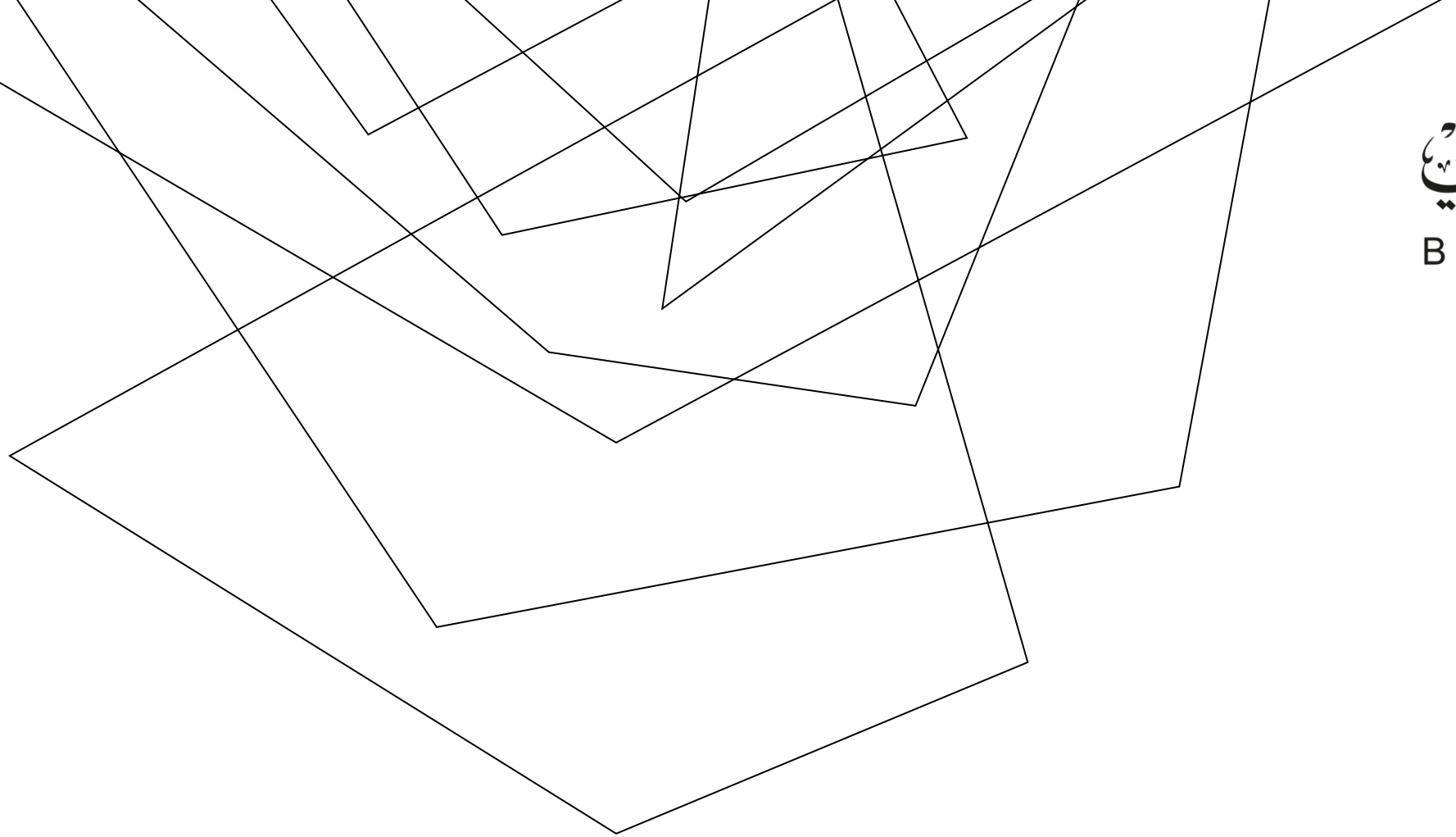


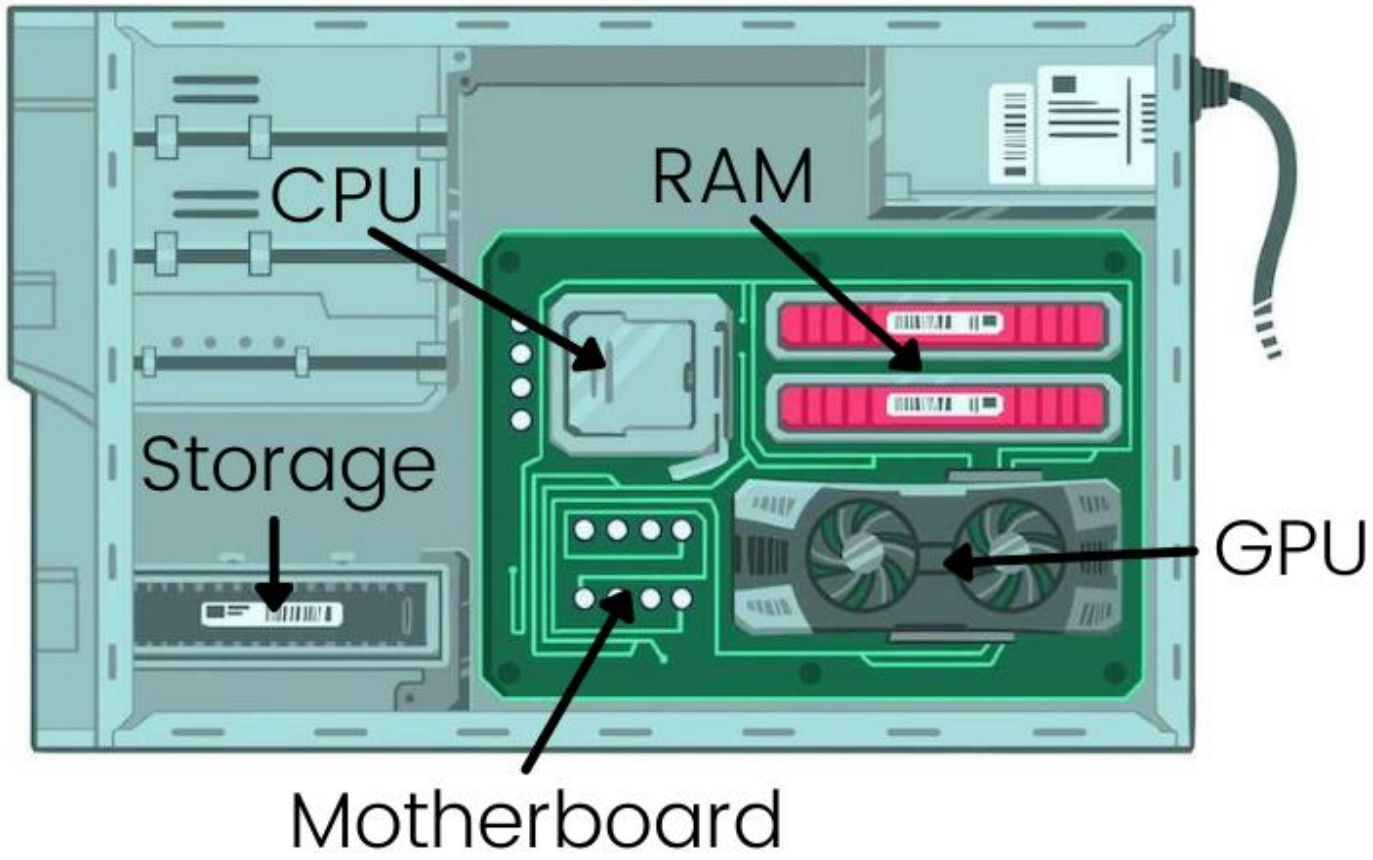
جامعة بيرزيت
BIRZEIT UNIVERSITY

MICROPROCESSORS

Shadi Daana

What is a computer?

What is a computer?



What is a computer?

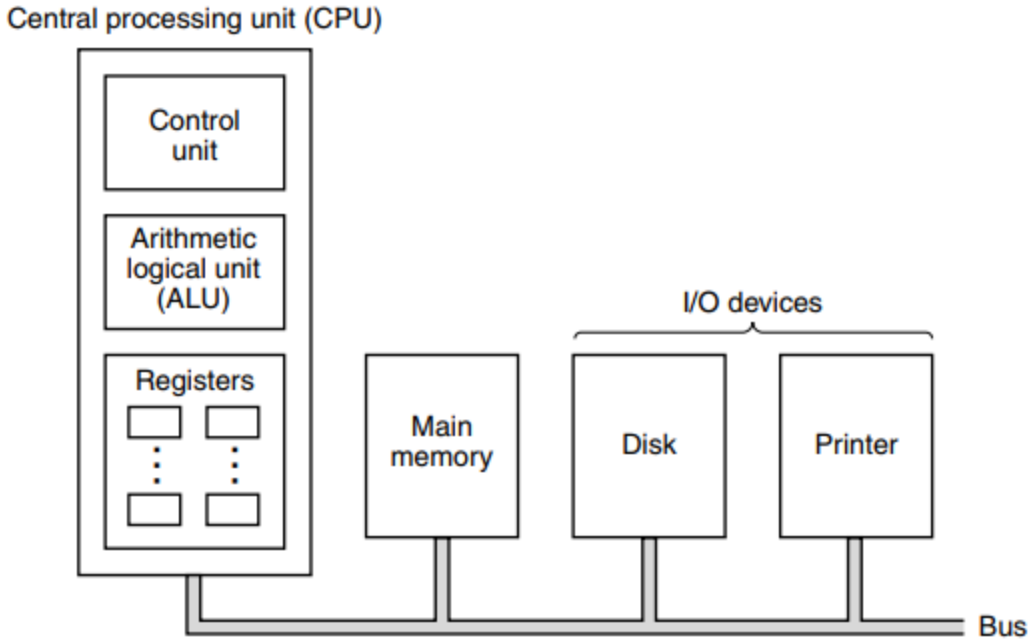


Figure 2-1. The organization of a simple computer with one CPU and two I/O devices.

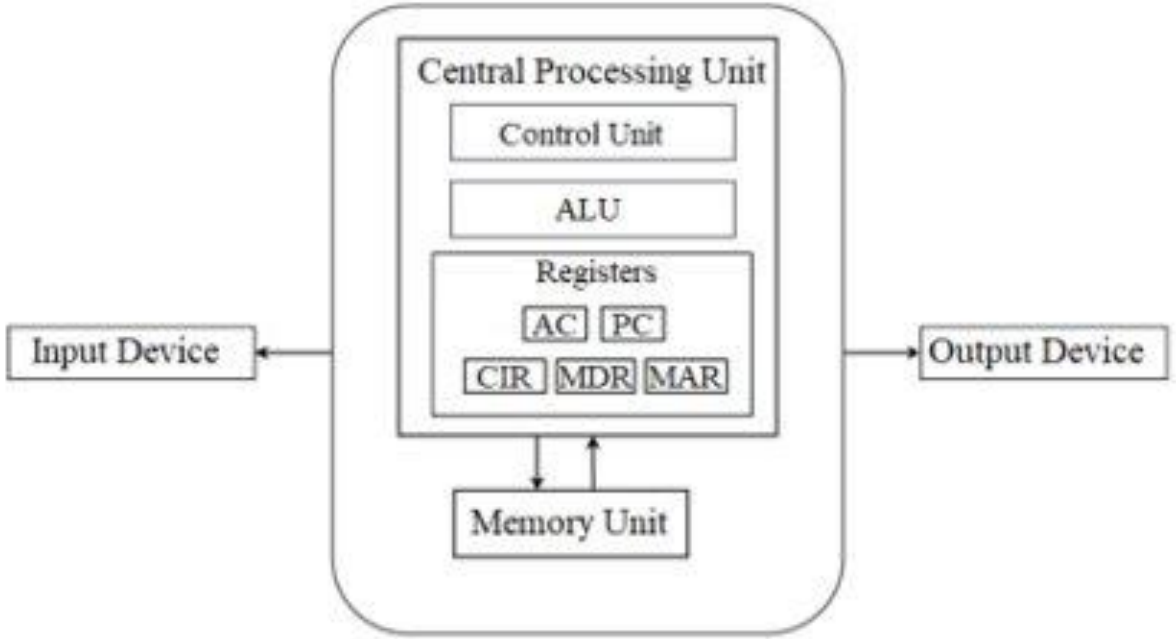
Source: Tanenbaum, A. S. (2016). *Structured computer organization*. Pearson Education India.

What is a computer?

- A computer refers to a digital electronic device (fixed **hardware**) that processes data according to **a set of instructions**.
- A sequence of instructions describing how to perform a certain task is called a **program**.
- The program or **software** consists of **instructions** that are stored within the computer memory and can be changed by loading new instructions into the memory
- This is called the **store-program** concept.
- The computer then **fetches** these instructions from memory and **executes** them

Central Processing Unit (CPU)

- CPU acts like the "**brain**" of a computer system. It contains the circuitry to interpret and execute program instructions.
- Its function is to execute programs stored in the main memory by **fetching** their instructions, examining them, and then **executing** them one after another



Central Processing Unit (CPU)

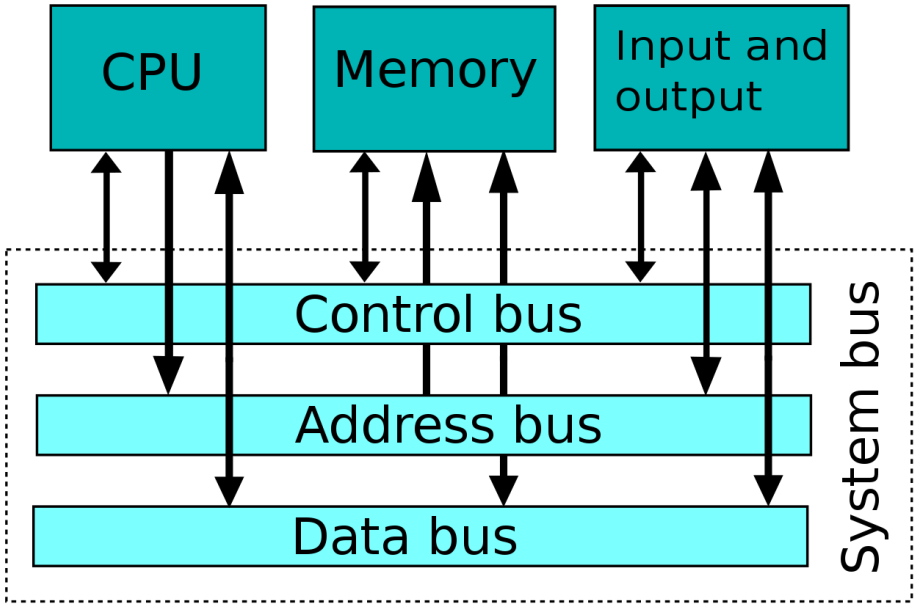
- **Control Unit (CU)**
  - The CU is responsible for fetching instructions from main memory and determining their type.
  - CU acts like a manager on a computer. CU receives orders from RAM in the form of instruction and decode (break) that instruction down into specific commands for other components inside a computer system.
  - It directs the data flow and the operation of the ALU.
- **Arithmetic/Logic Unit (ALU)**
  - ALU does all mathematical operations (arithmetic) (+/-/compare) and logical (AND/OR) calculations.

Central Processing Unit (CPU)

- **Registers**
  - A register is a small, very fast storage area inside CPU. It stores intermediate values from calculations or instructions that are needed again immediately.
  - They have very little capacity compared to main memory
  - In ARM Cortex -M processors there are 16 registers that are 32-bits wide
  - Reading from and writing data to the external main memory is comparably slow
  - Registers are internal to the CPU and can be accessed much more quickly than external memory
  - For instance, when a ALU is commanded to calculate A*(B+C), ALU needs to calculate B+C first, then ALU need to store the result for a moment and use the result to multiply A. It is faster for ALU to access register than store the data in memory units.

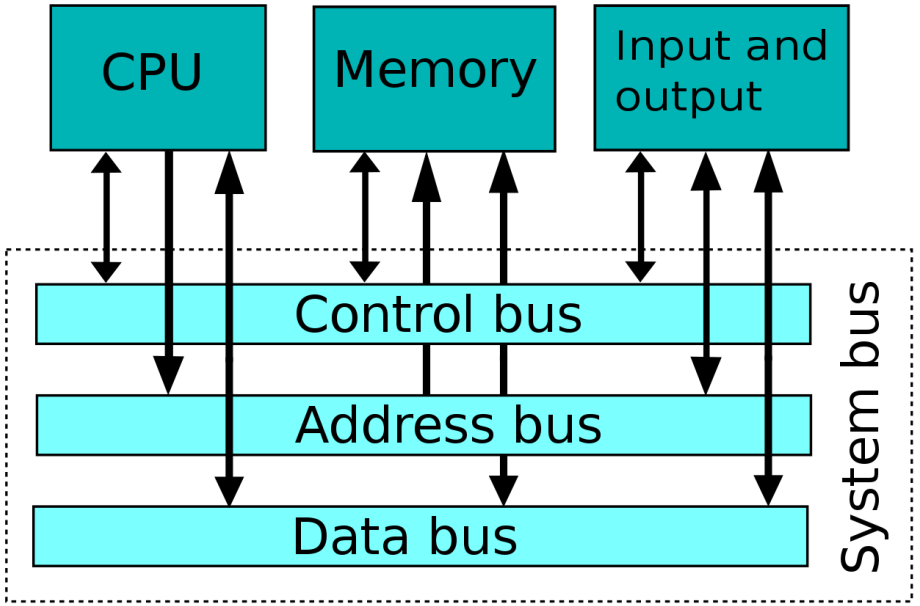
Buses

- The bus is the mechanism by which the CPU communicates with memory and devices.
- A bus is, at a minimum, a collection of wires but it also defines a protocol by which the CPU, memory, and devices communicate.
- One of the major roles of the bus is to provide an interface to memory
- The electrical wires are generally called the **System Bus** or **Front Side Bus (FSB)**

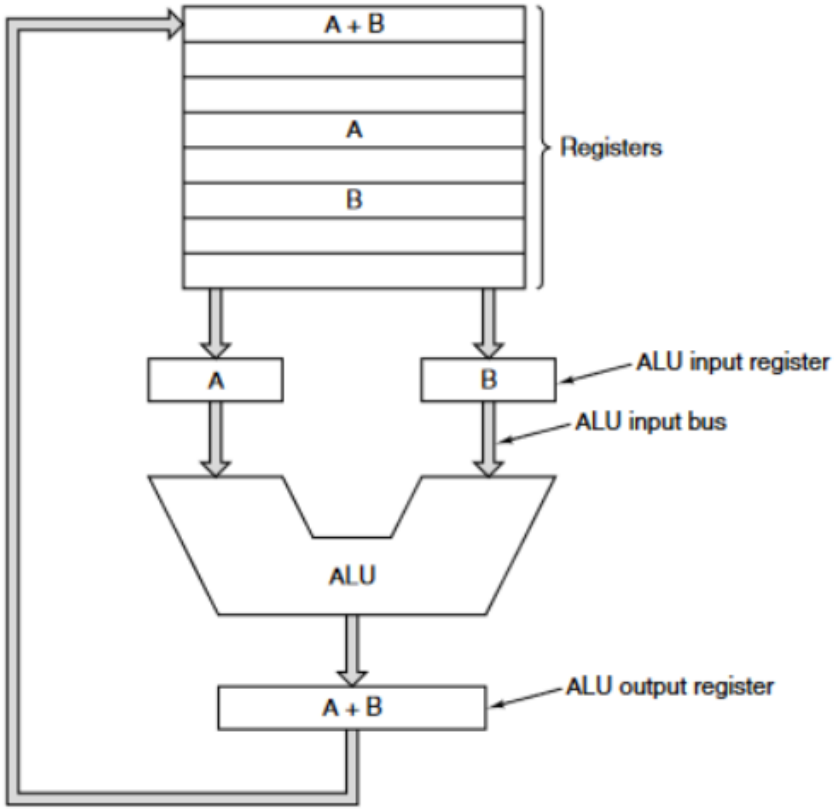


Buses

- There are three types of buses in a computer, which all flow together.
  - **Control bus** coordinates activity between various devices to prevent data collision. (i.e. clock, read/write, interrupts, DMA, etc.)
  - **Data bus** is a two-way bus that allows data to flow between devices. (8, 16, 32, or 64 bits)
  - **Address bus** is a one-way bus that tells devices where the data should go or is coming from. (8, 16, 32, or 64 bits)



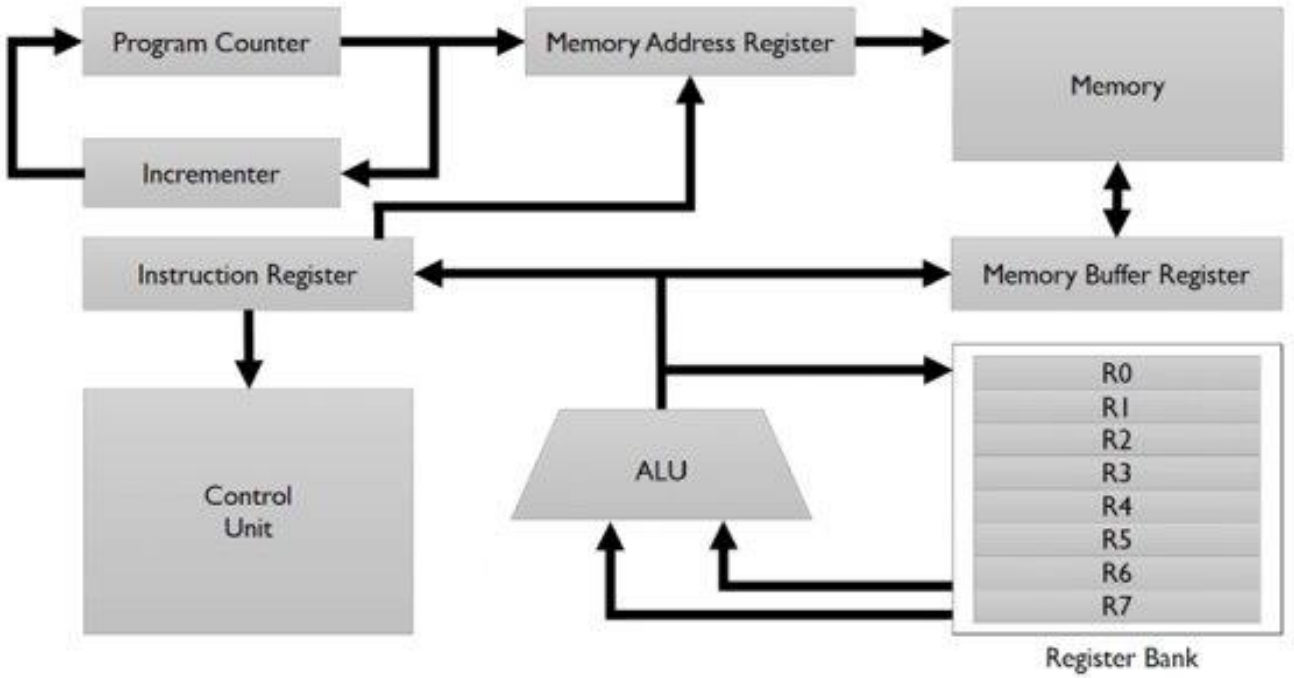
Datapath



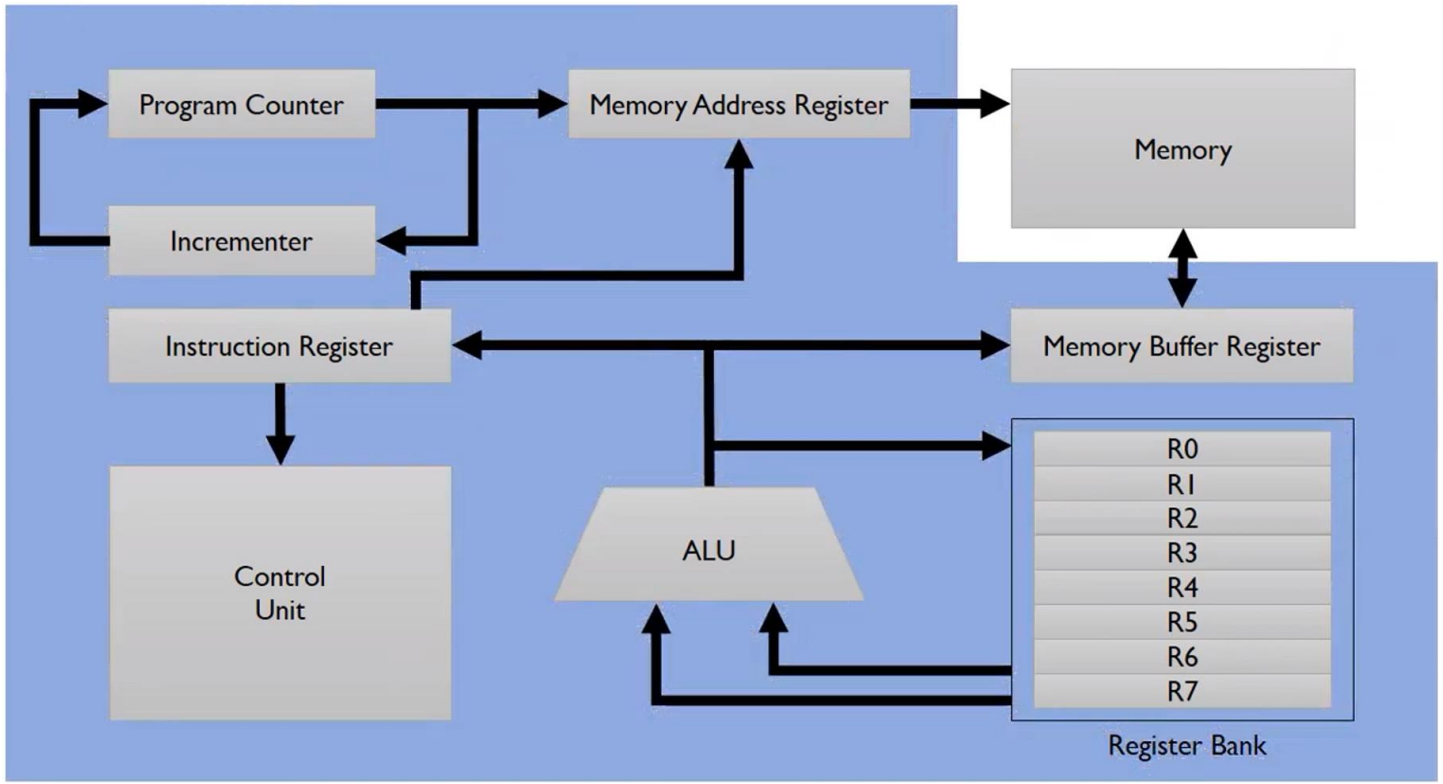
Datapath

- The CPU can be divided into a data section and a control section.
- The data section, which is also called the datapath, contains the registers and the ALU.
- The datapath is capable of performing certain operations on data items.
- The control section is basically the control unit, which issues control signals to the datapath.
- Internal to the CPU, data move from one register to another and between ALU and registers. Internal data movements are performed via local buses, which may carry data, instructions, and addresses.
- Externally, data moves from registers to memory and I/O devices, often by means of a system bus.
- Internal data movement among registers and between the ALU and registers may be carried out using different organizations including one-bus, two-bus, or three-bus organizations.
- Dedicated datapaths may also be used between components that transfer data between themselves more frequently. For example, the contents of the PC are transferred to the MAR to fetch a new instruction at the beginning of each instruction cycle.

Datapath



Datapath



Registers

- **Program Counter (PC)**: stores the address of the next instruction*
- The **incrementer**: is used to increment the value of stored **PC**
- **MAR** (memory address register): store memory that currently be accessed inside RAM
- **MDR/MBR** (memory data/buffer register): store data from memory
- **General-purpose registers**: General-purpose registers hold either data or an address
- **Instruction register (IR):** Stores the instruction currently being executed

*The name ‘‘program counter’’ is somewhat misleading because it has nothing to do with counting anything, but the term is universally used

Programming a CPU

- When we program often we use a high-level language (C, C++, etc.).
- These are converted into a lower-level language (LLL) using a **compiler**.
- The Assembly language is the language of actual CPU.
- In order for the CPU to understand the LLL, this is converted into binary/hex via an **assembler**.

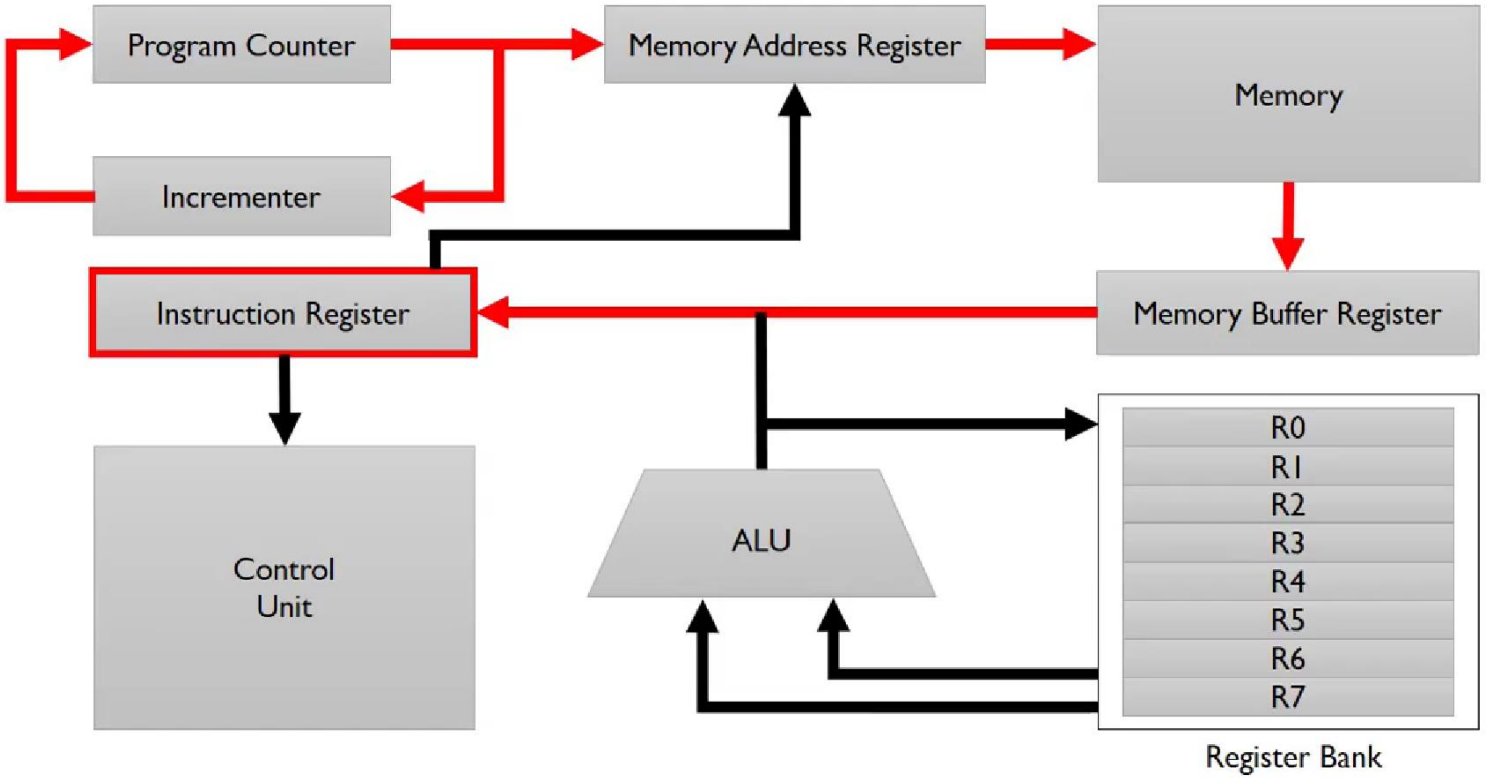
How does a CPU work?

- The program/data is stored in external memory in binary/hex is transferred into the CPU via the **system bus**.
- The CPU spends most of its time reading in the program, understanding what the command is and then doing the command.
- The CPU executes each instruction in a series of small steps. It fetches the next instruction from memory, determines the type of instruction, and then executes the instruction.
- This process is repeated for as long as the computer is powered.
- This sequence of steps is frequently referred to as the **instruction execution cycle** or the **fetch-decode-execute** cycle.

Instruction execution cycle - Fetch

- **During the fetch cycle**
  - The value in the PC is copied to the MAR
  - The location in the memory pointed at by MAR is read
  - The instruction is copied to the MBR
  - The content of the MBR is copied to the IR
  - The value in the PC is incremented

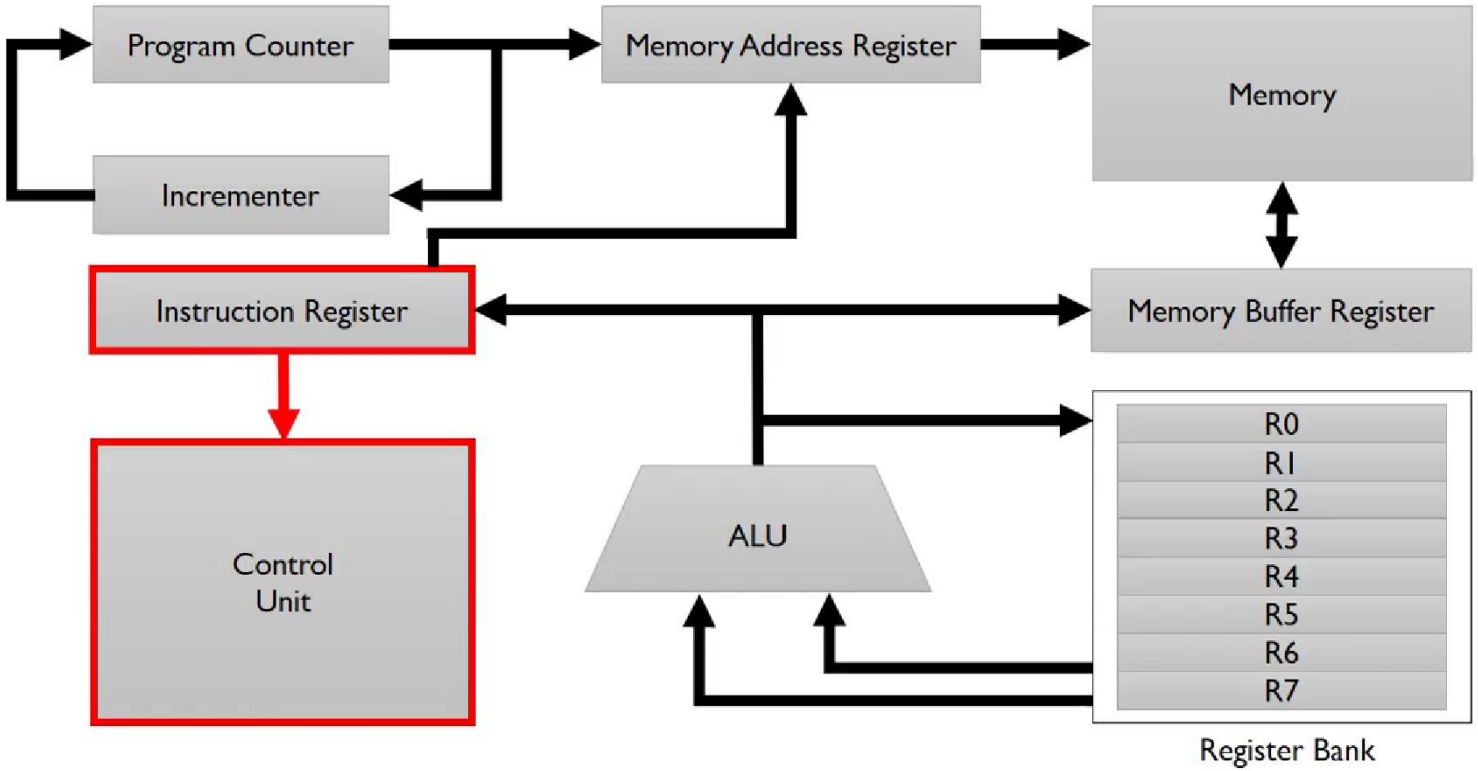
Instruction execution cycle



Instruction execution cycle - Decode

- In the next step, determining the type of instruction in the IR register (Decode)
- There are different types of instruction including:
  - **Data processing**
  - **Loading** data from memory
  - **Storing** data in memory
- Only data in registers can be operated on, data in main memory should be loaded first (**load-store concept**).

Instruction execution cycle - Decode

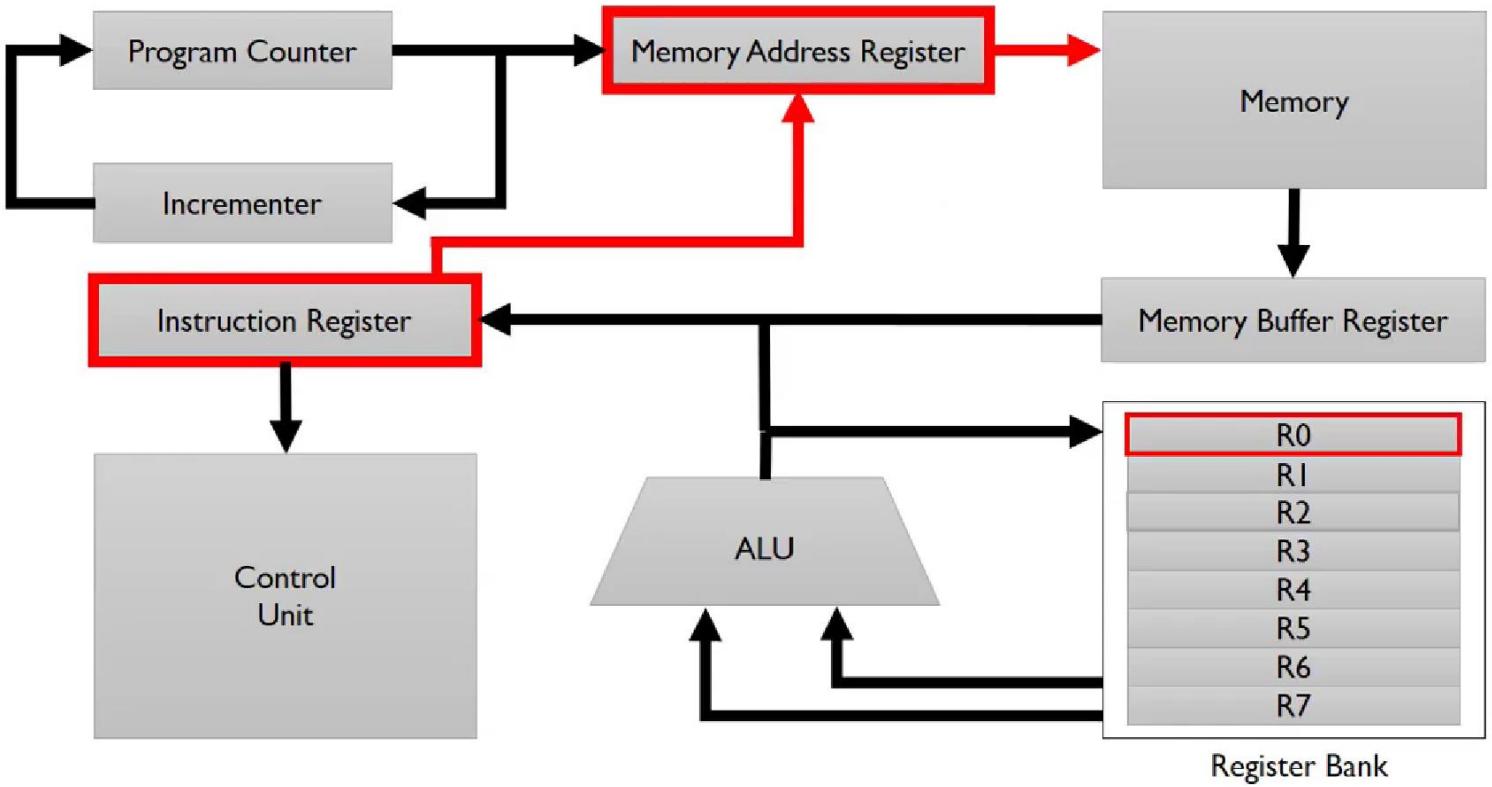


Instruction execution cycle - Execute

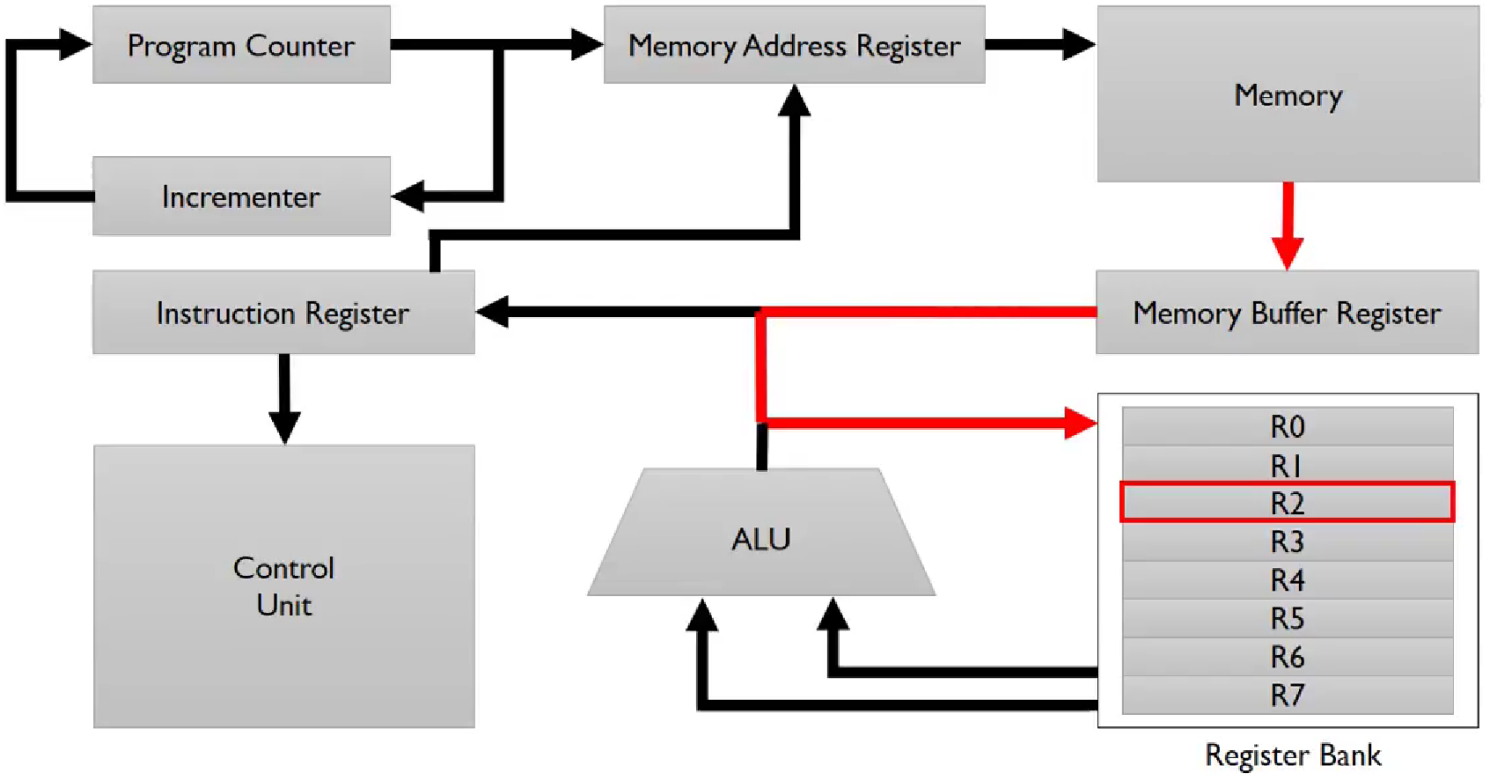
- For a **load** instruction data from memory is written to a register in the register bank
- The address of the memory location to read from may be stored in another register

LDR R2, [R0]

Instruction execution cycle - Execute

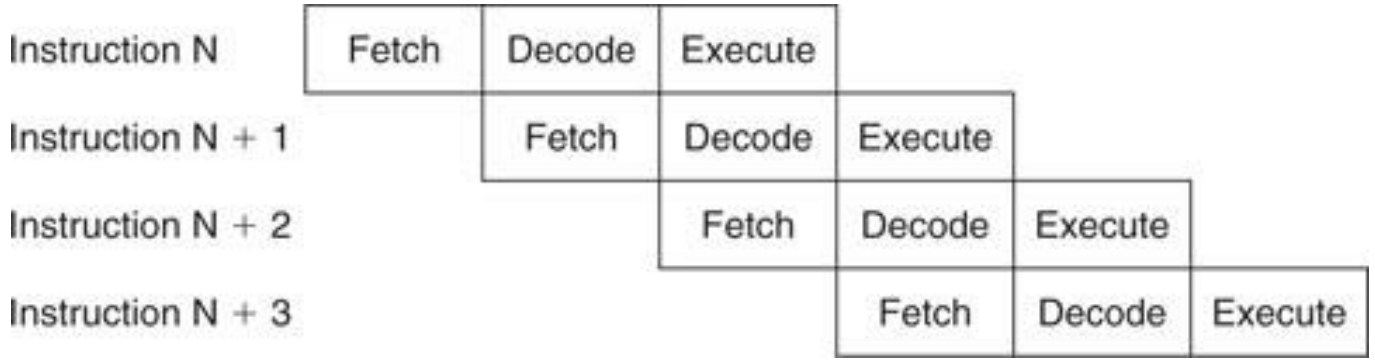


Instruction execution cycle - Execute



Pipelining

- The Cortex-M3/M4 processor has a three-stage pipeline.
- The **pipeline stages** are instruction fetch, instruction decode, and instruction execution



Architecture

- The term can be used in different ways
- The **instruction set architecture (ISA)** describes the instructions that the computer can execute and is the view that is most important to a computer programmer.
- The **term microarchitecture** refers to the way in which the ISA is implemented in hardware i.e. an ADD instruction maybe implemented using a binary adder
- The **system architecture** describes the complete system including the processor, memory, busses and peripheral.

Instruction Set Architecture

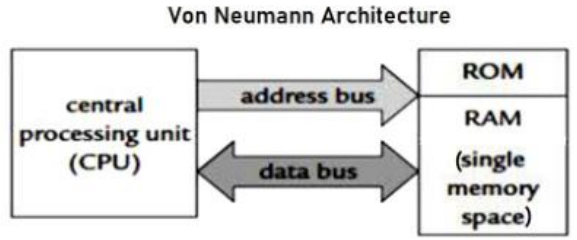
- An **Instruction Set Architecture (ISA)** is part of the abstract model of a computer that defines how the CPU is controlled by the software.
- The **ISA** acts as an interface between the hardware and the software, specifying both what the processor is capable of doing as well as how it gets done.
- In simple words, the ISA is defined as the entire group of commands that the processor can perform, to execute the program instructions

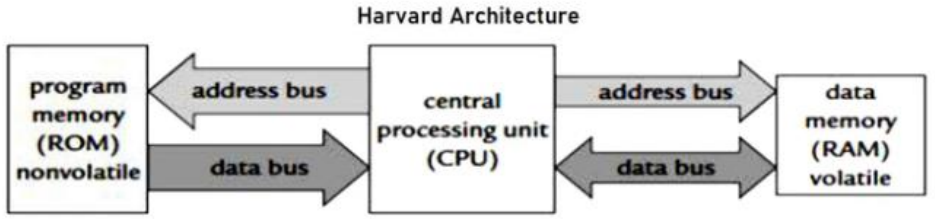
RISC and CISC

- A microcontroller is an example of a **reduced instruction set computer (RISC).**
- RISC machines have a simpler architecture with typically fewer instructions than a **complex instruction set computer (CISC)**
- In a CISC machine there can be lots of very complex instructions requiring complex hardware to implement them
- The instructions may take different amounts of time (clock cycles) to execute
- They may also be different sizes

Harvard VS Von Neumann

- The **Von Neumann architecture** is a type of computer architecture in which the central processing unit (CPU), memory, and input/output (I/O) devices all use a single shared bus for communication
- **Harvard architecture** is a type of computer architecture that has separate memory spaces for instructions and data



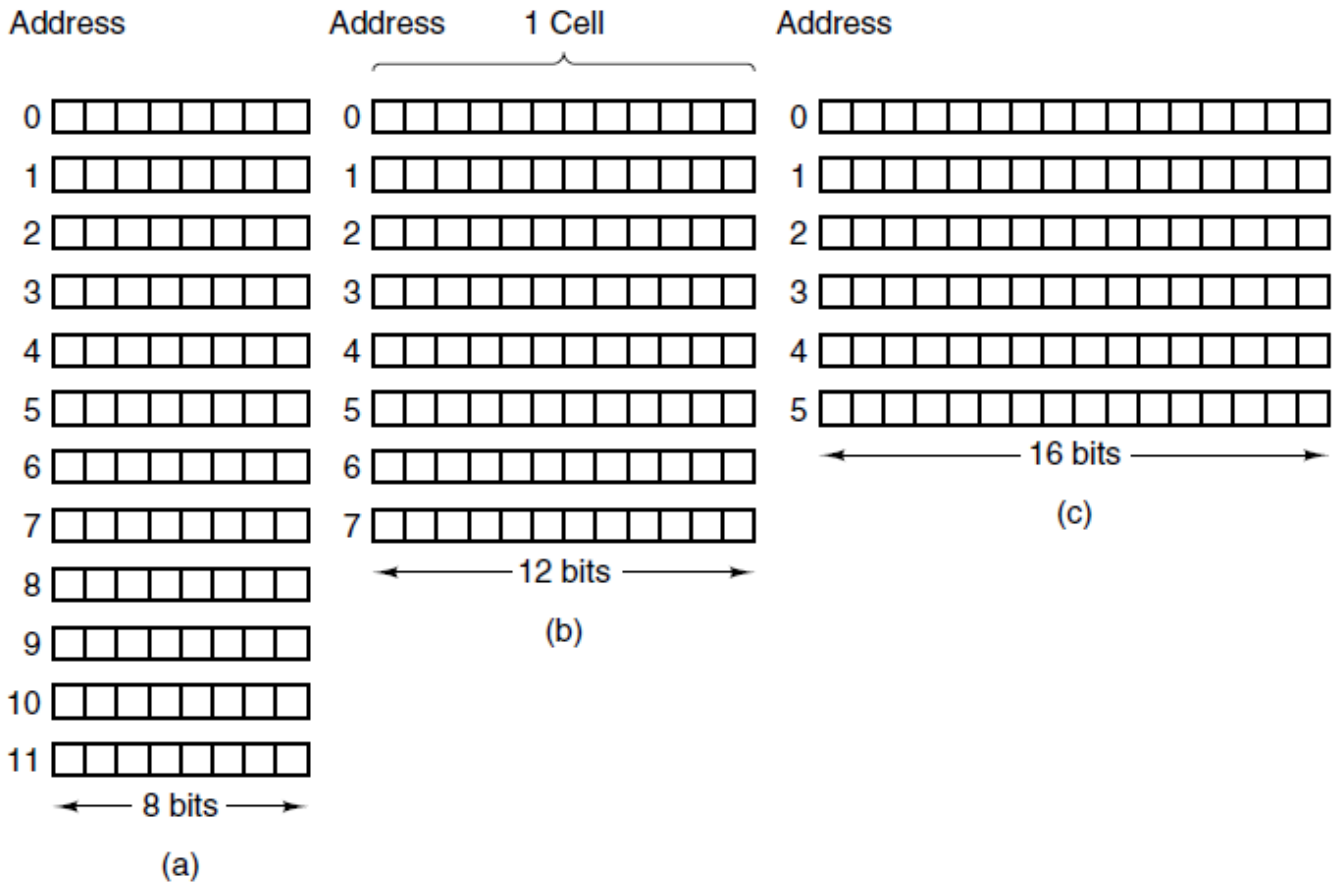


Memory

- Memories consist of a number of **cells** (or locations), each of which can store a piece of information.
- Each cell has a unique number, called its **address**, by which programs can refer to it.
- If a memory has ***n cells***, they will have addresses ***0 to n – 1***.
- All cells in a memory contain the same number of bits.
- If a cell consists of ***k bits***, it can hold any one of $\mathbf{2^k}$ different bit combinations
- If an address has ***m bits***, the maximum number of cells addressable is $\mathbf{2^m}$

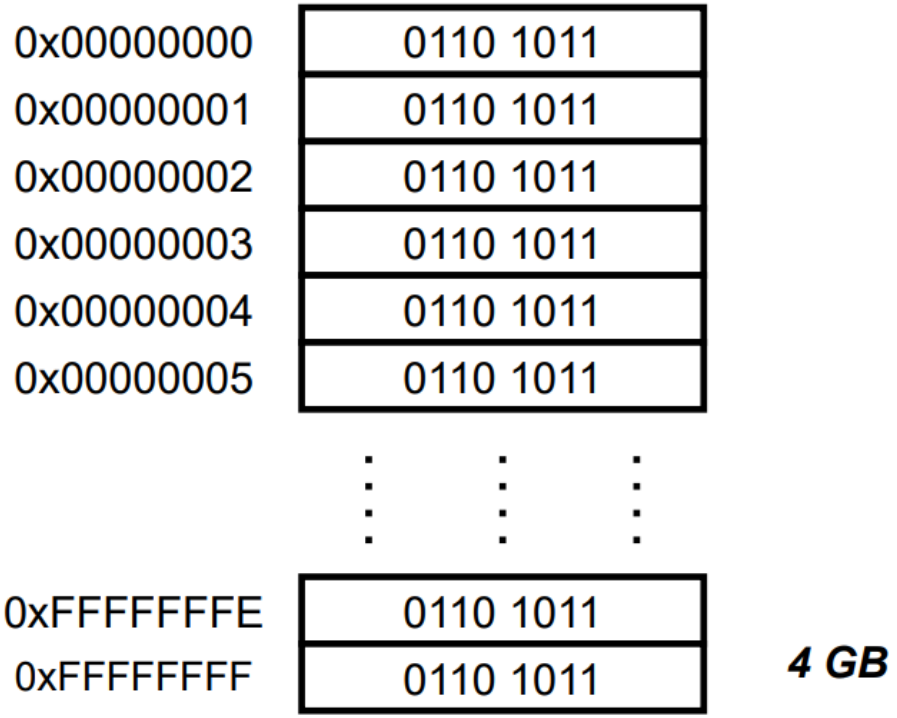
Memory

- This figure shows three different organizations for a 96-bit memory
- (a) needs at least 4 bits in order to express all the numbers from 0 to 11.
- A 3bit address is sufficient for 0-7.
- The number of bits of memory address is independent of the number of bits per cell



Memory

- The significance of the cell is that it is the smallest addressable unit.
- The memory cell is usually an **8-bit** cell, which is called a **byte.**
- A 32-bit address bus means 2^{32} = 4,294,967,296 bytes (4 GB) can be addressed
- **Bytes** are grouped into words.
- Usually, the defined bit length of a word is equivalent to the width of the computer's data bus.



Data Structure Alignment

- A computer with a 32-bit word has 4 bytes/word, whereas a computer with a 64-bit word has 8 bytes/word.
- **e.g.** Arm Cortex-M microprocessors are 32 bit.
- The word size therefore 32-bit (4 bytes).
- **Data words** then must be stored in an address that is a multiple of 4 for the most efficient performance.
- **Data alignment**: Data alignment means putting the data in memory at an address equal to some multiple of the word size.

0x00000000	0	1	2	3
0x00000004	4	5	6	7
0x00000008	8	9	10	11
0x0000000C	12	13	14	15
0x00000010	16	17	18	19
0x00000014	20	21	22	23
0x00000018	24	25	26	27

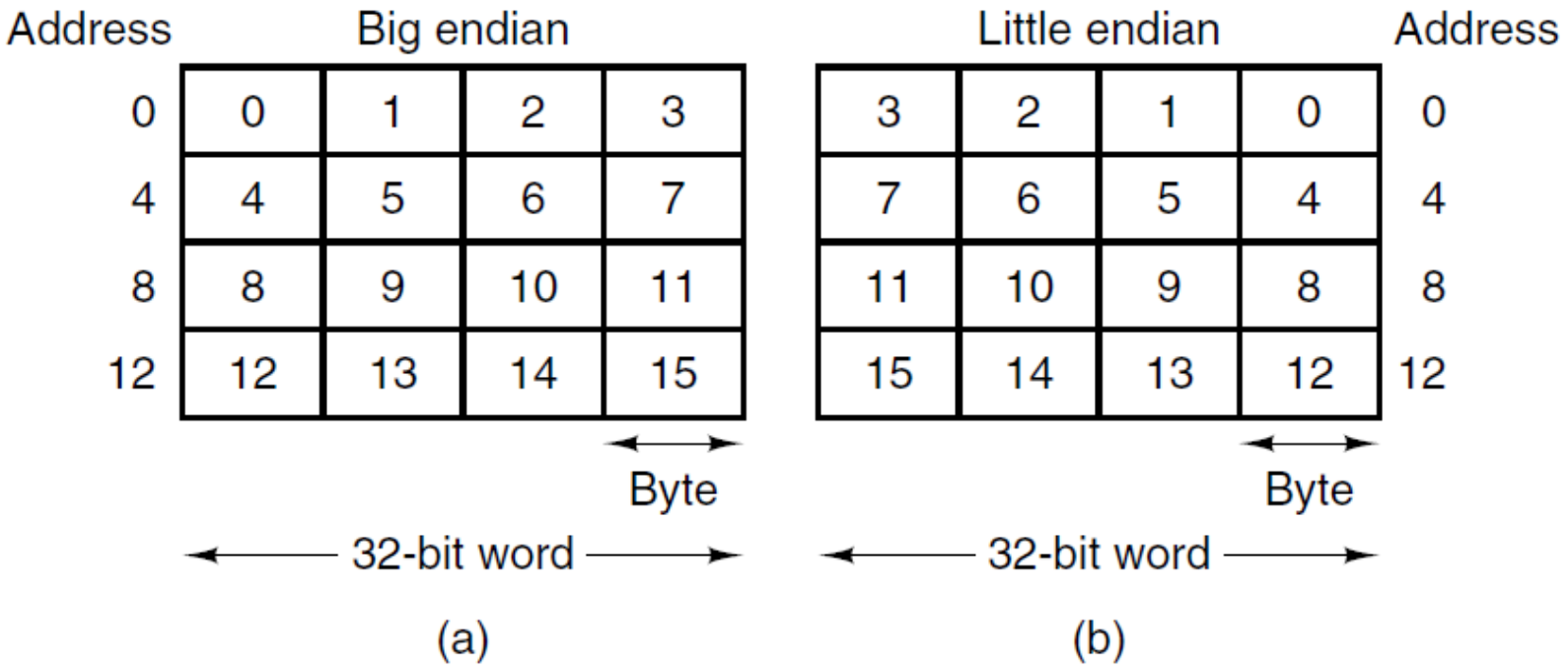
Data Structure Alignment

- 32-bit instructions are stored at memory addresses that are multiple of 4
- Therefore, the PC would also increment by 4 to fetch the next instruction

0x00000000	0	1	2	3
0x00000004	4	5	6	7
0x00000008	8	9	10	11
0x0000000C	12	13	14	15
0x00000010	16	17	18	19
0x00000014	20	21	22	23
0x00000018	24	25	26	27

Byte Ordering

- The bytes in a word can be numbered from left to right or right to left.



Questions?