2010-01-04 Qemu PCI card with IRQ

Plan

- Find the right way of triggering IRQ through PCI:
  - when to assert, when to deassert IRQ line;
  - when to start, when to cease interrupting;
- Make PCI device driver that would handle IRQs;

Worklog

- Modify qemu (easy part): add timer to qemu device, use qemu_irq_pulse inside its callback;
- Make PCI device driver: start with init and exit fns, pci_driver structure and device table;
- Make PCI probe fn; main steps:
  - pci_enable_device (enable device IO and memory);
  - request_irq (still there's no device open fn do it in probe);
  - pci_set_drvdata (to make our device accessible from the struct device level);
- Make PCI release fn, do cleanup of what probe done;
- Add simple irq handler and sysfs output;

Code: ftp://kkv.spb.su/pub/home/dumb/ws/lted/20100104

Conclusion

- when to assert/deassert IRQ line: no matter (at least for emulation);
  - although it is generally level-triggered, qemu treats them as edge-triggered, so that qemu_irq_pulse may be used;
  - anyway (e.g. for proper IRQ sharing) separate 'irq status' register is needed for the device function (not implemented in this demo, but hold on :);
- when to start/stop interrupts: doc contradiction;
  - PCI spec says that IRQ is enabled (interrupt disable == 0 in command register) upon reset, but linux pci documentation says that IRQ will go only when you explicitly enable them;
  - anyway, nothing happens (at least, nothing is visible) until request_irq;
  - even more: right place to request_irq is not probe routine, but rather device open routine;
- qemu 0.12.1 yields maximum of ~3000 PCI interrupts/second;
- using tabs in Linux is stupid):

qemu, PCI, IRQ, lted