

SRM Status Errors

Atomic instruction effect

Wed, Feb 5, 2003

There are occasional SRM errors that appear in the Linac alarm log report every day. Is it possible that they are caused by arcnet interrupt processing that occurs by chance during a task-level atomic operation, such as setting a bit in memory or advancing a counter? Arcnet communications with SRMs occurs at 15 Hz, which means it happens a million times a day, which gives a large number of chances to “perform the experiment that could rarely result in an interrupt occurring at just the wrong time.

This is actually a common potential problem with the system code based upon the PowerPC, which does not have atomic instructions, as compared with code that was originally written for the 68K processor, which does have atomic instructions.

Look for task-level code “atomic” logic that could be affected by arcnet interrupt processing.

What was the understanding coming out of analyzing the `QMonitor` logic in `OUTPQMON` that incremented the `USED` byte to effect time-out logic? The ethernet transmit interrupt code is to set the sign bit of that byte. Is there analogous logic relating to arcnet?

Not all systems exhibit SRM alarm messages to the same degree. `Node0625` and `node0626` are popular, as are `node062C` (`DIASRM`) and `node062F` (`DISRM`), if memory serves. Is there something about how these systems are set up that is different from others? (`Node0610` is also popular, but it is near a source of sparks.) Some system issue a broadcast request message, and others use individual requests to each SRM, which was thought to be more reliable, since a broadcast message does not get the built-in “free buffer enquiry” logic. Can it be that an atomic operation problem actually makes it worse when using individual SRM requests?

Here is a list of the number of SRM requests in the `RDATA` table for each Linac node:

610	BC	620	BC	62C	BC
611	3	621	BC	62D	3
612	1	622	BC	62E	BC
613	1	623	BC	62F	4
614	2	624	3		
615	1	625	BC		
617	BC	626	4		
61C	2	627	BC		
61E	2				