جامعة بيرزيت

BIRZEIT UNIVERSITY

Faculty of Engineering & Technology Electrical & Computer Engineering Department

Interfacing Techniques – ENCS4380

Autonomous Drone with Infrared Camera to Assist First Responders

Prepared by:

Maha Maher Mali	1200746

Instructor: Dr. Wasel Ghanem

Section: 1

Date: 25-12-2023

Table of Contents

Table of Figures

Question 1

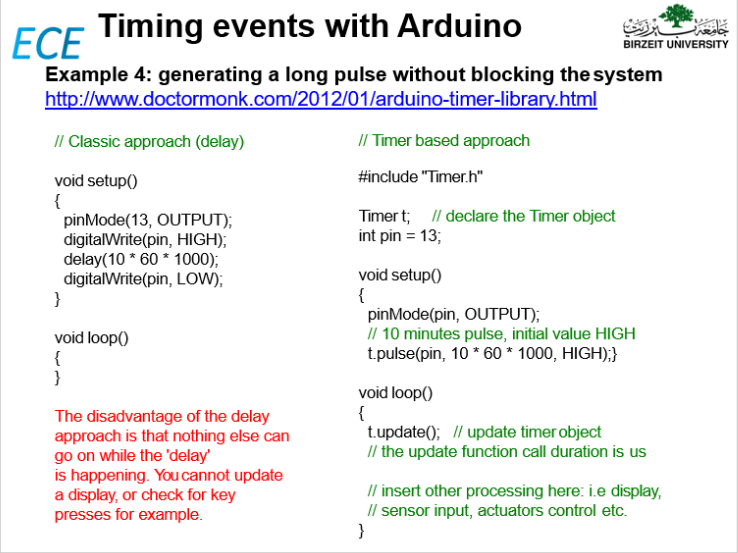
Definition

Delay Function: pauses the program for the amount of time (in milliseconds) specified as parameter.[1]

Timer Library: calls simple non-blocking timer library for calling functions in / at / every specified unit of time. Supports mills, micros, time rollover, and compile time configurable number of tasks.[2]

Example 4

Required: we want to resolve example 4 as shown in figure1 , but without using the timer library and delay function.



ECE Timing events with Arduino

BIRZEIT UNIVERSITY

Example 4: generating a long pulse without blocking the system

http://www.doctormonk.com/2012/01/arduino-timer-library.html

```
// Classic approach (delay)

void setup()
{
  pinMode(13, OUTPUT);
  digitalWrite(pin, HIGH);
  delay(10 * 60 * 1000);
  digitalWrite(pin, LOW);
}

void loop()
{
}
```

The disadvantage of the delay approach is that nothing else can go on while the 'delay' is happening. You cannot update a display, or check for key presses for example.

```
// Timer based approach

#include "Timer.h"

Timer t;    // declare the Timer object
int pin = 13;

void setup()
{
  pinMode(pin, OUTPUT);
  // 10 minutes pulse, initial value HIGH
  t.pulse(pin, 10 * 60 * 1000, HIGH);}

void loop()
{
  t.update();   // update timer object
  // the update function call duration is us

  // insert other processing here: i.e display,
  // sensor input, actuators control etc.
}
```

Figure 1:Example 4 Slide code

In this example we want to generate a long pulse without blocking the system and without using timer library and delay function. So, to achieve this I connected the circuit shown in figure 2 using tinker cad program.

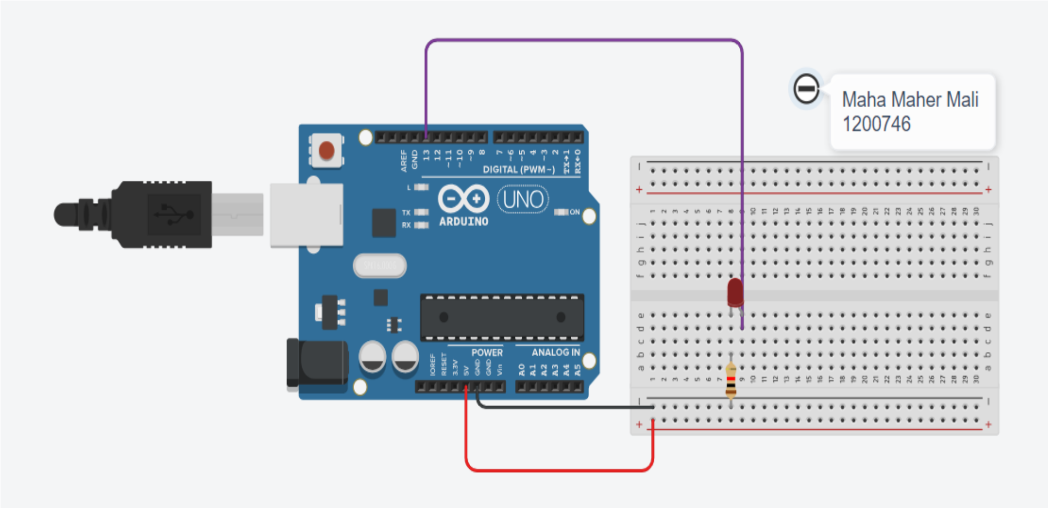
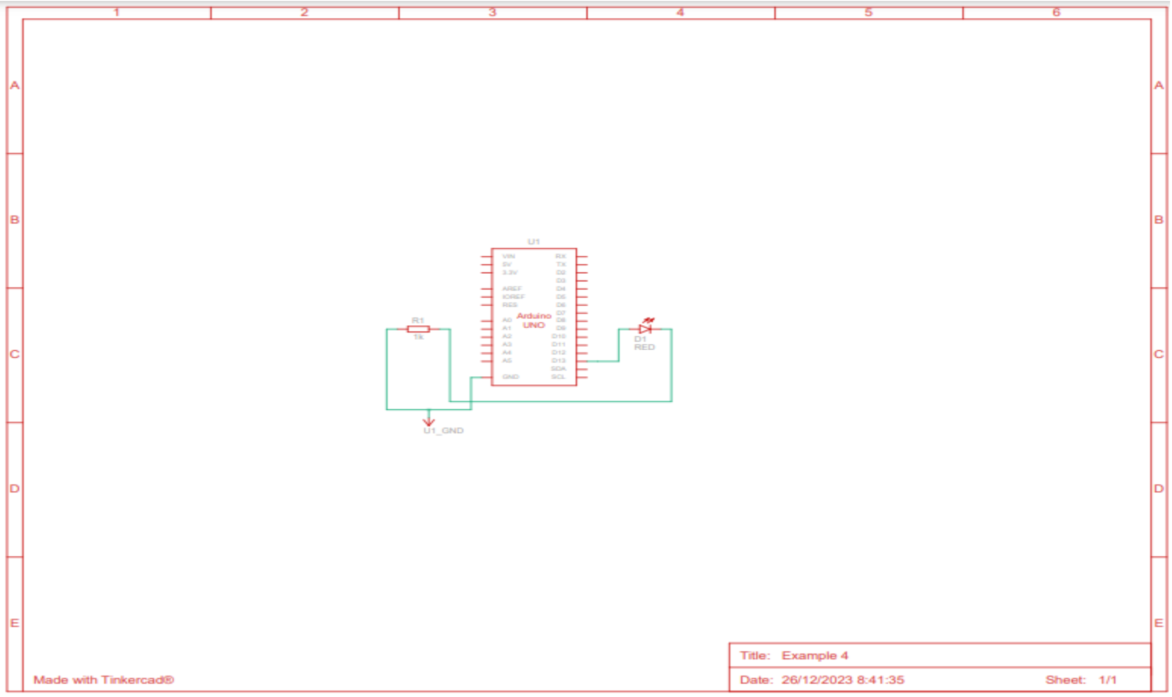


Figure 2:Example 4 Connection on Tinker Cad

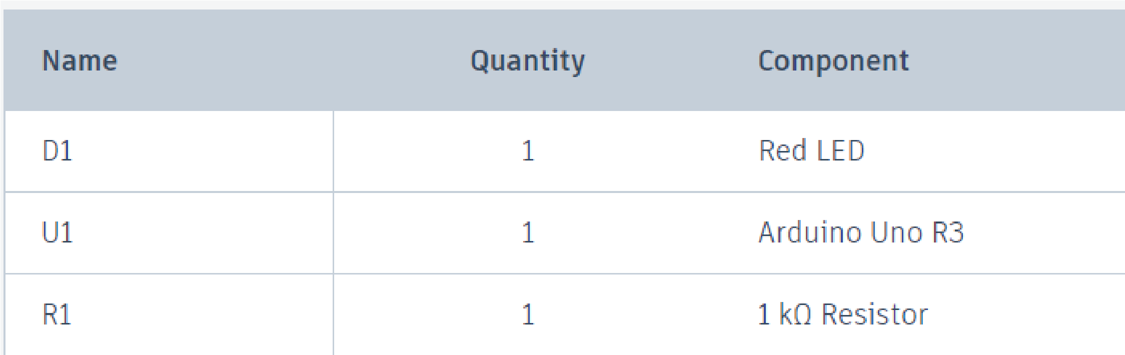


Made with Tinkercad®

Title: Example 4

Date: 26/12/2023 8:41:35 Sheet: 1/1

Figure 3: Example 4 Schematic View

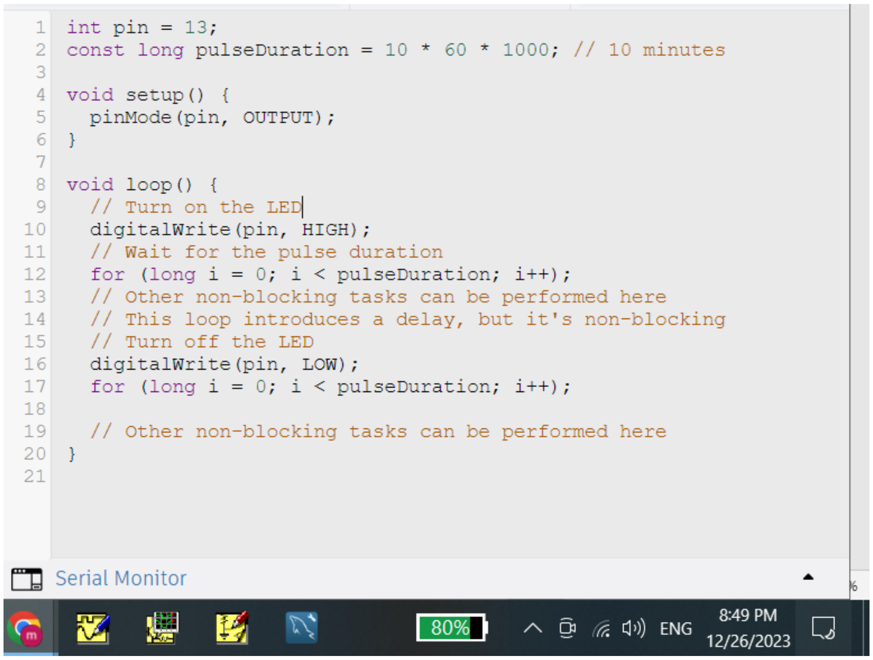


Name	Quantity	Component
D1	1	Red LED
U1	1	Arduino Uno R3
R1	1	1 kΩ Resistor

Figure 4: Example 4 Component List

Tinker Cad Link: https://www.tinkercad.com/things/0mlCjB31TA0-example-4/editel?returnTo=%2Fdashboard%3Ftype%3Dcircuits%26collection%3Ddesigns&sharecode=-Ho0dIZu2fOTHDXgtZulx7olDvQtigQWMe37ZenqayY

Code Description and Results



```
int pin = 13;
const long pulseDuration = 10 * 60 * 1000; // 10 minutes

void setup() {
  pinMode(pin, OUTPUT);
}

void loop() {
  // Turn on the LED
  digitalWrite(pin, HIGH);
  // Wait for the pulse duration
  for (long i = 0; i < pulseDuration; i++);
  // Other non-blocking tasks can be performed here
  // This loop introduces a delay, but it's non-blocking
  // Turn off the LED
  digitalWrite(pin, LOW);
  for (long i = 0; i < pulseDuration; i++);

  // Other non-blocking tasks can be performed here
}

```

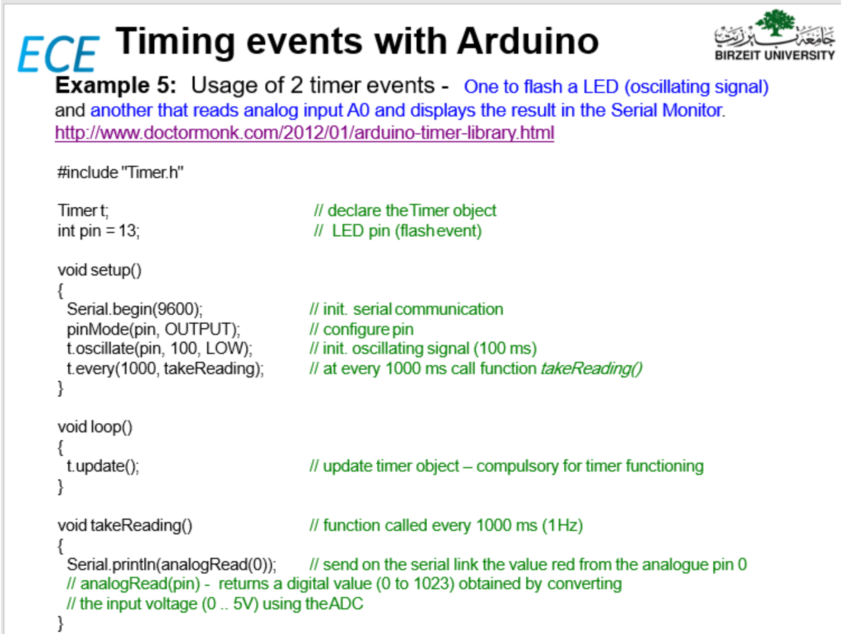
Figure 5:Example 4 Code

I set the pin 13 as an output pin (connect the LED to pin 13 from the Arduino), in the loop function, at first turn the LED on by writing HIGH to pin 13. Also, we create for loop that does nothing just for waste time (replacement for the delay function. After the loop finishes, we turn the LED off by writing LOW to pin 13 and then create another for loop to waste time before the LED turns on again.

The main goal of this code is generating long pulse duration without blocking the system, this is done through control the LED which is connected to pin 13 on the Arduino, the LED is turned on for a specified duration which is 10 minutes.

Example 5

Required: we want to resolve example 5 as shown in figure 5, but without using the timer library and delay function.



ECE Timing events with Arduino

BIRZEIT UNIVERSITY

Example 5: Usage of 2 timer events - One to flash a LED (oscillating signal) and another that reads analog input A0 and displays the result in the Serial Monitor.
http://www.doctormonk.com/2012/01/arduino-timer-library.html

```
#include "Timer.h"

Timer t;                              // declare the Timer object
int pin = 13;                         //  LED pin (flash event)

void setup()
{
  Serial.begin(9600);                 // init. serial communication
  pinMode(pin, OUTPUT);               // configure pin
  t.oscillate(pin, 100, LOW);         // init. oscillating signal (100 ms)
  t.every(1000, takeReading);         // at every 1000 ms call function takeReading()
}

void loop()
{
  t.update();                         // update timer object – compulsory for timer functioning
}

void takeReading()                    // function called every 1000 ms (1Hz)
{
  Serial.println(analogRead(0));      // send on the serial link the value red from the analogue pin 0
  // analogRead(pin) -  returns a digital value (0 to 1023) obtained by converting
  // the input voltage (0 .. 5V) using the ADC
}
```

Figure 6:Example 5 Slide Code

In this example we want to usage of 2 timer events one to flash a LED (oscillating signal) and another that reads analog input A0 and displays the result in the Serial Monitor. So, to achieve this I connected the circuit shown in figure 7 using tinker cad program.

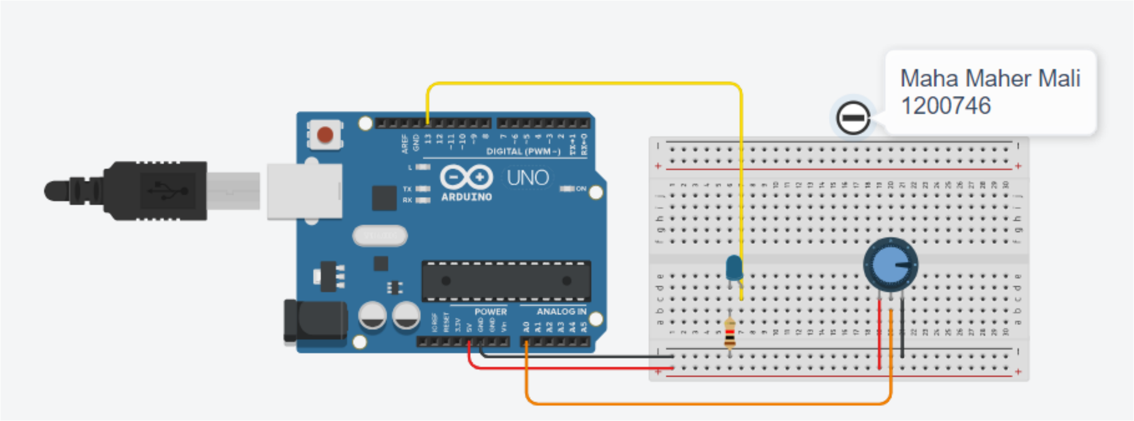

Maha Maher Mali
1200746

Figure 7: Example 5 Connection on Tinker Cad

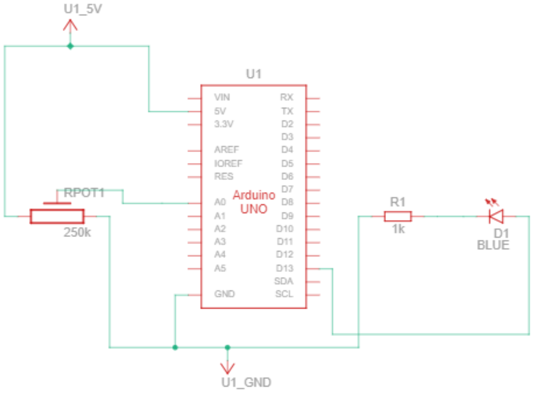
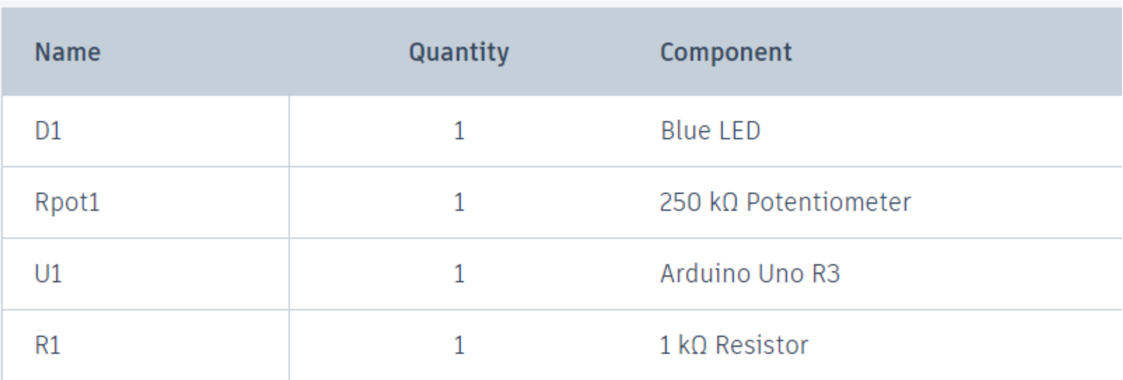


Figure 8: Example 5 Schematic View

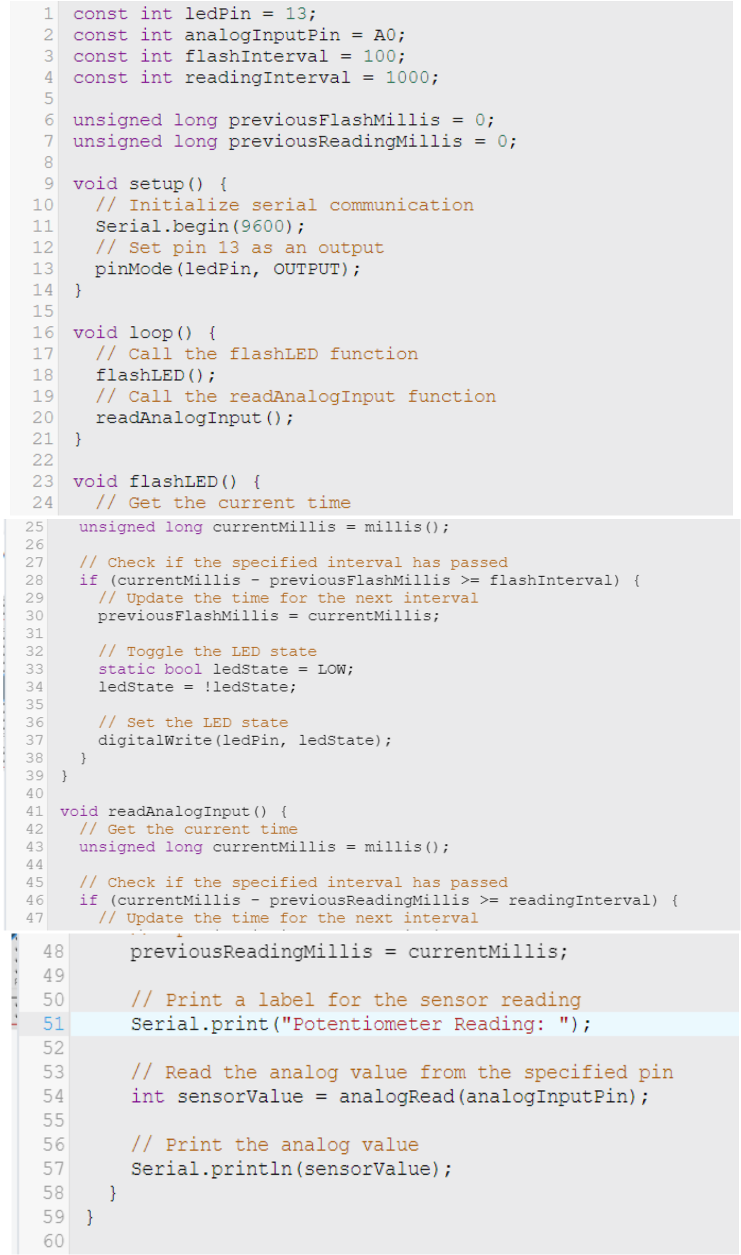


Name	Quantity	Component
D1	1	Blue LED
Rpot1	1	250 kΩ Potentiometer
U1	1	Arduino Uno R3
R1	1	1 kΩ Resistor

Figure 9:Example 5 Component List

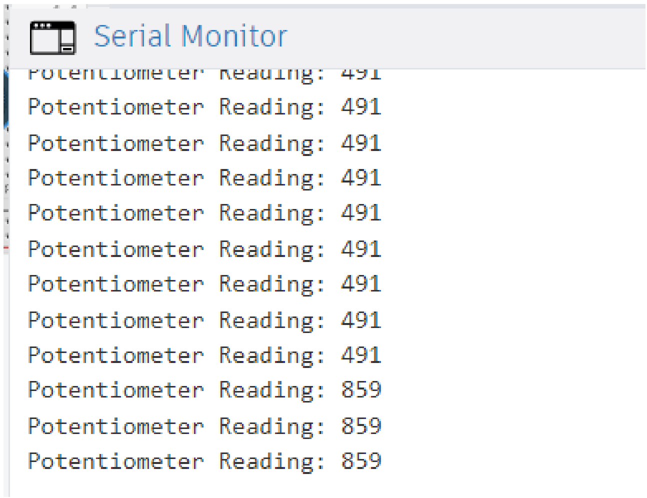
Tinker Cad Link: https://www.tinkercad.com/things/0KlSjFqIBCn-powerful-habbi-maimu/editel?sharecode=1FrRcoNOjIz9pMDqtyFN6vmtSzdgpLnbgDqE-rDYwgY

Code Description and Results



```
const int ledPin = 13;
const int analogInputPin = A0;
const int flashInterval = 100;
const int readingInterval = 1000;

unsigned long previousFlashMillis = 0;
unsigned long previousReadingMillis = 0;

void setup() {
  // Initialize serial communication
  Serial.begin(9600);
  // Set pin 13 as an output
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // Call the flashLED function
  flashLED();
  // Call the readAnalogInput function
  readAnalogInput();
}

void flashLED() {
  // Get the current time
  unsigned long currentMillis = millis();

  // Check if the specified interval has passed
  if (currentMillis - previousFlashMillis >= flashInterval) {
    // Update the time for the next interval
    previousFlashMillis = currentMillis;

    // Toggle the LED state
    static bool ledState = LOW;
    ledState = !ledState;

    // Set the LED state
    digitalWrite(ledPin, ledState);
  }
}

void readAnalogInput() {
  // Get the current time
  unsigned long currentMillis = millis();

  // Check if the specified interval has passed
  if (currentMillis - previousReadingMillis >= readingInterval) {
    // Update the time for the next interval
    previousReadingMillis = currentMillis;

    // Print a label for the sensor reading
    Serial.print("Potentiometer Reading: ");

    // Read the analog value from the specified pin
    int sensorValue = analogRead(analogInputPin);

    // Print the analog value
    Serial.println(sensorValue);
  }
}
```

Figure 10:Example 5 Code



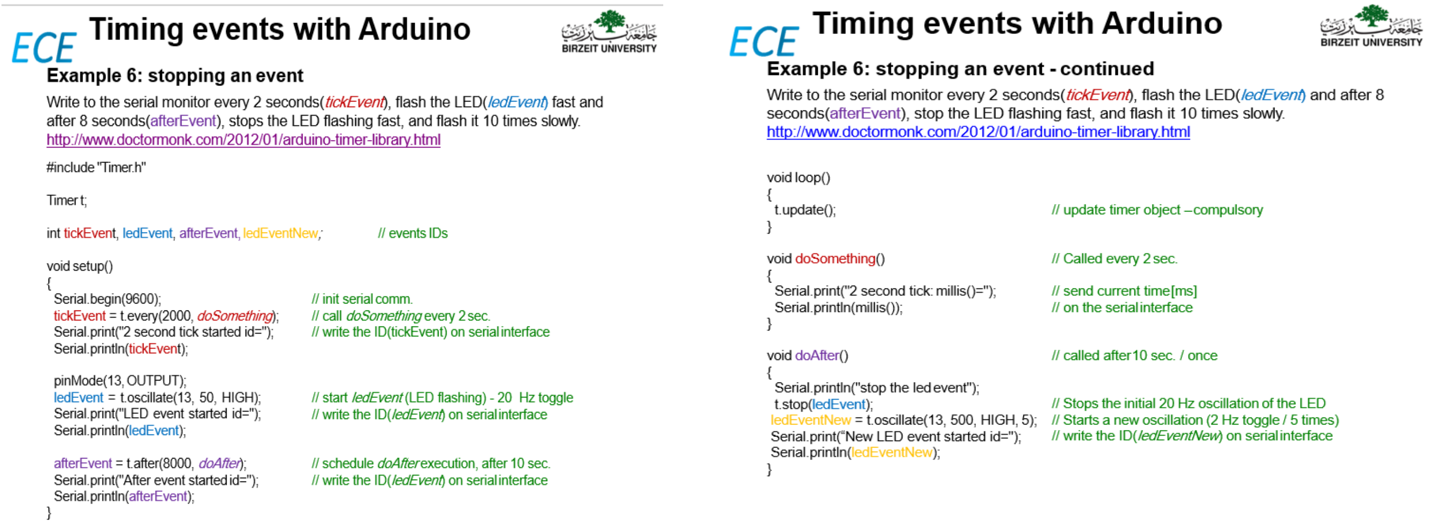
```
Potentiometer Reading: 491
Potentiometer Reading: 491
Potentiometer Reading: 491
Potentiometer Reading: 491
Potentiometer Reading: 491
Potentiometer Reading: 491
Potentiometer Reading: 491
Potentiometer Reading: 491
Potentiometer Reading: 491
Potentiometer Reading: 859
Potentiometer Reading: 859
Potentiometer Reading: 859
```

Figure 11:Serial Monitor Output Example 5

The main purpose of this code is to flash the LED on and off which is connected on pin 13 of the Arduino, the LED flashing on regular interval which is 100 milliseconds, we use the potentiometer as analog sensor reading to check sensor connected to pin A0, so every second the reading of the potentiometer (analog reading) will appear on Serial Monitor screen.

Example 6

Required: we want to resolve example 6 as shown in figure 12, but without using the timer library and delay function.



ECE **Timing events with Arduino**

Example 6: stopping an event

Write to the serial monitor every 2 seconds(*tickEvent*), flash the LED(*ledEvent*) fast and after 8 seconds(afterEvent), stops the LED flashing fast, and flash it 10 times slowly.
http://www.doctormonk.com/2012/01/arduino-timer-library.html

```
#include "Timer.h"

Timer t;

int tickEvent, ledEvent, afterEvent, ledEventNew;           // events IDs

void setup()
{
  Serial.begin(9600);                        // init serial comm.
  tickEvent = t.every(2000, doSomething);    // call doSomething every 2 sec.
  Serial.print("2 second tick started id="); // write the ID(tickEvent) on serial interface
  Serial.println(tickEvent);

  pinMode(13, OUTPUT);
  ledEvent = t.oscillate(13, 50, HIGH);      // start ledEvent (LED flashing) - 20  Hz toggle
  Serial.print("LED event started id=");     // write the ID(ledEvent) on serial interface
  Serial.println(ledEvent);

  afterEvent = t.after(8000, doAfter);       // schedule doAfter execution, after 10 sec.
  Serial.print("After event started id=");   // write the ID(ledEvent) on serial interface
  Serial.println(afterEvent);
}
```

ECE **Timing events with Arduino**

Example 6: stopping an event - continued

Write to the serial monitor every 2 seconds(*tickEvent*), flash the LED(*ledEvent*) and after 8 seconds(afterEvent), stop the LED flashing fast, and flash it 10 times slowly.
http://www.doctormonk.com/2012/01/arduino-timer-library.html

```
void loop()
{
  t.update();                                 // update timer object –compulsory
}

void doSomething()                            // Called every 2 sec.
{
  Serial.print("2 second tick: millis()=");   // send current time[ms]
  Serial.println(millis());                   // on the serial interface
}

void doAfter()                                // called after 10 sec. / once
{
  Serial.println("stop the led event");
  t.stop(ledEvent);                           // Stops the initial 20 Hz oscillation of the LED
 ledEventNew = t.oscillate(13, 500, HIGH, 5); // Starts a new oscillation (2 Hz toggle / 5 times)
 Serial.print("New LED event started id=");   // write the ID(ledEventNew) on serial interface
 Serial.println(ledEventNew);
}
```

Figure 12:Example 6 Slide Code

In this example we want to write on the serial monitor every 2 seconds (tickEvent), flash the LED (ledEvent) fast and after 8 seconds (afterEvent),stops the LED flashing fast, and flash it 10 times slowly So, to achieve this I connected the circuit shown in figure 13 using tinker cad program.

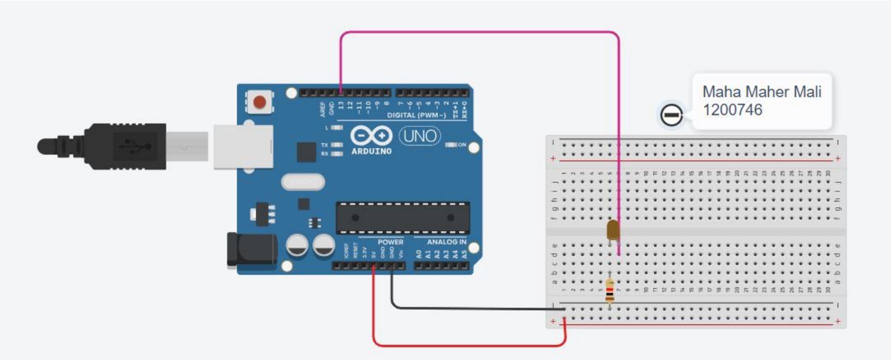


Figure 13:Example 6 Connection on Tinker Cad

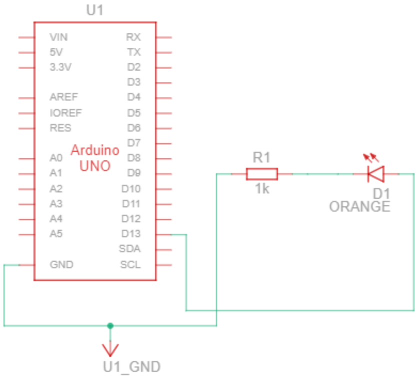
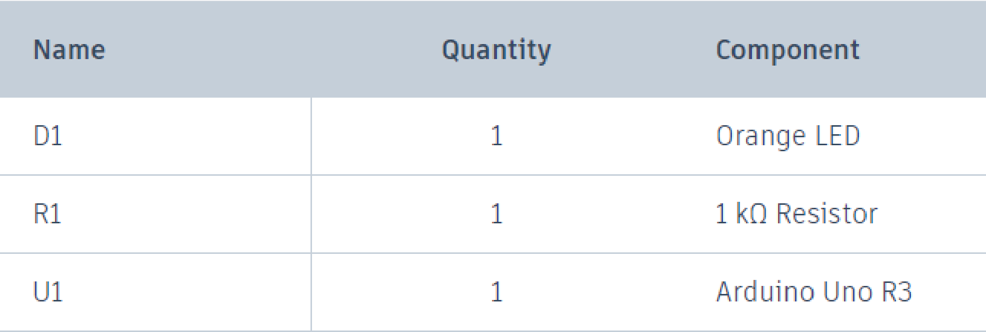


Figure 14: Example 6 Schematic View

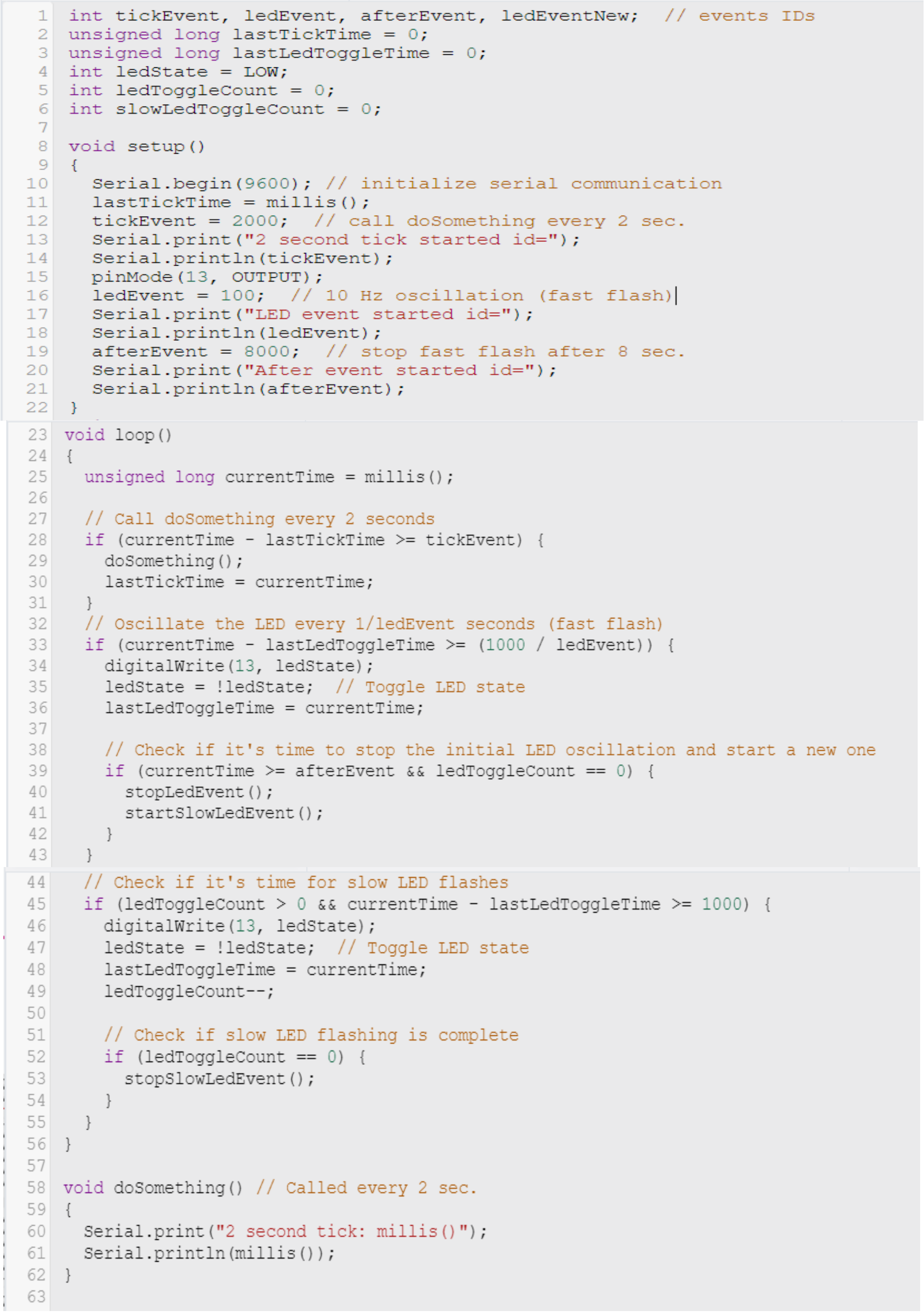


Name	Quantity	Component
D1	1	Orange LED
R1	1	1 kΩ Resistor
U1	1	Arduino Uno R3

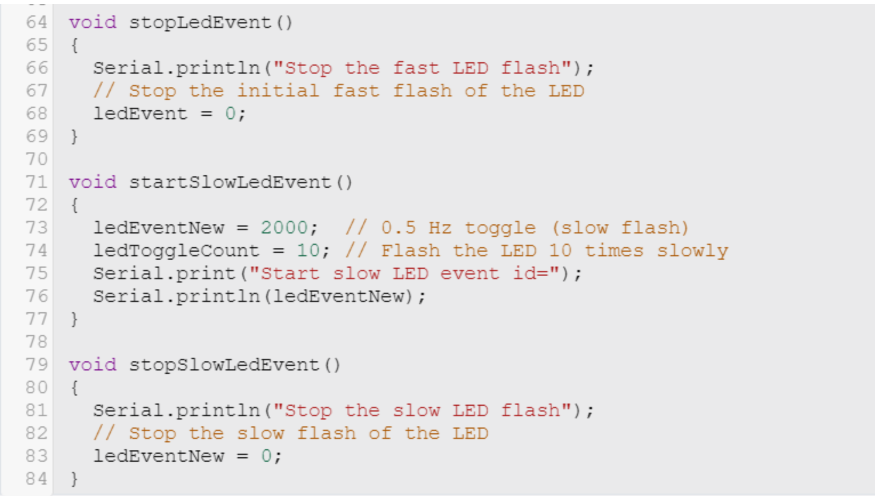
Figure 15: Example 6 Component List

Tinker Cad Link: https://www.tinkercad.com/things/c05vVvHzSdV-example-6/editel?returnTo=%2Fdashboard%3Ftype%3Dcircuits%26collection%3Ddesigns&sharecode=2w6aW4IizsnYpoFIHNv2vrWjK7W6bYHcuDFSgNF4tFw

Code Description and Results

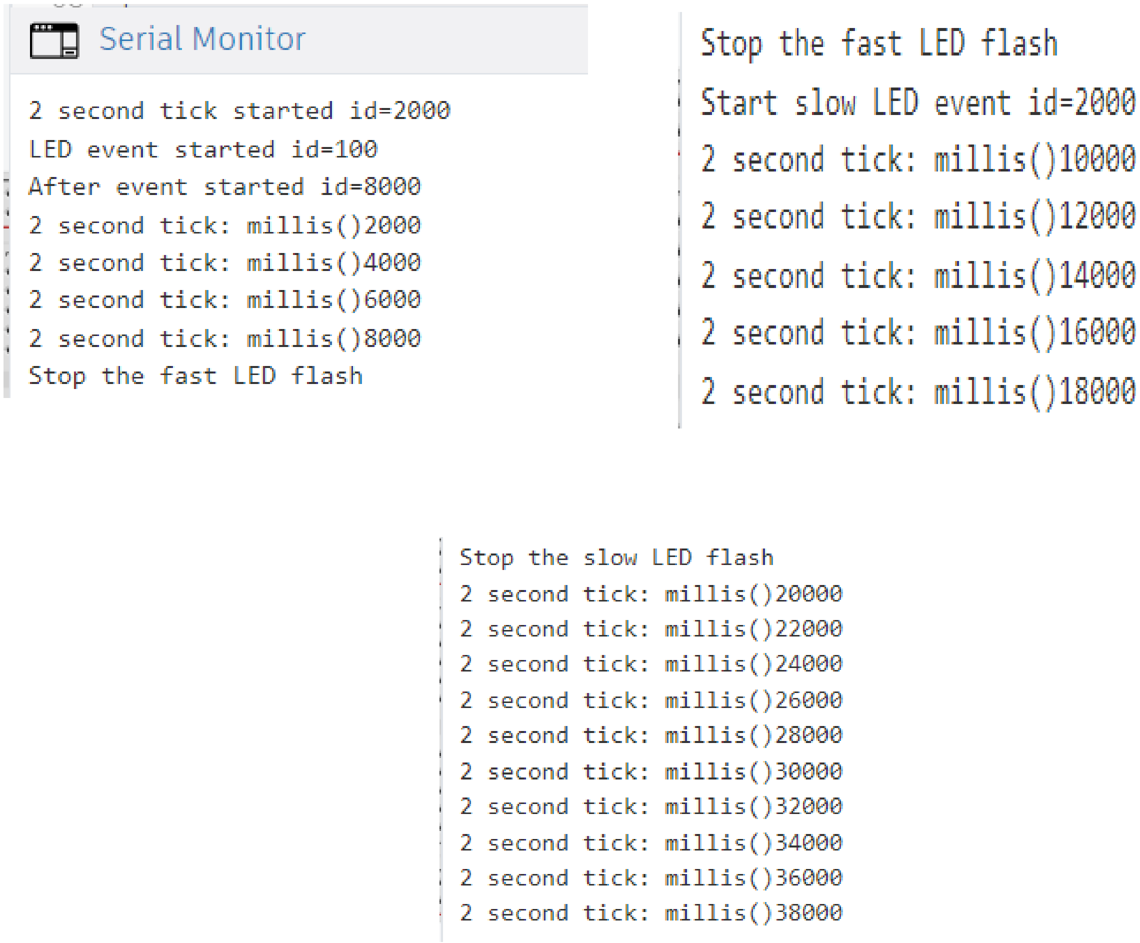


```
int tickEvent, ledEvent, afterEvent, ledEventNew;  // events IDs
unsigned long lastTickTime = 0;
unsigned long lastLedToggleTime = 0;
int ledState = LOW;
int ledToggleCount = 0;
int slowLedToggleCount = 0;

void setup()
{
  Serial.begin(9600); // initialize serial communication
  lastTickTime = millis();
  tickEvent = 2000;   // call doSomething every 2 sec.
  Serial.print("2 second tick started id=");
  Serial.println(tickEvent);
  pinMode(13, OUTPUT);
  ledEvent = 100;   // 10 Hz oscillation (fast flash)
  Serial.print("LED event started id=");
  Serial.println(ledEvent);
  afterEvent = 8000;  // stop fast flash after 8 sec.
  Serial.print("After event started id=");
  Serial.println(afterEvent);
}
void loop()
{
  unsigned long currentTime = millis();

  // Call doSomething every 2 seconds
  if (currentTime - lastTickTime >= tickEvent) {
    doSomething();
    lastTickTime = currentTime;
  }
  // Oscillate the LED every 1/ledEvent seconds (fast flash)
  if (currentTime - lastLedToggleTime >= (1000 / ledEvent)) {
    digitalWrite(13, ledState);
    ledState = !ledState;  // Toggle LED state
    lastLedToggleTime = currentTime;

    // Check if it's time to stop the initial LED oscillation and start a new one
    if (currentTime >= afterEvent && ledToggleCount == 0) {
      stopLedEvent();
      startSlowLedEvent();
    }
  }
  // Check if it's time for slow LED flashes
  if (ledToggleCount > 0 && currentTime - lastLedToggleTime >= 1000) {
    digitalWrite(13, ledState);
    ledState = !ledState;  // Toggle LED state
    lastLedToggleTime = currentTime;
    ledToggleCount--;

    // Check if slow LED flashing is complete
    if (ledToggleCount == 0) {
      stopSlowLedEvent();
    }
  }
}

void doSomething() // Called every 2 sec.
{
  Serial.print("2 second tick: millis()");
  Serial.println(millis());
}

```



```
void stopLedEvent()
{
  Serial.println("Stop the fast LED flash");
  // Stop the initial fast flash of the LED
  ledEvent = 0;
}

void startSlowLedEvent()
{
  ledEventNew = 2000;  // 0.5 Hz toggle (slow flash)
  ledToggleCount = 10; // Flash the LED 10 times slowly
  Serial.print("Start slow LED event id=");
  Serial.println(ledEventNew);
}

void stopSlowLedEvent()
{
  Serial.println("Stop the slow LED flash");
  // Stop the slow flash of the LED
  ledEventNew = 0;
}
```

Figure 16:Example 6 Code



```
2 second tick started id=2000
LED event started id=100
After event started id=8000
2 second tick: millis()2000
2 second tick: millis()4000
2 second tick: millis()6000
2 second tick: millis()8000
Stop the fast LED flash
```

```
Stop the fast LED flash
Start slow LED event id=2000
2 second tick: millis()10000
2 second tick: millis()12000
2 second tick: millis()14000
2 second tick: millis()16000
2 second tick: millis()18000
```

```
Stop the slow LED flash
2 second tick: millis()20000
2 second tick: millis()22000
2 second tick: millis()24000
2 second tick: millis()26000
2 second tick: millis()28000
2 second tick: millis()30000
2 second tick: millis()32000
2 second tick: millis()34000
2 second tick: millis()36000
2 second tick: millis()38000
```

Figure 17: Serial Monitor Output Example 6

At first, the LED flash quickly every 2 seconds, and prints on Serial Monitor Serial Monitor **time in seconds** (2000,4000,6000, 8000,...). Then after 8 seconds the LED will be stop flashing quickly and flash slowly 10 times. After 10 times the LED will flash quickly for 8 seconds then after 8 seconds will flash slow for 10 times. This process will be repeated.

The Serial Monitor output is used for debugging and keep track of the LED, also to provide information about the event’s timing which is (stop the fast LED flash, start slow LED event, stop the slow LED flash ….).

Question 2

Required: In this question we will print on the LCD "**ENCS4380: 2023" on the first row**, then we want to print "your name id#" which is **(Maha 1200746) in the second row**. Then the LCD should be update after 5 seconds and a new text should be **"the mission is DONE"** starting from the cursor (0,1) position and moving from the left to right until it goes out of position (16,2).

To solve this question and make sure that the code works correctly, I connected this circuit as shown in figure18 **,** using tinker cad program.

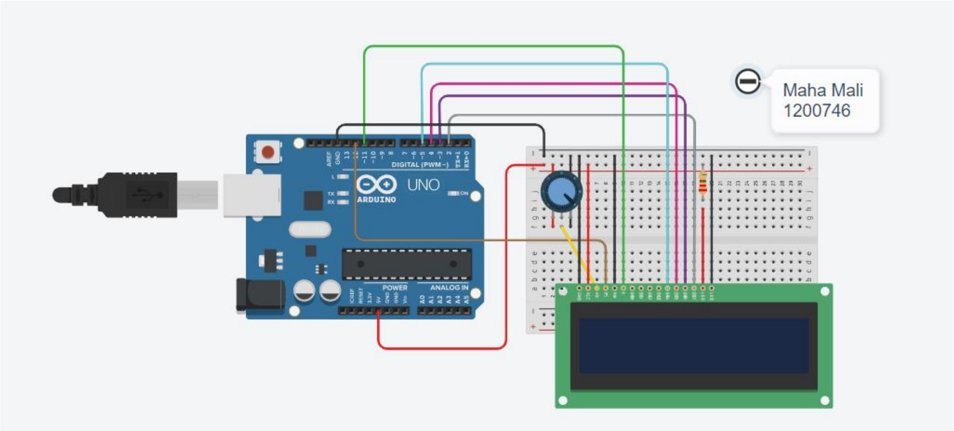


Figure 18: Question 2 Connection on Tinker Cad

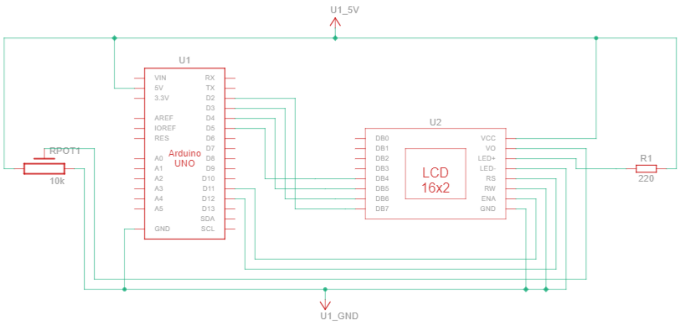
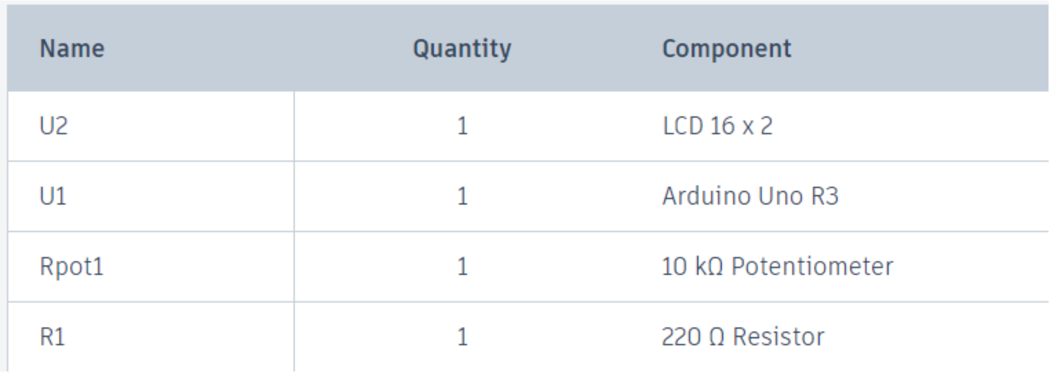


Figure 19: Question 2 Schematic View

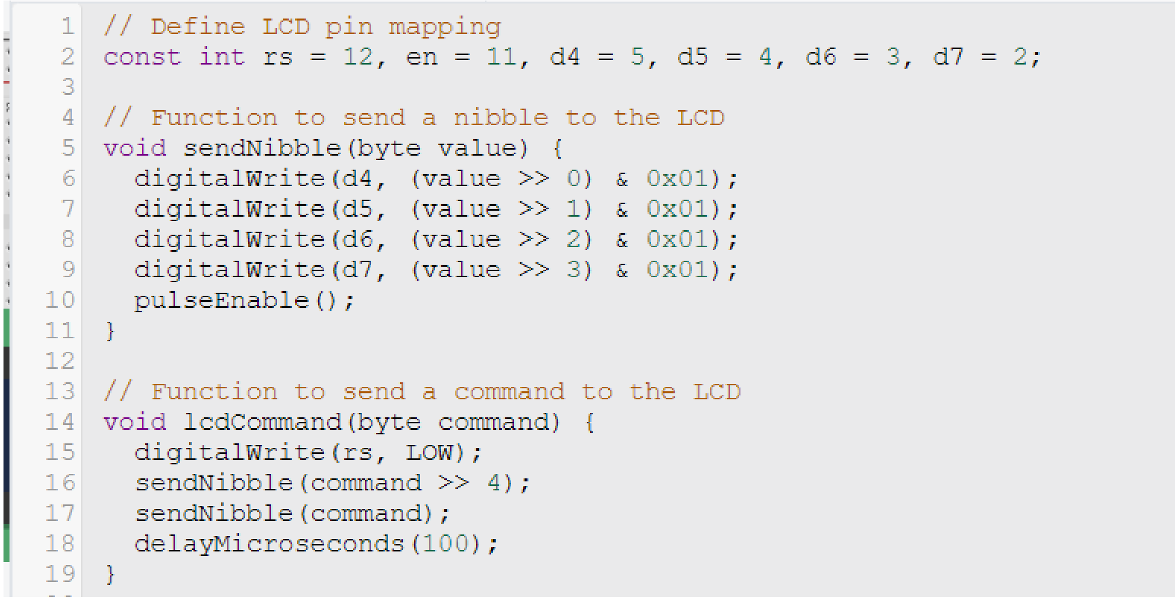


Name	Quantity	Component
U2	1	LCD 16 x 2
U1	1	Arduino Uno R3
Rpot1	1	10 kΩ Potentiometer
R1	1	220 Ω Resistor

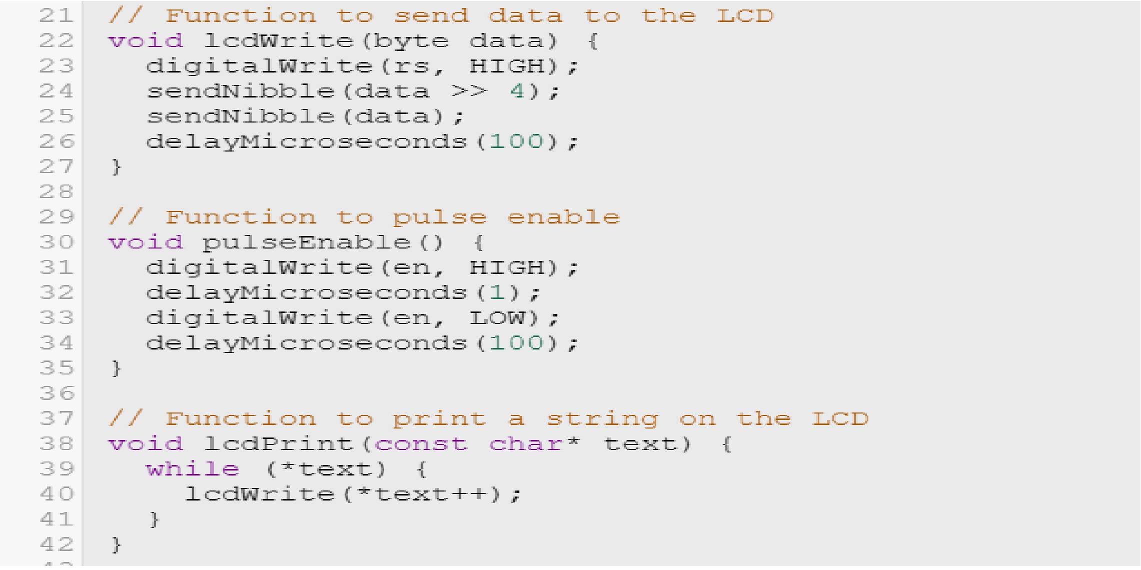
Figure 20: Question 2Component List

Tinker Cad Link: https://www.tinkercad.com/things/egVrUi0AzJj-lcd/editel?returnTo=%2Fdashboard%3Ftype%3Dcircuits%26collection%3Ddesigns&sharecode=viv4oVjTEH7P7pQSiv72Wy-NifqGcy7avOR2SPP0J3E

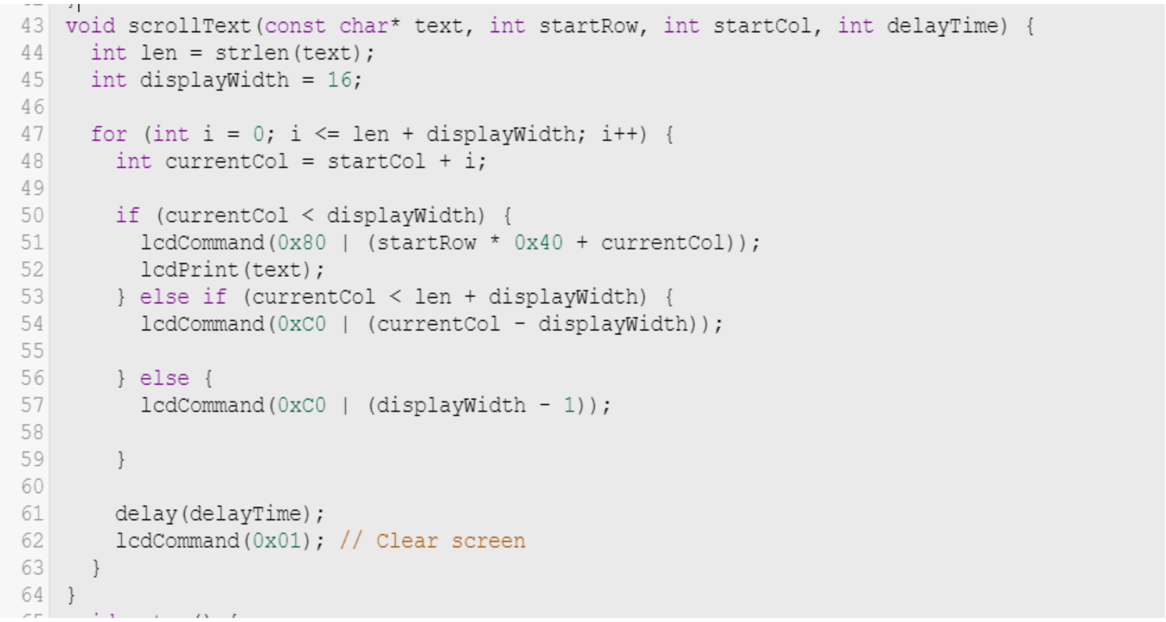
Code Description and Results



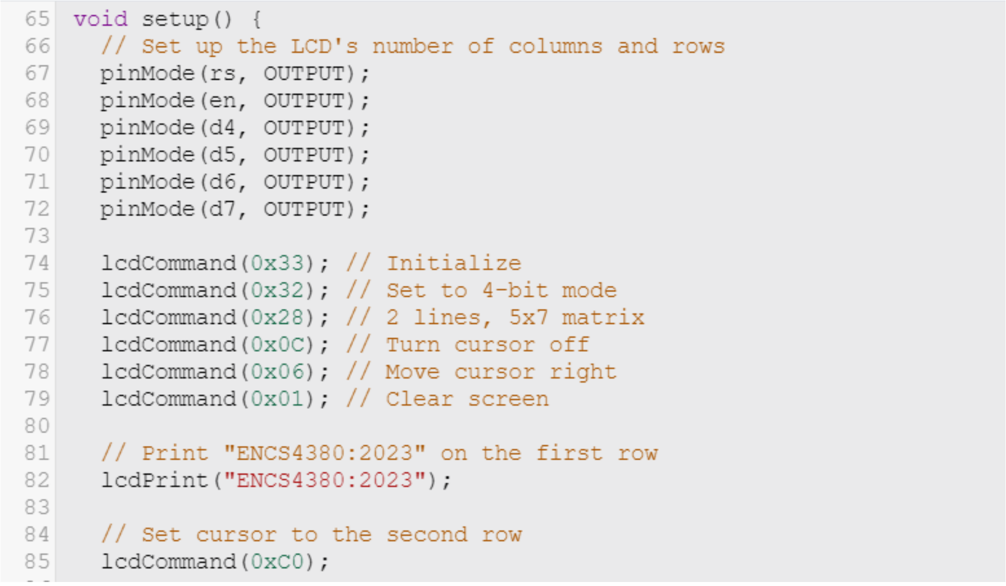
```
// Define LCD pin mapping
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;

// Function to send a nibble to the LCD
void sendNibble(byte value) {
  digitalWrite(d4, (value >> 0) & 0x01);
  digitalWrite(d5, (value >> 1) & 0x01);
  digitalWrite(d6, (value >> 2) & 0x01);
  digitalWrite(d7, (value >> 3) & 0x01);
  pulseEnable();
}

// Function to send a command to the LCD
void lcdCommand(byte command) {
  digitalWrite(rs, LOW);
  sendNibble(command >> 4);
  sendNibble(command);
  delayMicroseconds(100);
}
```



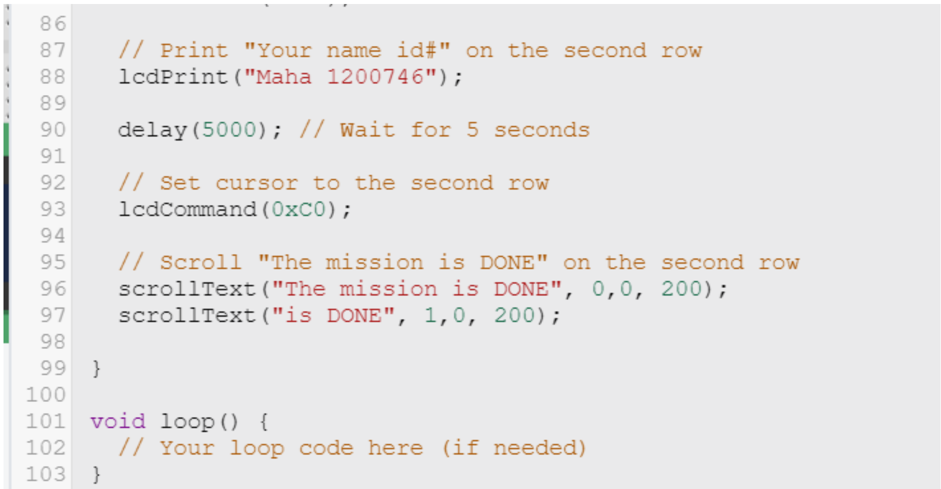
```
// Function to send data to the LCD
void lcdWrite(byte data) {
  digitalWrite(rs, HIGH);
  sendNibble(data >> 4);
  sendNibble(data);
  delayMicroseconds(100);
}

// Function to pulse enable
void pulseEnable() {
  digitalWrite(en, HIGH);
  delayMicroseconds(1);
  digitalWrite(en, LOW);
  delayMicroseconds(100);
}

// Function to print a string on the LCD
void lcdPrint(const char* text) {
  while (*text) {
    lcdWrite(*text++);
  }
}
```



```
void scrollText(const char* text, int startRow, int startCol, int delayTime) {
  int len = strlen(text);
  int displayWidth = 16;

  for (int i = 0; i <= len + displayWidth; i++) {
    int currentCol = startCol + i;

    if (currentCol < displayWidth) {
      lcdCommand(0x80 | (startRow * 0x40 + currentCol));
      lcdPrint(text);
    } else if (currentCol < len + displayWidth) {
      lcdCommand(0xC0 | (currentCol - displayWidth));

    } else {
      lcdCommand(0xC0 | (displayWidth - 1));

    }

    delay(delayTime);
    lcdCommand(0x01); // Clear screen
  }
}
```



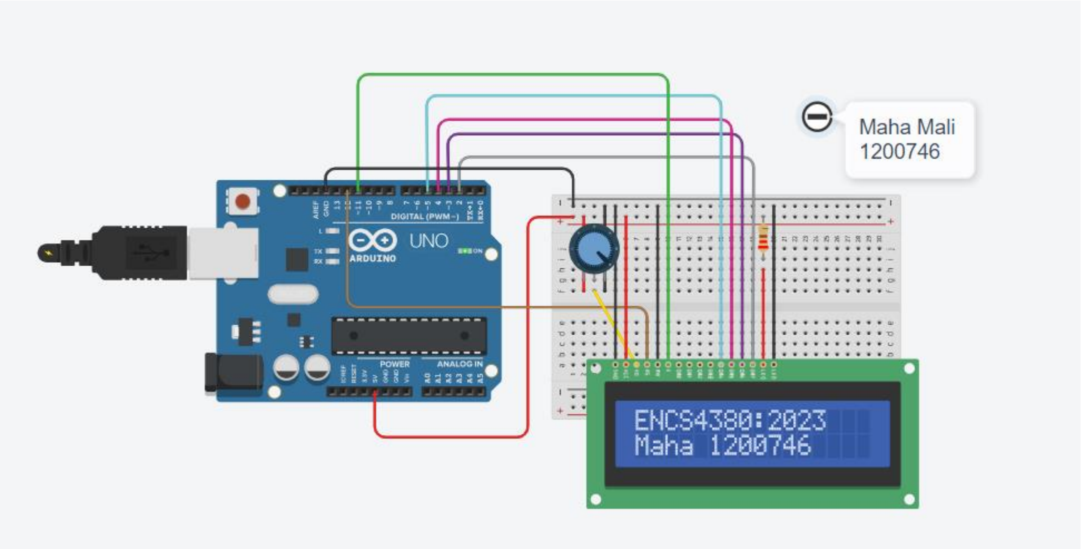
```
void setup() {
  // Set up the LCD's number of columns and rows
  pinMode(rs, OUTPUT);
  pinMode(en, OUTPUT);
  pinMode(d4, OUTPUT);
  pinMode(d5, OUTPUT);
  pinMode(d6, OUTPUT);
  pinMode(d7, OUTPUT);

  lcdCommand(0x33); // Initialize
  lcdCommand(0x32); // Set to 4-bit mode
  lcdCommand(0x28); // 2 lines, 5x7 matrix
  lcdCommand(0x0C); // Turn cursor off
  lcdCommand(0x06); // Move cursor right
  lcdCommand(0x01); // Clear screen

  // Print "ENCS4380:2023" on the first row
  lcdPrint("ENCS4380:2023");

  // Set cursor to the second row
  lcdCommand(0xC0);
```



```

  // Print "Your name id#" on the second row
  lcdPrint("Maha 1200746");

  delay(5000); // Wait for 5 seconds

  // Set cursor to the second row
  lcdCommand(0xC0);

  // Scroll "The mission is DONE" on the second row
  scrollText("The mission is DONE", 0,0, 200);
  scrollText("is DONE", 1,0, 200);

}

void loop() {
  // Your loop code here (if needed)
}
```

Figure 21:Qustion 2 Code

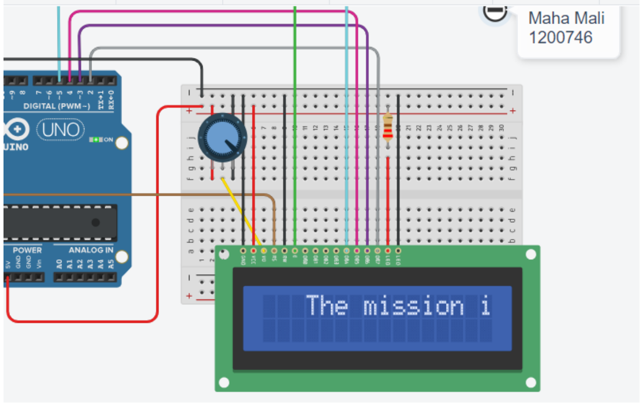
- At first, I checked that it prints **ENCS4380:2023** on the first row and **Maha 1200746** on the second row, for only 5 seconds. As shown in figure 21.



```
ENCS4380:2023
Maha 1200746
```

Figure 22:Print the Name and ID on the LCD

- Then I made sure that the text moved correctly from left to right, as shown in figure 22.



```
   The mission i
```

Figure 23:Text Move from Left to Right

- When text moves from left to right, the rest of the text appears on the second row of the screen.

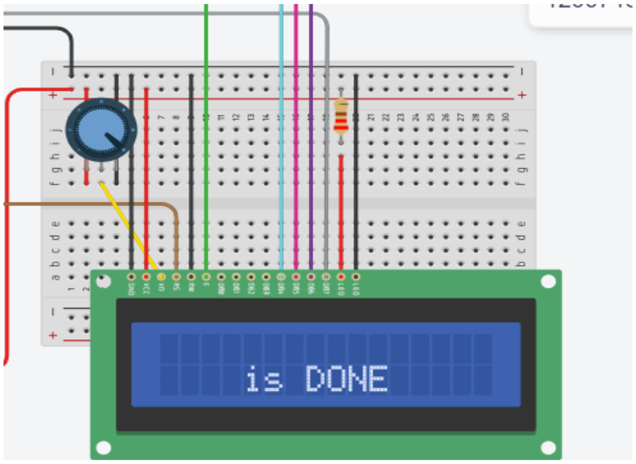


Figure 24:Rest of Text

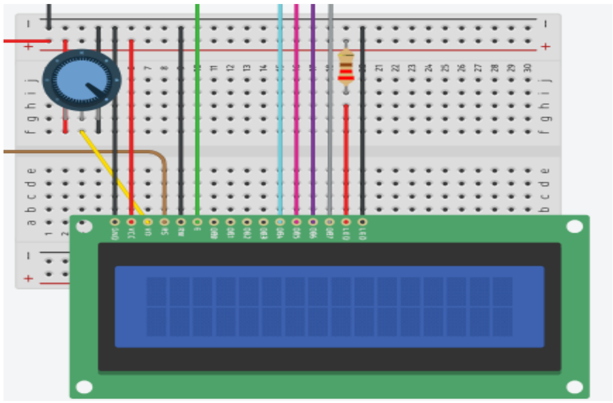


Figure 25:Empty Screen when going out position (16,2)

References

[1] https://www.arduino.cc/reference/en/language/functions/time/delay/ .
Accessed on 26-12-2023 at 11:50 AM.

[2] https://www.arduino.cc/reference/en/libraries/arduino-timer/ .
Accessed on 26-12-2023 at 12:50 PM.