Practical 5: To Realize Half Adder, Full Adder, Half Subtractor & Full Subtractor Using Basic Gates and NAND Gates

Adders and subtractors are digital circuits used for arithmetic operations in computers.
They are built using logic gates like **AND, OR, XOR, NOT, NAND** etc.

In this practical, we study:

- Half Adder
- Full Adder
- Half Subtractor
- Full Subtractor
  And how to implement them using **basic gates** and **NAND gates (universal gate)**.

HALF ADDER

💠 Function:

Adds two single-bit binary numbers.

Inputs: A, B

Outputs:

- **SUM** = $A \oplus B$
- **CARRY** = $A \cdot B$

(A) Realization Using Basic Gates

- SUM → XOR Gate
- CARRY → AND Gate

Truth Table:

A	B	SUM (A ⊕ B)	CARRY (A·B)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

(B) Realization Using NAND Gates

We know:

- XOR can be built using 4 NAND gates
- AND = NAND followed by NOT (which is also NAND)

Thus the whole Half Adder can be made using **only NAND** gates.

FULL ADDER

Function:

Adds three bits (A, B, Carry-in).

Inputs: A, B, Cin

Outputs:

- **SUM** = A ⊕ B ⊕ Cin
- **CARRY** = (A·B) + (Cin·(A ⊕ B))

(A) Realization Using Basic Gates

Uses:

- Two XOR gates → for SUM
- Two AND gates + one OR gate → for CARRY

Truth Table:

A	B	Cin	SUM	CARRY
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0

1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

(B) Realization Using NAND Gates

All basic operations (XOR, AND, OR) can be implemented using NAND gates only.
So Full Adder can be fully realized using **NAND-only circuit**.

HALF SUBTRACTOR

Function:

Subtracts two single-bit numbers: $A - B$

Inputs: A, B

Outputs:

- **DIFFERENCE** = $A \oplus B$
- **BORROW** = $\neg A \cdot B$

(A) Realization Using Basic Gates

- Difference → XOR gate
- Borrow → AND + NOT gate

Truth Table:

A	B	DIFFERENCE (A ⊕ B)	BORROW (¬A·B)
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

(B) Realization Using NAND Gates

- NOT using NAND
- AND using NAND
- XOR using NAND
  Thus Half Subtractor can also be formed completely with NAND gates.

FULL SUBTRACTOR

Function:

Subtracts three bits: A − B − Borrow-in

Inputs: A, B, Bin

Outputs:

- **DIFFERENCE** = A ⊕ B ⊕ Bin

- **BORROW** = (¬A·B) + (¬(A ⊕ B)·Bin)

(A) Realization Using Basic Gates

Uses:

- XOR gates
- AND gates
- OR gates

Truth Table:

A	B	Bin	DIFFERENCE	BORROW
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0

1	1	0	0	0
1	1	1	1	1

(B) Realization Using NAND Gates

As NAND gates can create all basic operations, the full subtractor can also be built using only NAND gates.