APPLICATION NOTE

AT09002: Atmel SAM I²C - SPI Bootloader

Description

This document is the Atmel® dual interface SAM I²C - SPI bootloader application note. It describes how to use the pre-programmed bootloader available for the following devices:

- SAMG51
- SAMG53
- SAMG54
- SAMG55
- SAMD20

Check the order code to verify that the bootloader is pre-programmed.

Note that there are differences in the various implementations. Those differences are described where applicable.

Features

- Application Update
- Force Bootloader Entry
- I²C and SPI Communication Interfaces
- PC Tools for Update Package Generation

Table of Contents

1. Entering Bootloader Mode

1.1 Introduction

Bootloader mode is entered automatically on power-up if the firmware on the device is missing or corrupt.

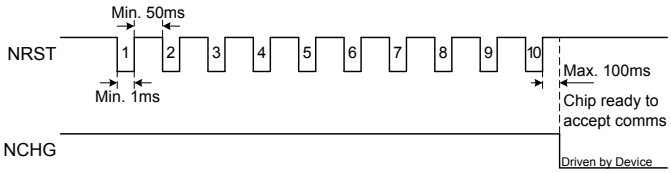
If the firmware is valid and the device is running in application mode, it can be put into bootloader mode by toggling the NRST line (see “NRST Toggling Force Bootloader Sequence” on page 3).

1.2 NRST Toggling Force Bootloader Sequence

With this sequence the NRST line is toggled ten times in a row without communicating via the I²C or SPI bus between the resets.

For SAMD20; if the NRST line is toggled more than ten times in a row, the device will exit bootloader mode and jump to the application if it is present and valid. For other devices; the device will stay in bootloader mode for consecutive toggles.

Figure 1-1. NRST Toggling Force Bootloader Sequence



1.2.1 Special Considerations

If the device in application mode is regularly reset by the host using its NRST line, it will enter bootloader mode after the force bootloader sequence detection: ten reset cycles. To avoid that, the application is required to clear the bootloader reset counter after a truly requested application entry.

The bootloader reset counter is located at the first address of the device SRAM and its clear value is: 0x00DA1981.

In addition to the bootloader reset counter, the device SRAM also includes a software reset type (start of SRAM + two words) to be used by the bootloader. This is a copy of the device reset cause register.

Note that for SAMD20 the interface select line (same as SPI slave select) is used to select whether to use SPI or I²C. If the interface select is low during startup, SPI is activated. If the interface select is high during startup, I²C is activated. If I²C mode is used, the interface select pin can be left unconnected since an internal pull-up will make the line go high. The selected level should be held for 50ms. For SPI mode the line will have to be released high, before communication can commence.

2. Communication Interfaces

2.1 Interface Selection

For SAMG, the interface used by the host to unlock the device as detailed in “Verify Bootloader Entry” on page 5 (either I²C or SPI) is considered as the selected host communication interface for the rest of bootloading process.

For SAMD20, the slave select pin is polled during startup. If the slave select pin is low during startup, SPI is selected. To ensure that the correct mode is entered, the slave select pin should be held low for 50ms after reset has been released. The pin has to be released before starting communication. To use I²C, hold the slave select high during startup.

2.1.1 I²C Configuration

The device responds to the specific I²C address 0x26. The SAMG devices are compatible with I²C Fast mode and I²C High Speed mode operation. The SAMD20 devices are compatible with I²C Fast mode only.

2.1.2 SPI Configuration

To allow correct communication between host and device, SPI needs to be configured with 8-bit data length and SPI mode 1.

3. Updating Application

3.1 Verify Bootloader Entry

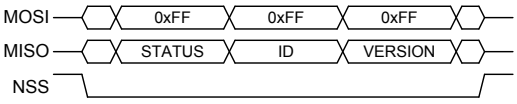
Upon bootloader start-up, the device is ready to send the extended status. The bootloader extended status consist of a 3-byte data as shown below:

1. The first byte is the standard status byte.
2. The second byte is the bootloader ID code. Refer to “Device Configuration” on page 10 for further information.
3. The third byte is the bootloader version.

Note that this 3-byte extended status read is necessary only once to obtain extended bootloader information. The rest of the application update process may be performed using standard status reads.

Figure 3-1. I²C Extended ID Mode Reading

Figure 3-2. SPI Extended ID Mode Reading



3.2 Unlocking the Device

Once the device has entered the bootloader mode, as described in “Entering Bootloader Mode” on page 3, the host needs to send the Application Update Unlock Command (0xDC followed by 0xAA as shown below) to the device to unlock it and enter the Application Update mode. Any other command will force application execution for SAMD20. On SAMG, the bootloader will wait for another unlock attempt if the CRC is invalid, a bootloader reset is required otherwise.

Figure 3-3. Application I²C Update Unlock Command

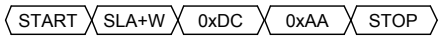
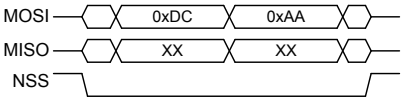


Figure 3-4. Application SPI Update Unlock Command



For SAMG55 and SAMD20 an application erase is executed after the unlock command has been received. The embedded length field in the application will be used to only delete the application area. Keep in mind that erases are performed in regions, so some additional pages might be erased for alignment.

3.3 Application Update Mode

3.3.1 Introduction

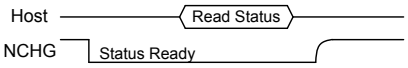
In application update mode, the device accepts multi-byte update commands when written to over the communication interface, and returns a single status byte when it is read.

3.3.2 Bootloader Status

As shown in Figure 3-5, “CHG Line Status Read Behavior” on page 6 and Figure 3-8, “Application Update Mode Program Flow” on page 7, the device indicates a transition between the states by asserting the NCHG line (i.e. by setting it low). Once the status has been read, the device releases the NCHG line (i.e. the device floats the NCHG line and then the pull-up returns it high). It then runs the processes associated with that

state. The device waits until the status is read, so that the host cannot miss a state transition. The NCHG line will go low once it is ready to receive a command as well.

Figure 3-5. CHG Line Status Read Behavior



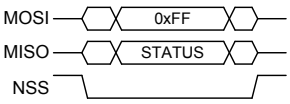
Note It is very important that the host checks the status of the NCHG line in order to know when it needs to read the status of the device. Checking status while the NCHG line is de-asserted is not supported.

3.3.2.1 Bootloader Standard Status

The bootloader standard status consist of a 1-byte data (as shown below) giving the state of the bootloader as detailed in Table 3-1, “Application Update Mode Status Codes” on page 7.

Figure 3-6. I²C 1-Byte Status Code Reading

Figure 3-7. SPI 1-Byte Status Code Reading



3.3.3 Bootloader State Machine

The application update mode works as a state machine as detailed in the figure Figure 3-8, “Application Update Mode Program Flow” on page 7. Table 3-1, “Application Update Mode Status Codes” on page 7 describes the behavior of each state.

Figure 3-8. Application Update Mode Program Flow

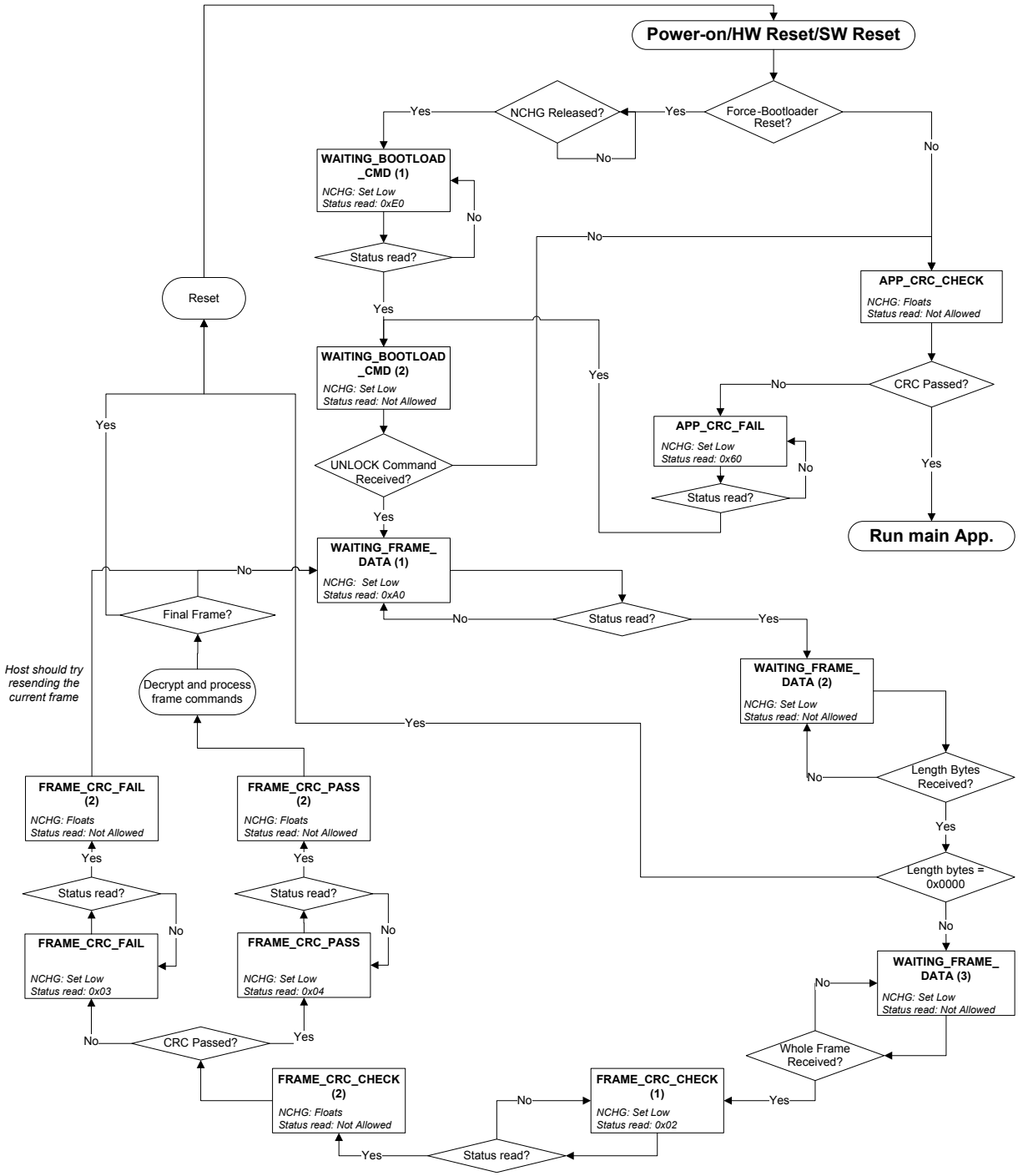


Table 3-1. Application Update Mode Status Codes

State	Data Transmission	Description
0xC0	Yes	**WAITING_BOOTLOAD_CMD.** The data returned by the device has bit 7 and 6 set. The remaining six least significant bits are part dependent and provide information about the bootloader version (see “Application Update Mode” on page 5). In this state, the device is awaiting the bootloader command. Sending any other byte than the bootloader command causes the device to leave bootloader mode.
0x80	Yes	**WAITING_FRAME_DATA.** The data returned by the device has bit 7 set. The remaining six least significant bits are part dependent and provide information about the bootloader version (see “Application Update Mode” on page 5).

State	Data Transmission	Description
		In this state, the device is awaiting frame data from the host. Once it has received a whole frame it moves to the CRC check state.
0x02	No	**FRAME_CRC_CHECK.** The host does nothing at this stage but waits for the CRC check result.
0x03	No	**FRAME_CRC_FAIL.** If the state machine reaches this state, the CRC check for the current frame has failed. The host should try resending the last frame once the device returns to the WAITING_FRAME_DATA state.
0x04	No	**FRAME_CRC_PASS.** If the state machine reaches this state, the current frame data has passed the CRC check, and the device is proceeding to process the frame records. The host should wait until the device returns to the WAITING_FRAME_DATA state before sending the next frame.
0x40	No	**APP_CRC_FAIL.** The data returned by the device has bit 6 set. The remaining six least significant bits are part dependent and provide information about the bootloader version (see “Application Update Mode” on page 5). If the state machine reaches this state, the CRC check for the currently stored application code failed. The host can make the device enter bootloader mode to recover the firmware by sending the bootloader command sequence.
0x06	No	**ERROR_DETECTED.** A flash error was detected during flash access. In this state, the device will leave bootloader mode, and the host can make the device enter bootloader mode again by using NRST toggling or NCHG and NRST sequence.

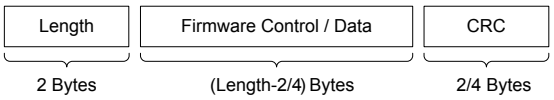
3.4 Data Frames

The application firmware is sent to the device as frames of data. Each frame (see Figure 3-9, “Data Frame Format” on page 8) consists of:

- A 2-byte length field, detailing the number of bytes to follow in the frame
- The firmware control / data field
- A CRC field, 4-byte for SAMD20, 2-byte for others

The maximum size of a frame is 532 (276 for SAMD20) bytes.

Figure 3-9. Data Frame Format



The application frames come from a firmware file generated using a tool provided by Atmel, refer to “Firmware File Generator” on page 9 for further details.

The final frame within a firmware file contains an embedded reset command that causes the device to reset itself when it has finished updating the application. After resetting, the device performs a CRC on the application, if CRC fails, it asserts the NCHG line and behaves according to incoming commands.

Alternatively, the host may cause the device to reset by sending a frame length word of zero.

If the device is reset part of the way through updating the application, the bootloader will re-enter bootloader mode since the application is corrupt.

4. Tools

The user bootloader package is delivered as a zip archive named `Atmel-42305-SAM-I2C-SPI-Bootloader_ApplicationNote_AT09002.zip`.

4.1 Firmware File Generator

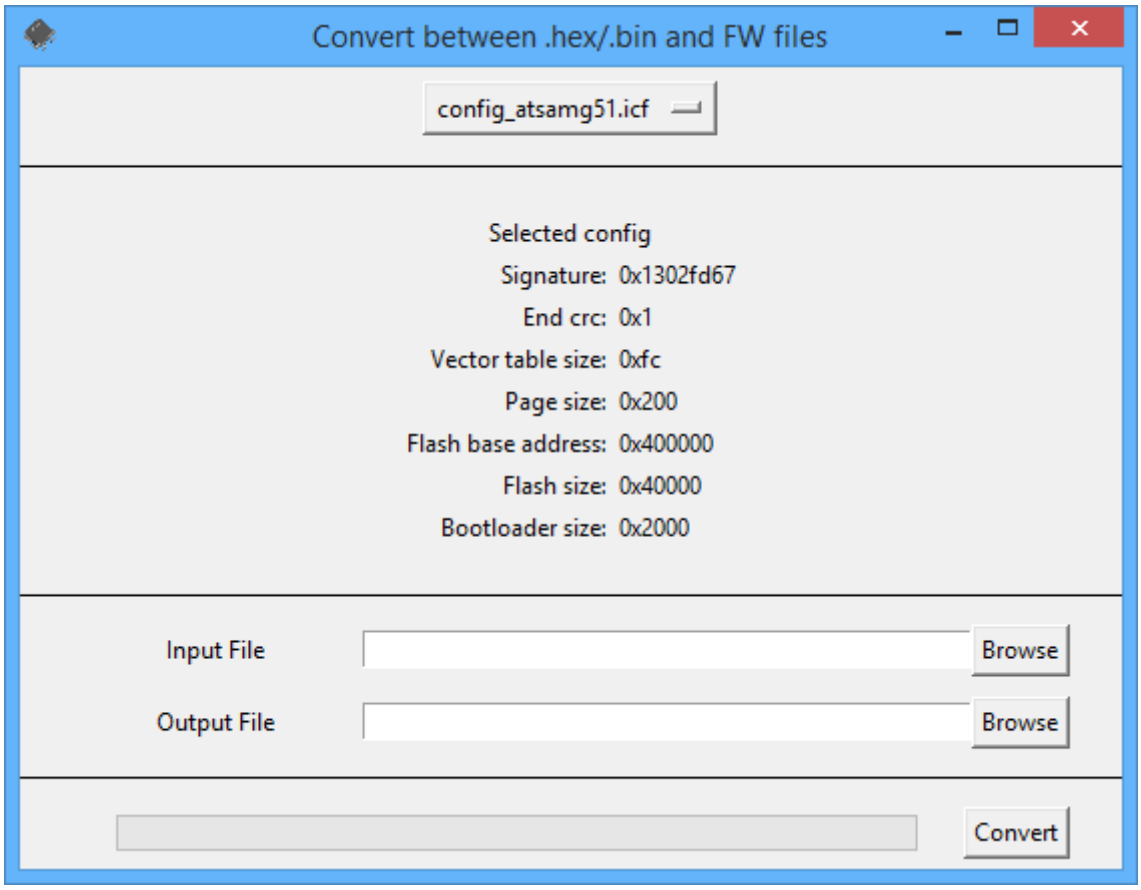
The firmware file is generated using the `hex2fw.py` tool. This tool must be called in command line as follow and generates the firmware file:

hex2fw.py -c config\config_#device_name#.icf -i input_file.hex/.bin -o output_file.*

The `config_#device_name#.cfg` is the configuration file for the device target e.g. `config_samg53.icf`.

The tool can also be used with a GUI like depicted in Figure 4-1, “hex2fw Graphical User Interface” on page 9.

Figure 4-1. hex2fw Graphical User Interface



4.2 Application Linker Scripts

Specific linker scripts must be made so that application build is aware of bootloader presence. See “Linker Script Configuration” on page 10 for device specific details. Some linker script examples are included in the zip file.

4.2.1 Reserved Code Area

For SAMG51, SAMG53, and SAMG54 8KB of flash is reserved for bootloader location. For SAMG55 the bootloader does not reserve any flash area. For SAMD20 the bootloader reserves 4KB of flash.

The CRC address is placed in flash after the vector table. The linker must be setup to not use that address. Make sure that the application code starts after the specified address.

4.2.2 Reserved Data Area

The first 12 bytes of the internal SRAM are reserved. The first word is the reset counter, second is software_reset_type, and the third is the reset cause.

5. Device Configuration

5.1 SAMG51/G53/G54

5.1.1 Firmware Configuration

Table 5-1. SAMG51/G53/G54 Bootloader Information

Parameter	Value	Comment
ID	0x24	Bootloader Identification Value
VERSION	0x02	Bootloader Code Version

5.1.2 Linker Script Configuration

Table 5-2. SAMG51/G53/G54 Linker Script Configuration

Memory	Start Address
FLASH	0x00402000
RAM	0x2000000C

CRC address is located at an offset of 0xFC from FLASH.

5.1.3 Hardware Configuration

Table 5-3. SAMG51/G53/G54 Hardware Configuration

Signal	Port #	Pin Type	Comment
NCHG	PA01	Open Drain Input/Output	Bootloader Handshake line, internally pulled up
SDA	PA03	Open Drain Input/Output	I²C Data line
SCL	PA04	Open Drain Input/Output	I²C Clock line
NSS	PA11	Input	SPI Slave Select line
MISO	PA12	Push-Pull Output	SPI Master In, Slave Out line
MOSI	PA13	Input	SPI Master Out, Slave In line
SCK	PA14	Input	SPI Clock line

5.2 SAMG55

5.2.1 Firmware Configuration

Table 5-4. SAMG55 Bootloader Information

Parameter	Value	Comment
ID	0x25	Bootloader Identification Value
VERSION	0x01	Bootloader Code Version

5.2.2 Linker Script Configuration

Table 5-5. SAMG55 Linker Script Configuration

Memory	Start Address
FLASH	0x00400000
RAM	0x2000000C

CRC address is located at an offset of 0x400 from FLASH.

5.2.3 Hardware Configuration

Table 5-6. SAMG55 Hardware Configuration

Signal	Port #	Pin Type	Comment
NCHG	PA01	Open Drain Input/Output	Bootloader Handshake line, internally pulled up

Signal	Port #	Pin Type	Comment
SDA	PA03	Open Drain Input/Output	I²C Data line
SCL	PA04	Open Drain Input/Output	I²C Clock line
NSS	PA11	Input	SPI Slave Select line
MISO	PA12	Push-Pull Output	SPI Master In, Slave Out line
MOSI	PA13	Input	SPI Master Out, Slave In line
SCK	PA14	Input	SPI Clock line

5.3 SAMD20

5.3.1 Firmware Configuration

Table 5-7. SAMD20 Bootloader Information

Parameter	Value	Comment
ID	0x40	Bootloader Identification Value
VERSION	0x02	Bootloader Code Version

5.3.2 Linker Script Configuration

Table 5-8. SAMD20 Linker Script Configuration

Memory	Start Address
FLASH	0x00001000
RAM	0x2000000C

CRC address is located at an offset of 0xFC from FLASH.

5.3.3 Hardware Configuration

Table 5-9. SAMD20 Hardware Configuration

Signal	Port #	Pin Type	Comment
NCHG	PA07	Open Drain Input/Output	Bootloader Handshake line, internally pulled up
SDA	PA08	Open Drain Input/Output	I²C Data line
SCL	PA09	Open Drain Input/Output	I²C Clock line
NSS	PA10	Input	SPI Slave Select line
IFSEL	PA10	Input	Interface Select line, internally pulled up
MISO	PA08	Push-Pull Output	SPI Master In, Slave Out line
MOSI	PA11	Input	SPI Master Out, Slave In line
SCK	PA09	Input	SPI Clock line

6. Special Considerations

The security bit is not set in factory on all devices. The bootloader does not have a mechanism for setting the security bit. The application should take care to set the security bit. After the initial upload of the application through the bootloader in the production line, the application will start and should automatically set the security bit if required.

The lockbit setting command of SAMD20 does only enable/disable the lockbits temporarily. It will be up to the application, or a second-stage bootloader, to handle the lockbits. The default lockbit values can not be changed when the security bit is set.

The bootloader does not utilize the watchdog timer, and will make sure to disable it. Some devices (e.g. SAMD20) have the possibility to set the fuses to permanently enable WDT. In this case, the bootloader might not work.

7. Document Revision History

Document revision	Date	Comment
42305B	05/2015	Corrected hex2fw package details and linker script configuration.
42305A	08/2014	Initial document release

Atmel | Enabling Unlimited Possibilities®

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