## Laboratory work №8

**Topic:** Research on the organisation of time control functions on the ATMega328 microprocessor

**Goal**: To get some hands-on experience reading control signals from buttons, suppressing contact bounce, and organising several simultaneous time control functions using the ATMega328 microprocessor on the Arduino UNO R3 board.Ensure sequential flashing/turning off of LEDs connected to specified ports each time the buttons connected to specified ports are pressed.

Task options are listed in Table 1.

| | LED 1 | | | LED 2 | | |
|---|---|---|---|---|---|---|
| | Control button port number | LED connection port number | LED on-off / time, C | Control button port number | LED connection port number | LED on/off time, c |
| 2 | 3 | 4 | 0.2/1.2 | 5 | 6 | 1.8/2.5 |

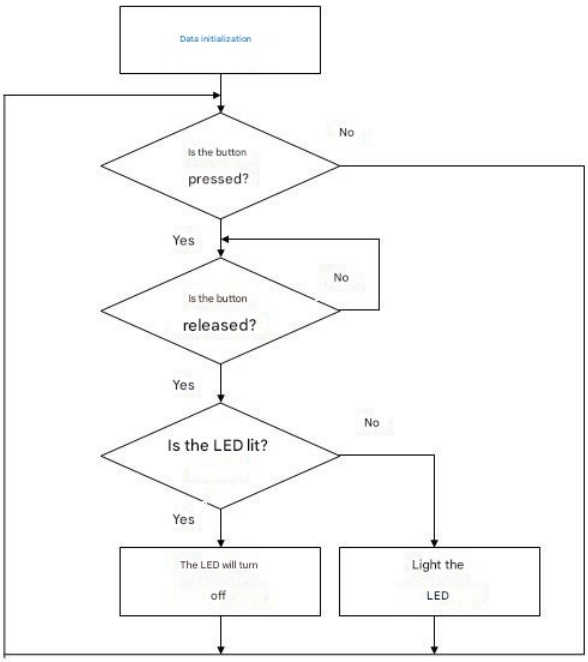


Figure 1 – Program algorithm

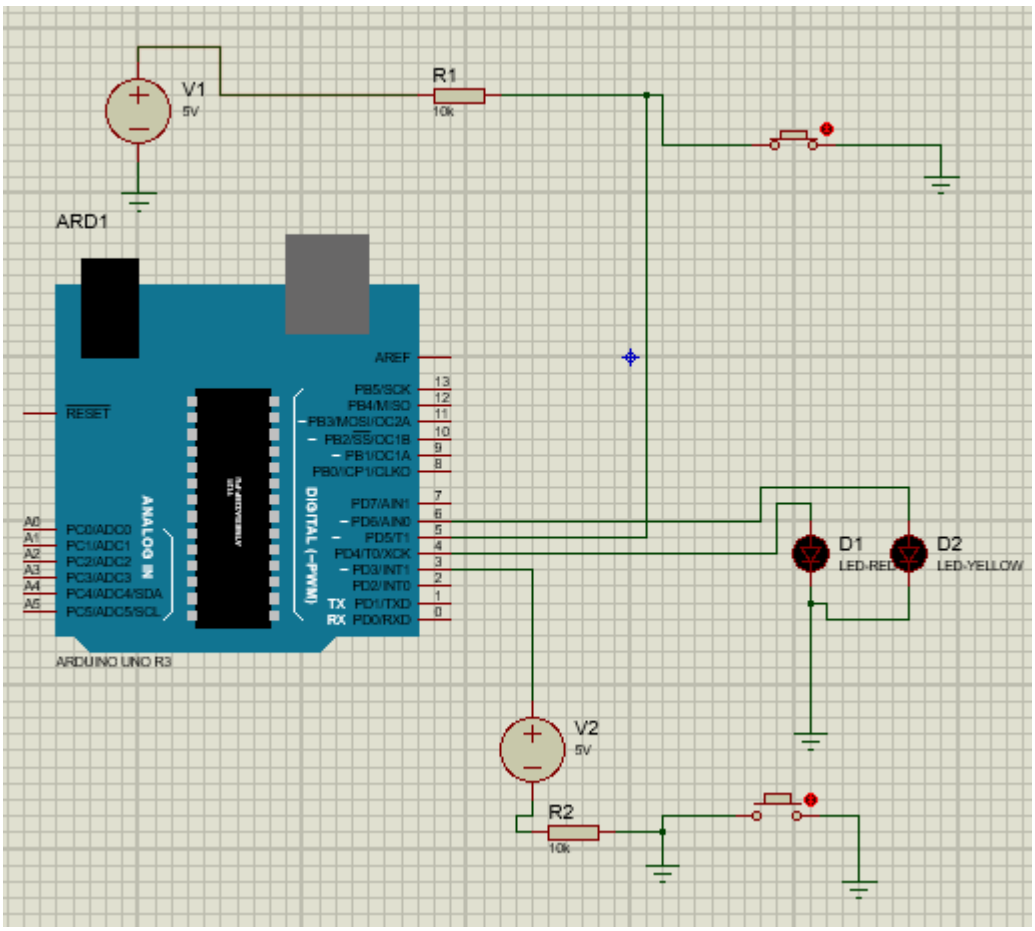


Figure 2 - Schema in Proteus

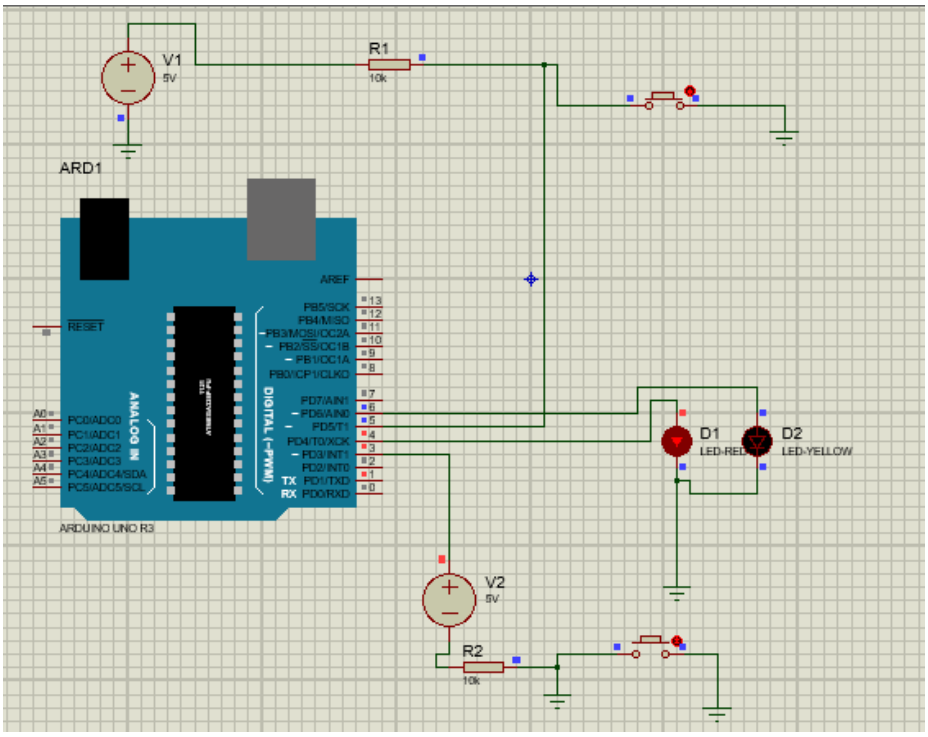


Figure 3 – Result of the work (LED 1 turns on)

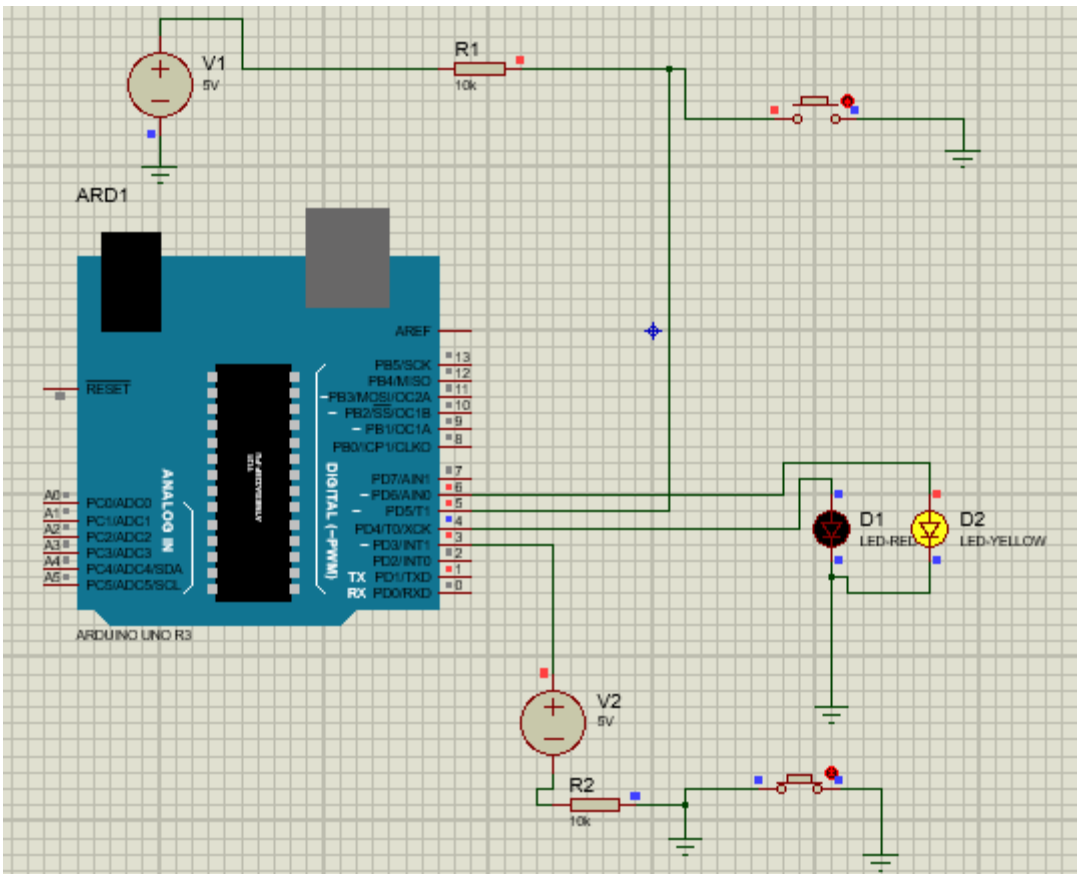


Figure 3 – Result of the work (LED 2 turns on)

**Program code**

```
int BUTTON1=3;
int BUTTON2=5;
boolean lastButton1 = LOW; // Variable to store
// the previous state of the button
boolean currentButton1 = LOW; // Variable to store the current
// state of the button
boolean ledOn1 = false; // Current state of the LED
// (on / off)
boolean lastButton2 = LOW; // Variable to store
// the previous state of the button
boolean currentButton2 = LOW; // Variable to store the current
// state of the button
boolean ledOn2 = false; // Current state of the LED
// (on / off)
class Flasher
{
// Class member variables
// Initialized at startup
```

```
int ledPin; // pin number with LED
long OnTime; // on time in milliseconds
long OffTime; // time when LED is off
// Current stateint ledState; // on/off state
unsigned long previousMillis; // last time the state changed
// The constructor creates a Flasher instance and initializes
// class member variables and state
public:
Flasher (int pin, long on, long off)
{
ledPin = pin;
pinMode (ledPin, OUTPUT);
OnTime = on;
OffTime = off;
ledState = LOW;
previousMillis = 0;
}
void Update ()
{
// check if it's time to change the LED state
unsigned long currentMillis = millis (); // current time in milliseconds
if ((ledState == HIGH) && (currentMillis - previousMillis>= OnTime))
{
ledState = LOW; // turn off
previousMillis = currentMillis; // remember the time
digitalWrite (ledPin, ledState); // implement a new state
}
else if ((ledState == LOW) && (currentMillis - previousMillis>= OffTime))
{
ledState = HIGH; // turn off
previousMillis = currentMillis; // save the time
digitalWrite (ledPin, ledState); // implement a new state
}
}
};
Flasher led1 (4, 200, 1200);
Flasher led2 (6, 1800, 2500);
void setup ()
{
Serial.begin(9600);
}
void loop ()
```

```
{
//led1
currentButton1 = debounce1 (lastButton1);
if (lastButton1 == LOW && currentButton1 == HIGH)
// if the button was pressed
{
ledOn1 = !ledOn1; // invert the LED state value
// output the ledOn state value to the serial port
// to monitor its value on the computer
// using the serial port monitor
Serial.println (ledOn1);
}
lastButton1 = currentButton1;
//led2
currentButton2 = debounce2 (lastButton1);
if (lastButton2 == LOW && currentButton2 == HIGH)
// if the button was pressed
{
ledOn2 = !ledOn2; // invert the LED state value
// output the ledOn state value to the serial port
// to monitor its value on the computer
// using the serial port monitor
Serial.println (ledOn2);
}
lastButton2 = currentButton2;
if(ledOn1==true)
{
led1.Update ();
}
if(ledOn2==true)
{
led2.Update ();
}
}
boolean debounce1 (boolean last1) {
boolean current1 = digitalRead (BUTTON1); // Read the button state
if (last1 != current1) // if changed ...
{
delay (5); // wait 5 ms
current1 = digitalRead (BUTTON1); // read the button state
return current1; // return the button state
}
```

```
}

boolean debounce2 (boolean last2) {
boolean current2 = digitalRead (BUTTON2); // Read the button state
if (last2 != current2) // if changed ...
{
delay (5); // wait 5 ms
current2= digitalRead (BUTTON2); // read the button state
return current2; // return the button state
}}
```

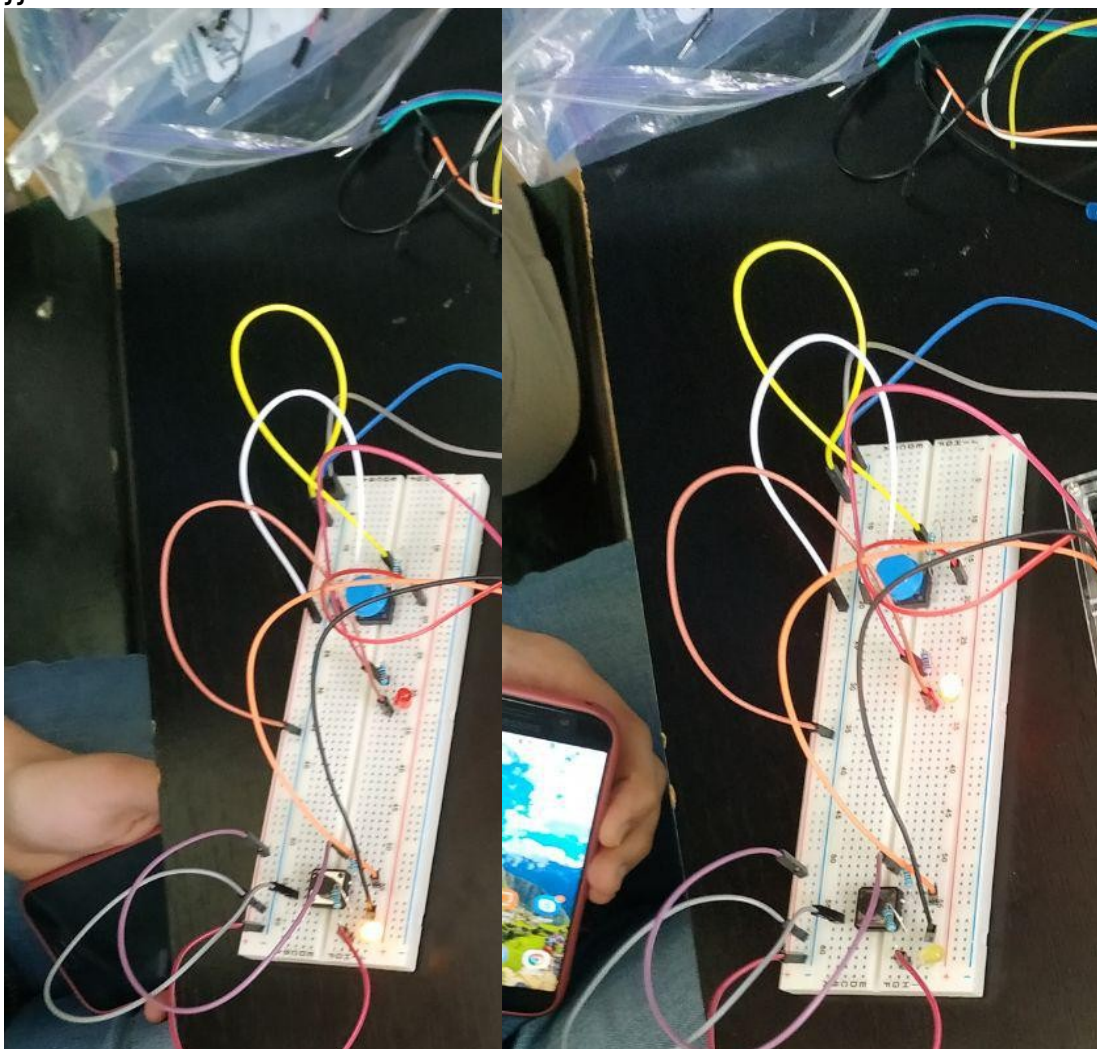

Figure 5 - Result of the robot's work

## Conclusion

We studied the organisation of time control functions on a microprocessor.

We gained practical skills in reading control signals from buttons, suppressing contact noise, and organising several simultaneous time

control functions using an ATMega328 microprocessor on an Arduino UNO R3 board.

ст.гр. КІТ-36
Надірян Г.О.
Варіант-12

## Лабораторна робота №8

**Тема:** Дослідження організації часових функцій керування на мікропроцесорі ATMega328

**Мета:** Одержання практичних навичок зчитування з кнопок керуючих сигналів, придушення брязкоту контактів та організації декількох одночасних часових функцій керування з використанням мікропроцесора ATMega328 на платі Arduino UNO R3.

Забезпечити почергове блимкання/виключення світлодіодів, підключених до заданих портів при кожному натисканні на кнопки, що підключені до заданих портів. Варіанти завдань наведені у таблиці 1.

| | Світлодіод 1 | | | Світлодіод 2 | | |
|---|---|---|---|---|---|---|
| | N порту управляючої кнопки | N порту підключення світлодіода | Час включення/ виключення світлодіоду, с | N порту управляючої кнопки | N порту підключення світлодіода | Час включення/ виключення світлодіоду, с |
| 2 | 3 | 4 | 0,2/1,2 | 5 | 6 | 1,8/2,5 |

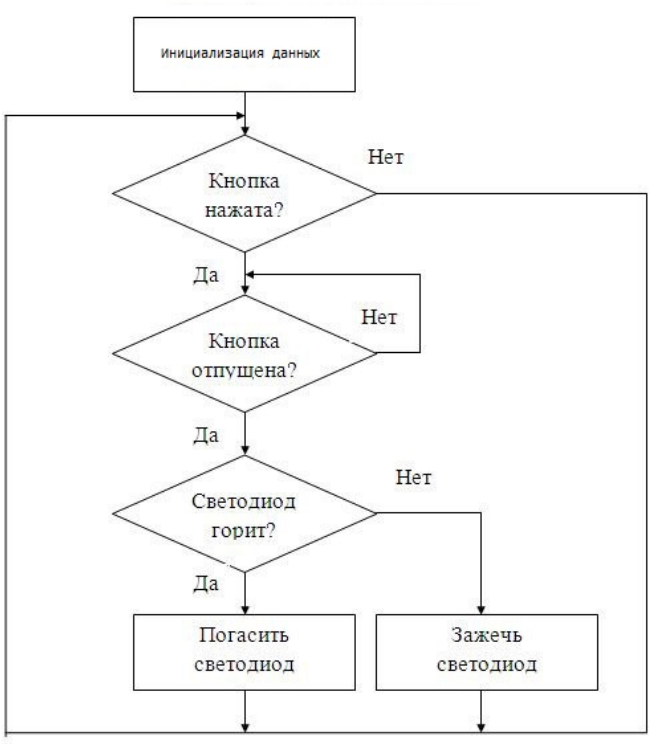


Рисунок 1 – алгоритм програми

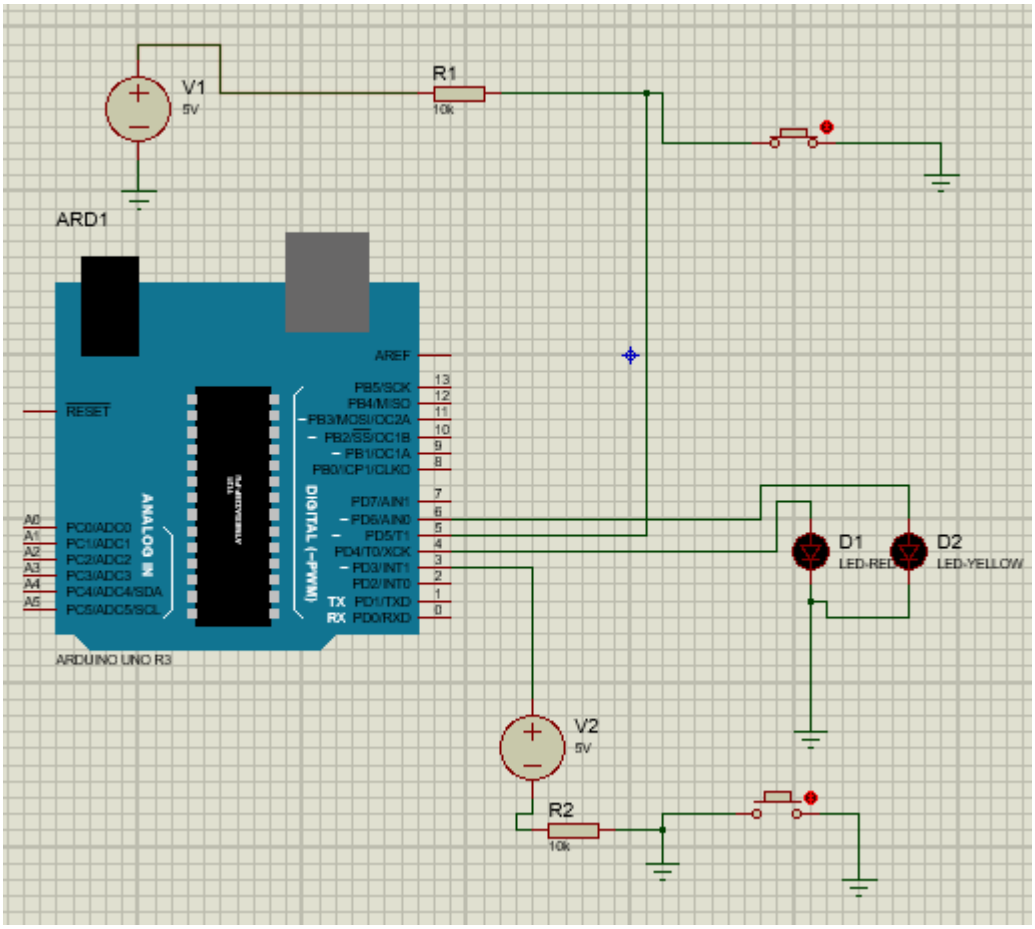


Рисунок 2 - Схема в Proteus

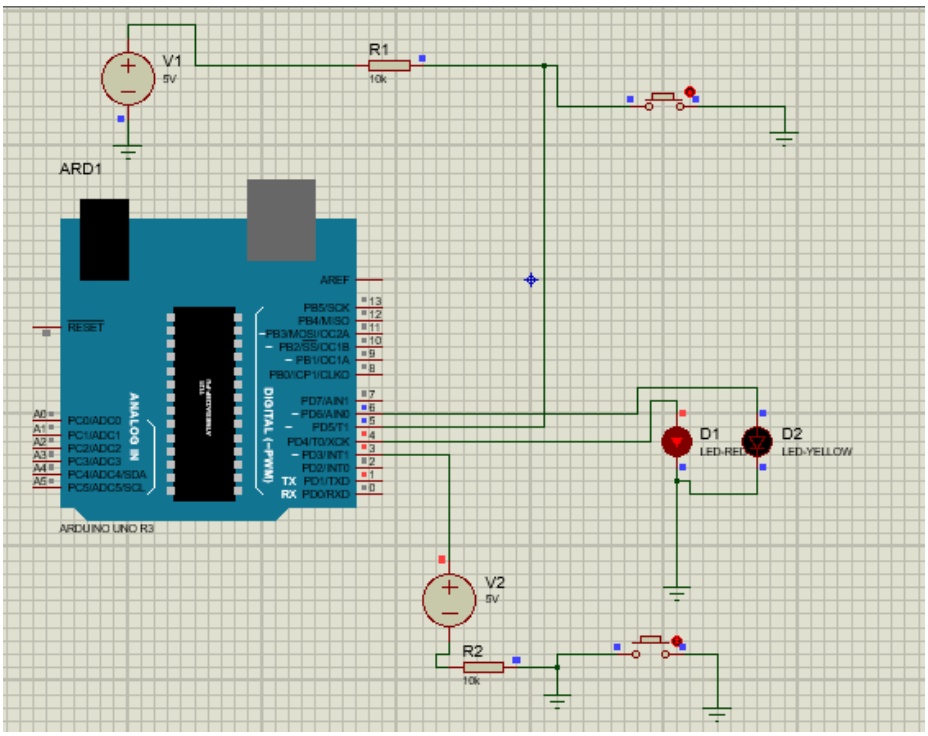


Рисунок 3 – Результат роботи  (включення світлодіоду 1)

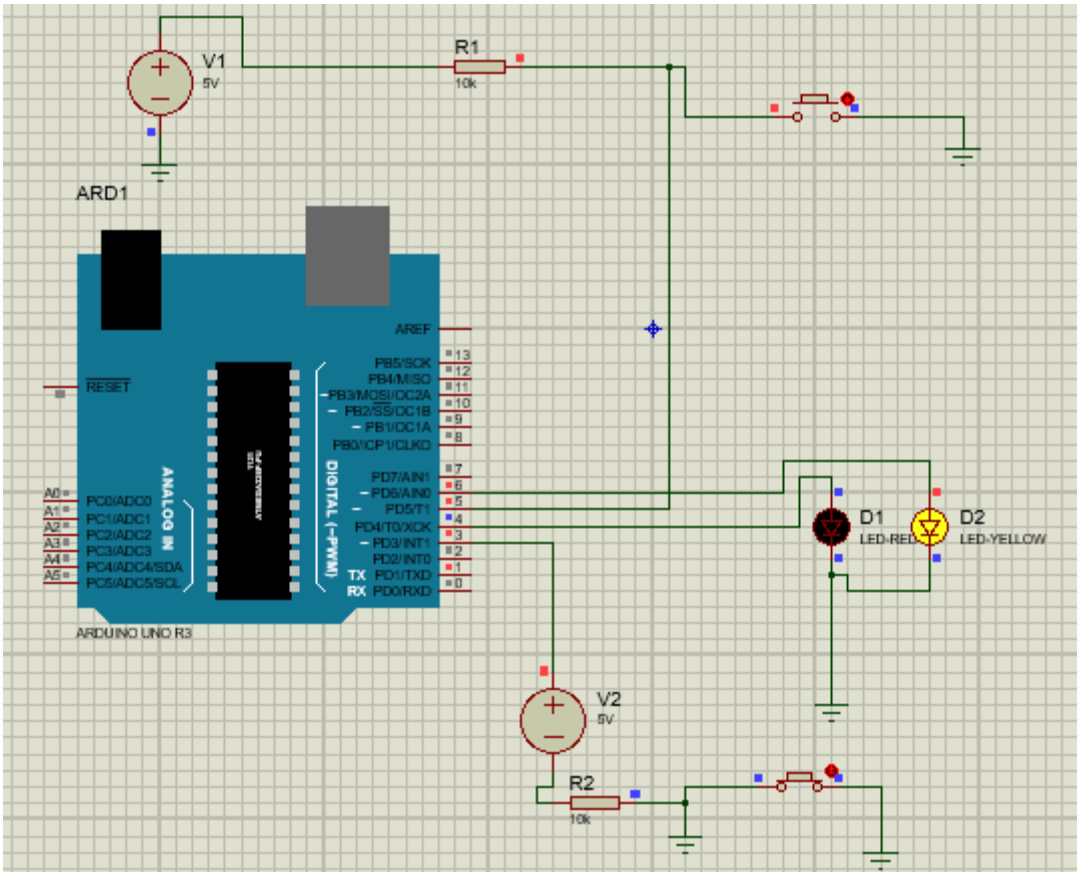


Рисунок 3 – Результат роботи  (включення світлодіоду 2)

**Код програми**

```
int BUTTON1=3;
```

```
int BUTTON2=5;
boolean lastButton1 = LOW; // Змінна для збереження
// попереднього стану кнопки
boolean currentButton1 = LOW; // Змінна для збереження поточного
// стану кнопки
boolean ledOn1 = false; // Поточний стан світлодіода
// (включений / виключений)
boolean lastButton2 = LOW; // Змінна для збереження
// попереднього стану кнопки
boolean currentButton2 = LOW; // Змінна для збереження поточного
// стану кнопки
boolean ledOn2 = false; // Поточний стан світлодіода
// (включений / виключений)
class Flasher
{
 // Змінні - члени класу
 // Ініціалізуються при запуску
 int ledPin; // номер піна із світлодіодом
 long OnTime; // час включення в мілісекундах
 long OffTime; // час, коли світлодіод вимкнений
 // Поточний стан
 int ledState; // стан вмикання / вимикання
 unsigned long previousMillis; // останній момент зміни стану
 // Конструктор створює екземпляр Flasher і ініціалізує
 // змінні-члени класу і стан
 public:
 Flasher (int pin, long on, long off)
 {
 ledPin = pin;
 pinMode (ledPin, OUTPUT);
 OnTime = on;
 OffTime = off;
 ledState = LOW;
 previousMillis = 0;
 }
 void Update ()
 {
 // з'ясовуємо чи не настав момент змінити стан світлодіода
 unsigned long currentMillis = millis (); // поточний час в мілісекундах
 if ((ledState == HIGH) && (currentMillis - previousMillis>= OnTime))
 {
 ledState = LOW; // вимикаємо
```

```
 previousMillis = currentMillis; // запам'ятовуємо момент часу
 digitalWrite (ledPin, ledState); // реалізуємо новий стан
 }
 else if ((ledState == LOW) && (currentMillis - previousMillis>= OffTime))
 {
 ledState = HIGH; // вимикаємо
 previousMillis = currentMillis; // запам'ятовуємо момент часу
 digitalWrite (ledPin, ledState); // реалізуємо новий стан
 }
 }
};
Flasher led1 (4, 200, 1200);
Flasher led2 (6, 1800, 2500);
void setup ()
{
  Serial.begin(9600);
}
void loop ()
{
  //led1
  currentButton1 = debounce1 (lastButton1);
  if (lastButton1 == LOW && currentButton1 == HIGH)
    // якщо кнопку натиснули
  {
    ledOn1 = !ledOn1; // інвертувати значення стану світлодіода
    // вивід значення стану ledOn у послідовний порт
    // для відслідковування його значення на комп'ютері
    // за допомогою монітору послідовного порта
    Serial.println (ledOn1);
  }
 lastButton1 = currentButton1;
 //led2
 currentButton2 = debounce2 (lastButton1);
  if (lastButton2 == LOW && currentButton2 == HIGH)
    // якщо кнопку натиснули
  {
    ledOn2 = !ledOn2; // інвертувати значення стану світлодіода
    // вивід значення стану ledOn у послідовний порт
    // для відслідковування його значення на комп'ютері
    // за допомогою монітору послідовного порта
    Serial.println (ledOn2);
  }
```

```
 lastButton2 = currentButton2;
 if(ledOn1==true)
 {
   led1.Update ();
 }
 if(ledOn2==true)
 {
   led2.Update ();
 }
}
boolean debounce1 (boolean last1) {
  boolean current1 = digitalRead (BUTTON1); // Зчитати стан кнопки
  if (last1 != current1) // якщо змінилося ...
  {
    delay (5); // чекаємо 5 мс
    current1 = digitalRead (BUTTON1); // зчитуємо стан кнопки
    return current1; // повертаємо стан кнопки
  }
}

boolean debounce2 (boolean last2) {
  boolean current2 = digitalRead (BUTTON2); // Зчитати стан кнопки
  if (last2 != current2) // якщо змінилося ...
  {
    delay (5); // чекаємо 5 мс
    current2= digitalRead (BUTTON2); // зчитуємо стан кнопки
    return current2; // повертаємо стан кнопки
  }
}
```

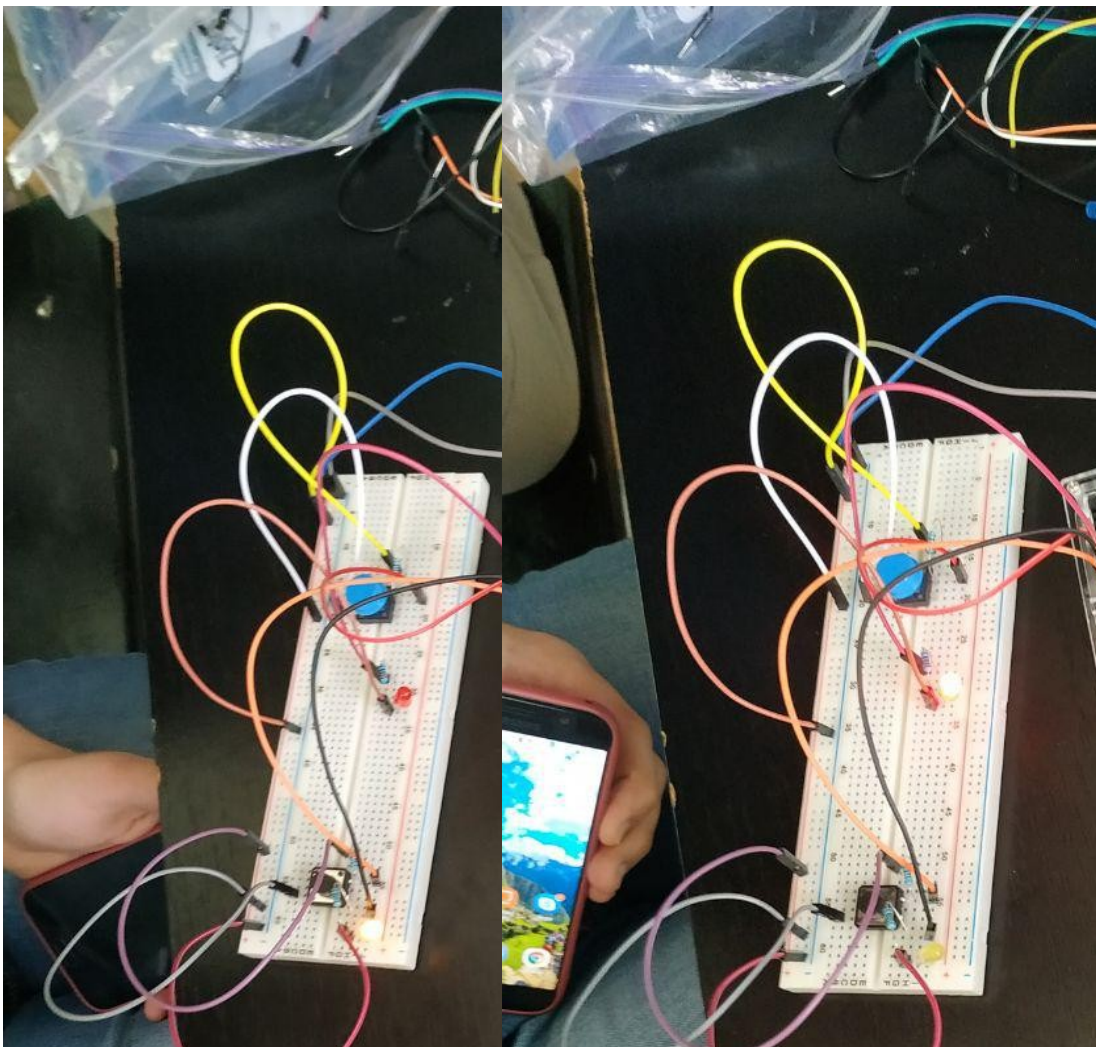

Рисунок 5 - Результат роботи

Дослідили організацію часових функцій керування на мікропроцесорі. Одержали практичні навичок зчитування з кнопок керуючих сигналів, придушення брязкоту контактів та організації декількох одночасних часових функцій керування з використанням мікропроцесора ATMega328 на платі Arduino UNO R3.