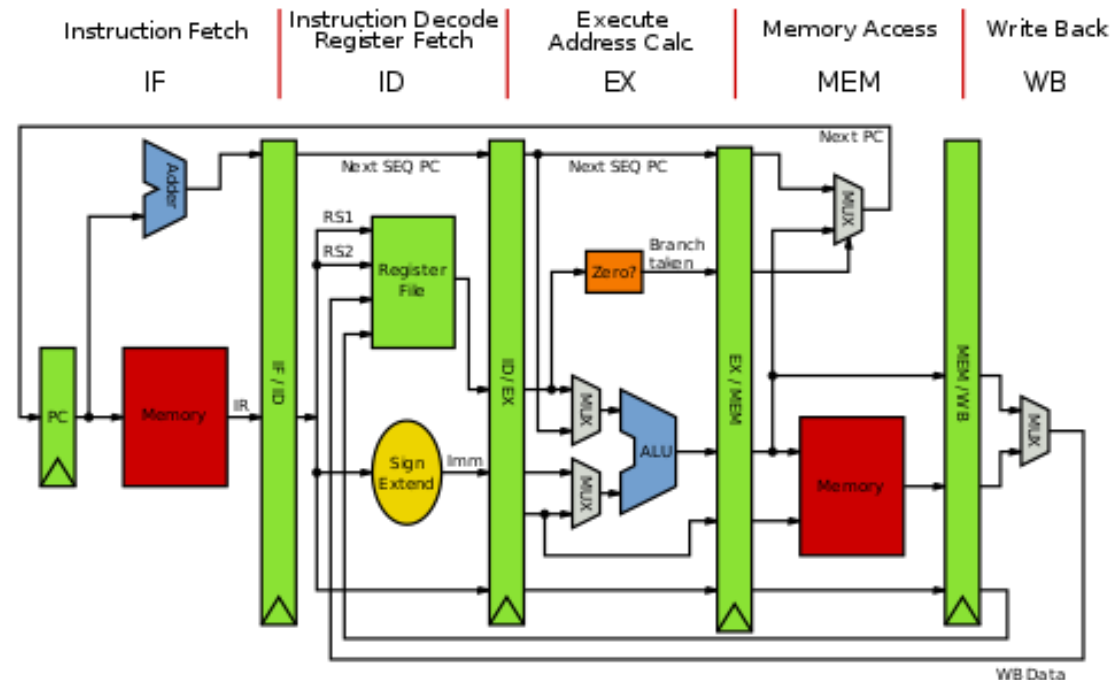


CHAPTER- 4

MARIE: AN INTRODUCTION TO A SIMPLE COMPUTER



Dr. Rania Baashirah
r.baashirah@ubt.edu.sa



Review

- We introduced MARIE
- **CPU** is Datapath + Control unit
- **Datapath** = Registers + ALU
- **Registers**: store data – flipflops – program counter and status register are special types
- **ALU** does math and logic operations
- All of these units are connected by **buses**: Address bus, Data bus, control lines
- **Common types of bus**:
 1. Processor-memory: high speed, closely matched to memory
 2. I/O bus
 3. Backplane bus: connects CPU, memory, I/O devices to share the bus
- **Forms of bus**: Synchronous, Asynchronous

Clocks

- A computer contains an internal clock that measures how quickly instructions can be executed.
- $\text{clock cycle time} = \frac{1}{\text{clock rate}} \text{ sec.}$
- $\text{clock rate} = \frac{1}{\text{clock cycle time}} \text{ Hz.}$
- **Examples:**
 - If a machine has 800 MHz clock rate, what is its clock cycle time?
 - If a machine has 2 ns cycle time, what is the clock rate?

Most machine instructions require one or two clock cycles, but some can take 35 or more.

Clocks

- $\text{clock cycle time} = \frac{1}{\text{clock rate}} \text{ sec.}$
- $\text{clock rate} = \frac{1}{\text{clock cycle time}} \text{ Hz.}$
- **Examples:**
 - If a machine has 800 MHz clock rate, what is its clock cycle time?
 - If a machine has 2 ns cycle time, what is the clock rate?

Clocks

- Machines are synchronous → when clock ticks, data changes
- Can we assume faster clock cycle → runs faster machine?
- We need to wait for data propagation in the circuits to complete in all registers.
- The minimum clock cycle time must be at least greater than the maximum propagation delay of the circuit.
- If the propagation circuit needs more than the clock cycle time then we can say that it need m clock cycle
 - m is usually 1 or 2, sometimes increased up to 35.

Clocks

- The overall execution time for a given program calculated as:

$$CPU\ Time = \frac{instructions}{program} * \frac{average\ cycles}{instruction} * \frac{seconds}{cycle}$$

- **Example:**
- A program consists of 300 instructions. Engineers believe that this computer takes 2 cycles to execute most of the instructions. Knowing that the single cycle is achieved in 2 nanosecond. Please calculate the overall execution time?
- **Execution Time =**
- **$300 \times 2 \times 2 \times 10^{-9} = 1200 \times 10^{-9} = 1200\text{ ns} = 1.2\text{ }\mu\text{s}$**

The Input/ Output Subsystem

- **(I/O) devices (peripherals):** allow us to communicate with the computer system.
- I/O is the transfer of data between primary memory and various peripherals.
- **Ex.** Keyboard, mouse, scanner, microphone → **Input**
- **Ex.** Monitor, printer, speaker → **Output**
- These I/O devices are not directly connected to CPU
- An **interface** handles data transfers → It converts the system bus signals to/from a format that is acceptable to the given device.
- The CPU communicates to external devices via **I/O registers**.

The Input/ Output Subsystem

- The CPU communicates to external devices in two ways:
- **Memory-mapped I/O:**
 - The register in the interface appear in the computer's memory map → there is no real difference between accessing the memory and accessing I/O device.
 - Like accessing Memory using address bus.
 - It speeds up the access, but uses up the memory space.
- **Instruction-based I/O:**
 - CPU has specialized instructions that perform input/output.
 - It does not use memory space, but requires specific I/O instructions.
 - It can only be used by CPU to execute these instructions.

Memory Organization And Addressing

- Memory is nothing but binary data arranged in rows.
- Each row, implemented by a register called **Memory Location**.
- Each row has a length of computer word size
- Usually one byte (**byte addressable = has unique address**), word size = 8-bits
- In Some architectures, word size = 16-bits or 32-bits or more (**word addressable**).

Address \longleftrightarrow 8-bit \longrightarrow

1								
2								
3								
4								
...								
N								

N 8-Bit Memory Locations

Address \longleftrightarrow 16-bit \longrightarrow

1															
2															
3															
4															
...															
M															

M 16-Bit Memory Locations

Addressing

- An Address is always represented by unsigned integer.
- Memory is byte addressable → each byte has unique address
- Each location (word) has its own address called **Memory Address**.
- If the system is 32-bit word, and it is a byte addressable → it manipulates 32-bit at the time
- It is a byte addressable → address of the first byte of the word will be the address of the whole word.
- Ex. Street (*memory*) with buildings (*words*) with unique addresses, each building has numbered apartments (*bytes*)
 - NOTE: also still each byte has its own address in this system.

Addressing

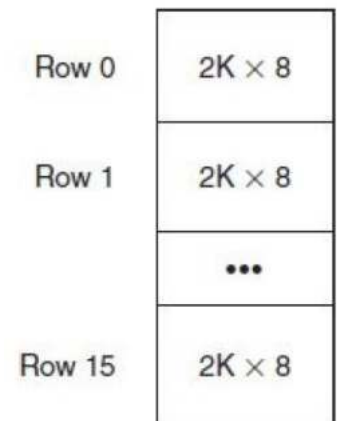
- Memory is using notation length \times width ($L \times W$).
 - For example, $4M \times 8$ means $4M$ (2^{22}) locations, 8 bits for a word.
 - For example, $4M \times 16$ means $2M$ (2^{22}) locations, 16 bits for a word.
 - For example, $2M \times 32$ means $2M$ (2^{21}) locations, 32 bits for a word.

Total Items	2	4	8	16	32
Total as a Power of 2	2^1	2^2	2^3	2^4	2^5
Number of Bits	1	2	3	4	??

if a computer has 2^N addressable units of memory, it requires N bits to uniquely address each unit.

Multiple RAMs

- Memory is built from RAM (Random Access Memory).
- Main memory is usually larger than one RAM chip.
- **Example:** $32K \times 8$ byte-addressable, and $2K \times 8$ RAM chip. In this case we need 16 chips.
- A single memory module causes sequentialization of access (only one memory access at a time).
- **Memory interleaving**, which splits memory across multiple memory modules.
- **Example:**
- **Design $32K \times 16$ byte-addressable, and $2K \times 8$ RAM chip**



Memory Banks

- Low-order interleaving

Module 0	Module 1	Module 2	Module 3	Module 4	Module 5	Module 6	Module 7
0	4	8	12	16	20	24	28
1	5	9	13	17	21	25	29
2	6	10	14	18	22	26	30
3	7	11	15	19	23	27	31

Example:

byte-addressable memory consisting of 8 modules of 4 bytes each, for a total of 32 bytes of memory



Module	Decimal Word Address	Binary Address	Address Split per Given Structure	Module Number	Offset in Module
Module 0	0	00000	000 00	0	0
	1	00001	000 01	0	1
	2	00010	000 10	0	2
	3	00011	000 11	0	3
Module 1	4	00100	001 00	1	0
	5	00101	001 01	1	1
	6	00110	001 10	1	2
	7	00111	001 11	1	3

Interrupts

- **Interrupts:** are events that alter (or interrupt) the normal flow of execution in the system.
- It can be triggered for many reasons, including:
 - I/O requests
 - Arithmetic errors (e.g., division by 0)
 - Arithmetic underflow or overflow
 - Hardware malfunction (e.g., memory parity error)
 - User-defined break points (such as when debugging a program)
 - Page faults (this is covered in detail in Chapter 6)
 - Invalid instructions (usually resulting from pointer issues)
- An interrupt can be initiated by the user or the system,
- It can be:
 - **Maskable:** disabled or (ignored)
 - **Non-maskable:** high-priority interrupt cannot be disabled and must be acknowledged and can result in terminating the program.

