Common Causes of QNX Neutrino Kernel Crashes#

If you see a QNX Neutrino kernel crash, chances are it's caused by one of the following. While this is by no means a complete list, the following are the high runner cases.

- segfault/sigbus in an interrupt handler

- incorrectly written callouts

- Operations that cause a machine check or other types of asynchronous faults in the system. If you're unlucky the fault gets "noticed" by the CPU when we're executing in the kernel crash. If a user-land thread was active at the time, a SIGBUS to the task is the usual end result.

- If there are hardware issues, high load can elicit crashes

- Memory controller configuration issues can cause random crashes

- low memory conditions can be hazardous. These generally would be failed assertions (ie. cannot allocate) which should not have failed.

- Having a JTAG attached to the board (not consistent from board to board/JTAG to JTAG) when jumping from startup into the OS (or any time after).

- A privileged thread steps on hardware that procnto cares about (the interrupt controller(s), the timer providing the clock, the memory controller, the MMU, etc.)

- A root process does a direct physical mapping (MAP_PHYS) to memory it shouldn't and overwrites important bits.

- A privileged thread programs a DMA engine to scribble over memory it shouldn't.

- Startup lies about something on the system page (e.g. where memory is, how much memory is in the system, what the capabilities of the CPU are, etc).