МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЧЕРНІГІВСЬКА ПОЛІТЕХНІКА»

Навчально-науковий інститут електронних та інформаційних технологій Кафедра інформаційних та комп’ютерних систем

**ЗВІТ**

про виконання лабораторних робіт

з дисципліни «Технологія інтернету речей»

Варіант №4

Виконав:

студент групи КІт-211

Керівник: к.т.н.

Оленченко І.Р.

Красножон О.В.

Чернігів 2023

**ЗМІСТ**

[ЛАБОРАТОРНА РОБОТА №1………………………………………….3](#Лабораторна1) [ЛАБОРАТОРНА РОБОТА №2………………………………………….8](#Лабораторна2) [ЛАБОРАТОРНА РОБОТА №3………………………………………….13](#Лабораторна3)

**1** **Лабораторна робота №1**

**РОЗРОБКА ПРИСТРОЇВ “ІНТЕРНЕТУ РЕЧЕЙ” В СЕРЕДОВИЩІ МОДЕЛЮВАННЯ PROTEUS**

**Мета роботи:** Ознайомитися із функціональними можливостями середовища віртуального моделювання Proteus. Ознайомитися із віртуальними моделями пристроїв “Інтернету речей”. Навчитися створювати схеми IoT-пристроїв і виконувати їх симуляцію. Отримати практичні навички відлагодження програм, написаних для мікроконтролерів AVR в середовищі Proteus.

**1 КОРОТКІ ТЕОРЕТИЧНІ ВІДОМОСТІ**

Розробником середовища віртуального моделювання Proteus є компанія Labcenter Electronics (Великобританія). Сайт розробника - http://www.labcenter.co.uk/. Відмінність цього середовища від аналогічних за призначенням (наприклад, Electronics Workbench, Multisim Ultiboard, MicroCap, Tina) полягає у розвиненій системі симуляції (інтерактивного налагодження як в режимі реального часу, так і покроково) для різних сімейств мікроконтролерів: Intel 8051, AVR і PIC (компанії Microchip), ARM (різних виробників) та ін.

Proteus має в своєму складі велику кількість як бібліотек окремих компонентів, так і периферійних пристроїв: світлодіодних і рідкокристалічних індикаторів, датчиків температури, годинників реального часу (RTC), інтерактивних елементів вводу/виводу (кнопок, перемикачів, віртуальних портів і віртуальних вимірювальних приладів), інтерактивних графічних об’єктів, які не завжди присутні в інших подібних програмах.

Середовище Proteus VSM складається із двох самостійних частин: ISIS і ARES. Основним призначенням підсистеми ARES є робота із проектування, трасування та виготовлення друкованих плат (PCB – Printed Circuit Board). Підсистема ISIS, що має безпосередній зв'язок із ARES, призначена для створення електричних принципових схем, написання та редагування коду програмованих пристроїв, а також моделювання і відлагодження їх роботи.

У спрощеному вигляді роботу в середовищі моделювання Proteus ISIS можна представити наступним алгоритмом:

1) Вибір необхідної елементної бази, створення схеми електричної принципової пристрою або системи, які розробляються.

2) Налаштування необхідних початкових параметрів компонентів для їх функціонування.

3) Розміщення необхідних віртуальних пристроїв (вимірювання та/або візуалізації), налаштування режимів їх роботи.

4) Написання та відлагодження програм для мікроконтролерів.

5) Симуляція або проведення спеціалізованого аналізу.

Більш детальне ознайомлення із можливостями середовища моделювання Proteus слід починати із ознайомлення із проектами-прикладами. Зокрема, в Proteus VSM реалізовано підтримку наборів інструментів для роботи із платформою Arduino. Це дозволяє розробляти прототипи проектів для Arduino безпосередньо в самому середовищі Proteus, тобто, редактор коду є вбудованим і не вимагає використання, наприклад, Arduino IDE. Така функція проекту є дуже корисною, оскільки проекти можна легко створити для плат Arduino різних версій.

**2 ХІД ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ**

Завдання для лабораторної роботи зображено в таблиці 2.1

Таблиця 2.1 – Завдання

|  |  |  |
| --- | --- | --- |
| № варіанту (бригади) | Конфігурація пристрою (компонент / кількість) | Примітки |
| 4 | BME280 / 1;  Servo / 3 | Кут повороту першого сервоприводу (-90°...+90°) визначається пропорційно виміряній температурі (-25°С...+65°С); другого – пропорційно виміряній відносній вологості (0%...90%); третього – пропорційно виміряному тиску (600 гПа...1050 гПа) |

Код програми наведено в лістингу 2.2

Лістинг 2.2

|  |
| --- |
| #include <Wire.h> |
| #include <Adafruit\_Sensor.h> |
| #include <Adafruit\_BME280.h> |
| #include <Servo.h> |
| #include <Adafruit\_I2CDevice.h> |
| #include <SPI.h> |
|  |
| #define SERVO\_PIN\_1 3 |
| #define SERVO\_PIN\_2 5 |
| #define SERVO\_PIN\_3 6 |
|  |
| #define MIN\_TEMP -25 |
| #define MAX\_TEMP 65 |
| #define MIN\_HUMIDITY 0 |
| #define MAX\_HUMIDITY 90 |
| #define MIN\_PRESSURE 600 |
| #define MAX\_PRESSURE 1050 |
|  |
| #define BME\_SCK 13 |
| #define BME\_MISO 12 |
| #define BME\_MOSI 11 |
| #define BME\_CS 10 |
|  |
| Adafruit\_BME280 bme(BME\_CS, BME\_MOSI, BME\_MISO, BME\_SCK); |
| Servo servo1, servo2, servo3; |
|  |
| void setup() { |
| // Встановлюємо швидкість передачі даних через інтерфейс I2C |
| Serial.begin(9600); |
|  |
| // Адреса пристрою |
| if (!bme.begin(0x76)) { |
| Serial.println("Could not find a valid BME280 sensor, check wiring!"); |
| while (1); |
| } |
| // Налаштування пінів для сервоприводів |
| servo1.attach(SERVO\_PIN\_1); |
| servo2.attach(SERVO\_PIN\_2); |
| servo3.attach(SERVO\_PIN\_3); |
|  |
| servo1.write(90); |
| servo2.write(90); |
| servo3.write(90); |
|  |
| delay(2000); |
| } |
| void loop() { |
| // Зчитування значень температури, відносної вологості та тиску |
| float temperature = bme.readTemperature(); |
| float humidity = bme.readHumidity(); |
| float pressure = bme.readPressure() / 100.0F; |
| // Обмеження значень температури, відносної вологості та тиску відповідно до вимог завдання |
| int servo1\_pos = map(temperature, MIN\_TEMP, MAX\_TEMP, 0, 180); |
| int servo2\_pos = map(humidity, MIN\_HUMIDITY, MAX\_HUMIDITY, 0, 180); |
| int servo3\_pos = map(pressure, MIN\_PRESSURE, MAX\_PRESSURE, 0, 180); |
|  |
| servo1\_pos = constrain(servo1\_pos, 0, 180); |
| servo2\_pos = constrain(servo2\_pos, 0, 180); |
| servo3\_pos = constrain(servo3\_pos, 0, 180); |
| // Керування сервоприводами |
| servo1.write(servo1\_pos); |
| servo2.write(servo2\_pos); |
| servo3.write(servo3\_pos); |
| // Затримка 1 секунда між кожним оновленням значень |
| delay(1000);  } |

На рисунку 2.1 зображено схему пристрою

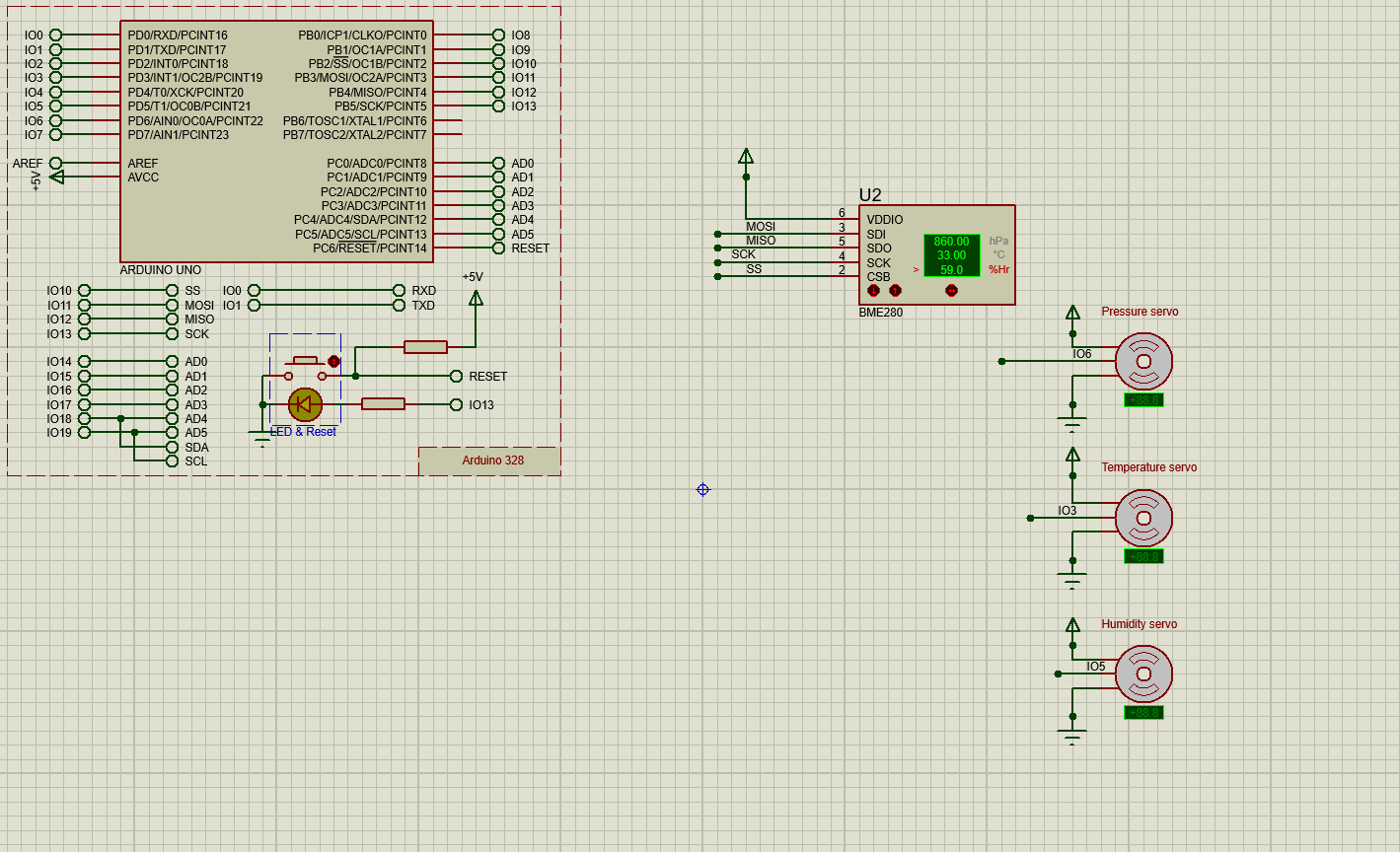


Рисунок 2.1 – Схема пристрою

На рисунку 2.2 та 2.3 зображено роботу пристрою.

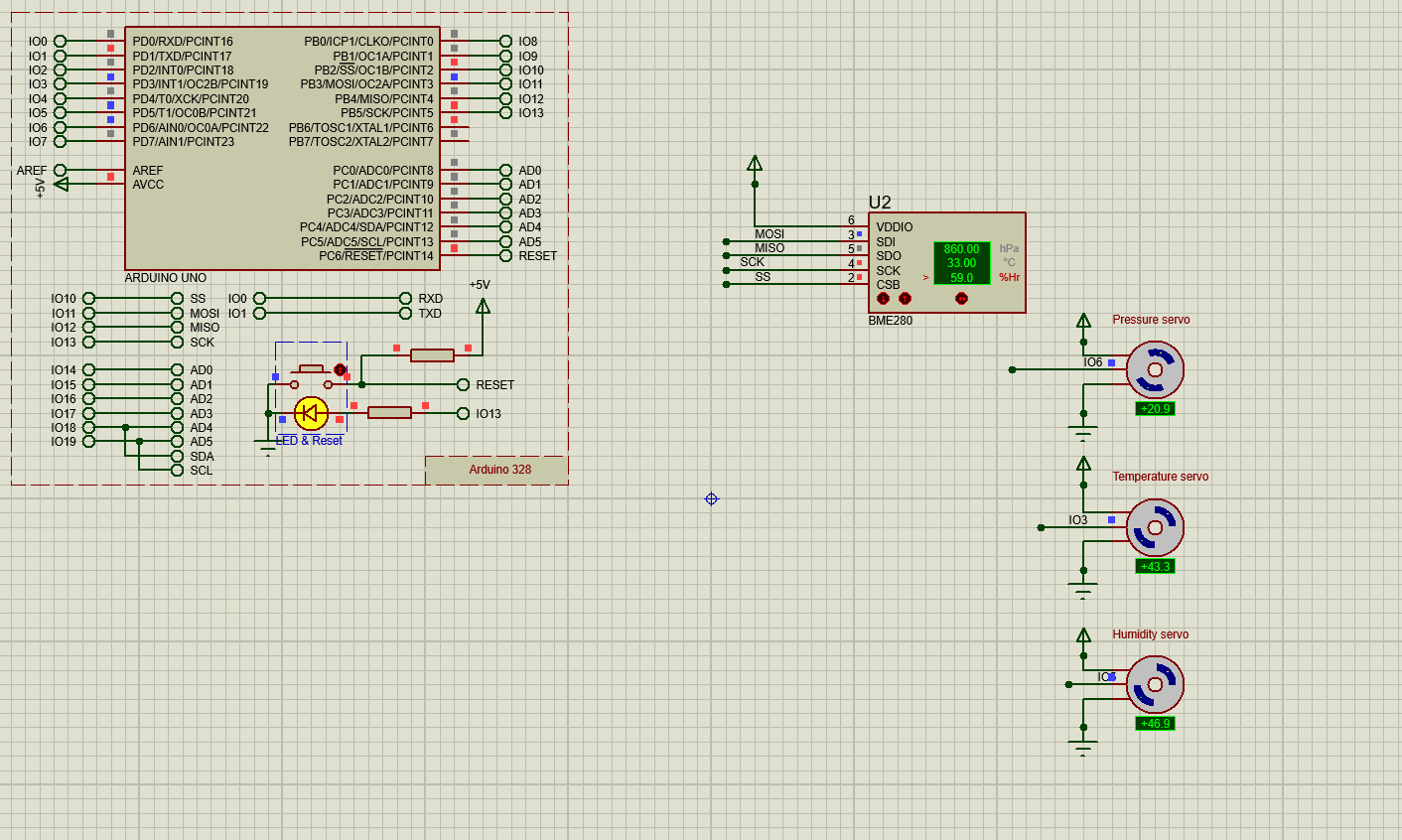


Рисунок 2.2 – Поворот сервоприводу до +

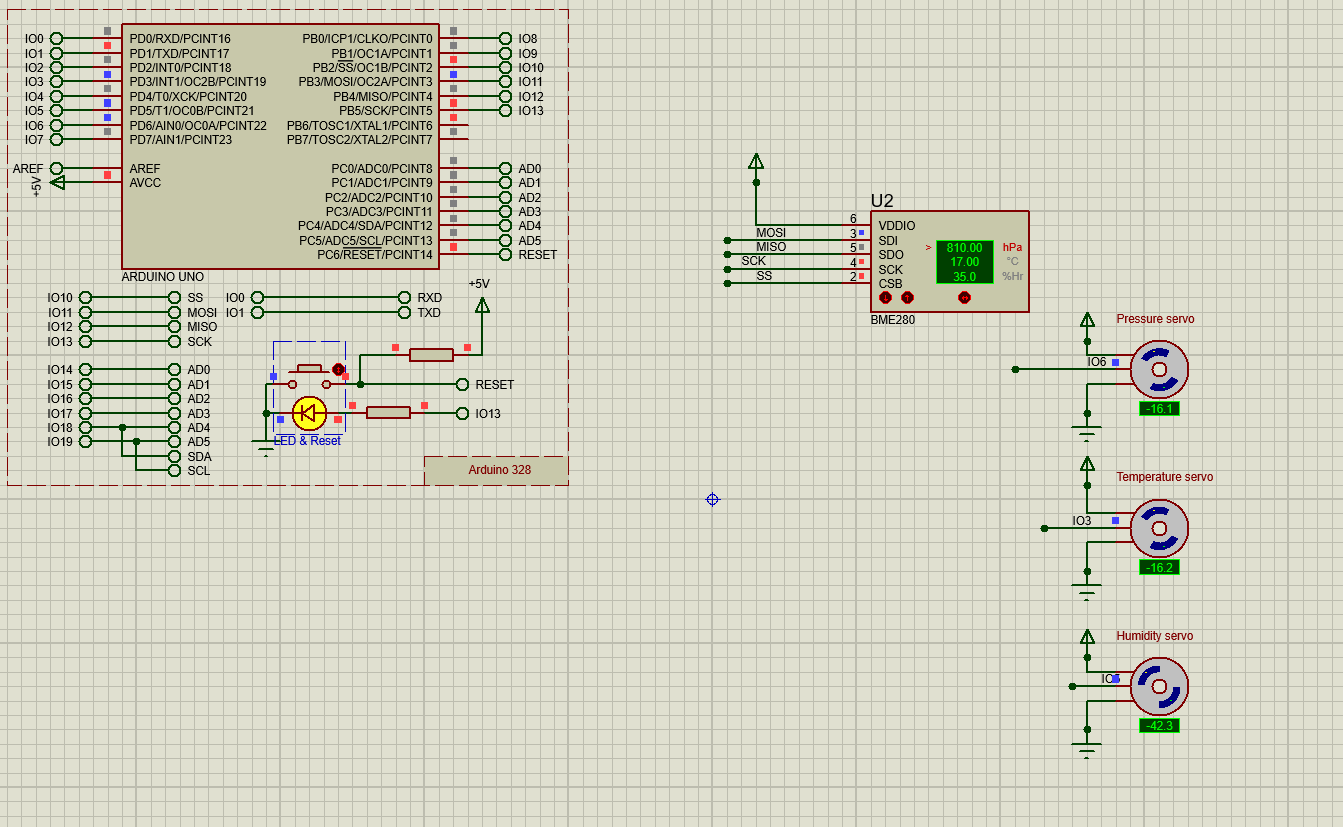


Рисунок 2.3 – Поворот сервоприводу до –

Особливості функціонування Proteus VSM:

* можливість створення та симуляції складних електронних схем з різними типами компонентів, включаючи мікроконтролери, сенсори та інші пристрої;
* наявність бібліотек компонентів, які містять багато моделей компонентів, що відповідають різним стандартам;
* зручний інтерфейс користувача, який дозволяє легко налаштовувати та керувати симуляцією електронної схеми;
* можливість відстежування та аналізу поведінки електронної схеми під час симуляції, включаючи відлагодження коду мікроконтролера та аналіз результатів симуляції;
* наявність функцій інспектора компонентів, що дозволяє користувачеві перевірити параметри та стан компонентів в режимі реального часу.

**ВИСНОВКИ**

Ознайомився із функціональними можливостями середовища віртуального моделювання Proteus. Ознайомився із віртуальними моделями пристроїв “Інтернету речей”. Навчився створювати схеми IoT-пристроїв і виконувати їх симуляцію. Отримав практичні навички відлагодження програм, написаних для мікроконтролерів AVR в середовищі Proteus.

**1** **Лабораторна робота №2**

**РОЗРОБКА ГРАНИЧНИХ ПРИСТРОЇВ “ІНТЕРНЕТУ РЕЧЕЙ” В СЕРЕДОВИЩІ МОДЕЛЮВАННЯ PROTEUS**

**Мета роботи:** Ознайомитися із функціональними можливостями середовища віртуального моделювання Proteus. Ознайомитися із віртуальними моделями пристроїв “Інтернету речей”. Навчитися створювати схеми IoT-пристроїв і виконувати їх симуляцію. Отримати практичні навички відлагодження програм, написаних для мікроконтролерів AVR в середовищі Proteus.

**1 КОРОТКІ ТЕОРЕТИЧНІ ВІДОМОСТІ**

Розробником середовища віртуального моделювання Proteus є компанія Labcenter Electronics (Великобританія). Сайт розробника - http://www.labcenter.co.uk/. Відмінність цього середовища від аналогічних за призначенням (наприклад, Electronics Workbench, Multisim Ultiboard, MicroCap, Tina) полягає у розвиненій системі симуляції (інтерактивного налагодження як в режимі реального часу, так і покроково) для різних сімейств мікроконтролерів: Intel 8051, AVR і PIC (компанії Microchip), ARM (різних виробників) та ін.

Proteus має в своєму складі велику кількість як бібліотек окремих компонентів, так і периферійних пристроїв: світлодіодних і рідкокристалічних індикаторів, датчиків температури, годинників реального часу (RTC), інтерактивних елементів вводу/виводу (кнопок, перемикачів, віртуальних портів і віртуальних вимірювальних приладів), інтерактивних графічних об’єктів, які не завжди присутні в інших подібних програмах.

Середовище Proteus VSM складається із двох самостійних частин: ISIS і ARES. Основним призначенням підсистеми ARES є робота із проектування, трасування та виготовлення друкованих плат (PCB – Printed Circuit Board). Підсистема ISIS, що має безпосередній зв'язок із ARES, призначена для створення електричних принципових схем, написання та редагування коду програмованих пристроїв, а також моделювання і відлагодження їх роботи.

У спрощеному вигляді роботу в середовищі моделювання Proteus ISIS можна представити наступним алгоритмом:

1. вибір необхідної елементної бази, створення схеми електричної принципової пристрою або системи, які розробляються;
2. налаштування необхідних початкових параметрів компонентів для їх функціонування;
3. розміщення необхідних віртуальних пристроїв (вимірювання та/або візуалізації), налаштування режимів їх роботи;
4. написання та відлагодження програм для мікроконтролерів;
5. симуляція або проведення спеціалізованого аналізу.

Більш детальне ознайомлення із можливостями середовища моделювання Proteus слід починати із ознайомлення із проектами-прикладами. Зокрема, в Proteus VSM реалізовано підтримку наборів інструментів для роботи із платформою Arduino. Це дозволяє розробляти прототипи проектів для Arduino безпосередньо в самому середовищі Proteus, тобто, редактор коду є вбудованим і не вимагає використання, наприклад, Arduino IDE. Така функція проекту є дуже корисною, оскільки проекти можна легко створити для плат Arduino різних версій.

**2 ХІД ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ**

Завдання для лабораторної роботи зображено в таблиці 2.1.

Таблиця 2.1 – Завдання до лабораторної роботи №2

|  |  |  |
| --- | --- | --- |
| № варіанту (бригади) | Конфігурація пристрою (компонент / кількість) | Примітки |
| 4 | BME280/1 | Кожні 10 с ведучий МК видає команду до веденого на вимірювання трьох параметрів.  Результати вимірювання датчиком, приєднаним до веденого МК, передаються ведучому.  Ведучий здійснює розрахунок середнього геометричного кожного із параметрів окремо і раз на хвилину виводить результати у віртуальний термінал |

Код програми наведено в лістингу 2.2

Контролер ARD1 – ведучий. Контролер ARD2 – ведений.

Лістинг 2.1 – Код програми ведучого контролера

|  |
| --- |
| #include<SPI.h>  void setup (void){  Serial.begin(115200);      SPI.begin();  SPI.setClockDivider(SPI\_CLOCK\_DIV8);  digitalWrite(SS,HIGH);  delay(100);  }  void loop(void){      double mTemperature=1, mAltitude = 1, mPressure = 1;  for(int i=0; i < 6; i++){  digitalWrite(SS, LOW);  SPI.transfer(1);  delay(30);  mTemperature = mTemperature \* (int)map(SPI.transfer(2), 0, 200, -100, 100) / 100.0;  delay(30);  mPressure = mPressure \* (int)map(SPI.transfer(3), 0, 254, 633, 887) / 100.0;  delay(30);  mAltitude = mAltitude \* (int)SPI.transfer(3) / 100.0;  delay(9910);  }    Serial.print("Temperature ");  Serial.println(pow(mTemperature, 1.0/6)\*100);  Serial.print("Pressure in mm Hg ");  Serial.println(pow(mPressure, 1.0/6)\*100);  Serial.print("Altitude ");  Serial.println(pow(mAltitude, 1.0/6)\*100);  } |

Лістинг 2.2 - Код програми веденого контролера

|  |
| --- |
| #include<SPI.h>  #include <Wire.h>  #include <Adafruit\_BMP280.h>  volatile boolean received;  volatile byte Slavereceived;  Adafruit\_BMP280 bmp;  void setup(){  Serial.begin(115200);    pinMode(MISO,OUTPUT);  SPCR |= \_BV(SPE);  received = false;  SPI.attachInterrupt();  if(!bmp.begin(0x77)) {  Serial.println(F("Could not find a valid BMP280 sensor,check wiring!"));  }  }  ISR (SPI\_STC\_vect) {  Slavereceived = SPDR;  received = true;  }  void loop(){  if(received){  if(Slavereceived==1){  SPDR = (byte)map((int)bmp.readTemperature(), -100, 100, 0, 200);  }  if(Slavereceived==2){  SPDR = (byte)map((int)(bmp.readPressure() \* 0.00750063755419211), 633, 887, 0, 254);  }  if(Slavereceived==3){  SPDR = (byte)bmp.readAltitude(1013.25);  }  received = false;  }  delay(10);  } |

На рисунку 2.1 зображено схему пристрою.

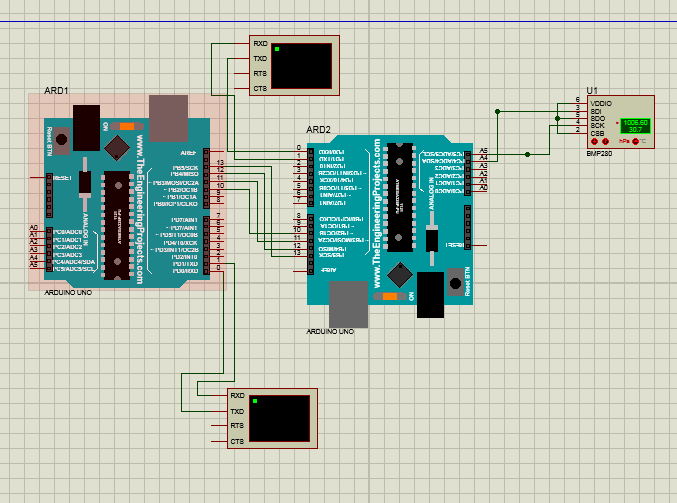


Рисунок 2.1 – Схема пристрою

На рисунку 2.2 та 2.3 зображено роботу пристрою.



Рисунок 2.2 – Початок роботи пристрою



Рисунок 2.3 – Отримання даних з веденого до ведучого та виведення середнього геометричного значення

Особливості функціонування Proteus VSM:

* можливість створення та симуляції складних електронних схем з різними типами компонентів, включаючи мікроконтролери, сенсори та інші пристрої;
* наявність бібліотек компонентів, які містять багато моделей компонентів, що відповідають різним стандартам;
* зручний інтерфейс користувача, який дозволяє легко налаштовувати та керувати симуляцією електронної схеми;
* можливість відстежування та аналізу поведінки електронної схеми під час симуляції, включаючи відлагодження коду мікроконтролера та аналіз результатів симуляції;
* наявність функцій інспектора компонентів, що дозволяє користувачеві перевірити параметри та стан компонентів в режимі реального часу.

**ВИСНОВКИ**

Ознайомився із функціональними можливостями середовища віртуального моделювання Proteus. Ознайомився із віртуальними моделями пристроїв “Інтернету речей”. Навчився створювати схеми IoT-пристроїв і виконувати їх симуляцію. Отримав практичні навички відлагодження програм, написаних для мікроконтролерів AVR в середовищі Proteus.

**1** **Лабораторна робота №3**

**РОЗРОБКА МЕРЕЖІ ПРИСТРОЇВ “ІНТЕРНЕТУ РЕЧЕЙ” В СЕРЕДОВИЩІ МОДЕЛЮВАННЯ PROTEUS ЗА ДОПОМОГОЮ ПРОТОКОЛУ MQTT**

**Мета роботи:** Ознайомитися із функціональними можливостями середовища віртуального моделювання Proteus. Ознайомитися із віртуальними моделями пристроїв “Інтернету речей”. Навчитися створювати схему мережі IoT-пристроїв і виконувати їх симуляцію. Отримати практичні навички відлагодження програм, написаних для мікроконтролерів AVR в середовищі Proteus.

**1 КОРОТКІ ТЕОРЕТИЧНІ ВІДОМОСТІ**

Розробником середовища віртуального моделювання Proteus є компанія Labcenter Electronics (Великобританія). Сайт розробника - http://www.labcenter.co.uk/. Відмінність цього середовища від аналогічних за призначенням (наприклад, Electronics Workbench, Multisim Ultiboard, MicroCap, Tina) полягає у розвиненій системі симуляції (інтерактивного налагодження як в режимі реального часу, так і покроково) для різних сімейств мікроконтролерів: Intel 8051, AVR і PIC (компанії Microchip), ARM (різних виробників) та ін.

Proteus має в своєму складі велику кількість як бібліотек окремих компонентів, так і периферійних пристроїв: світлодіодних і рідкокристалічних індикаторів, датчиків температури, годинників реального часу (RTC), інтерактивних елементів вводу/виводу (кнопок, перемикачів, віртуальних портів і віртуальних вимірювальних приладів), інтерактивних графічних об’єктів, які не завжди присутні в інших подібних програмах.

Середовище Proteus VSM складається із двох самостійних частин: ISIS і ARES. Основним призначенням підсистеми ARES є робота із проектування, трасування та виготовлення друкованих плат (PCB – Printed Circuit Board). Підсистема ISIS, що має безпосередній зв'язок із ARES, призначена для створення електричних принципових схем, написання та редагування коду програмованих пристроїв, а також моделювання і відлагодження їх роботи.

У спрощеному вигляді роботу в середовищі моделювання Proteus ISIS можна представити наступним алгоритмом:

1. вибір необхідної елементної бази, створення схеми електричної принципової пристрою або системи, які розробляються;
2. налаштування необхідних початкових параметрів компонентів для їх функціонування;
3. розміщення необхідних віртуальних пристроїв (вимірювання та/або візуалізації), налаштування режимів їх роботи;
4. написання та відлагодження програм для мікроконтролерів;
5. симуляція або проведення спеціалізованого аналізу.

Більш детальне ознайомлення із можливостями середовища моделювання Proteus слід починати із ознайомлення із проектами-прикладами. Зокрема, в Proteus VSM реалізовано підтримку наборів інструментів для роботи із платформою Arduino. Це дозволяє розробляти прототипи проектів для Arduino безпосередньо в самому середовищі Proteus, тобто, редактор коду є вбудованим і не вимагає використання, наприклад, Arduino IDE. Така функція проекту є дуже корисною, оскільки проекти можна легко створити для плат Arduino різних версій.

**2 ХІД ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ**

Завдання для лабораторної роботи зображено в таблиці 2.1.

Таблиця 2.1 – Завдання до лабораторної роботи №3

|  |  |  |
| --- | --- | --- |
| № варіанту (бригади) | Конфігурація пристрою (компонент / кількість) | Примітки |
| 4 | BME280/1 | Кожні 10 с на LCD по черзі відображаються значення температури, відносної вологості повітря і атмосферного тиску |

На рисунку 2.1 приведена схема алгоритму роботи програми відправки даних. На рисунку 2.2-2.3 приведена схема алгоритму роботи програми приймання даних.

На рисунку 2.4 приведено зображення виводу пристроїв після запуску емуляції. На рисунку 2.5-2.6 приведено зображення виводу пристрою під час роботи.

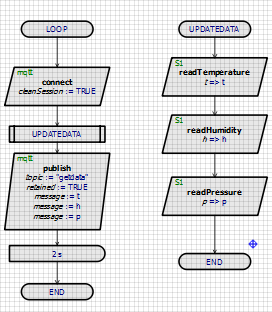


Рисунок 2.1 – Схема програми відправки даних

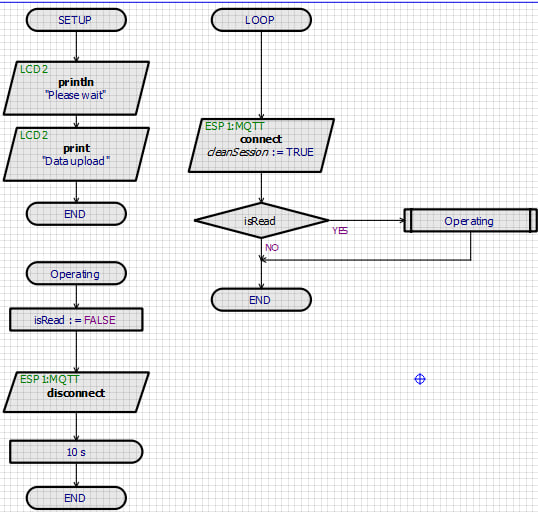


Рисунок 2.2 – Схема програми пристрою прийому даних. Ініціалізація

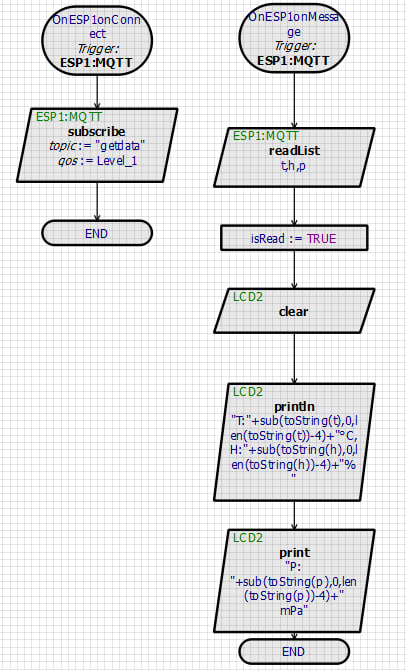


Рисунок 2.3 - Схема програми пристрою прийому даних. Обробники подій

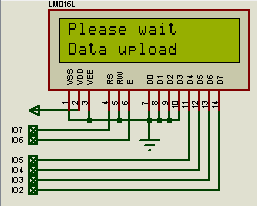


Рисунок 2.4 – Пристрій після завантаження

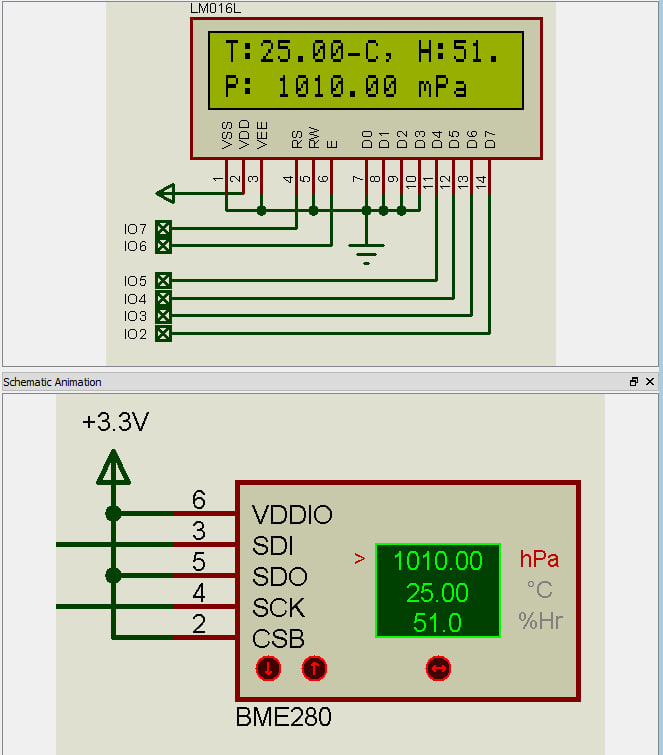


Рисунок 2.5 – Пристрій під час роботи

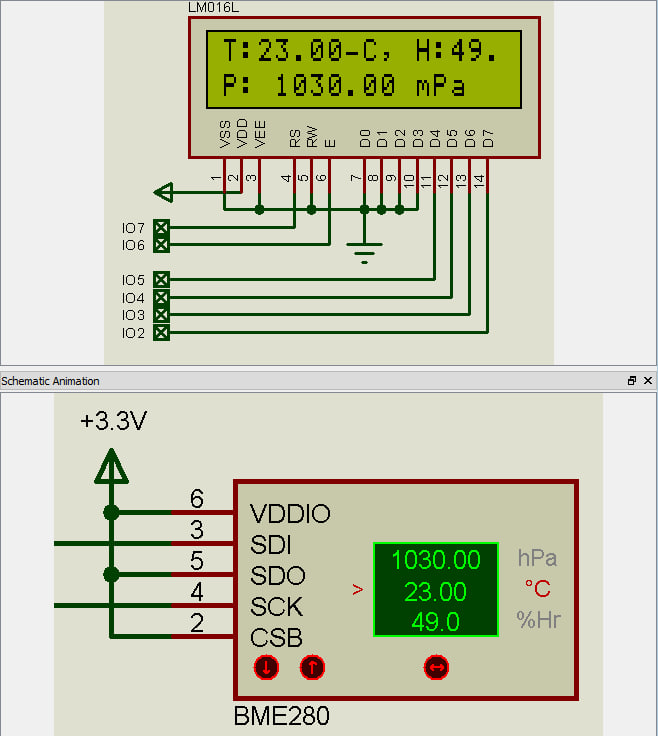


Рисунок 2.6 – Пристрій після зміни даних

Особливості функціонування Proteus VSM:

* можливість створення та симуляції складних електронних схем з різними типами компонентів, включаючи мікроконтролери, сенсори та інші пристрої;
* наявність бібліотек компонентів, які містять багато моделей компонентів, що відповідають різним стандартам;
* зручний інтерфейс користувача, який дозволяє легко налаштовувати та керувати симуляцією електронної схеми;
* можливість відстежування та аналізу поведінки електронної схеми під час симуляції, включаючи відлагодження коду мікроконтролера та аналіз результатів симуляції;
* наявність функцій інспектора компонентів, що дозволяє користувачеві перевірити параметри та стан компонентів в режимі реального часу.

**ВИСНОВКИ**

Ознайомився із функціональними можливостями середовища віртуального моделювання Proteus. Ознайомився із віртуальними моделями пристроїв “Інтернету речей”. Навчився створювати схеми IoT-пристроїв і виконувати їх симуляцію. Отримав практичні навички відлагодження програм, написаних для мікроконтролерів AVR в середовищі Proteus. Освоїли роботу з MQTT протоколом.