RX11

RX11 RELIABILITY

MD-11-DZRXA-D

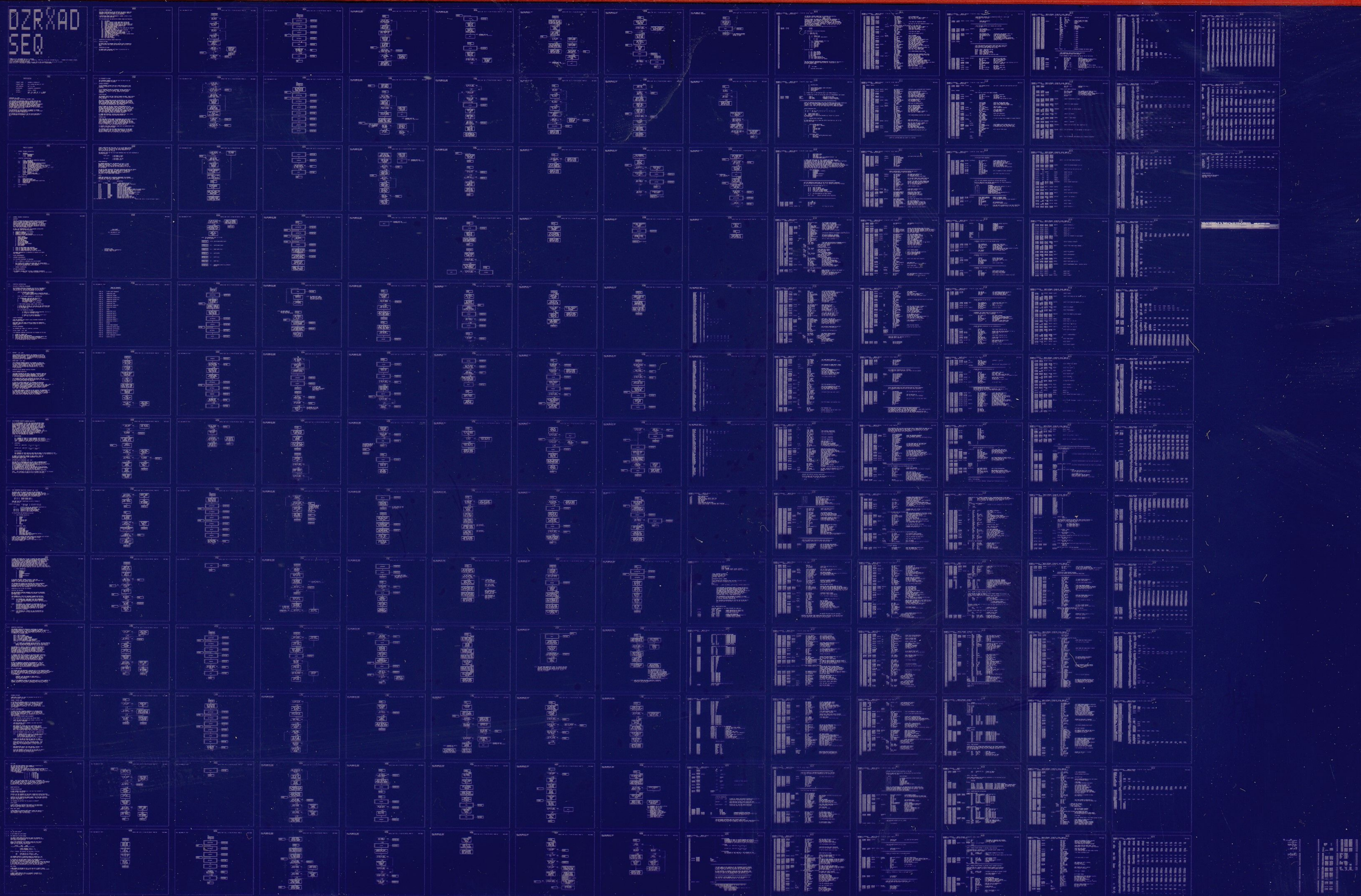
EP-DZRXA-D-DL-A

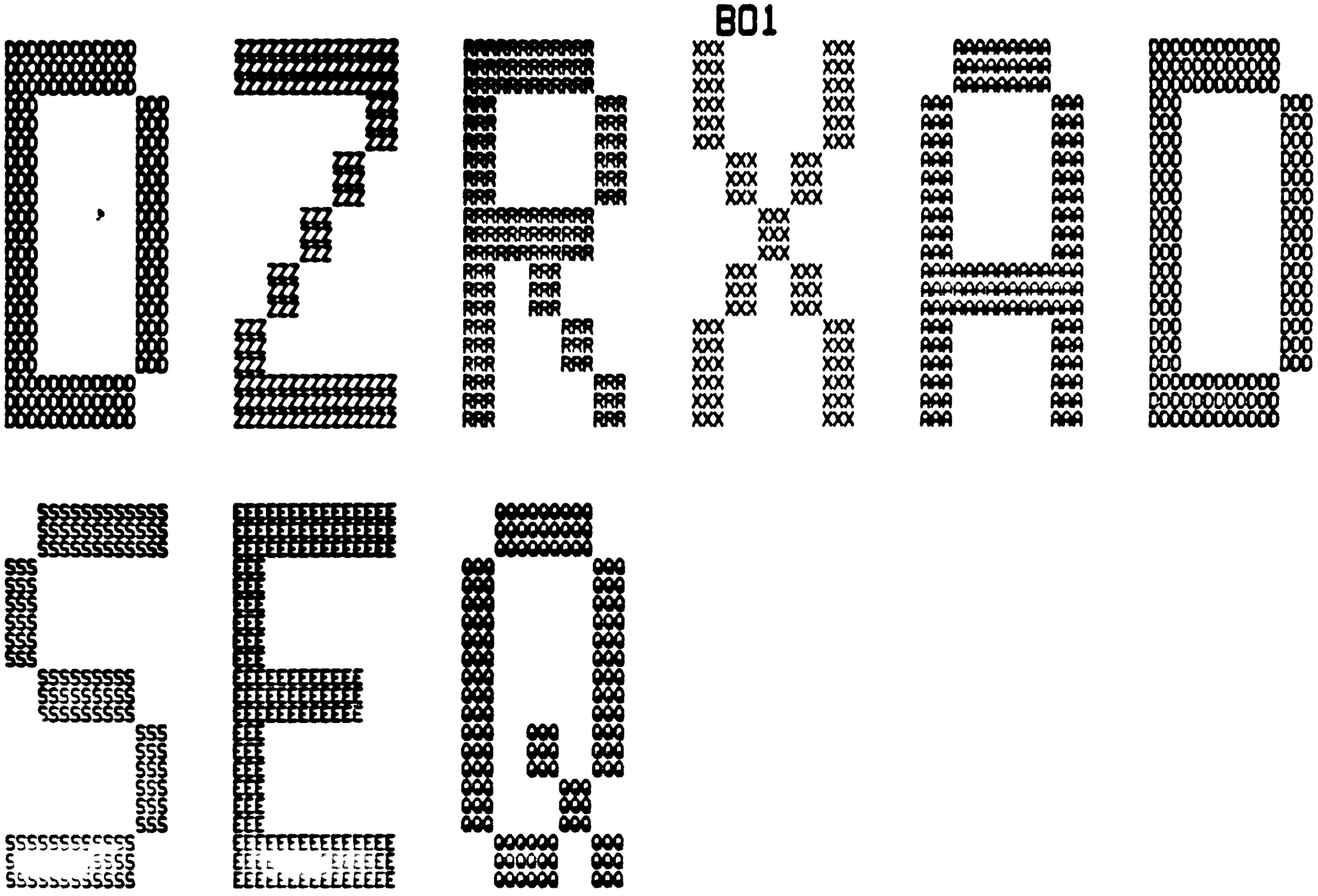
FEB 1976

FICHE 1 OF 1

digital

Made In U.S.A.





LPTSPL Version 101(2107) Running on MTA140
 START User WELLINGHAM,A[1,2] Job DZRXAD Seq. 5976 Date 17-Jan-76 14:43:25 Monitor RV22SD KI10 SYS#514 *START*
Request created: 17-Jan-76 14:13:41
File: WHTG:DZRXAD.SEQ(055)[404,3722] Created: 09-Jan-76 16:12:39 Printed: 17-Jan-76 14:43:26
QUEUE Switches: /FILE:ASCII /COPIES:1 /SPACING:1 /LIMIT:477 /FORMS:NORMAL

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZRXA-D-D

PRODUCT NAME: RX11 SYSTEM RELIABILITY TEST

DATE CREATED: DEC. 21,1975

MAINTAINER: DIAGNOSTIC ENGINEERING

AUTHOR: DAVID L. ADAMS

REVISED: SEPT. 12, 1975 BY D. L. ADAMS
NOV. 20, 1975 BY B. P. BURGESS

TABLE OF CONTENTS

1.0 GENERAL PROGRAM INFORMATION

1.1 ABSTRACT

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

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

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

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

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

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

1.2 SYSTEM REQUIREMENTS

1.2.1 HARDWARE REQUIREMENTS

THE FOLLOWING EQUIPMENT IS REQUIRED.

A. PDP-11 SERIES OF COMPUTER WITH MIN 8K MEMORY

B. RX11 FLOPPY DISK SYSTEM INCLUDING DUAL OR SINGLE DRIVE RXDI AND PDP-11 INTERFACE. (SEE SECTION 2.3 FOR SELECTION OF REGISTER ADDRESSES AND VECTOR ADDRESS)

C. CONSOLE TELEPRINTER

1.2.1 SOFTWARE REQUIREMENTS

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

2.0 OPERATING INSTRUCTIONS

2.0.1 OUTLINE OF OPREATING PROCEDURE

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

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

B. START THE PROGRAM RUNNING AT LOCATION 200

C. THE PROGRAM WILL TYPE OUT THE FOLLOWING:
   1. PROGRAM NAME AND REVISION
   2. RX11 REGISTER AND VECTOR ADDRESSES
   3. UNITS BEING TESTED
   4. "P"ATTERN, "T"EST, AND "S"EQUENCE

   IT THEN STARTS RUNNING UNDER THOSE CONDITIONS

D. IF THERE ARE NO ERRORS, AT THE END OF THE COMPLETED PASS A "D" AND "BELL" IS TYPED,AND THE PROGRAM WILL CONTINUE ON FOR ANOTHER PASS.

E. HALT THE PROGRAM AS FOLLOWS:

   1. IF THERE IS A HARDWARE SWITCH REGISTER, PUT SW 14 UP TO HALT AT THE END OF PASS.

   2. IF THERE IS NO SWITCH REGISTER, OR TO HALT AT ANY TIME, HALT THE PROCESSOR.

2.1 LOADING PROCEDURE

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

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

2.2 STARTING ADDRESSES

THE PROGRAM HAS THREE (3) STARTING LOCATIONS.

2.2.1 INITIAL START (LOC. 200)

THIS STARTING LOCATION INITIALIZES THE PROGRAM AS FOLLOWS:

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

2.2.2 RESTART (LOC. 202)

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

2.2.3 ERROR DUMP (LOC. 204)

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

2.3 OPERATOR ACTION BEFORE STARTING PROGRAM

2.3.1 DEVICE ADDRESS SELECTION

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

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

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

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

2.3.2 NON-STANDARD DISKETTE ADDRESS SELECTION SEQ 0006

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

A. DEFINITIONS:

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

B. LOCATIONS:

```
TRACKS LOC. 1200 BITS   14------8  6------0
                           ID        OD

SECTORS LOC. 1202 BITS  12------8  4------0
                          LAST      FIRST
```

C. RESTRICTIONS:

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

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

2.3.3 SOFTWARE SWITCH REGISTER (LOC. 176)

FOR THE PDP 11 PROCESSORS THAT DO NOT HAVE A HARDWARE SWITCH REGISTER, OR IF THE OPERATOR WISHES TO SELECT THE SOFTWARE SWITCH REGISTER, BY PUTTING ALL THE SWITCHES UP TO A "1", LOCATION 176 IS ASSIGNED AS THE SWITCH REGISTTER. 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 0007

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

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

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

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

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

THE SELECTIONS ARE DEFINED AS FOLLOWS:

P = DATA PATTERN SELECTION

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

T = FUNCTIONAL TESTS

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

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

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

S - TRACK SEQUENCING

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

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

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

2.4 OPERATOR ACTION TO RUN THE PROGRAM

2.4.1 STARTING THE PROGRAM

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

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

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

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

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

2.4.2 OPERATING CONDITIONS

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

SW15 = HALT ON ERROR
SW14 = HALT AT END OF PASS
SW13 = DON'T PRINT ERROR MESSAGE
SW12 = TYPE ONLY 10 DATA ERRORS
SW11 = NO RETRY ON ERROR, LOG HARD ERROR
SW08 = NO RECALIBRATION ON SEEK ERRORS
SW15-SW00 (1) = SELECT SOFTWARE SWITCH REGISTER

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

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

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

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

2.4.3 ACT11 & XXDP HOOKS

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

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

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

2.5 PROGRAM OPTIONS

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

2.5.1 PSEUDO - SCOPE LOOP

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

2.5.2 DISKETTE COMPATABILITY

TO CHECK FOR DATA TRANSFER COMPATABILITY BETWEEN DRIVE UNITS USE THE FOLLOWING PROCEDURE. (IT IS ASSUMED THE SYSTEM HAS DUAL DRIVE RXO1, IF NOT YOU MAY TEST COMPATABILITY BETWEEN DIFFERENT RX11 SYSTEMS BY RUNNING THE SAME TESTS ON BOTH.)
THIS TEST INSURES PROPER HEAD ALIGNMENT.

A. HAVE DISKETTES IN BOTH DRIVES AND DRIVES READY.

B. CLEAR THE OD/ID AND FIRST/LAST MEMORY LOCATION, FOR TOTAL DISKETTE TRANSFERS.

C. LOAD THE INITIAL START ADDRESS (LOC 200) AND START THE PROGRAM RUNNING.

D. WHEN ASKED FOR "DRIVE AND TEST CONDITIONS" SELECT BOTH DRIVES, ANY "P"ATTERN OF DATA, "S"EQUENCE #1 (INCRIMENT TRACKS TO INSURE ALL HEAD POSITIONS ARE ALIGNED) AND "T"EST 3 (WRITE/READ CHECK). THIS WILL WRITE THEN READ AND VERIFY THE DATA ON BOTH DISKETTES.

E. ALLOW THE PROGRAM TO RUN FOR AT LEAST 1 COMPLETE PASS.

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

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

F. AFTER COMPLETION OF THE PASS, PROGRAM HALTED. SWAP DISKETTES, AND AS YOU ONLY WANT TO READ VERIFY THE DATA, START THE PROGRAM AGAIN AT LOC 200 (INITIAL START).

G. WHEN REQUESTED SELECT THE SAME DRIVES, PATTERN, AND SEQUENCE, BUT SELECT TEST 4 (READ CHECK ONLY).

H. ALLOW THE PROGRAM TO RUN AS LONG AS YOU WISH TO VERIFY THAT DATA WRITTEN AND CHECKED ON ONE DRIVE CAN BE READ AND VERIFIED ON THE OTHER DRIVE.

2.6 RUN TIME

RUN TIME PER PASS DEPENDS UPON NUMBER OF A FUNCTIONS IN
THE TEST SELECTED AND THE TRACK SEQUENCE.
EXAMPLES OF RUN TIMES FOLLOW.
(THESE TIMES ARE FOR 1 DRIVE AND COMPLETE DISKETTE,MAXIMUM ID/OD,
FIRST/LAST LIMITS.IF OPERATING ON 2 DRIVES DOUBLE THE TIME.)
THESE TIMES ARE FOR A PDP 11/05 PROCESSOR AND MAY CHANGE SLIGHTLY FOR
A FASTER PROCESSOR.

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

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

3.0 ERROR DETECTION

3.1 PROGRAM DEFINITIONS

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

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

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

THE PROGRAM HAS DEFINED THE FOLLOWING AS ERRORS.

3.1.1 WRITE ERROR

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

3.1.2 READ (CRC) ERROR

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

3.1.3 CRC AND DATA ERROR

3.1.4 NO CRC ERROR BUT DATA ERROR

3.1.5 CRC ERROR BUT NO DATA ERROR

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

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

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

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

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

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

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

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

3.1.6 SEEK ERROR

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

3.1.7 PARITY ERROR

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

3.2 DEFINITIVE ERROR CODES

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

A DEFINITIVE ERROR CODE REPRESENTS (WHERE) WITHIN A FUNCTION AN ERROR WAS DETECTED.

THE FOLLOWING ARE THE DEFINITIVE ERROR CODES AND MEANINGS:

00 - NO ERROR
[illegible] - DRIVE 0 FAILED TO SEE HOME FROM INITIALIZE
[illegible] - DRIVE 1 FAILED TO SEE HOME FROM INITIALIZE
[illegible] - HOME FOUND WHEN STEPPING OUT 10 TRACKS FROM INIT
[illegible] - TRIED TO ACCESS A TRACK GREATER THAN 77(DECIMAL)
[illegible] - HOME WAS FOUND BEFORE DESIRED TRACK
[illegible] - SELF DIAGNOSTIC ERROR
70 - DESIRED SECTOR NOT FOUND AFTER SAMPLING 52 HEADERS
100 - WRITE PROTECT ERROR
110 - MORE THAN 400US 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 SUPPOSIDLY GOOD HEADER
150 - GOOD HEADER (NO CRC ERROR) BUT TRACK COMPARE ERROR
160 - IDAM NOT FOUND IN TIME
170 - DATA AM NOT FOUND IN TIME
200 - DATA CRC ERROR
210 - ALL PARITY ERRORS

3.3 UNEXPECTED OR MISSING ERROR CONDITIONS

3.3.1 MISSING DD MARK

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

3.3.2 UNEXPECTED DD MARK

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

3.3.3 NO INTERRUPT ON DONE

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

3.3.4 UNKNOWN INTERRUPT

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

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

3.4 POWER FAILURE

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

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

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

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

3.5 PROGRAM HUNG

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

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

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

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

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

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

4.0 ERROR REPORTING

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

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

5.0 HALTS

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

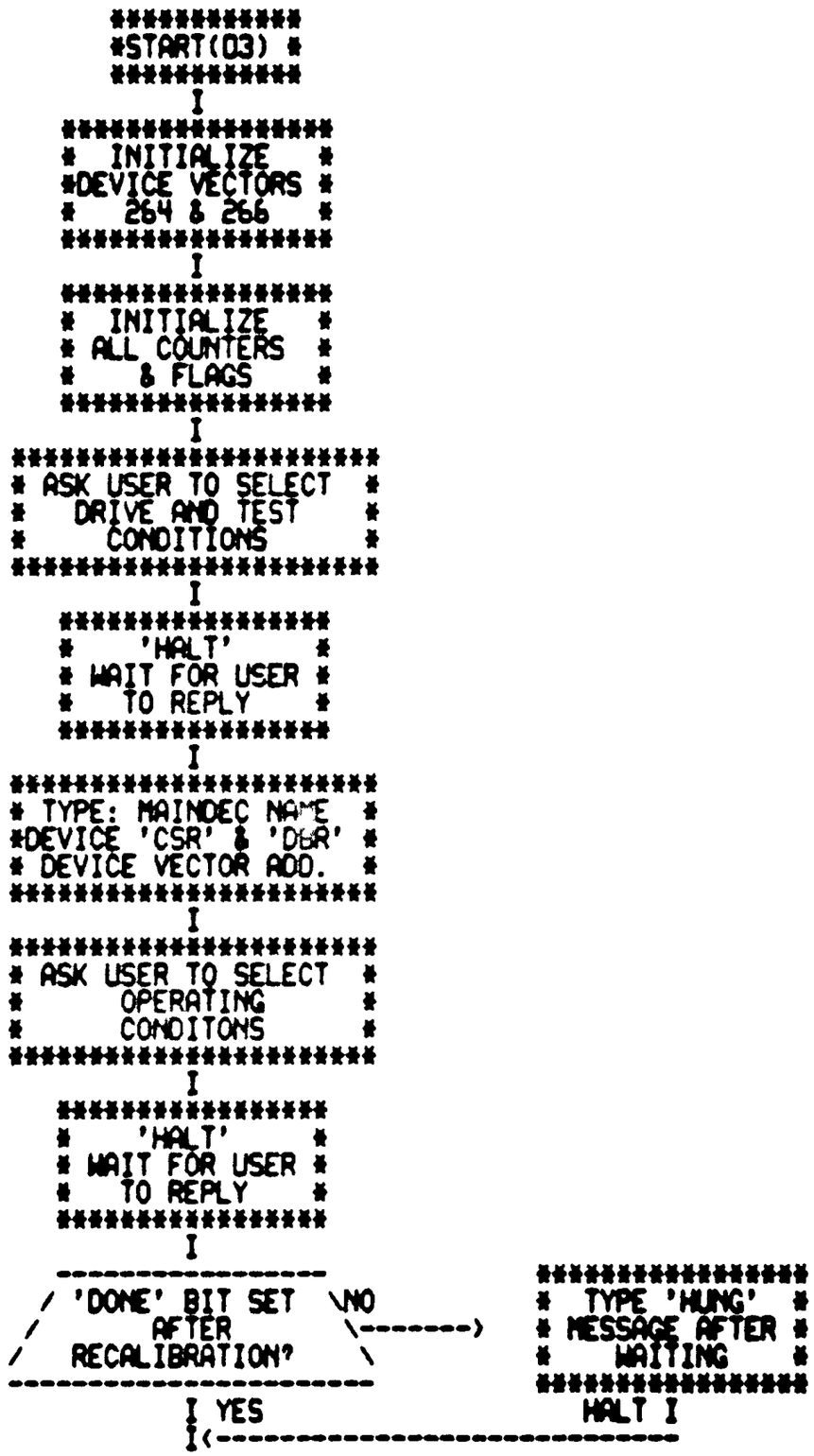
HALT #	TYPE	DEFINITION
HLT1:	NO ERROR	;CAUTION - LOAD MEDIUM ON UNIT 0
HLT3:	ERROR	;ERRORS FOUND ON RECAL FUNCTIONS
HLT4:	ERROR	;HARD PARITY ERRORS
HLT5:	ERROR	;NO DRIVES READY
HLT6:	ERROR	;SW15 SET ON PARITY ERROR
HLT7:	ERROR	;SW15 SET ON SEEK ERROR
HLT10:	ERROR	;SW15 SET ON CRC GENERATOR ERROR (NO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

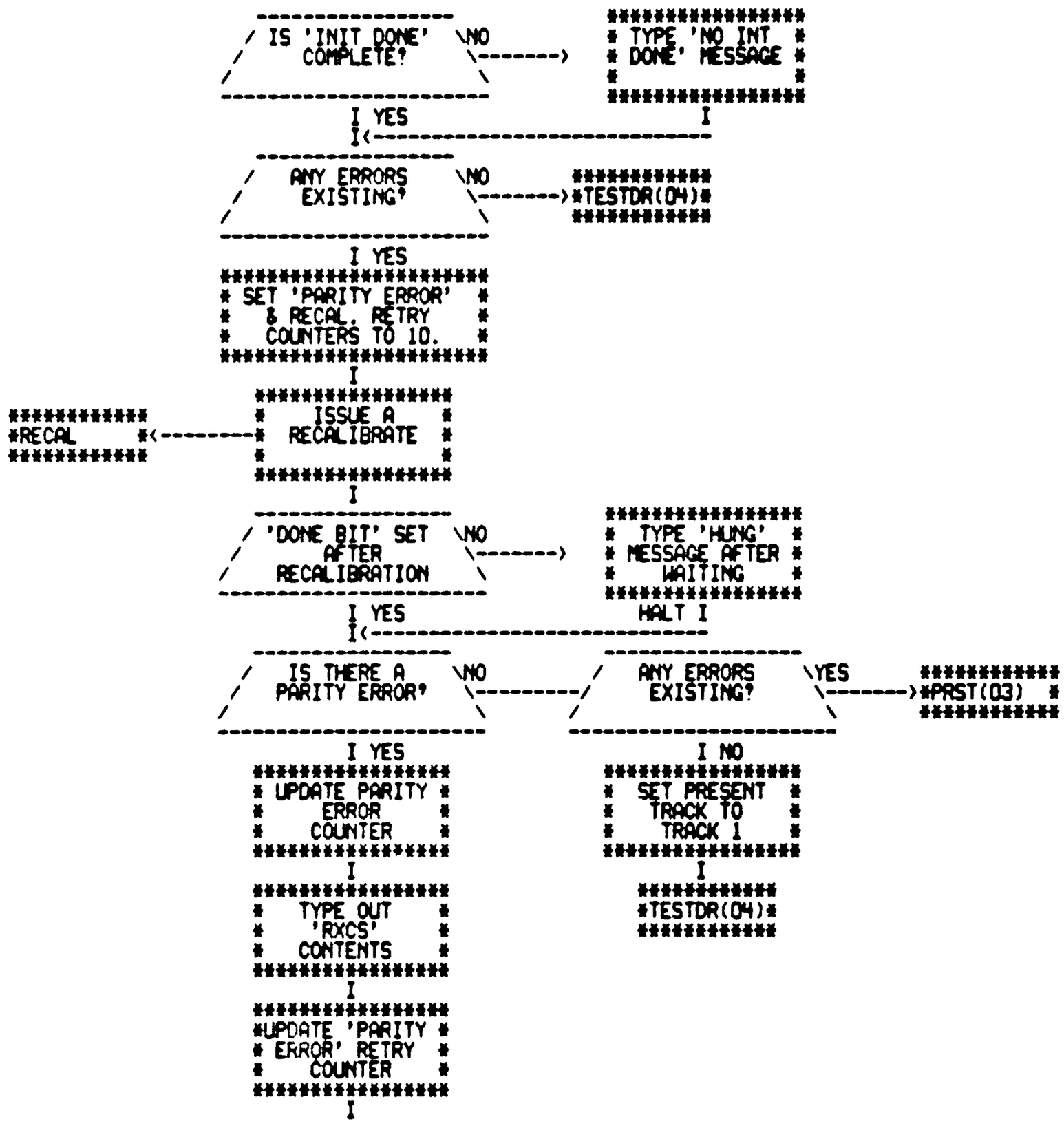
FLOW CHART
****

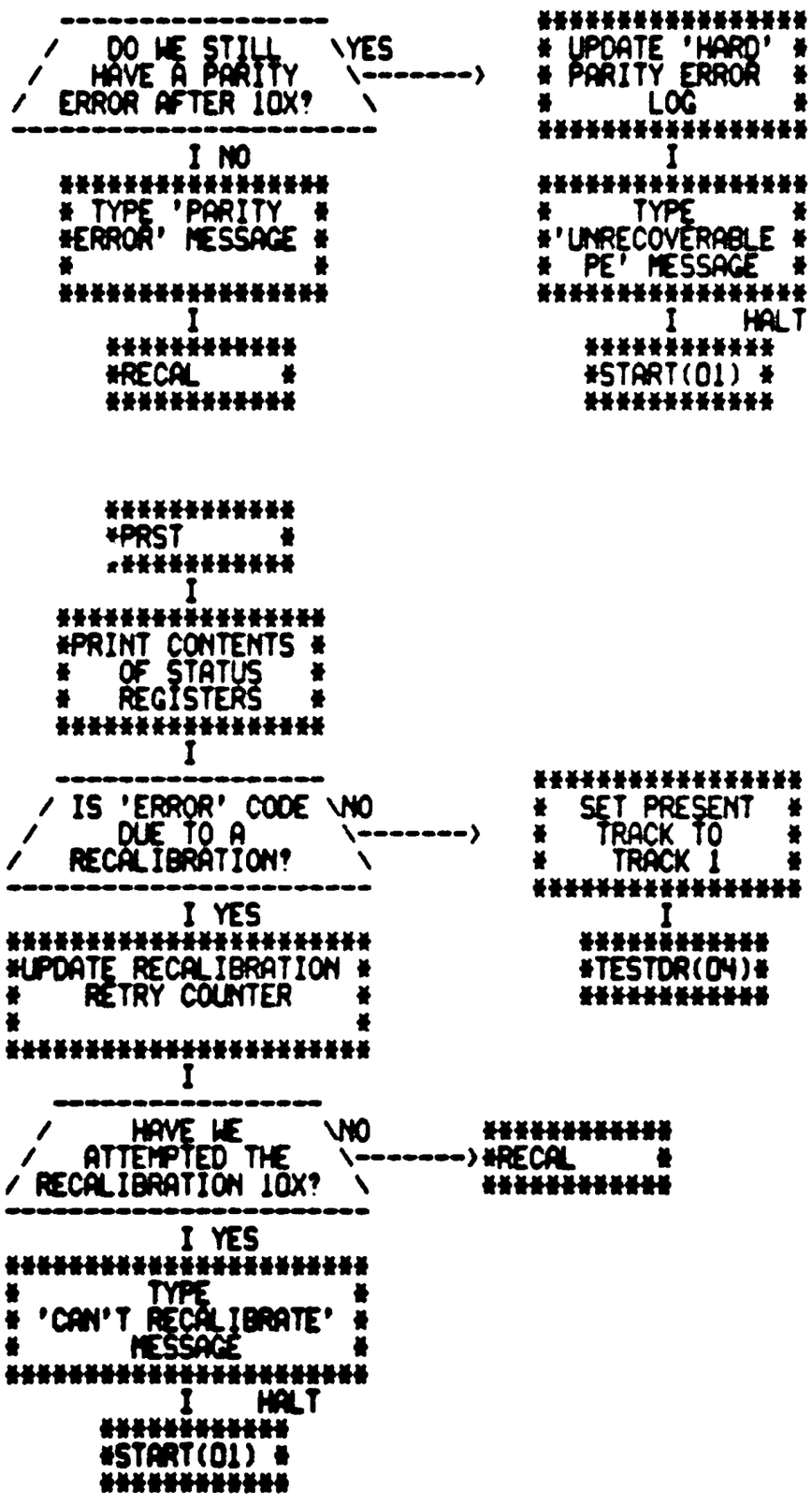
RX11 RELIABILITY TEST

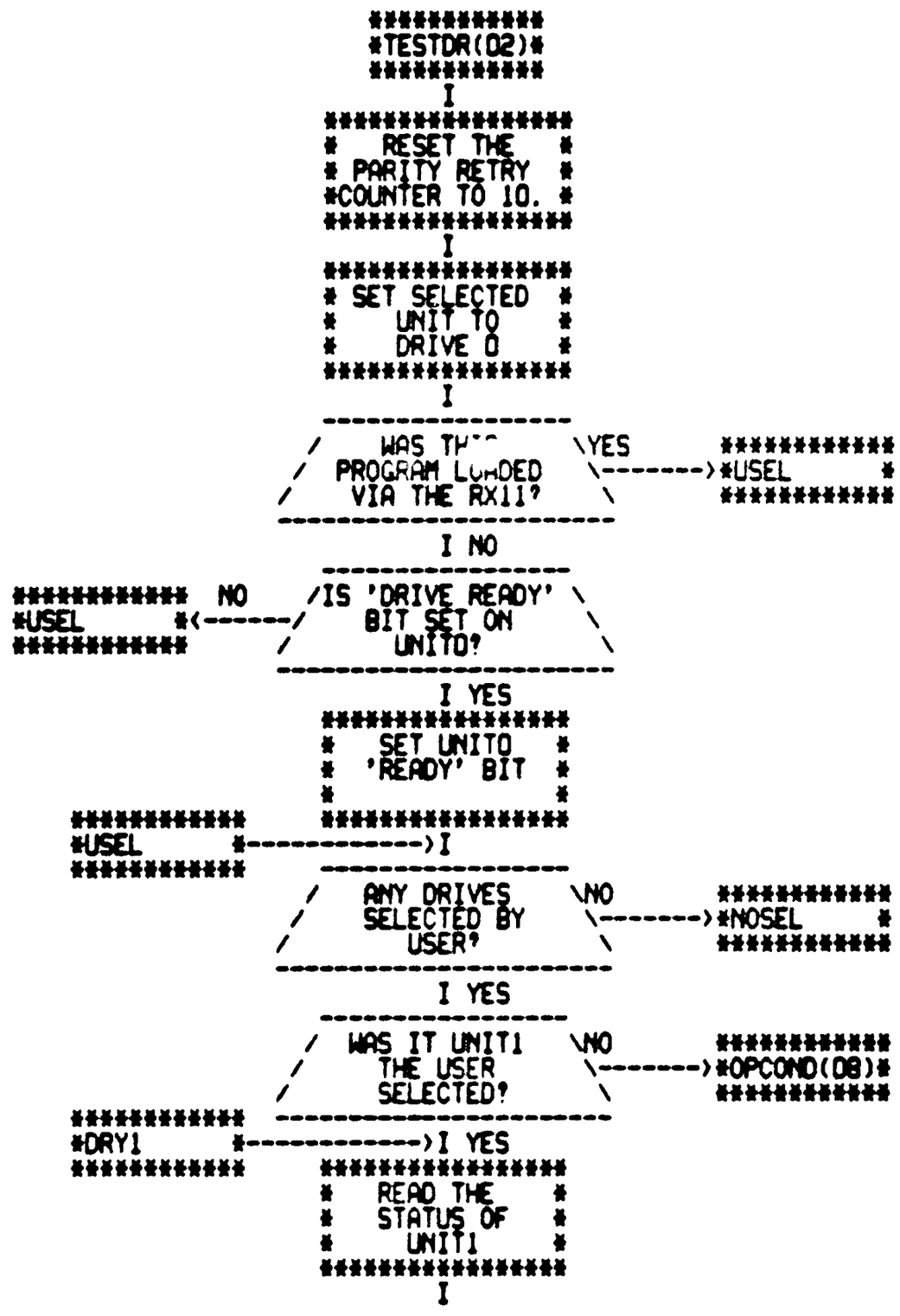
****

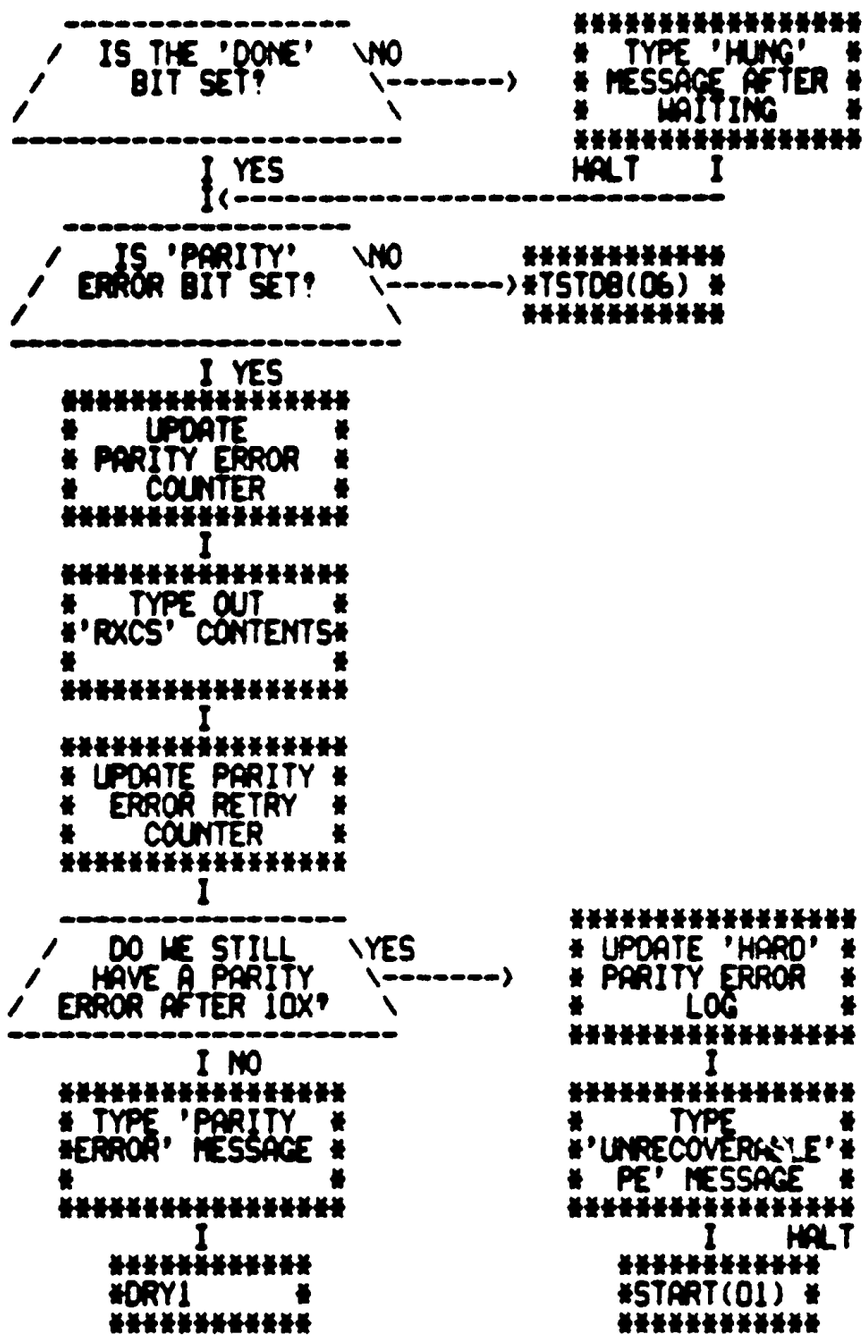
TABLE OF CONTENTS
****

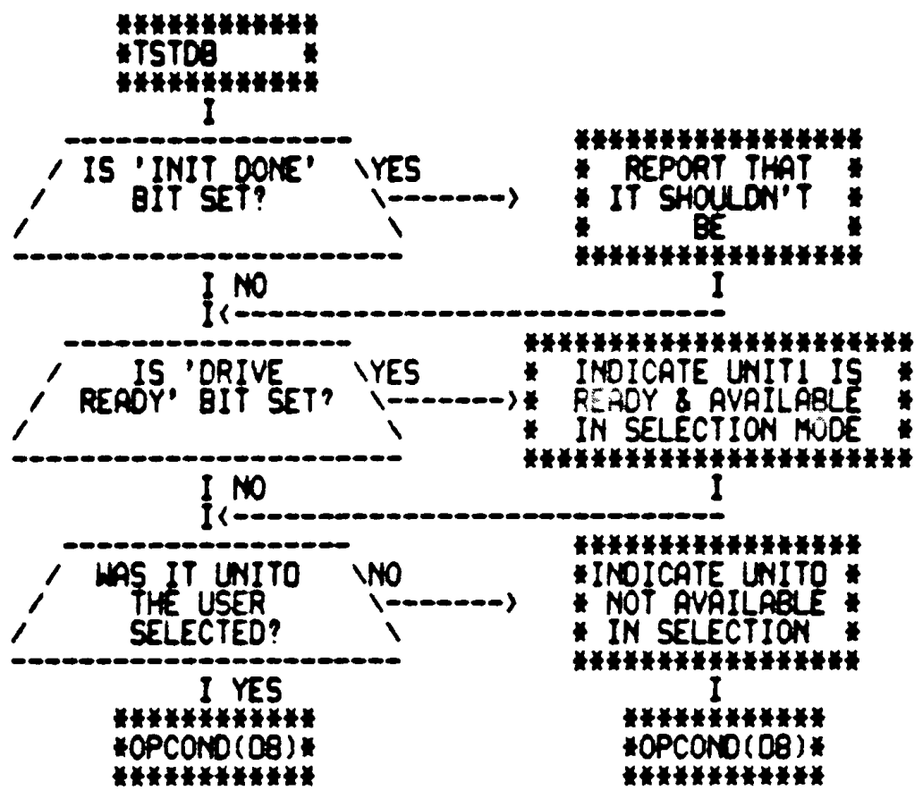

START(03)
INITIALIZE DEVICE VECTORS 264 & 266
INITIALIZE ALL COUNTERS & FLAGS
ASK USER TO SELECT DRIVE AND TEST CONDITIONS
'HALT'
WAIT FOR USER TO REPLY
TYPE: MAINDEC NAME DEVICE 'CSR' & 'DBR' DEVICE VECTOR ADD.
ASK USER TO SELECT OPERATING CONDITIONS
'HALT'
WAIT FOR USER TO REPLY
'DONE' BIT SET AFTER RECALIBRATION?
NO
TYPE 'HUNG' MESSAGE AFTER WAITING
HALT I
YES

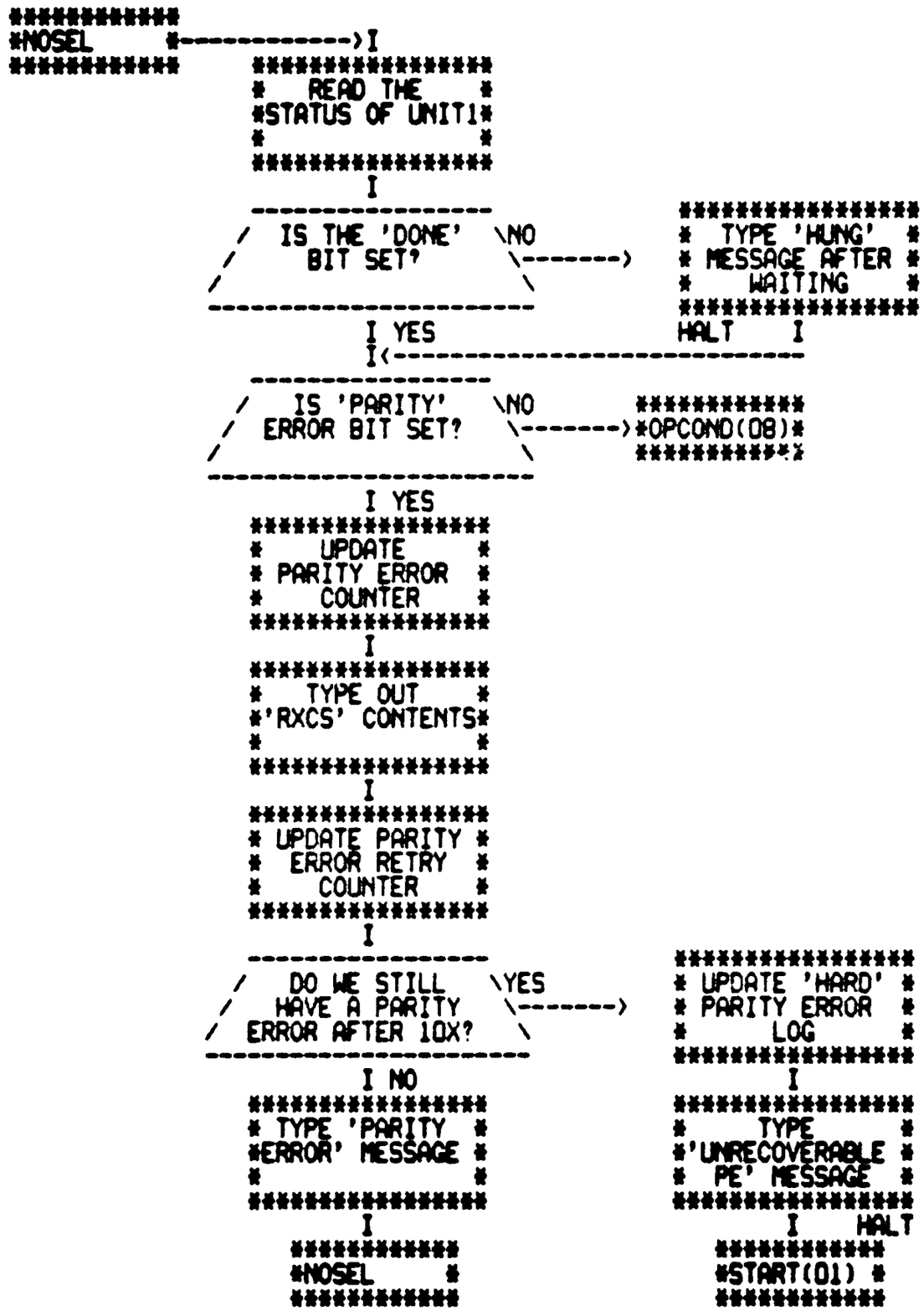

IS 'INIT DONE' COMPLETE?
NO
TYPE 'NO INT DONE' MESSAGE
YES
ANY ERRORS EXISTING?
NO
TESTDR(04)
YES
SET 'PARITY ERROR' & RECAL. RETRY COUNTERS TO 10.
ISSUE A RECALIBRATE
RECAL
'DONE BIT' SET AFTER RECALIBRATION
NO
TYPE 'HUNG' MESSAGE AFTER WAITING
HALT
YES
IS THERE A PARITY ERROR?
NO
ANY ERRORS EXISTING?
YES
PRST(03)
YES
UPDATE PARITY ERROR COUNTER
TYPE OUT 'RXCS' CONTENTS
UPDATE 'PARITY ERROR' RETRY COUNTER
NO
SET PRESENT TRACK TO TRACK 1
TESTDR(04)

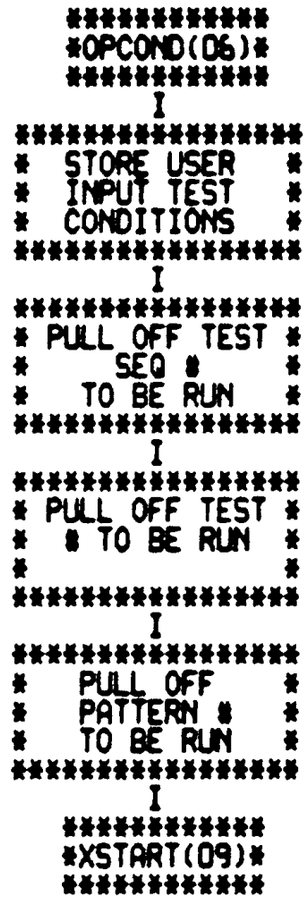

DO WE STILL HAVE A PARITY ERROR AFTER 10X?
YES
NO
UPDATE 'HARD' PARITY ERROR LOG
TYPE 'UNRECOVERABLE PE' MESSAGE
HALT
START(01)
TYPE 'PARITY ERROR' MESSAGE
RECAL
PRST
PRINT CONTENTS OF STATUS REGISTERS
IS 'ERROR' CODE DUE TO A RECALIBRATION?
NO
SET PRESENT TRACK TO TRACK 1
TESTDR(04)
YES
UPDATE RECALIBRATION RETRY COUNTER
HAVE WE ATTEMPTED THE RECALIBRATION 10X?
NO
RECAL
YES
TYPE 'CAN'T RECALIBRATE' MESSAGE
HALT
START(01)

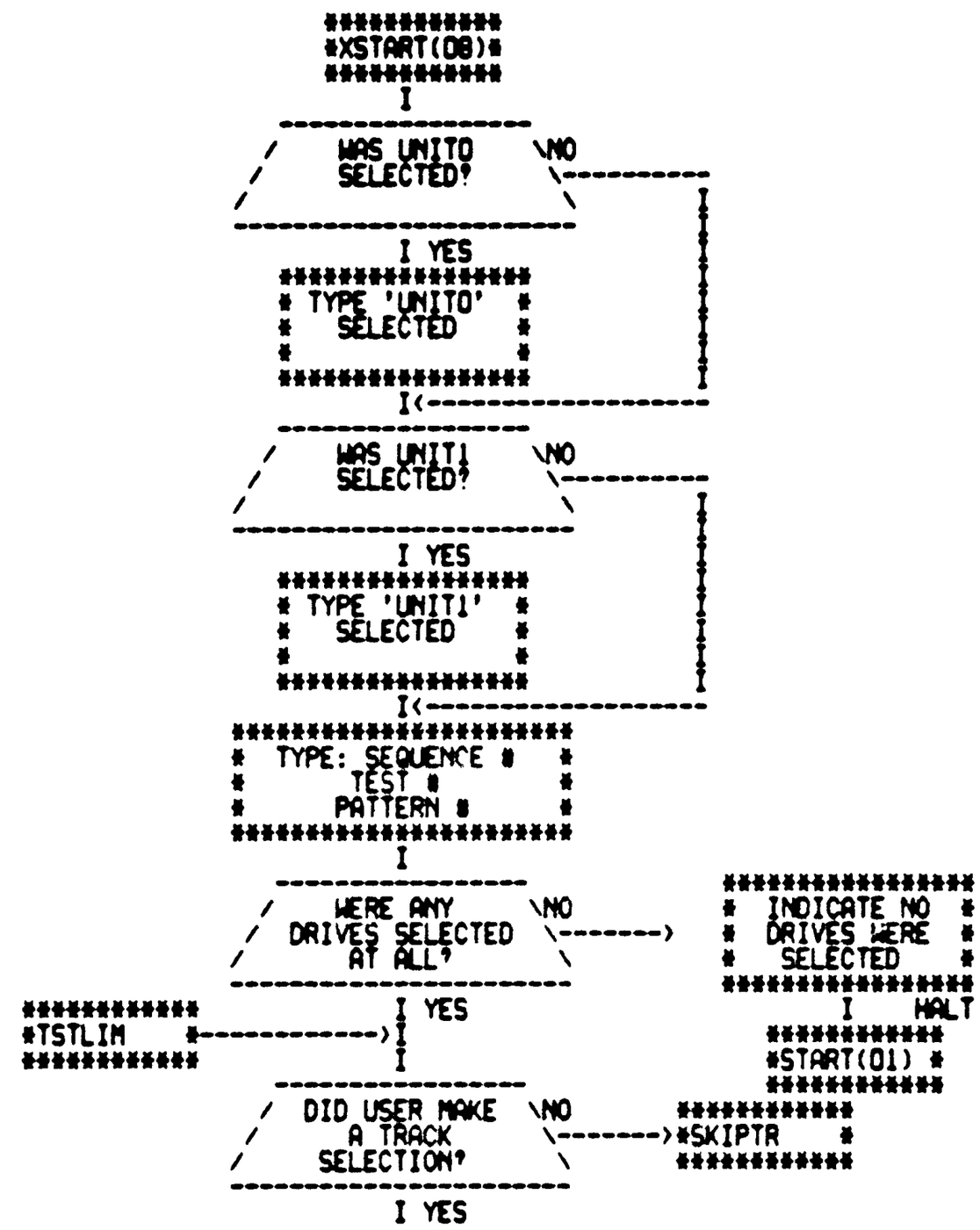

TESTDR(02)
RESET THE PARITY RETRY COUNTER TO 10.
SET SELECTED UNIT TO DRIVE 0
WAS THE PROGRAM LOADED VIA THE RX11?
YES
USEL
NO
IS 'DRIVE READY' BIT SET ON UNIT0?
NO
USEL
YES
SET UNIT0 'READY' BIT
USEL
ANY DRIVES SELECTED BY USER?
NO
NOSEL
YES
WAS IT UNIT1 THE USER SELECTED?
NO
OPCOND(08)
YES
DRY1
READ THE STATUS OF UNIT1

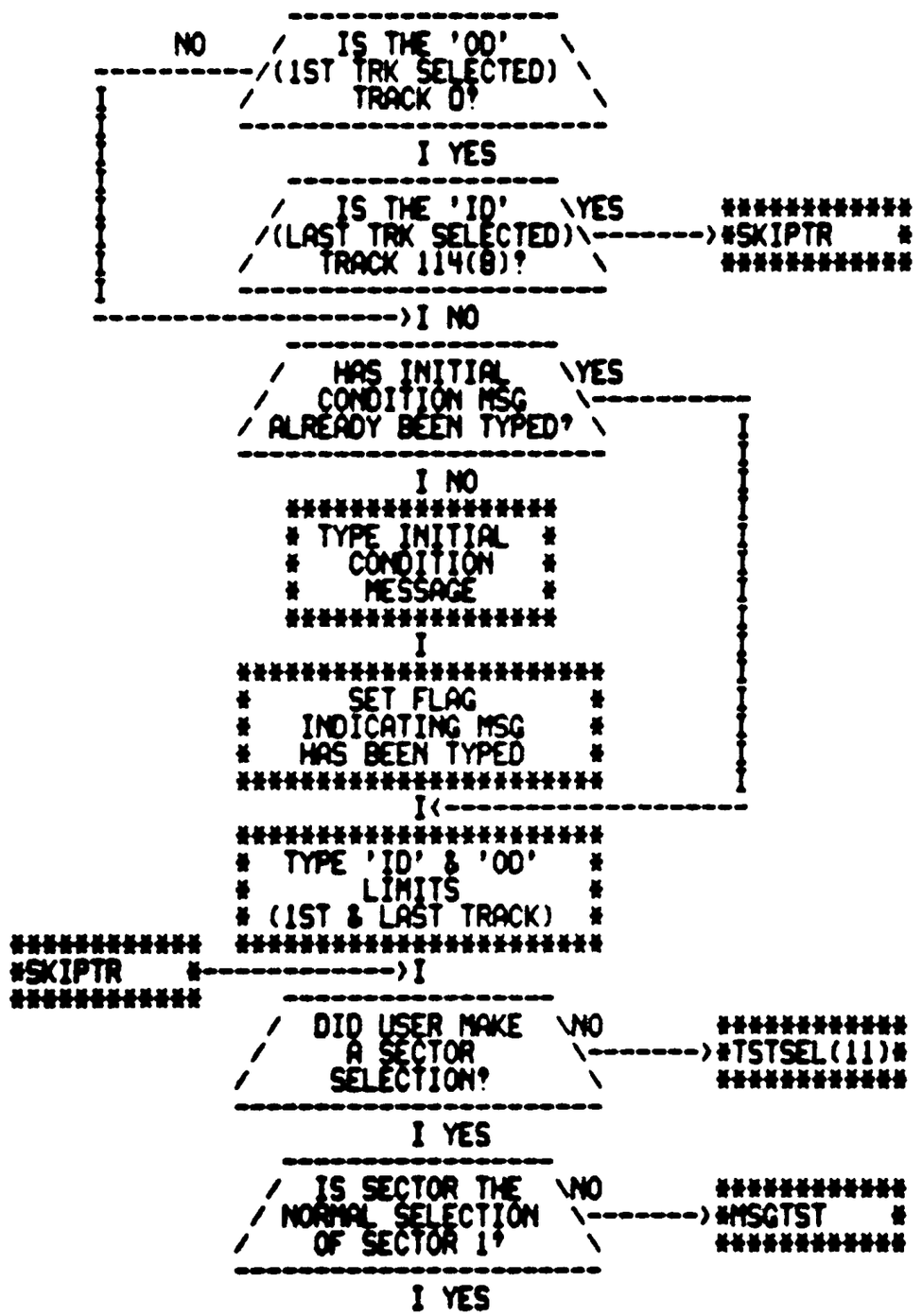

IS THE 'DONE' BIT SET?
NO
TYPE 'HUNG' MESSAGE AFTER WAITING
HALT
YES
IS 'PARITY' ERROR BIT SET?
NO
TSTD8(06)
YES
UPDATE PARITY ERROR COUNTER
TYPE OUT 'RXCS' CONTENTS
UPDATE PARITY ERROR RETRY COUNTER
DO WE STILL HAVE A PARITY ERROR AFTER 10X?
YES
UPDATE 'HARD' PARITY ERROR LOG
TYPE 'UNRECOVERABLE' PE' MESSAGE
HALT
NO
TYPE 'PARITY ERROR' MESSAGE
DRY1
START(01)

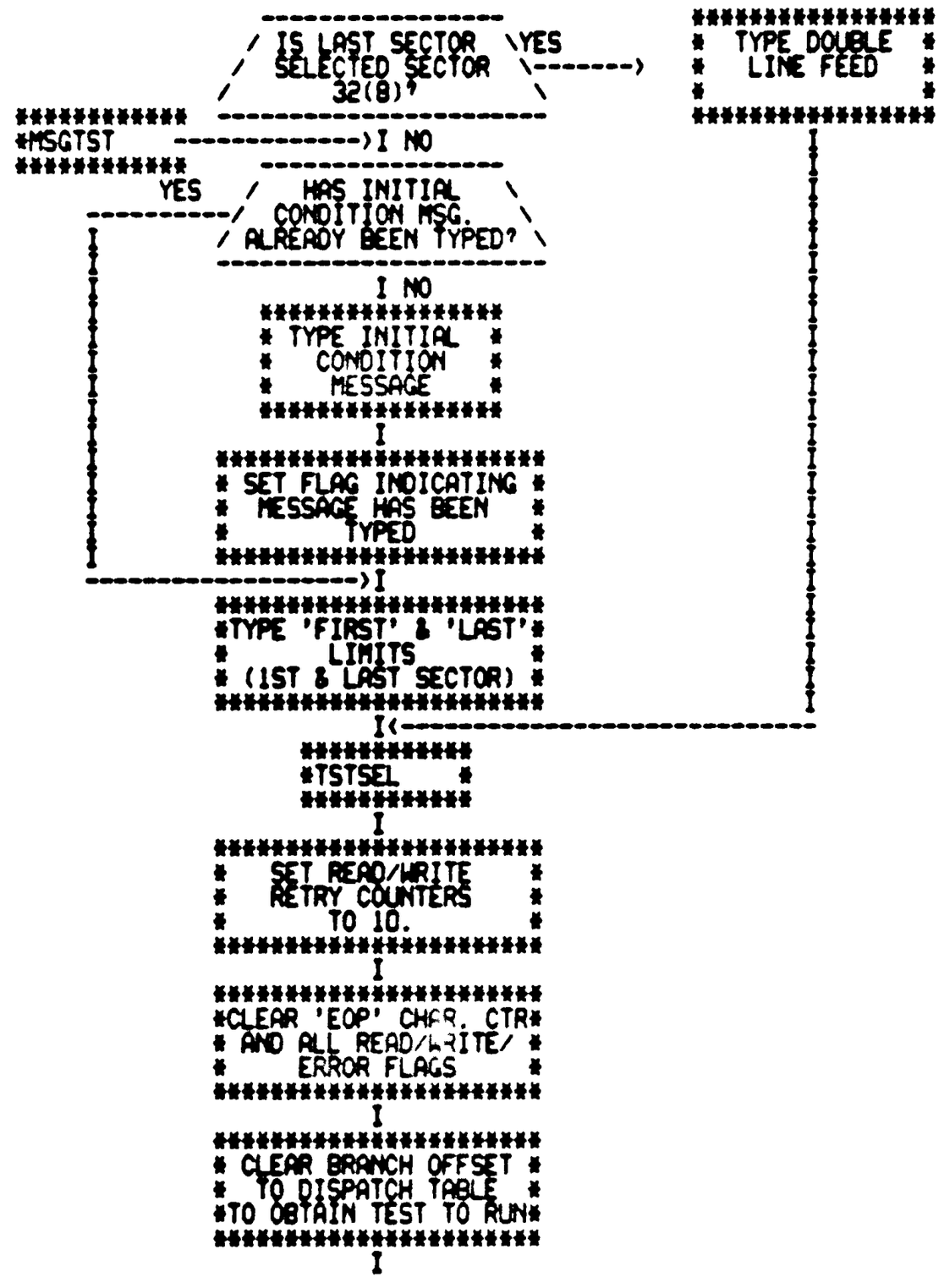

*TSTDB *
IS 'INIT DONE' BIT SET?
YES
REPORT THAT IT SHOULDN'T BE
NO
IS 'DRIVE READY' BIT SET?
YES
INDICATE UNIT1 IS READY & AVAILABLE IN SELECTION MODE
NO
WAS IT UNIT0 THE USER SELECTED?
NO
INDICATE UNIT0 NOT AVAILABLE IN SELECTION
YES
OPCOND(DB)
OPCOND(DB)

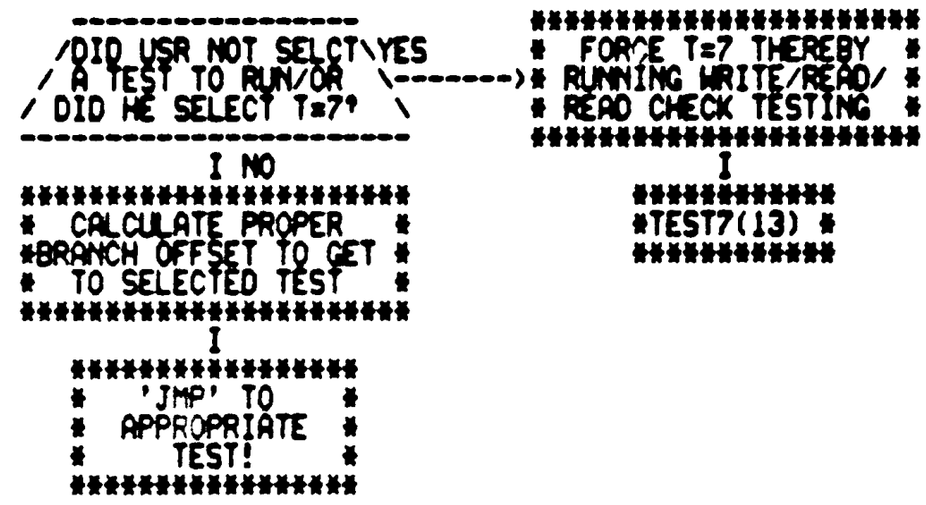

#NOSEL
READ THE STATUS OF UNIT1
IS THE 'DONE' BIT SET?
NO
TYPE 'HUNG' MESSAGE AFTER WAITING
HALT
YES
IS 'PARITY' ERROR BIT SET?
NO
#OPCOND(08)#
YES
UPDATE PARITY ERROR COUNTER
TYPE OUT 'RXCS' CONTENTS
UPDATE PARITY ERROR RETRY COUNTER
DO WE STILL HAVE A PARITY ERROR AFTER 10X?
YES
UPDATE 'HARD' PARITY ERROR LOG
TYPE 'UNRECOVERABLE PE' MESSAGE
HALT
#START(01) #
NO
TYPE 'PARITY ERROR' MESSAGE
#NOSEL


OPCOND(06)
STORE USER INPUT TEST CONDITIONS
PULL OFF TEST SEQ # TO BE RUN
PULL OFF TEST # TO BE RUN
PULL OFF PATTERN # TO BE RUN
XSTART(09)


XSTART(08)
WAS UNIT0 SELECTED?
NO
YES
TYPE 'UNIT0' SELECTED
WAS UNIT1 SELECTED?
NO
YES
TYPE 'UNIT1' SELECTED
TYPE: SEQUENCE # TEST # PATTERN #
WERE ANY DRIVES SELECTED AT ALL?
NO
INDICATE NO DRIVES WERE SELECTED
HALT
START(01)
TSTLIM
YES
DID USER MAKE A TRACK SELECTION?
NO
SKIPTR
YES

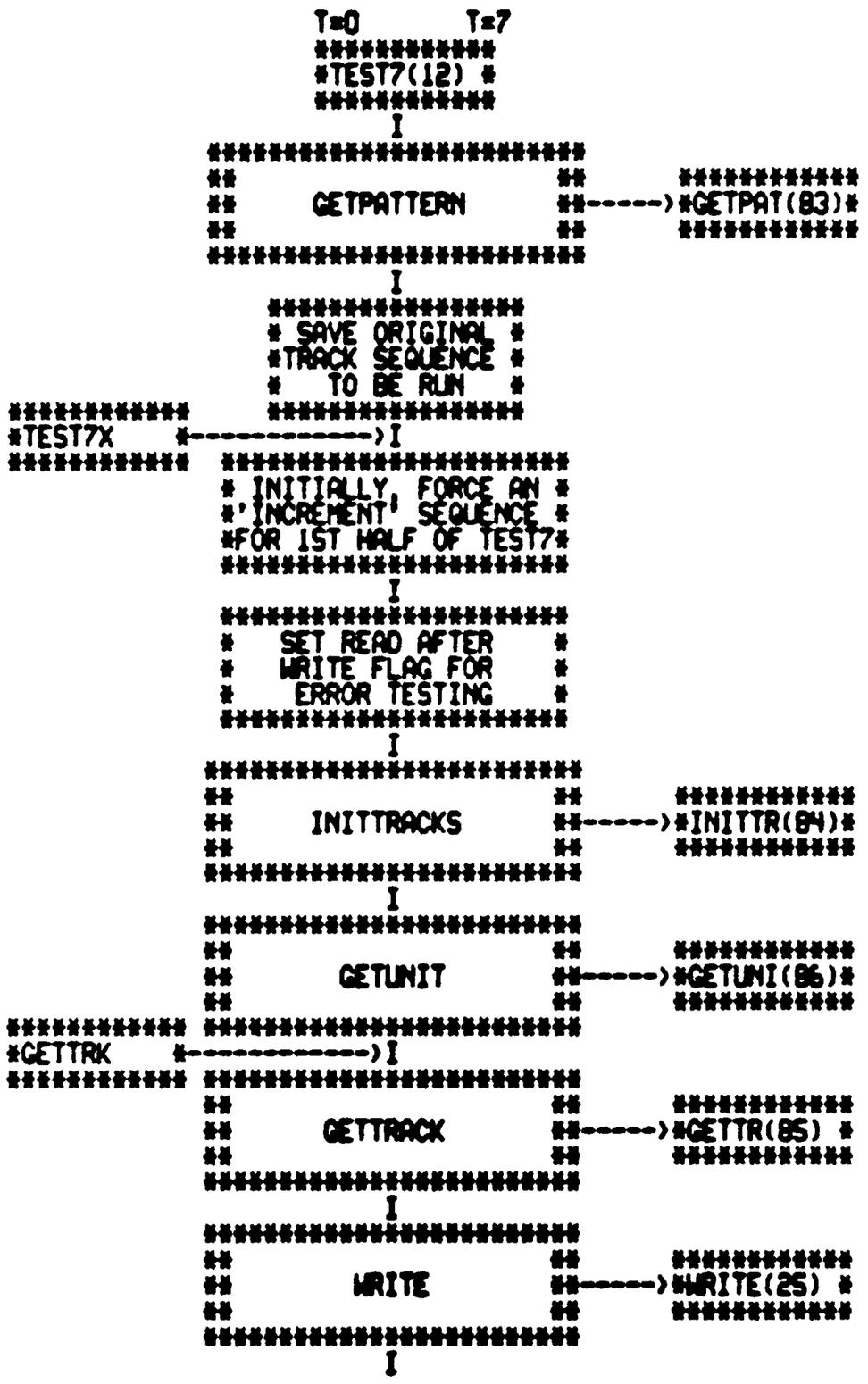

IS THE 'OD' (1ST TRK SELECTED) TRACK 0?
NO
I YES
IS THE 'ID' (LAST TRK SELECTED) TRACK 114(8)?
YES
#SKIPTR
I NO
HAS INITIAL CONDITION MSG ALREADY BEEN TYPED?
YES
I NO
TYPE INITIAL CONDITION MESSAGE
SET FLAG INDICATING MSG HAS BEEN TYPED
TYPE 'ID' & 'OD' LIMITS (1ST & LAST TRACK)
#SKIPTR
DID USER MAKE A SECTOR SELECTION?
NO
#TSTSEL(11)#
I YES
IS SECTOR THE NORMAL SELECTION OF SECTOR 1?
NO
#MSGTST
I YES

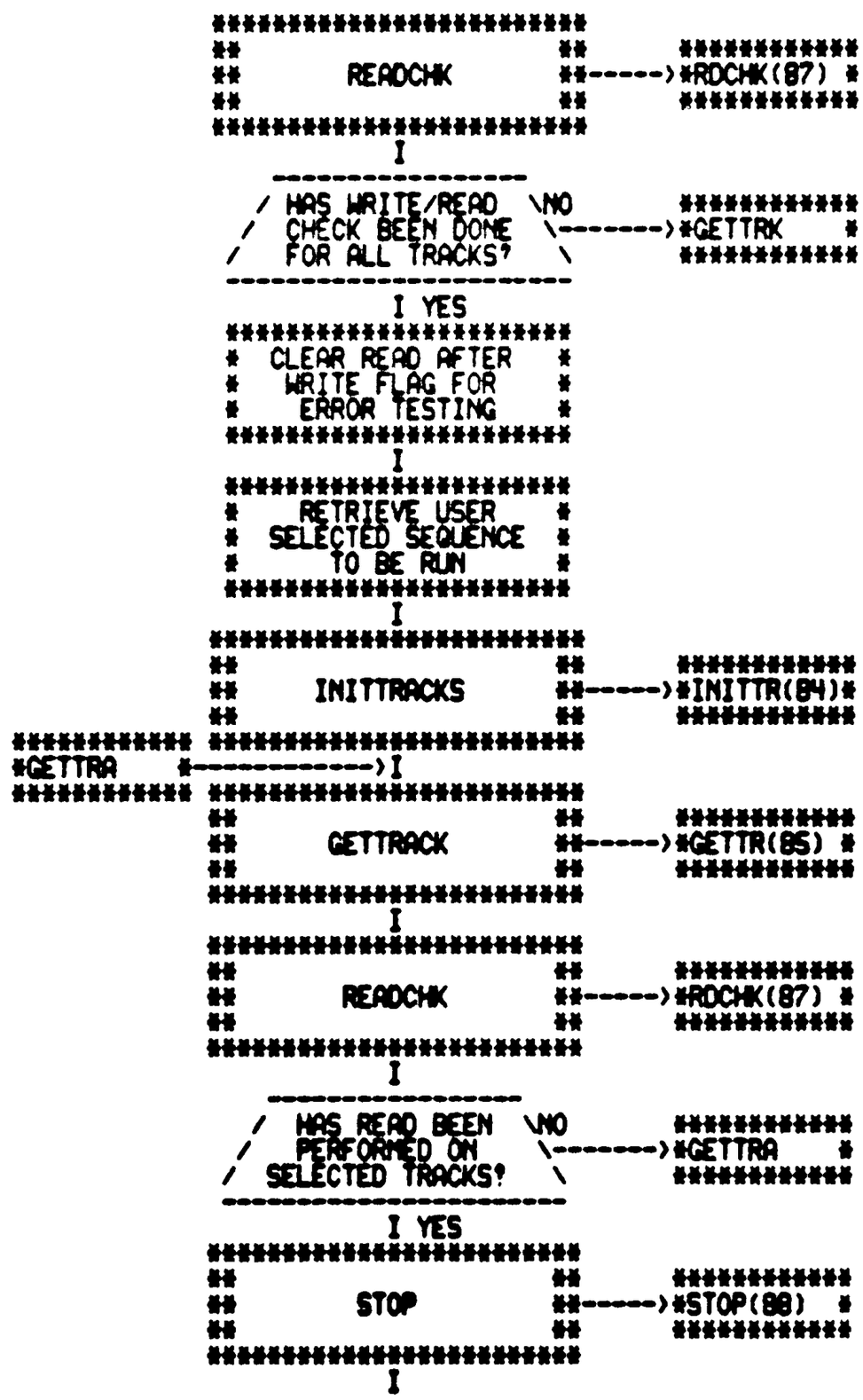

IS LAST SECTOR SELECTED SECTOR 32(8)?
YES
TYPE DOUBLE LINE FEED
*MSGTST
NO
HAS INITIAL CONDITION MSG. ALREADY BEEN TYPED?
NO
TYPE INITIAL CONDITION MESSAGE
SET FLAG INDICATING MESSAGE HAS BEEN TYPED
YES
*TYPE 'FIRST' & 'LAST' LIMITS (1ST & LAST SECTOR)
*TSTSEL
SET READ/WRITE RETRY COUNTERS TO 10.
CLEAR 'EOP' CHAR. CTR AND ALL READ/WRITE/ ERROR FLAGS
CLEAR BRANCH OFFSET TO DISPATCH TABLE
TO OBTAIN TEST TO RUN

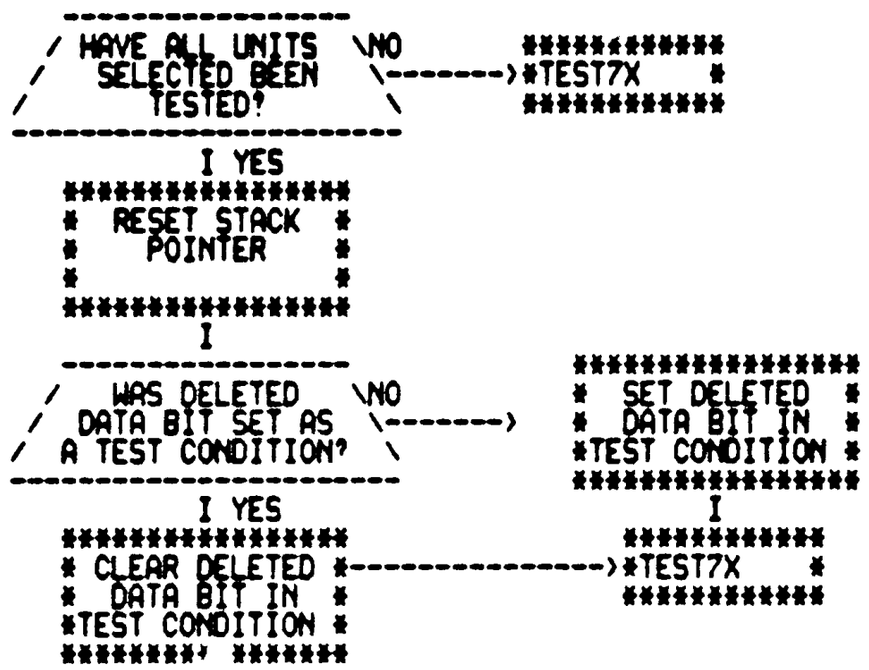


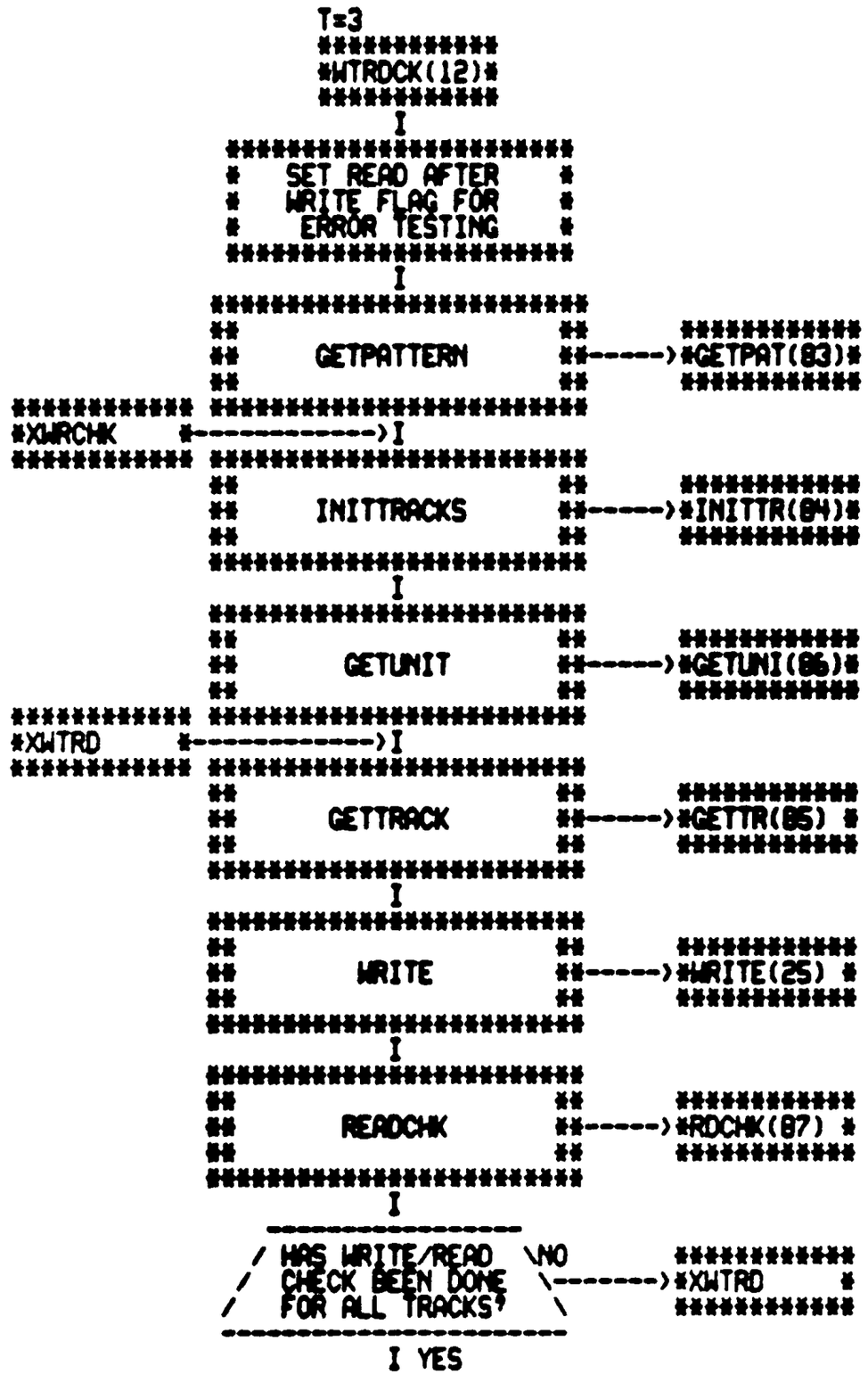
NOTE: THE APPROPRIATE TEST IS TO ONE
OF THE FOLLOWING:

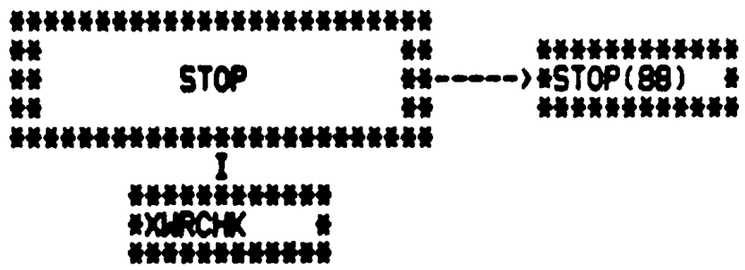
#TEST7(13) #	T=0/7;	WRITE/READ/READ CHECK
#WTRDCK(16)#	T=3 ;	WRITE/READ CHECK
#CKREAD(18)#	T=4 ;	READ CHECK ONLY
#WRTRD(19) #	T=2 ;	WRITE/READ
#WRTONL(21)#	T=1 ;	WRITE ONLY
#RDONLY(22)#	T=5 ;	READ ONLY
#DRVSWP(23)#	T=6 ;	WRITE/READ CHECK ON ALTERNATE DRIVES

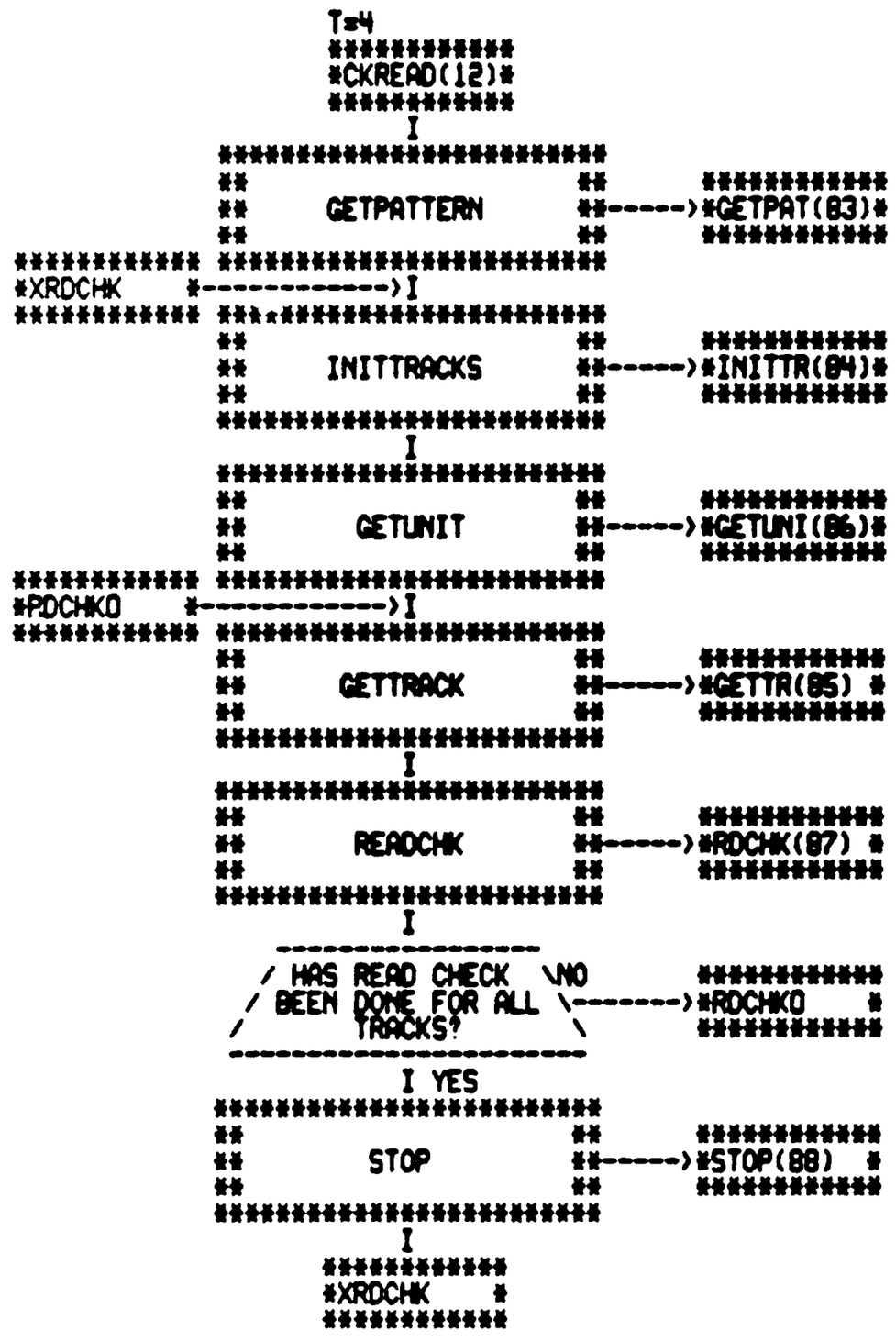

T=0
T=7
TEST7(12)
GETPATTERN
GETPAT(83)
SAVE ORIGINAL TRACK SEQUENCE TO BE RUN
TEST7X
INITIALLY, FORCE AN 'INCREMENT' SEQUENCE FOR 1ST HALF OF TEST7
SET READ AFTER WRITE FLAG FOR ERROR TESTING
INITTRACKS
INITTR(84)
GETUNIT
GETUNI(86)
GETTRK
GETTRACK
GETTR(85)
WRITE
WRITE(25)

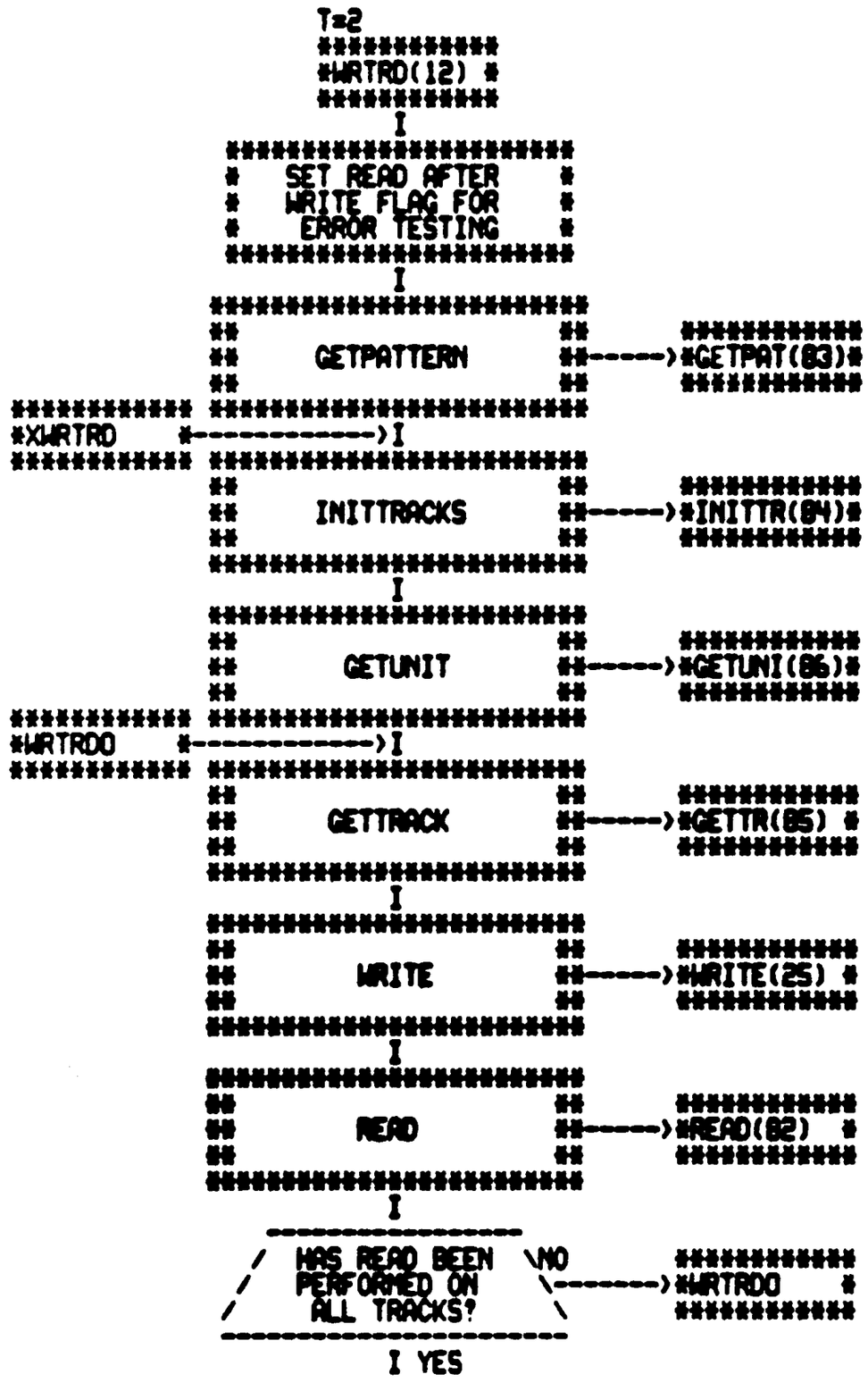

READCHK
RDCHK(87)
HAS WRITE/READ CHECK BEEN DONE FOR ALL TRACKS?
NO
GETTRK
YES
CLEAR READ AFTER WRITE FLAG FOR ERROR TESTING
RETRIEVE USER SELECTED SEQUENCE TO BE RUN
INITTRACKS
INITTR(84)
GETTRA
GETTRACK
GETTR(85)
READCHK
RDCHK(87)
HAS READ BEEN PERFORMED ON SELECTED TRACKS?
NO
GETTRA
YES
STOP
STOP(88)

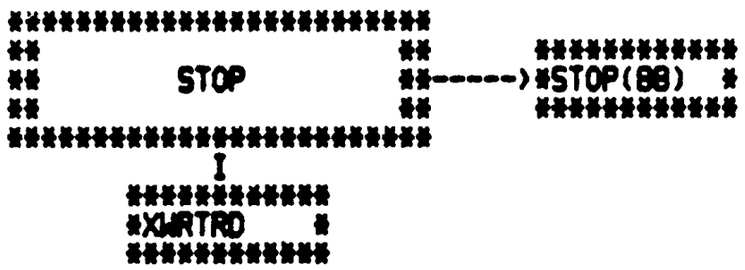

HAVE ALL UNITS SELECTED BEEN TESTED?
NO
*TEST7X *
YES
* RESET STACK POINTER *
WAS DELETED DATA BIT SET AS A TEST CONDITION?
NO
* SET DELETED DATA BIT IN *TEST CONDITION *
YES
* CLEAR DELETED DATA BIT IN *TEST CONDITION *
*TEST7X *

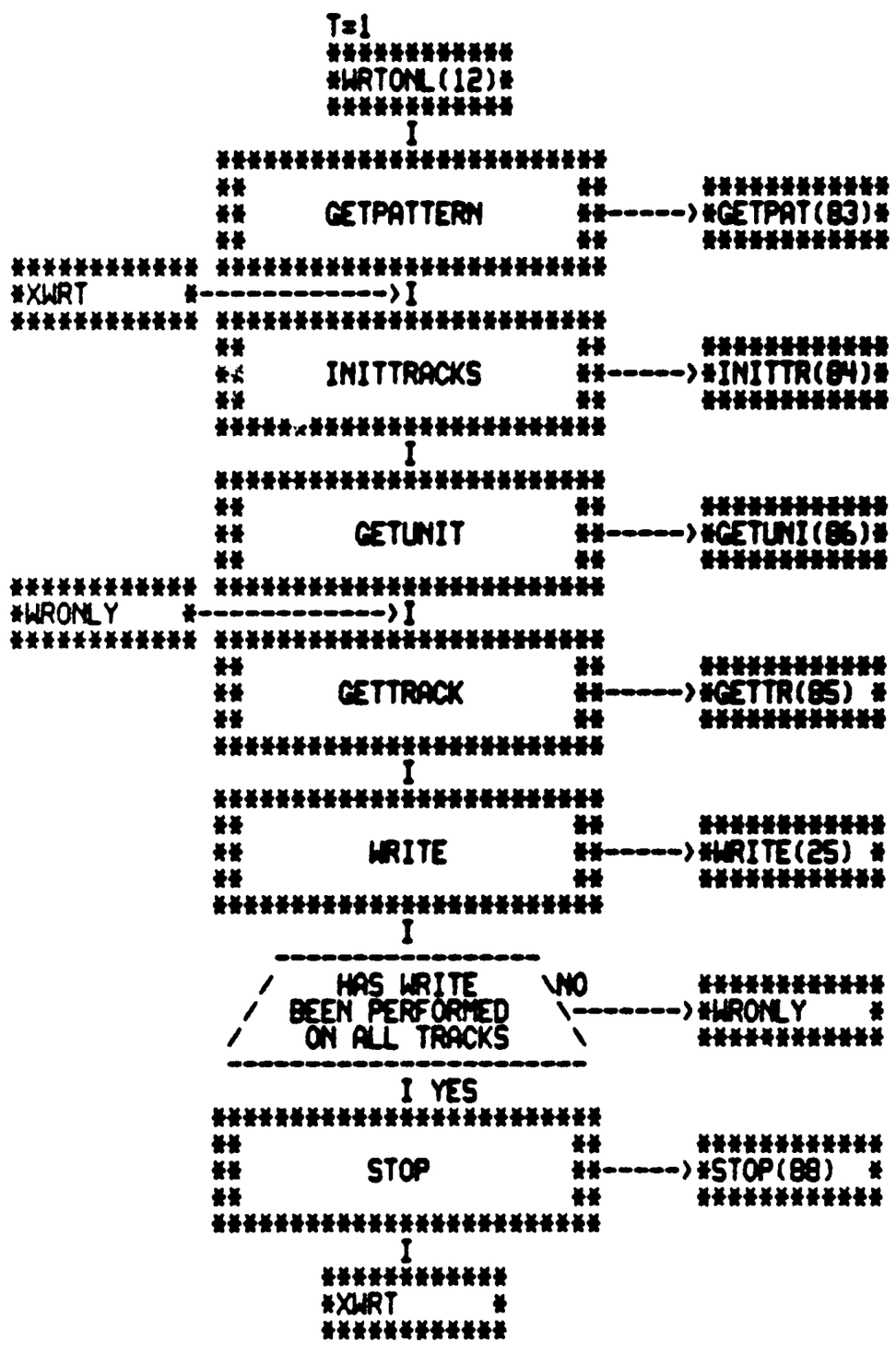

T=3
#WTRDCK(12)#
SET READ AFTER WRITE FLAG FOR ERROR TESTING
GETPATTERN
#GETPAT(83)#
#XWRCHK
INITTRACKS
#INITTR(84)#
GETUNIT
#GETUNI(86)#
#XWTRD
GETTRACK
#GETTR(85) #
WRITE
#WRITE(25) #
READCHK
#RDCHK(87) #
HAS WRITE/READ CHECK BEEN DONE FOR ALL TRACKS?
NO
#XWTRD #
YES

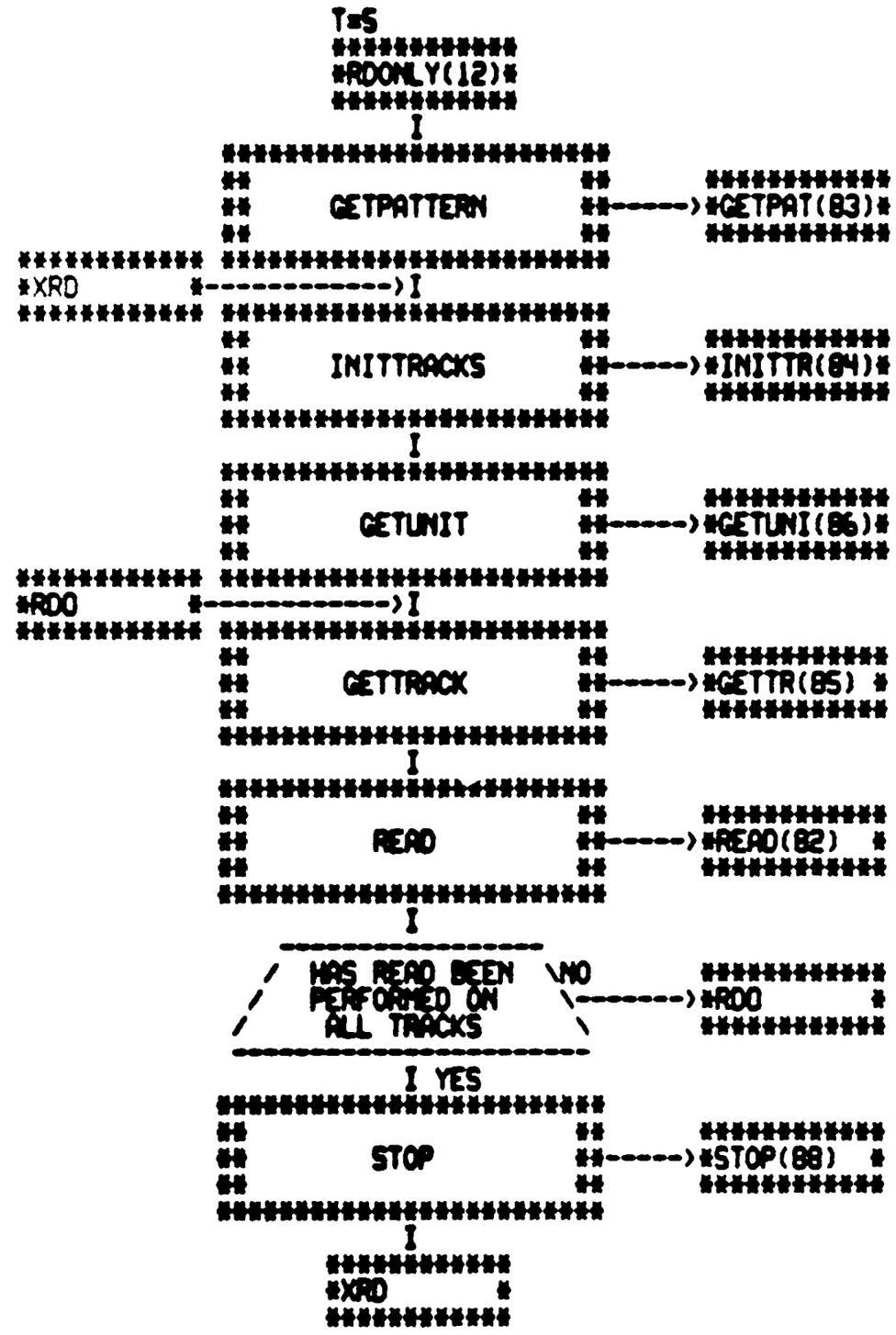

STOP
STOP(88)
XWRCHK

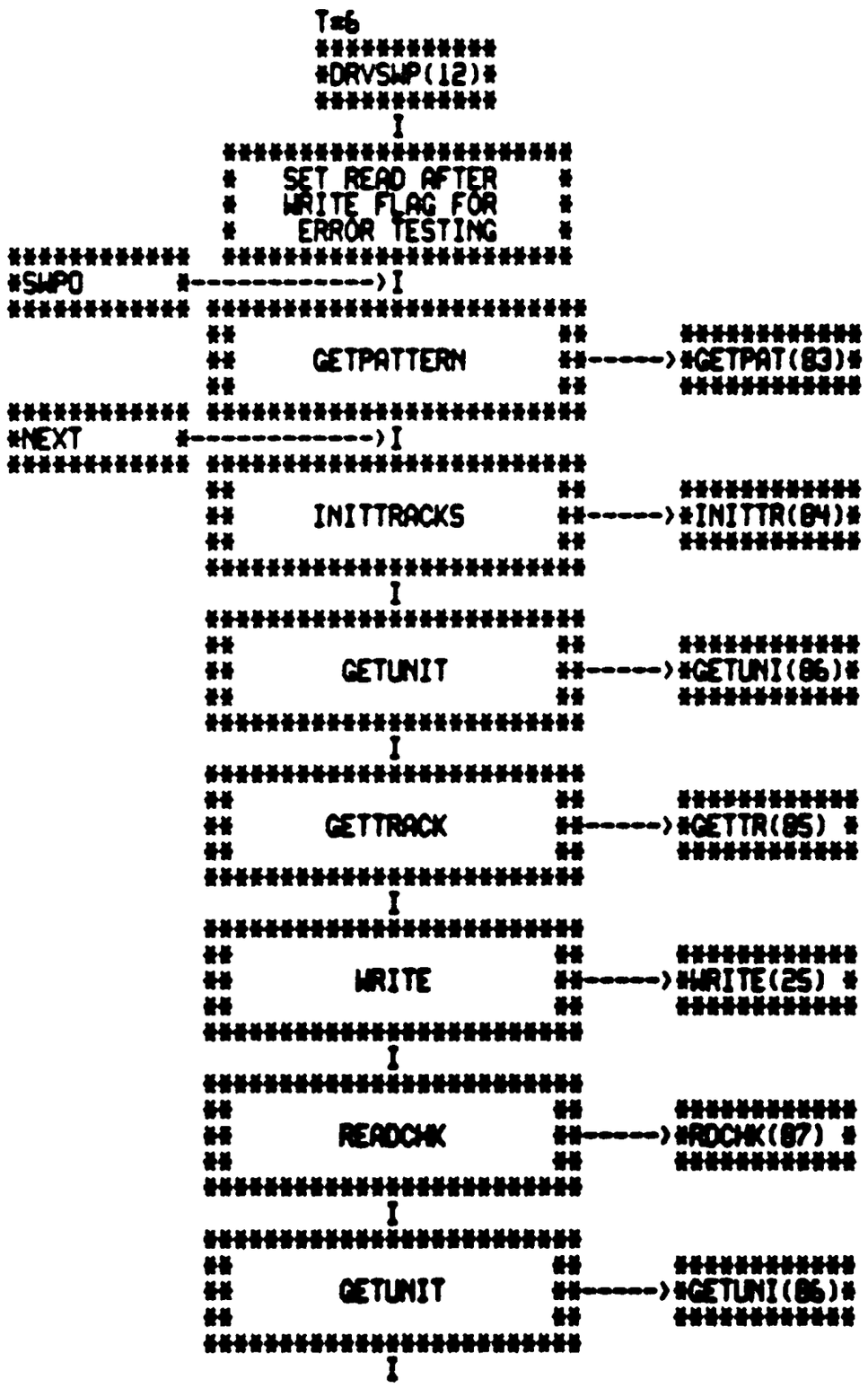

T=4
CKREAD(12)
GETPATTERN
GETPAT(83)
*XRDCHK
INITTRACKS
INITTR(84)
GETUNIT
GETUNI(86)
*PDCHKO
GETTRACK
*GETTR(85) *
READCHK
*RDCHK(87) *
HAS READ CHECK BEEN DONE FOR ALL TRACKS?
NO
*RDCHKO *
YES
STOP
*STOP(88) *
*XRDCHK *

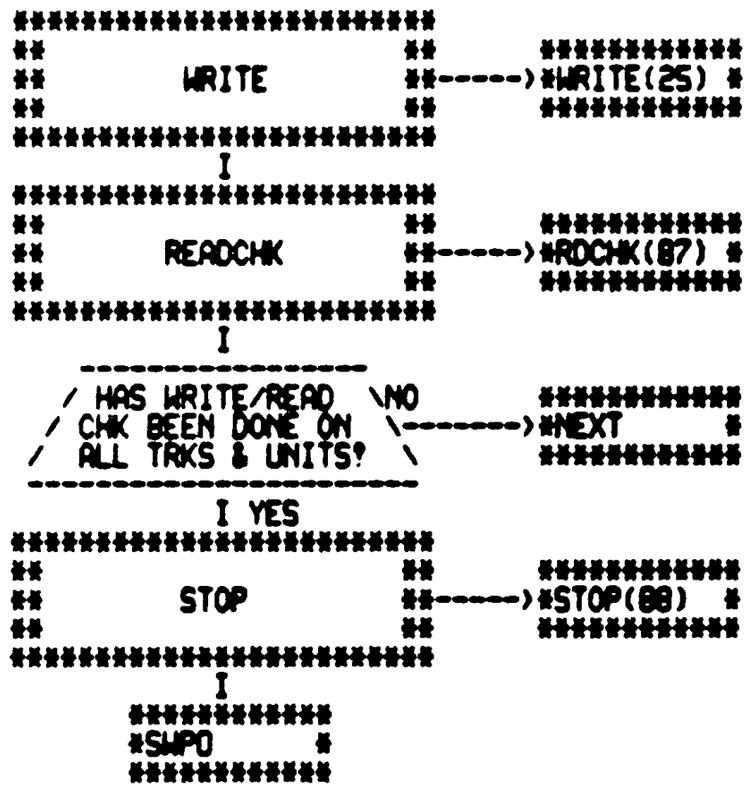

T=2
#WRTRD(12) #
SET READ AFTER WRITE FLAG FOR ERROR TESTING
GETPATTERN
#GETPAT(83)#
#XWRTRD #
INITTRACKS
#INITTR(84)#
GETUNIT
#GETUNI(86)#
#WRTRDO #
GETTRACK
#GETTR(85) #
WRITE
#WRITE(25) #
READ
#READ(82) #
HAS READ BEEN PERFORMED ON ALL TRACKS?
NO
#WRTRDO #
YES

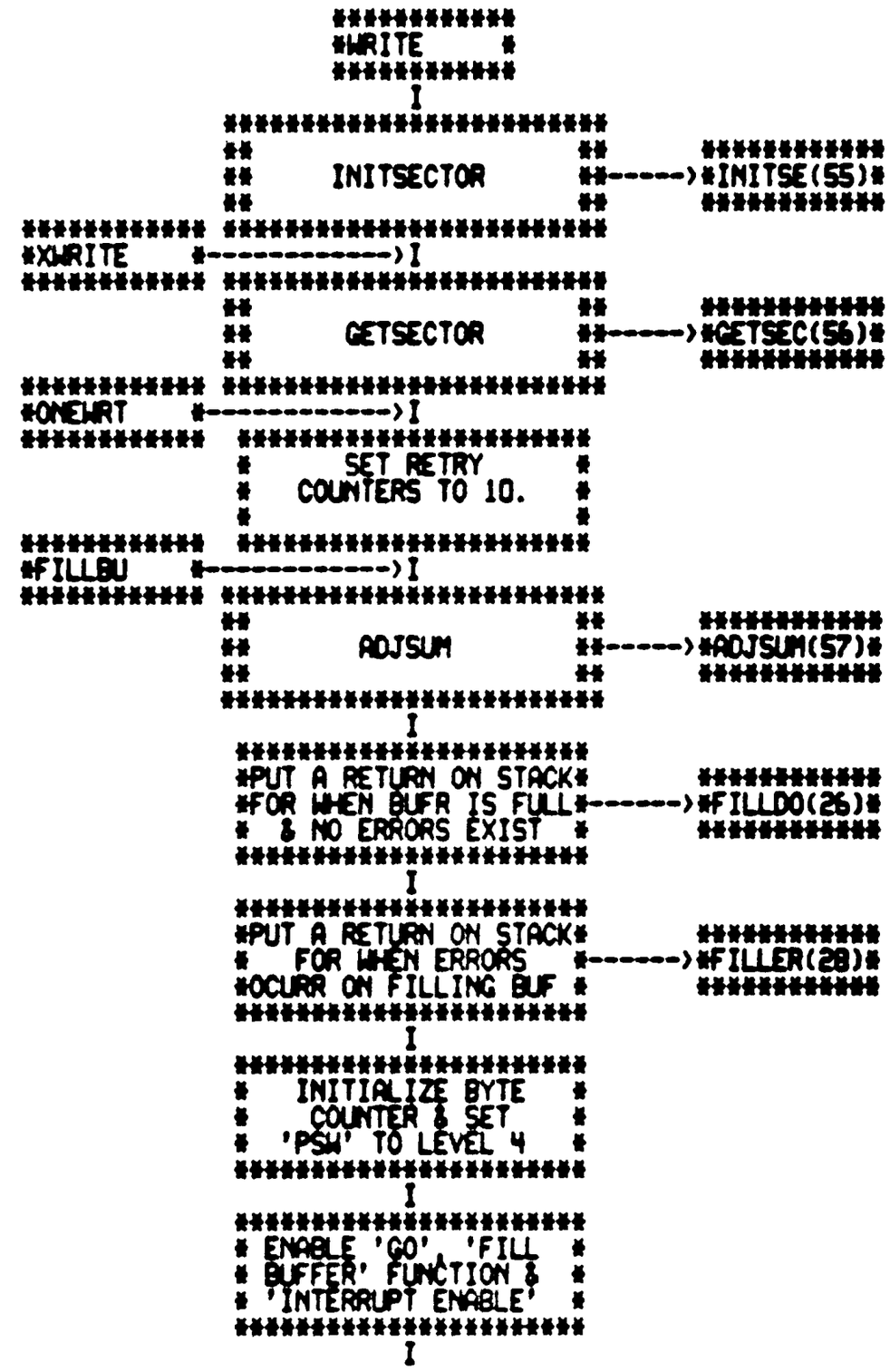

STOP
STOP(BB)
XWRTRD

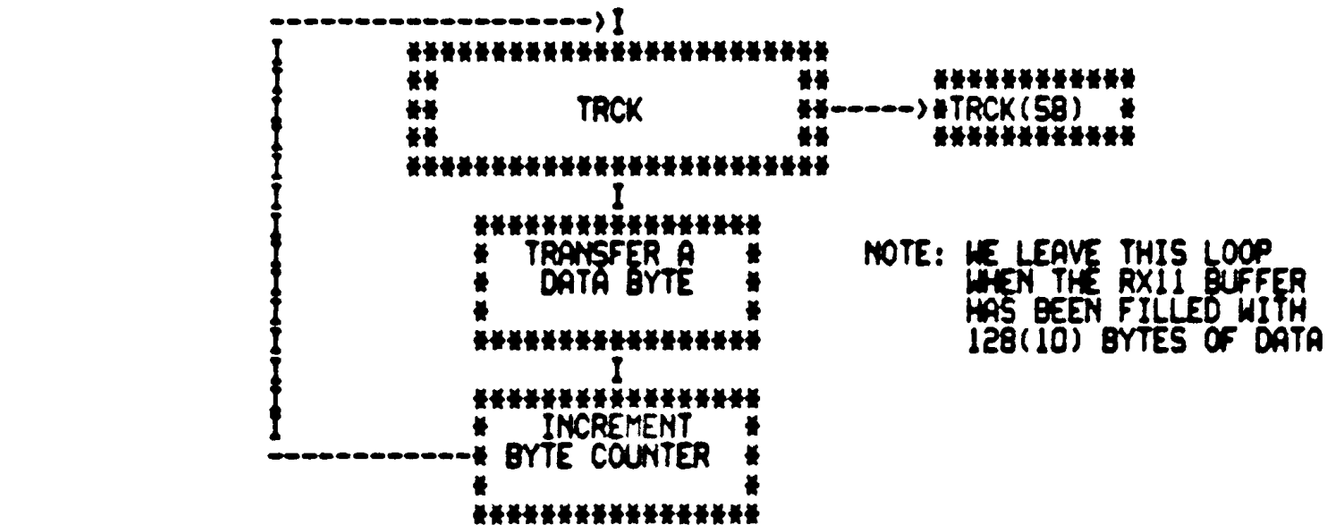

T=1
WRTONL(12)
GETPATTERN
GETPAT(83)
XWRT
INITTRACKS
INITTR(84)
GETUNIT
GETUNI(86)
WRONLY
GETTRACK
GETTR(85)
WRITE
WRITE(25)
HAS WRITE BEEN PERFORMED ON ALL TRACKS
NO
WRONLY
YES
STOP
STOP(88)
XWRT

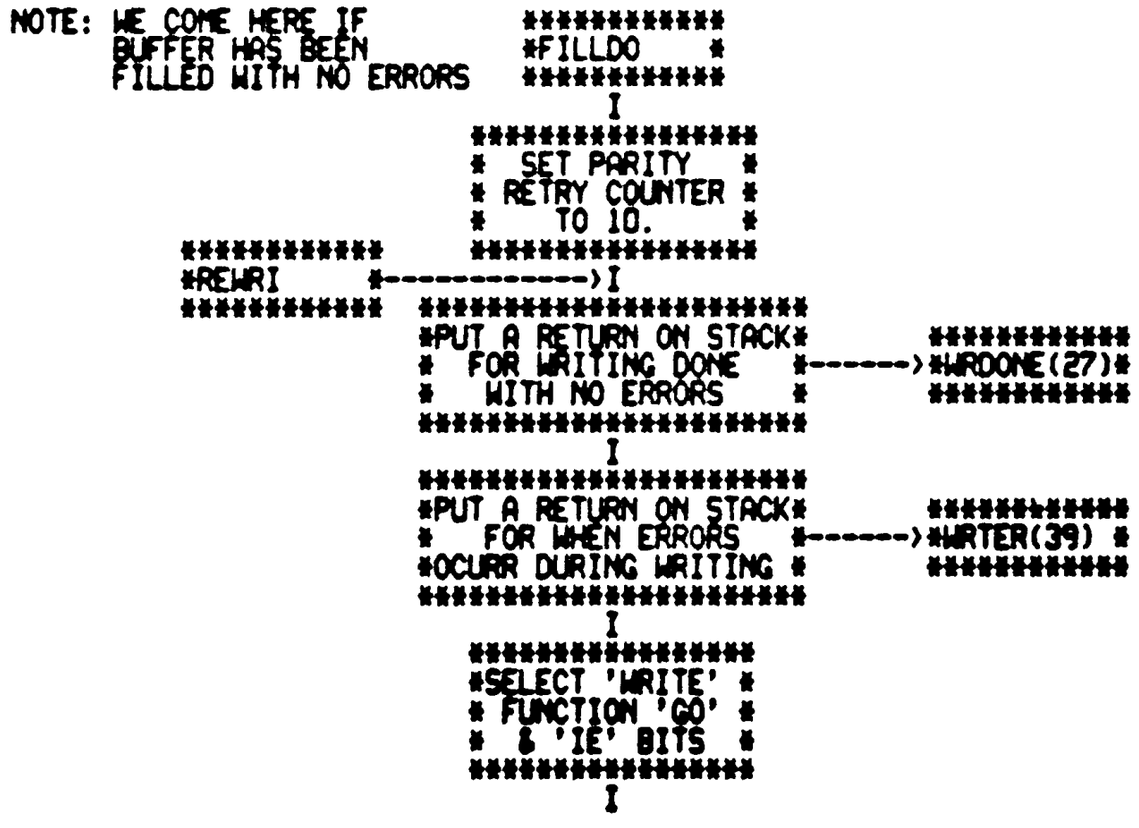

T=5
RDONLY(12)
GETPATTERN
GETPAT(83)
*XRD
INITTRACKS
INITTR(84)
GETUNIT
GETUNI(86)
*RDO
GETTRACK
*GETTR(85) *
READ
*READ(82) *
HAS READ BEEN PERFORMED ON ALL TRACKS
NO
*RDO *
YES
STOP
*STOP(88) *
*XRD *

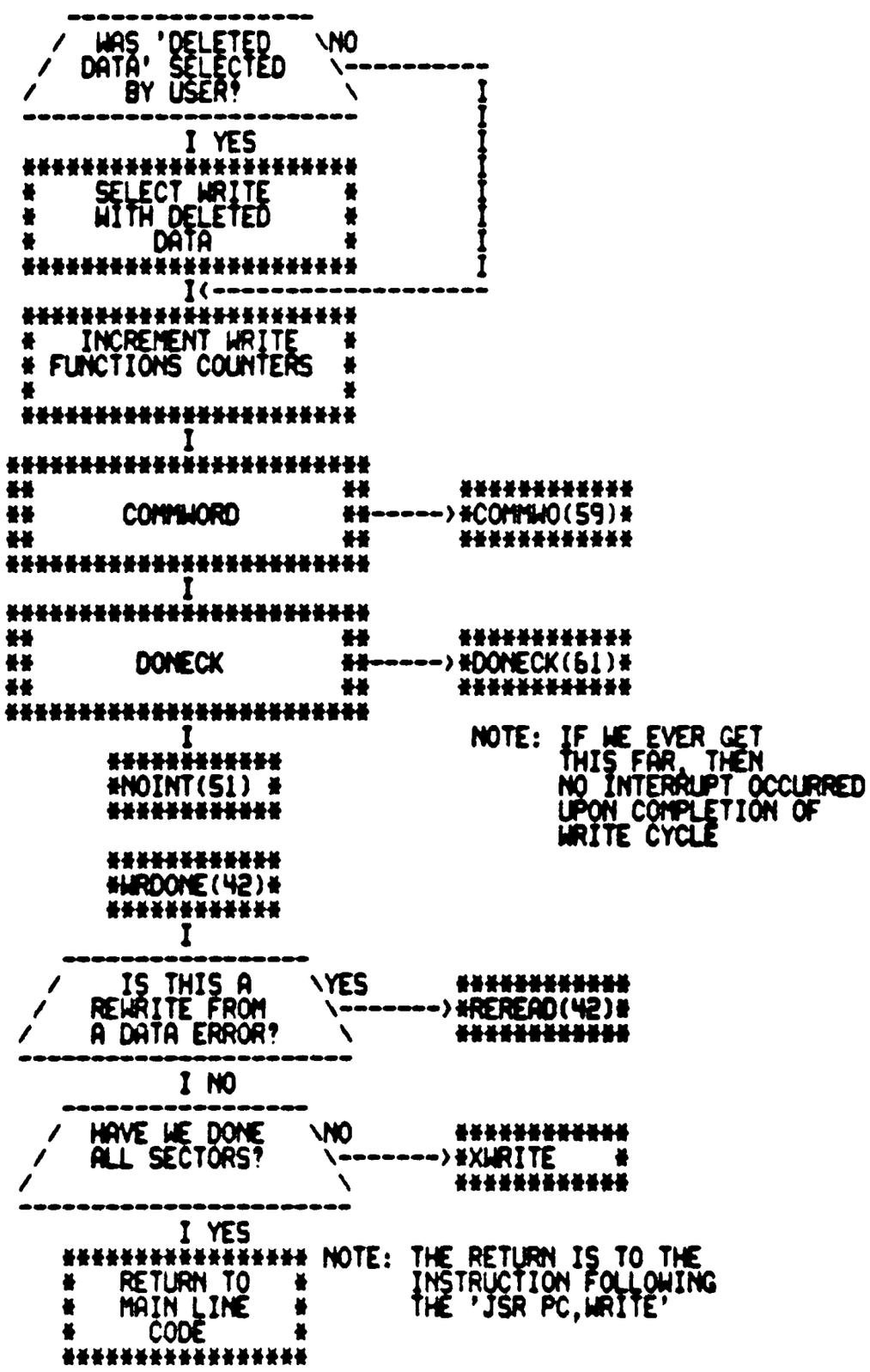

T=6
#DRVSWP(12)#
SET READ AFTER WRITE FLAG FOR ERROR TESTING
#SWP0
GETPATTERN
#GETPAT(83)#
#NEXT
INITTRACKS
#INITTR(84)#
GETUNIT
#GETUNI(86)#
GETTRACK
#GETTR(85) #
WRITE
#WRITE(25) #
READCHK
#RDCHK(87) #
GETUNIT
#GETUNI(86)#

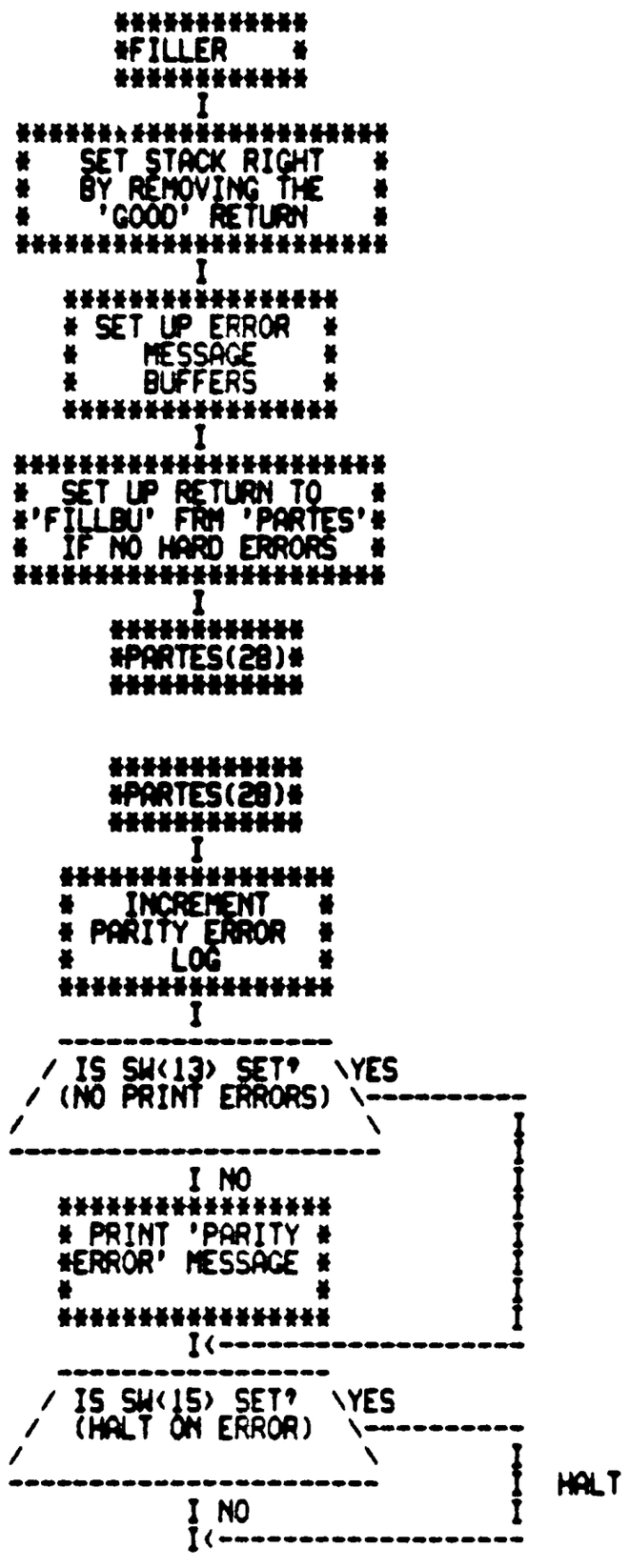

WRITE
WRITE(25)
READCHK
RDCHK(87)
HAS WRITE/READ CHK BEEN DONE ON ALL TRKS & UNITS?
NO
NEXT
YES
STOP
STOP(88)
SWPO

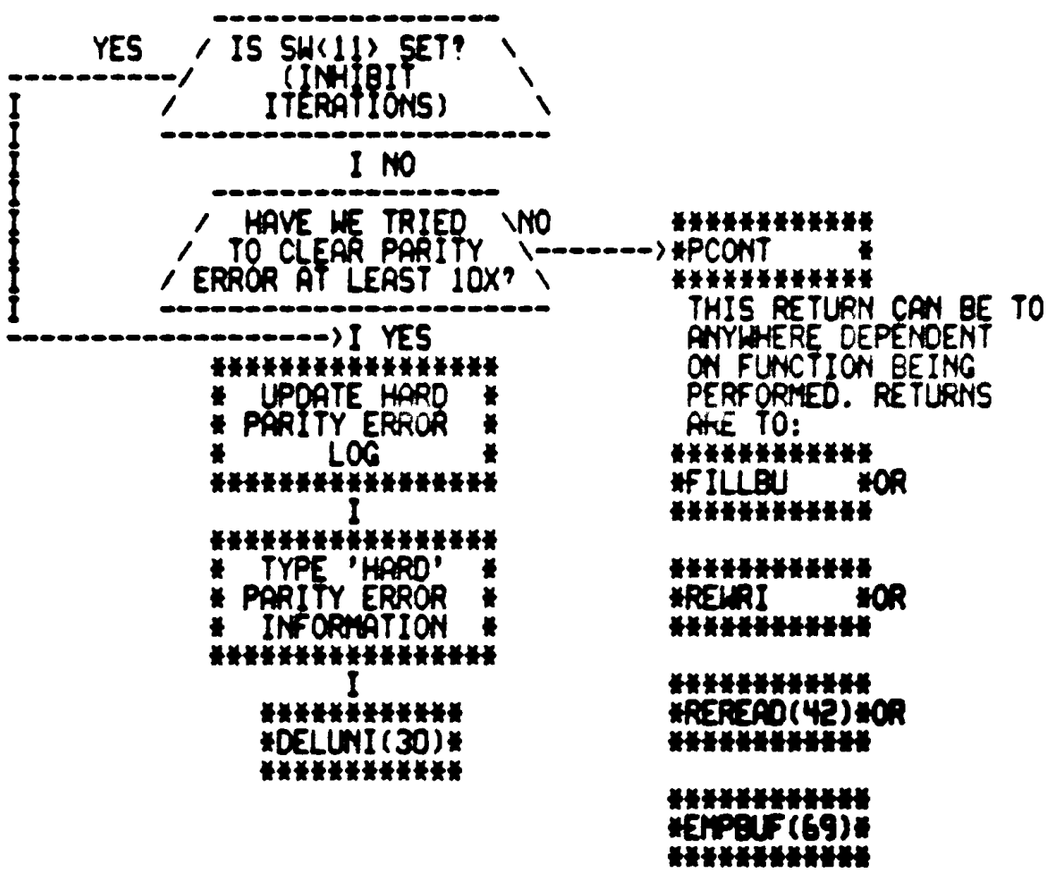

WRITE
INITSECTOR
INITSE(55)
XWRITE
GETSECTOR
GETSEC(56)
ONEWRT
SET RETRY COUNTERS TO 10.
FILLBU
ADJSUM
ADJSUM(57)
PUT A RETURN ON STACK FOR WHEN BUFR IS FULL & NO ERRORS EXIST
FILLDO(26)
PUT A RETURN ON STACK FOR WHEN ERRORS OCURR ON FILLING BUF
FILLER(28)
INITIALIZE BYTE COUNTER & SET 'PSW' TO LEVEL 4
ENABLE 'GO', 'FILL BUFFER' FUNCTION & 'INTERRUPT ENABLE'

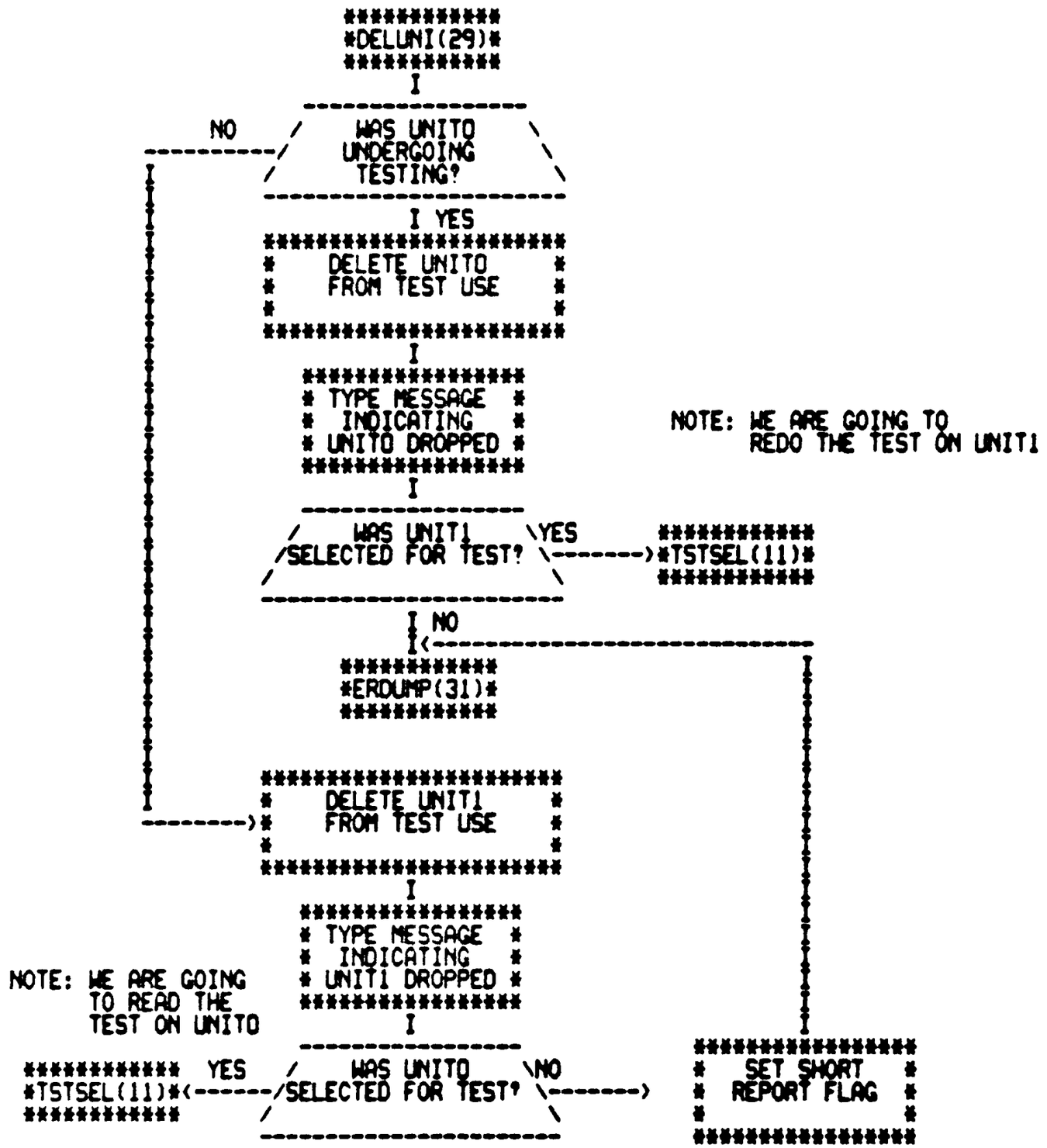

TRCK
TRCK(58)
TRANSFER A DATA BYTE
INCREMENT BYTE COUNTER
NOTE: WE LEAVE THIS LOOP WHEN THE RX11 BUFFER HAS BEEN FILLED WITH 128(10) BYTES OF DATA

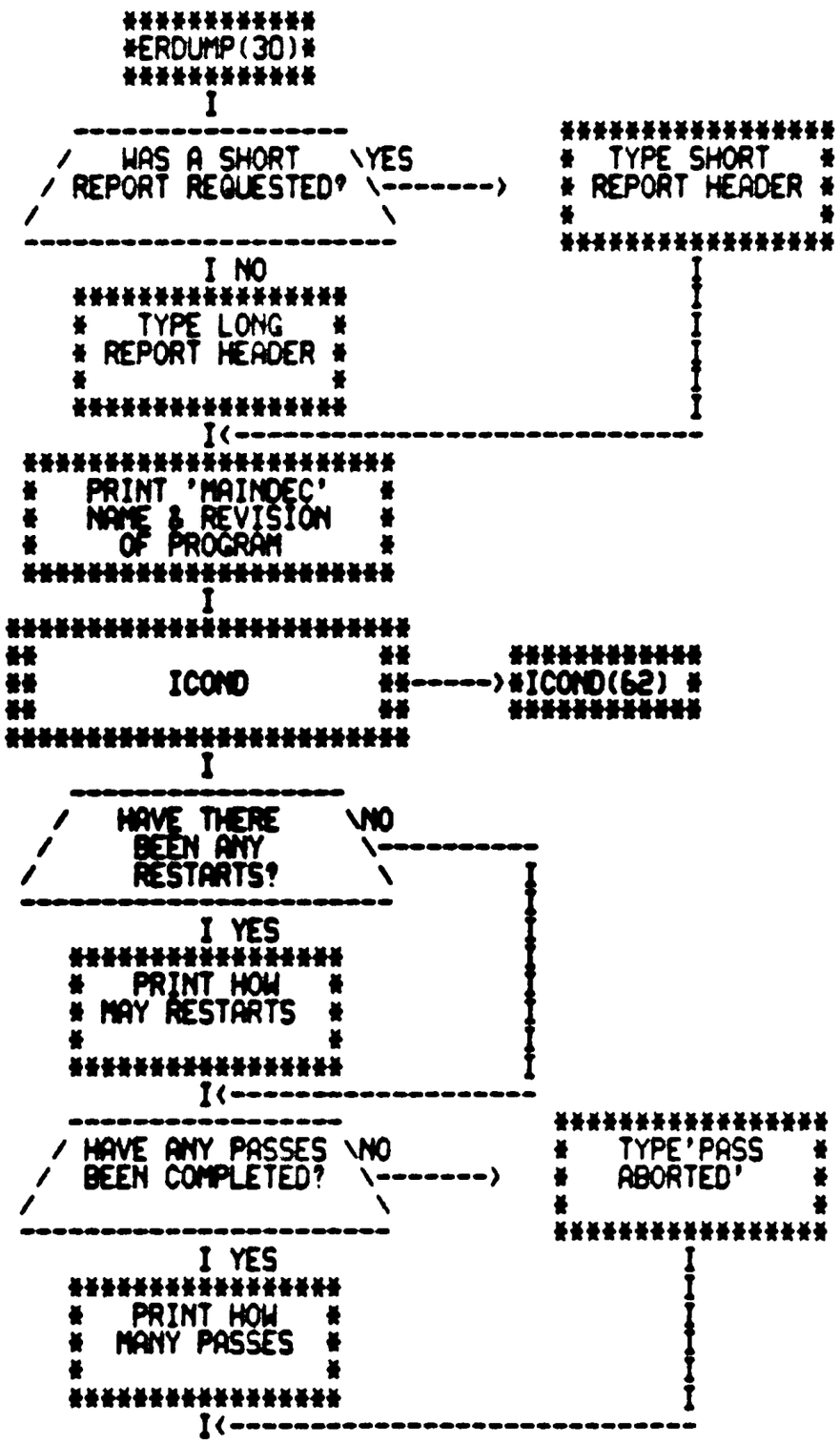

NOTE: WE COME HERE IF BUFFER HAS BEEN FILLED WITH NO ERRORS
FILLDO
SET PARITY RETRY COUNTER TO 10.
REWRI
PUT A RETURN ON STACK FOR WRITING DONE WITH NO ERRORS
WRDONE(27)
PUT A RETURN ON STACK FOR WHEN ERRORS OCURR DURING WRITING
WRTER(39)
SELECT 'WRITE' FUNCTION 'GO' & 'IE' BITS

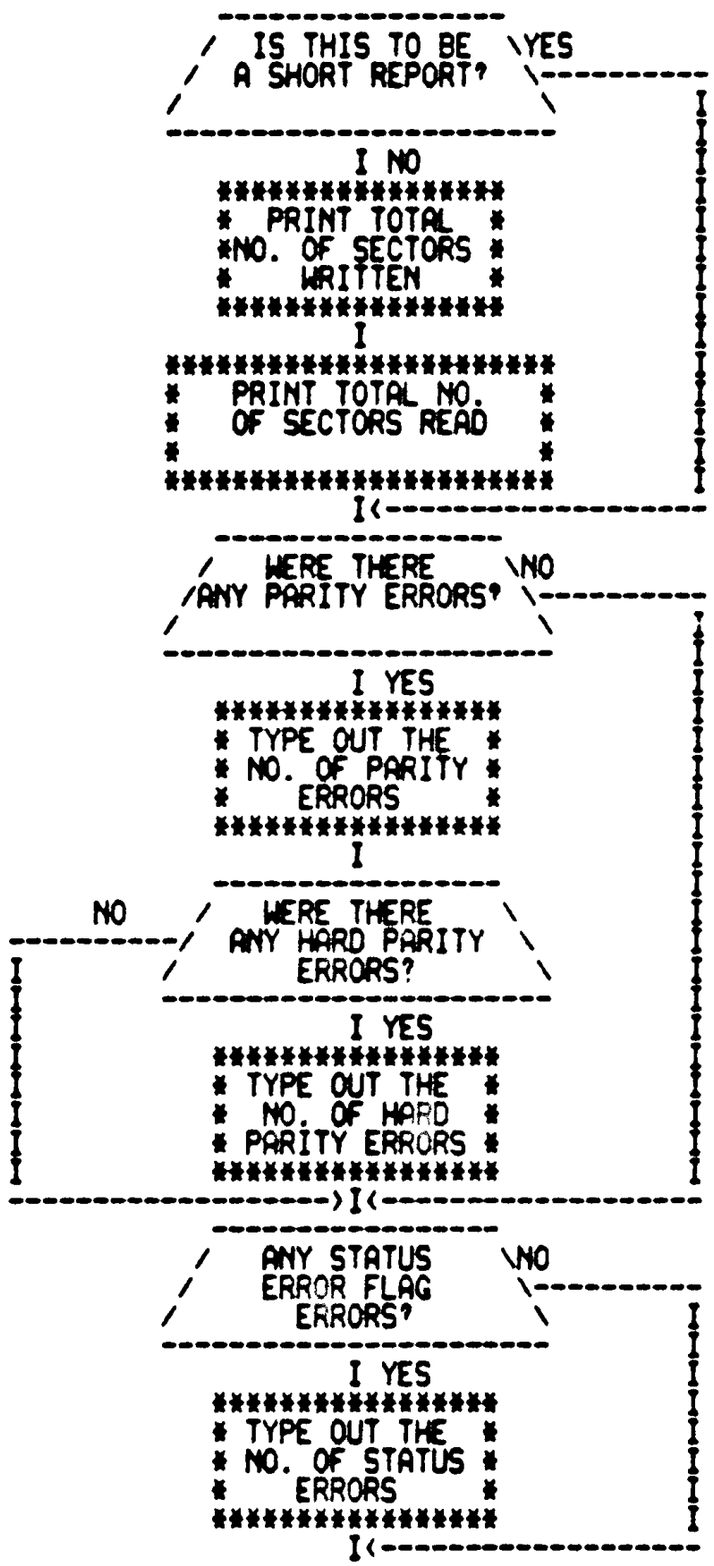

WAS 'DELETED DATA' SELECTED BY USER?
NO
YES
SELECT WRITE WITH DELETED DATA
INCREMENT WRITE FUNCTIONS COUNTERS
COMMWORD
COMMWO(59)
DONECK
DONECK(61)
NOINT(51)
WRDONE(42)
IS THIS A REWRITE FROM A DATA ERROR?
YES
REREAD(42)
NO
HAVE WE DONE ALL SECTORS?
NO
XWRITE
YES
RETURN TO MAIN LINE CODE
NOTE: IF WE EVER GET THIS FAR, THEN NO INTERRUPT OCCURRED UPON COMPLETION OF WRITE CYCLE
NOTE: THE RETURN IS TO THE INSTRUCTION FOLLOWING THE 'JSR PC,WRITE'

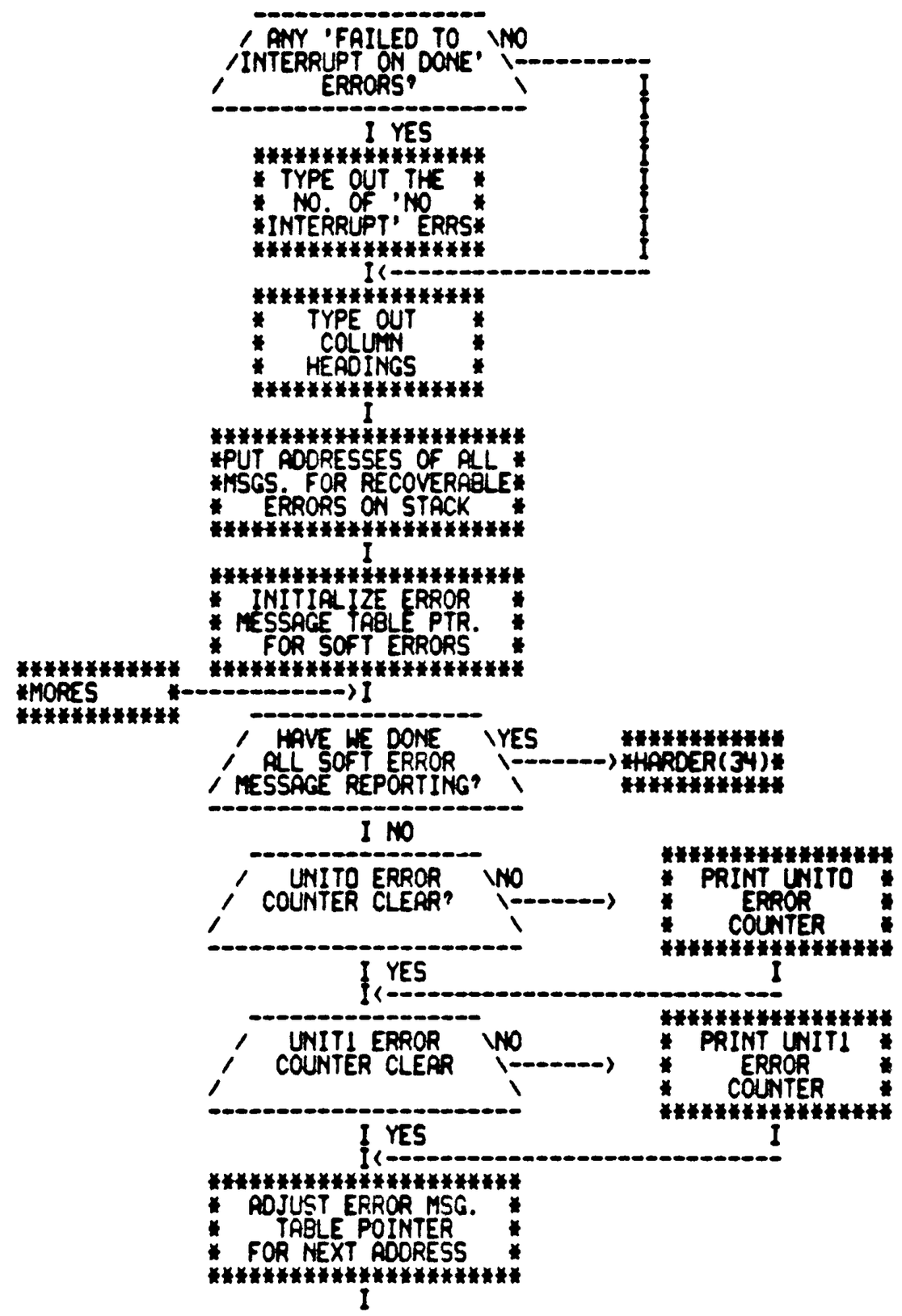

#FILLER #
SET STACK RIGHT BY REMOVING THE 'GOOD' RETURN
SET UP ERROR MESSAGE BUFFERS
SET UP RETURN TO 'FILLBU' FRM 'PARTES' IF NO HARD ERRORS
#PARTES(28)#
#PARTES(28)#
INCREMENT PARITY ERROR LOG
IS SW(13) SET? (NO PRINT ERRORS)
YES
NO
PRINT 'PARITY ERROR' MESSAGE
IS SW(15) SET? (HALT ON ERROR)
YES
HALT
NO

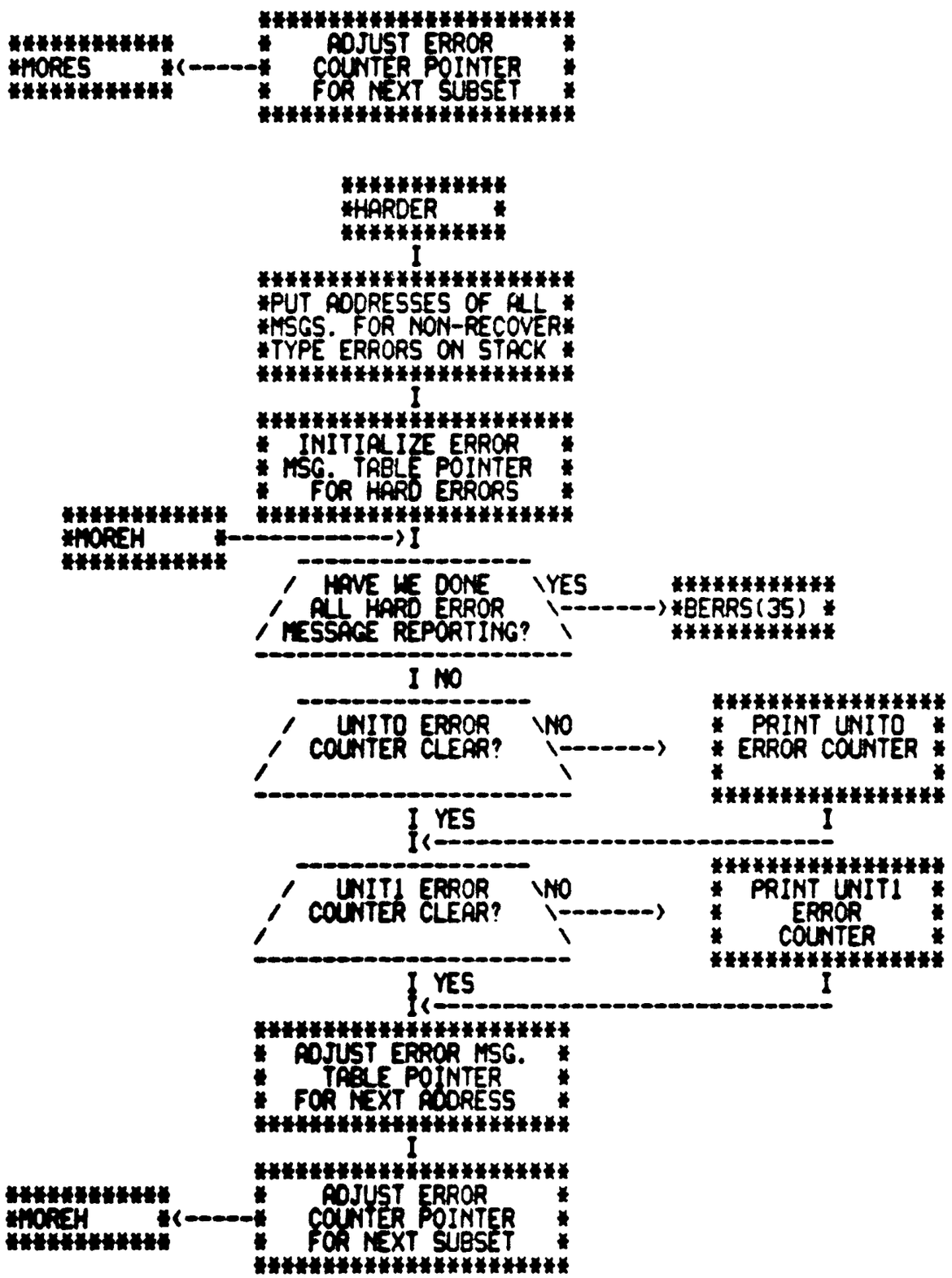

IS SW(11) SET? (INHIBIT ITERATIONS)
YES
I NO
HAVE WE TRIED TO CLEAR PARITY ERROR AT LEAST 10X?
NO
I YES
UPDATE HARD PARITY ERROR LOG
TYPE 'HARD' PARITY ERROR INFORMATION
DELUNI(30)
PCONT
THIS RETURN CAN BE TO ANYWHERE DEPENDENT ON FUNCTION BEING PERFORMED. RETURNS HERE TO:
FILLBU
OR
REWRI
OR
REREAD(42)
OR
EMPBUF(69)

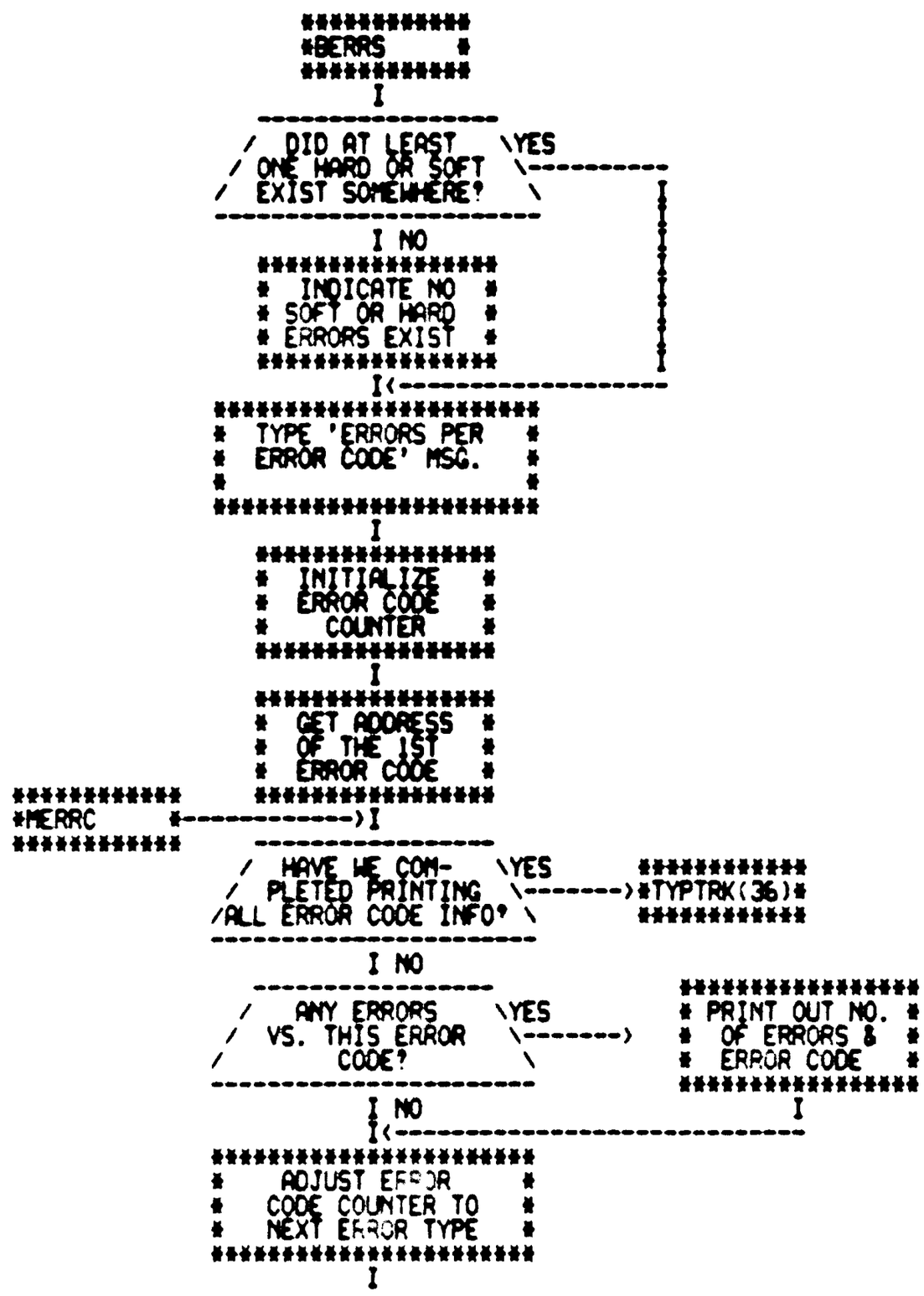

DELUNI(29)
WAS UNIT0 UNDERGOING TESTING?
NO
YES
DELETE UNIT0 FROM TEST USE
TYPE MESSAGE INDICATING UNIT0 DROPPED
WAS UNIT1 SELECTED FOR TEST?
YES
TSTSEL(11)
NO
ERDUMP(31)
DELETE UNIT1 FROM TEST USE
TYPE MESSAGE INDICATING UNIT1 DROPPED
WAS UNIT0 SELECTED FOR TEST?
YES
TSTSEL(11)
NO
SET SHORT REPORT FLAG
NOTE: WE ARE GOING TO REDO THE TEST ON UNIT1
NOTE: WE ARE GOING TO READ THE TEST ON UNIT0

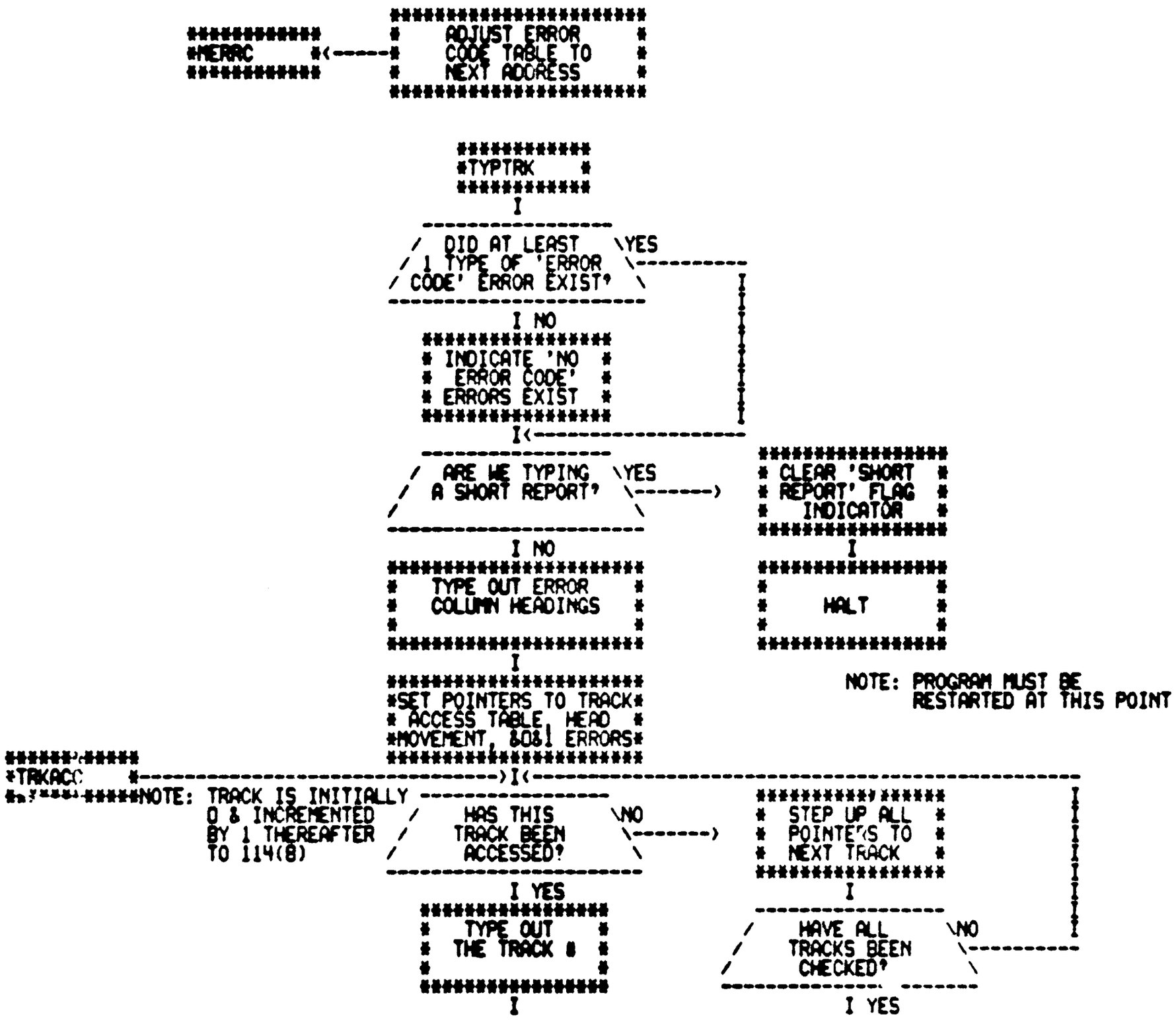

ERDUMP(30)
WAS A SHORT REPORT REQUESTED?
YES
TYPE SHORT REPORT HEADER
NO
TYPE LONG REPORT HEADER
PRINT 'MAINDEC' NAME & REVISION OF PROGRAM
ICOND
ICOND(62)
HAVE THERE BEEN ANY RESTARTS?
NO
YES
PRINT HOW MAY RESTARTS
HAVE ANY PASSES BEEN COMPLETED?
NO
TYPE'PASS ABORTED'
YES
PRINT HOW MANY PASSES

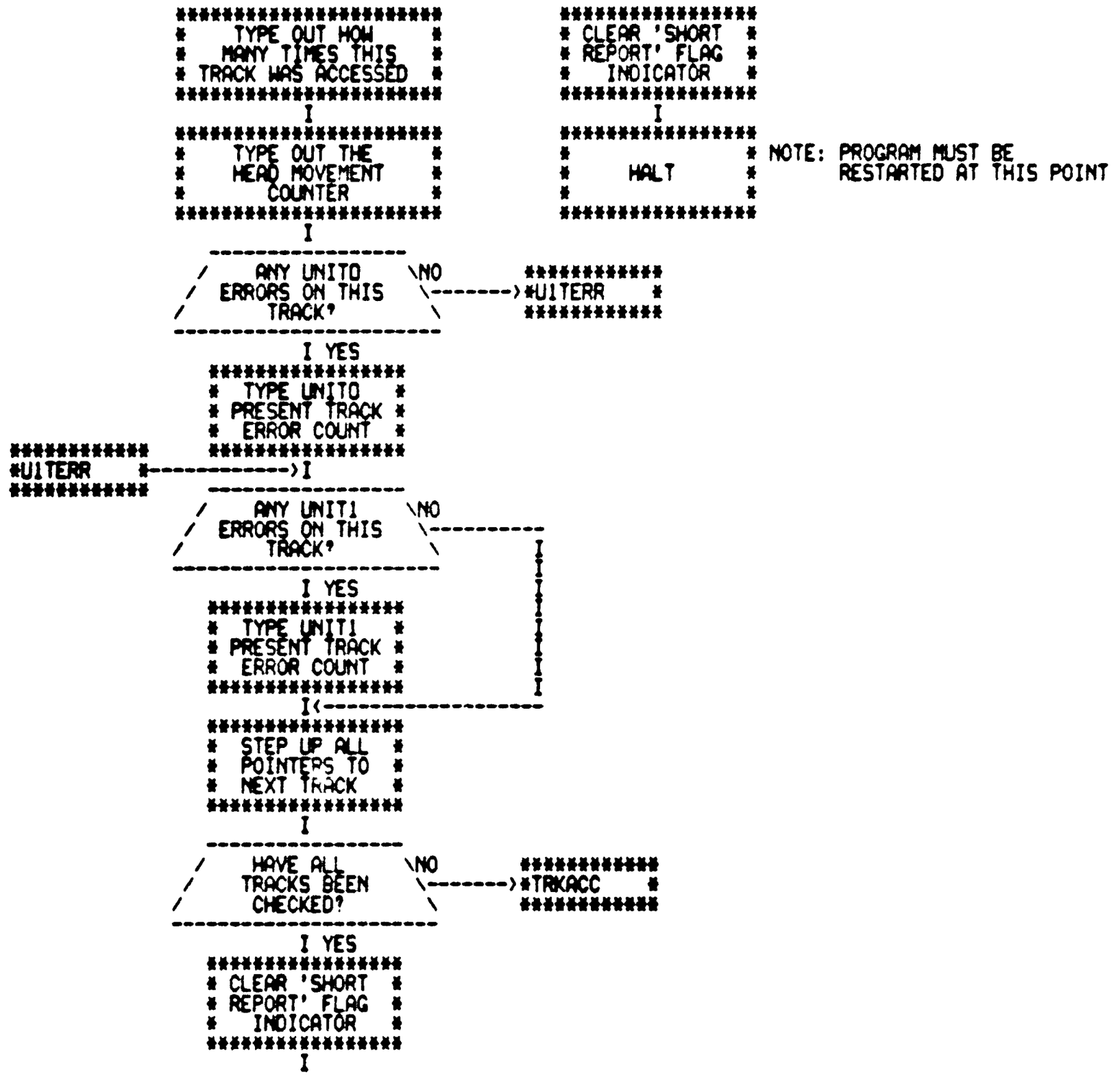

IS THIS TO BE A SHORT REPORT?
YES
NO
PRINT TOTAL NO. OF SECTORS WRITTEN
PRINT TOTAL NO. OF SECTORS READ
WERE THERE ANY PARITY ERRORS?
NO
YES
TYPE OUT THE NO. OF PARITY ERRORS
WERE THERE ANY HARD PARITY ERRORS?
NO
YES
TYPE OUT THE NO. OF HARD PARITY ERRORS
ANY STATUS ERROR FLAG ERRORS?
NO
YES
TYPE OUT THE NO. OF STATUS ERRORS


ANY 'FAILED TO INTERRUPT ON DONE' ERRORS?
NO
YES
TYPE OUT THE NO. OF 'NO INTERRUPT' ERRS
TYPE OUT COLUMN HEADINGS
PUT ADDRESSES OF ALL MSGS. FOR RECOVERABLE ERRORS ON STACK
INITIALIZE ERROR MESSAGE TABLE PTR. FOR SOFT ERRORS
MORES
HAVE WE DONE ALL SOFT ERROR MESSAGE REPORTING?
YES
HARDER(34)
NO
UNIT0 ERROR COUNTER CLEAR?
NO
YES
PRINT UNIT0 ERROR COUNTER
UNIT1 ERROR COUNTER CLEAR
NO
YES
PRINT UNIT1 ERROR COUNTER
ADJUST ERROR MSG. TABLE POINTER FOR NEXT ADDRESS


ADJUST ERROR COUNTER POINTER FOR NEXT SUBSET
MORES
HARDER
PUT ADDRESSES OF ALL MSGS. FOR NON-RECOVER TYPE ERRORS ON STACK
INITIALIZE ERROR MSG. TABLE POINTER FOR HARD ERRORS
MOREH
HAVE WE DONE ALL HARD ERROR MESSAGE REPORTING?
YES
BERRS(35)
NO
UNIT0 ERROR COUNTER CLEAR?
NO
PRINT UNIT0 ERROR COUNTER
YES
UNIT1 ERROR COUNTER CLEAR?
NO
PRINT UNIT1 ERROR COUNTER
YES
ADJUST ERROR MSG. TABLE POINTER FOR NEXT ADDRESS
ADJUST ERROR COUNTER POINTER FOR NEXT SUBSET
MOREH

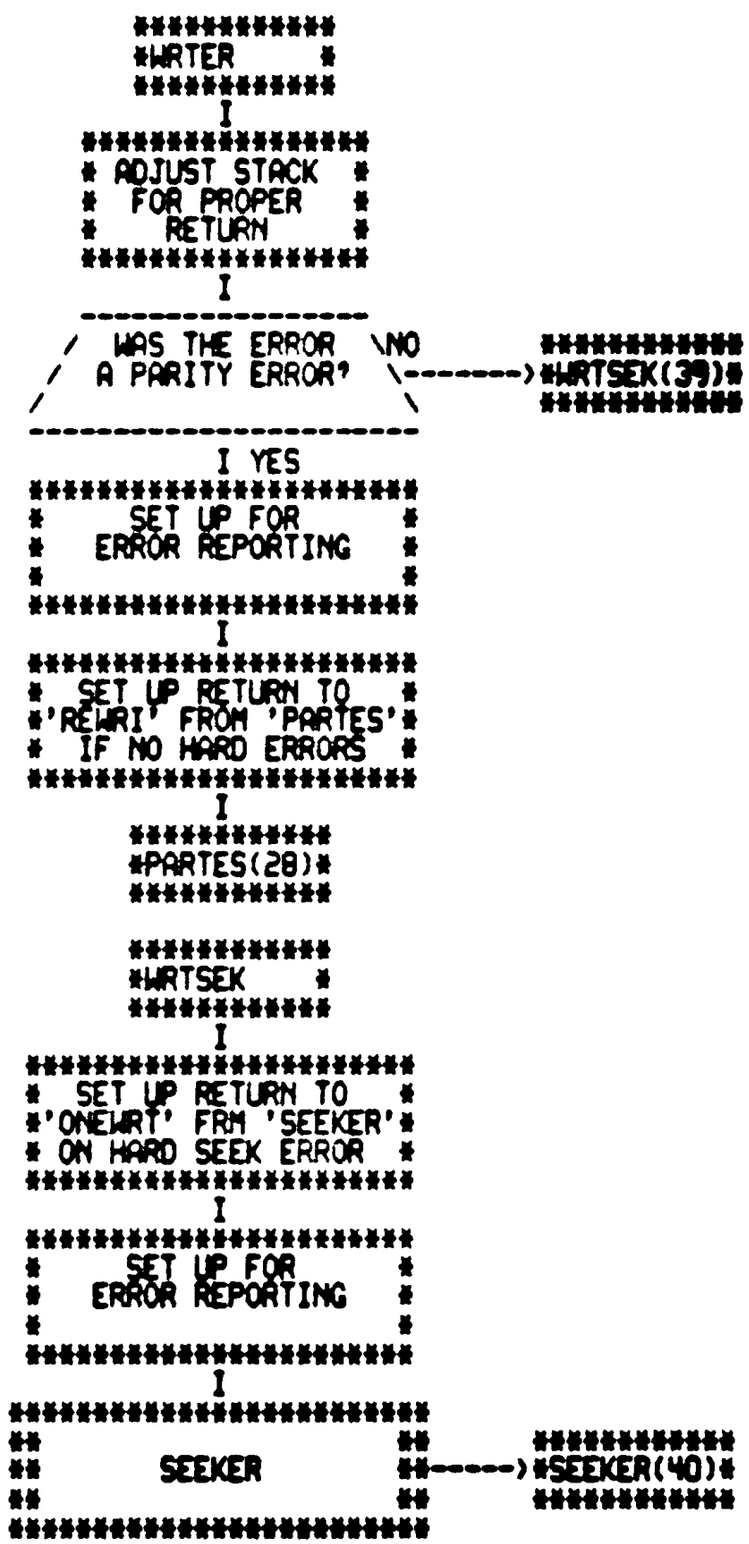

*BERRS
DID AT LEAST ONE HARD OR SOFT EXIST SOMEWHERE?
YES
NO
INDICATE NO SOFT OR HARD ERRORS EXIST
TYPE 'ERRORS PER ERROR CODE' MSG.
INITIALIZE ERROR CODE COUNTER
GET ADDRESS OF THE 1ST ERROR CODE
*MERRC
HAVE WE COMPLETED PRINTING ALL ERROR CODE INFO?
YES
TYPTRK(36)
NO
ANY ERRORS VS. THIS ERROR CODE?
YES
PRINT OUT NO. OF ERRORS & ERROR CODE
NO
ADJUST ERROR CODE COUNTER TO NEXT ERROR TYPE

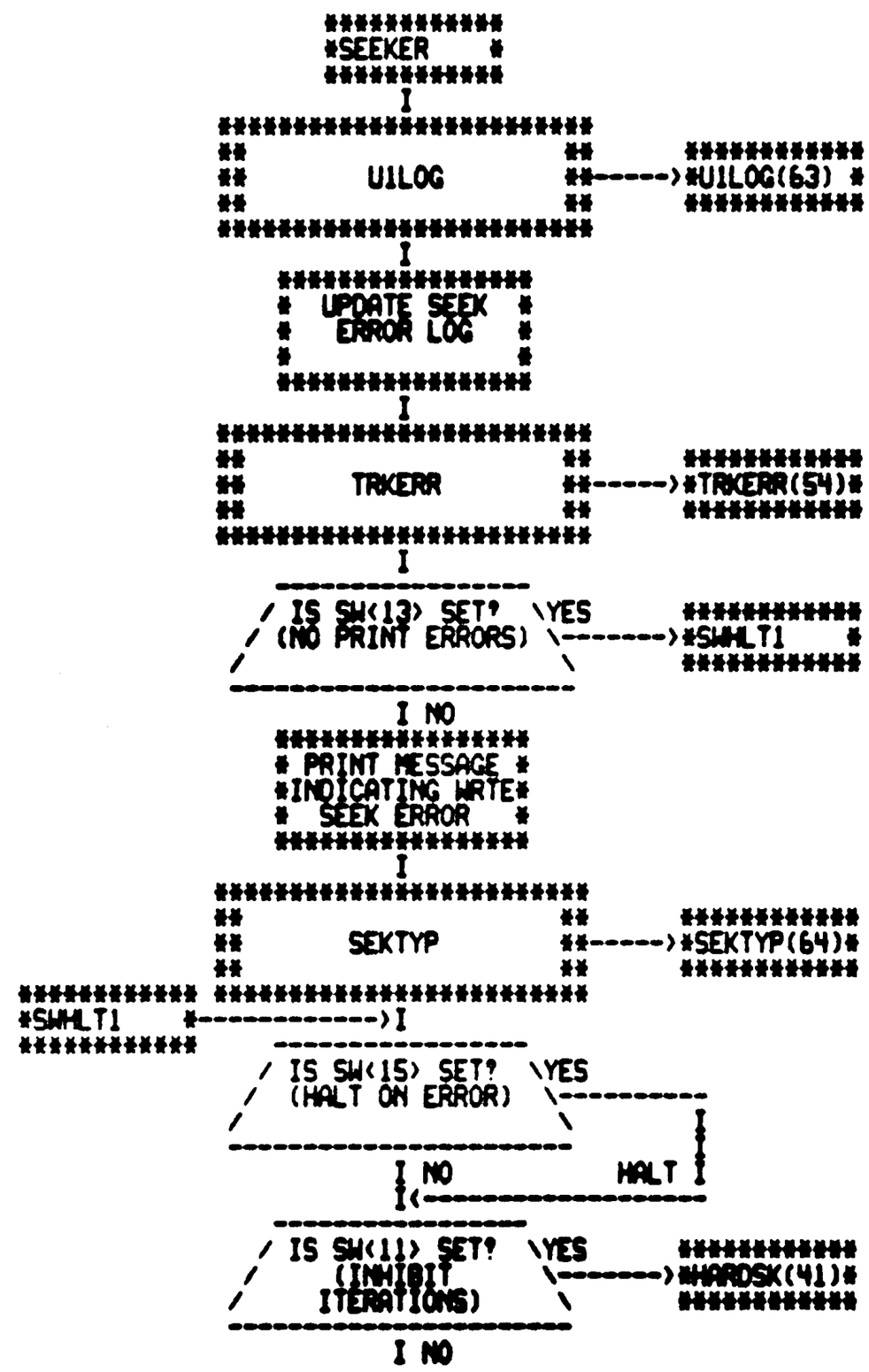

ADJUST ERROR CODE TABLE TO NEXT ADDRESS
#ERRAC
TYPTRK
DID AT LEAST 1 TYPE OF 'ERROR CODE' ERROR EXIST?
YES
NO
INDICATE 'NO ERROR CODE' ERRORS EXIST
ARE WE TYPING A SHORT REPORT?
YES
NO
CLEAR 'SHORT REPORT' FLAG INDICATOR
HALT
NOTE: PROGRAM MUST BE RESTARTED AT THIS POINT
TYPE OUT ERROR COLUMN HEADINGS
SET POINTERS TO TRACK ACCESS TABLE, HEAD MOVEMENT, &0&1 ERRORS
TRKACC
NOTE: TRACK IS INITIALLY 0 & INCREMENTED BY 1 THEREAFTER TO 114(8)
HAS THIS TRACK BEEN ACCESSED?
NO
YES
STEP UP ALL POINTERS TO NEXT TRACK
HAVE ALL TRACKS BEEN CHECKED?
NO
YES
TYPE OUT THE TRACK #

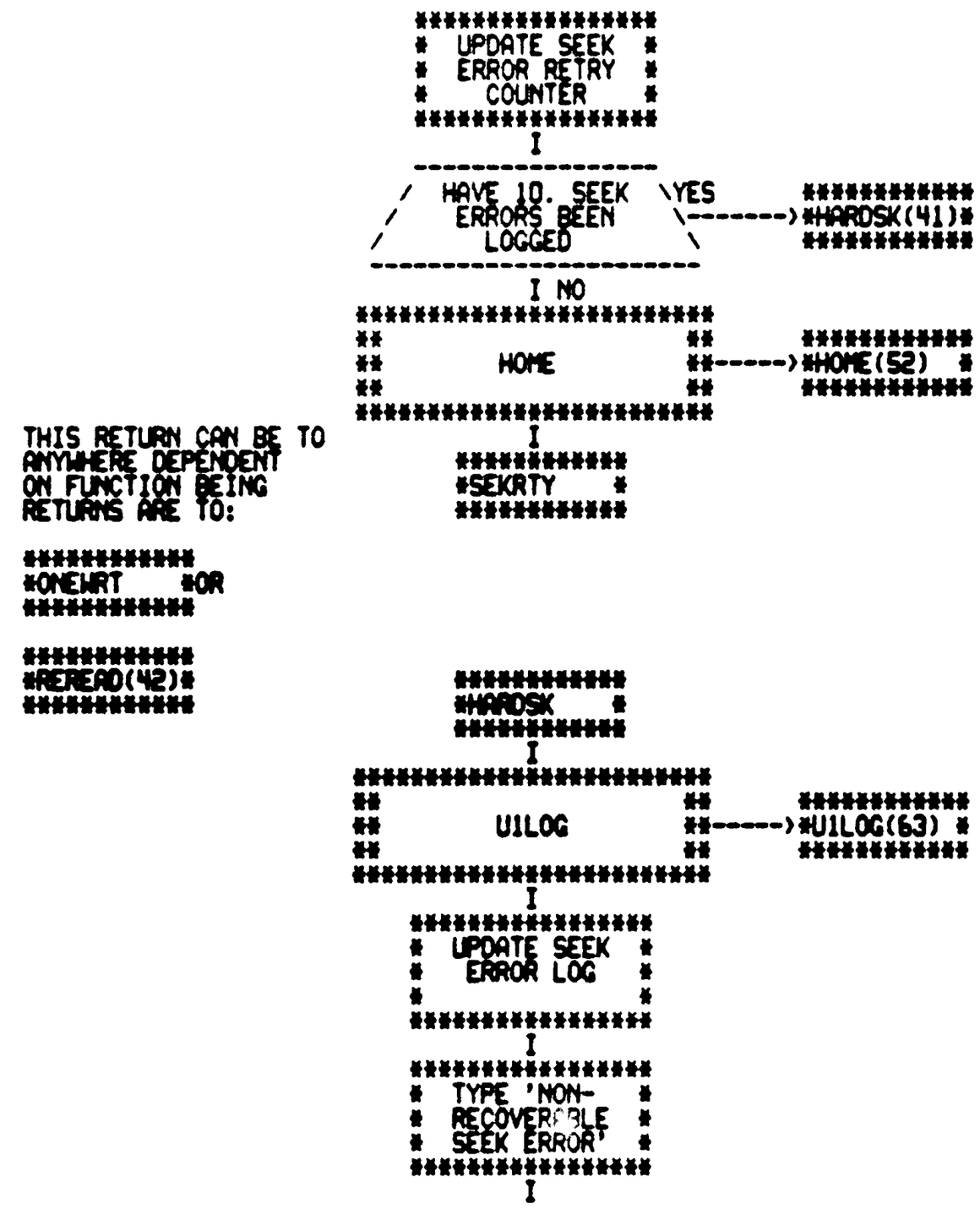

TYPE OUT HOW MANY TIMES THIS TRACK WAS ACCESSED
TYPE OUT THE HEAD MOVEMENT COUNTER
ANY UNIT0 ERRORS ON THIS TRACK?
NO
#UITERR
YES
TYPE UNIT0 PRESENT TRACK ERROR COUNT
#UITERR
ANY UNIT1 ERRORS ON THIS TRACK?
NO
YES
TYPE UNIT1 PRESENT TRACK ERROR COUNT
STEP UP ALL POINTERS TO NEXT TRACK
HAVE ALL TRACKS BEEN CHECKED?
NO
#TRKACC
YES
CLEAR 'SHORT REPORT' FLAG INDICATOR
CLEAR 'SHORT REPORT' FLAG INDICATOR
HALT
NOTE: PROGRAM MUST BE RESTARTED AT THIS POINT

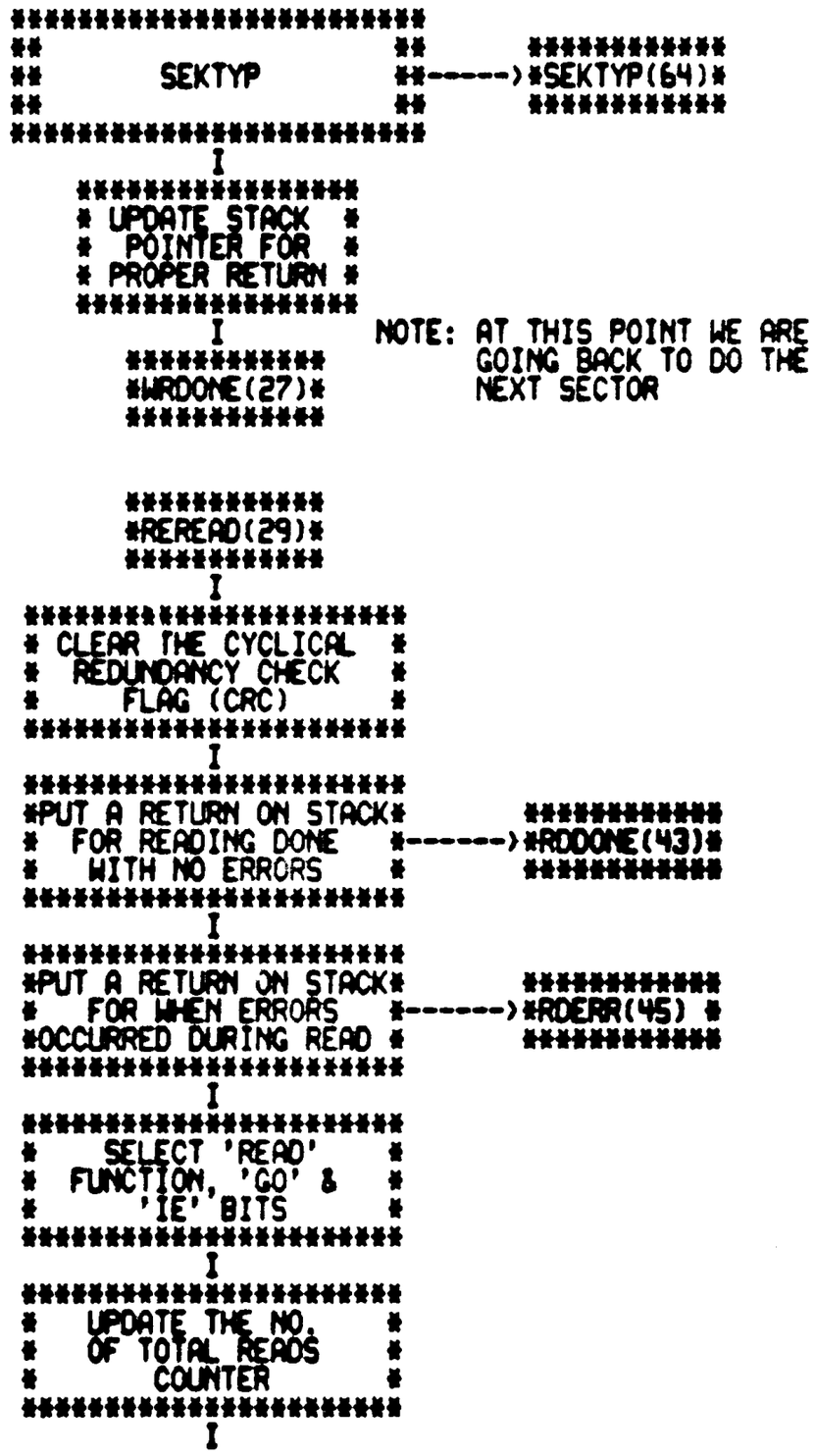
E05

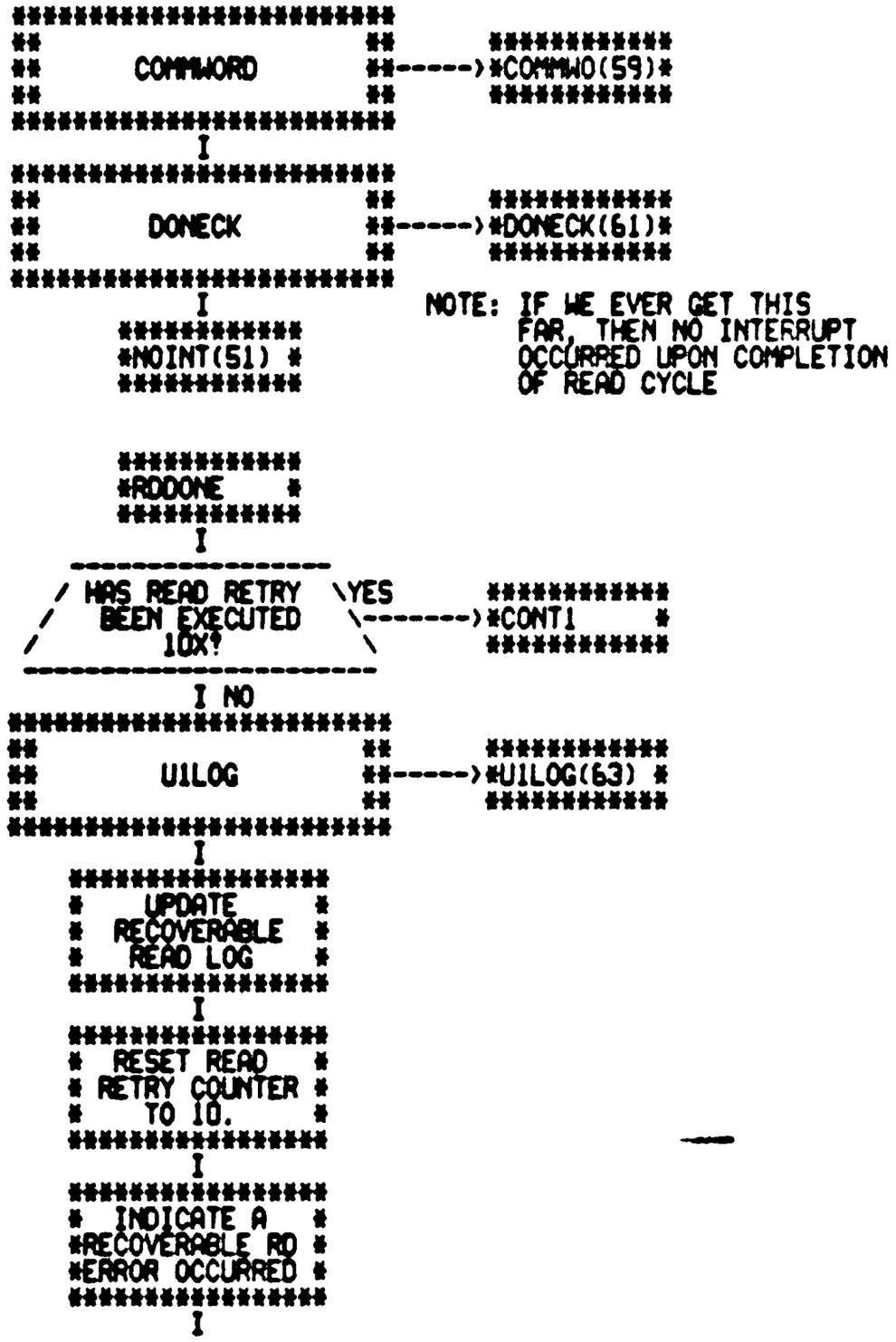

```
****************** NOTE: PROGRAM MUST BE
*                *       RESTARTED AT THIS POINT
*     HALT       *
*                *
******************
```

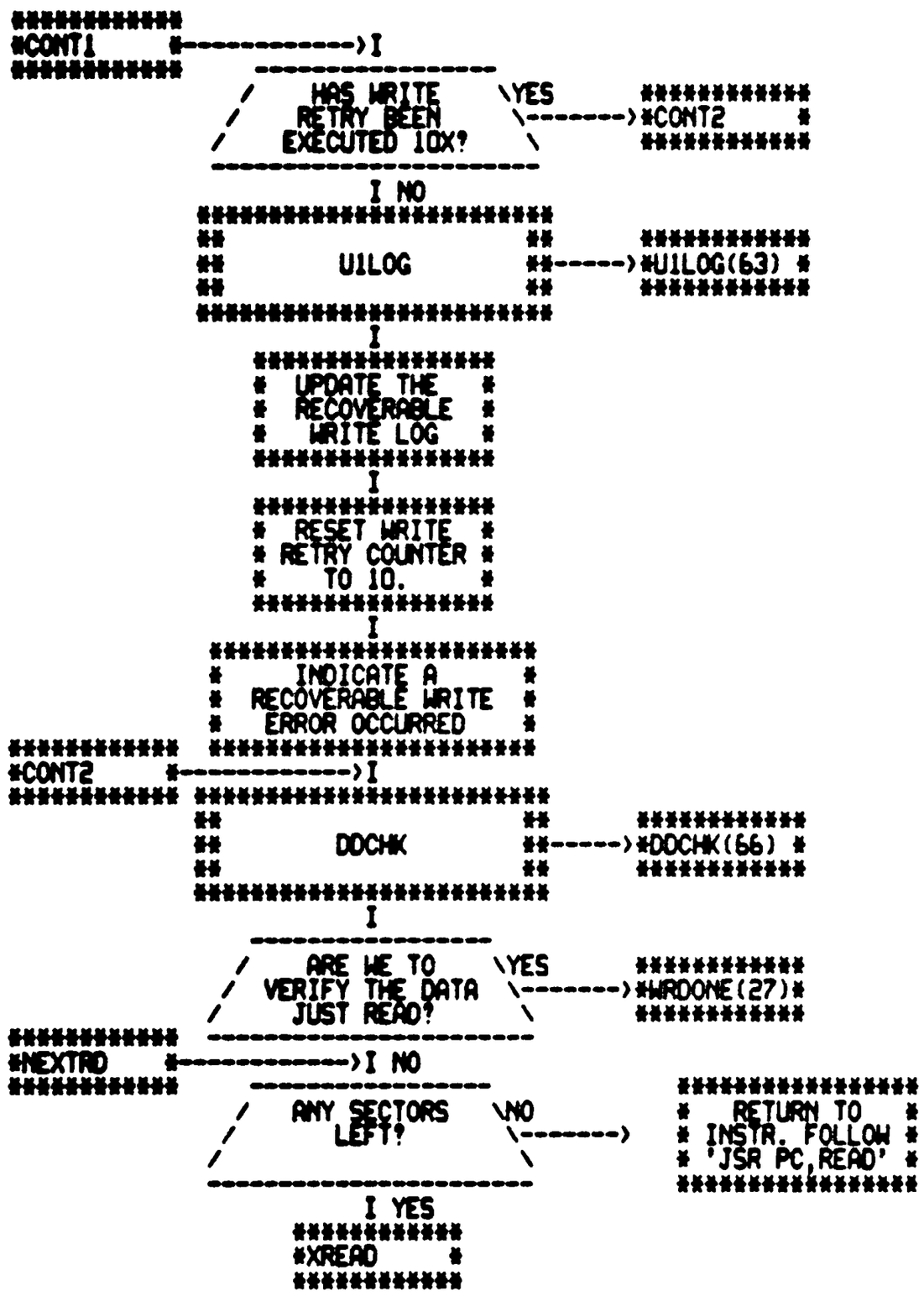

WRTER
ADJUST STACK FOR PROPER RETURN
WAS THE ERROR A PARITY ERROR?
NO
WRTSEK(39)
YES
SET UP FOR ERROR REPORTING
SET UP RETURN TO 'REWRI' FROM 'PARTES' IF NO HARD ERRORS
PARTES(28)
WRTSEK
SET UP RETURN TO 'ONEWRT' FRM 'SEEKER' ON HARD SEEK ERROR
SET UP FOR ERROR REPORTING
SEEKER
SEEKER(40)

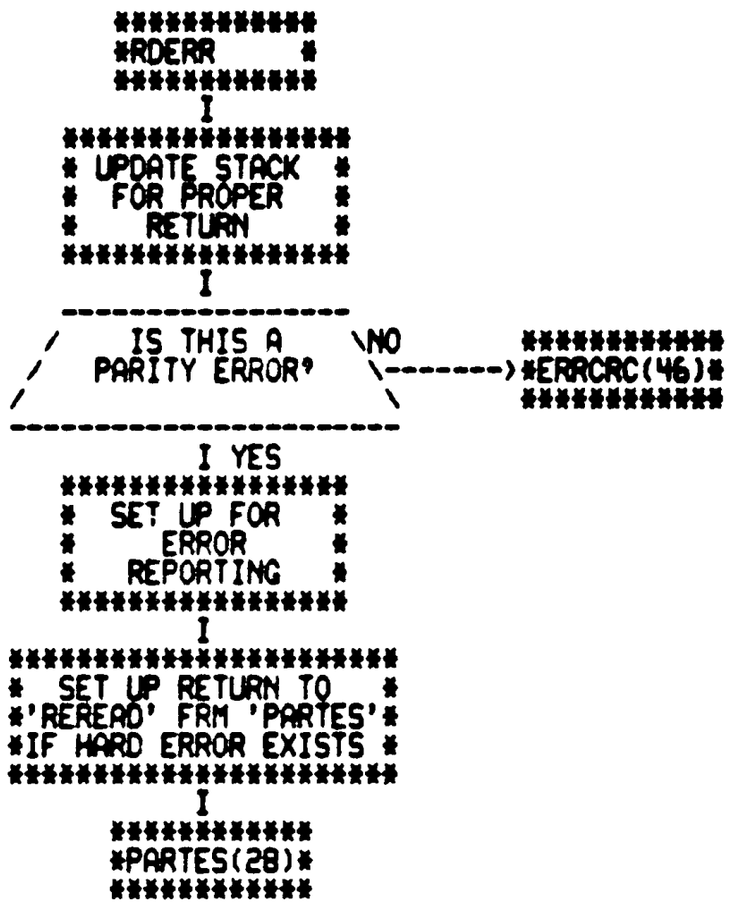

SEEKER
UILOG
UILOG(63)
UPDATE SEEK ERROR LOG
TRKERR
TRKERR(54)
IS SW(13) SET? (NO PRINT ERRORS)
YES
SWHLT1
NO
PRINT MESSAGE INDICATING WRTE SEEK ERROR
SEKTYP
SEKTYP(64)
SWHLT1
IS SW(15) SET? (HALT ON ERROR)
HALT
NO
IS SW(11) SET? (INHIBIT ITERATIONS)
HARDSK(41)
NO

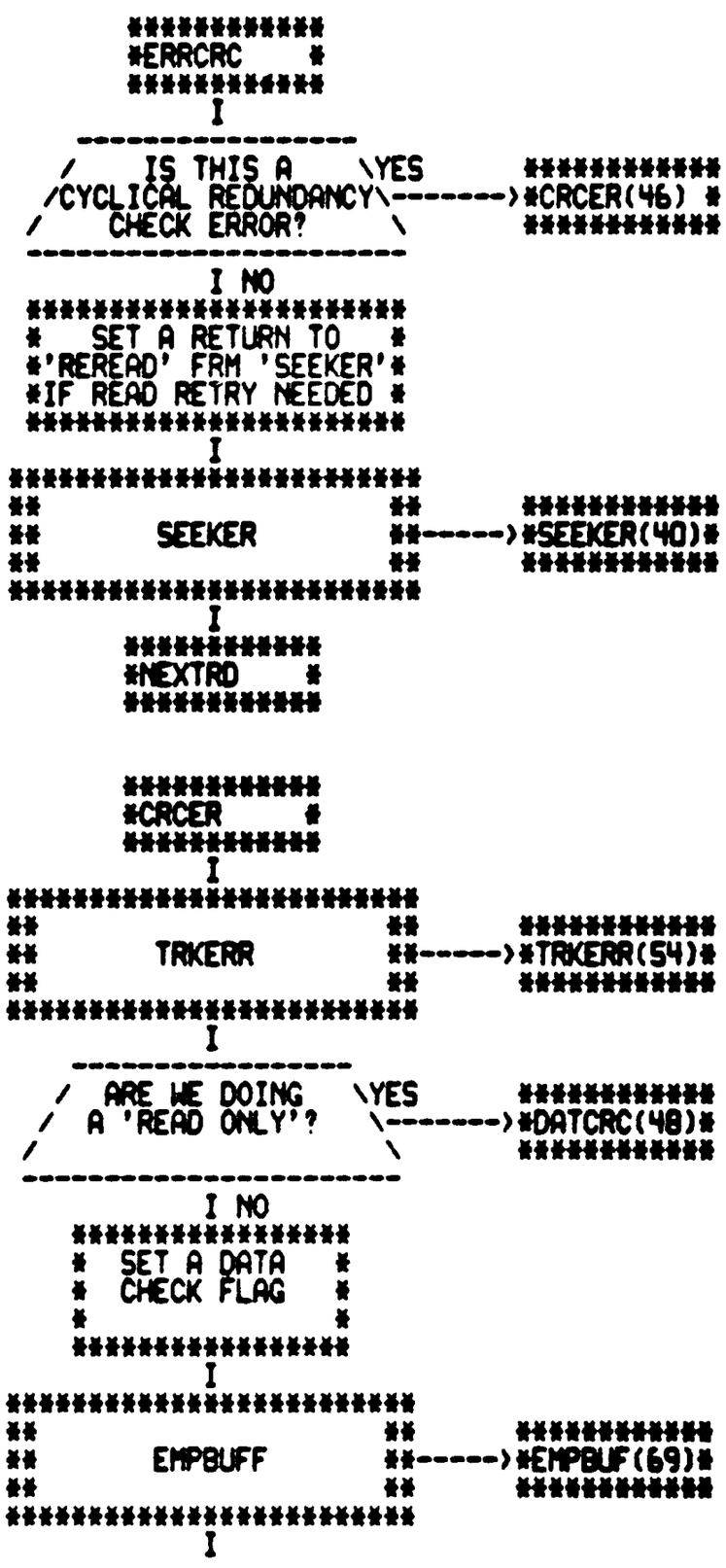

UPDATE SEEK ERROR RETRY COUNTER
HAVE 10. SEEK ERRORS BEEN LOGGED
YES
HARDSK(41)
NO
HOME
HOME(52)
THIS RETURN CAN BE TO ANYWHERE DEPENDENT ON FUNCTION BEING RETURNS ARE TO:
ONEWRT
OR
REREAD(42)
SEKRTY
HARDSK
U1LOG
U1LOG(63)
UPDATE SEEK ERROR LOG
TYPE 'NON-RECOVERABLE SEEK ERROR'

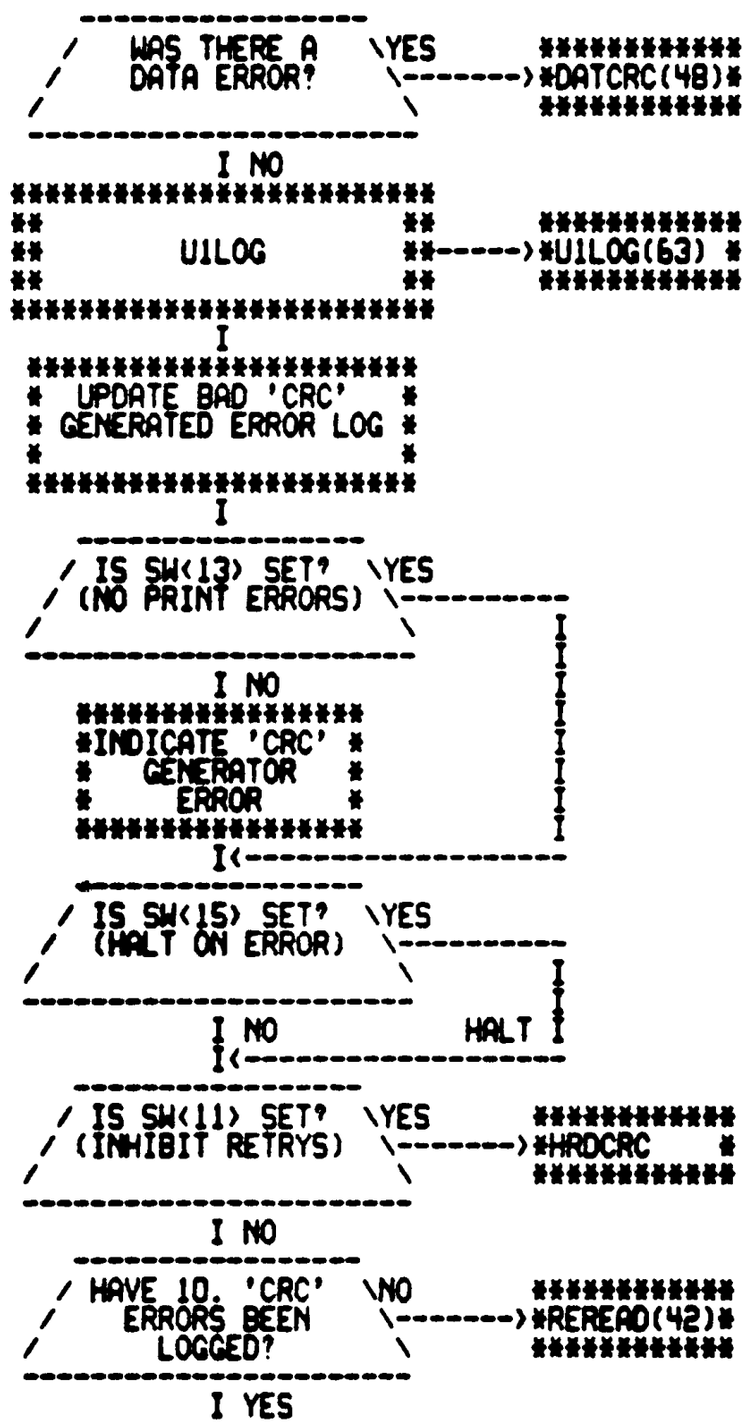

SEKTYP
SEKTYP(64)
UPDATE STACK POINTER FOR PROPER RETURN
NOTE: AT THIS POINT WE ARE GOING BACK TO DO THE NEXT SECTOR
WRDONE(27)
REREAD(29)
CLEAR THE CYCLICAL REDUNDANCY CHECK FLAG (CRC)
PUT A RETURN ON STACK FOR READING DONE WITH NO ERRORS
RDDONE(43)
PUT A RETURN ON STACK FOR WHEN ERRORS OCCURRED DURING READ
RDERR(45)
SELECT 'READ' FUNCTION, 'GO' & 'IE' BITS
UPDATE THE NO. OF TOTAL READS COUNTER

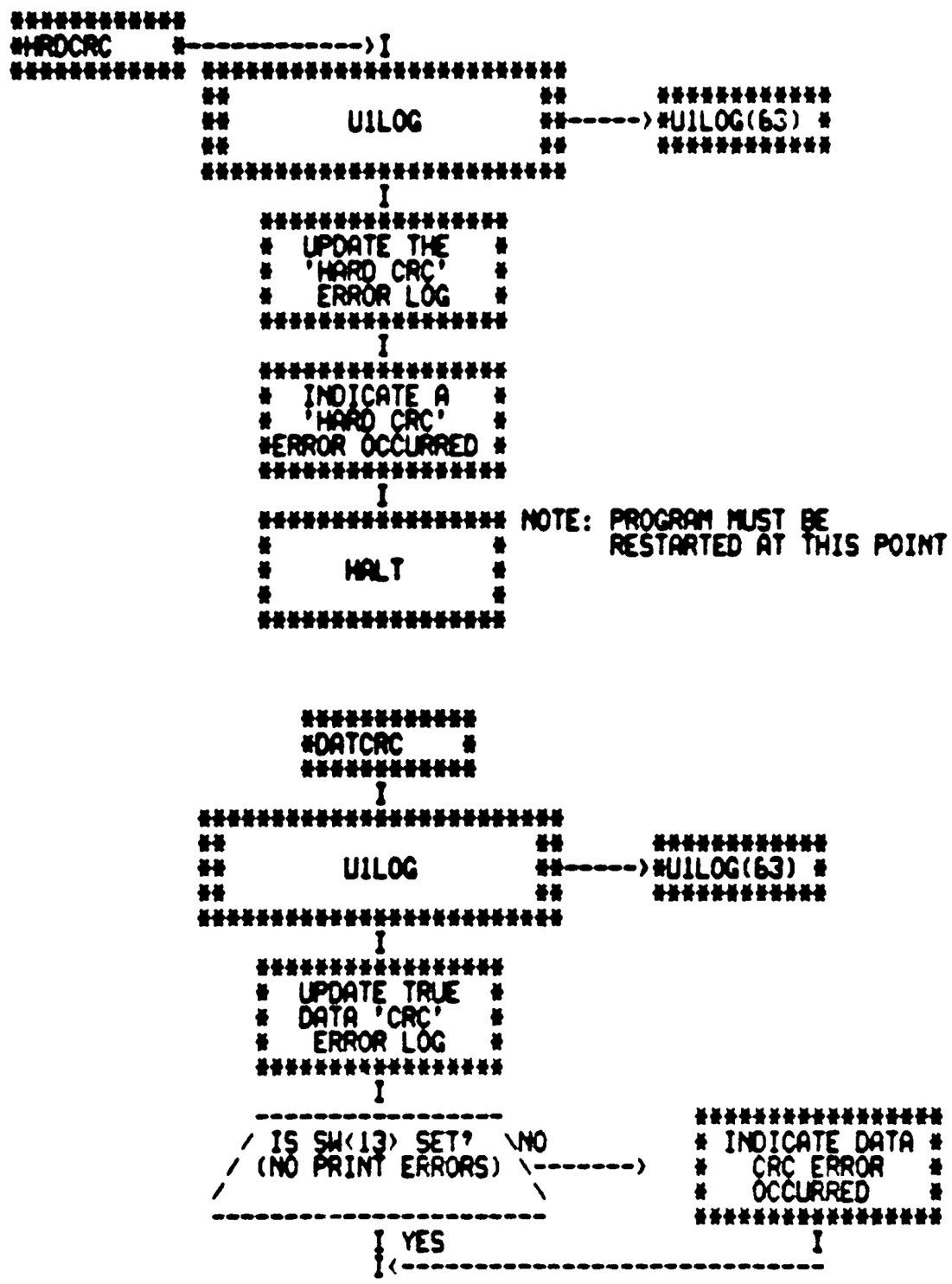

COMMWORD
COMMWO(59)
DONECK
DONECK(61)
NOTE: IF WE EVER GET THIS FAR, THEN NO INTERRUPT OCCURRED UPON COMPLETION OF READ CYCLE
NOINT(51)
RDDONE
HAS READ RETRY BEEN EXECUTED 10X?
YES
CONT1
NO
UILOG
UILOG(63)
UPDATE RECOVERABLE READ LOG
RESET READ RETRY COUNTER TO 10.
INDICATE A RECOVERABLE RD ERROR OCCURRED

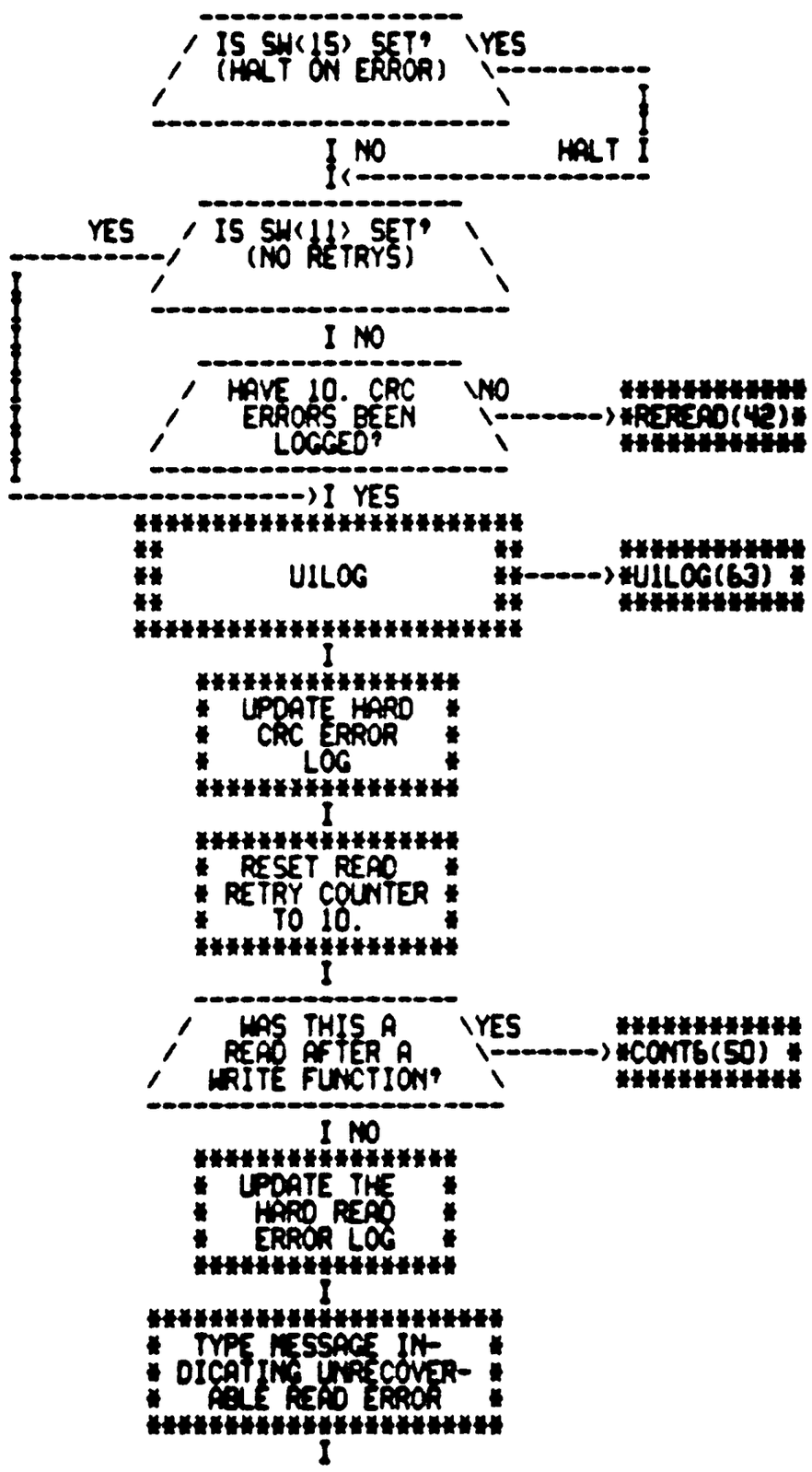

*CONT1
HAS WRITE RETRY BEEN EXECUTED 10X?
YES
*CONT2
NO
UILOG
*UILOG(63)
UPDATE THE RECOVERABLE WRITE LOG
RESET WRITE RETRY COUNTER TO 10.
INDICATE A RECOVERABLE WRITE ERROR OCCURRED
*CONT2
DDCHK
*DDCHK(66)
ARE WE TO VERIFY THE DATA JUST READ?
YES
*WRDONE(27)
*NEXTRD
NO
ANY SECTORS LEFT?
NO
RETURN TO INSTR. FOLLOW 'JSR PC,READ'
YES
*XREAD

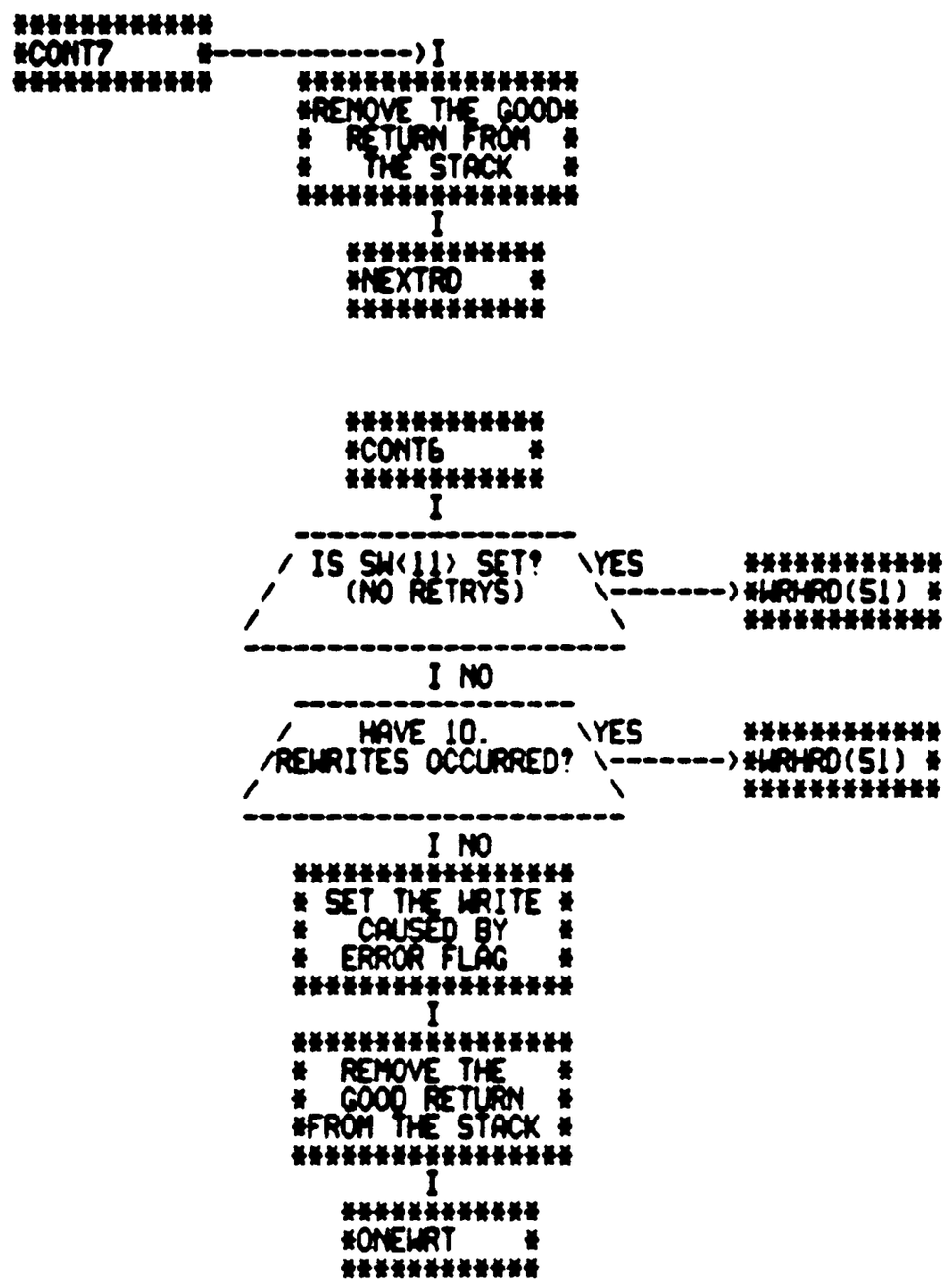

*RDERR *
UPDATE STACK FOR PROPER RETURN
IS THIS A PARITY ERROR?
NO
ERRCRC(46)
YES
SET UP FOR ERROR REPORTING
SET UP RETURN TO 'REREAD' FRM 'PARTES' IF HARD ERROR EXISTS
PARTES(28)

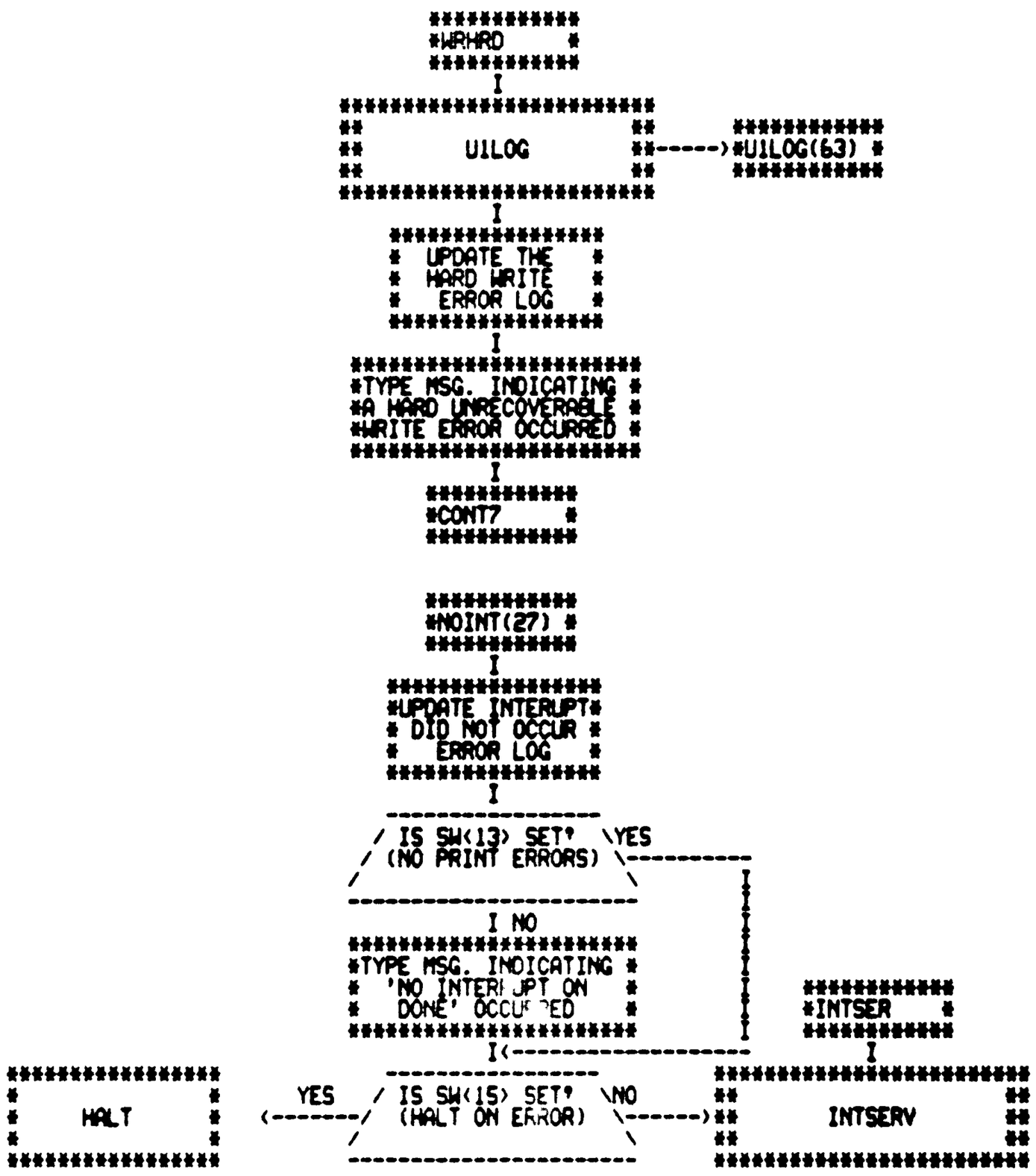

*ERRCRC *
IS THIS A CYCLICAL REDUNDANCY CHECK ERROR?
YES
*CRCER(46) *
NO
* SET A RETURN TO 'REREAD' FRM 'SEEKER' IF READ RETRY NEEDED *
SEEKER
SEEKER(40)
*NEXTRD *
*CRCER *
TRKERR
TRKERR(54)
ARE WE DOING A 'READ ONLY'?
YES
DATCRC(48)
NO
* SET A DATA CHECK FLAG *
EMPBUFF
EMPBUF(69)

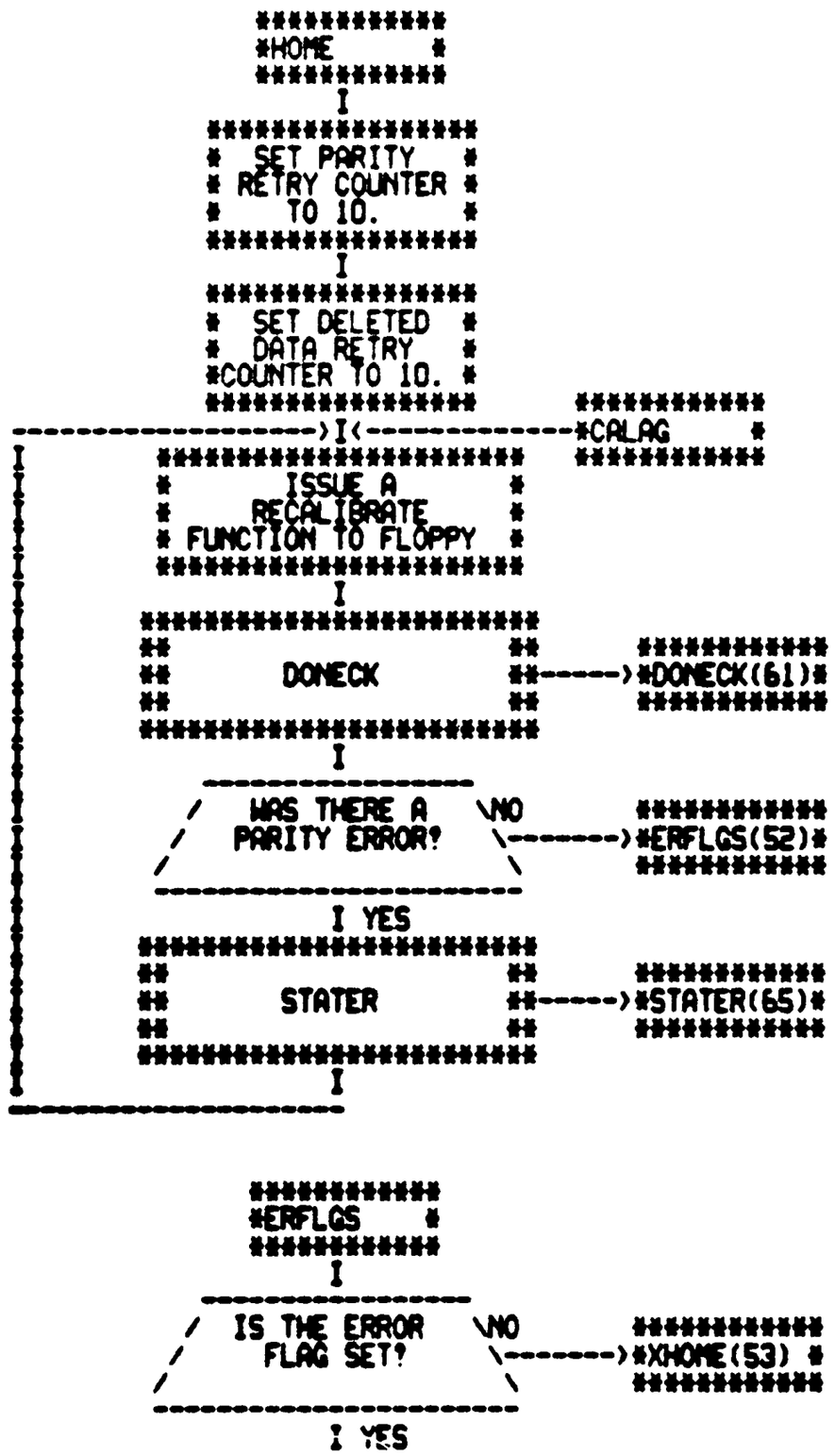

WAS THERE A DATA ERROR?
YES
DATCRC(48)
I NO
U1LOG
*U1LOG(63) *
UPDATE BAD 'CRC' GENERATED ERROR LOG
IS SW(13) SET? (NO PRINT ERRORS)
YES
I NO
INDICATE 'CRC' GENERATOR ERROR
IS SW(15) SET? (HALT ON ERROR)
YES
I NO
HALT
IS SW(11) SET? (INHIBIT RETRYS)
YES
*HRDCRC *
I NO
HAVE 10. 'CRC' ERRORS BEEN LOGGED?
NO
REREAD(42)
I YES

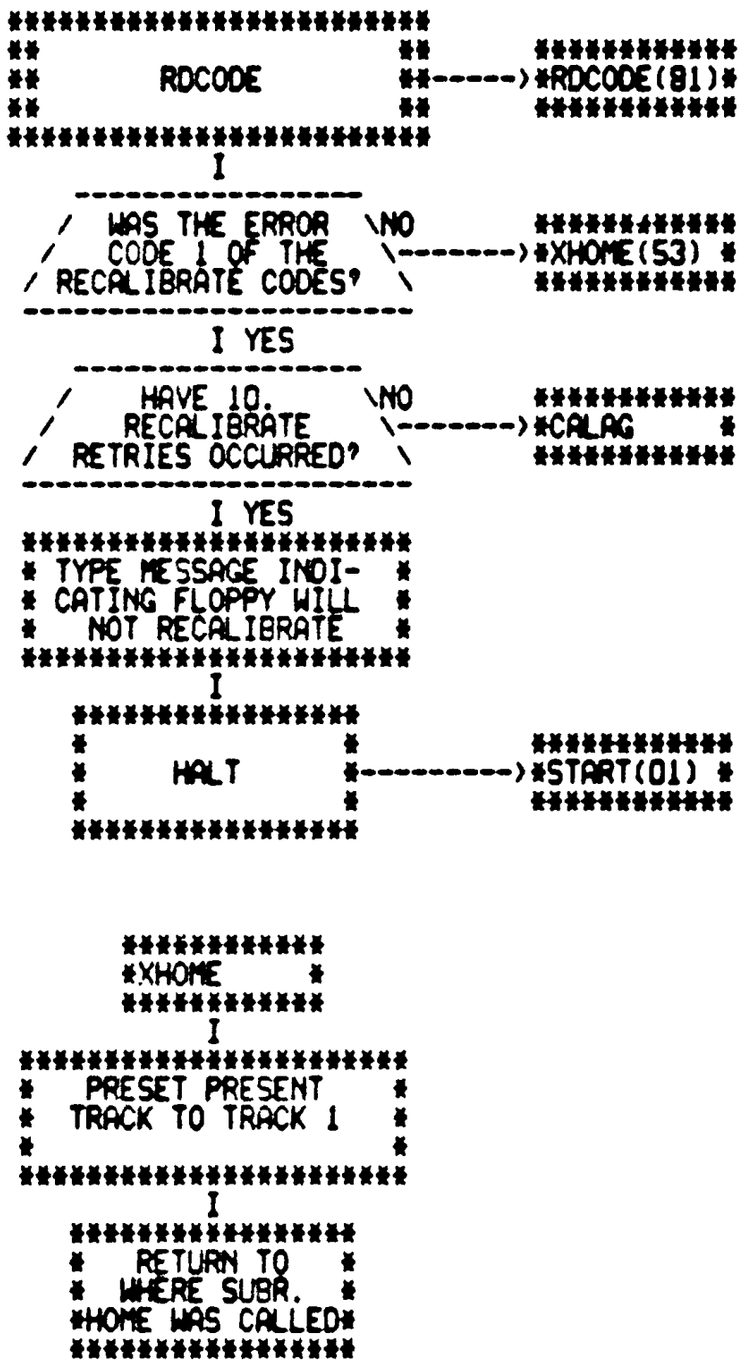

#HRDCRC
UILOG
UILOG(63)
UPDATE THE 'HARD CRC' ERROR LOG
INDICATE A 'HARD CRC' #ERROR OCCURRED
HALT
NOTE: PROGRAM MUST BE RESTARTED AT THIS POINT
#DATCRC
UILOG
UILOG(63)
UPDATE TRUE DATA 'CRC' ERROR LOG
IS SW(13) SET? (NO PRINT ERRORS)
NO
YES
INDICATE DATA CRC ERROR OCCURRED

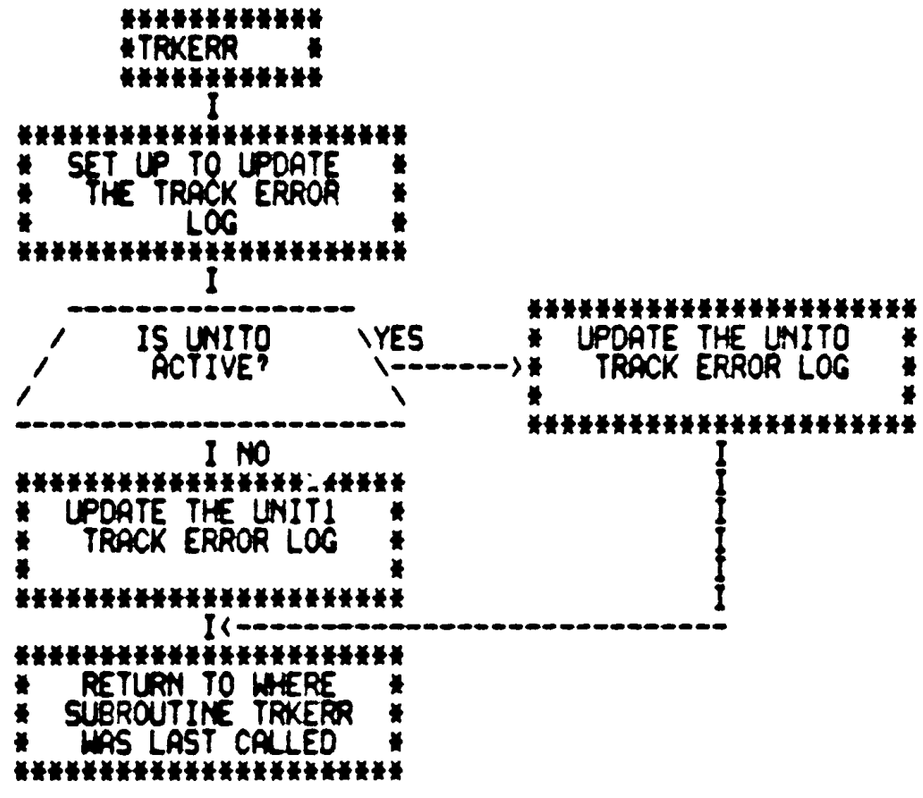

IS SW(15) SET? (HALT ON ERROR)
YES
HALT
NO
IS SW(11) SET? (NO RETRYS)
YES
NO
HAVE 10. CRC ERRORS BEEN LOGGED?
NO
REREAD(42)
YES
UILOG
UILOG(63)
UPDATE HARD CRC ERROR LOG
RESET READ RETRY COUNTER TO 10.
WAS THIS A READ AFTER A WRITE FUNCTION?
YES
CONT6(50)
NO
UPDATE THE HARD READ ERROR LOG
TYPE MESSAGE INDICATING UNRECOVERABLE READ ERROR

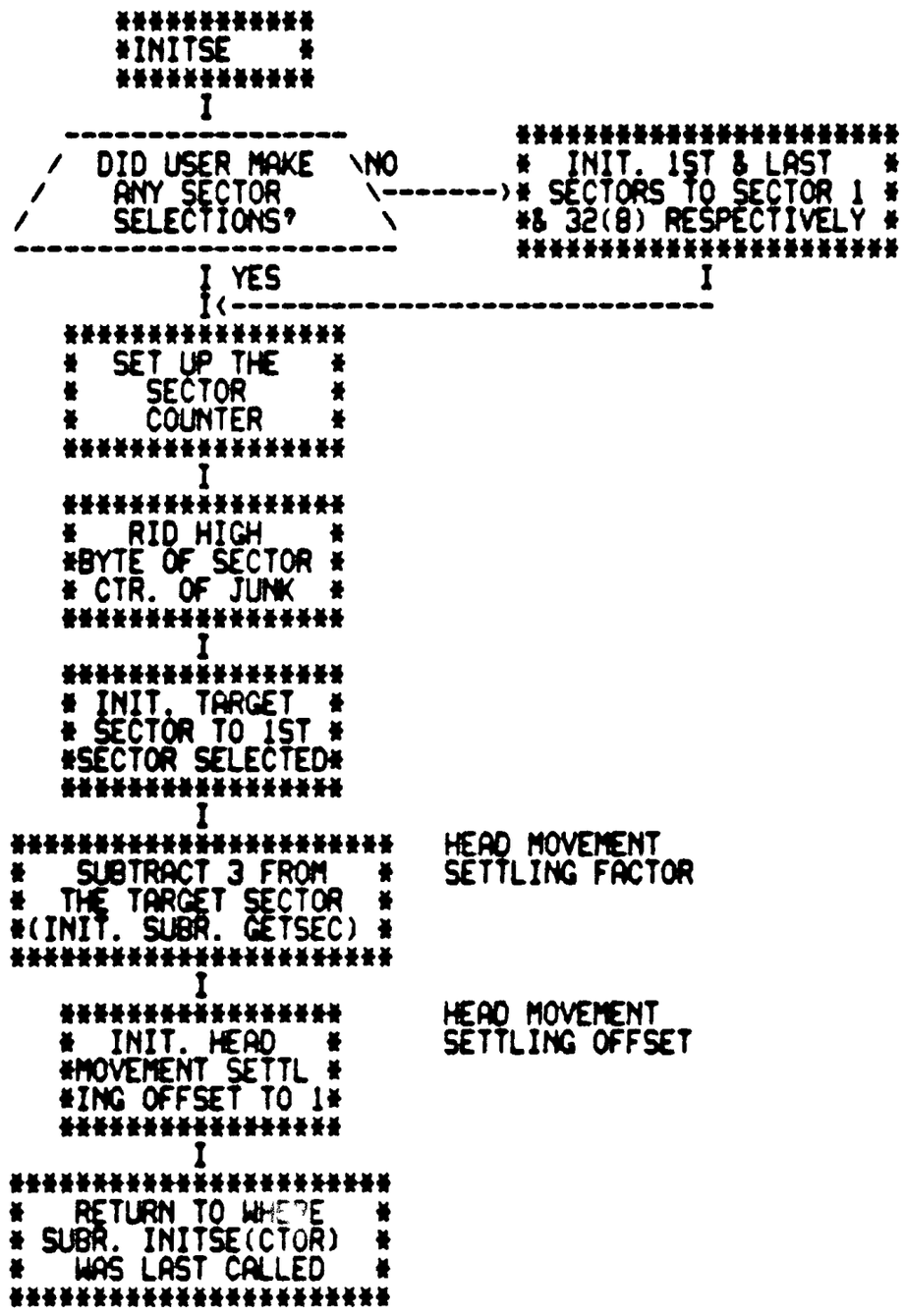

#CONT7
I
REMOVE THE GOOD RETURN FROM THE STACK
NEXTRD
CONT6
IS SW(11) SET? (NO RETRYS)
YES
WRHRD(51)
I NO
HAVE 10. REWRITES OCCURRED?
YES
WRHRD(51)
I NO
SET THE WRITE CAUSED BY ERROR FLAG
REMOVE THE GOOD RETURN FROM THE STACK
ONEWRT

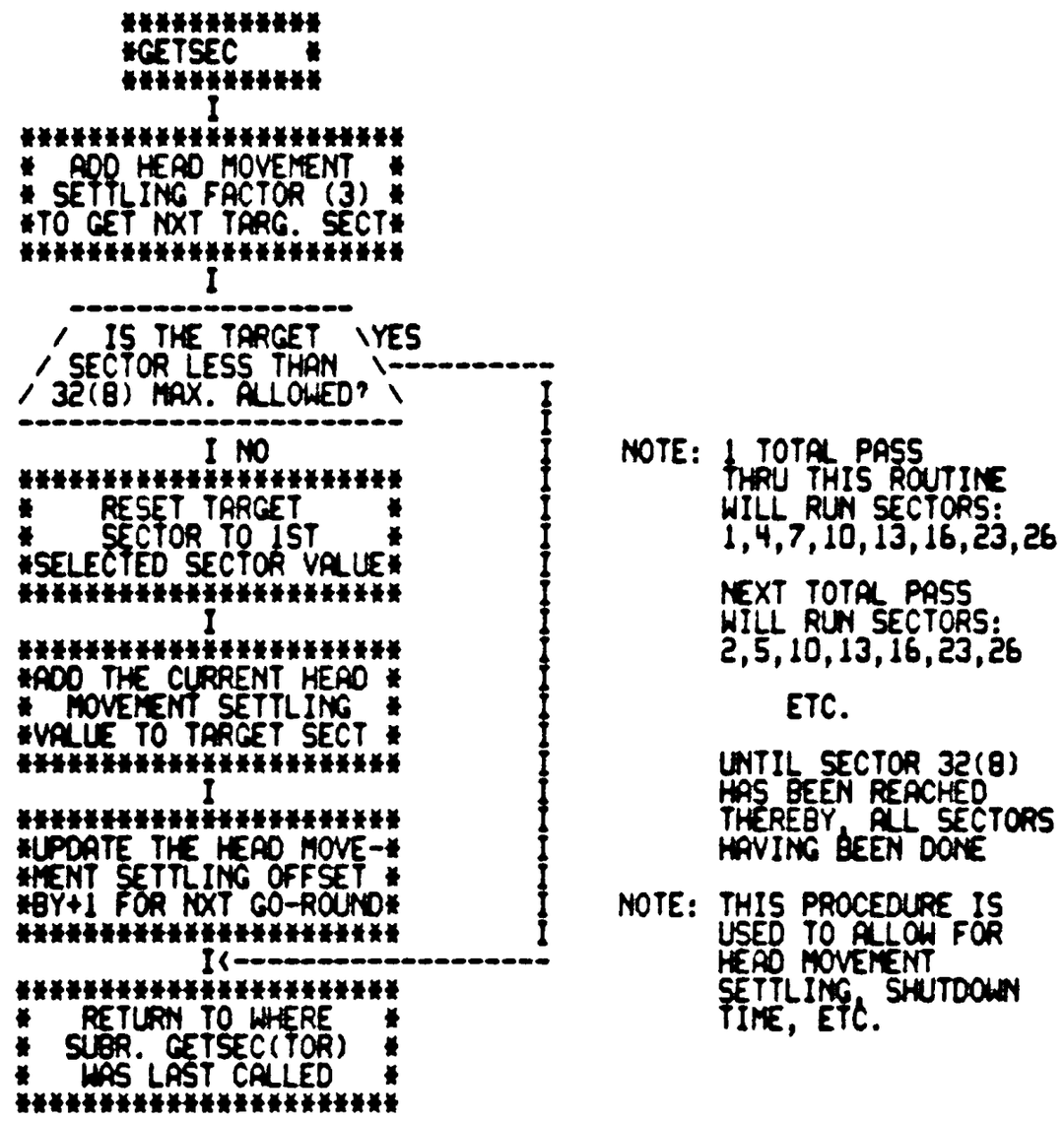

#WRHRD #
UILOG
#UILOG(63) #
UPDATE THE HARD WRITE ERROR LOG
#TYPE MSG. INDICATING A HARD UNRECOVERABLE WRITE ERROR OCCURRED #
#CONT7 #
#NOINT(27) #
#UPDATE INTERUPT DID NOT OCCUR ERROR LOG #
IS SW(13) SET? (NO PRINT ERRORS)
YES
NO
#TYPE MSG. INDICATING 'NO INTERRUPT ON DONE' OCCURRED #
IS SW(15) SET? (HALT ON ERROR)
YES
NO
HALT
#INTSER #
INTSERV

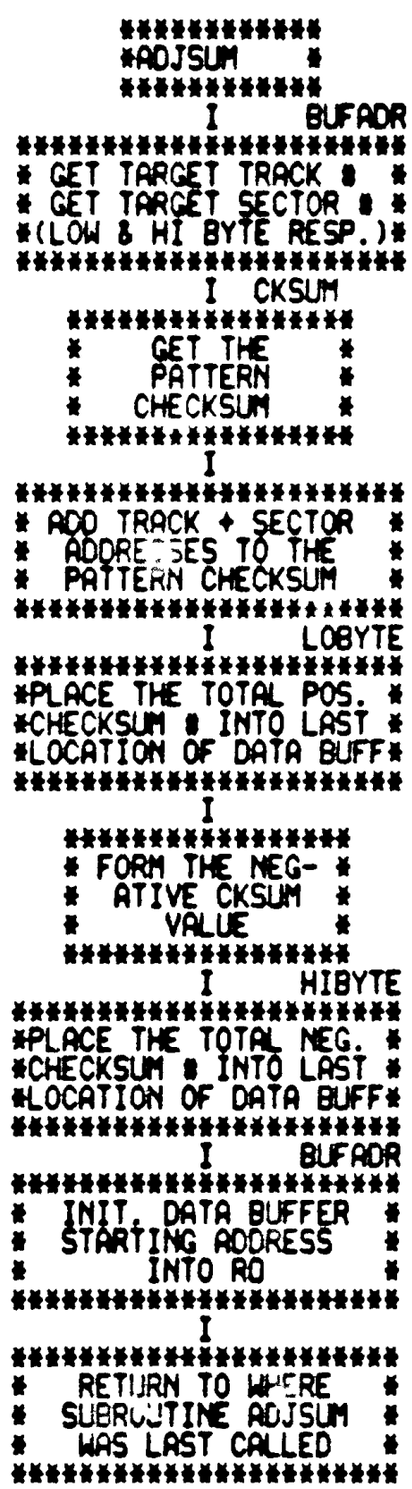

*HOME *
SET PARITY RETRY COUNTER TO 10.
SET DELETED DATA RETRY COUNTER TO 10.
*CALAG *
ISSUE A RECALIBRATE FUNCTION TO FLOPPY
DONECK
DONECK(61)
WAS THERE A PARITY ERROR?
NO
ERFLGS(52)
YES
STATER
STATER(65)
*ERFLGS *
IS THE ERROR FLAG SET?
NO
*XHOME(53) *
YES

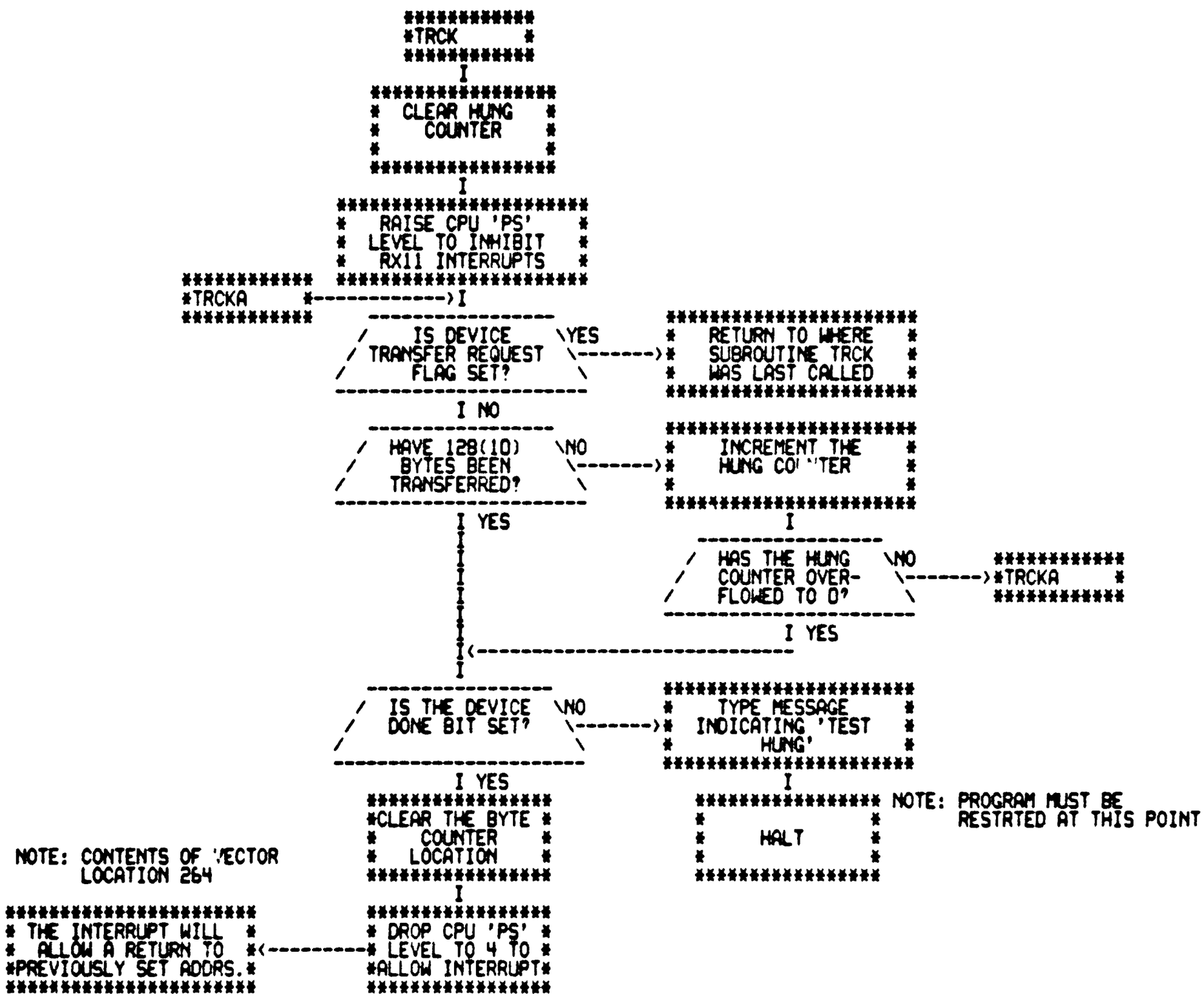

RDCODE
RDCODE(81)
WAS THE ERROR CODE 1 OF THE RECALIBRATE CODES?
NO
*XHOME(53) *
YES
HAVE 10. RECALIBRATE RETRIES OCCURRED?
NO
*CALAG *
YES
TYPE MESSAGE INDI- CATING FLOPPY WILL NOT RECALIBRATE
HALT
*START(01) *
*XHOME *
PRESET PRESENT TRACK TO TRACK 1
RETURN TO WHERE SUBR. *HOME WAS CALLED*

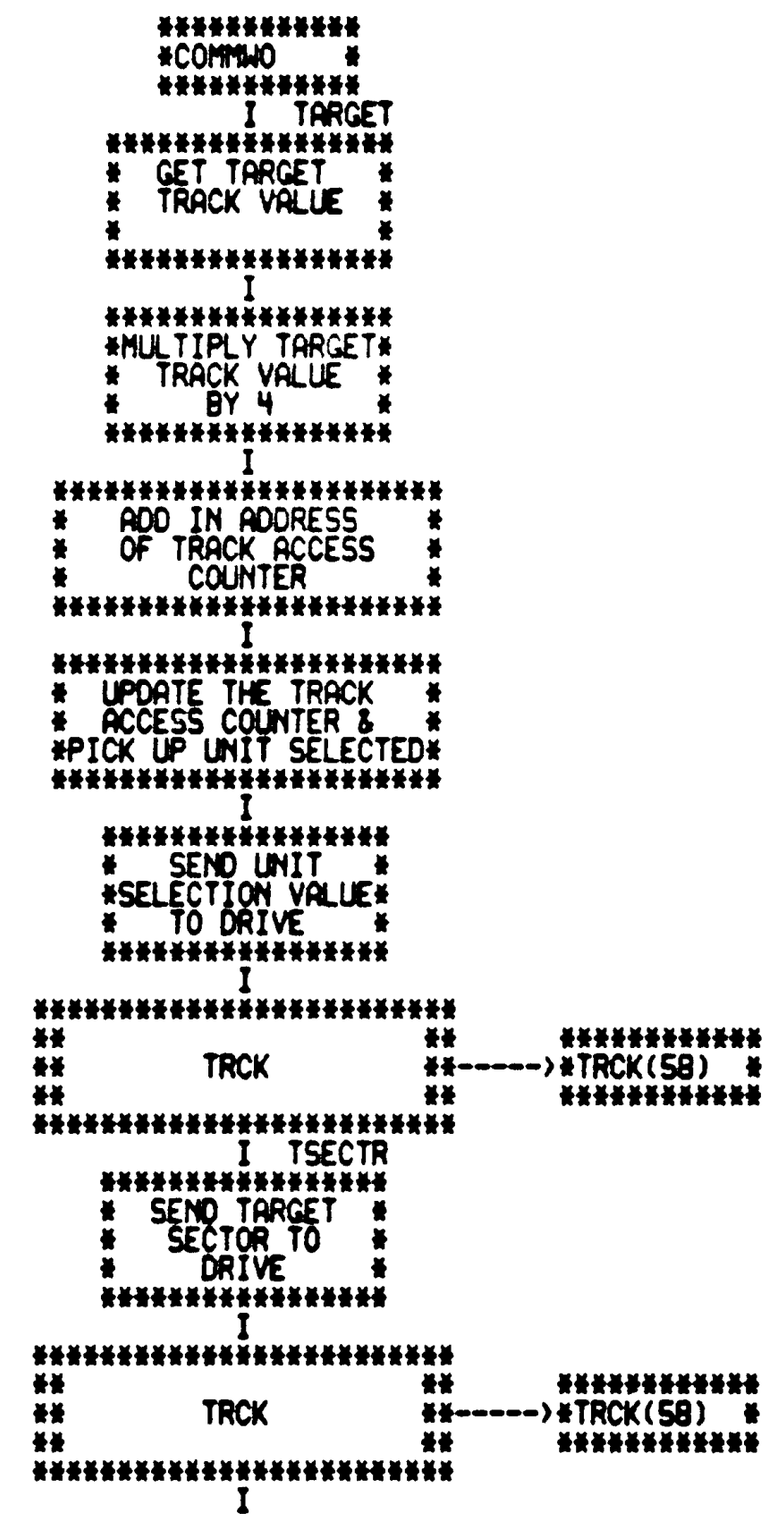

TRKERR
SET UP TO UPDATE THE TRACK ERROR LOG
IS UNIT0 ACTIVE?
YES
UPDATE THE UNIT0 TRACK ERROR LOG
NO
UPDATE THE UNIT1 TRACK ERROR LOG
RETURN TO WHERE SUBROUTINE TRKERR WAS LAST CALLED

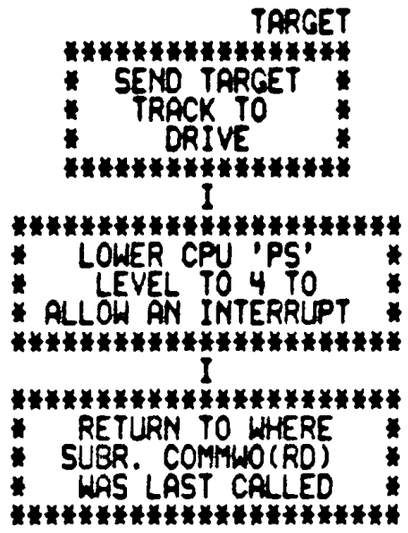

INITSE
DID USER MAKE ANY SECTOR SELECTIONS?
NO
INIT. 1ST & LAST SECTORS TO SECTOR 1 & 32(8) RESPECTIVELY
YES
SET UP THE SECTOR COUNTER
RID HIGH BYTE OF SECTOR CTR. OF JUNK
INIT. TARGET SECTOR TO 1ST SECTOR SELECTED
SUBTRACT 3 FROM THE TARGET SECTOR #(INIT. SUBR. GETSEC)
HEAD MOVEMENT SETTLING FACTOR
INIT. HEAD #MOVEMENT SETTL #ING OFFSET TO 1
HEAD MOVEMENT SETTLING OFFSET
RETURN TO WHERE SUBR. INITSE(CTOR) WAS LAST CALLED

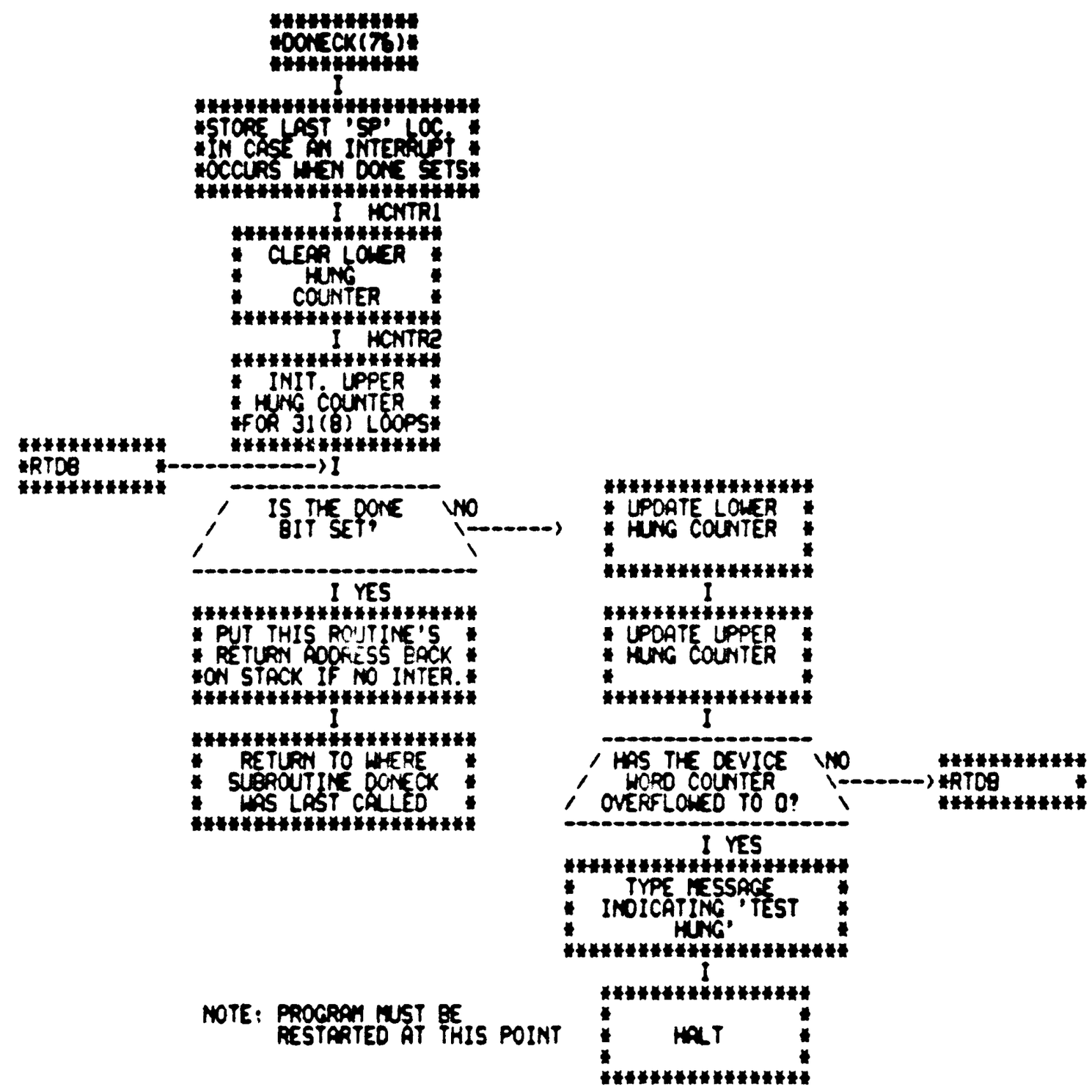

GETSEC
ADD HEAD MOVEMENT SETTLING FACTOR (3) TO GET NXT TARG. SECT
IS THE TARGET SECTOR LESS THAN 32(8) MAX. ALLOWED?
YES
NO
RESET TARGET SECTOR TO 1ST SELECTED SECTOR VALUE
ADD THE CURRENT HEAD MOVEMENT SETTLING VALUE TO TARGET SECT
UPDATE THE HEAD MOVEMENT SETTLING OFFSET BY+1 FOR NXT GO-ROUND
RETURN TO WHERE SUBR. GETSEC(TOR) WAS LAST CALLED
NOTE: 1 TOTAL PASS THRU THIS ROUTINE WILL RUN SECTORS: 1,4,7,10,13,16,23,26
NEXT TOTAL PASS WILL RUN SECTORS: 2,5,10,13,16,23,26
ETC.
UNTIL SECTOR 32(8) HAS BEEN REACHED THEREBY, ALL SECTORS HAVING BEEN DONE
NOTE: THIS PROCEDURE IS USED TO ALLOW FOR HEAD MOVEMENT SETTLING, SHUTDOWN TIME, ETC.

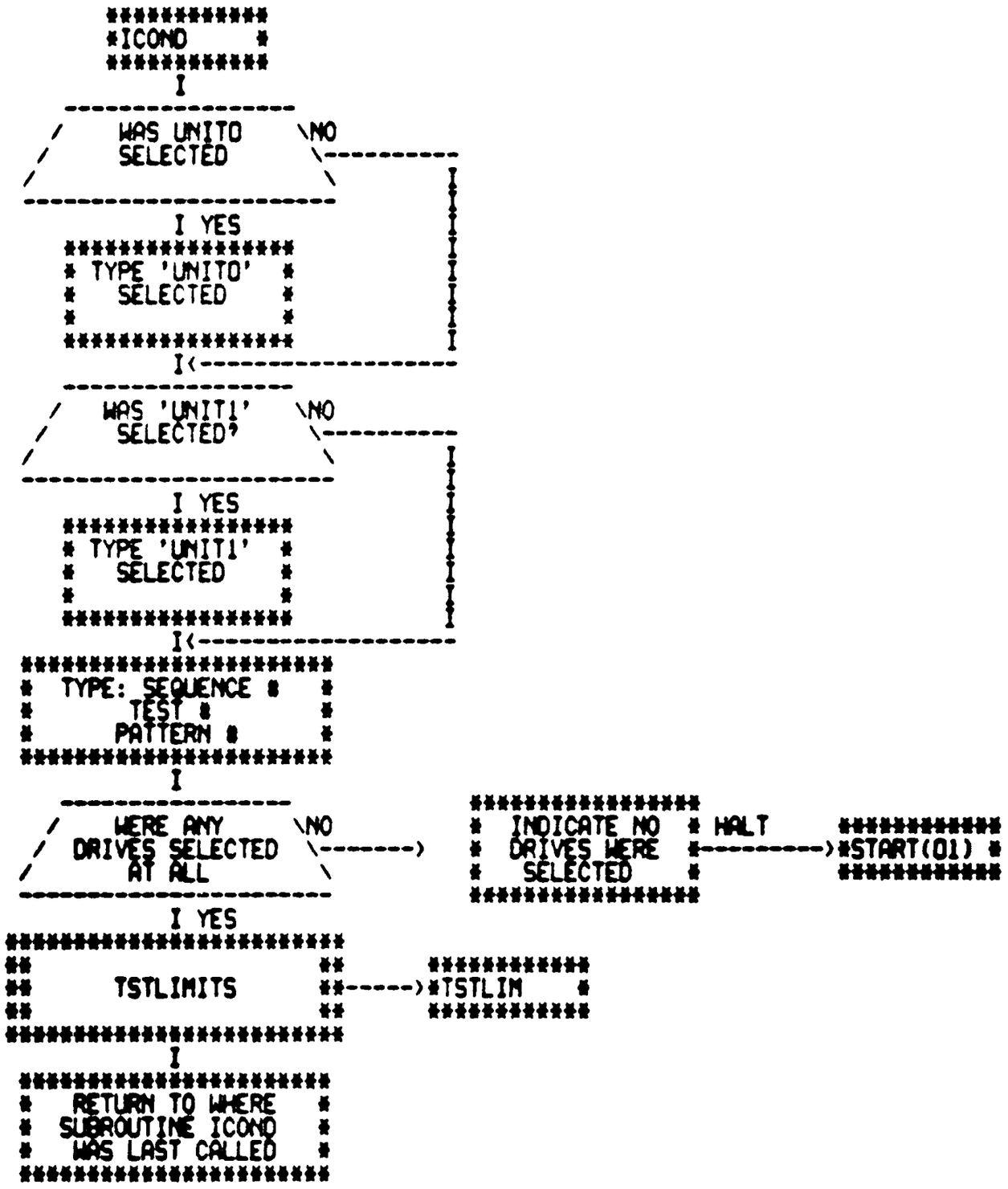

ADJSUM
BUFADR
GET TARGET TRACK #
GET TARGET SECTOR #
(LOW & HI BYTE RESP.)
CKSUM
GET THE PATTERN CHECKSUM
ADD TRACK + SECTOR ADDRESSES TO THE PATTERN CHECKSUM
LOBYTE
PLACE THE TOTAL POS. CHECKSUM # INTO LAST LOCATION OF DATA BUFF
FORM THE NEG- ATIVE CKSUM VALUE
HIBYTE
PLACE THE TOTAL NEG. CHECKSUM # INTO LAST LOCATION OF DATA BUFF
BUFADR
INIT. DATA BUFFER STARTING ADDRESS INTO R0
RETURN TO WHERE SUBROUTINE ADJSUM WAS LAST CALLED

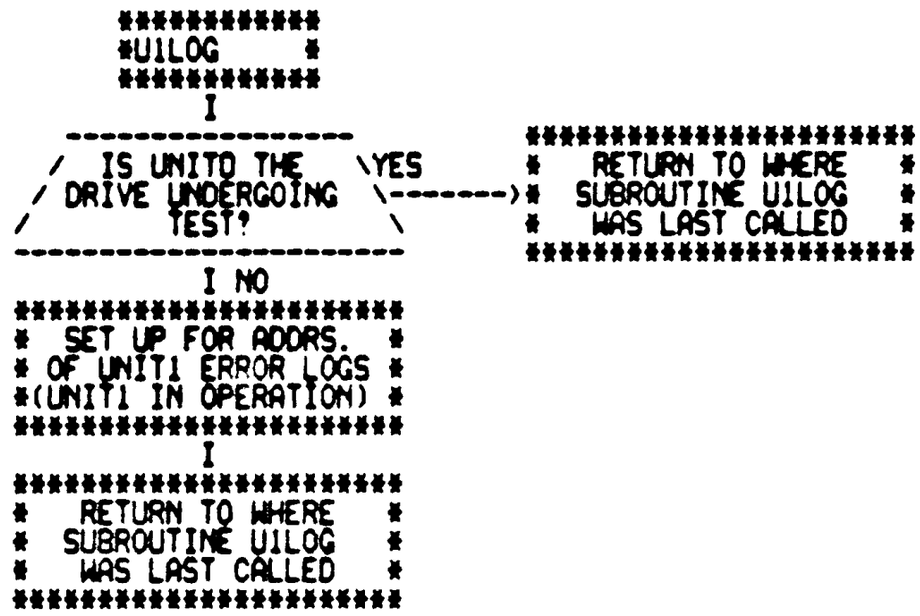

TRCK
CLEAR HUNG COUNTER
RAISE CPU 'PS' LEVEL TO INHIBIT RX11 INTERRUPTS
TRCKA
IS DEVICE TRANSFER REQUEST FLAG SET?
YES
RETURN TO WHERE SUBROUTINE TRCK WAS LAST CALLED
NO
HAVE 128(10) BYTES BEEN TRANSFERRED?
NO
INCREMENT THE HUNG COUNTER
HAS THE HUNG COUNTER OVER-FLOWED TO 0?
NO
TRCKA
YES
IS THE DEVICE DONE BIT SET?
NO
TYPE MESSAGE INDICATING 'TEST HUNG'
HALT
NOTE: PROGRAM MUST BE RESTRTED AT THIS POINT
YES
CLEAR THE BYTE COUNTER LOCATION
DROP CPU 'PS' LEVEL TO 4 TO ALLOW INTERRUPT
THE INTERRUPT WILL ALLOW A RETURN TO PREVIOUSLY SET ADDRS.
NOTE: CONTENTS OF VECTOR LOCATION 264


COMMWO
I TARGET
GET TARGET TRACK VALUE
MULTIPLY TARGET TRACK VALUE BY 4
ADD IN ADDRESS OF TRACK ACCESS COUNTER
UPDATE THE TRACK ACCESS COUNTER & PICK UP UNIT SELECTED
SEND UNIT SELECTION VALUE TO DRIVE
TRCK
TRCK(58)
I TSECTR
SEND TARGET SECTOR TO DRIVE
TRCK
TRCK(58)

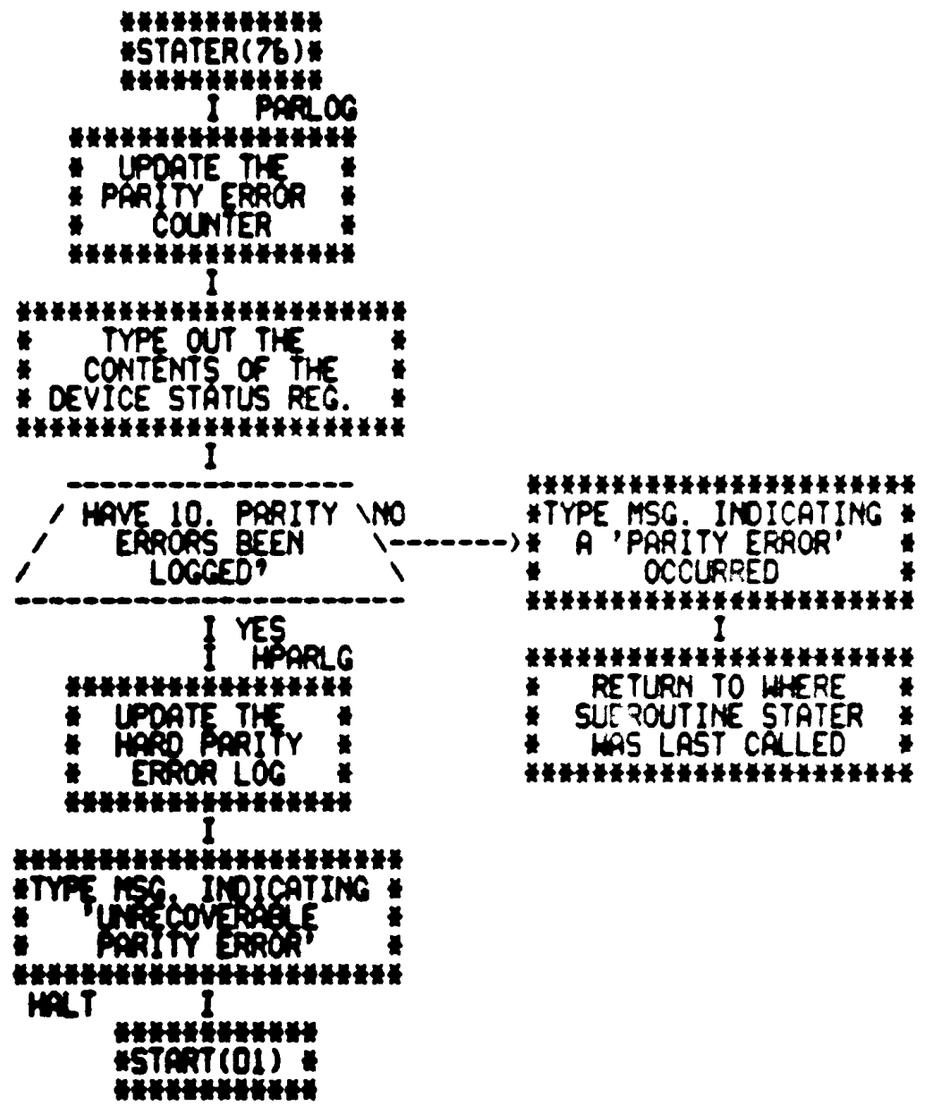

TARGET
SEND TARGET TRACK TO DRIVE
LOWER CPU 'PS' LEVEL TO 4 TO ALLOW AN INTERRUPT
RETURN TO WHERE SUBR. COMMWO(RD) WAS LAST CALLED

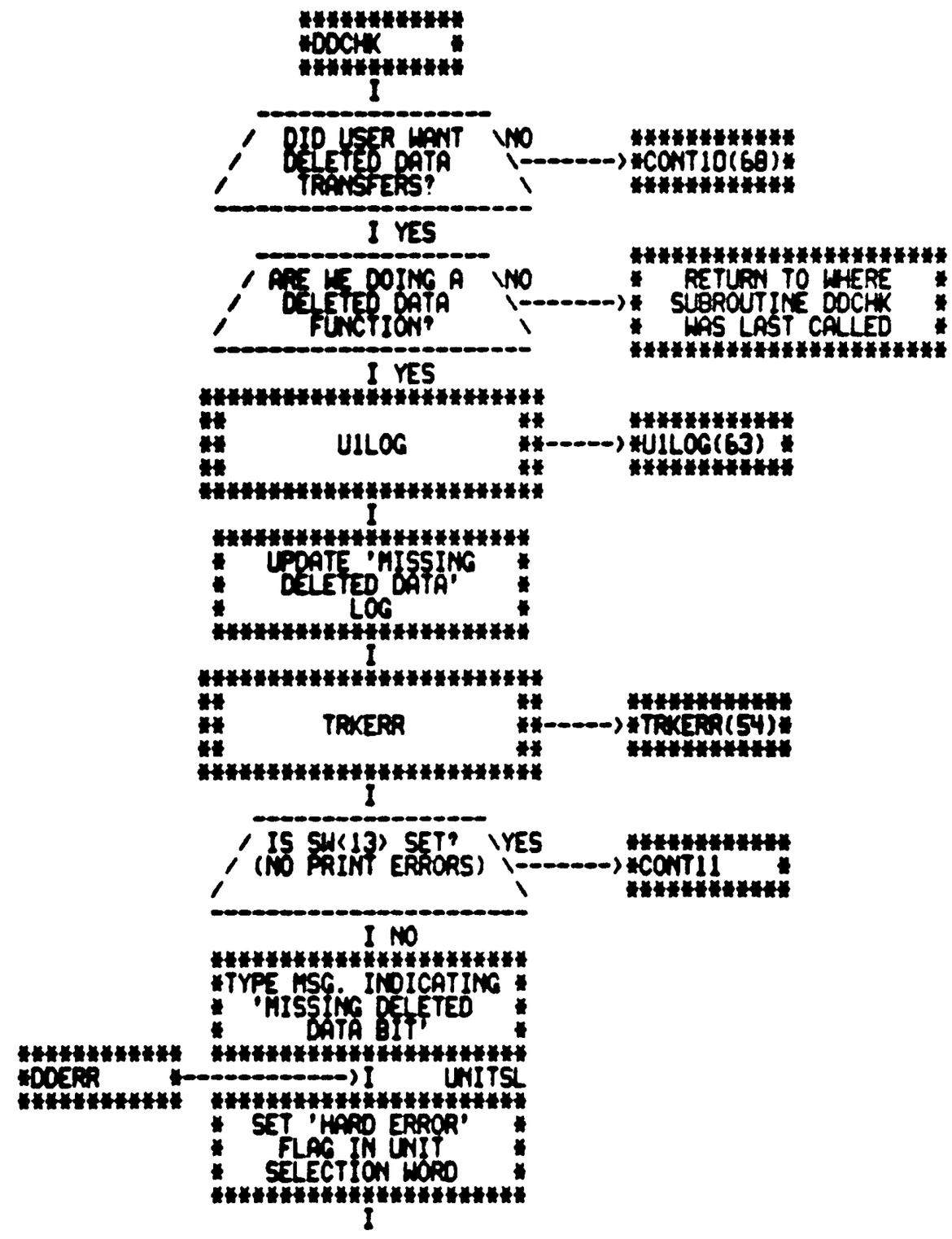

#DONECK(7%)#
#STORE LAST 'SP' LOC. #
#IN CASE AN INTERRUPT #
#OCCURS WHEN DONE SETS#
I HCNTR1
CLEAR LOWER
HUNG
COUNTER
I HCNTR2
INIT. UPPER
HUNG COUNTER
#FOR 31(8) LOOPS#
#RTDB#
IS THE DONE BIT SET?
NO
I YES
PUT THIS ROUTINE'S
RETURN ADDRESS BACK
#ON STACK IF NO INTER.#
RETURN TO WHERE
SUBROUTINE DONECK
WAS LAST CALLED
UPDATE LOWER
HUNG COUNTER
UPDATE UPPER
HUNG COUNTER
HAS THE DEVICE WORD COUNTER OVERFLOWED TO 0?
NO
#RTDB#
TYPE MESSAGE
INDICATING 'TEST
HUNG'
HALT
NOTE: PROGRAM MUST BE RESTARTED AT THIS POINT

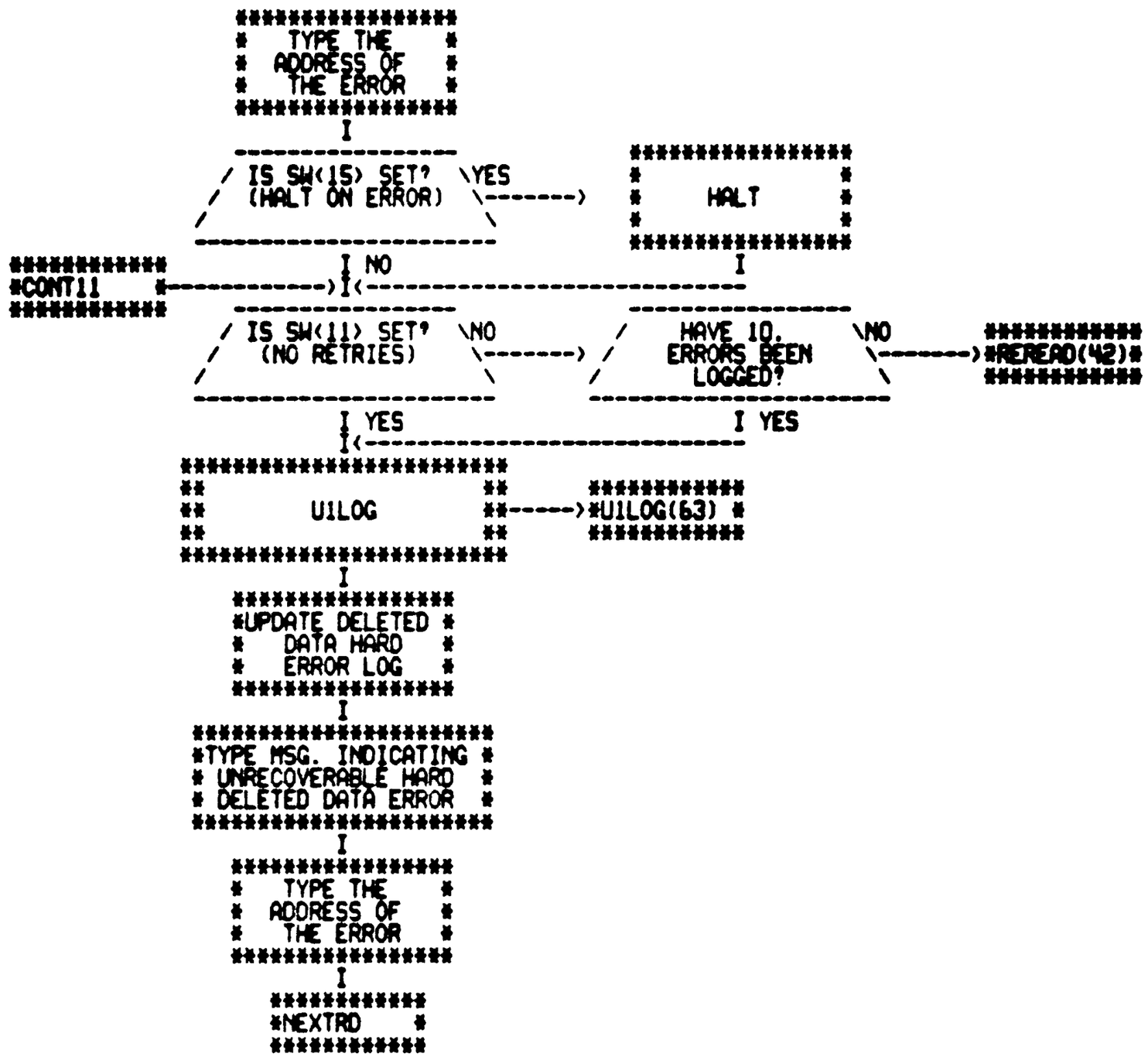

ICOND
WAS UNIT0 SELECTED
NO
YES
TYPE 'UNIT0' SELECTED
WAS 'UNIT1' SELECTED?
NO
YES
TYPE 'UNIT1' SELECTED
TYPE: SEQUENCE # TEST # PATTERN #
WERE ANY DRIVES SELECTED AT ALL
NO
HALT
INDICATE NO DRIVES WERE SELECTED
START(01)
YES
TSTLIMITS
TSTLIM
RETURN TO WHERE SUBROUTINE ICOND WAS LAST CALLED

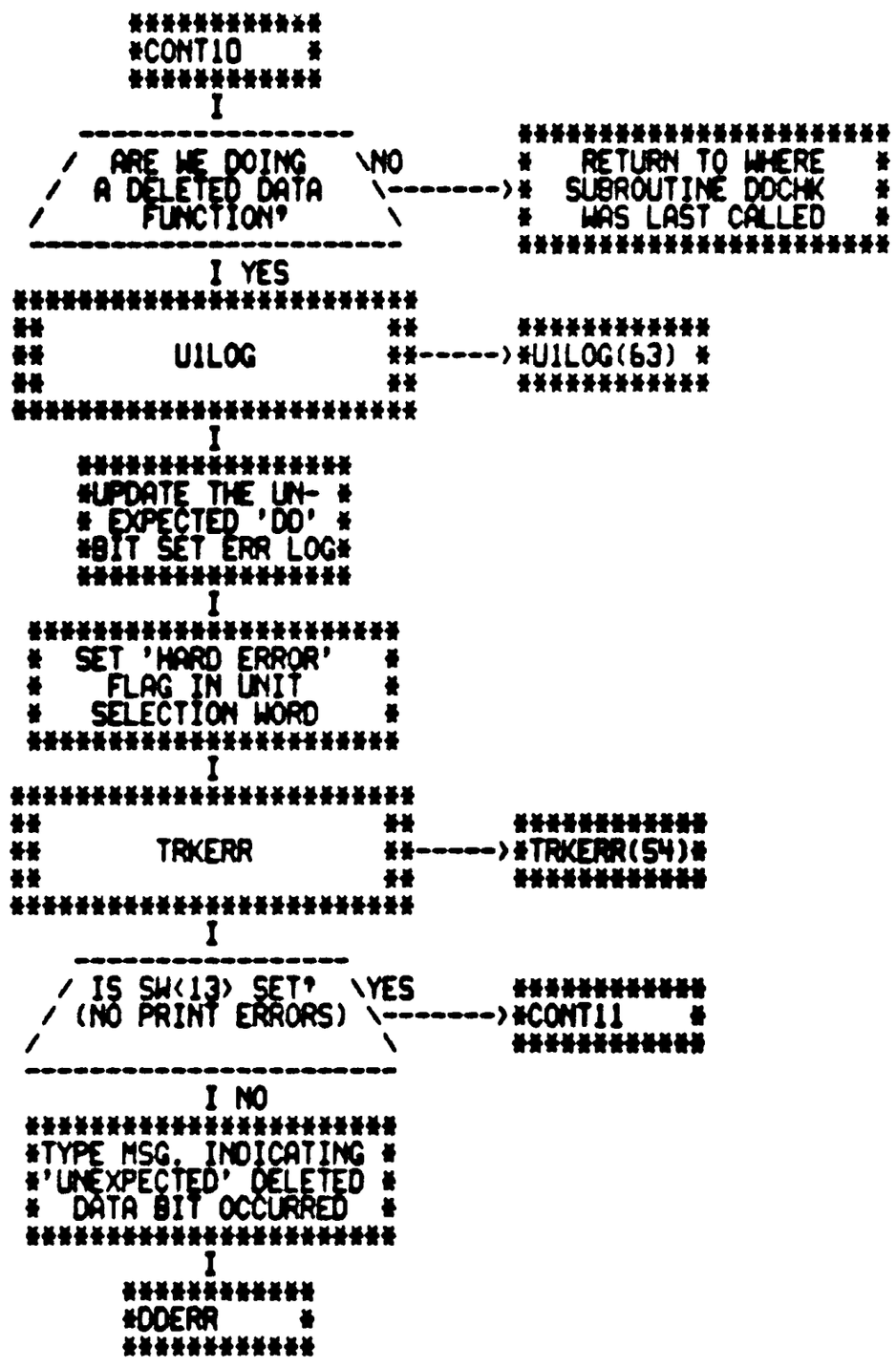

#UILOG #
IS UNIT0 THE DRIVE UNDERGOING TEST?
YES
RETURN TO WHERE SUBROUTINE UILOG WAS LAST CALLED
NO
SET UP FOR ADDRS. OF UNIT1 ERROR LOGS (UNIT1 IN OPERATION)
RETURN TO WHERE SUBROUTINE UILOG WAS LAST CALLED

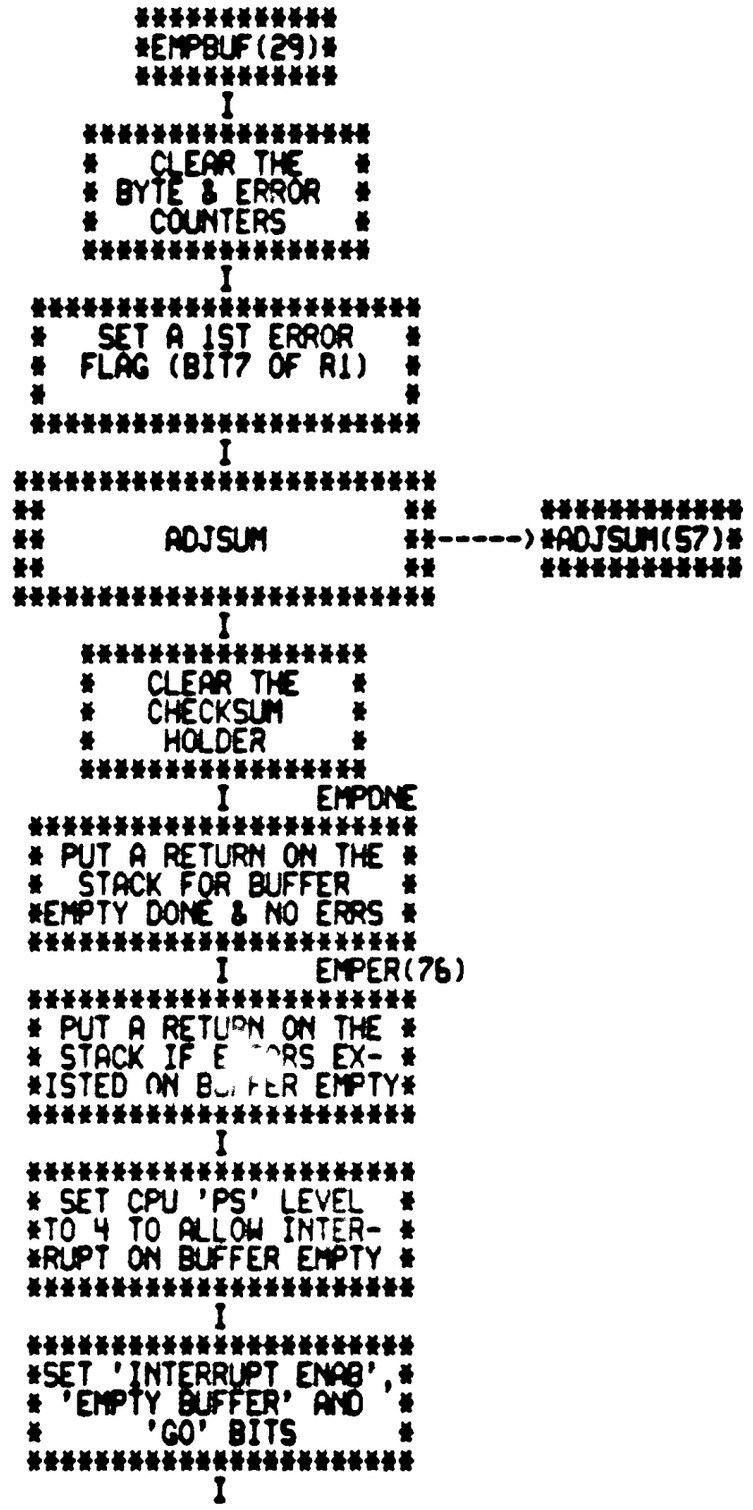

SEKTYP
TYPE MSG. INDICATING A 'SEEK ERROR' OCCURRED
TYPE THE ADDRESS OF THE TRACK FROM WHICH SEEK STARTED
RETURN TO WHERE SUBROUTINE SEKTYP WAS LAST CALLED

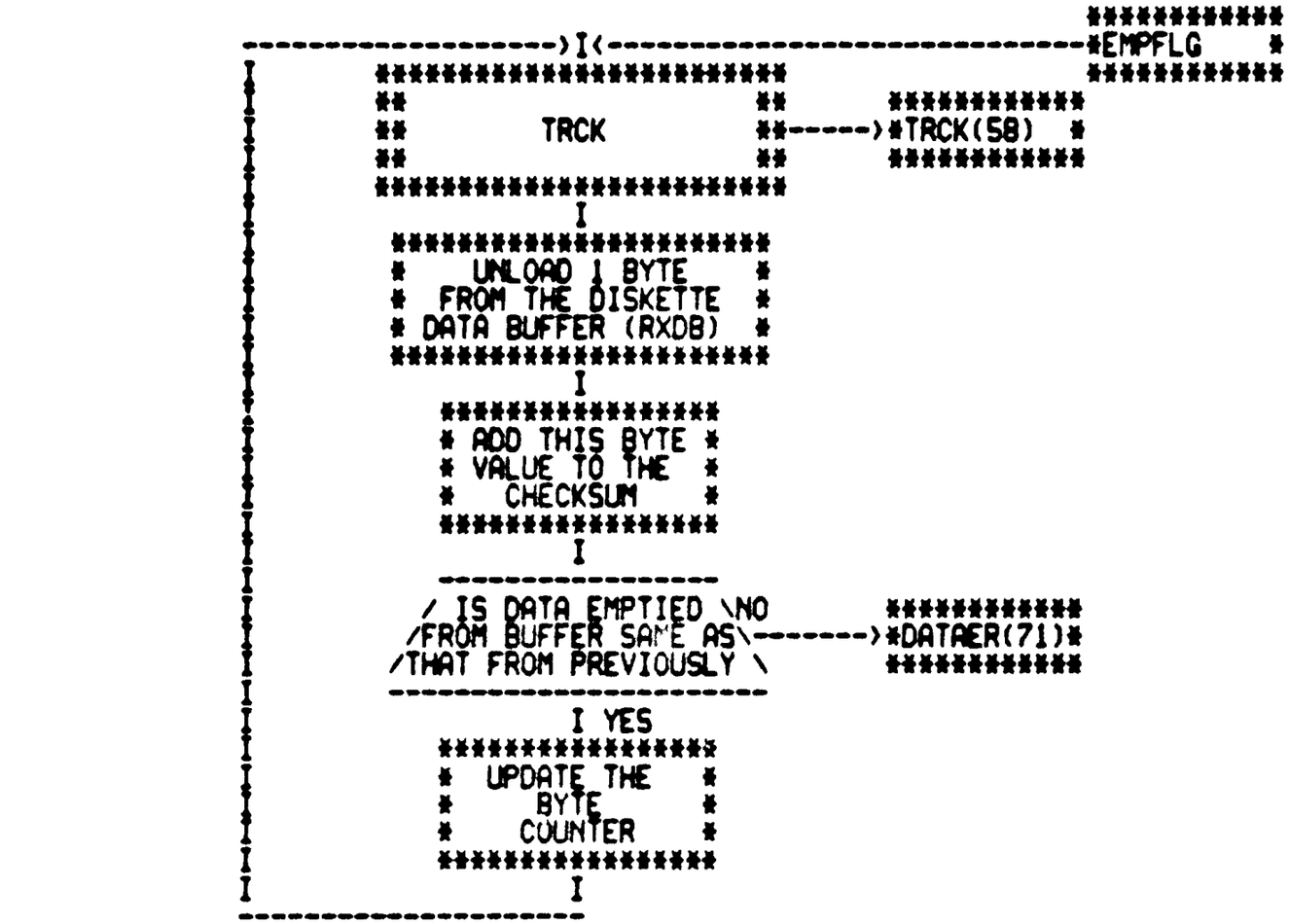

STATER(76)
I PARLOG
UPDATE THE PARITY ERROR COUNTER
TYPE OUT THE CONTENTS OF THE DEVICE STATUS REG.
HAVE 10. PARITY ERRORS BEEN LOGGED?
NO
TYPE MSG. INDICATING A 'PARITY ERROR' OCCURRED
RETURN TO WHERE SUBROUTINE STATER WAS LAST CALLED
YES
I HPARLG
UPDATE THE HARD PARITY ERROR LOG
TYPE MSG. INDICATING 'UNRECOVERABLE PARITY ERROR'
HALT
*START(01) *


#DDCHK #
DID USER WANT DELETED DATA TRANSFERS?
NO
#CONT10(68)#
YES
ARE WE DOING A DELETED DATA FUNCTION?
NO
* RETURN TO WHERE SUBROUTINE DDCHK WAS LAST CALLED *
YES
UILOG
#UILOG(63) #
UPDATE 'MISSING DELETED DATA' LOG
TRKERR
#TRKERR(54)#
IS SW(13) SET? (NO PRINT ERRORS)
YES
#CONT11 #
NO
#TYPE MSG. INDICATING 'MISSING DELETED DATA BIT'
#DDERR #
UNITSL
SET 'HARD ERROR' FLAG IN UNIT SELECTION WORD

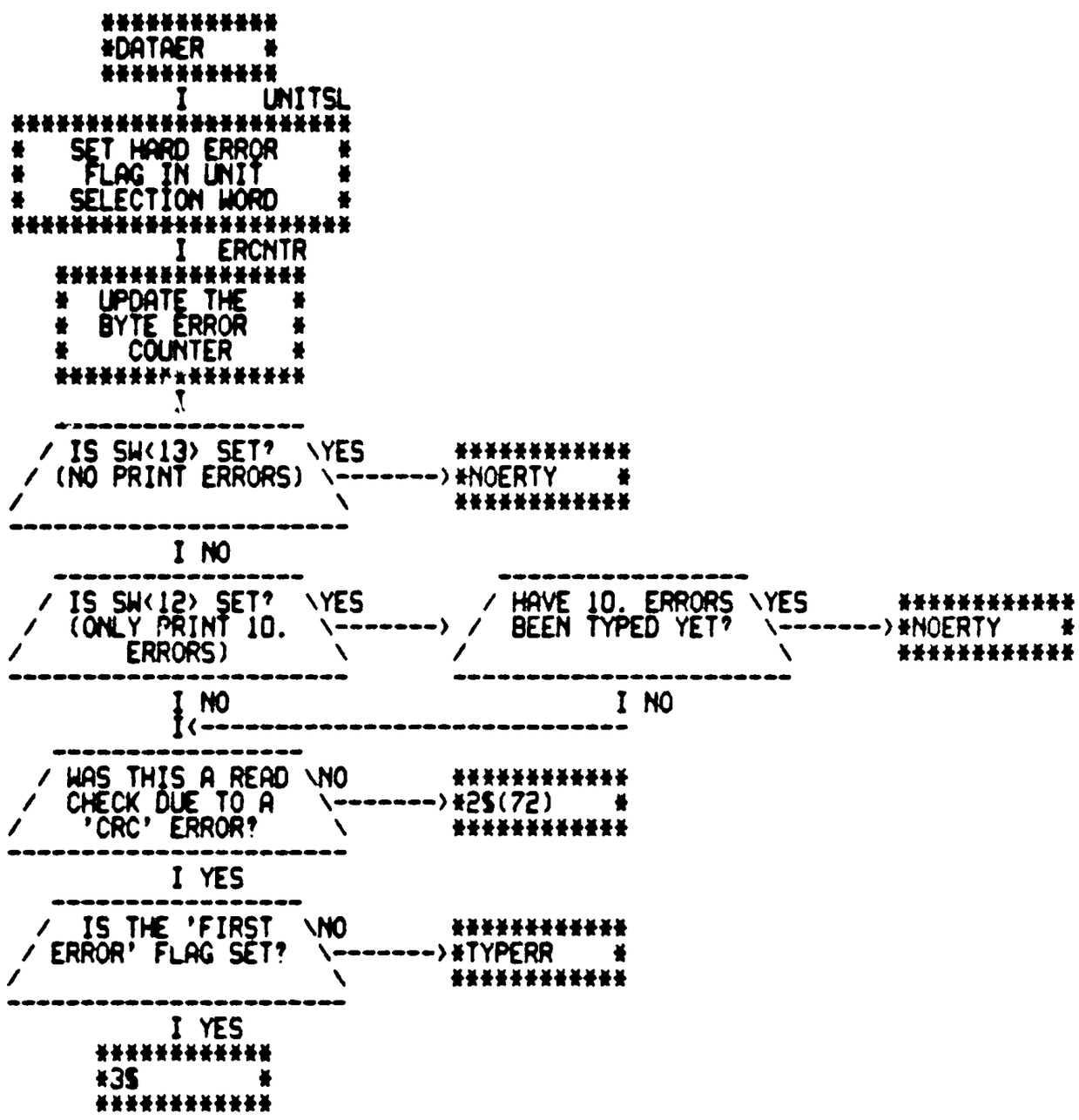

TYPE THE ADDRESS OF THE ERROR
IS SW(15) SET? (HALT ON ERROR)
YES
HALT
NO
#CONT11
IS SW(11) SET? (NO RETRIES)
NO
HAVE 10. ERRORS BEEN LOGGED?
NO
#REREAD(42)#
YES
YES
U1LOG
#U1LOG(63) #
UPDATE DELETED DATA HARD ERROR LOG
TYPE MSG. INDICATING UNRECOVERABLE HARD DELETED DATA ERROR
TYPE THE ADDRESS OF THE ERROR
#NEXTRD #

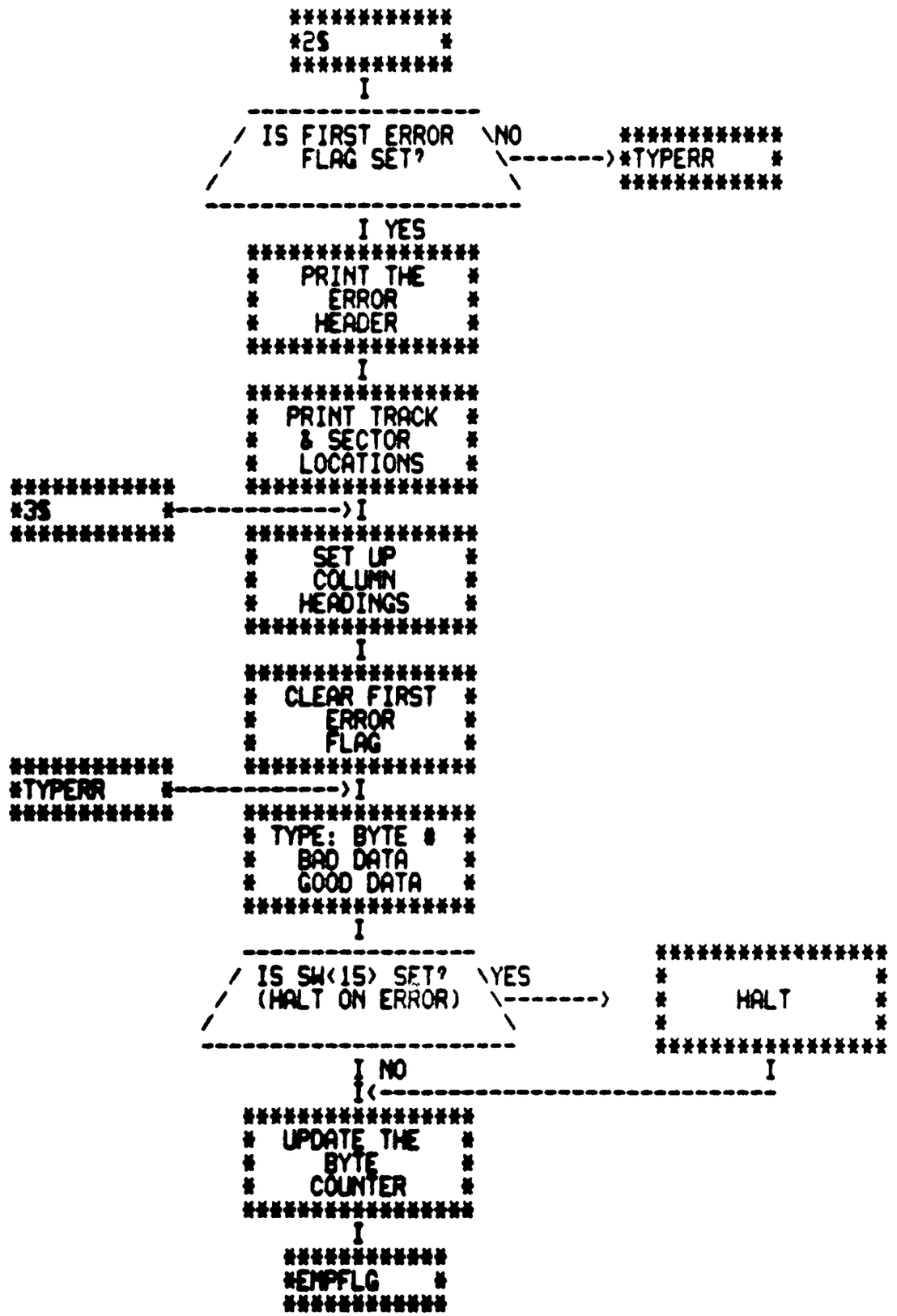

CONTIO
ARE WE DOING A DELETED DATA FUNCTION?
NO
RETURN TO WHERE SUBROUTINE DDCHK WAS LAST CALLED
YES
UILOG
UILOG(63)
UPDATE THE UN- EXPECTED 'DO' BIT SET ERR LOG
SET 'HARD ERROR' FLAG IN UNIT SELECTION WORD
TRKERR
TRKERR(54)
IS SW(13) SET? (NO PRINT ERRORS)
YES
CONTII
NO
TYPE MSG. INDICATING 'UNEXPECTED' DELETED DATA BIT OCCURRED
DDERR

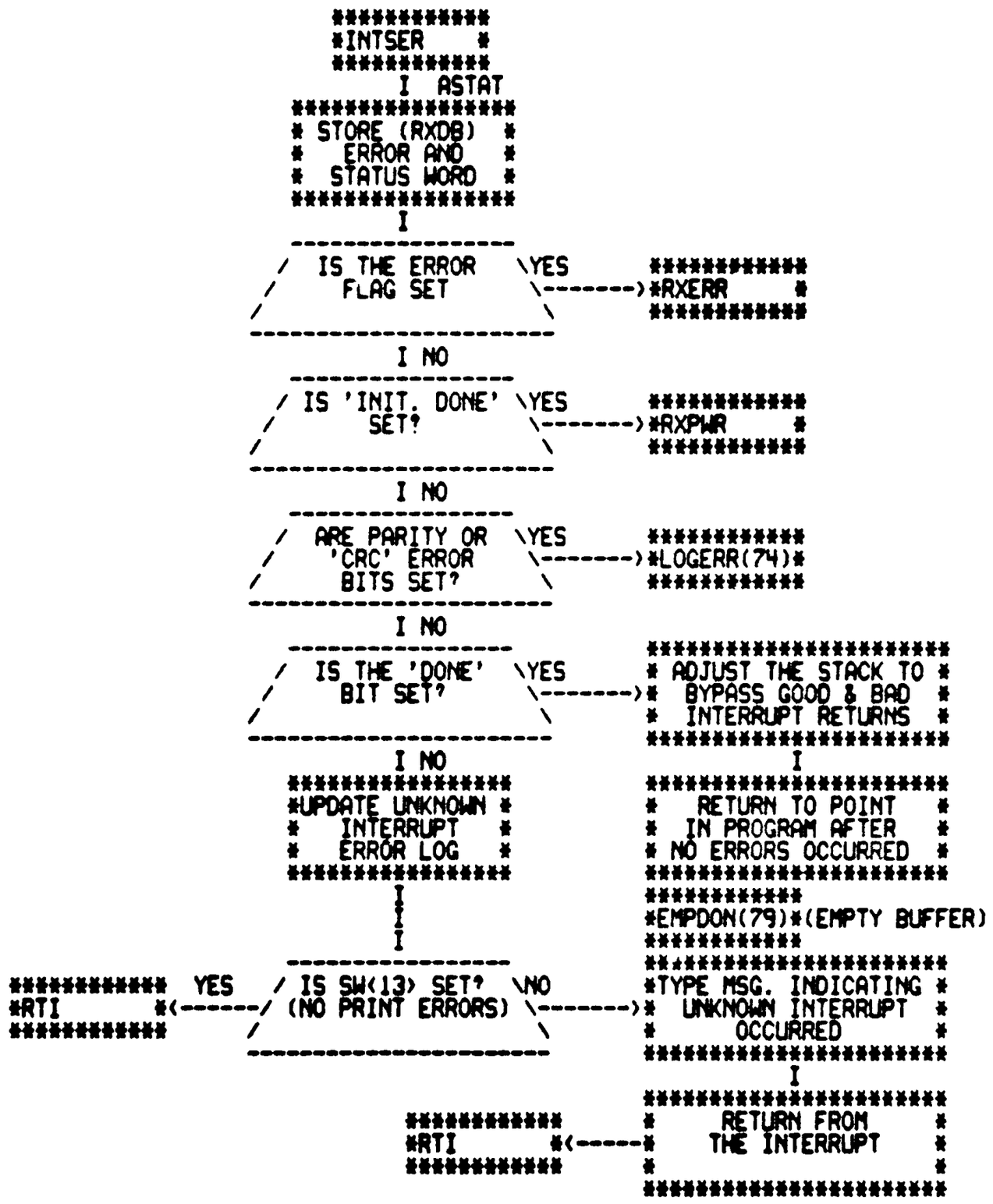

EMPBUF(29)
CLEAR THE BYTE & ERROR COUNTERS
SET A 1ST ERROR FLAG (BIT7 OF R1)
ADJSUM
ADJSUM(57)
CLEAR THE CHECKSUM HOLDER
EMPDNE
PUT A RETURN ON THE STACK FOR BUFFER EMPTY DONE & NO ERRS
EMPER(76)
PUT A RETURN ON THE STACK IF ERRS EX- ISTED ON BUFFER EMPTY
SET CPU 'PS' LEVEL TO 4 TO ALLOW INTER- RUPT ON BUFFER EMPTY
SET 'INTERRUPT ENAB', 'EMPTY BUFFER' AND 'GO' BITS

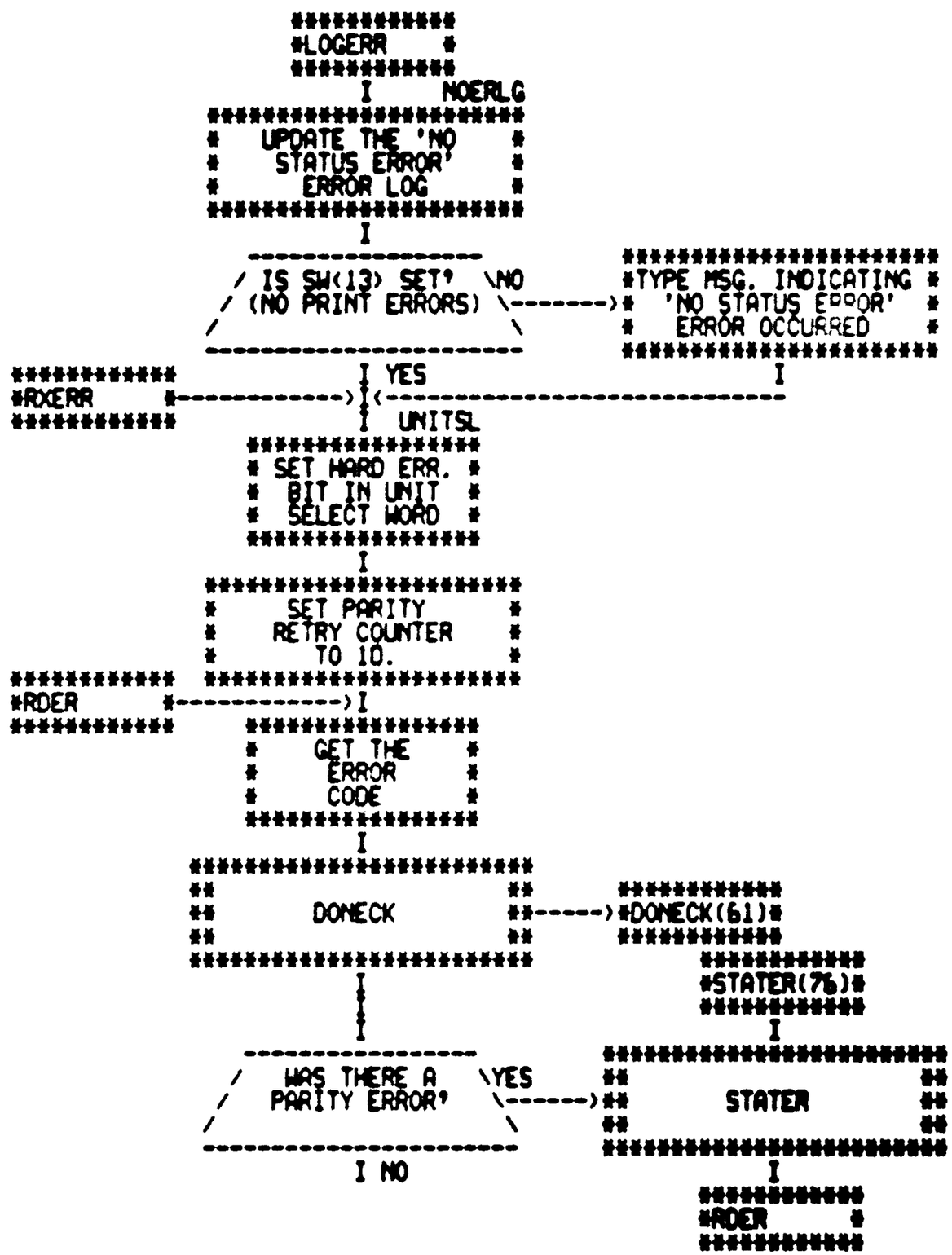


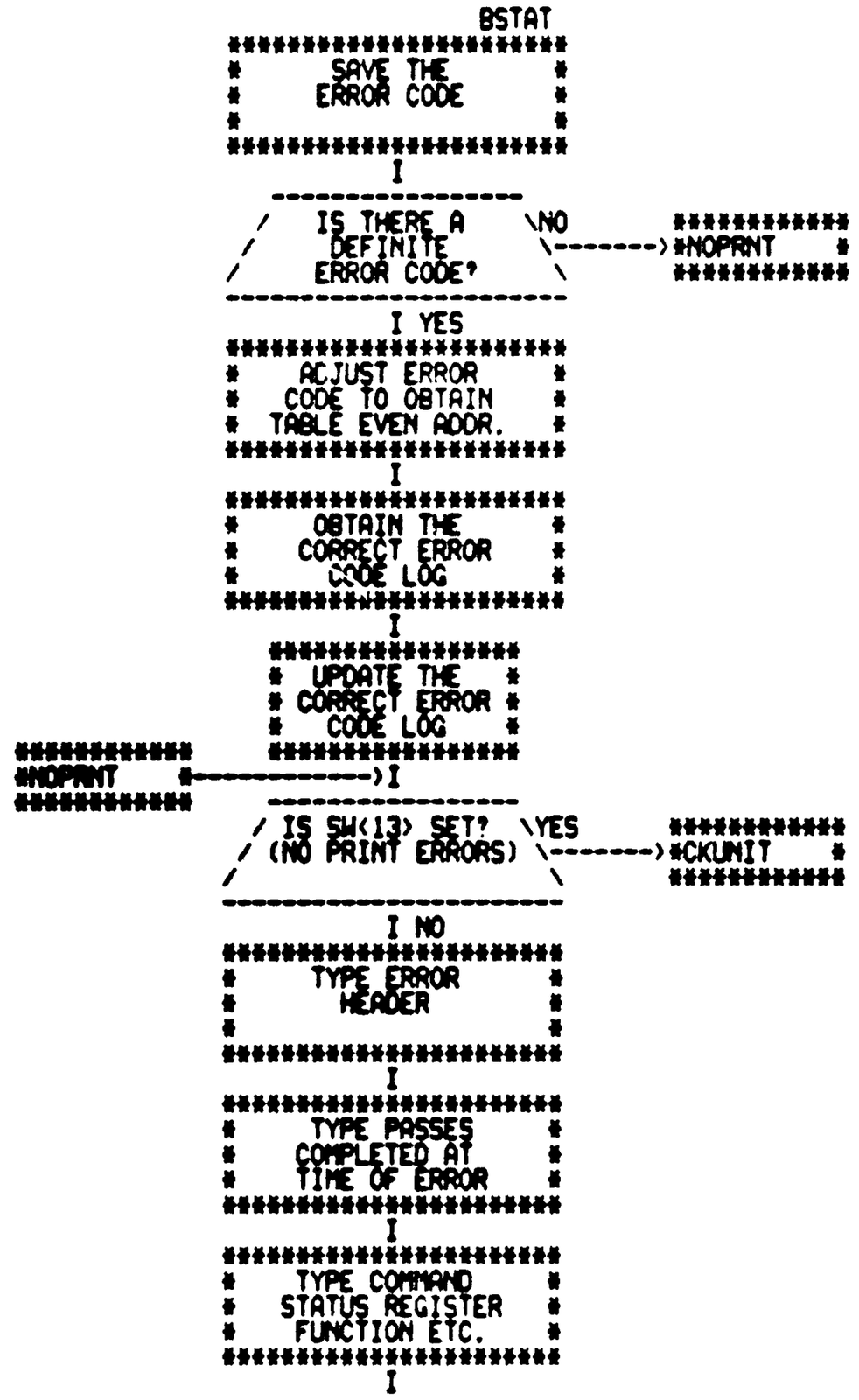
NOTE: THE LOOP FROM SUBROUTINE 'TRCK' TO 'UPDATE THE BYTE COUNTER' CONTINUES TILL 128(10) BYTES OF DATA HAVE BEEN FULFILLED, AT WHICH POINT IF THE 'IE' BIT HAD BEEN SET, AN INTERRUPT TO SUBROUTINE 'INTSER(V)' WOULD HAVE OCCURRED.

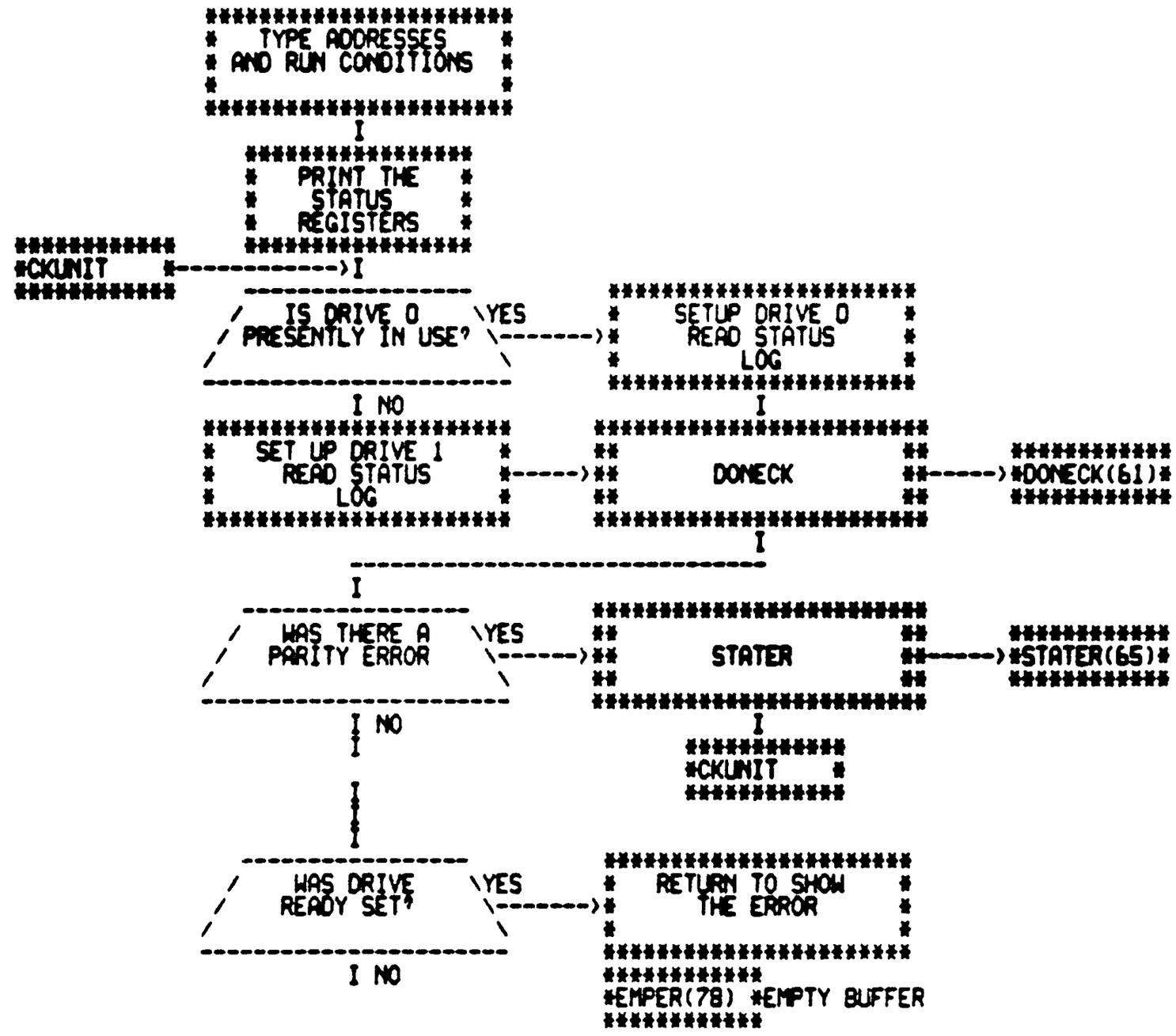

DATAER
UNITSL
SET HARD ERROR FLAG IN UNIT SELECTION WORD
ERCNTR
UPDATE THE BYTE ERROR COUNTER
IS SW(13) SET? (NO PRINT ERRORS)
YES
NOERTY
NO
IS SW(12) SET? (ONLY PRINT 10. ERRORS)
YES
NOERTY
HAVE 10. ERRORS BEEN TYPED YET?
YES
NOERTY
NO
NO
WAS THIS A READ CHECK DUE TO A 'CRC' ERROR?
NO
2$(72)
YES
IS THE 'FIRST ERROR' FLAG SET?
NO
TYPERR
YES
3$

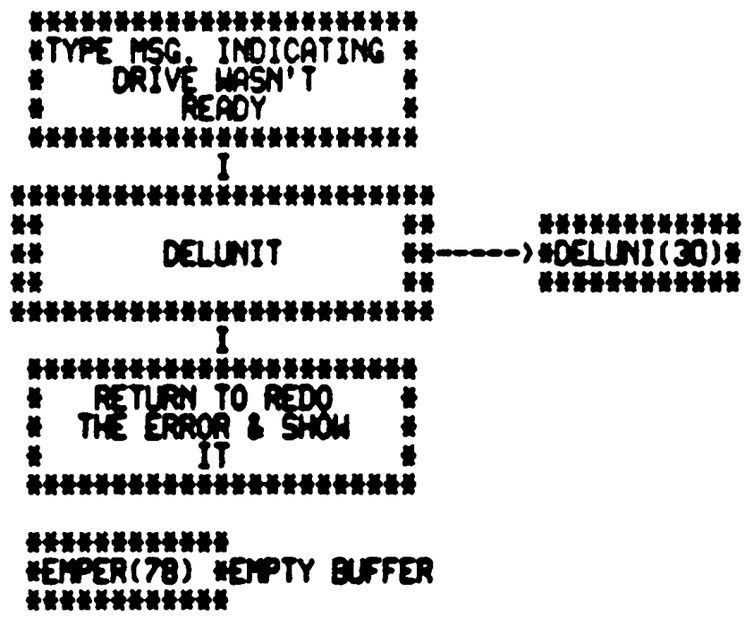

#2$
IS FIRST ERROR FLAG SET?
NO
#TYPERR
YES
PRINT THE ERROR HEADER
PRINT TRACK & SECTOR LOCATIONS
#3$
SET UP COLUMN HEADINGS
CLEAR FIRST ERROR FLAG
#TYPERR
TYPE: BYTE # BAD DATA GOOD DATA
IS SW(15) SET? (HALT ON ERROR)
YES
HALT
NO
UPDATE THE BYTE COUNTER
#EMPFLG

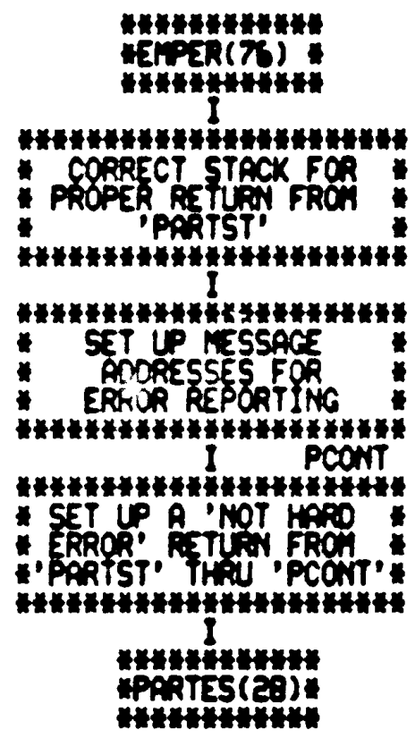

INTSER
I ASTAT
STORE (RXDB) ERROR AND STATUS WORD
IS THE ERROR FLAG SET
YES
RXERR
NO
IS 'INIT. DONE' SET?
YES
RXPWR
NO
ARE PARITY OR 'CRC' ERROR BITS SET?
YES
LOGERR(74)
NO
IS THE 'DONE' BIT SET?
YES
ADJUST THE STACK TO BYPASS GOOD & BAD INTERRUPT RETURNS
NO
UPDATE UNKNOWN INTERRUPT ERROR LOG
RETURN TO POINT IN PROGRAM AFTER NO ERRORS OCCURRED
EMPDON(79)(EMPTY BUFFER)
IS SW(13) SET? (NO PRINT ERRORS)
YES
RTI
NO
TYPE MSG. INDICATING UNKNOWN INTERRUPT OCCURRED
RETURN FROM THE INTERRUPT
RTI

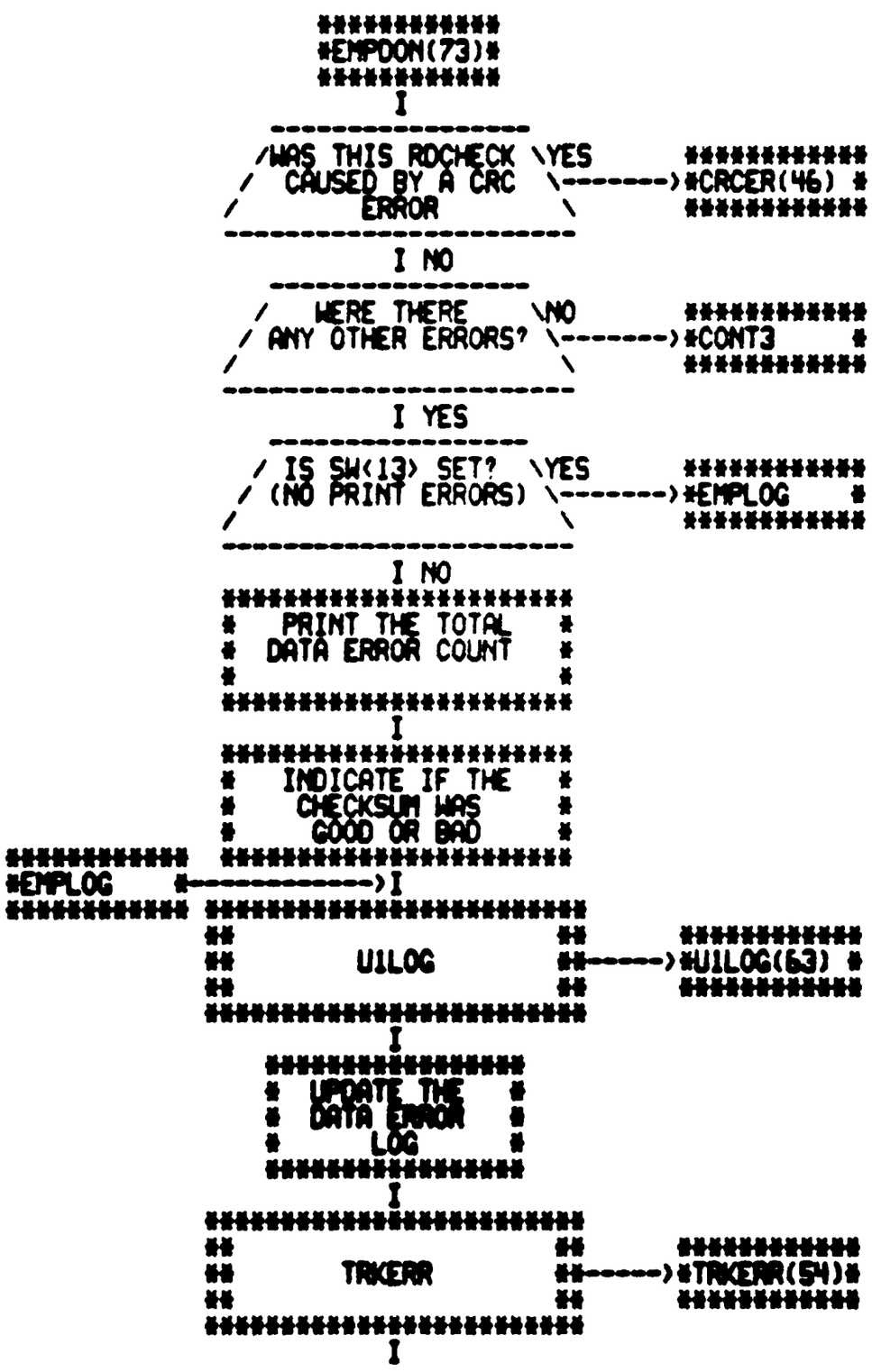

#LOGERR #
NOERLG
UPDATE THE 'NO STATUS ERROR' ERROR LOG
IS SW(13) SET? (NO PRINT ERRORS)
NO
TYPE MSG. INDICATING 'NO STATUS ERROR' ERROR OCCURRED
YES
#RXERR #
UNITSL
SET HARD ERR. BIT IN UNIT SELECT WORD
SET PARITY RETRY COUNTER TO 10.
#RDER #
GET THE ERROR CODE
DONECK
#DONECK(61)#
#STATER(7%)#
WAS THERE A PARITY ERROR?
YES
STATER
NO
#RDER #

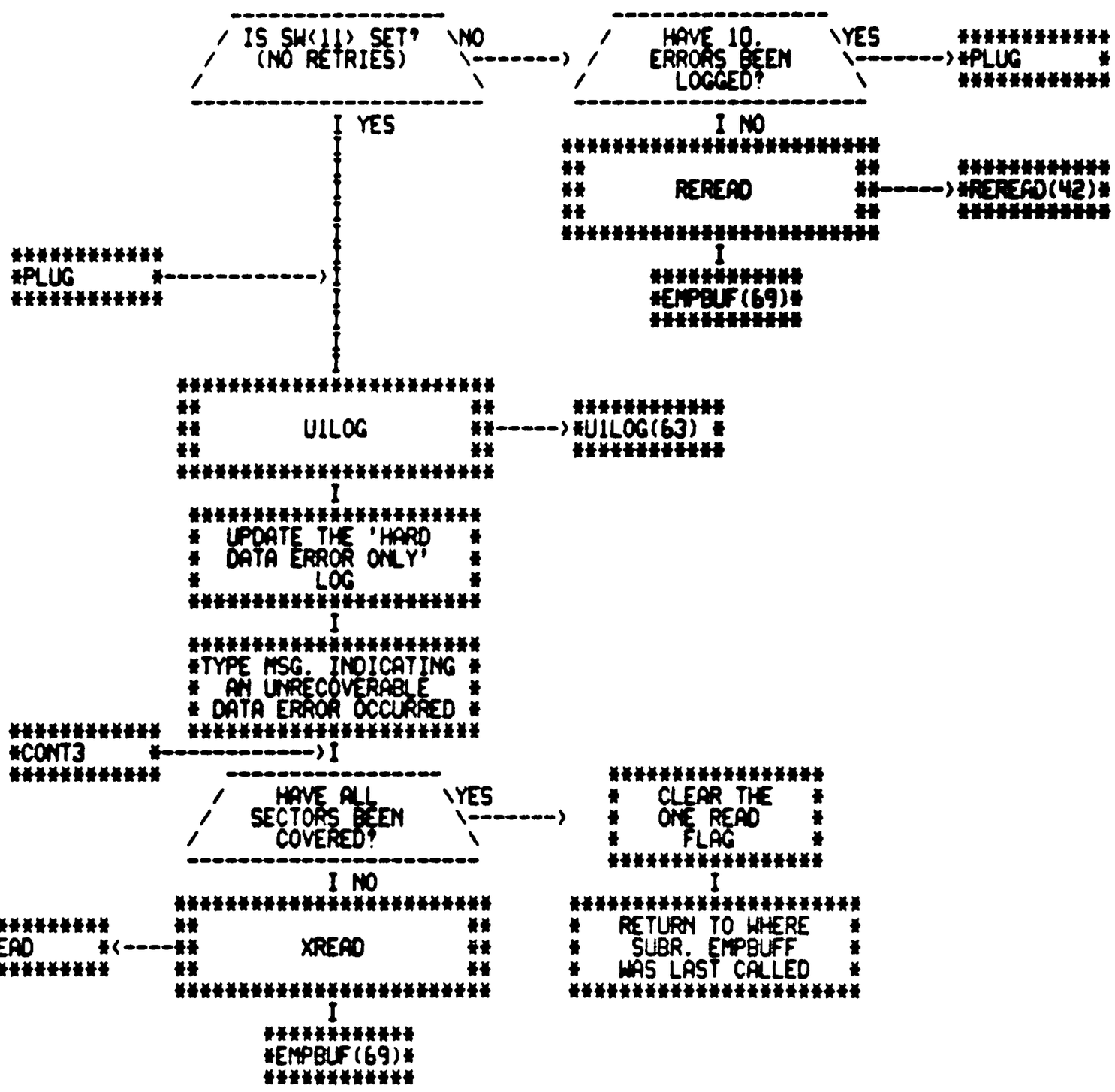

BSTAT
SAVE THE ERROR CODE
IS THERE A DEFINITE ERROR CODE?
NO
#NOPRNT
YES
ADJUST ERROR CODE TO OBTAIN TABLE EVEN ADDR.
OBTAIN THE CORRECT ERROR CODE LOG
UPDATE THE CORRECT ERROR CODE LOG
#NOPRNT
IS SW(13) SET? (NO PRINT ERRORS)
YES
#CKUNIT
NO
TYPE ERROR HEADER
TYPE PASSES COMPLETED AT TIME OF ERROR
TYPE COMMAND STATUS REGISTER FUNCTION ETC.

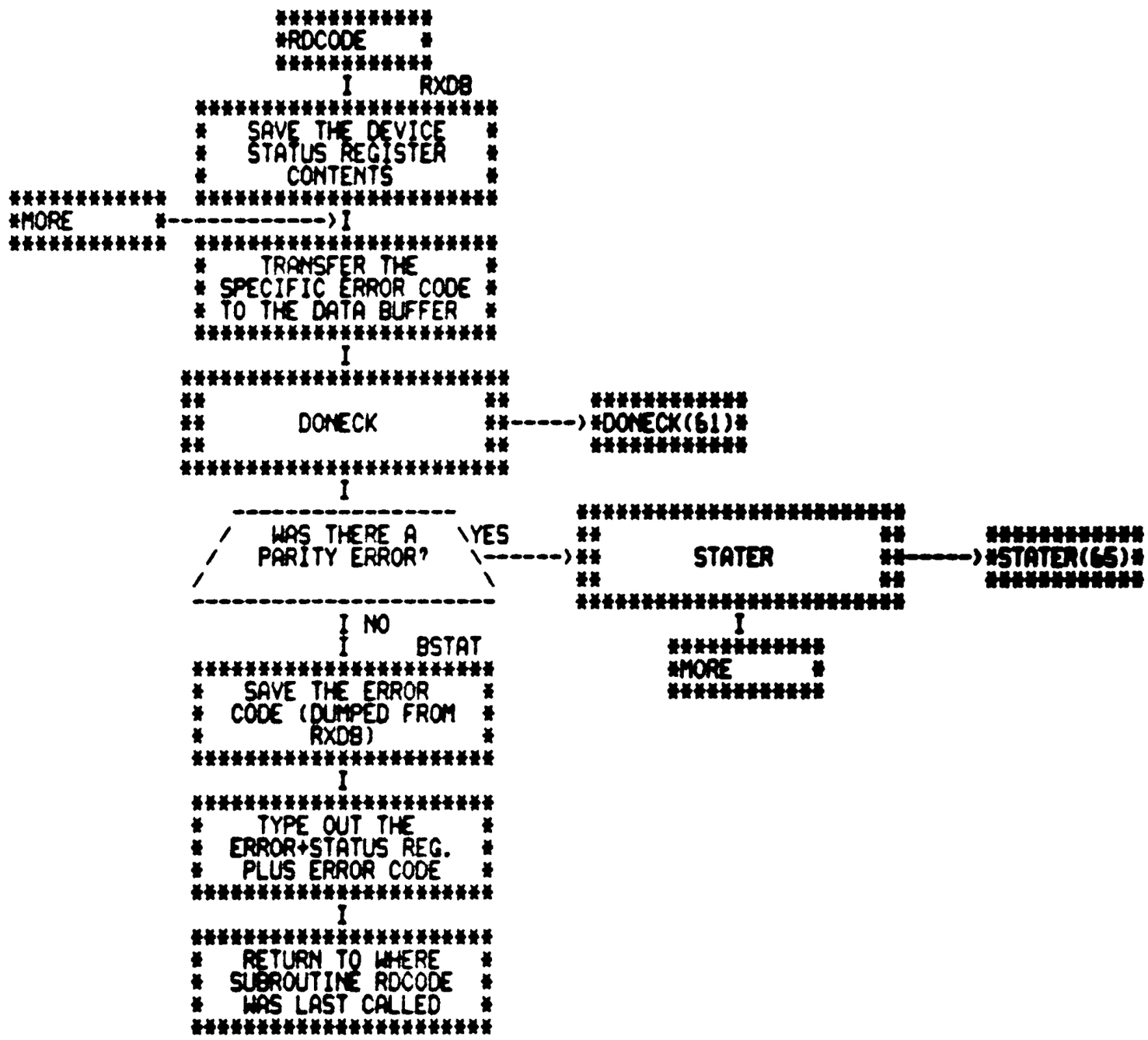

TYPE ADDRESSES AND RUN CONDITIONS
PRINT THE STATUS REGISTERS
#CKUNIT
IS DRIVE 0 PRESENTLY IN USE?
YES
SETUP DRIVE 0 READ STATUS LOG
NO
SET UP DRIVE 1 READ STATUS LOG
DONECK
#DONECK(61)
WAS THERE A PARITY ERROR
YES
STATER
#STATER(65)
NO
#CKUNIT
WAS DRIVE READY SET?
YES
RETURN TO SHOW THE ERROR
NO
#EMPER(78) #EMPTY BUFFER

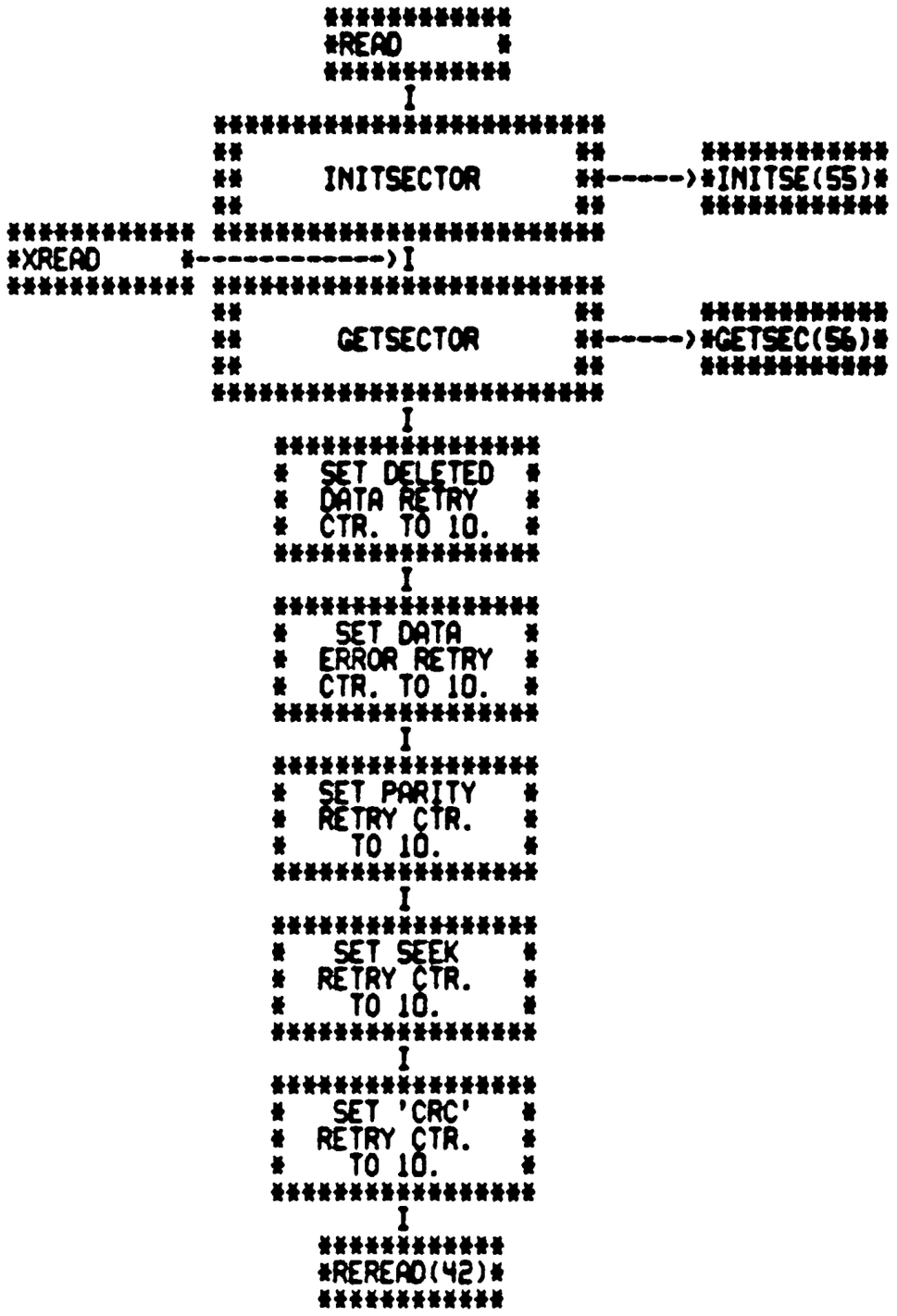

TYPE MSG. INDICATING DRIVE WASN'T READY
DELUNIT
DELUNI(30)
RETURN TO REDO THE ERROR & SHOW IT
EMPER(78) EMPTY BUFFER

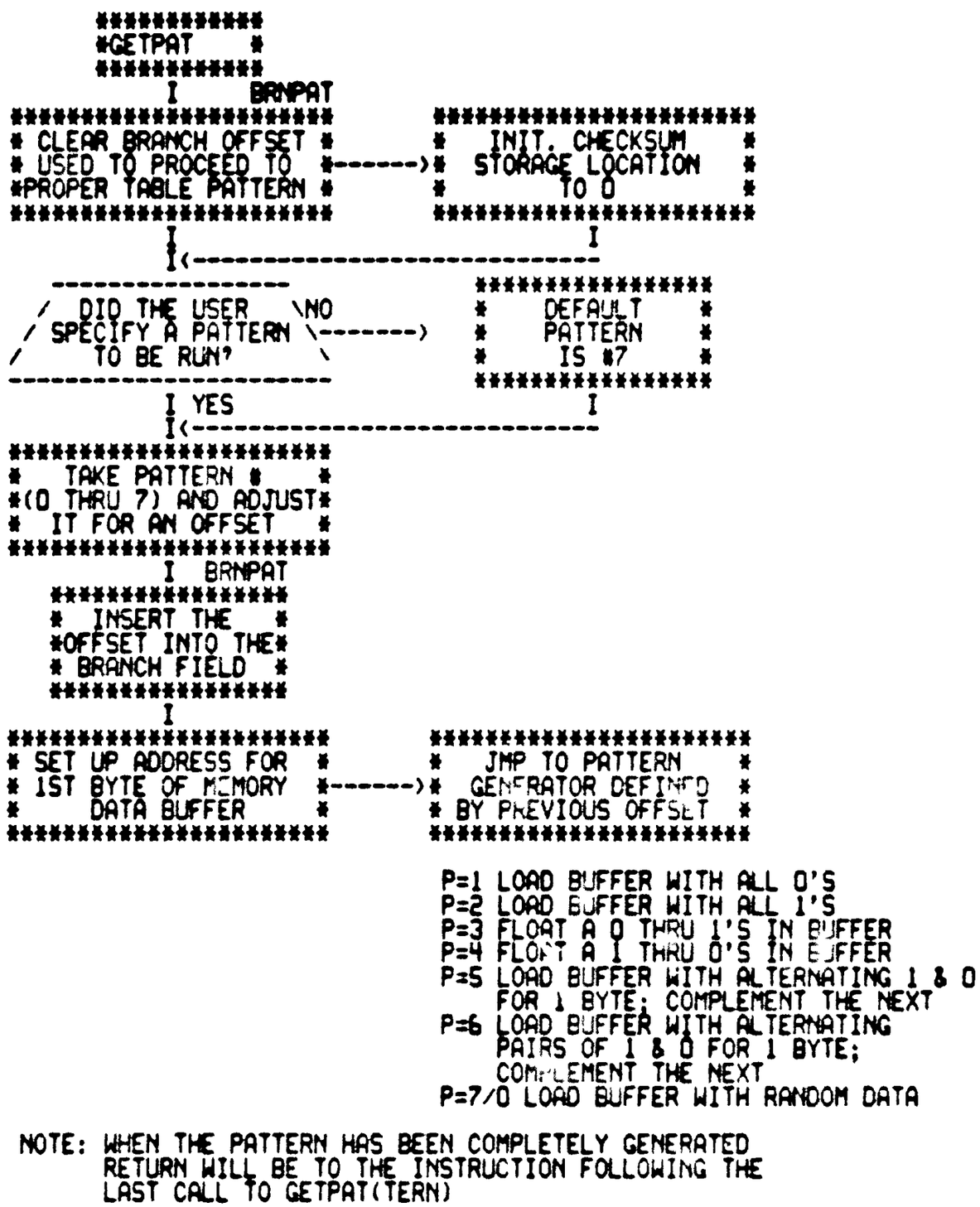

#EMPER(7%) #
CORRECT STACK FOR
PROPER RETURN FROM
'PARTST'
SET UP MESSAGE
ADDRESSES FOR
ERROR REPORTING
PCONT
SET UP A 'NOT HARD
ERROR' RETURN FROM
'PARTST' THRU 'PCONT'
#PARTES(28)#


#EMPOON(73)#
WAS THIS RDCHECK CAUSED BY A CRC ERROR
YES
#CRCER(46) #
NO
WERE THERE ANY OTHER ERRORS?
NO
#CONT3 #
YES
IS SW(13) SET? (NO PRINT ERRORS)
YES
#EMPLOG #
NO
PRINT THE TOTAL DATA ERROR COUNT
INDICATE IF THE CHECKSUM WAS GOOD OR BAD
#EMPLOG #
UILOG
#UILOG(63) #
UPDATE THE DATA ERROR LOG
TRKERR
#TRKERR(54)#

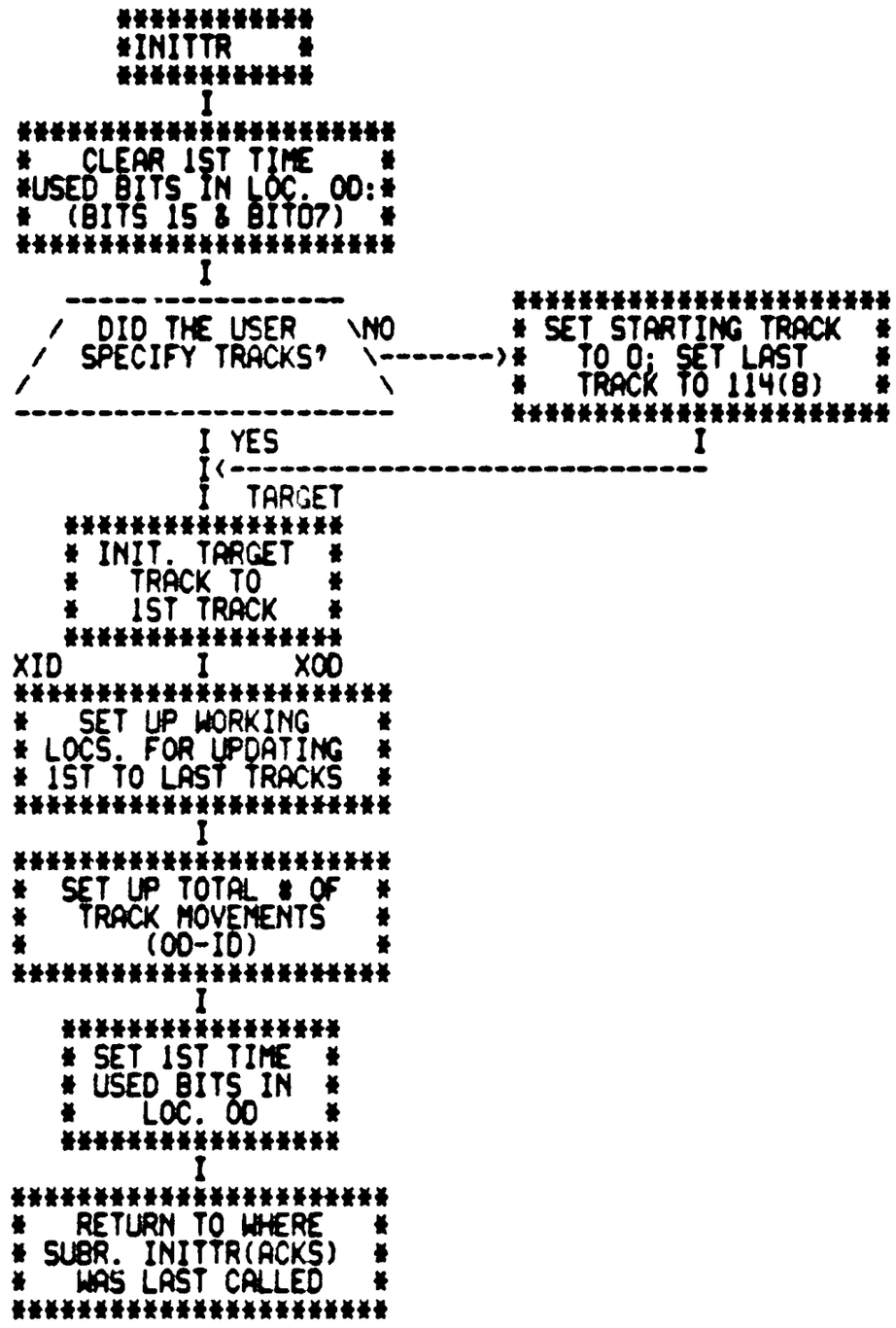

IS SW(11) SET? (NO RETRIES)
NO
YES
HAVE 10. ERRORS BEEN LOGGED?
YES
#PLUG
NO
REREAD
#REREAD(42)#
#PLUG
#EMPBUF(69)#
UILOG
#UILOG(63) #
UPDATE THE 'HARD DATA ERROR ONLY' LOG
#TYPE MSG. INDICATING AN UNRECOVERABLE DATA ERROR OCCURRED #
#CONT3
HAVE ALL SECTORS BEEN COVERED?
YES
NO
CLEAR THE ONE READ FLAG
RETURN TO WHERE SUBR. EMPBUFF WAS LAST CALLED
#XREAD
XREAD
#EMPBUF(69)#

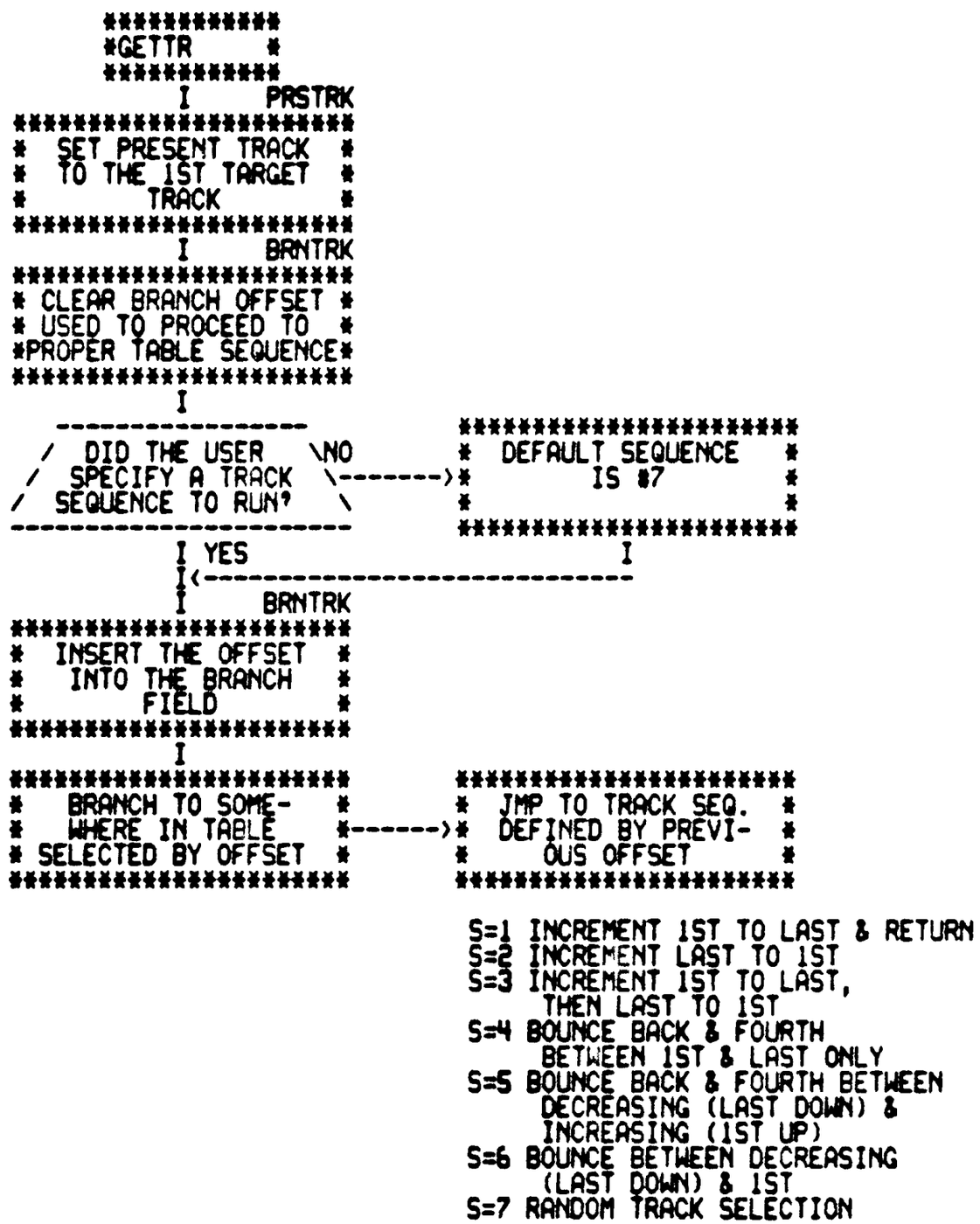

RDCODE
RXDB
SAVE THE DEVICE STATUS REGISTER CONTENTS
MORE
TRANSFER THE SPECIFIC ERROR CODE TO THE DATA BUFFER
DONECK
DONECK(61)
WAS THERE A PARITY ERROR?
YES
STATER
STATER(65)
NO
BSTAT
SAVE THE ERROR CODE (DUMPED FROM RXDB)
TYPE OUT THE ERROR+STATUS REG. PLUS ERROR CODE
RETURN TO WHERE SUBROUTINE RDCODE WAS LAST CALLED
MORE

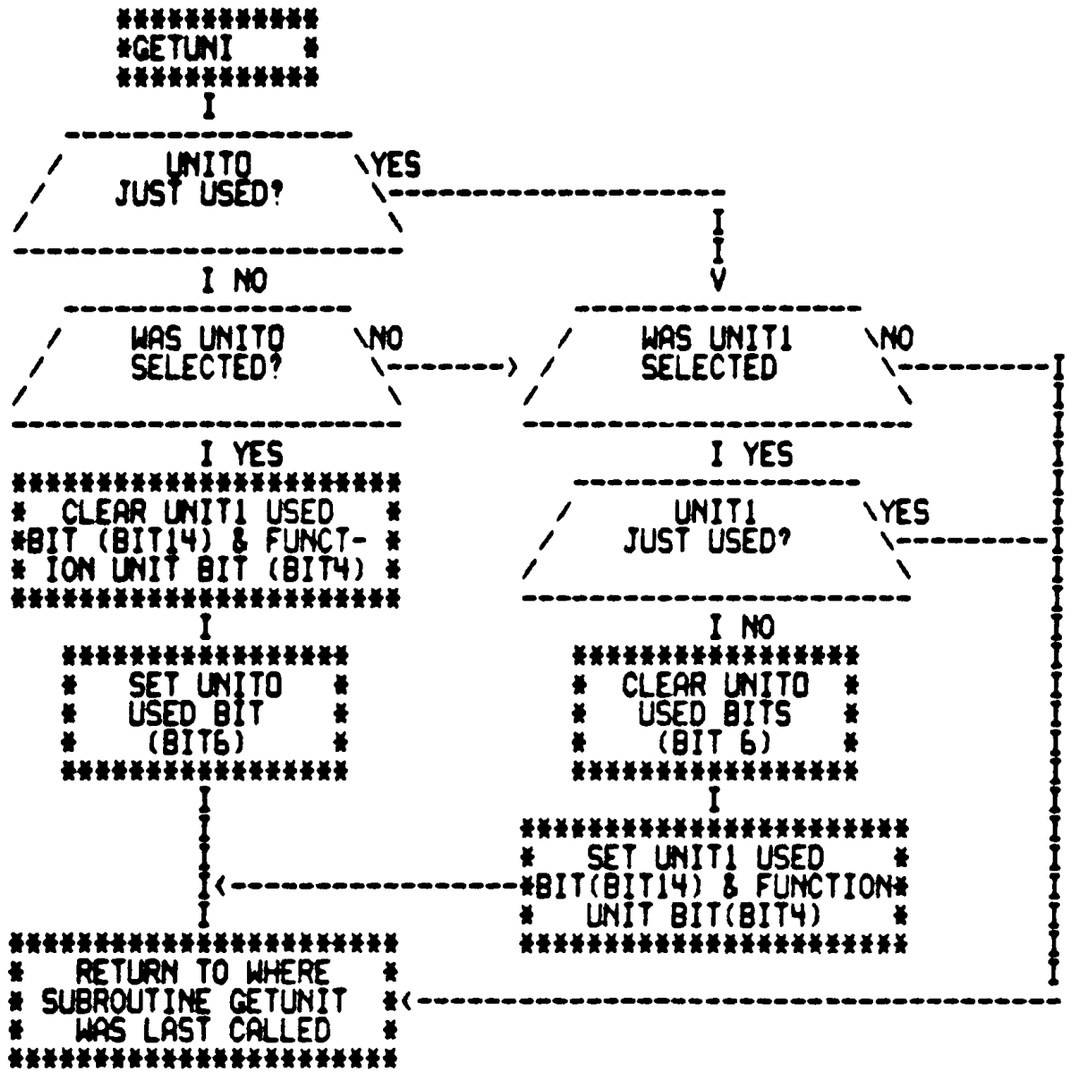

READ
INITSECTOR
INITSE(55)
XREAD
GETSECTOR
GETSEC(56)
SET DELETED DATA RETRY CTR. TO 10.
SET DATA ERROR RETRY CTR. TO 10.
SET PARITY RETRY CTR. TO 10.
SET SEEK RETRY CTR. TO 10.
SET 'CRC' RETRY CTR. TO 10.
REREAD(42)

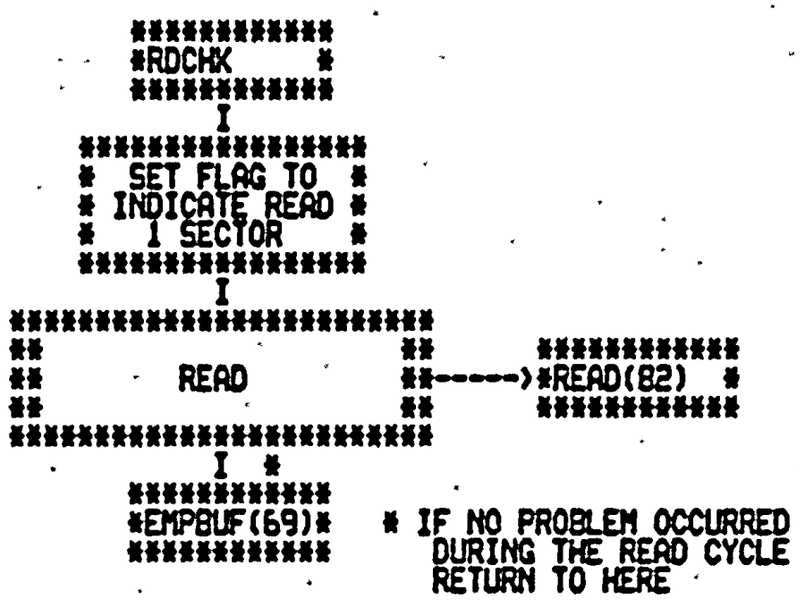


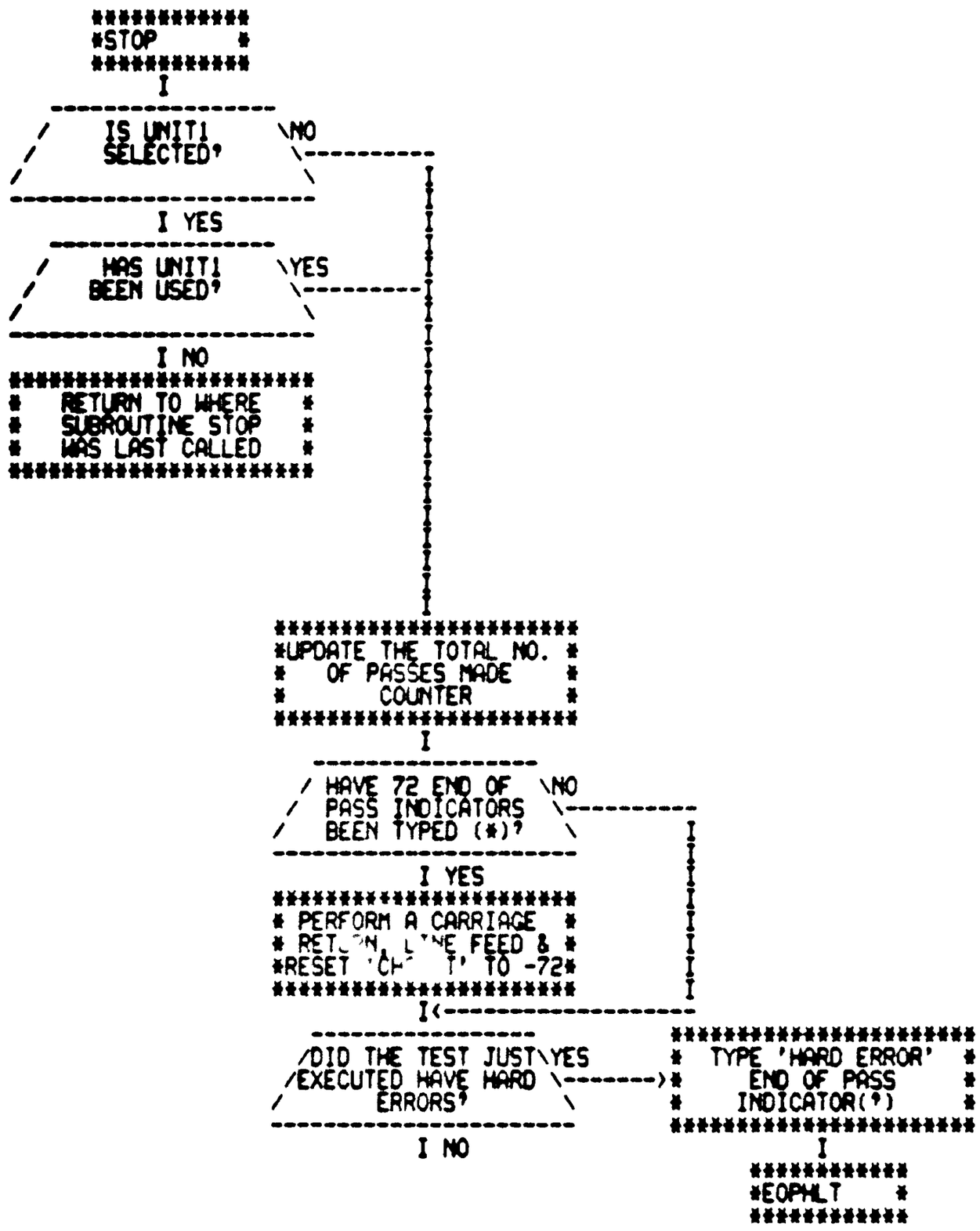
NOTE: WHEN THE PATTERN HAS BEEN COMPLETELY GENERATED
RETURN WILL BE TO THE INSTRUCTION FOLLOWING THE
LAST CALL TO GETPAT(TERN)

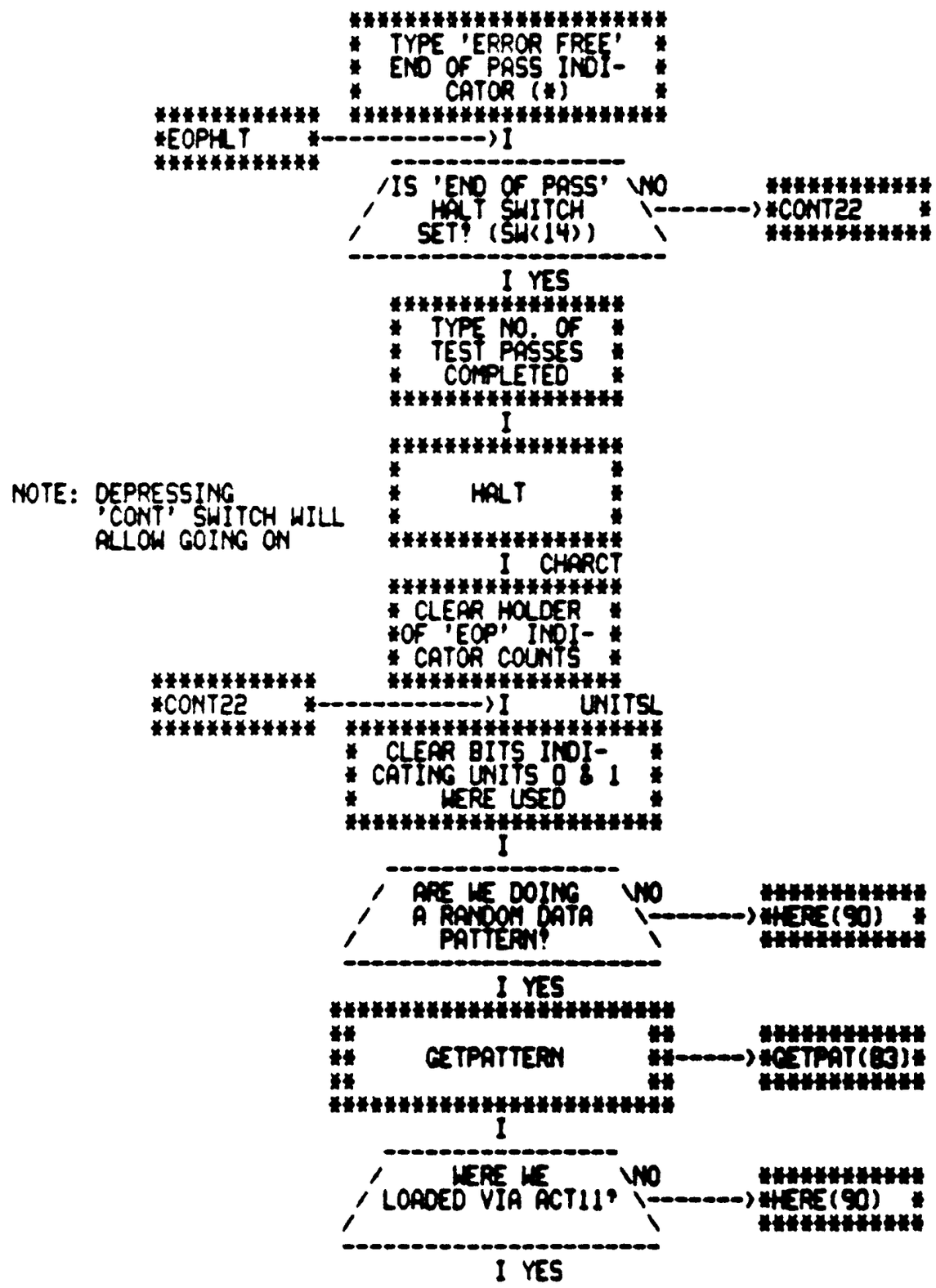

INITTR
CLEAR IST TIME
USED BITS IN LOC. OD:
(BITS 15 & BIT07)
DID THE USER SPECIFY TRACKS?
NO
SET STARTING TRACK TO 0; SET LAST TRACK TO 114(8)
YES
TARGET
INIT. TARGET TRACK TO IST TRACK
XID
XOD
SET UP WORKING LOCS. FOR UPDATING IST TO LAST TRACKS
SET UP TOTAL # OF TRACK MOVEMENTS (OD-ID)
SET IST TIME USED BITS IN LOC. OD
RETURN TO WHERE SUBR. INITTR(ACKS) WAS LAST CALLED

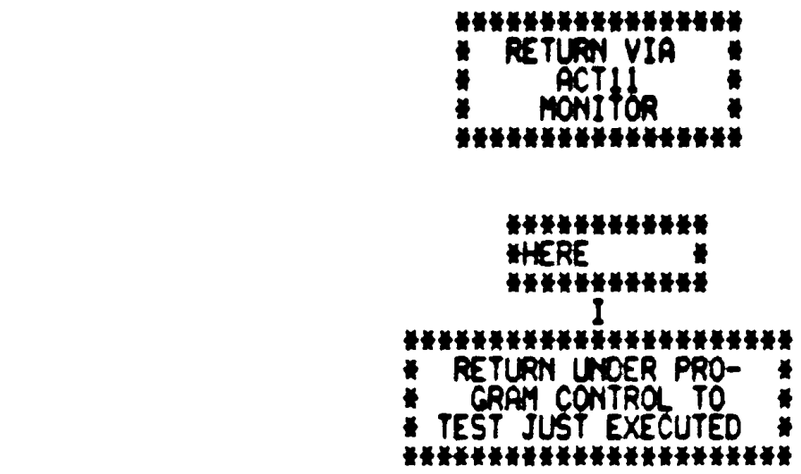

GETTR
PRSTRK
SET PRESENT TRACK TO THE 1ST TARGET TRACK
BRNTRK
CLEAR BRANCH OFFSET USED TO PROCEED TO PROPER TABLE SEQUENCE
DID THE USER SPECIFY A TRACK SEQUENCE TO RUN?
NO
DEFAULT SEQUENCE IS #7
YES
BRNTRK
INSERT THE OFFSET INTO THE BRANCH FIELD
BRANCH TO SOME-WHERE IN TABLE SELECTED BY OFFSET
JMP TO TRACK SEQ. DEFINED BY PREVI-OUS OFFSET
S=1 INCREMENT 1ST TO LAST & RETURN
S=2 INCREMENT LAST TO 1ST
S=3 INCREMENT 1ST TO LAST, THEN LAST TO 1ST
S=4 BOUNCE BACK & FOURTH BETWEEN 1ST & LAST ONLY
S=5 BOUNCE BACK & FOURTH BETWEEN DECREASING (LAST DOWN) & INCREASING (1ST UP)
S=6 BOUNCE BETWEEN DECREASING (LAST DOWN) & 1ST
S=7 RANDOM TRACK SELECTION


GETUNI
UNIT0 JUST USED?
YES
NO
WAS UNIT0 SELECTED?
NO
YES
CLEAR UNIT1 USED BIT (BIT14) & FUNCTION UNIT BIT (BIT4)
SET UNIT0 USED BIT (BIT6)
WAS UNIT1 SELECTED
NO
YES
UNIT1 JUST USED?
NO
CLEAR UNIT0 USED BITS (BIT 6)
SET UNIT1 USED BIT(BIT14) & FUNCTION UNIT BIT(BIT4)
RETURN TO WHERE SUBROUTINE GETUNIT WAS LAST CALLED


RDCHK
SET FLAG TO INDICATE READ 1 SECTOR
READ
READ(82)
EMPBUF(69)
IF NO PROBLEM OCCURRED DURING THE READ CYCLE RETURN TO HERE


*STOP *
IS UNIT1 SELECTED?
NO
I YES
HAS UNIT1 BEEN USED?
YES
I NO
RETURN TO WHERE SUBROUTINE STOP WAS LAST CALLED
UPDATE THE TOTAL NO. OF PASSES MADE COUNTER
HAVE 72 END OF PASS INDICATORS BEEN TYPED (*)?
NO
I YES
PERFORM A CARRIAGE RETURN, LINE FEED & RESET 'CHAR T' TO -72
DID THE TEST JUST EXECUTED HAVE HARD ERRORS?
YES
I NO
TYPE 'HARD ERROR' END OF PASS INDICATOR(?)
EOPHLT


TYPE 'ERROR FREE' END OF PASS INDI- CATOR (*)
*EOPHLT
IS 'END OF PASS' HALT SWITCH SET? (SW(14))
NO
*CONT22
I YES
TYPE NO. OF TEST PASSES COMPLETED
HALT
NOTE: DEPRESSING 'CONT' SWITCH WILL ALLOW GOING ON
I CHARCT
CLEAR HOLDER OF 'EOP' INDI- CATOR COUNTS
*CONT22
I UNITSL
CLEAR BITS INDI- CATING UNITS 0 & 1 WERE USED
ARE WE DOING A RANDOM DATA PATTERN?
NO
*HERE(90)
I YES
GETPATTERN
GETPAT(83)
WERE WE LOADED VIA ACT11?
NO
*HERE(90)
I YES


RETURN VIA
ACT11
MONITOR
HERE
RETURN UNDER PRO-
GRAM CONTROL TO
TEST JUST EXECUTED

DZRXAD.P11 TABLE OF CONTENTS

```
                                        .NLIST CND,MD,MC
                                        .LIST ME
                                        .ENABL ABS,AMA
                                        .MCALL .HEADER,.EQUAT,.$TYPE,.$TYPOCT
                                        .MCALL .$POWER,STARS,.$DB2D,.$SB2D,.$SAVE,.STRAP


                        .TITLE  MAINDEC-11-DZRXA-D
                        ;*COPYRIGHT (C) DEC. 21,1975
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY DAVID L ADAMS
                        ;*
                        ;*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 A LICENSE FOR USE ONLY
                                ;ON A SINGLE COMPUTER SYSTEM AND MAY BE COPIED ONLY WITH
                                ;THE INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS
                                ;SOFTWARE, OR ANY OTHER COPIES THEREOF, MAY NOT BE PROVIDED
                                ;OR OTHERWISE MADE AVAILABLE TO ANY OTHER PERSON
                                ;EXCEPT FOR USE ON SUCH SYSTEM AND TO ONE WHO AGREES TO
                                ;THESE LICENSE TERMS. TITLE TO 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.


                        .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             ;;PRIORITY LEVEL 6
000340                          PR7=    340             ;;PRIORITY LEVEL 7

                                ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                          SW15=   100000
040000                          SW14=   40000
020000                          SW13=   20000
010000                          SW12=   10000
004000                          SW11=   4000
002000                          SW10=   2000
001000                          SW09=   1000
000400                          SW08=   400
000200                          SW07=   200
000100                          SW06=   100
000040                          SW05=   40
000020                          SW04=   20
000010                          SW03=   10
000004                          SW02=   4
000002                          SW01=   2
000001                          SW00=   1
                                .EQUIV  SW09,SW9
                                .EQUIV  SW08,SW8
                                .EQUIV  SW07,SW7
                                .EQUIV  SW06,SW6
                                .EQUIV  SW05,SW5
                                .EQUIV  SW04,SW4
                                .EQUIV  SW03,SW3
                                .EQUIV  SW02,SW2
                                .EQUIV  SW01,SW1
                                .EQUIV  SW00,SW0

                                ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                          BIT15=  100000
040000                          BIT14=  40000
```

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

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


                                ;SPECIAL EQUATES


                000013          RDOSTAT =13
                000033          RD1STAT =33
                000017          RDER    =17
                000040          DONEBIT =40
                000101          FBIE    =101
                000103          EBIE    =103
                000105          WRTIE   =105
                000107          RDIE    =107
                000115          WTDOIE  =115
                040001          RECAL   =40001
                000200          PR4     =200
```

```
                    000340                          PR7     =340
                    000000                          OPEN    =0

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

                    000024                          .=24
000024  012734                                      $PWRDN
000026  000340                                      340

                    000034                          .=34
000034  012676                                      $TRAP                   ;ADDRESS OF TRAP SERVICE
000036  000340                                      340

                    000046                          .=46
000046  010674                                      LOGICAL                 ;ACT11 EOP HOOK

                    000052                          .=52
000052  000000                                      .WORD 0

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


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

                                        ;STARTING ADDRESSES
                                        ;
                                        ;INITIAL START  =200    /CLEARS ALL ERROR LOGS,RESETS COUNTERS,AND ALLOWS FOR
                                        ;                       /SELECTION OF DRIVES AND TEST CONDITIONS
                                        ;
                                        ;RESTART        =202    /USES PREVIOUS INITIAL START DRIVES AND TEST
                                        ;                       /SELECTION AND CONTINUES TO ACCUMULATE STATISTICAL DATA
                                        ;
                                        ;ERROR REPORT   =204    /PRINTS OUT ALL RUN AND ERROR CONDITIONS
                                        ;                       /ACUMULATED OVER THE RUN OF THE PROGRAM.
                                        ;

                    000200                          .=200
000200  000402                                      BR 1$
000202  000403                                      BR 2$
000204  000404                                      BR 3$
000206  000137  001220          1$:                 JMP SA200               ;OPERATOR SELECTED CONDITIONS
000212  000137  002532          2$:                 JMP RESTART             ;RESTART PROGRAM WITH PREVIOUS CONDITIONS
000216  000137  017622          3$:                 JMP ERDUMP              ;STATISTICAL ERROR PRINT OUT


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

                                        ;THE FOLLOWING LOCATIONS "OD","ID","FIRST",AND "LAST"
                                        ;MAY BE CHANGED BY THE OPERATOR MANUALLY HOWEVER FOLLOWING THESE RESTRICTIONS.
                                        ;(IF THESE LOCATIONS ARE LEFT CLEARED,MAX AND MIN VALUES ARE ASSUMED)
                                        ;
```

```
                                        ;       1. DEFINITATIONS:
                                        ;               OD=ADDRESS OF TRACK AT OUTTER DIAMETER (MIN VALUE=0)
                                        ;               ID=ADDRESS OF TRACK AT INNER DIAMETER (MAX VALUE=114 [OCTAL])
                                        ;               FIRST=ADDRESS OF FIRST SECTOR OF TRACK (MIN VALUE=1)
                                        ;               LAST=ADDRESS OF LAST SECTOR OF TRACK (MAX VALUE=32 [OCTAL])
                                        ;
                                        ;       2. LOCATIONS:
                                        ;               TRACKS  LOC. 1200 BITS 14-----8 6-----0
                                        ;                                              ID      OD
                                        ;
                                        ;               SECTORS LOC. 1202 BITS 12-----8 4-----0
                                        ;                                             LAST    FIRST
                                        ;
                                        ;       3. RESTRICTIONS:
                                        ;               THE CONTENTS OF "OD" MUST BE <= THE CONTENTS OF "ID"
                                        ;               THE CONTENTS OF FIRST MUST BE <= THE CONTENTS OF "LAST"

                001200                          .=1200
001200  000000                  OD:            0
                001201          ID=OD+1
001202  000000                  FIRST:         0
                001203          LAST=FIRST+1

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

                                        ;THE NEXT WORD IS THE LOCATION OF THE "INTERRUPT VECTOR" ADDRESS
                                        ;IF THE HARDWARE IS JUMPERED FOR OTHER THAN STANDARD (264) VECTOR
                                        ;ADDRESS, THEN LOAD INTO THIS LOCATION THE NEW VECTOR ADDRESS.
                                        ;
                                        ;IF THIS ADDRESS IS INCORRECT (DOES NOT MATCH THE HARDWARE) THE
                                        ;PROGRAM WILL HALT AT THE VECTOR ADDRESS THE HARDWARE IS JUMPERED
                                        ;TO, AS ALL OTHER VECTOR ADDRESSES WILL HAVE HALTS IN THEM.
                                        ;
                                        ;*****************************NOTE**********************************
                                        ;
                                        ;THE INTERRUPT VECTOR ADDRESS CAN NOT BE SET FOR ADDRESSES 200
                                        ;THRU 243, AS THESE ADDRESS ARE USED BY THE PROGRAM
                                        ;
                                        ;***************************************************************

001204  000264                  INTVEC: 264

                                        ;***************************************************************
                                        ;THE FOLLOWING TWO WORDS CONTAIN THE DEVICE CODES FOR
                                        ;THE RX11 INTERFACE REGISTERS
                                        ;       RXCS = COMMAND STATUS REGISTER
                                        ;       RXDB = DATA BUFFER REGISTER (USED AT VARIOUS TIMES FOR:)
                                        ;               (RXTA = TRACK ADDRESS)
                                        ;               (RXSA = SECTOR ADDRESS)
                                        ;               (RXES = ERROR STATUS)
```

```
                        ;IF THE RX11 SYSTEM UNDER TEST IS JUMPERED FOR REGISTER
                        ;ADDRESSES OTHER THAN STANDARD: RXCS = 177170
                        ;                               RXDB = 177172
                        ;PLACE IN THESE TWO WORDS THE CORRESPONDING NEW ADDRESSES FOR
                        ;THE REGISTERS.  THE PROGRAM WILL TYPE OUT THE REGISTER
                        ;ADDRESSES FOR VERIFICATION BY THE OPERATOR.  IF THE ADDRESSES
                        ;ARE INCORRECT (I.E. - DON'T MATCH THE HARDWARE) AND THE PROGRAM
                        ;IS RUN WITH THE INCORRECT ADDRESSES, A 'HUNG' CONDITION WILL BE
                        ;REIPORTED AND THE PROGRAM WILL 'HALT' AS THE RX11 WILL NOT BE
                        ;ABLE TO RESPOND TO COMMANDS.
                      ;*****************************************************************

001206  177170        RXCS:       177170
001210  177172        RXDB:       177172

                        ;BIT ASSIGNMENT IN THE RXCS REGISTER.
                        ;
                        ;KEY: R - READ ONLY BIT
                        ;     W - WRITE ONLY BIT
                        ;
                        ;       15 - R - ERROR FLAG
                        ;       14 - W - INITIALIZE (RECALIBRATE)
                        ;       13 -
                        ;       10      NOT USED
                        ;        8 -
                        ;        7 - R - TRANSFER REQUEST (TR) FLAG
                        ;        6 - R/W - INTERUPT ENABLE
                        ;        5 - R - DONE FLAG
                        ;        4 - W - UNIT SELECT
                        ;        3 - W - FUNCTION
                        ;        2 - W - FUNCTION
                        ;        1 - W - FUNCTION
                        ;        0 - W - GO
                        ;
                        ;FUNCTION CODES:
                        ;
                        ;       0 + GO = FILL BUFFER
                        ;       2 + GO = EMPTY BUFFER
                        ;       4 + GO = WRITE SECTOR
                        ;       6 + GO = READ SECTOR
                        ;       10 (NOT USED)
                        ;       12 + GO = READ STATUS "A"
                        ;       14 + GO = WRITE DELETED DATA
                        ;       16 + GO = READ STATUS "B" (CODES)
                        ;
                        ;
                        ;THE FOLLOWING BIT ASSIGNMENTS REPRESENTES THE STATUS AT THE END OF A
                        ;FUNCTION (EXCEPT FUNCTION "READ STATUS B") DISPLAYED IN THE
                        ;RX DATA BUFFER (RXDB).
                        ;
                        ;       15 -
                        ;       10     NOT USED
                        ;        8 -
                        ;        7 - SELECTED DRIVE READY *
```

```
                                        ;       6 - DELETED DATA
                                        ;       5 -
                                        ;       4 -
                                        ;       3 - WRITE PROTECT ERROR (WHEN AVAILABLE)
                                        ;       2 - INITIALIZE DONE **
                                        ;       1 - PARITY ERROR
                                        ;       0 - CRC ERROR
                                        ;
                                        ;*  VISIBLE ONLY IF THE FUNCTION WAS A #12 "READ STATUS A"
                                        ;
                                        ;**  VISIBLE ONLY AFTER AN INITIALIZE [KEY] OR [PROGRAMMED] WAS ISSUED.

001212  000000          DTESTP:         .WORD 0                 ;TEST SELECTION WORD
001214  177570          SWR:            .WORD DSWR              ;ADDRESS OF SWITCH REGISTER
001216  177570          DISPLAY:        .WORD DDISP             ; ADDRESS OF DISPLAY REGISTER

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

                                ;START OF OPERATOR SELECTABLE TEST,DATA PATTERNS,AND DRIVE UNIT CONDITIONS
                                ;SET THE TEST CONDITIONS WANTED (OR LEAVE 0) IN LOCATION CALLED
                                ;"DTESTP" (LOC. 1212), BEFORE STARTING THE PROGRAM. WHEN THE PROGRAM
                                ;STARTS IT WILL TYPE OUT THE TEST CONDITIONS IT IS OPERATING UNDER.
                                ;

                                ;SWITCH REGISTER BITS
                                ;
                                ; 15  14  13  12  11  10  9  8  7  6  5  4  3  2  1  0
                                ; U1  U0      NOT USED    D  P  P  P  T  T  T  S  S  S
                                ;
                                ; U1    SELECT DRIVE UNIT 1
                                ; U0    SELECT DRIVE UNIT 0
                                ;IF NEITHER DRIVE IS SPECIFIED,PROGRAM WILL SELECT ALL DRIVES
                                ;THAT ARE READY
                                ;
                                ; D =     DELETED DATA FUNCTIONS
                                ;
                                ;IF THIS SWITCH IS ON ALL READ AND WRITE FUNCTIONS WILL BE IN THE
                                ;DELETED DATA MODE
                                ;
                                ;
                                ; P =     DATA PATTERN SELECTION
                                ;       0       DO PATTERN 7 (RANDOM)
                                ;       1       ZEROS
                                ;       2       ONES
                                ;       3       FLOATING ZERO
                                ;       4       FLOATING ONE
                                ;       5       125
                                ;       6       314
                                ;       7       RANDOM
                                ;
                                ; T =     FUNCTIONAL TESTS
                                ;       0       DO TEST 7 (WRITE/READ/READ CHECK)
                                ;       1       WRITE ONLY
                                ;       2       WRITE/READ
```

```
                        ;       3       WRITE/READ CHECK
                        ;       4       READ CHECK ONLY
                        ;       5       READ ONLY
                        ;       6       WRITE/READ CHECK ON ALTERNATE DRIVES
                        ;       7       WRITE/READ/READ CHECK *
                        ;
                        ; * NOTE: THE FIRST HALF OF TEST 7 FORCES THE TRACK SEQUENCE
                        ;TO INCRIMENT UP THROUGH ALL THE TRACKS DOING WRITE / READ CHECK FUNCTIONS.
                        ;THIS VERIFIES THAT ALL THE TRACKS ARE ACCESSABLE.
                        ;THE SECOND HALF OF THE PASS WILL USE THE SEQUENCE SELECTED BY THE
                        ;OPERATOR AS INDICATED BELOW,AND ONLY READ CHECK THE DATA JUST WRITTEN.
                        ;THIS VERIFIES THAT THE DATA CAN BE READ FROM A TRACK,AFTER THE HEAD HAS BEEN
                        ;MOVED AWAY FROM AND BACK TO THAT TRACK. AT THE COMPLETION OF THE PASS
                        ;THE DELETED DATA BIT IN TEST CONDITIONS IS COMPLEMENTED AND THE NEXT
                        ;PASS WILL BE RUN UNDER THIS NEW CONDITION.
                        ;
                        ; S =    TRACK SEQUENCING
                        ;       0       DO SEQUENCE 7 (RANDOM)
                        ;       1       INCREMENT
                        ;       2       DECREMENT
                        ;       3       INCREMENT/DECREMENT
                        ;       4       BOUNCE
                        ;       5       DECREASING BOUNCE
                        ;       6       STROBE
                        ;       7       RANDOM


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

                                ;SET THE OPERATING CONDITIONS IN THE SWITCH REGISTER (HARDWARE).
                                ;OR THE SOFTWARE SWITCH REGISTER (LOC. 176) BEFORE STARTING THE PROGRAM.
                                ;
                                ;SWITCHES DO THE FOLLOWING WHEN SET TO "1".
                                ;
                                ;       SW 15   =HALT ON ERROR
                                ;       SW 14   =HALT AT END OF TEST
                                ;       SW 13   =DON'T PRINT ERROR MESSAGE
                                ;       SW 12   =TYPE ONLY 10 DATA ERRORS
                                ;       SW 11   =NO RETRY ON ERROR. LOG HARD ERROR
                                ;
                                ;       SW  8   =NO RECALIBRATION ON SEEK ERRORS
                                ;
                                ;       SW15 - SW0      =SELECT SOFTWARE SWITCH REGISTER

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


001220  012706  001200          SA200:          MOV #STACK,SP
001224  012746  000340                          MOV #PR7,-(SP)
001230  012746  001236                          MOV #2$,-(SP)
001234  000002                                  RTI
001236  012737  001256  000004  2$:             MOV #3$,4               ;SET TIME OUT VECTOR
```

```
001244  022777  177777  177742          CMP #177777,@SWR        ;IS SOFTWARE SWR SELECTED
001252  001402                          BEQ 4$                  ;YES, INSERT IT'S ADDRESS
001254  000407                          BR 5$                   ;BRANCH IF NO TIMEOUT TRAP OCCURS
001256  022626                  3$:     CMP (SP)+,(SP)+         ;RESTORE STACK AFTER TRAP
001260  012737  000176  001214  4$:     MOV #SWREG,SWR          ;POINT TO SOFTWARE SWITCH REG.
001266  012737  000174  001216          MOV #DISPREG,DISPLAY    ;POINT TO SOFTWARE DISP. REG.
001274  012737  000006  000004  5$:     MOV #6,4                ;RESTORE ERROR VECTOR
001302  013703  001204                  MOV INTVEC,R3           ;SET INTERUPT VECTOR LOCATIONS
001306  012723  007036                  MOV #INTSERV,(R3)+
001312  012713  000340                  MOV #340,(R3)
001316  012737  001234  011754          MOV #001234,RAN1        ;INITIALIZE CONSTANTS IN RANDOM NUMBER GENERATOR
001324  012737  000765  011756          MOV #000765,RAN2
001332  012705  015564                  MOV #FLAG,R5            ;SET UP TO CLEAR ALL COUNTERS
001336  005025                  1$:     CLR (R5)+               ;OF ERRORS,ACCESSES,AND PASSES
001340  022705  017604                  CMP #PASCNTR+2,R5       ;HAVE ALL COUNTERS BEEN CLEARED
001344  001374                          BNE 1$
001346  012705  015544          START1: MOV #RDRETRY,R5         ;SET ALL RETRY COUNTERS TO 10
001352  012725  000012          3$:     MOV #10.,(R5)+
001356  022705  015564                  CMP #SRETRY+2,R5
001362  001373                          BNE 3$
001364  104400  015315                  TYPE ,MREV              ;TYPE NAME AND REVISION OF PROGRAM RUNNING
001370  104400  014121                  TYPE ,MRXCS             ;TYPE OUT DEVICE REGISTER ADDRESSES
001374  013746  001206          MOV     RXCS,-(SP)      ;;SAVE RXCS FOR TYPEOUT
001400  104402                  TYPOS                   ;;GO TYPE--OCTAL ASCII
001402     006                  .BYTE   6               ;;TYPE 6 DIGITS
001403     000                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
001404  104400  014131                  TYPE ,MRXDB
001410  013746  001210          MOV     RXDB,-(SP)      ;;SAVE RXDB FOR TYPEOUT
001414  104402                  TYPOS                   ;;GO TYPE--OCTAL ASCII
001416     006                  .BYTE   6               ;;TYPE 6 DIGITS
001417     000                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
001420  104400  014102                  TYPE ,MINTVEC           ;AND VECTOR ADDRESS
001424  013746  001204          MOV     INTVEC,-(SP)    ;;SAVE INTVEC FOR TYPEOUT
001430  104402                  TYPOS                   ;;GO TYPE--OCTAL ASCII
001432     003                  .BYTE   3               ;;TYPE 3 DIGIT(S)
001433     000                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
001434  004737  006714                  JSR PC,DONECK           ;WAIT FOR DONE BIT AFTER RECAL
001440  032777  000004  177542          BIT #BIT2,@RXDB         ;IS INIT DONE SET
001446  001002                          BNE 1$                  ;YES SKIP NEXT INSTRUCTION
001450  104400  014467                  TYPE ,MINIT1            ;NO,PRINT NO INIT DONE ERROR
001454  005777  177526          1$:     TST @RXCS               ;WAS THERE AN ERROR ON START RECAL
001460  100002                          BPL TESTDR              ;IF NOT CONTINUE
001462  004737  002404                  JSR PC,HOME             ;YES,DO RECAL AGAIN
001466  012737  000012  015554  TESTDR: MOV #10.,P2RETRY        ;SET THE PARITY RETRY COUNTER
001474  005037  010464                  CLR UNITSEL             ;CLEAR UNIT SELECTION WORD

                                ;TEST ACT11 LOAD MEDIA INDICATOR

001500  123727  000041  000010          CMPB @#41,#10           ;DOES LOCATION 41 CONTAIN THE NUMBER 10?
001506  001003                          BNE     NOACT           ;BRANCH IF NOT
001510  104400  013575                  TYPE,   DOLOAD          ;INFORM USER TO REMOVE LOAD MEDIUM
                                                                ;FROM UNIT 0 AND REPLACE WITH
                                                                ;A 'SCRATCH' DISKETTE IF HE
                                                                ;WISHES TO TEST UNIT 0
```

```
001514  000000                    HLT1:    HALT                    ;WAIT FOR USER RESPONSE
001516  105777  177466            NOACT:   TSTB @RXDB              ;IS DRY BIT SET FOR UNIT 0 AFTER RECAL
001522  100003                             BPL 2$                  ;IF NOT SKIP NEXT INSTRUCTION
001524  052737  000200  010464             BIS #BIT7,UNITSEL       ;YES,SET UNIT 0 READY BIT
001532  032737  140000  001212    2$:      BIT #140000,DTESTP      ;WERE ANY DRIVES SPECIFIED
001540  001445                             BEQ NOSEL               ;NO,SEE IF UNIT 1 IS READY
001542  005737  001212                     TST DTESTP              ;WAS UNIT 1 SELECTED
001546  100044                             BPL OPCOND              ;NO,IT MUST BE UNIT 0
001550  004737  001572                     JSR PC,DRY1             ;SEE IF UNIT 1 IS READY
001554  032737  040000  001212             BIT #40000,DTESTP       ;WAS UNIT 0 SELECTED
001562  001036                             BNE OPCOND              ;YES,CONTINUE
001564  105037  010464                     CLRB UNITSEL            ;NO,CLEAR UNIT 0 READY BIT
001570  000433                             BR OPCOND

001572  012777  000033  177406    DRY1:    MOV #RD1STAT,@RXCS      ;READ STATUS OF UNIT 1
001600  004737  006714                     JSR PC,DONECK           ;WAIT FOR DONE BIT
001604  132777  000002  177376             BITB #2,@RXDB           ;IS PARITY ERROR BIT SET
001612  001411                             BEQ 2$                  ;NO,CONTINUE
001614  004737  001760                     JSR PC,STATER           ;YES,GO HANDLE PARITY ERROR
001620  000764                             BR DRY1                 ;REISSUE FUNCTION
001622  032777  000004  177360             BIT #BIT2,@RXDB         ;INIT DONE SHOULD NOT BE SET
001630  001402                             BEQ 2$
001632  104400  014521                     TYPE ,MINIT2            ;INIT DONE WAS SET PRINT ERROR
001636  105777  177346            2$:      TSTB @RXDB              ;IS UNIT 1 DRY BIT SET
001642  100003                             BPL 3$                  ;IF NOT SKIP NEXT INSTRUCTION
001644  052737  100000  010464             BIS #BIT15,UNITSEL      ;YES,SET UNIT 1 READY BIT
001652  000207                    3$:      RTS PC

001654  004737  001572            NOSEL:   JSR PC,DRY1             ;TEST UNIT 1 FOR DRY SET
001660  013737  001212  011134    OPCOND:  MOV DTESTP,SEQUEN       ;PUT INITIAL CONDITIONS IN SEQUENCE
001666  042737  177770  011134             BIC #177770,SEQUEN      ;CLEAR ALL BUT SEQUENCE BITS
001674  013737  001212  002052             MOV DTESTP,TEST         ;PUT INITIAL CONDITIONS IN TEST
001702  042737  177707  002052             BIC #177707,TEST        ;CLEAR ALL BUT TEST BITS
001710  006037  002052                     ROR TEST                ;RIGHT JUSTIFY TEST BITS
001714  006037  002052                     ROR TEST
001720  006037  002052                     ROR TEST
001724  013737  001212  010200             MOV DTESTP,PAT          ;PUT INITIAL CONDITIONS IN PATTERN
001732  006137  010200                     ROL PAT                 ;RIGHT JUSTIFY PATTERN BITS
001736  006137  010200                     ROL PAT
001742  000337  010200                     SWAB PAT
001746  042737  177770  010200             BIC #177770,PAT         ;CLEAR ALL BUT PATTERN BITS
001754  000137  002554                     JMP XSTART

001760  005237  015564            STATER:  INC PARLOG              ;INC PARITY ERROR COUNTER
001764  104400  014121                     TYPE ,MRXCS             ;PRINT THE RXCS CONTENTS
001770  017746  177212                MOV      @RXCS,-(SP)      ;;SAVE @RXCS FOR TYPEOUT
001774  104402                        TYPOS                     ;;GO TYPE--OCTAL ASCII
001776     003                        .BYTE    3                ;;TYPE 3 DIGIT(S)
001777     000                        .BYTE    0                ;;SUPPRESS LEADING ZEROS
002000  005337  015554                     DEC P2RETRY             ;HAVE 10 ERRORS OCCURED
002004  001405                             BEQ 1$                  ;YES,LOG HARD ERROR
002006  104400  014751                     TYPE ,MPAR              ;NO,TYPE PARITY ERROR
002012  104400  014004                     TYPE ,MCRLF
002016  000207                             RTS PC                  ;RETURN
```

```
002020 005237 015566         1$:          INC HPARLOG           ;INC HARD PARITY ERROR LOG
002024 104400 014707                      TYPE ,MUNREC          ;PRINT UNRECOVERABLE PARITY ERROR
002030 104400 014751                      TYPE ,MPAR
002034 104400 014004                      TYPE ,MCRLF
002040 104400 014634                      TYPE ,MHALT4
002044 000000                HLT4:        HALT                  ;HALT BECAUSE OF HARD PARITY ERROR
002046 000137 001220                      JMP SA200             ;IF CONTINUE IS PRESSED GO TO START
002052 000000                TEST:        0

002054 005002                TSTLIMITS:   CLR R2                ;MESSAGE FLAG
002056 005737 001200                      TST OD                ;TEST FOR NO SELECTION OF TRACKS
002062 001434                             BEQ 1$                ;IT WILL BE STANDARD LIMITS
002064 105737 001200                      TSTB OD               ;TEST FOR 0 OD TRACK
002070 001425                             BEQ 2$                ;YES,CHECK FOR STANDARD ID TRACK
002072 004737 002266         5$:          JSR PC,TSTMSG         ;HAS INITIAL MESSAGE BEEN TYPED
002076 104400 014442                      TYPE ,MOD             ;TYPE BOTH ID AND OD LIMITS
002102 113737 001200 002114               MOVB OD,7$
002110 004537 013374                      JSR R5,SGLDEC         ;TYPE SINGLE DECIMAL WORD
002114 000000                7$:          OPEN
002116 104400 014001                      TYPE ,DBLSP
002122 104400 014446                      TYPE ,MID
002126 113737 001201 002140               MOVB ID,10$
002134 004537 013374                      JSR R5,SGLDEC
002140 000000                10$:         OPEN
002142 000404                             BR 1$
002144 123727 001201 000114  2$:          CMPB ID,#114          ;TEST FOR MAXIMUM ID LIMIT
002152 001347                             BNE 5$
002154 005737 001202         1$:          TST FIRST             ;TEST FOR NO SELECTION OF SECTORS
002160 001441                             BEQ 13$               ;IT WILL BE STANDARD LIMITS
002162 123727 001202 000001               CMPB FIRST,#1         ;TEST FOR NORMAL FIRST SECTOR LIMIT
002170 001427                             BEQ 4$                ;YES,CHECK LAST SECTOR LIMIT
002172 004737 002266         6$:          JSR PC,TSTMSG
002176 104400 014001                      TYPE ,DBLSP
002202 104400 014452                      TYPE ,MFIRST          ;TYPE BOTH FIRST AND LAST LIMITS
002206 113737 001202 002220               MOVB FIRST,11$
002214 004537 013374                      JSR R5,SGLDEC
002220 000000                11$:         OPEN
002222 104400 014001                      TYPE ,DBLSP
002226 104400 014461                      TYPE ,MLAST
002232 113737 001203 002244               MOVB LAST,12$
002240 004537 013374                      JSR R5,SGLDEC
002244 000000                12$:         OPEN
002246 000404                             BR 3$
002250 123727 001203 000032  4$:          CMPB LAST,#32         ;TEST FOR NORMAL LAST SECTOR LIMIT
002256 001345                             BNE 6$
002260 104400 014172         3$:          TYPE ,DBLLF
002264 000207                13$:         RTS PC


002266 005702                TSTMSG:      TST R2                ;TEST MESSAGE FLAG
002270 001005                             BNE 1$
002272 104400 014400                      TYPE ,MNONSTD         ;MESSAGE HEADING HAD NOT BEEN TYPED
002276 104400 014004                      TYPE ,MCRLF
002302 005202                             INC R2                ;SET THE MESSAGE FLAG
```

```
                                1$:             RTS PC
                                ;
                                ICOND:          TYPE ,MCRLF
                                                TYPE ,MICON             ;TYPE INITIAL CONDITIONS
                                                TSTB @UNITSEL           ;TEST FOR DRIVE SELECTION
                                                BPL 1$
                                                TYPE ,MUNIT0            ;TYPE UNIT 0 SELECTED
                                1$:             TST @UNITSEL
                                                BPL 2$
                                                TYPE ,MUNIT1            ;TYPE UNIT 1 SELECTED
                                2$:             TYPE ,TAB
                                                JSR PC,TYPSEL           ;TYPE TEST SELECTIONS
                                                TYPE ,MCRLF
                                                TST @UNITSEL            ;WERE ANY DRIVES READY
                                                BNE CONT12              ;YES,CONTINUE
                                                TYPE ,MNODRY            ;NO,TYPE NO DRIVES MESSAGE
                                HLT5:           HALT                    ;NO DRIVES CAN'T CONTINUE
                                                JMP SA200               ;IF CONTINUE IS PRESSED RESTART
                                CONT12:         JSR PC,TSTLIMITS        ;TYPE NONSTANDARD TRK AND SEC LIMITS
                                                RTS PC
                                ;
                                HOME:           BIT #SW8,@SWR           ;TEST THE NO RECAL SWITCH
                                                BNE RTN                 ;RETURN IF THE SWITCH IS SET
                                                MOV #10.,P2RETRY        ;SET UP THE PARITY RETRY COUNTER
                                                MOV #10.,DDRETRY        ;USE THE DD RETRY COUNTER FOR RECAL RETRIES
                                2$:             MOV #RECAL,@RXCS        ;ISSUE RECAL FUNCTION
                                                JSR PC,DONECK           ;WAIT FOR DONE FLAG
                                                BIT #2,@RXDB            ;WAS THERE A PARITY ERROR
                                                BEQ 1$                  ;NO,CHECK FOR ANY OTHER ERROR
                                                JSR PC,STATER           ;YES,GO REPORT IT
                                                BR 2$                   ;RETRY RECAL
                                1$:             TST @RXCS               ;IS THE ERROR FLAG SET
                                                BPL XHOME               ;IF NOT RETURN
                                                JSR PC,RDCODE           ;YES,PRINT STATUS REGISTERS
                                                CMPB @STAT,#40          ;IS THE B CODE LESS THAN CODE 40
                                                                        ;(RECAL CODES ARE 10,20,AND 30)
                                                BGE XHOME               ;IF NOT RETURN
                                                DEC DDRETRY             ;HAVE 10 ERRORS OCCURED
                                                BNE 2$                  ;NO,RETRY
                                                TYPE ,MHALT3            ;YES,CAN'T CONTINUE AS RX11 WILL NOT RECAL
                                HLT3:           HALT
                                                JMP SA200               ;IF CONT. SWITCH IS PRESSED GO TO START
                                XHOME:          MOV #1,PRESTRK          ;RESET THE PRESENT TRACK TO TRACK 1
                                RTN:            RTS PC                  ;RETURN
                                ;
                                ;*****************************************************************
                                ;
                                ;DECODE TEST BITS OF INITIAL CONDITIONS
                                ;
                                ;BITS 3,4,AND 5 OF THE INITIAL SWR SELECTED THE TEST WHICH
                                ;IS TO BE PREFORMED. THEY ARE AS FOLLOWS:
                                ;
                                ;   BITS                TESTS
                                ; 5  4  3
```

```
                                        ;  0  0  0          NO TEST SELECTED (DEFAULT TO TEST 7)
                                        ;  0  0  1          WRITE ONLY
                                        ;  0  1  0          WRITE FOLLOWED BY READ
                                        ;  0  1  1          WRITE FOLLOWED BY READ AND VERIFY
                                        ;  1  0  0          READ AND VERIFY ONLY
                                        ;  1  0  1          READ ONLY (CRC CHECK)
                                        ;  1  1  0          READ AND VERIFY DELETED DATA
                                        ;  1  1  1          WRITE FOLLOWED BY READ FOLLOWED BY ANOTHER
                                        ;                   READ AND VERIFY

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

002532  012706  001200          RESTART:    MOV #STACK,SP           ;RESET THE STACK POINTER
002536  012746  000340                        MOV #PR7,-(SP)          ;AND INTERUPT LEVEL
002542  012746  002550                        MOV #1$,-(SP)
002546  000002                               RTI
002550  005237  017600          1$:         INC RESTCNTR            ;INC RESTART COUNTER
002554  004737  002306          XSTART:     JSR PC,ICOND            ;TYPE OUT INITIAL CONDITIONS
002560  005037  010706          TESTSEL:    CLR CHARCT              ;CLEAR EOP CHARACTER COUNTER
002564  012737  000012  015544              MOV #10.,RDRETRY        ;SET UP READ AND WRITE RETRY COUNTERS
002572  012737  000012  015546              MOV #10.,WTRETRY
002600  005037  003746                      CLR ERFLAG              ;CLEAR WRITE CAUSED BY ERROR FLAG
002604  005037  003744                      CLR RDAFTWT             ;CLEAR READ AFTER WRITE FLAG
002610  142737  000377  002646              BICB #377,@#BRONTEST    ;CLEAR OUT BRANCH OFFSET
002616  005737  002052                      TST TEST                ;IF NO TEST SPECIFIED FORCE TEST 7
002622  001003                              BNE 1$
002624  012737  000007  002052              MOV #7,TEST
002632  013704  002052          1$:         MOV TEST,R4
002636  005304                              DEC R4                  ;ADJUST TEST BITS FOR CORRECT
002640  006304                              ASL R4                  ;BRANCH OFFSET
002642  150437  002646                      BISB R4,@#BRONTEST      ;INSERT OFFSET TO BRANCH INST.
002646  000777                  BRONTEST:   BR .                    ;BRANCH BY TEST OFFSET
002650  000137  002704                      JMP WRTONLY             ;WRITE ONLY FUNCTION
002654  000137  003752                      JMP WRTFD               ;WRITE THEN READ FUNCTION
002660  000137  005260                      JMP WTRDCK              ;WRITE THEN READ VERIFY
002664  000137  006236                      JMP RDCHK               ;READ VERIFY
002670  000137  006276                      JMP RDONLY              ;READ ONLY FUNCTION
002674  000137  006336                      JMP DRVSWP              ;WRITE,AND READ VERIFY ON ALTERNATING DRIVES
002700  000137  006422                      JMP TEST7               ;WRITE,READ,FOLLOWED BY READ VERIFY


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

                                        ;THIS IS A WRITE ONLY FUNCTION USING DATA PATTERN AND
                                        ;TRACK SEQUENCE SPECIFIED BY INITIAL SWR SETTINGS
                                        ;  T = 1

002704  004737  010072          WRTONLY:    JSR PC,GETPATTERN       ;SET UP SOFTWARE DATA BUFFER
002710  004737  010710          XWRTONLY:   JSR PC,INITTRACKS       ;SET UP ID,OD,AND TRACK COUNTER
002714  004737  010374                      JSR PC,GETUNIT          ;SET UP DRIVE UNIT SELECTION
002720  004737  011020          1$:         JSR PC,GETTRACK         ;PICK UP NEXT TRACK
002724  004737  002744                      JSR PC,WRITE            ;DO THE WRITE FUNCTION
002730  005337  011122                      DEC TRKCNTR             ;TEST TRACK COUNTER
```

```
                                                BNE 1$
                                                JSR PC,STOP             ;CHECK FOR LAST DRIVE AND EOP
                                                BR XWRTONLY             ;NEXT PASS

                                WRITE:          JSR PC,INITSECTOR       ;SET UP FIRST,LAST,AND SECTOR COUNTER
                                XWRITE:         JSR PC,GETSECTOR        ;PICK UP NEXT SECTOR
                                                MOV #10.,SRETRY
                                ONEWRT:         MOV #10.,PRETRY         ;SET RETRY COUNTER
                                FILLBUF:        JSR PC,ADJSUM           ;ADJUST DATA BUFFER AND CHECK SUM FOR ADDRESSES
                                                MOV #FILLDONE,-(SP)     ;PUT GOOD RETURNON STACK
                                                MOV #FILLER,-(SP)       ;PUT ERROR RETURN ON STACK
                                                CLR BYTECNTR
                                                MOV #PR4,-(SP)
                                                MOV #1$,-(SP)
                                                RTI
                        176156  1$:             MOV #FBIE,@RXCS         ;EXICUTE FILLBUFER COMMAND
                                FILLFLAG:       JSR PC,TRCK             ;TEST FOR TRANSFER REQUEST FLAG
                                XFRBYTE:        MOVB (R0)+,@RXDB        ;TRANSFER DATA BYTE
                                                INC BYTECNTR
                                                BR FILLFLAG             ;WAIT FOR NEXT TR FLAG

                                FILLER:         TST (SP)+               ;REMOVE THE DONE RETURN FROM THE STACK
                                                MOV #MFIL,PTYP1+2       ;PUT ADDR OF FILLBUF MESSAGE IN PAR ERR TYPOUT 1
                                                MOV #MFIL,PTYP2+2       ;PUT ADDR OF FILLBUF MESSAGE IN PAR ERR TYPOUT 2
                                                MOV #FILLBUF,PCONT+2    ;IF NOT HARD ERR RETURN THROUGH PCONT TO FILLBUF
                                                JMP PARTEST             ;PRINT OUT PAR ERR AND TEST CONDITIONS FOR RETRY


                                PARTEST:        INC PARLOG              ;INCREMENT PARITY ERROR
                        176104                  BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
                                                BNE CONT4
                                PTYP1:          TYPE ,OPEN              ;PRINT THE PARITY ERROR MESSAGE
                                                TYPE ,MPAR
                                                TYPE ,MCRLF
                                CONT4:          TST @SWR                        ;TEST HALT ON ERROR SWITCH
                                                BPL CONT13
                                HLT6:           HALT                    ;HALT ON ERROR
                        176050  CONT13:         BIT #SW11,@SWR          ;TEST NO RETRY SWITCH
                                                BNE CONT5               ;IF SET LOG HARD ERROR
                                                DEC PRETRY              ;DECREMENT RETRY COUNTER
                                                TST PRETRY              ;HAVE 10 ERRORS OCCURED
                                                BEQ CONT5               ;IF CLEARED LOG HARD ERROR
                                PCONT:          JMP FILLBUF             ;RETURN TO RETRY TEST THROUGH HERE
                                CONT5:          INC HPRLOG              ;INC.HARD PARITY ERROR  COUNTER
                                                TYPE ,DRLLF
                                                TYPE ,MLTSEC            ;TYPE HARD PARITY ERROR
                                PTYP2:          TYPE ,OPEN
                                                TYPE ,MPAR
                                                TYPE ,MCRLF
                                                JMP DELUNIT             ;GO DELETE THE UNIT,CAN NOT CONTINUE

                ;SWITCH 9 OF INITIAL TEST CONDITIONS SWITCH SETTINGS IS THE DELETED DATA
                ;FUNCTION INDICATOR. WHEN THIS BIT IS SET ALL WRITE / READ FUNCTIONS WILL
                ;SET AND CHECK THE DELETED DATA BIT ON THE DISKETTE.
```

```
                                ;*****************************************************************
                                ;
003230  012737  000012  015556  FILLDONE:    MOV #10.,PRETRY          ;SET UP RETRY COUNTER
003236  012746  003310          REWRITE:     MOV #WRTDONE,-(SP)       ;SET GOOD RETURN ON STACK
003242  012746  003336                       MOV #WRTER,-(SP)         ;SET ERROR RETURN ON STACK
003246  112737  000105  003742               MOVB #WRTIE,FUNCTION     ;SET FUNCTION WORD TO WRITE
003254  032737  001000  001212               BIT #BIT9,DTESTP         ;TEST FOR WRITE DELETED DATA
003262  001403                               BEQ 1$
003264  112737  000115  003742               MOVB #WTDOIE,FUNCTION
003272  062737  000001  017570  1$:          ADD #1,WTCNTR            ;INC TOTAL WRITE FUNCTIONS COUNTER
003300  005537  017572                       ADC WTCNTR+2             ;DOUBLE PRECISION COUNTER
003304  004737  003644                       JSR PC,COMWORD           ;TRANSFER COMMAND TO DRIVE
003310  004737  006714                       JSR PC,DONECK            ;TEST FOR DONE FLAG
003314  000137  006776                       JMP NOINTER              ;NO INTERUPT ERROR
                                ;
003320  005737  003746          WRTDONE:     TST ERWRT                ;IS THIS A REWRITE FROM A DATA ERROR
003324  001004                               BNE 1$                   ;YES GO REREAD THIS SECTOR
003326  005337  012052                       DEC SECCNTR              ;NO,TEST SECTOR COUNTER
003332  001003                               BNE 2$                   ;NOT LAST SECTOR GO TO NEXT ONE
003334  000207                               RTS PC
003336  000137  004070          1$:          JMP REREAD               ;REREAD THE SECTOR
003342  000137  002750          2$:          JMP XWRITE
                                ;
003346  005726                  WRTER:       TST (SP)+                ;REMOVE THE DONE RETURN FROM THE STACK
003350  032737  000002  007650               BIT #BIT1,ASTAT          ;IS THIS A PARITY ERROR
003356  001413                               BEQ WRTSEK               ;NO, IT MUST BE A SEEK ERROR
                                ;PARITY ERROR DURING A WRITE FUNCTION
003360  012737  014742  003114               MOV #MWRITE,PTYP1+2      ;PUT ADDR OF WRITE MESSAGE IN PAR ER TYPOUT 1
003366  012737  014742  003202               MOV #MWRITE,PTYP2+2      ;PUT ADDR OF WRITE MESSAGE IN PAR ER TYPOUT 2
003374  012737  003226  003162               MOV #REWRITE,PCONT+2     ;IF NOT HARD ERR RETURN THROUGH PCONT TO REWRITE
003402  000137  003076                       JMP PARTEST              ;GO INC LOG AND TEST FOR RETRY
                                ;
                                ;SEEK ERROR DURING A WRITE FUNCTION
003406  012737  002762  003516  WRTSEK:      MOV #ONEWRT,SEKRTY+2     ;SETUP FOR WRT RETRY ON SEEK ERROR
                                ;(AFTER A RECAL. THE CONTENTS OF SECTOR 1,
                                ;TRACK 1 ARE LOADED INTO THE SECTOR BUFFER.
                                ;TO REWRITE THE CORRECT DATA THE PROGRAM
                                ;MUST REFILL THE SECTOR BUFFER.
003414  012737  014742  003454               MOV #MWRITE,STYP1+2      ;PUT ADDR OF WRITE MESSAGE IN SEEK ER TYPEOUT 1
003422  012737  014742  003544               MOV #MWRITE,STYP2+2      ;PUT ADDR OF WRITE MESSAGE IN SEEK ER TYPEOUT 2
003430  004737  003434                       JSR PC,SEEKER            ;RECORD SEEK ERROR
                                ;
003434  012703  015576  SEEKER:               MOV #ZSEKLOG,R3          ;SETUP INC INSTRUCTION FOR UNIT 0 LOG
003440  004737  017604                       JSR PC,U1LOG             ;TEST FOR UNIT 1 LOG
003444  005213                               INC (R3)                 ;INCREMENT SEEK ERROR LOG
003446  004737  003612                       JSR PC,TRKERR            ;INC ERROR PER TRACK COUNTERS
003452  032777  020000  175544               BIT #SW13,@SWR           ;CHECK DON'T PRINT ERROR SWITCH
003460  001004                               BNE SWHLT1
003462  104400  014742  STYP1:               TYPE ,MWRITE             ;PRINT WRITE (READ) SEEK ERROR
003466  004737  003560                       JSR PC,SEKTYP
003472  005777  175526  SWHLT1:              TST @SWR                 ;TEST THE HALT ON ERROR SWITCH
003476  100001                               BPL CONT14
003500  000000  HLT7:                       HALT                     ;HALT ON THE ERROR
003502  032777  004000  175514  CONT14:      BIT #SW11,@SWR           ;CHECK THE NO RETRY SWITCH
```

```
003500  001007                          BNE HARDSK              ;IF SET LOG HARD SEEK ERROR
003502  005337  015562                   DEC SRETRY              ;HAVE 10 ERRORS BEEN LOGED
003506  001404                          BEQ HARDSK              ;YES LOG HARD ERROR
003510  004737  002404                   JSR PC,HOME             ;RECALIBRATE DRIVES ON SEEK ERROR
003514  000137  002762          SEKRTY: JMP ONEWRT              ;NO,RETRY WRITE COMMAND (READ COMAND)
003520  012703  015636          HARDSK: MOV #ZHSEKLOG,R3        ;SET INC. INST.FOR UNIT 0 ERR LOG
003524  004737  017604                   JSR PC,U1LOG            ;TEST FOR UNIT 1 ERROR LOG
003530  005213                          INC (R3)                ;HARD SEEK ERROR
003532  104400  014172                   TYPE ,DBLLF
003536  104400  014707                   TYPE ,MUNREC            ;TYPE UNRECOVERABLE SEEK ERROR
003542  104400  014742          STYP2:  TYPE ,MWRITE            ;ON WRITE (READ) COMMAND
003546  004737  003560                   JSR PC,SEKTYP
003552  062706  000002                   ADD #2,SP               ;REMOVE SEEK ERROR FROM STACK POINTER
003556  000207                          RTS PC                  ;RETURN TO NEXT SECTOR BY DONE RETURN ON STACK

003560  104400  014727          SEKTYP: TYPE ,MSEEK             ;TYPE SEEK ERROR
003564  104400  013521                   TYPE ,MPRES             ;TYPE ADDRESS OF TRACK MOVED FROM
003570  013737  011126  003602           MOV PRESTRK,1$
003576  004537  013374                   JSR R5,SGLDEC
003602  000000                  1$:     OPEN
003604  104400  014004                   TYPE ,MCRLF
003610  000207                          RTS PC

003612  013705  011124          TRKERR: MOV TARGET,R5           ;SET UP TO INC ERROR PER TRACK COUNTER
003616  006305                          ASL R5                  ;ADJUST FOR EVEN ADDRESS
003620  062705  017104                   ADD #U0TRK,R5           ;ADDIN ADDERES OF UNIT 0 LOG
003624  032737  000020  010464           BIT #BIT4,UNITSEL       ;CHECK THE UNIT SELECTION BIT
003632  001402                          BEQ 1$                  ;IF CLEARED UNIT 0 IS ACTIVE
003634  062705  017336                   ADD #U1TRK,R5           ;ADJUST FOR UNIT 1 LOG
003640  005215                  1$:     INC (R5)                ;INC THE CORRECT COUNTER
003642  000207                          RTS PC

003644  013705  011124          COMMWORD: MOV TARGET,R5         ;GET TRACK NUMBER
003650  006305                          ASL R5                  ;MULTIPLY BY 4 TO DOUBLE PRECISION
003652  006305                          ASL R5                  ;INTERLEAVE COUNTER LOCATIONS
003654  062705  015734                   ADD #TKACC,R5           ;ADD ON ADDRESS OF TRACK ACCESS COUNTER
003660  062715  000001                   ADD #1,(R5)             ;INCREMENT THE COUNTER
003664  005565  000002                   ADC 2(R5)               ;ADD CARRY TO HIGH ORDER WORD
003670  153737  010464  003742           BISB UNITSEL,FUNCTION   ;SET UNIT SELECTION BIT IN COMMAND WORD
003676  013777  003742  175302           MOV FUNCTION,@RXCS      ;SEND OUT COMMAND TO DRIVE
003704  004737  006606                   JSR PC,TRCK             ;WAIT FOR TR FLAG
003710  113777  012054  175272           MOVB TSECTOR,@RXDB      ;SEND OUT TARGET SECTOR
003716  004737  006606                   JSR PC,TRCK             ;WAIT FOR TR FLAG
003722  113777  011124  175260           MOVB TARGET,@RXDB       ;SEND OUT TARGET TRACK
003730  005046                          CLR -(SP)               ;LOWER INTERUPT LEVEL TO ALLOW AN INTERUPT
003732  012746  003740                   MOV #1$,-(SP)
003736  000002                          RTI
003740  000207                  1$:     RTS PC

003742  000000          FUNCTION: 0
003744  000000          RDAFTWT:  0     ;READ AFTER WRITE FUNCTION FLAG
003746  000000          ERWRT:    0     ;WRITE CAUSED BY DATA ERROR FLAG
003750  000000          DATACK:   0     ;DATA CHECK ON CRC ERROR FLAG
```

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

                                        ;THIS IS A WRITE ALL SECTORS FOLLOWED BY A READ ALL SECTORS
                                        ;WITH DATA PATTERNS AND HEAD SEQUENCING SET BY INITIAL SWR
                                        ;   T = 2

003752  005137  003744          WRTRD:          COM RDAFTWT             ;SET READ AFTER WRITE FLAG
003756  004737  010072                          JSR PC,GETPATTERN
003762  004737  010710          XWRTRD:         JSR PC,INITTRACKS
003766  004737  010374                          JSR PC,GETUNIT
003772  004737  011020          1$:             JSR PC,GETTRACK
003776  004737  002744                          JSR PC,WRITE
004002  004737  004022                          JSR PC,READ
004006  005337  011122                          DEC TRKCNTR
004012  001367                                  BNE 1$
004014  004737  010512                          JSR PC,STOP
004020  000760                                  BR XWRTRD

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

                                        ;READ DATA FROM THE DISKETTE

004022  004737  011762          READ:           JSR PC,INITSECTOR
004026  004737  012062          XREAD:          JSR PC,GETSECTOR
004032  012737  000012  015550                  MOV #10.,DDRETRY        ;SET UP RETRY COUNTERS FOR DELETED DATA
004040  012737  000012  015552                  MOV #10.,DATARETRY      ;DATA ERROR
004046  012737  000012  015556                  MOV #10.,PRETRY         ;PARITY
004054  012737  000012  015562                  MOV #10.,SRETRY         ;SEEK
004062  012737  000012  015560                  MOV #10.,CRETRY         ;AND CRC ERRORS
004070  005037  003750          REREAD:         CLR DATACK              ;CLEAR CRC DATA CHECK FLAG
004074  012746  004140                          MOV #RDDONE,-(SP)       ;SET GOOD RETURN ON STACK
004100  012746  004302                          MOV #RDERR,-(SP)        ;SET READ ERROR RETURN ON STACK
004104  112737  000107  003742                  MOVB #RDIE,FUNCTION
004112  062737  000001  017574                  ADD #1,RDCNTR           ;INC TOTAL READ FUNCTIONS COUNTER
004120  005537  017576                          ADC RDCNTR+2            ;DOUBLE PRECISION COUNTER
004124  004737  003644                          JSR PC,COMMWORD
004130  004737  006714                          JSR PC,DONECK           ;TEST FOR DONE
004134  000137  006776                          JMP NOINTER             ;NO INTERUPT ON DONE


004140  022737  000012  015544  RDDONE:         CMP #10.,RDRETRY        ;IS READ RETRY EQUAL TO 10
004146  001420                                  BEQ CONT1               ;YES,NO ERRORS OCCURED
004150  012703  015612                          MOV #ZRDLOG,R3          ;SET INC. INST. FOR UNIT 0 ERROR LOG
004154  004737  017604                          JSR PC,UILOG            ;TEST FOR UNIT 1 ERROR LOG
004160  005213                                  INC (R3)                ;INC RECOVERABLE READ LOG
004162  012737  000012  015544                  MOV #10.,RDRETRY        ;RESET READ RETRY COUNTER
004170  104400  014172                          TYPE ,DALLF
004174  104400  014561                          TYPE ,MFC               ;TYPE RECOVERABLE READ ERROR
004200  104400  014626                          TYPE ,MREAD
004204  104400  014004                          TYPE ,MCRLF
004210  022737  000012  015546  CONT1:         CMP #10.,WTRETRY        ;IS WRITE RETRY EQUAL TO 10
004216  001420                                  BEQ CONT2               ;YES,NO ERRORS OCCURED
```

```
                                                MOV #ZWRTLOG,R3         ;SET INC.INST.FOR UNIT 0 ERROR LOG
                                                JSR PC,U1LOG            ;TEST FOR UNIT 1 LOG
                                                INC (R3)                ;INC RECOVERABLE WRITE LOG
                                                MOV #10.,WTRETRY        ;RESET THE WRITE RETRY COUNTER
                                                TYPE ,DBLLF
                                                TYPE ,MREC              ;TYPE RECOVERABLE WRITE ERROR
                                                TYPE ,MWRITE
                                                TYPE ,MCRLF
                                CONT2:          JSR PC,DDCHK            ;CHECK FOR DELETED DATA INDICATOR
                                                TST R1                  ;BIT 15 OF R1 IS READ 1 SECTOR FLAG
                                                BPL NEXTRD
                                                RTS PC                  ;IF SET,GO VERIFY DATA JUST READ
                                NEXTRD:         DEC SECCNTR
                                                BNE XREAD
                                                RTS PC                  ;READ FUNCTION IS DONE

                                RDERR:          TST (SP)+               ;REMOVE THE DONE RETURN FROM THE STACK
                                                BIT #BIT1,ASTAT         ;IS THIS A PARITY ERROR
                                                BEQ 1$                  ;NO, SEE IF ITS A CRC ERROR
                                        ;PARITY ERROR DURING A READ FUNCTION
                                                MOV #MREAD,PTYP1+2      ;PUT ADDR OF READ MESSAGE IN PAR ERR TYPEOUT 1
                                                MOV #MREAD,PTYP2+2      ;PUT ADDR OF READ MESSAGE IN PAR ER TYPEOUT 2
                                                MOV #REREAD,PCONT+2     ;IF HARD ERR RETURN THROUGH PCONT TO REREAD
                                                JMP PARTEST             ;RECORD PARITY ERROR AND RETRY FUNCTION
                                1$:             BIT #BIT0,ASTAT         ;IS THIS A CRC ERROR
                                                BNE CRCER               ;YES GO TEST AND LOG IT
                                        ;SEEK ERROR DURING A READ FUNCTION
                                                MOV #REREAD,SEKRTY+2    ;SET SEEK CONTINUE FOR READ RETRY
                                                MOV #MREAD,STYP1+2      ;SET ADDR OF READ MESSAGE IN SEEK ER TYPEOUT 1
                                                MOV #MREAD,STYP2+2      ;SET ADDR OF READ MESSAGE IN SEEK ER TYPEOUT 2
                                                JSR PC,SEEKER           ;RECORD SEEK ERROR
                                                BR NEXTRD               ;GO TO NEXT SECTOR,CAN'T READ THIS ONE
                                        ;CRC ERROR DETECTED WHILE READING
                                CRCER:          JSR PC,TRKERR           ;INC ERROR PER TRACK COUNTER
                                                TST R1                  ;IF READ ONLY, REPORT DATA CRC ERROR
                                                BPL DATACRC
                                                INC DATACK              ;SET DATA CHECK FLAG
                                                JSR PC,EMPBUFF          ;CHECK FOR A DATA ERROR
                                                TST ERCNTR              ;WAS THERE A DATA ERROR
                                                BNE DATACRC             ;YES,REREAD AND/OR REWRITE THE DATA
                                                MOV #ZCRCBAD,R3         ;SET INC.INST.FOR UNIT 0 ERROR LOG
                                                JSR PC,U1LOG            ;TEST FOR UNIT 1 ERROR LOGS
                                                INC (R3)                ;10,INC BAD CRC GENERATOR ERROR
                                                BIT #SW13,@SWR          ;TEST DON'T SWITCH
                                                BNE 2$
                                                TYPE ,MBADCRC           ;TYPE CRC GENERATOR ERROR
                                                TYPE ,MCRLF
                                2$:             TST @SWR                ;TEST HALT ON ERROR SWITCH
                                                BPL CONT15
                                HLT10:          HALT                    ;HALT ON ERROR
                                CONT15:         BIT #SW11,@SWR          ;CHECK NO RETRY SIWTCH
                                                BNE 3$                  ;IF SET LOG HARD ERROR
                                                DEC CRETRY              ;HAVE 10 ERRORS BEEN LOGED
                                                BEQ 3$                  ;YES,LOG HARD ERROR
```

```
 973  004510  000137  004070            JMP REREAD              ;NO,GO REREAD DATA
 974  004514  012703  015646     3$:    MOV #ZHCRCBAD,R3        ;SET INC.INST.FOR UNIT 0 ERROR LOG
 975  004520  004737  017604            JSR PC,U1LOG            ;TEST FOR UNIT 1 ERROR LOGS
 976  004524  005213                    INC (R3)
 977  004526  104400  014172            TYPE ,DBLLF
 978  004532  104400  014576            TYPE ,MHRDCRC           ;TYPE HARD CRC GENERATOR ERROR
 979  004536  104400  014004            TYPE ,MCRLF
 980  004542  104400  014656            TYPE ,MHALT11           ;AND HALT AS THE CRC GENERATOR DOESN'T WORK
 981  004546  000000            HLT11:  HALT
 982                        ;DATA CRC ERROR REREAD AND/OR REWRITE TO GET GOOD DATA
 983  004550  012703  015602    DATACRC: MOV #ZCRCLOG,R3        ;SET INC.INST.FOR UNIT 0 ERROR LOG
 984  004554  004737  017604            JSR PC,U1LOG            ;TEST FOR UNIT 1 ERROR LOGS
 985  004560  005213                    INC (R3)                ;TRUE DATA CRC ERROR
 986  004562  032777  020000  174424    BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
 987  004570  001004                    BNE 4$
 988  004572  104400  014670            TYPE ,MCRC              ;TYPE DATA CRC ERROR
 989  004576  104400  014004            TYPE ,MCRLF
 990  004602  005777  174406     4$:    TST @SWR                ;TEST HALT ON ERROR SWITCH
 991  004606  100001                    BPL CONT16
 992  004610  000000            HLT12:  HALT                    ;HALT ON ERROR
 993  004612  032777  004000  174374 CONT16: BIT #SW11,@SWR     ;TEST NO RETRY SWITCH
 994  004620  001005                    BNE 5$                  ;IF SET LOG HARD ERROR
 995  004622  005337  015544            DEC RDRETRY             ;HAVE 10 ERRORS OCCURED
 996  004626  001402                    BEQ 5$                  ;YES LOG HARD ERROR OR REWRITE DATA
 997  004630  000737  004070            JMP REREAD              ;NO,GO REREAD THIS SECTOR
 998  004634  012703  015642     5$:    MOV #ZHCRCLOG,R3        ;SET INC.INST.FOR UNIT 0 ERROR LOG
 999  004640  004737  017604            JSR PC,U1LOG            ;TEST FOR UNIY 1 ERROR LOGS
1000  004644  005213                    INC (R3)                ;INC HARD CRC ERROR LOG
1001  004646  012737  000012  015544    MOV #10.,RDRETRY        ;RESET READ RETRY COUNTER
1002  004654  005737  003744            TST RDAFTWT             ;IS THIS A READ AFTER WRITE FUNCTION
1003  004660  001021                    BNE CONT6               ;YES,GO REWRITE IT
1004  004662  012703  015652            MOV #ZHROLOG,R3         ;NO,SET INC.INST.FOR UNIT 0 ERROR LOG
1005  004666  004737  017604            JSR PC,U1LOG            ;TEST FOR UNIT 1 ERROR LOGS
1006  004672  005213                    INC (R3)                ;READ ONLY, HARD READ ERROR
1007  004674  104400  014172            TYPE ,DBLLF
1008  004700  104400  014707            TYPE ,MUNREC            ;TYPE UNRECOVERABLE READ ERROR
1009  004704  104400  014626            TYPE ,MREAD
1010  004710  104400  014004    CONT7:  TYPE ,MCRLF
1011  004714  062706  000002            ADD #2,SP               ;REMOVE READ DONE ADDRESS FROM STACK
1012  004720  000137  004272            JMP NEXTRD              ;READ NEXT SECTOR CAN'T READ THIS ONE
1013  004724  032777  004000  174262 CONT6: BIT #SW11,@SWR      ;TEST NO RETRY SWITCH
1014  004732  001011                    BNE 7$                  ;IF SET LOG HARD ERROR
1015  004734  005337  015546            DEC WTRETRY             ;HAVE 10 REWRITES BEEN DONE
1016  004740  001406                    BEQ 7$                  ;YES,LOG HARD WRITE ERROR
1017  004742  005137  003746            COM ERWRT               ;NO,SET THE WRITE CAUSED BY ERROR FLAG
1018  004746  062706  000002            ADD#2,SP                ;REMOVE RDDONE ADDRESS FROM STACK
1019  004752  000137  002762            JMP ONEWRT              ;REWRITE SECTOR THEN REREAD TO CHECK FOR ERROR
1020  004756  012703  015656     7$:    MOV #ZHWRTLOG,R3        ;SET INC.INST.FOR UNIT 0 ERROR LOG
1021  004762  004737  017604            JSR PC,U1LOG            ;TEST FOR UNIT 1 ERROR LOGS
1022  004766  005213                    INC (R3)                ;HARD WRITE ERROR
1023  004770  012737  000012  015546    MOV #10.,WTRETRY        ;RESET WRITE RETRY COUNTER
1024  004776  104400  014172            TYPE ,DBLLF
1025  005002  104400  014707            TYPE ,MUNREC            ;TYPE UNRECOVERABLE WRITE ERROR
1026  005006  104400  014742            TYPE ,MWRITE
```

```
1027  005012  000736                          BR CONT7
1028
1029
1030
1031  005014  032737  001000  001212  DOCHK:  BIT #BIT9,DTESTP        ;TEST BIT 9 AS TO DELETED DATA TRANSFER
1032  005022  001470                          BEQ CONT10
1033  005024  132737  000100  007650          BITB #BIT6,ASTAT        ;THIS IS A DELETED DATA FUNCTION
1034  005032  001111                          BNE RETURN              ;DO BIT SHOULD BE SET
1035  005034  012703  015626                  MOV #ZDOMIS,R3          ;SET INC INST FOR UNIT 0 ERROR LOG
1036  005040  004737  017604                  JSR PC,U1LOG            ;TEST FOR UNIT 1 ERROR LOGS
1037  005044  005213                          INC (R3)                ;INC MISSING DELETED DATA LOG
1038  005046  004737  003612                  JSR PC,TRKERR           ;INC ERROR PER TRACK COUNTER
1039  005052  032777  020000  174134          BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
1040  005060  001011                          BNE CONT11
1041  005062  104400  013440                  TYPE ,MDOMIS            ;TYPE MISSING DELETED DATA BIT
1042  005066  052737  000400  010464  DOERR:  BIS #BIT8,UNITSEL        ;SET HAD ERROR FLAG
1043  005074  004737  007476                  JSR PC,TYPADR           ;TYPE ADDRESS OF ERROR
1044  005100  104400  014004                  TYPE ,MCRLF
1045  005104  005777  174104          CONT11: TST @SWR                ;TEST HALT ON ERROR SWITCH
1046  005110  100001                          BPL CONT17
1047  005112  000000                  HLT13:  HALT                    ;HALT ON DELETED DATA ERROR
1048  005114  032777  004000  174072  CONT17: BIT #SW11,@SWR          ;TEST NO RETRY SWITCH
1049  005122  001005                          BNE 4$                  ;IF SET LOG HARD ERROR
1050  005124  005337  015550                  DEC DORETRY             ;HAVE 10 ERRORS BEEN LOGED
1051  005130  001402                          BEQ 4$                  ;YES LOG HARD ERROR
1052  005132  000137  004070                  JMP REREAD              ;NO, REREAD SECTOR
1053  005136  012703  015666          4$:     MOV #ZHDDLOG,R3         ;SET INC INST.FOR UNIT 0 ERROR LOG
1054  005142  004737  017604                  JSR PC,U1LOG            ;TEST FOR UNIT 1 ERROR LOGS
1055  005146  005213                          INC (R3)
1056  005150  104400  014172                  TYPE ,DBLLF
1057  005154  104400  014707                  TYPE ,MUNREC            ;TYPE UNRECOVERABLE DELETED DATA ERROR
1058  005160  104400  014547                  TYPE ,MDDER
1059  005164  104400  014004                  TYPE ,MCRLF
1060  005170  004737  007476                  JSR PC,TYPADR
1061  005174  104400  014004                  TYPE ,MCRLF
1062  005200  000137  004272                  JMP NEXTRD              ;READ NEXT SECTOR
1063  005204  032737  000100  007650  CONT10: BIT #BIT6,ASTAT         ;THIS IS NOT A DELETED DATA TRANSFER
1064  005212  001421                          BEQ RETURN
1065  005214  012703  015632                  MOV #ZUNXDD,R3          ;SET INC INST.FOR UNIT 0 ERROR LOG
1066  005220  004737  017604                  JSR PC,U1LOG            ;TEST FOR UNIT 1 ERROR LOGS
1067  005224  005213                          INC (R3)                ;UNEXPECTED DD BIT SET
1068  005226  052737  000400  010464          BIS #BIT8,UNITSEL       ;SET HAD ERROR FLAG
1069  005234  004737  003612                  JSR PC,TRKERR           ;INC ERROR PER TRACK COUNTER
1070  005240  032777  020000  173746          BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
1071  005246  001316                          BNE CONT11
1072  005250  104400  013414                  TYPE ,MUNXDD            ;TYPE UNEXPECTED DELETED DATA BIT
1073  005254  000704                          BR DOERR
1074  005256  000207                  RETURN: RTS PC
1075
1076
1077
1078                          ;*****************************************************************
1079
1080                                  ;WRITE ALL SECTORS,READ AND VERIFY ALL SECTORS
```

```
1081                                            ;   T = 3
1082
1083  005260  005137  003744    WTRDCK:         COM RDAFTWT             ;SET READ AFTER WRITE FLAG
1084  005264  004737  010072                    JSR PC,GETPATTERN
1085  005270  004737  010710    XWTRDCK:        JSR PC,INITTRACKS
1086  005274  004737  010374                    JSR PC,GETUNIT
1087  005300  004737  011020    1$:             JSR PC,GETTRACK
1088  005304  004737  002744                    JSR PC,WRITE
1089  005310  004737  005330                    JSR PC,READCHK
1090  005314  005337  011122                    DEC TRKCNTR
1091  005320  001367                            BNE 1$
1092  005322  004737  010512                    JSR PC,STOP
1093  005326  000760                            BR XWTRDCK
1094
1095
1096                                    ;*****************************************************************
1097
1098                                    ;READ A SECTOR,EMPTY THE SECTOR BUFFER AND VERIFY
1099                                    ;THE DATA READ AGAINST CORE DATA BUFFER
1100
1101  005330  052701  100000    READCHK:        BIS #BIT15,R1           ;SET READ ONE SECTOR FLAG
1102  005334  004737  004022                    JSR PC,READ             ;GO READ ONE SECTOR
1103  005340  005737  012052    EMPBUFF:        TST SECCNTR             ;IF CLEARED NO SECTOR WAS FOUND
1104  005344  001002                            BNE 2$
1105  005346  000137  006224                    JMP EXIT                ;GO TO NEXT TRACK CAN'T READ THIS ONE
1106  005352  005037  006226    2$:             CLR BYTECNTR            ;CLEAR THE BYTE AND ERROR COUNTERS
1107  005356  005037  006234                    CLR ERCNTR
1108  005362  052701  000200                    BIS #BIT7,R1            ;R1 BIT 7 IS USED AS FIRST ERROR FLAG
1109  005366  004737  005460                    JSR PC,ADJSUM           ;ADJUST DATA AND CK SUM FOR ADDRESSES
1110  005372  005037  005550                    CLR CKSUM               ;SET UP FOR CHECK SUM ACCUMULATION
1111  005376  012746  006010                    MOV #E[illegible]DONE,-(SP)     ;SET UP RETURN ADDRESSES
1112  005402  012746  005552                    MOV #E[illegible]ER,-(SP)
1113  005406  005046                            CLR -(SP)               ;LOWER INTERUPT LEVEL
1114  005410  012746  005416                    MOV #1$,-(SP)
1115  005414  000002                            RTI
1116  005416  012777  000103  173562  1$:       MOV #EBIE,@RXCS         ;LOAD EMPTY BUFFER FUNCTION
1117  005424  004737  006606        EMPFLAG:    JSR PC,TRCK             ;TEST FOR TR FLAG
1118
1119  005430  117737  173554  006230  CKBYTE:   MOVB @RXDB,BADBYTE     ;SAVE BYTE FROM DISKETTE
1120  005436  063737  006230  005[illegible]50  ADD BA[illegible]BYTE,CKSUM      ;ACCUMULATE CHECK SUM
1121  005444  123720  006230                    CMPB BADBYTE,(R0)+      ;COMPARE AGAINST GOOD BYTE
1122  005450  001054                            BNE DATAER              ;IF NOT EQUAL GO TO DATAER
1123  005452  005237  006226                    INC BYTECNTR
1124  005456  000762                            BR EMPFLAG              ;GET NEXT BYTE
1125
1126  005460  113737  011124  015344  ADJSUM:   MOVB TARGET,BUFADR      ;SET FIRST AND SECOND BYTES WITH ADDRESSES
1127  005466  113737  012054  015345            MOVB TSECTOR,BUFADR+1
1128  005474  013737  010372  00[illegible]50   MOV SUM,CKSUM           ;GET THE PATTERN SUM
1129  005502  063737  011124  00[illegible]50   ADD TARGET,CKSUM        ;ADD TRACK ADDRESS TO CHECK SUM
1130  005510  063737  012054  005550            ADD TSECTOR,CKSUM       ;ADD SECTOR ADDRESS TO CHECK SUM
1131  005516  113737  005550  015542            MOVB CKSUM,BUFADR+176   ;INSERT CHECK SUM TO DATA BUFFER
1132  005524  106337  005550                    AS[illegible]3 CKSUM            ;GENERATE NEGITIVE CHECK SUM
1133  005530  105437  005550                    NEG[illegible]8 CKSUM
1134  005534  113737  005550  015543            MOVB CKSUM,BUFADR+177   ;INSERT NEG,SUM INTO DATA BUFFER
```

```
1135  005542  012700  015344                        MOV #BUFADR,R0          ;SET ADDRESS OF BYTE IN R0
1136  005546  000207                                 RTS PC                  ;RETURN
1137
1138  005550  000000                    CKSUM:       0
1139
1140  005552  005726                    EMPER:       TST (SP)+               ;REMOVE THE DONE RETURN FROM THE STACK
1141  005554  012737  014334  003114                 MOV #MEMPTY,PTYP1+2     ;PUT ADDR OF EMPTYBUF MESSAGE IN PAR ER TYPOUT 1
1142  005562  012737  014334  003202                 MOV #MEMPTY,PTYP2+2     ;PUT ADDR OF EMPTYBUF MESSAGE IN PAR ER TYPOUT 2
1143  005570  012737  005340  003162                 MOV #E[illegible]UFF,PCONT+2  ;NOT HARD ERR RETURN THROUGH PCONT TO EMPTYBUF
1144  005576  000137  003076                        JMP PARTEST             ;REPORT PARITY ERROR
1145
1146  005602  052737  000400  010464    DATAER:    BIS #BIT8,UNITSEL       ;SET THE HAD ERROR FLAG
1147  005610  005237  006234                        INC ERCNTR              ;INC THE BYTE ERROR COUNTER
1148  005614  032777  020000  173372                BIT #SW13,@SWR          ;TEST PRINT ERROR SW IN SWR
1149  005622  001062                                BNE NOERTYP             ;DON'T PRINT THE ERROR
1150  005624  032777  010000  173362                BIT #SW12,@SWR          ;TEST PRINT 10 ERRORS SWITCH
1151  005632  001404                                BEQ 1$                  ;PRINT ALL ERRORS
1152  005634  023727  006234  000012                CMP ER[illegible]NTR,#10.  ;HAVE 10 ERRORS BEEN TYPED
1153  005642  003052                                BGT NOERTYP             ;YES,DON'T PRINT ANY MORE
1154  005644  005737  003750             1$:        TST DATACK              ;WAS THIS A READ CHECK DUE TO CRC ERROR
1155  005650  001403                                BEQ 2$                  ;NO,REPORT DATA ERRORS
1156  005[illegible]52  105701                      TSTB R1                 ;YES,TEST FIRST ERROR FLAG
1157  005[illegible]54  100017                      BPL TYPERR              ;TYPE THE DATA CRC ERROR
1158  005[illegible]56  000412                      BR 3$
1159  005660  105701                    2$:        TSTB R1                 ;TEST FIRST ERROR FLAG
1160  005[illegible]62  100014                      BPL TYPERR
1161  005[illegible]64  104400  014004              TYPE ,MCRLF
1162  005[illegible]70  104400  013461              TYPE ,MDERHDR           ;PRINT ERROR HEADER
1163  005674  104400  014004                        TYPE ,MCRLF
1164  005700  004737  007476                        JSR PC,TYPADR           ;PRINT TRACK AND SECTOR LOCATIONS
1165  005704  104400  013550             3$:        TYPE ,MCOL[illegible]JN  ;SET UP COLMUN HEADINGS
1166  005710  042701  000200                        BIC #BIT7,R1            ;CLEAR FIRST ER[illegible]R FLAG
1167  005714  013737  006226  005726    TYPERR:    MOV BYTECNTR,1$         ;PRINT BYTE NUMBER
1168  005722  004537  013374                        JSR R5,SGLDEC
1169  005726  000000                    1$:        OPEN
1170  005730  104400  014001                        TYPE ,DPLSP
1171  005734  013746  006230                        MOV BADBYTE,-(SP)       ;PRINT BYTE READ FROM DISKETTE
1172  005740  104402                                TYPOS
1173  005[illegible]742  000003                     .WORD 3
1174  005[illegible]744  104400  014001              TYPE ,DPLSP
1175  005[illegible]750  114037  006232              MOVB -(R0),GOODBYTE     ;GET GOOD BYTE
1176  005[illegible]754  005[illegible]00           INC R0                  ;RETURN R0 TO NEXT BYTE IN BUFFER
1177  005[illegible]6  013[illegible]46  006232      MOV GOODBYTE,-(SP)
1178  005[illegible]762  104403                      TYPON                   ;PRINT GOOD DATA
1179  005[illegible]764  104[illegible]0  014004    TYPE ,MCRLF
1180  005[illegible]770  005777  173220    NOERTYP:   TST @SWR                ;TEST HALT ON ERROR SWITCH
1181  005[illegible]774  100001                      BPL CONT20
1182  005[illegible]776  005[illegible]00           HLT14:  HALT
1183  006000  005[illegible]37  005[illegible]26    CONT20:  INC BYTECNTR
1184  006004  000137  005424                        JMP EMPFLAG
1185
1186  006010  005737  003750             EMPDONE:   TST DATACK              ;WAS THIS READCHECK CAUSED BY CRC ERROR
1187  006014  001401                                BEQ 1$                  ;NO,CONTINUE
1188  006016  000207                                RTS PC                  ;YES,RETURN TO CRC ERROR HANDLER
```

```
1189  006020  005737  006234          1$:     TST ERCNTR              ;WAS THERE ERRORS
1190  006024  001467                          BEQ CONT3               ;NO ERRORS
1191  006026  032777  020000  173160          BIT #SW13,@SWR          ;YES,TEST DON'T PRINT SWITCH
1192  006034  001024                          BNE 2$                  ;DON'T PRINT THE ERROR
1193  006036  104400  014265                  TYPE ,MERCT             ;PRINT THE TOTAL DATA ERROR COUNT
1194  006042  013737  006234  006054          MOV ERCNTR,3$
1195  006050  004537  013374                  JSR R5,SGLDEC
1196  006054  000000                  3$:     OPEN
1197  006056  104400  015261                  TYPE ,MSUM              ;INDICATE IF CHECK SUM WAS GOOD OR HAD ERRORS
1198  006062  105737  005550                  TSTB CKSUM
1199  006066  001403                          BEQ 4$
1200  006070  104400  015021                  TYPE ,MERS
1201  006074  000402                          BR 5$
1202  006076  104400  015254          4$:     TYPE,MGOOD
1203  006102  104400  014004          5$:     TYPE ,MCRLF
1204  006106  012703  015622          2$:     MOV #ZDATALOG,R3        ;SET INC.INST.FOR UNIT 0 ERROR LOG
1205  006112  004737  017604                  JSR PC,U1LOG            ;TEST FOR UNIT 1 ERROR LOGS
1206  006116  005213                          INC (R3)                ;INC DATA ERROR LOG
1207  006120  004737  003612                  JSR PC,TRKERR           ;INC ERROR PER TRACK COUNTER
1208  006124  032777  004000  173062          BIT #SW11,@SWR          ;TEST NO RETRY SWITCH
1209  006132  001007                          BNE 6$                  ;IF SET LOG HARD ERROR
1210  006134  005337  015552                  DEC DATARETRY           ;HAVE 10 ERRORS BEEN LOGED
1211  006140  001404                          BEQ 6$
1212  006142  004737  004070                  JSR PC,REREAD           ;NO,GO REREAD THE DATA
1213  006146  000137  005340                  JMP EMPBUFF             ;GO RECHECK THE DATA
1214  006152  012703  015662          6$:     MOV #ZHDATALOG,R3       ;YES,SET INC.INST.FOR UNIT 0 ERROR LOG
1215  006156  004737  017604                  JSR PC,U1LOG            ;TEST FOR UNIT 1 ERROR LOGS
1216  006162  005213                          INC (R3)                ;INC HARD DATA NO STATUS ERROR
1217  006164  104400  014172                  TYPE ,DBLLF
1218  006170  104400  014707                  TYPE ,MUNREC            ;TYPE UNRECOVERABLE DATA NO
1219  006174  104400  013461                  TYPE ,MDERHDR           ;STATUS ERROR
1220  006200  104400  014004                  TYPE ,MCRLF
1221  006204  005337  012052  CONT3:         DEC SECCNTR
1222  006210  001405                          BEQ EXIT
1223  006212  004737  004026                  JSR PC,XREAD            ;READ THE NEXT SECTOR
1224  006216  000137  005340                  JMP EMPBUFF
1225  006222  005001          3$:             CLR R1                  ;CLEAR THE ONE READ FLAG
1226  006224  000207          EXIT:           RTS PC
1227
1228  006226  000000          BYTECNTR:       0
1229  006230  000000          BADBYTE:        0
1230  006232  000000          GOODBYTE:       0
1231  006234  000000          ERCNTR:         0
1232
1233                          ;*****************************************************************
1234
1235                                  ;READ AND VERIFY ALL SECTORS WRITTEN BY
1236                                  ;PREVIOUS WRITE FUNCTION
1237
1238                                  ;   T = 4
1239
1240  006236  004737  010072  RDCHK:           JSR PC,GETPATTERN
1241  006242  004737  010710  XRDCHK:         JSR PC,INITTRACKS
1242  006246  004737  010374                  JSR PC,GETUNIT
```

```
1243  006252  004737  011020          1$:             JSR PC,GETTRACK
1244  006256  004737  005330                          JSR PC,READCHK
1245  006262  005337  011122                          DEC TRKCNTR
1246  006266  001371                                  BNE 1$
1247  006270  004737  010512                          JSR PC,STOP
1248  006274  000762                                  BR XRDCHK
1249
1250
1251                                  ;*****************************************************************
1252
1253                                          ;DO A READ ONLY FUNCTION ON ALL SECTORS
1254                                          ;THIS DOES NOT VERIFY THE DATA, ONLY TESTS FOR CRC ERRORS
1255                                          ;   T = 5
1256
1257  006276  004737  010072          RDONLY:         JSR PC,GETPATTERN
1258  006302  004737  010710          XRDONLY:        JSR PC,INITTRACKS
1259  006306  004737  010374                          JSR PC,GETUNIT
1260  006312  004737  011020          1$:             JSR PC,GETTRACK
1261  006316  004737  004022                          JSR PC,READ
1262  006322  005337  011122                          DEC TRKCNTR
1263  006326  001371                                  BNE 1$
1264  006330  004737  010512                          JSR PC,STOP
1265  006334  000762                                  BR XRDONLY
1266
1267
1268
1269                                  ;*****************************************************************
1270
1271                                          ;WRITE AND READ CHECK ANY DATA PATTERN,AND ANY TRACK SEQUENCE
1272                                          ;BUT ALTERNATE BETWEEN DRIVE UNITS ON EACH TRACK SELECTED.
1273                                          ;   T = 6
1274
1275  006336  005137  003744          DRVSWP:         COM RDAFTWT             ;SET THE READ AFTER WRITE FLAG
1276  006342  004737  010072                          JSR PC,GETPATTERN
1277  006346  004737  010710          2$:             JSR PC,INITTRACKS
1278  006352  004737  010374          1$:             JSR PC,GETUNIT          ;SET UP FOR UNIT 0
1279  006356  004737  011020                          JSR PC,GETTRACK         ;GET THE TRACK TO BE ACCESSED
1280  006362  004737  002744                          JSR PC,WRITE
1281  006366  004737  005330                          JSR PC,READCHK
1282  006372  004737  010374                          JSR PC,GETUNIT          ;NOW GO TO UNIT 1 ON THE SAME TRACK
1283  006376  004737  002744                          JSR PC,WRITE
1284  006402  004737  005330                          JSR PC,READCHK
1285  006406  005337  011122                          DEC TRKCNTR
1286  006412  001357                                  BNE 1$
1287  006414  004737  010512                          JSR PC,STOP
1288  006420  000752                                  BR 2$
1289
1290
1291                                  ;*****************************************************************
1292                                          ;THE FIRST HALF OF TEST 7 FORCES THE TRACK SEQUENCE
1293                                          ;TO INCRIMENT UP THROUGH ALL THE TRACKS DOING WRITE / READ CHECK FUNCTIONS.
1294                                          ;THIS VERIFIES THAT ALL THE TRACKS ARE ACCESSABLE.
1295                                          ;THE SECOND HALF OF THE PASS WILL USE THE SEQUENCE SELECTED BY THE
1296                                          ;OPERATOR AS INDICATED BELOW,AND ONLY READ CHECK THE DATA JUST WRITTEN.
```

```
1297                                    ;THIS VERIFIES THAT THE DATA CAN BE READ FROM A TRACK,AFTER THE HEAD HAS BEEN
1298                                    ;MOVED AWAY FROM AND BACK TO THAT TRACK. AT THE COMPLETION OF THE PASS
1299                                    ;THE DELETED DATA BIT IN TEST CONDITIONS IS COMPLEMENTED AND THE NEXT
1300                                    ;PASS WILL BE RUN UNDER THIS NEW CONDITION.
1301                                    ;   T = 0 OR 7
1302
1303  006422  004737  010072          TEST7:      JSR PC,GETPATTERN
1304  006426  013737  011134  006604              MOV SEQUEN,SEQSAV       ;SAVE THE SELECTED SEQUENCE
1305  006434  012737  000001  011134  TEST7X:     MOV #1,SEQUEN           ;FORCE INCREMENT SEQUENCE
1306  006442  005137  003744                      COM RDAFTWT             ;SET READ AFTER WRITE FLAG FOR ERROR TESTING
1307  006446  004737  010710                      JSR PC,INITTRACKS
1308  006452  004737  010374                      JSR PC,GETUNIT
1309  006456  004737  011020          1$:         JSR PC,GETTRACK
1310  006462  004737  002744                      JSR PC,WRITE
1311  006466  004737  005330                      JSR PC,READCHK
1312  006472  005337  011122                      DEC TRKCNTR             ;IS THE FIRST HALF OF THE PASS DONE
1313  006476  001367                              BNE 1$                  ;NO,CONTINUE WRITTING AND READING
1314  006500  005037  003744                      CLR RDAFTWT             ;YES,NOW DO A READ ONLY
1315  006504  013737  006604  011134              MOV SEQSAV,SEQUEN       ;USING SELECTED SEQUENCE
1316  006512  004737  010710                      JSR PC,INITTRACKS
1317  006516  004737  011020          2$:         JSR PC,GETTRACK
1318  006522  004737  005330                      JSR PC,READCHK
1319  006526  005337  011122                      DEC TRKCNTR             ;IS THIS HALF OF THE PASS DONE
1320  006532  001371                              BNE 2$                  ;NO,CONTINUE READ CHECKING
1321  006534  004737  010512                      JSR PC,STOP             ;YES,TEST FOR END OF PASS CONDITIONS
1322  006540  032737  040100  010464              BIT #40100,UNITSEL      ;HAVE ALL SELECTED UNITS BEEN USED
1323  006546  001332                              BNE TEST7X              ;IF A USED BIT IS STILL SET GO TO NEXT DRIVE
1324  006550  012706  001200                      MOV #1200,SP            ;RESET STACK ADDRESS FOR ACT11 EOP
1325  006554  032737  001000  001212              BIT #BIT9,DTESTP        ;NEXT PASS,COMPLEMENT D D BIT
1326  006562  001404                              BEQ 3$
1327  006564  042737  001000  001212              BIC #BIT9,DTESTP        ;IT WAS ON CLEAR IT
1328  006572  000720                              BR TEST7X
1329  006574  052737  001000  001212  3$:         BIS #BIT9,DTESTP        ;IT WAS OFF SET IT
1330  006602  000714                              BR TEST7X
1331
1332  006604  000000                    SEQSAV:   0
1333                                    ;*****************************************************************
1334
1335                                          ;TEST FOR TRANSFER FLAG AND PROGRAM NOT HUNG
1336
1337  006606  005037  006772          TRCK:       CLR HCNTR1              ;CLEAR HUNG COUNTER
1338  006612  012746  000340                      MOV #PR7,-(SP)          ;RAISE INTERUPT LEVEL TO INHIBIT INTERUPTS
1339  006616  012746  006624                      MOV #3$,-(SP)
1340  006622  000002                              RTI
1341  006624  105777  172356          3$:         TSTB @RXCS              ;TEST FOR TR FLAG
1342  006630  100001                              BPL 1$                  ;SKIP NEXT INST.IF NOT SET
1343  006632  000207                              RTS PC                  ;RETURN TO TRANSFER DATA
1344  006634  023727  006226  000200  1$:         CMP BYTECNTR,#128.      ;HAVE 128 BYTES BEEN TRANSFERED
1345  006642  001403                              BEQ 2$                  ;ALLOW AN INTERUPR IF EQUAL
1346  006644  005237  006772                      INC HCNTR1              ;IF THE HUNG COUNTER OVERFLOWES EITHER
1347                                                                      ;THERE WAS NO TR FLAG OR NO INTERUPT OCCURED
1348                                                                      ;AFTER 128 BYTES TRANSFERED
1349  006650  001365                              BNE 3$                  ;NO OVERFLOW RETEST FOR TR FLAG
1350  006652  132777  000040  172326  2$:         BITB #DONEBIT,@RXCS     ;TEST FOR A DONE FLAG WHICH WOULD INDICATE A
```

```
1351                                                            ;POSSABLE ERROR CONDITION EXISTS WHILE
1352                                                            ;TRANSFERING COMMANDS OR DATA
1353  006660  001407                        BEQ 4$              ;IF DONE NOT SET,TYPE HUNG
1354  006662  005037  006226                CLR BYTECNTR        ;CLEAR BYTE COUNTER SO IT  WON'T INTERFERE
1355                                                            ;WITH TESTING FOR DONE.
1356  006666  005726                        TST (SP)+           ;REMOVE THIS RETURN ADDRESS FROM STACK
1357  006670  005046                        CLR -(SP)           ;LOWER INTERUPT LEVEL TO ACCEPT AN INTERUPT
1358  006672  012746  006700                MOV #4$,-(SP)       ;THE INTERJPT WILL RETURN THROUGH PREVIOUSLY
1359  006676  000002                        RTI                 ;SET ADDRESSES
1360  006700  005237  006772      4$:       INC HCNTR1          ;ALLOW SOME TIME FOR FLAG TO SET
1361  006704  001362                        BNE 2$
1362  006706  104400  014364                TYPE ,MHUNG
1363  006712  000000              HLT21:    HALT                ;PROGRAM HUNG, SP (177706) CONTAINS RETURN
1364                                                            ;ADDRESS OF WAITING LOOP
1365
1366                                ;TEST FOR DONE FLAG AND PROGRAM HUNG
1367  006714  012603              DONECK:   MOV (SP)+,R3        ;TAKE THE RETURN ADDRESS OF THE STACK INCASE
1368                                                            ;THE PROGRAM IS EXPECTING AN INTERUPT AT DONE
1369  006716  005037  006772                CLR HCNTR1          ;CLEAR THE LOWER HUNG COUNTER
1370  006722  012737  177740  006774        MOV #177740,HCNTR2  ;SET UPER COUNTER FOR 31 FULL LOOPS
1371  006730  032777  000040  172250  2$:    BIT #DONEBIT,@RXCS  ;TEST FOR DONE FLAG
1372  006736  001402                        BEQ 1$              ;IF NOT SET SKIP RETURN
1373  006740  010346                        MOV R3,-(SP)        ;PUT RETURN ADDRESS BACK ON THE STACK
1374  006742  000207                        RTS PC              ;RETURN
1375  006744  062737  000001  006772  1$:    ADD #1,HCNTR1       ;WHEN THE DOUBLE WORD COUNTER OVERFLOWES
1376  006752  005537  006774                ADC HCNTR2          ;THE PROGRAM MUST BE HUNG
1377  006756  005737  006774                TST HCNTR2
1378  006762  001362                        BNE 2$              ;IF NO OVERFLOW RETEST FOR DONE
1379  006764  104400  014364                TYPE ,MHUNG
1380  006770  000000              HLT20:    HALT                ;PROGRAM HUNG,CONTENTS OF R3 (177703) HOLDES
1381                                                            ;ADDRESS OF WAITING TEST.
1382
1383  006772  000000              HCNTR1:   0
1384  006774  177740              HCNTR2:   177740
1385
1386                                ;AN INTERUPT DID NOT OCCURE AT A FUNCTION DONE FLAG.
1387
1388  006776  005237  015574      NOINTER:  INC INTER           ;INC INTERUPT ERROR COUNTER
1389  007002  032777  020000  172204        BIT #SW13,@SWR      ;TEST DON'T PRINT ERROR SWITCH
1390  007010  001004                        BNE 1$
1391  007012  104400  014212                TYPE ,MINTER        ;TYPE NO INTERUPT ON DONE ERROR
1392  007016  104400  014004                TYPE ,MCRLF
1393  007022  005777  172166      1$:       TST @SWR            ;TEST HALT ON ERROR SWITCH
1394  007026  100001                        BPL CONT21
1395  007030  000000              HLT15:    HALT                ;HALT ON ERROR
1396  007032  004737  007036      CONT21:   JSR PC,INTSERV      ;JSR TO INTSERV AS IF IT WAS AN INTERUPT
1397
1398                                ;*****************************************************************
1399
1400                                ;INTERUPT SERVICE
1401
1402  007036  117737  172146  007650  INTSERV:  MOVB @RXDB,ASTAT    ;SAVE THE ERROR AND STATUS WORD
1403  007044  005777  172136                TST @RXCS           ;TEST THE ERROR FLAG
1404  007050  100446                        BMI RXERROR         ;THERE WAS AN ERROR GO REPORT IT
```

```
1405  007052  032737  000004  007650          BIT #BIT2,ASTAT         ;IS INIT DONE SET
1406  007060  001402                          BEQ 2$                  ;NO,CONTINUE
1407  007062  000137  007462                  JMP RXPWR               ;YES,REPORT POWER FAILED AND RESTART
1408  007066  032737  000003  007650  2$:     BIT #3,ASTAT            ;ARE PAR OR CRC BITS SET
1409  007074  001022                          BNE 1$                  ;YES GO LOG ERROR
1410  007076  132777  000040  172102          BITB #DONEBIT,@RXCS      ;IS DONE SET
1411  007104  001013                          BNE 3$                  ;IF SET RETURN TO TEST
1412  007106  005237  015572                  INC UKNINT              ;INC UNKNOWN INTERUPT ERROR LOG
1413  007112  032777  020000  172074          BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
1414  007120  001004                          BNE 4$                  ;DON'T PRINT
1415  007122  104400  014244                  TYPE ,MUKNINT           ;TYPE UNKNOWN INTERUPT
1416  007126  104400  014004                  TYPE ,MCRLF
1417  007132  000002          4$:             RTI                     ;RETURN FROM THE INTERUPT
1418  007134  062706  000006          3$:     ADD #6,SP               ;BYPASS INTERUPT POINTERS ON STACK
1419  007140  000207                          RTS PC                  ;RETURN TO PROGRAM
1420  007142  005237  015570          1$:     INC NOERLOG             ;NO STATUS ERROR FLAG ERROR
1421  007146  032777  020000  172040          BIT #SW13,@SWR          ;TEST DON'T PRINT ERROR SWITCH
1422  007154  001004                          BNE RXERROR
1423  007156  104400  014766                  TYPE ,MNOFLAG           ;TYPE NO STATUS ERROR ERROR
1424  007162  104400  014004                  TYPE ,MCRLF
1425
1426  007166  052737  000400  010464  RXERROR:  BIS #BIT8,UNITSEL     ;SET HAD ERROR FLAG
1427  007174  012737  000012  015554          MOV #10.,RETRY          ;SET RETRY COUNTER
1428  007202  012777  000017  171776  2$:     MOV #RDER.,@RXCS        ;GET THE ERROR CODE
1429  007210  004737  006714                  JSR PC,DONECK           ;TEST FOR DONE FLAG
1430  007214  032777  000002  171766          BIT #2,@RXDB            ;WAS THERE A PARITY ERROR
1431  007222  001403                          BEQ 1$                  ;NO,CONTINUE
1432  007224  004737  001760                  JSR P  STATER           ;YES,GO REPORT THE PARITY ERROR
1433  007230  000764                          BR 2$                   ;REISSUE THE FUNCTION
1434  007232  117737  171752  007652  1$:     MOVB @RXDB,BSTAT        ;SAVE THE ERROR CODE IN B STATUS
1435  007240  005737  007652                  TST BSTAT               ;IS THERE A DEFINITE CODE
1436  007244  001407                          BEQ NOPRNT              ;NO   CONTINUE
1437  007246  013705  007652                  MOV BSTAT,R5            ;ADJUST ERROR CODE TO PRODUCE AN EVEN ADDR
1438  007252  006005                          ROR R5                  ;OF THE CORRESPONDING COUNTER
1439  007254  006005                          ROR R5
1440  007256  062705  015670                  ADD #ERCODE-2,R5        ;ADD ON ADDR OF ERROR LOG -2,AS THERE IS NO 0
1441                                                                  ;ERROR CODE. THE CONTENTS OF R5 WILL READJUST
1442                                                                  ;ADDRESS TO CORRECT LOG.
1443  007262  005215                          INC (R5)                ;INC THE CORRECT ERROR CODE COUNTER
1444  007264  032777  020000  171722  NOPRNT:  BIT #SW13,@SWR         ;TEST PRINT ERROR SWITCH IN SWR
1445  007272  001032                          BNE 2$
1446  007274  104400  014004                  TYPE ,MCRLF
1447  007300  104400  014015                  TYPE ,MERRHEADER         ;TYPE ERROR HEADER
1448  007304  104400  015074                  TYPE ,MPASS             ;TYPE PASSES COMPLETED AT ERROR
1449  007310  013737  017602  007322          MOV PASCNTR,1$
1450  007316  004537  013374                  JSR R5,SGLDEC
1451  007322  000000          1$:             OPEN
1452  007324  104400  014004                  TYPE ,MCRLF
1453  007330  104400  014121                  TYPE ,MRXCS              ;TYPE COMMAND STATUS REGISTER
1454  007334  013746  003742          MOV     FUNCTION,-(SP)  ;;SAVE FUNCTION FOR TYPEOUT
1455  007340  104402                  TYPOS                   ;;GO TYPE--OCTAL ASCII
1456  007342     003                  .BYTE   3               ;;TYPE 3 DIGIT(S)
1457  007343     000                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
1458  007344  004737  007476                  JSR PC,TYPADR           ;TYPE ADDRESSES AND RUN CONDITIONS
```

```
1459  007350  104400  014004                        TYPE ,MCRLF
1460  007354  004737  010030                        JSR PC,TYPCODE          ;PRINT THE STATUS REGISTERS
1461  007360  032737  000020  003742  2$:           BIT #BIT4,FUNCTION      ;WHAT DRIVE IS BEING USED
1462  007366  001006                                BNE 6$
1463  007370  012777  000013  171610                MOV #RDOSTAT,@RXCS      ;DRIVE 0 BEING USED
1464  007376  004737  006714        5$:             JSR PC,DONECK           ;TEST FOR DONE FLAG
1465  007402  000404                                BR 7$
1466  007404  012777  000033  171574  6$:           MOV #RD1STAT,@RXCS      ;DRIVE 1 BEING USED
1467  007412  000771                                BR 5$
1468  007414  032777  000002  171566  7$:           BIT #2,@RXDB            ;WAS THERE A PARITY ERROR
1469  007422  001403                                BEQ 3$                  ;NO CONTINUE
1470  007424  004737  001760                        JSR PC,STATER           ;YES REPORT THE ERROR
1471  007430  000753                                BR 2$                   ;REISSUE THE COMMAND
1472  007432  105777  171552        3$:             TSTB @RXDB              ;WAS DRIVE READY SET
1473  007436  100406                                BMI 4$                  ;YES RETURN
1474  007440  104400  013750                        TYPE ,MNODRY
1475  007444  104400  014004                        TYPE ,MCRLF
1476  007450  004737  007654                        JSR PC,DELUNIT          ;NO GO DELETE THAT UNIT
1477  007454  062706  000004        4$:             ADD #4,SP               ;MOVE ERROR RETURN TO TOP OF STACK
1478  007460  000207                                RTS PC
1479
1480  007462  104400  015276        RXPWR:          TYPE ,MRX11             ;ONLY THE RX11 POWER HAS FAILED
1481  007466  104400  013070                        TYPE ,$POWER            ;PRINT POWER FAILED
1482  007472  000137  002532                        JMP RESTART             ;GO TO RESTART
1483
1484
1485  007476  104400  013507        TYPADR:         TYPE ,MTRK              ;TYPE TRACK ADDRESS
1486  007502  013737  011124  007514                MOV TARGET,3$
1487  007510  004537  013374                        JSR R5,SGLOEC
1488  007514  000000        3$:                     OPEN
1489  007516  104400  013536                        TYPE ,MSECT             ;TYPE SECTOR ADDRESS
1490  007522  013737  012054  007542                MOV TSECTOR,2$
1491  007530  042737  177740  007542                BIC #177740,2$          ;CLEAR ALL BUT SECTOR ADDRESS
1492  007536  004537  013374                        JSR R5,SGLDEC
1493  007542  000000        2$:                     OPEN
1494  007544  104400  014001                        TYPE ,DBLSP
1495  007550  032737  000020  010464                BIT #BIT4,UNITSEL       ;WHITCH DRIVE IS BEING USED
1496  007556  001003                                BNE 1$
1497  007560  104400  014040                        TYPE ,MUNIT0            ;TYPE UNIT 0
1498  007564  000402                                BR TYPSEL
1499  007566  104400  014050        1$:             TYPE ,MUNIT1            ;TYPE UNIT 1
1500  007572  104400  014176        TYPSEL:         TYPE ,MPAT              ;TYPE PATTERN CODE
1501  007576  013746  010200                MOV     PAT,-(SP)       ;;SAVE PAT FOR TYPEOUT
1502  007602  104402                        TYPOS                   ;;GO TYPE--OCTAL ASCII
1503  007604     001                        .BYTE   1               ;;TYPE 1 DIGIT(S)
1504  007605     000                        .BYTE   0               ;;SUPPRESS LEADING ZEROS
1505  007606  104400  014001                        TYPE ,DBLSP
1506  007612  104400  014202                        TYPE ,MTEST             ;TYPE TEST CODE
1507  007616  013746  002052                MOV     TEST,-(SP)      ;;SAVE TEST FOR TYPEOUT
1508  007622  104402                        TYPOS                   ;;GO TYPE--OCTAL ASCII
1509  007624     001                        .BYTE   1               ;;TYPE 1 DIGIT(S)
1510  007625     000                        .BYTE   0               ;;SUPPRESS LEADING ZEROS
1511  007626  104400  014001                        TYPE ,DBLSP
1512  007632  104400  014206                        TYPE ,MSEQ              ;TYPE SEQUENCE CODE
```

```
1513  007636  013746  011134                  MOV     SEQUEN,-(SP)    ;;SAVE SEQUEN FOR TYPEOUT
1514  007642  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1515  007644     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
1516  007645     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
1517  007646  000207                          RTS PC
1518
1519  007650  000000              ASTAT:      0
1520  007652  000000              BSTAT:      0
1521
1522  007654  132737  000020  003742  DELUNIT:    BITB #BIT4,FUNCTION     ;TEST FOR PRESENTLY USED UNIT
1523  007662  001016                          BNE 3$
1524  007664  104400  014040                  TYPE ,MUNIT0            ;UNIT 0 HAS BEEN DELETED FROM USE
1525  007670  104400  014060                  TYPE ,MDELET
1526  007674  104400  014172                  TYPE ,DBLLF
1527  007700  042737  000200  010464          BIC #BIT7,UNITSEL       ;CLEAR UNIT 0 SELECTION BIT
1528  007706  005737  010464                  TST UNITSEL             ;WAS UNIT 1 SELECTED FOR USE
1529  007712  100020                          BPL 1$                  ;UNIT 1 NOT SELECTED GO DUMP ERROR REPORT
1530  007714  000137  002560          2$:     JMP TESTSEL             ;CONTINUE ON OTHER UNIT AT BEGINING OF TEST
1531  007720  104400  014050          3$:     TYPE ,MUNIT1            ;UNIT 1 HAS BEEN DELETED FROM USE
1532  007724  104400  014060                  TYPE,MDELET
1533  007730  104400  014172                  TYPE ,DBLLF
1534  007734  042737  100000  010464          BIC #BIT15,UNITSEL      ;CLEAR UNIT 1 SELECTION BIT
1535  007742  105737  010464                  TSTB UNITSEL            ;WAS UNIT 0 SELECTED FOR USE
1536  007746  100762                          BMI 2$                  ;YES CONTINUE ON UNIT 0
1537  007750  005137  021254                  COM SHORTRPT            ;SET SHORT REPORT FLAG
1538  007754  000137  017622          1$:     JMP ERDUMP              ;CAN'T CONTINUE TYPE OUT STATISTICAL REPORT
1539
1540
1541  007760  117737  171224  007650  RDCODE:     MOVB @RXDB,ASTAT        ;SAVE THE A STATUS
1542  007766  012777  000017  171212  2$:         MOV #RDER,@RXCS         ;READ THE B STATUS REGISTER
1543  007774  004737  006714                  JSR PC,DONECK           ;WAIT FOR DONE FLAG
1544  010000  032777  000002  171202          BIT #2,@RXDB            ;WAS THERE A PARITY ERROR
1545  010006  001403                          BEQ 1$                  ;NO,CONTINUE
1546  010010  004737  001760                  JSR PC,STATER           ;YES,REPORT THE PARITY ERROR
1547  010014  000764                          BR 2$                   ;RETRY READING STATUS B
1548  010016  117737  171166  007652  1$:     MOVB @RXDB,BSTAT        ;SAVE THE B STATUS CODES
1549  010024  104400  014004                  TYPE ,MCRLF
1550  010030  104400  014142          TYPCODE:    TYPE ,MASTAT            ;TYPE THE CONTENTS OF THE TWO STATUS REGISTERS
1551  010034  013746  007650                  MOV     ASTAT,-(SP)     ;;SAVE ASTAT FOR TYPEOUT
1552  010040  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1553  010042     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
1554  010043     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
1555  010044  104400  013772                  TYPE ,TAB
1556  010050  104400  014156                  TYPE ,MBSTAT
1557  010054  013746  007652                  MOV     BSTAT,-(SP)     ;;SAVE BSTAT FOR TYPEOUT
1558  010060  104402                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1559  010062     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
1560  010063     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
1561  010064  104400  014004                  TYPE ,MCRLF
1562  010070  000207                          RTS PC
1563
1564
1565                          ;*****************************************************************
1566
```

```
                                        ;DECODES THE DATA PATERN SELECTED IN THE SWR
                                        ;
                                        ;BITS 6,7 AND 8 OF THE INITIAL SWR SELECTED THE DATA PATTERN
                                        ;TO BE USED AS FOLLOWS:
                                        ;
                                        ;   BITS                DATA PATTERN
                                        ; 8  7  6
                                        ; 0  0  0               NO PATTERN SPECIFIED (FORCE RANDOM DATA)
                                        ; 0  0  1               ALL ZEROS
                                        ; 0  1  0               ALL ONES
                                        ; 0  1  1               FLOATING ZERO
                                        ; 1  0  0               FLOATING ONE
                                        ; 1  0  1               ALTERNATING BITS
                                        ; 1  1  0               ALTERNATING PAIRS OF BITS
                                        ; 1  1  1               RANDOM
                                        ;
                                        ;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 LAST
                                        ;TWO BYTES CONTAIN THE CHECK SUM NUMBERS.
                                    ;*******************************************************************

 010072  142737  000377  010140  GETPATTERN:   BICB #377,@#BRONPAT     ;CLEAR BRANCH OFFSET
 010100  005037  010372              CLR SUM                 ;SET UP FOR ACCUMULATION OF CHECK SUM
 010104  005737  010200              TST PAT                 ;IF NO PATTERN SPECIFIED FORCE PATTERN 7
 010110  001003                      BNE 1$
 010112  012737  000007  010200      MOV #7,PAT
 010120  013704  010200      1$:     MOV PAT,R4              ;GET PATTERN BITS
 010124  005304                      DEC R4                  ;ADJUST FOR CORRECT OFFSET
 010126  006304                      ASL R4
 010130  150437  010140              BISB R4,@#BRONPAT       ;INSERT OFFSET
 010134  012704  015346              MOV #BUFADR+2,R4        ;SET UP ADDRESS OF FIRST BYTE
 010140  000777          BRONPAT:    BR .                    ;BRANCH BY OFFSET SELECTED
 010142  000137  010202              JMP DATA0               ;000 DATA BYTE
 010146  000137  010214              JMP DATA1               ;377 DATA BYTE
 010152  000137  010224              JMP FLOAT0              ;FLOAT A 0 THROUGH ALL 1'S
 010156  000137  010266              JMP FLOAT1              ;FLOAT A 1 THROUGH ALL 0'S
 010162  000137  010274              JMP PAT125              ;125/052 DATA WORD
 010166  000137  010314              JMP PAT314              ;314/063 DATA WORD
 010172  000137  010324              JMP RANDATA             ;RANDOM DATA BYTE

 010176  000000          DATABYTE:   0
 010200  000000          PAT:        0

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

                                            ;LOAD SOFTWARE BUFFER WITH ALL ZEROS
                                            ;   P = 1

 010202  005037  010176      DATA0:          CLR DATABYTE
```

```
1621  010206  004737  010344          PATGEN:         JSR PC,LOAD             ;GO LOAD THE DATA BUFFER
1622  010212  000775                                  BR PATGEN
1623
1624                                  ;*****************************************************************
1625
1626                                          ;LOAD SOFTWARE BUFFER WITH ALL ONES
1627                                          ;   P = 2
1628
1629
1630  010214  112737  000377  010176  DATA1:          MOVB #377,DATABYTE
1631  010222  000771                                  BR PATGEN
1632
1633                                  ;*****************************************************************
1634
1635                                          ;FLOAT A 0 THROUGH ONES IN SOFTWARE BUFFER
1636                                          ;   P = 3
1637
1638
1639  010224  112737  000376  010176  FLOAT0:         MOVB #376,DATABYTE      ;SET UP A ONES FIELD
1640  010232  000261          XPATGEN:        SEC                             ;SET THE C BIT TO ROTATE THROUGH THE DATA
1641  010234  012702  000000          1$:             MOV #0,R2               ;CLR R2 (CAN'T USE "CLR" AS IT CLEARS "C" BIT)
1642  010240  103001                                  BCC 2$                  ;BR IF THE "C" BIT IS CLEARED
1643  010242  005202                                  INC R2                  ;SET R2 IF NOT
1644  010244  004737  010344          2$:             JSR PC,LOAD             ;GO LOAD THE DATA BUFFER
1645  010250  000241                                  CLC
1646  010252  005702                                  TST R2                  ;IS R2 NONZERO
1647  010254  001401                                  BEQ 3$
1648  010256  000261                                  SEC                     ;YES, SET THE "C" BIT
1649  010260  106137  010176          3$:             ROLB DATABYTE
1650  010264  000763                                  BR 1$
1651
1652                                  ;*****************************************************************
1653
1654                                          ;FLOAT A 1 THROUGH ALL ZEROS IN SOFTWARE BUFFER
1655                                          ;   P = 4
1656
1657
1658  010266  005037  010176          FLOAT1:         CLR DATABYTE
1659  010272  000757                                  BR XPATGEN
1660
1661                                  ;*****************************************************************
1662
1663                                          ;LOAD SOFTWARE BUFFER WITH ALTERNATING 1 AND 0 FOR
1664                                          ;ONE BYTE AND THE COMPLIMENT INTO THE NEXT
1665                                          ;   P = 5
1666
1667
1668  010274  112737  000125  010176  PAT125:         MOVB #125,DATABYTE
1669  010302  004737  010344          XXPATGEN:       JSR PC,LOAD
1670  010306  105137  010176                          COMB DATABYTE
1671  010312  000773                                  BR XXPATGEN
1672
1673                                  ;*****************************************************************
1674
```

```
1675                                            ;LOAD SOFTWARE BUFFER WITH ALTERNATING PAIRS OF 1 AND 0 AND
1676                                            ;COMPLIMENT INTO THE NEXT
1677                                            ;   P = 6
1678
1679  010314  112737  000314  010176  PAT314:         MOVB #314,DATABYTE
1680  010322  000767                                  BR XXPATGEN
1681
1682                                    ;*****************************************************************
1683
1684                                            ;LOAD SOFTWARE BUFFER WITH RANDOM DATA PATTERN
1685                                            ;   P = 0 OR 7
1686
1687  010324  004737  011666          RANDATA:        JSR PC,RANGEN           ;GET RANDOM NUMBER
1688  010330  113737  011760  010176                  MOVB RANDM,DATABYTE
1689  010336  004737  010344                          JSR PC,LOAD
1690  010342  000770                                  BR RANDATA
1691
1692  010344  063737  010176  010372  LOAD:           ADD DATABYTE,SUM        ;ACCUMULATE THE PATTERN CHECK SUM
1693  010352  113724  010176                          MOVB DATABYTE,(R4)+     ;LOAD THE DATA BUFFER
1694  010356  022704  015542                          CMP #BUFADR+176,R4      ;HAVE 124 BYTES BEEN GENERATED
1695  010362  001401                                  BEQ 1$                  ;IF YES,RETURN TO TEST
1696  010364  000207                                  RTS PC                  ;IF NO,RETURN TO PATTERN GENERATOR
1697  010366  005726                  1$:             TST (SP)+               ;TAKE PATTERN RETURN ADDRESS OF STACK
1698  010370  000207                                  RTS PC                  ;RETURN TO TEST
1699
1700  010372  000000                  SUM:            0
1701
1702                                    ;*****************************************************************
1703
1704                                            ;TEST FOR SELECTED UNITS,DRIVE READY,AND USED CONDITIONS
1705                                            ;ALSO CONTAINS A "HAD ERROR" FLAG TO BE TESTED AT EOP.
1706                                            ;THE BITS IN UNITSEL ARE USED AS FOLLOWS
1707                                            ;
1708                                            ;BIT15  =UNIT 1 SELECTED VIA SWR
1709                                            ;BIT14  =UNIT 1 USED BIT
1710                                            ;BIT8   =THIS PASS HAD AN ERROR
1711                                            ;BIT7   =UNIT 0 SELECTED VIA SWR
1712                                            ;BIT6   =UNIT 0 USED BIT
1713                                            ;BIT4   =UNIT SELECTION FOR FUNCTION WORD
1714
1715                                    ;*****************************************************************
1716
1717  010374  032737  000100  010464  GETUNIT:        BIT #BIT6,UNITSEL       ;HAS UNIT 0 JUST USED
1718  010402  001012                                  BNE 1$                  ;UNIT 0 USED CHECK UNIT 1
1719  010404  105737  010464                          TSTB UNITSEL            ;HAS UNIT 0 SELECTED
1720  010410  100007                                  BPL 1$                  ;NO GO TO UNIT 1
1721  010412  042737  040020  010464                  BIC #40020,UNITSEL      ;CLEAR UNIT 1 USED BIT AND FUNCTION UNIT BIT
1722  010420  052737  000100  010464                  BIS #BIT6,UNITSEL       ;SET UNIT 0 USED BIT
1723  010426  000207                                  RTS PC
1724  010430  005737  010464          1$:             TST UNITSEL             ;HAS UNIT 1 SELECTED
1725  010434  100012                                  BPL 2$                  ;NO RETURN
1726  010436  032737  040000  010464                  BIT #BIT14,UNITSEL      ;HAS UNIT 1 BEEN USED
1727  010444  001006                                  BNE 2$                  ;YES RETURN
1728  010446  042737  000100  010464                  BIC #BIT6,UNITSEL       ;CLEAR UNIT 0 USED BIT
```

```
1729  010454  052737  040020  010464                  BIS #40020,UNITSEL      ;SET UNIT 1 USED BIT AND FUNCTION UNIT BIT
1730  010462  000207                  2$:             RTS PC
1731
1732
1733
1734  010464  000000                  UNITSEL:        0
1735
1736                                          ;TEST THAT ALL UNITS HAVE BEEN ACCESSED
1737
1738  010466  005737  010464          DONE:           TST UNITSEL             ;IS UNIT 1 SELECTED
1739  010472  100006                                  BPL 1$                  ;NO RETURN
1740  010474  032737  040000  010464                  BIT #BIT14,UNITSEL      ;YES HAS IT BEEN USED
1741  010502  001002                                  BNE 1$                  ;YES RETURN
1742  010504  062706  000002                          ADD #2,SP               ;BYPASS NOT DONE RETURE ON STACK
1743  010510  000207                  1$:             RTS PC
1744
1745
1746  010512  004737  010466          STOP:           JSR PC,DONE             ;HAVE ALL DRIVES BEEN USED
1747  010516  005237  017602                          INC PASCNTR             ;KEEP TOTAL NUMBER OF PASSES
1748  010522  005737  010706                          TST CHARCT              ;HAVE 72 EOP INDICATORS BEEN TYPED
1749  010526  001005                                  BNE 3$
1750  010530  104400  014004                          TYPE ,MCRLF             ;YES DO A CRLF
1751  010534  012737  177670  010706                  MOV #-72.,CHARCT        ;RESET CHARACTER COUNTER
1752  010542  005237  010706          3$:             INC CHARCT
1753  010546  032737  000400  010464                  BIT #BIT8,UNITSEL       ;TEST HAD ERROR FLAG
1754  010554  001403                                  BEQ 2$                  ;NO ERROR PRINT *
1755  010556  104400  014011                          TYPE ,MEREOP            ;HAD ERROR PRINT ?
1756  010562  000404                                  BR 1$
1757  010564  104400  014007          2$:             TYPE ,MEOP              ;PRINT EOP INDICATOR
1758  010570  104400  014013                          TYPE ,MBELL
1759  010574  032777  040000  170412  1$:             BIT #SW14,@SWR          ;TEST EOP HALT SWITCH
1760  010602  001417                                  BEQ CONT22
1761  010604  104400  014004                          TYPE ,MCRLF
1762  010610  104400  015074                          TYPE ,MPASS             ;AT HALT ON PASS PRINT NUMBER OF PASSES COMPLETE
1763  010614  013737  017602  010626                  MOV PASCNTR,6$
1764  010622  004537  013374                          JSR R5,SGLDEC
1765  010626  000000                  6$:             OPEN
1766  010630  104400  014004                          TYPE ,MCRLF
1767  010634  000000                  HLT16:          HALT
1768  010636  005037  010706                          CLR CHARCT              ;THIS CAUSES A CRLF AFTER HALT AT E.O.P.
1769  010642  042737  040500  010464  CONT22:         BIC #40500,UNITSEL      ;CLEAR USED BITS, AND "HAD ERROR" BIT
1770  010650  022737  000007  010200                  CMP #7,PAT              ;IF RANDOM DATA GET NEW DATA FOR BUFFER
1771  010656  001012                                  BNE HERE
1772  010660  004737  010072                          JSR PC,GETPATTERN
1773  010664  013705  000042          2$:             MOV @#42,R5             ;ACT11 END OF PASS HOOKS
1774  010670  001405                                  BEQ HERE
1775  010672  000005                                  RESET
1776  010674  004715                  LOGICAL:        JSR PC,(R5)
1777  010676  000240                                  NOP
1778  010700  000240                                  NOP
1779  010702  000240                                  NOP
1780  010704  000207                  HERE:           RTS PC
1781
1782  010706  000000                  CHARCT:         0
```

```
1783
1784                                    ;*****************************************************************
1785
1786                                            ;INITIALIZE TRACK SEQUENCE
1787
1788
1789  010710  042737  100200  001200  INITTRACK:      BIC #100200,OD          ;CLEAR FIRST USED BITS
1790  010716  005737  001200                          TST OD                  ;TEST CONTENTS OF ID,OD FOR 0
1791  010722  001005                                  BNE 1$                  ;ID,OD SPECIFIED USE THEM
1792  010724  112737  000114  001201                  MOVB #114,ID            ;NONE SPECIFIED SET ID TO MAXIMUM
1793  010732  105037  001200                          CLRB OD                 ;SET OD TO MINIMUM
1794  010736  113737  001200  011124  1$:             MOVB OD,TARGET          ;INIT OD AS PRESENT TRACK
1795  010744  005037  011132                          CLR XID                 ;INIT WORKING ID AND OD LOCATIONS
1796  010750  113737  001201  011132                  MOVB ID,XID
1797  010756  005037  011130                          CLR XOD
1798  010762  113737  001200  011130                  MOVB OD,XOD
1799  010770  013737  011132  011122                  MOV XID,TRKCNTR         ;SET UP NUMBER OF TRACK MOVEMENTS
1800  010776  163737  011130  011122                  SUB XOD,TRKCNTR
1801  011004  005237  011122                          INC TRKCNTR
1802  011010  052737  100200  001200                  BIS #100200,OD          ;SET FIRST TIME BITS IN ID,OD
1803  011016  000207                                  RTS PC
1804
1805
1806
1807                                    ;*****************************************************************
1808
1809                                            ;TEST FOR HEAD SEQUENCE SELECTED BY INITIAL SWR SETTING
1810                                            ;
1811                                            ;BITS 0,1,AND 2 OF THE INITIAL SWR SELECTED THE TRACK SEQUENCING
1812                                            ;TO BE FOLLOWED AS INITICATED BELOW
1813                                            ;
1814                                            ;   BITS                SEQUENCE
1815                                            ; 0  0  0               NO SEQUENCE SPECIFIED (DEFAULT TO SEQ 7)
1816                                            ; 0  0  1               INCREMENT FROM OD TO ID
1817                                            ; 0  1  0               DECREMENT FROM ID TO OD
1818                                            ; 0  1  1               DO PREVIOUS 2 SEQUENCES
1819                                            ; 1  0  0               BOUNCE BETWEEN ID AND OD
1820                                            ; 1  0  1               DECREASING BOUNCE
1821                                            ; 1  1  0               STROBE BETWEEN OD AND DECREMENTING ID
1822                                            ; 1  1  1               RANDOM TRACK SELECTION
1823
1824                                    ;*****************************************************************
1825
1826  011020  113737  011124  011126  GETTRACK:       MOVB TARGET,PRESTRK     ;RESET TO PRESENT TRACK
1827  011026  142737  000377  011064                  BICB #377,@#BRONTRK     ;CLEAR OUT BRANCH OFFSET
1828  011034  005737  011134                          TST SEQUEN              ;IF NO SEQUENCE SPECIFIED FORCE SEQUENCE 7
1829  011040  001003                                  BNE 1$
1830  011042  012737  000007  011134                  MOV #7,SEQUEN
1831  011050  013704  011134          1$:             MOV SEQUEN,R4           ;GET SEQUENCE BITS
1832  011054  005304                                  DEC R4                  ;ADJUST FOR CORRECT OFFSET
1833  011056  006304                                  ASL R4
1834  011060  150437  011064                          BISB R4,@#BRONTRK       ;THIS BR INST.IS MODIFIED BY THE TEST CONDITIONS
1835                                                                          ;SELECTED ,TO BRANCH TO ONE OF THE SEQ. BELOW
1836
```

```
1837  011064  000777                BRONTRK:        BR .                    ;BRANCH BY OFFSET SELECTED
1838  011066  000137  011136                       JMP SEQ1                ;INCREMENT
1839  011072  000137  011172                       JMP SEQ2                ;DECREMENT
1840  011076  000137  011224                       JMP SEQ3                ;INCREMENT/DECREMENT
1841  011102  000137  011270                       JMP SEQ4                ;BOUNCE ID TO OD
1842  011106  000137  011354                       JMP SEQ5                ;DECREASING BOUNCE
1843  011112  000137  011506                       JMP SEQ6                ;STROBE
1844  011116  000137  011566                       JMP SEQ7                ;RANDOM
1845
1846  011122  000000                TRKCNTR:        0
1847  011124  000000                TARGET:         0
1848  011126  000000                PRESTRK:        0
1849  011130  000000                XOD:            0
1850  011132  000000                XID:            0
1851  011134  000000                SEQUEN:         0
1852
1853                                ;*****************************************************************
1854
1855                                        ;INCREMENT FROM OD+1 TO ID AND RETURN TO OD
1856                                        ;   S = 1
1857
1858  011136  042737  100200  001200 SEQ1:           BIC #100200,OD          ;CLEAR FIRST TIME BITS
1859  011144  123737  011132  011126                 CMPB XID,PRESTRK        ;PRESENT TRACK EQUAL TO ID
1860  011152  001004                                 BNE 1$                  ;NO GET NEW TRACK
1861  011154  113737  001200  011124                 MOVB OD,TARGET          ;YES RETURN TO OD
1862  011162  000461                                 BR NEWTRK               ;NEWTRK INCREMENTS THE TRACK MOVED TO COUNTER
1863                                                                          ;AND RETURNS TO NEXT TRACK
1864  011164  005237  011124          1$:           INC TARGET              ;ADD 1 TO TARGET TRACK
1865  011170  000456                                 BR NEWTRK
1866
1867                                ;*****************************************************************
1868
1869                                        ;DECREMENT FROM ID TO OD
1870                                        ;   S = 2
1871
1872  011172  105737  001201          SEQ2:         TSTB ID                 ;FIRST TIME BIT SET
1873  011176  100007                                 BPL 1$                  ;NO GET NEXT TRACK
1874  011200  042737  100200  001200                 BIC #100200,OD          ;YES CLEAR FIRST TIME BITS
1875  011206  113737  011132  011124                 MOVB XID,TARGET         ;GO TO ID
1876  011214  000444                                 BR NEWTRK
1877  011216  005337  011124          1$:           DEC TARGET              ;MOVE TO NEXT TRACK
1878  011222  000441                                 BR NEWTRK
1879
1880                                ;*****************************************************************
1881
1882                                        ;INCREMENT THEN DECREMENT TRACKS
1883                                        ;   S = 3
1884
1885  011224  105737  001200          SEQ3:         TSTB OD                 ;CHECK FIRST TIME BIT
1886  011230  100007                                 BPL 1$                  ;NOT FIRST TIME THROUGH
1887  011232  042737  100200  001200                 BIC #100200,OD          ;CLEAR FIRST TIME BITS
1888  011240  005337  011122                        DEC TRKCNTR
1889  011244  006337  011122                        ASL TRKCNTR             ;RESET TRACK COUNTER TO DOUBLE THE COUNT
1890  011250  013700  011132          1$:           MOV XID,R0
```

```
1891  011254  163700  011130                    SUB XOD,R0              ;GET DIFFERENCE BETWEEN ID AND OD
1892  011260  163700  011122                    SUB TRKCNTR,R0          ;IS TRACK COUNTER HALF DONE
1893  011264  002342                            BGE SEQ2                ;YES GO DECREMENT TRACKS
1894  011266  000723                            BR SEQ1                 ;NO CONTINUE INCREMENTING TRACKS
1895
1896
1897                          ;*****************************************************  **************
1898
1899                                  ;BOUNCE BETWEEN ID AND OD ONLY
1900                                  ;   S = 4
1901
1902  011270  042737  100200  001200  SEQ4:          BIC #100200,OD          ;CLEAR THE FIRST TIME BITS
1903  011276  123737  011126  001200                CMPB PRESTRK,OD         ;DID IT JUST DO OD
1904  011304  001404                                BEQ IDNEXT              ;YES
1905  011306  113737  001200  011124                MOVB OD,TARGET          ;NO GO TO OD TRACK
1906  011314  000404                                BR NEWTRK
1907  011316  113737  001201  011124  IDNEXT:       MOVB ID,TARGET          ;GO DO ID TRACK
1908  011324  000400                                BR NEWTRK
1909
1910                          ;*****************************************************************
1911
1912                                  ;INCREMENT THE "HEAD MOVED TO" COUNTERS AND RETURN
1913                                  ;THROUGH HERE FROM ALL TRACK SEQUENCES
1914
1915  011326  013705  011124  NEWTRK:          MOV TARGET,R5           ;GET TRACK ADDRESS
1916  011332  006305                            ASL R5                  ;MULTIPLY BY 4 TO DOUBLE PRECISION
1917  011334  006305                            ASL R5                  ;INTERLEAVE COUNTER ADDRESSES
1918  011336  062705  016420                    ADD #HDMOVE,R5          ;ADD ON ADDRESS OF HEAD MOVE COUNTER
1919  011342  062715  000001                    ADD #1,(R5)             ;INC COUNTER
1920  011346  005565  000002                    ADC 2(R5)               ;ADD CARRY TO HIGH ORDER WORD
1921  011352  000207                            RTS PC                  ;RETURN TO ACCESS NEXT TRACK
1922
1923
1924                          ;*****************************************************************
1925
1926                                  ;BOUNCE BETWEEN DECREASING ID AND INCREASING OD
1927                                  ;   S = 5
1928
1929  011354  105737  001201  SEQ5:            TSTB ID                 ;TEST FIRST TIME BIT OF ID
1930  011360  100011                            BPL 1$
1931  011362  005237  011122                    INC TRKCNTR             ;ADJUST FOR ONE EXTRA MOVEMENT TO TRK 00
1932  011366  142737  000200  001201            BICB #200,ID            ;CLEAR FIRST TIME BIT OF ID
1933  011374  113737  011132  011124            MOVB XID,TARGET         ;MOVE TO ID
1934  011402  000751                            BR NEWTRK
1935  011404  105737  001200  1$:              TSTB OD                 ;TEST FIRST TIME BIT OF OD
1936  011410  100007                            BPL 2$
1937  011412  142737  000200  001200            BICB #200,OD            ;CLEAR FIRST TIME BIT OF OD
1938  011420  113737  011130  011124            MOVB XOD,TARGET         ;MOVE TO OD
1939  011426  000737                            BR NEWTRK
1940  011430  132737  000001  011122  2$:        BITB #1,TRKCNTR         ;TEST COUNTER FOR ODD OR EVEN
1941  011436  001006                            BNE 3$                  ;ODD MOVE TO NEW ID
1942  011440  005237  011130                    INC XOD                 ;EVEN ADD 1 TO OD
1943  011444  113737  011130  011124            MOVB XOD,TARGET
1944  011452  000405                            BR LASTTRK              ;SEE IF LAST TRACK IS BEING ACCESSED
```

```
1945  011454  005337  011132          3$:             DEC XID                 ;SUBTRACT 1 FROM ID
1946  011460  113737  011132  011124                  MOVB XID,TARGET
1947  011466  123727  011122  000001  LASTTRK:        CMPB TRKCNTR,#1
1948  011474  001003                                  BNE XRETURN
1949  011476  113737  001200  011124                  MOVB OD,TARGET          ;THIS IS LAST TRACK RETURN TO OD
1950  011504  000710                  XRETURN:        BR NEWTRK
1951
1952                                    ;*****************************************************************
1953
1954                                            ;STROBE BETWEEN OD AND DECREASING ID
1955                                            ;   S = 6
1956
1957  011506  105737  001200          SEQ6:           TSTB OD                 ;CHECK FIRST TIME BIT
1958  011512  100007                                  BPL 1$                  ;NOT FIRST TIME
1959  011514  042737  100200  001200                  BIC #100200,OD          ;WAS FIRST TIME,CLEAR THE BITS
1960  011522  005337  011122                          DEC TRKCNTR             ;RESET COUNTER TO DOUBLE THE NUMBER OF TRACKS
1961  011526  006337  011122                          ASL TRKCNTR
1962  011532  123737  011126  001200  1$:             CMPB PRESTRK,OD         ;WAS OD JUST ACCESSED
1963  011540  001006                                  BNE ODNEXT              ;NO GO TO OD
1964  011542  113737  011132  011124                  MOVB XID,TARGET         ;DECREASING ID IS NEXT TRACK
1965  011550  005337  011132                          DEC XID
1966  011554  000664                                  BR NEWTRK
1967  011556  113737  001200  011124  ODNEXT:        MOVB OD,TARGET          ;OD IS NEXT TRACK
1968  011564  000660                                  BR NEWTRK
1969
1970
1971
1972                                    ;*****************************************************************
1973                                            ;RANDOM SEQUENCING OF TRACKS
1974                                            ;FOR PROPER USE OF RANDOM TRACK SEQUENCING, THE OD/ID LIMITS
1975                                            ;SHOULD NOT BE SET FOR LESS THAN HALF THE TRACKS.
1976                                            ;   S = 0 OR 7
1977
1978  011566  042737  100200  001200  SEQ7:           BIC #100200,OD          ;CLEAR THE FIRST TIME BITS
1979  011574  004737  011666                          JSR PC,RANGEN           ;GET A RANDOM NUMBER
1980  011600  042737  177600  011760                  BIC #177600,RANUM       ;CLEAR ALL BUT LOW 7 BITS
1981  011606  123737  011760  011132  IDCOMP:        CMPB RANUM,XID          ;IS RANUM LARGER THAN ID ADDRESS
1982  011614  003404                                  BLE ODCOMP              ;NO,RANUM IS LESS OR EQUAL TO ID
1983  011616  163737  011132  011760                  SUB XID,RANUM           ;YES,SUBTRACT ID FROM IT
1984  011624  000770                                  BR IDCOMP               ;SEE IF RANUM IS NOW OK
1985  011626  123737  011760  011130  ODCOMP:        CMPB RANUM,XOD          ;IS RANUM SMALLER THAN OD ADDRESS
1986  011634  002004                                  BGE PRESCHK             ;NO,RANUM IS GREATER OR EQUAL TO OD
1987  011636  063737  011130  011760                  ADD XOD,RANUM           ;YES,ADD OD TO RANUM
1988  011644  000770                                  BR ODCOMP               ;SEE IF RANUM IS NOW OK
1989  011646  123737  011760  011126  PRESCHK:       CMPB RANUM,PRESTRK      ;IF RANUM EQUALS PRESENT TRACK
1990  011654  001744                                  BEQ SEQ7                ;GET ANOTHER RANDOM NUMBER
1991  011656  013737  011760  011124                  MOV RANUM,TARGET        ;RANUM OK PUT IT IN TARGET TRACK
1992  011664  000620                                  BR NEWTRK
1993
1994  011666  012700  000001          RANGEN:         MOV #1,R0
1995  011672  063700  011754                          ADD RAN1,R0
1996  011676  063700  011756                          ADD RAN2,R0
1997  011702  042700  170000                          BIC #170000,R0
1998  011706  000241                                  CLC
```

```
 1999  011710  006100                          ROL R0
 2000  011712  006100                          ROL R0
 2001  011714  010037  011754                  MOV R0,RAN1
 2002  011720  005000                          CLR R0
 2003  011722  013700  011756                  MOV RAN2,R0
 2004  011726  006000                          ROR R0
 2005  011730  006000                          ROR R0
 2006  011732  063700  011754                  ADD RAN1,R0
 2007  011736  042700  170000                  BIC #170000,R0
 2008  011742  010037  011756                  MOV R0,RAN2
 2009  011746  010037  011760                  MOV R0,RANUM
 2010  011752  000207                          RTS PC
 2011
 2012  011754  000000              RAN1:       0
 2013  011756  000000              RAN2:       0
 2014  011760  000000              RANUM:      0
 2015
 2016                              ;*****************************************************************
 2017
 2018                                      ;SECTOR INITIALIZATION AND SELECTION
 2019
 2020  011762  005737  001202      INITSECTOR: TST FIRST                   ;TEST FIRST AND LAST FOR 0
 2021  011766  001005                          BNE 1$                      ;SECTORS SPECIFIED USE THEM
 2022  011770  005237  001202                  INC FIRST                   ;NONE SPECIFIED SET FIRST TO 1
 2023  011774  112737  000032  001203        MOVB #32,LAST               ;SET LAST TO MAXIMUM
 2024  012002  113737  001203  012052  1$:     MOVB LAST,SECCNTR           ;SET UP SECTOR COUNTER
 2025  012010  163737  001202  012052          SUB FIRST,SECCNTR
 2026  012016  005237  012052                  INC SECCNTR
 2027  012022  105237  012053                  CLRB SECCNTR+1
 2028  012026  113737  001202  012054        MOVB FIRST,TSECTOR          ;PUT FIRST SECTOR IN TARGET SECTOR
 2029  012034  163737  012060  012054          SUB THREE,TSECTOR           ;SUB 3 FROM TSECTOR AS FIRST TIME THROUGH
 2030                                                                      ;IT GETS ADDED BACK ON.
 2031  012042  012737  000001  012056          MOV #1,INTLEAV              ;SET INTERLEAVE OFFSET
 2032  012050  000207                          RTS PC
 2033
 2034  012052  000000              SECCNTR:    0
 2035  012054  000000              TSECTOR:    0
 2036  012056  000000              INTLEAV:    0
 2037  012060  000003              THREE:      3
 2038
 2039  012062  063737  012060  012054  GETSECTOR: ADD THREE,TSECTOR        ;ADD 3 FOR INTERLEAVING
 2040  012070  123737  001203  012054          CMPB LAST,TSECTOR
 2041  012076  002010                          BGE 1$                      ;NEW SECTOR IS WITHIN LIMITS
 2042  012100  113737  001202  012054          MOVB FIRST,TSECTOR          ;RESET TARGET SECTOR TO INTERLEAVE
 2043  012106  063737  012056  012054          ADD INTLEAV,TSECTOR         ;ADD ON INTERLEAVE OFFSET VALUE
 2044  012114  005237  012056                  INC INTLEAV                 ;UP DATE THE OFFSET VALUE
 2045  012120  000207              1$:         RTS PC
 2046
 2047
 2048                              ;*****************************************************************
 2049
 2050                              .SBTTL  TYPE ROUTINE
 2051
 2052                              ;*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
                                        ;*
                                        ;
012122  105737  012345          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
012126  100002                          BPL     1$              ;;BR IF YES
012130  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
012132  000407                          BR      3$              ;;LEAVE
012134  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
012136  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
012142  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
012144  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
012146  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
012150  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
012152  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
012156  000002                          RTI                     ;;RETURN
012160  122716  000011          4$:     CMPB    #THT,(SP)       ;;BRANCH IF <HT>
012164  001426                          BEQ     8$
012166  122716  000200                  CMPB    #TCRLF,(SP)     ;;BRANCH IF NOT <CRLF>
012172  001004                          BNE     5$
012174  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
012176  104400                          TYPE                    ;;TYPE A CR AND LF
012200  012347                          $CRLF
012202  000757                          BR      2$              ;;GET NEXT CHARACTER
012204  004737  012266          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
012210  123726  012344          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
012214  001352                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
012216  013746  012342                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                        ;;AND THE NULL CHAR.
012222  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
012226  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
012230  004737  012266                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
012234  105337  012332                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
012240  000770                          BR      7$              ;;LOOP

                                        ;HORIZONTAL TAB PROCESSOR

012242  112716  000040          8$:     MOVB    #40,(SP)        ;;REPLACE TAB WITH SPACE
012246  004737  012266          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
012252  132737  000007  012332          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
012260  001372                          BNE     9$              ;;TAB STOP
012262  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
012264  000726                          BR      2$              ;;GET NEXT CHARACTER
012266  105777  000044          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
012272  100375                          BPL     $TYPEC
```

```
2107  012274  116677  000002  000036          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2108  012302  122766  000015  000002          CMPB    #15,2(SP)       ;;BRANCH IF
2109  012310  001003                          BNE     1$              ;;NOT <CR>
2110  012312  105037  012332                  CLRB    $CHARCNT        ;;
2111  012316  000406                          BR      $TYPEX          ;;EXIT
2112  012320  122766  000012  000002  1$:     CMPB    #12,2(SP)       ;;BRANCH IF
2113  012326  0C   02                        BGE     $TYPEX          ;;<LF>
2114  012330  1C   27                        INCB    (PC)+           ;;INC SPACE
2115  012332  00   00          $CHARCNT:.WORD  0                       ;;COUNT
2116  012334  000207           $TYPEX: RTS     PC
2117                            ;;      EQUATES
2118          000011            THT=11
2119          000200            TCRLF=200
2120
2121  012336  177564            $TPS:   .WORD   177564          ;;TTY PRINTER STATUS REG. ADDRESS
2122  012340  177566            $TPB:   .WORD   177566          ;;TTY PRINTER BUFFER REG. ADDRESS
2123  012342     000            $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
2124  012343     002            $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
2125  012344     012            $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
2126  012345     000            $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
2127  012346     077            $QUES:  .ASCII  "?"             ;;QUESTION MARK
2128  012347     015     000    $CRLF:  .ASCIZ  <15>            ;;CARRAIGE RETURN
2129  012351     012     000    $LF:    .ASCIZ  <12>            ;;LINEFEED
2130          012354                    .EVEN
2131                            ;*****************************************************************
2132
2133                            .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
2134
2135                            ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
2136                            ;*OCTAL (ASCII) NUMBER AND TYPE IT.
2137                            ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
2138                            ;*CALL:
2139                            ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2140                            ;*      TYPOS                   ;;CALL FOR TYPEOUT
2141                            ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
2142                            ;*      .BYTE   M               ;;M=1 OR 0
2143                            ;*                                      ;;1=TYPE LEADING ZEROS
2144                            ;*                                      ;;0=SUPPRESS LEADING ZEROS
2145                            ;*
2146                            ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
2147                            ;*$TYPOS OR $TYPOC
2148                            ;*CALL:
2149                            ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2150                            ;*      TYPON                   ;;CALL FOR TYPEOUT
2151                            ;*
2152                            ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
2153                            ;*CALL:
2154                            ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2155                            ;*      TYPOC                   ;;CALL FOR TYPEOUT
2156
2157  012354  017646  000000    $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
2158  012360  116637  000001  012577    MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
2159  012366  112637  012601            MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
2160  012372  062716  000002            ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
```

```
2161  012376  000406                          BR      $TYPON
2162  012400  112737  000001  012577  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
2163  012406  112737  000006  012601          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
2164  012414  112737  000005  012576  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
2165  012422  010346                          MOV     R3,-(SP)        ;;SAVE R3
2166  012424  010446                          MOV     R4,-(SP)        ;;SAVE R4
2167  012426  010546                          MOV     R5,-(SP)        ;;SAVE R5
2168  012430  113704  012601                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
2169  012434  005404                          NEG     R4
2170  012436  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
2171  012442  112437  012600                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
2172  012446  113704  012577                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
2173  012452  016505  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
2174  012456  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
2175  012460  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
2176  012462  000404                          BR      3$              ;;GO DO MSB
2177  012464  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
2178  012466  006105                          ROL     R5
2179  012470  006105                          ROL     R5
2180  012472  010503                          MOV     R5,R3
2181  012474  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
2182  012476  105337  012600                  DECB    $OMODE          ;;TYPE THIS DIGIT?
2183  012502  100016                          BPL     7$              ;;BR IF NO
2184  012504  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
2185  012510  001002                          BNE     4$              ;;TEST FOR 0
2186  012512  005704                          TST     R4              ;;SUPPRESS THIS 0?
2187  012514  001403                          BEQ     5$              ;;BR IF YES
2188  012516  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
2189  012520  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
2190  012524  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
2191  012530  110337  012574                  MOVB    R3,8$           ;;SAVE FOR TYPING
2192  012534  104400  012574                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
2193  012540  105337  012576          7$:     DECB    $OCNT           ;;COUNT BY 1
2194  012544  003347                          BGT     2$              ;;BR IF MORE TO DO
2195  012546  002402                          BLT     6$              ;;BR IF DONE
2196  012550  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
2197  012552  000744                          BR      2$              ;;GO DO THE LAST DIGIT
2198  012554  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
2199  012556  012604                          MOV     (SP)+,R4        ;;RESTORE R4
2200  012560  012603                          MOV     (SP)+,R3        ;;RESTORE R3
2201  012562  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
2202  012570  012616                          MOV     (SP)+,(SP)
2203  012572  000002                          RTI                     ;;RETURN
2204  012574     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
2205  012575     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
2206  012576     000          $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
2207  012577     000          $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
2208  012600  000000          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
2209                          ;*****************************************************************
2210
2211                          .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
2212
2213                          ;*SAVE R0-R5
2214                          ;*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

012602                                  $SAVREG:
012602  010046                                 MOV     R0,-(SP)        ;;PUSH R0 ON STACK
012604  010146                                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
012606  010246                                 MOV     R2,-(SP)        ;;PUSH R2 ON STACK
012610  010346                                 MOV     R3,-(SP)        ;;PUSH R3 ON STACK
012612  010446                                 MOV     R4,-(SP)        ;;PUSH R4 ON STACK
012614  010546                                 MOV     R5,-(SP)        ;;PUSH R5 ON STACK
012616  016646  000022                         MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
012622  016646  000022                         MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
012626  016646  000022                         MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
012632  016646  000022                         MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
012636  000002                                 RTI

                                        ;*RESTORE R0-R5
                                        ;*CALL:
                                        ;*      RESREG
012640                                  $RESREG:
012640  012666  000022                         MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
012644  012666  000022                         MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
012650  012666  000022                         MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
012654  012666  000022                         MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
012660  012605                                 MOV     (SP)+,R5        ;;POP STACK INTO R5
012662  012604                                 MOV     (SP)+,R4        ;;POP STACK INTO R4
012664  012603                                 MOV     (SP)+,R3        ;;POP STACK INTO R3
012666  012602                                 MOV     (SP)+,R2        ;;POP STACK INTO R2
012670  012601                                 MOV     (SP)+,R1        ;;POP STACK INTO R1
012672  012600                                 MOV     (SP)+,R0        ;;POP STACK INTO R0
012674  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.

012676  010046                          $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
012700  016600  000002                         MOV     2(SP),R0        ;;GET TRAP ADDRESS
012704  005740                                 TST     -(R0)           ;;BACKUP BY 2
012706  111000                                 MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
012710  006300                                 ASL     R0              ;;POSITION FOR INDEXING
```

```
                012712  016000  012720                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
                012716  000200                          RTS     R0              ;;GO TO ROUTINE


                                        .SBTTL  TRAP TABLE

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

                                        ;       ROUTINE
                                        ;       -------
                012720                  $TRPAD:
                012720  012122                  $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
                012722  012400                  $TYPOC  ;;CALL=TYPOC    TRAP+1(104401)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
                012724  012354                  $TYPOS  ;;CALL=TYPOS    TRAP+2(104402)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
                012726  012414                  $TYPON  ;;CALL=TYPON    TRAP+3(104403)  TYPE OCTAL NUMBER (AS PER LAST CALL)
                012730  012602                  $SAVREG ;;CALL=SAVREG   TRAP+4(104404)  SAVE R0-R5 ROUTINE
                012732  012640                  $RESREG ;;CALL=RESREG   TRAP+5(104405)  RESTORE R0-R5 ROUTINE
                                        ;*****************************************************************


                                        .SBTTL  POWER DOWN AND UP ROUTINES

                                        ;POWER DOWN ROUTINE
                012734  012737  013062  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
                012742  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
                012750  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
                012752  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
                012754  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
                012756  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
                012760  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
                012762  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
                012764  010637  013066                  MOV     SP,$SAVR6       ;;SAVE SP
                012770  012737  013002  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
                012776  000000                          HALT
                013000  000776                          BR      .-2             ;;HANG UP

                                        ;POWER UP ROUTINE
                013002  013706  013066  $PWRUP: MOV     $SAVR6,SP       ;;GET SP
                013006  005037  013066          CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
                013012  005237  013066  1$:     INC     $SAVR6          ;;WAIT FOR THE INC
                013016  001375                  BNE     1$              ;;OF  WORD
                013020  012605                  MOV     (SP)+,R5        ;;POP STACK INTO R5
                013022  012604                  MOV     (SP)+,R4        ;;POP STACK INTO R4
                013024  012603                  MOV     (SP)+,R3        ;;POP STACK INTO R3
                013026  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
                013030  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
                013032  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
                013034  012737  012734  000024  MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
                013042  012737  000340  000026  MOV     #340,@#PWRVEC+2 ;;PRIO:7
                013050  104400                  TYPE                    ;REPORT THE POWER FAILURE
                013052  013070  $PWRMG: .WORD   $POWER                 ;;POWER FAIL MESSAGE POINTER
                013054  012716          MOV     (PC)+,(SP)              ;;RESTART AT RESTART
                013056  002532  $PWRAD: .WORD   RESTART                 ;;RESTART ADDRESS
                013060  000002          RTI
```

```
2323  013062  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
2324  013064  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
2325  013066  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
2326  013070  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
2327  013076  000122
2328                                          .EVEN
2329                                  ;*********************************************************************
2330
2331                                  .SBTTL  SINGLE LENGTH BINARY TO DECIMAL ASCIZ ROUTINE
2332
2333                                  ;*THIS ROUTINE WILL CONVERT A 16-BIT UNSIGNED BINARY NUMBER TO AN
2334                                  ;*UNSIGNED DECIMAL ASCIZ NUMBER.
2335                                  ;*CALL
2336                                  ;*      MOV     NUMBER,-(SP)    ;;PUT BINARY NUMBER ON THE STACK
2337                                  ;*      JSR     PC,@#$$B2D      ;;CALL
2338                                  ;*      RETURN                  ;;ADDRESS OF THE 1ST ASCIZ CHAR.IS ON THE STACK
2339
2340
2341  013100  016637  000002  013124  $$B2D:  MOV     2(SP),1$        ;;SAVE BINARY NUMBER
2342  013106  012746  013124                  MOV     #1$,-(SP)       ;;SET POINTER
2343  013112  004737  013130                  JSR     PC,@#$DB2D      ;;CALL DOUBLE LENGTH CONVERT
2344  013116  012666  000002                  MOV     (SP)+,2(SP)     ;;PICKUP POINTER
2345  013122  000207                          RTS     PC              ;;RETURN
2346  013124  000000  000000          1$:     .WORD   0,0
2347                                  ;*********************************************************************
2348
2349                                  .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
2350
2351                                  ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
2352                                  ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
2353                                  ;*POSITIVE.
2354                                  ;*CALL
2355                                  ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
2356                                  ;*      JSR     PC,@#$DB2D
2357                                  ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
2358                                                                  ;;IS ON THE STACK
2359
2360
2361  013130  104404                  $DB2D:  SAVREG                  ;;SAVE REGISTERS
2362  013132  016602  000002                  MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
2363  013136  012700  013310                  MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
2364  013142  010066  000002                  MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
2365  013146  012201                          MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
2366  013150  012202                          MOV     (R2)+,R2
2367  013152  012737  000012  013226          MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
2368  013160  012704  013240                  MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
2369  013164  012705  013242                  MOV     #$TNPWR+2,R5
2370  013170  005003                  1$:     CLR     R3              ;;CLEAR PARTIAL
2371  013172  161401                  2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
2372  013174  005602                          SBC     R2
2373  013176  161502                          SUB     (R5),R2
2374  013200  002402                          BLT     3$              ;;BR IF TEN POWER TO LARGE
2375  013202  005203                          INC     R3              ;;ADD 1 TO PARTIAL
2376  013204  000772                          BR      2$              ;;LOOP
```

```
2377  013206  062401                  3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
2378  013210  005502                          ADC     R2
2379  013212  062402                          ADD     (R4)+,R2
2380  013214  022525                          CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
2381  013216  052703  000060                  BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
2382  013222  110320                          MOVB    R3,(R0)+        ;;SAVE IT
2383  013224  005327                          DEC     (PC)+           ;;DONE?
2384  013226  000000                  4$:     .WORD   0
2385  013230  001357                          BNE     1$              ;;BR IF NO
2386  013232  105020                          CLRB    (R0)+           ;;TERMINATOR
2387  013234  104405                          RESREG                  ;;RESTORE REGISTERS
2388  013236  000207                          RTS     PC              ;;RETURN
2389  013240  145000                  $TNPWR: 145000                  ;;1.0E09
2390  013242  035632                          35632
2391  013244  160400                          160400                  ;;1.0E08
2392  013246  002765                          2765
2393  013250  113200                          113200                  ;;1.0E07
2394  013252  000230                          230
2395  013254  041100                          041100                  ;;1.0E06
2396  013256  000017                          17
2397  013260  103240                          103240                  ;;1.0E05
2398  013262  000001                          1
2399  013264  023420                          23420                   ;;1.0E04
2400  013266  000000                          0
2401  013270  001750                          1750                    ;;1.0E03
2402  013272  000000                          0
2403  013274  000144                          144                     ;;1.0E02
2404  013276  000000                          0
2405  013300  000012                          12                      ;;1.0E01
2406  013302  000000                          0
2407  013304  000001                          1                       ;;1.0E00
2408  013306  000000                          0
2409  013310  000014                  $DECVL: .BLKB   12.             ;;RESERVE STORAGE FOR ASCIZ STRING
2410
2411                                  ;*****************************************************************
2412
2413                                          ;TYPE NUMERICAL ASCIZ STRING,RIGHT JUSTIFIED
2414                                          ;REPLACING LEADING ZEROS WITH SPACES.
2415                                          ;
2416                                          ;FIRST ADDRESS OF ASCIZ STRING MUST BE ON TOP OF THE STACK
2417
2418  013324  010046                  RTJUST:         MOV R0,-(SP)            ;SAVE R0
2419  013326  016600  000004                         MOV 4(SP),R0            ;PICK UP ADDRESS OF ASCIZ STRING
2420  013332  010037  013364                         MOV R0,3$               ;SAVE ADDRESS FOR TYPE OUT
2421  013336  105710                  1$:             TSTB (R0)               ;IS THIS THE TERMINATOR
2422  013340  001406                                  BEQ 2$                  ;IF YES TYPE IT OUT
2423  013342  122710  000060                         CMPB #'0,(R0)           ;IS IT A ZERO
2424  013346  001005                                  BNE 4$                  ;IF NO GO PRINT IT
2425  013350  112720  000040                         MOVB #' ,(R0)+          ;IF YES REPLACE IT WITH A SPACE
2426  013354  000770                                  BR 1$                   ;TEST NEXT CHAR.
2427  013356  112740  000060          2$:             MOVB #'0,-(R0)          ;STRING OFF ALL ZEROS,PUT BACK THE LAST ONE
2428  013362  104400                  4$:             TYPE                    ;TYPE THE STRING
2429  013364  000000                  3$:             OPEN
2430  013366  012600                                  MOV (SP)+,R0            ;RESTORE R0
```

```
2431  013370  012616                          MOV (SP)+,(SP)          ;RESTORE THE STACK
2432  013372  000207                          RTS PC                  ;RETURN
2433
2434                                  ;TYPES 16 BIT WORD IN DECIMAL
2435
2436  013374  012546                  SGLDEC: MOV (R5)+,-(SP)         ;PUT NUMBER TO BE TYPED ON STACK
2437  013376  004737  013100                  JSR PC,@#$B2D           ;CONVERT NUMBER TO DECIMAL
2438  013402  062716  000005                  ADD #5,(SP)             ;MOVE ADDRESS OF ASCIZ STRING OVER BY 5 BYTES
2439                                                                  ;TO TYPE SINGLE DECIMAL NUMBER
2440  013406  004737  013324                  JSR PC,RTJUST           ;TYPE THE DECIMAL NUMBER
2441  013412  000205                          RTS R5
2442
2443
2444  013414  047125  054105  042520  MUNXDD: .ASCIZ "UNEXPECTED D D MARK"
2445  013422  052103  042105  042040
2446  013430  042040  046440  051101
2447  013436  000113
2448
2449  013440  020104  020104  040515  MDDMIS: .ASCIZ "D D MARK MISSING"
2450  013446  045522  046440  051511
2451  013454  044523  043516     000
2452
2453
2454  013461     104  052101  026101  MDERHDR: .ASCIZ "DATA, NO STATUS ERROR"
2455  013466  047040  020117  052123
2456  013474  052101  051525  042440
2457  013502  051122  051117     000
2458
2459  013507     040  047117  052040  MTRK:   .ASCIZ " ON TRACK"
2460  013514  040522  045503     000
2461
2462
2463  013521     040  043040  047522  MPRES:  .ASCIZ "  FROM TRACK"
2464  013526  020115  051124  041501
2465  013534  000113
2466
2467  013536  027440  051440  041505  MSECT:  .ASCIZ " / SECTOR"
2468  013544  047524  000122
2469
2470  013550  005015  041040  052131  MCOLMUN: .ASCIZ <15><12>" BYTE  BAD  GOOD"<15><12>
2471  013556  020105  041040  042101
2472  013564  020040  047507  042117
2473  013572  005015     000
2474
2475  013575     015  041412  052501  DOLOAD: .ASCII  <15><12>"CAUTION - IF YOU DESIRE TO TEST UNIT 0"
2476  013602  044524  047117  026440
2477  013610  044440  020106  047531
2478  013616  020125  042504  044523
2479  013624  042522  052040  020117
2480  013632  042524  052123  052440
2481  013640  044516  020124     060
2482  013645     015  051012  050105          .ASCII  <15><12>"REPLACE LOAD MEDIUM WITH A SCRATCH DISKETTE"
2483  013652  040514  042503  046040
2484  013660  040517  020104  042515
```

```
2485  013666  044504  046525  053440
2486  013674  052111  020110  020101
2487  013702  041523  042522  041524
2488  013710  020110  044524  045523
2489  013716  052105  042124
2490  013722  005015  044124  047105          .ASCIZ  <15><12>"THEN PRESS CONTINUE"
2491  013730  050040  042522  051523
2492  013736  041440  047117  044524
2493  013744  052516  000105
2494
2495  013750  047516  042040  044522  MNODRY:         .ASCIZ "NO DRIVES READY"<15><12>
2496  013756  042526  020123  042522
2497  013764  042101  006531  000012
2498
2499  013772  020040  020040  020040  TAB:            .ASCIZ <40><40><40><40><40><40>
2500  014000      000
2501
2502  014001      040  000040          DBLSP:          .ASCIZ <40><40>
2503
2504  014004  005015      000          MCRLF:          .ASCIZ <15><12>
2505
2506  014007      104      000          MEOP:           .ASCIZ "D"
2507
2508  014011      055      000          MEREOP:         .ASCIZ "-"
2509
2510  014013      007      000          MABELL:         .ASCIZ <07>
2511
2512  014015      105  051122  051117  MERHEADER:      .ASCIZ "ERROR CONDITIONS  "
2513  014022  041440  047117  044504
2514  014030  044524  047117  020123
2515  014036  000040
2516
2517  014040  047125  052111  030040  MUNIT0:         .ASCIZ "UNIT 0 "
2518  014046  000040
2519
2520  014050  047125  052111  030440  MUNIT1:         .ASCIZ "UNIT 1 "
2521  014056  000040
2522
2523  014060  040510  020123  042502  MDELET:         .ASCIZ "HAS BEEN DELETED."
2524  014066  047105  042040  046105
2525  014074  052105  042105  000056
2526
2527  014102  044440  052116  053040  MINTVEC:        .ASCIZ " INT VECTOR = "
2528  014110  041505  047524  020122
2529  014116  020075      000
2530
2531  014121      122  041530  020123  MRXCS:          .ASCIZ "RXCS = "
2532  014126  020075      000
2533
2534  014131      040  054122  041104  MRXDB:          .ASCIZ " RXDB = "
2535  014136  036440  000040
2536
2537  014142  052123  052101  051525  MASTAT:         .ASCIZ "STATUS A = "
2538  014150  040440  036440  000040
```

```
014156 052123 052101 051525 MBSTAT:     .ASCIZ "STATUS B = "
014164 041040 036440 000040

014172 005015 000012        DBLLF:      .ASCIZ <15><12><12>

014176 036520 000040        MPAT:       .ASCIZ "P= "

014202 036524 000040        MTEST:      .ASCIZ "T= "

014206 036523 000040        MSEQ:       .ASCIZ "S= "

014212 047516 044440 052116 MINTER:     .ASCIZ "NO INTERUPT AT DONE ERROR"
014220 051105 050125 020124
014226 052101 042040 047117
014234 020105 051105 047522
014242 000122

014244 047125 047113 053517 MUKNINT:    .ASCIZ "UNKNOWN INTERUPT"
014252 020116 047111 042524
014260 052522 052120    000

014265    124 052117 046101 MERCT:      .ASCIZ "TOTAL READ CHECK ERRORS = "
014272 051040 040505 020104
014300 044103 041505 020113
014306 051105 047522 051522
014314 036440 000040

014320 044506 046114 052502 MFIL:       .ASCIZ "FILLBUFFER "
014326 043106 051105 000040

014334 046505 052120 041131 MEMPTY:     .ASCIZ "EMPTYBUFFER "
014342 043125 042506 020122
014350    000

014351    104 044522 042526 MICON:      .ASCIZ "DRIVE(S)  "
014356 051450 020051 000040

014364 042524 052123 044040 MHUNG:      .ASCIZ "TEST HUNG"<15><12>
014372 047125 006507 000012

014400 047516 051516 040524 MNONSTD:    .ASCIZ "NONSTANDARD TRACK / SECTOR LIMITS"
014406 042116 051101 020104
014414 051124 041501 020113
014422 020057 042523 052103
014430 051117 046040 046511
014436 052111 000123

014442 042117 000075        MOD:        .ASCIZ "OD="

014446 042111 000075        MID:        .ASCIZ "ID="

014452 044506 051522 036524 MFIRST:     .ASCIZ "FIRST="
014460    000
```

F13


```
2593
2594  014461     114  051501  036524  MLAST:          .ASCIZ "LAST="
2595  014466     000
2596
2597  014467     111  044516  027124  MINIT1:         .ASCIZ "INIT.DONE NOT SET ERROR"<15><12>
2598  014474  047504  042516  047040
2599  014502  052117  051440  052105
2600  014510  042440  051122  051117
2601  014516  005015     000
2602
2603  014521     111  044516  027124  MINIT2:         .ASCIZ "INIT.DONE SET ERROR" <15><12>
2604  014526  047504  042516  051440
2605  014534  052105  042440  051122
2606  014542  051117  005015     000
2607
2608  014547     104  042040  042440  MODER:          .ASCIZ "D D ERROR"
2609  014554  051122  051117     000
2610
2611  014561     122  041505  053117  MREC:           .ASCIZ "RECOVERABLE "
2612  014566  051105  041101  042514
2613  014574  000040
2614
2615  014576  051103  020103  051105  MBADCRC:        .ASCIZ "CRC ERROR NO DATA ERROR"
2616  014604  047522  020122  047516
2617  014612  042040  052101  020101
2618  014620  051105  047522  000122
2619
2620  014626  042522  042101  000040  MREAD:          .ASCIZ "READ "
2621
2622  014634  040510  052114  032040  MHALT4:         .ASCIZ "HALT 4" <15><12>
2623  014642  005015     000
2624
2625  014645     110  0[illegible]  020124  MHALT3:    .ASCIZ "HALT 3" <15><12>
2626  014652  006463  0    2
2627
2628  014656  040510  052114  030440  MHALT11:        .ASCIZ "HALT 11" <15><12>
2629  014664  006461  000012
2630
2631  014670  040504  040524  041440  MCRC:           .ASCIZ "DATA CRC ERROR"
2632  014676  041522  042440  051122
2633  014704  051117     000
2634
2635
2636  014707     040  047125  042522  MUNREC:         .ASCIZ " UNRECOVERABLE "
2637  014714  047503  042526  040522
2638  014722  046102  020105     000
2639
2640  014727     123  042505  020113  MSEEK:          .ASCIZ "SEEK ERROR"
2641  014734  051105  047522  000122
2642
2643  014742  051127  052111  020105  MWRITE:         .ASCIZ "WRITE "
2644  014750     000
2645
2646  014751     120  051101  052111  MPAR:           .ASCIZ "PARITY ERROR"
```

```
2647  014756  020131  051105  047522
2648  014764  000122
2649
2650  014766  051105  047522  020122  MNOFLAG:        .ASCIZ "ERROR FLAG ERROR"
2651  014774  046106  043501  042440
2652  015002  051122  051117     000
2653
2654  015007     122  030530  020061  MDUMP:          .ASCIZ "RX11 DUMP"
2655  015014  052504  050115     000
2656
2657  015021     105  051122  051117  MERS:           .ASCIZ "ERRORS"
2658  015026  000123
2659
2660  015030  044123  051117  020124  MSHORT:         .ASCIZ "SHORT DUMP"
2661  015036  052504  050115     000
2662
2663  015043     122  051505  040524  MRESTART:       .ASCIZ "RESTARTS = "
2664  015050  052122  020123  020075
2665  015056     000
2666
2667  015057     120  051501  020123  MNOPAS:         .ASCIZ "PASS ABORTED"
2668  015064  041101  051117  042524
2669  015072  000104
2670
2671  015074  040520  051523  051505  MPASS:          .ASCIZ "PASSES = "
2672  015102  036440  000040
2673
2674  015106  051127  052111  042524  MWRTEN:         .ASCIZ "WRITTEN "
2675  015114  020116     000
2676
2677  015117     057  000040          MSLASH:         .ASCIZ "/ "
2678
2679  015122  000040                  SPACE:          .ASCIZ <40>
2680
2681  015124  051105  047522  051522  MCODES:         .ASCIZ "ERRORS PER ERROR CODES"<15><12>
2682  015132  050040  051105  042440
2683  015140  051122  051117  041440
2684  015146  042117  051505  005015
2685  015154     000
2686
2687  015155     124  040522  045503  MTKLOG:         .ASCIZ "TRACK  ACCESSED   MOVED TO    UNIT 0  UNIT 1 ERRORS"<15><12>
2688  015162  020040  041501  042503
2689  015170  051523  042105  020040
2690  015176  046440  053117  042105
2691  015204  052040  020117  020040
2692  015212  052440  044516  020124
2693  015220  020060  052440  044516
2694  015226  020124  020061  051105
2695  015234  047522  051522  005015
2696  015242     000
2697
2698  015243     050  047516  042516  MNONE:          .ASCIZ "(NONE)"<15><12>
2699  015250  006451  000012
2700
```

```
015254 047507 042117 000 MGOOD:          .ASCIZ "GOOD"

015261 040 041440 042510 MSUM:           .ASCIZ "  CHECK SUM "
015266 045503 051440 046525
015274 000040

015276 005015 054122 030461 MRX11:       .ASCIZ <15><12>"RX11 / RXV11"
015304 027440 051040 053130
015312 030461 000

015315 015 046412 044501 MREV:           .ASCIZ <15><12> "MAINDEC-11-DZRXA-D" <15><12>
015322 042116 041505 030455
015330 026461 055104 054122
015336 026501 006504 000012

                                         .EVEN

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

                                         ;THE FOLLOWING LOCATIONS ARE USED FOR DATA STORAGE,RETRY COUNTERS
                                         ;ACCESS COUNTERS ETC.

015344 000200                    BUFADR: .BLKB 200

                                         ;RETRY COUNTERS
015544 000012                    RDRETRY:    10.          ;FOR SETUP PURPOSE RDRETRY MUST BE FIRST ON RETRY LIST
015546 000012                    WTRETRY:    10.
015550 000012                    DDRETRY:    10.
015552 000012                    DATARETRY:  10.
015554 000012                    P2RETRY:    10.
015556 000012                    PRETRY:     10.
015560 000012                    CRETRY:     10.
015562 000012                    SRETRY:     10.          ;FOR SETUP PURPOSE SRETRY MUST BE LAST ON RETRY LIST

                                 ;GENERAL ERROR COUNTERS
015564 000000                    PARLOG:     0            ;FOR SETUP PURPOSE PARLOG MUST BE FIRST ON COUNTERS LIST
015566 000000                    HPARLOG:    0
015570 000000                    NOERLOG:    0
015572 000000                    UKNINT:     0
015574 000000                    INTER:      0

                                 ;DRIVE RELATED ERROR COUNTERS
015576 000000                    ZSEKLOG:    0            ;"Z" LOGS ARE FOR DRIVE UNIT "Z"ERO
015600 000000                    SEKLOG:     0            ;NON"Z" LOGS ARE FOR DRIVE UNIT 1
015602 000000                    ZCRCLOG:    0
015604 000000                    CRCLOG:     0            ; * * * NOTE: * * *
015606 000000                    ZCRCBAD:    0            ;ERROR CODES MUST NOT BE CHANGED
015610 000000                    CRCBAD:     0            ;FROM THIS ORDER. ERROR DUMP
015612 000000                    ZRDLOG:     0            ;ASSUMES TAGS OF LOGS ARE IN ORDER SHOWN.
015614 000000                    RDLOG:      0
015616 000000                    ZWRTLOG:    0
015620 000000                    WRTLOG:     0
015622 000000                    ZDATALOG:   0
015624 000000                    DATALOG:    0
```

```
2755  015626  000000                  ZDDMIS:         0
2756  015630  000000                  DDMIS:          0
2757  015632  000000                  ZUNXDD:         0
2758  015634  000000                  UNXDD:          0
2759  015636  000000                  ZHSEKLOG:       0
2760  015640  000000                  HSEKLOG:        0
2761  015642  000000                  ZHCRCLOG:       0
2762  015644  000000                  HCRCLOG:        0
2763  015646  000000                  ZHCRCBAD:       0
2764  015650  000000                  HCRCBAD:        0
2765  015652  000000                  ZHRDLOG:        0
2766  015654  000000                  HRDLOG:         0
2767  015656  000000                  ZHWRTLOG:       0
2768  015660  000000                  HWRTLOG:        0
2769  015662  000000                  ZHDATALOG:      0
2770  015664  000000                  HDATALOG:       0
2771  015666  000000                  ZHDDLOG:        0
2772  015670  000000                  HDDLOG:         0
2773
2774                                          ;ERROR CODES
2775  015672  000021                  ERCODE:         .BLKW 17.
2776
2777                                          ;THE FOLLOWING 2 BLOCKS OF WORDS ARE TRACK ACCESS COUNTER AND
2778                                          ;HEAD MOVED TO TRACK COUNTERS. THEY ARE DOUBLE PRECISION
2779                                          ;COUNTERS IN THE FOLLOWING FORMATT:
2780                                          ;
2781                                          ;LOC.X          LOW ORDER WORD TRACK 0
2782                                          ;LOC.X+2        HIGH ORDER WORD TRACK 0
2783                                          ;LOC.X+4        LOW ORDER WORD TRACK 1
2784                                          ;LOC.X+6        HIGH ORDER WORD TRACK 1
2785                                          ;ETC.
2786
2787                                          ;TOTAL ACCESS / TRACK
2788  015734  000232                  TKACC:          .BLKW 232
2789
2790                                          ;TOTAL HEAD MOVEMENT TO TRACK
2791  016420  000232                  HDMOVE:         .BLKW 232
2792
2793                                          ;THE FOLLOWING 2 BLOCKS OF COUNTERS ARE SINGLE PRECISION
2794
2795                                          ;ERROR PER TRACK ON UNIT 0
2796  017104  000115                  U0TRK:          .BLKW 115
2797
2798                                          ;ERROR PER TRACK ON UNIT 1
2799  017336  000115                  U1TRK:          .BLKW 115
2800
2801                                          ;TOTAL WRITE AND READ FUNCTIONS DOUBLE PRECISION COUNTERS
2802  017570  000000  000000          WTCNTR:         .WORD 0,0
2803  017574  000000  000000          RDCNTR:         .WORD 0,0
2804
2805                                          ;RESTART AND PASSES COMPLTED COUNTERS
2806  017600  000000                  RESTCNTR:       0
2807  017602  000000                  PASCNTR:        0               ;FOR SETUP PURPOSE PASCNTR MUST BE PLACED LAST
2808
```

```
2809  017604  032737  000020  010464  U1LOG:    BIT #BIT4,UNITSEL       ;TEST FOR UNIT IN OPERATION
2810  017612  001402                            BEQ 1$                  ;IF BIT 4 IS 0 RETURN AND INC. UNIT 0 COUNTERS
2811  017614  062703  000002                    ADD #2,R3               ;ADJUST THE CONTENTS OF R3 FOR
2812                                                                    ;THE ADDRESS OF UNIT 1 ERROR LOG COUNTERS.
2813  017620  000207                  1$:       RTS PC
2814
2815                                      ;STATISTICAL ERROR REPORT ROUTEEN
2816                                      ;PRINTS ALL ERROR LOGS AND OPERATING CONDITIONS
2817
2818                                      ;*****************************************************************
2819
2820  017622  012706  001200            ERDUMP:   MOV #STACK,SP
2821  017626  104400  014172                      TYPE ,DBLLF
2822  017632  005737  021254                      TST SHORTRPT            ;IS SHORT REPORT REQUESTED
2823  017636  001405                              BEQ 1$                  ;NO,PRINT LONG REPORT
2824  017640  104400  013772                      TYPE ,TAB
2825  017644  104400  015030                      TYPE ,MSHORT            ;YES,TYPE SHORT REPOR HEADER
2826  017650  000404                              BR 2$
2827  017652  104400  013772            1$:       TYPE ,TAB
2828  017656  104400  015007                      TYPE ,MDUMP             ;TYPE LONG REPORT HEADER
2829  017662  104400  015315            2$:       TYPE ,MREV              ;PRINT NAME AND REVISION OF PROGRAM
2830  017666  004737  002306                      JSR PC,ICOND            ;TYPE OUT SELECTED DRIVES AND TESTS
2831  017672  005737  017600                      TST RESTCNTR            ;HAVE THERE BEEN ANY RESTARTS
2832  017676  001412                              BEQ 3$                  ;NO
2833  017700  104400  015043                      TYPE ,MRESTART          ;YES,PRINT OUT HOW MANY
2834  017704  013737  017600  017716              MOV RESTCNTR,12$
2835  017712  004537  013374                      JSR R5,SGLDEC
2836  017716  000000            12$:      OPEN
2837  017720  104400  014172                      TYPE ,DBLLF
2838  017724  005737  017602            3$:       TST PASCNTR             ;HAVE ANY PASSES BEEN COMPLETED
2839  017730  001005                              BNE 4$                  ;YES,PRINT OUT HOW MANY
2840  017732  104400  015057                      TYPE ,MNOPAS            ;NO,TYPE PASS ABORTED
2841  017736  104400  014172                      TYPE ,DBLLF
2842  017742  000412                              BR 5$
2843  017744  104400  015074            4$:       TYPE ,MPASS             ;TYPE OUT NUMBER OF PASSES
2844  017750  013737  017602  017762              MOV PASCNTR,7$
2845  017756  004537  013374                      JSR R5,SGLDEC
2846  017762  000000            7$:       OPEN
2847  017764  104400  014004                      TYPE ,MCRLF
2848  017770  005737  021254            5$:       TST SHORTRPT            ;IS THIS A SHORT REPORT
2849  017774  001030                              BNE AERRS               ;YES,DON'T PRINT WRITE/READ TOTALS
2850
2851                                      ;PRINTS TOTAL NUMBER OF SECTORS WRITTEN AND/OR
2852                                      ;READ FOR THE DURATION OF RUN TIME.
2853  017776  104400  015106                      TYPE ,MWRTEN            ;PRINT WRT/RD MESSAGE
2854  020002  104400  015117                      TYPE ,MSLASH
2855  020006  104400  014626                      TYPE ,MREAD
2856  020012  104400  014001                      TYPE,DBLSP
2857  020016  012746  017570                      MOV #WTCNTR,-(SP)       ;ADDRESS OF TOTAL WRITES ON STACK
2858  020022  004737  013130                      JSR PC,@#$DB2D          ;TYPE TOTAL WRITES
2859  020026  004737  013324                      JSR PC,RTJUST
2860  020032  104400  015117                      TYPE ,MSLASH
2861  020036  012746  017574                      MOV #RDCNTR,-(SP)       ;ADDRESS OF READS ON STACK
2862  020042  004737  013130                      JSR PC,@#$DB2D          ;TYPE TOTAL READS
```

```
 2863  020046  004737  013324                        JSR PC,RTJUST
 2864  020052  104400  014172                        TYPE ,DBLLF
 2865
 2866                                    ;TYPE OUT MICRO CPU RELATED ERRORS
 2867  020056  104400  014004          AERRS:        TYPE ,MCRLF
 2868  020062  005737  015564                        TST PARLOG              ;WERE THERE ANY PARITY ERRORS
 2869  020066  001435                                BEQ 1$
 2870  020070  013737  015564  020102                MOV PARLOG,20$          ;YES,TYPE OUT THE NUMBER
 2871  020076  004537  013374                        JSR R5,SGLDEC
 2872  020102  000000                    20$:        OPEN
 2873  020104  104400  014001                        TYPE ,DBLSP
 2874  020110  104400  014751                        TYPE ,MPAR
 2875  020114  104400  014004                        TYPE,MCRLF
 2876  020120  005737  015566                        TST HPARLOG             ;WERE THERE ANY HARD PARITY ERRORS
 2877  020124  001416                                BEQ 1$
 2878  020126  013737  015566  020140                MOV HPARLOG,21$         ;YES,TYPE OUT THE COUNT
 2879  020134  004537  013374                        JSR R5,SGLDEC
 2880  020140  000000                    21$:        OPEN
 2881  020142  104400  014001                        TYPE ,DBLSP
 2882  020146  104400  014707                        TYPE ,MLSPEC
 2883  020152  104400  014751                        TYPE ,MPAR
 2884  020156  104400  014004                        TYPE ,MCRLF
 2885  020162  005737  015570          1$:           TST NOERLOG             ;ANY STATUS ERROR FLAG ERRORS
 2886  020166  001414                                BEQ 3$                  ;IF NONE CHECK NEXT ERROR
 2887  020170  013737  015570  020202                MOV NOERLOG,24$         ;YES,TYPE OUT COUNT
 2888  020176  004537  013374                        JSR R5,SGLDEC
 2889  020202  000000                    24$:        OPEN
 2890  020204  104400  014001                        TYPE ,DBLSP
 2891  020210  104400  014766                        TYPE ,MERFLAG
 2892  020214  104400  014004                        TYPE ,MCRLF
 2893  020220  005737  015574          3$:           TST INTER               ;ANY NO INTERUPT ON DONE ERRORS
 2894  020224  001414                                BEQ 4$                  ;IF NONE CHECK NEXT ERROR GROUP
 2895  020226  013737  015574  020240                MOV INTER,25$           ;YES,PRINT OUT NUMBER
 2896  020234  004537  013374                        JSR R5,SGLDEC
 2897  020240  000000                    25$:        OPEN
 2898  020242  104400  014001                        TYPE ,DBLSP
 2899  020246  104400  014212                        TYPE ,MINTER
 2900  020252  104400  014004                        TYPE ,MCRLF
 2901                                    ;TYPES DRIVE RELATED ERRORS
 2902  020256  104400  014172          4$:           TYPE ,DBLLF
 2903  020262  104400  014040                        TYPE ,MUNITO            ;TYPE OUT COLUMN HEADINGS
 2904  020266  104400  014050                        TYPE ,MUNITI
 2905  020272  104400  015122                        TYPE ,SPACE
 2906  020276  104400  015021                        TYPE ,MERS
 2907  020302  104400  014004                        TYPE ,MCRLF
 2908  020306  005002                                CLR R2                  ;R2 USED AS AN ERROR PRINTED FLAG
 2909  020310  013746  021256                        MOV ZERO,-(SP)          ;PUT ADDR OF ZERO MARK ON STACK
 2910  020314  012746  013414                        MOV #MUNXDD,-(SP)       ;PUT ADDRESSES OF ALL RECOVERABLE
 2911  020320  012746  013440                        MOV #MDDMIS,-(SP)       ;ERROR MESSAGES ON THE STACK
 2912  020324  012746  013461                        MOV #MDERHDR,-(SP)
 2913  020330  012746  014742                        MOV #MWRITE,-(SP)
 2914  020334  012746  014626                        MOV #MREAD,-(SP)
 2915  020340  012746  014576                        MOV #MBADCRC,-(SP)
 2916  020344  012746  014670                        MOV #MCRC,-(SP)
```

```
2917  020350  012746  014727                    MOV #MSEEK,-(SP)
2918  020354  012701  015576                    MOV #ZSEKLOG,R1         ;PUT FIRST ERROR COUNTER IN REGISTER
2919  020360  005716                  7$:       TST (SP)                ;GET ADDRESS OF MESSAGE
2920  020362  001436                            BEQ 5$                  ;IF 0 FINISHED SOFT ERRS DO HARD ERRS
2921  020364  005721                            TST (R1)+               ;IS DRIVE 0 COUNTER CLEAR
2922  020366  001005                            BNE 6$                  ;NO,GO PRINT ERROR COUNTER
2923  020370  005711                            TST (R1)                ;IS DRIVE 1 COUNTER CLEAR
2924  020372  001003                            BNE 6$                  ;NO,GO PRINT ERROR COUNTER
2925  020374  005721                            TST (R1)+               ;ADJUST R1 FOR NEXT UNIT 0 COUNTER
2926  020376  005726                            TST (SP)+               ;ADJUST STACK FOR NEXT ADDRESS
2927  020400  000767                            BR 7$                   ;GO TEST NEXT PAIR OF COUNTERS
2928  020402  005202                  6$:       INC R2                  ;MAKE ERROR PRINTED FLAG NON ZERO
2929  020404  014137  020414                    MOV -(R1),30$           ;TYPE CONTENTS OF UNIT 0 COUNTER
2930  020410  004537  013374                    JSR R5,SGLDEC
2931  020414  000000                  30$:      OPEN
2932  020416  104400  014001                    TYPE ,DBLSP
2933  020422  005721                            TST (R1)+               ;ADJUST R1 FOR UNIT 1 COUNTER
2934  020424  012137  020434                    MOV (R1)+,31$           ;TYPE COUNTER,GET ADDR OF NEXT IN R1
2935  020430  004537  013374                    JSR R5,SGLDEC
2936  020434  000000                  31$:      OPEN
2937  020436  104400  014001                    TYPE ,DBLSP
2938  020442  012637  020450                    MOV (SP)+,13$
2939  020446  104400                            TYPE                    ;TYPE ERROR MESSAGE FROM STACK
2940  020450  000000                  13$:      OPEN
2941  020452  104400  014004                    TYPE ,MCRLF
2942  020456  000740                            BR 7$                   ;GET NEXT COUNTER
2943  020460  104400  014004            5$:     TYPE ,MCRLF
2944  020464  013746  021256                    MOV ZERO,-(SP)          ;PUT ADDR OF ZERO MARK ON THE STACK
2945  020470  012746  014547                    MOV #MDDER,-(SP)        ;PUT ADDRESS OF ALL HARD ERROR
2946  020474  012746  013461                    MOV #MDERHDR,-(SP)      ;MESSAGES ON THE STACK
2947  020500  012746  014742                    MOV #MWRITE,-(SP)
2948  020504  012746  014626                    MOV #MREAD,-(SP)
2949  020510  012746  014576                    MOV #MBADCRC,-(SP)
2950  020514  012746  014670                    MOV #MCRC,-(SP)
2951  020520  012746  014727                    MOV #MSEEK,-(SP)
2952  020524  012701  015636                    MOV #ZHSEKLOG,R1        ;PUT FIRST HARD ERROR COUNTER IN R1
2953  020530  005716                  12$:      TST (SP)                ;GET ADDRESS OF MESSAGE FROM STACK
2954  020532  001437                            BEQ BERRS               ;IF 0 FINISHED ERROR LOGS
2955  020534  005721                            TST (R1)+               ;IS DRIVE 0 COUNTER CLEAR
2956  020536  001005                            BNE 11$                 ;NO,PRINT ERROR COUNTER AND MESSAGE
2957  020540  005711                            TST (R1)                ;IS DRIVE 1 COUNTER CLEAR
2958  020542  001003                            BNE 11$                 ;NO,GO PRINT THE ERROR
2959  020544  005721                            TST (R1)+               ;ADJUST R1 FOR NEXT UNIT 0 LOG
2960  020546  005726                            TST (SP)+               ;ADJUST STACK FOR NEXT ADDRESS
2961  020550  000767                            BR 12$
2962  020552  014137  020562          11$:      MOV -(R1),32$           ;TYPE CONTENTS OF UNIT 0 COUNTER
2963  020556  004537  013374                    JSR R5,SGLDEC
2964  020562  000000                  32$:      OPEN
2965  020564  104400  014001                    TYPE ,DBLSP
2966  020570  005721                            TST (R1)+               ;ADJUST R1 FOR UNIT 1 COUNTER
2967  020572  012137  020602                    MOV (R1)+,33$           ;TYPE UNIT 1 LOG
2968  020576  004537  013374                    JSR R5,SGLDEC
2969  020602  000000                  33$:      OPEN
2970  020604  104400  014001                    TYPE ,DBLSP
```

```
2971  020610  104400  014707                    TYPE ,MUNREC            ;TYPE UNRECOVERABLE
2972  020614  012637  020622                    MOV (SP)+,14$
2973  020620  104400                            TYPE                    ;ERROR MESSAGE FROM STACK
2974  020622  000000                 14$:       OPEN
2975  020624  104400  014004                    TYPE ,MCRLF
2976  020630  000737                            BR 12$                  ;GET NEXT LOG
2977                                  ;PRINTS ERRORS PER ERROR CODES
2978  020632  005702                 BERRS:     TST R2                  ;WAS AN ERROR PRINTED IN THE LAST GROUP
2979  020634  001004                            BNE 5$                  ;IF R2 IS NONZERO BRANCH
2980  020636  104400  013772                    TYPE ,TAB
2981  020642  104400  015243                    TYPE ,MNONE             ;TYPE NONE
2982  020646  005002                 5$:        CLR R2                  ;CLEAR ERROR PRINTED FLAG AGAIN
2983  020650  104400  014172                    TYPE ,DBLLF
2984  020654  104400  015124                    TYPE ,MCODES            ;TYPE ERROR PER ERROR CODE
2985  020660  012700  000001                    MOV #1,R0               ;SET UP CODE COUNTER
2986  020664  012701  015672                    MOV #ERCODE,R1          ;GET ADDR OF FIRST ERROR CODE
2987  020670  020027  000022         2$:        CMP R0,#22              ;IS THE LAST CODE PRINTED
2988  020674  001001                            BNE 3$                  ;NO,TEST THE NEXT ONE
2989  020676  000427                            BR TYPTRK               ;YES,GO PRINT TRACK INFOMATION
2990  020700  005721                 3$:        TST (R1)+               ;TEST FOR 0 WORD,GET NEXT ADDRESS
2991  020702  001002                            BNE 1$                  ;IF NOT 0 TYPE COUNT AND CODE NUMBER
2992  020704  005200                            INC R0                  ;INC CODE COUNTER
2993  020706  000770                            BR 2$
2994  020710  014137  020720         1$:        MOV -(R1),4$            ;PUT CONTENTS OF COUNTER IN OPEN
2995  020714  004537  013374                    JSR R5,SGLDEC           ;FOR TYPE OUT
2996  020720  000000                 4$:        OPEN
2997  020722  104400  013772                    TYPE ,TAB
2998
2999  020726  010002                            MOV R0,R2               ;PUT CODE NUMBER IN REGISTER TO
3000  020730  006302                            ASL R2                  ;TYPE IT OUT IN THE FROM OF ERROR CODE
3001  020732  006302                            ASL R2
3002  020734  006302                            ASL R2
3003  020736  010246                 MOV        R2,-(SP)                ;;SAVE R2 FOR TYPEOUT
3004  020740  104402                 TYPOS                              ;;GO TYPE--OCTAL ASCII
3005  020742     004                 .BYTE      4                       ;;TYPE 4 DIGIT(S)
3006  020743     001                 .BYTE      1                       ;;TYPE LEADING ZEROS
3007  020744  104400  014004                    TYPE ,MCRLF
3008  020750  005200                            INC R0                  ;SET UP FOR NEXT COUNTER
3009  020752  005721                            TST (R1)+               ;GET NEXT ERROR COUNTER
3010  020754  000745                            BR 2$
3011                                  ;PRINTS TRACK ACCESS,HEAD MOVEMENT,AND ERRORS PER TRACK
3012  020756  005702                 TYPTRK:    TST R2                  ;WAS AN ERROR PRINTED IN LAST GROUP
3013  020760  001004                            BNE 5$                  ;IF PRINTED ERROR FLAG NONZERO BRANCH
3014  020762  104400  013772                    TYPE ,TAB
3015  020766  104400  015243                    TYPE ,MNONE             ;TYPE NONE
3016  020772  104400  014172         5$:        TYPE ,DBLLF
3017  020776  005737  021254                    TST SHORTRPT            ;IF SHORT REPORT DON'T TYPE TRACK ACCESS OR ERRO
3018  021002  001121                            BNE 6$
3019  021004  104400  015155                    TYPE ,MTKLOG            ;TYPE FORMAT OF ERROR PRINT OUT
3020  021010  005037  021070                    CLR 2$                  ;TRACK ADDRESS COUNTER
3021  021014  012704  015730                    MOV #TKACC-4,R4         ;SET UP ADDRESS OF TRACK ACCESSED
3022  021020  012703  016414                    MOV #HDMOVE-4,R3        ;ADDRESS OF HEAD MOVEMENT COUNTER
3023  021024  012702  017104                    MOV #U0TRK,R2           ;UNIT 0 ERROR PER TRACK COUNTER
3024  021030  012701  017336                    MOV #U1TRK,R1           ;UNIT 1 ERROR PER TRACK COUNTER
```

```
3025  021034  062704  000004          1$:     ADD #4,R4           ;ADJUST FOR ADDRESS OF NEXT COUNTER
3026  021040  005714                          TST (R4)            ;HAS THIS TRACK BEEN ACCESSED
3027  021042  001010                          BNE 10$             ;YES,PRINT OUT THE COUNTERS
3028  021044  005764  000002                  TST 2(R4)           ;TEST UPPER WORD OF ACCESS COUNTER
3029  021050  001005                          BNE 10$
3030  021052  062703  000004                  ADD #4,R3           ;THIS TRACK HAS NOT BEEN USED. ADJUST
3031  021056  005722                          TST(R2)+            ;REGISTERS TO POINT TO NEXT TRACK COUNTERS
3032  021060  005721                          TST (R1)+
3033  021062  000463                          BR 12$              ;TEST FOR LAST TRACK
3034  021064  004537  013374          10$:    JSR R5,SGLDEC       ;TYPE TRACK NUMBER
3035  021070  000000          2$:             OPEN                ;THIS LOCATION USED AS TRACK ADDR COUNTER
3036  021072  104400  015122                  TYPE ,SPACE
3037  021076  010446                          MOV R4,-(SP)        ;TYPE TRACK ACCESSED COUNT
3038  021100  004737  013130                  JSR PC,@#$D82D
3039  021104  004737  013324                  JSR PC,RTJUST
3040  021110  104400  015122                  TYPE ,SPACE
3041  021114  062703  000004                  ADD #4,R3
3042  021120  010346                          MOV R3,-(SP)        ;TYPE HEAD MOVED TO COUNT
3043  021122  004737  013130                  JSR PC,@#$D82D
3044  021126  004737  013324                  JSR PC,RTJUST
3045  021132  104400  014001                  TYPE ,DBLSP
3046  021136  104400  015122                  TYPE ,SPACE
3047  021142  005712                          TST (R2)            ;ARE THERE ANY UNIT 0 ERRORS
3048  021144  001421                          BEQ 20$             ;NO,CHECK UNIT 1
3049  021146  012237  021156                  MOV (R2)+,3$        ;TYPE UNIT 0 ERRORS
3050  021152  004537  013374                  JSR R5,SGLDEC
3051  021156  000000          3$:             OPEN
3052  021160  005711                          TST (R1)            ;ARE THERE ANY UNIT 1 ERRORS
3053  021162  001420                          BEQ 7$              ;NO,GO ADJUST POINTERS
3054  021164  104400  015122                  TYPE ,SPACE
3055  021170  104400  014001          22$:    TYPE ,DBLSP
3056  021174  012137  021204                  MOV (R1)+,4$        ;TYPE UNIT 1 ERRORS
3057  021200  004537  013374                  JSR R5,SGLDEC
3058  021204  000000          4$:             OPEN
3059  021206  000407                          BR 11$              ;GO TO NEXT TRACK
3060
3061  021210  005722          20$:            TST (R2)+           ;NO ERRORS,ADJUST REG.FOR NEXT COUNTER
3062  021212  005711                          TST (R1)            ;ARE THERE ANY UNIT 1 ERRORS
3063  021214  001403                          BEQ 7$              ;NO,GO TO NEXT TRACK
3064  021216  104400  013772                  TYPE ,TAB           ;YES,TYPE TAB FOR PLACEMENT
3065  021222  000762                          BR 22$              ;GO TYPE THE NUMBER
3066  021224  005721          7$:             TST (R1)+           ;NO ERRORS,ADJUST REG. FOR NEXT COUNTER
3067  021226  104400  014004          11$:    TYPE ,MCRLF
3068  021232  005237  021070          12$:    INC 2$
3069  021236  023727  021070  000115          CMP 2$,#115         ;HAVE ALL TRACK ERRORS BEEN TYPED
3070  021244  001273                          BNE 1$              ;NO,TYPE THE NEXT ONE
3071  021246  005037  021254          6$:     CLR SHORTRPT        ;SET UP FOR LONG REPORT
3072  021252  000000          HLT17:          HALT                ;YES DONE
3073
3074  021254  000000          SHORTRPT:       0
3075  021256  000000          ZERO:           0
3076
3077          000001                  .END
```

```
ACCSUM  005460            711    1109    1126#
AERRS   020056           2849    2867#
ASTAT   007650            781     936     943    1033    1063    1402#   1405    1408    1519#   1541#   1551
BADBYT  006230           1119#   1120    1121    1171    1229#
BERRS   020632           2954    2978#
BIT0  = 000001            132#    943
BIT00 = 000001            122#    132
BIT01 = 000002            121#    131
BIT02 = 000004            120#    130
BIT03 = 000010            119#    129
BIT04 = 000020            118#    128
BIT05 = 000040            117#    127
BIT06 = 000100            116#    126
BIT07 = 000200            115#    125
BIT08 = 000400            114#    124
BIT09 = 001000            113#    123
BIT1  = 000002            131#    781     936
BIT10 = 002000            112#
BIT11 = 004000            111#
BIT12 = 010000            110#
BIT13 = 020000            109#
BIT14 = 040000            108#   1726    1740
BIT15 = 100000            107#    512    1101    1534
BIT2  = 000004            130#    470     507    1405
BIT3  = 000010            129#
BIT4  = 000020            128#    837    1461    1495    1522    2809
BIT5  = 000040            127#
BIT6  = 000100            126#   1033    1063    1717    1722    1728
BIT7  = 000200            125#    490    1108    1166    1527
BIT8  = 000400            124#   1042    1068    1146    1426    1753
BIT9  = 001000            123#    763    1031    1325    1327    1329
BPTVEC= 000014            139#
BRONPA  010140           1591#   1599#   1601#
BRONTE  002646            672#    679#    680#
BRONTR  011064           1827#   1834#   1837#
BSTAT   007652            629    1434#   1435    1437    1520#   1548#   1557
BUFADR  015344           1126#   1127#   1131#   1134#   1135    1600    1694    2723#
BYTECN  006226            714#    721#   1106#   1123#   1167    1183#   1228#   1344    1354#
CHARCT  010706            667#   1748    1751#   1752#   1768#   1782#
CKBYTE  005430           1119#
CKSUM   005550           1110#   1120#   1128#   1129#   1130#   1131    1132#   1133#   1134    1138#   1198
COMMAD  003644            768     843#    901
CONT1   004210            908     917#
CONT10  005204           1032    1063#
CONT11  005104           1040    1045#   1071
CONT12  002376            609     613#
CONT13  003136            738     740#
CONT14  003472            808     810#
CONT15  004472            967     969#
CONT16  004612            991     993#
CONT17  005114           1046    1048#
CONT2   004260            918     927#
CONT20  006000           1181    1183#
CONT21  007032           1394    1396#
```

```
CONT22  010642           1760    1769#
CONT3   006204           1190    1221#
CONT4   003126            733     737#
CONT5   003164            741     744     746#
CONT6   004724           1003    1013#
CONT7   004710           1010#   1027
CRCBAD  015610           2748#
CRCER   004402            944     952#
CRCLOG  015604           2746#
CRETRY  015560            894#    971#   2732#
DATABY  010176           1611#   1620#   1630#   1639#   1649#   1658#   1668#   1670#   1679#   1688#   1692    1693
DATACK  003750            863#    895#    955#   1154    1186
DATACR  004550            954     958     983#
DATAER  005602           1122    1146#
DATALO  015624           2754#
DATARE  015552            891#   1210#   2729#
DATAO   010202           1602    1620#
DATAI   010214           1603    1630#
DBLLF   014172            586     747     819     913     923     977    1007    1024    1056    1217    1526    1533    2543#
                         2821    2837    2841    2864    2902    2983    3016
DBLSP   014001            560     573     578    1170    1174    1494    1505    1511    2502#   2856    2873    2881    2890
                         2898    2932    2937    2965    2970    3045    3055
DDCHK   005014            927    1031#
DDERR   005066           1042#   1073
DDISP = 177570             54#    339
DDMIS   015630           2756#
DDRETR  015550            619#    632#    890#   1050#   2728#
DELUNI  007654            752    1476    1522#
DISPLA  001216            339#    438*
DISPRE  000174            184#    438
DONE    010466           1738#   1746
DONEBI= 000040            155#   1350    1371    1410
DONECK  006714            469     502     621     769     902    1367#   1429    1464    1543
DRVSWP  006336            686    1275#
DRY1    001572            495     501#    506     515
DSWR  = 177570             53#    338
DTESTP  001212            337#    491     493     496     516     518     523     763    1031    1325    1327#   1329#
DOLOAD  013575            483    2475#
EBIE  = 000103            157#   1116
EMPBUF  005340            956    1103#   1143    1213    1224
EMPDON  005010           1111    1186#
EMPER   005012           1112    1140#
EMPFLA  005024           1117#   1124    1184
EMTVEC= 000030            142#
ERCNTR  006234            957    1107#   1147#   1152    1189    1194    1231#
ERCODE  015672           1440    2775#   2986
ERDUMP  017622            208    1538    2820#
ERRVEC= 000004            135#
ERRRT   003746            670#    772     862#   1017#
EXIT    006224           1105    1222    1226#
FBIE  = 000101            156#    718
FILLBU  002770            711#    727     745
FILLDO  003220            712     759#
FILLER  003046            713     724#
```

```
FILLFL  003030            719#    722
FIRST   001202            239#    240     568     570     575    2020   2022*   2025    2028    2042
FLOATD  010224           1604    1639#
FLOATI  010256           1605    1658#
FUNCTI  003742            762*    765*    849*    850     860#    898*   1454    1461    1522
GETPAT  010072            697     872    1084    1240    1257    1276   1303    1591#   1772
GETSEC  012062            708     889    2039#
GETTRA  011020            700     875    1087    1243    1260    1279   1309    1317    1826#
GETUNI  010374            699     874    1086    1242    1259    1278   1282    1308    1717#
GNS   = ******  U        2281    2282    2283    2284    2285    2286
GOODBY  006232           1175*   1177    1230#
HARDSK  003520            811     813     816#
HCNTR1  006772           1337*   1346*   1360*   1369*   1375*   1383#
HCNTR2  006774           1370*   1376*   1377    1384#
HCRCBA  015650           2764#
HCRCLO  015644           2762#
HDATAL  015664           2770#
HDDLOG  015670           2772#
HDMOVE  016420           1918    2791#   3022
HERE    010704           1771    1774    1780#
HLT1    001514            487#
HLT10   004470            968#
HLT11   004546            981#
HLT12   004610            992#
HLT13   005112           1047#
HLT14   005776           1182#
HLT15   007030           1395#
HLT16   010634           1767#
HLT17   021252           3072#
HLT20   006770           1380#
HLT21   006712           1363#
HLT3    002514            635#
HLT4    002044            546#
HLT5    002370            611#
HLT6    003134            739#
HLT7    003470            809#
HOME    002404            475     616#    814
HPARLO  015566            541*    746*   2738#   2876    2878
HRDLOG  015554           2766#
HSEKLO  015640           2760#
HWRTLO  015660           2768#
ICOND   002306            597#    666    2830
ID    = 001201            238#    562     566    1792*   1796    1872   1907    1929    1932#
IDCOMP  011606           1981#   1984
IDNEXT  011316           1904    1907#
INITSE  011762            707     888    2020#
INITTR  010710            698     873    1085    1241    1258    1277   1307    1316    1789#
INTER   015574           1388*   2741#   2893    2895
INTLEA  012056           2031*   2036#   2043    2044*
INTSER  007036            441    1396    1402#
INTVEC  001204            261#    440     465
IOTVEC= 000020            140#
LAST  = 001203            240#    580     584    2023*   2024    2040
LASTTR  011466           1944    1947#
```

```
LOAD    010344          1621    1644    1669    1689    1692#
LOGICA  010674           178    1776#
MABELL  014013          1758    2510#
MASTAT  014142          1550    2537#
MBADCR  014576           964     978    2615#   2915    2949
MBSTAT  014156          1556    2540#
MCODES  015124          2681#   2984
MCOLMU  013550          1165    2470#
MCRC    014670           988    2631#   2916    2950
MCRLF   014004           539     544     593     597     607     736     751     831     916     926     965     979     989
                        1010    1044    1059    1061    1161    1163    1179    1203    1220    1392    1416    1424    1446
                        1452    1459    1475    1549    1561    1750    1761    1766    2504#   2847    2867    2875    2884
                        2892    2900    2907    2941    2943    2975    3007    3067
MDDER   014547          1058    2608#   2945
MDDMIS  013440          1041    2449#   2911
MDELET  014060          1525    1532    2523#
MDMOD   013461          1162    1219    2454#   2912    2946
MDUMP   015007          2654#   2828
MEMPTY  014334          1141    1142    2570#
MFOP    014007          1757    2506#
MRCT    014265          1193    2761#
MEREOP  014011          1755    2508#
MERHEA  014015          1447    2512#
MERS    015021          1200    2657#   2906
MFIL    014330           725     726    2567#
MFIRST  014452           574    2591#
MGOOD   015254          1202    2701#
MHALT1  014656           980    2628#
MHALT3  014645           634    2625#
MHALT4  014634           545    2622#
MHUNG   014364          1362    1379    2577#
MICON   014351           598    2574#
MID     014446           561    2589#
MINIT1  014467           472    2587#
MINIT2  014521           509    2603#
MINTER  014212          1391    2551#   2899
MINTVE  014102           464    2527#
MLAST   014461           579    2594#
MNODRY  013750           610    1474    2495#
MNOFLA  014766          1423    2650#   2891
MNONE   015243          2698#   2981    3015
MNONST  014400           592    2530#
MNOPAS  015057          2667#   2840
MOD     014442           556    2587#
MPAR    014751           538     543     735     750    2646#   2874    2883
MPASS   015074          1448    1762    2671#   2843
MPAT    014176          1500    2545#
MPRES   013521           827    2463#
MREAD   014626           915     939     940     947     948    1009    2620#   2855    2914    2948
MREC    014561           914     924    2611#
MRESTA  015043          2663#   2833
MREV    015315           453    2711#   2829
MRXCS   014121           454     531    1453    2531#
MRXDB   014131           459    2534#
```

```
                    1743*  1746*  1772*  1776*  1780*  1803*  1921*  1979*  2010*  2032*  2045*  2086*  2093*
                    2100*  2114*  2116*  2320   2343*  2345*  2383*  2388*  2432*  2437*  2440*  2813*  2830*
                    2858*  2859*  2862*  2863*  3038*  3039*  3043*  3044*
PCONT   003160       727*   745*   786*   941*  1143*
PIRQ  = 177772        52#
PIRQVE= 000240       146#
PRESCH  011646      1986   1989#
PRESTR  011126       637*   828   1826*  1848*  1859   1903   1962   1989
PRETRY  015556       710*   742*   743    759*   892*  2731#
PR0   = 000000        69#
PR1   = 000040        70#
PR2   = 000100        71#
PR3   = 000140        72#
PR4   = 000200        73#   162#   715
PR5   = 000240        74#
PR6   = 000300        75#
PR7   = 000340        76#   163#   429    662    1338
PS    = 177776        49#    50
PSW   = 177776        50#
PTYP1   003112       725*   734#   784*   939*  1141*
PTYP2   003200       726*   749#   785*   940*  1142*
PWRVEC= 000024       141*  2292*  2293*  2301*  2316*  2317*
P2RETR  015554       476*   536*   618*  1427*  2730#
RANDAT  010324      1608   1687#  1690
RANGEN  011666      1687   1979   1994#
RANJM   011760      1688   1990*  1981   1983*  1985   1987*  1989   1991   2009*  2014#
RAN1    011754       443*  1995   2001*  2006   2012#
RAN2    011756       444*  1996   2003   2008*  2013#
RDAFTW  003744       671*   861#   871*  1002   1083*  1275*  1306*  1314#
RDCHK   006236       684   1240#
RDCNTR  017574       899*   900*  2803#  2861
RDCODE  007760       628   1541#
RDDONE  004140       896    907#
RDER  = 000017       154#  1428   1542
RDERR   004302       897    935#
RDIE  = 000107       159#   898
RDLOG   015614      2750#
RDONLY  006276       685   1237#
RDRETR  015544       449    668*   907    912*   995*  1001*  2726#
RDOSTA= 000013       152#  1463
RD1STA= 000033       153#   501   1466
READ    004022       877    888#  1102   1261
READCH  005330      1089   1101#  1244   1281   1284   1311   1318
RECAL = 040001       161#   620
REREAD  004070       777    895#   941    946    973    997   1052   1212
RESREG= 104405      2286#  2387
RESTAR  002532       207    661#  1482   2321
RESTCN  017600       665*   2836#  2831   2834
RESVEC= 000010       136#
RETURN  005256      1034   1064   1074#
REWRIT  003226       760#   786
RTJUST  013324      2418#  2440   2859   2863   3039   3044
RTN     002530       617    638#
RXCS    001206       283#   455    473    501*   532    620*   626    718*   850*  1116*  1341   1350   1371
```

DZRXAD.P11 CROSS REFERENCE TABLE

```
                    1905*  1907*  1915   1933*  1938*  1943*  1946*  1949*  1964*  1967*  1991*
TBITVE= 000014       137*
TCRLF = 000000      2080   2119*
TEST    002052       518*   519*  520*   521*   522*   548*   673    675*   676    1507
TESTDR  001466       474    476*
TESTSE  002560       667*  1530
TEST7   006422       687   1303*
TEST7X  006434      1305*  1323   1328   1330
THREE   012060      2029   2037*  2039
THT   = 000011      2078   2118*
TKACC   015734       846   2788*  3021
TKVEC = 000060       144*
TPVEC = 000064       145*
TRAPVE= 000034       143*
TRCK    006606       719    851    853    1117   1337*
TRKCNT  011122       702*   878*  1090*  1245*  1262*  1285*  1312*  1319*  1799*  1800*  1801*  1846*  1888*
                    1889*  1892   1931*  1940   1947   1960*  1961*
TRKERR  003612       802    834*   952    1038   1069   1207
TRTVEC= 000014       138*
TSECTO  012054       852   1127   1130   1490   2028*  2029*  2035*  2039*  2040   2042*  2043*
TSTLIM  002054       550*   613
TSTMSG  002266       555    572    590*
TYPADR  007476      1043   1060   1164   1458   1485*
TYPCOD  010030      1460   1550*
TYPE  = 104400       453    454    459    464    472    483    509    531    538    539    542    543    544
                     545    556    560    561    573    574    578    579    586    592    593    597    598
                     601    604    605    607    610    634    734    735    736    747    748    749    750
                     751    805    819    820    821    826    827    831    913    914    915    916    923
                     924    925    926    964    965    977    978    979    980    988    989   1007   1008
                    1009   1010   1024   1025   1026   1041   1044   1056   1057   1058   1059   1061   1072
                    1161   1162   1163   1165   1170   1174   1179   1193   1197   1200   1202   1203   1217
                    1218   1219   1220   1362   1379   1391   1392   1415   1416   1423   1424   1446   1447
                    1448   1452   1453   1459   1474   1475   1480   1481   1485   1489   1494   1497   1499
                    1500   1505   1506   1511   1512   1524   1525   1526   1531   1532   1533   1549   1550
                    1555   1556   1561   1750   1755   1757   1758   1761   1762   1766   2083   2192   2281*
                    2318   2428   2821   2824   2825   2827   2828   2829   2833   2837   2840   2841   2843
                    2847   2853   2854   2855   2856   2860   2864   2867   2873   2874   2875   2891   2862
                    2883   2884   2830   2891   2932   2908   2939   2900   2902   2903   2904   2905   2906
                    2907   2932   2937   2939   2941   2943   2965   2970   2971   2973   2975   2980   2981
                    2983   2984   2997   3007   3014   3015   3016   3019   3036   3040   3045   3046   3054
                    3055   3064   3067
TYPERR  005714      1157   1160   1167*
TYPOC = 104401      2282*
TYPON = 104403      1178   2284*
TYPOS = 104402       456    461    466    533    1172   1455   1502   1508   1514   1552   1558   2283*  3004
TYPSEL  007572       606   1498   1500*
TYPTRK  020756      2989   3012*
UKNINT  015572      1412*  2740*
UNITSE  010464       477*   490*   498*   512*   599    602    608    837    849   1042*  1068*  1146*  1322
                    1426*  1495   1527*  1528   1534*  1535   1717   1719   1721*  1722*  1724   1726   1728*
                    1729*  1734*  1738   1740   1753   1769*  2809
UNXDO   015634      2758*
UOTRK   017104       836   2796*  3023
UILOG   017604       800    817    910    920    960    975    984    999   1005   1021   1036   1054   1066
```

```
                        1205    1215    2809#
U1TRK   017336           833    2799#   3024
WRITE   002744           701     707#    876    1088    1280    1283    1310
WRTDON  003310           760     772#
WRTER   003336           761     780#
WRTIE =  000105           158#    762
WRTLOG  015620          2752#
WRTONL  002704           681     697#
WRTRD   003752           682     871#
WRTSEK  003376           782     790#
WTCNTR  017570           766#    767#   2802#   2857
WTDOIE=  000115           160#    765
WTROCK  005260           683    1083#
WTRETR  015546           669#    917     922#   1015#   1023#   2727#
XFRBYT  003034           720#
XHOME   002522           627     631     637#
XID     011132          1795#   1796#   1799    1850#   1859    1875    1890    1933    1945#   1946    1964    1965#   1981
                        1983
XOD     011130          1797#   1798#   1800    1849#   1891    1938    1942#   1943    1985    1987
XPATGE  010232          1640#   1659
XRDCHK  006242          1241#   1248
XRDONL  006302          1258#   1265
XREAD   004026           889#    932    1223
XRETUR  011504          1948    1950#
XSTART  002554           528     666#
XWRITE  002750           708#    778
XWRTON  002710           698#    705
XWRTRD  003762           873#    881
XWTROC  005270          1085#   1093
XXPATG  010302          1669#   1671    1680
ZCRCBA  015606           959    2747#
ZCRCLO  015602           983    2745#
ZDATAL  015622          1204    2753#
ZDDMIS  015626          1035    2755#
ZERO    021256          2909    2944    3075#
ZHCRCB  015646           974    2763#
ZHCRCL  015642           998    2761#
ZHDATA  015662          1214    2769#
ZHDDLO  015666          1053    2771#
ZHRDLO  015652          1004    2765#
ZHSEKL  015636           816    2759#   2952
ZHWRTL  015656          1020    2767#
ZRDLOG  015612           909    2749#
ZSEKLO  015576           799    2743#   2918
ZUNXDO  015632          1065    2757#
ZWRTLO  015616           919    2751#
$CHARC  012332          2094#   2101    2110#   2115#
$CRLF   012347          2084    2128#
$D82D   013130          2343    2361#   2858    2862    3038    3043
$DECVL  013310          2363    2409#
$FILLC  012344          2087    2125#
$FILLS  012343          2124#
$HD   =  000003            20      21
$ILLUP  013062          2292    2323#
```

```
$LF     012351          2129#
$MAIL = ****** U        2072
$NULL   012342          2089    2123#
$OCNT   012576          2164*   2193*   2206#
$OMODE  012600          2159*   2163*   2168    2171*   2182*   2208#
$POWER  013070          1481    2319    2326#
$PWRAD  013056          2321#
$PWRDN  012734           170    2292#   2316
$P. MG  013052          2319#
$PWRUP  013002          2301    2306#
$QUES   012346          2127#
$RDCHR= ****** U        2285
$RDDEC= ****** U        2285
$RDLIN= ****** U        2285
$RDOCT= ****** U        2285
$RESRE  012640          2243#   2286
$R2A  = ****** U        2287
$SAVRE  012602          2227#   2285
$SAVR6  013066          2300*   2306    2307*   2308*   2325#
$SB2D   013100          2341#   2437
$SWR  = 160000            20      21#   2322
$TN   = 000001            20#
$TNPWR  013240          2368    2369    2389#
$TPB    012340          2107*   2122#
$TPFLG  012345          2066    2126#
$TPS    012336          2105    2121#
$TRAP   012676           174    2264#
$TRP  = 000006          2272#   2282#   2283#   2284#   2285#   2286#   2287#
$TRPAD  012720          2269    2280#
$TYPBN= ****** U        2285
$TYPDS= ****** U        2285
$TYPE   012122          2066#   2272    2281
$TYPEC  012266          2086    2093    2100    2105#   2106
$TYPEX  012334          2111    2113    2116#
$TYPOC  012400          2162#   2282
$TYPON  012414          2161    2164#   2284
$TYPOS  012354          2157#   2283
$OFILL  012577          2158*   2162*   2172    2207#
.     = 021260           166#    168     169#    173#    177#    180#    183#    202#    236#    680    1601    1837    2121
                        2122    2123    2124    2125    2126    2127    2128    2129    2130#   2303    2324    2409#   2723#
                        2775#   2788#   2791#   2796#   2799#
```

```
COMMEN    147#
ENDCOM    147#
ERROR      47#
ESCAPE    147#
MULT      147#
NEWTST    147#
POP       147#   2248    2310
PUSH      147#   2228    2294
SCOPE      48#
SETTRA   2272#   2282    2283    2284    2285    2286
SETUP     147#
SKIP      147#
SLASH     147#
SPACE     147#
STARS       6#    147#    188     211     243     263     281     341     406     424     640     659     691     757     865
          883    1078    1096    1233    1251    1269    1291    1333    1398    1565    1587    1615    1624    1633    1652
         1661    1673    1682    1702    1715    1784    1807    1824    1853    1867    1880    1897    1910    1924    1952
         1972    2016    2048    2131    2209    2255    2287    2329    2347    2411    2718    2818
TRMTRP   2272#
TYPBIN    147#
TYPDEC    147#
TYPNUM    147#
TYPOCS    147#    455     460     465     532    1454    1501    1507    1513    1551    1557    3003
TYPOCT    147#
TYPTXT    147#
$$ESCA    147#
$$NEWT    147#
$$SET    2272#   2282    2283    2284    2285    2286
$$SKIP    147#
.EQUAT      5#     42
.HEADE      5#     10
.SDB2D      6#   2347
.SPOWE      6#   2287
.SSAVE      6#   2209
.SSB2D      6#   2329
.STRAP      6#   2255
.STYPE      5#   2048
.STYPO      5#   2131
```

DZRXAD.P11 CROSS REFERENCE TABLE

```
          1221  1245  1262  1285  1312  1319  1597  1832  1877  1888  1945  1960  1965  2383
DECB      2091  2094  2182  2193
EM*         47
HALT       168   487   546   611   635   739   809   968   981   992  1047  1182  1363  1380  1395
          1767  2068  2302  2323  3072
INC        530   541   594   665   721   731   746   801   818   840   911   921   955   961   976
           985  1000  1006  1022  1037  1055  1067  1123  1147  1176  1183  1206  1216  1346  1360
          1388  1412  1420  1443  1643  1747  1752  1801  1864  1931  1942  2022  2026  2044  2188
          2196  2308  2375  2928  2992  3008  3068
INCB      2114
IOT         48
JMP        206   207   208   528   547   612   636   681   682   683   684   685   686   687   728
           745   752   770   777   778   787   815   903   942   973   997  1012  1019  1052  1062
          1105  1144  1184  1213  1224  1407  1492  1530  1538  1602  1603  1604  1605  1606  1607
          1608  1838  1839  1840  1841  1842  1843  1844
JSR        469   475   495   502   505   515   555   558   563   572   576   581   606   613   621
           624   628   666   697   698   699   700   701   704   707   708   711   719   768   769
           797   800   802   806   814   817   822   829   851   853   872   873   874   875   876
           877   880   888   889   901   902   910   920   927   949   952   956   960   975   984
           999  1005  1021  1036  1038  1043  1054  1060  1066  1069  1084  1085  1086  1087  1088
          1089  1092  1102  1109  1117  1164  1168  1195  1205  1207  1212  1215  1223  1240  1241
          1242  1243  1244  1247  1257  1258  1259  1260  1261  1264  1276  1277  1278  1279  1290
          1281  1282  1283  1294  1297  1303  1307  1308  1309  1310  1311  1316  1317  1318  1321
          1396  1429  1432  1450  1458  1460  1464  1470  1476  1487  1492  1543  1546  1621  1644
          1669  1687  1699  1746  1764  1772  1776  1979  2086  2093  2100  2343  2437  2440  2930
          2935  2945  2958  2959  2962  2963  2871  2879  2888  2836  2930  2935  2963  2968  2935
          3034  3038  3039  3043  3044  3050  3057
MOV        428   429   430   432   437   438   439   440   441   442   443   444   445   449   450
           455   460   465   476   501   516   518   523   532   618   619   620   637   661   662
           663   668   669   675   676   709   710   712   713   715   716   718   725   726   727
           759   760   761   784   785   786   790   795   796   799   816   828   834   843   850
           856   890   891   892   893   894   896   897   909   912   919   922   939   940   941
           946   947   948   959   974   983   998  1001  1004  1020  1023  1035  1053  1065  1111
          1112  1114  1116  1128  1135  1141  1142  1143  1167  1171  1177  1194  1204  1214  1304
          1305  1315  1324  1338  1339  1358  1367  1370  1373  1427  1428  1437  1449  1454  1463
          1466  1486  1490  1501  1507  1513  1542  1551  1557  1595  1596  1600  1641  1751  1763
          1773  1799  1830  1831  1890  1915  1991  1994  2001  2003  2008  2009  2031  2070  2071
          2075  2089  2157  2165  2166  2167  2173  2180  2198  2199  2200  2201  2202  2228  2229
          2230  2231  2232  2233  2234  2235  2236  2237  2244  2245  2246  2247  2248  2249  2250
          2251  2252  2253  2264  2265  2269  2292  2293  2294  2295  2296  2297  2298  2299  2300
          2301  2306  2310  2311  2312  2313  2314  2315  2316  2317  2320  2341  2342  2344  2362
          2363  2364  2365  2366  2367  2368  2369  2418  2419  2420  2430  2431  2436  2820  2834
          2844  2857  2861  2870  2878  2887  2895  2909  2910  2911  2912  2913  2914  2915  2916
          2917  2918  2929  2934  2938  2944  2945  2946  2947  2948  2949  2950  2951  2952  2962
          2967  2972  2985  2986  2994  2999  3003  3021  3022  3023  3024  3037  3042  3049  3056
MOVB       557   562   575   580   720   762   765   852   854   898  1119  1126  1127  1131  1134
          1175  1402  1434  1541  1548  1630  1639  1668  1679  1688  1693  1792  1794  1796  1798
          1826  1861  1875  1905  1907  1933  1938  1943  1946  1949  1964  1967  2023  2024  2028
          2042  2072  2099  2107  2158  2159  2162  2163  2164  2168  2171  2172  2191  2267  2332
          2425  2427
NEG       2169
NEGB      1133
NOP       1777  1778  1779
RESET     1775
```

DZRXAD.P11 CROSS REFERENCE TABLE

```
ROL      524   525  1999  2000  2175  2177  2178  2179  2181
ROLB    1649
ROR      520   521   522  1438  1439  2004  2005
RTI      431   664   717   857  1115  1340  1359  1417  2077  2203  2238  2254  2322
RTS      513   540   587   595   614   638   776   824   832   841   858   930   933  1074  1136
        1188  1226  1343  1374  1419  1478  1517  1562  1696  1698  1723  1730  1743  1780  1803
        1921  2010  2032  2045  2116  2270  2345  2388  2432  2441  2813
SBC     2372
SEC     1640  1648
SUB     1800  1891  1892  1983  2025  2029  2371  2373
SWAB     526
TRAP    2272  2282  2283  2284  2285  2286
TST      473   493   551   568   590   602   608   626   673   724   737   743   772   780   807
         928   935   953   957   966   990  1002  1045  1103  1140  1154  1180  1186  1189  1356
        1377  1393  1403  1435  1528  1593  1646  1697  1724  1738  1748  1790  1828  2020  2074
        2082  2103  2186  2266  2822  2831  2838  2848  2868  2876  2895  2893  2919  2921  2923
        2925  2926  2933  2953  2955  2957  2959  2960  2966  2978  2990  3009  3012  3017  3026
        3028  3031  3032  3047  3052  3061  3062  3066
TSTB     488   510   553   599  1156  1159  1198  1341  1472  1535  1719  1872  1885  1929  1935
        1957  2066  2105  2421
.ASCII  2127  2475  2482
.ASCIZ  2128  2129  2326  2444  2449  2454  2459  2463  2467  2470  2490  2495  2499  2502  2504
        2506  2508  2510  2512  2517  2520  2523  2527  2531  2534  2537  2540  2543  2545  2547
        2549  2551  2557  2561  2567  2570  2574  2577  2580  2587  2589  2591  2594  2597  2603
        2608  2611  2615  2620  2622  2625  2628  2631  2636  2640  2643  2646  2650  2654  2657
        2660  2663  2667  2671  2674  2677  2679  2681  2687  2698  2701  2703  2707  2711
.BLKB   2409  2723
.BLKW   2775  2788  2791  2796  2799
.BYTE    457   458   462   463   467   468   534   535  1456  1457  1503  1504  1509  1510  1515
        1516  1553  1554  1559  1560  2123  2124  2125  2126  2204  2205  2206  2207  3005  3006
.ENABL     4
.END    3077
.ENDC     15    47   133   147   189   212   240   264   282   342   407   425   458   459   463
         464   468   469   535   536   641   660   692   758   866   884  1079  1097  1234  1252
        1270  1292  1334  1399  1457  1458  1504  1505  1510  1511  1516  1517  1554  1555  1560
        1561  1566  1588  1616  1625  1634  1653  1662  1674  1683  1703  1716  1785  1808  1825
        1854  1868  1881  1898  1911  1925  1953  1973  2017  2049  2072  2132  2210  2256  2265
        2268  2281  2282  2283  2284  2285  2286  2287  2288  2300  2310  2320  2322  2329  2330
        2348  2412  2719  2819  3006  3007
.EQUIV    47    48    50    65    66    95    96    97    98    99   100   101   102   103   104
         123   124   125   126   127   128   129   130   131   132
.EVEN   2130  2328  2716
.IF       11    45   105   133   188   211   243   263   281   341   406   424   457   458   462
         463   467   468   534   535   640   659   691   757   865   883  1078  1096  1233  1251
        1269  1291  1333  1398  1456  1457  1503  1504  1509  1510  1515  1516  1553  1554  1559
        1560  1565  1587  1615  1624  1633  1652  1661  1673  1682  1702  1715  1784  1807  1824
        1853  1867  1880  1897  1910  1924  1942  1972  2016  2048  2072  2131  2209  2255  2264
        2268  2272  2282  2283  2284  2285  2286  2287  2300  2310  2318  2320  2322  2326  2329
        2347  2411  2718  2818  3005  3006
.IFF      47   189   212   244   264   282   342   407   425   457   458   462   463   468   535
         641   660   692   758   866   884  1079  1097  1234  1252  1270  1292  1334  1399  1457
        1504  1510  1516  1554  1560  1566  1598  1616  1625  1634  1653  1662  1674  1683  1703
        1716  1785  1808  1825  1854  1868  1881  1898  1911  1925  1953  1973  2017  2049  2132
        2210  2256  2265  2288  2320  2330  2348  2412  2719  2819  3006  3007
```

DZRXAD.P11 CROSS REFERENCE TABLE

```
.IIF      10    15    20    21   456   461   466   533  1455  1502  1508  1514  1552  1558  2121
        2122  2123  2124  2125  2126  2127  2128  2129  2130  2281  2282  2283  2284  2285  2286
        3004
.IRP    2228  2248  2294  2310
.LIST      3   147   168  2272  2281  2282  2283  2284  2285  2286  2287
.MACRO  2272
.MCALL     5     6   147
.NLIST     2   147   168  2272  2281  2282  2283  2284  2285  2286  2287
.REPT    168
.SBTTL    43  2050  2133  2211  2257  2273  2289  2331  2349
.TITLE    10
.WORD    167   181   337   338   339  1173  2115  2121  2122  2208  2319  2321  2346  2384  2802
        2803
```

ERRORS DETECTED: 0

```
*DZRXAD,DZRXAD/CRF/SOL=DZRXAD
RUN-TIME: 37 27 5 SECONDS
CORE USED:  12K
```

Spooler runtime 23 Seconds, 143 KCS, 918 disk reads, 1 disk writes, 187 pages printed