LAB REPORT-5

Name:K.Lakshmi Nirmala

Roll Number:2021101126

Group Number: G-6

PART-A

SR Latch

Aim/Objective of the experiement:

To built a SR latch using NOR gates

Electronics components used:

1)Breadboard.

2)Ardunio Uno R3

3)74HC02 IC(NOR gates)

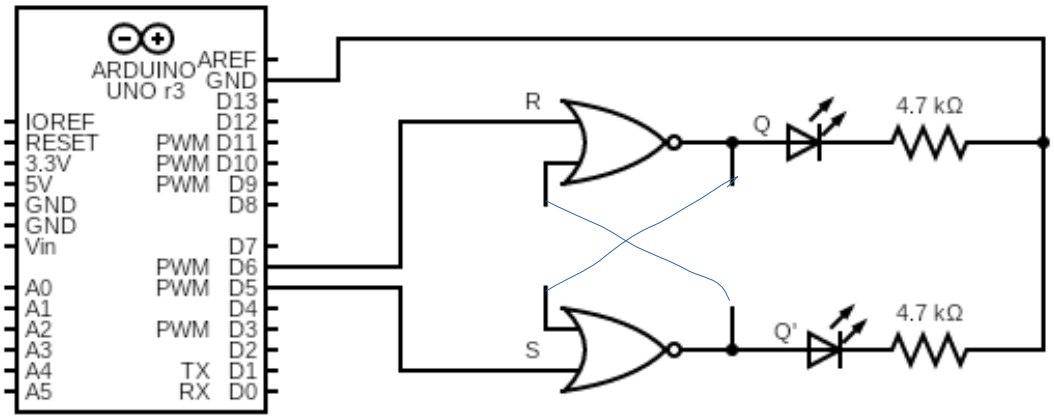
4)2-leds

5)3-Resistors.

6)Pushbutton.

7)Wires.

REFERENCE CIRCUIT:



PROCEDURE:

1)Take breadboard, take two 74HC02 Ics place it on the breadboard.

2)Give connections to the vcc and ground from ardunio to breadboard.

3)Set up the circuit in the tinkercad as shown in the reference circuit diagram mentioned above.

4)After completing the connections give the code for the ardunio to work the circuit.

5)**CODE:**

```
int pin1=6;

int pin2=5;

int S,R,r;

void setup()

{

   pinMode(pin1,OUTPUT);

   pinMode(pin2,OUTPUT);

   Serial.begin(9600);

}

void loop()

{
```

```
Serial.print("\S=");
while(Serial.available()==0){}
S=Serial.read();
S=S-'0';
Serial.println(S);

Serial.print("\R=");
while(Serial.available()==0){}
R=Serial.read();
R=R-'0';
Serial.println(R);

digitalWrite(pin1,R);
digitalWrite(pin2,S);

Serial.print("Press any key to enter new values\n");
while(Serial.available()==0){}
r=Serial.read();
delay(1000);
```

}

6)Start simulation after completing the code part and give inputs for the S and R from the serial monitor.

7)Observe the outputs and make a truth table for the SR latch.

CONCLUSION:

1)Truth table for the SR latch.

TRUTH TABLE FOR SRLATCH			
INPUTS		OUTPUTS	
S	R	Q	Q’
0	1	0	1
0	0	0	1
1	0	1	0
0	0	0	0
0	1	0	1
1	0	1	0
0	1	0	1
0	0	0	1
1	1	0	0
0	0	0	0
1	0	1	0
1	1	0	0
0	0	0	0
0	1	0	1
1	1	0	0
0	0	0	0

2)Observation from the SR latch truth table

i)If we give both S and R as zero latch was giving previous outputs irrespective to the prersent input(both zeroes).

ii)If we give both inputs S and R as one latch was giving output and finally it was goes to zero.

iii)From this, we conclude that SR latch works correctly when we give inputs 01,10 and 00. In SR latch using NOR gates inputs 1and1 are invalid state.

3)In this Experiement we learnt about how to built SR latch and their functioning.

LINK FOR THE TINKERCAD SIMULATION:

https://www.tinkercad.com/things/lC7iFRzdonQ-lab-5-sr-latch/editel?sharecode=LoCwc_BWLnGkS0U7t3J7o-HSEpHO8GWNF4jf1QstHq0

PART-B

JK MASTER SLAVE FLIP-FLOP

AIM/OBLECTIVE OF THE EXPERIEMENT:

To built and observe JK Master-Slave Flip-Flop

ELECTRONIC COMPONENTS USED:

1)Breadboard

2)Ardunio Uno R3

3)Two 74HC00 Ics

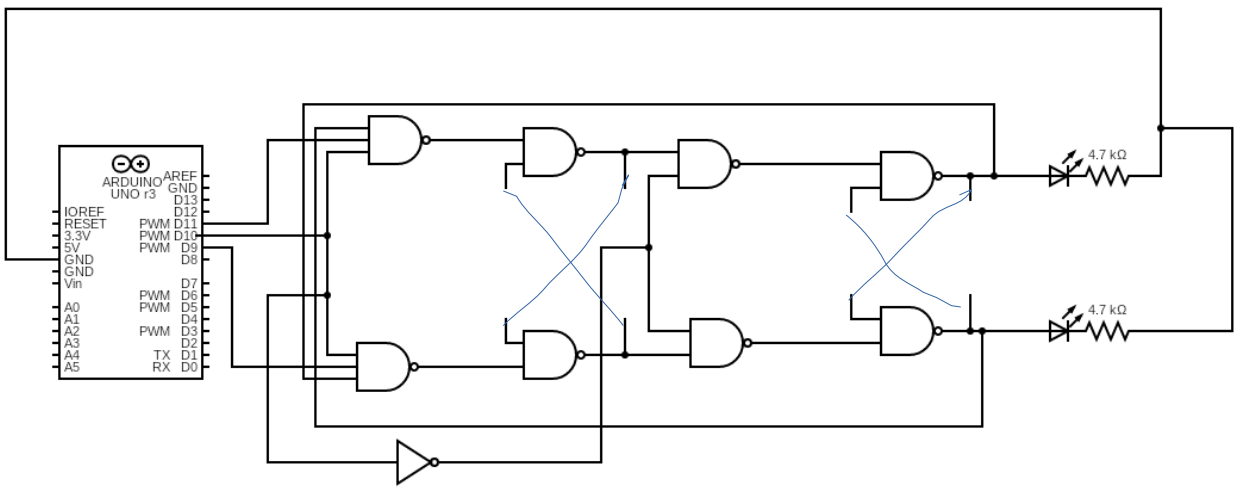
4)One 74HC04 IC

5)Push button

6)2 Resistors

7)2 LED’s

REFERENECE CIRCUIT:



PROCEDURE:

1)Take a breadboard,take TWO 74HC00 Ics and 74HC02 ic and 74HC04 ic and place it on the breadboard.

2)Give ground and power conncetions to the breadboard from ardunio.

3)Give connections to the Ics as shown in the reference circuit as mentioned above.

4)After completeing the connections give code to the ardunio in the code part in tinkercad.

5)Make sure that you have to give connections from clock as all times on for this experiement.

6)NOw enter the code and take readings from the serial monitor.

5)CODE:

```
int pin1=11;

int pin2=9;

int pin3=10;

int J,K,r;

void setup()

{
```

```
    pinMode(pin1,OUTPUT);
    pinMode(pin2,OUTPUT);
    pinMode(pin3,OUTPUT);
    Serial.begin(9600);
}
void loop()
{
  Serial.print("\J=");
  while(Serial.available()==0){}
  J=Serial.read();
  J=J-'0';
  Serial.println(J);

  Serial.print("\K=");
  while(Serial.available()==0){}
  K=Serial.read();
  K=K-'0';
  Serial.println(K);
```

```
  digitalWrite(pin1,J);
  digitalWrite(pin2,K);
  digitalWrite(pin3,HIGH);


  Serial.print("Press any key to enter new values\n");
  while(Serial.available()==0){}
  r=Serial.read();
  delay(1000);
}
```

6)Observe the working of the JK Master-Slave Flip-Flop by giving the outputs in serial monitor.

7)Tabulate the obtained outputs for the respective inputs in a truth table.

8)Observe the working functioning of JK Master-Slave Flip-Flop.

CONCLUSION:

1)JK Master Slave flipflop is nothing but its a combination of two latches one is master latch and another is slave latch.

2)From the given inputs we got these outputs in JK Master-Slave Flip-Flop:-

INPUTS		OUTPUTS	
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	1	Toggling	
0	0	0(undefined)	1(undefined)
1	0	1	0
1	1	Toggling	
0	0	0(undefined)	1(undefined)
0	1	0	1
1	1	Toggling	
0	0	0(undefined)	1(undefined)

3)From this we observe that JK Master slave flip flop toggles when we give both inputs as 1 and gives previous output when we give both inputs as 0.

4)In this Experiement we learnt how to built the JK Master slave flip flop .

LINK FOR THE TINKERCAD SIMULATION:

https://www.tinkercad.com/things/cj0SIXNGqOo-jk-master-slave-flipflop/editel?sharecode=OFp_oa5OPEUWJzIlMjTqI_Xpes_h9OLsv-9DXw0cLAg

PART-C

4-BIT UP DOWN COUNTER

AIM/OBJECTIVE OF THE EXPERIEMENT:

Implementing 4-bit up down counter using timer library in tinkercad.

ELECTRONIC COMPONENTS USED:

1)4 leds

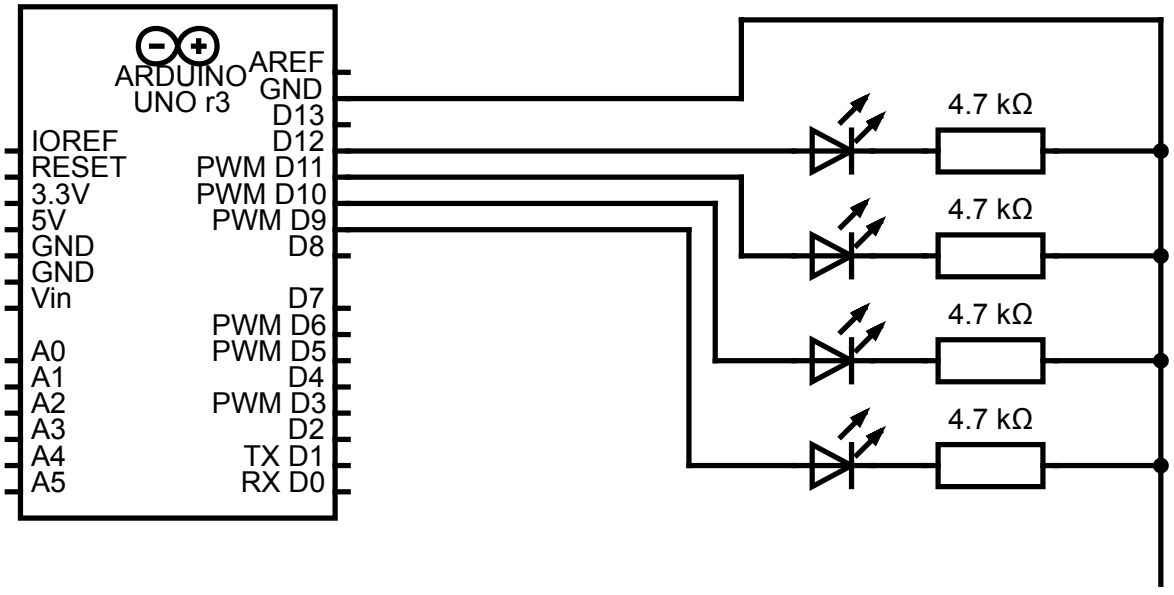
2)4 Resistors

3)Ardunio UNO R3

4)Breadboard

5)Wires

REFERENCE CIRCUIT:



PROCEDURE:

1)Take the breadboard and ardunio.

2)Give connections to the vcc and ground from ardunio to breadboard.

3)Connect the leds and resistors as shown in the reference circuit.Here each led represents the binary bit.

4)Now in the code part, modify the code for the 4 leds to work like a 4 bit updown counter.

5)Here in the essential code part initialise the led names and Timer .t

6)And then come to the setup part in the code use t.oscillate function to toggle the pins in a predefined time period that means each led work like a binary bit and give overall output as a 4 bit updown counter.

7)Now initialise a variable as repeat and equate it to t.every function to fire a function after a set of interval of time.

8)Now create a function (say stopAlltimers) and in this function write the code to stop the existing time using t.stop function and Here we need to give counter working method.

9)**Code:(code which is in the bigger text i.e., entered by us)**

```
#ifndef Event_h

#define Event_h
#include <inttypes.h>
#define EVENT_NONE 0
```

```
#define EVENT_EVERY 1
#define EVENT_OSCILLATE 2
class Event
{ public:
  Event(void);
  void update(void);
  void update(unsigned long now);
  int8_t eventType;
  unsigned long period;
  int repeatCount;
  uint8_t pin;
  uint8_t pinState;
  void (*callback)(void);
  unsigned long lastEventTime;
  int count;
};
#endif
#ifndef Timer_h
#define Timer_h
#include <inttypes.h>
#define MAX_NUMBER_OF_EVENTS (10)
#define TIMER_NOT_AN_EVENT (-2)
#define NO_TIMER_AVAILABLE (-1)
class Timer
{
public:
  Timer(void);
  int8_t every(unsigned long period, void (*callback)(void));
  int8_t every(unsigned long period, void (*callback)(void), int repeatCount);
  int8_t after(unsigned long duration, void (*callback)(void));
  int8_t oscillate(uint8_t pin, unsigned long period, uint8_t startingValue);
  int8_t oscillate(uint8_t pin, unsigned long period, uint8_t startingValue, int repeatCount);
  int8_t pulse(uint8_t pin, unsigned long period, uint8_t startingValue);
  int8_t pulseImmediate(uint8_t pin, unsigned long period, uint8_t pulseValue);
  void stop(int8_t id);
  void update(void);
  void update(unsigned long now);
protected:
  Event _events[MAX_NUMBER_OF_EVENTS];
  int8_t findFreeEventIndex(void);
};
#endif
Timer t;
int x = 0;
int Led0,Led1,Led2,Led3;
int repeat;
void setup() {
    Serial.begin(9600);
    pinMode(12, OUTPUT);
```

```
    pinMode(11, OUTPUT);
    pinMode(10, OUTPUT);
    pinMode(9, OUTPUT);

    Led3 = t.oscillate(12, 8*500, LOW);
    Led2 = t.oscillate(11, 4*500, LOW);
    Led1 = t.oscillate(10, 2*500, LOW);
    Led0 = t.oscillate(9, 500, LOW);
     x = 1;
    repeat = t.every(16*500, stopAllTimers);
}
void loop() {
    t.update();
}
void stopAllTimers() {
  t.stop(Led0);
  t.stop(Led1);
  t.stop(Led2);
  t.stop(Led3);
  if(x == 0){
    Led3 = t.oscillate(12, 8*500, LOW);
    Led2 = t.oscillate(11, 4*500, LOW);
    Led1 = t.oscillate(10, 2*500, LOW);
    Led0 = t.oscillate(9, 500, LOW);
    x = 1;
  }
  else if(x == 1){
    Led3 = t.oscillate(12, 8*500, HIGH);
    Led2 = t.oscillate(11, 4*500, HIGH);
    Led1 = t.oscillate(10, 2*500, HIGH);
    Led0 = t.oscillate(9, 500, HIGH);
    x = 0;
  }
}
```

```
Event::Event(void)
{
	eventType = EVENT_NONE;
}
void Event::update(void)
{
  unsigned long now = millis();
  update(now);
}
void Event::update(unsigned long now)
{	if (now - lastEventTime >= period)
	{
		switch (eventType)
		{
			case EVENT_EVERY:
				(*callback)();
				break;

			case EVENT_OSCILLATE:
				pinState = ! pinState;
				digitalWrite(pin, pinState);
				break;
		}
		lastEventTime = now;
		count++;
	}
	if (repeatCount > -1 && count >= repeatCount)
	{
		eventType = EVENT_NONE;
	}
}
Timer::Timer(void)
{
}
int8_t Timer::every(unsigned long period, void (*callback)(), int repeatCount)
{
	int8_t i = findFreeEventIndex();
	if (i == -1) return -1;
	_events[i].eventType = EVENT_EVERY;
	_events[i].period = period;
	_events[i].repeatCount = repeatCount;
	_events[i].callback = callback;
	_events[i].lastEventTime = millis();
	_events[i].count = 0;
	return i;
}
int8_t Timer::every(unsigned long period, void (*callback)())
{
	return every(period, callback, -1); // - means forever
}
int8_t Timer::after(unsigned long period, void (*callback)())
{
```

```
	return every(period, callback, 1);
}
int8_t Timer::oscillate(uint8_t pin, unsigned long period, uint8_t startingValue, int repeatCount)
{	int8_t i = findFreeEventIndex();
	if (i == NO_TIMER_AVAILABLE) return NO_TIMER_AVAILABLE;
	_events[i].eventType = EVENT_OSCILLATE;
	_events[i].pin = pin;
	_events[i].period = period;
	_events[i].pinState = startingValue;
	digitalWrite(pin, startingValue);
	_events[i].repeatCount = repeatCount * 2; // full cycles not transitions
	_events[i].lastEventTime = millis();
	_events[i].count = 0;
	return i;
}
int8_t Timer::oscillate(uint8_t pin, unsigned long period, uint8_t startingValue)
{
	return oscillate(pin, period, startingValue, -1); // forever
}
int8_t Timer::pulse(uint8_t pin, unsigned long period, uint8_t startingValue)
{
	return oscillate(pin, period, startingValue, 1); // once
}
int8_t Timer::pulseImmediate(uint8_t pin, unsigned long period, uint8_t pulseValue)
{
	int8_t id(oscillate(pin, period, pulseValue, 1));
	if (id >= 0 && id < MAX_NUMBER_OF_EVENTS) {
		_events[id].repeatCount = 1;
	}
	return id;
}
void Timer::stop(int8_t id)
{
	if (id >= 0 && id < MAX_NUMBER_OF_EVENTS) {
		_events[id].eventType = EVENT_NONE;
	}
}
void Timer::update(void)
{
	unsigned long now = millis();
	update(now);
}
void Timer::update(unsigned long now)
{	for (int8_t i = 0; i < MAX_NUMBER_OF_EVENTS; i++)
	{	if (_events[i].eventType != EVENT_NONE)
		{
			_events[i].update(now);
		}
	}
}
int8_t Timer::findFreeEventIndex(void)
{
```

```
        for (int8_t i = 0; i < MAX_NUMBER_OF_EVENTS; i++)
        {
                if (_events[i].eventType == EVENT_NONE)
                {
                        return i;
                }
        }
        return NO_TIMER_AVAILABLE;
}
```

9)Now start simulation and observe the output.

OBSERVATION:

1)Here we observe that 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 untill the simulation is stopped.

2)As per given below table our circuit works.

A	B	C	D
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

3)This table values repeats untill we stop simulation.

4)In this lab we learnt about how to use timer library in ardunio.

LINK FOR THE TINKERCAD SIMULATION:

https://www.tinkercad.com/things/2HsMEjGBc3h-part-c-lab5/editel?sharecode=hbIQZ6obKxPakAi-etIpqTy8x3fOB4D_j_9NoNdSpKg