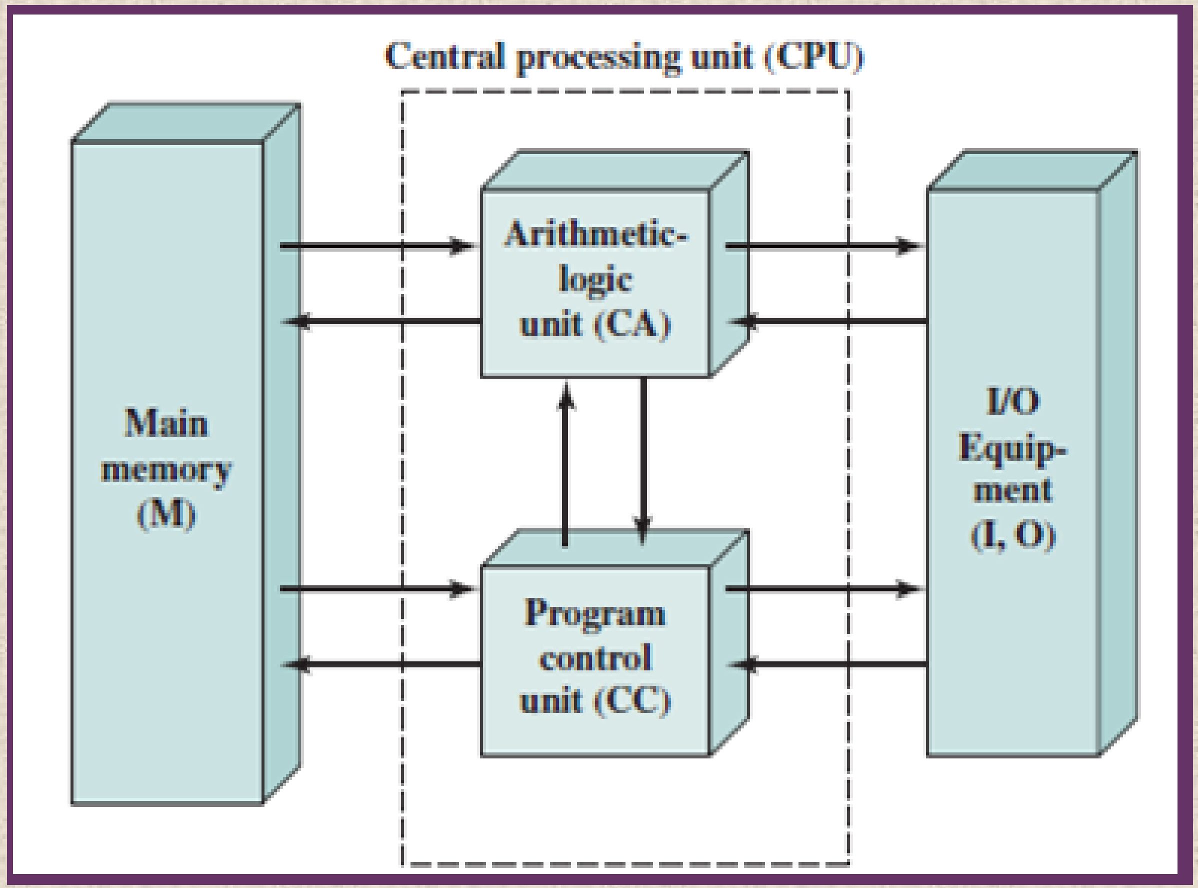
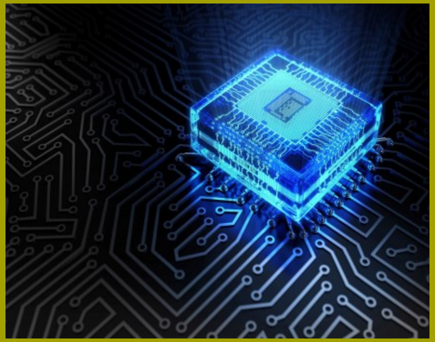
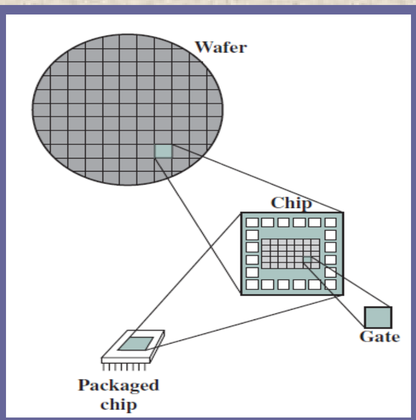
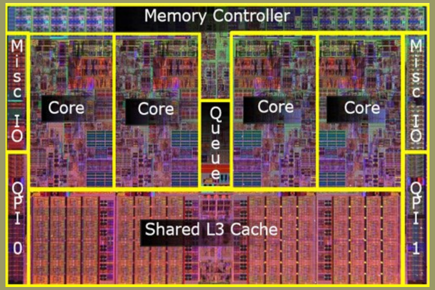
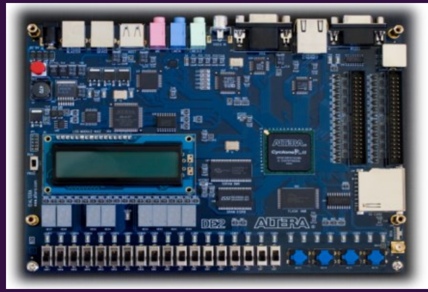








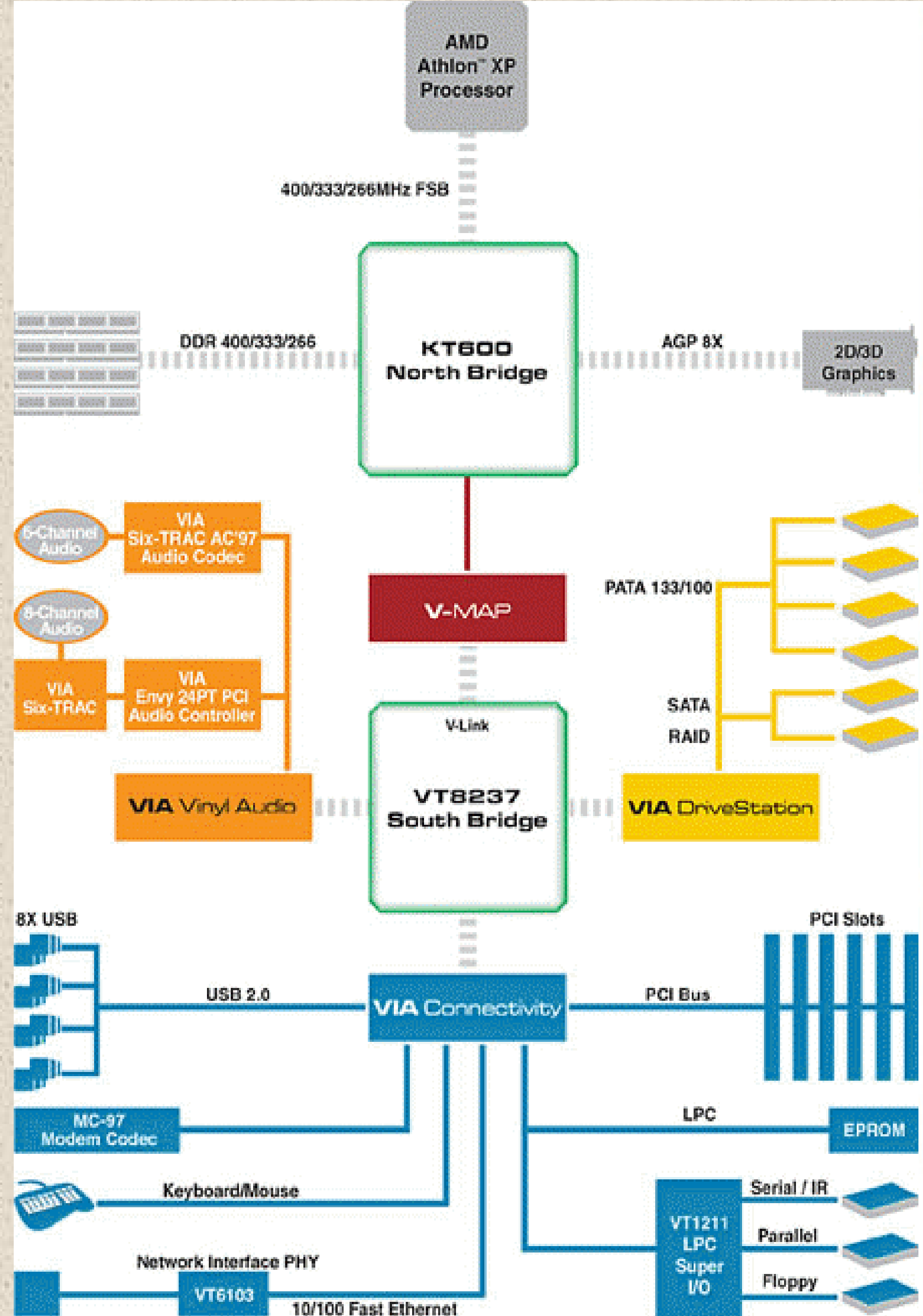


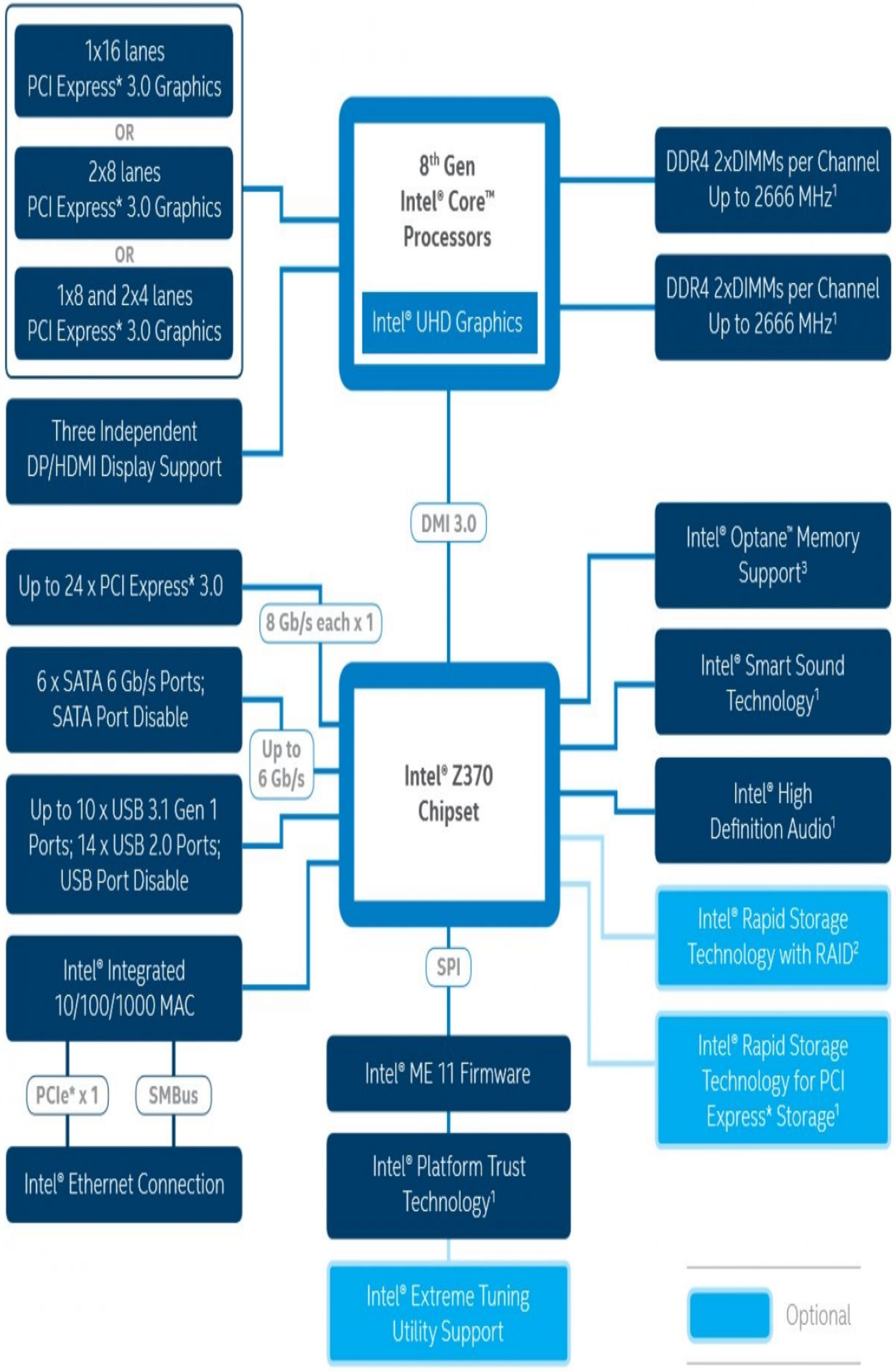
COMPUTER ORGANIZATION & ARCHITECTURE

Van-Khoa Pham (PhD.)

Chapter 7

Input/Output


AMD
Athlon™ XP
Processor
400/333/266MHz FSB
DDR 400/333/266
KT600
North Bridge
AGP 8X
2D/3D
Graphics
6-Channel
Audio
VIA
Six-TRAC AC'97
Audio Codec
8-Channel
Audio
VIA
Six-TRAC
VIA
Envy 24PT PCI
Audio Controller
V-MAP
PATA 133/100
SATA
RAID
V-Link
VT8237
South Bridge
VIA Vinyl Audio
VIA DriveStation
8X USB
USB 2.0
VIA Connectivity
PCI Bus
PCI Slots
MC-97
Modem Codec
LPC
EPROM
Keyboard/Mouse
Serial / IR
VT1211
LPC
Super
I/O
Parallel
Floppy
Network Interface PHY
VT6103
10/100 Fast Ethernet


1x16 lanes
PCI Express* 3.0 Graphics
OR
2x8 lanes
PCI Express* 3.0 Graphics
OR
1x8 and 2x4 lanes
PCI Express* 3.0 Graphics
Three Independent
DP/HDMI Display Support
8th Gen
Intel® Core™
Processors
Intel® UHD Graphics
DDR4 2xDIMMs per Channel
Up to 2666 MHz¹
DDR4 2xDIMMs per Channel
Up to 2666 MHz¹
DMI 3.0
Up to 24 x PCI Express* 3.0
8 Gb/s each x 1
6 x SATA 6 Gb/s Ports;
SATA Port Disable
Up to
6 Gb/s
Up to 10 x USB 3.1 Gen 1
Ports; 14 x USB 2.0 Ports;
USB Port Disable
Intel® Z370
Chipset
Intel® Optane™ Memory
Support³
Intel® Smart Sound
Technology¹
Intel® High
Definition Audio¹
Intel® Rapid Storage
Technology with RAID²
Intel® Rapid Storage
Technology for PCI
Express* Storage¹
Intel® Integrated
10/100/1000 MAC
PCIe* x 1
SMBus
Intel® Ethernet Connection
SPI
Intel® ME 11 Firmware
Intel® Platform Trust
Technology¹
Intel® Extreme Tuning
Utility Support
Optional

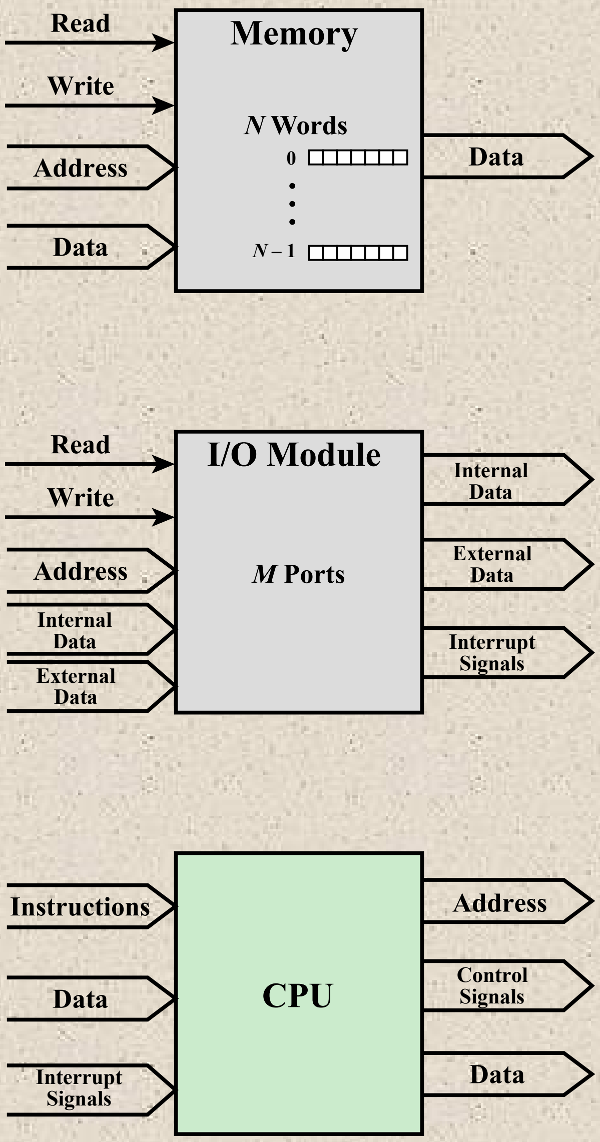


Figure 3.15 Computer Modules

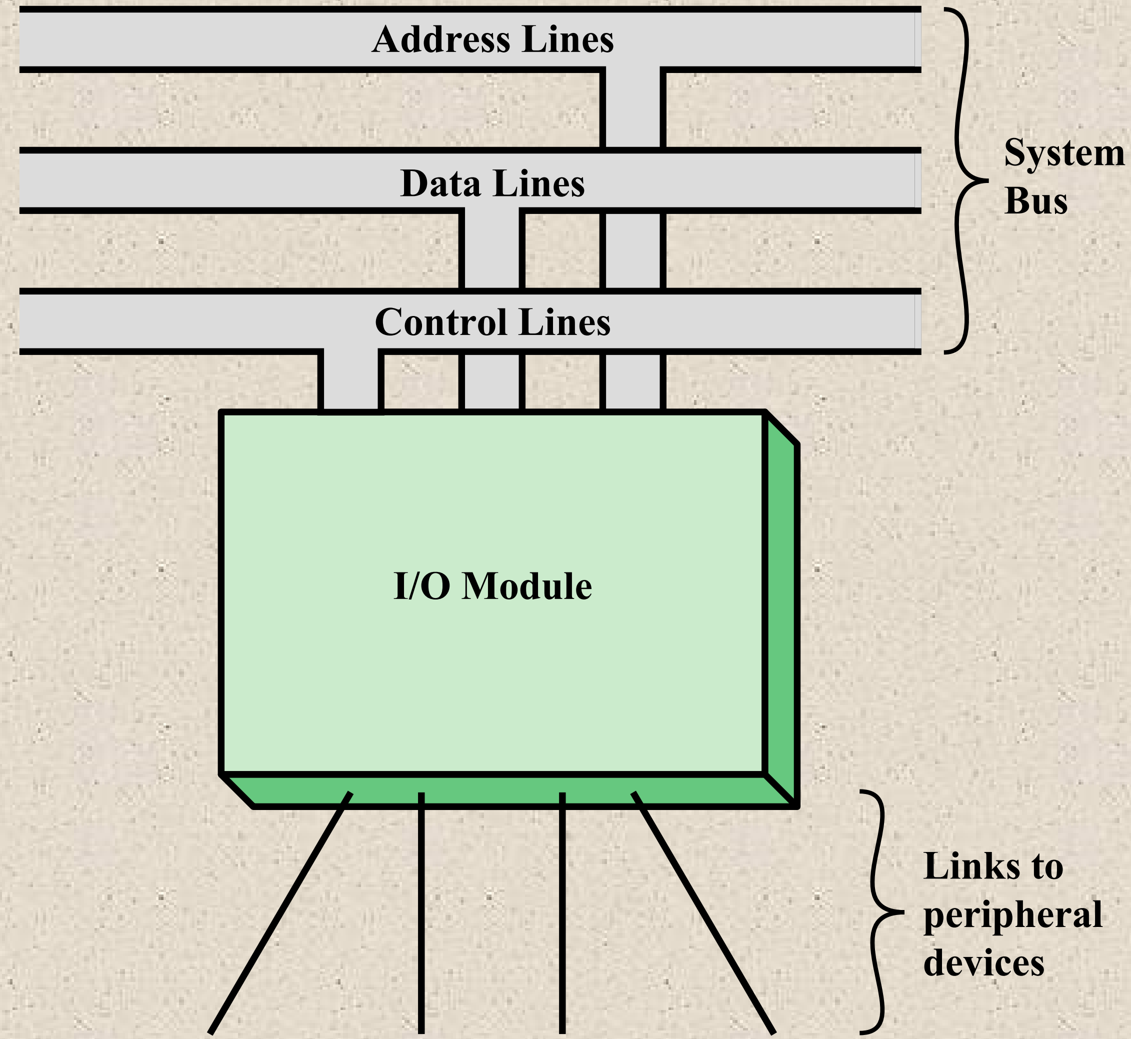
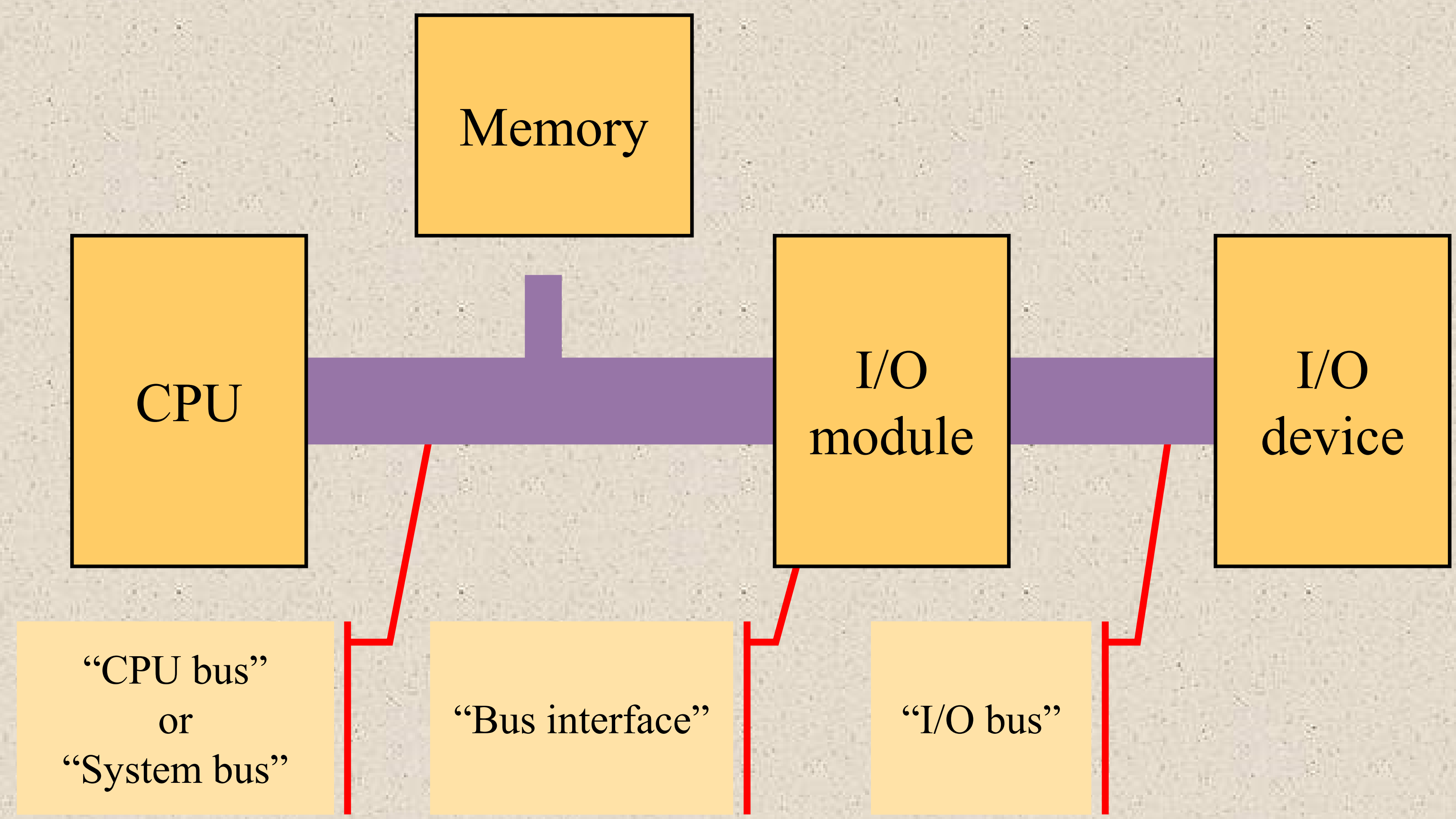


Figure 7.1 Generic Model of an I/O Module

CPU-Memory-I/O Architecture



why does not connect peripherals directly to the system bus?

- There are a wide variety of peripherals with various methods of operation. It would be impractical to incorporate the necessary logic within the processor to control a range of devices.
- The data transfer rate of peripherals is often **much slower than** that of the memory or processor. Thus, it is impractical to use the high-speed system bus to communicate directly with a peripheral.
- The data transfer rate of some peripherals is faster than that of the memory or processor. Again, the mismatch would lead to inefficiencies if not managed properly.
- Peripherals often use different data formats and word lengths than the computer to which they are attached.

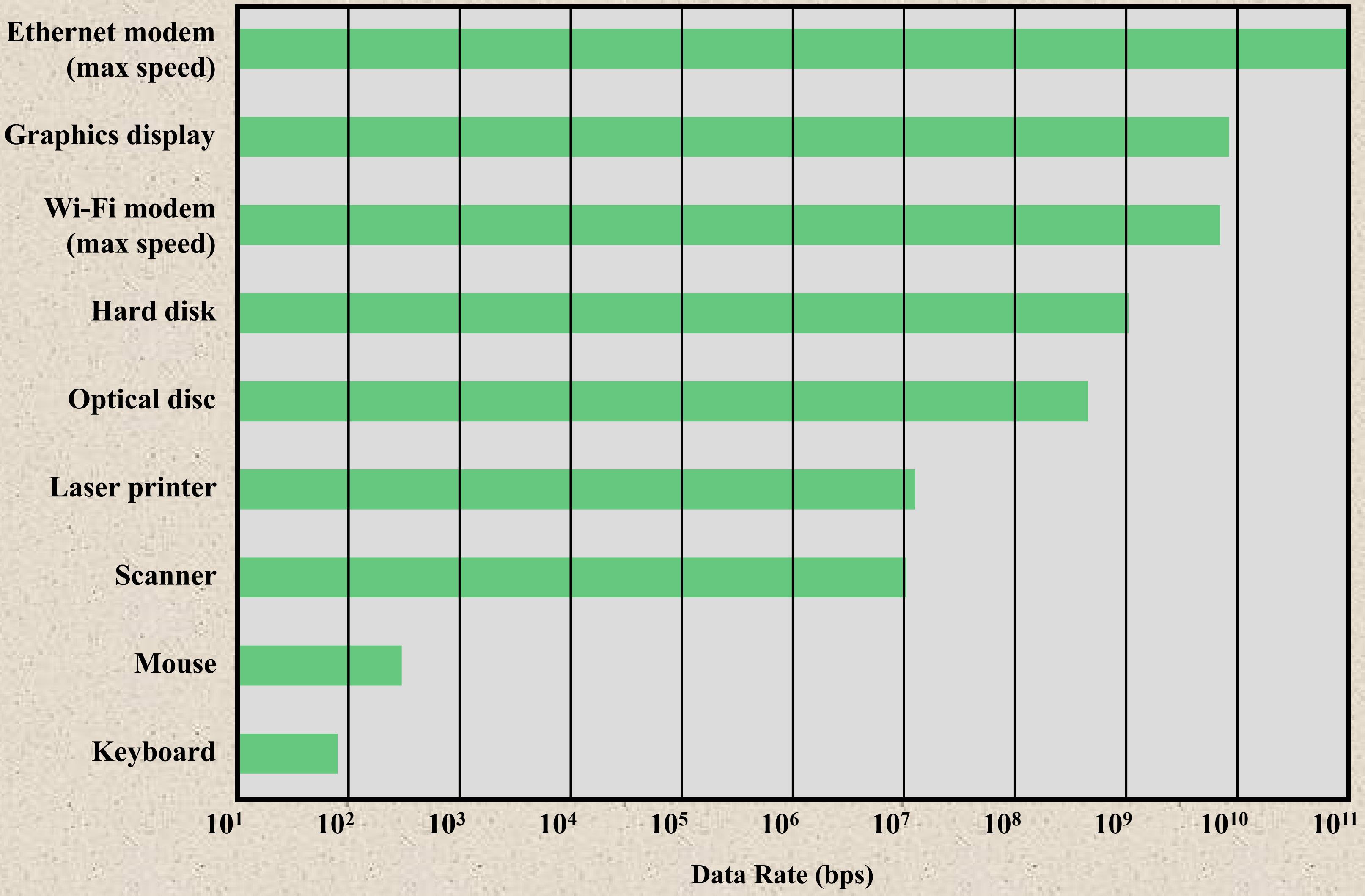


Figure 2.1 Typical I/O Device Data Rates

External Devices

- Provide a means of exchanging data between the external environment and the computer
- Attach to the computer by a link to an I/O module
  - The link is used to exchange control, status, and data between the I/O module and the external device
- *Peripheral device*
  - An external device connected to an I/O module

Three categories:

- Human readable
  - Suitable for communicating with the computer user
  - Video display terminals (VDTs), printers
- Machine readable
  - Suitable for communicating with equipment
  - Magnetic disk and tape systems, sensors and actuators
- Communication
  - Suitable for communicating with remote devices such as a terminal, a machine readable device, or another computer

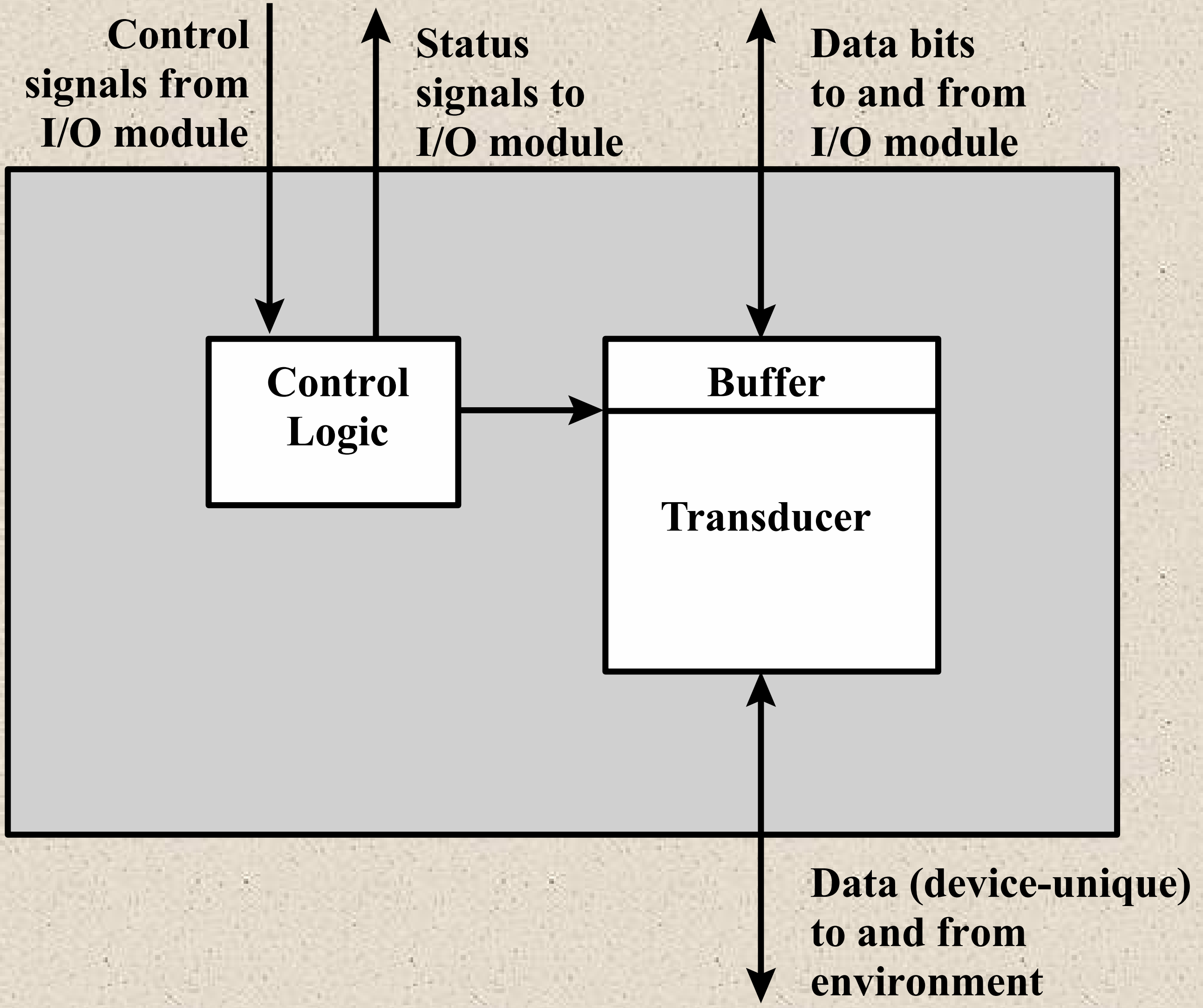


Figure 7.2 Block Diagram of an External Device

+

Keyboard/Monitor

- Most common means of computer/user interaction
- User provides input through the keyboard
- The monitor displays data provided by the computer

International Reference Alphabet (IRA)

- Basic unit of exchange is the character
  - Associated with each character is a code
  - Each character in this code is represented by a unique 7-bit binary code
    - 128 different characters can be represented
- Characters are of two types:
  - Printable
    - Alphabetic, numeric, and special characters that can be printed on paper or displayed on a screen
  - Control
    - Have to do with controlling the printing or displaying of characters
    - Example is carriage return
    - Other control characters are concerned with communications procedures

Keyboard Codes

- When the user depresses a key it generates an electronic signal that is interpreted by the transducer in the keyboard and translated into the bit pattern of the corresponding IRA code
- This bit pattern is transmitted to the I/O module in the computer
- On output, IRA code characters are transmitted to an external device from the I/O module
- The transducer interprets the code and sends the required electronic signals to the output device either to display the indicated character or perform the requested control function

The major functions for an I/O module fall into the following categories:

Control and timing

- Coordinates the flow of traffic between internal resources and external devices

Processor communication

- Involves command decoding, data, status reporting, address recognition

Device communication

- Involves commands, status information, and data

Data buffering

- Performs the needed buffering operation to balance device and memory speeds

Error detection

- Detects and reports transmission errors

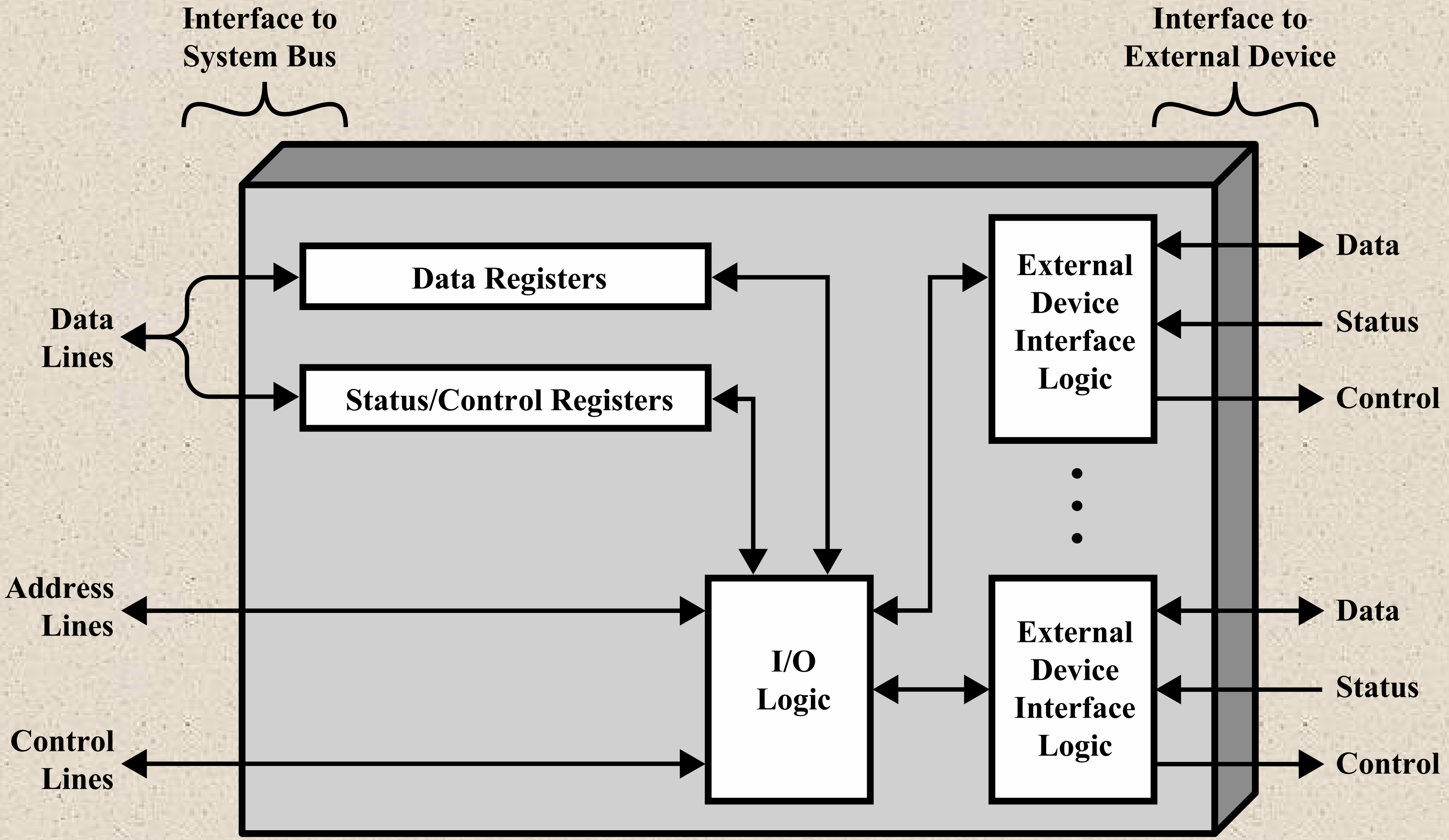


Figure 7.3 Block Diagram of an I/O Module

+

Programmed I/O

Three techniques are possible for I/O operations:

- Programmed I/O
  - Data are exchanged between the processor and the I/O module
  - Processor executes a program that gives it direct control of the I/O operation
  - When the processor issues a command it must wait until the I/O operation is complete
  - If the processor is faster than the I/O module this is wasteful of processor time
- Interrupt-driven I/O
  - Processor issues an I/O command, continues to execute other instructions, and is interrupted by the I/O module when the latter has completed its work
- Direct memory access (DMA)
  - The I/O module and main memory exchange data directly without processor involvement

Table 7.1
I/O Techniques

	No Interrupts	Use of Interrupts
I/O-to-memory transfer through processor	Programmed I/O	Interrupt-driven I/O
Direct I/O-to-memory transfer		Direct memory access (DMA)

I/O Commands

- There are four types of I/O commands that an I/O module may receive when it is addressed by a processor:

1) Control

   - used to activate a peripheral and tell it what to do

2) Test

   - used to test various status conditions associated with an I/O module and its peripherals

3) Read

   - causes the I/O module to obtain an item of data from the peripheral and place it in an internal buffer

4) Write

   - causes the I/O module to take an item of data from the data bus and subsequently transmit that data item to the peripheral

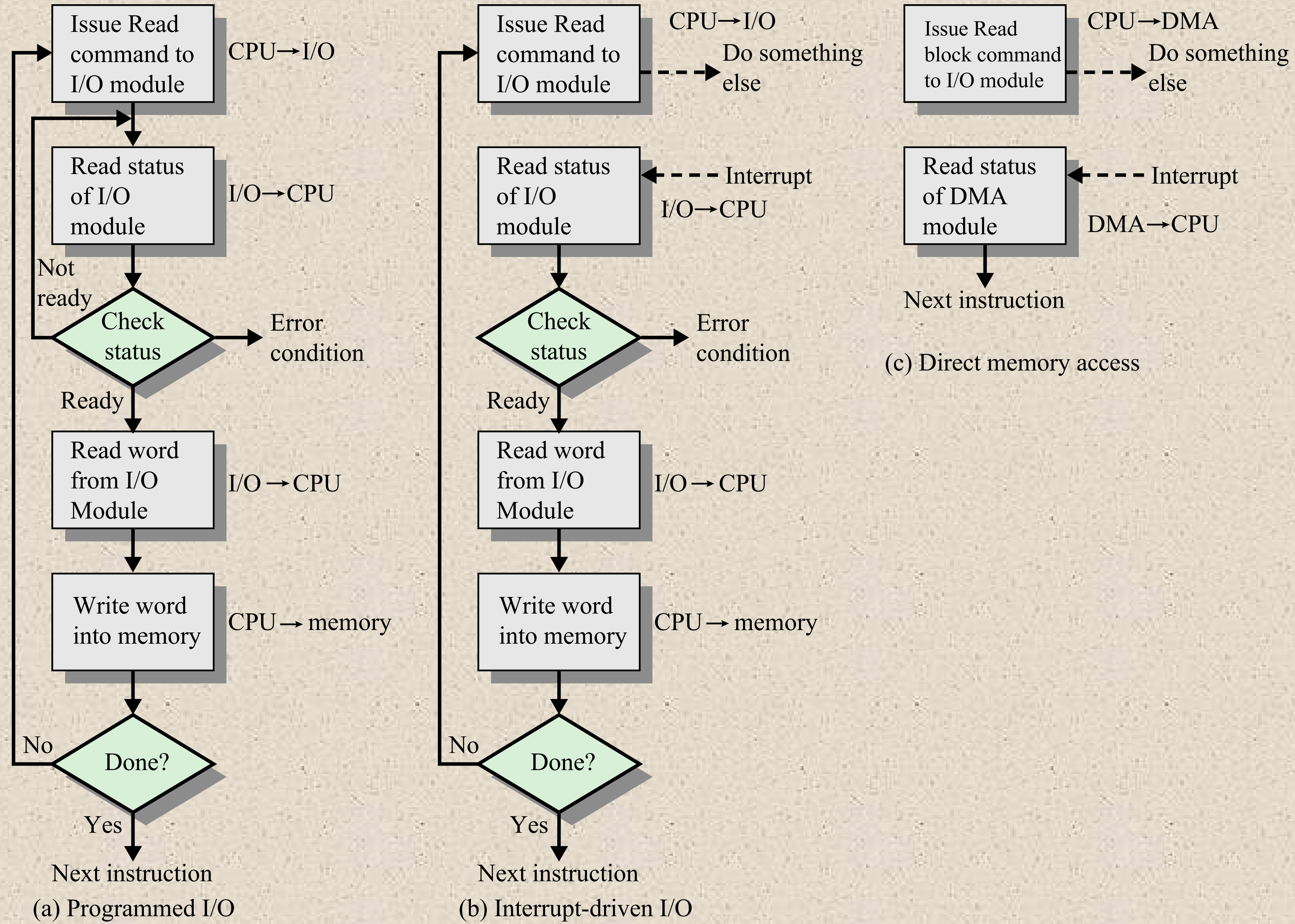


Figure 7.4 Three Techniques for Input of a Block of Data

I/O Instructions

With programmed I/O there is a close correspondence between the I/O-related instructions that the processor fetches from memory and the I/O commands that the processor issues to an I/O module to execute the instructions

The form of the instruction depends on the way in which external devices are addressed

Each I/O device connected through I/O modules is given a unique identifier or address

When the processor issues an I/O command, the command contains the address of the desired device

Thus each I/O module must interpret the address lines to determine if the command is for itself

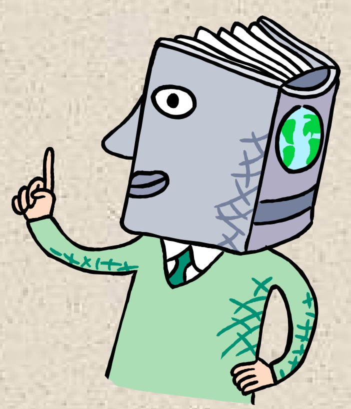
Memory-mapped I/O

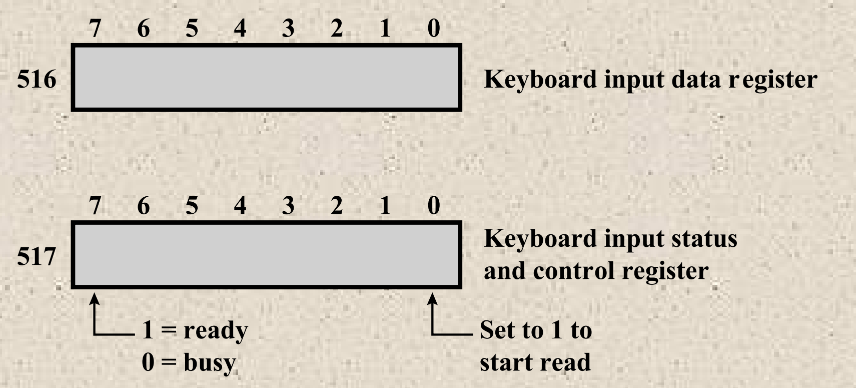
There is a single address space for memory locations and I/O devices

A single read line and a single write line are needed on the bus

I/O Mapping Summary

- Memory mapped I/O
  - Devices and memory share an address space
  - I/O looks just like memory read/write
  - No special commands for I/O
    - Large selection of memory access commands available
- Isolated I/O
  - Separate address spaces
  - Need I/O or memory select lines
  - Special commands for I/O
    - Limited set





ADDRESS	INSTRUCTION	OPERAND	COMMENT
200	Load AC	"1"	Load accumulator
	Store AC	517	Initiate keyboard read
202	Load AC	517	Get status byte
	Branch if Sign = 0	202	Loop until ready
	Load AC	516	Load data byte

(a) Memory-mapped I/O

ADDRESS	INSTRUCTION	OPERAND	COMMENT
200	Load I/O	5	Initiate keyboard read
201	Test I/O	5	Check for completion
	Branch Not Ready	201	Loop until complete
	In	5	Load data byte

(b) Isolated I/O

Figure 7.5 Memory-Mapped and Isolated I/O

Interrupt-Driven I/O

The problem with programmed I/O is that the processor has to wait a long time for the I/O module to be ready for either reception or transmission of data

An alternative is for the processor to issue an I/O command to a module and then go on to do some other useful work

The I/O module will then interrupt the processor to request service when it is ready to exchange data with the processor

The processor executes the data transfer and resumes its former processing

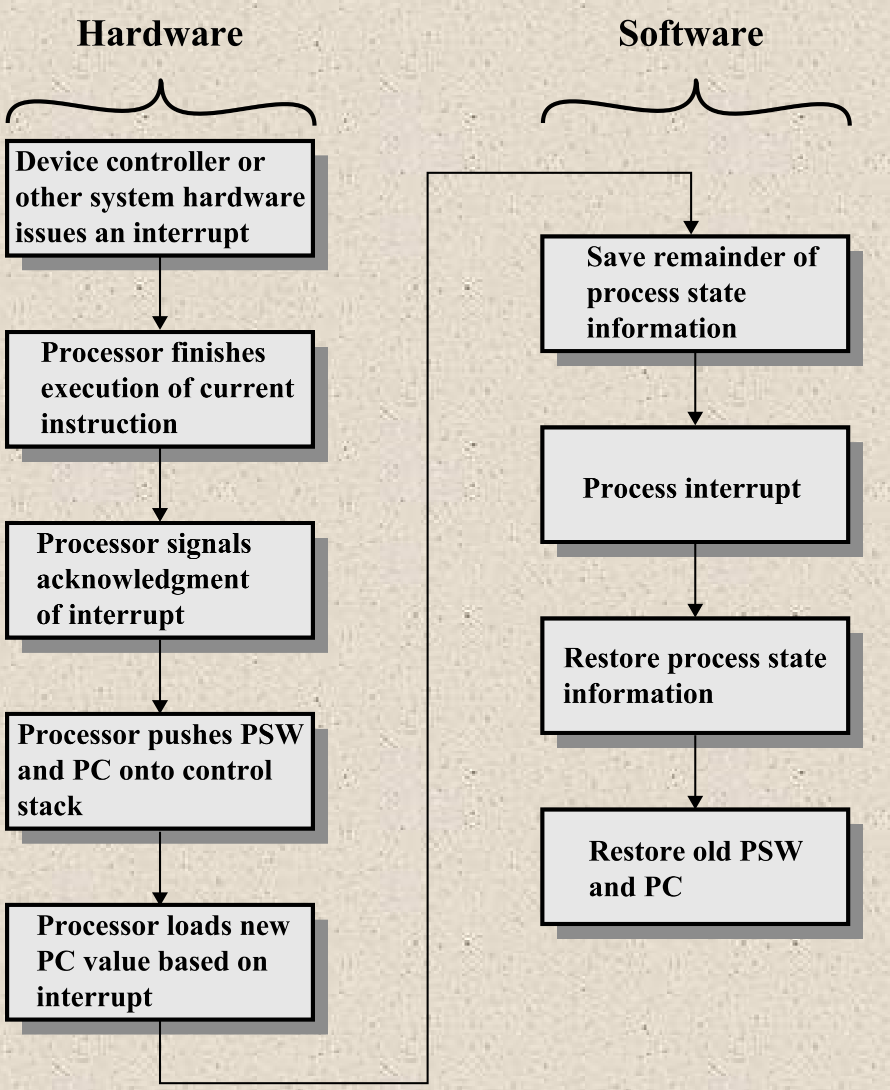


Figure 7.6 Simple Interrupt Processing

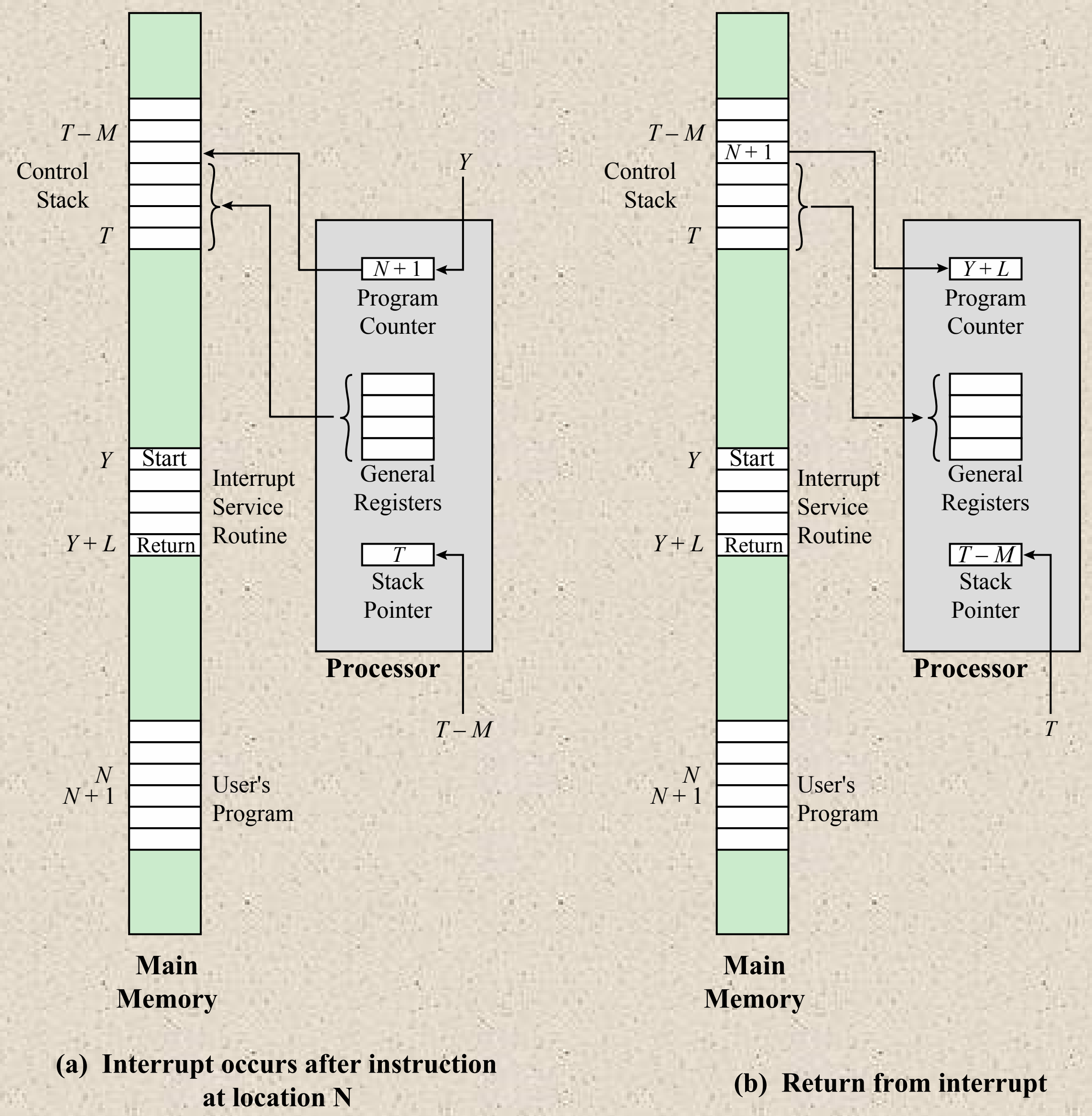


Figure 7.7 Changes in Memory and Registers for an Interrupt

Design Issues

Two design issues arise in implementing interrupt I/O:

- Because there will be multiple I/O modules how does the processor determine which device issued the interrupt?
- If multiple interrupts have occurred how does the processor decide which one to process?

+

Device Identification

Four general categories of techniques are in common use:

- **Multiple interrupt lines**
  - Between the processor and the I/O modules
  - Most straightforward approach to the problem
  - Consequently even if multiple lines are used, it is likely that each line will have multiple I/O modules attached to it

- **Software poll**
  - When processor detects an interrupt it branches to an interrupt-service routine whose job is to poll each I/O module to determine which module caused the interrupt
  - Time consuming

- **Daisy chain (hardware poll, vectored)**
  - The interrupt acknowledge line is daisy chained through the modules
  - Vector – address of the I/O module or some other unique identifier
  - Vectored interrupt – processor uses the vector as a pointer to the appropriate device-service routine, avoiding the need to execute a general interrupt-service routine first

- **Bus arbitration (vectored)**
  - An I/O module must first gain control of the bus before it can raise the interrupt request line
  - When the processor detects the interrupt it responds on the interrupt acknowledge line
  - Then the requesting module places its vector on the data lines

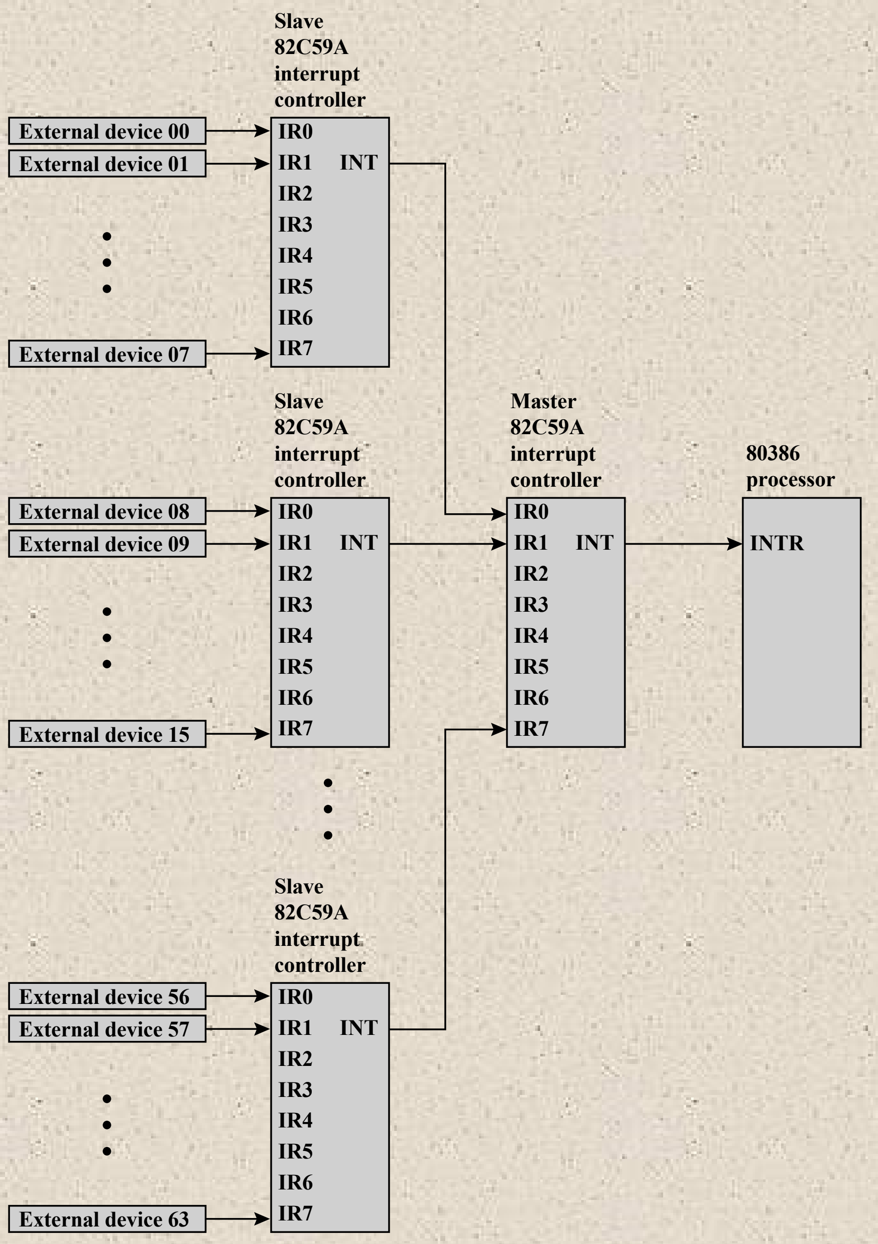


Figure 7.8 Use of the 82C59A Interrupt Controller

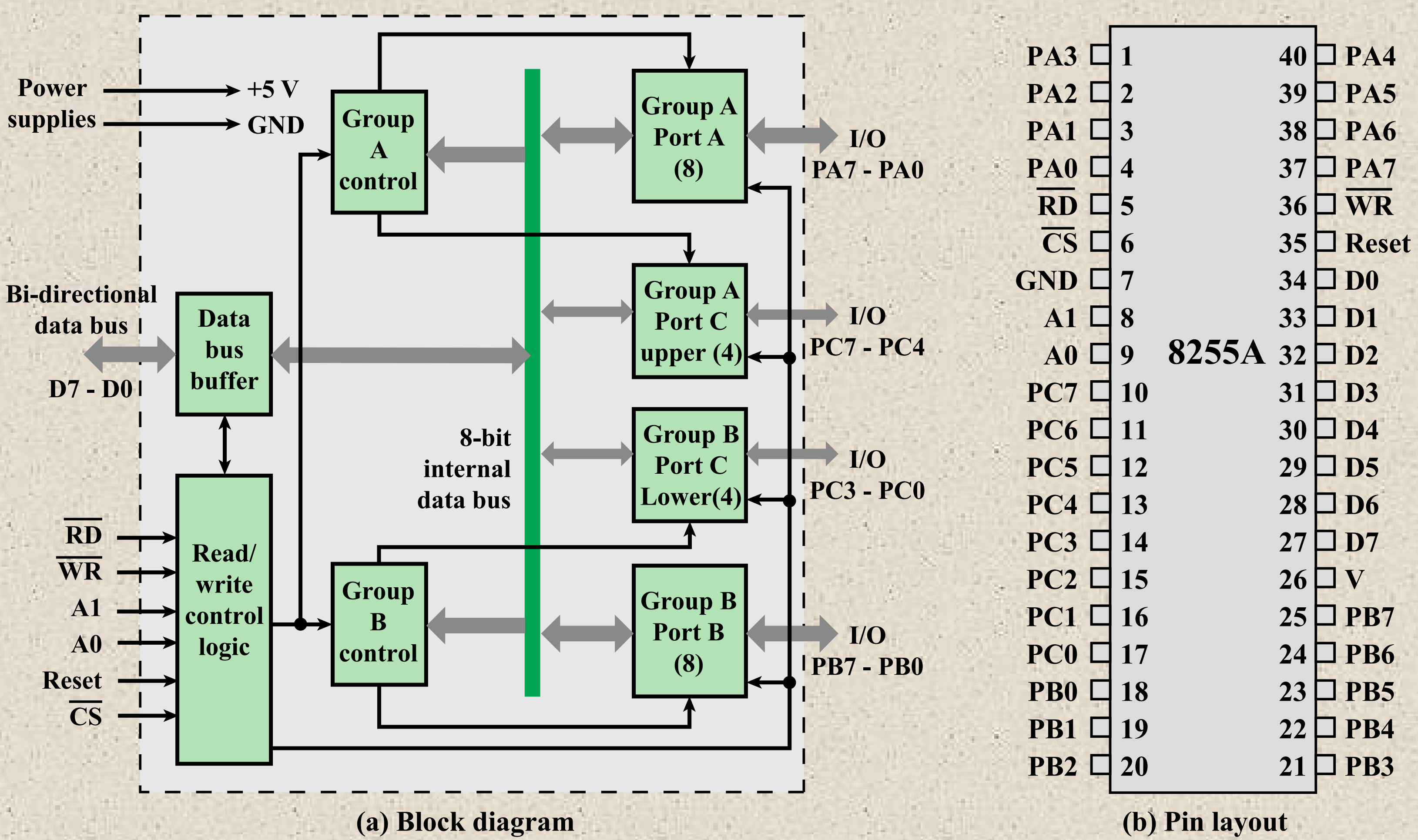
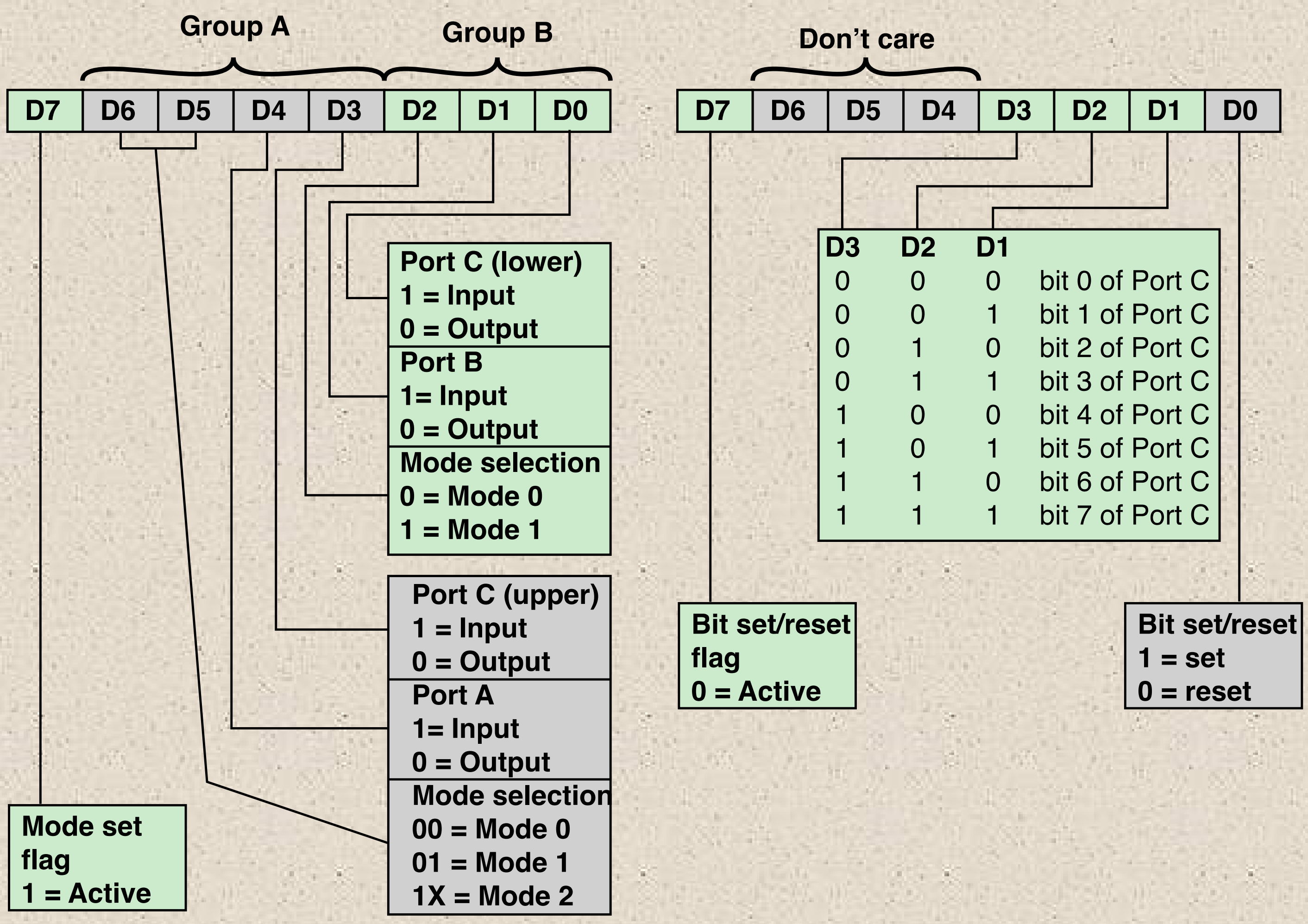


Figure 7.9 The Intel 8255A Programmable Peripheral Interface



(a) Mode definition of the 8255 control register to configure the 8255

(b) Bit definitions of the 8255 control register to modify single bits of port C

Figure 7.10 The Intel 8255A Control Word

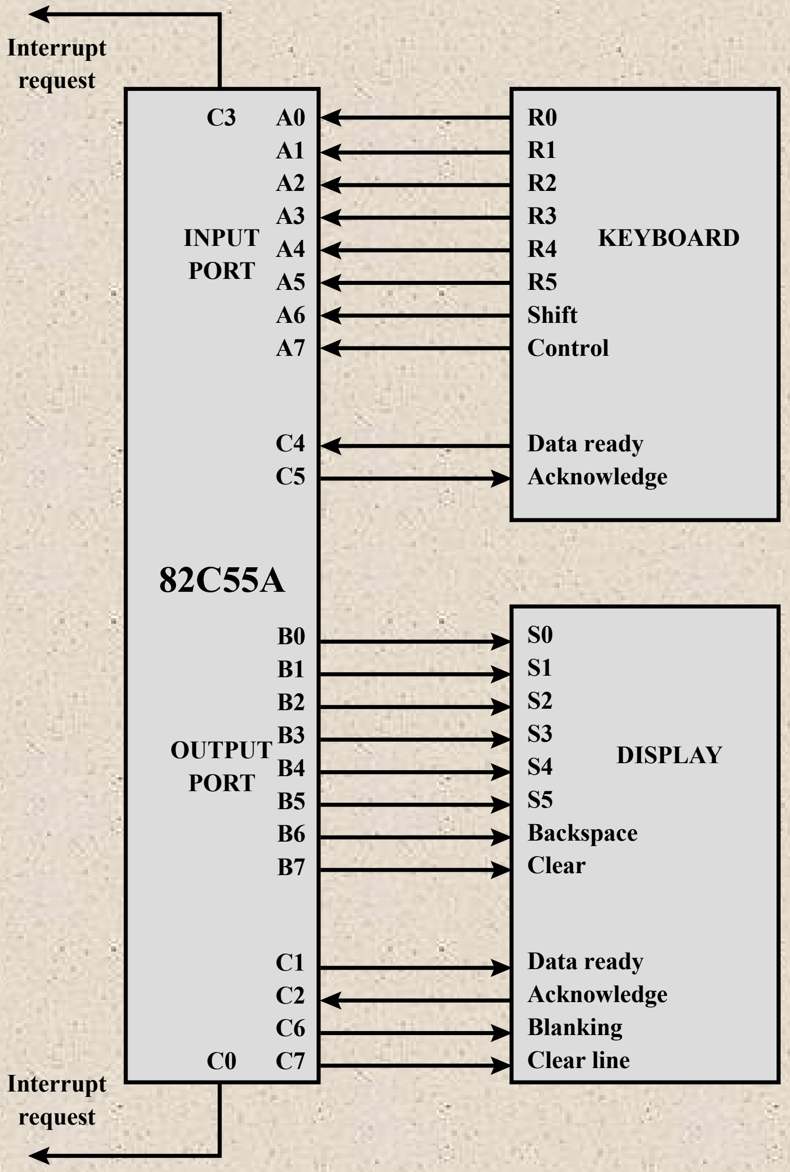


Figure 7.11 Keyboard/Display Interface to 82C55A

Summary

Chapter 7

Input/Output

- External devices
  - Keyboard/monitor
  - Disk drive
- I/O modules
  - Module function
  - I/O module structure
- Programmed I/O
  - Overview of programmed I/O
  - I/O commands/instructions
- Direct memory access
  - Drawbacks of programmed and interrupt-driven I/O
  - DMA function
  - Intel 8237A DMA controller
- Interrupt-driven I/O
  - Interrupt processing
  - Design issues
  - Intel 82C59A interrupt controller
  - Intel 82C55A programmable peripheral interface