K-OS Interrupts

Kai Chen

CS134c Spring 2003

Overview

- **Set up IDT**
- **Define and install ISR**
- **Remap and enable PIC**

Interrupt Basics

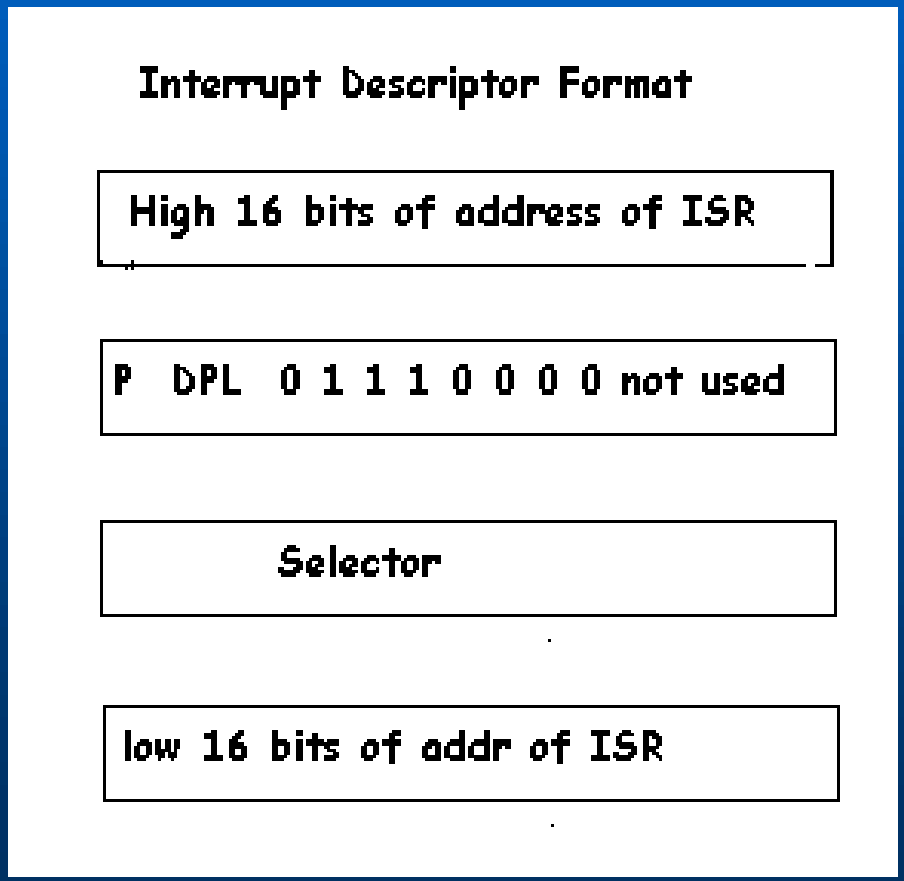
- **Hardware interrupt (IRQ)**
  - **Hardware fires IRQ**
  - **CPU finishes current instruction, looks at PIC to figure out which INT this IRQ maps to.**
  - **Fetch corresponding descriptor in IDT, and control is passed to ISR**
  - **Resume**
- **Software interrupt (exception, fault, trap …)**
  - **Same as hardware interrupts, except not remapped by PIC**

Interrupt Descriptor Table

- **An array of descriptors associating INTs with ISRs**
- **Each descriptor 8 bytes long**
- **At most 256 descriptors**

Interrupt Descriptor Format

- **Mainly contains a pointer to ISR**
  - **highest word: high 16 bits of address**
  - **lowest word: low 16 bits of address**
- **Other information**



Setting up IDT

- **IDT Structure:**

```
idtr:
    dw IDTLimit          ; idt size - 1
    dd IDTLBase          ; starting address
null_descriptor:         ; null descriptor
    ...                  ; 8 bytes for each descriptor
timer_descriptor:        ; descriptor for timer
    ...
kerboard_descriptor:     ; descriptor for keyboard
    ...
```

- **Loading IDT Register:**
  - LIDT [ldtr]

Interrupt Request

- **IRQs are hardware interrupts.**
  - **There are 16 IRQs in total**
  - **Two blocks of 8. Each associated with a PIC**
  - **Timer (IRQ 0), Keyboard (IRQ 1) etc ...**
- **In real mode**
  - **IRQ 0 - 7 mapped to INT 08h - 0Fh**
  - **IRQ 8 - 15 mapped to INT 70h - 77h**
- **Interferes with software interrupts in protected mode. Need to remap PIC**

Programmable Interrupt Controller

- There are 2 PICS
  - PIC1 -- Master -- IRQ 0 - 7
  - PIC2 -- Slave -- IRQ 8 - 15
- Communicate using ICWs
  - ICW1: specifies whether ICW4 is sent
  - ICW2: Remapping information (high 5 bits)
  - ICW3:
    - Master: each bit 0 if IRQ connected to peripheral, 1 to slave
    - Slave: specify connected to which IRQ in master
  - ICW4: Expecting EOI?
  - OCW1: Masking IRQs
  - OCW2: Only used for sending EOI

Interrupt Service Routine

- These are Interrupt Handlers.
- Assembly Wrapper
  - push registers
  - call real handler
  - signal EOI
  - pop registers
  - IRET
- Handler to be implemented in C

Putting All together

- **Setting up IDT**
  - **Populate descriptor with ISR**
  - **Insert descriptors into IDT**
  - **Fill in rest of IDT with null descriptor**
- **Enable PIC**
  - **Send ICW 1 - 4 to master and slave PICs**
  - **Mask off IRQs not used**