Published: 2/21/67
(Supersedes: BD.1.02, 9/7/66)

Identification

The Segment Symbol Table Produced by the PL/I Translator
D.B. Wagner

Purpose

Section BD.1.00 describes the general format for Segment Symbol Tables used in Multics. A number of points about the Segment Symbol Table are translator-dependent, and the present section describes the particulars for the PL/I Symbol Table.

Address Types

As noted in BD.1.00, address-types less than 512 are reserved for coordinated expansion of the standard list, and address-types greater than 512 may be used for address-types gername to only one translator. The following special address-type has been defined for the PL/I symbol table.

513 = address in procedure's internal static storage.

514 = returned value (address is parameter number).

Symbol Types

The following special symbol types have been defined for the PL/I symbol table.

513 = fixed, non-integer

514 = structure

515 = array other than as indicated in BD.1.00

516 = block

517 = programmer-named condition

518 = area

The Structure of the Table

The "root pointer" in the symbol table header points to a node whose associated information block has symbol-type equal to "external procedure". Branches run from this node to nodes for the entries in the procedure. The first of these nodes is the node for the "block" and contains branches to nodes for all the symbols defined in the outermost block.

Each symbol listed in the table has associated with it a node as described in BD.1.00. The pointers in the node to other entries in the symbol table are used as follows:

Type = 1: In the node for a block, these point to nodes for all symbols (including other block names) defined immediately internal to the block (i.e. internal to the block but not internal to any other block internal to it).

Type = 2: In the node for an entry name, these point to nodes for the parameters associated with the entry.

Type = 3: In the node for a data-structure, these point to nodes for substructures.

Type = 4: In the node for an array, this points to the node for a dummy entry with the data-description for the element of the array.

Type = 5: In the node for a variable with the controlled attribute this points to the node for the controlling pointer, if any was specified.

Type = 6: In the node for a pointer-variable, these point to nodes for any controlled variables which it controls.

Type = 7: In the node for an area, these point to dummy nodes giving the declarations specified in the area declaration.

Type = 8: In the node for a string, if the length is adjustable on a simple variable, this points to the symbol table node for that variable.

Type = 9: In the node for an array, if any lower bounds are adjustable on simple variables, these point to the symbol-table nodes for those variables.

Type =10: In the node for an array, if any upper bounds are adjustable on simple variables, these point to the symbol table nodes for those variables.

Type =11: In the node for an external procedure in which the *validate* option was specified, this points to a symbol table entry for the validating procedure.

Type =12: In the node for an external procedure for which the *callback* option was specified, these point to nodes for all the entries specified in the option.

Type =13: In the node for a block, these point to nodes for all symbols for which the *check* condition is enabled in the block.

Type =14: In the node for an entry for which the *sets* attribute was declared, these point to all the variables specified in the *sets* attribute.

Type =15: In the node for an entry for which the *uses* attribute was declared, these point to all the variables specified in the *uses* attribute.

Type =16: In the node for an internal or external procedure, these point to the nodes for the entries into the procedure.

Type =17: In the node for an internal or external procedure, this points to node for a block, as described later in "overall structure".

Type =18: In the node for an entry this points to a dummy node describing the returned value for the entry.

The Information Block

What follows is a list of the items in the information block, in the order in which they appear in the information block map.

Required items

1.	symbol type	
2.	address	address of specifier if the symbol-type normally takes one; address of data otherwise
3.	address type	

4. length of name in bits

5. name

Items needed for all variables

6. storage class
1 = automatic
2 = internal static
3 = external static
4 = controlled, based
5 = controlled, non-based

7. set
2 bits: 10 = set by program
01 = not set by program
00 = unknown

8. read
2 bits: 10 = read by program
01 = not read by program
00 = unknown

9. check condition somewhere enabled
1 bit: 1 means that somewhere the condition is enabled for the symbol.

Items needed for variables which normally take specifiers

10. data address
11. data address type
12. dope address
13. dope address type
14. free address — only needed for varying strings.
15. free address type

Items needed for the various string types

18. adjustable
1 bit: if 1 then next item is meaningless.

19. length or maximum length

1-bit items giving enabling status for various conditions (if symbol type = "block")

20. interrupt
21. overflow
22. zerodivide
23. fixedoverflow
24. conversion
25. size
26. subscriptrange

16. Precision
17. Scale

Item needed for all aggregates

27. packed	1 bit: 0 = aligned 1 = packed

Item needed for arrays

28. dimension information pointer	18 bits: either zero or the address relative to the symbol table header of a dope vector giving all declared dimensions and indicating ± infinity for adjustable dimensions.

1-bit items indicating options specified for "external procedure" symbol type

29. mastermode	
30. validate	
31. callback	
32. rename	Information concerning what was renamed is lost.

Other items related to the "external procedure" symbol type

33. free storage address	address of default area for allocations
34. free storage address type	probably = linkage address
35. internal static address	address of a block of storage in which all internal static data is kept.
36. internal static address type	probably = linkage address

Items related to the <u>picture</u> attribute

37. picture exists for variable	1 bit
38. picture address	
39. picture address type	

40. abnormal — 1 bit: indicates abnormal attribute was specified.

41. irreducible — 1 bit: indicates irreducible attribute was specified.

42. initial — 1 bit: indicates the initial attribute was specified.