Live Update Application on SAM E54 MCU Using MPLAB Harmony v3

AN3767

Introduction

The dual-bank Flash on the SAM E54 microcontroller (MCU) allows the application to implement the Live Update feature. The Live Update feature in an application is a piece of code used to program the application code (firmware) to the inactive bank in the internal Flash or non-volatile memory (NVM). The Live Update feature updates to a newer version of the firmware without affecting the working application on the active bank. While the Live Update feature of the current application is under progress, the existing version of the application continues to run.

The following are features of the Live Update application:

- The SAM E54 MCU with dual-bank Flash is divided into Bank A and Bank B, each having a bootloader and application code section.
- For the first time, the UART Fail-Safe Bootloader is programmed into Bank A, upon reset the device runs from Bank A. Then the Live Update application and the UART Fail-Safe Bootloader are merged using the script, and programmed using a UART Fail-Safe Bootloader into Bank B.
- The Live Update application communicates to the Host program (typically running on a PC) to receive the firmware through a serial interface, such as the UART.
- The code that implements the Live Update feature runs only when there is a request for the new firmware upgrade, otherwise it will be in the idle state allowing other application functions to run.

This document describes implementing the Live Update application on the SAM E54 MCU with the usage of the dual bank Flash.

Table of Contents

1. Hardware and Software Requirements

1.1 SAM E54 Curiosity Ultra Development Board

The SAM E54 Curiosity Ultra Development Board is a development kit for evaluating the SAM E54 microcontroller. The SAM E54 is based on an Arm® Cortex® -M4 capable of running at 120 MHz. The SAM E54 Curiosity Ultra Development Board includes an integrated programmer and debugger, hence additional hardware is not required to get started. Users can expand functionality through MikroElectronika mikroBUS™ Click™ adapter boards, add Ethernet connectivity with the Microchip PHY Daughter Board, add Wi-Fi™ connectivity capability using the Microchip expansion boards, and add audio input and output capability with Microchip audio daughter boards. With or without expansion boards, the SAM E54 Curiosity Ultra Development Board provides the freedom to develop for a variety of applications, including Bluetooth® audio, CAN, graphical user interface (GUI), Internet of Things (IoT), robotics development, and proof-of-concept designs.

The SAM E54 Curiosity Ultra Development Board is available at Microchip Direct.

1.2 MPLAB® X Integrated Development Environment (IDE) and MPLAB XC Compilers

The MPLAB X IDE is an expandable, highly configurable software program that incorporates powerful tools to help discover, configure, develop, debug, and qualify embedded designs for most of the Microchip microcontrollers.

The MPLAB X IDE is available at Microchip Website. The application, discussed in this document, uses MPLAB X IDE version 6.15. MPLAB XC Compilers are available at Microchip Website. The application, discussed in this document, uses MPLAB XC32 version 4.35.

1.3 MPLAB Harmony v3

MPLAB Harmony v3 is a fully integrated embedded software development framework that provides flexible and interoperable software modules that allows the dedication of resources for creating applications for the 32-bit PIC® and SAM devices, rather than dealing with device details, complex protocols, and library integration challenges.

It includes the MPLAB Code Configurator (MCC), an easy-to-use development tool with a GUI that simplifies device set up, library selection, configuration, and application development. The MPLAB Code Configurator (MCC) is available as a plug-in that integrates with the MPLAB X IDE and has a separate Java executable for stand-alone use with other development environments.

The application, discussed in this document, uses the following MPLAB Harmony v3 repositories. Users can download these repositories from GitHub or use the MPLAB Code Configurator (MCC).

- CSP (MPLAB Harmony v3 Chip Support Package)
- DEV_PACKS (MPLAB Harmony v3 Product Database)
- MCC (MPLAB Code Configurator)
- bootloader (Bootloader)
- bootloader_apps_uart (UART Bootloader Applications)
- MPLAB Harmony Reference Apps (MPLAB Harmony v3 Reference Applications)

1.4 Python

The application uses Python v3.10 scripts for merging the binaries (UART Fail-Safe Bootloader and Live Update application) and sending the merged binary from the Host PC to the SAM E54 Curiosity Ultra Development Board.

2. Design

The design of the Live Update application on the SAM E54 MCU uses the dual-bank Flash memory feature. The banks are named Bank A and Bank B. At any point in time, the application considers the bank on which it is currently executing as an active bank, while the other bank is marked as an inactive bank.

The dual-bank Flash enables programming of the inactive bank with a new version of the firmware while running the current version of the firmware from the active bank.

The Live Update application is divided into two tasks:

- Application Task
- Live Update Task

Application Task – Represents the end-user application. The current version of the end-user application will be running from the active bank of the dual-bank Flash, while the inactive bank is available to upgrade the end-user application with the latest version of the firmware. The Application Task discussed in this document is minimalistic in nature. The Application Task checks the NVM Control status register to identify the Flash bank, which has the latest version of the firmware on the active bank, and toggles LED1 or LED2 every 500 ms if it is running from Bank A, or LED1 or LED2 every 1000 ms if it is running from Bank B. For example, it starts with toggling LED1 and when there is a Live Update request, the LED2 starts toggling after programming the firmware to the inactive bank. Similarly, the process repeats for every successful application firmware update.

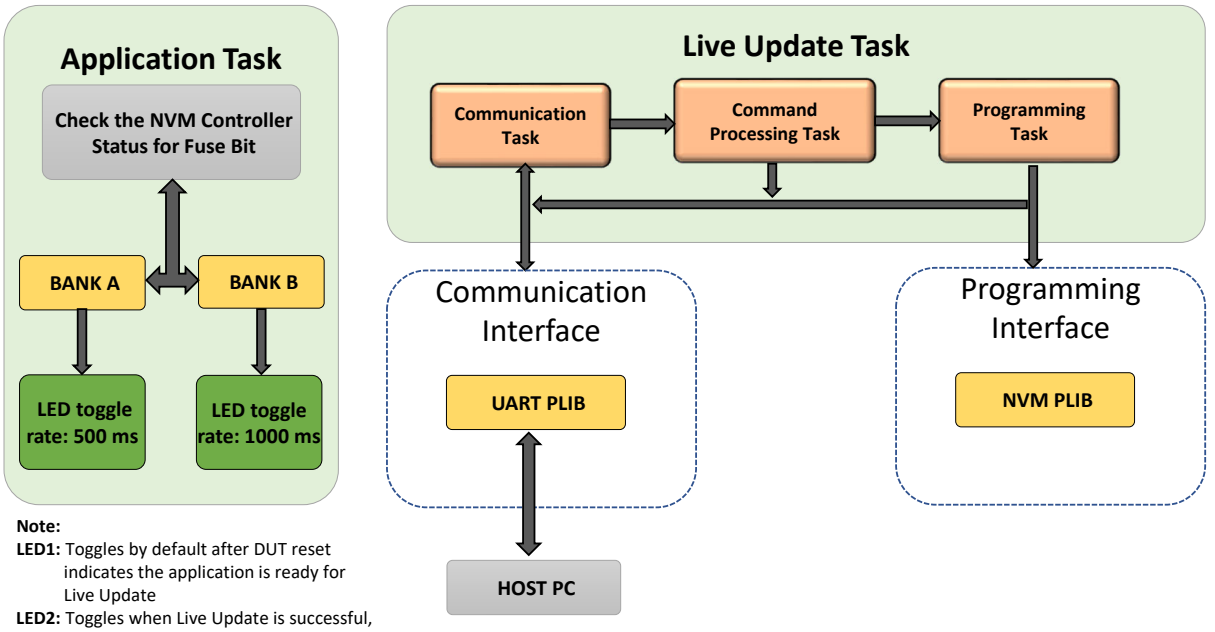
Live Update Task – Communicates with the personal computer Host application through a predefined communication protocol (See UART Live Update Protocol) and performs the firmware update at runtime when there is a request from the Host PC. Otherwise, it will be in an idle state and allows the application functions to run.

The Live Update task is divided into the following sub-tasks

- Communication Task.
- Command Processing Task.
- Programming Task.

The following diagram represents the Live Update application design.

Live Update Block Diagram



Note:
LED1: Toggles by default after DUT reset indicates the application is ready for Live Update
LED2: Toggles when Live Update is successful, means the firmware is programmed to the inactive bank

2.1 UART Live Update Protocol

The Live Update firmware communicates with the personal computer Host application by using a predefined communication protocol to exchange the data between the target and the Host, the protocol details are as follows.

The UART Live Update protocol is comprised of guard, data size, command, and data bytes, as shown in the following figure.

Figure 2-2. UART Live Update Protocol

- **Guard**
  - The guard is a constant value `0x5048434D`.
  - This value provides the protection against the spurious commands.
  - Live Update firmware always checks for the guard value at the start of packet reception and proceeds further accordingly.
- **Data Size**
  - This field indicates the number of data bytes to be received.
  - This value varies for different commands.
- **Command**
  - Indicates the command to be processed. Each command is 1 Byte in width.
  - The supported commands are as follows:
    - Unlock (0xA0)

  - Data (0xA1)
  - Verify (0xA2)
- **Data**
  - Contains the data to be processed based on the command.
  - Length of the data to be received is indicated by a data size field.
  - Bootloader receives the data in the size of words (4 bytes).
  - All data words must be sent in a little-endian (LSB first) format.

Response Codes

The Live Update will send a single character response code in response to each command. The sequential commands can only be sent after the response code is received for a previous command, or after a 100 ms timeout without a response.

The valid response codes are as follows:

- OK (0x50) – Command is received and processed successfully
- Error (0x51) – Errors during the processing of the command
- Invalid (0x52) – Invalid command is received
- CRC OK (0x53) – CRC verification was successful
- CRC Fail (0x54) – CRC verification failed

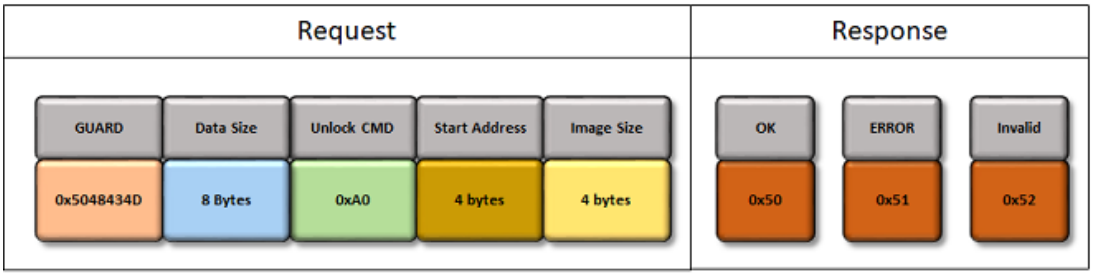
The following section Command Description covers a detailed description of the request and response codes used in the UART Live Update protocol.

2.2 Command Description

Unlock command

The Unlock command sequence with corresponding responses is shown in the following figure.

Figure 2-3. UART Live Update Unlock Command



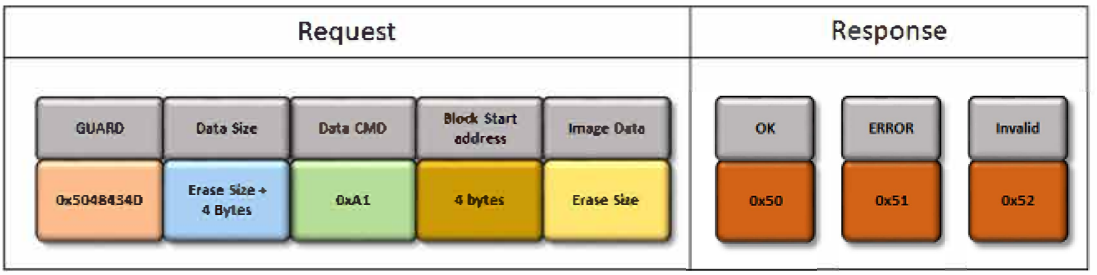
- The Unlock command must be issued before the first Data command:
  - It is used to calculate application start address and end address.
  - This information will be used to validate if the addresses sent are within the range of the Flash memory.
  - It will be used to validate whether the address coming with the data packet to be programmed is within the region for which the unlock command is invoked.
- Number of bytes of data to be received is 8 bytes (Start Address + Image Size)
- Start Address:
  - It is the application Start Address.
  - It is device dependent and should be always greater than or equal to the bootloader end address.

  - It must be aligned at an Erase Unit Size boundary.
  - To upgrade the bootloader itself this value must be set to '0'.
- The image size must be in increments of Erase Unit bytes, which depends on the device. For more information on Erase Unit Size, refer to *"Bootloader Sizing and Considerations"* in the MPLAB Harmony Bootloader Help.

Data command

The Data command sequence with corresponding responses is shown in the following figure.

Figure 2-4. UART Live Update Data Command

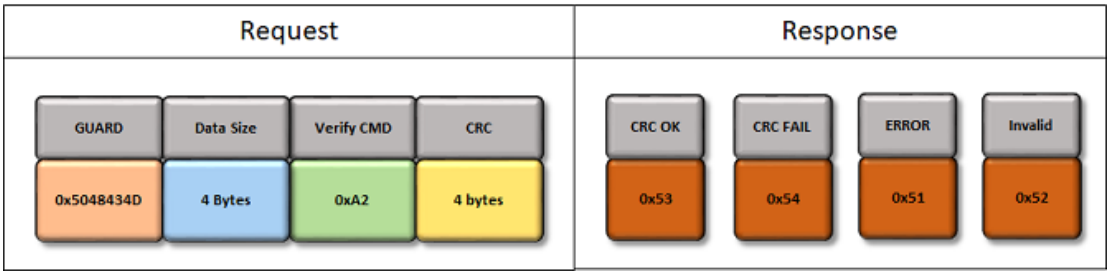


- Data command is used to send the image data.
- Data size is equal to the sum of block start address (4 Bytes).
- Erase Size is the size of the Flash to be erased (device dependent).
- Block start address must be located inside the region previously unlocked through the Unlock command.
- Attempts to request the write outside of the unlocked region will result in an error and supplied data will be discarded.

Verify command

The Verify command sequence with corresponding responses is shown in the following figure.

Figure 2-5. UART Live Update Verify Command



- Verify command is used to verify the image data sent and programmed.
- Image CRC is a standard IEEE® CRC32 with a polynomial of `0xEDB88320`.
- Internal CRC is calculated based on the values read from the Flash memory after programming, so it verifies the whole chain.
- Image CRC is calculated over the previously unlocked region.

2.3 Communication Task

The communication task is responsible for receiving data from the Host PC or embedded Host through the selected communication interface in interrupt mode. It validates the incoming packet from the Host with the expected header information before passing it to the command processor task.

2.4 Command Processing Task

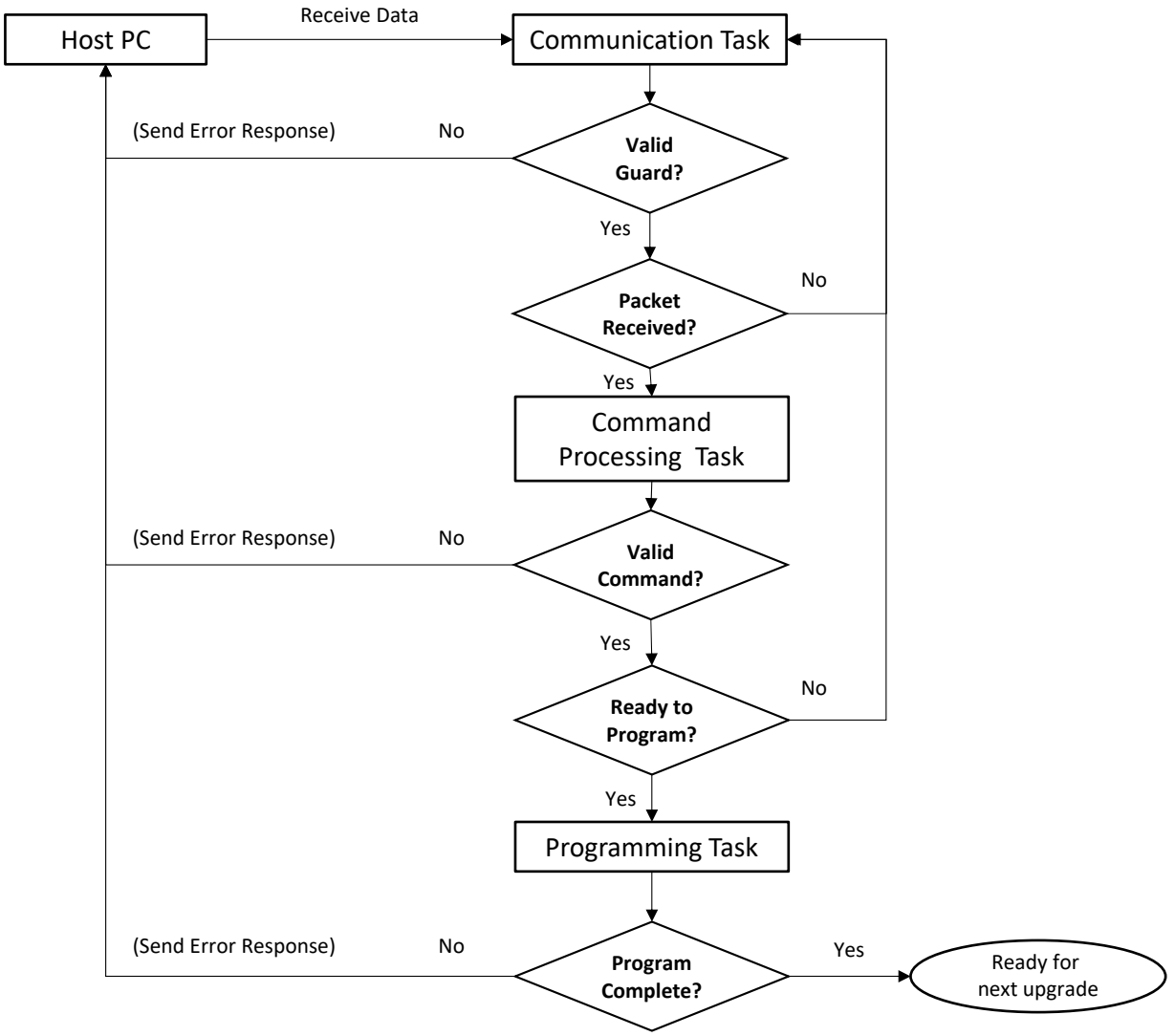
The Command Processing task processes the commands received from communication tasks and acts upon it. It provides the response back to the Host PC accordingly, and if the received command is a program command, then it gives control to the Programming task.

2.5 Programming Task

The Programming task is responsible for programming the internal Flash memory with a data packet received. It uses the NVM peripheral library to perform the unlock, erase, or write operations and invokes the communication task in parallel to receive the next packet while waiting for the Flash operation to complete.

The following figure shows the flowchart of the firmware upgrade execution:

Figure 2-6. Firmware Upgrade Execution Flowchart



2.6 Memory Layout

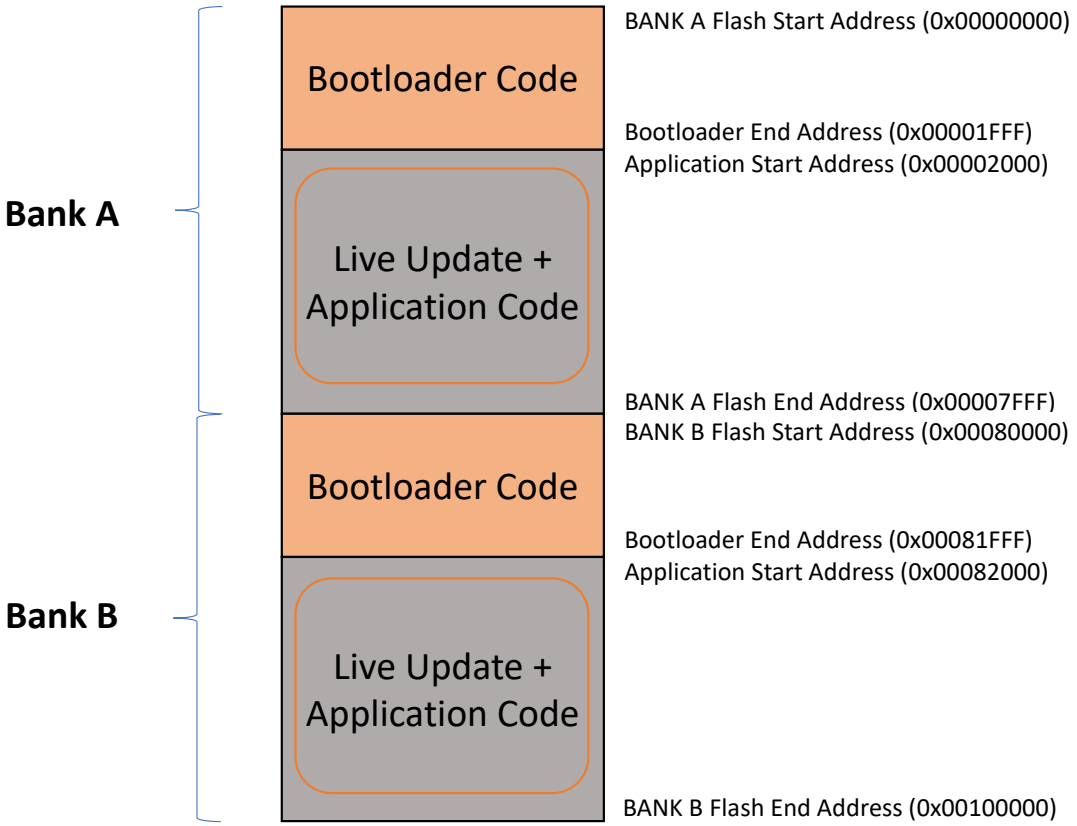
The Live Update application feature is supported for MCU devices which have dual bank support in Flash memory. The following sections represent the memory layout for the SAM E54 dual-bank Flash and the placement and configuration of the bootloader and Live Update application.

The SAM E54 Flash memory is configured on two banks: Bank A and Bank B. At the start of both the banks, the bootloader is situated and then followed by an application image.

By default, Bank A is mapped to the address `0x00000000` and Bank B is mapped to the address `0x00080000`. The bank mapped to the address `0x00000000` is referred to as the active bank (by default Bank A), whereas the other bank mapped to the address `0x00080000` is referred to as the inactive bank.

Note: The bank mapped at the address `0x00000000` is called as an active bank as the Cortex-M CPU architecture is designed to run the starting instruction from the address `0x00000000`.

Live Update memory layout SAM E54



2.7 Execution Flow

The first time, the Live Update application is programmed using the MPLAB® Harmony v3 UART Fail-Safe Bootloader. The following topic discusses the MPLAB Harmony v3 UART Fail-Safe Bootloader and Live Update application system-level execution flow.

UART Fail-Safe Bootloader System Level Execution Flow

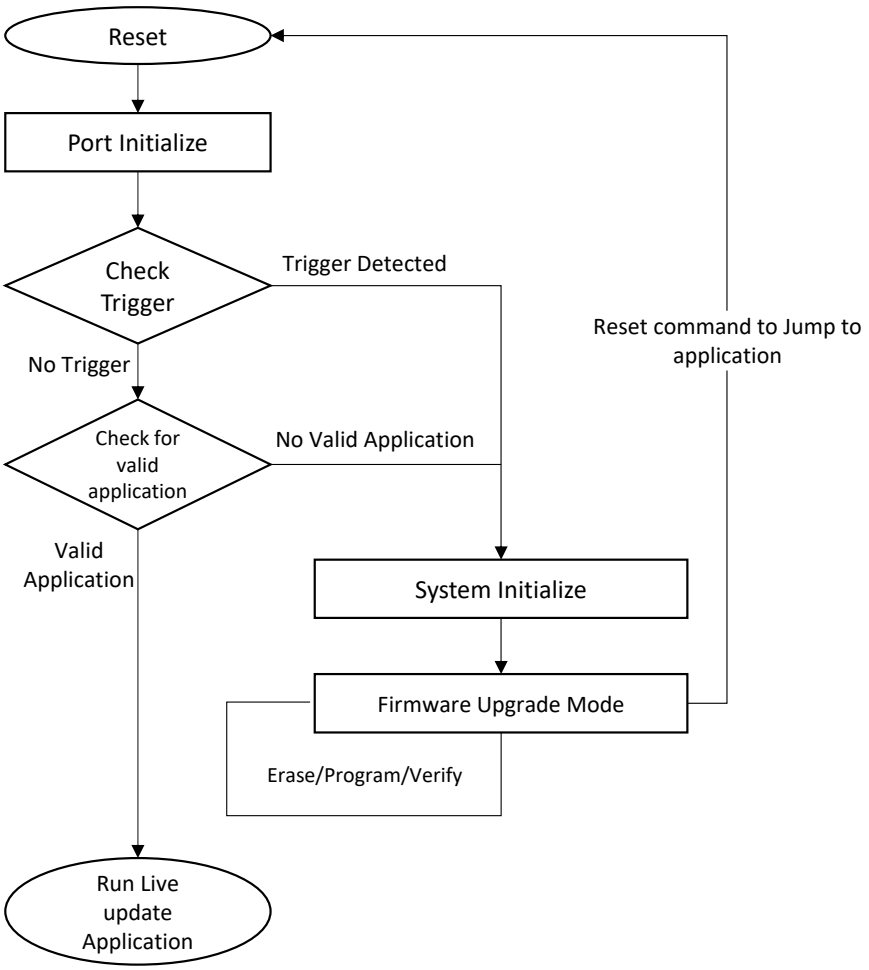
The UART Fail-Safe Bootloader code starts executing on a device reset. If there are no conditions to enter the firmware upgrade mode, the UART Fail-Safe Bootloader starts executing the user application. The UART Fail-Safe Bootloader performs Flash, Erase, and Program operations while in the firmware upgrade mode.

Trigger methods

- The Trigger method uses the on-board switch as a bootloader trigger pin to force entry to the bootloader at reset of the device.
- The Trigger method checks for a bootloader request pattern (`0x5048434D`) from the starting 16 bytes of RAM to force entry to the bootloader at reset of the device.

The following figure illustrates the MPLAB Harmony v3 UART Fail-Safe Bootloader system-level execution flow for programming of the Live Update application for the first time.

Figure 2-8. MPLAB Harmony v3 Bootloader Flow for Programming of the Live Update Application



The bootloader running from the active bank receives a merged image (bootloader + Live Update application) to upgrade the inactive bank at address `0x00080000`. After bootloader performs the successful upgrade, it notifies the Host application and the Live Update application waits for users to press the Switch SW2 to perform the Bank Swap and System Reset (BKSWRST). Pressing the Switch SW2 performs these actions:

- Swaps the memory banks to make the inactive bank active, and the active bank as inactive. The Bank A is made inactive while the Bank B is made active.
- Issues a reset command to run the upgraded application.

The information about which bank is mapped to the Flash address `0x00000000` is self-contained in special fuse bits in the Flash memory. These fuse bits can be erased or programmed individually. When the bootloader receives the BKSWRST command from the Host, it sets the BKSWRST bit in the Flash (NVM) control register. When the BKSWRST is set, the Flash (NVM) controller swaps the banks and sets the fuse bit (STATUS.AFIRST) based on the last status of the fuse bit (STATUS.AFIRST) as given below:

- STATUS.AFIRST = 0; Start address of the Bank B is mapped to `0x00000000`
- STATUS.AFIRST = 1; Start address of the Bank A is mapped to `0x00000000`

On reset, the Flash (NVM) controller checks the status of the fuse bit (STATUS.AFIRST) and jumps to the active memory bank to run the code.

For additional information on the MPLAB Harmony v3 UART Fail-Safe Bootloader, refer *<Harmony folder>/bootloader_apps_uart/apps/uart_fail_safe_bootloader/bootloader*.

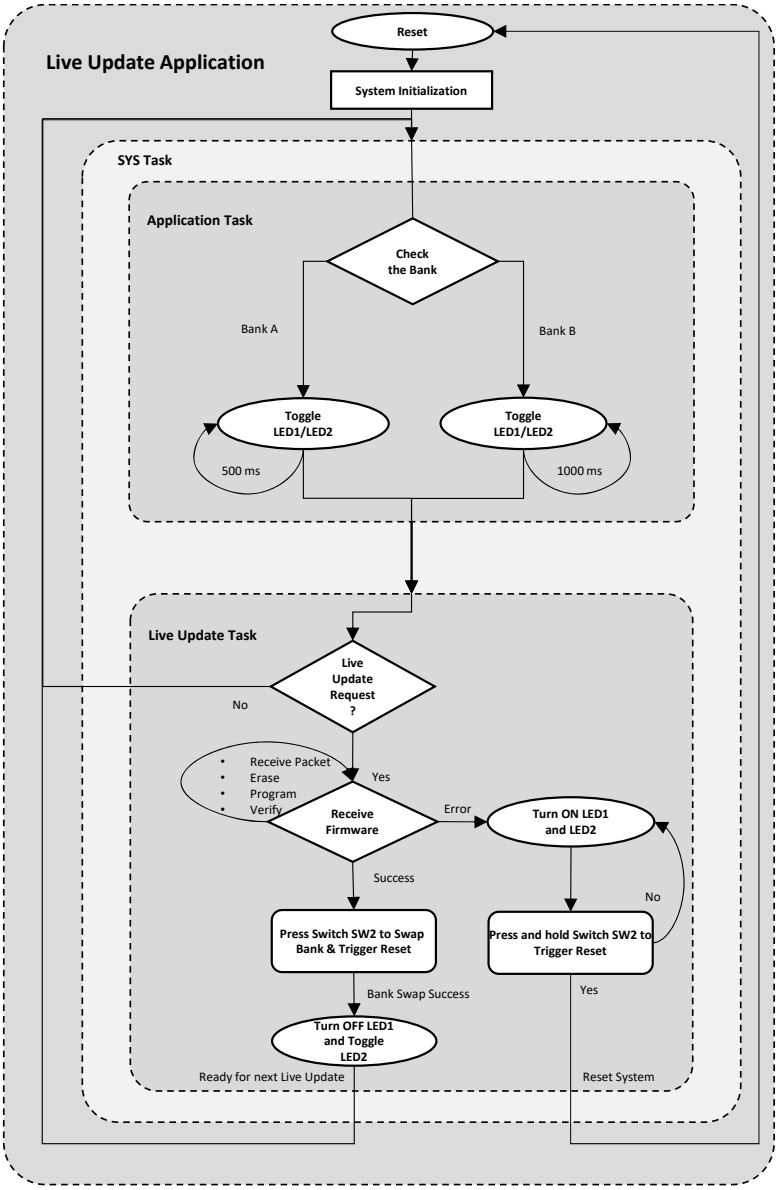
Live Update System Level Execution Flow

After successful programming of the Live Update application through the MPLAB Harmony v3 UART Fail-Safe Bootloader, the bootloader loads the Live Update application and CPU starts executing it.

The Live Update application receives a new application image from the Host. After the Live Update application performs a successful upgrade, it moves to an idle state and waits for the next Live Update request. The newly updated firmware will be loaded when the Switch SW2 is pressed. Pressing the switch causes the Live Update application to swap the memory banks to make the inactive bank active, and the active bank as inactive. The Bank A is made inactive while the Bank B is made active. In addition, it issues a Reset command to run the upgraded application.

The following flow diagram illustrates the Live Update application execution.

Figure 2-9. Live Update Application Execution Flow



The Live Update application execution sequence is as follows:

1. On device reset, the bootloader loads the Live Update application from the active bank of the dual-bank Flash memory.
2. The Live Update application does the system initialization and calls the Application Task followed by the Live Update Task.
3. Application Task checks the active bank from which the Live Update application is executing and toggles the LED1 every 500 ms if it is running from Bank A, or LED2 every 1000 ms if it is running from Bank B.
4. The Live Update Task runs only when there is a request for a firmware upgrade. Otherwise, it will be in idle state and the Application Task will continue to run.
5. If the application receives a request for the firmware upgrade, the Live Update Task does the following:
   - Starts receiving the packets when the Unlock command is received from Host PC.
   - Receives the firmware image when the Data command is received from Host PC and programs in the inactive bank.
   - Verifies the received firmware image when the Verify command is received from Host PC and waits for the user to press the Switch SW2 to swap the bank and perform a device reset to run the upgraded application.
6. If any error occurs in Step 5 or if the Host application requests to update the active bank and the address falls into the active bank serial sector then the Live Update Task sends an error response, aborts the programming operation, and turns ON the LED1 and LED2 until the Switch SW2 is pressed to trigger the system reset for restarting the Live Update again.
7. If Live Update is successful, then it turns OFF the LED1 and toggles the LED2 until the Switch SW2 is pressed to swap the bank and trigger the system reset to run the upgraded application.

3. Configuration

The Live Update application comprises of these applications:

- UART Fail-Safe Bootloader (`uart_fail_safe_bootloader_sam_e54_xpro`). This application consists of the bootloader which is used to program the Live Update application for the first time.
- Live Update application (`uart_same54_uart_live_update`). This application consists of implementation of the User Application Task and Live Update feature.

Note: The MPLAB Harmony v3 bootloader project is available in the MPLAB Harmony v3 Bootloader UART Apps repository, which is available at the following path: *<Harmony folder>/bootloader_apps_uart/apps/uart_fail_safe_bootloader/bootloader*.

The following paths provide additional information on the bootloader and UART Fail-Safe Bootloader configuration: MPLAB Harmony Bootloader Help and *<Harmony folder>/bootloader_apps_uart/docs/index.html*.

3.1 Bootloader Linker Script

The bootloader library uses a custom linker (`btl.ld`) script generated through the MCC. The MCC generates the following:

- The specified bootloader size.
- Starting address of the code and read only data, ROM (read-only memory) section address.
- Starting address of the read write data, RAM (random-access memory) section address as shown in the figure below.

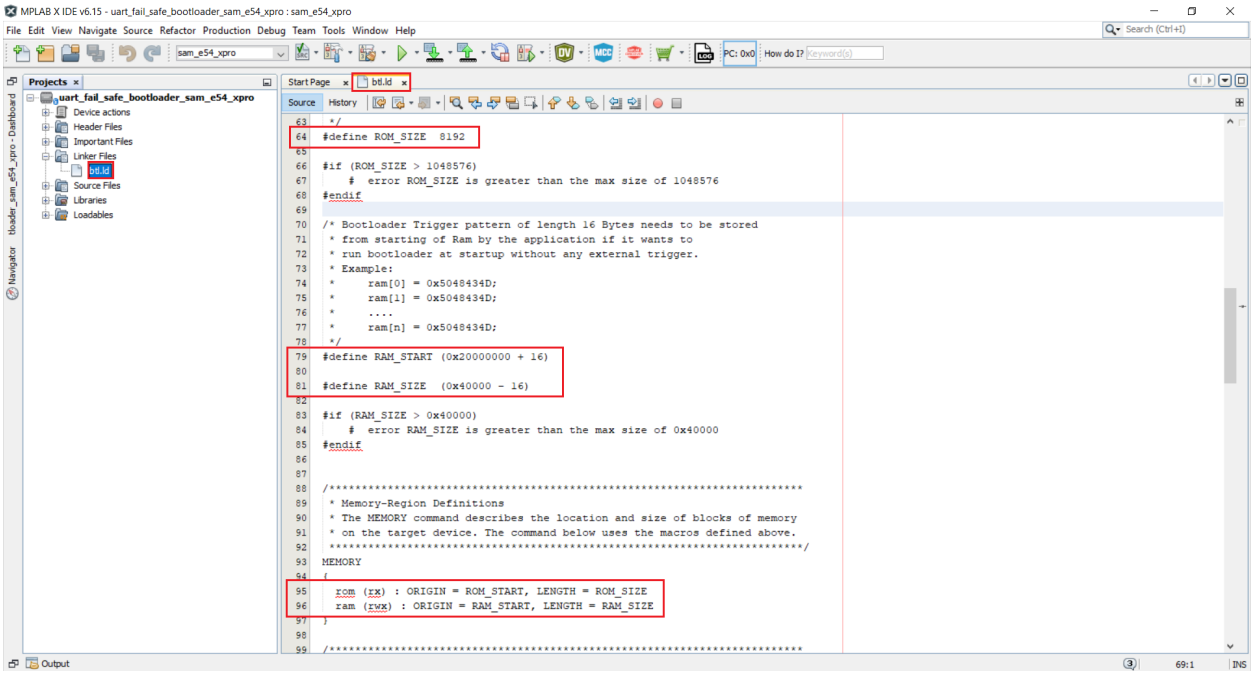
The values populated in the linker script are based on the bootloader component of the MCC configurations (Bootloader configuration).

Configure the Linker script for the bootloader to run from the RAM to achieve the simultaneous Flash memory write and reception of the next block of data.

The bootloader request pattern must be stored in 16 bytes of RAM on start by the application if it wants to run the bootloader at startup without any external trigger as shown in the following figure.

The bootloader size for the SAM E54 will be rounded off to the nearest erase unit size (8192 Bytes), even though the size of the bootloader is 1672 bytes in -O1 optimization. This enables adding of additional features on the bootloader and avoids application overlap with the bootloader.

Figure 3-1. Bootloader Custom Linker Script



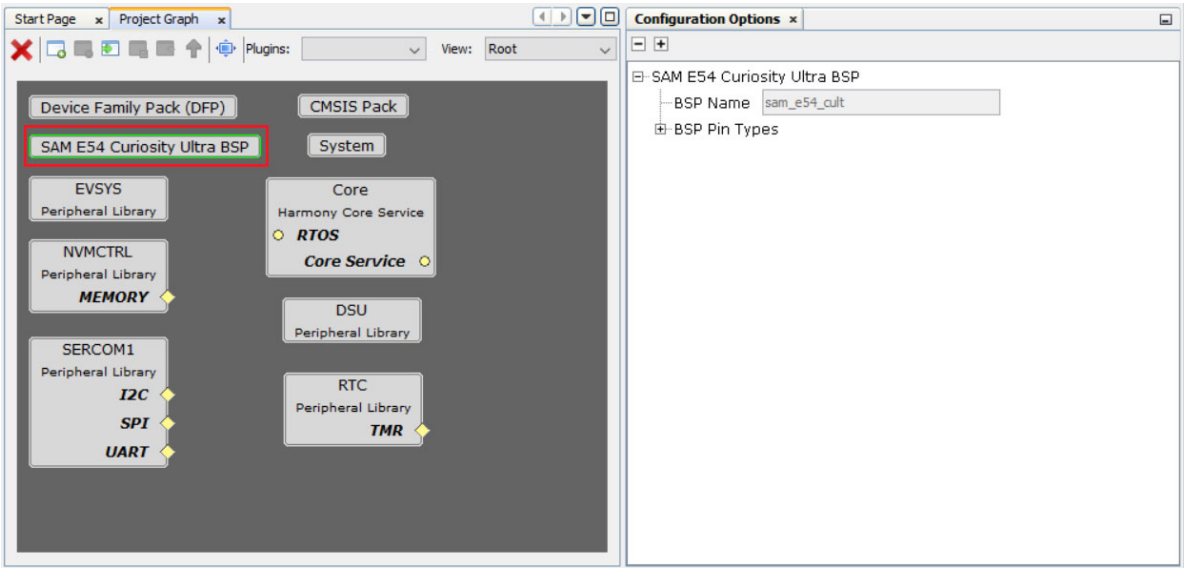
Note: Ensure that the memory region of the user application must not overlap with the memory region reserved for the bootloader.

3.2 MCC Configuration

Follow these steps to configure MCC, which is required to implement the Live Update application.

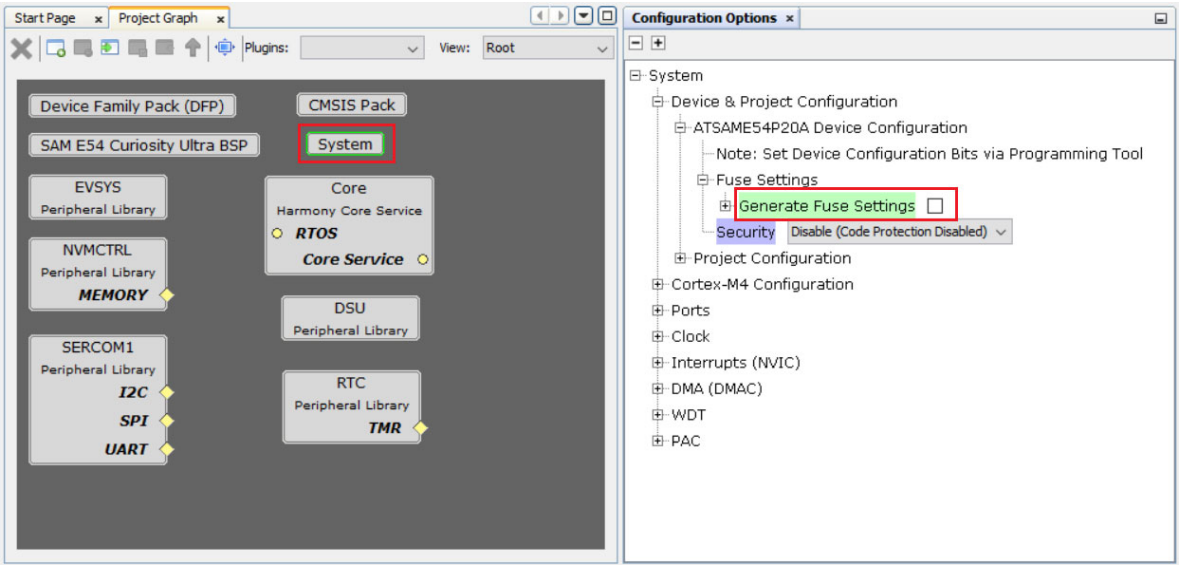
1. Click **SAM E54 Curiosity Ultra BSP** to add it to the switch and LED configurations.

Figure 3-2. Live Update Application BSP Configuration



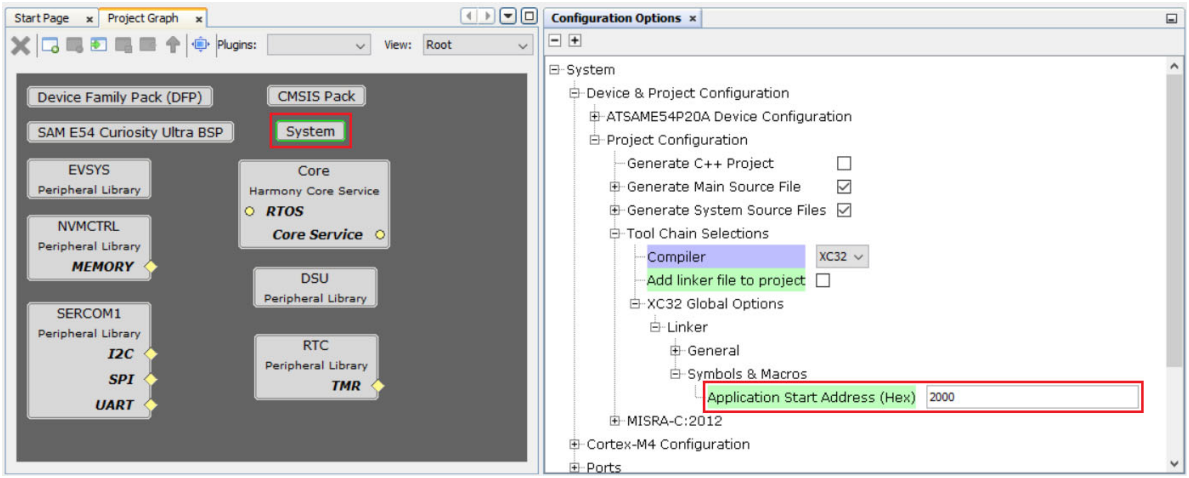
2. Remove the Device Fuse configurations from the custom linker script as it will be updated by the bootloader project.

Figure 3-3. Live Update Application Disable Fuse Settings



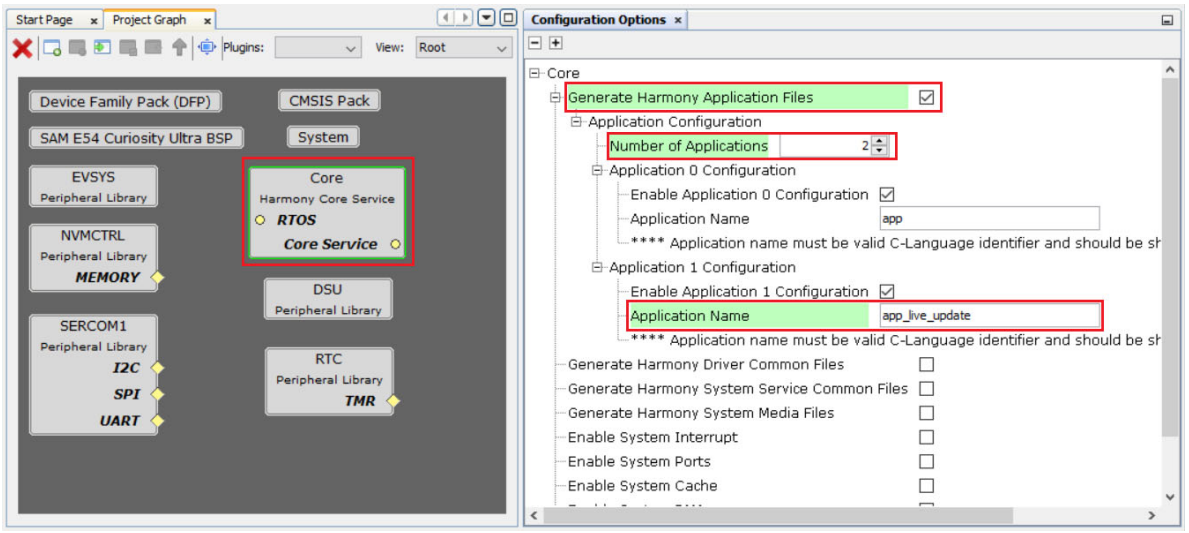
3. Configure the Live Update application start address that is matching with the application start address in the bootloader project.

Figure 3-4. Live Update Application Start Address Configuration



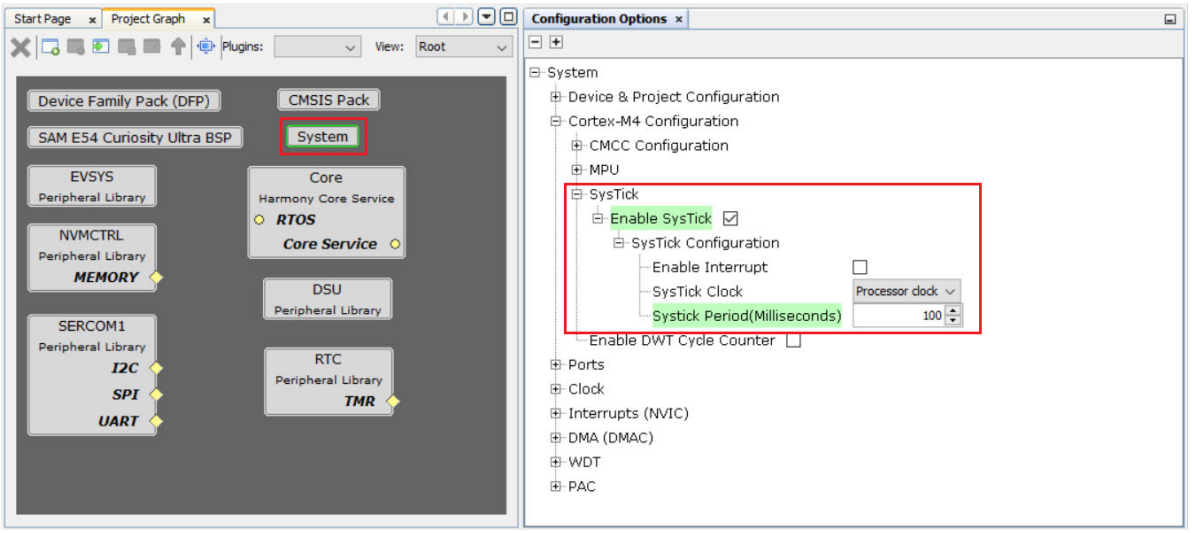
4. Add the MPLAB Harmony core component and configure the application and Live Update tasks. This generates the source file, such as `tasks.c`, which can be used to add if any sub-tasks and `user.h` for adding user configurations.

Figure 3-5. Live Update Application Harmony Core Configuration



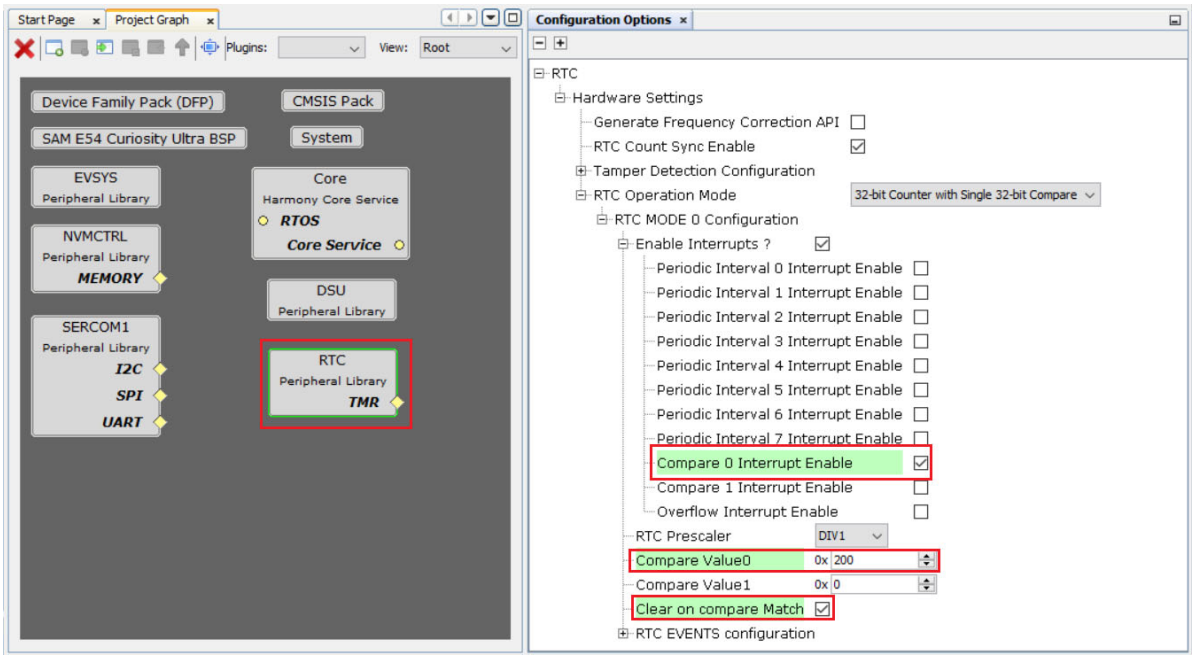
5. Add DSU module for CRC Calculations.
6. Configure the SysTick with a period (milliseconds) of 100 ms to receive the packet from the Host within a 100 ms turnaround time.

Figure 3-6. Live Update Application SysTick Configuration



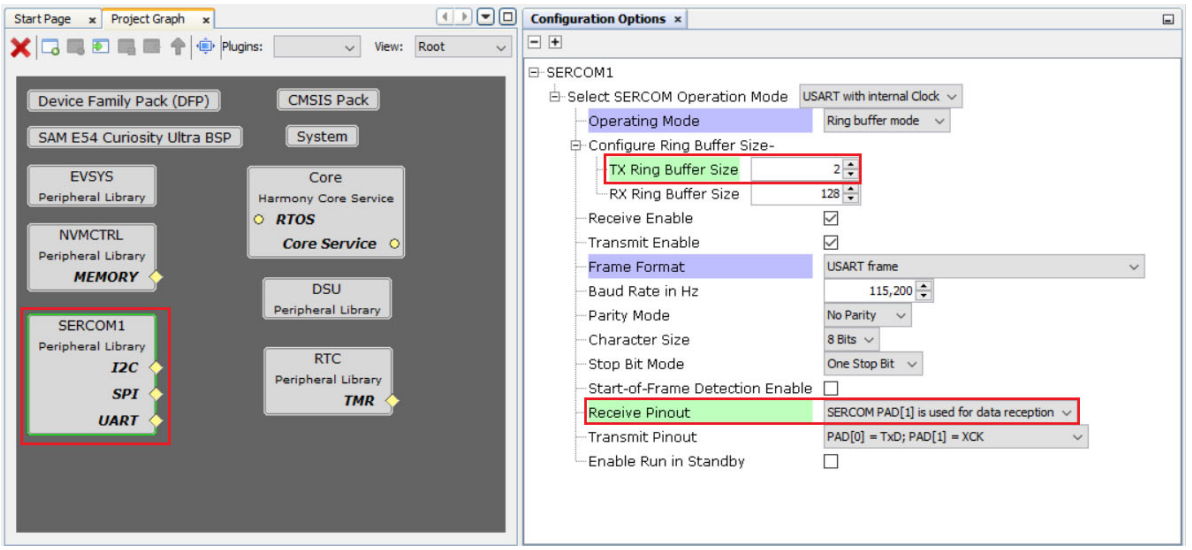
7. Add the RTC module to toggle the LED in different rates.

Figure 3-7. Live Update Application RTC Configuration



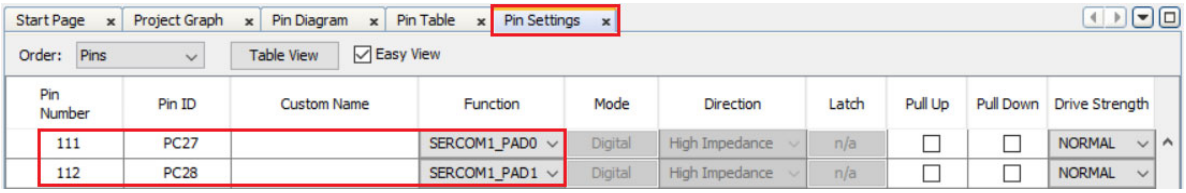
8. Add the UART module and enable the Ring Buffer to receive the Live Update application at run time from the Host PC.

Figure 3-8. Live Update Application UART Configuration



9. Launch the Pin Configurations plugin from *Project Graph > Plugins > Pin Configurations.*
10. Configure the UART pins to receive data from the Host PC.

Figure 3-9. Live Update Application UART Pin Configuration



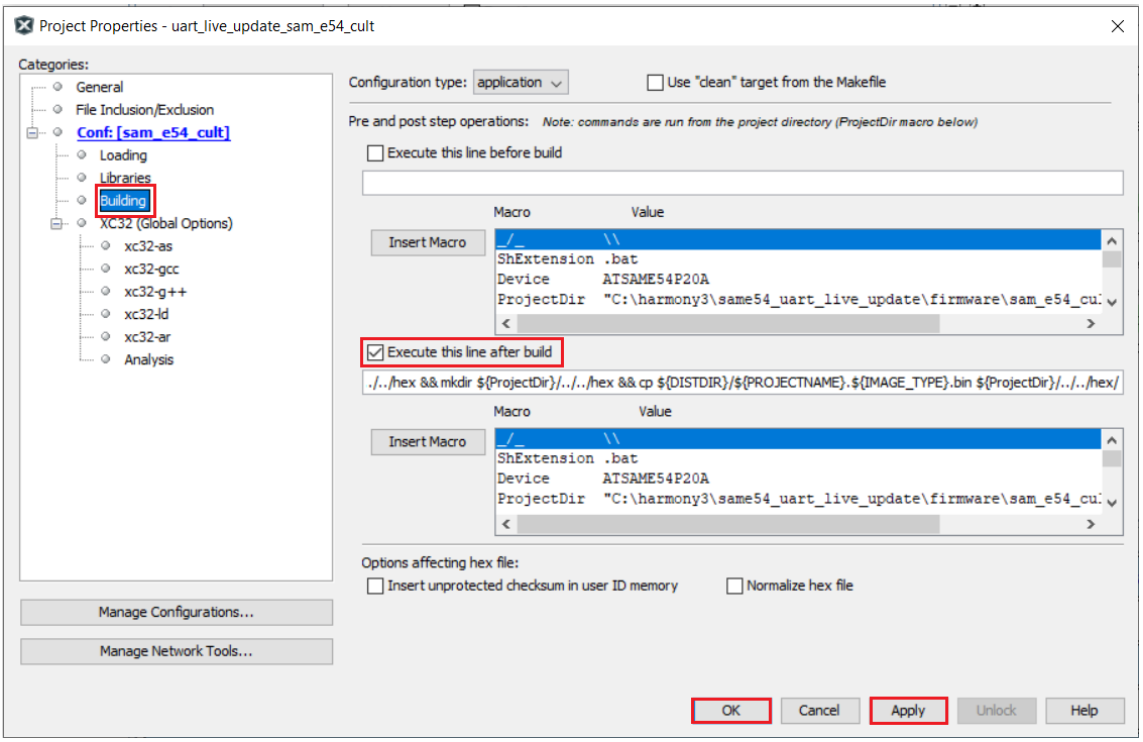
Note: The SERCOM1 is reserved for the Live Update task. It should not be used for any other purpose.

3.3 Project Settings

- **Preprocessor macro definitions:**
  - `ROM_ORIGIN` and `ROM_LENGTH` are the MPLAB XC32 linker variables which will be overridden with values provided in the MCC. See Live Update Application Start Address Configuration.
  - The application start address is auto-populated in the linker script with the value of the application start address provided in the MCC. See Bootloader Custom Linker Script after regeneration.
- **Execute command line after Build:**
  - The following build options can be used to generate the binary file from the .hex once the build is complete.

```
${MP_CC_DIR}/xc32-objcopy -I ihex -O binary ${DISTDIR}/${PROJECTNAME}.${IMAGE_TYPE}.hex ${DISTDIR}/${PROJECTNAME}.${IMAGE_TYPE}.bin
```

Figure 3-10. Live Update Application Binary Generation Setting



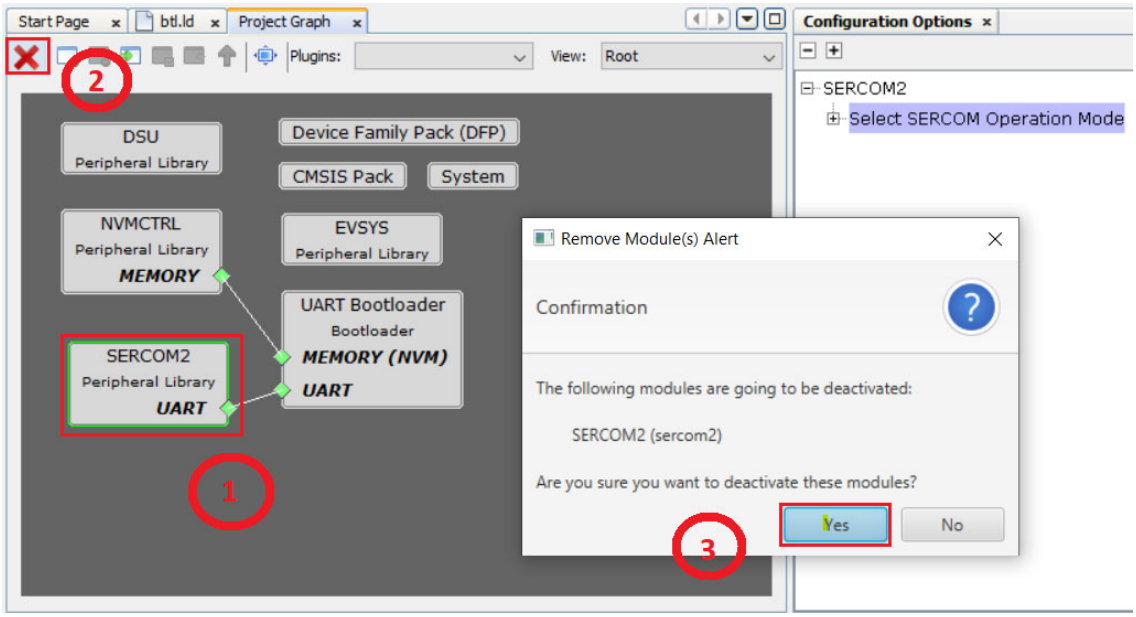
4. Running the Application

4.1 Running the Bootloader Application

Follow these steps to program the Live Update application using the MPLAB Harmony v3 UART Fail-Safe Bootloader:

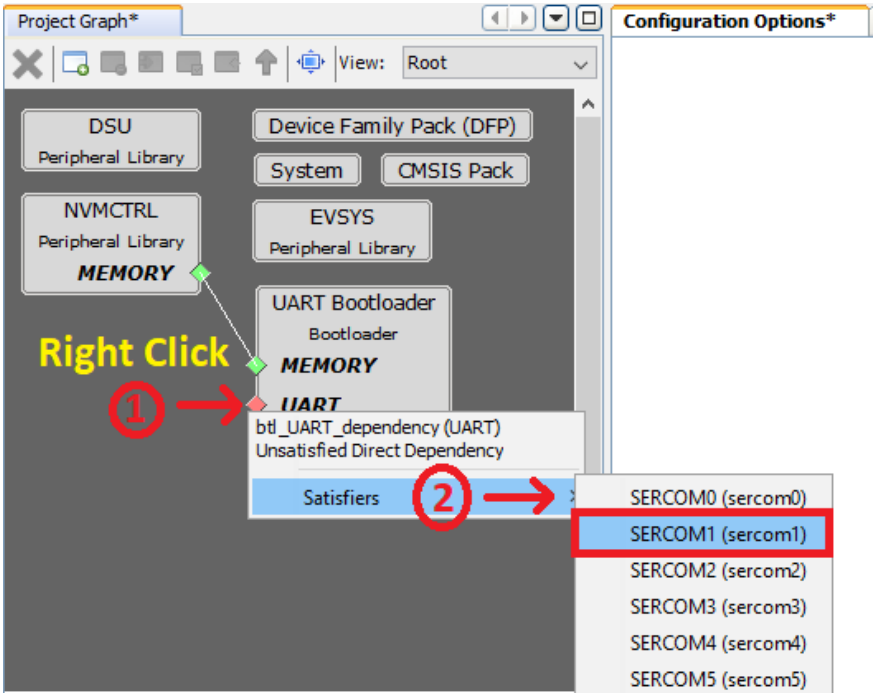
1. Download the MPLAB Harmony v3 Bootloader package.
2. Download the MPLAB Harmony v3 UART Bootloader Applications package.
3. Download the Live Update Application package.
4. Connect a micro-USB cable to the Debug port of the SAM E54 Curiosity Ultra Development Board.
5. Open the UART Fail-Safe Bootloader (path: *<Harmony folder>/bootloader_apps_uart/apps/uart_fail_safe_bootloader/bootloader/firmware/sam_e54_xpro.X*) using the MPLAB X IDE.
   Note: The project must be reconfigured to work on the SAM E54 Curiosity Ultra Development Board.
6. Launch MCC and follow these steps to reconfigure the project for the SAM E54 Curiosity Ultra Development Board.
   a. Remove the SERCOM2.

Figure 4-1. Bootloader MCC Reconfiguration - Remove SERCOM2 Peripheral



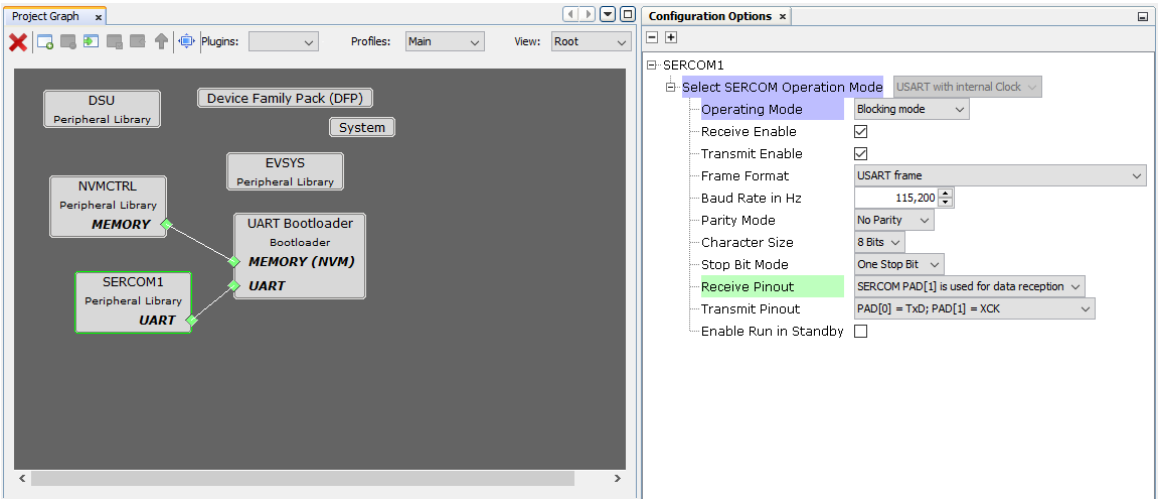
   b. Add the SERCOM1 to the Project Graph.

Figure 4-2. Bootloader MCC Reconfiguration - Add SERCOM1 Peripheral



c. Select the SERCOM1 Peripheral Library in the project graph and configure the Receive Pinout for data reception.

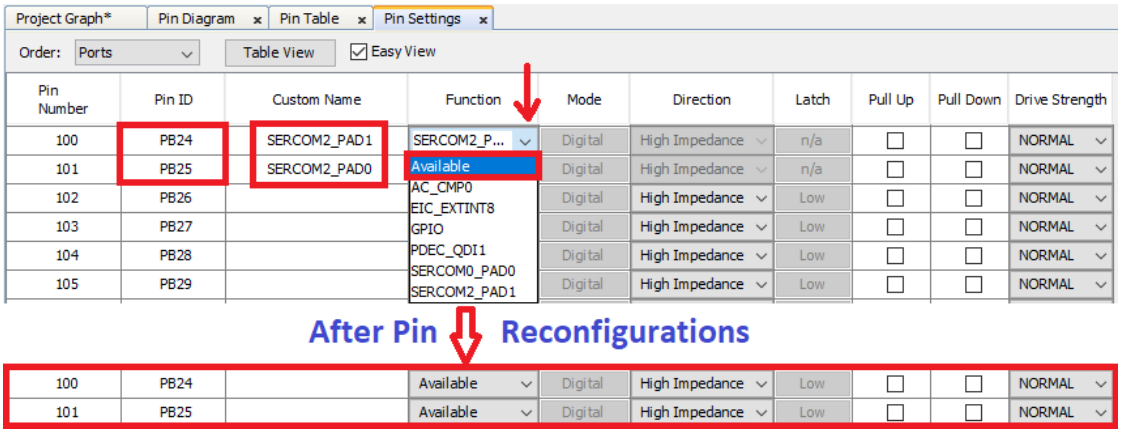
Figure 4-3. SERCOM1 RX Pinout Configuration



d. Launch the Pin Configurations plugin from *Project Graph > Plugins > Pin Configurations*.

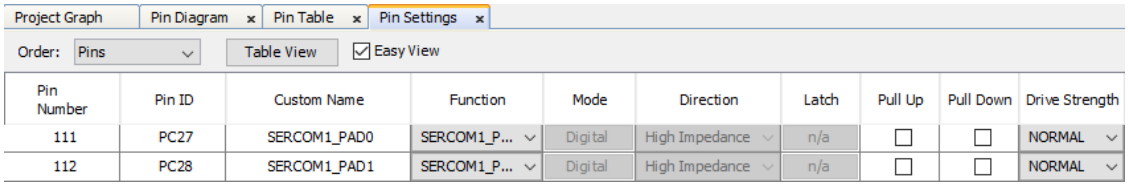
e. Remove the SERCOM2 pin configurations.

Figure 4-4. Bootloader MCC Reconfiguration - Remove SERCOM2 Pin Configurations



f. Configure the SERCOM1 pins.

Figure 4-5. Bootloader MCC Reconfiguration - Configure SERCOM1 Pins

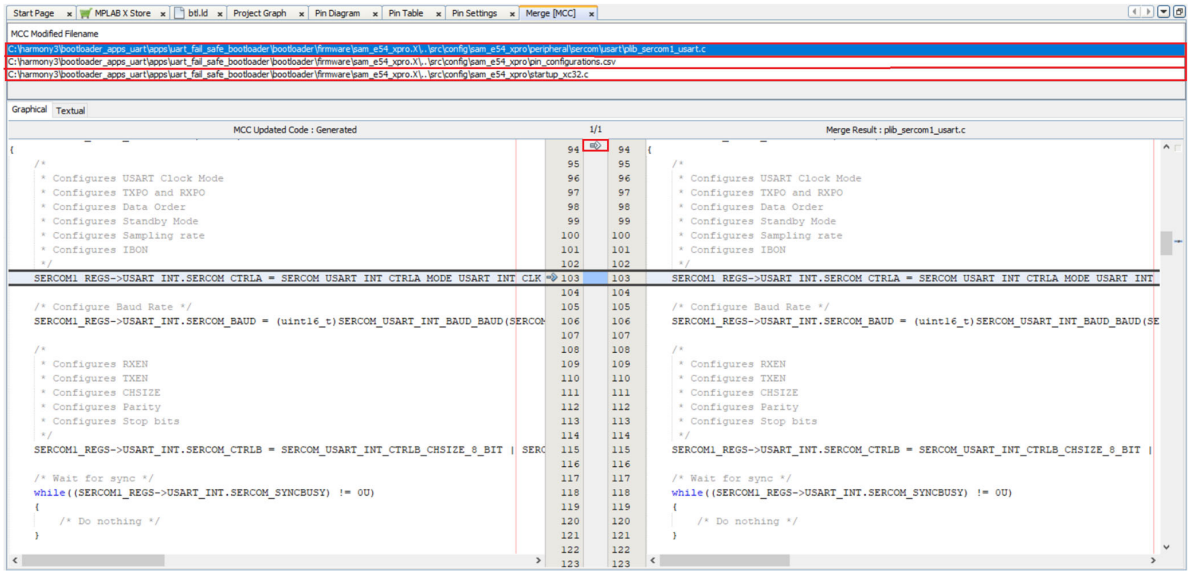


Notes:

- The SAM E54 Curiosity Ultra Development Board uses the SERCOM1 peripheral for the EDBG. Therefore, the SERCOM peripheral and its pins must be reconfigured.
- Do not close MCC as the project will be rebuilt

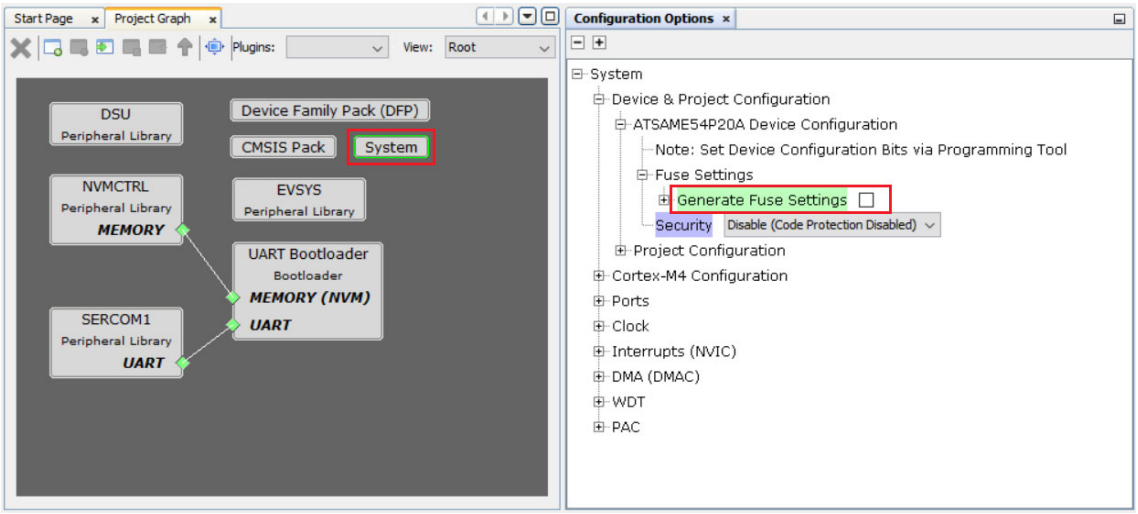
7. Regenerate the project. Build and program the UART Fail-Safe Bootloader using the MPLAB X IDE.
8. Before regenerating the new code, the MCC shows the files with the changes that are going to be done as the result of the reconfiguration. Click on the Replace All icon to replace all the changes.

Figure 4-6. MCC Regeneration - Merge [MCC]



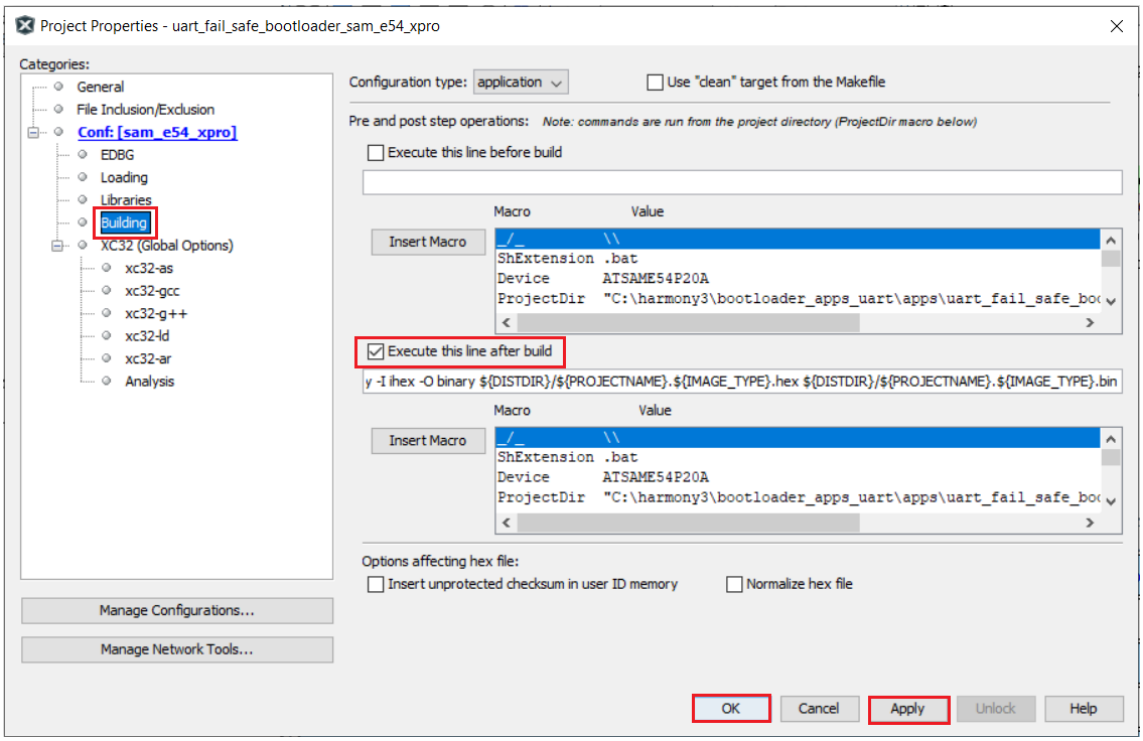
9. Follow these steps to rebuild the UART Fail-Safe Bootloader using MPLAB X IDE. This step is to create the bootloader binary image which will be merged with the Live Update application to create a single-binary image as shown in Step 9.
   a. Remove the Device Fuse configurations from the custom linker script as it will be updated by the UART Fail-Safe Bootloader project.

Figure 4-7. Bootloader MCC Reconfiguration - Remove Fuse Configurations



   b. In the Project Properties window, choose Building, and then select Execute this line after build in the MPLAB X IDE, as shown below.

Figure 4-8. Bootloader MCC Reconfiguration - Enable Build Option



10. Build the UART Fail-Safe Bootloader application again using the MPLAB X IDE, but do not program.
11. Download and build the Live Update Application (path: *<Live Update application folder>/firmware/sam_e54_cult.X*) using the MPLAB X IDE, but do not program.
12. Run the `btl_app_merge_bin.py` script from a command prompt to merge the generated bootloader binary and Live Update application binary.

```
python "<Harmony folder>\bootloader\tools\btl_app_merge_bin.py" -o 0x2000 -b "<Harmony
folder>\bootloader_apps_uart\apps\uart_fail_safe_bootloader\bootloader\firmware\sam_e54_xpr
o.X\dist\sam_e54_xpro\production\sam_e54_xpro.X.production.bin" -a "<Live Update
application folder>\same54_uart_live_update
\firmware\sam_e54_cult.X\dist\sam_e54_cult\production\sam_e54_cult.X.production.bin"
```

    a. The following output must be displayed on the command prompt:

Figure 4-9. Bootloader Binary Merge Result

```
##### Merged Bootloader and Application binaries to  btl_app_merged.bin #####
```

13. Run the `btl_host.py` from a command prompt to program the merged binary to the inactive bank on the dual-bank Flash. The merged binary `btl_app_merged.bin` will be generated in the path from where the `btl_app_merge_bin.py` was called from.

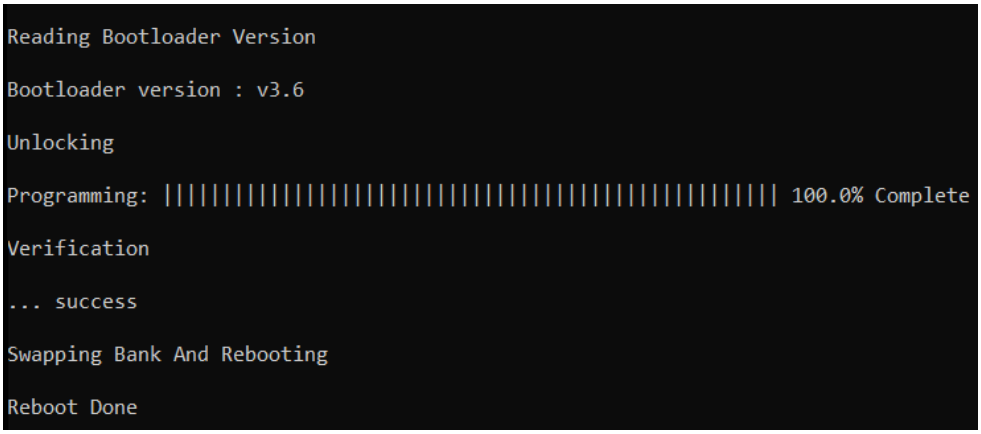
```
python <Harmony folder>/bootloader/tools/btl_host.py -v -s -i <COM PORT> -d same5x -a
0x2000 -f btl_app_merged.bin
```

Note:

- For additional information on the bootloader Host script, refer to the help for setting up the Host script available at: MPLAB Harmony Bootloader Help and *<Harmony folder>/bootloader_apps_uart/docs/index.html*

14. The following figure shows the successful programming of the application binary.

Figure 4-10. Bootloader Host Script Logs for Application Binary Programming



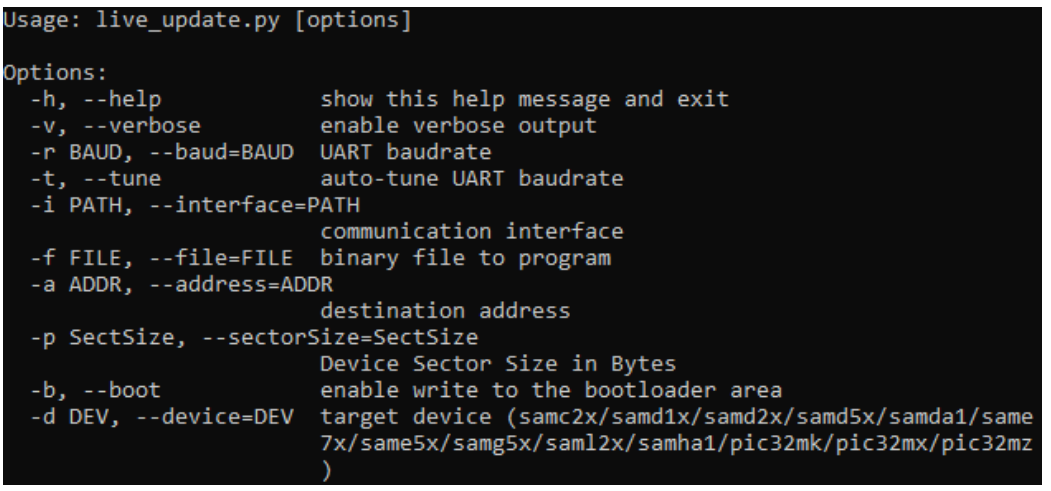
4.2 Running the Live Update Application

1. Follow Running the Bootloader application steps and download the Live Update Application, if not done already.
2. If the above step is successful, then LED1 must start blinking on the SAM E54 Curiosity Ultra Development Board.
3. Run the Host script `live_update.py` from the command prompt available at *<Live Update application folder>/scripts* to program the new version of the firmware to the inactive panel while the current version of the application is being run from the active panel.

```
python <Live Update application folder>\scripts\live_update.py -v -i <COM PORT> -d
same5x -a 0x80000 -f <path_of_btl_bin_file>\btl_app_merged.bin.
```

   a. The merged binary `btl_app_merged.bin` will be generated in the path from where the `btl_app_merge_bin.py` was called from.
   b. The following figure shows the Live Update script help.

Figure 4-11. Live Update Application Host Script Help Window



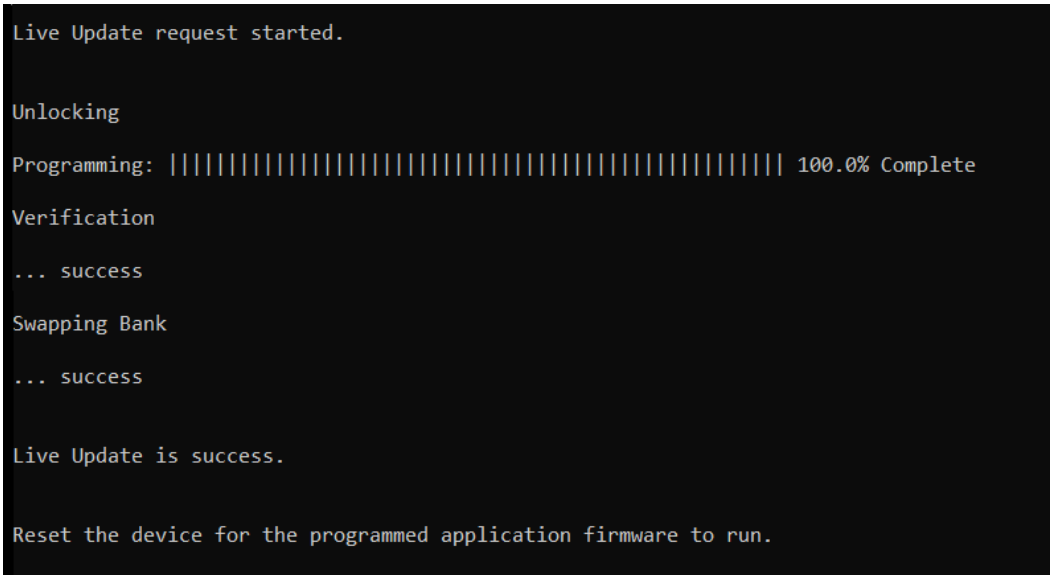

```
Command: python <python script> -v -s -i <COM PORT> -d <Device Name> -a <Address> -f
<live_update_application_image>
```

```
<python script>: live_update.py
<COM PORT>: Serial Communication Port
<Device Name>: same5x
<Address>: Application start address 0x9D100000
Example: python < Live update application folder \scripts\ live_update.py -v -s -i
COM6 -d same5x -a 0x80000 -f sam_e54_cult.X.production.bin
Note: Running the help command provides a brief overview of options available as shown
below.
Command: python <python script> --help
<python script>: live_update.py
```

Note: For additional information on the bootloader Host script, refer to the help for setting up the Host script available at: MPLAB Harmony Bootloader Help.

4. The following figure shows the Live Update firmware upgrade output.

Figure 4-12. Live Update Application Firmware Upgrade Output



5. LED1 stops blinking and LED2 starts blinking which indicates the application programming is successful.
   a. LED1 stops blinking and LED2 toggles for every 500 ms then application is running from Bank A.
   b. LED1 stops blinking and LED2 toggles for every 1000 ms then application is running from Bank B.
6. Press the Switch SW2 to swap the bank and reset the device for the programmed application firmware to run.
   a. LED2 stops blinking and LED1 toggles for every 500 ms then application is running from Bank A.
   b. LED2 stops blinking and LED1 toggles for every 1000 ms then application is running from Bank B.
7. In case of any error during the Live Update, both LED1 and LED2 will be turned ON. Use the following steps to recover from the error.
   a. Press and hold the Switch SW2 to reset the system, and then repeat step 4 to program the Live Update application.
   Or
   b. Reset or Power cycle the device and repeat from step 4 to program the Live Update application.

5. Conclusion

This document describes the implementation of a Live Update application. The Live Update application enables enhancing application functionality, and making it easier to introduce new features while the device continues to run the existing version of the application software. It allows devices to stay longer in the field and helps lower maintenance cost of the product. It also enables the scheduling of periodic and need based updates helping easier deployment of the product.

This document can be used as a reference to implement the Live Update feature in the user application.

The Live Update Task runs in Interrupt mode to update the firmware at run time; therefore users must ensure that the Application Task does not poll continuously whenever there is a request for a firmware update. Otherwise this leads to data loss or timeout while receiving the firmware from the Host PC.

6. References

The following documents are used as reference. For additional information, visit the Microchip website or contact a local Microchip sales office.

- MPLAB® Harmony v3:
  www.microchip.com/mplab/mplab-harmony
- For additional information about 32-bit Microcontroller Collaterals and Solutions, refer to:
  ww1.microchip.com/downloads/aemDocuments/documents/MCU32/ProductDocuments/ReferenceManuals/32-bit-Microcontroller-Collateral-and-Solutions-Reference-Guide-DS70005534.pdf
- Live Update Application on SAM E54 Curiosity Ultra Development Board:
  github.com/Microchip-MPLAB-Harmony/reference_apps/blob/master/apps/sam_e54_cult/same54_uart_live_update/readme.md
- SD Card USB Audio Player on SAM E54 Curiosity Ultra Development Board + maXTouch® Curiosity Pro Board using Legato Graphics:
  github.com/Microchip-MPLAB-Harmony/reference_apps/tree/v1.5.0/apps/sam_e54_cult/same54_sdcard_usb_audio_player
- MPLAB Harmony v3 Bootloader Help:
  < Harmony framework download folder>\bootloader\docs\index.html
- MPLAB Harmony v3 Core Help:
  microchip-mplab-harmony.github.io/core/GUID-C04D97AB-D6E0-4CF5-9A80-CA64E36B6199.html
- MPLAB Harmony v3 Bootloader application:
  < Harmony framework download folder>\bootloader\apps
- Getting Started with MPLAB Harmony v3 Peripheral Libraries on SAM D5x/E5x MCUs:
  developerhelp.microchip.com/xwiki/bin/view/software-tools/harmony/same54-getting-started-training-module/
- Dual-Bank Bootloader on SAM E54 Microcontroller (MCU) Using MPLAB Harmony v3
  ww1.microchip.com/downloads/aemDocuments/documents/MCU32/ApplicationNotes/ApplicationNotes/AN3508-Dual-Bank-Bootloader-on-SAM-E54-Microcontroller-Using-MPLAB-Harmony-v3-DS00003508.pdf
- MPLAB® Harmony v3 Reference Applications:
  github.com/MicrochipTech/MPLAB-Harmony-Reference-Apps
- MPLAB® Harmony v3 Getting Started Articles and Other Documents:
  www.microchip.com/mplab/mplab-harmony/mplab-harmony-articles-and-documentation
- MPLAB® Code Configurator Overview:
  microchipdeveloper.com/xwiki/bin/view/software-tools/mcc/#HGettingStartedwithMCC
- Create Your First Motor Control Application Using MPLAB® Harmony v3:
  developerhelp.microchip.com/xwiki/bin/view/software-tools/harmony/motor-control-getting-started-training-module/
- How to Build an Application by Adding a New PLIB, Driver, or Middleware to an Existing MPLAB Harmony v3 Project
  ww1.microchip.com/downloads/en/DeviceDoc/How_to_Build_Application_Adding_PLIB_%20Driver_or_Middleware%20_to_MPLAB_Harmony_v3Project_DS90003253A.pdf
- MPLAB® Harmony v3 SD Card Audio Player/Reader Tutorial:
  developerhelp.microchip.com/xwiki/bin/view/software-tools/harmony/archive/audio-player/
- Graphics Quick Start Applications for PIC32MZ and SAM MCUs:

github.com/Microchip-MPLAB-Harmony/gfx

- MPLAB Harmony v3 USB Stack:
  microchip-mplab-harmony.github.io/usb/frames.html?frmname=topic&frmfile=index.html
- Create Your First USB Device CDC Single Application:
  microchip-mplab-harmony.github.io/usb_apps_device/apps/cdc_com_port_single/readme.html
- Create Your First USB Host MSD Application:
  microchip-mplab-harmony.github.io/usb_apps_device/apps/msd_basic/readme.html
- MPLAB Harmony v3 TCP/IP Help:
  microchip-mplab-harmony.github.io/net/GUID-D4AB047B-AA57-433C-9975-88FC7E7798B0.html
- Create Your First TCP/IP Application:
  microchip-mplab-harmony.github.io/net/GUID-60A0C5B2-638B-4B28-9F43-745322863B88.html
- MPLAB Harmony v3 Application Development Guide for the MPLAB Harmony v2 users:
  ww1.microchip.com/downloads/en/Appnotes/
  MPLAB_Harmonyv3_Application_Development_%20Guide_for_%20MPLAB_Harmonyv2_Users_DS
  00003388A.pdf

7. Revision History

Revision B - 06/2024

The document was updated throughout to reflect the change to the new MCC tool from the older MHC tool.

The following updates were performed for this revision:

- Updates to software versions were applied for:
  - MPLAB® X Integrated Development Environment (IDE) and XC Compilers
  - MPLAB Harmony v3
  - Python
- New Images, updated code, and processes were implemented for:
  - Bootloader Linker Script
  - MCC Configuration
  - Project Settings
  - Running the Bootloader Application
  - Running the Live Update Application
- The References section was updated with all new links reflecting the updated tools

Revision A - 12/2020

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: 978-1-6683-4695-2

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