AMERICAN INTERNATIONAL UNIVERSITY–BANGLADESH (AIUB)

FACULTY OF SCIENCE & TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

MICROPROCESSOR AND EMBEDDED SYSTEMS

Spring 2024-2025

Section: Q**, Group: 08**

LAB REPORT ON

Study of a Digital Timer using millis() function of Arduino to avoid problems associated with the delay () function.

Experiment No:04

Supervised By

Protik Parvez Sheikh

Submitted By

MD. AL-IMRAN SAYEM

Group Member

No.	Name	ID
1.	**MD. AL-IMRAN SAYEM**	**22-48023-2**
2.	MD. TANJIL TASHRIK ZIM	22-48021-2
3.	MD. ABRAR RAFID SHARIAR	22-48055-2
4.	HUMAIRA ZANNAT	22-47789-2
5	SANJUKTA DAS	22-47801-2

Date of Submission: April 8, 2025

Title: Study of a Digital Timer using millis() function of Arduino to avoid problems associated with the delay() function.

Objective:

The objective of this experiment is to design a digital timer using Arduino's millis () function to control six LEDs, turning one on every minute without using the delay() function. This approach keeps the system responsive, allowing other operations to run simultaneously. A tilt switch is used as a digital input to reset the timer and LEDs, helping students understand non-blocking time management, digital inputs, and sequential output control in embedded systems.

Apparatus:

1. Arduino Uno/Arduino Mega Microcontroller Board
2. Tilt sensor (One)
3. LEDs (Six)
4. Resistors (One 10 kilo ohm and Six 100 ohm)

Experimental Setup:

Picture for **LED control with tilt Sensor and millis() function**

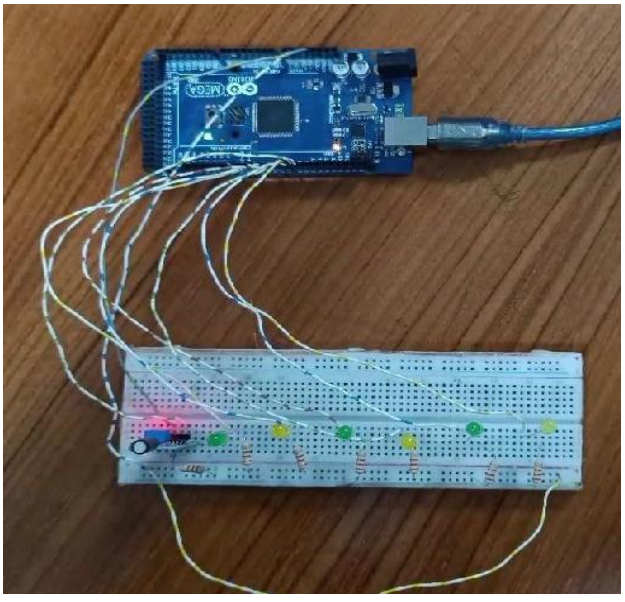

Figure 1: LED control with millis() function and tilt sensor

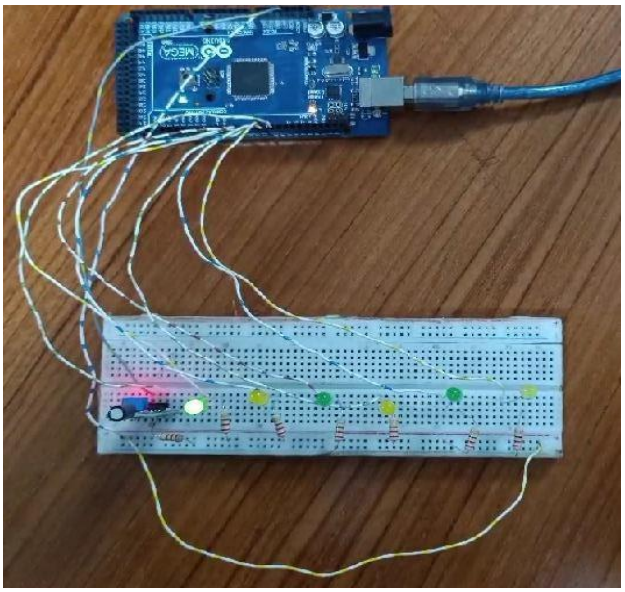

Figure 2: LED 1 on for 1 minute

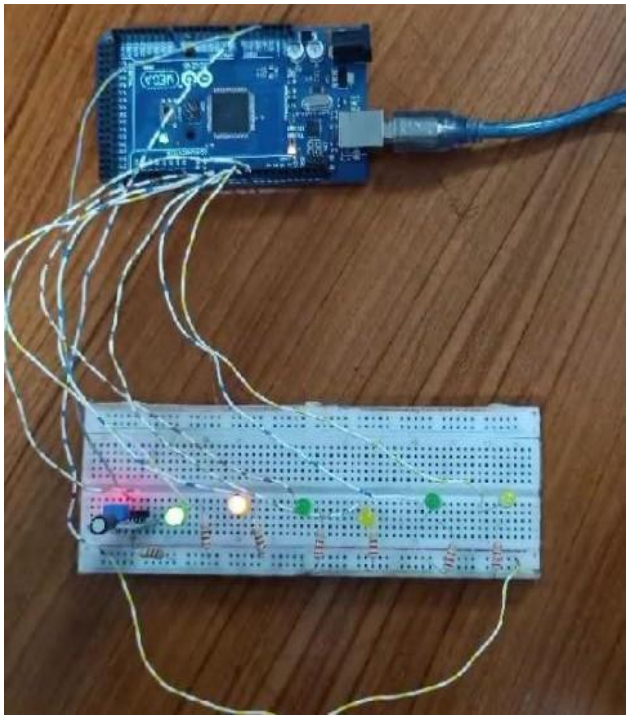

Figure 3: LED 2 on for 1 minute

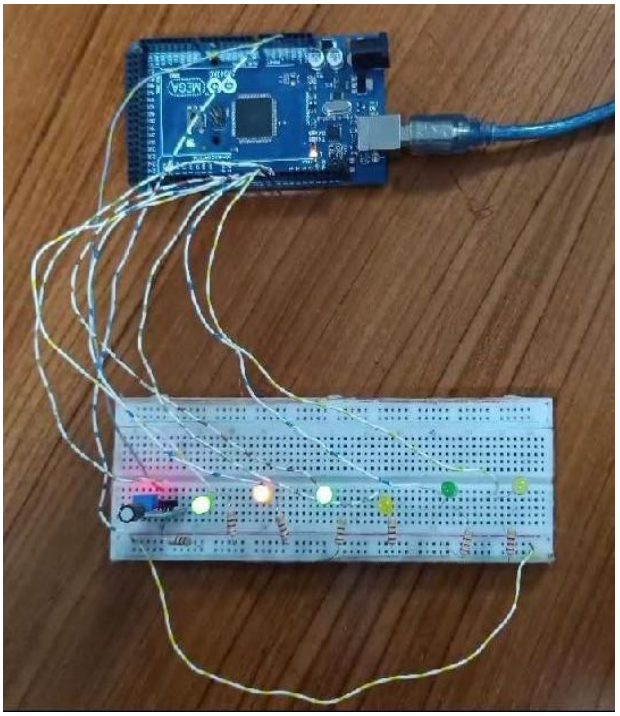

Figure 4: LED 3 on for 1 minute

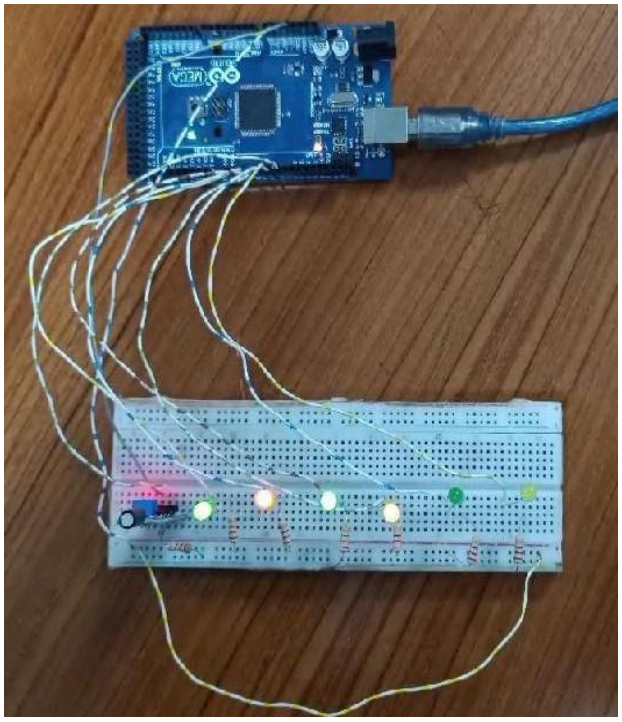

Figure 5: LED 4 on for 1 minute

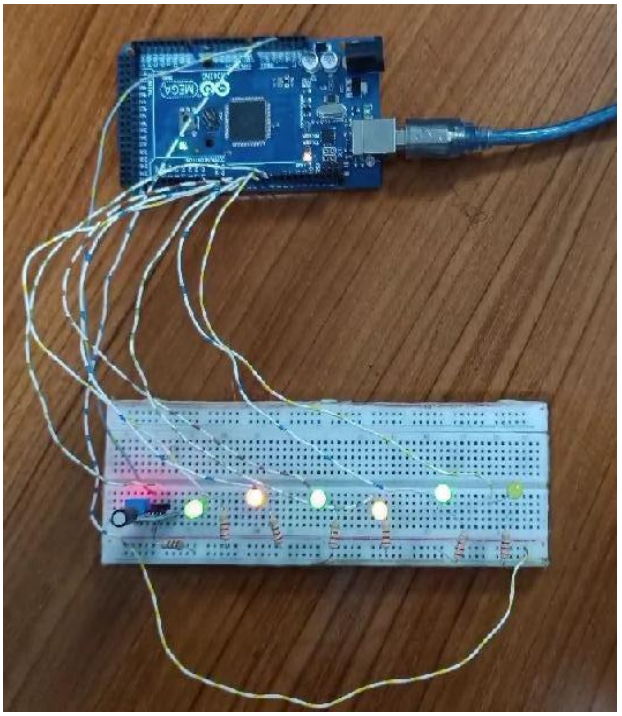

Figure 6: LED 5 on for 1 minute

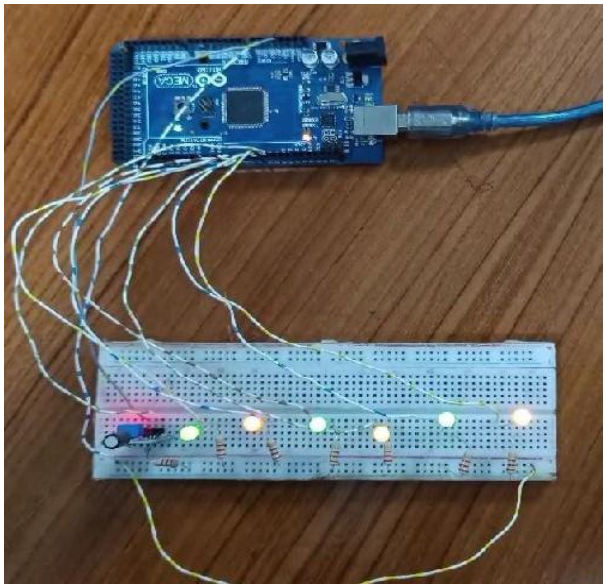

Figure 7: LED 6 on for 1 minute

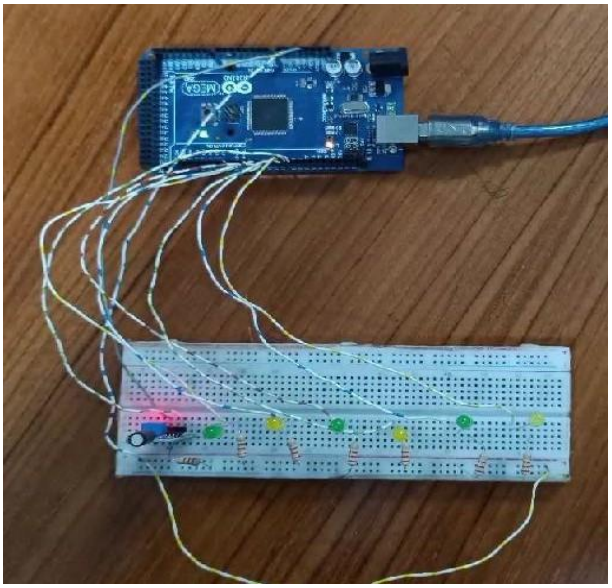

Figure 8: Reset LED by tilt sensor

Code:

Code for LED control with millis() function

```
const int SwitchPin = 8;
unsigned long PreviousTime = 0;
int SwitchState = 0;
int PrevSwitchState = 0;
int led = 2;
long interval = 60000;
void setup() {
        for (int x = 2; x < 8; x++)
        {
                pinMode(x, OUTPUT);
        }
        pinMode(SwitchPin, INPUT);
}

void loop() {
        unsigned long CurrentTime = millis();
        if (CurrentTime - PreviousTime > interval) {
                PreviousTime = CurrentTime;
                digitalWrite(led, HIGH);
```

```
            led++;
            if (led == 7) {
            }
        }
        SwitchState = digitalRead(SwitchPin);
        if (SwitchState != PrevSwitchState) {
            for (int x = 2 ; x < 8; x++)
            {
                digitalWrite(x, LOW);
            }
            led = 2;
            PreviousTime = CurrentTime;
        }
        PrevSwitchState = SwitchState;
}
```

Question Answer:

Simulation for LED control with Tilt sensor and millis() function

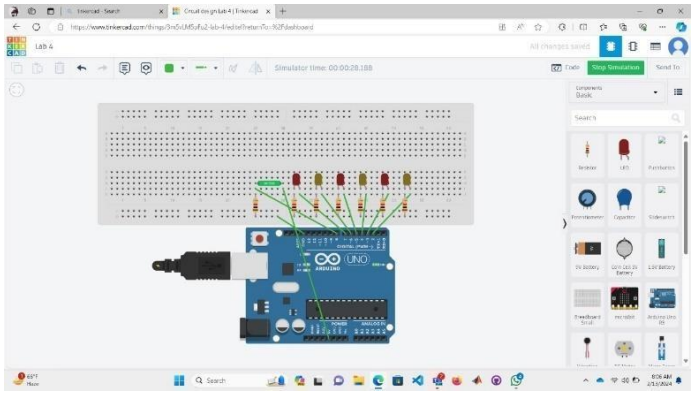

Figure 1: LED control by millis()function,tilt sensor

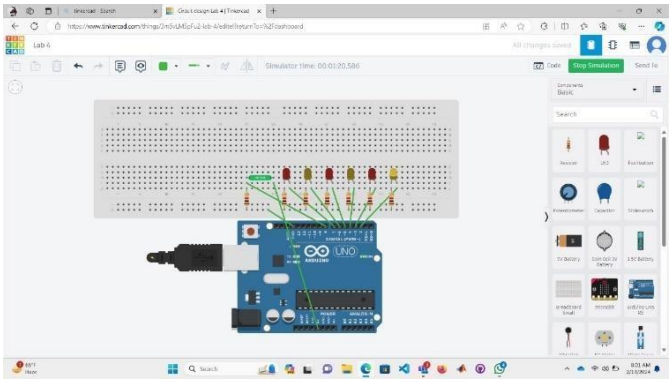

Figure 2: LED 1 on for 1 minute

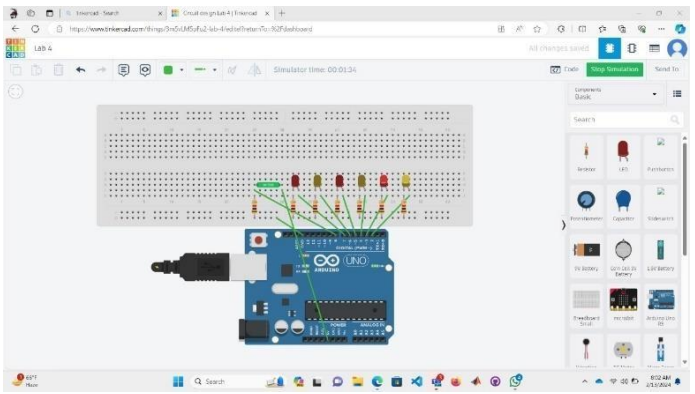

Figure 3: LED 2 on for 1 minute

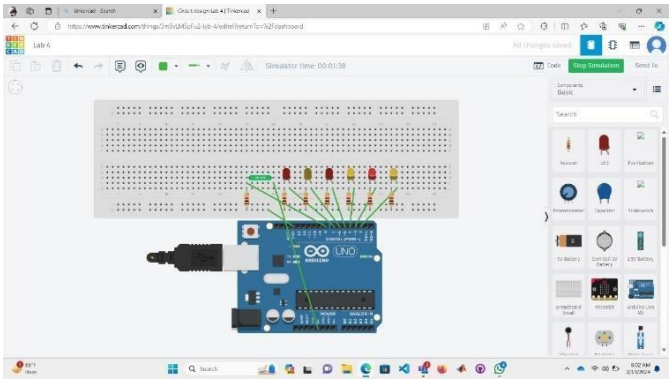

Figure 4: LED 3 on for 1 minute

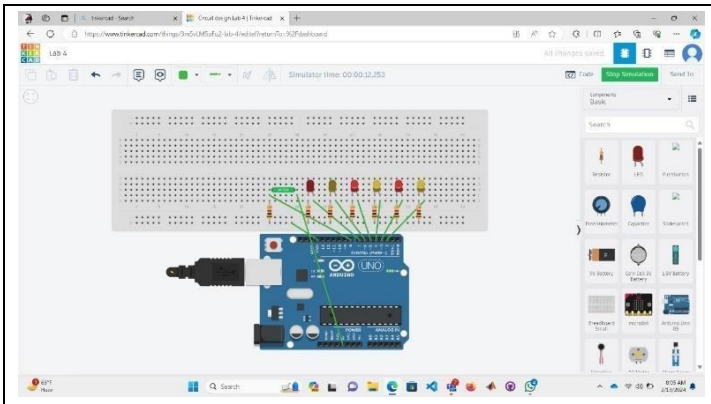

Figure 5: LED 4 on for 1 minute

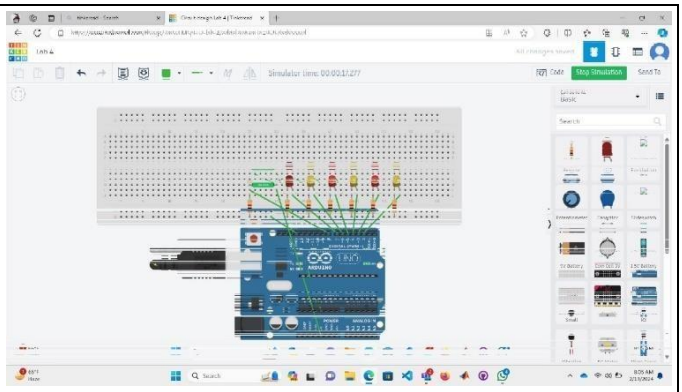

Figure 6: LED 5 on for 1 minute

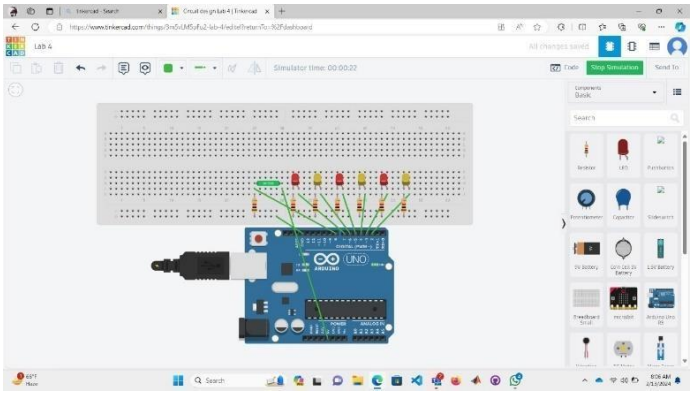

Figure 7: LED 6 on for 1 minute

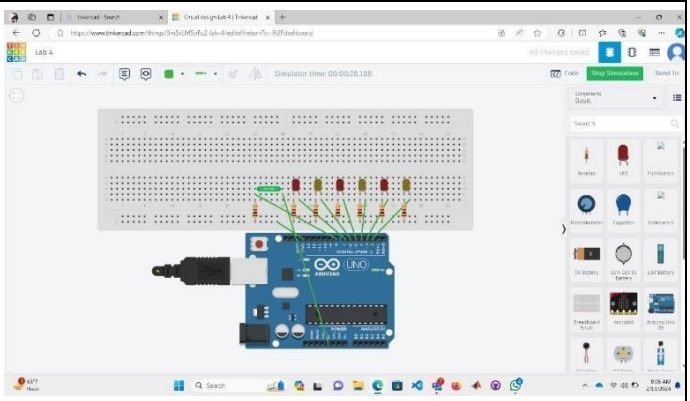

Figure 8: Reset LED by tilt sensor

Discussion and Conclusion:

The study aimed to design a digital timer using the millis() function in Arduino to overcome the limitations associated with the delay() function. The delay() function in Arduino can lead to blocking code execution, making it unsuitable for multitasking or time-sensitive applications. By employing millis(), a non-blocking alternative, the timer can perform other tasks while counting time accurately. The timer was integrated with a tilt sensor to initiate the timing process upon detecting movement. This enhances user interaction, making the timer more practical for various applications. The tilt sensor, acting as a trigger, eliminates the need for manual initiation, adding an element of automation. Moreover, by avoiding the delay() function, the system remains responsive, allowing for concurrent execution of tasks. This is crucial for real-world applications where multitasking is essential. The implementation ensures that the timer remains accurate, providing a reliable timekeeping mechanism.

In conclusion, the utilization of the millis() function in Arduino, coupled with a tilt sensor, enhances the efficiency and versatility of the digital timer. The shift from delay() to millis() mitigates the blocking issues associated with the former, making the timer more responsive. The inclusion of a tilt sensor introduces a hands-free aspect to the timer, catering to diverse user scenarios. This study showcases the significance of adopting non-blocking mechanisms in Arduino programming for improved functionality and responsiveness in time-sensitive applications. The digital timer with millis() function and tilt sensor integration presents a

robust solution suitable for a range of projects where accurate timing and user-friendly interaction are paramount.

Reference:

1. https://www.arduino.cc/.
2. ATMega328 manual
3. https://www.avrfreaks.net/forum/tut-c-newbies-guide-avr-timers
4. http://maxembedded.com/2011/06/avr-timers-timer0/