AN3417

Indoor Air Quality Monitor: Firmware Creation Using Atmel START and MPLAB® Code Configurator (MCC)

Introduction

Authors: Babashree Ingale, Ankit Phougat, Microchip Technology Inc.

The indoor Air Quality Monitor (AQM) is implemented using the AVR-IoT WG development board from Microchip, and sensors and several click boards from MikroElektronika. The AVR-IoT WG development board is equipped with Microchip's ATmega4808 microcontroller (AVR® MCU), ATECC608A CryptoAuthentication™ secure element, and ATWINC1510 Wi-Fi® module.

The AQM monitors the humidity and temperature along with main airborne contaminants such as particulate matter ($PM_{2.5}$), Carbon dioxide (CO_2) and total volatile organic components (TVOC). The microcontroller processes these acquired readings and calculates the Air Quality Index (AQI) from the readings of the $PM_{2.5}$ sensor. The AQI and other acquired air quality parameters are stored on external EEPROM, displayed on the OLED, and uploaded to Google Cloud.

AQM Documentation Overview

The following documents cover the AQM system documentation.

1. This application note covers the usage of AVR-IoT stack (AVR-IoT WG source code) for the AQM application. Additionally, it covers configuration details of microcontroller peripherals and click boards, using Atmel START and MCC. The indoor AQM application is created on top of the AVR-IoT WG source code.
2. The *"AN3403 - Indoor Air Quality Monitor: Concept and Implementation"* application note describes the hardware and firmware overview of the AQM system and its power considerations.
3. The *"Indoor Air Quality Monitor: User Guide"* covers hardware setup, hardware connections, operating steps, LED indications, provisioning of the AVR-IoT WG board's Wi-Fi module, and visualization of AQM Data over the cloud.

It is recommended to read the *"AN3403 - Indoor Air Quality Monitor: Concept and Implementation"* application note before reading this firmware creation guide.

Table of Contents

1. Indoor Air Quality Monitor Firmware Creation Using Atmel START

This section covers the list of steps to generate AVR-IoT WG source code with Atmel START, and the process to create indoor AQM application firmware.

Table 1-1 lists the software tools and versions used for the creation of the firmware using Atmel START. It is recommended to use tools of the mentioned or higher versions.

Table 1-1. Software Tools and Versions

Software Tool	Version
Atmel Studio 7	v7.0.2397
Atmel START	v1.7.279
ATmega_DFP	v1.4.331
AVR GCC	v5.4.0
avr8-gnu	v3.6.2.1778

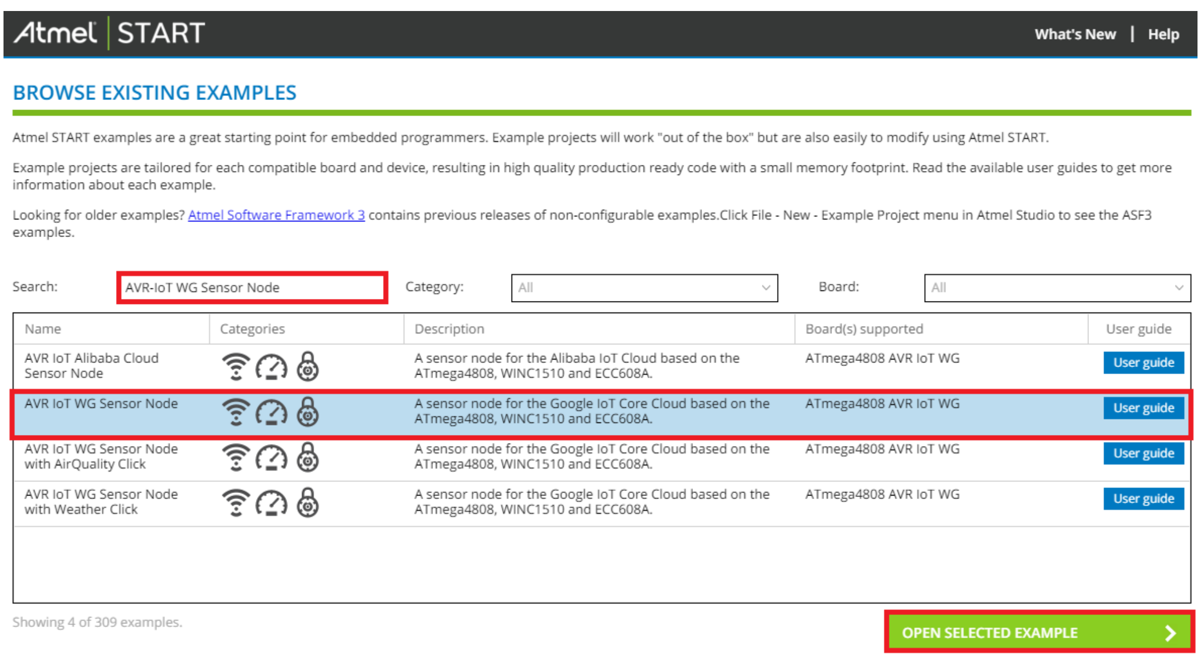
1.1 Generating AVR-IoT WG Sensor Node Example Project

- Open Atmel Studio. Go to *File → New → Atmel START Example Project*.
- Wait until the *BROWSE EXISTING EXAMPLE* page loads.

Tip: Maximize the page for better experience.

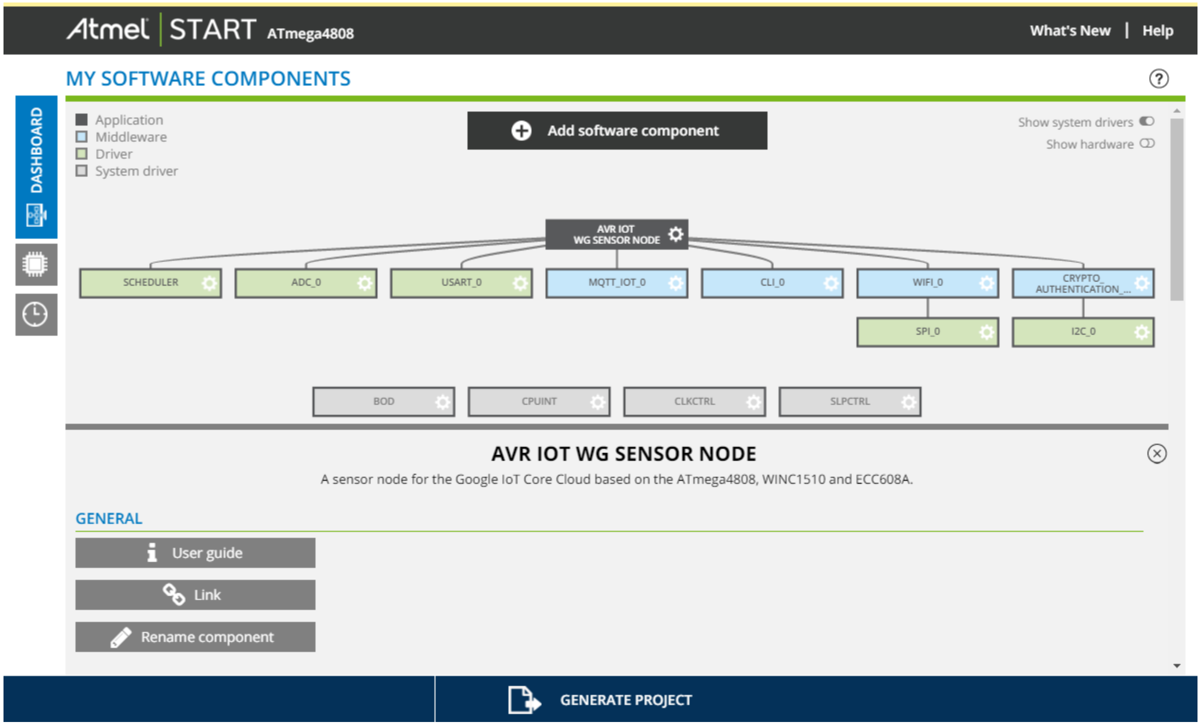
- Search and select **AVR-IoT WG Sensor Node** example. Then click **OPEN SELECTED EXAMPLE.**

Figure 1-1. AVR-IoT WG Sensor Node Project



- The *Dashboard* view of the microcontroller peripherals and its driver's configuration window opens.

Figure 1-2. Dashboard View



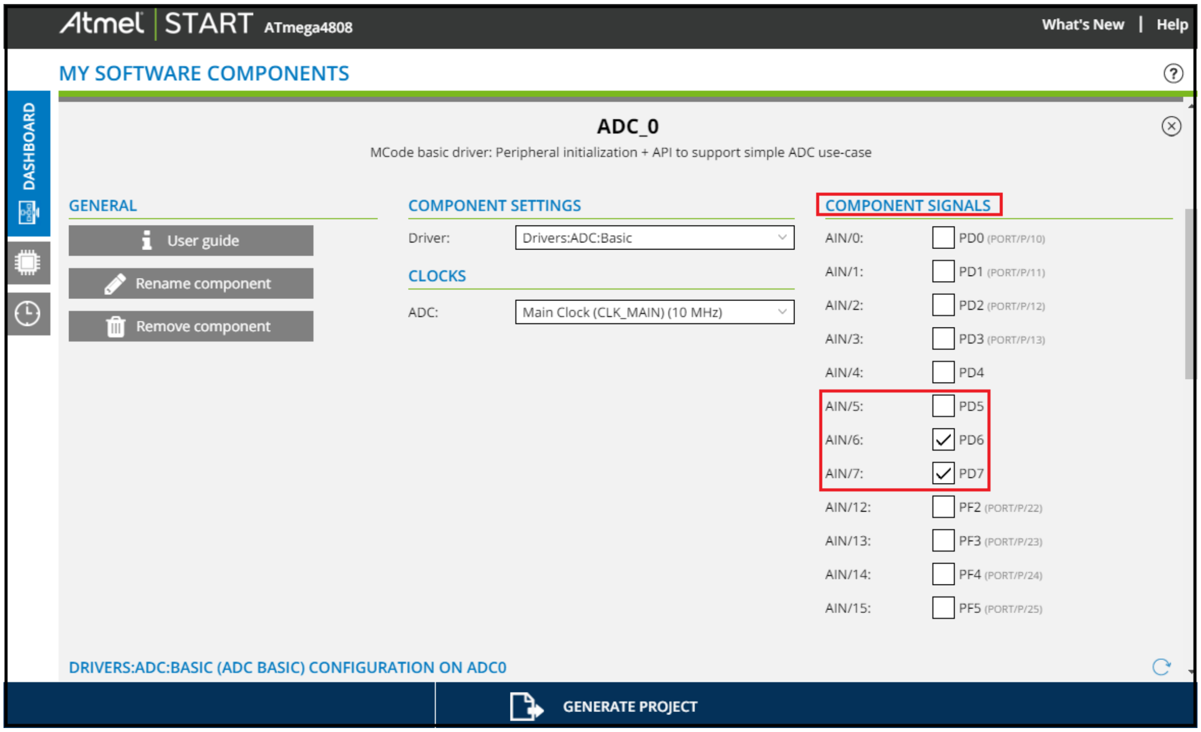
1.2 Configuring Additional Peripherals

The subsections below explain the configuration of additional modules/peripherals required for the indoor Air Quality Monitor application.

1.2.1 ADC

- The SHT31 analog sensor is used for humidity and temperature measurement. The ADC peripheral is configured to read the sensor output. The **ADC** peripheral driver is added to the project as part of the AVR-IoT WG Sensor Node example project. So, there is no need to add the **ADC** peripheral driver once more to the project.
- To configure ADC, click **ADC_0.** The ADC configuration window appears.
- The temperature and humidity output pins of the SHT31 sensor are connected to the port pins PD6 and PD7, respectively. To configure these port pins as ADC channels, go to the *COMPONENT SIGNALS* section and select pins **PD6** and **PD7** as ADC pins. Deselect pin **PD5**, because the light sensor connected to PD5 on the AVR-IoT WG board is not used in the indoor AQM application.

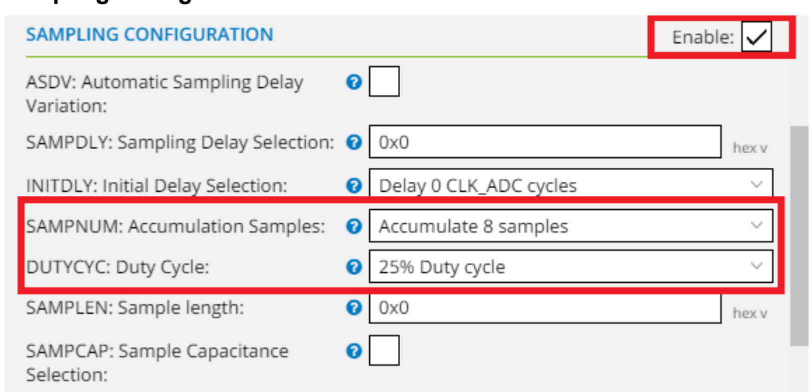
Figure 1-3. ADC PIN Configuration



Note: The AQM hardware has two temperature sensors: One is the analog sensor (SHT31-ARP), and another one is the on-board (AVR-IoT WG board) digital sensor MCP9808. The AQM uses the MCP9808 digital sensor for monitoring temperature.

- The Sample Accumulation feature of the ADC is used for the indoor AQM application. In the ADC configuration window, enable *SAMPLING CONFIGURATION*. Configure *SAMPNUM: Accumulation Samples* to *Accumulate 8 samples*, and *DUYYCUC: Duty Cycle* to *25% Duty cycle*, in the *SAMPLING CONFIGURATION* section.

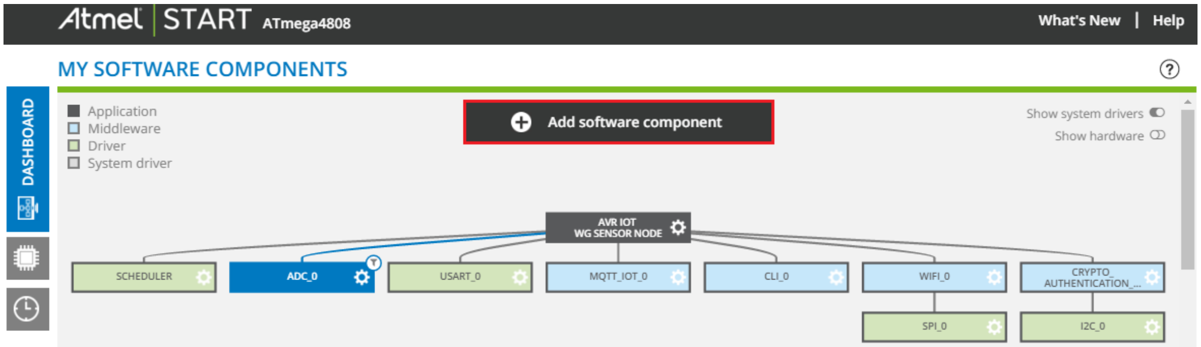
Figure 1-4. ADC Sampling Configuration



1.2.2 UART

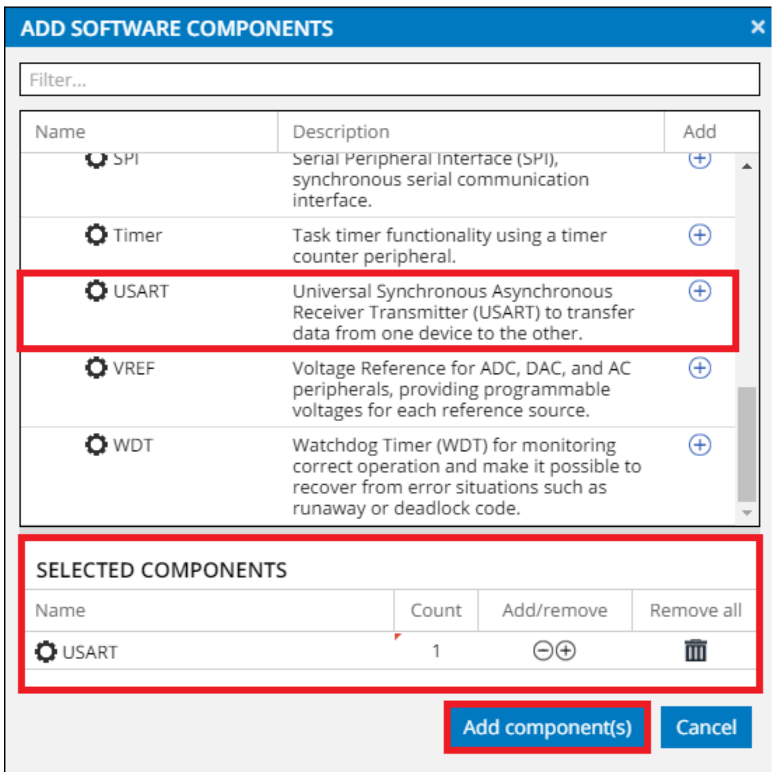
- The SPS30 $PM_{2.5}$ sensor communicates with the microcontroller through the UART interface.
- To configure UART1, add the **USART** peripheral driver by clicking **Add Software Component**.

Figure 1-5. Add Software Component



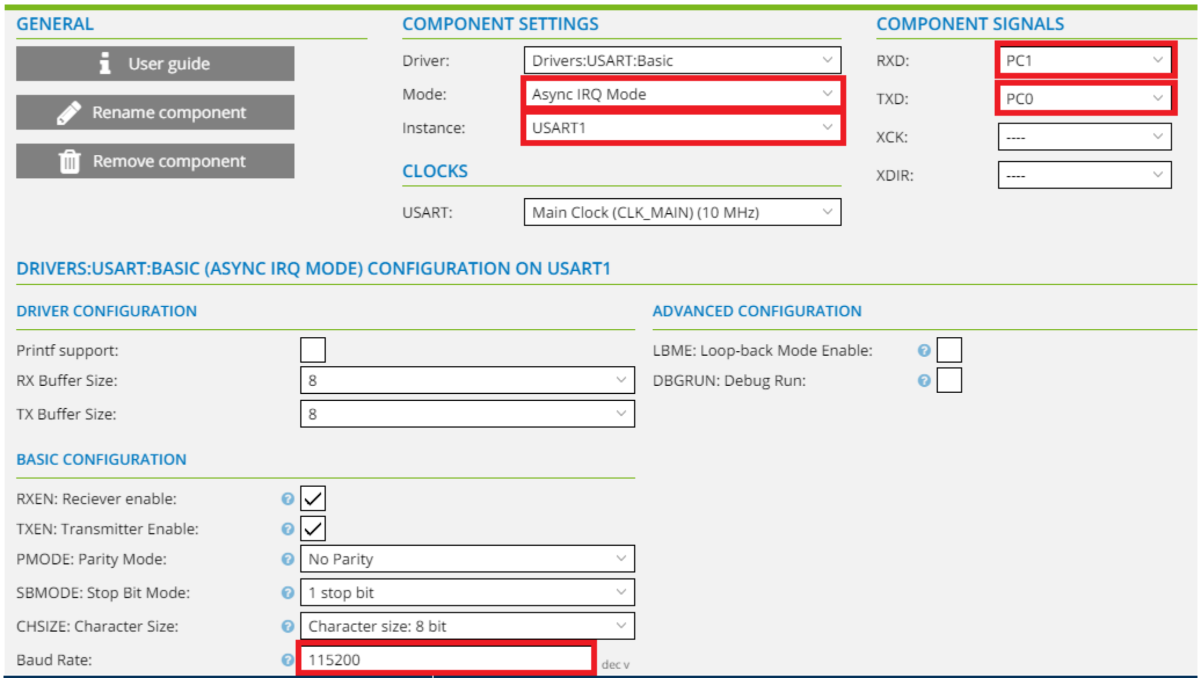
- Expand the **Drivers** tab and add the **USART** peripheral driver. The USART appears on the *SELECTED COMPONENTS* box. Click **Add component(s)**.

Figure 1-6. USART Module Driver Addition



- In the *Dashboard* view, click **USART_1** to configure the UART peripheral. Configure *Mode* as *Async IRQ Mode*, and *Instance* as *USART1* in the *COMPONENT SETTINGS* section. Configure *RXD* to *PC1*, and *TXD* to *PC0* in the *COMPONENT SIGNALS* section. Set *Baud Rate* to *115200* in the *BASIC CONFIGURATION* section.

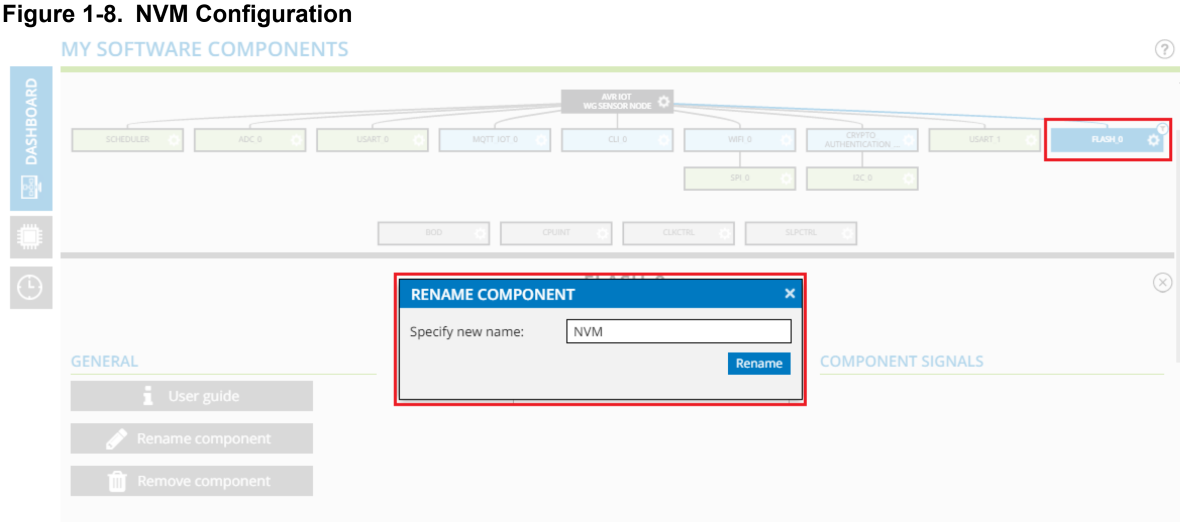
Figure 1-7. USART Module Configuration



1.2.3 NVM

- The internal EEPROM is used to store the default parameters and the external EEPROM's send and write indices. The internal EEPROM is accessed using the NVM control peripheral.
- To configure NVM control, add the **Flash** peripheral driver by following the steps described in the USART section.
- Select **FLASH_0** in the *Dashboard* view.
- Rename component as *NVM*.

Figure 1-8. NVM Configuration

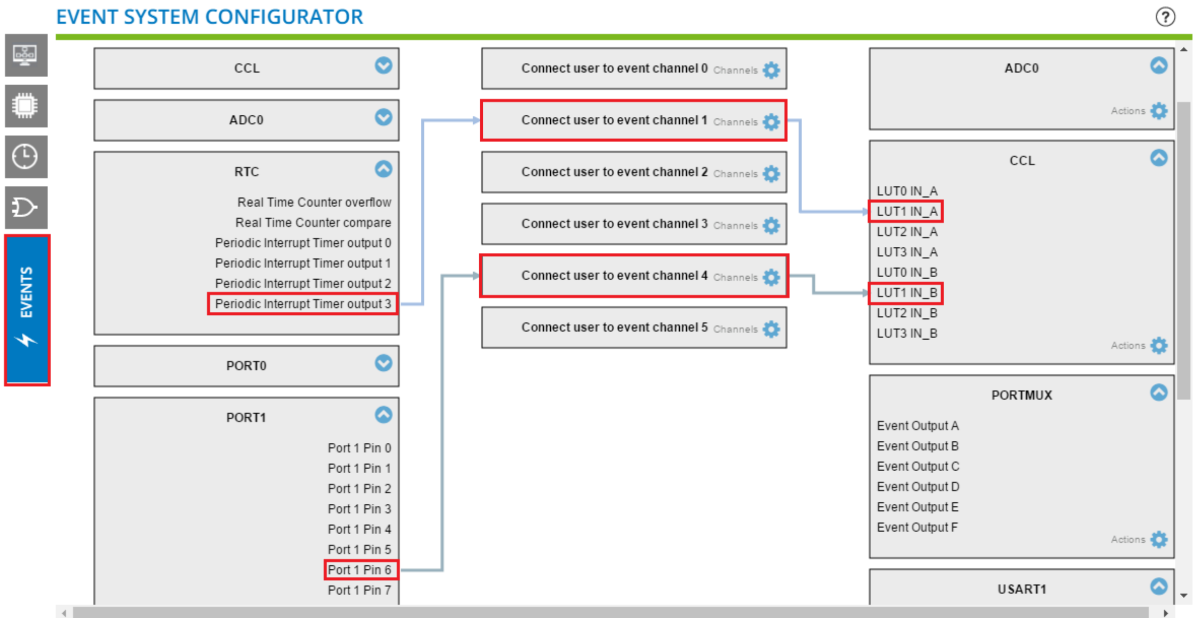


1.2.4 Event System

- The on-board switch on the AVR-IoT WG board is used to turn on the OLED display and to change the parameter to be displayed on the OLED. The on-board switch SW0 on the AVR-IoT WG board is connected to PORT1_PIN6 of the microcontroller. The switch debounce mechanism is implemented to detect switch press event without any software intervention, using Event System and CCL peripherals of the microcontroller.

- Add both **Event System** and **Digital Glue Logic (CCL)** drivers by following the steps as described in the *USART* section.
- Click the **EVENT** symbol to configure the Event System peripheral, as shown in Figure 1-9. The *EVENT SYSTEM CONFIGURATOR* window opens.
- Connect *Periodic Interrupt Timer output 3* to *Connect user to event channel 1*, and *Port 1 Pin 6* to *Connect user to event channel 4*. Click, hold, and drag to the channel for connection. Thus, PIT3 and Port1 Pin6 act as event generators for channel 1 and channel 4, respectively.

Figure 1-9. Event System Configuration

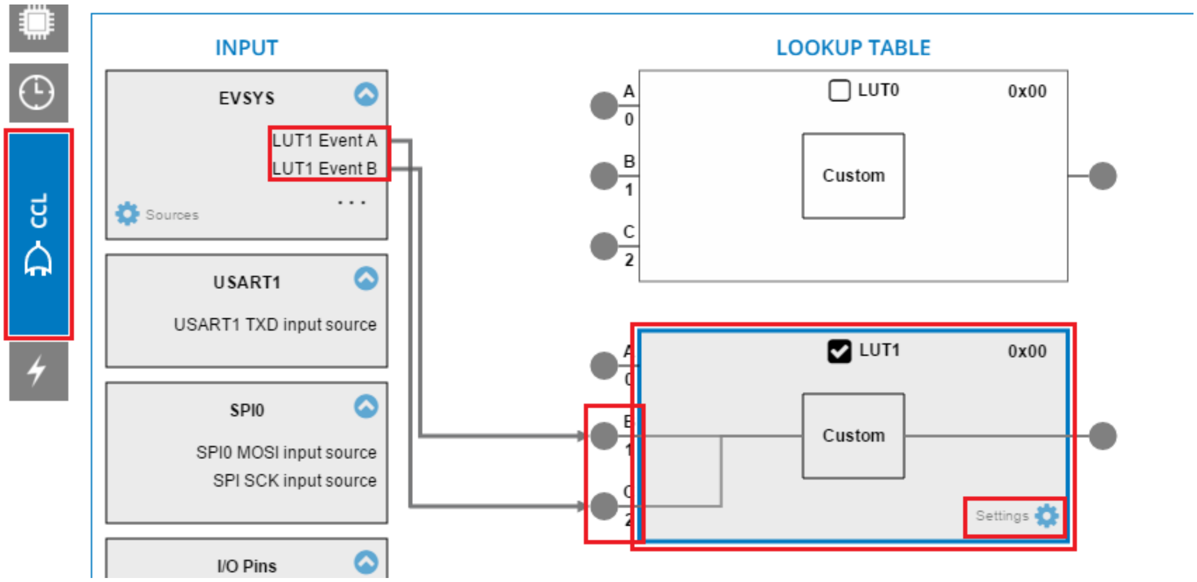


- In the same way, connect *Connect user to event channel 1* to *LUT1IN_A* and *Connect user to event channel 1* to *LUT1IN_B*, as shown in Figure 1-9. Thus, LUT1IN_A and LUT1IN_B of the CCL act as event user for channel 1 and channel 4, respectively.

1.2.5 CCL

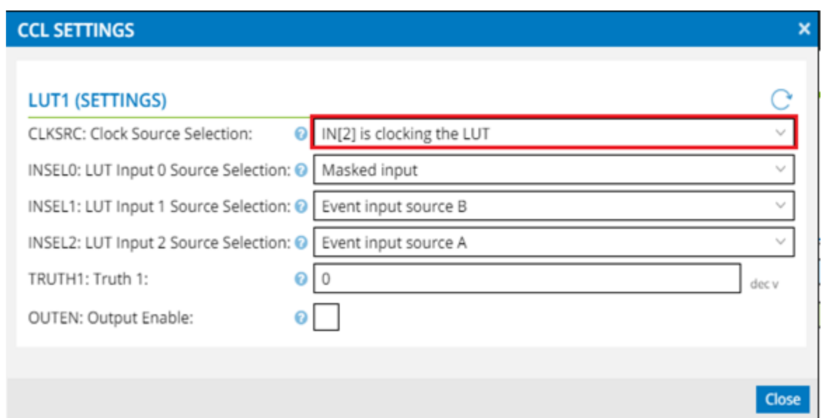
- To implement the switch debounce mechanism without any software intervention, Look-Up Table 1 (LUT1) of the CCL peripheral with filter option is used. The CCL peripheral generates an interrupt for each switch press event.
- Click the **CCL** symbol, as shown in Figure 1-10. The *CCL CONFIGURATOR* window opens.
- Click **LUT1** to select **LUT1** for configuration. Enable the LUT1 by checking the **LUT1 checkbox**.

Figure 1-10. LUT1 Inputs Configuration



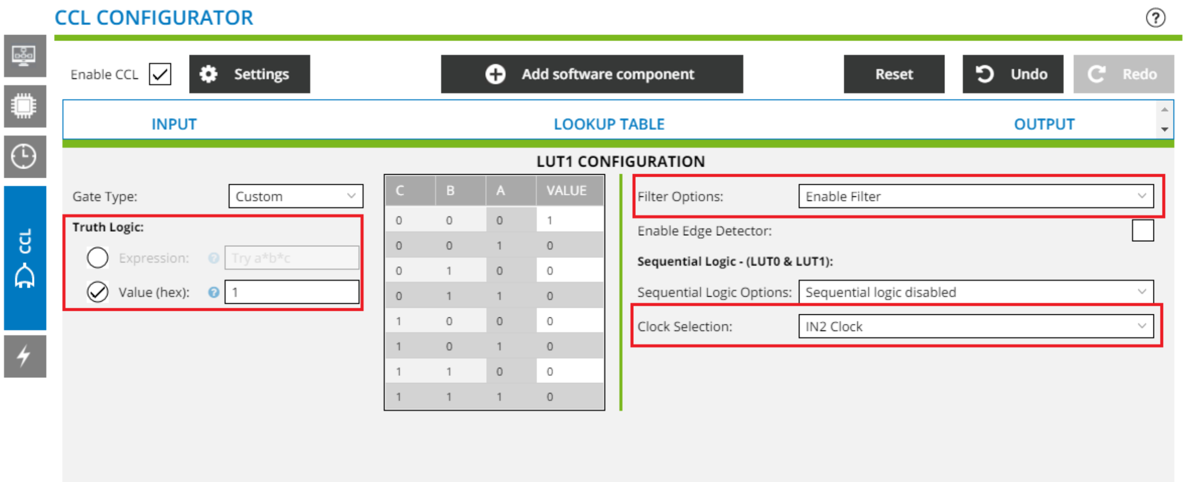
- Connect *LUT1 Event A* to the *C* input of the LUT1 and *LUT1 Event B* to the *B* input of the LUT1, as shown in Figure 1-10.
- Click the **Settings** icon inside the LUT1, as shown in Figure 1-10.
- Configure *CLKSRC: Clock Source Selection* to *IN[2] is clocking the LUT* and close the settings window.

Figure 1-11. LUT1 Clock Source Configuration



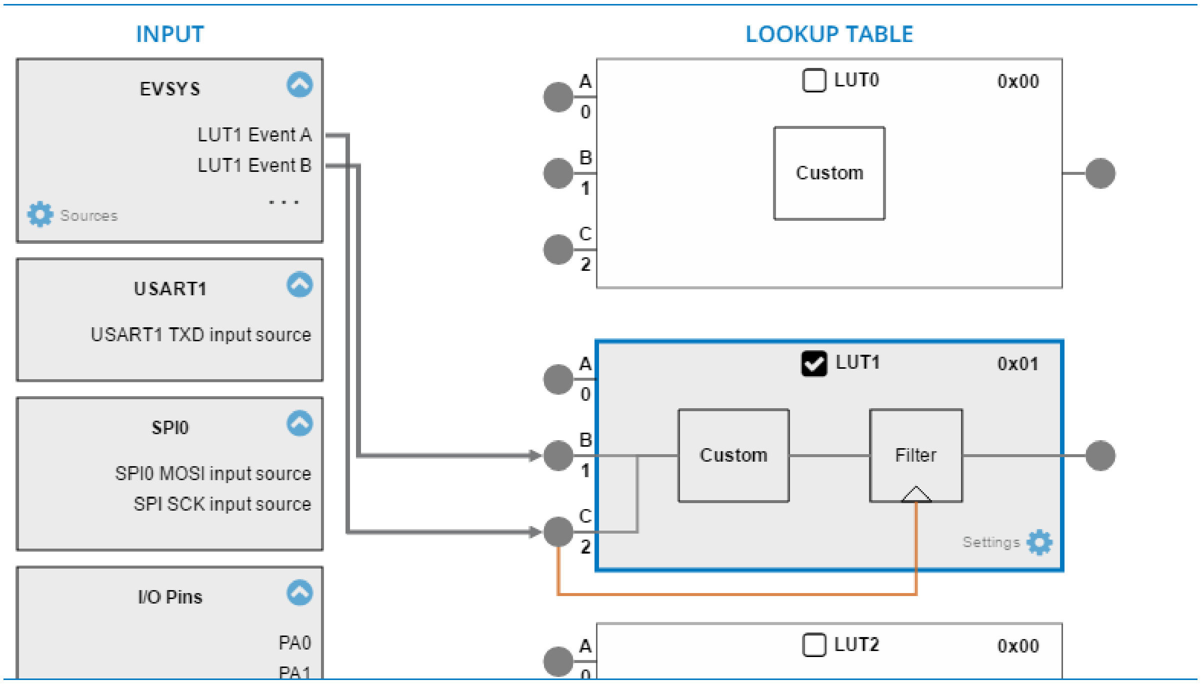
- Go to the *LUT1 CONFIGURATION* section. In *Truth Logic*, set *Value(hex)* to *1*. Configure *Filter Options* to *Enable Filter* and configure *Clock Selection* to *IN2 Clock*.

Figure 1-12. LUT1 Configuration



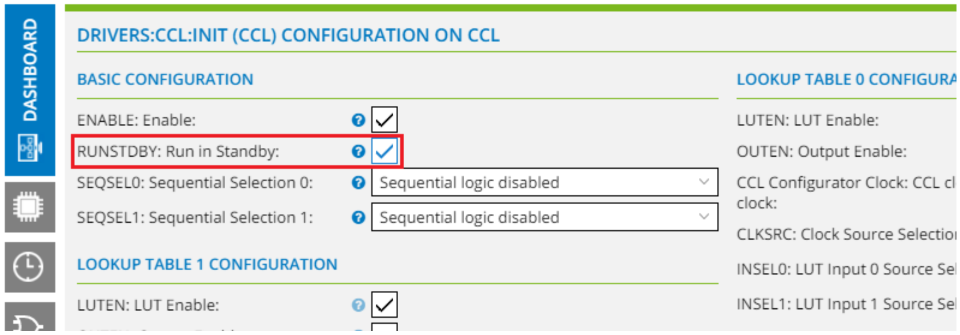
- Finally, LUT1 looks as shown in Figure 1-13.

Figure 1-13. LUT1 after Configuration



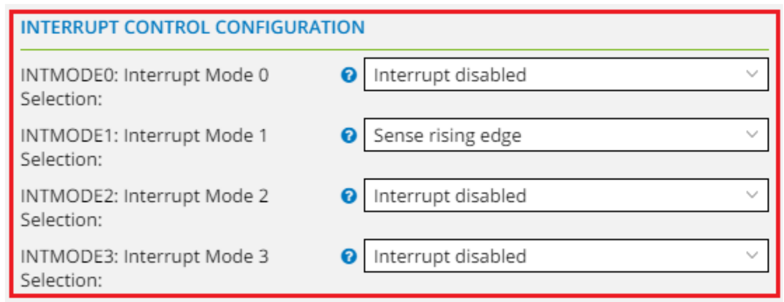
- The microcontroller goes to Standby sleep mode periodically. To detect switch press event while the microcontroller is in Standby sleep mode, the CCL has to run in Standby sleep mode.
- Go to *Dashboard* view and select **Digital Glue Logic**.
- Check the **RUNSTDBY: Run in Standby** option in the *BASIC CONFIGURATION* section.

Figure 1-14. Run in Standby Bit Configuration



- In the *INTERRUPT CONTROL CONFIGURATION* section, configure *INTMODE1: Interrupt Mode 1 Selection* to *Sense rising edge*, and all other Modes Selection to *Interrupt disabled*.

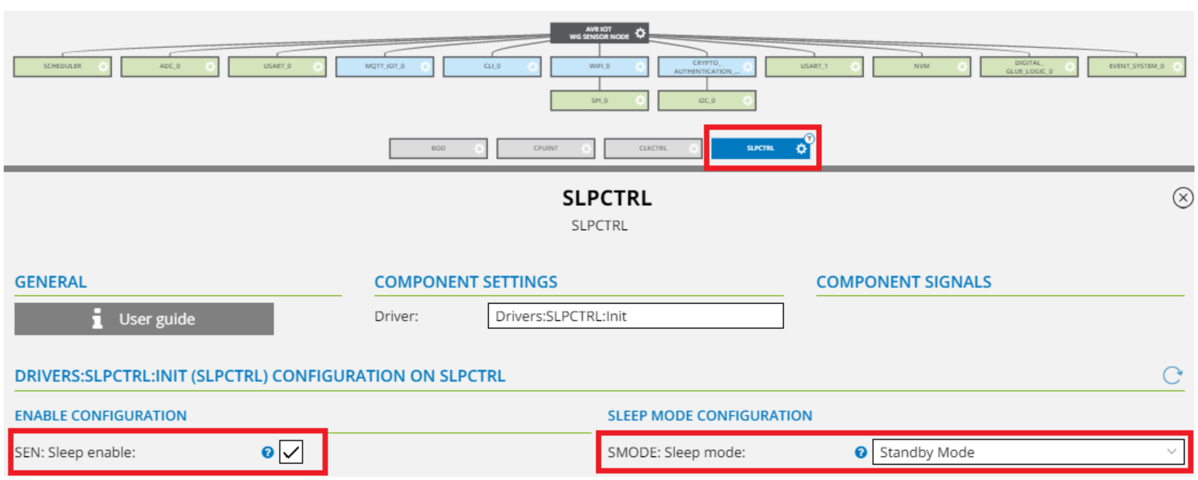
Figure 1-15. Interrupt Control Configuration



1.2.6 Sleep Mode

- The AQM application uses Standby sleep mode. The microcontroller goes to Sleep mode when there is no task to be performed.
- To configure the Sleep mode, click **SLPCTRL** in the *Dashboard* view. Enable *SEN: Sleep enable* and configure *SMODE: Sleep mode* to *Standby Mode*.

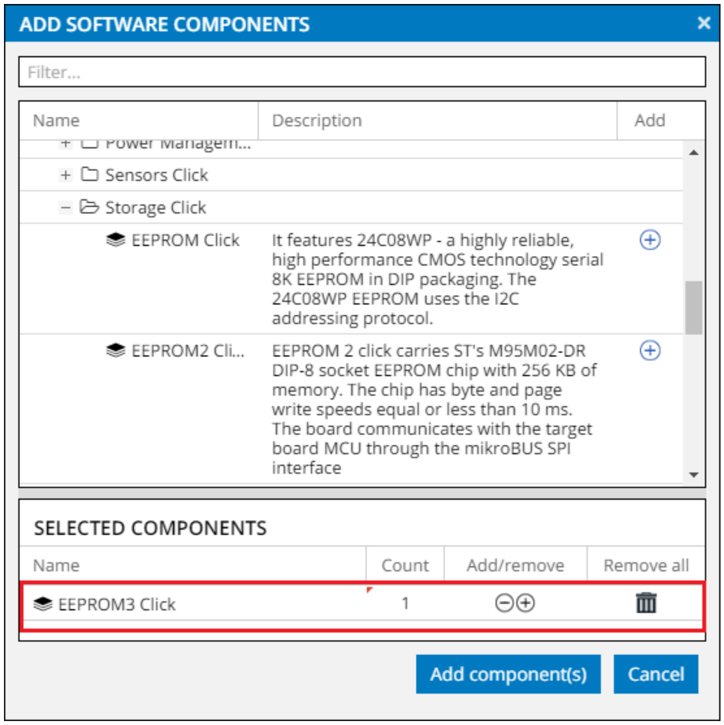
Figure 1-16. Sleep Mode Configuration



1.2.7 EEPROM 3 Click

Air quality sensor readings are stored on external EEPROM, present on the EEPROM 3 click board. Include EEPROM 3 by clicking **Add Software Component**. Expand *Middleware*, then *Storage Click*, and add **EEPROM3 Click**.

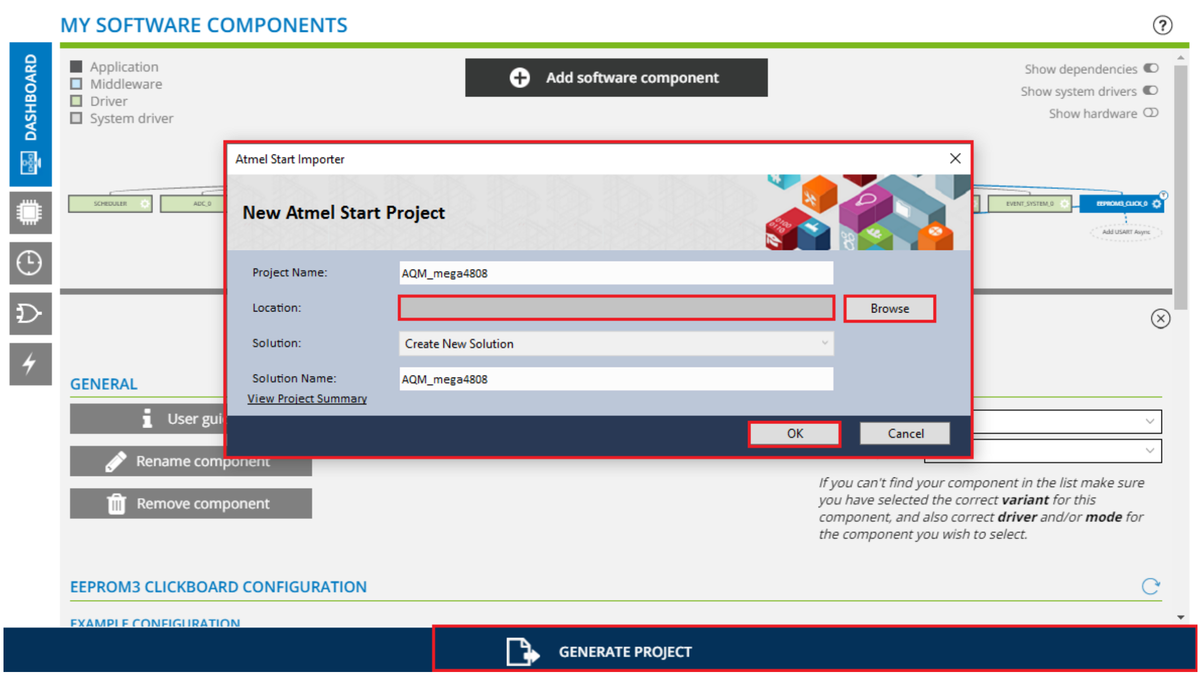
Figure 1-17. EEPROM 3 Click Addition



1.3 Generating START Code Files

- To generate the code, click **GENERATE PROJECT** present at the bottom. The *New Atmel Start Project* window appears for setting up the project parameters. Enter the project name and location and then click **OK**, as shown in Figure 1-18.

Figure 1-18. Project Generation



- Close the start window as the configuration and code generation are completed.

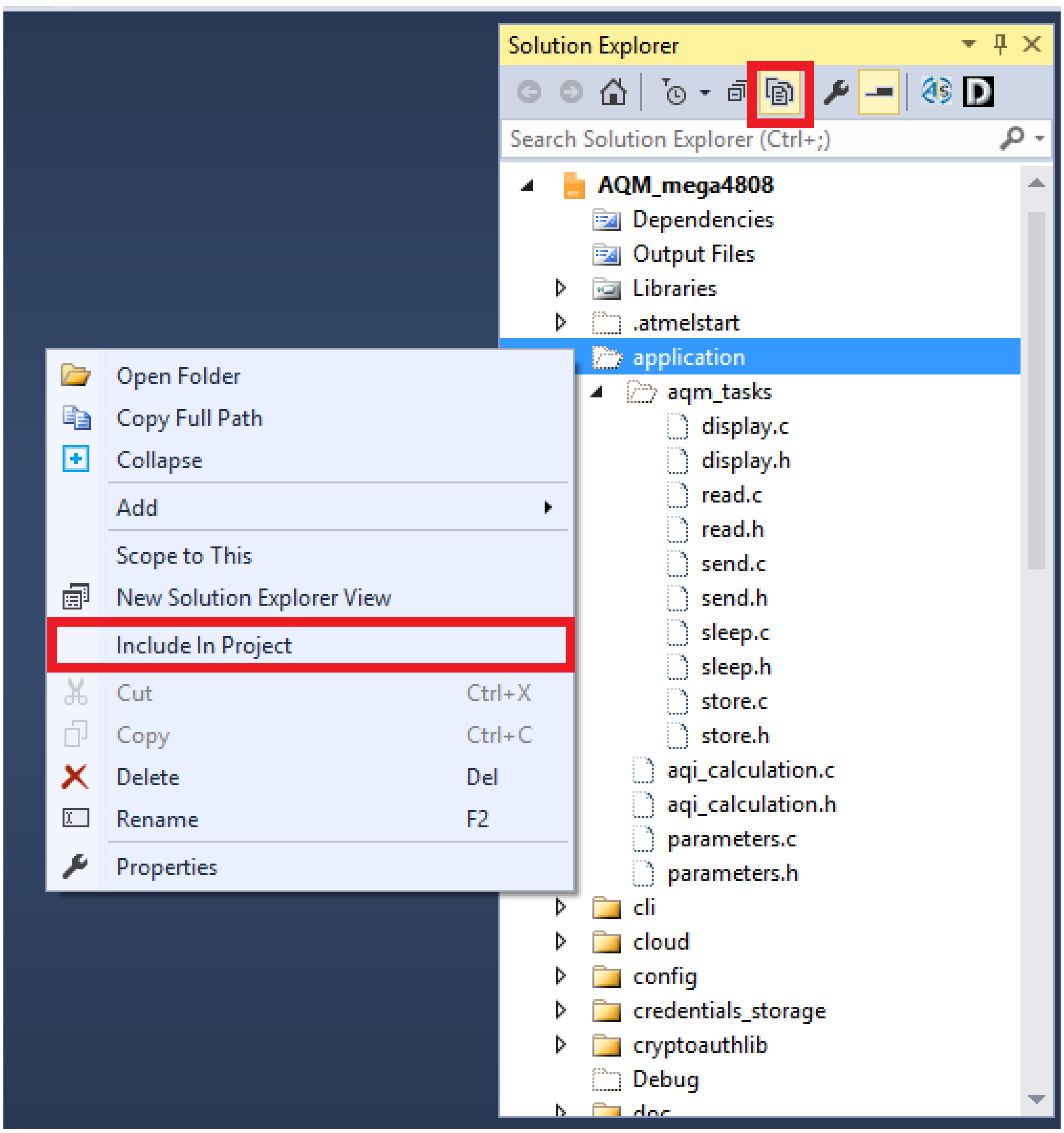
1.4 Adding Application Code and Sensor Drivers

- The firmware is available for download on GitHub, as below:

View Code Example on GitHub (Atmel Studio)
Click to browse repository

- Copy the **application** and **sensor** folders from the downloaded firmware and place them inside the project folder.
- To add both folders to the project, go to *Solution Explorer* in Atmel Studio and click the **Show All Files** symbol, as shown in Figure 1-19. Now, the **application** and **sensor** folders added in the previous step can be seen. Right click the **application** folder and select **Include in project**, as shown in Figure 1-19.
- Repeat these steps for adding the **sensor** folder.

Figure 1-19. Adding Application Files



1.5 Modifying Atmel START Generated Code

The Atmel START generated AVR-IoT WG source code and some of the peripherals' APIs need modifications. The user needs to compare both the AVR-IoT WG source code generated by Atmel START and the firmware available for download on GitHub, as below, to find more details about the required changes. Table 1-2 lists files and functions along with change description, where changes are required.

View Code Example on GitHub (Atmel Studio)
Click to browse repository

Table 1-2. List of Changes to START Generated Code

Source File	Function	Change Description
`main.c`	`SendToCloud()`	A dedicated task is created to send data to the cloud. Thus, this function is no longer required.
	`main()`	Parameter initialization and Cloud configuration.
`application_manager.c`	`application_init()`	PIT initialization.
		Turn on the display.
		Created application tasks.
	`MAIN_dataTask()`	No call to `SendtoCloud()` function.
`cryptoauthlib\lib\jwt\atca_jwt.c`	`atca_jwt_finalize()`	Switched to use the hardware SHA function during the JWT creation (to save memory).

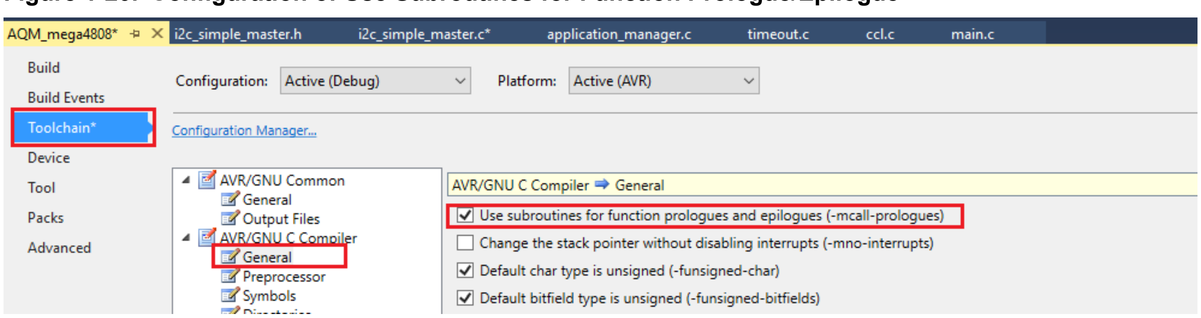
1.6 Compiler Settings

This section explains about the compiler settings which are required for code optimization.

1.6.1 Activate the "Use subroutines for function prologue/epilogue"

- Right click on the project name and select **Properties**.
- In the **Toolchain** tab, click **General** under *AVR/GNU C Compiler*. Enable *Use subroutines for function prologue/epilogue* option, as shown in Figure 1-20.

Figure 1-20. Configuration of Use Subroutines for Function Prologue/Epilogue

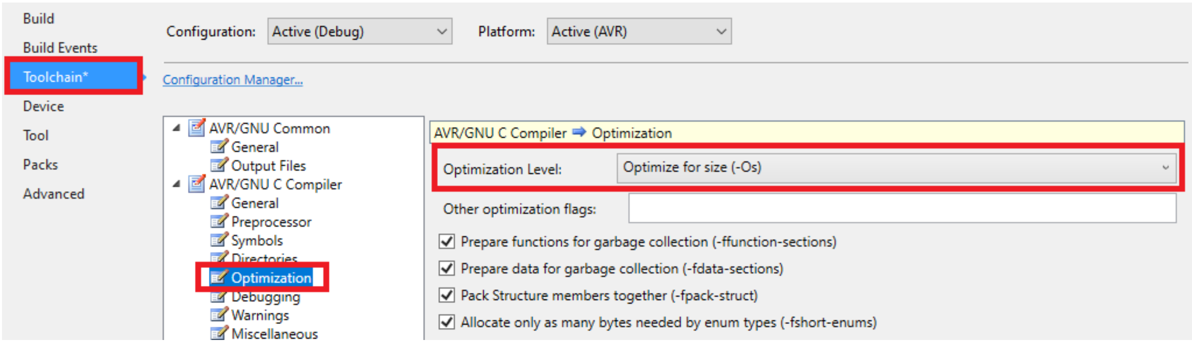


Note: This is required to optimize the code size.

1.6.2 Compiler Optimization Setting

- Right click on the project name and select **Properties**.
- In the **Toolchain** tab, click **Optimization** under *AVR/GNU C Compiler*. Choose *Optimize for size (-Os)* from the drop-down menu of the *Optimization Level*.

Figure 1-21. Compiler Optimization Setting



2. Indoor Air Quality Monitor Firmware Creation Using MCC

This section covers the list of steps to generate AVR-IoT WG source code with MCC, and the process to create indoor AQM application firmware.

Table 2-1 lists the software tools and versions used for the creation of the firmware using MCC. It is recommended to use tools of mentioned or higher versions.

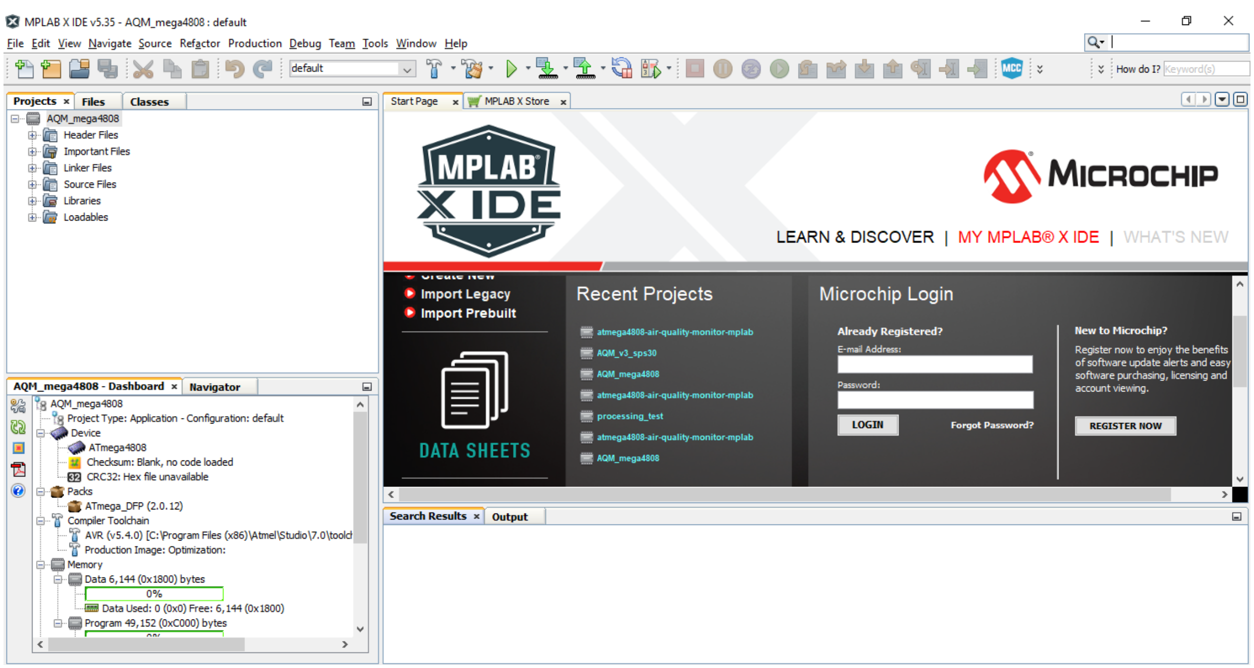
Table 2-1. Software Tools and Versions

Software Tool	Version
MPLAB® X	v5.35
AVR GCC	v5.4.0
avr8-gnu	v3.6.2.1778
MCC	v3.95.0
8-bit AVR MCUs Library	v2.1.0
AVR-IoT WG Sensor Node	v1.2.0
Foundation services	v0.1.34
MikroElektronika Click Library	v1.1.1
Atmega_DFP	v2.1.87

2.1 Creating a New MPLAB® X Project

Go to *File → New Project* and select **Standalone project**. Select **ATmega4808** as the device, and the **AVR** (GCC Compiler) as the compiler. Enter the project name and location details and create the project. The Start page of a new MPLAB X project appears, as shown in Figure 2-1.

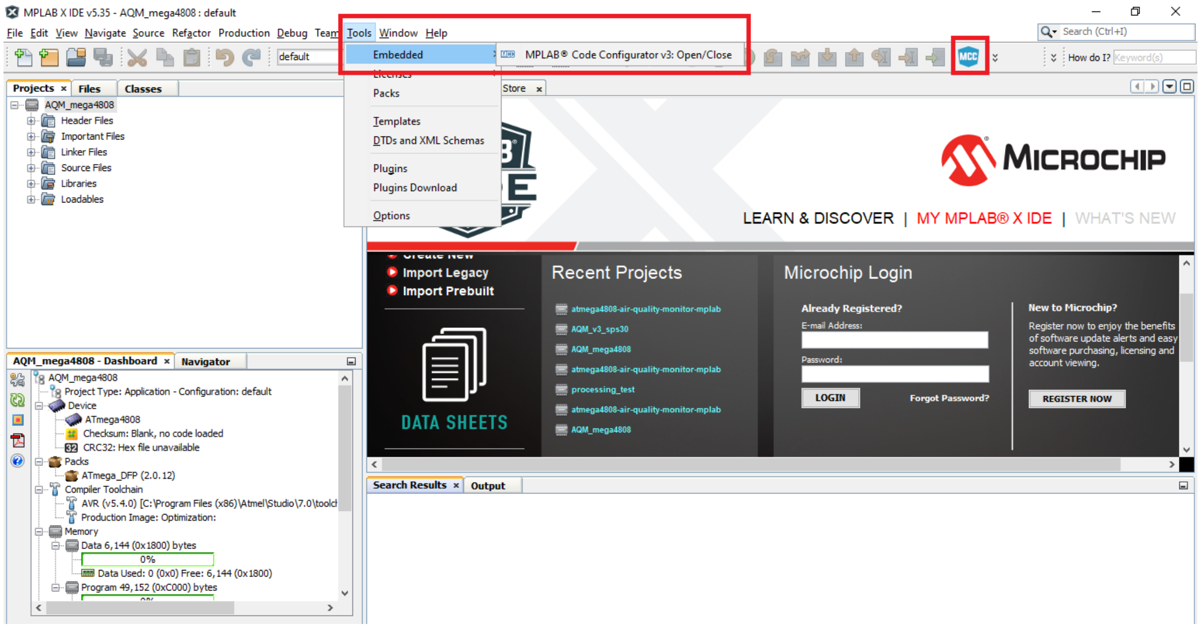
Figure 2-1. Start Page of a New Project Created with MPLAB® X



2.2 Generating AVR-IoT WG Sensor Node Example Project

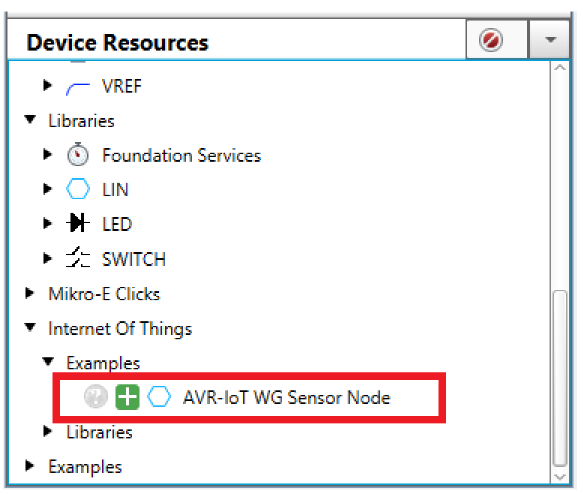
- Open MCC by clicking the **MCC** icon with MPLAB X, or click on *Tools → Embedded → MPLAB X Code Configurator v3 Open/Close*.

Figure 2-2. Open MCC



- Wait for the MCC configuration window to open. Once the configuration window appears for the new project, click **Save**.
- In the *Device Resources* section, expand the **Internet of Things** tab. Add the **AVR-IoT WG Sensor Node** example project.

Figure 2-3. AVR-IoT WG Sensor Node



- With the above steps, *AVR-IoT WG Sensor Node* is added to the project.

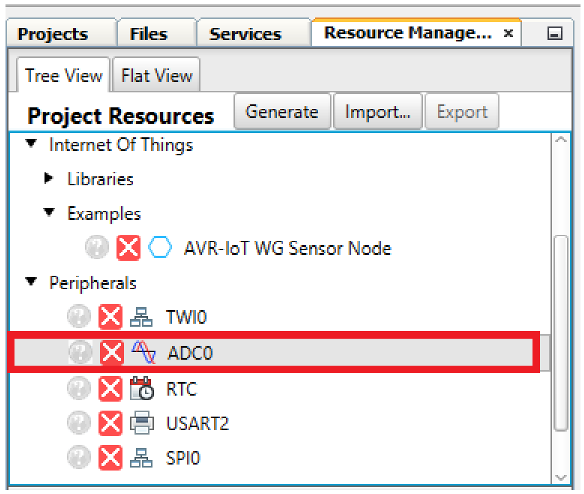
2.3 Configuring Additional Peripherals

The subsections below explain the configuration of other MCC modules/peripherals required for the indoor Air Quality Monitor application.

2.3.1 ADC0

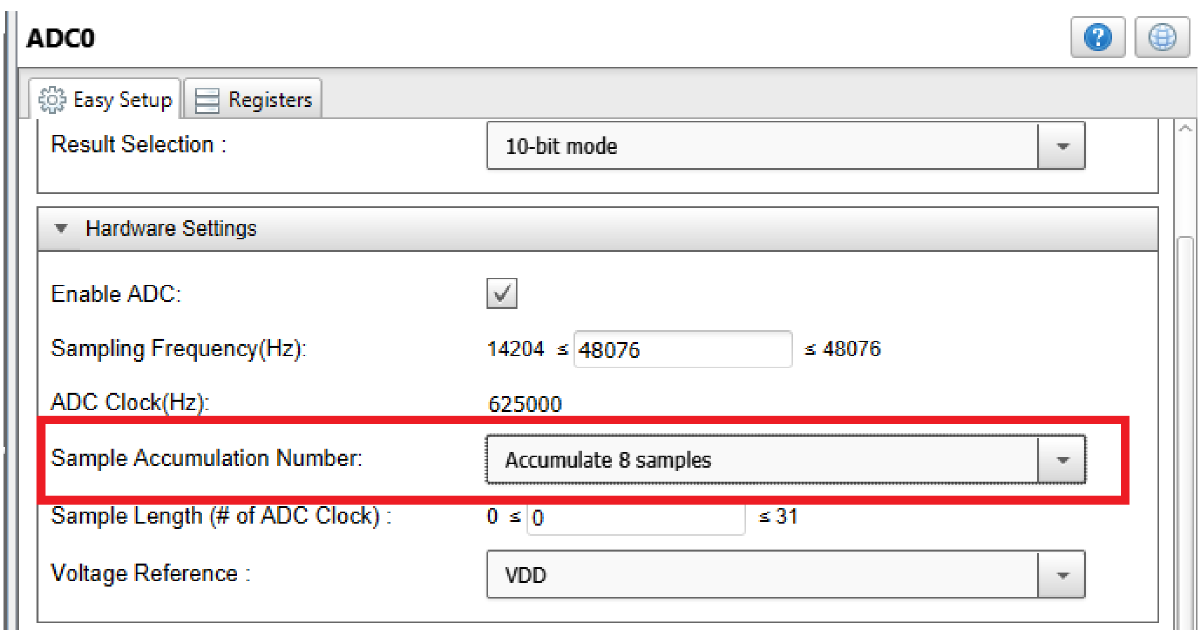
- The SHT31 analog sensor is used for humidity and temperature measurement. The ADC0 peripheral is configured to read the sensor output. The **ADC0** peripheral driver is added to the project as part of the AVR-IoT WG Sensor Node example project. There is no need to add the **ADC0** peripheral driver once more to the project.
- To configure ADC, click the **ADC0** peripheral under the **Project Resources** tab.

Figure 2-4. ADC Peripheral Selection



- The ADC0 configuration window opens.
- The Sample Accumulation feature of the ADC is used in the indoor AQM application. In the *Hardware Settings* section, configure *Sample Accumulation Number* to *Accumulate 8 samples*.

Figure 2-5. ADC0 Configuration



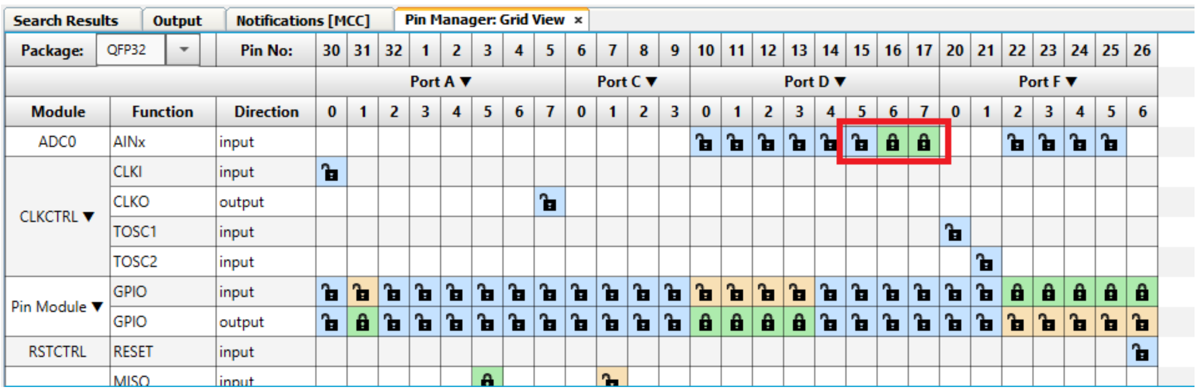
Note: The AQM hardware has two temperature sensors: One is the analog sensor (SHT31-ARP), and another one is the on-board digital sensor MCP9808 on the AVR-IoT WG board. The AQM application uses the MCP9808 digital sensor for monitoring temperature.

2.3.2 Pin Manager

- The temperature and humidity output pins of the SHT31 sensor are connected to the port pins PD6 and PD7, respectively.

- To configure PD6 and PD7 pins as ADC channels, go to *Pin Manager Grid View*. Configure PD6 and PD7 for **ADC0**. Pin PD5, which is connected to the light sensor output, is preconfigured as the ADC channel by the AVR-IoT WG Sensor Node example project. Deselect the **pin PD5 configuration** as ADC channel, because the light sensor is not used in the indoor AQM application.

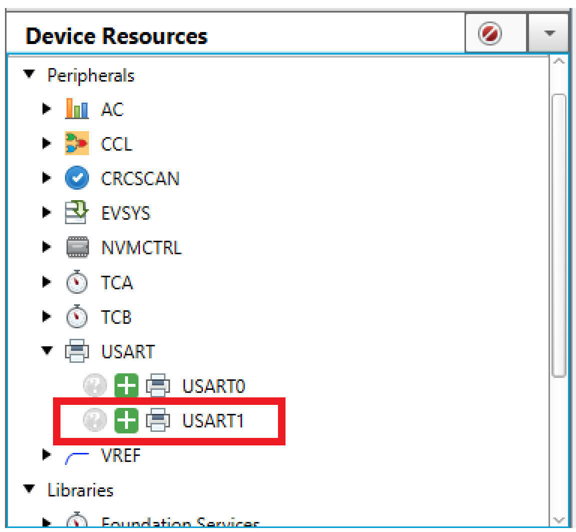
Figure 2-6. ADC Pins Configuration Using Pin Manager



2.3.3 UART1

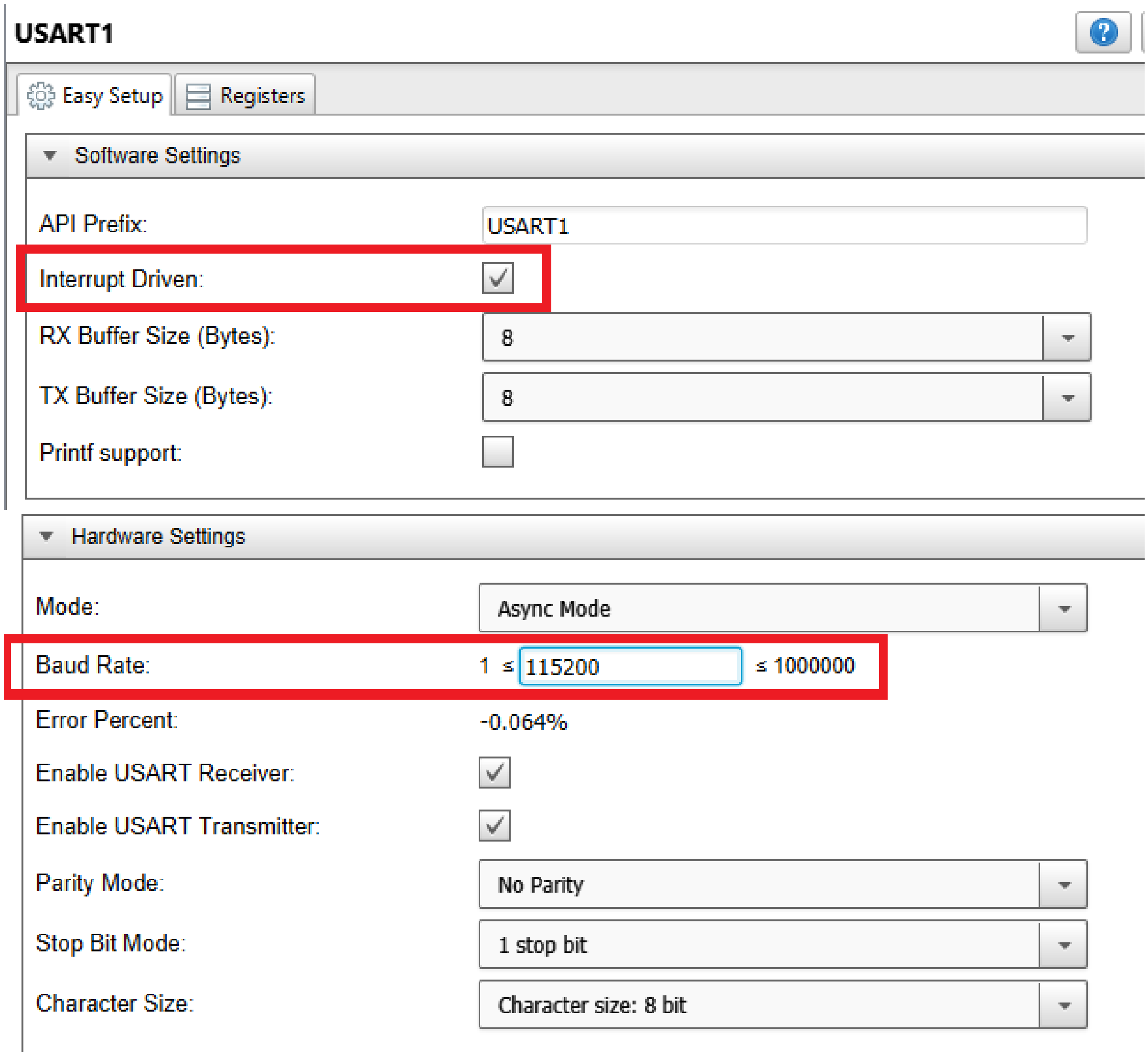
- The $PM_{2.5}$ sensor SPS30 communicates with the microcontroller through the UART interface.
- To configure UART, add the **USART1** driver from the *Device Resources* section. Navigate through *Device Resources → Peripherals → USART → USART1* and click the **Add** symbol.

Figure 2-7. USART1 Peripheral Selection



- The USART1 configuration window opens.
- In the *Software Settings* section, enable the *Interrupt Driven* option. Set the *Baud Rate* to *115200* as the SPS30 sensor supports a baud rate of *115200*.

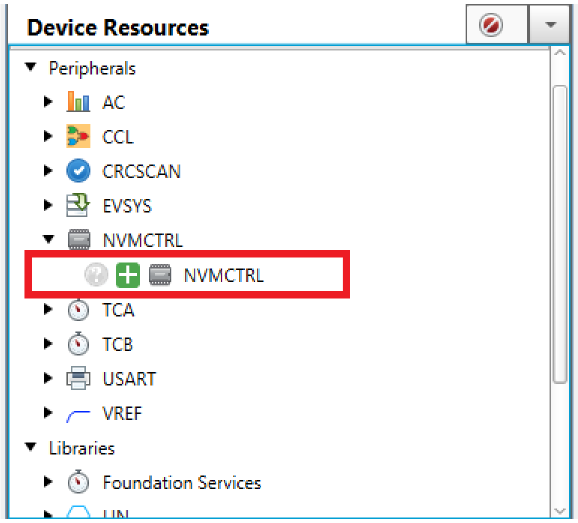
Figure 2-8. USART1 Configuration



2.3.4 NVM

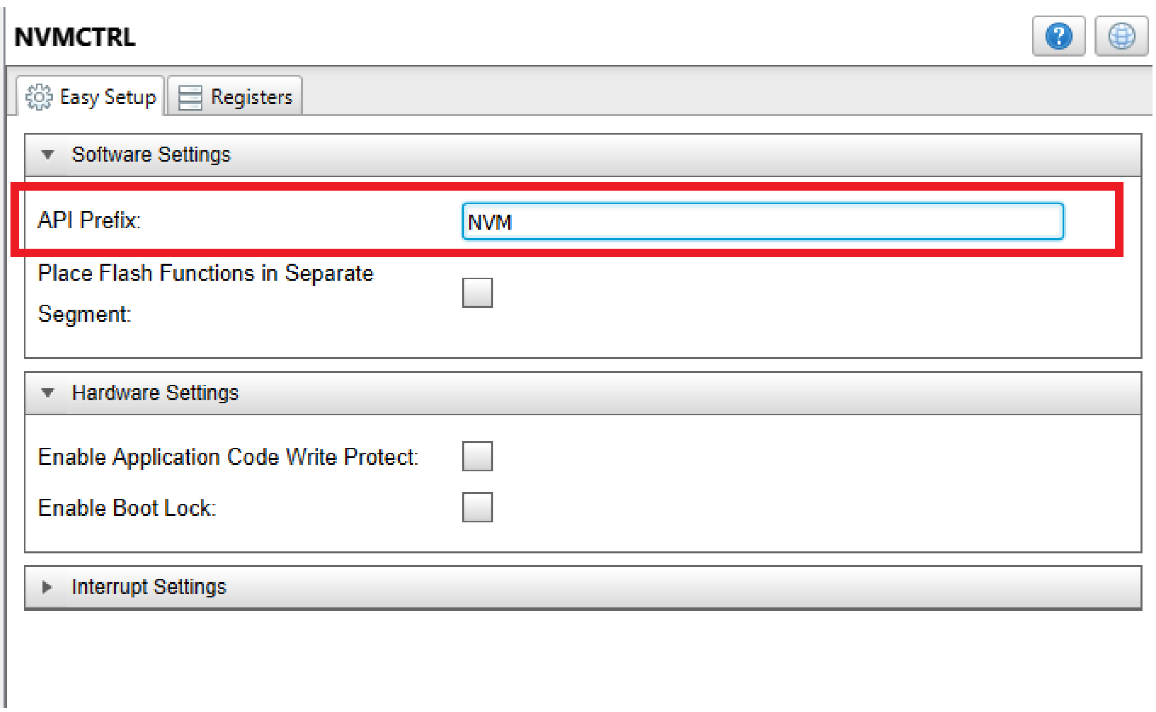
- The internal EEPROM is used to store the default parameters and external EEPROM's indices. The internal EEPROM is accessed using the NVM control peripheral.
- To configure NVM control, add the **NVM control** driver from the *Device Resources* section. Navigate through *Device Resources → Peripherals → NVMCTRL* and click the **Add** symbol.

Figure 2-9. NVM Control Peripheral Selection



- The NVMCTRL configuration window opens.
- In the *Software Settings* section, configure the *API Prefix* to *NVM*.

Figure 2-10. NVM Control Configuration

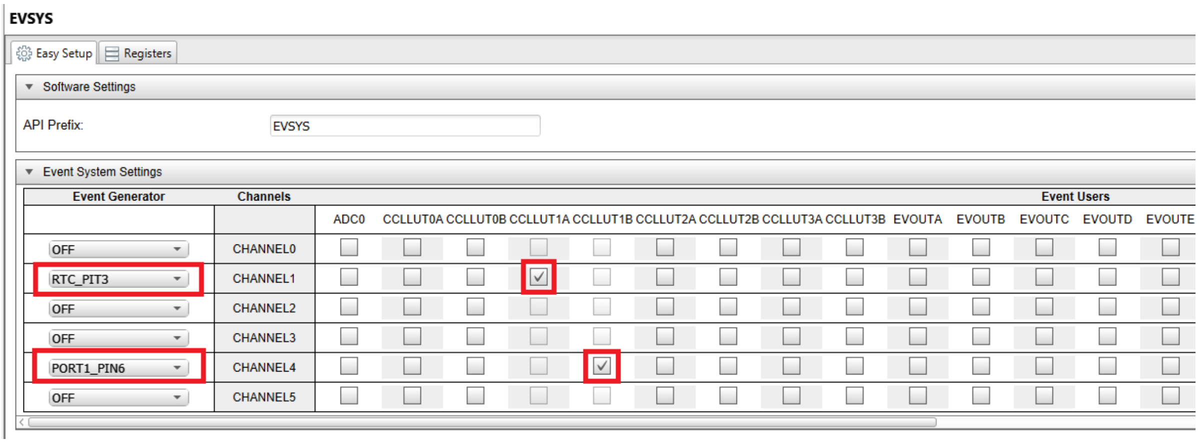


2.3.5 Event System

- The on-board switch on the AVR-IoT WG board is used to turn on the OLED display and to change the parameters to be displayed on the OLED. The on-board switch SW0 on the AVR-IoT WG board is connected to the PORT1_PIN6 of the microcontroller. The switch debounce mechanism is implemented without any software intervention, using the Event System and CCL peripherals of the microcontroller.
- To configure the Event System, add the **Event System** driver from the *Device Resources* section. Navigate through *Device Resources → Peripherals → EVSYS* and click the **Add** symbol. The *EVSYS* window opens.
- Event channel 1 is used to connect the Periodic Interrupt Timer output to the LUT1 of the CCL, and event channel 4 is used to connect the switch port pin to the LUT1 of the CCL.

- Configure the *Event Generator* as *RTC_PIT3*, and the *Event User* as *CCLLUT1A* for *CHANNEL 1*, as shown in Figure 2-11.
- Configure *PORT1_PIN6* as *Event Generator*, and *CCLLUT1B* as *Event User* for *CHANNEL 4*, as shown in Figure 2-11.

Figure 2-11. Event System Configuration

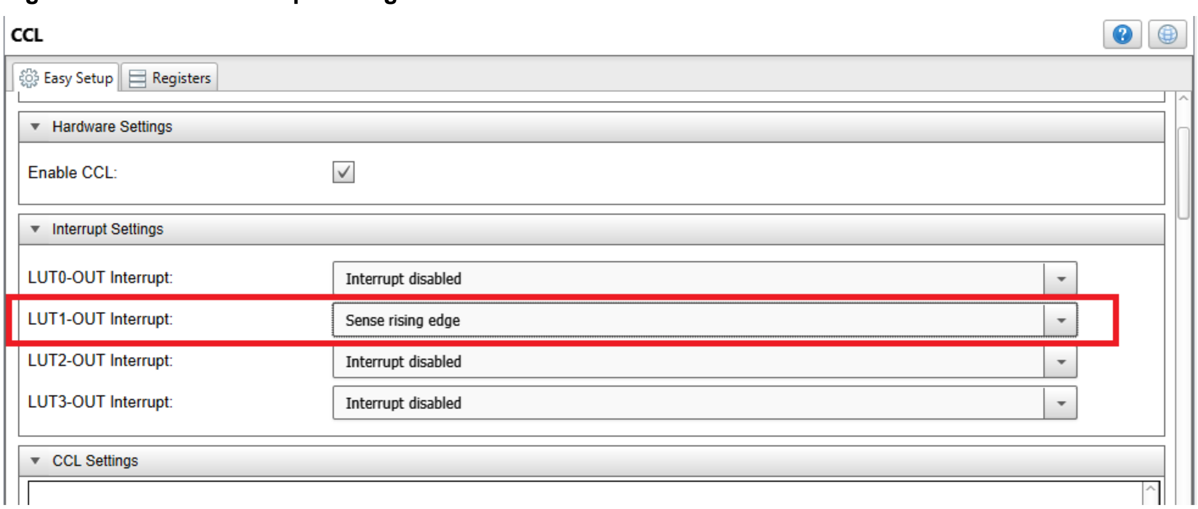


Note: The *Load peripheral CCL* warning appears to load the CCL peripheral. Ignore the warning as the next subsection explains about loading CCL peripheral.

2.3.6 CCL

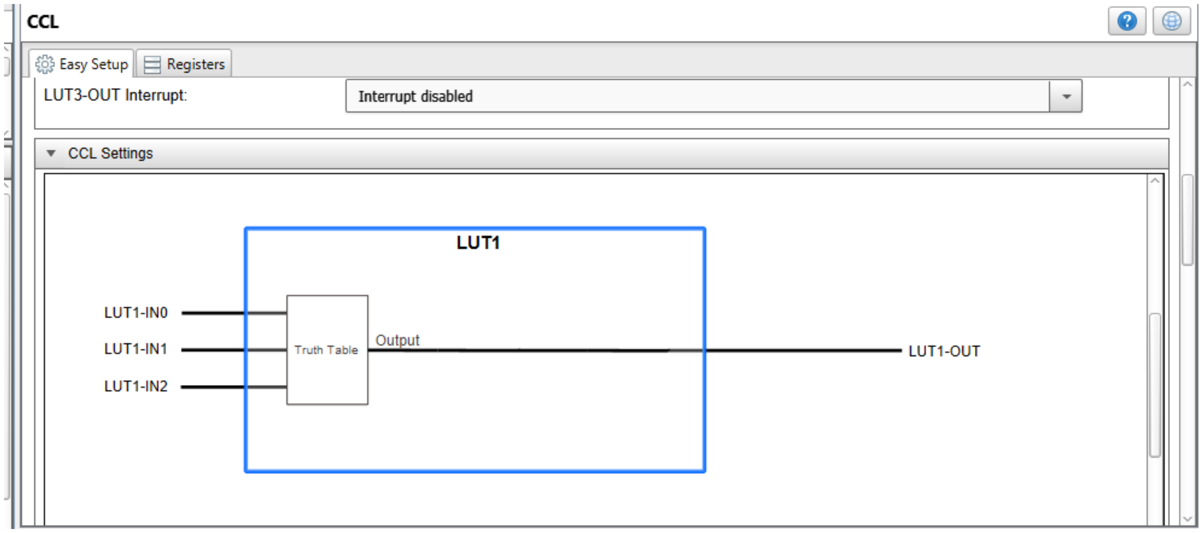
- To implement switch debounce, LUT1 of the CCL peripheral with filter option is used.
- The CCL peripheral generates an interrupt for each switch press event.
- To configure the CCL peripheral, add the **CCL** driver from *Device Resources*. Navigate through *Device Resources → Peripherals → CCL* and click the **Add** symbol.
- The CCL configuration window opens.
- In the *Interrupt Settings* window, choose *Sense rising edge* from the drop-down menu of the *LUT1-OUT Interrupt* option.

Figure 2-12. LUT1 Interrupt Configuration



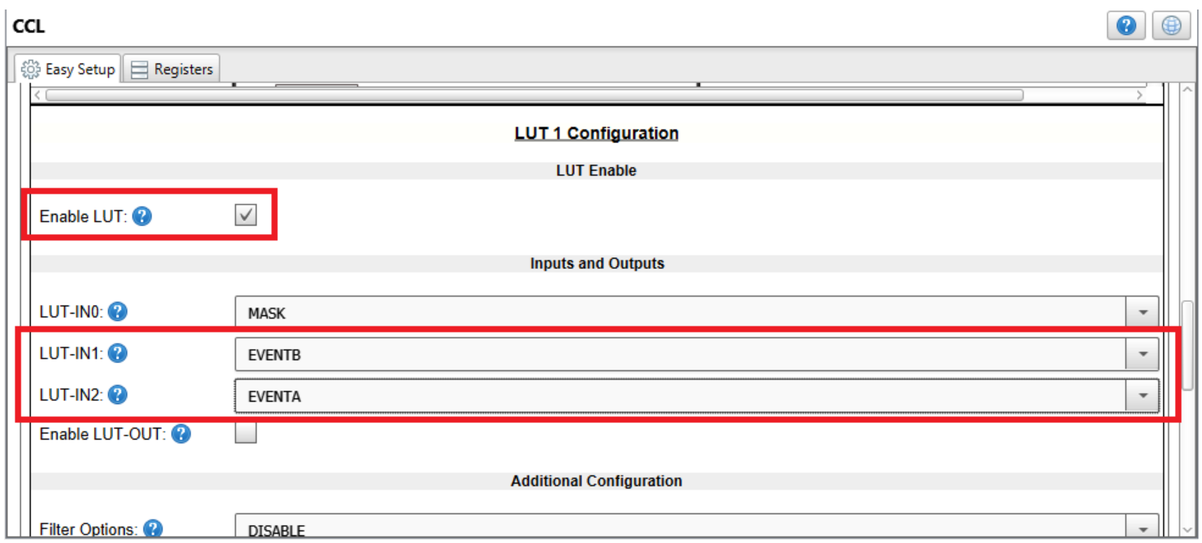
- Select **LUT1** by clicking **LUT1** in the *CCL Settings* section.

Figure 2-13. LUT1 Selection



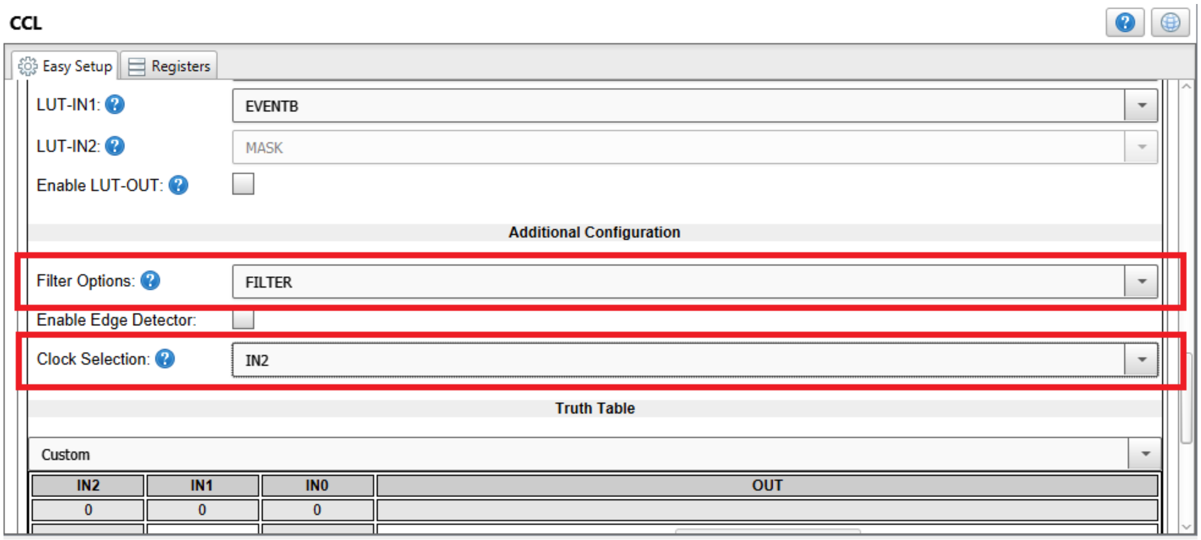
- To enable LUT1, check the **Enable LUT** option in the *LUT1 Configuration* section, as shown in Figure 2-14.
- Select **EVENTB** from the drop-down menu of the *LUT-IN1* option, as shown in Figure 2-14. As *PORT1_PIN6* is connected to *EVENTB* through event channel 4, the switch signal acts as the first input to LUT1.
- Select **EVENTA** from the drop-down menu of the *LUT-IN2* option, as shown in Figure 2-14. As *RTC_PIT3* is connected to *EVENTA* through event channel 1, it provides the clock signal to the CCL filter.

Figure 2-14. LUT1 Configuration



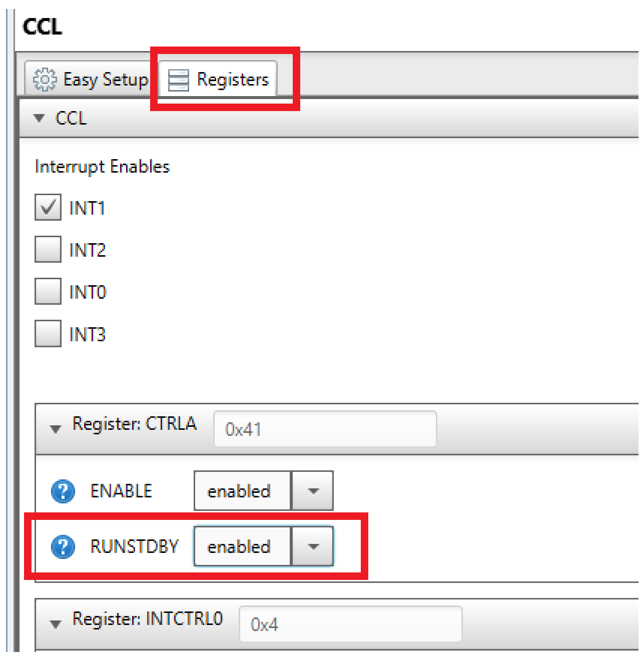
- In the *Additional Configuration* section of the CCL configuration window, configure *Filter Options* to *FILTER* to enable the filter, as shown in Figure 2-15.
- Configure *Clock Selection* to *IN2*.

Figure 2-15. Enable Filter Mode and Clock Selection in CCL LUT1



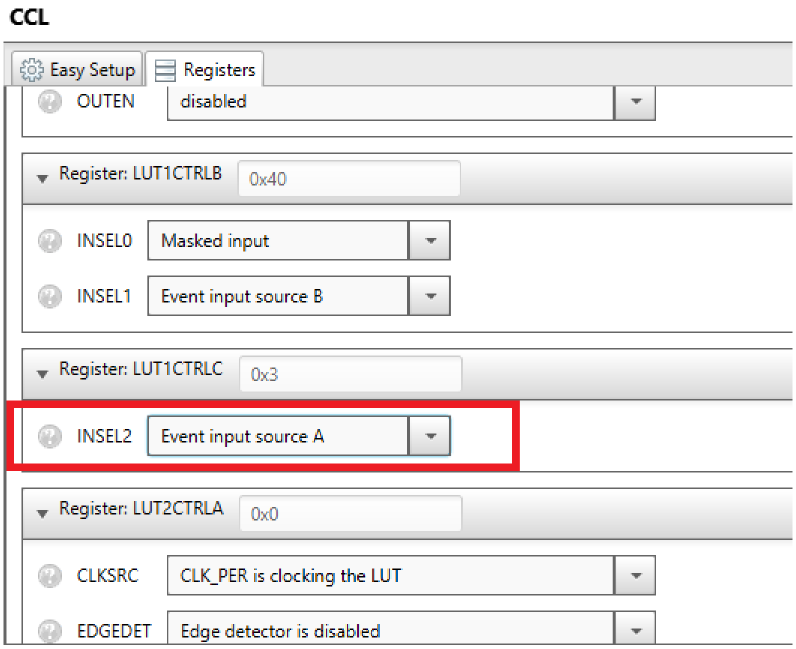
- The microcontroller goes to Standby sleep mode periodically. To detect switch press event while the microcontroller is in Standby sleep mode, the CCL has to run in Standby sleep mode. In the **Registers** tab of the CCL configuration window, configure the *RUNSTDBY* option to *enabled* under the CTRLA register.

Figure 2-16. Run in Standby Bit Configuration



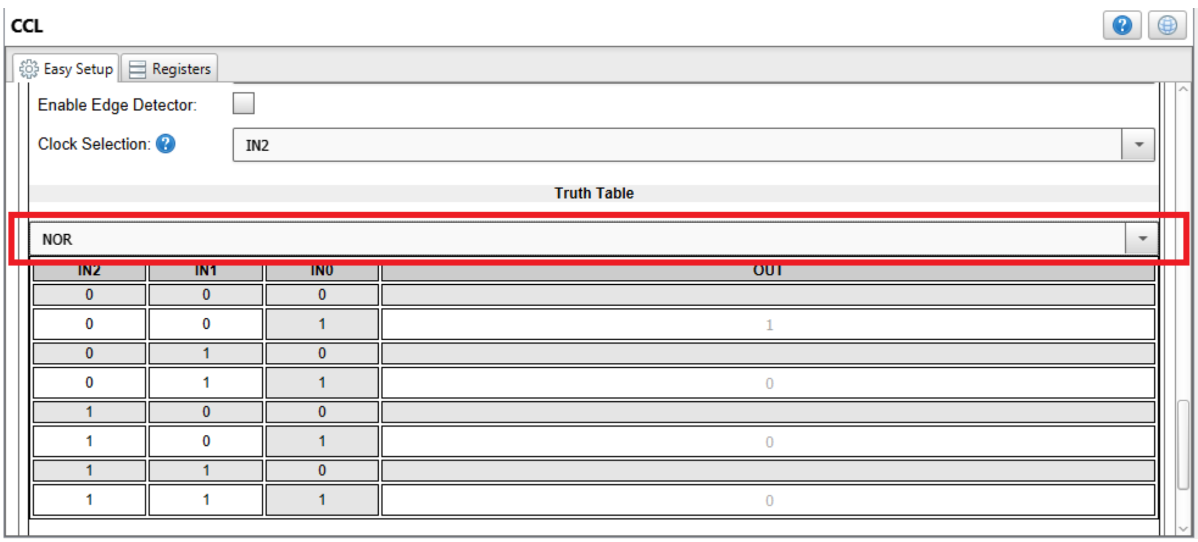
- Configure *INSEL2* as *Event input source A* in the LUT1CTRLC register. INSEL2 is reconfigured because its configuration is changed to masked input after Clock Selection.

Figure 2-17. LUT1 Input 2 Configuration



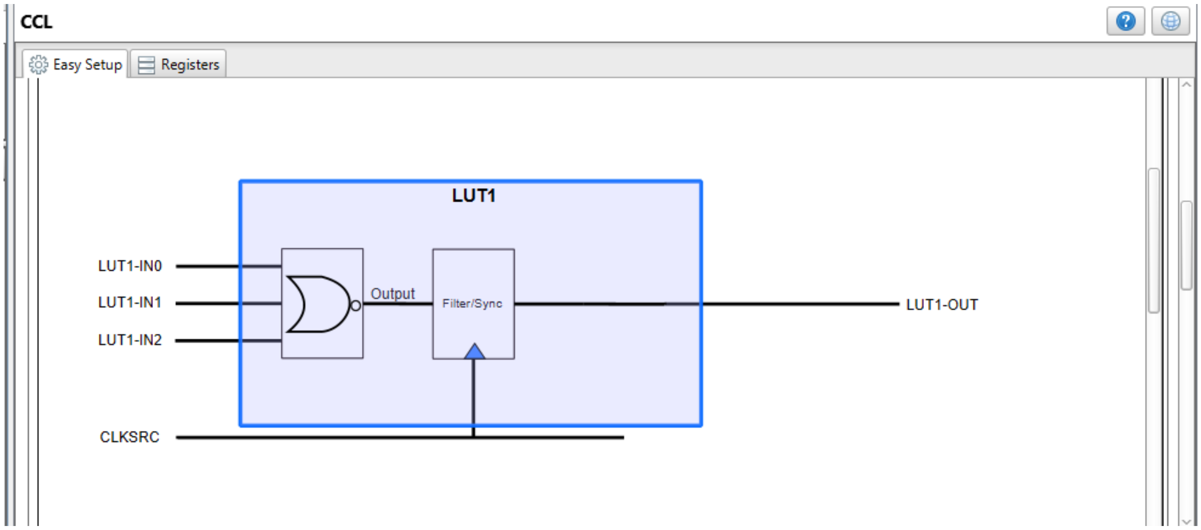
- Open the *Truth Table* in the **EASY Setup** tab of the CCL configuration window and configure it to *NOR*.

Figure 2-18. Truth Table Configuration



- After the completion of all the above configurations, LUT1 looks as shown in Figure 2-19.

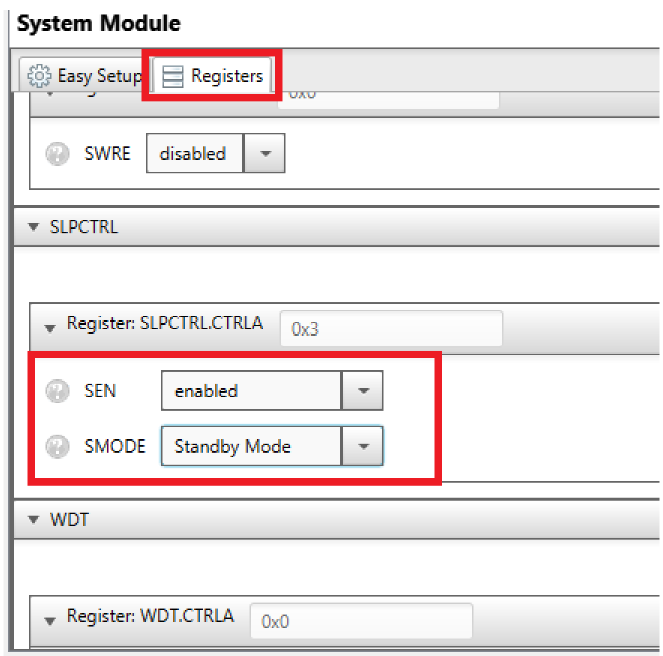
Figure 2-19. LUT1 after Configurations



2.3.7 Sleep Mode

- The microcontroller goes to Standby sleep mode after completion of all the tasks.
- To configure the Sleep mode, click **System Module** in the *Project Resources* section. The *System Module* window opens.
- Click the **Registers** tab, as shown in Figure 2-20.
  Note: The **Registers** tab may take some time to open.
- Go to the SLPCTRL.CTRLA register, and enable the Sleep mode by configuring *SEN* to *enabled* and *SMODE* to *Standby Mode*.

Figure 2-20. Standby Sleep Mode Configuration

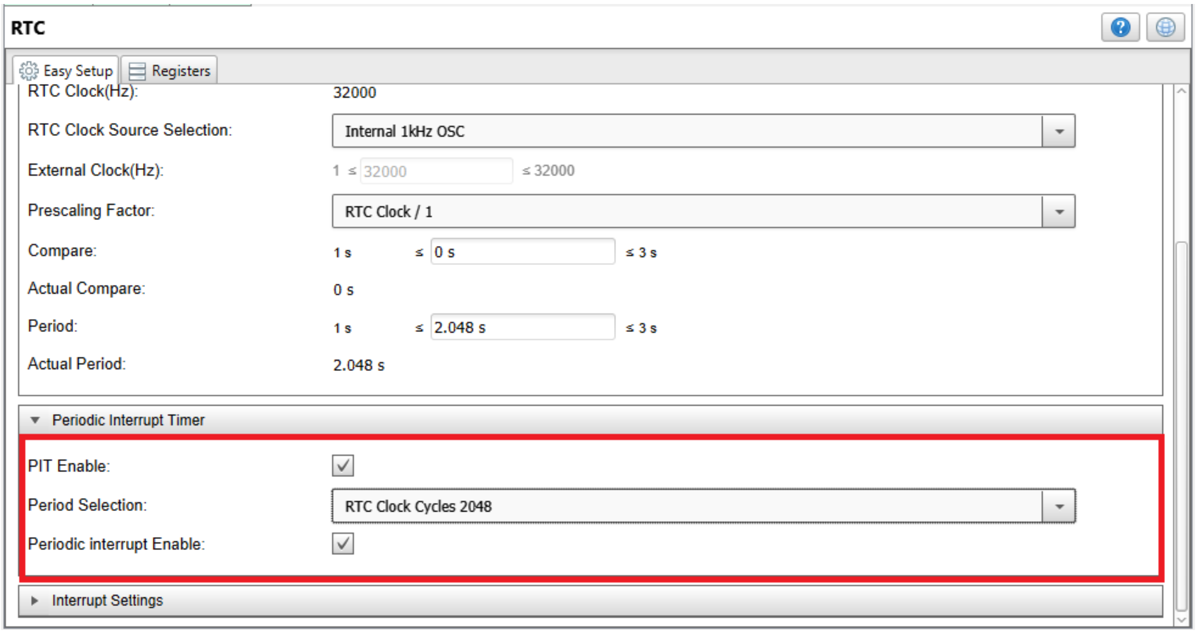


2.3.8 PIT

- The microcontroller periodically wakes up from sleep mode whenever it gets a Peripheral Interrupt Timer (PIT) interrupt. The PIT is part of the RTC module. To configure the PIT, select **RTC** from the *Project Resources* by navigating through *Project Resources → Peripherals → RTC*.

- In the *Periodic Interrupt Timer* section, enable the *PIT Enable* option. Configure *Period Selection* to *RTC Clock cycle 2048*. This divides the RTC clock by 2048.
- Enable the *Periodic interrupt Enable* option.

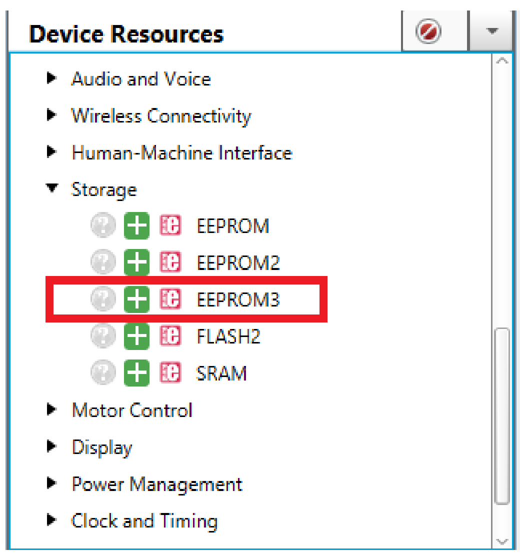
Figure 2-21. PIT Configuration



2.3.9 EEPROM 3 Click

- Air quality sensors readings are stored on the external EEPROM present on the EEPROM 3 click board. The EEPROM 3 click board is supported by the MCC framework. Include the **EEPROM3** click driver to the project from the *Device Resources* section. Navigate through *Device Resources → Mikro-E Clicks → Storage → EEPROM3* and click the **Add** symbol.

Figure 2-22. EEPROM 3 Click Addition



- The EEPROM 3 click configuration window opens. There is no additional configuration required for the EEPROM 3 click.

2.4 Generating MCC Code Files

- Click the **Generate** button, located on the left-hand corner of the *Resource Manager → Project Resources* window, and wait for the code generation to complete.

Note: A pop-up might appear, saying that the configuration has warnings. Click **YES** to continue as all the warnings have been taken care of.

2.5 Adding Application Code and Sensor Drivers

- The firmware is available for download on GitHub, as below:

View Code Example on GitHub (MPLAB® X)
Click to browse repository

- Copy the **application** and **sensor** folders from the downloaded firmware and place them inside the project folder.
- To add both folders to the project, go to the *Project* window and select the **Source Files** folder. Right click to open the content menu and select **Add Existing Items from folder** option. Click **Add folder**. Pick the **application** folder that was just copied containing the application code, then click the **select** option. Repeat these steps to add the **sensor** folder containing the sensor drivers. After selecting both folders, click the **Add** button.
- For a more detailed procedure on folders addition to a project, refer to the page on Microchip Developer Help website.

2.6 Modifying MCC Generated Code

The MCC generated AVR-IoT WG source code and some of the peripherals' APIs need modifications. The user needs to compare both the AVR-IoT WG source code generated by MCC and the firmware available for download on GitHub, as below, to find more details about the changes. Table 2-2 lists files and functions along with the change description, where changes are required.

View Code Example on GitHub (MPLAB® X)
Click to browse repository

Table 2-2. List of Changes to MCC Generated Code

Source File	Function	Change Description
`main.c`	`SendToCloud()`	A dedicated task is created to send data to the cloud. Thus, this function is no longer required.
	`main()`	Parameter initialization and Cloud configuration.
`mcc_generated_files/src/ccl.c`	`ISR(CCL_CCL_vect)`	This interrupt occurs upon switch press.
`mcc_generated_files/src/rtc.c`		Set PIT ISR callback function.
	`RTC_Initialize()`	Wait for all registers to be synchronized.
	`ISR(RTC_PIT_vect)`	Clear the PIT flag at the end.

...........continued

Source File	Function	Change Description
mcc_generated_files/ application_manager.c	application_init()	Turn on the display.
		Created application tasks.
	MAIN_dataTask()	No call to SendtoCloud() function.
mcc_generated_files/ cryptoauthlib/lib/jwt/ atca_jwt.c	atca_jwt_finalize()	Updated atcab_hw_sha2_256() function uses the hardware SHA function during the JWT creation (to save memory).
mcc_generated_files/cloud/ crypto_client/crypto_client.c		Commented out ATCA_PRINTF definition (to save memory).
mcc_generated_file/config/ cryptoauthlib_config.h		Commented out ATCA_PRINTF definition (to save memory).
mcc_generated_file/ EEPROM3_driver.c		I²C functions name changed as per the MCC generated I²C drivers.
mcc_generated_files/ cryptoauthlib/lib/hal/hal_i2c.c	I2C_0_wake_up()	I2C_BUSY function name changed as per the MCC generated I²C drivers.
	hal_i2c_writeNBytes()	
	hal_i2c_readNBytes()	

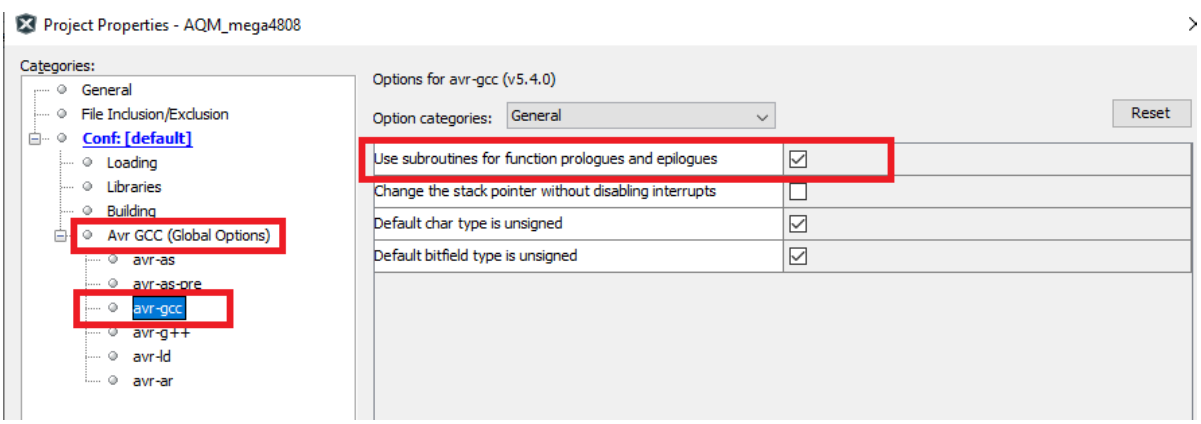
2.7 Compiler Settings

This section explains the compiler settings which are required for code optimization.

2.7.1 Activate "Use subroutines for function prologue/epilogue"

- Right click on the project name and select **Properties**. Click **avr-gcc** under AVR GCC Global Options. Enable the *Use subroutines for function prologue/epilogue* option.

Figure 2-23. Configuration of "Use subroutines for function prologue/epilogue"

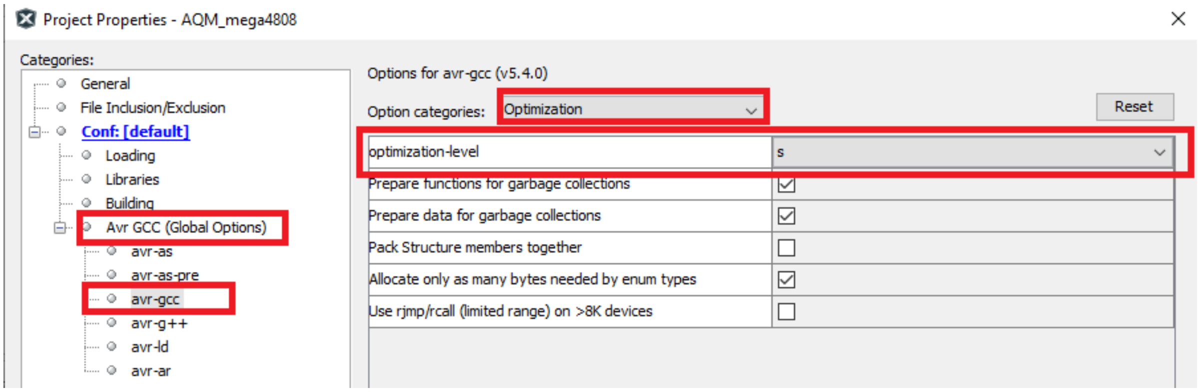


- Click **Apply** and then **OK**.
  Note: This is required to optimize the code size.

2.7.2 Compiler Optimization Setting

- The application firmware is optimized for size to efficiently use the code memory. Click on the project name and select **Properties**. Click **avr-gcc** under AVR GCC Global Options. Select **Optimization** from the option categories drop-down menu. Set *optimization-level* to *"s"*.

Figure 2-24. Compiler Optimization Setting



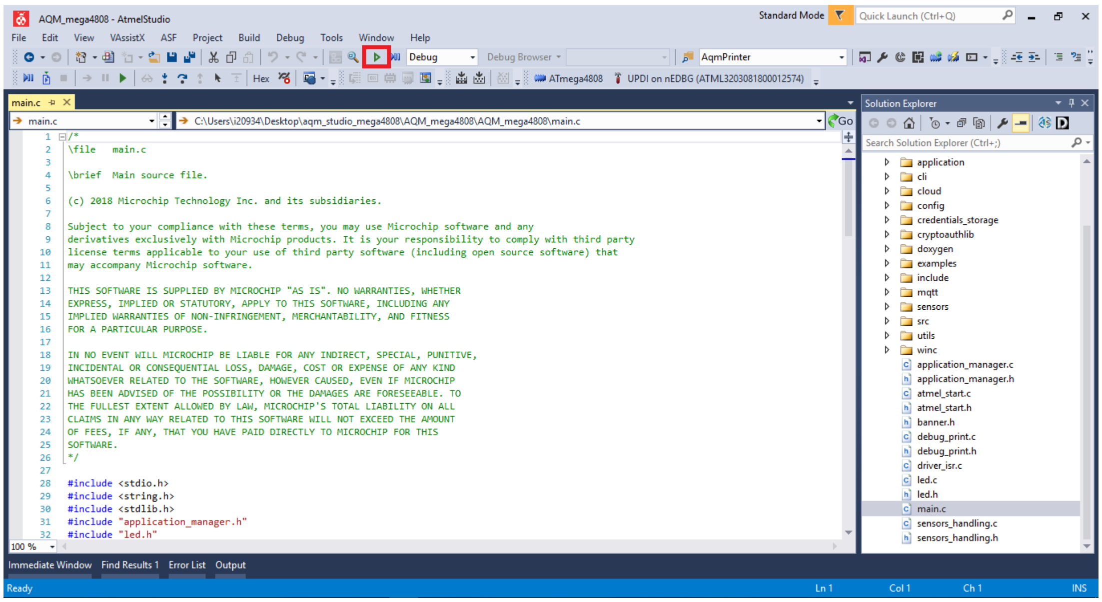
- Click **Apply** and then **OK**.

3. Programming the Application

3.1 Programming with Atmel Studio

- To program the device, connect the AVR-IoT WG board to the PC using a Micro-USB cable.
- Click the **Start without Debugging** (Ctrl + Alt + F5) icon.

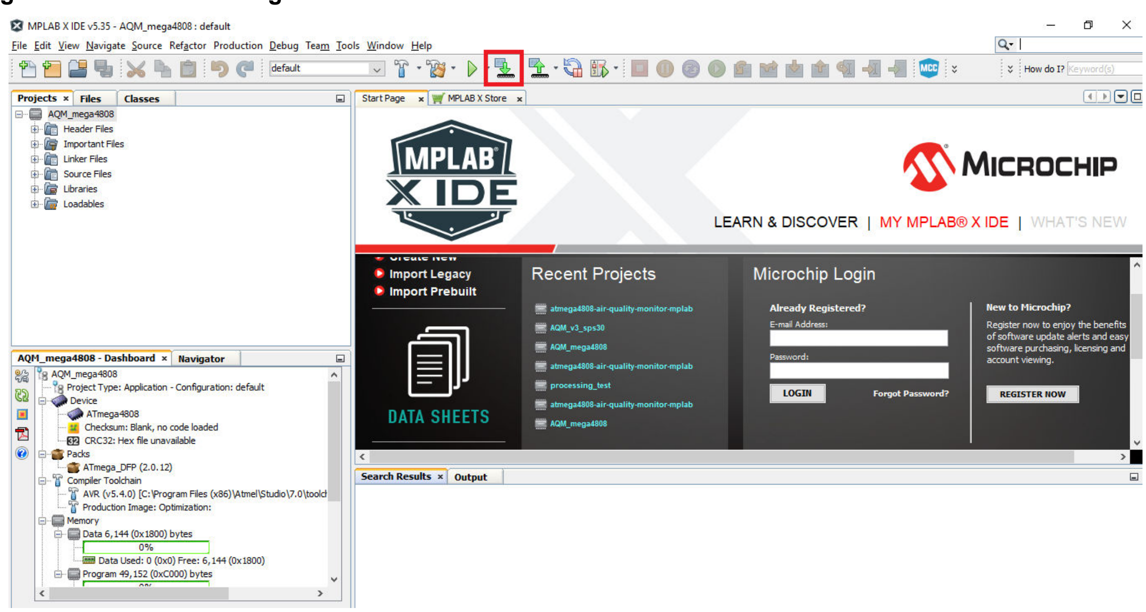
Figure 3-1. Start without Debugging



3.2 Programming with MPLAB® X IDE

- To program the device, connect the AVR-IoT WG board to the PC using a Micro-USB cable.
- Click the **Make and Program Device** icon.

Figure 3-2. Make and Program Device



4. Conclusion

This document covers details of the indoor AQM application firmware creation on top of the AVR-IoT WG source code. The document also explains the process of how to generate AVR-IoT WG source code with Atmel START and MCC.

5. Revision History

Document Revision	Date	Comments
A	03/2020	Initial document release

The Microchip Website

Microchip provides online support via our website at http://www.microchip.com/. This website is used to make files and information easily available to customers. Some of the content available includes:

- **Product Support** – Data sheets and errata, application notes and sample programs, design resources, user's guides and hardware support documents, latest software releases and archived software
- **General Technical Support** – Frequently Asked Questions (FAQs), technical support requests, online discussion groups, Microchip design partner program member listing
- **Business of Microchip** – Product selector and ordering guides, latest Microchip press releases, listing of seminars and events, listings of Microchip sales offices, distributors and factory representatives

Product Change Notification Service

Microchip's product change notification service helps keep customers current on Microchip products. Subscribers will receive email notification whenever there are changes, updates, revisions or errata related to a specified product family or development tool of interest.

To register, go to http://www.microchip.com/pcn and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Embedded Solutions Engineer (ESE)
- Technical Support

Customers should contact their distributor, representative or ESE for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in this document.

Technical support is available through the website at: http://www.microchip.com/support

Microchip Devices Code Protection Feature

Note the following details of the code protection feature on Microchip devices:

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as "unbreakable."

Code protection is constantly evolving. We at Microchip are committed to continuously improving the code protection features of our products. Attempts to break Microchip's code protection feature may be a violation of the Digital Millennium Copyright Act. If such acts allow unauthorized access to your software or other copyrighted work, you may have a right to sue for relief under that Act.

Legal Notice

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with

Trademarks

ISBN: 978-1-5224-5704-6

Quality Management System

For information regarding Microchip's Quality Management Systems, please visit http://www.microchip.com/quality.

Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
Corporate Office 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 Technical Support: http://www.microchip.com/support Web Address: http://www.microchip.com **Atlanta** Duluth, GA Tel: 678-957-9614 Fax: 678-957-1455 **Austin, TX** Tel: 512-257-3370 **Boston** Westborough, MA Tel: 774-760-0087 Fax: 774-760-0088 **Chicago** Itasca, IL Tel: 630-285-0071 Fax: 630-285-0075 **Dallas** Addison, TX Tel: 972-818-7423 Fax: 972-818-2924 **Detroit** Novi, MI Tel: 248-848-4000 **Houston, TX** Tel: 281-894-5983 **Indianapolis** Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380 **Los Angeles** Mission Viejo, CA Tel: 949-462-9523 Fax: 949-462-9608 Tel: 951-273-7800 **Raleigh, NC** Tel: 919-844-7510 **New York, NY** Tel: 631-435-6000 **San Jose, CA** Tel: 408-735-9110 Tel: 408-436-4270 **Canada - Toronto** Tel: 905-695-1980 Fax: 905-695-2078	**Australia - Sydney** Tel: 61-2-9868-6733 **China - Beijing** Tel: 86-10-8569-7000 **China - Chengdu** Tel: 86-28-8665-5511 **China - Chongqing** Tel: 86-23-8980-9588 **China - Dongguan** Tel: 86-769-8702-9880 **China - Guangzhou** Tel: 86-20-8755-8029 **China - Hangzhou** Tel: 86-571-8792-8115 **China - Hong Kong SAR** Tel: 852-2943-5100 **China - Nanjing** Tel: 86-25-8473-2460 **China - Qingdao** Tel: 86-532-8502-7355 **China - Shanghai** Tel: 86-21-3326-8000 **China - Shenyang** Tel: 86-24-2334-2829 **China - Shenzhen** Tel: 86-755-8864-2200 **China - Suzhou** Tel: 86-186-6233-1526 **China - Wuhan** Tel: 86-27-5980-5300 **China - Xian** Tel: 86-29-8833-7252 **China - Xiamen** Tel: 86-592-2388138 **China - Zhuhai** Tel: 86-756-3210040	**India - Bangalore** Tel: 91-80-3090-4444 **India - New Delhi** Tel: 91-11-4160-8631 **India - Pune** Tel: 91-20-4121-0141 **Japan - Osaka** Tel: 81-6-6152-7160 **Japan - Tokyo** Tel: 81-3-6880- 3770 **Korea - Daegu** Tel: 82-53-744-4301 **Korea - Seoul** Tel: 82-2-554-7200 **Malaysia - Kuala Lumpur** Tel: 60-3-7651-7906 **Malaysia - Penang** Tel: 60-4-227-8870 **Philippines - Manila** Tel: 63-2-634-9065 **Singapore** Tel: 65-6334-8870 **Taiwan - Hsin Chu** Tel: 886-3-577-8366 **Taiwan - Kaohsiung** Tel: 886-7-213-7830 **Taiwan - Taipei** Tel: 886-2-2508-8600 **Thailand - Bangkok** Tel: 66-2-694-1351 **Vietnam - Ho Chi Minh** Tel: 84-28-5448-2100	**Austria - Wels** Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 **Denmark - Copenhagen** Tel: 45-4485-5910 Fax: 45-4485-2829 **Finland - Espoo** Tel: 358-9-4520-820 **France - Paris** Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 **Germany - Garching** Tel: 49-8931-9700 **Germany - Haan** Tel: 49-2129-3766400 **Germany - Heilbronn** Tel: 49-7131-72400 **Germany - Karlsruhe** Tel: 49-721-625370 **Germany - Munich** Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 **Germany - Rosenheim** Tel: 49-8031-354-560 **Israel - Ra'anana** Tel: 972-9-744-7705 **Italy - Milan** Tel: 39-0331-742611 Fax: 39-0331-466781 **Italy - Padova** Tel: 39-049-7625286 **Netherlands - Drunen** Tel: 31-416-690399 Fax: 31-416-690340 **Norway - Trondheim** Tel: 47-72884388 **Poland - Warsaw** Tel: 48-22-3325737 **Romania - Bucharest** Tel: 40-21-407-87-50 **Spain - Madrid** Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 **Sweden - Gothenberg** Tel: 46-31-704-60-40 **Sweden - Stockholm** Tel: 46-8-5090-4654 **UK - Wokingham** Tel: 44-118-921-5800 Fax: 44-118-921-5820