS 0?
 3622  013622  001403                        BEQ     5$              ;;BR IF YES
 3623  013624  005204                4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
 3624  013626  052703  000060              BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
 3625  013632  052703  000040        5$:   BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
 3626  013636  110337  013702              MOVB    R3,8$           ;;SAVE FOR TYPING
 3627  013642  104400  013702              TYPE    ,8$             ;;GO TYPE THIS DIGIT
 3628  013646  105337  013704        7$:   DECB    $OCNT           ;;COUNT BY 1
 3629  013652  003347                        BGT     2$              ;;BR IF MORE TO DO
 3630  013654  002402                        BLT     6$              ;;BR IF DONE
 3631  013656  005204                        INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
 3632  013660  000744                        BR      2$              ;;GO DO THE LAST DIGIT
 3633  013662  012605                6$:     MOV     (SP)+,R5        ;;RESTORE R5
 3634  013664  012604                        MOV     (SP)+,R4        ;;RESTORE R4
 3635  013666  012603                        MOV     (SP)+,R3        ;;RESTORE R3
 3636  013670  016666  000002  000004        MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
 3637  013676  012616                        MOV     (SP)+,(SP)
 3638  013700  000002                        RTI                     ;;RETURN
 3639  013702     000                8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
 3640  013703     000                        .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
 3641  013704     000                $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
 3642  013705     000                $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
 3643  013706  000000                $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

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

                                .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES

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

013710                          $SAVREG:
013710  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
013712  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
013714  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
013716  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
013720  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
013722  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
013724  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
013730  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
013734  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
013740  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
013744  000002                          RTI

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

013746                          $RESREG:
013746  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
013752  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
013756  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
013762  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
013766  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
013770  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
013772  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
013774  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
013776  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
014000  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
014002  000002                          RTI
```

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

                                .SBTTL  TRAP DECODER

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

014004  010046          $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
014006  016600  000002          MOV     2(SP),R0        ;;GET TRAP ADDRESS
014012  005740                  TST     -(R0)           ;;BACKUP BY 2
014014  111000                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
014016  006300                  ASL     R0              ;;POSITION FOR INDEXING
014020  016000  014026          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
014024  000200                  RTS     R0              ;;GO TO ROUTINE


                                .SBTTL  TRAP TABLE

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

                                ;       ROUTINE
                                ;       -------
014026                  $TRPAD:
014026  013230                  $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
014030  013506                  $TYPOC  ;;CALL=TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
014032  013462                  $TYPOS  ;;CALL=TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
014034  013522                  $TYPON  ;;CALL=TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
014036  013710                  $SAVREG ;;CALL=SAVREG   TRAP+4(104404)  SAVE R0-R5 ROUTINE
014040  013746                  $RESREG ;;CALL=RESREG   TRAP+5(104405)  RESTORE R0-R5 ROUTINE
014042  006336                  XSUBSCOPE       ;;CALL=SUBSCOPE TRAP+6(104406)
```

```
                                        ;*****************************************************************
                                        
                                        .SBTTL  POWER DOWN AND UP ROUTINES
                                        
                                        ;POWER DOWN ROUTINE
014044  012737  014172  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
014052  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
014060  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
014062  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
014064  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
014066  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
014070  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
014072  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
014074  010637  014176                  MOV     SP,$SAVR6       ;;SAVE SP
014100  012737  014112  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
014106  000000                          HALT
014110  000776                          BR      .-2             ;;HANG UP
                                        
                                        ;POWER UP ROUTINE
014112  013706  014176          $PWRUP: MOV     $SAVR6,SP       ;;GET SP
014116  005037  014176                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
014122  005237  014176          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
014126  001375                          BNE     1$              ;;OF  WORD
014130  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
014132  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
014134  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
014136  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
014140  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
014142  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
014144  012737  014044  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
014152  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
014160  104400                          TYPE                    ;REPORT THE POWER FAILURE
014162  014200                  $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
014164  012716                          MOV     (PC)+,(SP)      ;;RESTART AT RESTART
014166  001306                  $PWRAD: .WORD   RESTART         ;;RESTART ADDRESS
014170  000002                          RTI
014172  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
014174  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
014176  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
014200  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
014206  000122
                                        .EVEN
```

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

                                        .SBTTL  SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE

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

014210  016637  000002  014234  $$B2D:  MOV     2(SP),1$        ;;SAVE BINARY NUMBER
014216  012746  014234                  MOV     #1$,-(SP)       ;;SET POINTER
014222  004737  014240                  JSR     PC,@#$D82D      ;;CALL DOUBLE LENGTH CONVERT
014226  012666  000002                  MOV     (SP)+,2(SP)     ;;PICKUP POINTER
014232  000207                          RTS     PC              ;;RETURN
014234  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,@#$D82D
                                        ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
                                                                        ;;IS ON THE STACK


014240  104404                  $D82D:  SAVREG                  ;;SAVE REGISTERS
014242  016602  000002                  MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
014246  012700  014420                  MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
014252  010066  000002                  MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
014256  012201                          MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
014260  012202                          MOV     (R2)+,R2
014262  012737  000012  014336          MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
014270  012704  014350                  MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
014274  012705  014352                  MOV     #$TNPWR+2,R5
014300  005003                  1$:     CLR     R3              ;;CLEAR PARTIAL
014302  161401                  2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
014304  005602                          SBC     R2
014306  161502                          SUB     (R5),R2
014310  002402                          BLT     3$              ;;BR IF TEN POWER TO LARGE
014312  005203                          INC     R3              ;;ADD 1 TO PARTIAL
014314  000772                          BR      2$              ;;LOOP
014316  062401                  3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
014320  005502                          ADC     R2
014322  062402                          ADD     (R4)+,R2
014324  022525                          CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
014326  052703  000060                  BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
014332  110320                          MOVB    R3,(R0)+        ;;SAVE IT
```

```
3819  014334  005327                      DEC     (PC)+           ;;DONE?
3820  014336  000000              4$:     .WORD   0
3821  014340  001357                      BNE     1$              ;;BR IF NO
3822  014342  105020                      CLRB    (R0)+           ;;TERMINATOR
3823  014344  104405                      RESREG                  ;;RESTORE REGISTERS
3824  014346  000207                      RTS     PC              ;;RETURN
3825  014350  145000              $TNPWR: 145000                  ;;1.0E09
3826  014352  035632                      35632
3827  014354  160400                      160400                  ;;1.0E08
3828  014356  002765                      2765
3829  014360  113200                      113200                  ;;1.0E07
3830  014362  000230                      230
3831  014364  041100                      041100                  ;;1.0E06
3832  014366  000017                      17
3833  014370  103240                      103240                  ;;1.0E05
3834  014372  000001                      1
3835  014374  023420                      23420                   ;;1.0E04
3836  014376  000000                      0
3837  014400  001750                      1750                    ;;1.0E03
3838  014402  000000                      0
3839  014404  000144                      144                     ;;1.0E02
3840  014406  000000                      0
3841  014410  000012                      12                      ;;1.0E01
3842  014412  000000                      0
3843  014414  000001                      1                       ;;1.0E00
3844  014416  000000                      0
3845  014420  000014              $DECVL: .BLKB   12.             ;;RESERVE STORAGE FOR ASCIZ STRING
3846
```

```
3847                                    ;*****************************************************************
3848
3849                                            ;TYPE NUMERICAL ASCIZ STRING,RIGHT JUSTIFIED
3850                                            ;REPLACING LEADING ZEROS WITH SPACES.
3851                                            ;
3852                                            ;FIRST ADDRESS OF ASCIZ STRING MUST BE ON TOP OF THE STACK
3853
3854  014434  010046                RTJUST:         MOV R0,-(SP)            ;SAVE R0
3855  014436  016600  000004                       MOV 4(SP),R0            ;PICK UP ADDRESS OF ASCIZ STRING
3856  014442  010037  014474                       MOV R0,3$               ;SAVE ADDRESS FOR TYPE OUT
3857  014446  105710                1$:             TSTB (R0)               ;IS THIS THE TERMINATOR
3858  014450  001406                                BEQ 2$                  ;IF YES TYPE IT OUT
3859  014452  122710  000060                       CMPB #'0,(R0)           ;IS IT A ZERO
3860  014456  001005                                BNE 4$                  ;IF NO GO PRINT IT
3861  014460  112720  000040                       MOVB #' ,(R0)+          ;IF YES REPLACE IT WITH A SPACE
3862  014464  000770                                BR 1$                   ;TEST NEXT CHAR.
3863  014466  112740  000060        2$:             MOVB #'0,-(R0)          ;STRING OFF ALL ZEROS,PUT BACK THE LAST ONE
3864  014472  104400                4$:             TYPE                    ;TYPE THE STRING
3865  014474  000000                3$:             OPEN
3866  014476  012600                                MOV (SP)+,R0            ;RESTORE R0
3867  014500  012616                                MOV (SP)+,(SP)          ;RESTORE THE STACK
3868  014502  000207                                RTS PC                  ;RETURN
3869
3870                                          ;TYPES 16 BIT WORD IN DECIMAL
3871
3872  014504  012546                SGLDEC:         MOV (R5)+,-(SP)         ;PUT NUMBER TO BE TYPED ON STACK
3873  014506  004737  014210                       JSR PC,@#$SB2D          ;CONVERT NUMBER TO DECIMAL
3874  014512  062716  000005                       ADD #5,(SP)             ;MOVE ADDRESS OF ASCIZ STRING OVER BY 5 BYTES
3875                                                                        ;TO TYPE SINGLE DECIMAL NUMBER
3876  014516  004737  014434                       JSR PC,RTJUST           ;TYPE THE DECIMAL NUMBER
3877  014522  000205                                RTS R5
```

F14


```
                                                .SBTTL  MESSAGES

                                                MDTESTP:        .ASCIZ "TEST PARAMETERS: "
                                                MXEHEADER:      .ASCIZ <15><12><12> "ERROR  FAST   FAPT          GOOD   BAD     PASS"
                                                MUNXDD:         .ASCIZ "UNEXPECTED D D MARK"<15><12>
                                                MDDMIS:         .ASCIZ "D D MARK MISSING"<15><12>
                                                MDERHDR:        .ASCIZ "DATA, NO STATUS ERROR"
                                                MTRK:           .ASCIZ " ON TRACK"
                                                MPRES:          .ASCIZ "  FROM TRACK"
                                                MSECT:          .ASCIZ " / SECTOR"
                                                MCOLMUN:        .ASCIZ <15><12>" BYTE  BAD  GOOD"<15><12>
                                                TAB:            .ASCIZ <40><40><40><40><40><40>
                                                DBLSP:          .ASCIZ <40><40>
                                                MCRLF:          .ASCIZ <15><12>
```

```
3932  015033    015  042412  051122  MERHEADER:      .ASCIZ <15><12>"ERROR CONDITIONS:  TEST PC = "
3933  015040  051117  041440  047117
3934  015046  044504  044524  047117
3935  015054  035123  020040  042524
3936  015062  052123  050040  020103
3937  015070  020075     000
3938
3939  015073    125  044516  020124  MUNIT0:         .ASCIZ "UNIT 0 "
3940  015100  020060     000
3941
3942  015103    125  044516  020124  MUNIT1:         .ASCIZ "UNIT 1 "
3943  015110  020061     000
3944
3945  015113    122  041530  020123  MRXCS:          .ASCIZ "RXCS = "
3946  015120  020075     000
3947
3948  015123    123  040524  052524  MASTAT:         .ASCIZ "STATUS A = "
3949  015130  020123  020101  020075
3950  015136     000
3951
3952  015137    123  040524  052524  MBSTAT:         .ASCIZ "STATUS B = "
3953  015144  020123  020102  020075
3954  015152     000
3955
3956  015153    015  005012     000  DBLLF:          .ASCIZ <15><12><12>
3957
3958  015157    116  020117  047111  MINTER:         .ASCIZ "NO INTERRUPT AT DONE ERROR"
3959  015164  042524  051122  050125
3960  015172  020124  052101  042040
3961  015200  047117  020105  051105
3962  015206  047522  000122
3963
3964  015212  047125  047113  053517  MUKNINT:        .ASCIZ "UNKNOWN INTERRUPT"
3965  015220  020116  047111  042524
3966  015226  051122  050125  000124
3967
3968  015234  047524  040524  020114  MERCT:          .ASCIZ "TOTAL READ CHECK ERRORS = "
3969  015242  042522  042101  041440
3970  015250  042510  045503  042440
3971  015256  051122  051117  020123
3972  015264  020075     000
3973
3974  015267    106  046111  041114  MFIL:           .ASCIZ "FILLBUFFER "
3975  015274  043125  042506  020122
3976  015302     000
3977
3978  015303    105  050115  054524  MEMPTY:         .ASCIZ "EMPTYBUFFER "
3979  015310  052502  043106  051105
3980  015316  000040
3981
3982  015320  052040  040522  045503  MLIMTRK:        .ASCIZ " TRACKS 52,53,114,0  "
3983  015326  020123  031065  032454
3984  015334  026063  030461  026064
3985  015342  020060  000040
```

```
                                
015346  042117  000075          MOD:            .ASCIZ "OD="
                                
015352  020040  042111  000075  MID:            .ASCIZ "  ID="
                                
015360  020040  044506  051522  MFIRST:         .ASCIZ "  FIRST:"
015366  036524     000
                                
015371     040  046040  051501  MLAST:          .ASCIZ "  LAST:"
015376  036524     000
                                
015401     103  041522  042440  MBADCRC:        .ASCIZ "CRC ERROR NO DATA ERROR"
015406  051122  051117  047040
015414  020117  040504  040524
015422  042440  051122  051117
015430     000
                                
015431     122  040505  020104  MREAD:          .ASCIZ "READ "
015436     000
                                
015437     104  052101  020101  MCRC:           .ASCIZ "DATA CRC ERROR"
015444  051103  020103  051105
015452  047522  000122
                                
015456  042523  045505  042440  MSEEK:          .ASCIZ "SEEK ERROR"
015464  051122  051117     000
                                
015471     127  044522  042524  MWRITE:         .ASCIZ "WRITE "
015476  000040
                                
015500  040520  044522  054524  MPAR:           .ASCIZ "PARITY ERROR"
015506  042440  051122  051117
015514     000
                                
015515     105  051122  051117  MNOFLAG:        .ASCIZ "ERROR FLAG ERROR"
015522  043040  040514  020107
015530  051105  047522  000122
                                
015536  040502  000104          MBAD:           .ASCIZ "BAD"
                                
015542  000040                  SPACE:          .ASCIZ <40>
                                
015544  047507  042117     000  MGOOD:          .ASCIZ "GOOD"
                                
015551     040  041440  042510  MSUM:           .ASCIZ "  CHECK SUM "
015556  045503  051440  046525
015564  000040
                                
015566  005015  054122  030461  MRX11:          .ASCIZ <15><12>"RX11 / RXV11"
015574  027440  051040  053130
015602  030461     000
                                
015605     015  005012  040515  MREV:           .ASCIZ <15><12><12> "MAINDEC-11-DZRXB-D"
015612  047111  042504  026503
```

```
  4040  015620  030461  042055  051132
  4041  015626  041130  042055     000
  4042
  4043  015633     015  052412  042516  LOC4M:  .ASCIZ  <15><12>"UNEXPECTED TRAP TO LOC. 4 OCCURRED"
  4044  015640  050130  041505  042524
  4045  015646  020104  051124  050101
  4046  015654  052040  020117  047514
  4047  015662  027103  032040  047440
  4048  015670  041503  051125  042522
  4049  015676  000104
  4050
  4051  015700  005015  047125  054105  LOC10M: .ASCIZ  <15><12>"UNEXPECTED TRAP TO LOC. 10 OCCURRED"
  4052  015706  042520  052103  042105
  4053  015714  052040  040522  020120
  4054  015722  047524  046040  041517
  4055  015730  020056  030061  047440
  4056  015736  041503  051125  042522
  4057  015744  000104
  4058
  4059  015746  050075  000103          PCM:    .ASCIZ  "=PC"
  4060
  4061  015752  005015  051124  041501  OD2BIG: .ASCII  <15><12>"TRACK LIMITS SELECTED OUT OF RANGE"
  4062  015760  020113  044514  044515
  4063  015766  051524  051440  046105
  4064  015774  041505  042524  020104
  4065  016002  052517  020124  073117
  4066  016010  051040  047101  042507
  4067  016016  005015  042504  040506          .ASCIZ  <15><12>"DEFAULTING TO "
  4068  016024  046125  044524  043516
  4069  016032  052040  020117     000
  4070
  4071  016037     015  051412  041505  S2BIG:  .ASCII  <15><12>"SECTOR LIMITS SELECTED OUT OF RANGE"
  4072  016044  047524  020122  044514
  4073  016052  044515  051524  051440
  4074  016060  046105  041505  042524
  4075  016066  020104  052517  020124
  4076  016074  043117  051040  047101
  4077  016102  042507
  4078  016104  005015  042504  040506          .ASCIZ  <15><12>"DEFAULTING TO "
  4079  016112  046125  044524  043516
  4080  016120  052040  020117     000
  4081
  4082  016125     015  041412  052501  DOLOAD: .ASCII  <15><12>"CAUTION - IF YOU DESIRE TO TEST UNIT 0"
  4083  016132  044524  047117  026440
  4084  016140  044440  020106  047531
  4085  016146  020125  042504  044523
  4086  016154  042522  052040  020117
  4087  016162  042524  052123  052440
  4088  016170  044516  020124     060
  4089  016175     015  051012  050105          .ASCII  <15><12>"REPLACE LOAD MEDIUM WITH A SCRATCH DISKETTE"
  4090  016202  040514  042503  046040
  4091  016210  040517  020104  042515
  4092  016216  044504  046525  053440
  4093  016224  052111  020110  020101
```

```
4094  016232  041523  040522  041524
4095  016240  020110  044504  045523
4096  016246  052105  042524
4097  016252  005015  044124  047105          .ASCIZ  <15><12>"THEN PRESS CONTINUE"
4098  016260  050040  042522  051523
4099  016266  041440  047117  044524
4100  016274  052516  000105
4101
4102                                          .EVEN
4103
4104                                  ;*****************************************************************
4105
4106                                          ;THE FOLLOWING LOCATIONS ARE USED FOR DATA STORAGE,RETRY COUNTERS
4107                                          ;ACCESS COUNTERS ETC.
4108
4109  016300  000200                  BUFADR:         .BLKB 200
4110
4111          000001                          .END
```

```
ADJSUM  010532           2260    2333    2351    2681    2945    2963#
ASTAT   011670           2736    2839    2846    2891    2907    3068*   3071    3074    3148#   3152*   3162
BADBYT  011210           2956*   2957    2958    3003    3048#
BIT0  = 000001            173#   2846
BIT00 = 000001            163#    173
BIT01 = 000002            162#    172
BIT02 = 000004            161#    171
BIT03 = 000010            160#    170
BIT04 = 000020            159#    169
BIT05 = 000040            158#    168
BIT06 = 000100            157#    167
BIT07 = 000200            156#    166
BIT08 = 000400            155#    165
BIT09 = 001000            154#    164
BIT1  = 000002            172#   2736    2839
BIT10 = 002000            153#
BIT11 = 004000            152#
BIT12 = 010000            151#
BIT13 = 020000            150#   2796
BIT14 = 040000            149#    430    1377    3339    3353
BIT15 = 100000            148#    443    2937    3373    3411
BIT2  = 000004            171#   3071
BIT3  = 000010            170#
BIT4  = 000020            169#   3140    3479
BIT5  = 000040            168#   1485    2571
BIT6  = 000100            167#    629     783    1028    1189    1233    1247    1287    1305    1347    1366    1434    1538
                         1777    1796    1895    2891    2907    3330    3335    3341
BIT7  = 000200            166#    632     633    1328    1437    1438    1542    2386    2392    2944    2998    3416    3481
BIT8  = 000400            165#    518     784     806    2466    2498    2897    2909    2983    3091
BIT9  = 001000            164#
BPTVEC= 000014            180#
BRLEV   001214            363#   2430
BSTAT   011672           3099*   3149#   3160*   3168
BUFADR  016300           1570    1673    2963*   2964*   2968*   2971*   2972    3183    3285    4109#
BUSERR  005756            208    2398#
BYTECN  011206           2684*   2692*   2942*   2960*   2999    3015*   3047#
CATERR  002660           1116    1121*   1124#   1136*
CCOUNT  002336            419*    787     790*    791     793*    817#   2465*
CEXIT   004074           1011    1249    1306    1367    1553    1563    1670    1756    1757#
CKBYTE  010502           2956#
CKSUM   010622           2946*   2957*   2965*   2966*   2967*   2968    2969*   2970*   2971    2975#   3030
COMMWO  007424           2271    2721    2776#   2818
CONT10  010270           2890    2907#
CONT11  010240           2894    2900*   2911
CONT13  007110           2707    2709#
CONT14  007354           2759    2761#
CONT15  010070           2866    2868#
CONT16  010136           2880    2882#
CONT17  010250           2901    2903#
CONT20  011036           3013    3015#
CONT21  011252           3062    3064#
CONT4   007100           2702    2706#
COUNT   012172           3194    3269#
CPUPRI  006026           1280    1339    2430#
```

L14


```
CRCER   010016           2847    2855#
DATABY  012044           3198#   3207#   3217#   3223#   3233#   3241#   3250#   3252#   3261#   3269#   3270#   3279#   3283
                         3284
DATACK  007474           2787#   2813#   2857#   3018
DATACR  010106           2856    2860    2875#
DATAER  010654           2959    2983#
DATAO   012050           3188    3207#
DATAI   012062           3189    3217#
DBLLF   015153           3956#
DBLSP   015025           3002    3006    3139    3928#
DDCHK   010162           2830    2889#
DDERR   010222           2897#   2914
DDISP = 177570             95#    371
DEATH   002662           1120    1130#
DEXIT   004106           1760#   1835    1969    2023    2075    2210    2247    2270    2338    2354    2367    2393
DING    006272           2439    2508#   2928
DISPLA  001220            371#    410*
DISPRE  000174            231#    410
DONE    012450           2674    3351#
DONEBI= 000040            195#   2725    2822    3076
DRVSWP  006614           2366    2648#
DSWR  = 177570             94#    370
DTESTP  001212            361#    426     430     441     447     807
DOLOAD  016125            435    4082#
EBIE  = 000103            197#   2952
EMPBUF  010410           2858    2939#   2979    3039    3043
EMPDON  011046           2947    3018#
EMPER   010624           2948    2977#
EMPFLA  010474           2953#   2954    2961    3016
EMTVEC= 000030            183#
EPCSCO  006314           2456*   2457*   2517#   2549
ERCNTR  011214           2859    2943*   2984*   2989    3021    3026    3050#
ERMSG   010332           2895    2912    2918#   2993    3058    3103
ERRORS  006312            519*   2458*   2516#   2522    2531*   2537    3089#
ERRVEC= 000004            176#
EXIT    011202           2941    3041    3044#
FAST    002342            521*    710*    819#   2478
FBEB    003712           1571    1580#   1586    1674    2263
FBIE  = 000101            196#   2688
FILLBU  006746           2681#   2698    2711    2745    2765
FILLDO  007130           2682    2714#   2741
FILLER  007024           2683    2695#
FILLFL  007004           2689#   2690    2693
FIRST   001202            294#    295     481     483*    486     489     501*    3454    3456*   3459    3462    3476
FIRSTT  002040            709#    812
FLOATO  012072           3190    3223#
FLOATI  012134           3191    3241#
FUNCTI  007472           2268*   2352*   2717*   2720*   2776*   2777    2786#   2817*   3105
GETPAT  012002           1569    2259    2332    2350    2648    2666    3183#
GETSEC  013144           2680    2812    3472#
GETTRA  012626           2637    2651    2669    3392#
GETUNI  012356           2257    2349    2636    2650    2654    2668    3330#
GNS   = ****** U         3716    3717    3718    3719    3720    3721    3722
GOODBY  011212           3007*   3009    3049#
```

```
HANGER  006560            421*   2573*   2624*   2628#   2714*   2727*   2814*   2824*
HANGPL  006562           2574*   2625*   2629#
HERE    002444            843     849#
HLT10   010066           2867#
HLT12   010134           2881#
HLT13   010246           2902#
HLT14   011034           3014#
HLT15   011250           3063#
HLT6    007106           2708#
HLT7    007352           2760#
HOME    007476           2761    2789#   2803
HUNGUP  006440           2575    2582#
ID    = 001201            293#    460     462     470    3376
ILLADR  004734           1852    1985    2038    2079#   2084
INCERR  006072           2458#   2459
INITSE  013046           2679    2811    3454#
INITTR  012474           2635    2649    2667    3368#
INTLEA  013142           3465*   3470#   3477    3478*
INTSER  011256           2244    2364    2389    3064    3068#
IOTVEC= 000020            181#
KRXVEC  001204            297#    964    1018    1182    1223    1265    1321    2243    2363    2388
LAST  = 001203            295#    484     486     492    3457*   3458    3474
LIMITS  001522            452#    477
LIMTRK  012662           3394    3408#
LOAD    012232           3208    3228    3251    3271    3280    3283#
LOCKUP  002366            707     831#
LOC10M  015700           2414    4051#
LOC4M   015633           2398    4043#
LOGICA  002434            224     845#
MABELL  006310            795    2510    2513#
MASTAT  015123           3161    3948#
MBAD    015536           3032    4024#
MEADCR  015401           2863    3997#
MBSTAT  015137           3167    3952#
MCOLMU  014771           2997    3920#
MCRC    015437           2877    4006#
MCRLF   015030            445     451     494     789     800    2470    2705    2773    2864    2878    2899    2927    2995
                         3011    3035    3060    3081    3088    3102    3110    3124    3170    3930#
MDDMIS  014657           2896    3899#
MDERHD  014702           2994    3905#
MDTEST  014524            446    3881#
MEMRTY  015303           2978    3978#
MERCT   015234           3025    3968#
MERHEA  015033           2918    3932#
MFIL    015267           2696    3974#
MFIRST  015360            488    3991#
MGOOD   015544           3034    4028#
MHUNGP  006476           2605    2615#
MID     015352            469    3989#
MILTST  002346            814     823#
MINTER  015157           3059    3958#
MLAST   015371            491    3994#
MLIMTR  015320            455    3982#
MNOFLA  015515           3087    4020#
```

```
MOD     015346            464    3987#
MORETE  002032            707#   1758    1761    2604    2613
MPAR    015500           2704    3123    4016#
MPASS   006527            801    2620#   2923
MPRES   014742           2768    3913#
MREAD   015431           2842    2850    4003#
MREV    015605            424    4038#
MRXCS   015113           3104    3118    3945#
MRX11   015566           3115    4034#
MSECT   014757           3132    3917#
MSEEK   015456           2767    4010#
MSJM    015551           3029    4030#
MTRK    014730           3127    3910#
MUKNIN  015212           3080    3964#
MUNITO  015073           3142    3939#
MUNITI  015103           3144    3942#
MUNXCD  014631           2913    3894#
MWRITE  015471           2739    2750    2756    4013#
MX      002344            786*    794     821#   1757*   1760*
MXEHEA  014546           2468    3885#
NEXTRD  007714           2712    2829    2832    2834#   2844    2852    2870    2885    2906
NOERTY  011026           2986    2990    3012#
NOINTE  011216           2729    2826    3056#
NOMORE  002144            719     783#
NOPRIN  006244           2464    2498#
NOPRNT  011452           3100#
NOSCOP  006324           2531#   2558
NXREAD  011164           2980    3022    3037    3040#
OD      001200            292#    293     444*    452     462     465     476*   3370*   3371    3374    3378    3382*   3408
                         3411*   3413    3416*   3430*   3433    3443    3445*
OO2BIG  015752            473    4061#
OK2GO   002664           1123    1135#
OPEN  = 000000            202#    804     2495    2703    2926    3001    3028    3146    3865
PARTES  007054           2699    2701#   2742    2845    2981
PARTYP  011544           3097    3118#   3158
PASS    002340            420*    796*    802     818#   2493    2924
PAT     012046           1561*   1748*   1753*   2258*   2329*   2331*   2346*   2348*   2362*   2375*   2387*   3185    3199#
PATGEN  012054           3208#   3209    3218
PATTER  012024           3187    3188#
PAT125  012142           3192    3250#
PAT314  012162           3193    3261#
PC    =%000007            107#    707*    840*    845*    849*   1280*   1339*   1378*   1494*   1562*   1569*   1571*   1572*
                         1580*   1583*   1585*   1654*   1669*   1674*   1675*   1754*   1755*   1772*   1791*   1852*   1908*
                         1985*   2013*   2038*   2065*   2079*   2083*   2087*   2091*   2095*   2100*   2103*   2115*   2118*
                         2124*   2127*   2133*   2188*   2246*   2257*   2259*   2260*   2263*   2269*   2271*   2272*   2278*
                         2317*   2318*   2330*   2332*   2333*   2335*   2337*   2347*   2349*   2350*   2351*   2353*   2366*
                         2391*   2444*   2499*   2511*   2567*   2576*   2627*   2635*   2636*   2637*   2638*   2641*   2648*
                         2649*   2650*   2651*   2652*   2653*   2654*   2655*   2656*   2659*   2666*   2667*   2668*   2669*
                         2670*   2671*   2674*   2675*   2679*   2680*   2681*   2721*   2732*   2751*   2757*   2761*   2764*
                         2774*   2778*   2781*   2784*   2792*   2798*   2804*   2806*   2811*   2812*   2818*   2830*   2833*
                         2836*   2851*   2858*   2895*   2898*   2912*   2915*   2928*   2929*   2938*   2945*   2973*   2993*
                         2996*   3020*   3038*   3042*   3045*   3058*   3064*   3084*   3093*   3097*   3103*   3109*   3111*
                         3113*   3125*   3145*   3154*   3158*   3171*   3208*   3228*   3251*   3271*   3278*   3280*   3287*
                         3289*   3309*   3336*   3343*   3356*   3383*   3412*   3417*   3421*   3423*   3434*   3436*   3447*
```

```
                    3449*  3466*  3482*  3521*  3528*  3535*  3549*  3551*  3756   3779*  3781*  3819*  3824*
                    3868*  3673*  3876*
PCM       015746    2405   2422   4059#
PCONT     007124    2697*  2710   2712#  2740*  2844#  2980*
PCSCOP    006374     521    709*   710   2481   2532*  2559   2561#  2718   2828   2889   2919
PIRQ   = 177772      93#
PIRQVE= 000240      187#
PLOOP     007120    2698*  2711#  2741*  2843*  2979*
PRESTR    012652    2769   2805*  3392*  3400#  3431
PRETES    001726     495    518#
PR0    = 000000     110#
PR1    = 000040     111#
PR2    = 000100     112#
PR3    = 000140     113#
PR4    = 000200     114#  1274
PR5    = 000240     115#  1333
PR6    = 000300     116#
PR7    = 000340     117#   209    211    215    217    219    221    412    968    976    1022   1071   1186
                    1209   1227   1236   1269   1290   1325   1350   2245   2365   2390
PS     = 177776      90#    91
PSW    = 177776      91#  1274*  1333*
PTYP1     007064    2696*  2703#  2739*  2842*  2978*
PWRVEC= 000024      182#  3728*  3729*  3737*  3752*  3753*
RANDAT    012212    3195   3278#  3281
RANGEN    012262    3278   3293#
RANUM     012354    3279   3308*  3313#
RAN1      012350     417*  3294   3300*  3305   3311#
RAN2      012352     418*  3295   3302   3307*  3312#
RDCODE    011674    2798   3152#
RDDONE    007672    2815   2828#
RDER   = 000017     194#  3092   3153
RDERR     007724    2816   2838#
RDIE   = 000107     199#  2817
RDONLY    006564    2246   2635#
READ      007576    2638   2811#  2938
READCH    010400    2653   2656   2671   2937#
REBEGI    002276     635    706    799    806#  2406   2423
RECAL  = 040001     201#  2334   2791
REREAD    007606    2813#  2843   2849   2871   2886   2905   3038
RESERR    006002     210   2414#
RESREG= 104405      3721#  3823
RESTAR    001306     397    412#  3117   3757
RESVEC= 000010      177#
RETURN    010330    2892   2908   2915#
RTJUST    014434    3854#  3876
RTN       007574    2790   2806#
RTSPC     006410    2565   2567#
RXCS      001206     299#   415    523    783*   863#   865   1028*  1142   1144   1189*  1194   1196   1233*
                    1247*  1287*  1305*  1347*  1366*  1377*  1381   1484*  1485   1487   1497   1568*  1587
                    1672*  1771*  1790*  1849*  1851*  1855   1907*  1910   1913   1982*  1984*  1988   2012*
                    2035*  2037*  2041   2064*  2104   2114*  2120   2129   2136   2187*  2190   2193   2262*
                    2274   2277*  2334*  2564   2571   2688*  2689   2725   2777*  2791#  2794   2822   2952*
                    2953   3069   3076   3092*  3119   3153*
RXDB      001210     341#   415*   416*   627    919   1432   1536   1574*  1589   1680   1775   1794   1893
```

DZRXB0.P11 CROSS REFERENCE TABLE

```
                    1914    1924    2002    2016    2054    2068    2086*   2094*   2121    2123*   2130    2132*   2176
                    2194    2202    2265*   2276    2280    2691*   2780*   2783*   2956    3068    3095    3099    3152
                    3156    3160
RXERR0  011376      3070    3086    3089*   3090
RXPWR   011530      3073    3115*
R0      =%000000      98*    395*    401*    422     522*    524     626*    633*    634     864*    866     918*    920
                    1141*   1142    1193*   1194    1283    1343    1380*   1382    1431*   1438*   1439    1488*   1496*
                    1498    1539*   1543*   1544    1679*   1682    1774*   1778    1793*   1797    1846*   1854*   1856
                    1892*   1896    1912*   1923*   1926    1978*   1987*   1989    2001*   2003    2015*   2017    2031*
                    2040*   2042    2053*   2055    2067*   2069    2113*   2135*   2137    2174*   2177    2192*   2201*
                    2203    2265    2315*   2430*   2435*   2437*   2438*   2439*   2440*   2441*   2442*   2487    2691
                    2958    2972*   3007    3008*   3293*   3294*   3295*   3296*   3298*   3299*   3300    3301*   3302*
                    3303*   3304*   3305*   3306*   3307    3308    3505    3506*   3507    3510*   3663    3688*   3699
                    3700*   3701    3702*   3703*   3704*   3705*   3730    3751*   3799*   3800    3818*   3822*   3854
                    3855*   3856    3857    3859    3861*   3863*   3866*
R1      =%000001      99*    523*    524     628*    629*    632*    634     865*    866     919*    920     1144*   1196*
                    1262*   1318*   1328*   1381*   1382    1433*   1434*   1437*   1439    1487*   1497*   1498    1537*
                    1538*   1542*   1544    1589*   1680*   1682    1776*   1777*   1778    1795*   1796*   1797    1855*
                    1856    1894*   1895*   1896    1913*   1924*   1926    1988*   1989    2002*   2003    2016*   2017
                    2041*   2042    2054*   2055    2068*   2069    2104*   2120*   2129*   2136*   2137    2176*   2177
                    2193*   2202*   2203    2276*   2316*   2432    2490    2831    2855    2937*   2944*   2991    2998*
                    3044*   3664    3687*   3731    3750*   3801*   3807*   3813*
R2      =%000002      100*    627*    628     964*    965     967*    968*    969*    1018*   1019    1021*   1022*   1023*
                    1182*   1183    1185*   1186*   1187*   1223*   1224    1226*   1227*   1228*   1265*   1266    1268*
                    1269*   1270*   1321*   1322    1324*   1325*   1326*   1432*   1433    1536*   1537    1567*   1575*
                    1622    1671*   1681*   1775*   1776    1794*   1795    1845*   1893*   1894    1914*   1925*   1979*
                    2000*   2032*   2052*   2081*   2089*   2097*   2112*   2121*   2130*   2175*   2194*   2200*   2243*
                    2244*   2245*   2261*   2266*   2280*   2363*   2364*   2365*   2388*   2389*   2390*   2484    3225*
                    3227*   3230    3665    3686*   3732    3749*   3798*   3801    3802*   3808*   3809*   3814*   3815*
R3      =%000003      101*    708     709     712     807*    808*    809     811*    3600    3609*   3615*   3616*   3619*
                    3624*   3625*   3626    3635*   3666    3685*   3733    3748*   3806*   3811*   3817*   3818
R4      =%000004      102*   1570*   1574    1673*   1679    2471*   2472*   2473    3183*   3284*   3285    3601    3603*
                    3604*   3605*   3606    3607*   3621    3623*   3631*   3634*   3667    3684*   3734    3747*   3804*
                    3807    3813    3815
R5      =%000005      103*    803*    842*    845     2494*   2925*   3000*   3027*   3185*   3186*   3187    3602    3608*
                    3610*   3612*   3613*   3614*   3615    3633*   3668    3683*   3735    3746*   3805*   3809    3816
                    3872    3877*
R6      =%000006      104*    106
R7      =%000007      105*    107
SAVREG= 104404      3720*   3797
SA200   001232       237     401*    815
SA202   001222       238     395*
SCOPIN  006356      2523    2550    2557*
SDN     006412      1378    1494    1585    1772    1791    1908    2013    2065    2083    2091    2100    2118    2127
                    2133    2188    2272    2278    2335    2571*   2792    3093    3154
SECCNT  013136      2730*   2834*   2939    3040*   3413*   3459*   3460*   3461*   3468*
SECLMT  001624       456     472     481*
SEEKER  007324      2751    2754*   2851
SEKRTY  007372      2745*   2763    2765*   2849*
SEKTYP  007376      2757    2767*
SEQUEN  012660       454*   2386*   2392*   3368    3373*   3393    3403*
SEQ1    012760      3395    3430*
SEQ2    013014      3396    3443*
SQLDEC  014504       803    2494    2925    3000    3027    3872*
```

```
SP      =X000006        106#  396*  402*  403*  408   411   412*  413*  447*  465*  470*  489*  492*
                        711*  839*  965*  969   971*  972*  976*  977*  1019* 1023  1025* 1026* 1071*
                       1072* 1183* 1187  1204* 1205* 1209* 1210* 1224* 1228  1230* 1231* 1236* 1237*
                       1266* 1270  1272* 1276  1277  1279  1283* 1284* 1290* 1291* 1322* 1326  1331*
                       1335  1336  1338  1343* 1344* 1350* 1351* 1582* 2102* 2400* 2406* 2417* 2423*
                       2456  2471  2473* 2478* 2481* 2484* 2487* 2490* 2532  2549  2559* 2566* 2609*
                       2626* 2682* 2683* 2685* 2686* 2695  2715* 2716* 2722* 2723* 2735  2769* 2815*
                       2816* 2819* 2820* 2838  2884* 2919* 2947* 2948* 2949* 2950* 2977  3003* 3009*
                       3083* 3105* 3112* 3119* 3128* 3135* 3162* 3168* 3288  3355* 3505* 3506  3507*
                       3509  3510  3511* 3513  3515  3517  3522  3524* 3526* 3534* 3538  3542  3543
                       3547  3592* 3593  3594  3595* 3600* 3601* 3602* 3608  3633  3634  3635  3636*
                       3637* 3663* 3664* 3665* 3666* 3667* 3668* 3669* 3670* 3671* 3672* 3679* 3680*
                       3681* 3682* 3683  3684  3685  3686  3687  3688  3699* 3700  3730* 3731* 3732*
                       3733* 3734* 3735* 3736  3742* 3746  3747  3748  3749  3750  3751  3756* 3777
                       3778* 3780* 3798  3800* 3854* 3855  3866  3867* 3872* 3874*
SPACE   015542          2477  2480  2483  2486  2489  2492  4026#
STACK = 001200            87#   290   396   402   711
STKLMT= 177774            92#
STR     006376          1580  2079  2087  2095  2115  2124  2564#  2778  2781
STYP1   007334          2750* 2756# 2850*
SUBSCO= 104406           903   961  1135  1174  1201  1219  1416  1478  1493  1531  1618  1784  1887
                        1902  1919  1995  2008  2048  2060  2169  2183  2199  2314  3722#
SUM     012260          2965  3184* 3283* 3291#
SWHLT1  007344          2755  2758#
SWR     001216           370#  405   409*  798   831   837   2463  2503  2508  2557  2602  2701  2706
                        2709  2754  2758  2762  2789  2796  2799  2802  2861  2865  2868  2875  2879
                        2882  2893  2900  2903  2910  2985  2987  3012  3023  3036  3056  3061  3078
                        3085  3100
SWREG   000176           232#   409
SW0     = 000001         145#  2508
SW00    = 000001         135#   145
SW01    = 000002         134#   144
SW02    = 000004         133#   143
SW03    = 000010         132#   142
SW04    = 000020         131#   141
SW05    = 000040         130#   140
SW06    = 000100         129#   139
SW07    = 000200         128#   138
SW08    = 000400         127#   137
SW09    = 001000         126#   136
SW1     = 000002         144#
SW10    = 002000         125#   831
SW11    = 004000         124#  2557  2709  2762  2802  2868  2882  2903  3036
SW12    = 010000         123#   837
SW13    = 020000         122#  2463  2602  2701  2754  2861  2875  2893  2910  2985  3023  3056  3078
                        3085  3100
SW14    = 040000         121#   798
SW15    = 100000         120#  2503
SW2     = 000004         143#
SW3     = 000010         142#
SW4     = 000020         141#
SW5     = 000040         140#
SW6     = 000100         139#
SW7     = 000200         138#
```

```
SW8    = 000400            137#  2789
SW9    = 001000            136#  2987
S2BIG    016037            496   4071#
TAB      015016           3166   3925#
TARGET   012650           2255*  2783   2963   2966   3128   3374*  3392   3399*  3410*  3415*  3418   3420*  3422*
                          3433*  3435*  3446*  3448*
TBITVE=  000014            178#
TCRLF  = 000200           3515   3554#
TESTS    002064            709    712    713#
THT    = 000011           3513   3553#
TKVEC  = 000060            185#
TPVEC  = 000064            186#
TRAPVE= 000034             184#
TRKCNT   012646           2639*  2657*  2672*  3379*  3380*  3381*  3384*  3386*  3389*
TRKLMT   001542            453    460#
TRTVEC=  000014            179#
TSECTO   013140           2256*  2780   2964   2967   3133   3462*  3463*  3469*  3472*  3473*  3474   3476*  3477*
                          3481*
TYPADR   011572           2898   2996   3109   3127#
TYPCOD   011742           3111   3161#
TYPE   = 104400            424    435    445    446    451    455    464    469    473    488    491    494    496
                           789    794    795    800    801    814   2398   2405   2414   2422   2468   2470   2477
                          2480   2483   2486   2489   2492   2510   2605   2703   2704   2705   2756   2767   2768
                          2773   2863   2864   2877   2878   2896   2899   2913   2918   2923   2927   2994   2995
                          2997   3002   3006   3011   3025   3029   3032   3034   3035   3059   3060   3080   3081
                          3087   3088   3102   3104   3110   3115   3116   3118   3123   3124   3127   3132   3139
                          3142   3144   3161   3166   3167   3170   3518   3627   3716#  3754   3864
TYPERR   010752           2992   2999#
TYPOC  = 104401           3717#
TYPON  = 104403            471    490    493   2479   2482   2485   2488   2491   3010   3169   3719#
TYPOS  = 104402            448    466   2402   2419   2474   2610   2770   2920   3004   3106   3120   3129   3136
                          3163   3718#
TONOTH   002036            708#   713
TT1      002446            713    863#
TT10     003766            716   1661#
TT11     004054            716   1753#
TT12     004044            716   1748#
TT13     004120            716   1769#
TT14     004244            716   1845#
TT15     004436            716   1978#
TT16     004576            716   2031#
TT17     005044            716   2112#
TT2      002566            713   1018#
TT20     005310            719   2220#  2828
TT21     005334            719   2255#
TT21X    005436           2269   2271#  2353
TT22     005520            719   2327#
TT23     005576            719   2346#
TT24     005644            719   2362#
TT24X    005652           2363#  2376
TT25     005676            719   2375#  2718   2889
TT26     005710            719   2386#
TT3      003110            713   1262#
TT4      003246            713   1318#
```

F15


```
T5        003412            713    1377#
T6        003636            713    1561#
T6FILL    003654           1562    1567#  1754    2330    2347
T7        003770            713    1669#
T7EMPT    004000           1669    1671#  1755    2317    2337
UNITSE    012446            425*    428*   440*    443*    518*    630     784     806*    841*   1435    1540    1769    1788
                           1847    1980   2033    2327    2466   2498*   2776    2897*   2909*  2983*   3091*   3140    3330
                           3332    3334*  3335*   3337    3339   3341*   3342*   3347*   3351   3353    3479
WRITE     006736           2652    2655   2670    2679#
WRTDON    007226           2715    2730#  2740    2752
WRTER     007242           2716    2735#
WRTIE =   000105            198#   2717
WRTSEK    007302           2737    2745#
WTDOIE=   000115            200#   2720
WTROCK    006670           2391    2666#
XERROR    006060            218    2456#
XFRBYT    007012           2691#
XHOME     007566           2795    2805#
XID       012656           3375*   3376*  3379    3402#   3431
XOD       012654           3377*   3378*  3380    3401#
XPATGE    012100           3224#   3242
XREAD     007602           2812#   2835   3042
XSA202    001470            423     429    442     444#
XSCOPE    006316            214    2522#
XSDN      006540           2572    2624#
XSUBSC    006336           2537#   3722
XWRITE    006742           2680#   2697   2733
XWTROC    006674           2667#   2676
XXPATG    012150           3251#   3253   3262
$CHARC    013440           3529*   3536   3545*   3550#
$CRLF     013455           3519    3563#
$DB2D     014240           3779    3797#
$DECVL    014420           3799    3845#
$FILLC    013452           3522    3560#
$FILLS    013451           3559#
$HD    =  000003             21      22
$ILLUP    014172           3728    3759#
$LF       013457           3564#
$MAIL =   ****** U         3507
$NULL     013450           3524    3558#
$OCNT     013704           3599*   3628*  3641#
$OMODE    013706           3594*   3608*  3603    3606*   3617*   3643#
$POWER    014200           3116    3755   3762#
$PWRAD    014166           3757#
$PWRDN    014044            216    3728#  3752
$PWRMG    014162           3755#
$PWRUP    014112           3737    3742#
$QUES     013454           3562#
$RDCHR=   ****** U         3720
$RDOEC=   ****** U         3720
$RDLIN=   ****** U         3720
$RDOCT=   ****** U         3720
$RESRE    013746           3678#   3721
$R2A   =  ****** U         3722
```

DZRXBD.P11 CROSS REFERENCE TABLE

```
$SAVRE  013710          3662#  3720
$SAVR6  014176          3736*  3742   3743*  3744*  3761#
$SB2D   014210          3777#  3873
$SWR  = 160000            21     22#  3758
$TN   = 000001            21#
$TNPWR  014350          3804   3805   3825#
$TPB    013446          3542*  3557#
$TPFLG  013453          3501   3561#
$TPS    013444          3540   3556#
$TRAP   014004           220   3699#
$TRP  = 000007          3707#  3717#  3718#  3719#  3720#  3721#  3722#  3723#
$TRPAD  014026          3704   3715#
$TYPBN= ****** U        3720
$TYPOS= ****** U        3720
$TYPE   013230          3501#  3707   3716
$TYPEC  013374          3521   3528   3535   3540#  3541
$TYPEX  013442          3546   3548   3551#
$TYPOC  013506          3597#  3717
$TYPON  013522          3596   3599#  3719
$TYPOS  013462          3592#  3718
$OFILL  013705          3593*  3597*  3607   3642#
.     = 016500           204#   206    207#   213#   223#   227#   230#   234#   290#   827#  1749   2622#  3556
                        3557   3558   3559   3560   3561   3562   3563   3564   3565#  3739   3760   3845#  4109#
```

```
COMMEN     6#   188#   531   644   873   927   987  1031  1079  1150  1389  1448  1505  1595  1629
        1693  1804  1863  1933  2146  2222  2286  2583
ENDCOM     7#   188#   609   702   899   957  1006  1066  1109  1170  1412  1474  1527  1614  1648
        1737  1830  1883  1964  2165  2239  2310  2598
ERROR     88#   529   641   871   925   985  1077  1148  1200  1218  1242  1297  1360  1387  1446
        1492  1503  1551  1593  1627  1691  1783  1802  1861  1901  1918  1931  1994  2007  2021
        2047  2059  2073  2144  2182  2198  2208  2284  2582
ESCAPE   188#
MULT     188#
M7727    241#   578   683  1393  1718  1808  1937  2290
M7846    239#   530   643   872   926   986  1030  1078  1149  1447  1504  1594  1628  1692  1862
        2145
NEWTST   188#
POP      188#  3683  3746
PUSH     188#  3663  3730
SCOPE     89#   520  1010  1246  1304  1365  1552  1652  1741  1834  1968  2022  2074  2209
SETPRI     6#  1283  1290  1343  1350
SETTRA  3707#  3717  3718  3719  3720  3721  3722
SETUP    188#
SKIP     188#
SLASH    188#
SPACE    188#
STARS      6#   188#   244   245   246   257   258   259   264   265   266   285   286   287   378
         389  2448  2449  2453  2454  2643  2661  3052  3179  3202  3211  3236  3244  3255  3264
        3273  3328  3390  3405  3425  3438  3483  3566  3644  3690  3723  3765  3783  3847  4104
TRMTRP  3707#
TYPBIN   188#
TYPDEC   188#
TYPNUM   188#
TYPOCS     5#   188#   447   465  2400  2417  2769  2919  3105  3119  3128  3135  3162
TYPOCT   188#
TYPTXT   188#
$$ESCA   188#
$$NEWT   188#
$$SET   3707#  3717  3718  3719  3720  3721  3722
$$SKIP   188#
.EQUAT     5#    83
.HEADE     5#    11
.$DB2D     6#  3783
.$POWE     6#  3723
.$SAVE     6#  3644
.$SB2D     6#  3765
.$TRAP     5#  3690
.$TYPE     5#  3483
.$TYPO     5#  3566
```

DZRXBD.P11 CROSS REFERENCE TABLE

```
ADC     2574  3814
ADD      416   839  1582  2102  2457  2566  2573  2626  2884  2957  2966  2967  3083  3112  3283
        3294  3295  3305  3355  3473  3477  3511  3595  3605  3813  3815  3874
ASL      811  2437  2438  2439  2440  2441  3186  3703
ASLB    2969
BCC     3226
BEQ      406   431   525   635   785   799   832   838   843   867   921  1143  1195  1383  1440
        1486  1499  1545  1623  1[illegible]3  1779  1798  1857  1897  1927  1990  2004  2018  2043  2056
        2070  2138  2178  2204  24[illegible]  2504  2550  2558  2575  2603  2710  2726  2737  2823  2829
        2840  2904  2908  3019  3022  3031  3037  3041  3072  3090  3096  3157  3231  3286  3372
        3394  3419  3480  3514  3622  3858
BGE     3475  3548
BGT     2990  3629
BHI      461   463   485   487
BHIS     810
BIC      444   518   629   632   633   783   806   808   841  1189  1247  1305  1366  1434  1437
        1438  1538  1542  1777  1796  1895  2392  2442  2998  3134  3296  3306  3334  3341  3370
        3411  3416  3430  3445  3472  3619
BIS      428   440   443  1028  1233  1287  1328  1347  1377  2334  2386  2498  2897  2909  2937
        2944  2983  3091  3335  3342  3373  3382  3481  3624  3625  3817
BISB    2776
BIT      426   430   784   798   831   837  1485  2463  2466  2503  2508  2557  2571  2602  2701
        2709  2725  2736  2754  2762  2789  2796  2802  2822  2839  2846  2861  2868  2875  2882
        2893  2903  2907  2910  2985  2987  3023  3036  3056  3071  3074  3078  3085  3095  3100
        3140  3156  3330  3339  3353  3479
BITB    2891  3076  3536
BLT     3527  3630  3810
BMI      631  1436  1541  2434  3070  3369  3395
BNE      423   427   434   453   482   788   792  1120  2464  2467  2509  2523  2538  2572  2640
        2658  2673  2702  2719  2728  2731  2755  2763  2790  2797  2803  2825  2835  2847  2860
        2862  2869  2876  2883  2890  2892  2894  2911  2940  2959  2986  2988  3024  3057  3075
        3077  3079  3086  3101  3141  3331  3340  3354  3432  3455  3508  3516  3523  3537  3544
        3620  3745  3821  3860
BPL      442  1588  1770  1789  1848  1911  1981  2034  2191  2275  2328  2565  2690  2707  2759
        2795  2800  2832  2856  2866  2880  2901  2954  2992  3013  3062  3333  3338  3352  3409
        3414  3444  3502  3541  3618
BR       235   236   397   407   429   456   472   477   495   502   706   812  1123  1275  1278
        1334  1337  1354  1379  1495  1576  1581  1586  1742  1749  1773  1792  1850  1853  1909
        1983  1986  2014  2036  2039  2066  2080  2082  2084  2085  2088  2090  2092  2093  2096
        2098  2101  2105  2116  2117  2119  2122  2125  2126  2128  2131  2134  2189  2264  2267
        2273  2279  2336  2676  2693  2752  2779  2782  2793  2852  2870  2914  2961  3033  3094
        3098  3143  3155  3159  3209  3218  3234  3242  3253  3262  3272  3281  3385  3387  3396
        3504  3520  3530  3539  3546  3596  3611  3632  3739  3760  3812  3862
BVS      797
CLC     3229  3297
CLR      401   419   420   421   425   454   476   519   793   863   918   971  1025  1136  1204
        1230  1262  1318  1488  1543  1567  1671  1753  1845  1846  1892  1925  1978  1979  2000
        2031  2032  2052  2053  2086  2094  2112  2113  2174  2175  2200  2261  2315  2316  2329
        2346  2465  2531  2624  2684  2685  2714  2722  2813  2814  2819  2942  2943  2946  2949
        3044  3184  3207  3241  3301  3375  3377  3422  3609  3743  3806
CLRB    3461  3545  3822
CMP      405   408   524   634   791   809   866   920  1142  1194  1276  1335  1382  1439  1498
        1544  1622  1682  1778  1797  1856  1896  1926  1989  2003  2017  2042  2055  2069  2137
        2177  2203  2549  2718  2828  2889  2989  3285  3418  3816
```

DZRXB0.P11 CROSS REFERENCE TABLE

```
CMPB    432   460   462   484   486  2958  3431  3474  3513  3515  3522  3543  3547  3859
COMB   3252
DEC    2435  2639  2657  2672  2730  2834  3040  3448  3819
DECB   3526  3529  3617  3628
EMT      88
HALT    206   439   805   833  1130  2505  2708  2760  2801  2867  2881  2902  3014  3063  3503
       3738  3759
INC     395   790   796  1575  1681  2081  2089  2097  2266  2331  2348  2458  2692  2727  2824
       2857  2960  2984  3008  3015  3089  3227  3270  3381  3435  3456  3460  3478  3623  3631
       3744  3811
INCB   1121  3549
IOT      89
JMP     237   238   712   815  1011  1249  1306  1367  1553  1563  1670  1756  1758  1761  1835
       1969  2023  2075  2210  2247  2270  2338  2354  2367  2376  2393  2604  2613  2699  2711
       2712  2729  2733  2742  2765  2826  2845  2871  2885  2886  2905  2906  2941  2981  3016
       3039  3043  3073  3117  3187
JSR     707   803   845  1280  1339  1378  1494  1562  1569  1571  1580  1585  1669  1674  1754
       1755  1772  1791  1852  1908  1985  2013  2038  2065  2079  2083  2087  2091  2095  2100
       2115  2118  2124  2127  2133  2188  2246  2257  2259  2260  2263  2269  2271  2272  2278
       2317  2330  2332  2333  2335  2337  2347  2349  2350  2351  2353  2366  2391  2494  2499
       2635  2636  2637  2638  2648  2649  2650  2651  2652  2653  2654  2655  2656  2666  2667
       2668  2669  2670  2671  2674  2679  2680  2681  2721  2751  2757  2761  2778  2781  2792
       2798  2811  2812  2818  2830  2851  2858  2895  2898  2912  2925  2928  2938  2945  2993
       2996  3000  3027  3038  3042  3058  3064  3093  3097  3103  3109  3111  3154  3158  3208
       3228  3251  3271  3278  3280  3521  3528  3535  3779  3873  3876
MOV     396   402   403   404   409   410   411   412   413   415   417   418   447   465   501
        521   522   523   626   627   628   709   710   711   786   802   807   842   864   865
        919   964   965   967   968   969   972   976   977  1018  1019  1021  1022  1023  1026
       1071  1072  1141  1144  1182  1183  1185  1186  1187  1193  1196  1205  1209  1210  1223
       1224  1226  1227  1228  1231  1236  1237  1265  1266  1268  1269  1270  1272  1273  1274
       1277  1279  1283  1284  1290  1291  1321  1322  1324  1325  1326  1331  1332  1333  1336
       1338  1343  1344  1350  1351  1380  1381  1431  1432  1433  1484  1487  1496  1497  1536
       1537  1539  1561  1568  1570  1589  1672  1673  1748  1757  1760  1771  1774  1775  1776
       1790  1793  1794  1795  1849  1851  1854  1855  1893  1894  1907  1912  1913  1914  1923
       1924  1982  1984  1987  1988  2001  2002  2012  2015  2016  2035  2037  2040  2041  2054
       2064  2067  2068  2104  2114  2120  2121  2123  2129  2130  2132  2135  2136  2176  2187
       2192  2193  2194  2201  2202  2243  2244  2245  2255  2256  2258  2262  2268  2276  2277
       2280  2352  2362  2363  2364  2365  2375  2387  2388  2389  2390  2400  2406  2417  2423
       2430  2456  2471  2473  2478  2481  2484  2487  2490  2493  2532  2559  2625  2682  2683
       2686  2688  2696  2697  2698  2715  2716  2723  2739  2740  2741  2745  2750  2769  2777
       2791  2805  2815  2816  2820  2842  2843  2844  2849  2850  2919  2924  2947  2948  2950
       2952  2965  2972  2978  2979  2980  2999  3003  3009  3026  3092  3105  3119  3128  3133
       3135  3153  3162  3168  3183  3185  3225  3269  3293  3300  3302  3307  3308  3379  3384
       3386  3410  3415  3420  3446  3465  3505  3506  3510  3524  3592  3600  3601  3602  3608
       3615  3633  3634  3635  3636  3637  3663  3664  3665  3666  3667  3668  3669  3670  3671
       3672  3679  3680  3681  3682  3683  3684  3685  3686  3687  3688  3699  3700  3704  3728
       3729  3730  3731  3732  3733  3734  3735  3736  3737  3742  3746  3747  3748  3749  3750
       3751  3752  3753  3756  3777  3778  3780  3798  3799  3800  3801  3802  3803  3804  3805
       3854  3855  3856  3866  3867  3872
MOVB    470   483   489   492  1574  1679  1680  2265  2691  2717  2720  2780  2783  2817  2956
       2963  2964  2968  2971  3007  3068  3099  3152  3160  3217  3223  3250  3261  3279  3284
       3374  3376  3378  3392  3433  3457  3458  3462  3476  3507  3534  3542  3593  3594  3597
       3598  3599  3603  3606  3607  3626  3702  3818  3861  3863
NEG    3604
```

DZRXBD.P11 CROSS REFERENCE TABLE

```
          3756    3758    3762    3765    3783    3847    4104
.IFF        88     245     246     247     258     259     260     265     266     267     286     287     288     379     390
          450     468    2403    2404    2420    2421    2449    2450    2454    2455    2644    2662    2772    2921    2922
         3053    3108    3122    3131    3138    3165    3180    3203    3212    3237    3245    3256    3265    3274    3329
         3391    3406    3426    3439    3484    3567    3645    3691    3700    3724    3756    3766    3784    3848    4105
.IIF        11      16      21      22     448     466    2401    2418    2770    2920    3106    3120    3129    3136    3163
         3556    3557    3558    3559    3560    3561    3562    3563    3564    3565    3716    3717    3718    3719    3720
         3721    3722
.IRP     3663    3683    3730    3746
.LIST       3     188     206     531     613     644     706     873     903     927     961     987    1010    1031    1070
         1079    1113    1150    1174    1389    1416    1448    1478    1505    1531    1595    1618    1629    1652    1693
         1741    1804    1834    1863    1887    1933    1968    2146    2169    2222    2243    2286    2314    2583    2602
         3707    3716    3717    3718    3719    3720    3721    3722    3723
.MACRO    239     241    3707
.MCALL      5       6       7     188
.NLIST      2     188     206     531     613     644     706     873     903     927     961     987    1010    1031    1070
         1079    1113    1150    1174    1389    1416    1448    1478    1505    1531    1595    1618    1629    1652    1693
         1741    1804    1834    1863    1887    1933    1968    2146    2169    2222    2243    2286    2314    2583    2602
         3707    3716    3717    3718    3719    3720    3721    3722    3723
.PAGE      42      82    2446
.REPT     206     532     609     645     702     874     899     928     957     988    1006    1032    1066    1080    1109
         1151    1170    1390    1412    1449    1474    1506    1527    1596    1614    1630    1648    1694    1737    1805
         1830    1864    1883    1934    1964    2147    2165    2223    2239    2287    2310    2584    2598
.SBTTL     84     242     262     288     339     391     503     850    1012    1250    1308    1368    1554    1655    1663
         1744    1751    1763    1836    1971    2024    2107    2212    2250    2319    2341    2356    2370    2378    2446
         2518    2630    2677    2808    2932    3066    3172    3315    3358    3450    3485    3568    3646    3692    3708
         3725    3767    3785    3878
.TITLE     11
.WORD     205     208     209     210     211     228     370     371    1124    3005    3550    3556    3557    3643    3755
         3757    3782    3820
```

ERRORS DETECTED: 0

```
*DZRXBD,DZRXBD/CRF/SOL=DZRXBD
RUN-TIME:  36 29 5 SECONDS
CORE USED:  12K
```