Embedded Systems Programming

x86 System Architecture and PCI Bus (Module 9)

Yann-Hang Lee
Arizona State University
yhlee@asu.edu
(480) 727-7507

Summer 2014
Interrupt and APIC

- **Interrupt in 8086**
  - Two pins: NMI and INTR
  - *Interrupt Acknowledge Cycle* to fetch the interrupt vector number from 8259

- **APIC**
  - In Pentium and P6 processors
  - Receives interrupts and sends to core for handling
  - APIC bus: bi-directional data signals (APICD[1:0]) and clock (APICCLK)
  - Inter-processor interrupt messages for multi-processor systems
  - Static and dynamic (based on the priority of executing tasks) distribution
Hardware Initialization and Reset

- **Reset processor state**
  - EIP=0000FFFFOH, CS=F000H (segment) and FFFF0000H (base)
  - Disable paging, cache, and in real-address mode

- **Execute the first instruction at physical address FFFFFFFFOH.**
  - The EPROM containing the software initialization code or BIOS should be located at the upper memory space (including this address)
  - Run in real-mode, invalidate the TLBs, set up a GDT for selector 0x08 (code) and 0x10 (data), switch to protected mode
  - Start other components on motherboard (FPU, APIC, southbridge, etc.)
Typical x86 System Architecture

- **Chipset**
  - North Bridge
  - South Bridge
  - Firmware Hub

- Various chipsets available from Intel to meet performance requirements

- FSB, DMI/Hub interface

- System control hub (SCH) - GMCH and ICH are merged into one chip
Host Bridge in Quark

- A central hub that routes transactions to and from Quark CPU core, DRAM controller, and other functional blocks.
- CPU core ↔ PCI devices
  - via MMIO and IO accesses
PCI Bus

- Release 2.1 -- 66MHz, 32-bit and 64-bit connectors.
  - 3.3V or 5V based on PCI chip set’s buffer/drivers

- Agent, bus master (initiator) and slave (target)
- Bus transaction:
  - bus masters issue requests ⇒ arbitration ⇒ bus grant
  - issues address and command and begins a cycle frame (transaction)
    - memory, I/O, configuration read/write commands
  - a target is selected (device select)
  - it is ready to complete the data transfer phase

3.3V key 5V key 64-bit portion

1 12,13 50,51 62 94
PCI Bus Signals

A typical PCI read transaction

PCI master device

- CLK
- FRAME
- AD[63:32]
- C/BE[7:4]
- REQ64
- ACK64
- Misc control
- INT REQ
- BIST signals
- Error reporting
- REQ
- GNT
- RST

Real-time Systems Lab, Computer Science and Engineering, ASU
PCI Bus Operation

- **Address phase**
  - At the same time, initiator identifiers target device and the type of transaction
  - The initiator assert the FRAME# signal
  - Every PCI target device latch the address and decode it

- **Data Phase**
  - Number of data bytes to be transformed is determined by the number of Command/Byte Enable signals asserted by initiator
  - Both of initiator and target must be ready to complete data phase
  - IRDY# and TRDY# used

- **Transaction completion and return of bus to idle state**
  - By deasserting the FRAME# but asserting IRDY#
  - When the last data transfer has completed the initiator returns the PCI bus to idle state by deasserting IRDY#
PCI Commands

- Address and data phases
- PCI allows the use of up to 16 different 4-bit commands
  - Configuration commands
  - Memory commands
  - I/O commands
  - Special-purpose commands
- A command is presented on the C/BE# bus by the initiator during an address phase (a transaction’s first assertion of FRAME#)

<table>
<thead>
<tr>
<th>C/BE[3::0]#</th>
<th>Command Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>Interrupt Acknowledge</td>
</tr>
<tr>
<td>0001</td>
<td>Special Cycle</td>
</tr>
<tr>
<td>0010</td>
<td>I/O Read</td>
</tr>
<tr>
<td>0011</td>
<td>I/O Write</td>
</tr>
<tr>
<td>0100</td>
<td>Reserved</td>
</tr>
<tr>
<td>0101</td>
<td>Reserved</td>
</tr>
<tr>
<td>0110</td>
<td>Memory Read</td>
</tr>
<tr>
<td>0111</td>
<td>Memory Write</td>
</tr>
<tr>
<td>1000</td>
<td>Reserved</td>
</tr>
<tr>
<td>1001</td>
<td>Reserved</td>
</tr>
<tr>
<td>1010</td>
<td>Configuration Read</td>
</tr>
<tr>
<td>1011</td>
<td>Configuration Write</td>
</tr>
<tr>
<td>1100</td>
<td>Memory Read Multiple</td>
</tr>
<tr>
<td>1101</td>
<td>Dual Address Cycle</td>
</tr>
<tr>
<td>1110</td>
<td>Memory Read Line</td>
</tr>
<tr>
<td>1111</td>
<td>Memory Write and Invalidate</td>
</tr>
</tbody>
</table>
Supplementary Slides
Interrupt Handling

- IO APIC delivers interrupt message to local APIC
  - Programmable vector number for each interrupt source
- Implied priority based on vector number
  - Local APIC determines when to service the interrupt relative to the other activities of the processor
  - Priority = vector / 16
- Locate gate from IDT
  - Far call to the handler
  - (SS, ESP), EFLAGS, CS, EIP, and Error code are saved in stack
Message Bus Register Access

- Indirect access via PCI configuration space
  - Message Bus Control Reg. (MCR) - PCI[B:0,D:0,F:0]+D0h
  - Message Data Reg. (MDR) - PCI[B:0,D:0,F:0]+D4h
  - Message Control Reg. eXtension (MCRX) - PCI[B:0,D:0,F:0]+D8h
- Uses the MCR/MCRX as an index register and MDR as the data register.
- Writes to the MCR trigger message bus transactions
- MCR description

<table>
<thead>
<tr>
<th>Field</th>
<th>MBPR Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>OpCode (typically 10h for read, 11h for write)</td>
<td>31:24</td>
</tr>
<tr>
<td>Port</td>
<td>23:16</td>
</tr>
<tr>
<td>Offset/Register</td>
<td>15:08</td>
</tr>
<tr>
<td>Byte Enable</td>
<td>07:04</td>
</tr>
</tbody>
</table>
PCI Optimizations and Additional Features

- **Push bus efficiency toward 100% under common simple usage**
- **Bus parking**
  - retain bus grant for previous master until another makes request
  - granted master can start next transfer without arbitration
- **Arbitrary burst length**
  - initiator and target can exert flow control with xRDY
  - discount with STOP (abort or retry, by target), FRAME (by master) and GNT (by arbiter)
- **Delayed (pended, split-phase) transactions**
  - free the bus after request to slow device
- **Additional Features**
  - Interrupts: support for controlling I/O devices
  - Cache coherency: support for I/O and multiprocessors
  - Locks: support timesharing, I/O, and MPs
  - Configuration Address Space (plug and play)