A Smart Port Card Tutorial

---

Hardware

John DeHart
Washington University
jdd@arl.wustl.edu
http://www.arl.wustl.edu/~jdd

References: New Links from Kits References Page

- Intel Embedded Module:
  - Data Sheet
  - Design Guide
- 430HX Chipset
  - NorthBridge
  - SouthBridge
- System FPGA
- Memory
- Mobile Pentium with MMX
  - Software Developer Manuals 1,2,3
  - Datasheet
- APIC
- Cache
Motivation

• Active Networking
• Network Probe
• High performance router architectures
  – PC as router is VERY limited
  – (Gigabit/s + Processing) on each port
  – MSR: Multi-Service multiport Router

The Smart Port Card

• Hardware:
  – SPC as a PC
    • How do they each boot?
  – SPC Hardware Components
    • What roles do they play?
Typical Pentium PC

How NetBSD Boots on a PC

Components:
- Pentium
- Boot ROM (replaced by BIOS in modern systems?)
- BIOS
- Bootloader
- Kernel
Sketch of How a PC Boots

... at least what I understand...

- Pentium after Reset:
  - fetches its first instruction from location 0xFFFFFFFF0
- Boot Code must be located at 0xFFFFFFFF0
- Boot Code jumps to BIOS located in ROM
  - Boot Code may actually be part of the BIOS...
- BIOS copies itself into memory (Shadow)
- BIOS remaps memory
  - future accesses to BIOS addresses go to memory instead of ROM.
- BIOS performs system configuration (some proprietary)
  - Motherboard Details
  - Pentium Details
  - NB/SB Chipset Details
  - Device configuration: IRQs, Memory maps...
  
How a PC Boots (continued…)

- BIOS loads bootloader into memory (from disk…)
- BIOS jumps to bootloader
- Bootloader performs some more configuration:
  - Pentium control registers
  - Cache configuration
  - Memory/Page model
- Bootloader determines what to run next.
- Bootloader may have to do some device configuration.
  - e.g. to get OS from a disk.
- Bootloader loads OS kernel into memory
- Bootloader jumps to start of OS kernel
How a PC Boots (continued…)

- Kernel does some OS-specific configuration:
  - for NetBSD look in: sys/arch/i386/i386/locore.s
  - Determines what CPU it has ("cpuid" instruction)
  - Paging
  - Virtual Memory
What SPC Needs

SPC Architecture
SPC Components

- **APIC**
  - PCI Bus Master
- **Pentium Embedded Module**
  - 166 MHz MMX Pentium Processor
    - L1 Cache: 16KB Data, 16KB Code
  - L2 cache: 512 KB
  - NorthBridge - 33 MHz, 32 bit PCI Bus
    - PCI Bus Master
- **System FPGA**
  - PCI Bus Slave
    - Xilinx XC4020XLA-1 FPGA
    - 20K Equivalent Gates
    - ~ 75% used
SPC Components (continued)

• Memory
  – EDO DRAM
  – 64MB (Max for current design)
  – SO DIMM
• Switch Interface - 1 Gb Utopia
• Link Interface - 1 Gb Utopia
• UART
  – Two Serial Ports
    • NetBSD system console
    • TTY port

System FPGA

• Coded in VHDL
• PCI slave device
• Replaces some of the PIIX3 (south bridge)
• Replaces some of the BIOS
• Replaces some of the Super IO Chip
• Provides reset capability
System FPGA: PIIX3 Functionality

- Programmable Interrupt Controller (PIC)
  - Four Interrupts supported and statically assigned:
    - PIT (IRQ 0)
    - APIC (IRQ 5)
    - COM1 (IRQ 4)
    - COM2 (IRQ 3)
  - Static fully-nested interrupt priority structure.
  - Specific End of Interrupt is the only EOI mode supported
- Programmable Interval Timer (PIT)
  - Generates a clock interrupt for NetBSD every ~10ms
- Reset - covered in a later slide

System FPGA: BIOS Functionality

- Interrupt functionality replaced by static values
- Simple 16 word by 32-bit “ROM”
  - Implements loop waiting for location 0xFFE00 to change value
  - Then jumps to boot loader code
- Does **NOT** perform configuration of Northbridge
  - This will be done by the boot loader
- Does **NOT** perform PCI configuration of APIC
  - This will be done by the APIC Driver
System FPGA: Super IO Chip Functionality

• UART Interface
  – Two Serial lines supported
  – Fixed IRQs

• Real Time Clock
  – only the register accesses of the RTC are supported
  – no interrupts supported
  – i.e. supported only so NetBSD didn’t need to change
  – i.e. no alarms will be generated

System FPGA: Reset

• SPC needs a reset before *every* download:
  – switch reset:
    • causes SPC to be reset
    • causes all connections in switch to be lost
  – System FPGA reset
    • causes SPC to be reset
    • has no effect on the switch

• Normal SouthBridge reset:
  – *I/O Register*: 0xCF9
  – Hard Reset: assert CPURST, PCIRST#, and RSTDRV
    • write 0xCF9 0x02 (00000010b)
    • write 0xCF9 0x06 (00000110b)
  – Soft Reset: assert INIT
    • write 0xCF9 0x00 (00000000b)
    • write 0xCF9 0x04 (00000100b)
System FPGA: Reset

• SPC Reset:
  – a sequence of two writes to memory addresses
  – APIC Control cells can write to
    • memory addresses
    • configuration registers
    • NOT I/O Registers! Argh...
  – To *mimic* the reset structure of the SB we use:
    • 0xFFFFFFF0
    • 0xFFFFFFF4
  – Hard Reset (all we really care about)
    • write 0xFFFFFFF0 0x02 (00000010b)
    • write 0xFFFFFFF4 0x06 (00000110b)

Caveats

• Intel Embedded Module problem
  – Memory corruption caused by noise on M/A bus
  – Hopefully it will be fixed before we ship
  – We work around it with 25 MHz PCI and no HLT
    • This reduces the probability of noise on the bus
• PCI Bus
  – 33 MHz vs. 25 MHz
• NetBSD Kernel HLT instruction
  – if absolutely nothing to do, NetBSD does a “HLT”
  – this reduces power consumption
  – also causes large power/current swings
  – We have removed the HLT instruction for the SPC
• Serial Cables
  – RS232 is not necessarily hot-swappable
    • sometimes you can get away with it but not always