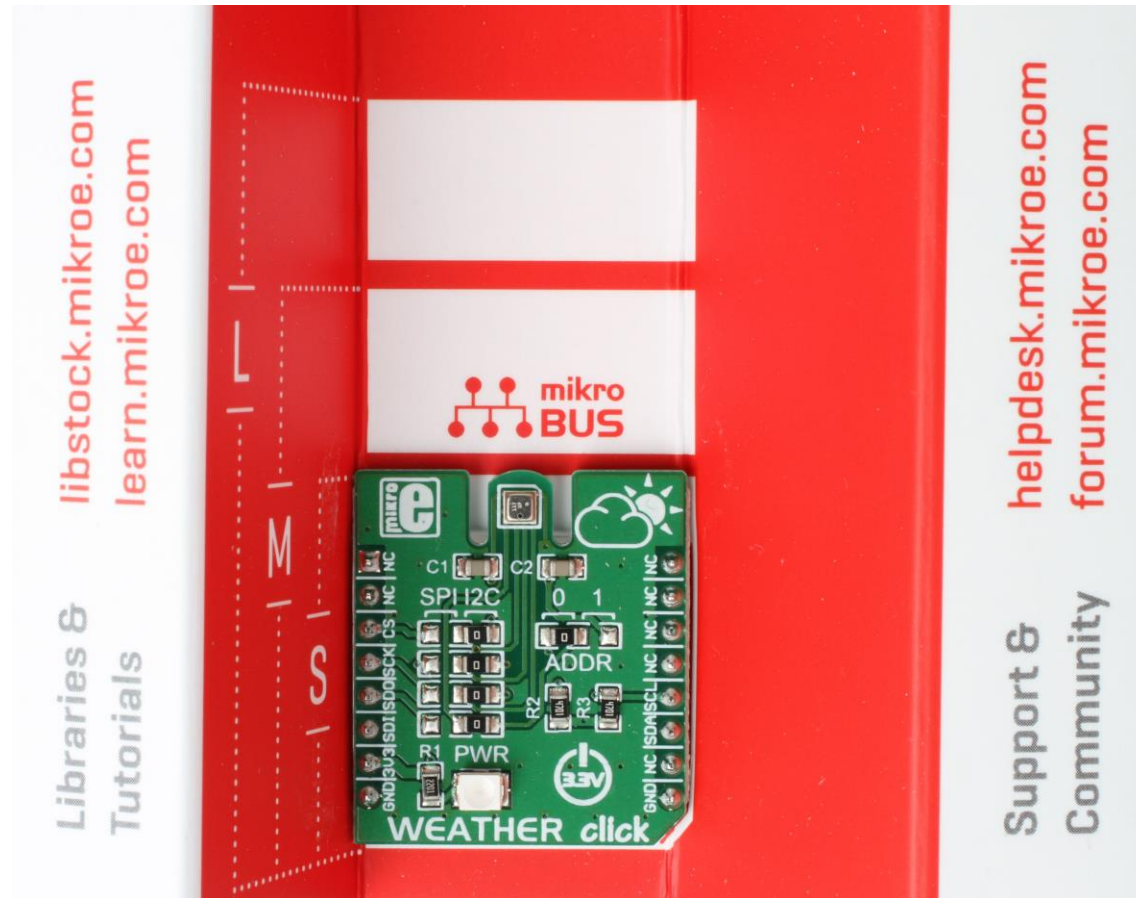


IoT Development Tools for PIC32



EasyPIC Fusion v7

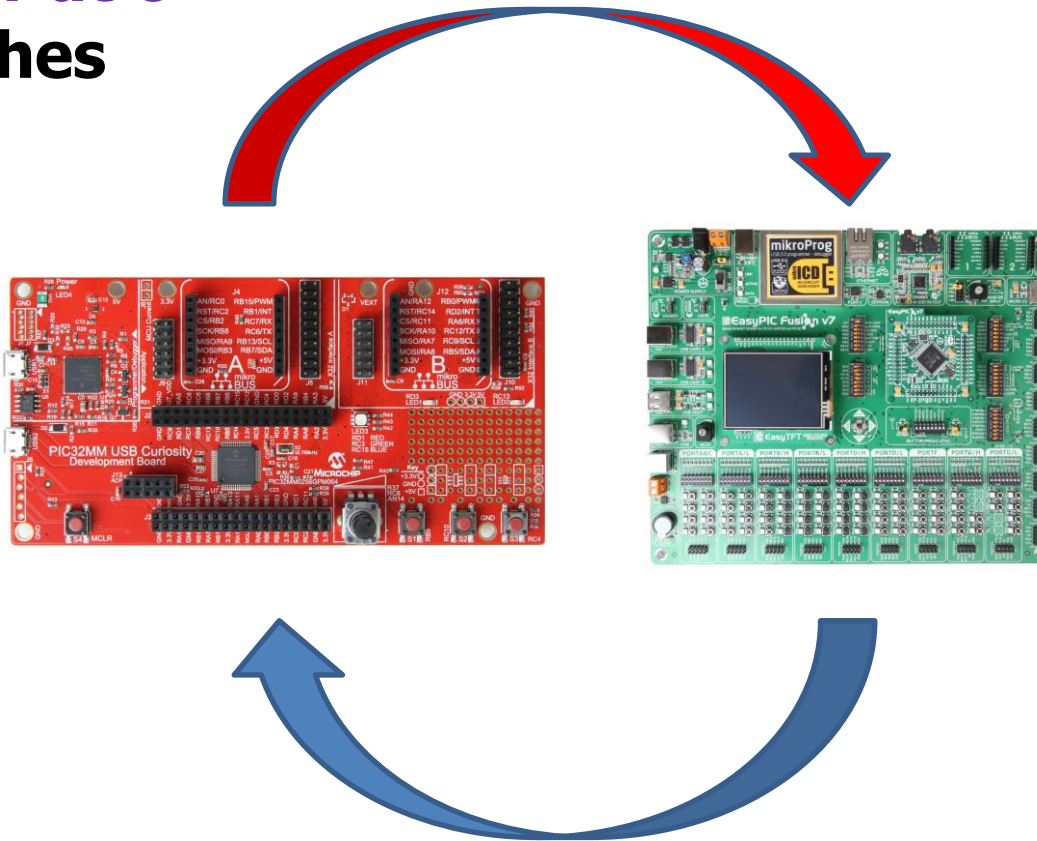
February 1, 2018

FRED EADY

IoT Development Tools for PIC32

AGENDA

- **Hardware**
- **Bluetooth and a PIC32MM USB Curiosity**
- **The Weather at 6**
- **Buenas Noches**

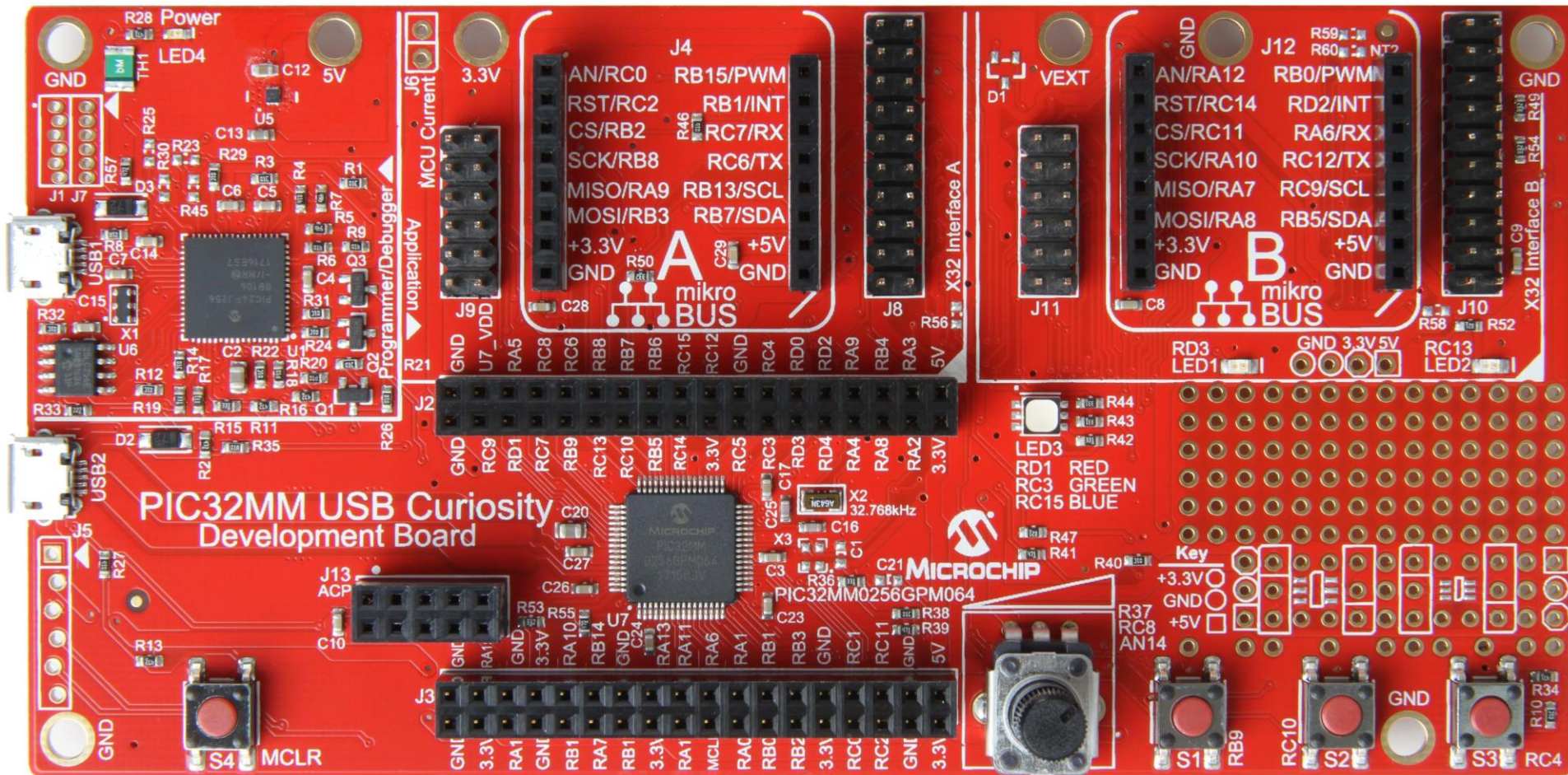


Hardware – click Part Numbers



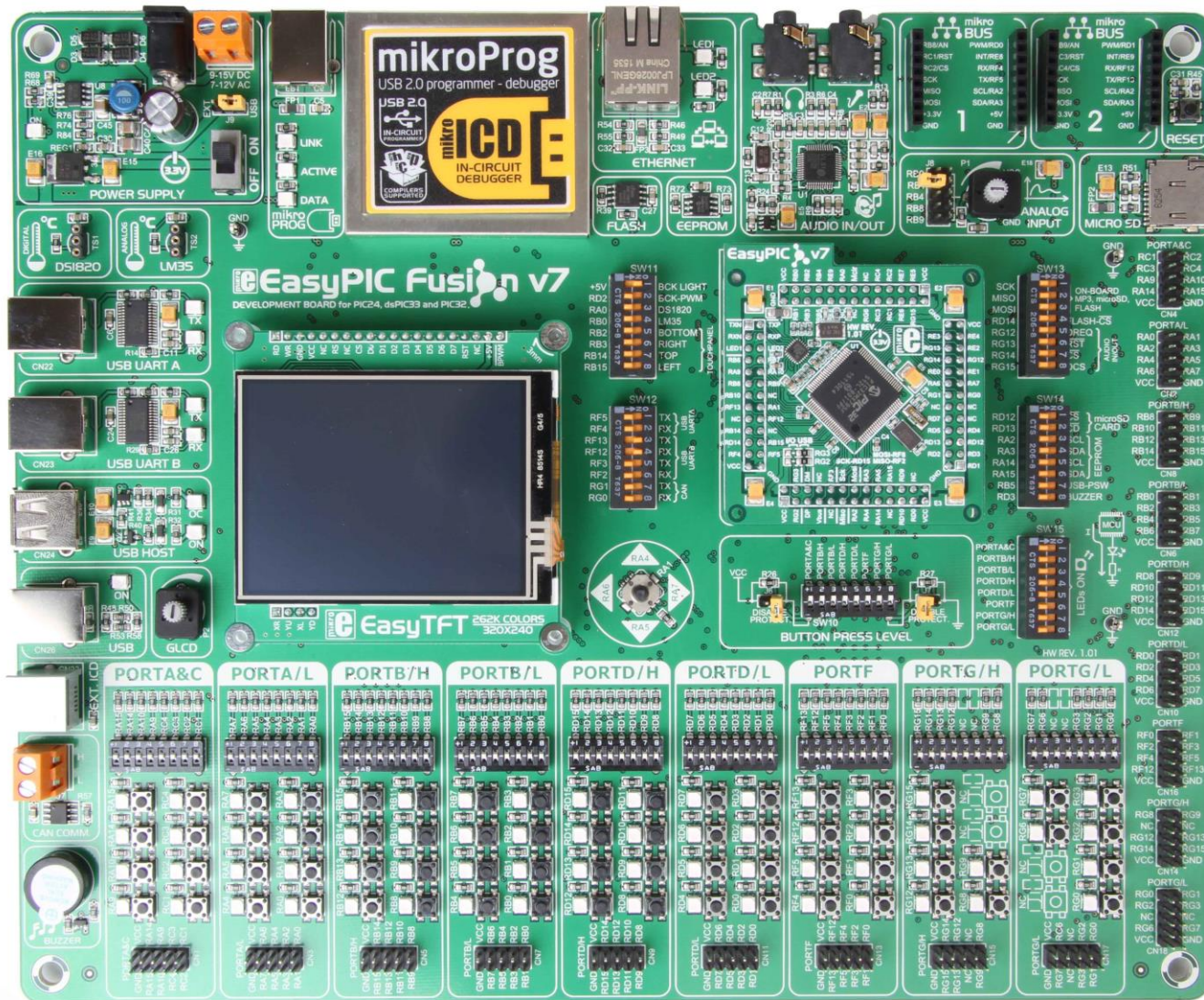
IoT Development Tools for PIC32

Hardware – PIC32MM USB Curiosity



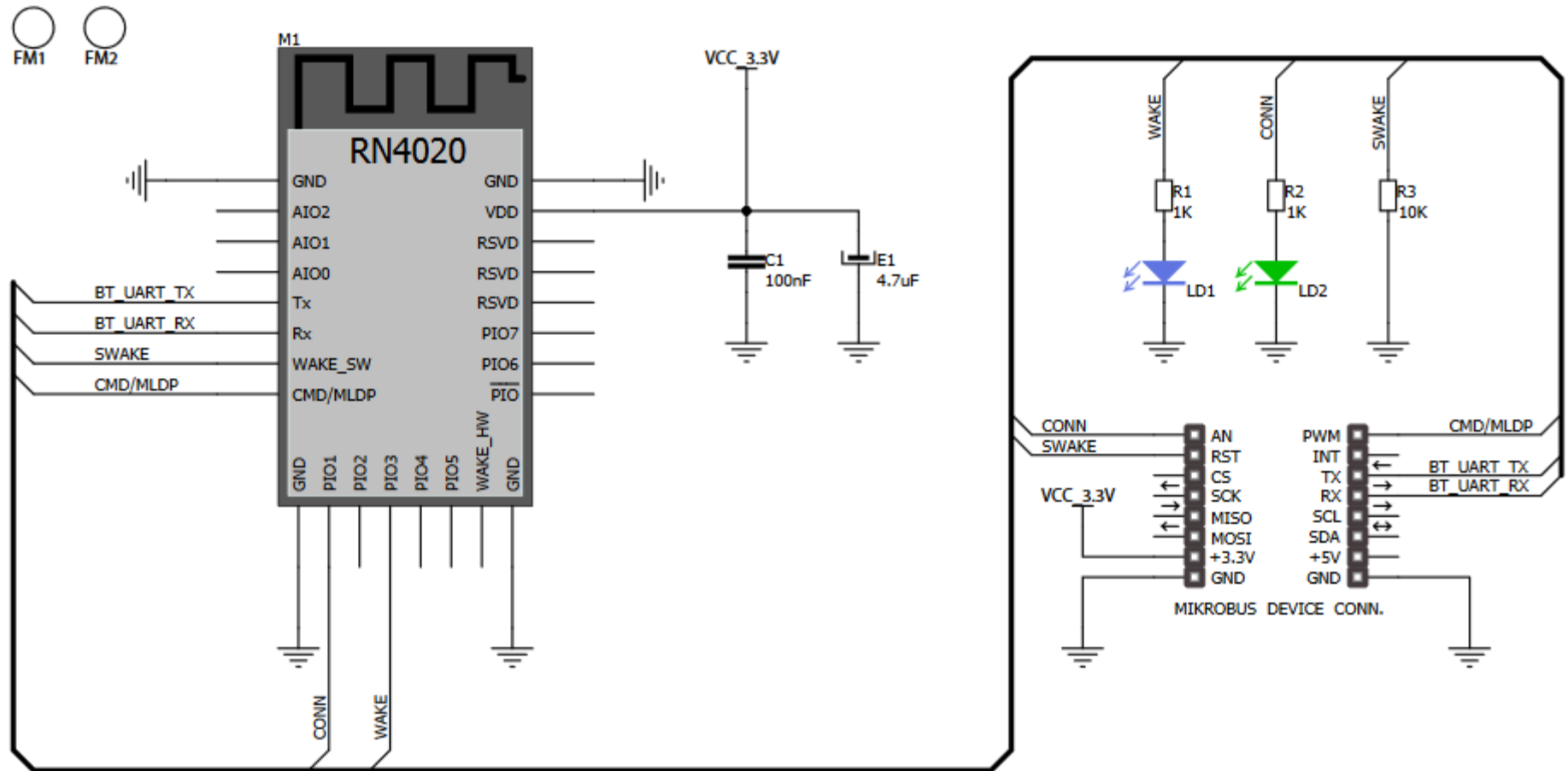
IoT Development Tools for PIC32

Hardware – EasyPIC Fusion v7



IoT Development Tools for PIC32

Bluetooth and a PIC32MM USB Curiosity



IoT Development Tools for PIC32

Bluetooth and a PIC32MM USB Curiosity

MPLAB X IDE v4.05

File Edit View Navigate Source Refactor Production Debug Team Tools Window Help

default

How do I? Keyword(s)

Projects Files Services Resource Management [MCC] x

Project Resources

Generate Import... Export

System Module

Libraries

Foundation Services

Peripherals

UART1 [Foundation Services Library by Microchip Technology, Inc.]

Mikro-E Clicks

Wireless Connectivity

BLE2

Device Resources

Documents

PIC32MM0256GPM064 Product Page

Peripherals

ADC

CDAC

CLC

Comparator

DMA with CRC

versions

p32MM_BLUonly - Dashboard Navigator Versions [MCC] x

MPLAB® Code Configurator (Plugin) v3.45.1

Libraries

Microchip Technology, Inc.

Microcontrollers and Peripherals

PIC10 / PIC12 / PIC16 / PIC18 MCUs (v1.55)

PIC24 / dsPIC33 / PIC32MM MCUs (v1.45)

PIC32MX MCUs (v1.35)

Software

8-bit Bootloader Library (v2.2.0)

CoAP Library

Dac Library

Foundation Services Library (v0.1.23)

LIN Library (v2.2)

LoRaWAN Library

MCP794XX I2C RTCC

MikroElektronika Click Library (v1.0.26)

mTouch Capacitive Sensing Library (v2.30)

TCP/IP Lite Stack

Pin Module x System Module x Interrupt Module x BLE2 x

Information Configuration Advanced Settings

Description

BLE2 click features the RN4020 module from Microchip, that integrates RF, a baseband controller, and a command API processor, making it a complete Bluetooth Low Energy solution. The click communicates with the target board MCU through mikroBUS™ RX, TX and AN (CMD), PWM (con.), and RST (wake) lines. The board is designed to use 3.3V power supply only.

This click is EUSART ONLY.

Product Page

<https://shop.mikroe.com/click/wireless-connectivity/ble-2>

Pin Manager: Package View x

PIN 64 63 62 61 60 59 58 57 56 55 54 53 52 51 50 49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1

RA10 RB3 RB11 RB10 RA14 RA15 VDD1 VCAP RC9 RA5 RD1 RC8 RC7 RC6 RB9 RA7 RB14 RB15 AVSS1 AVDD1 RA13 RA12[BLE2_Conn]Conn RA11 NMCLR RA6[UIRX] RA0[PGC2] RA1[PGD2] RB0[BLE2_Cmd_Mldp]Cmd_Mldp RB1 RB2 RB3 VDD3 VSS3 RC1 RC2 VDD4 VSS4 CLKO RA2 RA3 RA4 RA5 RA6 RA7 RA8 RA9 RD4 RD5 RD6 RD7 RD8 RD9 RD10 RD11 RD12 RD13

MICROCHIP

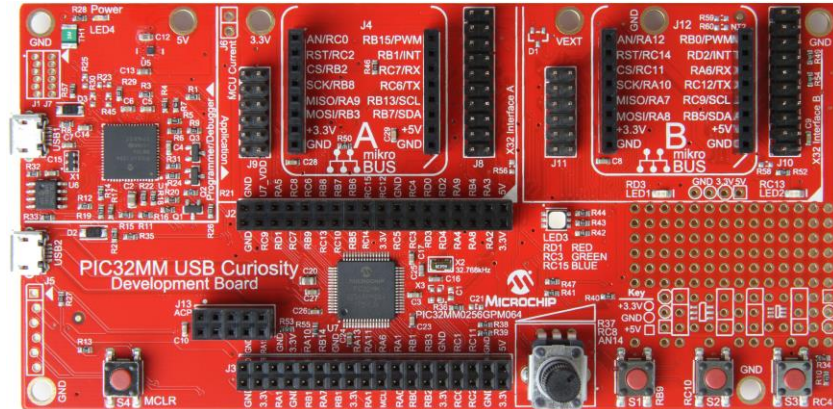
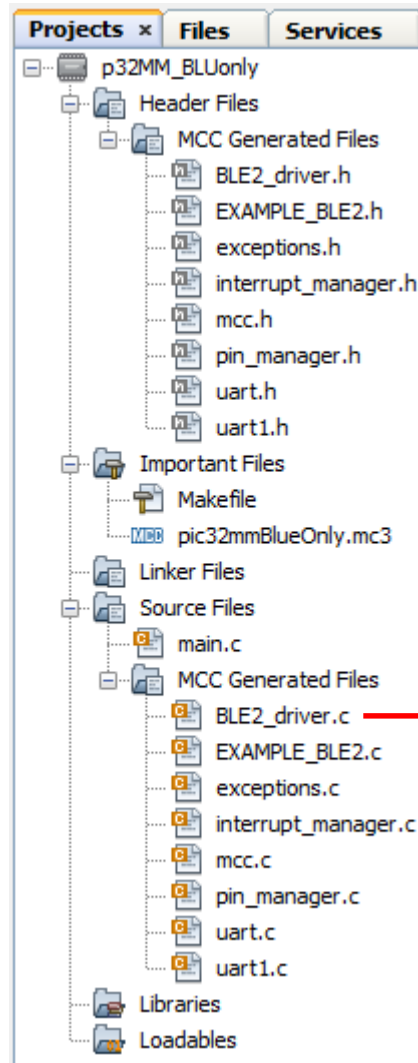
PIC32MM0256GPM064

Output - MPLAB® Code Configurator Notifications [MCC] Pin Manager: Grid View x

Package:	QFN64	Pin No:	11	12	25	26	29	54	10	1	27	30	64	8	7	6	59	58	13	14	15	16	28	44	46	48	49	60	61	63	2	3	19	20	21	35	36	37	50	51	52	55	45	
Module	Function	Direction	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	0	1	2	3	4	5	6	7	8	9	10	11	13	14	15	0	1	2	3	4	5	6	7	8	9	10
BLE2	Cmd_Mldp	output																																										
	Conn	input																																										
	Wake	output																																										
ICD	PGCx	input																																										
	PGDx	input																																										
	CLKI	input																																										
	CLKO	output																																										

IoT Development Tools for PIC32

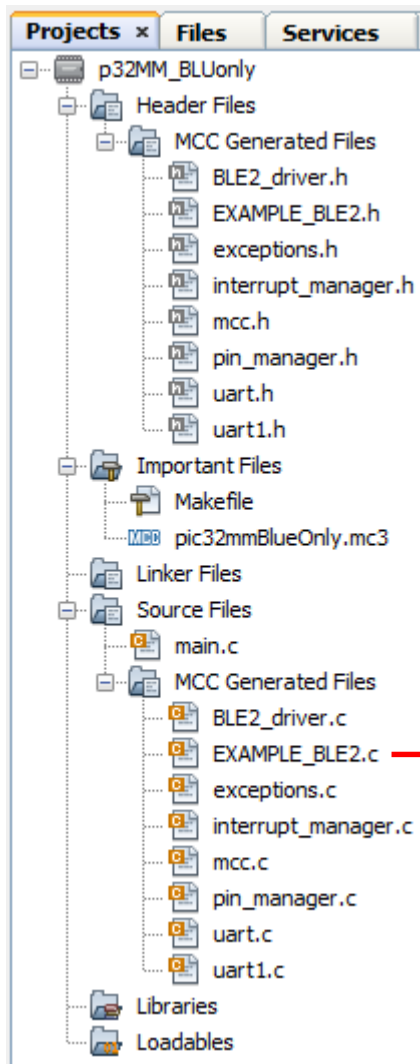
Bluetooth and a PIC32MM USB Curiosity



```
52 void BLE2_SendBuffer(uint8_t *buffer, uint8_t length)
53 {
54     while(length-->0)
55         BLE2_SendByte(*buffer++);
56 }
57 void BLE2_SendByte(uint8_t byte)
58 {
59     uart[BLE2].Write(byte);
60 }
```


IoT Development Tools for PIC32

Bluetooth and a PIC32MM USB Curiosity



```
85 void EXAMPLE_setupBLE2 (const char* name)
86 {
87     char* exampleRead;
88     uart[BLE2].SetRxISR(EXAMPLE_CaptureReceivedMessage);
89     BLE2_WakeModule(); // Wake Module using GPIO
90     EXAMPLE_blockingWait(200); // Wait for "CMD" Response
91     exampleRead = EXAMPLE_GetResponse(); // Read "CMD\r\n" from Buffer
92     strcpy(exampleReadStorage, exampleRead); // Store for use/reference
93     EXAMPLE_ReadyReceiveBuffer(); // Prepare for next message
94     BLE2_EnterCommand_Mode(); // Enter "Command Mode" via GPIO
95     EXAMPLE_sendAndWait("SF,1\r\n"); // Factory Reset
96     EXAMPLE_sendAndWait("SR,30000800\r\n"); // Setup Services
97     BLE2_SendString("S-,"); // Set Serialized Name
98     BLE2_SendString(name);
99     EXAMPLE_sendAndWait("\r\n");
100    BLE2_SendString("SN,"); // Set Name
101    BLE2_SendString(name);
102    EXAMPLE_sendAndWait("\r\n");
103    EXAMPLE_sendAndWait("A\r\n"); // Advertise
104    EXAMPLE_sendAndWait("R,1\r\n"); // Reset
105    EXAMPLE_blockingWait(120);
106    BLE2_EnterCommand_Mode(); // Enter "Command Mode" via GPIO
107    EXAMPLE_sendAndWait("B\r\n"); // Bond
108    EXAMPLE_sendAndWait("WC\r\n"); // Clear Scripts
109 }
```

IoT Development Tools for PIC32

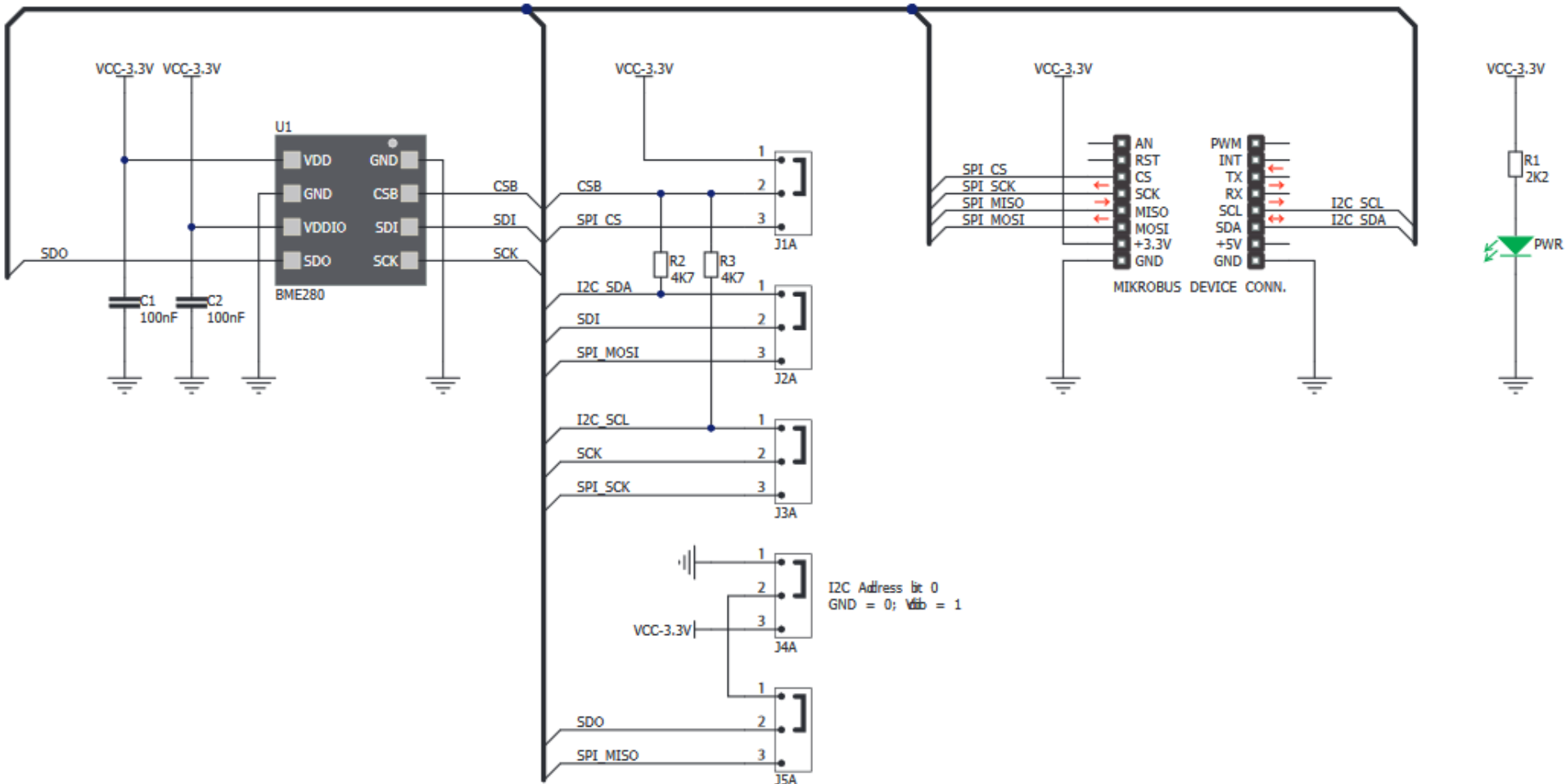
Bluetooth and a PIC32MM USB Curiosity

```
75 uint8_t message[6] = {"Freddy"};
76 int main(void)
77 {
78     // initialize the device
79     SYSTEM_Initialize();
80
81     // When using interrupts, you need to set the Global Interrupt Enable bits
82     // Use the following macros to:
83
84     // Enable the Global Interrupts
85     INTERRUPT_GlobalEnable();
86
87     // Disable the Global Interrupts
88     //INTERRUPT_GlobalDisable();
89     EXAMPLE_setupBLE2 ("Blue");
90     while (1)
91     {
92         // Add your application code
93         EXAMPLE_sendMessageOverBLE2(message);
94         ctDelaysms(1000);
95     }
96 }
```



IoT Development Tools for PIC32

The Weather at 6



IoT Development Tools for PIC32

The Weather at 6

The screenshot displays the mikroC PRO for PIC32 Package Manager v.3.6.0 interface. The 'Click_BLE2' package is selected, showing its details in the 'General Info' tab. The package is a dependency for the 'BLE2_hal' and 'BLE2_hw' libraries. The target compiler is set to 'mikroC PRO for PIC32'. The package name is 'Click_BLE2', the author is 'Viktor Milovanovic', the contact email is 'viktor@mikroe.com', and the web page is 'www.mikroe.com'. The package description is empty. The package information section at the bottom states: 'Basic package info: select compiler and enter name, author, email address, web page and description for package.'

Target compiler: mikroC PRO for PIC32
Package name: Click_BLE2
Author: Viktor Milovanovic
Contact email address: viktor@mikroe.com
Web page: www.mikroe.com

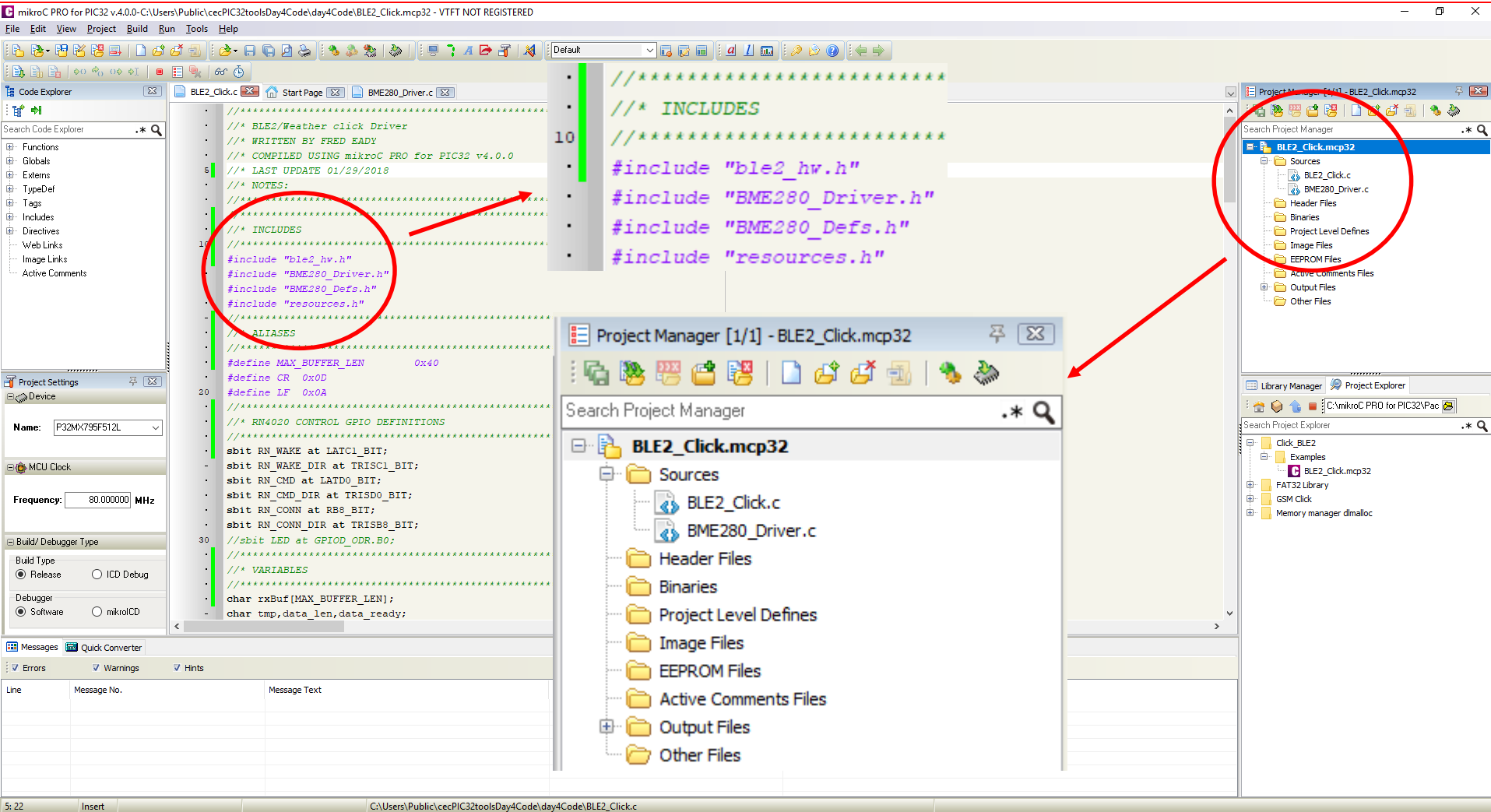
Package description:

Package information: Basic package info: select compiler and enter name, author, email address, web page and description for package.

Target compiler: mikroC PRO for PIC32 Package name: C:\Users\Fred\Downloads\1463149011_ble2_click_mikroc_pic32.mpkg

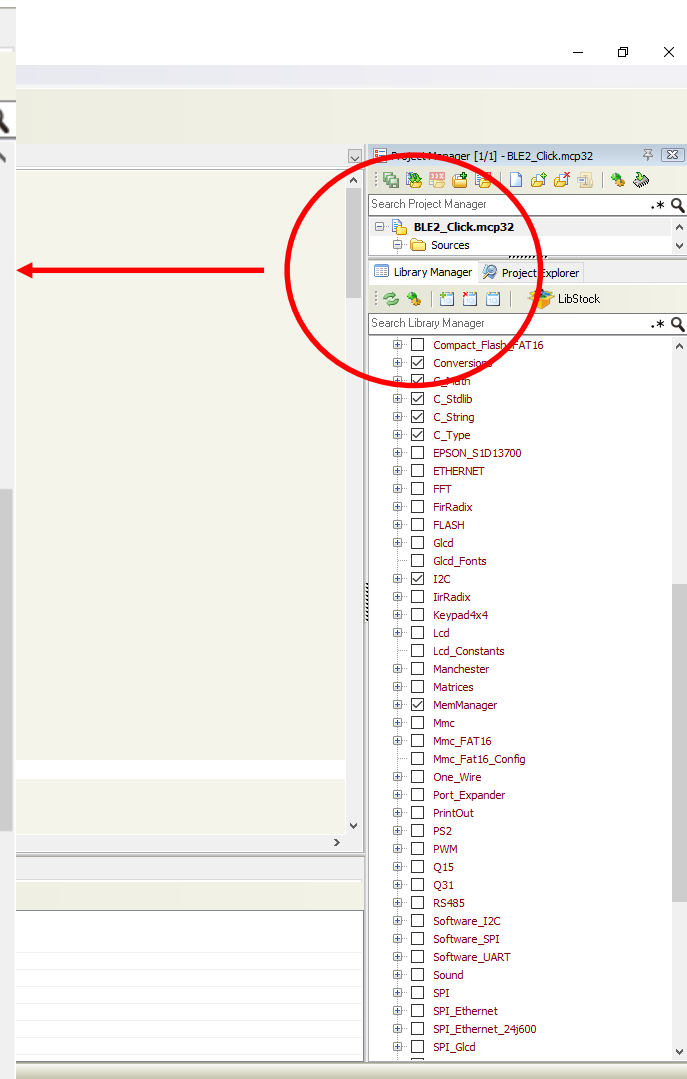
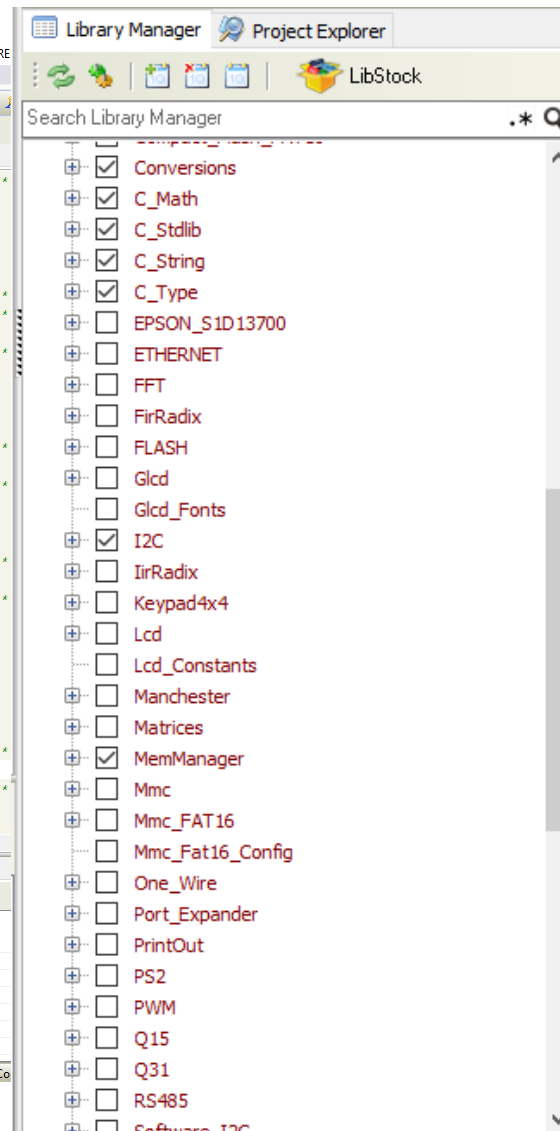
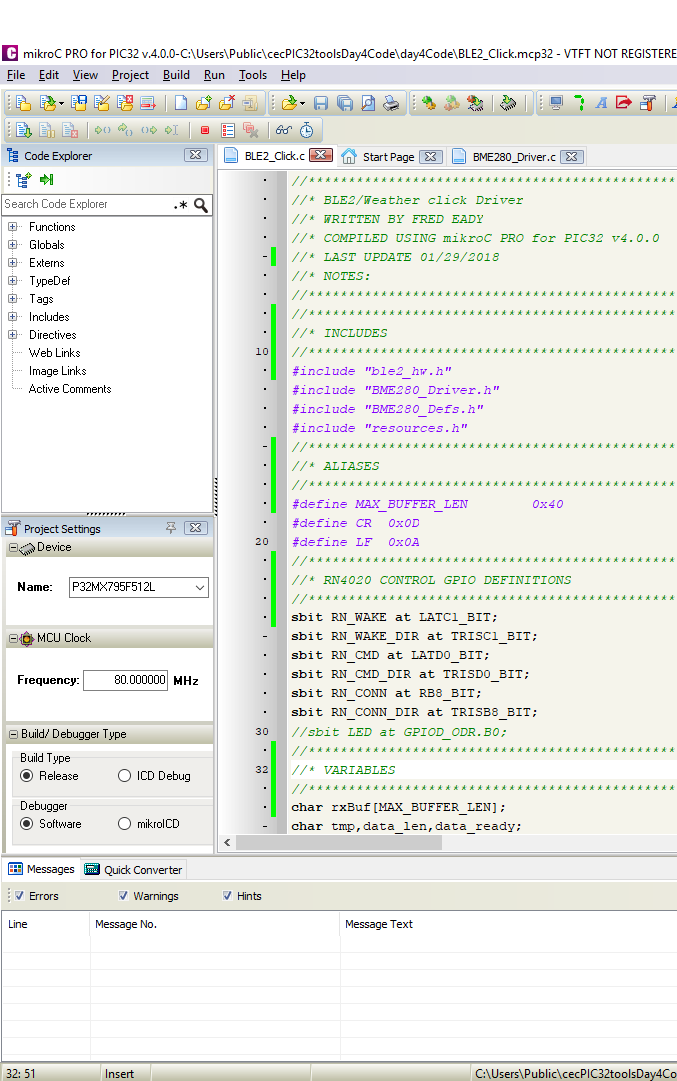
IoT Development Tools for PIC32

The Weather at 6



IoT Development Tools for PIC32

The Weather at 6



IoT Development Tools for PIC32

The Weather at 6

The screenshot shows the mikroC PRO for PIC32 Help window. The title bar reads "mikroC PRO for PIC32 Help". The menu bar includes "Hide", "Back", "Print", "mikroE Support", and "mikroE Forum". The left sidebar has tabs for "Contents", "Index", "Search", and "Favorites". Below the tabs is a search bar with the text "Type in the keyword to find:". The main content area is titled "UART Library" and "mikroC PRO for PIC32 Libraries > Hardware Libraries >". It displays the documentation for the **UARTx_Write** function.

Prototype	<code>void UARTx_Write(unsigned _data);</code>
Description	The function transmits a byte via the <code>UART</code> module.
Parameters	<ul style="list-style-type: none">■ <code>_data</code>: data to be sent.
Returns	Nothing.
Requires	<p>Routine requires at least one <code>UART</code> module.</p> <p>Used <code>UART</code> module must be initialized before using this routine. See UARTx_Init and UARTx_Init_Advanced routines.</p>
Example	<pre>unsigned data = 0x1E; ... UART1_Write(data);</pre>
Notes	<ul style="list-style-type: none">■ <code>UART</code> library routines require you to specify the module you want to use. To select the desired <code>UART</code> module, simply change the letter <code>x</code> in the routine prototype for a number from 1 to 6.■ Number of <code>UART</code> modules per MCU differs from chip to chip. Please, read the appropriate datasheet before utilizing this library.

Below the main content area, the **UARTx_Write_Text** function is partially visible, showing its prototype: `void UARTx_Write_Text(char * UART text);`

The screenshot shows the Library Manager window. The title bar reads "Library Manager" and "Project Explorer". The menu bar includes "LibStock". The search bar contains the text "Search Library Manager". The list of functions is as follows:

- ☒ **UART**
- `UART1_Data_Ready`
- `UART1_Init`
- `UART1_Init_Advanced`
- `UART1_Read`
- `UART1_Read_Text`
- `UART1_Tx_Idle`
- `UART1_Write`
- `UART1_Write_Text`
- `UART2_Data_Ready`
- `UART2_Init`
- `UART2_Init_Advanced`
- `UART2_Read`
- `UART2_Read_Text`
- `UART2_Tx_Idle`
- `UART2_Write`
- `UART2_Write_Text`

IoT Development Tools for PIC32

The Weather at 6

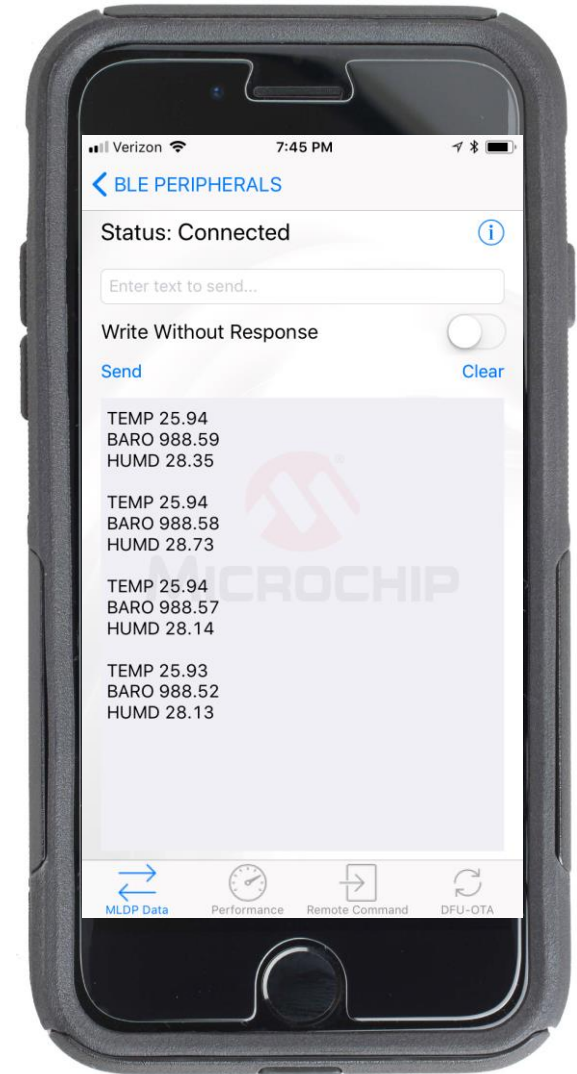
```
140 //*****  
// * INITIALIZE RN4020  
//*****  
void BLE_Init()  
{  
    RN_WAKE = 1;  
    while (!wait_response("CMD"));           //Wait to response CMD  
    ble2_reset_to_factory_default(1);        //Reset RN4020 to factory settings  
    while (!wait_response("AOK"));           //Wait to response AOK  
    ble2_set_device_name("BLE2_Click");      //Set name BLE2_Click  
    while (!wait_response("AOK"));           //Wait to response AOK  
    ble2_set_supported_features(30000800);  //Auto Advertise (Table: 2-5 in Datasheet)  
150 while (!wait_response("AOK"));           //wait to response AOK  
    ble2_device_reboot();                    //reboot  
    while (!wait_response("Reboot"));        //wait to response Reboot  
    ble2_bond(1);  
    ble2_clear_script();  
}
```



IoT Development Tools for PIC32

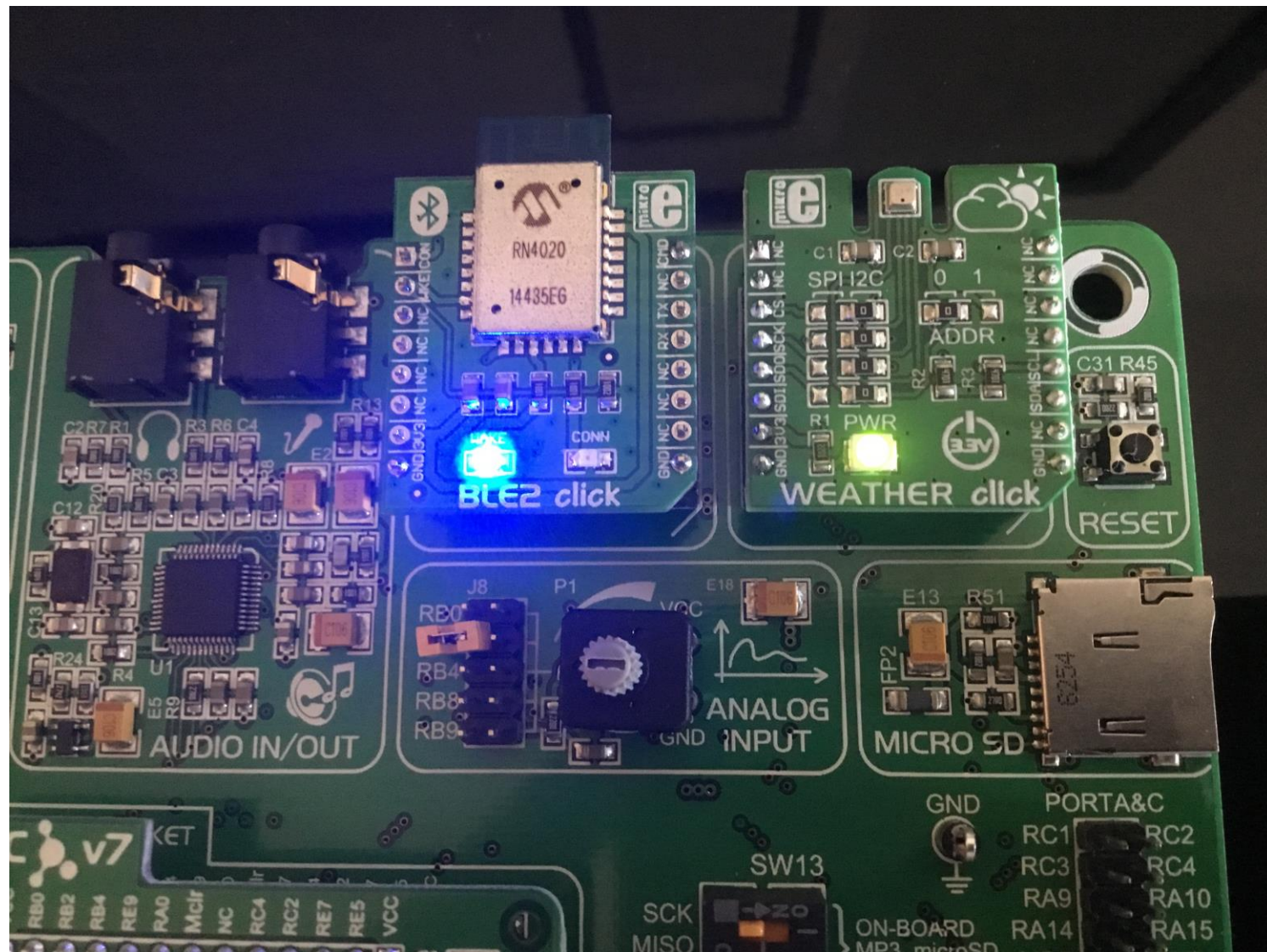
The Weather at 6

```
do{  
    while(BME280_IsMeasuring());  
    BME280_ReadMeasurements();  
    tmp_f = BME280_GetTemperature();  
    sprintf(txBuf, "TEMP %.2f", tmp_f);  
    BLE2_SendBuffer(txBuf, sizeof(txBuf));  
    BLE2_SendByte(CR);  
    BLE2_SendByte(LF);  
180    tmp_f = BME280_GetPressure();  
    sprintf(txBuf, "BARO %.2f", tmp_f);  
    BLE2_SendBuffer(txBuf, sizeof(txBuf));  
    BLE2_SendByte(CR);  
    BLE2_SendByte(LF);  
    tmp_f = BME280_GetHumidity();  
    sprintf(txBuf, "HUMD %.2f", tmp_f);  
    BLE2_SendBuffer(txBuf, sizeof(txBuf));  
    BLE2_SendByte(CR);  
    BLE2_SendByte(LF);  
190    BLE2_SendByte(CR);  
    BLE2_SendByte(LF);  
    BME280_INIT();  
    delay_ms(1500);  
}while(1);
```



IoT Development Tools for PIC32

Buenas Noches



Presented by: