LIN Bootloader

1. Relevant Devices

This application note applies to the following devices:

C8051F50x, C8051F51x, C8051F54x, C8051F58x, and C8051F59x.

2. Introduction

A bootloader enables field updates of application firmware. A Local Interconnect Network (LIN) bootloader enables firmware updates over the LIN bus. The LIN 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
- Master Programmer Firmware
- Active Data Source Software
- Application Firmware Image (Hex File)

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.

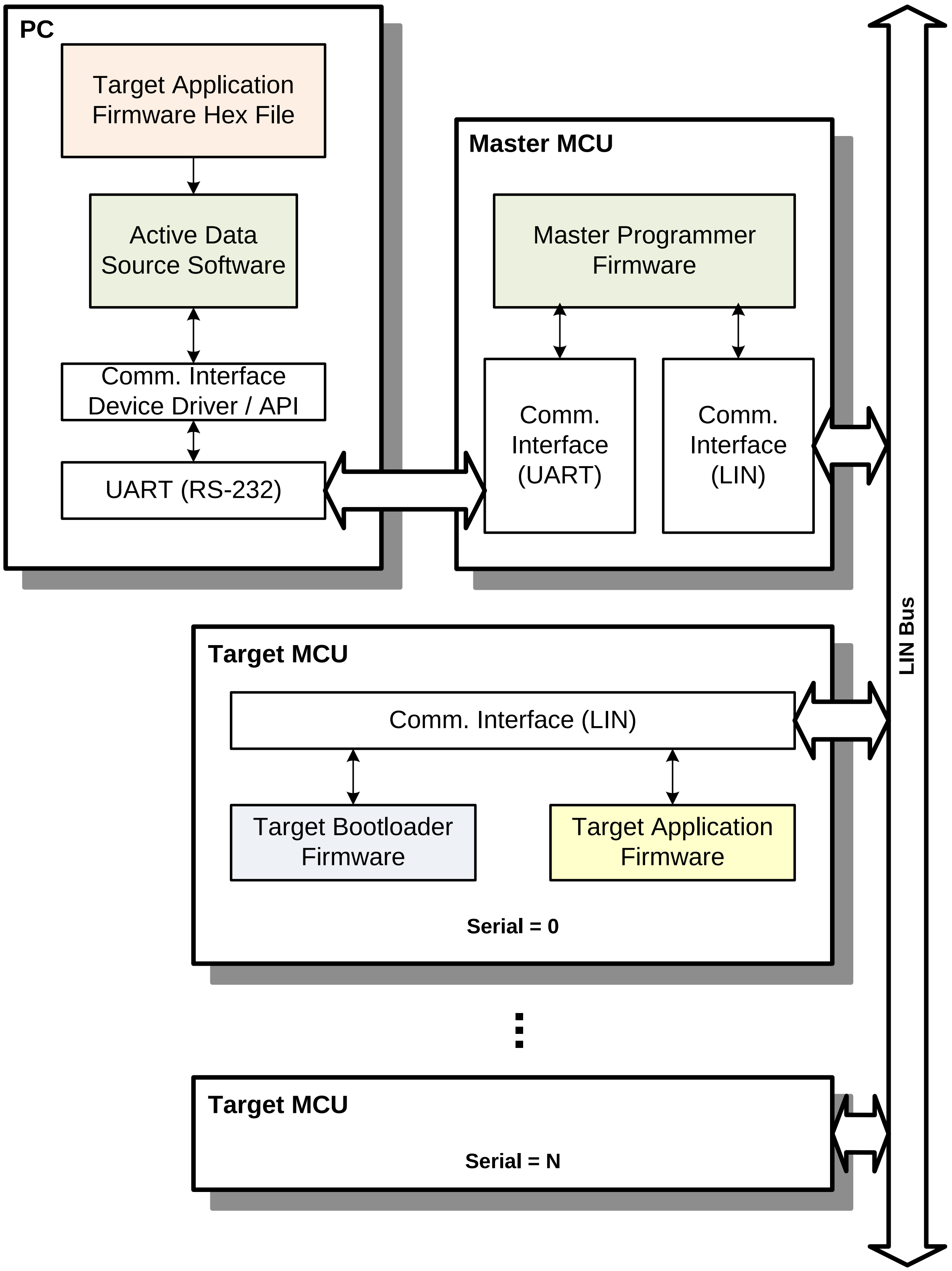

PC
Target Application Firmware Hex File
Active Data Source Software
Comm. Interface Device Driver / API
UART (RS-232)
Master MCU
Master Programmer Firmware
Comm. Interface (UART)
Comm. Interface (LIN)
LIN Bus
Target MCU
Comm. Interface (LIN)
Target Bootloader Firmware
Target Application Firmware
Serial = 0
Target MCU
Serial = N

Figure 1. Firmware Update Setup

3. Target Bootloader Profile

The LIN target bootloader firmware allows application firmware updates in the field over the LIN bus. The LIN bootloader code builds under the Keil toolchain with a total size of 1911 bytes. Because the bootloader and application have to be split on page boundaries, the bootloader takes up a total of 2 kB (= four flash pages) code space, with 512 bytes (= one flash page) of that total located on the last flash page, i.e., the one containing the lock byte. This means that the application firmware starts at address 0x0600 and ends one page short of the last flash page. Figure 2 shows the LIN bootloader memory map. Figure 3 shows the code space utilization of the bootloader grouped by functional blocks.

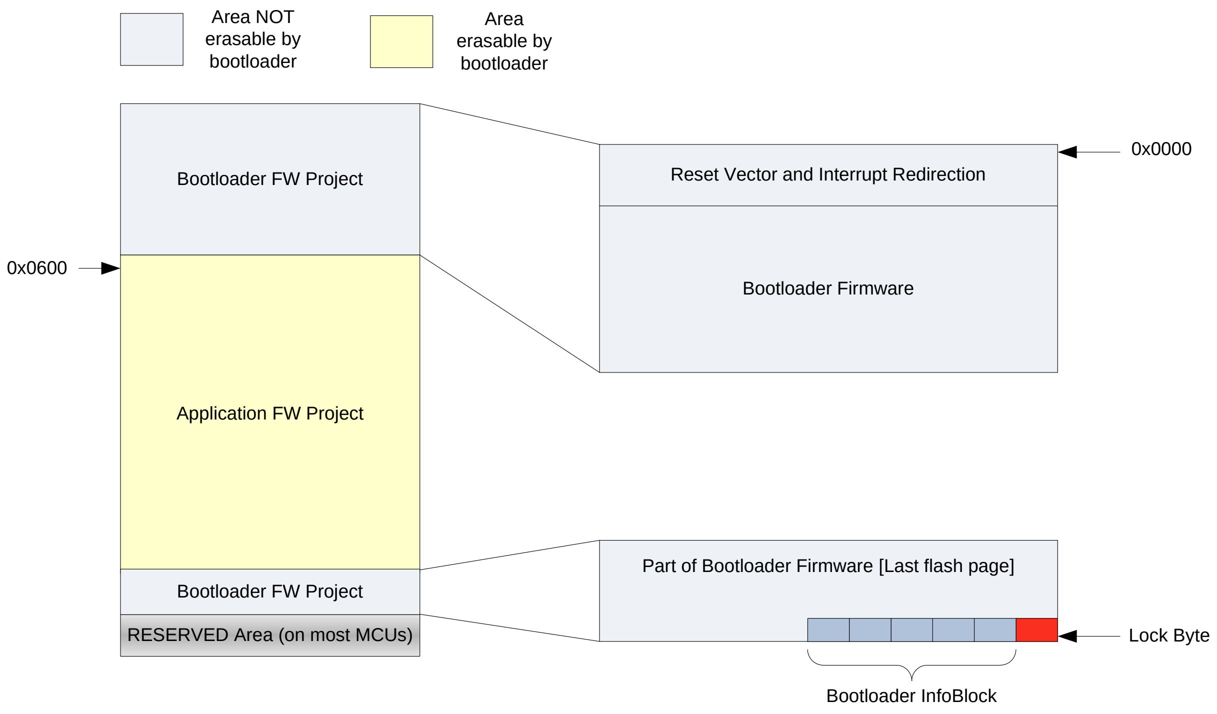

Area NOT erasable by bootloader
Area erasable by bootloader
Bootloader FW Project
0x0600
Application FW Project
Bootloader FW Project
RESERVED Area (on most MCUs)
Reset Vector and Interrupt Redirection
0x0000
Bootloader Firmware
Part of Bootloader Firmware [Last flash page]
Lock Byte
Bootloader InfoBlock

Figure 2. LIN Target Bootloader Memory Map

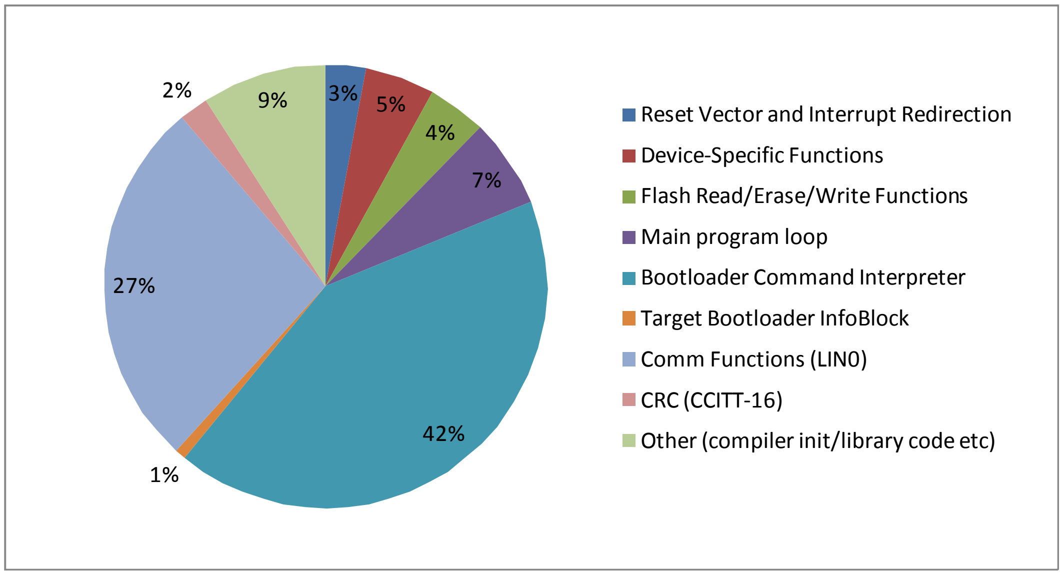


Figure 3. LIN Target Bootloader Code Space Utilization Profile

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_PRODUCT_CODE[1]	Any 8-bit value
TGT_BL_TYPE[2]	8-bit value: 0x82
TGT_FLASH_PAGE_SIZE[3]	Number of bytes per flash page: 512
TGT_FLASH_PAGE_SIZE_CODE[4]	8-bit value: 9
APP_FW_START_ADDR[5]	24-bit value: 0x000600
APP_FW_END_ADDR[6]	24-bit value: 0x00FBFF
DEV_SERIAL_CHECK_ENABLED[7]	Binary: 1
TGT_DEVICE_SERIAL0[8]	8-bit value: 0x01
TGT_DEVICE_SERIAL1[8]	8-bit value: 0x00

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 LIN bootloader protocol (see “Fxxx_BL130_LIN_Interface.h”).
3. Should be changed based on the MCU data sheet.
4. 2^n encoding: 2^9 = 512 bytes.
5. Starting address of App FW.
6. Ending address of App FW (includes App InfoBlock and Signature bytes).
7. 0 = Disabled; 1 = Enabled. When enabled, device serial number is matched for TGT_ENTER_BL_MODE command.
8. Ensure a UNIQUE serial number for each device on the same LIN network.

Table 2. Fxxx_TargetBL_Config.h

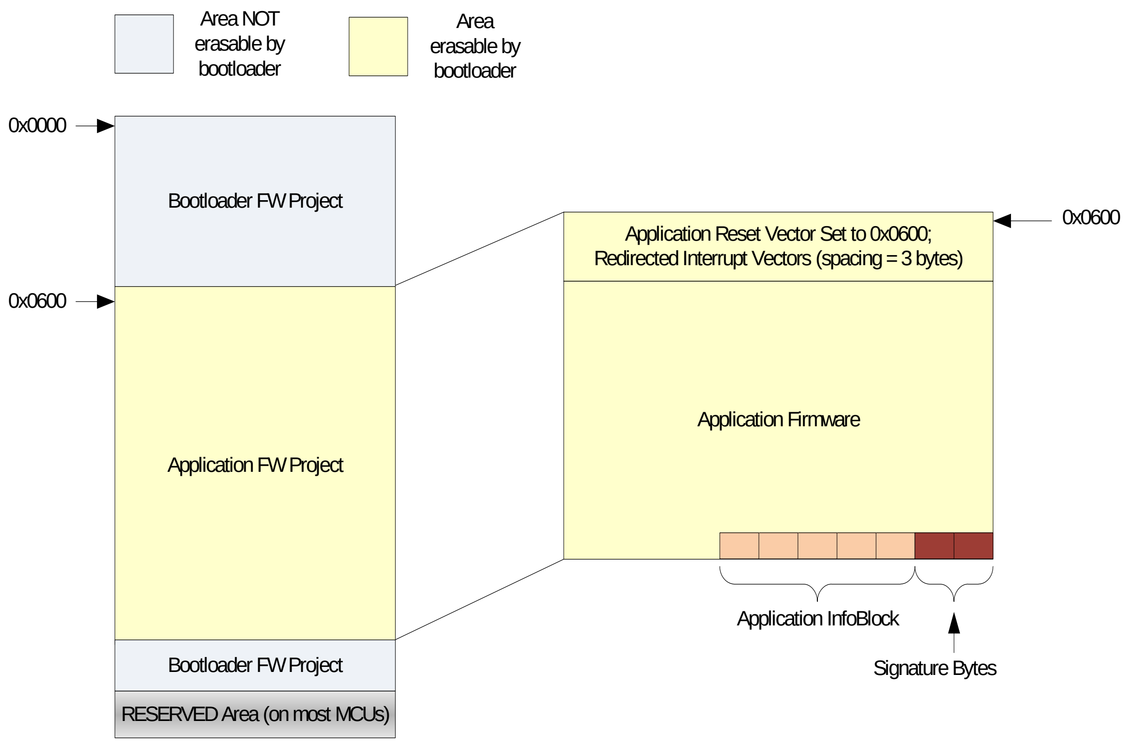
Parameter	Options
CODE_BANKING	Binary: 0[1]
	Binary: 1[2]
BOOTLOADER_PIN_OVERRIDE[3]	Binary: 1
TGT_BL_FW_INFOBLOCK_LENGTH[4]	8-bit value: 16
TGT_BL_FW_VERSION_LOW and TGT_BL_FW_VER-SION_HIGH[5]	8-bit values: 0 and 1
TGT_BL_BUF_SIZE[6]	Max number of bytes in bootloader receive buffer: 32
TGT_BL_BUF_SIZE_CODE[6]	8-bit value: 3
TGT_CRC_TYPE[7]	8-bit value: 0x40
TGT_BL_FW_INFOBLOCK_ADDR[8]	24-bit value: 0x00FCFE

Notes:

1. 0 = Disabled; 1 = Enabled. Disable for MCUs with flash memory 64 kB or less.
2. Enable for MCUs with flash memory greater than 64 kB.
3. 0 = Disabled; 1 = Enabled. When enabled, the bootloader will check a pin state on reset to see if it should stay in BL mode.
4. See Table 1 in AN533.
5. BL v1.0: Low = 0 and High = 1.
6. 2^n encoding: $2^3 = 8$ bytes.
7. This denotes that the BL uses the Silicon Labs-defined 16-bit CRC (CCITT-16) (see “Fxxx_CRC064_CCITT16.c”).
8. Start address of BL InfoBlock so that it ends adjacent to the lock byte.

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 4.


Area NOT erasable by bootloader
Area erasable by bootloader
0x0000
Bootloader FW Project
0x0600
Application FW Project
Bootloader FW Project
RESERVED Area (on most MCUs)
0x0600
Application Reset Vector Set to 0x0600; Redirected Interrupt Vectors (spacing = 3 bytes)
Application Firmware
Application InfoBlock
Signature Bytes

Note: The application firmware starts at address 0x0600 in this example.

Figure 4. 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. This template includes the following files:

- Header Files
  - Fxxx_BL130_LIN_Interface.h
  - Fxxx_Target_Config.h
  - Fxxx_Target_Interface.h
  - Fxxx_TargetApp_Config.h
- Source Files
  - F50x_TargetApp_LIN0_BLsupport.c
  - F50x_TargetApp_Startup.A51
  - Fxxx_TargetApp_InfoBlock.c

A typical application would use all files in the template, but that does not mean all the files in the template are required. Files can be omitted or used as a reference to create code that is more suitable to the application.

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, while the master programmer can retrieve this data using the TGT_Get_Info command.

Table 3. Fxxx_TargetApp_Config.h

Parameter	Options
TGT_APP_FW_VERSION_LOW and TGT_APP_FW_VERSION_HIGH[1]	8-bit values: 0 and 1
TGT_APP_FW_INFOBLOCK_LENGTH[2]	8-bit value: 9

Notes:
1. App v1.0: Low = 0 and High = 1.
2. See Table 5 in AN533.

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 AN533. The following is a summary of the changes needed when using the Keil toolchain for the 'F50x MCU family:

1. Add "F50x_TargetApp_Startup.A51" to application firmware project and build list; this changes the reset vector from 0x0000 to 0x0600.
2. Add these options to the compiler command line of the project: INTVECTOR(0x600) INTERVAL(3).
3. Add these options to the linker command line of the project: CODE(0x600-0xF9FD, ?CO?FXXX_TARGETAPP_INFOBLOCK(0xF9F4)).
4. Add 'Fxxx_TargetApp_InfoBlock.c' to the application project and build list.
5. [OPTIONAL] Add "F50x_TargetApp_LIN0_BLsupport.c" to the application project and build list, then insert calls in the application to call the functions in that source file to recognize the TGT_Enter_BL_Mode command and take appropriate action.
6. Check hardware design to allow the use of a GPIO pin as a fail-safe trigger to enter bootload mode. In the LIN bootloader example, port pin P1.4 is used for this purpose. To disable or change this, see "Fxxx_TargetBL_Config.h" and "F50x_TargetBL_DevSpecific.c" in the Target Bootloader firmware project. If this is disabled, then the application has to provide some other way of entering bootload mode.

4.4. Target Application Examples

Two versions of an application firmware example ("F50x_LIN0_Slave_Blinky") are included with the source code. These examples show the application firmware settings needed to co-exist with the bootloader and also demonstrate firmware updates by showing slightly different functionality between the two versions (v1 and v1.1). See AN533 for the firmware update steps.

5. Master Programmer and Data Source Examples

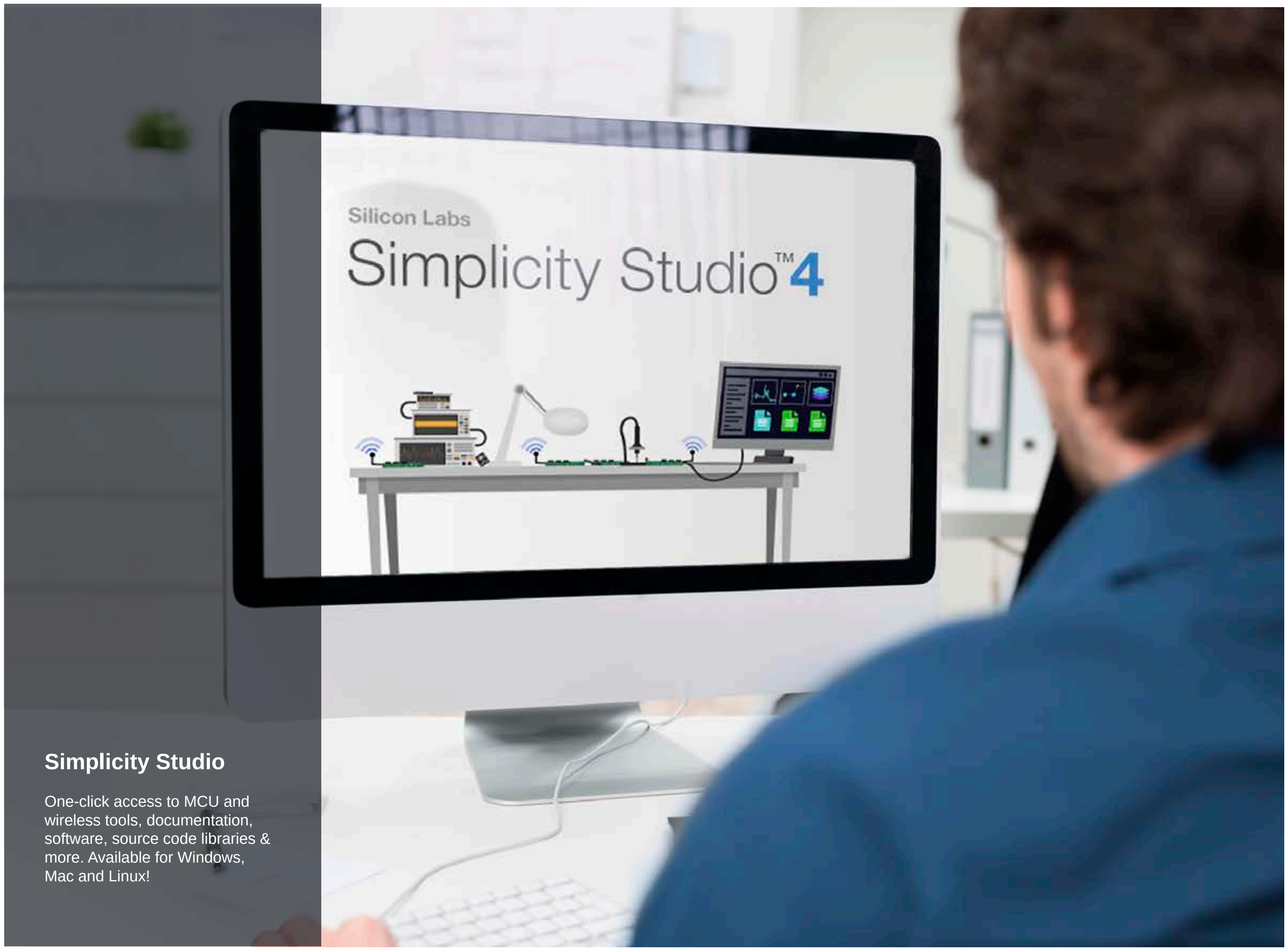
A master programmer example that runs on the C8051F500 MCU is included with the LIN bootloader source code. This master programmer example can update the firmware on any Silicon Labs MCU with a LIN interface that implements the LIN protocol as detailed in the above sections and per the specifications in Section 5. This example code can be used as-is, or can be used as a reference to implement this functionality on another MCU. The master programmer includes code to communicate with the active data source PC software via the UART. Code banking support is included, so the master programmer can work with targets that have more than 64 kB of flash memory. 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 installer and source code are included in the file “AN533SW.zip”, which is available for download from: http://www.silabs.com/products/mcu/Pages/ApplicationNotes.aspx

6. LIN Protocol Details

The LIN protocol details used by the LIN bootloader are shown in Table 4. Any changes to these details would have to be duplicated in all three projects (Master Programmer, Target Bootloader, and Target Application). If changes are made to any of the protocol details, the “TGT_BL_TYPE” value in “Fxxx_Target_Config.h” should also be changed from the Silicon Labs standard value of 0x82 (decimal 130) to a value within the user-defined range of 0x01 through 0x7F, and the “130” in the file name “Fxxx_BL130_LIN_Interface.h” should be changed appropriately as well.

Table 4. LIN Protocol Details

Parameter	Value	Parameter Definition and File Name
LIN Baud Rate	19200	Comm_Init() function in F50x_Comm_LIN0.c
Auto Baud	Disabled	Comm_Init() function in F50x_Comm_LIN0.c
LIN Message Size	8 bytes	LIN_MESSAGE_SIZE in Fxxx_BL130_LIN_Interface.h
Master Transmit Message ID (Master to Target)	0x04	ID_MASTER_TO_SLAVE_BL_CMD in Fxxx_BL130_LIN_Interface.h
Master Transmit Message ID (Master to Target)	0x05	ID_MASTER_TO_SLAVE_WR_CMD in Fxxx_BL130_LIN_Interface.h
Master Receive Message ID (Target to Master)	0x06	ID_SLAVE_TO_MASTER_BL_RSP in Fxxx_BL130_LIN_Interface.h



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