WRITING TO FLASH FROM FIRMWARE ON SiM3XXXX DEVICES

1. Introduction

This application note applies to the SiM3Cxxx, SiM3Uxxx, and SiM3Lxxx device families. The Flash memory on all Silicon Labs MCU devices is readable and writable from application code. This capability allows user software to store values, such as calibration constants or system parameters, to the Flash and to implement a boot-loading feature in which user firmware can be updated in-system from a remote site. The Flash that is not used by application code can be treated like an EEPROM, thus negating the need to connect an external EEPROM to the device.

This document starts with the basics of accessing Flash from application code on any device, including device-specific details. Then, it discusses advanced routines that can be developed from the basic routines. Finally, it describes precautions to take when writing or erasing Flash. Example code for the basic routines for all devices is installed with the Precision32 SDK, which can be downloaded from www.silabs.com/32bit-software.

2. Key Points

- The voltage supply monitor must be enabled in the VMONn module during Flash write and erase operations.
- The voltage supply monitor must be enabled as a reset source in the RSTSRCn module during Flash write and erase operations.

  Note: Any write or erase operations initiated while the voltage supply monitor is disabled or the voltage supply monitor is disabled as a reset source will be ignored.
- A Lock and Key sequence must be executed before executing a write or erase operation. The Flash interface can be unlocked for one or multiple write or erase operations.
- Writes to the WRDATA register while the Flash interface is locked or an incorrect unlock sequence will permanently lock the Flash interface until the next device reset.
- Firmware should disable interrupts when writing or erasing Flash to ensure the Flash interface accesses are sequential in time and take the minimum time possible.
- For all Flash write operations, firmware will stall unless operating from a memory space other than Flash.
- Interrupts posted during a Flash write or erase operation will be held pending until the completion of the Flash operation, after which they will be serviced in priority order.

3. Flash Essentials

Different device series have many similarities for Flash, including page sizes, lock bits, and the instructions used to read and write to Flash. The main differences are the amount of Flash available, how the voltage supply monitor is enabled, how the voltage supply monitor is enabled as a reset source, and how registers are modified to allow Flash writes and erases. Although the core stalls during Flash write and erase operations, peripherals (USARTn, SARADCn, TIMERn, etc.) remain active. Interrupts posted during a Flash write or erase operation are held until the Flash operation has completed, after which they are serviced in priority order.

3.1. Flash Organization

The Flash memory on most devices is organized into a set of 1024-byte pages. See the Memory Organization chapter of the device reference manual for specific information. Figures 1, 2, 3, and 4 show the Flash organization for the SiM3Uxxx devices.

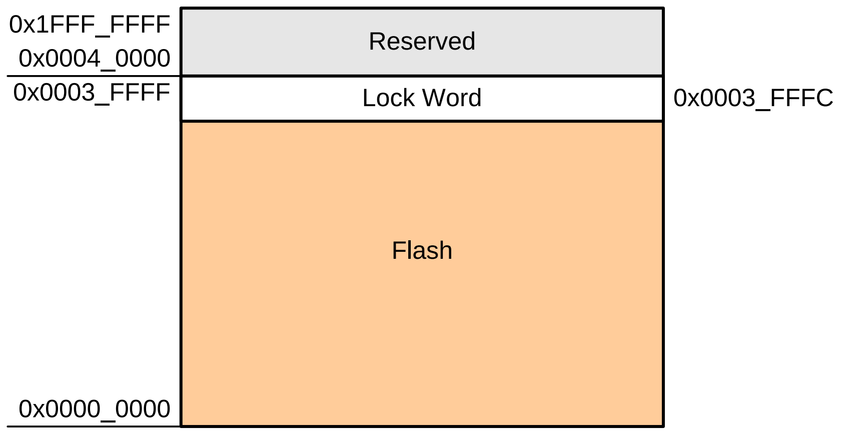


Figure 1. SiM3U16x Flash Memory Map (256 kB)

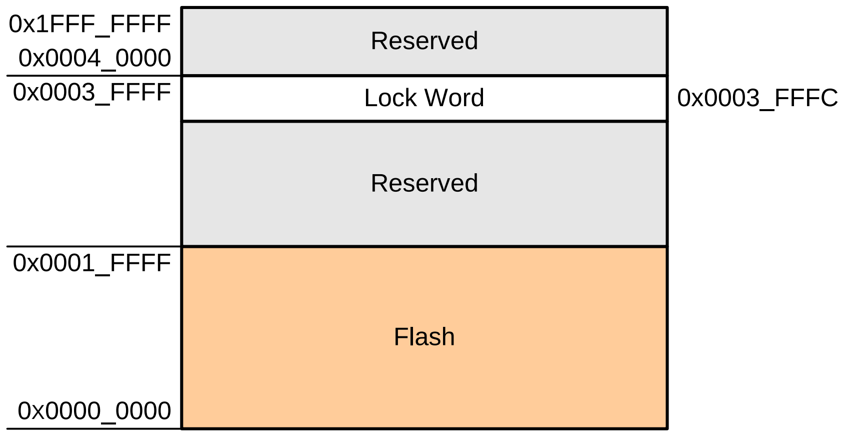


Figure 2. SiM3U15x Flash Memory Map (128 kB)

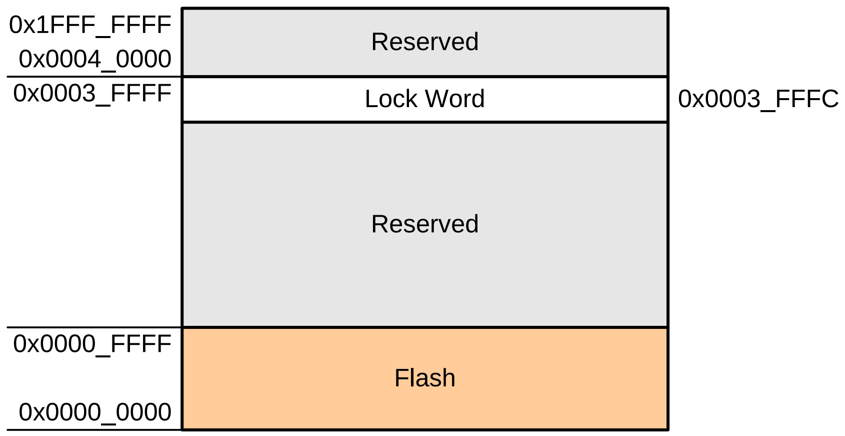


Figure 3. SiM3U14x Flash Memory Map (64 kB)

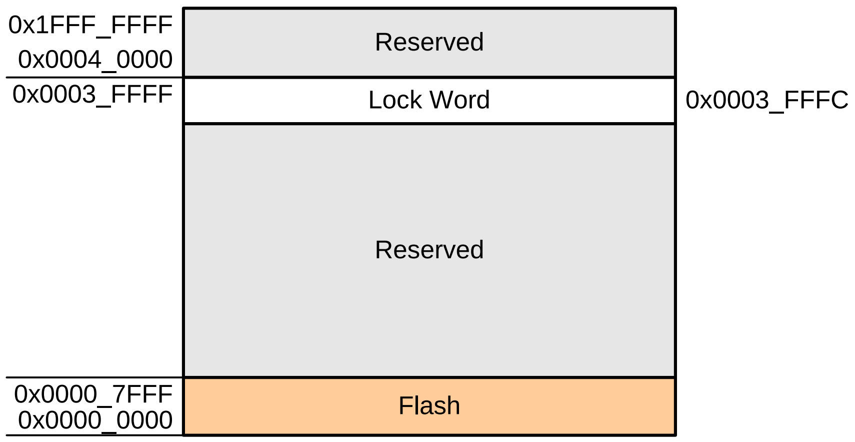


Figure 4. SiM3U13x Flash Memory Map (32 kB)

3.2. Locking Flash

The Flash can be locked by writing to the lock word in the Flash memory region. A value of 0xFFFF_FFFF or 0x0000_0000 at this location will unlock the Flash. Any other value written to this location will lock the entire Flash from external (debugger) writes or reads until:

- An erase operation is initiated from firmware.
- An erase operation is initiated through the debug port (SWD/JTAG).
- Firmware writes 0x0000_0000 to the lock word.

The location for the Flash lock word will differ between devices, so consult the device reference manual for more information.

3.3. Device-Specific Notes

Various MCUs have features that require consideration when accessing Flash.

3.3.1. SiM3Cxxx, SiM3Uxxx, and SiM3Lxxx Flash Lock and Key

All SiM3Cxxx, SiM3Uxxx, and SiM3Lxxx devices' writes and erases to Flash are protected with a lock and key function. The Flash Lock and Key Register (FLASHCTRLn) must be written with the correct key codes in sequence before Flash operations may be performed. The key codes are: 0xA5, 0xF1 for a single write or erase operation, and 0xA5, 0xF2 for multiple writes or erase operations. The timing does not matter, but the codes must be written in order. If the key codes are written out of order or the wrong codes are written, Flash writes and erases will be disabled until the next system reset. Flash writes and erases will also be disabled if a Flash write or erase is attempted before the key codes have been written properly. For a single write or erase, the Flash lock resets after each write or erase; the key codes must be written again before a following Flash operation can be performed. For multiple writes or erases, the Flash remains unlocked until the lock sequence is written (0xA5, 0x5A); the key codes do not need to be written before a following Flash operation can be performed.

3.3.2. SiM3Cxxx, SiM3Uxxx, and SiM3Lxxx Flash Read Timing

The SiM3Cxxx, SiM3Uxxx, and SiM3Lxxx devices require that the speed mode (SPMD) and the read-store enable (RDSEN) bits be set appropriately, depending on the AHB frequency. In addition, the Flash read timing mode (FLRTMD) bit can be configured to save power at slower clock frequencies. See the FLASHCTRLn chapter in the device's reference manual for detailed information on the read timing ranges.

3.3.3. SiM3Cxxx, SiM3Uxxx, and SiM3Lxxx VDD High Threshold

The SiM3Cxxx, SiM3Uxxx, and SiM3Lxxx devices have two settings for the VDD monitor threshold: Standard and High. The High setting increases the VDD Low (early warning) and VDD reset thresholds by approximately 300 mV. The High setting is recommended when operating at faster AHB clock speeds. See the VMONn chapter in the device's reference manual for detailed information on enabling the High threshold.

3.4. Flash Write and Erase Operations

The basic write operation writes a single half-word to Flash. The erase operation applies to a full page of Flash. Flash write and erase operations on Silicon Labs MCU devices are accomplished by using the WRDATA and WRADDR registers within the FLASHCTRLn module. The setting of the ERASEEN field within the FLASHCTRLn module determines whether a write or an erase will execute. Flash erase operations occur on page boundaries. The erase operation sets all the bits in the Flash page to logic 1. Flash write operations, which clear bits to logic 0, occur on single-byte boundaries. To successfully complete a write to Flash, the target bytes must be erased to 0xFFFF because the write instruction can only clear bits in a half-word.

3.5. Flash Read Operations

The basic read operation reads a single word from Flash. Flash read operations are accomplished by reading an address within the Flash region of the device memory map. See the Memory Organization chapter in the device's reference manual for detailed information on the Flash region boundaries.

4. Basic Flash Operations

The basic procedure for basic Flash operations is the same on all devices. Some devices require setting additional registers to enable Flash operations. The procedures in this section include the exceptions for the various device families. The code that implements these routines for each device family is in the FLASHCTRL examples, which are installed with the Precision32 SDK.

4.1. Reading a Single Half-Word

1. Disable interrupts.
2. Read the byte.
3. Restore interrupts, if originally enabled.

4.2. Writing a Single Half-Word

All bits, fields, and registers referred to in this sequence are in the FLASHCTRLn module.

1. Ensure the voltage supply monitor is enabled and enabled as a reset source (device reset sources and VMONn modules).
2. Disable erase operations (ERASEEN = 0).
3. Write the destination address to the WRADDR register.
4. Disable interrupts.
5. Write the initial unlock value to KEY (0xA5).
6. Write the single unlock value to KEY (0xF1).
7. Write the data half-word to WRDATA in right-justified format.
8. (Optional) If executing code from a memory space other than Flash, poll on the BUSYF flag until hardware clears it to 0.
9. Enable interrupts.

4.3. Writing Multiple Half-Words to Sequential Flash Addresses

To write a sequential set of bytes to Flash, code should execute from a memory space other than Flash. All bits, fields, and registers referred to in this sequence are in the FLASHCTRLn module.

1. Ensure the voltage supply monitor is enabled and enabled as a reset source (device reset sources and VMONn modules).
2. Disable erase operations (ERASEEN = 0).
3. Write the initial destination address to the WRADDR register.
4. Enable sequential writes (SQWEN = 1).
5. Disable interrupts.
6. Write the initial unlock value to KEY (0xA5).
7. Write the multiple unlock value to KEY (0xF2).
8. Write the data half-word to WRDATA in right-justified format.
9. (Optional) Poll on the BURSTS flag until the buffer has room for more data. If code is executing from RAM, this allows the core to perform other actions until a write operation completes and the buffer has room. The AHB bus will automatically stall until the operation completes if firmware writes data to WRDATA when the buffer is full.
10. Repeat steps 8 and 9 until all data is written. Hardware automatically increments the WRADDR field by 2 after each write operation.
11. (Optional) If executing code from a memory space other than Flash, poll on the BUSYF flag until hardware clears it to 0.
12. Write the multiple lock value to KEY (0x5A).
13. Enable interrupts.

4.4. Writing Multiple Half-Words to Non-Sequential Flash Addresses

All bits, fields, and registers referred to in this sequence are in the FLASHCTRLn module. To write multiple bytes to non-sequential addresses in Flash:

1. Ensure the voltage supply monitor is enabled and enabled as a reset source (device reset sources and VMONn modules).
2. Disable erase operations (ERASEEN = 0).
3. Disable interrupts.
4. Write the initial unlock value to KEY (0xA5).
5. Write the multiple unlock value to KEY (0xF2).
6. Write the destination address to WRADDR.
7. Write the data half-word to WRDATA in right-justified format.
8. (Optional) If executing code from a memory space other than Flash, poll on the BUSYF flag until hardware clears it to 0.
9. Repeat steps 6, 7, and 8 until all data is written.
10. Write the multiple lock value to KEY (0x5A).
11. Enable interrupts.

4.5. Erasing a Flash Page

All bits, fields, and registers referred to in this sequence are in the FLASHCTRLn module. To erase a page of Flash:

1. Ensure the voltage supply monitor is enabled and enabled as a reset source (device reset sources and VMONn modules).
2. Write the address of a byte in the Flash page to WRADDR.
3. Enable erase operations (ERASEEN = 1).
4. Disable interrupts.
5. Write the initial unlock value to KEY (0xA5).
6. Write the single unlock value to KEY (0xF1).
7. Write any value to WRDATA in right-justified format to initiate the page erase.
8. (Optional) If executing code from a memory space other than Flash, poll on the BUSYF flag until hardware clears it to 0.
9. Enable interrupts.

4.6. Erasing Multiple Flash Pages

All bits, fields, and registers referred to in this sequence are in the FLASHCTRLn module. To erase multiple pages of Flash:

1. Ensure the voltage supply monitor is enabled and enabled as a reset source (device reset sources and VMONn modules).
2. Enable erase operations (ERASEEN = 1).
3. Disable interrupts.
4. Write the initial unlock value to KEY (0xA5).
5. Write the multiple unlock value to KEY (0xF2).
6. Write the address of a byte in the Flash page to WRADDR.
7. Write any value to WRDATA in right-justified format to initiate the page erase.
8. (Optional) If executing code from a memory space other than Flash, poll on the BUSYF flag until hardware clears it to 0.
9. Repeat steps 6, 7, and 8 for each page.

10. Write the multiple lock value to KEY (0x5A).
11. Enable interrupts.

4.7. Example Code Implementation Notes

The FLASHCTRL example code that is installed with the Precision32 SDK contains functions that can be copied and pasted into applications. These functions can be modified as needed but should be used as a starting point for Flash write and erase operations.

5. Flash Write and Erase Guidelines

The following guidelines are strongly recommended for any system containing routines that write or erase Flash from code.

5.1. Voltage Supply Maintenance and the Voltage Supply Monitor

1. If the system power supply is subject to voltage or current “spikes”, add sufficient transient protection devices to the power supply to ensure that the supply voltages listed in the Absolute Maximum Ratings table in the device data sheet are not exceeded.
2. If the device has a minimum voltage supply rise time specification, ensure that it is met. If the system cannot meet the rise time specification, add an external voltage supply brownout circuit to the RESETb pin of the device that holds the device in reset until the voltage supply reaches the VDD High Supply Monitor Threshold and re-asserts RESETb if the voltage supply drops below the VDD High Supply Monitor Threshold.
3. Enable the on-chip voltage supply monitor and enable the voltage supply monitor as a reset source as early in code as possible. This should be the first set of instructions executed after the Reset Vector. For C-based systems, this will involve modifying the startup code in the Precision32 HAL. Ensure that there are no delays in software between enabling the voltage supply monitor and enabling the voltage supply monitor as a reset source.
4. As an added precaution, explicitly enable the voltage supply monitor and enable the voltage supply monitor as a reset source inside the functions that write and erase Flash memory. The voltage supply monitor enable instructions should be placed at the beginning of the Flash write and erase functions.
5. Ensure that all writes to the RESETEN register explicitly set the VMONREN bit to a 1. Areas to check are initialization code, which enables other reset sources, such as the Missing Clock Detector or Comparator, and instructions that force a Software Reset. A global search on “RESETEN” can quickly verify this.
6. If the device has a high and low threshold setting for the voltage supply monitor, enable the high setting.

The following sections show how to enable the voltage supply monitor, enable the voltage supply monitor as a reset source, and enable the high threshold setting where applicable using both HAL and non-HAL methods on various device families.

5.1.1. SiM3Cxxx/SiM3U1xx/SiM3L1xx Devices

Enable voltage supply monitor:

```
SI32_VMON_0->CONTROL.VMONEN = 1; // non-HAL

SI32_VMON_A_enable_vdd_supply_monitor(SI32_VMON_0); // HAL
```

Enable voltage supply monitor as a reset source:

```
SI32_RSTSRC_0->RESETEN.VMONREN = 1; // non-HAL

SI32_RSTSRC_A_enable_vdd_monitor_reset_source(SI32_RSTSRC_0); // HAL
```

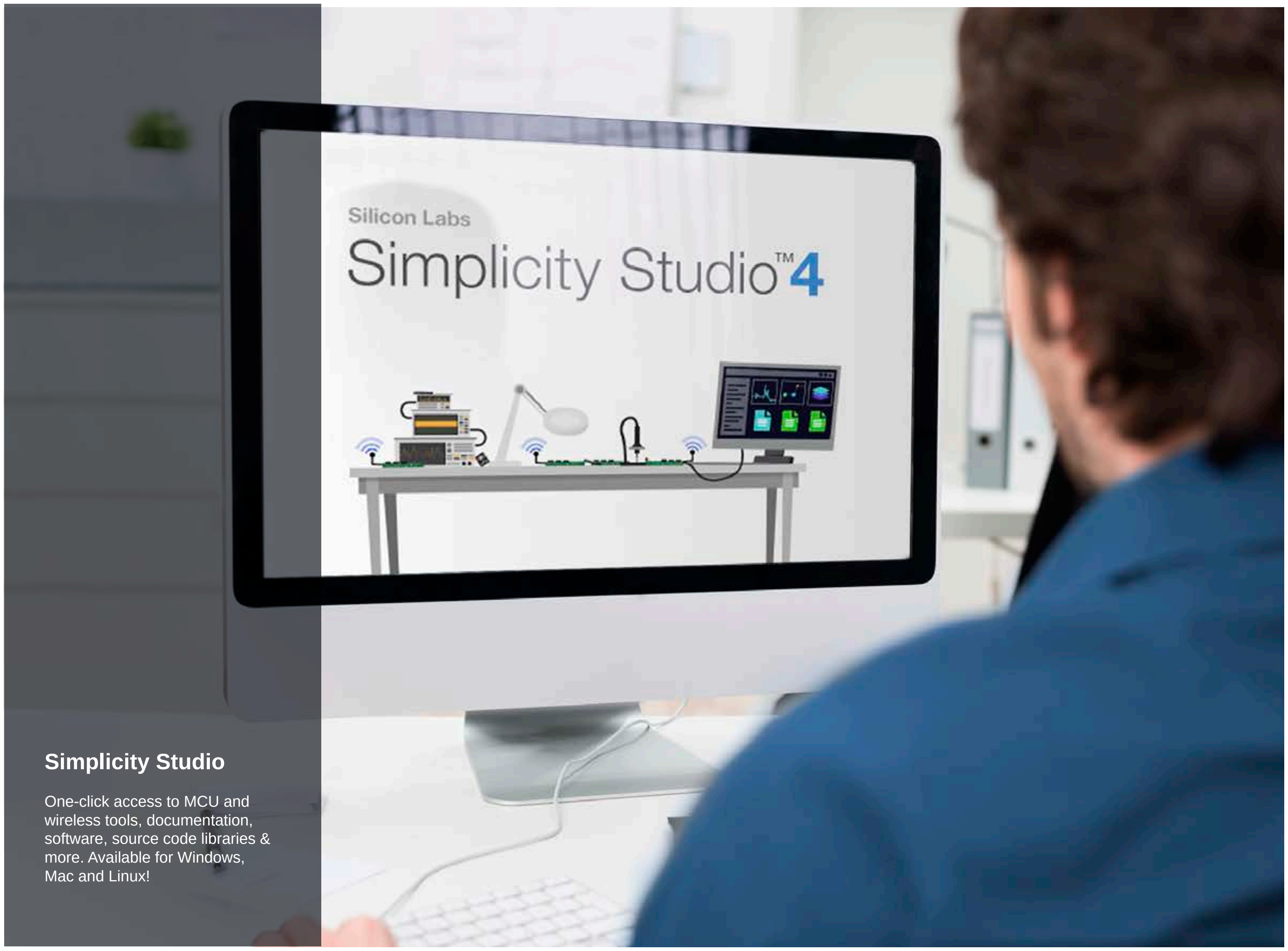
Enable the high voltage monitor threshold:

```
SI32_VMON_0->CONTROL.VDDHITHEN = 1; // non-HAL

SI32_VMON_A_select_vdd_high_threshold(SI32_VMON_0); // HAL
```

5.2. AHB Clock

1. If operating from an external crystal, be advised that crystal performance is susceptible to electrical interference and is sensitive to layout and changes in temperature. If the system is operating in an electrically noisy environment, use one of the internal oscillators or an external CMOS clock.
2. If operating from the external oscillator, switch to one of the internal oscillator during Flash write or erase operations. The external oscillator can continue to run, and the core can switch back to the external oscillator after the Flash operation completes.



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