SCELBAL

STRINGS SUPPLEMENT

(8008/8080)

SCELBI COMPUTER CONSULTING INC.

* * * * * NOTICE * * * * *

Before shipping new programs we like to have a few people try loading and testing the program from the actual printed copy as an extra check. The first couple of people who tested this manual were tripped up by the appearance of the assembled listings at two points in this manual: On page 58 just before the label STORP - there is NOTHING in the byte at address 52 067. That is the way it is supposed to be but make sure you skip that location when loading the program! Same situation exists on page 61 just before the label PPRINT - there is NOTHING (intentionally) in address 53 123. (Better go make a mark at those places now before you start loading the program!)

Also, there is a typographical error on page 66 near the middle of the page. Change line number 90 to read:

```
90  IF A$(J)<=A$(K)  GOTO  130
```

The "less than" sign was not type set and leaving it out of the statement will cause the sort program to end prematurely in some cases. (Also note that the sort program sorts by first character only!)

Finally, be extremely careful making the necessary patches to the original SCELBAL program (as indicated by the listings on pages 45 and 46). Making a mistake there (as one of our testers did!) can cause one a lot of grief in trying to get the STRINGS Supplement running!

SCELBAL STRINGS SUPPLEMENT (8008/8080)

Author:

Mark Arnold

IMPORTANT NOTICE

ACKNOWLEDGEMENT

Typesetting of this manual accomplished through the dedicated efforts of:

Miss Gabrielle Tingley

Program proofing and testing by:

Raymond Edwards

SCELBAL STRINGS SUPPLEMENT

STRING CAPABILITIES FOR SCELBAL

The STRINGS supplement to SCELBAL presented in this publication adds the following character string manipulating capabilities to the language.

1. Up to 64 strings and/or string arrays. (Up to 80 characters long.)

2. Substring capabilities as follows:
   a. The right part of a string
   b. The middle of a string
   c. The left part using b.
   d. A string array can be substringed in the same expression

3. Two additional numeric functions:
   a. LEN--returns the length of a string
   b. ASC--returns the decimal ASCII value of 1'st character of string

4. One additional string function: CHR$ (Which replaces CHR)

5. String arrays do not require dimensions.

6. Concatenation of string expressions.

7. Input and output of strings.

8. Comparison of string expressions.

STRING VARIABLES

A string variable may be any letter followed by a dollar sign. For example, A$ would be a legal string variable. A string variable may be subscripted in the normal fashion: B$(3) would yield the third element of the string array B$. The difference between numeric arrays and string arrays is that unsubscripted string variables are treated the same as one with a subscript of one, so A$ and A$(1) reference the same string. String arrays do not require or allow a dimension to be specified in a DIM statement. This feature allows the full string capability to be implemented in a system without the array option installed.

SUBSTRINGS

It is often desirable to access certain characters within a string by specifying the starting and stopping positions in that string. This capability is known as accessing a substring. To access J characters starting with the N'th character in a string A$, the format would be: A$(:N;J), where N and J could be any expression. For example, if A$ contained "ABCDE," then A$(:2;3) would yield the result "BCD," and A$(:1;4) would yeild "ABCD." A string array could also be substringed: B$(4:2;3) would yield the second through fourth characters of the fourth element of B$. If the semicolon and expression following it are omitted, the result will be all characters to the right of and including the N'th character. For example: A$(:3) will result in "CDE." Subscripted strings are handled in a similar fashion: B$(5:3) would result in all characters to the right of the second character of the fifth element of B$.

CHR$ FUNCTION

The CHR$ function is used to generate a single character string by converting the decimal value of its argument to ASCII. For example, CHR$(193) would result in the string "A." This string function replaces the old CHR function.

STRING LITERALS

The string literal is just like the old text in a PRINT statement: either single or double quotes enclosing the characters that form the string. For example, "THIS STRING" or 'ABCD $ 44.'

STRING CONCATENATIONS

Strings can be concatenated using the + operator. Concatenation is the joining together of two or more strings. For example, "AB" + 'CD' forms "ABCD," and A$+B$(8:4)+'Q' forms a string of A$ joined with the fourth character through the end of the eighth element of B$ and the literal 'Q.'

STRING EXPRESSION

A string expression is any string variable, string array, string literal, use of CHR$ function, or any concatenation of these. For example: A$+'THIS' or CHR$(N)+'T'+ W$(D+E:6;J). String expressions are legal in PRINT statements (where they replace the old text strings) and on the right of an = in a LET.

STRING LET

The string LET statement is similar to the regular LET, and may take two forms:

string = string expression
or
string array = string array expression

For example: A$='EXAMPLE' or C$(N)= A$+D$(:3) or 30 LET A$=A$+C$.

STRING OUTPUT

A string may be output in a PRINT statement subject to the normal rules for spacing and tabbing along with numeric data. For example: PRINT 'AB'+'CD' would print ABCD, or PRINT A$;2*2,B$ would print A$, then immediately print 4, then tab to the next column and print B$.

STRING INPUT

Strings or string arrays can be input using the INPUT statement in the normal fashion. For example: INPUTA$,B$(3),N would print a ? and ask for the string value of A$, then when the CR was entered, would print another ? and ask for B$(3), and then would finally input N in the normal fashion. Note that this feature replaces the old automatic conversion of ASCII input using the $.

STRING COMPARISON

String expressions can be compared using the normal comparison operations, such as =, >, <, <=, >=, or <>. If the condition is satisfied, a value of 1.0 is returned as a numeric result, and 0 is returned otherwise. The comparison goes character by character until unequal characters are found, or until all of the characters in the shortest string have been tested. In the former case, the test comparison is made between the two unequal characters, and in the latter, the length is used as the deciding factor.

ADDITIONAL FUNCTIONS

Three new functions add additional power to the language:

LEN(A$): This function returns the length of a string or string array as a decimal number. For example, if A$ has the value as in the previous example, LEN(A$) returns 5.

ASC(A$): This function returns the decimal value of the first character of the string or string array specified in ASCII. For example, ASC(A$) would return 193, because A$(:1;1) has a value of "A."

VAL(A$): This function converts the characters in the string from an ASCII representation of a decimal number to its numeric value. For example, VAL('2') returns 2.

These functions should be used only at the beginning of an expression. The arguments of these functions should be either a plain string, such as A$, or a string array subscripted by a regular variable, i.e., B$(J). So LEN(A$) and ASC(C$(N6)) would be legal, but LEN(C$(6)) and ASC(A$(:2)) would not be legal. (The

reason for this restriction is that on an 8008 system using a function like LEN(A$(6)) pushes the PC stack down more than 8 levels. An 8080 system would not have this problem.)

TRANSLATION FROM OTHER BASIC'S

Programs written for other BASIC's can probably be translated to SCELBAL with strings as follows:

RIGHT$(A$,N) becomes A$(:N)

LEFT$(A$,N) becomes A$(:1;N)

MID$(A$,N,J) becomes A$(:N;J)

The reason this format was chosen over the normal "function format" is that the SCELBAL notation is more concise and requires less memory to implement.

IMPLEMENTING STRING HANDLING CAPABILITIES

In order to implement the string extension to SCELBAL, the EVAL routine has to be modified to recognize a string expression and evaluate it. A flag, SMODE, tells if the result of an expression is string or numeric. Also, the STOSYM and ARRAY routines have to be modified in order to store string values properly. Additionally, the SCR command must initialize the strings properly, and the PRINT and INPUT statements have to be changed slightly to accommodate strings.

If EVAL encounters a ' or " or $, it assumes the expression is a string expression, so it starts over at a routine called SEVAL. The SEVAL routine must save some of the information used in the EVAL routine (such as the starting and stopping positions of the evaluation) since the SEVAL routine is a recursive routine, i.e., it calls itself. The reason it must do this is because a string expression may contain a numeric expression that must be evaluated, as in A$(N+3*J:N;J). This section of SEVAL also initializes a special STRing ACCumulator where strings are built up.

The SEVAL routine concatenates the text literals as they appear, and sets a no-concatenate flag to inform other routines that nothing else has to be done at that point.

Strings and string arrays require four bytes of information before they can be concatenated onto the STRACC by the CONCAT routine. First is the name of the string, a single letter. Next is the subscript (remember that if no subscript is present, 1 is assumed). Then the pointer to the starting character in the substring, known as the SUBSTR PNTR. This is originally initialized to 1 so if no SUBSTR PNTR is present in a string, the first character is assumed.

The last byte needed is the substring length (SUBSTR LEN) which tells how long the substring should be. If the entire substring is desired, as in the case of A$(:N) or just A$, SUBSTR LEN is set to -1. This information is found when SEVAL encounters the $ and the (.

When the + sign is encountered in a string expression, the no-concatenate flag is tested to find out if anything needs to be done. If a string literal or CHR$ function has not already concatenated something onto the STRACC, the CONCAT routine is called to concatenate the appropriate string as indicated by the name, subscript, substring pointer, and substring length. It starts by looking up the string in a table of string names and subscripts using a routine labeled SLOOK. The SLOOK routine finds the String Variable (SVAR) name and subscript combination and returns a corresponding pointer indicating where the string is stored to a location known as SPNTR. The CONCAT routine then uses the information in the SUBSTR PNTR and SUBSTR LEN to concatenate the appropriate substring onto the string accumulator. The CONCAT routine is also called at the end of the SEVAL routine to concatenate the last string onto the STRACC.

If a comparison operator, such as =, is found during string evaluation, a token value for that operator is saved, and the CONCAT routine is called, if necessary, to concatenate the last string. Next, the STRACC is saved in a special STRING COMPARISON buffer so that it may be tested against the contents of the string accumulator when SEVAL is finished.

finished. The validity of the condition indicated by the string token will be shown as a numeric result of 1.0 or 0.

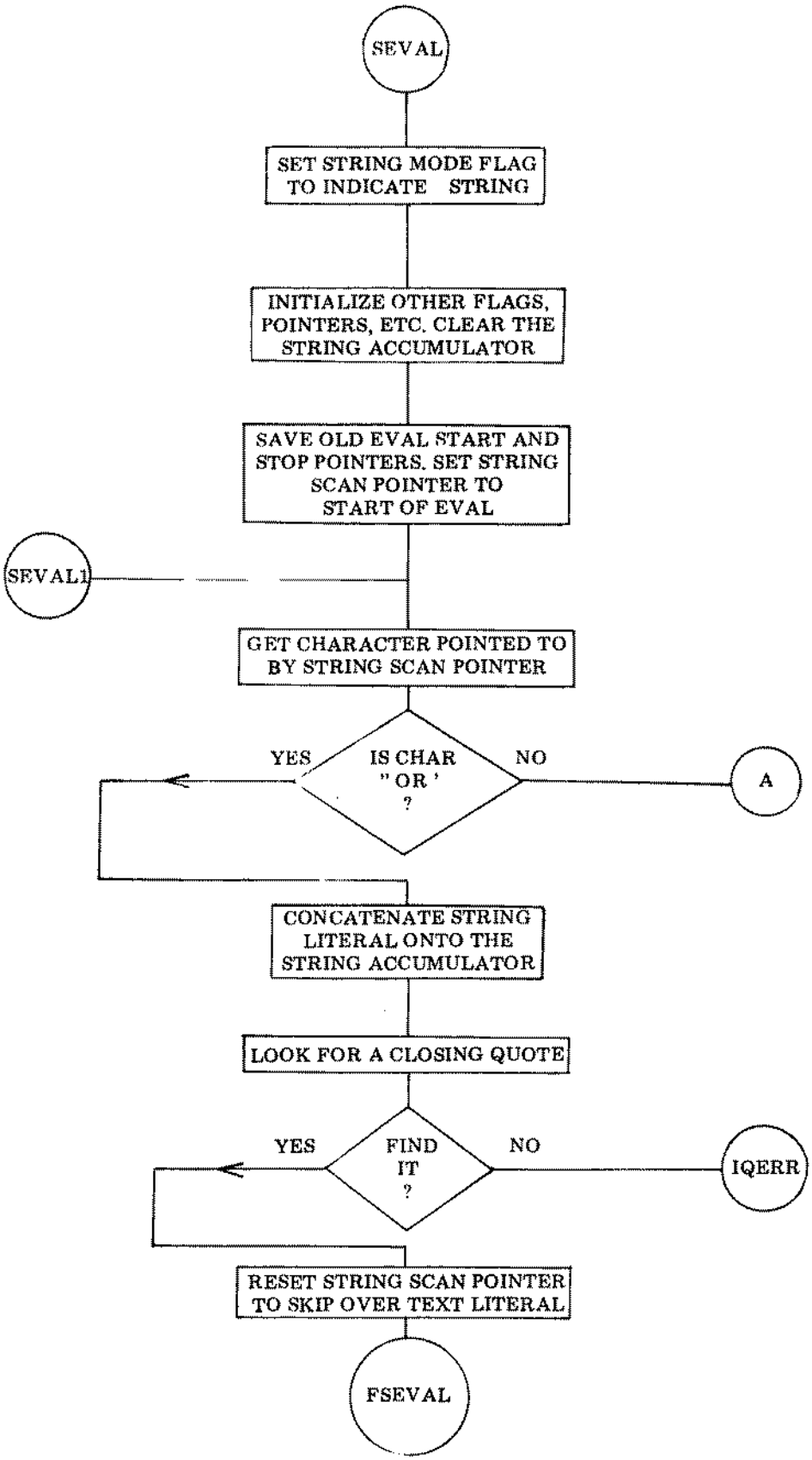
The STOSYM routine must be modified to realize when it should store the STRACC in the string storage area. If the SMODE flag is set, a patch directs control to a routine labeled SSTORE which checks to see if the array flag is set. If it is, the name and subscript of the string variable have already been calculated and placed in a special temporary string variable area (TSVAR) by a patch to the ARRAY routine. If the array flag is not set, the SSTORE routine puts the name and a subscript of 1 in TSVAR. The name and subscript are then transferred to SVAR and looked up by the SLOOK routine. If the string is new, the store operation is relatively trivial since all that has to be done is to move the STRACC to the end of the string storage area (indicated by the ENDSTR pointer), modify the ENDSTR pointer to the new end of string storage area and the pointer in the STRTAB to point to that location. If, however, the string already existed, all strings below the old value of the string must be moved over that string to save storage. Then the pointers in the STRTAB must be modified so that the pointers correspond to the revised addresses where the strings are now located. Finally, the routine can continue as if it had processed a new string.

There is another modification that must be made to the EVAL routine: the three functions that return a numeric result, but have string arguments (LEN, ASC, VAL) must be recognized. When these functions are used, the string value must be evaluated by the SEVAL routine, which may in turn call the EVAL routine. Again the recursive problem occurs, but is more serious since it may cause the PC stack of an 8008 to be pushed down more than 7 levels. The programmer must guard against this possibility by limiting the argument portion of a string function when using an 8008 system as discussed elsewhere. When the string result is returned to the SFAPCH routine, the appropriate action is taken according to the string function token.

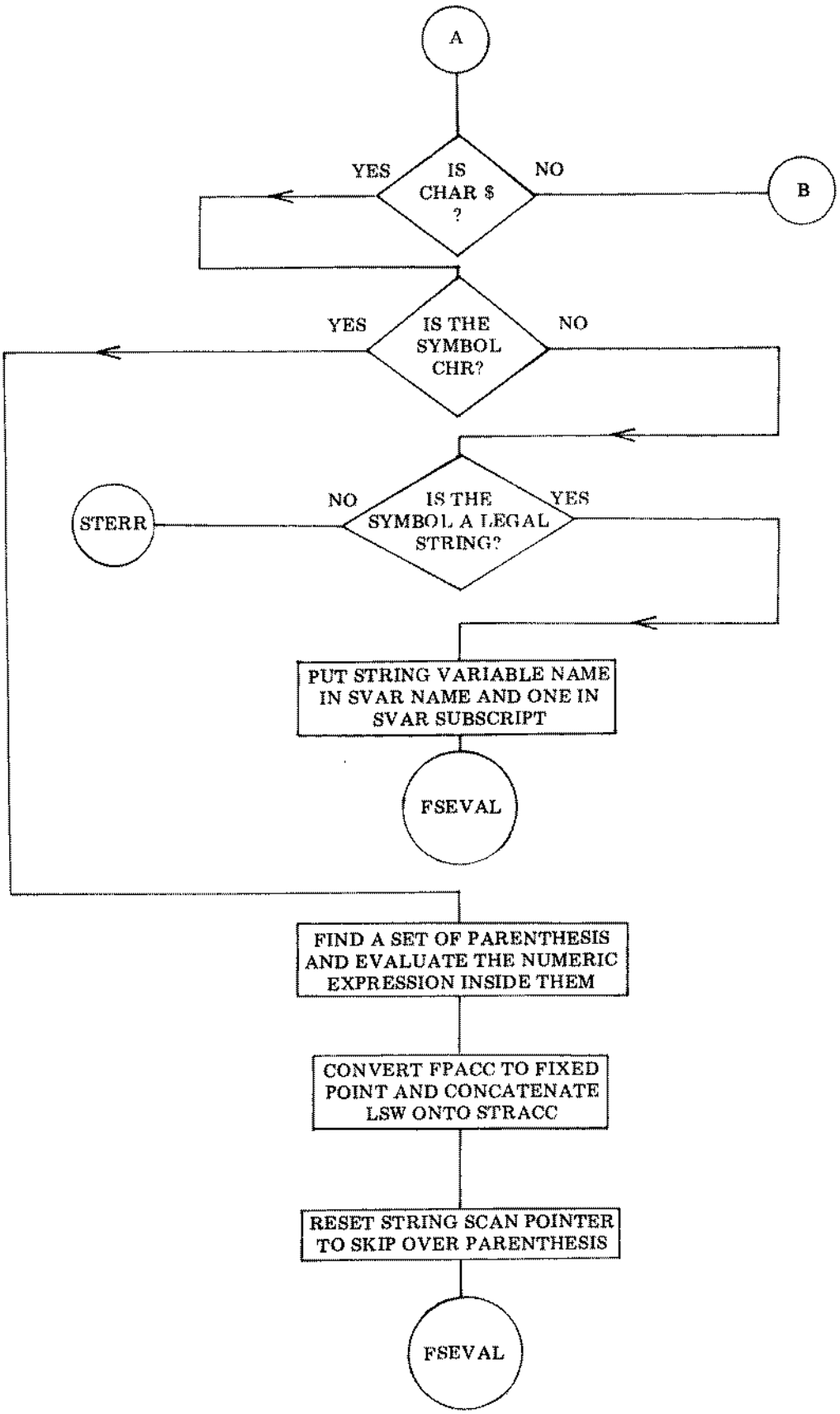
The modifications to the PRINT and INPUT statements are fairly simple. The patches to the PRINT statement allow the routine to ignore semicolons that occur inside parentheses, such as A$(:2;3), and ignore all characters inside quotes, rather than print them out. The main purpose of the INPUT patch is to recognize when a string should be input into the STRACC instead of a number into the FPACC.

Another patch worth mentioning is the modification that causes part of the EVAL routine to function as an independent subroutine to look for the comparison operators. This routine, labeled LTEQGT, is called by both the old EVAL routine, and the new SEVAL routine.

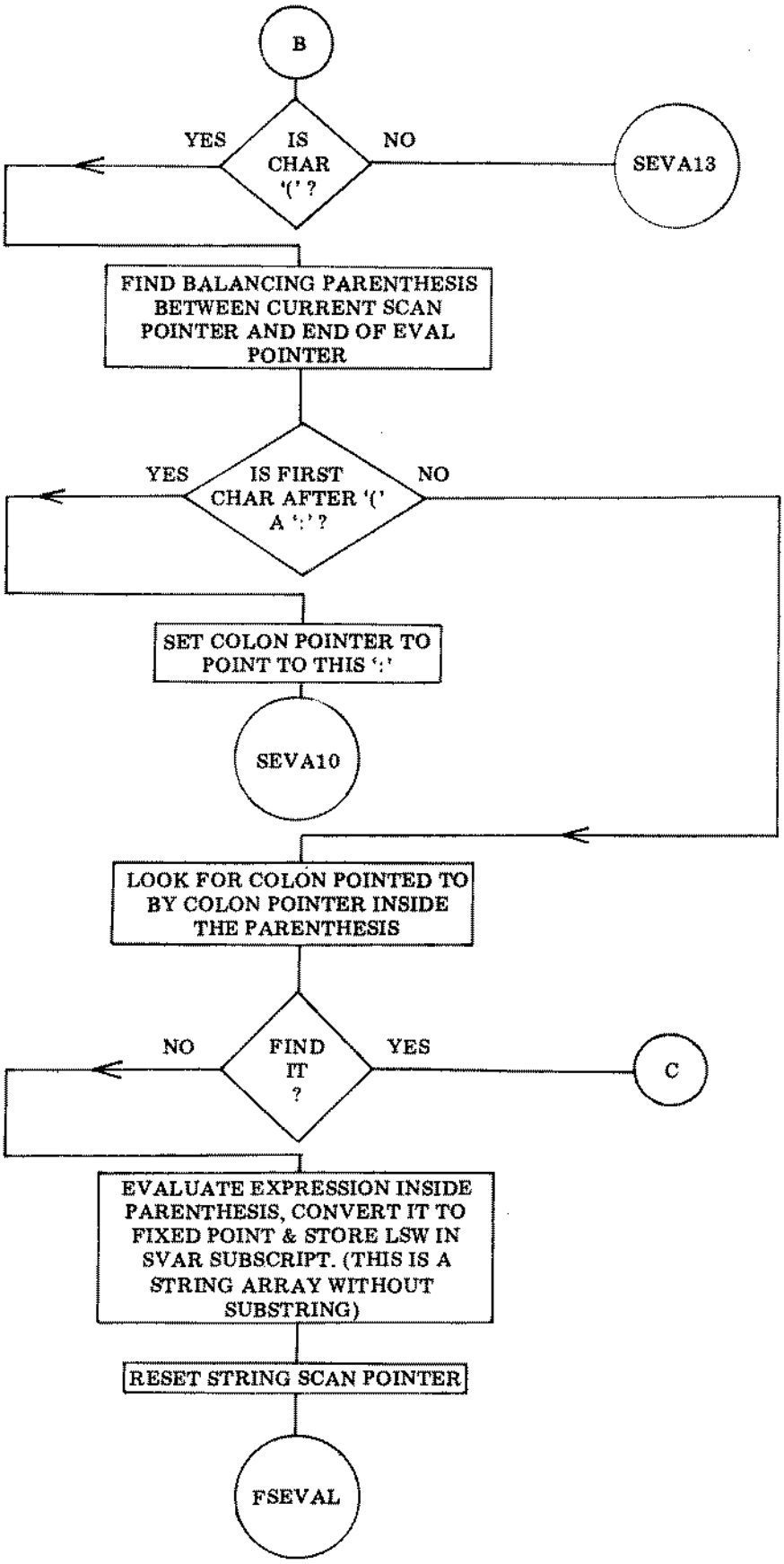
The reader can glean more information about the exact nature of the modifications by studying the accompanying flowcharts and source listings.


SEVAL
SET STRING MODE FLAG TO INDICATE STRING
INITIALIZE OTHER FLAGS, POINTERS, ETC. CLEAR THE STRING ACCUMULATOR
SAVE OLD EVAL START AND STOP POINTERS. SET STRING SCAN POINTER TO START OF EVAL
SEVAL1
GET CHARACTER POINTED TO BY STRING SCAN POINTER
YES
IS CHAR " OR ' ?
NO
A
CONCATENATE STRING LITERAL ONTO THE STRING ACCUMULATOR
LOOK FOR A CLOSING QUOTE
YES
FIND IT ?
NO
IQERR
RESET STRING SCAN POINTER TO SKIP OVER TEXT LITERAL
FSEVAL

```
SEVAL,  LLI 017         Load L with address of NOCONCAT flag
        LHI 045         $$ Load H with string page
        LMI 000         Reset NOCANCAT flag
        INL             Register L points to STRACC (CC)
        LMI 000         Clear STRACC
        LLI 002         Load L with address of SUBSTR pointer
        LMI 001         Put 1 in SUBSTR pointer
        INL             Point to SUBSTR length
        LMI 377         Put -1 in SUBSTR length (whole string)
        INL             Point to string FUNCTION TOKEN
        LMI 000         Initialize to zero
        CAL CLESYM      Clear the symbol table
        LLI 375         Load L with address of SMODE
        LMI 001         Set SMODE to string
        LLI 276         Load L with address of EVAL start
        LBM             Load start and stop into B and C
        INL
        LCM
        LLI 376         Load L with address of TEMP SEVAL start
        LMB             And stop pointers and put old eval pointers
        INL             There from B and C
        LMC
        LLI 371         Load L with address of SEVAL SCAN pointer
        LMB             Put start of EVAL pointer in SEVAL SCAN pointer

SEVAL1, LLI 371         Load L with address of SEVAL SCAN pointer
        LHI 026         ** Load H with pointer page
        CAL GETCHR      Get character pointed to be SEVAL pointer
        JTZ FSEVAL      Ignore space
        CPI 247         Is character a single quote (') ?
        JTZ SEVAL2      If so, have text literal
        CPI 242         Is character a double quote (") ?
        JFZ SEVAL5      If not, test for other characters

SEVAL2, LLI 367         Load L with address of quote type
        LMA             Store opening quote there
        LLI 371         Load L with address of SEVAL pointer
        LBM             Add 1 to SEVAL pointer
        INB
        INL             Load L with address of text pointer
        LMB             Store SEVAL POINTER +1 there

SEVAL3, LLI 372         Load L with address of LITERAL pointer
        CAL GETCHR      Get character pointed to by LITERAL pointer
        LLI 367         Load L with address of quote type
        CPM             Is character the closing quote?
        JTZ SEVAL4      If so, finish up
        LLI 020         Load L with address of STRACC
        LHI 045         $$ Load H with string page
        CAL CONCT1      Concatenate character onto STRACC
        LLI 372         Load L with address of LITERAL pointer
```

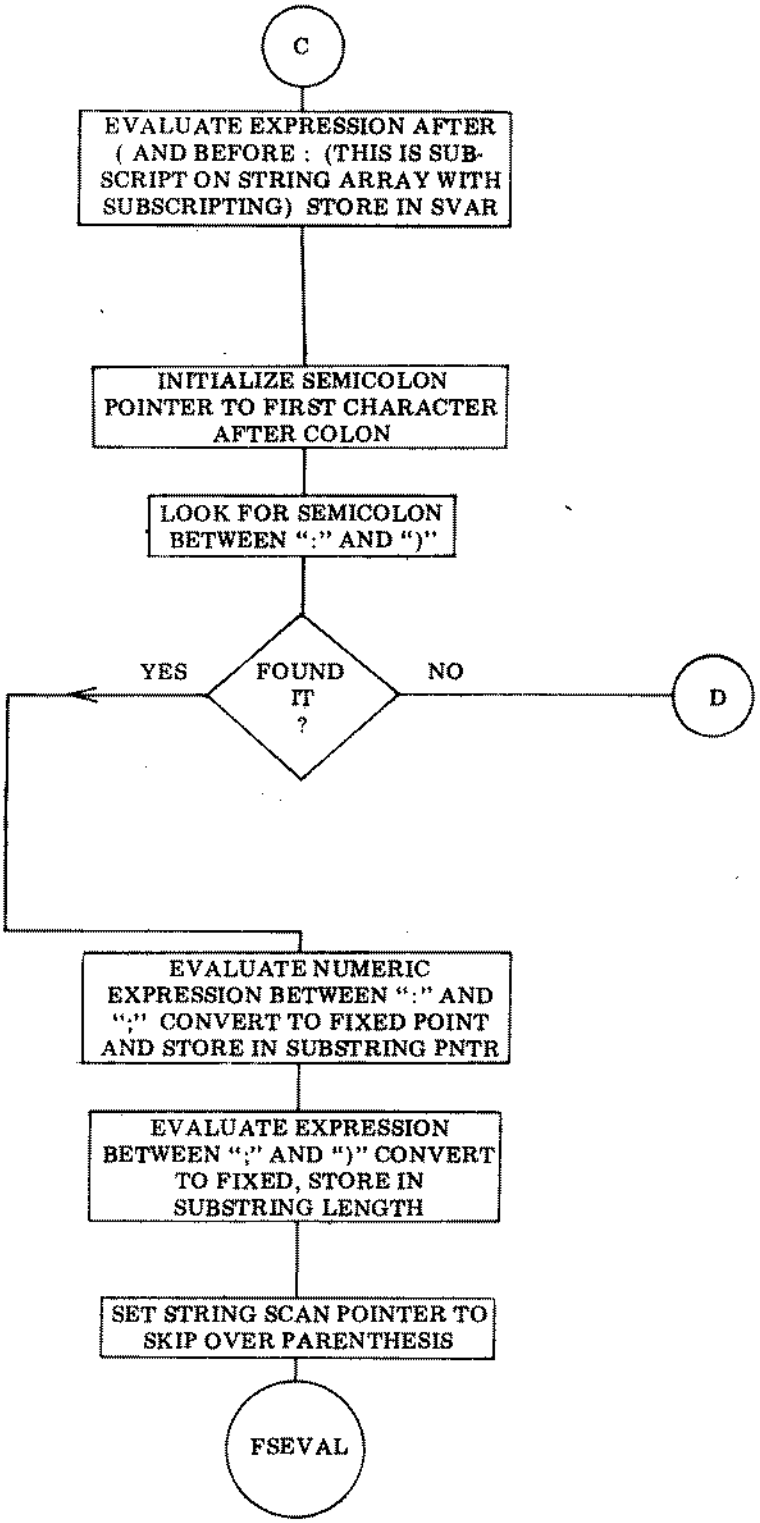

A
YES
IS CHAR $?
NO
B
YES
IS THE SYMBOL CHR?
NO
NO
IS THE SYMBOL A LEGAL STRING?
YES
STERR
PUT STRING VARIABLE NAME IN SVAR NAME AND ONE IN SVAR SUBSCRIPT
FSEVAL
FIND A SET OF PARENTHESIS AND EVALUATE THE NUMERIC EXPRESSION INSIDE THEM
CONVERT FPACC TO FIXED POINT AND CONCATENATE LSW ONTO STRACC
RESET STRING SCAN POINTER TO SKIP OVER PARENTHESIS
FSEVAL

```
        LHI 026         ** Load H with pointer page
        CAL SELOOP      Loop until end of SEVAL
        JFZ SEVAL3      Done yet?
        JMP IQERR       If no closing quote, error

SEVAL4, LLI 372         Load L with address of LITERAL pointer
        LBM             Load LITERAL pointer into B
        LLI 371         Put LITERAL pointer in SEVAL pointer
        LMB
        LLI 017         Load L with address of NOCANCAT flag
        LHI 045         $$ Load H with STRING page
        LMI 001         Set NOCANCAT flag to prevent concat
        JMP FSEVAL      Continue with string eval

SEVAL5, CPI 244         Is character a $ ?
        JFZ SEVAL6      If not, keep testing
        LEI 374         Load E with address of CHR string
        LDI 045         $$ Load H with STRING page
        LLI 120         Load L with address of SYMBOL
        CAL STRCP       Test if symbol = CHR
        JTZ CHR         If it does, have CHR function
        LLI 120         Load L with address of SYMBOL
        LHI 026         ** Load H with page of SYMBOL
        LAM             Get SYMBOL (CC)
        CPI 001         Is (CC) = 1?
        JFZ STERR       If not, can't be legal string
        INL             L points to first character of symbol
        LAM             Get string name
        LLI 000         Load L with address of SVAR name
        LHI 045         $$ Load H with STRING page
        LMA             Put string name in SVAR name
        INL             Point to SVAR SUBSCRIPT
        LMI 001         Put 1 in SVAR SUBSCRIPT
        CAL CLESYM      Clear the symbol
        JMP FSEVAL      Continue with string eval

   CHR, LLI 371         Load L with address of SEVAL pointer
        LBM             Add 1 to SEVAL pointer
        INB
        LMB
        LLB             Load L with SEVAL pointer
        LAM             Get character pointed to by SEVAL pointer
        CPI 250         Is character a parenthesis "(" ?
        JFZ CHR         If not, keep looking
        LLI 276         Load L with address of start of EVAL pointer
        INB             Add one to skip over "("
        LMB             Start numeric evaluation just beyond "("
        LDB             Put pointer to beyond "(" in D
        LLI 377         Load L with address of end of SEVAL pointer
        LEM             Load E with end of SEVAL pointer
        CAL PARNB       Find balancing parenthesis between D and E
```

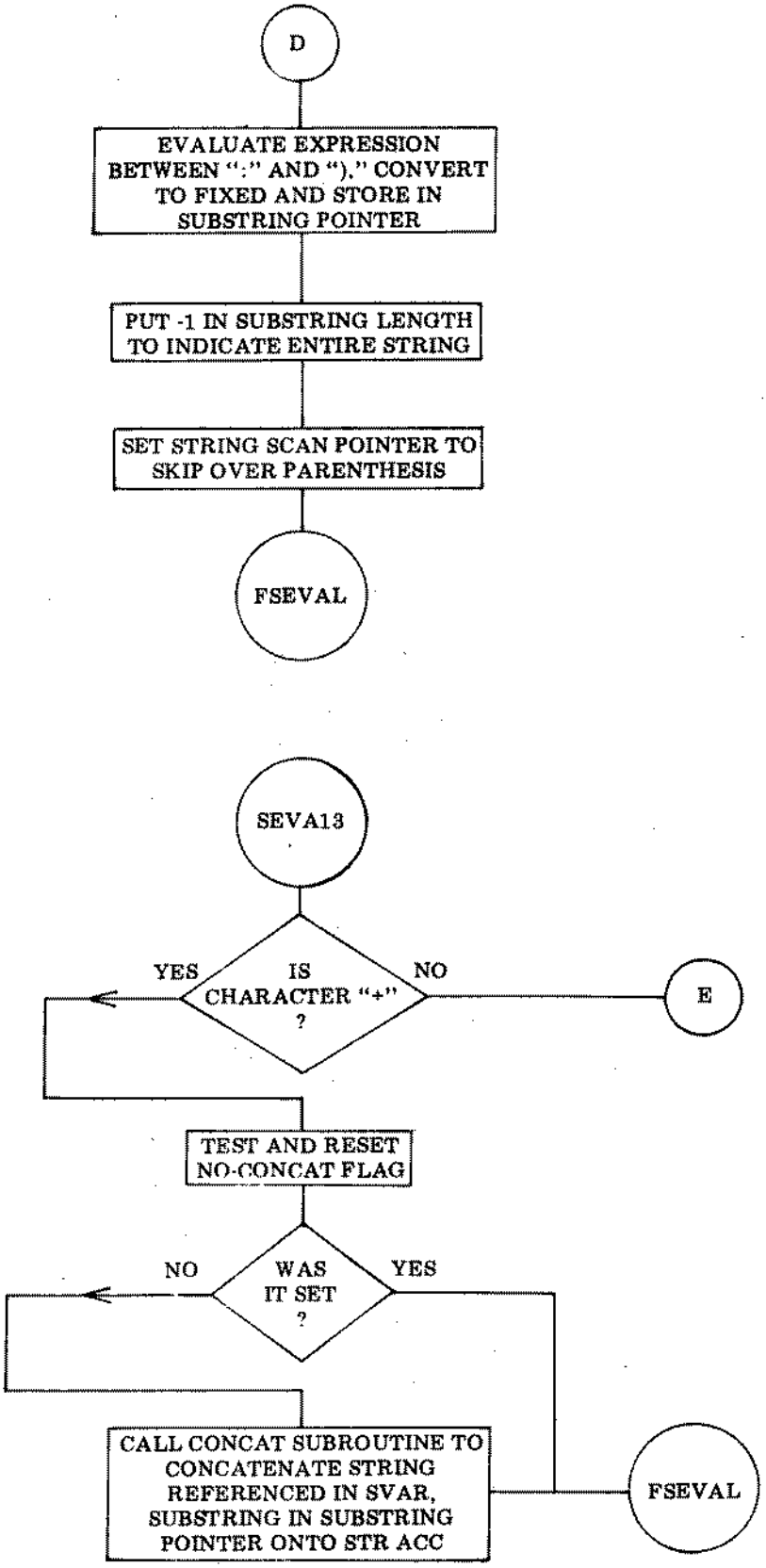

B
IS CHAR '(' ?
YES
NO
SEVA13
FIND BALANCING PARENTHESIS BETWEEN CURRENT SCAN POINTER AND END OF EVAL POINTER
IS FIRST CHAR AFTER '(' A ':' ?
YES
NO
SET COLON POINTER TO POINT TO THIS ':'
SEVA10
LOOK FOR COLON POINTED TO BY COLON POINTER INSIDE THE PARENTHESIS
FIND IT ?
NO
YES
C
EVALUATE EXPRESSION INSIDE PARENTHESIS, CONVERT IT TO FIXED POINT & STORE LSW IN SVAR SUBSCRIPT. (THIS IS A STRING ARRAY WITHOUT SUBSTRING)
RESET STRING SCAN POINTER
FSEVAL

```
          LLI 277       Load L with address of end of EVAL pointer
          DCB           Subtract 1 from pointer to balance parenthesis
          LMB           Put pointer just before to balance parenthesis end pntr
          CAL EVALS     Evaluate numeric expression between parenthesis
          CAL FPFIX     Convert the FPACC to fixed point
          LLI 124       Load L with address of FPACC LSW
          LAM           Get fixed byte
          LLI 020       Load L with address of STRACC
          LHI 045       $$ Load H with STRING PAGE
          CAL CONCT1    Concat new fixed byte to STRACC
          LLI 017       Load L with address of NOCANCAT flag
          LMI 001       Set NOCANCAT flag so SEVAL does not concat
          CAL CLESYM    Clear the symbol
          LLI 277       Load L with address of EVAL end pointer
          LBM           Add one to it
          INB
          LLI 371       Load L with address of SEVAL pointer
          LMB           Store 1 + end of EVAL pointer there
          JMP FSEVAL    Continue with string evaluation

SEVAL6,   CPI 250       Is character a "(" ?
          JFZ SEVA13    If not, keep looking
          LLI 371       Load L with address of SEVAL pointer
          LDM           Load D with pointer to "("
          IND           Add 1 to skip over "("
          LLI 377       Load L with address of end of SEVAL pointer
          LEM           Load E with end of SEVAL pointer
          CAL PARNB     Balance parenthesis
          LLI 372       Load L with address of end of PARENTHESIS pointer
          LMB           Store pointer to balancing parenthesis there
          DCL           Point to SEVAL pointer
          LBM           Add 1 to SEVAL pointer
          INB
          LLB           Point to SEVAL pointer +1 after "("
          LAM           Get character just after "("
          CPI 272       Test if it is a colon ":"
          JFZ SEVA16    If not colon, keep looking
          LLI 373       Load L with address of COLON pointer
          LMB           Store pointer to ":" in COLON pointer
          JMP SEVA10    Continue to test for substring

SEVA16,   LLI 373       Load L with address of COLON pointer
          LMB           Store pointer to just after "(" in COLON pointer

SEVAL7,   LLI 373       Load L with address of COLON pointer
          CAL GETCHR    Get character pointed to by COLON pointer
          CPI 272       Is character a colon ":" ?
          JFZ SEVAL8    If not, keep looking
          LLI 371       Load L with address of SEVAL pointer
          LBM           Add 1 to SEVAL pointer
          INB
```

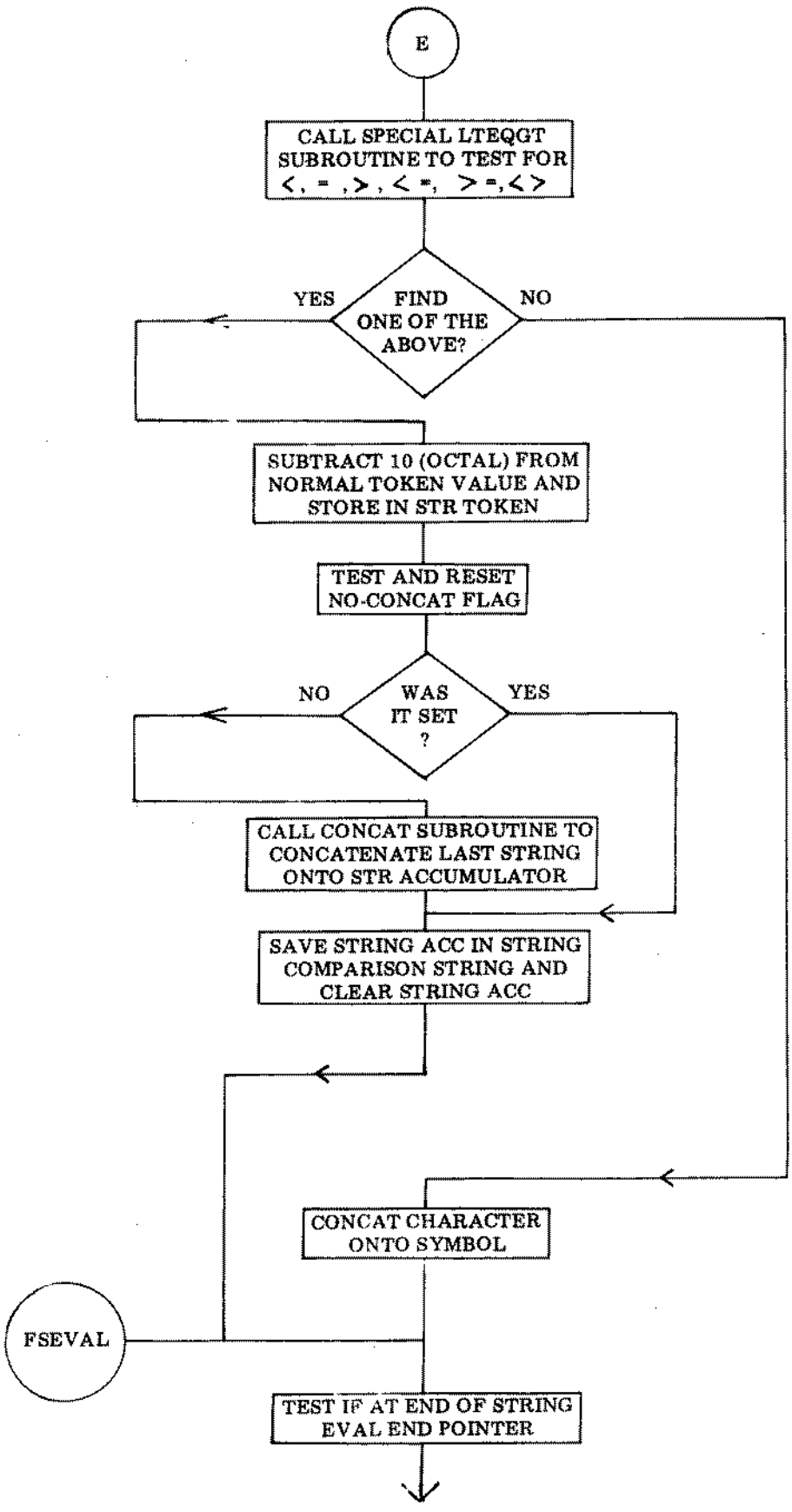
```
        LLI 276     Load L with address of start of EVAL pointer
        LMB         Store SEVAL pointer in start of EVAL pointer
        LLI 373     Load L with address of COLON pointer
        LBM         Subtract 1 from COLON pointer
        DCB
        LLI 277     Load L with address of end of EVAL pointer
        LMB         Store pointer to just before ":" in end EVAL
        CAL EVALS   Evaluate expression between "(" and ":"
        CAL FPFIX   Fix floating point value of expression
        LLI 124     Load L with address of LSW of FPACC
        LAM         Get fixed byte
        LLI 001     Load L with address of SVAR SUBSCRIPT
        LHI 045     $$ Load H with STRING page
        LMA         Store subscript in SVAR
        JMP SEVA10  Continue subscripted substringing

SEVAL8, LLI 373     Load L with address of COLON pointer
        CAL S2LOOP  Loop until before "("
        JFZ SEVAL7  Continue if not done

SEVAL9, LLI 371     Load L with address of SEVAL pointer
        LBM         Add 1 to SEVAL pointer
        INB
        INL         Pointer to balance PARENTHESIS pointer
        LCM         Subtract 1 from PARENTHESIS pointer
        DCC
        LLI 276     Load L with address of start of EVAL pointer
        LMB         Start EVAL after "("
        INL         Finish EVAL pointer
        LMC         Finish EVAL before ")"
        CAL EVALS   Evaluate subscript between "(" and ")"
        CAL FPFIX   Convert subscript to fixed byte
        LLI 124     Load L with address of fixed byte
        LAM         Get it
        LLI 001     Load L with address of SVAR SUBSCRIPT
        LHI 045     $$ Load H with string page
        LMA         Put subscript there
        LLI 372     Load L with address of BAL PARN pntr
        LHI 026     ** Load H with pointer page
        LBM         Get pointer to BAL PARN
        LLI 371     Load L with address of SEVAL pointer
        LMB         But BAL PARN pointer there to skip over subscript
        JMP FSEVAL  Continue with string evaluation

SEVA10, LLI 373     Load L with address of COLON pointer
        LHI 026     ** Load H with pointer page
        LBM         Add 1 to COLON pointer
        INB
        INL         SEMICOLON pointer
        LMB         Start looking for ";" after ":"
```


C
EVALUATE EXPRESSION AFTER (AND BEFORE : (THIS IS SUB-SCRIPT ON STRING ARRAY WITH SUBSCRIPTING) STORE IN SVAR
INITIALIZE SEMICOLON POINTER TO FIRST CHARACTER AFTER COLON
LOOK FOR SEMICOLON BETWEEN ":" AND ")"
YES
FOUND IT ?
NO
D
EVALUATE NUMERIC EXPRESSION BETWEEN ":" AND ";" CONVERT TO FIXED POINT AND STORE IN SUBSTRING PNTR
EVALUATE EXPRESSION BETWEEN ";" AND ")" CONVERT TO FIXED, STORE IN SUBSTRING LENGTH
SET STRING SCAN POINTER TO SKIP OVER PARENTHESIS
FSEVAL

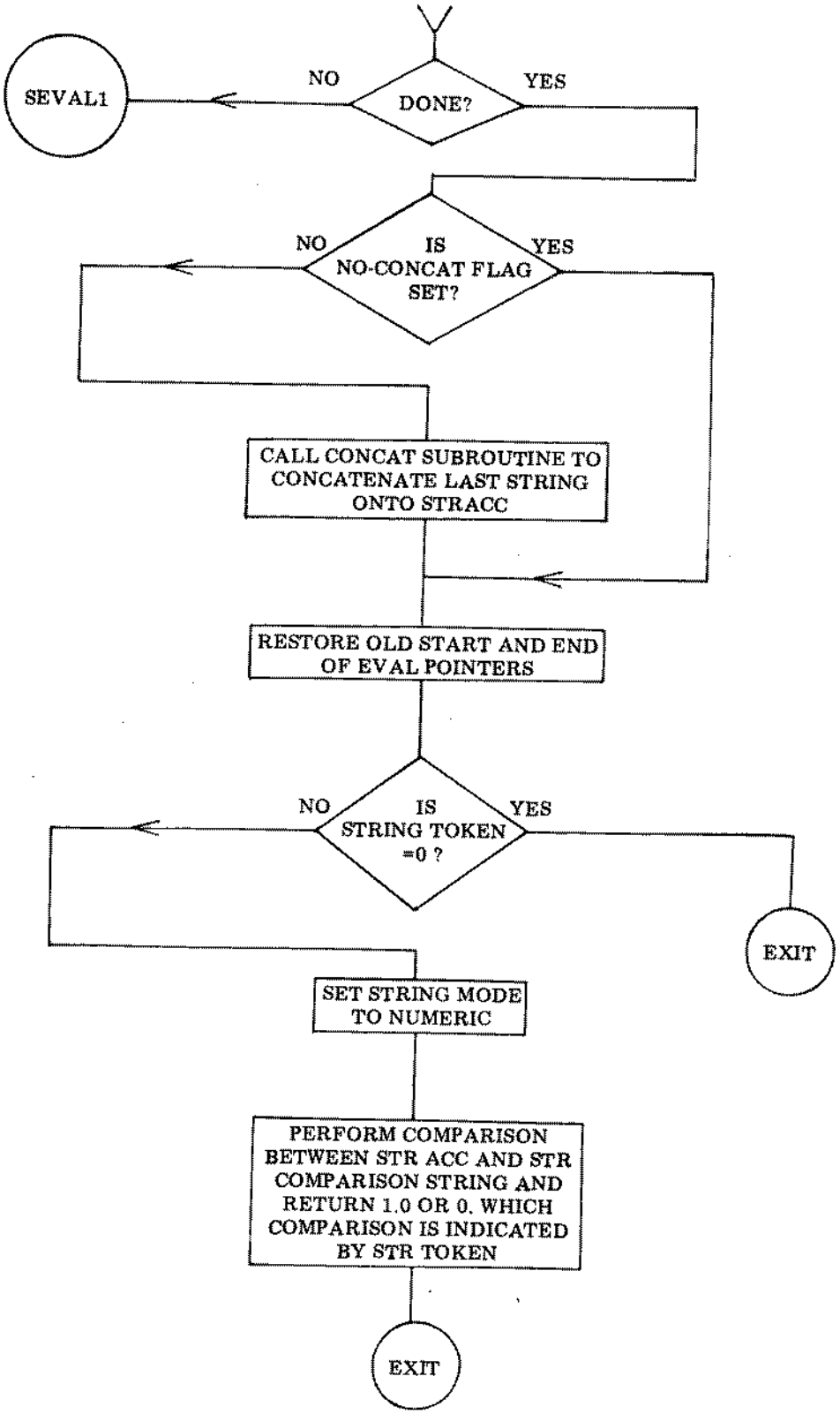
```
SEVA11,  LLI 374      Load L with address of SEMICOLON pointer
         CAL GETCHR   Get character pointed to by SEMICOLON pointer
         CPI 273      Is character a semicolon ";" ?
         JFZ SEVA12   If not, keep looking
         LLI 373      Load L with address of COLON pointer
         LBM          Add 1 to COLON pointer
         INB
         INL          Point to SEMICOLON pointer
         LCM          Subtract 1 from SEMICOLON pointer
         DCC
         LLI 276      Load L address at start of EVAL pointer
         LMB          Start EVAL after ":"
         INL          End of EVAL pointer
         LMC          End EVAL before ";"
         CAL EVALS    Evaluate SUBSTR pointer between ":" and ";"
         CAL FPFIX    Convert it to fixed
         LLI 124      Load L address of fixed byte
         LAM          Get it
         LLI 002      Load L with address of SUBSTR pointer
         LHI 045      $$ Load H with string page
         LMA          Put SUBSTR pointer there
         LLI 374      Load L with address of SEMICOLON pointer
         LHI 026      ** Load H with pointer page
         LBM          Add 1 to SEMICOLON pointer
         INB
         LLI 372      Load L address to BAL PARN pointer
         LCM          Subtract 1 from BAL PARN pointer
         DCC
         LLI 276      Load L with start of EVAL pointer
         LMB          Start EVAL just after ";"
         INL          Finish EVAL pointer
         LMC          Finish EVAL before ")"
         CAL EVALS    Evaluate substring length
         CAL FPFIX    Convert it to fixed byte
         LLI 124      Load L with address of FIXED byte
         LAM          Get it
         LLI 003      Load L with address of SUBSTR length
         LHI 045      $$ Load H with STRING page
         LMA          Put SUBSTR length there
         LLI 372      Load L with address of BAL PARN pointer
         LHI 026      ** Load H with pointer page
         LBM          Get BAL PARN pointer
         DCL          SEVAL pointer
         LMB          Put BAL PARN pointer there to skip over (. .)
         JMP FSEVAL   Continue to evaluate string expression

SEVA12,  LLI 374      Load L with address of SEMICOLON pointer
         CAL S2LOOP   Loop until just before ")"
         JFZ SEVA11   If not done, continue
         LLI 373      Load L with address of COLON pointer
         LBM          Add 1 to COLON pointer
```


D
EVALUATE EXPRESSION BETWEEN ":" AND ")," CONVERT TO FIXED AND STORE IN SUBSTRING POINTER
PUT -1 IN SUBSTRING LENGTH TO INDICATE ENTIRE STRING
SET STRING SCAN POINTER TO SKIP OVER PARENTHESIS
FSEVAL
SEVA13
IS CHARACTER "+" ?
YES
NO
E
TEST AND RESET NO-CONCAT FLAG
WAS IT SET ?
NO
YES
CALL CONCAT SUBROUTINE TO CONCATENATE STRING REFERENCED IN SVAR, SUBSTRING IN SUBSTRING POINTER ONTO STR ACC
FSEVAL

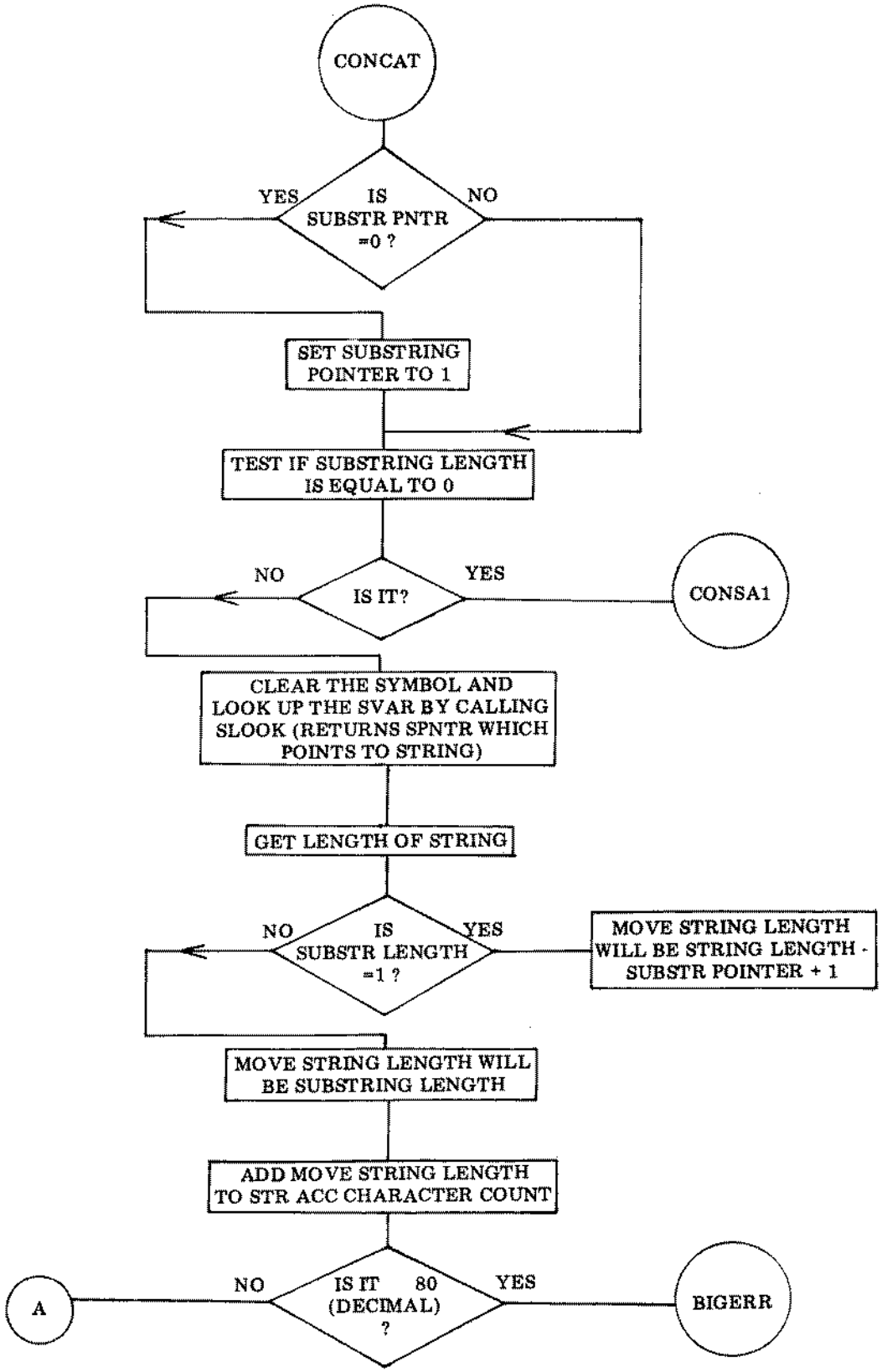
```
        INB
        DCL         BAL PARN pointer
        LCM         Subtract 1 from BAL PARN pointer
        DCC
        LLI 276     Load L with start of EVAL pointer
        LMB         Start EVAL after ":"
        INL         Finish EVAL pointer
        LMC         Finish EVAL before ")"
        CAL EVALS   Evaluate between ":" and ")"
        CAL FPFIX   Fix SUBSTR pointer to byte
        LLI 124     Load L with address of fixed byte
        LAM         Get it
        LLI 002     Load L with address of SUBSTR pointer
        LHI 045     $$ Load H with STRING page
        LMA         Put SUBSTR pointer there
        INL         SUBSTR length
        LMI 377     Put -1 in SUBSTR length for whole string
        LLI 372     Load L with address of BAL PARN pointer
        LHI 026     ** Load H with pointer page
        LBM         Get BAL PARN pointer
        DCL         SEVAL pointer
        LMB         Put BAL PARN pointer in SEVAL pointer, skip (. .)
        JMP FSEVAL  Continue with string evaluation

SEVA13, CPI 253     Is character a "+" (concatenate operation)?
        JFZ SEVA14  If not, keep looking
        LLI 017     Load L with address of NOCANCAT flag
        LHI 045     $$ Load H with STRING page
        LAM         Get NOCONCAT flag
        LMI 000     Reset NOCONCAT flag
        NDA         Is NOCONCAT flag set?
        CTZ CONCAT  If not, concatenate latest string
        JMP FSEVAL  Continue with string evaluation

SEVA14, LLI 371     Load L with address of SEVAL pointer
        LBM         Get SEVAL pointer
        LLI 200     Load L with address of EVAL pointer
        LMB         Put SEVAL pointer there
        CAL LTEQGT  Test for comparison operations
        LLI 176     Load L with address of PARSE pointer
        LBM         Get token to tell if found comparison operation
        INB         Exercise B to see if found operation
        DCB
        JTZ SEVA15  If no operation found, concatenate scan character
        LAB         Get TOKEN into A
        SUI 010     Subtract 8 from TOKEN
        LLI 004     Load L with address of STRING TOKEN
        LHI 045     $$ Load H with STRING page
        LMA         Put STRING TOKEN there
        LLI 017     Load L with address of NOCANCAT flag
        LAM         Get NOCANCAT flag
```


E
CALL SPECIAL LTEQGT SUBROUTINE TO TEST FOR <, = ,>, < =, > =,< >
YES
FIND ONE OF THE ABOVE?
NO
SUBTRACT 10 (OCTAL) FROM NORMAL TOKEN VALUE AND STORE IN STR TOKEN
TEST AND RESET NO-CONCAT FLAG
NO
WAS IT SET ?
YES
CALL CONCAT SUBROUTINE TO CONCATENATE LAST STRING ONTO STR ACCUMULATOR
SAVE STRING ACC IN STRING COMPARISON STRING AND CLEAR STRING ACC
CONCAT CHARACTER ONTO SYMBOL
FSEVAL
TEST IF AT END OF STRING EVAL END POINTER

```
        LMI 000         Reset NOCANCAT flag
        NDA             Test NOCANCAT flag
        CTZ CONCAT      If not set, concatenate last string on STRACC
        LHI 045         $$ Load H with STRING page
        LLI 020         Load L with address of STRACC
        LDH             Load H with STRING page
        LEI 140         Load L with address of string compare string
        CAL MOVEC       Save STRACC for later comparison
        LLI 020         Load L with address of STRACC
        LMI 000         Clear the STRACC
        LLI 200         Load L with address of EVAL pointer
        LHI 026         ** Load H with pointer page
        LBM             Get EVAL pointer
        LLI 371         Load L with address of SEVAL pointer
        LMB             Put EVAL pointer there
        JMP FSEVAL      Continue with string evaluation

SEVA15, CAL CONCTS      Concatenate character onto symbol

FSEVAL, LLI 371         Load L with address of SEVAL pointer
        LHI 026         ** Load H with pointer page
        CAL SELOOP      Loop until end of string expression
        JFZ SEVAL1      If not done, continue
        LLI 017         Load L with address of NOCANCAT flag
        LHI 045         $$ Load H with STRING page
        LAM             Get NOCANCAT flag
        NDA             Test NOCANCAT flag
        CTZ CONCAT      If not set, concatenate last string
        LLI 376         Load L with address of TEMP SEVAL start pointers
        LHI 026         ** Load H with pointer page
        LBM             Get start of EVAL pointer
        INL             End of SEVAL pointer
        LCM             Get end of SEVAL pointer
        LLI 276         Load L with address of start of EVAL pointer
        LMB             Restore start and end of EVAL pointers
        INL
        LMC
        LLI 004         Load L with address of STRING TOKEN
        LHI 045         $$ Load H with STRING page
        LAM             Get STRING TOKEN
        NDA             Is STRING TOKEN 0?
        RTZ             Return if it is
        LLI 375         Load L with address of SMODE
        LHI 026         ** Load H with pointer page
        LMI 000         Set SMODE back to numeric
        LLI 140         Load L with address of string comparison string
        LHI 045         $$ Load H with STRING page
        LDH             Load D with STRING page
        LEI 020         Load E with address of STRACC
        CPI 001         Is STRING TOKEN for LT?
        JFZ STC1        If not, try something else
```

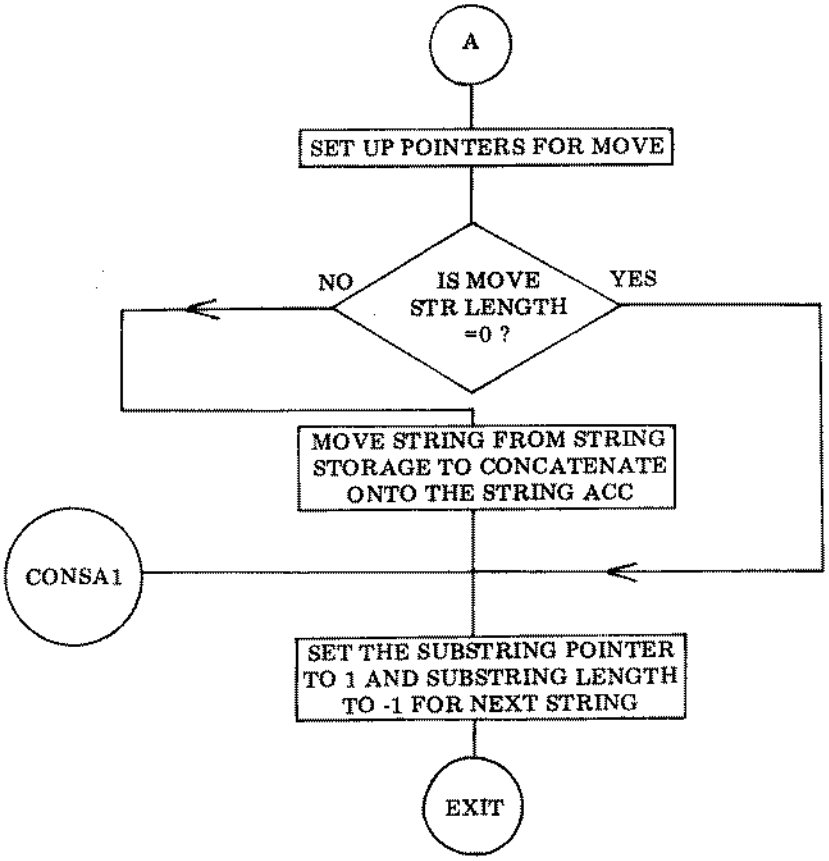

SEVAL1
NO
DONE?
YES
NO
IS
NO-CONCAT FLAG
SET?
YES
CALL CONCAT SUBROUTINE TO
CONCATENATE LAST STRING
ONTO STRACC
RESTORE OLD START AND END
OF EVAL POINTERS
NO
IS
STRING TOKEN
=0 ?
YES
EXIT
SET STRING MODE
TO NUMERIC
PERFORM COMPARISON
BETWEEN STR ACC AND STR
COMPARISON STRING AND
RETURN 1.0 OR 0. WHICH
COMPARISON IS INDICATED
BY STR TOKEN
EXIT

```
        CAL SSTRCP      Compare string comparison string with STRACC
        LHI 001         ** Load H with floating point page
        JMP LT1         Go to special less than entry point for test

STC1,   CPI 002         Other token matching routines similar to above
        JFZ STC2
        CAL SSTRCP
        LHI 001
        JMP EQ1

STC2,   CPI 003
        JFZ STC3
        CAL SSTRCP
        LHI 001
        JMP GT1

STC3,   CPI 004
        JFZ STC4
        CAL SSTRCP
        LHI 001
        JMP LE1

STC4,   CPI 005
        JFZ STC5
        CAL SSTRCP
        LHI 001
        JMP GE1

STC5,   CAL SSTRCP
        LHI 001
        JMP NE1

SELOOP, LBM             Add 1 to pointer, test against
        INB             End of SEVAL pointer
        LMB
        DCB
        LLI 377
        LAM
        CPB
        RET

S2LOOP, LBM             Add 1 to pointer, test against
        INB             BAL PARN pointer -1
        LMB
        DCB
        LLI 372
        LAM
        SUI 001
        CPB
        RET
```

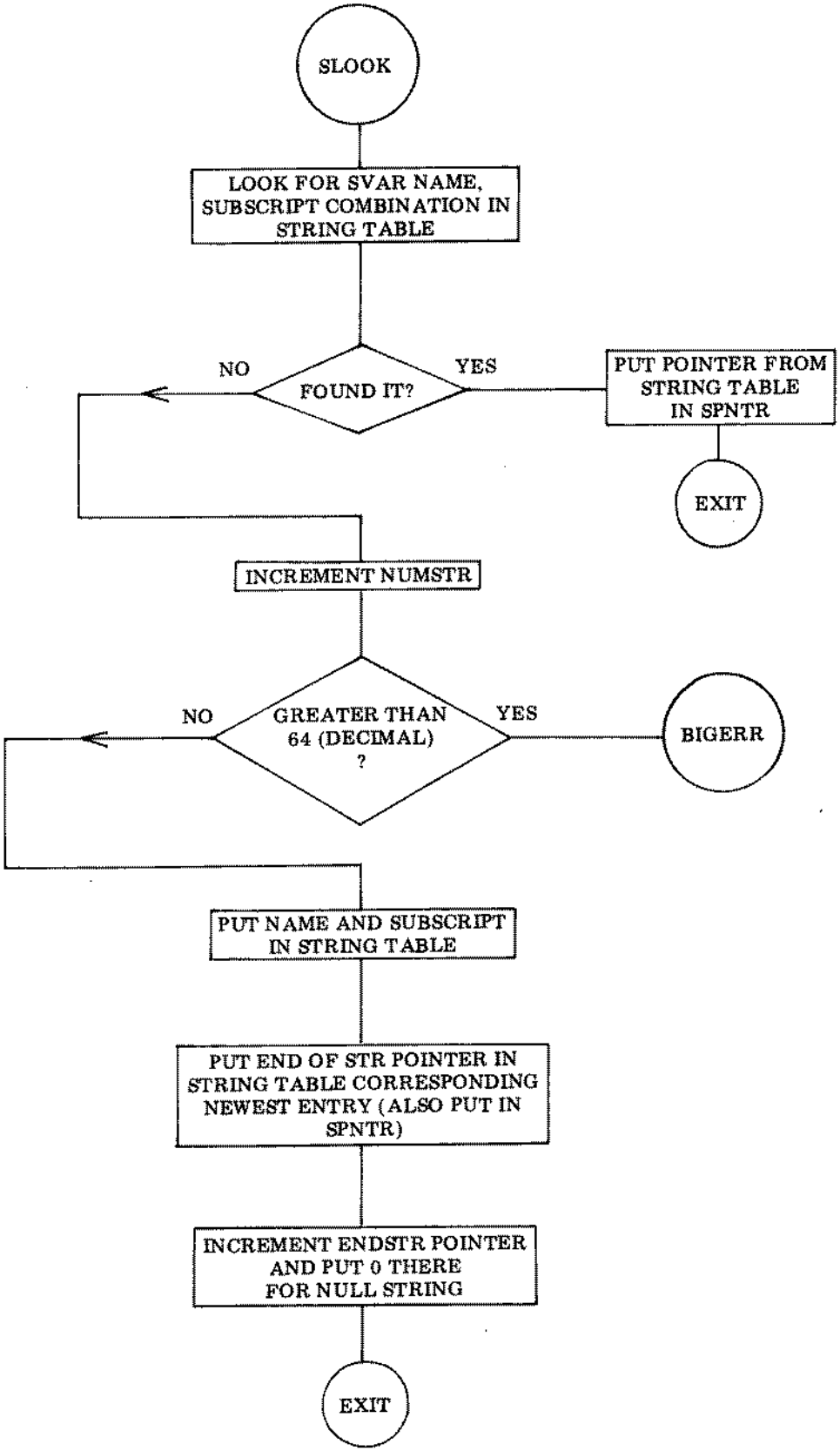

CONCAT
IS SUBSTR PNTR =0 ?
YES
NO
SET SUBSTRING POINTER TO 1
TEST IF SUBSTRING LENGTH IS EQUAL TO 0
IS IT?
NO
YES
CONSA1
CLEAR THE SYMBOL AND LOOK UP THE SVAR BY CALLING SLOOK (RETURNS SPNTR WHICH POINTS TO STRING)
GET LENGTH OF STRING
IS SUBSTR LENGTH =1 ?
NO
YES
MOVE STRING LENGTH WILL BE STRING LENGTH - SUBSTR POINTER + 1
MOVE STRING LENGTH WILL BE SUBSTRING LENGTH
ADD MOVE STRING LENGTH TO STR ACC CHARACTER COUNT
IS IT 80 (DECIMAL) ?
NO
YES
A
BIGERR

```
SSTRCP,  CAL SAVEHL   Save H, L, and D, E
         LAM          Get (CC) of first string
         CAL SWITCH   Point to other string
         LBM          Get (CC) for second string
         CAL SWITCH   Point to first string
         CPB          Compare lengths
         JTZ SSTRZ    If equal, test for (")
         JFS SSTRCL   Second is shortest
         LBA          First is shorter

SSTRCL,  CAL ADV      Next character
         LAM          Get it
         CAL SWITCH   Second string
         CAL ADV      Next character

SSTRCE,  CPM          Compare characters
         RFZ          Return if less than or greater than
         DCB          Decrement (CC)
         JFZ SSTRCL   Continue if not 0
         CAL RESTHL   Originate strings
         LAM          First (CC)
         CAL SWITCH   Second string
         CPM          Compare lengths
         RET

SSTRZ,   NDA          Is length = 0?
         RTZ          Return if it is
         JMP SSTRCL   Otherwise, as normal

CONCAT,  LHI 045      $$ Load H with STRING page
         LLI 002      Load L with address of SUBSTR pointer
         LAM          Get SUBSTR pointer
         NDA          Is SUBSTR pointer = 0?
         JFZ CONCT0   If not, continue
         LMI 001      Replace 0 with 1

CONCT0,  LLI 003      Load L with address of SUBSTR length
         LAM          Get SUBSTR length
         NDA          Is SUBSTR length = 0?
         JTZ CONSA1   If so, don't concatenate
         CAL CLESYM   Clear symbol
         CAL SLOOK    Look up SVAR in STRTAB
         LLI 005      Load L with address of SPNTR
         LHI 045      $$ Load H with STRING page
         LDM          Load SPNTR in D and E
         INL
         LEM
         LHD          Put SPNTR in H and L
         LLE          To point to (CC) of string
         LBM          Get actual length of string from (CC)
         LHI 045      $$ Load H with STRING page
```

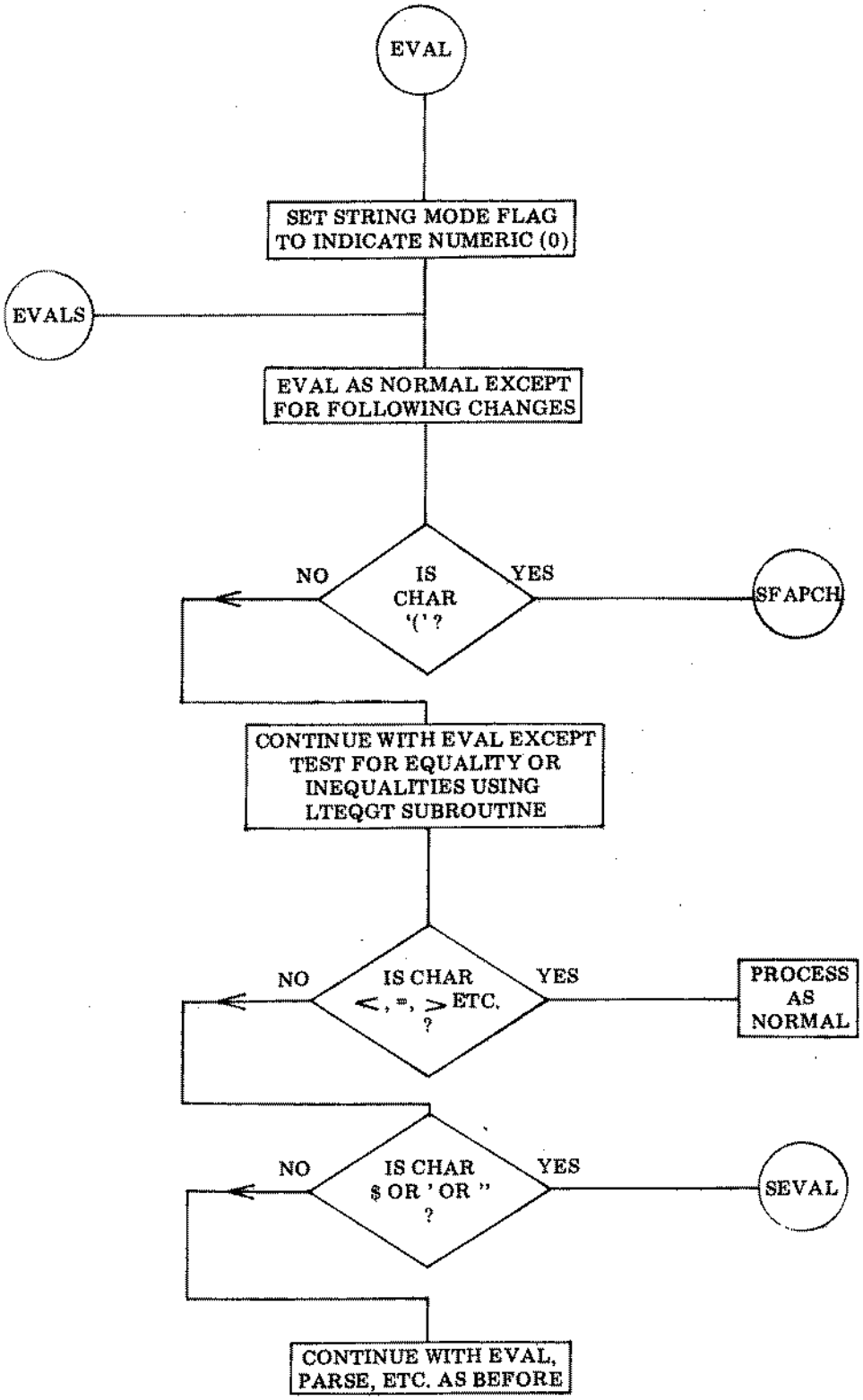
```
        LLI 003         Load L with address of SUBSTR length
        LAM             Get SUBSTR length
        CPI 377         Is SUBSTR length = -1?
        JFZ CONCA1      If not, have move length
        LLI 002         Load L with address of SUBSTR pointer
        LAB             Get string length in accumulator
        SUM             Subtract SUBSTR pointer from string length
        LBA             Put difference in B
        INB             Subtract 1 to form move string length
        JMP CONSAC      Continue to concatenate string onto STRACC

CONCA1, LLI 003         Load L with address of SUBSTR length
        LBM             The SUBSTR length is move string length

CONSAC, LLI 020         Load L with address of STRACC
        LHI 045         $$ Load H with STRING page
        LAM             Get (CC) of STRACC
        ADB             Add move string length to it
        CPI 121         Compare new length with 81 decimal
        JFS BIGERR      If new length greater than 80, BG error
```



```
          LCA         Save new length in C
          LAM         Get old length in A
          LMC         Put new length back in STRACC (CC)
          ADI 021     Add address to STRACC+1 to get new address
          LCA         Store address for move in C temporarily
          LLI 005     Load L with address of SPNTR
          LDM         Get high part of SPNTR in D
          INL         Low part of SPNTR
          LAM         Get low part of SPNTR in A
          LLI 002     Load L with address of SUBSTR pointer
          ADM         Add SUBSTR length to SPNTR low byte
          LEA         Save sum in E
          LAD         Get high part SPNTR in A
          ACI 000     Add carry to high SPNTR
          LDA         Save it in D
          LHD         Put sum in H and L to point to character string
          LLE         First character after string (CC)
          LDI 045     $$ Load D with STRING page
          LEC         Put address of end of STRACC in E
          INB         Exercise B, test to see if zero
          DCB
          LAA
          CFZ MOVEPG  If len grtr or less than 0, concat SUBSTR onto STRACC

CONSA1,   LLI 002     Load L with address of SUBSTR pointer
          LHI 045     $$ Load H with STRING page
          LMI 001     Re-initialize SUBSTR pointer to 1, next string
          INL         SUBSTR length
          LMI 377     Re-initialize SUBSTR length to -1, hold next string
          RET         Return to caller

SLOOK,    LLI 013     Load L with address of NUMSTR
          LHI 045     $$ Load H with STRING page
          LCM         Put NUMSTR in C
          LLI 260     Load L with address of pointer to STRTAB
          LDM         Load D page of STRTAB
          LEI 000     Load E with address of start of STRTAB
          INC         Exercise C to test if zero
          DCC
          JTZ SLOOK3  If equal, no strings exist, create new

SLOOK1,   LLE         Put pointer to STRTAB in H and L
          LHD
          LAM         Get string name in STRTAB
          LLI 000     Load L with address of SVAR name
          LHI 045     $$ Load H with STRING page
          CPM         Compare SVAR name to STRTAB
          JFZ SLOOK2  If not equal, keep looking for SVAR
          LLE         Put pointer in H and L
          LHD
          INL         Point to STRTAB subscript
```

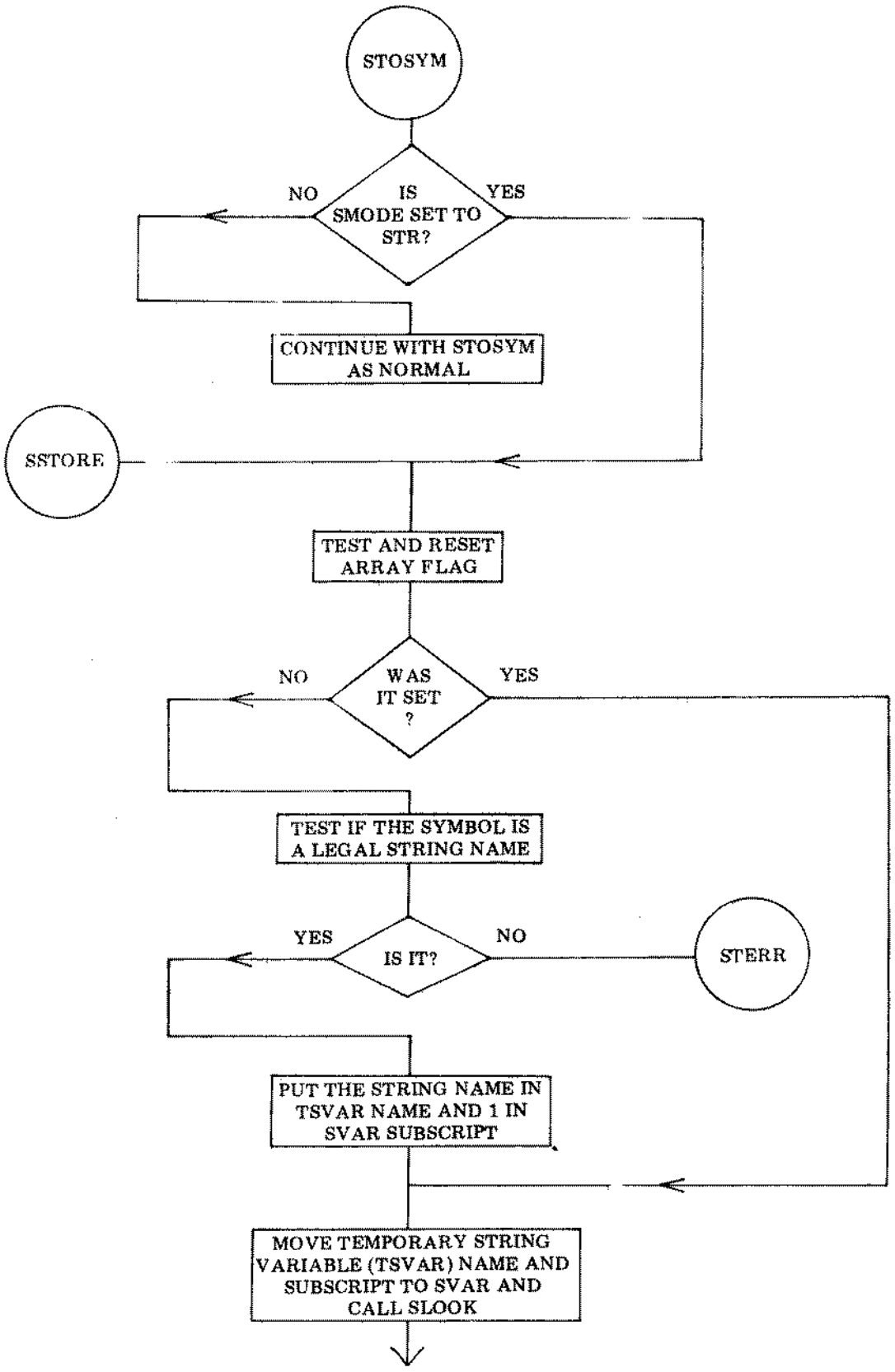

SLOOK
LOOK FOR SVAR NAME, SUBSCRIPT COMBINATION IN STRING TABLE
FOUND IT?
NO
YES
PUT POINTER FROM STRING TABLE IN SPNTR
EXIT
INCREMENT NUMSTR
GREATER THAN 64 (DECIMAL) ?
NO
YES
BIGERR
PUT NAME AND SUBSCRIPT IN STRING TABLE
PUT END OF STR POINTER IN STRING TABLE CORRESPONDING NEWEST ENTRY (ALSO PUT IN SPNTR)
INCREMENT ENDSTR POINTER AND PUT 0 THERE FOR NULL STRING
EXIT

```
          LAM       Get STRTAB subscript
          LLI 001   Load L with address of SVAR subscript
          LHI 045   $$ Load H with STRING page
          CPM       Compare STRTAB subscript with SVAR subscript
          JFZ SLOOK2 If not equal, keep looking
          LLE       Put pointer in H and L
          LHD
          INL       Add two to L to point to high part of pointer
          LHI 045   $$ Load H with STRING page
          LMB       Store new end of string back
          INL
          LMC
          LHB       Put new ENDSTR in H and L
          LLC
          LMI 000   Put 0 at location pointed to by end of string
          LCI 000   0 in C indicates new string
          RET       Return to caller

PARNB,    LCI 001   Initialize C to first PARENTHESIS
          LHI 026   ** Load H with pointer page
          LLD       Put start location in L
          INE       Add 1 to finish location

PARNB1,   LAM       Get character
          CPI 250   Is it "(" ?
          JFZ PARNB2 If not, keep looking
          INC       Increase PAREN counter
          JMP PARNB3 Continue

PARNB2,   CPI 251   Is character ")" ?
          JFZ PARNB3 If not, keep looking
          DCC       Decrement PAREN counter

PARNB3,   INC       Exercise C to test if 0
          DCC
          LBL       Put pointer in B
          RTZ       Return if parentheses balanced
          INL       Point to next character
          LAL       Get new pointer
          CPE       Test if limit
          JFZ PARNB1 If not, keep trying
          JMP PARNER If no balance, "I(" error

EVALPC,   LLI 375   Load L with address of SMODE
          LHI 026   ** Load H with pointer page
          LMI 000   Initialize SMODE to numeric

EVALS,    LLI 227   Load L with address of ARTH SP
          LHI 001   ** Load H with FP page
          JMP EVALQ Continue evaluation
```

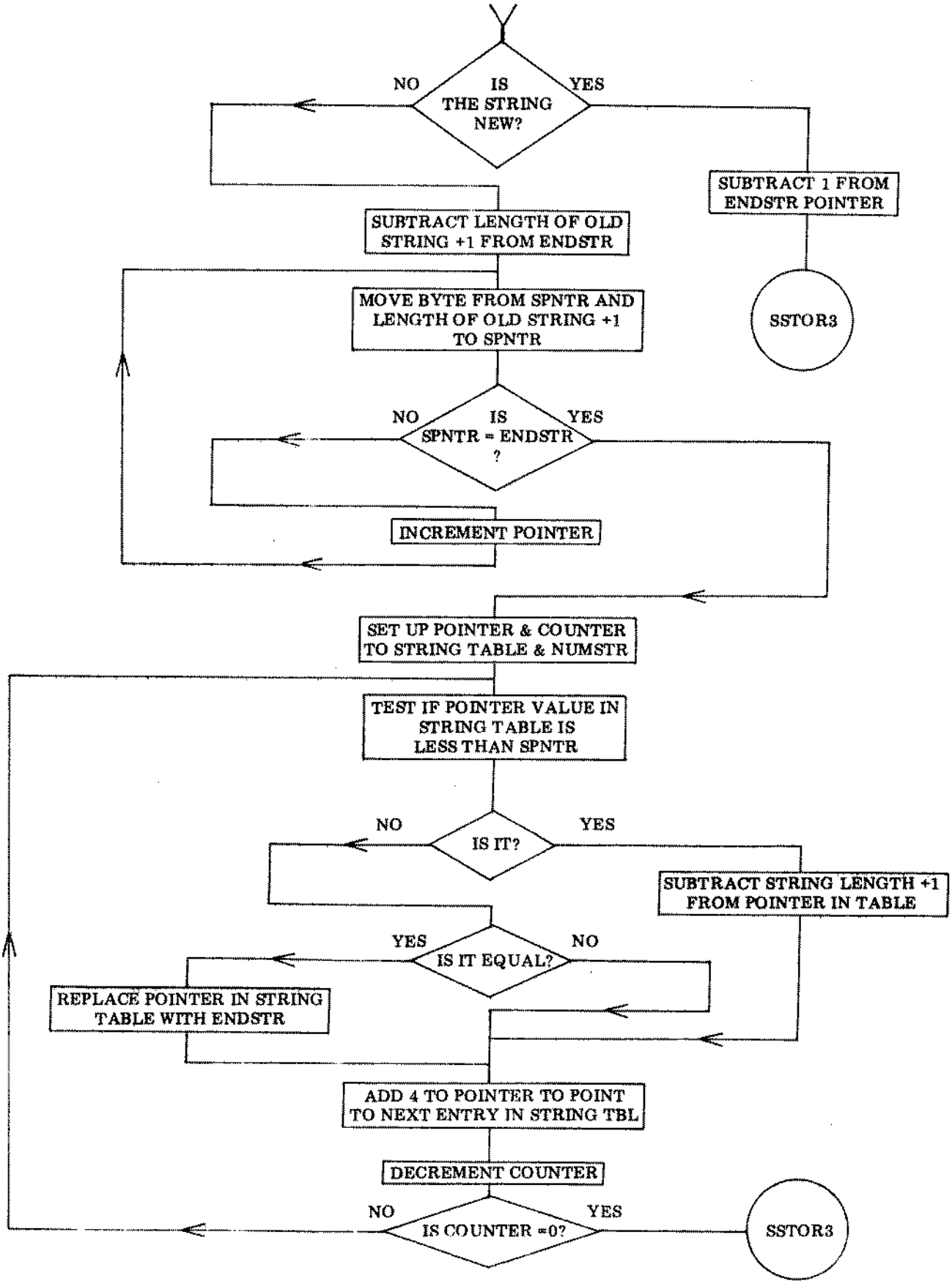

EVAL
SET STRING MODE FLAG TO INDICATE NUMERIC (0)
EVALS
EVAL AS NORMAL EXCEPT FOR FOLLOWING CHANGES
NO
IS CHAR '(' ?
YES
SFAPCH
CONTINUE WITH EVAL EXCEPT TEST FOR EQUALITY OR INEQUALITIES USING LTEQGT SUBROUTINE
NO
IS CHAR <, =, > ETC. ?
YES
PROCESS AS NORMAL
NO
IS CHAR $ OR ' OR " ?
YES
SEVAL
CONTINUE WITH EVAL, PARSE, ETC. AS BEFORE

```
SCANCP,  CPI 244        Is character "$" ?
         JTZ SEVAL      Must be string
         CPI 247        Is character single quote (') ?
         JTZ SEVAL      Must be string
         CPI 242        Is character double quote (") ?
         JTZ SEVAL      Must be string
         CAL LTEQGT     Test for inequalities
         LLI 176        Load L with address of TOKEN
         LBM            Get TOKEN
         INB            Exercise B to test if 0
         DCB
         JFZ SCANFN     If token 0, no comparison operation found
         JMP SCAN16     Found a comparison operation

LTEQGT,  LLI 176        Load L with address of TOKEN
         LMI 000        Initialize TOKEN to 0 for LTEQGT subroutine
         JMP SCAN9      Use modified part of old EVAL to check
         INL            To string corresponding to this name
         LDM            Get STRING pointer in D and E
         INL
         LEM
         LLI 005        Load L with address of SPNTR
         LHI 045        $$ Load H with STRING page
         LMD            Store D and E in SPNTR
         INL
         LME
         RET            Return to caller

SLOOK2,  LAE            Get low part of pointer in accumulator
         ADI 004        Add 4 to pointer to point to next string
         LEA            Restore pointer in E
         DCC            Decrement counter
         JFZ SLOOK1     Continue if more strings to go

SLOOK3,  LLI 013        Load L with address of NUMSTR
         LHI 045        $$ Load H with STRING page
         LBM            Add 1 to NUMSTR for new string
         INB
         LMB
         LAB            Get new NUMSTR in accumulator
         CPI 101        Compare with limit of number of strings
         JFS BIGERR     If too many, BG
         LLI 000        Load L with address of SVAR name
         LAM            Get SVAR name in accumulator
         LHD            Put pointer in H and L
         LLE
         LMA            Put SVAR name in string name for new string
         LLI 001        Load L with address of SVAR subscript
         LHI 045        $$ Load H with STRING page
         LAM            Get SVAR subscript
         LHD            Put pointer to STRTAB in H and L
```

```
        LLE
        INL         Next byte for STRTAB subscript
        LMA         Store SVAR subscript in STRTAB
        LLI 015     Load L with address of ENDSTR
        LHI 045     $$ Load H with STRING page
        LBM         Get ENDSTR in B and C
        INL
        LCM
        LLI 005     Load L with address of SPNTR
        LMB         Store ENDSTR in SPNTR since this new string
        INL
        LMC
        LHD         Put pointer to STRTAB in H and L
        LLE
        INL         Add 2 to H and L to point to STRING pointer
        INL
        LMB         Store ENDSTR in pointer in STRTAB
        INL
        LMC
        LAC         Get low part ENDSTR in C
        ADI 001     Add 1 to it
        LCA         Restore
        LAB         Get high part
        ACI 000     Add carry
        LBA         Restore
        LLI 015     Load L with address of ENDSTR

STERR,  LAI 323
        LCI 324
        JMP ERROR

IQERR,  LAI 311
        LCI 321
        JMP ERROR

STORP,  LLI 375     Load L with address of SMODE
        LHI 026     ** Load H with pointer page
        LAM         Get SMODE flag into accumulator
        NDA         Test SMODE flag
        LLI 201     Load L with address of ARRAY flag
        LHI 027     ** Load H with page of ARRAY flag
        JTZ STORP1  If SMODE flag is numeric, regular store
```

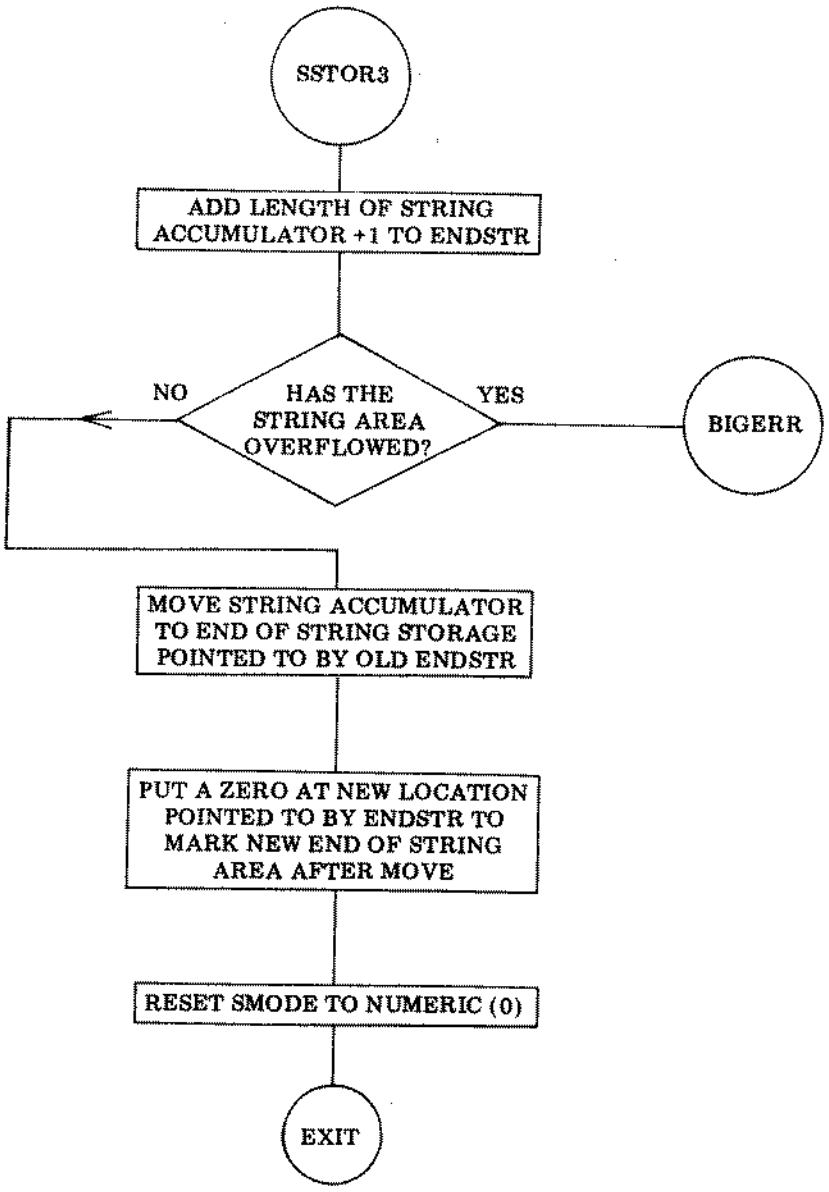

STOSYM
NO
IS SMODE SET TO STR?
YES
CONTINUE WITH STOSYM AS NORMAL
SSTORE
TEST AND RESET ARRAY FLAG
NO
WAS IT SET ?
YES
TEST IF THE SYMBOL IS A LEGAL STRING NAME
YES
IS IT?
NO
STERR
PUT THE STRING NAME IN TSVAR NAME AND 1 IN SVAR SUBSCRIPT
MOVE TEMPORARY STRING VARIABLE (TSVAR) NAME AND SUBSCRIPT TO SVAR AND CALL SLOOK

```
SSTOR,  LAM         Get ARRAY flag into accumulator
        LMI 000     Reset ARRAY flag
        NDA         Test ARRAY flag
        JFZ SSTOR1  If ARRAY flag set, already have TSVAR
        LLI 120     Load L with address of SYMBOL
        LHI 026     ** Load H with pointer page
        LAM         Get length of SYMBOL
        CPI 002     Is length 2 for letter-$ combination?
        JFZ STERR   If not, have string error
        ADL         Add address to point to last character
        LLA         Point to last character in SYMBOL
        LAM         Get last character of SYMBOL into accumulator
        CPI 244     Is last character a dollar sign ($) ?
        JFZ STERR   If not, STring ERRor
        DCL         Point to string name in SYMBOL
        LAM         Get string name
        LLI 261     Load L with address of TSVAR name
        LHI 045     $$ Load H with STRING page
        LMA         Put string name in TSVAR name
        INL         Point to TSVAR subscript
        LMI 001     Put 1 in TSVAR subscript

SSTOR1, LLI 261     Load L with address of TSVAR name
        LHI 045     $$ Load H with STRING page
        LDM         Load TSVAR name and subscript in D and E
        INL
        LEM
        LLI 000     Load L with address of SVAR name
        LMD         Store TSVAR in SVAR
        INL
        LME
        CAL SLOOK   Look up string to be stored
        LAC         Get accumulator into C to tell if string is new
        NDA         Test if C was 0, meaning string is new
        JFZ SSTOR8  If string is not new, continue below
        LLI 016     Load L with address of low part of ENDSTR
        LHI 045     $$ Load H with STRING page
        LAM         Get low part of ENDSTR in accumulator
        SUI 001     Subtract 1 from low part of ENDSTR
        LMA         Restore to ENDSTR
        DCL         Point to high part of ENDSTR
        LAM         Get high part of ENDSTR in accumulator
        SBI 000     Subtract carry from high part of ENDSTR
        LMA         Restore in ENDSTR
        JMP SSTOR3  Avoid modifying pointers since new string

SSTOR8, LLI 005     Load L with address of SPNTR
        LHI 045     $$ Load H with STRING page
        LDM         Get SPNTR into D and E
        INL
        LEM
```


IS THE STRING NEW?
NO
YES
SUBTRACT 1 FROM ENDSTR POINTER
SSTOR3
SUBTRACT LENGTH OF OLD STRING +1 FROM ENDSTR
MOVE BYTE FROM SPNTR AND LENGTH OF OLD STRING +1 TO SPNTR
IS SPNTR = ENDSTR ?
NO
YES
INCREMENT POINTER
SET UP POINTER & COUNTER TO STRING TABLE & NUMSTR
TEST IF POINTER VALUE IN STRING TABLE IS LESS THAN SPNTR
IS IT?
NO
YES
SUBTRACT STRING LENGTH +1 FROM POINTER IN TABLE
IS IT EQUAL?
YES
NO
REPLACE POINTER IN STRING TABLE WITH ENDSTR
ADD 4 TO POINTER TO POINT TO NEXT ENTRY IN STRING TBL
DECREMENT COUNTER
IS COUNTER =0?
NO
YES
SSTOR3

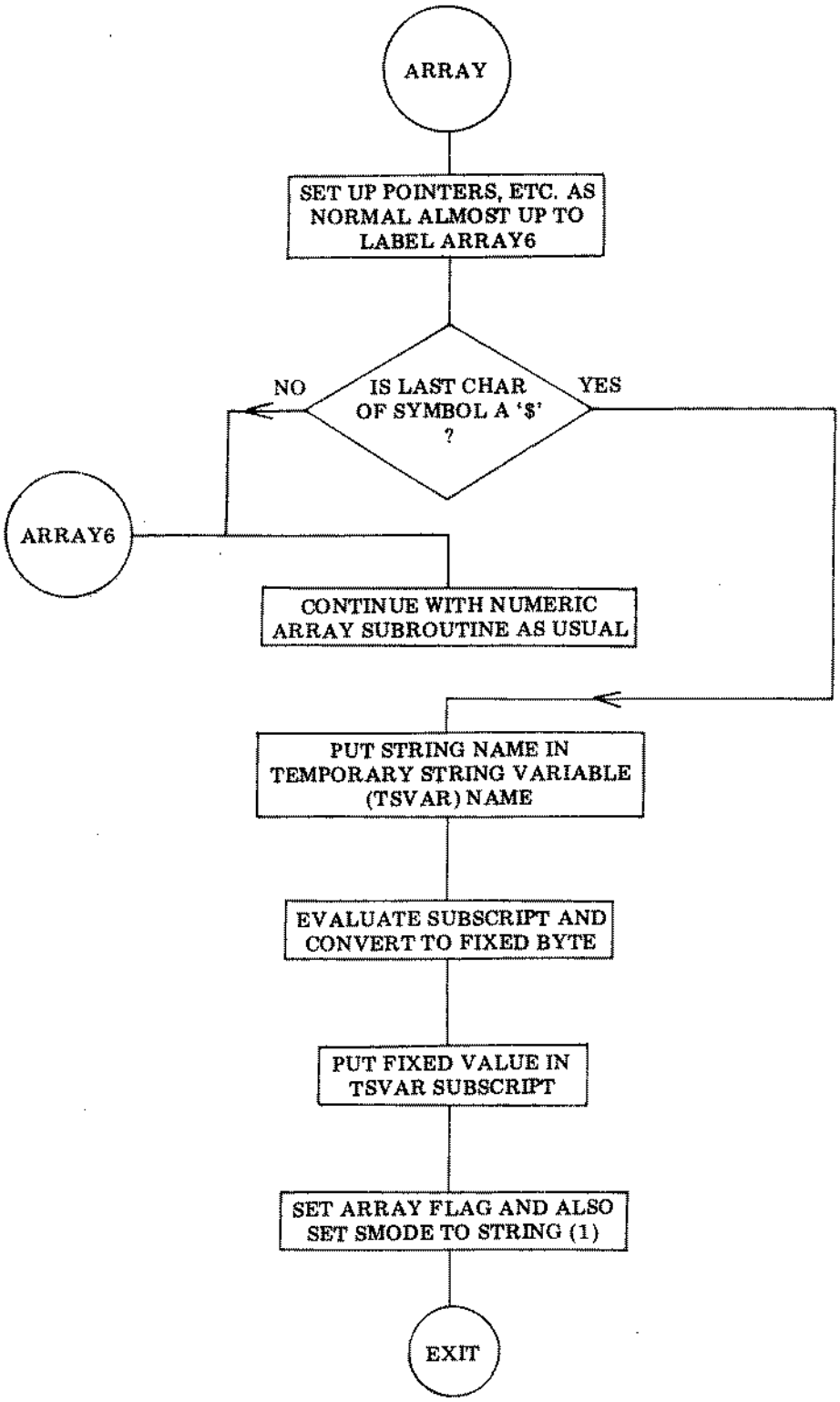
```
        LHD         Put SPNTR into H and L to point to (CC) of
        LLE         Old string value
        LCM         Get old length into register C
        LLI 007     Load L with address of string length
        LHI 045     $$ Load H with STRING page
        LMC         Store string length there
        INC         Add 1 to length of old string
        LLI 016     Load L with address of low part of ENDSTR
        LAM         Get low part of ENDSTR
        SUC         Subtract string length +1 from ENDSTR
        LMA         Restore low byte to ENDSTR
        DCL         Point to high byte of ENDSTR
        LAM         Get high byte into accumulator
        SBI 000     Subtract carry from high byte of ENDSTR
        LMA         Restore high byte to ENDSTR
SSTOR2, LAE         Get low part of SPNTR into accumulator
        ADC         Add string length +1 to SPNTR
        LLA         Point to SPNTR + length + 1
        LAD         Get high part of SPNTR into accumulator
        ACI 000     Add carry to it
        LHA         Point to high part
        LBM         Get byte of other string to be moved
        LLE         Put SPNTR in H and L
        LHD
        LMB         Store byte at new location
        LLI 015     Load L with address of ENDSTR
        LHI 045     $$ Load H with STRING page
        LAD         Get high part of SPNTR into accumulator
        CPM         Compare with high part of ENDSTR
        JFZ SSTOR9  If not same, keep moving bytes
        INL         Point to low part of ENDSTR
        LAE         Get low part of SPNTR into accumulator
        CPM         Compare low part of SPNTR with ENDSTR
        JTZ SSTOR0  If SPNTR = ENDSTR, done moving strings

SSTOR9, CAL ADVDE   Add 1 to SPNTR in D and E
        JMP SSTOR2  Keep moving strings

SSTOR0, LLI 005     Load L with address of original SPNTR
        LHI 045     $$ Load H with STRING page
        LDM         Get original SPNTR in D and E
        INL
        LEM
        LLI 013     Load L with address of NUMSTR
        LBM         Load B with NUMSTR for counter
        LLI 260     Load L with address of pointer to STRTAB
        LHM         Get page of STRTAB
        LLI 002     Load L with address of pointers in STRTAB

SSTOR4, LAM         Get high part of pointer in STRTAB
        CPD         Compare with high part of SPNTR
```

```
         JTZ SSTOR5     If same, have same string - must change pointer
         JTS SSTOR7     If pointer less than SPNTR, don't modify pointer
         JMP SSTOR6     Otherwise, may or may not be same

SSTOR5,  INL            Point to low part of pointer
         LAM            Get low part of pointer in STRTAB into accumulator
         DCL            Restore pointer back to high part
         CPE            Compare low part of pointer with low part of SPNTR
         JTC SSTOR7     If pointer less than SPNTR, don't modify pointer
         JFZ SSTOR6     If pointer greater than SPNTR, modify the pointer
         LAL            Save L register in accumulator temporarily
         LLI 263        Load L with address of temporary register storage
         LHI 045        $$ Load H with STRING page
         LMB            Save registers B and C there
         INL
         LMC
         LLI 015        Load L with address of ENDSTR
         LBM            Load B and C with ENDSTR pointer
         INL
         LCM
         LLI 260        Load L with address of pointer to STRTAB
         LHM            Point to STRTAB page
         LLA            Restore L to former value
         LMB            Replace STRTAB pointer with ENDSTR
         INL
         LMC
         LLI 263        Load L with address of temporary register storage
         LHI 045        $$ Load H with STRING page
         LBM            Restore B and C to former values
         INL
         LCM
         LLI 260        Load L with address of pointer to STRTAB
         LHM            Point to STRTAB page
         LLA            Restore L to original value
         JMP SSTOR7     Continue to modify STRTAB pointers

SSTOR6,  INL            Point to low part of pointer in STRTAB
         LAM            Get low part of pointer
         SUC            Subtract number of bytes destroyed (length +1)
         LMA            Restore modified pointer in STRTAB
         DCL            Point to high part of pointer in STRTAB
         LAM            Get high part of pointer
         SBI 000        Subtract carry from high part
         LMA            Restore high part of pointer to STRTAB

SSTOR7,  LAL            Get L into accumulator
         ADI 004        Add 4 to arrive at next pointer in STRTAB
         LLA            Restore pointer with new value
         DCB            Decrement counter of number of strings
         JFZ SSTOR4     If not done, go back and modify more
```



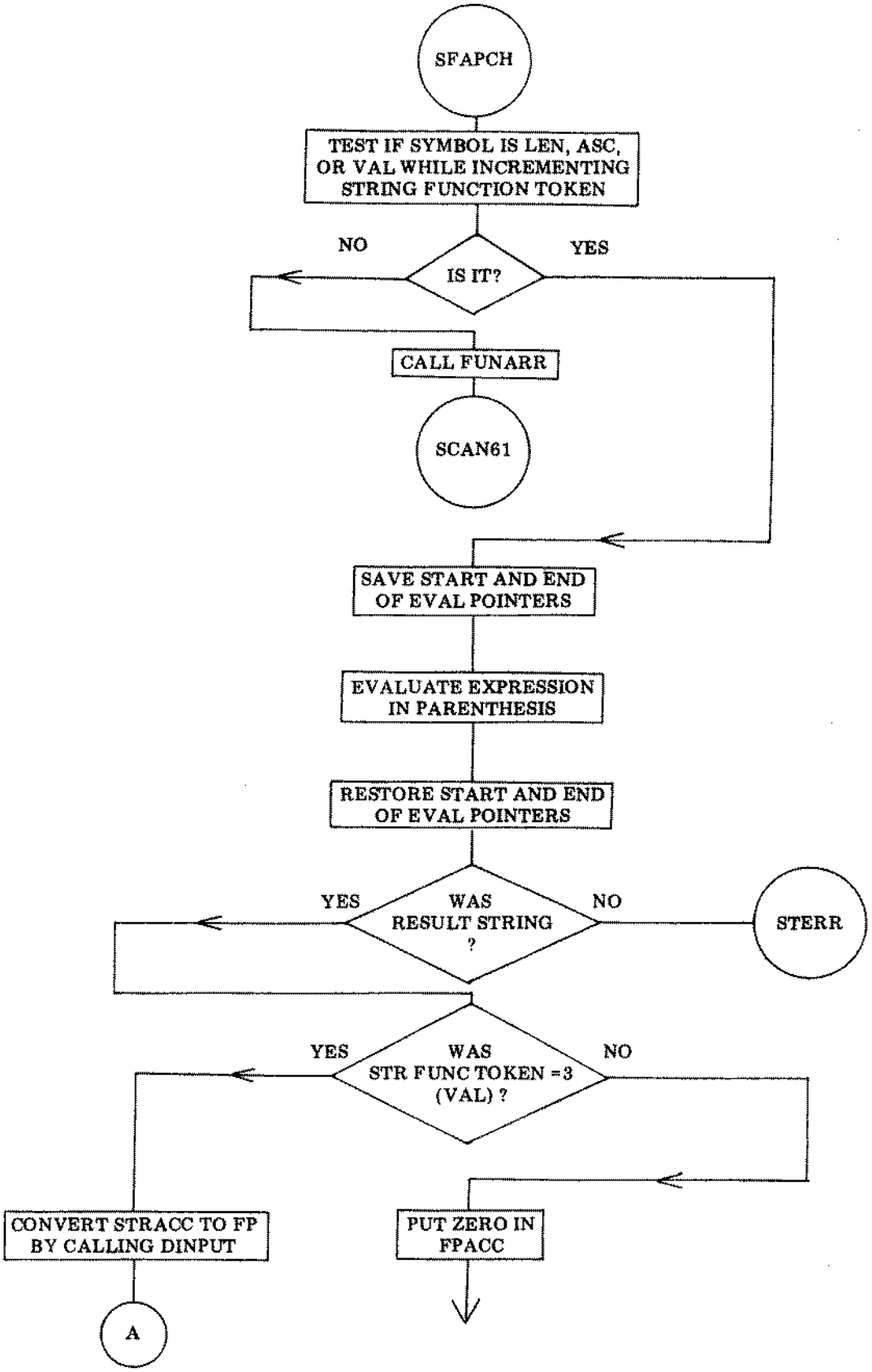

```
SSTOR3,  LLI 015      Load L with address of ENDSTR
         LHI 045      $$ Load H with STRING page
         LDM          Load D and E with ENDSTR pointer
         INL
         LEM
         LLI 020      Load L with address of STRACC
         LAM          Get (CC) of STRACC into accumulator
         ADI 001      Add 1 to (CC)
         ADE          Add low part of ENDSTR
         LEA          Restore sum in E
         LAD          Get high part of ENDSTR in A
         ACI 000      Add carry to high part of ENDSTR
         LDA          Restore value in D
```

```
        CPI 045     $$ Test against maximum page of string area
        JTZ BIGERR  If would cause string to overflow, BG error
        LLI 015     Load L with address of ENDSTR pointer
        LDM         Get old ENDSTR value in D
        LMA         Store new ENDSTR value back (high)
        INL         Point to low byte of ENDSTR
        LAM         Get low byte of old ENDSTR
        LME         Store low byte of new ENDSTR back
        LEA         Put low byte of old ENDSTR in E
        LLI 020     Load L with address of STRACC
        CAL MOVEC   Move STRACC to end of string area
        LLI 015     Load L with address of ENDSTR
        LHI 045     $$ Load H with STRING page
        LDM         Get ENDSTR pointer into D and E
        INL
        LEM
        LHD         Put ENDSTR pointer into H and L
        LLE
        LMI 000     Put 0 at new location pointed to by ENDSTR
        LLI 375     Load L with address of SMODE flag
        LHI 026     ** Load H with pointer page
        LMI 000     Reset SMODE to numeric
        RET         Return to caller

PPRINT, LLI 203     Load L with address of PRINT pointer
        CAL LOOP    Loop until end of statement
        JFZ PRINT2  If not done, go back
        JMP PRINT3  Continue as normal with PRINT

PPRIN7, LLI 203     Load L with address of PRINT pointer
        LDM         Get PRINT pointer in D
        IND         Add 1 to skip over parentheses
        LLI 000     Load L with address of (CC)
        LEM         Look for parentheses to end of statement
        CAL PARNB   Find balancing parentheses to skip unwanted ;
        LLI 203     Load L with address of PRINT pointer
        LMB         Put pointer to balance parentheses there
        JMP PPRINT  Continue to loop as normal

POUTSF, LLI 375     Load L with address of SMODE
        LHI 026     ** Load H with pointer page
        LAM         Get SMODE flag into accumulator
        NDA         Test if SMODE is numeric (0)
        JTZ PFPOUT  If so, print FP number as normal
        LLI 020     Load L with address of STRACC
        LHI 045     $$ Load H with STRING page
        JMP TEXTC   Print STRING ACC and return
```

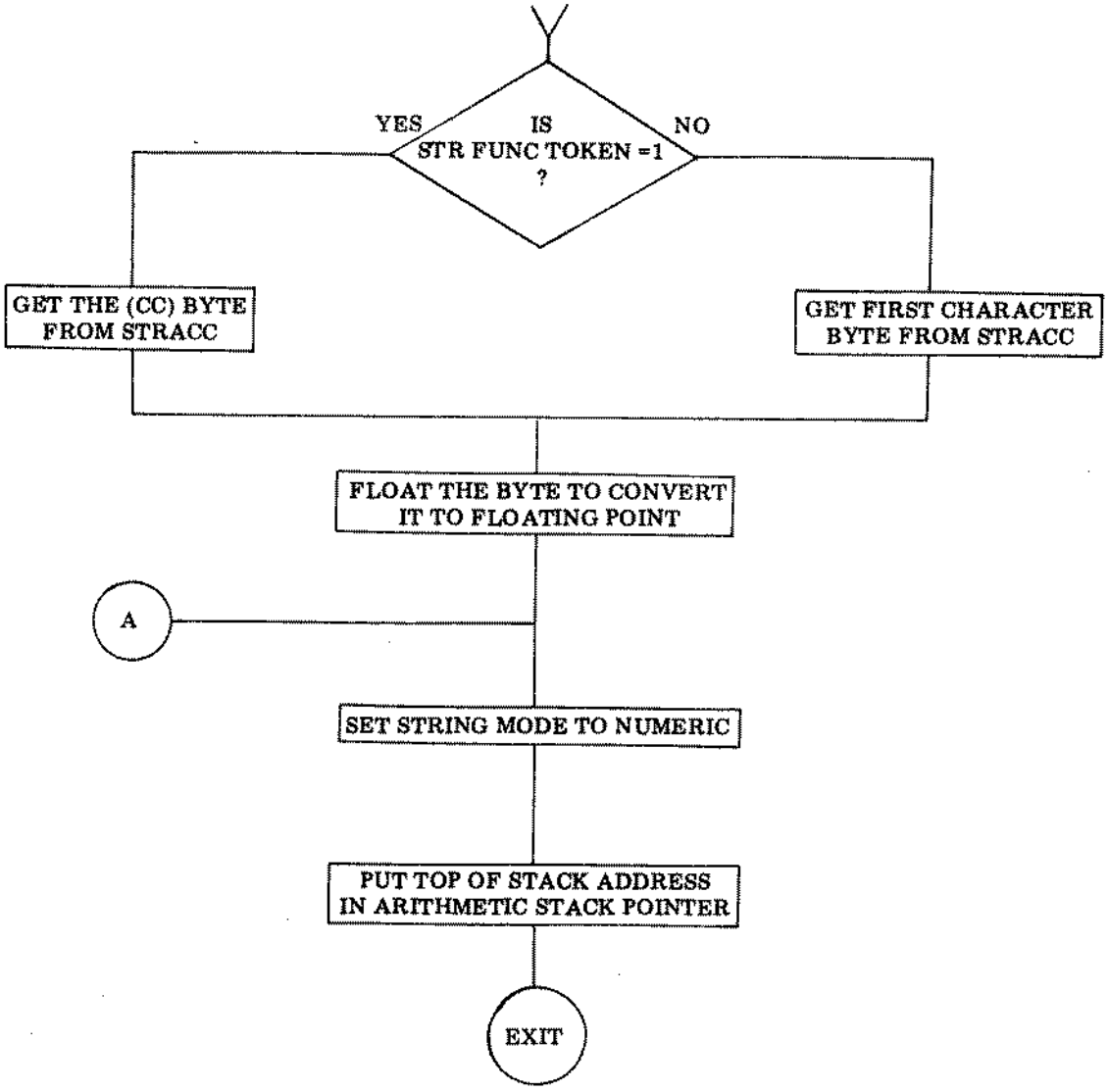

ARRAY
SET UP POINTERS, ETC. AS NORMAL ALMOST UP TO LABEL ARRAY6
IS LAST CHAR OF SYMBOL A '$' ?
NO
YES
ARRAY6
CONTINUE WITH NUMERIC ARRAY SUBROUTINE AS USUAL
PUT STRING NAME IN TEMPORARY STRING VARIABLE (TSVAR) NAME
EVALUATE SUBSCRIPT AND CONVERT TO FIXED BYTE
PUT FIXED VALUE IN TSVAR SUBSCRIPT
SET ARRAY FLAG AND ALSO SET SMODE TO STRING (1)
EXIT

```
ARRAYP,  LLI 120      Load L with address of SYMBOL
         LAM          Get (CC) of SYMBOL into accumulator
         ADL          Add L to (CC) to get last character
         LLA          Put sum in L to point to last character
         LAM          Get last character of SYMBOL
         CPI 244      Is last character dollar sign ($) ?
         JTZ SARRAY   If so, have string array
         LLI 207      Load L with address of ARRAY LOOP counter
         LMI 000      Initialize loop counter to 0
         JMP ARRAY6   Continue with numeric array as normal

SARRAY,  DCL          Get first character of SYMBOL
         LAM          Get string name into ACC
         LLI 261      Load L with address of TSVAR name
         LHI 045      $$ Load H with STRING page
         LMA          Put name of string in TSVAR name
         CAL EVAL     Evaluate subscript expression
         CAL FPFIX    Convert value of fixed point
         LLI 124      Load L with address of FPACC LSW
         LHI 001      ** Load H with address of FP page
         LAM          Get subscript into ACC
         LLI 262      Load L with address of TSVAR subscript
         LHI 045      $$ Load H with STRING page
         LMA          Put subscript in TSVAR subscript
         LLI 375      Load L with address of SMODE
         LHI 026      ** Load H with pointer page
         LMI 001      Set SMODE flag to indicate string array
         LHI 027      ** Load H with address of pointer page
         LLI 201      Load L with address of ARRAY FLAG
         LMI 001      Set ARRAY FLAG to indicate string array
         RET          Return to caller

SCRPCH,  LLI 260      Load L with address of pointer to STRTAB
         LHI 045      $$ Load H with STRING page
         LBM          Get pointer to STRTAB in B
         INB          Increment to point to string storage area
         LLI 013      Load L with address of NUMSTR
         LMI 000      Initialize to 0 to clear strings
         LLI 015      Load L with address of ENDSTR pointer
         LMB          Initialize ENDSTR to beginning of string area
         INL
         LMI 000      Starts at location 0
         LHB          Point to beginning of string area
         LLI 000      Starts at location 0
         LMI 000      Initialize to 0 for new string
         JMP EXEC     Print READY, etc.

PINPUT,  CAL CLESYM   Clear symbol
         LLI 375      Load L with address of SMODE flag
         LMI 000      Set it to numeric mode
         RET          Return from this patch
```

```
INPUTS,  LLI 375      Load L with address of SMODE flag
         LHI 026      ** Load H with pointer page
         LMI 001      Set mode to string
         LAI 277      Load accumulator with ASCII value for question mark
         CAL ECHO     Print question mark
         LLI 020      Load L with address of STRACC
         LHI 045      $$ Load H with STRING page
         JMP STRIN    Input STRACC and return

INSTRP,  CAL SAVEHL
         CAL SWITCH
         LAM
         CPI 247
         JTZ INSTRQ
         CPI 242
         JFZ SWITCH

INSTRQ,  INL
         CPM
         JTZ SWITCH
         LBH
         LCL
         LLI 000
         LHI 026
         LAM
         CPE
         LHB
         LLC
         JTZ IQERR
         JMP INSTRQ

SFAPCH,  LLI 012      Load L with address of STring FUNction TOKEN
         LHI 045      $$ Load H with STRING page
         LMI 001      Initialize STring FUNction TOKEN to 1
         LLI 360      Load L w/ address of 1st string function name (length)

SFAPC1,  LDI 026      ** Load D with pointer page
         LEI 120      Load L with address of SYMBOL
         CAL STRCP    Compare SYMBOL with string functions
         JTZ SFAPC3   If have match, process string function
         LLE          Restore H and L to point to string function names
         LHD

SFAPC2,  INL          Skip over last byte
         LAM          Get next character in string function name table
         NDI 300      Is it a character (parity marking)?
         JFZ SFAPC2   Yes, go back and try again
         LDH          Save pointer to new string function in D and E
         LEL
         LLI 012      Load L with address of string function token
         LHI 045      $$ Load H with STRING page
```


SFAPCH
TEST IF SYMBOL IS LEN, ASC, OR VAL WHILE INCREMENTING STRING FUNCTION TOKEN
NO
IS IT?
YES
CALL FUNARR
SCAN61
SAVE START AND END OF EVAL POINTERS
EVALUATE EXPRESSION IN PARENTHESIS
RESTORE START AND END OF EVAL POINTERS
YES
WAS RESULT STRING ?
NO
STERR
YES
WAS STR FUNC TOKEN =3 (VAL) ?
NO
CONVERT STRACC TO FP BY CALLING DINPUT
PUT ZERO IN FPACC
A

```
        LBM     Increment string function token
        INB
        LMB
        LHD     Restore H and L with address of
        LLE     Next string function name
        LAB     Get STring FUNction TOKEN (counter) into accumulator
        CPI 004 Is it at limit of functions?
        JFZ SFAPC1  No, keep looking
        CAL FUNARR  Yes, must be numeric function or array
        JMP SCAN61  Continue with normal evaluation

SFAPC3, LLI 276 Load L with address of start of EVAL pointer
        LHI 026 ** Load H with POINTER page
        LBM     Get start and end EVAL pointers in B and C
        INL
        LCM
        LLI 010 Load L with address of temporary start/stop storage
        LHI 045 $$ Load H with STRING page
        LMB     Store start and stop pointers there because
        INL     Of recursive calling
        LMC
        LLI 200 Load L with address of EVAL pointer
        LHI 026 ** Load H with pointer page
        LBM     Get it and add 1 to it
        INB
        LLI 276 Load L with address of start of EVAL pointer
        LMB     Start evaluation just after "("
        INL     Point to old stop evaluation pointer
        LDB     Load D with pointer to just after "("
        LEM     Load E with pointer to end of original evaluation
        CAL PARNB   Balance parentheses between "(" and end EVAL
        LLI 277 Load L with address of end of EVAL pointer
        DCB     Subtract 1 to point just before ")"
        LMB     Evaluate to just before ")"
        CAL EVAL    Evaluate string argument of function
        LLI 375 Load L with address of SMODE flag
        LHI 026 ** Load H with pointer page
        LAM     Get SMODE flag
        NDA     Test if flag is set
        JTZ STERR   If it is numeric argument, error
        LLI 010 Load L with addr of temporary start/stop EVAL pntrs
        LHI 045 $$ Load H with STRING page
        LDM     Get start and stop pointers in D and E
        INL
        LEM
        LLI 277 Load L with address of end of EVAL pointer
        LHI 026 ** Load H with pointer page
        LBM     Get end EVAL pointer in B
        INB     Add 1 to end EVAL pointer
        LLI 200 Load L with address of EVAL pointer
        LMB     Put EVAL end pointer +1 there to skip (. .)
```




```
LLI 276     Load L with address of start of EVAL pointer
LMD         Restore original start and stop pointers there
INL
LME
LLI 012     Load L with address of STring FUNction TOKEN
LHI 045     $$ Load H with STRING page
LAM         Get STring FUNction TOKEN into accumulator
CPI 003     Is it token for value?
JFZ SFAPC7  No, must be other function
LLI 020     Load L with address of STRACC
CAL DINPUT  Convert STRACC ASCII to FPACC floating point
JMP SFAPC8  Continue to process string function
```

```
SFAPC7,  CAL FP0        Put 0 in FPACC
         LLI 012        Load L with address of STring FUNction TOKEN
         LHI 045        $$ Load H with STRING page
         LAM            Get STring FUNction TOKEN
         CPI 001        Is it token for length?
         LLI 020        Set up address of STRACC in L
         LHI 045        $$ Load H with address of STRING page
         JFZ SFAPC4     Not length, must be ASCII
         LBM            Get (CC) of STRACC
         JMP SFAPC5     Continue to process length function

SFAPC4,  INL            Point to first character of STRACC
         LBM            Get first character of STRACC into B

SFAPC5,  LLI 124        Load L with address of FPACC LSW
         LHI 001        ** Load H with floating point page
         LMB            Store byte to be converted (length or ASCII)
         CAL FPFLT      Convert result to floating point

SFAPC8,  LLI 375        Load L with address of SMODE flag
         LHI 026        ** Load H with pointer page
         LMI 000        Reset SMODE to numeric
         LLI 227        Load L with address of arithmetic SP
         LHI 001        ** Load H with floating point page
         LMI 230        Reinitialize arithmetic SP to beginning
         JMP SCAN10     Continue to evaluate expression
```

STRINGS SUPPLEMENT MEMORY ALLOCATION FOR BUFFERS. TABLES, AND TEMPORARY DATA

The STRINGS SUPPLEMENT utilizes various locations in memory for the storage of counters, pointers, temporary data registers, etc. Additional memory is required for the storage of the strings themselves, to serve as string buffers, and for the strings symbolic table. The following list shows the areas used for these purposes in the assembled version of the STRINGS. SUPPLEMENT routines presented herein.

On page 26 the following locations:

LOC	
367	QUOTE Indicator
371	SEVAL Scan Pointer
372	Literal Pointer and Right Parenthesis Pointer
373	Colon Pointer
374	Semicolon Pointer
375	String Mode Indicator (SMODE)
376	Start of String Evaluation
377	End of String Evaluation

On page 45 (strings pointer page):

000		String Variable (SVAR) name
001		String Variable (SVAR) subscript
002		Substring Pointer
003		Substring Length
004		String Token
005		High Part of String Pointer (SPNTR)
006		Low Part of SPNTR
007		Length of String
010		Temporary storage of Start of Eval Pointer
011		Temporary storage of End of Eval Pointer
012		String Function Token
013		Number of Strings (NUMSTR)
014		Free
015		High Part of End of Strings Pointer (ENDSTR)
016		Low Part of ENDSTR
017		No-Concatenate Flag (NOCONCAT)
020		cc for String Accumulator (STRACC)
021		
.		
.		
.		
140		String Comparison String
.		
.		
.		
260		Indirect Pointer to page of String Table (STRTAB) At page indicated by contents of this location, and to String Storage Area starting on page following STRTAB (must be initialized to proper page, 42 in this version).
261		Temporary String Variable (TSVAR) name (Used by SSTORE)
262		TSVAR Subscript
263		Temporary Storage for Register B
264		Temporary Storage for Register C
.		
.		
.		
360	003	cc for LEN
361	314	
362	305	
363	316	
364	003	cc for ASC
365	301	
366	323	
367	303	
370	003	cc for VAL
371	326	
372	301	
373	314	

374	003	cc for CHR
375	303	
376	310	
377	322	

The page of the String Table is pointed to by location 260 on page 45. The STRTAB is organized as follows:

000	xxx	Name of first string (ASCII alphabetic character)
001	xxx	Subscript of first string
002	xxx	High part of pointer to starting address of string
003	xxx	Low part of pointer to string
004	xxx	Name of second string
005	xxx	Subscript of second string
006	xxx	High part of pointer to string
007	xxx	Low part of pointer to string
.		
.		
.	etc.	

The actual string storage is on the next page. The strings are stored in standard string format:

000	xxx	cc for string (not necessarily first string in table)
001	YYY	First character of string
.		
.		
aaa	zzz	Last character of string
	etc.	

ENDSTR points to the end of this area, where a zero byte is located.

ASSEMBLED LISTINGS OF STRINGS SUPPLEMENT

The following pages contain assembled listings of the STRINGS SUPPLEMENT routines just described in source form. Two sets of listings are provided side-by-side. One for the 8008, the other for the 8080. The listing starts with the various patches that must be made to the main portion of the original SCELBAL program. It then continues with the routines described herein as they would appear when assembled to reside in pages 46 through 54 of memory. The assembled version shown assumes that page 45 in the computer system is available as a temporary buffer/pointer/counter storage area etc. (as indicated above). Additionally, page 42 is assumed to be available for use as the strings symbol table, and pages 43 and 44 for storage of strings. If the user desires to change the memory area used for the latter two purposes, then the value stored at location 260 on page 45 in the assembled listing must be altered. If such is the case, then the upper limit of the strings storage area should be redefined (see location 060 on page 53 of the listing that follows). Additionally, the upper limit of the program buffer storage area would need to be redefined (see location 222 on page 12 of the main portion of SCELBAL

which is patched to contain 041 in the following listing).

NOTE: If the user intends to take full advantage of the capabilities of manipulating strings, then the user will undoubtably desire to allocate much more than the two pages of memory assigned in the assembled listing. The two pages allocated in the version herein provides for storage of only 7 to 8 long 80 character strings. Providing enough room for up to 64 such strings dictates assigning about 16 pages (4K) of memory. Space for such storage may be obtained by reducing the size of the user program buffer area or assigning it to memory addresses above page 54 (or page 57 if DIMension capability is utilized).

A double dollar sign indication ($$) in the following listing indicates values that would need to be changed if the supplement was reassembled in another area of memory and the temporary buffer/pointer/counter storage area was changed. Similarly, the double asterisk indicator (**) has the same meaning that it was given in the original SCELBAL publication.

8008

```
03 224  104 370 051        EVAL, JMP EVALPC
03 227  300                EVALQ, LAA

04 021  104 015 054              JMP SFAPCH
04 024  066 176            SCAN61, LLI 176

04 066  110 005 052              JFZ SCANCP

04 140  007                      RET

04 203  007                      RET

04 210  013                      RFZ
04 211  300                      LAA
04 212  300                      LAA

04 246  007                      RET

04 255  007                      RET

04 264  007                      RET

04 273  007                      RET

10 055  104 070 052              JMP STORP
10 060  300                STORP1, LAA

11 066  104 265 053              JMP SCRPCH

12 221  074 041                  CPI 041

13 021  106 347 053              CAL INSTRP

14 033  074 250                  CPI 250
14 035  150 137 053              JTZ PPRIN7
14 040  104 124 053              JMP PPRINT

14 061  104 075 014              JMP PRINT4
```

8080

```
03 224  303 370 051        EVAL, JMP EVALPC
03 227  177                EVALQ, LAA

04 021  303 015 054              JMP SFAPCH
04 024  056 176            SCAN61, LLI 176

04 066  302 005 052              JFZ SCANCP

04 140  311                      RET

04 203  311                      RET

04 210  300                      RFZ
04 211  177                      LAA
04 212  177                      LAA

04 246  311                      RET

04 255  311                      RET

04 264  311                      RET

04 273  311                      RET

10 055  303 070 052              JMP STORP
10 060  177                STORP1, LAA

11 066  303 265 053              JMP SCRPCH

12 221  376 041                  CPI 041

13 021  315 347 053              CAL INSTRP

14 033  376 250                  CPI 250
14 035  312 137 053              JTZ PPRIN7
14 040  303 124 053              JMP PPRINT

14 061  303 075 014              JMP PRINT4
```

8008					8080			
14 114	152 157 053		PRINT5,	CTZ POUTSF	14 114	314 157 053		PRINT5, CTZ POUTSF
14 206	300			LAA	14 206	177		LAA
14 207	300			LAA	14 207	177		LAA
14 210	300			LAA	14 210	177		LAA
14 215	300			LAA	14 215	177		LAA
14 216	300			LAA	14 216	177		LAA
14 220	066 203			LLI 203	14 220	056 203		LLI 203
14 230	150 124 053			JTZ PPRINT	14 230	312 124 053		JTZ PPRINT
14 233	300			LAA	14 233	177		LAA
14 234	300			LAA	14 234	177		LAA
14 235	300			LAA	14 235	177		LAA
14 236	066 203			LLI 203	14 236	056 203		LLI 203
16 365	106 315 053			CAL PINPUT	16 365	315 315 053		CAL PINPUT
17 114	150 325 053			JTZ INPUTS	17 114	312 325 053		JTZ INPUTS
17 117	066 375			LLI 375	17 117	056 375		LLI 375
17 121	307			LAM	17 121	176		LAM
17 122	240			NDA	17 122	247		NDA
17 123	110 325 053			JFZ INPUTS	17 123	302 325 053		JFZ INPUTS
17 126	076 000			LMI 000	17 126	066 000		LMI 000
17 130	104 140 017			JMP INPUTN	17 130	303 140 017		JMP INPUTN
45 260	042	$$		042	45 260	042	$$	042
45 360	003			003 /LEN	45 360	003		003 /LEN
45 361	314			314	45 361	314		314
45 362	305			305	45 362	305		305
45 363	316			316	45 363	316		316
45 364	003			003 /ASC	45 364	003		003 /ASC
45 365	301			301	45 365	301		301
45 366	323			323	45 366	323		323
45 367	303			303	45 367	303		303
45 370	003			003 /VAL	45 370	003		003 /VAL
45 371	326			326	45 371	326		326
45 372	301			301	45 372	301		301
45 373	314			314	45 373	314		314
45 374	003			003 /CHR	45 374	003		003 /CHR
45 375	303			303	45 375	303		303
45 376	310			310	45 376	310		310
45 377	322			322	45 377	322		322
55 234	104 177 053			JMP ARRAYP	55 234	303 177 053		JMP ARRAYP
55 237	300			LAA	55 237	177		LAA
46 000	066 017		SEVAL,	LLI 017	46 000	056 017		SEVAL, LLI 017
46 002	056 045	$$		LHI 045	46 002	046 045	$$	LHI 045
46 004	076 000			LMI 000	46 004	066 000		LMI 000
46 006	060			INL	46 006	054		INL
46 007	076 000			LMI 000	46 007	066 000		LMI 000
46 011	066 002			LLI 002	46 011	056 002		LLI 002

8008				8080			
46 013	076 001		LMI 001	46 013	066 001		LMI 001
46 015	060		INL	46 015	054		INL
46 016	076 377		LMI 377	46 016	066 377		LMI 377
46 020	060		INL	46 020	054		INL
46 021	076 000		LMI 000	46 021	066 000		LMI 000
46 023	106 255 002		CAL CLESYM	46 023	315 255 002		CAL CLESYM
46 026	066 375		LLI 375	46 026	056 375		LLI 375
46 030	076 001		LMI 001	46 030	066 001		LMI 001
46 032	066 276		LLI 276	46 032	056 276		LLI 276
46 034	317		LBM	46 034	106		LBM
46 035	060		INL	46 035	054		INL
46 036	327		LCM	46 036	116		LCM
46 037	066 376		LLI 376	46 037	056 376		LLI 376
46 041	371		LMB	46 041	160		LMB
46 042	060		INL	46 042	054		INL
46 043	372		LMC	46 043	161		LMC
46 044	066 371		LLI 371	46 044	056 371		LLI 371
46 046	371		LMB	46 046	160		LMB
46 047	066 371		SEVAL1, LLI 371	46 047	056 371		SEVAL1, LLI 371
46 051	056 026	**	LHI 026	46 051	046 026	**	LHI 026
46 053	106 240 002		CAL GETCHR	46 053	315 240 002		CAL GETCHR
46 056	150 047 050		JTZ FSEVAL	46 056	312 047 050		JTZ FSEVAL
46 061	074 247		CPI 247	46 061	376 247		CPI 247
46 063	150 073 046		JTZ SEVAL2	46 063	312 073 046		JTZ SEVAL2
46 066	074 242		CPI 242	46 066	376 242		CPI 242
46 070	110 162 046		JFZ SEVAL5	46 070	302 162 046		JFZ SEVAL5
46 073	066 367		SEVAL2, LLI 367	46 073	056 367		SEVAL2, LLI 367
46 075	370		LMA	46 075	167		LMA
46 076	066 371		LLI 371	46 076	056 371		LLI 371
46 100	317		LBM	46 100	106		LBM
46 101	010		INB	46 101	004		INB
46 102	060		INL	46 102	054		INL
46 103	371		LMB	46 103	160		LMB
46 104	066 372		SEVAL3, LLI 372	46 104	056 372		SEVAL3, LLI 372
46 106	106 240 002		CAL GETCHR	46 106	315 240 002		CAL GETCHR
46 111	066 367		LLI 367	46 111	056 367		LLI 367
46 113	277		CPM	46 113	276		CPM
46 114	150 143 046		JTZ SEVAL4	46 114	312 143 046		JTZ SEVAL4
46 117	066 020		LLI 020	46 117	056 020		LLI 020
46 121	056 045	$$	LHI 045	46 121	046 045	$$	LHI 045
46 123	106 314 002		CAL CONCT1	46 123	315 314 002		CAL CONCT1
46 126	066 372		LLI 372	46 126	056 372		LLI 372
46 130	056 026	**	LHI 026	46 130	046 026	**	LHI 026
46 132	106 243 050		CAL SELOOP	46 132	315 243 050		CAL SELOOP
46 135	110 104 046		JFZ SEVAL3	46 135	302 104 046		JFZ SEVAL3
46 140	104 060 052		JMP IQERR	46 140	303 060 052		JMP IQERR
46 143	066 372		SEVAL4, LLI 372	46 143	056 372		SEVAL4, LLI 372
46 145	317		LBM	46 145	106		LBM
46 146	066 371		LLI 371	46 146	056 371		LLI 371
46 150	371		LMB	46 150	160		LMB
46 151	066 017		LLI 017	46 151	056 017		LLI 017
46 153	056 045	$$	LHI 045	46 153	046 045	$$	LHI 045
46 155	076 001		LMI 001	46 155	066 001		LMI 001
46 157	104 047 050		JMP FSEVAL	46 157	303 047 050		JMP FSEVAL

8008

```
46 162  074 244          SEVAL5,  CPI 244
46 164  110 331 046               JFZ SEVAL6
46 167  046 374                   LEI 374
46 171  036 045     $$            LDI 045
46 173  066 120                   LLI 120
46 175  106 332 002               CAL STRCP
46 200  150 235 046               JTZ CHR
46 203  066 120                   LLI 120
46 205  056 026     **            LHI 026
46 207  307                       LAM
46 210  074 001                   CPI 001
46 212  110 051 052               JFZ STERR
46 215  060                       INL
46 216  307                       LAM
46 217  066 000                   LLI 000
46 221  056 045     $$            LHI 045
46 223  370                       LMA
46 224  060                       INL
46 225  076 001                   LMI 001
46 227  106 255 002               CAL CLESYM
46 232  104 047 050               JMP FSEVAL

46 235  066 371          CHR,     LLI 371
46 237  317                       LBM
46 240  010                       INB
46 241  371                       LMB
46 242  361                       LLB
46 243  307                       LAM
46 244  074 250                   CPI 250
46 246  110 235 046               JFZ CHR
46 251  066 276                   LLI 276
46 253  010                       INB
46 254  371                       LMB
46 255  331                       LDB
46 256  066 377                   LLI 377
46 260  347                       LEM
46 261  106 325 051               CAL PARNB
46 264  066 277                   LLI 277
46 266  011                       DCB
46 267  371                       LMB
46 270  106 376 051               CAL EVALS
46 273  106 000 020               CAL FPFIX
46 276  066 124                   LLI 124
46 300  307                       LAM
46 301  066 020                   LLI 020
46 303  056 045     $$            LHI 045
46 305  106 314 002               CAL CONCT1
46 310  066 017                   LLI 017
46 312  076 001                   LMI 001
46 314  106 255 002               CAL CLESYM
46 317  066 277                   LLI 277
46 321  317                       LBM
46 322  010                       INB
46 323  066 371                   LLI 371
46 325  371                       LMB
46 326  104 047 050               JMP FSEVAL
```

8080

```
46 162  376 244          SEVAL5,  CPI 244
46 164  302 331 046               JFZ SEVAL6
46 167  036 374                   LEI 374
46 171  026 045     $$            LDI 045
46 173  056 120                   LLI 120
46 175  315 332 002               CAL STRCP
46 200  312 235 046               JTZ CHR
46 203  056 120                   LLI 120
46 205  046 026     **            LHI 026
46 207  176                       LAM
46 210  376 001                   CPI 001
46 212  302 051 052               JFZ STERR
46 215  054                       INL
46 216  176                       LAM
46 217  056 000                   LLI 000
46 221  046 045     $$            LHI 045
46 223  167                       LMA
46 224  054                       INL
46 225  066 001                   LMI 001
46 227  315 255 002               CAL CLESYM
46 232  303 047 050               JMP FSEVAL

46 235  056 371          CHR,     LLI 371
46 237  106                       LBM
46 240  004                       INB
46 241  160                       LMB
46 242  150                       LLB
46 243  176                       LAM
46 244  376 250                   CPI 250
46 246  302 235 046               JFZ CHR
46 251  056 276                   LLI 276
46 253  004                       INB
46 254  160                       LMB
46 255  120                       LDB
46 256  056 377                   LLI 377
46 260  136                       LEM
46 261  315 325 051               CAL PARNB
46 264  056 277                   LLI 277
46 266  005                       DCB
46 267  160                       LMB
46 270  315 376 051               CAL EVALS
46 273  315 000 020               CAL FPFIX
46 276  056 124                   LLI 124
46 300  176                       LAM
46 301  056 020                   LLI 020
46 303  046 045     $$            LHI 045
46 305  315 314 002               CAL CONCT1
46 310  056 017                   LLI 017
46 312  066 001                   LMI 001
46 314  315 255 002               CAL CLESYM
46 317  056 277                   LLI 277
46 321  106                       LBM
46 322  004                       INB
46 323  056 371                   LLI 371
46 325  160                       LMB
46 326  303 047 050               JMP FSEVAL
```

8008

```
46 331  074 250          SEVAL6,  CPI 250
46 333  110 326 047               JFZ SEVA13
46 336  066 371                   LLI 371
46 340  337                       LDM
46 341  030                       IND
46 342  066 377                   LLI 377
46 344  347                       LEM
46 345  106 325 051               CAL PARNB
46 350  066 372                   LLI 372
46 352  371                       LMB
46 353  061                       DCL
46 354  317                       LBM
46 355  010                       INB
46 356  361                       LLB
46 357  307                       LAM
46 360  074 272                   CPI 272
46 362  110 373 046               JFZ SEVA16
46 365  066 373                   LLI 373
46 367  371                       LMB
46 370  104 124 047               JMP SEVA10

46 373  066 373          SEVA16,  LLI 373
46 375  371                       LMB

46 376  066 373          SEVAL7,  LLI 373
47 000  106 240 002               CAL GETCHR
47 003  074 272                   CPI 272
47 005  110 047 047               JFZ SEVAL8
47 010  066 371                   LLI 371
47 012  317                       LBM
47 013  010                       INB
47 014  066 276                   LLI 276
47 016  371                       LMB
47 017  066 373                   LLI 373
47 021  317                       LBM
47 022  011                       DCB
47 023  066 277                   LLI 277
47 025  371                       LMB
47 026  106 376 051               CAL EVALS
47 031  106 000 020               CAL FPFIX
47 034  066 124                   LLI 124
47 036  307                       LAM
47 037  066 001                   LLI 001
47 041  056 045          $$       LHI 045
47 043  370                       LMA
47 044  104 124 047               JMP SEVA10

47 047  066 373          SEVAL8,  LLI 373
47 051  106 254 050               CAL S2LOOP
47 054  110 376 046               JFZ SEVAL7

47 057  066 371          SEVAL9,  LLI 371
47 061  317                       LBM
47 062  010                       INB
47 063  060                       INL
47 064  327                       LCM
47 065  021                       DCC
```

8080

```
46 331  376 250          SEVAL6,  CPI 250
46 333  302 326 047               JFZ SEVA13
46 336  056 371                   LLI 371
46 340  126                       LDM
46 341  024                       IND
46 342  056 377                   LLI 377
46 344  136                       LEM
46 345  315 325 051               CAL PARNB
46 350  056 372                   LLI 372
46 352  160                       LMB
46 353  055                       DCL
46 354  106                       LBM
46 355  004                       INB
46 356  150                       LLB
46 357  176                       LAM
46 360  376 272                   CPI 272
46 362  302 373 046               JFZ SEVA16
46 365  056 373                   LLI 373
46 367  160                       LMB
46 370  303 124 047               JMP SEVA10

46 373  056 373          SEVA16,  LLI 373
46 375  160                       LMB

46 376  056 373          SEVAL7,  LLI 373
47 000  315 240 002               CAL GETCHR
47 003  376 272                   CPI 272
47 005  302 047 047               JFZ SEVAL8
47 010  056 371                   LLI 371
47 012  106                       LBM
47 013  004                       INB
47 014  056 276                   LLI 276
47 016  160                       LMB
47 017  056 373                   LLI 373
47 021  106                       LBM
47 022  005                       DCB
47 023  056 277                   LLI 277
47 025  160                       LMB
47 026  315 376 051               CAL EVALS
47 031  315 000 020               CAL FPFIX
47 034  056 124                   LLI 124
47 036  176                       LAM
47 037  056 001                   LLI 001
47 041  046 045          $$       LHI 045
47 043  167                       LMA
47 044  303 124 047               JMP SEVA10

47 047  056 373          SEVAL8,  LLI 373
47 051  315 254 050               CAL S2LOOP
47 054  302 376 046               JFZ SEVAL7

47 057  056 371          SEVAL9,  LLI 371
47 061  106                       LBM
47 062  004                       INB
47 063  054                       INL
47 064  116                       LCM
47 065  015                       DCC
```

8008					8080				
47 066	066 276			LLI 276	47 066	056 276			LLI 276
47 070	371			LMB	47 070	160			LMB
47 071	060			INL	47 071	054			INL
47 072	372			LMC	47 072	161			LMC
47 073	106 376 051			CAL EVALS	47 073	315 376 051			CAL EVALS
47 076	106 000 020			CAL FPFIX	47 076	315 000 020			CAL FPFIX
47 101	066 124			LLI 124	47 101	056 124			LLI 124
47 103	307			LAM	47 103	176			LAM
47 104	066 001			LLI 001	47 104	056 001			LLI 001
47 106	056 045	$$		LHI 045	47 106	046 045	$$		LHI 045
47 110	370			LMA	47 110	167			LMA
47 111	066 372			LLI 372	47 111	056 372			LLI 372
47 113	056 026	**		LHI 026	47 113	046 026	**		LHI 026
47 115	317			LBM	47 115	106			LBM
47 116	066 371			LLI 371	47 116	056 371			LLI 371
47 120	371			LMB	47 120	160			LMB
47 121	104 047 050			JMP FSEVAL	47 121	303 047 050			JMP FSEVAL
47 124	066 373		SEVA10,	LLI 373	47 124	056 373		SEVA10,	LLI 373
47 126	056 026	**		LHI 026	47 126	046 026	**		LHI 026
47 130	317			LBM	47 130	106			LBM
47 131	010			INB	47 131	004			INB
47 132	060			INL	47 132	054			INL
47 133	371			LMB	47 133	160			LMB
47 134	066 374		SEVA11,	LLI 374	47 134	056 374		SEVA11,	LLI 374
47 136	106 240 002			CAL GETCHR	47 136	315 240 002			CAL GETCHR
47 141	074 273			CPI 273	47 141	376 273			CPI 273
47 143	110 247 047			JFZ SEVA12	47 143	302 247 047			JFZ SEVA12
47 146	066 373			LLI 373	47 146	056 373			LLI 373
47 150	317			LBM	47 150	106			LBM
47 151	010			INB	47 151	004			INB
47 152	060			INL	47 152	054			INL
47 153	327			LCM	47 153	116			LCM
47 154	021			DCC	47 154	015			DCC
47 155	066 276			LLI 276	47 155	056 276			LLI 276
47 157	371			LMB	47 157	160			LMB
47 160	060			INL	47 160	054			INL
47 161	372			LMC	47 161	161			LMC
47 162	106 376 051			CAL EVALS	47 162	315 376 051			CAL EVALS
47 165	106 000 020			CAL FPFIX	47 165	315 000 020			CAL FPFIX
47 170	066 124			LLI 124	47 170	056 124			LLI 124
47 172	307			LAM	47 172	176			LAM
47 173	066 002			LLI 002	47 173	056 002			LLI 002
47 175	056 045	$$		LHI 045	47 175	046 045	$$		LHI 045
47 177	370			LMA	47 177	167			LMA
47 200	066 374			LLI 374	47 200	056 374			LLI 374
47 202	056 026	**		LHI 026	47 202	046 026	**		LHI 026
47 204	317			LBM	47 204	106			LBM
47 205	010			INB	47 205	004			INB
47 206	066 372			LLI 372	47 206	056 372			LLI 372
47 210	327			LCM	47 210	116			LCM
47 211	021			DCC	47 211	015			DCC
47 212	066 276			LLI 276	47 212	056 276			LLI 276
47 214	371			LMB	47 214	160			LMB
47 215	060			INL	47 215	054			INL
47 216	372			LMC	47 216	161			LMC
47 217	106 376 051			CAL EVALS	47 217	315 376 051			CAL EVALS

8008

```
47 222  106 000 020          CAL FPFIX
47 225  066 124              LLI 124
47 227  307                  LAM
47 230  066 003              LLI 003
47 232  056 045      $$      LHI 045
47 234  370                  LMA
47 235  066 372              LLI 372
47 237  056 026      **      LHI 026
47 241  317                  LBM
47 242  061                  DCL
47 243  371                  LMB
47 244  104 047 050          JMP FSEVAL

47 247  066 374      SEVA12, LLI 374
47 251  106 254 050          CAL S2LOOP
47 254  110 134 047          JFZ SEVA11
47 257  066 373              LLI 373
47 261  317                  LBM
47 262  010                  INB
47 263  061                  DCL
47 264  327                  LCM
47 265  021                  DCC
47 266  066 276              LLI 276
47 270  371                  LMB
47 271  060                  INL
47 272  372                  LMC
47 273  106 376 051          CAL EVALS
47 276  106 000 020          CAL FPFIX
47 301  066 124              LLI 124
47 303  307                  LAM
47 304  066 002              LLI 002
47 306  056 045      $$      LHI 045
47 310  370                  LMA
47 311  060                  INL
47 312  076 377              LMI 377
47 314  066 372              LLI 372
47 316  056 026      **      LHI 026
47 320  317                  LBM
47 321  061                  DCL
47 322  371                  LMB
47 323  104 047 050          JMP FSEVAL

47 326  074 253      SEVA13, CPI 253
47 330  110 351 047          JFZ SEVA14
47 333  066 017              LLI 017
47 335  056 045      $$      LHI 045
47 337  307                  LAM
47 340  076 000              LMI 000
47 342  240                  NDA
47 343  152 350 050          CTZ CONCAT
47 346  104 047 050          JMP FSEVAL

47 351  066 371      SEVA14, LLI 371
47 353  317                  LBM
47 354  066 200              LLI 200
47 356  371                  LMB
47 357  106 042 052          CAL LTEQGT
47 362  066 176              LLI 176
47 364  317                  LBM
```

8080

```
47 222  315 000 020          CAL FPFIX
47 225  056 124              LLI 124
47 227  176                  LAM
47 230  056 003              LLI 003
47 232  046 045      $$      LHI 045
47 234  167                  LMA
47 235  056 372              LLI 372
47 237  046 026      **      LHI 026
47 241  106                  LBM
47 242  055                  DCL
47 243  160                  LMB
47 244  303 047 050          JMP FSEVAL

47 247  056 374      SEVA12, LLI 374
47 251  315 254 050          CAL S2LOOP
47 254  302 134 047          JFZ SEVA11
47 257  056 373              LLI 373
47 261  106                  LBM
47 262  004                  INB
47 263  055                  DCL
47 264  116                  LCM
47 265  015                  DCC
47 266  056 276              LLI 276
47 270  160                  LMB
47 271  054                  INL
47 272  161                  LMC
47 273  315 376 051          CAL EVALS
47 276  315 000 020          CAL FPFIX
47 301  056 124              LLI 124
47 303  176                  LAM
47 304  056 002              LLI 002
47 306  046 045      $$      LHI 045
47 310  167                  LMA
47 311  054                  INL
47 312  066 377              LMI 377
47 314  056 372              LLI 372
47 316  046 026      **      LHI 026
47 320  106                  LBM
47 321  055                  DCL
47 322  160                  LMB
47 323  303 047 050          JMP FSEVAL

47 326  376 253      SEVA13, CPI 253
47 330  302 351 047          JFZ SEVA14
47 333  056 017              LLI 017
47 335  046 045      $$      LHI 045
47 337  176                  LAM
47 340  066 000              LMI 000
47 342  247                  NDA
47 343  314 350 050          CTZ CONCAT
47 346  303 047 050          JMP FSEVAL

47 351  056 371      SEVA14, LLI 371
47 353  106                  LBM
47 354  056 200              LLI 200
47 356  160                  LMB
47 357  315 042 052          CAL LTEQGT
47 362  056 176              LLI 176
47 364  106                  LBM
```

8008

```
47 365  010                    INB
47 366  011                    DCB
47 367  150 044 050            JTZ SEVA15
47 372  301                    LAB
47 373  024 010                SUI 010
47 375  066 004                LLI 004
47 377  056 045        $$      LHI 045
50 001  370                    LMA
50 002  066 017                LLI 017
50 004  307                    LAM
50 005  076 000                LMI 000
50 007  240                    NDA
50 010  152 350 050            CTZ CONCAT
50 013  056 045        $$      LHI 045
50 015  066 020                LLI 020
50 017  335                    LDH
50 020  046 140                LEI 140
50 022  106 046 012            CAL MOVEC
50 025  066 020                LLI 020
50 027  076 000                LMI 000
50 031  066 200                LLI 200
50 033  056 026        **      LHI 026
50 035  317                    LBM
50 036  066 371                LLI 371
50 040  371                    LMB
50 041  104 047 050            JMP FSEVAL

50 044  106 310 002    SEVA15, CAL CONCTS

50 047  066 371        FSEVAL, LLI 371
50 051  056 026        **      LHI 026
50 053  106 243 050            CAL SELOOP
50 056  110 047 046            JFZ SEVAL1
50 061  066 017                LLI 017
50 063  056 045        $$      LHI 045
50 065  307                    LAM
50 066  240                    NDA
50 067  152 350 050            CTZ CONCAT
50 072  066 376                LLI 376
50 074  056 026        **      LHI 026
50 076  317                    LBM
50 077  060                    INL
50 100  327                    LCM
50 101  066 276                LLI 276
50 103  371                    LMB
50 104  060                    INL
50 105  372                    LMC
50 106  066 004                LLI 004
50 110  056 045        $$      LHI 045
50 112  307                    LAM
50 113  240                    NDA
50 114  053                    RTZ
50 115  066 375                LLI 375
50 117  056 026        **      LHI 026
50 121  076 000                LMI 000
50 123  066 140                LLI 140
50 125  056 045        $$      LHI 045
50 127  335                    LDH
```

8080

```
47 365  004                    INB
47 366  005                    DCB
47 367  312 044 050            JTZ SEVA15
47 372  170                    LAB
47 373  326 010                SUI 010
47 375  056 004                LLI 004
47 377  046 045        $$      LHI 045
50 001  167                    LMA
50 002  056 017                LLI 017
50 004  176                    LAM
50 005  066 000                LMI 000
50 007  247                    NDA
50 010  314 350 050            CTZ CONCAT
50 013  046 045        $$      LHI 045
50 015  056 020                LLI 020
50 017  124                    LDH
50 020  036 140                LEI 140
50 022  315 046 012            CAL MOVEC
50 025  056 020                LLI 020
50 027  066 000                LMI 000
50 031  056 200                LLI 200
50 033  046 026        **      LHI 026
50 035  106                    LBM
50 036  056 371                LLI 371
50 040  160                    LMB
50 041  303 047 050            JMP FSEVAL

50 044  315 310 002    SEVA15, CAL CONCTS

50 047  056 371        FSEVAL, LLI 371
50 051  046 026        **      LHI 026
50 053  315 243 050            CAL SELOOP
50 056  302 047 046            JFZ SEVAL1
50 061  056 017                LLI 017
50 063  046 045        $$      LHI 045
50 065  176                    LAM
50 066  247                    NDA
50 067  314 350 050            CTZ CONCAT
50 072  056 376                LLI 376
50 074  046 026        **      LHI 026
50 076  106                    LBM
50 077  054                    INL
50 100  116                    LCM
50 101  056 276                LLI 276
50 103  160                    LMB
50 104  054                    INL
50 105  161                    LMC
50 106  056 004                LLI 004
50 110  046 045        $$      LHI 045
50 112  176                    LAM
50 113  247                    NDA
50 114  310                    RTZ
50 115  056 375                LLI 375
50 117  046 026        **      LHI 026
50 121  066 000                LMI 000
50 123  056 140                LLI 140
50 125  046 045        $$      LHI 045
50 127  124                    LDH
```

8008

```
50 130  046 020              LEI 020
50 132  074 001              CPI 001
50 134  110 147 050          JFZ STC1
50 137  106 267 050          CAL SSTRCP
50 142  056 001       **     LHI 001
50 144  104 130 006          JMP LT1

50 147  074 002        STC1, CPI 002
50 151  110 164 050          JFZ STC2
50 154  106 267 050          CAL SSTRCP
50 157  056 001       **     LHI 001
50 161  104 145 006          JMP EQ1

50 164  074 003        STC2, CPI 003
50 166  110 201 050          JFZ STC3
50 171  106 267 050          CAL SSTRCP
50 174  056 001       **     LHI 001
50 176  104 162 006          JMP GT1

50 201  074 004        STC3, CPI 004
50 203  110 216 050          JFZ STC4
50 206  106 267 050          CAL SSTRCP
50 211  056 001       **     LHI 001
50 213  104 202 006          JMP LE1

50 216  074 005        STC4, CPI 005
50 220  110 233 050          JFZ STC5
50 223  106 267 050          CAL SSTRCP
50 226  056 001       **     LHI 001
50 230  104 222 006          JMP GE1

50 233  106 267 050        STC5, CAL SSTRCP
50 236  056 001       **     LHI 001
50 240  104 237 006          JMP NE1

50 243  317           SELOOP, LBM
50 244  010                  INB
50 245  371                  LMB
50 246  011                  DCB
50 247  066 377              LLI 377
50 251  307                  LAM
50 252  271                  CPB
50 253  007                  RET

50 254  317           S2LOOP, LBM
50 255  010                  INB
50 256  371                  LMB
50 257  011                  DCB
50 260  066 372              LLI 372
50 262  307                  LAM
50 263  024 001              SUI 001
50 265  271                  CPB
50 266  007                  RET

50 267  106 317 022   SSTRCP, CAL SAVEHL
50 272  307                  LAM
50 273  106 356 022          CAL SWITCH
50 276  317                  LBM
```

8080

```
50 130  036 020              LEI 020
50 132  376 001              CPI 001
50 134  302 147 050          JFZ STC1
50 137  315 267 050          CAL SSTRCP
50 142  046 001       **     LHI 001
50 144  303 130 006          JMP LT1

50 147  376 002        STC1, CPI 002
50 151  302 164 050          JFZ STC2
50 154  315 267 050          CAL SSTRCP
50 157  046 001       **     LHI 001
50 161  303 145 006          JMP EQ1

50 164  376 003        STC2, CPI 003
50 166  302 201 050          JFZ STC3
50 171  315 267 050          CAL SSTRCP
50 174  046 001       **     LHI 001
50 176  303 162 006          JMP GT1

50 201  376 004        STC3, CPI 004
50 203  302 216 050          JFZ STC4
50 206  315 267 050          CAL SSTRCP
50 211  046 001       **     LHI 001
50 213  303 202 006          JMP LE1

50 216  376 005        STC4, CPI 005
50 220  302 233 050          JFZ STC5
50 223  315 267 050          CAL SSTRCP
50 226  046 001       **     LHI 001
50 230  303 222 006          JMP GE1

50 233  315 267 050        STC5, CAL SSTRCP
50 236  046 001       **     LHI 001
50 240  303 237 006          JMP NE1

50 243  106           SELOOP, LBM
50 244  004                  INB
50 245  160                  LMB
50 246  005                  DCB
50 247  056 377              LLI 377
50 251  176                  LAM
50 252  270                  CPB
50 253  311                  RET

50 254  106           S2LOOP, LBM
50 255  004                  INB
50 256  160                  LMB
50 257  005                  DCB
50 260  056 372              LLI 372
50 262  176                  LAM
50 263  326 001              SUI 001
50 265  270                  CPB
50 266  311                  RET

50 267  315 317 022   SSTRCP, CAL SAVEHL
50 272  176                  LAM
50 273  315 356 022          CAL SWITCH
50 276  106                  LBM
```

8008

```
50 277  106 356 022              CAL SWITCH
50 302  271                      CPB
50 303  150 343 050              JTZ SSTRZ
50 306  120 312 050              JFS SSTRCL
50 311  310                      LBA

50 312  106 377 002     SSTRCL,  CAL ADV
50 315  307                      LAM
50 316  106 356 022              CAL SWITCH
50 321  106 377 002              CAL ADV

50 324  277             SSTRCE,  CPM
50 325  013                      RFZ
50 326  011                      DCB
50 327  110 312 050              JFZ SSTRCL
50 332  106 337 022              CAL RESTHL
50 335  307                      LAM
50 336  106 356 022              CAL SWITCH
50 341  277                      CPM
50 342  007                      RET

50 343  240               SSTRZ, NDA
50 344  053                      RTZ
50 345  104 312 050             JMP SSTRCL

50 350  056 045      $$  CONCAT,  LHI 045
50 352  066 002                  LLI 002
50 354  307                      LAM
50 355  240                      NDA
50 356  110 363 050              JFZ CONCT0
50 361  076 001                  LMI 001

50 363  066 003          CONCT0, LLI 003
50 365  307                      LAM
50 366  240                      NDA
50 367  150 111 051              JTZ CONSA1
50 372  106 255 002              CAL CLESYM
50 375  106 123 051              CAL SLOOK
51 000  066 005                  LLI 005
51 002  056 045      $$          LHI 045
51 004  337                      LDM
51 005  060                      INL
51 006  347                      LEM
51 007  353                      LHD
51 010  364                      LLE
51 011  317                      LBM
51 012  056 045      $$          LHI 045
51 014  066 003                  LLI 003
51 016  307                      LAM
51 017  074 377                  CPI 377
51 021  110 035 051              JFZ CONCA1
51 024  066 002                  LLI 002
51 026  301                      LAB
51 027  227                      SUM
51 030  310                      LBA
51 031  010                      INB
51 032  104 040 051              JMP CONSAC
```

8080

```
50 277  315 356 022              CAL SWITCH
50 302  270                      CPB
50 303  312 343 050              JTZ SSTRZ
50 306  362 312 050              JFS SSTRCL
50 311  107                      LBA

50 312  315 377 002     SSTRCL,  CAL ADV
50 315  176                      LAM
50 316  315 356 022              CAL SWITCH
50 321  315 377 002              CAL ADV

50 324  276             SSTRCE,  CPM
50 325  300                      RFZ
50 326  005                      DCB
50 327  302 312 050              JFZ SSTRCL
50 332  315 337 022              CAL RESTHL
50 335  176                      LAM
50 336  315 356 022              CAL SWITCH
50 341  276                      CPM
50 342  311                      RET

50 343  247               SSTRZ, NDA
50 344  310                      RTZ
50 345  303 312 050             JMP SSTRCL

50 350  046 045      $$  CONCAT,  LHI 045
50 352  056 002                  LLI 002
50 354  176                      LAM
50 355  247                      NDA
50 356  302 363 050              JFZ CONCT0
50 361  066 001                  LMI 001

50 363  056 003          CONCT0, LLI 003
50 365  176                      LAM
50 366  247                      NDA
50 367  312 111 051              JTZ CONSA1
50 372  315 255 002              CAL CLESYM
50 375  315 123 051              CAL SLOOK
51 000  056 005                  LLI 005
51 002  046 045      $$          LHI 045
51 004  126                      LDM
51 005  054                      INL
51 006  136                      LEM
51 007  142                      LHD
51 010  153                      LLE
51 011  106                      LBM
51 012  046 045      $$          LHI 045
51 014  056 003                  LLI 003
51 016  176                      LAM
51 017  376 377                  CPI 377
51 021  302 035 051              JFZ CONCA1
51 024  056 002                  LLI 002
51 026  170                      LAB
51 027  226                      SUM
51 030  107                      LBA
51 031  004                      INB
51 032  303 040 051              JMP CONSAC
```

8008

```
51 035  066 003          CONCA1, LLI 003
51 037  317                      LBM

51 040  066 020          CONSAC, LLI 020
51 042  056 045  $$               LHI 045
51 044  307                      LAM
51 045  201                      ADB
51 046  074 121                  CPI 121
51 050  120 222 002              JFS BIGERR
51 053  320                      LCA
51 054  307                      LAM
51 055  372                      LMC
51 056  004 021                  ADI 021
51 060  320                      LCA
51 061  066 005                  LLI 005
51 063  337                      LDM
51 064  060                      INL
51 065  307                      LAM
51 066  066 002                  LLI 002
51 070  207                      ADM
51 071  340                      LEA
51 072  303                      LAD
51 073  014 000                  ACI 000
51 075  330                      LDA
51 076  353                      LHD
51 077  364                      LLE
51 100  036 045  $$               LDI 045
51 102  342                      LEC
51 103  010                      INB
51 104  011                      DCB
51 105  300                      LAA
51 106  112 050 012              CFZ MOVEPG

51 111  066 002          CONSA1, LLI 002
51 113  056 045  $$               LHI 045
51 115  076 001                  LMI 001
51 117  060                      INL
51 120  076 377                  LMI 377
51 122  007                      RET

51 123  066 013           SLOOK, LLI 013
51 125  056 045  $$               LHI 045
51 127  327                      LCM
51 130  066 260                  LLI 260
51 132  337                      LDM
51 133  046 000                  LEI 000
51 135  020                      INC
51 136  021                      DCC
51 137  150 220 051              JTZ SLOOK3

51 142  364          SLOOK1, LLE
51 143  353                  LHD
51 144  307                  LAM
51 145  066 000              LLI 000
51 147  056 045  $$           LHI 045
51 151  277                  CPM
51 152  110 210 051          JFZ SLOOK2
51 155  364                  LLE
51 156  353                  LHD
```

8080

```
51 035  056 003          CONCA1, LLI 003
51 037  106                      LBM

51 040  056 020          CONSAC, LLI 020
51 042  046 045  $$               LHI 045
51 044  176                      LAM
51 045  200                      ADB
51 046  376 121                  CPI 121
51 050  362 222 002              JFS BIGERR
51 053  117                      LCA
51 054  176                      LAM
51 055  161                      LMC
51 056  306 021                  ADI 021
51 060  117                      LCA
51 061  056 005                  LLI 005
51 063  126                      LDM
51 064  054                      INL
51 065  176                      LAM
51 066  056 002                  LLI 002
51 070  206                      ADM
51 071  137                      LEA
51 072  172                      LAD
51 073  316 000                  ACI 000
51 075  127                      LDA
51 076  142                      LHD
51 077  153                      LLE
51 100  026 045  $$               LDI 045
51 102  131                      LEC
51 103  004                      INB
51 104  005                      DCB
51 105  177                      LAA
51 106  304 050 012              CFZ MOVEPG

51 111  056 002          CONSA1, LLI 002
51 113  046 045  $$               LHI 045
51 115  066 001                  LMI 001
51 117  054                      INL
51 120  066 377                  LMI 377
51 122  311                      RET

51 123  056 013           SLOOK, LLI 013
51 125  046 045  $$               LHI 045
51 127  116                      LCM
51 130  056 260                  LLI 260
51 132  126                      LDM
51 133  036 000                  LEI 000
51 135  014                      INC
51 136  015                      DCC
51 137  312 220 051              JTZ SLOOK3

51 142  153          SLOOK1, LLE
51 143  142                  LHD
51 144  176                  LAM
51 145  056 000              LLI 000
51 147  046 045  $$           LHI 045
51 151  276                  CPM
51 152  302 210 051          JFZ SLOOK2
51 155  153                  LLE
51 156  142                  LHD
```

8008

```
51 157  060                    INL
51 160  307                    LAM
51 161  066 001                LLI 001
51 163  056 045        $$      LHI 045
51 165  277                    CPM
51 166  110 210 051            JFZ SLOOK2
51 171  364                    LLE
51 172  353                    LHD
51 173  060                    INL
51 174  060                    INL
51 175  337                    LDM
51 176  060                    INL
51 177  347                    LEM
51 200  066 005                LLI 005
51 202  056 045        $$      LHI 045
51 204  373                    LMD
51 205  060                    INL
51 206  374                    LME
51 207  007                    RET

51 210  304            SLOOK2, LAE
51 211  004 004                ADI 004
51 213  340                    LEA
51 214  021                    DCC
51 215  110 142 051            JFZ SLOOK1

51 220  066 013        SLOOK3, LLI 013
51 222  056 045        $$      LHI 045
51 224  317                    LBM
51 225  010                    INB
51 226  371                    LMB
51 227  301                    LAB
51 230  074 101                CPI 101
51 232  120 222 002            JFS BIGERR
51 235  066 000                LLI 000
51 237  307                    LAM
51 240  353                    LHD
51 241  364                    LLE
51 242  370                    LMA
51 243  066 001                LLI 001
51 245  056 045        $$      LHI 045
51 247  307                    LAM
51 250  353                    LHD
51 251  364                    LLE
51 252  060                    INL
51 253  370                    LMA
51 254  066 015                LLI 015
51 256  056 045        $$      LHI 045
51 260  317                    LBM
51 261  060                    INL
51 262  327                    LCM
51 263  066 005                LLI 005
51 265  371                    LMB
51 266  060                    INL
51 267  372                    LMC
51 270  353                    LHD
51 271  364                    LLE
51 272  060                    INL
51 273  060                    INL
```

8080

```
51 157  054                    INL
51 160  176                    LAM
51 161  056 001                LLI 001
51 163  046 045        $$      LHI 045
51 165  276                    CPM
51 166  302 210 051            JFZ SLOOK2
51 171  153                    LLE
51 172  142                    LHD
51 173  054                    INL
51 174  054                    INL
51 175  126                    LDM
51 176  054                    INL
51 177  136                    LEM
51 200  056 005                LLI 005
51 202  046 045        $$      LHI 045
51 204  162                    LMD
51 205  054                    INL
51 206  163                    LME
51 207  311                    RET

51 210  173            SLOOK2, LAE
51 211  306 004                ADI 004
51 213  137                    LEA
51 214  015                    DCC
51 215  302 142 051            JFZ SLOOK1

51 220  056 013        SLOOK3, LLI 013
51 222  046 045        $$      LHI 045
51 224  106                    LBM
51 225  004                    INB
51 226  160                    LMB
51 227  170                    LAB
51 230  376 101                CPI 101
51 232  362 222 002            JFS BIGERR
51 235  056 000                LLI 000
51 237  176                    LAM
51 240  142                    LHD
51 241  153                    LLE
51 242  167                    LMA
51 243  056 001                LLI 001
51 245  046 045        $$      LHI 045
51 247  176                    LAM
51 250  142                    LHD
51 251  153                    LLE
51 252  054                    INL
51 253  167                    LMA
51 254  056 015                LLI 015
51 256  046 045        $$      LHI 045
51 260  106                    LBM
51 261  054                    INL
51 262  116                    LCM
51 263  056 005                LLI 005
51 265  160                    LMB
51 266  054                    INL
51 267  161                    LMC
51 270  142                    LHD
51 271  153                    LLE
51 272  054                    INL
51 273  054                    INL
```

8008

```
51 274  371                    LMB
51 275  060                    INL
51 276  372                    LMC
51 277  302                    LAC
51 300  004 001                ADI 001
51 302  320                    LCA
51 303  301                    LAB
51 304  014 000                ACI 000
51 306  310                    LBA
51 307  066 015                LLI 015
51 311  056 045        $$      LHI 045
51 313  371                    LMB
51 314  060                    INL
51 315  372                    LMC
51 316  351                    LHB
51 317  362                    LLC
51 320  076 000                LMI 000
51 322  026 000                LCI 000
51 324  007                    RET

51 325  026 001         PARNB, LCI 001
51 327  056 026        **      LHI 026
51 331  363                    LLD
51 332  040                    INE

51 333  307            PARNB1, LAM
51 334  074 250                CPI 250
51 336  110 345 051            JFZ PARNB2
51 341  020                    INC
51 342  104 353 051            JMP PARNB3

51 345  074 251         PARNB2, CPI 251
51 347  110 353 051            JFZ PARNB3
51 352  021                    DCC

51 353  020            PARNB3, INC
51 354  021                    DCC
51 355  316                    LBL
51 356  053                    RTZ
51 357  060                    INL
51 360  306                    LAL
51 361  274                    CPE
51 362  110 333 051            JFZ PARNB1
51 365  104 104 006            JMP PARNER

51 370  066 375         EVALPC, LLI 375
51 372  056 026        **      LHI 026
51 374  076 000                LMI 000
51 376  066 227          EVALS, LLI 227
52 000  056 001        **      LHI 001
52 002  104 227 003            JMP EVALQ

52 005  074 244         SCANCP, CPI 244
52 007  150 000 046            JTZ SEVAL
52 012  074 247                CPI 247
52 014  150 000 046            JTZ SEVAL
52 017  074 242                CPI 242
52 021  150 000 046            JTZ SEVAL
```

8080

```
51 274  160                    LMB
51 275  054                    INL
51 276  161                    LMC
51 277  171                    LAC
51 300  306 001                ADI 001
51 302  117                    LCA
51 303  170                    LAB
51 304  316 000                ACI 000
51 306  107                    LBA
51 307  056 015                LLI 015
51 311  046 045        $$      LHI 045
51 313  160                    LMB
51 314  054                    INL
51 315  161                    LMC
51 316  140                    LHB
51 317  151                    LLC
51 320  066 000                LMI 000
51 322  016 000                LCI 000
51 324  311                    RET

51 325  016 001         PARNB, LCI 001
51 327  046 026        **      LHI 026
51 331  152                    LLD
51 332  034                    INE

51 333  176            PARNB1, LAM
51 334  376 250                CPI 250
51 336  302 345 051            JFZ PARNB2
51 341  014                    INC
51 342  303 353 051            JMP PARNB3

51 345  376 251         PARNB2, CPI 251
51 347  302 353 051            JFZ PARNB3
51 352  015                    DCC

51 353  014            PARNB3, INC
51 354  015                    DCC
51 355  105                    LBL
51 356  310                    RTZ
51 357  054                    INL
51 360  175                    LAL
51 361  273                    CPE
51 362  302 333 051            JFZ PARNB1
51 365  303 104 006            JMP PARNER

51 370  056 375         EVALPC, LLI 375
51 372  046 026        **      LHI 026
51 374  066 000                LMI 000
51 376  056 227          EVALS, LLI 227
52 000  046 001        **      LHI 001
52 002  303 227 003            JMP EVALQ

52 005  376 244         SCANCP, CPI 244
52 007  312 000 046            JTZ SEVAL
52 012  376 247                CPI 247
52 014  312 000 046            JTZ SEVAL
52 017  376 242                CPI 242
52 021  312 000 046            JTZ SEVAL
```

8008

```
52 024  106 042 052          CAL LTEQGT
52 027  066 176              LLI 176
52 031  317                  LBM
52 032  010                  INB
52 033  011                  DCB
52 034  110 351 003          JFZ SCANFN
52 037  104 276 004          JMP SCAN16

52 042  066 176       LTEQGT, LLI 176
52 044  076 000               LMI 000
52 046  104 100 004          JMP SCAN9

52 051  006 323        STERR, LAI 323
52 053  026 324               LCI 324
52 055  104 226 002          JMP ERROR

52 060  006 311        IQERR, LAI 311
52 062  026 321               LCI 321
52 064  104 226 002          JMP ERROR

52 070  066 375        STORP, LLI 375
52 072  056 026     **        LHI 026
52 074  307                  LAM
52 075  240                  NDA
52 076  066 201              LLI 201
52 100  056 027     **        LHI 027
52 102  150 060 010          JTZ STORP1

52 105  307            SSTOR, LAM
52 106  076 000               LMI 000
52 110  240                  NDA
52 111  110 150 052          JFZ SSTOR1
52 114  066 120              LLI 120
52 116  056 026     **        LHI 026
52 120  307                  LAM
52 121  074 002              CPI 002
52 123  110 051 052          JFZ STERR
52 126  206                  ADL
52 127  360                  LLA
52 130  307                  LAM
52 131  074 244              CPI 244
52 133  110 051 052          JFZ STERR
52 136  061                  DCL
52 137  307                  LAM
52 140  066 261              LLI 261
52 142  056 045     $$        LHI 045
52 144  370                  LMA
52 145  060                  INL
52 146  076 001              LMI 001

52 150  066 261       SSTOR1, LLI 261
52 152  056 045     $$        LHI 045
52 154  337                  LDM
52 155  060                  INL
52 156  347                  LEM
52 157  066 000              LLI 000
52 161  373                  LMD
52 162  060                  INL
52 163  374                  LME
```

8080

```
52 024  315 042 052          CAL LTEQGT
52 027  056 176              LLI 176
52 031  106                  LBM
52 032  004                  INB
52 033  005                  DCB
52 034  302 351 003          JFZ SCANFN
52 037  303 276 004          JMP SCAN16

52 042  056 176       LTEQGT, LLI 176
52 044  066 000               LMI 000
52 046  303 100 004          JMP SCAN9

52 051  076 323        STERR, LAI 323
52 053  016 324               LCI 324
52 055  303 226 002          JMP ERROR

52 060  076 311        IQERR, LAI 311
52 062  016 321               LCI 321
52 064  303 226 002          JMP ERROR

52 070  056 375        STORP, LLI 375
52 072  046 026     **        LHI 026
52 074  176                  LAM
52 075  247                  NDA
52 076  056 201              LLI 201
52 100  046 027     **        LHI 027
52 102  312 060 010          JTZ STORP1

52 105  176            SSTOR, LAM
52 106  066 000               LMI 000
52 110  247                  NDA
52 111  302 150 052          JFZ SSTOR1
52 114  056 120              LLI 120
52 116  046 026     **        LHI 026
52 120  176                  LAM
52 121  376 002              CPI 002
52 123  302 051 052          JFZ STERR
52 126  205                  ADL
52 127  157                  LLA
52 130  176                  LAM
52 131  376 244              CPI 244
52 133  302 051 052          JFZ STERR
52 136  055                  DCL
52 137  176                  LAM
52 140  056 261              LLI 261
52 142  046 045     $$        LHI 045
52 144  167                  LMA
52 145  054                  INL
52 146  066 001              LMI 001

52 150  056 261       SSTOR1, LLI 261
52 152  046 045     $$        LHI 045
52 154  126                  LDM
52 155  054                  INL
52 156  136                  LEM
52 157  056 000              LLI 000
52 161  162                  LMD
52 162  054                  INL
52 163  163                  LME
```

8008

```
52 164  106 123 051          CAL SLOOK
52 167  302                  LAC
52 170  240                  NDA
52 171  110 214 052          JFZ SSTOR8
52 174  066 016              LLI 016
52 176  056 045      $$      LHI 045
52 200  307                  LAM
52 201  024 001              SUI 001
52 203  370                  LMA
52 204  061                  DCL
52 205  307                  LAM
52 206  034 000              SBI 000
52 210  370                  LMA
52 211  104 035 053          JMP SSTOR3

52 214  066 005      SSTOR8, LLI 005
52 216  056 045      $$      LHI 045
52 220  337                  LDM
52 221  060                  INL
52 222  347                  LEM
52 223  353                  LHD
52 224  364                  LLE
52 225  327                  LCM
52 226  066 007              LLI 007
52 230  056 045      $$      LHI 045
52 232  372                  LMC
52 233  020                  INC
52 234  066 016              LLI 016
52 236  307                  LAM
52 237  222                  SUC
52 240  370                  LMA
52 241  061                  DCL
52 242  307                  LAM
52 243  034 000              SBI 000
52 245  370                  LMA
52 246  304      SSTOR2, LAE
52 247  202                  ADC
52 250  360                  LLA
52 251  303                  LAD
52 252  014 000              ACI 000
52 254  350                  LHA
52 255  317                  LBM
52 256  364                  LLE
52 257  353                  LHD
52 260  371                  LMB
52 261  066 015              LLI 015
52 263  056 045      $$      LHI 045
52 265  303                  LAD
52 266  277                  CPM
52 267  110 300 052          JFZ SSTOR9
52 272  060                  INL
52 273  304                  LAE
52 274  277                  CPM
52 275  150 306 052          JTZ SSTOR0

52 300  106 064 013  SSTOR9, CAL ADVDE
52 303  104 246 052          JMP SSTOR2
```

8080

```
52 164  315 123 051          CAL SLOOK
52 167  171                  LAC
52 170  247                  NDA
52 171  302 214 052          JFZ SSTOR8
52 174  056 016              LLI 016
52 176  046 045      $$      LHI 045
52 200  176                  LAM
52 201  326 001              SUI 001
52 203  167                  LMA
52 204  055                  DCL
52 205  176                  LAM
52 206  336 000              SBI 000
52 210  167                  LMA
52 211  303 035 053          JMP SSTOR3

52 214  056 005      SSTOR8, LLI 005
52 216  046 045      $$      LHI 045
52 220  126                  LDM
52 221  054                  INL
52 222  136                  LEM
52 223  142                  LHD
52 224  153                  LLE
52 225  116                  LCM
52 226  056 007              LLI 007
52 230  046 045      $$      LHI 045
52 232  161                  LMC
52 233  014                  INC
52 234  056 016              LLI 016
52 236  176                  LAM
52 237  221                  SUC
52 240  167                  LMA
52 241  055                  DCL
52 242  176                  LAM
52 243  336 000              SBI 000
52 245  167                  LMA
52 246  173      SSTOR2, LAE
52 247  201                  ADC
52 250  157                  LLA
52 251  172                  LAD
52 252  316 000              ACI 000
52 254  147                  LHA
52 255  106                  LBM
52 256  153                  LLE
52 257  142                  LHD
52 260  160                  LMB
52 261  056 015              LLI 015
52 263  046 045      $$      LHI 045
52 265  172                  LAD
52 266  276                  CPM
52 267  302 300 052          JFZ SSTOR9
52 272  054                  INL
52 273  173                  LAE
52 274  276                  CPM
52 275  312 306 052          JTZ SSTOR0

52 300  315 064 013  SSTOR9, CAL ADVDE
52 303  303 246 052          JMP SSTOR2
```

8008

```
52 306  066 005          SSTOR0, LLI 005
52 310  056 045    $$            LHI 045
52 312  337                      LDM
52 313  060                      INL
52 314  347                      LEM
52 315  066 013                  LLI 013
52 317  317                      LBM
52 320  066 260                  LLI 260
52 322  357                      LHM
52 323  066 002                  LLI 002

52 325  307              SSTOR4, LAM
52 326  273                      CPD
52 327  150 340 052              JTZ SSTOR5
52 332  160 025 053              JTS SSTOR7
52 335  104 014 053              JMP SSTOR6

52 340  060              SSTOR5, INL
52 341  307                      LAM
52 342  061                      DCL
52 343  274                      CPE
52 344  140 025 053              JTC SSTOR7
52 347  110 014 053              JFZ SSTOR6
52 352  306                      LAL
52 353  066 263                  LLI 263
52 355  056 045    $$            LHI 045
52 357  371                      LMB
52 360  060                      INL
52 361  372                      LMC
52 362  066 015                  LLI 015
52 364  317                      LBM
52 365  060                      INL
52 366  327                      LCM
52 367  066 260                  LLI 260
52 371  357                      LHM
52 372  360                      LLA
52 373  371                      LMB
52 374  060                      INL
52 375  372                      LMC
52 376  066 263                  LLI 263
53 000  056 045    $$            LHI 045
53 002  317                      LBM
53 003  060                      INL
53 004  327                      LCM
53 005  066 260                  LLI 260
53 007  357                      LHM
53 010  360                      LLA
53 011  104 025 053              JMP SSTOR7

53 014  060              SSTOR6, INL
53 015  307                      LAM
53 016  222                      SUC
53 017  370                      LMA
53 020  061                      DCL
53 021  307                      LAM
53 022  034 000                  SBI 000
53 024  370                      LMA
```

8080

```
52 306  056 005          SSTOR0, LLI 005
52 310  046 045    $$            LHI 045
52 312  126                      LDM
52 313  054                      INL
52 314  136                      LEM
52 315  056 013                  LLI 013
52 317  106                      LBM
52 320  056 260                  LLI 260
52 322  146                      LHM
52 323  056 002                  LLI 002

52 325  176              SSTOR4, LAM
52 326  272                      CPD
52 327  312 340 052              JTZ SSTOR5
52 332  372 025 053              JTS SSTOR7
52 335  303 014 053              JMP SSTOR6

52 340  054              SSTOR5, INL
52 341  176                      LAM
52 342  055                      DCL
52 343  273                      CPE
52 344  332 025 053              JTC SSTOR7
52 347  302 014 053              JFZ SSTOR6
52 352  175                      LAL
52 353  056 263                  LLI 263
52 355  046 045    $$            LHI 045
52 357  160                      LMB
52 360  054                      INL
52 361  161                      LMC
52 362  056 015                  LLI 015
52 364  106                      LBM
52 365  054                      INL
52 366  116                      LCM
52 367  056 260                  LLI 260
52 371  146                      LHM
52 372  157                      LLA
52 373  160                      LMB
52 374  054                      INL
52 375  161                      LMC
52 376  056 263                  LLI 263
53 000  046 045    $$            LHI 045
53 002  106                      LBM
53 003  054                      INL
53 004  116                      LCM
53 005  056 260                  LLI 260
53 007  146                      LHM
53 010  157                      LLA
53 011  303 025 053              JMP SSTOR7

53 014  054              SSTOR6, INL
53 015  176                      LAM
53 016  221                      SUC
53 017  167                      LMA
53 020  055                      DCL
53 021  176                      LAM
53 022  336 000                  SBI 000
53 024  167                      LMA
```

8008

```
53 025  306              SSTOR7,  LAL
53 026  004 004                   ADI 004
53 030  360                       LLA
53 031  011                       DCB
53 032  110 325 052               JFZ SSTOR4

53 035  066 015          SSTOR3,  LLI 015
53 037  056 045  $$               LHI 045
53 041  337                       LDM
53 042  060                       INL
53 043  347                       LEM
53 044  066 020                   LLI 020
53 046  307                       LAM
53 047  004 001                   ADI 001
53 051  204                       ADE
53 052  340                       LEA
53 053  303                       LAD
53 054  014 000                   ACI 000
53 056  330                       LDA
53 057  074 045  $$               CPI 045
53 061  150 222 002               JTZ BIGERR
53 064  066 015                   LLI 015
53 066  337                       LDM
53 067  370                       LMA
53 070  060                       INL
53 071  307                       LAM
53 072  374                       LME
53 073  340                       LEA
53 074  066 020                   LLI 020
53 076  106 046 012               CAL MOVEC
53 101  066 015                   LLI 015
53 103  056 045  $$               LHI 045
53 105  337                       LDM
53 106  060                       INL
53 107  347                       LEM
53 110  353                       LHD
53 111  364                       LLE
53 112  076 000                   LMI 000
53 114  066 375                   LLI 375
53 116  056 026  **               LHI 026
53 120  076 000                   LMI 000
53 122  007                       RET

53 124  066 203          PPRINT,  LLI 203
53 126  106 003 003               CAL LOOP
53 131  110 002 014               JFZ PRINT2
53 134  104 043 014               JMP PRINT3

53 137  066 203          PPRIN7,  LLI 203
53 141  337                       LDM
53 142  030                       IND
53 143  066 000                   LLI 000
53 145  347                       LEM
53 146  106 325 051               CAL PARNB
53 151  066 203                   LLI 203
53 153  371                       LMB
53 154  104 124 053               JMP PPRINT
```

8080

```
53 025  175              SSTOR7,  LAL
53 026  306 004                   ADI 004
53 030  157                       LLA
53 031  005                       DCB
53 032  302 325 052               JFZ SSTOR4

53 035  056 015          SSTOR3,  LLI 015
53 037  046 045  $$               LHI 045
53 041  126                       LDM
53 042  054                       INL
53 043  136                       LEM
53 044  056 020                   LLI 020
53 046  176                       LAM
53 047  306 001                   ADI 001
53 051  203                       ADE
53 052  137                       LEA
53 053  172                       LAD
53 054  316 000                   ACI 000
53 056  127                       LDA
53 057  376 045  $$               CPI 045
53 061  312 222 002               JTZ BIGERR
53 064  056 015                   LLI 015
53 066  126                       LDM
53 067  167                       LMA
53 070  054                       INL
53 071  176                       LAM
53 072  163                       LME
53 073  137                       LEA
53 074  056 020                   LLI 020
53 076  315 046 012               CAL MOVEC
53 101  056 015                   LLI 015
53 103  046 045  $$               LHI 045
53 105  126                       LDM
53 106  054                       INL
53 107  136                       LEM
53 110  142                       LHD
53 111  153                       LLE
53 112  066 000                   LMI 000
53 114  056 375  **               LLI 375
53 116  046 026                   LHI 026
53 120  066 000                   LMI 000
53 122  311                       RET

53 124  056 203          PPRINT,  LLI 203
53 126  315 003 003               CAL LOOP
53 131  302 002 014               JFZ PRINT2
53 134  303 043 014               JMP PRINT3

53 137  056 203          PPRIN7,  LLI 203
53 141  126                       LDM
53 142  024                       IND
53 143  056 000                   LLI 000
53 145  136                       LEM
53 146  315 325 051               CAL PARNB
53 151  056 203                   LLI 203
53 153  160                       LMB
53 154  303 124 053               JMP PPRINT
```

8008

```
53 157  066 375          POUTSF, LLI 375
53 161  056 026     **           LHI 026
53 163  307                      LAM
53 164  240                      NDA
53 165  150 314 014              JTZ PFPOUT
53 170  066 020                  LLI 020
53 172  056 045     $$           LHI 045
53 174  104 121 003              JMP TEXTC

53 177  066 120          ARRAYP, LLI 120
53 201  307                      LAM
53 202  206                      ADL
53 203  360                      LLA
53 204  307                      LAM
53 205  074 244                  CPI 244
53 207  150 221 053              JTZ SARRAY
53 212  066 207                  LLI 207
53 214  076 000                  LMI 000
53 216  104 240 055              JMP ARRAY6

53 221  061              SARRAY, DCL
53 222  307                      LAM
53 223  066 261                  LLI 261
53 225  056 045     $$           LHI 045
53 227  370                      LMA
53 230  106 224 003              CAL EVAL
53 233  106 000 020              CAL FPFIX
53 236  066 124                  LLI 124
53 240  056 001     **           LHI 001
53 242  307                      LAM
53 243  066 262                  LLI 262
53 245  056 045     $$           LHI 045
53 247  370                      LMA
53 250  066 375                  LLI 375
53 252  056 026     **           LHI 026
53 254  076 001                  LMI 001
53 256  056 027     **           LHI 027
53 260  066 201                  LLI 201
53 262  076 001                  LMI 001
53 264  007                      RET

53 265  066 260          SCRPCH, LLI 260
53 267  056 045     $$           LHI 045
53 271  317                      LBM
53 272  010                      INB
53 273  066 013                  LLI 013
53 275  076 000                  LMI 000
53 277  066 015                  LLI 015
53 301  371                      LMB
53 302  060                      INL
53 303  076 000                  LMI 000
53 305  351                      LHB
53 306  066 000                  LLI 000
53 310  076 000                  LMI 000
53 312  104 266 010              JMP EXEC

53 315  106 255 002      PINPUT, CAL CLESYM
53 320  066 375                  LLI 375
```

8080

```
53 157  056 375          POUTSF, LLI 375
53 161  046 026     **           LHI 026
53 163  176                      LAM
53 164  247                      NDA
53 165  312 314 014              JTZ PFPOUT
53 170  056 020                  LLI 020
53 172  046 045     $$           LHI 045
53 174  303 121 003              JMP TEXTC

53 177  056 120          ARRAYP, LLI 120
53 201  176                      LAM
53 202  205                      ADL
53 203  157                      LLA
53 204  176                      LAM
53 205  376 244                  CPI 244
53 207  312 221 053              JTZ SARRAY
53 212  056 207                  LLI 207
53 214  066 000                  LMI 000
53 216  303 240 055              JMP ARRAY6

53 221  055              SARRAY, DCL
53 222  176                      LAM
53 223  056 261                  LLI 261
53 225  046 045     $$           LHI 045
53 227  167                      LMA
53 230  315 224 003              CAL EVAL
53 233  315 000 020              CAL FPFIX
53 236  056 124                  LLI 124
53 240  046 001     **           LHI 001
53 242  176                      LAM
53 243  056 262                  LLI 262
53 245  046 045     $$           LHI 045
53 247  167                      LMA
53 250  056 375                  LLI 375
53 252  046 026     **           LHI 026
53 254  066 001                  LMI 001
53 256  046 027     **           LHI 027
53 260  056 201                  LLI 201
53 262  066 001                  LMI 001
53 264  311                      RET

53 265  056 260          SCRPCH, LLI 260
53 267  046 045     $$           LHI 045
53 271  106                      LBM
53 272  004                      INB
53 273  056 013                  LLI 013
53 275  066 000                  LMI 000
53 277  056 015                  LLI 015
53 301  160                      LMB
53 302  054                      INL
53 303  066 000                  LMI 000
53 305  140                      LHB
53 306  056 000                  LLI 000
53 310  066 000                  LMI 000
53 312  303 266 010              JMP EXEC

53 315  315 255 002      PINPUT, CAL CLESYM
53 320  056 375                  LLI 375
```

8008

```
53 322  076 000                LMI 000
53 324  007                    RET

53 325  066 375          INPUTS, LLI 375
53 327  056 026     **         LHI 026
53 331  076 001                LMI 001
53 333  006 277                LAI 277
53 335  106 202 003            CAL ECHO
53 340  066 020                LLI 020
53 342  056 045     $$         LHI 045
53 344  104 014 003            JMP STRIN

53 347  106 317 022    INSTRP, CAL SAVEHL
53 352  106 356 022            CAL SWITCH
53 355  307                    LAM
53 356  074 247                CPI 247
53 360  150 370 053            JTZ INSTRQ
53 363  074 242                CPI 242
53 365  110 356 022            JFZ SWITCH

53 370  060            INSTRQ, INL
53 371  277                    CPM
53 372  150 356 022            JTZ SWITCH
53 375  315                    LBH
53 376  326                    LCL
53 377  066 000                LLI 000
54 001  056 026     **         LHI 026
54 003  307                    LAM
54 004  274                    CPE
54 005  351                    LHB
54 006  362                    LLC
54 007  150 060 052            JTZ IQERR
54 012  104 370 053            JMP INSTRQ

54 015  066 012        SFAPCH, LLI 012
54 017  056 045     $$         LHI 045
54 021  076 001                LMI 001
54 023  066 360                LLI 360

54 025  036 026     **  SFAPC1, LDI 026
54 027  046 120                LEI 120
54 031  106 332 002            CAL STRCP
54 034  150 077 054            JTZ SFAPC3
54 037  364                    LLE
54 040  353                    LHD

54 041  060            SFAPC2, INL
54 042  307                    LAM
54 043  044 300                NDI 300
54 045  110 041 054            JFZ SFAPC2
54 050  335                    LDH
54 051  346                    LEL
54 052  066 012                LLI 012
54 054  056 045     $$         LHI 045
54 056  317                    LBM
54 057  010                    INB
54 060  371                    LMB
54 061  353                    LHD
```

8080

```
53 322  066 000                LMI 000
53 324  311                    RET

53 325  056 375          INPUTS, LLI 375
53 327  046 026     **         LHI 026
53 331  066 001                LMI 001
53 333  076 277                LAI 277
53 335  315 202 003            CAL ECHO
53 340  056 020                LLI 020
53 342  046 045     $$         LHI 045
53 344  303 014 003            JMP STRIN

53 347  315 317 022    INSTRP, CAL SAVEHL
53 352  315 356 022            CAL SWITCH
53 355  176                    LAM
53 356  376 247                CPI 247
53 360  312 370 053            JTZ INSTRQ
53 363  376 242                CPI 242
53 365  302 356 022            JFZ SWITCH

53 370  054            INSTRQ, INL
53 371  276                    CPM
53 372  312 356 022            JTZ SWITCH
53 375  104                    LBH
53 376  115                    LCL
53 377  056 000                LLI 000
54 001  046 026     **         LHI 026
54 003  176                    LAM
54 004  273                    CPE
54 005  140                    LHB
54 006  151                    LLC
54 007  312 060 052            JTZ IQERR
54 012  303 370 053            JMP INSTRQ

54 015  056 012        SFAPCH, LLI 012
54 017  046 045     $$         LHI 045
54 021  066 001                LMI 001
54 023  056 360                LLI 360

54 025  026 026     **  SFAPC1, LDI 026
54 027  036 120                LEI 120
54 031  315 332 002            CAL STRCP
54 034  312 077 054            JTZ SFAPC3
54 037  153                    LLE
54 040  142                    LHD

54 041  054            SFAPC2, INL
54 042  176                    LAM
54 043  346 300                NDI 300
54 045  302 041 054            JFZ SFAPC2
54 050  124                    LDH
54 051  135                    LEL
54 052  056 012                LLI 012
54 054  046 045     $$         LHI 045
54 056  106                    LBM
54 057  004                    INB
54 060  160                    LMB
54 061  142                    LHD
```

8008

```
54 062  364                    LLE
54 063  301                    LAB
54 064  074 004                CPI 004
54 066  110 025 054            JFZ SFAPC1
54 071  106 100 007            CAL FUNARR
54 074  104 024 004            JMP SCAN61

54 077  066 276        SFAPC3, LLI 276
54 101  056 026        **      LHI 026
54 103  317                    LBM
54 104  060                    INL
54 105  327                    LCM
54 106  066 010                LLI 010
54 110  056 045        $$      LHI 045
54 112  371                    LMB
54 113  060                    INL
54 114  372                    LMC
54 115  066 200                LLI 200
54 117  056 026        **      LHI 026
54 121  317                    LBM
54 122  010                    INB
54 123  066 276                LLI 276
54 125  371                    LMB
54 126  060                    INL
54 127  331                    LDB
54 130  347                    LEM
54 131  106 325 051            CAL PARNB
54 134  066 277                LLI 277
54 136  011                    DCB
54 137  371                    LMB
54 140  106 224 003            CAL EVAL
54 143  066 375                LLI 375
54 145  056 026        **      LHI 026
54 147  307                    LAM
54 150  240                    NDA
54 151  150 051 052            JTZ STERR
54 154  066 010                LLI 010
54 156  056 045        $$      LHI 045
54 160  337                    LDM
54 161  060                    INL
54 162  347                    LEM
54 163  066 277                LLI 277
54 165  056 026        **      LHI 026
54 167  317                    LBM
54 170  010                    INB
54 171  066 200                LLI 200
54 173  371                    LMB
54 174  066 276                LLI 276
54 176  373                    LMD
54 177  060                    INL
54 200  374                    LME
54 201  066 012                LLI 012
54 203  056 045        $$      LHI 045
54 205  307                    LAM
54 206  074 003                CPI 003
54 210  110 223 054            JFZ SFAPC7
54 213  066 020                LLI 020
```

8080

```
54 062  153                    LLE
54 063  170                    LAB
54 064  376 004                CPI 004
54 066  302 025 054            JFZ SFAPC1
54 071  315 100 007            CAL FUNARR
54 074  303 024 004            JMP SCAN61

54 077  056 276        SFAPC3, LLI 276
54 101  046 026        **      LHI 026
54 103  106                    LBM
54 104  054                    INL
54 105  116                    LCM
54 106  056 010                LLI 010
54 110  046 045        $$      LHI 045
54 112  160                    LMB
54 113  054                    INL
54 114  161                    LMC
54 115  056 200                LLI 200
54 117  046 026        **      LHI 026
54 121  106                    LBM
54 122  004                    INB
54 123  056 276                LLI 276
54 125  160                    LMB
54 126  054                    INL
54 127  120                    LDB
54 130  136                    LEM
54 131  315 325 051            CAL PARNB
54 134  056 277                LLI 277
54 136  005                    DCB
54 137  160                    LMB
54 140  315 224 003            CAL EVAL
54 143  056 375                LLI 375
54 145  046 026        **      LHI 026
54 147  176                    LAM
54 150  247                    NDA
54 151  312 051 052            JTZ STERR
54 154  056 010                LLI 010
54 156  046 045        $$      LHI 045
54 160  126                    LDM
54 161  054                    INL
54 162  136                    LEM
54 163  056 277                LLI 277
54 165  046 026        **      LHI 026
54 167  106                    LBM
54 170  004                    INB
54 171  056 200                LLI 200
54 173  160                    LMB
54 174  056 276                LLI 276
54 176  162                    LMD
54 177  054                    INL
54 200  163                    LME
54 201  056 012                LLI 012
54 203  046 045        $$      LHI 045
54 205  176                    LAM
54 206  376 003                CPI 003
54 210  302 223 054            JFZ SFAPC7
54 213  056 020                LLI 020
```

8008

```
54 215  106 044 023              CAL DINPUT
54 220  104 262 054              JMP SFAPC8

54 223  106 157 017     SFAPC7,  CAL FP0
54 226  066 012                  LLI 012
54 230  056 045      $$          LHI 045
54 232  307                      LAM
54 233  074 001                  CPI 001
54 235  066 020                  LLI 020
54 237  056 045      $$          LHI 045
54 241  110 250 054              JFZ SFAPC4
54 244  317                      LBM
54 245  104 252 054              JMP SFAPC5

54 250  060             SFAPC4,  INL
54 251  317                      LBM

54 252  066 124         SFAPC5,  LLI 124
54 254  056 001      **          LHI 001
54 256  371                      LMB
54 257  106 064 020              CAL FPFLT

54 262  066 375         SFAPC8,  LLI 375
54 264  056 026      **          LHI 026
54 266  076 000                  LMI 000
54 270  066 227                  LLI 227
54 272  056 001      **          LHI 001
54 274  076 230                  LMI 230
54 276  104 301 004              JMP SCAN10
```

8080

```
54 215  315 044 023              CAL DINPUT
54 220  303 262 054              JMP SFAPC8

54 223  315 157 017     SFAPC7,  CAL FP0
54 226  056 012                  LLI 012
54 230  046 045      $$          LHI 045
54 232  176                      LAM
54 233  376 001                  CPI 001
54 235  056 020                  LLI 020
54 237  046 045      $$          LHI 045
54 241  302 250 054              JFZ SFAPC4
54 244  106                      LBM
54 245  303 252 054              JMP SFAPC5

54 250  054             SFAPC4,  INL
54 251  106                      LBM

54 252  056 124         SFAPC5,  LLI 124
54 254  046 001      **          LHI 001
54 256  160                      LMB
54 257  315 064 020              CAL FPFLT

54 262  056 375         SFAPC8,  LLI 375
54 264  046 026      **          LHI 026
54 266  066 000                  LMI 000
54 270  056 227                  LLI 227
54 272  046 001      **          LHI 001
54 274  066 230                  LMI 230
54 276  303 301 004              JMP SCAN10
```

EXAMPLE PROGRAMS OF STRINGS SUPPLEMENT OPERATION

The two programs shown on the next several pages illustrate some of the new capabilities when the STRINGS SUPPLEMENT routines are installed. The first example shows how a portion of a character string may be manipulated. The second example illustrates a simple (first character) alphabetical sort operation.

SCELBAL users should have no difficulty applying the new string handling capabilities. The material presented in the first several pages of this publication contains information with which the programmer should become well acquainted. Don't forget to take advantage of the new ASC, LEN, and VAL functions when the opportunity arises!

```
10 PRINT 'INPUT STRING';
15 INPUT I$
30 PRINT 'SEARCH STRING';
40 INPUT S$
50 PRINT 'REPLACE STRING';
55 INPUT R$
60 L=LEN(I$)
70 L1=LEN(S$)
```

```
80  FOR N=1 TO L-L1+1
90  IF I$(:N;L1)=S$ GOTO 130
100  NEXT N
110  PRINT 'NOT FOUND'
120  GOTO 10
130  I$=I$(:1;N-1)+R$+I$(:N+L1)
140  PRINT I$
150  END

RUN
INPUT STRING?THIS IS A TEST OF SCELBAL.
SEARCH STRING?SCELBAL.
REPLACE STRING?STRING SCELBAL!
THIS IS A TEST OF STRING SCELBAL!


10  PRINT 'HOW MANY';
20  INPUT N
30  FOR J=1 TO N
40  INPUT A$(J)
50  NEXT J
55  PRINT 'THE UNSORTED STRINGS ARE:'
60  GOSUB 1000
70  FOR J=1 TO N-1
80  FOR K=J+1 TO N
90  IF A$(J)  =A$(K) GOTO 130
100  T$=A$(J)
110  A$(J)=A$(K)
120  A$(K)=T$
130  NEXT K
140  NEXT J
150  PRINT 'THE SORTED STRINGS ARE:'
160  GOSUB 1000
170  END
1000  FOR J=1 TO N
1010  PRINT A$(J)
1020  NEXT J
1030  RETURN


RUN
HOW MANY?10
?TONY
?BILL
?JANET
?ALICE
?STEVE
?CHARLIE
?MARGARET
?VIRGINIA
?EUNICE
?GEORGE
```

```
THE UNSORTED STRINGS ARE:
TONY
BILL
JANET
ALICE
STEVE
CHARLIE
MARGARET
VIRGINIA
EUNICE
GEORGE
THE SORTED STRINGS ARE:
ALICE
BILL
CHARLIE
EUNICE
GEORGE
JANET
MARGARET
STEVE
TONY
VIRGINIA
```

SYMBOL TABLES

The following is a list of the labels defined for the STRINGS SUPPLEMENT routines. The list is ordered alphabetically. The second column shows the address of the label in the assembled version of the program presented in this publication. The third column presents the page number where the label occurs in the source listing presented in this publication.

ARRAYP	53 177	37
CHR	46 235	8
CONCA1	51 035	22
CONCAT	50 350	21
CONCT0	50 363	21
CONSA1	51 111	23
CONSAC	51 040	22
EVALPC	51 370	25
EVALS	51 376	25
FSEVAL	50 047	17
INPUTS	53 325	38
INSTRP	53 347	38
INSTRQ	53 370	38
IQERR	52 060	28
LTEQGT	52 042	27
PARNB	51 325	25
PARNB1	51 333	25
PARNB2	51 345	25
PARNB3	51 353	25
PINPUT	53 315	37
POUTSF	53 157	35
PPRIN7	53 137	35
PPRINT	53 124	35
SARRAY	53 221	37
SCANCP	52 005	27
SCRPCH	53 265	37
SELOOP	50 243	19
SEVA10	47 124	11
SEVA11	47 134	13
SEVA12	47 247	13
SEVA13	47 326	15
SEVA14	47 351	15
SEVA15	50 044	15
SEVA16	46 373	10
SEVAL	46 000	6
SEVAL1	46 047	6
SEVAL2	46 073	6
SEVAL3	46 104	6

SEVAL4	46 143	8
SEVAL5	46 162	8
SEVAL6	46 331	10
SEVAL7	46 376	10
SEVAL8	47 047	11
SEVAL9	47 057	11
SFAPC1	54 025	38
SFAPC2	54 041	38
SFAPC3	54 077	40
SFACP4	54 250	41
SFAPC5	54 252	41
SFAPC7	54 223	41
SFAPC8	54 262	41
SFAPCH	54 015	38
SLOOK	51 123	23
SLOOK1	51 142	23
SLOOK2	51 210	27
SLOOK3	51 220	27
SSTOR	52 105	30
SSTOR0	52 306	32
SSTOR1	52 150	30
SSTOR2	52 246	32
SSTOR3	53 035	34
SSTOR4	52 325	32
SSTOR5	52 340	33
SSTOR6	53 014	33
SSTOR7	53 025	33
SSTOR8	52 214	30
SSTOR9	52 300	32
SSTRCE	50 324	21
SSTRCL	50 312	21
SSTRCP	50 267	21
SSTRZ	50 343	21
STC1	50 147	19
STC2	50 164	19
STC3	50 201	19
STC4	50 216	19
STC5	50 233	19
STERR	52 051	28
STORP	52 070	28
S2LOOP	50 254	19

The following is a list of the labels referred to by the STRINGS SUPPLEMENT that are in the original SCELBAL publication. The list is arranged alphabetically. The second column shows the address of the label in the original assembled version of SCELBAL. The third column indicates the chapter and page where the label appeared in the source listing section of the book.

ADV	02 377	5-8
ADVDE	13 064	4-12
ARRAY6	55 240	9-14
BIGERR	02 222	5-6
CLESYM	02 255	5-7
CONCT1	02 314	5-7
CONCTS	02 310	5-7
DINPUT	23 044	10-19
ECHO	03 202	5-10
EQ1	06 145	8-14
ERROR	02 226	5-6
EVAL	03 224	7-2
EVALQ	03 227	7-2
EXEC	10 266	4-2
FP0	17 157	6-28
FPFIX	02 000	10-3
FPFLT	20 064	10-4
FUNARR	07 100	9-3
GE1	06 222	8-14
GETCHR	02 240	5-6
GT1	06 162	8-14
INPUTN	17 140	6-28
LE1	06 202	8-14
LOOP	03 003	5-8
LT1	06 130	8-13
MOVEC	12 046	4-9
MOVEPG	12 050	4-9
NE1	06 237	8-14
PARNER	06 104	8-13
PFPOUT	14 314	6-10
PRINT	13 345	6-5
PRINT2	14 002	6-5
PRINT3	14 043	6-8
PRINT4	14 075	6-8
PRINT5	14 114	6-8
QUOTE	14 203	6-10
RESTHL	22 337	10-17
SAVEHL	22 317	10-16
SCAN9	04 100	7-6
SCAN10	04 301	7-7
SCAN16	04 276	7-7
SCAN61	04 024	7-5
SCANFN	03 351	7-5
STORP1	10 060	6-49
STRCP	02 332	5-7
STRIN	03 014	5-8
SWITCH	22 356	10-17
TEXTC	03 121	5-9