



# Compiling ATSAM21 Bootloader

Created by lady ada

```
Done burning bootloader.
target halted due to debug-request, current mode: Thread
xPSR: 0x81000000 pc: 0x00000158 msp: 0x20008000
** Programming Started **
auto erase enabled
wrote 16384 bytes from file C:\Users\ladyada\AppData\Local\Arduino15\packages\arduino\hardware\samd\1.6.2/boot
** Programming Finished **
** Verify Started **
verified 6328 bytes in 0.543031s (11.380 KiB/s)
** Verified OK **
** Resetting Target **
shutdown command invoked
```

Arduino/Genuino Zero (Programming Port) on COM19

<https://learn.adafruit.com/compiling-m0-atsamd21-bootloader>

Last updated on 2024-06-03 01:53:04 PM EDT

# Table of Contents

## Compile

---

3

- [Download Latest Bootloader Code](#)
- [Programming into an Arduino Zero w/EDBG](#)
- [Feather M0 or Others](#)
- [Atmel Studio](#)

---

# Compile

Note that this tutorial is for 'unlocked' fresh ATSAM chips. If you have a pre-programmed Feather M0 or Zero you may need to unlock the bootloader bits, we don't have a tutorial on doing that

**WE DO NOT PROVIDE ANY SUPPORT FOR THIS TUTORIAL OR, IN GENERAL, HOW TO PROGRAM BOOTLOADERS OR ERASE CHIPS.**

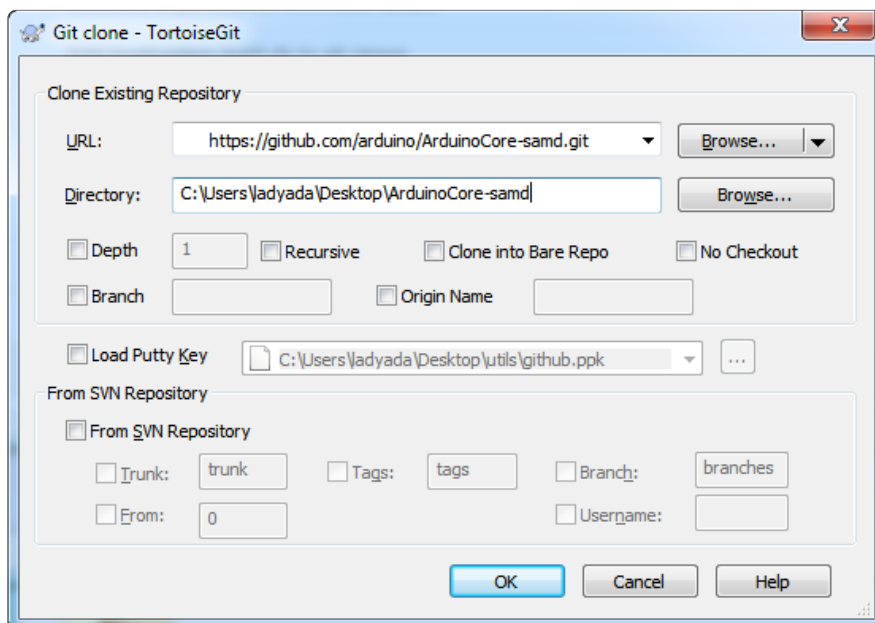
This isn't an in-depth tutorial, just some notes I took since I was asked a few times how to modify/compile the bossac bootloader!

## Download Latest Bootloader Code

The latest Arduino SAMD core is at <https://github.com/arduino/ArduinoCore-samd> (<https://adafru.it/mdi>)

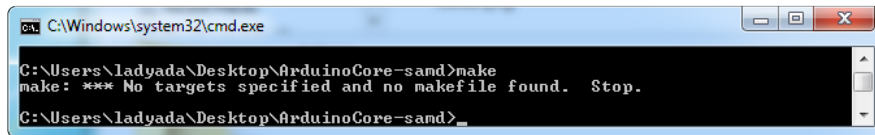
You can [download a zip of the current snapshot](https://adafru.it/mdj) (<https://adafru.it/mdj>) or you can use git to clone the repository

Its better to do this than compile in the board-support-package location (e.g. C:\user\blah\local\arduino15) because you'll lose your changes if you update the BSP!



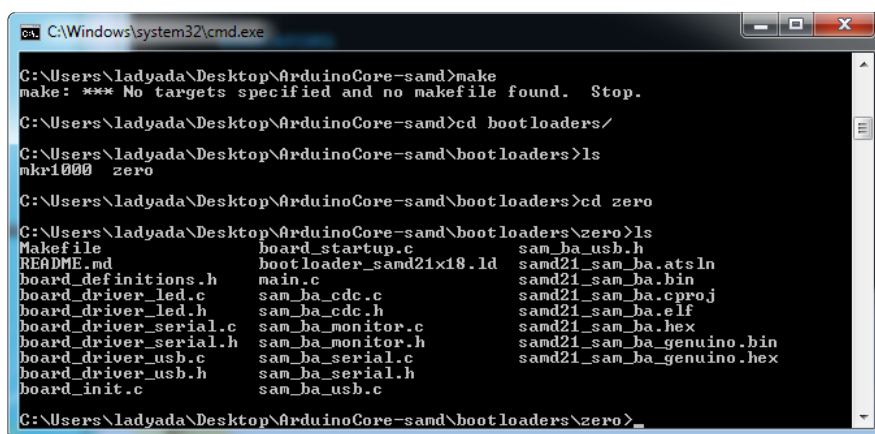
You'll need to have some basic command line compilation tools already installed like **make** - such as MSys or Cygwin ([which we discuss here \(https://adafru.it/mdk\)](https://adafru.it/mdk)) or install [make as a binary \(https://adafru.it/mdl\)](https://adafru.it/mdl)

Just make sure when you run **make** in a command line you get that it is looking for a makefile



```
C:\Windows\system32\cmd.exe
C:\Users\ladyada\Desktop\ArduinoCore-sand>make
make: *** No targets specified and no makefile found. Stop.
C:\Users\ladyada\Desktop\ArduinoCore-sand>_
```

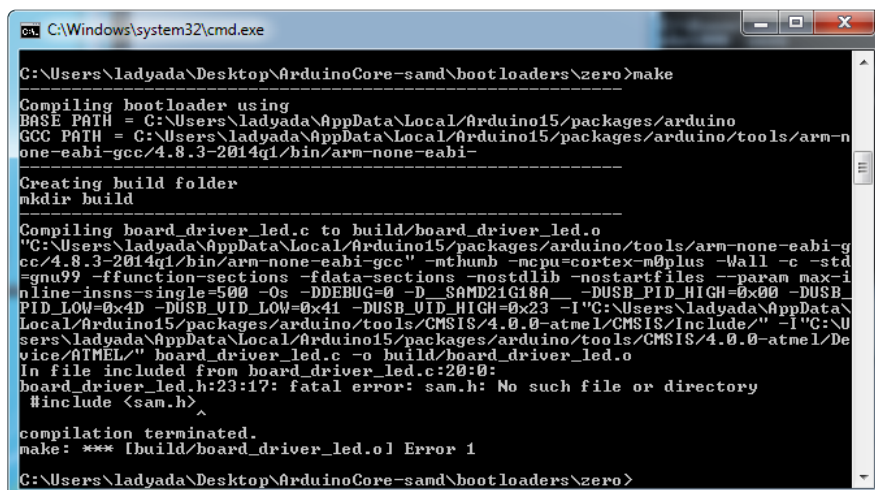
Change into the **bootloaders/zero** directory



```
C:\Windows\system32\cmd.exe
C:\Users\ladyada\Desktop\ArduinoCore-sand>make
make: *** No targets specified and no makefile found. Stop.
C:\Users\ladyada\Desktop\ArduinoCore-sand>cd bootloaders/
C:\Users\ladyada\Desktop\ArduinoCore-sand\bootloaders>ls
mkr1000 zero
C:\Users\ladyada\Desktop\ArduinoCore-sand\bootloaders>cd zero
C:\Users\ladyada\Desktop\ArduinoCore-sand\bootloaders\zero>ls
Makefile          board_startup.c      sam_ba_usb.h
README.md         boot_loader_samd21x18.ld  samd21_sam_ba_atmel
board_definitions.h  main.c              samd21_sam_ba_bin
board_driver_led.c  sam_ba_cdc.c        samd21_sam_ba_cproj
board_driver_led.h  sam_ba_cdc.h        samd21_sam_ba_elf
board_driver_serial.c  sam_ba_monitor.c   samd21_sam_ba_hex
board_driver_serial.h  sam_ba_monitor.h   samd21_sam_ba_genuino_bin
board_driver_usb.c   sam_ba_serial.c     samd21_sam_ba_genuino_hex
board_driver_usb.h   sam_ba_serial.h
board_init.c        sam_ba_usb.c
```

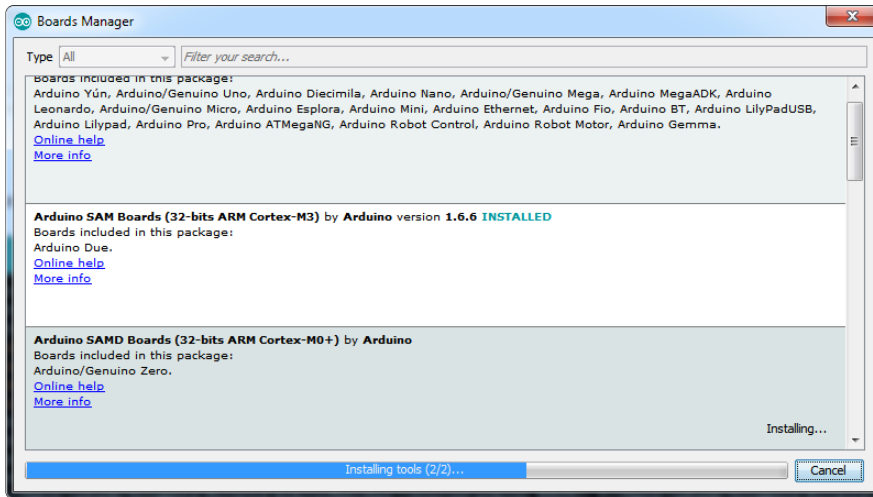
If you try to **make** all here and you get

**fatal error: sam.h: No such file or directory**

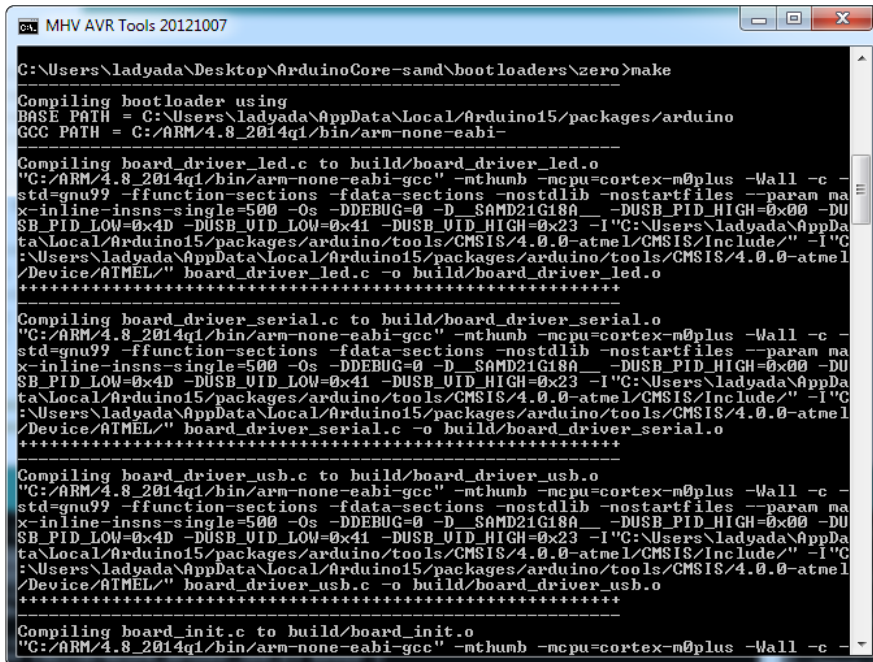


```
C:\Windows\system32\cmd.exe
C:\Users\ladyada\Desktop\ArduinoCore-sand\bootloaders\zero>make
Compiling bootloader using
BASE_PATH = C:\Users\ladyada\AppData\Local\Arduino15\packages\arduino
GCC_PATH = C:\Users\ladyada\AppData\Local\Arduino15\packages\arduino\tools/arm-none-eabi-gcc/4.8.3-2014q1/bin/arm-none-eabi-
Creating build folder
mkdir build
Compiling board_driver_led.c to build/board_driver_led.o
"C:\Users\ladyada\AppData\Local\Arduino15\packages\arduino\tools/arm-none-eabi-gcc/4.8.3-2014q1/bin/arm-none-eabi-gcc" -mthumb -mcpu=cortex-m0plus -Wall -c -std=gnu99 -ffunction-sections -fdata-sections -nostdlib -nostartfiles --param max-inline-insns-single=500 -Os -DDEBUG=0 -D_SAMD21G18A -DUSB_PID_HIGH=0x00 -DUSB_PID_LOW=0x4D -DUSB_UID_LOW=0x41 -DUSB_UID_HIGH=0x23 -I"C:\Users\ladyada\AppData\Local\Arduino15\packages\arduino\tools\CMSIS/4.0.0-atmel\CMSIS/Include/" -I"C:\Users\ladyada\AppData\Local\Arduino15\packages\arduino\tools\CMSIS/4.0.0-atmel/Device/ATMEL/" board_driver_led.c -o build/board_driver_led.o
In file included from board_driver_led.c:20:0:
board_driver_led.h:23:17: fatal error: sam.h: No such file or directory
#include <sam.h>
^
compilation terminated.
make: *** [build/board_driver_led.o] Error 1
C:\Users\ladyada\Desktop\ArduinoCore-sand\bootloaders\zero>
```

Make sure you've installed the Arduino SAM and SAMD packages through the board manager



Try make all again for success!



```

MHV AVR Tools 20121007

Compiling sam_ba_serial.c to build/sam_ba_serial.o
"C:/ARM/4.8_2014q1/bin/arm-none-eabi-gcc" -mthumb -mcpu=cortex-m0plus -Wall -c -std=gnu99 -ffunction-sections -fdata-sections -nostdlib -nostartfiles --param max-inline-insns-single=500 -Os -DDEBUG=0 -D_SAMD21G18A -DUSB_PID_HIGH=0x00 -DUSB_PID_LOW=0x4D -DUSB_VID_LOW=0x41 -DUSB_VID_HIGH=0x23 -I"C:\Users\ladyada\AppData\Local\Arduino15\packages\arduino\tools\CMSIS/4.0.0-atmel\CMSIS/Include/" -I"C:\Users\ladyada\AppData\Local\Arduino15\packages\arduino\tools\CMSIS/4.0.0-atmel/Device/ATMEL/" sam_ba_serial.c -o build/sam_ba_serial.o
*****

Creating ELF binary
"C:/ARM/4.8_2014q1/bin/arm-none-eabi-gcc" -L. -Lbuild -mthumb -mcpu=cortex-m0plus -Wall -Wl,--cref -Wl,--check-sections -Wl,--gc-sections -Wl,--unresolved-symbols=report-all -Wl,--warn-common -Wl,--warn-section-align -Wl,--warn-unresolved-symbols --specs=mano.specs --specs=nosys.specs -Os -Wl,--gc-sections --save-temps -Tbootloader_samd21x18.ld -Wl,-Map,"build/samd21_sam_ba.map" -o "build/samd21_sam_ba.elf" -Wl,--start-group build/board_driver_led.o build/board_driver_serial.o build/board_driver_usb.o build/board_init.o build/board_startup.o build/main.o build/sam_ba_usb.o build/sam_ba_cdc.o build/sam_ba_monitor.o build/sam_ba_serial.o -lm -Wl,--end-group
"C:/ARM/4.8_2014q1/bin/arm-none-eabi-nm" "build/samd21_sam_ba.elf" >"build/samd21_sam_ba_symbols.txt"
"C:/ARM/4.8_2014q1/bin/arm-none-eabi-size" --format=sysv -t -x build/samd21_sam_ba.elf
build/samd21_sam_ba.elf :
section      size          addr
.vectors     0x40          0x0
.text        0x16bc       0x40
.data        0x58         0x20000000
.bss        0x340       0x20000058
.ARM.attributes 0x28         0x0
.comment     0x70         0x0
.debug_frame 0x48         0x0
Total       0x1b74

C:\Users\ladyada\Desktop\ArduinoCore-samd\bootloaders\zero>

```

Before distributing, make sure to change the VID/PID in the makefile:

```

X Makefile - XEmacs
File Edit View Cmds Tools Options Buffers Makefile Help
Open Dired Save Print Cut Copy Paste Undo Spell Replace Mail Info Compile Debug News
Makefile
# -----
# Compiler options
CFLAGS=-mthumb -mcpu=cortex-m0plus -Wall -c -std=gnu99 -ffunction-sections -fdata-sections -nostdlib -nostartfiles --param max-inline-insns-single=500
ifdef DEBUG
CFLAGS+=-g3 -O1 -DDEBUG=1
else
CFLAGS+=-Os -DDEBUG=0
endif

# My M0
CFLAGS_EXTRA?=-D_SAMD21G18A -DUSB_PID_HIGH=0x00 -DUSB_PID_LOW=0x0B -DUSB_VID_LOW=0x92 -DUSB_VID_HIGH=0x23

# Arduino Zero (PID == 0x004D)
#CFLAGS_EXTRA?=-D_SAMD21G18A -DUSB_PID_HIGH=0x00 -DUSB_PID_LOW=0x4D -DUSB_VID_LOW=0x41 -DUSB_VID_HIGH=0x23
# Genuino Zero (PID == 0x024D)
# CFLAGS_EXTRA?=-D_SAMD21G18A -DUSB_PID_HIGH=0x02 -DUSB_PID_LOW=0x4D -DUSB_VID_LOW=0x41 -DUSB_VID_HIGH=0x23
# Arduino MKR1000 (PID == 0x004E)
# CFLAGS_EXTRA?=-D_SAMD21G18A -DUSB_PID_HIGH=0x00 -DUSB_PID_LOW=0x4E -DUSB_VID_LOW=0x41 -DUSB_VID_HIGH=0x23
# Genuino MKR1000 (PID == 0x024E)
# CFLAGS_EXTRA?=-D_SAMD21G18A -DUSB_PID_HIGH=0x02 -DUSB_PID_LOW=0x4E -DUSB_VID_LOW=0x41 -DUSB_VID_HIGH=0x23

INCLUDES=-I"${MODULE_PATH}/tools\CMSIS/4.0.0-atmel\CMSIS/Include/" -I"${MODULE_PATH}/tools\CMSIS/4.0.0-atmel/Device/ATMEL/"

# -----
# Linker options
LDFLAGS=-mthumb -mcpu=cortex-m0plus -Wall -Wl,--cref -Wl,--check-sections -Wl,--unresolved-symbols=report-all -Wl,--warn-common -Wl,--warn-section-align -Wl,--warn-unresolved-symbols --specs=mano.specs --specs=nosys.specs -Os -Wl,--gc-sections --save-temps -Tbootloader_samd21x18.ld -Wl,-Map,"build/samd21_sam_ba.map" -o "build/samd21_sam_ba.elf" -Wl,--start-group build/board_driver_led.o build/board_driver_serial.o build/board_driver_usb.o build/board_init.o build/board_startup.o build/main.o build/sam_ba_usb.o build/sam_ba_cdc.o build/sam_ba_monitor.o build/sam_ba_serial.o -lm -Wl,--end-group

```

then run `make clean` and `make all` again to make fresh files

```

C:\Users\ladyada\Desktop\ArduinoCore-samd\bootloaders\zero>
cc/4.8.3-2014q1/bin/arm-none-eabi-gcc" -L. -Lbuild -mthumb -mcpu=cortex-m0plus -
Wall -Wl,--cref -Wl,--check-sections -Wl,--gc-sections -Wl,--unresolved-symbols=
report-all -Wl,--warn-common -Wl,--warn-section-align -Wl,--warn-unresolved-symb
ols --specs=nano.specs --specs=nosys.specs -Os -Wl,--gc-sections -save-temps -Tb
ootloader_samd21x18.ld -Wl,-Map,"build/samd21_sam_ba.map" -o "build/samd21_sam_b
a.elf" -Wl,--start-group build/board_driver_led.o build/board_driver_serial.o bu
ild/board_driver_usb.o build/board_init.o build/board_startup.o build/main.o bu
ild/sam_ba_usb.o build/sam_ba_cdc.o build/sam_ba_monitor.o build/sam_ba_serial.o
-lm -Wl,--end-group
"C:\Users\ladyada\AppData\Local\Arduino15\packages\arduino\tools/arm-none-eabi-g
cc/4.8.3-2014q1/bin/arm-none-eabi-nm" "build/samd21_sam_ba.elf" >"build/samd21_s
am_ba_symbols.txt"
"C:\Users\ladyada\AppData\Local\Arduino15\packages\arduino\tools/arm-none-eabi-g
cc/4.8.3-2014q1/bin/arm-none-eabi-size" --format=sysv -t -x build/samd21_sam_ba
.elf
build/samd21_sam_ba.elf :
section      size      addr
.vectors     0x40      0x0
.text        0x16bc   0x40
.data        0x58     0x20000000
.bss         0x340    0x20000058
.ARM.attributes 0x28     0x0
.comment     0x70     0x0
.debug_frame 0x48     0x0
Total       0x1b74

-----
Creating flash binary
"C:\Users\ladyada\AppData\Local\Arduino15\packages\arduino\tools/arm-none-eabi-g
cc/4.8.3-2014q1/bin/arm-none-eabi-objcopy" -O binary build/samd21_sam_ba.elf sam
d21_sam_ba.bin
-----
Creating flash binary
"C:\Users\ladyada\AppData\Local\Arduino15\packages\arduino\tools/arm-none-eabi-g
cc/4.8.3-2014q1/bin/arm-none-eabi-objcopy" -O ihex build/samd21_sam_ba.elf samd2
1_sam_ba.hex
C:\Users\ladyada\Desktop\ArduinoCore-samd\bootloaders\zero>

```

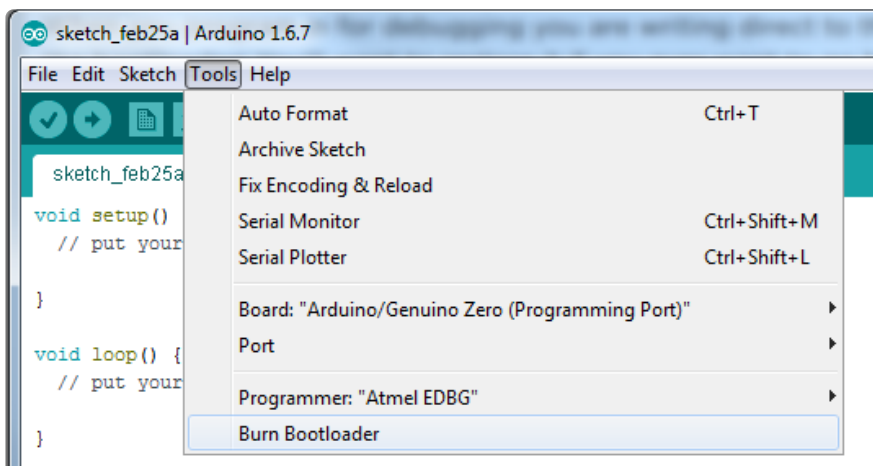
When you're done, you'll have two new files, `samd21_sam_ba.hex` and `samd21_sam_ba.bin`

## Programming into an Arduino Zero w/EDBG

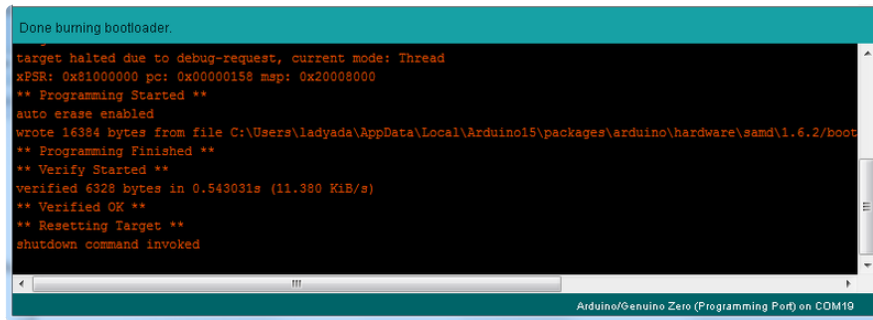
The easiest way to do this is just replace `samd21_sam_ba.bin` in `C:\Users\ladyada\AppData\Local\Arduino15\packages\arduino\hardware\samd\1.6.4\bootloaders\zero` (or where-ever the current version of your Arduino SAMD package is with the compiled `samd21_sam_ba.bin`

Launch the IDE, select **Arduino Zero (programming port)** from the **Tools->Board** menu, and **Atmel EDBG** as the **Tools->Programmer**

Then select **Burn Bootloader**



It only takes a few seconds to burn in the bootloader:



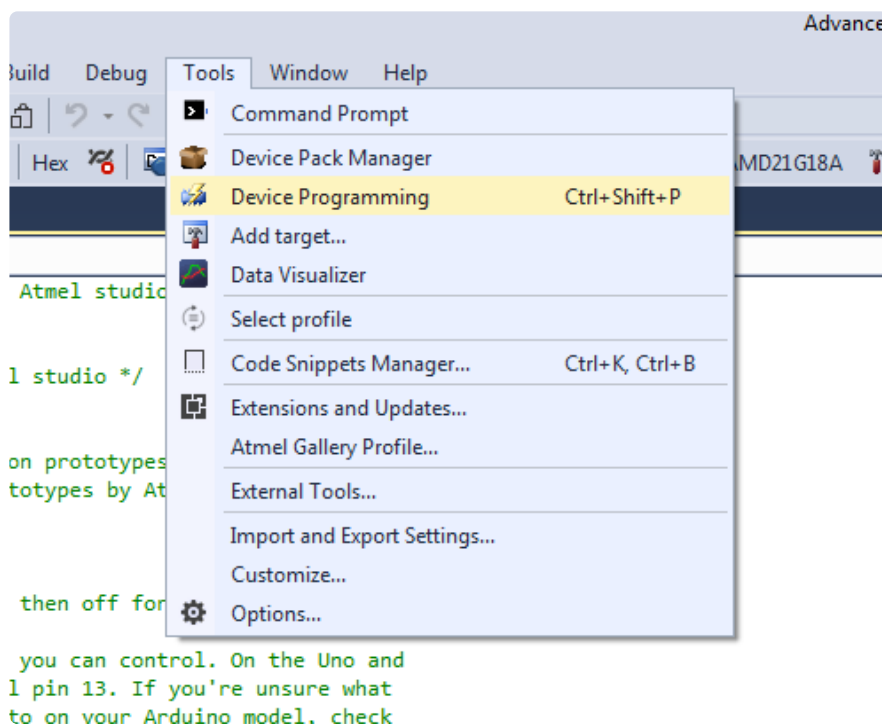
## Feather M0 or Others

For this, you'll need to use an **Atmel Studio** or **adalink** setup, since you're using a J-Link or stlink.

### Atmel Studio

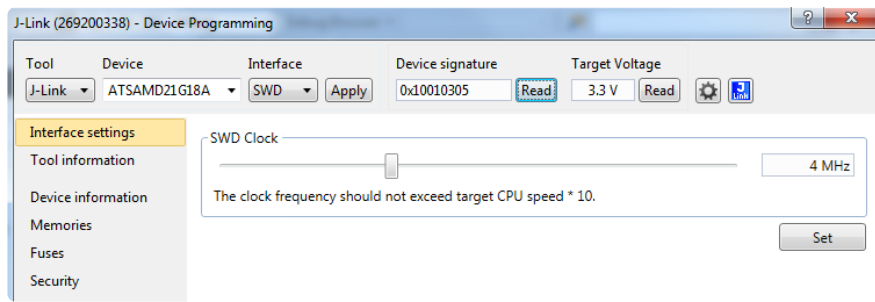
If you're using Atmel Studio, [install it \(https://adafru.it/mdm\)](https://adafru.it/mdm)

Plug in your Jlink and connect it to the SWD pins. OK now you have your debugger plugged in, its good to check that it works, select **Device Programming**



Under **Tool** make sure you can select **J-Link**



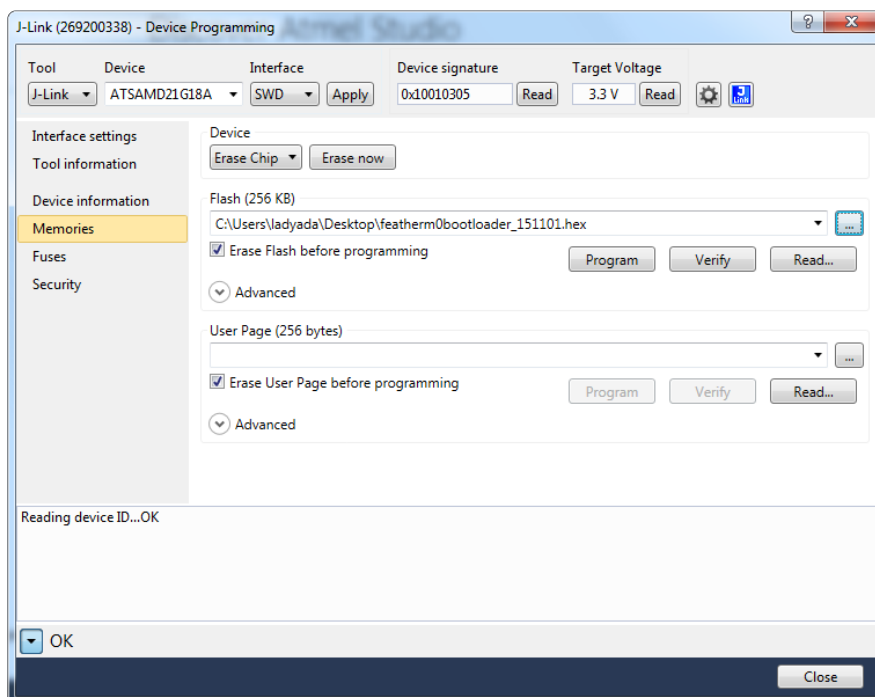


Select **ATSAMD21G18A** as the device, **SWD** as the interface and hit **Apply**

You can then **Read** the Device Signature. Make sure this all works before you continue!

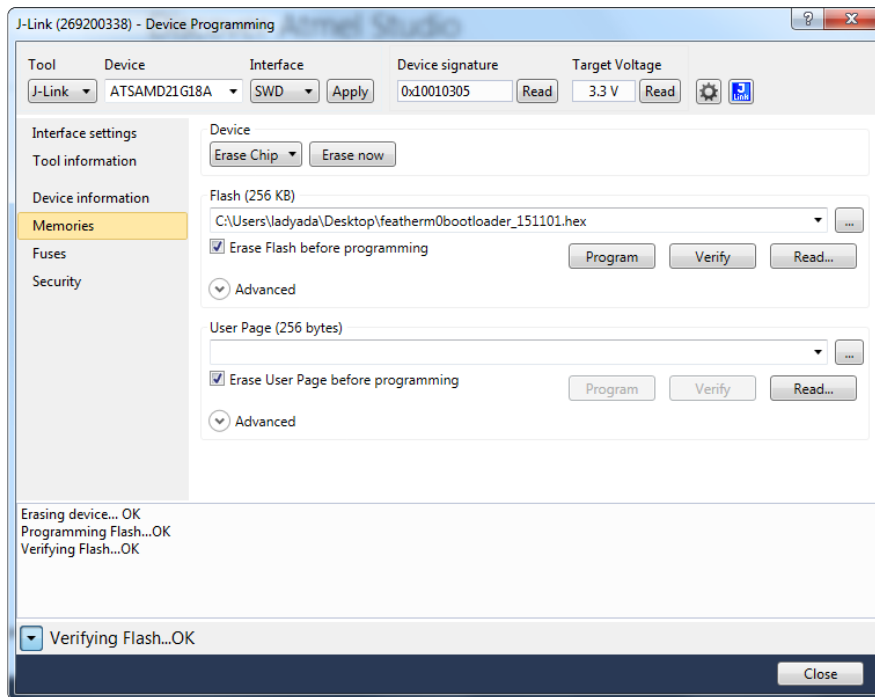
If you are asked to update the J-Link firmware, its OK to do so now.

Next click on **Memories** in the left hand side



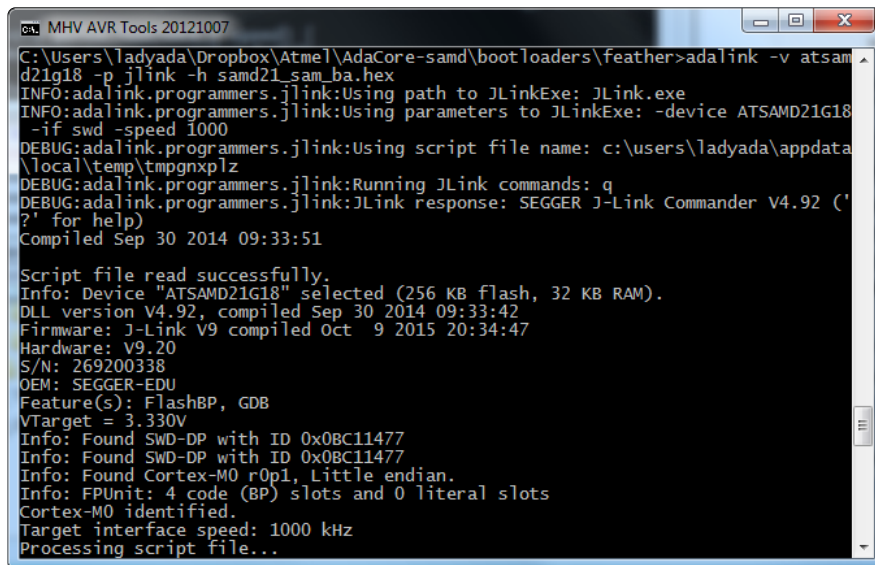
Next to the **Flash (256 KB)** section, click the triple-dots and select the **hex** bootloader file generated from the previous **make all**

Then click **Program** to program it in



If you don't have Windows [you can install adalink](https://adafru.it/fPq) (https://adafru.it/fPq)

Then run `adalink -v atsamd21g181 -p jlink -h samd21_sam_ba.hex`



```

MHV AVR Tools 20121007
Info: FPUnit: 4 code (BP) slots and 0 literal slots
Cortex-M0 identified.
Target interface speed: 1000 kHz
Processing script file...

Reset delay: 0 ms
Reset type NORMAL: Resets core & peripherals via SYSRESETREQ & VECTRESET bit.

Downloading file [C:\Users\ladyada\Dropbox\Atmel\AdaCore-samd\bootloaders\feather\samd21_sam_ba.hex]...
WARNING: CPU is running at very low speed (1329 kHz). Readback will be performed instead of CRC calculation.
Info: J-Link: Flash download: Flash download into internal flash skipped. Flash contents already match
Info: J-Link: Flash download: Total time needed: 0.368s (Prepare: 0.118s, Compare: 0.238s, Erase: 0.000s, Program: 0.000s, Verify: 0.000s, Restore: 0.010s)
O.K.

Reset delay: 0 ms
Reset type NORMAL: Resets core & peripherals via SYSRESETREQ & VECTRESET bit.

Script processing completed.

C:\Users\ladyada\Dropbox\Atmel\AdaCore-samd\bootloaders\feather>

```

You can also check out <https://learn.adafruit.com/programming-microcontrollers-using-openocd-on-raspberry-pi> (<https://adafru.it/mdn>) if you want hints on using OpenOCD directly!