Bit 7: Trace sequence of calls to EXECSEM and literals passed during graph generation.

Bit 8: After each parse rule has modified the graph, output it anew.

Bit 9: Trace graph when generation is complete.

Bit 10: Output time used in each segment of translation.

Bit 11: Print the string representation of the right-side graph grammar as read in.

Bit 12: Trace the sequence of parse rules used.

IV. THE INTERPRETER

The graphs that the TWS generates are amenable to execution with a very trivial interpreter. This section will attempt to describe such an interpreter, even though one is not currently in existence.

The structure of the completed program graph is detailed in Pratt (12); an informal description will lay the groundwork for the interpreter design here. Basically, the program graph will consist of two main system nodes, P and CEP, with some less important stacks at the system level (E-stack, L-stack). The P node forms a top-level instruction node with the one instruction "fetch-instruction" node operating on the nodes BRANCH, Q, and CIP (current instruction pointer). CIP contains the program graph and its associated sequence of instruction nodes.

All the primitive operations, including the primitive "fetch-instruction", are defined as graph transformations which change the state of the abstract machine. The transformation definitions are included in Appendix H. A careful analysis of the primitive "fetch-instruction" reveals that it places the next instruction node as the value of node Q and increments the instruction pointer. Interpretation is then logically a two-step process: execute the instruction in node P (fetch the next instruction), then execute the instructi in node Q (execute the instruction just fetched). The above process continues until no more instructions can be fetched.

An interpreter to accomplish the above process need be no more than a set of subroutines written in GROPE which perform the primitive graph transformations needed. One main subroutine could be written as the executive which executed the "fetch-instruction", checked to see if node Q was empty, and then called the subroutine designated by the function in the instruction of node Q. The graph transforming routines themselves are quite uncomplicated; GROPE provides many flexible means of following arcs to retrieve and change the values of the α, β, etc. nodes mentioned in the transformation description.

V. EVALUATION AND DIRECTION

The TWS described in this work does not have the benefit of extensive use at this point, nor does it approach the translation problem in a manner similar enough to previous attempts to permit close comparison. The system is moving into a user status at the University of Texas fairly rapidly, however, so concrete results should be forthcoming.

In the meantime, several aspects of the system are notable enough to require comment. The general theme in the design of this TWS has been generality and simplicity at the expense of speed and efficiency. It was determined that an implementation of this sort should be based on the formal translation of string languages to graph representations, without the heuristics that seem to populate so many TWS's whose primary concern is an efficient translation. This TWS was designed to translate a large class of string languages into their graph representations given the appropriate pair grammar with a minimum of modification. To this end, it accomplishes its task quite well.

The usefulness of the TWS will become clear when more analysis work is done on the program graph model. If such work proves fruitful, then the TWS will surely make a sizeable contribution to the field of program modeling. Indeed, even if the specific graph models are never treated formally, an important step forward will have been made when it is shown that the TWS-generated program model can be executed. When an interpreter is available, the capability will then exist to model an implementation of a language. The virtual machine which supports a language can be graphically depicted simply and quickly through the appropriate pair grammar definition. It seems clear that the various methods of implementing a certain language feature could be changed at will using this technique.

The use of H-graphs to represent programs is showing quite a bit of promise in current correctness-proving work as well. With the capability of now creating program models which include semantic information, perhaps program verification studies can be aided by this system. Certainly the delineation of control paths into arcs connecting instruction nodes makes determination of "what happens next" in a program clearer.

Finally, the TWS may prove useful in the field of automata theory, since the translation produced is nothing more than a finite-state automaton. An interpreter may be used to drive this automaton through its various "states" in a step-by-step fashion, stopping when an "interesting" state has been entered. The state diagram of a recognizer for a particular grammar could be constructed and tested, for example

In this light, then, the road ahead is clear. The interpreter (for ALGOL 60) comes next. Once this is completed, an extensive testing program must be begun to show whether or not the entire system can effectively model the semantics of an ALGOL program. Several choices remain after that for further work. New pair grammars should be developed for other programming languages; not only will they further test the TWS-they will also test the range of applicability of pair grammars

This translator writing system will undoubtedly be followed by many more sophisticated ones as the theory of program semantics grows. They will be more efficient, more general, and will handle a wider variety of output forms than this one. They will eventually take the task of translation from the compiler writer's hands completely. This TWS may be looked upon as merely a step in the right direction--a step away from the heuristically-oriented TWS's of the past and toward the formally-defined ones of the future.

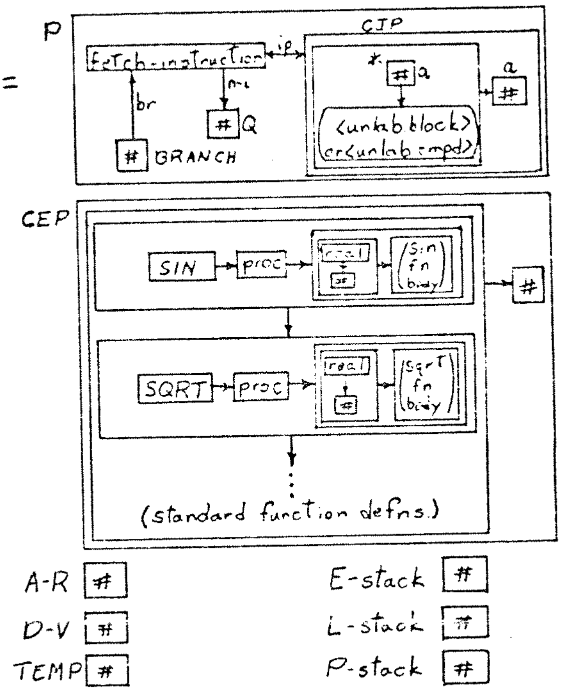
Appendix A: ALGOL 60 Pair Grammar

Formal definition of the translator

Programs and Statements

1.* <program>::=<unlab.block> or<unlab.cmpd>

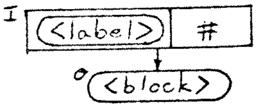
(<program>) ::=



Note: This rule defines the initial state of the abstract machine and the initial valnes of all "system" nodes.

2 <block>::=<label>:<block>

(<block>) ::=

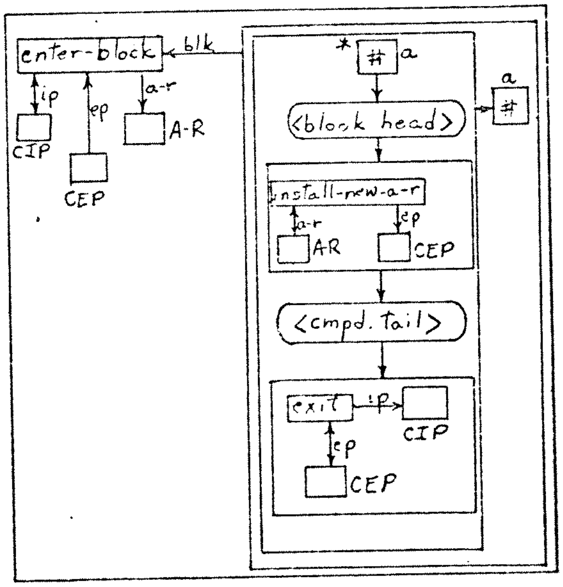


3 <block>::=<unlab.block>

(<block>) ::= (<unlab. block>)

4 <unlab.block>::=<block head>;<cmpd.tail>

(<unlab.block>)::=



5 <block head>::= begin <declar>

6 <block head>::=<block head>;<declar>

7 <declar> ::= <type decl.>

8 <declar> ::= <array decl.>

9 <declar> ::= <proc. decl.>

10 <declar> ::= <switch decl.>

11* <declar> ::= <label decl.>

12 <cmpd.stmt>::= <label>:<cmpd. stmt.>

13 <cmpd.stmt.>::= <unlab.cmpd.>

14 <unlab.cmpd.>::= begin <cmpd. tail>

15 <cmpd.tail>::=<stmt.> end

16 <cmpd.tail>::=<stmt.>;<cmpd.tail>

17 <stmt.> ::= <uncond. stmt.>

18 <stmt.> ::= <cond. stmt.>

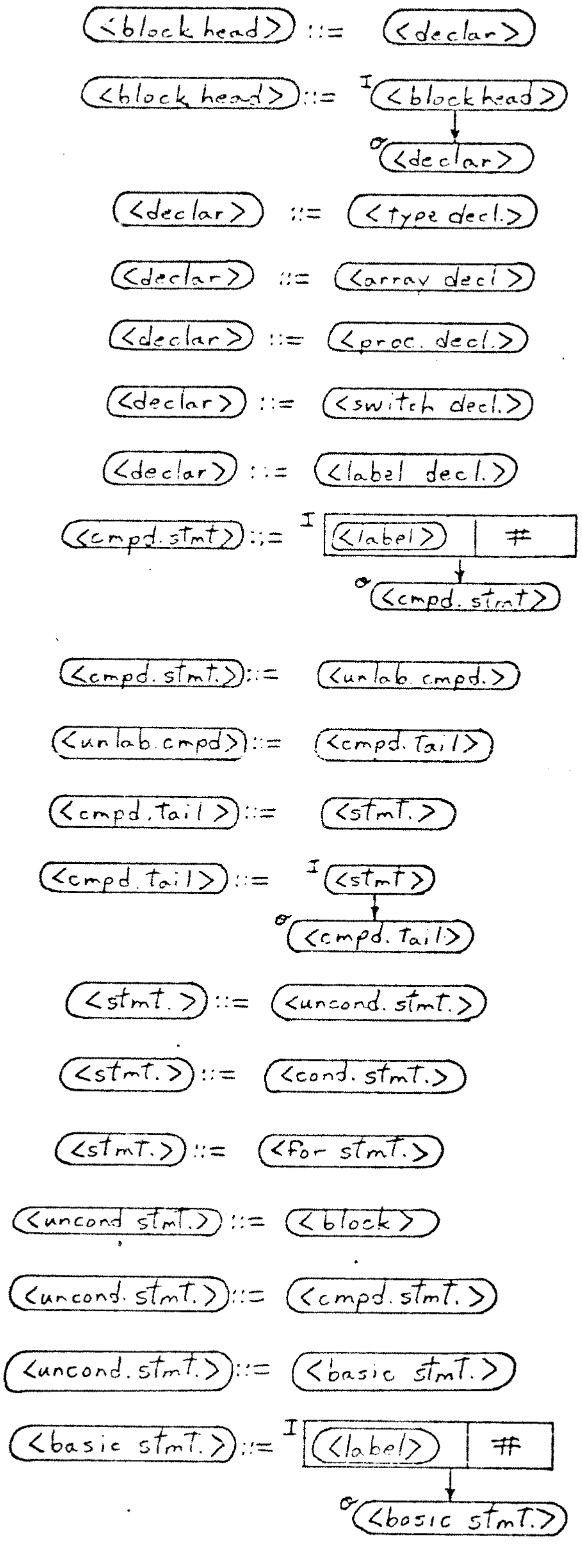
19 <stmt> ::= <for stmt.>

20 <uncond.stmt> ::= <block>

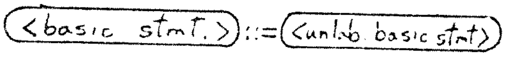
21 <uncond. stmt.>::= <cmpd. stmt.>

22 <uncond. stmt.> ::= <basic stmt.>

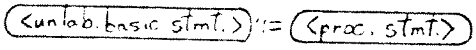
23 <basic stmt.>::= <label>:<basic stmt.>



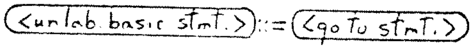
24 <basic stmt.>::= <unlab. basic stmt.>



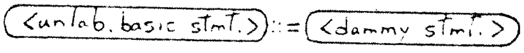
25 <unlab. basic stmt>::= <proc. stmt.>



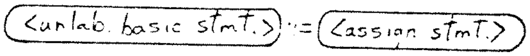
26 <unlab. basic stmt.>:= <go to stmt.>



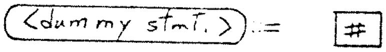
27 <unlab. basic stmt.>::= <dummy stmt.>



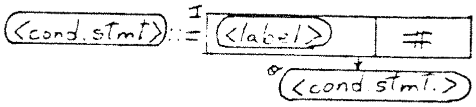
28 <unlab basic stmt.>::= <assign. stmt.>



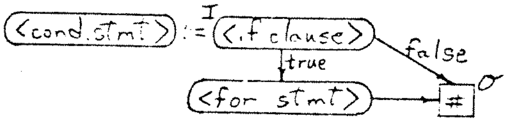
29 <dummy stmt.> ::= (empty)



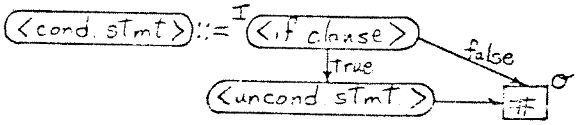
30 <cond. stmt> ::= <label>:<cond. stmt.>



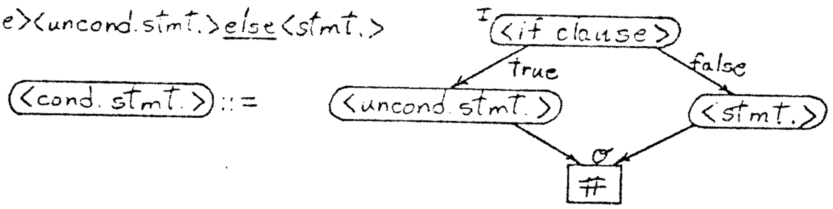
31 <cond. stmt.> ::= <if clause><for stmt>



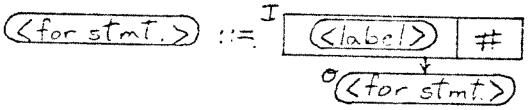
32* <cond. stmt.> ::= <if clause><uncond. stmt>



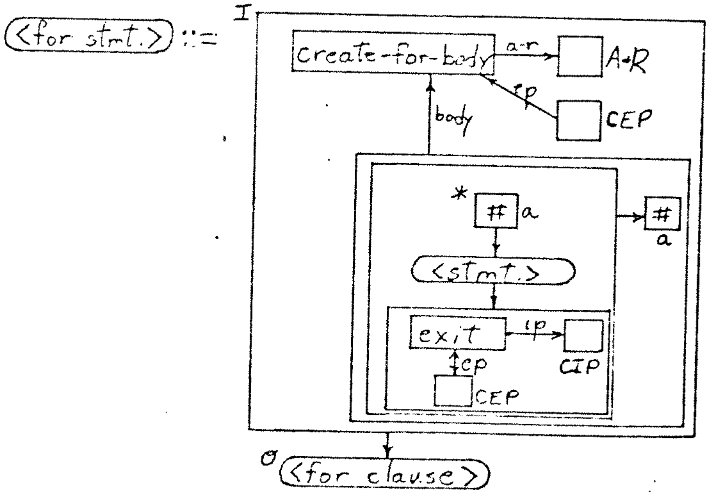
33* <cond. stmt.>::=<if clause><uncond. stmt.>else<stmt.>



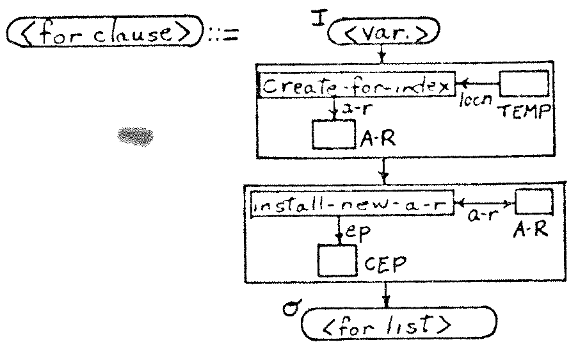
34 <for stmt.> ::= <label>:<for stmt.>



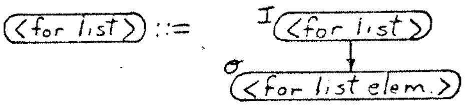
35 <for stmt.> ::= <for clause><stmt.>



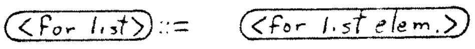
36 <for clause>::=for <var.>:=<for list>do



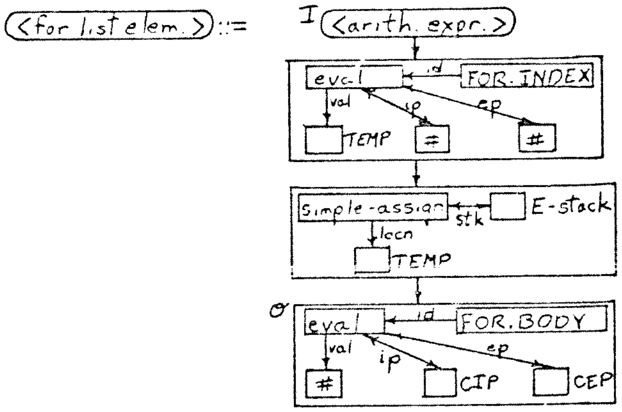
37 <for list>::=<for list>,<for list elem.>



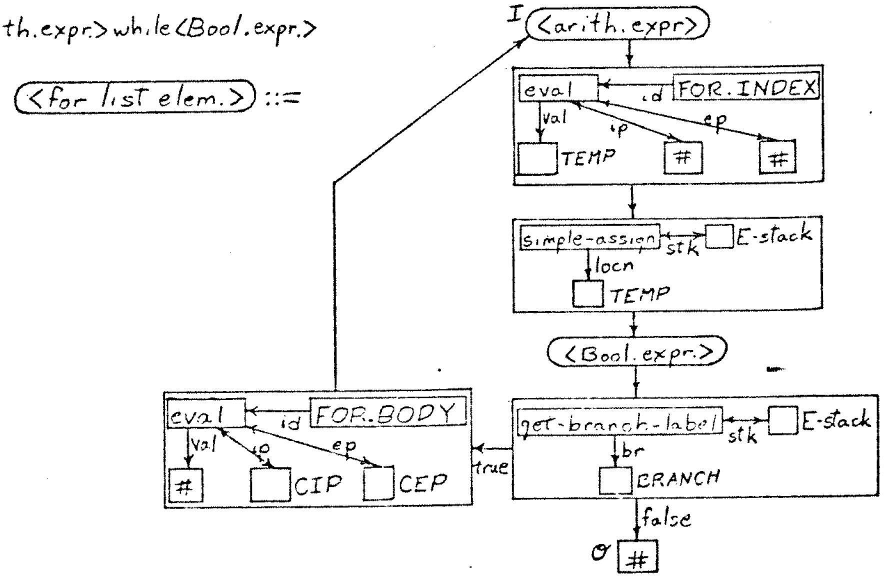
38 <for list>::=<for list elem.>



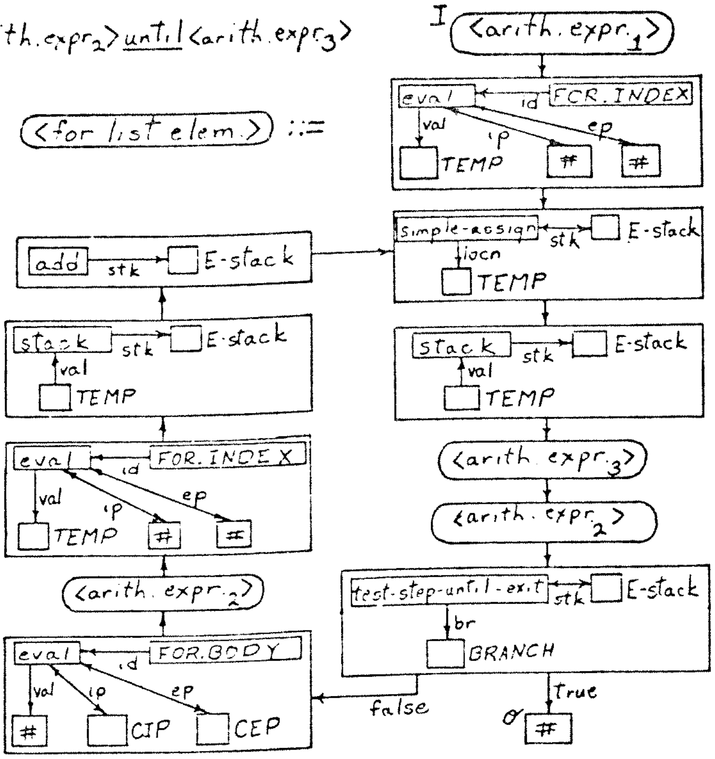
39 <for list elem>::=<arith.expr>



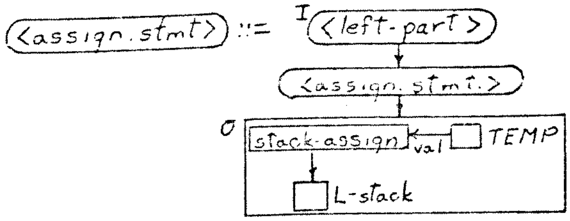
40 <for list elem.>::=<arith.expr>while<Bool.expr.>



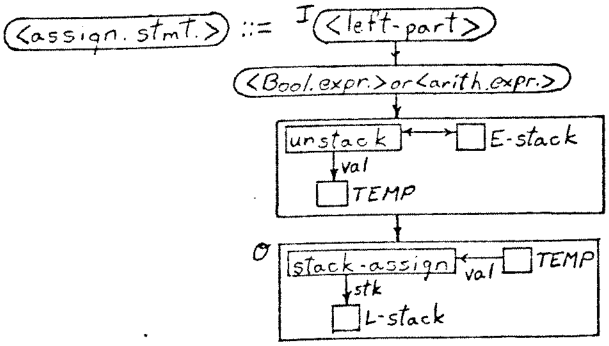
41 <for list elem.>::=<arith.expr$_1$> step<arith.expr$_2$> until<arith.expr$_3$>



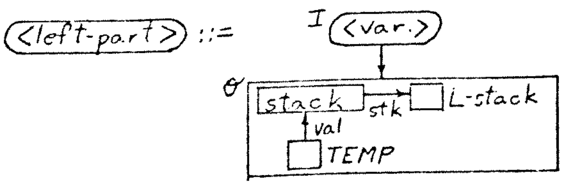
42* <assign.stmt>::=<left-part><assign.stmt.>



43* <assign.stmt>::=<left-part><Bool.expr> or<arith.expr>

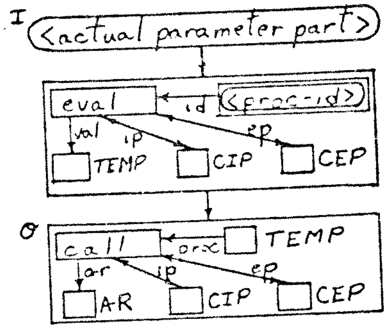


44 <left-part>::=<var.>



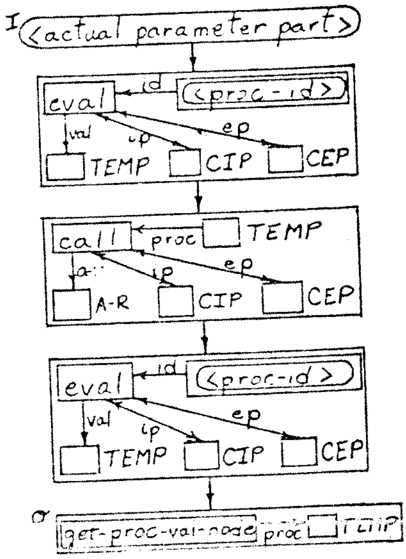
45 <proc. stmt.>::=<proc-id><actual parameter part>

<proc. stmt.> ::=



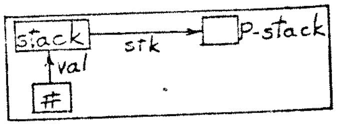
46 <function desig.>::=<proc-id><actual parameter part>

<function desig.> ::=



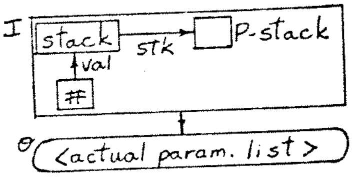
47*<actual parameter part>::= ()

<actual parameter part>::=



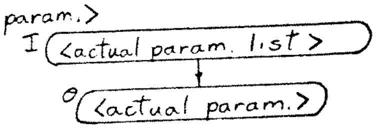
48 <actual parameter part>::=(<actual param. list>)

<actual parameter part>::=



49 <actual param. list>::=<actual param. list><param. delim.><actual param.>

<actual param. list> ::=

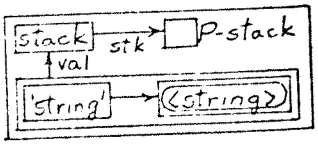


50 <actual param. list>::=<actual param.>

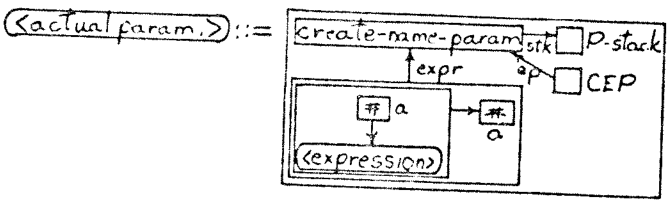
<actual param. list> ::= <actual param.>

51 <actual param.>::= <string>

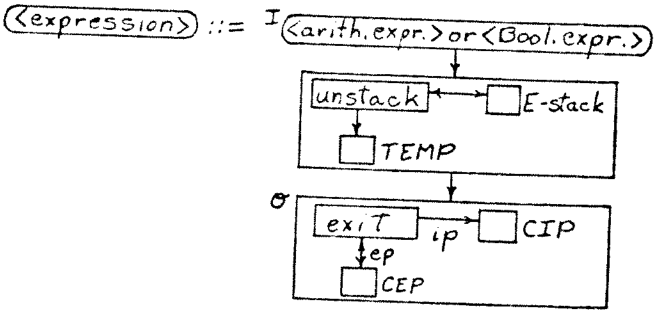
<actual param.> ::=



52 <actual param.>::=<expression>

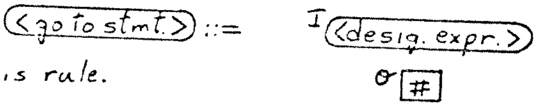


53 <expression>::=<arith.expr.> or <Bool. expr.>

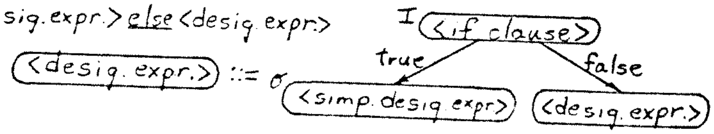


54 <goto stmt.>::= go to <desig expr.>

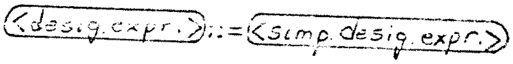
Note: The graph is disconnected by this rule.



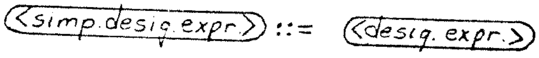
55 <desig.expr.>::=<if clause><simp.desig.expr.> else <desig.expr.>



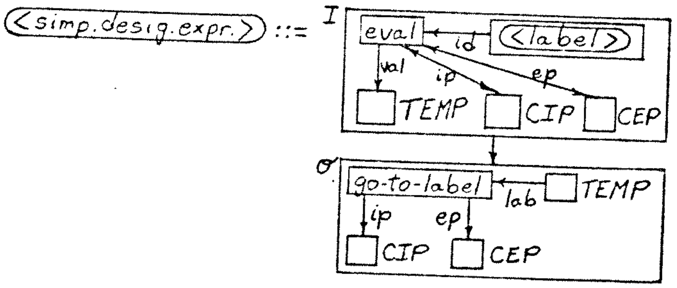
56 <desig.expr.>::=<simp.desig.expr.>



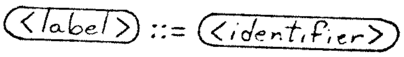
57 <simp.desig.expr.>::=(<desig.expr.>)



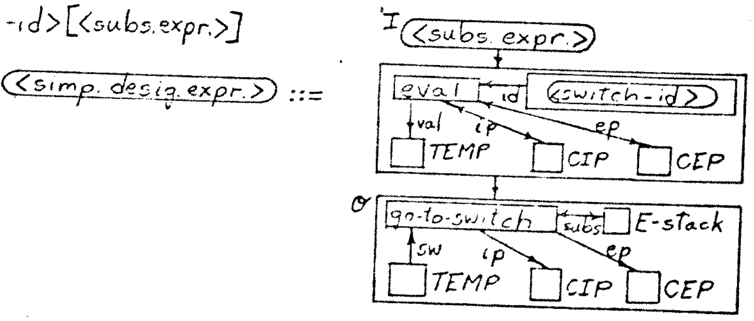
58 <simp.desig.expr.>::= <label>



59 <label>::=<identifier>

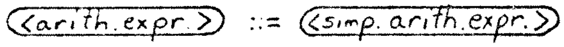


60* <simp.desig.expr.>::= <switch-id>[<subs.expr.>]

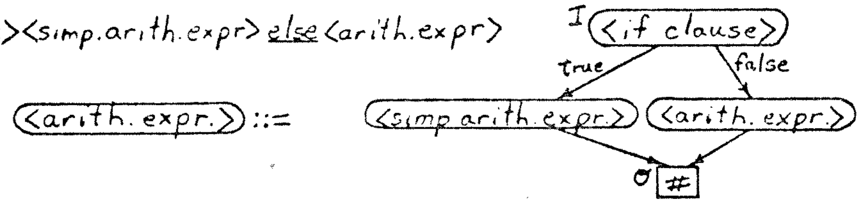


Expressions and Variables

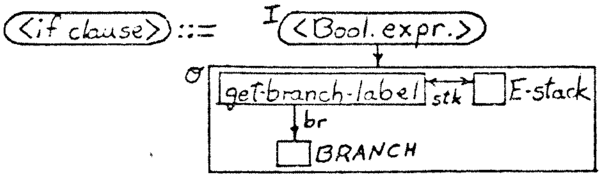
61 <arith.expr.>::=<simp.arith.expr.>



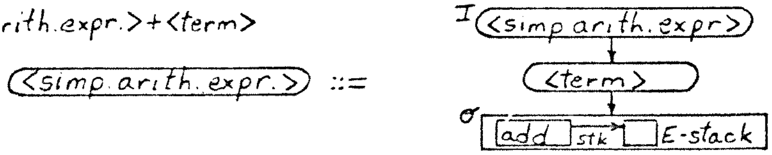
62 <arith.expr.>::=<if clause><simp.arith.expr> else <arith.expr>



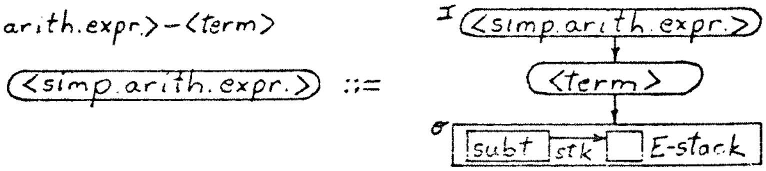
63 <if clause> ::= if <Bool.expr.> then



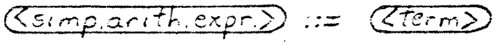
64* <simp.arith.expr.>::=<simp.arith.expr.>+<term>



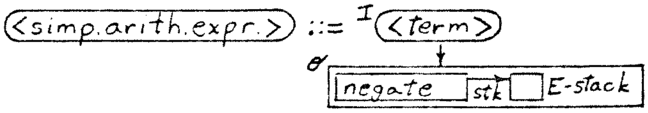
65* <simp.arith.expr.>::=<simp.arith.expr.>−<term>



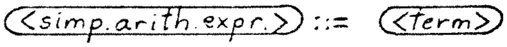
66* <simp.arith.expr.>::= + <term>



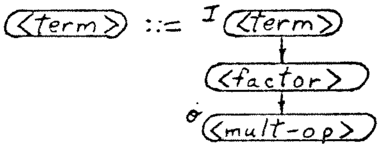
67* <simp.arith.expr.> ::= −<term>



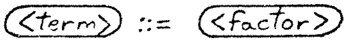
68 <simp.arith.expr.> ::= <term>



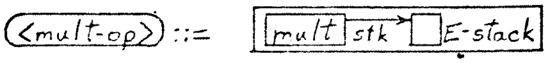
69 <term>::=<term><mult-op><factor>



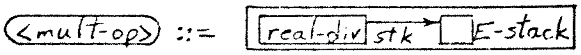
70 <term> ::= <factor>



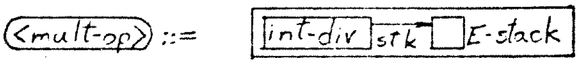
71 <mult-op>:: = *



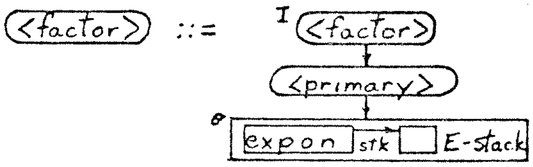
72 <mult-op>::= /



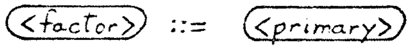
73 <mult-op>::= ÷



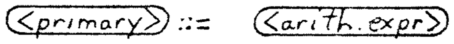
74 <factor>::=<factor>↑<primary>



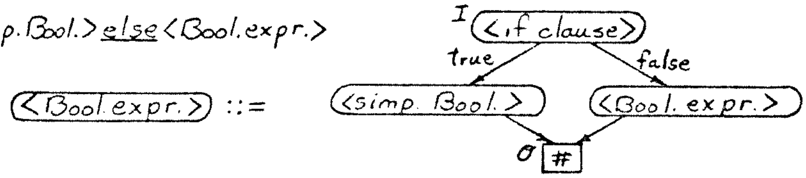
75 <factor>::=<primary>



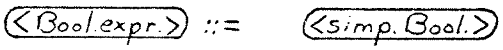
76 <primary>::= (<arith.expr.>)



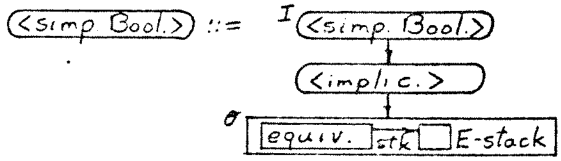
77 <Bool.expr.>::=<if clause><simp.Bool.>else<Bool.expr.>



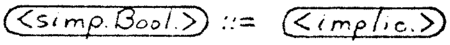
78 <Bool.expr.>::=<simp.Bool.>



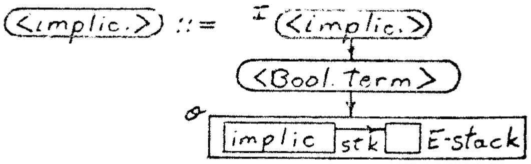
79 <simp. Bool.>::=<simp. Bool>≡<implic.>



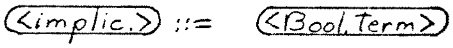
80 <simp. Bool.>::= <implic.>



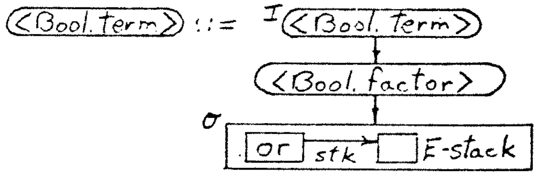
81 <implic.> ::= <implic.>⊃<Bool.term>



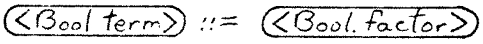
82 <implic.>::=<Bool. Term>



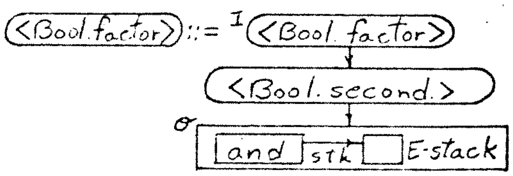
83 <Bool. term>::=<Bool.term>∨<Bool.factor>



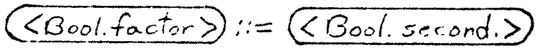
84 <Bool.term>::=<Bool.factor>



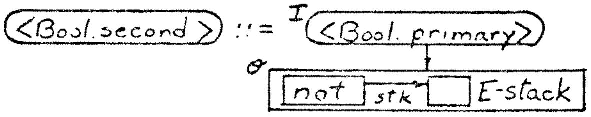
85 <Bool.factor>::=<Bool.factor>∧<Bool.second.>



86 <Bool.factor>::=<Bool.second.>

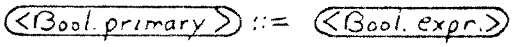


87 <Bool.second.>::= ¬<Bool.primary>

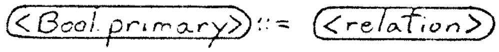


88 <Bool.second.>::= <Bool.primary>

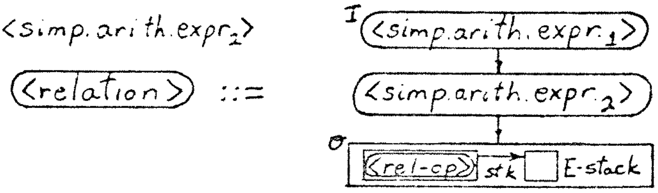
89 <Bool.primary>::= (<Bool.expr.>)



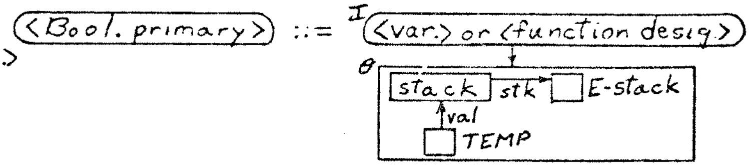
90 <Bool.primary>::= <relation>



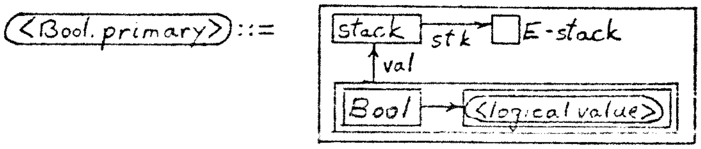
91 <relation>::=<simp.arith.expr.$_1$><rel-op><simp.arith.expr$_2$>



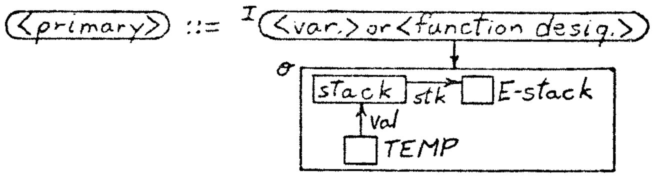
92 <Bool.primary>::=<var.>
or <function desig.>



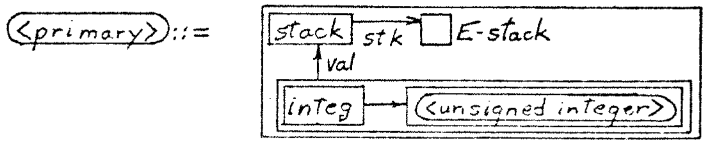
93 <Bool.primary>::=<logical value>



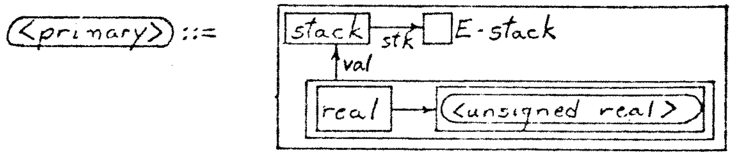
94 <primary>::= <var.>
or <function desig.>



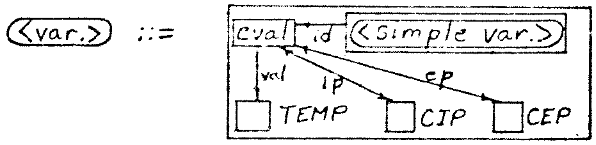
95* <primary>::=<unsigned integer>



96* <primary>::=<unsigned real>

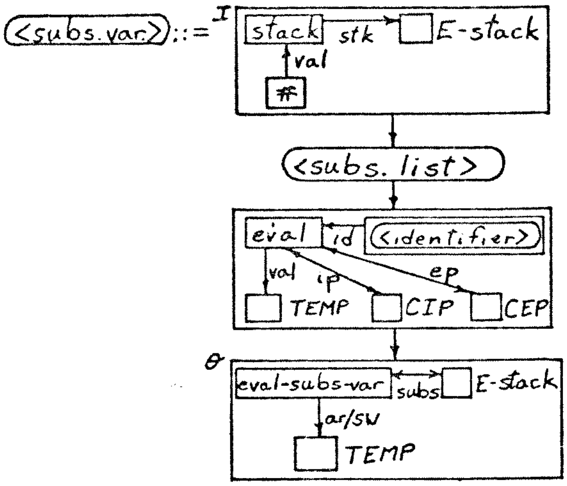


97 <var.>::=<simple var.>

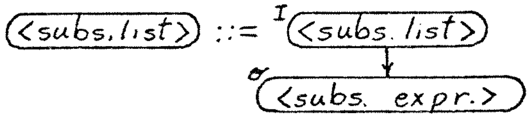


98 <var.>::=<subs var.> <var.> ::= <subs.var.>

99* <subs.var.>::=<identifier>[<subs. list>]



100 <subs. list>::=<subs. list>,<subs. expr.>

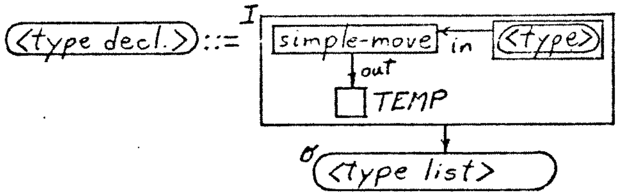


101 <subs. list>::=<subs. expr.> <subs. list>::= <subs. expr.>

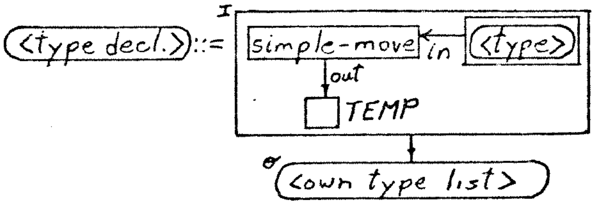
102 <subs. expr.>::=<arith. expr.> <subs. expr.> ::= <arith. expr.>

Declarations

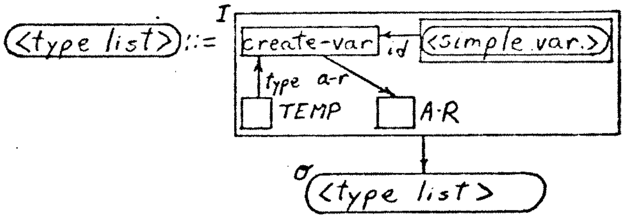
103* <type decl.>::=<type><type list>



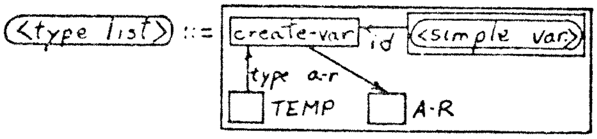
104* <type decl.>::=own <type><own type list>



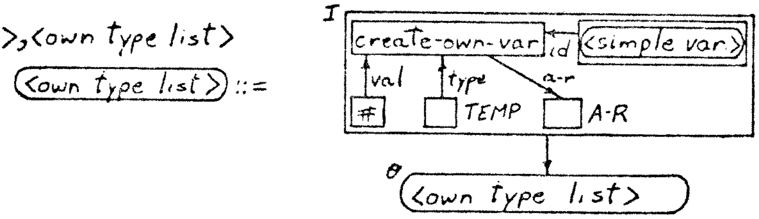
105 <type list>::=<simple var.>,<type list>



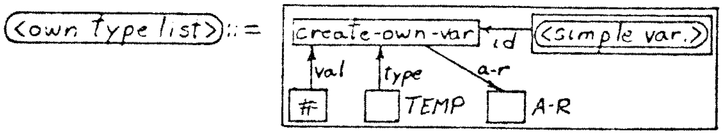
106 <type list>::=<simple var.>



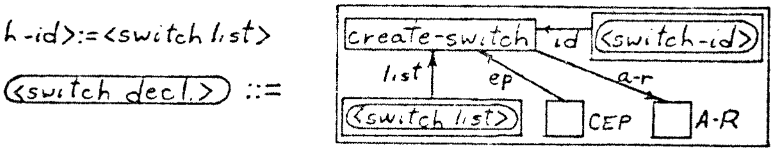
107* <own type list>::=<simple var.>,<own type list>



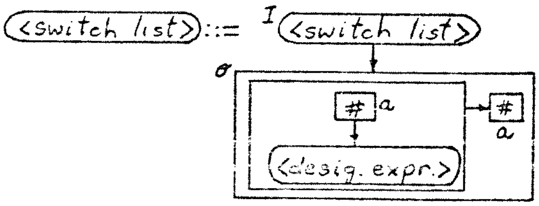
108* <own type list>::=<simple var.>



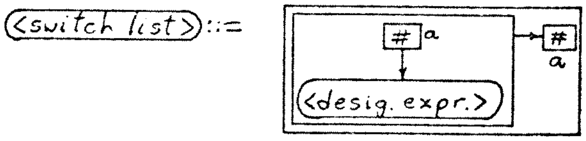
109 <switch decl.>::=switch <switch-id>:=<switch list>



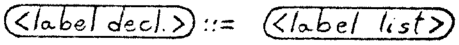
110 <switch list>::=<switch list>,<desig.expr.>



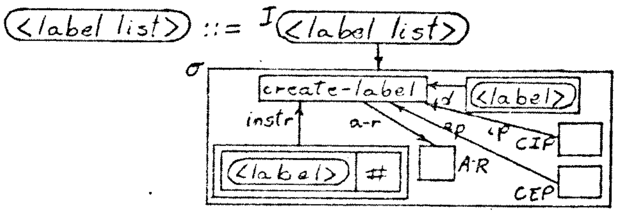
111 <switch list>::=<desig.expr.>



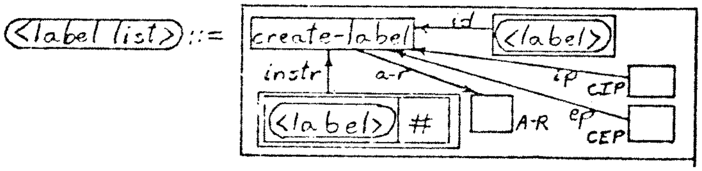
112* <label decl.>::=label <label list>



113* <label list>::=<label list>, <label>

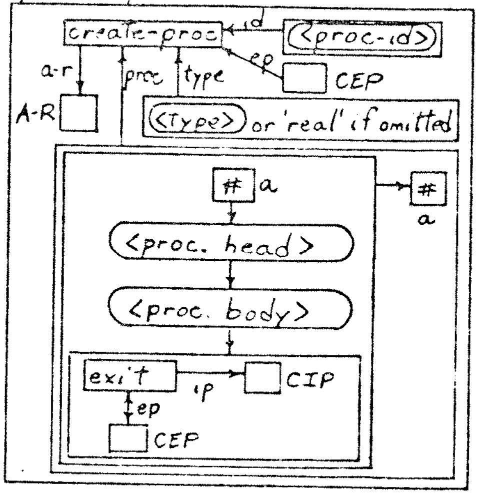


114* <label list>::=<label>



115* <proc. decl.>::=<type>procedure<proc-id><proc. head><proc. body>
or omit

<proc. decl.> ::=

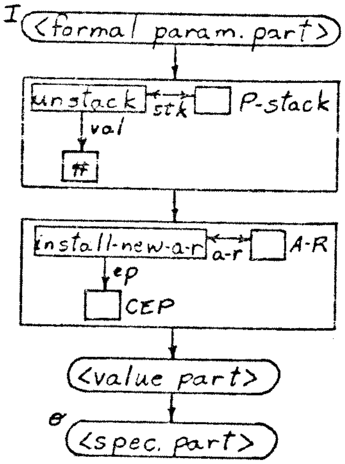


116 <proc. body>::=<stmt.> <proc. body> ::= <stmt.>

117 <proc. body>::= <code> <proc. body>::= <code>

118 <proc. head>::=<formal param. part>;<value part><spec. part>

<proc. head> ::=

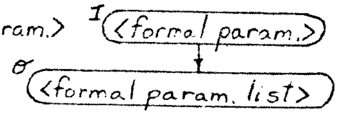


119 <formal param. part>::= empty <formal param. part>::= #

120 <formal param. part>::=(<formal param. list>) <formal param. part>::=<formal param. list>

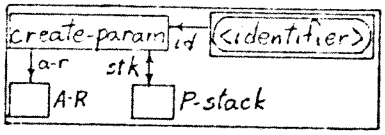
121 <formal param. list>::=<formal param. list><param. delim.><formal param.>

<formal param. list> ::=

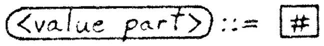


122 <formal param. list>::=<formal param.> <formal param. list>::= <formal param.>

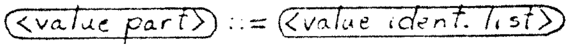
123 <formal param.>::=<identifier> <formal param.>::=



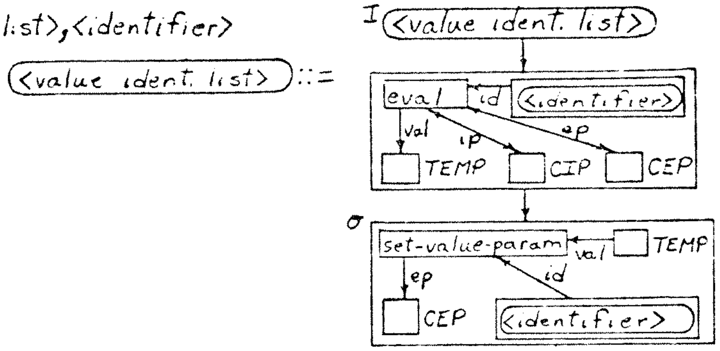
124 <value part>::= empty



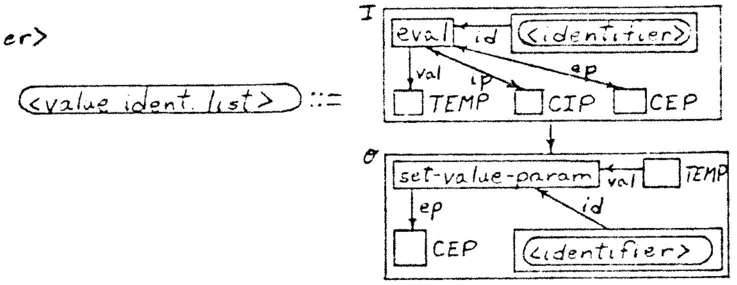
125* <value part>::= value<value ident. list>;



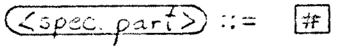
126* <value ident. list>::=<value ident. list>,<identifier>



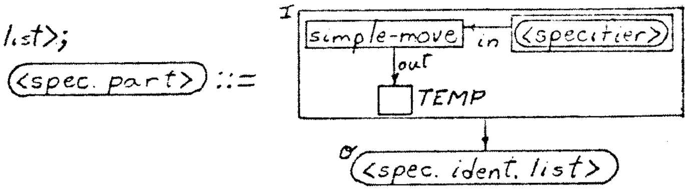
127* <value ident. list> ::= <identifier>



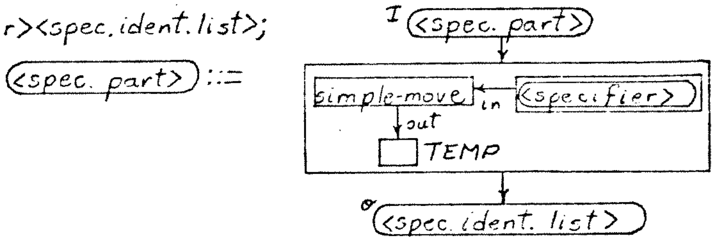
128 <spec. part>::= empty



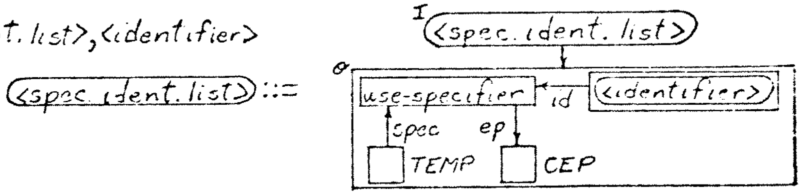
129* <spec. part>::=<specifier><spec. ident. list>;



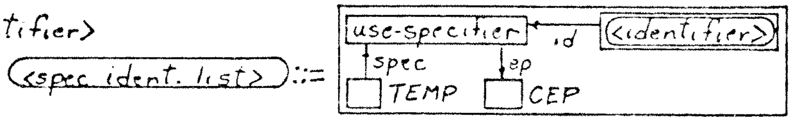
130* <spec. part>::=<spec. part><specifier><spec. ident. list>;



131* <spec. ident. list>::=<spec. ident. list>,<identifier>



132* <spec. ident. list>::=<identifier>



133*<array decl.>::= array <real array list> (<array decl.>) ::= (<real array list>)

134* <array decl.>::=real array <real array list> (<array decl.>)::= (<real array list>)

135* <array decl.>::=own real array <own real list> (<array decl.>)::= (<own real list>)

136* <array decl.>::= integer array <integ array list> (<array decl.>)::= (<integ. array list>)

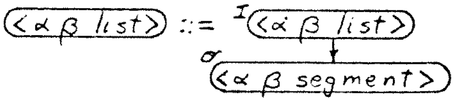
137* <array decl.>::=own integer array <own integ. list> (<array decl.>)::= (<own integ. list>)

138* <array decl.>::=Boolean array <Bool. array list> (<array decl.>)::= (<Bool. array list>)

139* <array decl.>::=own Boolean array <own Bool. list> (<array decl.>)::= (<own Bool. list>)

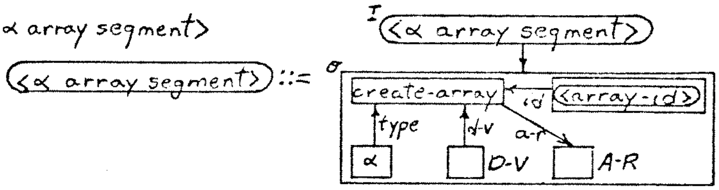
140* <α β list>::=<α β segment> (<α β list>) ::= (<α β segment>)

141* <$\alpha\beta$ list>::=<α β list>,<α β segment>



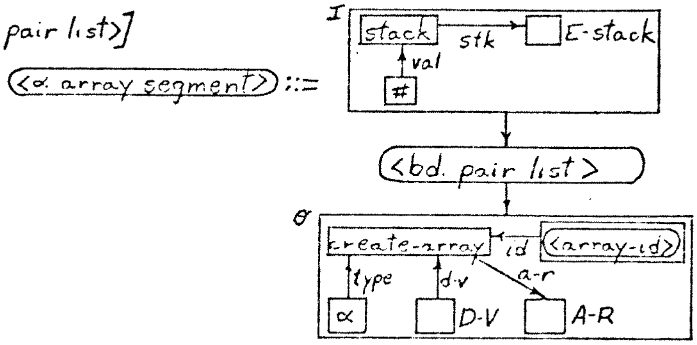
where $\alpha\beta$ = one of {real array
integ. array
Bool. array
own real
own integ.
own Bool.

142* <α array segment>::=<array-id>,<α array segment>

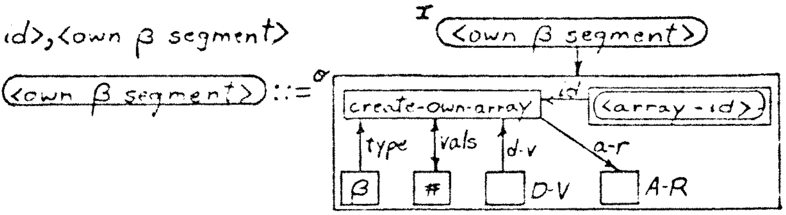


143* <α array segment>::=<array-id>[<bd. pair list>]

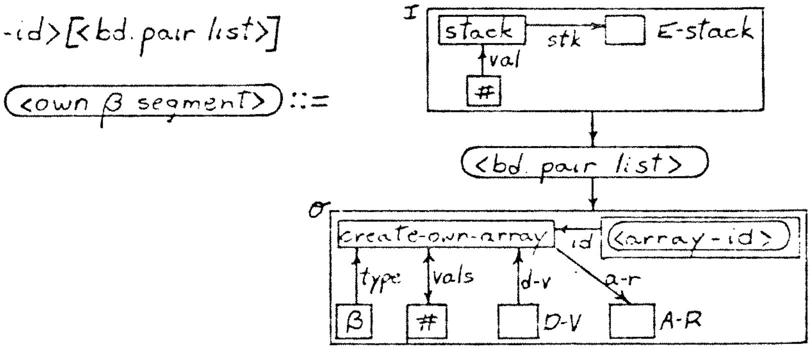
where α = one of {real
integ.
Bool.



144* <own β segment>::=<array-id>,<own β segment>

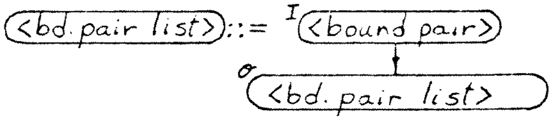


145* <own β segment>::=<array-id>[<bd. pair list>]

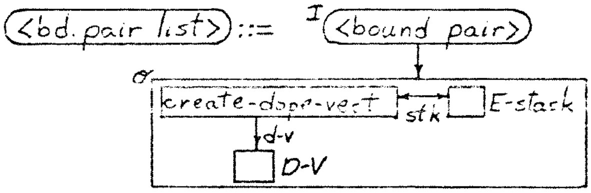


where β = one of { real, integ., Bool.

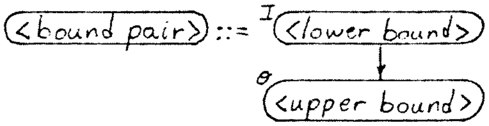
146 <bd. pair list>::=<bound pair>,<bd. pair list>



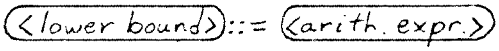
147 <bd. pair list>::=<bound pair>



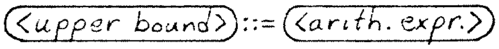
148 <bound pair>::=<lower bound>:<upper bound>



149 <lower bound>::=<arith. expr.>



150 <upper bound>::=<arith. expr.>



Appendix B: Reserved Word List

Reserved Word List

FCHINS	fetch-instruction
ENTBLK	enter-block
INSNAR	install-new-a-r
EXITX	exit
CRFRBDY	create-for-body
CRFRNDX	create-for-index
EVAL	eval
SIMPASG	simple-assign
GTBRLBL	get-branch-label
ADD	add
STACKX	stack
TSUEX	test-step-until-exit
STKASGN	stack-assign
UNSTACK	unstack
CALL	call
GTPRVLN	get-proc-val-node
CRNMPAR	create-name-parameter
GOTOLAB	go-to-label
GOTOSW	go-to-switch
SUBT	subt
NEGATE	negate
MULT	mult
REALDIV	realdiv
INTDIV	intdiv
EXPON	expon
EQUIV	equiv
IMPLIC	implic
OR	or
AND	and
NOT	not
RELOP	relop
EVSBVAR	eval-sub-var
SIMPMOV	simple-move
CRVAR	create-variable
CROVAR	create-own-var.
CRSW	create-switch
CRLAB	create-label
CRPROC	create-proc
CRPAR	create-parameter
STVLPAR	set-value-param
USESPEC	use-specifier

Appendix C: Graph Mini-Language Representation and the FEP's to Recognize It

```
1: <PROGRAM> = [*P (*FCH FCHINS) ↑BR↑ (*BRANCH) ↓*FCH NI↓ (*Q)
               ≡*FCH IP≡ [*CIP [(*A) ↓↓ <UNLABBLOCK>] ↓*A↓]]
               (*CEP) (*AR) (*DV) (*TEMP) (*ESTACK) (*LSTACK) (*PSTACK) /
169: <PROGRAM> = [*P (*FCH FCHINS) ↑BR↑ (*BRANCH) ↓*FCH NI↓ (*Q)
                ≡*FCH IP≡ [*CIP [(*A) ↓↓ <UNLABCMPD>] ↓*A↓]]
                 (*CEP) (*AR) (*DV) (*TEMP) (*ESTACK) (*LSTACK) (*PSTACK) /
2: <BLOCK> = [<LABEL>] ↓↓ <BLOCK> /
3: <BLOCK> = <UNLABBLOCK> /
4: <UNLABBLOCK> = [(ENTRBLK) ≡IP *CIP≡ ↑EP *CEP↑ ↓AR *AR↓ ↑BLK↑
                  [[(*A) ↓↓ <BLOCKHEAD> ↓↓ [(INSNAR) ↑AR *AR↑ ↓EP *CEP↓]
                   ↓↓ <CMPDTAIL> ↓↓ [(EXITX) ↓IP *CIP↓ ≡EP *CEP≡]] ↓*A↓]] /
5: <BLOCKHEAD> = <DECLAR> /
6: <BLOCKHEAD> = <BLOCKHEAD> ↓↓ <DECLAR> /
7: <DECLAR> = <TYPEDEC> /
8: <DECLAR> = <ARRAYDEC> /
9: <DECLAR> = <PROCDEC> /
10: <DECLAR> = <SWITCHDEC> /
11: <DECLAR> = <LABELDEC> /
12: <CMPDSTMT> = [<LABEL>] ↓↓ <CMPDSTMT> /
13: <CMPDSTMT> = <UNLABCMPD> /
14: <UNLABCMPD> = <CMPDTAIL> /
15: <CMPDTAIL> = <STMT> /
16: <CMPDTAIL> = <STMT> ↓↓ <CMPDTAIL> /
17: <STMT> = <UNCOND> /
18: <STMT> = <COND> /
19: <STMT> = <FORSTMT> /
20: <UNCOND> = <BLOCK> /
21: <UNCOND> = <CMPDSTMT> /
22: <UNCOND> = <BASIC> /
23: <BASIC> = [<LABEL>] ↓↓ <BASIC> /
24: <BASIC> = <UNLBASIC> /
25: <UNLBASIC> = <PROC> /
26: <UNLBASIC> = <GOTO> /
28: <UNLBASIC> = <ASSIGN> /
30: <COND> = [<LABEL>] ↓↓ <COND> /
31: <COND> = <*NT1 IFCLAUSE> ↓TRUE↓ <FORSTMT> ↓↓ () ↑FALSE *NT1↑ /
32: <COND> = <*NT1 IFCLAUSE> ↓TRUE↓ <UNCOND> ↓↓ () ↑FALSE *NT1↑/
33: <COND> = <*NT1 IFCLAUSE> ↓TRUE↓ <*NT2 UNCOND> ↓*NT1 FALSE↓
            <STMT> ↓↓ () ↑*NT2↑ /
34: <FORSTMT> = [<LABEL>] ↓↓ <FORSTMT> /
35: <FORSTMT> = [(CRFRBDY) ↓AR *AR↓ ↑EP *CEP↑ ↑BODY↑
               [[(*A) ↓↓ <STMT> ↓↓ [(EXITX) ≡EP *CEP≡ ↓IP *CIP↓]]
                ↓*A↓]] ↓↓ <FORCLAUSE> /
36: <FORCLAUSE> = <VAR> ↓↓ [(CRFRNDX) ↓AR *AR↓ ↑LOCN *TEMP↑] ↓↓
                 [(INSNAR) ↓EP *CEP↓ ≡AR *AR≡] ↓↓ <FORLIST> /
37: <FORLIST> = <FORLIST> ↓↓ <FORLISTELM> /
38: <FORLIST> = <FORLISTELM> /
39: <FORLISTELM> = <AREXP> ↓↓ [(*EV EVAL) ↓VAL *TEMP↓ ≡EP *CEP≡ ≡IP≡ ()
                      ↑*EV ID↑ (FORINDEX)] ↓↓ [(SIMPASG) ↓LOCN *TEMP↓
                 ≡STK *ESTACK≡] ↓↓ [(*EV EVAL) ≡IP *CIP≡ ≡EP *CEP≡ ↓VAL↓ ()
                  ↑*EV ID↑ (FORBODY)] /
40: <FORLISTELM> = <*NT1 AREXP> ↓↓ [(*EV EVAL) ↓VAL *TEMP↓ ≡EP *CEP≡
                 ≡IP≡ () ↑*EV ID↑ (FORINDEX)] ↓↓
                [(SIMPASG) ↓LOCN *TEMP↓ ≡STK *ESTACK≡] ↓↓ <BOOLEXP> ↓↓
                [*T1 (GTBRLBL) ↓BR *BRANCH↓ ≡STK *ESTACK≡] ↓TRUE↓
               [(*EV EVAL) ≡IP *CIP≡ ≡EP *CEP≡ ↓VAL↓ () ↑*EV ID↑
                  (FORBODY)] ↓*NT1↓ ↓*T1 FALSE↓ () /
41: <FORLISTELM> = <AREXP> ↓↓ [(*EV EVAL) ↓VAL *TEMP↓ ≡EP *CEP≡
                  ≡IP≡ () ↑*EV ID↑ (FORINDEX)] ↓↓
                [*NO (SIMPASG) ↓LOCN *TEMP↓ ≡STK *ESTACK≡] ↑↑
               [(ADD) ↓STK *ESTACK↓] ↑↑ [(STACKX) ↑VAL *TEMP↑
```

```
                ↓STK *ESTACK↓] ↑↑ [(*EV EVAL) ↓VAL *TEMP↓ ≡EP *CEP≡
                  ≡IP≡ () ≡*EV ID≡ (FORINDEX)] ↑↑ <*NT1 AREXP> ↑↑
                 [(*EV EVAL) ≡IP *CIP≡ ≡EP *CEP≡ ↑ID↑ (FORBODY)
              ↓*EV VAL↓ ()] ↑FALSE↑ [*N1 (TSUEX) ↓BR *BRANCH↓ ≡STK *ESTACK≡]
             ↑*NT1↑ ↑*NT1 0↑ <AREXP> ↑↑ [(STACKX) ↑VAL *TEMP↑
             ↓STK *ESTACK↓] ↑*N0↑ ↓*N1 TRUE↓ () /
42: <ASSIGN> = <LEFTPART> ↓↓ <ASSIGN> ↓↓ [(STKASGN) ↓STK *LSTACK↓ ↑VAL *TEMP↑] /
43: <ASSIGN> = <LEFTPART> ↓↓ <AREXP> ↓↓ [(UNSTACK) ↓VAL *TEMP↓
                   ≡STK *ESTACK≡] ↓↓ [(STKASGN) ↓STK *LSTACK↓ ↑VAL *TEMP↑] /
170: <ASSIGN> = <LEFTPART> ↓↓ <BOOLEXP> ↓↓ [(UNSTACK) ↓VAL *TEMP↓
                  ≡STK *ESTACK≡] ↓↓ [(STKASGN) ↓STK *LSTACK↓ ↑VAL *TEMP↑] /
44: <LEFTPART> = <VAR> ↓↓ [(STACKX) ↑VAL *TEMP↑ ↓STK *LSTACK↓] /
45: <PROC> = <ACTPARPRT> ↓↓ [(EVAL) ↓VAL *TEMP↓ ≡IP *CIP≡ ≡EP *CEP≡
             ↑ID↑ [<PROCID>]] ↓↓ [(CALL) ↓AR *AR↓ ≡IP *CIP≡ ≡EP *CEP≡
            ↑PROC *TEMP↑] /
46: <FUNCDESIG> = <ACTPARPRT> ↓↓ [(FVAL) ↓VAL *TEMP↓ ≡IP *CIP≡ ≡EP *CEP≡
                   ↑ID↑ [<PROCID>]] ↓↓ [(CALL) ↓AR *AR↓ ≡IP *CIP≡
                  ≡EP *CEP≡ ↑ID↑ [<PROCID>]] ↓↓ [(GTPRVLN) ↓PROC *TEMP↓] /
47: <ACTPARPRT> = [(STACKX) ↓STK *PSTACK↓ ↑VAL↑ ()] /
48: <ACTPARPRT> = [(STACKX) ↓STK *PSTACK↓ ↑VAL↑ ()] ↓↓ <ACTPARLIST> /
49: <ACTPARLIST> = <ACTPARLIST> ↓↓ <ACTPARAM> /
50: <ACTPARLIST> = <ACTPARAM> /
51: <ACTPARAM> = [(STACKX) ↓STK *PSTACK↓ ↑VAL↑ [(STRING) ↓↓ [<STRING>]]] /
52: <ACTPARAM> = [(CHNMPAR) ↓STK *PSTACK↓ ↑EP *CEP↑ ↑EXPR↑
                 [[(*A) ↓↓ <EXPRESSION>] ↓*A↓]] /
53: <EXPRESSION> = <AREXP> ↓↓ [(UNSTACK) ≡STK *ESTACK≡ ↓VAL *TEMP↓]
                   ↓↓ [(EXITX) ≡EP *CEP≡ ↓IP *CIP↓] /
171: <EXPRESSION> = <BOOLEXP> ↓↓ [(UNSTACK) ≡STK *ESTACK≡ ↓VAL *TEMP↓]
                    ↓↓ [(EXITX) ≡EP *CEP≡ ↓IP *CIP↓] /
54: <GOTO> = <DESIGEXPR> () /
55: <DESIGEXPR> = <*NT1 IFCLAUSE> ↓FALSE↓ <DESIGEXPR> ↓*NT1 TRUE↓ <SDESIG> /
56: <DESIGEXPR> = <SDESIG> /
57: <SDESIG> = <DESIGEXPR> /
58: <SDESIG> = [(EVAL) ↓VAL *TEMP↓ ≡IP *CIP≡ ≡EP *CEP≡ ↑ID↑
                  [<LABEL>]] ↓↓ [(GOTOLAB) ↓IP *CIP↓ ↓EP *CEP↓ ↑LAB *TEMP↑] /
59: <LABEL> = <IDENT> /
60: <SDESIG> = <SUBSEXPR> ↓↓ [(EVAL) ↓VAL *TEMP↓ ≡IP *CIP≡ ≡EP *CEP≡
                ↑ID↑ [<SWITCHID>]] ↓↓ [(GOTOSW) ↑SW *TEMP↑ ↓IP *CIP↓
                ↓EP *CEP↓ ≡SUBS *ESTACK≡] /
61: <AREXP> = <SAEXP> /
62: <AREXP> = <*NT1 IFCLAUSE> ↓TRUE↓ <*NT2 SAEXP> ↓*NT1 FALSE↓
               <AREXP> ↓↓ () ↑*NT2↑ /
63: <IFCLAUSE> = <BOOLEXP> ↓↓ [(GTBRLBL) ↓BR *BRANCH↓ ≡STK *ESTACK≡] /
64: <SAEXP> = <SAEXP> ↓↓ <TERM> ↓↓ [(ADD) ↓STK *ESTACK↓] /
65: <SAEXP> = <SAEXP> ↓↓ <TERM> ↓↓ [(SUBT) ↓STK *ESTACK↓] /
66: <SAEXP> = <TERM> /
67: <SAEXP> = <TERM> ↓↓ [(NEGATE) ↓STK *ESTACK↓] /
68: <SAEXP> = <TERM> /
69: <TERM> = <TERM> ↓↓ <FACTOR> ↓↓ <MULTOP> /
70: <TERM> = <FACTOR> /
71: <MULTOP> = [(MULT) ↓STK *ESTACK↓] /
72: <MULTOP> = [(REALDIV) ↓STK *ESTACK↓] /
73: <MULTOP> = [(INTDIV) ↓STK *ESTACK↓] /
74: <FACTOR> = <FACTOR> ↓↓ <PRIMARY> ↓↓ [(EXPON) ↓STK *ESTACK↓] /
75: <FACTOR> = <PRIMARY> /
76: <PRIMARY> = <AREXP> /
77: <BOOLEXP> = <*NT1 IFCLAUSE> ↓TRUE↓ <*NT2 SIMPBOOL> ↓*NT1 FALSE↓
                <BOOLEXP> ↓↓ () ↑*NT2↑ /
78: <BOOLEXP> = <SIMPBOOL> /
79: <SIMPBOOL> = <SIMPBOOL> ↓↓ <IMPLIC> ↓↓ [(EQUIV) ↓STK *ESTACK↓] /
80: <SIMPBOOL> = <IMPLIC> /
81: <IMPLIC> = <IMPLIC> ↓↓ <BOOLTERM> ↓↓ [(IMPLIC) ↓STK *ESTACK↓] /
```

```
82: <IMPLIC> = <BOOLTERM> /
83: <BOOLTERM> = <BOOLTERM> ↓↓ <BOOLFAC> ↓↓ [(OR) ↓STK *ESTACK↓] /
84: <BOOLTERM> = <BOOLFAC> /
85: <BOOLFAC> = <BOOLFAC> ↓↓ <BOOLSEC> ↓↓ [(AND) ↓STK *ESTACK↓] /
86: <BOOLFAC> = <BOOLSEC> /
87: <BOOLSEC> = <BOOLPRIM> ↓↓ [(NOT) ↓STK *ESTACK↓] /
88: <BOOLSEC> = <BOOLPRIM> /
89: <BOOLPRIM> = <BOOLEXPR> /
90: <BOOLPRIM> = <RELATION> /
91: <RELATION> = <SAEXP> ↓↓ <SAEXP> ↓↓ [(RELOP) ↓STK *ESTACK↓
                    ↑OPR↑ [<RELOP>]] /
92: <BOOLPRIM> = <VAR> ↓↓ [(STACKX) ↓STK *ESTACK↓ ↑VAL *TEMP↑] /
173: <BOOLPRIM> = <FUNCDESIG> ↓↓ [(STACKX) ↓STK *ESTACK↓ ↑VAL *TEMP↑] /
93: <BOOLPRIM> = [(STACKX) ↓STK *ESTACK↓ ↑VAL↑ [(BOOLEAN) ↓↓ [<LOGVAL>]]] /
94: <PRIMARY> = <VAR> ↓↓ [(STACKX) ↓STK *ESTACK↓ ↑VAL *TEMP↑] /
172: <PRIMARY> = <FUNCDESIG> ↓↓ [(STACKX) ↓STK *ESTACK↓ ↑VAL *TEMP↑] /
95: <PRIMARY> = [(STACKX) ↓STK *ESTACK↓ ↑VAL↑ [(INTEGER) ↓↓ [<UNSINTEG>]]] /
96: <PRIMARY> = [(STACKX) ↓STK *ESTACK↓ ↑VAL↑ [(REAL) ↓↓ [<UNSREAL>]]] /
97: <VAR> = [(EVAL) ↓VAL *TEMP↓ ≡IP *CIP≡ ≡EP *CEP≡ ↑ID↑ [<IDENT>]] /
98: <VAR> = <SUBSVAR> /
99: <SUBSVAR> = [(STACKX) ↓STK *ESTACK↓ ↑VAL↑ ()] ↓↓ <SUBSLIST> ↓↓
                [(EVAL) ↓VAL *TEMP↓ ≡IP *CIP≡ ≡EP *CEP≡ ↑ID↑ [<IDENT>]] ↓↓
                 [(EVSBVAR) ↓ARRAY *TEMP↓ ≡SUBS *ESTACK≡] /
100: <SUBSLIST> = <SUBSLIST> ↓↓ <SUBSEXPR> /
101: <SUBSLIST> = <SUBSEXPR> /
102: <SUBSEXPR> = <AREXP> /
103: <TYPEDEC> = [(SIMPMOV) ↓OUT *TEMP↓ ↑IN↑ [<TYPE>]] ↓↓ <TYPELIST> /
104: <TYPEDEC> = [(SIMPMOV) ↓OUT *TEMP↓ ↑IN↑ [<TYPE>]] ↓↓ <OWNTYPLIST> /
105: <TYPELIST> = [(CRVAR) ↑TYPE *TEMP↑ ↓AR *AR↓ ↑ID↑ [<IDENT>]] ↓↓
                 <TYPELIST> /
106: <TYPELIST> = [(CRVAR) ↑TYPE *TEMP↑ ↓AR *AR↓ ↑ID↑ [<IDENT>]] /
107: <OWNTYPLIST> = [(*CR CROVAR) ↑TYPE *TEMP↑ ↓AR *AR↓ ↑VAL↑ ()
                    ↑*CR ID↑ [<IDENT>]] ↓↓ <OWNTYPLIST> /
108: <OWNTYPLIST> = [(*CR CROVAR) ↑TYPE *TEMP↑ ↓AR *AR↓ ↑VAL↑ ()
                    ↑*CR ID↑ [<IDENT>]] /
109: <SWITCHDEC> = [(*CR CRSW) ↑EP *CEP↑ ↓AR *AR↓ ↑LIST↑ [<SWITCHLIST>]
                 ↑*CR ID↑ [<SWITCHID>]] /
110: <SWITCHLIST> = <SWITCHLIST> ↓↓ [[(*A) ↓↓ <DESIGEXPR>] ↓*A↓] /
111: <SWITCHLIST> = [[(*A) ↓↓ <DESIGEXPR>] ↓*A↓] /
112: <LABELDEC> = <LABELLIST> /
113: <LABELLIST> = <LABELLIST> ↓↓ [(*CR CRLAB) ↓AR *AR↓ ↑EP *CEP↑
                   ↑IP *CIP↑ ↑INSTR↑ [*N] <LABEL>] ↑*CR ID *N]↑] /
114: <LABELLIST> = [(*CR CRLAB) ↓AR *AR↓ ↑EP *CEP↑ ↑IP *CIP↑ ↑INSTR↑
                    [*N] <LABEL>] ↑*CR ID *N]↑] /
115: <PROCDEC> = [(*CR CRPROC) ↓AR *AR↓ ↑EP *CEP↑ ↑TYPE↑ [<TYPE>]
                 ↑*CR ID↑ [<PROCID>] ↑*CR PROC↑ [[(*A) ↓↓ <PROCHEAD>
                 ↓↓ <PROCBODY> ↓↓ [(EXITX) ≡EP *CEP≡ ↓IP *CIP↓]] ↓*A↓]] /
174: <PROCDEC> = [(*CR CRPROC) ↓AR *AR↓ ↑EP *CEP↑ ↑TYPE↑ [(REAL)]
                  ↑*CR ID↑ [<PROCID>] ↑*CR PROC↑ [[(*A) ↓↓ <PROCHEAD> ↓↓
                <PROCBODY> ↓↓ [(EXITX) ≡EP *CEP≡ ↓IP *CIP↓]] ↓*A↓]] /
116: <PROCBODY> = <STMT> /
117: <PROCBODY> = <CODE> /
118: <PROCHEAD> = <FRMPARPART> ↓↓ [(UNSTACK) ≡STK *PSTACK≡ ↓VAL↓ ()] ↓↓
                   [(INSNAR) ≡AR *AR≡ ↓EP *CEP↓] ↓↓ <VALUEPART> ↓↓
                    <SPECPART> /
119: <FRMPARPART> = () /
120: <FRMPARPART> = <FRMPARLIST> /
121: <FRMPARLIST> = <FRMPAR> ↓↓ <FRMPARLIST> /
122: <FRMPARLIST> = <FRMPAR> /
123: <FRMPAR> = [(CRPAR) ↓AR *AR↓ ≡STK *PSTACK≡ ↑ID↑ [<IDENT>]] /
124: <VALUEPART> = () /
125: <VALUEPART> = <VALIDLIST> /
```

```
126: <VALIDLIST> = <VALIDLIST> ↓↓ [(EVAL) ↓VAL *TEMP↓ ≡IP *CIP≡ ≡EP *CEP≡
                   ↑ID↑ [*N] <IDENT>]] ↓↓ [(STVLPAR) ↓EP *CEP↓ ↑VAL *TEMP↑
                    ↑ID *N1↑] /
127: <VALIDLIST> = [(EVAL) ↓VAL *TEMP↓ ≡IP *CIP≡ ≡EP *CEP≡ ↑ID↑ [*N] <IDENT>]]
                   ↓↓ [(STVLPAR) ↓EP *CEP↓ ↑VAL *TEMP↑ ↑ID *N1↑] /
128: <SPECPART> = () /
129: <SPECPART> = [(SIMPMOV) ↓OUT *TEMP↓ ↑IN↑ [<SPECIFIER>]] ↓↓
                   <SPECIDLIST> /
130: <SPECPART> = <SPECPART> ↓↓ [(SIMPMOV) ↓OUT *TEMP↓ ↑IN↑ [<SPECIFIER>]]
                  ↓↓ <SPECIDLIST> /
131: <SPECIDLIST> = <SPECIDLIST> ↓↓ [(USESPEC) ↑SPEC *TEMP↑ ↓EP *CEP↓
                    ↑ID↑ [<IDENT>]] /
132: <SPECIDLIST> = [(USESPEC) ↑SPEC *TEMP↑ ↓EP *CEP↓ ↑ID↑
                   [<IDENT>]] /


200: <IDENT> = <LITERAL> /
201: <PROCID> = <LITERAL>  /
202: <STRING> = <LITERAL> /
203: <UNSINTEG> = <LITERAL>    /
204: <UNSREAL> = <LITERAL>   /
205: <RELOP> = <LITERAL> /
206: <TYPE> = <LITERAL>    /
207: <LOGVAL> = <LITERAL> /
208: <SPECIFIER> = <LITERAL> /
 /
 N LAB   L5    L4       L3       L2       L1      ↱  R3  R2       R1       ACTION   S JUMP        N   L R

 1 IN                                     <SG>    ↱                       /                 -0 *RN0   2     1   1 1
 2 RN0                                    <NUMBR>                                  EXEC    1 *RD0   6     2   1 0
 3                                        /                                        HALT   -0  HLT  56     3   1 0
 4                                        F-0-F                                           -0  HLT  56     4   1 0
 5                                        <SG>                                     ERROR   6  HLT  56     5   1 0
 6 RD0                  /        <NUMBR>  :       ↱                       /                 -0 *RL0  10     6   3 1
 7                                        <SG>                                     ERROR   7  HLT  56     7   1 0
 8 ND0                                    [                                        EXEC    8 *NL0  12     8   1 0
 9                                        (                                               -0 *NL0  12     9   1 0
10 RL0                                    <                                               -0 *NL0  12    10   1 0
11                                        <SG>                                     ERROR   1  HLT  56    11   1 0
12 NL0                                    *                                               -0 *NM0  14    12   1 0
13                                        <SG>    ↱      <N-LAB>  <SG>     EXEC    5  NL1  25    13   1 2
14 NM0                                    <IDENT> ↱               <NAME>   EXEC    6 *NM1  17    14   1 1
15                                        <NUMBR> ↱               <NAME>   EXEC   15 *NM1  17    15   1 1
16                                        <SG>                                     ERROR   2  HLT  56    16   1 0
17 NM1                  *        <NAME>   <SG>    ↱      <N-LAB>  <SG>              -0  NL1  25    17   3 2
18                               <NAME>   <SG>    ↱      <VALUE>  <SG>              -0  VL1  22    18   2 2
19 VL0                                    <IDENT> ↱               <VALUE>  EXEC    6 *VL1  22    19   1 1
20                                        <NUMBR> ↱               <VALUE>  EXEC   15 *VL1  22    20   1 1
21                                        <SG>    ↱      <VALUE>  <SG>     EXEC    5  VL1  22    21   1 2
22 VL1         <        <N-LAB>  <VALUE>  >                                               -0 *NT1  35    22   4 0
23             (        <N-LAB>  <VALUE>  )       ↱               <NODE>   EXEC   13 *ND1  37    23   4 1
24                                        <SG>                                     ERROR   3  HLT  56    24   1 0
25 NL1                  (        <N-LAB>  <SG>                                            -0  VL0  19    25   3 0
26                      [        <N-LAB>  <SG>                                     EXEC    3  ND0   8    26   3 0
27                      <        <N-LAB>  <SG>                                            -0  VL0  19    27   3 0
28                      ↑        <N-LAB>  <SG>                                            -0  AL0  43    28   3 0
29                      ↓        <N-LAB>  <SG>                                            -0  AL0  43    29   3 0
30                      ≡        <N-LAB>  <SG>                                            -0  AL0  43    30   3 0
31       ↓     <N-LAB>  <A-LAB>  <N-LAB>  ↓       ↱               <ARC>    EXEC    7 *AR1  46    31   5 1
32       ↑     <N-LAB>  <A-LAB>  <N-LAB>  ↑       ↱               <ARC>    EXEC    2 *AR1  46    32   5 1
33       ≡     <N-LAB>  <A-LAB>  <N-LAB>  ≡       ↱               <ARC>    EXEC   14 *AR1  46    33   5 1
34                                        <SG>                                     ERROR   4  HLT  56    34   1 0
35 NT1   <     <N-LAB>  <VALUE>  >        =       ↱      <NTERM>  =        EXEC    9 *ND0   8    35   5 2
36       <     <N-LAB>  <VALUE>  >        <SG>    ↱      <NODE>   <SG>     EXEC   10  ND1  37    36   5 2
37 ND1                           <NODE>   ↑                                               -0 *NL0  12    37   2 0
38                               <NODE>   ↓                                               -0 *NL0  12    38   2 0
39                               <NODE>   ≡                                               -0 *NL0  12    39   2 0
40                               <NODE>   ]       ↱      <S-0-N>  ]                       -0  SN1  50    40   2 2
41                               <NODE>   /       ↱      <S-0-N>  /                       -0  SN1  50    41   2 2
42                               <NODE>   <SG>                                            -0  ND0   8    42   2 0
43 AL0                                    <IDENT> ↱               <A-LAB>  EXEC    6 *NL0  12    43   1 1
44                                        <NUMBR> ↱               <A-LAB>  EXEC   15 *NL0  12    44   1 1
45                                        <SG>    ↱      <A-LAB>  <SG>     EXEC    5  NL0  12    45   1 2
46 AR1                           <ARC>    ↑                                               -0 *NL0  12    46   2 0
47                               <ARC>    ↓                                               -0 *NL0  12    47   2 0
48                               <ARC>    ≡                                               -0 *NL0  12    48   2 0
49                               <ARC>    <SG>    ↱      <S-0-A>  <SG>              -0  SA1  54    49   2 2
50 SN1                  <NODE>   <S-0-N>  <SG>    ↱      <S-0-N>  <SG>     EXEC   12  SN1  50    50   3 2
51             [        <N-LAB>  <S-0-N>  ]       ↱               <NODE>   EXEC   11 *ND1  37    51   4 1
52       /     <NTERM>  =        <S-0-N>  /       ↱               /        EXEC    4 *RN0   2    52   5 1
53                                        <SG>                                     ERROR   5  HLT  56    53   1 0
54 SA1                  <ARC>    <S-0-A>  <SG>    ↱      <S-0-A>  <SG>              -0  SA1  54    54   3 2
55                      <NODE>   <S-0-A>  <SG>    ↱      <NODE>   <SG>              -0  ND1  37    55   3 2
56 HLT                                    <SG>                                     HALT   -0  HLT  56    56   1 0
```

Appendix D: Graph Specifier Output Tables

```
PACKED SEQUENCES OF EXEC-SET CALLS, LITERAL RULES, AND RIGHT-SIDE POINTER TABLES

  N                    POINTERS                 EXEC CALLS              LITERAL RULES

0001         00000000010000000001     15060603061011060501     PROGRAM
0002         00000000550000000023     05060615050602050605     P
0003         00000000600000000026     06101605050615050607     FCH
0004         00000000620000000027     05050515050603051003     FCHINS
0005         00000001110000000042     07050506131412060507     BR
0006         00000001130000000043     05061505061314141413     BRANCH
0007         00000001160000000045     15050615050615050615     FCH
0010         00000001200000000046     14141414150506150506     NI
0011         00000001220000000047     00000000000004141414     P
0012         00000001240000000050     15060603061011060501     FCH
0013         00000001260000000051     05060615050602050605     TP
0014         00000001300000000052     06101605050615050607     CTP
0015         00000001330000000055     05050515050603051003     A
0016         00000001350000000056     07050506131412060507     UNLABBLOCK
0017         00000001370000000057     05061505061314141413     A
0020         00000001410000000060     15050615050615050615     CEP
0021         00000001440000000062     14141414150506150506     AR
0022         00000001460000000063     00000000000004141414     DV
0023         00000001500000000064     12060503061011060501     TEMP
0024         00000001520000000065     04141206050705050513     FSTACK
0025         00000001540000000066                              LSTACK
0026         00000001560000000067     00000412060511060501     PSTACK
0027         00000001600000000070     15060503051011060501     PROGRAM
0030         00000001630000000073     06050206060516060605     P
0031         00000001650000000074     10030510020506050706     FCH
0032         00000001670000000075     05070505051505060305     FCHINS
0033                                  05030510070505051206     BR
0034         00000001710000000076     07050605020606051506     BRANCH
0035                                  05051206050705050513     FCH
0036         00000001730000000077     06051506050305100705     NI
0037         00000001760000000102     14141413160606050705     P
0040         00000002050000000105     04131413070505061314     FCH
0041         00000002140000000110                              TP
0042         00000002260000000114     00000412060511060501     CTP
0043         00000002310000000117     05050512060511060501     A
0044         00000002510000000127     00000000041412060507     UNLABCMPD
0045         00000002660000000135     00000412060511060501     A
0046         00000002710000000137     00000412060511060501     CEP
0047         00000002730000000140     00000412060511060501     AR
0050         00000003260000000153     00000412060511060501     DV
0051         00000003750000000172     00000412060511060501     TEMP
0052         00000005000000000223     12060503051011060501     FSTACK
0053         00000005100000000227     04141206050705050513     LSTACK
0054         00000005420000000243                              PSTACK
0055         00000005510000000245     00000412060511060501     BLOCK
0056         00000005750000000256     00000412060511060501     LABEL
0057         00000006240000000270     00000412060511060501     BLOCK
0060         00000006310000000273     05050512060511060501     BLOCK
0061         00000006370000000277     00000000041412060507     UNLABBLOCK
0062         00000006420000000301     00000412060511060501     UNLABBLOCK
0063         00000006440000000302     00000412060511060501     ENTBLK
0064         00000006530000000307     00000412060511060501     TP
0065         00000006650000000314     00000412060511060501     CTP
0066         00000007150000000326     00000412060511060501     FP
0067         00000007170000000330     00000412060511060501     CEP
0070         00000007270000000333     12060503051011060501     AR
0071         00000007310000000334     04141206050705050513     AR
0072         00000007330000000335                              BLK
```

```
0073          00000007540000000344   00000412060511060501        B
0074          00000007560000000345   00000412060511060501        BLOCKHEAD
0075          00000010020000000355   00000412060511060501        INSNAR
0076          00000010040000000356   00000412060511060501        AR
0077          00000010160000000362   12060503051011060501        AR
0100          00000010250000000365   04141206050705050513        FP
0101          00000010330000000371                               CEP
0102          00000010410000000375   05060512060611060501        CMPUTATL
0103          00000010430000000376   05050705050512060507        EXITX
0104          00000010500000000401   00000414140206060515        IP
0105          00000010520000000402   05060512060611060501        CIP
0106          00000010560000000405   05050705050512060507        FP
0107          00000010600000000406   00000414140206060515        CEP
0110          00000010640000000410   05060512060611060501        A
0111          00000010700000000412   06050705060612060607        BLOCKHEAD
0112          00000010740000000414   05061505050705050512        DECLAR
0113          00000011020000000420   00000000041414140205        BLOCKHEAD
0114          00000011040000000421   12060503051011060501        BLOCKHEAD
0115          00000011060000000422   04141206050705050513        DECLAR
0116          00000011200000000426                               DECLAR
0117          00000011220000000427   15060503051011060501        TYPEDEC
0120          00000011300000000433   06050206060507060605        DECLAR
0121          00000011320000000434   05060305110305100205        ARRAYDEC
0122          00000011400000000440   05051206050705050515        DECLAR
0123          00000011420000000441   06051506050305100705        PROCDEC
0124          00000011500000000445   13141413070606051606        DECLAR
0125          00000011520000000446   05050513141307050506        SWITCHDEC
0126          00000011600000000452   00000000041412060507        DECLAR
0127          00000011620000000453   05050512060511060501        LABELDEC
0130          00000011670000000456   06060515060503051007        CMPUSTMT
0131          00000011710000000457   07050505130206060507        LABEL
0132          00000011730000000460   07060605150605030510        CMPUSTMT
0133          00000011750000000461   05070505051316060605        CMPUSTMT
0134          00000012050000000466   00000000041414141206        UNLABCMPD
0135          00000012230000000474   05050512060511060501        UNLABCMPD
0136          00000012320000000501   00000000041412060507        CMPUTATL
0137          00000012500000000507   00000412060511060501        CMPUTATL
0140          00000012570000000514   05050512060511060501        STMT
0141          00000012660000000521   06060515060603051007        CMPUTATL
0142          00000013000000000525   05160506051606060507        STMT
0143          00000013020000000526   14150605020506061505        CMPUTATL
0144          00000013260000000537   05030510070505051314        STMT
0145          00000013310000000541   16060605070606051506        UNCOND
0146          00000013330000000542   06060305100705050513        STMT
0147          00000013350000000543   05160606051606060515        COND
0150          00000013440000000547   02050606150505070506        STMT
0151          00000013530000000553   04141414131414150605        FORSTMT
0152          00000013640000000560                               UNCOND
0153          00000013740000000564   05050512060611060501        BLOCK
0154          00000014100000000571   06060515060603051007        UNCOND
0155          00000014230000000576   05160506051606060507        CMPUSTMT
0156          00000014370000000603   14150605020506061505        UNCOND
0157          00000014440000000607   05030510070505051314        BASIC
0160          00000014500000000612   16060605070606051506        BASIC
0161          00000014520000000613   05051206050705050513        LABEL
0162          00000014720000000620   06051506050306100705        BASIC
0163          00000015110000000625   05060513160606050706        BASIC
0164          00000015670000000653   06060515060603051007        UNLBASIC
0165          00000015710000000654   05070506051606060516        UNLBASIC
0166          00000015730000000655   14150605020506061505        PROC
0167          00000016100000000664   07050606070505061314        UNLBASIC
0170          00000016110000000665   04141414141414150505        GOTO
```

```
0171            00000016130000000666                           UNLBASIC
0172            00000016160000000670    05050512060511060501   ASSIGN
0173            00000016200000000671    06060515060603051007   COND
0174            00000016300000000675    05160506051606060507   LABEL
0175            00000016310000000676    14150605020505061505   COND
0176            00000016330000000677    05030610070506051314   COND
0177            00000016560000000706    16060605070606051506   NT1
0200            00000017000000000715    06050305100205050513   IFCLAUSE
0201            00000017010000000716    02050505130706060515   TRUE
0202            00000017100000000722    02060605150605030510   FORSTMT
0203            00000017200000000727    10020505051307060605   FALSE
0204            00000017310000000734    05070606051506060305   NT1
0205                                    15050516050605160606   COND
0206                                    13141415060516050606   NT1
0207                                    05050512060602050505   IFCLAUSE
0210                                    06060515060603051002   TRUE
0211                                    05020506051606060516   UNCOND
0212                                    14150505070506061506   FALSE
0213                                    05030610020506051314   NT1
0214                                    16060605070606051506   COND
0215                                    05020517060205050613   NT1
0216                                    05030510020506051206   IFCLAUSE
0217                                    07060605020606051506   TRUE
0220                                    05070506060205050613   NT2
0221                                    14141414141414141505   UNCOND
0222                                    00000000000004141414   NT1
0223                                    05050512060511060501   FALSE
0224                                    05100705050512060507   STMT
0225                                    06050706060515060503   NT2
0226                                    00000000041414130206   FORSTMT
0227                                    05050512060511060501   LABEL
0230                                    05100705060512060507   FORSTMT
0231                                    06050706060515060503   FORSTMT
0232                                    03051007050505131606   CREADY
0233                                    06060507060605150605   AR
0234                                    00000000041414141302   AR
0235                                    05050512060511060501   FP
0236                                    05100705050512060507   CEP
0237                                    06050706060515060503   BODY
0240                                    03051007050505131606   A
0241                                    06060507060605150605   STMT
0242                                    00000000041414141302   EXITX
0243                                    05050512060511060501   FP
0244                                    06060515060503051007   CEP
0245                                    00000414130706060502   TP
0246                                    05050512050511060501   CTP
0247                                    06060515060503051007   A
0250                                    05160606051606060507   FORCLAUSE
0251            00000000270000000012    13120605030510020506   FORCLAUSE
0252            00000005250000000235    05030510070505051314   VAR
0253            00000007010000000321    16060605070606051506   CREFNDX
0254            00000012410000000504    14130206060516060605   AR
0255            00000012140000000471    00000000300000000414   AR
0256            00000015400000000640    05050512060511060501   LOCN
0257                                    06060515060503051007   TEMP
0260                                    05160606051606060507   INSNAR
0261                                    13120605030510020506   FP
0262                                    05030510070505051314   CEP
0263                                    16060605070606051506   AR
0264                                    05100205060516060605   AR
0265                                    05050513141312060503   FORLIST
0266                                    06060515060503051007   FORLIST
```

```
267                                    00000000041414141307    FORLIST
270                                    15060503051011060501    FORLISTELM
271                                    05050205050507060605    FORLIST
272                                    00000000000004131415    FORLISTELM
273                                    15050503051011060501    FORLISTELM
274                                    05050205050507060605    AREXP
275                                    12060507050505131415    FV
276                                    00000000000000000414    FVAL
277                                    05050512050511060501    VAL
300                                    00000000041412060507    TEMP
301                                    00000412050511060501    EP
302                                    15060503051011060501    CEP
303                                    05100205050507060605    TP
304                                    05100705050515060503    FV
305                                    04131413141312060503    TC
306                                                            FORINDEX
307                                    15050503051011060501    SIMPASG
310    00000017410000000740            06050206050507060605    LOCN
311    00000017430000000741            050503051 0305100205    TEMP
312    00000017450000000742            13141206050705050515    STK
313    00000017470000000743            00000413141307050506    FSTACK
314    00000017510000000744            05050512050511060501    FV
315    00000017530000000745            06060515050503051007    FVAL
316    00000017550000000746            07050505150706060516    TP
317    00000017570000000747            16060505150605030516    CTP
320    00000017610000000750            00000414141307060605    FP
321                                    05050512050511060501    CEP
322                                    06060515050503051007    VAL
323                                    07050505150706060516    FV
324                                    16050605150605030516    TC
325                                    00000414141307060605    FORBODY
326                                    15050512050511060501    FORLISTELM
327                                    00000000000000000414    T1
330                                    05050512050611060501    AREXP
331                                    06050705050612060507    FV
332                                    00000000000004141412    FVAL
333                                    00000412050511060501    VAL
334                                    00000412050511060501    TEMP
335                                    15050503051011060501    EP
336                                    06051606050507060605    CEP
337                                    05030510060506051606    TP
340                                    10070505051314131206    FV
341                                    05070606051506050305    TC
342                                    04141302050605070606    FORINDEX
343                                                            SIMPASG
344                                    00000412050511060501    LOCN
345                                    05050512050511060501    TEMP
346                                    06060515050503051007    STK
347                                    05160606051606060507    FSTACK
350                                    13120605030510020506    BOOLEXP
351                                    05030510070505051314    T1
352                                    07060605060606051506    GTRREL
353                                    14131606050507060605    BR
354                                    00000000000000000414    BRANCH
355                                    00000412050511060501    STK
356                                    05060512050611060501    FSTACK
357                                    06050705050612060607    TRUE
360                                    05061505050705050512    FV
361                                    00000000041414140205    FVAL
362                                    05050512050511060501    TP
363                                    06060515050503051007    CTP
364                                    00000414131606060507    FP
```

```
0365     05050512060511060501     CEP
0366     05100705060512060507     VAL
0367     14130706060515060503     FV
0370     00000000000000000414     TD
0371     05050512060511060501     FORBODY
0372     05100705060512060507     NT1
0373     14130706060515060503     T1
0374     00000000000000000414     FALSE
0375     00000412060511060501     FORLISTELM
0376     05050512060511060501     AREXP
0377     06060515060503051007     FV
0400     00000000000004141307     FVAL
0401     00000412060511060501     VAL
0402     05050512060511060501     TEMP
0403     06050705060512060507     FP
0404     00000000000004141412     CEP
0405     00000412060511060501     TP
0406     15060503051011060501     FV
0407     00000000041307060605     TD
0410     15060503051011060501     FORINDEX
0411     00000000041307060605     NO
0412     15060503051011060501     SIMPASG
0413     00000000041307060605     LOCN
0414     05050512060511060501     TEMP
0415     05100705050512060507     STK
0416     14130706060515060503     FSTACK
0417     00000000000000000414     ADD
0420     00000412060511060501     STK
0421     00000412060511060501     FSTACK
0422     05060512060611060501     STACKX
0423     06050705060612060607     VAL
0424     05061505050705050512     TEMP
0425     00000000041414140205     STK
0426     00000412060511060501     FSTACK
0427     05050512060511060501     FV
0430     05100705050512060507     FVAL
0431     14130706060515060503     VAL
0432     00000000000000000414     TEMP
0433     00000412060511060501     FP
0434     05050512060511060501     CEP
0435     05100705060512060507     TP
0436     14130706060515060503     FV
0437     00000000000000000414     TD
0440     00000412060511060501     FORINDEX
0441     05050512060511060501     NT1
0442     05100705050512060507     AREXP
0443     14130706060515060503     FV
0444     00000000000000000414     FVAL
0445     00000412060511060501     TP
0446     05050512060511060501     CTP
0447     05100705060512060507     FP
0450     14130706060515060503     CFP
0451     00000000000000000414     TD
0452     00000412060511060501     FORBODY
0453     05050512060511060501     FV
0454     06060515060503051007     VAL
0455     00000000000004141307     FALSE
0456     00000412060511060501     N1
0457     00000412060511060501     TSUEX
0460     00000412060511060501     HR
0461     05050512060511060501     BRANCH
0462     05100705050512060507     STK
```

```
0463     06050706060515060503     FSTACK
0464     14131206050305100205     NT1
0465     00000000000004141413     NT1
0466     05050512060511060501       0
0467     06060515060503051007     AREXP
0470     00000414130206060507     STACKX
0471     05050512060511060501     VAL
0472     06060515060503051007     TEMP
0473     00000414130206060507     STK
0474     15060503051011060501     FSTACK
0475     05100205060507060605     N0
0476     05100705050515060503     N1
0477     04131413141312060503     TRUE
0500                              ASSIGN
0501     05050512060511060501     LEFTPART
0502     06060515060503051007     ASSIGN
0503     00000414130206060507     STKASGN
0504     05050512060511060501     STK
0505     06060515060503051007     LSTACK
0506     00000414130206060507     VAL
0507     15060503051011060501     TEMP
0510     05100205060507060605     ASSIGN
0511     05100705050515060503     LEFTPART
0512     04131413141312060503     AREXP
0513                              UNSTACK
0514     15060503051011060501     VAL
0515     05100205060507060605     TEMP
0516     05100705050515060503     STK
0517     04131413141312060503     FSTACK
0520                              STKASGN
0521     15060503051011060501     STK
0522     06051606060507060605     LSTACK
0523     05030510020506051606     VAL
0524     00000000041314131206     TEMP
0525     00000412060511060501     ASSIGN
0526     15060503051011060501     LEFTPART
0527     05050205060507060605     BOOLEXP
0530     12060507050505131415     UNSTACK
0531     15060503051007050505     VAL
0532     06051606060507060605     TEMP
0533     05030510020506051606     STK
0534     10070505051314131206     FSTACK
0535     05070606051506050305     STKASGN
0536     00000414141413160606     STK
0537     05050512060511060501     LSTACK
0540     00000000041412060507     VAL
0541     00000412060511060501     TEMP
0542     00000412060511060501     LEFTPART
0543     15060503051011060501     VAR
0544     05100205060507060605     STACKX
0545     05050513141312060503     VAL
0546     00000000041412060507     TEMP
0547     15060503051011060501     STK
0550     05100205060507060605     LSTACK
0551     05050513141312060503     PROC
0552     00000000041412060507     ACTPARPRT
0553     15060503051011060501     FVAL
0554     06050706060502060605     VAL
0555     14131206050305100205     TEMP
0556     04141206050705050513     TP
0557                              CTP←
0560     15060503051011060501     FP
```

```
0561                    06050706060502060605      CEP
0562                    14131206050305100205      ID
0563                    00000000000000000413      PROCID
0564                    15060603051011060501      CALL
0565                    06050706060502060605      AR
0566                    10020506061505050205      AR
0567                    05131414131206050305      IP
0570                    00000414120605070505      CIP
0571                    15060603051011060501      FP
0572                    06050706060502060605      CEP
0573                    10020506061505050205      PROC
0574                    04131414131206050305      TEMP
0575                                              FUNCDESIG
0576                    15060603051011060501      ACTPARPRT
0577                    06050706060502060605      EVAL
0600                    06131206050305100205      VAL
0601                    13120605030510020506      TEMP
0602                    00000000000004131414      IP
0603                    05050512060511060501      CIP
0604                    15050603051003051007      FP
0605                    06131412050507050505      CEP
0606                    00000000041413070505      ID
0607                    03051003051011060501      PROCID
0610                    12060507050505150506      CALL
0611                    00000413070505061314      AR
0612                    00000412050511060501      AR
0613                    05050512050511060501      IP
0614                    06060515060603051007      CIP
0615                    05020606050206060507      FP
0616                    13120605030610020506      CEP
0617                    00000414131402060606      ID
0620                    15060603051011060501      PROCID
0621                                              CIPRVLN
0622                                              PROC
0623                                              TEMP
0624                                              ACTPARPRT
0625                                              STACKX
0626                                              STK
0627                                              PSTACK
0630                                              VAL
0631                                              ACTPARPRT
0632                                              STACKX
0633                                              STK
0634                                              PSTACK
0635                                              VAL
0636                                              ACTPARLIST
0637                                              ACTPARLIST
0640                                              ACTPARLIST
0641                                              ACTPARAM
0642                                              ACTPARLIST
0643                                              ACTPARAM
0644                                              ACTPARAM
0645                                              STACKX
0646                                              STK
0647                                              PSTACK
0650                                              VAL
0651                                              STRING
0652                                              STRING
0653                                              ACTPARAM
0654                                              CRNMPAR
0655                                              STK
0656                                              PSTACK
```