

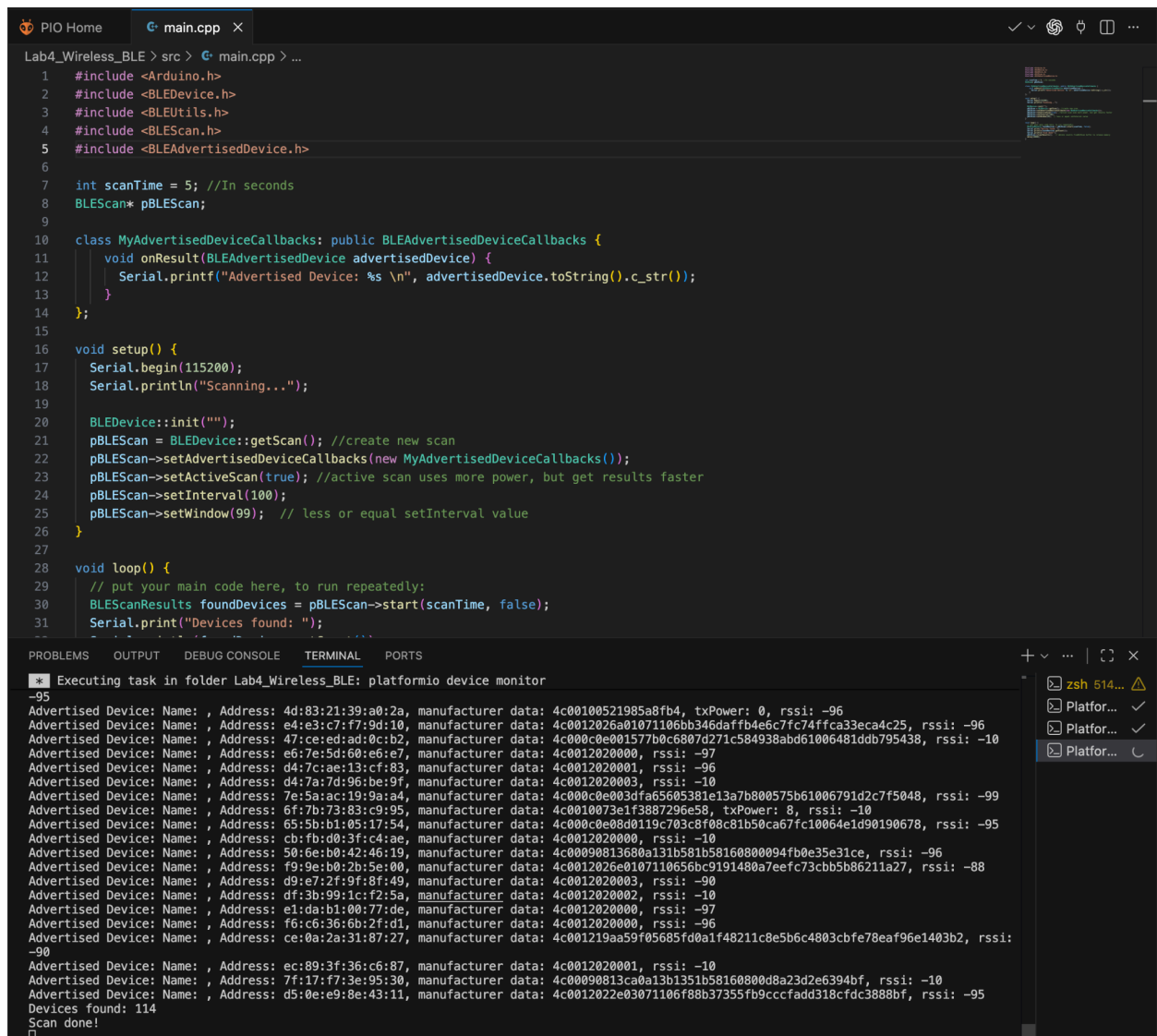
Lab 4: Wireless

Lewis Liu, Iris Yang

Github Link: <https://github.com/IrisYzh/TECHIN-514-Hardware-Software-Lab>

Section 1: MCU as BLEScanner

Screenshot of your serial monitor displaying the number of Bluetooth devices detected using your MCU as BLEScanner



The screenshot shows the Arduino IDE interface. The top pane displays the `main.cpp` file for the `Lab4_Wireless_BLE` project. The code includes headers for `Arduino.h`, `BLEDevice`, `BLEUtils`, `BLEScan`, and `BLEAdvertisedDevice`. It defines a `MyAdvertisedDeviceCallbacks` class with an `onResult` method that prints the details of each discovered device. The `setup` function initializes the serial port at 115200 baud and configures the BLE scanner with a 5-second scan time, active scanning, and a 100ms interval. The `loop` function repeatedly starts the scan and prints the number of devices found.

The bottom pane shows the serial monitor output, which displays a list of discovered Bluetooth devices. Each entry includes the device name, address, manufacturer data, txPower, and rssi. The output ends with "Devices found: 114" and "Scan done!".

```
Lab4_Wireless_BLE > src > main.cpp > ...
1  #include <Arduino.h>
2  #include <BLEDevice.h>
3  #include <BLEUtils.h>
4  #include <BLEScan.h>
5  #include <BLEAdvertisedDevice.h>
6
7  int scanTime = 5; //In seconds
8  BLEScan* pBLEScan;
9
10 class MyAdvertisedDeviceCallbacks: public BLEAdvertisedDeviceCallbacks {
11     void onResult(BLEAdvertisedDevice advertisedDevice) {
12         Serial.printf("Advertised Device: %s \n", advertisedDevice.toString().c_str());
13     }
14 };
15
16 void setup() {
17     Serial.begin(115200);
18     Serial.println("Scanning...");
19
20     BLEDevice::init("");
21     pBLEScan = BLEDevice::getScan(); //create new scan
22     pBLEScan->setAdvertisedDeviceCallbacks(new MyAdvertisedDeviceCallbacks());
23     pBLEScan->setActiveScan(true); //active scan uses more power, but get results faster
24     pBLEScan->setInterval(100);
25     pBLEScan->setWindow(99); // less or equal setInterval value
26 }
27
28 void loop() {
29     // put your main code here, to run repeatedly:
30     BLEScanResults foundDevices = pBLEScan->start(scanTime, false);
31     Serial.print("Devices found: ");
32     Serial.println(foundDevices.getCount());
33 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

* Executing task in folder Lab4_Wireless_BLE: platformio device monitor

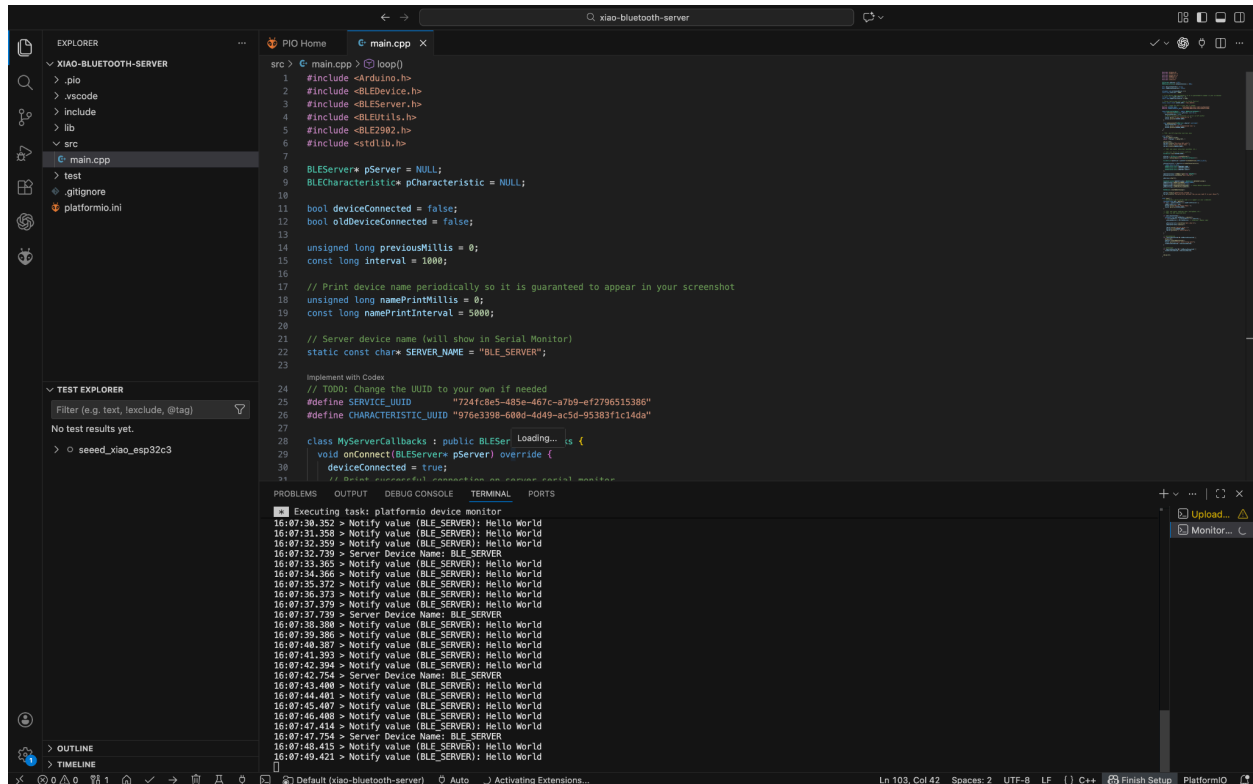
```
-95
Advertised Device: Name: , Address: 4d:83:21:39:a0:2a, manufacturer data: 4c00100521985a8fb4, txPower: 0, rssi: -96
Advertised Device: Name: , Address: e4:e3:c7:f7:9d:10, manufacturer data: 4c0012026a01071106bb346daffb4e6c7fc74ffca33eca4c25, rssi: -96
Advertised Device: Name: , Address: 47:ce:ed:ad:0c:b2, manufacturer data: 4c000c0e001577b0c6807d271c584938abd61006481ddb795438, rssi: -10
Advertised Device: Name: , Address: e6:7e:5d:60:e6:e7, manufacturer data: 4c0012020000, rssi: -97
Advertised Device: Name: , Address: d4:7c:ae:13:cf:83, manufacturer data: 4c0012020001, rssi: -96
Advertised Device: Name: , Address: d4:7a:7d:96:be:9f, manufacturer data: 4c0012020003, rssi: -10
Advertised Device: Name: , Address: 7e:5a:ac:19:9a:a4, manufacturer data: 4c000c0e003dfa65605381e13a7b00575b61006791d2c7f5048, rssi: -99
Advertised Device: Name: , Address: 6f:7b:73:83:c9:95, manufacturer data: 4c0010073e1f3887296e58, txPower: 8, rssi: -10
Advertised Device: Name: , Address: 65:5b:b1:05:17:54, manufacturer data: 4c000c0e08d0119c703c8f08c81b50ca67fc10064e1d90190678, rssi: -95
Advertised Device: Name: , Address: cb:fb:d0:3f:c4:ae, manufacturer data: 4c0012020000, rssi: -10
Advertised Device: Name: , Address: 50:6e:b0:42:46:19, manufacturer data: 4c00090813680a131b581b58160800094fb0e35e31ce, rssi: -96
Advertised Device: Name: , Address: f9:9e:b0:2b:5e:00, manufacturer data: 4c0012026e0107110656bc9191480a7eefc73cbb5b86211a27, rssi: -88
Advertised Device: Name: , Address: d9:e7:2f:9f:8f:49, manufacturer data: 4c0012020003, rssi: -90
Advertised Device: Name: , Address: df:3b:99:1c:f2:5a, manufacturer data: 4c0012020002, rssi: -10
Advertised Device: Name: , Address: e1:da:b1:00:77:de, manufacturer data: 4c0012020000, rssi: -97
Advertised Device: Name: , Address: f6:c6:36:6b:2f:d1, manufacturer data: 4c0012020000, rssi: -96
Advertised Device: Name: , Address: ce:0a:2a:31:87:27, manufacturer data: 4c001219aa59f05685fd0a1f48211c8e5b6c4803cbfe78eaf96e1403b2, rssi: -90
Advertised Device: Name: , Address: ec:89:3f:36:c6:87, manufacturer data: 4c0012020001, rssi: -10
Advertised Device: Name: , Address: 7f:17:f7:3e:95:30, manufacturer data: 4c00090813ca0a13b1351b58160800d8a23d2e6394bf, rssi: -10
Advertised Device: Name: , Address: d5:0e:e9:8e:43:11, manufacturer data: 4c0012022e03071106f88b37355fb9cccfadd318cfdc3888bf, rssi: -95
Devices found: 114
Scan done!

```

Section 2: Establishing Bluetooth Connection

Screenshot of the serial monitor of your client device to show successful connection with your server device (make sure the server device's name is included).

Server



The screenshot displays the Arduino IDE interface with a project named 'xiao-bluetooth-server'. The Explorer panel on the left shows the file structure, including 'main.cpp'. The main.cpp file is open in the editor, showing the following code:

```
src > main.cpp > loop()
1 #include <Arduino.h>
2 #include <BLEDevice.h>
3 #include <BLEServer.h>
4 #include <BLEUtils.h>
5 #include <BLE2902.h>
6 #include <stdlib.h>
7
8 BLEServer pServer = NULL;
9 BLECharacteristic pCharacteristic = NULL;
10
11 bool deviceConnected = false;
12 bool oldDeviceConnected = false;
13
14 unsigned long previousMillis = 0;
15 const long interval = 1000;
16
17 // Print device name periodically so it is guaranteed to appear in your screenshot
18 unsigned long namePrintMillis = 0;
19 const long namePrintInterval = 5000;
20
21 // Server device name (will show in Serial Monitor)
22 static const char* SERVER_NAME = "BLE_SERVER";
23
24 Implement with Codes
25 // TODO: Change the UUID to your own if needed
26 #define SERVICE_UUID "724c8e5-485e-467c-a709-ef2796515386"
27 #define CHARACTERISTIC_UUID "976e3398-688d-4d49-ac5d-95383f1c14da"
28
29 class MyServerCallbacks : public BLEServerCallbacks {
30   void onConnect(BLEServer* pServer) override {
31     deviceConnected = true;
32   }
33   // Brief successful connection on some serial monitor
34 }
```

The Serial Monitor at the bottom shows the output of the program, indicating a successful connection and periodic data transmission:

```
Executing task: platformio device monitor
16:07:34.352 > Notify value (BLE_SERVER): Hello World
16:07:35.358 > Notify value (BLE_SERVER): Hello World
16:07:36.359 > Notify value (BLE_SERVER): Hello World
16:07:37.359 > Server Device Name: BLE_SERVER
16:07:38.365 > Notify value (BLE_SERVER): Hello World
16:07:39.366 > Notify value (BLE_SERVER): Hello World
16:07:40.372 > Notify value (BLE_SERVER): Hello World
16:07:41.373 > Notify value (BLE_SERVER): Hello World
16:07:42.379 > Notify value (BLE_SERVER): Hello World
16:07:43.380 > Server Device Name: BLE_SERVER
16:07:44.386 > Notify value (BLE_SERVER): Hello World
16:07:45.387 > Notify value (BLE_SERVER): Hello World
16:07:46.393 > Notify value (BLE_SERVER): Hello World
16:07:47.394 > Notify value (BLE_SERVER): Hello World
16:07:48.394 > Server Device Name: BLE_SERVER
16:07:49.400 > Notify value (BLE_SERVER): Hello World
16:07:50.401 > Notify value (BLE_SERVER): Hello World
16:07:51.407 > Notify value (BLE_SERVER): Hello World
16:07:52.414 > Notify value (BLE_SERVER): Hello World
16:07:53.415 > Server Device Name: BLE_SERVER
16:07:54.421 > Notify value (BLE_SERVER): Hello World
```

Client

The screenshot displays a Visual Studio Code (VS Code) IDE interface for a project named "xiao-bluetooth-client". The Explorer pane on the left shows the project structure, including folders like "pio", ".vscode", "include", "lib", "src", and files like "main.cpp", "test", ".gitignore", and "platformio.ini". The main editor window shows the "main.cpp" file, which contains C++ code for a BLE client. The code includes headers for Arduino, BLEDevice, BLEUtils, BLEScan, and BLEAdvertisedDevice. It defines static variables for service and characteristic UUIDs, and a static boolean for connection status. The code also includes a function "notifyCallback" that prints the server name and data received from the BLE server. The terminal at the bottom shows the execution of the program, which outputs "data: Hello World" and "Notify from BLE_SERVER" messages. The right sidebar shows the "Build with Agent" panel, which includes a "Generate Agent Instructions" button and a "Build with Agent" button.

```
src > main.cpp > ...
1 #include <Arduino.h>
2 #include <BLEDevice.h>
3 #include <BLEUtils.h>
4 #include <BLEScan.h>
5 #include <BLEAdvertisedDevice.h>
6
7 // TODO: change these UUIDs to match your server
8 static BLEUUID serviceUUID("724fc8e5-485e-467c-a7b9-ef2796515386");
9 static BLEUUID charUUID("976e3398-600d-4d49-ac5d-95383f1c14da");
10
11 static boolean doConnect = false;
12 static boolean connected = false;
13 static boolean doScan = false;
14
15 static BLERemoteCharacteristic* pRemoteCharacteristic = nullptr;
16 static BLEAdvertisedDevice* myDevice = nullptr;
17
18 // NEW: store server name for screenshot
19 static String serverName = "Unknown";
20
21 static void notifyCallback(
22     BLERemoteCharacteristic* pBLERemoteCharacteristic,
23     uint8_t* pData,
24     size_t length,
25     bool isNotify) {
26
27     Serial.print("Notify from ");
28     Serial.print(serverName);
```

Executing task: C:\Users\lbj04\platformio\penv\Scripts\platformio.exe device monitor --environment seed_xiao_esp32c3

data: Hello World

Writing to BLE_SERVER: Time since boot: 26

Notify from BLE_SERVER | Char 976e3398-600d-4d49-ac5d-95383f1c14da | len 11

data: Hello World

Writing to BLE_SERVER: Time since boot: 27

Notify from BLE_SERVER | Char 976e3398-600d-4d49-ac5d-95383f1c14da | len 11

data: Hello World

Writing to BLE_SERVER: Time since boot: 28

Notify from BLE_SERVER | Char 976e3398-600d-4d49-ac5d-95383f1c14da | len 11

data: Hello World

Ln 18, Col 37 Spaces: 2 UTF-8 LF C++ Finish Setup PlatformIO

Build with Agent

AI responses may be inaccurate.

Generate Agent Instructions to onboard AI onto your codebase.

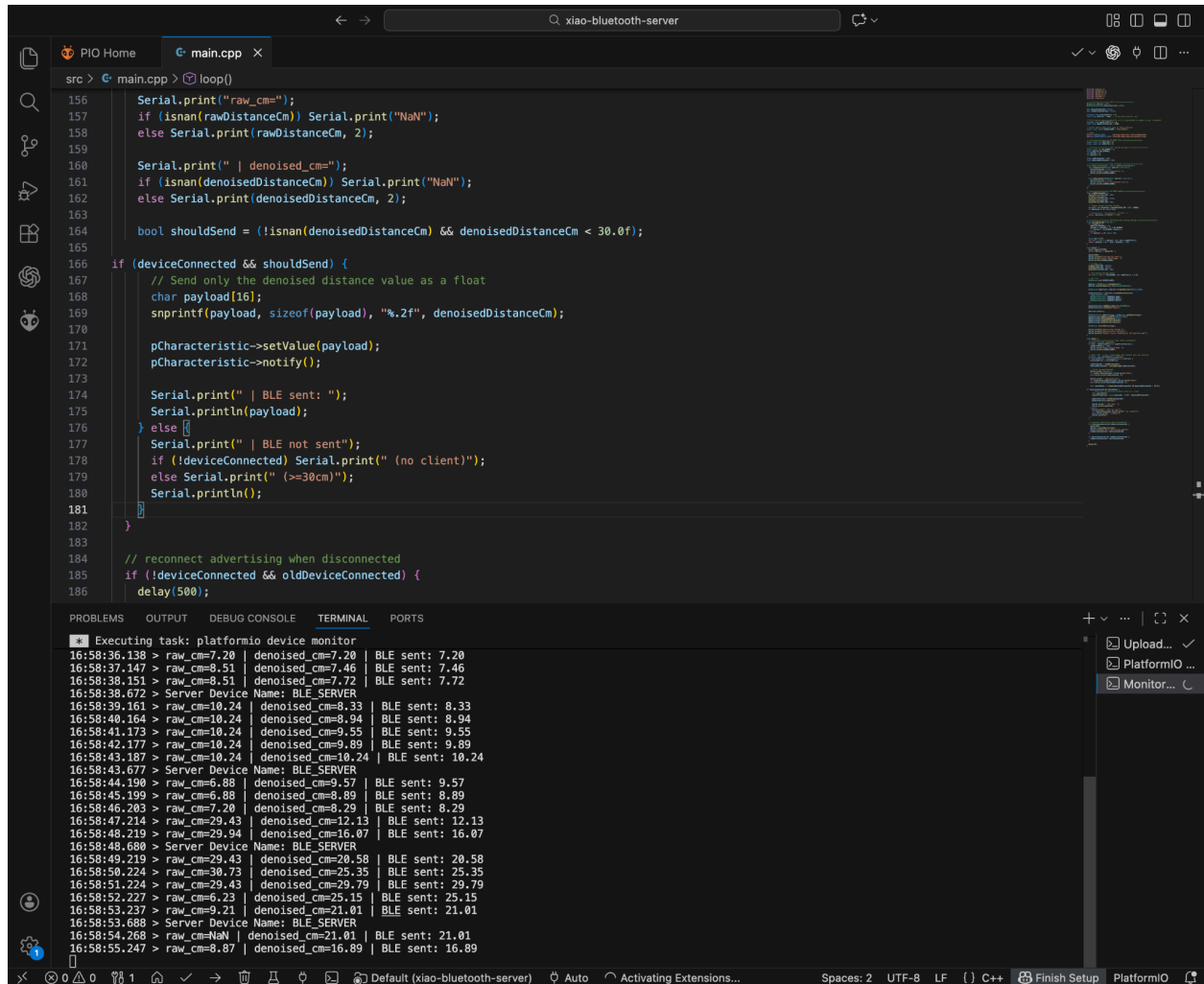
main.cpp

Describe what to build next

Agent Auto

4.3 Sensor Device Signal Processing and Data Transmission

Screenshot of the serial monitor of your server device to show the raw and denoised sensor data.

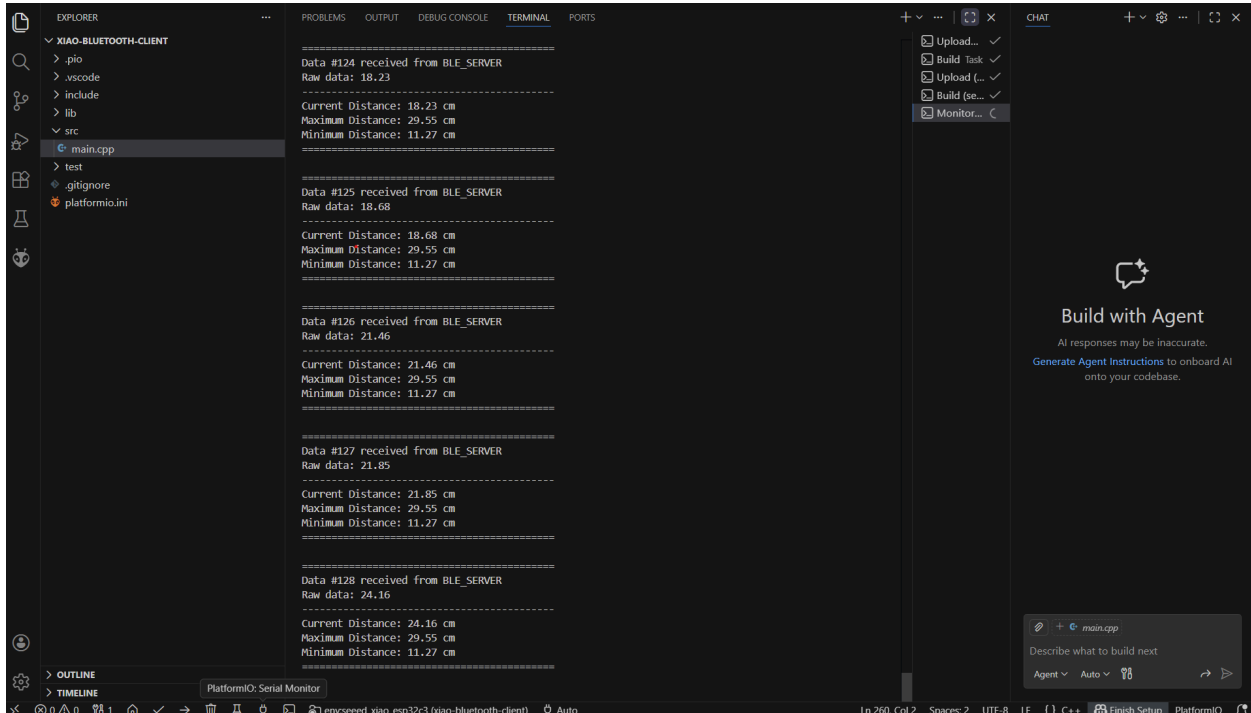


The screenshot displays the Arduino IDE interface. The main.cpp file is open, showing code for a BLE server. The serial monitor is active, displaying the output of the program. The output shows raw sensor data (raw_cm) and denoised sensor data (denoised_cm) being sent via BLE. The output is formatted as follows:

```
16:58:36.138 > raw_cm=7.20 | denoised_cm=7.20 | BLE sent: 7.20
16:58:37.147 > raw_cm=8.51 | denoised_cm=7.46 | BLE sent: 7.46
16:58:38.151 > raw_cm=8.51 | denoised_cm=7.72 | BLE sent: 7.72
16:58:38.672 > Server Device Name: BLE_SERVER
16:58:39.161 > raw_cm=10.24 | denoised_cm=8.33 | BLE sent: 8.33
16:58:40.164 > raw_cm=10.24 | denoised_cm=8.94 | BLE sent: 8.94
16:58:41.173 > raw_cm=10.24 | denoised_cm=9.55 | BLE sent: 9.55
16:58:42.177 > raw_cm=10.24 | denoised_cm=9.89 | BLE sent: 9.89
16:58:43.187 > raw_cm=10.24 | denoised_cm=10.24 | BLE sent: 10.24
16:58:43.677 > Server Device Name: BLE_SERVER
16:58:44.190 > raw_cm=6.88 | denoised_cm=9.57 | BLE sent: 9.57
16:58:45.199 > raw_cm=6.88 | denoised_cm=8.89 | BLE sent: 8.89
16:58:46.203 > raw_cm=7.20 | denoised_cm=8.29 | BLE sent: 8.29
16:58:47.214 > raw_cm=29.43 | denoised_cm=12.13 | BLE sent: 12.13
16:58:48.219 > raw_cm=29.94 | denoised_cm=16.07 | BLE sent: 16.07
16:58:48.680 > Server Device Name: BLE_SERVER
16:58:49.219 > raw_cm=29.43 | denoised_cm=20.58 | BLE sent: 20.58
16:58:50.224 > raw_cm=30.73 | denoised_cm=25.35 | BLE sent: 25.35
16:58:51.224 > raw_cm=29.43 | denoised_cm=29.79 | BLE sent: 29.79
16:58:52.227 > raw_cm=6.23 | denoised_cm=25.15 | BLE sent: 25.15
16:58:53.237 > raw_cm=9.21 | denoised_cm=21.01 | BLE sent: 21.01
16:58:53.688 > Server Device Name: BLE_SERVER
16:58:54.268 > raw_cm=NaN | denoised_cm=21.01 | BLE sent: 21.01
16:58:55.247 > raw_cm=8.87 | denoised_cm=16.89 | BLE sent: 16.89
```

4.4 Display Device Signal Processing and Data Receiving

Screenshot of the serial monitor of your client device to show the currently-reading, maximum, and minimum data transmitted from your server device.



The screenshot shows the PlatformIO IDE interface with the Serial Monitor open. The Explorer pane on the left shows the project structure for 'XIAO-BLUETOOTH-CLIENT'. The Serial Monitor displays the following data:

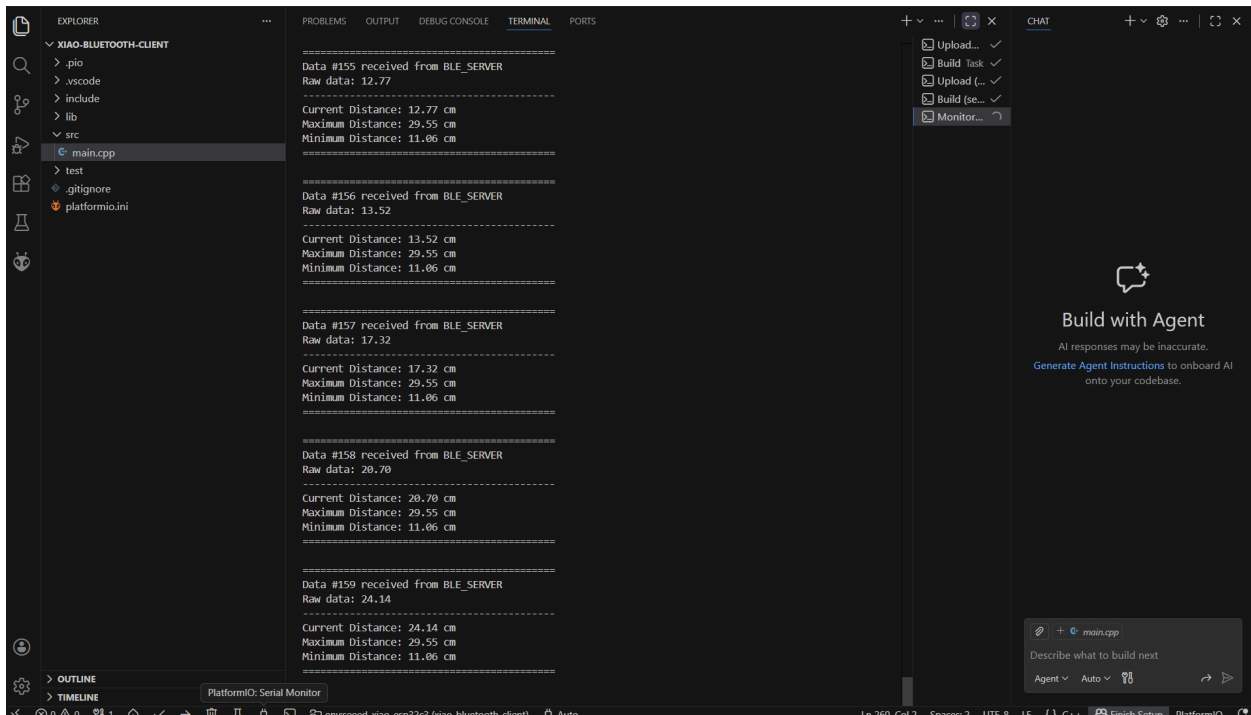
```
=====
Data #124 received from BLE_SERVER
Raw data: 18.23
=====
Current Distance: 18.23 cm
Maximum Distance: 29.55 cm
Minimum Distance: 11.27 cm
=====

Data #125 received from BLE_SERVER
Raw data: 18.68
=====
Current Distance: 18.68 cm
Maximum Distance: 29.55 cm
Minimum Distance: 11.27 cm
=====

Data #126 received from BLE_SERVER
Raw data: 21.46
=====
Current Distance: 21.46 cm
Maximum Distance: 29.55 cm
Minimum Distance: 11.27 cm
=====

Data #127 received from BLE_SERVER
Raw data: 21.85
=====
Current Distance: 21.85 cm
Maximum Distance: 29.55 cm
Minimum Distance: 11.27 cm
=====

Data #128 received from BLE_SERVER
Raw data: 24.16
=====
Current Distance: 24.16 cm
Maximum Distance: 29.55 cm
Minimum Distance: 11.27 cm
=====
```



The screenshot shows the continuation of the Serial Monitor data:

```
=====
Data #155 received from BLE_SERVER
Raw data: 12.77
=====
Current Distance: 12.77 cm
Maximum Distance: 29.55 cm
Minimum Distance: 11.06 cm
=====

Data #156 received from BLE_SERVER
Raw data: 13.52
=====
Current Distance: 13.52 cm
Maximum Distance: 29.55 cm
Minimum Distance: 11.06 cm
=====

Data #157 received from BLE_SERVER
Raw data: 17.32
=====
Current Distance: 17.32 cm
Maximum Distance: 29.55 cm
Minimum Distance: 11.06 cm
=====

Data #158 received from BLE_SERVER
Raw data: 20.70
=====
Current Distance: 20.70 cm
Maximum Distance: 29.55 cm
Minimum Distance: 11.06 cm
=====

Data #159 received from BLE_SERVER
Raw data: 24.14
=====
Current Distance: 24.14 cm
Maximum Distance: 29.55 cm
Minimum Distance: 11.06 cm
=====
```

XIAO-BLUETOOTH-CLIENT

> .pio

> .vscode

> include

> lib

> src

main.cpp

> test

.gitignore

platformio.ini

OUTLINE

TIMELINE

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

Upload...

Build Task

Upload...

Build (se...

Monitor...

Build with Agent

AI responses may be inaccurate.

Generate Agent Instructions to onboard AI onto your codebase.

+ main.cpp

Describe what to build next

Agent Auto

Ln 260, Col 2

Spaces: 2

UTF-8

LF

C++

Finish Setup

PlatformIO

=====

Data #180 received from BLE_SERVER

Raw data: 25.75

=====

Current Distance: 25.75 cm

Maximum Distance: 29.94 cm

Minimum Distance: 11.06 cm

=====

Data #181 received from BLE_SERVER

Raw data: 28.42

=====

Current Distance: 28.42 cm

Maximum Distance: 29.94 cm

Minimum Distance: 11.06 cm

=====

Data #182 received from BLE_SERVER

Raw data: 28.30

=====

Current Distance: 28.30 cm

Maximum Distance: 29.94 cm

Minimum Distance: 11.06 cm

=====

Data #183 received from BLE_SERVER

Raw data: 25.34

=====

Current Distance: 25.34 cm

Maximum Distance: 29.94 cm

Minimum Distance: 11.06 cm

=====

Data #184 received from BLE_SERVER

Raw data: 20.71

=====

Current Distance: 20.71 cm

Maximum Distance: 29.94 cm

Minimum Distance: 11.06 cm

=====