SMART ARM-based Microcontroller

# AT10839: SAM L22 I2C Slave Bootloader

APPLICATION NOTE

## Introduction

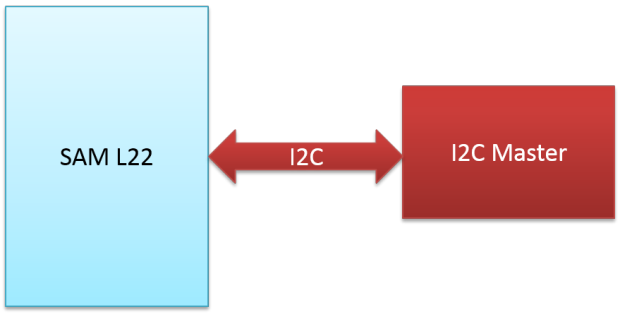
Many electronic designs evolve rapidly and there is a growing need for being able to update products, which have already been shipped or sold. Microcontrollers that support boot loader functionalities facilitates updating of the application flash section without the need of an external programmer. These microcontrollers are of great use in situations where the application has to be updated on the field. The bootloader may use various interfaces such as SPI, UART, $I^2C$, and Ethernet, etc.

This application note describes how to make use of the In-System programming capability of the Atmel® | SMART SAM L22 series devices using an $I^2C$ slave interface.

### Features

- Application for self programming
- Uses $I^2C$ Slave interface
- $I^2C$ master sends the data to be programmed over $I^2C$ bus
- Resets the device after programming and starts executing application

**Figure -1 SAML22 $I^2C$ Slave Bootloader**

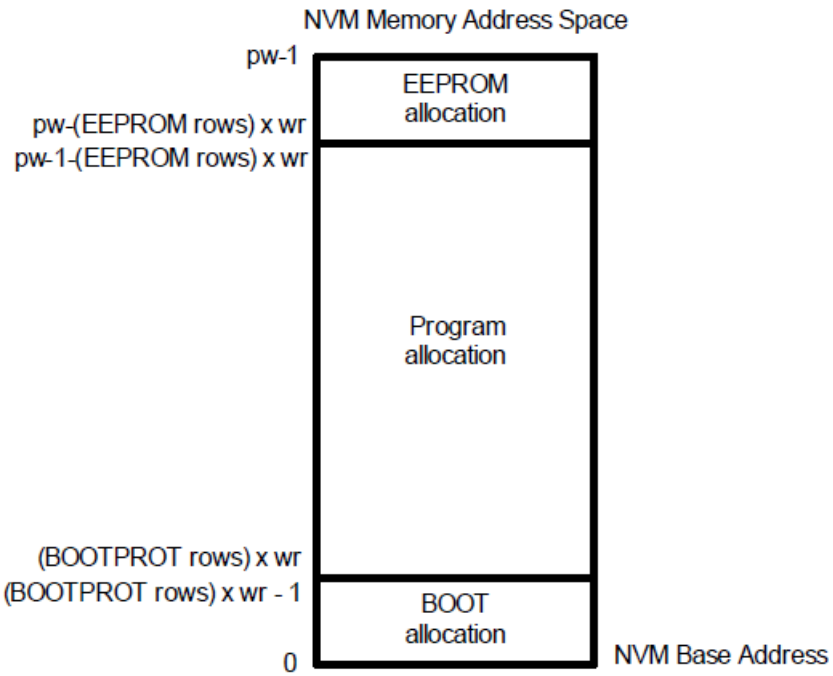


# Table of Contents

# 1. Program Memory Organization

The lower rows in the NVM main address space can be allocated as a boot section by using the BOOTPROT fuses of the device. The BOOT memory section is protected both by the lock bit(s) corresponding to this address space, and the BOOTPROT[2:0] fuse.

**Figure 1-1 EEPROM and BOOT Allocation**




This bootloader implementation consumes less than 8KB of Flash memory. 8KB is 32 rows of NVM Memory space starting from address 0x00000000. BOOTPROT fuses on the device can be set to protect first 32 rows of the program memory, which are allocated for the BOOT section. So, the end user application should be generated with a starting address of 0x00002000.

## 2. Prerequisites

The end user application to be programmed into the program memory of the SAM L22 using the bootloader should be generated with starting address offset of 0x2000 (APP_START_ADDRESS).

# 3. Bootloader Process

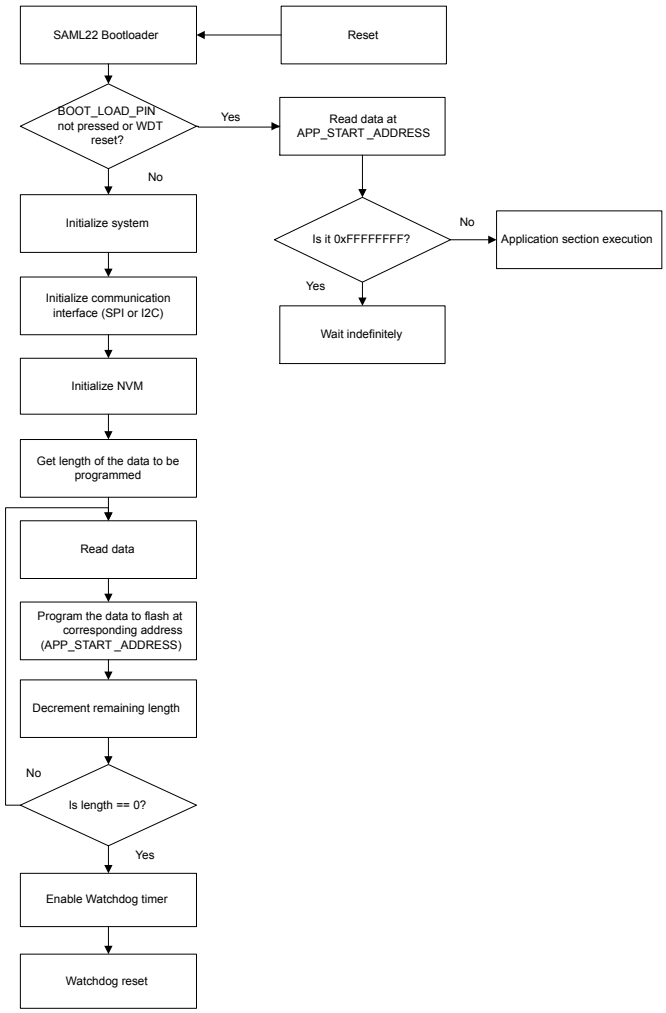
## 3.1. Boot Check

The bootloader is located at the start of the program memory and it is executed at each reset/power-on sequence. The Boot check sequence is as follows.

- The bootloader first checks the status of a user configurable BOOT_LOAD_PIN. If the pin is pulled low it continues execution in bootloader mode.
- Otherwise it reads the first location of the application section (0x00002000), which contains the stack pointer address
- Bootloader checks whether stack pointer address is set to 0xFFFFFFFF. If this is true, the application section is assumed to be empty and it enters an infinite loop.
- If not, it jumps to the application section and starts execution normally from the start of the loaded application

Configuration of the BOOT_LOAD_PIN and disabling of the Watchdog module in this boot mode check routine are made with direct peripheral register access to enable a quick decision to be made on the execution of the loaded application or bootloader.

**Figure 3-1 Bootloader Process**



## 3.2. Bootloader Process

The NVM module, board hardware, and system clocks are initialized according to the configurations in the `conf_clocks.h` and `conf_board.h` header files. Consequently, the I²C slave module is initialized with the configurations specified in the `conf_bootloader.h` header file. The I²C Master and Slave communication is explained below.

- The bootloader then waits for the reception of four bytes of data from the I²C master, which represent the length of the data to be programmed
- After transferring the length value, the I²C master continues sending the data to be programmed in blocks of NVMCTRL_PAGE_SIZE (64 bytes is the page size for SAM L22 devices)
- Data is received in the bootloader and programmed to program memory starting from APP_START_ADDRESS (offset of 0x2000 from NVM start address)
- An acknowledgment byte ‘s’ is transferred from I²C slave bootloader to I²C master to indicate it has received the data and finished programming it
- This is repeated until the entire length of data is programmed to program memory

This communication is relatively simple. No complex protocol is followed in this bootloader implementation. Hence there is no specific check performed on Length of application binary or application data received. I²C master has to ensure length of binary not exceeding flash size and data integrity etc. This simple implementation helps the users in selecting I²C Master to work with this I²C Slave bootloader. The master has to be programmed to work as per the above mentioned protocol. Refer SAM L22 Device datasheet for more information on NVM, SERCOM Module, etc.

## 3.3. Start Application

After the programming has completed, the bootloader code enables the Watchdog Timer with a timeout period of 256 clock cycles and waits in a loop for Watchdog to reset the device.

# 4. Hardware Setup

## 4.1. $I^2C$ Slave Lines

The SAM L22 device mounted on the SAM L22 Xplained Pro kit is used as the $I^2C$ slave. The $I^2C$ master should be connected to PB30(PIN11) and PB31(PIN12) on External header 2 (EXT2) of the SAM L22 Xplained Pro kit.

## 4.2. Bootloader Enable Pin

The Push button (SW0) on this kit will be configured as BOOT_LOAD_PIN and LED0 will be used to display the bootloader status. LED0 will be ON when the device is in bootloader mode. In SAM L22 Xplained Pro kit, SW0 is connected to PC01 and PC27 is connected to LED0.

# 5. Application Specific Configurations

## 5.1. Clock Configuration

The SAM L22 Internal 16MHz Oscillator (OSC16M in 4MHz mode) is used as the system clock in this implementation, without any prescaling. The Peripheral buses also run at 4MHz without any prescaling.

A 1kHz oscillator clock from internal ultra-low power 32kHz is configured and enabled in 32kHz Oscillator Controller to use WDT.

These configurations should be made in the `conf_clocks.h` header file.

## 5.2. Bootloader Configuration

The application starting address to program the user application is configured to 0x00002000, because the bootloader only uses first 32 rows of the program memory. The BOOT_LOAD_PIN and BOOT_LED are set to SW0 and LED0 on the kit.

These configurations should be made in the `conf_bootloader.h` header file.

In addition to the above configurations, $I^2C$ slave configurations are made in the `conf_bootloader.h` header file. SERCOM5 is to be used for $I^2C$ slave interface with slave address of 0x15. PIN11 and PIN12 on external header 2 (EXT2) are configured as SDA and SCL respectively.

# 6. Revision History

| Doc. Rev. | Date | Comments |
|---|---|---|
| 42623A | 11/2015 | Initial document release. |

**Atmel Corporation** 1600 Technology Drive, San Jose, CA 95110 USA **T:** (+1)(408) 441.0311 **F:** (+1)(408) 436.4200 | **www.atmel.com**