**BÁO CÁO THỰC HÀNH**

**GIAO TIẾP ĐIỀU KHIỂN THIẾT BỊ NGOẠI VI**

*Tên bài thực hành:Đề Số 1 - Giám Sát Trạng Thái Đóng Mở Cửa*

*Họ Và Tên :*

* Nguyễn Thành Nguyên – 18018791

Ngày thực hiện: … 16… / … 06… / 2021……

**PHẦN I: Chuẩn đầu ra môn học -1: Trình bày được cấu tạo, nguyên lý hoạt động và biết sử dụng các thành phần trong hệ thống, thiết bị điện tử**

* 1. ***Mô tả tóm tắt nội dung bài thực hành***
* Nội dung và yêu cầu :
* Hệ thống giám sát trạng thái đóng/ mở cổng : Dùng một switch để điều khiển trạng thái cổng, trạng thái cổng được hiện thị qua 2 led xanh ( báo trạng thái cổng ĐÓNG ) và đỏ ( báo trạng thái cổng MỞ ), hiện thị trạng thái lên LCD và giao diện Visual Studio, giao diện studio hiện thị được trạng thái và số lần đóng mở cửa.
* Linh kiện cần dùng :

+ PIC18f4550

+ Arduino Mega 2560  
+ Một Switch  
+ Một Led Xanh

+ Một Led Đỏ

+ Hai điện trở 330 ôm

+ Một điện trở 10k ôm

+ Mạch bluetooth HC-05

+ Một LCD 16x02

* 1. ***Sơ đồ khối của bài thực hành.***



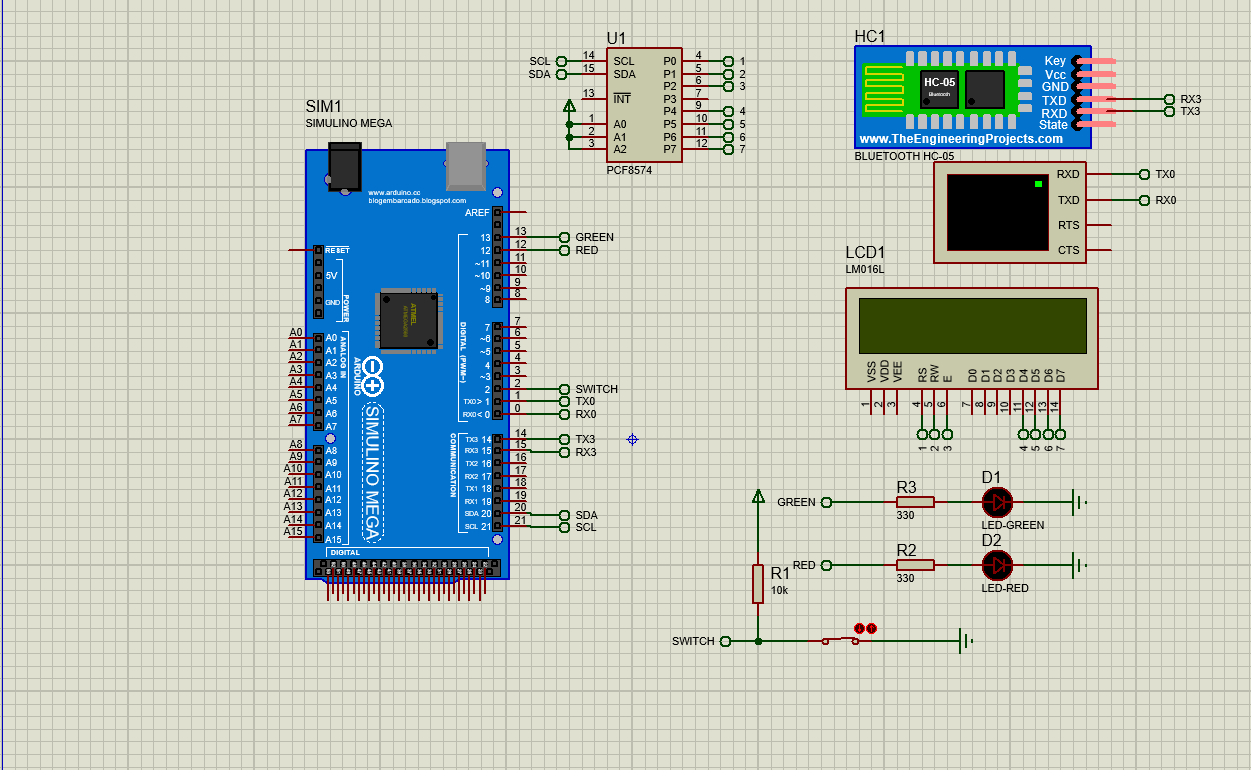
Giải thích từng khối :

* Khối Hardware bao gồm một switch và hai led ( xanh và đỏ ) đóng vai trò là ngoại vi được điều khiển để thực hiện chức năng đóng/mở cửa và hiển thị trạng thái cửa thông qua Bluetooth.
* Khối Firmware bao gồm các kit 18F4550 hoặc Arduino dùng để nạp code thực hiện các chức năng điều khiển ngoại vi led, nút nhấn
* Khối Software gồm Port Bluetooth là cổng giao tiếp giữa phần máy tính đến vi điều khiển dùng để truyền dữ liệu .Computer dóng vai trò như là lập trình để nạp vào vi điều khiển cụ thể đây là lập trình giao tiếp ngoại vi thông qua cổng Bluetooth.

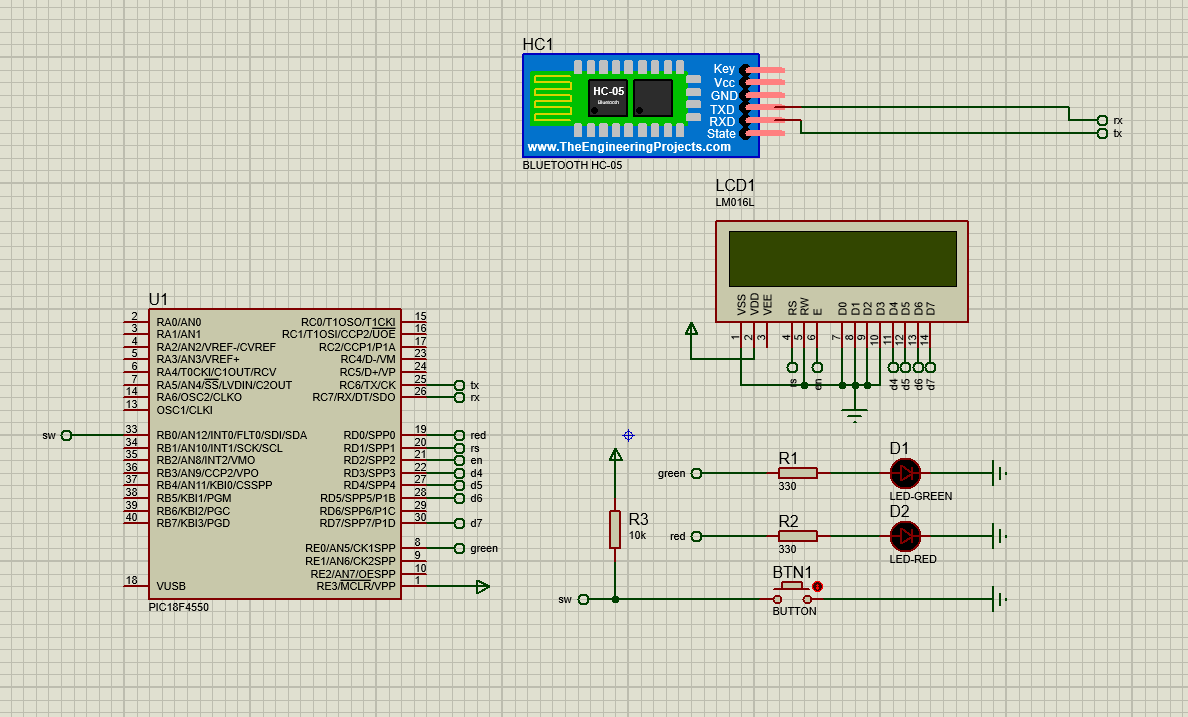
**PHẦN II: Chuẩn đầu ra môn học 2: Có khả sử dụng các phần mềm lập trình, mô phỏng**

***2.1. Khả năng sử dụng phần mềm mô phỏng, vẽ mạch - Proteus***

**2.1.1 Mạch – Proteus cho Arudino**



**2.1.1 Mạch – Proteus cho PIC**

****

***2.2. Khả năng sử dụng phần mềm lập trình Arduino***

Ảnh có chứa văn bản

Mô tả được tạo tự động

Ảnh có chứa văn bản

Mô tả được tạo tự động

***2.3. Khả năng sử dụng phần mềm lập trình cho PIC - MikroC***

Ảnh có chứa văn bản, ảnh chụp màn hình, máy tính

Mô tả được tạo tự động

Ảnh có chứa văn bản

Mô tả được tạo tự động

***2.4. Khả năng sử dụng phần mềm viết giao diện Visual Studio C#.***

Ảnh có chứa văn bản

Mô tả được tạo tự động

Ảnh có chứa văn bản, màn hình, ảnh chụp màn hình, trong nhà

Mô tả được tạo tự động

**PHẦN III. Chuẩn đầu ra môn học 3: Có khả năng phân tích, thiết kế giải thuật và lập trình ứng dụng**

***3.1. Lưu đồ giải thuật cho Arduino hoặc PIC***

***3.1.1 Lưu đồ chương trình ngắt***



***3.1.2 Lưu đồ chương trình chính***



***3.2. Lập trình code cho Arduino hoặc PIC***

**3.2.1 Code cho Arduino**

#include <Wire.h>

#include <LiquidCrystal\_I2C.h>

LiquidCrystal\_I2C lcd(0x27,16,2);

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#define max\_count\_ReceiveData 10

#define led\_Pin 13 // led xanh close

#define led1\_Pin 12 // led do close

#define sw\_Pin 2

#define led\_on HIGH

#define led\_off LOW

#define led1\_on HIGH

#define led1\_off LOW

int count = 0;

int led1\_state = 0;

int led2\_state = 0;

int last\_led1\_state = 0;

int last\_led2\_state = 0;

int count1 = 0;

int count2 = 0;

int count3 = 0;

int count5 = 0;

int count6 = 0;

int last\_swStatus = 0;

String data\_change = "";

String data\_change1 = "";

char ReceiveData;

char ReceiveData1;

char start\_sign = '@';

char end\_sign = '&';

char start1\_sign = '@';

char end1\_sign = '&';

char count\_ReceiveData = 0;

char buf\_string\_receive[max\_count\_ReceiveData];

char start\_Data, end\_Data;

char receive\_complete = 0;

String buf\_string\_send1 = "";

String buf\_string\_send = "";

void setup() {

// put your setup code here, to run once:

pinMode(led\_Pin, OUTPUT);

pinMode(led1\_Pin, OUTPUT);

digitalWrite(led\_Pin, led\_on);

digitalWrite(led1\_Pin, led1\_off);

pinMode(sw\_Pin, INPUT\_PULLUP);

Serial3.begin(4800, SERIAL\_8N1);

Serial.begin(9600);

Serial.println("|------------------------------------------------------------------------|");

Serial.println("| EXERCISE 2:OVERSEES OPENING AND CLOSING OPERATIONS - ARDUINO MEGA 2560 |");

Serial.println("|------------------------------------------------------------------------|");

lcd.init();

lcd.backlight();

lcd.setCursor(0,0);

lcd.print("STATUS");

lcd.setCursor(7,0);

lcd.print("CLOSED");

lcd.setCursor(0,1);

lcd.print("OP");

lcd.setCursor(7,1);

lcd.print("CL");

}

void SW\_ISR()

{

count++;

count6 = count ;

buf\_string\_send ="@S" + String(count6) + '&';

Serial3.print(buf\_string\_send);

Serial.println(buf\_string\_send);

}

void SW\_ISR1()

{

count5++;

count6 = count5 ;

buf\_string\_send1 = "@s" + String(count6) + '&';

Serial3.print(buf\_string\_send1);

Serial.print(buf\_string\_send1);

}

void loop() {

delay(50);

// put your main code here, to run repeatedly:

int swStatus = digitalRead(sw\_Pin);

if (swStatus != last\_swStatus){

if (swStatus == LOW)

{

digitalWrite(led\_Pin,led\_off);

digitalWrite(led1\_Pin,led1\_on);

Serial.println(" >>> Door Opened");

Serial.println("|------------------------------------------------------------------------|");

lcd.init();

lcd.backlight();

lcd.setCursor(1,0);

lcd.print("STATUS");

lcd.setCursor(0,1);

lcd.print("OP");

lcd.setCursor(7,1);

lcd.print("CL");

lcd.setCursor(8,0);

lcd.print("OPENED");

lcd.setCursor(4,1);

lcd.print(count6);

lcd.setCursor(11,1);

lcd.print(count6);

}

else

{

digitalWrite(led\_Pin,led\_on);

digitalWrite(led1\_Pin,led1\_off);

Serial.println(" >>> Door Closed");

Serial.println("|------------------------------------------------------------------------|");

lcd.init();

lcd.backlight();

lcd.setCursor(1,0);

lcd.print("STATUS");

lcd.setCursor(0,1);

lcd.print("OP");

lcd.setCursor(7,1);

lcd.print("CL");

lcd.setCursor(8,0);

lcd.print("CLOSED");

lcd.setCursor(4,1);

lcd.print(count6);

lcd.setCursor(11,1);

lcd.print(count6+1);

}

}

last\_swStatus = swStatus;

led1\_state = digitalRead(led\_Pin);

led2\_state = digitalRead(led1\_Pin);

if (led1\_state != last\_led1\_state){

if(led1\_state == HIGH){

count1++;

count3 = count1 - 1;

SW\_ISR();

serialEvent3();

Serial.println(data\_change);

Serial.println("|------------------------------------------------------------------------|");

}

}

delay(50);

last\_led1\_state = led1\_state;

if (led2\_state != last\_led2\_state){

if(led2\_state == HIGH){

count2++;

SW\_ISR1();

serialEvent3();

Serial.println(data\_change1);

Serial.println("|------------------------------------------------------------------------|");

}

}

last\_led2\_state = led2\_state;

}

void serialEvent3()

{

ReceiveData = Serial3.read();

if(ReceiveData == start\_sign)

{

count\_ReceiveData = 0;

start\_Data = ReceiveData;

buf\_string\_receive[count\_ReceiveData] = ReceiveData;

}

else if (ReceiveData = start1\_sign)

{

count\_ReceiveData = 0;

start\_Data = ReceiveData;

buf\_string\_receive[count\_ReceiveData] = ReceiveData;

}

if(ReceiveData == end\_sign)

{

end\_Data = ReceiveData;

buf\_string\_receive[count\_ReceiveData] = ReceiveData;

}

else if (ReceiveData == end1\_sign)

{

end\_Data = ReceiveData;

buf\_string\_receive[count\_ReceiveData] = ReceiveData;

}

if((start\_Data == start\_sign) && (end\_Data == end\_sign))

{

receive\_complete = 1;

count\_ReceiveData = 0;

start\_Data = NULL;

end\_Data = NULL;

}

else if ((start\_Data == start1\_sign) && (end\_Data == end1\_sign))

{

receive\_complete = 1;

count\_ReceiveData = 0;

start\_Data = NULL;

end\_Data = NULL;

}

else

{

buf\_string\_receive[count\_ReceiveData] = ReceiveData;

count\_ReceiveData++;

if(count\_ReceiveData > max\_count\_ReceiveData)

{

count\_ReceiveData = 0;

}

}

receive\_complete = 0;

count\_ReceiveData = 0;

}

void Clear\_buf\_string\_receive(void){

for(char i = 0; i<= max\_count\_ReceiveData; i++){

buf\_string\_receive[i] = NULL;

}

}

**3.2.2 Code cho PIC**

#define on 1

#define off 0

#define max\_count\_ReceiveData 10

#define len\_code 5

unsigned char i;

char ReceiveData;

char buffer[15];

char buffer1[15];

char buffer2[15];

char buffer3[15];

int count =0;

int count1 =0;

int count2 = 0;

int count3 = 0;

bit oldstate;

char led\_on\_code[]= "le\_on";

char led\_off\_code[]= "le\_of";

char start\_sign ='@';

char end\_sign = '&';

char count\_ReceiveData = 0;

char buf\_string\_receive[max\_count\_ReceiveData];

char start\_Data, end\_Data;

char receive\_complete = 0;

// LCD //

sbit LCD\_RS at LATD1\_bit;

sbit LCD\_EN at LATD2\_bit;

sbit LCD\_D4 at LATD3\_bit;

sbit LCD\_D5 at LATD4\_bit;

sbit LCD\_D6 at LATD5\_bit;

sbit LCD\_D7 at LATD7\_bit;

sbit LCD\_RS\_Direction at TRISD1\_bit;

sbit LCD\_EN\_Direction at TRISD2\_bit;

sbit LCD\_D4\_Direction at TRISD3\_bit;

sbit LCD\_D5\_Direction at TRISD4\_bit;

sbit LCD\_D6\_Direction at TRISD5\_bit;

sbit LCD\_D7\_Direction at TRISD7\_bit;

void Clear\_buf\_string\_receive(void)

{

for (i=0;i<max\_count\_ReceiveData; i++)

{

buf\_string\_receive[i] = '\0';

}

}

void interrupt(void)

{

if(INTCON.INT0IF == 1)

{

INTCON.INT0IF = 0;

count++ ;

sprintf(buffer,"@s%d&", count);

sprintf(buffer2,"%d",count+1);

UART1\_Write\_Text(buffer);

}

while(INTCON.INT0IF == 0);

{

INTCON.INT0IF = 0;

count1++ ;

sprintf(buffer1,"@S%d&", count1+1);

sprintf(buffer3,"%d",count1+1);

UART1\_Write\_Text(buffer1);

}

if(PIR1.RCIF == 1)

{

PIR1.RCIF = 0;

ReceiveData = UART1\_Read();

if (ReceiveData == start\_sign)

{

count\_ReceiveData = 0;

start\_Data = ReceiveData;

buf\_string\_receive[count\_ReceiveData] = ReceiveData;

}

if (ReceiveData == end\_sign)

{

end\_Data = ReceiveData;

buf\_string\_receive[count\_ReceiveData] = ReceiveData;

}

if ((start\_Data == start\_sign) && (end\_Data == end\_sign))

{

receive\_complete = 1;

count\_ReceiveData = 0;

start\_Data = '\0';

end\_Data = '\0';

}

else

{

buf\_string\_receive[count\_ReceiveData] = ReceiveData;

count\_ReceiveData++;

if(count\_ReceiveData > max\_count\_ReceiveData)

{

count\_ReceiveData = 0;

}

}

receive\_complete = 0;

count\_ReceiveData = 0;

Clear\_buf\_string\_receive();

}

}

void main()

{

Lcd\_Init();

Lcd\_Cmd(\_Lcd\_CLEAR);

Lcd\_Cmd(\_LCD\_CURSOR\_OFF);

Lcd\_Out(2,12,"1");

Lcd\_Out(2,5,"0");

ADCON1 |= 0x0F;

CMCON |= 7;

PORTB= 0x00;

LATB = 0x00;

TRISB.TRISB0 = 1;

PORTE = 0x00;

LATE = 0x00;

TRISE.TRISE0 = off;

PORTD = 0x00;

LATD = 0x00;

TRISD.TRISD0 = off;

INTCON.INT0IF = 0;

INTCON.INT0IE = 1;

INTCON2.INTEDG0 =1;

PIR1.RCIF = 0;

PIE1.RCIE = 1;

INTCON.GIE =1;

INTCON.PEIE = 1;

UART1\_Init(4800);

while(1)

{

Lcd\_Out(1,1, "STATUS");

Lcd\_Out(2,2, "OP");

Lcd\_Out(2,9, "CL");

if(PORTB.B0 == 1 )

{

LATE.B0 = 1;

LATD.B0 = 0;

Lcd\_Out(1,8,"CLOSED");

Lcd\_out(2,12,buffer3);

}

else

{

LATD.B0 = 1;

LATE.B0 = 0;

Lcd\_Out(1,8,"OPENED");

Lcd\_out(2,5,buffer2);

}

}

}

***3.3. Lập trình giao diện VS C#***

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

using System.IO.Ports;

using System.Xml;

namespace GTBNV\_EX2\_DESO1\_NguyenThanhNguyen\_18018791

{

public partial class Form1 : Form

{

string ReceiveData = String.Empty;

string TransmitData = String.Empty;

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

Serial\_Port.PortName = "Select COM Port...";

Serial\_Port.BaudRate = 4800;

Serial\_Port.DataBits = 8;

Serial\_Port.Parity = Parity.None;

Serial\_Port.StopBits = StopBits.One;

string[] ports = SerialPort.GetPortNames();

foreach (string port in ports)

{

comboBox.Items.Add(port);

}

}

private void Form1\_FormClosed(object sender, FormClosedEventArgs e)

{

if (Serial\_Port.IsOpen)

Serial\_Port.Close();

}

private void button1\_Click(object sender, EventArgs e)

{

if (comboBox.Text == "")

MessageBox.Show("Select COM Port. ", "Warning", MessageBoxButtons.OK, MessageBoxIcon.Warning);

else if (comboBox2.Text == "")

MessageBox.Show("Select baudrate for COM Port. ", "Warning", MessageBoxButtons.OK, MessageBoxIcon.Warning);

else

{

try

{

if (Serial\_Port.IsOpen)

{

MessageBox.Show("COM Port is connected and ready for use.", " Information", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

else

{

Serial\_Port.Open();

MessageBox.Show(comboBox.Text + " is connected ", "Information", MessageBoxButtons.OK, MessageBoxIcon.Information);

textBox4.BackColor = Color.Lime;

textBox4.Text = "Connecting...";

comboBox.Enabled = false;

comboBox2.Enabled = false;

ReceiveData = String.Empty;

TransmitData = String.Empty;

textBox3.Text = "Closed";

ovalShape1.BackColor = Color.Green;

textBox2.Text = "1";

}

}

catch (Exception)

{

textBox4.BackColor = Color.Red;

textBox4.Text = " Disconnected ";

MessageBox.Show("COM Port is not found. Please check your COM or Cable.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

private void button2\_Click(object sender, EventArgs e)

{

try

{

if (Serial\_Port.IsOpen)

{

Serial\_Port.Close();

textBox4.BackColor = Color.Red;

textBox4.Text = "Disconnected";

MessageBox.Show("COM port is disconnected", "Information", MessageBoxButtons.OK, MessageBoxIcon.Information);

comboBox.Enabled = true;

comboBox2.Enabled = true;

textBox3.Text = "";

ovalShape1.BackColor = Color.DarkGray;

textBox1.Text = "0";

textBox2.Text = "0";

ovalShape2.BackColor = Color.DarkGray;

}

else

{

MessageBox.Show("COM port is disconnected. Please reconnect to use", "Information", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

}

catch (Exception)

{

MessageBox.Show("Disconnection appear error.Unable to disconnect.", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

private void button3\_Click(object sender, EventArgs e)

{

DialogResult answer = MessageBox.Show("Do you want to exit the program?", "Question", MessageBoxButtons.YesNo, MessageBoxIcon.Question);

if (answer == DialogResult.Yes)

{

if (Serial\_Port.IsOpen)

{

Serial\_Port.Close();

}

this.Close();

}

}

private void comboBox\_SelectedIndexChanged(object sender, EventArgs e)

{

Serial\_Port.Close();

textBox4.BackColor = Color.Red;

textBox4.Text = "Disconnected";

Serial\_Port.PortName = comboBox.Text;

}

private void comboBox2\_SelectedIndexChanged(object sender, EventArgs e)

{

Serial\_Port.Close();

textBox4.BackColor = Color.Red;

textBox4.Text = "Disconnected";

Serial\_Port.BaudRate = Convert.ToInt32(comboBox2.Text);

}

private void Serial\_Port\_DataReceived(object sender, SerialDataReceivedEventArgs e)

{

CheckForIllegalCrossThreadCalls = false;

ReceiveData = Serial\_Port.ReadTo("&");

if (ReceiveData.Substring(0, 1) == "@")

{

if (ReceiveData.Substring(1, 1) == "S")

{

textBox2.Text = ReceiveData.Substring(2);

ReceiveData = String.Empty;

ovalShape1.BackColor = Color.Green;

ovalShape2.BackColor = Color.DarkGray;

textBox3.Text = "Closed";

}

else if (ReceiveData.Substring(1, 1) == "s")

{

textBox1.Text = ReceiveData.Substring(2);

ReceiveData = String.Empty;

ovalShape2.BackColor = Color.Red;

ovalShape1.BackColor = Color.DarkGray;

textBox3.Text = "Openned";

}

}

}

private void textBox2\_TextChanged(object sender, EventArgs e)

{

}

}

}

***3.4. Biên dịch code và sửa lỗi.***

Ảnh có chứa văn bản

Mô tả được tạo tự động

Ảnh có chứa văn bản

Mô tả được tạo tự động

Ảnh có chứa văn bản, ảnh chụp màn hình, màn hình, trong nhà

Mô tả được tạo tự động

**PHẦN IV: Chuẩn đầu ra môn học 4: Có khả năng lắp ráp, cân chỉnh các board mạch điện tử theo qui trình**

***4.1. Quy trình Lắp ráp và Test thực nghiệm***

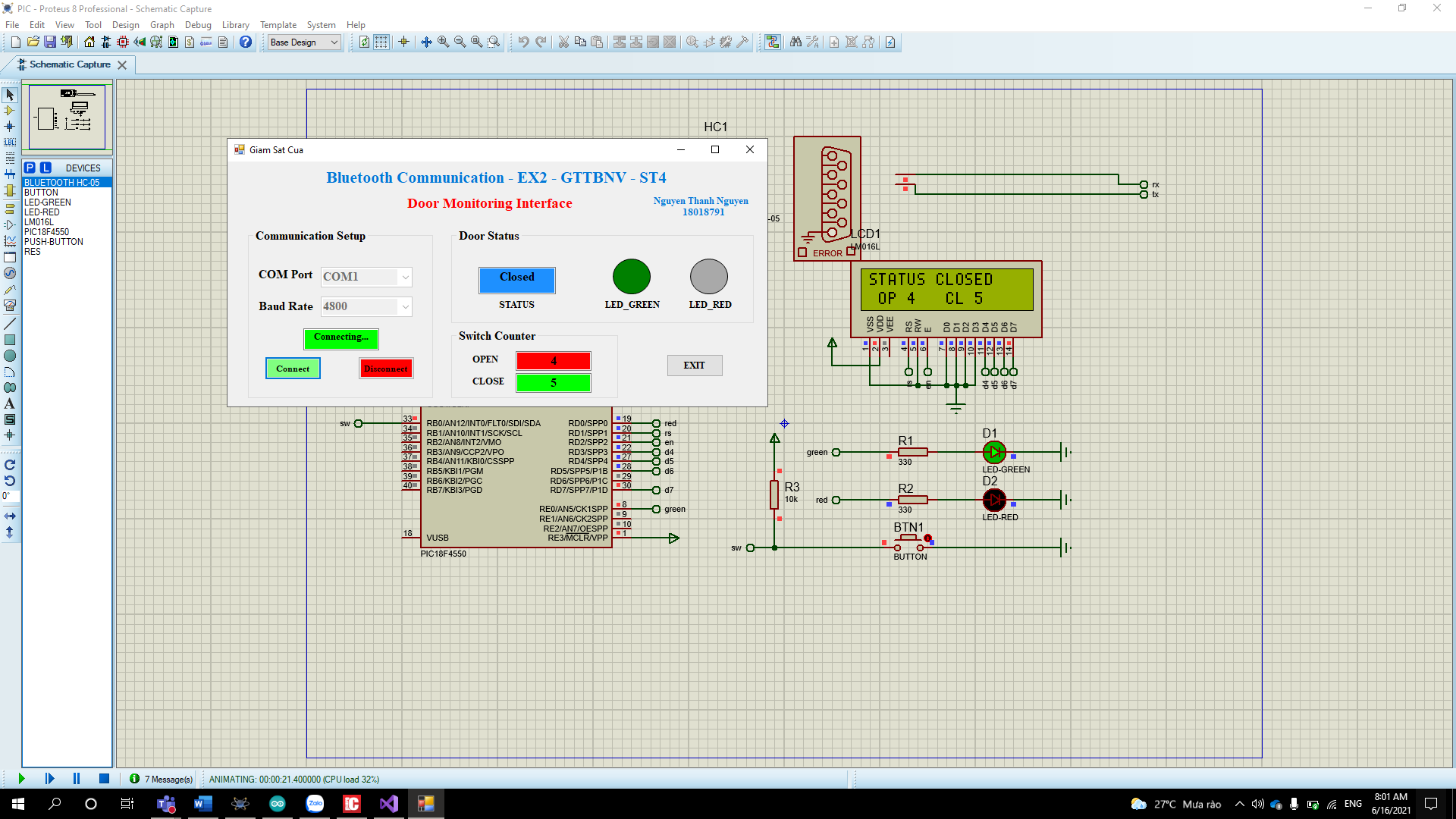
**Bước 1**: Test code LCD.

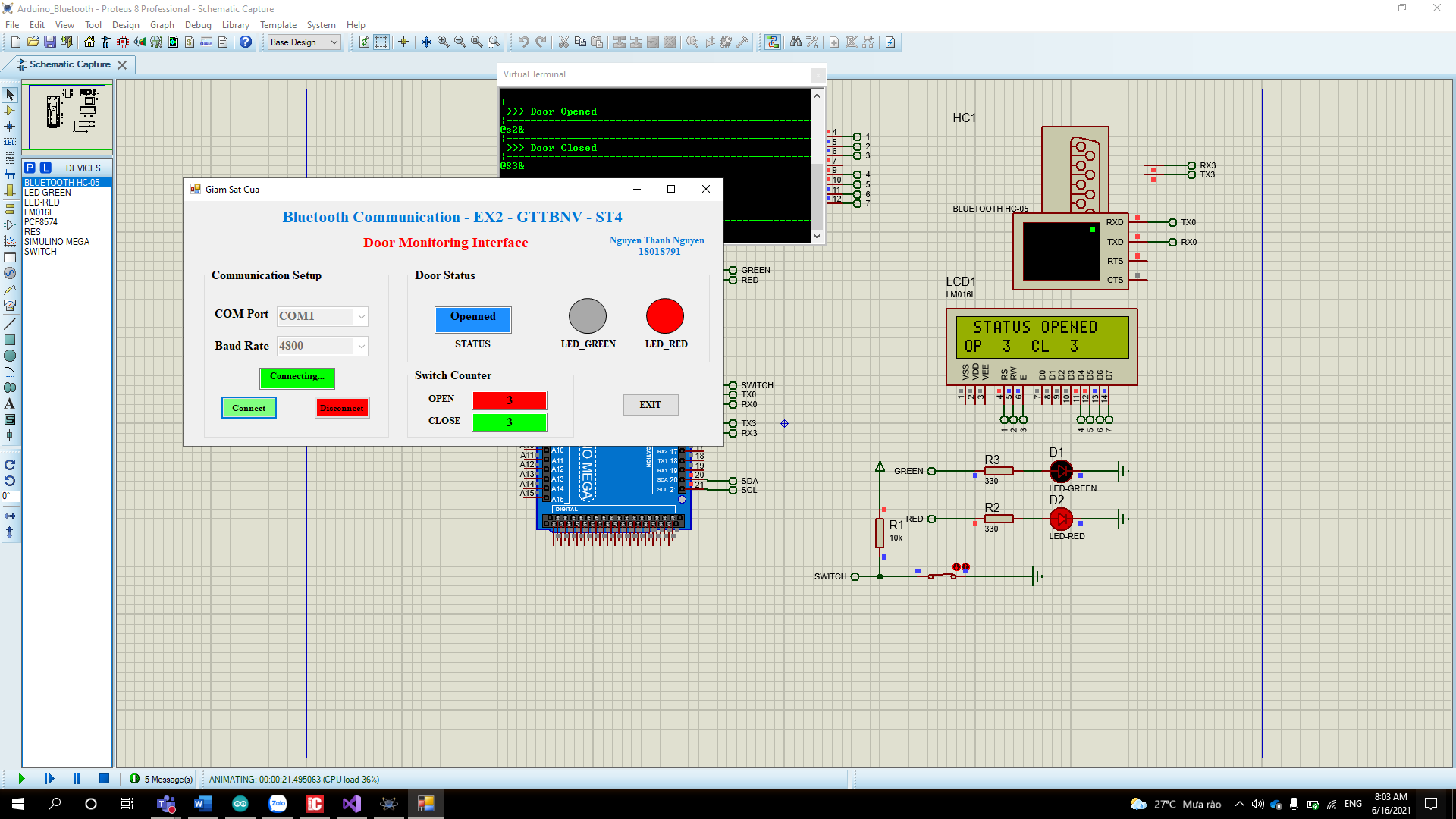
**Bước 2**: Test code UART qua COM Bluetooth.

**Bước 3**: Test điều khiển thủ công đồng thời xuất ra LCD.  
**Bước 4**: Test điều khiển thủ công đồng thời xuất ra LCD + nhận và xuất qua COM Bluetooth.

**Bước 5**: Test code toàn bộ đề tài giao tiếp với Visual Studio.

***4.2. Lắp ráp mạch thực tế***





***4.3. Khả năng cân chỉnh bo mạch***

**PHẦN V: Kết luận bài thực hành**

Kết quả code thu được hoàn toàn phù hợp với kịch bản đã lập trình và không phát sinh lỗi ngoài ý muốn.