# Lab Report: 5

Name: Atharv Bhatt
Roll number: 2023101089
Group number: 3

## Experiment 5: Latches, Flip-flops, and Counters

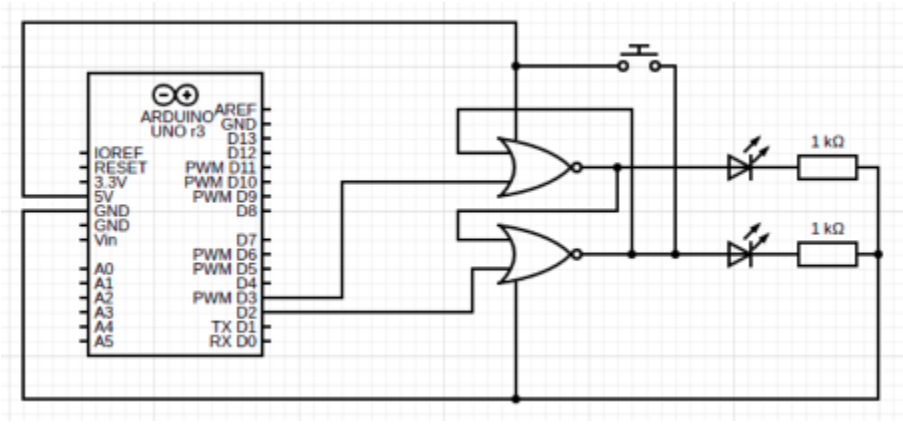
### PART – A: RS LATCH

**Aim/Objective:** Design an RS Latch using NOR gates, assemble it on a breadboard, and observe its behavior.

## Electronic Components Used:

1. Breadboard
2. Arduino Uno R3
3. 74HC32 - Quad OR gate
4. 74HC04 - Hex inverter
5. 74HC11 - Triple 3-Input AND gate
6. Jumper Wires
7. LED
8. Digital test kit
9. Resistors (1 Kohm)
10. MultiMeter

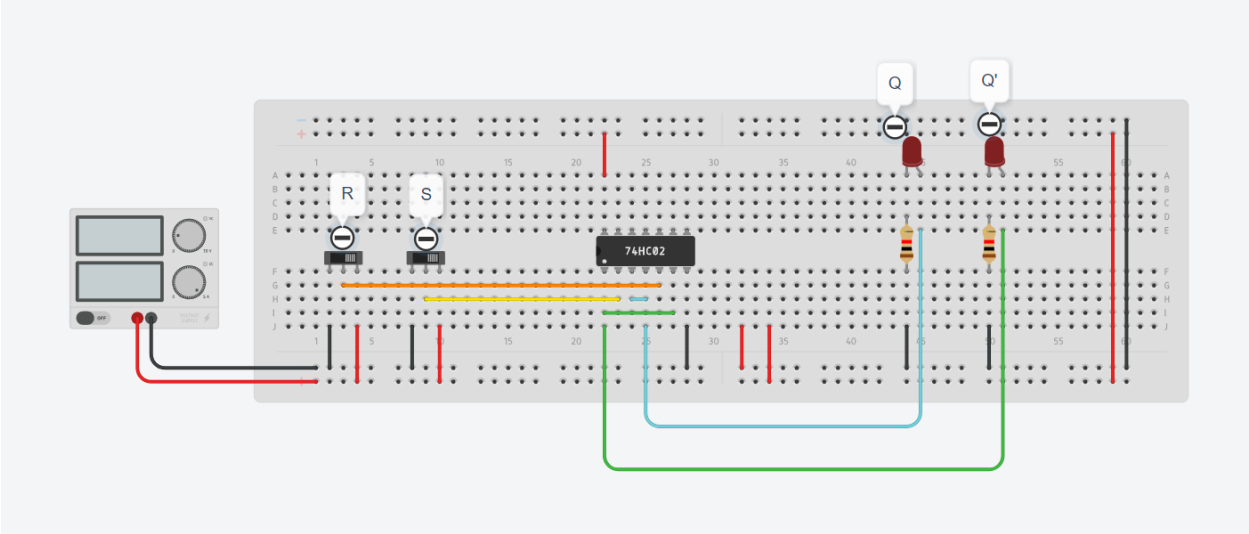
## Reference Circuit



## Procedure

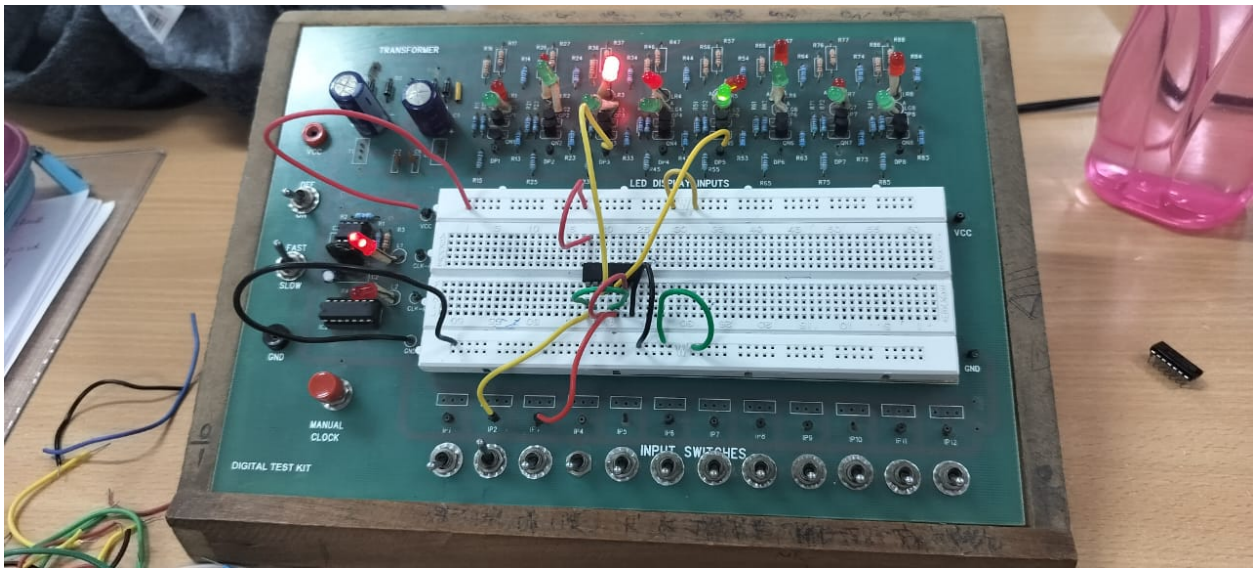
1. Connect the GND pin of the Arduino Uno R3 to the lower negative line of the breadboard.
2. Connect the 5V pin of Arduino Uno R3 to the lower live line of the breadboard.
3. Connect the lower live line to the upper live line of the breadboard.
4. Connect the lower negative line to the upper negative line of the breadboard.
5. Place a 2-Input NOR Gate, 74HC32, with its live pin connected to the upper live row and its ground pin to the lower negative row.
6. Take Multimeter readings for varying values of input logic levels.

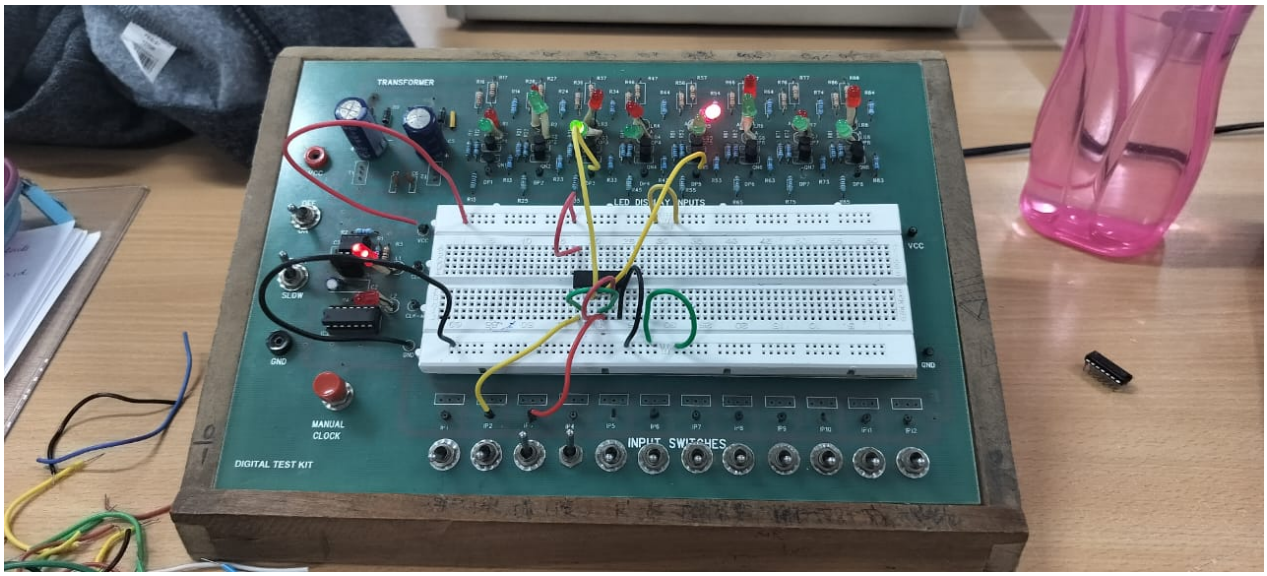
## Tinkercad




https://www.tinkercad.com/things/30R1Mp5yIGt-lab-5-rs-latch/editel?sharecode=vxa5MJpQ8rH-6iMJvT5-6SfAApja1zU20bn3ClYlCCI

## Images




## Observations:

| S | R | Q | ~Q |
|---|---|---|---|
| 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 0 |

| | | | |
|---|---|---|---|
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 |
| 0 | 0 | Unpredictable | |
| 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 0 | 0 | Unpredictable | |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 0 | 0 |
| 0 | 0 | Unpredictable | |

## Conclusion

We can make an RS LATCH using basic logic gates to understand its working and use in-memory storage. This Latch can be expected to work correctly till row number 9 as it can give different outputs.

**NOTE:** With RS latch, we get an undesired forbidden state for R=S=1 in case of RS LATCH using NOR gate and R=S=0 in case of RS LATCH using NAND gate.
We make use of JK FLIP FLOP to avoid this forbidden state.

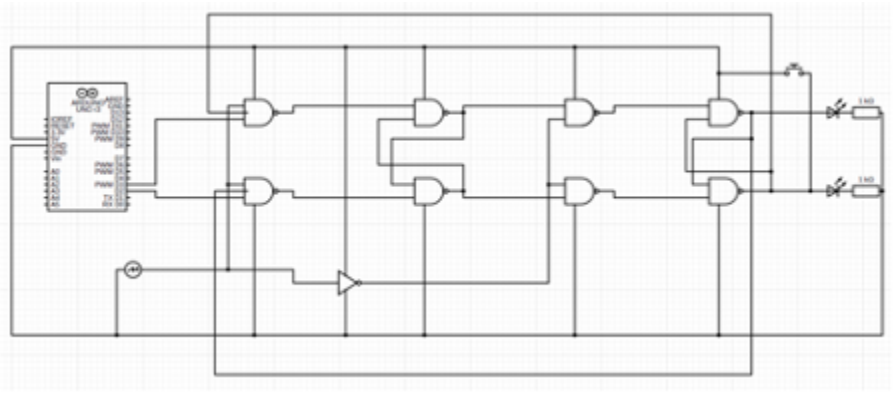
# PART – B: JK Master-Slave Flip-Flop

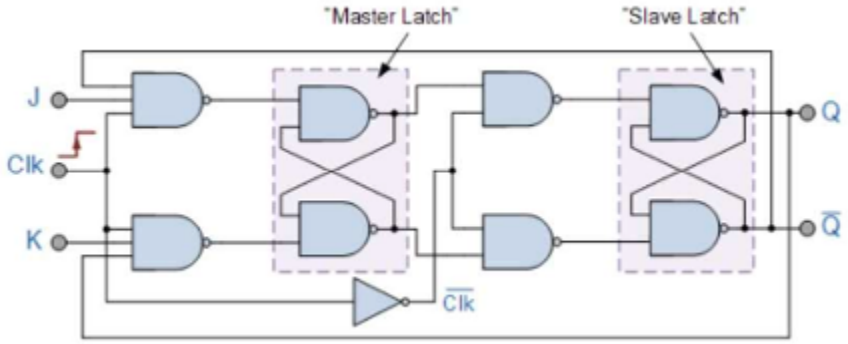
**Aim/Objective:** Design a JK Master-Slave Flip-Flop and observe its behavior.

## Electronic Components Used:

1. Breadboard
2. Arduino Uno R3
3. 74HC32 - Quad OR gate
4. 74HC04 - Hex inverter
5. 74HC11 - Triple 3-Input AND gate
6. Jumper Wires
7. LED
8. Digital test kit
9. Resistors (1 Kohm)
10. MultiMeter

## Reference Circuit

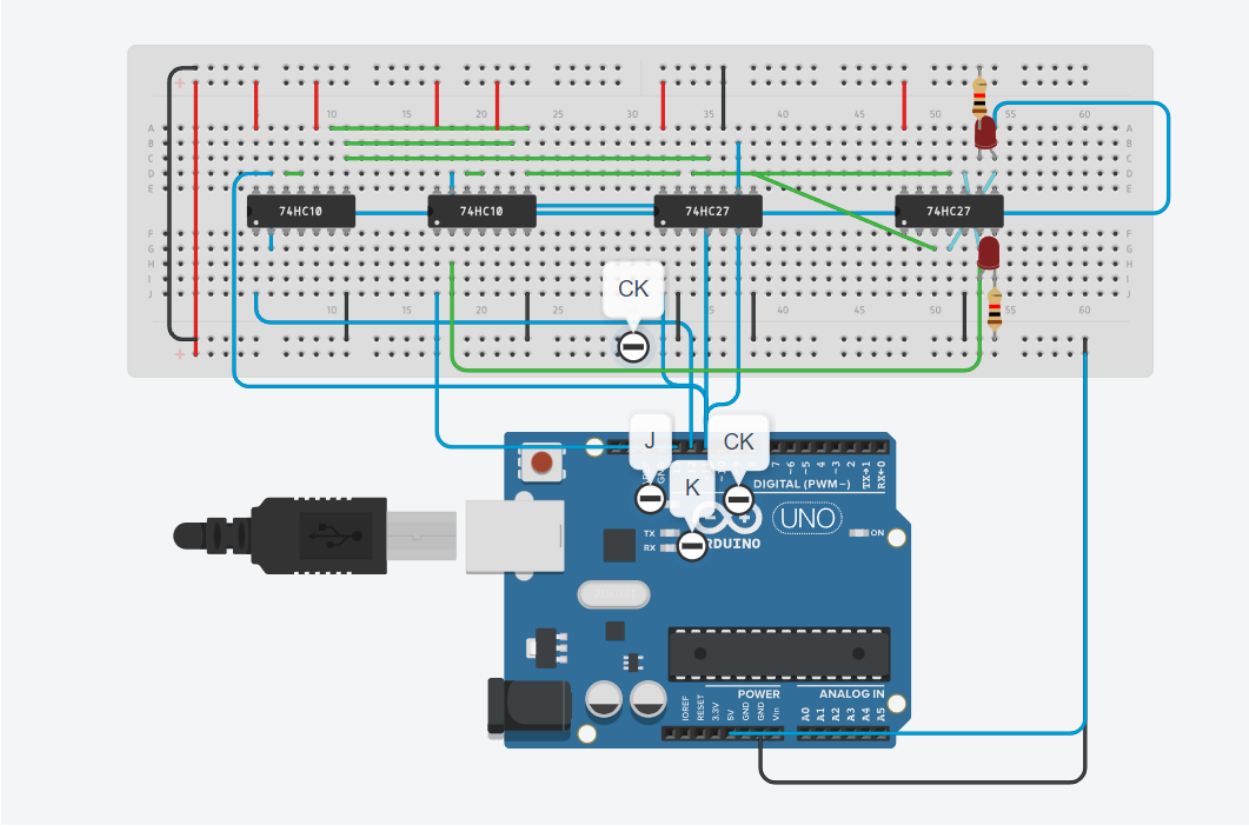




## Procedure

1. Connect the GND pin of Arduino Uno R3 to the lower negative line of the breadboard.
2. Connect the 5V pin of Arduino Uno R3 to the lower live line of the breadboard.
3. Connect the lower live line to the upper live line of the breadboard.
4. Connect the lower negative line to the upper negative line of the breadboard.
5. Place one Triple input NAND gate, 74HC10, with its live pins connected to the upper live row and their ground pins associated with the lower negative row.
6. Place two 2-Input NAND gates 74HC00 with their live pins connected to the upper live row and their ground pins connected to the lower negative row.

## Tinkercad



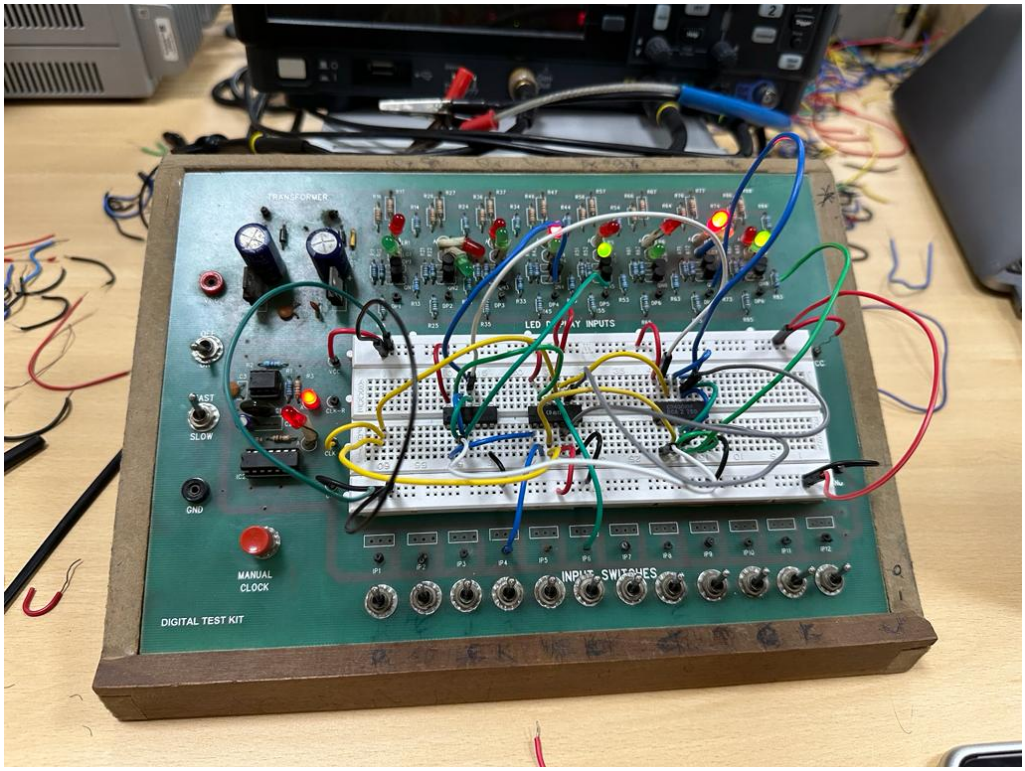

https://www.tinkercad.com/things/4WQ1sWgfaJB-jk-master-slave-flip-flop/editel?sharecode=WH9tXWrL3lfM13acBwE5EWwlNCux5A66i2fWixrtkl0

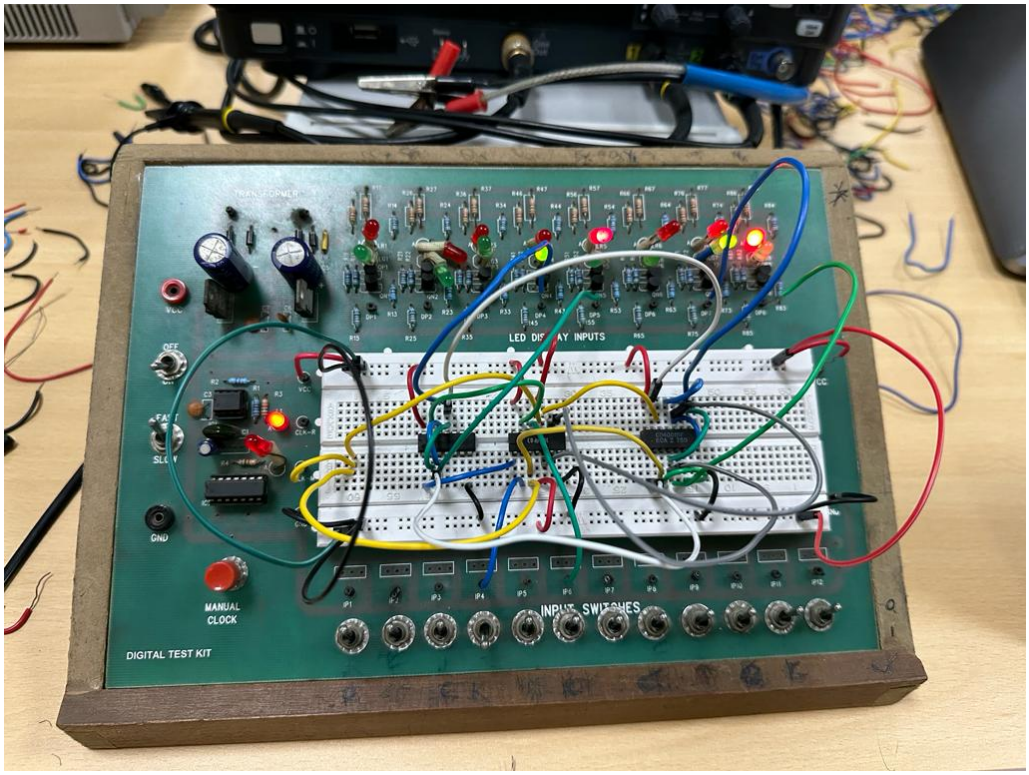
## Observations

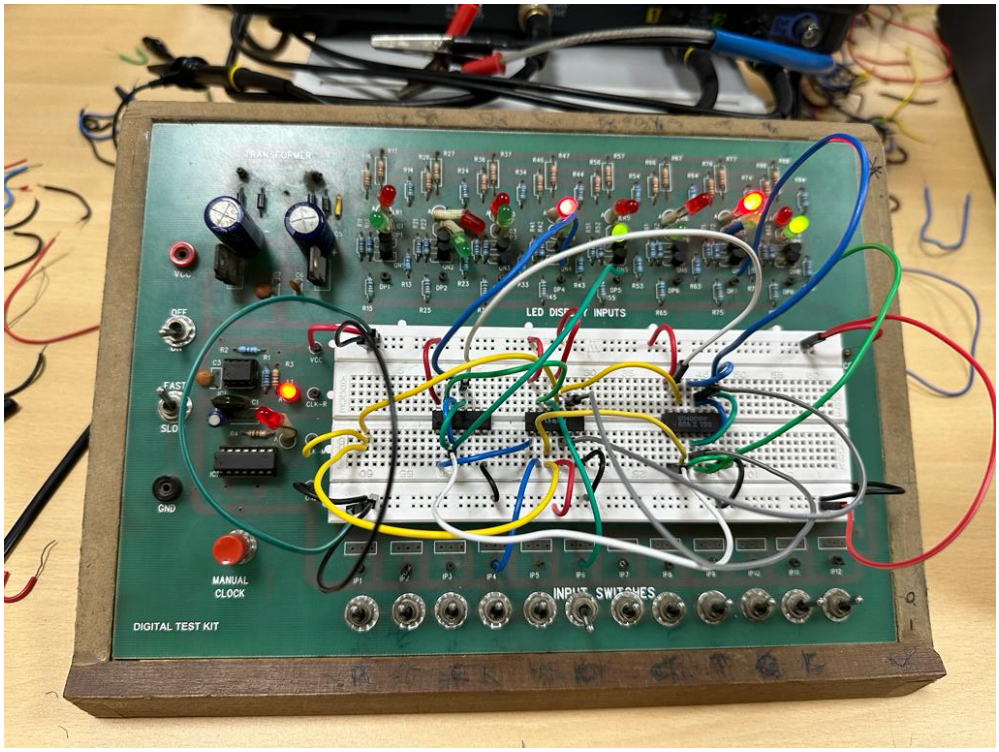
| J - K FLIP-FLOP TRUTH TABLE | | | |
|---|---|---|---|
| **J** | **K** | **ACTION** | **Qn + 1** |
| 0 | 0 | HOLD | Q |
| 0 | 1 | CLEAR | 0 |
| 1 | 0 | SET | 1 |
| 1 | 1 | TOGGLE | Q ' |

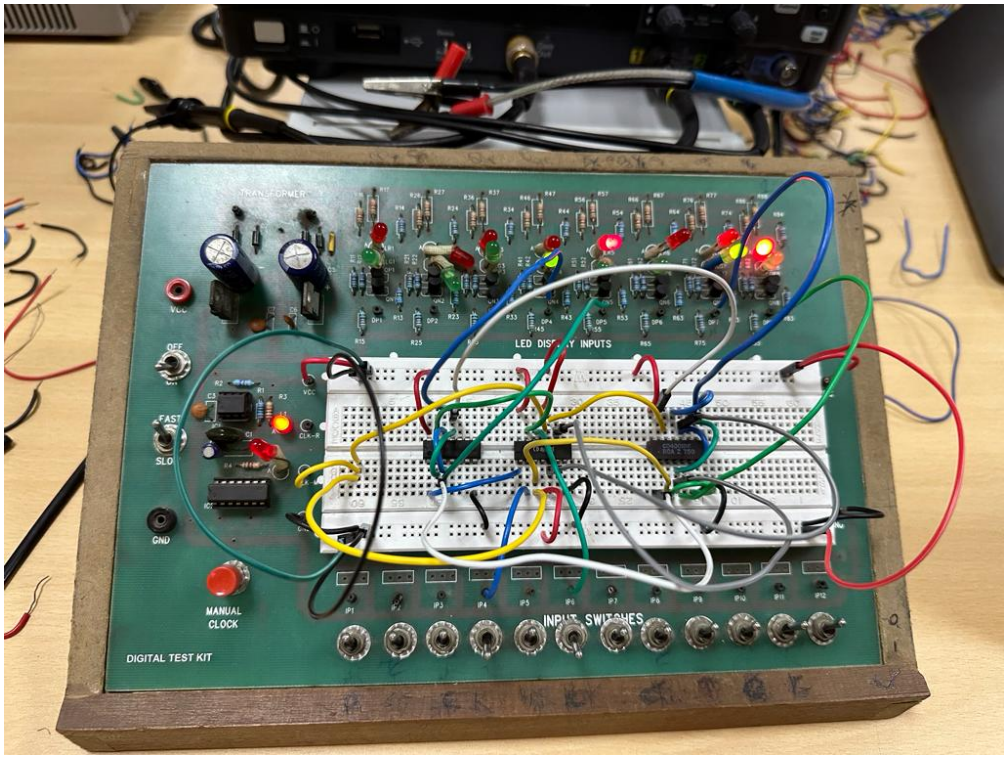
| **J** | **K** | **Q** | **~Q** |
|---|---|---|---|
| 1 | 0 | 1 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 1 | 0 |
| 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 1 |
| 1 | 0 | Toggle | |
| 0 | 1 | 1 (0) | 0 (1) |
| 1 | 0 | 1 | 0 |
| 1 | 0 | Toggle | |
| 0 | 1 | 1 (0) | 0 (1) |
| 0 | 1 | 0 | 1 |
| 1 | 1 | Toggle | |
| 0 | 0 | 1 (0) | 0 (1) |

## Images:


TRANSFORMER
LED DISPLAY INPUTS
FAST
SLOW
GND
MANUAL
CLOCK
INPUT SWITCHES
DIGITAL TEST KIT


TRANSFORMER
LED DISPLAY INPUTS
GND
MANUAL
CLOCK
INPUT SWITCHES
DIGITAL TEST KIT


TRANSFORMER
LED DISPLAY INPUTS
OFF
FAST
SLO
GND
MANUAL
CLOCK
INPUT SWITCHES
DIGITAL TEST KIT


TRANSFORMER
LED DISPLAY INPUTS
OFF
FAST
SLO
GND
MANUAL
CLOCK
INPUT SWITCHES
DIGITAL TEST KIT

## Conclusion

We can build a JK FLIP FLOP using basic logic gates and understand its working and its advantages over RS LATCH.

**ADVANTAGES OF JK FLIP FLOP OVER RS LATCH:**

- We can overcome the unpredictable state ( occurs when both R=S=1 ) in RS LATCH using a clock in JK FLIP FLOP.
- We can manually change the clock so that we can make a change at any desired time, but in an RS LATCH, we cannot make a change at the desired time.
- Because in JK FLIP FLOP, output would be changed only when the state of clock changes as in RS LATCH, outputs would be changed just after the change in inputs.

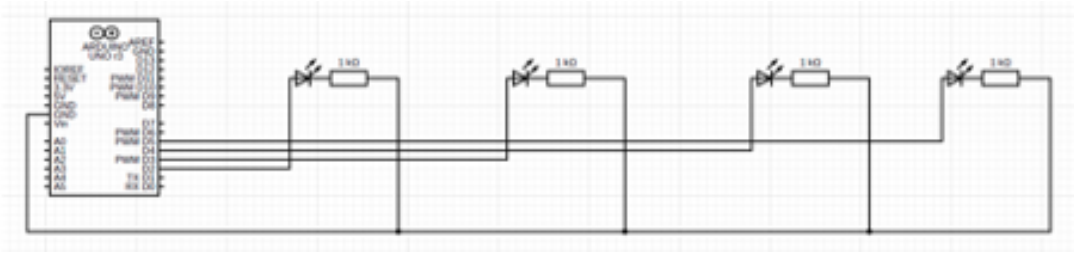
# PART – C: 4-bit Up-Down Counter

**Aim/Objective:** Build a 4-bit Up-Down Counter using an Arduino.

## Electronic Components Used:

1. Breadboard
2. Arduino Uno R3
3. LEDs
4. Jumper Wires

## Reference Circuit



## Procedure

1. **Setup the Hardware:**
    - Connect the GND (Ground) pin of the Arduino Uno to the lower negative line of the breadboard.
    - Connect the 5V pin of the Arduino Uno to the lower live line of the breadboard.
    - Ensure that the lower live line is connected to the upper live line of the breadboard, forming a continuous power rail.
    - Similarly, connect the lower negative line to the upper negative line to form a continuous ground rail.
    - Place four LEDs on the breadboard, each in separate rows, making sure the longer leg (anode) of the LEDs is connected to the upper positive rail, and the shorter leg (cathode) is connected to individual pins on the breadboard.

2. **Connect Arduino Pins:**

- Connect four digital output pins of the Arduino (e.g., pins 10, 11, 12, and 13) to the respective cathodes of the LEDs. Use jumper wires for these connections.

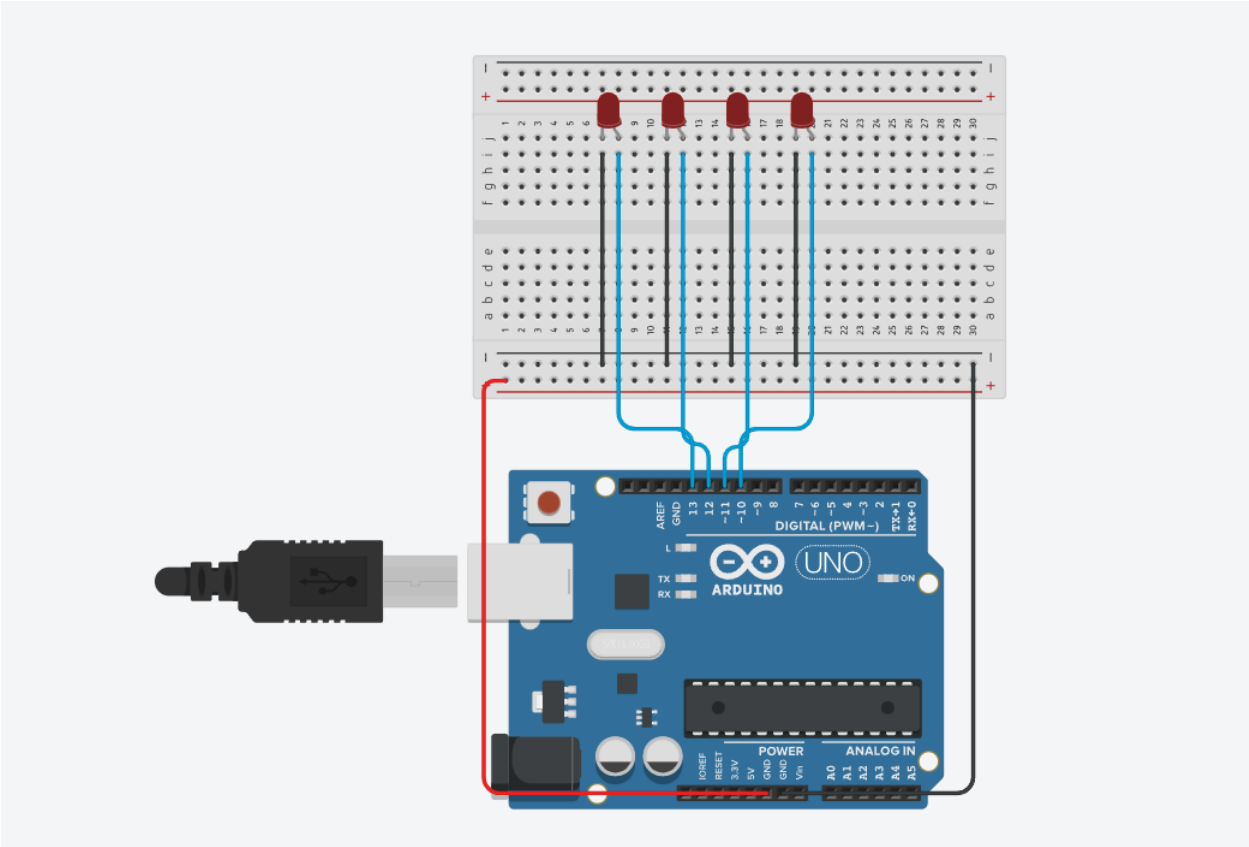
## 3. Write and Upload the Arduino Code:

- Write an Arduino sketch for the 4-bit Up-Down Counter. The code should include instructions for counting up from 0 to 15 and then counting down to 0 in a continuous loop.
- Initialize a Timer object to control the counter's timing.
- Utilize the Timer library to handle the oscillation of LEDs to represent the count.
- Use appropriate digitalWrite statements to control the state of LEDs, simulating binary counting.
- Implement a mechanism to detect when the counter reaches the maximum value (15) and switch to counting down.
- To implement the down counter, use if conditions to change the counter's direction when it reaches the maximum count.
- The complete code should toggle the LEDs accordingly to represent the binary count, and the timer should manage the counter's speed.
- After writing the code, upload it to the Arduino Uno.

## 4. Running the Experiment:

- Power up the Arduino Uno by connecting it to your computer or a power source.
- Observe the LEDs on the breadboard. They should cycle through the binary count, moving up from 0 (0000) to 15 (1111), then counting down from 15 back to 0.
- The counting sequence should continue until you manually stop the Arduino or disconnect the power source.

## Tinkercad

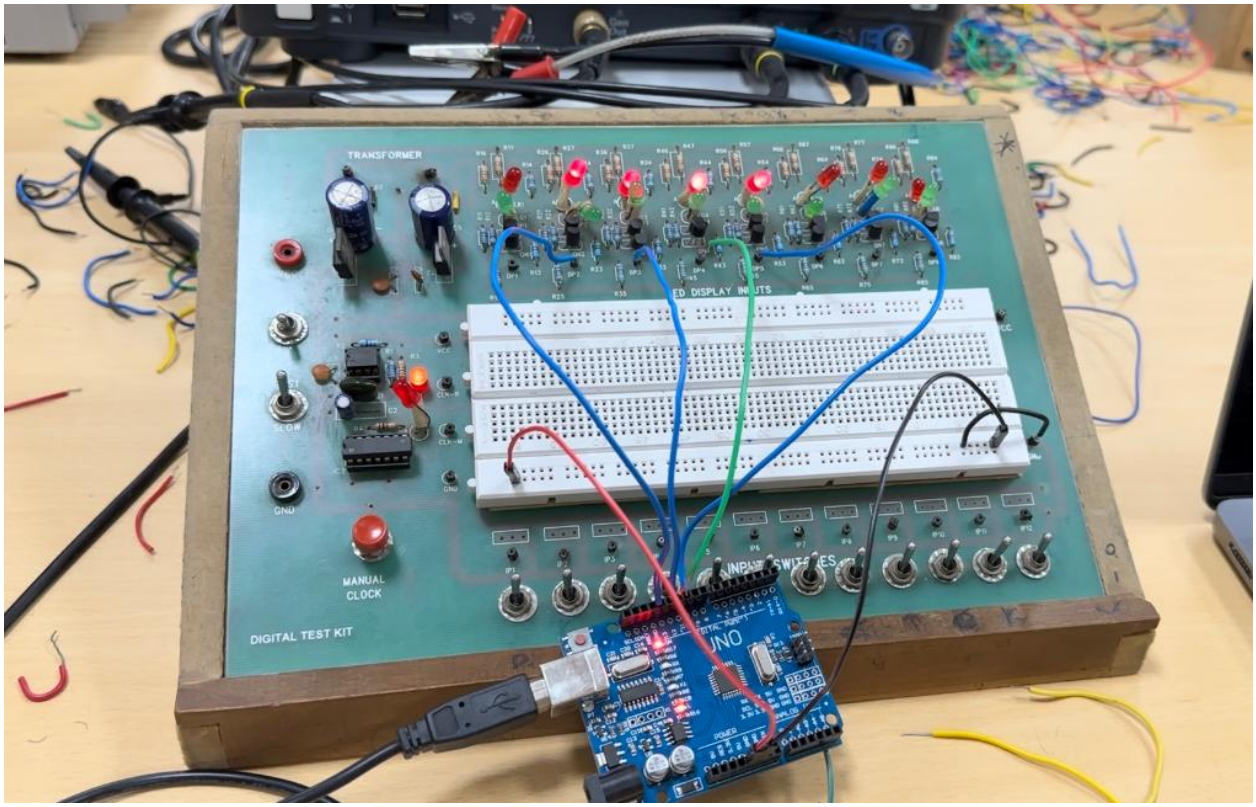



https://www.tinkercad.com/things/baI30rRQ90z-4-bit-up-down-counter/editel?sharecode=45L-KAUnpVRca-2o_1PEcREd-XDWFHJZNM81qCkp5FY

## Observations:

The frequency of the most significant bulb is the least and for the others, it becomes double the previous and due to this, the ripple counter first goes UP from 0 (0000) to 15 (1111), then goes DOWN from 15 to 0, then goes UP, and this cycle repeats until the simulation is stopped.

## Images:




## Video of  4-bit Up-Down Counter

https://drive.google.com/file/d/18fuo0CdzJARtq9U13GBB4VdOM82ZyMMq/view?usp=share_link

## Conclusion

We verified that the 4-bit Up-Down Counter functions as expected. It should count in binary from 0 to 15 and then back to 0, effectively demonstrating the desired up-down counting behavior.