Introduction to MPLAB Harmony v3 for Atmel Studio and ASF Users

AN3346

Abstract

This document introduces the MPLAB® X IDE and MPLAB Harmony v3 framework for users to get started with new tools and solutions for 32-bit Arm® microcontrollers. This document helps Atmel® Studio, Atmel START, and ASF users to familiarize with MPLAB X IDE and MPLAB Harmony v3.

This document also describes MPLAB X IDE features, MPLAB Harmony v3 architecture, and cite references to equivalent features in Atmel Studio or ASF to enable better understanding.

Table of Contents

1. MPLAB X Integrated Development Environment (IDE)

MPLAB X IDE is a software tool that helps in developing embedded applications on Microchip microcontroller units (MCUs) and digital signal controllers. MPLAB X IDE can be installed on Windows, Linux and MAC operating systems. This helps the user to continue the software development for an MCU without any operating system dependencies.

MPLAB X IDE comprises of the following:

- A full-featured programmer-friendly text editor
- Utilities required for MCU firmware development, such as compiler, assembler, Linker
- Debugger engine with a set of powerful debug utilities, such as breakpoints, watch windows, I/O views
- A project manager that provides integration and communication between IDE and language tools
- Software simulators
- Plugins (to add to the capabilities of MPLAB X IDE), which can be installed and used

MPLAB X IDE supports a wide range of Microchip's 8-bit, 16-bit and 32-bit MCU and MPU portfolios. This document is specific to 32-bit (Arm and PIC32) microcontrollers. To check a specific device for support, refer to the device support documents listed in the References section of this document.

1.1 Installation and Toolchain

To install MPLAB X IDE, users need to download the appropriate installer from the Microchip website and follow the prompts. MPLAB X IDE is a NetBeans-based platform.

The toolchains for MPLAB X IDE must be installed separately because it is not bundled with the MPLAB installer. This provides flexibility to the user to install MPLAB XC8, MPLAB XC16 or MPLAB XC32-based compilers (or all the three compilers).

MPLAB XC8 supports all 8-bit PIC® and AVR® MCUs. MPLAB XC16 supports all 16-bit PIC MCUs and dsPIC® Digital Signal Controllers (DSCs). MPLAB XC32/32++ supports all 32-bit PIC and SAM MCUs and MPUs.

Note: The toolchain installation is handled differently in Atmel Studio. The Atmel Studio installer package is bundled with toolchains for AVR, AVR32 and SAM devices, and users can select required toolchain while installing. Therefore unlike Atmel Studio, users must install the MPLAB XC8, MPLAB XC16, or MPLAB XC32 compilers separately.

1.2 Licensing

The toolchain (MPLAB XC8, MPLAB XC16, or MPLAB XC32) comes with basic optimization features for code size and speed. It also comes with a 60-day free trial license. If software development requires advance-level of optimization, then PRO licenses are available to unlock the full potential of the MPLAB XC C compilers. Refer to the list of flexible licensing options available on the Microchip website.

Notes:

1. Atmel Studio comes with a standard Arm GNU C compiler for free.
2. Free C++ licensing is available in MPLAB X IDE.

1.3 MPLAB Packs to Install Device Family Packs (DFPs)

The MPLAB Pack Manager can add or remove device support from MPLAB X IDE. A newer version of the DFP or new device support can be added using the Packs in MPLAB X IDE, there is no need to install MPLAB X IDE from scratch. This eliminates the patching of MPLAB X IDE or smaller installer to patch few files.

To open the Packs, from MPLAB X IDE select *Tools > Packs*.

Note: Atmel Studio handles the device part packs similarly.

1.4 Key Features

The following are key features of the MPLAB X IDE:

- Multiple configurations for the same projects:
  - A single project can build the same set of source files in various ways. Each configuration has its own Compiler Toolchain, Connected Hardware Tool, and Device.
- Supports multiple versions of the same compiler
- Intelligent text editor with the following features:
  - Provides auto-completion
  - Track changes within system using local history
  - Users can configure required code format style
- Customizable workspace and multi-screen support:
  - Tasks window
  - Navigator window
  - Project Status window
  - Software Call stack window
- Extend the MPLAB X IDE functionality with the MPLAB IDE Plugin Manager

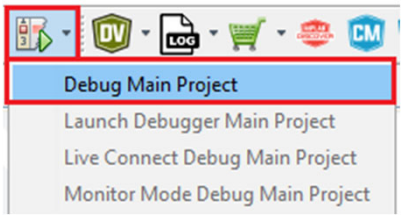
1.5 Programming and Debugging Microcontrollers

The MPLAB X IDE supports Microchip various programming and debugging tools, which can be referenced from the device-support documents listed in the section References.

The user can start debugging session in the MPLAB X IDE by choosing any one of these options:

- From the toolbar, click  (the Debug Project icon).
- Choose the **Debug Main Project** from the drop-down menu:

Figure 1-1. Debug Main Project



Note: If multiple projects are opened in the MPLAB X IDE, the one which is set as the main project will be targeted for the debug session.

During the debugging session, the following basic and advanced features required for debugging are available in the MPLAB X IDE:

- Step Into, Step Over, Step Out, Run to cursor
- Setting breakpoints
- Hold in reset, Run
- Viewing a variable in Watch window
- Device peripheral register content in I/O view

- Device's memory content in Memory view
- The Disassembly view of the code when instruction level debugging is needed

Note: Simulator support is available for few devices. Refer to the device support document for additional information, which are listed in the section References.

1.6 MPLAB X IDE Plugin

Users can extend the functionality of the MPLAB X IDE by adding plugins. The user can select the functionality based on their application requirement. Users might need advance features only based on their application requirement; however, all users might not use Motor Bench Development Suite, RTOS Viewer, Power Monitor, MPLAB Data Visualizer and so on, therefore these are provided as plugins. Due to this the MPLAB X IDE can be kept smaller in size, and it prevent users from loading the entire features set. This will reduce the memory and bandwidth requirement in the PC which enhances user experience with the tool.

Note: The MPLAB X IDE plugins are comparable to the extensions in Atmel Studio.

1.7 MPLAB Code Configurator (MCC) Plugin

The MPLAB® Code Configurator (MCC) is a free graphical programming environment that generates easy-to-understand C code for the project. It provides an intuitive interface to configure peripherals and functions specific to the application. This tool can be launched from *Tools > Embedded > MPLAB® Code Configurator v5: Open/Close*.

The MCC Content Manager Tool within the MCC helps to select the type of content and provides better flexibility by supporting content management and versioning at an individual component level. This document describes content delivery using the GitHub distribution and repository structure in detail.

Note: MPLAB Harmony v3 is a fully integrated embedded software development framework for 32-bit microcontrollers (MCUs) and microprocessors (MPUs).

2. MPLAB Harmony v3 Software Development Framework

The MPLAB Harmony v3 architecture is discussed in detail in the following sections.

2.1 Introduction to MPLAB Harmony v3

MPLAB Harmony v3 together with the MPLAB X IDE, enhances the application development experience through the simple and user-friendly peripheral libraries (PLIBs), abstracted drivers, and modular software downloads.

The MPLAB® Code Configurator (MCC) is a free graphical programming environment that generates easy-to-understand C code for the project. It provides an intuitive interface to configure peripherals and functions specific to the application. MCC supports 8-bit, 16-bit, and 32-bit devices, including PIC®, AVR®, SAM microcontrollers (MPUs, MCUs), and dsPIC® Digital Signal Controllers (DSCs). The MCC consists of three content types: MCC Melody, MCC Classic, and MPLAB Harmony, offering application libraries and system and peripheral drivers for embedded software development.

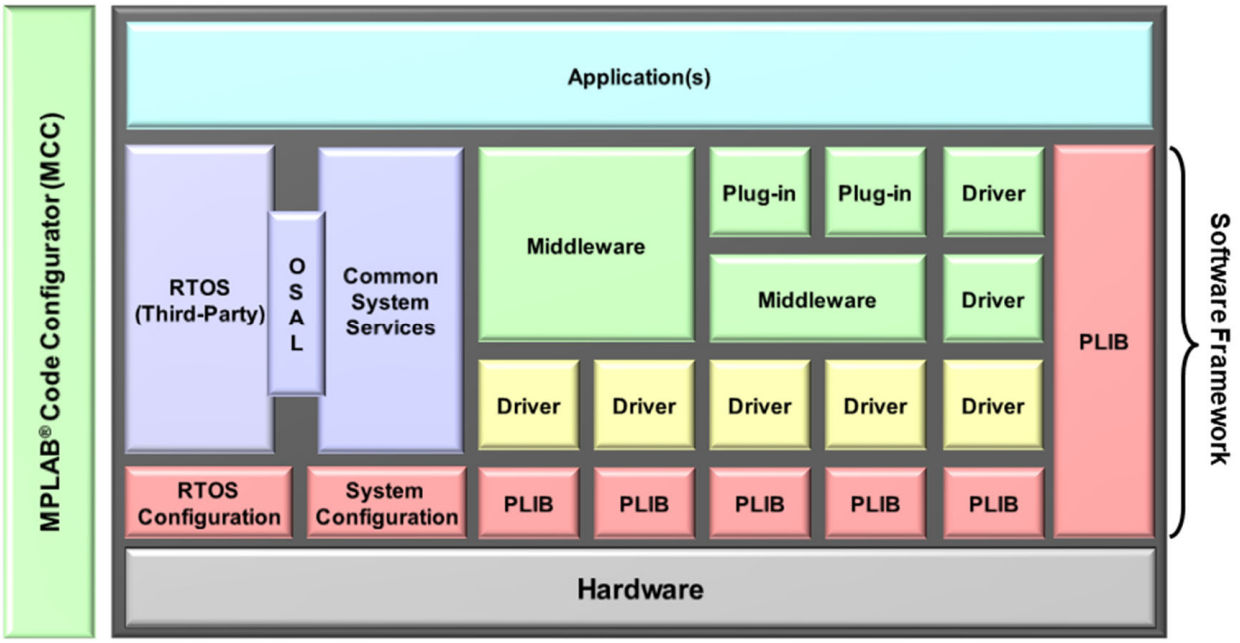
MPLAB Harmony v3 is a fully integrated embedded software development framework for 32-bit microcontrollers (MCUs) and microprocessors (MPUs).

The key features of MPLAB Harmony v3 are as follows:

- Unified software development framework for 32-bit PIC, SAM MCUs, and MPUs
- MCC GUI tool to ease configuration of hardware components, for example, Clock, Pin, Interrupt, ADC, DMA, Memory Protection Unit (MPU)
- Framework downloader to simplify the process of downloading the firmware libraries, demonstration applications, and extensions to the MCC GUI
- Simple, light weight, and user-friendly peripheral libraries (PLIBs)
- Modular and interoperable drivers and system services
- Rich set of middleware
- Support for RTOS and other third-party libraries
- Flexible firmware design models

MPLAB Harmony v3 has a layered architecture that consists of several software modules. These modules are portable, compatible to each other and they communicate and exchange data and information with each other. The figure below illustrates the MPLAB Harmony architecture.

Figure 2-1. MPLAB Harmony Architecture



- **PLIB:** This layer is the closest to the hardware with no abstraction and direct register access.
- **Driver:** This layer is the software peripheral drivers written on top of the PLIBs.
  Note: The term driver has a different meaning in MPLAB Harmony v3, ASFv3, and Atmel START. A MPLAB Harmony v3 driver is rich in nature and abstracted from the hardware, whereas ASFv3 driver has a little software handling and complete hardware access. Atmel START drivers are divided into HAL, HPL, and HRI layers.
- **System Configuration:** The system service manages shared system resources. These resources can be used by drivers, middleware or applications, such as Timer service, debug console, Clock, and DMA configurations.
- **Middleware:** Middleware typically consists of softwares that involves using protocols (sometimes related to IEEE® standards) such as, USB software stack, TCP/IP network stack, Graphics stack, Bluetooth, and other Wireless stacks
- **Third-party libraries:** Consists of few RTOS options, Graphics stack from Segger, Secure Socket Layer (SSL), Transport Layer Security (TLS), and Cryptographic libraries from WolfSSL Inc®.
- **Plugin:** Custom Bluetooth Profiles or USB function classes are available as plug-ins. For example, the USB protocol stack is a plug-in.
- **MCC:** A GUI tool used to configure the MPLAB Harmony v3 projects and generate codes. Users can configure the clock, pins, peripherals and so on to fit the needs of their application and generate the code. Then PLIBs and driver content might change according to the configuration. The code rendering or code generation is part of the MCC, and is used to maintain the drivers, and to avoid unwanted code.

2.2 Modular Downloads and Repository Structure

Microchip maintains a repository on GitHub that provides a catalog of MPLAB Harmony v3 repositories. The MCC Content Manager will read this catalog and download the library packages (and other content) selected by the user. The MCC clones them locally to make them available for including in the projects.

The distribution of the MPLAB Harmony v3 software framework over GitHub has advantages over the typical installer-based content delivery. Installer or zip file-based content delivery will install a full

set of framework source files every time the installer is run. GitHub will update only when changes made to each file, which simplifies content delivery and reduces the bandwidth required to install numerous files every time.

Note: Users must be aware which repositories are required for their application. Knowing the repository structure will help the user to understand their requirement.

The following are MPLAB Harmony v3 repositories:

- bootloader
- wolfssl
- wolfssh
- touch
- USB
- crypto
- core
- bsp (board support package)
- csp (chip support package)
- wolfMQTT
- MCC
- smartenergy
- motor_control
- gfx_apps
- gfx
- audio
- bt
- net
- micrium_ucos3
- CMSIS-FreeRTOS (From Arm GitHub)

The following repositories are required to start the MPLAB Harmony v3 framework.

- **Chip Support Package (csp) repository:** Contains peripheral libraries (PLIBs), and application demonstrations that use the PLIBs for all peripherals supported. The PLIBs are written such a way that it communicates with the hardware without any abstraction. The csp repository has the following structure:
  - **apps/:** This folder contains reference links for all peripherals across the devices supported by MPLAB Harmony v3. Each peripheral can have multiple PLIB demonstrations.
  - **peripheral/:** This folder has PLIB for almost all the peripherals in all the devices supported in MPLAB Harmony v3.
  - **arch/:** This folder has software logic for device support based on the MCU architecture. MPLAB Harmony v3 supports Arm Cortex M0+, M4F, M7, M23, M33, Cortex A5, and MIPS architecture-based devices.
  - **doc/ or docs/:** These folders have the help documents for PLIBs and PLIB demonstrations.

Tip: In the `csp_apps_pic32/sam` repository and the apps folder, the folder names are the same as the peripheral names in the data sheet. For example, general purpose I/O ports are named differently in different device families. Some devices have an I/O peripheral named GPIO, and in other devices it is PIO or PORT. Check the device data sheet and navigate to the folder according to the peripheral naming in the data sheet. This will help identifying the right demonstration for a peripheral.

- **Core repository:** This repository contains drivers and services with user-friendly abstractions of peripherals and shared resources based on which middleware and applications are developed. Drivers and services provide advanced capabilities, such as buffer queuing and peripheral sharing. The core repository also provides an Operating System Abstraction Layer (OSAL). MPLAB Harmony v3 supports multiple third party real-time operating systems (RTOS). MPLAB Harmony drivers, services, and middleware use RTOS-specific methods to ensure a *thread safe* operation and RTOS compatibility. To use RTOS-specific methods, MPLAB Harmony libraries have to call different functions for each supported RTOS. To do this, MPLAB Harmony v3 defines an Operating System Abstraction Layer (OSAL). The OSAL is a lightweight OS compatibility layer that abstracts away the RTOS-specific details and provides a consistent set of functions that the MPLAB Harmony libraries may call to obtain the RTOS-specific capabilities they need to protect shared resources.
  The core repository has the following structure:
  - **apps/:** This folder contains reference links for core components, such as drivers (I^2C, SPI, USART, etc.,), file systems, FreeRTOS, and the system timer service.
  - **driver/:** This folder has MPLAB Harmony v3 drivers for I^2C, USART, SPI, SDMMC, SDSPI, Memory. These drivers are built on top of PLIBs.
  - **osal/:** This folder has a wrapper layer on top of the RTOS that are supported.
  - **system/:** This folder has system services, such as DMA, Virtual File system, Interrupts, I/O Ports, Debug console, command console, Timers, and Random number.
  - **doc/ or docs/:** These folders have the help documents for core components and demonstrations.
- **Board Support Package (bsp) repository:** The bsp repository contains board specific information. This includes I/O pin details, such as LEDs and switches on the board connected to the I/Os.

2.3 Flexible Firmware Design Models

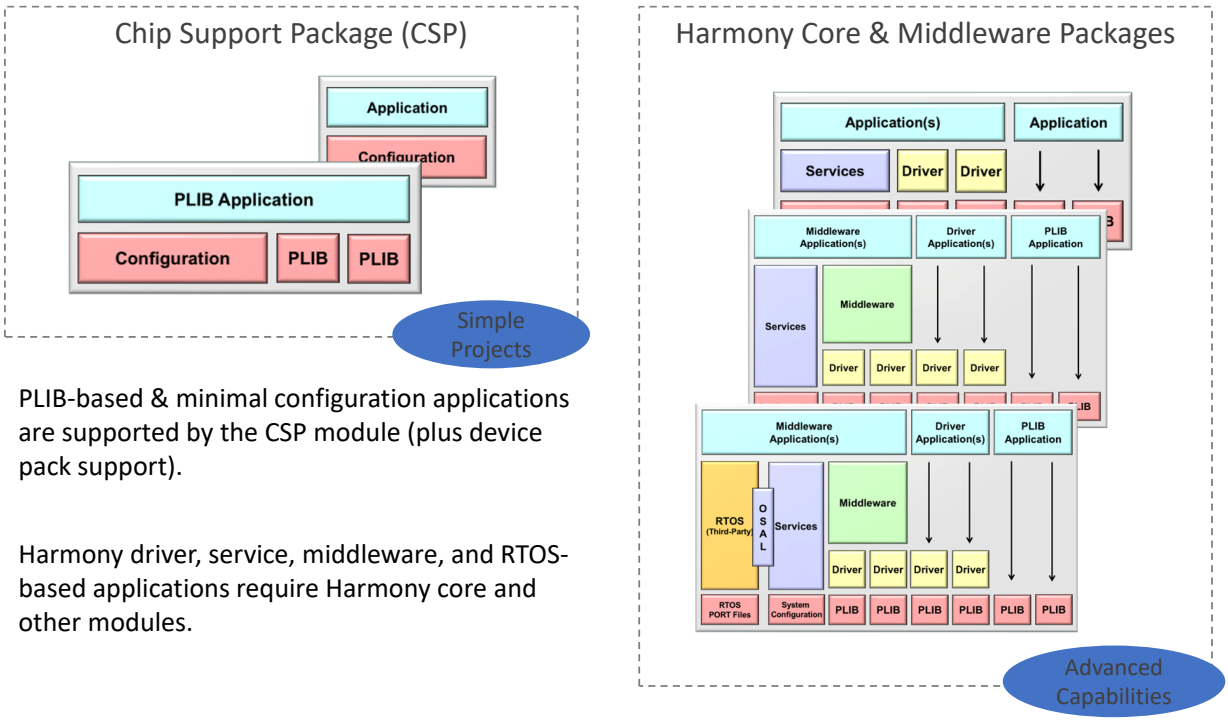
The MPLAB Harmony v3 architecture allows implementing a variety of applications from small real-time applications to larger feature rich applications.

The following design models are supported by MPLAB Harmony v3:

- Simple device configuration and initialization using MCC
- Peripheral library-based projects
- Powerful, conflict-free driver-based projects
- Projects requiring MPLAB Harmony middleware
- RTOS-based projects for optimum Central Processing Unit (CPU) utilization

MPLAB Harmony v3 libraries are provided in packages, and maintained in GIT repositories on GitHub. The MPLAB Harmony v3 Chip Support Package (CSP) supports simple device initialization and PLIB-based projects, relying only on the appropriate Device Family Pack (DFP), and standard 'C' libraries and tools. It has no external module or component dependencies. Advanced projects that require abstraction from the hardware, operating system support, or powerful middleware capabilities will rely on the MPLAB Harmony v3 core package.

Figure 2-2. MPLAB Harmony v3 Libraries



2.4 Advanced Software Framework v3 and Atmel START Architecture

Atmel START (ASFv3): The Advanced Software Framework (ASF) provides software drivers and libraries to build applications for Atmel megaAVR®, AVR XMEGA®, AVR UC3 and SAM devices. ASF is designed to help develop and bring together the different components of software design.

The ASF folder is divided into six sub-folders based on family of devices. The following list displays the contents of the ASF root folder:

- avr32/
- common/
- mega/
- sam/
- thirdparty/
- xmega/

Each architecture folder and the common directory are divided into several sub-directories, these directories contain various modules, boards, drivers, components, services and utilities. The following sub directories provides an overview of how the various modules are wired together.

- applications/
- boards/
- components/
- drivers/
- services/
- utils/

Atmel START (ASFv4): This is the next generation of ASF built to work as a web-based user interface with GUI for the ease of configuration and usability. ASFv4 represents a complete redesign and implementation of the whole framework when compared with ASFv3. The redesign is made by making a significant change in architecture. The drivers are made as use-case drivers and the folder structure is modified to fit this new architecture. The driver layer is divided into Hardware Abstraction Layer (HAL), Hardware Proxy layer (HPL), and Hardware Register Interface (HRI) layer. HAL remains common across devices, and HPL and HRI vary according to the hardware used.

Figure 2-3. MPLAB Harmony v3, ASF3, ASF4 and Atmel START Architecture Overview

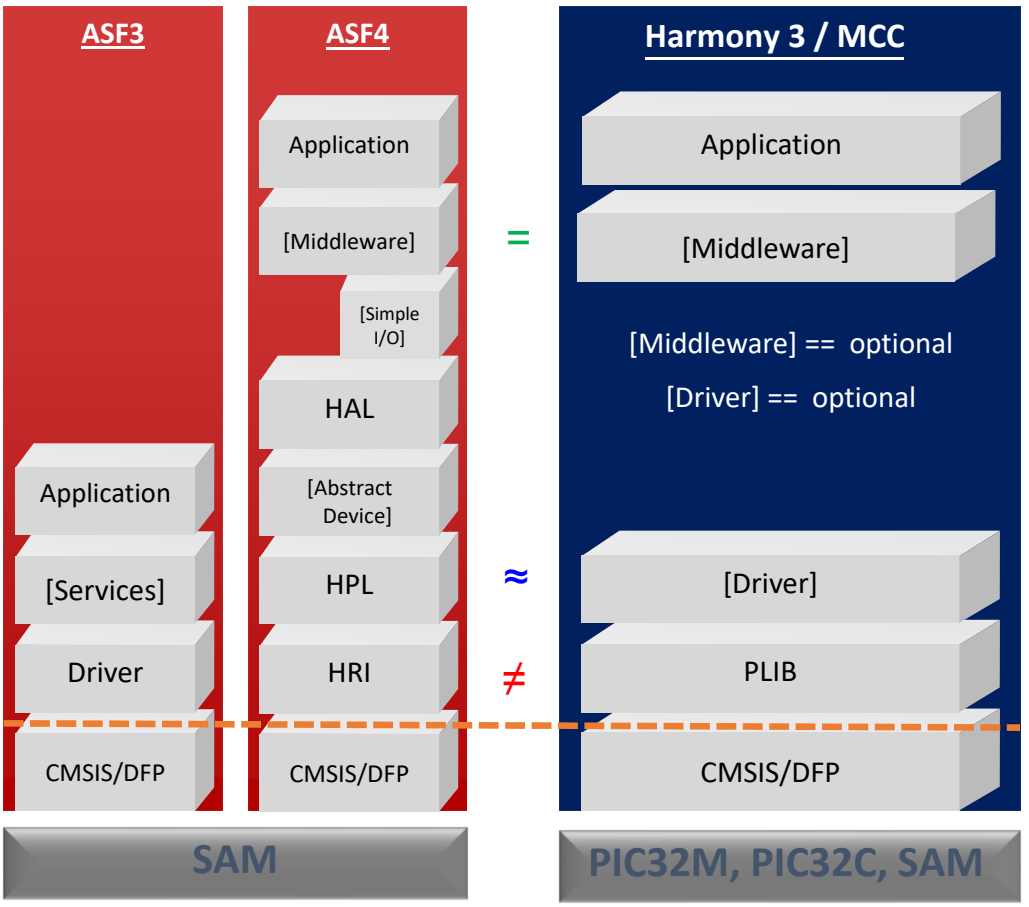


Table 2-1. MPLAB Harmony v3, ASFv3 and Atmel START Architecture Information

	Harmony v3	Atmel START	ASF v3
Configuration specific code generation	Yes	Yes	No
Can get Standalone project	Yes	No (Uses native header files from Atmel Studio)	Yes
GUI Aided Tool	Yes	Yes	No
Other Toolchain support	Yes (IAR)	Yes (IAR, Keil makefile)	Yes (IAR)
Modular download	Yes (GitHub-based)	Yes (Individual project)	No
Cloud based Tool	No	Yes	No
Integrated with IDE	Yes (MPLAB)	Yes (Atmel Studio)	Yes (Atmel Studio)
Framework available as standalone zip	No	No	Yes

.....continued

	Harmony v3	Atmel START	ASF v3
Content Distribution	GitHub repository (to be cloned locally)	Cloud	Part of Atmel Studio Installation
Peripheral Driver model	Peripheral Configuration specific (plibs)	Use-case specific	Full generic driver
Business logic available	Yes	No	No
Native Compiler support	XC32 compiler	GCC	GCC

2.5 Significance of Peripheral Libraries (PLIBs) in MPLAB Harmony v3

This is the lowest level interface to the peripherals of a device. This is named as a library, but it consists of source (`.c`) and header (`.h`) files. Typically, each peripheral instance has a source and header file associated with it. The Application Program Interface (API) implementation is kept simple, such that the most of it will be direct peripheral register access.

Most API names and structures are consistent across devices which helps in driver portability. Typically, driver layers use PLIB API calls. Therefore, PLIBs are closely associated with peripherals, and these vary from device-to-device. Most PLIBs can be used either in polled mode or in interrupt mode. To realize polled mode, PLIBs have blocking calls. Interrupt mode supports ISR functionality with call-back support. There is no error checking, sanity checks, or state maintenance in a PLIB. The PLIB has clean code and no conditional macros (`#if, #elif.`) for readability.

Table 2-2. Example of Basic PLIB APIs Availability

Usage	UART/SERCOM_USART	SPI/SERCOM_SPI	TWIHS/SERCOM_I2C
Initialization	UARTx_Initialize()	SPIx_Initialize()	TWIHSx_Initialize()
Transaction	UARTx_Write()	SPIx_Write()	TWIHSx_Write()
	UARTx_Read()	SPIx_Read()	TWIHSx_Read()
	-	SPIx_WriteRead()	TWIHSx_WriteRead()
Status	UARTx_ReadIsBusy()	SPIx_IsBusy()	TWIHSx_IsBusy()
	UARTx_WriteIsBusy()		

2.6 Advantages of MPLAB Harmony Drivers

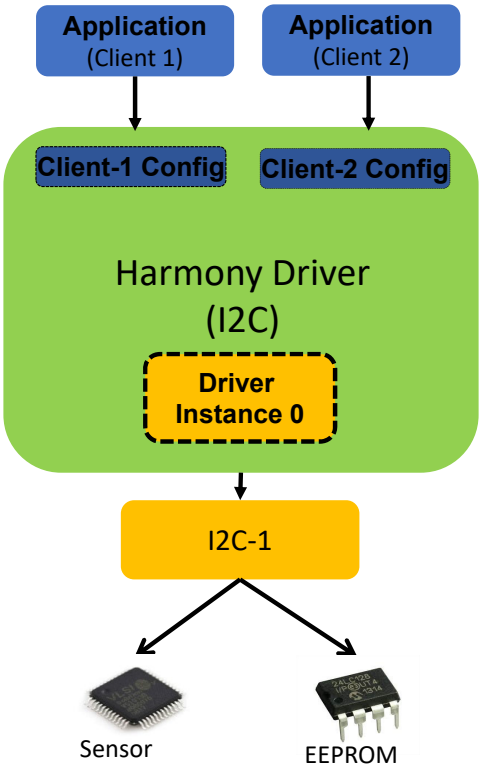
The MPLAB Harmony drivers provide an abstracted interface to access the hardware. This enabless portability of the applications and middleware across platforms. This is the preferred method to access peripherals when the software needs to be ported across hardware platforms.

When two different clients in an application access the same instance of a peripheral, the driver must be capable of accessing the correct sensor to give data to the correct client. The MPLAB Harmony v3 driver is capable of handling this multi-client scenario.

The following are key features of the MPLAB Harmony drivers:

Multi-Client Operations

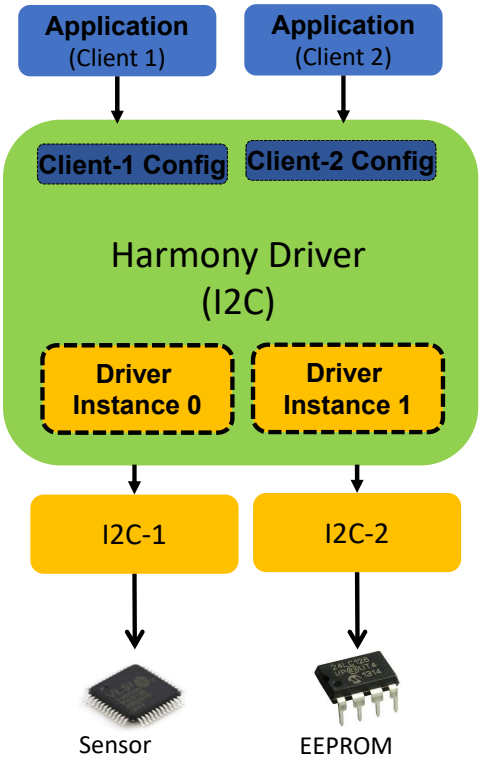
Figure 2-4. Multi-Client Operations



Multi-Instance Operation

In a Multi-Instance scenario, the same driver (code) is capable of handling many instances of a peripheral. Application clients can call the same driver function to perform the job with the respective peripheral instance.

Figure 2-5. Multi-Instance Operation



- Same API for all the instances of a peripheral:
  - Application remains same when the peripheral instance is changed

I²C Driver API	TWISH/SERCOM I2C PLIB API
DRV_I2C_WriteTransferAdd(...)	SERCOMx_I2C1_Write(...)
	SERCOMx_I2C2_Write(...)
DRV_I2C_ReadTransferAdd(...)	SERCOMx_I2C1_Read(...)
	SERCOMx_I2C2_Read(...)

- Support for DMA Transfer mode - Driver Interface or API remains same with or without DMA Transfer mode
- Buffer/Queue Support
- Error checking
- Sanity checks
- RTOS ready
- Thread safe

Drivers will not use blocking APIs. Both sync and async driver types are available in MPLAB Harmony v3.

MPLAB Harmony v3 Drivers:

There are two MPLAB Harmony v3 drivers, Asynchronous (async) and Synchronous (sync). The driver details are as follows:

- Asynchronous (async)
  - Non-blocking APIs

  - Works seamlessly in bare-metal and RTOS environment
  - Interrupt and Thread safe
- Synchronous (sync)
  - Blocking APIs
  - Suitable for use in RTOS environment
  - Interrupt and Thread safe

Note: The term sync and async in MPLAB Harmony v3 have a different meaning than in Atmel START. In Atmel START, available driver types are sync, async and RTOS. sync means polled driver, async means interrupt based, and RTOS driver is an RTOS compliant driver.

2.7 Middleware Support

MPLAB Harmony v3 middleware provides sophisticated software libraries that support key technologies, such as industry-leading graphics, TCP/IP networking with broad protocol support, USB Host and Device capabilities for the most commonly used device classes, cryptographic capabilities, and so on. These software libraries are built upon MPLAB Harmony v3 drivers and services, therefore the middleware works with all supported Microchip 32-bit devices, while still allowing direct access for peripherals that are not used by the MPLAB Harmony libraries.

Note: Supported middleware details are provided in the section References.

2.8 Operating Systems Abstraction Layer

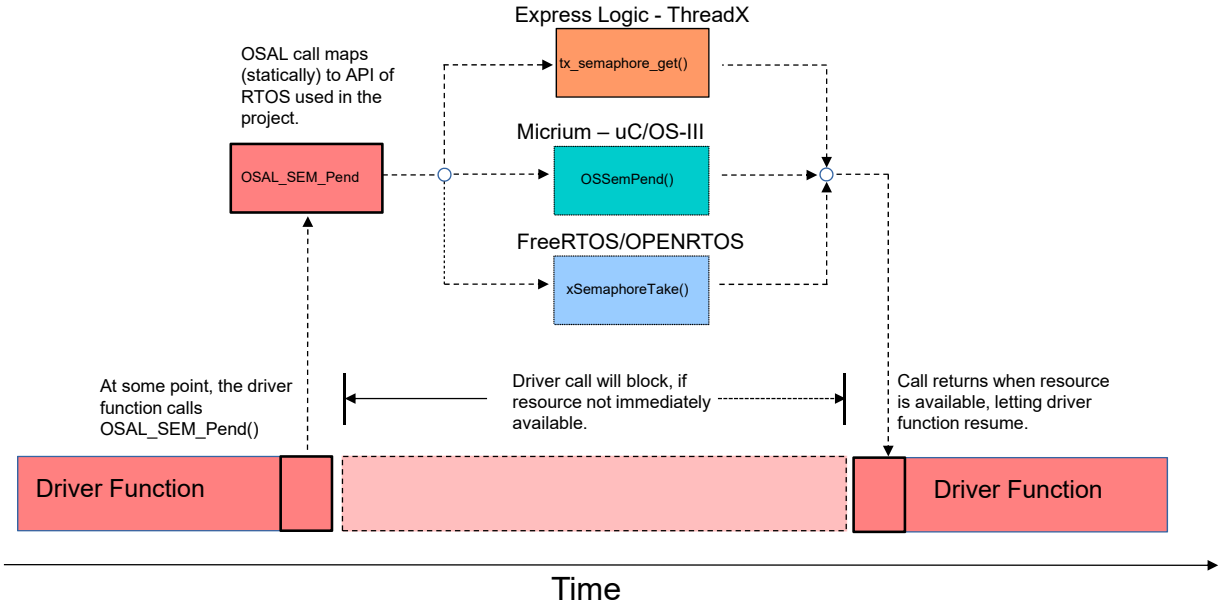
MPLAB Harmony v3 supports multiple RTOS. Each RTOS has its own API call to create tasks, run the scheduler, semaphore, or mutex handling. To enable the compatibility of MPLAB Harmony v3 drivers in a non-RTOS environment (Bare-Metal) and across various RTOSs, MPLAB Harmony v3 provides an abstraction layer known as Operating Systems Abstraction Layer (OSAL). OSAL supports a minimal feature set that was abstracted from many different RTOS products for maximum compatibility. The focus of the OSAL is providing the minimum set of mechanisms necessary for safe operation of libraries in a multi threaded environment.

Primary OSAL Features

The OSAL features are as follows:

- **Mutexes:** Used to manage the ownership of resources, therefore can only be accessed exclusively by a single thread at a time
- **Semaphores:** Primarily used to synchronize between different threads
- **Critical sections:** Used to protect a section of code, thus it must be executed atomically (uninterrupted or indivisibly)
- **Memory allocation:** Used to support RTOS-specific memory management (Abstracts `malloc` and `free` if no RTOS-specific memory management)
- **Initialization and Information:** OSAL support and OS information

Figure 2-6. OSAL and RTOS Mapping



3. MPLAB Harmony v3 GitHub Repository Handling

MPLAB Harmony v3 must be used with MPLAB X IDE. After installing MPLAB X IDE, the MCC Content Manager and MPLAB Code Configurator plugin are available to use.

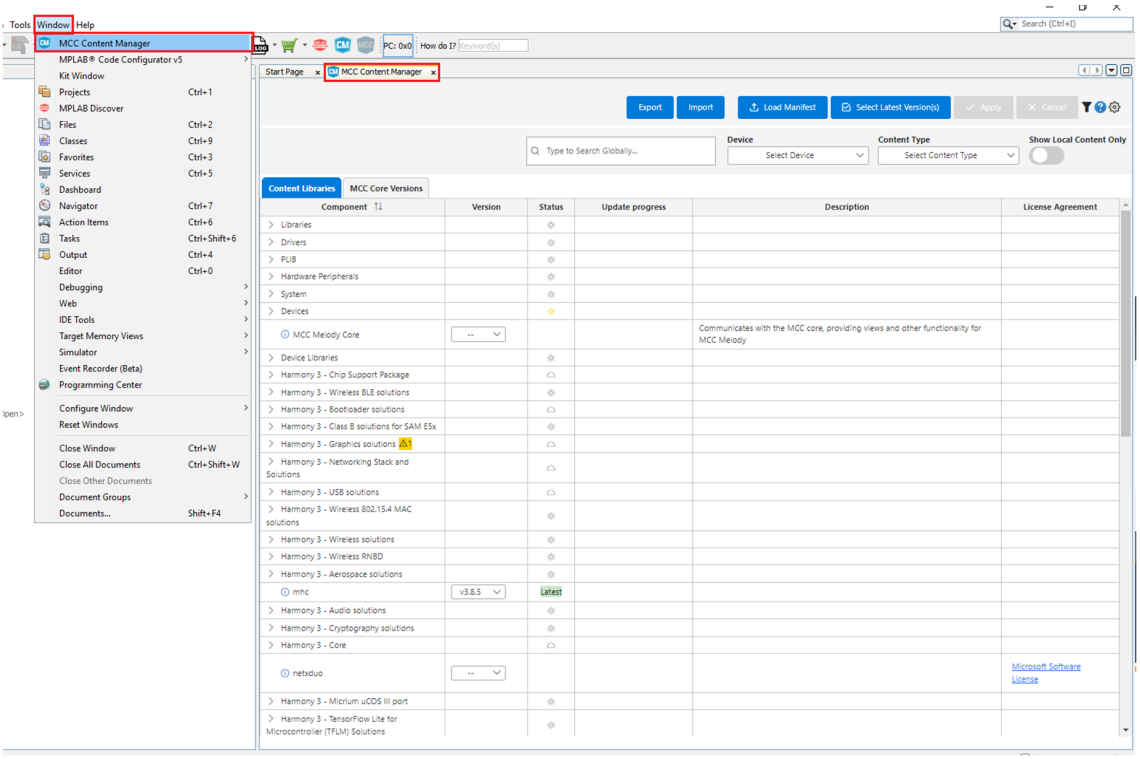
A significant change is that MPLAB Harmony v3 content is now delivered through GitHub. The MCC Content Manager helps in cloning the public GitHub repositories to a PC. The user can select which repositories to be cloned. Once the repositories are cloned, the user can update them regularly to keep the content synchronized with the master branch on the cloud.

3.1 Downloading Packages Using the MCC Content Manager

Using the MCC Content Manager, users can download packages in the MPLAB X IDE. Follow these steps to download packages:

1. Launch MPLAB X IDE.
2. Select *Window > MCC Content Manager.* The MCC Content Manager Window will be displayed.
   a. **MCC Content Manager:** Helps in downloading and updating MPLAB Harmony repositories from Remote GitHub. Users can download selected repositories from GitHub.

Figure 3-1. MCC Content Manager



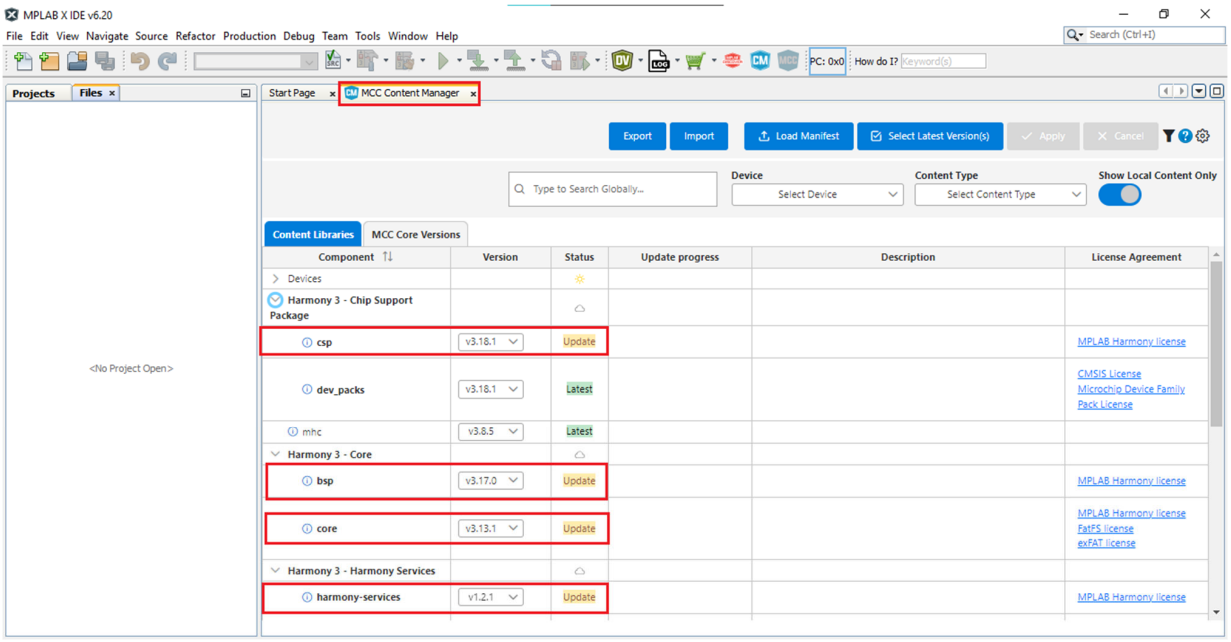
3. While opening the MCC Content Manager, specify a path to clone framework repositories.
   Note: First time, MCC Content Manager will list all the available repositories in the public GitHub link. The user can select the required repositories and then click Download to clone the selected repositories to the PC. Thereafter, the MCC Content Manager will list if any new repositories added by Microchip. If needed, the user can select the newly added repositories to download.

3.2 Updating the Repositories Using the MCC Content Manager

If a repository is downloaded, the MCC Content Manager will query the tag information of the branch on the PC and what is available on the Web. If the Web has a newer version, the user can update the appropriate repository.

If the user has modified a repository, such as changing a configuration, then the user can clean the repository using the MCC Content Manager and update the repository.

Figure 3-2. Updating Repositories



Tip: As internet link strength and server response speed vary, therefore this step might take some time.

4. Integration Between MPLAB X IDE and MPLAB Harmony v3

The MPLAB Code Configurator will launch the MPLAB Harmony v3 framework development environment.

4.1 MPLAB Harmony v3 User Interface

One of the significant features of MPLAB Harmony v3 is the graphical user interface (GUI) which helps to configure the MCU features easily. The following list provides examples of some of the GUI windows under the MCC GUI:

- Project Graph
- Clock Configuration
- Pin Configuration
- NVIC Configuration
- DMA Configuration
- ADC Configuration
- Touch Configuration
- Qspin Motor Control Configuration

Note: The MPLAB Code Configurator is based on the device family features.

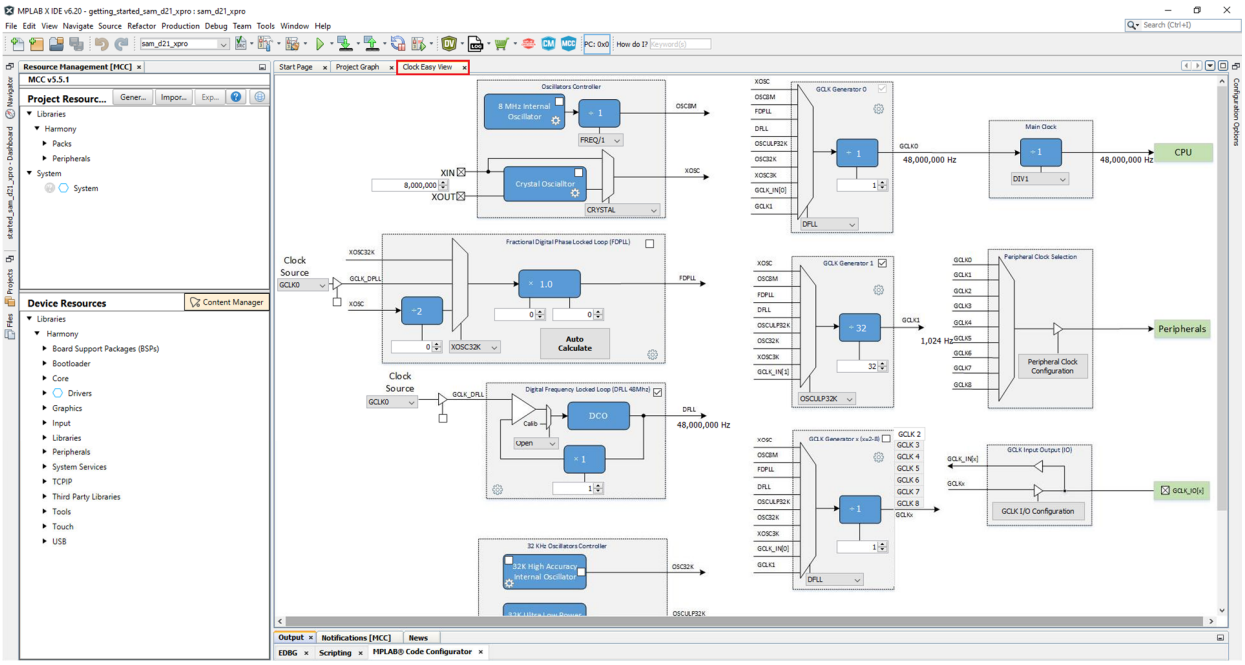
The Project Graph shows all the modules used in the project, and their connection with other modules to give a visual representation of dependency between the modules. Similarly, other GUI windows are very helpful in configuring the corresponding modules.

Configuring these modules is possible without using these easy-to-use GUIs. This can be done by using the other basic UI configuration tree. When using the UI configuration trees, the user needs to pay more attention to understand the hardware and configure it correctly in the tree view.

For example, many Microchip Arm devices have lot of flexibility in clock configurations. If the user had to configure the clock without the Clock Configuration, the user should fully understand the clock distribution of the MCU, and configure the clock using the UI configuration tree.

The Clock Configuration GUI in the MCC, helps by visually showing the clock tree, status and values on the UI which is similar to the clock distribution diagram available in the data sheet. Therefore, the user can configure them easily. The Clock Configuration shows the CPU, other clock values, and the Clock mask status. If there is any change in clock masks, DPLL ratio, or prescalars, the Clock Configuration updates the corresponding values for the clock and status accordingly and shows the current values.

Figure 4-1. Clock Configuration (Project Graph > Plugins > Clock Configuration)



Other GUIs, such as the Pin Configuration, NVIC Configuration, and ADC Configuration ease the configuration.

Note: Atmel START has similar GUIs for the clock, pinmux, and Qtouch. Users familiar with the GUIs in Atmel START will find the MPLAB Harmony v3 GUIs user friendly. In addition, MPLAB Harmony v3 has more GUIs, such as the DMA, NVIC, and Event Configuration to ease the configuration of a peripheral.

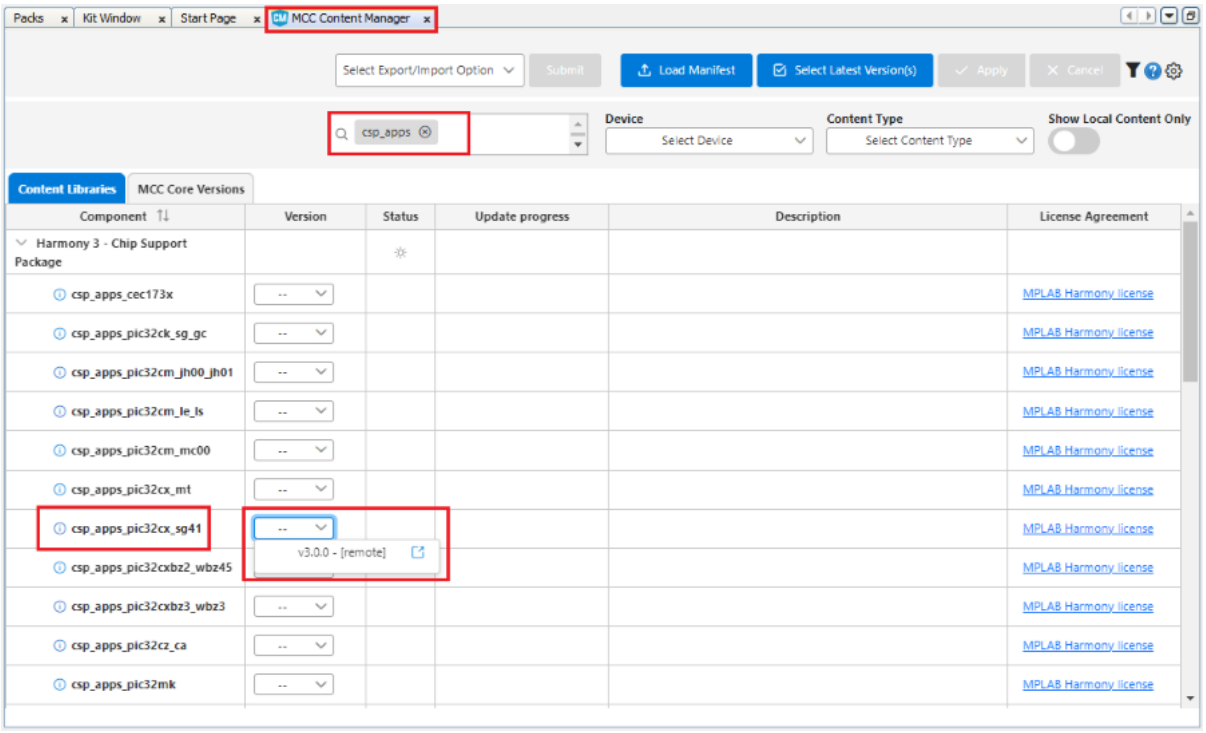
4.2 MPLAB Harmony v3 – Exploring a PLIB Demonstration Project

After installing MPLAB X IDE and MPLAB Code Configurator, the user can open any one of the PLIB or core demonstration example MPLAB Harmony v3 projects.

To start with a simple project, an example PLIB demonstration project is explored. This sub-section will help the user to open a PLIB demonstration, use the MCC, and explore project windows to gain familiarity with the programs.

1. Launch MPLAB X IDE.
2. From the toolbar, select *File > Open Project*.
3. Go to the path where the `csp` repository is downloaded.
4. Go to one of the `csp_apps`.
   Note: In MPLAB X IDE, use the MCC Content Manager to download the `csp_apps` repository for the respective device, along with the corresponding version of the selected packages.

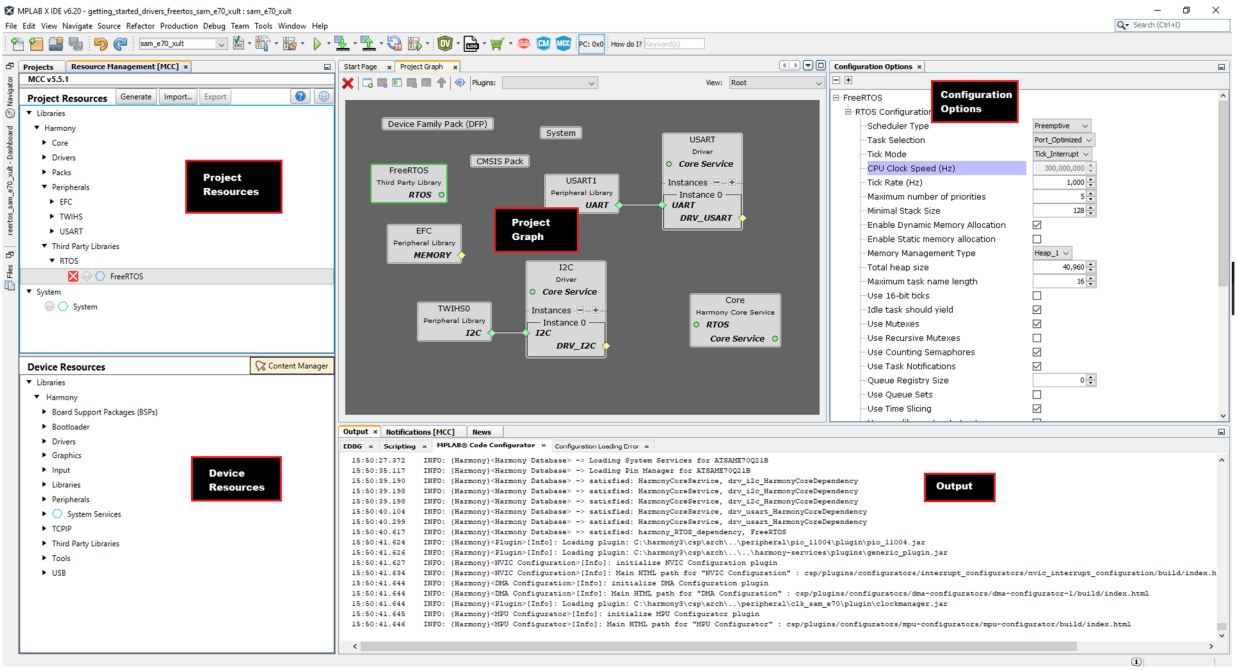
Figure 4-2. MCC Content Manager



5. In the `apps` folder, select an example peripheral from the following location: *port* > *port_led_on_off_polling* > *firmware* > *select project* for any board of user's choice.

After opening an MPLAB Harmony v3 project, launch MCC by going to *Tools* > *Embedded* > *MPLAB Code Configurator v5: Open/Close.* The following figure shows the MPLAB Harmony v3 project opened along with the MPLAB Code Configurator in MPLAB X IDE.

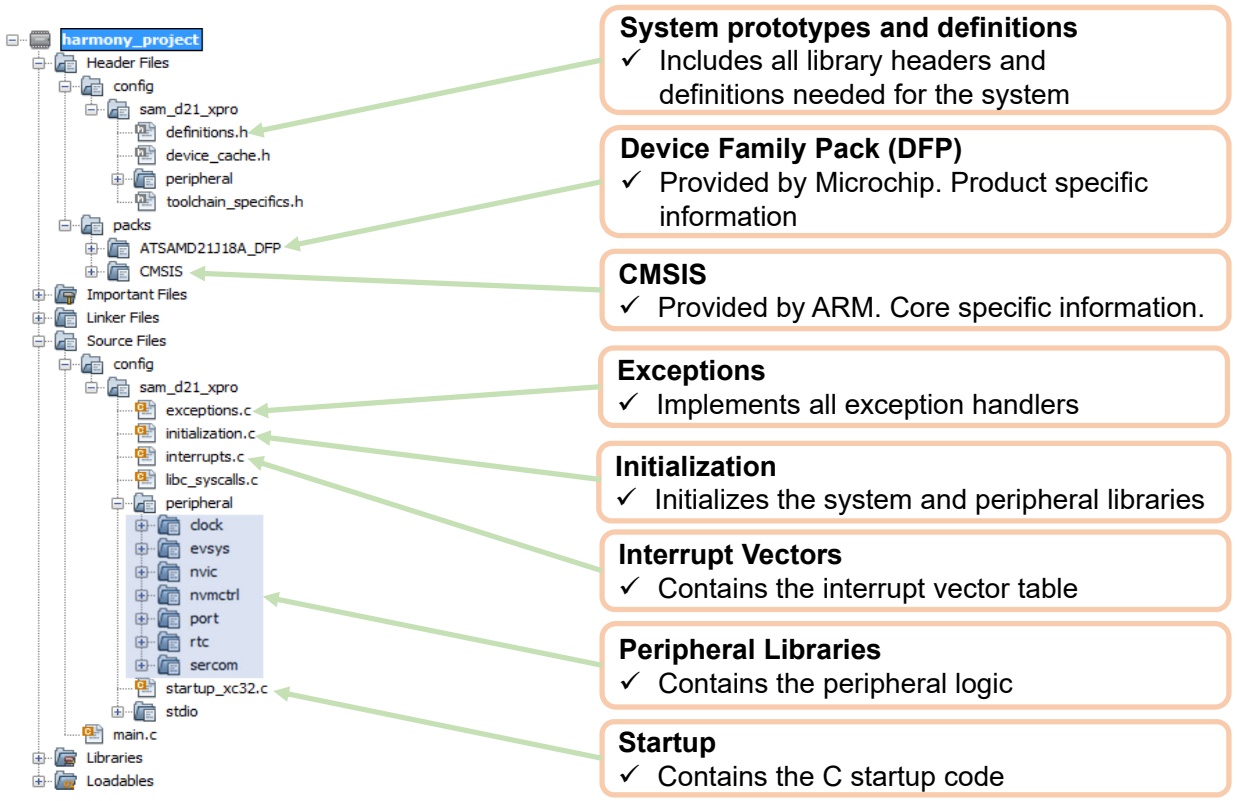
Figure 4-3. Project Window When MCC Opens



4.3 MPLAB Harmony v3 Project Folder Structure

The following figure illustrates the structure of a PLIB project in MPLAB Harmony v3.

Figure 4-4. MPLAB Harmony v3 PLIB Project Folder Structure

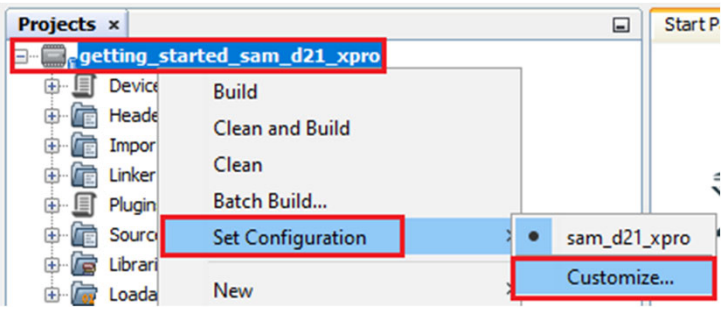


4.4 Compiling, Programming, and Debugging PLIB Demonstration Project

The opened PLIB demonstration project can be compiled and programmed as is. All PLIB demonstrations available in the repository must be compiled as is, with no need for regenerating the project. To compile and program a PLIB demonstration, the user must select the DFP, compiler toolchain, and hardware tool (the tool must be connected to the PC) to be used with this project.

1. Right-click on the project, and then select *Set Configuration > Customize...*.

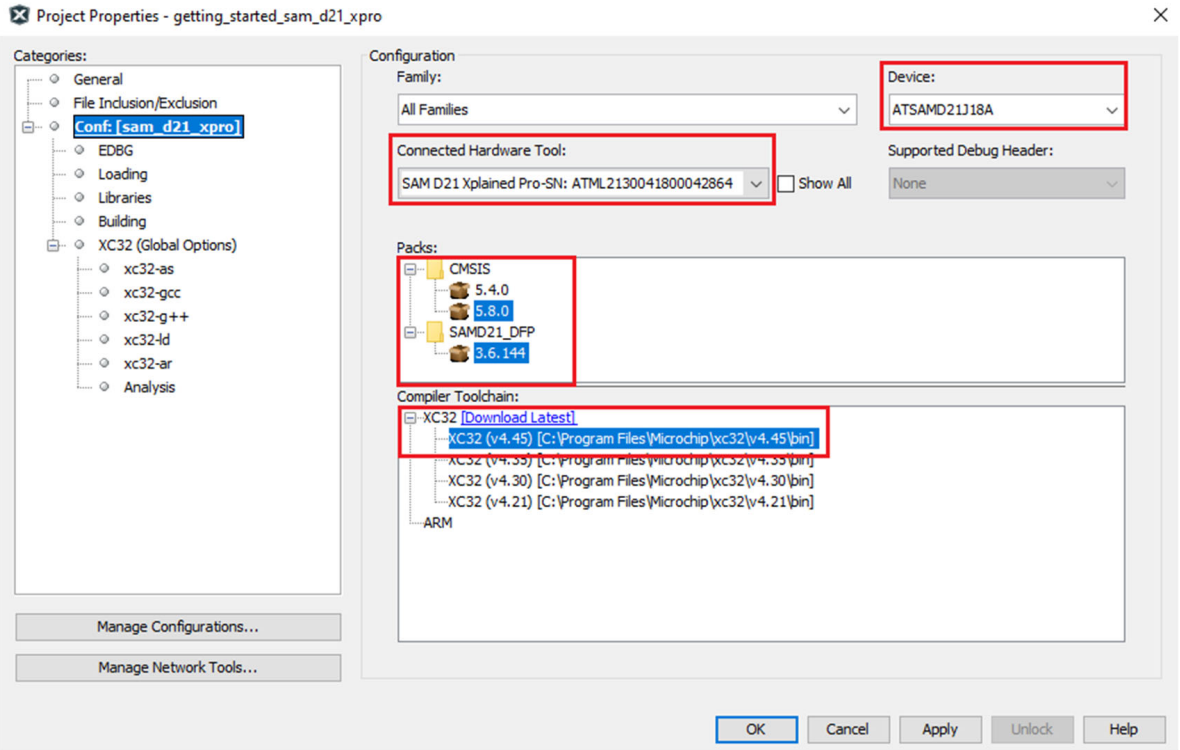
Figure 4-5. Set Configuration



2. In the Project Properties window, under Categories select **Conf: [sam_d21_xpro].**
3. In the right Configuration property page, select the DFP, Hardware tool, and compiler toolchain version (if multiple versions are installed) to compile and then program the project in the device.

4. Click **OK**.

Figure 4-6. Project Properties

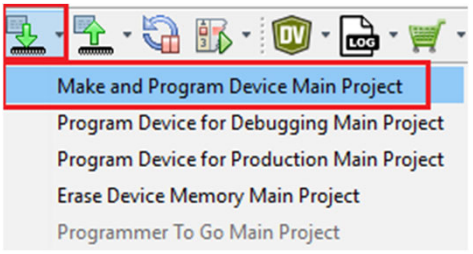


The following options can be changed in the Project Properties window:

- General File Inclusion, File Exclusion, MACRO addition
- XC32 compiler settings, such as optimization, preprocessing settings, message settings, and additional optional compiler switches
- General linker settings, linker symbols and macros and linker switches
- XC32 assembler options
- Debugger program options, memories to program, debug communication interface speed

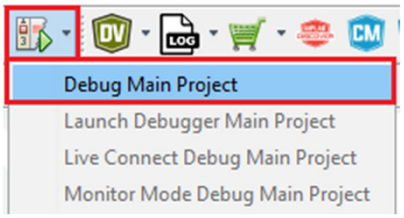
5. Select **Make and Program Device Main Project** from the drop-down list to compile the project and program the device with compiler code.

Figure 4-7. Make and Program Device



6. To start the debug session, from MPLAB X IDE, select **Debug Main Project** from the drop-down list, or click on the Debug Main Project icon. This enables the user to debug the project.

Figure 4-8. Debug Main Project

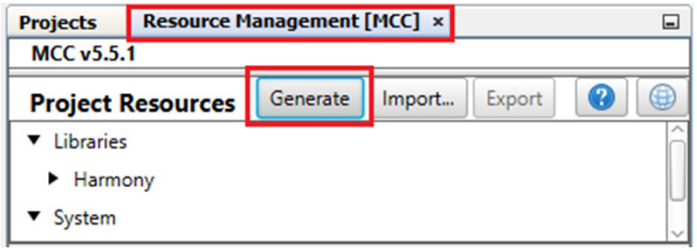


4.5 Modifying Components, Configurations and Regenerating Code for the PLIB Demonstration

To modify or change a peripheral in the Project Graph window, the user must open the MPLAB Code Configurator first. Once the MCC is opened, the user can add or remove any peripheral to the Project Graph. In the Tree view, change the configuration of the included peripherals per the needs of the application and regenerate the code.

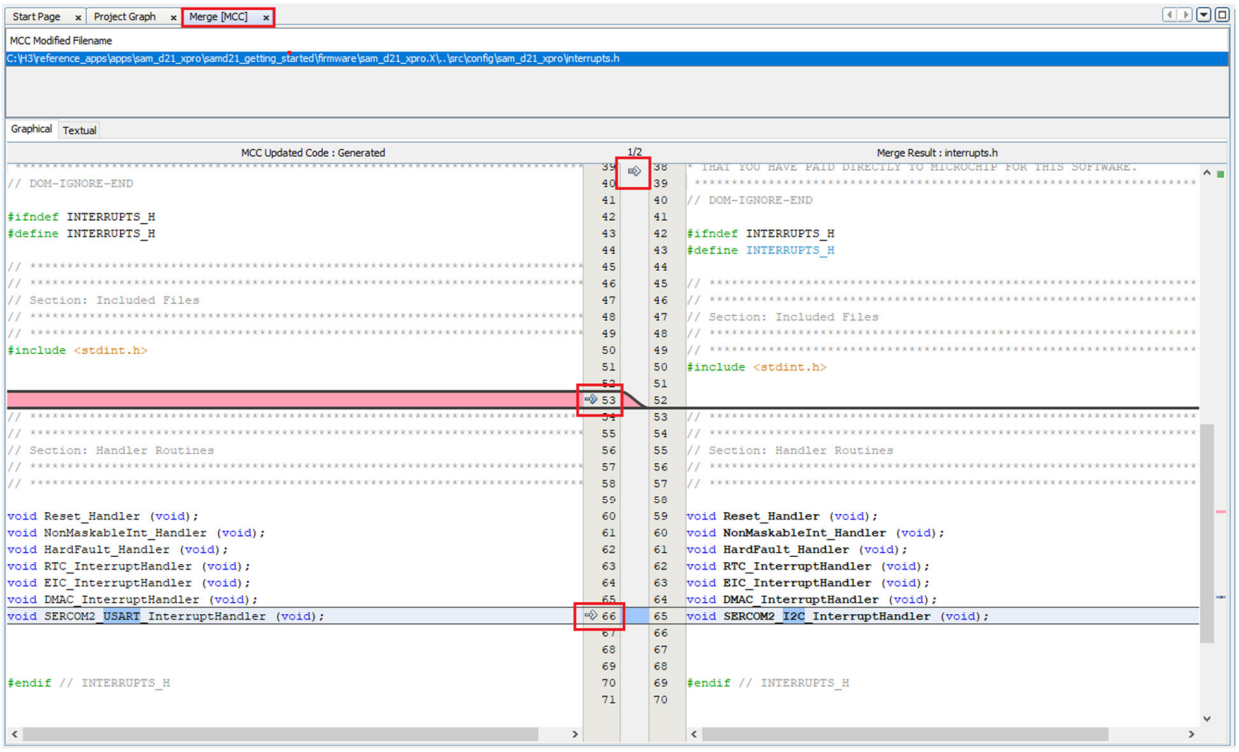
To regenerate the code for the new configuration, go to *Resource Management [MCC] > Project Resources* and then click **Generate.**

Figure 4-9. Generate Code



Before regenerating the new code, the MCC shows the files with the changes that are going to be done (line wise) as the result of the re-configuration. The user can accept or reject the changes (line wise) based on the result.

Figure 4-10. Line Wise Change



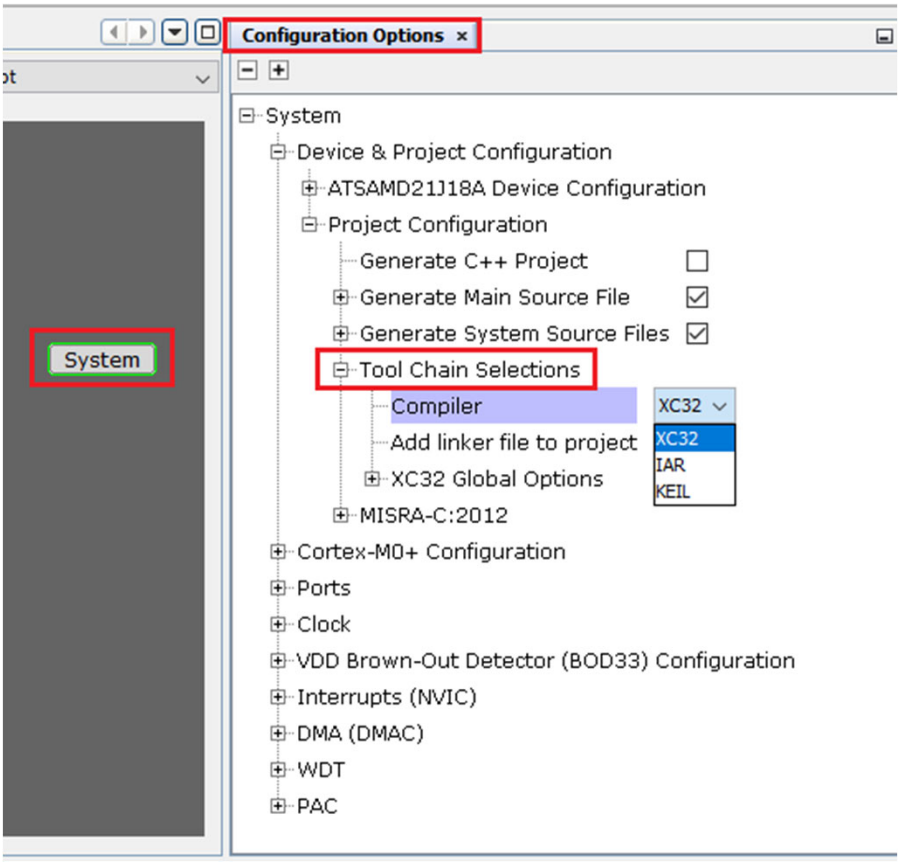
Note: This option is useful to view the changes made by the user and MCC to each file.

4.6 MPLAB Harmony v3 Services - Standard I/O (STDIO), Toolchain Supports

MPLAB Harmony v3 has support for Standard I/O (STDIO). Users can use the standard `printf` and `scanf` through the USART module in the hardware using STDIO module.

Similarly, users who are using IAR IDE, can make use of generating the project for the IAR toolchain. In *Project Graph > System module* the device and project configuration settings for the toolchain are available. Currently XC32, IAR, and KEIL toolchains are available.

Figure 4-11. Toolchain Selection



When selecting the IAR toolchain, the `ipcf` file will be generated along with the project files.

Note: Users familiar with the IAR IDE support in Atmel START will find this easy to use, because the Atmel START also uses same `ipcf` file for extending the support to the IAR IDE. Ipcf is a project connection file which is the standard method used by IAR to generate the project from other platforms.

5. Example Applications

5.1 Example Demonstrations Available in MPLAB Harmony v3

The apps folder in the `csp_apps_pic32/sam` and `core_apps_pic32/sam` repositories contains example applications for peripherals. The `apps` folder is organized by the peripheral. Each peripheral has a folder associated with it, and all examples for the peripheral are available within the folder. Each peripheral may have multiple example projects to explore.

The PLIB-based example projects are available in the `csp_apps_pic32/sam` repository. The following path describes the location of the PLIB example projects.

RTC example: *csp_apps_pic32cm_le_ls > apps > rtc > rtc_periodic_timeout > firmware > pic32cm_le00_curiosity_pro.X* for the appropriate board.

SPI example: *csp_apps_sam_d21_da1 > apps > sercom>spi > master > spi_eeprom_write_read > firmware* for the appropriate board.

The DRIVE-based example projects are available in the `core_apps_pic32/sam` repository. Specific example projects for `async` and `sync` driver types are also available. The following path describes the location of the DRIVER example projects.

I²C example: *core_apps_sam_d21_da1 > apps > driver > i2c > async > i2c_eeprom > firmware* for the appropriate board.

USART example: *core_apps_sam_e70 > core_apps_sam_e70_s70_v70_v71 > apps > driver > usart > sync > usart_echo > firmware* for the appropriate board.

Note: ASFv3 contains example projects for the peripherals that are similar to MPLAB Harmony v3. Atmel START is a web-based tool and example projects are listed separately in it. Users can downloaded example projects from the Microchip website: www.microchip.com/.

5.2 Creating a Simple Application Demonstration Using MPLAB Harmony v3

This section provides a brief overview on creating a simple PLIB demonstration. For step-by-step migration details, refer to these documents :

- ASF3 to MPLAB Harmony 3 Migration Guide
- Atmel START to MPLAB Harmony 3 Migration Guide

The following steps contain high-level information for creating an application demonstration in MPLAB Harmony v3:

1. Launch MPLAB X IDE, and then select *File > New Project*.
2. Choose Project: Under Categories, select **Microchip Embedded** and for Projects, select **Application Project(s)**.
3. Select Device: select the right device to be used and Tools.
4. Select Compiler: Choose the compiler version.
5. Configuration database setup: Select the MPLAB Harmony repositories to be loaded for the project.
6. The MPLAB Harmony project will be opened along with the MPLAB Code Configurator. By default, the project graph will be opened in which a few modules are added to the Project.
7. Add, connect, and configure the modules. The user can add peripherals required for the application. Connect them in the project graph and configure them in the UI tree per the application requirement.
8. After adding and configuring the modules, regenerate the code along with main and application source files.

9. The user can write application logic and implement the application functionality.

6. Appendix

6.1 Establishing Connection with Target Hardware Using MPLAB X IDE

In Atmel Studio, one of the most used windows is Device Programming. In MPLAB X IDE, these functions are done using another application called MPLAB Integrated Programming Environment (IPE). This application is installed along with MPLAB X IDE. When needed, this application can be launched and used to ensure that the connection between the target and the debugger is established correctly.

Users can use this IPE to erase the chip, read or write the HEX files to the chip, and verify the content with application image available.

Fuse programming: In Atmel Studio, fuses are programmed separately but this is handled differently in MPLAB X IDE. MPLAB X IDE allows the user to view and generate the code (for fuse configuration) which the user can copy in the source file. MPLAB X IDE programs both the application image and the fuses at the same time.

Launch *MPLAB X IDE > Windows > Target Memory Views > Configuration Bits*.

6.2 Basic Understanding of GIT

MPLAB Harmony v3 contents are delivered through GitHub. If users are new to GIT, it is good to have some basic understanding of what GIT is and how it works. GIT is different from other version control systems. It is a distributed version control system and not a centralized version control system. It saves the data lot differently than SVN does. Refer to the section References for information on GIT basics.

6.3 Import Atmel START Project into MPLAB X IDE

Follow these steps to import the Atmel START project into the MPLAB X IDE:

1. Open MPLAB X IDE.
2. Select *file > Import > START MPLAB Project*.
3. Upload the `.atzip` file using the Browse option.
4. Ensure that the correct device has been selected, choose the programming tool.
5. Select the compiler toolchain as Arm.
6. Specify the location of the project and verify all the information is accurate.

6.4 Adding Custom Linker Script to MPLAB Harmony v3 Project

The MPLAB Harmony v3 project uses a default linker script that comes with MPLAB X IDE installation. A few projects might need custom linker scripts. The MPLAB Harmony v3 supports adding custom linker scripts to a project. Use the following steps to include the correct linker script in the project:

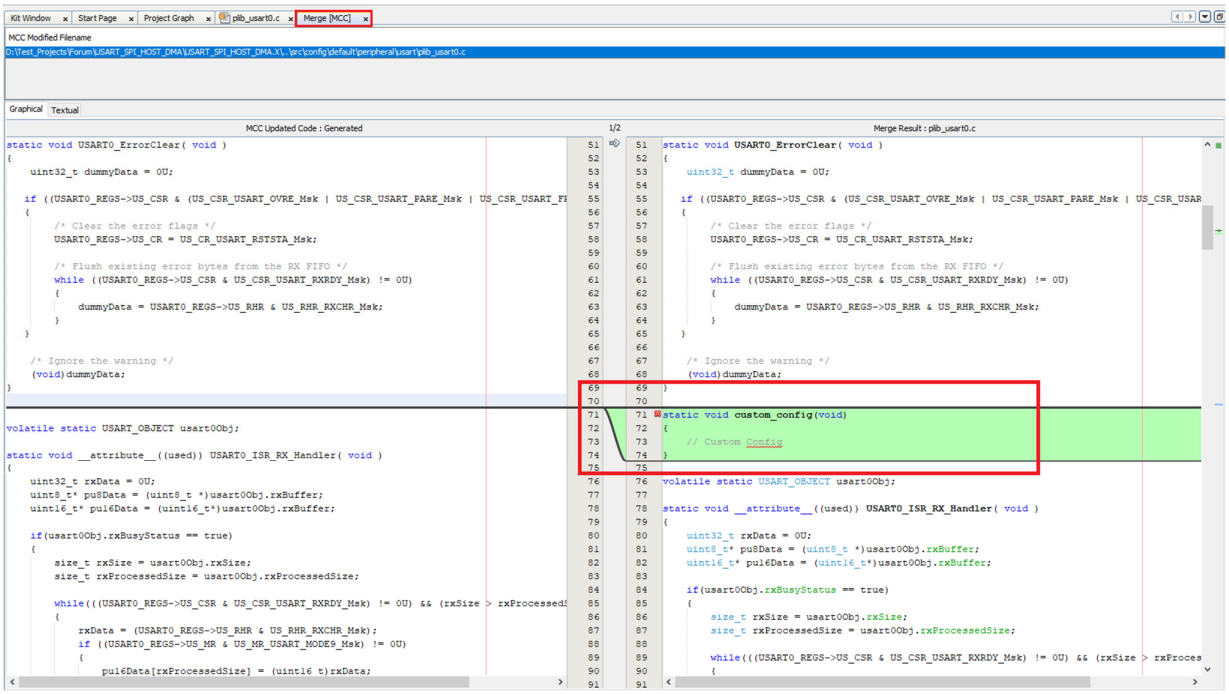
1. Right-click on the Linker Files folder in *project > Add Existing item > select specific linker script* which needs to be linked.
2. *Project Properties > XC32 global options > xc32-ld > Additional options > add linker switch* to use the following linker file for linking, `[-T ..\..\..\yy\xxxx.ld]`.

6.5 Modifying the PLIB and Maintaining it While Regenerating the Code

There are some rare use cases where a specific feature or minor modification must be done in a PLIB (for example, API addition). In those cases, the user can modify the PLIB as needed. After the manual PLIB change, the user can change the PLIB configuration. While regenerating the code after this change, the PLIB will be changed according to the new configuration.

The user can maintain their manual change and allow only the configuration changes which are updated in the PLIBs. To do this, when the merge window appears, the user can accept or reject the changes in the MPLAB Harmony v3 framework files.

Figure 6-1. Keeping Custom Changes in PLIB While Regenerating The Code



7. References

For additional information regarding Microchip products and services, visit the Microchip website or contact a local Microchip sales representative.

- MPLAB X IDE:
  www.microchip.com/en-us/tools-resources/develop/mplab-x-ide
- MPLAB X IDE XC Compilers:
  www.microchip.com/en-us/tools-resources/develop/mplab-xc-compilers
- MPLAB X IDE Device Support:
  www.microchip.com/content/dam/mchp/documents/DEV/ProductDocuments/SupportingCollateral/Device_Support.pdf
- MPLAB Harmony v3 web link:
  www.microchip.com/en-us/tools-resources/configure/mplab-harmony
- MPLAB Harmony GitHub Link:
  https://github.com/Microchip-MPLAB-Harmony
- How to Setup MPLAB Harmony v3 Software Development Framework:
  ww1.microchip.com/downloads/en/DeviceDoc/How_to_Setup_MPLAB_%20Harmony_v3_Software_Development_Framework_DS90003232C.pdf
- ASF web link:
  www.microchip.com/en-us/tools-resources/develop/libraries/advanced-software-framework
- Atmel START to MPLAB Harmony 3 Migration Guide AN (DS00005413)
  ww1.microchip.com/downloads/aemDocuments/documents/MCU32/ProductDocuments/UserGuides/Atmel_START_to_MPLAB_Harmony3_Migration_Guide_DS70005413A.pdf
- ASFv3 to MPLAB Harmony 3 Migration Guide (DS70005412)
  ww1.microchip.com/downloads/en/DeviceDoc/ASF3_to_Harmony3_%20Migration_%20Guide_DS70005412A.pdf
- GIT basics:
  git-scm.com/docs
- For additional information about 32-bit Microcontroller Collaterals and Solutions, refer to the *32-bit Microcontroller Collateral and Solutions Reference Guide* (DS70005534):
  ww1.microchip.com/downloads/aemDocuments/documents/MCU32/ProductDocuments/ReferenceManuals/32-bit-Microcontroller-Collateral-and-Solutions-Reference-Guide-DS70005534.pdf
- For other relevant information, refer to the Microchip website.
  www.microchip.com/

8. Revision History

Revision B - 09/2024

The following updates were performed for this revision:

- Throughout the document, the MHC was updated to the MCC to reflect a tools upgrade
- The following sections were updated with new hyperlinks, images, and text to reflect the tools upgrade to the MCC:
  - Introduction to MPLAB Harmony v3
  - Modular Downloads and Repository Structure
  - Flexible Firmware Design Models
  - MPLAB Harmony v3 GitHub Repository Handling
  - Downloading Packages Using the MCC Content Manager
  - Updating the Repositories Using the MCC Content Manager
  - MPLAB Harmony v3 User Interface
  - MPLAB Harmony v3 - Exploring a PLIB Demonstration Project
  - Compiling, Programming and Debugging a PLIB Demonstration Project
  - Modifying Components, Configurations and Regenerating Code for the PLIB Demonstration
  - Harmony v3 Services - Standard I/O (STDIO), Toolchain Supports
  - Creating a Simple Application Demonstration Using Harmony v3
  - Modifying the PLIB and Maintaining it While Regenerating the Code
- Updated References with all new links

Revision A - 01/2020

This is the initial release of the document.

Microchip Information

The Microchip Website

Microchip provides online support via our website at 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 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: www.microchip.com/support

Microchip Devices Code Protection Feature

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

- Microchip products meet the specifications contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is secure when used in the intended manner, within operating specifications, and under normal conditions.
- Microchip values and aggressively protects its intellectual property rights. Attempts to breach the code protection features of Microchip product is strictly prohibited and may violate the Digital Millennium Copyright Act.
- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of its code. Code protection does not mean that we are guaranteeing the product is "unbreakable". Code protection is constantly evolving. Microchip is committed to continuously improving the code protection features of our products.

Legal Notice

This publication and the information herein may be used only with Microchip products, including to design, test, and integrate Microchip products with your application. Use of this information in any other manner violates these terms. Information regarding device applications is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure

that your application meets with your specifications. Contact your local Microchip sales office for additional support or, obtain additional support at www.microchip.com/en-us/support/design-help/client-support-services.

THIS INFORMATION IS PROVIDED BY MICROCHIP "AS IS". MICROCHIP MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WHETHER EXPRESS OR IMPLIED, WRITTEN OR ORAL, STATUTORY OR OTHERWISE, RELATED TO THE INFORMATION INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE, OR WARRANTIES RELATED TO ITS CONDITION, QUALITY, OR PERFORMANCE.

IN NO EVENT WILL MICROCHIP BE LIABLE FOR ANY INDIRECT, SPECIAL, PUNITIVE, INCIDENTAL, OR CONSEQUENTIAL LOSS, DAMAGE, COST, OR EXPENSE OF ANY KIND WHATSOEVER RELATED TO THE INFORMATION OR ITS USE, HOWEVER CAUSED, EVEN IF MICROCHIP HAS BEEN ADVISED OF THE POSSIBILITY OR THE DAMAGES ARE FORESEEABLE. TO THE FULLEST EXTENT ALLOWED BY LAW, MICROCHIP'S TOTAL LIABILITY ON ALL CLAIMS IN ANY WAY RELATED TO THE INFORMATION OR ITS USE WILL NOT EXCEED THE AMOUNT OF FEES, IF ANY, THAT YOU HAVE PAID DIRECTLY TO MICROCHIP FOR THE INFORMATION.

Use of Microchip devices in life support and/or safety applications is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless Microchip from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any Microchip intellectual property rights unless otherwise stated.

Trademarks

The Microchip name and logo, the Microchip logo, Adaptec, AVR, AVR logo, AVR Freaks, BesTime, BitCloud, CryptoMemory, CryptoRF, dsPIC, flexPWR, HELDO, IGLOO, JukeBlox, KeeLoq, Kleer, LANCheck, LinkMD, maXStylus, maXTouch, MediaLB, megaAVR, Microsemi, Microsemi logo, MOST, MOST logo, MPLAB, OptoLyzer, PIC, picoPower, PICSTART, PIC32 logo, PolarFire, Prochip Designer, QTouch, SAM-BA, SenGenuity, SpyNIC, SST, SST Logo, SuperFlash, Symmetricom, SyncServer, Tachyon, TimeSource, tinyAVR, UNI/O, Vectron, and XMEGA are registered trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

AgileSwitch, ClockWorks, The Embedded Control Solutions Company, EtherSynch, Flashtec, Hyper Speed Control, HyperLight Load, Libero, motorBench, mTouch, Powermite 3, Precision Edge, ProASIC, ProASIC Plus, ProASIC Plus logo, Quiet-Wire, SmartFusion, SyncWorld, TimeCesium, TimeHub, TimePictra, TimeProvider, and ZL are registered trademarks of Microchip Technology Incorporated in the U.S.A.

Adjacent Key Suppression, AKS, Analog-for-the-Digital Age, Any Capacitor, AnyIn, AnyOut, Augmented Switching, BlueSky, BodyCom, Clockstudio, CodeGuard, CryptoAuthentication, CryptoAutomotive, CryptoCompanion, CryptoController, dsPICDEM, dsPICDEM.net, Dynamic Average Matching, DAM, ECAN, Espresso T1S, EtherGREEN, EyeOpen, GridTime, IdealBridge, IGaT, In-Circuit Serial Programming, ICSP, INICnet, Intelligent Paralleling, IntelliMOS, Inter-Chip Connectivity, JitterBlocker, Knob-on-Display, MarginLink, maxCrypto, maxView, memBrain, Mindi, MiWi, MPASM, MPF, MPLAB Certified logo, MPLIB, MPLINK, mSiC, MultiTRAK, NetDetach, Omniscient Code Generation, PICDEM, PICDEM.net, PICkit, PICtail, Power MOS IV, Power MOS 7, PowerSmart, PureSilicon, QMatrix, REAL ICE, Ripple Blocker, RTAX, RTG4, SAM-ICE, Serial Quad I/O, simpleMAP, SimpliPHY, SmartBuffer, SmartHLS, SMART-I.S., storClad, SQI, SuperSwitcher, SuperSwitcher II, Switchtec, SynchroPHY, Total Endurance, Trusted Time, TSHARC, Turing, USBCheck, VariSense, VectorBlox, VeriPHY, ViewSpan, WiperLock, XpressConnect, and ZENA are trademarks of Microchip Technology Incorporated in the U.S.A. and other countries.

SQTP is a service mark of Microchip Technology Incorporated in the U.S.A.

The Adaptec logo, Frequency on Demand, Silicon Storage Technology, and Symmcom are registered trademarks of Microchip Technology Inc. in other countries.

GestIC is a registered trademark of Microchip Technology Germany II GmbH & Co. KG, a subsidiary of Microchip Technology Inc., in other countries.

ISBN: 978-1-6683-0223-1

Quality Management System

For information regarding Microchip's Quality Management Systems, please visit 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: www.microchip.com/support Web Address: www.microchip.com	**Australia - Sydney** Tel: 61-2-9868-6733	**India - Bangalore** Tel: 91-80-3090-4444	**Austria - Wels** Tel: 43-7242-2244-39 Fax: 43-7242-2244-393
Atlanta Duluth, GA Tel: 678-957-9614 Fax: 678-957-1455	**China - Beijing** Tel: 86-10-8569-7000	**India - New Delhi** Tel: 91-11-4160-8631	**Denmark - Copenhagen** Tel: 45-4485-5910 Fax: 45-4485-2829
Austin, TX Tel: 512-257-3370	**China - Chengdu** Tel: 86-28-8665-5511	**India - Pune** Tel: 91-20-4121-0141	**Finland - Espoo** Tel: 358-9-4520-820
Boston Westborough, MA Tel: 774-760-0087 Fax: 774-760-0088	**China - Chongqing** Tel: 86-23-8980-9588	**Japan - Osaka** Tel: 81-6-6152-7160	**France - Paris** Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79
Chicago Itasca, IL Tel: 630-285-0071 Fax: 630-285-0075	**China - Dongguan** Tel: 86-769-8702-9880	**Japan - Tokyo** Tel: 81-3-6880- 3770	**Germany - Garching** Tel: 49-8931-9700
Dallas Addison, TX Tel: 972-818-7423 Fax: 972-818-2924	**China - Guangzhou** Tel: 86-20-8755-8029	**Korea - Daegu** Tel: 82-53-744-4301	**Germany - Haan** Tel: 49-2129-3766400
Detroit Novi, MI Tel: 248-848-4000	**China - Hangzhou** Tel: 86-571-8792-8115	**Korea - Seoul** Tel: 82-2-554-7200	**Germany - Heilbronn** Tel: 49-7131-72400
Houston, TX Tel: 281-894-5983	**China - Hong Kong SAR** Tel: 852-2943-5100	**Malaysia - Kuala Lumpur** Tel: 60-3-7651-7906	**Germany - Karlsruhe** Tel: 49-721-625370
Indianapolis Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380	**China - Nanjing** Tel: 86-25-8473-2460	**Malaysia - Penang** Tel: 60-4-227-8870	**Germany - Munich** Tel: 49-89-627-144-0 Fax: 49-89-627-144-44
Los Angeles Mission Viejo, CA Tel: 949-462-9523 Fax: 949-462-9608 Tel: 951-273-7800	**China - Qingdao** Tel: 86-532-8502-7355	**Philippines - Manila** Tel: 63-2-634-9065	**Germany - Rosenheim** Tel: 49-8031-354-560
Raleigh, NC Tel: 919-844-7510	**China - Shanghai** Tel: 86-21-3326-8000	**Singapore** Tel: 65-6334-8870	**Israel - Hod Hasharon** Tel: 972-9-775-5100
New York, NY Tel: 631-435-6000	**China - Shenyang** Tel: 86-24-2334-2829	**Taiwan - Hsin Chu** Tel: 886-3-577-8366	**Italy - Milan** Tel: 39-0331-742611 Fax: 39-0331-466781
San Jose, CA Tel: 408-735-9110 Tel: 408-436-4270	**China - Shenzhen** Tel: 86-755-8864-2200	**Taiwan - Kaohsiung** Tel: 886-7-213-7830	**Italy - Padova** Tel: 39-049-7625286
Canada - Toronto Tel: 905-695-1980 Fax: 905-695-2078	**China - Suzhou** Tel: 86-186-6233-1526	**Taiwan - Taipei** Tel: 886-2-2508-8600	**Netherlands - Drunen** Tel: 31-416-690399 Fax: 31-416-690340
	China - Wuhan Tel: 86-27-5980-5300	**Thailand - Bangkok** Tel: 66-2-694-1351	**Norway - Trondheim** Tel: 47-72884388
	China - Xian Tel: 86-29-8833-7252	**Vietnam - Ho Chi Minh** Tel: 84-28-5448-2100	**Poland - Warsaw** Tel: 48-22-3325737
	China - Xiamen Tel: 86-592-2388138		**Romania - Bucharest** Tel: 40-21-407-87-50
	China - Zhuhai Tel: 86-756-3210040		**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