AN136: Silicon Labs Production Programming Options

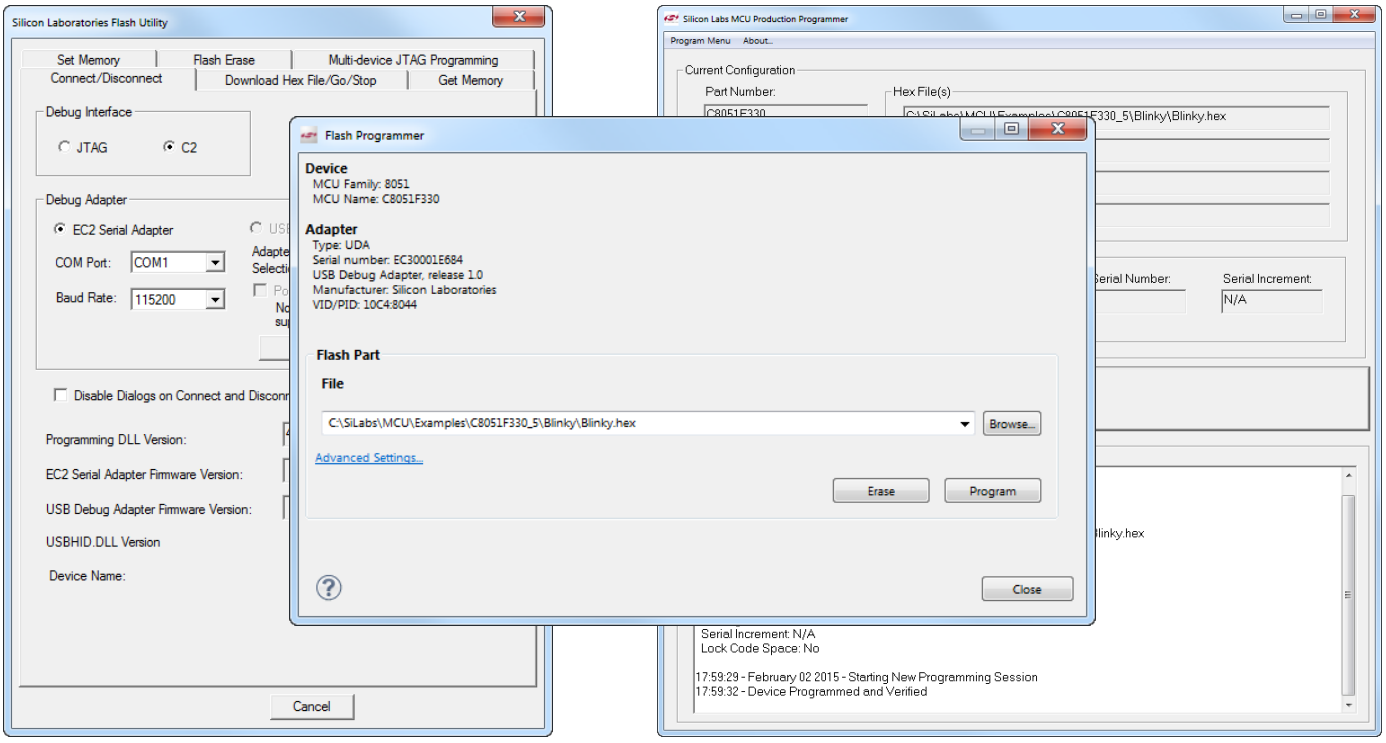
This application note gives an overview of production programming options available for Silicon Labs devices.

The two main methods for programming uninitialized devices are in-system programming and pre-programming. The most appropriate type of programming depends on the number of devices being programmed and whether access is available to the debug pins (JTAG, C2 interface, or Serial Wire Debug) of the device. Once devices have been programmed, a bootloader can be used to update application code using the UART or another supported interface.

Additional information on production programming can be found on the Silicon Labs website: https://www.silabs.com/products/mcu/Pages/ProgrammingOptions.aspx.

KEY POINTS

- When programming a device in-system, it is necessary that the programming master and the device being programmed share a common ground.
- Devices only need to be programmed once. Firmware updates can be received over any communications protocol supported by the system.
- The Silicon Labs USB Debug Adapter or the Starter Kit on-board debug adapter can be used for development and to program devices during production. Silicon Labs provides the tools necessary to streamline the debugging platform for production.



1. Relevant Programming Documentation

The following documents are available on http://www.silabs.com/support/pages/document-library.aspx

- *AN0062: Programming Internal Flash Over the Serial Wire Debug Interface* — This application note explains how to access the debug interface on Silicon Labs MCUs which utilize SWD.
- *AN127: Flash Programming via the C2 Interface* — This application note describes how to program the flash or EPROM on Silicon Labs microcontrollers that use the C2 interface.
- *AN105: Programming Flash Through the JTAG Interface* — This document describes how to program the FLASH memory on C8051 devices through the JTAG port.
- *UG162: Simplicity Commander Reference Guide* — This document describes how and when how to use the Command-Line Interface (CLI) of Simplicity Commander.
- *AN117: Using the C8051FXXX/TXXX On-Chip Interface Utilities Dynamic Link Library* — This application note provides in detail how to load a hex file onto the device and read and write to memory.
- *AN124: Pin Sharing Techniques for the C2 Interface* — This application note guides developers through maintaining access to the required C2 pins without interrupting user function or configuration.
- *AN169: USBXpress Programmer's Guide* — This application note guides developers through interfacing and programming Silicon Labs' USB MCUs and communication bridges.

Application Notes can be accessed on the Silicon Labs website (www.silabs.com/interface-appnotes) or in Simplicity Studio using the [**Application Notes**] tile.

2. In-System Programming

In-system programming involves programming devices after installation in the end system. In this scenario, access to the debug pins (JTAG, C2 interface, or Serial Wire Debug) is provided in the end system to enable connection to a programming master. This programming master can be a Silicon Labs USB Debug Adapter (UDA), Starter Kit (EFM8, EFM32, or wireless STK) on-board debug adapter, custom hardware, or for JTAG devices, a JTAG Boundary Scan test system that supports the programming of Silicon Labs devices.

2.1 Designing a System that Supports In-System Programming

Whether using the Silicon Labs USB Debug Adapter or Starter Kit on-board debug adapter, or building a custom programming master, an in-system programmable system needs to provide access to the debug pins (JTAG, C2 interface, or Serial Wire Debug) of the target device.

The pins required to program JTAG devices are TCK, TMS, TDI, TDO and GND. It is necessary that the programming Master and the device being programmed share a common ground.

For C2 devices, access to C2CK, C2D, and GND are required. See *AN124: Pin Sharing Techniques for the C2 Interface* in 1. Relevant Programming Documentation or on the Silicon Labs website (http://www.silabs.com/8bit-appnotes) for more information on pin sharing with the C2 interface. The development board schematics provided in Simplicity Studio or in the kit's User Guide can also be used as an example.

The EFM32 family uses Serial Wire Debug (SWD) which is a two-wire protocol for accessing the ARM debug interface. The pins required to program these devices are SWDIO and SWCLK.

1. SWDIO: a bidirectional data line
2. SWCLK: a clock driven by the host

For more information on programming over the SWD interface, please refer to *AN0062: Programming Internal Flash Over the Serial Wire Debug Interface* in 1. Relevant Programming Documentation or on the Silicon Labs website (http://www.silabs.com/32bit-appnotes).

The standalone 32-bit programmer using a Giant Gecko Starter Kit can be found in *AN1011: EFM32 Standalone Programmer* at http://www.silabs.com/32bit-appnotes.

2.2 Using the Debug Adapters

The Silicon Labs USB Debug Adapter or Starter Kit on-board debug adapter used for system development can also be used to program devices during production.

Table 2.1. Debug Adapter and Device Compatibilities

Adapter	Device Familiy
8-bit USB Debug Adapter	C8051 and EFM8 MCUs, Si106x/Si108x Wireless MCUs
32-bit USB Debug Adapter	SiM3C1xx, SiM3U1xx, SiM3L1xx MCUs
Silicon Labs Starter Kit or Wireless Starter Kit	EFM8 and EFM32 MCUs, EZR32 Wireless MCUs, and EFR32 Wireless SoCs **Note:** This method is recommended for Wireless Gecko (EFR32) devices.

A detailed 32-bit example can be found at this knowledge base article: http://community.silabs.com/t5/32-bit-MCU/Debugging-with-the-EFM32-Starter-Development-Kits/td-p/97969. The standalone 32-bit programmer using a Giant Gecko Starter Kit can be found in *AN1011: EFM32 Standalone Programmer* at http://www.silabs.com/32bit-appnotes.

A detailed 8-bit example for using the C2 interface to program the flash can be found in *AN127: Flash Programming via the C2 Interface* in 1. Relevant Programming Documentation or at http://www.silabs.com/8bit-appnotes.

2.2.1 Simplicity Studio Flash Programmer

Simplicity Studio includes a Flash Programmer that can be used with any supported EFM8, Gecko MCU (EFM32), Wireless Gecko (EFR32 and EZR32), or C8051 device and both the USB Debug Adapter and Starter Kit on-board debug adapter. The Flash Programmer allows the user to upload their hex or binary file, select the range and load it onto the device. Simplicity Studio is available from the Silicon Labs website: http://www.silabs.com/simplicity-studio.

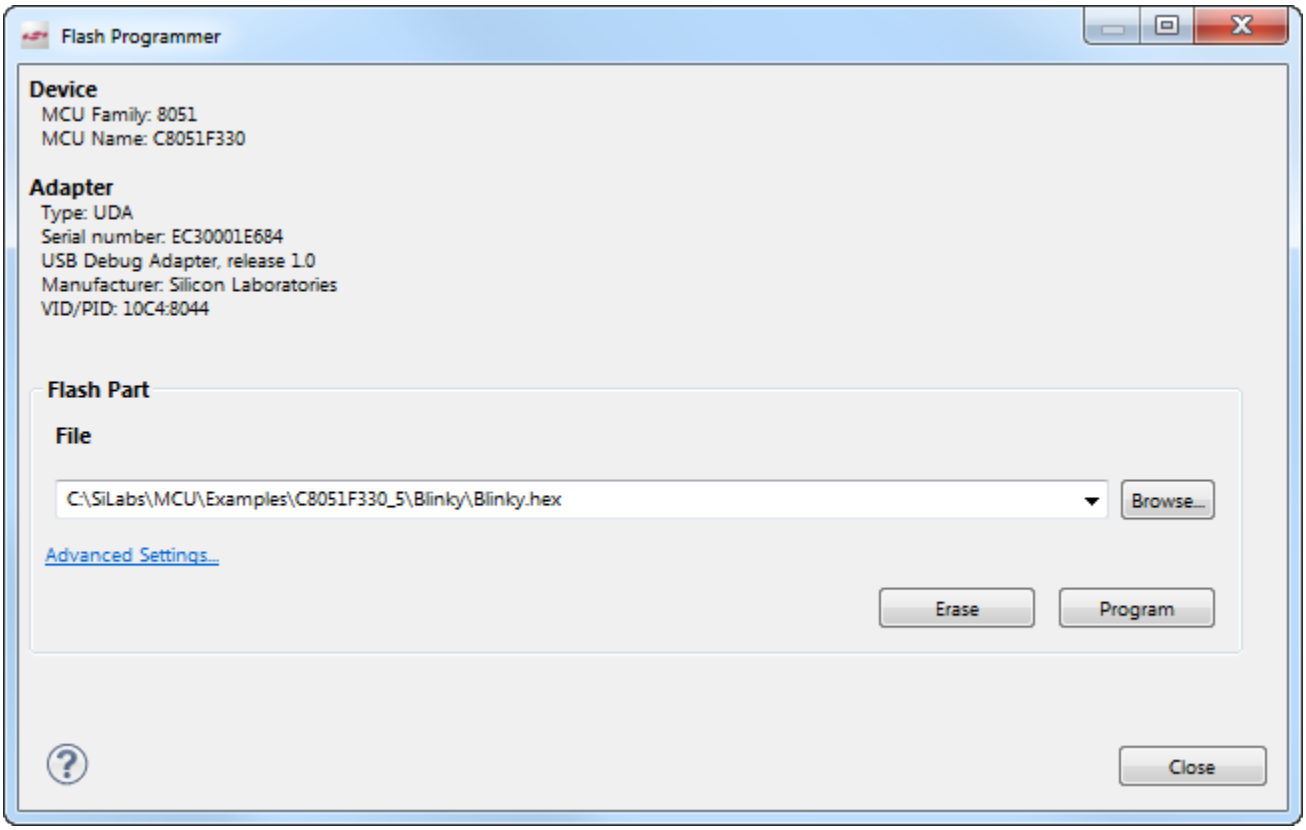


Figure 2.1. EFM8 and C8051 Flash Programmer

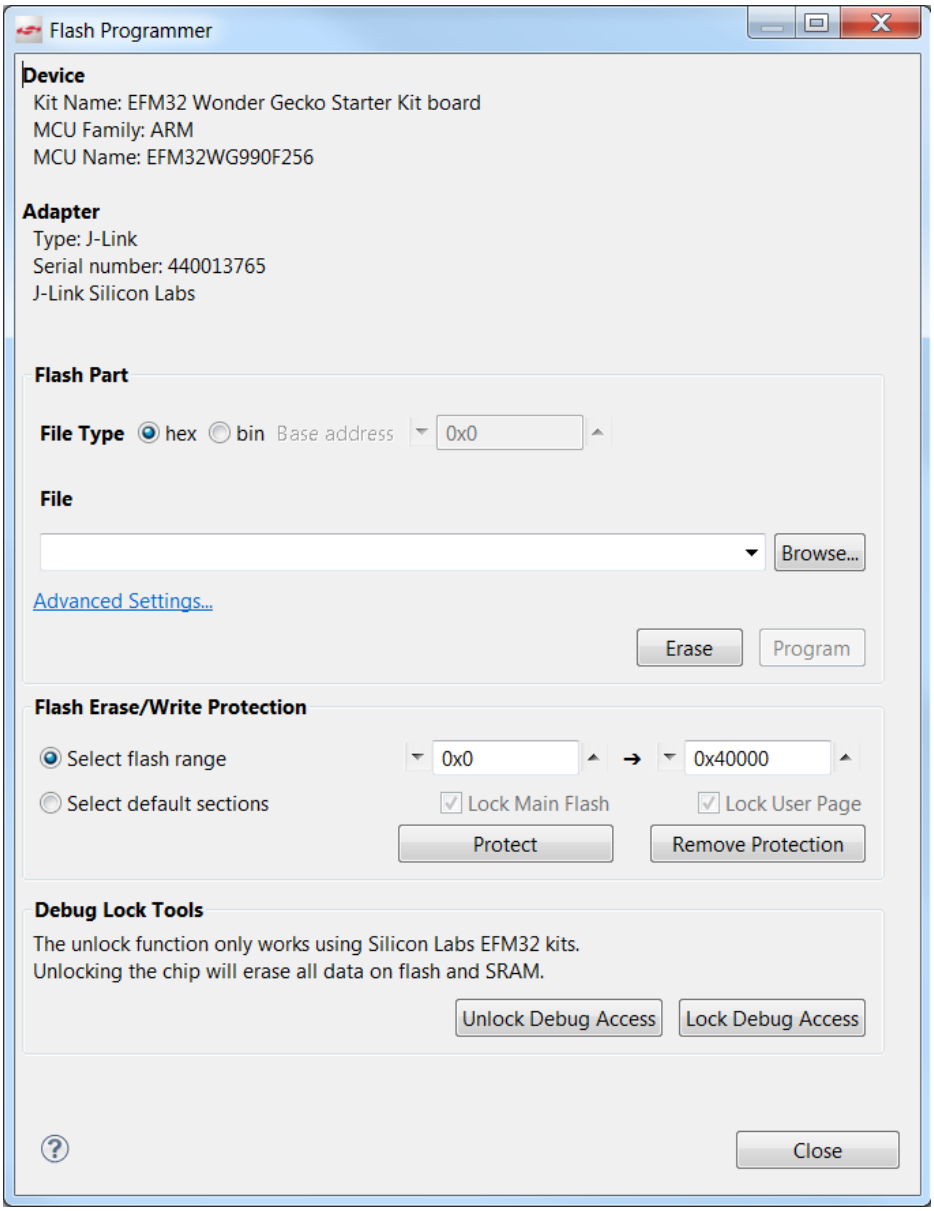


Figure 2.2. EFM32 Flash Programmer

2.2.2 Simplicity Commander

Simplicity Commander is a more advanced flash programmer that offers the user a kit programmer, flash programmer, and scriptable command line interface to communicate with the SWD interface. It supports EFM32, EZR32, and EFR32 devices. Simplicity Commander is a complete programming tool that allows the user to do the following:

- Flash their own application.
- Configure their own applications.
- Create binaries for production

Simplicity Commander is bundled with Simplicity Studio (www.silabs.com/simplicity). In Simplicity Studio v3, use the [**File**] menu to access Simplicity Commander or use Ctrl+3 to search for it. In Simplicity Studio v4, access this software executable on disk by going to `developer/adapter_packs/commander` in the Simplicity Studio installation location.

For more information on Simplicity Commander refer to *UG162: Simplicity Commander Reference Guide* in 1. Relevant Programming Documentation.

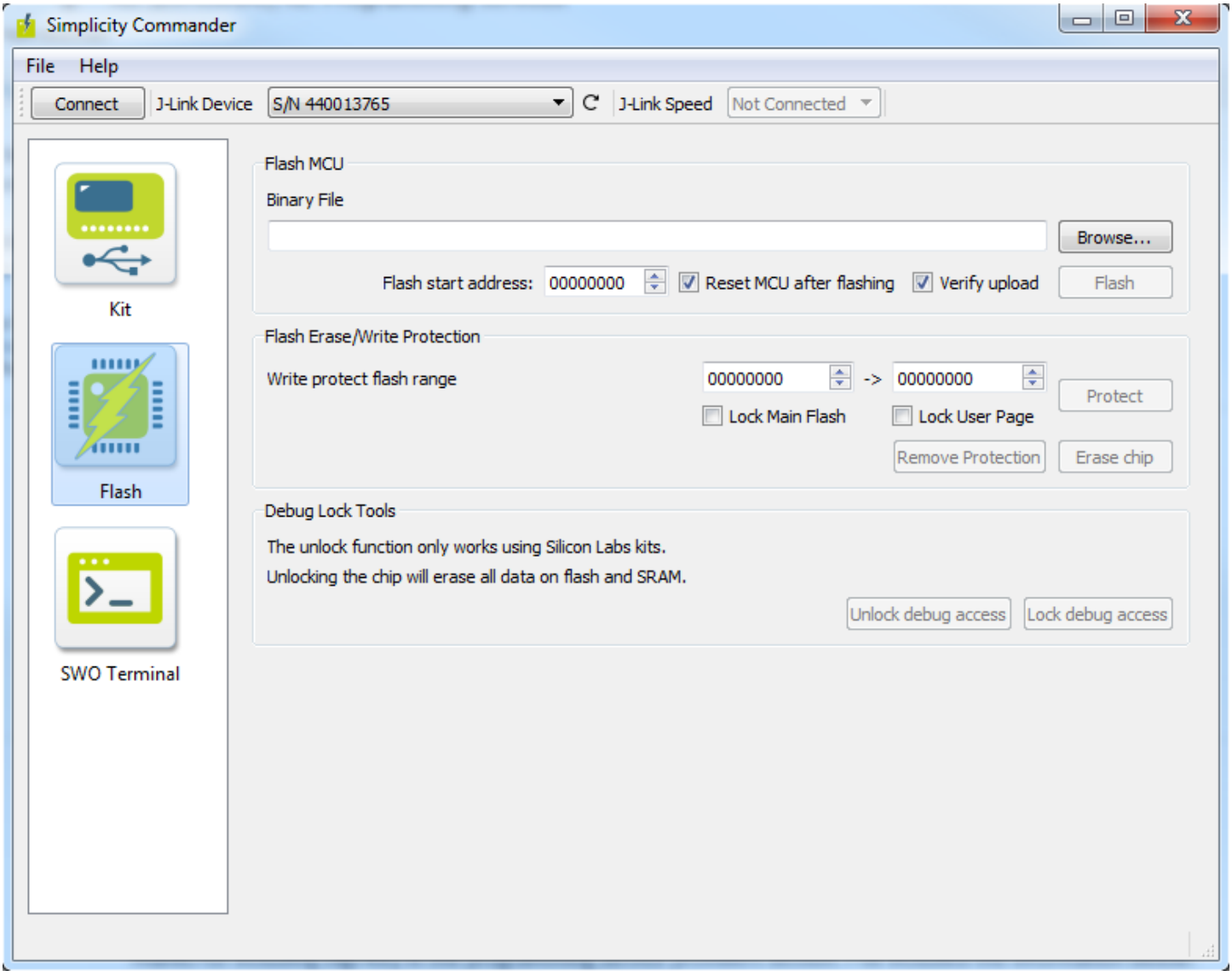


Figure 2.3. Simplicity Commander

2.2.3 Flash Programming Utilities

The Flash Programming Utilities for 8-bit devices is available on the Silicon Labs website (http://www.silabs.com/8bit-software) and is a standalone tool that supports the USB Debug Adapter, C8051 and EFM8 devices. This software has both a command-line version and a console version for Windows. In addition, the DLL is described in detail in AN117 in 1. Relevant Programming Documentation or on http://www.silabs.com/8bit-appnotes and can be used in custom software.

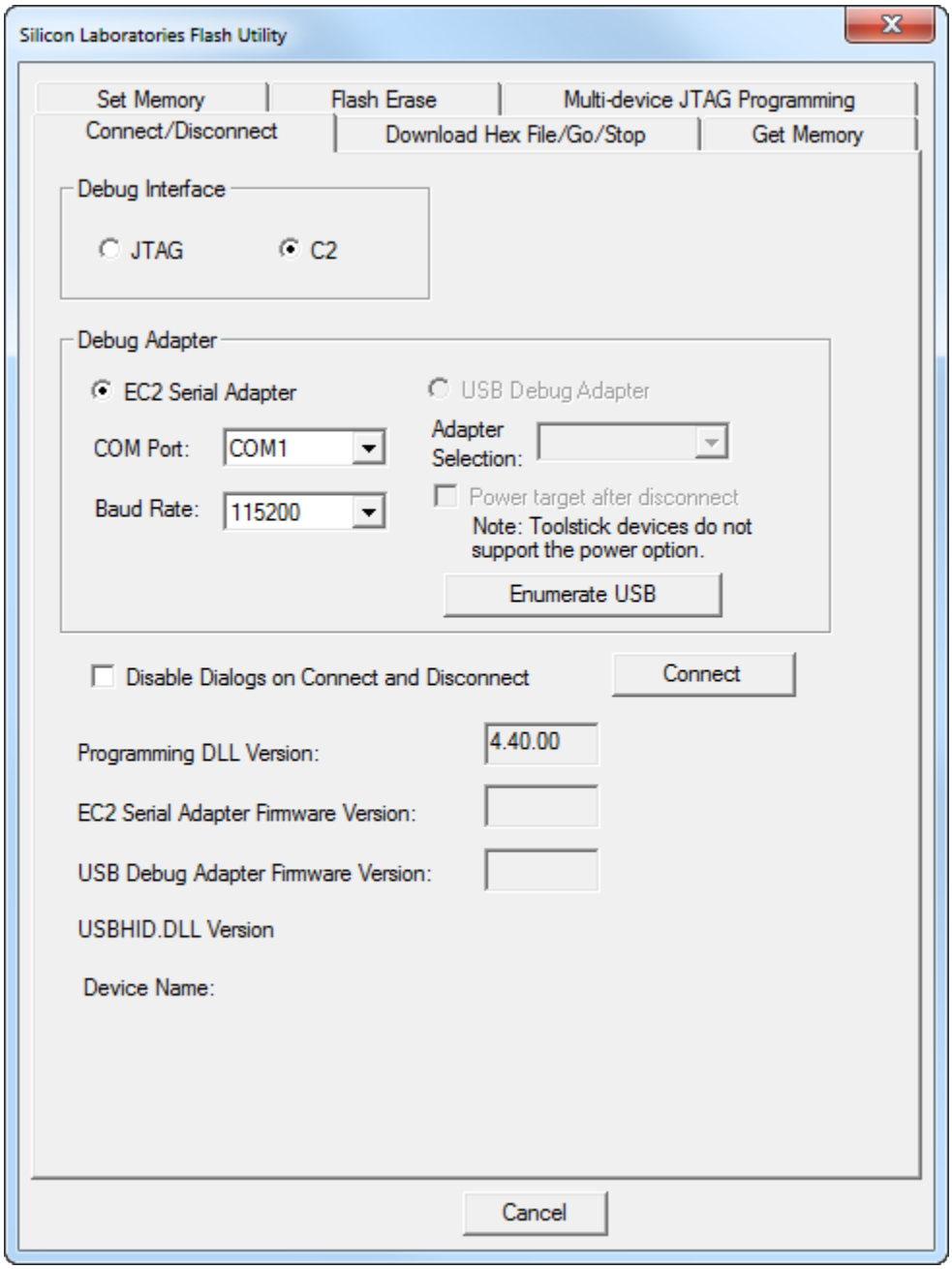


Figure 2.4. Flash Programming Utilities

2.2.4 MCU Production Programmer

The MCU Production Programmer is available on the Silicon Labs website (http://www.silabs.com/32bit-software and http://www.silabs.com/8bit-software). For EFM32 this production programmer if very similar to the flash programmer found in Simplicity Studio but provides fewer options aimed at a smoother production programming experience. The 8-bit production programmer supports the USB Debug Adapter, C8051, and EFM8 devices only, and uses the same DLL as the Flash Programming Utilities, but provides a simpler interface specifically for production programming.

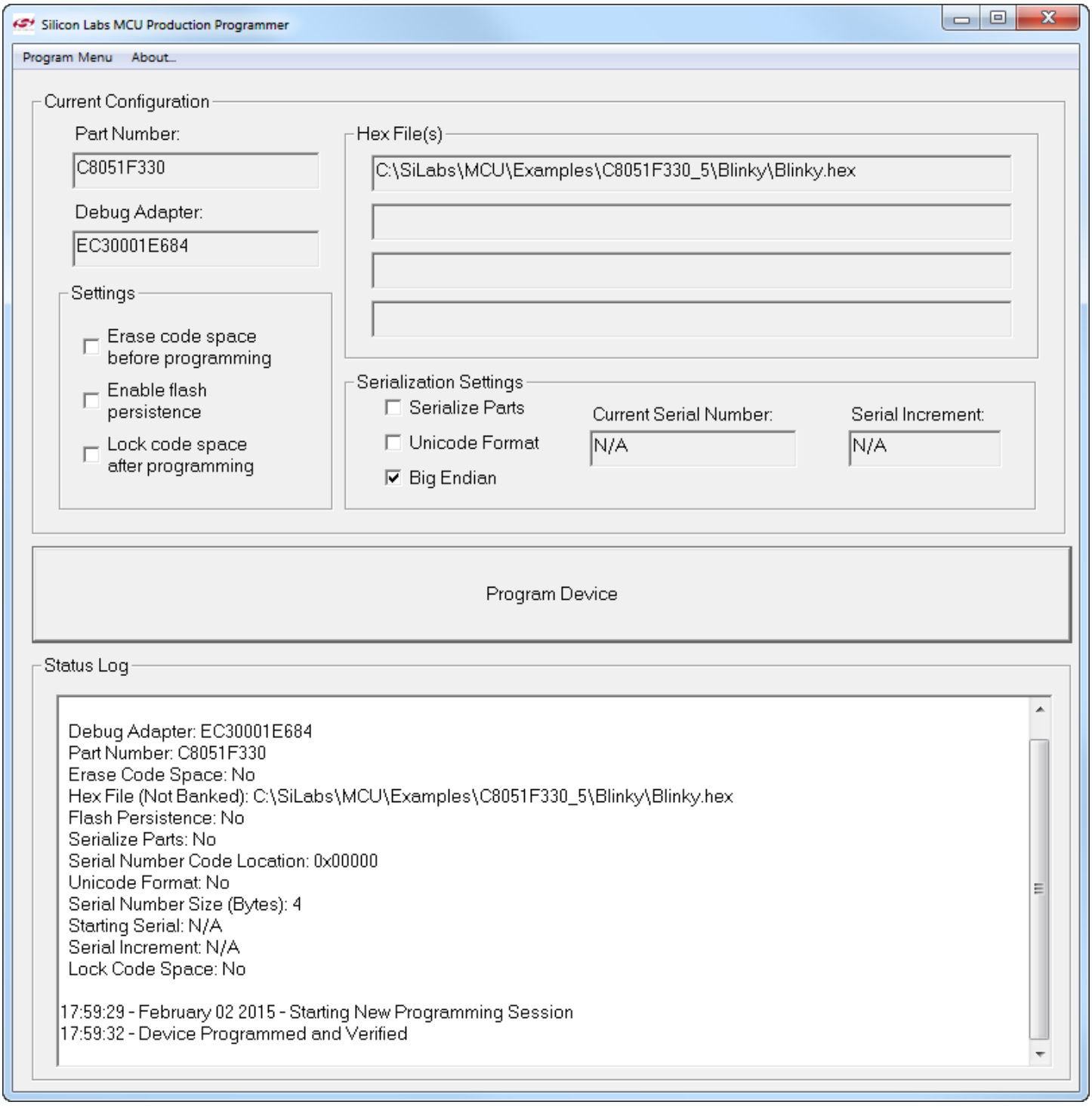


Figure 2.5. C8051 Production Programmer

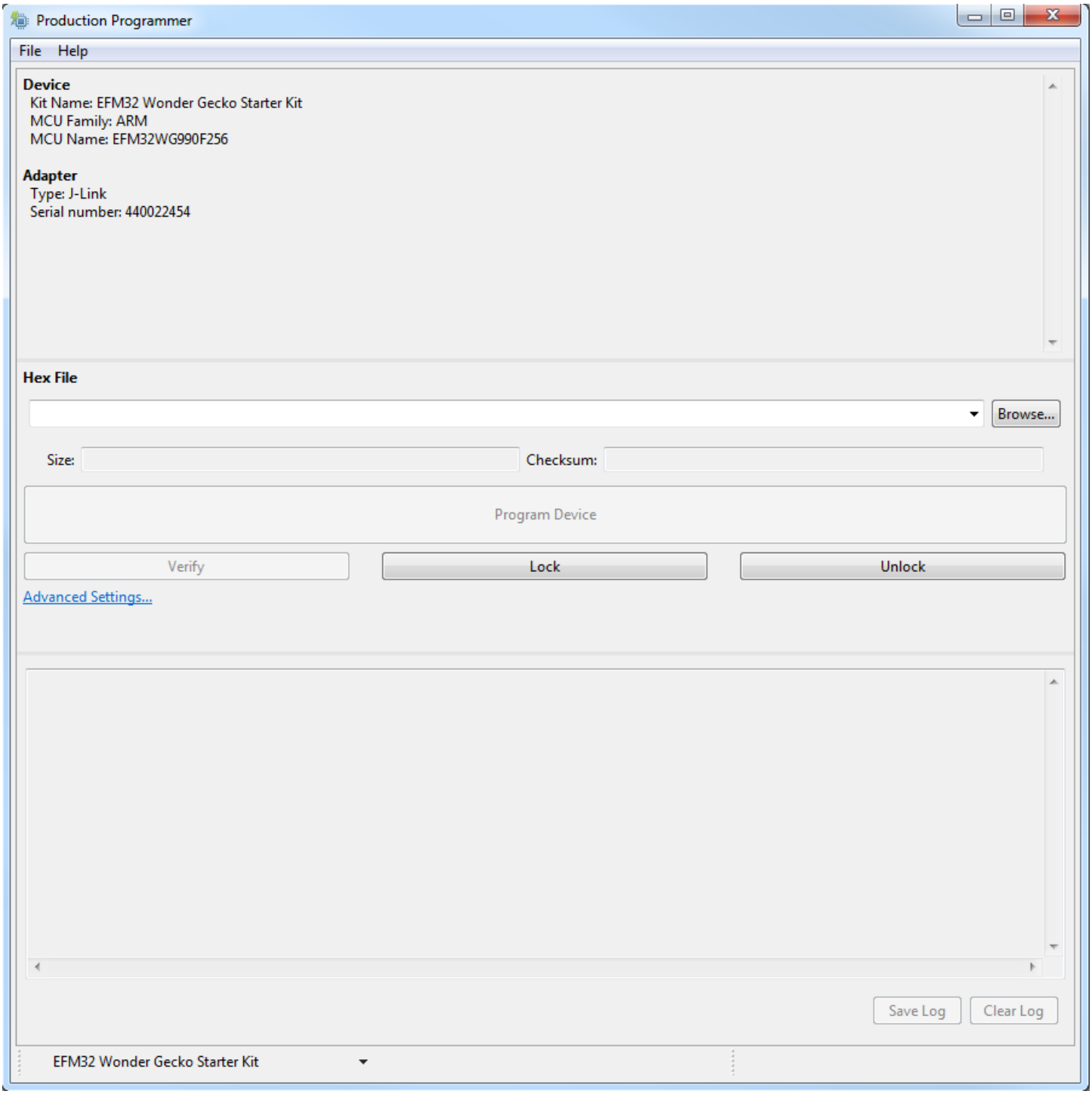


Figure 2.6. EFM32 Production Programmer

3. Pre-Programmed Devices

Pre-programmed devices are useful for end systems that do not provide access to the debug pins on the device. Devices are programmed before being installed in the end system. Pre-programming options include Silicon Labs' in-house programming service, creating custom hardware to program devices, using a third party programmer, or using a third party programming service.

3.1 In-House Programming

Silicon Labs offers pre-programmed devices. To do this the binary or hex file must be provided. This is one of the fastest programming options available but is subject to minimum order quantities (MOQ) and an additional cost. This option does not offer the flexibility or the dynamics of some of the other options and therefore it is not well suited for rapidly-changing programs. For this option, contact your local sales representative.

http://www.silabs.com/buysample/pages/contact-sales.aspx?view=map

3.2 Custom Hardware and Third Party Programmers

Another option for production programming is to build custom hardware to program the device prior to installation in the end system. This type of programmer would typically have one or more sockets to hold unprogrammed devices. Refer to the 1. Relevant Programming Documentation section of this document to find the application note title and details for implementing a custom programmer for each interface. Additionally application notes can be downloaded from the Silicon Labs website: http://www.silabs.com/8bit-appnotes and http://www.silabs.com/32bit-appnotes.

Support for Silicon Labs devices is also being integrated into third party production programmers. These programmers can range in price, speed, and number of devices programmed at once. A list of suppliers can be found in the knowledge base article: http://community.silabs.com/t5/32-bit-MCU-Knowledge-Base/3rd-party-programmers/ta-p/167422. Contact these suppliers for more information about their programming solutions.

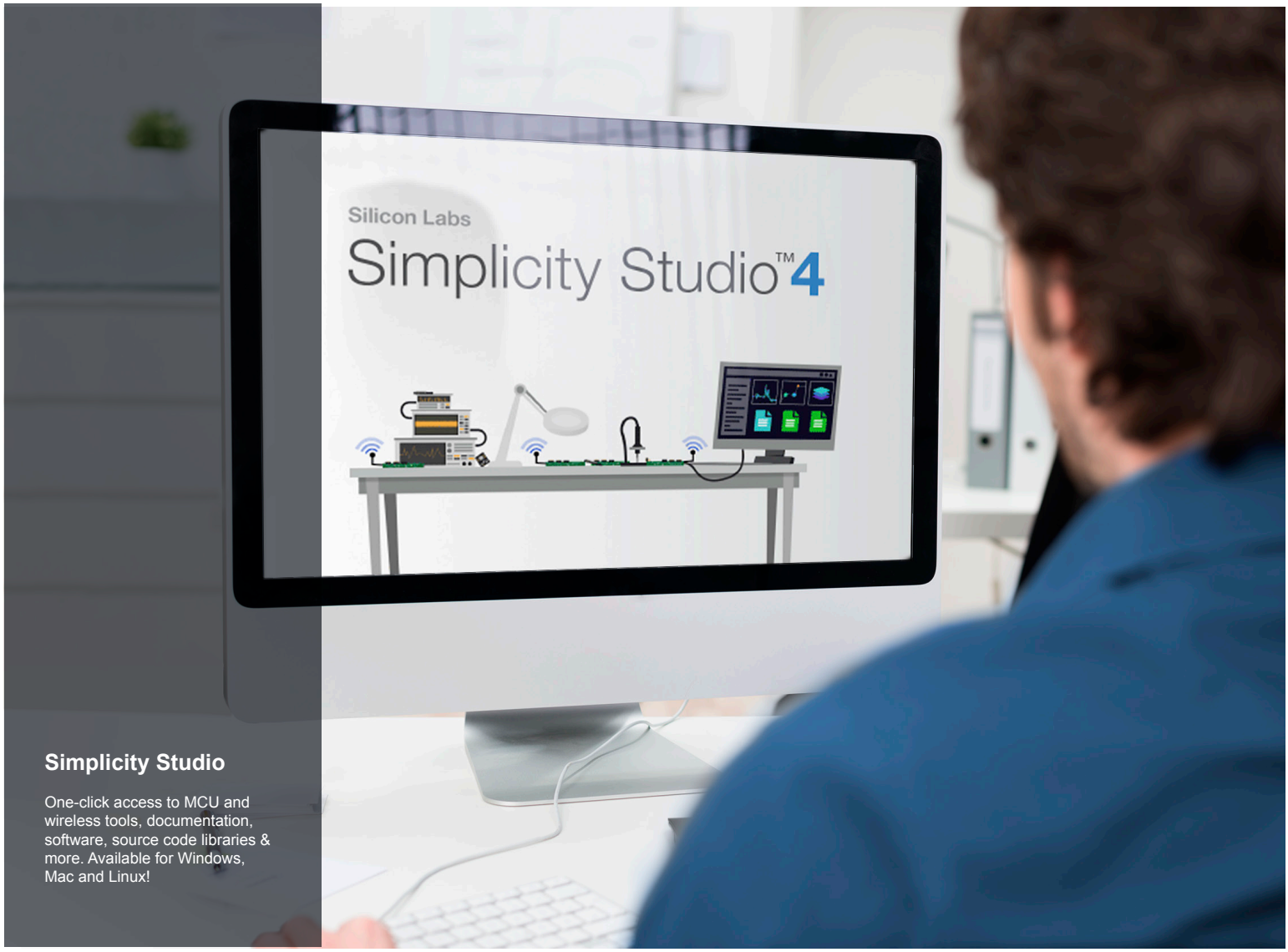
3.3 Third Party Programming Services

The last production programming option is to have a third party service program the devices. Third party services can be the entity that the device is ordered from or an entirely separate entity. Many Silicon Labs distributors offer production programming and, like in house programming, it may be subject to a MOQ and have an additional charge. For a list of distributor contacts please visit the knowledge base article: http://community.silabs.com/t5/32-bit-MCU-Knowledge-Base/3rd-Party-Pre-Programmed-Devices/ta-p/167404.

4. Updating Firmware

All Silicon Labs 32-bit and 8-bit MCUs have the ability to program flash from application code. This is done via bootloaders that are able to program or re-program the flash. Bootloaders allow firmware updates through application code and can receive the update through any of the communications peripherals such as the UART, SMBus/I2C, USB etc. There are many types of bootloaders for a variety of devices and many Silicon Labs MCUs have a pre-programmed bootloader from the factory. For more information on specific bootloaders, refer to the documents listed below which can be found at: http://www.silabs.com/8bit-appnotes and http://www.silabs.com/32bit-appnotes.

- *AN945: EFM8 Factory Bootloader User Guide* — This document describes the factory-programmed bootloaders available in EFM8 devices.
- *AN0003: UART Bootloader* — This application note provides details on the UART bootloader in EFM32 and EZR32 devices.
- *AN0042: USB/UART Bootloader* — This application note gives details on programming the EFM32 or EZR32 through an UART or an USB CDC class virtual UART without the need for a debugger.
- *AN533: Modular Bootloader Framework for Silicon Labs C8051Fxxx Microcontrollers* — This application note describes a modular bootloader framework that can be used to implement a bootloader system for any communication channel.
- *AN534: Can Bootloader* — This document describes how to enable firmware updates over the CAN bus in relevant C8051 devices.
- *AN535: Lin Bootloader* — This document describes how to enable firmware updates over the LIN bootloader in relevant C8051 devices.
- *AN767: SMBus Bootloader* — This document describes how to enable firmware updates over the SMBus in relevant C8051 and EFM8 devices.
- *AN200: USB Bootloader With Shared USBXpress Library* — This document describes how to enable firmware updates over the SMBus in relevant C8051 and EFM8 devices.



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