Practical No. 7

Title: Shifting an 8-bit Number by One Bit

Objective: To perform a left or right shift operation on an 8-bit number.

Theory:

A **bit shift operation** moves all bits of a binary number left or right by a specified number of positions. In an 8-bit system, shifting affects how the binary number is interpreted.

Types of Shifting:

1. **Logical Shift:**
   - **Left Shift (LSL - Logical Shift Left)**: Each bit moves one place to the left, and a **0** is inserted in the least significant bit (LSB).
   - **Right Shift (LSR - Logical Shift Right)**: Each bit moves one place to the right, and a **0** is inserted in the most significant bit (MSB).
2. **Arithmetic Shift:**
   - **Left Arithmetic Shift (ASL)**: Same as logical left shift.
   - **Right Arithmetic Shift (ASR)**: Preserves the sign bit (MSB remains unchanged for signed numbers).

Example (8-bit number = 01100101)

- **Left Shift:** 01100101 → 11001010
- **Right Shift:** 01100101 → 00110010

Shifting left multiplies the number by **2**, while shifting right divides the number by **2**.

Materials/Tools Required:

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

Procedure:

1. **Initialize Registers**: Load an 8-bit number into a register (e.g., A register in an 8085 processor).
2. **Perform Shift Operation**:
   - **For left shift**, use the **SHL (Shift Left)** instruction.
   - **For right shift**, use the **SHR (Shift Right)** instruction.
3. **Store the Result**: Save the shifted number in memory or display it on an output device.
4. **Observe Carry Flag**:
   - If shifting left, the MSB is moved into the **carry flag**.

    - If shifting right, the LSB is moved into the **carry flag**.

Observations:

- Left shift increases the number’s value (multiplies by 2), while right shift decreases it (divides by 2).
- If a **1** is shifted out, it affects the **carry flag**, which may be useful in multi-byte operations.
- In **arithmetic right shift**, the sign bit is preserved for negative numbers.

Conclusion:

Shifting an 8-bit number by one bit is successfully performed using shift instructions. The operation is useful in multiplication, division, and bitwise manipulations.

Applications:

- Used in microprocessor arithmetic operations for fast multiplication and division.
- Essential in digital logic for bit manipulation and data encryption.
- Applied in embedded systems for optimizing binary computations.