Published: 03/04/67

Identification

Global Strategies in EPL
D. B. Wagner

Purpose

By "global strategies" in EPL we mean the details of implementation which affect communication between separately compiled procedures, and also those details which debugging aids and run-time library procedures depend upon for correct operation. Most of these details are the same as in PL/I and are contained in the BP. Sections of MSPM, especially BP.2.01, BP.2.02, and BP.4.00.

The Sections BN.5.00 - BN.5.02 give the details of EPL implementation which are for one reason or another missing from the BP. Sections or which are different in EPL and PL/I. The present Section gives details of several minor differences in global strategy and gives references to Sections which describe major differences.

Double Fixed-Point Numbers

BP.0.01 states that "the length of the largest number in the implementation" (the number N discussed on p.32 of the "-3" manual) is 71. EPL instead takes this number to be 63.

The result of a fixed-point division is always floating-point, since EPL cannot support the scaled result required by PL/I ("-3", p.32).

The above two points are language issues, not strategy issues. EPL programs will have no difficulty communicating fixed-point numbers with programs compiled by other versions of PL/I in Multics, since fixed-point numbers in EPL are represented as specified in BP.2.01. The error comment from EPL's Pass 2, "EPL double fixed incompatible with PL/I", is just noise.

Label Variables

Section BP.2.01 notes that label variables are six words long, the last two words containing "error-checking information" which is not yet specified. EPL labels are six words long also, but the last two words contain garbage. [This is a bug: there should at least be an indication that no error-checking information exists. However it is not at all clear what ought to go in those two words so there is no point to worrying about it.]

Implementation of Storage Classes

The implementation of storage classes is important to debugging aids and also to data-directed input when and if that becomes a reality. Section BP.4.00 describes storage classes in PL/I and everything there applies to EPL except for the description of internal static storage. Since the fancy mechanism of "unique identifiers" is not available in CTSS and on the GE 635, the internal static storage for an external procedure a is contained in a block of storage at

<stat_>|[a]

which is grown at first reference.

The observant reader will notice that there is some danger of naming conflicts here, in particular of some other external procedure using an external variable named a. [This can be lived with simply because external variables are seldom used in Multics development. However, there is a fix which is so simple that it really ought to be done: the name for the in-reference into stat_ should be formed by concatenating a "." with the segment name.]

Blocks, On-conditions, and the Non-local Go To

The implementation of blocks in PL/I has not yet been documented. See BN.5.01 for blocks in EPL.

The implementation of on-conditions in PL/I will use Multics primitives which are not yet available. The present EPL implementation is described in BN.5.02.

The present EPL implementation of the non-local go to is described in BN.5.01. It will be unsatisfactory as soon as the protection mechanisms begin to be used in Multics.

Varying Strings

Only the "long" form of varying strings is used in EPL. See BP.2.01.

Non-Standard Specifiers

The specifiers for string constants in EPL do not use its pairs as specified in BP.2.02 but instead use arg indirect words. Thus a procedure receiving a non-varying string as an argument must always access its specifier through indirect references.