

**МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ  
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ “ЛЬВІВСЬКА ПОЛІТЕХНІКА”**

Кафедра САП



**ЗВІТ**

до виконання лабораторної роботи №6

На тему: "робота з символічним LCD HD44780"

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 LCD з контролером HD44780 у 4 бітному режимі підключення та дослідити можливості виведення на дисплей інформації; закріпити навички роботи з цифровими портами, тактовими кнопками, масивами.

## Теоретичні відомості

Можливість оснащення рідкокристалічних дисплеїв задньою підсвіткою дозволяє їх експлуатувати в умовах зі зниженою або нульовою освітленістю. Саме виконання РК - модулів з розширеним діапазоном робочих температур від -20°C до +70°C дозволяє використовувати їх в складних експлуатаційних умовах, наприклад в мобільних, польових або бортових пристроях.

Мікросхема драйвера HD44780 може потенційно управляти двома рядками по 40 символів в кожному, з матрицею символа  $5\times 7$  точок. В модулях з чотирма рядками по 40 символів використовуються дві однотипних мікросхеми драйвера. Драйвери – контролери підтримують також символи з матрицею  $5\times 10$  точок, але останні роки РК – модулі з такою матрицею практично не зустрічаються, тому можна вважати, що фактично є тільки символи  $5\times 7$  точок. На теперішній час є кілька різних більш менш стандартних форматів рідкокристалічних дисплеїв (символів  $\times$  рядок):  $8\times 2$ ,  $16\times 1$ ,  $16\times 2$ ,  $16\times 4$ ,  $20\times 1$ ,  $20\times 2$ ,  $20\times 4$ ,  $24\times 2$ ,  $40\times 2$ ,  $40\times 4$ . Також можна зустріти менш поширені формати:  $8\times 1$ ,  $12\times 2$ ,  $32\times 2$  та інші. Взагалі то, драйвер HD44780 не накладає принципових обмежень на комбінацію та кількість символів відображення. Модуль може мати будь-яку кількість символів від 1 до 80, хоча в деяких комбінаціях програмна адресація символів може виявитися не дуже зручною.

В рамках одного формату можуть виготовлятися РК – модулі декількох конструктивів, які відрізняються як габаритами рідкокристалічних дисплеїв (і, як результат, розмірами символів), так і розмірами плати і посадки. Вивчаючи продукцію різних фірм-виробників рідкокристалічних дисплеїв, можна переконатися, що одні формати і конструктиви є власними розробками і не мають аналогів в номенклатурі решти фірм, інші є фактично стандартами і виготовляються більшістю виробників. В якості прикладу можна назвати РК – модулі формату  $24\times 2$  PC2402-A від Powertip, ED24200 від EDT, DMC-24227 від Optrex, SC2402A від Bolymin, MDLS-24265 від Varitronix, PVC240202 від Picvue та інші. Всі ці модулі мають однакові конструктивні розміри і є взаємозамінними.

Для підключення РК – модуля з системою управління використовується паралельна синхронна шина, яка налічує 8 або 4 (вибирається програмно) ліній даних DB0...DB7, лінію вибору операції R/W, лінію вибору реєстра RS і лінію стропування/синхронізації E. Крім ліній управління шиною є дві лінії для подачі напруги живлення 5V – GND і V<sub>cc</sub>, і лінія для подачі напруги живлення

драйвера РК – модуля –  $V_o$ . Вказані вище назви ліній шини є стандартними, але існує безліч різних варіантів розміщення виводів (контактів) у кожного конкретного конструктиву модуля рідкокристалічного дисплею. Наприклад в нашому варіанті, шістнадцяте контактне поле розміщено горизонтально в лівій частині РК - модуля наступним чином: 1 –  $V_{ss}$  (GND), 2 – VDD,  $V_{cc}$  (+5V), 3 –  $V_o$  (підсвітка), 4 – RS (вибір регістра), 5 – RW (GND), 6 – E (синхронізація), 7 – D0 (GND), 8 – D1 (GND), 9 – D2 (GND), 10 – D3 (GND), 11 – D4. 12 – D5, 13 – D6, 14 – D7, 15 – K (катод підсвітки, GND), 16 – A (анод підсвітки, +5V) (Рис. ). Модуль з дисплеєм на шістнадцять символів по два рядки з зеленою підсвіткою.

У порівняні зі звичайними 7-сегментними модулями LCD – модулі на контролері HD44780 мають на порядок більші можливості. Кількість рядків на екрані у різних моделях – 1,2 або 4, число символів: 8, 10, 16, 20, 24, 30, 32 або 40. Кожне знакомісце на дисплей представляє собою матрицю розміром  $5 \times 8$  пікселів. Індикатор має світлодіодну або люмінесцентну підсвітку практично будь-якого кольору свічення. LCD - модулі працюють в температурному діапазоні від  $-20^{\circ}\text{C}$  до  $+70^{\circ}\text{C}$ , що дозволяє їх застосовувати в переносній, польовій та бортовій апаратурі з жорсткими експлуатаційними умовами, На Рис. 3.17 зображене зовнішній вигляд LCD – модуля і схему його підключення до МК. Hitachi HD44780 – це алфавітно – цифровий матричний контролер рідкокристалічних дисплейів, розроблений Hitachi в 1980-х роках. Напруга живлення контролера HD44780 5 В (рідше 3 В). Струм споживання самого контролера дуже малий 100...200 мА, а підсвітка в залежності від виробника складає 80...120 мА. Для роботи деяких LCD – модулів може бути потрібне додаткове джерело напруги від'ємної полярності. Технологія виготовлення LCD - модулів постійно вдосконалюється, що в цілому позитивно позначається на їх розмірах і електричних характеристиках.

`lcd.begin (cols, rows)` ініціалізує інтерфейс для взаємодії з LCD- індикатором та задає розміри (ширину і висоту) області виведення екрану, де `lcd`: змінна типу LiquidCrystal, `cols`: кількість стовпців екрану, `rows`: кількість рядків екрану. При роботі з LCD-дисплеєм, функція `begin()` повинна викликатися першою і передувати іншим командам з бібліотеки LiquidCrystal.

`lcd.clear ()` очищає LCD-екран і переміщує курсор в лівий верхній кут.

`lcd.home ()` переміщує курсор в лівий верхній кут екрану (наступний текст буде виводиться з початку екрану).

`lcd.setCursor (col, row)` встановлює позицію, в якій буде виводитися наступний текст, де `col`: координата X позиції курсора (0 означає перший стовпець); `row`: координата Y позиції курсора (0 означає перший рядок).

`lcd.write (data)` виводить символ на LCD-індикатор, де `data`: символ, який необхідно вивести на екран.

`lcd.print(data) / lcd.print(data, BASE)` виводить текст на LCD- індикатор, де *data*: дані, які необхідно вивести (тип char, byte, int, long або string); *BASE* (не обов'язковий параметр): основа системи числення, в якій необхідно виводити числа: BIN. двійкова, DEC. десяткова, OCT. вісімкова, HEX.шістнадцятькова.

`lcd.cursor() / lcd.noCursor()` показує/ не показує на LCD-екрані курсор: символ підкреслення в тій позиції, куди буде виведий наступний символ.

`lcd.blink() / lcd.noBlink()` включає / відключає на LCD-індикаторі курсор, що мигає.

`lcd.noDisplay() / lcd.display()` вимикає / вмикає LCD-екран. Текст, якщо не відображається на екрані, зберігається в пам'яті.

`lcd.scrollDisplayLeft() / lcd.scrollDisplayRight()` здійснює прокручування вмісту дисплея (весь текст і курсор) на один символ ліворуч / праворуч.

`lcd.autoscroll()` включає автоматичну прокрутку тексту на LCD. Це означає, що при виведенні кожного нового символу, всі попередні символи будуть зсуватись на одну позицію. Якщо встановлений режим перегляду тексту зліва-направо (за замовчуванням), то прокрутка буде здійснюватися ліворуч; якщо встановлений режим зправа-наліво, то прокрутка буде здійснюватися праворуч. Таким чином, кожен новий символ буде виводиться в одній і тій же позиції LCD.

`lcd.noAutoscroll()` функція відключає автоматичну прокрутку тексту в LCD.

`lcd.leftToRight() / lcd.rightToLeft()` встановлює режим перегляду тексту на LCD зліва-направо (режим за замовчуванням) / зправа-наліво.

`lcd.createChar(num, data)` створює символ користувача для LCD- індикатора. Дисплей підтримує до 8 символів користувача (пронумерованих від 0 до 7) розміром 5x8 пікселів. Зовнішній вигляд кожного символу користувача задається масивом з восьми байт, кожен з яких характеризує відповідний рядок. П'ять молодших біт кожного байта визначають стан пікселів у відповідному рядку. Для того, щоб вивести певний символ користувача, використовується функцію `write()` з його номером, де *num*: номер призначеного символу користувача, який необхідно створити (від 0 до 7); *data*: дані у пікселях символу користувача

## Завдання

1. Реалізувати програму, яка виводить прізвище, ім'я та по батькові у вигляді рядка на LCD-індикаторі, що «біжить».

Код програми Arduino IDE:

```
#include <LiquidCrystal.h>
```

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

```
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
```

```
void setup() {
```

```
    lcd.begin(16, 2);
```

```
    lcd.print("Nazar Havryliuk");
```

```
    delay(1000);
```

```
}
```

```
void loop() {
```

```
    // Зсув на 13 позицій (довжина тексту) вліво
```

```
    for (int position = 0; position < 13; position++) {
```

```
        lcd.scrollDisplayLeft();
```

```
        delay(150);
```

```
}
```

```
// Зсув на 29 позицій вправо (довжина тексту + ширина екрана)
```

```
    for (int position = 0; position < 29; position++) {
```

```
        lcd.scrollDisplayRight();
```

```
        delay(150);
```

```
}
```

```
// Повернення тексту до початкового положення (16 позицій вліво)
```

```
    for (int position = 0; position < 16; position++) {
```

```
        lcd.scrollDisplayLeft();
```

```

delay(150);

}

```

```

delay(1000);

}

```

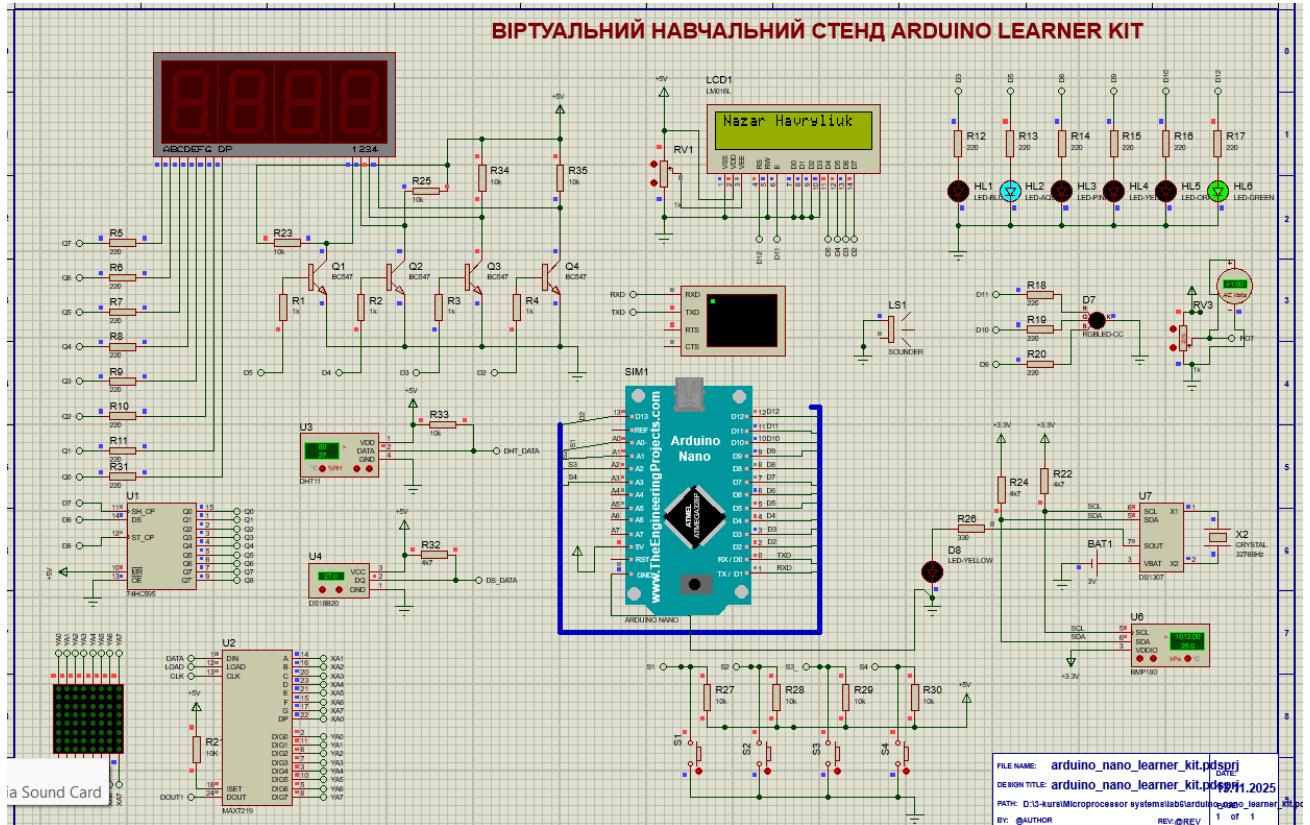


Рис. . Результат программи

Код програми MK AVR:

```
#include <avr/io.h> /* Defines pins, ports, etc */
```

```
#define F_CPU 16000000
```

```
#include <util/delay.h> /* Functions to waste time */
```

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

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

```
#define rgbLED_DDR DDRB
#define rgbLED_PORT PORTB

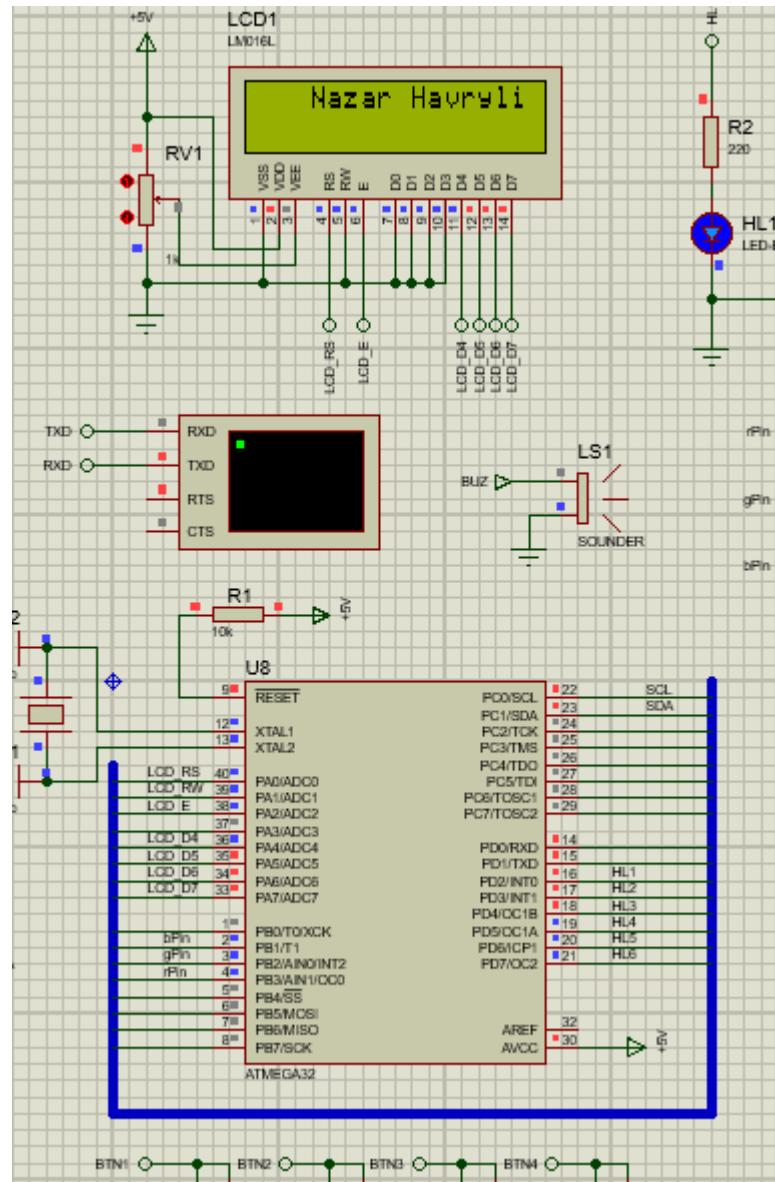
#define LEDS_DDR DDRD
#define LEDS_PORT PORTD

int main(void) {
    int i = 3;
    DDRD |= (1<<PD0)|(1<<PD1)|(1<<PD2)|(1<<PD3)|(1<<PD4);
    PORTD |= (1<<PD0);
    PORTD |= (1<<PD1);
    PORTD |= (1<<PD2);
    PORTD |= (1<<PD3);
    PORTD |= (1<<PD4);

/* Ініціалізація LCD */
LCDinit();
LCDcursorOFF();
LCDstring("Nazar Havryliuk");
LCDGotoXY(0,1);
LCDstring(" ");
_delay_ms(2000);
//LCDclr();

// ----- Event loop ----- //
while (1) {
    // Зсув на 13 позицій (довжина тексту) вліво
```

```
for (int position = 0; position < 13; position++) {  
    LCDshiftLeft(1);  
    _delay_ms(150);  
}  
  
// Зсув на 29 позицій вправо (довжина тексту +  
ширина екрана)  
for (int position = 0; position < 29; position++) {  
    LCDshiftRight(1);  
    _delay_ms(150);  
}  
  
// Повернення тексту до початкового  
положення (16 позицій вліво)  
for (int position = 0; position < 16; position++) {  
    LCDshiftLeft(1);  
    _delay_ms(150);  
}  
  
_delay_ms(1000);  
}  
return (0); /* This line is never reached */  
}
```



## Рис. . Результат програми

2. Внести зміни до попередньої програми, щоб виводилася ще поточна

## Код програми Arduino IDE:

```
#include <LiquidCrystal.h>
```

```
// Підключення LCD: RS, E, D4, D
```

oldSetup() {

```
lcd.setCursor(0,1);
lcd.print("14.11.2025");
delay(1000);

}

void loop() {
    // Зсув на 13 позицій (довжина тексту) вліво
    for (int position = 0; position < 13; position++) {
        lcd.scrollDisplayLeft();
        delay(150);
    }

    // Зсув на 29 позицій вправо (довжина тексту + ширина екрана)
    for (int position = 0; position < 29; position++) {
        lcd.scrollDisplayRight();
        delay(150);
    }

    // Повернення тексту до початкового положення (16 позицій вліво)
    for (int position = 0; position < 16; position++) {
        lcd.scrollDisplayLeft();
        delay(150);
    }

    delay(1000);
}
```

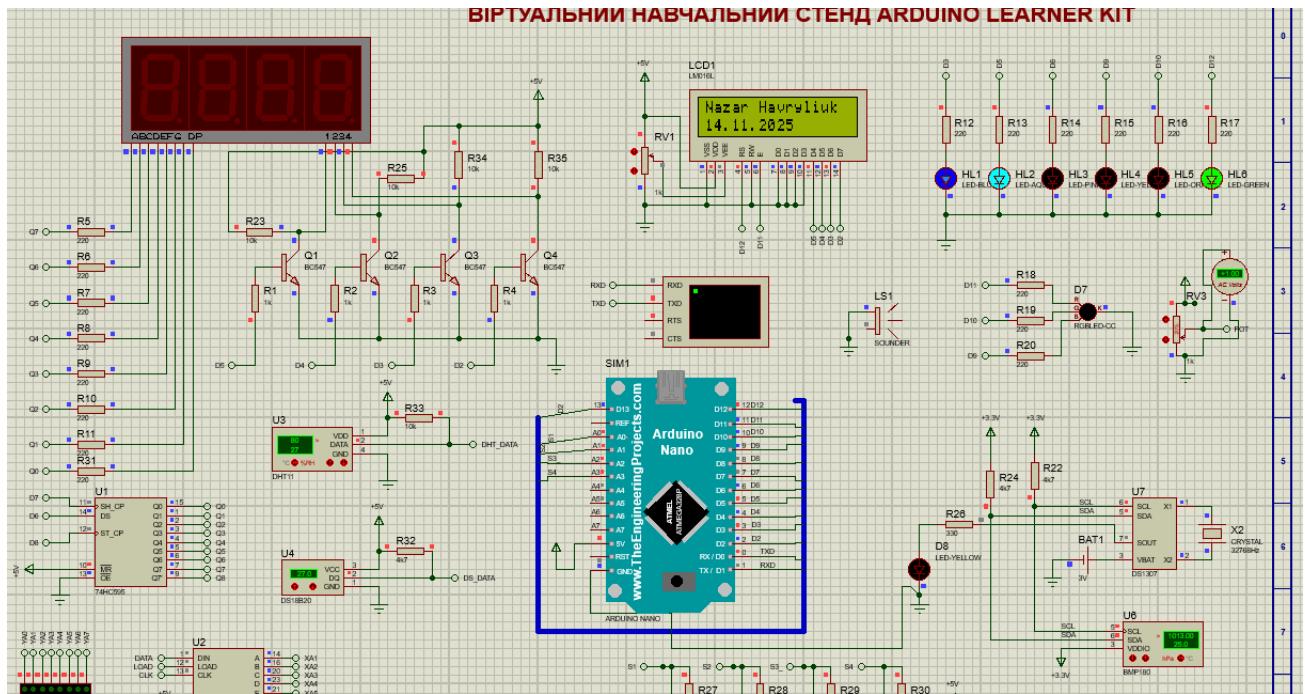


Рис. . Результат программи

Код програми MK AVR:

```
#include <avr/io.h> /* Defines pins, ports, etc */
```

```
#define F_CPU 16000000
```

```
#include <util/delay.h> /* Functions to waste time */
```

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

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

```
#define rgbLED_DDR DDRB
```

```
#define rgbLED_PORT PORTB
```

```
#define LEDS_DDR DDRD
```

```
#define LEDS_PORT PORTD
```

```
int main(void) {
    int i = 3;

    DDRD |= (1<<PD0)|(1<<PD1)|(1<<PD2)|(1<<PD3)|(1<<PD4);
    PORTD |= (1<<PD0);
    PORTD |= (1<<PD1);
    PORTD |= (1<<PD2);
    PORTD |= (1<<PD3);
    PORTD |= (1<<PD4);

    /* Ініціалізація LCD */
    LCDinit();
    LCDcursorOFF();
    LCDstring("Nazar Havryliuk");
    LCDGotoXY(0,1);
    LCDstring("14.11.2025");
    _delay_ms(2000);
    //LCDclr();

    // ----- Event loop ----- //
    while (1) {
        // Зсув на 13 позицій (довжина тексту) вліво
        for (int position = 0; position < 13; position++) {
            LCDshiftLeft(1);
            _delay_ms(150);
        }

        // Зсув на 29 позицій вправо (довжина тексту + ширина екрана)
        for (int position = 0; position < 29; position++) {
```

```
LCDshiftRight(1);
_delay_ms(150);
}

// Повернення тексту до початкового положення (16 позицій вліво)
for (int position = 0; position < 16; position++) {
    LCDshiftLeft(1);
    _delay_ms(150);
}

_delay_ms(1000);

}
return (0); /* This line is never reached */
}
```

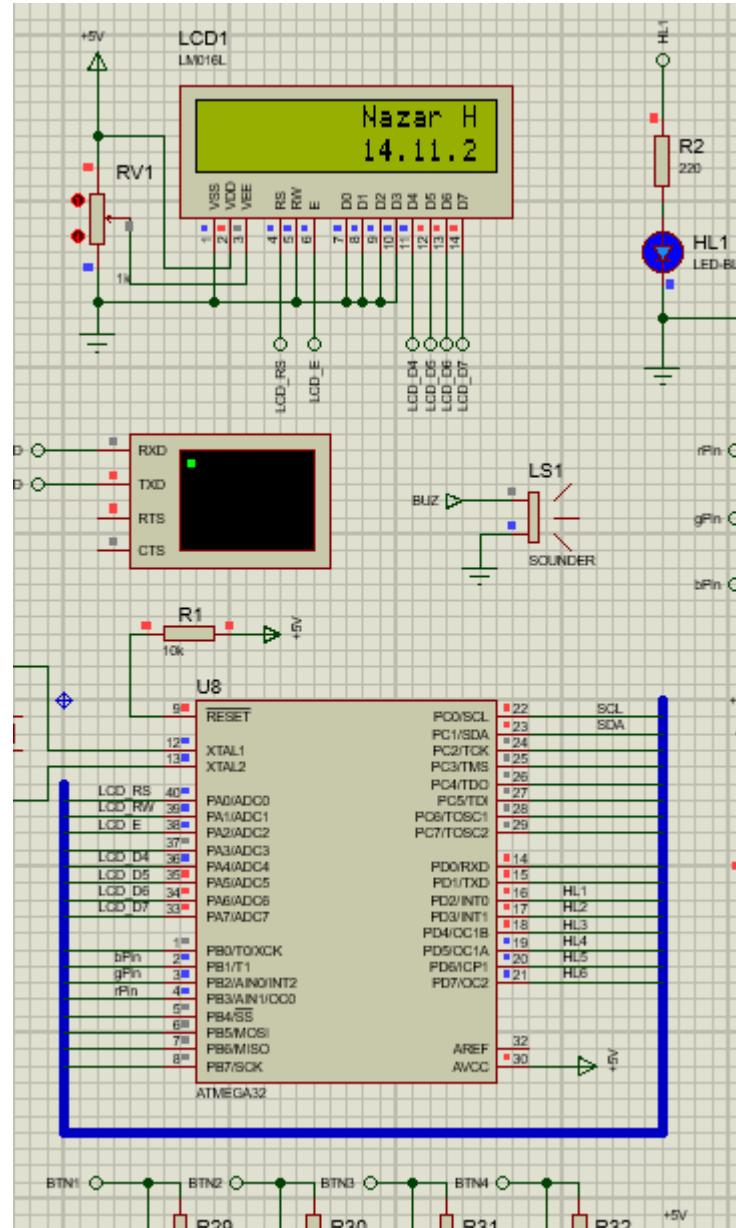


Рис. . Результат програми

3. Реалізувати програму виведення випадкового числа на LCD-індикаторі за натисненням кнопки.

Код програми Arduino IDE:

```
#include <LiquidCrystal.h>
```

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

```
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
```

```
int count=0000;
```

```
bool buttonsPres=false;
```

```
int currentTime,lastInteraptTime=0;
```

```
void setup() {  
    lcd.begin(16,2);  
    pinMode(A0,INPUT_PULLUP);  
    Serial.begin(9600);  
}  
  
void loop() {  
    Serial.println(count);  
    currentTime=millis();  
    if(digitalRead(A0) == LOW && buttonsPres == false && currentTime - lastInteraptTime>= 50)  
    {  
        lcd.clear();  
        count=random(0,99999999);  
        lcd.print(count);  
        buttonsPres = true;  
        lastInteraptTime= currentTime;  
    }  
  
    if(digitalRead(A0) == HIGH && buttonsPres == true )  
    {  
        buttonsPres= false;  
        lastInteraptTime= currentTime;  
    }  
}
```

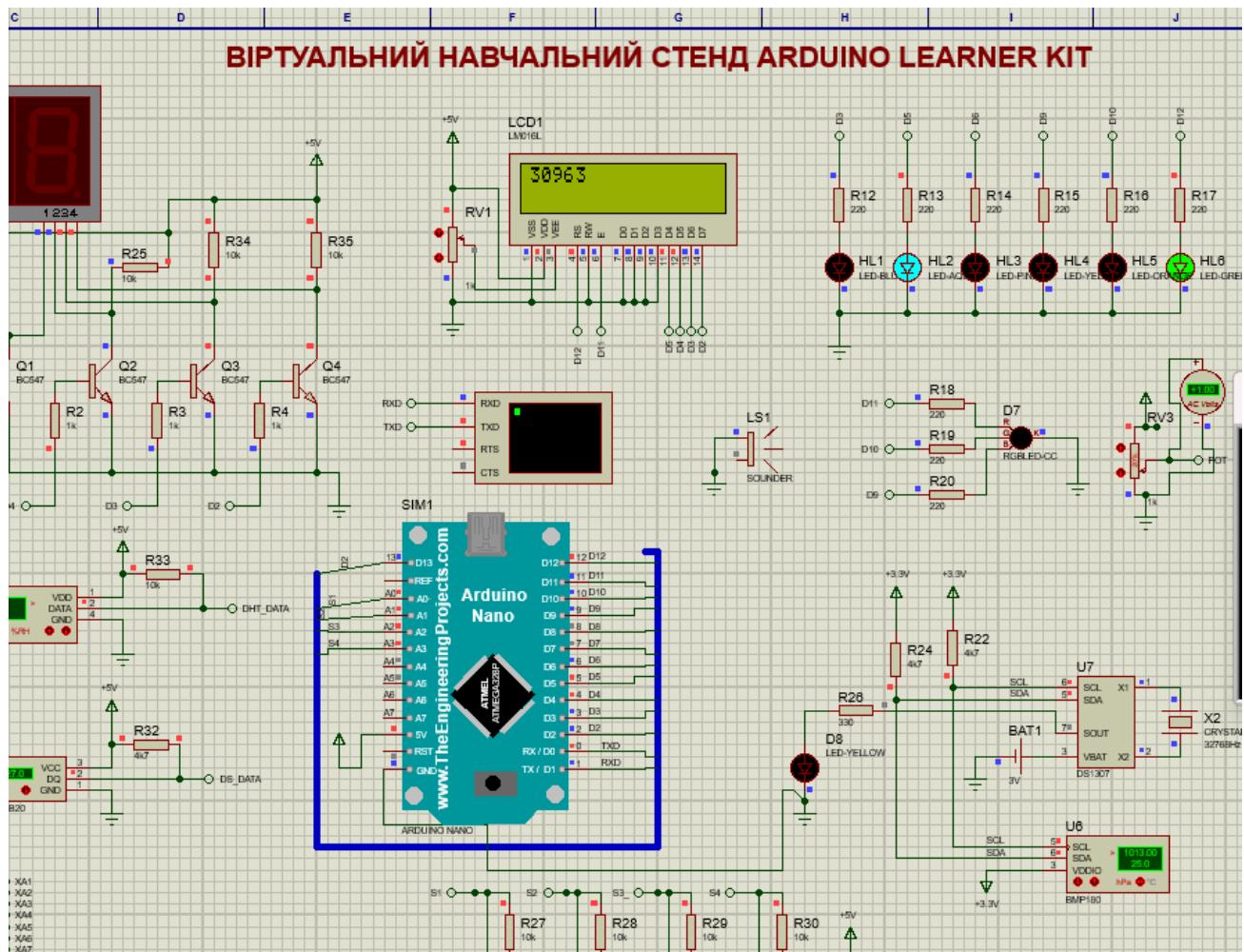


Рис. . Результат программи

Код програми MK AVR:

```
#include <avr/io.h> /* Defines pins, ports, etc */
```

```
#define F_CPU 16000000
```

```
#include <util/delay.h> /* Functions to waste time */
```

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

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

```
#define rgbLED_DDR DDRB
```

```
#define rgbLED_PORT PORTB
```

```
#define LEDS_DDR DDRD
#define LEDS_PORT PORTD

int main(void) {
    int count=1111;
    DDRD |= (1<<PD0)|(1<<PD1)|(1<<PD2)|(1<<PD3)|(1<<PD4);
    PORTD |= (1<<PD0);
    PORTD |= (1<<PD1);
    PORTD |= (1<<PD2);
    PORTD |= (1<<PD3);
    PORTD |= (1<<PD4);

    LCDinit();
    LCDcursorOFF();

    while (1) {
        count =rand() % 10000;
        char buffer[10];
        utoa(count, buffer, 10);
        LCDclr();
        LCDstring(buffer);
        _delay_ms(2000);
    }
    return (0); /* This line is never reached */
}
```

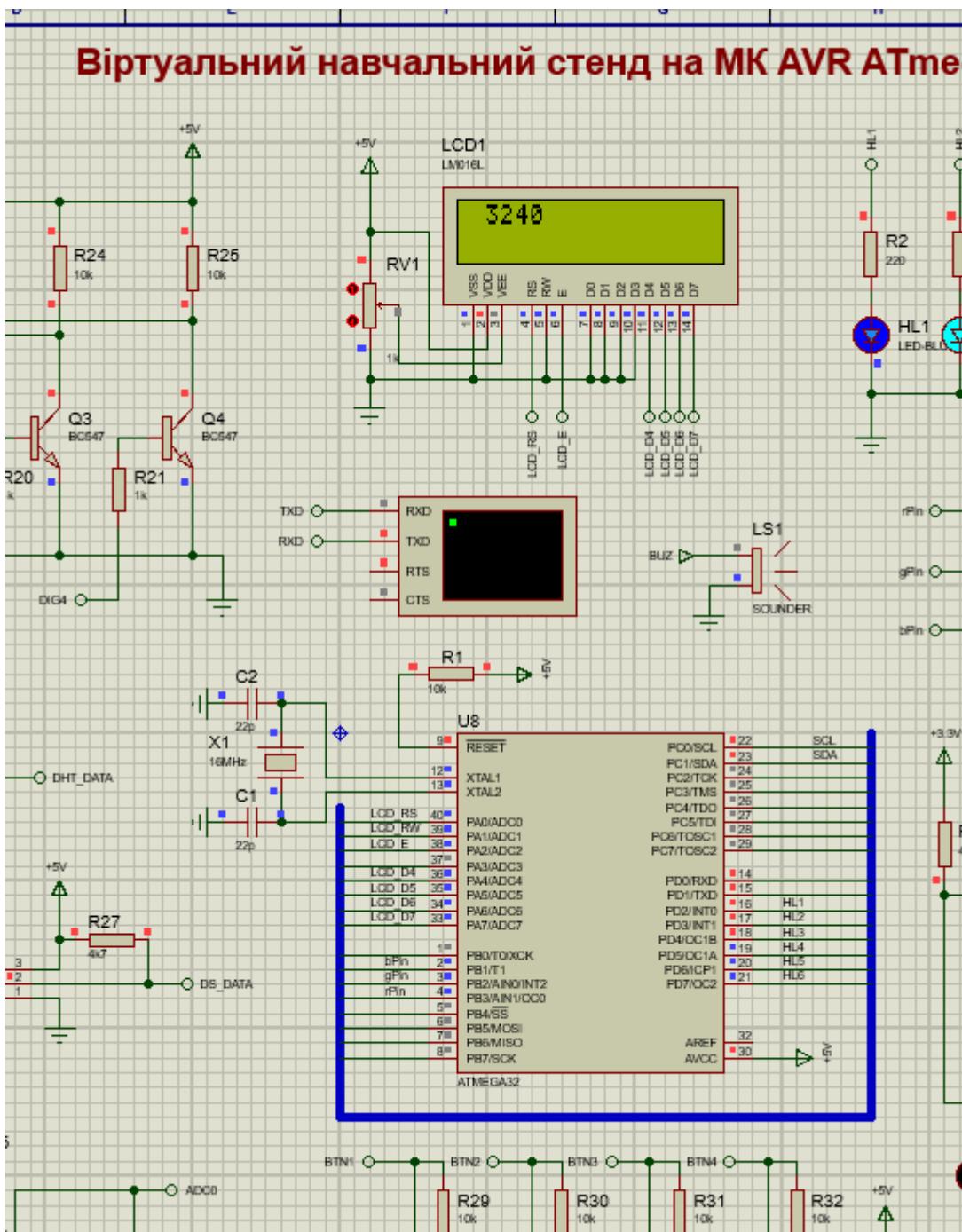


Рис. . Результат программи

4. Реалізувати програму, яка відображає на LCD-індикаторі напис “LOAD”. За натисненням кнопки S1 відображається напис “PLAY”, а S2 “STOP”.

Код програми Arduino IDE:

```
#include <LiquidCrystal.h>
```

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

```
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
```

```
int count=0000;
```

```
bool buttonsPres[]={false,false};

int currentTime;

int lastInteraptTime[]={0,0};

const byte swPins[]={A0,A1};

void setup() {

    lcd.begin(16,2);

    pinMode(A0,INPUT_PULLUP);

    Serial.begin(9600);

    lcd.print("LOAD");

}

void loop() {

    currentTime=millis();

    for(int i=0;i<2;i++)

    {

        if(digitalRead(swPins[i]) == LOW && buttonsPres[i] == false && currentTime - lastInteraptTime[i]>= 50)

        {

            lcd.clear();

            if(i==0)

            {

                lcd.print("PLAY");

            }

            if(i==1)
```

```
{  
    lcd.print("STOP");  
}  
  
buttonsPres[i] = true;  
lastInteraptTime[i]= currentTime;  
}  
  
  
if(digitalRead(swPins[i]) == HIGH && buttonsPres[i] == true )  
{  
    buttonsPres[i]= false;  
    lastInteraptTime[i]= currentTime;  
}  
}  
}
```

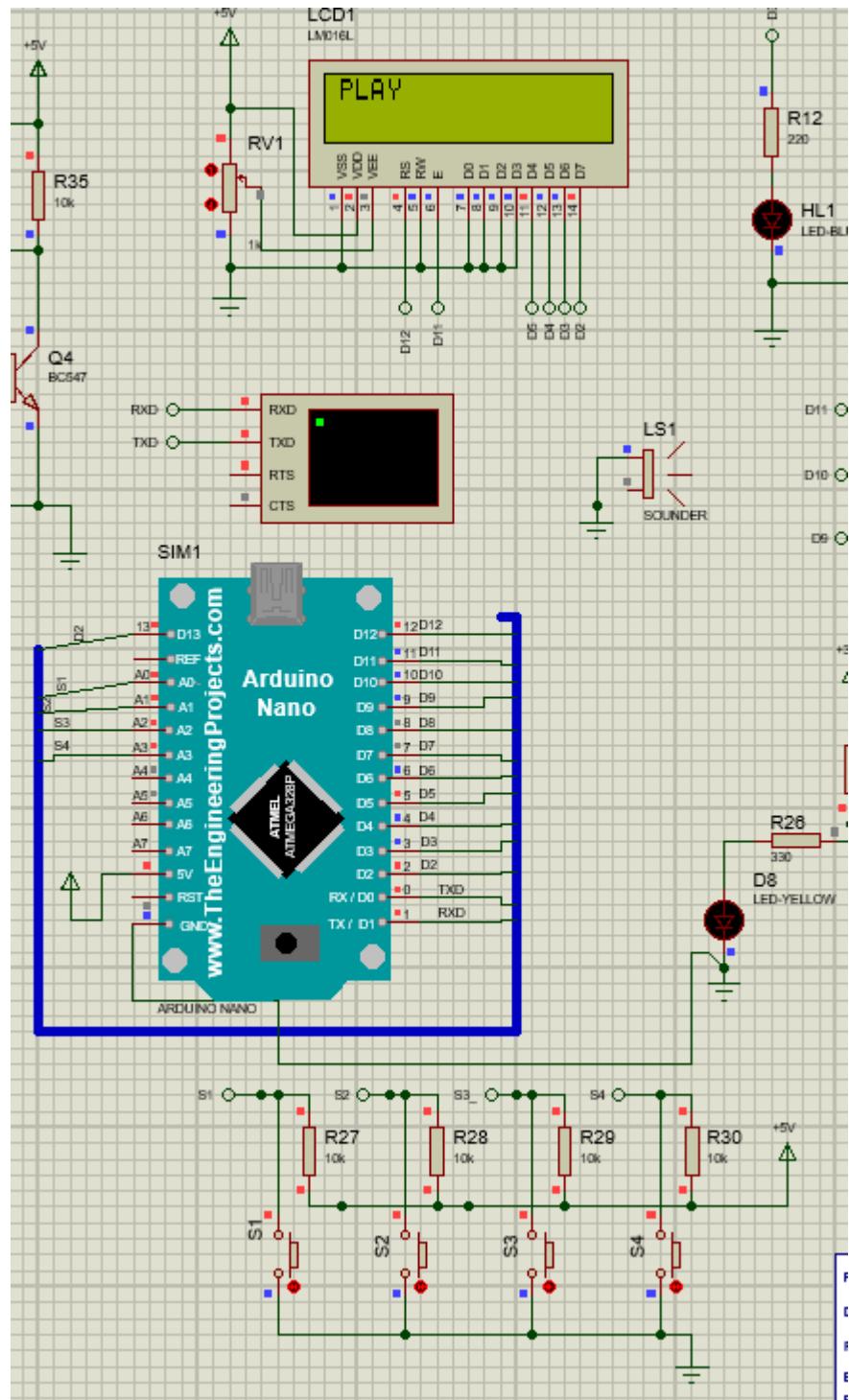


Рис. . Результат программи

Код програми MK AVR:

```
#include <avr/io.h> /* Defines pins, ports, etc */
```

```
#define F_CPU 16000000
```

```
#include <util/delay.h> /* Functions to waste time */
```

```

#include <stdio.h> //input/output library
#include <stdbool.h>
#include "lcd_lib.h" // LCD library
#include <avr/interrupt.h>
#define rgbLED_DDR DDRB
#define rgbLED_PORT PORTB

#define LEDS_DDR DDRD
#define LEDS_PORT PORTD

unsigned long currentTime;
const uint8_t swPins[] = {6,7,};
volatile unsigned long lastInteraptTime[] = {0,0};
volatile bool buttonsPres[] = {false,false};
volatile uint32_t millis_count = 0;

ISR(TIMER0_COMP_vect)
{
    millis_count++;
}

void timer0_init(void)
{
    TCCR0 = (1 << WGM01) | (1 << CS01) | (1 << CS00);
    OCR0 = 249;
    TIMSK |= (1 << OCIE0);
    sei();
}

```

```
}
```

```
uint32_t millis(void)
```

```
{
```

```
    uint32_t ms;
```

```
    cli();
```

```
    ms = millis_count;
```

```
    sei();
```

```
    return ms;
```

```
}
```

```
int main(void) {
```

```
    timer0_init();
```

```
    DDRD |= (1<<PD0)|(1<<PD1)|(1<<PD2)|(1<<PD3)|(1<<PD4);
```

```
    PORTD |= (1<<PD0);
```

```
    PORTD |= (1<<PD1);
```

```
    PORTD |= (1<<PD2);
```

```
    PORTD |= (1<<PD3);
```

```
    PORTD |= (1<<PD4);
```

```
    DDRC &= ~(1<<6);
```

```
    PORTC |= (1<<6);
```

```
    DDRC &= ~(1<<7);
```

```
    PORTC |= (1<<7);
```

```

LCDinit();
LCDcursorOFF();
LCDclr();
LCDstring(" LOAD");
while (1) {
    currentTime = millis();
    for(int i = 0; i < 2; i++)
    {
        if(!(PIN_C & (1 << swPins[i]))) && (currentTime - lastInteraptTime[i] >= 50) && buttonsPres[i] == false)
        {
            LCDclr();
            if(i==0)
            {
                LCDstring(" PLAY");
            }
            if(i==1)
            {
                LCDstring(" STOP");
            }
            buttonsPres[i] = true;
            lastInteraptTime[i] = currentTime;
        }
        if((PIN_C & (1 << swPins[i])) && buttonsPres[i] == true && (currentTime - lastInteraptTime[i] >= 50))
        {
            buttonsPres[i] = false;
        }
    }
}

```

```

        lastInteraptTime[i] = currentTime;
    }

}

_delay_ms(50);

}

return (0); /* This line is never reached */

}

```

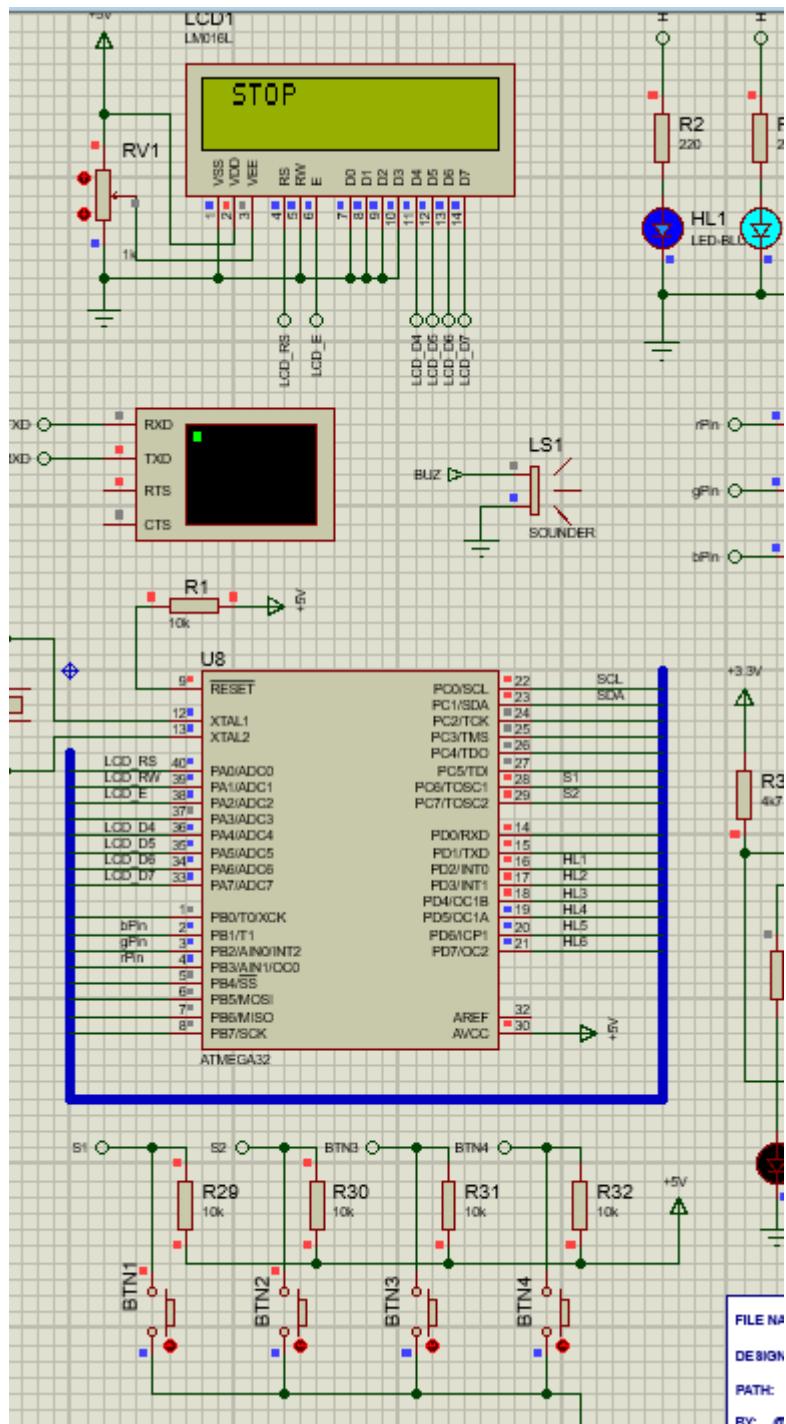


Рис. . Результат программи

**Висновок:** У ході виконання лабораторної роботи я ознайомився з принципом роботи LCD з контролером HD44780 у 4 бітному режимі підключення та дослідив можливості виведення на дисплей інформації; закріпив навички роботи з цифровими портами, тактовими кнопками, масивами.