Computer Architecture and Organization

COMPUTER FUNCTION

Lecture 3

Dr.Shada Mabgar

COMPUTER FUNCTIONS

- The basic function performed by a computer is execution of a program, which consists of a set of instructions stored in memory .
- The processor does the actual work by executing instructions specified in the program .

COMPUTER FUNCTION

- Instruction processing consists of two steps :

1. The processor reads **(fetches)** instructions from memory .
2. **Executes** each instruction.

- Program execution consists of repeating the process of instruction fetch and instruction execution.

COMPUTER FUNCTION

- The processing required for a single instruction is called an **instruction cycle.**
- Each instruction cycle consists of the following phases:

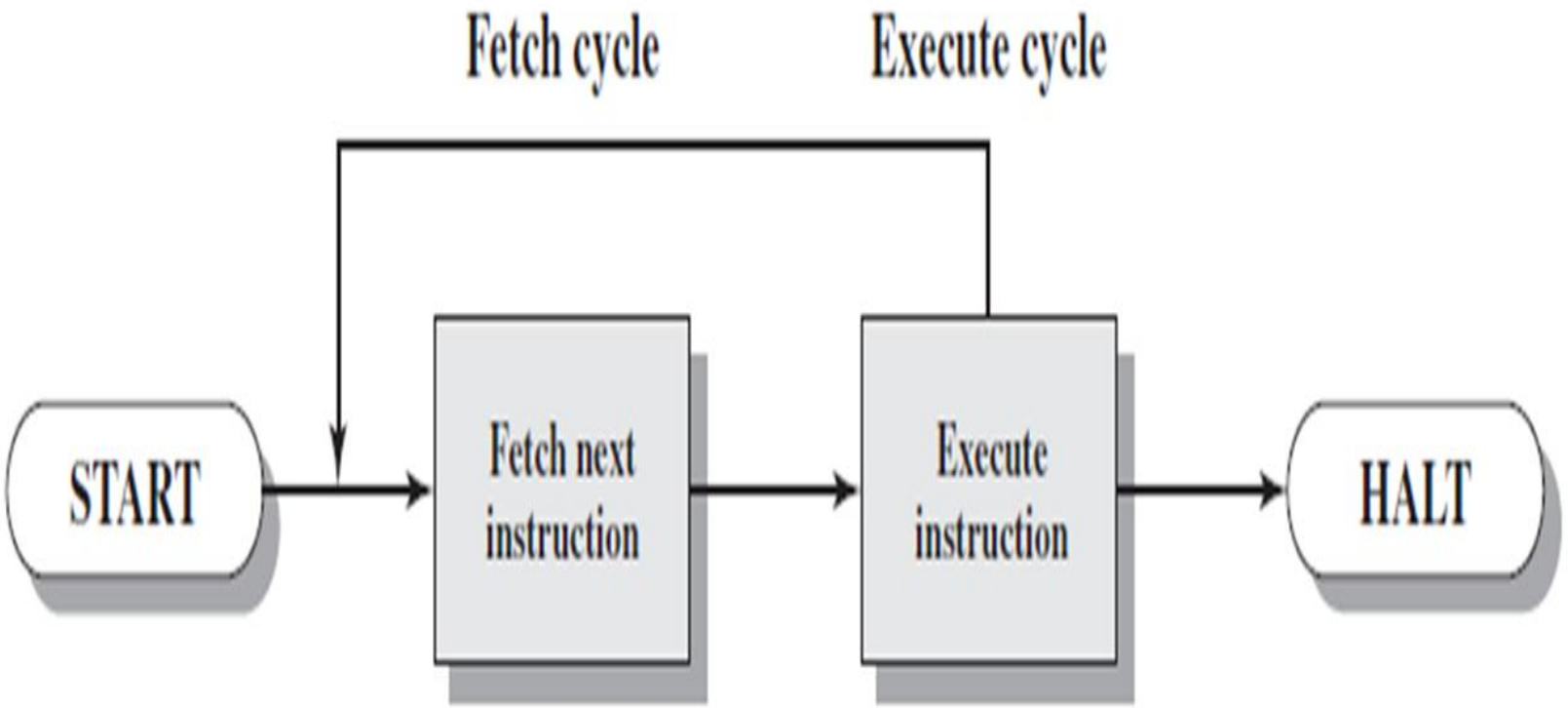
  1- Fetch instruction from memory.

  2- Decode the instruction.

  3- Read the effective address from memory.

  4- Execute the instruction.
- Using the simplified two-steps are referred to as the **fetch cycle and the execute cycle .**
- **Program execution** halts only if the machine is turned off, some sort of unrecoverable error occurs, or a program instruction that halts the computer is encountered.

COMPUTER FUNCTION



Basic Instruction Cycle

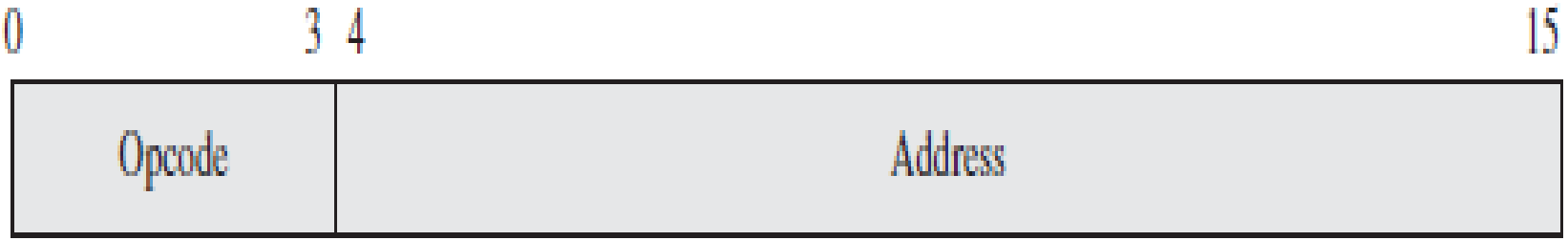
Instruction Fetch and Execute

- At the beginning of each instruction cycle, the processor fetches an instruction from memory .
- A register called the **program counter (PC)** holds the address of the instruction to be fetched next.
- The processor always **increments** the **PC** after each instruction fetch so that it will fetch the next instruction in sequence .
- The fetched instruction is loaded into a register in the processor known as the **instruction register (IR).**
- The instruction contains bits that specify the action the processor is to take.
- The processor interprets the instruction and performs the required action. In general, these actions fall into **four categories :**

Instruction Categories

- **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.

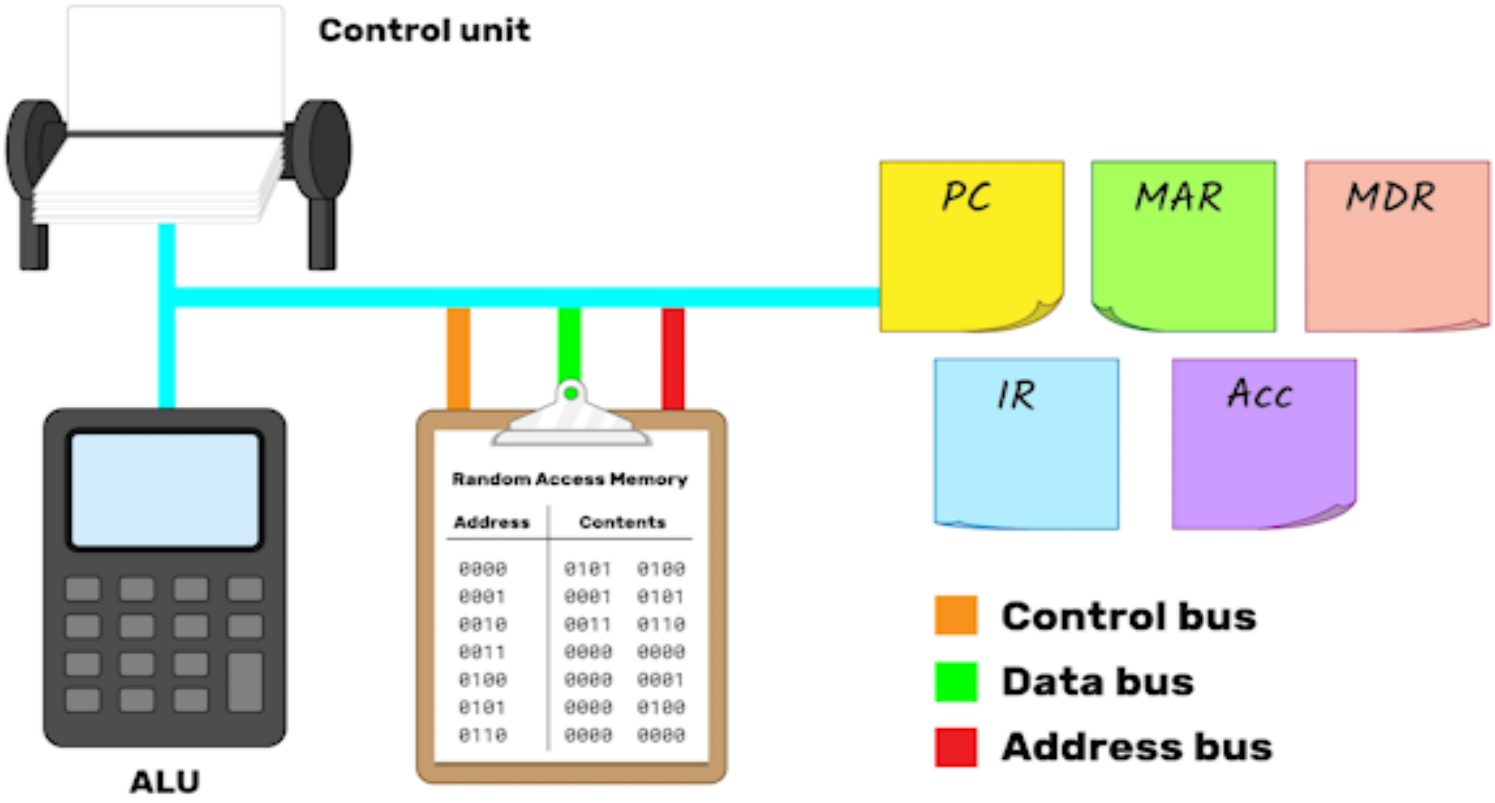
Instruction Format



- **Opcode**-specifies operation(add ,subtract,divide)
- **Address -**contain one or more operands, which indicate where in registers or memory the data required for the operation is located .

Fetch-Decode-Execute Cycle Example

This example to look at how the CPU can perform calculation ,using a process known as the fetch – decode – execute cycle .

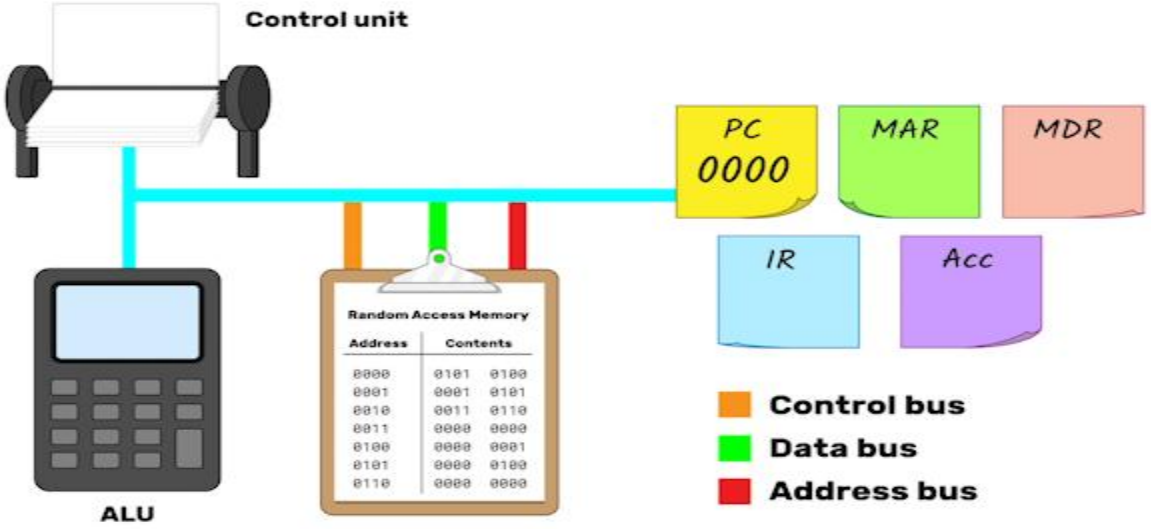


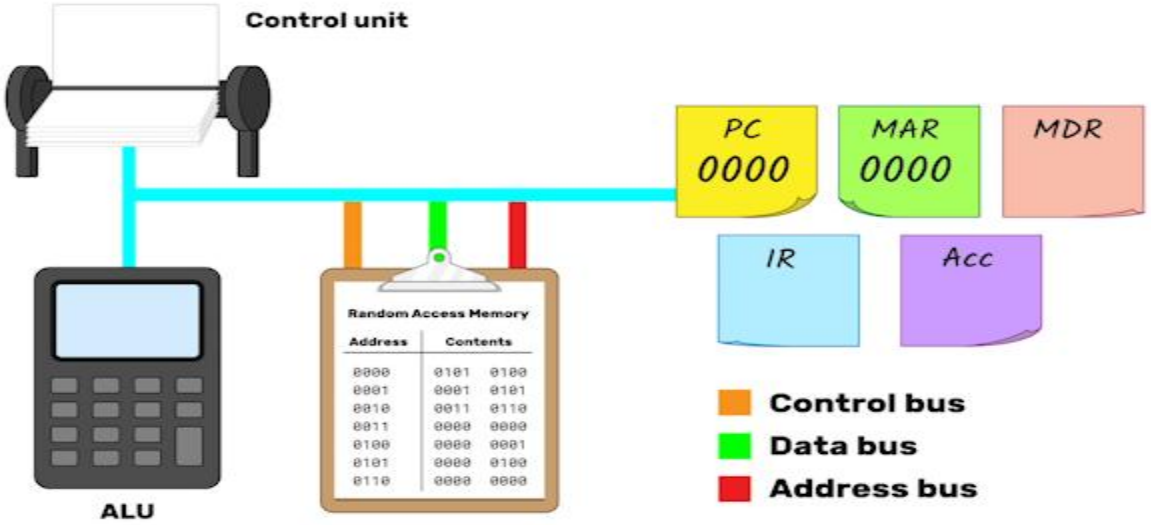
Fetch-Decode-Execute Cycle Example

Fetch

- The **program counter (PC)** starts at **0000**.
- This means that the **first** address in **RAM** where the computer will look for an instruction is **0000.**
- The computer needs somewhere to store the current address in RAM that it is looking for.
- This is what the memory address register **(MAR)** is for.
- The address **0000** is therefore copied into the **MAR.**

Fetch-Decode-Execute Cycle Example

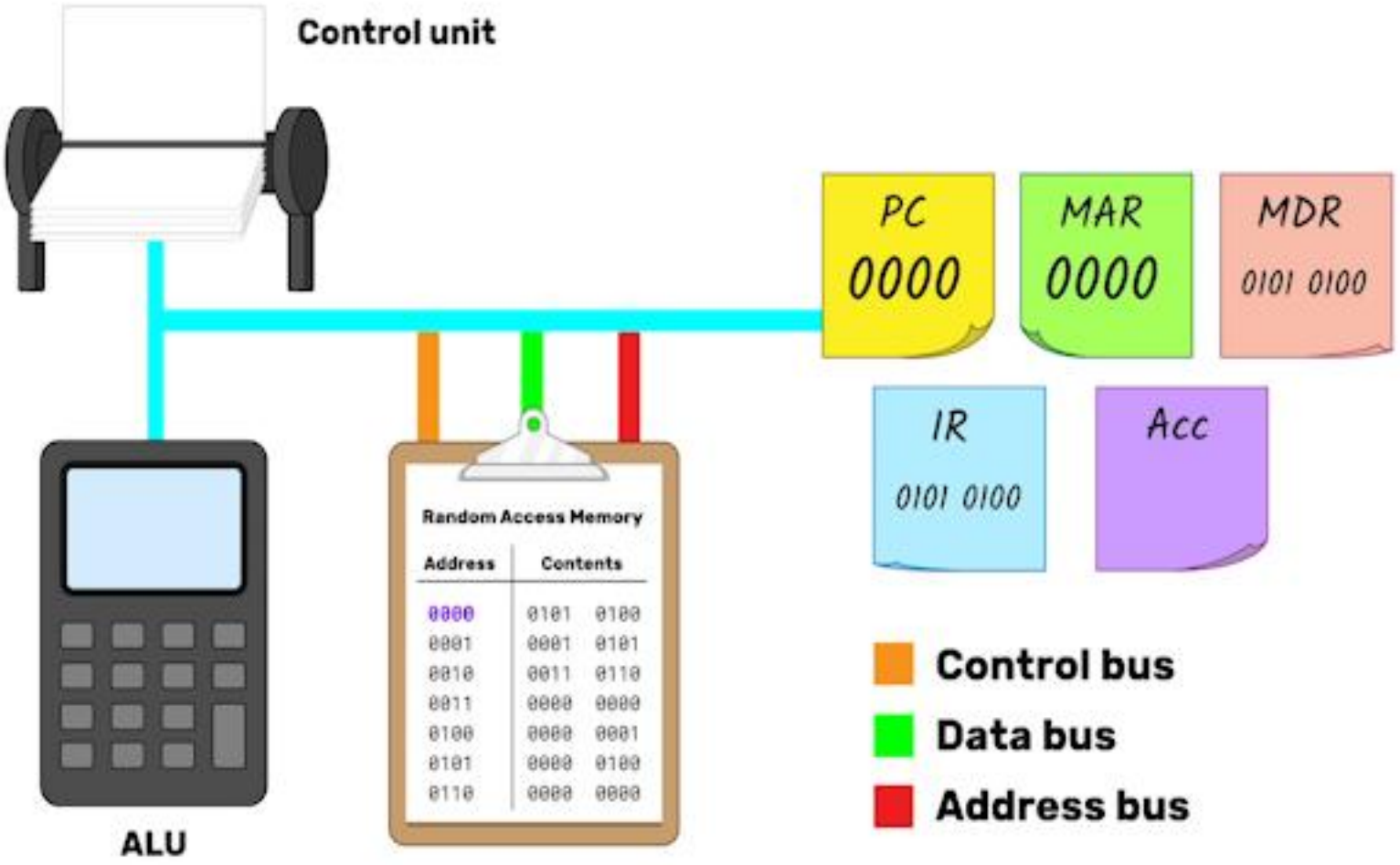




Fetch-Decode-Execute Cycle
Example

- A signal is now sent down through the **address bus** to the RAM.
- The control unit sends out a memory **read signal**, and the contents of the address **0000** are copied through the **data bus** to the memory data register **(MDR).**
- As the data fetched during the fetch stage is an instruction, it is copied into the instruction register **(IR).**
- As the first instruction has been fetched, the system is at the end of the **fetch stage** of the cycle.
- The **program counter** can be incremented by **1**, so the system is ready to read the next instruction when the next fetch cycle starts.

Fetch-Decode-Execute Cycle Example



Fetch-Decode-Execute Cycle Example

Decode

- The instruction needs to be **decoded.**
- It is sent via the data bus to the **control unit**, where it is split into two parts. The **first part** is the operation code or **opcode**, which in this example CPU is the first **four** bits.
- This is the command that the computer will carry out.
- The **second part**, in this case the second four bits, is the operand.
- This is an address in RAM where data will be read from or written to, depending on the operation.
- The **control unit** can translate opcodes into instructions.
- So here the control unit translates the opcode **0101** into a **LOAD FROM RAM** instruction.

Fetch-Decode-Execute Cycle
Example

Execute

- The **operand** is copied to the **MAR**, as it provides the address of the data to be loaded (0100 in this case).
- The data at address **0100** is then fetched from **RAM** and passed up the **data bus** to the **MDR.**
- As it is **not** an instruction but simply **data**, it is then passed to the **accumulator (Acc).**
- **This is a complete fetch-decode-execute cycle.**

Fetch- Decode and Execute Cycle Example

The 2nd Fetch, Decode and Execute Cycle : The **PC** now holds **0001**, so you fetch, decode, and execute the instruction at that address.

1- The PC is at 0001, so this is the next instruction to be fetched.

2- The instruction opcode and address operand are placed in the IR, and the PC increased by 1 again.

3- The instruction is decoded, while the address of the data to be acted upon is placed in the MAR. The instruction turns out to be ADD, which adds two pieces of data together.

4- The new data is fetched from the address and eventually ends up in the accumulator, along with the results of the previous cycle.

To finish off this stage, the two values in the accumulator are passed into the ALU, where they can be added together, as was instructed by the opcode.

The result is then placed back into the accumulator. **That's a second cycle complete**

Fetch- Decode and Execute Cycle
Example

The 3rd Fetch, Decode and Execute Cycle : The last cycle is for the instruction at **0010**. It uses the opcode **0011**, which is **STORE** and the operand **0110**, which is the last address in the RAM shown.

This cycle takes the results of the addition in the accumulator and stores it back into RAM at address 0110, as requested.

Fetch- Decode and Execute Cycle Example

- To recap, our program used three instructions to add two numbers and store the result in memory:
- The first instruction LOADed a piece of data from a specified address.
- The second ADDed this to the data found in another address.
- The final instruction STOREd the result of the addition back into a specified address in memory.

Fetch- Decode and Execute Cycle

- When a Process is executed by the CPU and when a user Request for another Process then this will create disturbance for the Running Process.
- This is also called as the **Interrupt.**

Types of Interrupts

- Generally there are three types of **Interrupts** :

1- Internal Interrupt.

2- Software Interrupt.

3- External Interrupt.

- The **Internal Interrupts** are those which are occurred due to Some Problem in the Execution For Example When a user performing any Operation which contains any Error and which contains any type of Error.

Types of Interrupts

- The **External Interrupt** occurs when any Input and Output Device request for any Operation and the CPU will Execute that instructions first For Example When a Program is executed and when we move the Mouse on the Screen then the CPU will handle this External interrupt first and after that he will resume with his Operation .
- The **Software Interrupts** are those which are made some call to the System for Example while we are Processing Some Instructions and when we wants to Execute one more Application Programs

Classification of Instructions

- **Data movement instructions** are used to move data between registers and between memory and registers .
  - **MOVE Ri , Rj**
- **Arithmetic/logical instructions**: Arithmetic and logical instructions are those used to perform arithmetic and logical manipulation of registers and memory contents.
  - **ADD R1,R2,R0**
  - **SUBTRACT R1,R2,R0**

Classification of Instructions

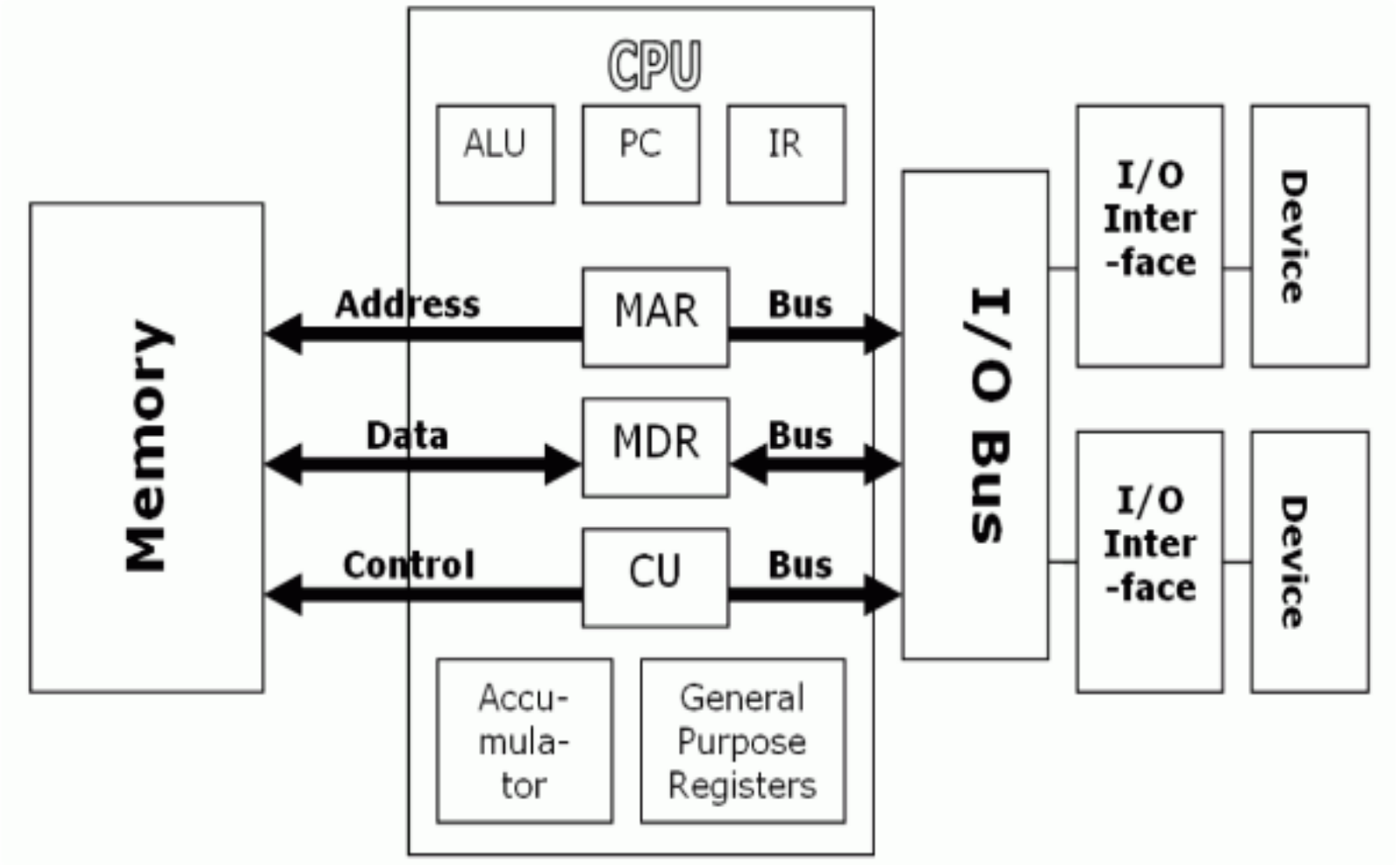
- **Sequencing Instructions :**Control (sequencing) instructions are used to change the sequence in which instructions are executed.

  - Jump Label (Jump to the subroutine which starts at Label).

- **Input / Output Instructions :** are used to transfer data between the computer and peripheral devices .

  -Input R5, PORT (Read from i/o port “PORT” to registerR5).

Computer Components :Top - Level View



Summary

Step	Fetch execute cycle steps	Simplified description
1	The PC contains the address of the memory location that has the next instruction which has to be fetched	PC has address of next instruction
2	This address is then copied from the PC to the MAR via the address bus	PC copied to the MAR
3	The contents (instruction) at the memory location (address) contained in MAR are then copied into the MDR	Lookup MAR and get contents. Copy contents into the MDR
4	The contents (instruction) in the MDR is then copied and placed into the CIR	Copy MDR contents into the CIR
5	The value in the PC is then incremented by 1 so that it now points to the next instruction which has to be fetched	PC is then incremented by 1
6	The instruction is finally decoded and then executed by sending out signals (via control bus) to the various components of the computer	The instruction is decoded and then executed
7	Repeat	