Published: 04/27/67

Identification

Dope vector builder for adjustable aggregates
tdope_
D. B. Wagner and N. I. Morris

Purpose

When the declaration of an EPL aggregate contains an expression instead of a simple constant for some "extent" (a string length or array bound), it is said to be adjustable. The aggregates declared below, for example, are adjustable:

```
dcl a(n) floating;

dcl 1 b ctl(p),
      2 m fixed,
      2 q char (p->b.m);
```

The dope vector (see BP.2.02) for an adjustable aggregate cannot be assembled into the procedure segment as other dope vectors can. Instead the dope vector must be built up at execution time. Dope for automatic adjustable aggregates is generated at block entry and dope for based adjustable aggregates is built at each reference. The procedure tdope_ is called by EPL-compiled programs to help in the task of building these dope vectors.

Usage

At execution time, the EPL-compiled code generates a skeleton dope vector containing only the following information:

1) All identity codes, both in "substructure pointer words" and in "breakdowns".

2) All extents, computed when necessary from the user's declaration.

The compiled code then calls tdope_ as follows:

```
call tdope_ (dope, size);
```

where dope is the skeleton dope vector and size is a word into which tdope_ will store the size in words of the aggregate.

Tdope_ goes through the dope vector and fills in the following items:

1) all offsets
2) all multipliers in array dope
3) the "size" word in structure dope.

The rules used in determining these items are those given in BP.2.01 and repeated below:

1) Array

   An array is stored contiguously in row major order (right-most subscript varying most rapidly). An array of structures is a repetition of structures. If the elementary data item of an array occupies more than one word, then the elementary data items always begin on even word boundaries.

2) Structures

   The elements of a structure are stored contiguously in the order of their declaration. If an element of a structure is itself an aggregate, then all the members of that aggregate taken together constitute the storage for that element. If the origin of a structure requires an even word boundary (i.e., the structure is an elementary data item which occupies more than one word), then the origins of all containing structures are also even. A maximal structure of character or bit class (but not mixed) always begins at a word boundary. For any two consecutive items in a packed structure, which are not character or bit class, the second item starts at the next boundary natural to that element type. In aligned structure every item starts on a word boundary.

Implementation

Tdope_ tests the identifier bits in the dope vector and transfers to one of three internal sections: structure, string, or array. If, while processing the dope for a structure, a pointer to substructure dope is discovered, tdope_ will call itself recursively via one of two internal entries: tdope_1 and tdope_2.

When tdope_, working on the dope for a structure, discovers that some substructure or elementary data item must be at an even location, it makes sure that the offset of

that substructure or elementary data item is an even number of words. In addition it sets a flag to be returned in the case of a recursive call. This signals tdope_ that a substructure must be at an even location.

Similarly when tdope_, processing the dope for an array, discovers that the elementary data item of the array requires an even location, it sets the even flag.

It should be noted that this use of the even flag is needed only for subaggregates (that is, when tdope_ calls itself recursively), since EPL always places major aggregates in even locations.

Saving Cycles

Tdope_ is organized so that a simpler time- and space-saving call can be made to it. It is the following, where .u0 is a four-word "utility" space in the current stack frame:

```
eapbp   (dope)

stpbp   sp|.u0

eapbp   sp|.u0 +2

stpbp   sp|.u0 +2

call <tdope_>|[tdope_] (sp|.u0-2)
```

Here the argument list has a garbage header, and the second argument pointer points to itself and will be clobbered when tdope_ stores the aggregate size into it.