CSE 360-Computer Architecture

Lecture-3

Computer Components

Dr. Shamim Akhter

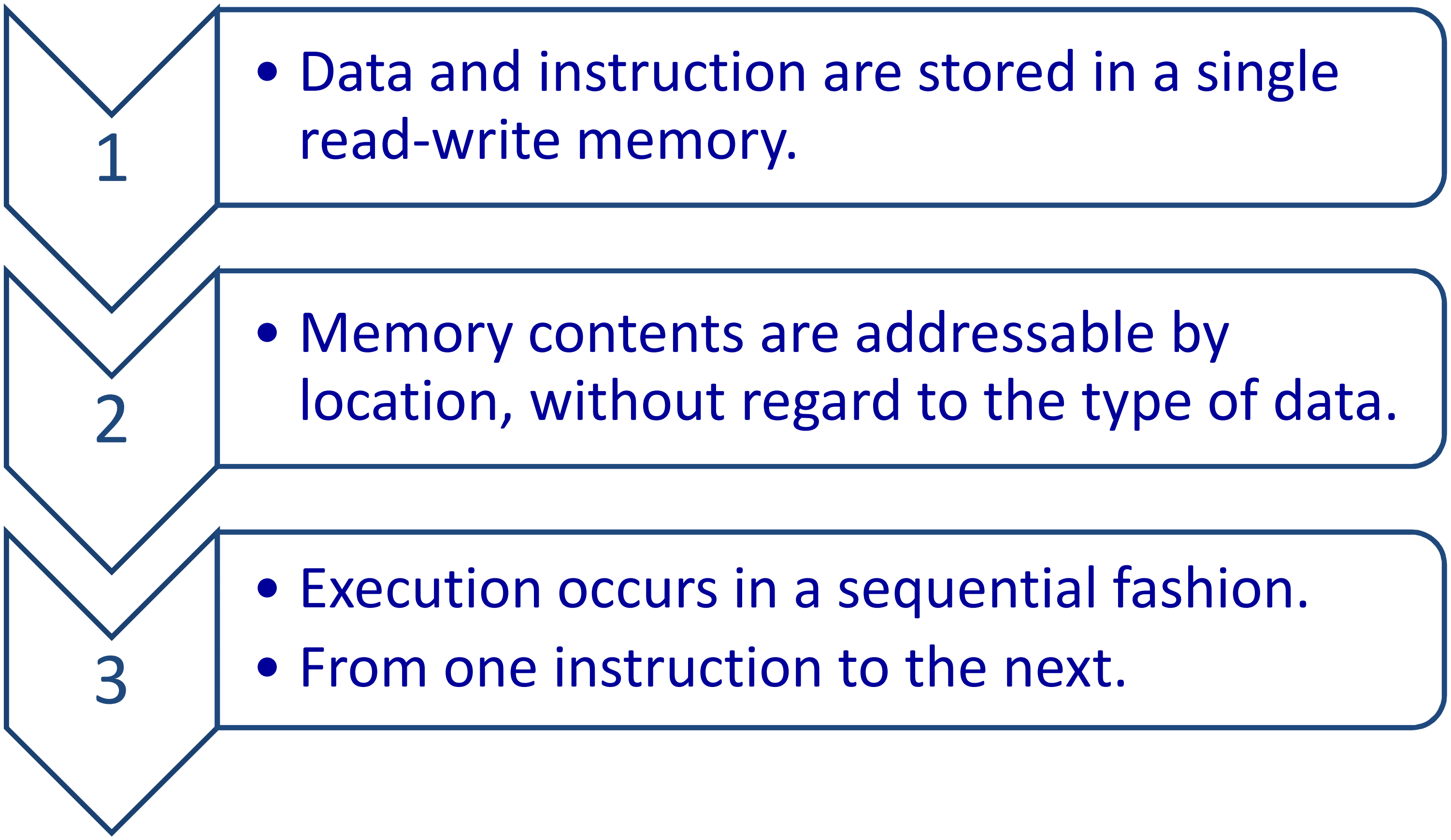
Associate Professor

Department of Computer Science and Engineering

email: shamimakhter@ewubd.edu

Von Neumann Architecture

Based on three(3) key concepts:



1

- Data and instruction are stored in a single read-write memory.

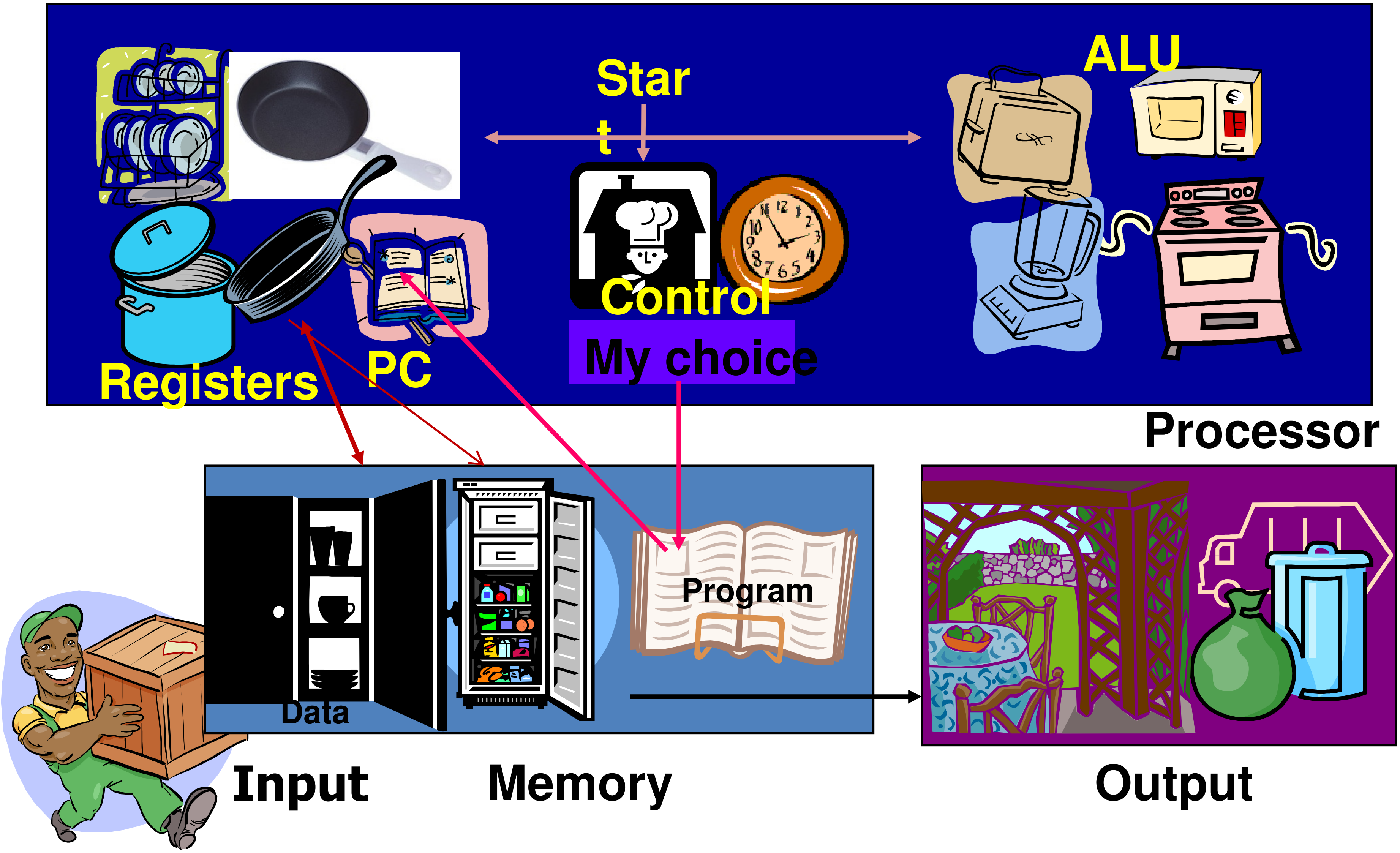
2

- Memory contents are addressable by location, without regard to the type of data.

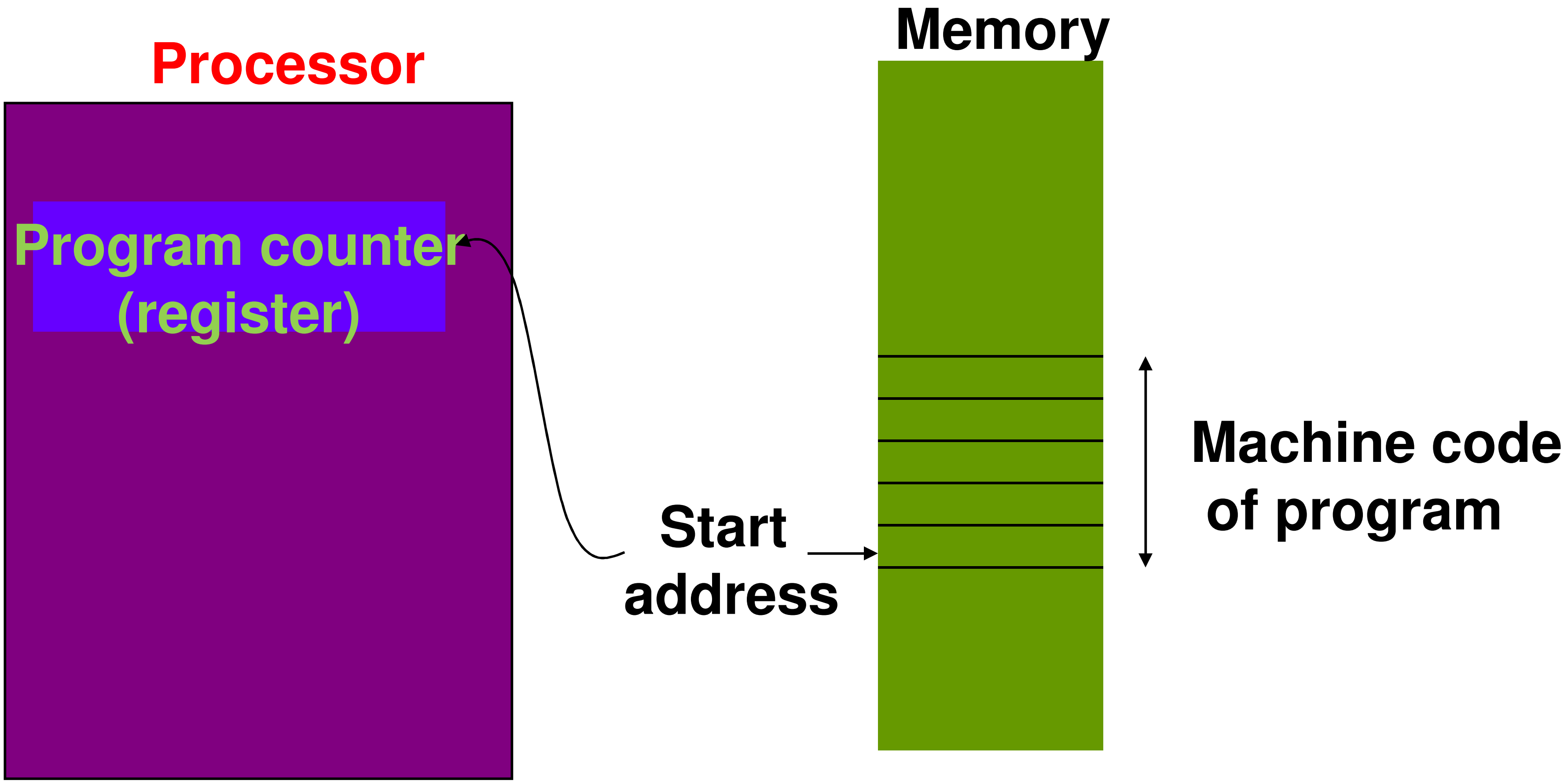
3

- Execution occurs in a sequential fashion.
- From one instruction to the next.

Von Neumann Kitchen

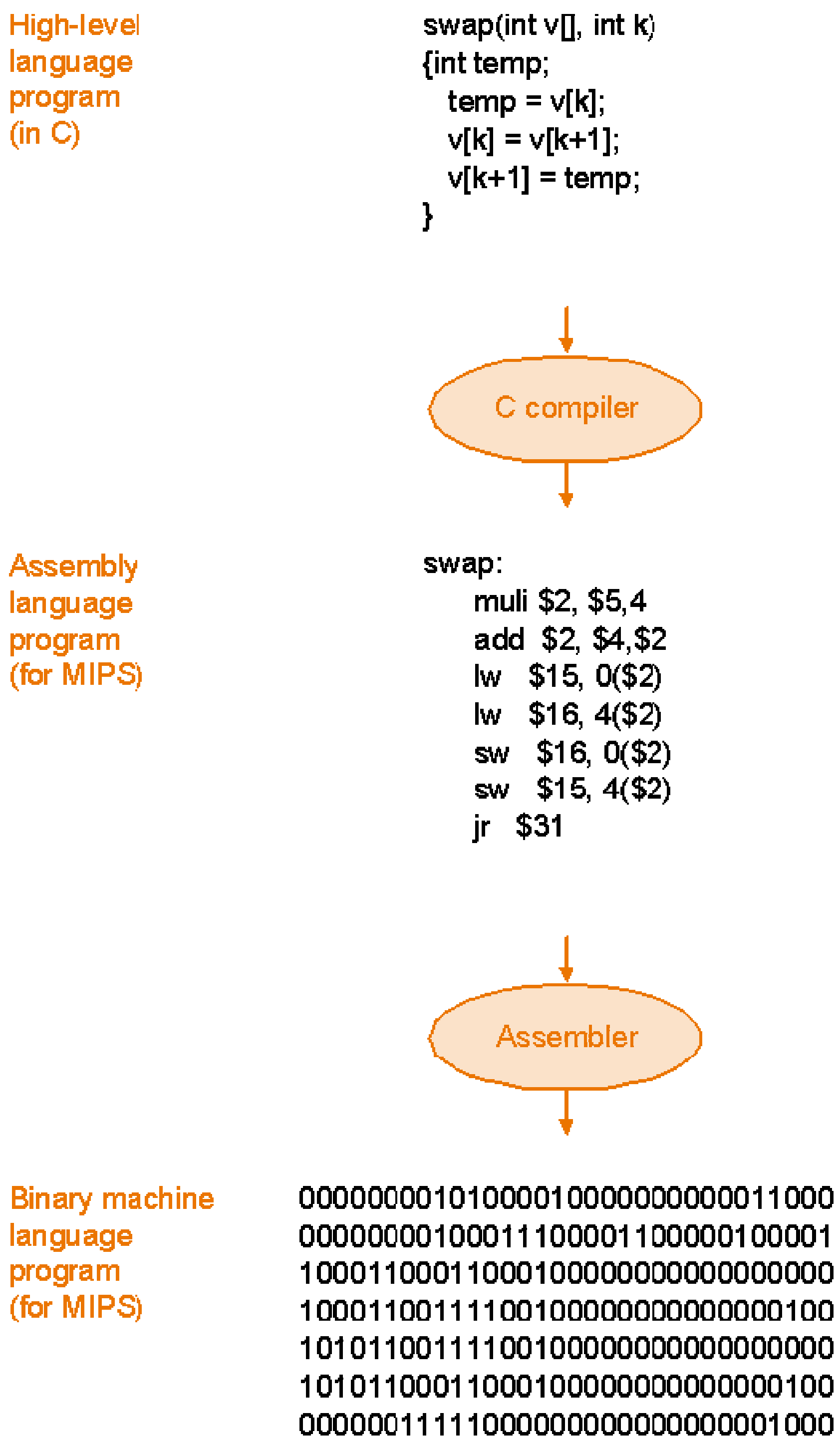


Where is the Program?



Stored Program Concept

The idea that instructions and data of many types can be stored in memory as numbers.


High-level language program (in C)
C compiler
Assembly language program (for MIPS)
Assembler
Binary machine language program (for MIPS)

```
swap(int v[], int k)
{int temp;
   temp = v[k];
   v[k] = v[k+1];
   v[k+1] = temp;
}
```

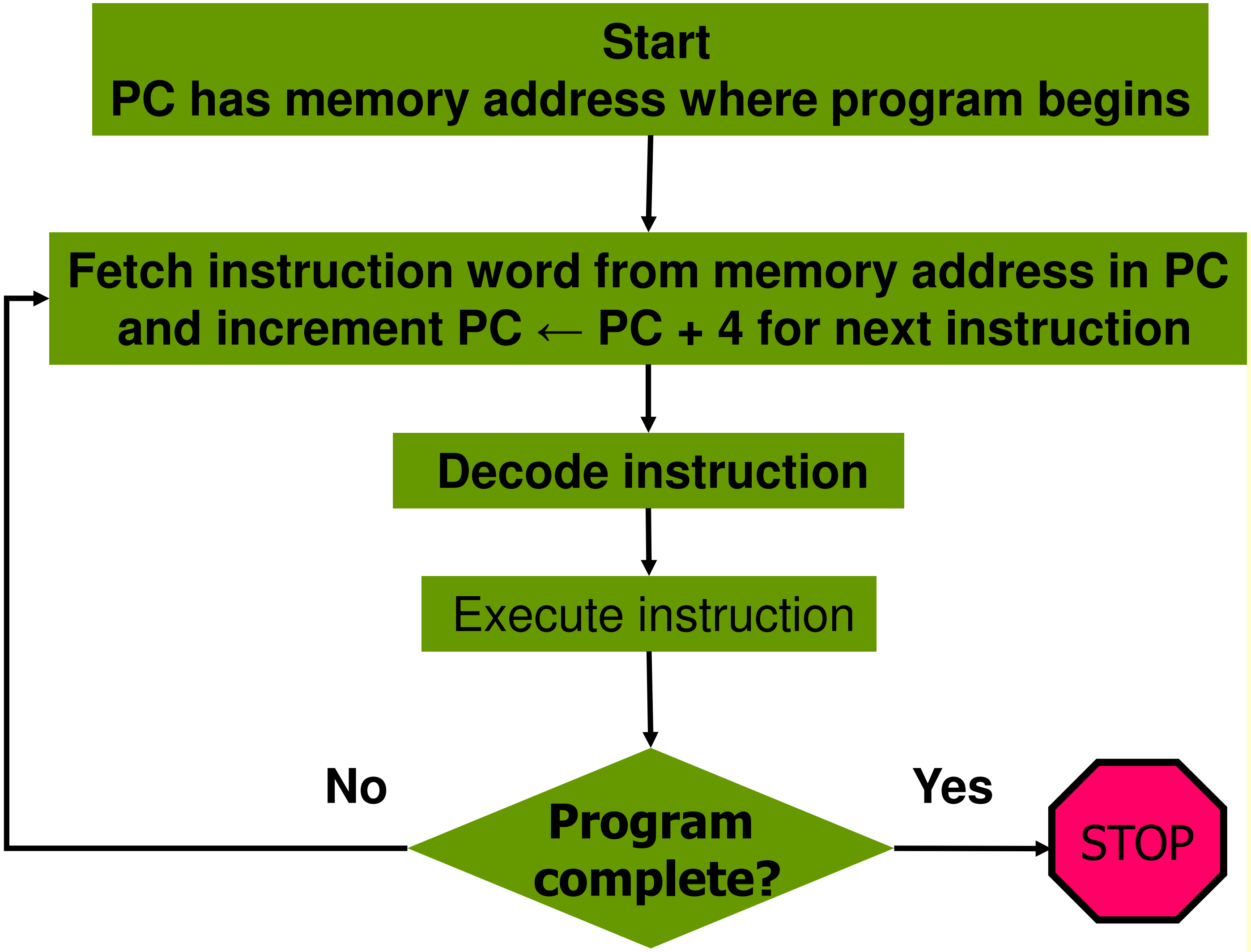
```
swap:
      muli $2, $5,4
      add  $2, $4,$2
      lw   $15, 0($2)
      lw   $16, 4($2)
      sw   $16, 0($2)
      sw   $15, 4($2)
      jr   $31
```

```
00000000101000010000000000011000
00000000100011100001100000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
00000011111000000000000000001000
```

Where Does It All Begin?

- In a register called *program counter (PC)*.
- PC contains the memory address of the next instruction to be executed.
- In the beginning, PC contains the address of the memory location where the program begins.

How Does It Run?



Start
PC has memory address where program begins

Fetch instruction word from memory address in PC and increment PC ← PC + 4 for next instruction

Decode instruction

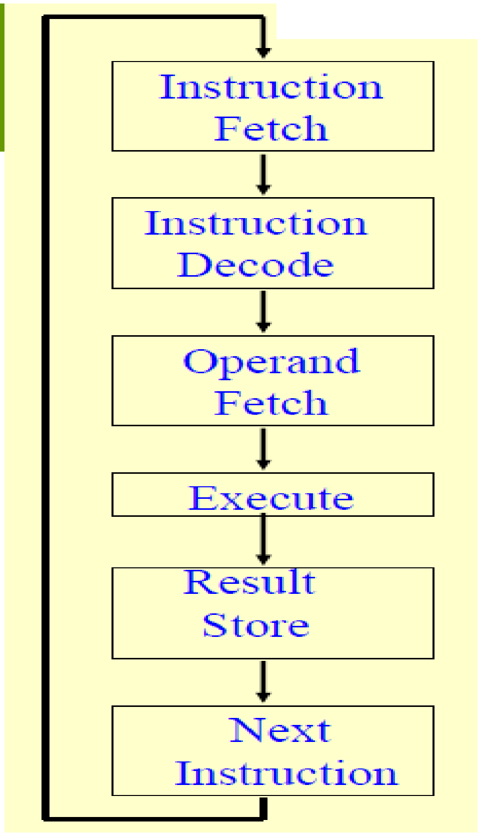
Execute instruction

Program complete?

No

Yes

STOP



Instruction Fetch

Instruction Decode

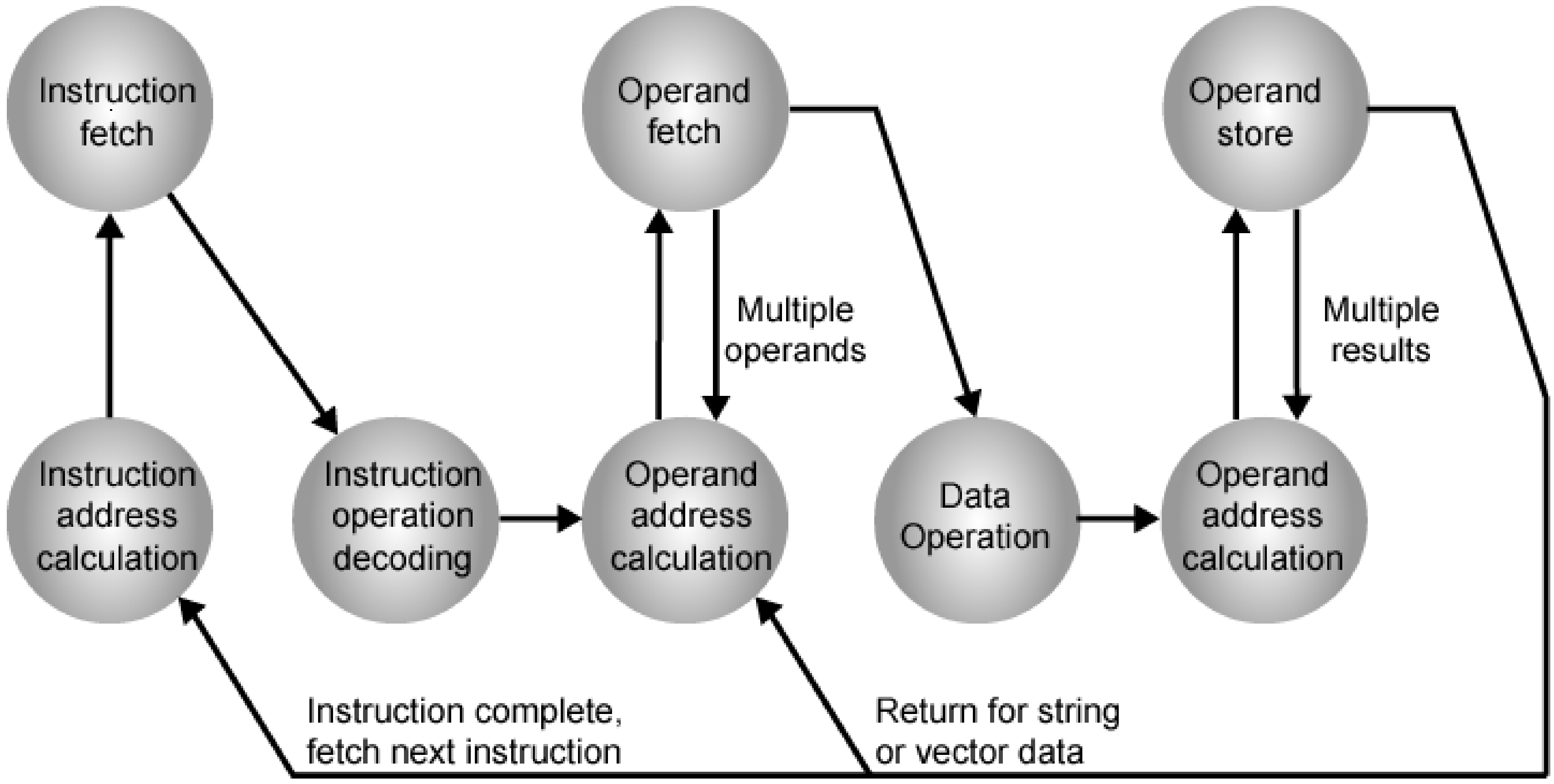
Operand Fetch

Execute

Result Store

Next Instruction

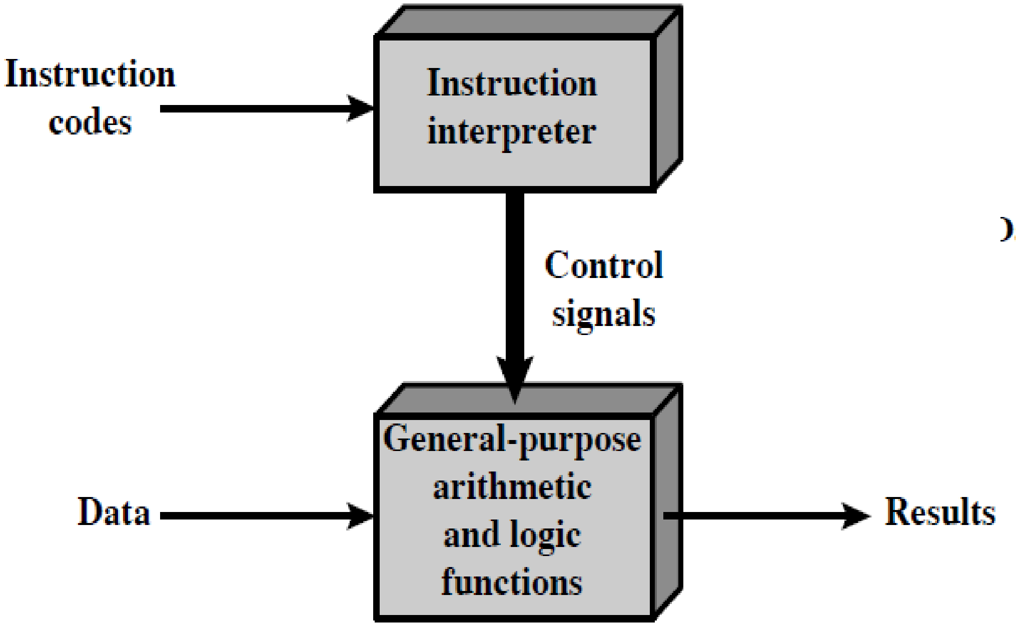
Instruction Cycle State Diagram


Instruction fetch
Operand fetch
Operand store
Multiple operands
Multiple results
Instruction address calculation
Instruction operation decoding
Operand address calculation
Data Operation
Operand address calculation
Instruction complete, fetch next instruction
Return for string or vector data

Program Execution Concept

Programming connects various components in the desired configuration. Two different approaches

Entire program is a sequence of codes/instructions (also called *software*)

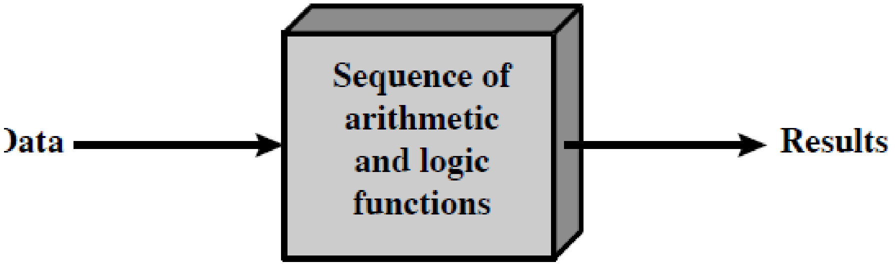


(b) Programming in software

General purpose configuration ALU operations

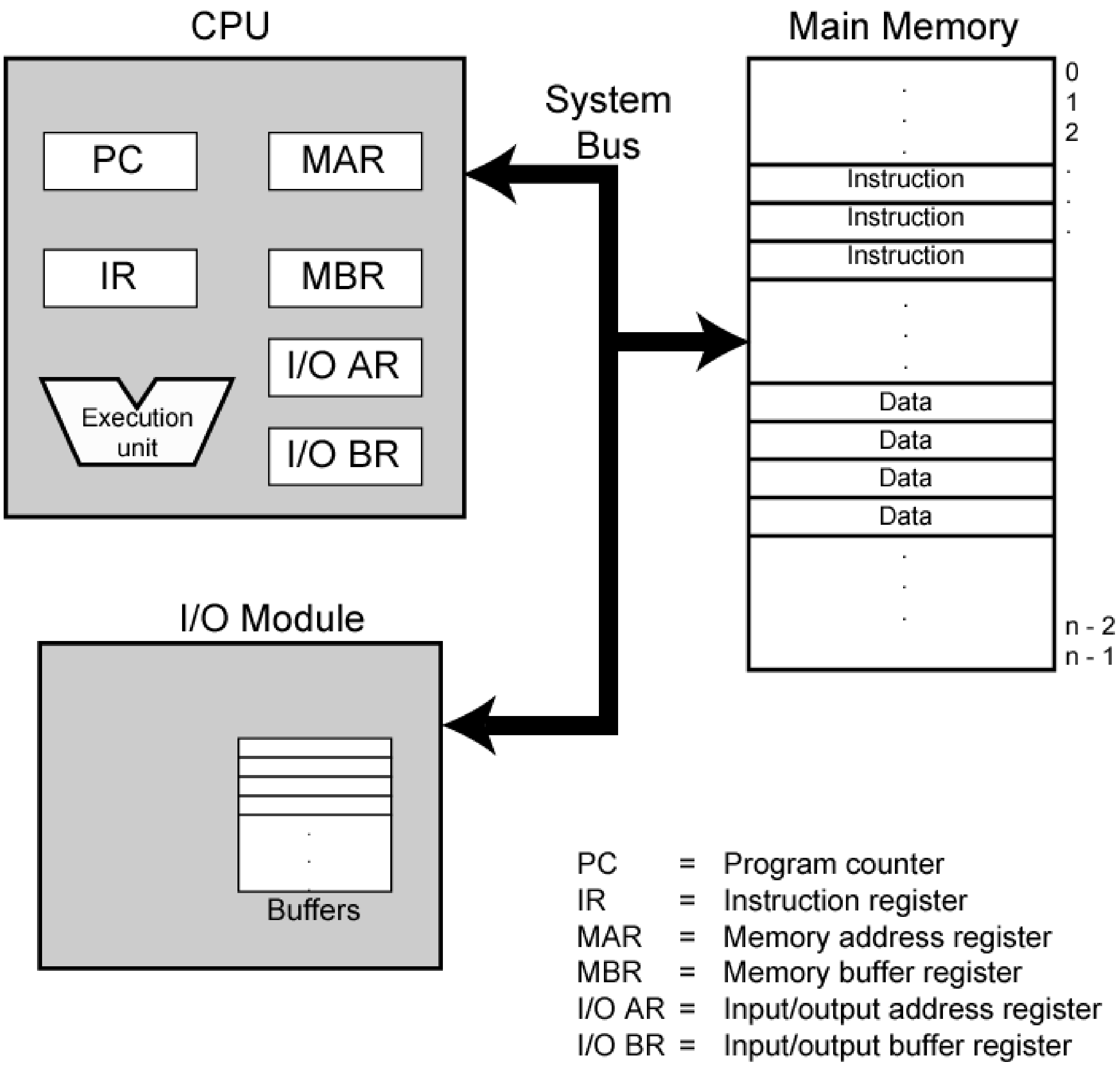
In the form of H/W

Hardwire



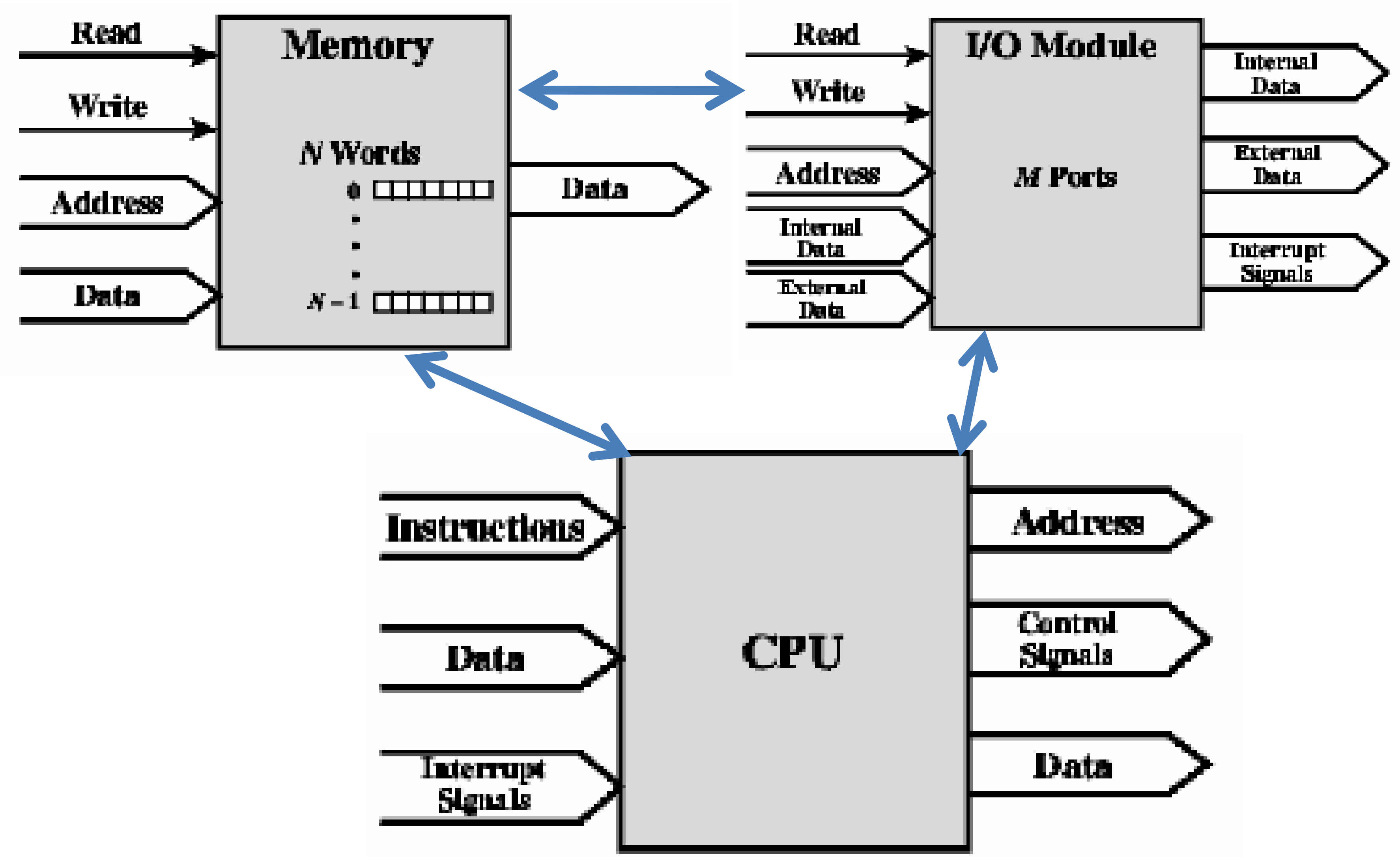
(a) Programming in hardware

Computer Components

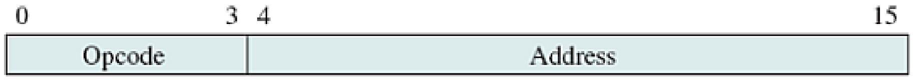


PC	=	Program counter
IR	=	Instruction register
MAR	=	Memory address register
MBR	=	Memory buffer register
I/O AR	=	Input/output address register
I/O BR	=	Input/output buffer register

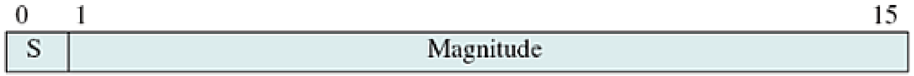
Computer Modules with Interconnection Structure



Example of a Program Execution
@ Hypothetical Machine



(a) Instruction format



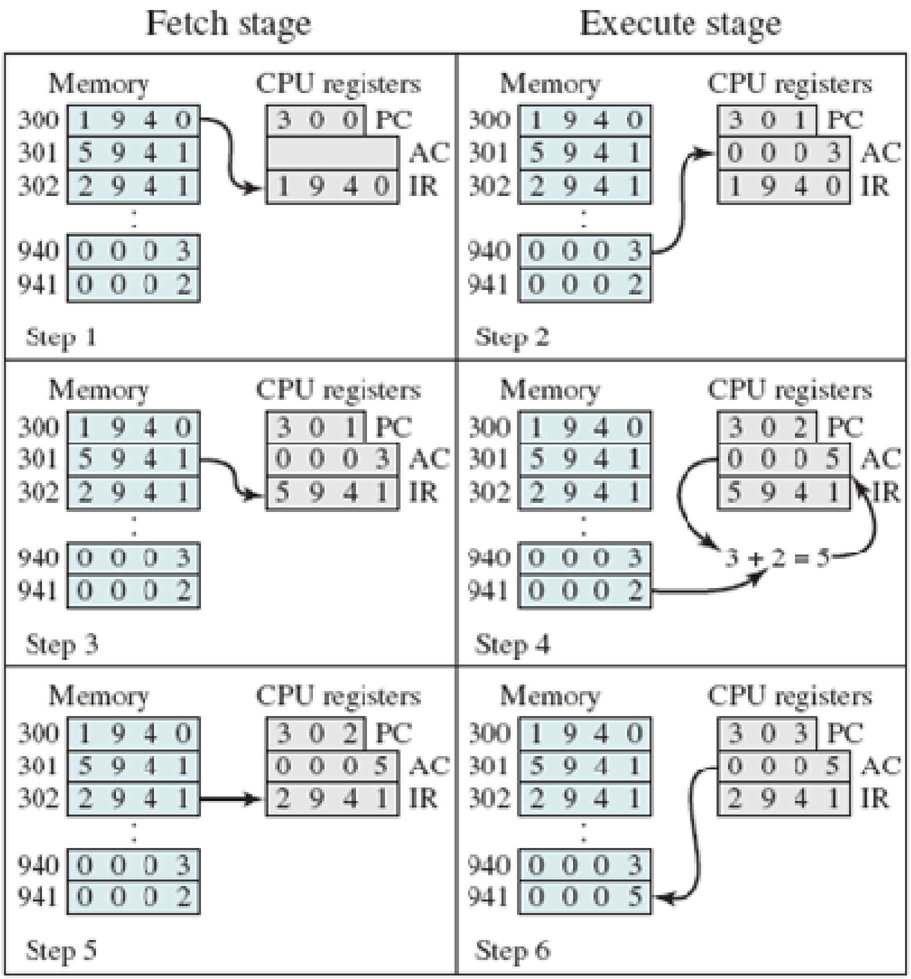
(b) Integer format

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

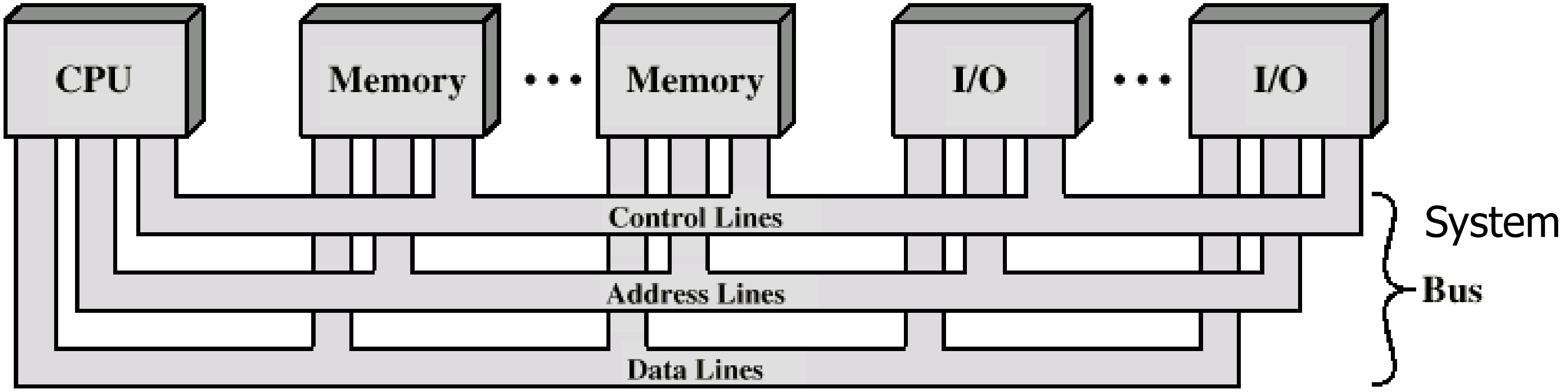
(d) Partial list of opcodes



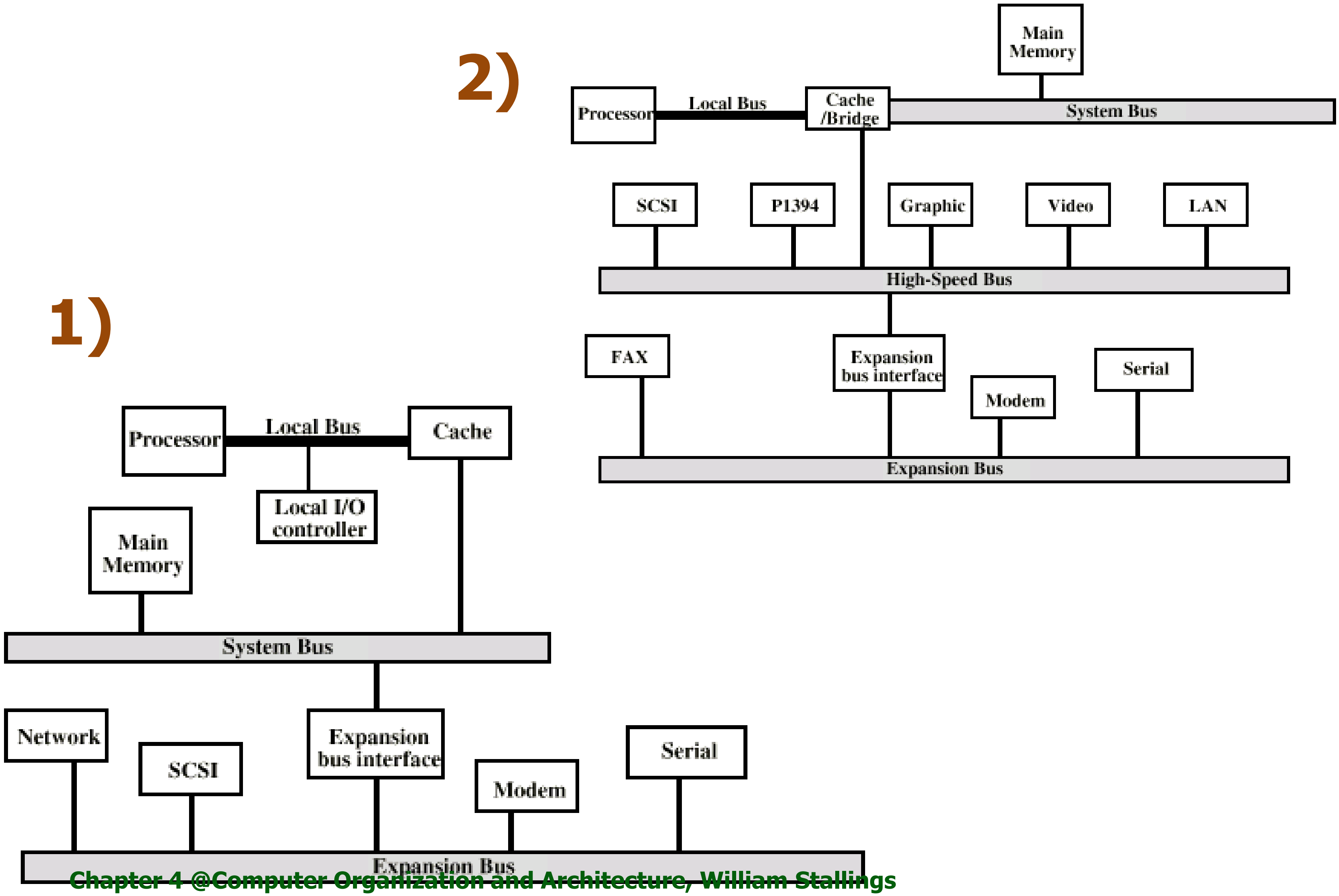
Bus Interconnection

- What is BUS?
  - A bus (group of electrical lines/wires) is a shared transmission medium, that carries computer signals.
  - Computer signals: 1 bit memory address, a sequence of data bits, or timing control that turns a device on or off.

Bus Structure



Multiple Bus Configuration



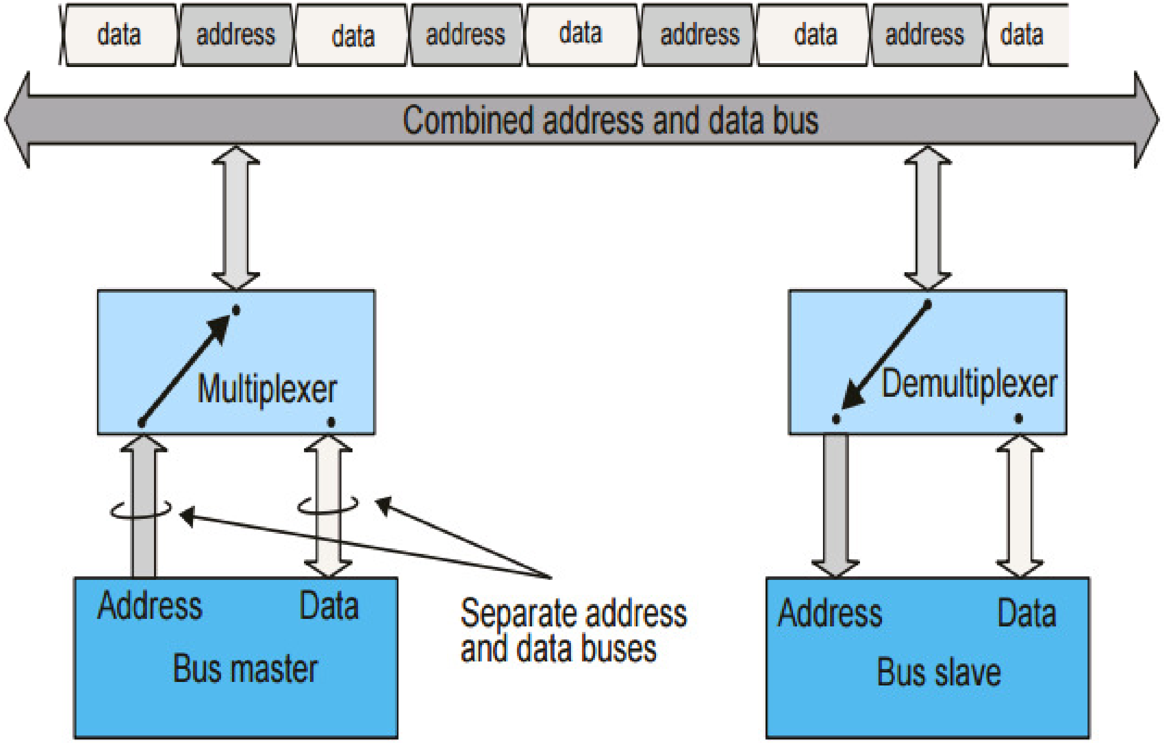
Chapter 4 @Computer Organization and Architecture, William Stallings

Element of Bus Design: Bus Width

- Data
- Address
- Control

Element of Bus Design: Type

- Dedicated
- Multiplexed

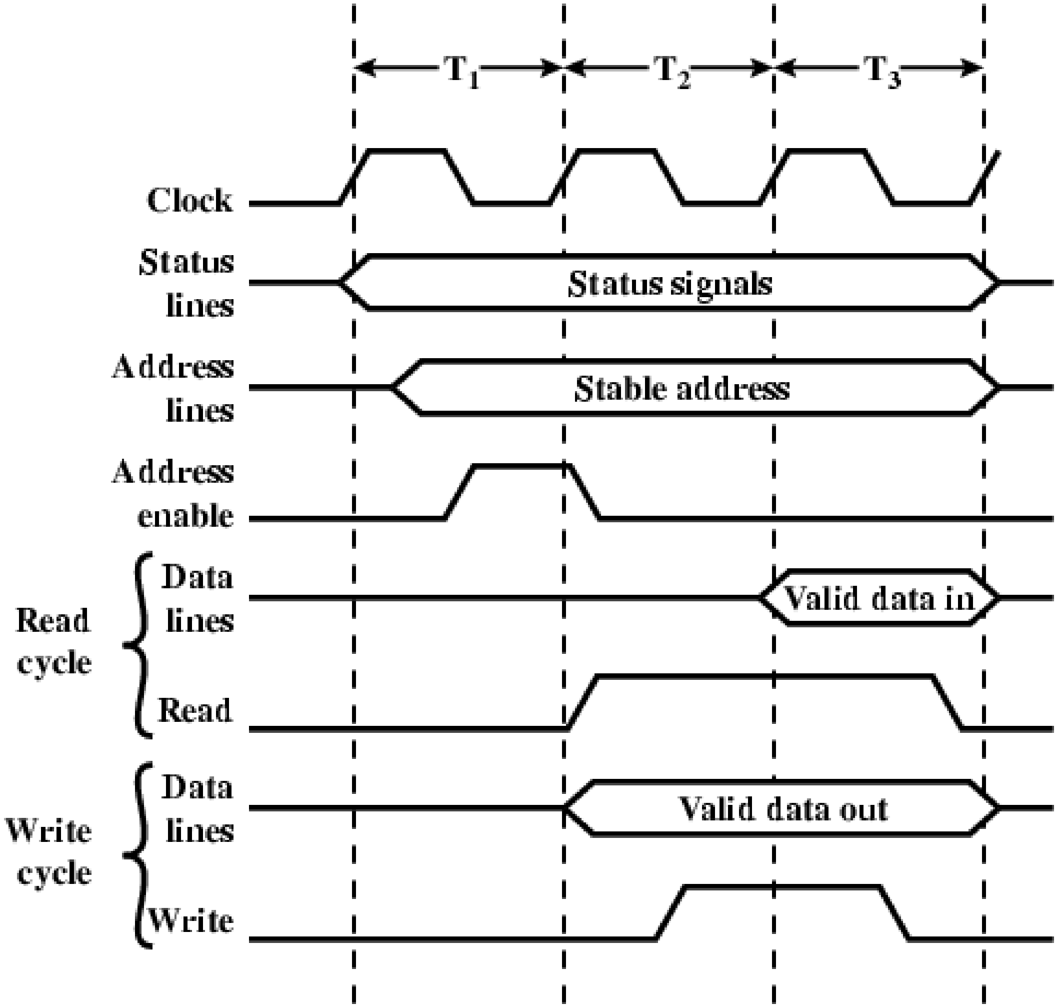


Element of Bus Design: Arbitration

- **Centralized:**
  - an arbitration circuit (bus controller/ arbiter) receives requests from the contending bus masters and then decides which of them is to be given control of the bus.
  - may be part of CPU or separate.
- **Distributed:**
  - No central controller each module may claim the bus
  - Each module contains access control logic and the modules act together to share the bus.

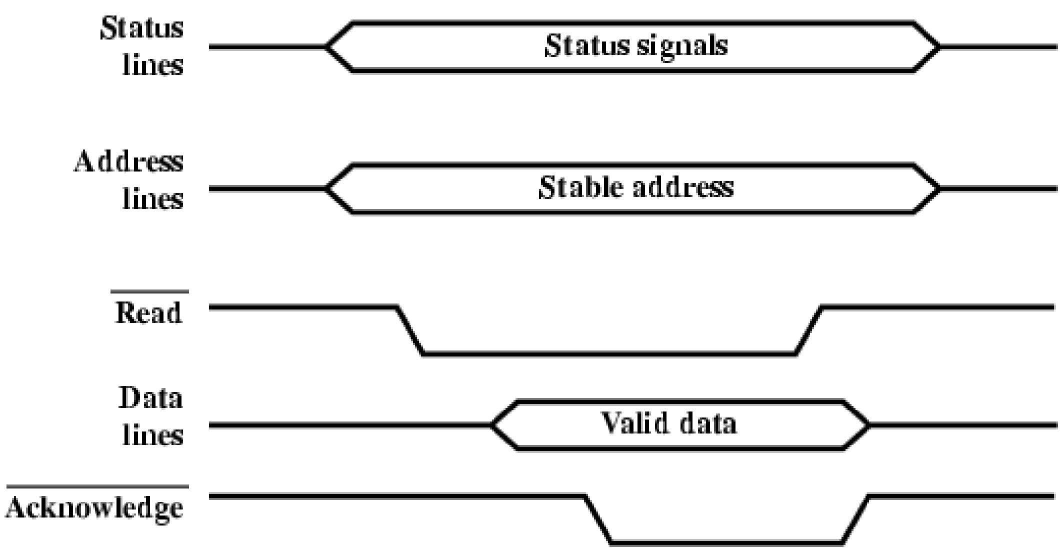
Element of Bus Design: Timing

Synchronous

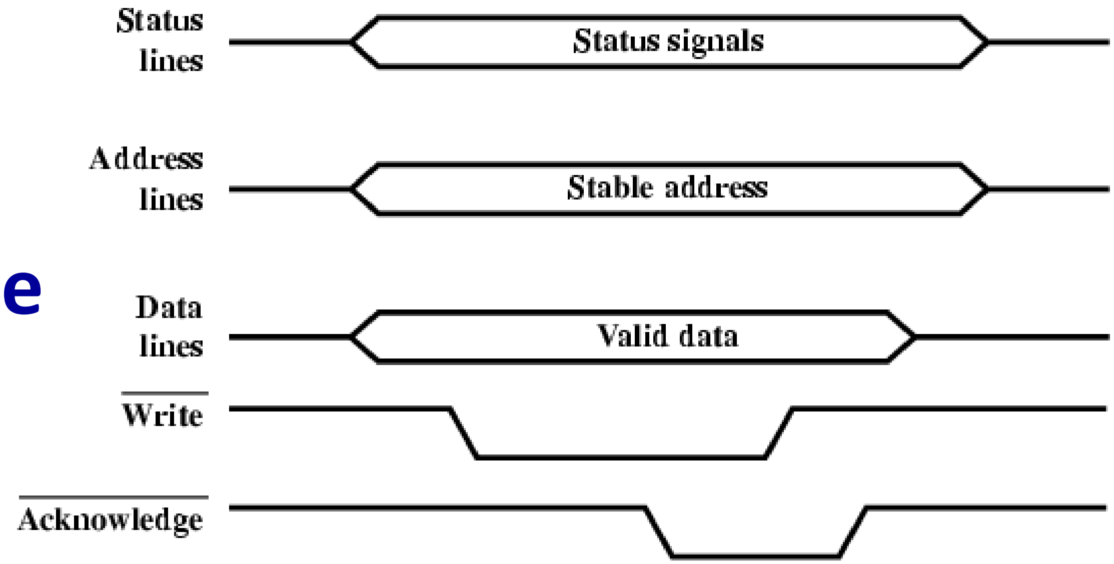


Element of Bus Design: Timing

Asynchronous Read



Asynchronous Write



Chapter 4 @Computer Organization and Architecture, William Stallings