Published: 04/04/67

Identification

Pre-link Module
D. H. Johnson

Purpose

This section describes the pre-link module which is executed during Multics system initialization. It is used to link all external references in the hard-core supervisor. Pre-linking avoids the necessity of having a linker operating in the hard-core supervisor during Multics execution. It allows hard-core supervisor linkage sections to be shared system wide without having to contend with the interlock problem which would exist if a linker wrote into the linkage sections. Another purpose of the pre-link module is to combine and condense linkage section information for hard-core supervisor segments. This function of the pre-link module reduces both the number and the size of the linkage section segments in the template descriptor. This section assumes that the reader is familiar with the Bootstrap Initializer (MSPM BL.4), the Initializer Control Program (MSPM BL.5.01), the segment loading table (MSPM BL.2), the initialization linker (MSPM BL.7.01), and linkage section structure (MSPM BD.7).

Introduction

The pre-link module has the responsibility during Multics initialization to see to it that hard-core supervisor linkage is properly arranged and that all linkage paths are completed. The pre-link module is assisted in its job by the initialization linker and the initialization data segment grower, datmk (MSPM BL.7.03). More specifically, the tasks of the pre-link module are:

1. Most hard-core supervisor linkage sections are combined into several special linkage segments.

2. Link pairs in the hard-core supervisor linkage sections are changed from faults to correct machine addresses.

3. Linkages to undefined segments are redirected through a special linkage segment that may be referenced later by the Multics linker.

There are two entries in the pre-link module to accomplish these three tasks. The first entry, which does tasks 1 and 2, is

called several times by the Initializer Control Program. Each time this entry is called the hard-core supervisor linkage segments that have been loaded by the segment loader (MSPM BL.6.01) since the last call are considered for combining into either a wired down, loaded, or active special linkage segment. Then all of the currently loaded hard-core supervisor linkage sections are examined and linkage faults are changed, if possible, to correct machine addresses. References to undefined segments are skipped. The second entry to the pre-link module, which does task 3 above, is called once by the Initializer Control Program after all hard-core supervisor segments have been loaded and linked. The procedure invoked at this entry examines all of the pre-linked linkage segments. All link pairs that still contain faults are considered to be references to segments outside the hard-core supervisor. These linkages are placed into a special out reference linkage segment which is not in the hard-core ring of protection. An indirect machine address pointing to the moved linkage is placed in the link pair. When all of the pre-linked linkage segments have been examined, all linkage faults have been eliminated from the hard-core supervisor.

When the pre-link module is used, Multics initialization is in the following state.

1. The hard-core supervisor segments that the pre-linker must reference have been loaded from the Multics System Tape. The referenced segments include text segments, as well as linkage section segments, if pure linkage definition information has been stored at the end of the text segments by the programming language translators. The definition pointers in the headers of the referenced linkage sections have been set by the segment loader.

2. The SLT contains entries for linkage as well as text segments. Each SLT entry contains linkage information items that are needed by the pre-link module. Some of the items are set from information contained on the system tape while others are determined during initialization. Linkage segments have names which conform to Multics standards.

3. SLT entries exist for the special linkage segments that are used by the pre-linker. These segments are initially empty and their descriptor segment entries may initially contain missing segment faults. The core management procedures, mentioned below, are responsible for providing memory for these segments. There are four special linkage segments. Their names and descriptions are given below.

a. wired_hcs.link

This linkage segment is used to copy headers, link pairs, and entries contained in all combinable wired down linkage sections for the hard-core supervisor. Entries are machine instructions necessary for transferring control from one segment to another (MSPM BD.7.02-.03). This linkage segment will contain one linkage section for each linkage section copied. It is a per-system segment in the hard-core ring.

b. loaded_hcs.link

Same as wired down except for loaded (in the Multics sense) hard-core supervisor segments.

c. active_hcs.link

Same as wired down except for active hard-core supervisor segments.

d. out_hcs.link

This linkage segment is used to re-build all links and link definitions for references from the hard-core supervisor to segments residing in other rings of protection. It is a per-process segment which will be accessible by the Multics linker in ring 1. It will contain one linkage section.

4. A core manager is functioning which will handle missing segment and missing page faults. For the first part of initialization this is done by the Initializing Core Manager (MSPM BL.6.03). Later the Multics file system takes over this function.

5. The initialization linker (MSPM BL.7.01) is designed to set links in hard-core supervisor linkage sections as well as Multics Initializer linkage sections. The linker is callable by the pre-link module to set links.

Implementation

The pre-link module has two entry points. The first is invoked by:

```
call pre_linker$combine_set ;
```

This procedure combines and links hard-core supervisor linkage sections. The method described below is used.

1. A pass is made through the hard-core supervisor entries in the SLT to enter combinable linkage sections into the appropriate special combined linkage segment. The range of segment numbers currently assigned to the hard-core supervisor is kept at the beginning of the SLT. Each SLT entry is processed as follows:

a. If the linkage segment provided switch is OFF for the entry, the segment does not have a linkage section. The entry is skipped and the next entry is tested.

b. The entry represents a hard-core supervisor segment that has a linkage section.

   If the pre-linked switch is ON for the entry, this segment has been pre-linked by a previous call to the pre_linker$combine_set procedure- The entry is skipped and control goes to step a. to test the next entry.

c- The pre-linked switch for the entry is set ON.

   If the combine-linkage switch is OFF for the entry, the linkage section for this segment is not combinable and should be pre-linked where it currently resides. The entry is skipped and control goes to step a. to test the next entry.

d. The linkage section is combinable. The text-linkage segment number item in this entry contains the segment number of the linkage section. A linkage block is appended to the appropriate combined linkage segment for each linkage block in the original linkage section. The definition pointer, the links, and the entries are copied from the original block(s) into the new linkage block(s). The definitions are not copied. The structure of pointers in linkage sections allows this information to be copied without change. (See Comment 2 below.) The particular combined segment is determined by the segment status item in the SLT entry. It must be either wired down, loaded, or active. The pre-linker maintains pointers to the next available location in each of the combined segments. The combined-linkage section segment number and combined linkage offset items in the SLT entry are set. Control then goes to step a. for the next entry.

2. When the pass through the SLT described in paragraph 1 is finished, all newly loaded linkage sections have been combined, if requested, into their final segment residence. The pre-linker now makes a pass through the special combined (wired down, loaded, and active) and the non-combinable linkage segments to change faults in link pairs to machine addresses. The non-combinable linkage sections are found by searching for hard-core supervisor SLT entries with the following characteristics.

- Linkage segment provided switch - ON
- Pre-linked switch - ON
- Combine-linkage switch - OFF

The segment number of the non-combinable linkage section is contained in the text-linkage segment number item.

Each linkage segment is processed as follows:

a. Every word located at an even offset is tested for an fi modifier.

   If the tag field of the word is not an fi modifier, the word is skipped and the next even word is tested.

b. The existence of an fi modifier indicates that the word examined is the first of a two word link pair which has not been set.

   The pre-linker calls the initialization linker to place the correct machine address in the link pair. The initialization linker is called just as if the hardware had detected the fault while executing the word, i.e., the fault interceptor had been invoked (MSPM BL.5.02). There is only one restriction placed on the linker by the pre-linker. Type 2 external references (ITB - indirect to base) are not permitted since the pre-linker does not know the correct base address register values to pass to the linker. The trap before link feature is permitted.

   The machine conditions passed to the linker are fabricated so that it appears that a linkage fault actually occurred. However, only an incomplete set of machine conditions is passed.

   The linker may return control to an error routine with one of several possible error codes. If the error indicates that the segment referenced is not loaded, the error is ignored and control goes to step a. above to test the next even word. The link pair has not been changed. It may be set on a following call to pre_linker$combine_set if the segment referenced is loaded in the interim. All other errors cause initialization to stop.

   If the normal return from the linker is received, the link pair has been changed to a correct machine address. Control goes to step a. to test the next even word.

The second entry to the pre-link module is given control by:

```
call pre_linker$redirect ;
```

This procedure arranges the linkage for references from the hard-core supervisor to segments in other protection rings. The steps are described below.

A pass is made through the special combined (wired down, loaded, and active) and the non-combinable hard-core supervisor linkage sections to redirect all link pairs containing linkage faults to the special combined linkage segment for out references. All non-combinable linkage segments are found as described in paragraph 2 under procedure pre_linker$combine_set.

Each linkage segment is processed as follows:

a. Every word located at an even offset is tested for an fi modifier.

   If the tag field of the word is not an fi modifier, the word is skipped and the next even word is tested.

b. If an fi modifier is found, the word tested and the next word contain pointers to the linkage definition. The link pair and the entire linkage definition for the reference are copied into the next available locations of segment out_hcs.link. As linkage information is placed into this combined linkage segment, relative pointers must be re-computed. An ITS pair with an indirect modifier is placed in the original link pair to point to the unlinked link pair constructed in out_hcs.link.

   Control goes to step a. to test the next even location.

Comments on the pre-linker

1. The strategy of placing hard-core linkage in its final resting place before linking is important. The trap before link and trap before definition features make it possible for the hard-core supervisor to obtain control during pre-linking. If, while the hard-core supervisor is executing, a linkage fault is recognized by the hardware, the linker will then be able to dynamically link the reference using the correct linkage segment. However, if the hard-core supervisor references a segment that has not yet been loaded when it is executing, the linker is forced to terminate initialization.

2. The pre-linker assumes that the hard-core supervisor linkage sections are structured in the standard manner; i.e., each block contains information in the following order:

   1. header
   2. links and entries
   3. definitions (These may be in some other segment.)

The information to be copied from the original linkage section to the combined linkage segment is contained in the part of the linkage block(s) immediately following the header(s). If the definition pointer is not pointing into the linkage block, the entire block will be copied. If the definition pointer is pointing into the linkage block, only the information between the header and the location indicated by the definition pointer will be copied. If the copied linkage section contains more than one linkage block the next block and previous block pointers will have to be re-computed.

3. Any linkage building procedures used during initialization on behalf of the hard-core supervisor should take into consideration the linkage segment arrangement outlined in this document. Linkage constructed during initialization for combinable hard-core supervisor segments should be placed in the proper combined linkage segment using the pointer to the next available location. The linkage building procedures are also responsible for updating these pointers. If possible, linkage information should be added for a segment using the segment's linkage block which was created by the pre-linker. If a new linkage block is constructed in the combined segment, that block should be chained to the one created by the pre-linker.

4. The various linkage sections combined into the special linkage segments are independent. In particular, the linker is not aware that the linkage section it is examining may be part of a combined linkage segment.

5. The combinable hard-core supervisor linkage section segments should be listed on the system tape as initialization segments so that they will be assigned segment numbers in the initialization portion of the descriptor. These segments, which are not needed after initialization, will not be in the template.