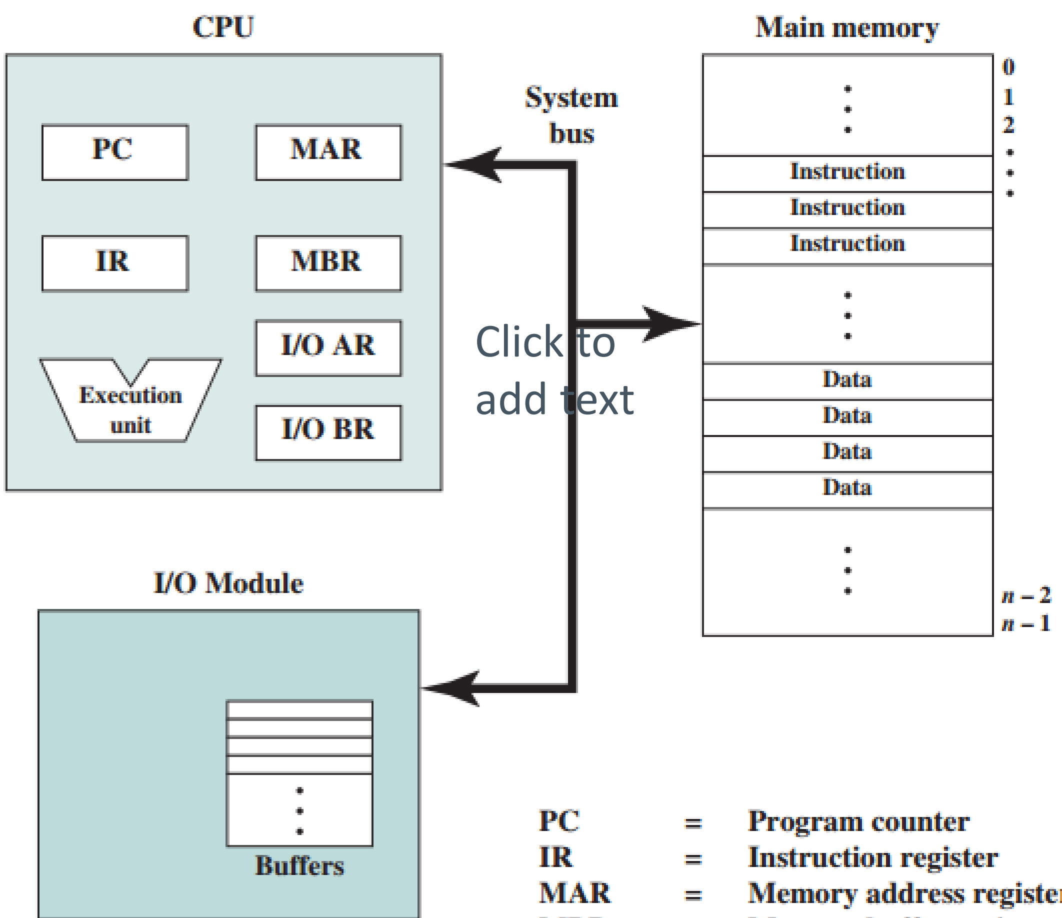
Chương 3

Hệ thống máy tính

Top-level view of computer components

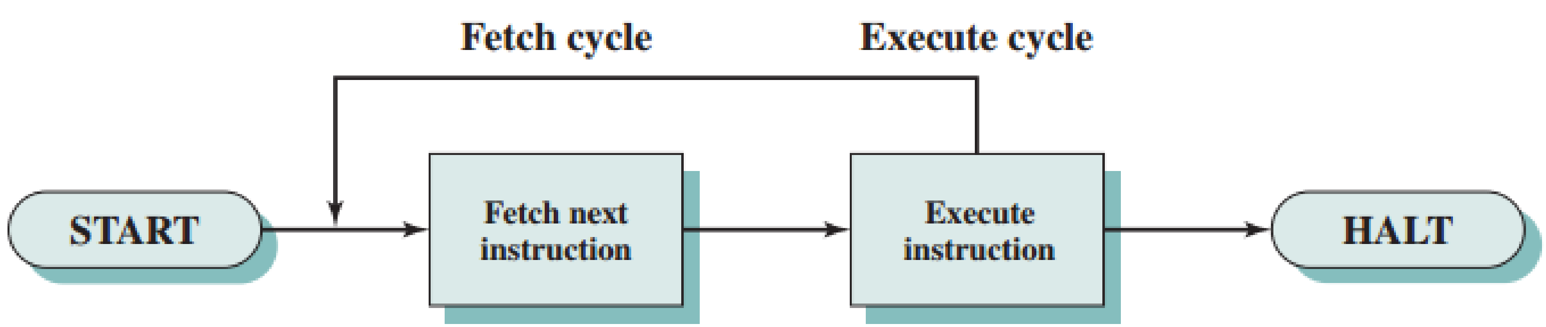


Click to add text

PC	=	Program counter
IR	=	Instruction register
MAR	=	Memory address register
MBR	=	Memory buffer register
I/O AR	=	Input/output address register
I/O BR	=	Input/output buffer register

Computer function

- Basic function performed by a computer is execution of a program (i.e. a set of instructions)
- Instruction processing: read (fetch) instructions from memory and execute each instruction
- Basic instruction cycle:



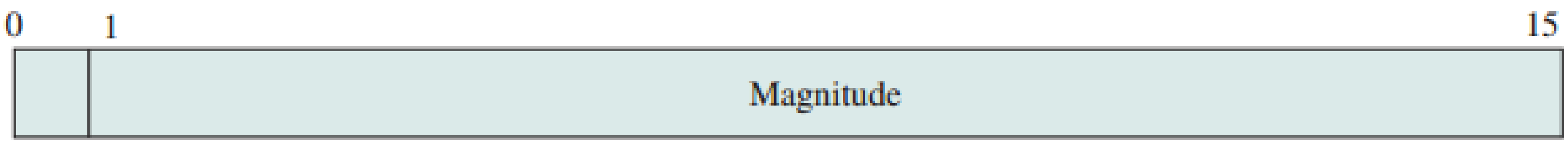
Computer function (cont.)

› 4 categories of instructions:

- Processor-memory: Data may be transferred from processor to memory or from memory to processor.
- Processor-I/O: Data may be transferred to or from a peripheral device by transferring between the processor and an I/O module.
- Data processing: The processor may perform some arithmetic or logic operation on data.
- Control: An instruction may specify that the sequence of execution be altered.

Example of program execution

(a) Instruction format



(b) Integer format

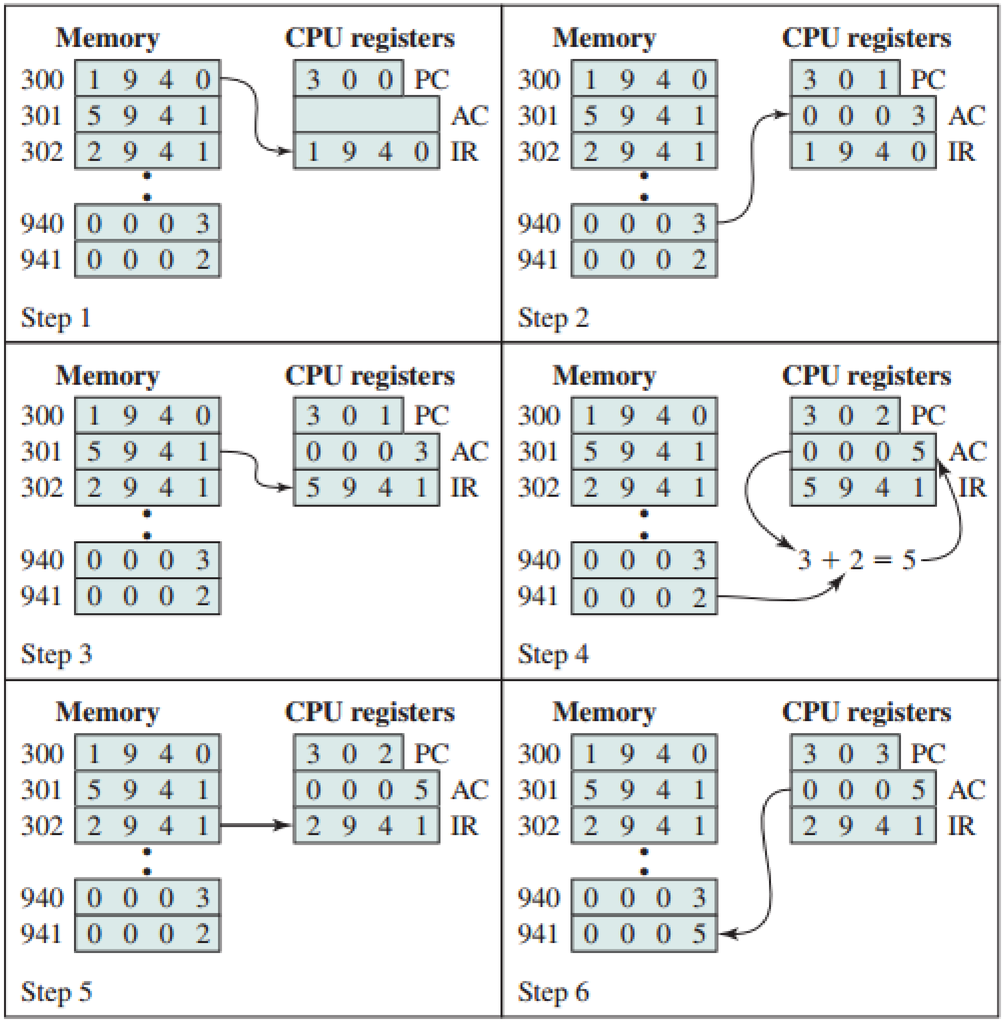
Program counter (PC) = Address of instruction
Instruction register (IR) = Instruction being executed
Accumulator (AC) = Temporary storage

(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

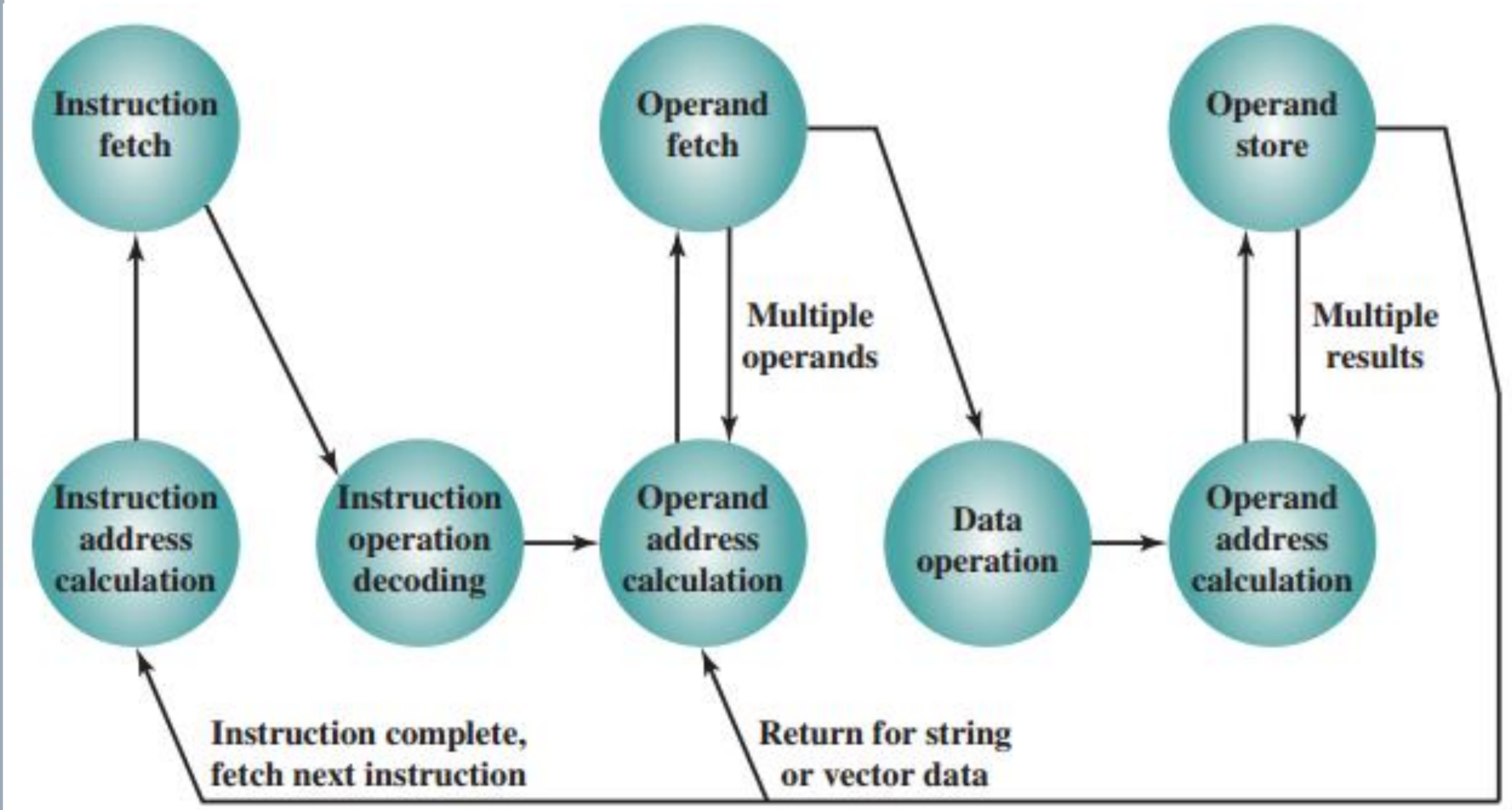
Example of program execution (cont.)



Example of program execution (cont.)

- Execution cycle for a particular instruction may involve more than one reference to memory
- Steps of processing instruction ADD B,A (i.e. stores the sum of the contents of memory locations B and A
  into memory location A):
  - Fetch the ADD instruction.
  - Read the contents of memory location A into the processor.
  - Read the contents of memory location B into the processor.
  - Add the two values.
  - Write the result from the processor to memory location A.

Instruction cycle state diagram



Instruction cycle state diagram (cont.)

- Instruction address calculation (iac): Determine the address of the next instruction to be executed. Usually, this involves adding a fixed number to the address of the previous instruction.
- Instruction fetch (if): Read instruction from its memory location into the processor.
- Instruction operation decoding (iod): Analyze instruction to determine type of operation to be performed and operand(s) to be used.
- Operand address calculation (oac): If the operation involves reference to and operand in memory or available via I/O, then determine the address of the operand.
- Operand fetch (of): Fetch the operand from memory or read it in from I/O.
- Data operation (do): Perform the operation indicated in the instruction.
- Operand store (os): Write the result into memory or out to I/O.

Interrupts

- A mechanism by which other modules (I/O, memory) may interrupt the normal processing of the processor
- 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.

Interrupt action

- Interrupts can occur without warning.
- When an interrupt occurs, the program counter and status flags are saved in a special location.
- New program counter and status flags are loaded. The location may be determined by the type of interrupt.
- A interrupt is similar to a function call, the return address is pushed on the stack and execution jumps to another location.

Program Flow Control

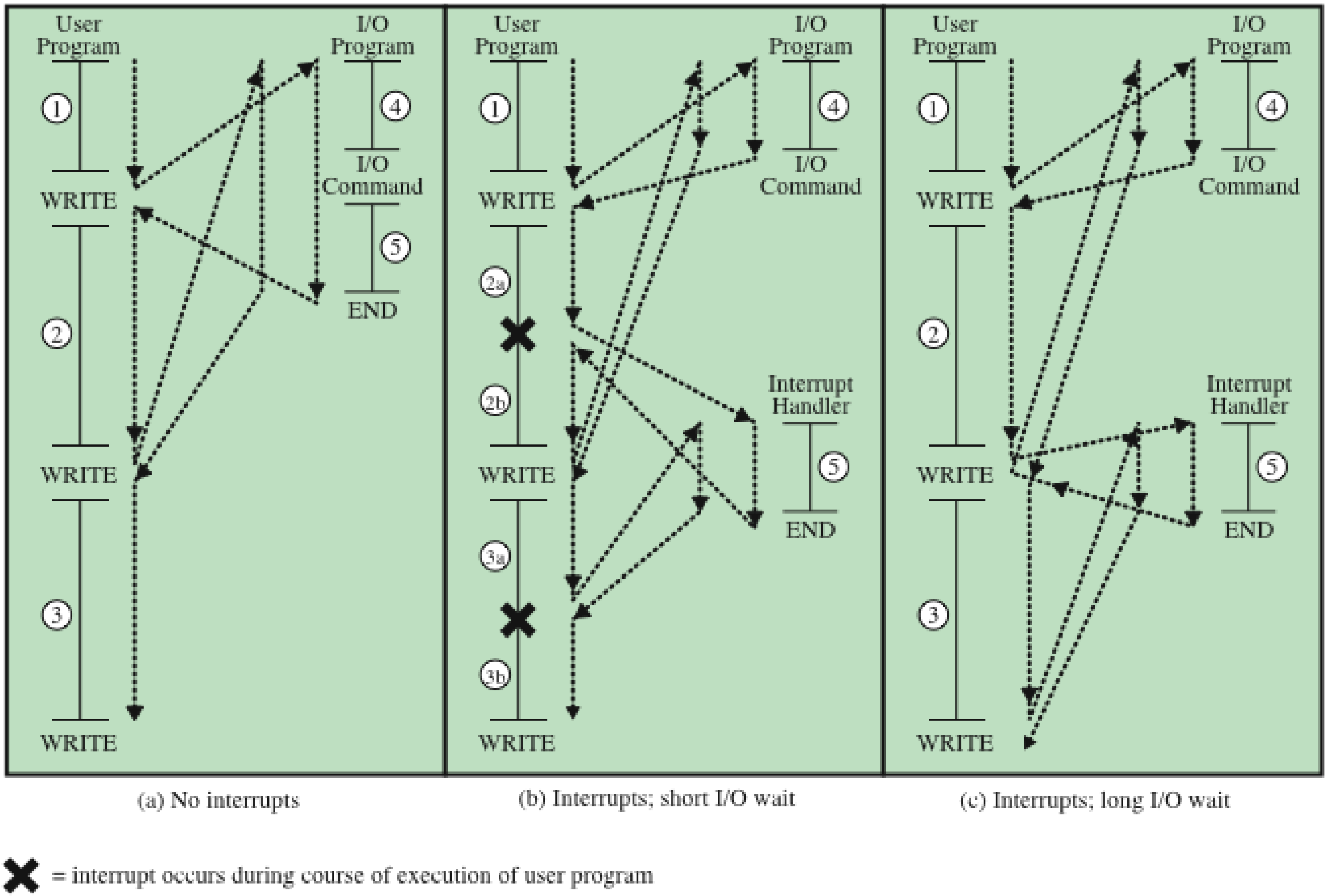
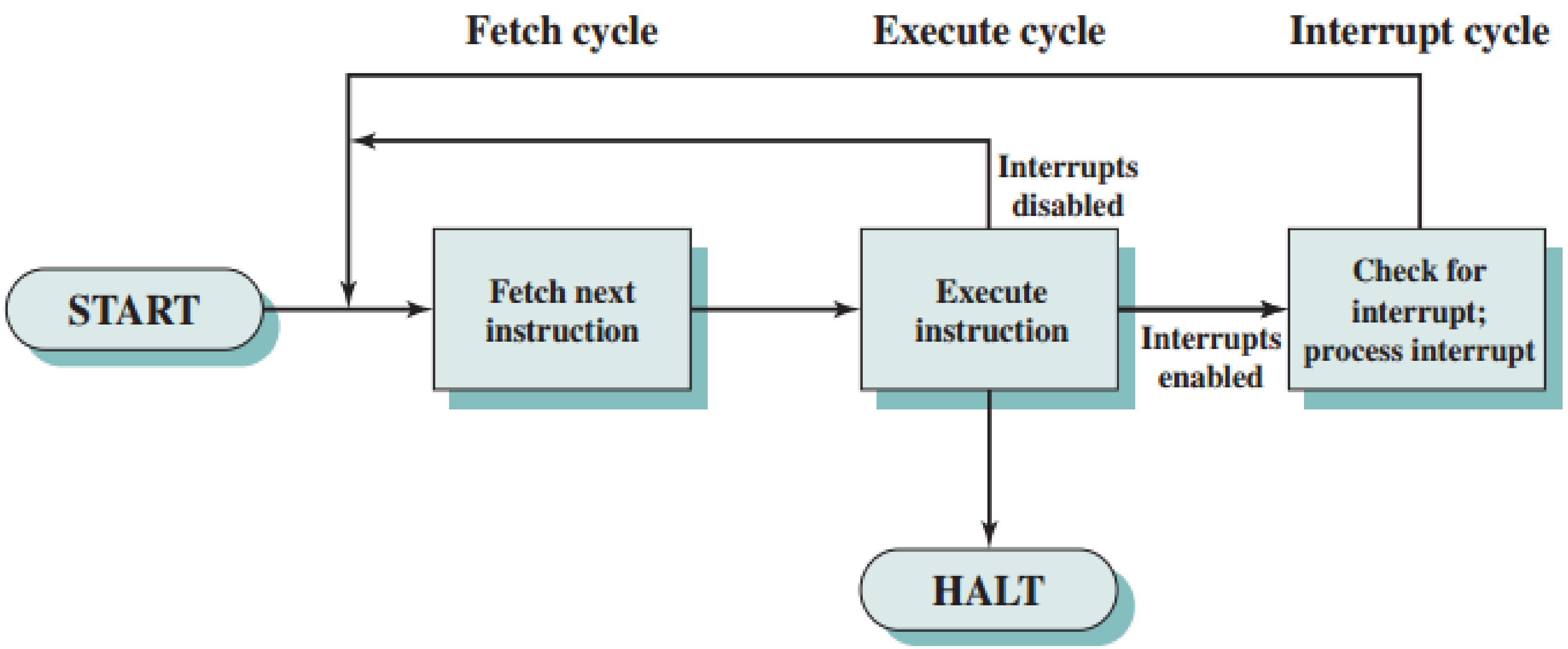
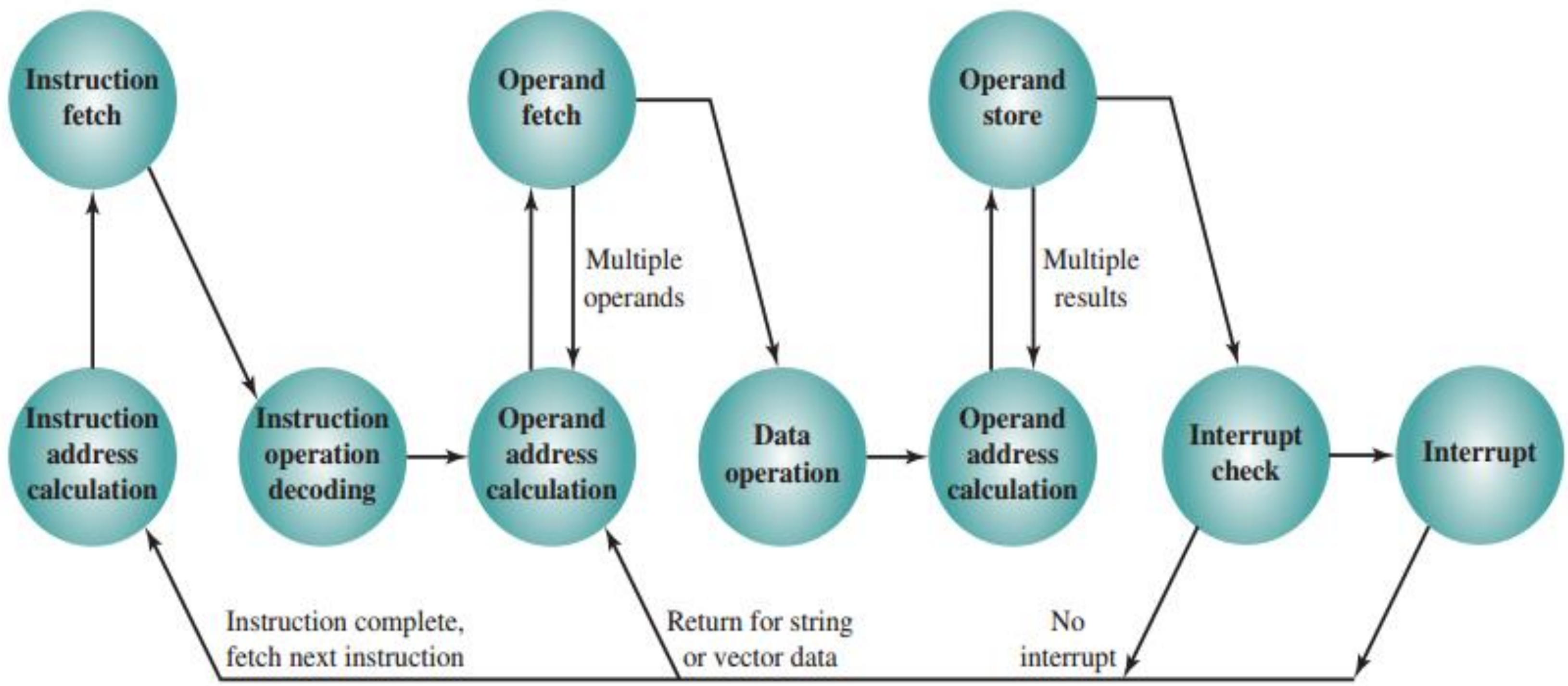


Figure 3.7 Program Flow of Control Without and With Interrupts

Instruction cycle with interrupts



Instruction cycle state diagram with interrupts



Multiple interrupts

› Two approaches for handling multiple interrupts:
  - Disabled interrupt: ignore other interrupts request signals while an interrupt is being processed
  - Define priorities for interrupts and allow an interrupt with higher priority to cause a lower-priority handler to be itself interrupted

Bus interconnection

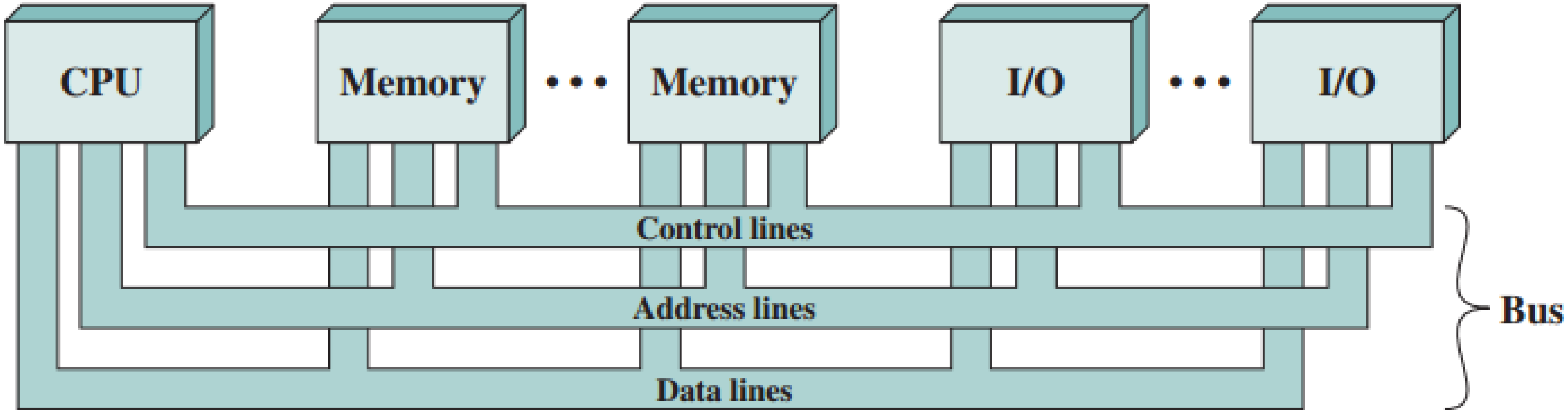
- Communication pathway connecting the various components of a computer
- Types of transfers:
  - Memory to processor: The processor reads an instruction or a unit of data from memory.
  - Processor to memory: The processor writes a unit of data to memory.
  - I/O to processor: The processor reads data from an I/O device via an I/O module.
  - Processor to I/O: The processor sends data to the I/O device.
  - I/O to or from memory: For these two cases, an I/O module is allowed to exchange data directly with memory, without going through the processor, using direct memory access (DMA).

Bus structure

› Data lines (data bus)

› Address lines (address bus)

› Control lines



Bus structure (cont.)

- Typical control lines:
  - Memory write: causes data on the bus to be written into the addressed location
  - Memory read: causes data from the addressed location to be placed on the bus
  - I/O write: causes data on the bus to be output to the addressed I/O port
  - I/O read: causes data from the addressed I/O port to be placed on the bus
  - Transfer ACK: indicates that data have been accepted from or placed on the bus
  - Bus request: indicates that a module needs to gain control of the bus
  - Bus grant: indicates that a requesting module has been granted control of the bus
  - Interrupt request: indicates that an interrupt is pending
  - Interrupt ACK: acknowledges that the pending interrupt has been recognized
  - Clock: is used to synchronize operations
  - Reset: initializes all modules

Elemer

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

Bus arbitration methods

- The bus arbitration protocol determines which device gets to use the bus at any given time.
- Bus arbitration can be centralized or distributed
- In a centralized scheme, a single hardware device, referred to as a bus controller or arbiter, is responsible for allocating time on the bus
- In a distributed scheme, there is no central controller; each module contains access control logic and the modules act together to share the bus.

Bus types

- Dedicated bus:
  - There are separate wires for data and addresses
  - A store operation can put both the address and the data on the bus at the same time
  - High throughput, less contention
  - Increased size and cost
- Multiplexed bus:
  - The same lines are used at different times to hold either data or addresses
  - Multiplexed buses require fewer lines.
  - More complex circuitry is needed
  - Potential reduction in performance

Bus timing