Directions to install the ESP32 board on Arduino IDE

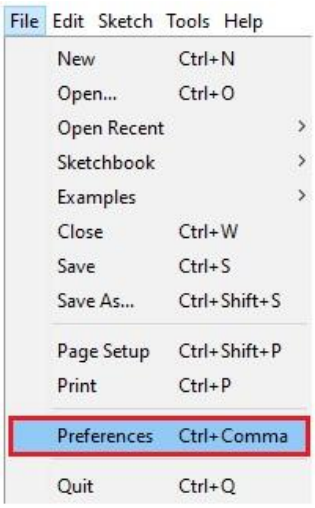
Installing the ESP32 Board in Arduino IDE

Before starting this installation method, make sure you have the latest version of the Arduino IDE 1.8.19 installed in your computer.

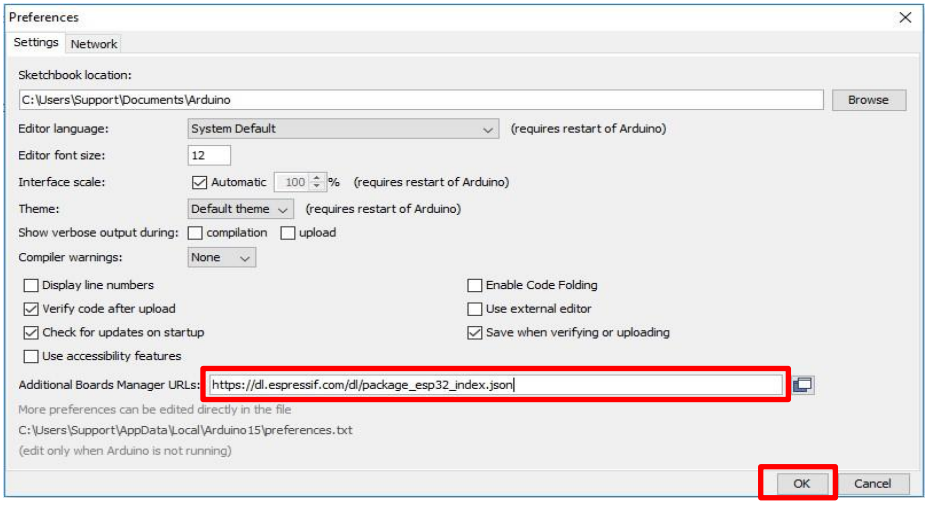
If you don't, install it from https://www.arduino.cc/en/software , continue with this tutorial.

To install the ESP32 board in your Arduino IDE, follow these below instructions:

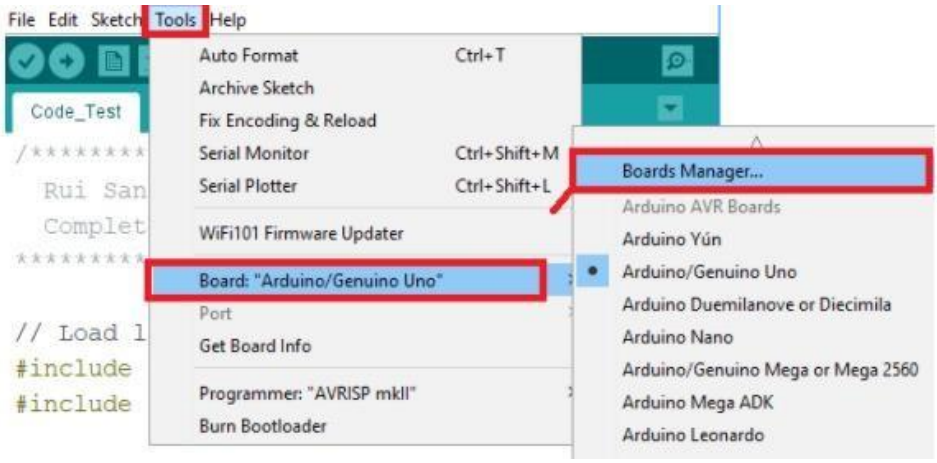
Step 1: In your Arduino IDE, go to **File**> **Preferences**



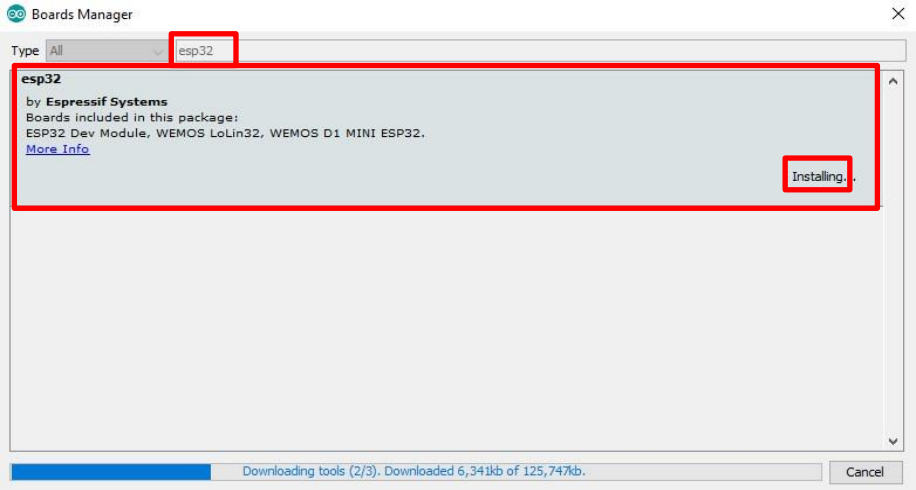
Step 2: Enter **https://dl.espressif.com/dl/package_esp32_index.json** into the "Additional Board Manager URLs" field as shown in the figure below. Then, click the "OK" button:



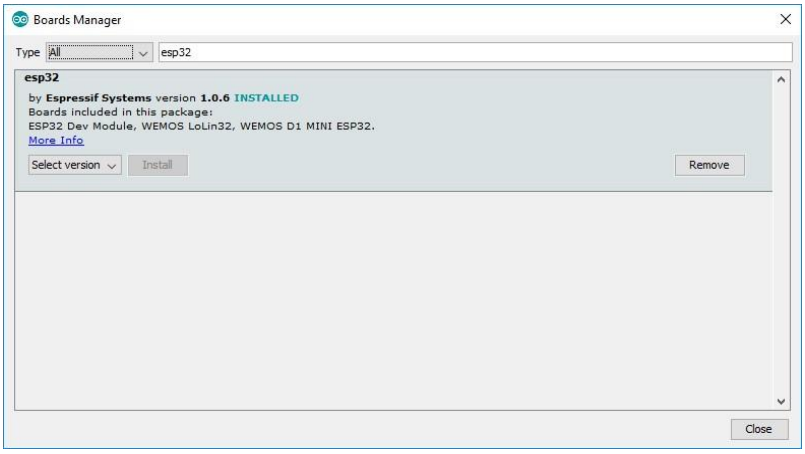
Step 3: Open the Boards Manager. Go to **Tools** > **Board** > **Boards Manager…**



Step 4: Search for **ESP32** and press install button for the "**ESP32 by Espressif Systems**":



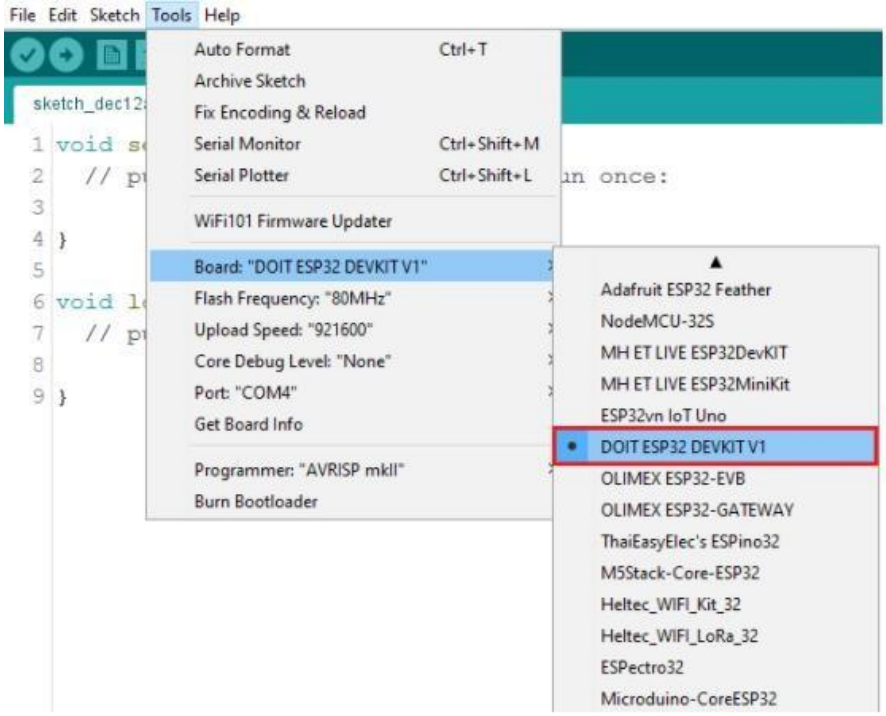
Step 5: ESP32 will be installed after a few seconds.



Testing the Installation

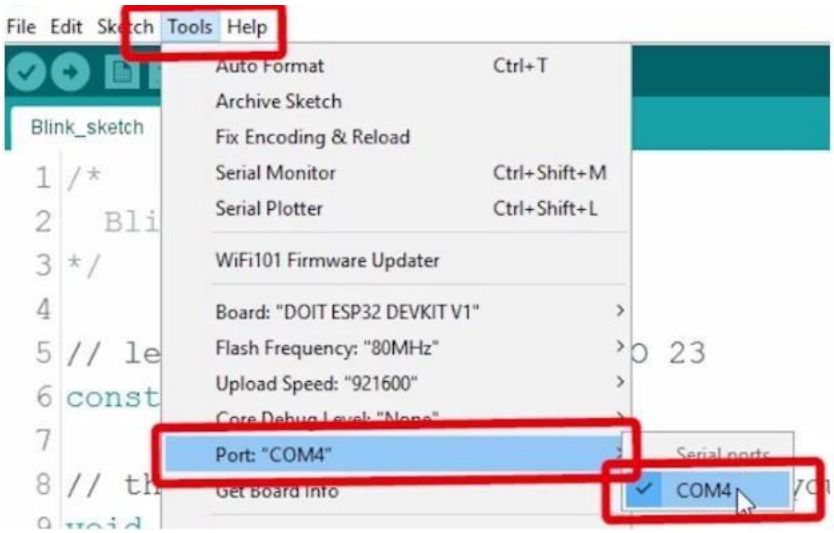
Step 1: Plug the ESP32 Trainer Kit to your computer. With your Arduino IDE open, follow these steps:

Step 2 : Select your Board in **Tools** > **Board** menu (it's the **DOIT ESP32 DEVKIT V1**)

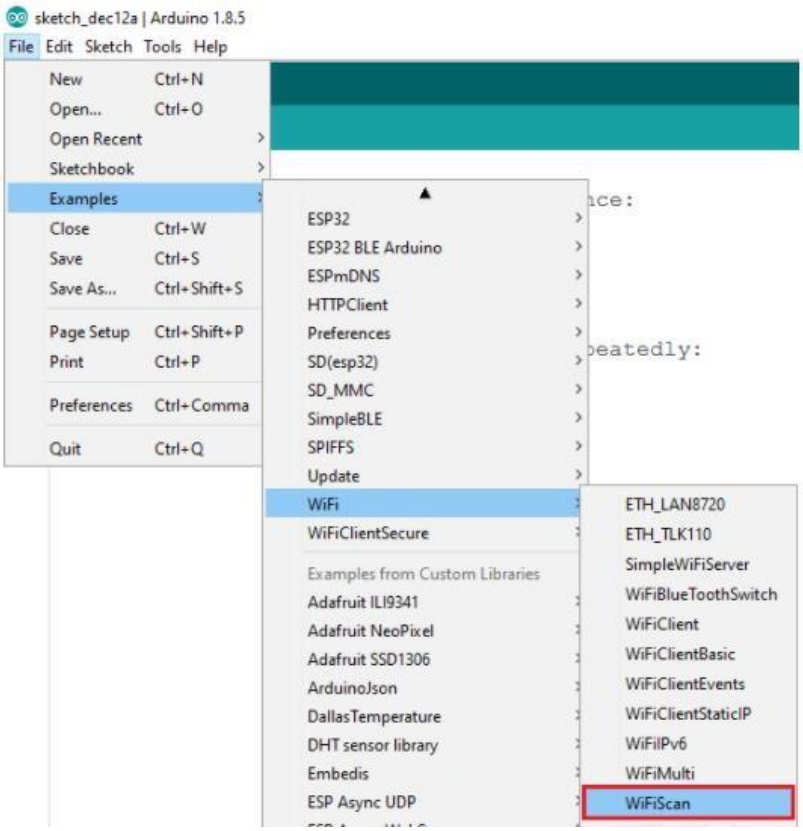


Select the Port (if you don't see the COM Port in your Arduino IDE, you need to install the FTDI Drivers: https://ftdichip.com/drivers/d2xx-drivers/

For Installation Guide**,** CLICK HERE



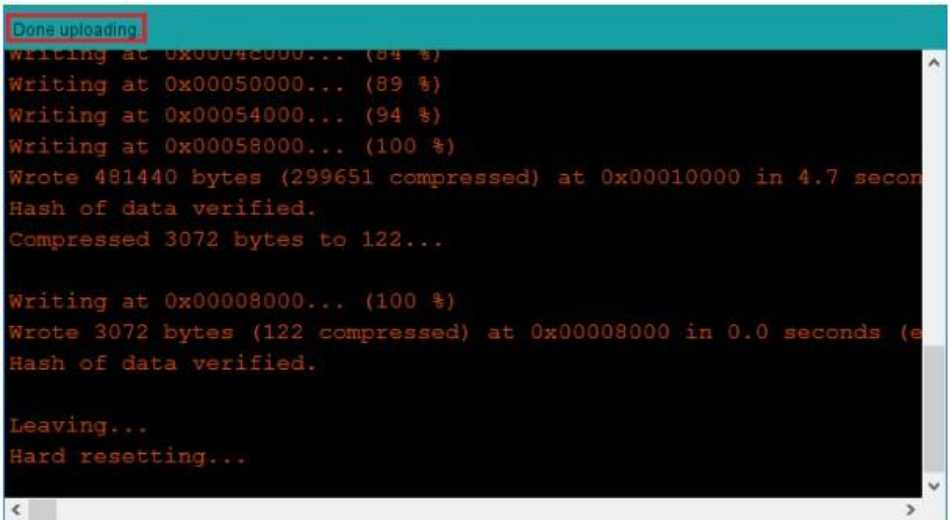
Step 3: Open the following example under **File** > **Examples** > **WiFi (ESP32)** > **WiFiScan**



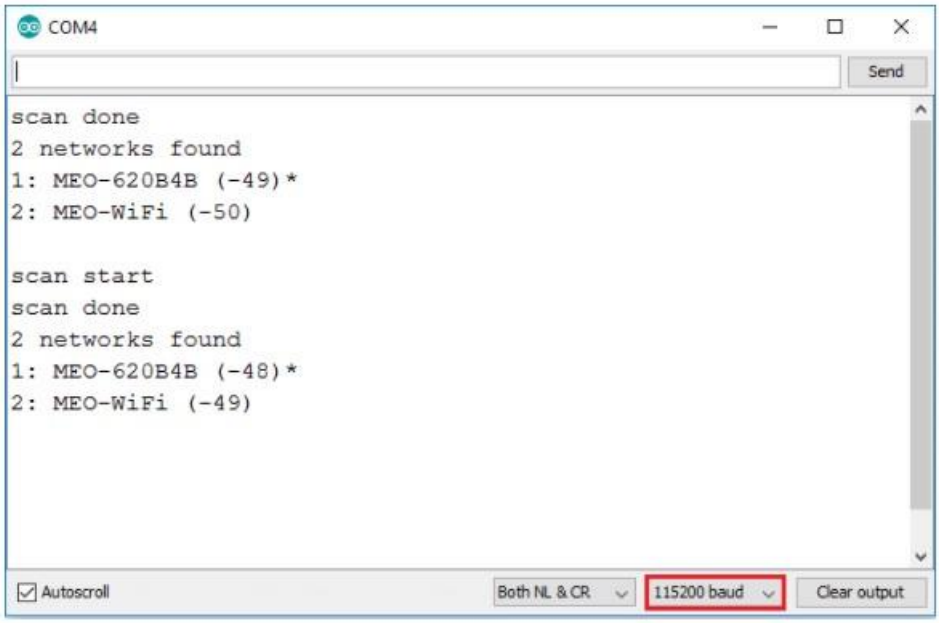
Step 4: A new sketch opens in your Arduino IDE:

Step 5: Press the **Upload** button in the Arduino IDE. Wait a few seconds while the code compiles and uploads to your board.

If everything went as expected, you should see a "**Done uploading.**" message.

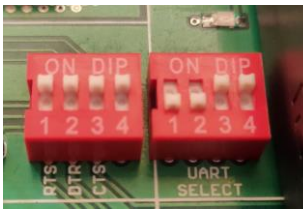


Step 6: Open the Arduino IDE Serial Monitor at a baud rate of 115200:
Step 7: Press the ESP32 on-board **Reset** button and you should see the networks available near your ESP32:



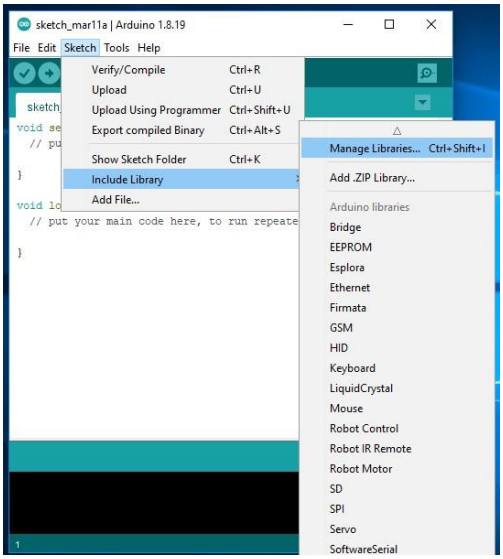
NOTE: Before uploading any codes to the Biomedical Interfacing Kit, make settings.

1. Make UART SELECT 3, 4 as HIGH and 1, 2 as LOW.
2. Make RTS, DTR, CTS as HIGH.
3. To use RS232, make the UART SELECT 1, 2 as HIGH and 3, 4 as LOW.

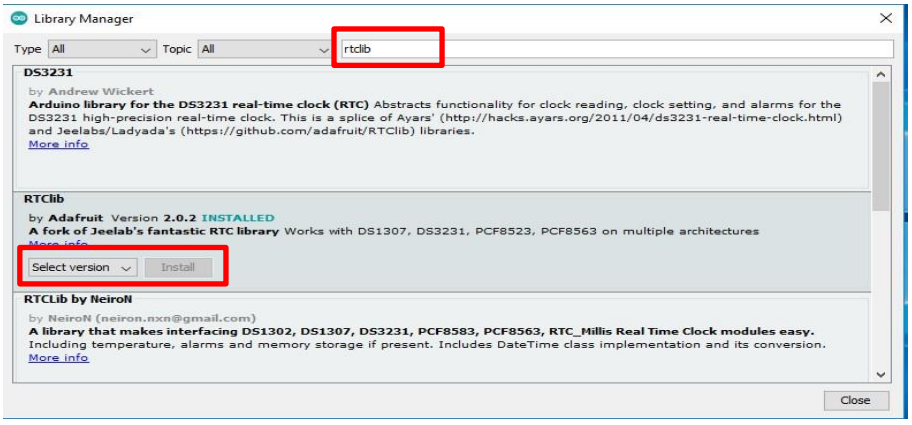


Installing Libraries

Step 1: To install a new library into your Arduino IDE. Open the IDE and click to the "Sketch" menu and then Include Library → Manage Libraries.



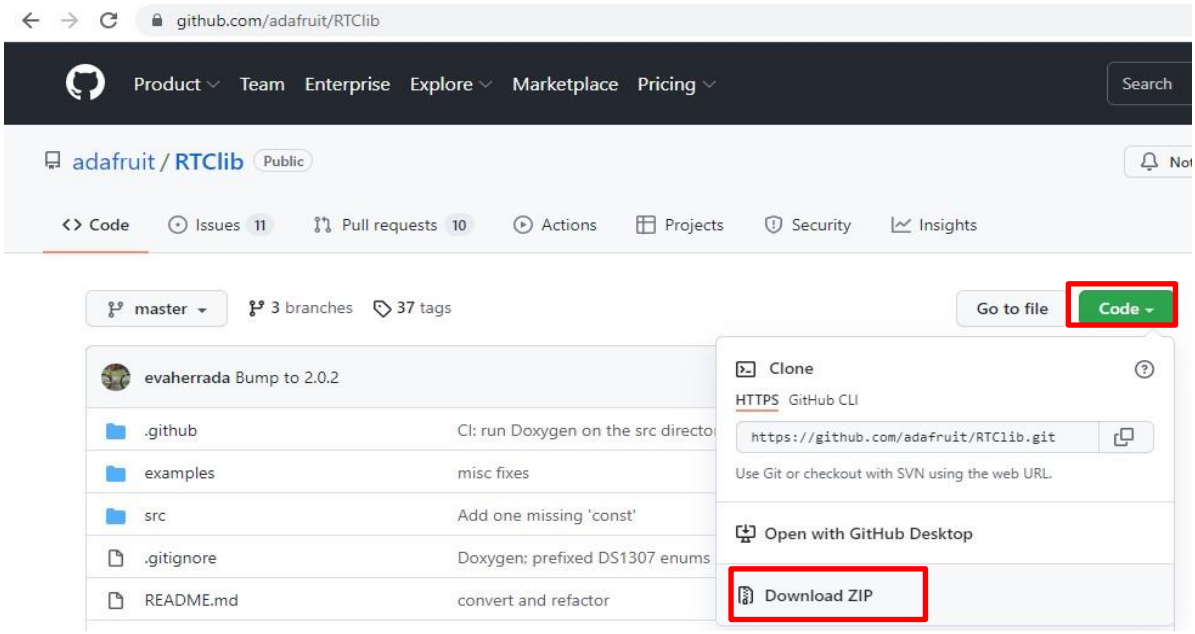
Step 2: Then the Library Manager will open and you will find a list of libraries that are already installed or ready for installation. In this example we will install the RTC library(i.e rtclib).Enter the library name to find it, click on it, then select the version of the library you want to install. Sometimes only one version of the library is available.Then click on install.If you don't find the library then refer page 12**.**



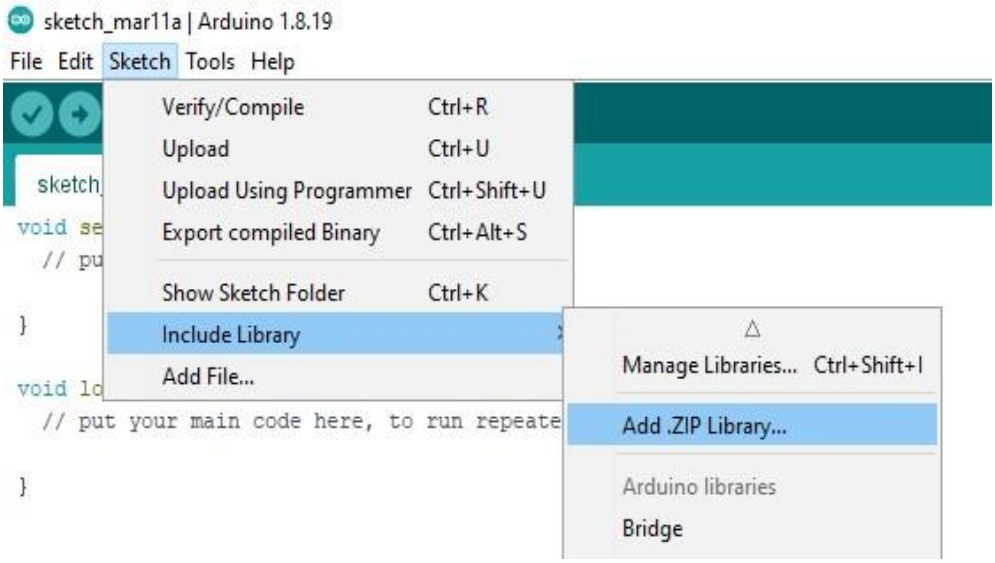
Step 3: Wait for the IDE to install the new library. Downloading may take time depending on your connection speed. Once it has finished, an *Installed* tag should appear next to the RTC library. Then click on close.

Another Method for installing and importing a .zip Library

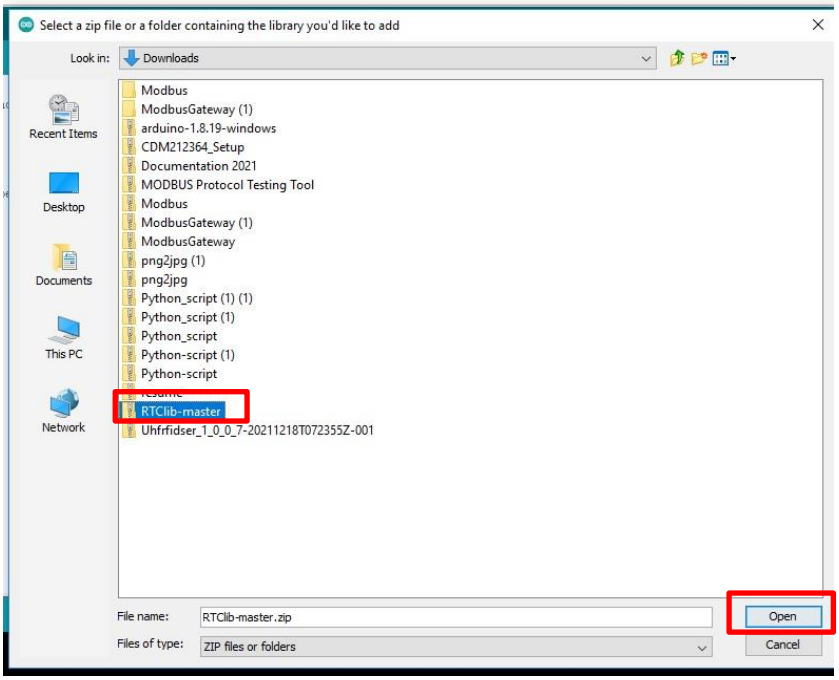
Step 1 : Go to Google, search for the library(i.e rtclib) you want to install, click on download ZIP.



Step 2: In the Arduino IDE, go to Sketch → Include Library → Add .ZIP Library.



Step 3: Select the library you would like to add. Go to the .zip file's downloaded location and open it.



Libraries Link

1.	Keypad Library	(Keypad.h)	Download
2.	SD Card Libraries		
		(FS.h)	Download
		(SD.h)	Download
		(SPI.h)	Download
3.	OLED Display	(Adafruit_GFX.h)	Download
		(Adafruit_SSD1327.h)	Download
4.	RTC Libraries	(RTClib.h)	Download
5.	SPO2 Libraries	(MAX30100_PulseOximeter.h)	Download
6.	Body Temperature Libraries	(Adafruit_MLX90614.h)	Download

EXPERIMENT NO 1

Blinking an LED

Aim:

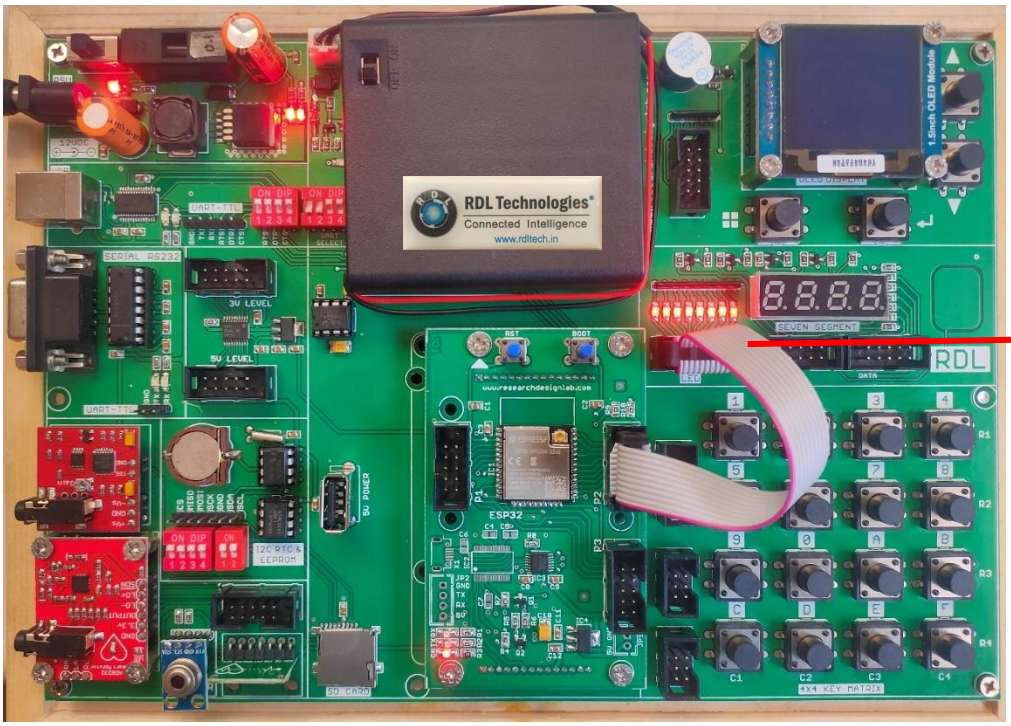
Interfacing LED's with Biomedical Interfacing Kit.

Description:

To learn how to programme an Biomedical Interfacing Kit to blink an LED by connecting an LED'S to its digital pins.

Hardware Requirement:

Biomedical Interfacing Kit and FRC cable.



Procedure:

1. Connect P2 port and LED port using FRC cable as shown above fig.
2. Connect the USB cable to the board.
3. Open Arduino IDE. Select DOIT ESP32 DEVKIT V1 in boards and select COM port.
4. Now Write the program, verify and Upload it.

5. Now you can see the LED blink on the Biomedical Interfacing Kit.

Program:

```
const int L1=23, L2=22, L3=21, L4=19, L5=18, L6=5, L7=17,
L8=16; //initializing LED pins
void setup()
{Serial.begin(115200);
pinMode(L1, OUTPUT); // Set all Port P2 pins as output
pinMode(L2, OUTPUT);
pinMode(L3, OUTPUT);
pinMode(L4, OUTPUT);
pinMode(L5, OUTPUT);
pinMode(L6, OUTPUT);
pinMode(L7, OUTPUT);
pinMode(L8, OUTPUT);
}

void loop()
{
digitalWrite(L1, HIGH);
digitalWrite(L2, HIGH);
digitalWrite(L3, HIGH);
digitalWrite(L4, HIGH);
digitalWrite(L5, HIGH);
digitalWrite(L6, HIGH);
digitalWrite(L7, HIGH);
digitalWrite(L8, HIGH);
delay(1000);
digitalWrite(L1, LOW);
digitalWrite(L2, LOW);
digitalWrite(L3, LOW);
digitalWrite(L4, LOW);
digitalWrite(L5, LOW);
digitalWrite(L6, LOW);
digitalWrite(L7, LOW);
digitalWrite(L8, LOW);
delay(1000);

}
```

EXPERIMENT NO 2

Seven Segment Displays

Aim:

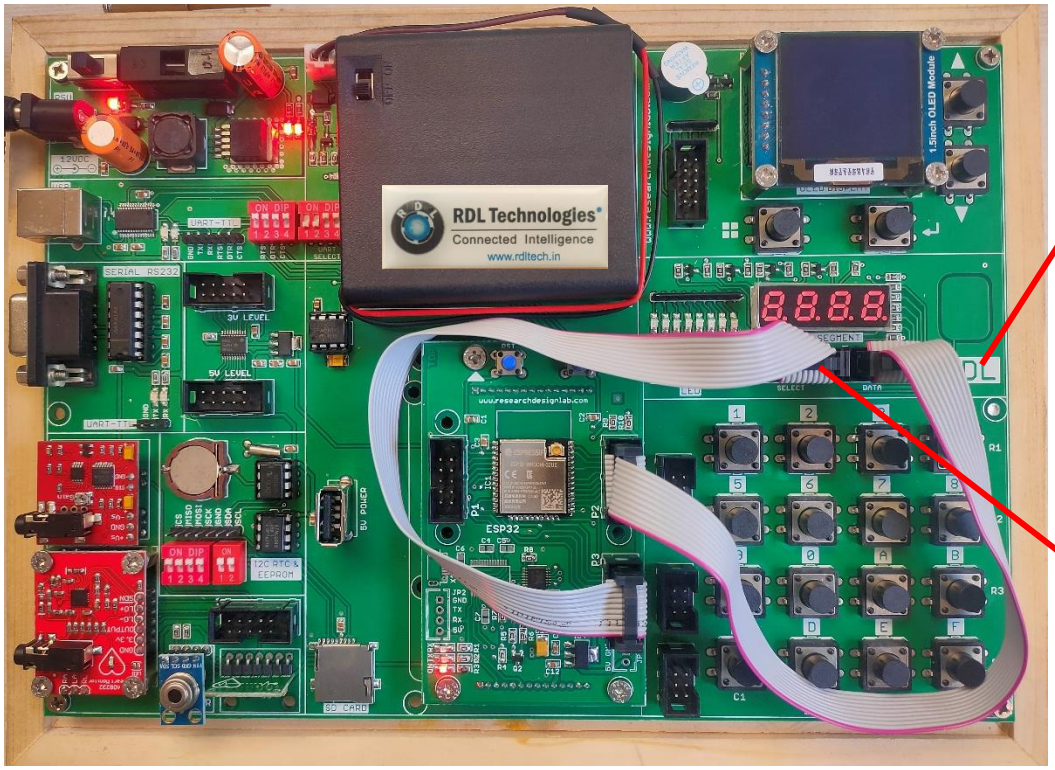
Interfacing Seven Segment Display with Biomedical Interfacing Kit.

Description:

To display numbers in Seven Segment Display.

Hardware Requirement:

Biomedical Interfacing Kit and FRC cable.



Connect P2 to DATA port.

Connect P3 to SELECT port.

Procedure:

1. Connect P2 port with DATA port and connect P3 port with SELECT port using FRC cable as shown above.
2. Connect the USB cable to the board.
3. Open Arduino IDE .Select DOIT ESP32 DEVKIT V1 in boards and select COM port.
4. Now write the program, verify and Upload it.

5. Now you can see that number starts displaying on the seven segments on the Biomedical Interfacing Kit.

Program:

```
const int sel1=25, sel2=26, sel3=4, sel4=2; //initializing
selection pins -Port P3
const int a=16 ,b=17, c=5, d=18, e=19, f=21, g=22, dp=23;
//initializing data pins -Port P2
void setup()
{
pinMode(sel1,OUTPUT); //declaring Selection Pins as output
pinMode(sel2,OUTPUT);
pinMode(sel3,OUTPUT);
pinMode(sel4,OUTPUT);
digitalWrite(sel1,LOW); //selecting all 4 digits of 7-Segment
display by making it LOW
digitalWrite(sel2,LOW);
digitalWrite(sel3,LOW);
digitalWrite(sel4,LOW);
pinMode(a,OUTPUT); //declaring data pins as output
pinMode(b,OUTPUT);
pinMode(c,OUTPUT);
pinMode(d,OUTPUT);
pinMode(e,OUTPUT);
pinMode(f,OUTPUT);
pinMode(g,OUTPUT);
pinMode(dp,OUTPUT);
delay(100);
}
void loop()
{
 // print 0
 digitalWrite(a,LOW);
 digitalWrite(b,LOW);
 digitalWrite(c,LOW);
 digitalWrite(d,LOW);
 digitalWrite(e,LOW);
 digitalWrite(f,LOW);
 digitalWrite(g,HIGH);
 digitalWrite(dp,LOW);
 delay(2000);
// print 1
 digitalWrite(a,HIGH);
 digitalWrite(b,LOW);
 digitalWrite(c,LOW);
 digitalWrite(d,HIGH);
 digitalWrite(e,HIGH);
 digitalWrite(f,HIGH);
 digitalWrite(g,HIGH);
 digitalWrite(dp,HIGH);
```

```
 delay(2000);


// print 2
 digitalWrite(a,LOW);
 digitalWrite(b,LOW);
 digitalWrite(c,HIGH);
 digitalWrite(d,LOW);
 digitalWrite(e,LOW);
 digitalWrite(f,HIGH);
 digitalWrite(g,LOW);
 digitalWrite(dp,LOW);
 delay(2000);

// print 3
 digitalWrite(a,LOW);
 digitalWrite(b,LOW);
 digitalWrite(c,LOW);
 digitalWrite(d,LOW);
 digitalWrite(e,HIGH);
 digitalWrite(f,HIGH);
 digitalWrite(g,LOW);
 digitalWrite(dp,LOW);
 delay(2000);

// print 4
 digitalWrite(a,HIGH);
 digitalWrite(b,LOW);
 digitalWrite(c,LOW);
 digitalWrite(d,HIGH);
 digitalWrite(e,HIGH);
 digitalWrite(f,LOW);
 digitalWrite(g,LOW);
 digitalWrite(dp,LOW);
 delay(2000);
// print 5
 digitalWrite(a,LOW);
 digitalWrite(b,HIGH);
 digitalWrite(c,LOW);
 digitalWrite(d,LOW);
 digitalWrite(e,HIGH);
 digitalWrite(f,LOW);
 digitalWrite(g,LOW);
 digitalWrite(dp,LOW);
 delay(2000);
// print 6
 digitalWrite(a,LOW);
 digitalWrite(b,HIGH);
 digitalWrite(c,LOW);
 digitalWrite(d,LOW);
 digitalWrite(e,LOW);
```

```
 digitalWrite(f,LOW);
 digitalWrite(g,LOW);
 digitalWrite(dp,LOW);
 delay(2000);
// print 7
 digitalWrite(a,LOW);
 digitalWrite(b,LOW);
 digitalWrite(c,LOW);
 digitalWrite(d,HIGH);
 digitalWrite(e,HIGH);
 digitalWrite(f,HIGH);
 digitalWrite(g,HIGH);
 digitalWrite(dp,HIGH);
 delay(2000);

// print 8
 digitalWrite(a,LOW);
 digitalWrite(b,LOW);
 digitalWrite(c,LOW);
 digitalWrite(d,LOW);
 digitalWrite(e,LOW);
 digitalWrite(f,LOW);
 digitalWrite(g,LOW);
 digitalWrite(dp,LOW);
 delay(2000);
 // print 9
 digitalWrite(a,LOW);
 digitalWrite(b,LOW);
 digitalWrite(c,LOW);
 digitalWrite(d,LOW);
 digitalWrite(e,HIGH);
 digitalWrite(f,LOW);
 digitalWrite(g,LOW);
 digitalWrite(dp,LOW);
 delay(2000);
}
```

EXPERIMENT NO 3

Hex Keypad

Aim:

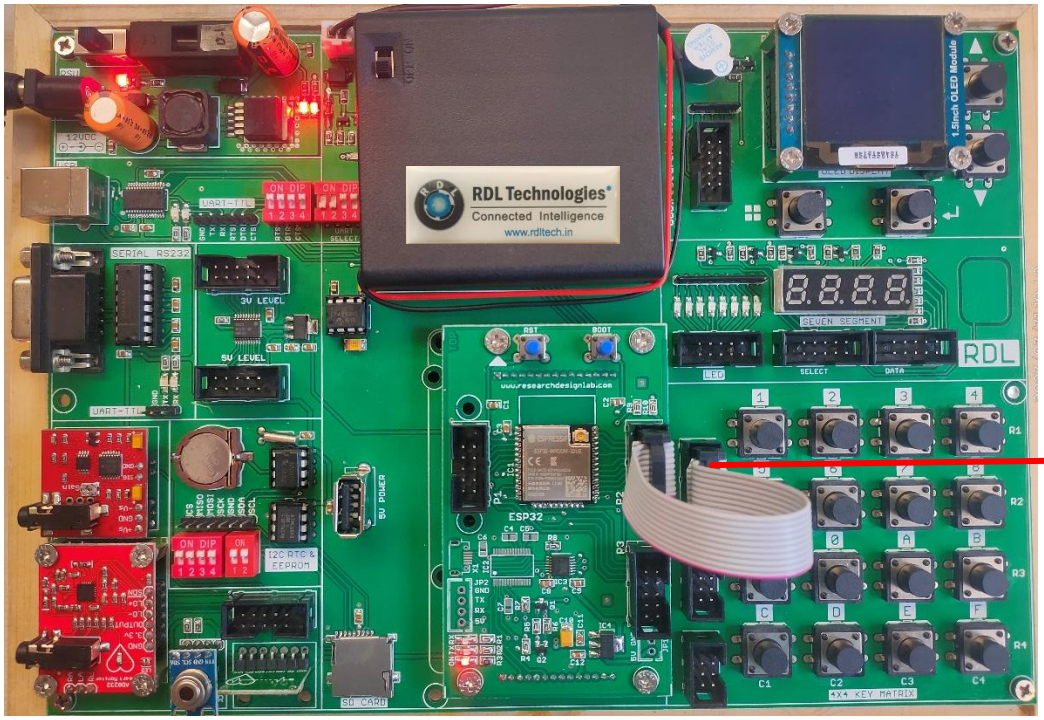
To interface 4x4 Hex Keypad with Biomedical Interfacing Kit.

Description:

To display the pressed key on the serial monitor.

Hardware Required:

Biomedical Interfacing Kit and FRC Cable.



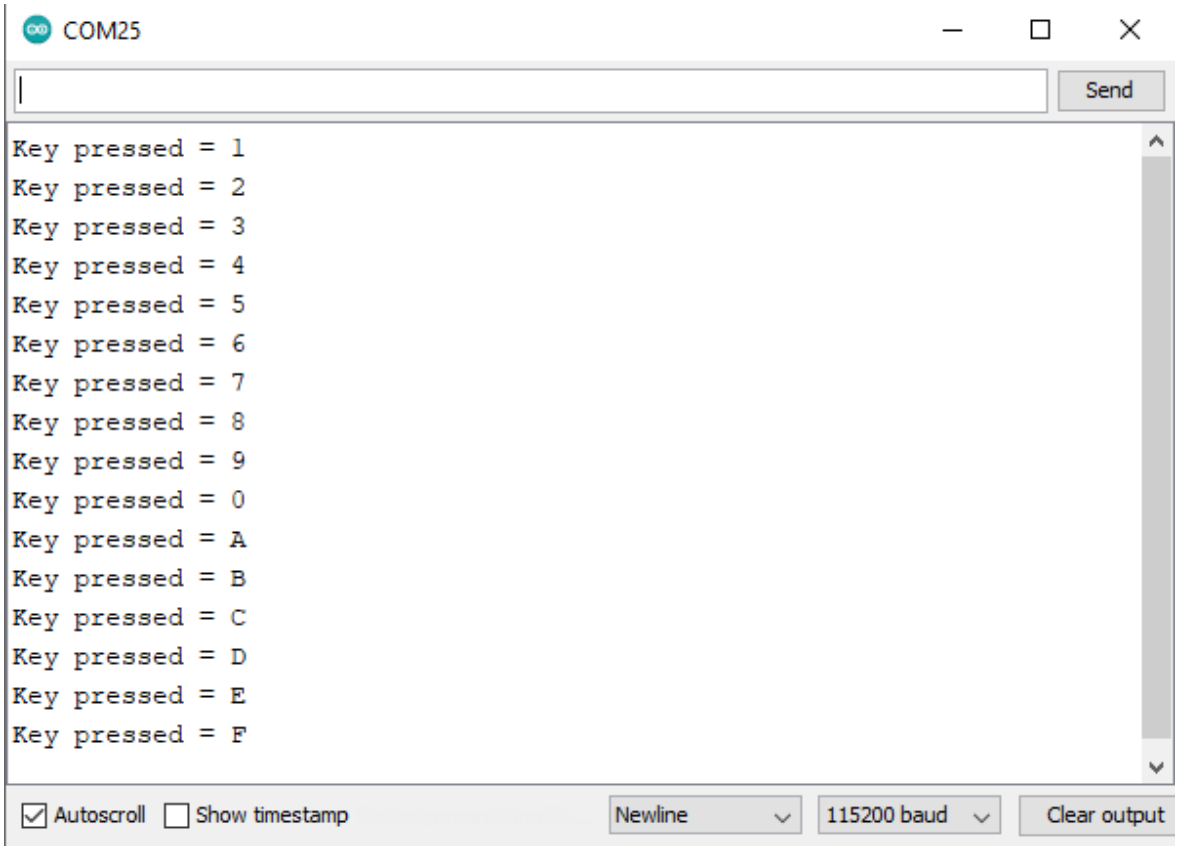
Procedure:

1. Connect P2 port and Keypad port using FRC cable as shown above.
2. Connect the USB cable to the board.
3. Open Arduino IDE .Select DOIT ESP32 DEVKIT V1 in boards and select COM port.
4. Now Write the program, verify and Upload it.
5. After uploading is done open serial monitor to observe the output.
6. On your serial monitor, the number appears for each switch pressed.

Program:

```
#include <Keypad.h>
char keys[4][4]={
{'1','2','3','4'},{'5','6','7','8'},{'9','0','A','B'},{'C','D'
,'E','F'}}; //defining characters of 4X4 Key Matrix
byte rowPin[4]={16,17,5,18}; //declaring the rows and column
pins (Port P2)
byte colPin[4]={19,21,22,23};
//byte rowPin[4]={34,35,32,33}; //declaring the rows and
column pins (Port P2)
//byte colPin[4]={36,39,25,26};
Keypad keypad=Keypad(makeKeymap(keys),rowPin,colPin,4,4); //
Creating 4X4 Keypad Matrix
void setup()
{
Serial.begin(115200);
  pinMode(16, INPUT);
  pinMode(17, INPUT);
  pinMode(5, INPUT);
  pinMode(18, INPUT);
  pinMode(19, INPUT);
  pinMode(21, INPUT);
  pinMode(23, INPUT);
  pinMode(22, INPUT);
}
void loop()
{
char pressed=keypad.getKey(); //This function will fetch the
character being pressed
if(pressed)
{
Serial.print("Key pressed = ");
Serial.println(pressed);
}
delay(500);
}
```

Output:


COM25
Send
Key pressed = 1
Key pressed = 2
Key pressed = 3
Key pressed = 4
Key pressed = 5
Key pressed = 6
Key pressed = 7
Key pressed = 8
Key pressed = 9
Key pressed = 0
Key pressed = A
Key pressed = B
Key pressed = C
Key pressed = D
Key pressed = E
Key pressed = F
Autoscroll
Show timestamp
Newline
115200 baud
Clear output

EXPERIMENT NO 4

RTC (Real Time Clock)

Aim:

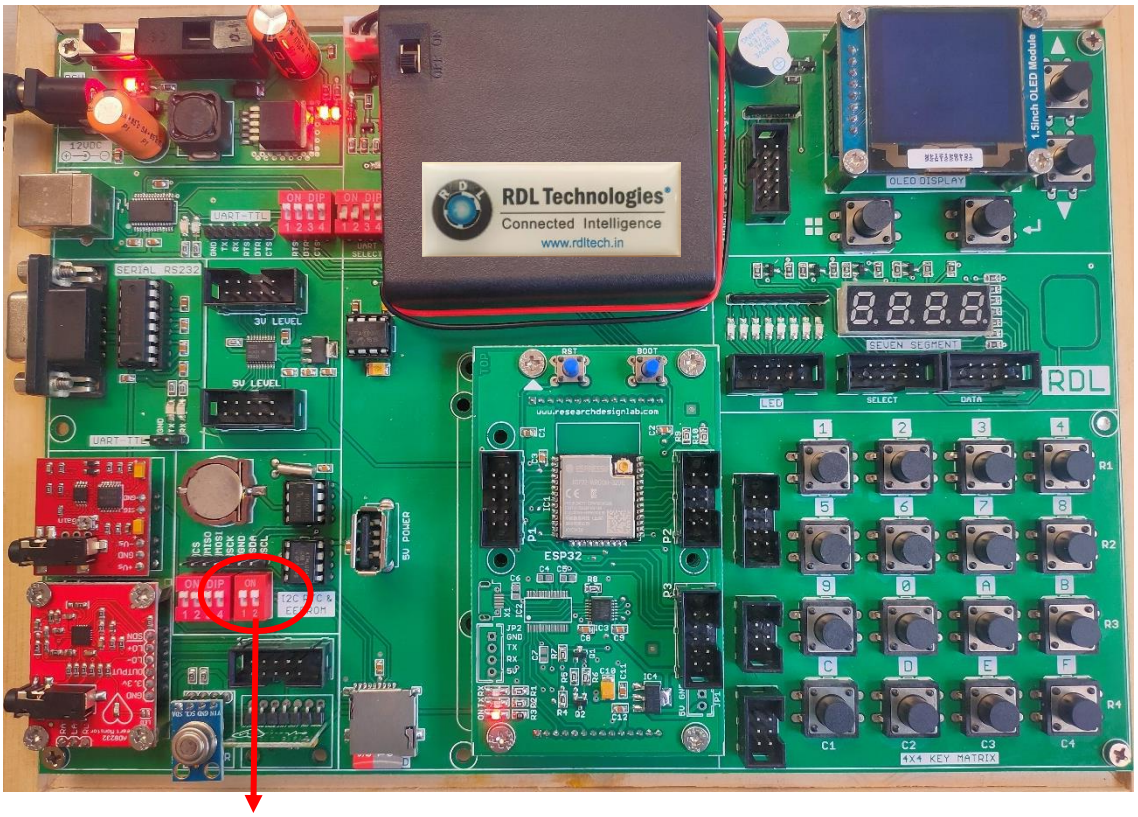
Interfacing Real Time Clock module with Biomedical Interfacing Kit.

Description:

To display Date and Time on the serial monitor using ESP 32 microcontroller development board.

Hardware required:

Biomedical Interfacing Kit and RTC Battery.



Make I2C Pins ON

Procedure:

1. Connect the USB cable to the board.
2. Open Arduino IDE .Select DOIT ESP32 DEVKIT V1in boards and select COM port.

3. Write the program, verify and Upload it.
4. Open the serial monitor to observe the output.

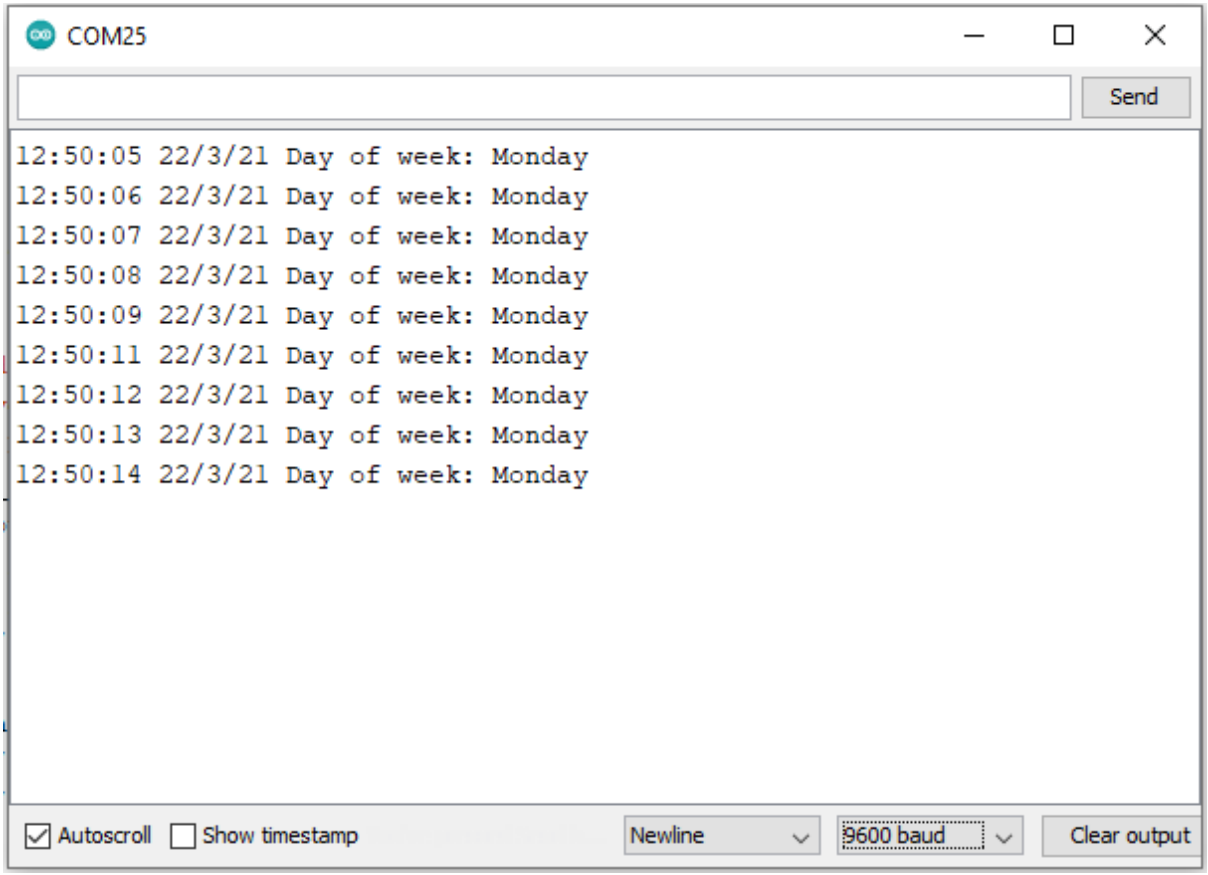
Program:

```
#include "Wire.h"
#define DS1307_I2C_ADDRESS 0x68
// Convert normal decimal numbers to binary coded decimal
byte decToBcd(byte val){
return( (val/10*16) + (val%10) );
}
// Convert binary coded decimal to normal decimal numbers
byte bcdToDec(byte val){
return( (val/16*10) + (val%16) );
}
void setup(){
Wire.begin();
Serial.begin(9600);
delay(1000);
 // set the initial time here:
setDS1307time(00,50,12,2,22,3,21); // DS1307 seconds, minutes,
hours, day, date, month, year
 delay(1000);
}
void setDS1307time(byte second, byte minute, byte hour, byte
dayOfWeek, byte
dayOfMonth, byte month, byte year){
// sets time and date data to DS1307
Wire.beginTransmission(DS1307_I2C_ADDRESS);
Wire.write(0); // set next input to start at the seconds
register
Wire.write(decToBcd(second)); // set seconds
Wire.write(decToBcd(minute)); // set minutes
Wire.write(decToBcd(hour)); // set hours
Wire.write(decToBcd(dayOfWeek)); // set day of week (1=Sunday,
7=Saturday)
Wire.write(decToBcd(dayOfMonth)); // set date (1 to 31)
Wire.write(decToBcd(month)); // set month
Wire.write(decToBcd(year)); // set year (0 to 99)
Wire.endTransmission();
}
void readDS1307time(byte *second, byte *minute, byte *hour,
byte *dayOfWeek, byte *dayOfMonth, byte *month, byte *year)
{
Wire.beginTransmission(DS1307_I2C_ADDRESS);
Wire.write(0); // set DS1307 register pointer to 00h
Wire.endTransmission();
delay(100);
Wire.requestFrom(DS1307_I2C_ADDRESS, 7);
// request seven bytes of data from DS1307 starting from
register 00h
```

```
*second = bcdToDec(Wire.read() & 0x7f);
*minute = bcdToDec(Wire.read());
*hour = bcdToDec(Wire.read() & 0x3f);
*dayOfWeek = bcdToDec(Wire.read());
*dayOfMonth = bcdToDec(Wire.read());
*month = bcdToDec(Wire.read());
*year = bcdToDec(Wire.read());
 Wire.endTransmission();
}
void displayTime(){
byte second, minute, hour, dayOfWeek, dayOfMonth, month, year;
// retrieve data from DS1307
readDS1307time(&second, &minute, &hour, &dayOfWeek,
&dayOfMonth, &month,
&year);
// send it to the serial monitor
Serial.print(hour, DEC);
// convert the byte variable to a decimal number when
displayed
Serial.print(":");
if (minute<10){
 Serial.print("0");
}
Serial.print(minute, DEC);
Serial.print(":");
if (second<10){
 Serial.print("0");
}
Serial.print(second, DEC);
Serial.print(" ");
Serial.print(dayOfMonth, DEC);
Serial.print("/");
Serial.print(month, DEC);
Serial.print("/");
Serial.print(year, DEC);
Serial.print(" Day of week: ");
switch(dayOfWeek){
case 1:
 Serial.println("Sunday");
 break;
case 2:
 Serial.println("Monday");
 break;

case 3:
 Serial.println("Tuesday");
 break;
case 4:
 Serial.println("Wednesday");
 break;
case 5:
```

```
 Serial.println("Thursday");
 break;
case 6:
 Serial.println("Friday");
 break;
case 7:
 Serial.println("Saturday");
 break;
}
}
void loop(){
displayTime(); // display the real-time clock data on the
Serial Monitor,
delay(1000); // every second
}
```

Output:



EXPERIMENT NO 5

SD Card

Aim:

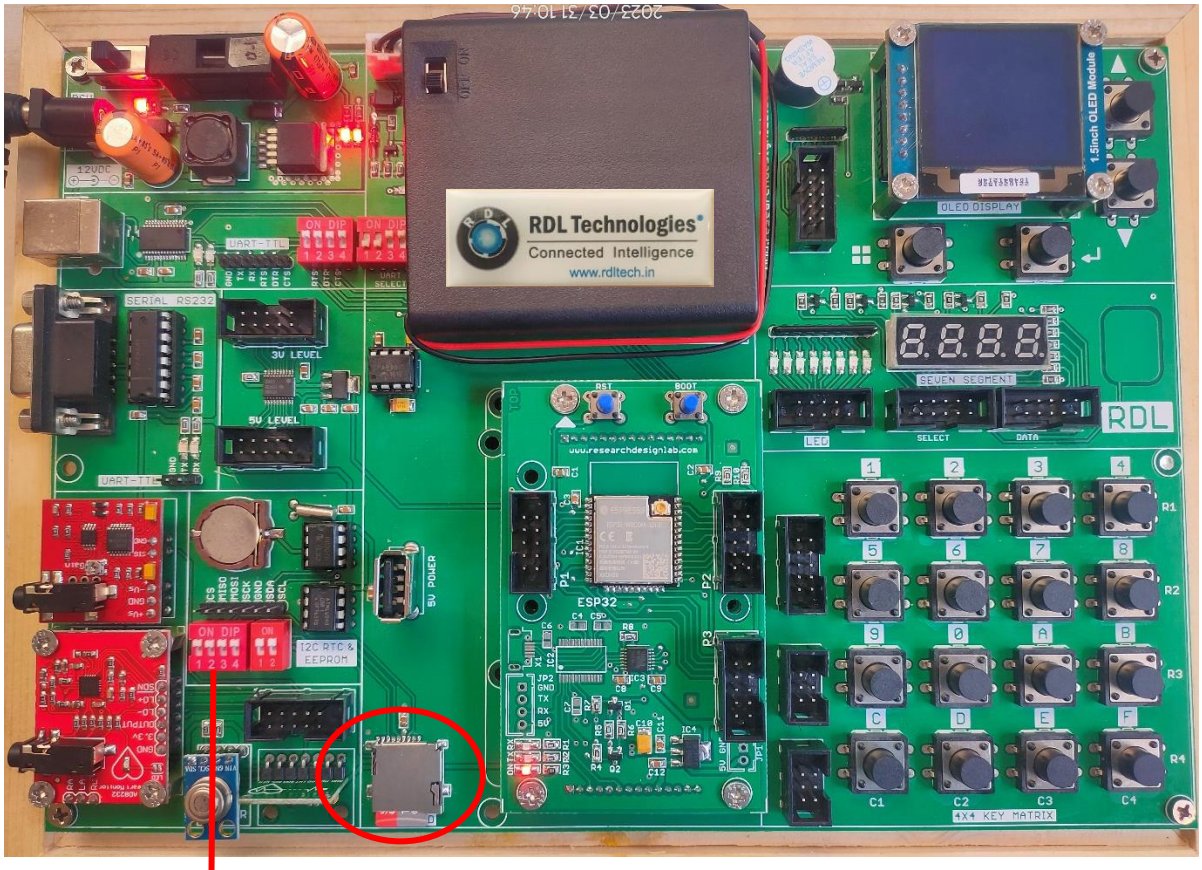
Interfacing SD card with Biomedical Interafcing Kit to list the directories stored in memory card.

Description:

To read the stored directories in SD card using Biomedical Interfacing Kit.

Hardware required:

ESP32-Microcontroller Development board and SD Card.



Make SPI Pins ON

Procedure:

1. Insert the SD Card in the slot given in the board.
2. Connect the USB cable to the board.
3. Open Arduino IDE .Select DOIT ESP32 DEVKIT V1 in boards and select COM port.

4. Write the program, verify and Upload it.
5. Open the serial monitor to observe the output.

Program:

```
#include "FS.h"
#include "SD.h"
#include "SPI.h"

void listDir(fs::FS &fs, const char * dirname, uint8_t
levels){
    Serial.printf("Listing directory: %s\n", dirname);

    File root = fs.open(dirname);
    if(!root){
        Serial.println("Failed to open directory");
        return;
    }
    if(!root.isDirectory()){
        Serial.println("Not a directory");
        return;
    }

    File file = root.openNextFile();
    while(file){
        if(file.isDirectory()){
            Serial.print("  DIR : ");
            Serial.println(file.name());
            if(levels){
                listDir(fs, file.path(), levels -1);
            }
        } else {
            Serial.print("  FILE: ");
            Serial.print(file.name());
            Serial.print("  SIZE: ");
            Serial.println(file.size());
        }
        file = root.openNextFile();
    }
}

void createDir(fs::FS &fs, const char * path){
    Serial.printf("Creating Dir: %s\n", path);
    if(fs.mkdir(path)){
        Serial.println("Dir created");
    } else {
        Serial.println("mkdir failed");
    }
}

void removeDir(fs::FS &fs, const char * path){
```

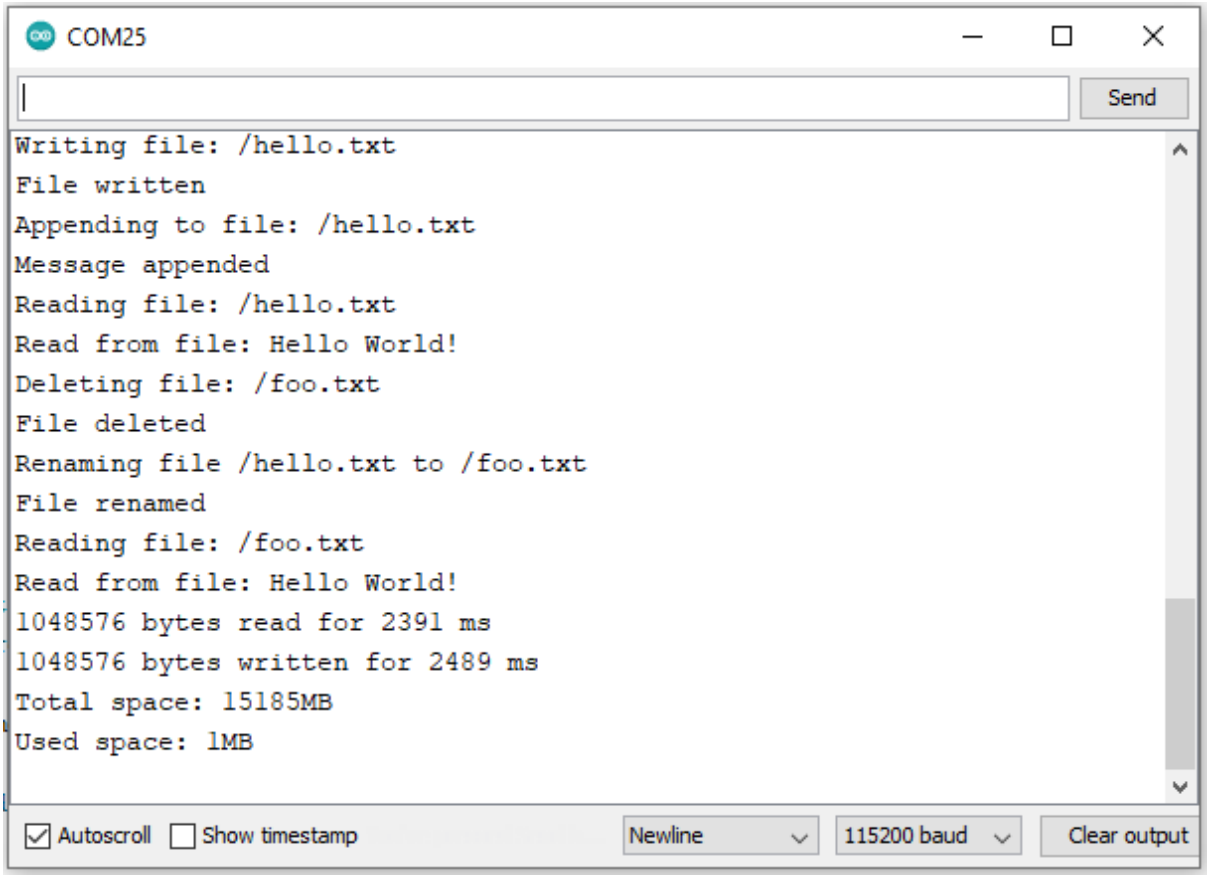
```
    Serial.printf("Removing Dir: %s\n", path);
    if(fs.rmdir(path)){
        Serial.println("Dir removed");
    } else {
        Serial.println("rmdir failed");
    }
}

void readFile(fs::FS &fs, const char * path){
    Serial.printf("Reading file: %s\n", path);

    File file = fs.open(path);
    if(!file){
        Serial.println("Failed to open file for reading");
        return;
    }

    Serial.print("Read from file: ");
    while(file.available()){
        Serial.write(file.read());
    }
    file.close();
}

void writeFile(fs::FS &fs, const char * path, const char *
message){
    Serial.printf("Writing file: %s\n", path);

    File file = fs.open(path, FILE_WRITE);
    if(!file){
        Serial.println("Failed to open file for writing");
        return;
    }
    if(file.print(message)){
        Serial.println("File written");
    } else {
        Serial.println("Write failed");
    }
    file.close();
}

void appendFile(fs::FS &fs, const char * path, const char *
message){
    Serial.printf("Appending to file: %s\n", path);

    File file = fs.open(path, FILE_APPEND);
    if(!file){
        Serial.println("Failed to open file for appending");
        return;
    }
    if(file.print(message)){
```

```
        Serial.println("Message appended");
    } else {
        Serial.println("Append failed");
    }
    file.close();
}

void renameFile(fs::FS &fs, const char * path1, const char *
path2){
    Serial.printf("Renaming file %s to %s\n", path1, path2);
    if (fs.rename(path1, path2)) {
        Serial.println("File renamed");
    } else {
        Serial.println("Rename failed");
    }
}

void deleteFile(fs::FS &fs, const char * path){
    Serial.printf("Deleting file: %s\n", path);
    if(fs.remove(path)){
        Serial.println("File deleted");
    } else {
        Serial.println("Delete failed");
    }
}

void testFileIO(fs::FS &fs, const char * path){
    File file = fs.open(path);
    static uint8_t buf[512];
    size_t len = 0;
    uint32_t start = millis();
    uint32_t end = start;
    if(file){
        len = file.size();
        size_t flen = len;
        start = millis();
        while(len){
            size_t toRead = len;
            if(toRead > 512){
                toRead = 512;
            }
            file.read(buf, toRead);
            len -= toRead;
        }
        end = millis() - start;
        Serial.printf("%u bytes read for %u ms\n", flen, end);
        file.close();
    } else {
        Serial.println("Failed to open file for reading");
    }
```

```
    file = fs.open(path, FILE_WRITE);
    if(!file){
        Serial.println("Failed to open file for writing");
        return;
    }

    size_t i;
    start = millis();
    for(i=0; i<2048; i++){
        file.write(buf, 512);
    }
    end = millis() - start;
    Serial.printf("%u bytes written for %u ms\n", 2048 * 512,
end);
    file.close();
}

void setup(){
    Serial.begin(115200);
    if(!SD.begin()){
        Serial.println("Card Mount Failed");
        return;
    }
    uint8_t cardType = SD.cardType();

    if(cardType == CARD_NONE){
        Serial.println("No SD card attached");
        return;
    }

    Serial.print("SD Card Type: ");
    if(cardType == CARD_MMC){
        Serial.println("MMC");
    } else if(cardType == CARD_SD){
        Serial.println("SDSC");
    } else if(cardType == CARD_SDHC){
        Serial.println("SDHC");
    } else {
        Serial.println("UNKNOWN");
    }

    uint64_t cardSize = SD.cardSize() / (1024 * 1024);
    Serial.printf("SD Card Size: %lluMB\n", cardSize);

    listDir(SD, "/", 0);
    createDir(SD, "/mydir");
    listDir(SD, "/", 0);
    removeDir(SD, "/mydir");
    listDir(SD, "/", 2);
    writeFile(SD, "/hello.txt", "Hello ");
```

```
    appendFile(SD, "/hello.txt", "World!\n");
    readFile(SD, "/hello.txt");
    deleteFile(SD, "/foo.txt");
    renameFile(SD, "/hello.txt", "/foo.txt");
    readFile(SD, "/foo.txt");
    testFileIO(SD, "/test.txt");
    Serial.printf("Total space: %lluMB\n", SD.totalBytes() /
(1024 * 1024));
    Serial.printf("Used space: %lluMB\n", SD.usedBytes() /
(1024 * 1024));
}

void loop(){

}
```

Output:



EXPERIMENT NO 6

OLED

Aim:

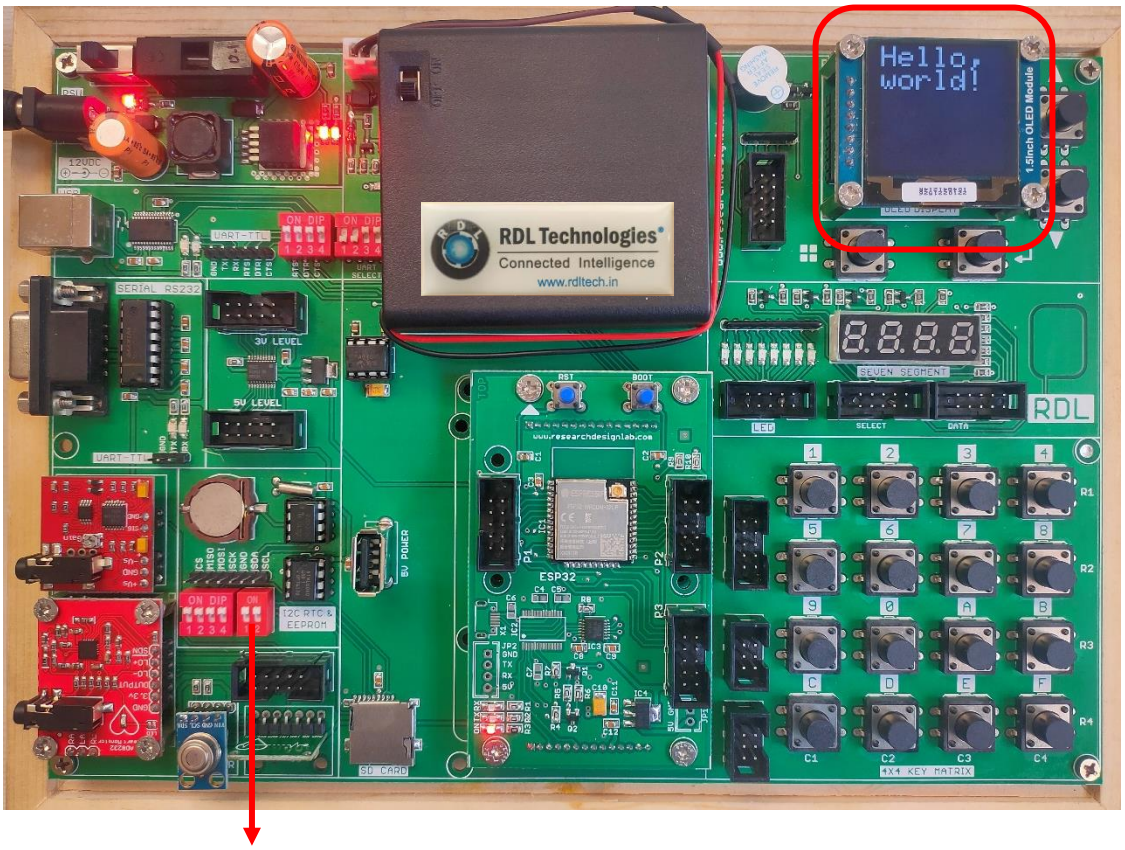
To Interface OLED Display with Biomedical Interfacing Kit.

Description:

To display message on OLED screen.

Hardware required:

Biomedical Interfacing Kit.



Make I2c Pins ON

Procedure:

1. Connect the USB cable to the board.
2. Open Arduino IDE .Select DOIT ESP32 DEVKIT V1 in boards and select COM port.
3. Write the program, verify and Upload it.

4. Now you can see the output displaying the message on OLED of Biomedical Interfacing Board.

Program:

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1327.h>

#define OLED_SDA 21
#define OLED_SCL 22

Adafruit_SSD1327 display(128, 128, &Wire, -1);

void setup() {
  Serial.begin(9600);
  Wire.begin(OLED_SDA, OLED_SCL);
  display.begin();
  display.clearDisplay();
  display.setTextColor(SSD1327_WHITE);
  display.setTextSize(3);
  display.setCursor(0, 0);
  display.println("Hello, world!");
  display.display();
}

void loop() {
}
```

EXPERIMENT NO 7

EMG Sensor

Aim:

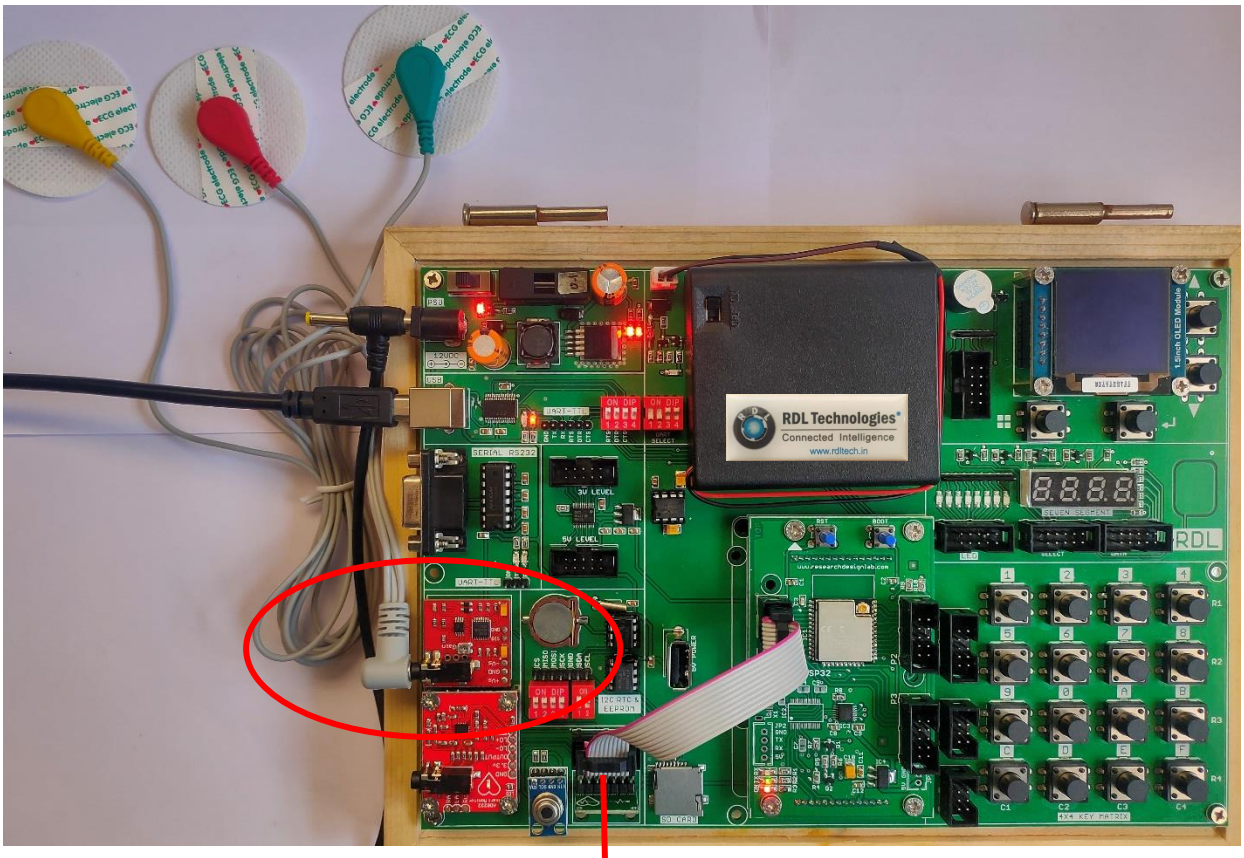
To Interface EMG (Electromyography) Sensor with Biomedical Interfacing Kit.

Description:

To monitor the Muscle activity in the Serial Plotter.

Hardware required:

Biomedical Interfacing Kit.



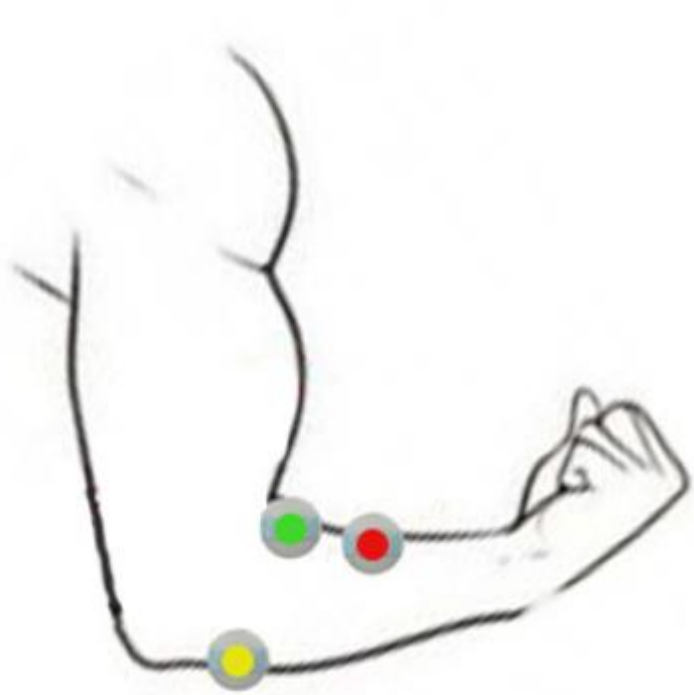
Connect P1 to the port shown in the above image.

Procedure:

1. Insert the Electrode in the slot given in the board.
2. Connect the USB cable to the board.
3. Open Arduino IDE .Select DOIT ESP32 DEVKIT V1 in boards and select COM port.
4. Write the program, verify and Upload it.
5. Connection of electrodes;

   - Green electrode: Place this electrode on the middle of the desired muscle
   - Red electrode: Place this electrode at the end of the desired muscle.
   - Yellow electrode: Place the last electrode on a bony or non-muscular part of the body near to the desired muscle.

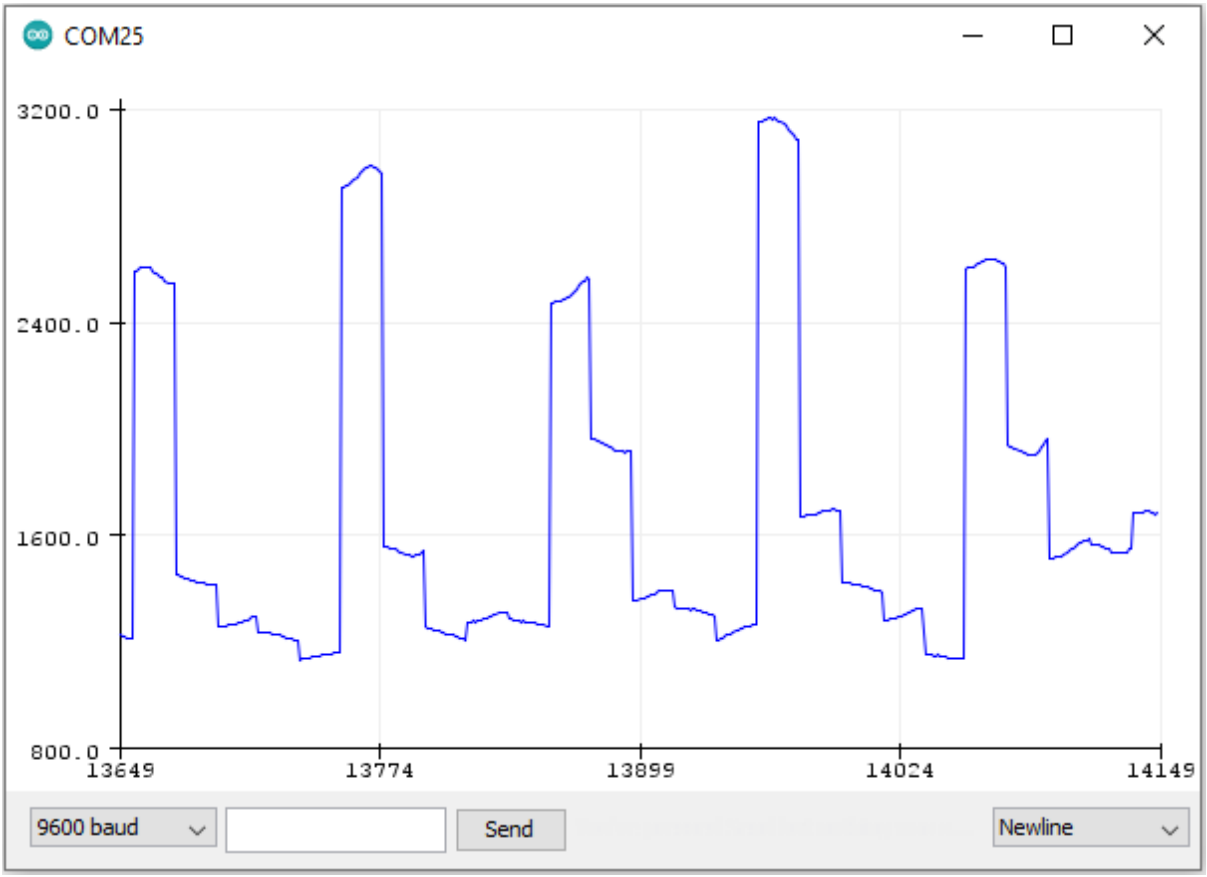
6. Open the serial plotter to observe the output.



Program:

```
#define EMG_PIN 39
void setup() {
Serial.begin(9600);
}

void loop() {
Serial.println(analogRead(EMG_PIN));
}
```

Output:



EXPERIMENT NO 8

ECG Sensor

Aim:

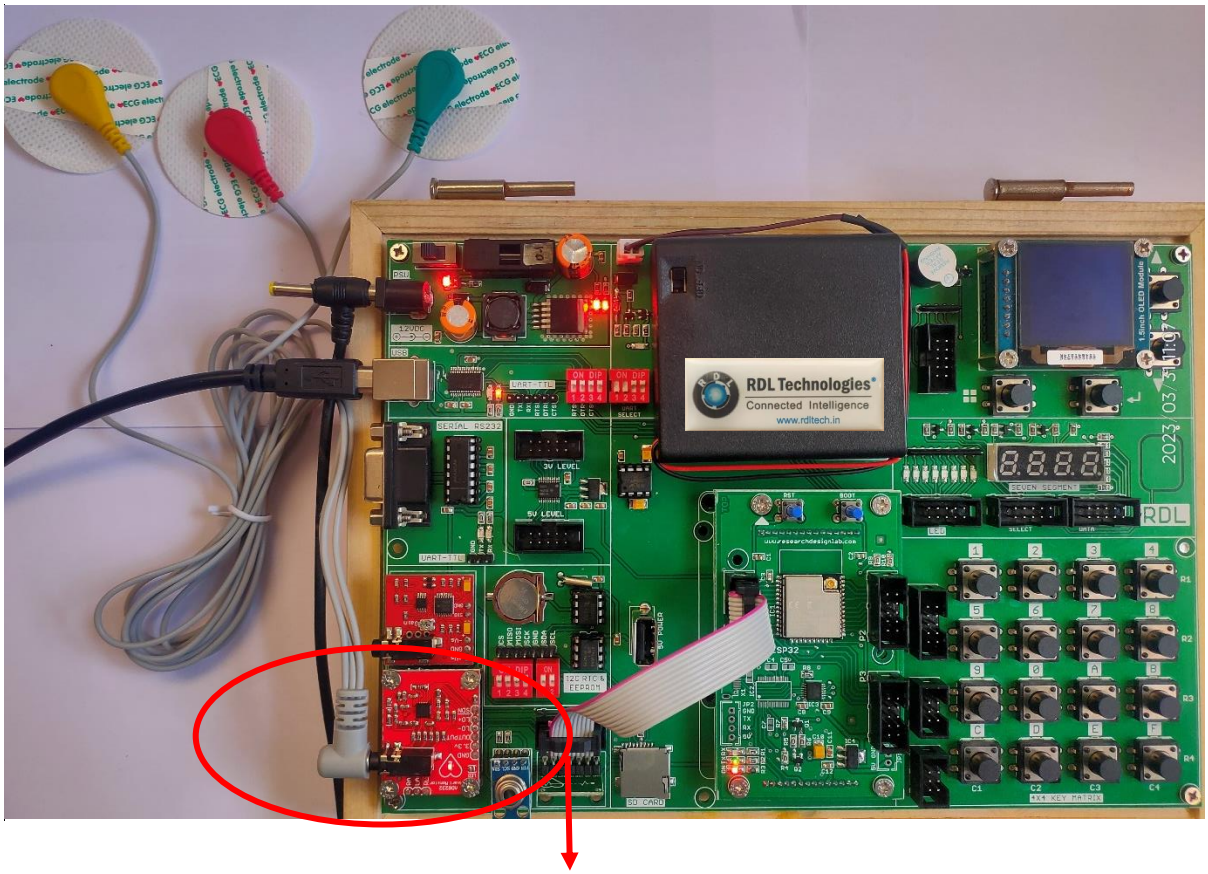
To Interface ECG (Electrocardiogram) Sensor with Biomedical Interfacing Kit.

Description:

To measure the electrical activity of the heart.

Hardware required:

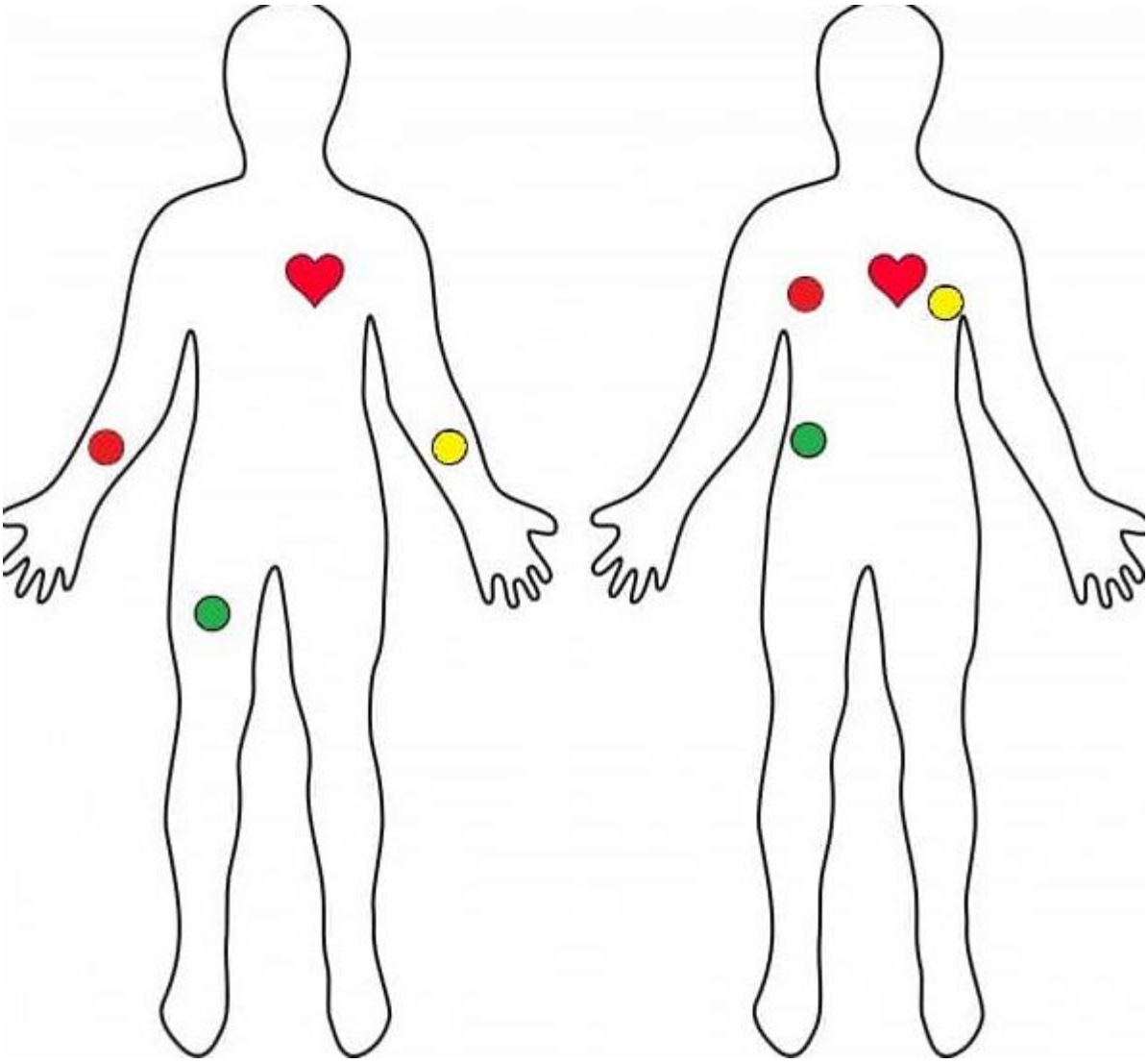
Biomedical Interfacing Kit.



Connect P1 to the port shown in the above image.

Procedure:

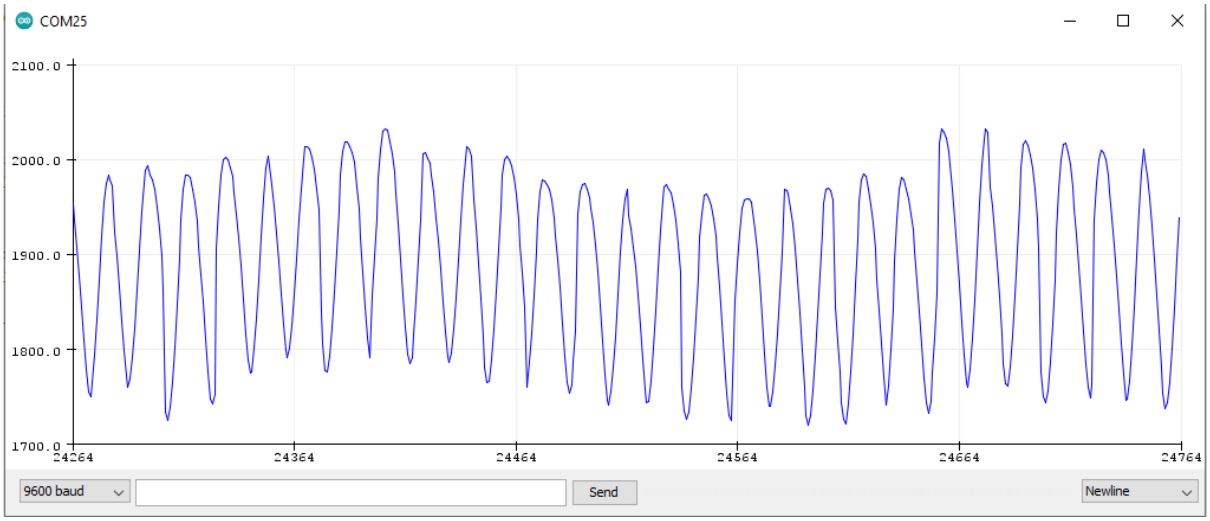
1. Insert the Electrode in the slot given in the board.
2. Connect the USB cable to the board.
3. Open Arduino IDE .Select DOIT ESP32 DEVKIT V1 in boards and select COM port.
4. Write the program, verify and Upload it.
5. Connection of electrodes;
    - Green electrode: Place this electrode on the middle of the desired muscle
    - Red electrode: Place this electrode at the end of the desired muscle.
    - Yellow electrode: Place the last electrode on a bony or non-muscular part of the body near to the desired muscle.
6. Open the serial plotter to observe the output.



Program:

```
#define LO_P 33
#define LO_N 32
#define Output 36
void setup() {
// initialize the serial communication:
Serial.begin(9600);
pinMode(LO_P, INPUT); // Setup for leads off detection LO +
pinMode(LO_N, INPUT); // Setup for leads off detection LO -

}

void loop() {

if((digitalRead(LO_P) == 1)||(digitalRead(LO_N) == 1)){
Serial.println('!');
}
else{
// send the value of analog input 0:
Serial.println(analogRead(Output));
}
//Wait for a bit to keep serial data from saturating
delay(1);
}
```

Output:



EXPERIMENT NO 9

Pulse Oximeter and Heart Rate Sensor

Aim:

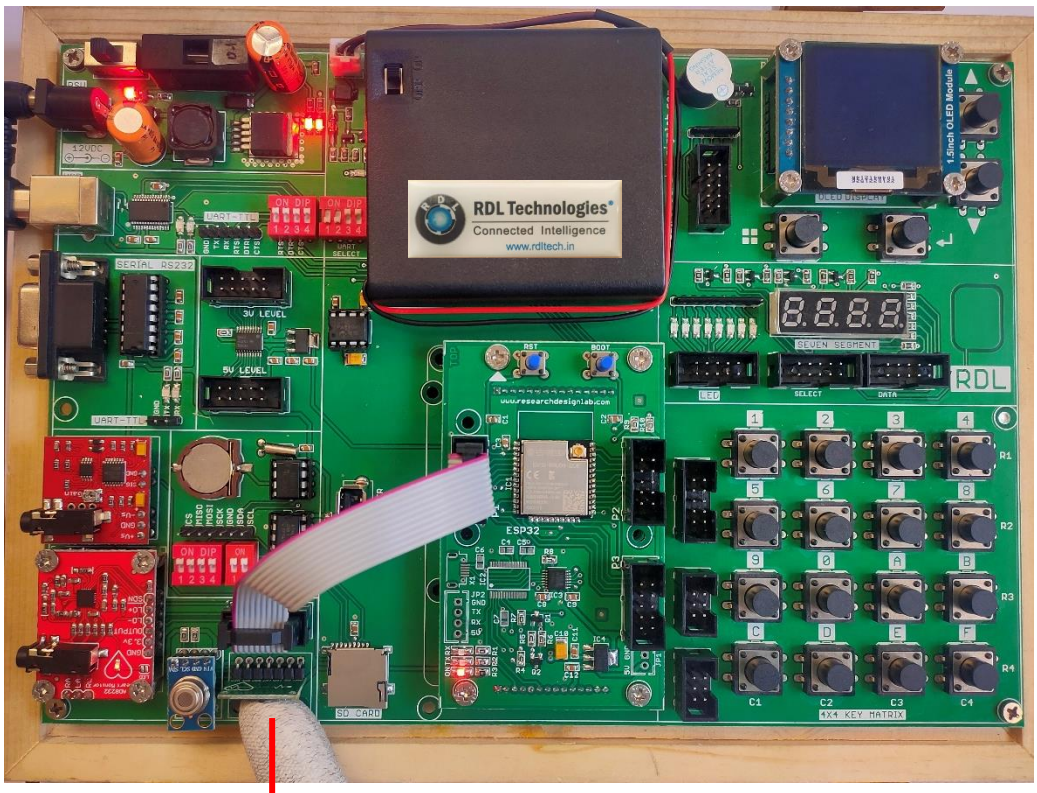
To Interface Pulse Oximeter and Heart Rate Sensor.

Description:

To monitor the Heart Rate/SpO2.

Hardware required:

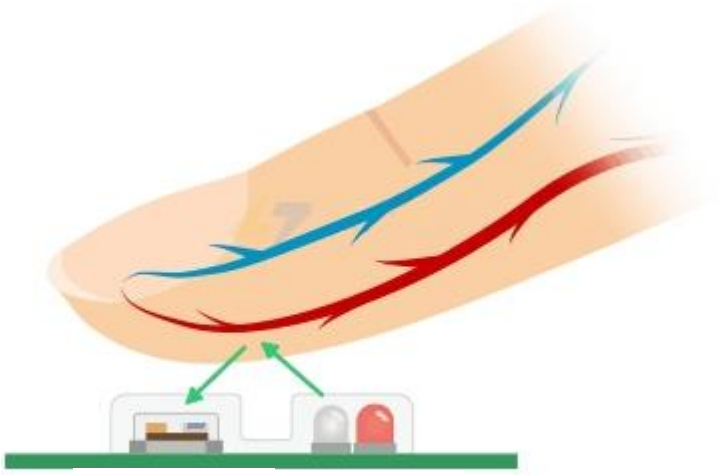
Biomedical Interfacing Kit.



Place the finger here.

Procedure:

- Connect the USB cable to the board.
- Open Arduino IDE .Select DOIT ESP32 DEVKIT V1 in boards and select COM port.
- Write the program, verify and Upload it.
- Now you can see the output displaying on serial monitor.

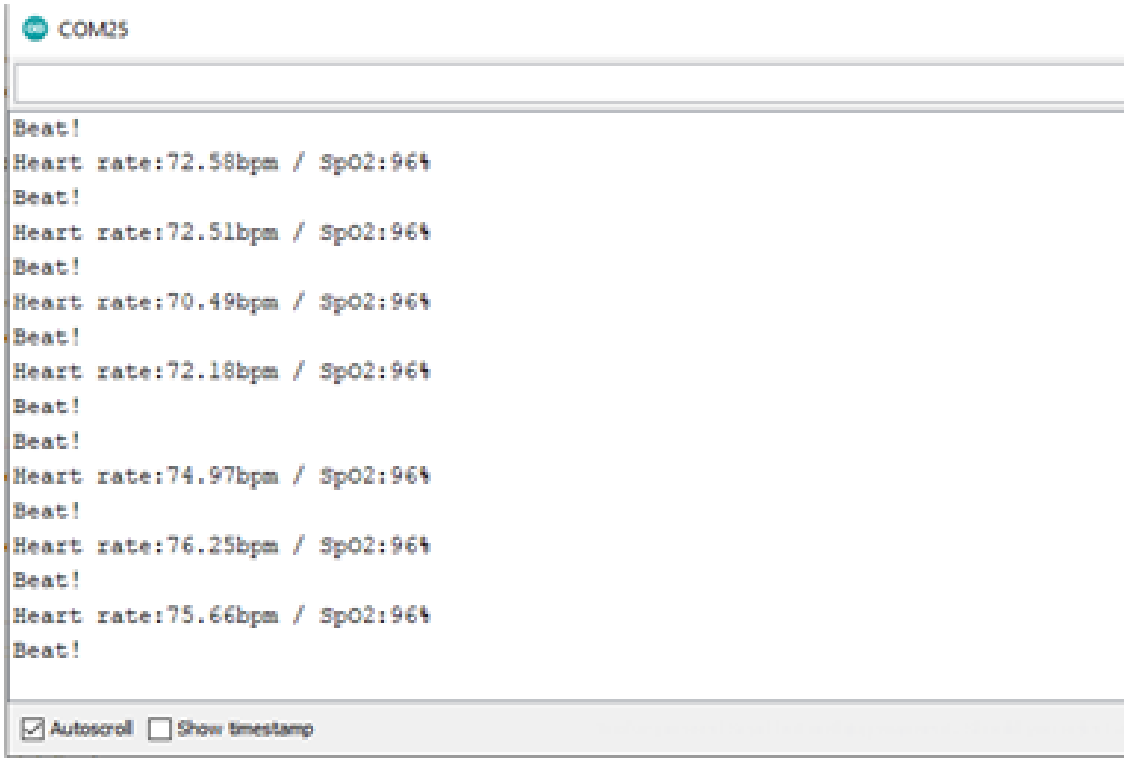


Program:

```
#include <Wire.h>
#include <MAX30100_PulseOximeter.h>

#define REPORTING_PERIOD_MS     1000

// Create a PulseOximeter object
PulseOximeter pox;

// Time at which the last beat occurred
uint32_t tsLastReport = 0;

// Callback routine is executed when a pulse is detected
void onBeatDetected() {
    Serial.println("Beat!");
}

void setup() {
    Serial.begin(9600);

    Serial.print("Initializing pulse oximeter..");

    // Initialize sensor
    if (!pox.begin()) {
        Serial.println("FAILED");
        for(;;);
    } else {
        Serial.println("SUCCESS");
    }

  // Configure sensor to use 7.6mA for LED drive
  pox.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);
```

```
    // Register a callback routine
    pox.setOnBeatDetectedCallback(onBeatDetected);
}

void loop() {
    // Read from the sensor
    pox.update();

    // Grab the updated heart rate and SpO2 levels
    if (millis() - tsLastReport > REPORTING_PERIOD_MS) {
        Serial.print("Heart rate:");
        Serial.print(pox.getHeartRate());
        Serial.print("bpm / SpO2:");
        Serial.print(pox.getSpO2());
        Serial.println("%");

        tsLastReport = millis();
    }
}
```

Output:



EXPERIMENT NO 10

Infrared Temperature Sensor

Aim:

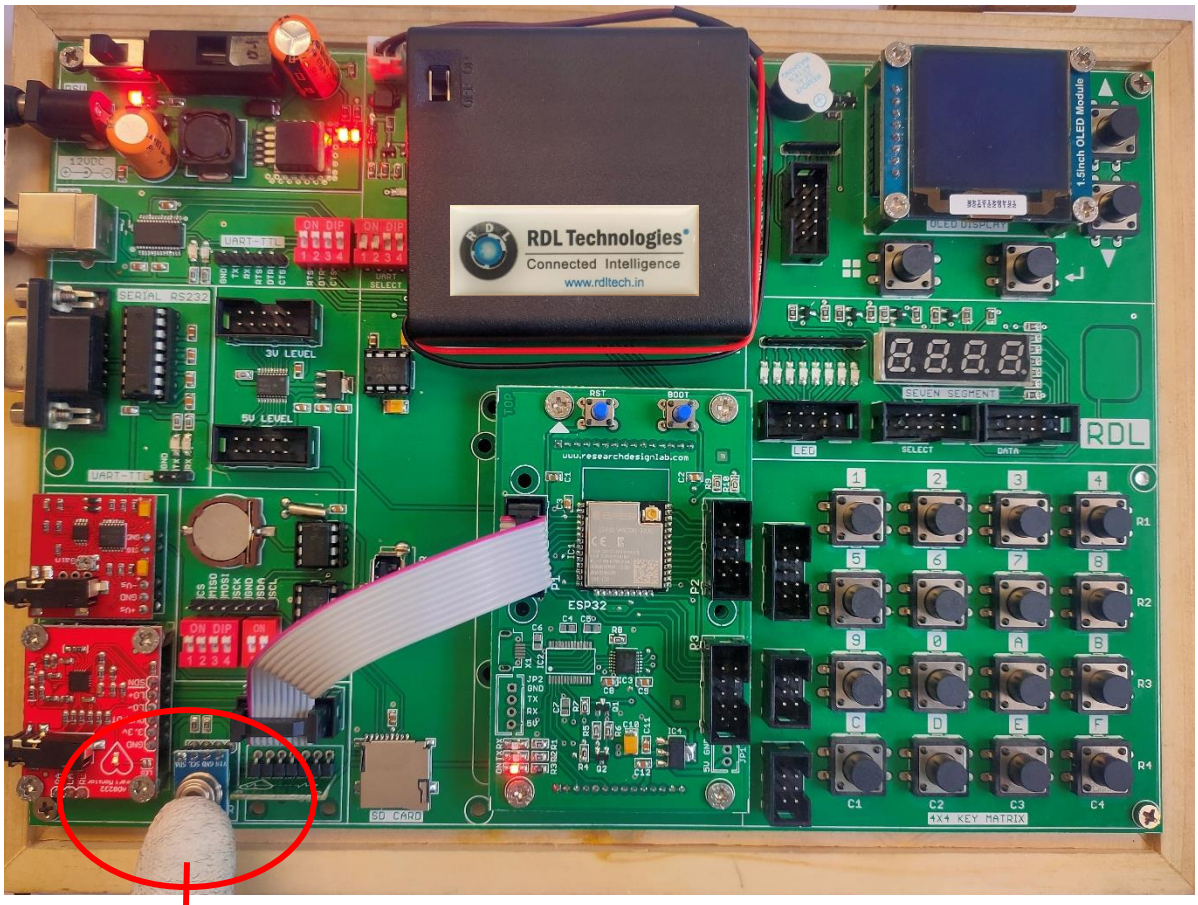
To Interface Infrared Temperature Sensor with Biomedical Interfacing Kit.

Description:

To measure the temperature of the body or object.

Hardware required:

Biomedical Interfacing Kit



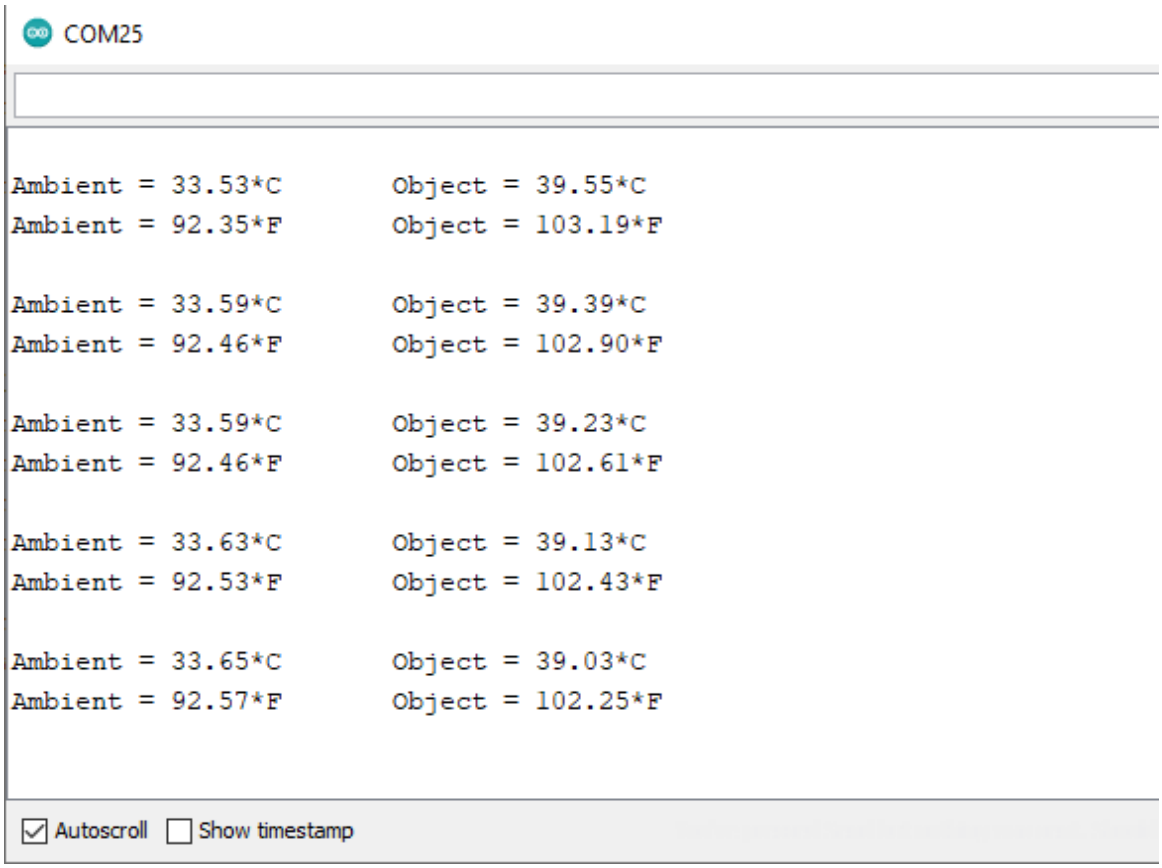
Procedure:

- Connect the USB cable to the board.
- Open Arduino IDE .Select DOIT ESP32 DEVKIT V1 in boards and select COM port.
- Write the program, verify and Upload it.
- Now you can see the output displaying in the serial monitor.

Porgram:

```
#include <Wire.h>
#include <Adafruit_MLX90614.h>

Adafruit_MLX90614 mlx = Adafruit_MLX90614();

void setup() {
  Serial.begin(9600);

  Serial.println("Adafruit MLX90614 test");

  mlx.begin();
}

void loop() {
  Serial.print("Ambient = "); Serial.print(mlx.readAmbientTempC());
  Serial.print("*C\tObject = "); Serial.print(mlx.readObjectTempC());
Serial.println("*C");
  Serial.print("Ambient = "); Serial.print(mlx.readAmbientTempF());
  Serial.print("*F\tObject = "); Serial.print(mlx.readObjectTempF());
Serial.println("*F");

  Serial.println();
  delay(500);
}
```

Output:


COM25
Ambient = 33.53*C Object = 39.55*C
Ambient = 92.35*F Object = 103.19*F
Ambient = 33.59*C Object = 39.39*C
Ambient = 92.46*F Object = 102.90*F
Ambient = 33.59*C Object = 39.23*C
Ambient = 92.46*F Object = 102.61*F
Ambient = 33.63*C Object = 39.13*C
Ambient = 92.53*F Object = 102.43*F
Ambient = 33.65*C Object = 39.03*C
Ambient = 92.57*F Object = 102.25*F
Autoscroll
Show timestamp