TB3190

How to Customize ASFv3 SAM-BA Bootloader on Cortex-M0+ Microcontrollers

Introduction

The SAM Boot Assistant (SAM-BA™) allows In-System Programming (ISP) from the USB or a UART host without any external programming interface. Typically, in ROM-based SAM MCUs, the internal ROM contains the SAM-BA monitor firmware, whereas all the Cortex®-M0+ based MCUs do not have an internal ROM.

The SAM-BA support for Cortex-M0+ based devices is provided by loading the SAM-BA monitor firmware into the Flash memory. The SAM-BA monitor firmware acts as a bootloader, which can accept commands from the SAM-BA Host/GUI. The SAM-BA GUI sends the SAM-BA applet to the SAM-BA monitor and the SAM-BA monitor loads the applet firmware in SRAM. The SAM-BA applet is a firmware, which runs on SRAM to process the SAM-BA commands received by the SAM-BA monitor.

The SAM-BA bootloader firmware is available in the ASFv3 as standalone examples.

Note:
To open the SAMD21 SAM-BA bootloader, users can open the sample project from Atmel Studio by performing these actions: From the *File* > *New* > *Example project* > and then select the Device Family as SAMD21 and choose the SAM0 SAM-BA bootloader Example – *SAMD21 Xplained Pro* project.

This document is focused on the ASFv3-based SAM-BA bootloader.

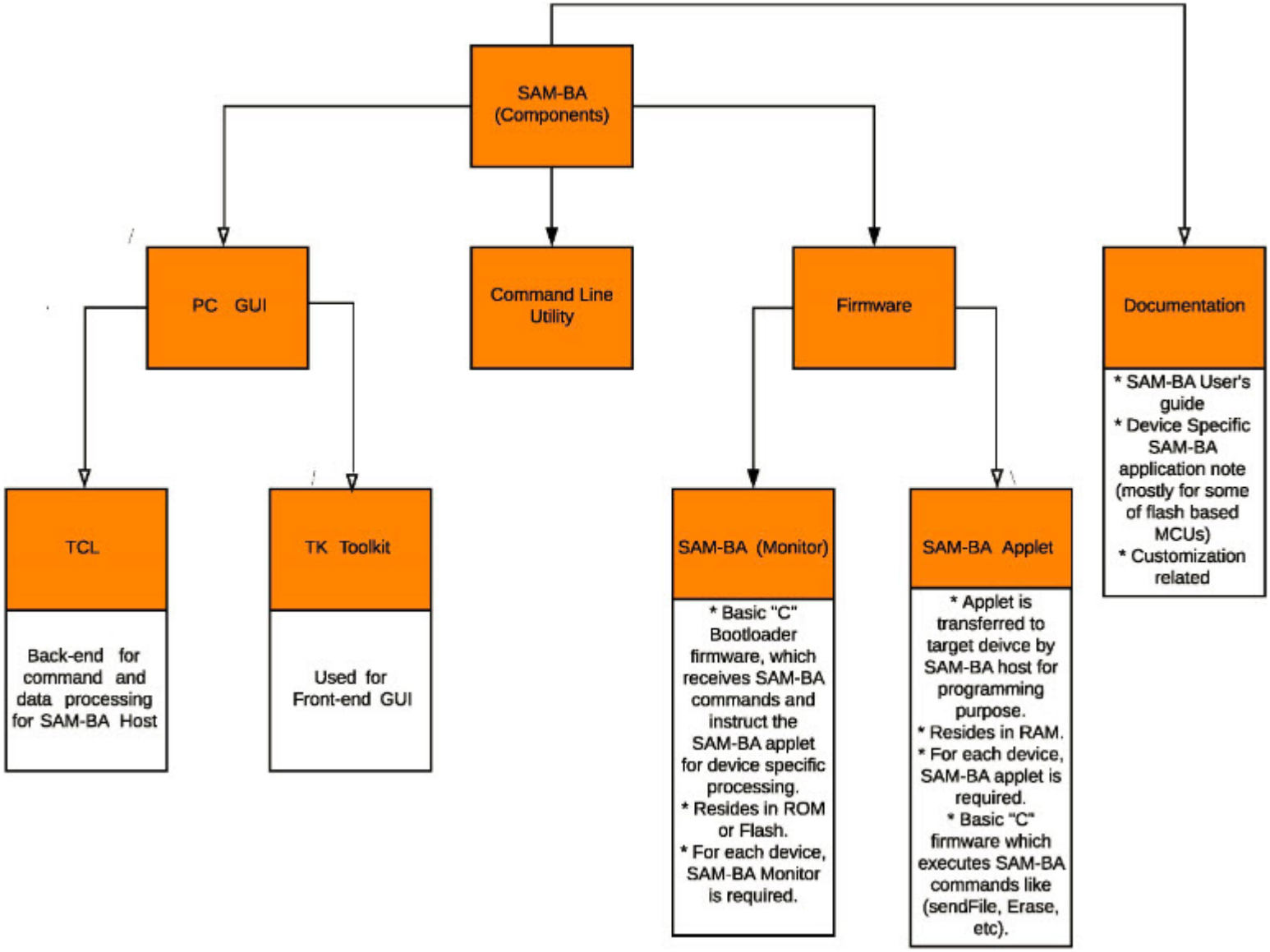
Table of Contents

1. Concept

For the SAM-BA GUI to program and Flash the Cortex-M0+ based MCU, the following are required:

- The SAM-BA bootloader firmware needs to be Flashed on the Cortex-M0+ based MCU. This firmware will only be able to get the request from the SAM-BA GUI and process the supported commands.
- The SAM-BA GUI and its components.
- The TCL script, which performs the basic validation of the MCU chosen, and aids with the SAM-BA GUI in the execution of applet commands.
- The SAM-BA applet, which performs the actual firmware request made from the SAM-BA GUI.

Figure 1-1. SAM-BA Block Diagram



This document will explain the available customization on the SAM-BA bootloader firmware, using the applet and TCL script with an example.

2. Solution or Implementation

In Cortex-M0+ based SAM-BA solutions, the existing solutions (the SAM-BA bootloader source, applets and so on) are provided only for the Xplained Pro boards. Instructions on how to use the SAM-BA for a user board, and a device other than the device used with the Xplained Pro board is discussed in the following section.

Customizing the SAM-BA to use a different Serial Communication Interface (SERCOM) for SAMD21E16B (from SAMD21J18A Xplained Pro)

Follow these steps to customize a ASF SAMD21 Xplained pro-based project for the SAMD21E16B using a user board in Atmel Studio 7.0:

1. Once the SAM0 SAM-BA bootloader example is loaded in Atmel Studio, from the *Project* > *Properties* > *Device* option of the ASF project, the user needs to change the device to select the SAMD21E16B.
2. From the *Project* > *Properties* > *Toolchain* > *ARM/GNU C Compiler* >*Symbols* option of the ASF project, the user needs to replace the existing definition of __SAMD21J18A__ with the __SAMD21E16B__ .
3. The ASF recognizes the boards developed by Microchip (for example, the Xplained Pro board). The boards which are developed using SAM products are recognized by ASF as user boards. The user needs to modify the settings of the project to indicate they are developing a user board-based project. From the *Project* > *Properties* > *Toolchain* > *ARM/GNU C Compiler* > *Symbols* option of the project, remove the existing definition of *BOARD* (from the SAMD21 Xplained Pro) and add a new definition as *BOARD=USER_BOARD.*
4. Add a new file with the name `user_board.h` in the `src\ASF\sam0\boards` folder. This file should contain all the required components and pin definitions of the user board, similar to the `samd21_xplained_pro.h`.
5. Copy the `board_init.c` file from the `ASF\sam0\boards\samd21_xplained_pro` folder to the `ASF\sam0\boards` folder.
6. Remove the folder `ASF\sam0\boards\samd21_xplained_pro` from the project.
7. The linker script must be updated to match for the SAMD21E16B. Typically, the RAM, Flash and default stack size option in the linker script must be modified to match the device.
8. All other customization in the ASF bootloader is based on project needs, that is, adding or removing an LED, and so on (optional).

Note:
Users can replace the linker script for the SAMD21J18A with the linker script of SAMD21E16B. The linker script is available in the `src\ASF\sam0\utils\linker_scripts\samd21\gcc` folder for the default project created (i.e. SAMD21J18A). The linker script for the SAMD21E16B can be obtained by creating a new project for the SAMD21E16B from Atmel Studio.

Customizing ASF SAM-BA bootloader configurations

The following are the available configurations in the ASF SAM-BA bootloader:

1. Clock configurations – Refer to the following Microchip Knowledge article for different clock configuration options.
2. Bootloader configurations – Refer to the following table for different bootloader specific configurations. Users should not modify all other available configurations in the bootloader configuration file `conf_bootloader.h`.

Table 2-1. Available Configurations for the SAM-BA Bootloader

S.No	Configuration Name	Configuration Value	Remarks
1	APP_START_ADDRESS	Based on the bootloader firmware size, this value can be set.	For example, 0x6000
2	BOOT_LOAD_PIN	User defined GPIO pin for Button access	Used to enter the Bootloader mode if pressed. For example, PIN_PA11
3	BOOT_LED	User defined GPIO pin for LED access	Used only if DEBUG_ENABLE is enabled. For example, PIN_PA10
4	DEBUG_ENABLE	Used to display notification of whether the device is in Bootloader mode or Application mode.	For example, false
5	BOOT_USART_MODULE	Used to enter the SERCOM instance used by the bootloader	For example, SERCOM0
6	BOOT_USART_MUX_SETTINGS	User defined SERCOM MUX settings	For example, USART_RX_1_TX_0_XCK_1
7	BOOT_USART_PADx	User defined SERCOM TX/RX settings	For example, PINMUX_PA08C_SERCOM0_PAD0
8	BOOT_USART_PADy	User defined SERCOM TX/RX settings	For example, PINMUX_PA08C_SERCOM0_PAD1
9	BOOT_USART_GCLK_SOURCE	User defined clock source	For example, GCLK_GENERATOR_0

S.No	Configuration Name	Configuration Value	Remarks
		of SAM-BA bootlaoder	
10	CONF_USBCDC_INTERFACE_SUPPORT	User defined bootloader option	This configuration will be defined if the USB bootloader option needs to be enabled.

Customizing ASF SAM-BA TCL Script

1. From the SAM-BA installation folder, select the *\tcl_lib\samd21_xplained_pro \samd21_xplained_pro.tcl* file and then update the *devicesList* variable to include the SAMD21E16B variant in the list of devices supported.
2. Based on the device selected and the SRAM occupied by the SAM-BA bootloader, update the appletAddr and appletMailboxAddr accordingly.

Customizing ASF SAM-BA Applet Linker script for SAMD21E16B and latest ASF

1. From the SAM-BA installation folder, select the *applets\samd21j18a\sam-ba_applets\linker_script \sram_samba.lds* file and update the sections which are based on the device chosen, and the SRAM size occupied by the SAM-BA bootloader.

Customizing ASF SAM-BA Applet Makefile for SAMD21E16B and latest ASF

1. From the SAM-BA installation folder, select the *applets\samd21j18a\sam-ba_applets\flash\Makefile.*
2. Update the CHIP_NAME to the name of the new device, for example, __SAMD21E16B__ and update the ASF_BRANCH_PATH to the latest ASF standalone directory.

The following figure displays the changes to be made in the Makefile to use the SAMD21E16B for the ASF 3.36.2 or later ASFv3 version.

Note: This is also applicable for other versions of ASF.

Figure 2-1. Changes to be Made in the Makefile to Use SAMD21E16B for ASF 3.36.2

```
ifeq (samd21j18a,$(CHIP))
BOARD_DIR = samd21_xplained_pro
CHIP_NAME = __SAMD21E16B__
endif

MEMORIES = sram

# Trace level used for compilation
# (can be overriden by adding TRACE_LEVEL=#number to the command-line)
47 FILTERED LINES
LD = $(CROSS_COMPILE)ld
SIZE = $(CROSS_COMPILE)size
OBJCOPY = $(CROSS_COMPILE)objcopy
OBJDUMP = $(CROSS_COMPILE)objdump

# Flags
ASF_BRANCH_PATH = D:\TEMP_ASF\asf-standalone-archive-3.36.2.65\xdk-asf-3.36.2
#GCC_LIB_PATH = $(ASF_BRANCH_PATH)/thirdparty/CMSIS/Lib/GCC
#INCLUDES = -I$(PATH_ATML_LIB_BOARD)
#INCLUDES += -I$(PATH_ATML_LIB_BOARD)/include
#INCLUDES += -I$(PATH_ATML_LIB_BOARD)/source
#INCLUDES += -I$(PATH_ATML_LIB_CHIP)
#INCLUDES += -I$(PATH_ATML_LIB_CHIP)/include
12 FILTERED LINES
INCLUDES += -I$(ASF_BRANCH_PATH)/sam0/drivers/system/interrupt
INCLUDES += -I$(ASF_BRANCH_PATH)/sam0/drivers/nvm
INCLUDES += -I$(ASF_BRANCH_PATH)/thirdparty/CMSIS
INCLUDES += -I$(ASF_BRANCH_PATH)/thirdparty/CMSIS/Include
#INCLUDES += -I$(ASF_BRANCH_PATH)/thirdparty/CMSIS/Lib
#INCLUDES += -I$(ASF_BRANCH_PATH)/thirdparty/CMSIS/Lib/GCC
INCLUDES += -I$(ASF_BRANCH_PATH)/sam0/drivers/system/reset/reset_sam_d_r_h
INCLUDES += -I$(ASF_BRANCH_PATH)/sam0/drivers/system/interrupt/system_interrupt_samd21
INCLUDES += -I$(ASF_BRANCH_PATH)/sam0/drivers/system/clock/clock_samd21_r21_da_ha1
INCLUDES += -I$(ASF_BRANCH_PATH)/sam0/drivers/system/power/power_sam_d_r_h
```

Note: The Included path dependencies can be modified, which are based on the ASF version used.

Customizing ASF SAM-BA Applet Source for SERCOM0

1. In the *applet_main method >sam-ba installation >\applets\samd21j18a\sam-ba_applets\flash\ flash_app_main.c*, modify the code to use SERCOM0, instead of SERCOM3.

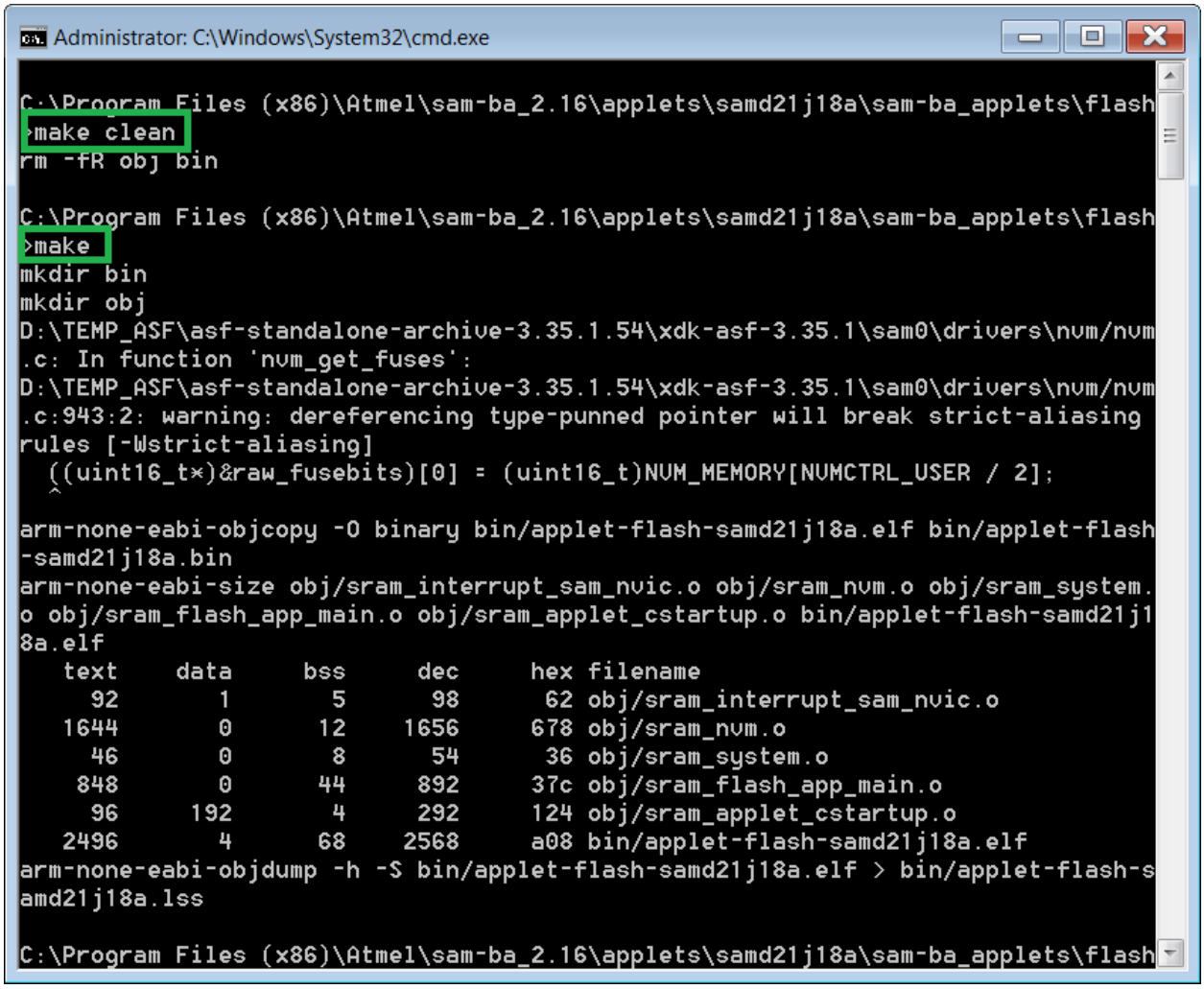
   Figure 2-2. Modifying the Code

```
exit:
    /* Acknowledge the end of command */
    TRACE_INFO("\tEnd of Applet %x %x.\n\r",
               (uint32_t)pMailbox->command,
               (uint32_t)pMailbox->status);
    /* Notify the host application of the end of the command processing */
    pMailbox->command = ~(pMailbox->command);

    SERCOM0->USART.DATA.reg = 0x6;
    return 0;
}
```

2. Open the Atmel Studio 7.0 command prompt, and move to the folder location *sam-ba installation* > *\applets\samd21j18a\sam-ba_applets\flash* and then execute the commands highlighted in the green box as shown in the following figure.

Figure 2-3. Opening Atmel Studio



Known Issues:

- The ASF SAM-BA bootloader for Cortex-M0+ is not optimized for memory constraint variants of the SAM Cortex-M0+ family and therefore may not work for SRAM < 8KB.
- The SAM-BA applets, TCL scripts, and monitor firmware are not available for some of the devices (for example, SAMD21E16B) in SAM-BA v2.18 release.
- When downloading files of considerable size (i.e., > 32KB), the SAM-BA GUI may appear to hang, but it is processing the request in the background.
- Some versions of GCC compilers may have issues concerning optimization, causing the SAM-BA GUI to hang, such as GCC 6.3.1.508.

Tips and Tricks:

- USB-Serial converter: The EDBG chip on the Xplained Pro boards acts as a full speed USB-Serial Bridge, therefore to have similar performance, users need to use a full speed USB-Serial converter.

3. Other Relevant Resources

For further information on the SAM-BA bootloader, refer to the following documents available for download form the following locations:

- http://www.microchip.com/avr-support/advanced-software-framework-(asf)
- http://asf.atmel.com/docs/latest/sam0.applications.samba_bootloader.samc21_xplained_pro/html/index.html
- http://www.microchip.com/DevelopmentTools/ProductDetails.aspx?PartNO=Atmel%20SAM-BA%20In-system%20Programmer
- https://microchip.secure.force.com/microchipknowledge/articles/en_US/FAQ/How-to-change-main-clock-source-for-SAMR21-SAMD21-SAMD20-SAMD10-SAMD11-in-ASF-project
- http://ww1.microchip.com/downloads/en/AppNotes/Atmel-42366-SAM-BA-Bootloader-for-SAM-D21_ApplicationNote_AT07175.pdf

4. Frequently Asked Questions

How are unused Flash locations filled with known data while building the project?

Sometimes it is required to fill unused Flash locations with known data. The following steps illustrate one way of achieving this.

For ARM devices:

1. Create a dummy section in the Application. This is required as the linker needs a section to fill data.

```
/** Creating a dummy section to fill unused flash with known data (0xFF) */
       const U8 u8Dummy __attribute__ ((section(".fill_known_data"))) = 0xFF;
```

2. Fill this section with known data using the linker script.
   – Let linker relocate the initialized data.

```
.relocate :
               {
                       . = ALIGN(4);
                       _srelocate = .;
                       *(.ramfunc .ramfunc.*);
                       *(.data .data.*);
                        . = ALIGN(4);
                         _erelocate = .;
               } > ram AT > rom
```

   In the default script, find .relocate : AT (_etext), and change that to match the given code.

   – Fill the section with FF till the end of the rom

```
.Fill_FF :
               {
                       _FillStartAddress = .; /* Provides start address of Filled
location */
                       KEEP(*(.fill_known_data))
                       FILL(0xFF);
                       . = ORIGIN(rom) + LENGTH(rom) - _FillStartAddress; /* Fill
till end of rom */
                        _FillEndAddress = .; /* Provides end address of Filled
location */
               } AT > rom
```

In the previous example, unused locations are filled with 0xFF. By changing 0xFF to another value, unused locations can be filled with known data.

Why does my application use so much RAM?

The linker scripts allocate memory for the stack on startup. The size of this allocation is defined in the linker script. By default, this is a rather large value. This is done so that the user should not run into stack overflow issues easily, as this kind of issue is hard to debug. Every application should configure this value, for optimum memory use.

The Microchip Web Site

Microchip provides online support via our web site at http://www.microchip.com/. This web site is used as a means to make files and information easily available to customers. Accessible by using your favorite Internet browser, the web site contains the following information:

- **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 (FAQ), technical support requests, online discussion groups, Microchip consultant 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

Customer Change Notification Service

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

To register, access the Microchip web site at http://www.microchip.com/. Under "Support", click on "Customer Change Notification" and follow the registration instructions.

Customer Support

Users of Microchip products can receive assistance through several channels:

- Distributor or Representative
- Local Sales Office
- Field Application Engineer (FAE)
- Technical Support

Customers should contact their distributor, representative or Field Application Engineer (FAE) for support. Local sales offices are also available to help customers. A listing of sales offices and locations is included in the back of this document.

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

Microchip Devices Code Protection Feature

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

- Microchip products meet the specification contained in their particular Microchip Data Sheet.
- Microchip believes that its family of products is one of the most secure families of its kind on the market today, when used in the intended manner and under normal conditions.
- There are dishonest and possibly illegal methods used to breach the code protection feature. All of these methods, to our knowledge, require using the Microchip products in a manner outside the operating specifications contained in Microchip's Data Sheets. Most likely, the person doing so is engaged in theft of intellectual property.
- Microchip is willing to work with the customer who is concerned about the integrity of their code.

- Neither Microchip nor any other semiconductor manufacturer can guarantee the security of their code. Code protection does not mean that we are guaranteeing the product as “unbreakable.”

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

Legal Notice

Trademarks

ISBN: 978-1-5224-2834-3

Quality Management System Certified by DNV

ISO/TS 16949

Microchip received ISO/TS-16949:2009 certification for its worldwide headquarters, design and wafer fabrication facilities in Chandler and Tempe, Arizona; Gresham, Oregon and design centers in California and India. The Company's quality system processes and procedures are for its PIC® MCUs and dsPIC® DSCs, KEELOQ® code hopping devices, Serial EEPROMs, microperipherals, nonvolatile memory and analog products. In addition, Microchip's quality system for the design and manufacture of development systems is ISO 9001:2000 certified.

Worldwide Sales and Service

AMERICAS	ASIA/PACIFIC	ASIA/PACIFIC	EUROPE
Corporate Office 2355 West Chandler Blvd. Chandler, AZ 85224-6199 Tel: 480-792-7200 Fax: 480-792-7277 Technical Support: http://www.microchip.com/support Web Address: www.microchip.com **Atlanta** Duluth, GA Tel: 678-957-9614 Fax: 678-957-1455 **Austin, TX** Tel: 512-257-3370 **Boston** Westborough, MA Tel: 774-760-0087 Fax: 774-760-0088 **Chicago** Itasca, IL Tel: 630-285-0071 Fax: 630-285-0075 **Dallas** Addison, TX Tel: 972-818-7423 Fax: 972-818-2924 **Detroit** Novi, MI Tel: 248-848-4000 **Houston, TX** Tel: 281-894-5983 **Indianapolis** Noblesville, IN Tel: 317-773-8323 Fax: 317-773-5453 Tel: 317-536-2380 **Los Angeles** Mission Viejo, CA Tel: 949-462-9523 Fax: 949-462-9608 Tel: 951-273-7800 **Raleigh, NC** Tel: 919-844-7510 **New York, NY** Tel: 631-435-6000 **San Jose, CA** Tel: 408-735-9110 Tel: 408-436-4270 **Canada - Toronto** Tel: 905-695-1980 Fax: 905-695-2078	**Australia - Sydney** Tel: 61-2-9868-6733 **China - Beijing** Tel: 86-10-8569-7000 **China - Chengdu** Tel: 86-28-8665-5511 **China - Chongqing** Tel: 86-23-8980-9588 **China - Dongguan** Tel: 86-769-8702-9880 **China - Guangzhou** Tel: 86-20-8755-8029 **China - Hangzhou** Tel: 86-571-8792-8115 **China - Hong Kong SAR** Tel: 852-2943-5100 **China - Nanjing** Tel: 86-25-8473-2460 **China - Qingdao** Tel: 86-532-8502-7355 **China - Shanghai** Tel: 86-21-3326-8000 **China - Shenyang** Tel: 86-24-2334-2829 **China - Shenzhen** Tel: 86-755-8864-2200 **China - Suzhou** Tel: 86-186-6233-1526 **China - Wuhan** Tel: 86-27-5980-5300 **China - Xian** Tel: 86-29-8833-7252 **China - Xiamen** Tel: 86-592-2388138 **China - Zhuhai** Tel: 86-756-3210040	**India - Bangalore** Tel: 91-80-3090-4444 **India - New Delhi** Tel: 91-11-4160-8631 **India - Pune** Tel: 91-20-4121-0141 **Japan - Osaka** Tel: 81-6-6152-7160 **Japan - Tokyo** Tel: 81-3-6880- 3770 **Korea - Daegu** Tel: 82-53-744-4301 **Korea - Seoul** Tel: 82-2-554-7200 **Malaysia - Kuala Lumpur** Tel: 60-3-7651-7906 **Malaysia - Penang** Tel: 60-4-227-8870 **Philippines - Manila** Tel: 63-2-634-9065 **Singapore** Tel: 65-6334-8870 **Taiwan - Hsin Chu** Tel: 886-3-577-8366 **Taiwan - Kaohsiung** Tel: 886-7-213-7830 **Taiwan - Taipei** Tel: 886-2-2508-8600 **Thailand - Bangkok** Tel: 66-2-694-1351 **Vietnam - Ho Chi Minh** Tel: 84-28-5448-2100	**Austria - Wels** Tel: 43-7242-2244-39 Fax: 43-7242-2244-393 **Denmark - Copenhagen** Tel: 45-4450-2828 Fax: 45-4485-2829 **Finland - Espoo** Tel: 358-9-4520-820 **France - Paris** Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79 **Germany - Garching** Tel: 49-8931-9700 **Germany - Haan** Tel: 49-2129-3766400 **Germany - Heilbronn** Tel: 49-7131-67-3636 **Germany - Karlsruhe** Tel: 49-721-625370 **Germany - Munich** Tel: 49-89-627-144-0 Fax: 49-89-627-144-44 **Germany - Rosenheim** Tel: 49-8031-354-560 **Israel - Ra'anana** Tel: 972-9-744-7705 **Italy - Milan** Tel: 39-0331-742611 Fax: 39-0331-466781 **Italy - Padova** Tel: 39-049-7625286 **Netherlands - Drunen** Tel: 31-416-690399 Fax: 31-416-690340 **Norway - Trondheim** Tel: 47-7289-7561 **Poland - Warsaw** Tel: 48-22-3325737 **Romania - Bucharest** Tel: 40-21-407-87-50 **Spain - Madrid** Tel: 34-91-708-08-90 Fax: 34-91-708-08-91 **Sweden - Gothenberg** Tel: 46-31-704-60-40 **Sweden - Stockholm** Tel: 46-8-5090-4654 **UK - Wokingham** Tel: 44-118-921-5800 Fax: 44-118-921-5820