Computer Organization and Assembly Language

Introduction

About Me

Aleena Ahmad

Department of Computer Science

FAST-NU- LHR

aleena.ahmad@nu.edu.pk

Office Location: Exam Hall (office # 119)

What is the course about?

- This course is about looking at lower level details of how computers works.
- In order to do so, we will see computer organization and architecture.
- You will also learn assembly language for Intel 8088 processor.
- At this point you have taken a course on High level language (C++), so its good time to get started looking at
  - How processor (CPU) is organized?
  - How program is executed on processor?
  - How processor interacts with memory and I/O devices?

Books

- Assembly Language Programming Lecture Notes by Bilal Hashmi (BH).
- Assembly Language for x86 Processors Seventh Edition Kip R. Irvine (KI)
- Computer Organization and Architecture Designing for Performance Tenth Edition by William Stallings (WS)

Tentative Evaluation and Grading Policy

- You will be evaluated on quizzes, 2 mid term exams, 1 final exam and multiple assignments and a project.
- Tentative Percentage Grade Distribution:

  - QUIZZES 10
  - MIDTERMS 30
  - FINAL 45
  - ASSIGNMENTS 5
  - Project 10

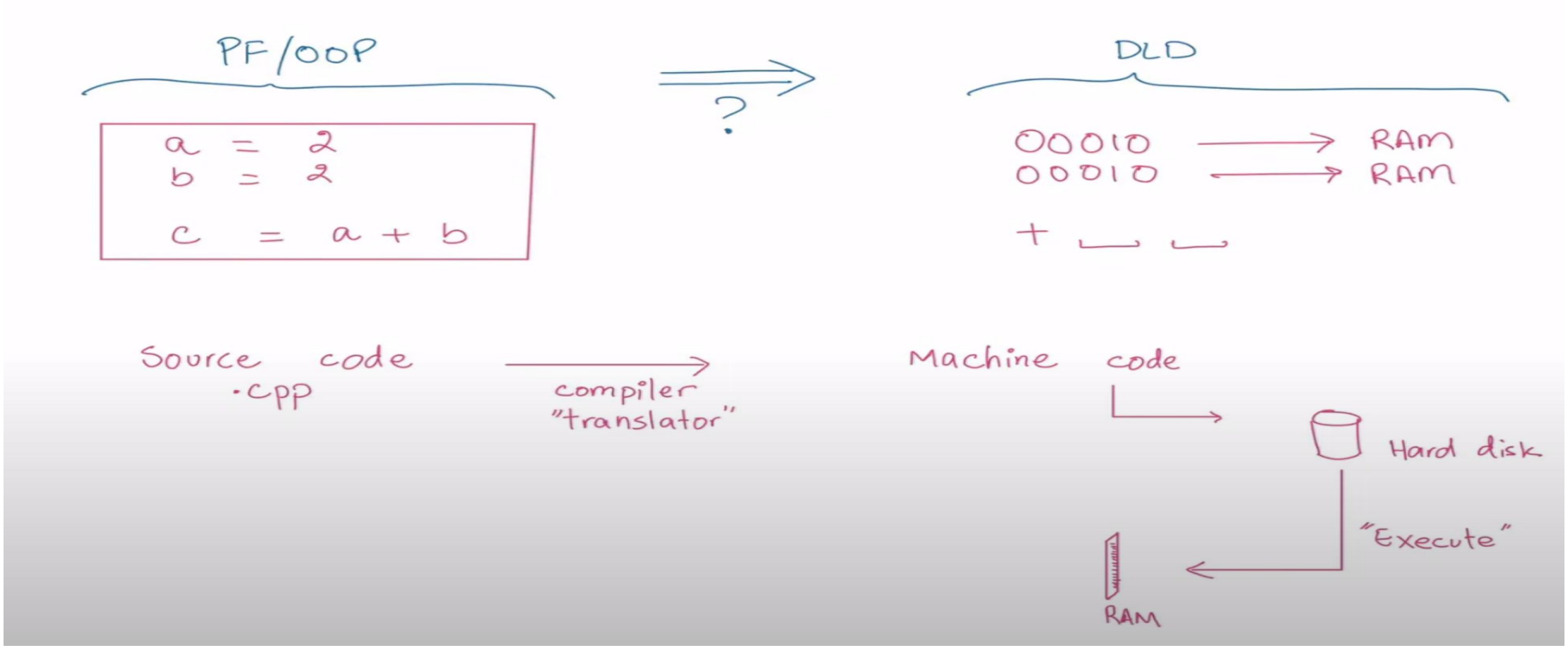
Academic Integrity

- Plagiarism and Cheating against academic integrity. Both parties involved in such cases will face strict penalty (negative marking, F grade, DC)
- CODE/ ASSIGNMENT SHARING is strictly prohibited.
- Keep in mind that by sharing your code/assignment you are not helping anyone rather hindering the learning process or the other person.
- No excuse will be entertained if your work is stolen or lost. To avoid such incidents
  - Keep back up of your code on safe online storage, such as Google Drive, Drop box or One drive.
  - Do not leave your work on university lab computer, transfer your work to online storage and delete from the university lab computer (empty recycle bin as well)

Question

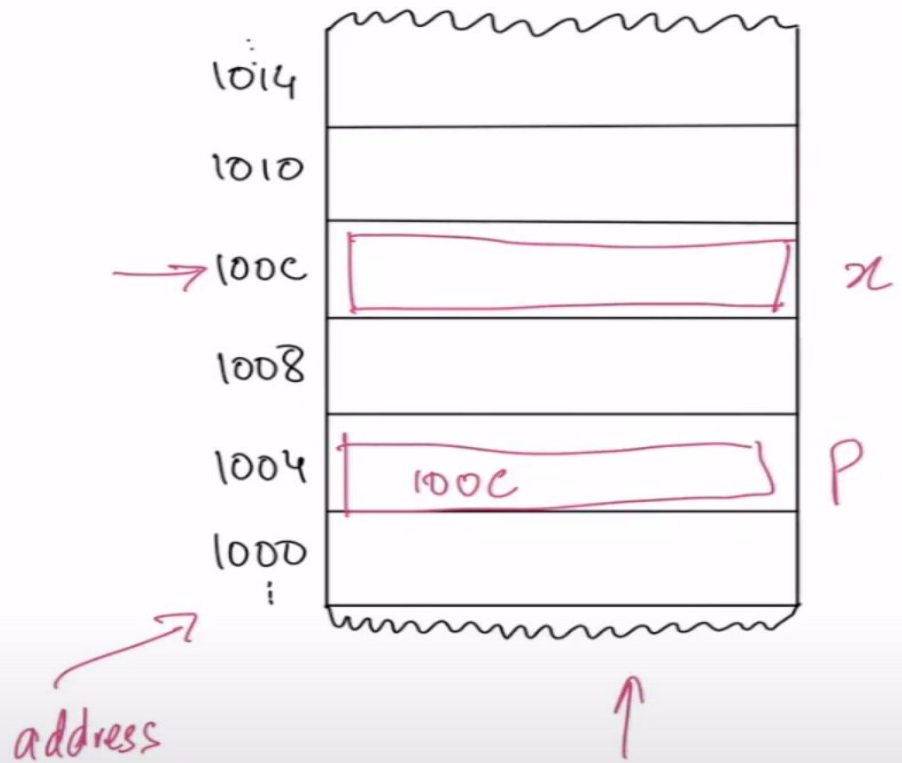
- What specifications do you look at when you see a computer (lets say a laptop)?

High level to Low Level



Example

```
int   x ;
int   * p ;
x =   25
p =   & x ;
```

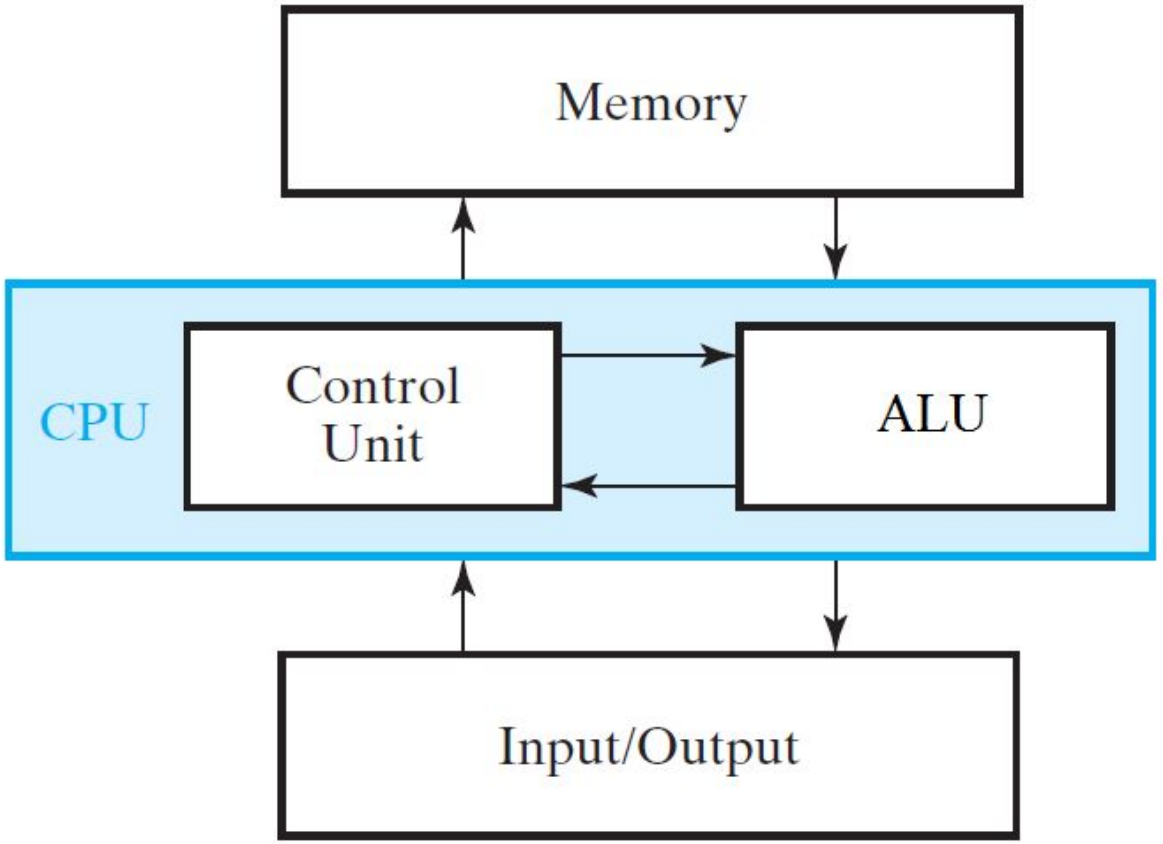


Memory Address Structure

Components of Digital Computer

Computer uses these components in order to provide different functionalities (programs):

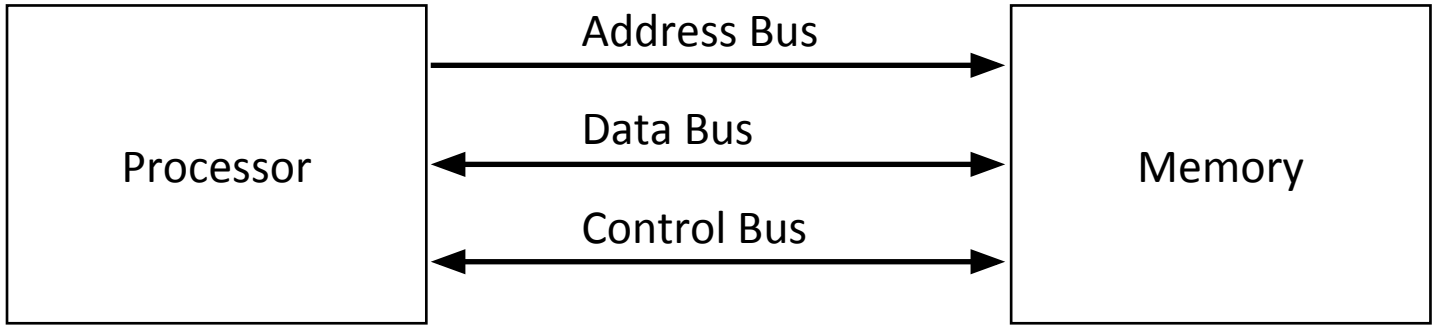
- **Memory** stores program and data (input, output and intermediate data)
- **Central Processing Unit (CPU)**
  - -Performs Operations
  - **-Arithmetic Logic Unit (ALU)** performs arithmetic and data-processing operations as specified by program
  - **Control Unit** supervises the flow of information between the various units
  - ALU and Control Unit together form CPU
- **Input Devices** such as Mouse, Keyboard, DVD drives, CD-ROM, Microphones, Scanner, Webcam etc. are used to take data input in programs
- **Output Devices** such as LCD, Printer and Loudspeakers etc. are used to present result to the user



Tasks need to be performed

- There must be a mechanism to inform memory that we want to do the read operation
- There must be a mechanism to inform memory that we want to read precisely which element
- There must be a mechanism to transfer that data element from memory to processor

Buses



Buses

- A bus is a communication pathway connecting two or more devices. For example Processor to/from Memory, I/O to/from processor, I/O to/from Memory
- The group of bits that the processor uses to inform the memory about which element to read or write is collectively known as the address bus.
- Another important bus called the data bus is used to move the data from the memory to the processor in a read operation and from the processor to the memory in a write operation.
- The third group consists of miscellaneous independent lines used for control purposes. For example, one line of the bus is used to inform the memory about whether to do the read operation or the write operation. These lines are collectively known as the control bus

Adress Bus

- When a processor needs to read or write to a memory location, it specifies that memory location on the address bus (the value to be read or written is sent on the data bus).
- **Unidirectional** – Address always travels from processor to memory.
- An address bus is a computer bus (a series of lines/wires connecting two or more devices) that is **used to specify a physical address**.
- The number of address bus lines varies from one microcomputer to another.
- The width of the address bus determines the amount of memory a system can address.

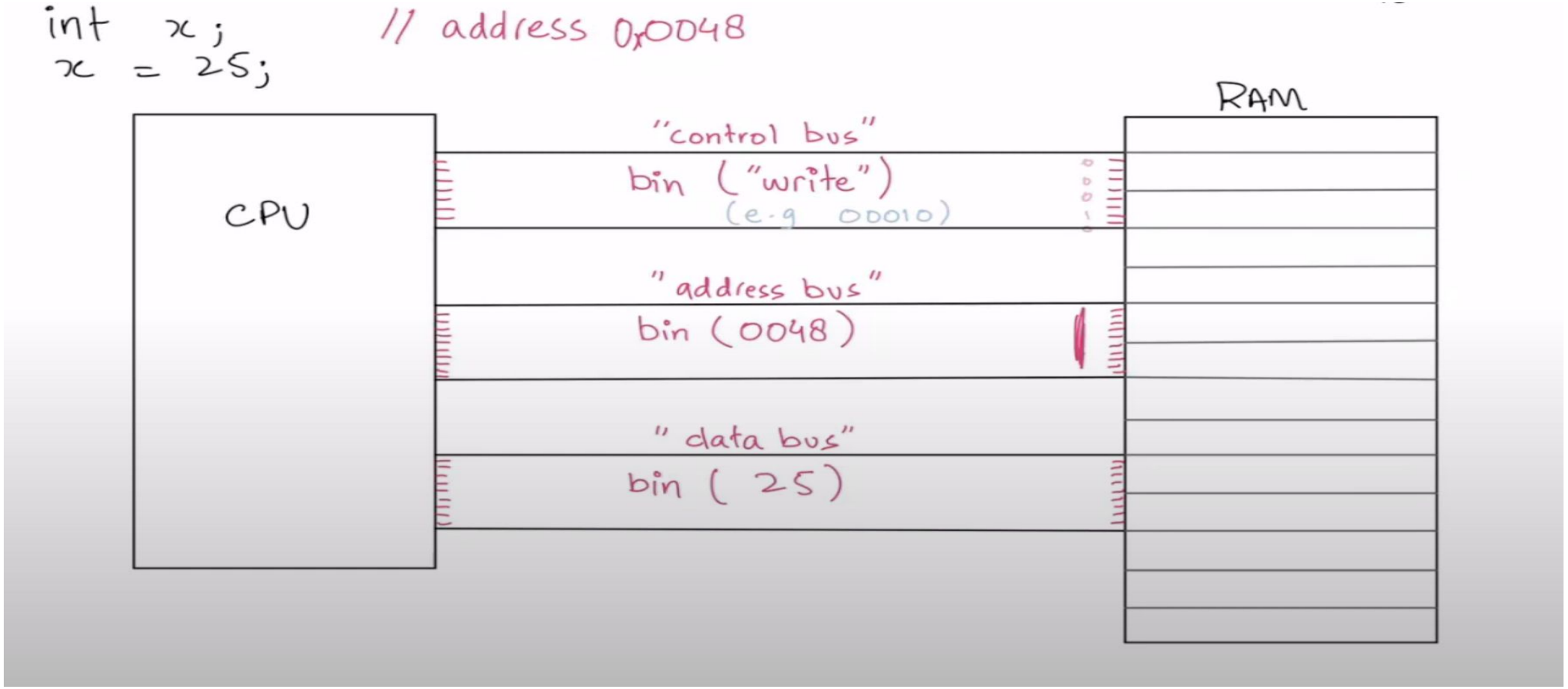
Data Bus

- The data bus is the **set of wires over which data passes between the CPU and the memory and I/0**.
- **Bidirectional**
- The data can either be instructions for the CPU, or information the CPU is passing to or from I/0 ports.

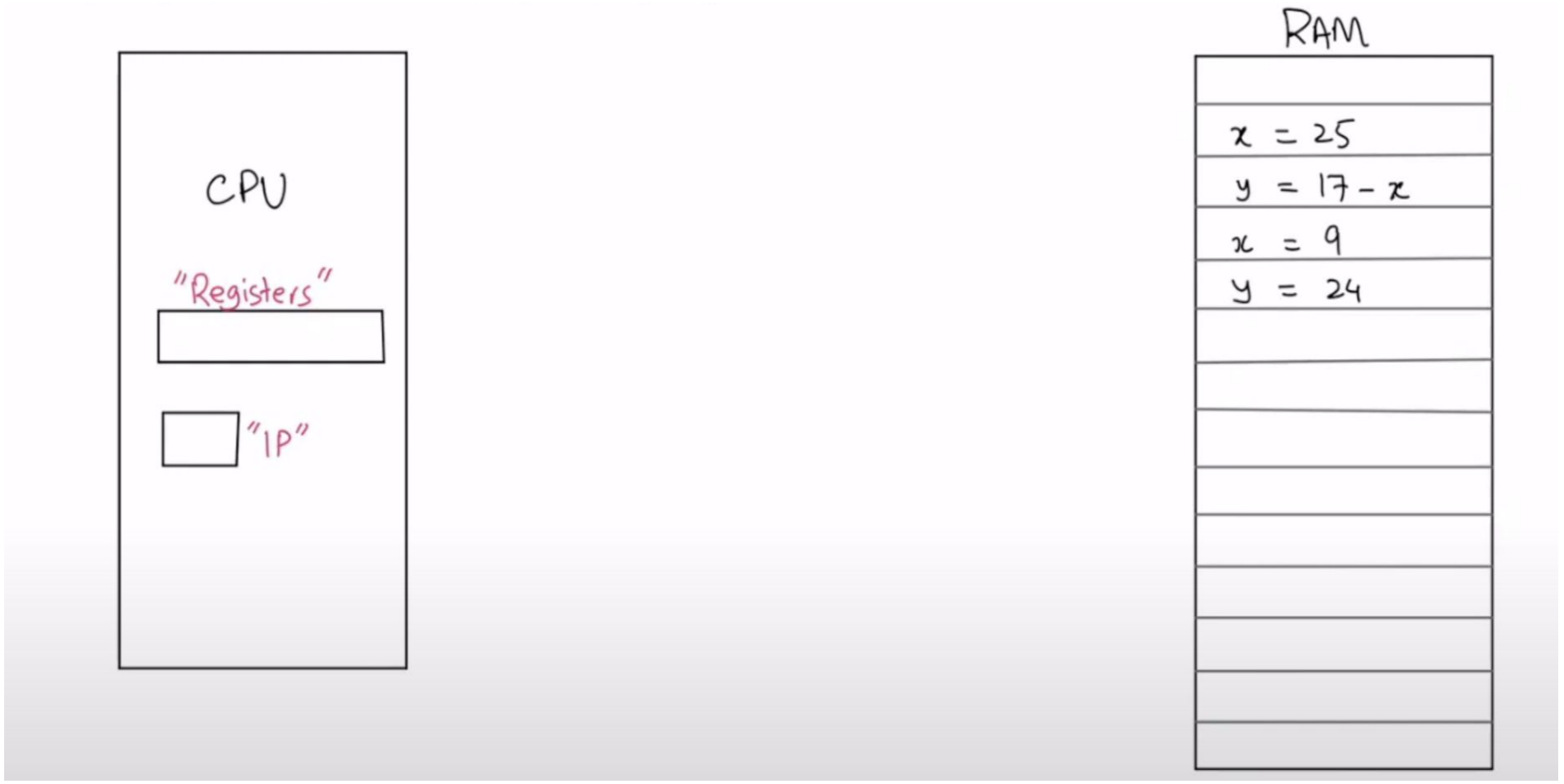
Control Bus

- Control bus carries commands from the CPU and returns status signals from the devices.
- For example, if the data is being read or written to the device the appropriate line (read or write) will be active (logic one).

Example


int x;
x = 25;
// address 0x0048
RAM
CPU
"control bus"
bin ("write")
(e.g 00010)
"address bus"
bin (0048)
"data bus"
bin (25)

Question



Problems

- RAM is slow
- CPU needs to keep track of which instruction it is executing

How instructions are executed

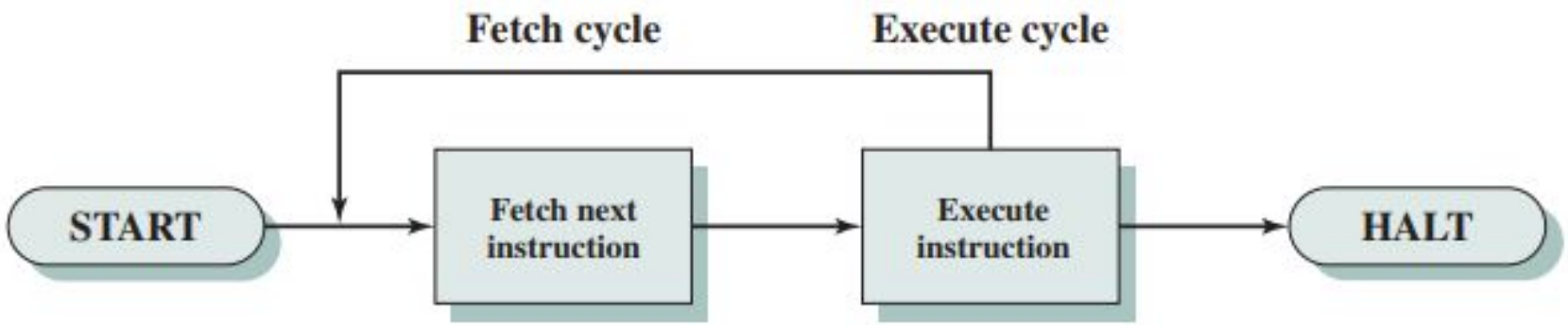


Figure 3.3 Basic Instruction Cycle

A word about registers

- Registers are named storage locations in the CPU that hold intermediate results of operations.
- As they are located inside CPU they are fastest to access.
- Each processor has limited number of registers.

Instruction Group

- Data Movement Instructions
  - These instructions are used to move data from one place to another.
  - These places can be registers, memory, or even inside peripheral devices. Some examples are:

```
mov ax, bx
lad 1234
```

- Arithmetic and Logic Instructions
  - Arithmetic instructions like addition, subtraction, multiplication, division and Logical instructions like logical and, logical or, logical xor, or complement are part of this group.
  - Some examples are:

```
and ax, 1234
add bx, 0534
add bx, [1200]
```

Instruction Group

- Program Control Instructions
  - These are instructions that control the program execution and flow by playing with the instruction pointer and altering its normal behavior to point to the next instruction.
  - Some examples are:

```
cmp ax, 0
jne 1234
```

- Special Instructions
  - Another group called special instructions works like the special service commandos. They allow changing specific processor behaviors and are used to play with it. They are used rarely but are certainly used in any meaningful program. Some examples are:

```
cli
sti
```

  - Where cli clears the interrupt flag and sti sets it

Intel IAPX88

- Intel Advanced Processor Extension 88 (8088)
- 16-bit processor + 1 MB Memory
- Accumulator and all registers of 16 bits