

**Харківський національний економічний університет  
імені Семена Кузнеця**

**ЗВІТ**

**З ВИКОНАННЯ Лабораторної роботи №1  
за дисципліною: “Безпека інтернет-речей”  
На тему: «Швидке розроблення пристроїв IoT в  
середовищі віртуального моделювання Proteus»  
Варіант № 4**

**Виконав: студент факультету  
Інформаційних технологій**

**3 курсу, спец. Кібербезпека,  
групи 6.04.125.010.21.2**

**Бойко Вадим Віталійович**

**Перевірив:**

**Лимаренко В'ячеслав  
Володимирович**

**ХНЕУ ім. С. Кузнеця  
2024**

**Мета:** ознайомлення з віртуальними моделями пристроїв IoT в програмному середовищі Proteus, отримання практичних навиків швидкого налагодження програм для AVR мікроконтролерів в середовищі Proteus.

**Завдання:**

1. Ознайомитися з прикладами проектів працюючих схем в програмному середовищі Proteus для Arduino
  - a. Arduino 4 Channel Relay;
  - b. Arduino Cyrillic LCD;
  - c. Arduino Motor Shield.
2. Описати процеси, які відбуваються при моделюванні схем—прикладів

**Зміст звіту**

1. Короткий опис змісту виконання роботи.
2. Опис прикладів симуляції в Proteus:
  - a. принцип роботи;
  - b. приклади осцилограм;
  - c. дані обміну в терміналі;
  - d. оцінку використовуваних ресурсів (пам'яті програм і даних, виводів).
3. Висновки по роботі

### **Хід роботи:**

#### **1. Короткий опис змісту виконання роботи**

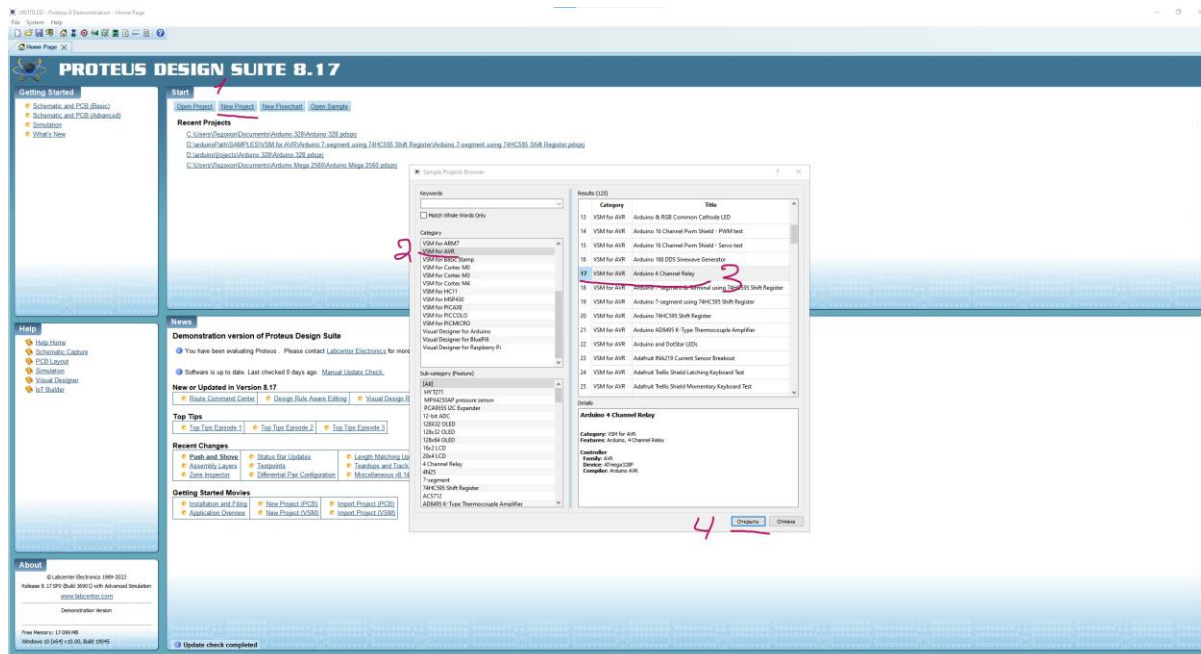
Робота має наступні етапи:

- a. Ознайомлення з середовищем Proteus:
  - i. Вивчення інтерфейсу Proteus, його панелей інструментів та бібліотек компонентів.
  - ii. Ознайомлення з основними функціональними можливостями програми.
- b. Вивчення прикладів проектів:
  - i. Аналіз готових віртуальних схем, побудованих на основі Arduino.
  - ii. Дослідження принципів роботи цих схем, розуміння їх логіки.
- c. Створення власної віртуальної схеми:
  - i. Розробка та моделювання віртуальної схеми з використанням віртуальних компонентів.
  - ii. Написання програми для мікроконтролера, який буде використовуватися в цій схемі.
- d. Аналіз результатів:
  - i. Спостереження за роботою віртуальної схеми в Proteus.
  - ii. Аналіз сигналів, що проходять через схему, та оцінка її характеристик.
- e. Налагодження:
  - i. Виявлення та виправлення помилок, що можуть виникнути в схемі або програмі.
  - ii. Тестування та оптимізація віртуальної схеми для досягнення бажаних результатів.
- f. Підготовка звіту:
  - i. Оформлення звіту, який буде описувати:
    1. Поставлені завдання.
    2. Розроблену віртуальну схему.
    3. Написану програму для мікроконтролера.
    4. Результати моделювання та аналізу.
    5. Висновки зроблені під час роботи

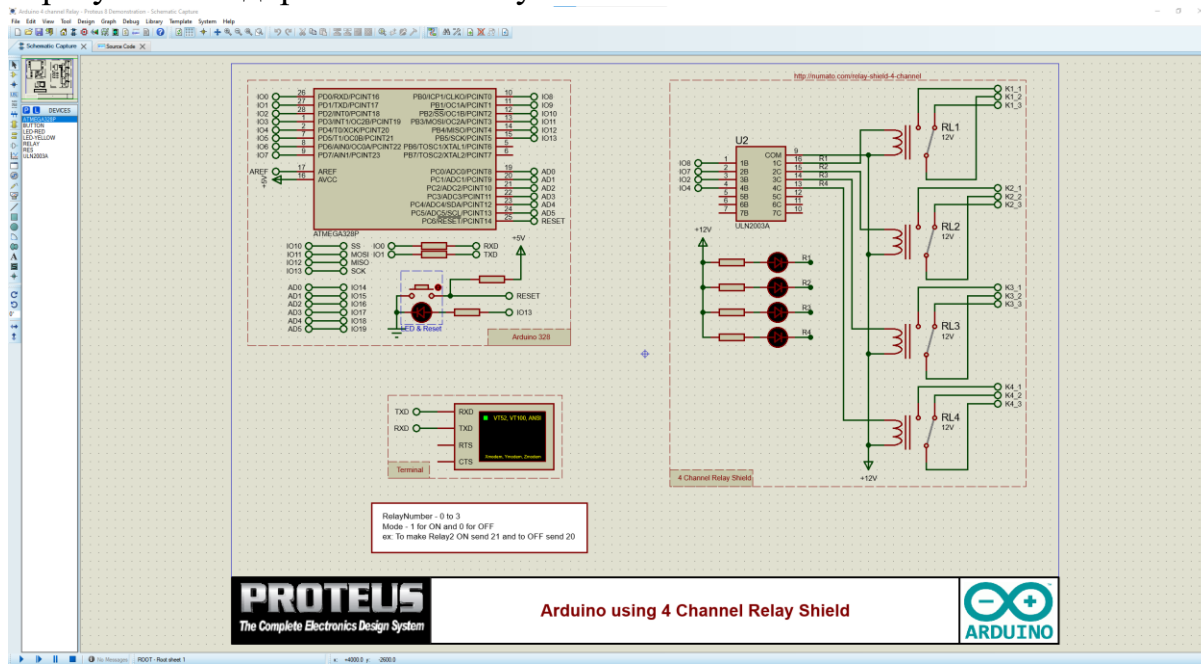
#### **2. Опис прикладів симуляції в Proteus**

##### **a. Arduino 4 Channel Relay**

Відкрию програму та оберу «VSM for AVR» та у меню справа оберу «Arduino 4 Channel Relay», та натисну «открыть»



В результаті відкривається наступна схема



Також можна подивитись на код програми

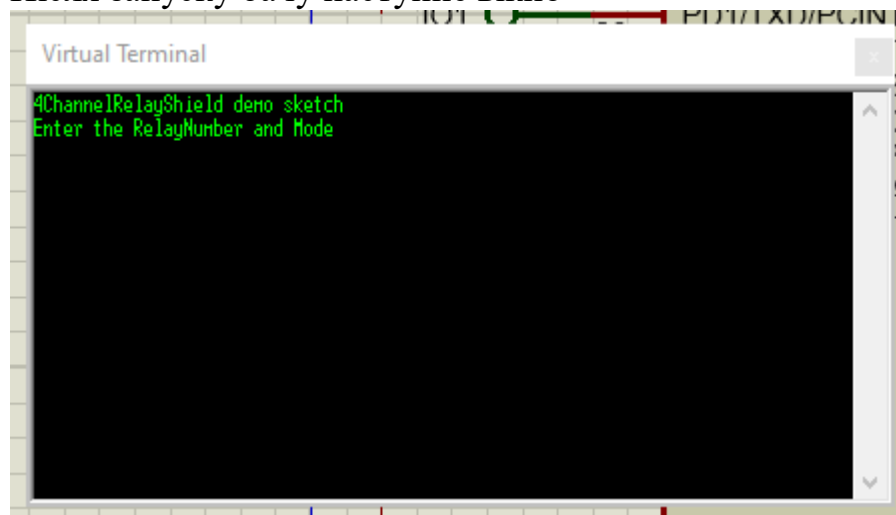
Arduino 4 channel Relay - Proteus 8 Demonstration - Source Code

```
1  /*
2  * Demo sketch for Testing 4ChannelRelayShield
3  * visit http://www.numato.com to buy 4ChannelRelayShield
4  * License: CC BY-SA
5  */
6  /*
7  * Relays 0,1,2,3 are connected to the Arduino digital pins 8,7,2,4 respectively using a ULN2003 IC.
8  * Writing HIGH to these digital pins makes corresponding relay to ON.
9  * Writing LOW to these digital pins makes corresponding relay to OFF.
10 */
11 byte RelayPins[4] = {8,7,2,4};
12 char Input[1];
13
14 void setup()
15 {
16   for(int i = 0; i < 4; i++) pinMode(RelayPins[i],OUTPUT); // Set all Relay pins to output
17   Serial.begin(9600);
18   delay(100);
19   Serial.println("4ChannelRelayShield demo sketch");
20   Serial.println("Enter the RelayNumber and Mode");
21   /*
22   * RelayNumber - 0 to 3
23   * Mode - 1 for ON and 0 for OFF
24   * ex: To make Relay2 ON send 21 and to OFF send 20
25   */
26   for(int j = 0; j < 4; j++) digitalWrite(RelayPins[j],LOW); // Default all Relays to off state
27 }
28
29 void loop()
30 {
31   String Status;
32   if (Serial.available())
33   {
34     Serial.readBytesUntil(13, Input, 2); // Read bytes until pressed ENTER
35     int State = Input[1] - 48;
36     int RelayNum = Input[0]-48;
37     if((State == 0) || (State == 1))
38     {
39       digitalWrite(RelayPins[RelayNum], State); // Make Relay ON or OFF
40     }
41     else
42     {
43       Serial.println("Incorrect parameter");
44     }
45
46     Status = String("Relay ") + String(RelayNum) + String(" State is set to ") + String(State);
47     Serial.println(Status);
48   }
49 }
50
51
52
53
```

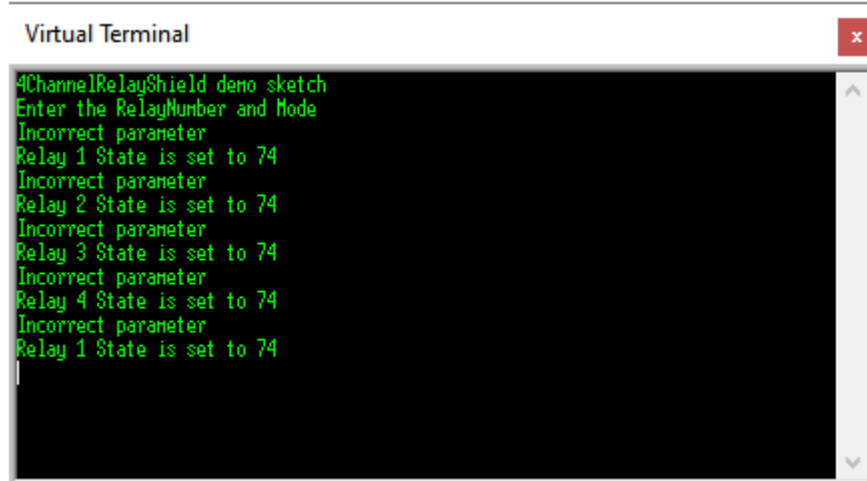
Для запуску натисну внизу на кнопку



Після запуску бачу наступне вікно



Для тесту спробую поклацати термінал

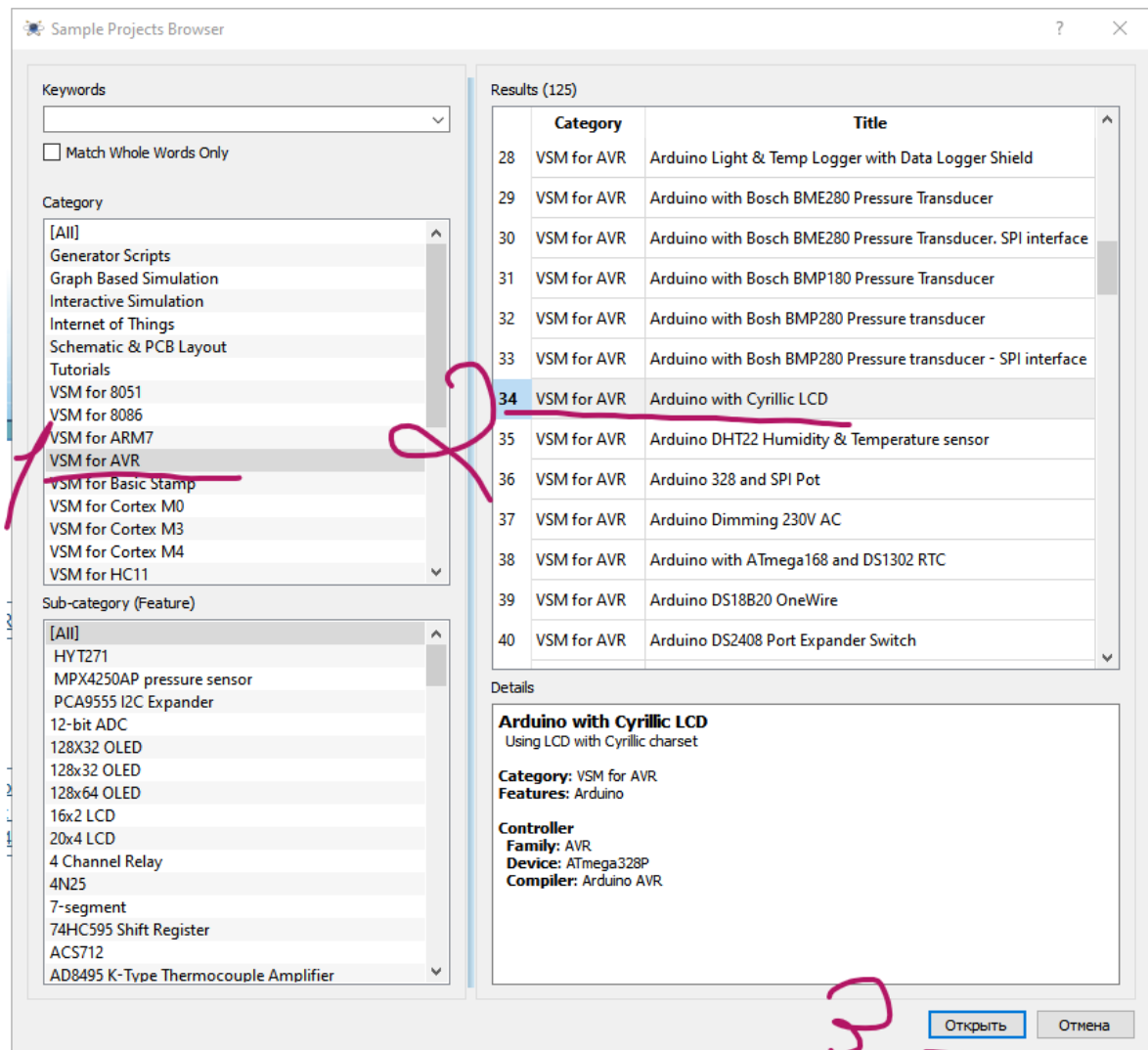


```
Virtual Terminal
4ChannelRelayShield demo sketch
Enter the RelayNumber and Mode
Incorrect parameter
Relay 1 State is set to 74
Incorrect parameter
Relay 2 State is set to 74
Incorrect parameter
Relay 3 State is set to 74
Incorrect parameter
Relay 4 State is set to 74
Incorrect parameter
Relay 1 State is set to 74
```

Опишу як це працює:

- i. Реле під'єднано до ULN2003A, що в свою чергу під'єднано до ардуіно
- ii. Також через канали телеметрії під'єднано термінал

б. Перейду до наступного прикладу 3



Знову подивлюсь код

Source Code X

main.ino X

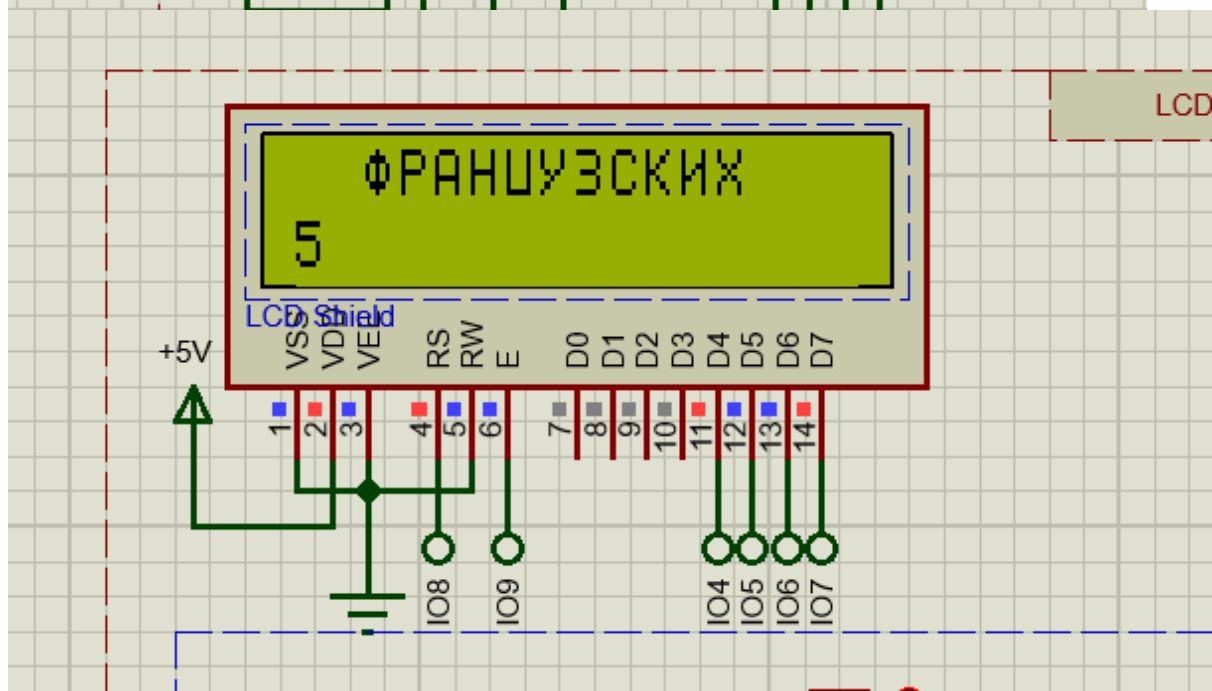
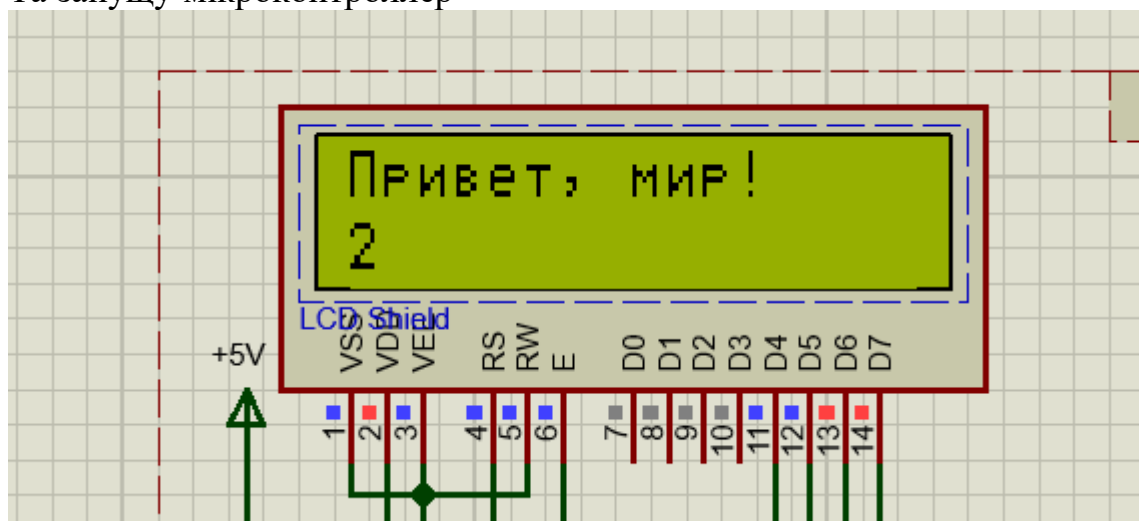
stal.cpp

stal.hpp

```
3  *
4  * CyrillicLiquidCrystal recodes symbols from Win1251 to Cyrillic LCD charset.
5  *
6  * To set custom character set to the display, open the LCD display's properties
7  * and select corresponding .BMP file in the Advanced Properties - Charset.
8  *
9  * Using Arduino with ATmega328P and Arduino LCD Shield
10 */
11
12 // include the library code:
13 #include <LiquidCrystal.h>
14 #include "CyrillicLiquidCrystal.hpp"
15
16 // initialize the library with the numbers of the interface pins
17 CyrillicLiquidCrystal lcd(8, 9, 4, 5, 6, 7);
18
19 void setup()
20 { // set up the LCD's number of columns and rows:
21   lcd.begin(16, 2);
22   // Print a message to the LCD.
23   lcd.print("Привет, мир!");
24 }
25
26 void loop()
27 { // set the cursor to column 0, line 1
28   // (note: line 1 is the second row, since counting begins with 0):
29   lcd.setCursor(0, 1);
30   // print the number of seconds since reset:
31   lcd.print(millis()/1000);
32   lcd.setCursor(0, 0);
33   switch (millis()/1000)
34   { case 3:
35     // On the 3rd second, test the character set
36     lcd.print("ТАК СЪЕШЬ ЖЕ ЕЩЁ");
37     break;
38     case 4:
39     lcd.print(" ЭТИХ МЯГКИХ ");
40     break;
41     case 5:
42     lcd.print(" ФРАНЦУЗСКИХ ");
43     break;
44     case 6:
45     lcd.print(" БУЛОЧЕК, ");
46     break;
47     case 7:
48     lcd.print(" ДА ВЫПЕЙ ЧАЮ ");
49     break;
50     case 9:
51     lcd.print("так съешь же ещё");
52     break;
53     case 10:
54     lcd.print(" ЭТИХ МЯГКИХ ");
55     break;
56     case 11:
57     lcd.print(" французских ");
58     break;
59     case 12:
60     lcd.print(" булочек, ");
61     break;
62     case 13:
63     lcd.print(" да выпей чаю ");
64     break;
65     case 15:
66     lcd.print(" ");
67     break;
68     case 16:
69     // Switch recode off
70     lcd.enableRecode(false);
71     lcd.print("проверка связи ");
72     break;
73   }
74 }
75
76
```



Та запуску мікроконтроллер



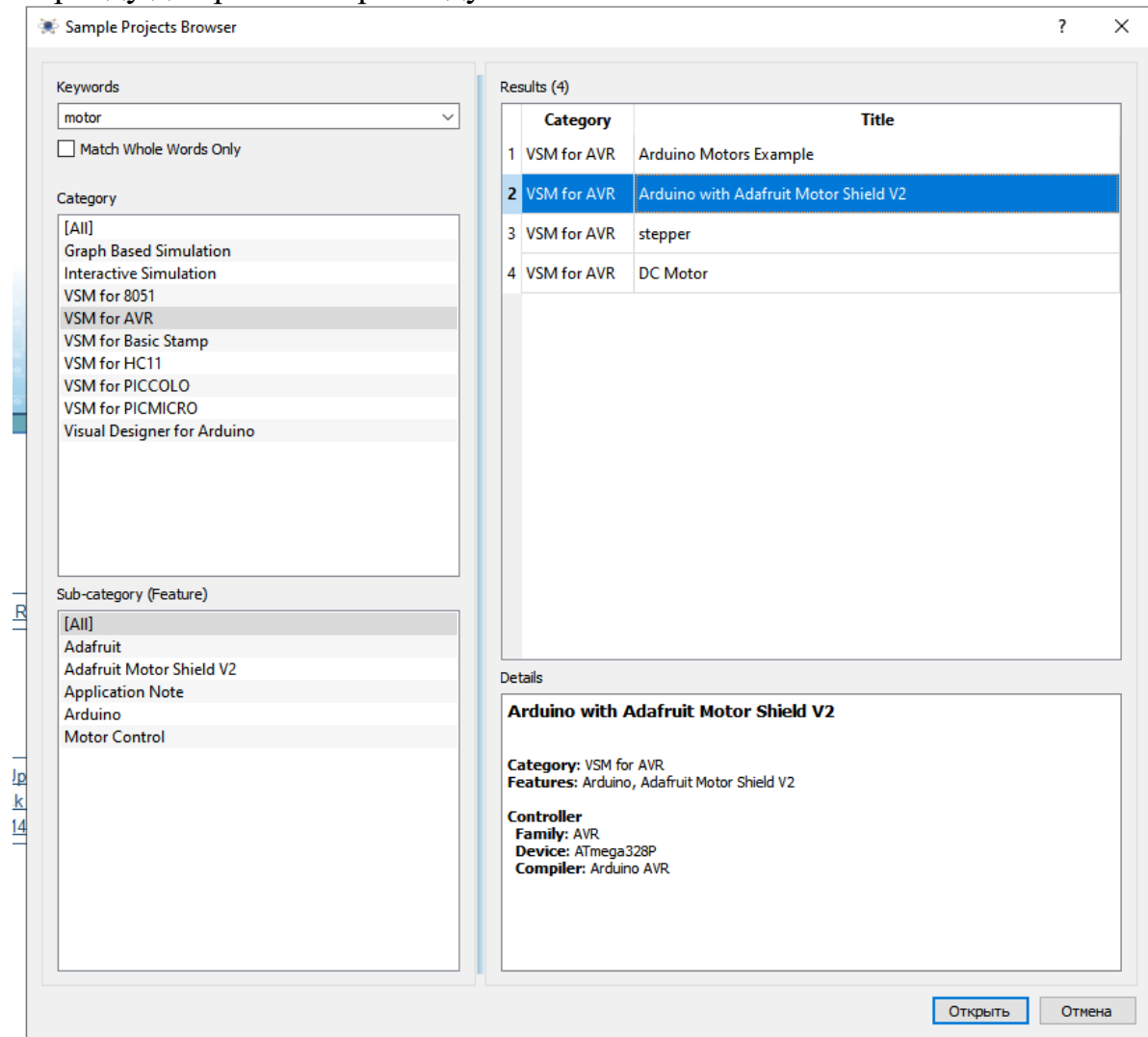
можна побачити, що змінюються дані на дисплеї

Та зупиню програму натиснув на

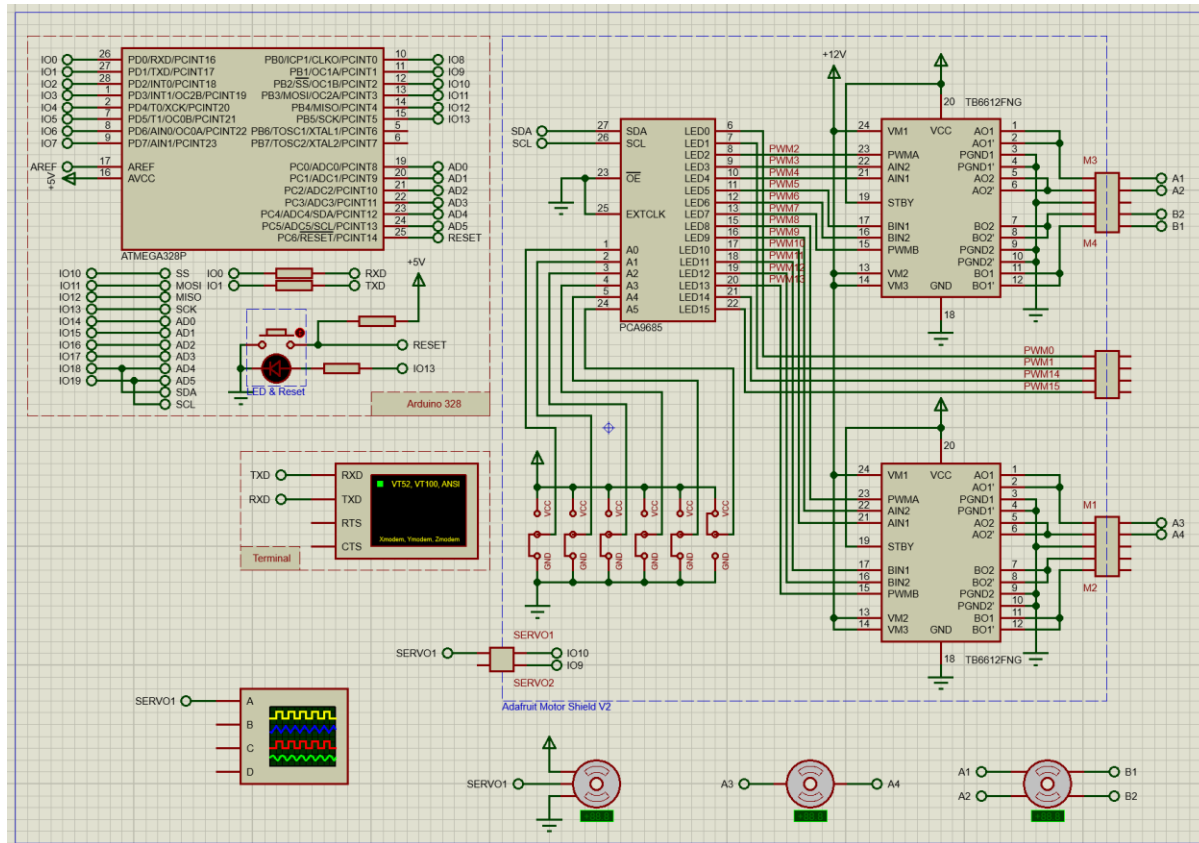


Проте щоб зрозуміти як воно працює достатньо подивитись на під'єднання до дисплею, та можна зрозуміти, що є канали по яким передається інформація

с. Перейду до третього прикладу



Бачу наступну схему

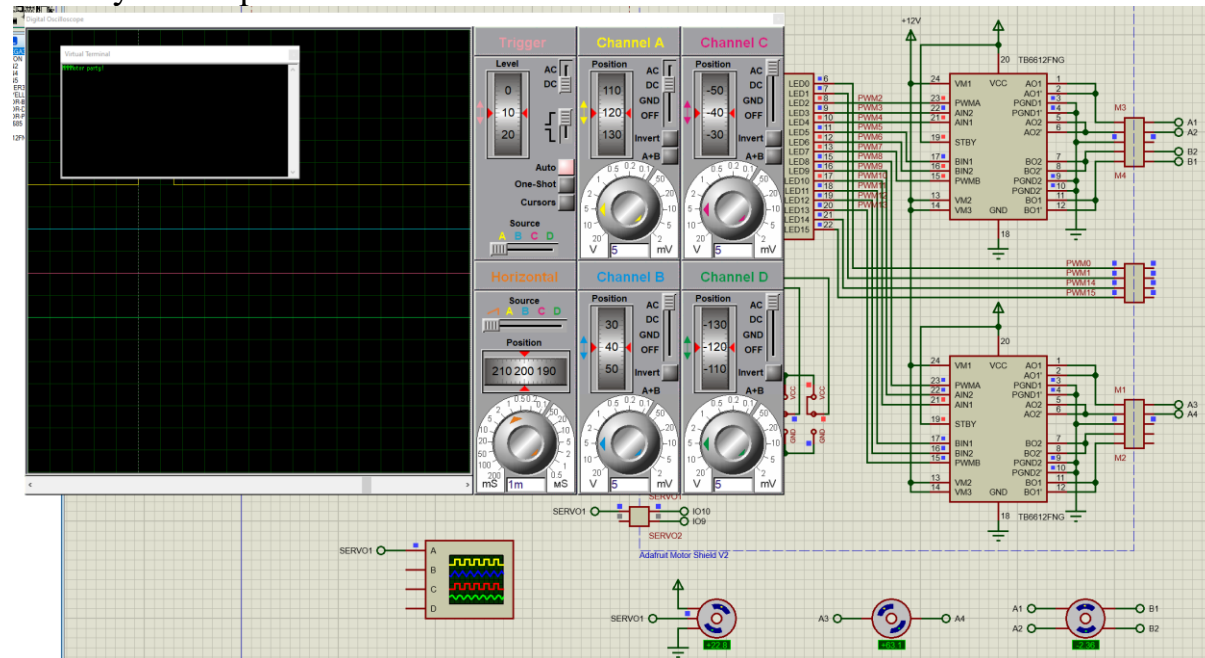


Та переходжу до коду

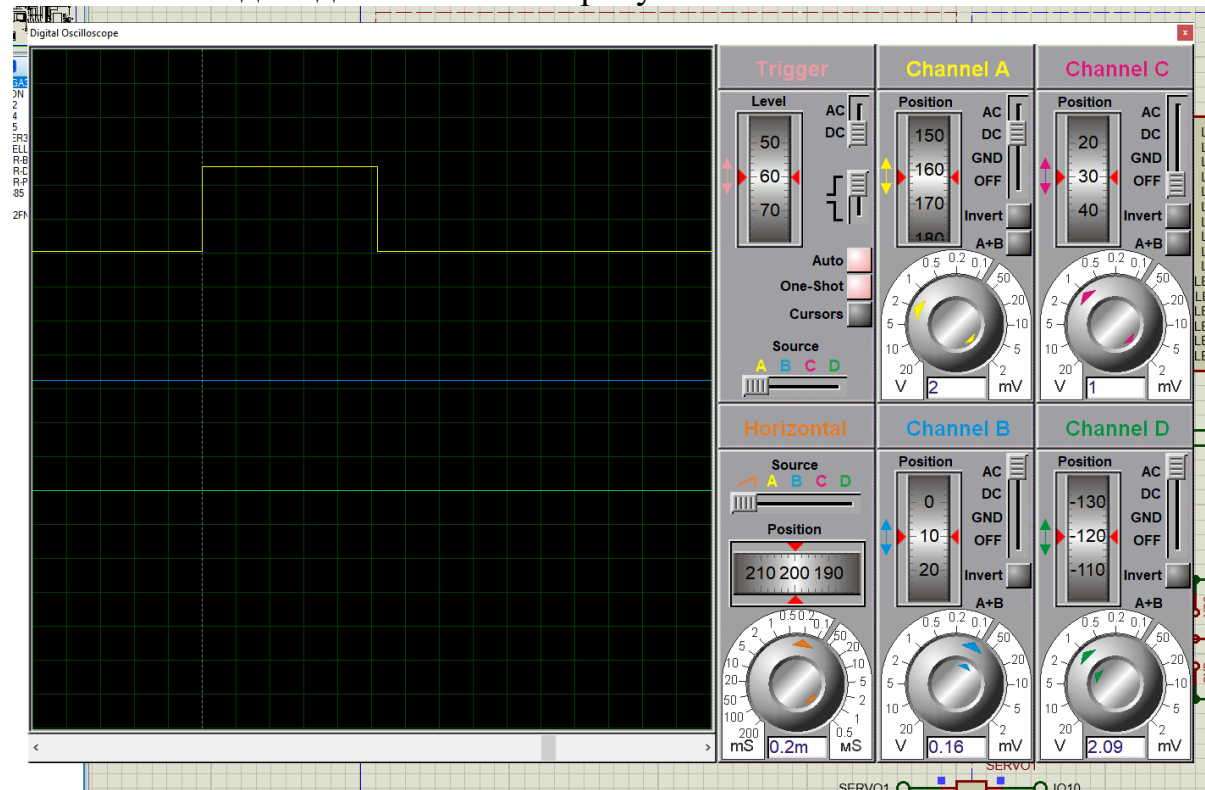
main.ino

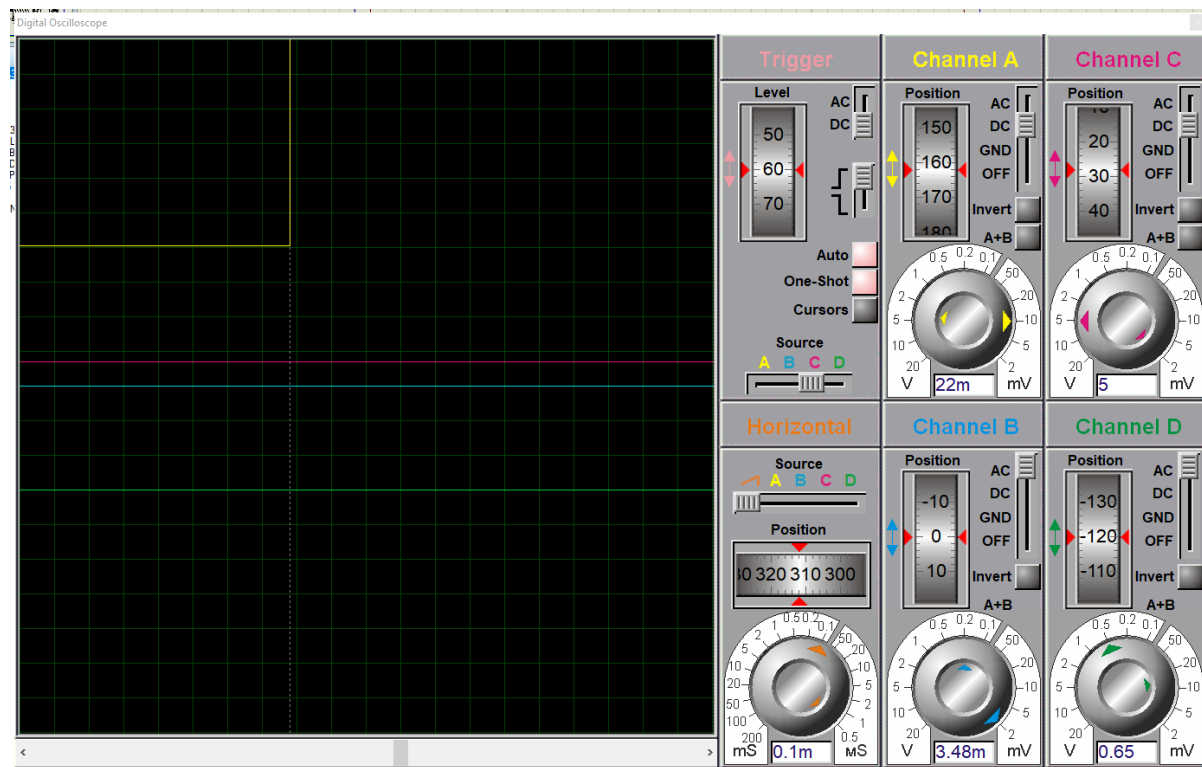
```
16 #include <Adafruit_MotorShield.h>
17 #include "Adafruit_PWM_Servo_Driver.h"
18 #include <Servo.h>
19
20 // Create the motor shield object with the default I2C address
21 Adafruit_MotorShield AFMS = Adafruit_MotorShield();
22 // Or, create it with a different I2C address (say for stacking)
23 // Adafruit_MotorShield AFMS = Adafruit_MotorShield(0x61);
24
25 // Connect a stepper motor with 200 steps per revolution (1.8 degree)
26 // to motor port #2 (M3 and M4)
27 //Adafruit_StepperMotor *myStepper = AFMS.getStepper(200, 2);
28
29 // Simulated Stepper motor with 72 step per revolution (5 degree). For simulation purposes.
30 Adafruit_StepperMotor *myStepper = AFMS.getStepper(72, 2);
31
32 // And connect a DC motor to port M1
33 Adafruit_DCMotor *myMotor = AFMS.getMotor(1);
34
35 // We'll also test out the built in Arduino Servo Library
36 Servo servol;
37
38
39 void setup() {
40   Serial.begin(9600);           // set up Serial Library at 9600 bps
41   Serial.println("Motor party!");
42
43   //AFMS.begin(); // create with the default frequency 1.6KHz
44   AFMS.begin(50); // For simulation purposes default frequency must be made low.
45
46   // Attach a servo to pin #10
47   servol.attach(10);
48
49   // turn on motor M1
50   myMotor->setSpeed(200);
51   myMotor->run(RELEASE);
52
53   // setup the stepper
54   myStepper->setSpeed(10); // 10 rpm
55 }
56
57 int i;
58 void loop() {
59   myMotor->run(FORWARD);
60   for (i=0; i<255; i++) {
61     servol.write(map(i, 0, 255, 0, 180));
62     myMotor->setSpeed(i);
63     myStepper->step(1, FORWARD, INTERLEAVE);
64     delay(3);
65   }
66
67   for (i=255; i!=0; i--) {
68     servol.write(map(i, 0, 255, 0, 180));
69     myMotor->setSpeed(i);
70     myStepper->step(1, BACKWARD, INTERLEAVE);
71     delay(3);
72   }
73
74   myMotor->run(BACKWARD);
75   for (i=0; i<255; i++) {
76     servol.write(map(i, 0, 255, 0, 180));
77     myMotor->setSpeed(i);
78     myStepper->step(1, FORWARD, DOUBLE);
79     delay(3);
80   }
81
82   for (i=255; i!=0; i--) {
83     servol.write(map(i, 0, 255, 0, 180));
84     myMotor->setSpeed(i);
85     myStepper->step(1, BACKWARD, DOUBLE);
86     delay(3);
87   }
88 }
89
```

Та запускаю проект



Змінюючи введені дані змінюється рисунок на схемі





3. Висновок: я попрацював з середовищем програми Proteus, подивився на приклади, та проаналізував як воно працює