N65-29802
(ACCESSION NUMBER)
39
(PAGES)
(NASA CR OR TMX OR AD NUMBER)
(THRU)
1
(CODE)
08
(CATEGORY)
FACILITY FORM 602

X-545-65-115

NASA TM X-55947

AN IMPROVED APPROACH TO TRACE ROUTINES

FEBRUARY 1965

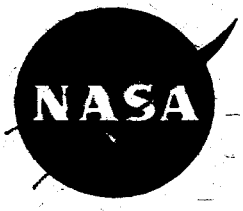
GPO PRICE $ _____

CFSTI PRICE(S) $ _____

Hard copy (HC) 3.00

Microfiche (MF) .50

ff 653 July 65



GODDARD SPACE FLIGHT CENTER
GREENBELT, MARYLAND

X-545-65-115

AN IMPROVED APPROACH TO TRACE ROUTINES

Eugene I. Grunby
Data Processing Branch
Data Systems Division

Goddard Space Flight Center
Greenbelt, Maryland

CONTENTS

PART 1

AN IMPROVED APPROACH TO TRACE ROUTINES

INTRODUCTION

Analyzing the flow of executed computer programs is among the most fundamental and essential techniques of debugging. With increasing sophistication in both computer hardware and software determining program flow is becoming progressively more difficult. Complicating factors are address indexing, indirect addressing, automatic allocation of relocatable subroutines, a wealth of resident monitor and library routines, chaining, and both manual and automatic overlay facilities.

The classic approach to unwinding these factors and presenting useful diagnostic information to the program has been and will continue to be — at least for the foreseeable future - the use of trace programs. Through the use of hardware interrupts and interpretive software routines, changes in sequential addressing of computer programs by jumps (branches) and tests (compares) can be detected, interpreted, and reported to the programmer.

Problems

Despite the value of trace routines, several serious drawbacks have deterred programmers from their use. Perhaps the worst of these are considerably extended execution time and overabundance of printed diagnostic material. Among the secondary problems is the inability of users to provide "own-coding" subroutines for extended usage.

This document proposes a solution to these problems. It also describes usage of the trace routine embodying these improvements for the Univac 1107.

Option Control

Increasing the speed and efficiency of trace programs can be approached on two fronts: first, application of utmost skill and ingenuity in developing the routine; second, allowing flexible control of option-subroutines to eliminate unwanted processing. The former approach will not be discussed because it is ultimately limited by hardware; the latter approach however, is limited only by imagination.

When developing a routine, a systems analyst first asks, "What should it do?" Partial answers come from the historic solutions to similar problems; further answers are obtained by polling likely users of the routine for recommendations. Generally, the analyst cannot obtain agreement without devising a system encompassing all options. Such a task is burdensome not only to the developer, but to the user as well. Furthermore, it defies speed and efficiency.

Enigmatically a solution to option control is to eliminate all but one option: that is, to allow the use of arbitrary externally defined subroutines which are callable by the trace routine. For discussion purposes, such a subroutine will be termed a subservient.

The trace routine can be written to leave the same basic information available to all subservients. Normally this will be the origin and destination addresses representing a change in sequential addressing, and will be called a FROM-TO address pair. In this manner all subservients can be independent. It is now possible to construct a basic set of subservients, each performing a unique function. In addition, any subservient can be called by another, so that a composite of functions can be selected to fulfill a required task.

The advantage of this procedure from a user's standpoint is that he himself can select a subservient for his problem. In the simplest case, he can request a single subservient made available on a standard library; in this instance, no coding is required. In a more advanced situation, he may wish to construct his own subservient, consisting of imbedded calls to any number of library subservients. Thus, only those required components are executed. Also, no option-testing logic is necessary, as the user has coded these options into his own subservient.

Minimization of Printing

When using most print options in a trace program, the user often finds himself surrounded with paper. Also, it is likely that exaggerated execution time or high print load has prevented him from reaching the conditions he wants traced. Even if these conditions were encountered, mechanical problems of shuffling paper may discourage him from further pursuits. Circumstances such as these often cause a trace routine to be relegated to a position of secondary importance.

An initial step to an effective solution is to eliminate printing for all but the most essential information, the FROM-TO address pairs. Such a decision is not too severe because the ability to construct and nest subservients allows the user to reconstruct reports of more exacting information. Also, as will be favored by some computer installations, this approach places the heaviest burden on those programmers requiring the most demanding printouts, but does not penalize the average user. Unfortunately, adoption of this initial step is not entirely adequate in resolving the problem of sheer paper volume.

If it were possible, the most desirable solution would be to avoid printing completely. Perhaps the next best alternative is to retain trace information in a core table for selective printing at the end of the job. In reality, however, few tables would be extensive enough to retain significant data; core would speedily become exhausted.

Despite apparent problems development of an adequate core table is still practicable. By storing cyclically in the table — overlaying from top to bottom — a fixed amount of data is maintained. In addition, only the most recent address pairs are left available because predecessors for each position in the table have been overstored. This constant updating will continue until tracing is terminated by user request, normal end of job, or abnormal end of job. The import of this data at abnormal completion is obvious.

To help offset limitations imposed by a cyclic table, options can be provided which allow the user to establish the length of this table. Also, enhancements can be provided for efficient use of the table: for example, a repeat count for iterated address pairs can optimize storage by compressing the data.

Finally, it is necessary to print the table. As this now occurs after program execution, various post mortem routines are usually available in utility packages to provide this core dump. Characteristically, such routines maximize output on the printed line, provide reasonable options for format control, and operate efficiently. A combination like this is usually more than adequate for programmer needs.

Extended Usage

Knowing all subservients are furnished with the same FROM-TO address pair information, the user can code and nest virtually any special trace techniques he desires. A variety of library and resident monitor subroutines is generally available to assist him. Among the most significant applications for the trace routine described in Section II are:

1. Dynamic dumping of arbitrarily selected memory locations at the time of each traced instruction

2. Continuous testing for jumps (branches) to illegal destinations, and execution of an error routine for these events

3. Continuous testing to detect which instruction or I/0 operation is overstoring specific cells in memory, and execution of an error routine for these events

4. Computation of the distribution of operation codes in an executing program, including - at the user's discretion - library and resident monitor routines

5. Continuous detection and logging of points where characteristic overflow, characteristic underflow, divide overflow, occur

6. Various combinations of the above practices

Note that no modifications, other than a call to the trace routine, are needed to apply these techniques to existing programs.

PART II

A TRACE ROUTINE FOR THE UNIVAC 1107

INTRODUCTION

The trace routine is a blend between a software and hardware trace. Software components are necessary to allow for restoring register R3 after a trace interrupt occurs. They also permit establishing the locations of all executed JMGI/MOJP (not provided by a trace interrupt) and of all instructions performing a skip. Hardware components of this routine are used solely for detecting that a jump instruction was executed, for obtaining the address to which control was to be transferred, and for returning control to the trace routine.

Sleuth II Calling Sequences

In general, the SLEUTH II calling sequence for initiating trace is:

```
LMJ     11, TRC$
 +      interrupt routine entry point, mode
 +      lower core limit, upper core limit
 +      address of trace storage table, length of table (dependent)
```

Parameters are interpreted as follows:

Mode: value Ø means trace jumps only
value 1 means trace jumps and skips only
value 2 means trace all instructions

Interrupt routine entry point: Starting line of routine to which control is transferred after detecting a condition determined by the selected mode. Upon entry to this routine, AØ will contain the address of the jump or skip; A1 will contain the address for the next sequential instruction. Entry to the interrupt routine is made by an SLJ instruction.

Upper and lower core limits: Core limits in which the trace is permitted to report to the interrupt routine.

Address of trace storage table, length of table: These parameters are needed only when using standard interrupt routines described later. These routines develop tables of trace information in user-defined core regions.

All tracing is terminated by the calling sequence

```
LMJ     11, TRCX$
```

Re-entry without reinitialization to the last trace function is by

```
LMJ     11, TRCR$
```

The procedures equivalent to these calls are:

```
T$RC    interrupt routine entry point, mode                    ;
        lower core limit, upper core limit                     ;
        address of trace storage table, length of table
T$RCX
T$RCR
```

The General Interrupt Routine Facility

By supplying a label to his own routine, the user can process all traced instructions that occur with either the FROM or TO address in his designated core limits. These addresses are available to him in AØ and A1 respectively. (The contents of AØ and A1 before trace interrupt occurred may be found in cells TRCAØ$ and TRCA1$) If registers other than AØ and A1 are used in the interrupt routine, they must be saved and restored by the user before returning control. Entry to the interrupt routine is made by an SLJ instruction.

The calling sequence for this routine is:

```
LMJ     11, TRC$
 +      routine name, tract mode of Ø, 1, or 2
 +      lower core limit, upper core limit
```

or

```
T$RC    routine name, mode                                     ;
        lower core limit, upper core limit
```

The TRCY$ Standard Interrupt Routine

This routine stores information into a user-specified storage table. Storage is made cyclically; that is, once the area is filled, overlaying will occur from the top. A single negative zero is placed after the last written value to signal a division between the old area and the new.

When either the traced FROM address or TO address is within the designated core limits, an address pair in the format +(FROM, TO) is inserted in

the table. However, if this address pair is identical to the previous pair, a repeat count, N, is developed and stored instead. This is identified by the format (-Ø, N).

If a call to TRCY$ is made via T$RC from within the designated core area, the address pair +(address of call, address of next sequential instruction) is generated.

The calling sequence for this routine is:

```
    LMJ       11, TRC$
  +TRCY$,     trace mode of Ø, 1, or 2
    +         lower core limit, upper core limit
    +         storage table address, table length
```

or

```
    T$RC      TRCY$, mode     lower core limit, upper core limit     ;
              storage table address, table length
```

The TRCP$ Standard Interrupt Routine

This routine sets a one-bit in the user's storage table to show a traced instruction in the designated core area was executed. The table is constructed so that each bit of a 36-bit word represents an address relative to the lower core limit address. If the indicated table length is exceeded, no storage will occur but the trace will be continued.

If a call to TRCP$ via T$RC is made from within the designated core area, a one-bit corresponding to the linkage address is set.

The calling sequence for this routine is:

```
    LMJ       11, TRC$
  +TRCP$,     trace mode of Ø, 1 or 2
    +         lower core limit, upper core limit
    +         storage table address, table length
```

or

```
    T$RC      TRCP$, mode     lower core limit, upper core limit     ;
              storage table address, table length
```

Nested Calls to Interrupt Routines

It is possible to define a general interrupt routine which can make use of the TRCY$ and TRCP$ routines. The following techniques must be used:

1. Before using TRCY$ or TRCP$, registers AØ and A1 should contain the values of the FROM and TO addresses supplied to the general interrupt routine by the trace program. Calls to the standard interrupt routines are made by SLJ TRCP$ or SLJ TRCY$. Registers AØ and A1 are altered before return from these calls.

2. Before initiating the trace routine, the required routines TRCY$ or TRCP$ must be initialized by making T$RC calls with the desired standard and the general interrupt routine names. The mode and the lower and upper core limits must be identical in all calls. Furthermore, the linkage initiating the general interrupt routine must be made last.

Example of Nested Interrupt Routine Usage

Problem: Find the origin of a jump to location Ø. Take the MERR$ exit if the instruction is executed. In the continued process of testing, record trace information with both TRCY$ and TRCP$.

Sample Solution:

```
          T$RC    TRCY$,1   Ø1ØØØØ,Ø1Ø44Ø    ;
                  CYBIN, 1ØØØ
          T$RC    TRCP$,1 Ø1ØØØØ,Ø1Ø44Ø     ;
                  PBIN, 8
          T$RC    UTRACE,1  Ø1ØØØØ,Ø1Ø44Ø
                      .
                      .
                      .

UTRACE    J       $
          JZ      13,MERR$    . TO address = Ø,  FROM in AØ
          S       12, SAVE12
          S       13, SAVE13
          SLJ     TRCY$
          L       12, SAVE12
          L       13, SAVE13
          SLJ     TRCP$
          J       UTRACE   . return to trace routine
SAVE12    +Ø
SAVE13    +Ø
CYBIN     RES     1ØØØ
PBIN      RES     8
```

Description: The first linkage initializes TRCY$ to store FROM-TO address pairs cyclically in table CYBIN. Only 1ØØØ address pairs will be

retained. The second linkage initializes TRCP$. The eight cells of 36 bits each in table PBIN is sufficient to contain all information for the designated core limits. The third linkage establishes the user-defined routine UTRACE. Since it is the last called, it takes precedence over the previous calls without disrupting initialization of TRCY$ and TRCP$.

At the interrupt for each traced jump instruction, the trace routine leaves the origin address of the jump in AØ and the destination address in A1. An SLJ to the users UTRACE is made. The user tests against A1 to see if it is zero. If so, the MERR$ exit is taken. AØ is unchanged and can be found in the automatic MERR$ thin-film dump. If A1 is non-zero, the user first saves AØ and A1 from future destruction by TRCY$. He calls TRCY$ and then restores AØ and A1 before calling TRCP$. There is no need to restore AØ and A1 before returning to the main trace program.

FORTRAN IV Calling Sequences

In general all trace-initiating calling sequences are in the form:

CALL NTRC (I, M, $X, $Y, BIN, L)

All FORIV trace routines are terminated by:

CALL NTRCX

All fields of NTRC must be provided. They are interpreted as follows: (the mnemonics used are not relevant).

Field I - a numeric value 1 identifies TRCY$
2 identifies TRCP$

Field M - a numeric value Ø traces jumps only
1 traces jumps and skips only
2 traces all instructions

Field $X - the statement number corresponding to the lower core limit

Field $Y - the statement number corresponding to the upper core limit

Field BIN - name of the array where storage will occur

Field L - the number of cells available in this array for storage

Action Taken on Incorrect Parameters

There is one error condition for which corrective actions are made and parameter interpretation continued. If the low core limit and upper core limit

are reversed, these are shifted. For all other parameter errors, the designated trace routine is not initiated. Instead the console message:

"**(N)TRC($) CALL ERROR FROM XXXXXX"

is typed, and control is restored to the calling program.

The following list summarizes these remaining error conditions:

1. Specified core limits do not encompass a portion of core memory ØØØØØØ-Ø17777.

2. Storage table start location is not in core.

3. Last address of core storage table exceeds location Ø17777.

4. The label identifying the interrupt routine entry point is undefined.

5. An illegal mode was specified.

6. Re-entry was attempted without prior use of the trace routine.

Trace Limitations

The limitations listed here, which may appear imposing, are normally quite remote from the mainstream of normal programs. Most references are to I/O interrupts which at worst only reduce the effectiveness of trace and do not introduce error conditions. The restrictions are:

1. Trace may not be initiated with interrupts disabled (i.e., usually during coding) unless the first instruction below the linkage is a PAIJ I/O interrupt. Without this instruction, the trace routine assumes that interrupts may be allowed and initially performs a AAIJ. No special coding is required to initiate the trace during "main-line coding".

2. The interrupt "location" MITRC$ is used by the trace. Any attempts to change MITRC$ will allow the change, terminate the trace, and return control to the main program without introducing errors. Such a condition results when an unsolicited "E" key-in or MERR$, MXXX$, MEXIT$ (or monitor equivalent) exit occurs.

3. The trace routine will not trace I/O interrupts. The real-time clock interrupt is in this class. However, once the interrupt has been processed, the trace will resume at the interrupted instruction.

4. All I/0 interrupts which interrupt the trace routine must restore control with interrupts allowed. (All monitor resident and library routines are written in this manner.)

5. Jumps to the trace routine which are themselves traced will first turn off the trace before being executed. Subsequent logic depends on the entry point.

6. The trace routine is vulnerable to being overwritten by erroneous programs. However, it cannot be overlaid by a MAP because it resides in independent core.

7. All registers other than AØ and A1 used in the general interrupt routine must be saved and restored before re-entry to the trace routine.

8. The address saved in parking register R3 after a traced repeated sequence will be the location of an execute remote instruction in the trace routine itself.

9. If an error interrupt occurs, the address of the offending instruction will be the same location mentioned in item 8 only if the interrupt location contains an EX instruction ultimately referencing an LMJ or SLJ.

10. If an error interrupt location contains an PAIJ or an EX ultimately referencing it, interrupts are prevented belatedly (that is, interrupts are allowed first by the trace routine but prevented before the next main program instruction is executed).

11. Error interrupt locations containing jumps must not require indirect addressing, B-loop modifications, or B-box incrementation. Should these conditions arise, the trace routine will not be able to recognize an error interrupt. Further errors may result.

12. Error interrupts occuring in the user's interrupt routine or from I/0 interrupt coding are not traced but proceed according to the logic of the monitor system.

Trace Program Logic

To provide complete information, an adequate trace program must contain a software trace to complement the 1107 hardware trace facility. The software component of this routine saves register R3 (normally used to hold memory lockout information) from destruction by the trace interrupt; prevents I/0

interrupts at critical phases, to assure no changes in R3 can be made during interrupt coding; obtains the address of all traced MOJP instructions (not provided by hardware); and detects all instructions executing skips.

The software trace component performs these duties in the following sequence:

1. Establishes address of next sequential instruction
2. Prevents all I/0 interrupts
3. Saves contents of register R3
4. Obtains operation code of next sequential instruction, and performs special functions on an exceptional set of operations
5. Restores all thin film registers used by the trace routine
6. Establishes hardware trace mode
7. Executes remotely on next sequential instruction
8. If the instruction was not a jump, sets switch to show if a skip was executed
9. Saves R3 in event it was changed by a load or store
10. Jumps to routines to process all non-jump instructions

The exceptional set of operations occurs principally because of limitations placed upon executing remotely or by the I/0 scheme used by the trace routine. Specifically, instructions SLJ and LMJ must not be executed remotely by the trace routine, for the return address would be incorrect. The instructions PAIJ and AAIJ must be sensed in order to inform later parts of the program to leave I/0 in the desired state. Also, AAIJ must not be performed then but must be delayed until the trace interrupt occurs. Tests for nesting of instruction EX must be made to assure that previously mentioned instructions are not obscured.

The hardware component of the trace routine performs the following tasks:

1. Restores R3, which was destroyed by trace interrupt

2. Restores I/0 interrupt state to that expected by traced program

3. Obtains the destination address of the traced jump, and performs special functions on an exceptional set

4. Determines if the jump was intended as an entry into trace routine itself; if so, allows it

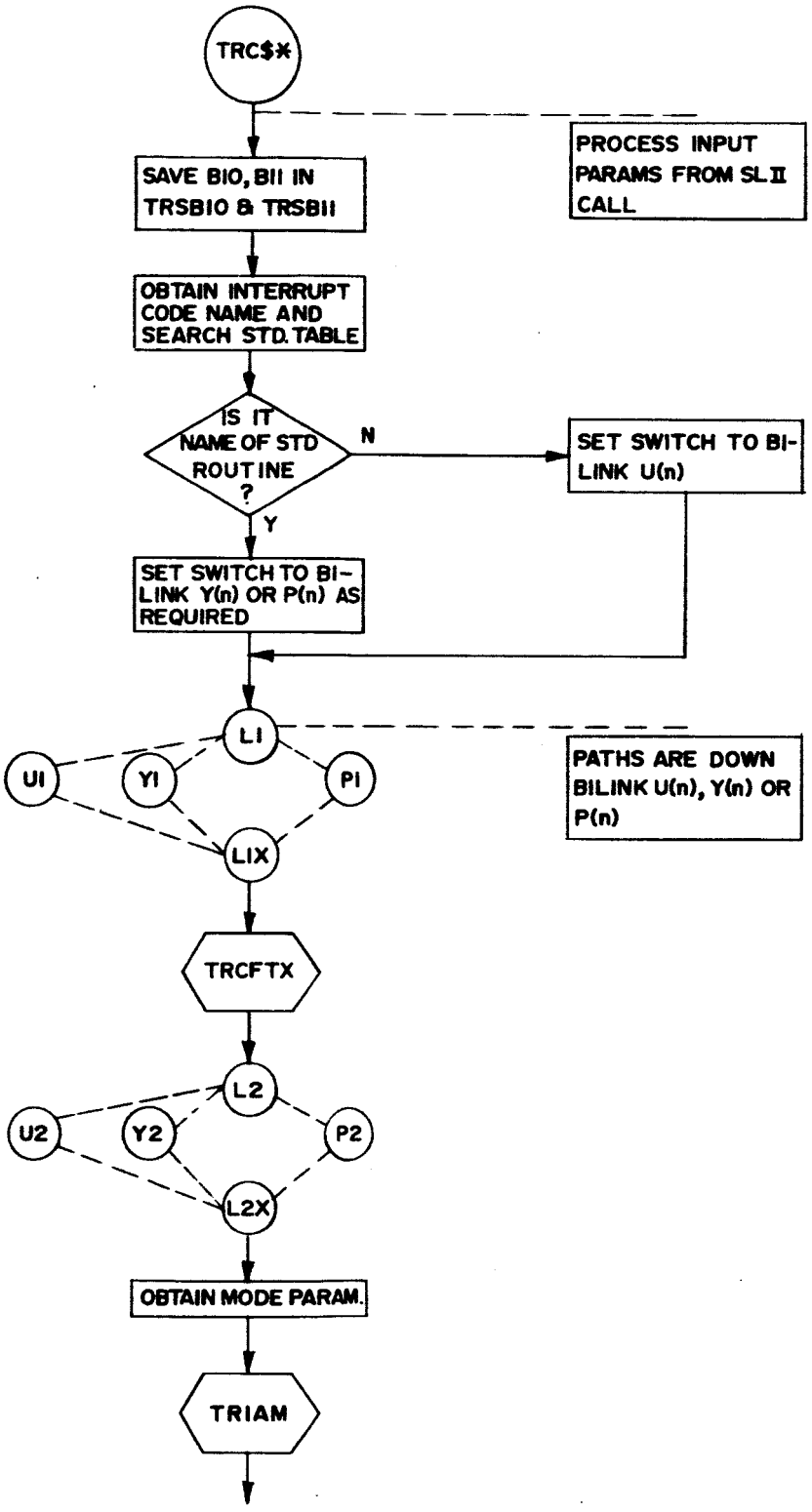
5. Determines if the jump origin or destination address is in user's core; if so, calls interrupt coding

6. Transfers control back to the software trace component

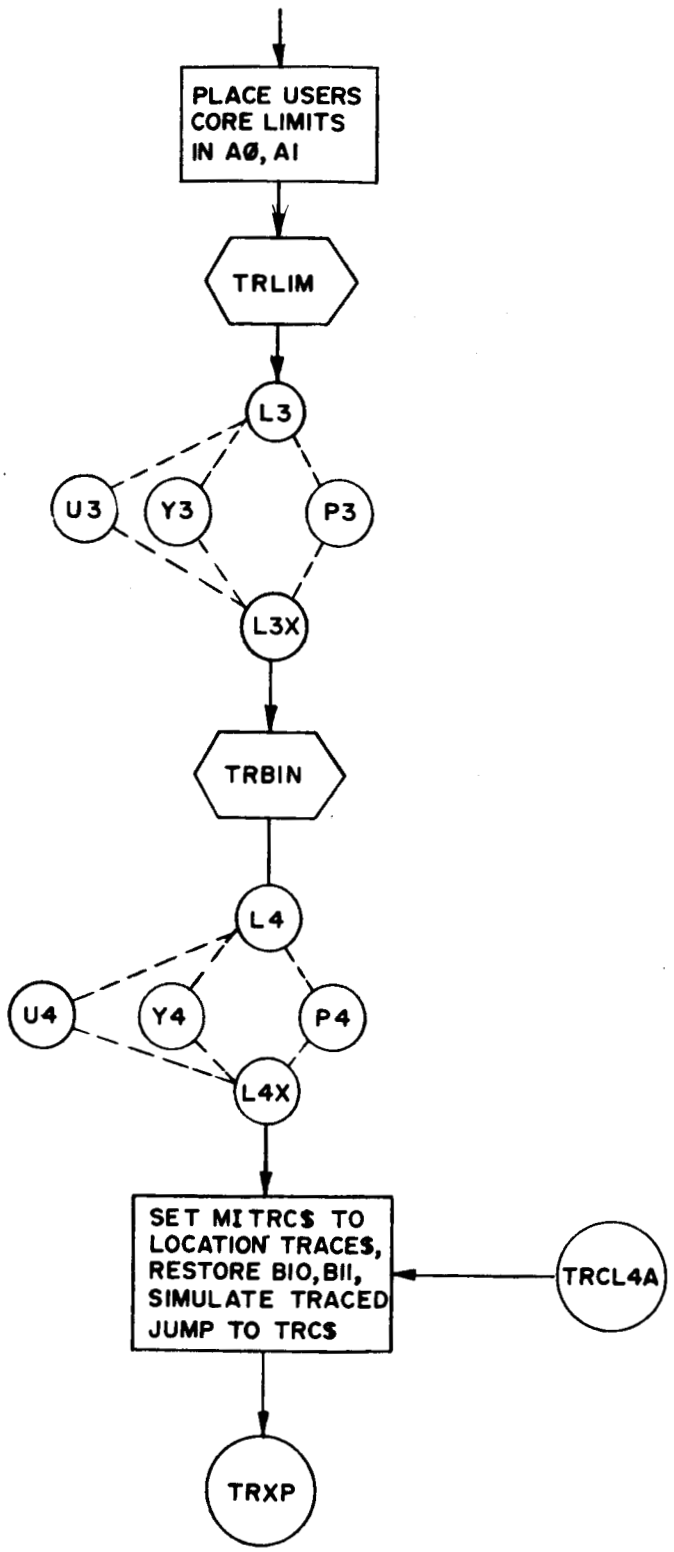
Among the exceptional destination addresses are MREA$ and MSEA$. If a jump to MSEA$ or MREA$ is made, and if a request is made to reset MIALL$ (all error actions) or to change MITRC$ alone (the trace location), all thin film is restored to that expected by the external program, and tracing is terminated.

The other exceptional addresses are concerned with error interrupt destinations. If a match of the intended destination of an instruction to the BHIU-fields of the instruction in an error interrupt location is found, it is assumed an error interrupt just occurred. Special testing and correcting procedures are then followed to ensure that the true error address is reported to the error routine, rather than the address of the execute remote in the trace program.

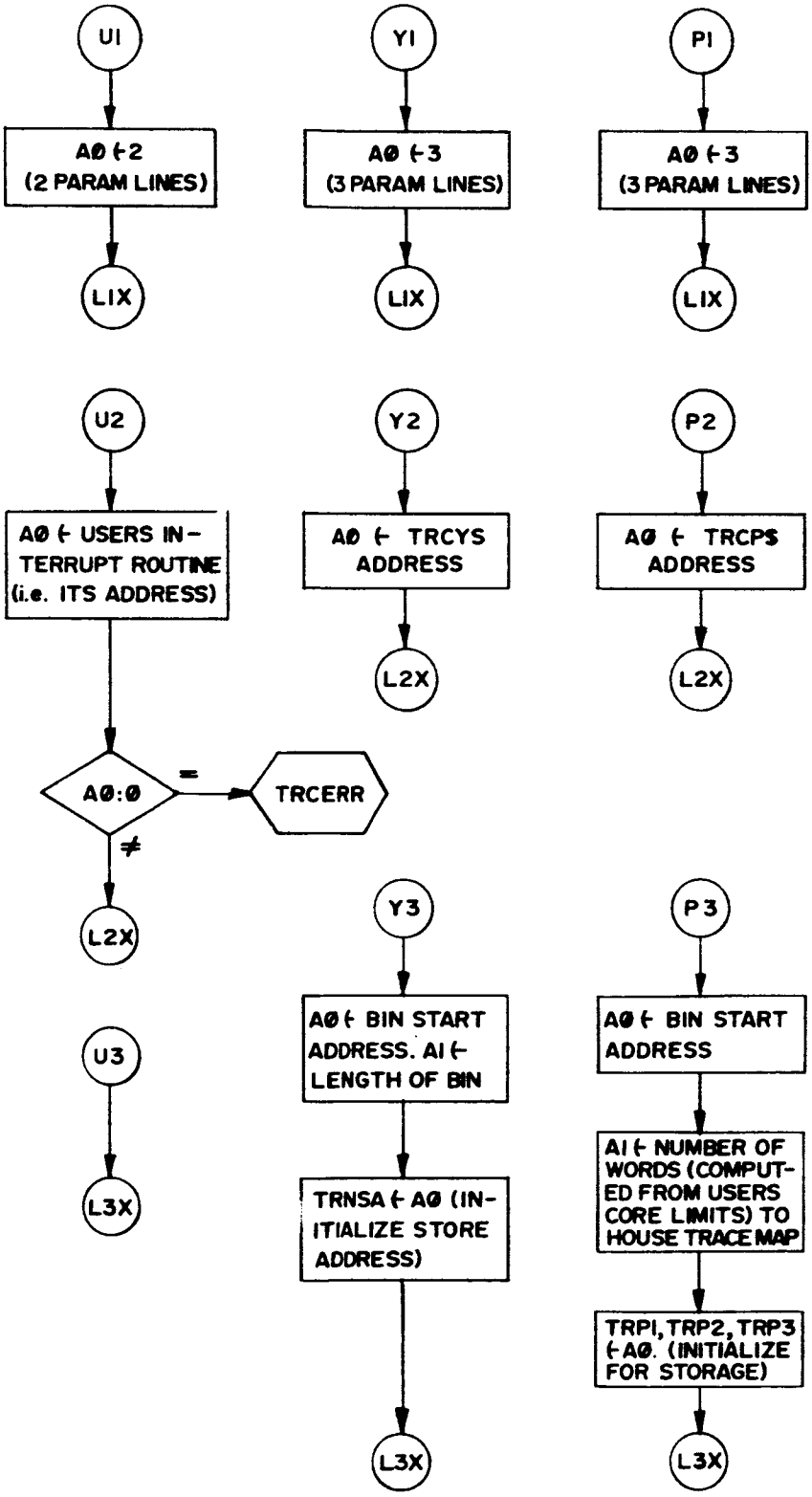
APPENDIX: TRACE ROUTINE FLOW DIAGRAMS


TRC$*
PROCESS INPUT PARAMS FROM SL II CALL
SAVE BIO, BII IN TRSBIO & TRSBII
OBTAIN INTERRUPT CODE NAME AND SEARCH STD. TABLE
IS IT NAME OF STD ROUTINE ?
N
Y
SET SWITCH TO BI-LINK U(n)
SET SWITCH TO BI-LINK Y(n) OR P(n) AS REQUIRED
LI
UI
YI
PI
LIX
PATHS ARE DOWN BILINK U(n), Y(n) OR P(n)
TRCFTX
L2
U2
Y2
P2
L2X
OBTAIN MODE PARAM.
TRIAM

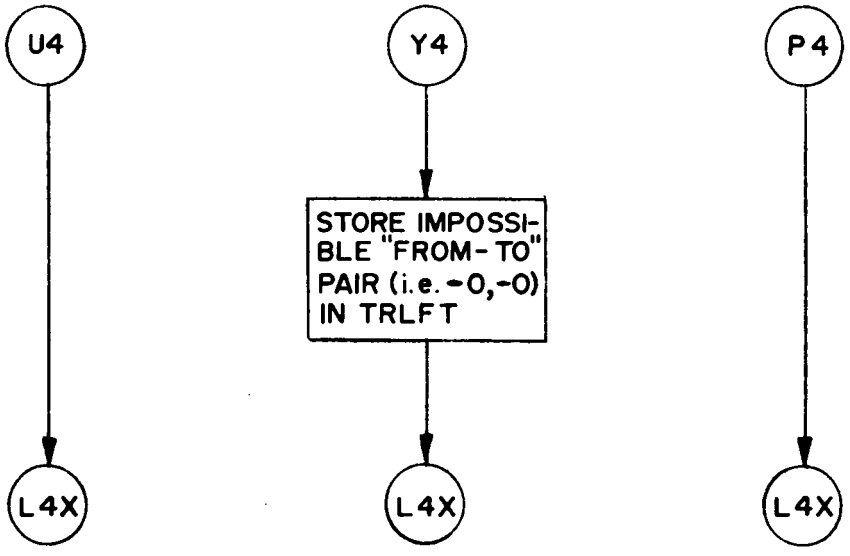
Trace 1


PLACE USERS CORE LIMITS IN AØ, AI
TRLIM
L3
U3
Y3
P3
L3X
TRBIN
L4
U4
Y4
P4
L4X
SET MITRC$ TO LOCATION TRACE$, RESTORE BIO, BII, SIMULATE TRACED JUMP TO TRC$
TRCL4A
TRXP

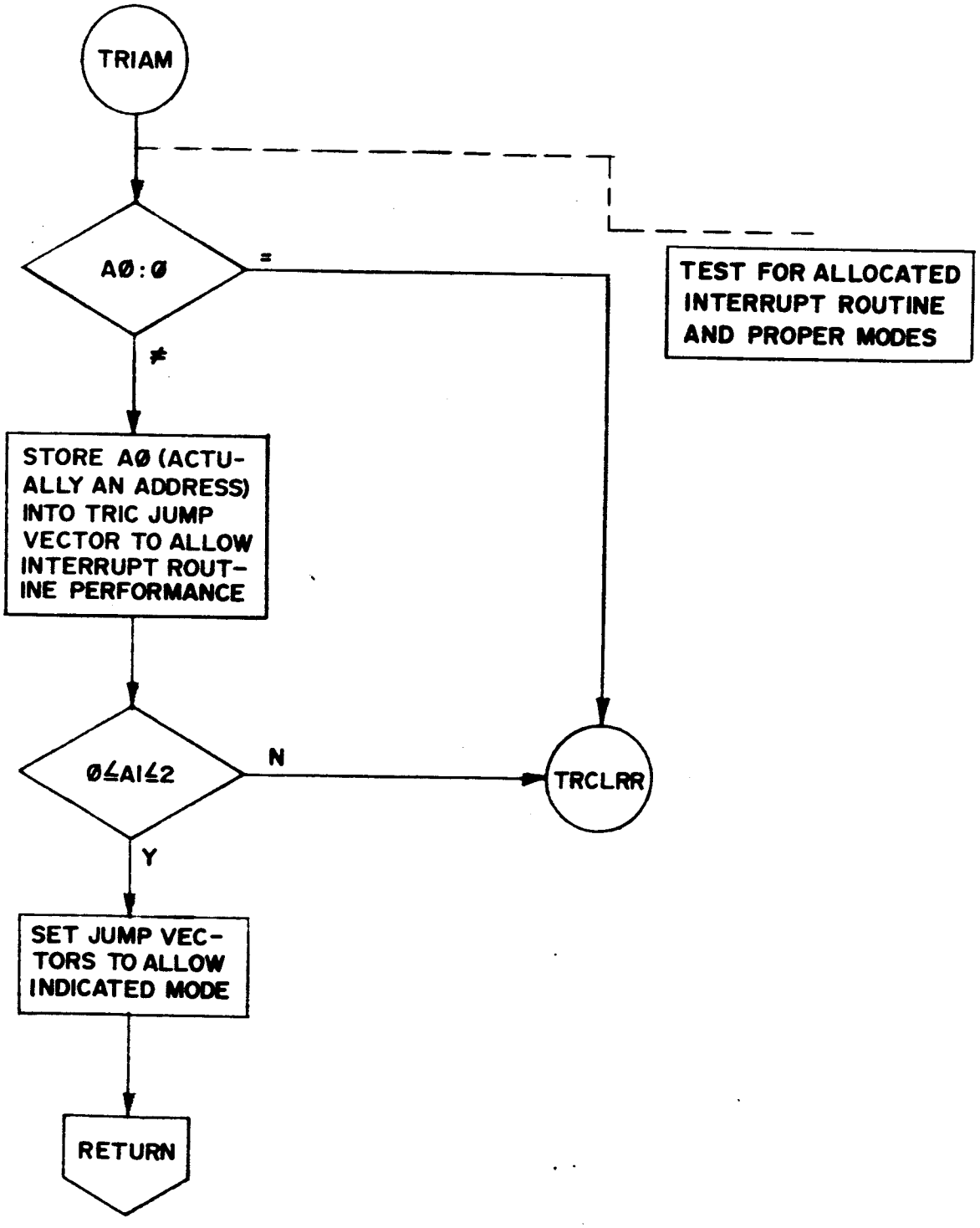
Trace 2


U1
A0 ← 2
(2 PARAM LINES)
LIX
Y1
A0 ← 3
(3 PARAM LINES)
LIX
P1
A0 ← 3
(3 PARAM LINES)
LIX
U2
A0 ← USERS IN-
TERRUPT ROUTINE
(i.e. ITS ADDRESS)
Y2
A0 ← TRCYS
ADDRESS
L2X
P2
A0 ← TRCPS
ADDRESS
L2X
A0:0
=
TRCERR
≠
L2X
U3
L3X
Y3
A0 ← BIN START
ADDRESS. A1 ←
LENGTH OF BIN
TRNSA ← A0 (IN-
ITIALIZE STORE
ADDRESS)
L3X
P3
A0 ← BIN START
ADDRESS
A1 ← NUMBER OF
WORDS (COMPUT-
ED FROM USERS
CORE LIMITS) TO
HOUSE TRACE MAP
TRP1, TRP2, TRP3
← A0. (INITIALIZE
FOR STORAGE)
L3X

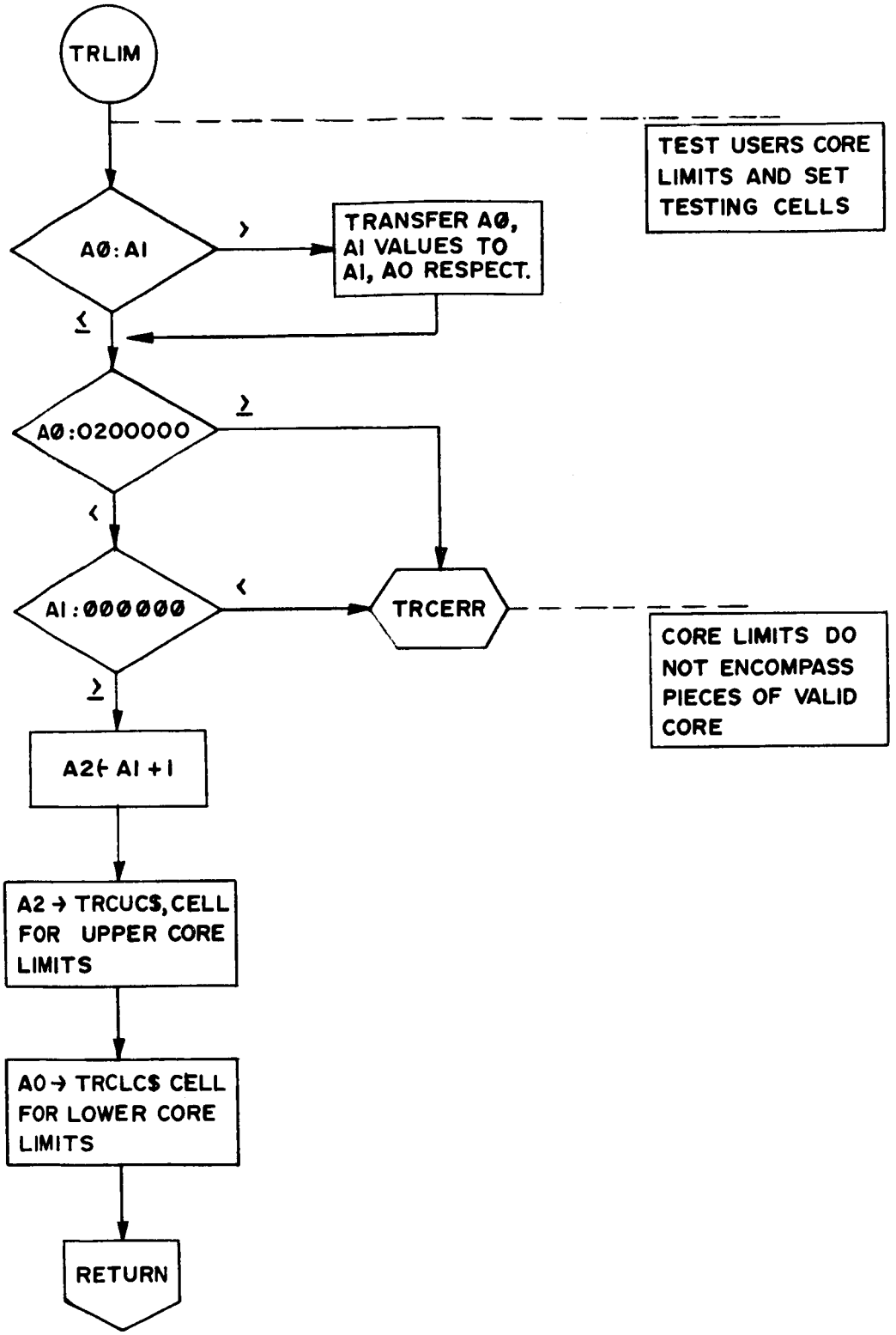
Trace 3


U4
Y4
P4
STORE IMPOSSI-
BLE "FROM-TO"
PAIR (i.e.-0,-0)
IN TRLFT
L4X
L4X
L4X

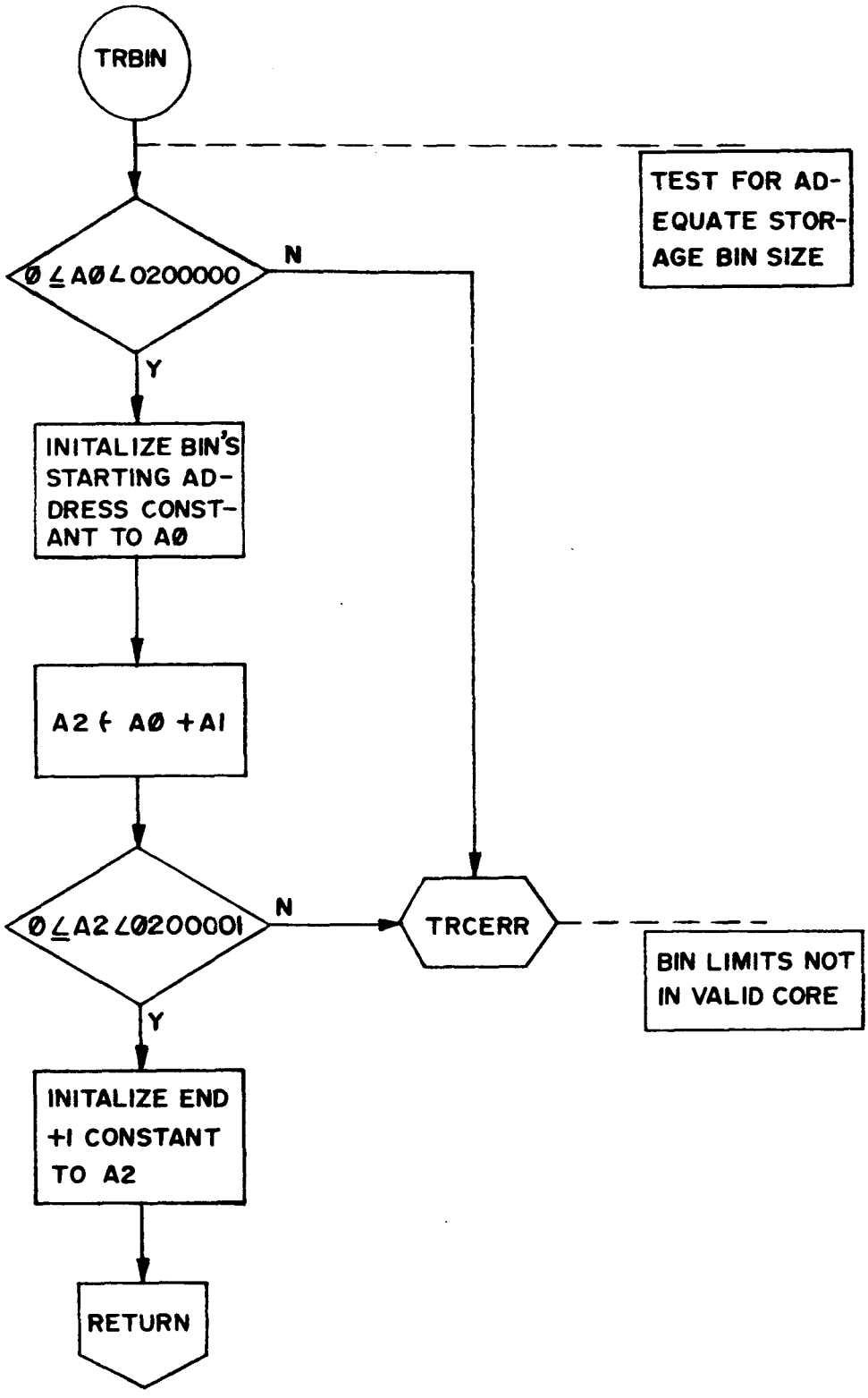
Trace 4


TRIAM
A0:0
=
≠
TEST FOR ALLOCATED INTERRUPT ROUTINE AND PROPER MODES
STORE A0 (ACTUALLY AN ADDRESS) INTO TRIC JUMP VECTOR TO ALLOW INTERRUPT ROUTINE PERFORMANCE
0≤AI≤2
N
TRCLRR
Y
SET JUMP VECTORS TO ALLOW INDICATED MODE
RETURN

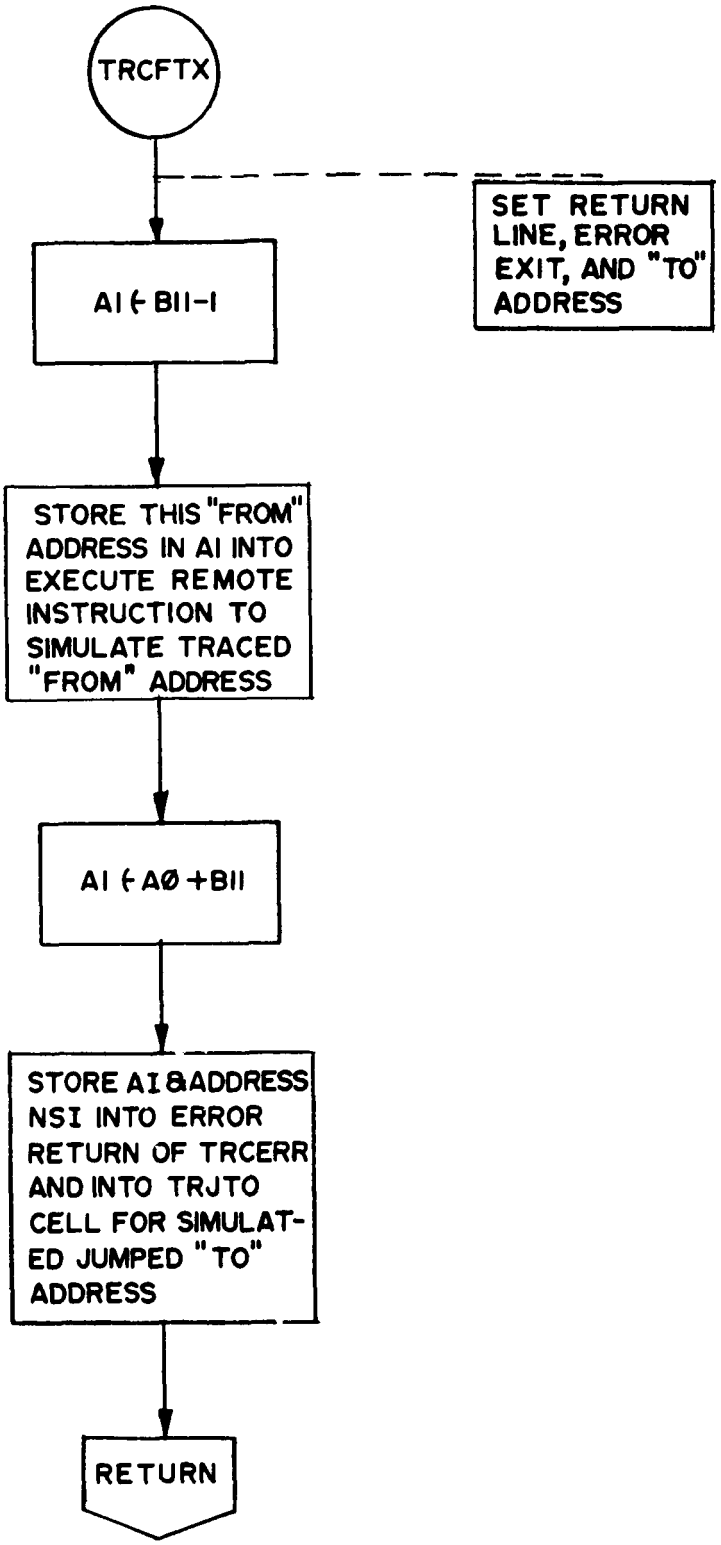
Trace 5


TRLIM
TEST USERS CORE LIMITS AND SET TESTING CELLS
A0:A1
>
TRANSFER A0, A1 VALUES TO A1, A0 RESPECT.
≤
A0:0200000
≥
<
A1:000000
<
TRCERR
CORE LIMITS DO NOT ENCOMPASS PIECES OF VALID CORE
≥
A2← A1 +1
A2 → TRCUC$, CELL FOR UPPER CORE LIMITS
A0 → TRCLC$ CELL FOR LOWER CORE LIMITS
RETURN

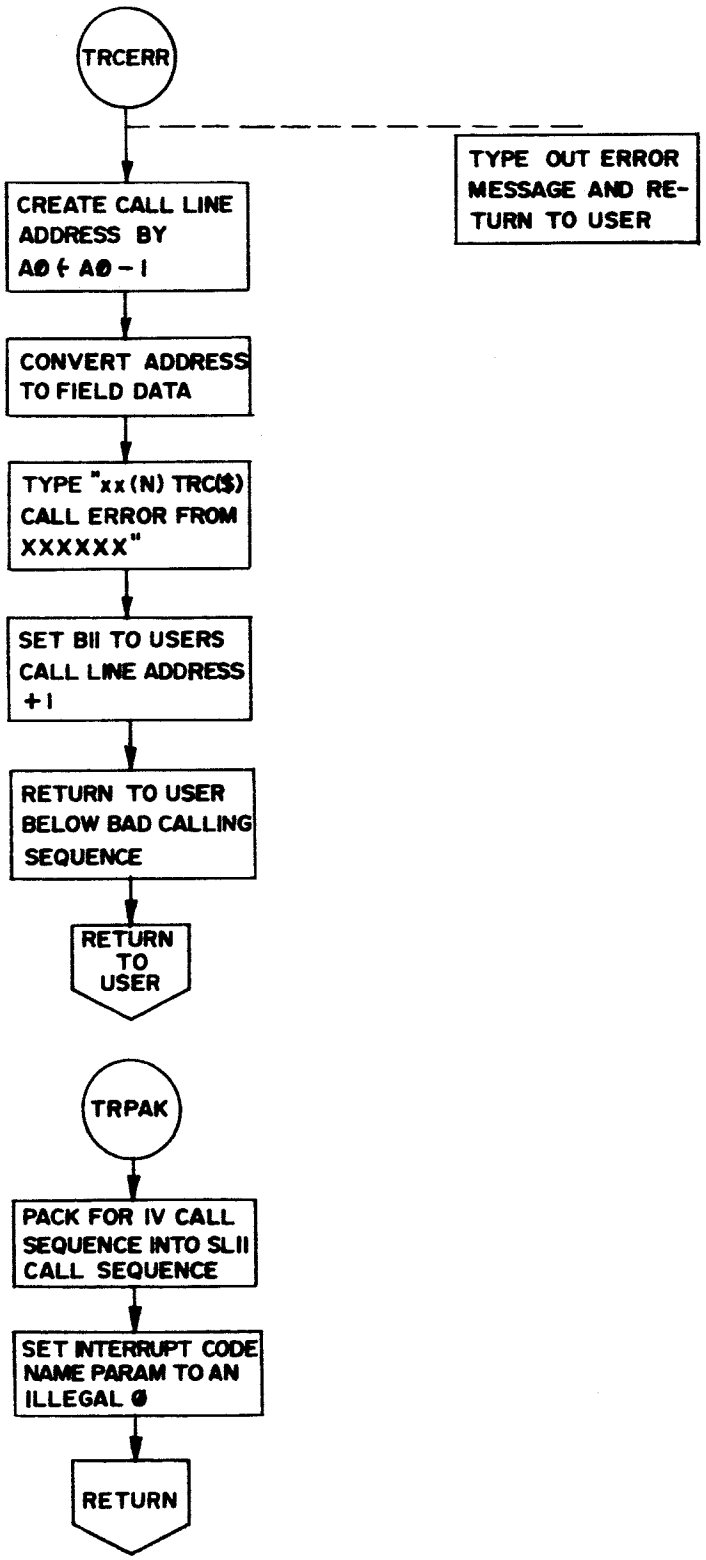
Trace 6


TRBIN
TEST FOR ADEQUATE STORAGE BIN SIZE
Ø ≤ A Ø < 0200000
N
Y
INITALIZE BIN'S STARTING ADDRESS CONSTANT TO AØ
A2 ← AØ + A1
Ø ≤ A2 < 0200001
N
TRCERR
BIN LIMITS NOT IN VALID CORE
Y
INITALIZE END +1 CONSTANT TO A2
RETURN

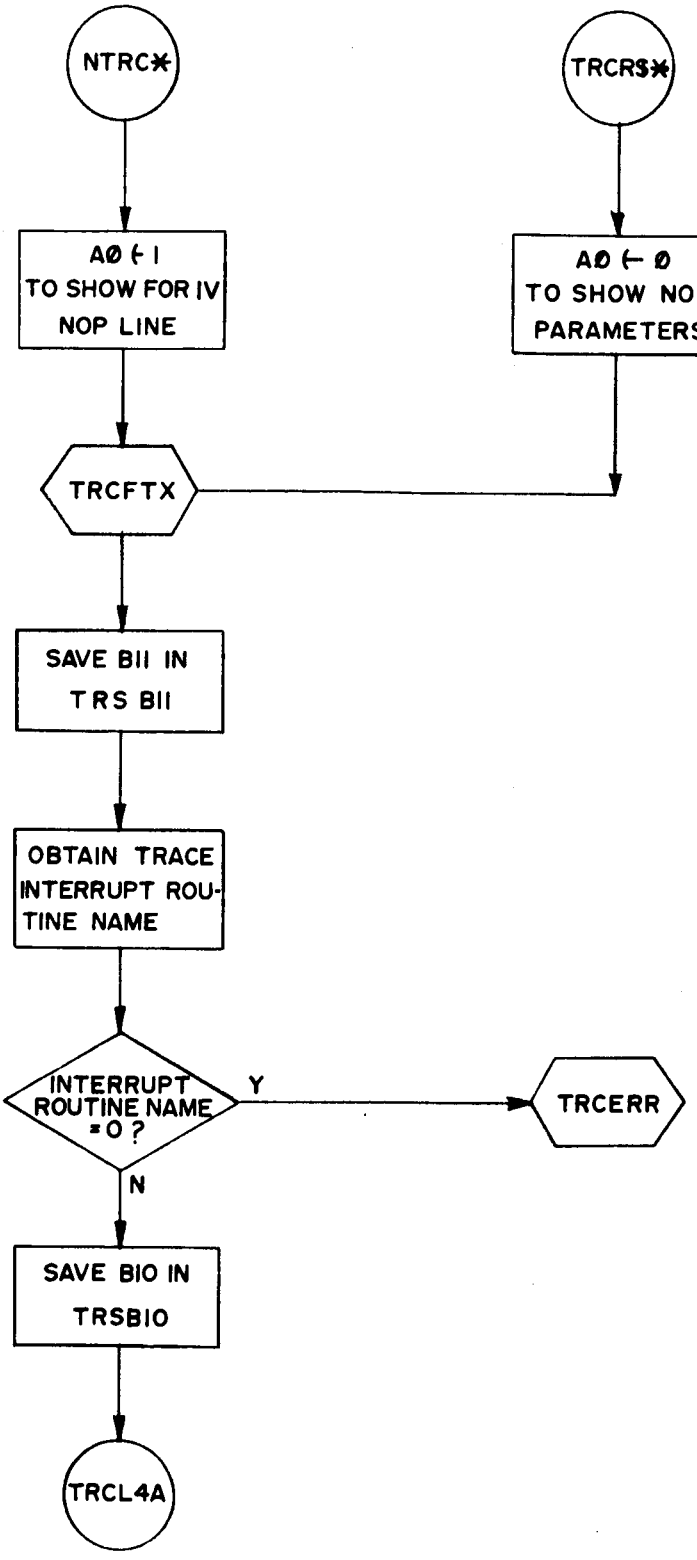
Trace 7


TRCFTX
SET RETURN LINE, ERROR EXIT, AND "TO" ADDRESS
AI ← BII-1
STORE THIS "FROM" ADDRESS IN AI INTO EXECUTE REMOTE INSTRUCTION TO SIMULATE TRACED "FROM" ADDRESS
AI ← AØ + BII
STORE AI & ADDRESS NSI INTO ERROR RETURN OF TRCERR AND INTO TRJTO CELL FOR SIMULATED JUMPED "TO" ADDRESS
RETURN

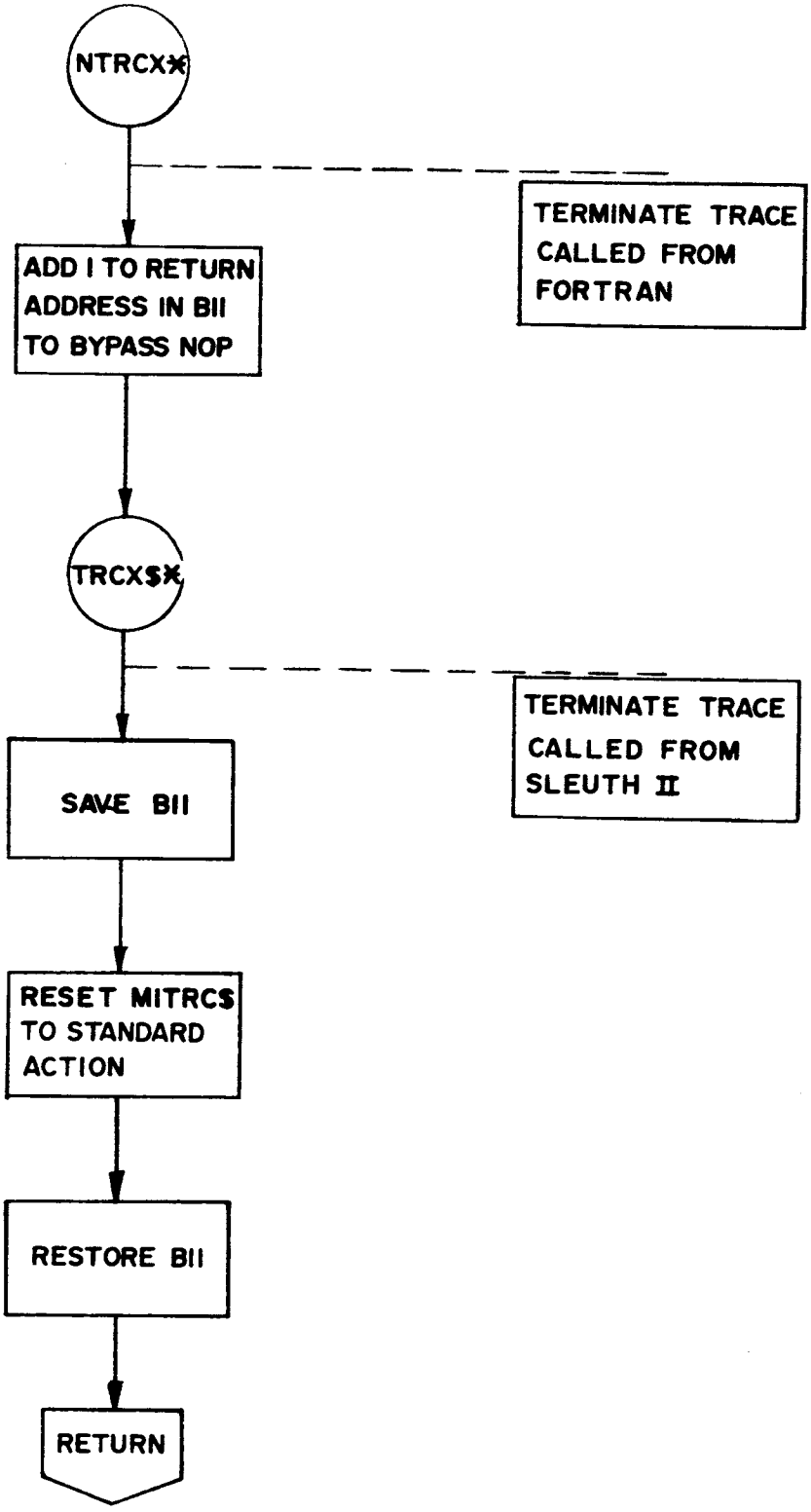
Trace 8


TRCERR
TYPE OUT ERROR MESSAGE AND RETURN TO USER
CREATE CALL LINE ADDRESS BY A0 ← A0 - 1
CONVERT ADDRESS TO FIELD DATA
TYPE "xx(N) TRC($) CALL ERROR FROM XXXXXX"
SET B11 TO USERS CALL LINE ADDRESS +1
RETURN TO USER BELOW BAD CALLING SEQUENCE
RETURN TO USER
TRPAK
PACK FOR IV CALL SEQUENCE INTO SL11 CALL SEQUENCE
SET INTERRUPT CODE NAME PARAM TO AN ILLEGAL 0
RETURN

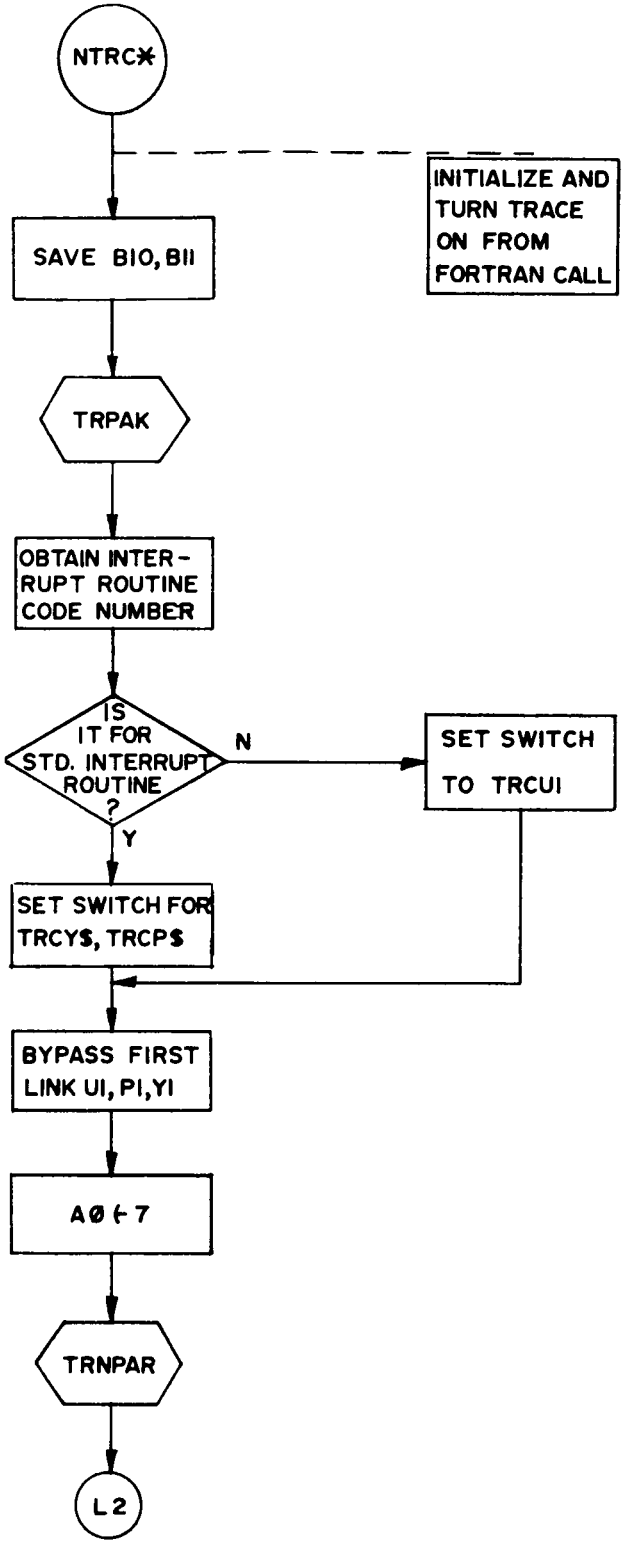
Trace 9


NTRC*
TRCRS*
A0 ← 1
TO SHOW FOR IV
NOP LINE
A0 ← 0
TO SHOW NO
PARAMETERS
TRCFTX
SAVE B11 IN
TRS B11
OBTAIN TRACE
INTERRUPT ROU-
TINE NAME
INTERRUPT
ROUTINE NAME
=0 ?
Y
TRCERR
N
SAVE B10 IN
TRSB10
TRCL4A

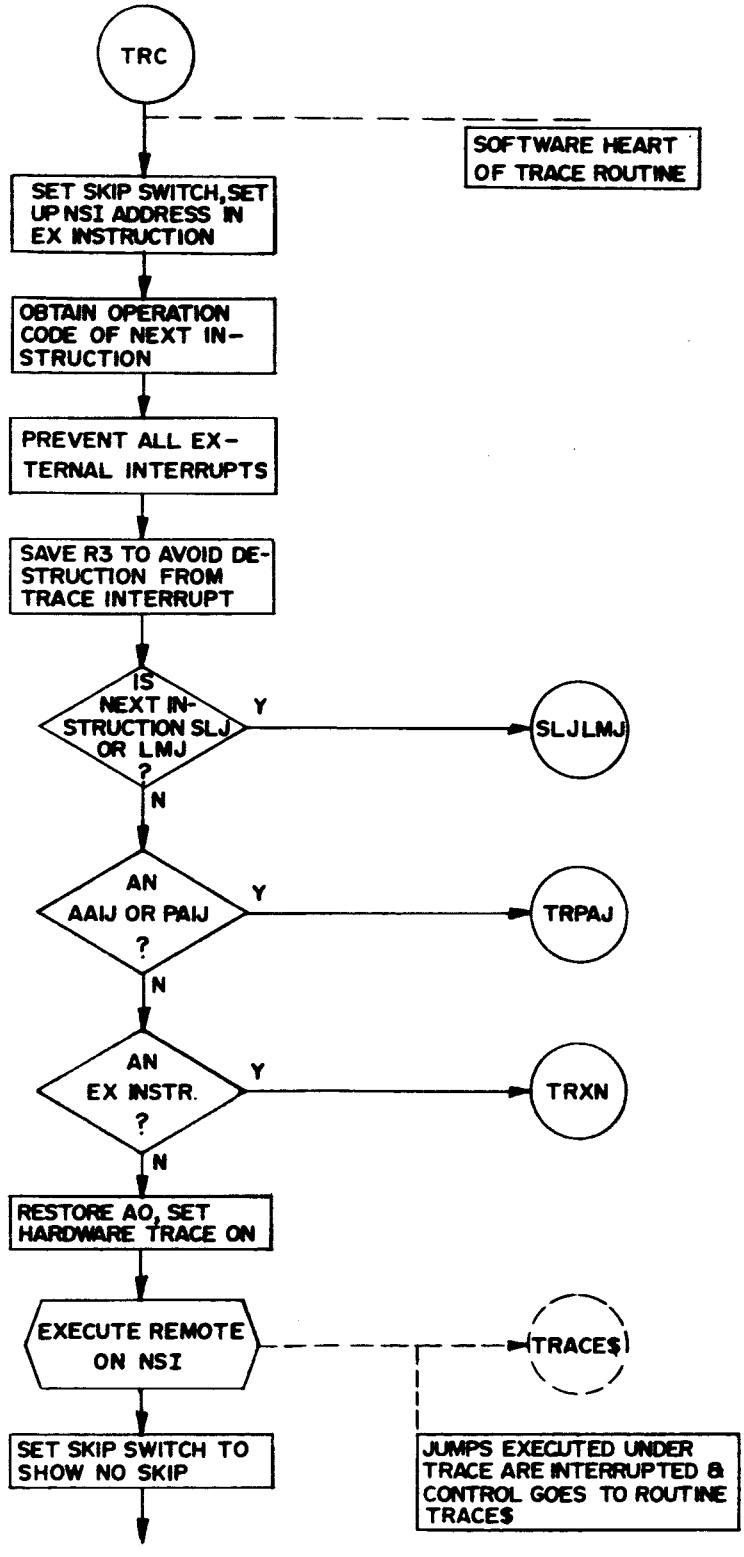
Trace 10


NTRCX*
TERMINATE TRACE CALLED FROM FORTRAN
ADD 1 TO RETURN ADDRESS IN BII TO BYPASS NOP
TRCX$*
TERMINATE TRACE CALLED FROM SLEUTH II
SAVE BII
RESET MITRC$ TO STANDARD ACTION
RESTORE BII
RETURN

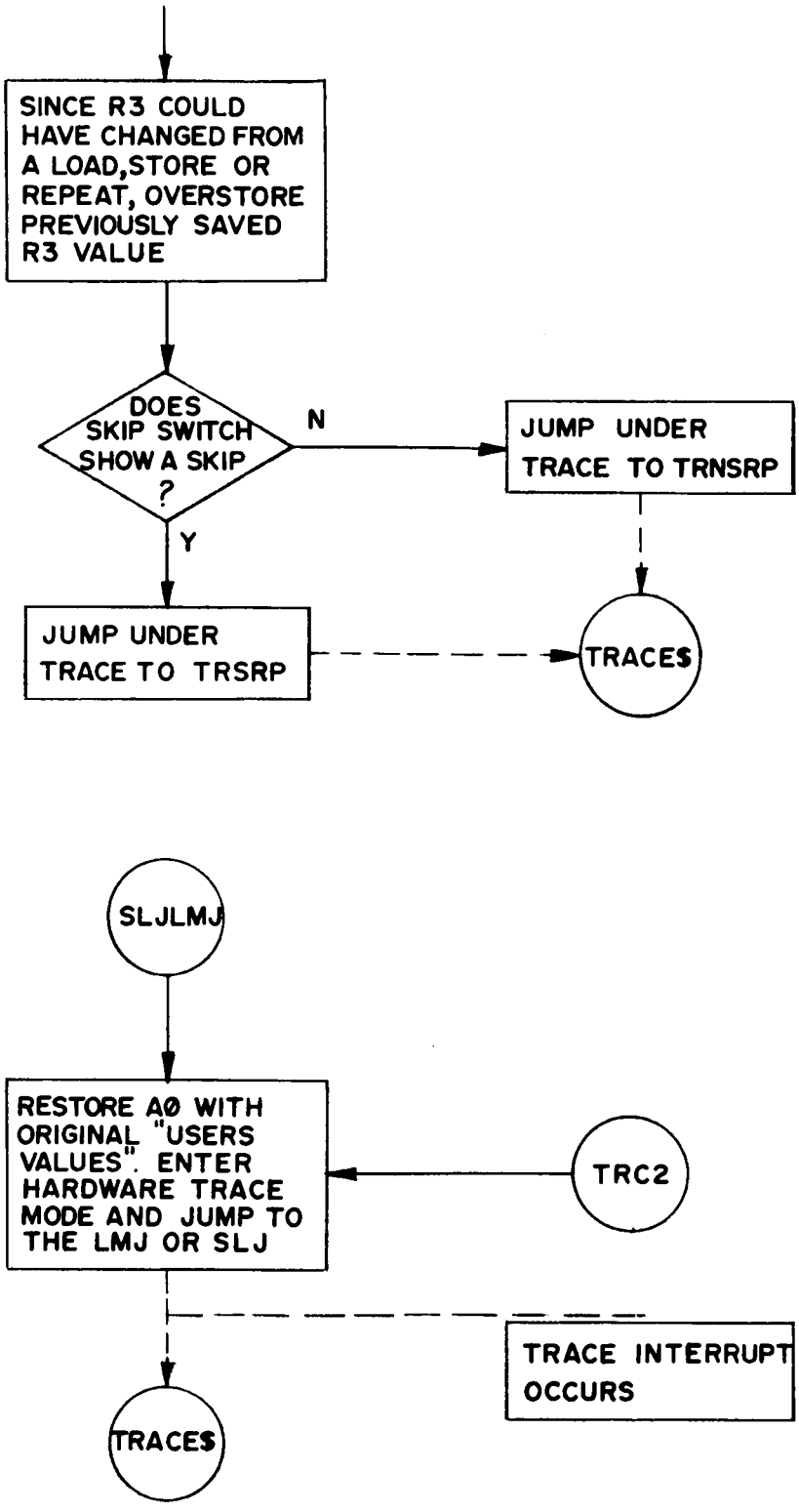
Trace 11


NTRC*
INITIALIZE AND TURN TRACE ON FROM FORTRAN CALL
SAVE BIO, BII
TRPAK
OBTAIN INTER-RUPT ROUTINE CODE NUMBER
IS IT FOR STD. INTERRUPT ROUTINE ?
N
SET SWITCH TO TRCUI
Y
SET SWITCH FOR TRCY$, TRCP$
BYPASS FIRST LINK UI, PI, YI
A0 ← 7
TRNPAR
L2

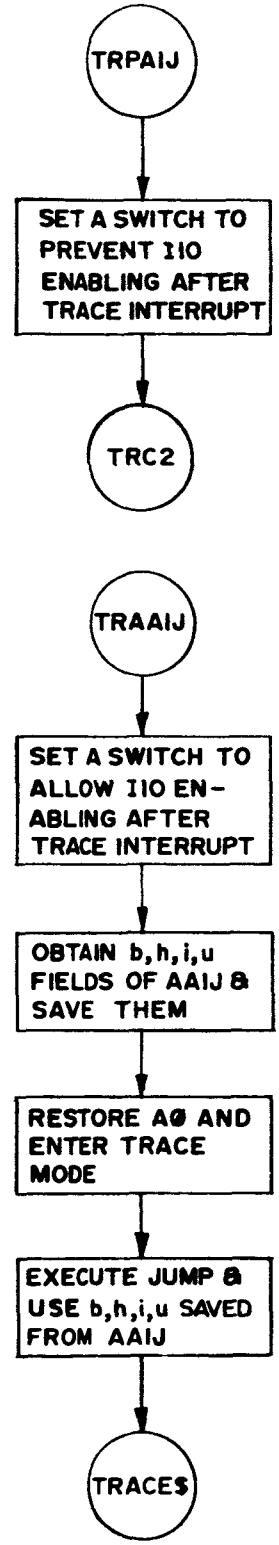
Trace 12


TRC
SOFTWARE HEART OF TRACE ROUTINE
SET SKIP SWITCH, SET UP NSI ADDRESS IN EX INSTRUCTION
OBTAIN OPERATION CODE OF NEXT IN-STRUCTION
PREVENT ALL EX-TERNAL INTERRUPTS
SAVE R3 TO AVOID DE-STRUCTION FROM TRACE INTERRUPT
IS NEXT IN-STRUCTION SLJ OR LMJ ?
Y
SLJLMJ
N
AN AAIJ OR PAIJ ?
Y
TRPAJ
N
AN EX INSTR. ?
Y
TRXN
N
RESTORE AO, SET HARDWARE TRACE ON
EXECUTE REMOTE ON NSI
TRACE$
SET SKIP SWITCH TO SHOW NO SKIP
JUMPS EXECUTED UNDER TRACE ARE INTERRUPTED & CONTROL GOES TO ROUTINE TRACE$

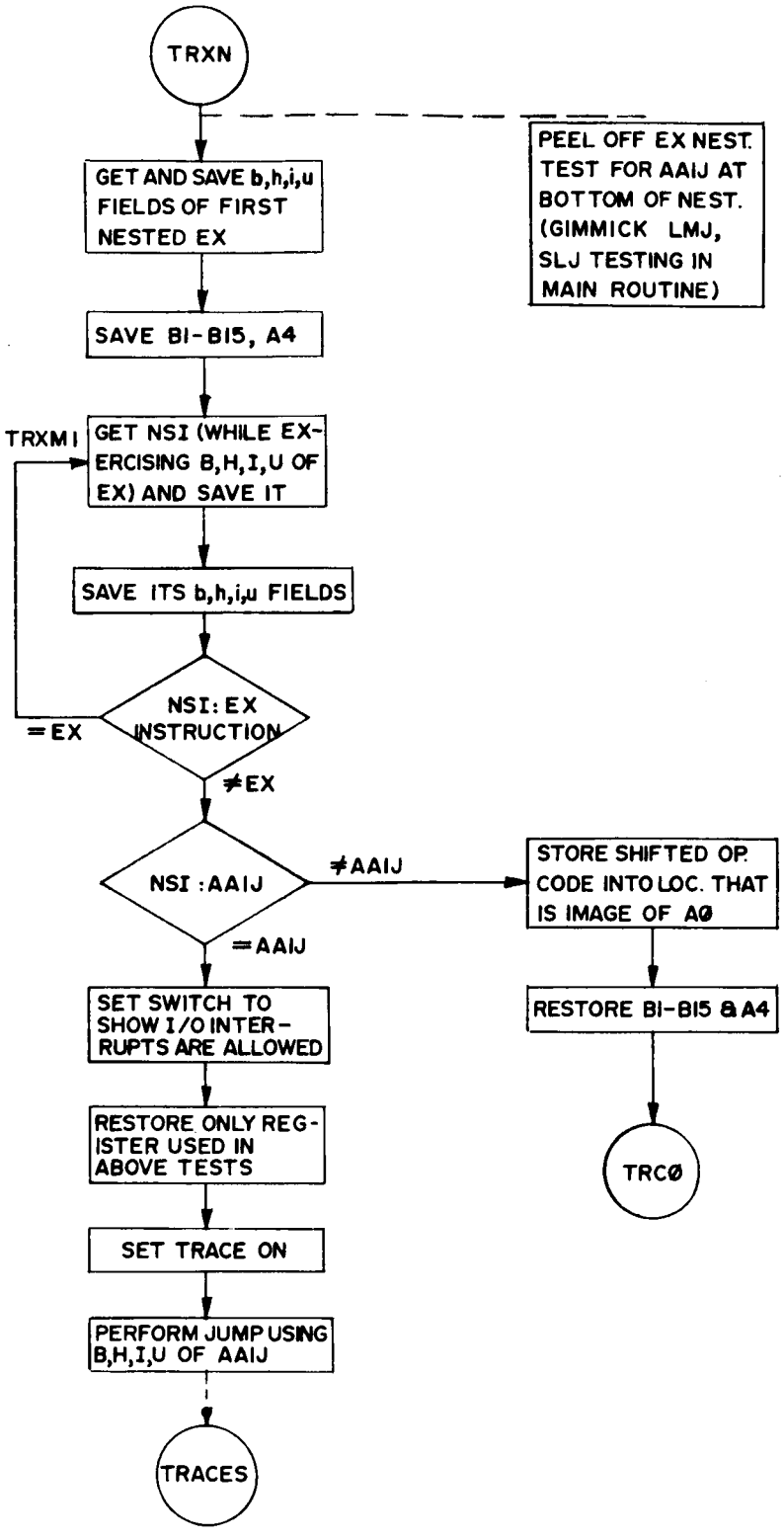
Trace 13


SINCE R3 COULD HAVE CHANGED FROM A LOAD,STORE OR REPEAT, OVERSTORE PREVIOUSLY SAVED R3 VALUE
DOES SKIP SWITCH SHOW A SKIP ?
N
Y
JUMP UNDER TRACE TO TRNSRP
JUMP UNDER TRACE TO TRSRP
TRACE$
SLJLMJ
RESTORE A0 WITH ORIGINAL "USERS VALUES". ENTER HARDWARE TRACE MODE AND JUMP TO THE LMJ OR SLJ
TRC2
TRACE INTERRUPT OCCURS
TRACE$

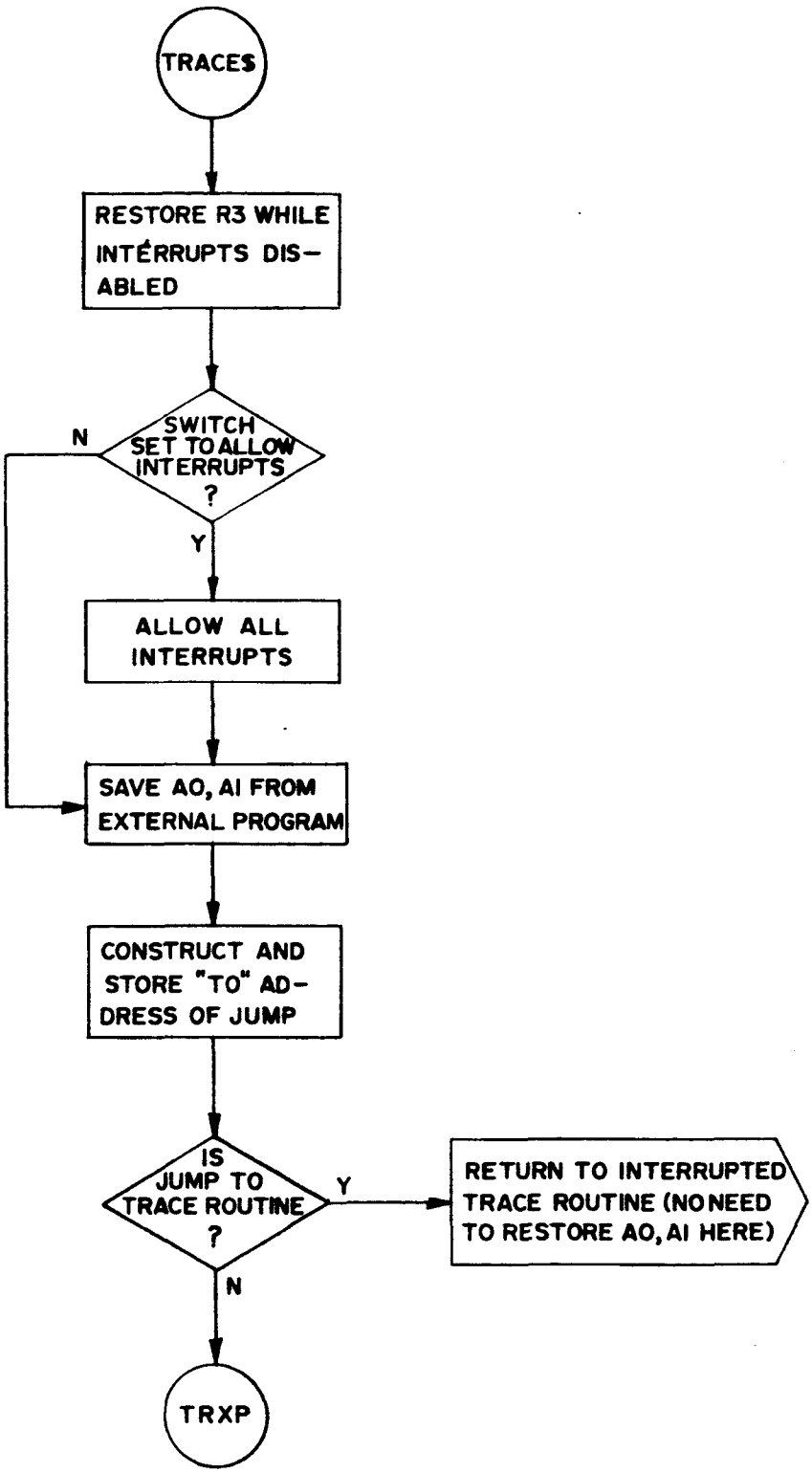
Trace 14


TRPAIJ
SET A SWITCH TO PREVENT I1O ENABLING AFTER TRACE INTERRUPT
TRC2
TRAAIJ
SET A SWITCH TO ALLOW I1O EN-ABLING AFTER TRACE INTERRUPT
OBTAIN b,h,i,u FIELDS OF AAIJ & SAVE THEM
RESTORE A0 AND ENTER TRACE MODE
EXECUTE JUMP & USE b,h,i,u SAVED FROM AAIJ
TRACE$

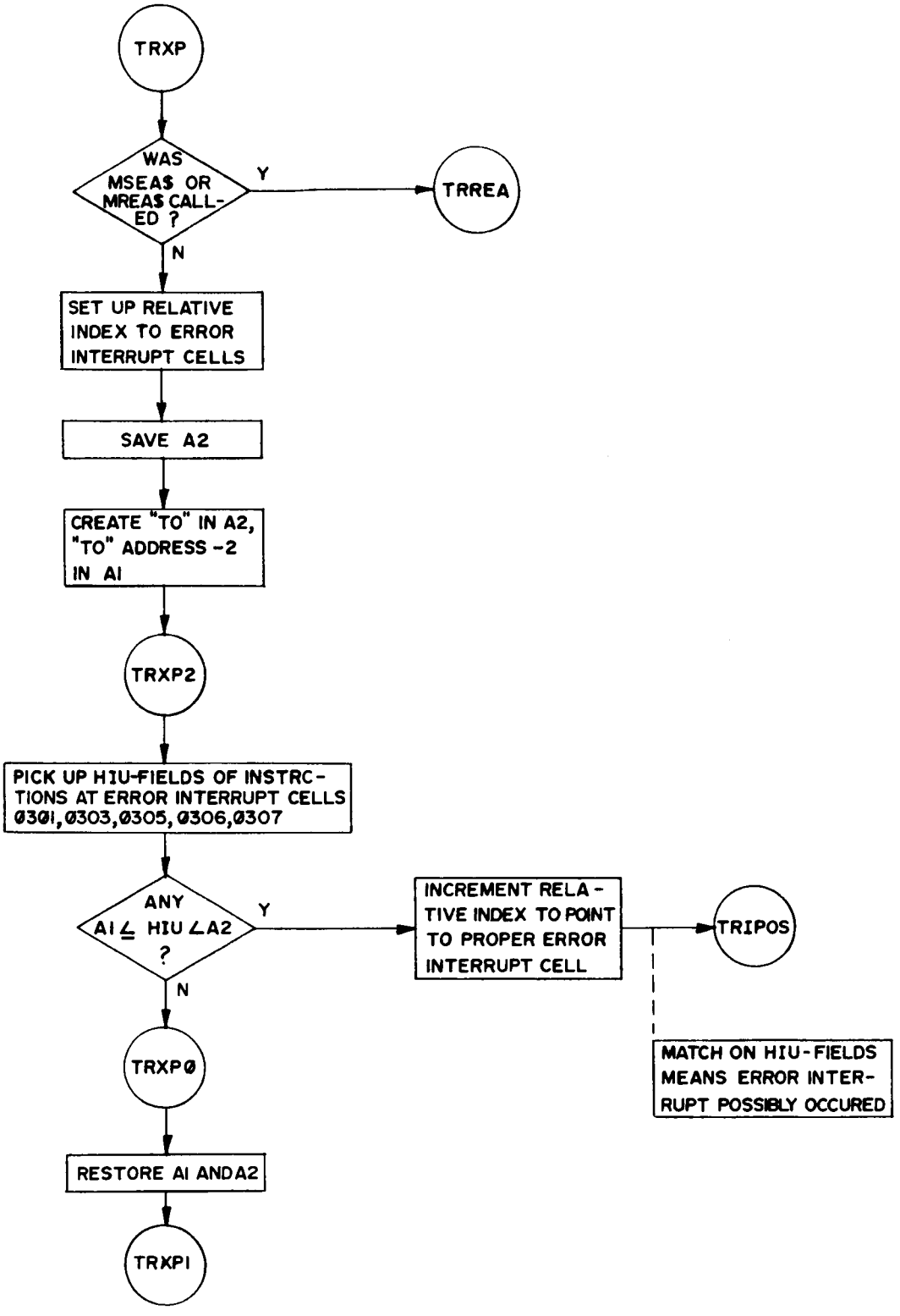
Trace 15


TRXN
PEEL OFF EX NEST. TEST FOR AAIJ AT BOTTOM OF NEST. (GIMMICK LMJ, SLJ TESTING IN MAIN ROUTINE)
GET AND SAVE b,h,i,u FIELDS OF FIRST NESTED EX
SAVE B1-B15, A4
TRXM1
GET NSI (WHILE EXERCISING B,H,I,U OF EX) AND SAVE IT
SAVE ITS b,h,i,u FIELDS
NSI: EX INSTRUCTION
=EX
≠EX
NSI : AAIJ
≠AAIJ
STORE SHIFTED OP. CODE INTO LOC. THAT IS IMAGE OF A0
=AAIJ
SET SWITCH TO SHOW I/O INTERRUPTS ARE ALLOWED
RESTORE B1-B15 & A4
RESTORE ONLY REGISTER USED IN ABOVE TESTS
TRC0
SET TRACE ON
PERFORM JUMP USING B,H,I,U OF AAIJ
TRACES

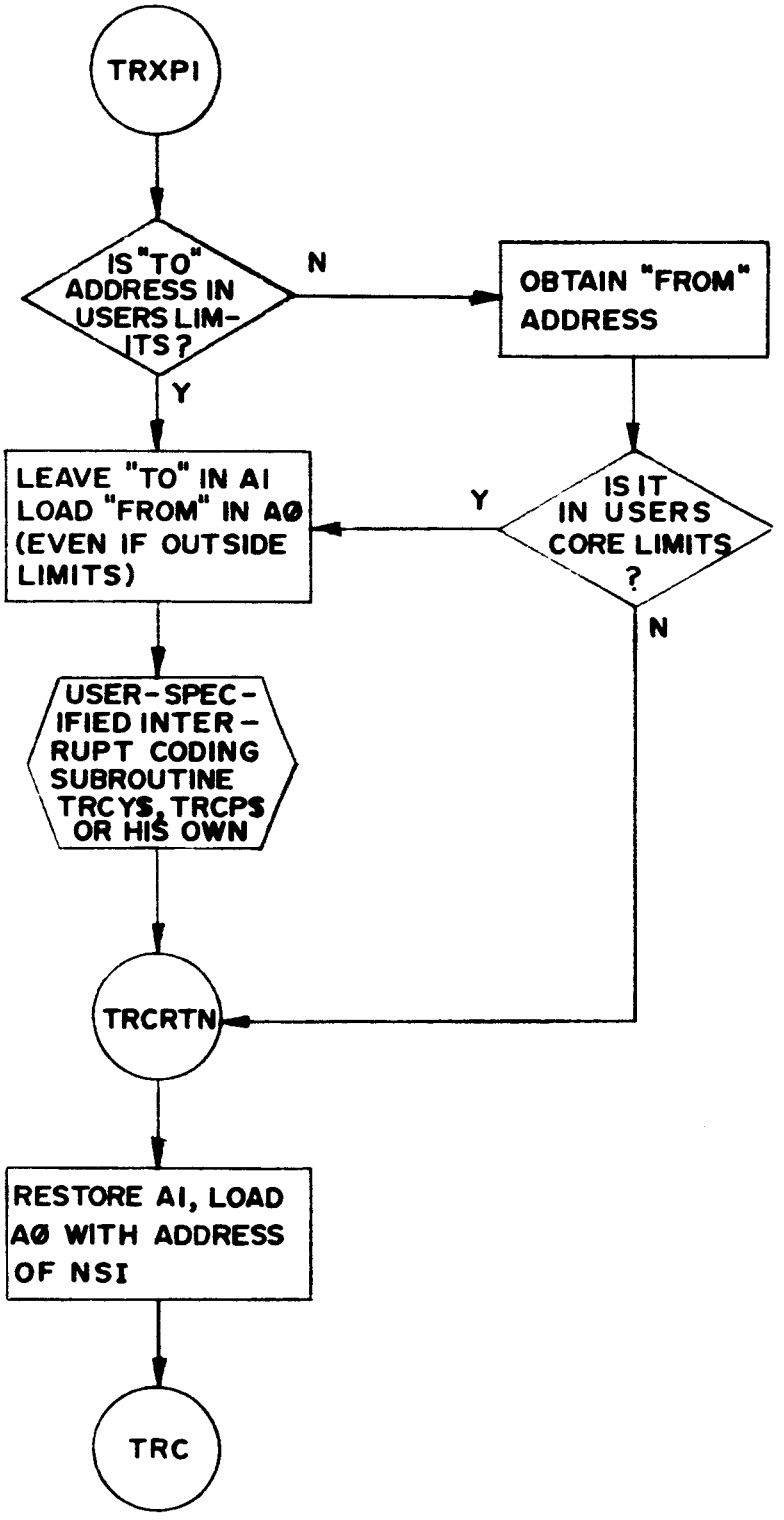
Trace 16


TRACE$
RESTORE R3 WHILE INTERRUPTS DIS-ABLED
SWITCH SET TO ALLOW INTERRUPTS ?
N
Y
ALLOW ALL INTERRUPTS
SAVE AO, AI FROM EXTERNAL PROGRAM
CONSTRUCT AND STORE "TO" AD-DRESS OF JUMP
IS JUMP TO TRACE ROUTINE ?
Y
RETURN TO INTERRUPTED TRACE ROUTINE (NO NEED TO RESTORE AO, AI HERE)
N
TRXP

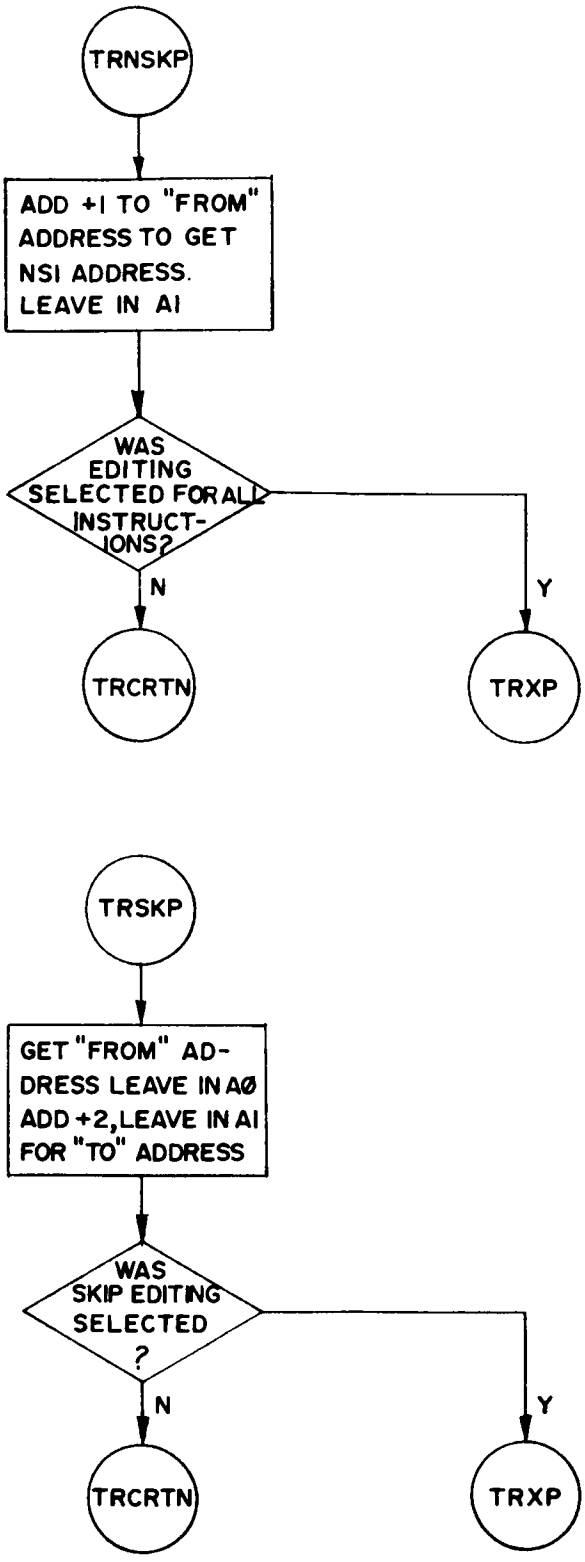
Trace 17


TRXP
WAS MSEA$ OR MREA$ CALL-ED ?
Y
TRREA
N
SET UP RELATIVE INDEX TO ERROR INTERRUPT CELLS
SAVE A2
CREATE "TO" IN A2, "TO" ADDRESS -2 IN AI
TRXP2
PICK UP HIU-FIELDS OF INSTRC-TIONS AT ERROR INTERRUPT CELLS 0301,0303,0305,0306,0307
ANY AI ≤ HIU < A2 ?
Y
INCREMENT RELA-TIVE INDEX TO POINT TO PROPER ERROR INTERRUPT CELL
TRIPOS
MATCH ON HIU-FIELDS MEANS ERROR INTER-RUPT POSSIBLY OCCURED
N
TRXP0
RESTORE AI AND A2
TRXPI

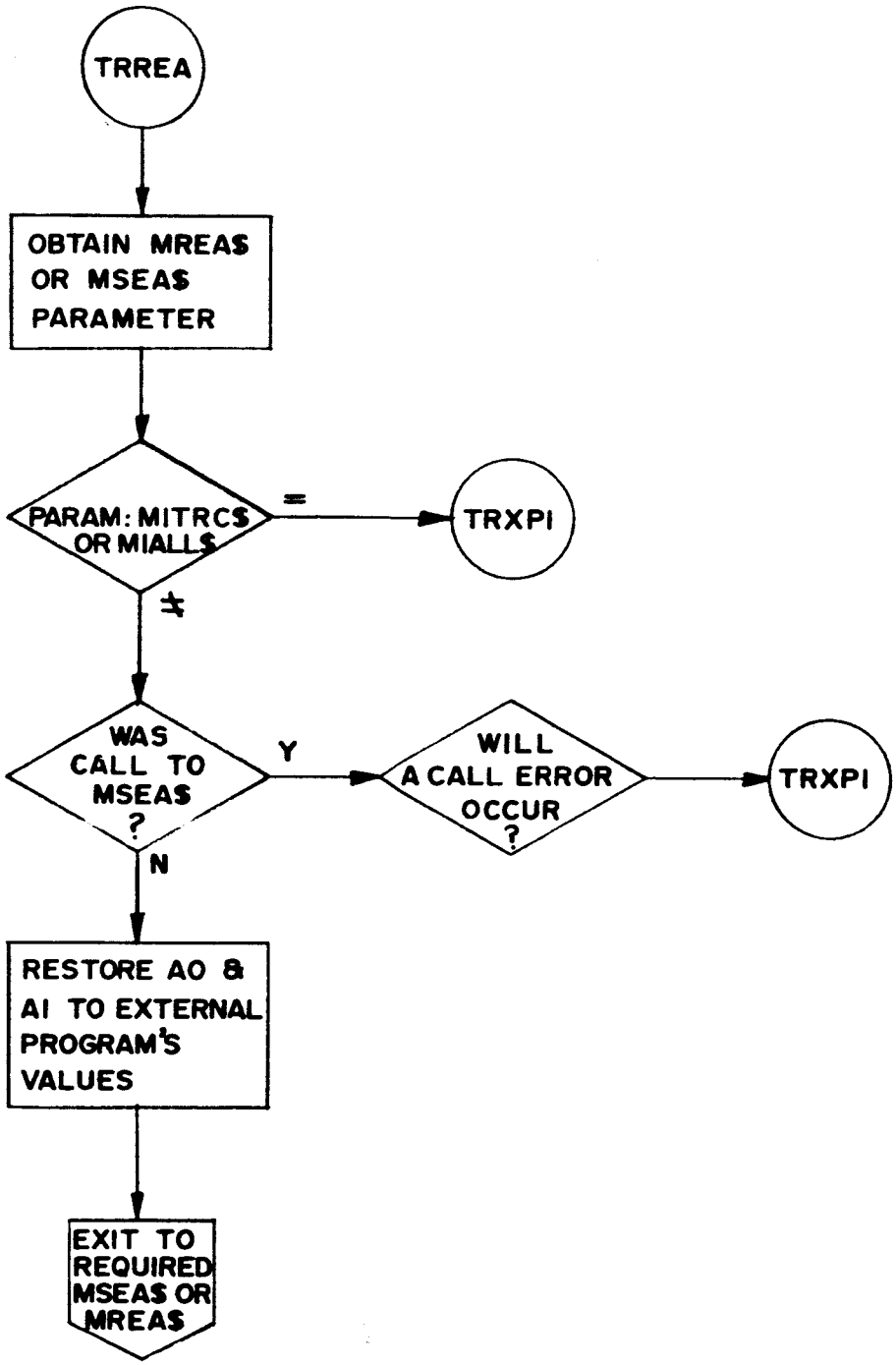
Trace 18


TRXPI
IS "TO" ADDRESS IN USERS LIM-ITS?
N
OBTAIN "FROM" ADDRESS
Y
LEAVE "TO" IN A1 LOAD "FROM" IN A0 (EVEN IF OUTSIDE LIMITS)
Y
IS IT IN USERS CORE LIMITS ?
N
USER-SPEC-IFIED INTER-RUPT CODING SUBROUTINE TRCY$, TRCP$ OR HIS OWN
TRCRTN
RESTORE A1, LOAD A0 WITH ADDRESS OF NSI
TRC

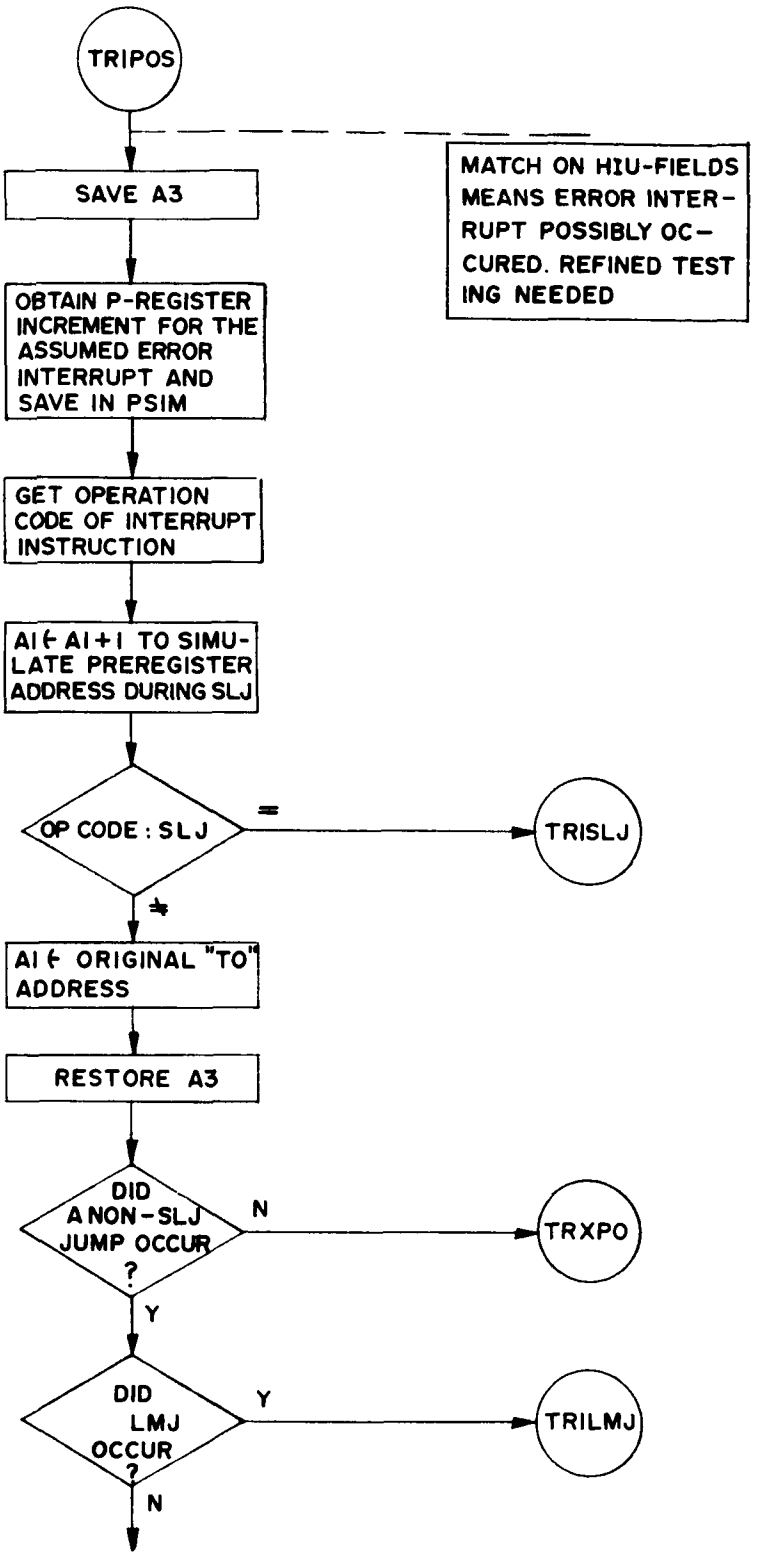
Trace 19


TRNSKP
ADD +1 TO "FROM" ADDRESS TO GET NSI ADDRESS. LEAVE IN A1
WAS EDITING SELECTED FOR ALL INSTRUCT-IONS?
N
Y
TRCRTN
TRXP
TRSKP
GET "FROM" AD-DRESS LEAVE IN A0 ADD +2, LEAVE IN A1 FOR "TO" ADDRESS
WAS SKIP EDITING SELECTED ?
N
Y
TRCRTN
TRXP

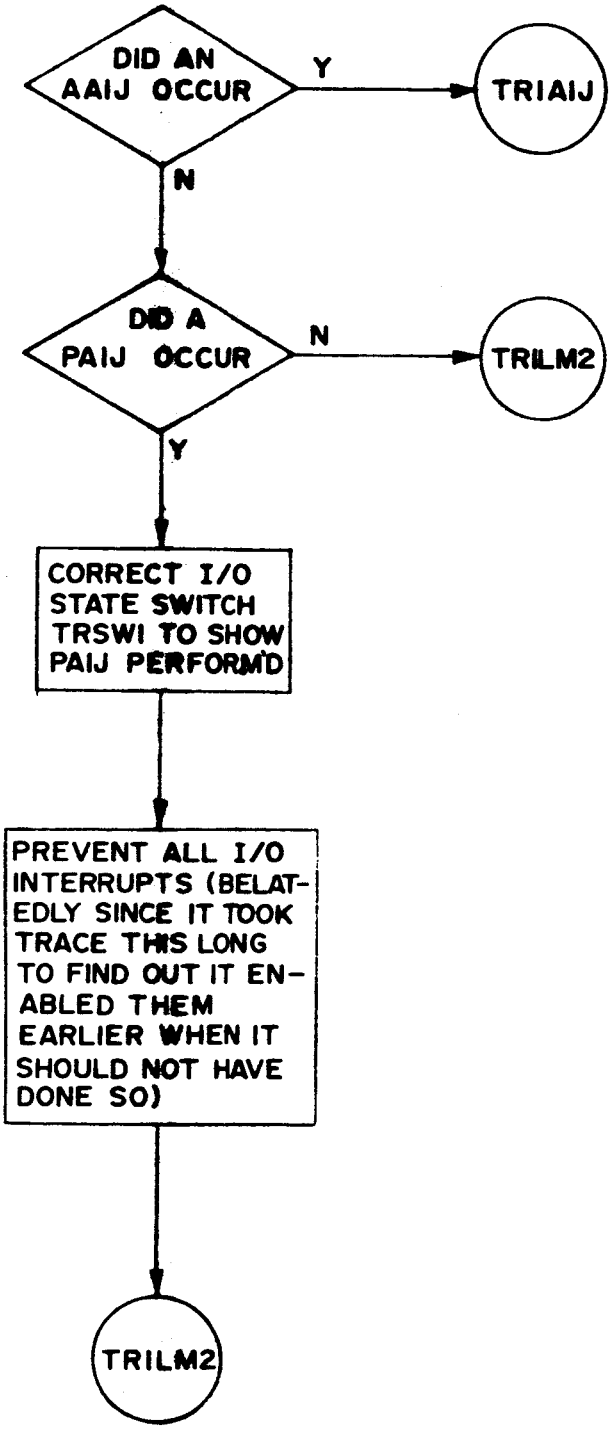
Trace 20


TRREA
OBTAIN MREA$ OR MSEA$ PARAMETER
PARAM: MITRC$ OR MIALL$
=
TRXPI
≠
WAS CALL TO MSEA$?
Y
WILL A CALL ERROR OCCUR ?
TRXPI
N
RESTORE AO & AI TO EXTERNAL PROGRAM'S VALUES
EXIT TO REQUIRED MSEA$ OR MREA$

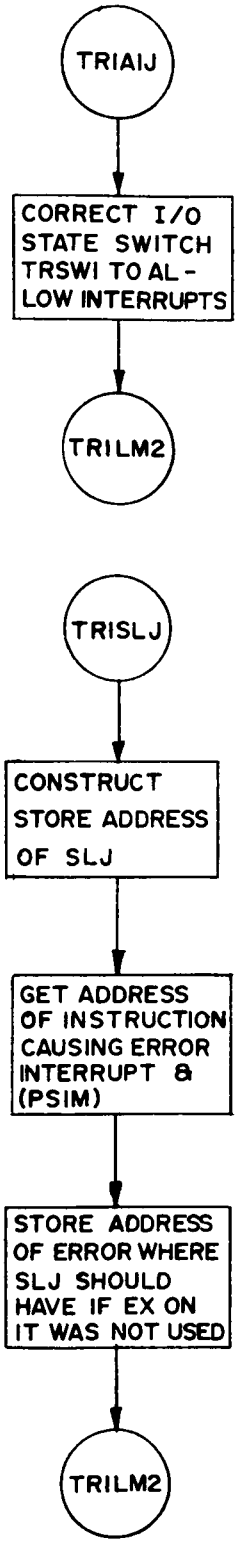
Trace 21


TRIPOS
MATCH ON HIU-FIELDS MEANS ERROR INTERRUPT POSSIBLY OCCURED. REFINED TESTING NEEDED
SAVE A3
OBTAIN P-REGISTER INCREMENT FOR THE ASSUMED ERROR INTERRUPT AND SAVE IN PSIM
GET OPERATION CODE OF INTERRUPT INSTRUCTION
AI ← AI+1 TO SIMULATE PREREGISTER ADDRESS DURING SLJ
OP CODE : SLJ
=
TRISLJ
≠
AI ← ORIGINAL "TO" ADDRESS
RESTORE A3
DID A NON-SLJ JUMP OCCUR ?
N
TRXPO
Y
DID LMJ OCCUR ?
Y
TRILMJ
N

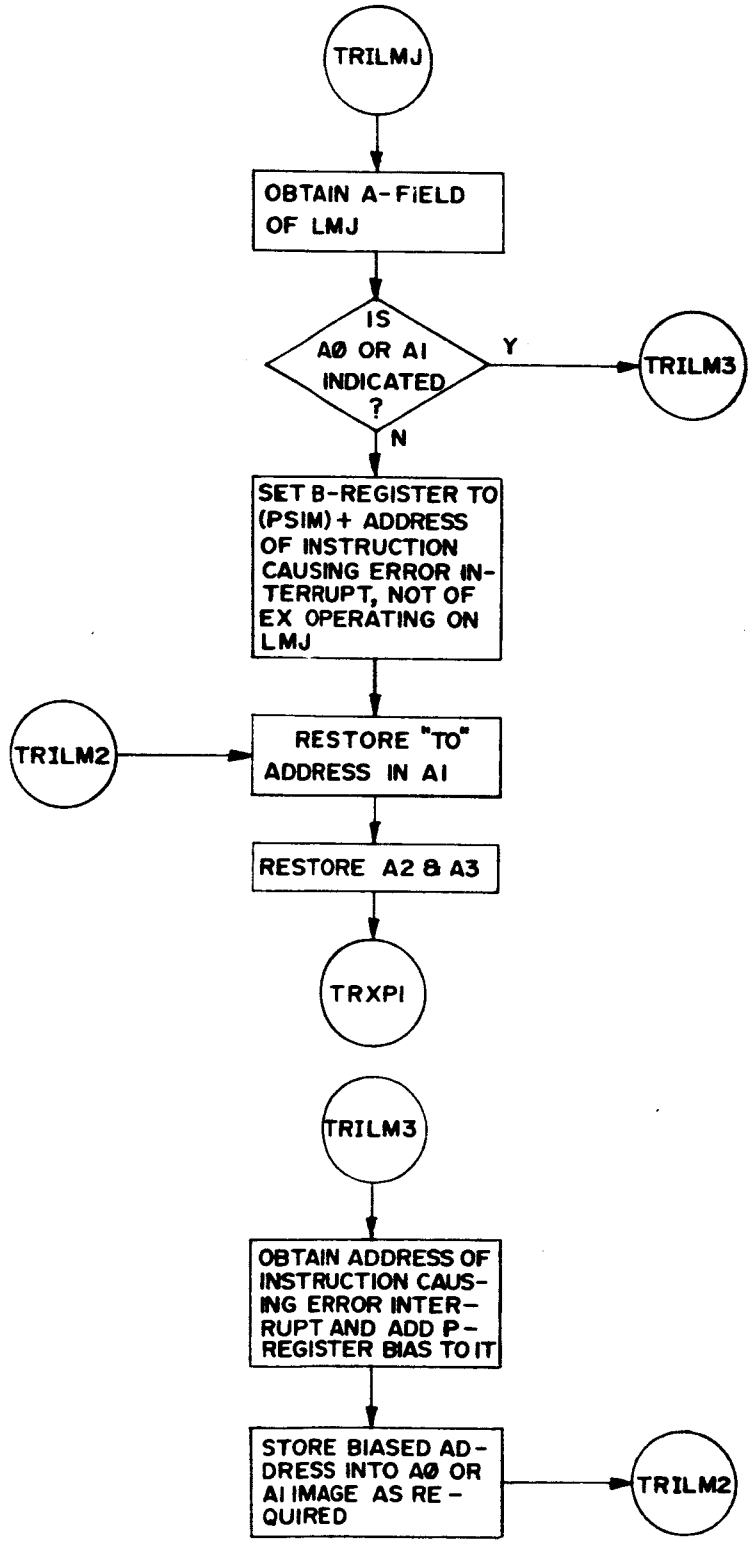
Trace 22


DID AN AAIJ OCCUR
Y
TRIAIJ
N
DID A PAIJ OCCUR
N
TRILM2
Y
CORRECT I/O STATE SWITCH TRSWI TO SHOW PAIJ PERFORMD
PREVENT ALL I/O INTERRUPTS (BELATEDLY SINCE IT TOOK TRACE THIS LONG TO FIND OUT IT ENABLED THEM EARLIER WHEN IT SHOULD NOT HAVE DONE SO)
TRILM2

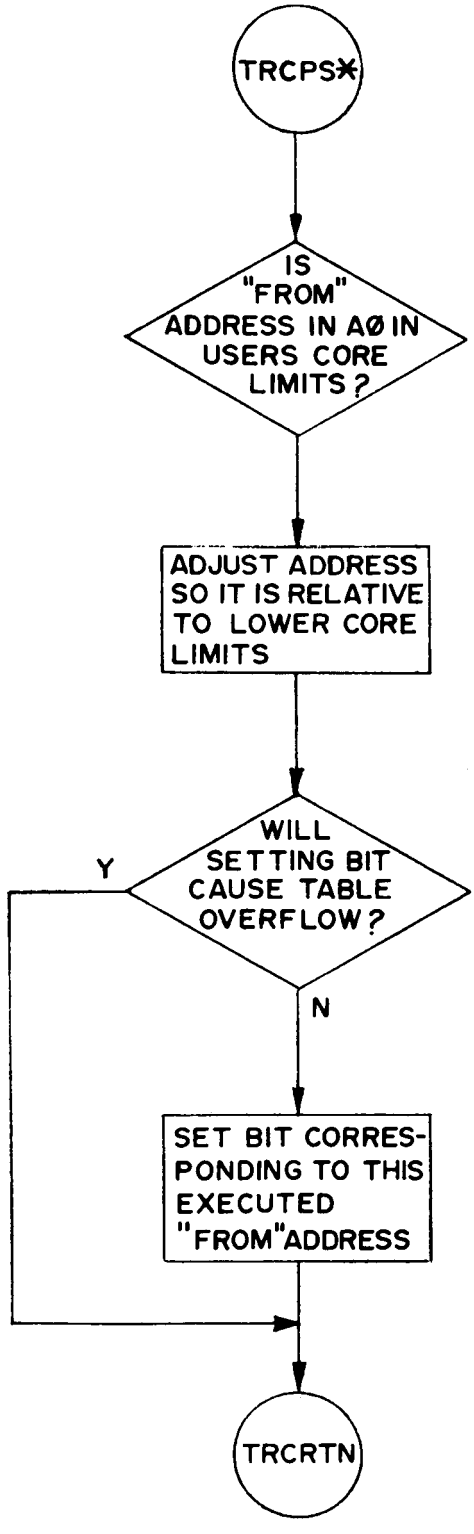
Trace 23


TRIAIJ
CORRECT I/O STATE SWITCH TRSWI TO AL - LOW INTERRUPTS
TRILM2
TRISLJ
CONSTRUCT STORE ADDRESS OF SLJ
GET ADDRESS OF INSTRUCTION CAUSING ERROR INTERRUPT & (PSIM)
STORE ADDRESS OF ERROR WHERE SLJ SHOULD HAVE IF EX ON IT WAS NOT USED
TRILM2

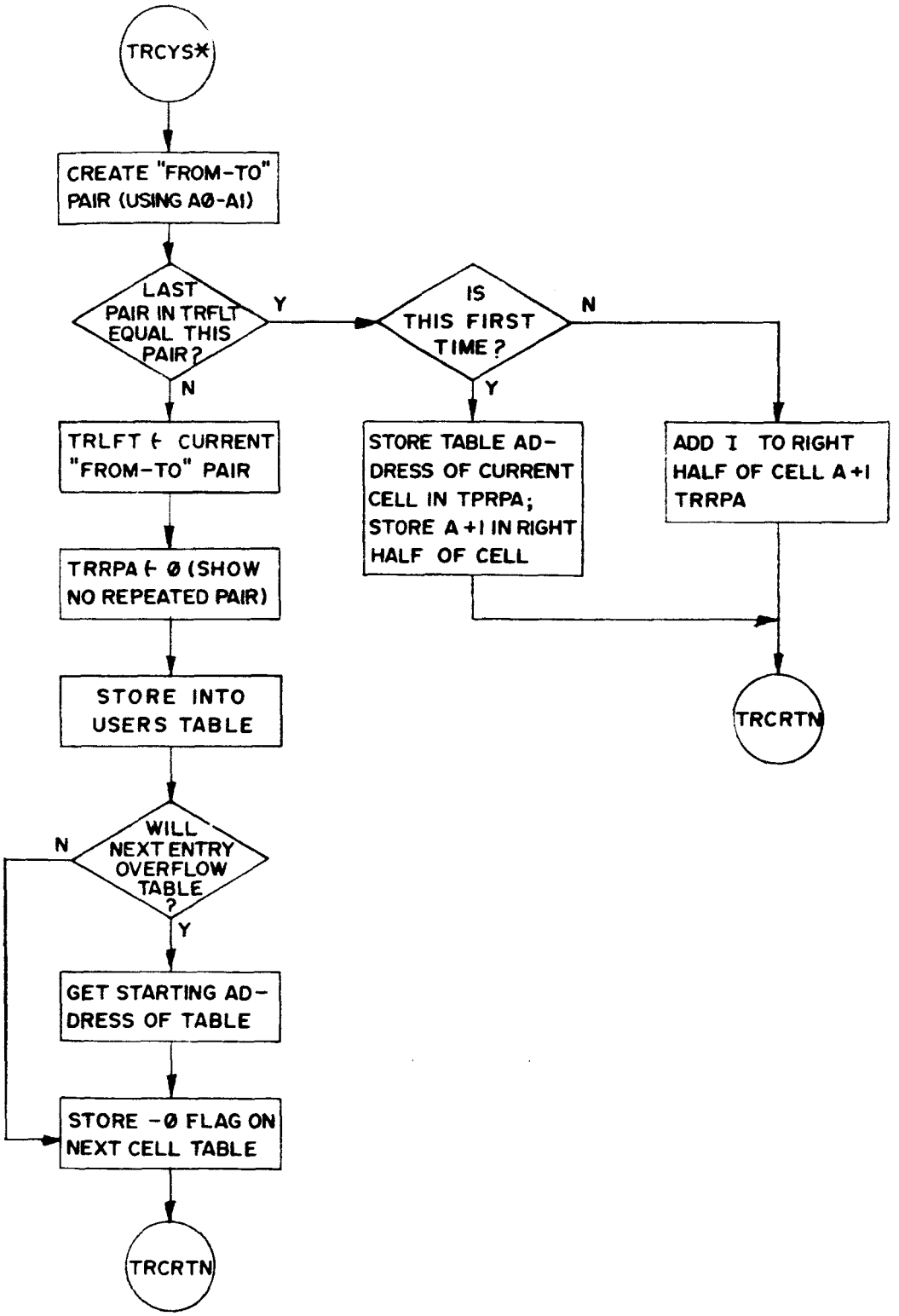
Trace 24


TRILMJ
OBTAIN A-FIELD OF LMJ
IS A0 OR A1 INDICATED ?
Y
TRILM3
N
SET B-REGISTER TO (PSIM) + ADDRESS OF INSTRUCTION CAUSING ERROR INTERRUPT, NOT OF EX OPERATING ON LMJ
TRILM2
RESTORE "TO" ADDRESS IN A1
RESTORE A2 & A3
TRXP1
TRILM3
OBTAIN ADDRESS OF INSTRUCTION CAUSING ERROR INTERRUPT AND ADD P-REGISTER BIAS TO IT
STORE BIASED ADDRESS INTO A0 OR A1 IMAGE AS REQUIRED
TRILM2

Trace 25


TRCPS*
IS "FROM" ADDRESS IN AØ IN USERS CORE LIMITS?
ADJUST ADDRESS SO IT IS RELATIVE TO LOWER CORE LIMITS
WILL SETTING BIT CAUSE TABLE OVERFLOW?
Y
N
SET BIT CORRESPONDING TO THIS EXECUTED "FROM" ADDRESS
TRCRTN

Trace 26


TRCYS*
CREATE "FROM-TO" PAIR (USING A0-A1)
LAST PAIR IN TRFLT EQUAL THIS PAIR?
Y
N
IS THIS FIRST TIME?
N
Y
TRLFT ← CURRENT "FROM-TO" PAIR
STORE TABLE ADDRESS OF CURRENT CELL IN TPRPA; STORE A+1 IN RIGHT HALF OF CELL
ADD 1 TO RIGHT HALF OF CELL A+1 TRRPA
TRRPA ← 0 (SHOW NO REPEATED PAIR)
STORE INTO USERS TABLE
TRCRTN
WILL NEXT ENTRY OVERFLOW TABLE?
N
Y
GET STARTING ADDRESS OF TABLE
STORE -0 FLAG ON NEXT CELL TABLE
TRCRTN

Trace 27

```
 LIAM    ASM,*      TRACE
ASSEMBLED BY UNIVAC 1107 SLEUTH II DATED- JANUARY 3,1964
THIS ASSEMBLY WAS DONE ON 01 APR 65 AT 08:06:14
000001                    000000000014          A0        EQU       12
000002                    000000000015          A1        EQU       13
000003                    000000000016          A2        EQU       14
000004                    000000000017          A3        EQU       15
000005                    000000000020          A4        EQU       16
000006                    000000000012          B10       EQU       10
000007                    000000000013          B11       EQU       11
000008                    000000000103          MLC       EQU       67
000009                    000000000101          R1        EQU       65
000010                    000000000103          R3        EQU       67
000011                    777777777777          FILL      EQU       -0
000012                                          $(1)
000013                                                    INFO      5    1          . INDEPENDENT TO AVOID OVERLAY
000014      01    000000  74 13 13 00 0 001015  TPFA      LMC       B11,*ERR$       . FIRST GENERATED LINE (ALSO GUARD LINE)
000015            000001  74 13 13 00 0 001015            LMC       B11,*ERR$       . GUARD LINE FOR ILLEGAL ENTRANCE
000016                                          /
000017                                          . SET UP PARAMATERS SUBROUTINES
000018                                          .
000019            000002  74 04 00 00 0 000002  TRIAM     J         $               . A0=INTERRUPT ADDRESS SWITCH  A1=MODE
                    *   20-35 +CC   1
000020            000003  74 00 00 00 0 000056            JZ        A0,TRCERR       . ADDRESS SWITCH ILLEGAL
                    *   20-35 +CC   1
000021            000004  06 01 14 00 0 000411            S,1       A0,TRIC         .SWITCH IS ACTUALLY INTERRUPT CODE TRG
                    *   20-35 +CC   1
000022            000005  54 16 01 00 0 000004            TLE,14    A1,4
000023            000006  60 00 00 00 0 000015            TP        A1
000024            000007  74 04 00 00 0 000056            J         TRCERR          . MODE ERROR
                    *   20-35 +CC   1
000025            000010  27 16 1E 00 0 000412            L,4       A2,TRCRTN
                    *   20-35 +CC   1
000026            000011  27 16 17 00 0 000373            L,4       A3,TRXP
                    *   20-35 +CC   1
000027            000012  06 01 16 00 0 000430            S,1       A2,TRSKP1       . MODE 0 = JUMPS ONLY
                    *   20-35 +CC   1
000028            000013  06 01 1E 00 0 000424            S,1       A2,TRNSK1
                    *   20-35 +CC   1
000029            000014  50 00 00 00 0 000015            TZ        A1
000030            000015  06 01 17 00 0 000430            S,1       A3,TRSKP1       . MODE 1 = JUMPS AND SKIPS
                    *   20-35 +CC   1
000031            000016  53 16 01 00 0 000002            TNE,14    A1,2
000032            000017  06 01 17 00 0 000424            S,1       A3,TRNSK1       . MODE 2 = ALL INSTRUCTIONS
                    *   20-35 +CC   1
000033            000020  74 04 00 00 0 000002            J         TRIAM
                    *   20-35 +CC   1
000034                                          .
000035            000021  74 04 00 00 0 000021  TRLIM     J         $               . A0=LOW CORE LIMITS,A1=UPPER LIMITS
                    *   20-35 +CC   1
000036            000022  54 00 01 00 0 000014            TLE       A1,A0
000037            000023  73 01 00 00 0 000044            DSC       A0,36           . CORRECT LIMITS REVERSAL
000038            000024  54 00 00 00 0 000076            TLE       A0,(0200000)    . TEST WITHIN PHYSICAL CORE
                    *   20-35 +CC   0
000039            000025  54 16 01 00 0 000000            TLE,14    A1,0
000040            000026  74 04 00 00 0 000056            J         TRCERR
```

```
                     *  ERR-35 +CC   1
000041   000027  06 00 14 00 0 000033          S       A0+TRCLCS       . STORE INTO GENERALLY AVAIL CELL
                     *  ERR-35 +CC   0
000042   000030  20 16 01 00 0 000001          AL+14   A1+1            . ADJUST UPPER LIMIT FOR +1 LOGIC
000043   000031  06 00 1E 00 0 000032          S       A2+TRCLCS
                     *  ERR-35 +CC   0
000044   000032  74 04 00 00 0 000021          J       TRLIM
                     *  ERR-35 +CC   1
000045                                    .
000046   000033  74 04 00 00 0 000033  TRBIN   J       *               . TRACE BIN ADDRESS=A0  LENGTH=A1
                     *  ERR-35 +CC   1
000047   000034  54 00 00 00 0 000076          TLE     A0+(0200000)    . TEST IF PROPER
                     *  ERR-35 +CC   0
000048   000035  54 16 00 00 0 000000          TLE+14  A0+0
000049   000036  74 04 00 00 0 000056          J       TRCERR
                     *  ERR-35 +CC   1
000050   000037  06 01 14 00 0 000073          S+1     A0+TRTBS        . BIN START ADDRESS IN (1+FILL)
                     *  ERR-35 +CC   0
000051   000040  20 00 01 00 0 000014          AL      A1+A0           . OBTAIN END ADDRESS+1
000052   000041  54 00 02 00 0 000077          TLE     A2+(0200001)    . (NO STORAGE WILL OCCUR IN END +1)
                     *  ERR-35 +CC   0
000053   000042  54 16 02 00 0 000000          TLE+14  A2+0
000054   000043  74 04 00 00 0 000056          J       TRCERR
                     *  ERR-35 +CC   1
000055   000044  06 01 1E 00 0 000072          S+1     A2+TRTBE        . STORE END+1 IN (1+FILL)
                     *  ERR-35 +CC   0
000056   000045  74 04 00 00 0 000033          J       TRBIN
                     *  ERR-35 +CC   1
000057                                    .
000058   000046  74 04 00 00 0 000046  TRCFTX  J       *               . TRCFTX SHOULD BE USED BEFORE OTHERS
                     *  ERR-35 +CC   1
000059                                                                    . A0=NUMBER PARAM LINES
000060   000047  11 16 01 00 0 000001          LA+14   A1+1
000061   000050  24 00 15 00 0 000013          A       A1+B11
000062   000051  06 01 15 00 0 000254          S+1     A1+TREX         . SIMULATE **FROM** ADDRESS
                     *  ERR-35 +CC   1
000063   000052  20 00 00 00 0 000013          AL      A0+B11
000064   000053  06 01 15 00 0 000034          S+1     A1+TRJTO        . SIMULATE **TO** ADDRESS
                     *  ERR-35 +CC   0
000065   000054  06 01 15 00 0 000065          S+1     A1+TRERRX       . SET UP ERROR EXIT IN CASE
                     *  ERR-35 +CC   1
000066   000055  74 04 00 00 0 000046          J       TRCFTX
                     *  ERR-35 +CC   1
000067                                    .
000068   000056  27 01 14 00 0 000043  TRCERR  L+1     A0+TRSB11       . **(N)TRCS CALL ERROR FROM XXXXXX
                     *  ERR-35 +CC   0
000069   000057  25 16 14 00 0 000001          AN+14   A0+1            . CALL LINE
000070   000060  74 13 13 00 0 001063          E+EQ    6+KADD+6
         000061  06 0005 000031
                     *  IP-35 +CC   0
000071   000062  74 13 13 00 0 001032          K+TYPE  K1+(K1A-K1)*6
         000063  00 0044 000024
                     *  IP-35 +CC   0
000072   000064  27 00 13 00 0 000043          L       B11+TRSB11      . RESET B11 TO USERS CALL ADDRESS+1
                     *  ERR-35 +CC   0
000073   000065  74 04 00 00 0 177777  TRERRX  J       FILL            . RETURN MAIN PROGRAM  NO TRACE
```

```
000074                                          .
000075          000066  74 04 06 00 0 000066   TRPAK   J       $               . FOR IV PAK INTO SLII PARAMS
                *  20-35 +CC  1
000076          000067  05 02 00 00 0 000064           SZ,2    TRNPAR          . ILLEGAL INTERRUPT CODE NAME FOR
                *  20-35 +CC  0                                                 . LATER POSSIBLE CALL ERROR
000077
000078          000070  27 00 14 13 1 000001           L       A0,*1,B11
000079          000071  06 01 14 00 0 000064           S,1     A0,TRNPAR       . MODE
                *  20-35 +CC  0
000080          000072  27 00 14 13 0 000002           L       A0,2,B11
000081          000073  06 02 14 00 0 000065           S,2     A0,TRNPAR+1     . LOWER CORE LIMITS
                *  20-35 +CC  0
000082          000074  27 00 14 13 0 000003           L       A0,3,B11
000083          000075  06 01 14 00 0 000065           S,1     A0,TRNPAR+1     . UPPER CORE LIMITS
                *  20-35 +CC  0
000084          000076  27 00 14 13 0 000004           L       A0,4,B11
000085          000077  06 02 14 00 0 000066           S,2     A0,TRNPAR+2     . BIN ADDRESS
                *  20-35 +CC  0
000086          000100  27 00 14 13 1 000005           L       A0,*5,B11
000087          000101  06 01 14 00 0 000066           S,1     A0,TRNPAR+2     . BIN LENGTH
                *  20-35 +CC  0
000088          000102  74 04 06 00 0 000066           J       TRPAK
                *  20-35 +CC  1
000089                                          /
000090                                          . ENTRY POINTS AND INTERRUPT CODING INITIALIZATION
000091                                          .
000092          000103  06 00 13 00 0 000043   TRCS*   S       B11,TRSB11
                *  20-35 +CC  0
000093          000104  06 00 12 00 0 000042           S       B10,TRSB10
                *  20-35 +CC  0
000094          000105  23 16 01 00 0 000003           L,64    R1,TRVE-TRV     . SEARCH TRACE VECTOR TABLE
000095          000106  27 00 14 00 0 000100           L       A0,(1,-1)
                *  20-35 +CC  0
000096          000107  27 02 15 13 0 000000           L,2     A1,,B11         . INTERRUPT ROUTINE SWITCH
000097          000110  62 02 01 14 2 000022           SE,2    A1,TRV+1,*A0    . TEST FOR STD. INTERRUPT CODE NAME
                *  20-35 +CC  0
000098          000111  05 00 00 00 0 000014           SZ      A0              . NOT STANDARD
000099          000112  27 01 14 14 0 000021           L,1     A0,TRV,A0
                *  20-35 +CC  0
000100                                          . PRIME LINK
000101          000113  74 13 12 14 0 000000   TRCL1   LMC     B10,,A0         . GET NO. PARAM LINES
000102          000114  72 01 00 00 0 000046           SLC     TRCETX
                *  20-35 +CC  1
000103          000115  74 13 12 12 0 000000   TRCL2   LMC     B10,,B10        . GET INTERRUPT LOCATION
000104          000116  27 01 15 13 0 000000           L,1     A1,,B11         . GET MODE
000105          000117  72 01 00 00 0 000002           SLC     TRIAM           .
                *  20-35 +CC  1
000106          000120  27 02 14 13 0 000001           L,2     A0,1,B11        . LOWER CORE ADDRESS
000107          000121  27 01 15 13 0 000001           L,1     A1,1,B11        . UPPER CORE ADDRESS
000108          000122  72 01 00 00 0 000021           SLC     TRLIM           . CHECK THEM AND SET UP
                *  20-35 +CC  1
000109          000123  74 13 12 12 0 000000   TRCL3   LMC     B10,,B10        . GET BIN ADDRESS AND LENGTH
000110          000124  72 01 00 00 0 000033           SLC     TRBIN           . CHECK THEM AND SET UP
                *  20-35 +CC  1
000111          000125  74 13 12 12 0 000000   TRCL4   LMC     B10,,B10        . SET MITRCS LOCATION
000112                                          TRCL4A                          . TRACE RE-ENTRY POINT AFTER SET UP
```

```
000113          000126  74 13 13 00 0 001007            MSSEA   MITPC$++ ( SLU TRACES )
                000127  000000 000003
                000130  7204000000356
                    *   <0-35 +CC  1
000114          000131  27 00 13 00 0 000043            L       B11+TRSB11
                    *   <0-35 +CC  0
000115          000132  06 02 13 00 0 000074            S+4     B11+TRSW1       . INITIALLIZE I/O SWITCH STATE TO AAIU
                    *   <0-35 +CC  0
000116          000133  27 00 12 00 0 000042            L       B10+TRSB10
                    *   <0-35 +CC  0
000117          000134  27 01 15 00 0 000034            L+1     A1+TRJTO
                    *   <0-35 +CC  0
000118          000135  74 04 00 00 0 000373            J       TRXP            . SIMULATE TRACED CALLING SEQUENCE
                    *   <0-35 +CC  1
000119                                          .
000120                                          TRCUI
000121                                          . LINK1
000122          000136  27 16 14 00 0 000002            L+14    AU+2            . 2 PARAM LINES
000123          000137  74 13 12 12 0 000000            LMC     B10,,B10
000124                                          . LINK2
000125          000140  27 02 14 13 0 000000            L+4     AU,,B11         . GET INTERRUPT SWITCH
000126          000141  74 00 00 00 0 000056            JZ      AU+TRCERR       . FOR IV CALL OR UNDEFINED LABEL ERRORS
                    *   <0-35 +CC  1
000127          000142  74 13 12 12 0 000000            LMC     B10,,B10
000128                                          . LINK3
000129          000143  74 13 12 12 0 000001            LMC     B10+1+B10       . SKIP OVER SLU TRBIN
000130                                          . LINK4
000131          000144  74 13 12 12 0 000000            LMC     B10,,B10
000132                                          .
000133                                          TRCYI
000134                                          . LINK1
000135          000145  27 16 14 00 0 000003            L+14    AU+3
000136          000146  74 13 12 12 0 000000            LMC     B10,,B10
000137                                          . LINK2
000138          000147  27 16 14 00 0 000527            L+14    AU+TRCY$        . INTERRUPT CODE ADDRESS(ALSO SWITCH NAME
                    *   <0-35 +CC  1
000139          000150  74 13 12 12 0 000000            LMC     B10,,B10
000140                                          . LINK3
000141          000151  27 02 14 13 0 000002            L+4     AU+2+B11
000142          000152  06 01 14 00 0 000067            S+1     AU+TRNSA        . INITIALIZE NEXT SEQ. ADDRESS
                    *   <0-35 +CC  0
000143          000153  27 01 15 13 0 000002            L+1     A1+2+B11        . LENGTH
000144          000154  74 13 12 12 0 000000            LMC     B10,,B10
000145                                          . LINK4
000146          000155  05 00 00 00 0 000014            SZ      AU
000147          000156  02 00 00 00 0 000070            SN      AU+TRLFT        . -0 IMPOSSIBLE LAST **FROM-TO** PAIR
                    *   <0-35 +CC  0
000148          000157  74 13 12 12 0 000000            LMC     B10,,B10
000149                                          TRCPI
000150                                          . LINK1
000151          000160  27 16 14 00 0 000003            L+14    AU+3            . 3 PARAM LINES
000152          000161  74 13 12 12 0 000000            LMC     B10,,B10
000153                                          . LINK2
000154          000162  27 16 14 00 0 000561            L+14    AU+TRCP$        . INTERUPT CODE ADDRESS(ALSO SWITCH NAME
                    *   <0-35 +CC  1
000155          000163  74 13 12 12 0 000000            LMC     B10,,B10
```

```
000156                                          . LINK3                                 . A0=LOW CORE ADDRESS  A1= UPPER
000157          000164  25 00 15 00 0 000014           AR      A1,A0                   . DEVELOP CORE RANGE / 36 (BITS)
000158          000165  24 16 15 00 0 000107           A,14    A1,36+35                . COVERED QUOTIENT+1
000159          000166  73 03 01 00 0 000044           DSL     A1,36
000160          000167  34 16 01 00 0 000044           DI,14   A1,36                    . CREATE BIN LENGTH IN A1
000161          000170  27 02 14 13 0 000002           L,2     A0,2,B11
000162          000171  06 01 14 00 0 000577           S,1     A0,TRP1                 . INITIALIZE
                    *   20-35 +CC   1
000163          000172  06 01 14 00 0 000600           S,1     A0,TRP2
                    *   20-35 +CC   1
000164          000173  06 01 14 00 0 000571           S,1     A0,TRP3
                    *   20-35 +CC   1
000165          000174  74 13 12 12 0 000000           LMJ     B10,*B10
000166                                          . LINK4
000167          000175  74 13 12 12 0 000000           LMJ     B10,*B10
000168                                          .
000169                                          .
000170          000176  06 00 13 00 0 000043  TRCX$*   . LMJ     B11,TRCX$               . DISABLE ANY TRACE ROUTINE
                    *   20-35 +CC   0                S       B11,TRSB11              . LMJ USUALLY WAS UNDER TRACE
000171          000177  74 13 13 00 0 001010           LMJ     B11,=REA$
000172          000200  000000 000003                 +       0,=ITRC$
000173          000201  27 00 13 00 0 000043           L       B11,TRSB11              . SET B11 TO AID USER DEBUG
                    *   20-35 +CC   0
000174          000202  74 04 00 13 0 000000           J       ,B11
000175                                          .
000176          000203  24 16 13 00 0 000001  NTRCX*   A,14    B11,1                   . BYPASS NOP S LINE OF FOR IV
000177          000204  74 04 00 00 0 000176           J       TRCX$
                    *   20-35 +CC   1
000178                                          .
000179                                          NTRC*                                   . FOR IV CALL NTRC(I,M,$X,$Y,NBIN,L)
000180          000205  06 00 13 00 0 000043           S       B11,TRSB11
                    *   20-35 +CC   0
000181          000206  06 00 12 00 0 000042           S       B10,TRSB10
                    *   20-35 +CC   0
000182          000207  72 01 00 00 0 000066           SLJ     TRPAK
                    *   20-35 +CC   1
000183          000210  27 00 14 13 1 000000           L       A0,*0,B11               . INTERRUPT ROUTINE NUMBER
000184          000211  54 16 00 00 0 000004           TLE,14  A0,TRVE-TRV+1           . TEST FOR STD. ROUTINE
000185          000212  54 16 00 00 0 000001           TLE,14  A0,1
000186          000213  05 00 00 00 0 000014           SZ      A0                      . ILLEGAL SWITCH
000187          000214  27 01 14 14 0 000021           L,1     A0,TRV,A0               . GET INTERRUPT INITIALIZATION ADDRESS
                    *   20-35 +CC   0
000188          000215  74 13 12 14 0 000000           LMJ     B10,,A0                 . USE LINK1 TO BYPASS IT
000189          000216  27 16 14 00 0 000007           L,14    A0,7                    . NO. PARAM LINES+ NOP LINE
000190          000217  72 01 00 00 0 000046           SLJ     TRCFTX
                    *   20-35 +CC   1
000191          000220  27 16 13 00 0 000064           L,14    B11,TRNPAR              . GIMMICK TRNPAR REFERENCING
                    *   20-35 +CC   0
000192          000221  74 04 00 00 0 000115           J       TRCL2                   . CONTINUE WITH TRC$ LINK2
                    *   20-35 +CC   1
000193          000222  27 16 14 00 0 000001  NTRCR*   L,14    A0,1                    . COUNT FORIV NOP LINE AS 1 PARAM
000194          000223  74 04 00 00 0 000225           J       TRCP$1
                    *   20-35 +CC   1
000195          000224  05 00 00 00 0 000014  TRCR$*   SZ      A0                      . NO PARAMS BELOW SLII CALL
000196          000225  72 01 00 00 0 000046  TRCP$1   SLJ     TRCFTX                  . SET UP ERROR EXIT AND RETURN LINE
                    *   20-35 +CC   1
000197          000226  06 00 13 00 0 000043           S       B11,TRSB11
```

```
                    *  <0-35 +CC  0
000198  000227  27 01 14 00 0 000411          L,1       AU,TRIC
                    *  <0-35 +CC  1
000199  000230  53 16 00 00 0 177777          TNE,14    AU,FILL          . TEST FOR INITIALIZED INTERRUPT ROUTINE
000200  000231  74 04 00 00 0 000056          J         TRCERR           . NO INTERRUPT NAME = NO PRIOR TRACE
                    *  <0-35 +CC  1
000201  000232  06 00 12 00 0 000042          S         B10,TRSB10
                    *  <0-35 +CC  0
000202  000233  74 04 00 00 0 000126          J         TRCL4A           . TAKE LINK 4 AS RE-ENTRY POINT
                    *  <0-35 +CC  1
000203                                        /
000204                                        . SOFTWARE HEART OF TRACE ROUTINE
000205                                        .
000206                                  TRC   . ASSUMES AU IS ADDRESS NSI,TRSAU CONTAINS SAVED AU, TRACE DISENGAGED
000207  000234  06 01 14 00 0 000074          S,1       AU,TRSW1         . INITIALIZE SKIP SWITCH
                    *  <0-35 +CC  0
000208  000235  06 01 14 00 0 000254          S,1       AU,TREX          . SET UP NEXT ADDRESS
                    *  <0-35 +CC  1
000209  000236  27 00 14 14 0 000000          L         AU,,A0           . BRING NEXT INSTRUCTION
000210  000237  73 02 00 00 0 000032          SSL       AU,36-10         . SHIFT FOR FUNCTION CODE
000211  000240  72 13 00 00 0 000241          PAIJ      $+1              . AVOID TRACING PARASITES
                    *  <0-35 +CC  1
000212                                                                   . AND I/O INTERRUPT BETWEEN ETMJ AND JUMP
000213  000241  04 00 03 00 0 000041          S         PLC,TRSR3        . SAVE R3 FROM TRACE DESTRUCTION
                    *  <0-35 +CC  0
000214                                  TRC0
000215  000242  52 16 00 00 0 001713          TE,14     AU,01713         . LMJ OP CODE
000216  000243  53 16 00 00 0 001641          TNE,14    AU,01641         . SLJ OP CODE
000217  000244  74 04 00 00 0 000262          J         SLJLMJ
                    *  <0-35 +CC  1
000218  000245  52 16 00 00 0 001653          TE,14     AU,01653
000219  000246  53 16 00 00 0 001707          TNE,14    AU,01707         . AAIJ OP CODE
000220  000247  74 04 00 00 0 000264          J         TRPAJ            . PAIJ OR AAIJ OP CODE
                    *  <0-35 +CC  1
000221  000250  53 16 00 00 0 001650          TNE,14    AU,01650         . EX OP CODE
000222  000251  74 04 00 00 0 000276          J         TRXN             . CAUTION ON EX CHAIN
                    *  <0-35 +CC  1
000223  000252  27 00 14 00 0 000035  TRC1    L         AU,TRSA0
                    *  <0-35 +CC  0
000224  000253  72 12 00 00 0 000254          ETMJ      TREX             . ENGAGE TRACE FOR POSSIBLE JUMP NEXT
                    *  <0-35 +CC  1
000225                                        .
000226  000254  72 10 00 00 0 177777  TREX    EX        FILL             . *** TRACE MONITOR ***
000227                                        .
000228  000255  05 01 00 00 0 000074          SZ,1      TRSW1            . SHOW NO SKIP
                    *  <0-35 +CC  0
000229  000256  04 00 03 00 0 000041          S         R3,TRSR3         . R3 COULD CHANGE FROM STORE, LOAD,
                    *  <0-35 +CC  0
000230                                                                   . OR REPEATED SEQUENCE
000231  000257  51 01 00 00 0 000074          TNZ,1     TRSW1
                    *  <0-35 +CC  0
000232  000260  74 04 00 00 0 000421          J         TRNSKP           . NO SKIP ( THIS JUMP TRACED)
                    *  <0-35 +CC  1
000233  000261  74 04 00 00 0 000425          J         TRSKP            . SKIP OCCURED (THIS JUMP TRACED )
                    *  <0-35 +CC  1
000234                                  TRC2                             .
```

```
000235          000262  27 00 14 00 0 000035   SLOLMJ   L        AU*TRSA0
                    *   ∠0-35 +CC  0
000236          000263  72 12 00 00 1 000254            ETMJ     *TREX          • EXCUTE WITHIN EXTERNAL PROGRAM
                    *   ∠0-35 +CC  1
000237          000264  52 16 00 00 0 001653   TRPAJ    TE*14    AU*01653
000238          000265  74 04 00 00 0 000270            J        TRAAIJ         • AAIJ OP CODE
                    *   ∠0-35 +CC  1
000239          000266  05 02 00 00 0 000074   TRPAIJ   SZ*2     TRSW1          • SET SW TO PREVENT I/O ENABLE AFTER
                    *   ∠0-35 +CC  0
000240                                                                         • A TRACED PAIJ INSTRUCTION
000241          000267  74 04 00 00 0 000262            J        TRC2           • NOT PAIJ INSTRUCTIONIN SITU FOR SPEED
                    *   ∠0-35 +CC  1
000242          000270  06 02 14 00 0 000074   TRAAIJ   S*2      AU*TRSW1       • SET SW TO ALLOW I/O AFTER R3 RESTORE
                    *   ∠0-35 +CC  0
000243          000271  27 00 14 00 1 000254            L        AU*=TREX       • GET AAIJ   B*H*I*U
                    *   ∠0-35 +CC  1
000244          000272  06 00 14 00 0 000075            S        AU*TRBHIU      • SAVE BHIU
                    *   ∠0-35 +CC  0
000245          000273  27 00 14 00 0 000035            L        AU*TRSA0
                    *   ∠0-35 +CC  0
000246          000274  72 12 00 00 0 000275            ETMJ     $+1            • ALLOW TO BRING CONTROL BACK TO TRACE
                    *   ∠0-35 +CC  1
000247          000275  74 04 00 00 1 000075            J        *TRBHIU        • USE BHIU WITHOUT AAIJ
                    *   ∠0-35 +CC  0
000248                                                                         • TREX HAS FROM ADDRESS OF AAIJ
000249                                                                         • TRACE ROUTINE THINKS JUMP CAME FROM
000250                                                                         • EXTERNAL PROGRAM
000251          000276  27 00 14 00 1 000254   TRAN     L        AU*=TREX       • GET EX BHIU
                    *   ∠0-35 +CC  1
000252          000277  06 00 14 00 0 000075            S        AU*TRBHIU      • SAVE BHIU
                    *   ∠0-35 +CC  0
000253          000300  27 00 14 00 0 000035            L        AU*TRSA0       • (FILM COMPLETELY RESTORED)
                    *   ∠0-35 +CC  0
000254                                          I        DO       16 * S I*REG-1+I • SAVE B1-B15*A4
                000301  06 00 01 00 0 000044
                    *   ∠0-35 +CC  0
                000302  06 00 02 00 0 000045
                    *   ∠0-35 +CC  0
                000303  06 00 03 00 0 000046
                    *   ∠0-35 +CC  0
                000304  06 00 04 00 0 000047
                    *   ∠0-35 +CC  0
                000305  06 00 05 00 0 000050
                    *   ∠0-35 +CC  0
                000306  06 00 06 00 0 000051
                    *   ∠0-35 +CC  0
                000307  06 00 07 00 0 000052
                    *   ∠0-35 +CC  0
                000310  06 00 10 00 0 000053
                    *   ∠0-35 +CC  0
                000311  06 00 11 00 0 000054
                    *   ∠0-35 +CC  0
                000312  06 00 12 00 0 000055
                    *   ∠0-35 +CC  0
                000313  06 00 13 00 0 000056
                    *   ∠0-35 +CC  0
```

```
                000314  06 00 14 00 0 000057
                    *   20-35 +CC   0
                000315  06 00 15 00 0 000060
                    *   20-35 +CC   0
                000316  06 00 16 00 0 000061
                    *   20-35 +CC   0
                000317  06 00 17 00 0 000062
                    *   20-35 +CC   0
                000320  01 00 04 00 0 000063
                    *   20-35 +CC   0
000255          000321  10 00 04 00 1 000075   TRAN1   L       A4,@TRBHIO          . GET NEXT INSTR. AND USE BHIO
                    *   20-35 +CC   0
000256          000322  01 00 04 00 0 000075           S       A4,TRBHIO           . SAVE BHIO
                    *   20-35 +CC   0
000257          000323  73 02 04 00 0 000032           SSL     A4,36-10            . GET OP CODE
000258          000324  53 16 04 00 0 001650           TRE,14  A4,01650            . EX OP CODE
000259          000325  74 04 00 00 0 000321           J       TRXN1               . LOOP ON ANOTHER EX LEVEL
                    *   20-35 +CC   1
000260          000326  52 16 04 00 0 001707           TE,14   A4,01707            . AAIJ OP CODE
000261          000327  74 04 00 00 0 000334           J       TRXN2
                    *   20-35 +CC   1
000262          000330  01 02 04 00 0 000074           S,2     A4,TRSW1            . SET SW TO SHOW @@IJ
                    *   20-35 +CC   0
000263          000331  10 00 04 00 0 000063           L       A4,REG+15           .
                    *   20-35 +CC   0
000264          000332  72 12 00 00 0 000333           ETMJ    *+1
                    *   20-35 +CC   1
000265          000333  74 04 00 00 1 000075           J       @TRBHIO             . TREX HAS ''FROM'' ADDRESS OF 1ST EX
                    *   20-35 +CC   0
000266
000267          000334  01 00 04 00 0 000057   TRAN2   S       A4,REG+11           . SAVE SHIFTED OP CODE TO RESTORE A0
                    *   20-35 +CC   0                                               . GO TO EXERCISE B,H,I,U OF AAIJ
000268                                          I       DO      16 * L I,REG-1+I
                000335  27 00 01 00 0 000044
                    *   20-35 +CC   0
                000336  27 00 02 00 0 000045
                    *   20-35 +CC   0
                000337  27 00 03 00 0 000046
                    *   20-35 +CC   0
                000340  27 00 04 00 0 000047
                    *   20-35 +CC   0
                000341  27 00 05 00 0 000050
                    *   20-35 +CC   0
                000342  27 00 06 00 0 000051
                    *   20-35 +CC   0
                000343  27 00 07 00 0 000052
                    *   20-35 +CC   0
                000344  27 00 10 00 0 000053
                    *   20-35 +CC   0
                000345  27 00 11 00 0 000054
                    *   20-35 +CC   0
                000346  27 00 12 00 0 000055
                    *   20-35 +CC   0
                000347  27 00 13 00 0 000056
                    *   20-35 +CC   0
                000350  27 00 14 00 0 000057
```

```
                    *  20-35 +CC  0
                  000351  27 00 15 00 0 000060
                    *  20-35 +CC  0
                  000352  27 00 16 00 0 000061
                    *  20-35 +CC  0
                  000353  27 00 17 00 0 000062
                    *  20-35 +CC  0
                  000354  10 00 04 00 0 000063
                    *  20-35 +CC  0
000269            000355  74 04 00 00 0 000242          J          TRC0              . RESUME OP CODE TEST ON LAST NEST INSTR.
                    *  20-35 +CC  1
000270                                          /
000271                                          . HARDWARE HEART OF TRACE ROUTINE
000272                                          .
000273                                          TRACE$   . TRACED JUMP ENTRANCE   JUMP MAY BE IN TRACE ROUTINE ITSELF
000274            000356  74 04 00 00 0 000356          J          $                 . RETURN LINE
                    *  20-35 +CC  1
000275            000357  23 00 03 00 0 000041          L          MLC+@RSR3         . RESTORE MLC AFTER TRACE DESTRUCTION
                    *  20-35 +CC  0
000276            000360  50 02 00 00 0 000074          TZ+2       TRSW1             . TEST FOR I/O STATE REQUIRED
                    *  20-35 +CC  0
000277            000361  74 07 00 00 0 000362          AAIJ       $+1               . PERFORM REQUIRED AAIJ (DELAYED BEFORE)
                    *  20-35 +CC  1
000278            000362  06 00 14 00 0 000035          S          A0+TRSA0          . SAVE A0
                    *  20-35 +CC  0
000279            000363  06 00 15 00 0 000036          S          A1+TRSA1          . SAVE A1
                    *  20-35 +CC  0
000280            000364  27 01 15 00 0 000356          L+1        A1+TRACE$         . JUMPED **TO** ADDRESS+1
                    *  20-35 +CC  1
000281            000365  25 16 15 00 0 000001          AN+14      A1+1              . CORRECT FOR +1
000282            000366  06 01 15 00 0 000034          S+1        A1+TRJTO
                    *  20-35 +CC  0
000283            000367  54 16 01 00 0 000605          TLE+14     A1+TRLA+1         . LAST ADDRESS OF TRACE CODE
                    *  20-35 +CC  1
000284            000370  54 16 01 00 0 000000          TLE+14     A1+TRFA           . FIRST ADDRESS OF TRACE CODE
                    *  20-35 +CC  1
000285            000371  74 04 00 00 0 000373          J          TRXP              . JUMP NOT TO TRACE CODE
                    *  20-35 +CC  1
000286            000372  74 04 00 15 0 000000          J          0+A1
000287                                          .
000288                                          .
000289                                          TRXP                                 . JUMP IS TO EXTERNAL PROGRAM
000290            000373  52 16 01 00 0 001007          TE+14      A1+MSEA$          . WILL MSEA$ CHANGE MITRC$
000291            000374  53 16 01 00 0 001010          TNE+14     A1+MPEA$          . CHECK FOR RESET ERROR ACTION
000292            000375  74 04 00 00 0 000431          J          TRREA
                    *  20-35 +CC  1
000293            000376  27 00 14 00 0 000101          L          A0+(1+0)          . INTERRUPT LOCATION INDEX
                    *  20-35 +CC  0
000294            000377  06 00 16 00 0 000037          S          A2+TRSA2          . SAVE A2
                    *  20-35 +CC  0
000295            000400  25 16 15 00 0 000002          AN+14      A1+2              . JUMP-TO ADDRESS MINUS 2
000296            000401  20 16 01 00 0 000002          AD+14      A1+2              . JUMP-TO ADDRESS
000297            000402  74 04 00 00 0 000000          J          TRXP2             . CONTINUE IN ALTERNATE BANK FOR SPEED
                    *  20-35 +CC  0
000298            000403  27 01 15 00 0 000034  TRAP0   L+1        A1+TRJTO
                    *  20-35 +CC  0
```

```
000299  000404  27 00 16 00 0 000037          L       A6,TRSA2
                *  20-35 +CC  0
000300                                TRAP1
000301  000405  27 01 14 00 0 000254          L,4     AU,TREX
                *  20-35 +CC  1
000302  000406  54 00 01 00 0 000032          TLL     A1,TRCUC$
                *  20-35 +CC  0
000303  000407  54 00 01 00 0 000033          TLL     A1,TRCLC$
                *  20-35 +CC  0
000304  000410  74 04 00 00 0 000415          J       TREROM          . ''TO'' FAILS, TEST ''FROM''
                *  20-35 +CC  1
000305                                        .
000306                                        . AU=''FROM'' ADDRESS, A1=''TO'' ADDRESS  ONE MUST BE IN CORE AREA
000307  000411  72 01 00 00 0 177777  TRIC    SLC     FILL            . GO TO INTERRUPT CODING
000308                                        .
000309  000412  27 00 15 00 0 000036  TRCRTN  L       A1,TRSA1        . RESTORE A1
                *  20-35 +CC  0
000310  000413  27 01 14 00 0 000034          L,4     AU,TRJTO        . ADDRESS OF NSI
                *  20-35 +CC  0
000311  000414  74 04 00 00 0 000234          J       TRC             . RESUME TRACE
                *  20-35 +CC  1
000312                                TREROM  . TEST ''FROM'' ADDRESS FOR IN USERS CORE AREA
000313  000415  54 00 00 00 0 000032          TLL     AU,TRCUC$
                *  20-35 +CC  0
000314  000416  54 00 00 00 0 000033          TLL     AU,TRCLC$       . LOWER CORE TEST
                *  20-35 +CC  0
000315  000417  74 04 00 00 0 000412          J       TRCRTN          . OUTSIDE USERS CORE AREA
                *  20-35 +CC  1
000316  000420  74 04 00 00 0 000411          J       TRIC            . INSIDE USERS CORE AREA
                *  20-35 +CC  1
000317                                TRNSKP                          . NO SKIP OCCURED AT LAST INSTRUCTION
000318  000421  27 01 14 00 0 000254          L,4     AU,6REX
                *  20-35 +CC  1
000319  000422  20 16 00 00 0 000001          AU,14   AU,1            . CREATE NSI ADDRESS IN A1
000320  000423  06 01 15 00 0 000034          S,4     A1,TRJTO        . SET DESTINATION ADDRESS
                *  20-35 +CC  0
000321  000424  74 04 00 00 0 177777  TRNSK1  J       FILL            . FILL=TRXP FOR EDIT, TRCRTN FOR NONE
000322                                TRSKP                           . SKIP CODED TO SIMULATE A JUMP
000323  000425  27 01 14 00 0 000254          L,4     AU,TREX
                *  20-35 +CC  1
000324  000426  20 16 00 00 0 000002          AU,14   AU,2            . FORM ''TO'' ADDRESS
000325  000427  06 01 15 00 0 000034          S,4     A1,TRJTO        . INIT.
                *  20-35 +CC  0
000326  000430  74 04 00 00 0 177777  TRSKP1  J       FILL            . FILL= TRXP FOR EDIT, TRCRTN FOR NONE
000327  000431  27 01 14 13 0 000000  TRREA   L,4     AU,0,B11        . GET MREAS PARAMETER
000328  000432  52 16 00 00 0 000003          TE,14   AU,MITRC$
000329  000433  53 16 00 00 0 000000          TNE,14  AU,MIALL$
000330  000434  74 04 00 00 0 000436          J       TRREA1          . MREAS OR MSEAS WILL CANCEL TRACE MODE
                *  20-35 +CC  1
000331  000435  74 04 00 00 0 000405          J       TRXP1           . RESUME TRACE ROUTINE
                *  20-35 +CC  1
000332  000436  53 16 01 00 0 001007  TRREA1  TNE,14  A1,MSEA$
000333  000437  74 00 00 00 0 000405          JZ      AU,TRXP1        . TRACE MSEAS CALL ERROR
                *  20-35 +CC  1
000334  000440  27 00 14 00 0 000035          L       AU,TRSA0
                *  20-35 +CC  0
```

```
000335  000481  27 00 15 00 0 000036          L       A1,*TRSA1                       .
            *   20-35 +CC  0
000336  000482  74 04 00 00 1 000034          J       *TRJTO          . TAKE J TO MREA$ OR MSEA$
            *   20-35 +CC  0
000337  000483  06 00 17 00 0 000040  TRIPOS  S       A3,TRSA3        . APPLY MORE STRINGENT TESTS
            *   20-35 +CC  0
000338  000484  27 15 17 14 0 000010          L,3     A3,TRILOC-1,A0  . SET P REGISTER ADJUSTMENT
            *   20-35 +CC  0
000339  000485  06 10 17 00 0 000016          S,0     A3,PSIP         . FOR ASSUMED ERROR INTERRUPT
            *   20-35 +CC  0
000340  000486  27 00 17 14 1 000010          L       A3,*TRILOC-1,A0 . GET INTERRUPT INSTR. IN
            *   20-35 +CC  0
000341  000487  73 02 03 00 0 000032          SSL     A3,36-10        . GET OP CODE
000342  000490  24 16 15 00 0 000001          A,14    A1,1            . MAKE ADDRESS AN SLJ U-FIELD SHOULD HAVE
000343  000491  53 16 03 00 0 001641          TNE,14  A3,01641        . SLJ OP CODE
000344  000492  53 01 01 14 1 000010          TNE,1   A1,*TRILOC-1,A0 . IF = AN SLJ INTERRUPT INSTR. WAS XQT
            *   20-35 +CC  0
000345  000493  74 04 00 00 0 000500          J       TRISLJ          . ABOVE TEST ASSUMES HI-FIELDS =0
            *   20-35 +CC  1
000346  000494  27 01 15 00 0 000034          L,1     A1,TRJTO        . RESTORE JUMP-TO ADDRESS
            *   20-35 +CC  0
000347  000495  27 00 17 00 0 000040          L       A3,TRSA3
            *   20-35 +CC  0
000348  000496  53 16 02 14 1 000010          TNE,14  A2,@TRILOC-1,A0
            *   20-35 +CC  0
000349  000497  74 04 00 00 0 000462          J       TRIJMP          . NON-SLJ JUMP ON INTERRUPT WAS XQT
            *   20-35 +CC  1
000350  000460  27 00 17 00 0 000040          L       A3,TRSA3
            *   20-35 +CC  0
000351  000461  74 04 00 00 0 000403          J       TRXP0           . NO ERROR INTERRUPT ACTUALLY OCCURED
            *   20-35 +CC  1
000352  000462  27 00 15 14 1 000010  TRIJMP  L       A1,@TRILOC-1,A0 . GET INTERRUPT INSTR.
            *   20-35 +CC  0
000353  000463  73 02 01 00 0 000032          SSL     A1,36-10        . GET OP CODE
000354  000464  53 16 01 00 0 001641          TNE,14  A1,01641        . SLJ
000355  000465  74 04 00 00 0 000500          J       TRISLJ
            *   20-35 +CC  1
000356  000466  53 16 01 00 0 001713          TNE,14  A1,01713        . LMJ
000357  000467  74 04 00 00 0 000505          J       TRILMJ
            *   20-35 +CC  1
000358  000470  53 16 01 00 0 001707          TNE,14  A1,01707        . AAIJ TEST FOR INTERRUPT LOCATION
000359  000471  74 04 00 00 0 000476          J       TRIAIJ
            *   20-35 +CC  1
000360  000472  52 16 01 00 0 001653          TE,14   A1,01653        . FAIJ TEST FOR INTERRUPT LOCATION
000361  000473  74 04 00 00 0 000517          J       TRILM2
            *   20-35 +CC  1
000362  000474  05 02 00 00 0 000074          SZ,2    TRSW1           . CORRECT I/O STATE SWITCH
            *   20-35 +CC  0
000363  000475  72 13 00 00 0 000517          PAIJ    TRILM2          . BELATED PAIJ ( COULD BE TOO LATE)
            *   20-35 +CC  1
000364  000476  06 02 15 00 0 000074  TRIAIJ  S,2     A1,TRSW1        . CORRECT I/O STATE SWITCH
            *   20-35 +CC  0
000365  000477  74 07 00 00 0 000517          AAIJ    TRILM2
            *   20-35 +CC  1
000366  000500  27 01 15 14 1 000010  TRISLJ  L,1     A1,@TRILOC-1,A0 . GET SLJ'S HIU-FIELDS (ASSUME HI=0)
            *   20-35 +CC  0
```

```
000367  000501  27 01 14 00 0 000254          L,1     AU*TREX         . GET CORRECT ERROR ADDRESS
                *  20-35 +CC  1
000368  000502  24 10 14 00 0 000016          A,C     AU*PSIM         . SIMULATE P REGISTER
                *  20-35 +CC  0
000369  000503  06 01 14 15 0 000000          S,1     AU*,A1          . STORE 18 CORE OR 36 FILM BITS
000370                                                                  . A1=20 OR LESS IS ILLEGAL (SEE LCBCUR)
000371  000504  74 04 00 00 0 000517          J       TRILM2
                *  20-35 +CC  1
000372  000505  27 00 15 14 1 000010  TRILMJ  L       A1,*TRILOC-1,A0 . GET LMJ INSTR.
                *  20-35 +CC  0
000373  000506  73 00 01 00 0 000032          SSC     A1,26
000374  000507  73 02 01 00 0 000040          SSC     A1,32           . GET A-FIELD OF LMJ
000375  000510  52 16 01 00 0 000014          TE,14   A1,A0           .
000376  000511  53 16 01 00 0 000015          TNE,14  A1,A1
000377  000512  74 04 00 00 0 000523          J       TRILM3          . A0 OR A1 IMAGES MUST BE CHANGED
                *  20-35 +CC  1
000378  000513  27 00 14 15 0 000000  TRILM1  L       AU*,A1          . CORRECT B0-B11,B14,OR B15
000379  000514  26 01 14 00 0 000254          LXM,1   AU*TREX         . BRING DOWN CORRECT ERROR ADDRESS
                *  20-35 +CC  1
000380  000515  24 10 14 00 0 000016          A,C     AU*PSIM         . SIMULATE P REGISTER
                *  20-35 +CC  0
000381  000516  06 00 14 15 0 000000          S       AU*,A1
000382  000517  27 01 15 00 0 000034  TRILM2  L,1     A1*TRUTO
                *  20-35 +CC  0
000383  000520  27 00 16 00 0 000037          L       A2*TRSA2
                *  20-35 +CC  0
000384  000521  27 00 17 00 0 000040          L       A3*TRSA3
                *  20-35 +CC  0
000385  000522  74 04 00 00 0 000405          J       TRXP1           . RESUME WITH USER LIMITS TESTS
                *  20-35 +CC  1
000386  000523  27 01 14 00 0 000254  TRILM3  L,1     AU*TREX         . BRING DOWN CORRECT ADDRESS
                *  20-35 +CC  1
000387  000524  24 10 14 00 0 000016          A,C     AU*PSIM         . SIMULATE P REGISTER
                *  20-35 +CC  0
000388  000525  72 10 00 15 0 000003          EX      TRILM4-A0,A1    . CHANGE A0 OR A1 IMAGE
                *  20-35 +CC  0
000389  000526  74 04 00 00 0 000517          J       TRILM2
                *  20-35 +CC  1
000390
000391                                  /
000392                                  . THE FOLLOWING ROUTINES ARE THE STD. INTERRUPT ROUTINES
000393                                  .
000394                                  TRCY$*                          . STORE CYCLIC -- "FROM-TO" OR REPEAT
000395  000527  74 04 00 00 0 000527          . A0=FROM ADDRESS  A1=TO ADDRESS
                *  20-35 +CC  1                J       $               . ACCOMMODATE SLJ
000396  000530  73 00 01 00 0 000022          SSC     A1,18
000397  000531  73 01 00 00 0 000066          DSC     A0,18+36        . A0=FROM-TO PAIR
000398  000532  27 00 15 00 0 000067          L       A1,TRNSA        . NEXT ADDRESS FOR STORAGE
                *  20-35 +CC  0
000399  000533  53 00 00 00 0 000070          TNE     AU*TRLFT        . IS LAST PAIR EQU CURRENT PAIR
                *  20-35 +CC  0
000400  000534  74 04 00 00 0 000546          J       TREQU
                *  20-35 +CC  1
000401  000535  01 00 00 00 0 000070          SA      AU*TRLFT        . SAVE LAST PAIR
                *  20-35 +CC  0
000402  000536  05 00 00 00 0 000071          S2      TRRPA           . SET REPEAT COUNT/SWITCH
```

```
                    *  ∠0-35 +CC  0
000403  000537  06 00 14 15 2 000000  TRCY1   S       A0,,*A1          • STORE IN TRCTBL
000404  000540  55 00 01 00 0 000072          TG      A1,TRTBE         • SEE IF END TABLE WILL BE EXCEED
                    *  ∠0-35 +CC  0
000405  000541  27 00 15 00 0 000073          L       A1,TRTBS
                    *  ∠0-35 +CC  0
000406  000542  06 00 15 00 0 000067          S       A1,TRNSA
                    *  ∠0-35 +CC  0
000407  000543  11 16 00 00 0 000000          LN,14   A0,0             • -0
000408  000544  06 00 14 15 0 000000          S       A0,,A1           • STORE NEG 0 MARK BETWEEN CYCLES
000409  000545  74 04 00 00 0 000527          J       TRCY$            • RETURN
                    *  ∠0-35 +CC  1
000410  000546  50 00 00 00 0 000071  TREQU   TZ      TRRPA            • IS THIS FIRST EQU FOR PAIR
                    *  ∠0-35 +CC  0
000411  000547  74 04 00 00 0 000554          J       TREQU1           • NOT FIRST
                    *  ∠0-35 +CC  1
000412  000550  06 00 15 00 0 000071          S       A1,TRRPA         • SAVE REPEAT ADDRESS
                    *  ∠0-35 +CC  0
000413  000551  27 16 14 00 0 000001          L,14    A0,1             • REPEAT COUNT INITIALIZED
000414  000552  06 01 14 15 2 000000          S,1     A0,,*A1          • PRESERVE -0 LEFT HALF FOR REPEAT FLAG
000415  000553  74 04 00 00 0 000540          J       TRCY1+1
                    *  ∠0-35 +CC  1
000416  000554  27 00 15 00 0 000071  TREQU1  L       A1,TRRPA
                    *  ∠0-35 +CC  0
000417  000555  27 16 14 00 0 000001          L,14    A0,1             • INCREMENT LAST CNT
000418  000556  24 01 14 15 0 000000          A,1     A0,,A1           • REPEAT MODULUS 2*/18
000419  000557  06 01 14 15 0 000000          S,1     A0,,A1           • LEAVE LEFT HALF A -0 TO FLAG REPEAT
000420  000560  74 04 00 00 0 000527          J       TRCY$            • RETURN
                    *  ∠0-35 +CC  1
000421                                       /
000422                                       • TRACED ORIGIN POINT SUBROUTINE TO TRACE
000423                                       • A0= FROM ADDRESS        A0 AND A1 ARE DESTRUCTIBLE
000424                                       TRCP$*
000425  000561  74 04 00 00 0 000561          J       $                • ACCOMMODATE SLJ
                    *  ∠0-35 +CC  1
000426  000562  54 00 00 00 0 000032          TLE     A0,TRCUC$        • UPPER CORE TEST
                    *  ∠0-35 +CC  0
000427  000563  54 00 00 00 0 000033          TLE     A0,TRCLC$        • LOWER CORE TEST
                    *  ∠0-35 +CC  0
000428  000564  74 04 00 00 0 000561          J       TRCP$            • OUTSIDE USERS CORE-RETURN
                    *  ∠0-35 +CC  1
000429  000565  06 00 1∠ 00 0 000037          S       A2,TRSA2
                    *  ∠0-35 +CC  0
000430  000566  25 01 14 00 0 000033          AN,1    A0,TRCLC$          • OBTAIN ADDRESS RELATIVE TO LOWER
                    *  ∠0-35 +CC  0
000431  000567  73 03 00 00 0 000044          DSL     A0,36
000432  000570  34 16 00 00 0 000044          OI,14   A0,36
000433  000571  27 16 1∠ 14 0 177777  TRP3    L,14    A2,FILL,A0
000434  000572  55 01 02 00 0 000072          TG,1    A2,TRTBE         • TEST TO AVOID TABLE OVERFLOW
                    *  ∠0-35 +CC  0
000435  000573  74 04 00 00 0 000601          J       TRP4             • AVOID
                    *  ∠0-35 +CC  1
000436  000574  06 00 17 00 0 000040          S       A3,TRSA3
                    *  ∠0-35 +CC  0
000437  000575  27 00 1∠ 00 0 000102          L       A2,( 1*/35 )
                    *  ∠0-35 +CC  0
```

```
000438          000376  73 02 02 15 0 000000            SSL     A2,+A1
000439          000377  60 00 02 14 0 177777  TRP1      OR      A2,FILL,A0          . SET BIT FOR TRACED ORIGIN POINT
000440          000000  06 00 17 14 0 177777  TRP2      S       A0,FILL,A0          . STORE IN TABLE
000441                                        TRP4
000442          000001  27 00 16 00 0 000037            L       A2,TRSA2
                    *   20-35 +CC  0
000443          000002  27 00 17 00 0 000040            L       A0,TRSA3
                    *   20-35 +CC  0
000444          000003  74 04 00 00 0 000561            J       TRCPS               . RETURN
                    *   20-35 +CC  1
000445          000004  777777777777  TRLA  -0                                    . LAST ADDRESS OF TRACE CODE
000446                                        /
000447                                        $(0)
000448                                        INFO    6    0                      . INDEPENDENT TO AVOID OVERLAY
000449                                        TRAP2
000450      00  000000  56 01 01 14 2 000301            TN,1    A1,0301,*A0         . TEST INTERRUPT LOCATION'S  U-FIELD
000451                                                                            . AGAINST JUMP-TO ADDRESS RANGE
000452          000001  57 01 01 14 2 000302            TNN,1   A1,0302,*A0        . MEMORY LOCKOUT .
000453          000002  74 04 00 00 0 000443            J       TRIPOS              . POSSIBLE ERROR INTERRUPT JUMP
                    *   20-35 +CC  1
000454          000003  56 01 01 14 2 000303            TN,1    A1,0303,*A0         . CHAR. OVERFLOW
000455          000004  57 01 01 14 2 000303            TNN,1   A1,0303,*A0        .CHAR. UNDERFLOW
000456          000005  74 04 00 00 0 000443            J       TRIPOS
                    *   20-35 +CC  1
000457          000006  57 01 01 14 2 000303            TNN,1   A1,0303,*A0        . DIVIDE OVERFLOW
000458          000007  74 04 00 00 0 000443            J       TRIPOS
                    *   20-35 +CC  1
000459                                                                            . NO ERROR INTERRUPT JUMP XQT
000460          000010  74 04 00 00 0 000403            J       TRXP0
                    *   20-35 +CC  1
000461                                        T       FORM    6,30
000462          000011  01 0000000301        TRILOC  T       1,0301              . INDEX 0   ILLEGAL FUNCTION
000463          000012  01 0000000303                T       1,0303              . INDEX 1   MEMORY LOCKOUT
000464          000013  02 0000000305                T       2,0305              . INDEX 2   CHAR. UNDERFLOW
000465          000014  02 0000000306                T       2,0306              . INDEX 3   CHAR. OVERFLOW
000466          000015  02 0000000307                T       2,0307              . INDEX 4   DIVIDE OVERFLOW
000467          000016  00 00 00 00 00 77  PSIM  +0,0,0,0,0,FILL                  . SIMULATED P REGISTER ADJUSTMENT
000468          000017  06 01 14 00 0 000035  TRILM4   S,1     A0,TRSA0            . CORRECT THE A0 IMAGE
                    *   20-35 +CC  0
000469          000020  06 01 14 00 0 000036            S,1     A0,TRSA1            . CORRECT THE A1 IMAGE
                    *   20-35 +CC  0
000470          000021  777777 000136        TRV     +       -0,TRCUI            . IMPOSSIBLE ADDRESS/SWITCH VALUE
                    *   18-35 +CC  1
000471          000022  000527 000145                +       TRCPS,TRCYI         . INTERRUPT CODE SWITCH(ALSO ADDR),SETUP
                    *   00-17 +CC  1
                    *   18-35 +CC  1
000472          000023  000561 000160                +       TRCPS,TRCPI         . INTERRUPT CODE SWITCH(ALSO ADDR),SETUP
                    *   00-17 +CC  1
                    *   18-35 +CC  1
000473                                        TRVE
000474          000024  505051254031  K1      '**(N)TRC(S) CALL ERROR FROM *'
                000025  271051474005
                000026  100621210512
                000027  272724270513
                000030  272422050505
000475          000031  000000000000  KALD    =0
```

```
000476                                  K1A
000477          000032  77777-777777    TPCLC*  +FILL
000478          000033  777777777777    TPCLC$  +FILL
000479          000034  000000 777777   TPCTO   +0+FILL                 . TRACED ADDR. TO TAKE AS NSI
000480                                  TPCA0**
000481          000035  000000000000    TPSA0   +0
000482                                  TPCA1**
000483          000036  000000000000    TPSA1   +0
000484          000037  000000000000    TPSA2   +0
000485          000040  000000000000    TPSA3   +0
000486          000041  000000000000    TPSR3   +0
000487          000042  000000000000    TPSB10  +0
000488          000043  000000000000    TPSB11  +0
000489          000044                  REG     REQ     16              . TEMP CELLS FOR B1-B15,A4 IF EX CHAIN
000490          000064                  TRNPAR  REQ     3               . FOR IV PARAMS PACKED IN SLII CALL SEQ.
000491          000067  000001 777777   TRNSA   +       1+FILL          . NEXT CYCLIC TABLE ADDRESS
000492          000070  777777777777    TRLFT   -0
000493          000071  000000000000    TRRPA   +0
000494          000072  000001 777777   TRTBE   +       1+FILL          . 1+END OF TRC TABLE
000495          000073  000001 777777   TRTBS   +       1+FILL          . 1+START OF TRC TABLE
000496          000074  000001 000000   TRSW1   +       1+0             . INTERRUPT SWITCH. SKIP SWITCH
000497                                                                  . 0=PREVENT AAIJ          0=NO SKIP
000498          000075  77777-777777    TRBHIU  +FILL                   . UP CODE + DESIRED BHIU BITS OF AAIJ OR
000499                                                                  . NESTED EX INSTRUCTION
000500                                          END
                000076  000002000000
                000077  000002000001
                000100  000001 777776
                000101  000001 000000
                000102  000000000000
```