

eLearning

Wi-Fi[®] SoC Module *WFI32E01* 參考範例應用篇



A Leading Provider of Smart, Connected and Secure Embedded Control Solutions



SMART | CONNECTED | SECURE

King Chou, Pr. ESE

2021



Microchip - Chinese [Traditional]

199 位訂閱者

已訂閱



首頁

影片

播放清單

頻道

討論

簡介



上傳的影片 ▶ 全部播放

Switch-Node Layout

1:34:49

ePOW007—電源分佈網路與電源完整性

觀看次數：29次 · 3 週前

練習5: 为你的ATSAMC21G17A 加入CAN的傳送功能

43:03

ATSAMC21G17A 101—跟著Microchip使用APP Nano...

觀看次數：97次 · 1 個月前

練習4: 加入MCP9800溫度Sensor的讀取並顯示於OLED第三行

29:43

ATSAMC21G17A 101—跟著Microchip使用APP Nano...

觀看次數：28次 · 1 個月前

SSD1306 GDDRAM的放大圖

每一Page有128個Bytes的資料，控制位28*8個點，總共有8個Pages = 128 * 64點！

38:25

ATSAMC21G17A 101—跟著Microchip使用APP Nano...

觀看次數：31次 · 1 個月前

練習2: 中斷操作—EIC以及TC的中斷操作法 使用PLUS & callback

53:47

ATSAMC21G17A 101—跟著Microchip使用APP Nano...

觀看次數：39次 · 1 個月前

練習3: 中斷操作—EIC以及TC的中斷操作法 使用PLUS & callback

28:10

ATSAMC21G17A 101—跟著Microchip使用APP Nano...

觀看次數：27次 · 1 個月前

Featured Channels



Microchip Technology

5.38萬 位訂閱者



Microchip - Chinese [Simplified]

540 位訂閱者



Microchip - Japanese

343 位訂閱者



Microchip - Korean

1310 位訂閱者



Microchip Makes

1.87萬 位訂閱者

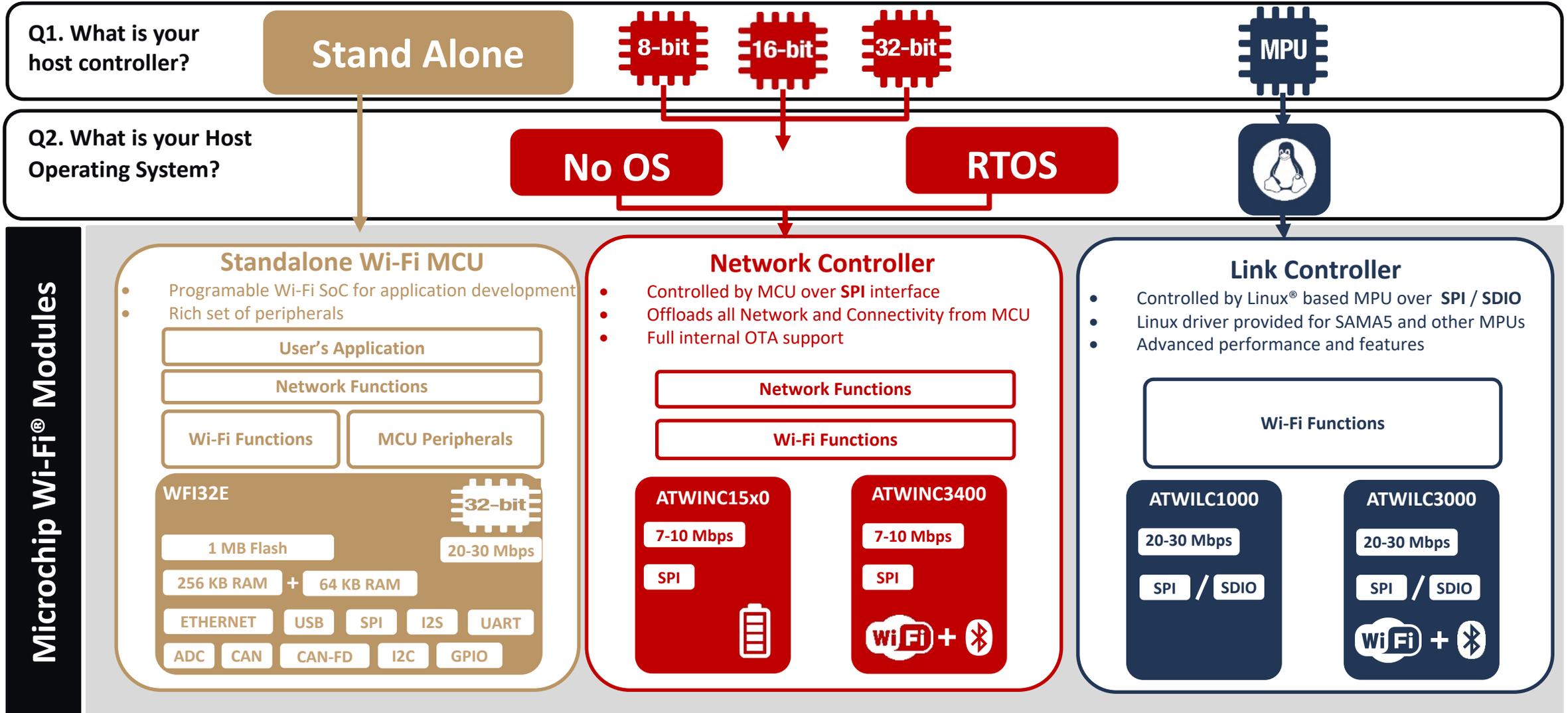
eLearning

Wi-Fi® SoC Module *WFI32E01* 介紹及軟體應用篇



A Leading Provider of Smart, Connected and Secure Embedded Control Solutions

Microchip Wi-Fi® Overview

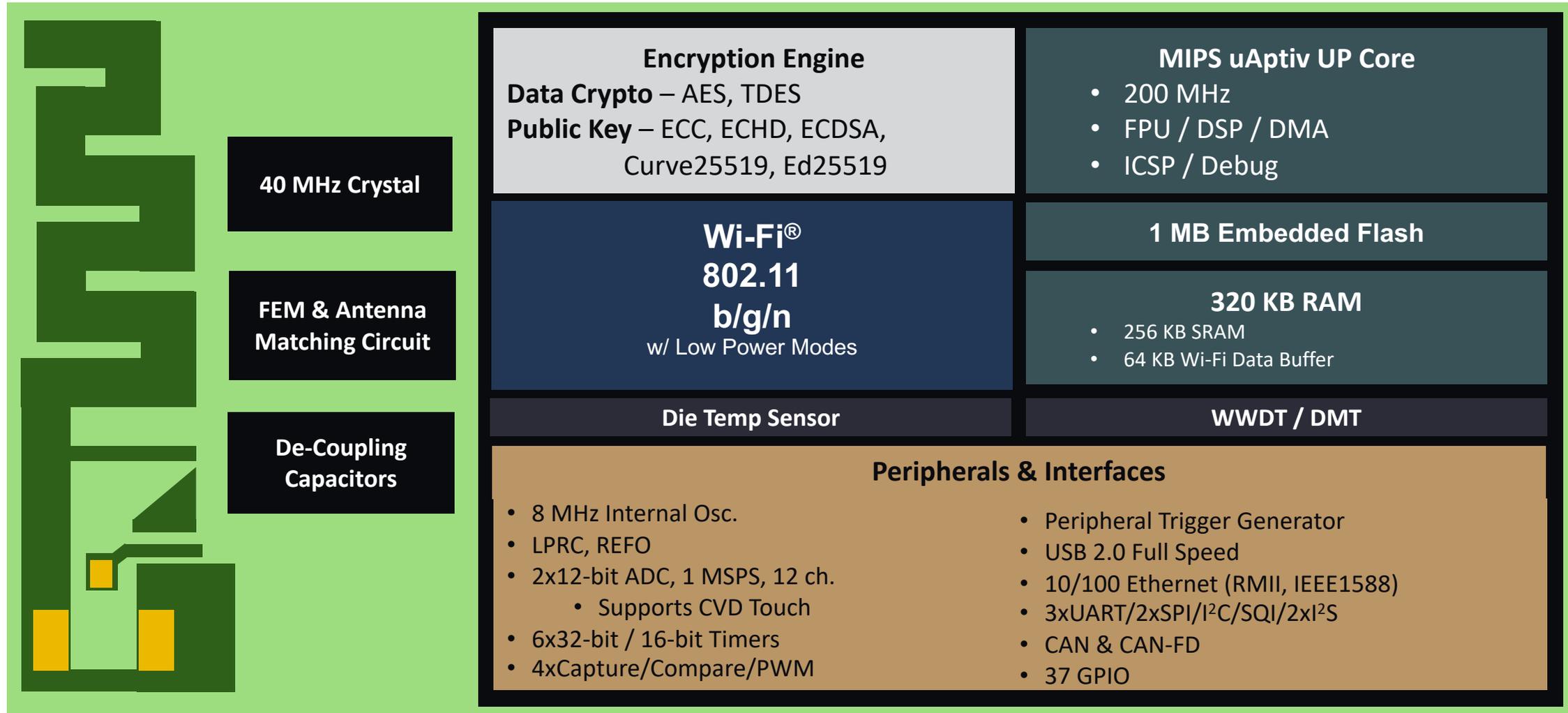


WFI32E01UE

Block Diagram for 54-Pad Module

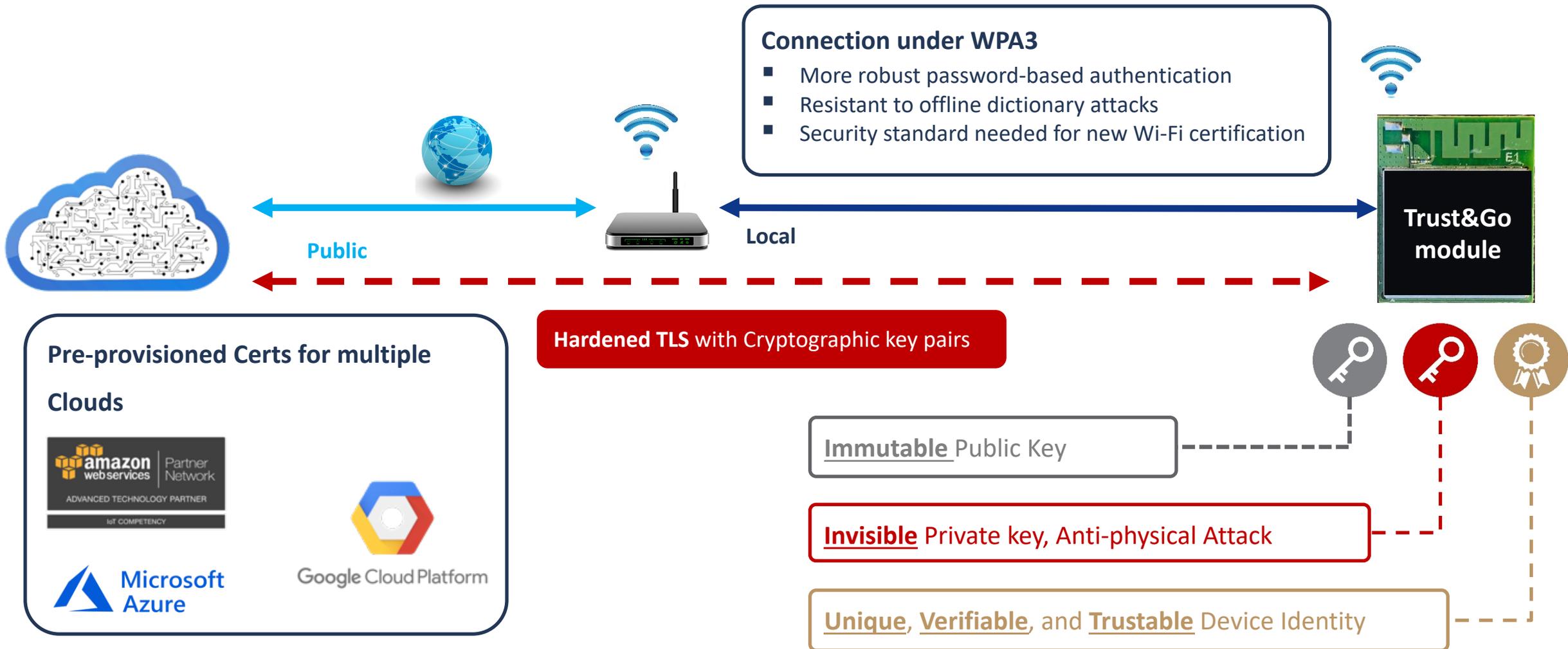


- **Module Package:**
 - 54 pins SMD
 - 24.5 x 20.5 x 2.5 mm
- **Single Supply Voltage**
 - Operating Voltage: 3.0V to 3.6V
- **Operating Temperature: -40°C to +85 °C**



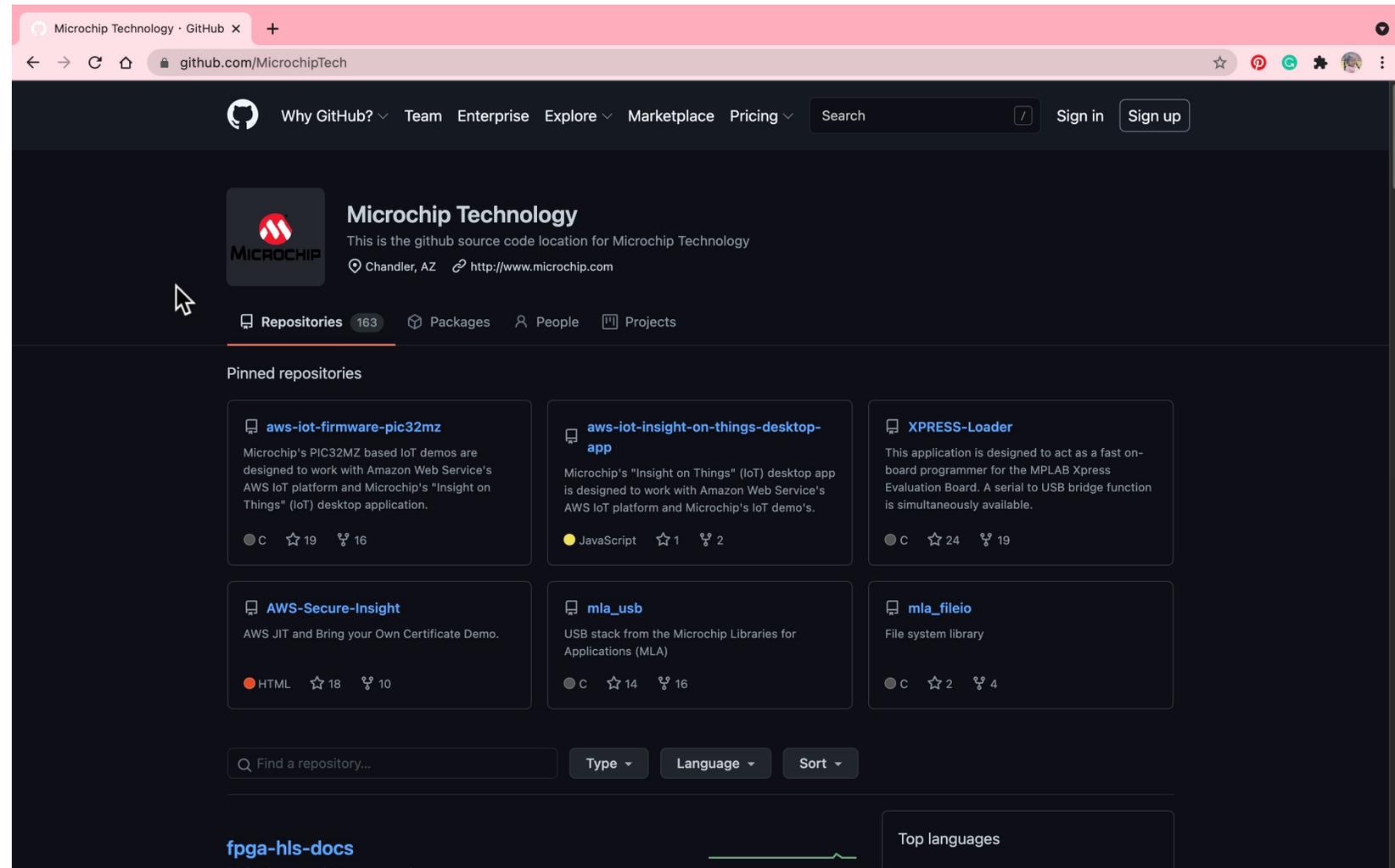
Build a Secure and Trustful IoT Connection

Trust&Go Wi-Fi® module: WFI32E01PC-I



PIC32MZW1_Workshop

- https://github.com/MicrochipTech/PIC32MZW1_Workshop



The screenshot shows the GitHub profile page for Microchip Technology. The browser address bar displays "github.com/MicrochipTech". The profile header includes the Microchip logo, the name "Microchip Technology", and a bio: "This is the github source code location for Microchip Technology". It also lists the location "Chandler, AZ" and the website "http://www.microchip.com". Navigation tabs for "Repositories" (163), "Packages", "People", and "Projects" are visible. The "Pinned repositories" section features six items:

- aws-iot-firmware-pic32mz**: Microchip's PIC32MZ based IoT demos are designed to work with Amazon Web Service's AWS IoT platform and Microchip's "Insight on Things" (IoT) desktop application. (19 stars, 16 forks)
- aws-iot-insight-on-things-desktop-app**: Microchip's "Insight on Things" (IoT) desktop app is designed to work with Amazon Web Service's AWS IoT platform and Microchip's IoT demo's. (1 star, 2 forks)
- XPRESS-Loader**: This application is designed to act as a fast on-board programmer for the MPLAB Xpress Evaluation Board. A serial to USB bridge function is simultaneously available. (24 stars, 19 forks)
- AWS-Secure-Insight**: AWS JIT and Bring your Own Certificate Demo. (18 stars, 10 forks)
- m1a_usb**: USB stack from the Microchip Libraries for Applications (MLA). (14 stars, 16 forks)
- m1a_fileio**: File system library. (2 stars, 4 forks)

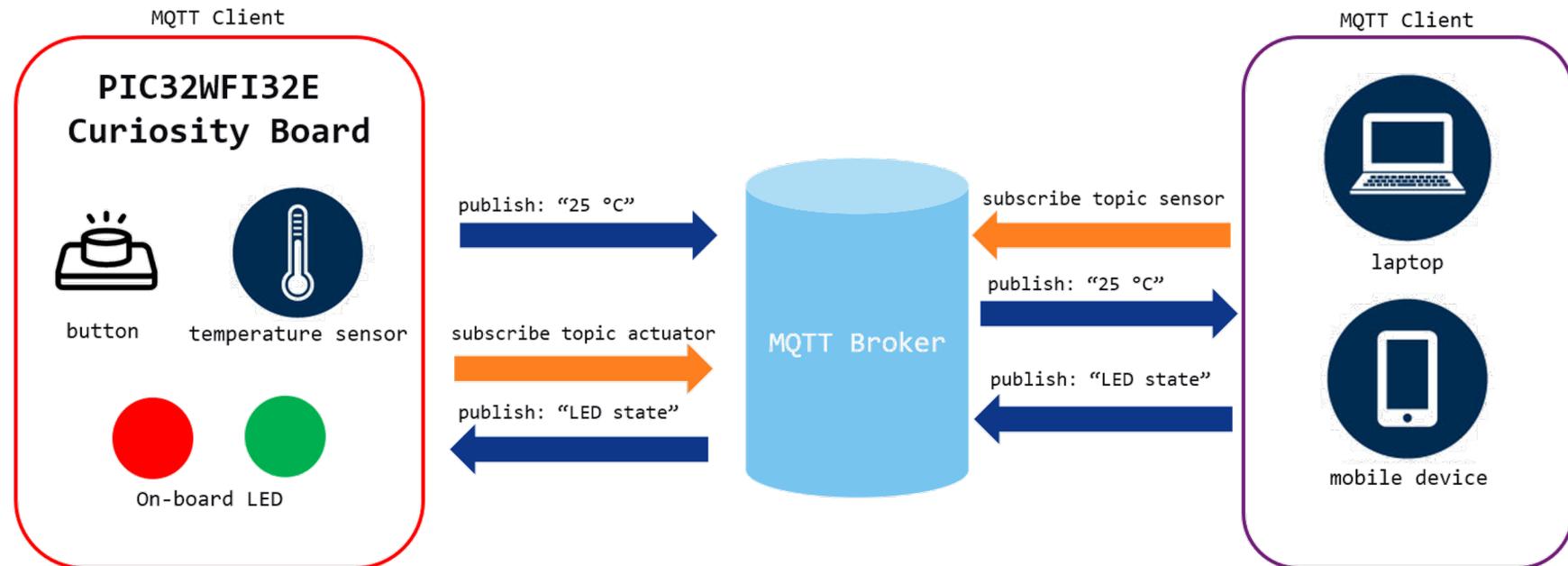
At the bottom, there is a search bar "Find a repository...", filters for "Type", "Language", and "Sort", and a "Top languages" section.

- **Reference Example Codes**
 - Develop a Secured Smart Home Application from scratch
 - Wi-Fi[®] Touch and OLED Display
 - Wi-Fi Provisioning over BLE
 - Wi-Fi to CAN Bus Bridge

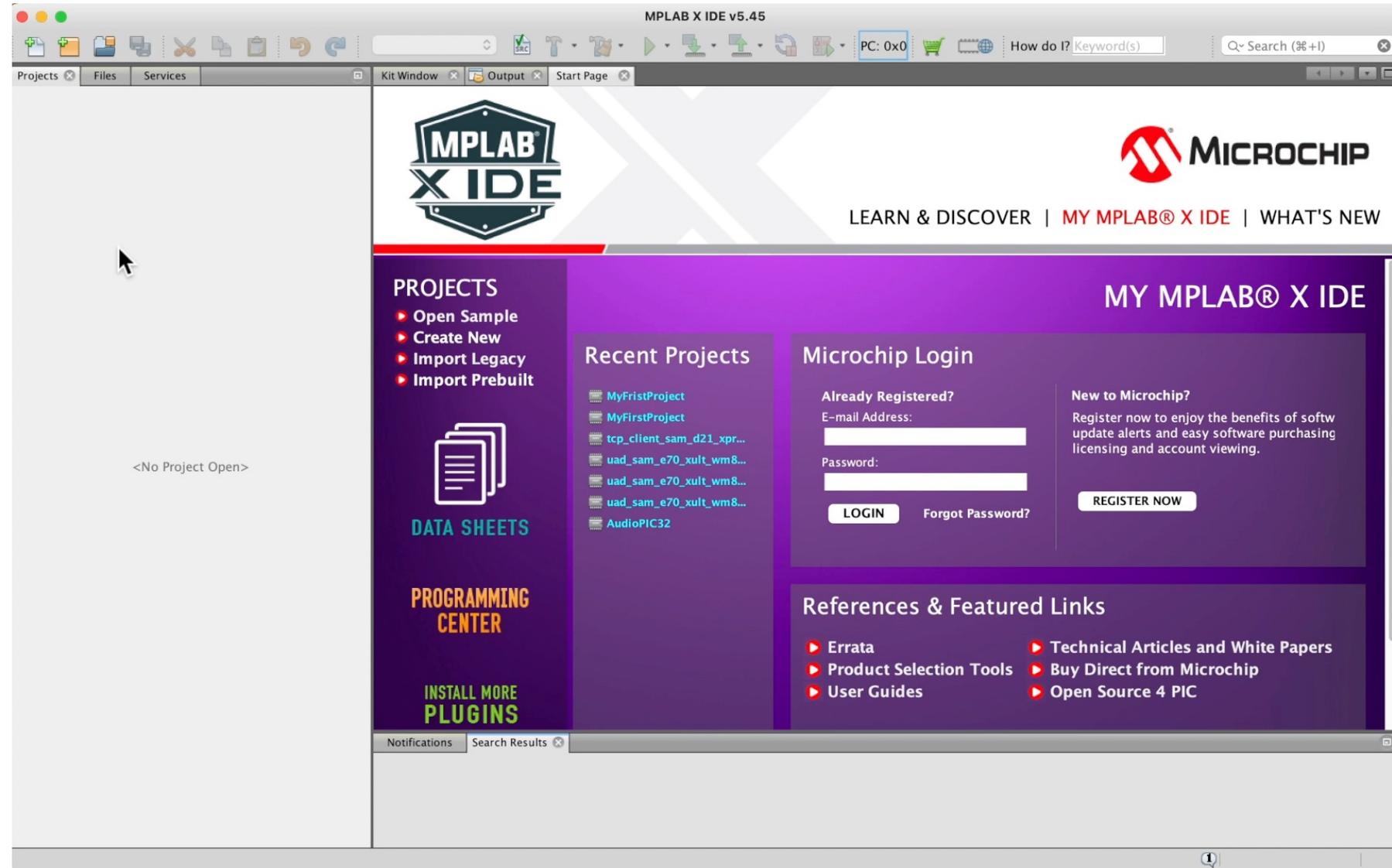
- **Reference Example Codes**
 - Develop a Secured Smart Home Application from scratch
 - Wi-Fi[®] Touch and OLED Display
 - Wi-Fi Provisioning over BLE
 - Wi-Fi to CAN Bus Bridge

Develop a Secured Smart Home Application from scratch

- Make a simple Smart Home device based on WFI32E Curiosity Board connected to an MQTT Broker.
- The object will be monitored and controlled remotely thru a laptop or a mobile device. https://github.com/MicrochipTech/PIC32MZW1_Workshop/tree/master/06_develop



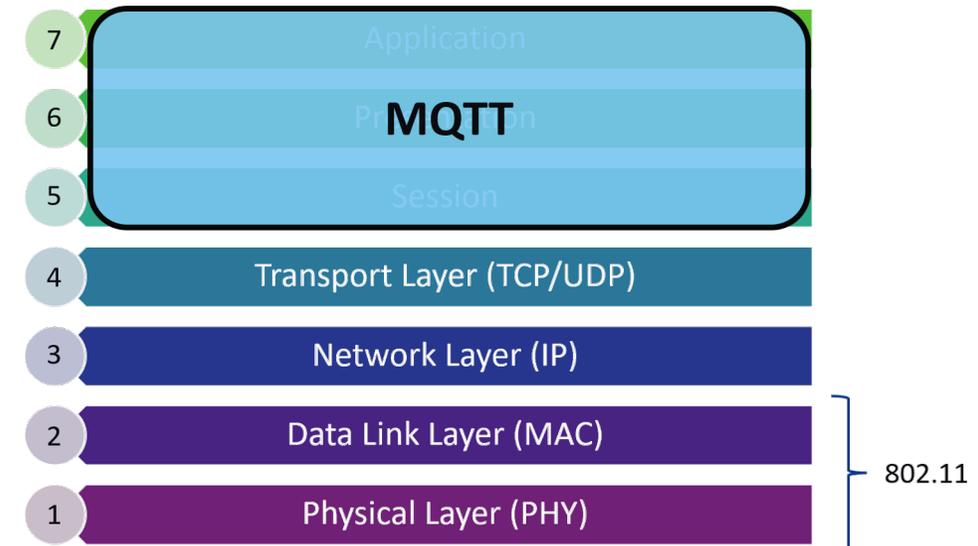
Develop a Secured Smart Home Application from scratch



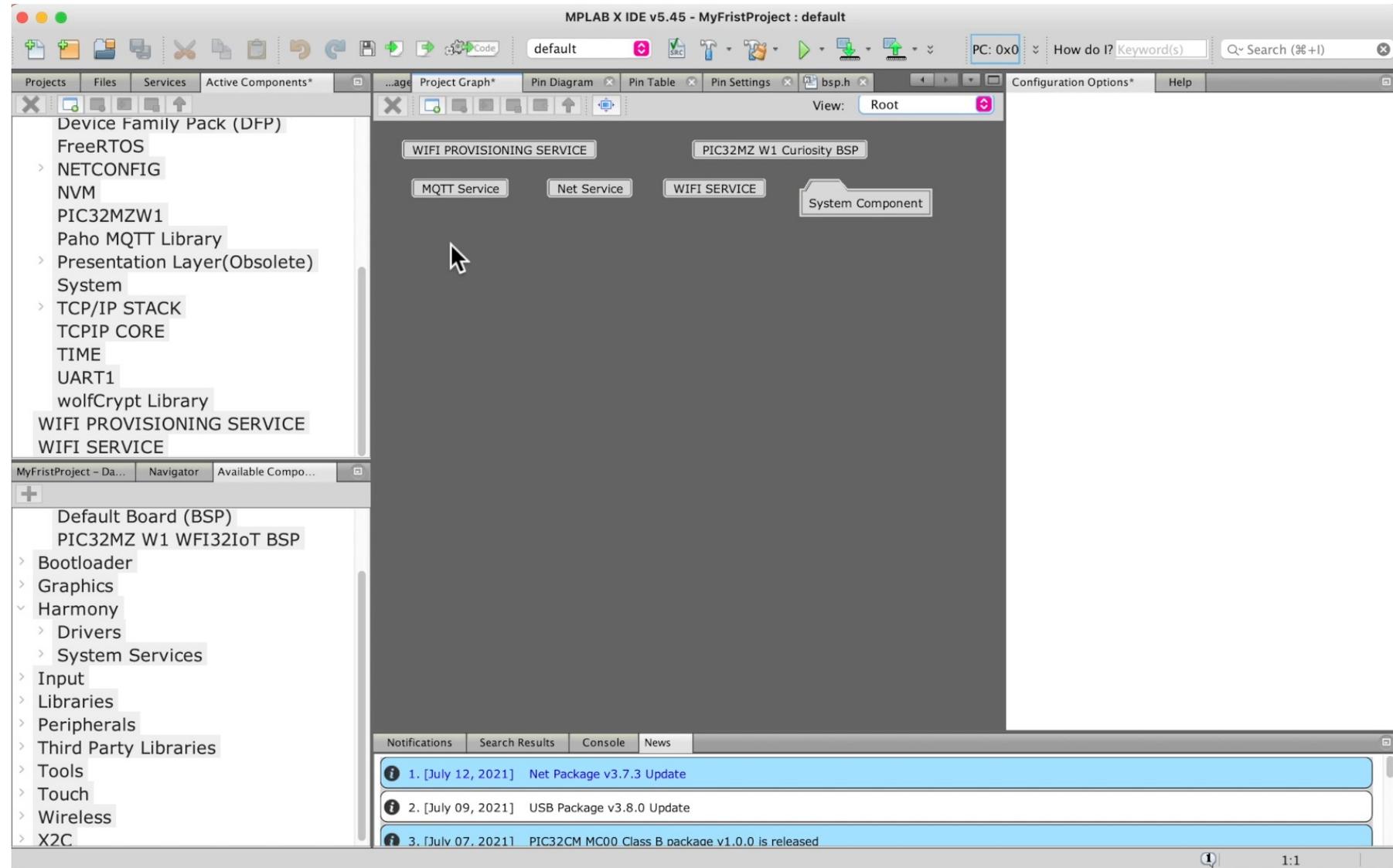
Develop a Secured Smart Home Application from scratch

MQTT Service

- MQTT Service is an Harmony Networking layer which include components such as: *Net Service, Wi-Fi Service, NVM, Core Timer, UART1, Cryptographic, wolfCrypt Library, Time, Core, FreeRTOS, Console, Command, Debug, PIC32MZW1, Netconfig, TCPIP Core, BA414E, TCP/IP Application Layer Configuration, Netconfig, IPv4, ARP, UDP, ICMPv4, Wi-Fi Provisioning Service, Presentation Layer, Paho MQTT Library,...*
- Notice that FreeRTOS component has been added and the Application is configured for FreeRTOS support.
- The MQTT protocol is based on TCP/IP. Both the client and the broker need to have a TCP/IP stack.

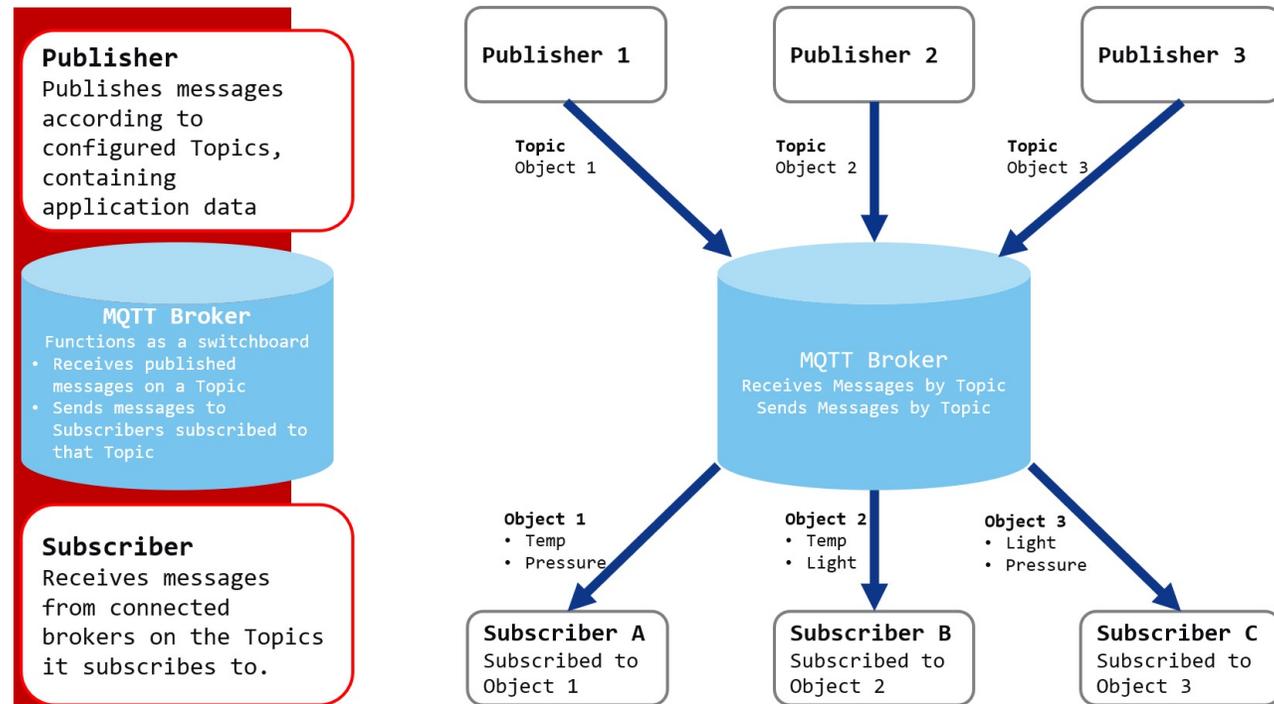


Develop a Secured Smart Home Application from scratch



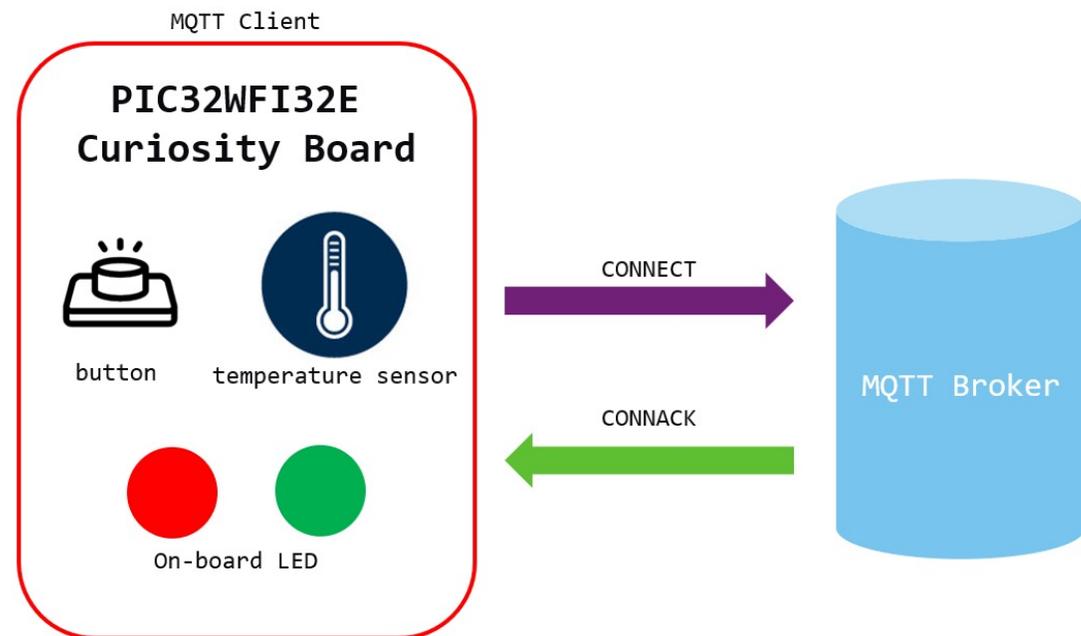
Develop a Secured Smart Home Application from scratch

- To share information between MQTT clients over MQTT broker, you need to provide MQTT topics which is nothing more than a form of addressing. The MQTT connection is always between one client and the broker. Clients never connect to each other directly.



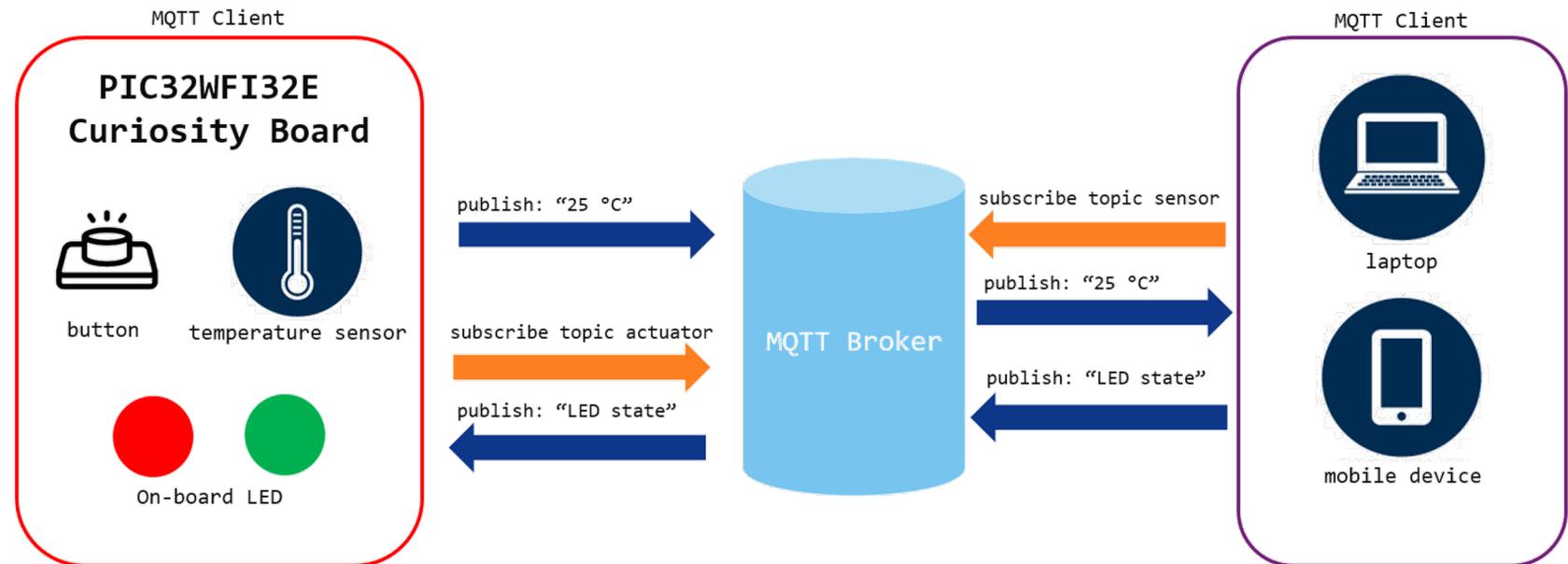
Develop a Secured Smart Home Application from scratch

- First, the client (PIC32MZ W1) sends a **CONNECT** message to the broker to initiate a connection
- The broker responds with a **CONNACK** message and a status code.
- Once the connection is established, the broker keeps it open until the client sends a disconnect command or the connection breaks.



Develop a Secured Smart Home Application from scratch

- The client (PIC32MZ W1) should subscribe to a topic to receive the command messages that another client sends for changing the state of the actuator (PIC32MZW1's on-board LED).
- The client (PIC32MZ W1) should publish sensor data to Broker over another topic. Thus, another client will be able to monitor the data.



Develop a Secured Smart Home Application from scratch

The screenshot displays the MPLAB X IDE v5.45 interface for a project named "MyFristProject". The main window shows a "Project Graph" view with several components: "WIFI PROVISIONING SERVICE", "PIC32MZ W1 Curiosity BSP", "MQTT Service", "Net Service", "WIFI SERVICE", and "System Component". The "MQTT Service" component is selected, and its configuration options are visible on the right side of the IDE. The configuration is divided into several sections:

- Basic Configuration:**
 - Broker Name: test.mosquitto.org
 - Server Port: 1,883
 - Enable TLS:
 - Client Id: (empty field)
 - Network Interfaces: WIFI
- Advanced Configuration:**
 - Enable Auto Reconnect:
 - Enable Clean Sesion:
 - KeepAlive Interval: 60
 - Extra Credentials:
 - Last Will and Testament:
- Other Options:**
 - Subscribe Topic:
 - Publish to Topic:
 - Debug:

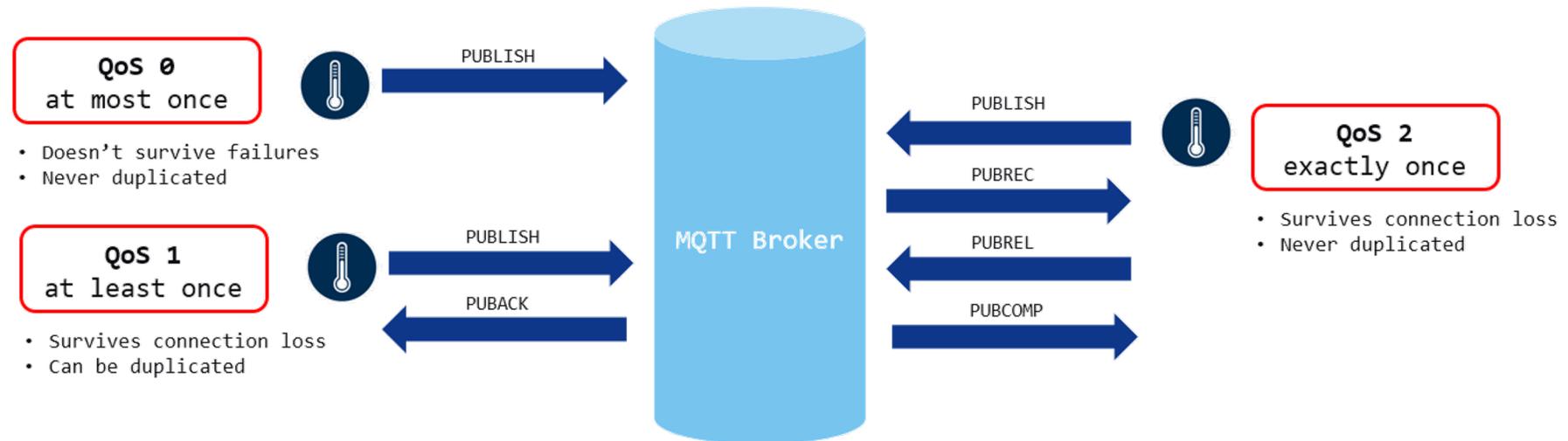
The bottom of the IDE shows a "Notifications" pane with three messages:

1. [July 12, 2021] Net Package v3.7.3 Update
2. [July 09, 2021] USB Package v3.8.0 Update
3. [July 07, 2021] PIC32CM MC00 Class B package v1.0.0 is released

Develop a Secured Smart Home Application from scratch

QoS (Quality of Service) contains 3 levels for the messages emitted:

- **QoS 0** guarantees that the message is delivered **at most once**, or it is not delivered at all. The sender does not care about the message after broadcasting (fire and forget).
- **QoS 1** guarantees that the message is always delivered **at least once**. If the sender does not receive an acknowledgement, the message can be sent multiple times.
- **QoS 2** guarantees that the message is always delivered **exactly once** and no messages are lost. This is ensured by a two-step confirmation with additional messages.



Develop a Secured Smart Home Application from scratch

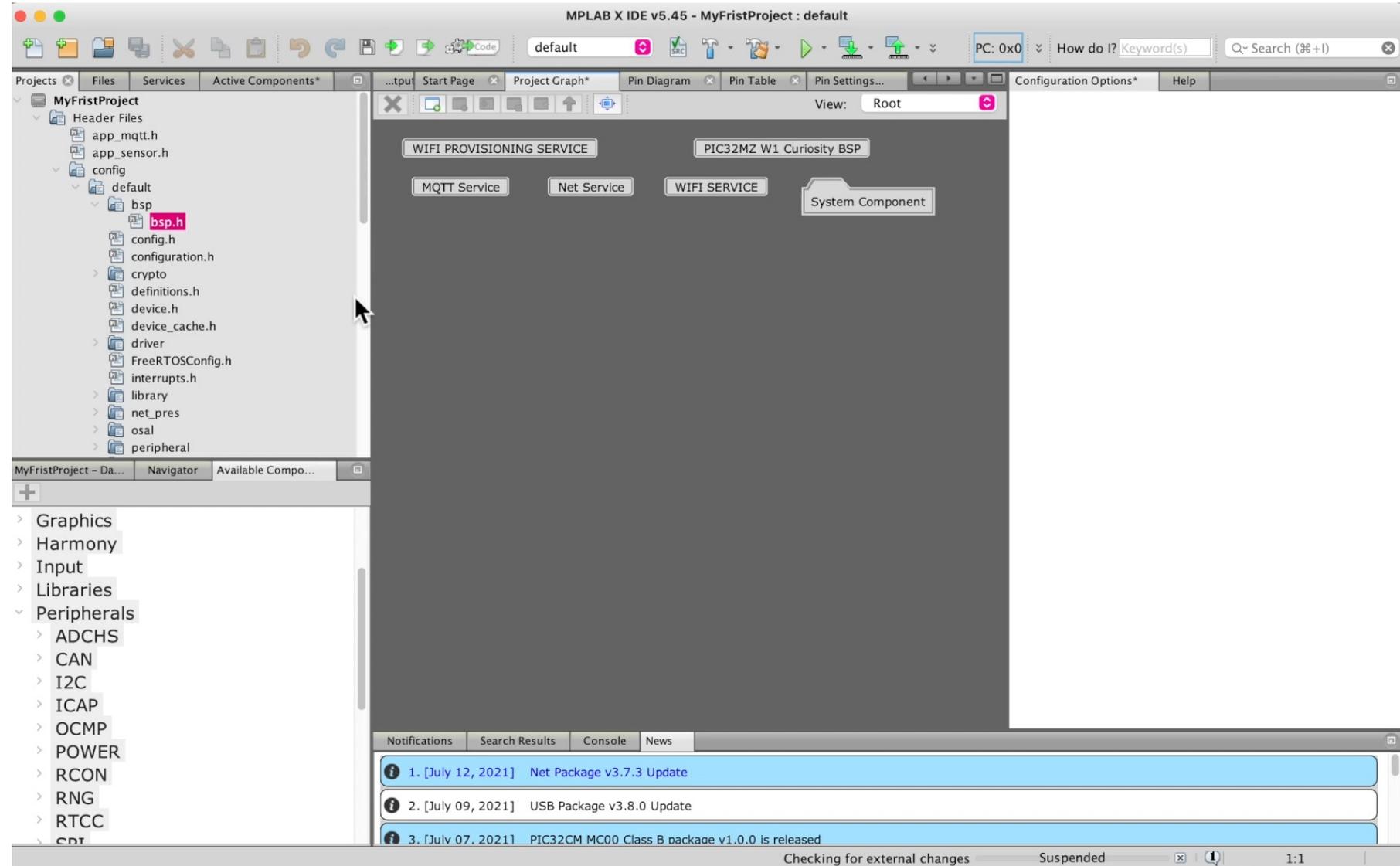
The screenshot displays the MPLAB X IDE v5.45 interface for a project named "MyFristProject : default". The central workspace shows a project graph with components: "WIFI PROVISIONING SERVICE", "PIC32MZ W1 Curiosity BSP", "MQTT Service", "Net Service", "WIFI SERVICE", and "System Component". The "MQTT Service" component is selected, and its configuration options are shown on the right. The configuration includes:

- Basic Configuration:**
 - Broker Name: test.mosquitto.org
 - Server Port: 1,883
 - Enable TLS:
 - Client Id:
 - Network Interfaces: WIFI
- Advanced Configuration:**
 - Enable Auto Reconnect:
 - Enable Clean Sesion:
 - KeepAlive Interval: 60
 - Extra Credentials:
 - Last Will and Testament:
- Subscribe Topic:**
 - Topic Name: pic32mz_w1/actuator
 - QOS: At least once (1)
- Publish to Topic:**
 - Topic Name: pic32mz_w1/sensor
 - QOS: At least once (1)
 - Retain Message:
- Debug:**

The bottom of the IDE shows a console window with three notifications:

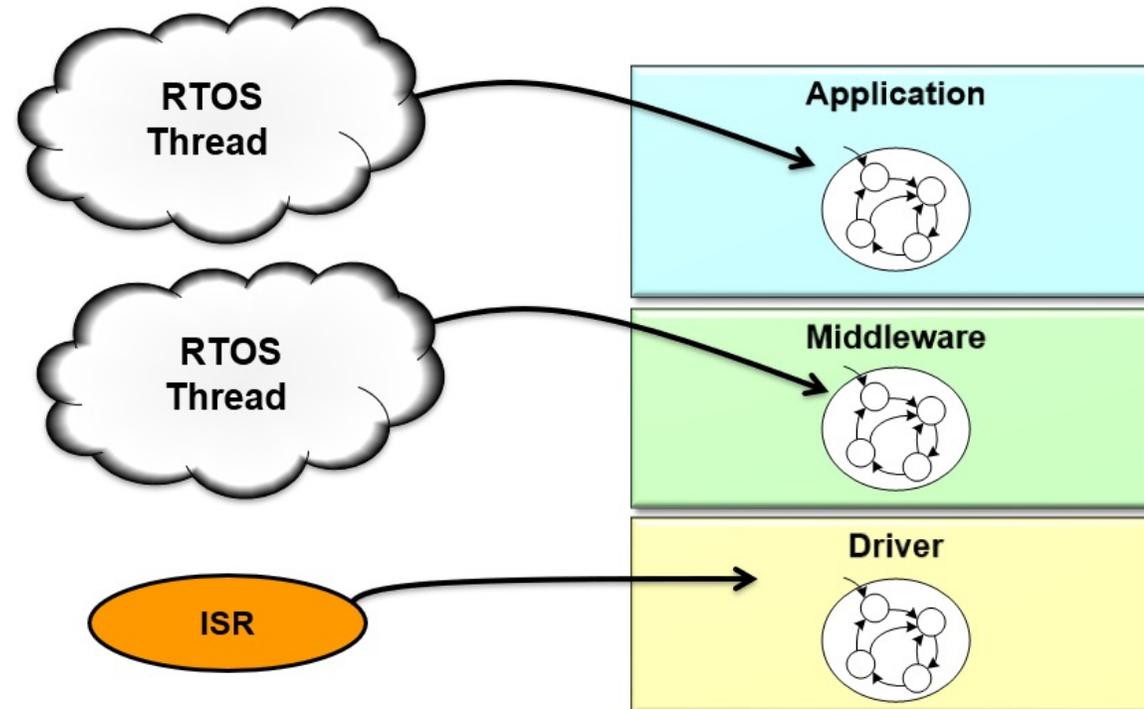
1. [July 12, 2021] Net Package v3.7.3 Update
2. [July 09, 2021] USB Package v3.8.0 Update
3. [July 07, 2021] PIC32CM MC00 Class B package v1.0.0 is released

Develop a Secured Smart Home Application from scratch



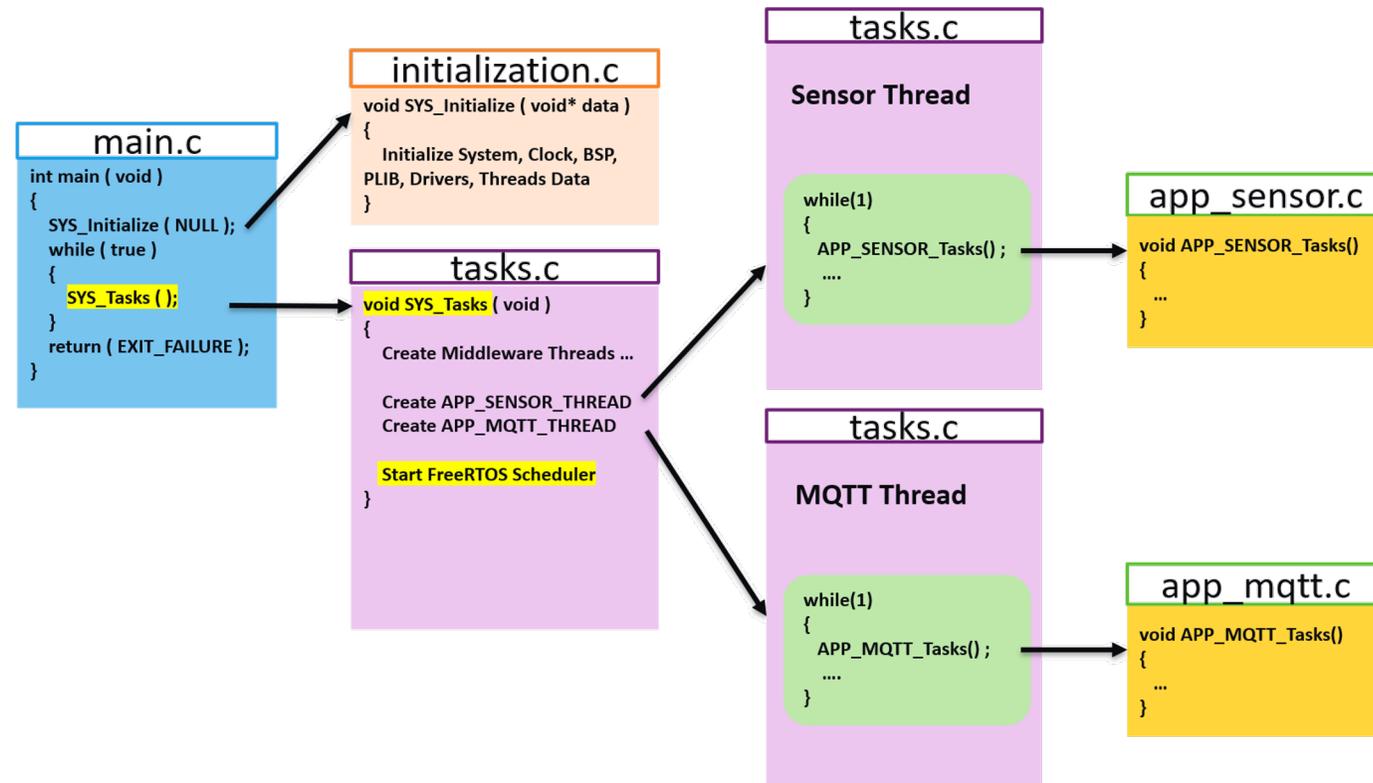
Develop a Secured Smart Home Application from scratch

- The project is using FreeRTOS library to create application threads & intercommunicate between application threads.
- Allows the individual task to be assigned a priority
- Interrupts support the best real-time response latency



Develop a Secured Smart Home Application from scratch

- The System task routine then calls the RTOS scheduler, thereafter all RTOS threads will be managed by the scheduler and the **SYS_Tasks** function will not return.
- MHC also generates the individual daemon threads that will run the MPLAB Harmony libraries and the application tasks (**_APP_SENSOR_Tasks** and **_APP_MQTT_Tasks**).



Develop a Secured Smart Home Application from scratch

- The **main.c** file calls the **SYS_Tasks()** routine, which create the sensor and MQTT threads and makes a call to FreeRTOS API **vTaskStartScheduler()** to start the scheduler.

```
tasks.c

void _APP_SENSOR_Tasks( void *pvParameters )
{
    while(1)
    {
        APP_SENSOR_Tasks();
        vTaskDelay(50 / portTICK_PERIOD_MS);
    }
}

void _APP_MQTT_Tasks( void *pvParameters )
{
    while(1)
    {
        APP_MQTT_Tasks();
        vTaskDelay(50 / portTICK_PERIOD_MS);
    }
}

void SYS_Tasks ( void )
{
    xTaskCreate((TaskFunction_t) _APP_SENSOR_Tasks, "APP_SENSOR_Tasks", 1024, NULL, 1, &xAPP_SENSOR_Tasks);
    xTaskCreate((TaskFunction_t) _APP_MQTT_Tasks, "APP_MQTT_Tasks", 1024, NULL, 1, &xAPP_MQTT_Tasks);

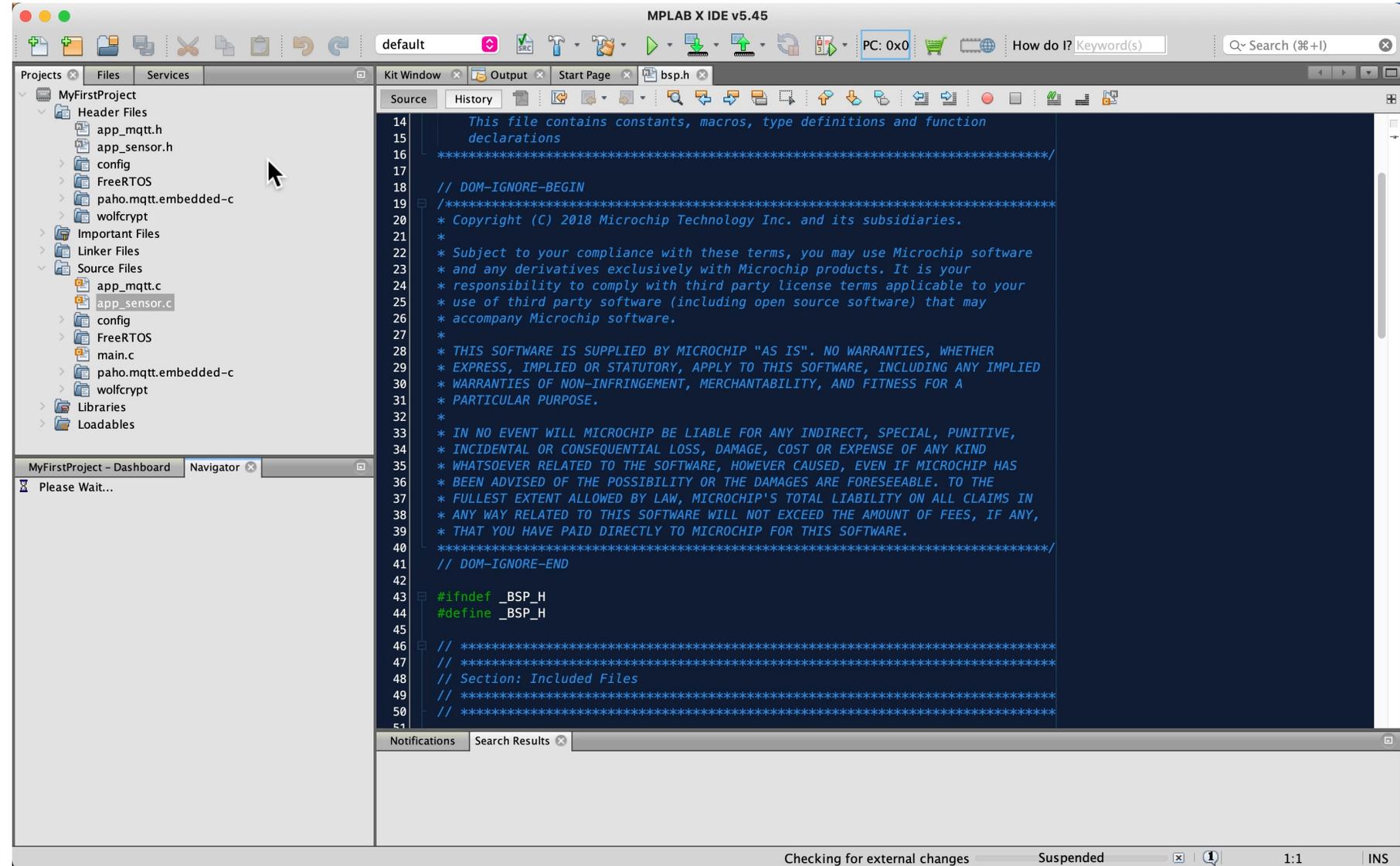
    vTaskStartScheduler(); /* This function never returns. */
}

Thread Functions
FreeRTOS Specific
```

Develop a Secured Smart Home Application from scratch

- The MPLAB[®] Harmony components configured for MQTT and RTOS based application and you are ready to begin developing your application logic of two independent Application Tasks.
 - **_APP_SENSOR_Tasks**
 - **_APP_MQTT_Tasks**
 - publishing data to temperature topic
 - subscribing to actuator topic and control the on-board LEDs
- The application is already developed and is available in the files:
 - app_sensor.c,
 - app_sensor.h,
 - app_mqtt.c,
 - app_mqtt.h

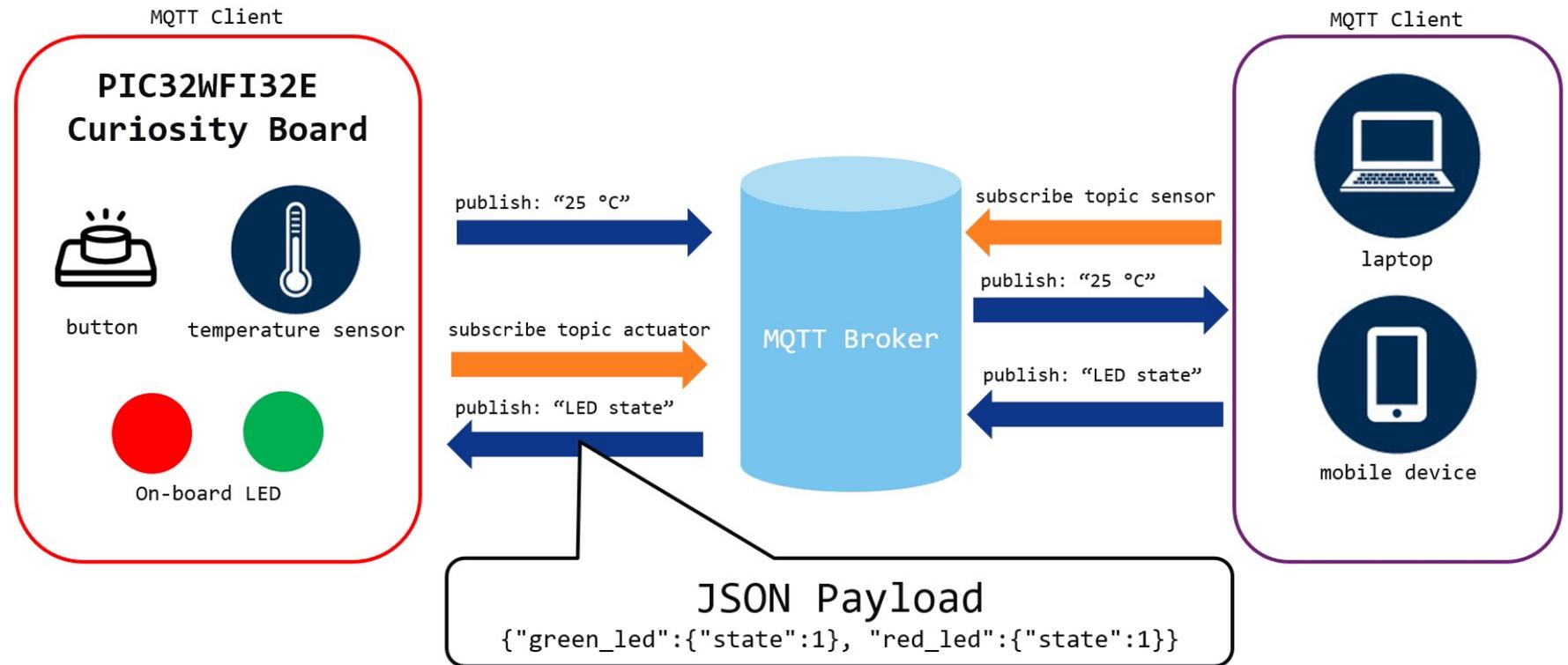
Develop a Secured Smart Home Application from scratch



Develop a Secured Smart Home Application from scratch

```
192 // According to TC1047 datasheet, sensor out voltage V0 = 500mV @0°C and V0 = 750mV @+25°C
193 float ambientTemp = (mVolts - 500);
194 // Voltage slope output = Temp coeff. = 10mV/°C
195 ambientTemp /= 10.0;
196 // Store temperature in App data
197 app_sensorData.temp = ambientTemp;
198 /* Prepare for the next averaging cycle */
199 adcAccumulateNum = 0;
200 adcCountAccumulate = 0;
201 SYS_CONSOLE_PRINT("[app_sensor_thread] Temp=%0.1f\r\n", app_sensorData.temp);
202
203
204 /* TODO: Use FreeRTOS queue to notify the MQTT task to publish the temperature value */
205 // Don't block if the queue is already full.
206 xQueueSend(queueHandle, &app_sensorData.temp, (TickType_t) 0);
207
208 }
209 app_sensorData.adcReady = false;
210
211 /* TODO: move to the APP_SENSOR_STATE_MONITOR_SWITCH state */
212 app_sensorData.state = APP_SENSOR_STATE_MONITOR_SWITCH;
213 break ;
214
215 /* The default state should never be executed. */
216 default:
217 {
218     /* TODO: Handle error in application's state machine. */
219     break;
220 }
221 }
222
223
224
225 /******
226 End of File
227 */
228
```

Develop a Secured Smart Home Application from scratch



Develop a Secured Smart Home Application from scratch

Demo

The image displays a composite view of a smart home application development and deployment process. On the left, a terminal window titled 'COM16 - Tera Term VT' shows the following log output:

```
[app_mqtt] MqttCallback(): Published Msg(131) to Topic  
[app_sensor_thread] Temp=42.2  
[app_sensor_thread] Temp=42.2  
[app_mqtt] MqttCallback(): Published Msg(132) to Topic  
[app_sensor_thread] Temp=42.2  
[app_sensor_thread] Temp=42.2  
[app_mqtt] MqttCallback(): Published Msg(133) to Topic  
[app_sensor_thread] Temp=42.3  
[app_sensor_thread] Temp=42.3  
[app_mqtt] MqttCallback(): Published Msg(134) to Topic  
[app_sensor_thread] Temp=42.3  
[app_mqtt] MqttCallback(): Published Msg(135) to Topic  
[app_sensor_thread] Temp=42.3  
[app_sensor_thread] Temp=42.3  
[app_mqtt] MqttCallback(): Published Msg(136) to Topic  
[app_sensor_thread] Temp=42.2  
[app_sensor_thread] Temp=42.2
```

In the center, a MQTT client interface is shown with a 'Messages' tab selected. The interface includes a 'Connect' section with the following settings:

- Name: Default
- Host: test.mosquitto.org
- Port: 1883
- Username: username
- Password: password
- TLS/SSL:
- Add to favorites:

The 'Messages' section shows a 'NUMBER OF MESSAGES' slider set to 10, and a message list that is currently empty, displaying 'NO MESSAGES, SUBSCRIBE TO SOME TOPICS'.

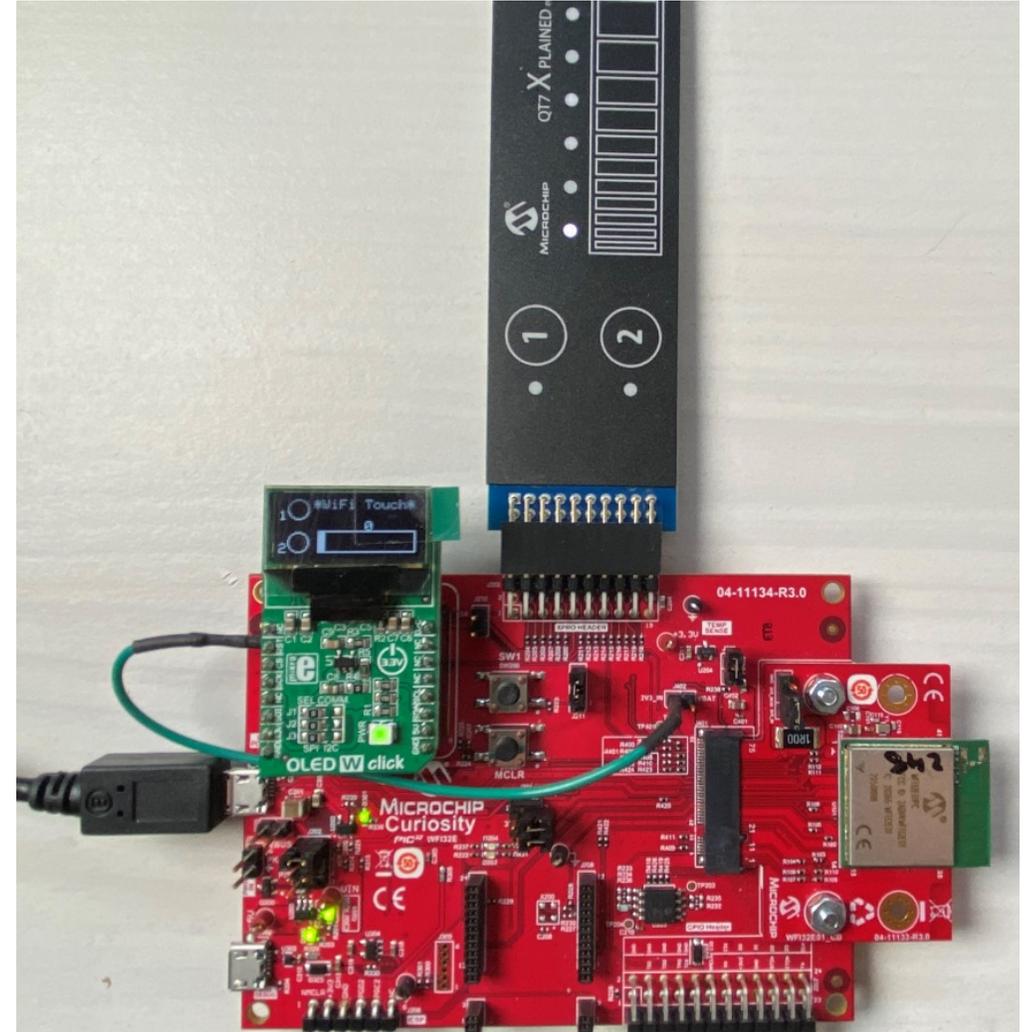
On the right, a physical Raspberry Pi board is shown, which is the hardware component of the smart home application. The board is a red Raspberry Pi 4 Model B, featuring a Microchip Curiosity board and a Wi-Fi module. The board is connected to a USB cable and is shown in a close-up view.

- **Reference Example Codes**

- Develop a Secured Smart Home Application from scratch
- **Wi-Fi[®] Touch and OLED Display**
- Wi-Fi Provisioning over BLE
- Wi-Fi to CAN Bus Bridge

Wi-Fi Touch and OLED Display

- Sample application showcasing Wi-Fi connectivity, Capacitive Touch and OLED Display control.
- The application acts as a TCP Server to which a TCP Client can connect and visualize QT7 Touch Xpro data.
- The Touch data are also printed on an OLED Display.

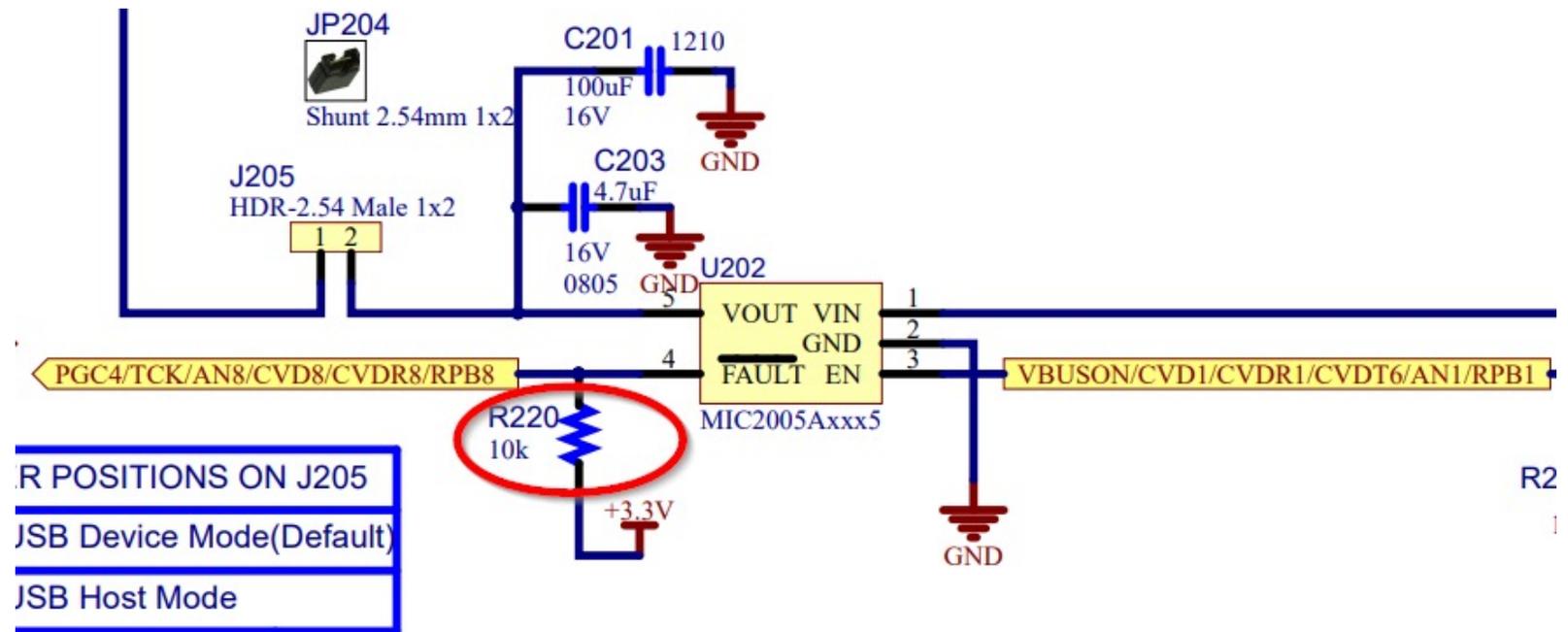


https://github.com/MicrochipTech/PIC32MZW1_Workshop/tree/master/07_projects#wi-fi-touch-and-oled-display

Wi-Fi Touch and OLED Display

Touch example with CVD and QT7 Xpro

- One of the slider sensor line (Y-line2) is connected to U202 and the line is loaded with 10k (R220) resistor. You could remove R220 to not interfere the measurement of the slider sensor.



Wi-Fi Touch and OLED Display

Touch example with CVD and QT7 Xpro

- QT7 uses one slider & 2 touch buttons.

QT7 Xpro Header Pin	Function	Description	WFI32E01 Module Pin
1	ID	-	-
2	GND	-	-
3	Y-LINE-5	Driven shield	RB6 (X1 in MHC)
4	Y-LINE-1	Button 1 Sensor	RA14 (button 0/Y14 in MHC)
5	LED0	LED for Slider	RB12 (LED_SLIDER_6 in MHC)
6	LED6	LED for Button 1	RK6 (LED_BUTTON_1 in MHC)
7	Y-LINE-2	Slider Sensor	RB8 (slider 0/channel 4/Y8 in MHC)
8	Y-LINE-3	Slider Sensor	RA13 (slider 0/channel 3/Y15 in MHC), shared with Temp sensor)
9	Y-LINE-4	Slider Sensor	RA10 (slider 0/channel 2/Y17 in MHC)
10	Y-LINE-0	Button 2 Sensor	RB2 (button 1/Y2 in MHC)

Wi-Fi Touch and OLED Display

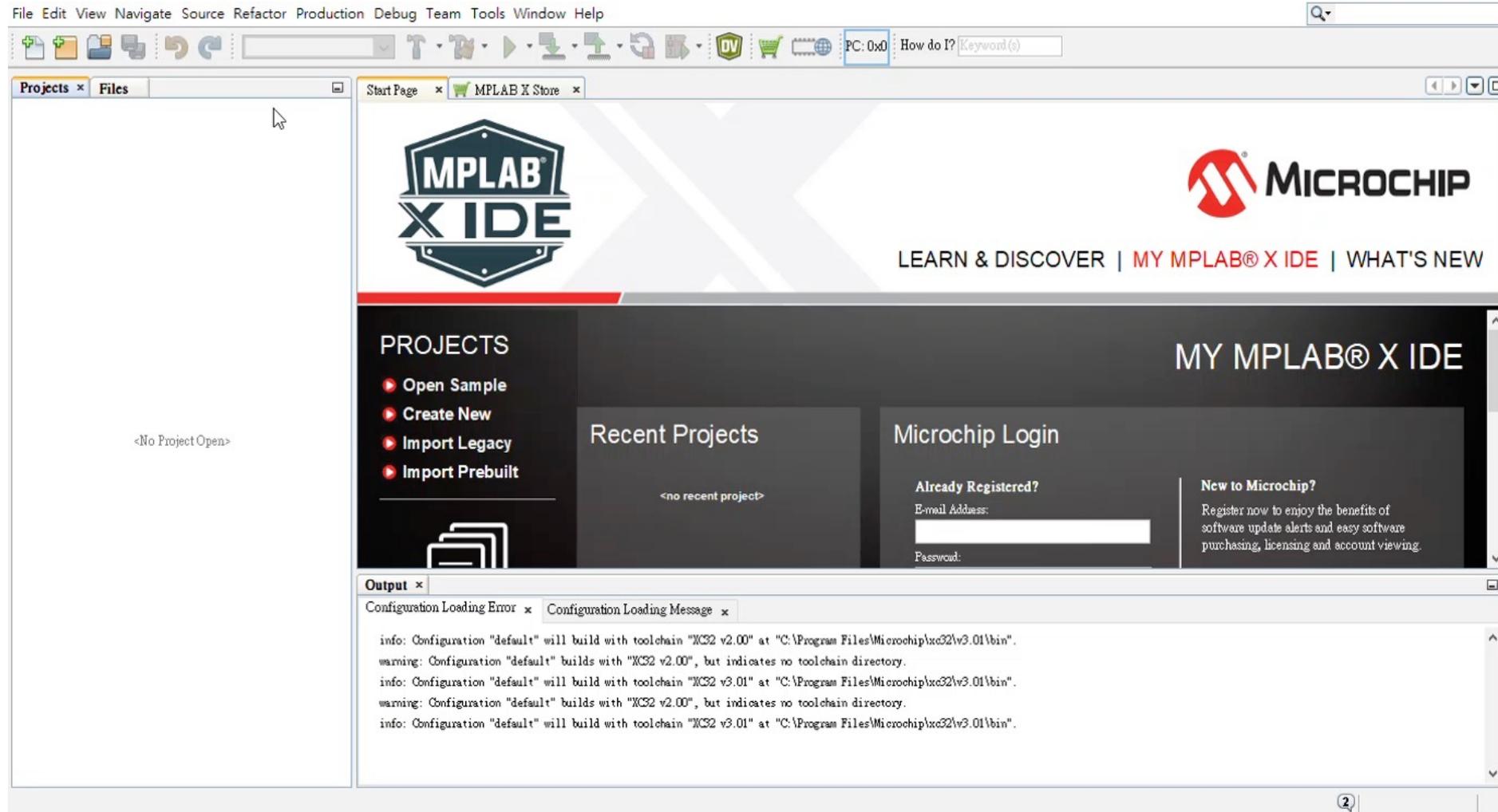
Touch example with CVD and QT7 Xpro

- QT7 uses one slider & 2 touch buttons.

QT7 Xpro Header Pin	Function	Description	WFI32E01 Module Pin
11	LED7	LED for Button 2	RA5 (LED_BUTTON_2)
12	LED1	LED for Slider	RA4 (LED_SLIDER_5)
13	NC	-	-
14	NC	-	-
15	LED2	LED for Slider	RB7 (LED_SLIDER_4)
16	LED3	LED for Slider	RK5 (LED_SLIDER_2)
17	LED4	LED for Slider	RK4 (LED_SLIDER_3)
18	LED5	LED for Slider	RA11 (LED_SLIDER_1)
19	GND	-	-
20	VCC	-	-

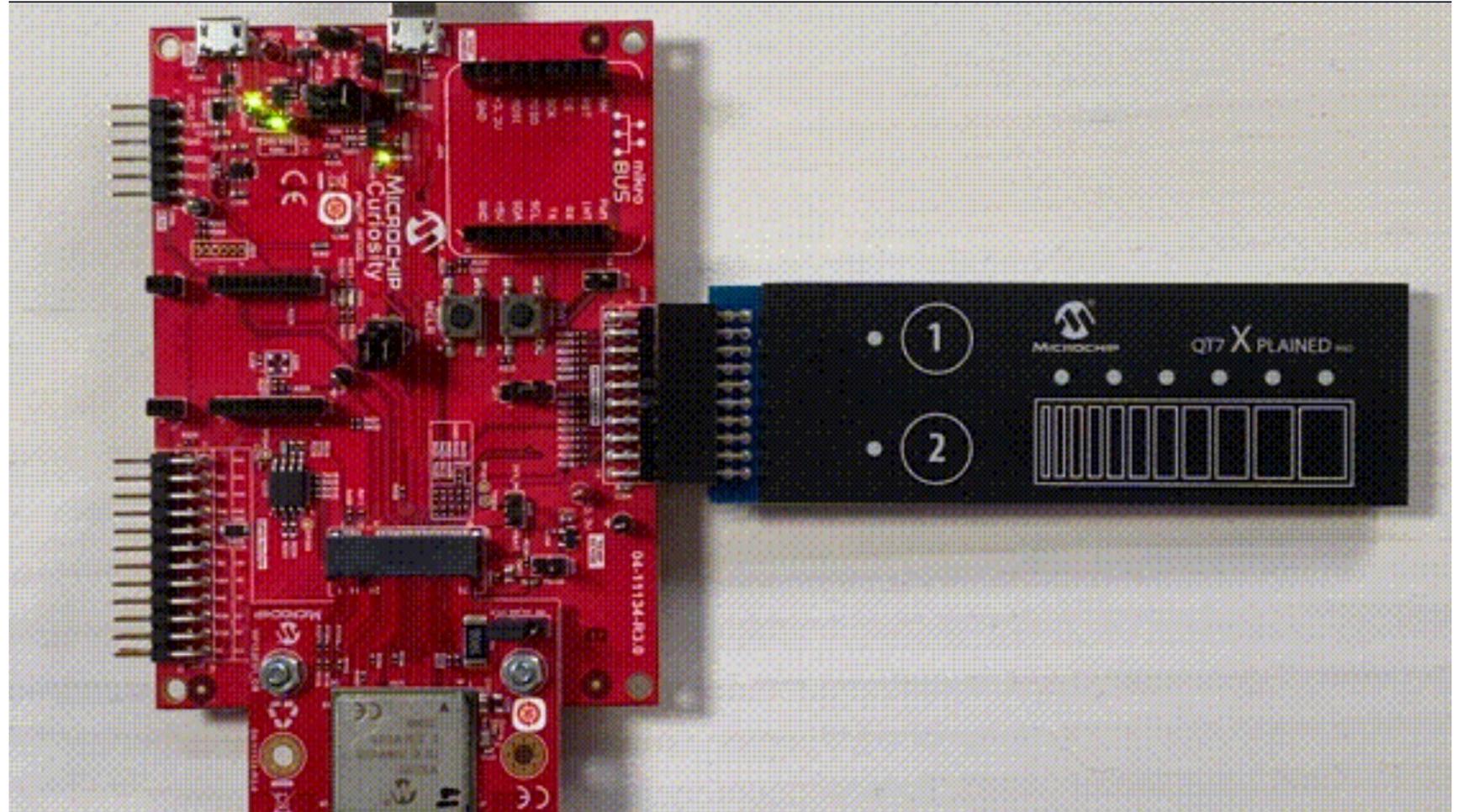
Wi-Fi Touch and OLED Display

Touch example with CVD and QT7 Xpro



Wi-Fi Touch and OLED Display

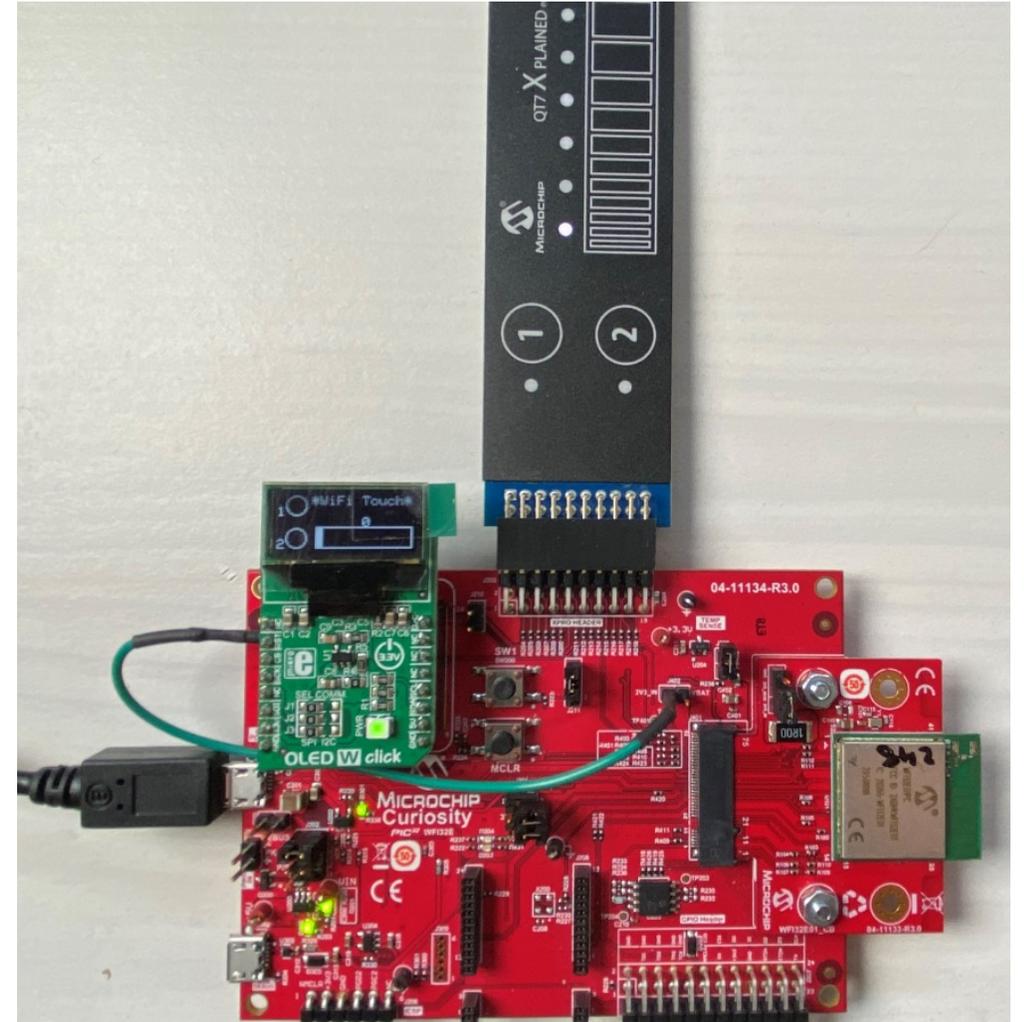
Touch example with CVD and QT7 Xpro



Wi-Fi Touch and OLED Display

Touch example with QT7 Xpro & OLED W Click

- The [OLED W click](#) carries a 96 x 39px white monochrome passive matrix OLED display. To drive the display, [OLED W click](#) features an [SSD1306](#) controller.
- In this demo, the [OLED W click](#) communicates with the WFI32 MCU through SPI lines.



Wi-Fi Touch and OLED Display

Touch example with QT7 Xpro & OLED W Click

- SPI2 for OLED W click interface
- Touch Library using ADCHS and TMR2 components
- Wi-Fi Service, Wi-Fi provisioning and Net services

The image displays the Microchip Studio Project Graph interface, showing a system component view of a device. The graph includes various libraries and services, such as NVM, CORE TIMER, UART1, Cryptographic (Crypto) Library, wolfCrypt Library, TIME, CONSOLE, COMMAND, TCP/IP CORE, DEBUG, NETCONFIG, Core, Presentation Layer, and SPI2. The SPI2 component is highlighted with a red box and labeled "OLED W Click".

The Configuration Options* window is open, showing the configuration for the SPI2 component. The Baud Rate in Hz is set to 1 000 000. The SPI Mode is set to 0. The configuration options include:

- Enable Interrupts?
- Master Mode Enable bit
- Clock Polarity Select bit
- SPI Clock Edge Select bit
- SPI Data Input Sample Phase bit
- Master Mode Slave Select Enable bit
- Data Width 8-bit
- Master Clock Enable bit
- Dummy Data 0x FF

The Net Service configuration window is also open, showing the configuration for the Net Service component. The configuration options include:

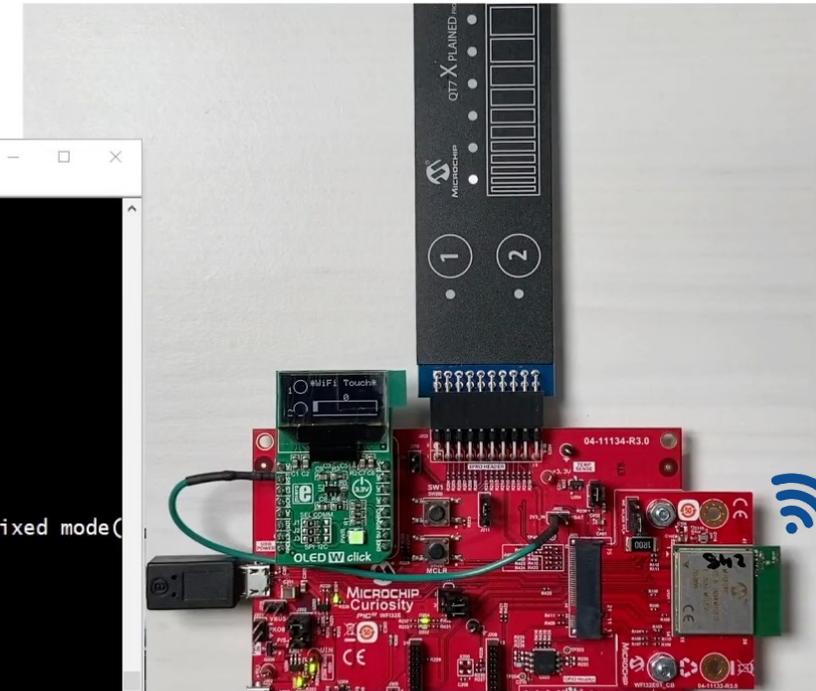
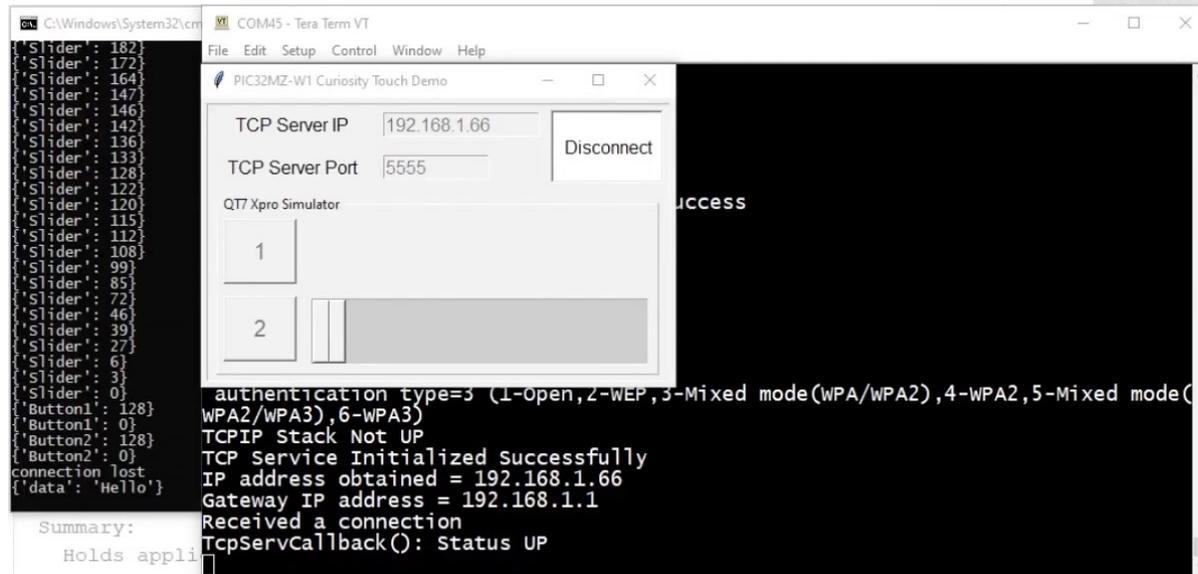
- Supported Interfaces: WIFI_ONLY
- Debug
- Instance 0
- Intf: WIFI
- Ip Protocol: TCP
- Mode: SERVER
- Enable Auto Connect
- Enable TLS
- Server Port: 5 555
- Host Name/ IP Address: 192.168.1.1
- Instance 1

The Wi-Fi Service configuration window is also open, showing the configuration for the Wi-Fi Service component. The configuration options include:

- WIFI SERVICE
- Device Mode: STA
- STA Mode
- AP Mode
- Advanced Configuration

Wi-Fi Touch and OLED Display

Touch example with QT7 Xpro & OLED W Click

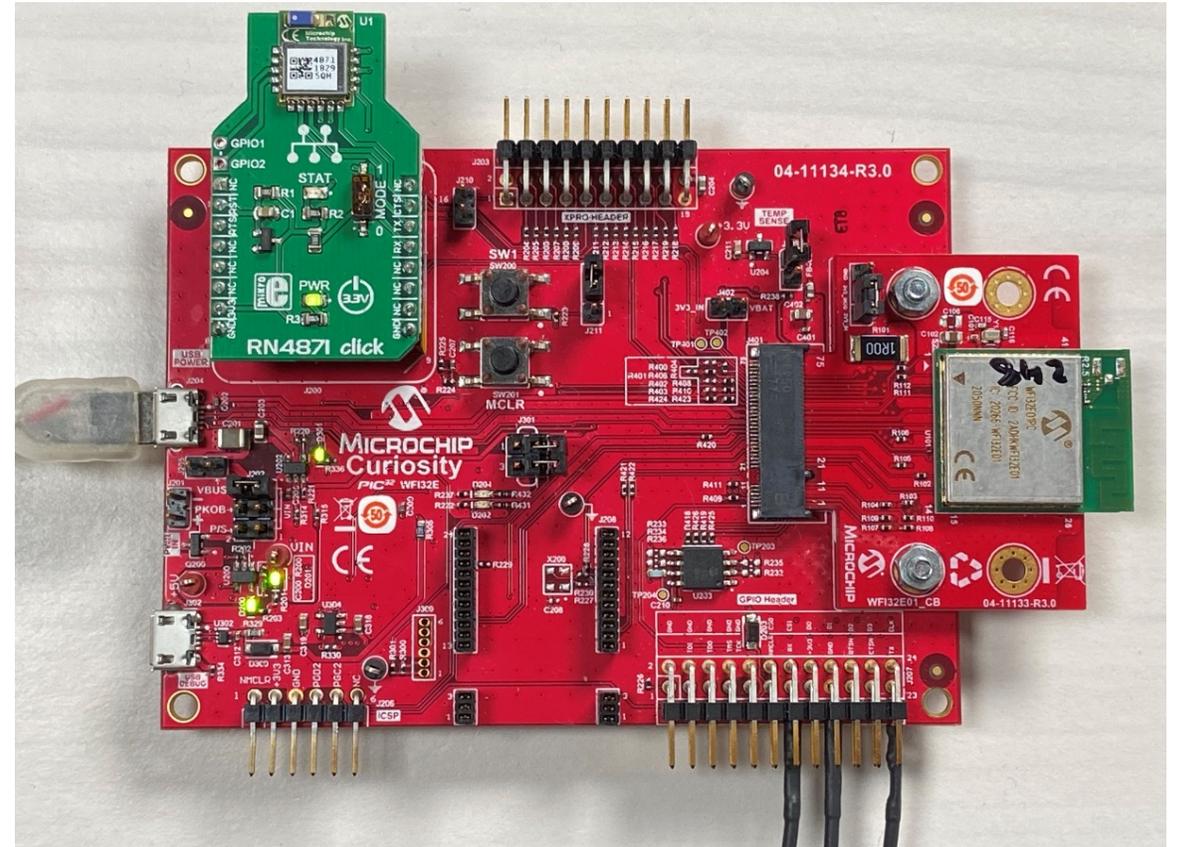


- **Reference Example Codes**

- Develop a Secured Smart Home Application from scratch
- Wi-Fi[®] Touch and OLED Display
- **Wi-Fi Provisioning over BLE**
- Wi-Fi to CAN Bus Bridge

Wi-Fi Provisioning over BLE

- Attach external BLE device to WFI32E Curiosity board and enable Wi-Fi communication and configuration over BLE.
- Watch the video and see how to enable Wi-Fi provisioning over BLE with WFI32E Curiosity board



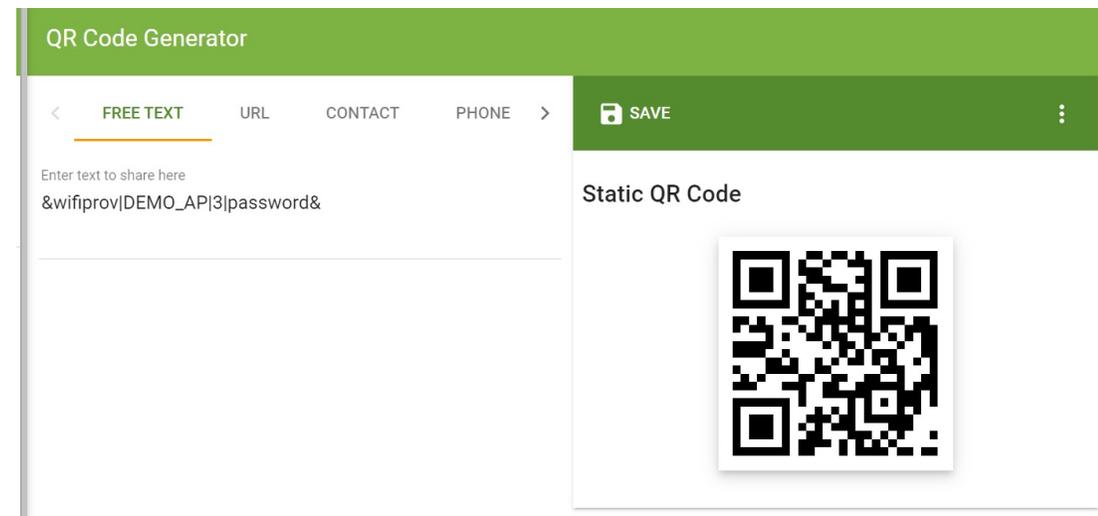
https://github.com/MicrochipTech/PIC32MZW1_Workshop/tree/master/07_projects#wi-fi-provisioning-over-ble

Wi-Fi Provisioning over BLE

Generate QR code for Wi-Fi provisioning

- A QR code is used in the demo to provision the Wi-Fi configuration over BLE.
- **Frame Format: `&wifiprov|<ssid>|<authtype>|<password>&`**
 - Where **&** is used to indicate the start and the end of the frame.
 - **wifiprov** is required and used as a command keyword.
 - **|** is required and used as a separator.
 - **<ssid>** is the name of the router / network.
 - Auth type represents the security type:
 - OPEN mode; WPAWPA2 (Mixed) mode; WPA2 mode
 - WPA2WPA3 (Mixed) mode; WPA3 mode
 - password is not required in Open mode
- e.g.: **`&wifiprov|DEMO_AP|3|password&`**
- Create your own QR code from:

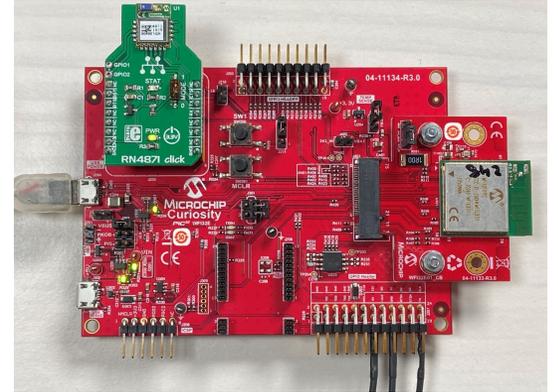
<https://www.the-qr-code-generator.com/>



Wi-Fi Provisioning over BLE

Download from PIC32MZW1_Workshop

- Go to [PIC32MZW1_Workshop/07_projects/resources/software/](#)
- Download [ble_provisioning.zip](#)
- Build and program the code
- Open Tera Term to observe console logs
- Application starts in AP mode



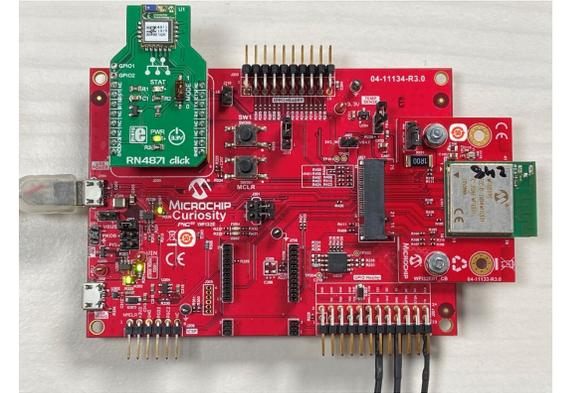
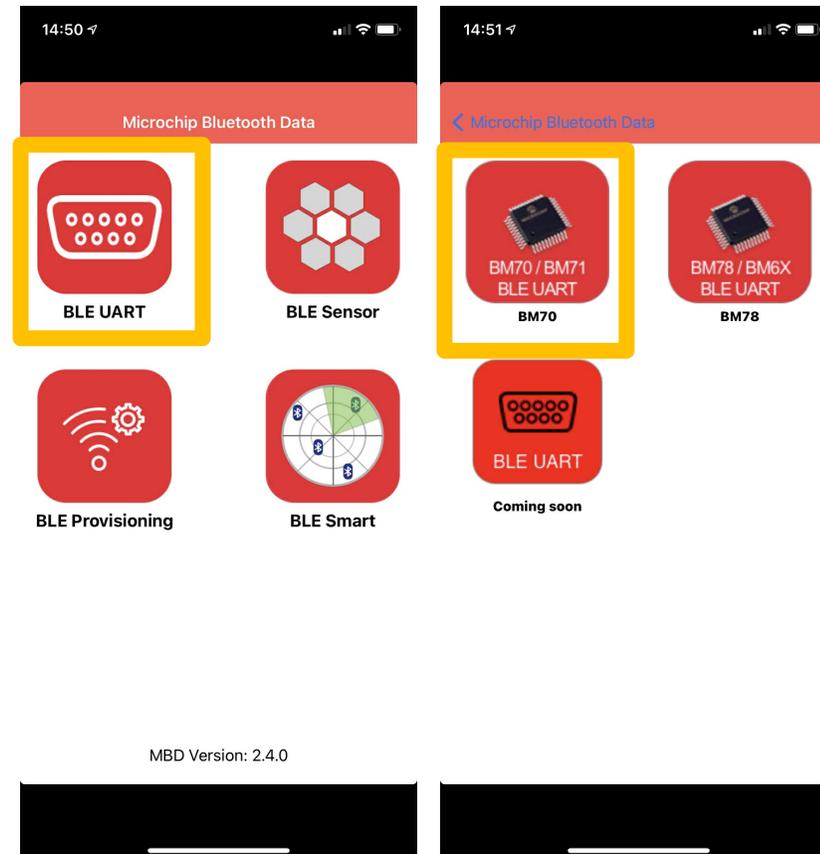
```
TCP/IP Stack: Initialization Started
TCP/IP Stack: Initialization Ended - success

mode=1 (0-STA,1-AP) saveConfig=1

AP Configuration :
channel=1
ssidVisibility=1
ssid=DEMO_AP_SOFTAP
passphrase=password
authentication type=4 (1-Open,2-WEP,3-Mixed mode(WPA/WPA2),4-WPA2,5-Mixed mode(WPA2/WPA3),6-WPA3)
PIC32MZW1 AP Mode IP Address: 192.168.1.1
[APP_BLE] Init.
[APP_BLE] Configuration done.
Open Microchip Bluetooth Data App
- Select BLE UART and BM70
- Connect to your device WFI32_xxxx
- Select Transparent
- Frame format for Wi-Fi provisioning over BLE:
&wifiprov|<ssid>|<authtype>|<password>&
1: Open, 3: WPAWPA2, 4: WPA2, 5: WPA2WPA3, 6: WPA3
e.g. &wifiprov|DEMO_AP|3|password&
```

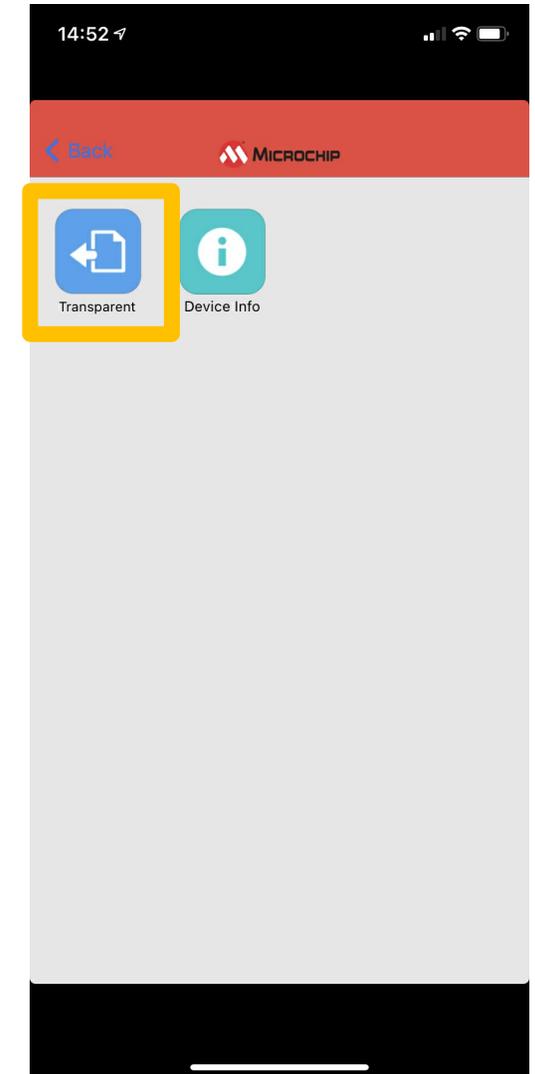
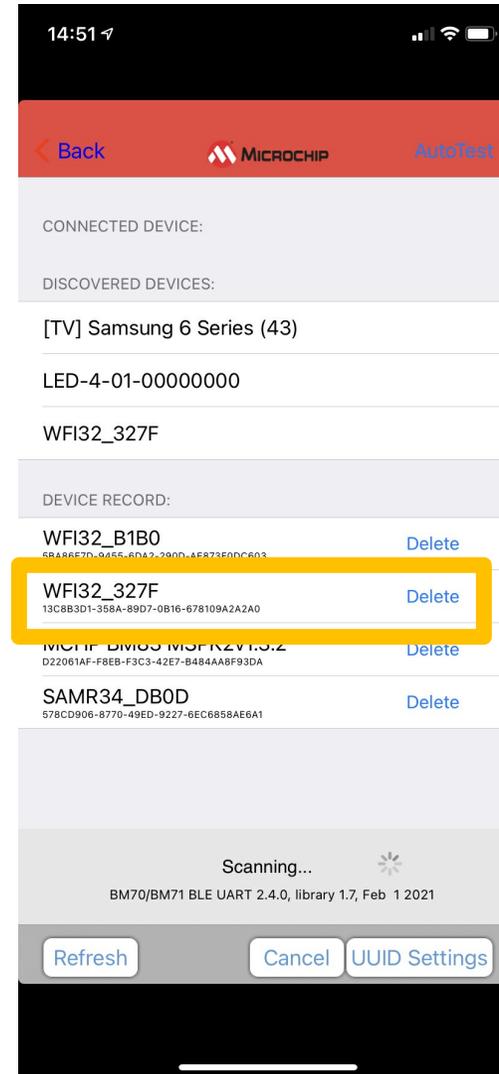
Wi-Fi Provisioning over BLE

- Scan the **QR code** from the smartphone
- Copy your own Wi-Fi provisioning frame
- Open **Microchip Bluetooth Data App**
- Select **BLE UART** then **BM70**



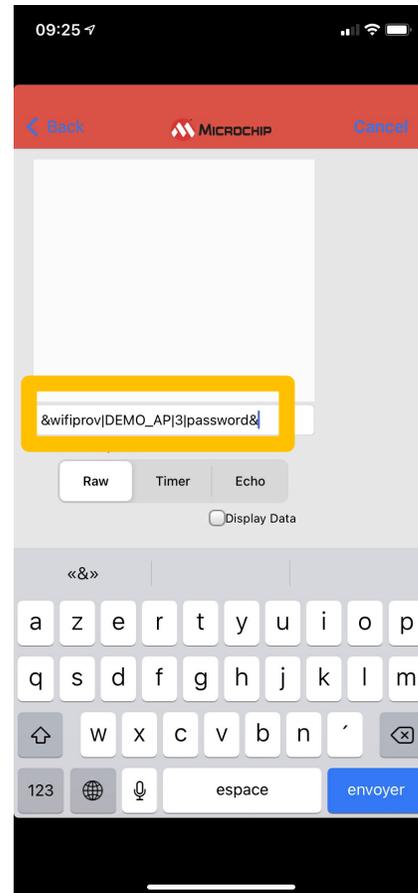
Wi-Fi Provisioning over BLE

- Connect and select your WFI32_xxxx device
 - [APP_BLE] Connected
- Select Transparent option
 - [APP_BLE] Transparent stream opened



Wi-Fi Provisioning over BLE

- Paste (or enter the data manually) and send the Wi-Fi provisioning frame
- Application restarts in STA mode using the new Wi-Fi configuration
- Application gets and IP address from the network



```
[APP_BLE] Frame received
SSID: DEMO_AP - AUTH: 3 - PASS: password
Wi-Fi Configuration done. TCP/IP Stack: Initialization Started
TCP/IP Stack: Initialization Ended - success

mode=0 (0-STA,1-AP) saveConfig=1

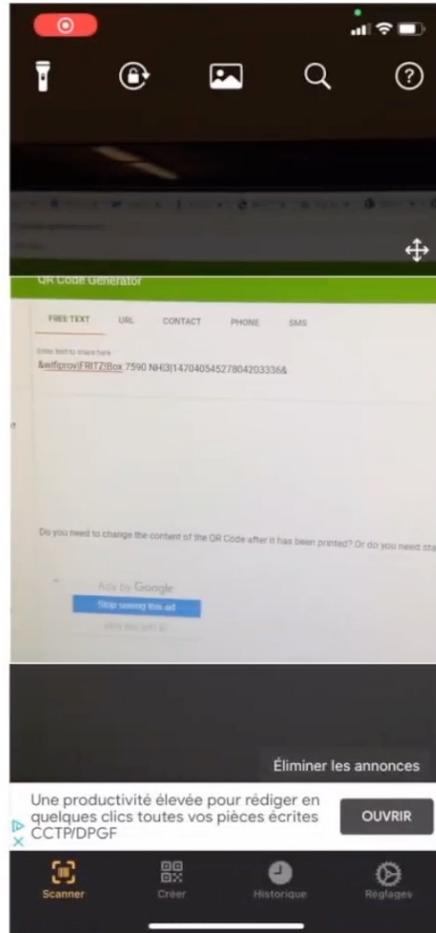
STA Configuration :
channel=0
autoConnect=1
ssid=DEMO_AP
passphrase=password
authentication type=3 (1-Open,2-WEP,3-Mixed mode(WPA/WPA2),4-WPA2,5-Mixed mode(WPA2/WPA3),6-WPA3)
[APP_BLE] Init.
[APP_BLE] Configuration done.
Open Microchip Bluetooth Data App
- Select BLE UART and BM70
- Connect to your device WFI32_xxxx
- Select Transparent
- Frame format for Wi-Fi provisioning over BLE:
&wifiprov|<ssid>|<authtype>|<password>&
1: Open, 3: WPAWPA2, 4: WPA2, 5: WPA2WPA3, 6: WPA3
e.g. &wifiprov|DEMO_AP|3|password&
Trying to connect to SSID : DEMO_AP
STA Connection failed.

Trying to connect to SSID : DEMO_AP
STA Connection failed.

IP address obtained = 192.168.1.149
Gateway IP address = 192.168.1.1
```

Wi-Fi Provisioning over BLE

Demo



- **Reference Example Codes**

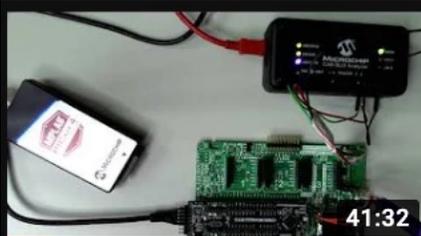
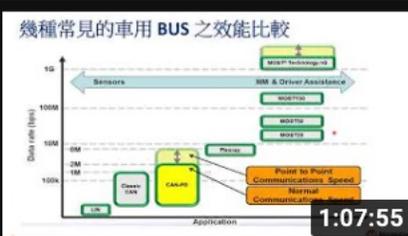
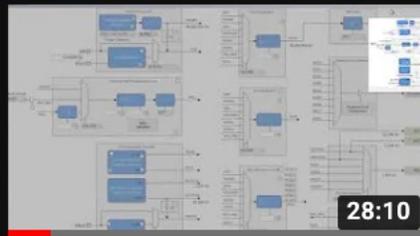
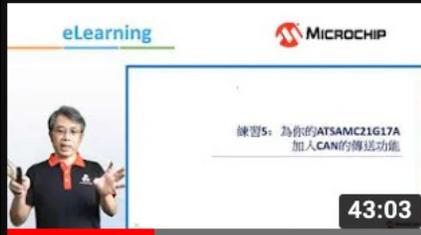
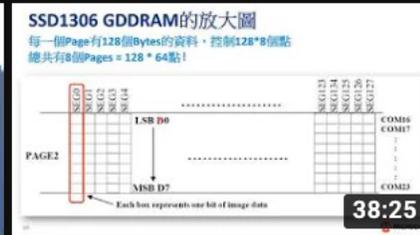
- Develop a Secured Smart Home Application from scratch
- Wi-Fi[®] Touch and OLED Display
- Wi-Fi Provisioning over BLE
- **Wi-Fi to CAN Bus Bridge**

Wi-Fi to CAN Bus Bridge

Review APP-Nano-C21-D21-TW of eLearning Classes

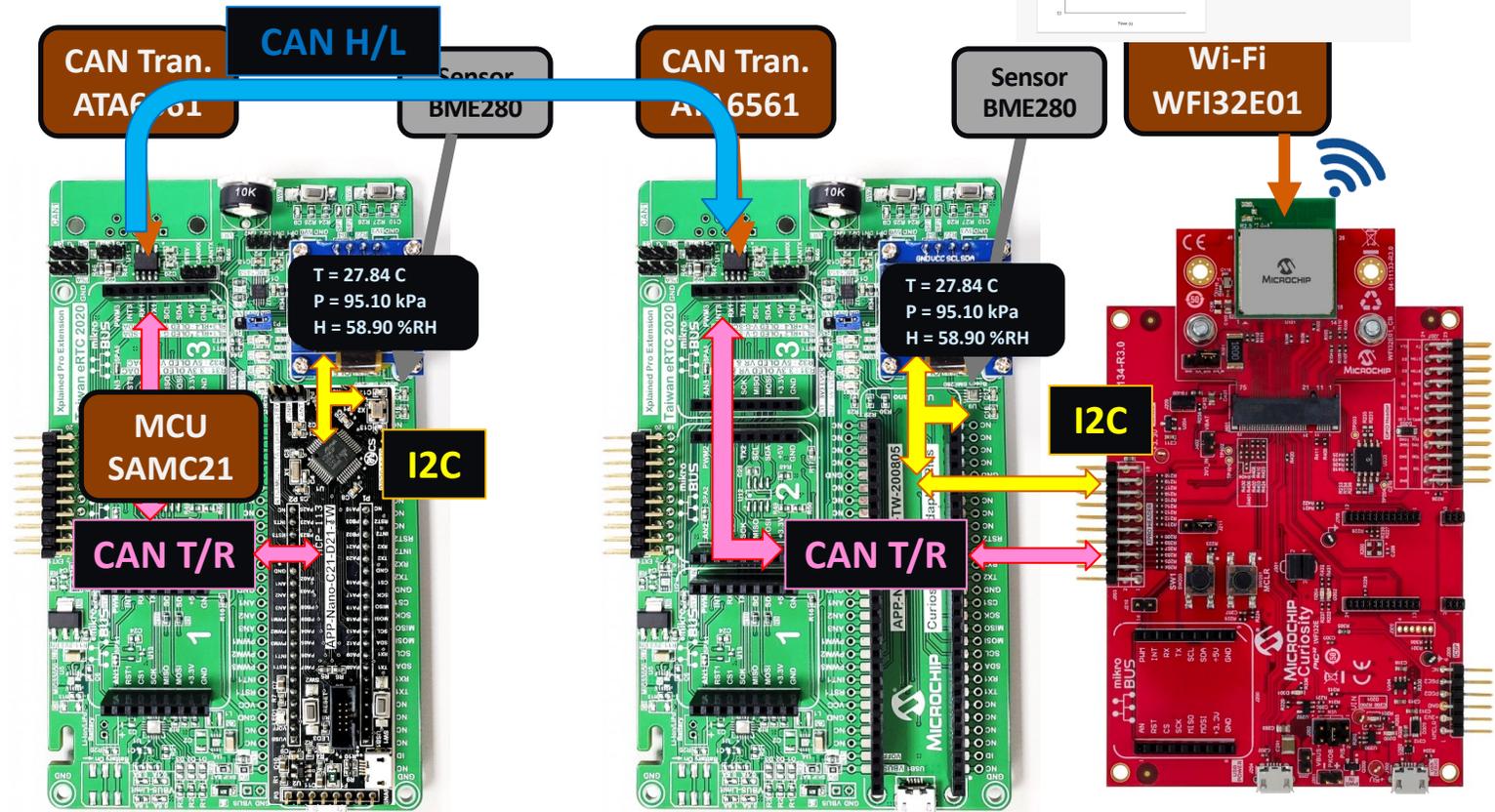
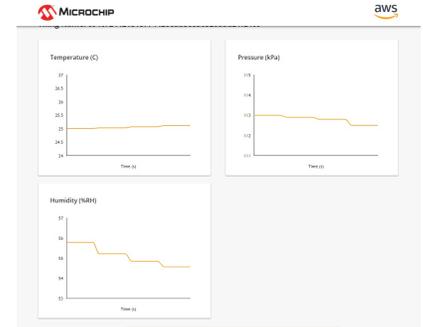
- CAN BUS基礎 & Harmony – 課程講解
- CAN BUS基礎 & Harmony (使用ATSAMC21) – 實驗講解
- ATSAMC21G17A-101 跟著Microchip使用APP-Nano-C21-D21-TW 為這個世界帶來更多的樂趣1 ~ 6

<https://youtu.be/z3wg4OjW0Vs>

 <p>41:32</p>	 <p>1:07:55</p>	 <p>31:12</p>	 <p>28:10</p>
CAN BUS基礎 & Harmony (使用ATSAMC21) – 實驗...	CAN BUS基礎 & Harmony – 課程講解	ATSAMC21G17A-101——跟 著Microchip使用APP-Nano...	ATSAMC21G17A 101——跟 著Microchip使用APP Nano...
 <p>43:03</p>	 <p>29:43</p>	 <p>38:25</p>	 <p>53:47</p>
ATSAMC21G17A 101——跟 著Microchip使用APP Nano...	ATSAMC21G17A 101——跟 著Microchip使用APP Nano...	ATSAMC21G17A 101——跟 著Microchip使用APP Nano...	ATSAMC21G17A 101——跟 著Microchip使用APP Nano...

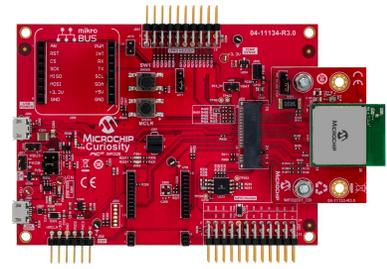
Wi-Fi to CAN Bus Bridge

Application & System Block Diagram

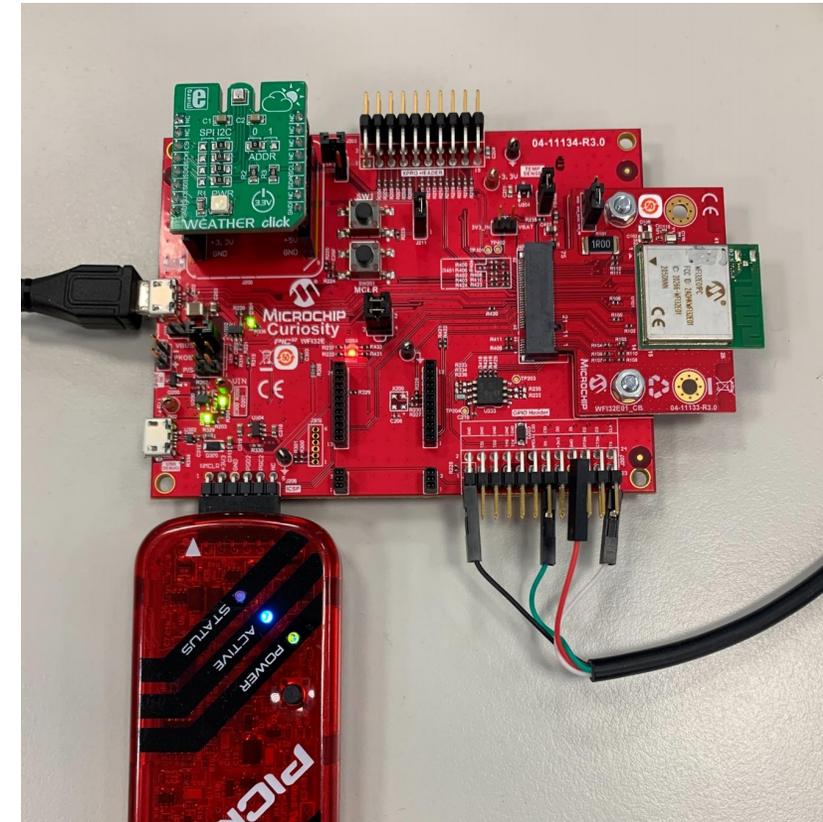


Wi-Fi to CAN Bus Bridge

PIC32WFI32E Curiosity Board - Environment Setup

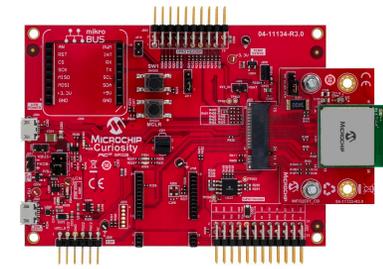


- Visit https://github.com/MicrochipTech/PIC32MZW1_Workshop/tree/master/03_setup
- Follow the instructions to setup the PIC32MZW1 curiosity board and setup your development environment.
- Please install **MPLAB® X IDE v5.45** or above, otherwise you might not be able to open MPLAB® Harmony 3 Configurator in PIC32MZW1_Curiosity_OOB.



Wi-Fi to CAN Bus Bridge

PIC32WFI32E Curiosity Board - Environment Setup



- Download PIC32MZW1_Curiosity_OOB Example
 - Visit https://github.com/MicrochipTech/PIC32MZW1_Curiosity_OOB

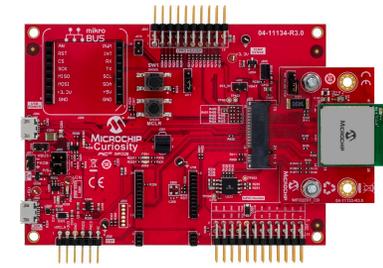
The screenshot shows a GitHub repository page for 'PIC32MZW1_Curiosity_OOB'. At the top, it indicates the current branch is 'master', with 2 branches and 7 tags. A 'Go to file' button and a green 'Code' button are visible. The repository is owned by 'Vysakh Pillai' and has a commit message 'Upgrade to latest H3 version'. The file list includes:

File/Folder	Commit Message
doc	Create .dummy
resources/media	Upgrade to latest H3 version
src/firmware	Upgrade to latest H3 version
.gitignore	update U2TX PPS for v2.3 boards
README.md	Upgrade to latest H3 version
faq.md	Update faq.md

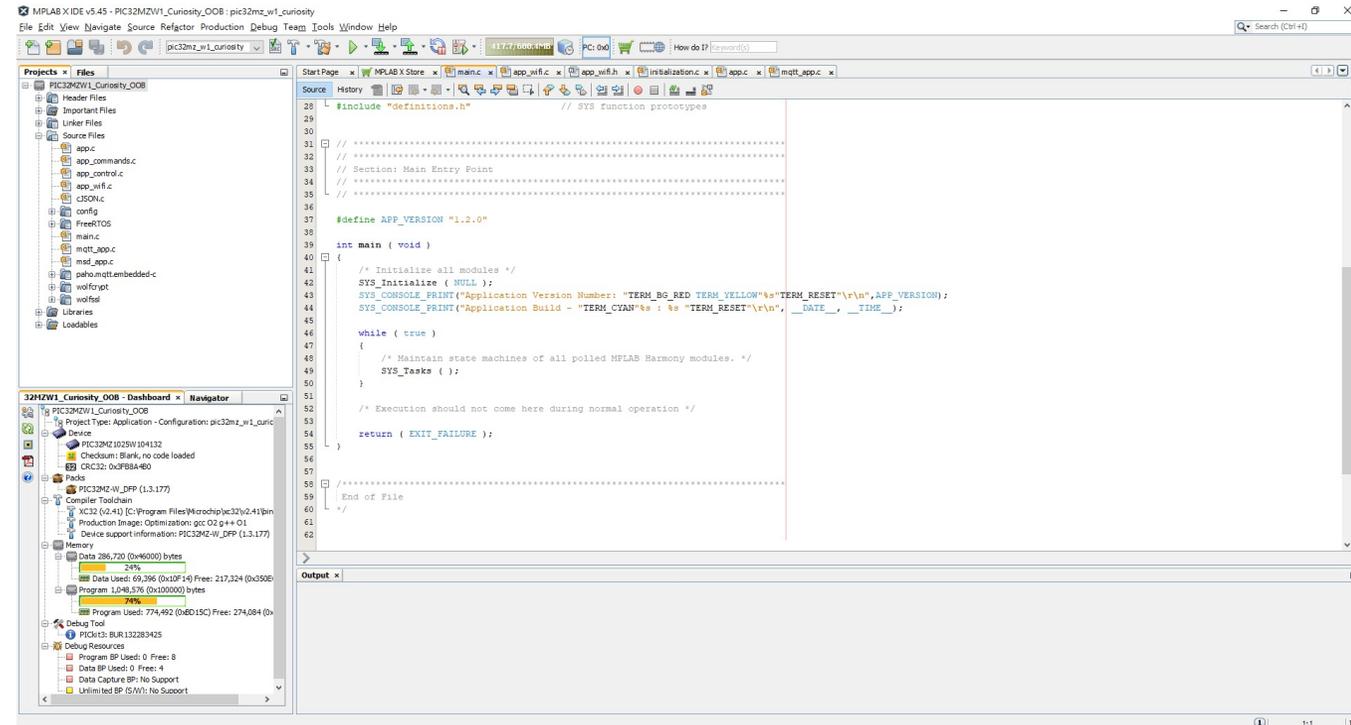
On the right, a 'Clone' dropdown menu is open, showing the 'HTTPS' option with the URL `https://github.com/MicrochipTech/PIC32`. Other options include 'Open with GitHub Desktop' and 'Download ZIP'. The repository was updated '4 months ago'.

Wi-Fi to CAN Bus Bridge

PIC32WFI32E Curiosity Board - Environment Setup

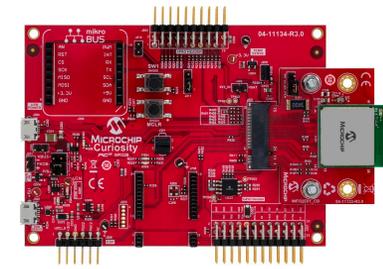


- Open the project by **MPLAB X IDE v5.45** or above
 - The project is located at
...\\PIC32MZW1_Curiosity_OOB-master\\src\\firmware**PIC32MZW1_Curiosity_OOB.X**

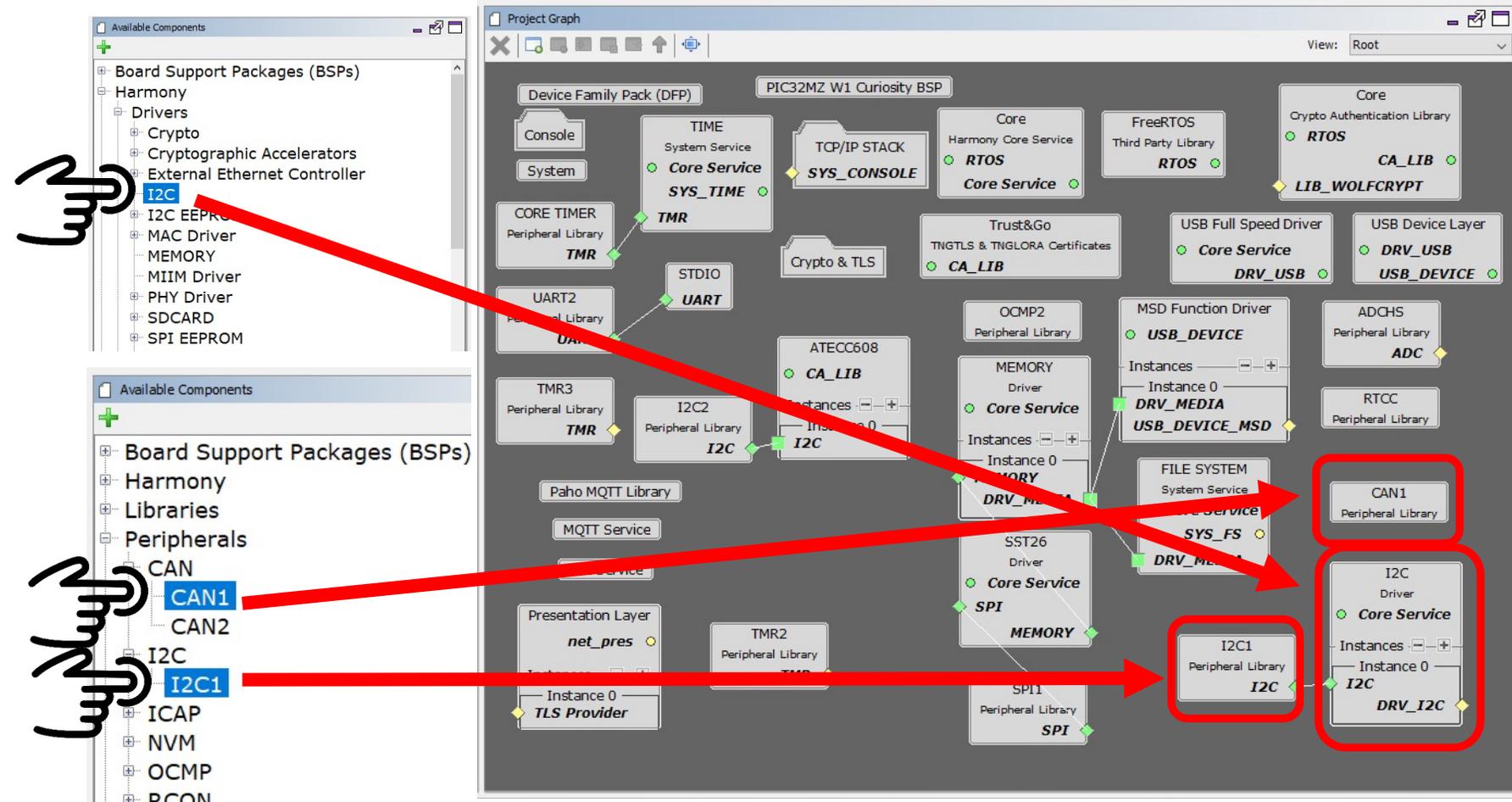


Wi-Fi to CAN Bus Bridge

PIC32WFI32E Curiosity Board - Harmony 3 Configurator

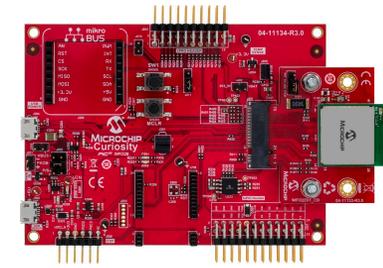


- Add I2C and CAN interfaces

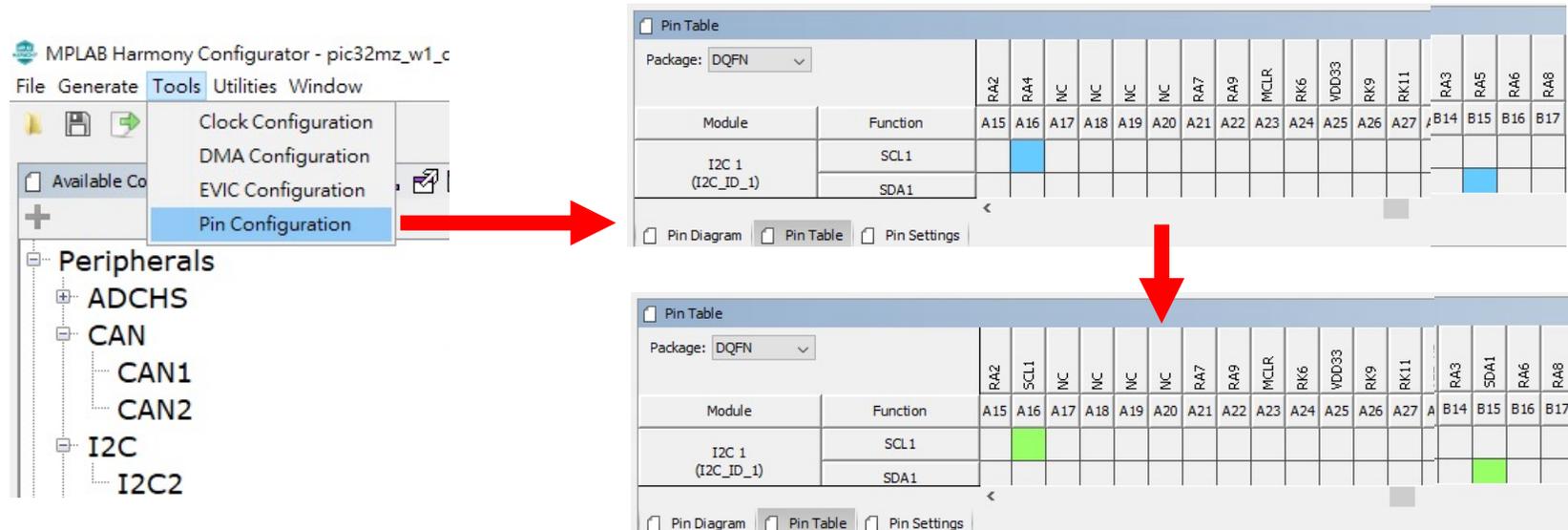
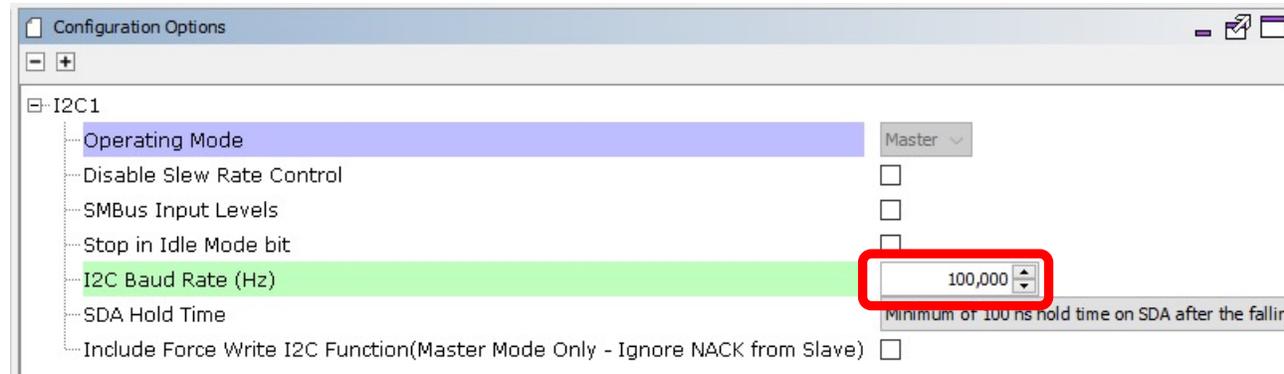


Wi-Fi to CAN Bus Bridge

PIC32WFI32E Curiosity Board - Harmony 3 Configurator

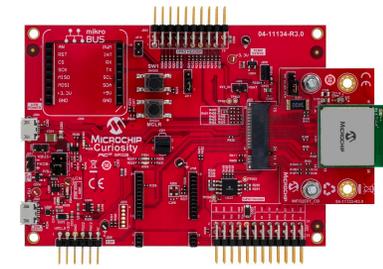


- Change I2C1 Baud Rate (Hz) to 100,000 and set I2C1 pins



Wi-Fi to CAN Bus Bridge

PIC32WFI32E Curiosity Board - Harmony 3 Configurator



- Change CAN1 Baudrate Prescaler to 9 and set CAN1 pins

Configuration Options

CAN1

- CAN Operation Mode: Set Normal Operation mode
- Interrupt Mode:
- Bit Timing Calculation
 - Clock Frequency: 100,000,000
 - Nominal Bit Timing
 - Automatic Nominal Bit Timing Calculation:
 - Bit Rate (Kbps): 500
 - Sample Point %: 75
 - Baudrate Prescaler: 9
 - Propagation Segment Time (ns): 300
 - Total Time Quanta (TQ): 10
 - Synchronization Jump Width (TQ): 3
 - Time Quanta (ns): 200.000
 - Calculated Bit Rate (Kbps): 500
 - Error %: 0.000
 - Sample of the CAN Bus Line bit: Bus line is sampled once at the sample p...
- FIFO Configuration
- Filter Configuration
- Acceptance Mask Configuration
- Timestamp Enable:

MPLAB Harmony Configurator - pic32mz_w1_c

File Generate Tools Utilities Window

- Clock Configuration
- DMA Configuration
- EVIC Configuration
- Pin Configuration

Available Co

Peripherals

- ADCHS
- CAN
 - CAN1
 - CAN2
- I2C
 - I2C2

Pin Table

Package: DQFN

Module	Function	RB12	RB10	NC	NC	NC	LED_RE...	LED_GR...	SWITCH..	RA12	RA14
CAN1 (CAN_ID_1)	C1RX										
	C1TX										

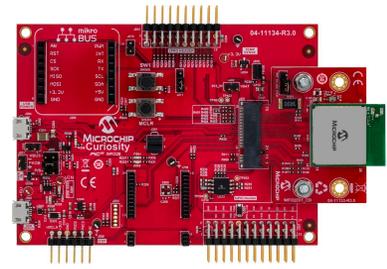
Pin Table

Package: DQFN

Module	Function	C1TX	RB10	NC	NC	NC	LED_RE...	LED_GR...	SWITCH..	RA12	CLRX
CAN1 (CAN_ID_1)	C1RX										
	C1TX										

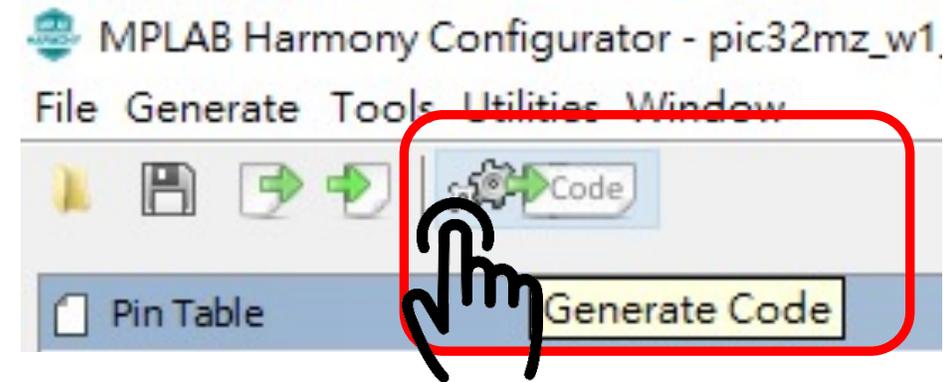
Wi-Fi to CAN Bus Bridge

PIC32WFI32E Curiosity Board - Harmony 3 Configurator



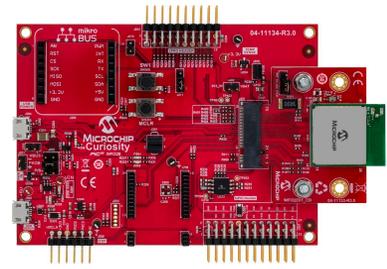
- **Generate Code**

- Please make sure to Generate Code whenever you do any modification, otherwise, the code will not be generated!



Wi-Fi to CAN Bus Bridge

PIC32WFI32E Curiosity Board - Add Application Code



```
MPLAB X IDE v5.45 - PIC32MZW1_Curiosity_OOB : pic32mz_w1_curiosity
File Edit View Navigate Source Refactor Production Debug Team Tools Window Help
pic32mz_w1_curiosity 667.6 / 18.16 MHz MDC PC: 0x0 How do I? Search (Ctrl+F)

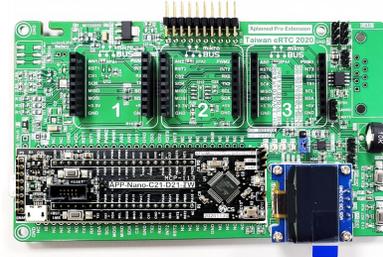
Projects x Files
PIC32MZW1_Curiosity_OOB
  Header Files
  Linker Files
  Source Files
  app.c
  app_commands.c
  app_control.c
  app_wifi.c
  cJSON.c
  config
  FreeRTOS
  main.c
  mqtt_app.c
  msd_app.c
  paho.mqtt.embedded-c
  wolfcrypt
  wolfssl
  Libraries
  Loadables

app_control.c
447 void APP_CONTROL_Tasks(void) {
448     uint32_t status = 0;
449     uint16_t tempValue = 0;
450     uint8_t rx_message[8];
451     uint32_t rx_messageID = 0;
452     uint8_t rx_messageLength = 0;
453     //uint8_t length = 0;
454     CAN_MSG_RX_ATTRIBUTE msgAttr = CAN_MSG_RX_DATA_FRAME;
455     static int counter = 0;
456
457     WDT_Clear();
458     switch (app_controlData.state) {
459     case APP_CONTROL_STATE_INIT:
460     {
461         ADCBS_CallbackRegister(ADCS_CH15, ADC_ResultHandler, (uintptr_t) NULL);
462         TMR3_Start(); /*TMR3 is used for ADC trigger*/
463
464         RTCC_CallbackRegister(RTCC_Callback, (uintptr_t) NULL);
465         setup_rtcc();
466         indicator_on();
467
468         /* Open I2C driver instance */
469         appData.drivI2CHandle = DRV_I2C_Open( DRV_I2C_INDEX_0, DRV_IO_INTENT_READWRITE);
470
471         OLED_Init();
472         OLED_CLS();
473
474         app_controlData.state = APP_CONTROL_STATE_MONITOR_CONNECTION;
475         break;
476     }
477     case APP_CONTROL_STATE_MONITOR_CONNECTION:
478     {
```

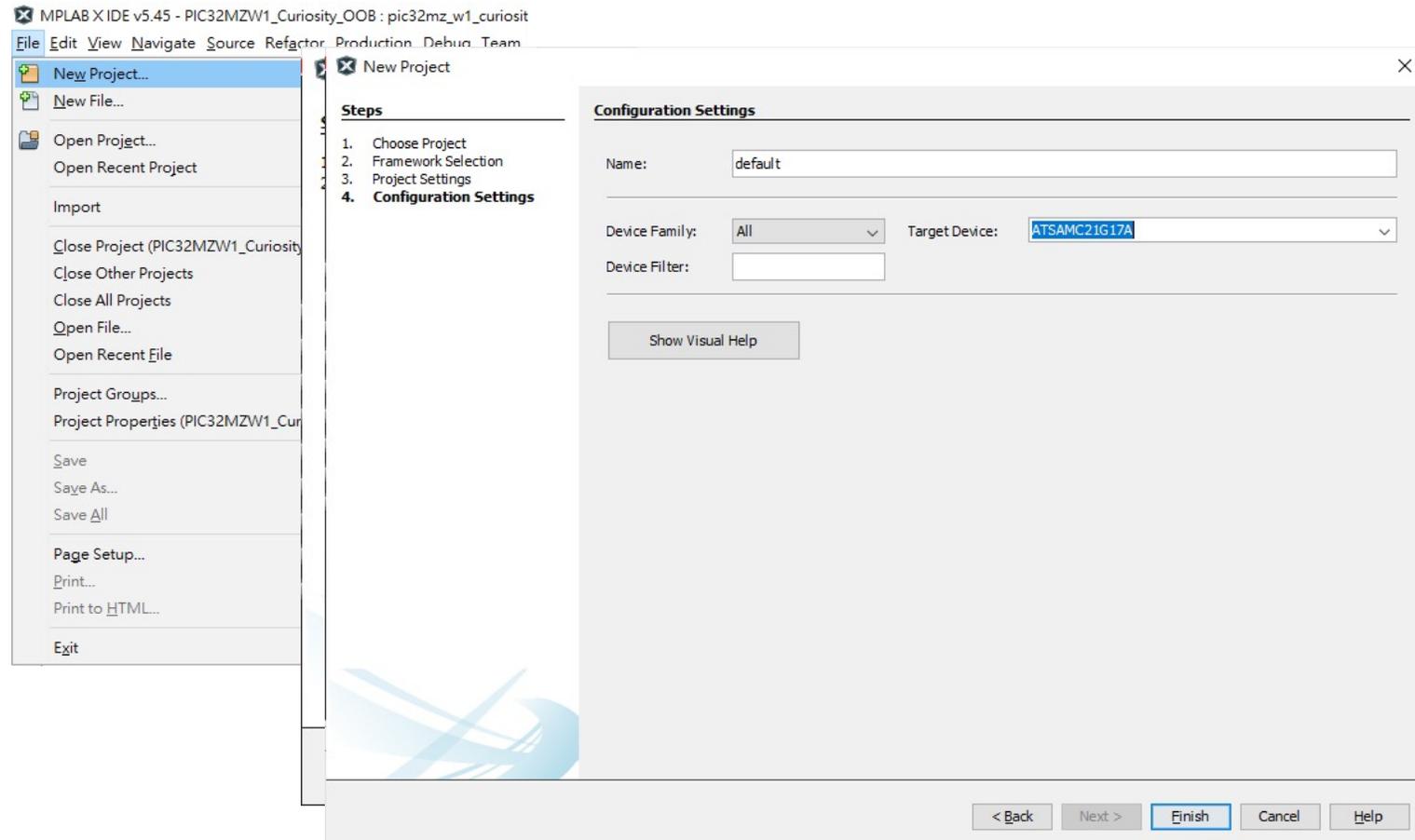
Output x Call Stack
Configuration Loading Error x Atmel-ICE x PIC32MZW1_Curiosity_OOB (Clean, Build, ...) x
BUILD SUCCESSFUL (total time: 1m 19s)
Loading code from C:\Charley\Microchip\Trainings\20210311_WiFi32_SMC21\Demo_OOB_CAN_RX_CAN_RX_OLED\PIC32M01_Curiosity_OOB\sroffirmware\PIC32M01_Curiosity_OOB\dist\pic32mz_w1_curiosity\production\PIC32M01_Curiosity_OOB.X.production.hex ...
Program loaded with pack: PIC32M2_WIF1.1.3.177.Microchip
Loading completed

Wi-Fi to CAN Bus Bridge

SAMC21 Nano Board - Environment Setup

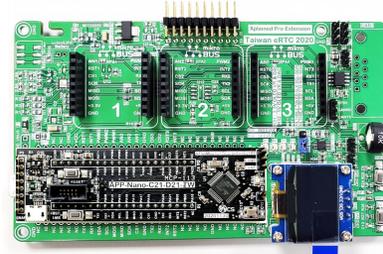


- Select: ATSAMC21G17A

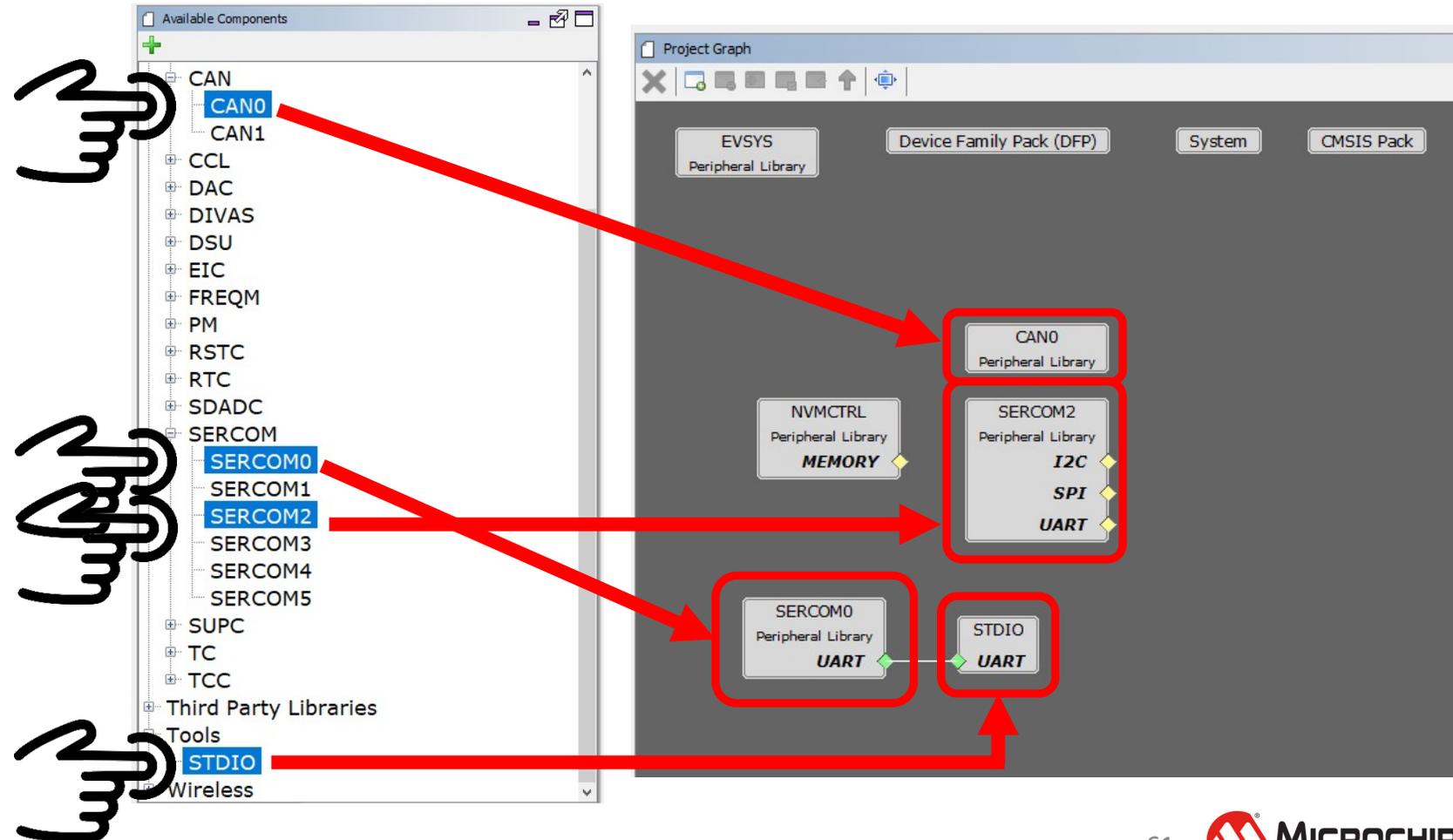


Wi-Fi to CAN Bus Bridge

SAMC21 Nano Board - Harmony 3 Configurator

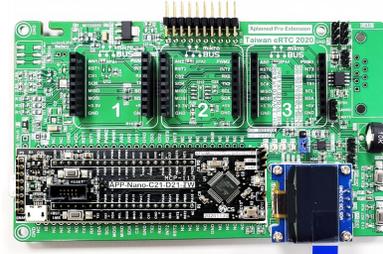


- Add I2C , UART, and CAN interfaces

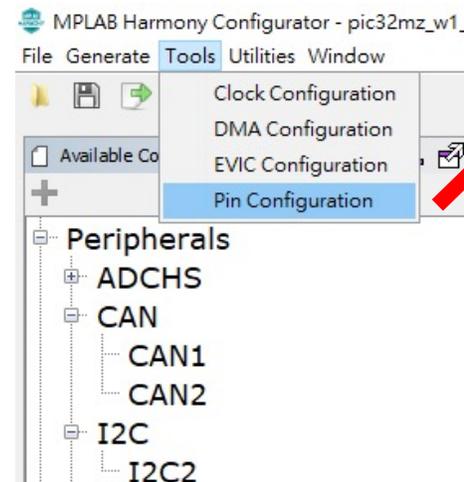
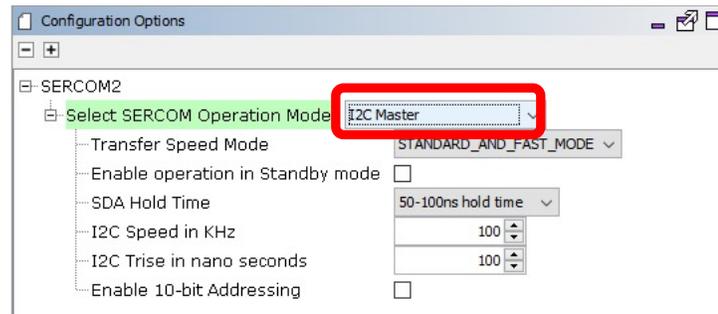


Wi-Fi to CAN Bus Bridge

SAMC21 Nano Board - Harmony 3 Configurator



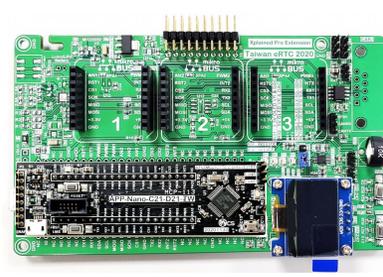
- Change SERCOM2 to I2C Master, and set SERCOM0, SERCOM2, and CAN0 pins
 - PA8 & PA9 are SERCOM0
 - PA12 & PA13 are SERCOM2
 - PA24 & PA25 are CAN0



Pin Number	Pin ID	Custom Name	Function	Mode	Direction	Latch	Pull Up	Pull Down	Drive Strength
1	PA00		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
2	PA01		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
3	PA02		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
4	PA03		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
5	GNDANA			Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
6	VDDANA			Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
7	PB08		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
8	PB09		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
9	PA04		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
10	PA05		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
11	PA06		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
12					High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
13	PA08	SERCOM0_P...	Digital	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
14	PA09	SERCOM0_P...	Digital	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
15					High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
16	PA11		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
17	VDDIO			Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
18	GNDIO			Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
19	PB10		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
20					High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
21	PA12	SERCOM2_P...	Digital	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
22	PA13	SERCOM2_P...	Digital	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
23					High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
24	PA15		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
25	PA16		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
26	PA17		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
27	PA18		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
28	PA19		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
29	PA20		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
30	PA21		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
31	PA22		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
32	PA23		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
33	PA24	CAN0_TX	Digital	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
34	PA25	CAN0_RX	Digital	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
35					High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
36	VDDIO			Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
37	PB22		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
38	PB23		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
39	PA27		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
40	RESET_N			Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
41	PA28		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
42	GNDIO			Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL

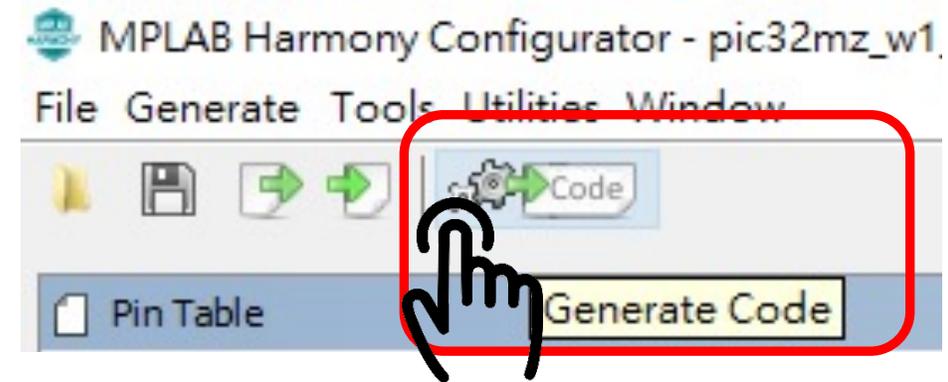
Wi-Fi to CAN Bus Bridge

SAMC21 Nano Board - Harmony 3 Configurator



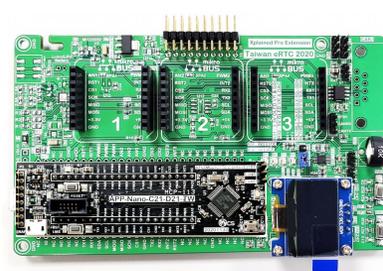
- **Generate Code**

- Please make sure to Generate Code whenever you do any modification, otherwise, the code will not be generated!



Wi-Fi to CAN Bus Bridge

SAMC21 Nano Board - Add Application Code



```
MPLAB X IDE v5.45 - can_demo_sam_c21n_xpro : sam_c21n_xpro
File Edit View Navigate Source Refactor Production Debug Team Tools Window Help
sam_c21n_xpro 806.3/115.4MHz MDC PC: 0x0 How do I? [Keyword(s)]

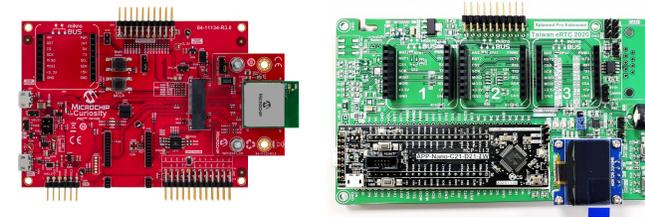
Projects x Files
can_demo_sam_c21n_xpro
  Header Files
  Important Files
  Linker Files
  Source Files
  config
  main.c
  Libraries
  Loadables
  PIC32MZ1_Curiosity_OOB

main.c x
Source History
200
201 int main ( void )
202 {
203     uint32_t status = 0;
204     uint16_t bitMask = 0x00FF;
205     uint8_t rx_message[8], tx_message[8];
206     uint32_t rx_messageID = 0, tx_messageID = 123;
207     uint8_t rx_messageLength = 0;
208     CAN_MSG_RX_FRAME_ATTRIBUTE msgFrameAttr = CAN_MSG_RX_DATA_FRAME;
209     uint32_t getDataDelay = 0, resetDelay = 0;
210     CALIBR_BME280_STATES calibrStage = CALIBR_BME280_T_P_WRITE;
211     uint8_t tempWrite = 0, writeStage = 0;
212     int tempTemp = 0, tempPress = 0, tempHumid = 0;
213
214     appData.state = APP_BME280_STATE_INIT;
215
216     /* Initialize all modules */
217     SYS_Initialize ( NULL );
218
219     SERCOM2_I2C_CallbackRegister(i2cEventHandler, 0);
220
221     /* Set Message RAM Configuration */
222     CAN0_MessageRAMConfigSet(Can0MessageRAM);
223
224     while ( true )
225     {
226         /* Check the application's current state. */
227         switch(appData.state)
228         {
229             case APP_BME280_STATE_INIT:
230             {
231                 /* Open I2C driver instance */
232                 appData.state = APP_BME280_STATE_READY_WAIT;
233             }
234         }
235     }
236 }

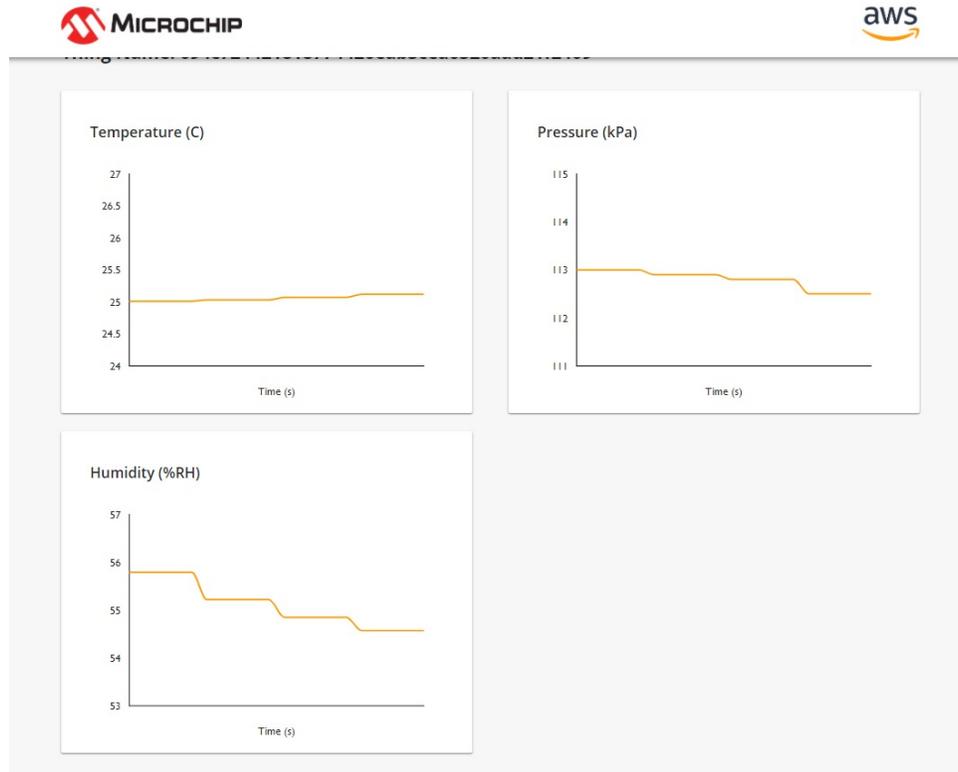
Output x Call Stack
Configuration Loading Error x Atmel-ICE x
Loading script file C:\Program Files\Microchip\MPLAB\v5.45\pack\Microchip\SAMC21_DFP\3.4.70\samc21\scripts\dap_cortex-m0plus.py
```

Wi-Fi to CAN Bus Bridge

Demo



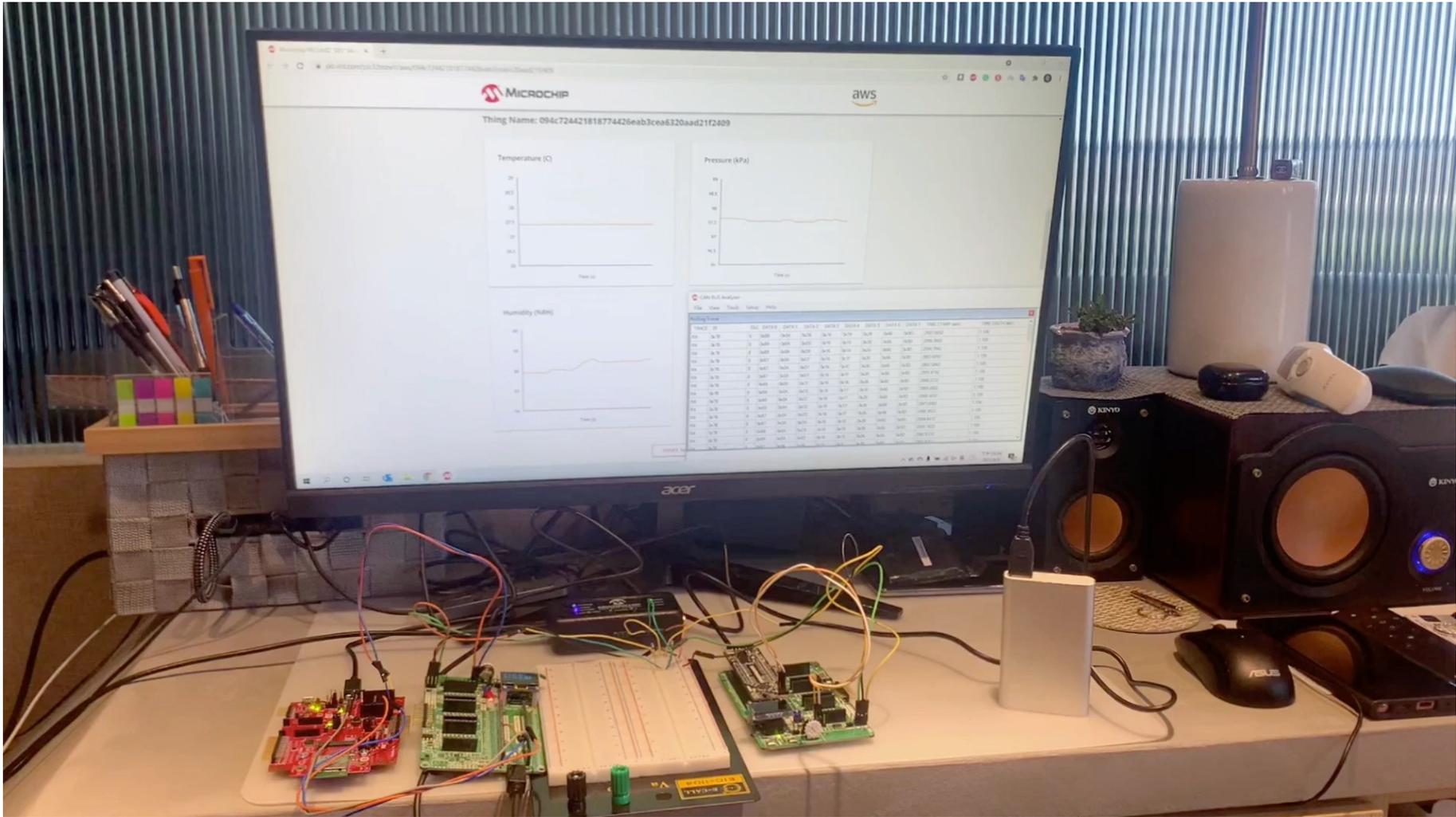
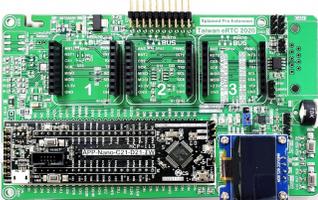
- Follow the instructions in https://github.com/MicrochipTech/PIC32MZW1_Curiosity_OOB
You can get the results below:



```
COM26 - Tera Term VT
File Edit Setup Control Window Help
Cloud config clientID "094c724421818774426eab3cea6320aad21f2409"
Found valid MQTT config
Device SerialNumber is : 0123F5062AF5440D01
TCPIP Stack Not UP
APP_WIFI: SSID is charleylin
APP_WIFI: Connecting to Wi-Fi
Temperature = 25.18 C Pressure = 112.15 kPa Humidity = 54.55 %RH
Temperature = 25.15 C Pressure = 112.28 kPa Humidity = 54.58 %RH
Temperature = 25.13 C Pressure = 112.39 kPa Humidity = 54.62 %RH
APP_WIFI: WiFi Connected
Temperature = 25.05 C Pressure = 112.83 kPa Humidity = 54.63 %RH
MqttCallback(): MQTT Connected
MqttCallback(): Subscribed to Topic '$aws/things/094c724421818774426eab3cea6320aad21f2409/shadow/update/#'
Temperature = 25.02 C Pressure = 112.96 kPa Humidity = 54.86 %RH
Temperature = 25.02 C Pressure = 112.97 kPa Humidity = 55.00 %RH
Temperature = 25.01 C Pressure = 113.03 kPa Humidity = 55.03 %RH
Temperature = 24.97 C Pressure = 113.24 kPa Humidity = 55.08 %RH
Temperature = 24.96 C Pressure = 113.29 kPa Humidity = 55.16 %RH
Temperature = 24.99 C Pressure = 113.13 kPa Humidity = 55.34 %RH
Temperature = 25.08 C Pressure = 112.66 kPa Humidity = 55.14 %RH
Temperature = 25.16 C Pressure = 112.24 kPa Humidity = 54.86 %RH
```

Wi-Fi to CAN Bus Bridge

Demo





MICROCHIP UNIVERSITY

Microchip has been delivering superior learning for our clients over the last 23 years at our annual users' conference. Our goal for the new online Microchip University Program is to provide you with all the same information you need to design robust embedded control systems with Microchip solutions.

All Microchip University courses are being offered for free at this time.

 All Courses Shows a listing of all courses currently available	 New Courses This group will highlight our newest courses.	 Development Tools and Compilers This group of classes will help you get up to speed on the MPLAB® X IDE and build a solid foundation with C programming and understand the nuances of the C compilers.	 Software Frameworks This group of classes will help you with software framework tools such as the MPLAB Code Configurator (MCC) and Harmony.	 Embedded System Design This category includes general system design classes on topics such as hardware design, Core Independent Peripherals (CIPs), bootloaders, clocking and timing, signal integrity and PCB design
 Programmable Logic, FPGAs and SOC This group will cover our FPGA and other programmable logic devices as well as SOC products.	 Embedded Linux This group of classes will take you all the way through the steps of setting up a Linux development computer, interfacing to sensors, creating graphics and on to building your own software distribution.	 Safety and Security Classes on Functional Safety topics as well as Security, Secure Boot and Trust Platform Devices	 Interface This group of classes covers topics on low-speed serial communication such as I2C, SPI and RS-232 as well as automotive/industrial communication protocols such as CAN and LIN	 Connectivity Courses in this group cover Networking, USB and Bluetooth
 IoT (Internet of Things) Wireless communication and other topics related to building networks to control multiple devices	 Analog and Sensors Courses on analog design, simulation and interfacing to sensors	 Motor Control These courses cover a broad range of motor control topics	 Power Management Courses on power supplies, power conversion, batteries and battery charging.	



Microchip - Chinese [Traditional]

199 位訂閱者

已訂閱



首頁

影片

播放清單

頻道

討論

簡介



上傳的影片

▶ 全部播放

Switch-Node Layout

- High Side Drain Traces
- High Side Drain Trace inductance must be matched to drain/source capacity
- Gate trace impedance is critical and must be balanced
- Gate registers will add inductance and may produce the problem you try to solve...
- High side gate drive trace and switch-node-to-bootstrap cap trace has major influence on speed and ring

1:34:49

ePOW007—電源分佈網路與電源完整性

觀看次數：29次 · 3 週前

eLearning MICROCHIP

練習5: 为你的ATSAMC21G17A 加入CAN的傳送功能

43:03

ATSAMC21G17A 101—跟著Microchip使用APP Nano...

觀看次數：97次 · 1 個月前

eLearning MICROCHIP

練習4: 加入MCP9800溫度Sensor的讀取並顯示於OLED第三行

29:43

ATSAMC21G17A 101—跟著Microchip使用APP Nano...

觀看次數：28次 · 1 個月前

SSD1306 GDDRAM的放大圖

每一頁Page有128個Bytes的資料，控制位28*8個點，總共有8個Pages = 128 * 64點！

38:25

ATSAMC21G17A 101—跟著Microchip使用APP Nano...

觀看次數：31次 · 1 個月前

eLearning MICROCHIP

練習2: 中斷操作——EIC以及TC的中斷操作法 使用PLUS & callback

53:47

ATSAMC21G17A 101—跟著Microchip使用APP Nano...

觀看次數：39次 · 1 個月前

28:10

ATSAMC21G17A 101—跟著Microchip使用APP Nano...

觀看次數：27次 · 1 個月前

Featured Channels



Microchip Technology

5.38萬 位訂閱者



Microchip - Chinese [Simplified]

540 位訂閱者



Microchip - Japanese

343 位訂閱者



Microchip - Korean

1310 位訂閱者



Microchip Makes

1.87萬 位訂閱者

eLearning

Wi-Fi® SoC Module *WFI32E01* 參考範例應用篇

Thank you!