TO: MSPM Distribution
FROM: T. H. Van Vleck
SUBJECT: MSPM BL.4.02
DATE: 02/29/68

This section has been revised to reflect the actual implementation.

Published: 02/28/68
(Supersedes: BL.4.02, 03/27/67)

Identification

Bootstrap 2
T. H. Van Vleck

Purpose

Bootstrap 2 receives control from bootstrap 1 at location zero after all of collection 1 has been loaded. Its purpose is to initialize the stacks, the SLT manager, and the FIM, and to call the pre-linker to pre-link collection 1. Bootstrap 2 terminates with a standard call to the Initializer.

Bootstrap 2 executes in slave mode and is impure.

When bootstrap 1 gives control to bootstrap 2, the bases are paired, and bases SP-SB point to the stack. At this time,

X1 = segment # of SLT manager
X2 = segment # of SLT
X3 = processor tag

The following steps are executed:

1. Set base register LP-LB to point to the linkage section of bootstrap 2. It is assumed that the segment number of the linkage section is one greater than the segment number of bootstrap 2.

2. Initialize the ring 0 stack by doing a standard SAVE. Also set <stack>|0 to point to the beginning of the stack at <stack>|8.

3. Initialize the SLT manager by calling <slt_manager>|0 with a pointer to the SLT.

4. Call the SLT manager at <slt_manager>|2 to get a pointer to the segment <pre_link_1>.

5. Call the pre-linker to pre-link collection 1.

6. Initialize the FIM, which must have pointers to the PDS, the PRDS, and its own linkage section. Pointers are generated by bootstrap 2 and stored into the FIM.

7. Initialize the PDS and PRDS.
The six quantities

pds$stb_pointer	prds$stb_pointer
pds$sreg_pointer	prds$sreg_pointer
pds$scu_pointer	prds$scu_pointer

are computed and stored in the respective segments.

8. Initialize the PDF and the fault-stack contained in it by setting location <pdf>|0 and making a dummy stack frame at the top of the fault stack.

9. Initialize the ITS pairs in the fault vector so that all interrupts are sent to segment <ignore> and all faults except directed fault 0 and timer runout are sent to segment <stop>. Directed fault 0 will be sent to the FIM and timer runout to <ignore>.

10. Change the SDW for the FIM, which has been "data, slvacc, wpermt" so that we could store pointers into it, to "masprc, slvacc" so that it can work.

11. Set up the segment <initialization_constants>. The following data items are set.

    bootload_cpu_ptr
    bootload_gioc_ptr
    bootload_gioc_port
    bootload_cpu_tag

12. Zero the SDW for bootstrap 1. This must be done because bootstrap 1 lies within the mailboxes and core control will call PANIC if segments overlap.

13. Call the Initializer.