# Experiment 07

Aditya Tripathy - ee24btech11001, Durgi Swaraj Sharma - ee24btech11018

**Abstract**
In Experiment-07, we used T Flip-Flops(FFs) to make a Mod-7 Asynchronous Counter, with the help of a NAND gate.

## 1 Objective

To design a Mod-7 Asynchronous Counter using T Flip-Flops and demonstrate its performance using a Cathode Ray Oscilloscope with a clock from an Arduino.

## 2 Apparatus

- 3x T Flip-Flops [we had no access to T Flip-Flops, so we used JK Flip-Flops HD74LS76AP with appropriate changes to make them function like a T FF.]
- 3x LEDs
- 3-input NAND gate [we used DM74LS10]
- Oscilloscope
- Arduino
- DC power supply
- Connecting wires and probes

## 3 Theory and Circuit

Counters are sequential circuits that store the number of times an event has occured. They can have a sequence, like counting in binary numbers. Up counters count in incrementing fashion, while down counters count in decrementing fashion. They can be made using Flip-Flops and a clock signal.

In Asynchronous Counters, only the first FF gets the clock signal, and the following FFs are driven by the outputs of the previous FFs. The FFs are cascaded, and the changes propagate in a Ripple fashion.

Using T FFs, asynchronous counters can be contructed by using one FFs output $Q$ as the clock signal for the next, while maintaining $T$ at HIGH. If the FFs are positive edge triggered, this design will be a down counter, and if the FFs are negative edge triggered, this desgin will be an up counter. Up and down counters can be changed to their opposite types by inverting the output signal of the FFs before it is used to drive the next FF.

We want to design a Mod-7 counter wil the T FFs. That means we want our counter to count between the numbers possible from a mod-7 operation: 0,1,2,3,4,5,6. The minimum number of FFs needed for this would be 3, with which we will be able to represent numbers from 0-7. But we want the counter to set to 0 after 6, instead of the default behaviour of counting to 7 and then setting to 0. For this functioning we shall employ the use of the $\overline{clr}$ pins of the FF, when when set to a LOW will set the FF output to 0. Note that $\overline{pre}$ should remain at a HIGH throught our circuit and operation, and $\overline{clr}$ should only go LOW when the counter reaches 7, or 111 in binary. This condition is checked using a NAND gate to perform the operation $(Q_0Q_1Q_2)'$ and using this as the $\overline{clr}$ signal.

Based on these ideas, the circuit can be realized with positive edge triggered FFs as follows.

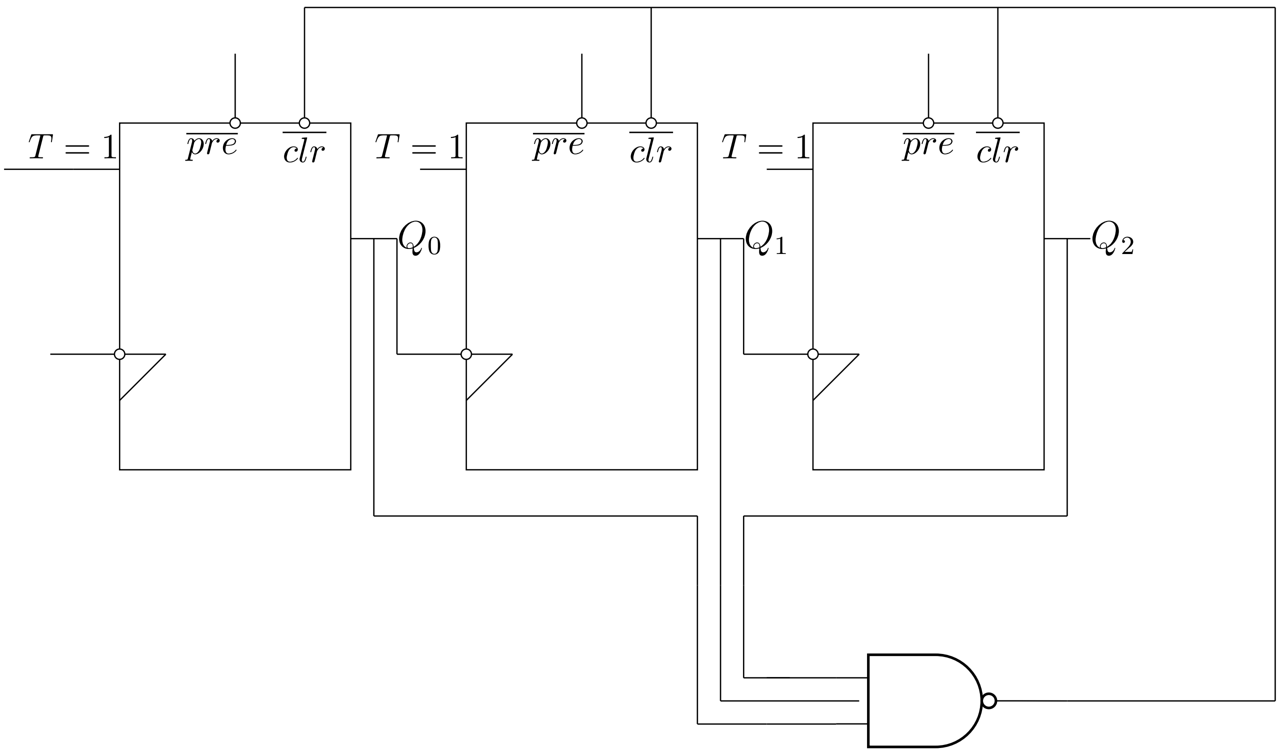


Figure 1: Mod-7 counter

# 4 Procedure

## 4.1 The Circuit

Before we even begin building the circuit, we should be aware of the specifications of the ICs we're working with. For that we should refer to the datasheets.

In our case, the HD74LS76AP is a negative edge triggered JK Flip-FLop, and that means we don't need to use a signal inverter before we use the outputs as the next FF's clock signals for an up counter implementation. It otherwise functions like any JK Flip-Flop, and to use it as a T FF we just need to use both the J and K pins as T (J=T, K=T), and set them to HIGH to use as a counter.

- Connect the J K pins of the three FFs to HIGH(5V).

The DM74LS10 has multiple simple 3-input NAND gates of which we need to use only one for the *clr* condition.

- Connect the outputs of the three FFs $Q_0$, $Q_1$, $Q_2$ to three inputs of the NAND gate. (Make sure you're using the same NAND gate on the IC)
- Connect the output of the NAND gate to the $\overline{clr}$ pins of the three FFs.
- Set the $\overline{pre}$ pins of the FFs to HIGH.
- Connect GND to ground, and $V_{cc}$ to the appropriate power voltages for the ICs as given in the datasheets.

We shall use an Arduino to generate the clock signal. Runnning the following code on an Arduino gives us a 1Hz square wave signal from pin D13.

```
#include <Arduino.h>

void squareWave()
{
  digitalWrite(13, 1);
  delay(500);
  digitalWrite(13, 0);
  delay(500);
}
void setup() {
    pinMode(13, OUTPUT);
}

void loop() {
    squareWave();
}
```

- Connect the D13 pin of the Arduino to the clock(clk) pin of the first FF.
- Because our FFs are negative edge triggered, just connect the FF outputs to the clk pins of the next FFs directly.

## 4.2 Visualizing the Operation

We will use 3 LEDs to visualize the outputs of the FFs. The LEDs will light up when the output signal is high, helping us visualize the bits of the counter.

- Connect the longer pins of the LEDs to the output pins of the Flip-Flops, and the other pin to ground

The FFs have enough internal resistance to not burn the LEDs.

To study the signals over time, we shall use a dual-channel oscilloscope.

- Connect one of the probes to measure the arduino clock signal
- Connect the other probe to measure the output signal of the first FF.
- Observe the signals on the oscilloscope, and repeat for the other two output signals

# 5 Observations

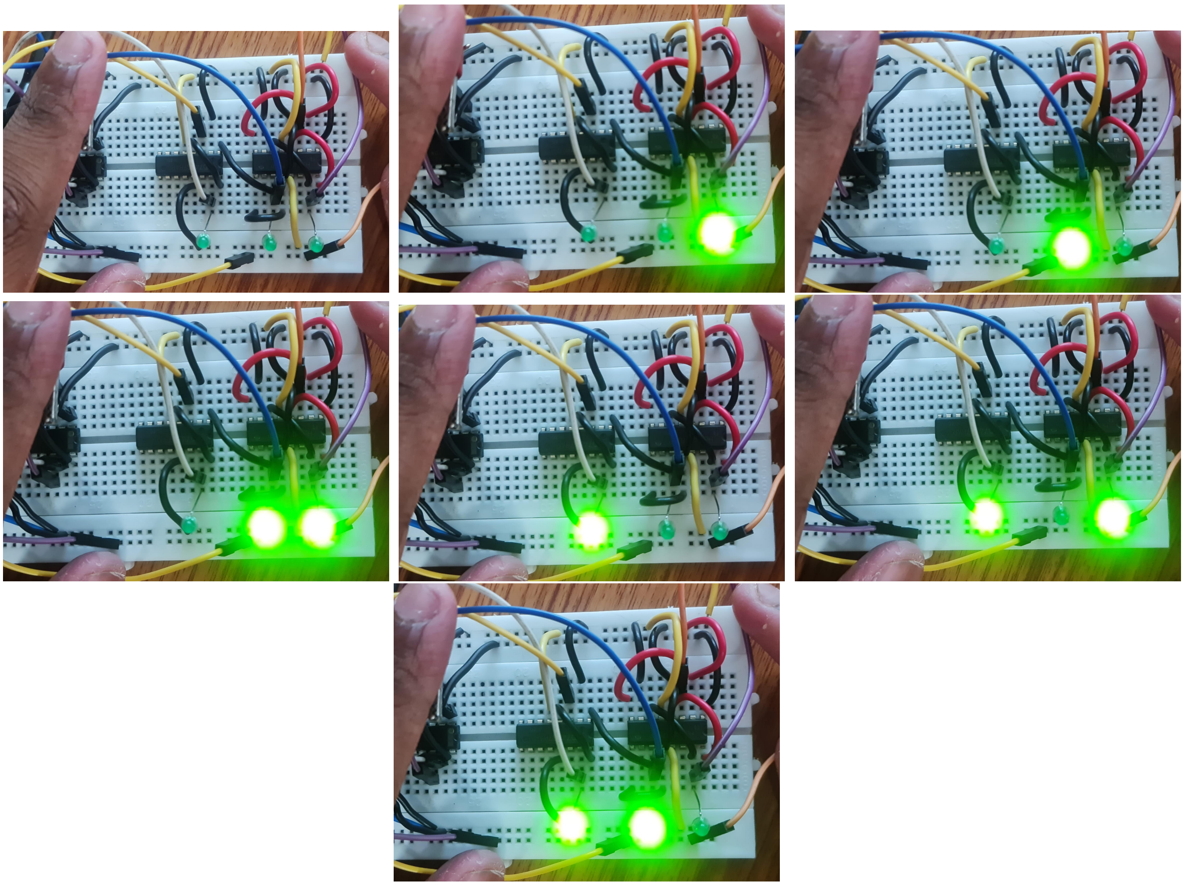

Figure 2: Numbers 0-6 represented on the LEDs in binary, bit of least significance on the right.

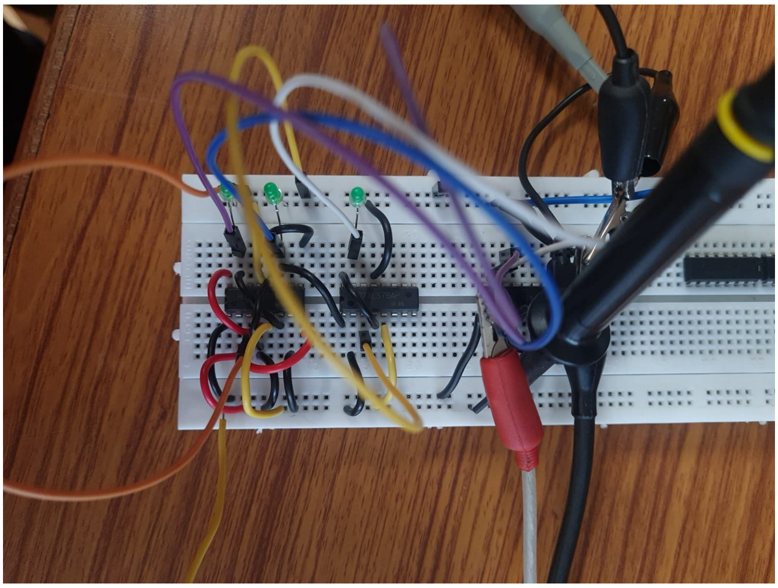

Figure 3: The Circuit

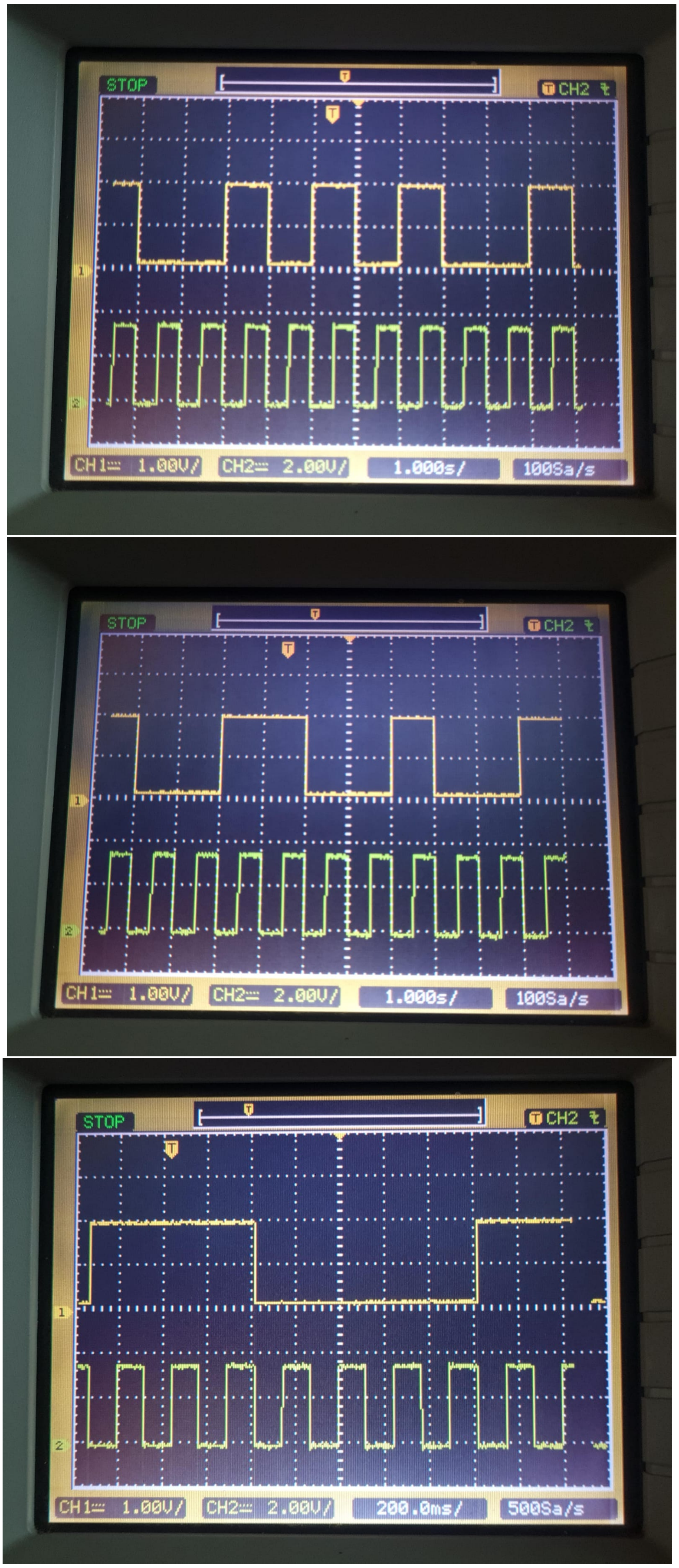


Figure 4: The signals $Q_0$, $Q_1$, $Q_2$ plotted with the initial clock signal, respectively.

As shown in the images above, we get the following state sequence in our circuit.

| **Count** | **Q2** | **Q1** | **Q0** | **Decimal Equivalent** | **LED State** |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | Off-Off-Off |
| 1 | 0 | 0 | 1 | 1 | Off-Off-On |
| 2 | 0 | 1 | 0 | 2 | Off-On-Off |
| 3 | 0 | 1 | 1 | 3 | Off-On-On |
| 4 | 1 | 0 | 0 | 4 | On-Off-Off |
| 5 | 1 | 0 | 1 | 5 | On-Off-On |
| 6 | 1 | 1 | 0 | 6 | On-On-Off |
| 7 | 0 | 0 | 0 | 0 (Reset) | Off-Off-Off |

Table 1: State Sequence of our Mod-7 Counter, as shown on the LEDs

# 6 Conclusion

The sequential circuit functioned like a mod-7 counter, with the Arduino providing clock signal.