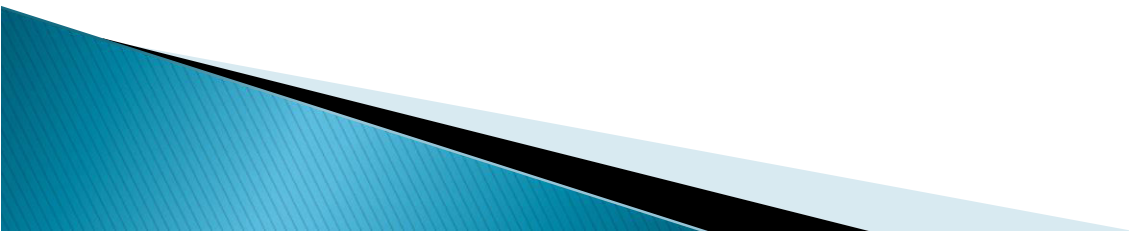


# 8259 Programmable Interrupt Controller



# Introduction

- ▶ An interrupt is an event which informs the CPU that its service (action) is needed.
- ▶ Sources of interrupts:
  - internal fault (e.g.. divide by zero, overflow)
  - software
  - external hardware :
    - maskable
    - nonmaskable
  - reset

# Basic Procedure for Processing Interrupts

- ▶ When an interrupt is executed, the *mp*:
  - finishes executing its current instruction (if any).
  - saves (PUSH) the flag register, IP and CS register in the stack.
  - goes to a fixed memory location.
  - reads the address of the associated ISR.
  - Jumps to that address and executes the ISR.
  - gets (PULL) the flag register, CS:IP register from the stack.
  - continues executing the previous job (if any).

# 8088/86 Hardware Interrupts pins

**INTR:** Interrupt Request.

- Input signal into the CPU
- If it is activated, the CPU will finish the current **instruction** and respond with the interrupt acknowledge operation
- Can be masked (ignored) thru instructions CLI and STI

▶ **NMI:** NonMaskable interrupt.

- Input signal
- Cannot be masked or unmasked thru CLI and STI
- Examples of use: power frailer. Memory error

▶ **INTA:** Interrupt Acknowledge.

- Output signal

# The Interrupt flag

- ▶ IF (Interrupt Enable Flag) D9: used to mask any hardware interrupt that may come in from the INTR pin.
- ▶ When IF=0, all hardware interrupt requests through INTR are masked.
- ▶ This has no effect on interrupts coming from the **NMI** pin or “INT nn” instructions.
- ▶ **CLI** sets IF to 0, **STI** sets IF to 1.

# INT $n$ and ISR

- ▶  $n$  is multiplied by 4
- ▶ In the address “ $4n$ ” the offset address the ISR is found.
- ▶ Example: Intel has set aside **INT 2** for the **NMI** interrupt.
- ▶ Whenever the NMI pin is activated, the CPU jumps to physical memory location 00008 to fetch the CS:IP of the interrupt service routine associated with the NMI.

# 8259

- ▶ 8259 is Programmable Interrupt Controller (PIC)
- ▶ It is a tool for managing the interrupt requests.
- ▶ 8259 is a very flexible peripheral controller chip:
  - PIC can deal with up to 64 interrupt inputs
  - interrupts can be masked
  - various priority schemes can also programmed.
- ▶ originally (in PC XT) it is available as a separate IC
- ▶ Later the functionality of (*two PICs*) is in the motherboards chipset.
- ▶ In some of the modern processors, the functionality of the *PIC* is built in.

# Pin description

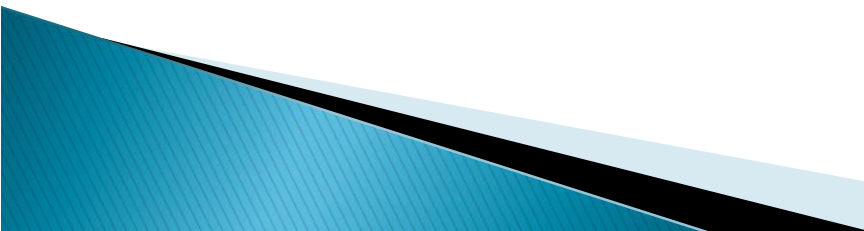
- ▶ 8-bit bi-directional data bus, one address line is needed, PIC has two control registers to be programmed, you can think of them as two output ports or two memory location.
  - ▶ The direction of data flow is controlled by RD and WR.
  - ▶ CS is as usual connected to the output of the address decoder.
  - ▶ Interrupt requests are output on INT which is connected to the INTR of the processor. Int. acknowledgment is received by INTA.
  - ▶ IR0-IR7 allow 8 separate interrupt requests to be inputted to the PIC.
  - ▶ sp/en=1 for master , sp/en=0 for slave.
  - ▶ CAS0-3 inputs/outputs are used when more than one PIC to cascaded.
- 



FIGURE 9-4 Block diagram and pin definitions for the 8259A Programmable Interrupt Controller (PIC). (Courtesy of Intel Corporation.)

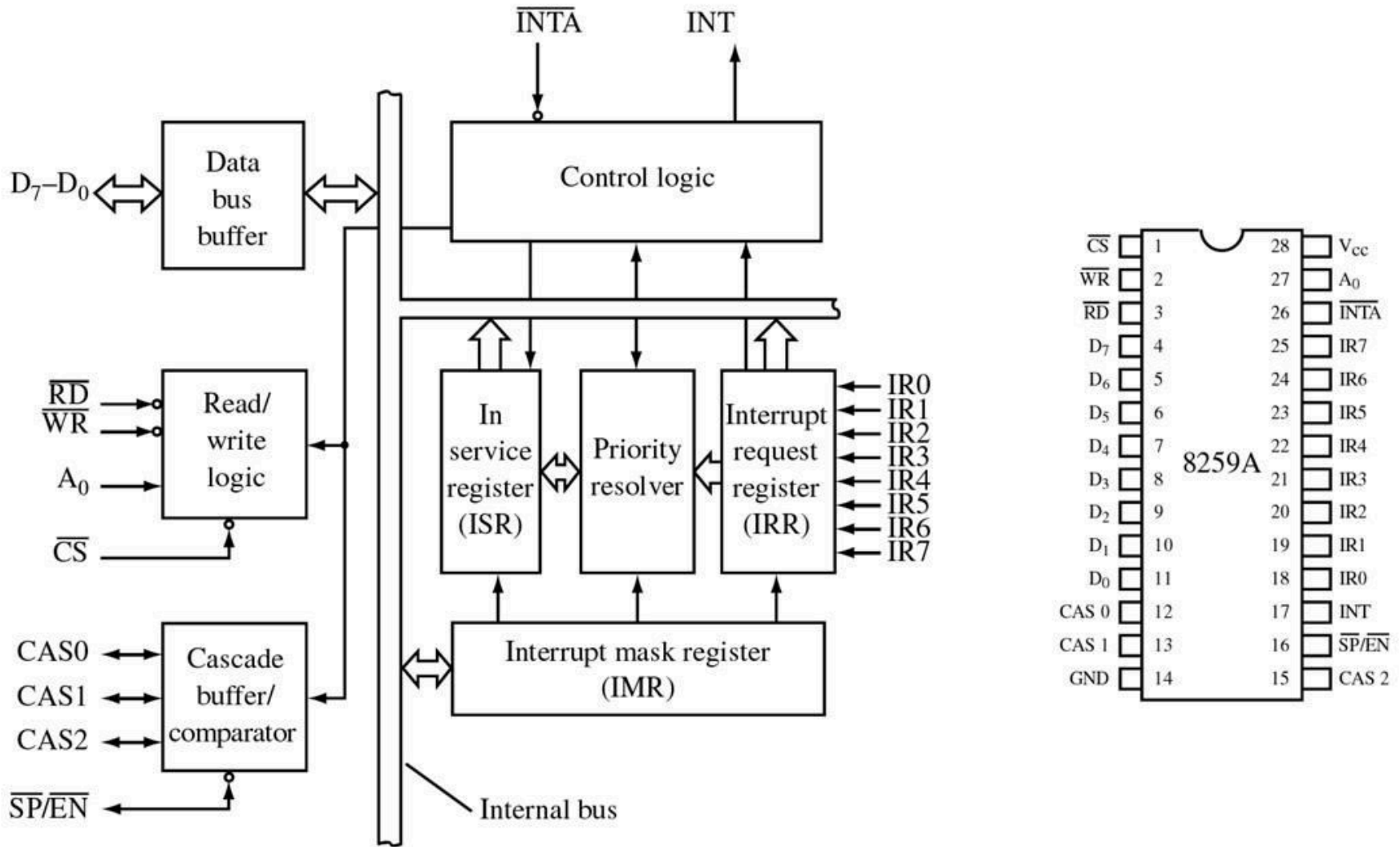


FIGURE 9-5 Interfacing the PIC to the 386 and 486 processors. Two I/O ports are required.

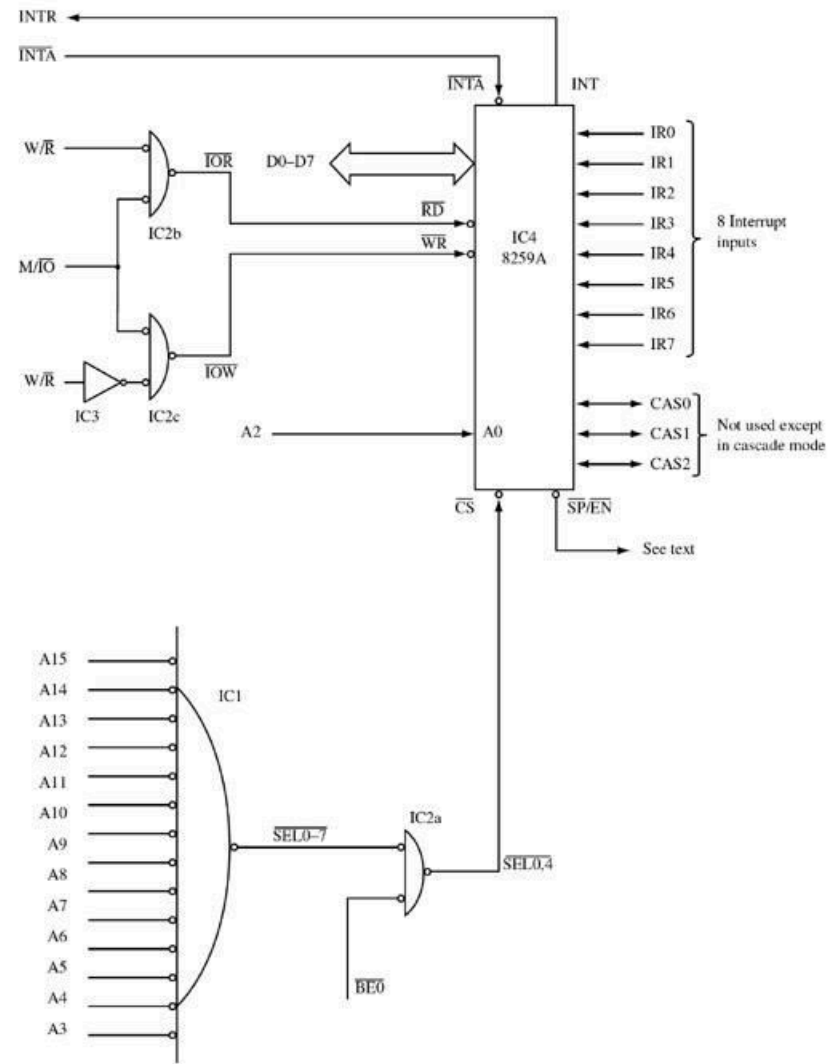
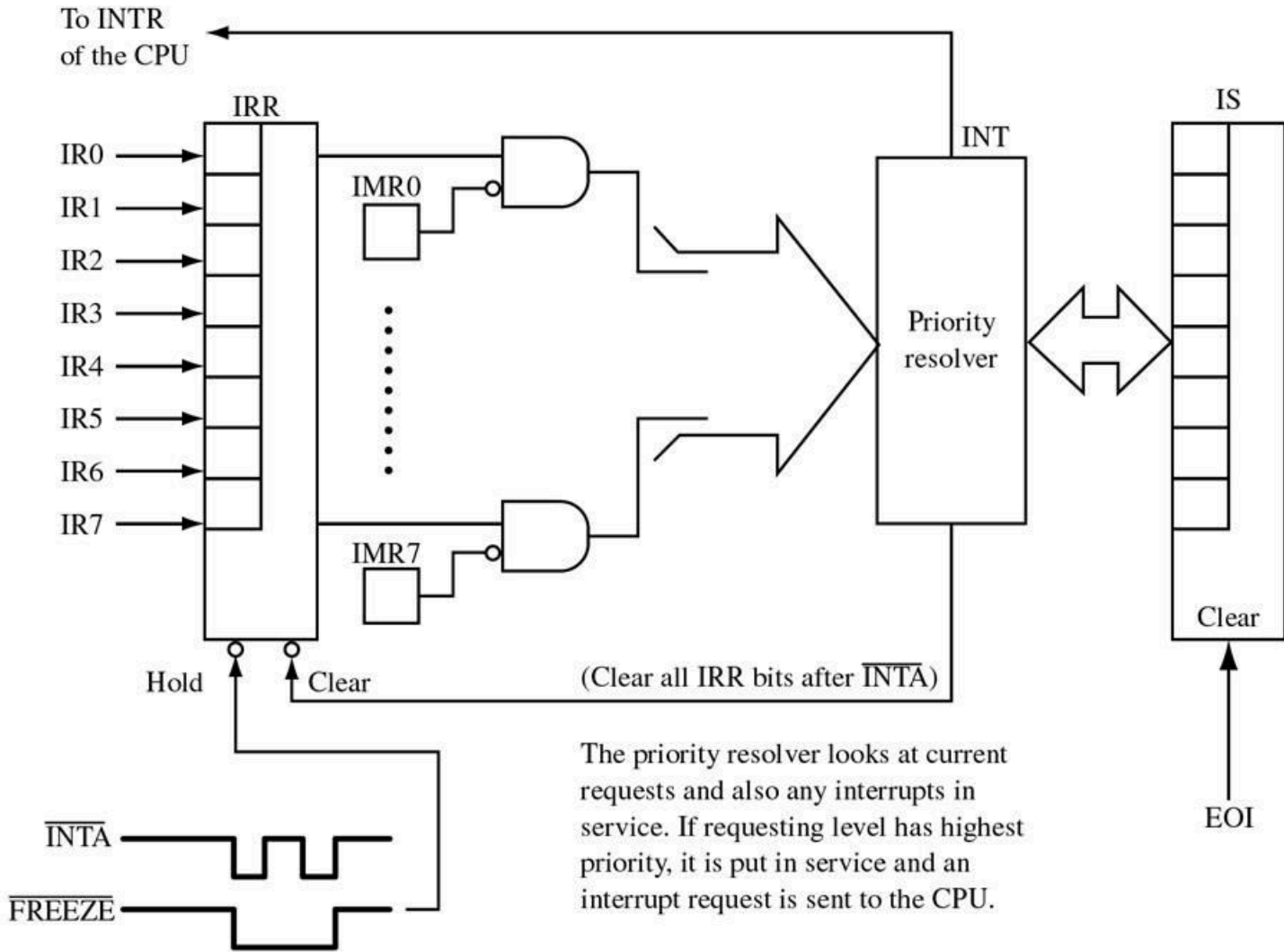
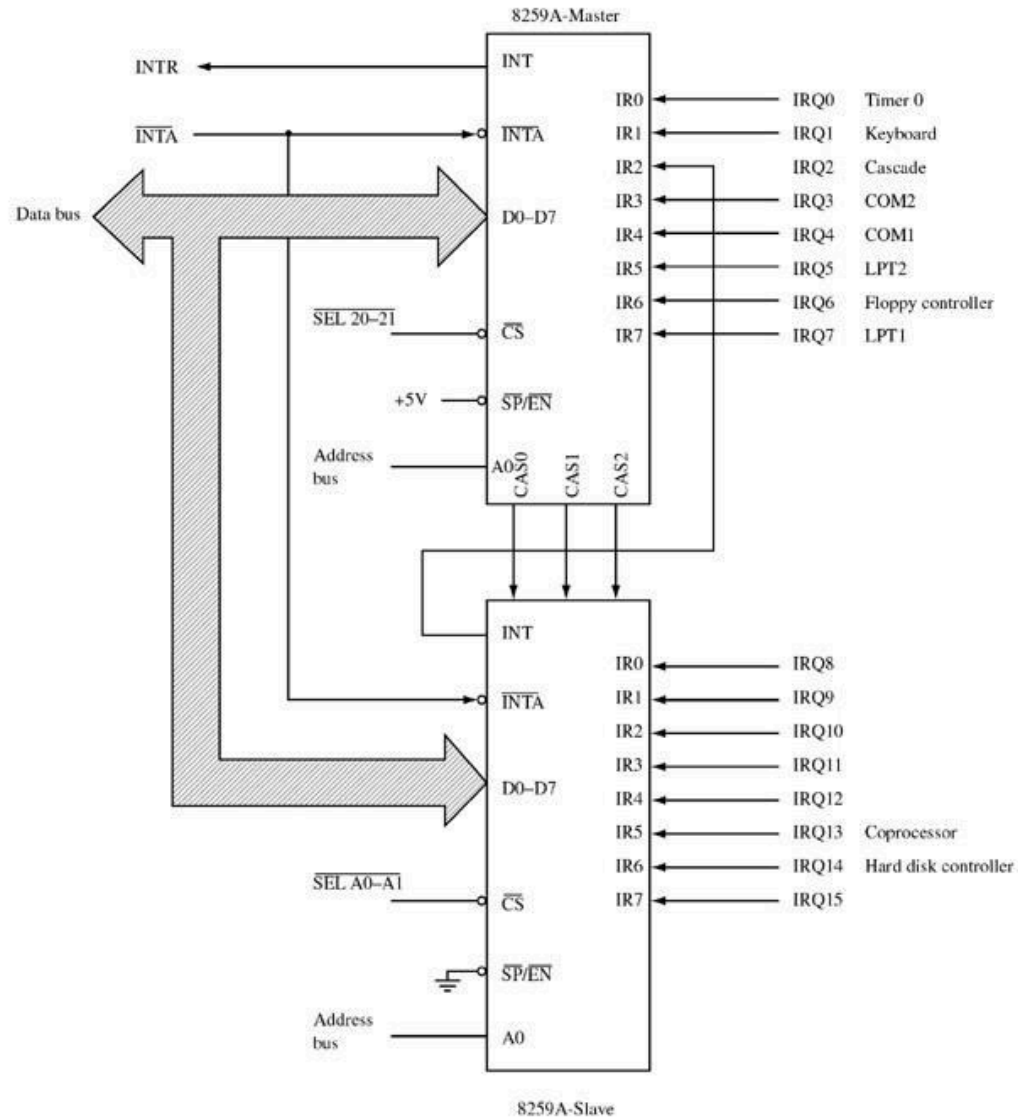


FIGURE 9-7 All interrupt requests must pass through the PIC's interrupt request register (IRR) and interrupt mask register (IMR). If put in service, the appropriate bit of the in-service (IS) register is set.



# Example of two cascaded PICs



# OPERATION

- ▶ PIC is to be initialized and programmed to control its operation.
- ▶ The operation in simple words:  
when an interrupt occurs , the PIC determines the highest priority, activates the processor via its INTR input, and sends the type number onto the data bus when the processor acknowledges the interrupt.
- ▶ Priority:  
What is used in PC is *fully nested mode*. That is the lowest numbered IRQ input has highest priority. Lower priority interrupts will not be forwarded to the processor until the higher priority interrupts have been serviced.

# Modes

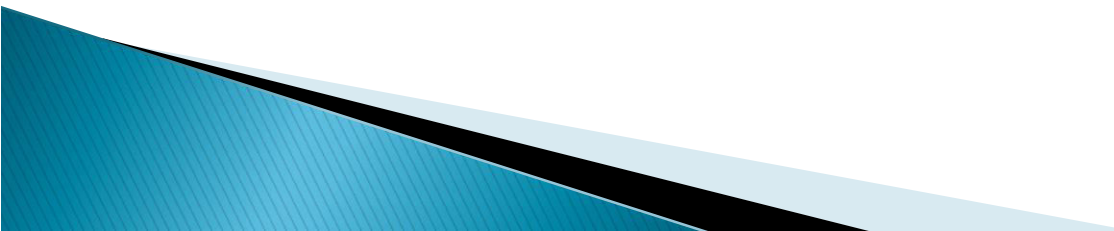
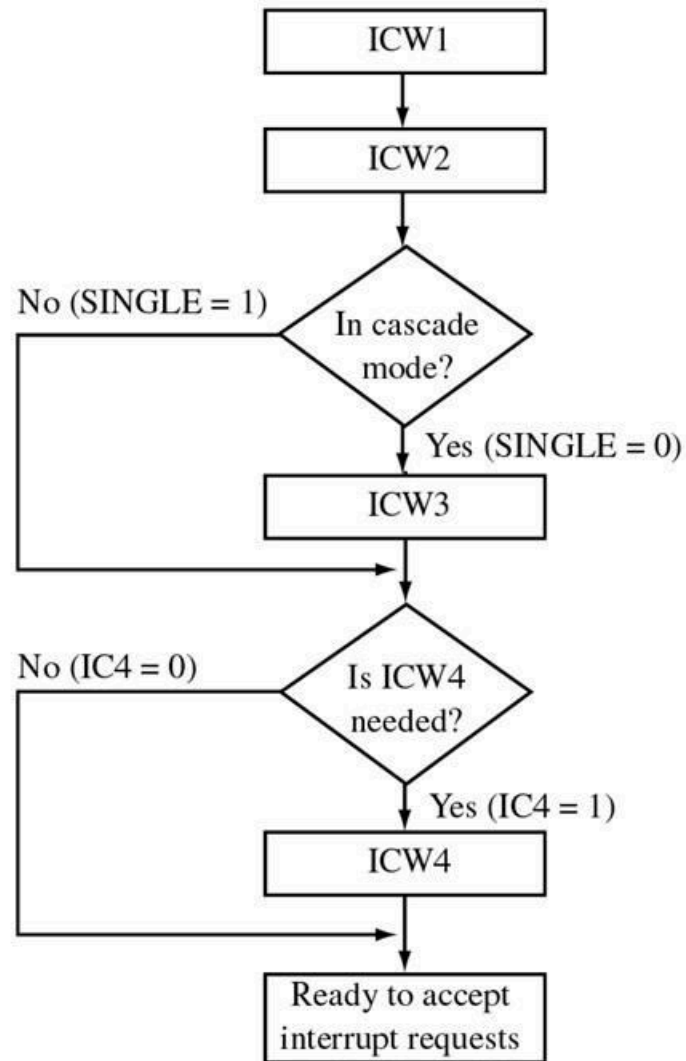
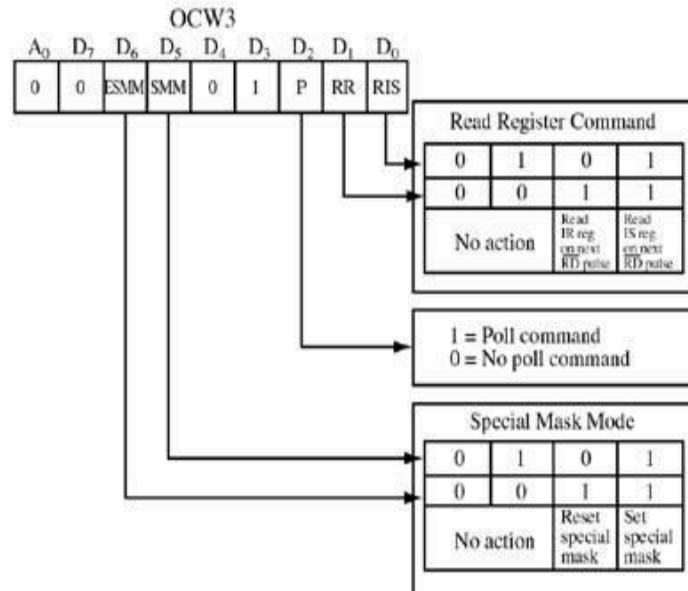
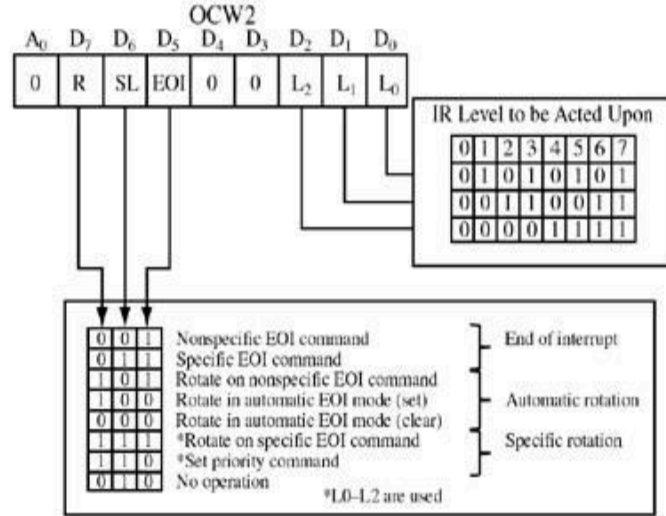
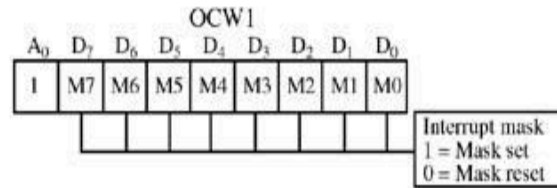
- ▶ Fully Nested mode
  - ▶ Special Fully Nested mode
  - ▶ Nonspecific Rotating
  - ▶ Specific Rotating
  - ▶ Special Mask
  - ▶ Polling
- 

FIGURE 9-12 8259A initialization sequence. (Courtesy of Intel Corporation.)







# DMA

- ▶ Direct Memory Access.
- ▶ In memory–memory or memory–peripherals communication, the processor is a “middleman” which is not really needed.
- ▶ Used with HOLD HOLDA signals.
- ▶ DMA requires another processor – The DMA Controller or DMAC– to generate the memory and I/O addresses.
- ▶ 8237 is a DMAC.
- ▶ In IBM PC, 8237 was used to speed up the read or write operation by the slow 8088 processor.
- ▶ Nowadays, It is usually used by sound cards and by memory controllers to generate row address for refreshing.

FIGURE 9-17 A DMA controller allows the peripheral to interface directly with memory without processor intervention. This allows the data transfer rate to approach the access time of memory.

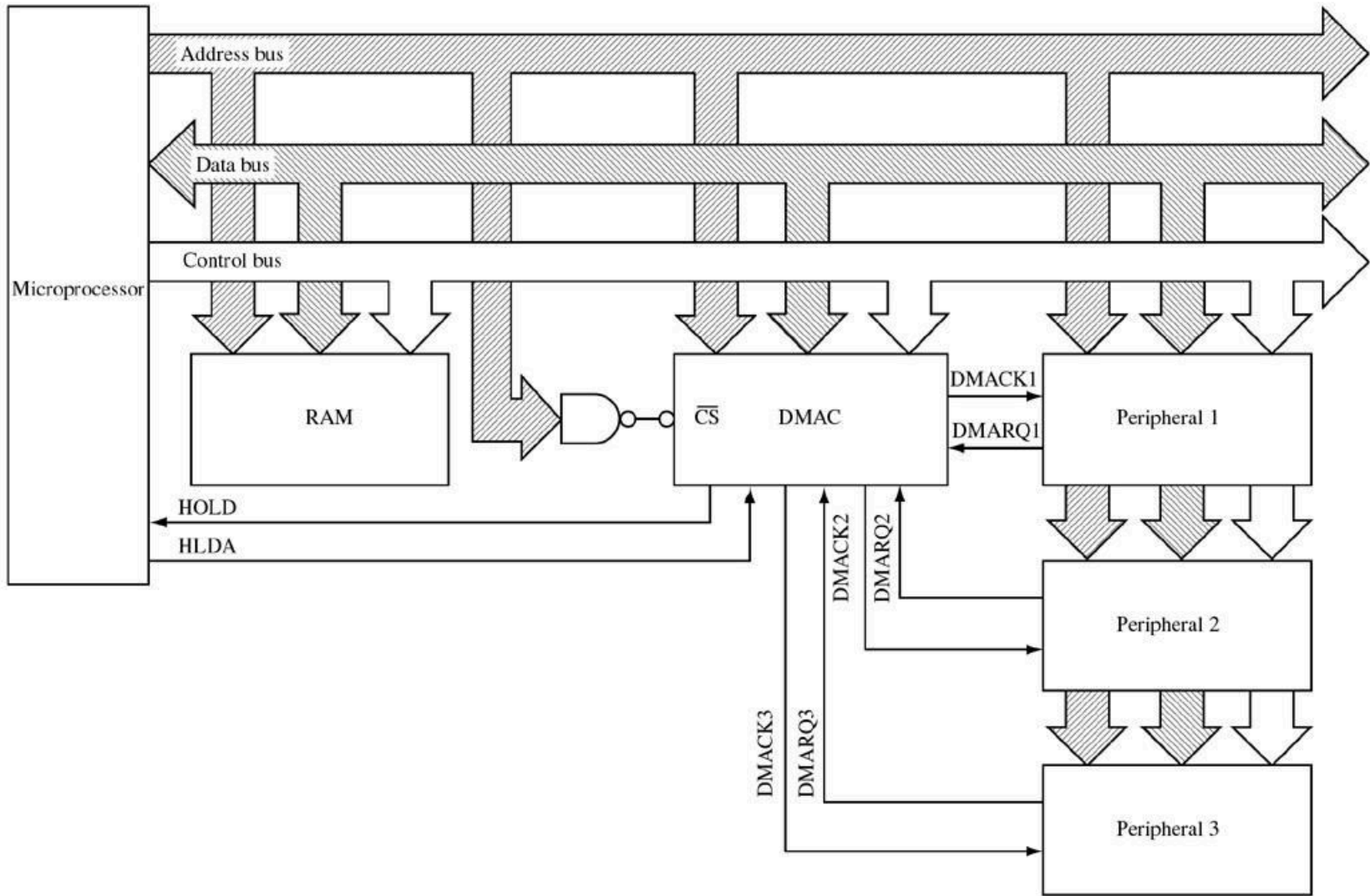
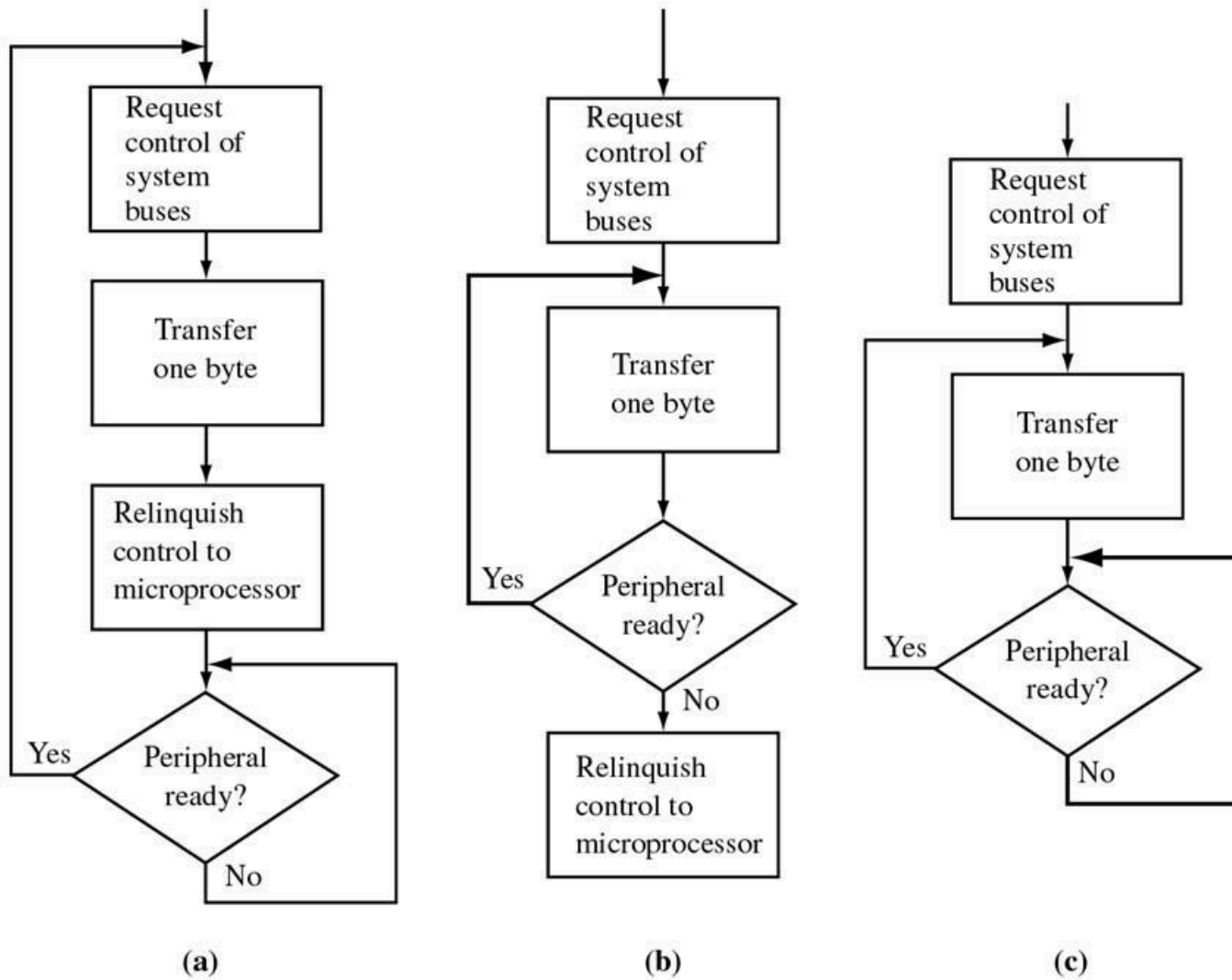
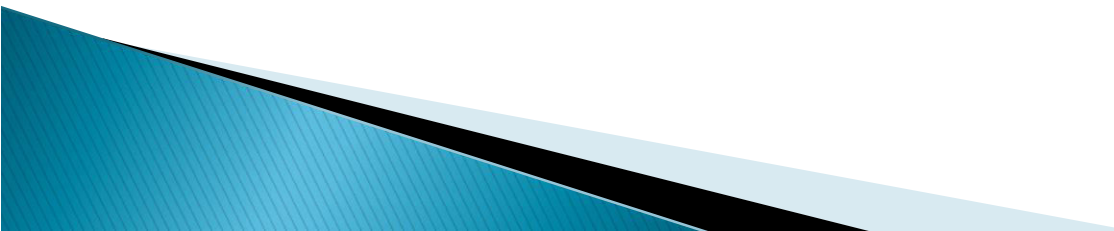


FIGURE 9-18 Three methods (MODES) of DMA operation: (a) byte; (b) burst; (c) block.

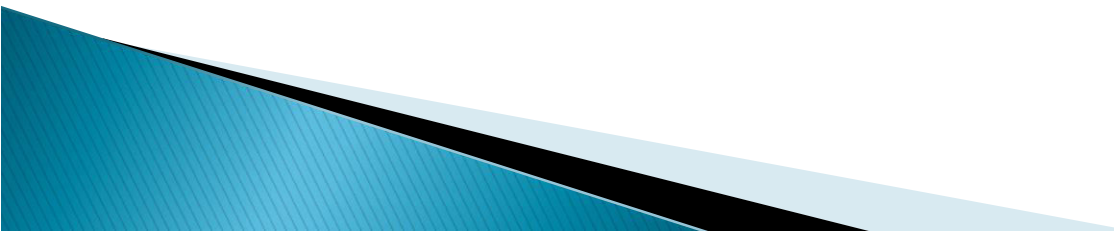


# IO/summery

Three ways to synchronize the processor to data rate of peripherals:

- 1- Polling: which provides a fast response but it the processor recourses are dedicated to one peripheral.
  - 2- Interrupt approach: is much more efficient. the processor only services the peripheral when data is required. requires high software overhead.
  - 3-DMA is a third solution but it increases the complexity of the hardware system.
- 

# Serial I/O

- ▶ Microprocessors are by nature parallel machines. They transmit/receive data in parallel bits (8,16,32,64).
  - ▶ It is required sometimes to send the data serially (one bit at a time).
  - ▶ serial transmitting is slower but requires less wires and it is easier to send it for long distances.
- 

# Synchronous vs. Asynchronous

- ▶ Asynchronous : Start bits, Stop bits, and Data
- ▶ Ex: the data byte is 7BH:

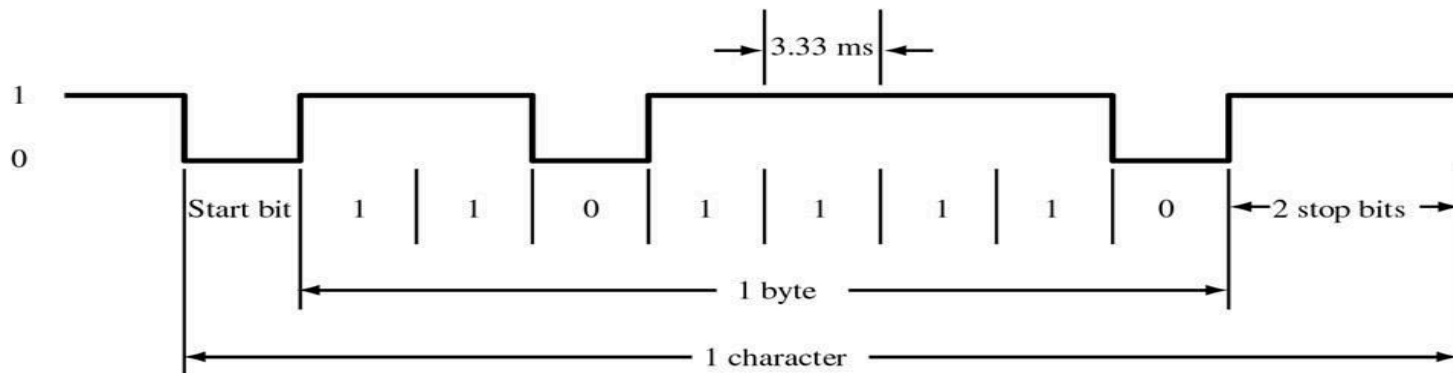
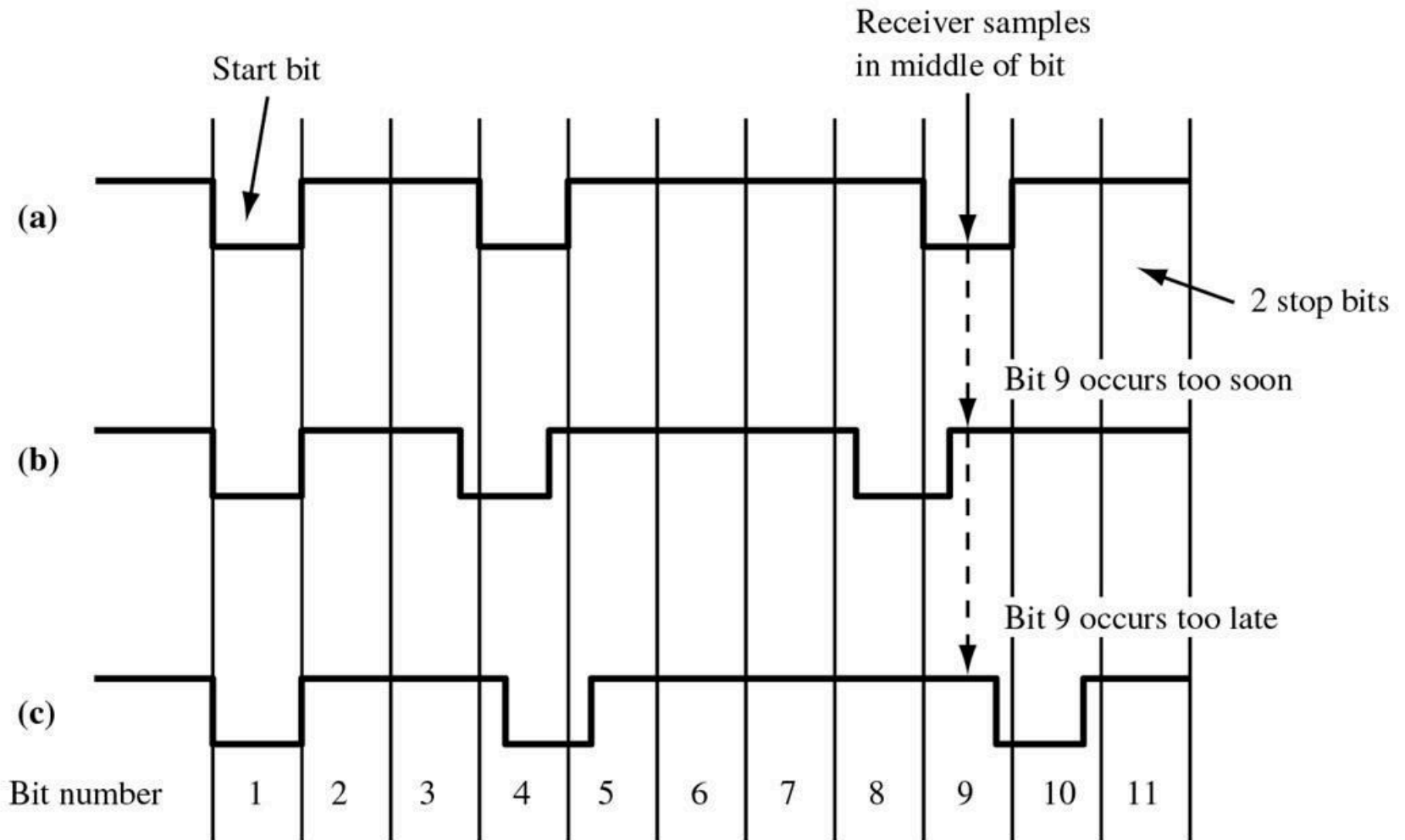
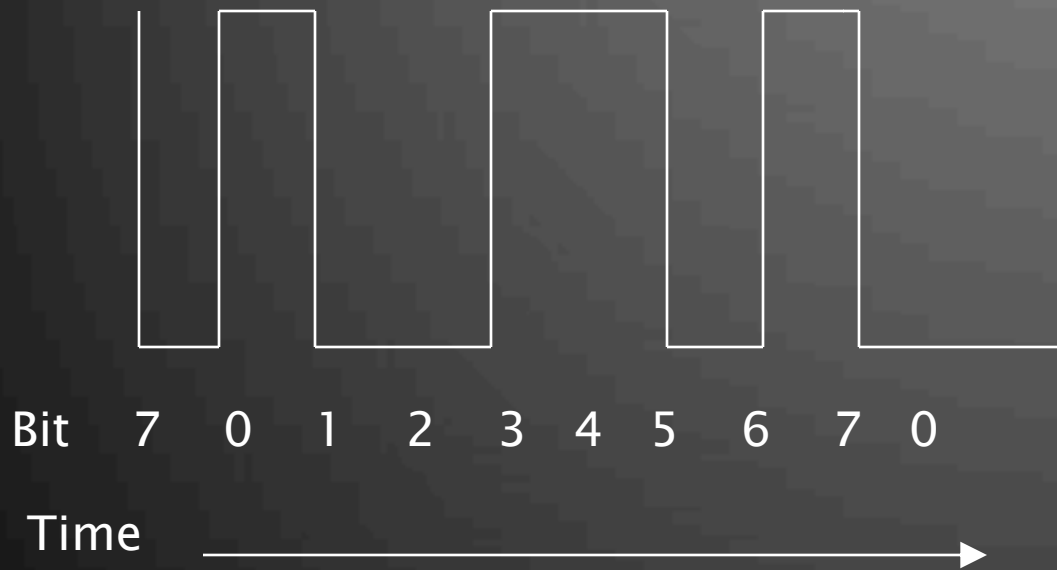


FIGURE 10-5 (a) Serial data transmitted at the proper rate. (b) The data rate is too fast. (c) The data rate is too slow.



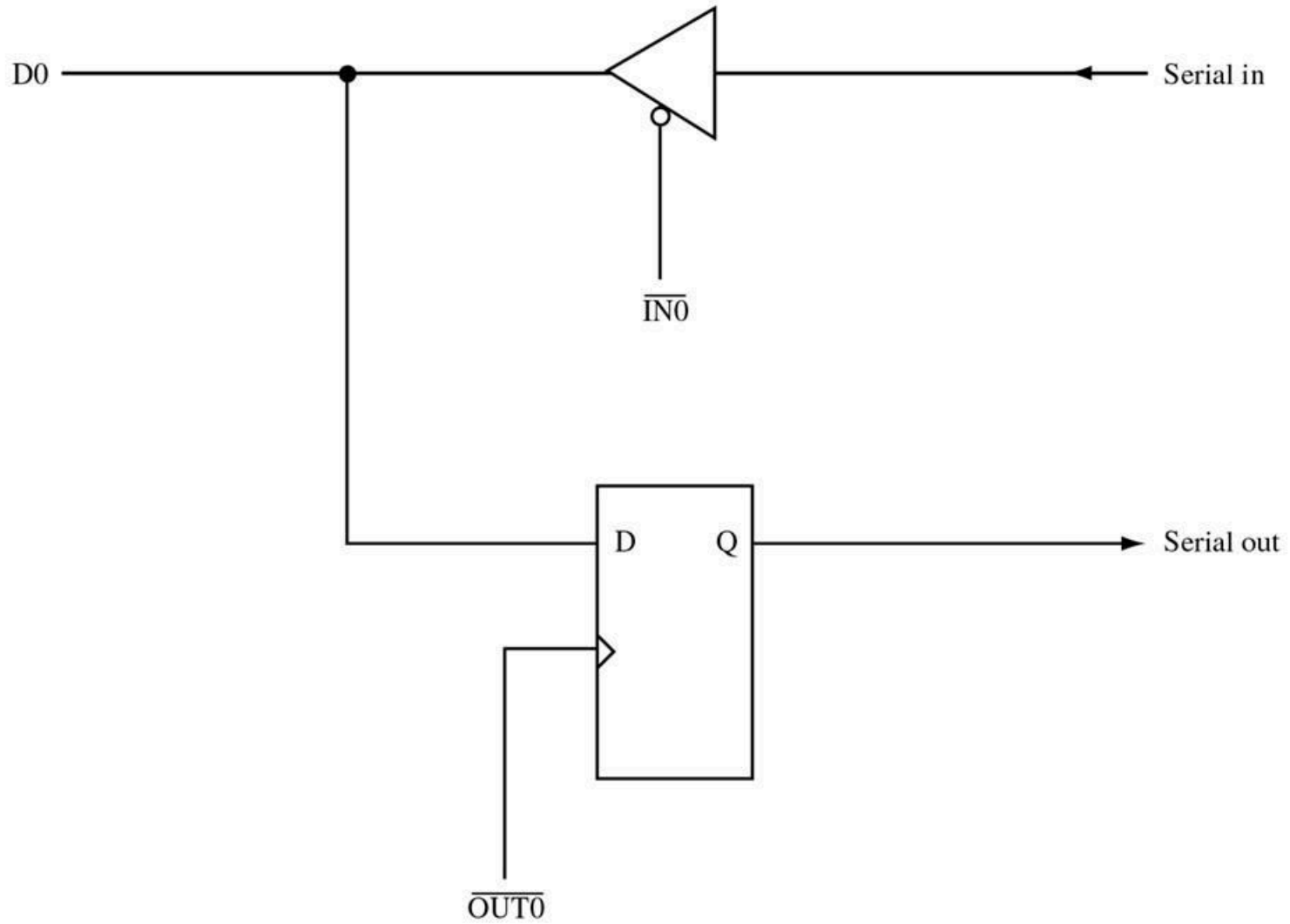
# Serial Frame (Synchronous)



No start or stop bits, timing synchronized with special ASCII characters (SYN)



FIGURE 10-2 One-bit input and output port. With appropriate software this circuit can function as a serial I/O channel.



# UART/USART

- ▶ Writing a program compatible with all different serial communication protocols is difficult and it is an inefficient use of microprocessor.

UART:

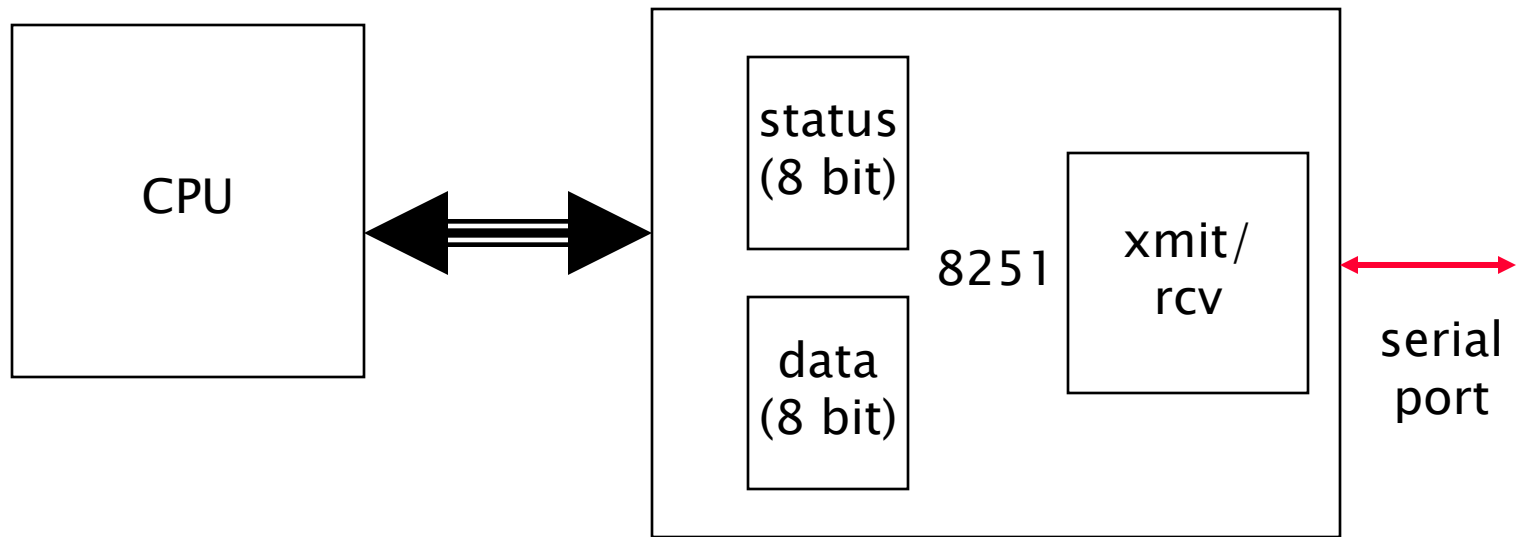
Universal Asynchronous Receiver/Transmitter chip.

USART:

Universal Synchronous/Asynchronous Receiver/Transmitter chip.

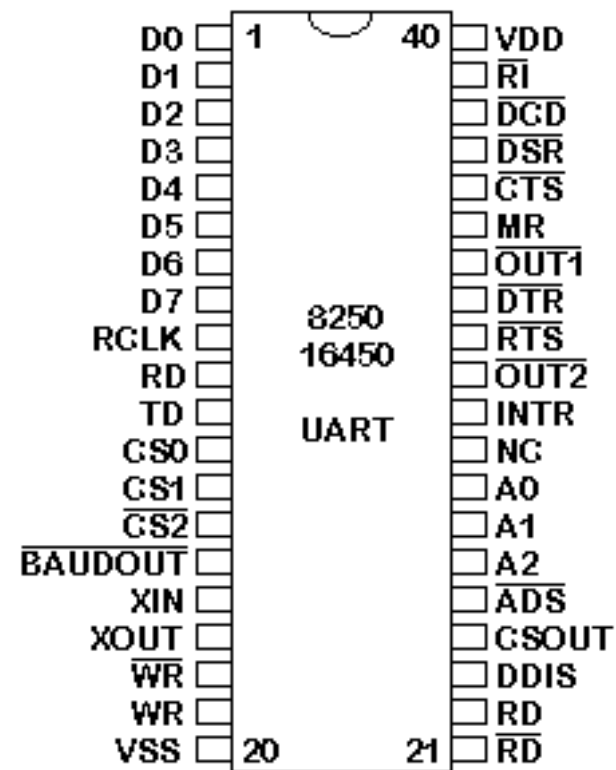
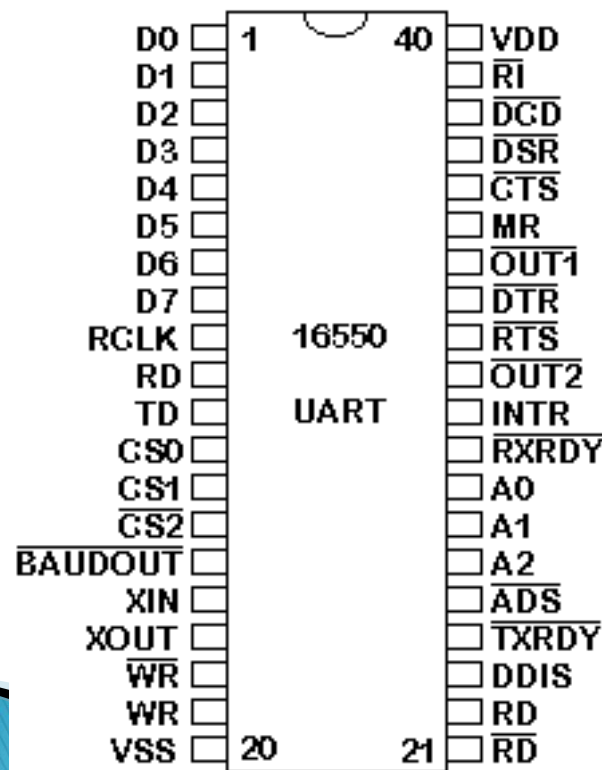
- ▶ The microprocessor sends/receives the data to the UART in parallel, while with I/O, the UART transmits/receive data serially.
- ▶ So, the UART appears to the microprocessor to be a conventional parallel port.
- ▶ 8251 functions are integrated into standard PC interface chip.

# UART / CPU interface



# UART/USART

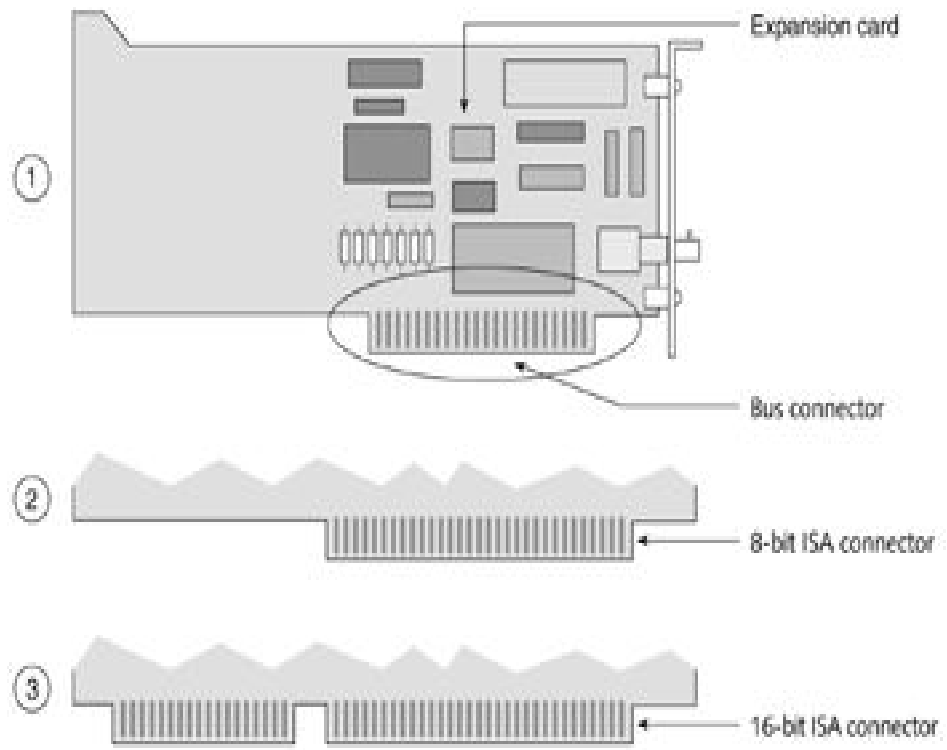
- ▶ 8251 USART
- ▶ 8250/16450 UART is a newer version of 8251.
- ▶ 16550 is the latest version UART.

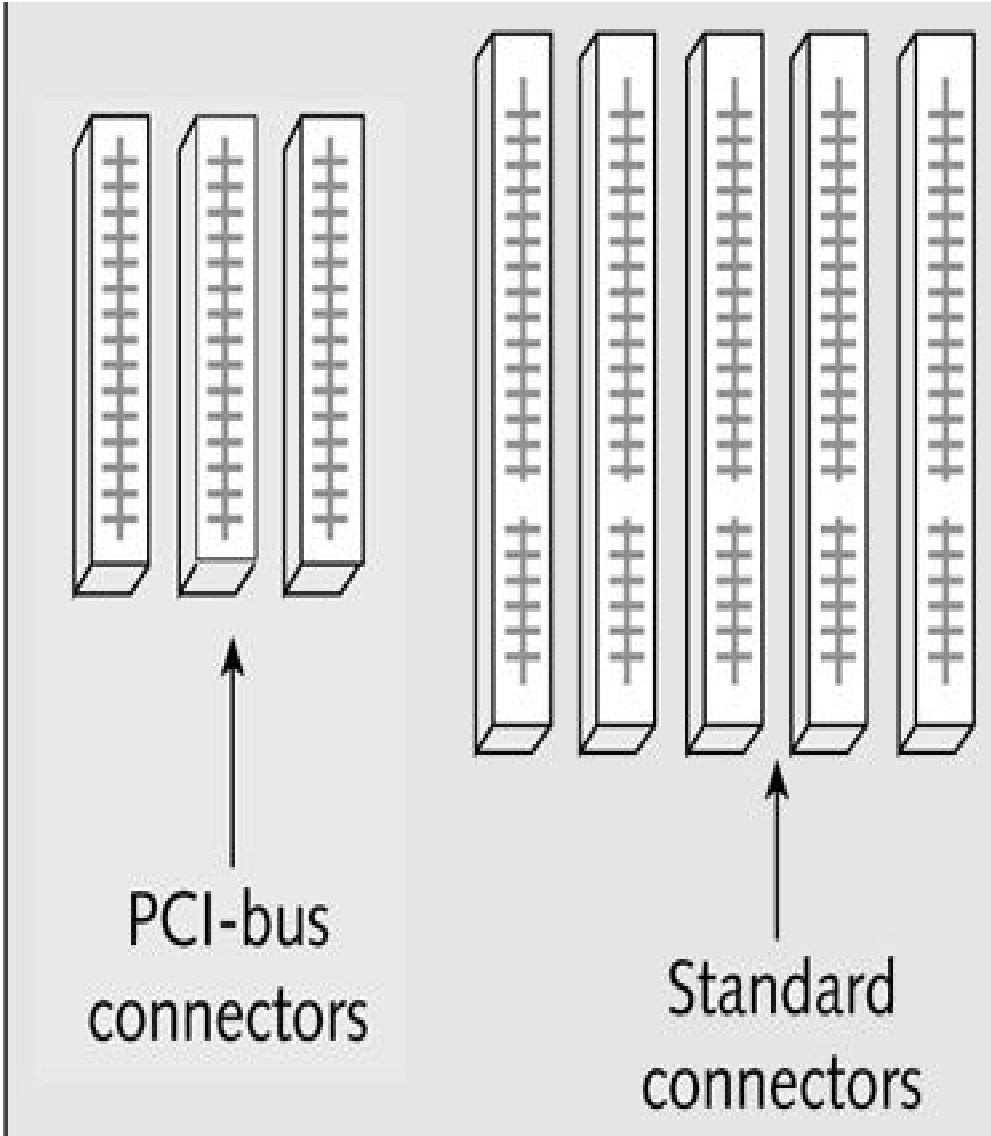


# Bus Standards

- ▶ ISA (Industry Standard Architecture)
  - 8 or 16 bit, fast disappearing
- ▶ PCI (Peripheral Component Interconnect)
  - 32 bit, still widely in use

	ISA	EISA	VL-Bus	PCI
Data path width in bits	8 or 16	32	32	32 or 64
Data bus speed in MHz	5.33 or 8.33	8.33	33	33 or 66
Data throughput in MB/sec	5.33 or 8.33	33	132	132 or 264
Maximum number of slots	8	8	2	4
Bus masters supported	No	Yes	Yes	Yes
Parity checking	No	No	No	Yes





# Graphics Cards

- ▶ Old cards are ISA or PCI
- ▶ Now: AGP (Accelerated Graphics Port)
  - Connects directly to CPU and RAM
  - Fast: 66 MHz on a 32-bit bus
  - No other devices sharing same bus



Is there any thing going to replace serial ports, parallel ports and some expansion cards.

Maybe

# USB

(Universal Serial Bus)



# ▶ Features of USB

- ▶ **One type of device cable.** USB also standardizes connectors and cables. USB cables have two connectors: an A connector and a B connector. The A connector is the end that goes into the computer, and the B connector goes into the device. The total cable length between devices must not exceed 5 meters, or 16 feet.
- ▶ **Operating System support.** USB driver support is built into the latest versions of the Windows and Apple operating systems, but Windows 98, Windows 2000, MAC OS 8.1 or higher offer much more USB support.
- ▶ **Two device speeds.** Low speed (1.5 Mbps) is mostly used for input devices such as mice and keyboards, while high speed (**up to 12 Mbps**) is used mostly for video/audio capture devices and storage devices.
- ▶ **Hot pluggable.** Devices can be attached to and detached from the computer without turning off the system. No jumper or IRQ settings are necessary.

# continue

- ▶ **Plug-and-Play.** Once the device is connected to the computer, the system automatically recognizes the device connected and installs the appropriate drivers.
- ▶ **127 peripherals.** USB makes it possible to simultaneously use and connect up to 127 devices to a single bus. The computer typically has 2 USB ports, so USB hubs are used to connect additional devices to the computer. USB hubs have multiple USB ports for connection of USB devices and for daisy chaining one or more hubs.
- ▶ **Bus-powered and self-powered.** USB supports both bus-powered and self-powered devices. Good examples of bus-powered and self-powered devices are USB hubs. USB hubs can draw power either from the host device (bus-powered) or from an external AC power supply (self-powered). Each downstream port on a bus-powered hub typically supplies up to 100 mA. On the other hand, each downstream port on a self-powered hub typically supplies up to 500mA.