Published: 11/09/67

Identification

Forcing a link
D. L. Boyd and D. H. Johnson

Purpose

This procedure, link_fault$force, forces an existing link to be set in a linkage section. A link pair with a fault tag 2 (ft2 or fi) modifier is replaced by an appropriate ITS pair. There are restrictions on traps before linking that are described below.

Introduction

This procedure is an entry in the link_fault segment (i.e., the linker, see BD.7.04). The information is duplicated here in order to have all user linkage maintenance routines in one section of the MSPM.

Usage

The call to force a link is

```
     link_fault$force (pointlp,option,bases);
     dcl pointlp ptr, bases (8) bit (36), option bin(17);
```

The arguments are:

1.	pointlp	pointer to a link pair (in some linkage section) which is to be forced.
2.	option	if = 0, ignore trap before link. if = 1, allow trap before link.
3.	bases	bases to use for itb type link.

Argument one, <u>pointlp</u>, points to a link pair in some linkage section. If the link has not been set (i.e., the word pair pointed to by <u>pointlp</u> contains an ft2 modifier), then link_fault$force will set the link (i.e., replace the word pair containing the ft2 modifier with an appropriate ITS pair). If the link is already set (i.e., the word pair pointed to by <u>pointlp</u> contains an ITS pair), link_fault does nothing.

Argument two, <u>option</u>, is a switch that allows a trap, or call, before link. A trap before link means that construction of the link is suspended while another procedure is executed. If argument two, <u>option</u>, equals 1, a request to trap before link is allowed. If <u>option</u> equals 0, a request to trap before link is ignored.

This option allows users to effectively eliminate traps in the linkage section without modifying the linkage information. However, the user of link_fault$force must be cautious about allowing traps. As an example, if a procedure calls link_fault$force to set the link that was left waiting by the original trap, an infinite loop will have been created.

The third argument, bases, contains the contents of the base registers necessary for an ITB external reference. This argument may be omitted if the reference is not an ITB. If the bases are needed and they are zero or not given, it is an error.

The link_fault procedure checks the arguments given. The argument-validating procedure, validate_arg (BD.9.03), is called to see if the procedure calling link_fault at entry [force] is allowed to reference the segment containing the link to be forced as well as the segment containing the bases, if given.

When an error is detected, seterr (BY.11.01) is called to put identifying information in <error_out>. The condition "link_fault_err" is then signaled. The following errors are detected:

Error Number	Meaning
11	Tried to trap before link or definition with call pointer equal to 0.
12	Illegal external reference type code.
13	Fault occurred in a linkage section with no link definitions.
14	External symbol definition not found in linkage section.
15	Segment not found.
21	The second argument option, in the call to [force] was undefined.
22	Bases needed and not supplied or incorrect in call to entry [force].
31	Link not set. Illegal ring access involved in arguments of call to [force].
41	The scu data (machine conditions) were not valid.