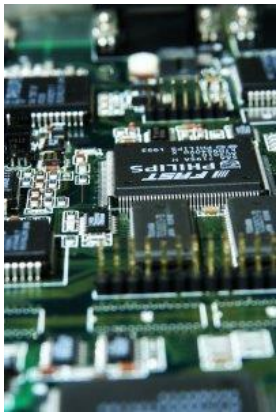
CSE 321a

Computer Organization (1)

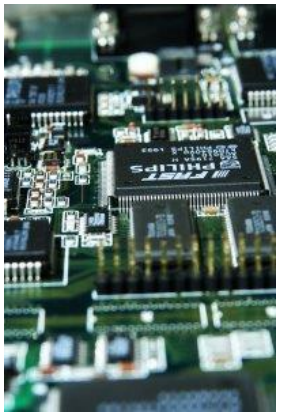
تنظيم الحاسبات (1)



3rd year, Computer Engineering

Fall 2016

Lecture #2



Dr. Hazem Ibrahim Shehata

Dept. of Computer & Systems Engineering

Credits to Dr. Ahmed Abdul-Monem Ahmed for the slides

Administrivia

- Website:
  - —http://hshehata.github.io/courses/zu/cse321a/
- Lecture Day/Time:
  - —Tuesday 10:15am - 12:45am
- Office Hours:
  - —TBA!
- Tutorials:
  - —Starting next week on Sunday at 2:55pm (just once!)
  - —Regular day/time: Thursday 10:00am – 11:30am
- Assignment #1:
  - —To be early next week.
  - —Work in groups of two.

Ch. 3: A Top-Level View of Computer Function and Interconnection

Overview

- Computer Structure (Components)
- Computer Function
  - —Instruction Fetch and Execute
  - —Interrupts
- Interconnection Structures
- Bus Interconnection
  - —Bus Structure
  - —Multiple-Bus Hierarchies
  - —Elements of Bus Design

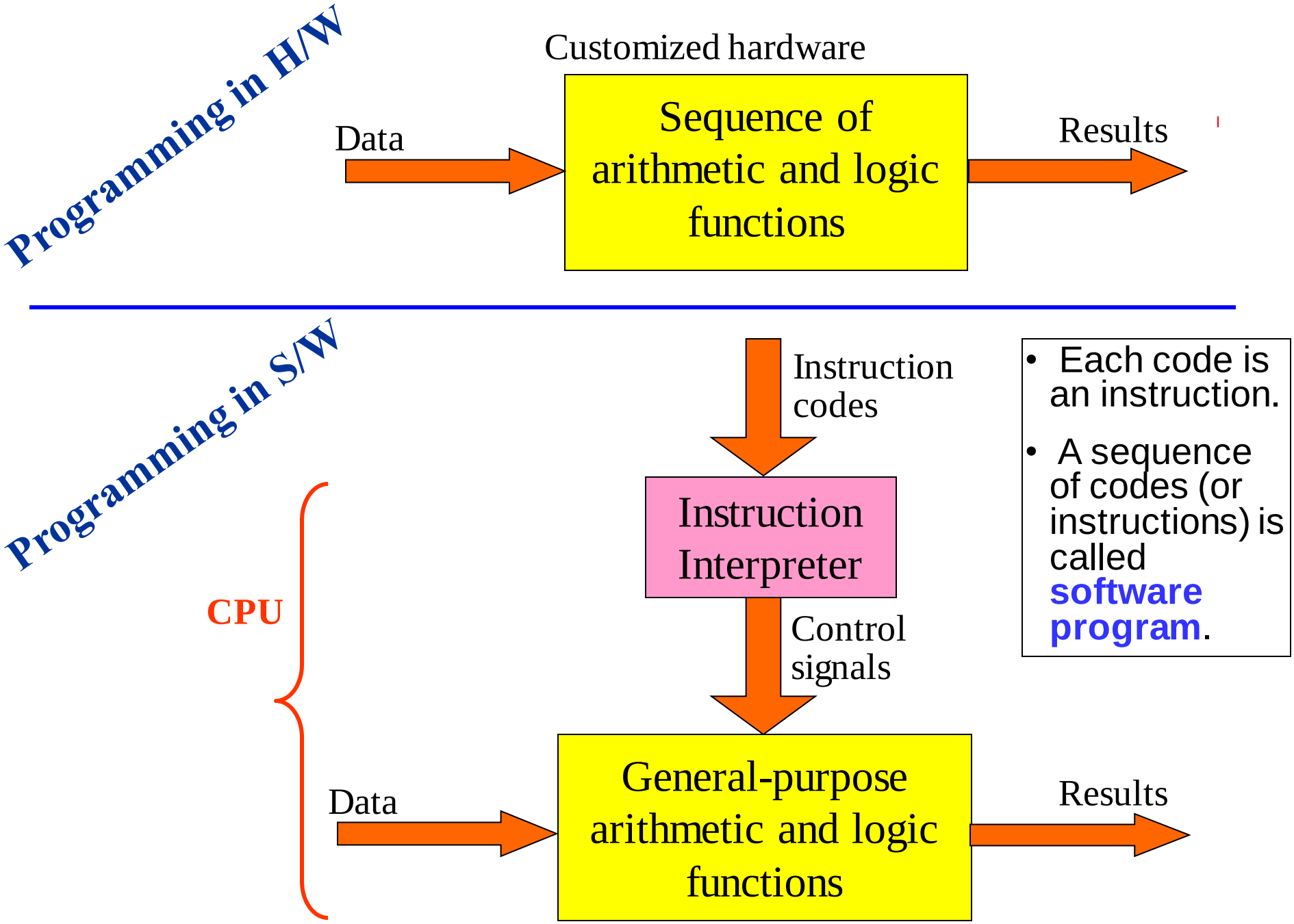
Von Neumann (Princeton) Architecture

- Almost all contemporary computer designs are based on three basic concepts introduced by John von Neumann in 1945:

1. Data and instructions are stored in a single read-write memory.
2. The contents of this memory are addressable by location, without regard to the type of data contained there.
3. Execution occurs in a sequential fashion (unless modified) from one instruction to the next.

Program Concept

- Using a fixed set of logic components, there are two ways to build a system that performs a specific computation on some input data:
  1. Special-purpose (customized) hardware:
     - Logic components are connected and configured specifically to perform a particular computation.
     - Type of computation is specified by way in which components are connected/configured → hardwired programming.
     - To change computation, rewire components → inflexible!
  2. General-purpose hardware:
     - Logic components are connected to form a general-purpose hardware that gets configured using set of input control signals.
     - Type of computation (reflected by control signals values) is specified by sequence of input codes → software programming.
     - To change computation, change code → flexible!

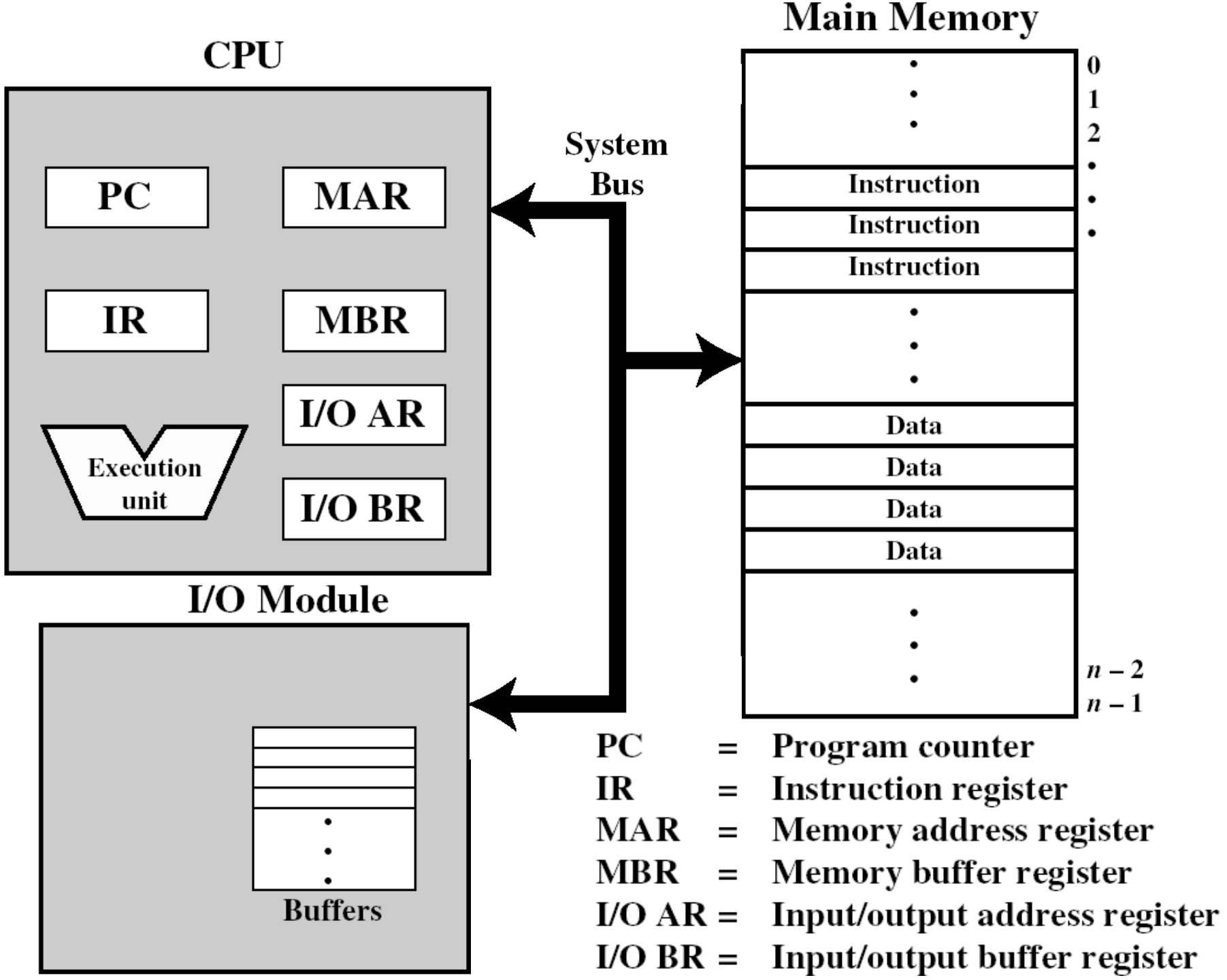

Programming in H/W
Customized hardware
Data
Sequence of arithmetic and logic functions
Results
Programming in S/W
Instruction codes
Instruction Interpreter
Control signals
CPU
Data
General-purpose arithmetic and logic functions
Results
• Each code is an instruction.
• A sequence of codes (or instructions) is called software program.

What is a program?

- A sequence of steps.
- For each step, an arithmetic or logical operation could be done.
- For each operation, a different set of control signals are generated.
  - —Each operation is assigned a unique code.
    - e.g., ADD ➔ 9D, MOVE ➔ E3, … etc.
  - —A hardware segment accepts the code and issues the control signals.
- That's it. We have a computer!

Computer Components

- Each computer component takes some part in implementing the "s/w programming" approach:
  - —Control unit (CU) acts as instruction interpreter.
  - —Arithmetic and logic unit (ALU) implements general-purpose arithmetic and logic functions.
  - —Both CU and ALU constitute most of central processing unit (CPU).
  - —Data/code get into system, and results get out through input/output (I/O) units.
  - —Data, code, and results are stored temporarily while being processed in main memory (MM) unit.

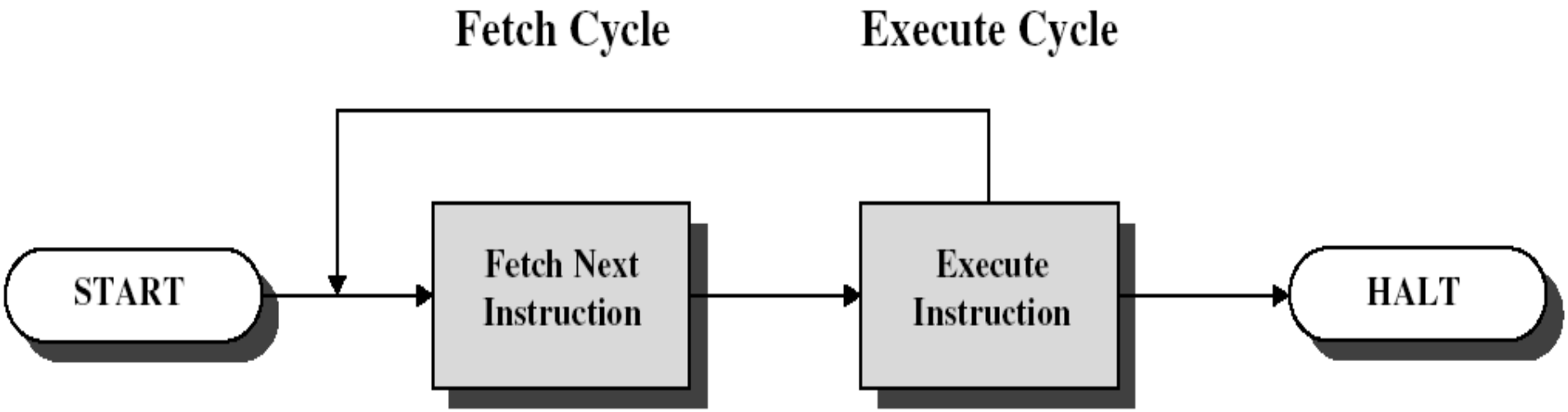

Main Memory
CPU
PC
MAR
IR
MBR
I/O AR
I/O BR
Execution unit
System Bus
0
1
2
Instruction
Instruction
Instruction
Data
Data
Data
Data
n – 2
n – 1
I/O Module
Buffers
PC = Program counter
IR = Instruction register
MAR = Memory address register
MBR = Memory buffer register
I/O AR = Input/output address register
I/O BR = Input/output buffer register

Overview

- Computer Structure (Components)
- Computer Function
  - —Instruction Fetch and Execute
  - —Interrupts
- Interconnection Structures
- Bus Interconnection
  - —Bus Structure
  - —Multiple-Bus Hierarchies
  - —Elements of Bus Design

Instruction Cycle

- Two steps:
  - —Fetch: CPU reads instructions from memory, one at a time.
  - —Execute: CPU executes instructions.



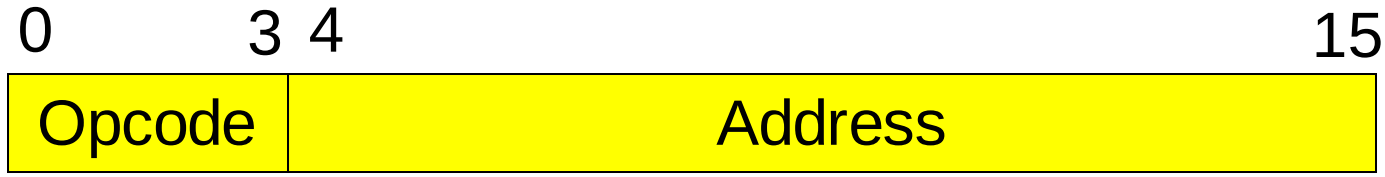
Fetch Cycle

- Program Counter (PC) holds address of next instruction to fetch.
- Processor fetches instruction from memory location pointed to by PC.
- Increment PC.
  - Unless told otherwise
- Instruction loaded into Instruction Register (IR).
- Processor interprets instruction and performs required actions in execute cycle.

Execute Cycle

- Processor-memory
  - Data transfer between CPU and main memory
- Processor-I/O
  - Data transfer between CPU and I/O module
- Data processing
  - Some arithmetic or logical operation on data
- Control
  - Alteration of sequence of operations
  - e.g., jump
- Combination of the above

Hypothetical Machine



Instruction Format

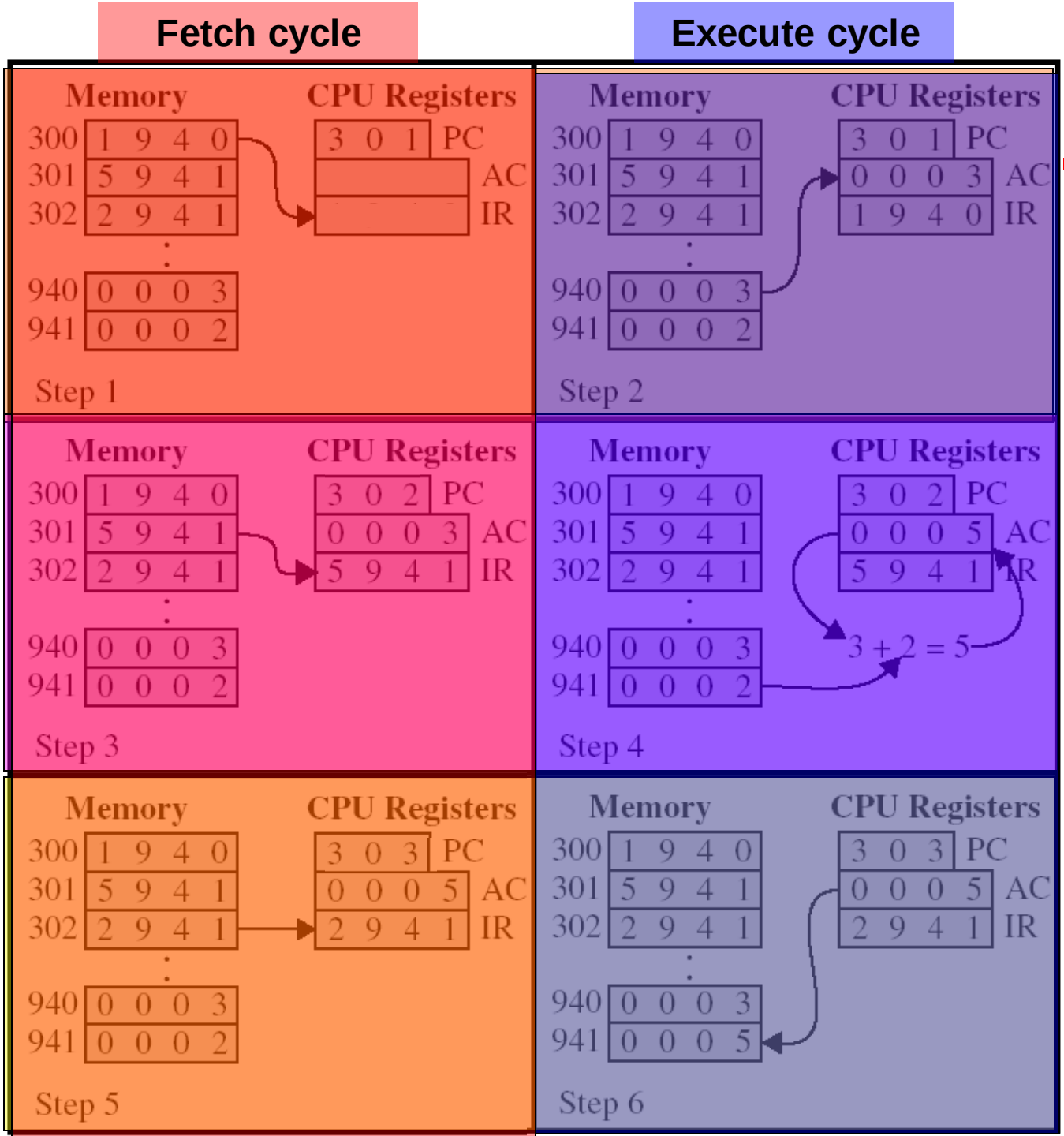
Integer Format

Program Counter (PC)
Instruction Register (IR)
Accumulator (AC)

CPU Registers

0001 = Load AC from memory.
0010 = Store AC to memory.
0101 = Add to AC from memory.

Partial List of Opcodes

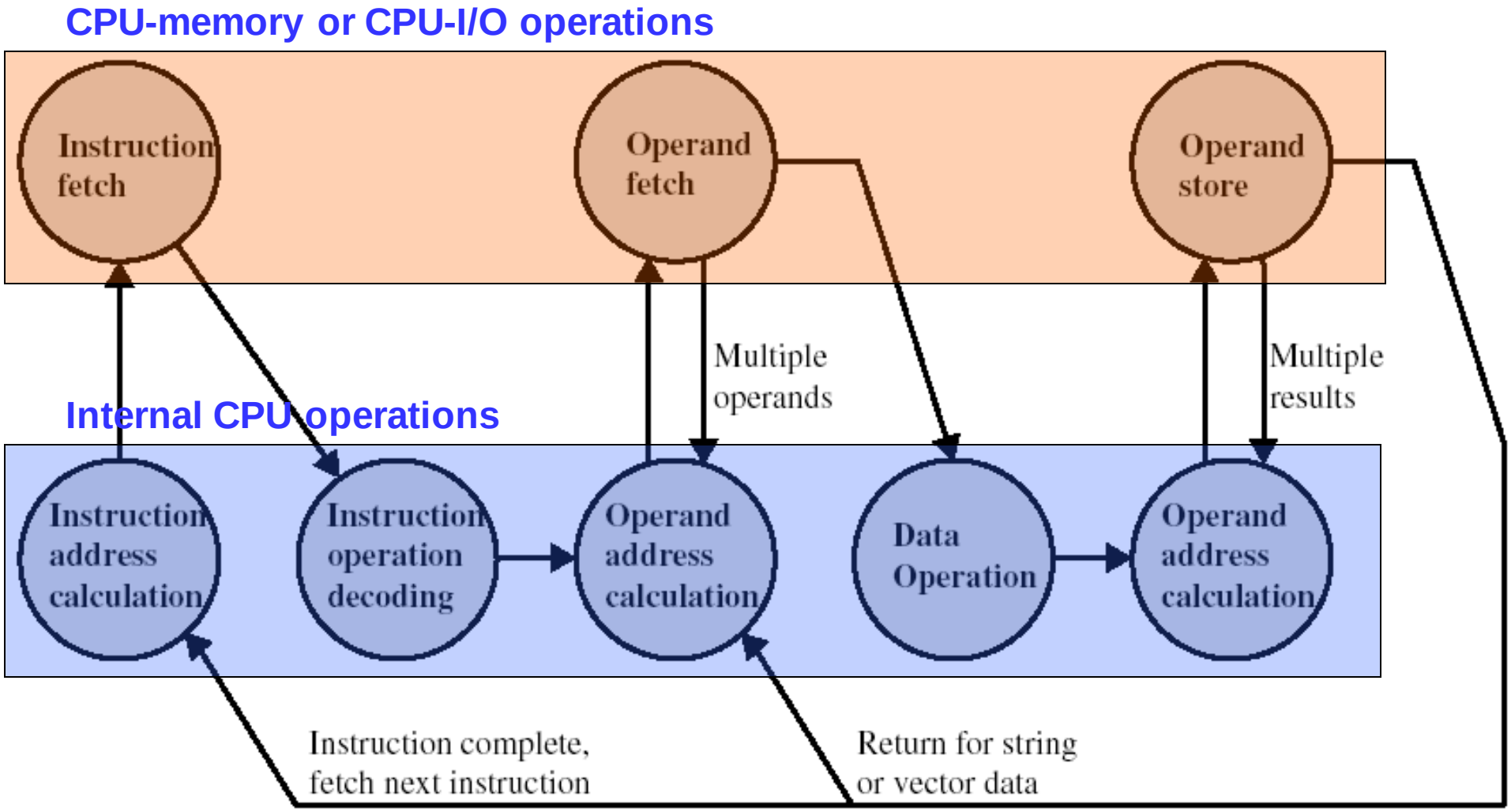


First instruction

Second instruction

Third instruction

Instruction Cycle: State Diagram



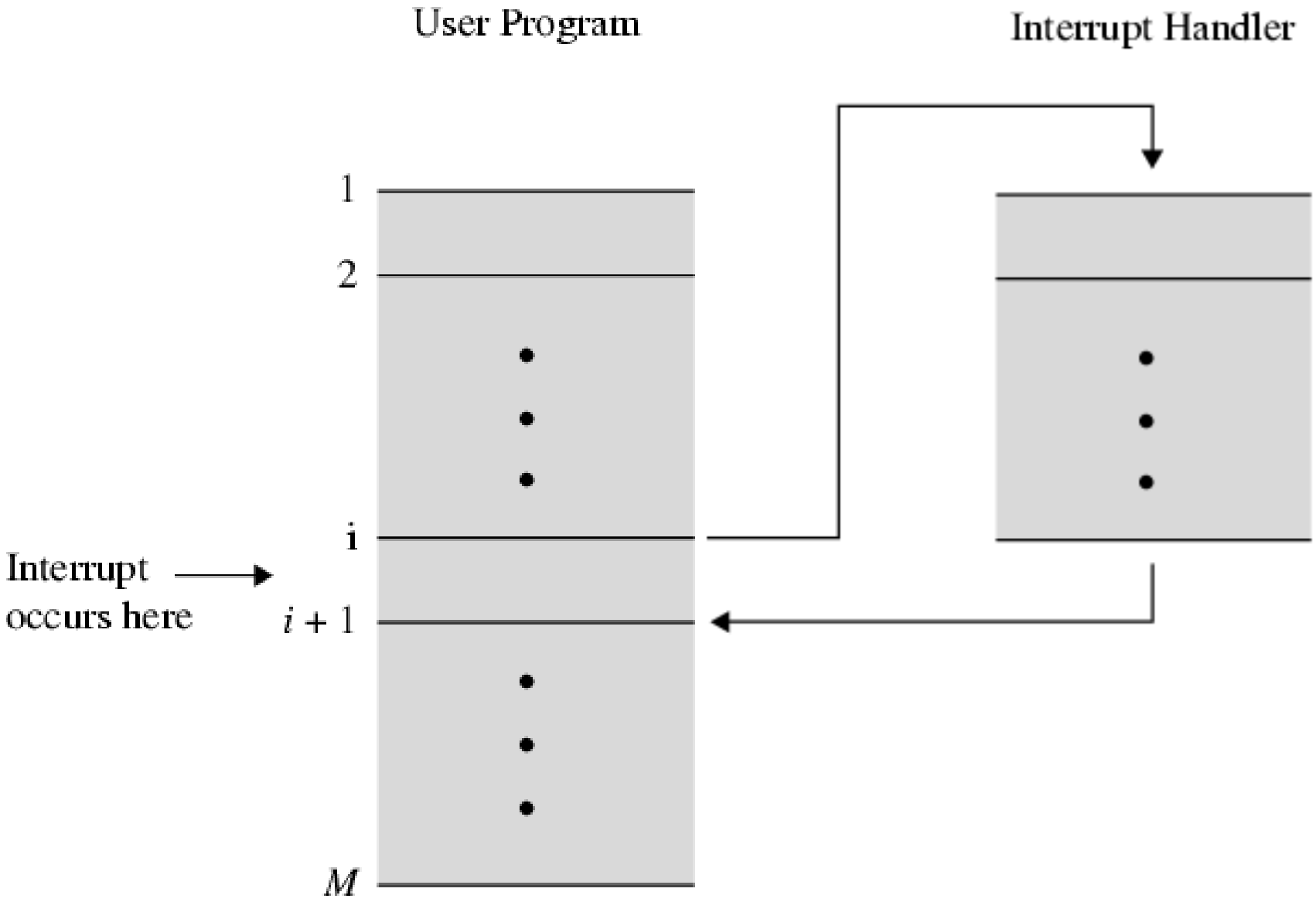
Overview

- Computer Structure (Components)
- Computer Function
  - —Instruction Fetch and Execute
  - —Interrupts
- Interconnection Structures
- Bus Interconnection
  - —Bus Structure
  - —Multiple-Bus Hierarchies
  - —Elements of Bus Design

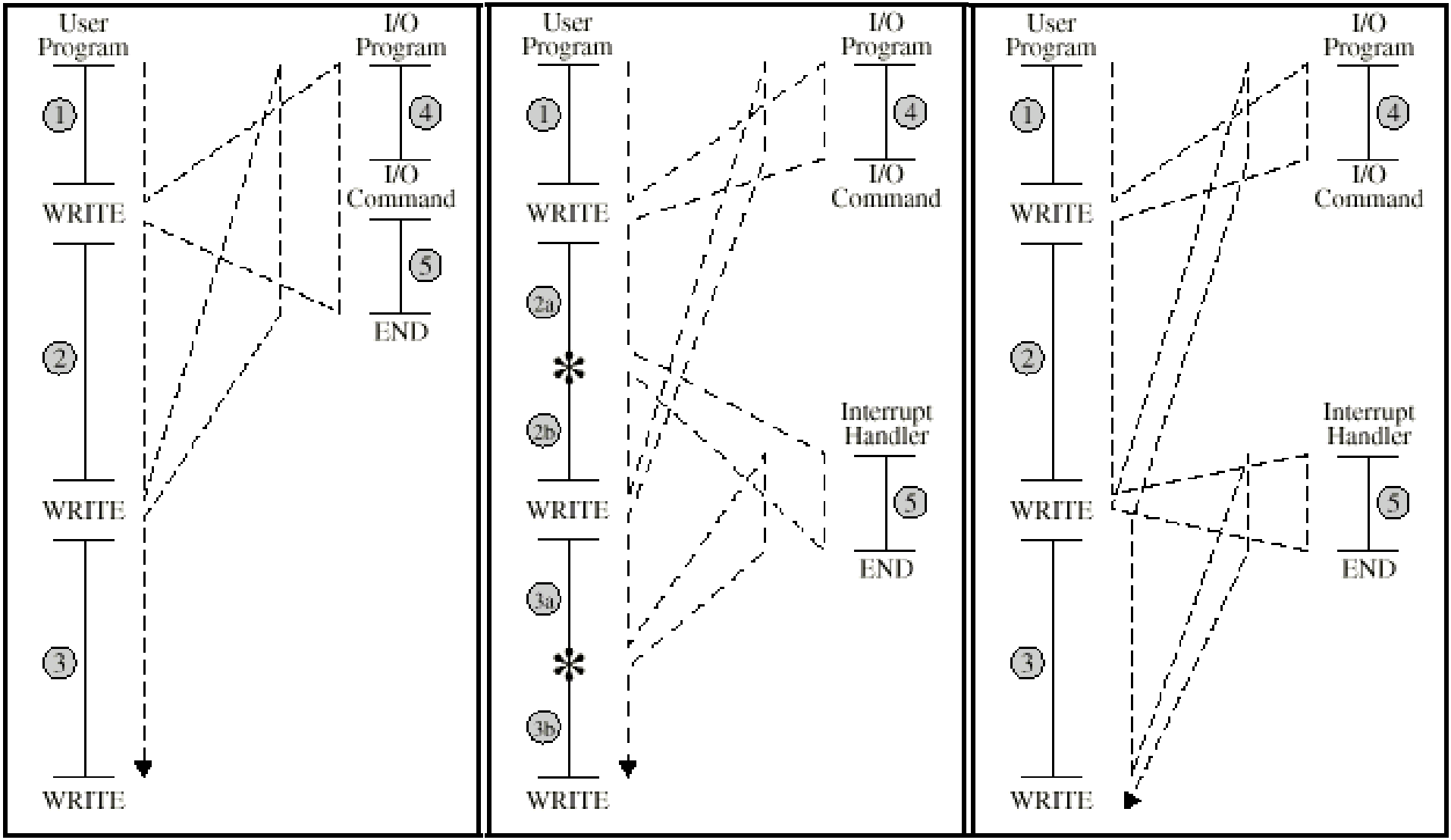
Interrupts

- Interrupts are provided to improve processing efficiency.
- Mechanism by which other modules (e.g. I/O) may interrupt normal sequence of processing
- Most common classes of interrupts:
  - Program
    - e.g. overflow, division by zero
  - Timer
    - Generated by internal processor timer
    - Used in pre-emptive multi-tasking (i.e., CPU time sharing).
  - I/O
    - From I/O controller
  - Hardware failure
    - e.g. memory parity error

Transfer of Control via Interrupts



Program Flow Control

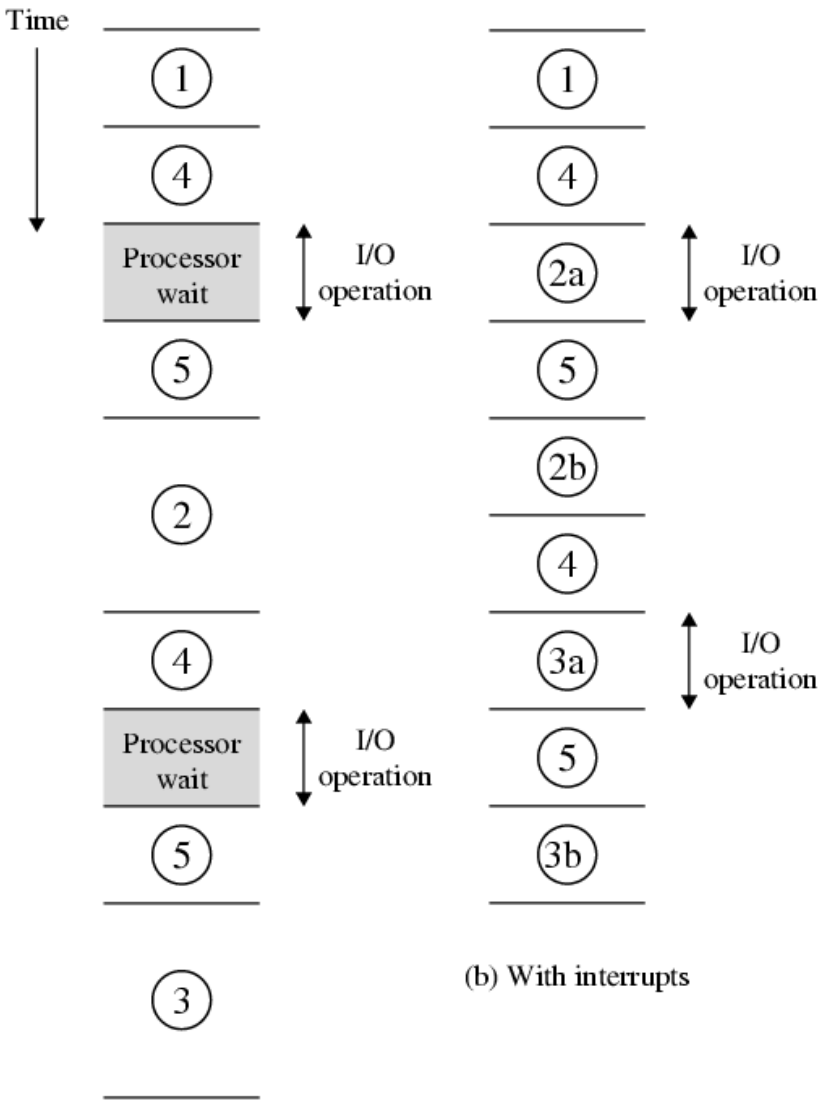


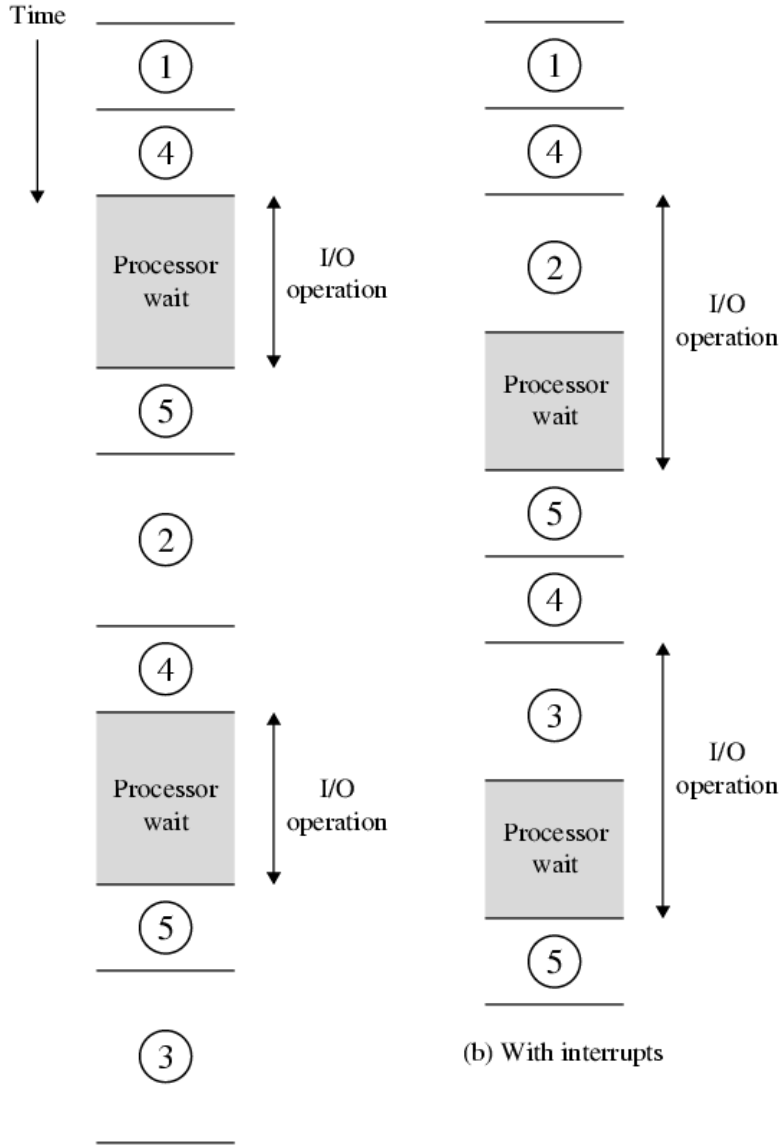
(a) No interrupts

(b) Interrupts; short I/O wait

(c) Interrupts; long I/O wait

Program Timing

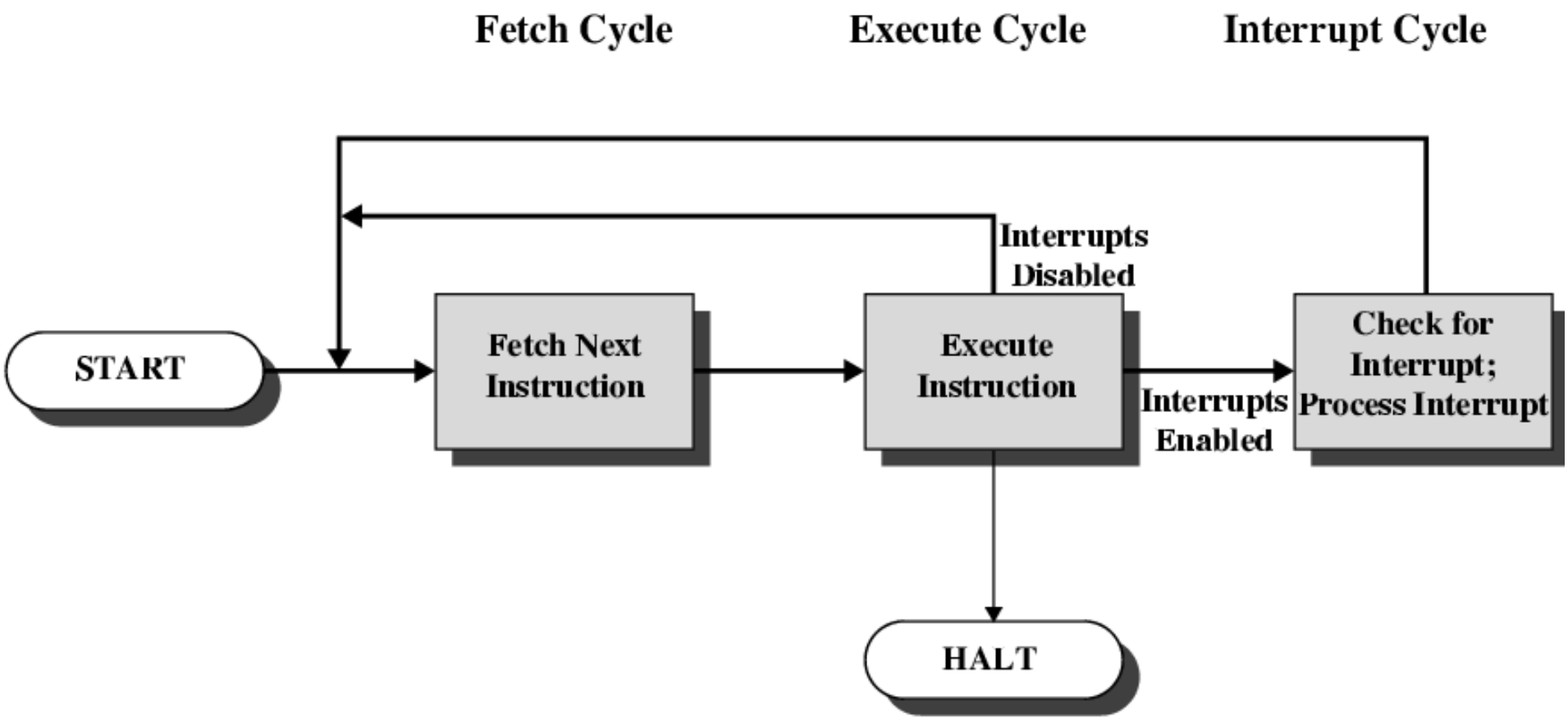




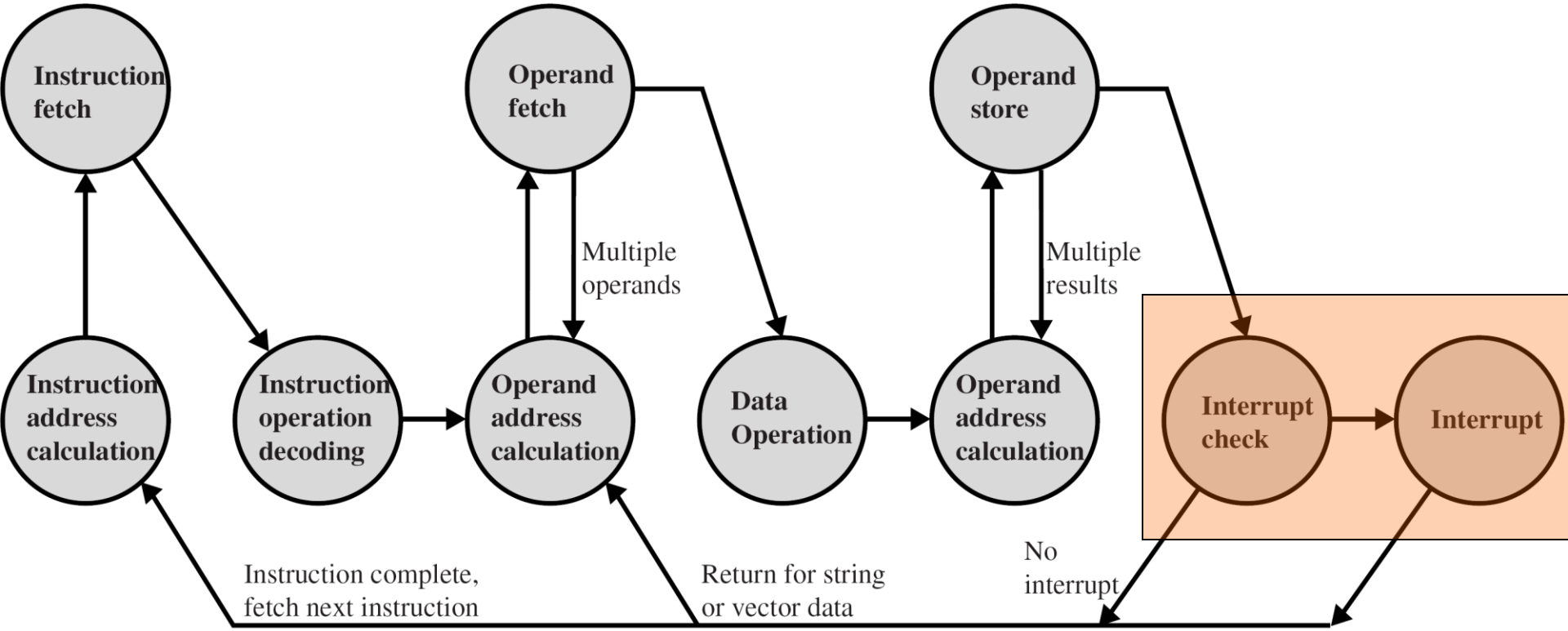
Interrupt Cycle

- Added to instruction cycle.
- Processor checks for interrupt
  - —Indicated by an interrupt signal
- If no interrupt:
  - —fetch next instruction
- If interrupt pending:
  - —Suspend execution of current program.
  - —Save context (PC & relevant data).
  - —Set PC to start address of interrupt handler routine.
  - —Process interrupt.
  - —Restore context and continue interrupted program.

Instruction Cycle (with interrupts)



Instruction Cycle: State Diagram (with interrupts)



Reading Material

- Stallings, Chapter 3:
  - Pages 66 – 80