# Practical No. 15

**Title:** Implementation of Booth's Algorithm for Arithmetic Operations

**Objective:** To implement **Booth's Algorithm** for performing multiplication of signed binary numbers using shifting and addition.

**Theory:**

Booth's Algorithm is an efficient method for multiplying two signed binary numbers in **two's complement representation**. It reduces the number of addition and subtraction operations by encoding the multiplier bits effectively.

**Steps of Booth's Algorithm:**

1. **Initialize the Registers:**
    - **Multiplicand (M)** – The number being multiplied.
    - **Multiplier (Q)** – The number by which the multiplicand is multiplied.
    - **Q-1** – Extra bit used for tracking shifts.
    - **Accumulator (A)** – Stores intermediate results.
2. **Perform Booth's Encoding on the Multiplier:**
    - Read two bits at a time: **(Q0 and Q-1)**
    - Apply the following rules:
        - 00 → No operation (only shift right).
        - 01 → Add **M** to **A** (then shift right).
        - 10 → Subtract **M** from **A** (then shift right).
        - 11 → No operation (only shift right).
3. **Perform Arithmetic Right Shift (ARS):**
    - Shift the bits of **A, Q,** and **Q-1** right as a unit.
    - Preserve the sign bit during shifting.
4. **Repeat the Process for n Bits:**
    - Continue until all bits of the multiplier are processed.

**Example (Multiplication of -6 and 3 in 4-bit Representation):**

| Step | A (Accumulator) | Q (Multiplier) | Q-1 | Operation |
|---|---|---|---|---|
| 0 | 0000 | 0011 | 0 | Initial |
| 1 | 1010 | 0011 | 0 | A = A - M |
| 2 | 1101 | 0001 | 1 | Shift Right |
| 3 | 1110 | 0000 | 1 | Shift Right |
| 4 | 1111 | 0000 | 0 | Final Result (-18 in 2's complement) |

**Materials/Tools Required:**

- Microprocessor/microcontroller (e.g., 8085/8051)
- Assembler/Simulator
- Computer system with programming software
- Binary calculator (optional)

**Procedure:**

1. **Initialize Registers:**
    - Store the **multiplicand (M)** in one register.
    - Store the **multiplier (Q)** in another register.
    - Set **Q-1** to **0**.
    - Set **Accumulator (A)** to **0**.
2. **Perform Booth's Algorithm:**
    - Read the **LSB of Q (Q0)** and **Q-1**.
    - Perform the appropriate operation (Addition, Subtraction, or No Operation).
    - Apply **Arithmetic Right Shift (ARS)** on **A, Q,** and **Q-1**.
    - Repeat until all bits are processed.
3. **Store and Display the Result:**
    - The final result is stored in **A & Q registers**.
    - Convert to decimal to verify correctness.

**Observations:**

- The algorithm efficiently handles signed multiplication.

- Using Booth's encoding minimizes the number of addition/subtraction operations.
- The result is stored in two registers representing the final product.

**Conclusion:**

Booth's Algorithm is successfully implemented for multiplying two signed 8-bit numbers using **shifting and arithmetic operations**. The method optimizes performance by reducing redundant operations.

**Applications (Optional):**

- Used in microprocessors for **signed binary multiplication**.
- Applied in **digital signal processing** and **computer arithmetic**.
- Helps in **hardware implementation** of arithmetic logic units (ALUs).