Unit 2

Register Transfer and Microoperations

2.1 Register and Register Transfer Language

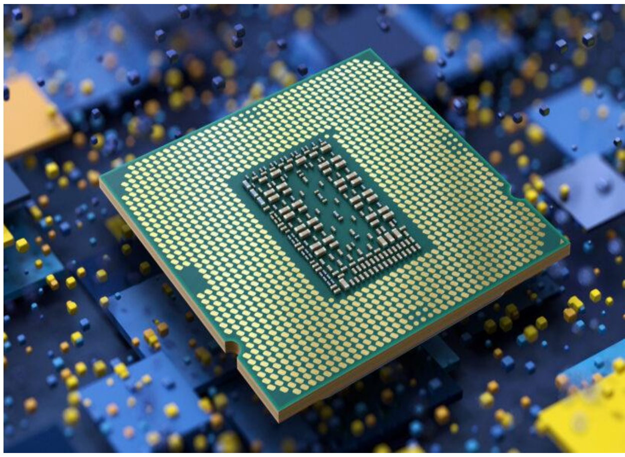
2.2 Bus and Memory Transfers

2.3 Arithmetic, Logic and Shift Micro-operations

2.4 Arithmetic Logic Shift Unit

Sanjeev Thapa, Er. DevOps, SRE, CKA, RHCSA, RHCE, RHCSA-Openstack, MTCNA, MTCTCE, UBSRS, HEv6, Research Evangelist

What is a CPU?



The **CPU (Central Processing Unit)** is the **brain of a computer or mobile device**.
It performs all major operations such as:

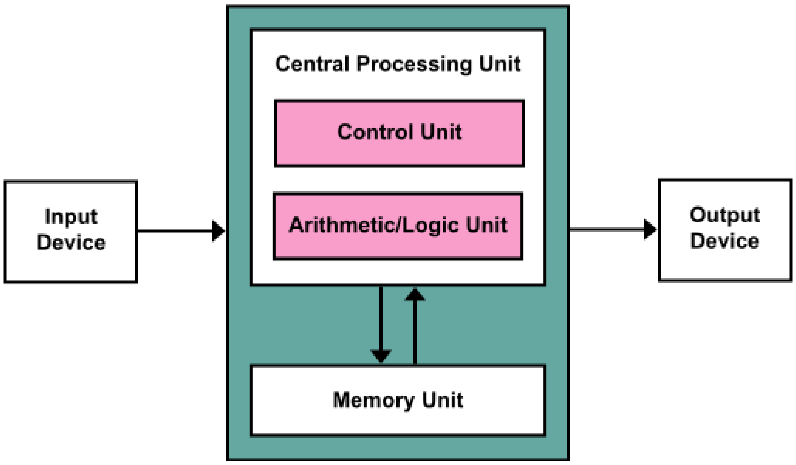
- Fetching instructions
- Decoding them
- Executing mathematical and logical tasks
- Storing or transferring results

A CPU works billions of times per second (GHz speed).

Modern examples

- In laptops: Intel i5, i7, AMD Ryzen CPUs
- In mobile phones: Snapdragon, Apple A16 Bionic, MediaTek, Exynos

Even apps like Facebook, TikTok, PUBG, banking apps all run because the CPU executes the instructions behind them.



CPU Components

A CPU has three main internal units:

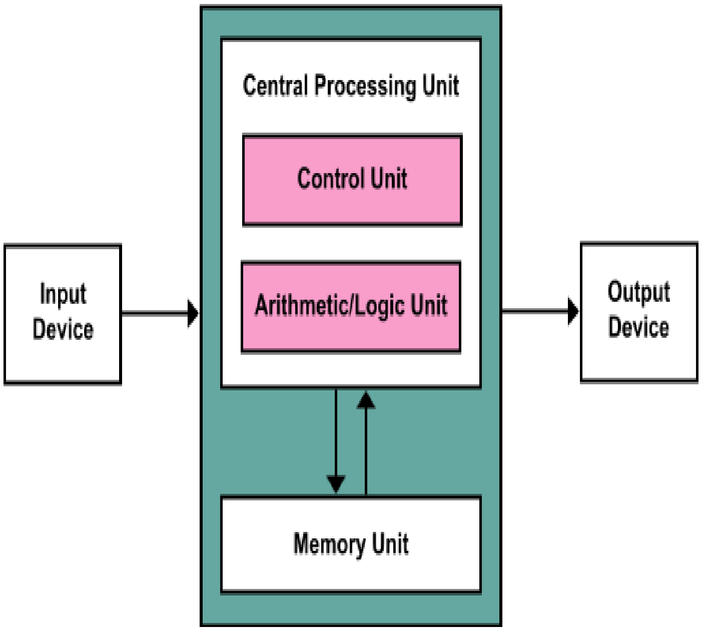
1. Register Set

Small storage areas inside CPU used for fast data access.

2. ALU (Arithmetic Logic Unit)

Performs operations like addition, subtraction, AND, OR, shifts, comparisons, etc.

3. CU (Control Unit)



Sanjeev Thapa, Er. DevOps, SRE, CKA, RHCSA, RHCE, RHCSA-Openstack, MTCNA, MTCTCE, UBSRS, HEv6, Research Evangelist

Controls the flow of data, signals, timing, fetch–decode–execute cycle.

Modern mobile/PC CPUs also include:

- Cache memory (L1, L2, L3)
- Floating point unit (FPU)
- Vector processing units (NEON in ARM, SIMD on Intel)
- Pipelines
- AI accelerators

2.1 Register and Register Transfer Language (RTL)

What is a Register?

A register is a **small, fast storage unit** inside the CPU used to hold data temporarily while instructions are being executed.

Examples of CPU Registers

- **PC (Program Counter):** holds address of next instruction
- **IR (Instruction Register):** holds current instruction
- **MAR (Memory Address Register):** holds memory address
- **MDR (Memory Data Register):** holds data being transferred
- **ACC/AC (Accumulator):** used for arithmetic/logic operations
- **General Purpose Registers (R0, R1, R2 …)**

Registers are faster than cache and RAM.

Register Transfer Language (RTL)

RTL is a symbolic notation that describes how data moves between registers and what operation is performed.

Common RTL Syntax

- R1 ← R2
  Transfer contents of R2 to R1

- R3 ← R1 + R4
  ALU adds R1 and R4 and stores result in R3
- PC ← PC + 1
  Program Counter increments
- If (Z = 1) then PC ← AR
  Conditional branch if Zero flag is set

Example (Simple Instruction Execution)

Instruction: ADD R1, R2
RTL Steps:

1. R3 ← R1 + R2
2. PC ← PC + 1

This means CPU adds values from R1 and R2, stores in R3, then goes to next instruction.

2.2 Bus and Memory Transfers

A **bus** is a set of parallel wires used to transfer data among CPU, memory, and I/O devices.

Types of Buses

1. Data Bus

Carries actual data (numbers, characters).

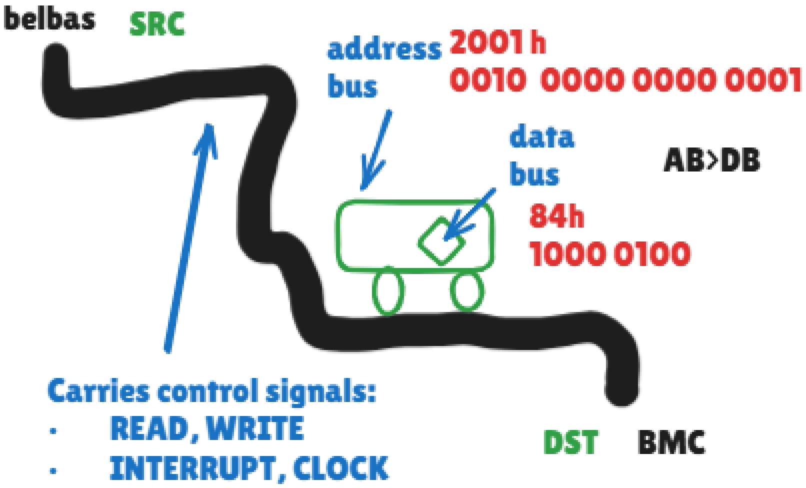
2. Address Bus

Carries memory addresses that CPU wants to access.

3. Control Bus

Carries control signals:

- READ
- WRITE
- INTERRUPT
- CLOCK



Memory Transfer Operations

1. Memory Read

MDR ← Memory[MAR]

CPU reads data from memory into MDR.

2. Memory Write

Memory[MAR] ← MDR

CPU writes data from MDR to memory.

Example of Bus Operation

You open a photo on your phone:

1. CPU sends photo’s memory address on **address bus**
2. RAM sends photo data on **data bus**
3. Control bus coordinates read/write timing

2.3 Arithmetic, Logic, and Shift Micro-operations

Micro-operations are small operations performed on registers.

A. Arithmetic Micro-operations

Performed by the Arithmetic Unit of the ALU.

Examples

- R1 ← R2 + R3
- R4 ← R4 - 1
- PC ← PC + 1
- AC ← AC + DR

Example

If R2 = 5 and R3 = 9
R1 ← R2 + R3 → R1 = 14

B. Logic Micro-operations

Operate on bits using logic gates.

Operations

- AND
- OR

- XOR
- NOT
- NAND, NOR

Example

If
R1 = 10110110 R2 = 11001100

Then
R3 ← R1 AND R2 → R3 = 10000100
R4 ← NOT R1 → 01001001

C. Shift Micro-operations

Used for shifting/rotating bits.

Types

- Logical Shift Left (LSL)
- Logical Shift Right (LSR)
- Arithmetic Shift Left
- Arithmetic Shift Right
- Rotate Left (ROL)
- Rotate Right (ROR)

Example

Value: 1011 (binary = 11 decimal)

Logical Shift Left:
1011 → 0110 (multiplies by 2 → result = 22)

Logical Shift Right:
1011 → 0101 (divides by 2 → result = 5)

Step-by-step shift and rotate operations on the hexadecimal data:

A = **25h** and B = **3Bh**.

---

1. Convert Hex to Binary
A = 25h

2 = 0010
5 = 0101

A = **0010 0101**

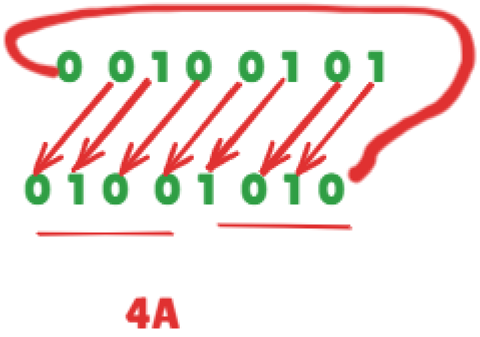
B = 3Bh
3 = 0011
B = 1011

B = **0011 1011**

✅ **SHIFT & ROTATE OPERATIONS**
We perform all operations **bit-by-bit**.

A = 25h = 0010 0101

1. Logical Shift Left (LSL)
Shift left, insert **0** on the right.
Original: 0010 0101
LSL: 0100 1010
Result = **4Ah**



2. Logical Shift Right (LSR)
Shift right, insert **0** on the left.
Original: 0010 0101
LSR: 0001 0010
Result = **12h**

3. Arithmetic Shift Left (ASL)
Same as logical shift left (for unsigned); may cause overflow.
Original: 0010 0101
ASL: 0100 1010
Result = **4Ah**

4. Arithmetic Shift Right (ASR)
Shift right, fill **sign bit** (MSB).
A = 25h = 0010 0101 → sign bit = **0**
Original: 0010 0101
ASR: 0001 0010 (MSB=0 shifted in)
Result = **12h**

5. Rotate Left (ROL)
Leftmost bit moves to rightmost position.
Original: 0010 0101
ROL: 0100 1010 (same as LSL, MSB=0)
Result = **4Ah**

6. Rotate Right (ROR)
Rightmost bit moves to leftmost position.
Original: 0010 0101
ROR: 1001 0010
Binary **1001 0010** = **92h**

✓ RESULTS FOR A = 25h

Operation	Binary Output	Hex Output
LSL	0100 1010	4Ah
LSR	0001 0010	12h
ASL	0100 1010	4Ah
ASR	0001 0010	12h
ROL	0100 1010	4Ah
ROR	1001 0010	92h

Now perform same operations for B = 3Bh = 0011 1011

B = 3Bh = 0011 1011

1. Logical Shift Left (LSL)
Original: 0011 1011
LSL: 0111 0110
Hex: **76h**

2. Logical Shift Right (LSR)
Original: 0011 1011
LSR: 0001 1101
Hex: **1Dh**

3. Arithmetic Shift Left (ASL)
Same as LSL.
ASL: 0111 0110
Hex: **76h**

4. Arithmetic Shift Right (ASR)
Sign bit = 0 (positive number)
Original: 0011 1011
ASR: 0001 1101
Hex: **1Dh**

5. Rotate Left (ROL)
MSB (0) becomes LSB.
Original: 0011 1011
ROL: 0111 0110
Hex: **76h**

6. Rotate Right (ROR)
LSB (1) becomes MSB.
Original: 0011 1011
ROR: 1001 1101
Binary **1001 1101** = **9Dh**

✓ **RESULTS FOR B = 3Bh**

Operation	Binary Output	Hex Output
LSL	0111 0110	76h
LSR	0001 1101	1Dh
ASL	0111 0110	76h
ASR	0001 1101	1Dh
ROL	0111 0110	76h
ROR	1001 1101	9Dh

2.4 Arithmetic Logic Shift Unit (ALSU)

The ALSU combines **Arithmetic operations**, **Logic operations**, and **Shift operations** into a single hardware unit controlled by the Control Unit (CU).

It is essentially the heart of the CPU's execution unit.

Functions of ALSU

- Addition, subtraction, increment, decrement
- Bitwise AND, OR, XOR, NOT
- Logical/Arithmetic shifts
- Rotate operations

Example: ADD Instruction Execution

Instruction: ADD R1, R2

ALSU performs:

1. Fetch R1 and R2
2. Apply addition
3. Store result in R1 (or destination register)

Example: Shift and Logic Combination

Operation:

$R3 \leftarrow (R1 + R2) << 1$

Steps:

1. ALSU adds R1 and R2
2. Shifter shifts result left (×2)
3. Result stored in R3

Real-Life Example (Image Processing in Mobile)

When your phone brightens an image:

- Addition (pixels + brightness value)
- AND/OR operations for masking
- Shifts for scaling (×2, ÷2)

ALSU performs all of this in microseconds.

MCQs

1. A register is used for:
A. Long-term storage
B. Temporary high-speed storage
C. Storing files
D. Storing OS only

2. The notation R1 ← R2 in RTL means:
A. Move R1 to R2
B. Move R2 to R1
C. Add R1 and R2
D. Swap R1 and R2

3. Which register holds the address of the next instruction?
A. IR
B. MAR
C. PC
D. MDR

4. In RTL, the command PC ← PC + 1 means:

A. Reset PC
B. Increment PC
C. Load memory into PC
D. Decrement PC

5. The bus used to carry data between CPU and memory is:
A. Control bus
B. Address bus
C. Data bus
D. Instruction bus

6. The instruction "Memory Read" means:
A. CPU writes data to RAM
B. CPU reads data from RAM
C. RAM sends address to CPU
D. Data is erased

7. In memory transfer, MAR stores:
A. Actual data
B. Address of data
C. Control signals
D. Results of ALU

8. Which micro-operation performs addition?
A. Logic operation
B. Shift operation
C. Arithmetic operation
D. Rotate operation

9. The operation R1 ← R1 AND R2 belongs to:
A. Logic micro-operations
B. Arithmetic micro-operations
C. Shift micro-operations
D. Rotate micro-operations

10. Logical Shift Left mainly performs:
A. Division by 2
B. Multiplication by 2
C. Rotate bits
D. No operation

11. Arithmetic Shift Right preserves which bit?
A. Least significant bit
B. Carry bit
C. Sign bit
D. Overflow bit

12. Rotate Left (ROL) operation moves:
A. LSB to MSB
B. MSB to LSB
C. Both sides shift
D. Deletes MSB

13. ALSU stands for:

A. Arithmetic-Level Sequential Unit
B. Arithmetic Logic Shift Unit
C. Advanced Logical Shift Utility
D. Address Logic Storage Unit

14. Which operation is *not* performed by ALSU?
A. Arithmetic operations
B. Logic operations
C. Shift operations
D. Memory read/write

15. Which bus sends commands like READ or WRITE?
A. Data bus
B. Control bus
C. Address bus
D. Power bus

✅ **ANSWER KEY**
1-B
2-B
3-C
4-B
5-C
6-B
7-B
8-C
9-A
10-B
11-C
12-B
13-B
14-D
15-B