2 CP disaster sheet

1) processor A is making a map entry, finds a block missing; processor B creates the block;
processor A decrements map counts on blocks & garbage up newly created block & any that follow.

2) processor A is making a process, finds all necessary blocks for map entry present; processor B deletes one of the blocks; processor A goes to disaster.
 Freed phrasing, scheduling, compactification. ECS code, DAE, + other theological questions. SF's are supposed to give damn fast responses, like for a real-time something or another. Thus, when an SF is \{awakened\} \{interrupted\}, the scheduler has to do something snappy. The current guy has to be suspended + the SF freed up. Some part time \textit{fjedl}. Normally, they need real enuf of the CE to accommodate the SF could be copied by brute force to ECS. The SF brought in + run. The CE then restored + allowed to run. A know of some kinda:

1) The CE may be in the middle of a system call which is executing ECS code in some buffer. The SF may nuke out the buffer. It seems like the contents of the buffers have somehox to be preserved.

2) ECS may be all bent' cause compactification is in progress. The compactifier has to be told to cool it. It might require some piece of time \( t_2 \) to get itself straight. In this case, it must...
be incremental.

3) The allocator isn't really reentrant.
The fight between $k_1$ and $k_2$ is decisive. Two some things should be noted:

1) The SF's map may have to be recompiled.

2) The compactifier may be moving something gaudy toward. Either
   a) it is allowed to finish
   b) a mechanism for half-moving something had to be think up.

3) The SF better hadn't cause anything to be allocated (even destroyed is annoying).

4) If there's more than 1 SF, things get complicated fast.

5) How long can I LOCK remain set?
Details of initiating a SF

A process may be fired up by:

1) getting an event
2) receiving an interrupt
3) being created.

For now, we consider only 1 (we'll include 2+3 later if it fails out of it we're forced to).

A) The interrupt code calls the event code calls the scheduler. The scheduler detects that it's a SF awakening.

B) The scheduler may have to determine what was interrupted:

1) user
2) system

so as to get into (out of) monitor mode correctly.