

## Re: [PATCH 2.6.16] Shared interrupts sometimes lost

---

*Source:* <http://linux.derkeiler.com/Mailing-Lists/Kernel/2006-04/msg01169.html>

---

- *From:* Neil Brown <neilb@xxxxxxx>
  - *Date:* Sun, 9 Apr 2006 16:02:34 +1000
- 

On Saturday April 8, rlrevell@xxxxxxxxxxxx wrote:

On Sat, 2006-04-08 at 14:10 +1000, Neil Brown wrote:

To explain what I think is happening, let me start with a very simple case. A number of PCI devices (this one included) have a number of events which can trigger an interrupt. The events which are current are presented as bits in a register, and are ORed together (and possibly masked by another register) to make the IRQ line. When 1's are written to any bits in this register, it acknowledges the event and clears the bit. A typical code fragment is

```
events = read_register(INTERRUPTS);
write_register(INTERRUPTS, events);
... handle each 1 bits in events ....
```

Isn't a more typical IRQ handler:

```
while (events = read_register(INTERRUPTS) != 0) {
...handle each bit in events and ACK it...
}
```

Maybe... I admit that I generalised from 2 examples: rt2500 and yenta. However I don't think it makes a big difference to the problem with shared interrupts (assuming my analysis is right).

The loop isn't really necessary if you are sure that unserved interrupts will be re-asserted (as level-triggered interrupts would be) (though it may still help performance) and the loop isn't sufficient if you need to be sure that all events get services (as there may be more in the shared-chain).

Thanks,  
NeilBrown

Re: [PATCH 2.6.16] Shared interrupts sometimes lost

—

To unsubscribe from this list: send the line "unsubscribe linux-kernel" in the body of a message to majordomo@xxxxxxxxxxxxxxxxx

More majordomo info at <http://vger.kernel.org/majordomo-info.html>

Please read the FAQ at <http://www.tux.org/lkml/>