William Stallings
Computer Organization and Architecture
9th Edition

Chapter 3

A Top-Level View of Computer Function and Interconnection

Computer Components

- Contemporary computer designs are based on concepts developed by John von Neumann at the Institute for Advanced Studies, Princeton
- Referred to as the *von Neumann architecture* and is based on three key concepts:
  - Data and instructions are stored in a single read-write memory
  - The contents of this memory are addressable by location, without regard to the type of data contained there
  - Execution occurs in a sequential fashion (unless explicitly modified) from one instruction to the next
- *Hardwired program*
  - The result of the process of connecting the various components in the desired configuration

Hardware and Software Approaches

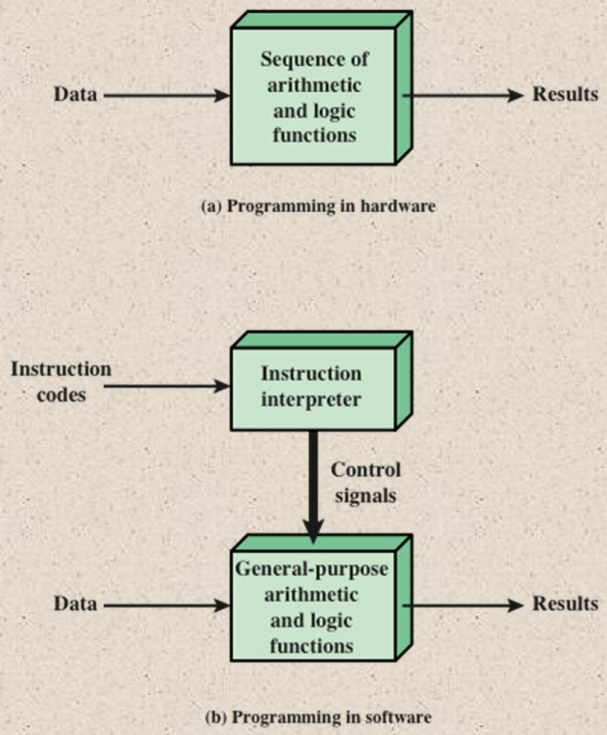


Figure 3.1 Hardware and Software Approaches

Software

- A sequence of codes or instructions
- Part of the hardware interprets each instruction and generates control signals
- Provide a new sequence of codes for each new program instead of rewiring the hardware

Major components:

- CPU
  - Instruction interpreter
  - Module of general-purpose arithmetic and logic functions
- I/O Components
  - Input module
    - Contains basic components for accepting data and instructions and converting them into an internal form of signals usable by the system
  - Output module
    - Means of reporting results

Software

I/O Components

Memory address register (MAR)

- Specifies the address in memory for the next read or write

Memory buffer register (MBR)

- Contains the data to be written into memory or receives the data read from memory

I/O address register (I/OAR)

- Specifies a particular I/O device

I/O buffer register (I/OBR)

- Used for the exchange of data between an I/O module and the CPU

MEMORY

MAR

MBR

Computer Components: Top Level View

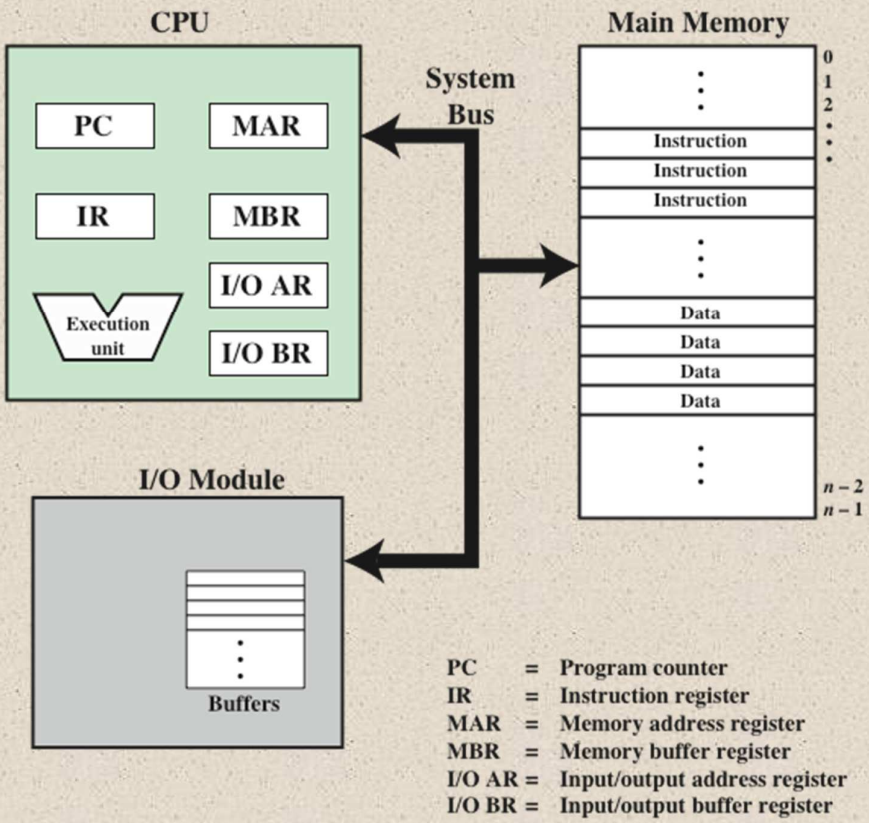


Figure 3.2 Computer Components: Top-Level View

Basic Instruction Cycle

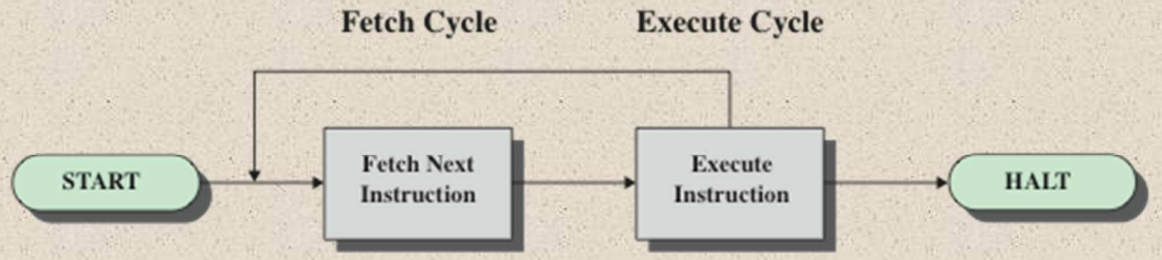
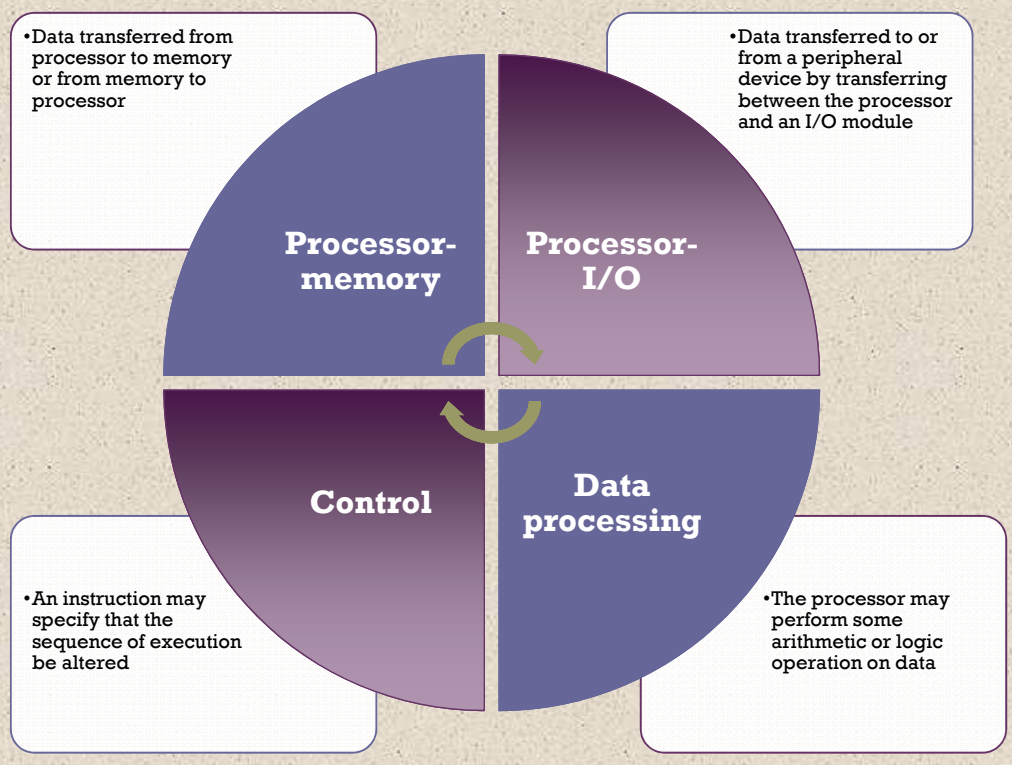


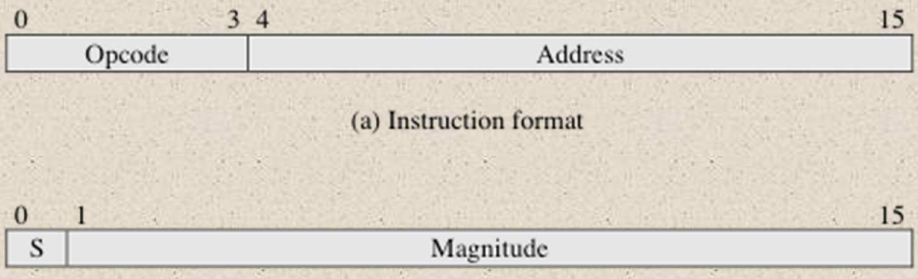
Figure 3.3 Basic Instruction Cycle

Fetch Cycle

- At the beginning of each instruction cycle the processor fetches an instruction from memory
- The program counter (PC) holds the address of the instruction to be fetched next
- The processor increments the PC after each instruction fetch so that it will fetch the next instruction in sequence
- The fetched instruction is loaded into the instruction register (IR)
- The processor interprets the instruction and performs the required action

Action Categories





(a) Instruction format

(b) Integer format

Program Counter (PC) = Address of instruction
Instruction Register (IR) = Instruction being executed
Accumulator (AC) = Temporary storage

(c) Internal CPU registers

0001 = Load AC from Memory
0010 = Store AC to Memory
0101 = Add to AC from Memory

(d) Partial list of opcodes

Figure 3.4 Characteristics of a Hypothetical Machine

Example of Program Execution

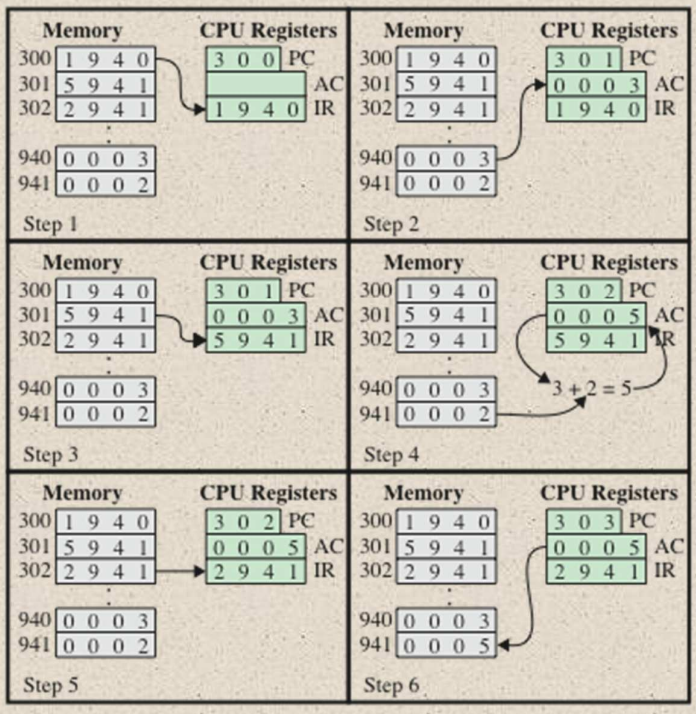


Figure 3.5 Example of Program Execution (contents of memory and registers in hexadecimal)

Instruction Cycle State Diagram

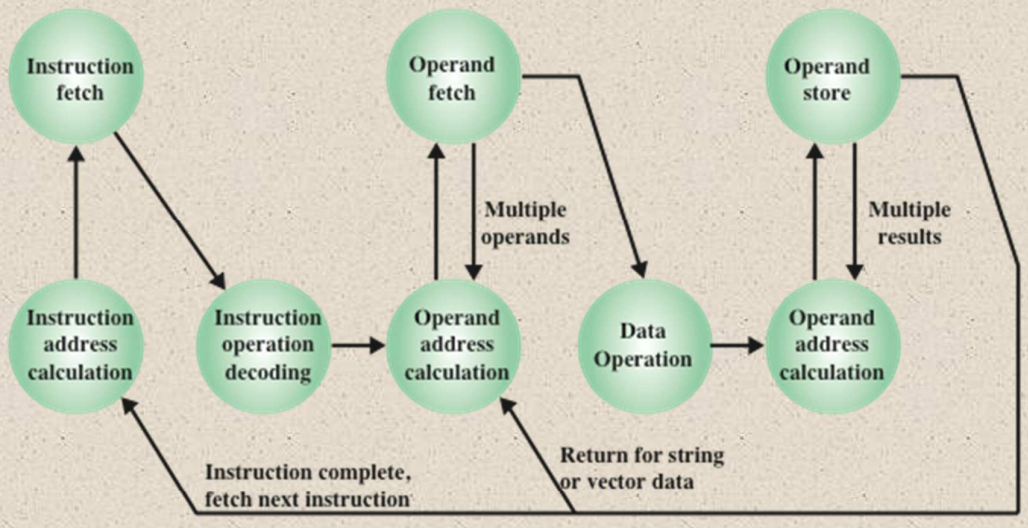


Figure 3.6 Instruction Cycle State Diagram

Classes of Interrupts

Program	Generated by some condition that occurs as a result of an instruction execution, such as arithmetic overflow, division by zero, attempt to execute an illegal machine instruction, or reference outside a user's allowed memory space.
Timer	Generated by a timer within the processor. This allows the operating system to perform certain functions on a regular basis.
I/O	Generated by an I/O controller, to signal normal completion of an operation, request service from the processor, or to signal a variety of error conditions.
Hardware failure	Generated by a failure such as power failure or memory parity error.

Program Flow Control

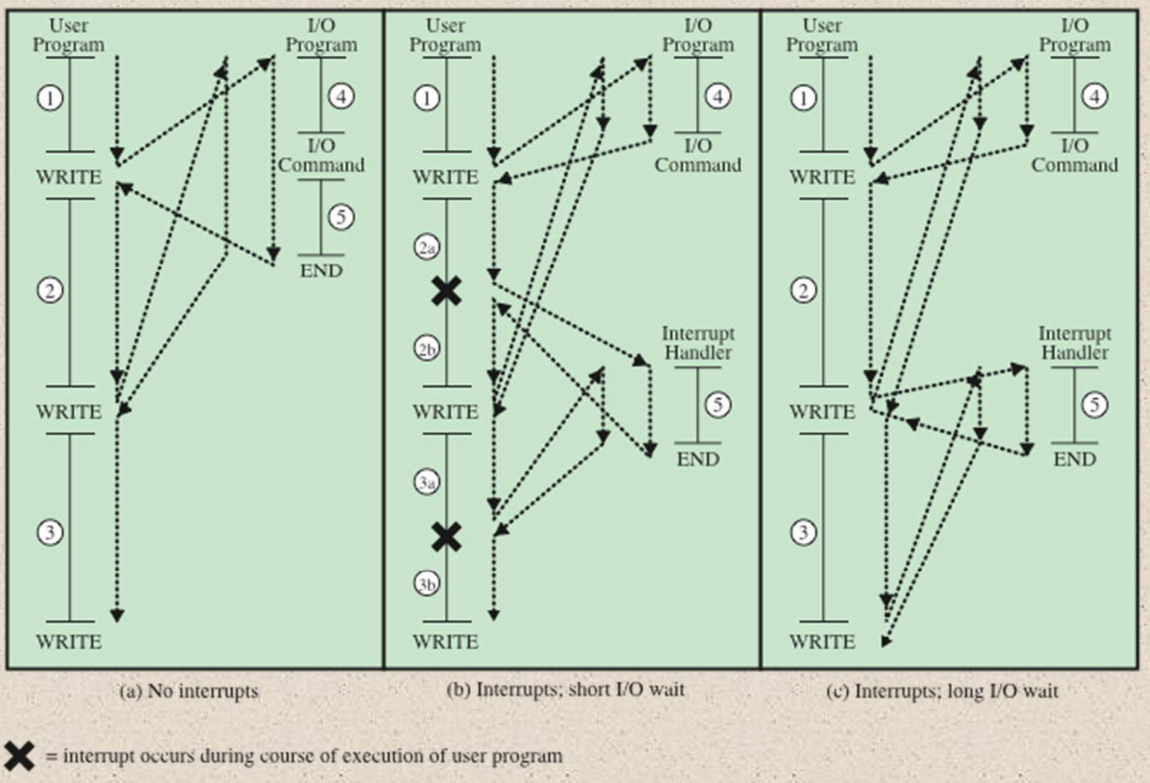


Figure 3.7 Program Flow of Control Without and With Interrupts

Transfer of Control via Interrupts

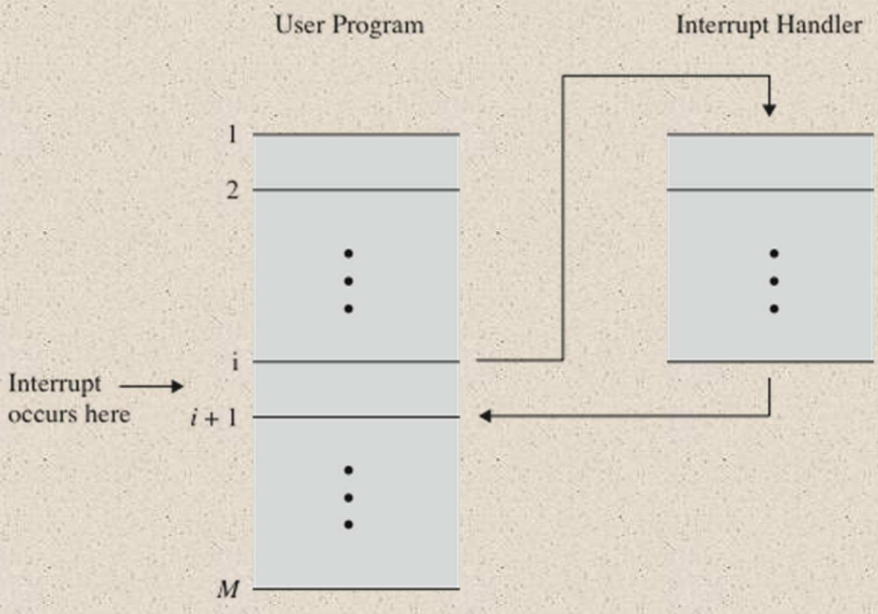


Figure 3.8 Transfer of Control via Interrupts

Instruction Cycle With Interrupts

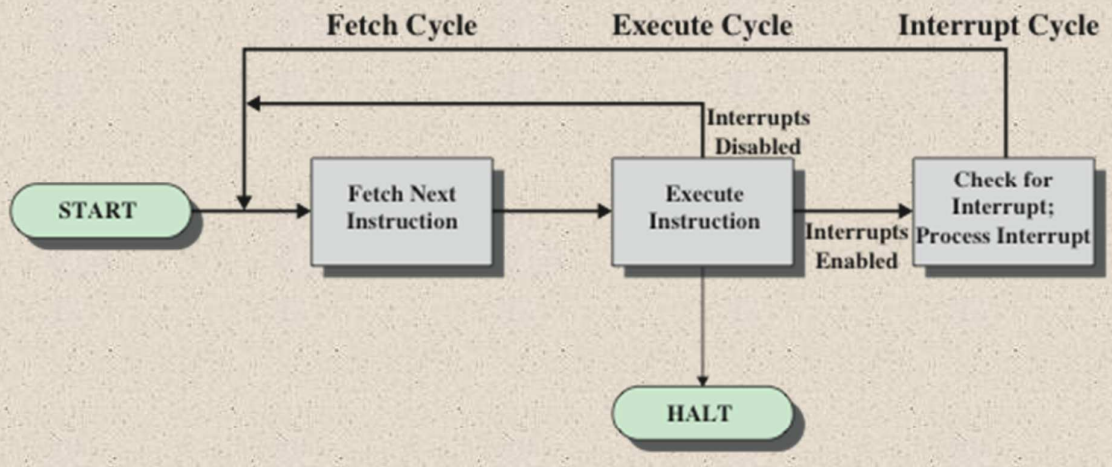


Figure 3.9 Instruction Cycle with Interrupts

Program Timing: Short I/O Wait

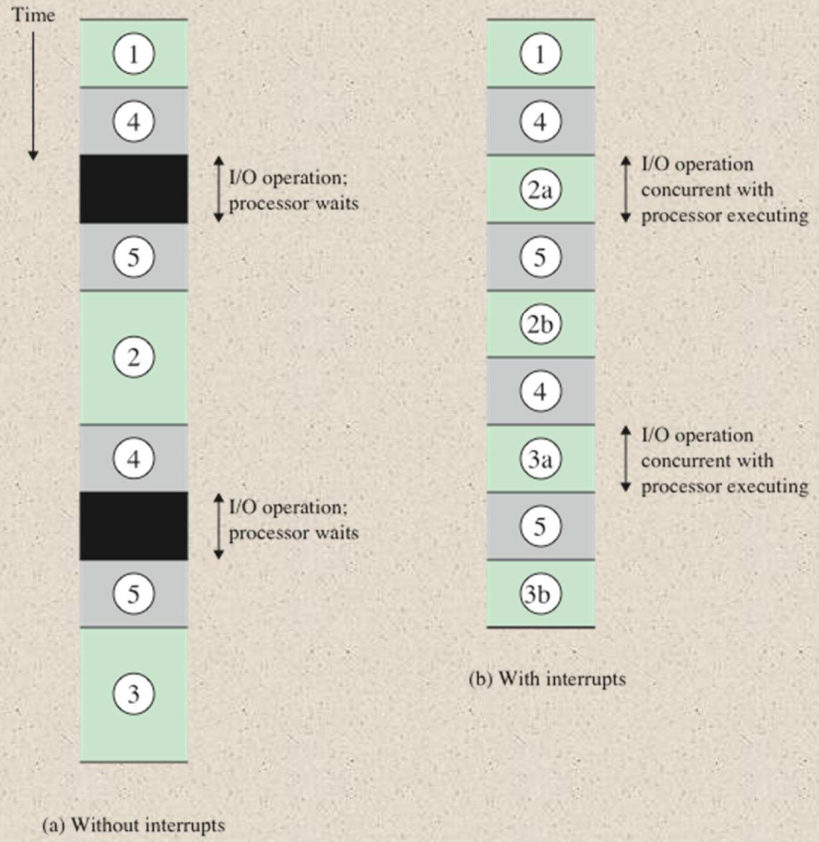


Figure 3.10 Program Timing: Short I/O Wait

Program Timing: Long I/O Wait

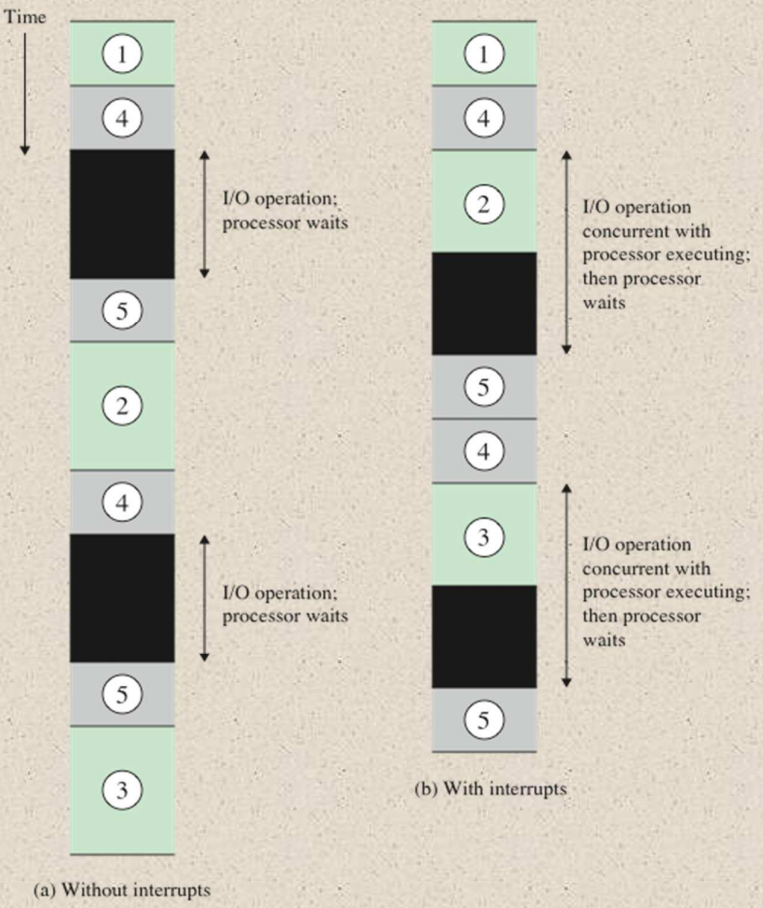


Figure 3.11 Program Timing: Long I/O Wait

Instruction Cycle State Diagram With Interrupts

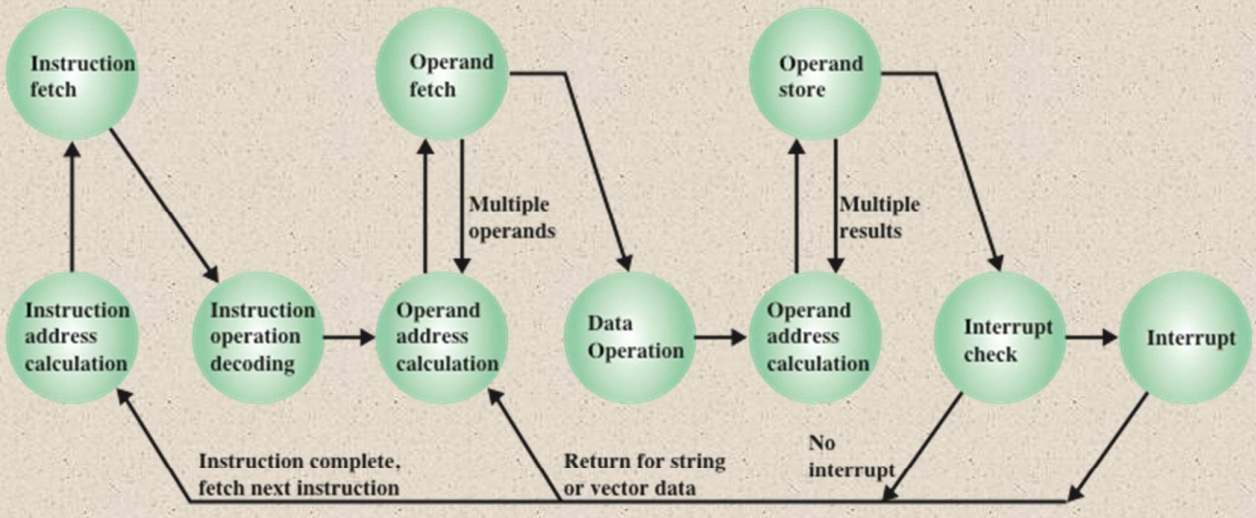
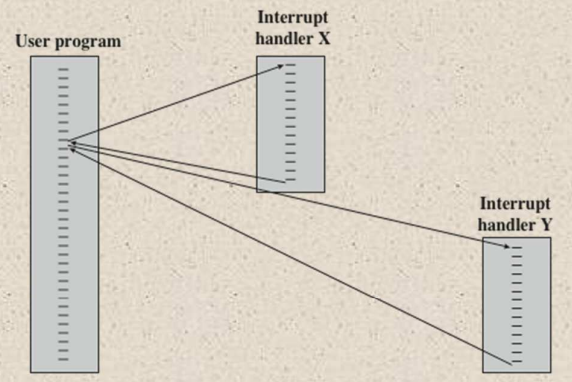


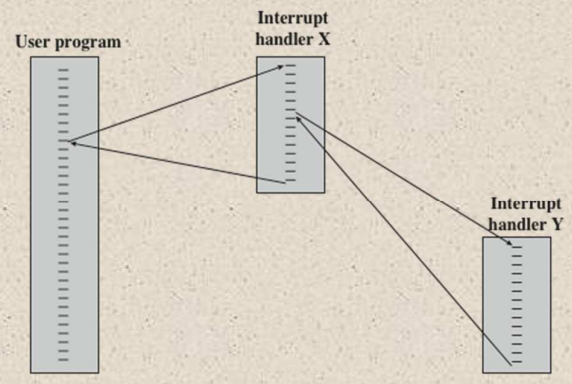
Figure 3.12 Instruction Cycle State Diagram, With Interrupts

Transfer of Control

Multiple Interrupts



(a) Sequential interrupt processing



(b) Nested interrupt processing

Figure 3.13 Transfer of Control with Multiple Interrupts

+ Time Sequence of Multiple Interrupts

Example

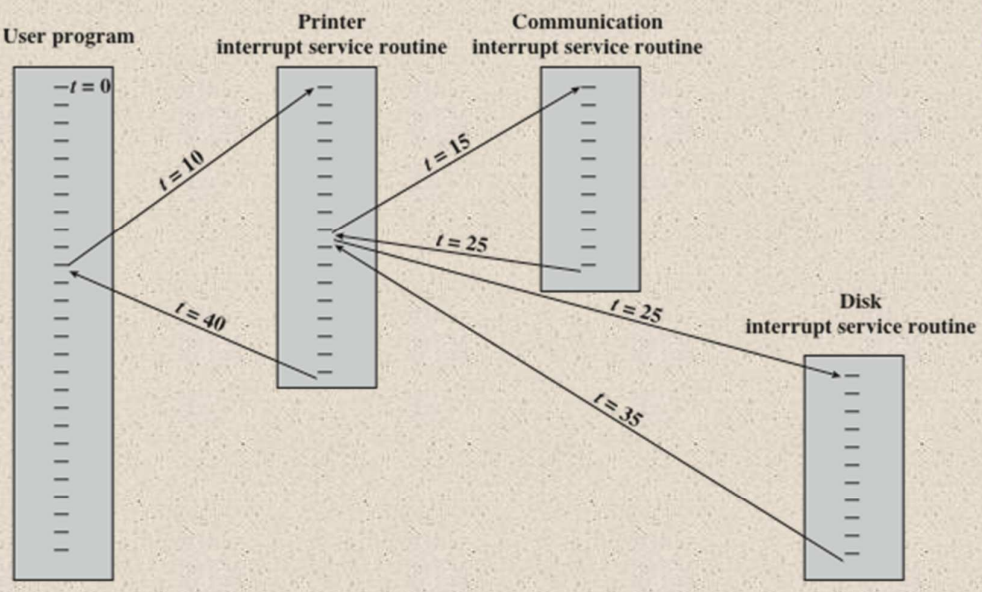


Figure 3.14 Example Time Sequence of Multiple Interrupts

I/O Function

- I/O module can exchange data directly with the processor
- Processor can read data from or write data to an I/O module
  - Processor identifies a specific device that is controlled by a particular I/O module
  - I/O instructions rather than memory referencing instructions
- In some cases it is desirable to allow I/O exchanges to occur directly with memory
  - The processor grants to an I/O module the authority to read from or write to memory so that the I/O memory transfer can occur without tying up the processor
  - The I/O module issues read or write commands to memory relieving the processor of responsibility for the exchange
  - This operation is known as direct memory access (DMA)

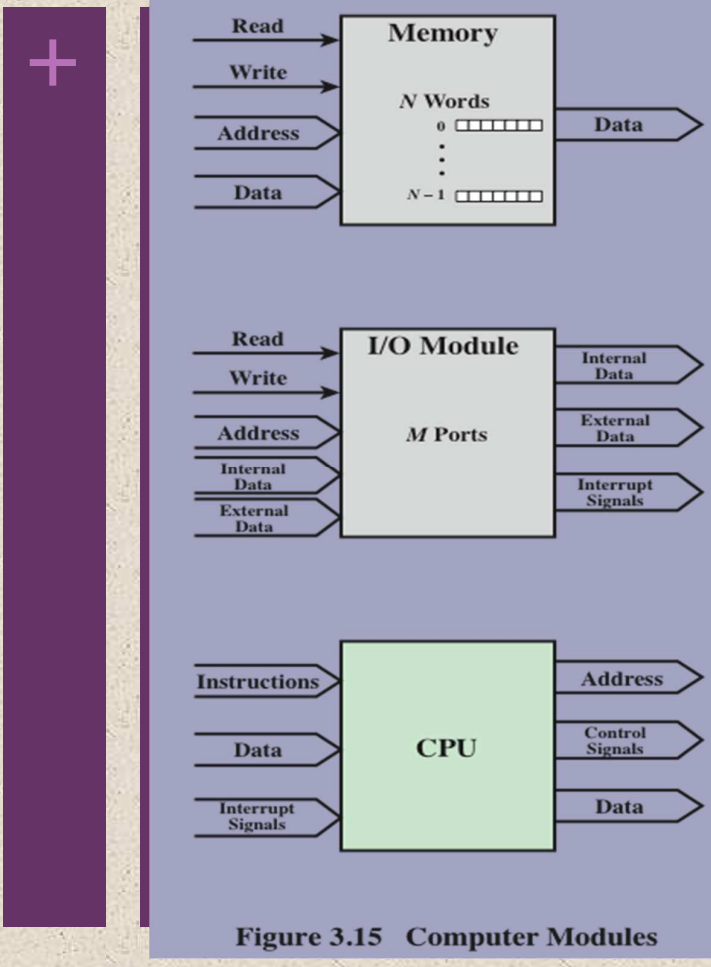


Figure 3.15 Computer Modules

Computer Modules

The interconnection structure must support the following types of transfers:

Memory to processor	Processor to memory	I/O to processor	Processor to I/O	I/O to or from memory
Processor reads an instruction or a unit of data from memory	**Processor writes a unit of data to memory**	**Processor reads data from an I/O device via an I/O module**	**Processor sends data to the I/O device**	**An I/O module is allowed to exchange data directly with memory without going through the processor using direct memory access**

Bus Interconnection

A communication pathway connecting two or more devices

- Key characteristic is that it is a shared transmission medium

Signals transmitted by any one device are available for reception by all other devices attached to the bus

- If two devices transmit during the same time period their signals will overlap and become garbled

Typically consists of multiple communication lines

- Each line is capable of transmitting signals representing binary 1 and binary 0

Computer systems contain a number of different buses that provide pathways between components at various levels of the computer system hierarchy

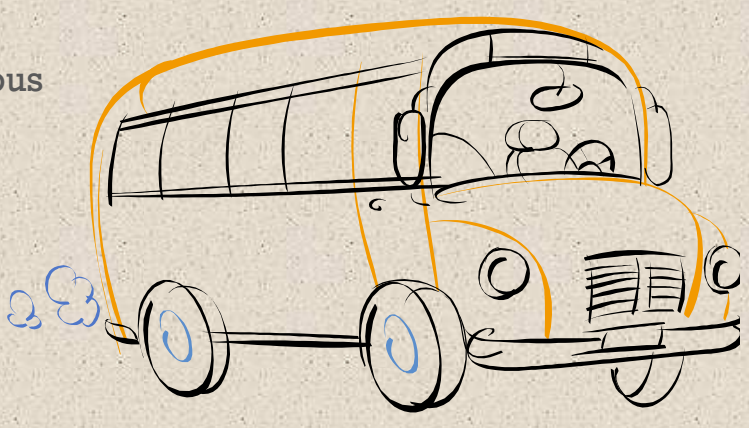
System bus

- A bus that connects major computer components (processor, memory, I/O)

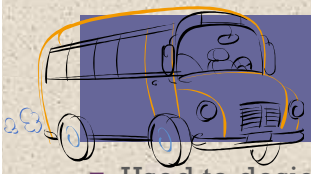
The most common computer interconnection structures are based on the use of one or more system buses

Data Bus

- Data lines that provide a path for moving data among system modules
- May consist of 32, 64, 128, or more separate lines
- The number of lines is referred to as the *width* of the data bus
- The number of lines determines how many bits can be transferred at a time
- The width of the data bus is a key factor in determining overall system performance

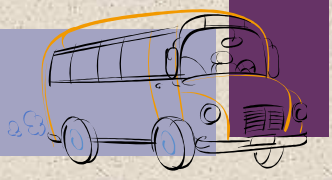


Address Bus



- Used to designate the source or destination of the data on the data bus
  - If the processor wishes to read a word of data from memory it puts the address of the desired word on the address lines
- Width determines the maximum possible memory capacity of the system
- Also used to address I/O ports
  - The higher order bits are used to select a particular module on the bus and the lower order bits select a memory location or I/O port within the module

Control Bus



- Used to control the access and the use of the data and address lines
- Because the data and address lines are shared by all components there must be a means of controlling their use
- Control signals transmit both command and timing information among system modules
- Timing signals indicate the validity of data and address information
- Command signals specify operations to be performed

Bus Interconnection Scheme

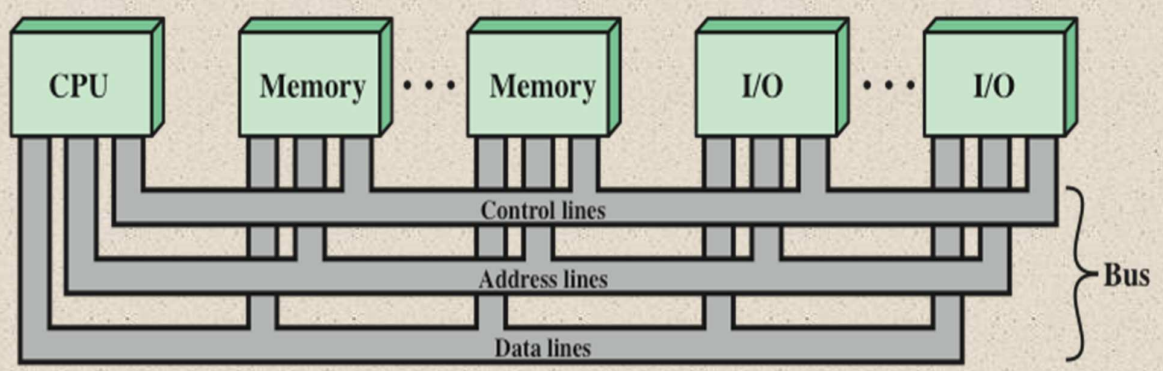


Figure 3.16 Bus Interconnection Scheme

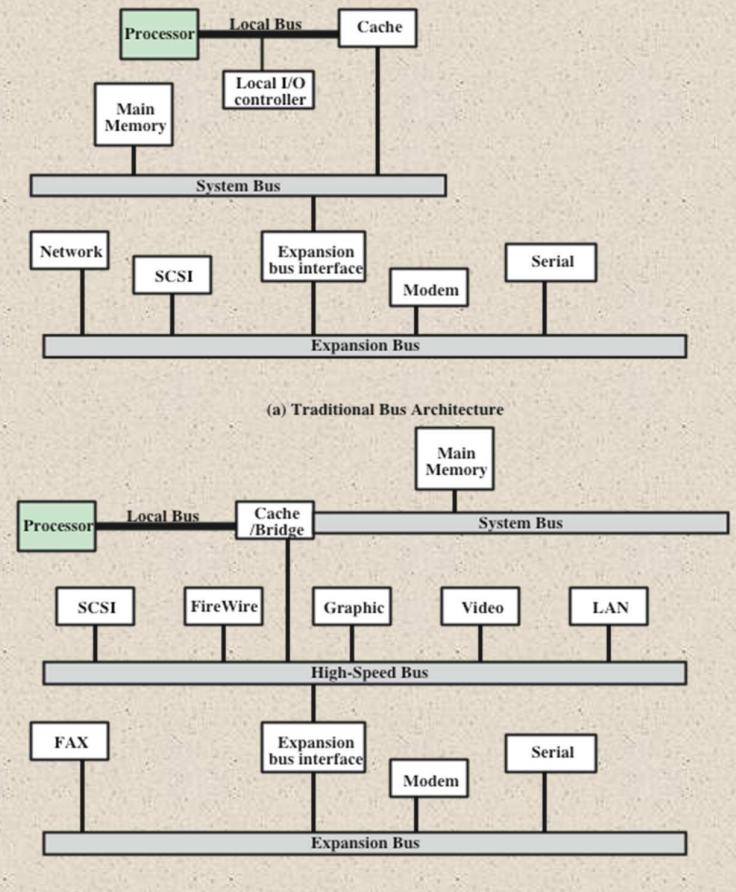


Figure 3.17 Example Bus Configurations

Bus Configurations

Elements of Bus Design

Type	**Bus Width**
Dedicated	Address
Multiplexed	Data
Method of Arbitration	**Data Transfer Type**
Centralized	Read
Distributed	Write
Timing	Read-modify-write
Synchronous	Read-after-write
Asynchronous	Block

Timing of Synchronous Bus Operations

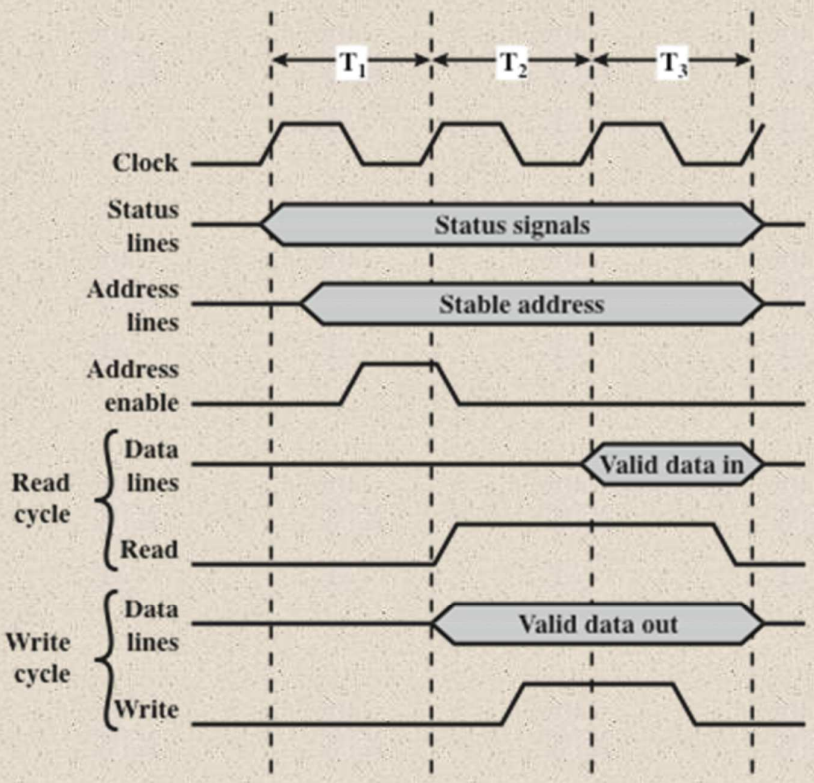
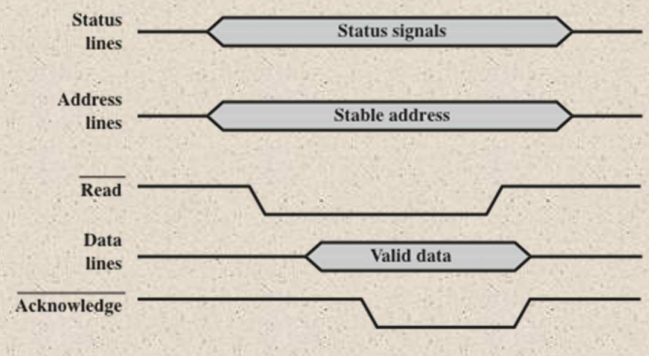
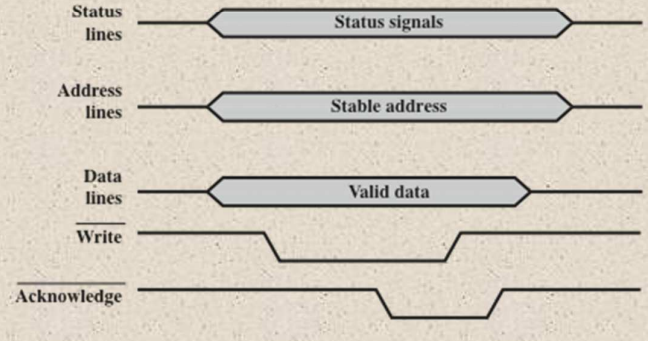


Figure 3.18 Timing of Synchronous Bus Operations

Timing of Asynchronous Bus Operations



(a) System bus read cycle



(b) System bus write cycle

Figure 3.19 Timing of Asynchronous Bus Operations

Summary

Chapter 3

A Top-Level View of Computer Function and Interconnection

- Computer components
- Computer function
  - Instruction fetch and execute
  - Interrupts
  - I/O function
- Interconnection structures
- Bus interconnection
  - Bus structure
  - Multiple bus hierarchies
  - Elements of bus design
- Point-to-point interconnect
  - QPI physical layer
  - QPI link layer
  - QPI routing layer
  - QPI protocol layer
- PCI express
  - PCI physical and logical architecture
  - PCIe physical layer
  - PCIe transaction layer
  - PCIe data link layer