okay, here are some hints for debugging the lower-level parts of linux/parisc.

1. Absolute addresses

A lot of the assembly code currently runs in real mode, which means absolute addresses are used instead of virtual addresses as in the rest of the kernel. To translate an absolute address to a virtual address you can lookup in System.map, add __PAGE_OFFSET (0x10000000 currently).

2. HPMCs

When real-mode code tries to access non-existent memory, you'll get an HPMC instead of a kernel oops. To debug an HPMC, try to find the System Responder/Requestor addresses. The System Requestor address should match (one of the) processor HPAs (high addresses in the I/O range); the System Responder address is the address real-mode code tried to access.

Typical values for the System Responder address are addresses larger than __PAGE_OFFSET (0x10000000) which mean a virtual address didn't get translated to a physical address before real-mode code tried to access it.

3. Q bit fun

Certain, very critical code has to clear the Q bit in the PSW. What happens when the Q bit is cleared is the CPU does not update the registers interruption handlers read to find out where the machine was interrupted - so if you get an interruption between the instruction that clears the Q bit and the RFI that sets it again you don't know where exactly it happened. If you're lucky the IAOQ will point to the instrucion that cleared the Q bit, if you're not it points anywhere at all. Usually Q bit problems will show themselves in unexplainable system hangs or running off the end of physical memory.