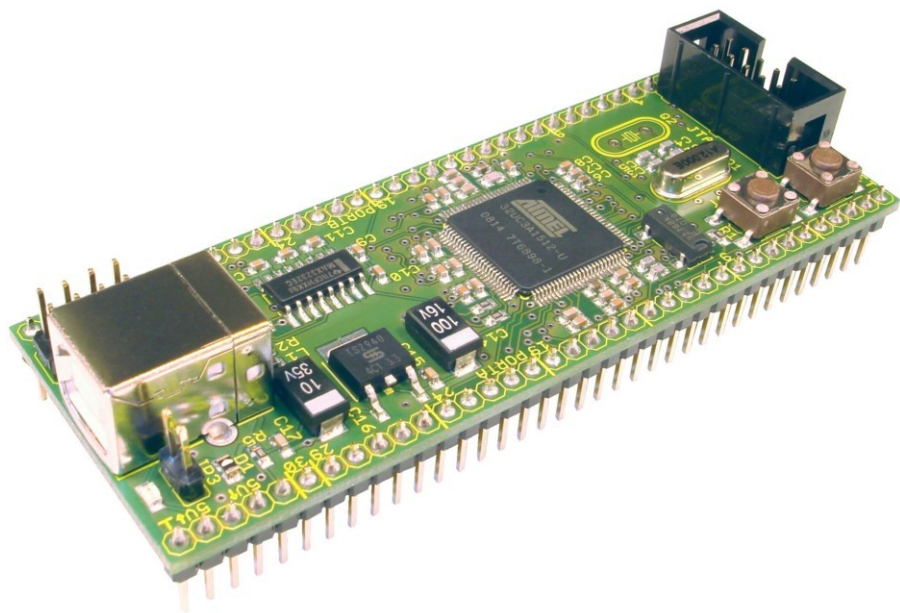
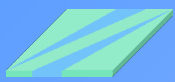


# **First Step**

*From Download to  
the Programming*

**Revision from 23.03.2011**





## Directory

<b>1.</b>	<b>Instructions</b>	<b>3</b>
<b>2.</b>	<b>Download</b>	<b>4</b>
<b>3.</b>	<b>Installation</b>	<b>5</b>
<b>4.</b>	<b>AVR32 Studio</b>	<b>6</b>
<b>4.1.</b>	<b>New Project</b>	<b>8</b>
<b>4.2.</b>	<b>Creation of a new Source Code File</b>	<b>8</b>
<b>4.3.</b>	<b>Adding the Library</b>	<b>10</b>
<b>4.4.</b>	<b>Compilation of the Project</b>	<b>12</b>
<b>5.</b>	<b>Hardware</b>	<b>13</b>
<b>5.1.</b>	<b>Start the Boot Loader</b>	<b>13</b>
<b>5.2.</b>	<b>Installation of the USB-Driver</b>	<b>14</b>
<b>6.</b>	<b>Programming</b>	<b>15</b>
<b>6.1.</b>	<b>Expanding the Program</b>	<b>15</b>
<b>6.2.</b>	<b>Installation of the Programmers</b>	<b>18</b>
<b>6.3.</b>	<b>Programming with AVR32 Studio</b>	<b>19</b>
<b>7.</b>	<b>Sources</b>	<b>21</b>

# 1. Instructions

You will find similar information also on the web page of Atmel. In this document we introduce you with the help of our product, AVR32 module, our experiences with AVR32.

This document allows you a quick access to the AVR32 controller's family. It leads you from downloading of the free software and tools of Atmel up to the programming of the controller.

One knows that the first step is very hard. We have done this step and present it in this document.

To make acquaintance with AVR32 you need the following:

- Software
  - AVR32 Studio
  - AVR32 GNU Toolchain
  - AVR32 UC3 Software Framework
  - FLIP 3.3.1 or higher
  
- Hardware
  - AVR32-Module AL-UC3AEB with AT32UC3A1512 Controller **or**
  - AVR32-Module AL-UC3BMB with AT32UC3B0256 Controller **or**
  - AVR32-Board AL-UC3AVRBIT with AT32UC3A0512 Controller
  
- Programmer
  - AVR JTAGICE MKII **or**
  - USB-Boot loader (on each Controller UC3-series installed)

There are two options to program AVR32 controller. The first one is AVR JTAGICE MKII. A big disadvantage of this programming option is a high expense and that is why it is interesting for a few. As a second option there is the free USB-Boot Loader to choose. How the name already tells, it needs only the USB connection with the computer.

Every controller of the UC3 series is preprogrammed with the USB-Boot Loader, it means that all three above mentioned hardwares can be programmed with an USB cable.

## 2. Download

You will find the complete software on the web page of Atmel.

- **AVR32 Studio 2.6 for Windows (installer)** (272 MB, revision 2.6, updated 9/10)  
[http://www.atmel.com/dyn/products/tools\\_card.asp?tool\\_id=4116](http://www.atmel.com/dyn/products/tools_card.asp?tool_id=4116)  
this file contains the development environment for AVR32 - controller. Before you can download this software, you should fill every (\*) field. After that appears a window with download link.
- **AVR32 GNU Toolchain** (53 MB, revision 2.1.6, updated 3/09)  
[http://www.atmel.com/dyn/products/tools\\_card.asp?tool\\_id=4118](http://www.atmel.com/dyn/products/tools_card.asp?tool_id=4118)  
this file contains the C libraries, flash programming tools, assembler, linker and compilers for Windows and Linux
- **AVR32 UC3 (A oder B) Software Framework**  
[http://www.atmel.com/dyn/products/tools\\_card.asp?tool\\_id=4192](http://www.atmel.com/dyn/products/tools_card.asp?tool_id=4192)  
this file contains numerous examples, drivers, source codes, ready projects, HTML documentation, software services...
- **FLIP 3.4.2 for Windows (Java Runtime Environment included)**  
[http://www.atmel.com/dyn/products/tools\\_card.asp?tool\\_id=3886](http://www.atmel.com/dyn/products/tools_card.asp?tool_id=3886)  
FLIP (**FL**exible **I**n-system **P**rogrammer) supports In system programming of Flash devices through RS232, USB or CAN. This file contains also the driver of USB-Boot Loader.

## 3. Installation

1. As first you have to install **AVR32Studio-2.x.x-Setup.exe**

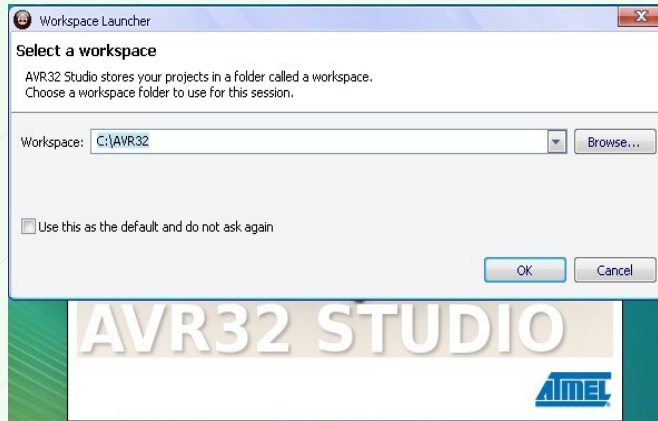


Follow the instructions in the window and install AVR32 studio. The installation will take some minutes. Let your Firewall allow the complete installation.

2. As a next step you have to install **avr32-gnu-toolchain-2.x.x.exe**
3. Unzip **AVR-SoftwareFramework-2.3.1.zip**, e.g to C:\
4. The last step is the installation of **JRE - Flip Installer – 3.4.2.exe**

# 4. AVR32 Studio

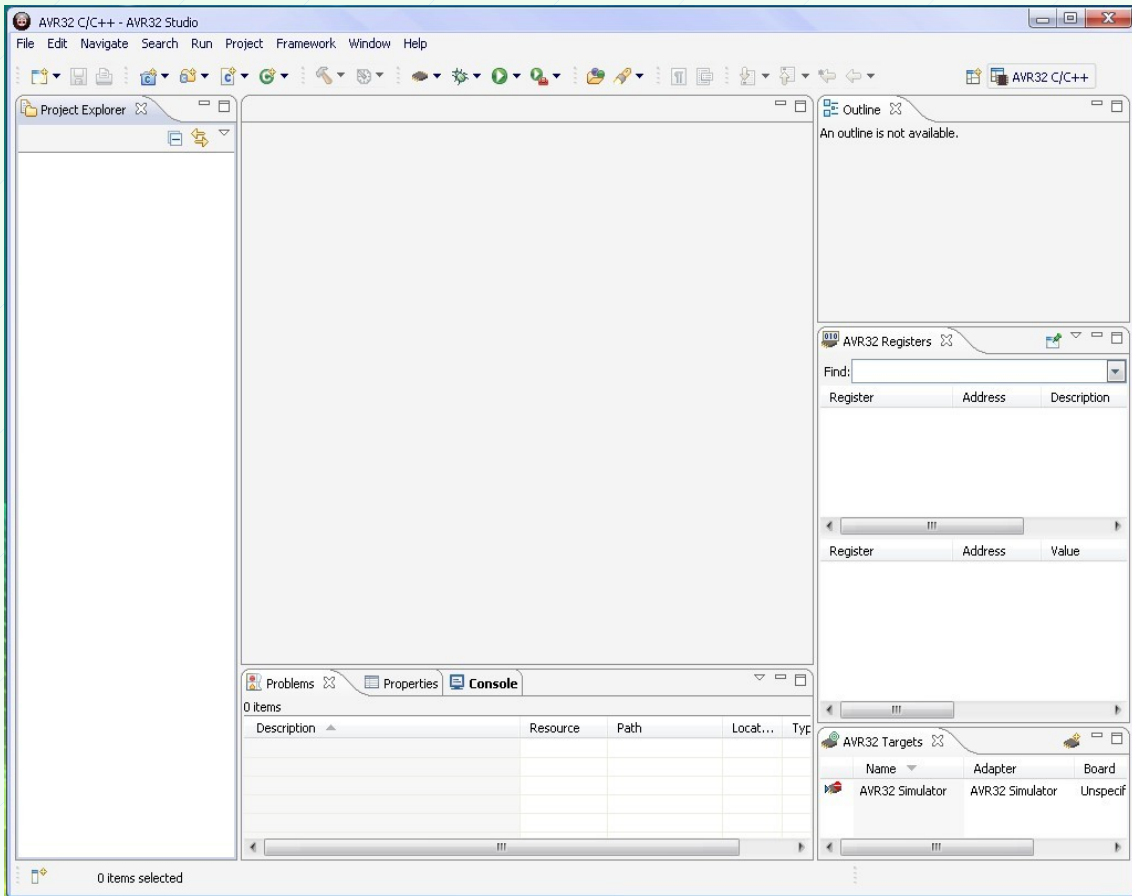
Start AVR32 Studio *Start* → *Programmen* → *Atmel AVR Tools* → *AVR32Studio*.



Select in the field *Workspace* in which folder your project should be saved and click the button "OK".



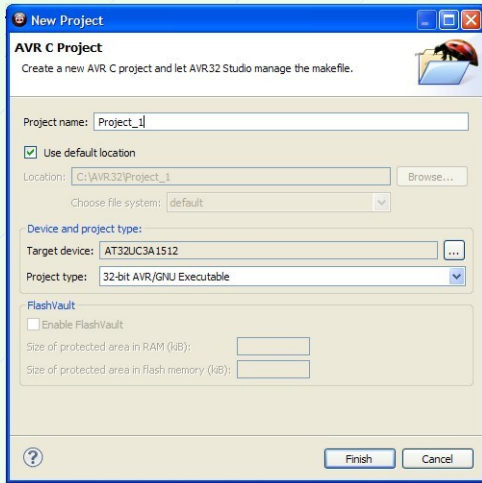
After that the window "Welcome" appears. Here you can get more information about the software, hardware and programmer. Click the white cross behind the word "Welcome" to close this window.



The upper picture of development environment appears. As you see, this development environment has less similarity with AVR 4 studio.

In the next chapters we will learn more about the AVR32 studio.

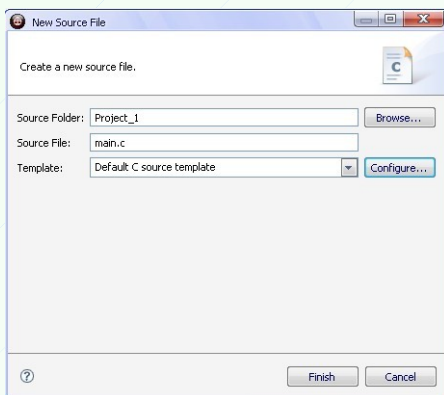
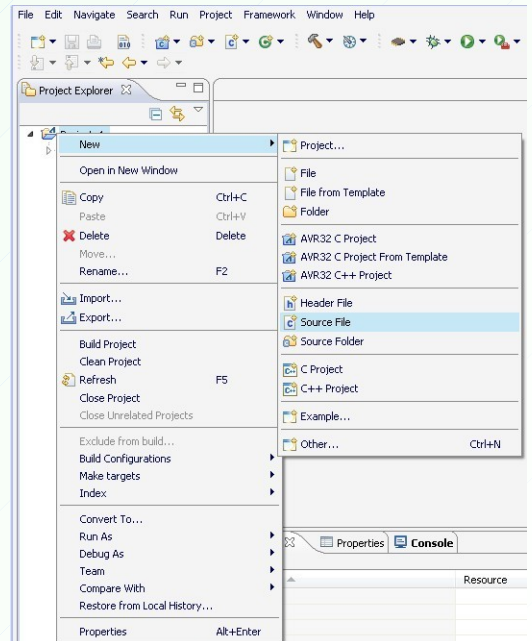
## 4.1. New Project



Start a new Project *File* → *New* → *AVR32 C Project*. Give a name to the project in the field *Project name*, e.g. **Project\_1**. Choose the controller of your hardware in the field *Target device*, e.g. for AVR32 module **UC3A1512**, and in the field *Project type* **32-bit AVR/GNU Executable**. Click afterwards the button "Finish".

## 4.2. Creation of a new Source Code File

Create a new source code file *File* → *New* → *Source File* or click with the right mouse button on your Project, e.g. **Project\_1**, and choose *New* → *Source File*.



Give a name in the field *Source File*, e.g. **main.c** and click the button "Finish".

Now we will write a small program in the new created source file. In this program we change periodically the level of the pin 3 on the port A.

```
#include "gpio.h" //driver of atmel include in AVR32 UC3A Framework C:\AT32UC3x-1.x.x\DRIVERS\GPIO
#include "compiler.h" //driver of atmel include in AVR32 UC3A Framework C:\AT32UC3x-1.x.x\UTILS

int main(void)
{
    U32 i; //you will find this definition of >U32< in driver "compiler.h"

    while(1)
    {
        gpio_set_gpio_pin(AVR32_PIN_PA03); //set the pin 3 on port A as high-output

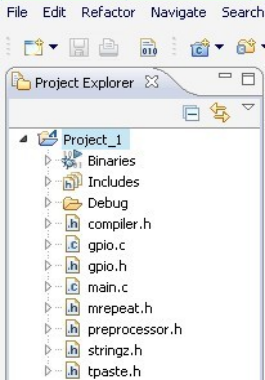
        for(i=0; i<1000; i++); //wait loop

        gpio_clr_gpio_pin(AVR32_PIN_PA03); //set the pin 3 on port A as low-output

        for(i=0; i<1000; i++); //wait loop
    }
}
```

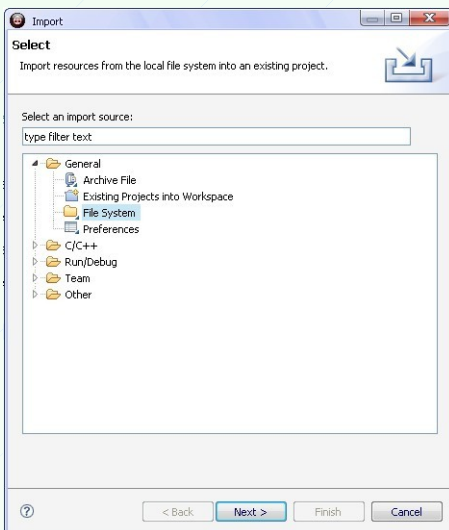
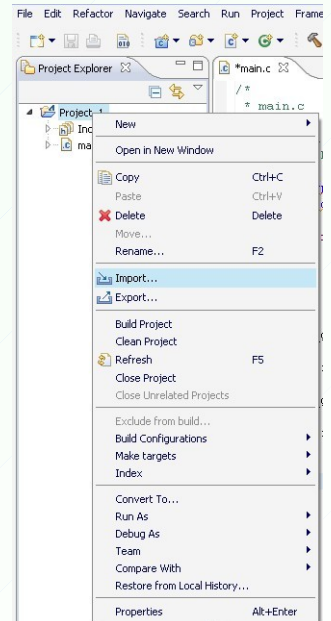
Take over the source code illustrated on top in main.c.

## 4.3. Adding the Library

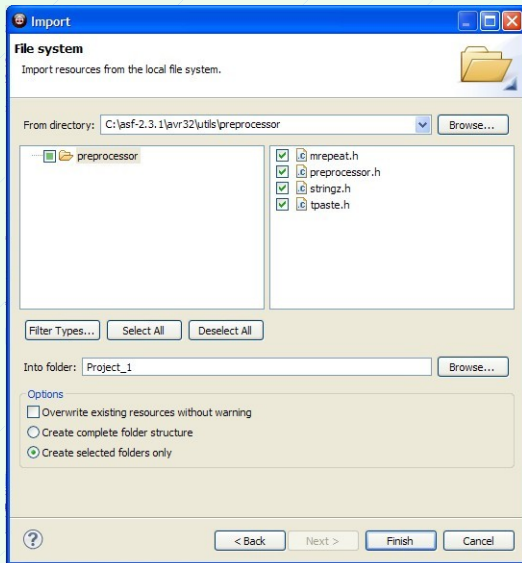


After we have written the program, we have to insert a couple of libraries to our project. You will find these libraries in *AVR32 UC3 Framework*, e.g. in the folder `C:\asf-2.3.1\avr32`

Click with the right mouse button on the project name and choose *Import...* or on the *File* → *Import...*



Choose in the folder *General* → *File System* and click the button „Next >“.



In this window "Import" we can add the libraries to our project. Enter in the field *From directory*: the library folder, e.g. <C:\asf-2.3.1\avr32\utils\preprocessor>. The contents of this folder appears in the lower right field. Choose the required libraries and click the button "Finish".

Unfortunately, it is not possible to include all libraries at a blow if they are in different folders. That's why in our case we have to open the window "Import" three times.

You will find the required libraries in the following folders:

- compiler.h und parts.h → <C:\asf-2.3.1\avr32\utils>
- gpio.h und gpio.c → <C:\asf-2.3.1\avr32\drivers\gpio>
- mrepeat.h, preprocessor.h, stringz.h und tpaste.h → <C:\asf-2.3.1\avr32\utils\preprocessor>

### Important!!!

After you have included all above files, you have to open the header-file *compiler.h* and change the line

```
#include "interrupt.h"
```

with

```
//#include "interrupt.h"
```

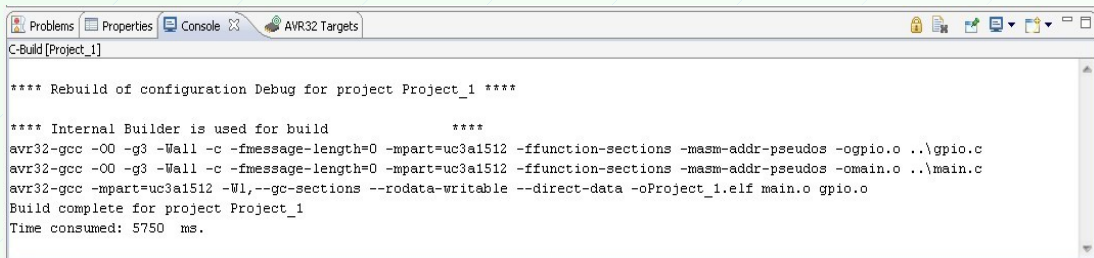
## 4.4. Compilation of the Project

At first you have to save the complete Project File → *Save ALL*.

Compile the project:

- Project → *Build ALL* **or**
- key sequence [*Strg*]+[*B*] **or**
- right mouse button on your project → *Build Project*

You will see the result of the compilation in the window "Console" how it is shown in lower picture.



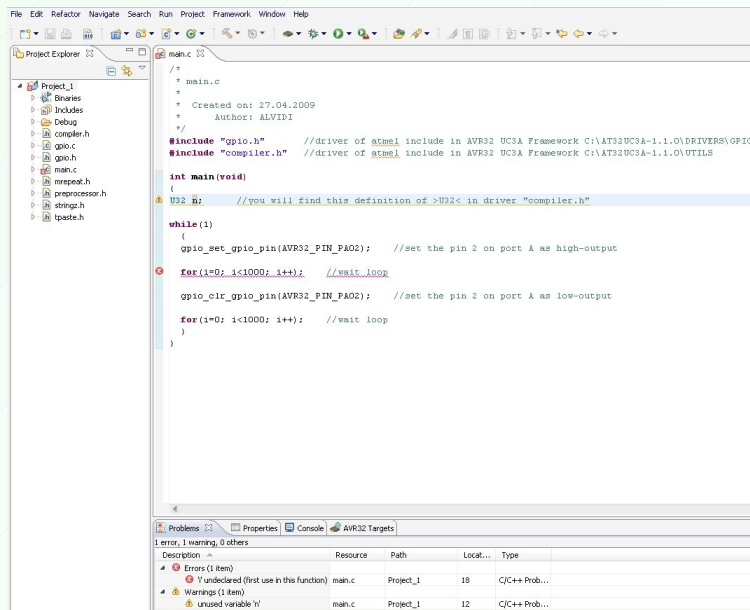
```

C-Build [Project_1]

**** Rebuild of configuration Debug for project Project_1 ****

**** Internal Builder is used for build ****
avr32-gcc -O0 -g3 -Wall -c -fmessage-length=0 -mpart=uc3a1512 -ffunction-sections -masm-addr-pseudos -ogpio.o ../gpio.c
avr32-gcc -O0 -g3 -Wall -c -fmessage-length=0 -mpart=uc3a1512 -ffunction-sections -masm-addr-pseudos -omain.o ../main.c
avr32-gcc -mpart=uc3a1512 -Wl,--gc-sections --rodata-writable --direct-data -oProject_1.elf main.o gpio.o
Build complete for project Project_1
Time consumed: 5750 ms.
    
```

**If** during the compilation mistakes were found in the program, they would appear in the window "Problems". The lower picture introduces this case.



```

main.c
/*
 * main.c
 * Created on: 27.04.2009
 * Author: ALVIDI
 */
#include "gpio.h" //driver of atmel include in AVR32 UC3A Framework C:\AT32UC3A-1.1.0\DRIVERS\GPIO
#include "compiler.h" //driver of atmel include in AVR32 UC3A Framework C:\AT32UC3A-1.1.0\UTILS

int main(void)
{
    U32 B; //you will find this definition of >U32< in driver "compiler.h"

    while(1)
    {
        gpio_set_gpio_pin(AVR32_PIN_PA02); //set the pin 2 on port A as high-output
        for(i=0; i<1000; i++) //wait loop

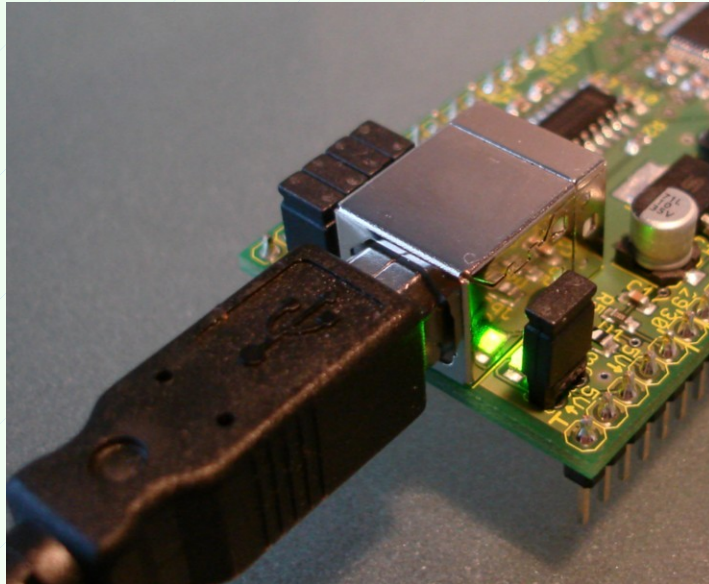
        gpio_clr_gpio_pin(AVR32_PIN_PA02); //set the pin 2 on port A as low-output
        for(i=0; i<1000; i++) //wait loop
    }
}
    
```

Description	Resource	Path	Locat...	Type
1 error, 1 warning, 0 others				
Errors (1 item)				
Undeclared (first use in this function)	main.c	Project_1	10	C/C++ Prob...
Warnings (1 item)				
unused variable 'i'	main.c	Project_1	12	C/C++ Prob...

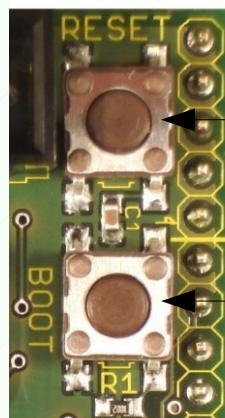
# 5. Hardware

## 5.1. Start the Boot Loader

Connect your hardware via USB with computer, e.g. AVR32-Module. When the jumper JP3 is set on AVR32-Module, the module will be supplied via USB with 5V and the *Power LED* will give green light.



Hold the "BOOT"-key low-pressed and press for a short time the "RESET"-key. Therefore you start the boot loader.



2. press for a short time

1. hold low-pressed

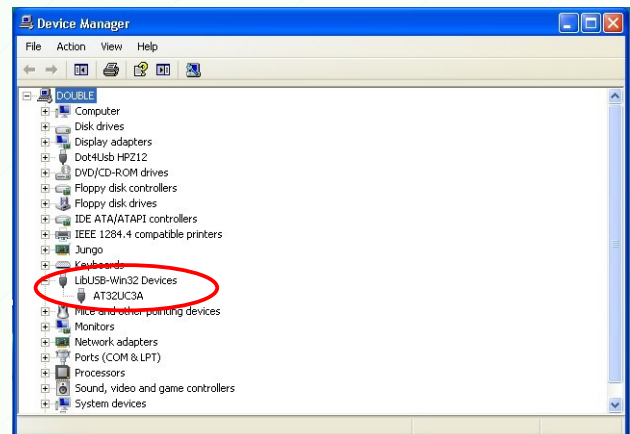
## 5.2. Installation of the USB-Driver

When the USB connection is available and boot loader has been started you get the picture below.



Choose *Install from a list or location (Advanced)* and click the button „Next >“.

Enter in white field the folder of the driver <C:\Program Files\Atmel\FIip 3.4.2\usb> and click the button „Next >“.



After a successful installation appears the left upper picture. In the right picture *Device Manager* the connected device will be visible under **LibUSB-Win32 Devices**→**AT32UC3A**

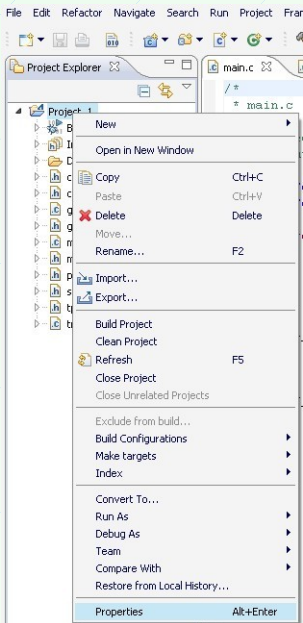
# 6. Programming

## 6.1. Expanding the Program

To program with USB-boot loader we must expand our program with two libraries:

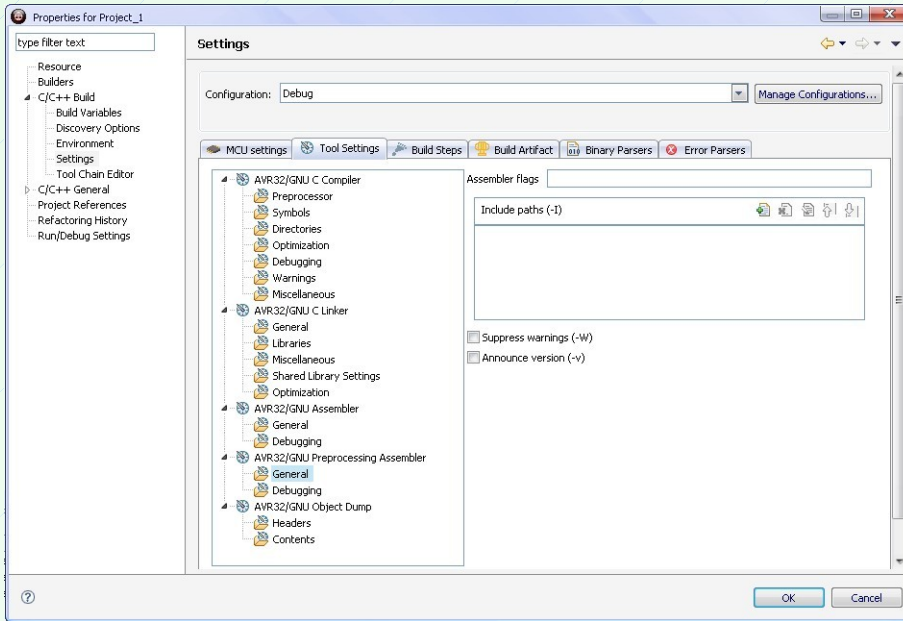
- trampoline\_uc3.h and trampoline\_uc3.S → <C:\asf-2.3.1\avr32\utils\startup>

To include library to the project, see chapter 4.3 *Adding the Library*.



In the next step we have to include two paths to our project. Click with the right mouse button on the project name and choose *Properties* or on *File* → *Properties*

Choose in the left part of this window *C/C++Build* → *Settings*. Click the sub-window *Tool settings* → *AVR32/GNU Preprocessing Assembler* → *General*.



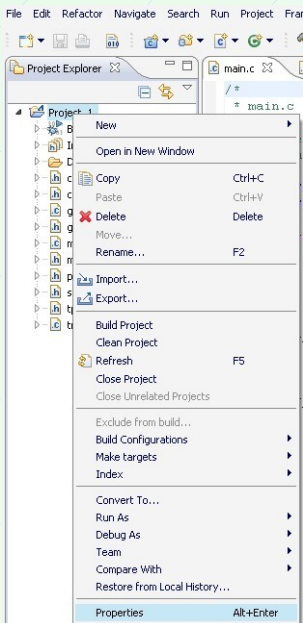
In the right part of this window we are seeing the field *Include paths (-I)*. In this field we have to include two folders. Click the white leaf with green cross and include one by one following two folders:

[C:\asf-2.3.1\avr32\utils](#) and [C:\asf-2.3.1\avr32\utils\preprocessor](#)

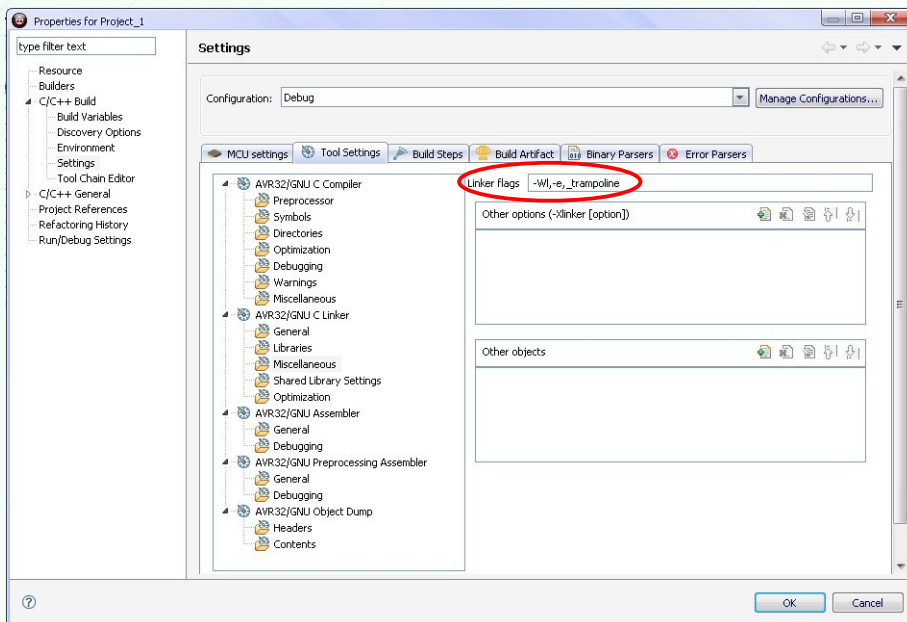


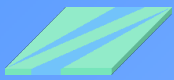
In this step we must inform the linker that we work with the boot loader.

Click with the right mouse button on the project name and choose *Properties* or on *File* → *Properties*



Choose in the left window *C/C++ Build* → *Settings*. Click the sub-window *Tool settings* → *AVR32/GNU C Linker* → *Miscellaneous*. Print in the field **Linker flags** *-Wl,-e, \_trampoline* and confirm the changes with the button "OK,".



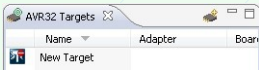


## 6.2. Installation of the Programmer

You will find in pdf-File *AVR32 UC3 USB DFU Bootloader* of Atmel, how to program with boot loader.

See: [http://www.atmel.com/dyn/resources/prod\\_documents/doc7745.pdf](http://www.atmel.com/dyn/resources/prod_documents/doc7745.pdf)

Open the window **AVR32 Targets**. You will find this window also under *Window* → *Show View* → *AVR32 Targets*.



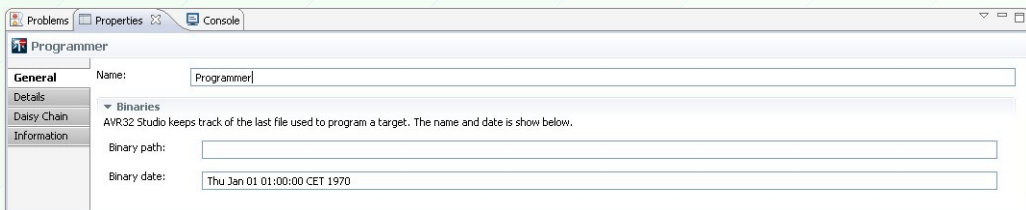
Click on the symbol "Creates a new target" in the gray field. Afterwards appears in the white field *New Target*.



With the right mouse button on *New Target*, choose *Properties*.

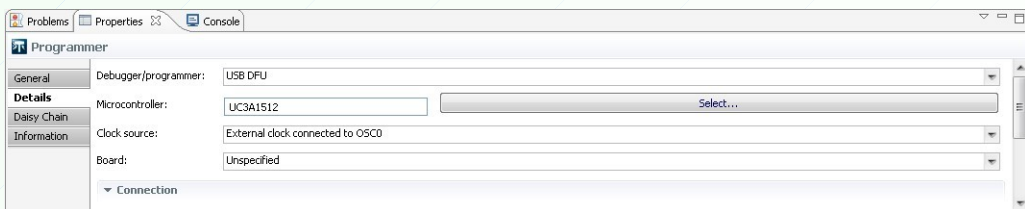
In the window *Properties* we have to configure our programmer.

Write in the field *General* → *Name*: a name of your programmer, e.g. Programmer.



In the window *Properties* → *Details* will be set the following hardware parameters:

- \* - **Debugger/programmer**: USB DFU
- \* - **Microcontroller**: UC3A1512
- \* - **Clock source**: Internal RC oscillator or External clock connected to OSC0

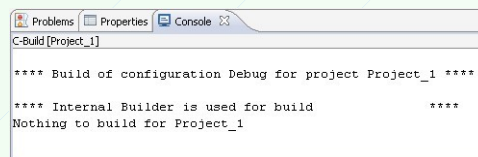


## 6.3. Programming with AVR32 Studio

All settings made before are one-time. However, we have brought the most difficult part behind us. It remains the pleasant part.

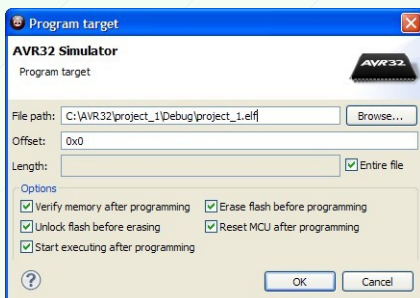
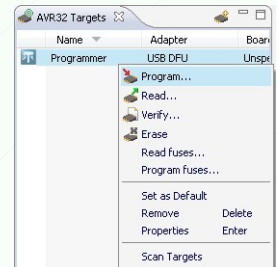
1. Start the boot loader (see chapter 5.1. *Start the Boot Loader*).
2. Save the complete project *File* → *Save All*
3. Compile the project (see chapter 4.4. *Compilation of the Project*)

If you get the following message, it means that since the last compilation nothing has been changed in the program.



```
C-Build [Project_1]
**** Build of configuration Debug for project Project_1 ****
**** Internal Builder is used for build ****
Nothing to build for Project_1
```

4. Click with the right mouse button on your preconfigured **Programmer** in the window *AVR32 Targets* and choose there *Program...*

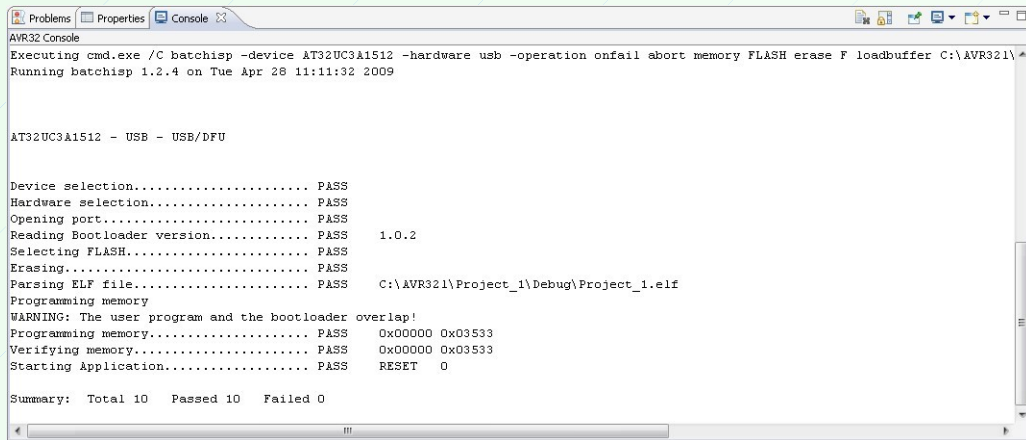


5. Enter in the field *File path*: compiled ELF-file. In our case it is the following address:  
[C:\AVR32\Project\\_1\Debug\Project\\_1.elf](C:\AVR32\Project_1\Debug\Project_1.elf)

**Important!**

Choose **all** option fields as it is shown in the left picture and click afterwards the button "OK,".

The result of the programming will be detailed shown in the window *Console* (see picture below).



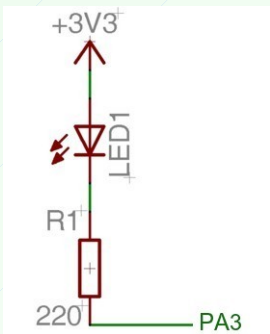
```
AVR32 Console
Executing cmd.exe /C batchisp -device AT32UC3A1512 -hardware usb -operation onfail abort memory FLASH erase F loadbuffer C:\AVR321\
Running batchisp 1.2.4 on Tue Apr 28 11:11:32 2009

AT32UC3A1512 - USB - USB/DFU

Device selection..... PASS
Hardware selection..... PASS
Opening port..... PASS
Reading Bootloader version..... PASS 1.0.2
Selecting FLASH..... PASS
Erasing..... PASS
Parsing ELF file..... PASS C:\AVR321\Project_1\Debug\Project_1.elf
Programming memory
WARNING: The user program and the bootloader overlap!
Programming memory..... PASS 0x00000 0x03533
Verifying memory..... PASS 0x00000 0x03533
Starting Application..... PASS RESET 0

Summary: Total 10 Passed 10 Failed 0
```

To make sure that the program functions properly, connect on port A pin 3 in a row resistor 220  $\Omega$  and a LED like in the lower circuit. If the LED flashes, you have made everything right.



You will find other examples for AVR32 controller series UC3 in the AVR32 UC3 Software Framework. In the folder <C:\asf-2.3.1\avr32\drivers> you will find also the drivers and source code examples of ADC, PWM, RTC, USART...

## 7. Sources

- **AVR32015: AVR32 Studio getting started**  
[http://www.atmel.com/dyn/resources/prod\\_documents/doc32086.pdf](http://www.atmel.com/dyn/resources/prod_documents/doc32086.pdf)
- **AVR32 UC3 USB DFU Bootloader**  
[http://www.atmel.com/dyn/resources/prod\\_documents/doc7745.pdf](http://www.atmel.com/dyn/resources/prod_documents/doc7745.pdf)