

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”

Кафедра САП



ЗВІТ

до виконання лабораторної роботи №8

На тему: ”Дослідження роботи датчика температури і вологості DHT11”

з курсу “Мікропроцесорні системи ”

Варіант - 4

Виконав:

Студент гр.ПП-31

Гаврилюк Н. О.

Прийняв:

Доцент кафедри САП

Головатий А.І.

ЛЬВІВ - 2025

Мета: ознайомитись з принципом роботи та зчитуванням даних датчика температури та вологості DHT11; закріпити навички виведення інформації на семисегментний індикатор з використанням регістру зсуву 74HC595 та LCD-індикатор.

Завдання

1. Реалізувати програму, яка виводить на LCD-індикатор значення температури та вологості з датчика DHT11 та керує RGB світлодіодом. Якщо $t > 18^{\circ}\text{C}$, то світиться синій світлодіод; якщо $t \geq 25^{\circ}\text{C}$, то світиться зелений світлодіод; якщо $t \geq 33^{\circ}\text{C}$, то світиться червоний світлодіод; якщо $t \leq 18^{\circ}\text{C}$, то RGB світлодіод не світиться.

Код програми Arduino IDE:

```
#include <LiquidCrystal.h>

#include "DHT.h"

#define R_pin A1
#define G_pin A2
#define B_pin A3
#define DHTPIN A0

DHT dht(DHTPIN, DHT11);

// Підключення LCD: RS, E, D4, D5, D6, D7
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

int temp, hum;

void RGB(int temp)
{
    if(temp >= 33)
    {
        digitalWrite(R_pin, HIGH);
        digitalWrite(G_pin, LOW);
        digitalWrite(B_pin, LOW);
    }
}
```

```

else if(temp>=25)
{
    digitalWrite(R_pin,LOW);
    digitalWrite(G_pin,HIGH);
    digitalWrite(B_pin,LOW);
}
else if(temp>18)
{
    digitalWrite(R_pin,LOW);
    digitalWrite(G_pin,LOW);
    digitalWrite(B_pin,HIGH);
}
else
{
    digitalWrite(R_pin,LOW);
    digitalWrite(G_pin,LOW);
    digitalWrite(B_pin,LOW);
}
}

void setup() {
    lcd.begin(16, 2);
    analogReference(INTERNAL); // встановлюємо опорну напругу 1.1 В
    pinMode(R_pin, OUTPUT);
    pinMode(G_pin, OUTPUT);
    pinMode(B_pin, OUTPUT);
    dht.begin();
}

```

```

void loop() {

    temp =dht.readTemperature(); // правильне перетворення у °C

    hum =dht.readHumidity();

    RGB(temp);

    lcd.clear();

    lcd.print("Temp: ");

    lcd.print(temp);

    lcd.print(" C");

    lcd.setCursor(0,1);

    lcd.print("Humidity: ");

    lcd.print(hum);

    lcd.print(" %");

    delay(1000);

}

```

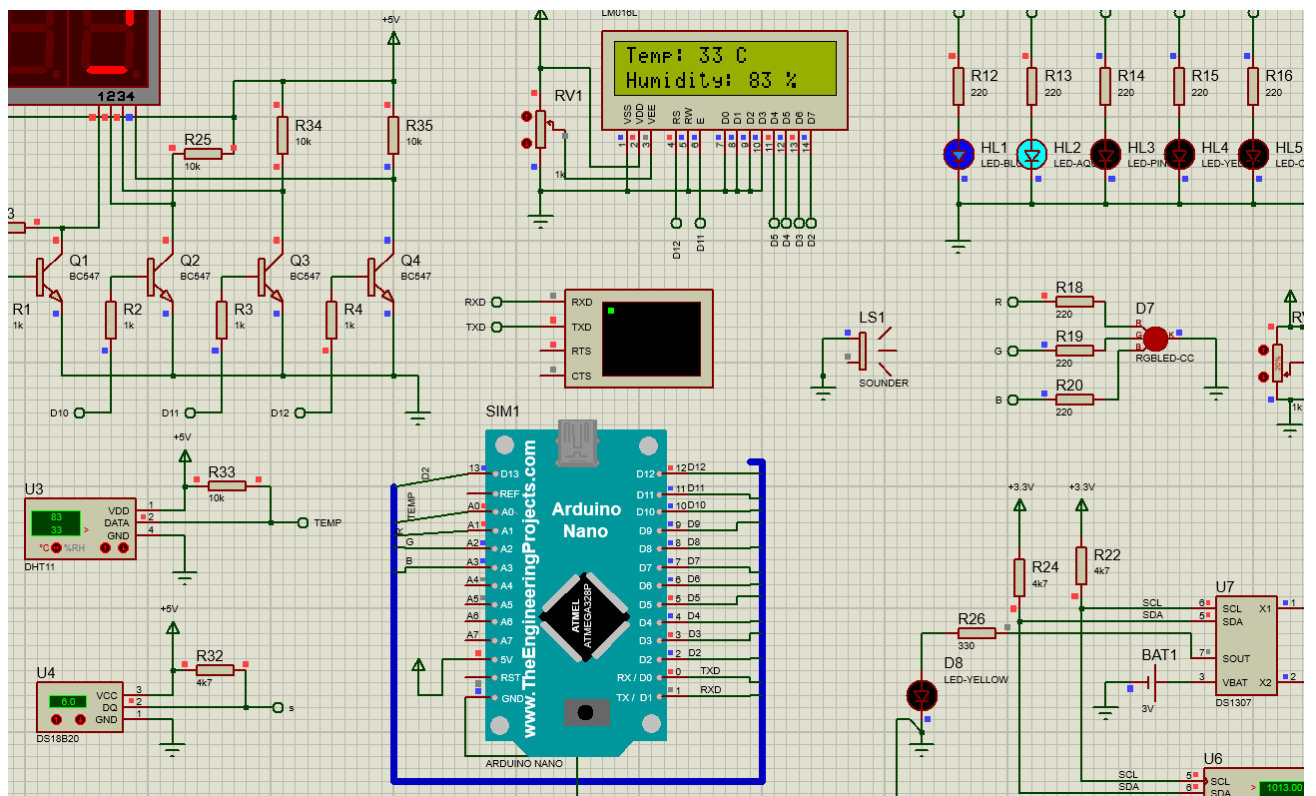


Рис. 1. Результат програми

Код програми МК AVR:

```
#define F_CPU 16000000UL

#include <avr/io.h>

#include <util/delay.h>

#include <stdio.h> //input/output library

#include "lcd_lib.h" // LCD library


// ----- LCD -----

#define LCD_RS PD6

#define LCD_E PD7

#define LCD_D4 PD5

#define LCD_D5 PD4

#define LCD_D6 PD3

#define LCD_D7 PD2

#define DHT_PORT PORTA

#define DHT_DDR DDRA

#define DHT_PINR PINA

#define DHT_PIN PA3


int temp, hum;


uint8_t dht_read(uint8_t *temp, uint8_t *hum);


// ----- RGB -----

#define R_PIN PC1
```

```
#define G_PIN PC2
```

```
#define B_PIN PC3
```

```
void RGB(int temp)
```

```
{
```

```
if(temp >= 33) {
```

```
    PORTC |= (1<<R_PIN);
```

```
    PORTC &= ~(1<<G_PIN);
```

```
    PORTC &= ~(1<<B_PIN);
```

```
}
```

```
else if(temp >= 25) {
```

```
    PORTC &= ~(1<<R_PIN);
```

```
    PORTC |= (1<<G_PIN);
```

```
    PORTC &= ~(1<<B_PIN);
```

```
}
```

```
else if(temp > 18) {
```

```
    PORTC &= ~(1<<R_PIN);
```

```
    PORTC &= ~(1<<G_PIN);
```

```
    PORTC |= (1<<B_PIN);
```

```
}
```

```
else {
```

```
    PORTC &= ~(1<<R_PIN);
```

```
    PORTC &= ~(1<<G_PIN);
```

```
    PORTC &= ~(1<<B_PIN);
```

```
}
```

```
}
```

```
//
```

```
=====
```

```
int main(void)
```

```
{
```

```
  DDRD |=
```

```
(1<<LCD_RS)|(1<<LCD_E)|(1<<LCD_D4)|(1<<LCD_D5)|(1<<LCD_D6)|(1<<LCD_D7);
```

```
  DDRC |= (1<<R_PIN)|(1<<G_PIN)|(1<<B_PIN);  // RGB
```

```
  LCDinit();
```

```
  char buf[5];
```

```
  char buf2[5];
```

```
  while(1)
```

```
  {
```

```
    dht_read(&temp, &hum);
```

```
        sprintf(buf, "%d", temp);
```

```
        sprintf(buf2, "%d", hum);
```

```
        RGB(temp);
```

```
        LCDclr();
```

```
        LCDcursorOFF();
```

```
        LCDstring(" Temp: ");
```

```
        LCDstring(buf);
```

```
        LCDstring(" C");
```

```
LCDGotoXY(0,1);
```

```
LCDstring("Humidity: ");
```

```
LCDstring(buf2);
```

```
LCDstring(" %");
```

```
    _delay_ms(1000);
```

```
}
```

```
}
```

```
uint8_t dht_read(uint8_t *temp, uint8_t *hum)
```

```
{
```

```
    uint8_t data[5] = {0};
```

```
    // стартовий сигнал
```

```
    DHT_DDR |= (1<<DHT_PIN);    // PA4 → вихід
```

```
    DHT_PORT &= ~(1<<DHT_PIN);  // низький рівень
```

```
    _delay_ms(20);
```

```
    DHT_PORT |= (1<<DHT_PIN);   // високий рівень
```

```
    _delay_us(40);
```

```
    DHT_DDR &= ~(1<<DHT_PIN);   // PA4 → вхід
```

```
    // чекати відповіді DHT11
```

```
    uint16_t timeout = 0;
```



```

while((DHT_PINR & (1<<DHT_PIN))) { if(timeout++ > 10000) return 1; }

while(!(DHT_PINR & (1<<DHT_PIN)));
while( (DHT_PINR & (1<<DHT_PIN)));

// приём 40 бит
for(uint8_t i=0; i<40; i++)
{
    while(!(DHT_PINR & (1<<DHT_PIN)));
    __delay_us(30);

    if(DHT_PINR & (1<<DHT_PIN))
        data[i/8] |= (1 << (7 - (i%8)));

    while(DHT_PINR & (1<<DHT_PIN));
}

if((data[0] + data[1] + data[2] + data[3]) != data[4]) return 2;

*hum = data[0];
*temp = data[2];

return 0;
}

```

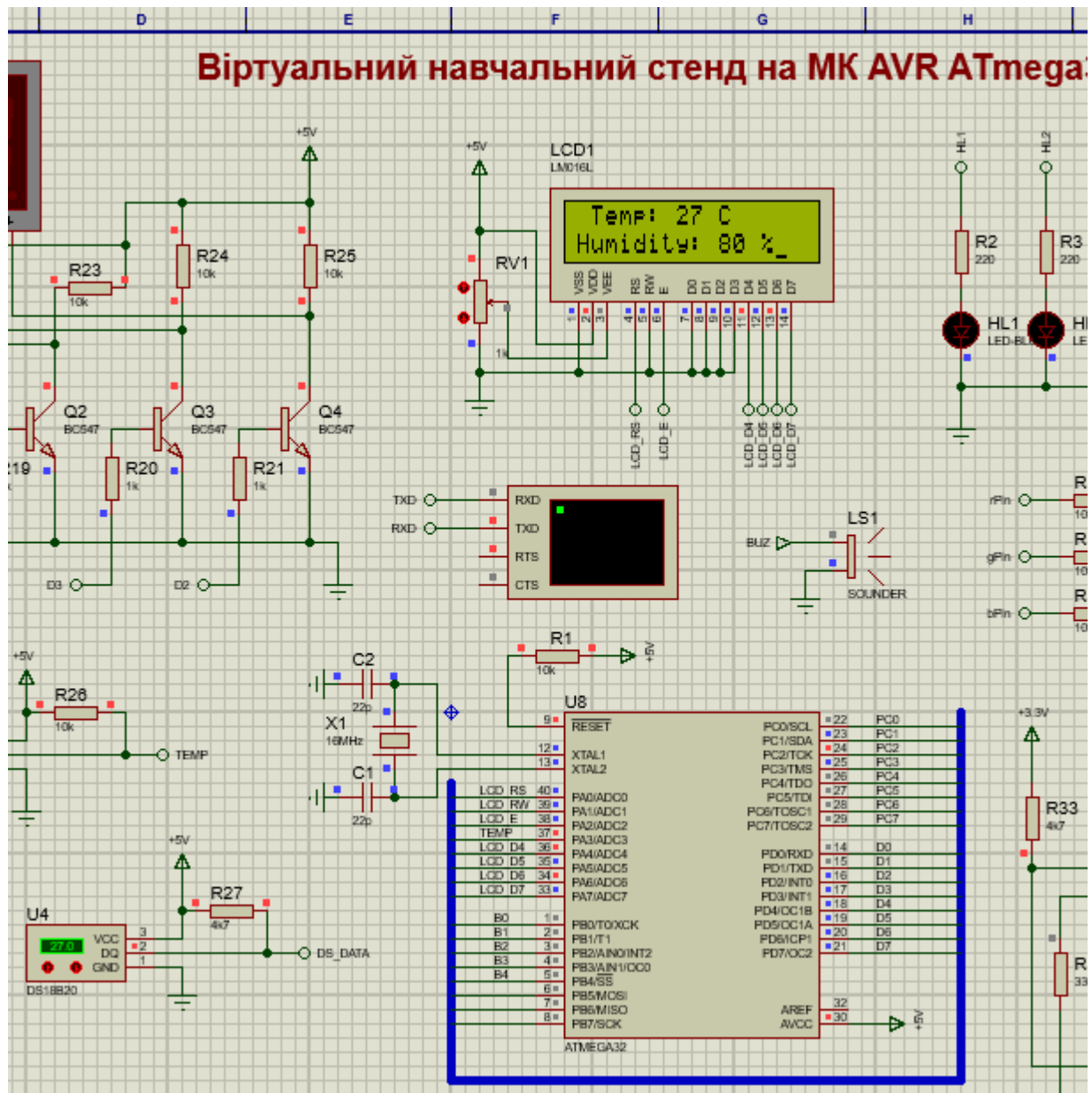


Рис. 2.Результат програми

2. Реалізувати програму, яка виводить на семисегментний індикатор значення температури та вологості з датчика DHT11 та керує RGB світлодіодом. Якщо вологість $h < 40\%$, то світиться синій світлодіод; якщо $60\% \geq h \geq 40\%$, то світиться зелений світлодіод; якщо $h > 60\%$, то світиться червоний світлодіод.

Код програми Arduino IDE:

```
#include <LiquidCrystal.h>

#include "DHT.h"

#define R_pin A1
#define G_pin A2
#define B_pin A3
```

```

#define DHTPIN A0

DHT dht(DHTPIN, DHT11);

// Підключення LCD: RS, E, D4, D5, D6, D7

const int numeral[10] = {

    B11111100,//0

    B01100000,//1

    B11011010,//2

    B11110010,//3

    B01100110,//4

    B10110110,//5

    B10111110,//6

    B11100000,//7

    B11111110,//8

    B11110110,//9

};

int temp,hum;

// H,G,F,E,D,C,B,A

const int segmentPins[] = {13, 8, 7, 6, 5, 4, 3, 2};

const int nbrDigits = 4;


// digital 0 1 2 3

const int digitPins[nbrDigits] = {9, 10, 11, 12};


void RGB(int temp)

{

    if(hum>60)

    {

        digitalWrite(R_pin,HIGH);

```

```

        digitalWrite(G_pin,LOW);
        digitalWrite(B_pin,LOW);
    }
    else if(hum>=40)
    {
        digitalWrite(R_pin,LOW);
        digitalWrite(G_pin,HIGH);
        digitalWrite(B_pin,LOW);
    }
    else if(hum<40)
    {
        digitalWrite(R_pin,LOW);
        digitalWrite(G_pin,LOW);
        digitalWrite(B_pin,HIGH);
    }
    else
    {
        digitalWrite(R_pin,LOW);
        digitalWrite(G_pin,LOW);
        digitalWrite(B_pin,LOW);
    }
}

void setup() {
    for(int i = 0; i < 8; i++) {
        pinMode(segmentPins[i], OUTPUT);
    }
    for(int i = 0; i < nbrDigits; i++) {

```

```
    pinMode(digitPins[i], OUTPUT);
}

analogReference(INTERNAL); // встановлюємо опорну напругу 1.1 В
pinMode(R_pin, OUTPUT);
pinMode(G_pin, OUTPUT);
pinMode(B_pin, OUTPUT);
Serial.begin(9600);
dht.begin();
}
```

```
void showNumber(int number) {
    if(number == 0) {
        showDigit(0, nbrDigits - 1);
    } else {
        for(int digit = nbrDigits - 1; digit >= 0; digit--) {
            if(number > 0) {
                showDigit(number % 10, digit);
                number = number / 10;
            }
        }
    }
}
```

```
void showDigit(int number, int digit)
{
    //Serial.println(number);
    //Serial.println(digit);
    digitalWrite(digitPins[digit], HIGH);
}
```

```

for(int segment = 1; segment < 8; segment++) {
    boolean isBitSet = bitRead(numeral[number], segment);
    digitalWrite(segmentPins[segment], isBitSet);
}
delay(5);
digitalWrite(digitPins[digit], LOW);
}

int time;
void loop() {
    time++;
    temp =dht.readTemperature(); // правильне перетворення у °C
    hum =dht.readHumidity();
    Serial.println(temp);
    RGB(temp);
    if(time<50)
    {
        showNumber(temp);
    }
    else if(time<100)
    {
        showNumber(hum);
    }
    else
    {
        time=0;
    }
}

```

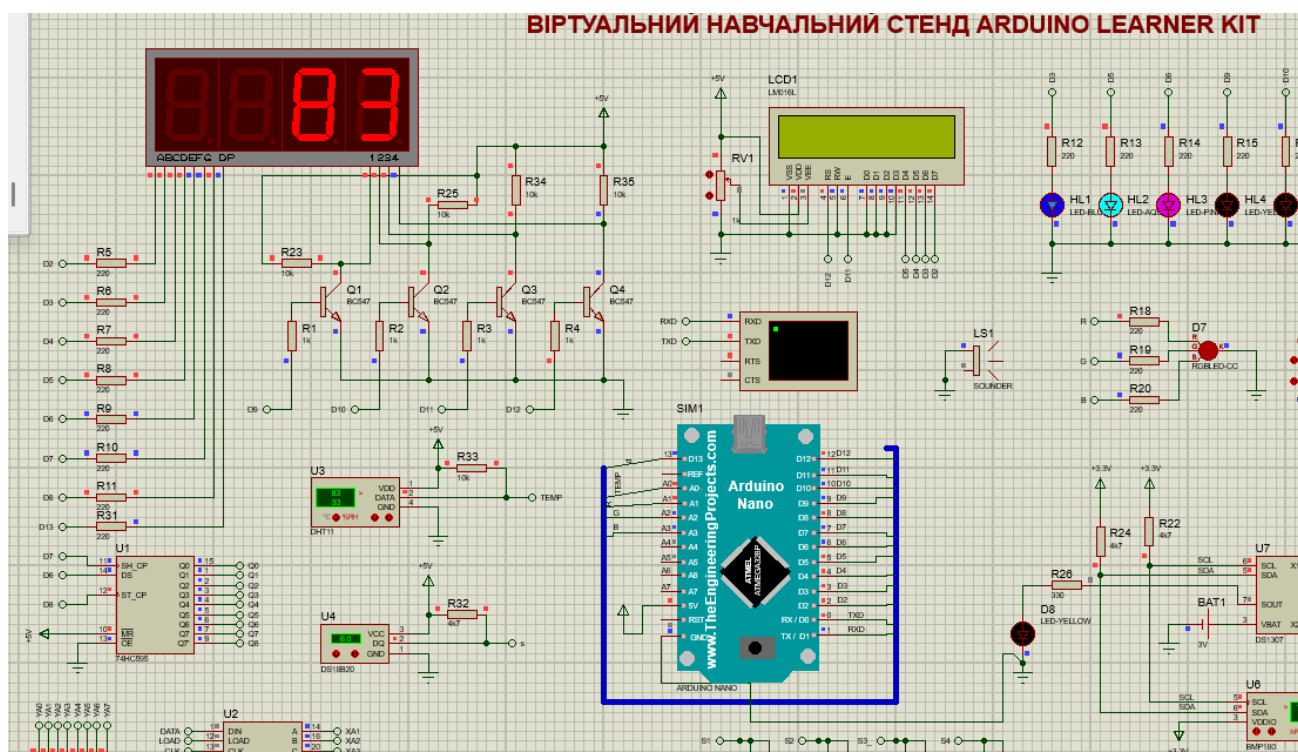


Рис. 3.Результат програми

Код програми МК AVR:

```
#define F_CPU 16000000UL
```

```
#include <avr/io.h>
```

```
#include <util/delay.h>
```

```
#include <stdio.h> //input/output library
```

```
#include "lcd_lib.h" // LCD library
```

```
#include <avr/interrupt.h>
```

```
// ----- LCD -----
```

```
#define SegA PD0
```

```
#define SegB PD1
```

```
#define SegC PD2
```

```
#define SegD PD3
```

```
#define SegE PD4
```

```
#define SegF PD5
```

```
#define SegG PD6
```

```
#define DHT_PORT PORTA
```

```
#define DHT_DDR DDRA
```

```
#define DHT_PINR PINA
```

```
#define DHT_PIN PA3
```

```
#define Dig1 PB1
```

```
#define Dig2 PB2
```

```
#define Dig3 PB3
```

```
#define Dig4 PB4
```

```
#define R_PIN PC1
```

```
#define G_PIN PC2
```

```
#define B_PIN PC3
```

```
int temp, hum;
```

```
uint8_t dht_read(uint8_t *temp, uint8_t *hum);
```

```
volatile uint16_t count = 1234;
```

```
volatile uint8_t current_digit = 1;
```

```
void RGB(int hum)
```

```
{
```

```
if(hum > 60) {
```

```
    PORTC |= (1<<R_PIN);
```

```
    PORTC &= ~(1<<G_PIN);
```

```
    PORTC &= ~(1<<B_PIN);
```



```

}
else if(hum >= 40) {

    PORTC &= ~(1<<R_PIN);
    PORTC |= (1<<G_PIN);
    PORTC &= ~(1<<B_PIN);

}
else if(hum < 40) {

    PORTC &= ~(1<<R_PIN);
    PORTC &= ~(1<<G_PIN);
    PORTC |= (1<<B_PIN);

}
else {

    PORTC &= ~(1<<R_PIN);
    PORTC &= ~(1<<G_PIN);
    PORTC &= ~(1<<B_PIN);

}
}

void disp_off() {
PORTB &= ~((1<<Dig1) | (1<<Dig2) | (1<<Dig3) | (1<<Dig4));
}

void disp(uint8_t number) {
PORTD &=
~((1<<SegA)|(1<<SegB)|(1<<SegC)|(1<<SegD)|(1<<SegE)|(1<<SegF)|(1<<SegG));
switch(number) {

    case 0: PORTD |=
(1<<SegA)|(1<<SegB)|(1<<SegC)|(1<<SegD)|(1<<SegE)|(1<<SegF); break;

    case 1: PORTD |= (1<<SegB)|(1<<SegC); break;

```

```

                                case 2: PORTD |=
(1<<SegA)|(1<<SegB)|(1<<SegD)|(1<<SegE)|(1<<SegG); break;

                                case 3: PORTD |=
(1<<SegA)|(1<<SegB)|(1<<SegC)|(1<<SegD)|(1<<SegG); break;

                                case 4: PORTD |=
(1<<SegB)|(1<<SegC)|(1<<SegF)|(1<<SegG); break;

                                case 5: PORTD |=
(1<<SegA)|(1<<SegC)|(1<<SegD)|(1<<SegF)|(1<<SegG); break;

                                case 6: PORTD |=
(1<<SegA)|(1<<SegC)|(1<<SegD)|(1<<SegE)|(1<<SegF)|(1<<SegG); break;

                                case 7: PORTD |=
(1<<SegA)|(1<<SegB)|(1<<SegC); break;

                                case 8: PORTD |=
(1<<SegA)|(1<<SegB)|(1<<SegC)|(1<<SegD)|(1<<SegE)|(1<<SegF)|(1<<SegG);
break;

                                case 9: PORTD |=
(1<<SegA)|(1<<SegB)|(1<<SegC)|(1<<SegD)|(1<<SegF)|(1<<SegG); break;
}
}

```

```

ISR(TIMER1_OVF_vect) {
    disp_off();
    switch(current_digit) {

                                case 1: disp(count / 1000); PORTB |= (1<<Dig1);
break;

                                case 2: disp((count/100) % 10); PORTB |=
(1<<Dig2); break;

                                case 3: disp((count/10) % 10); PORTB |=
(1<<Dig3); break;

                                case 4: disp(count % 10); PORTB |= (1<<Dig4);
break;

    }
}

```

```
current_digit = (current_digit % 4) + 1;
}
```

```
int main(void) {
    DDRD |=
    (1<<SegA)|(1<<SegB)|(1<<SegC)|(1<<SegD)|(1<<SegE)|(1<<SegF)|(1<<SegG);
    DDRB |= (1<<Dig1)|(1<<Dig2)|(1<<Dig3)|(1<<Dig4);
    DDRC |= (1<<R_PIN)|(1<<G_PIN)|(1<<B_PIN);
    DDRA &= ~(1 << PA1);
    PORTA |= (1 << PA1);
```

```
TCCR1A = 0;
TCCR1B = (1<<CS10);
TIMSK |= (1<<TOIE1);
sei();
```

```
int time;
while (1) {
```

```
    time++;
```

```
    dht_read(&temp, &hum);
```

```
    RGB(hum);
```

```
    if(time<50)
```

```
    {
```

```
        count =temp;
```

```
    }
```

```
    else if(time<100)
```

```

        {
            count =hum;
        }
        else
        {
            time=0;
        }
    }
}

```

```

uint8_t dht_read(uint8_t *temp, uint8_t *hum)

```

```

{

```

```

uint8_t data[5] = {0};

```

```

// стартовий сигнал

```

```

DHT_DDR |= (1<<DHT_PIN);    // PA4 → вихід

```

```

DHT_PORT &= ~(1<<DHT_PIN);  // низький рівень

```

```

_delay_ms(20);

```

```

DHT_PORT |= (1<<DHT_PIN);    // високий рівень

```

```

_delay_us(40);

```

```

DHT_DDR &= ~(1<<DHT_PIN);    // PA4 → вхід

```

```

// чекати відповіді DHT11

```

```

uint16_t timeout = 0;

```

```

while((DHT_PINR & (1<<DHT_PIN))) { if(timeout++ > 10000) return 1; }

```

```

while(!(DHT_PINR & (1<<DHT_PIN)));

```

```

while( (DHT_PINR & (1<<DHT_PIN)));

```

```

// прийом 40 біт
for(uint8_t i=0; i<40; i++)
{
    while(!(DHT_PINR & (1<<DHT_PIN)));
    _delay_us(30);

    if(DHT_PINR & (1<<DHT_PIN))
        data[i/8] |= (1 << (7 - (i%8)));

    while(DHT_PINR & (1<<DHT_PIN));
}

if((data[0] + data[1] + data[2] + data[3]) != data[4]) return 2;

*hum = data[0];
*temp = data[2];

return 0;
}

```

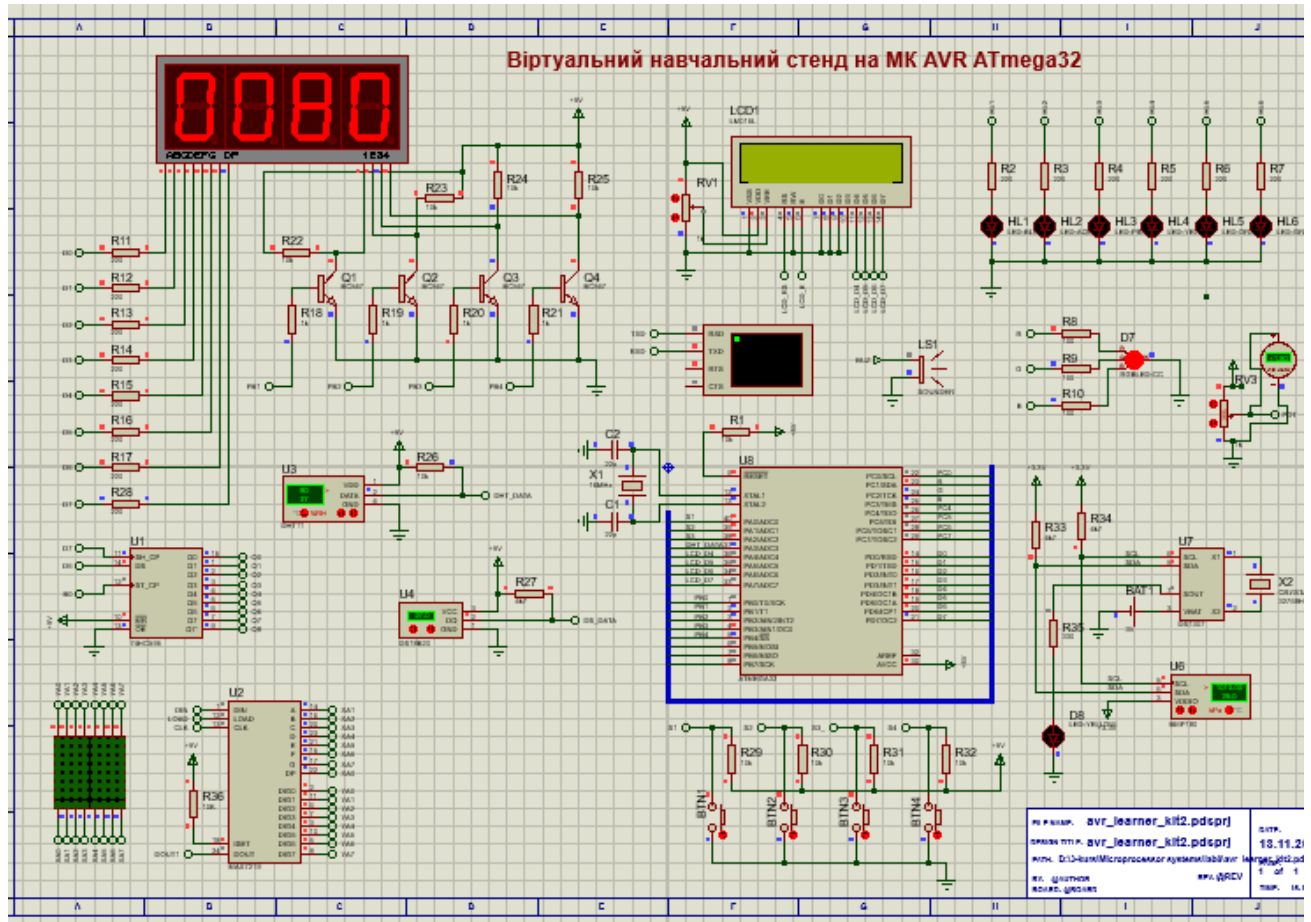


Рис. 4.Результат програми

3. Реалізувати програму, яка виводить на семисегментний індикатор з використанням регістру зсуву 74HC595 значення температури та вологості з датчика DHT11 та керує RGB світлодіодом. Якщо вологість $h < 40\%$, то світиться синій світлодіод; якщо $60\% \geq h \geq 40\%$, то світиться зелений світлодіод; якщо $h > 60\%$, то світиться червоний світлодіод.

Код програми Arduino IDE:

```
#include <LiquidCrystal.h>

#include "DHT.h"

#define R_pin A1

#define G_pin A2

#define B_pin A3

#define DHTPIN A0

#define clockPin 7 // SH_CP
```

```

#define dataPin 6 // SER
#define latchPin 8 // ST_CP

int current_digit;
int temp,hum,cur ;
DHT dht(DHTPIN, DHT11);
void disp(byte number);
void disp_off();

void RGB(int temp)
{
    if(hum>60)
    {
        digitalWrite(R_pin,HIGH);
        digitalWrite(G_pin,LOW);
        digitalWrite(B_pin,LOW);
    }
    else if(hum>=40)
    {
        digitalWrite(R_pin,LOW);
        digitalWrite(G_pin,HIGH);
        digitalWrite(B_pin,LOW);
    }
    else if(hum<40)
    {
        digitalWrite(R_pin,LOW);
        digitalWrite(G_pin,LOW);
        digitalWrite(B_pin,HIGH);
    }
}

```

```
    }  
    else  
    {  
        digitalWrite(R_pin,LOW);  
        digitalWrite(G_pin,LOW);  
        digitalWrite(B_pin,LOW);  
    }  
}
```

```
void setup() {  
    pinMode(5, OUTPUT);  
    pinMode(4, OUTPUT);  
    pinMode(3, OUTPUT);  
    pinMode(2, OUTPUT);  
    pinMode(clockPin, OUTPUT);  
    pinMode(dataPin, OUTPUT);  
    pinMode(latchPin, OUTPUT);  
    analogReference(INTERNAL); // встановлюємо опорну напругу 1.1 В  
    pinMode(R_pin, OUTPUT);  
    pinMode(G_pin, OUTPUT);  
    pinMode(B_pin, OUTPUT);  
    Serial.begin(9600);  
    disp_off();  
    dht.begin();  
  
    // Timer1 module overflow interrupt configuration  
    TCCR1A = 0;  
    TCCR1B = 1; // prescaler = 1
```



```

    TCNT1 = 0;

    TIMSK1 = 1; // enable Timer1 overflow interrupt
}

ISR(TIMER1_OVF_vect) {
    disp_off();

    switch (current_digit) {
        case 1:
            disp(cur / 1000);
            digitalWrite(5, HIGH);
            break;
        case 2:
            disp((cur / 100) % 10);
            digitalWrite(4, HIGH);
            break;
        case 3:
            disp((cur / 10) % 10);
            digitalWrite(3, HIGH);
            break;
        case 4:
            disp(cur % 10);
            digitalWrite(2, HIGH);
            break;
    }

    current_digit = (current_digit % 4) + 1;
}

```

```

int time;

void loop() {
    time++;

    temp =dht.readTemperature(); // правильне перетворення у °C
    hum =dht.readHumidity();

    RGB(temp);
    Serial.println(temp);
    RGB(temp);
    if(time<50)
    {
        cur=temp;
    }
    else if(time<100)
    {
        cur=hum;
    }
    else
    {
        time=0;
    }
}

```

```

void disp(byte number) {
    byte pattern;
    switch (number) {
        case 0: pattern = 0xFC; break;
        case 1: pattern = 0x60; break;
    }
}

```

```
case 2: pattern = 0xDA; break;
case 3: pattern = 0xF2; break;
case 4: pattern = 0x66; break;
case 5: pattern = 0xB6; break;
case 6: pattern = 0xBE; break;
case 7: pattern = 0xE0; break;
case 8: pattern = 0xFE; break;
case 9: pattern = 0xF6; break;
default: pattern = 0x00; break;
}
```

```
digitalWrite(latchPin, LOW);          // зупиняємо оновлення виходів
shiftOut(dataPin, clockPin, MSBFIRST, pattern); // передаємо дані
digitalWrite(latchPin, HIGH);         // оновлюємо виходи
}
```

```
void disp_off() {
    digitalWrite(5, LOW);
    digitalWrite(4, LOW);
    digitalWrite(3, LOW);
    digitalWrite(2, LOW);
}
```

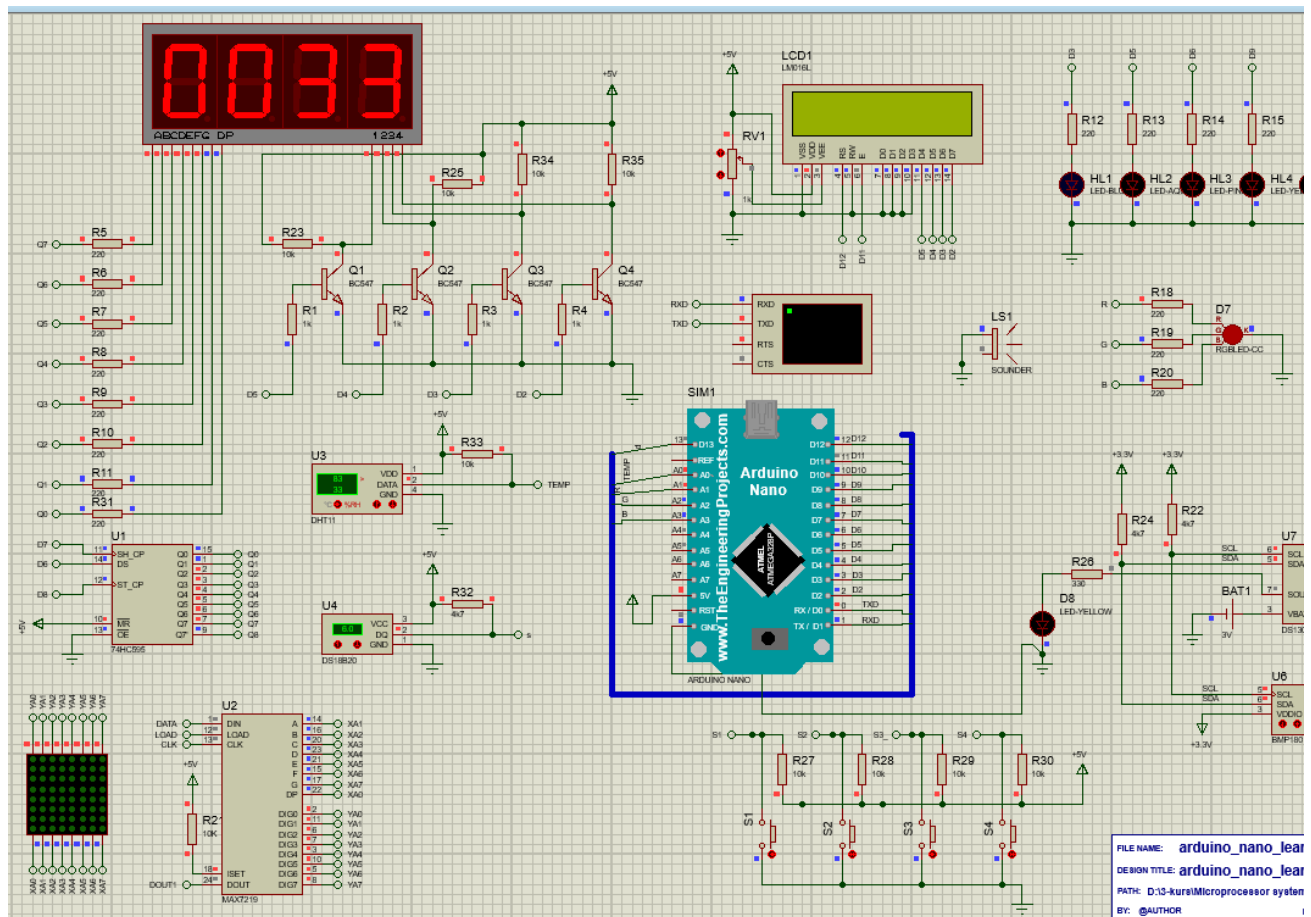


Рис. 5.Результат програми

Код програми МК AVR:

```
#define F_CPU 16000000UL
```

```
#include <avr/io.h>
```

```
#include <util/delay.h>
```

```
#include <stdio.h> //input/output library
```

```
#include "lcd_lib.h" // LCD library
```

```
#include <avr/interrupt.h>
```

```
unsigned char dataPin = 6;
```

```
unsigned char clockPin = 7;
```

```
unsigned char latchPin = 0;
```

```
#define DHT_PORT PORTA
```

```
#define DHT_DDR DDRA
```

```
#define DHT_PINR PINA
```

```
#define DHT_PIN PA3
```

```
#define Dig1 PB1
```

```
#define Dig2 PB2
```

```
#define Dig3 PB3
```

```
#define Dig4 PB4
```

```
#define R_PIN PC1
```

```
#define G_PIN PC2
```

```
#define B_PIN PC3
```

```
int temp, hum;
```

```
unsigned char current_digit = 1;
```

```
unsigned char digitPins[4] = {5,4,3,2};
```

```
uint8_t dht_read(uint8_t *temp, uint8_t *hum);
```

```
volatile uint16_t count = 0;
```

```
uint8_t digits[10] = {0xFC,0x60,0xDA,0xF2,0x66,0xB6,0xBE,0xE0,0xFE,0xF6};
```

```
void shiftOutByte(uint8_t data) {
```

```
for(int i=7;i>=0;i--) {
```

```
    if(data & (1<<i)) PORTD |= (1<<dataPin);
```

```
    else          PORTD &= ~(1<<dataPin);
```

```
    PORTD |= (1<<clockPin);
```

```
    PORTD &= ~(1<<clockPin);
```

```
}
```

```
}
```

```

void disp_digit(unsigned char number) {

for(int i=0;i<4;i++) PORTD &= ~(1<<digitPins[i]);


PORTB &= ~(1<<latchPin);
shiftOutByte(digits[number]);
PORTB |= (1<<latchPin);


PORTD |= (1<<digitPins[current_digit-1]);


}


ISR(TIMER1_OVF_vect) {
unsigned char d;
switch(current_digit) {

                                case 1: d = (count / 1000) % 10; break;
                                case 2: d = (count / 100) % 10; break;
                                case 3: d = (count / 10) % 10; break;
                                case 4: d = count % 10; break;

}
disp_digit(d);
current_digit = (current_digit % 4) + 1;
}

```

```

void RGB(int hum)
{
    if(hum > 60) {

        PORTC |= (1<<R_PIN);
        PORTC &= ~(1<<G_PIN);
        PORTC &= ~(1<<B_PIN);

    }
    else if(hum >= 40) {

        PORTC &= ~(1<<R_PIN);
        PORTC |= (1<<G_PIN);
        PORTC &= ~(1<<B_PIN);

    }
    else if(hum < 40) {

        PORTC &= ~(1<<R_PIN);
        PORTC &= ~(1<<G_PIN);
        PORTC |= (1<<B_PIN);

    }
    else {

        PORTC &= ~(1<<R_PIN);
        PORTC &= ~(1<<G_PIN);
        PORTC &= ~(1<<B_PIN);

    }
}

int main(void) {

```

```
        DDRD |=  
(1<<dataPin)|(1<<clockPin)|(1<<digitPins[0])|(1<<digitPins[1])|(1<<digitPins[2])|(1<<digitPins[3]);
```

```
        DDRB |= (1<<latchPin);
```

```
        DDRC |= (1<<R_PIN)|(1<<G_PIN)|(1<<B_PIN);
```

```
TCCR1A = 0;
```

```
TCCR1B = (1<<CS10);
```

```
TIMSK |= (1<<TOIE1);
```

```
sei();
```

```
int time;
```

```
while (1) {
```

```
    time++;
```

```
    dht_read(&temp, &hum);
```

```
    RGB(hum);
```

```
    if(time<50)
```

```
    {
```

```
        count =temp;
```

```
    }
```

```
    else if(time<100)
```

```
    {
```

```
        count =hum;
```

```
    }
```

```
    else
```

```
    {
```

```
        time=0;
```

```
    }
```



```
}  
}
```

```
uint8_t dht_read(uint8_t *temp, uint8_t *hum)
```

```
{
```

```
cli();
```

```
uint8_t data[5] = {0};
```

```
// стартовий сигнал
```

```
DHT_DDR |= (1<<DHT_PIN);    // PA4 → вихід
```

```
DHT_PORT &= ~(1<<DHT_PIN);  // низький рівень
```

```
_delay_ms(20);
```

```
DHT_PORT |= (1<<DHT_PIN);   // високий рівень
```

```
_delay_us(40);
```

```
DHT_DDR &= ~(1<<DHT_PIN);   // PA4 → вхід
```

```
// чекати відповіді DHT11
```

```
uint16_t timeout = 0;
```

```
while((DHT_PINR & (1<<DHT_PIN))) { if(timeout++ > 10000) return 1; }
```

```
while(!(DHT_PINR & (1<<DHT_PIN)));
```

```
while( (DHT_PINR & (1<<DHT_PIN)));
```

```
// прийом 40 біт
```

```
for(uint8_t i=0; i<40; i++)
```

```
{
```

```
while(!(DHT_PINR & (1<<DHT_PIN)));
```

```
_delay_us(30);
```

```
        if(DHT_PINR & (1<<DHT_PIN))
            data[i/8] |= (1 << (7 - (i%8)));

        while(DHT_PINR & (1<<DHT_PIN));
    }

    sei();

    if((data[0] + data[1] + data[2] + data[3]) != data[4]) return 2;

    *hum = data[0];
    *temp = data[2];

    return 0;
}
```

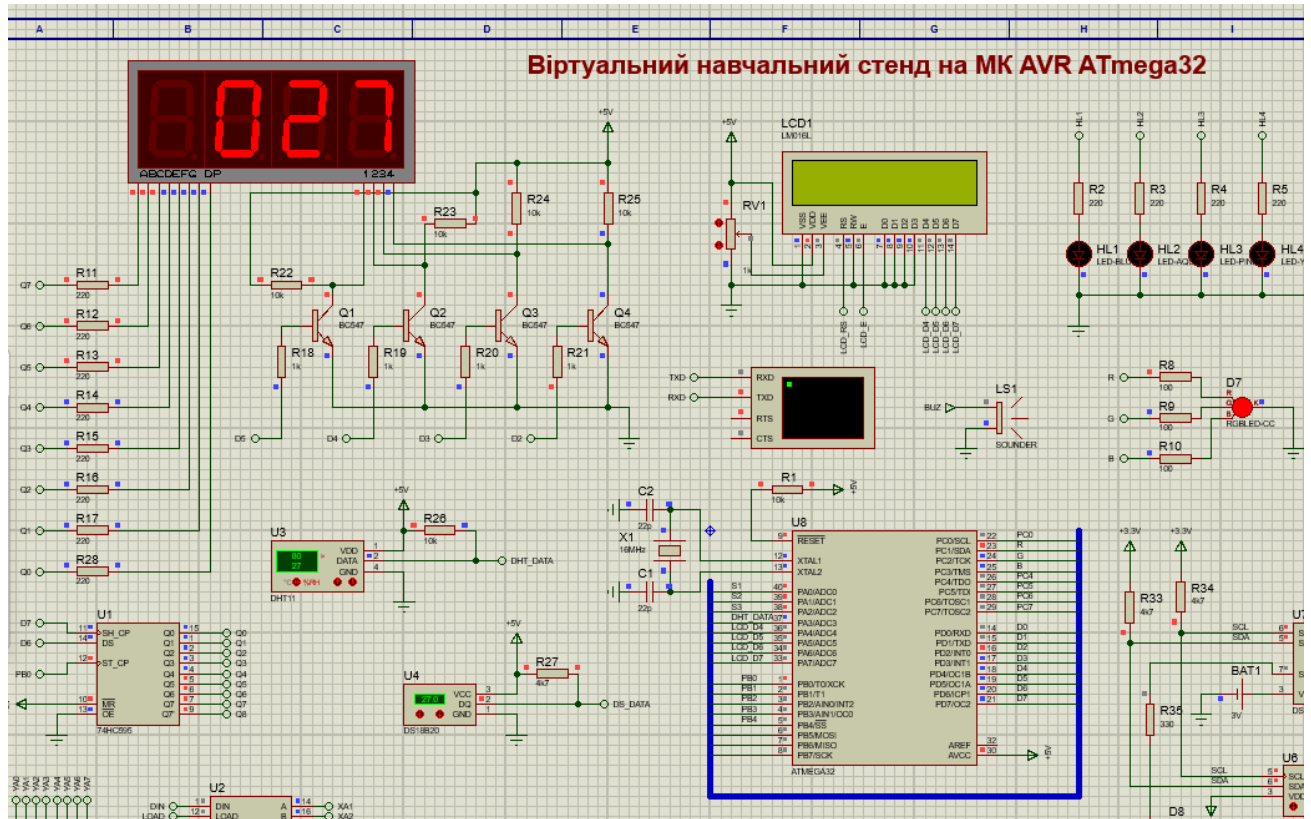


Рис. 6.Результат програми

Висновок: У ході виконання лабораторної роботи я ознайомився з принципом роботи та зчитуванням даних датчика температури та вологості DHT11; закріпив навички виведення інформації на семисегментний індикатор з використанням регістру зсуву 74HC595 та LCD-індикатор.