AVR 8-bit Microcontrollers

AVR42788: SD Card Bootloader for XMEGA A1U

APPLICATION NOTE

Introduction

The SD card bootloader implemented in this application note allows flashing image from an external SD card. The SD card bootloader will be loaded in the Boot Loader flash section.

The user needs to write the image file to be flashed into the SD card and this bootloader will write the image to the Application flash section.

This application note explains how the SD card bootloader is implemented and how it should be used with the Atmel® XMEGA® A1U Xplained Pro.

Feature

- Application for self-programming on Atmel AVR® XMEGA A1U devices
- The binary file in the SD card used for firmware update
- Configurable I/O start condition
- FatFs file access with SD card accessed by SPI

Table of Contents

1. Overview

This application note describes a simple implementation of SD Card bootloader on ATxmega128A1U. The firmware runs on an XMEGA A1U Xplained Pro, connected by an I/O1 Xplained Pro holding a microSD card. The SD card is used as the SPI Slave and ATxmega128A1U acts as SPI Master. The binary file (test.bin) to be programmed is stored in the SD card. In Boot mode, the boot loader reads contents of the file and flashes image to the application section of the MCU flash memory.

1.1. Hardware Interface

The I/O1 Xplained Pro board should be connected to the EXT1 connector of the XMEGA A1U Xplained Pro. ATxmega128A1U SPIC on EXT1 is used for SD card communication (see Table 1-1). In addition, PQ2 for user button SW0 is used as Boot mode detection pin, and LED0 controlled by PQ3 indicates the programming is ongoing (see Table 1-2)

Table 1-1. Functions for Used Pins on EXT1

Pin on EXT1	XMEGA A1U pin	Functions for used pins
1 ID	-	
2 GND	-	
3 ADC+	PA0	
4 ADC-	PA4	
5 GPIO1	PE6	
6 GPIO2	PE7	
7 PWM+	PE1	
8 PWM-	PE0	
9 IRQ/GPIO	PR0	
10 SPI_SS_B/GPIO	PR1	SD card presence detection, input pin, low level stands for presence
11 TWI_SDA	PC0	
12 TWI_SCL	PC1	
13 USART_RX	PC2	
14 USART_TX	PC3	
15 SPI_SS_A	PC4	SPIC SS, SPIC used for SD Card access
16 SPI_MOSI	PC5	SPIC MOSI, SPIC used for SD card access
17 SPI_MISO	PC6	SPIC MISO, SPIC used for SD card access
18 SPI_SCK	PC7	SPIC SCK, SPIC used for SD card access
19 GND	-	
20 VCC	-	

Table 1-2. Control and Status Pins

XMEGAA1U pin	Function	Description
PQ2	Input pin, connected to user button SW0	Detects this pin when starting from reset or power on. Low level: enter Boot mode High level: go on with other detections (flash content at starting address)
PQ3	Output pin, connected to LED0 and LED0 lights on with low level output on this pin	If SD card is detected in Boot mode, LED0 lights on until re-programming is finished

1.2. SD Card

A microSD card is used to store the binary file to be flashed. This binary file is required to be named by "test.bin" and placed at the root directory of the microSD card. It should be formatted with FAT file system (default) under Window OS and mounted onto the microSD card socket on the I/O1 Xplained Pro.

1.3. Firmware Availability

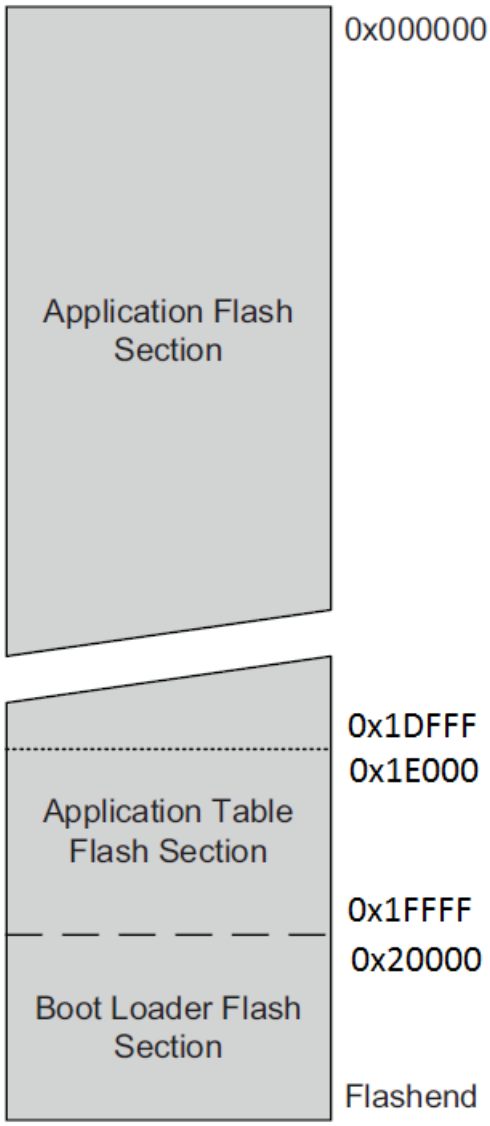
The firmware source code of this application note can be found from Atmel | START. Customization can be done with the source code by the users.

2. Bootloader Implementation

2.1. Self-Programing

ATxmega128A1U flash program memory includes an application section of 120KB, an application table section of 8KB, and a boot loader section of 8KB. The flash map is shown in the figure below. For this application note, the bootloader implementation targets to re-program the application flash section.

Figure 2-1. Flash Section Map

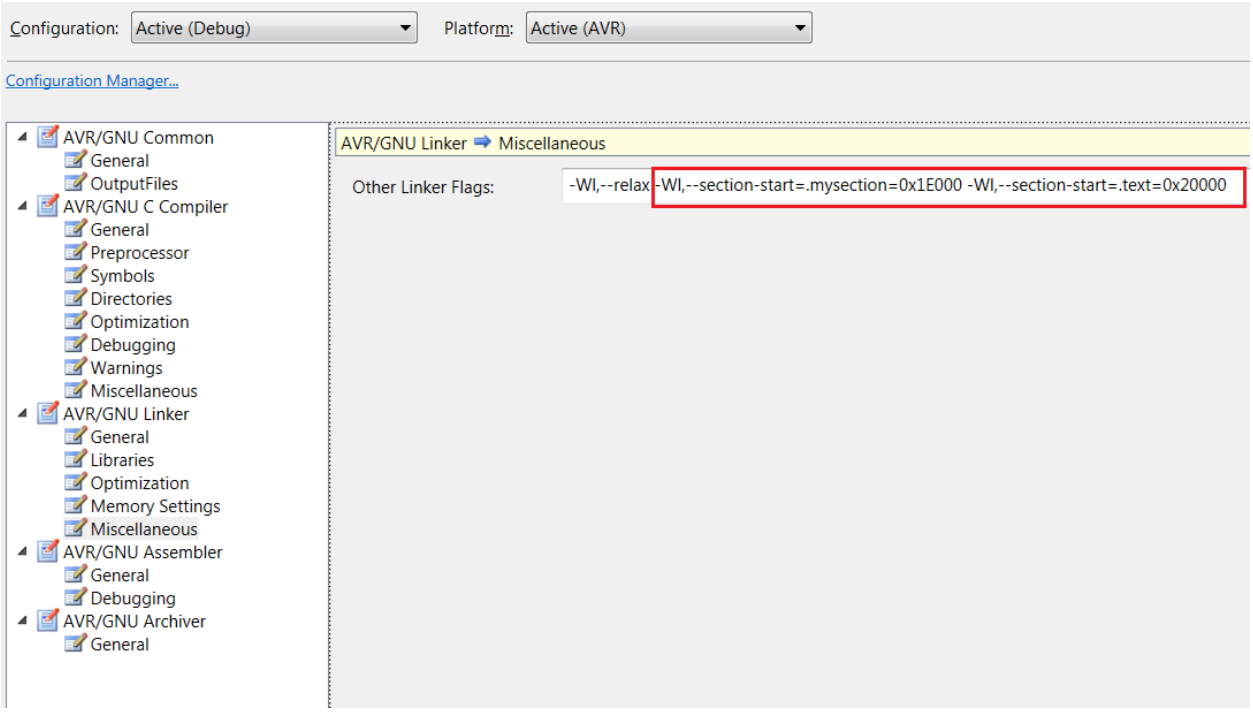


Reading and writing (including erasing) the flash memory from the application code inside the device is referred to as self-programming for flash memory. A bootloader is such an application for this purpose. For ATxmega128A1U, to perform flash erase and write, the SPM instruction must be called and this instruction can only be performed from the boot loader flash section.

For this bootloader implementation, the code size is approximately 12KB, and the code is placed in the boot loader section and the application table section. When erasing or writing a page located inside the application section, only the boot loader section can be read while the application table section cannot be read. So the code relating to reading binary file and memory program must be placed at the boot loader section. To enable the code placement in the two sections, the linker options must be modified for an Atmel Studio project. Add the following linker options in the project property window, as shown in the figure below.

Wl,--section-start=.mysection=0x1E000 -Wl,--section-start=.text=0x20000

Figure 2-2. Add Linker Options in Atmel Studio



For the code to be located at application table section, add compilation attribute directives on functions definition.

```
#if defined(__GNUC__)
void PMIC_SetVectorLocationToBoot( void )__attribute__((section(".mysection")));
#endif
```

2.2. FATFS Based File Access

FatFs R0.09 is used to access the binary file on the SD card. It has been ported to Atmel devices and is ready for usage as the third party module through Atmel Software Framework (ASF), which is released with Atmel Studio or individually. The needed source files for the FatFs module are included in the firmware pack.

In this bootloader implementation, only read operation is necessary while write operation is not. So to save memory space, the FatFs module should be configured to read-only and the table below lists some main settings.

Table 2-1. Main Settings of FATFS Module

Setting name	Value	Description
_FS_TINY	1	When _FS_TINY is set to 1, FatFs uses the sector buffer in the file system object instead of the sector buffer in the individual file object for file data transfer. This reduces memory consumption 512 bytes each file object.
_FS_READONLY	1	Only read operation is required for the binary file
_FS_MINIMIZE	3	The minimized level to remove functions not used in this implementation
_VOLUMES	1	Only 1 volume (logical drive) is used
_MAX_SS	512	Always set 512 for memory card
_USE_ERASE	0	Disable sector erase feature
_WORD_ACCESS	0	Byte-by-byte access

2.3. Bootloader Protocol

MCU executes from the bootloader section at each reset/power-on sequence. Initially it checks the status on a configurable BOOT_LOAD_PIN. The BOOT_LOAD_PIN needs to be configured as input with pull-up before checked.

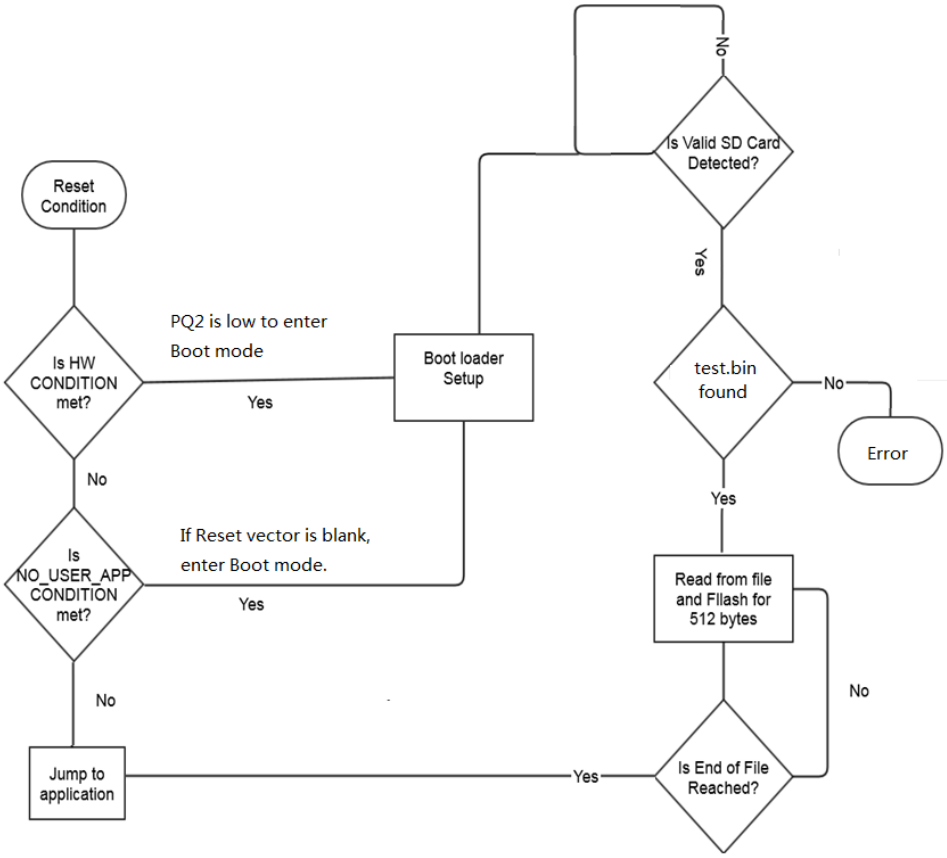
- If the pin is pulled low, enter boot mode
- If the pin is high, read the address 0x0000 of application section
  - If 0xFFFF is read out, the application section is empty and will enter boot mode
  - If not, jump to the application section and run the application code

Once entering boot mode, the following procedures are to be performed step by step.

- Initialize the on-board ports, system clock, SD MMC, FatFs, etc.
- Keep detecting SD card presence until a card is detected
- Turn on LED0
- Mount the SD card for file system access
- Open the file test.bin
- Read file for a block of MAX_BUF_SIZE (configurable)
- Program the data to application flash section starting from 0x0000
- Repeat read and program till the end of file
- Turn off LED0
- Remap the reset vector to the application section
- Jump to the starting address at 0x0000

The firmware chart is shown in the figure below.

Figure 2-3. Firmware Chart



3. Setup Procedure

This chapter describes the setup procedure from creating an example project based on firmware pack to running a bootloader demo.

In the following, Atmel Studio is taken as an example to describe the setup procedure.

1. Download the example firmware package for Atmel Studio from Atmel | START.
2. Create firmware project with the firmware package in Atmel Studio (7 or higher).
3. Set linker options and pre-defined symbles for the project in Atmel Studio.
   On the project property window, trace ToolChain--AVR/GNU Linker--Miscellaneous, and add the following line:

   -Wl,--relax -Wl,--section-start=.mysection=0x1E000 -Wl,--section-start=.text=0x20000

   Trace ToolChain--AVR/GNU C Compiler--Symbols, and add the following symbols:

   BOARD=XMEGA_A1U_XPLAINED_PRO
   IOPORT_XMEGA_COMPAT
   SD_MMC_ENABLE
4. Build the firmware and generate a bootloader image file (.elf, .hex, etc).
5. Connect the I/O1 Xplained Pro board to the XMEGA A1U Xplained Pro board at EXT1 connector, and connect the latter to the PC by the EDBG USB port on the board.
6. Program the fuse bit BOOTRST, which makes the XMEGA A1U run from the bootloader section.
7. Program bootloader image into the flash memory.
8. Store test.bin to the SD card, and mount the card on the I/O1 Xplained Pro board.
9. If SW0 is pressed after reset or no application firmware in application flash section, the boot mode is triggered. It will try to find an SD card and re-program the application section with the test.bin file in the SD card.

4. Revision History

Doc Rev.	Date	Comments
42788A	10/2016	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**