CSE 360-Computer Architecture

Lecture-3

Computer Components

Dr. Shamim Akhter

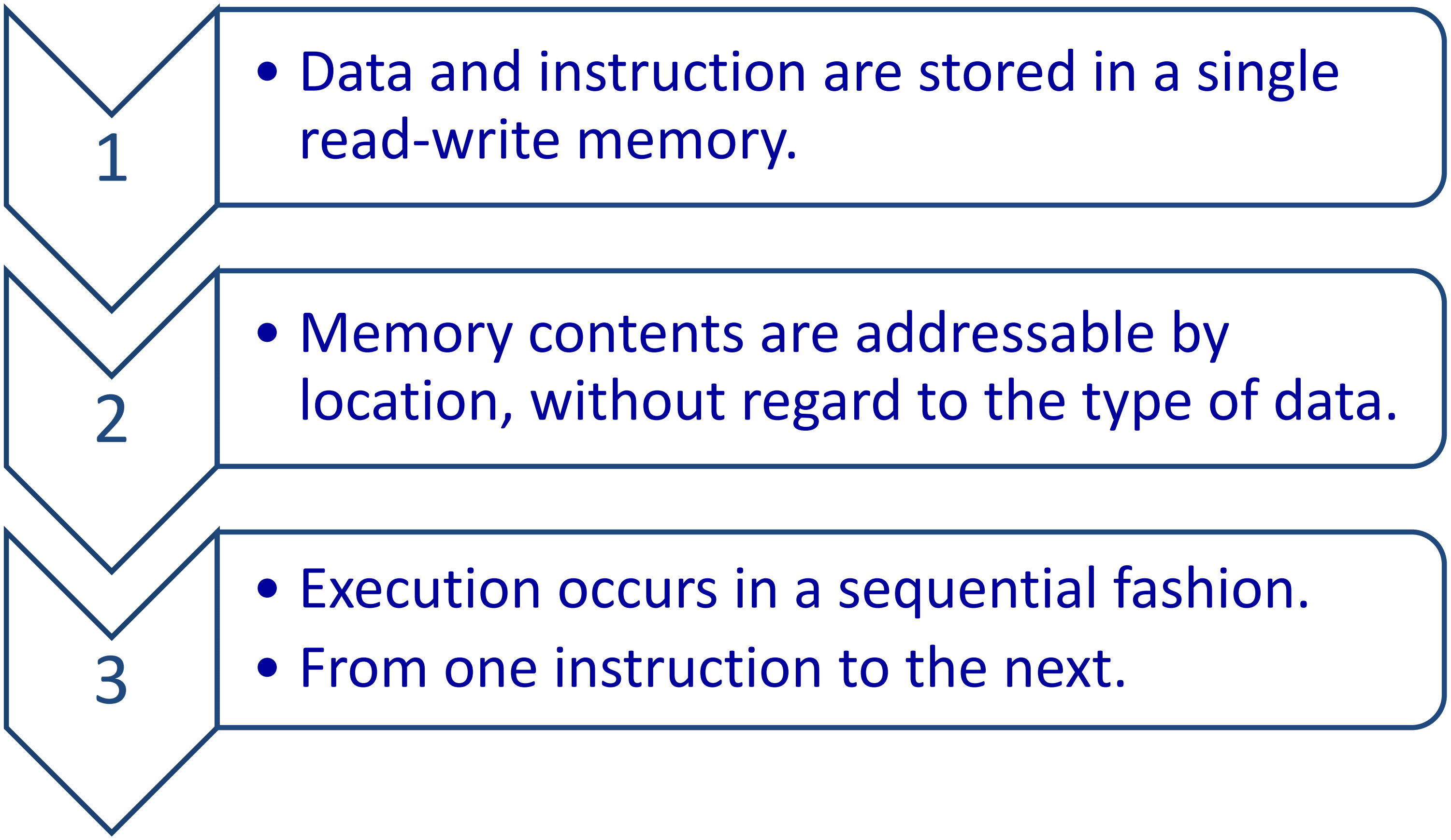
Associate Professor

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.

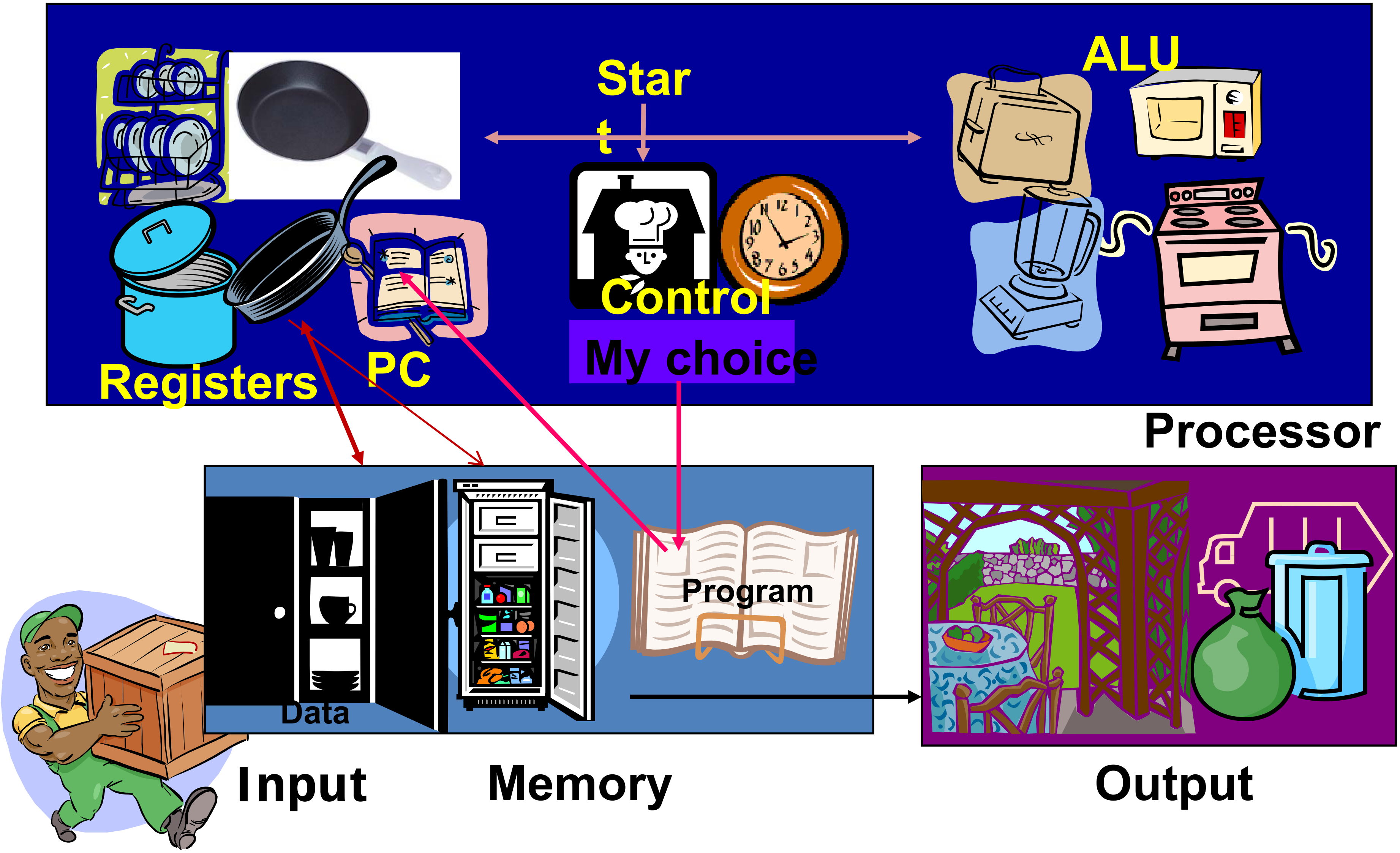
2

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

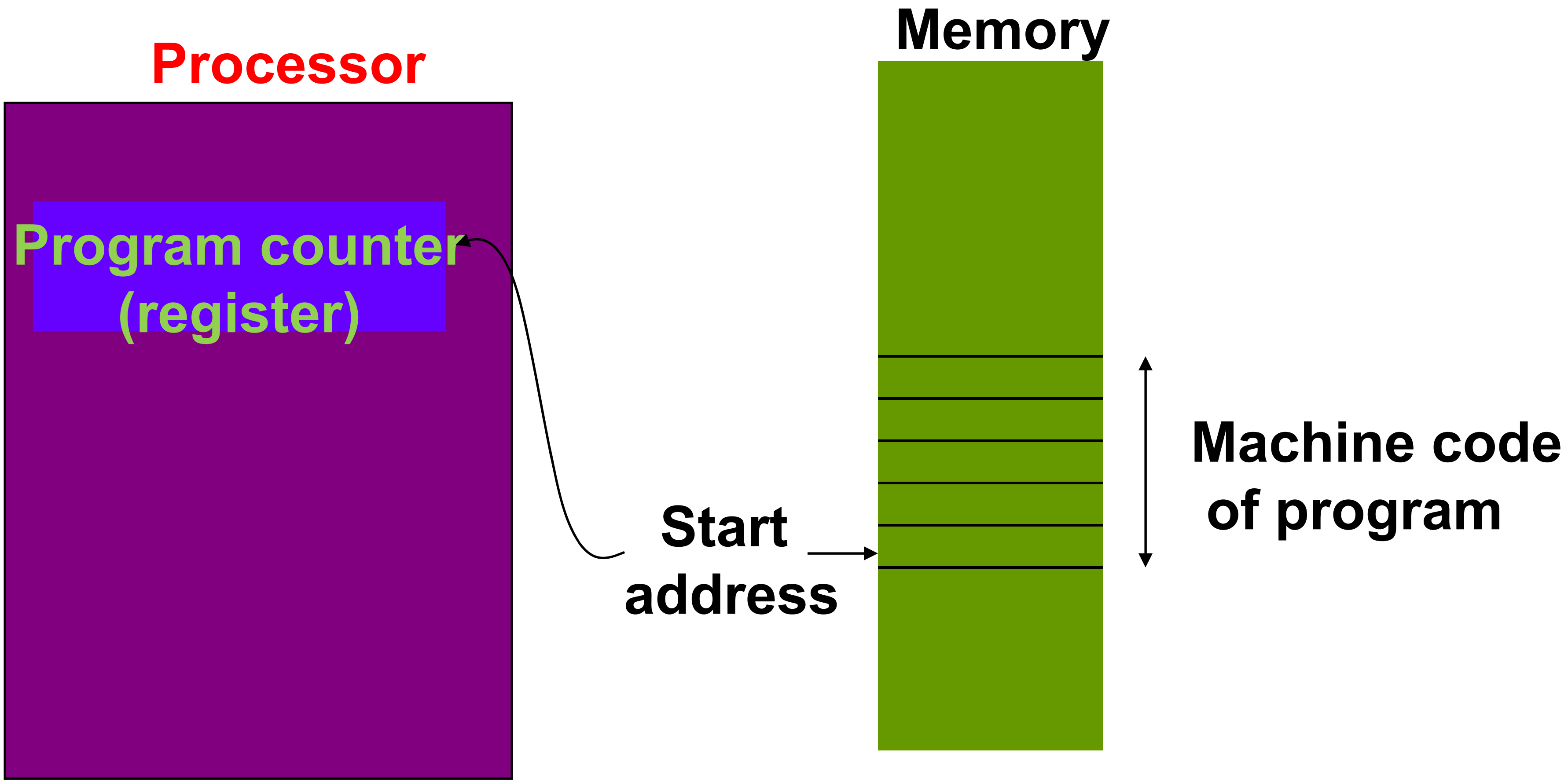
3

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

Von Neumann Kitchen

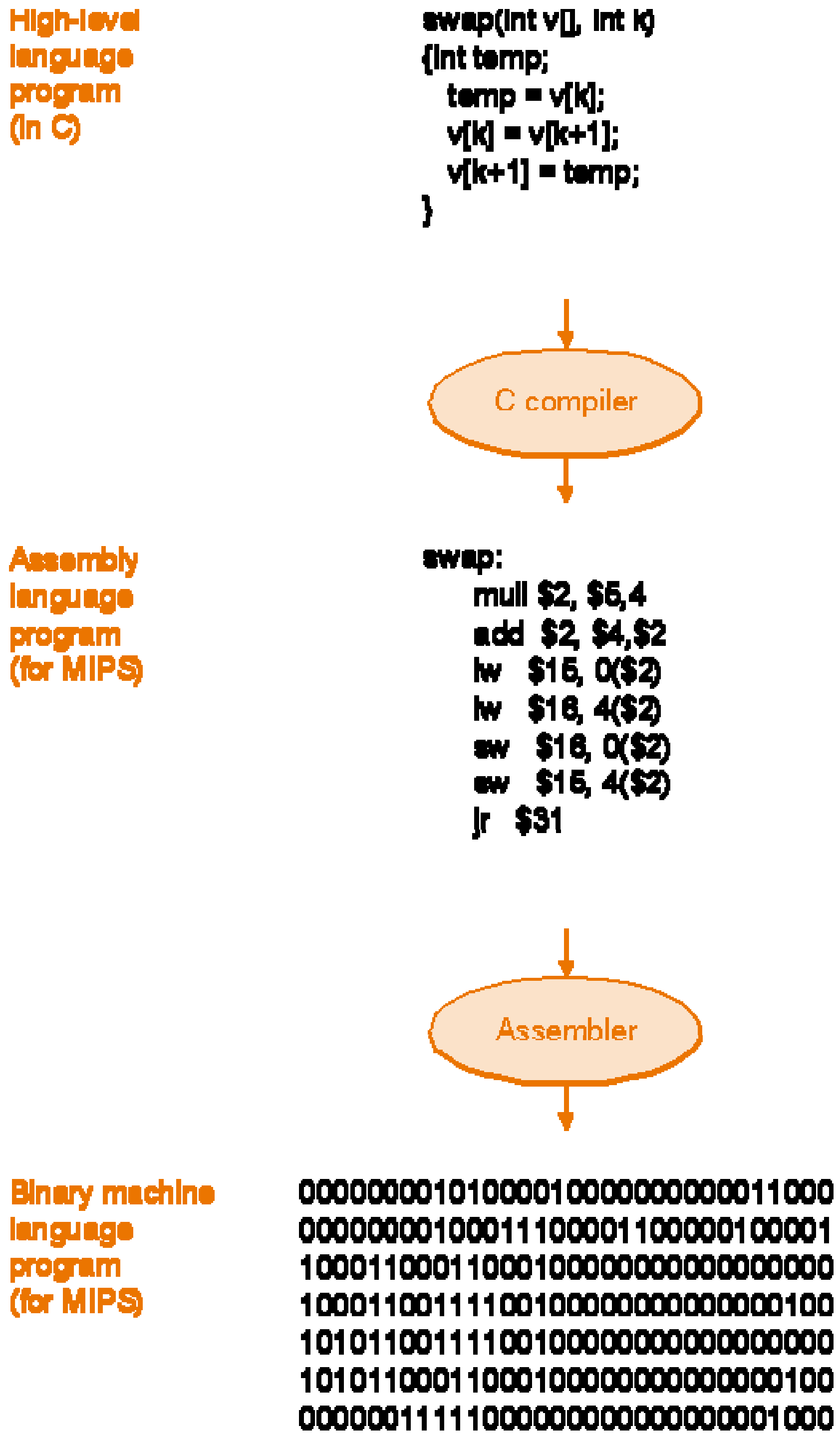


Where is the Program?



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)

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

Assembly language program (for MIPS)

```
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
```

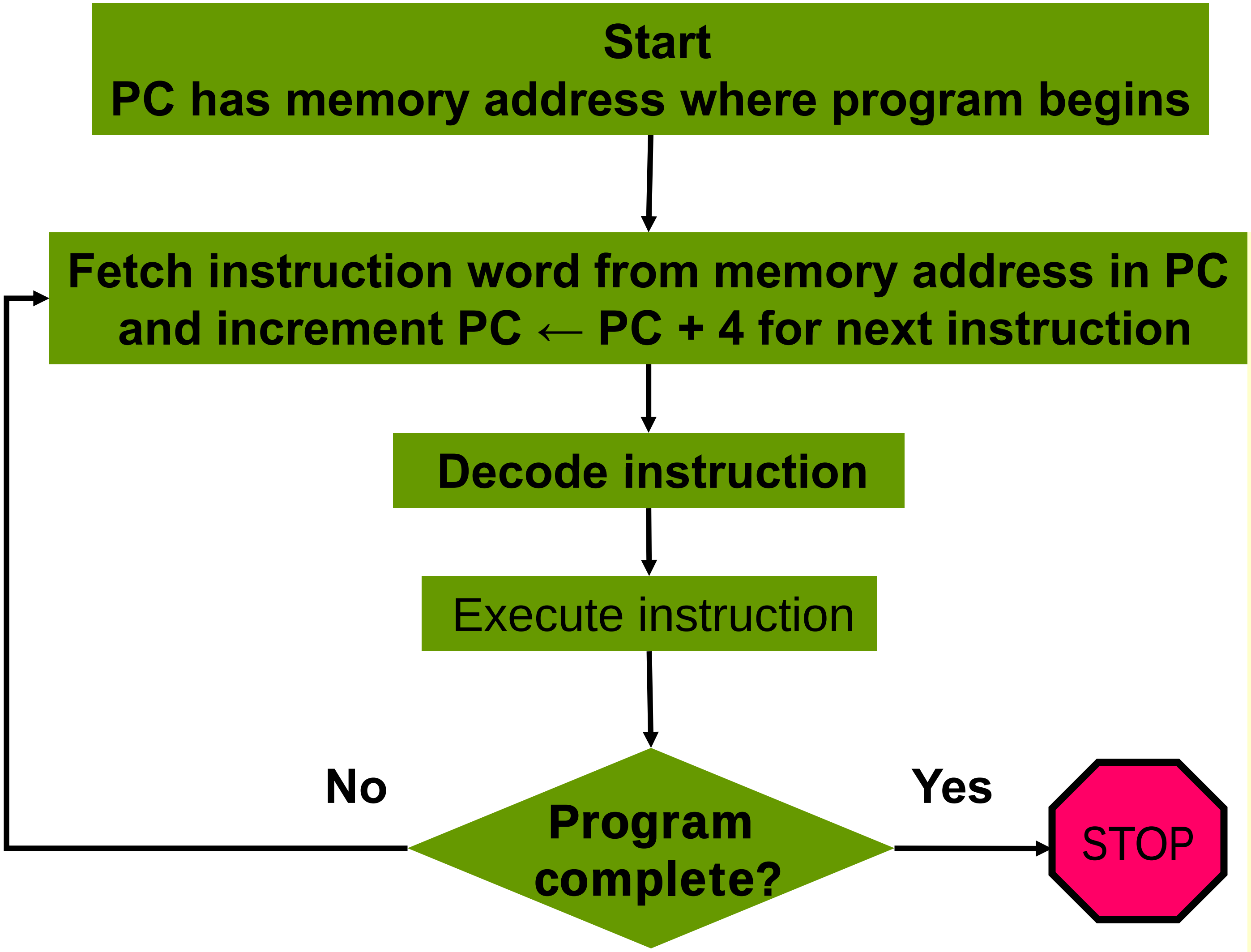
Binary machine language program (for MIPS)

```
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

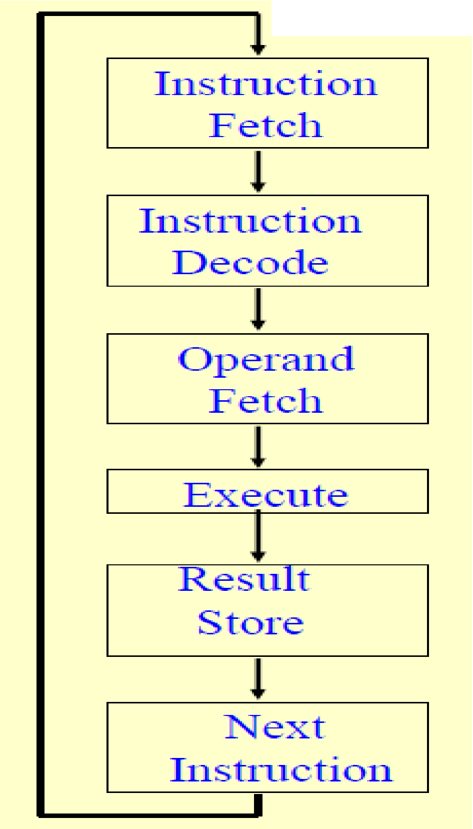
Execute instruction

Program complete?

No

Yes

STOP



Instruction Fetch

Instruction Decode

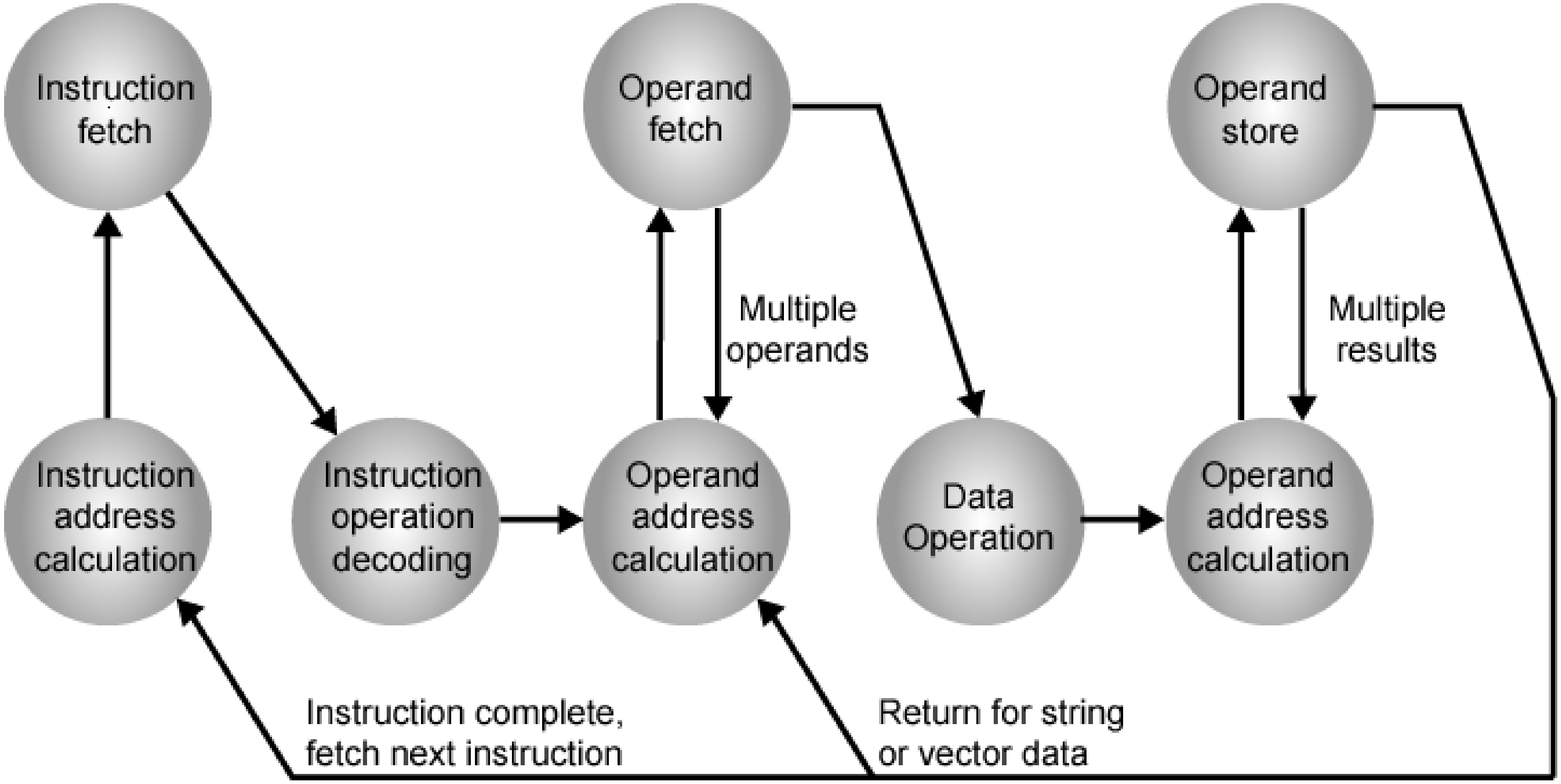
Operand Fetch

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

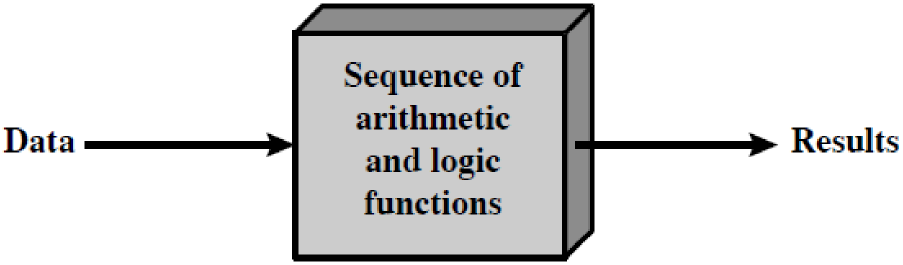
Program Execution Concept

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

General purpose configuration ALU operations

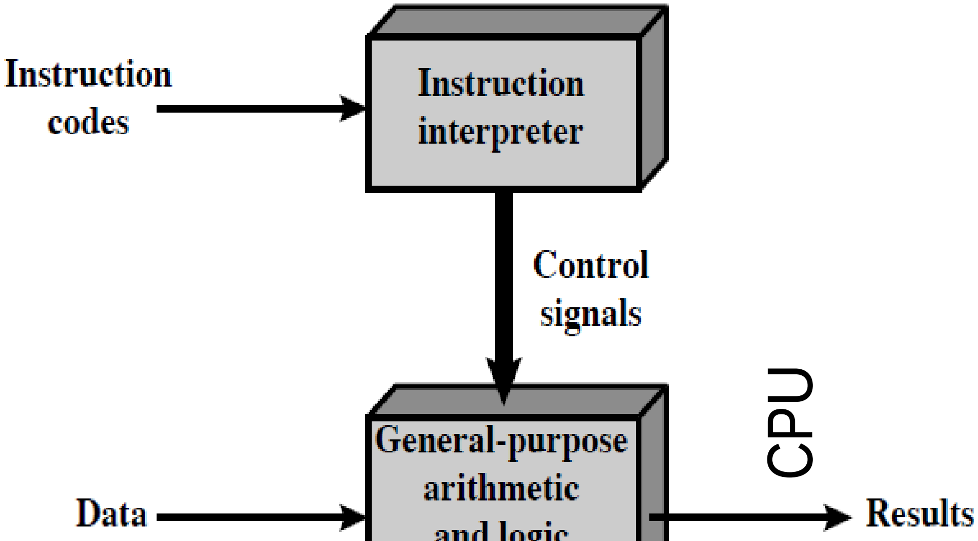
In the form of H/W

Hardwire



(a) Programming in hardware

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



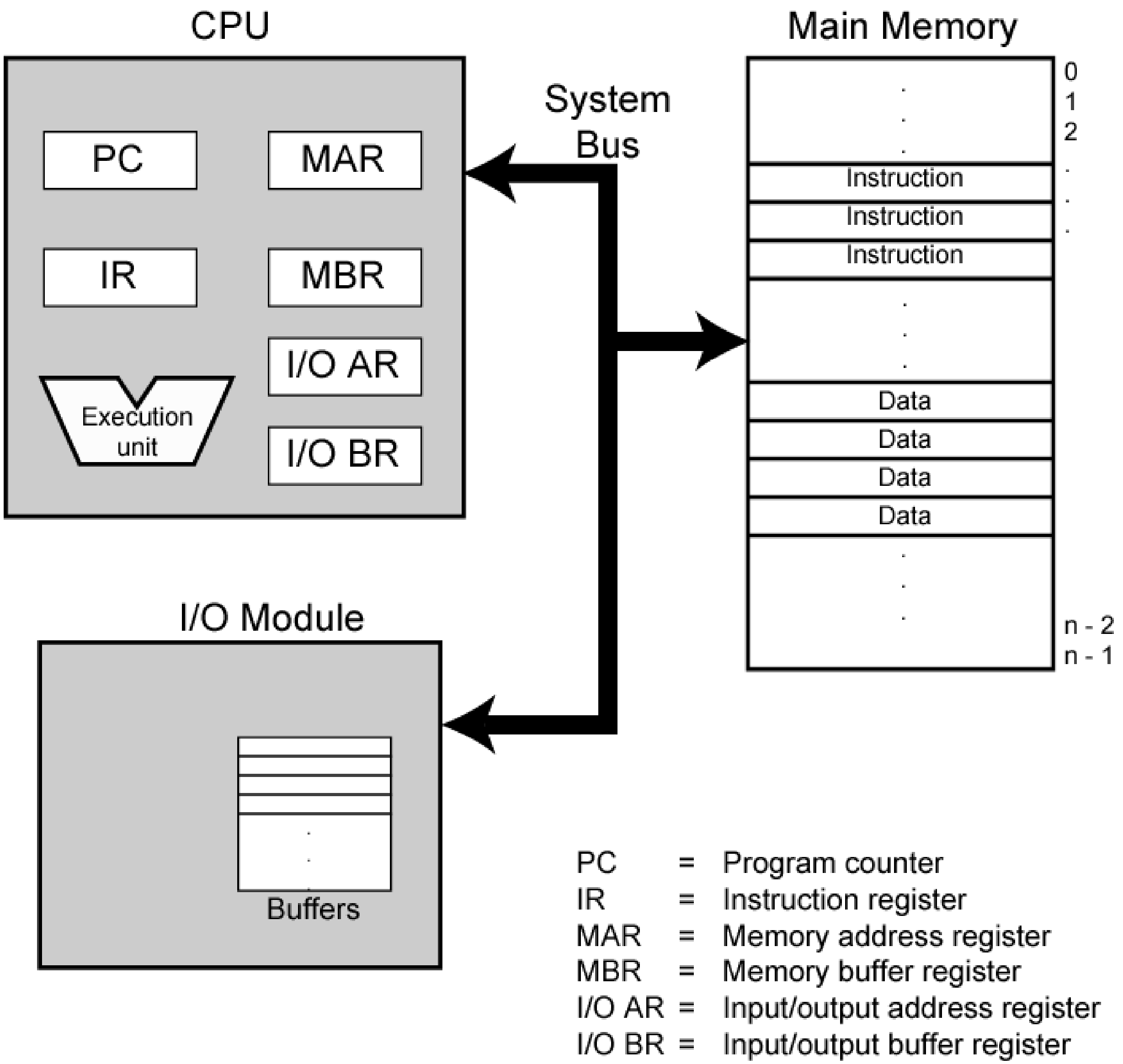
Data and Instruction need to convert as internal form and them put into them system.

I / O components

Need storage for Inputted data and instructions -> Memory module

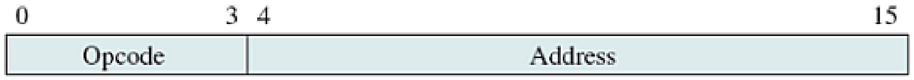
(b) Programming in software

Top Level Computer Components

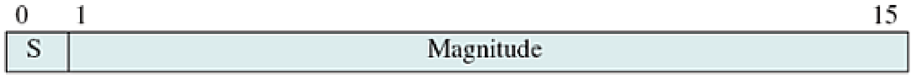


PC	=	Program counter
IR	=	Instruction register
MAR	=	Memory address register
MBR	=	Memory buffer register
I/O AR	=	Input/output address register
I/O BR	=	Input/output buffer register

Example of a Program Execution
@ Hypothetical Machine



(a) Instruction format



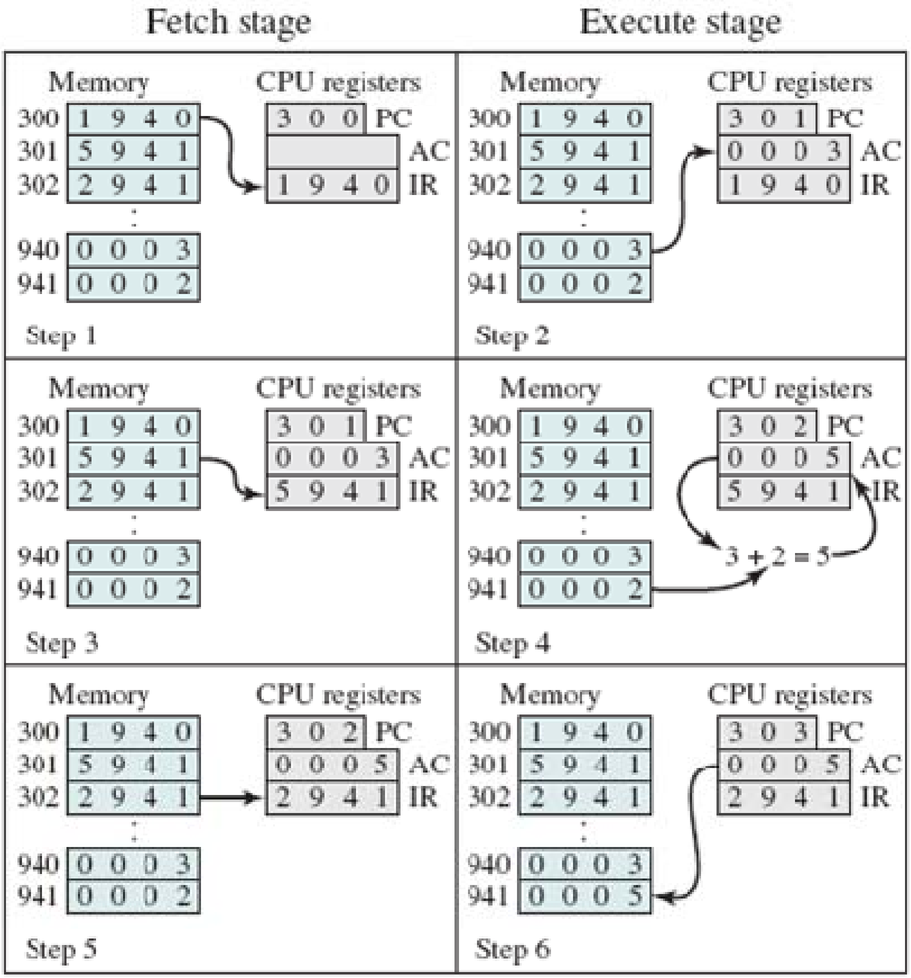
(b) Integer format

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

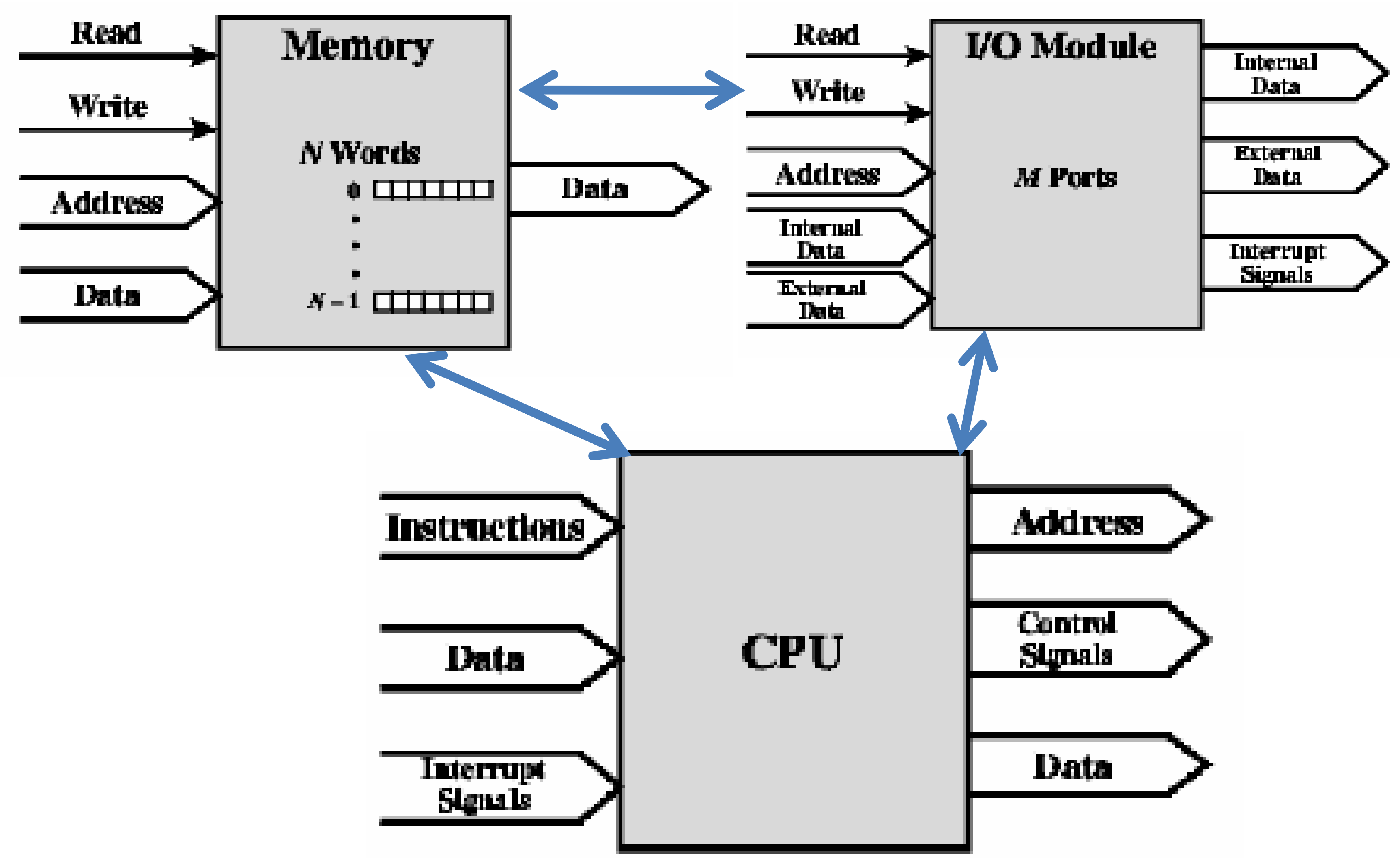
(c) Internal CPU registers

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

(d) Partial list of opcodes

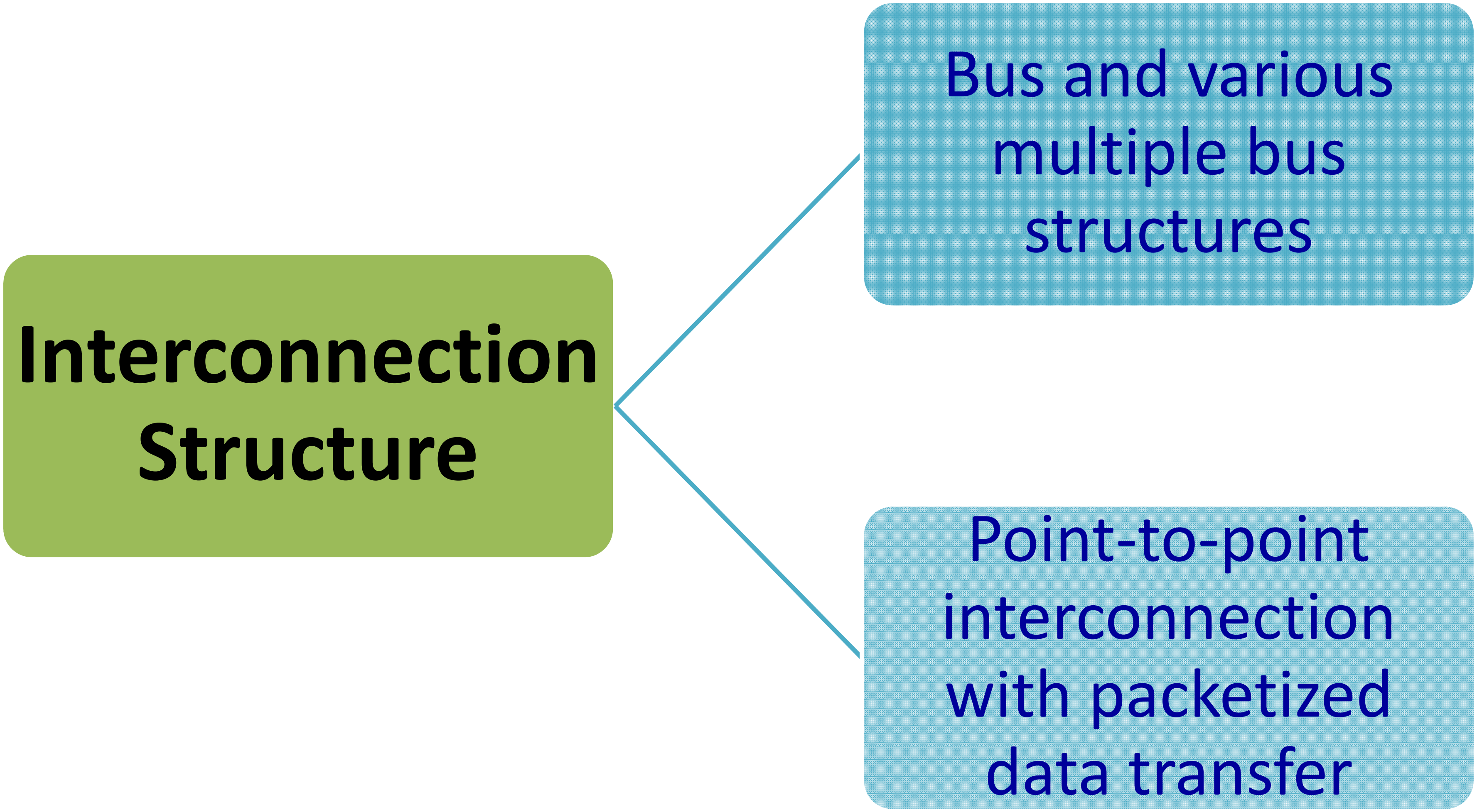


Computer Modules with Interconnection Structure



Chapter 4 @Computer Organization and Architecture, William Stallings

Interconnection Structure



Interconnection Structure

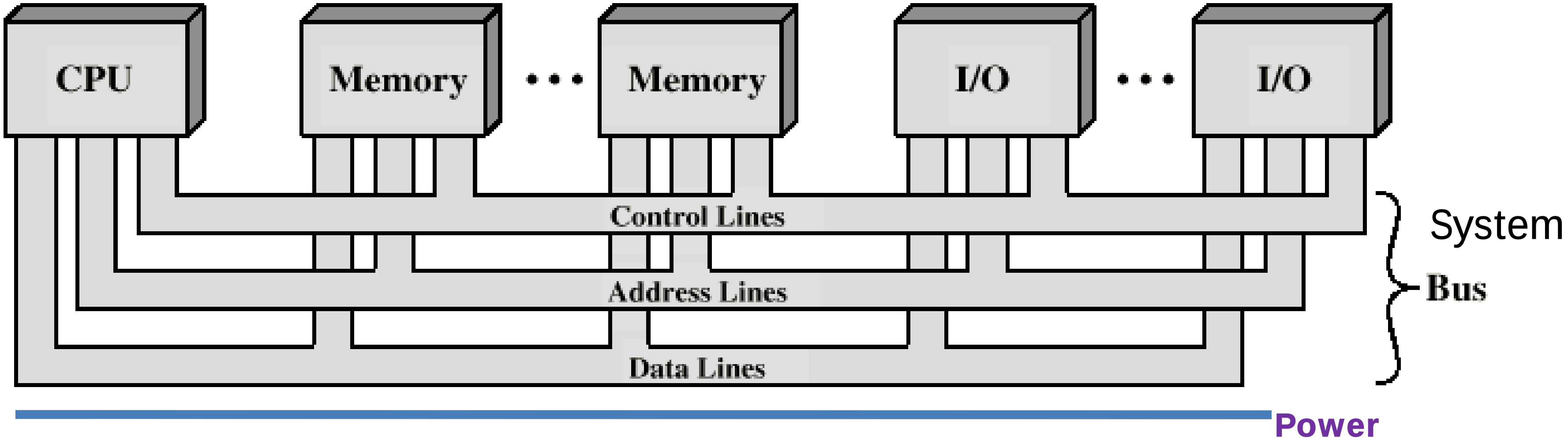
Bus and various multiple bus structures

Point-to-point interconnection with packetized data transfer

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



The operation of the bus is as follows:

-obtain the control to use the bus, transfer data via bus

-transfer a request to the other module over the appropriate control and address lines.

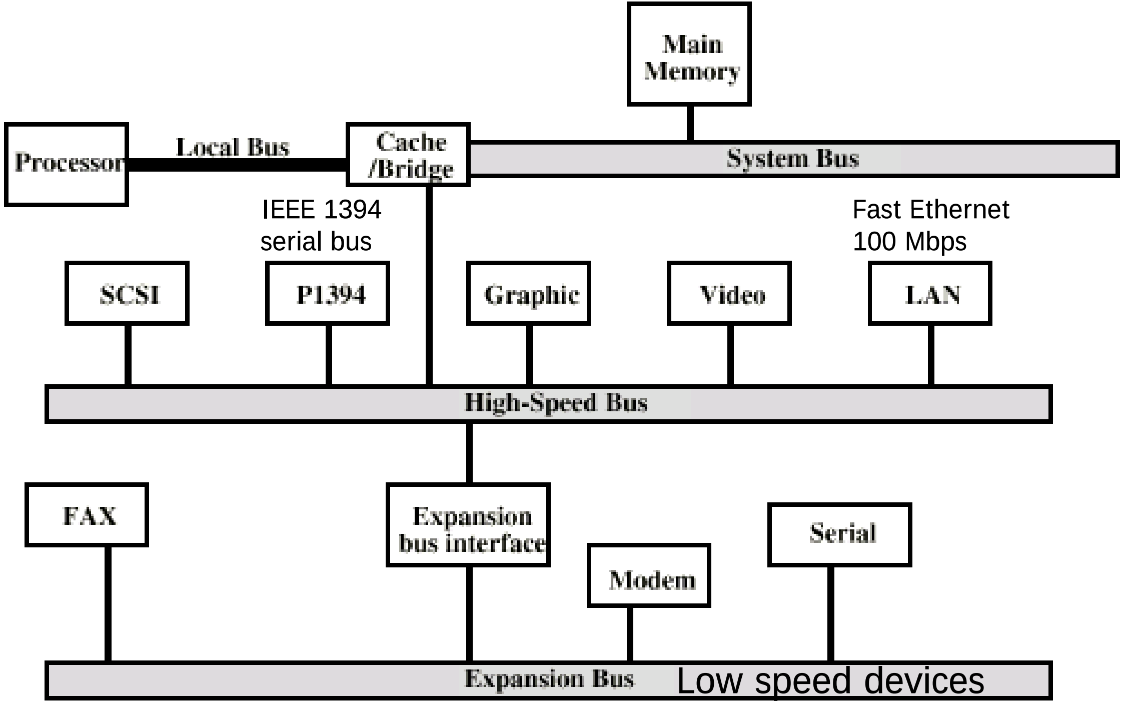
Bus performance suffers

- Two main causes:
  - more devices attached to the bus,
    - greater bus length and greater bus delay. Bus control passes from one device to another frequently and affects performance.
  - data transfer bottleneck
    - Aggregated data transfer demands capacity of the bus (32 to 64 bits)
    - However, data rates generated by the devices (NIC, graphics and video controllers) growing rapidly, creates bottleneck to the single bus system.

Solution: Bus Hierarchies

Multiple Buses Configuration

2) High speed bus (Mezzanine) architecture



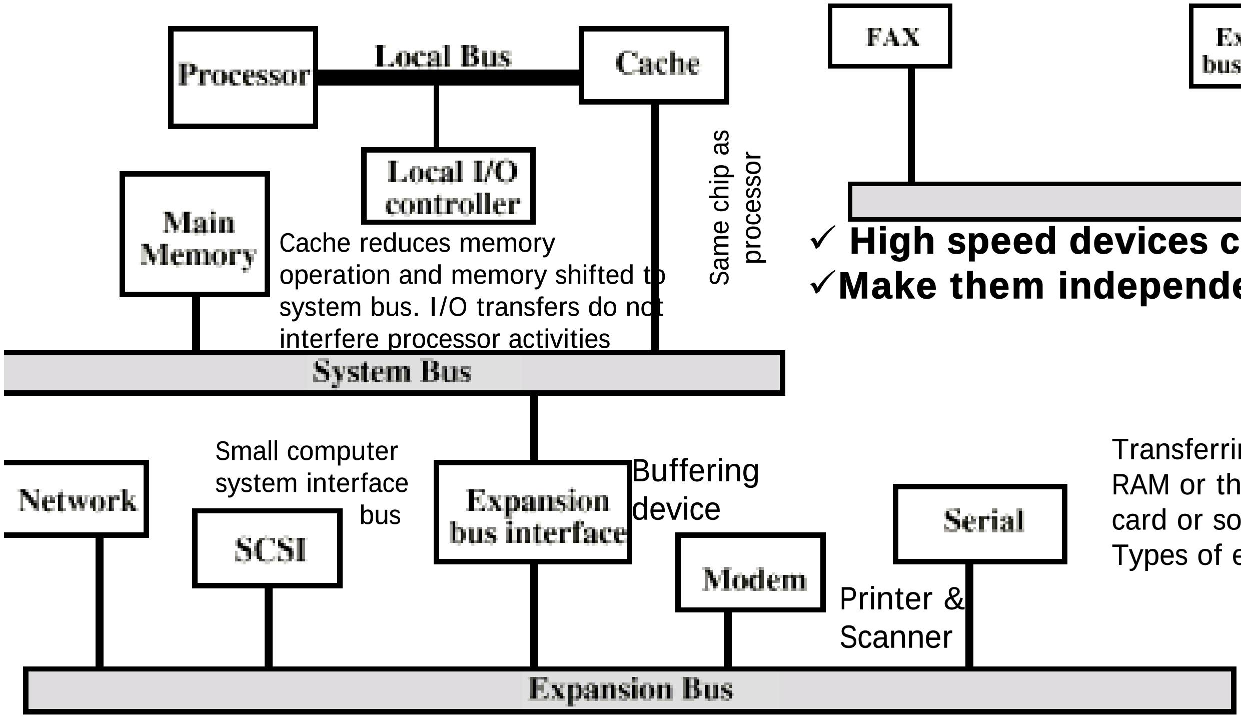
IEEE 1394 serial bus

Fast Ethernet 100 Mbps

Low speed devices

- ✓ **High speed devices closer integration with the processor**
- ✓**Make them independent from processor**

1) Typical bus architecture



Same chip as processor

Cache reduces memory operation and memory shifted to system bus. I/O transfers do not interfere processor activities

Small computer system interface bus

Buffering device

Printer & Scanner

Transferring information between internal hardware, such as RAM or the CPU, and expansion devices such as a graphics card or sound card.
Types of expansion buses with diverse features, such as:

Industry Standard Architecture (ISA)
Accelerated Graphics Port (AGP)
Peripheral Component Interconnect (PCI)
PCI Express (PCI-X)

Chapter 4 @Computer Organization and Architecture, William Stallings

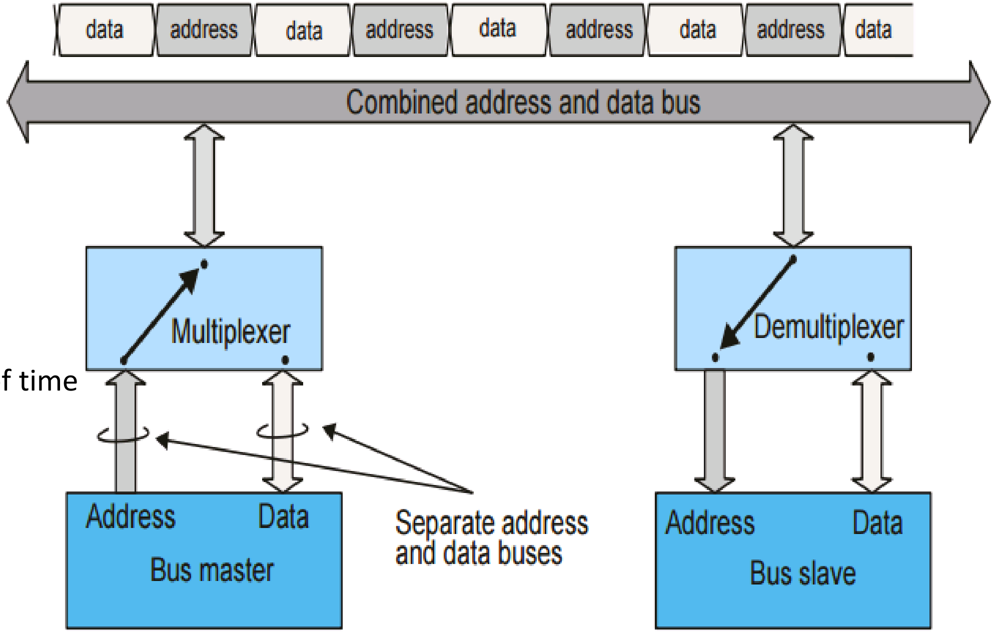
Element of Bus Design: Bus Width

- Data- width of the bus (32, 64, 128 separate lines)
  - determines overall performance of the system
  - e.g.: 32 bits wide data bus and 64 bits instruction needs twice memory accesses in a instruction cycle
- Address- width of the address line (8, 16, or 32 bits)
  - determines the maximum possible memory capacity.
  - Higher order bits are used to select the particular module
    - » Memory (module 0) **0**1111111
    - » I/O port (module 1) **1**0000001
- Control-transmits both command and timing information
  - timing signal determines the validity of data and address information
  - command signal specify operations to be performed.
  - memory write/read, I/O write/read, Bus req/grant, interrupt req/ack data transfer ack, clock, reset

Element of Bus Design: Type

- Dedicated- line is permanently assigned either to one function (data/address) or to a physical subset of components.
  - Adv: High throughput, less bus contention
  - Dis Adv Increase size and cost

- Multiplexed/shared-Address Valid or Data Valid control lines
  - Advantages
    - Few lines,
    - Save space and cost
  - Disadvantages
    - Time multiplexing
      - transfer needs to be done specific period of time
    - More complex circuitry
    - Performance reduction



Element of Bus Design: Arbitration

- **Centralized:**
  - an arbitration circuit (bus controller/ arbiter)
    - receives requests from the contending bus requesters/masters and then decides which of them is to be given control of the bus.
  - may be part of CPU or separate.
  - I/O module or CPU is assigned to memory bus.

- **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

Timing refers to the way in which events are coordinated on the bus.

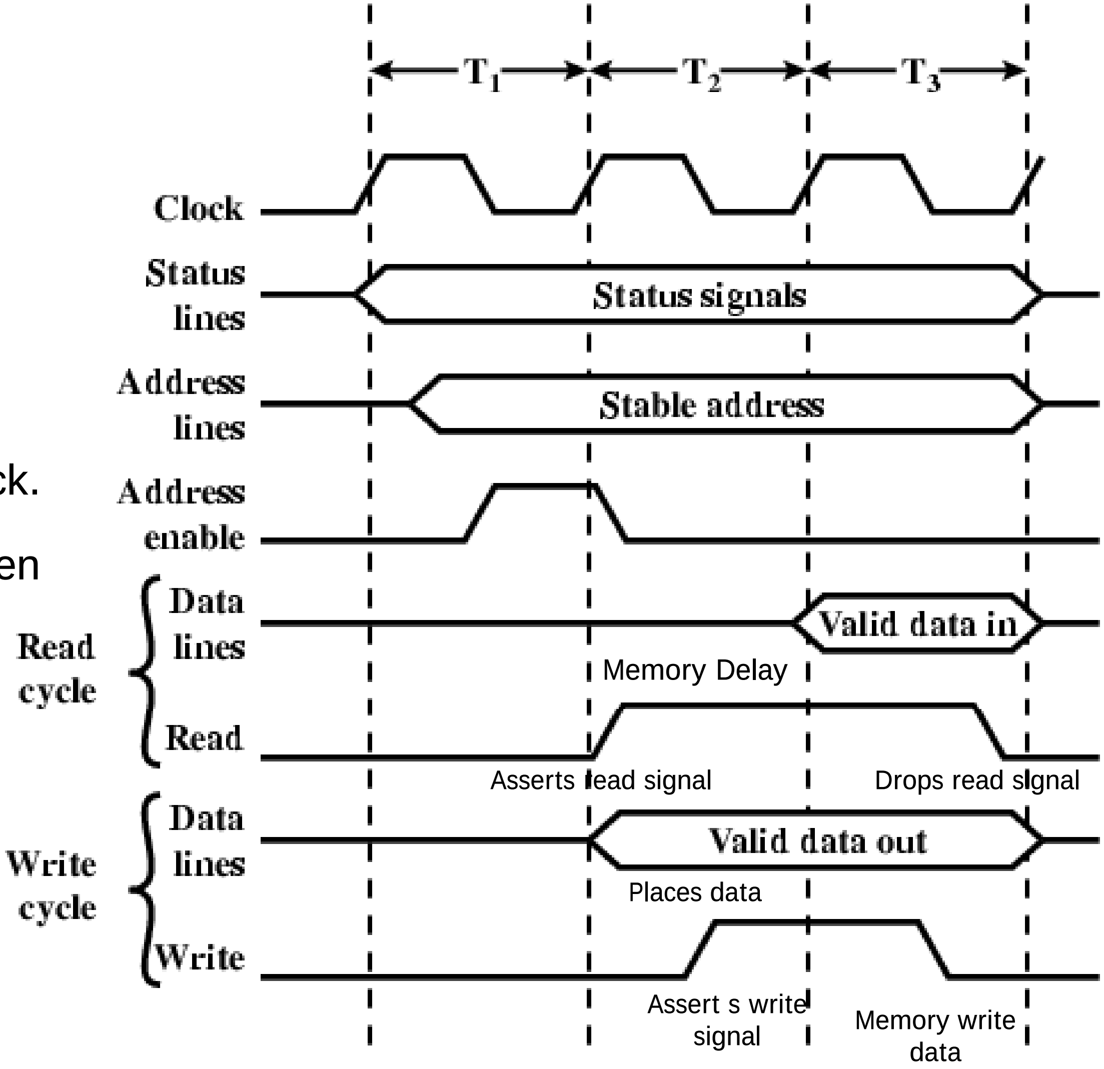
Synchronous Timing

The occurrence of events on the bus are determined by a clock.

Bus signal changes at leading edge of clock.

Synchronous read/write operations between

Processor and Memory modules

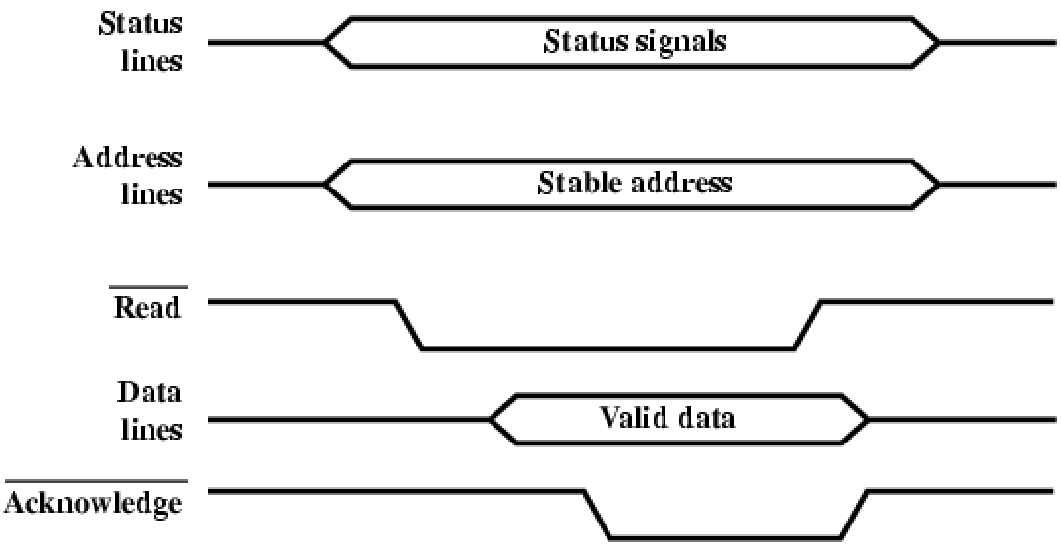


Element of Bus Design: Timing

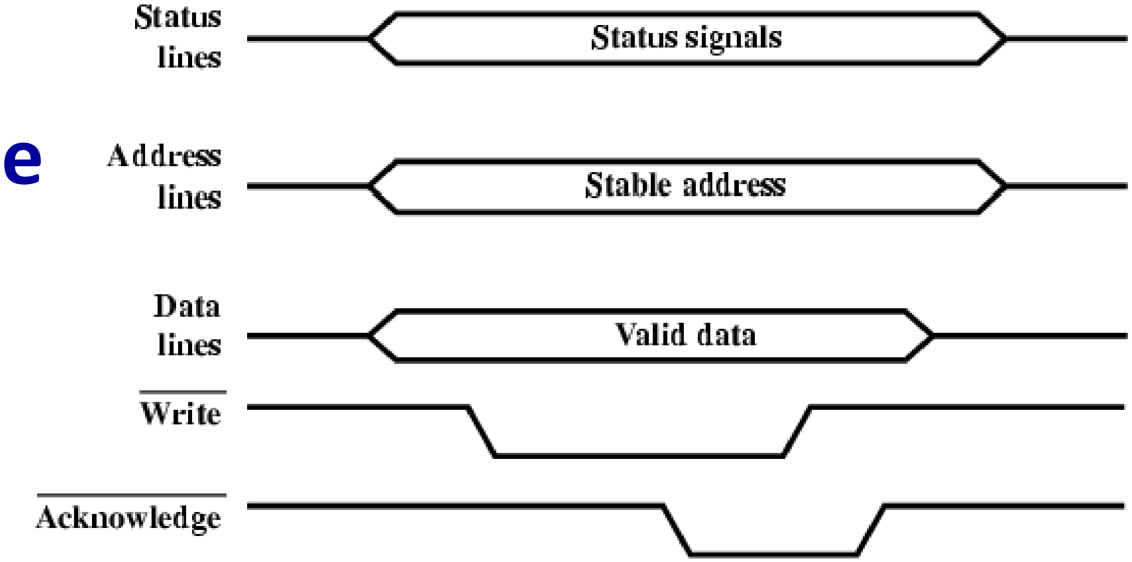
Asynchronous Read

- •Processor places address and status signal
- •Issues a read signal
- •Memory read address liens puts data in data lines
- •Sends acknowledge signal to processor

- •Processor read complete deasserts read signal
- •Memory then drop ack signal and remove data lines
- •Processor then remove address info



Asynchronous Write



- •Processor drops write signal
- •Memory drops ack signal

Chapter 4 @Computer Organization and Architecture, William Stallings

Point-to-Point Interconnection

- Shared bus to point-to-point technology
  - Increasing data rates makes difficult to perform the **synchronization and arbitration functions** in a timely fashion.
  - Adjustment (increasing rate & reducing latency) of data rates with multiprocessor, multicore, memory in chip components.
  - Point-to-point provides
    - Higher data rate
    - Lower latency
    - Better scalability

QuickPath Interconnection (QPI)

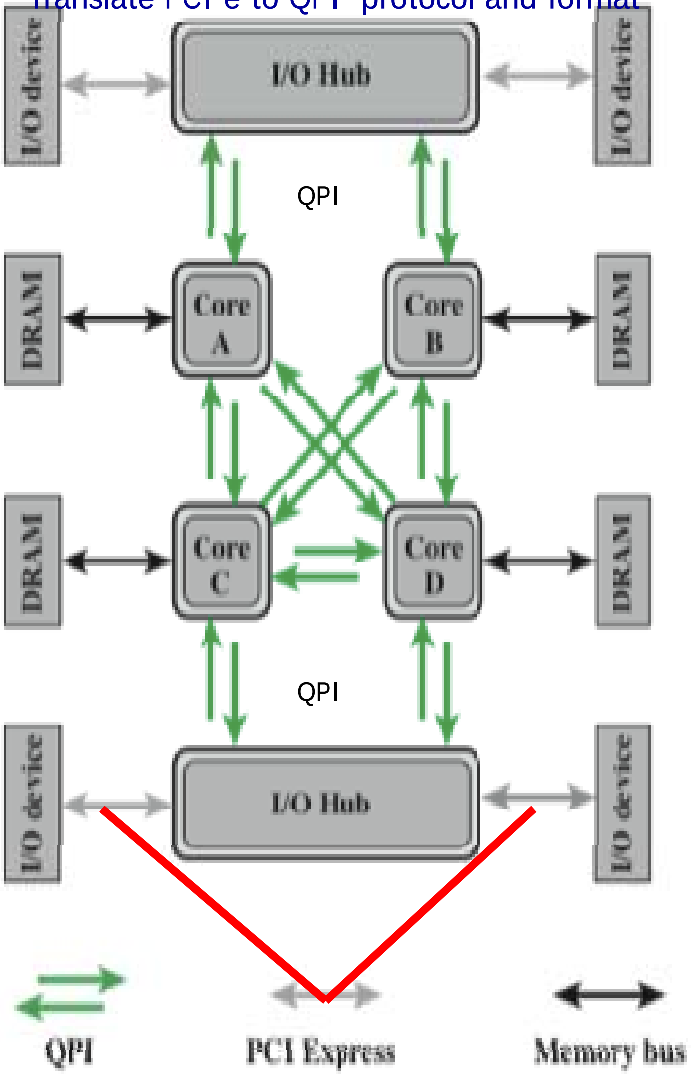
Intel 2008

- Characteristics of QPI:
  - Multiple direct connections
    - direct pair wise connections between components
    - removes the requirement of arbitration
  - Layered protocol architecture
    - Rather than use of simple control signals, processor level interconnections **use layered protocol architecture** (TCP/IP based data network)
  - Packetized data transfer
    - Data are not send as raw bit stream
    - Data sent as a sequence of packets, includes control headers and error control codes

QuickPath Interconnection (QPI)

Work as switch

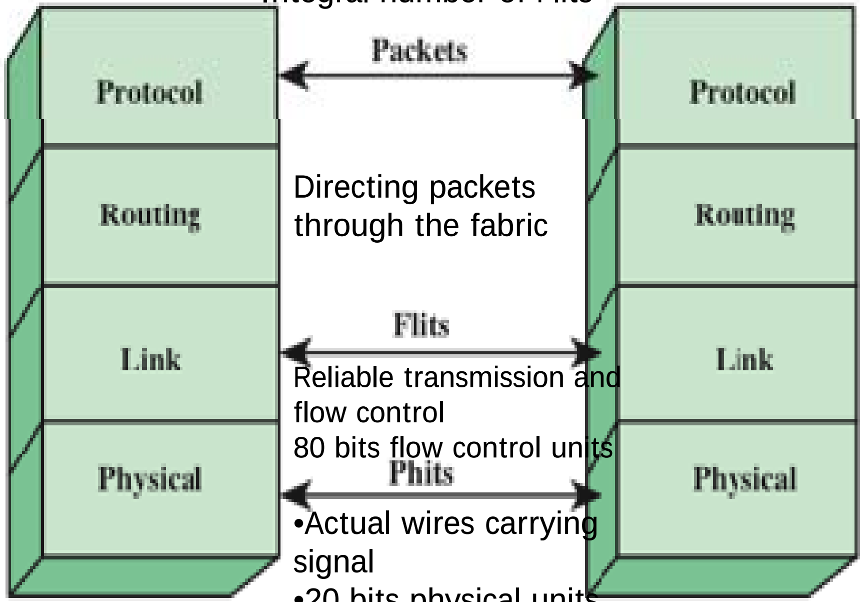
Translate PCI e to QPI protocol and format



QPI

QPI

Rules of exchanging packets
Integral number of Flits



Directing packets through the fabric

Reliable transmission and flow control
80 bits flow control units

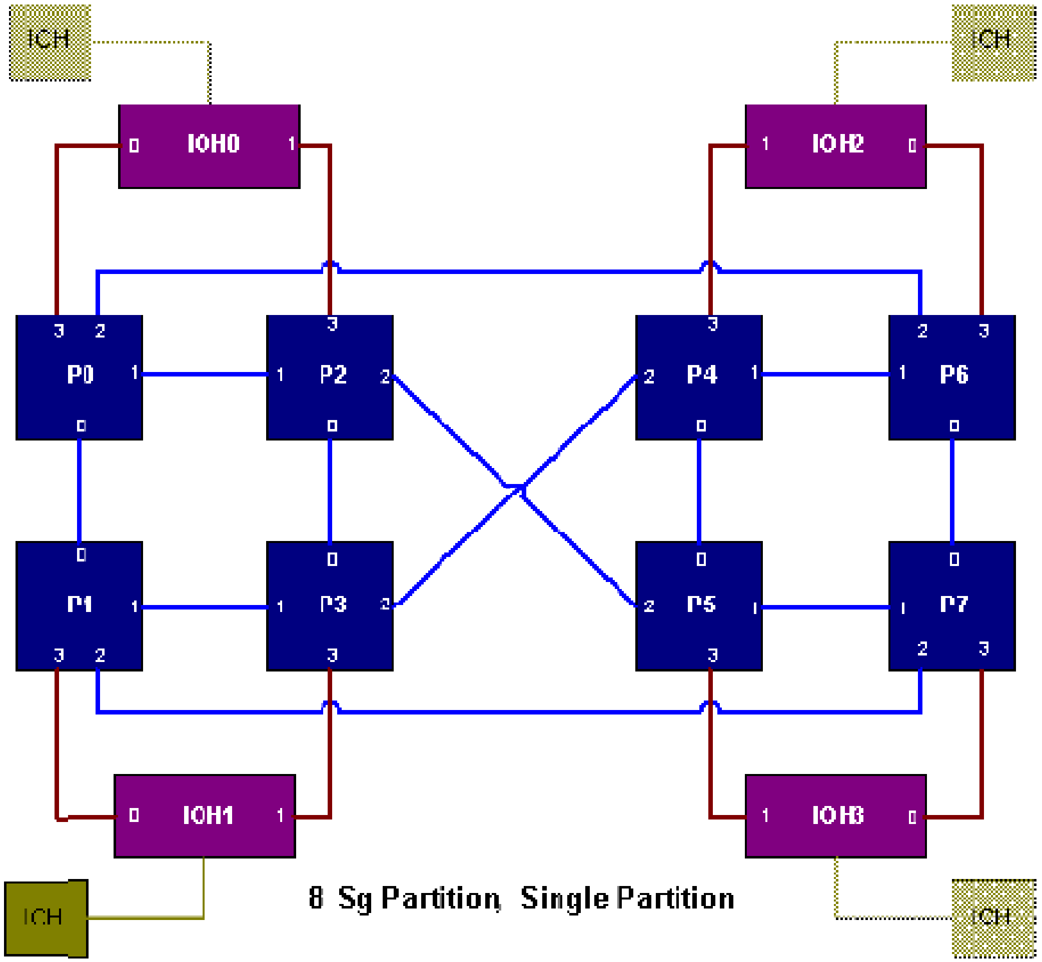
- Actual wires carrying signal
- 20 bits physical units

QPI provides

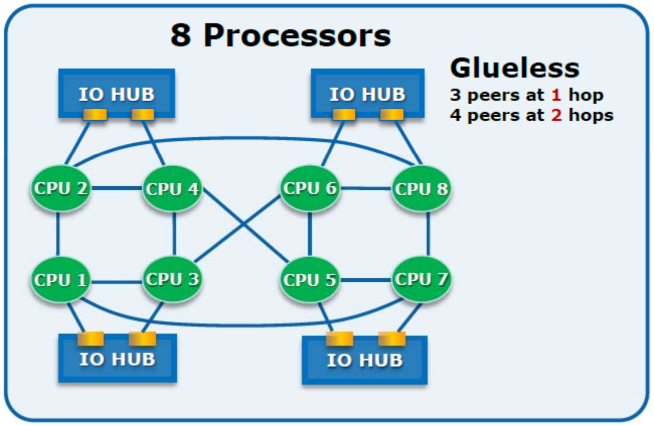
-point to point interconnection
-Socket to socket connections
-Socket to chipset connection
-Build scalable connection
-6.4 GT/s transfers 20 bits /T
-Up to 16 GB/sec
- Bidirectional 32 GB/sec

FOR Large # of cores
Three links…
Route traffic through the intermediate processors

Chapter 4 @Computer Organization and Architecture, William Stallings



8 Sg Partition, Single Partition


8 Processors
Glueless
3 peers at 1 hop
4 peers at 2 hops
IO HUB
IO HUB
CPU 2
CPU 4
CPU 6
CPU 8
CPU 1
CPU 3
CPU 5
CPU 7
IO HUB
IO HUB