Lab Report 6: Latches, Flip-Flops and Counters

V.L.S. Bhargav
Table Number: 13
Roll Number: 2025102061
Group: G17

Aim:

To study and implement the fundamental sequential circuit elements — RS latch and JK flip-flop — using CMOS logic gates, and to design a 4-bit up-down counter using Arduino. The objective is to verify their theoretical and practical behavior.

Components Required:

- Digital Trainer Kit / Breadboard
- Connecting wires and switches
- LEDs for output visualization
- IC CD4001 (Quad 2-input NOR Gate)
- IC CD4012 (Quad 4-input NAND Gate)
- Arduino UNO

PART A: RS Latch

Aim:

To assemble and test an RS latch using NOR gates and verify its truth table.

Circuit Implementation:

The RS latch is formed using two NOR gates from the CD4001 IC. The inputs S and R are connected to switches, while outputs Q and $\overline{Q}$ are connected to LEDs for observation.

Reference Circuit:

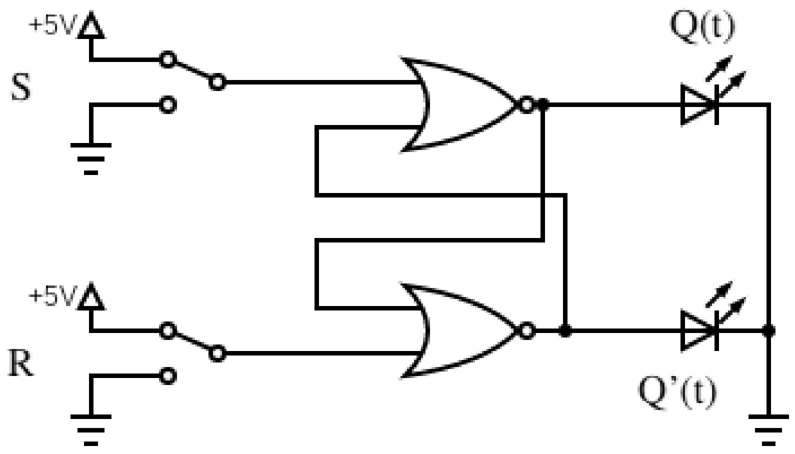
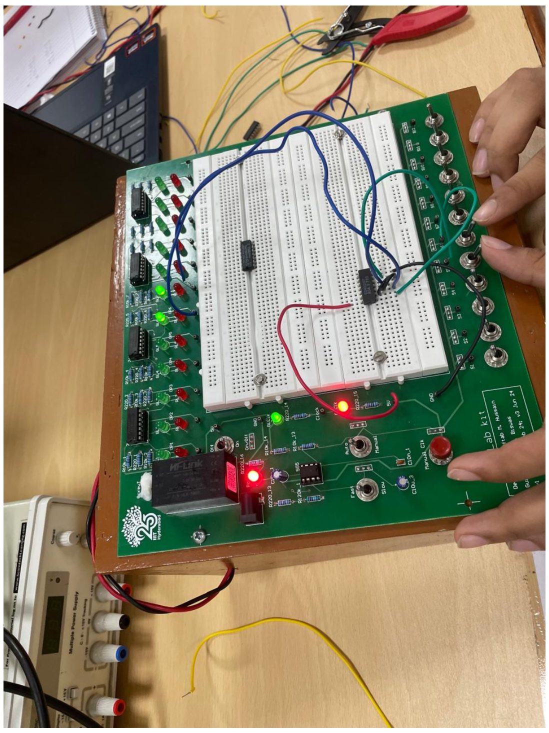


Figure 1: Reference circuit of SR Latch gates.

Procedure:

1. Assemble two **NOR gates** in a **cross-coupled configuration** on the breadboard to form the **RS latch**.
2. Connect the input switches to the **S (Set)** and **R (Reset)** terminals, and link the outputs **Q** and $\overline{\text{Q}}$ to **LED indicators**.
3. Provide appropriate **power connections (VCC and GND)** to the **CD4001 IC**.
4. Apply different **input combinations** such as (S,R) = 01, 00, 10, 00, etc., and observe the resulting output states.
5. Record the **LED responses** for Q and $\overline{\text{Q}}$ corresponding to each input combination.
6. Compare the observed results with the **theoretical truth table** of the RS latch to verify correct functionality.

Visual Circuit:


(a) First RS Latch setup.

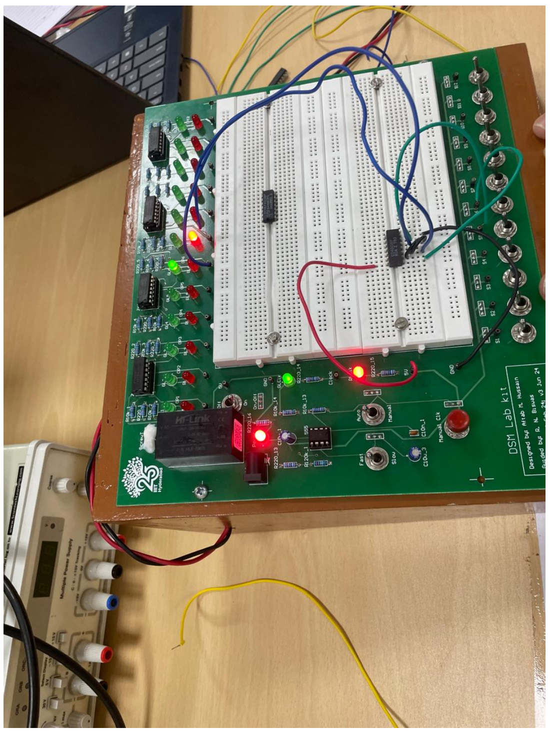

(b) Second RS Latch setup.

Figure 2: Breadboard implementations of RS Latch.

Observation:

S	R	Q (Next)	$\overline{Q}$
0	0	Memory	Memory
0	1	0	1
1	0	1	0
1	1	Invalid	Invalid

Table 1: Truth Table of RS Latch

Result and Analysis:

The NOR-based RS latch worked correctly, demonstrating memory and bistable behavior as per theory.

Circuit Simulation:

Click here to open the simulation

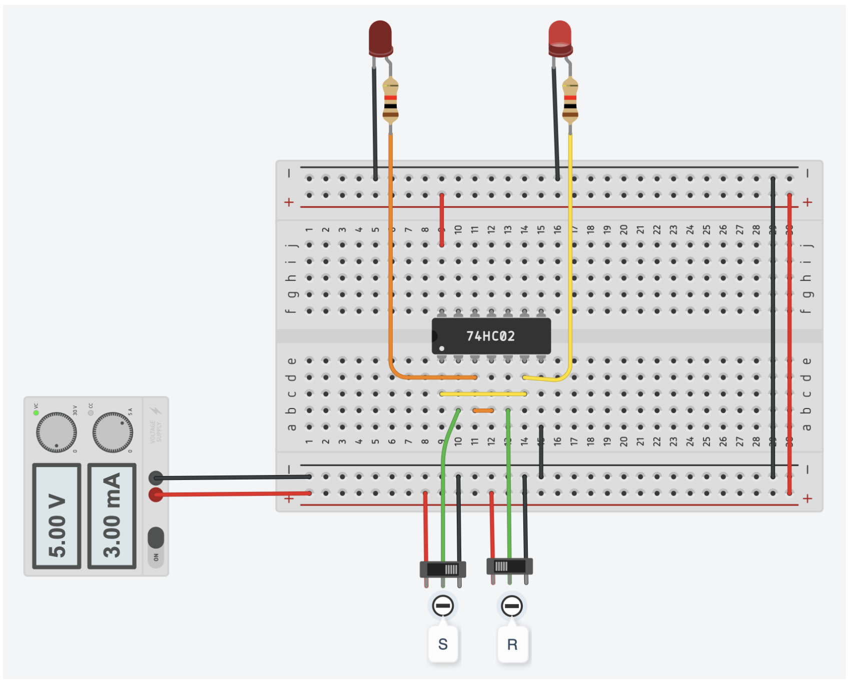


Figure 3: Simulation of SR Latch.

PART B: JK Master–Slave Flip-Flop

Aim:

To study the working of a JK master–slave flip-flop using NAND gates and verify its characteristic table.

Circuit Implementation:

The circuit is constructed using two NAND-based latches (master and slave) from IC CD4012. The clock signal controls the transfer of information from the master latch to the slave latch.

Reference Circuit:

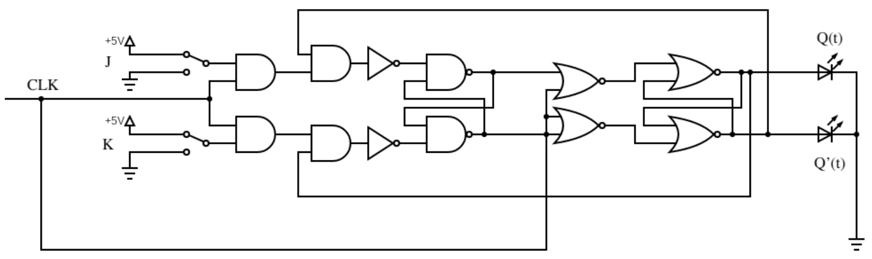


Figure 4: Reference circuit of JK Master–Slave Flip-Flop using NAND gates.

Procedure:

1. Construct two **NAND latch** stages in a **master–slave configuration** on the breadboard using the above reference circuit.
2. Connect the input terminals **J**, **K**, and **CLK**, along with the output terminals **Q** and $\overline{\mathrm{Q}}$.
3. Provide appropriate **power connections (VCC and GND)** to all ICs used in the circuit.
4. Use **LED indicators** to observe the logic states of outputs **Q** and $\overline{\mathrm{Q}}$.
5. Apply different **test combinations of J and K**, and toggle the **clock signal** to monitor state transitions.
6. Record the **output responses** for each input and clock condition, and verify them against the **truth table** of the JK flip-flop.

Visual Circuit:

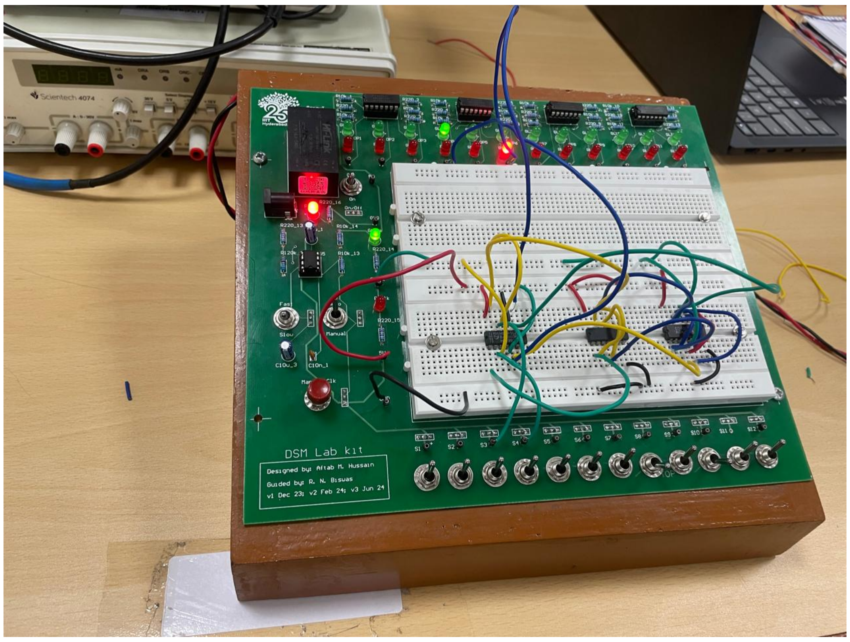

Scientech 4074
DSM Lab kit
Designed by: Aftab M. Hussain
Guided by: R. N. Biswas
v1 Dec 23; v2 Feb 24; v3 Jun 24

Figure 5: Physical circuit of JK Master–Slave Flip-Flop.

Observation:

J	K	Q(n)	Q(n+1)
0	0	Q	No Change
0	1	Q	Reset (0)
1	0	Q	Set (1)
1	1	Q	Toggle

Table 2: Characteristic Table of JK Flip-Flop

Result and Analysis:

The JK flip-flop demonstrated stable, reset, set, and toggle operations synchronized with the clock input.

Circuit Simulation:

Click here to open the simulation

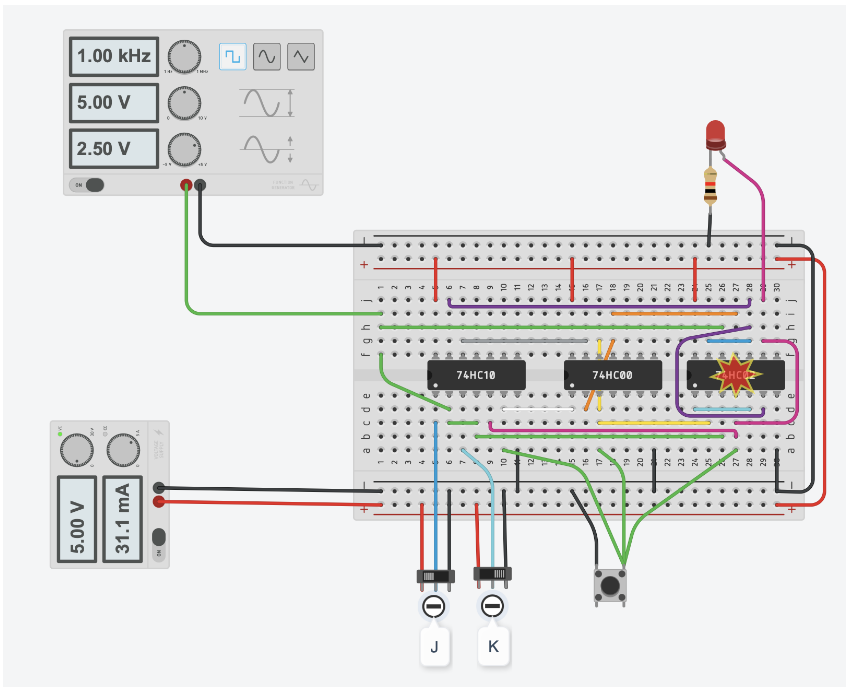


Figure 6: Simulation of JK Flip-Flop.

PART C: 4-bit Up–Down Counter (Arduino)

Aim:

To implement a 4-bit up–down ripple counter using Arduino and LEDs.

Circuit Implementation:

Four Arduino digital pins (2–5) are connected to LEDs through current-limiting resistors to represent binary output bits. The counter increments from 0000 to 1111 and then decrements back.

Reference Circuit:

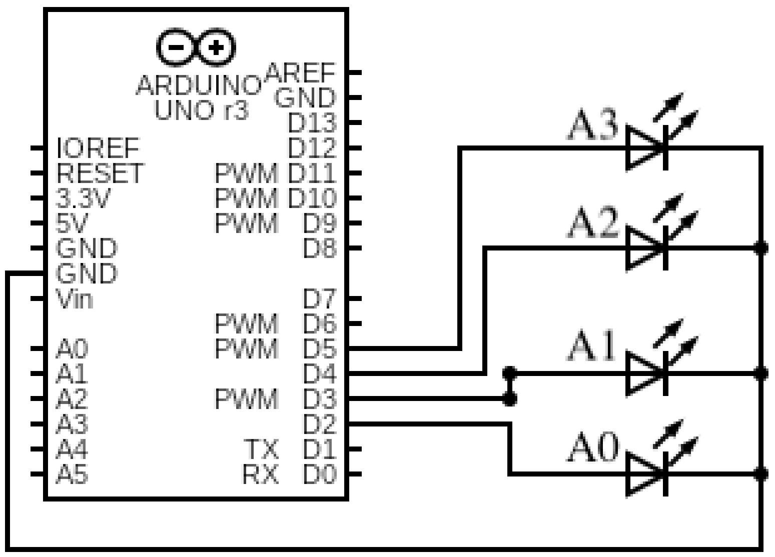


Figure 7: Reference circuit diagram of 4-bit Up–Down Counter using Arduino.

Procedure:

1. Connect four **LEDs** to Arduino digital pins **2–5** through **220Ω current-limiting resistors**, representing the 4-bit binary output.
2. Provide proper **power connections (5V and GND)** between the Arduino and the breadboard.
3. Upload the **Arduino code** implementing a 4-bit up–down ripple counter, ensuring correct pin assignments for the LEDs.
4. Observe the **LED blinking sequence** as the counter increments from **0000 to 1111** (up count) and then decrements back to **0000** (down count).
5. Verify the observed LED transitions with the expected **binary counting sequence** and record the results.

Visual Circuit:

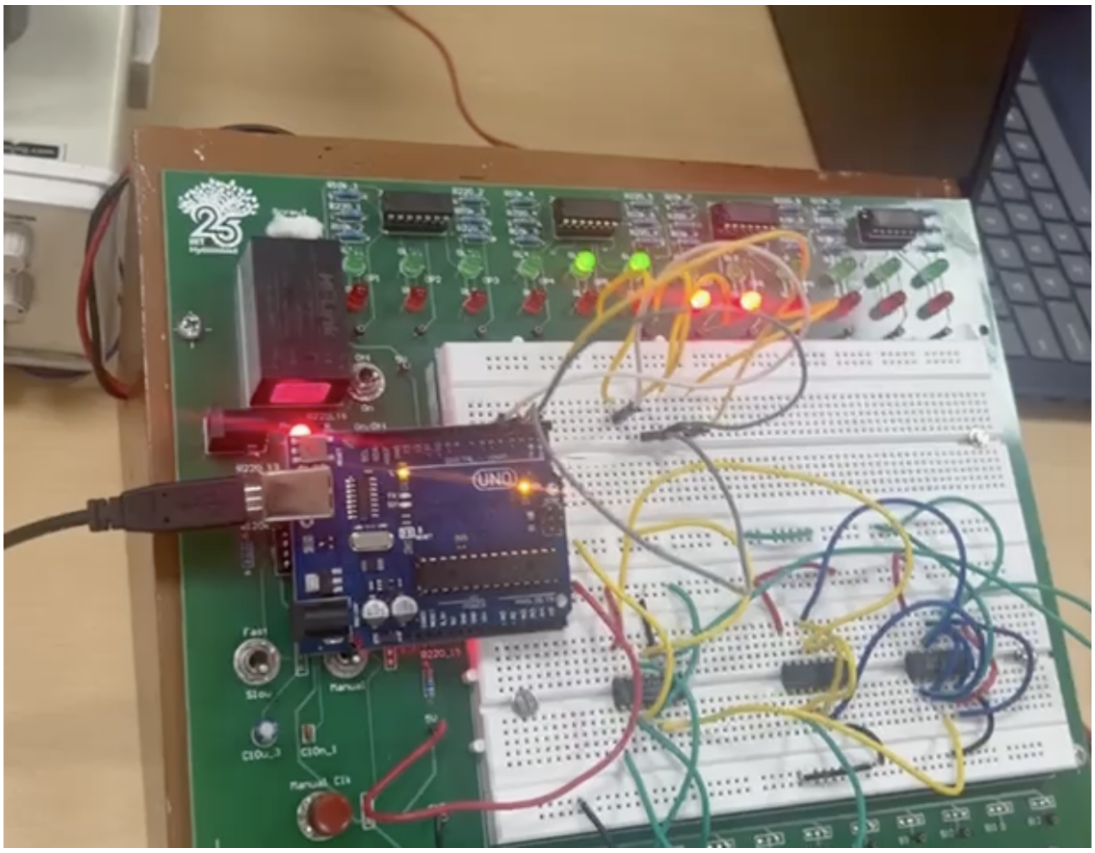


Figure 8: Physical implementation of 4-bit Up–Down Counter.

Arduino Code:

```
int ledPins[4] = {2, 3, 4, 5};
int count = 0;
bool up = true;

void setup() {
  for (int i = 0; i < 4; i++) pinMode(ledPins[i], OUTPUT);
}

void displayCount(int num) {
  for (int i = 0; i < 4; i++)
    digitalWrite(ledPins[i], (num >> i) & 1);
}

void loop() {
  displayCount(count);
  delay(1000);

  if (up) count++;
  else count--;

  if (count > 15) {
    count = 15;
    up = false;
  }
  if (count < 0) {
    count = 0;
    up = true;
  }
}
```

Listing 1: 4-bit Up–Down Counter using Arduino

Observation:

- LEDs changed in binary order, showing up and down counting.
- Speed controlled by delay.

Result and Analysis:

The Arduino-based counter correctly produced a 4-bit binary up–down count, verifying sequential logic in software.

<u>Circuit Simulation:</u>

Click here to open the simulation

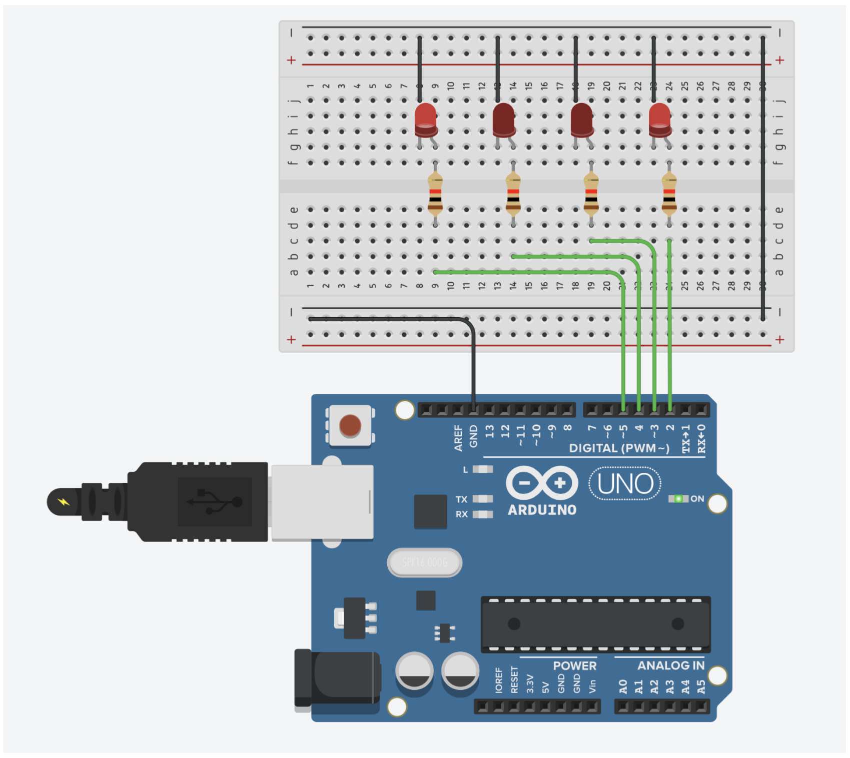


Figure 9: Simulation of ripple counter.

Final Conclusion:

- RS latch and JK flip-flop verified sequential storage principles.
- The Arduino counter confirmed up–down counting functionality.
- The experiment reinforced understanding of memory, synchronization, and counting in digital circuits.

References:

- Digital Systems and Microcontrollers Lab Manual
- CMOS Logic IC Datasheets (CD4001, CD4012)
- Arduino Reference Documentation