UART BOOTLOADER

1. Introduction

A bootloader enables field updates of application firmware. A Universal Asynchronous Receiver/Transmitter (UART) bootloader enables firmware updates over the UART. The UART bootloader described in this application note is based on the Silicon Labs Modular Bootloader Framework. This framework is described in detail in the application note “AN533: Modular Bootloader Framework for Silicon Labs C8051Fxxx Microcontrollers”, which can be downloaded from here: http://www.silabs.com/products/mcu/Pages/ApplicationNotes.aspx.

The following components are part of the firmware update setup:

- Target Bootloader Firmware
- Active Data Source Software
- Target Application Firmware

The firmware update setup is shown in Figure 1. Details about the steps involved in updating the firmware can be found in the Firmware Update Process Flow Diagram in application note, “AN533: Modular Bootloader Framework for Silicon Labs C8051Fxxx Microcontrollers”.

The code accompanying this application note is originally written for several specific devices, but can be ported to other devices in the Silicon Labs microcontroller range.

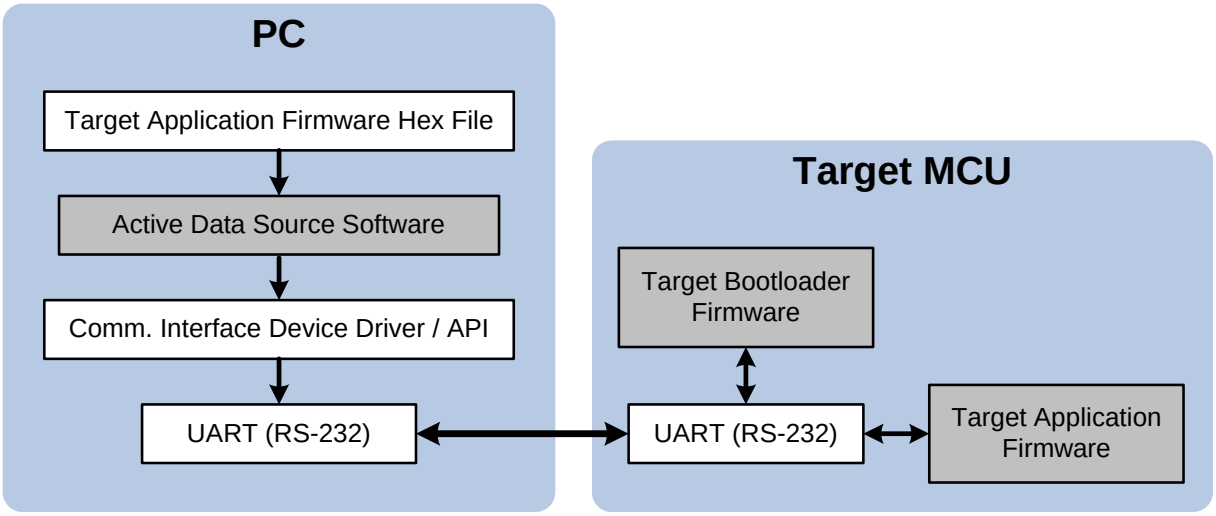


Figure 1. Firmware Update Setup

2. Getting Started

This section provides step-by-step instructions to use the included UART bootloader examples.

2.1. Preparation

Unzip the AN778SW software package to a location on memory. Each example contains:

- **DataSource_Software —** contains the PC application that communicates with the Target Bootloader and downloads the application image.
- **Sample_User_Application —** contains a simple example application project.
- **TargetBootloader —** contains the UART bootloader source using the Modular Framework.

2.2. Procedure

1. Go to the TargetBootloader directory and open the UART bootloader project (F33x_UART_TargetBL or another device project file).
2. Compile and download the bootloader code to a target board.
3. After downloading, power down the target board.
4. Run the **SerialBootloaderDataSource.exe** executable in the **DataSource_Software\SerialBootloaderDataSource\SerialBootloaderDataSource\bin\Release** directory.

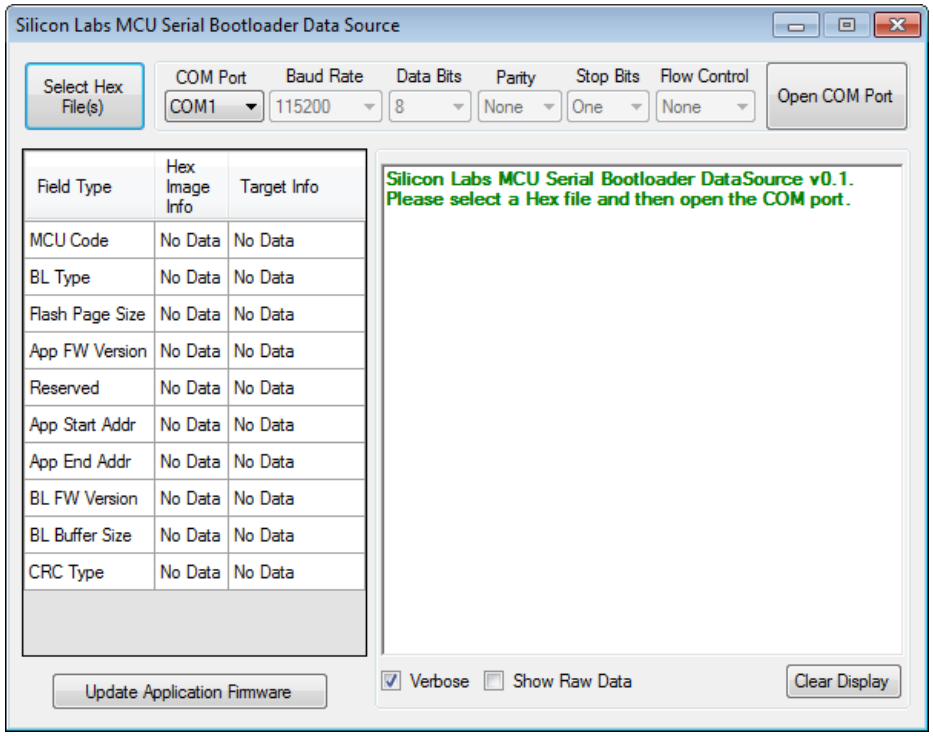


Figure 2. Running the Active Data Source Software

5. Choose a COM port in the application, select the application HEX file in the **Sample_User_Application** directory, and click the **Open COM Port** button.

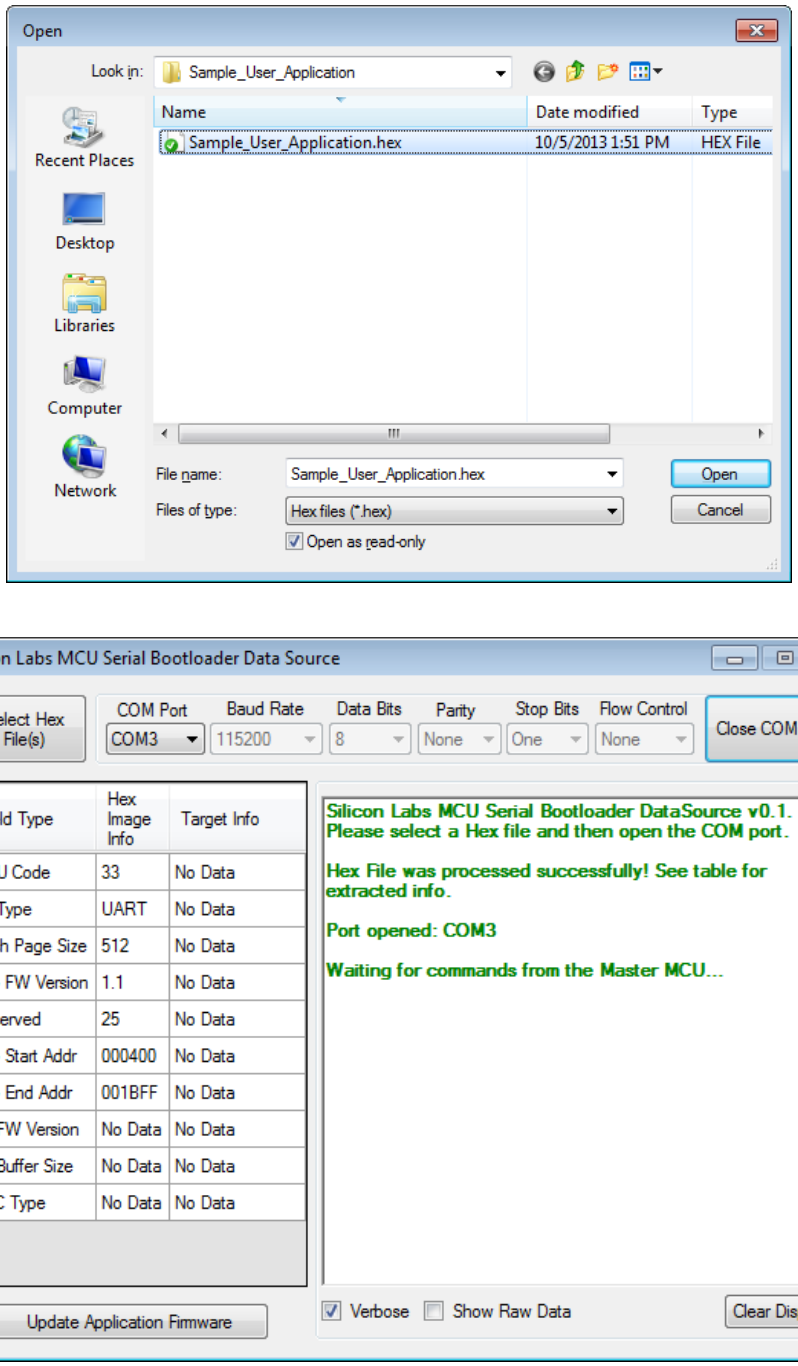


Figure 3. Opening the Target MCU COM Port

6. Power up the target board. The bootloader will print text in red in the application display window when it's ready to update the application code.

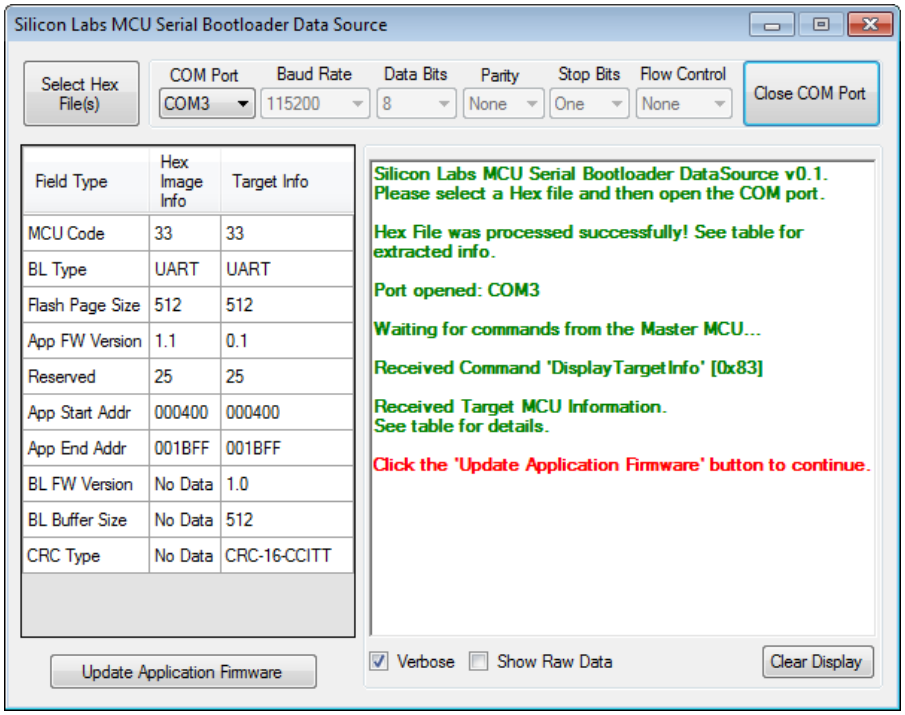


Figure 4. Target Bootloader Ready to Update Application Code

7. Click the **Update Application Firmware** button to download the selected application firmware to the target board.

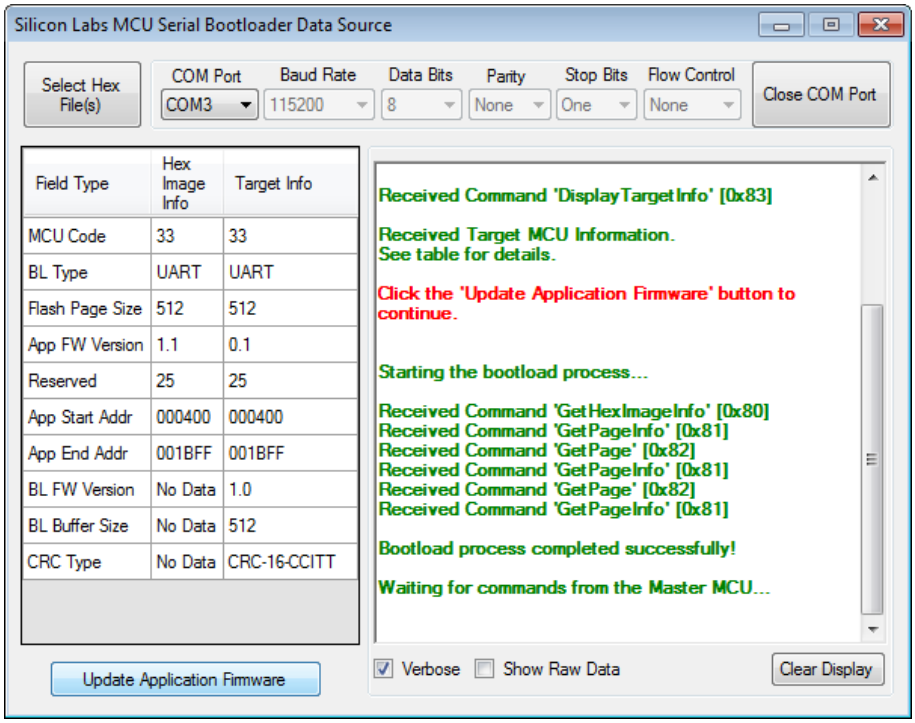


Figure 5. Downloading the Application Code

8. Once the download completes successfully, the bootloader will jump to the application code (Blinky).

3. Target Bootloader Profile

The UART target bootloader firmware allows application firmware updates in the field over UART. The UART bootloader code builds under the Keil toolchain. The bootloader firmware is stored in address 0x0000-0x0400 and last flash page. This means that the application firmware starts at address 0x0400 and ends one page short of the last flash page. Figure 6 shows the UART bootloader memory map. Figure 7 shows an example code space utilization of the bootloader grouped by functional blocks based on version 1.1 of the 'F330 bootloader firmware. For detailed information on the latest source code size, see the Excel file in the software package accompanying this document.

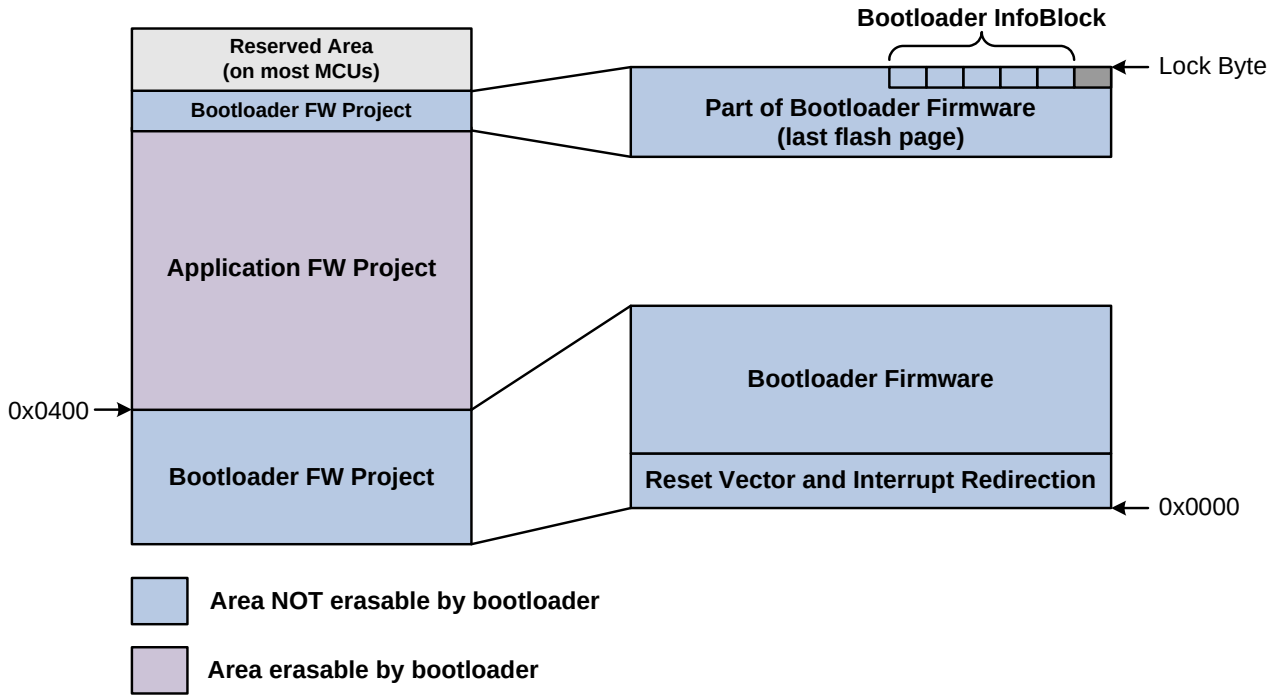


Figure 6. UART Target Bootloader Memory Map

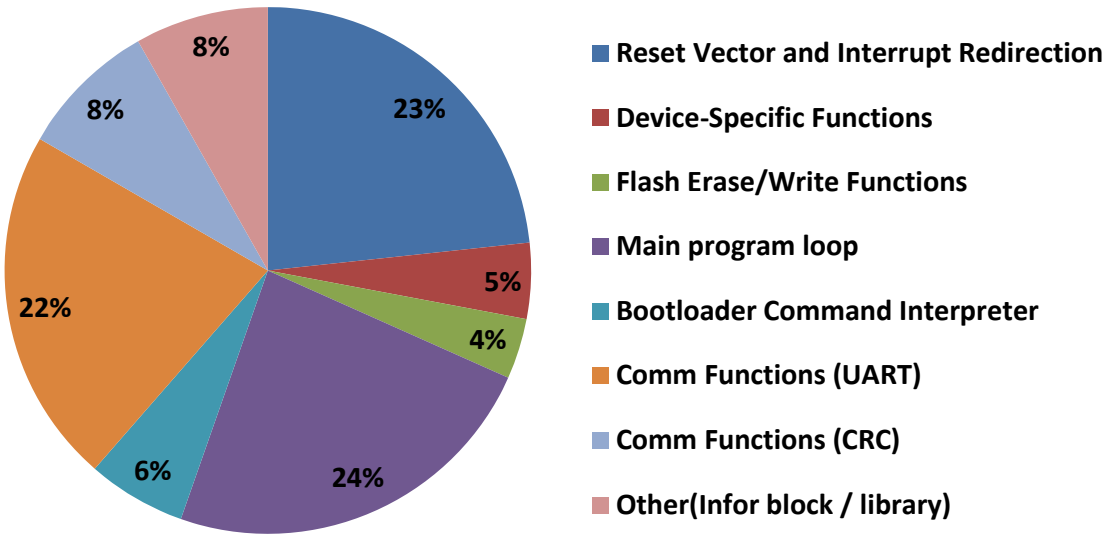


Figure 7. UART Target Bootloader Example Code Space Utilization Profile

3.1. Configurable Options

The target bootloader has the following parameters that can be configured. These parameters are located in two header files as grouped in Table 1 and Table 2.

Table 1. Fxxx_Target_Config.h

Parameter	Options
TGT_MCU_CODE[1]	Any 8-bit value
TGT_BL_TYPE[2]	8-bit value: **0x01**
TGT_FLASH_PAGE_SIZE[3]	Number of bytes per flash page: **512**
TGT_FLASH_PAGE_SIZE_CODE[4]	8-bit value: **1**
APP_FW_START_ADDR[5]	16-bit value: **0x0400**
APP_FW_END_ADDR[6]	16-bit value: **0x1BFF**

Notes:
1. This can be used to identify a product line among many different products.
2. This denotes that the BL uses Silicon Labs-defined UART bootloader protocol (see **Fxxx_BL129_UART_Interface.h**).
3. Should be changed based on the MCU data sheet.
4. 1 means 512 bytes/page; 2 means 1024 bytes/page.
5. Starting address of App FW.
6. Ending address of App FW (includes App InfoBlock and Signature bytes). It should be changed based on the size of Flash.

Table 2. Fxxx_TargetBL_Config.h

Parameter	Options
TGT_BL_FW_INFOBLOCK_LENGTH[1]	8-bit value: **19**
TGT_BL_FW_VERSION_LOW and TGT_BL_FW_VERSION_HIGH[2]	8-bit values: **1 and 1**
TGT_BL_BUF_SIZE_CODE[3]	8-bit value: **0**

Notes:
1. See Table 1 in application note, “AN533: Modular Bootloader Framework for Silicon Labs C8051Fxxx Microcontrollers”.
2. BL v1.1. Low = 1 and High = 1.
3. This valve determines the max packet size when the PC sends a page of data to the bootloader..
   The buffer size = page_size/(1<<buffer_size_code).

4. Target Application Profile

The target application firmware needs to fit within the allocated application area in flash memory. The application firmware memory map is shown in Figure 8.

Note: The application firmware starts at address 0x0400 in this example.

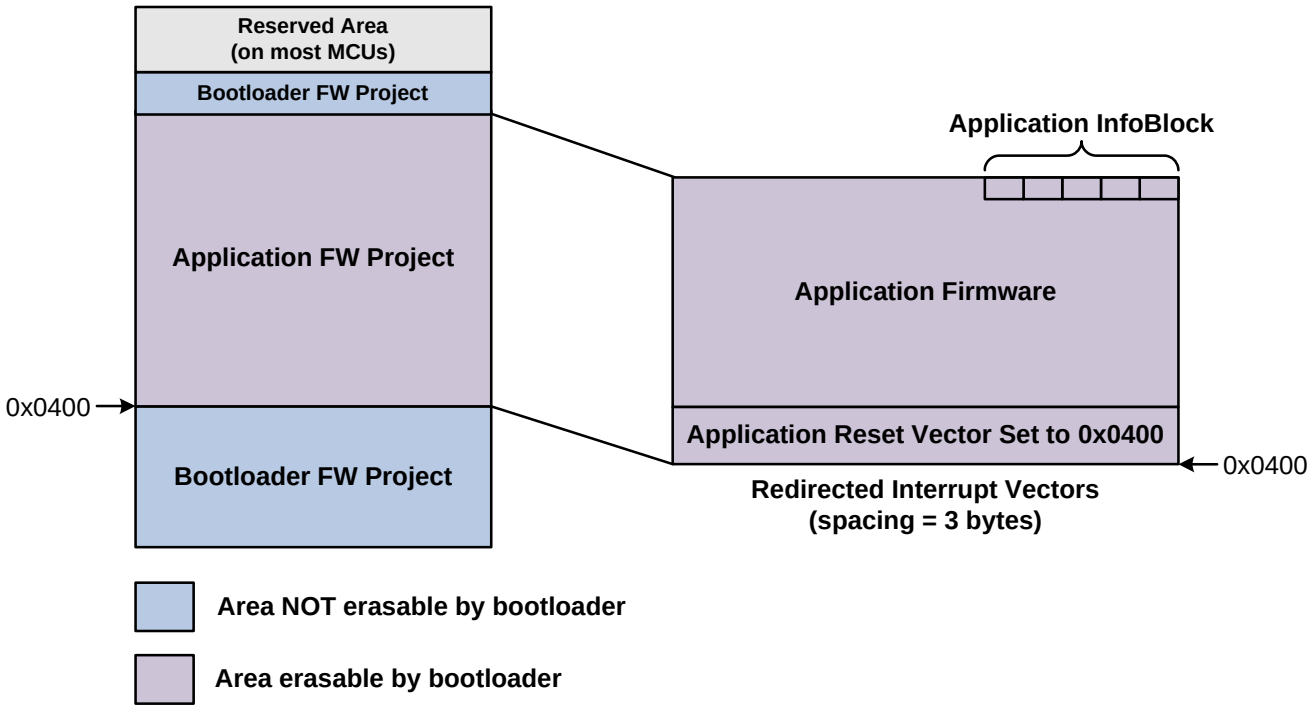


Figure 8. Target Application Memory Map

4.1. Target Application Template

A target application template is included for easy integration with existing application code or to use as a starting point. For example, the C8051F33x template includes the following files:

- F330_Blinky.c
- STARTUP.A51
- F33x_InfoBlock.c

4.2. Configurable Options

The application firmware should always keep its version number updated in the Application InfoBlock whenever a new version is built so that the application hex file includes this information. The active data source software can interpret this information from the hex.

Table 3. F33x_InfoBlock.c

Parameter	Options
TGT_APP_FW_VERSION_LOW and TGT_APP_FW_VERSION_HIGH[1]	8-bit values: **2 and 1**
TGT_APP_FW_INFOBLOCK_LENGTH[2]	8-bit value: **7**

Notes:
1. App v1.2: Low = 2 and High = 1.
2. See Table 5 in application note, “AN533: Modular Bootloader Framework for Silicon Labs C8051Fxxx Microcontrollers”.

4.3. Making an Application Bootloader Aware

A series of simple steps can be used to make an existing application firmware project “bootloader aware”, i.e., allow it to co-exist with the bootloader. These steps are described in detail in application note, “AN533: Modular Bootloader Framework for Silicon Labs C8051Fxxx Microcontrollers”. The following is a summary of the changes needed when using the Keil toolchain for the C8051F33x MCU family:

1. Add **STARTUP.A51** to the application firmware project and build list; this changes the reset vector from 0x0000 to 0x0400.
2. Add these options to the **compiler** command line of the project:

```
INTVECTOR(0x0400) INTERVAL(3)
```

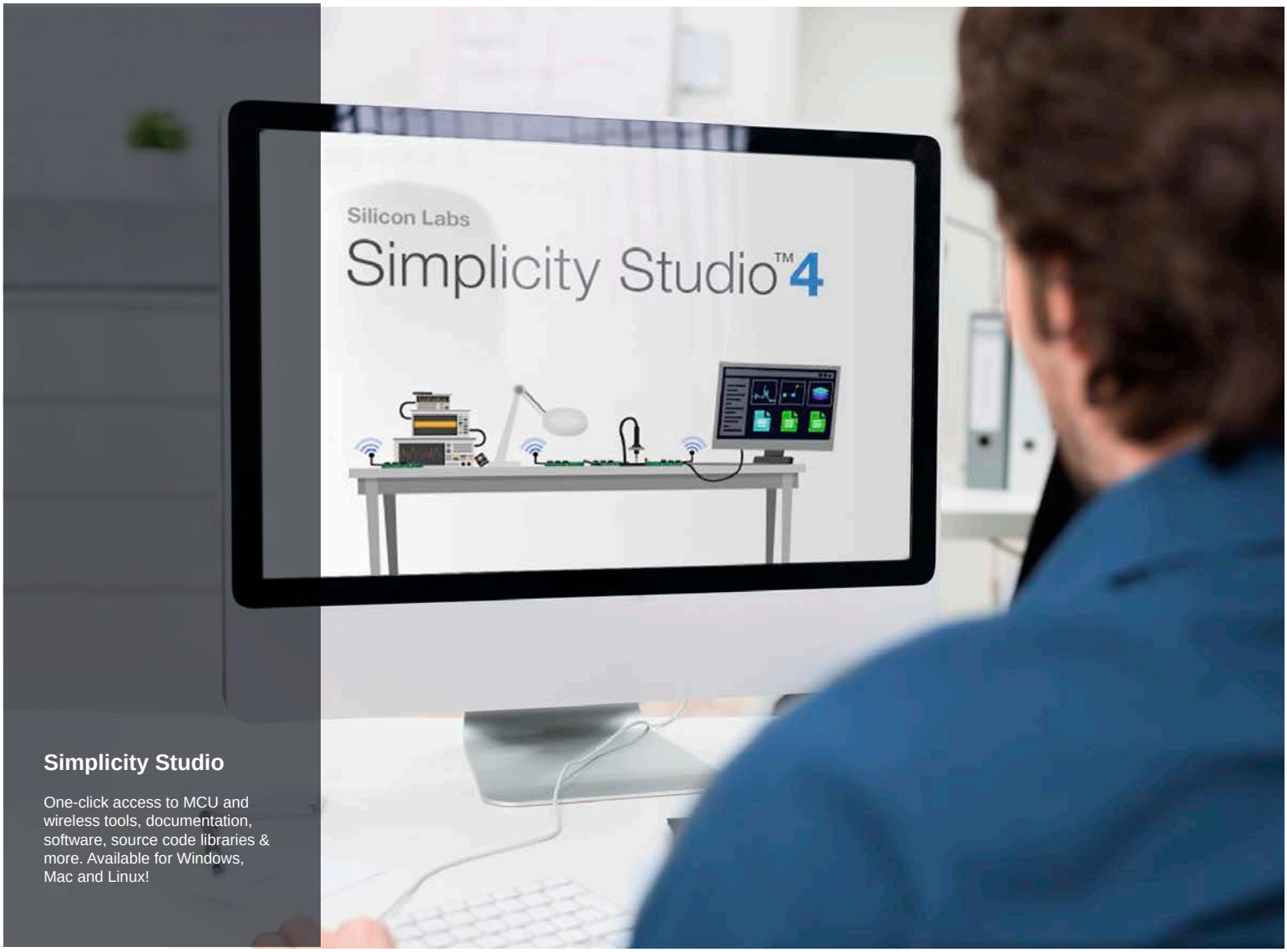
3. Add these options to the **linker** command line of the project:

```
CODE(0x0400-0x1BFF, ?CO?F33X_INFOBLOCK(0x1BF5))
```

4. Add **F33x_InfoBlock.c** to the application project and build list.
5. (Optional) Add code to the application to recognize the TGT_Enter_BL_Mode command and take appropriate action.
6. Check the hardware design to allow the use of a GPIO pin as a fail-safe trigger to enter bootload mode. In the UART bootloader example, port pin P0.7 is used for this purpose. To disable or change this, see **Fxxx_TargetBL_Main.c** in the Target Bootloader firmware project. If this is disabled, then the application has to provide some other way of entering bootload mode.

5. Data Source Examples

The Silicon Labs MCU Serial Bootloader Data Source software included with the modular bootloader framework is an example of an active data source software. This is described in application note, “AN533: Modular Bootloader Framework for Silicon Labs C8051Fxxx Microcontrollers”. The software source code is included with UART bootloader source code.



Simplicity Studio

One-click access to MCU and wireless tools, documentation, software, source code libraries & more. Available for Windows, Mac and Linux!

IoT Portfolio
www.silabs.com/IoT

SW/HW
www.silabs.com/simplicity

Quality
www.silabs.com/quality

Support and Community
community.silabs.com

SILICON LABS

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

http://www.silabs.com