To: J3/INTERP
Subject: Interpretation request on the TARGET attribute
From: Malcolm Cohen.
Date: 2002-08-13

NUMBER: 000100
TITLE: TARGET attribute and structure components
KEYWORDS: TARGET, POINTER
DEFECT TYPE:
STATUS: J3 consideration in progress

QUESTION:

The standard says ([110:43-44]), of R737 <pointer-assignment-stmt>,
  "Constraint: The <variable> shall have the TARGET attribute or be a subobject
               of an object with the TARGET attribute, or it shall have the
               POINTER attribute."

It also says ([75:22-23]), of structure components
  "A structure component has the ... TARGET ... attribute if the parent
   object has the attribute."

The <<parent object>> is defined ([75:16]) as
   "the data object whose name is the leftmost part name".

Consider:
```
  TYPE t
    REAL x
  END TYPE
  TYPE(t),POINTER :: p
  REAL,POINTER :: q
  ALLOCATE(p)
  q => p%x
```

Q1. Is this standard-conforming?

If the answer to Q1 is Yes, this still appears to leave unspecified the case of A%B%C where A does not have the TARGET attribute and B has the POINTER attribute. The parent object of A%B%C is just A, and A does not have the TARGET attribute.

Therefore the conclusion is that A%B%C does not have the TARGET attribute.

Note: Although the standard says that "Allocation of a pointer creates an object that implicitly has the TARGET attribute" [81:47-48], is it not entirely sure that this shows that A%B%C has the attribute for two reasons:
(1) Allocation is not the only way of establishing pointer associations.
(2) According to the quoted text, whether A%B refers to an object with the TARGET attribute is immaterial; the only question is whether A itself has the TARGET attribute.

Q2. Does this A%B%C have the TARGET attribute?

Note: the following example may be relevant:

```
  TYPE t
    TYPE(t2),POINTER :: p
```

```
  END TYPE
  TYPE t2
    REAL x
  END TYPE
  TYPE(t) v
  TYPE(t2),POINTER :: z
  REAL,POINTER :: q
  ALLOCATE(v%p)
  !
  ! (a)
  !
  z => v%p
  q => z%x
  !
  ! (b)
  !
  q => v%p%x
```

The pointer assignments following (a) are clearly valid: the first assigns one pointer to another, and if the answer to Q1 is Yes, the second one follows from that.

But the (a) statements have the same effect as the (b) statement, so it would be inconsistent to disallow (b).

Q3. Apparently, substrings never have the INTENT, TARGET or PARAMETER attribute. Is this a deliberate omission?

ANSWER:

Q1: Yes. P points to an object with the TARGET attribute, and X is a subobject of this object. Therefore, "Q => P%X" is conforming.

Q2: The standard leaves this case unspecified, but the intent is that any part of an object with the TARGET attribute should itself have the attribute. An edit is supplied to remedy this omission in the standard.

Q3: No, this omission was inadvertant. An edit is supplied to fix this.

EDITS:

[74:33] Append to paragraph
"A substring has the INTENT, TARGET, or PARAMETER attribute if its parent has the attribute, but it never has the POINTER attribute."
{Fix omission. Text parallels that for array elements and sections.}

[75:22] Delete ", TARGET,".

[75:23] After "." insert
"A structure component has the TARGET attribute if the parent object has the attribute or if any <part-ref> except the rightmost has the POINTER attribute."

[110:43-44] Delete
"be a subobject of an object with the TARGET attribute or it shall have".
{Now that propagation of TARGET is fixed, we don't need the weasel words.}

SUBMITTED BY: Malcolm Cohen.

HISTORY: 02-233 m162 Submitted

02-233r1 m162 Revised, passed by J3 meeting.