



DesignNews

IoT Device Prototyping with Microchip Curiosity Development Boards

Day 4:

AVR-IoT Cellular Mini Primer

Sponsored by



Webinar Logistics

- Turn on your system sound to hear the streaming presentation.
- If you have technical problems, click “Help” or submit a question asking for assistance.
- Participate in ‘Attendee Chat’ by maximizing the chat widget in your dock.



Fred Eady

Visit 'Lecturer Profile' in your console for more details.

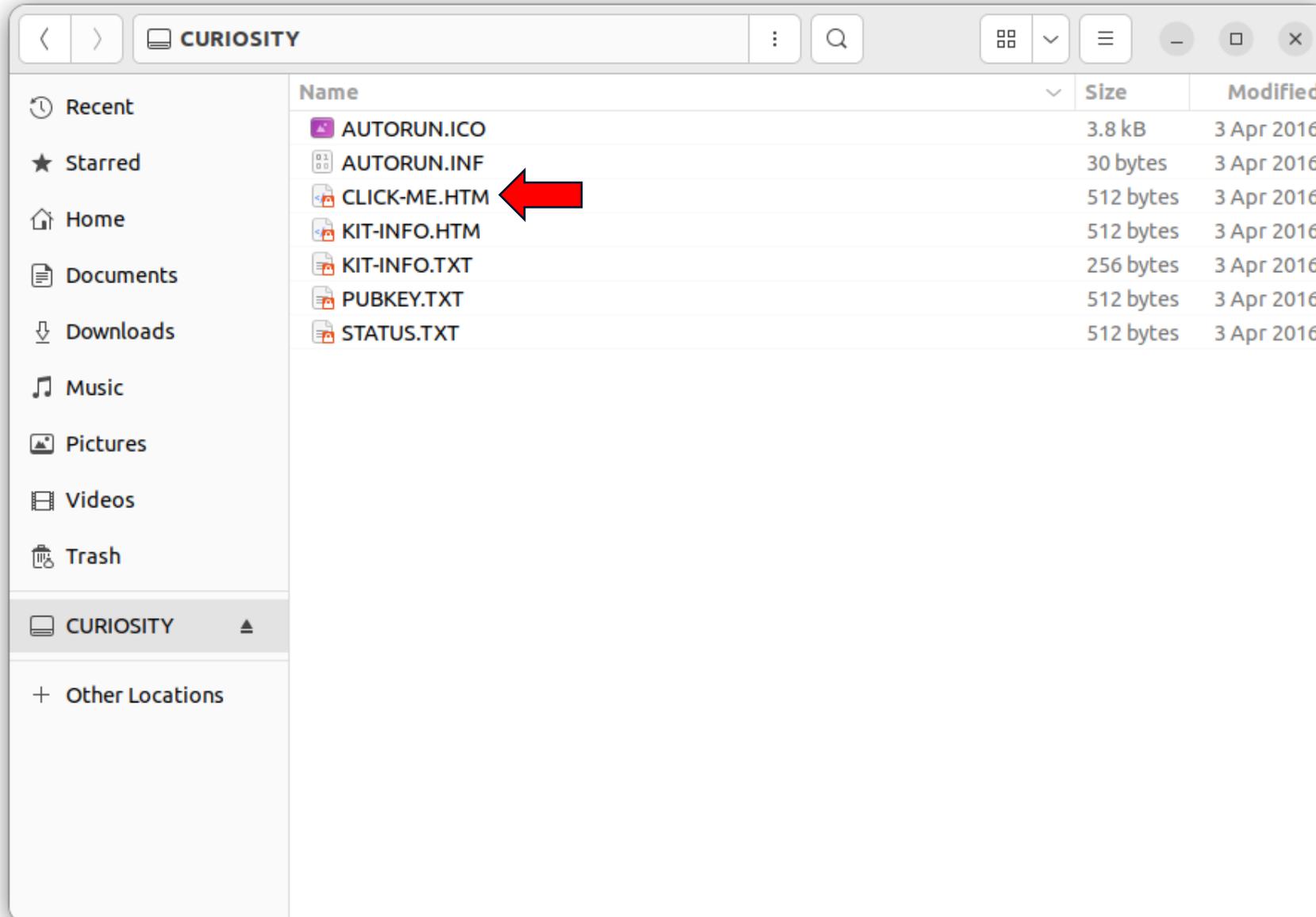
AGENDA

- **Configure Visual Studio Code for Arduino Development**
- **Code an AWS MQTT Arduino Application**

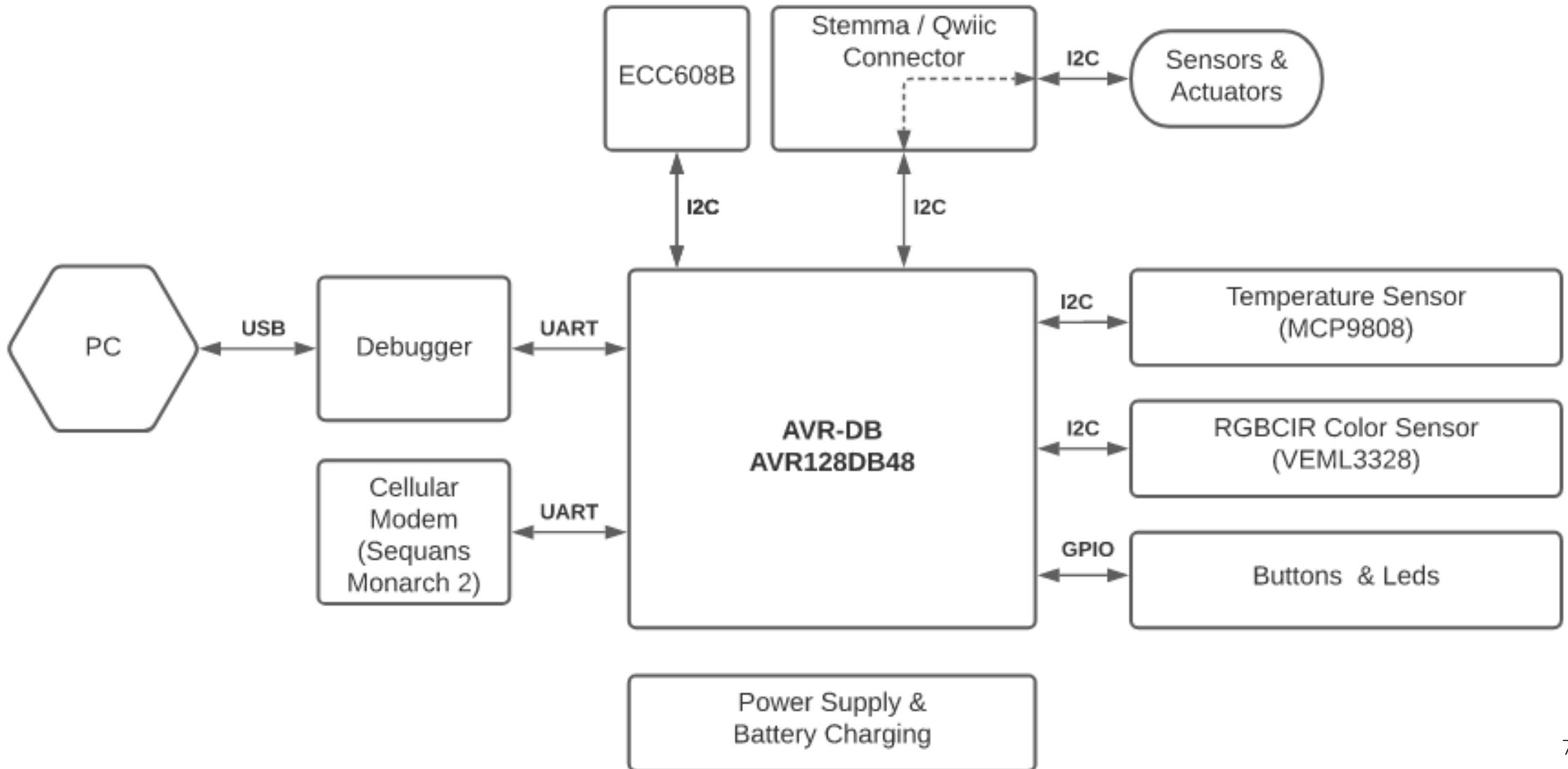




Reset Button

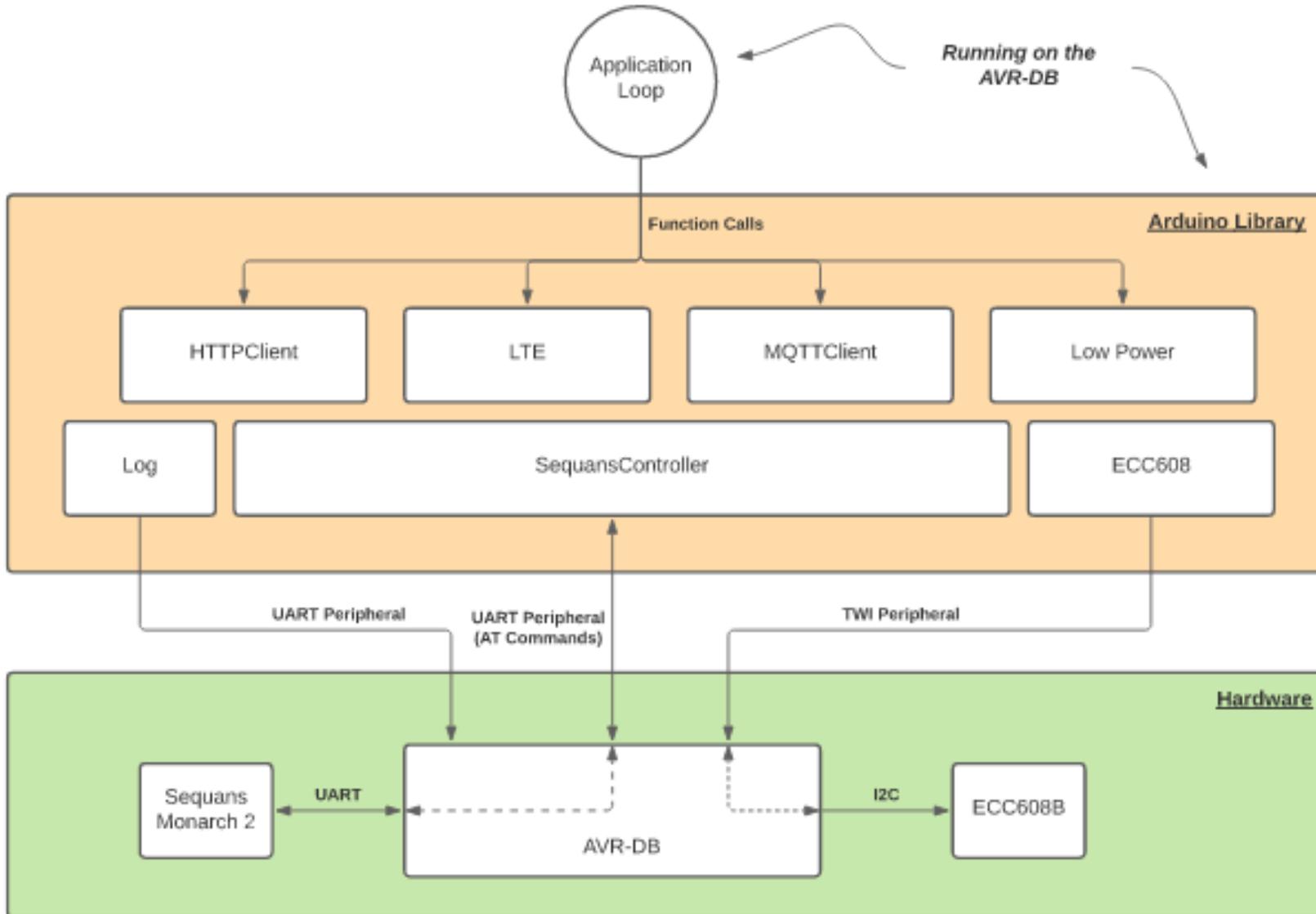


Hardware System Overview





Programmer's System Overview





Install the Arduino Extension for Visual Studio Code

Legacy IDE (1.8.X)



Arduino IDE 1.8.19

The open-source Arduino Software (IDE) makes it easy to write and upload it to the board. This software can be used with any Arduino board.

Refer to the [Getting Started](#) page for Installation instructions.

The screenshot shows the Visual Studio Code extension marketplace interface. At the top, there are tabs for 'Settings' and 'Extension: Arduino'. The main content area displays the 'Arduino' extension by Microsoft, version v0.4.12, marked as a 'Preview'. It has 1,536,526 downloads and a 4.5-star rating from 99 reviews. The extension is currently installed, with buttons for 'Disable', 'Uninstall', and a settings gear icon. Below the extension name, it says 'This extension is enabled globally.' There are tabs for 'Details', 'Feature Contributions', 'Changelog', 'Dependencies', and 'Runtime Status'. The 'Details' tab is active, showing the title 'Visual Studio Code extension for Arduino' and a 'chat on gitter' button. The main text welcomes users to the extension and lists its features: IntelliSense and syntax highlighting for Arduino sketches, the ability to verify and upload sketches, a built-in board and library manager, a built-in example list, and a built-in serial monitor. On the right side, there are sections for 'Categories' (Programming Languages, Debuggers, Snippets, Formatters), 'Extension Resources' (Marketplace Repository, License, Microsoft), and 'More Info'.



Install the DxCore



DxCore by Spence Konde Version 1.4.10 **INSTALLED**

Boards included in this package:

AVR DA-series: AVR128DA28, AVR128DA32, AVR128DA48, AVR128DA64, AVR64DA28, AVR64DA32, AVR64DA48, AVR64DA64, AVR32DA28, AVR32DA32, AVR32DA48
 AVR DB-series: AVR128DB28, AVR128DB32, AVR128DB48, AVR128DB64, AVR64DB28, AVR64DB32, AVR64DB48, AVR64DB64, AVR32DB28, AVR32DB32, AVR32DB48
 AVR DD-series support planned pending availability. Works with Microchip Curiosity Nano boards via the builtin USB port.
 1.4.10: The latest critical bugfix for 1.4.x issues - this should finally make the new attachInterrupt implementation work.
 1.4.0: A HUGE update bringing the total rewrite of Wire.h with master+slave and dual mode support, major serial improvements (including half-duplex and RS485), and loads of additional features and fixes. See the documentation for the full list!, Supported UPDI programmers: SerialUPDI (serial adapter w/diode or resistor), jtag2updi, nEDBG, mEDBG, EDBG, SNAP, Atmel-ICE and PICKIT4 - or use one of those to load Optiboot (included) for serial programming., DxCore 1.5.0 is not yet available, but upon release, it will be auge update adding support for the entire DD product line as well as a shedload of fixes.

[More info](#)

1.4.10

Install

Remove

Arduino configuration

Arduino: Additional Urls

Additional URLs for 3rd party packages.

https://github.com/stm32duino/BoardManagerFiles/raw/main/package_stmicroelectronics_index.json

https://github.com/earlephilhower/arduino-pico/releases/download/global/package_rp2040_index.json

http://drazzy.com/package_drazzy.com_index.json

Add Item

Settings

Search set

User Workspace

Commonly Used

> Text Editor

> Workbench

> Window

> Features

> Application

> Security

∨ Extensions

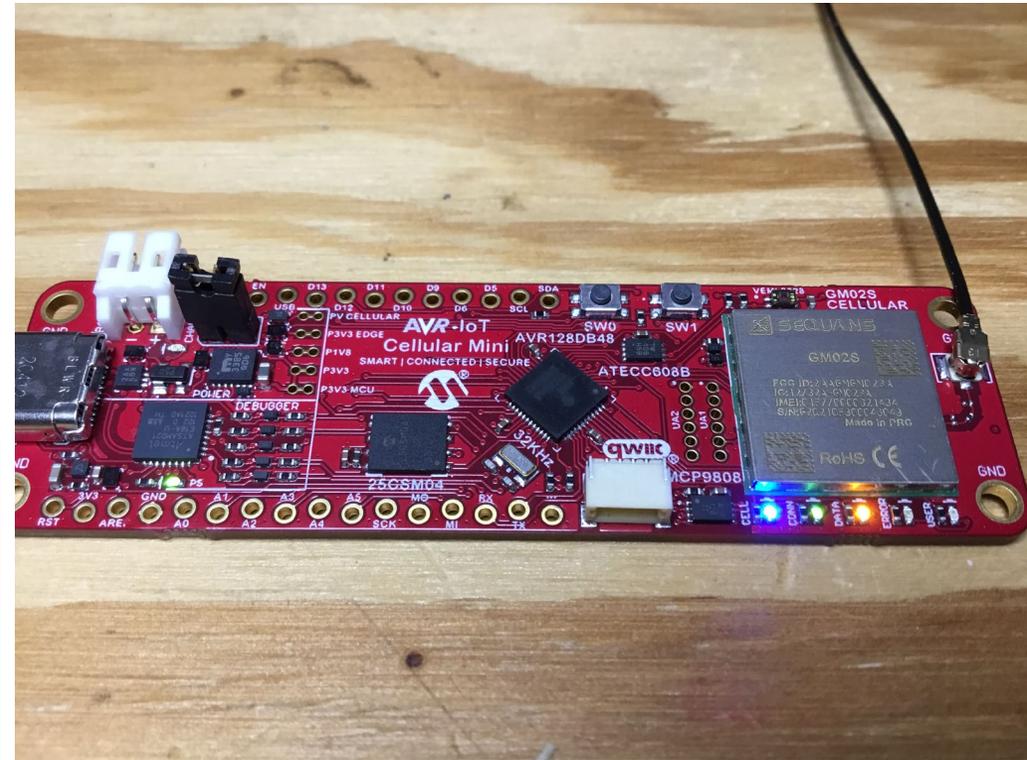
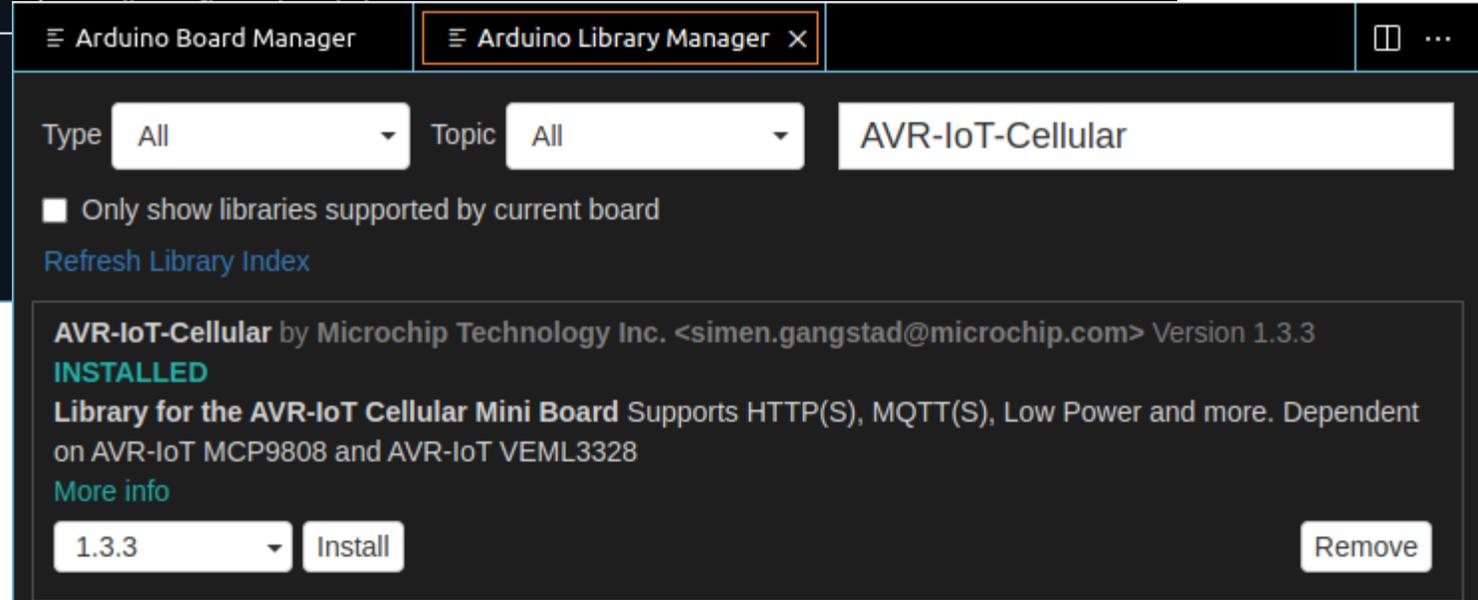
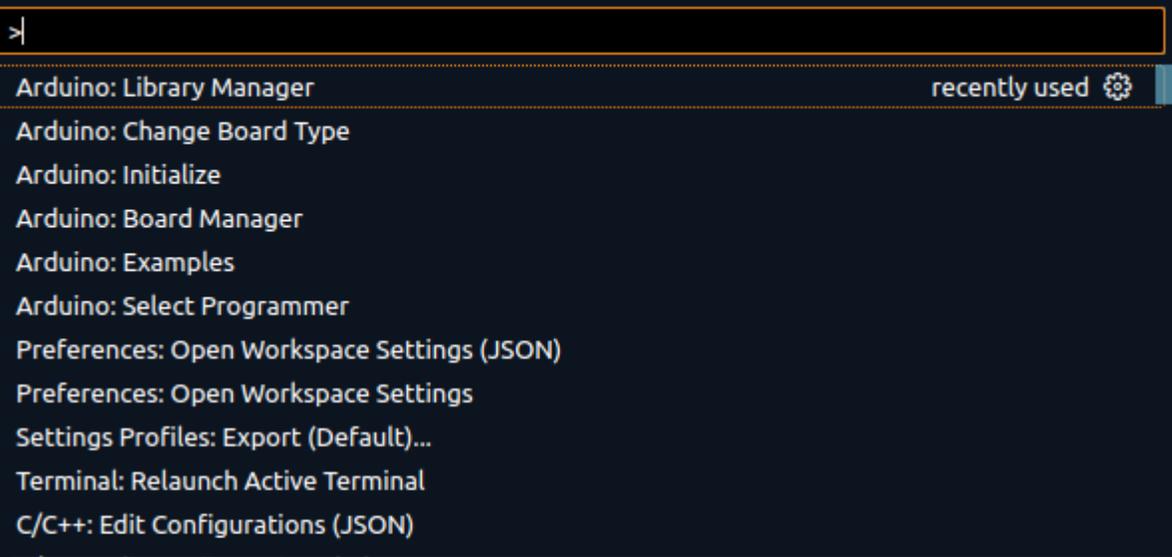
.ipynb Support

.NET Install Tool

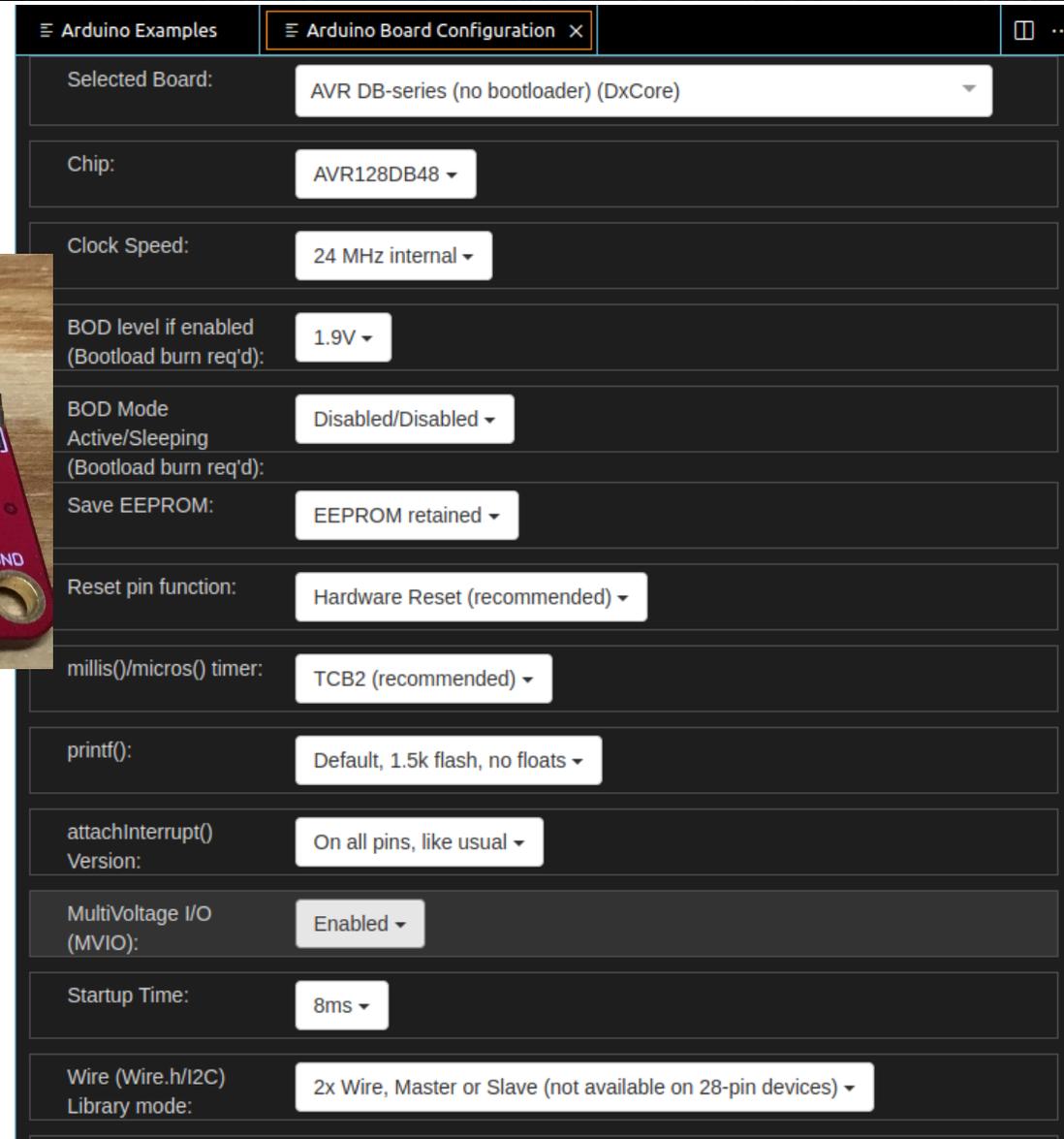
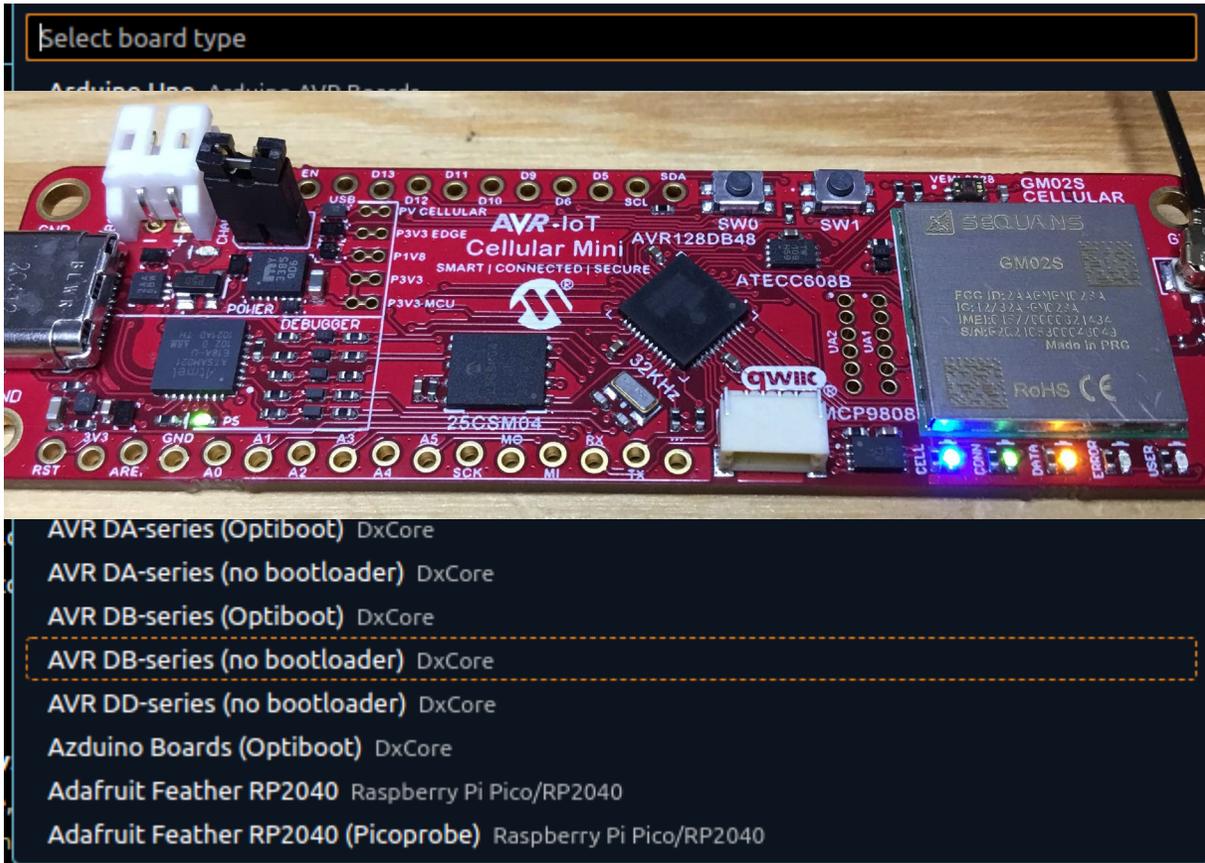
Arduino configuration



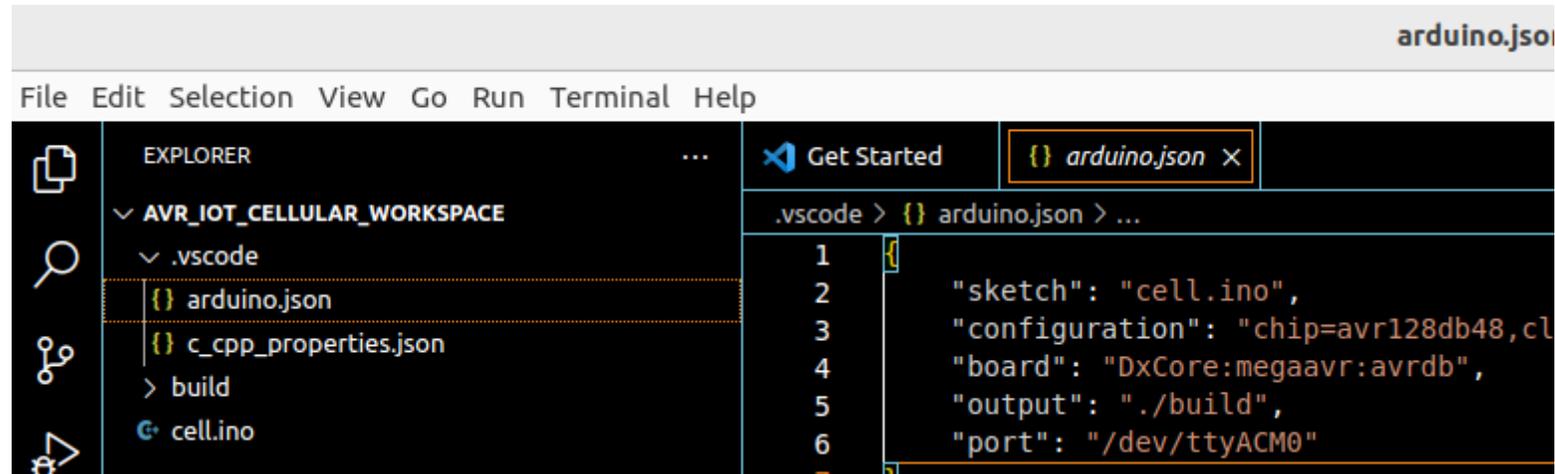
Install the AVR-IoT Cellular Library



Configure the Arduino Development Target



Create a build Directory



```
arduino.json

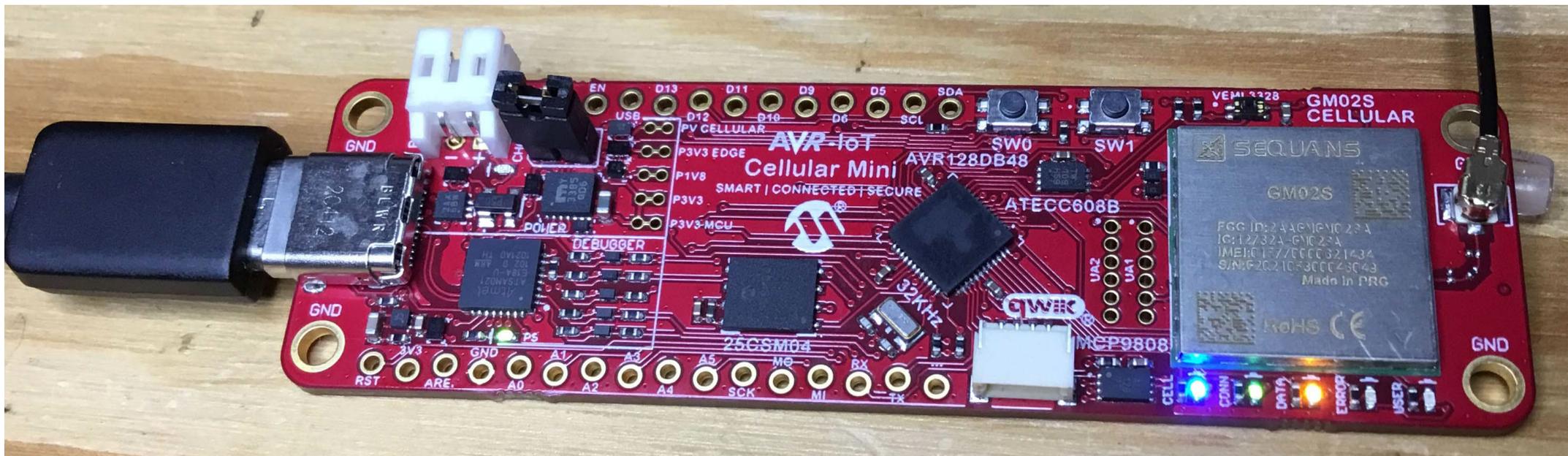
File Edit Selection View Go Run Terminal Help

EXPLORER
AVR_IOT_CELLULAR_WORKSPACE
  .vscode
    {} arduino.json
    {} c_cpp_properties.json
  > build
  cell.ino

Get Started {} arduino.json x

.vscode > {} arduino.json > ...

1 {}
2 "sketch": "cell.ino",
3 "configuration": "chip=avr128db48,cl
4 "board": "DxCore:megaavr:avrdb",
5 "output": "./build",
6 "port": "/dev/ttyACM0"
```



Includes – Variables - Definitions

```
1  /*******  
2  /** AVR-IoT Cellular Mini Driver  
3  /** Written by Fred Eady for CEC December 2022  
4  /** Rev 1.00 A  
5  /** Last Update 11/17/2022  
6  /*******  
7  
8  #include <Arduino.h>  
9  #include <ecc608.h>  
10 #include <led_ctrl.h>  
11 #include <log.h>  
12 #include <lte.h>  
13 #include <mqtt_client.h>  
14 #include <mcp9808.h>  
15 // AWS topic format  
16 #define MQTT_SUB_TOPIC_FMT "%s/sensors"  
17 #define MQTT_PUB_TOPIC_FMT "%s/sensors"  
18 // Temperature Variables  
19 int8_t temF = 0;  
20 uint16_t res = 0; //resolution  
21 // SUB-PUB Buffers  
22 char mqtt_sub_topic[128];  
23 char mqtt_pub_topic[128];  
24 char temMsg[16]; // MCP9808 message buffer
```

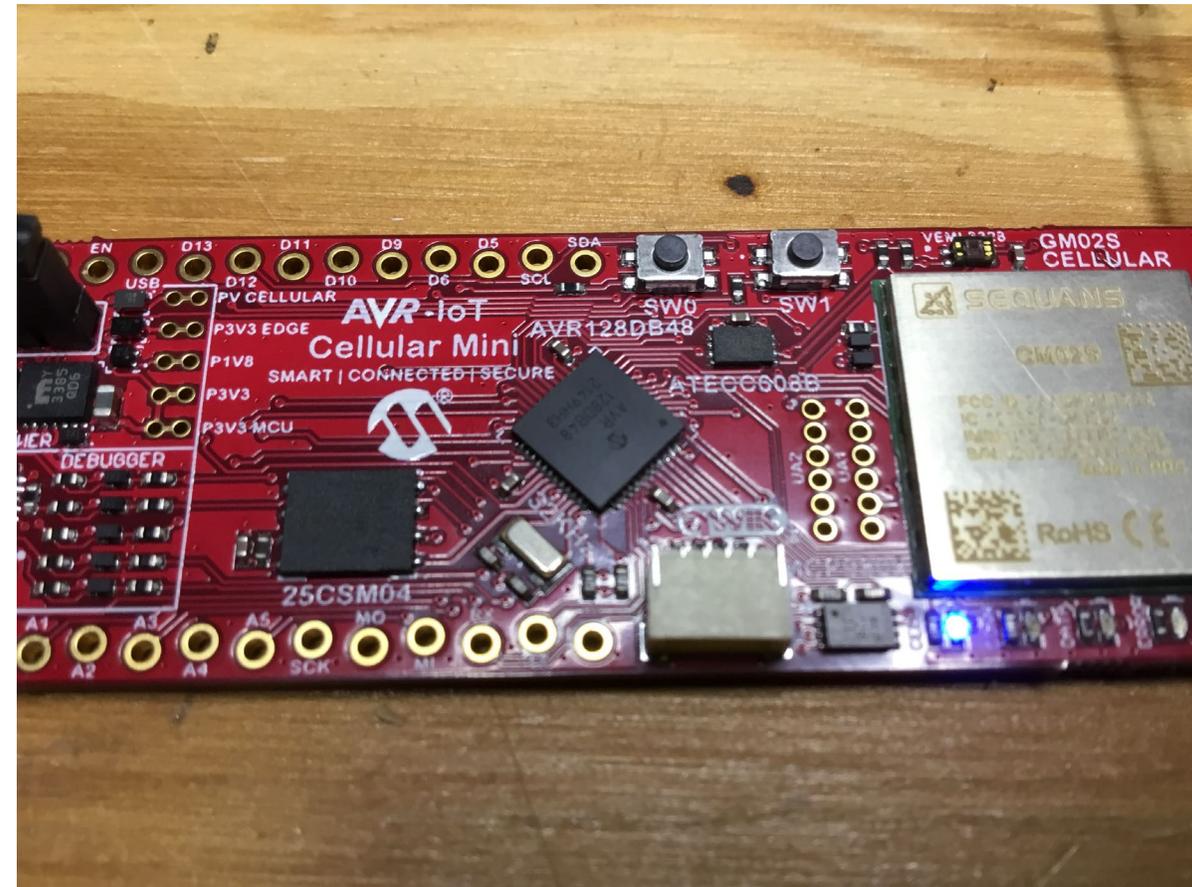
Initialize MQTT Topics Function

```
26  bool initMQTTTopics()
27  {
28      ECC608.begin();
29
30      // Find the thing ID and set the publish and subscription topics
31      uint8_t thingName[128];
32      uint8_t thingNameLen = sizeof(thingName);
33
34      // -- Get the thingname
35      uint8_t err = ECC608.getThingName(thingName, &thingNameLen);
36      if (err != ECC608.ERR_OK)
37      {
38          Log.error("Could not retrieve thingname from the ECC");
39          return false;
40      }
41
42      sprintf(mqtt_sub_topic, MQTT_SUB_TOPIC_FMT, thingName);
43      sprintf(mqtt_pub_topic, MQTT_PUB_TOPIC_FMT, thingName);
44      return true;
45  }
```



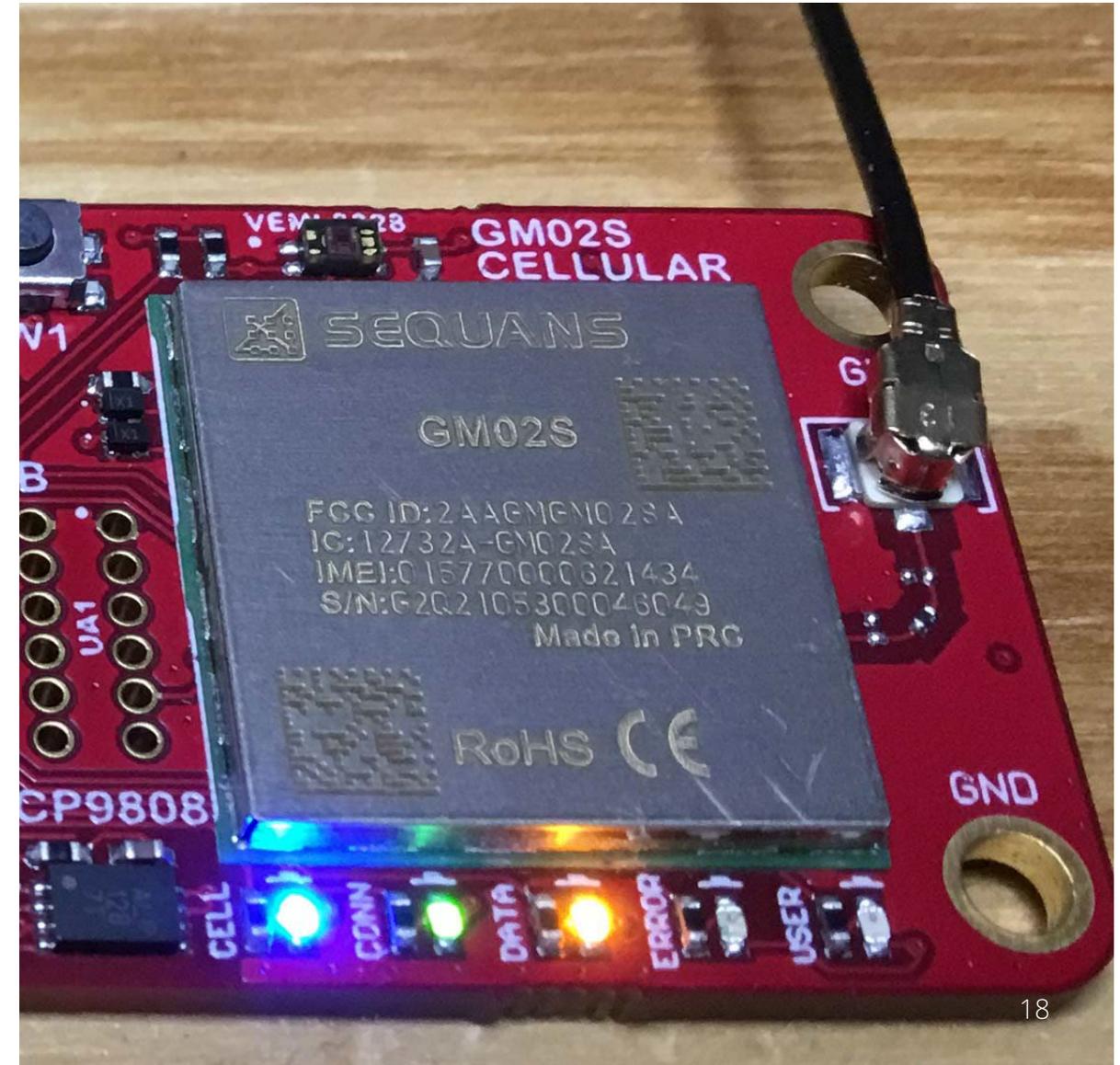
Connect to the Network

```
47 void setup()
48 {
49   Log.begin(115200);
50   LedCtrl.begin();
51   LedCtrl.startupCycle();
52   Mcp9808.begin();
53
54   Log.info("Starting MQTT Temperature App for AWS\r\n");
55
56   if (initMQTTTopics() == false)
57   {
58     Log.error("Unable to initialize the MQTT topics. Stopping...");
59     while (1) {}
60   }
61
62   if (!Lte.begin())
63   {
64     Log.error("Failed to connect to operator");
65
66     // Halt here
67     while (1) {}
68   }
69 }
```



Connect to the MQTT Broker

```
70 // Attempt to connect to the broker
71 if (MqttClient.beginAWS())
72 {
73     Log.infof("Connecting to broker");
74
75     while (!MqttClient.isConnected())
76     {
77         Log.rawf(".");
78         delay(500);
79     }
80
81     Log.rawf(" OK!\r\n");
82
83     // Subscribe to the topic
84     MqttClient.subscribe(mqtt_sub_topic);
85 }
86 else
87 {
88     Log.rawf("\r\n");
89     Log.error("Failed to connect to broker");
90
91     // Halt here
92     while (1) {}
93 }
```

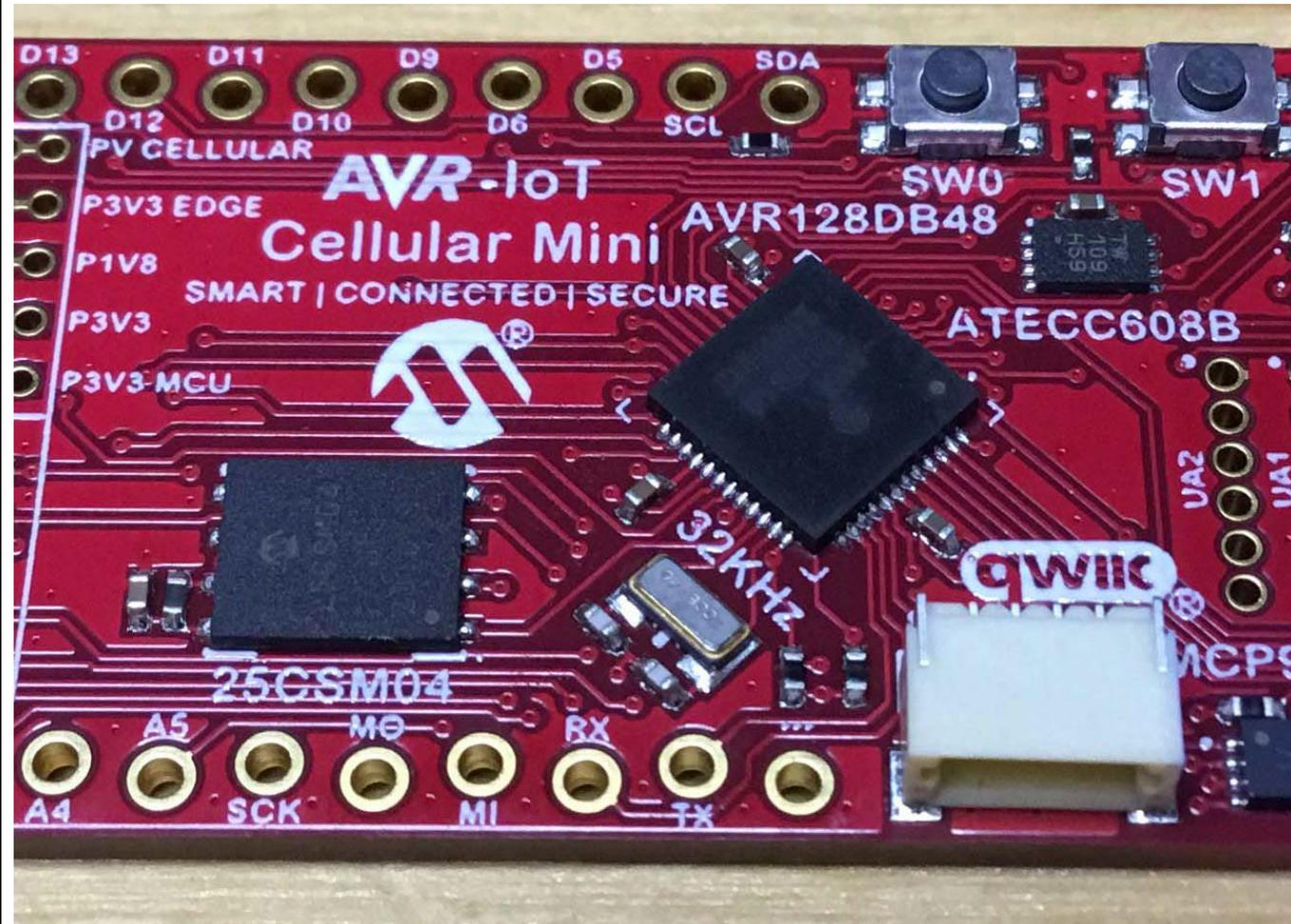


MQTT Publish and Receive

```

95 // MQTT publish and receive
96 for (uint8_t i = 0; i < 1; i++)
97 {
98   temF = (int)Mcp9808.readTempF();
99   sprintf(temMsg, "Temp (*F): %d", temF);
100
101   bool published_successfully =
102     MqttClient.publish(mqtt_pub_topic, temMsg);
103
104   if (published_successfully)
105   {
106     Log.info("Published message");
107   }
108   else
109   {
110     Log.error("Failed to publish\r\n");
111   }
112
113   delay(2000);
114
115   String message = MqttClient.readMessage(mqtt_sub_topic);
116
117   // Read message will return an empty string if there were no new
118   // messages, so anything other than that means that there were a
119   // new message
120   if (message != "")
121   {
122     Log.infof("Got new message: %s\r\n", message.c_str());
123   }
124
125   delay(2000);
126 }
127
128 Log.info("Closing MQTT connection");
129 MqttClient.end();
130 }

```





Setup MPLAB Data Visualizer

The screenshot shows the MPLAB IDE interface with the Data Visualizer tool configured for the AVR-IoT Cellular Mini project. The workspace contains a 'Terminal' and an 'XY Plot' window. The 'Connections' pane shows the 'AVR-IoT Cellular Mini' project with 'DGI' enabled. The 'Variable Streamers' pane shows 'Serial Ports' expanded, with 'ttyACM0' selected. The 'Dashboard Widgets' pane shows 'ttyACM0 Settings' with the following configuration:

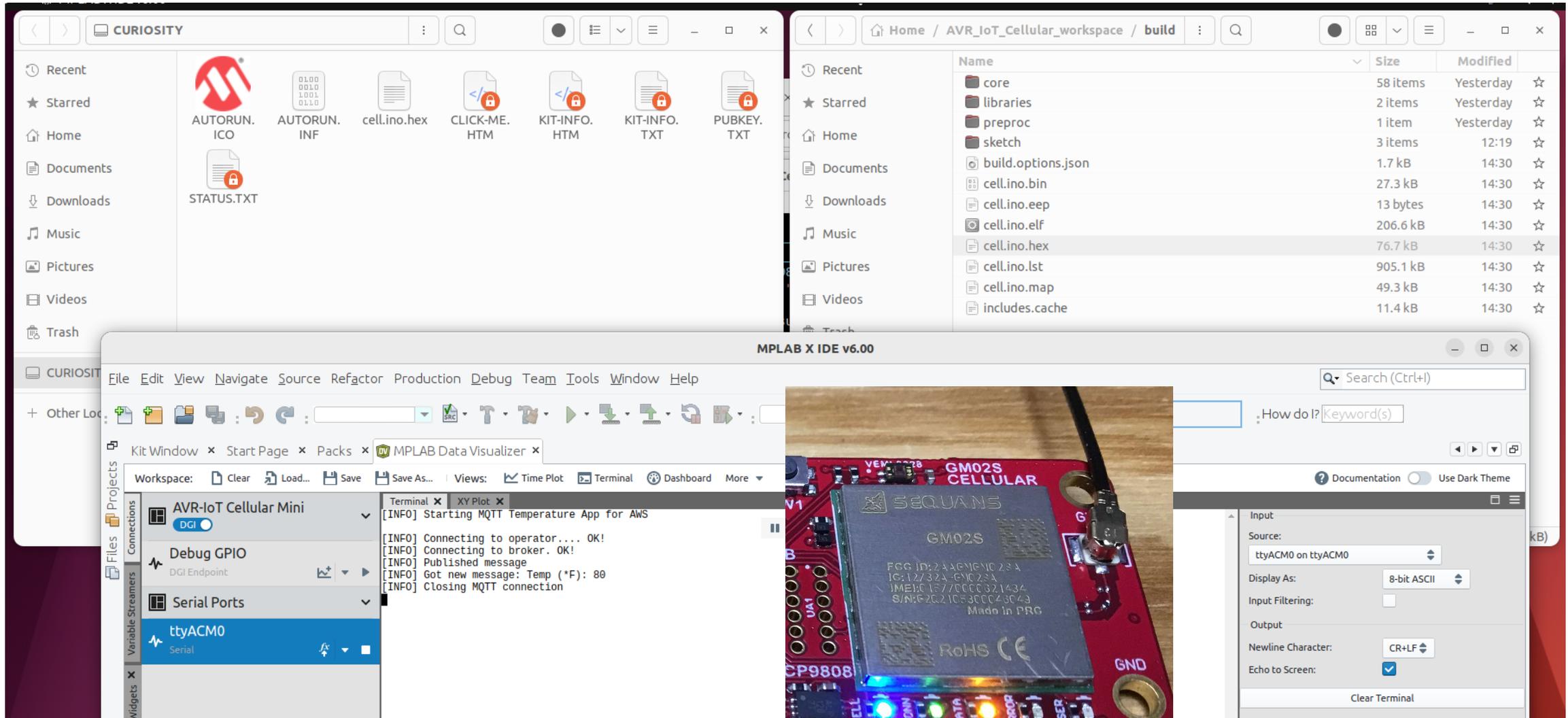
- Baud Rate: 115200
- Char Length: 8 bits
- Parity: None
- Stop Bits: 1 bit

The screenshot shows the configuration dialog for the Data Visualizer. The 'Input' section is expanded, showing the following settings:

- Source: ttyACM0 on ttyACM0
- Display As: 8-bit ASCII
- Input Filtering:
- Output section:
- Newline Character: CR+LF
- Echo to Screen:

A 'Clear Terminal' button is located at the bottom of the dialog.

Execute the Arduino MQTT Temperature Application



The image displays a development environment for an AVR-IoT Cellular Mini board. It includes a file explorer showing project files, the MPLAB X IDE v6.00 interface with a terminal window, and a photograph of the physical board with its components.

File Explorer (Left): Shows the project directory structure with files like `AUTORUN.ICO`, `AUTORUN.INF`, `cell.ino.hex`, `CLICK-ME.HTM`, `KIT-INFO.HTM`, `KIT-INFO.TXT`, `PUBKEY.TXT`, and `STATUS.TXT`.

File Explorer (Right): Shows the `build` directory contents:

Name	Size	Modified
core	58 items	Yesterday
libraries	2 items	Yesterday
preproc	1 item	Yesterday
sketch	3 items	12:19
build.options.json	1.7 kB	14:30
cell.ino.bin	27.3 kB	14:30
cell.ino.eep	13 bytes	14:30
cell.ino.elf	206.6 kB	14:30
cell.ino.hex	76.7 kB	14:30
cell.ino.lst	905.1 kB	14:30
cell.ino.map	49.3 kB	14:30
includes.cache	11.4 kB	14:30

MPLAB X IDE v6.00 (Center): The terminal window shows the following output:

```
[INFO] Starting MQTT Temperature App for AWS
[INFO] Connecting to operator... OK!
[INFO] Connecting to broker. OK!
[INFO] Published message
[INFO] Got new message: Temp (*F): 80
[INFO] Closing MQTT connection
```

The IDE interface also shows a "Kit Window" with "AVR-IoT Cellular Mini" selected, and a "Serial Ports" window with "ttyACM0" selected.

Physical Device (Bottom Center): A photograph of the AVR-IoT Cellular Mini board. The board is red and features a "SEQUANS GM02S CELLULAR" module. The module's label includes: FCG ID: 2AAGMCNC 2SA, IC: 12732A-GMC 2SA, JMEHC 15770000321434, S/N: F2C210530CC43043, Made in PRG, and RoHS CE. The board has several LEDs labeled CELL, CONN, DATA, ERROR, and USER, and a GND pin.

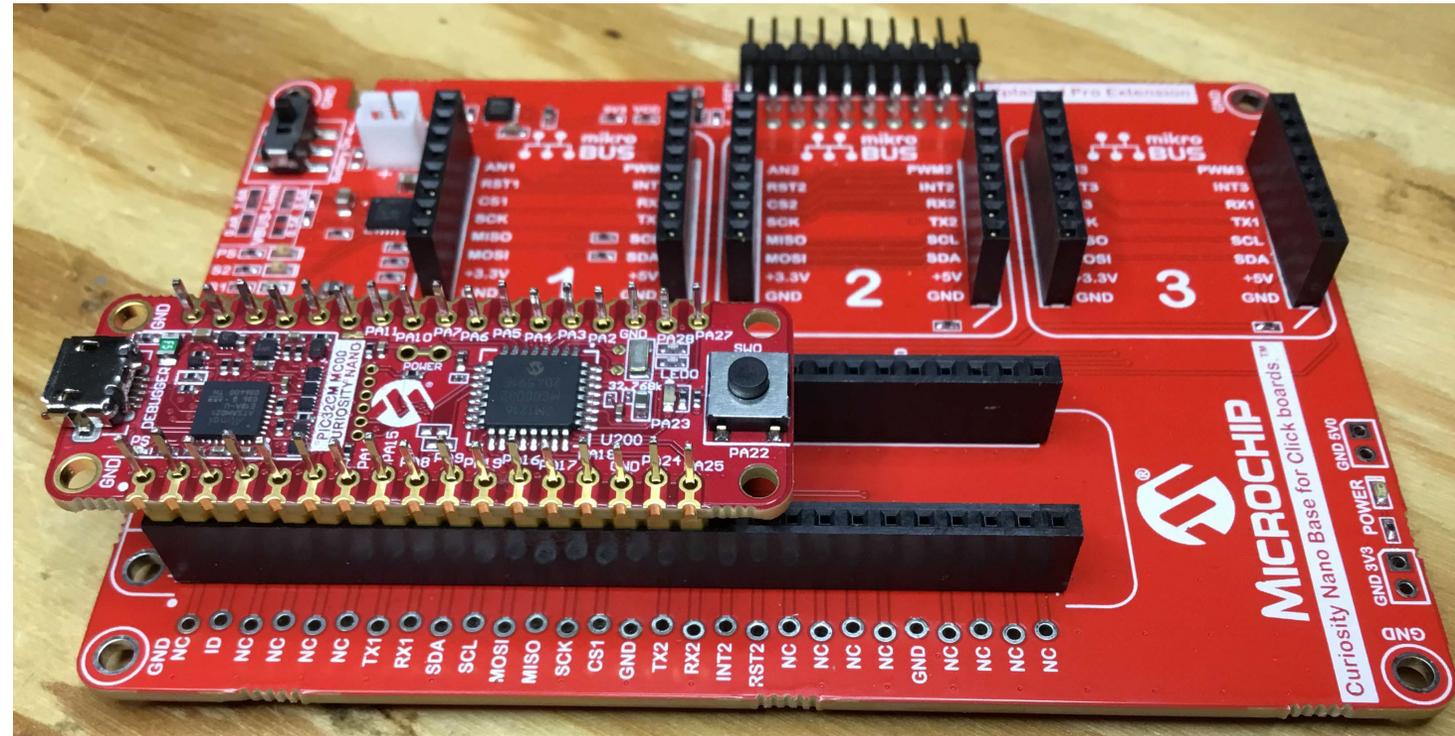
Terminal Window (Right): Shows the serial terminal configuration for "ttyACM0 on ttyACM0". The "Display As" is set to "8-bit ASCII" and "Echo to Screen" is checked.

MORE TO COME..

Thank you for attending!!!

Please consider the resources below:

- [microchip.com](https://www.microchip.com)
- **MPLAB Data Visualizer User Guide**
- **AVR-IoT Cellular Mini User Guide**
- **Visual Studio Code Extensions**





Thank You

Sponsored by

