Computer Organization & Architecture

7-4 Multiple Devices Interrupt System

Design Issues

Wang Guohua

School of Software Engineering

Contents of this lecture

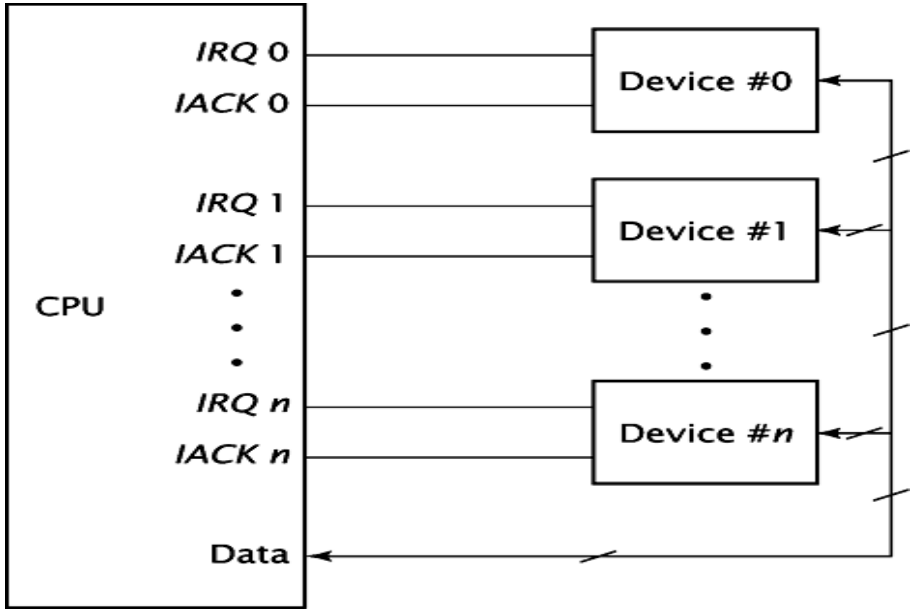
- Multiple Devices Interrupt System Design Issues
  - Identify Interrupt Source
  - Multi-level Interrupt
  - Simultaneous Interrupt

Multiple Devices Interrupt System Design Issues

- How do you identify the module issuing the interrupt? How the processor obtain the starting address of the appropriate routine of different devices?—Identify Interrupt Source
- Should a device be allowed to interrupt the processor while another interrupt is being serviced? —Multi-level Interrupt
- How should two or more simultaneous interrupt requests be handled? —Simultaneous Interrupt

Identify Interrupt Source (1)

- Case1: Multiple Interrupt-request Lines
  - Provide multiple interrupt lines between the processor and the I/O interface.
  - Even if multiple lines are used, it is likely that each line will have multiple I/O interface attached to it.



Identify Interrupt Source (2)

- Case2: Common Interrupt-request Line
  - Polling (Non-vectored Interrupt)
  - Vectored Interrupt
- Polling (Non-vectored Interrupt)
  - What is non-vectored Interrupt?
    - An interrupt is received by the CPU, and it jumps the program counter to a fixed address in hardware.
  - Useful for small systems where there are few interrupt sources and the software structure is straightforward.

Identify Interrupt Source (3)

- Polling (Non-vectored Interrupt) (ctd.)
  - Implementation methods
    - The processor reads the status register of each I/O interface. (When a device raises an interrupt request, it sets IRQ bit in its status register to 1.)

Identify Interrupt Source (4)

- Polling (Nonvectored Interrupt) (ctd.)
  - Registers in the Keyboard and Display Interfaces

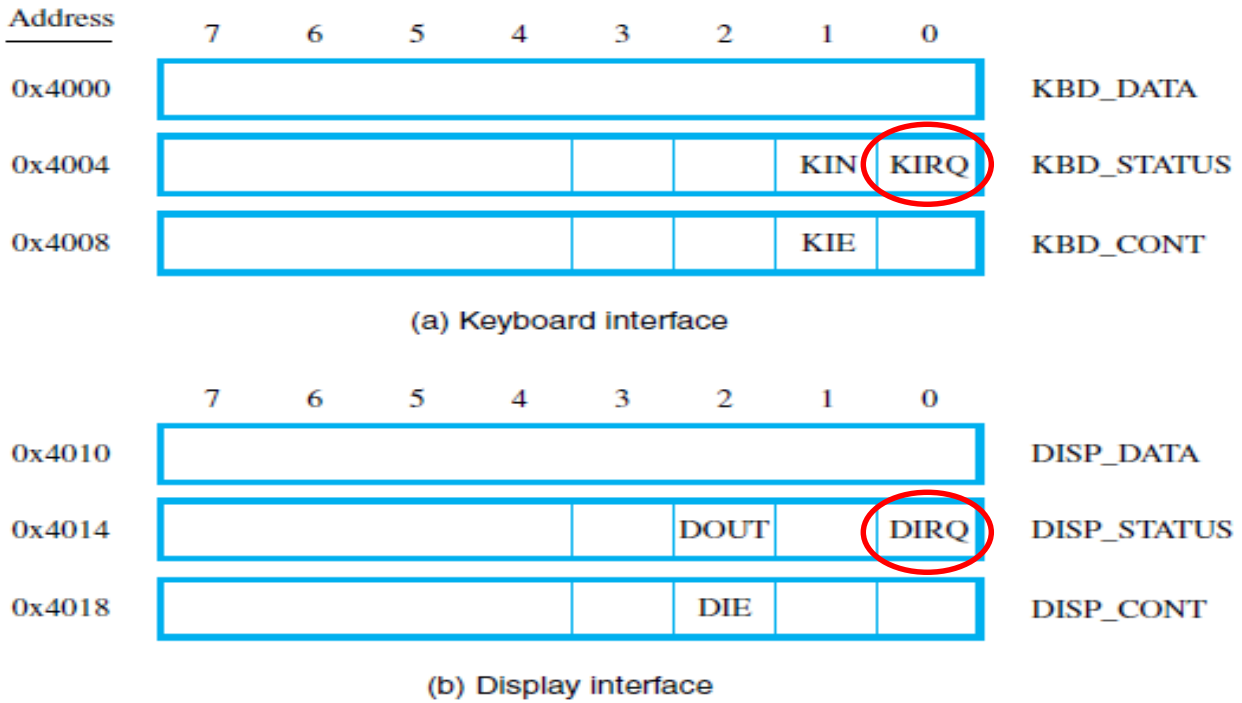


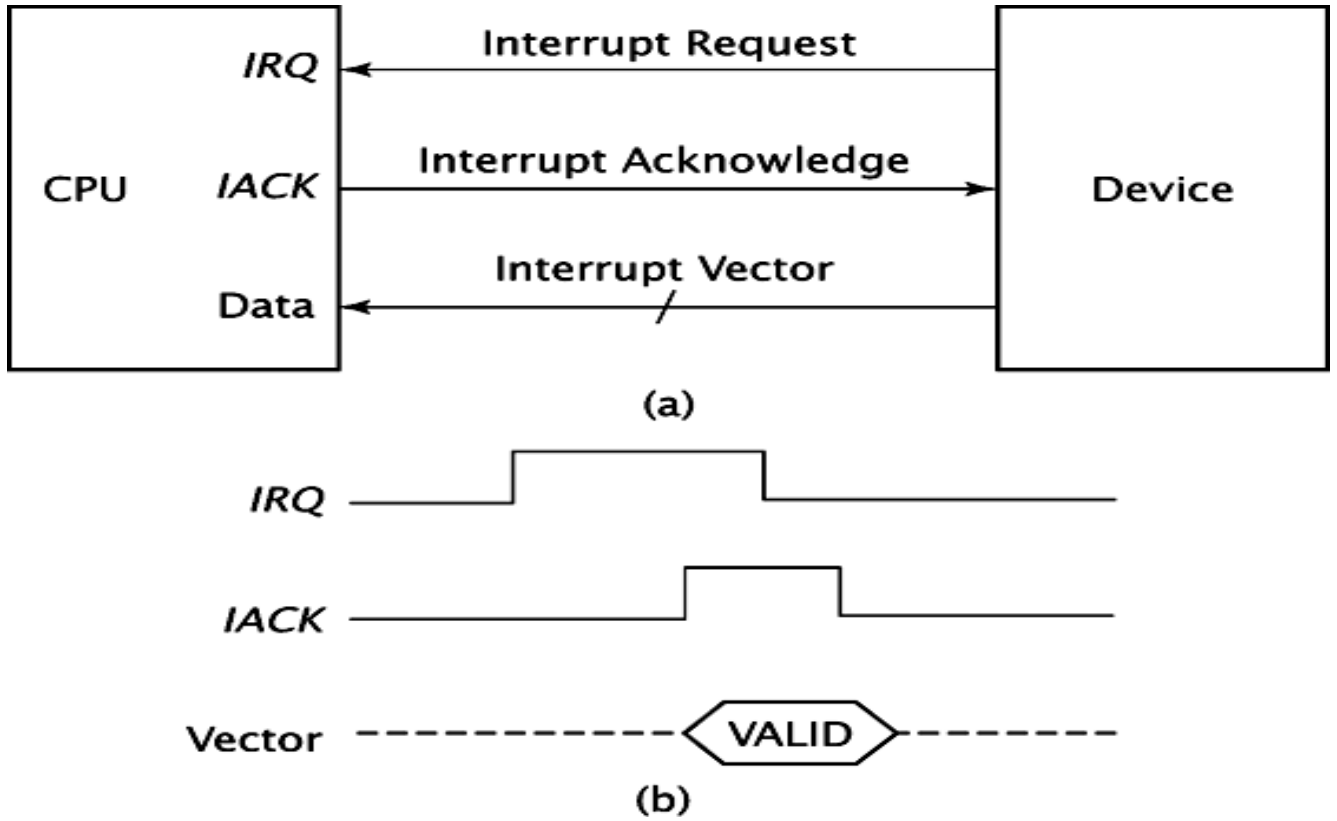
Figure 3.3 Registers in the keyboard and display interfaces.

Identify Interrupt Source (5)

- Vectored Interrupt
  - What is vectored Interrupt?
    - An interrupt scheme where the device identifies itself by sending interrupt vector code to the processor when requesting an interrupt.
  - Interrupt Vector
    - An interrupt vector is the memory address of an interrupt handler.
    - Interrupt vector table contains the memory addresses of all interrupt handlers.
    - Interrupt vector code is an index into the interrupt vector table.

Identify Interrupt Source (6)

- Vectored Interrupt (ctd.)
  - Figure



Multi-level Interrupt (1)

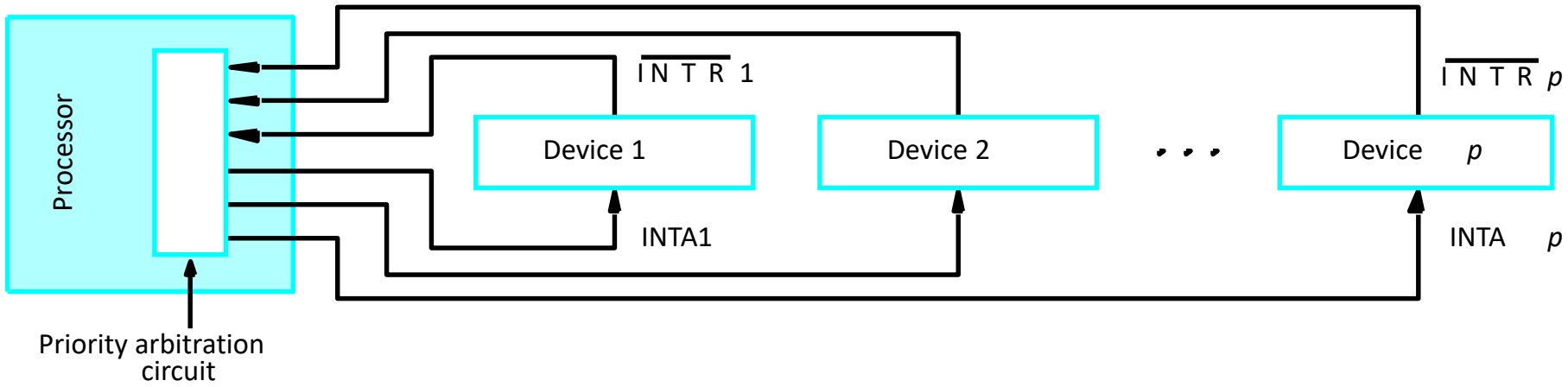
- Single-level Interrupt
  - Whether one device or several I/O devices, interrupts should be disabled during the execution of an interrupt-service routine.
- Multi-level Interrupt (Interrupt Nesting)
  - It is necessary to accept another device interrupt during the execution of an interrupt-service routine for a device.
  - I/O devices are organized in a priority structure. An interrupt request from a high-priority device should be accepted while the processor is serving another request from a lower-priority device.

Multi-level Interrupt (2)

- Multi-Level Priority
  - Assign a priority level to the processor that can be changed under program (privileged instructions) control.
  - The priority level of the processor is the priority of the program that is currently being executed.
  - The processor accepts interrupts only from devices that have priorities higher than its own.
  - At the time the execution of an interrupt-service routine for some device is started, the priority of the processor is raised to that of the device.
  - The processor’s priority can be encoded in a few bits of the processor status register.

Multi-level Interrupt (3)

- Multi-Level Priority (ctd.)
  - Figure



Implementation of interrupt priority using individual interrupt-request and acknowledge lines.

Simultaneous Interrupt (1)

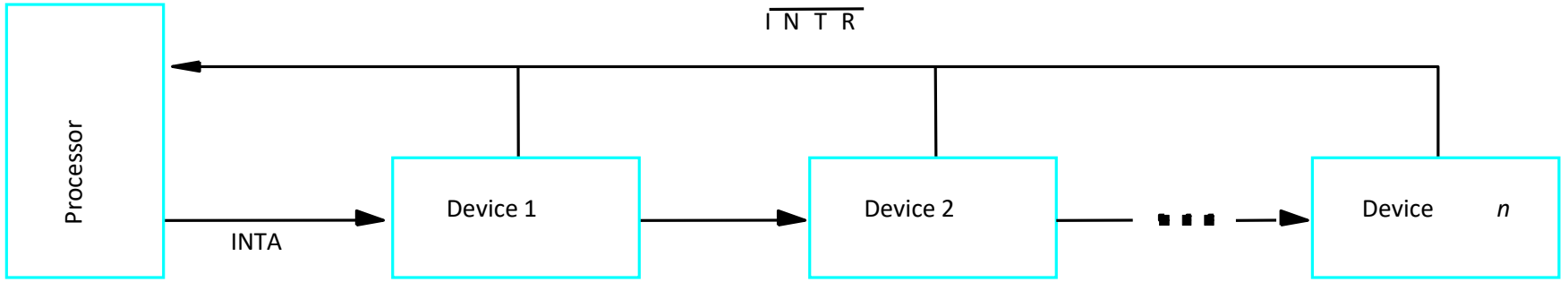
- Case 1: Individual device has individual interrupt-request and acknowledge lines
  - The processor simply accepts the request having the highest priority.
  - It allows the processor to accept interrupt requests from some devices but not from others, depending upon their priorities.
- Case 2: Several devices share one interrupt-request line
  - Software Poll
  - Daisy Chain (Hardware Poll)
  - Priority Group

Simultaneous Interrupt (2)

- Software Poll
  - The processor poll the status register of the I/O device.
  - Priority is determined by the order in which the devices are polled.

Simultaneous Interrupt (3)

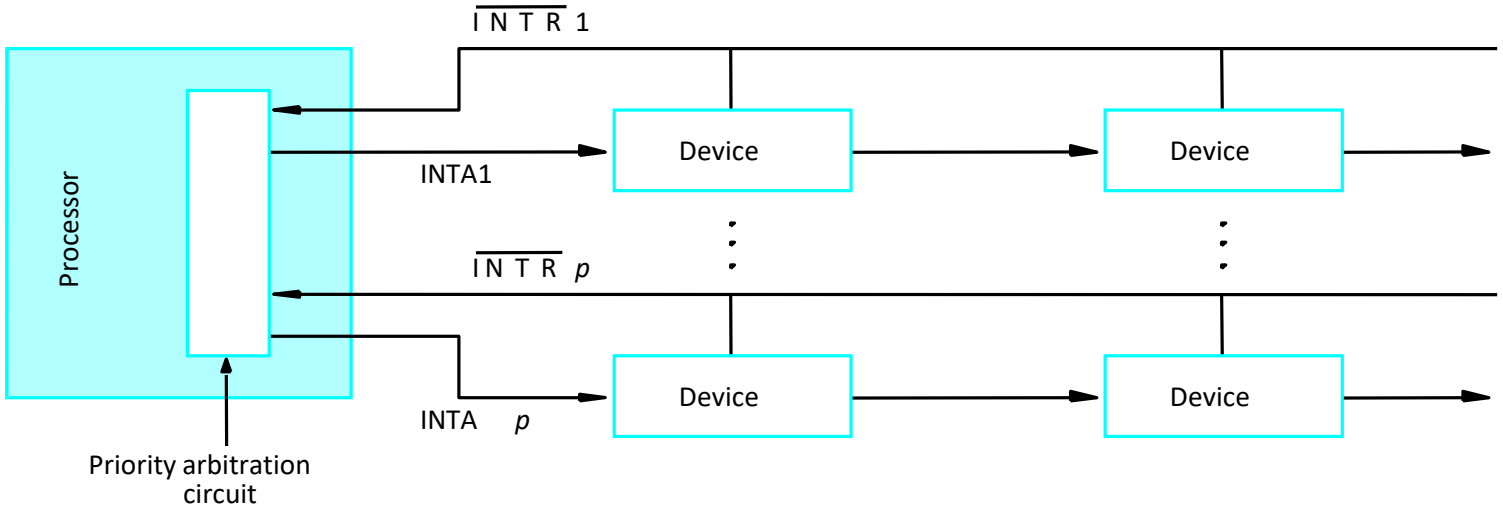
- Daisy Chain
  - Figure



  - In the daisy chain arrangement, the device that is electrically closest to the processor has the highest priority.
  - The second device along the chain has second highest priority, and so on.

Simultaneous Interrupt (4)

- Priority Group
  - Figure



Quiz (1)

1. In the process of determining which device is requesting an interrupt, the device identifies itself directly by sending a signal or a binary code to the processor. This approach is called _____.

   A. polling

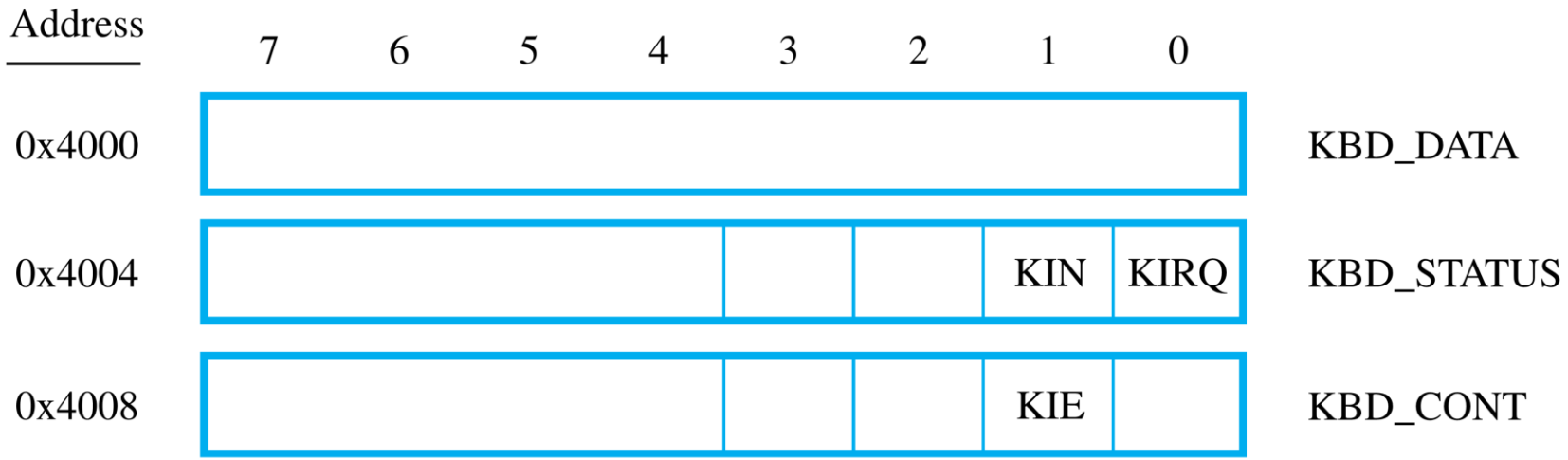
   B. non-vectored interrupt

   C. vectored interrupt

   D. interrupt vector

Quiz (2)

2. The following figure shows the interface of a keyboard. Write functions of (1) KIN (2) KIRQ (3) KIE.



Quiz (3)

Solution:

(1) The KIN bit is the status flag for the processor to read to determine when a character code has been placed in KBD_DATA. If KIN=1, then the character has been in KBD_DATA, otherwise, KIN=0.

(2) The KIRQ bit is the status flag whether the keyboard is requesting an interrupt. If KIRQ=1, it shows that the keyboard raises an interrupt, otherwise, KIRQ=0.

(3) The KIE bit is the bit to control whether the keyboard can raise an interrupt. If KIE=1, then the keyboard is placed into a mode in which it is allowed to interrupt the processor whenever it is ready for an I/O transfer.