

---

## Using SAM-BA on the SAM4L-EK

---

AN-4574

### Prerequisites

---

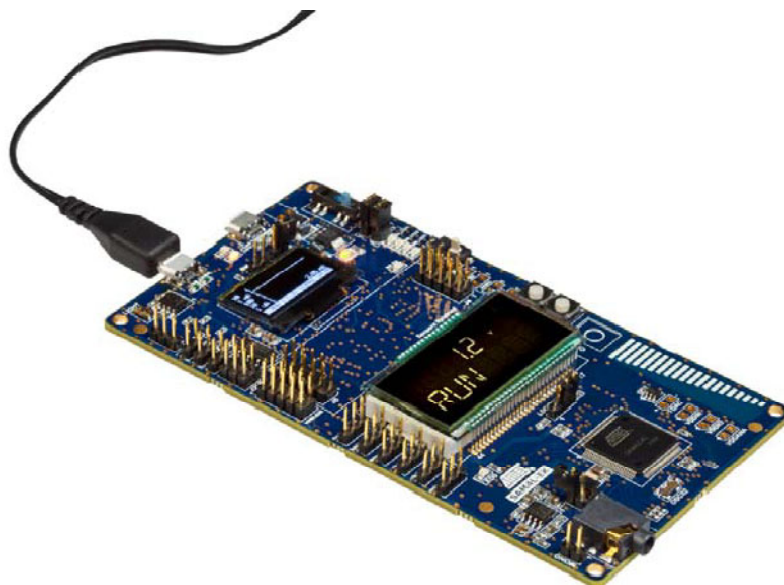
- **Hardware Prerequisites**
  - Atmel® SAM4L-EK Evaluation Kit
- **Software Prerequisites**
  - Atmel Studio 6.1 update 2.0 (build 2730) or higher
  - Atmel Software Frameworks 3.11.0 or higher
  - Atmel SAM-BA® 2.12
  - Atmel SAM-BA 2.12 patch 5
  - Free Hexa Editor 1.6.0 or later (Frhed)
- **Estimated Completion Time:** 45 min

### Introduction

---

The goal of this Hands-On is to:

- Understand how to deploy SAM-BA boot on the SAM4L
- Understand the differences in the SAM-BA boot process with other SAM3/SAM4 devices
- Modify an existing application in order to make it co-exist with SAM-BA boot in internal flash



## Table of Contents

---

Prerequisites .....	1
Introduction .....	1
Icon Key Identifiers .....	3
1. Training Module Architecture .....	4
1.1 Atmel Studio Extension Delivery Case (.vsix).....	4
1.2 Atmel Training Executable Delivery Case (.exe) .....	4
2. Prerequisites .....	5
2.1 Atmel SAM-BA In-system Programming Software .....	5
2.2 Free Hexa Editor (Fhed) .....	5
3. Introduction .....	6
4. Assignment 1: Program SAM-BA Monitor in SAM4L Flash using Atmel Studio.....	9
5. Assignment 2: Program the Application in SAM4L Flash using SAM-BA .....	13
6. Assignment 3: Update Application Linker File.....	19
7. Assignment 4: Update SAM4L Flash User Page to Allow Boot Selection between SAM-BA Monitor and the User Application	21
8. Conclusion .....	27
9. Revision History .....	28

## Icon Key Identifiers

---



### INFO

Delivers contextual information about a specific topic



### TIPS

Highlights useful tips and techniques



### TO DO

Highlights objectives to be completed



### RESULT

Highlights the expected result of an assignment step



### WARNING

Indicates important information



### EXECUTE

Highlights actions to be executed out of the target when necessary

## 1. Training Module Architecture

This training material can be retrieved through different Atmel deliveries:

- As an Atmel Studio Extension (.vsix file) usually found on the Atmel Gallery web site (<http://gallery.atmel.com/>) or using the Atmel Studio Extension manager
- As an Atmel Training Executable (.exe file) usually provided during Atmel Training sessions

Depending on the delivery type, the different resources needed by this training material (hands-on documentation, datasheets, application notes, software & tools) will be found on different locations.

### 1.1 Atmel Studio Extension Delivery Case (.vsix)

Once the extension installed, you can open and create the different projects using “*New Example Project from ASF...*” in Atmel Studio.



#### INFO

The projects installed from an extension are usually under “*Atmel Training > Atmel Corp. Extension Name*”.

There are different projects which can be available depending on the extension:

- **Hands-on Documentation:** contains the documentation as required resources
- **Hands-on Assignment:** contains the initial project that may be required to start
- **Hands-on Solution:** contains the final application which is a solution for this hands-on



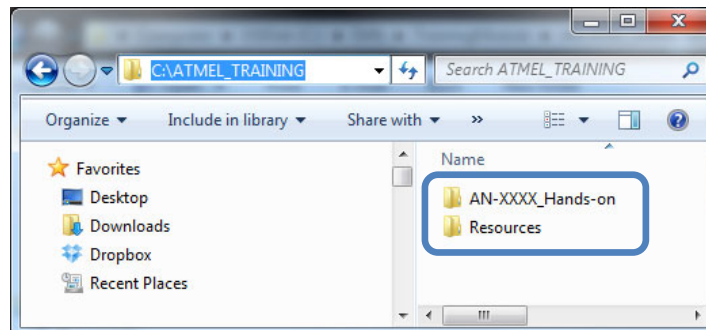
#### INFO

Each time a reference is made to some resources in the following pages, the user must refer to the **Hands-on Documentation** project folder.

### 1.2 Atmel Training Executable Delivery Case (.exe)

Depending where the executable has been installed, you will find the following architecture which is composed by two main folders:

- **AN-XXXX\_Hands-on:** contains the initial project that may be required to start and a solution
- **Resources:** contains required resources (datasheets, software & tools...)



#### INFO

Unless a specific location is specified, each time a reference is made to some resources in the following pages, the user must refer to this **Resources** folder.

## 2. Prerequisites



### TIPS

If you got an Atmel Training Executable (.exe), you will also find all the following Software & Tools in the [Resources\Software](#) folder.

### 2.1 Atmel SAM-BA In-system Programming Software

The SAM-BA GUI tool provides a means of easily programming various Atmel ARM® processor-based microcontrollers.

Key features of the SAM-BA UI are:

- Ability to perform in-system programming through JTAG, RS232 or USB interfaces
- Ability to be used via a Graphical User Interface or command lines
- Dump of memory content
- User scripts executable from SAM-BA GUI or a shell

SAM-BA and its patches can be preinstalled from Atmel web site at the following address:

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



### TO DO

Download and Install SAM-BA GUI 2.12 and SAM-BA 2.12 Patch 5 or higher.

- Go to <http://www.atmel.com/tools/ATMELSAM-BAIN-SYSTEMPROGRAMMER.aspx> and download the latest SAM-BA version including patch(es) if available
- Execute the [sam-ba\\_2.12.exe](#) file then [sam-ba\\_2.12\\_patch5.exe](#)



### RESULT

SAM-BA GUI is now correctly installed and patched on your PC.

### 2.2 Free Hexa Editor (Frehed)

Frehed is a free binary file editor for Windows®. It will be used to edit the binary files loaded in NAND Flash memory later in this hands-on module.

Frehed can be preinstalled from the official website at the following address:

<http://frhed.sourceforge.net/en/>



### TO DO

Install Frhed Hex Editor 1.6.0 or higher.

- Go to <http://frhed.sourceforge.net/en/> and download the latest Software version
- Execute the [Frehed-1.6.0-Setup.exe](#) file and follow the instructions



### RESULT

Frehed is now correctly installed on your PC.

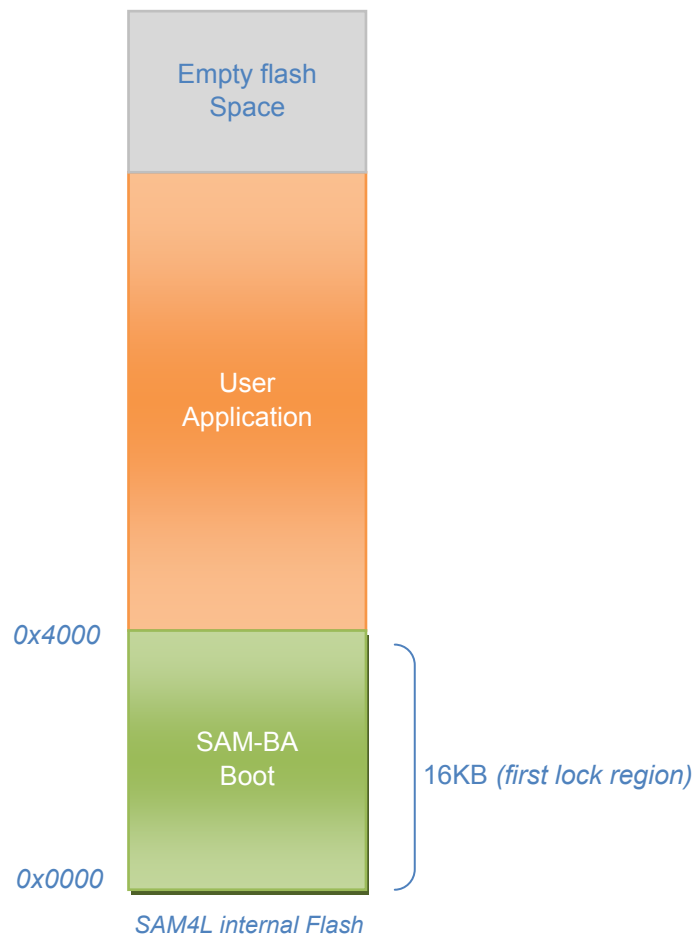
### 3. Introduction

SAM4L devices are factory programmed with the Atmel SAM Boot Assistant in the first lock region of flash memory. This bootloader relies on a monitor (SAM-BA monitor) and allows In-System Programming (ISP) from USB or serial port (UART) host without any external programming interface.

This SAM-BA Boot version is compatible with existing SAM-BA GUI software tools but has some differences compared to other SAM devices. The goal of this hands-on is to understand these differences.

Unlike other SAM3/SAM4 products using a ROM monitor, the ATSAM4L SAM-BA is stored in flash memory at address range [0x0000 to 0x4000]. Since SAM-BA is stored in flash memory, if the application requires the entire flash space and does not need the SAM-BA, it can be erased using a JTAG/SWD debugger.

To use SAM-BA together with an application the user needs to link the application starting at 0x4000:



By default, the flash region dedicated for SAM-BA (0x0000 to 0x4000) is locked by the General-purpose Fuse bit lock 0 on every device shipped.

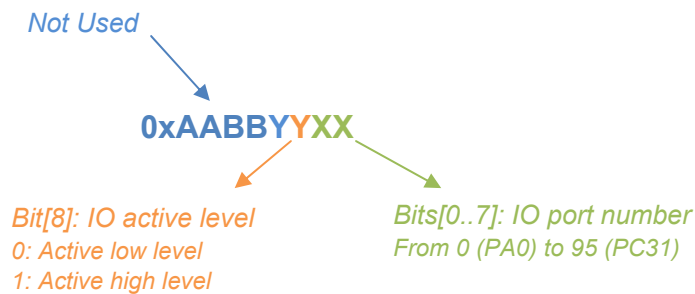
When loading the binary image into the device, only the upper part of flash starting at 0x4000 address should be programmed. Any attempt to write to the SAM-BA Boot region using SAM-BA GUI commands will be aborted and will throw an error.

When SAM-BA co-exists with the application code in internal flash, the behavior of the device at reset (boot from SAM-BA, or boot from the application) can be modified through the Flash user page, which contains the monitor configuration word.

This word, located at offset 0x10 in the user page, allows defining three different behaviors:

1. *Launch SAM-BA on reset*: If the monitor configuration word value is 0x4D4153 (“SAM” in ASCII), SAM-BA is systematically executed on reset. Application code is never executed.
2. *Force pin selection*: Selection between SAM-BA monitor and application code execution is done depending on the state of user-defined I/O. This I/O and its active level are defined in the configuration word. On reset and after a debounce period of 100ms, SAM-BA samples the I/O: if the I/O is in its active level SAM-BA is executed. If not, SAM-BA directly jumps to the application code then executes it.

In that case, the format of the configuration word is:

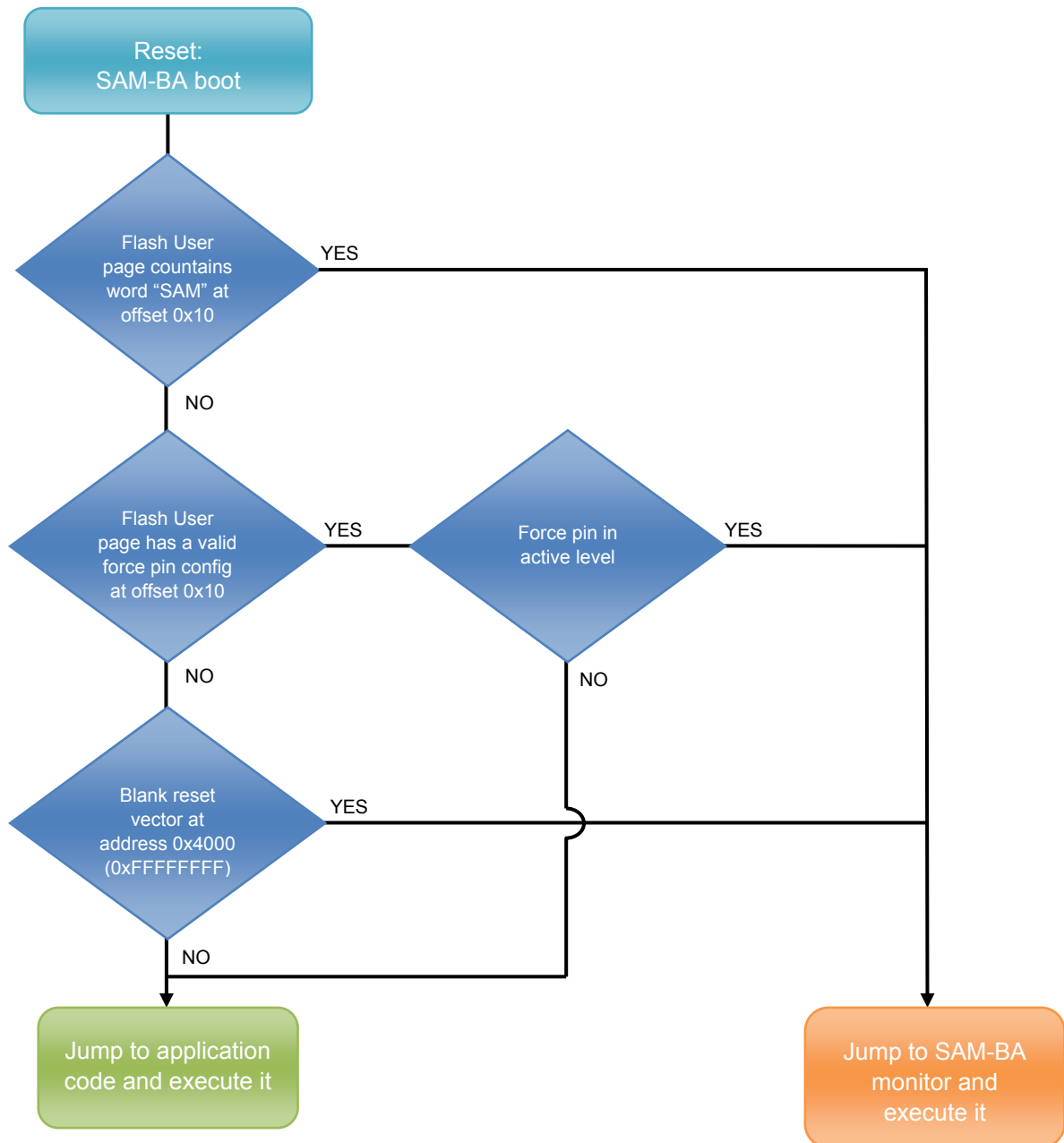


3. Any other configuration word value (such as 0xFFFFFFFF) is detected as invalid by SAM-BA and the application code will be systematically executed. (SAM-BA monitor is never executed).



**WARNING** Note that the flash user page also contains critical systems configuration settings such as BODs and watchdog settings. To keep the user page settings the user should first save the content of the flash user page, modify the content then write it back to the device.

Here is a flowchart that summarise the boot sequence when SAM-BA boot and user application co exist on the chip.





#### 4. Assignment 1: Program SAM-BA Monitor in SAM4L Flash using Atmel Studio

In this assignment you will have to load SAM-BA in internal flash and then lock the corresponding flash region in order to avoid any accidental overwriting.

There are two different programming methods: one with Atmel Studio 6.1, and another one using a programming script that uses J-Link DLL for non-user of Atmel Studio.

In this section we are going to experiment the method with Studio 6.1.

You can find more information on the second method in the application note called SAM-BA for SAM4L.

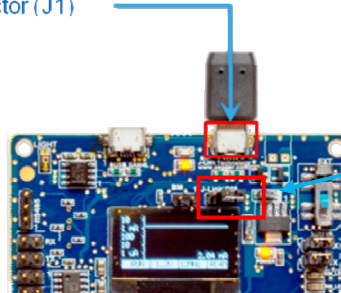


##### TO DO

Erase the flash content.

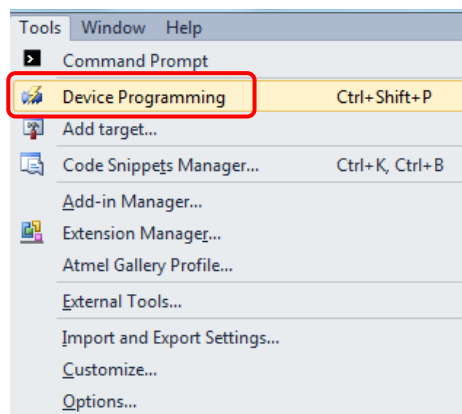
- Verify that SAM4L-EK is powered by the J-Link OB USB connector (J1) and that J-Link DIS jumper is not connected

J-Link OB USB connector (J1)

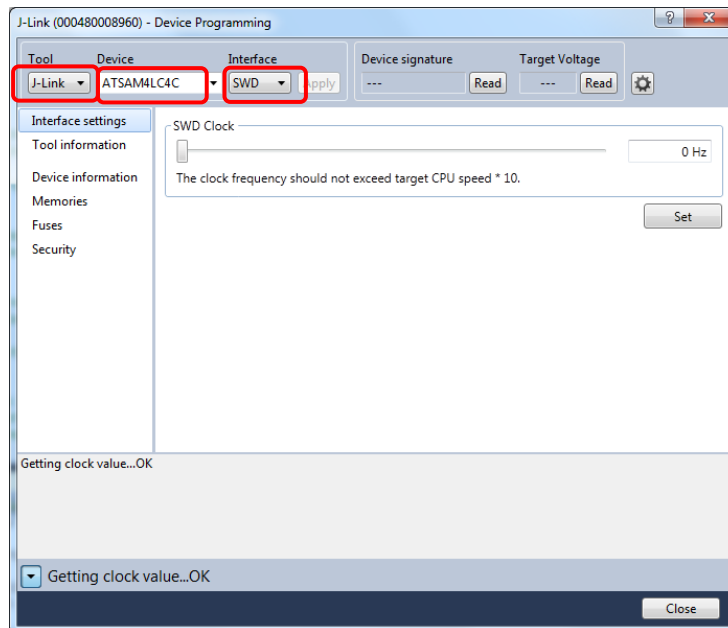


J-Link DIS jumper

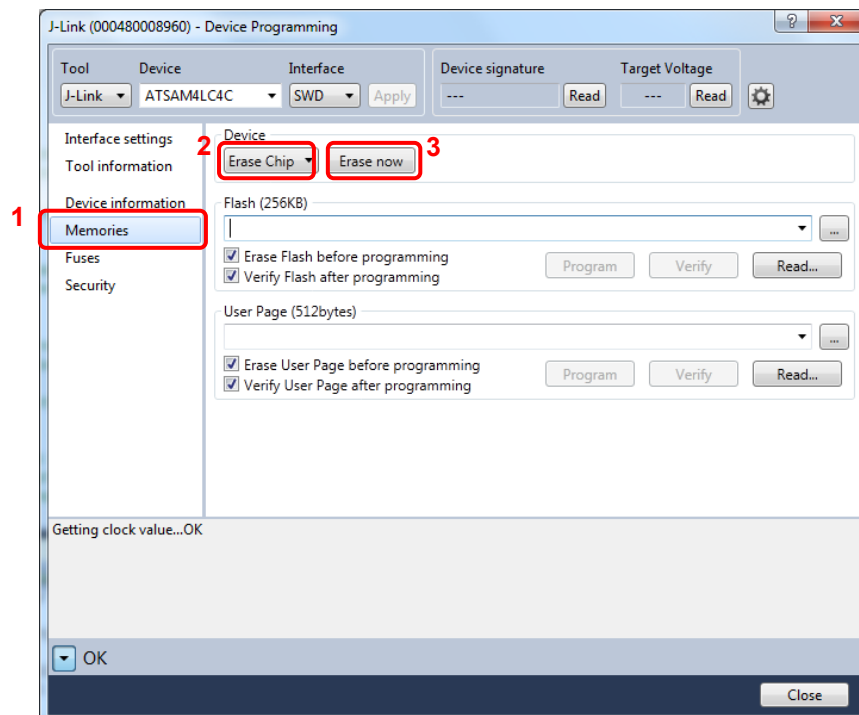
- Open Atmel Studio 6.1
- In *Tools* menu, select *Device Programming*



- A pop-up window is displayed: as shown in the below screenshot, select **J-Link** as the *Tool*, **ATSAM4LC4C** in the *Device list*, **SWD** as *Interface* then click on **Apply** button




On the left column of the pop-up window, click on **Memories** (1), select **Erase Chip** in the drop-down menu (2) then click on **Erase now** button (3).



**RESULT** Flash has been erased.



## TO DO Program SAM-BA in internal flash

- As show below click on the  icon and select the `sam4l_sam-ba_image.hex` file which is the image of SAM-BA boot monitor.

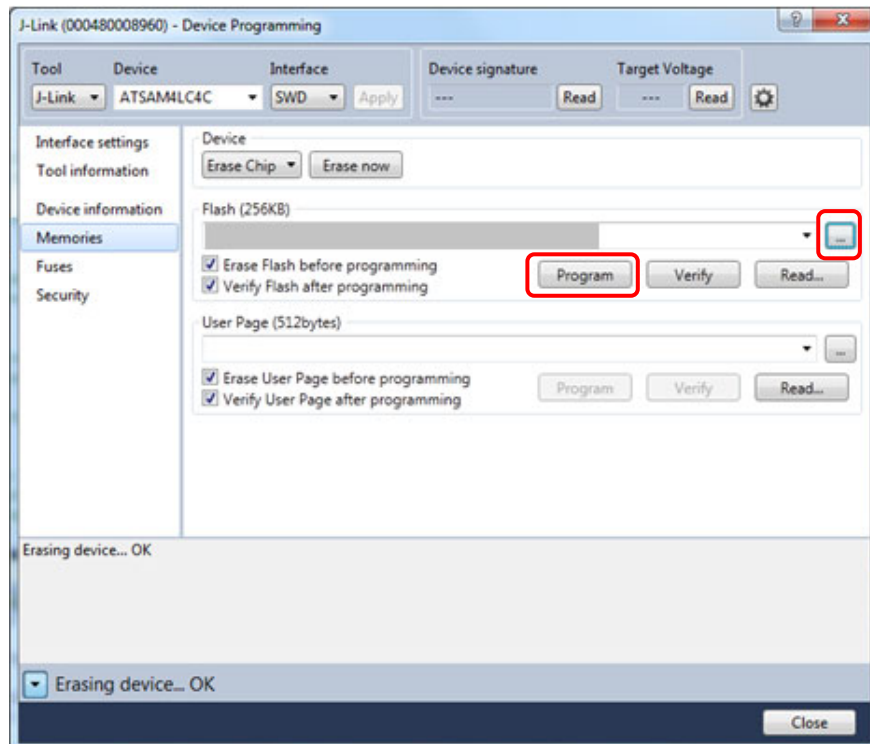
This file is located in a different location based on the delivery type:

### Atmel Training Executable Delivery case

- Find the file in the “`AN-4574_SAM4L-EK_Using_SAM-BA\assignments`” folder (which is located in the `ATMEL_TRAINING` installation folder)

### Atmel Studio Extension Delivery case

- Find the file in the `Hands-on Documentation` project folder
- Click on Program



## RESULT SAM-BA is now programmed in internal flash.



## INFO

As said previously, there is another method to program SAM-BA for users who don't have Atmel Studio.

This method, on a programming script and command lines, is described in the application note called “SAM-BA for SAM4L”: [http://www.atmel.com/Images/Atmel-42051-SAM-BA-for-SAM4L\\_Application-Note\\_AT03454.pdf](http://www.atmel.com/Images/Atmel-42051-SAM-BA-for-SAM4L_Application-Note_AT03454.pdf).

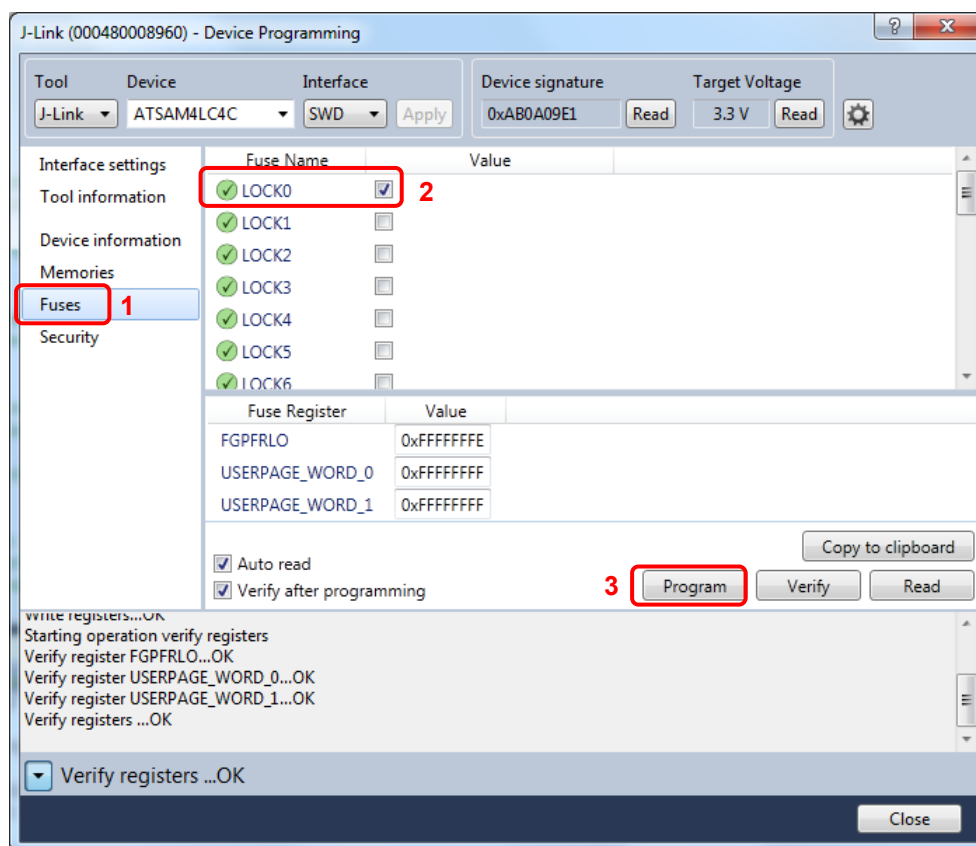


## TO DO

Lock the flash region where SAM-BA has been programmed.

In order to protect SAMBA from being erased, the flash lock bit 0 must be set.

- On the left of the window, click on **Fuses** (1), tick the checkbox of the **LOCK0** bit (2), then click the **Program** button (3)



## RESULT

Flash region 0 has been locked. SAM-BA is now protected and cannot be erased, unless user intentionally unlocks the lock bit before or perform an erase command through JTAG/SWD.

## 5. Assignment 2: Program the Application in SAM4L Flash using SAM-BA

In this assignment, we are going to program an application in flash, and see how it can co-exist with SAMBA in internal flash.

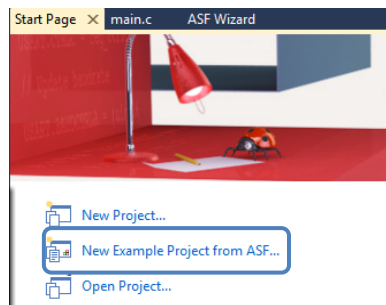
In order to use SAM-BA programming feature, the application will be compiled in Atmel Studio 6.1, but it will be programmed through USB using SAM-BA GUI.



### TO DO

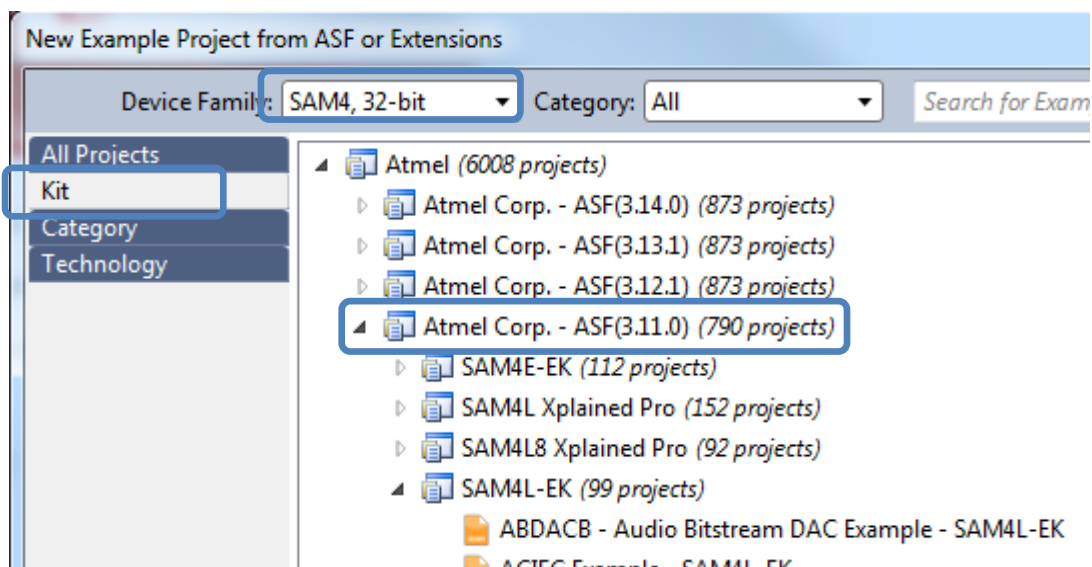
Open an application code from ASF (Atmel Software Framework) and compile it with Atmel Studio 6.1.

- Firstly close the programming tool windows to come back to the start screen of Atmel Studio 6.1
- Click on *New Example Project from ASF*:

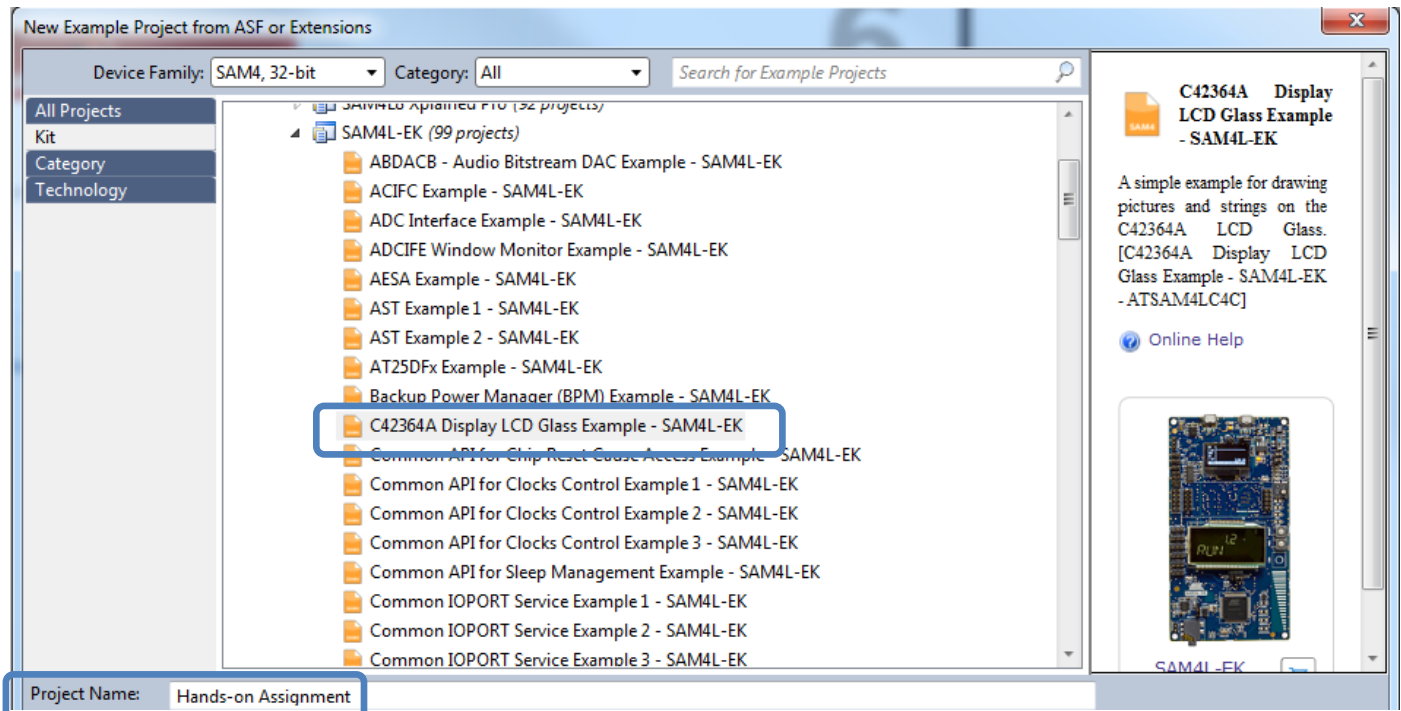


In the New Example Project window:

- Select Device Family SAM4, 32-bit
- Select Kit section and click on *Atmel > ASF Version 3.11.0 > SAM4L-EK*



- Then select C42364A Display LCD Glass Example - SAM4L-EK in the example list
- Rename the Project to “[Hands-on Assignment](#)”




- Finally, select the following location to save your new project:

#### Atmel Training Executable Delivery Case

- Save the [Hands-on Assignment](#) project to:  
“[AN-4574\\_SAM4L-EK\\_Using\\_SAM-BA\assignments](#)”  
(folder located in the ATMEL\_TRAINING installation folder).

#### Atmel Studio Extension Delivery Case

- Add the [Hands-on Assignment](#) project to the [Hands-on Documentation](#) solution
- Click OK and accept the different license agreements and click on *Finish*
- Build the project as is by clicking on the  icon or by pressing the *F7* key



#### **RESULT**

The project example is now compiled and can be programmed in flash using SAM-BA 2.12.



#### **INFO**

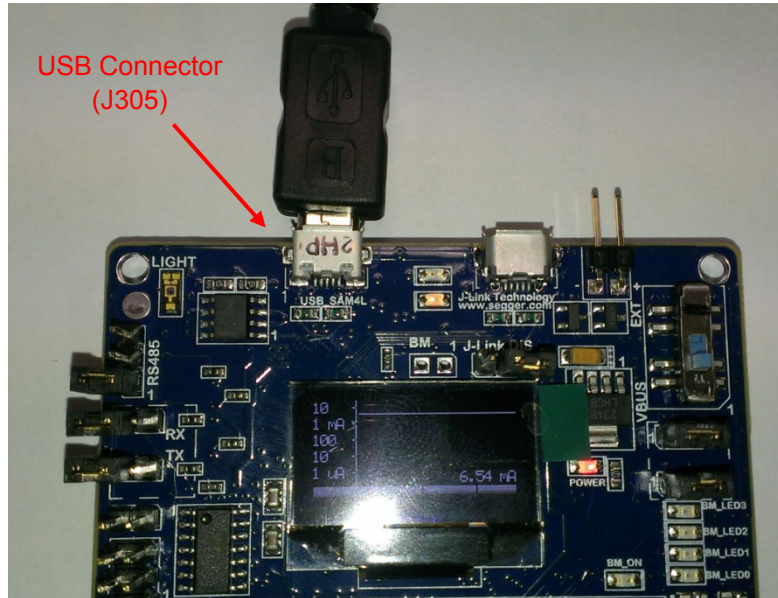
The purpose of this example is to switch on the segment LCD display backlight and all the segments and icons.



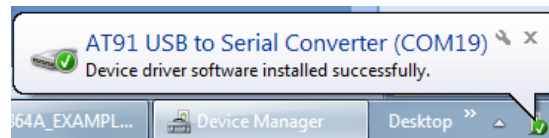
## TO DO

Program the application via USB with SAM-BA 2.12.

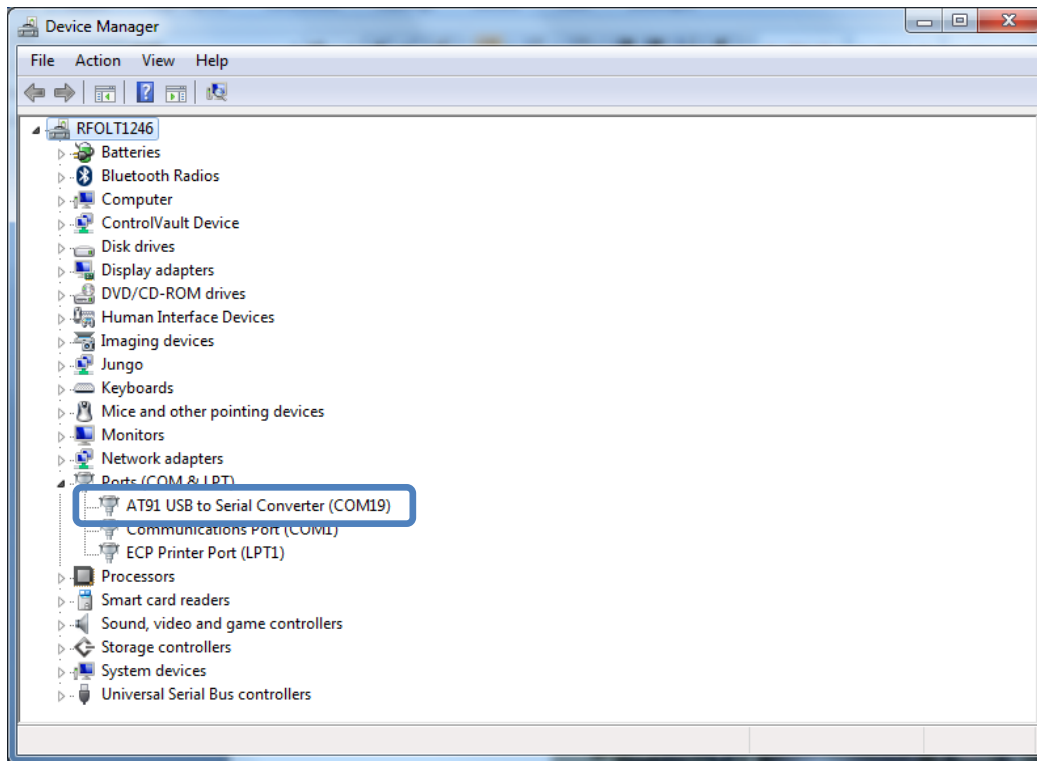
- Remove the USB cable from the USB J-Link OB connector and power-up the board using the standard USB connector (J305) as shown in the below picture:



SAM-BA is executed and the AT91 USB to Serial driver should be automatically installed (first time use only):



- You should have an AT91 USB CDC COM port created in the Windows Device Manager

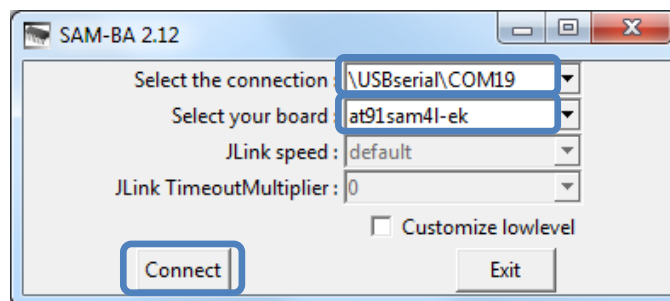


#### INFO

If the driver mounted is not AT91 USB to Serial, the driver must be updated manually. The .inf file to select is located in SAM-BA install directory: [Atmel/sam-ba\\_2.12/drv](http://Atmel/sam-ba_2.12/drv).



- Start SAM-BA 2.12
- Select `\\USBserial\\COMxx` as connection and `at91sam4l-ek` as the board, then click on Connect




#### INFO

If the `at91sam4l-ek` board does not appear in the board list, verify that samba patch 5 has been correctly installed on your computer.



The SAM-BA GUI main window is now displayed.

The *Send File* command is going to be used to program the binary file of the application in flash.

- Click on the  icon (1) and browse your computer to select the binary file to program



#### TIPS

The file to select is [Hands-on Assignment.bin](#) and is located in the [Hands-on Assignment/Debug](#) folder.

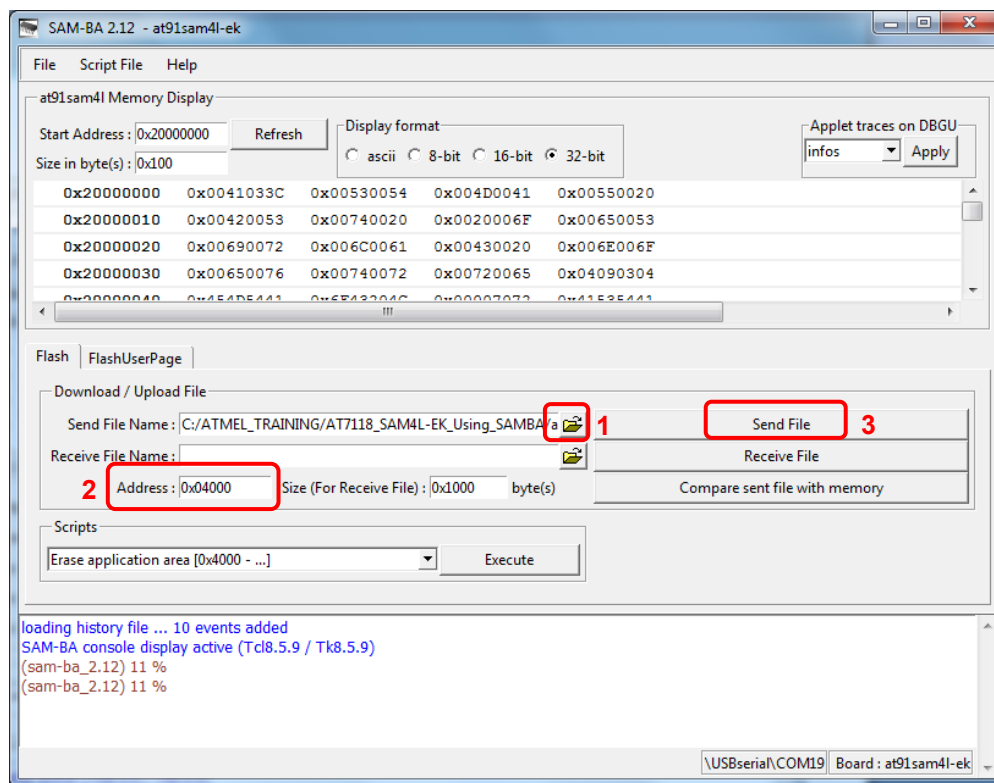
- Change the *Address* field to [0x4000](#) (2)



#### INFO

As explained in introduction, flash address range from 0x00000 to 0x04000 is dedicated from SAM-BA monitor. So the application must be programmed starting for address 0x04000.

- Click on [Send File](#) button (3)



**RESULT**

The application is now programmed in flash memory from address 0x04000.

- Close SAM-BA 2.12 and reset the board
- Considering the configuration word in the flash User Page has not been modified (0xFFFFFFFF), the application should start. If not, could you explain why?

**TIPS**

A step was missing in this assignment.

To use SAM-BA together with an application the user needs to link the application at 0x4000.

As a consequence, the linker file of the Atmel Studio 6.1 project must be modified before compiling the application.

The addresses used to jump to a function are hardcoded by the linker file. So if the code is moved in another memory address range without changing the way it is linked, the code will jump to a wrong address during its execution.

## 6. Assignment 3: Update Application Linker File

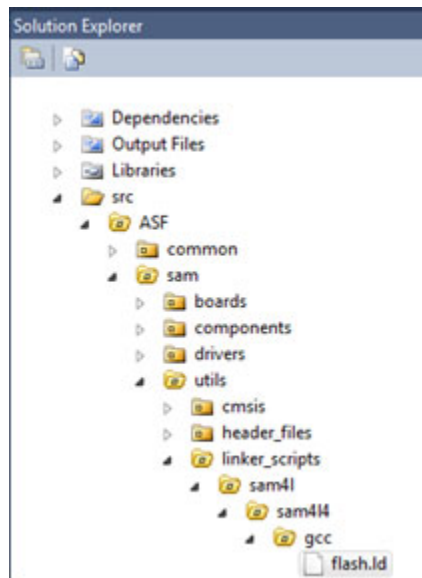
In this assignment, we will firstly modify the linker file of the Atmel Studio 6.1 application project, then the new binary file will be programmed with SAM-BA 2.12 as done previously.



### TO DO

Modify the linker file of the application in Atmel Studio 6.1.

- In the C42364A project example under Atmel Studio 6.1, open the [flash.ld](#) file (linker file), which is located in `src/ASF/sam/utils/linker_scripts/sam4l/sam4l4/gcc`



- Browse the file and modify the below value in red:

```
rom (rx) : ORIGIN = 0x00000000, LENGTH = 0x00040000 /* flash, 256K */
```

must be replaced by:

```
rom (rx) : ORIGIN = 0x00004000, LENGTH = 0x0003C000 /* flash, 256K */
```

- Save the changes by clicking on the icon
- Rebuild the solution by clicking on menu *Build > Rebuild Solution*



### WARNING

The use of the Rebuild Solution command is mandatory. The use of Build Solution won't work here because it does not take into account the changes made in the linker file.




### RESULT

The linker file has been modified and a new binary has been generated. It can be now loaded in flash memory using SAM-BA 2.12.



### TO DO

Program the application linked at address 0x4000 with SAM-BA 2.12.

- Reprogram SAM-BA on the target using previous steps described in Assignment 1
- Open SAM-BA 2.12
- Select `\\USBserial\\COMxx` as connection and `at91sam4l-ek` as the board, and then click on [Connect](#)
- Click on the  icon and browse your computer to select the binary file to program



### TIPS

The file to select is [Hands-on Assignment.bin](#) and is located in the [Hands-on Assignment/Debug](#) folder.

- Change the *Address* field to `0x04000`
- Click on [Send File](#) button
- Close SAM-BA 2.12 and reset the board



### RESULT

The application should now be executed: the Segment LCD display is switched on.

## 7. Assignment 4: Update SAM4L Flash User Page to Allow Boot Selection between SAM-BA Monitor and the User Application

In the previous assignments, you saw how to program both SAM-BA monitor and a user application in internal flash and execute the application.

In this assignment, you are going to experiment how to boot from SAM-BA monitor while it co-exists with the application in flash.

As explained in introduction, the behavior of the device at reset (boot from SAM-BA, or boot from the application) can be modified through the Flash user page which contains the monitor configuration word.

SAM-BA monitor is executed if the specific word 0x4D4153 ("SAM" in ASCII) is written or if a force pin is configured and detected as active (see flowchart at the beginning of this document).

In this assignment you will have to configure the PIO PC03 (i.e. push button PB0) as the force pin. Thus when the push button PB0 is pressed while powering up the board, SAM-BA monitor is executed. If not, SAM-BA jumps directly to the application then executes it.

As we forced the application execution in the previous assignment, we first have to erase the whole content of the flash then program again SAM-BA as we did in the first assignment.



### TO DO

Erase the whole content of the flash and program SAM-BA at address 0x0000 with Atmel Studio 6.1.

- Change the power interface of the SAM4L-EK and connect the USB cable to the USB J-Link OB connector (J1)
- Using Atmel Studio, erase the flash content by following the same steps as the first assignment of this hands-on and program the image of SAM-BA Boot



### RESULT

SAM-BA has been re-programmed in flash. There is no application code at address 0x4000 for now.



### TO DO


Read the flash user page with SAM-BA 2.12.

- In order to boot from SAM-BA boot, remove the USB cable from the USB J-Link OB connector and power-up the board using the standard USB connector (J305)
- Start SAM-BA 2.12, select [\USBserial\COMxx](#) as connection and [at91sam4l-ek](#) as board then click on [Connect](#). The SAM-BA GUI main window is now displayed.



## INFO

The user page also contains BODs and watchdog settings, and it is up to the user to not to modify it. In order to keep those settings, the right procedure to modify the user page is to read it first, modify it using a hexa file editor (Fhred) then program it again.

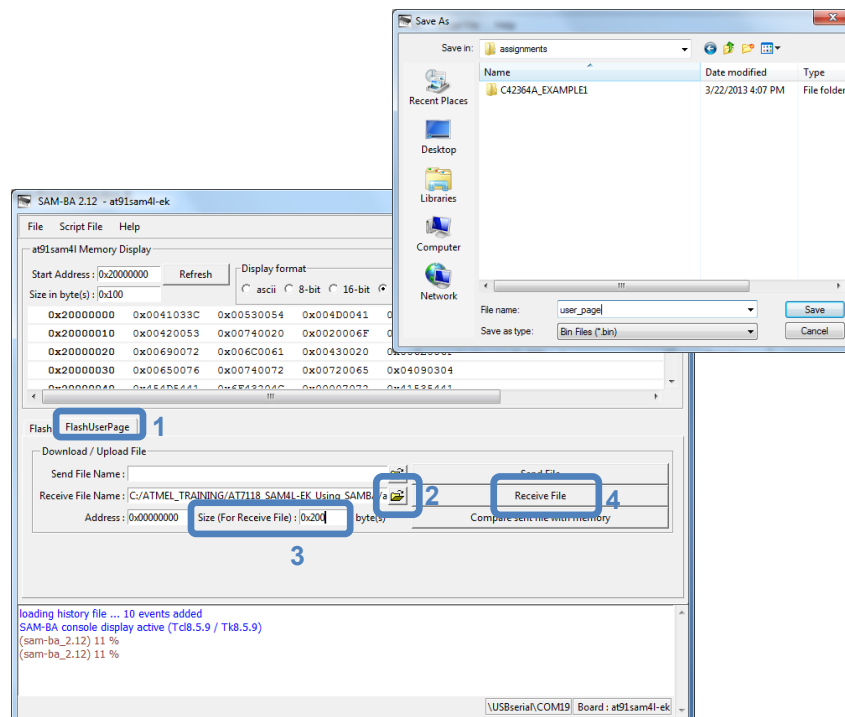
- In SAM-BA GUI main window, select the **FlashUserPage** tab (1)
- In order to retrieve the flash user page, we have to use the *Receive File* function of SAM-BA GUI. Click on the  icon on the left of the *Receive File* button (2)
- Browse your computer to define the location of the file to be received. Name this file **user\_page** (or **user\_page.bin**), and save it in the following directory:

### Atmel Training Executable Delivery case

- “AN-4574\_SAM4L-EK\_Using\_SAM-BA\assignments” folder (which is located in the ATMEL\_TRAINING installation folder).

### Atmel Studio Extension Delivery case

- Hands-on Documentation project folder
- Change the *Size (For Receive File)* to **0x200** (3)
- Finally click on **Receive File** button to retrieve the flash user page (4)



## RESULT

Flash user page has been saved and can be now modified.



## TO DO

Compute and modify the monitor configuration word in flash user page.



## INFO

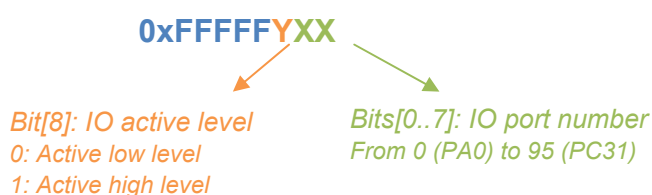
The push button PB0 (connected to PC03 on the SAM4LC4) will be used as the force pin to choose if the device boots from SAM-BA monitor or the application. When the active level (**low level** for the push button) is detected, SAM-BA monitor has to be executed. If not, the application (not loaded yet) is executed.

- Compute the value of the configuration word using the below tips

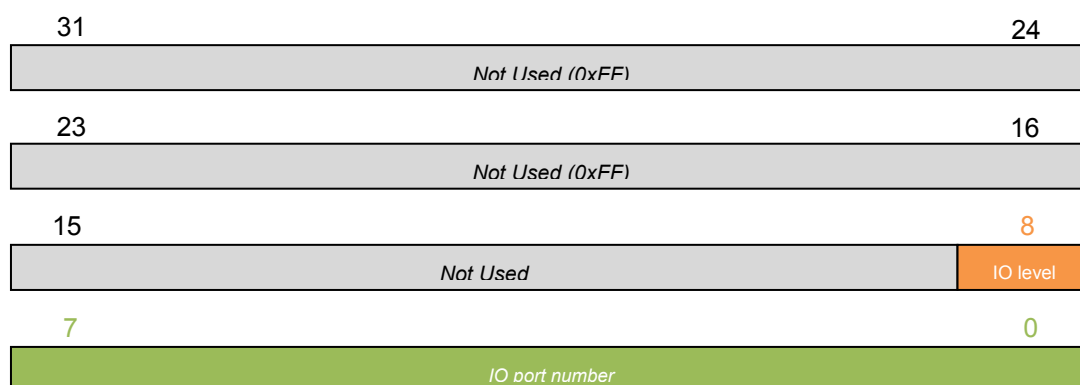


## TIPS

The configuration word has the following format:



Or:





## TIPS


The I/O port number of PC03 is given in the below extract of the datasheet, then must be converted in hexadecimal:

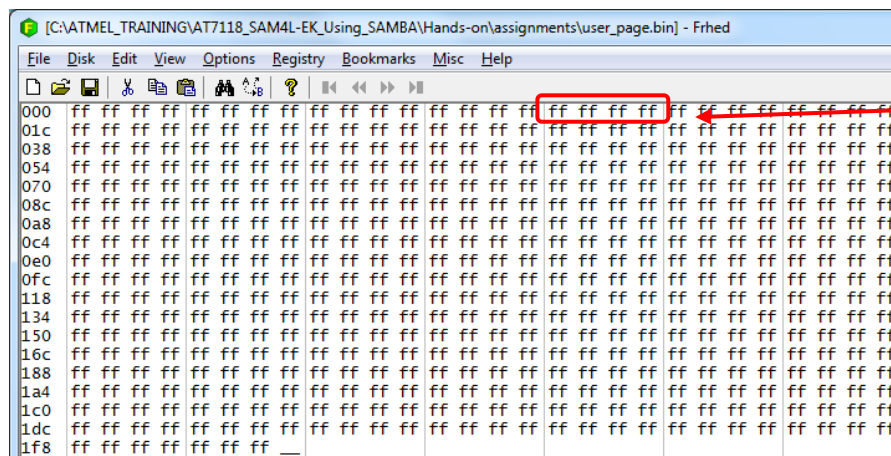
ATSAM4LC		ATSAM4LS		Pin	GPIO	Supply
QFN	VFBGA	QFN	VFBGA			
94	D8	94	D8	PB15	47	LCDC
1	A10	1	A10	PC00	64	VDDIO
2	C8	2	C8	PC01	65	VDDIO
3	C7	3	C7	PC02	66	VDDIO
4	B7	4	B7	PC03	67	VDDIO
9	C5	9	C5	PC04	68	VDDIO
10	C6	10	C6	PC05	69	VDDIO
11	B6	11	B6	PC06	70	VDDIO



## TIPS

Active level of the push button is low.


- Open Frhed Hex Editor
- Click on the **Open** icon  then browse you computer to select the *user\_page.bin* file
- Finally click on **Open**. The User page is displayed, and the configuration word to update is located at offset 0x10:



- Modify the configuration word with the value you computed



**WARNING** LSB are displayed first by Frhed. For example, if value is 0xABCDEF01, Frhed displays it as: 01 EF CD AB.

- Save the changes by clicking on the icon  and close Frhed Editor




**RESULT** The monitor configuration word has been modified.





## TO DO

Write the new flash user page with SAM-BA 2.12.

- If you closed SAM-BA 2.12, open it again and connect to the board through USB
- Select the [FlashUserPage](#) tab (1)
- In order to write the flash user page, we have to use the *Send File* function of SAM-BA GUI. Click on the  icon on the left of the *Send File* button (2)
- Browse your computer and select the [user\\_page.bin](#) file (3).

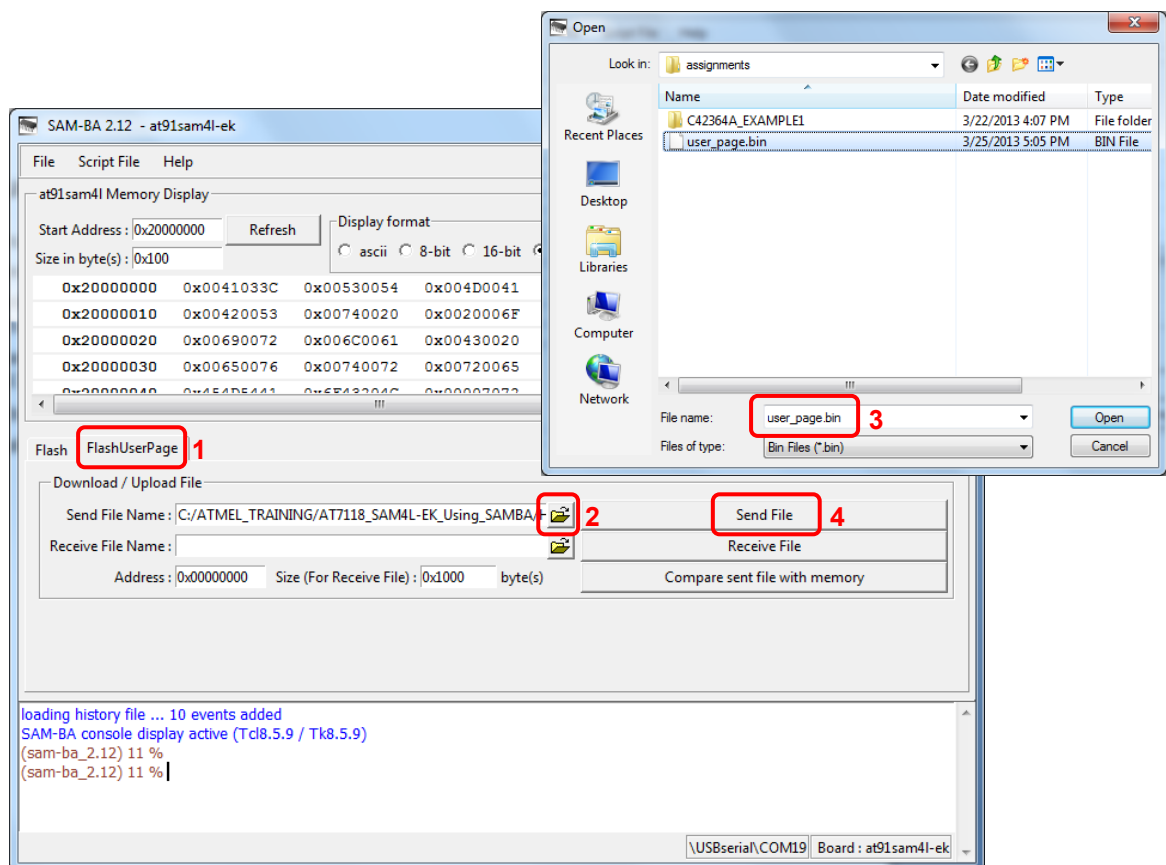
This file should be in the following directory:

### Atmel Training Executable Delivery case

- “[AN-4574\\_SAM4L-EK\\_Using\\_SAM-BA\assignments](#)” folder (which is located in the ATMEL\_TRAINING installation folder).

### Atmel Studio Extension Delivery case

- [Hands-on Documentation](#) project folder
- Click on [Open](#) and on [Send File](#) button to retrieve the flash user page (4)




## RESULT

The flash user page has been modified and programmed.



## TO DO

Program the user application with SAM-BA 2.12.

- Select the [Flash](#) tab
- In order to write the flash user page, we have to use the *Send File* function of SAM-BA GUI. Click on the  icon on the left of the *Send File* button
- Browse your computer and select the [Hands-on Assignment.bin](#) located in:

### Atmel Training Executable Delivery case

- [Hands-on Assignment/Debug](#) folder

### Atmel Studio Extension Delivery case

- [Hands-on Documentation](#) project folder
- Click on [Open](#)
- Change the *Adress* field to [0x04000](#)
- Finally click on [Send File](#) button to retrieve the flash user page



## RESULT

Application has been programmed at address 0x4000.

- SAM-BA 2.12 can be closed and you can switch-off the board
- You can now verify if the value of the monitor configuration word you set is correct or not. If the configuration word is correct, the device should behaves as follow:
  - By default when the board is powered through USB connector (J305), the application is executed
  - When the board is powered through J305 **and PB0 is pressed at the same time**, SAM-BA monitor is executed. You should see *the AT91 USB CDC COM* driver mounted on your PC.



## INFO

If the device does not behave as explained previously, you have to check the value of the configuration word and restart from the beginning of the assignment 4.

## 8. Conclusion

As you have seen in this hands-on, it is possible to make SAM-BA co-existing with an application in internal flash. If SAM-BA monitor is not needed, it can be erased so that the whole content of the flash can be used for the user application.

For more information on SAM-BA for SAM4L devices, you can refer to the application note called “SAM-BA for SAM4L”:

This application note comes with a package containing the binary file of SAM-BA, its source code and a programming script for non-Atmel Studio users, and also a ready-to-use linker file for GCC, Atmel Studio or IAR™ users. This application note can be downloaded for the below link:

[http://www.atmel.com/Images/Atmel-42051-SAM-BA-for-SAM4L\\_Application-Note\\_AT03454.pdf](http://www.atmel.com/Images/Atmel-42051-SAM-BA-for-SAM4L_Application-Note_AT03454.pdf)

## 9. Revision History

Doc. Rev.	Date	Comments
42245A	02/2014	Initial document release



Enabling Unlimited Possibilities®

**Atmel Corporation**

1600 Technology Drive  
San Jose, CA 95110  
USA

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

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

[www.atmel.com](http://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 Bldg.  
1-6-4 Osaki, Shinagawa-ku  
Tokyo 141-0032

JAPAN

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

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

© 2014 Atmel Corporation. All rights reserved. / Rev.: 42245A - 02/2014

Atmel®, Atmel logo and combinations thereof, Enabling Unlimited Possibilities®, SAM-BA®, and others are registered trademarks or trademarks of Atmel Corporation or its subsidiaries. ARM® and others are registered trademarks of ARM Ltd. Windows® is a registered trademark of Microsoft Corporation in U.S. and or other countries. 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.