APPLICATION NOTE

AT07175: SAM-BA Bootloader for SAM D21

Atmel SAM D21

Introduction

Atmel® SAM Boot Assistant (Atmel SAM-BA®) allows In-System Programming (ISP) from USB or UART host without any external programming interface. Unlike existing SAM products, ROM monitor is not available in SAM D21 and SAM-BA will be loaded in Flash memory.

This SAM-BA version is compatible with existing SAM-BA software tools but has some differences compared to other SAM devices. These differences are explained in this document.

This application note complements the SAM-BA user guide and explains how the SAM-BA should be used on a SAM D21 design.

Features

- Allows to program, verify and secure an Atmel SAM D21 device without debugger
- USB and UART connection
- Allows the end user to update application firmware from bootloader
- Configurable I/O start condition
- Source code available, can be customized to user's needs
- Compatible with SAM-BA v2.13 or above

1 Requirements

1.1 Hardware Requirements

The Atmel SAM D21 SAM Boot Assistant (SAM-BA) supports serial communication via UART or USB device port.

Table 1-1. UART Mode Requirements

Signal name	Recommended pin connection	Description
PA23	Connect to host (115200, 8, N, 1)	SERCOM3 PAD1 (UART RXD)
PA22	Connect to host (115200, 8, N, 1)	SERCOM3 PAD0 (UART TXD)

Table 1-2. USB Mode Requirements

Signal name	Recommended pin connection	Description
PA25	Connect to host	USB D+ pin
PA24	Connect to host	USB D- pin

Table 1-3. Hardware Bootloader Entry

Signal name	Recommended pin connection	Description
PA15	Connect to host or “Bootloader access switch”	The bootloader will check PA15 on reset to determine if the bootloader monitor shall start. This feature allows the end user to reprogram the device even if the application is corrupted or unable to start the SAM-BA monitor

1.2 Software Requirements

1.2.1 Application Constraints

Unlike existing SAM products using a ROM monitor, on the Atmel SAM D21 SAM-BA is stored in flash memory at 0x00000000 and started on reset. For SAM-BA with only one interface enabled (either USB or UART), it is stored in flash memory from 0x0000 – 0x1000. For SAM-BA with both USB and UART interfaces enabled, the firmware size exceeds 4kB. So flash region 0x0000 – 0x2000 (8kB) will be allocated for SAM-BA.

SAM-BA bootloader is not factory programmed on SAM D21 devices and has to be programmed using an external programmer. Since SAM-BA will be stored in flash memory, if the application requires the entire flash space and does not need the bootloading feature, SAM-BA can be erased using external programmer.

To use SAM-BA together with an application, the user needs to link the application starting at 0x1000 when only one interface (either USB or UART) is enabled and at 0x2000 when both USB and UART interfaces are enabled. The procedure to modify the start address in an IAR™ project and an Atmel Studio project is explained in Atmel AT04189: UART Based SAM-BA Bootloader for SAM D20.

Figure 1-1. Memory Map of ATSAMD21J18 with an Application and SAM-BA with both USB and UART

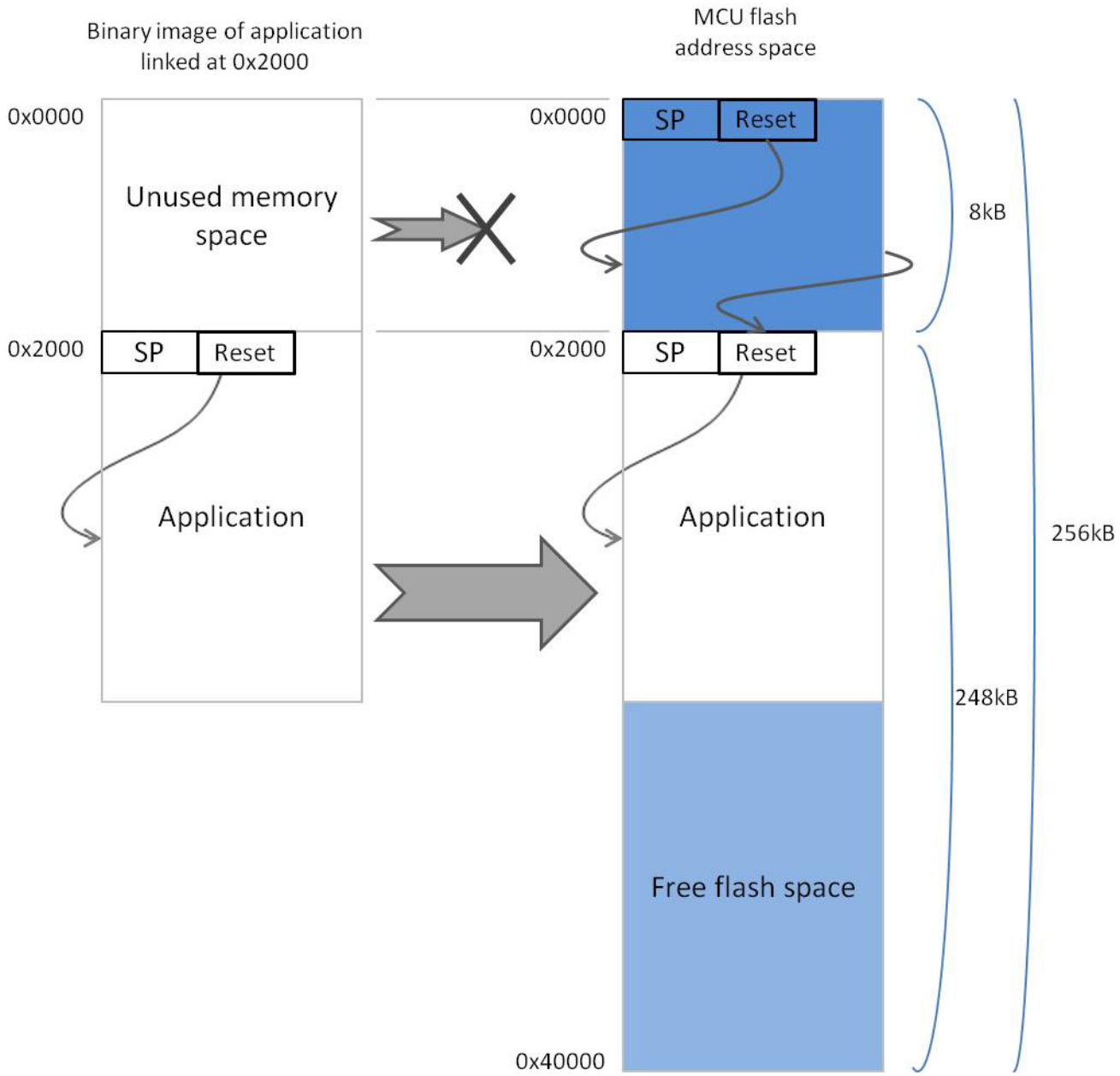
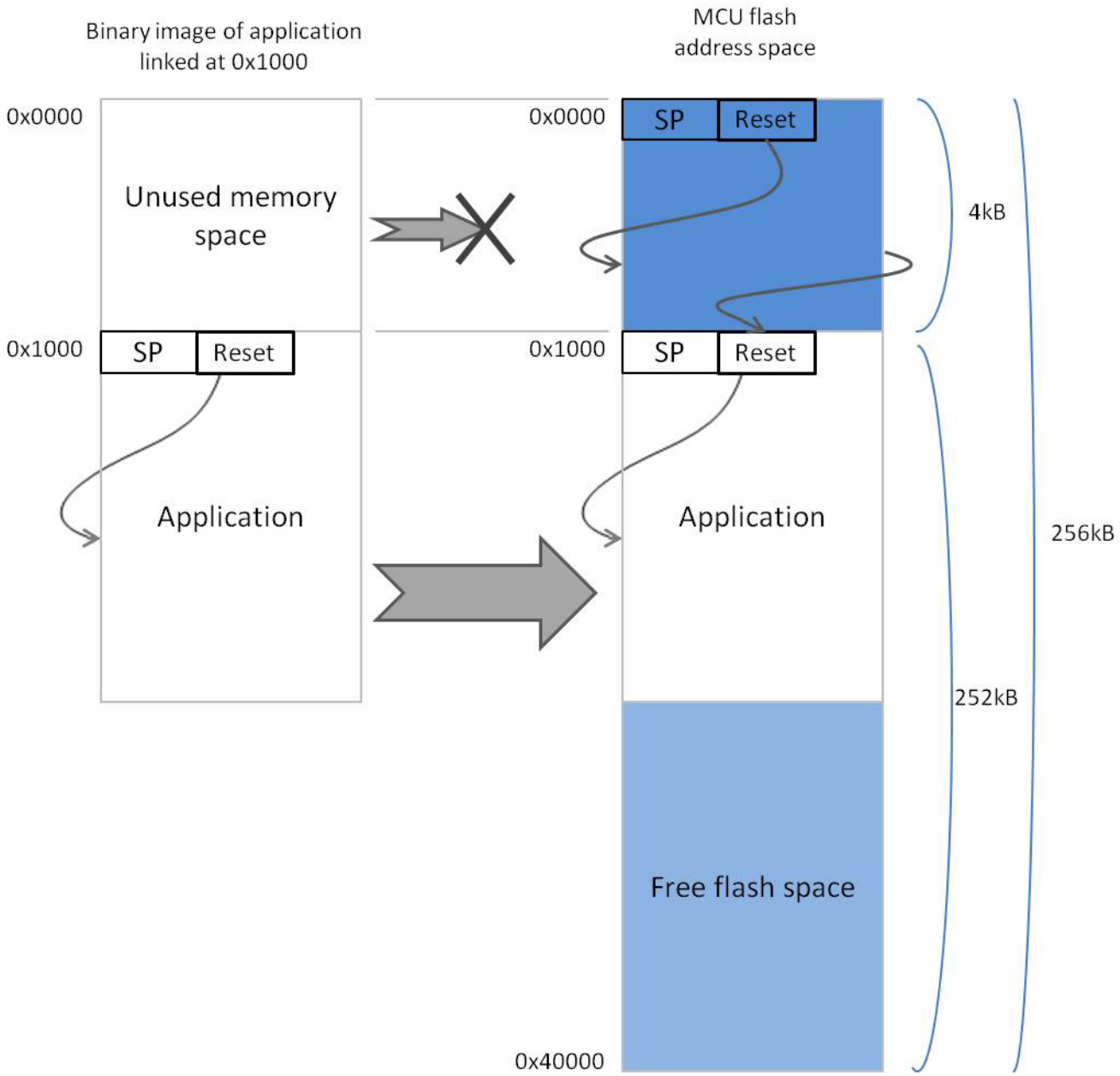


Figure 1-2. Memory Map of ATSAMD21J18 Device with an Application and SAM-BA with either USB or UART



When loading the application binary image to the device, only the part of flash after the SAM-BA should be programmed.

Any attempt to write the SAM-BA region using SAM-BA commands will be aborted and will throw an error.

2 Using the Bootloader

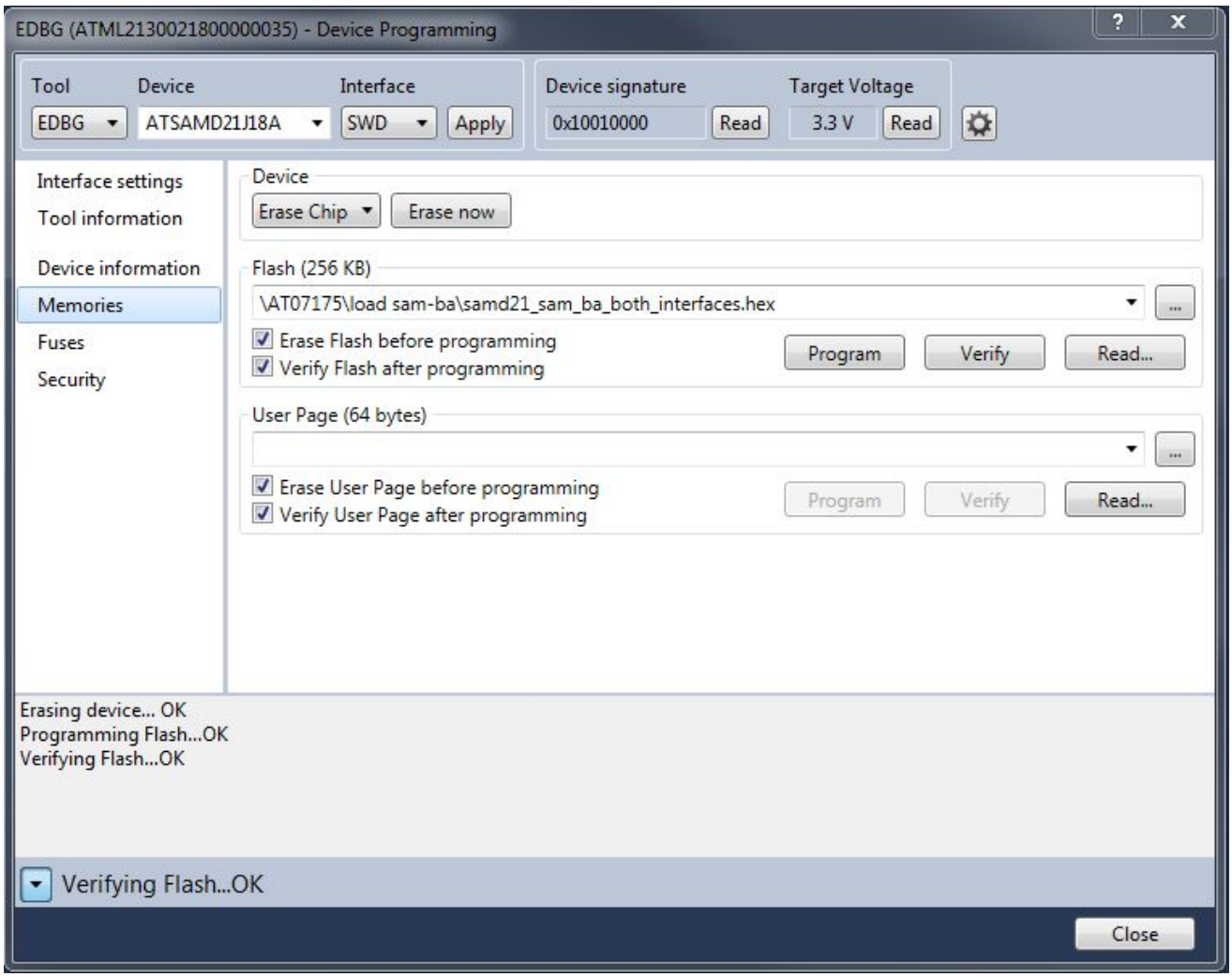
2.1 Programming the Bootloader

Programming the Atmel SAM Boot Assistant can be done using SWD debugger:

A. In Atmel Studio, open “Tools\Device Programming”, select debugger, and Apply.
B. From the ‘Memories’ tab, launch a chip erase.
C. Fill the path to SAM-BA image in the Flash box and program.

Note: SAM-BA images in folder ‘load sam-ba’ in AT07175.zip are built for ATSAMD21J18 device. For other devices, the SAM-BA monitor has to be rebuilt. Refer Chapter 4 for more details.

Figure 2-1. Device Programming



Alternatively command-line programming tool ‘atprogram’ can also be used for programming the ‘hex file.

Example: atprogram –t edbg –i swd –d atsamd21j18a program –f samd21_sam_ba_both_interfaces.hex.

2.2 Entering the Bootloader

Like other Atmel SAM devices, the SAM D21 SAM-BA relies on a monitor; this monitor is entered according to the boot process conditions described in Figure 2-2, Figure 2-3, and Figure 2-4.

SAM-BA monitor activation can be requested in one of the following ways:

- External condition: Reset the part and make sure the Hardware Bootloader Entry pin (PA15) is pulled low when reset is released. A common usage is to use a push button accessible by the user as a bootloader trigger. The user simply has to hold the push button when powering up the device.
- Internal condition: On erased devices or when the application reset vector (@Application start address + 4) is blank (0xFFFFFFFF)

Figure 2-2. Boot Process of Atmel SAM-BA using both USB and UART

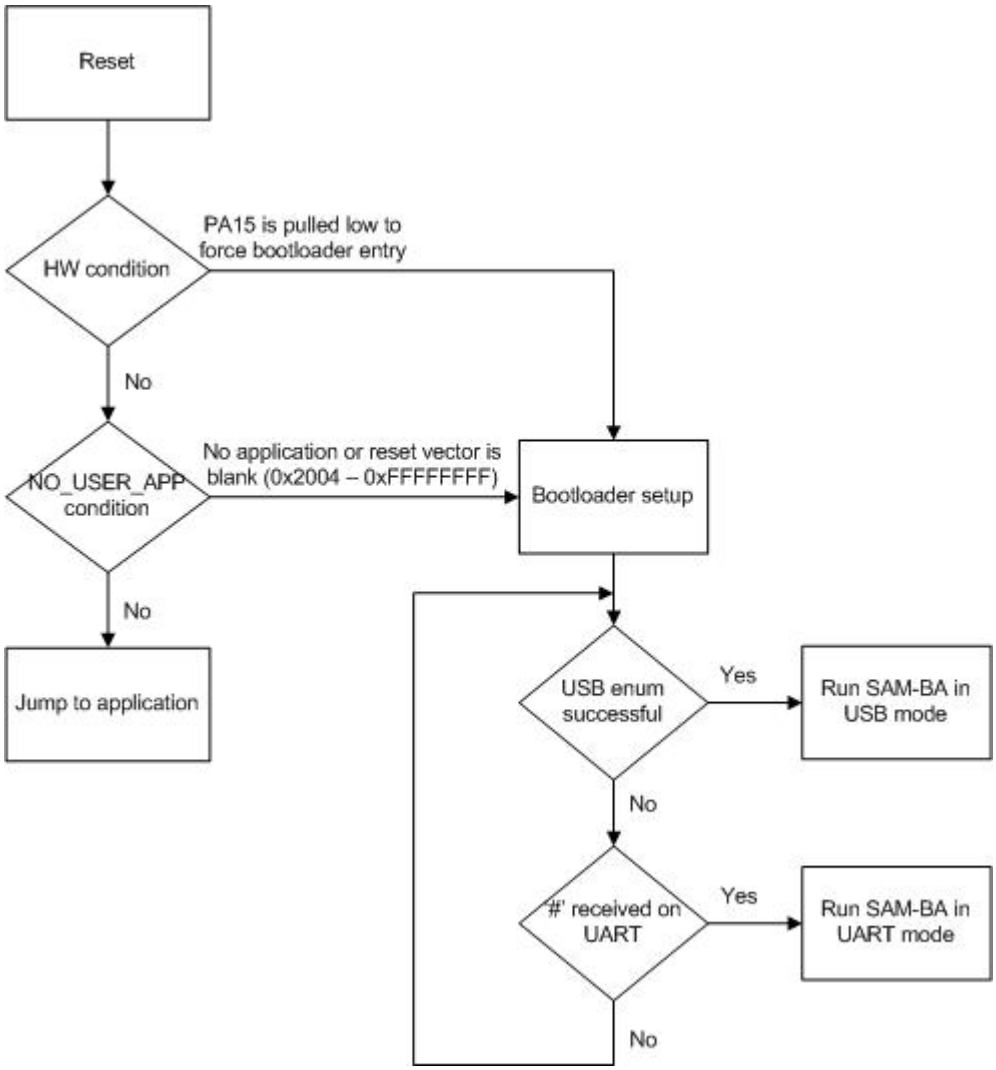


Figure 2-3. Boot Process of Atmel SAM-BA using USB

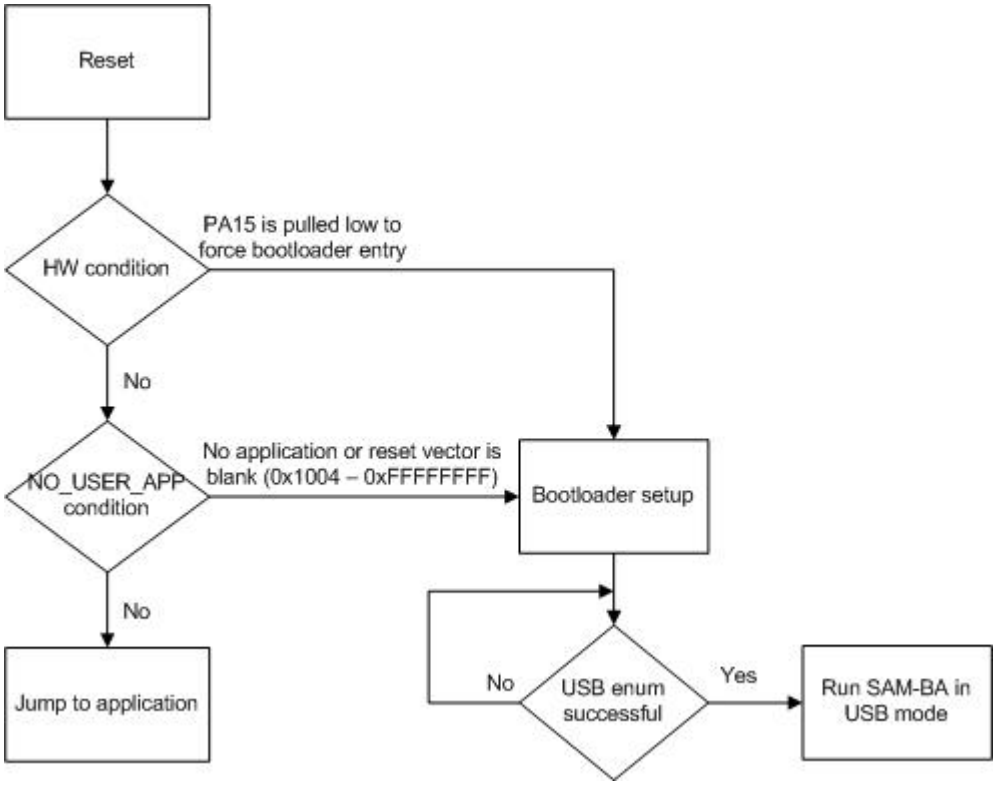
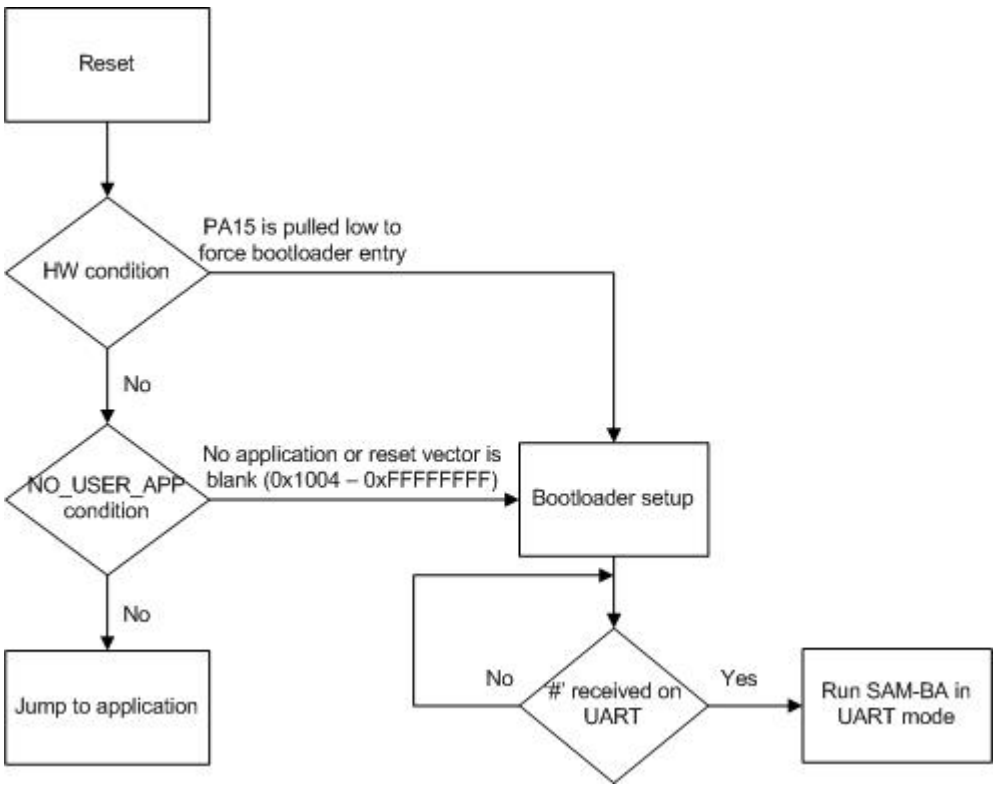


Figure 2-4. Boot Process of Atmel SAM-BA using UART



Code security concerns: When SAM-BA monitor is entered, it allows read and write access to the entire memory map of the device. It also allows the host to upload and execute software (applets) on the device.

After these preliminary steps, the SAM-BA monitor will enter a loop and test whether an USB enumeration has succeeded or a '#' (sharp) character is received on PA23 (SERCOM3 PAD1) line. The first satisfied condition will start the monitor in the respective mode: UART or USB.

2.3 Bootloader Configuration

SAM-BA monitor uses OSC8M as the system clock source (GCLK_GEN0). When USB interface option is selected, DFLL is enabled in USB Clock Recovery mode and is selected as clock source for USB module using GCLK_GEN1.

SERCOM3 connected to EDBG is used for UART communication and uses OSC8M as its clock source from GCLK Generator 0.

2.4 Selecting the Communication Interface

It is possible to compile SAM-BA monitor with one of the following options:

- Only UART interface enabled
- Only USB CDC interface enabled
- Both the UART and USB interfaces enabled and select the interface run-time

Table 2-1. SAM-BA Communication interface

Communication interface	SAM_BA_INTERFACE define value	Code footprint	Application start address
Only UART interface	SAM_BA_UART_ONLY	< 4kB	0x1000
Only USB CDC interface	SAM_BA_USBCDC_ONLY	< 4kB	0x1000
Both UART and USB CDC interfaces	SAM_BA_BOTH_INTERFACES	< 8kB	0x2000

Note: The application start address mentioned here is with SAM-BA monitor compiled in High (size) Optimization. Optimization setting must not be changed during compiling to maintain the bootloader size (and hence the application start address)

SAM_BA_INTERFACE define should be added in:

- IAR Project Options -> C/C++ Compiler -> Preprocessor -> Defined symbols
- IAR Project Options -> Assembler -> Preprocessor -> Defined symbols

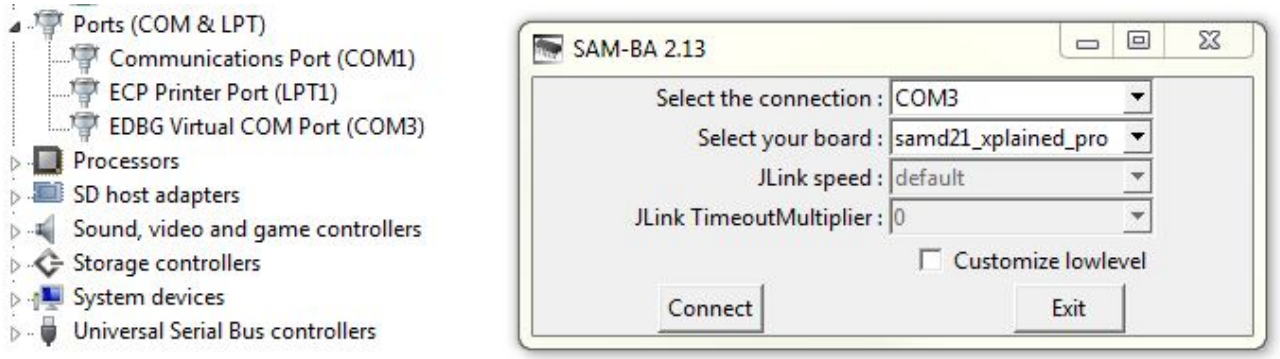
SAM-BA monitor stores the application start address at 0x20000000. SAM-BA PC application reads the application start address from this location.

2.5 Using the Bootloader with SAM-BA GUI

2.5.1 Connecting from SAM-BA PC Application

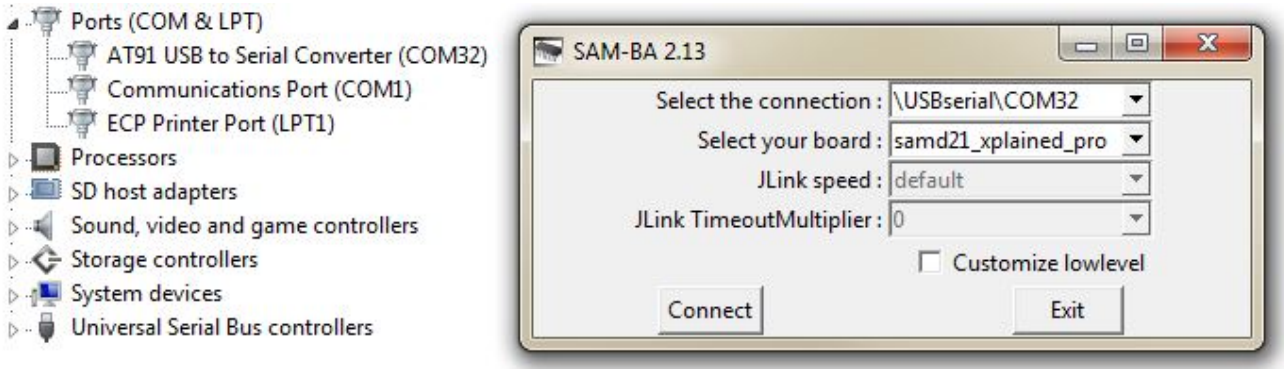
For using SAM-BA monitor with UART interface, connect SAM D21 Xplained Pro to the PC via DEBUG USB port using the micro-USB cable as the UART pins are connected to it.

Figure 2-5. Connecting from SAM-BA PC Application using UART



For using SAM-BA monitor with USB interface, connect SAM D21 Xplained Pro to the PC via TARGET USB port using the micro-USB cable.

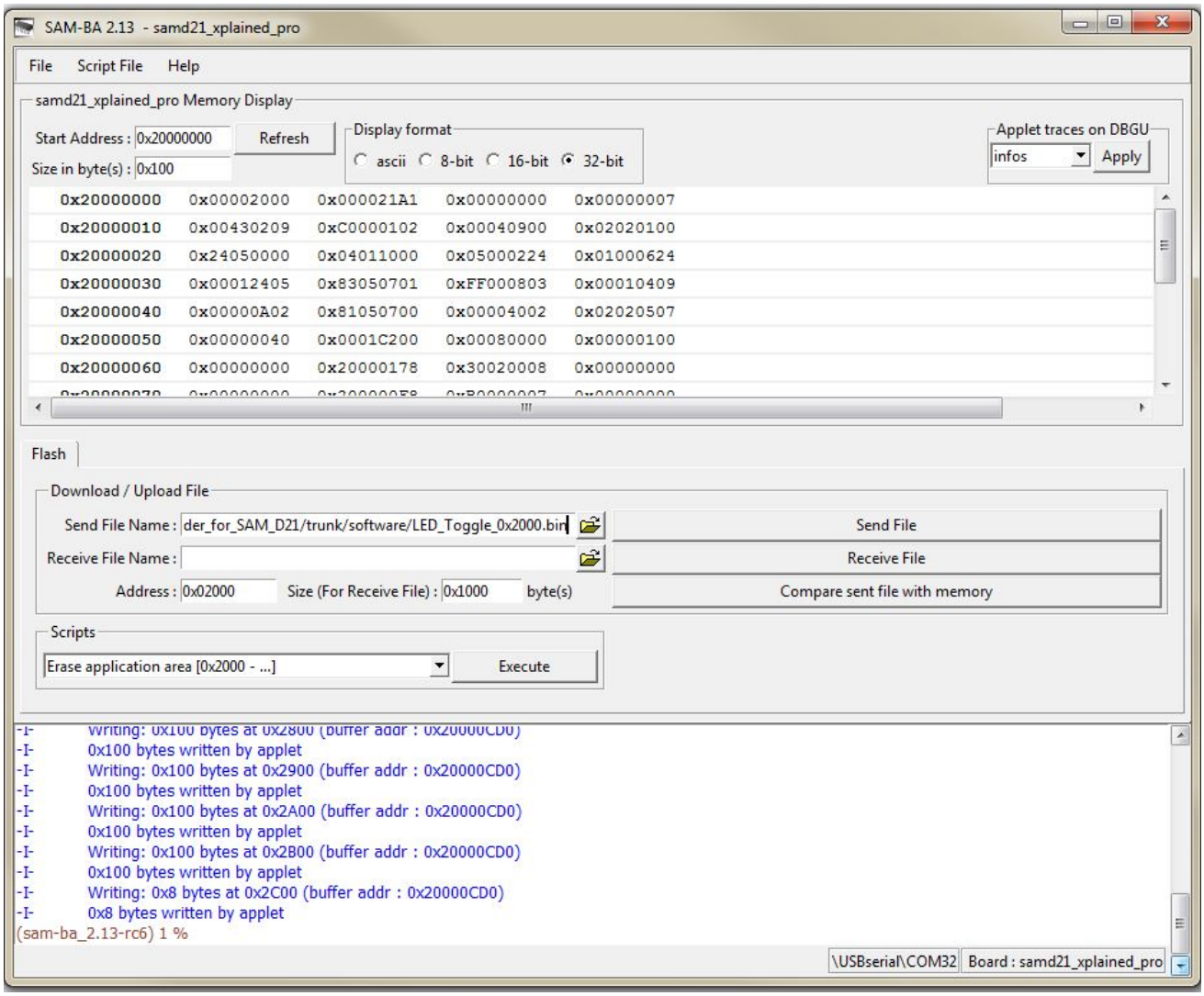
Figure 2-6. Connecting from SAM-BA PC Application using USB CDC



2.5.2 Flash Loading

Loading the flash contents is done by using the Flash tab. When uploading a program to flash memory, the start address needs to be above 0x1000 when using either USB or UART and above 0x2000 when using both USB and UART, otherwise the transfer will abort.

Figure 2-7. Flash Programming



2.5.3 Scripts

Atmel SAM-BA application comes with the following predefined scripts (see Table 2-2):

Table 2-2. Predefined Scripts

Script name	Description
Set Security Bit	Set the security bit to secure the device (Refer to NVMCTRL chapter in the device datasheet for more information)
Read Security Bit	Read the current security state
Erase application area	Erase all application code (SAM-BA part won't be erased)
Invalidate application	Erase first page of application
Read Fuses	Returns the values of fuse settings (Refer to NVM User Row Mapping section in the device datasheet for more information)
Read Lock Fuses	Read the current lock settings
Read DeviceID	Read the Device Identification register

Script name	Description
Set Lock Bit [0:15]	Set the specified lock bit to prevent any erasure of flash memory region (Refer to the NVMCTRL chapter in the device datasheet for more information)
Unlock all	Unlock all flash memory regions

3 Re-building SAM-BA Monitor

The SAM-BA monitor IAR project in AT07175.zip uses ATSAMD21J18A device by default. To use the SAM-BA monitor with other SAM D20 devices, the monitor IAR project has to be re-built with following changes in the project settings.

1. Open SAM-BA monitor IAR project and go to Project -> Options -> General Options -> Target -> Processor Variant -> select the required SAM D21 device.
2. Now go to C/C++ Compiler -> Preprocessor -> Defined Symbols -> edit the device name macro. For example, for ATSAMD21G18A device; change the macro to __ATSAMD21G18A__ (Note: Double underscores should be used).

4 Supported Devices

The following devices in SAM D21 family are supported by the SAM-BA version 2.13 or above.

1. ATSAMD21E15A
2. ATSAMD21E16A
3. ATSAMD21E17A
4. ATSAMD21E18A
5. ATSAMD21G15A
6. ATSAMD21G16A
7. ATSAMD21G17A
8. ATSAMD21G18A
9. ATSAMD21J15A
10. ATSAMD21J16A
11. ATSAMD21J17A
12. ATSAMD21J18A

5 Software Package Contents

The software package with this application note contains the following:

1. Source code of SAM D21 SAM-BA monitor and the associated IAR project.
2. Example binary image of an application which starts at 0x1000 and 0x2000 – The application continuously toggles LED0 on SAM D21 Xplained Pro.
3. SAM-BA monitor image hex with only UART interface enabled.
4. SAM-BA monitor image hex with only USB CDC interface enabled.
5. SAM-BA monitor image hex with both UART and USB interfaces enabled.

Note: To ensure minimal size for the bootloader, the source code as well as the project file is available only for the IAR compiler.

6 References

6.1 Device Datasheet

The device datasheet contains the block diagrams of the peripherals and details about implementing firmware for the device. It also contains the electrical specifications and expected characteristics of the device.

Datasheet is available on www.atmel.com in the Documents section of Atmel SAM D21 product page.

6.2 Atmel SAM-BA User Guide

The SAM-BA User Guide contains detailed reference information on how to use SAM-BA.

The user guide comes with the SAM-BA package available on www.atmel.com.

6.3 ARM Documentation on Cortex-M0+ Core

- Cortex®-M0+ Devices Generic User Guide revision r0p1
- Cortex-M0+ Technical Reference Manual revision r0p1

6.4 Atmel SAM-BA In-system Programmer

Latest Atmel SAM-BA In-system Programmer and related patches can be downloaded from

http://www.atmel.com/tools/ATMELSAM-BAIN-SYSTEMPROGRAMMER.aspx.

6.5 Atmel Studio

The latest version of Atmel Studio can be downloaded from http://www.atmel.com/tools/atmelstudio.aspx.

7 Revision History

Doc Rev.	Date	Comments
42366A	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**