
Atmel AT03304: SAM D20 I²C Slave Bootloader

SAM D20**Description**

As many electronic designs evolve rapidly 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, are of great use in situations where the application has to be updated on the field. The boot loader may use various interfaces like SPI, UART, I²C, Ethernet etc.

This application note describes how to make use of the In-System programming capability of the Atmel® SAM D20 series devices using an I²C slave interface.

Features

- Application for self programming
- Uses I²C Slave interface
- I²C master sends the data to be programmed over I²C bus
- Resets the device after programming and starts executing application

Figure 1. SAM D20 I²C Slave Bootloader

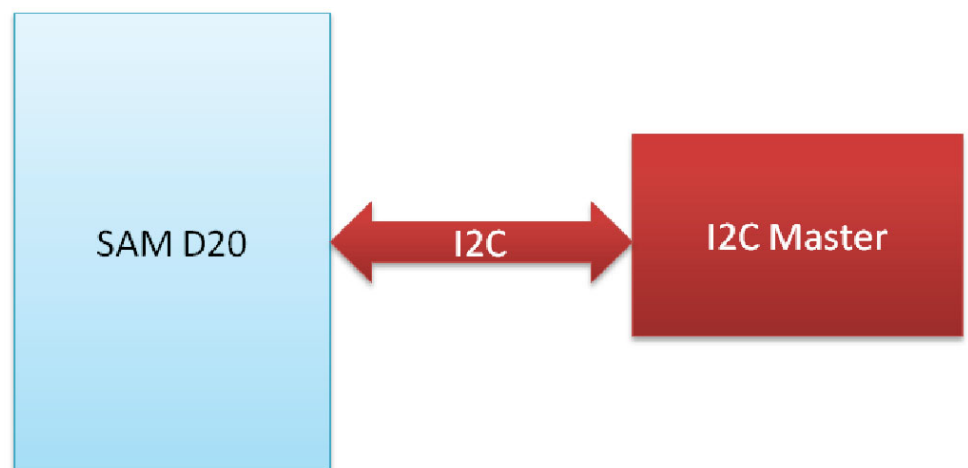


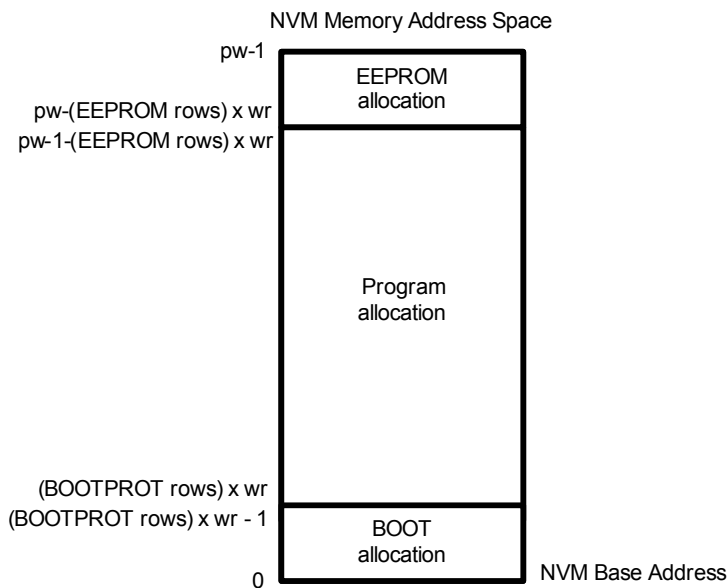
Table of Contents

1. Program Memory Organization	3
2. Prerequisites	3
3. Bootloader Process	3
3.1 Boot Check	3
3.2 Bootloader Process	5
3.3 Start Application	5
4. Hardware Setup	5
4.1 I ² C Slave Lines	5
4.2 Bootloader Enable Pin	5
5. Application Specific Configurations	5
5.1 Clock Configuration	5
5.2 Bootloader Configuration	5
6. Doxygen Documentation	6
7. Revision History	7

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 approximately 8KB of Flash memory, which are 32 rows of Program 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

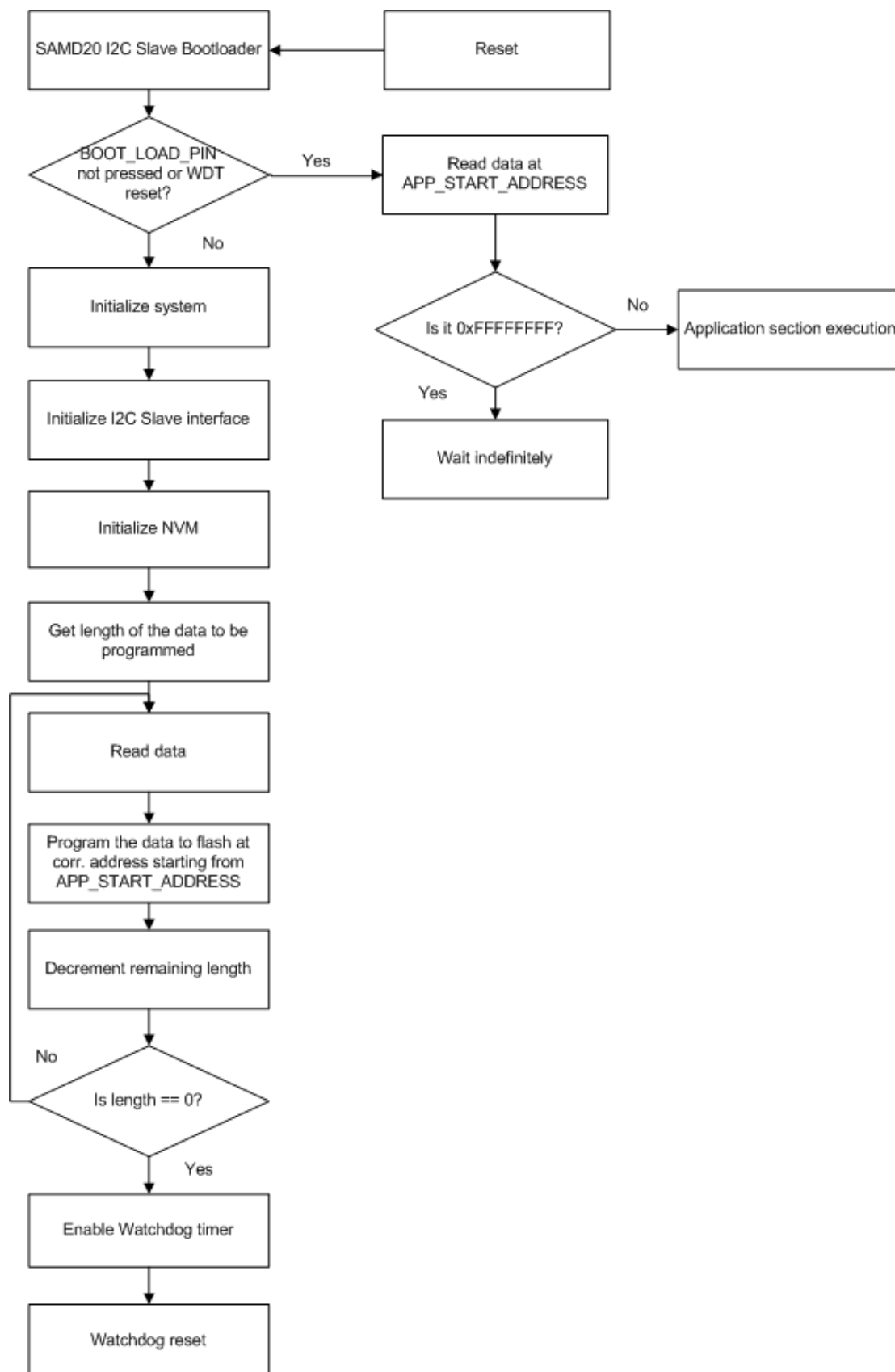
The end user application to be programmed into the program memory of the SAM D20 using the bootloader should be generated with starting address offset of 0x2000.

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 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 and checks whether it 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 bootloader then waits for the reception of four bytes of data from 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`. Data is received in the bootloader and programmed to Program memory starting from `APP_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.

3.3 Start Application

Once the programming is completed, the bootloader 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 I²C Slave Lines

The SAM D20 mounted on the SAM D20 Xplained Pro kit is used as the I²C slave. The I²C master should be connected to PIN11 and PIN12 on External header 2 (EXT2) of the SAM D20 Xplained Pro board.

4.2 Bootloader Enable Pin

The 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.

5. Application Specific Configurations

5.1 Clock Configuration

The SAM D20 Internal 8MHz Oscillator (OSC8M) is used as the system clock in this implementation, without any prescaling. The APBA and APBB bus also run at 8MHz without any prescaling.

The Internal 32kHz Oscillator (OSC32K) is enabled and configured as clock source for the Generic Clock Generator 4 channel. The Watchdog timer is configured with Generic Clock Generator 4 as the clock source.

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, I²C slave configurations are made in the `conf_bootloader.h` header file. SERCOM2 is to be used for I²C slave interface with slave address of `0x15`. PIN11 and PIN12 on external header 2 (EXT2) are configured as SDA and SCL respectively.

6. Doxygen Documentation

All source code is prepared for automatic documentation generation using Doxygen. Doxygen is a tool for generating documentation from source code by analyzing the source code and using special keywords. For more details about Doxygen, visit <http://www.doxygen.org>.

7. Revision History

Doc. Rev.	Date	Comments
42205A	10/2013	Initial document release

**Atmel Corporation**

1600 Technology Drive
San Jose, CA 95110
USA

Tel: (+1)(408) 441-0311

Fax: (+1)(408) 487-2600

www.atmel.com

Atmel Asia Limited

Unit 01-5 & 16, 19F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
HONG KONG

Tel: (+852) 2245-6100

Fax: (+852) 2722-1369

Atmel Munich GmbH

Business Campus
Parking 4
D-85748 Garching b. Munich
GERMANY

Tel: (+49) 89-31970-0

Fax: (+49) 89-3194621

Atmel Japan G.K.

16F Shin-Osaki Kangyo Building
1-6-4 Osaki, Shinagawa-ku
Tokyo 141-0032
JAPAN

Tel: (+81)(3) 6417-0300

Fax: (+81)(3) 6417-0370

© 2013 Atmel Corporation. All rights reserved. / Rev.: 42205A-SAM-10/2013

Atmel®, Atmel logo and combinations thereof, Enabling Unlimited Possibilities®, and others are registered trademarks or trademarks of Atmel Corporation or its subsidiaries. Other terms and product names may be trademarks of others.

Disclaimer: The information in this document is provided in connection with Atmel products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Atmel products. EXCEPT AS SET FORTH IN THE ATMEL TERMS AND CONDITIONS OF SALES LOCATED ON THE ATMEL WEBSITE, ATMEL ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS AND PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ATMEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Atmel makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and products descriptions at any time without notice. Atmel does not make any commitment to update the information contained herein. Unless specifically provided otherwise, Atmel products are not suitable for, and shall not be used in, automotive applications. Atmel products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.