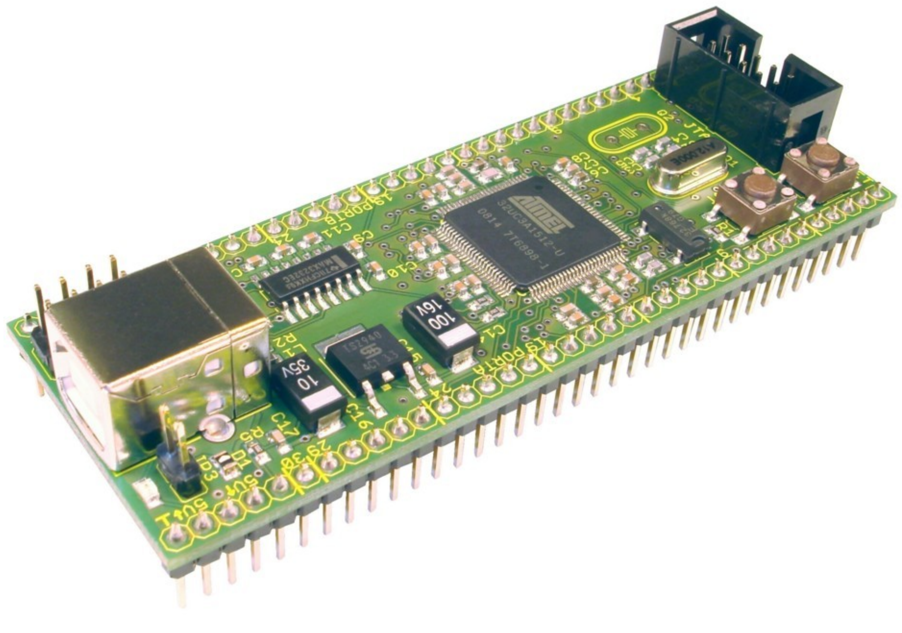
ALVIDI

First Step

From Download to the Programming



Directory

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
  - Evaluation Kit of Atmel ATEVK1100 with AT32UC3A0512 Controller **or**
  - Evaluation Kit of Atmel ATEVK1101 with AT32UC3B0256 Controller **or**
  - AVR32-Module of us AL-UC3AEB with AT32UC3A1512 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.x.x** (265 MB, revision 2.1.1, updated 2/09)
  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.

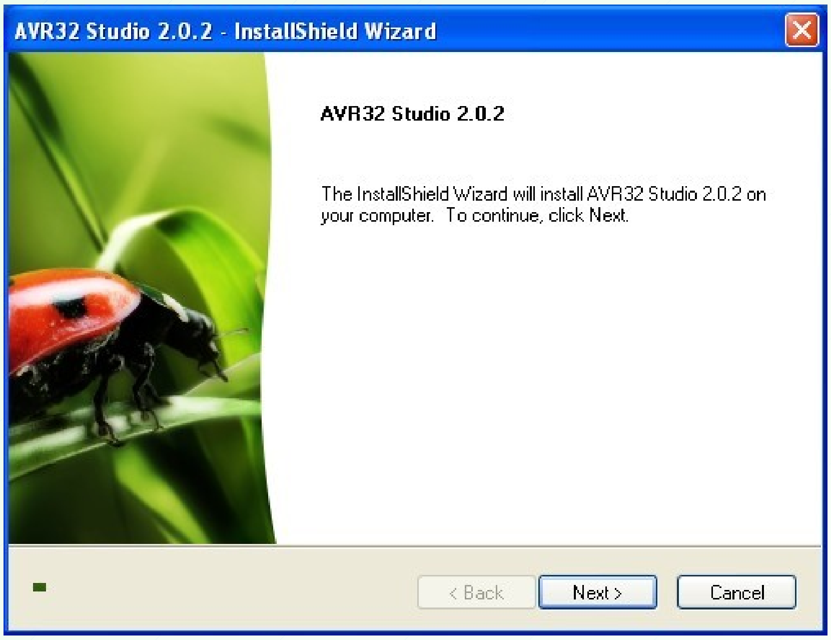
- **AVR32 GNU Toolchain** (53 MB, revision 2.1.6, updated 3/09)
  http://www.atmel.com/dyn/products/tools_card.asp?tool_id=4118
  this file contains the C libraries, flash programming tools, assembler, linker and compilers for Windows and Linux

- **AVR32 UC3 (A oder B) Software Framework**
  http://www.atmel.com/dyn/products/tools_card.asp?tool_id=4192
  this file contains numerous examples, drivers, source codes, ready projects, HTML documentation, software services...

- **FLIP 3.3.2 for Windows (Java Runtime Environement included)**
  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**


AVR32 Studio 2.0.2 - InstallShield Wizard

AVR32 Studio 2.0.2

The InstallShield Wizard will install AVR32 Studio 2.0.2 on your computer. To continue, click Next.

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

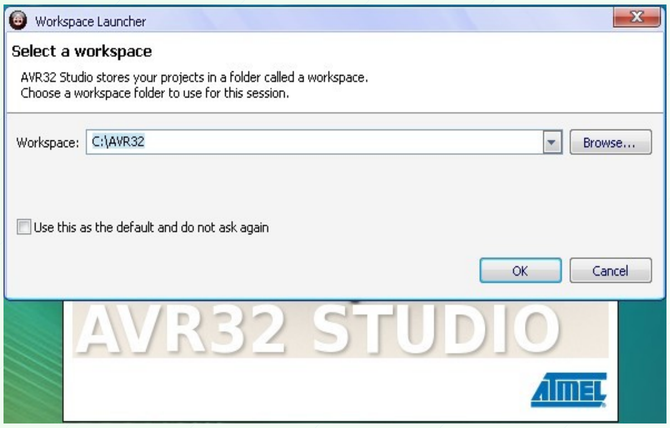
2. As a next step you have to install **avr32-gnu-toolchain-2.x.x.exe**

3. Unzip **AT32UC3A-SoftwareFramework-1.x.x.zip**, e.g to C:\

4. The last step is the installation of **JRE - Flip Installer – 3.3.2.exe**

4. AVR32 Studio

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



Workspace Launcher

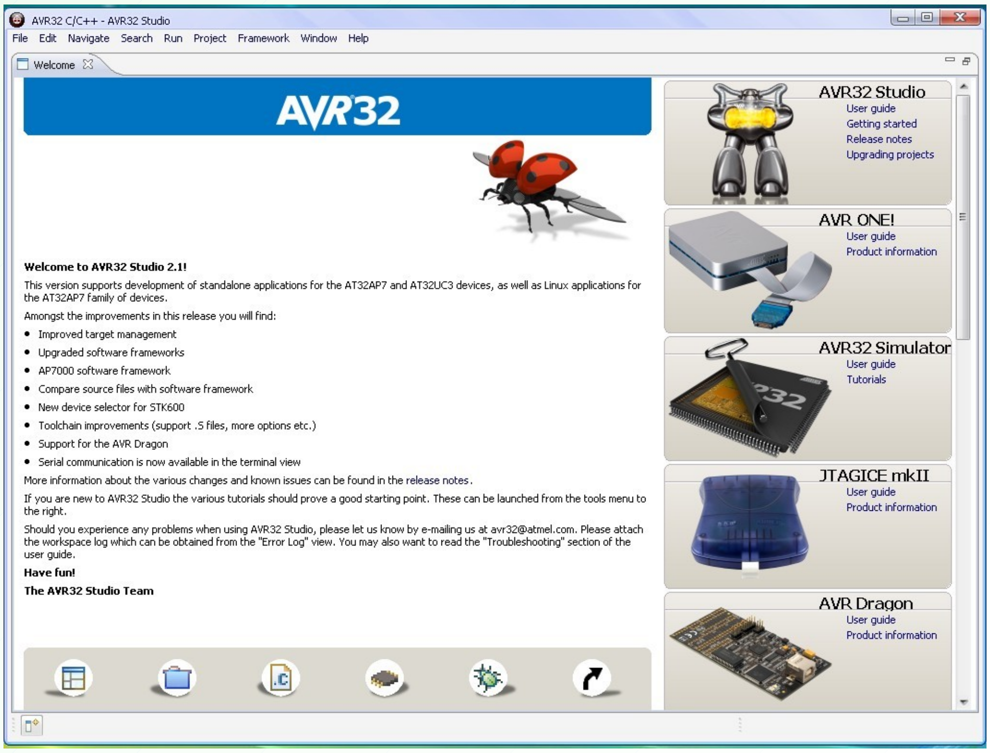
Select a workspace

AVR32 Studio stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace: C:\AVR32

☐ Use this as the default and do not ask again

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



Welcome to AVR32 Studio 2.1!

This version supports development of standalone applications for the AT32AP7 and AT32UC3 devices, as well as Linux applications for the AT32AP7 family of devices.

Amongst the improvements in this release you will find:

- Improved target management
- Upgraded software frameworks
- AP7000 software framework
- Compare source files with software framework
- New device selector for STK600
- Toolchain improvements (support .S files, more options etc.)
- Support for the AVR Dragon
- Serial communication is now available in the terminal view

More information about the various changes and known issues can be found in the release notes.

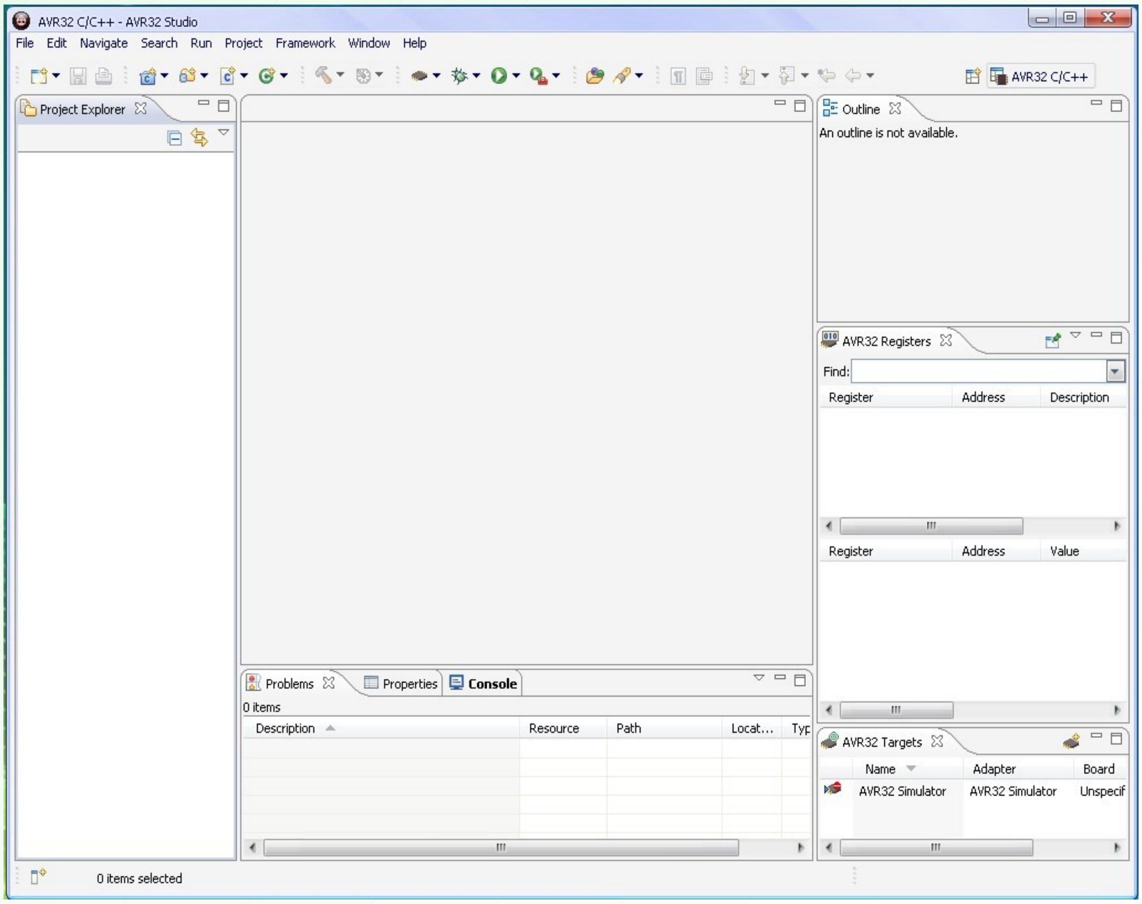
If you are new to AVR32 Studio the various tutorials should prove a good starting point. These can be launched from the tools menu to the right.

Should you experience any problems when using AVR32 Studio, please let us know by e-mailing us at avr32@atmel.com. Please attach the workspace log which can be obtained from the "Error Log" view. You may also want to read the "Troubleshooting" section of the user guide.

Have fun!

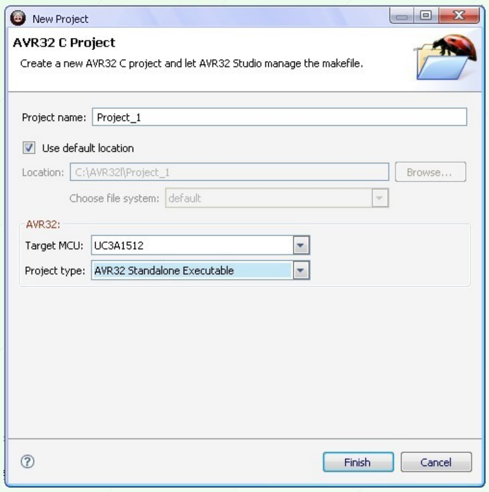
The AVR32 Studio Team

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.


AVR32 C/C++ - AVR32 Studio
File Edit Navigate Search Run Project Framework Window Help
AVR32 C/C++
Project Explorer
Outline
An outline is not available.
AVR32 Registers
Find:
Register Address Description
Register Address Value
Problems Properties Console
0 items
Description Resource Path Locat... Typ
AVR32 Targets
Name Adapter Board
AVR32 Simulator AVR32 Simulator Unspecif
0 items selected

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


New Project

AVR32 C Project

Create a new AVR32 C project and let AVR32 Studio manage the makefile.

Project name: Project_1

Use default location

Location: C:\AVR32I\Project_1

Choose file system: default

AVR32:

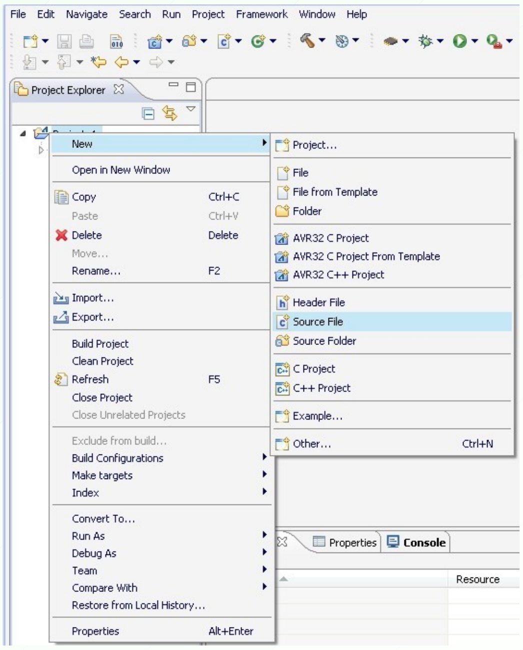
Target MCU: UC3A1512

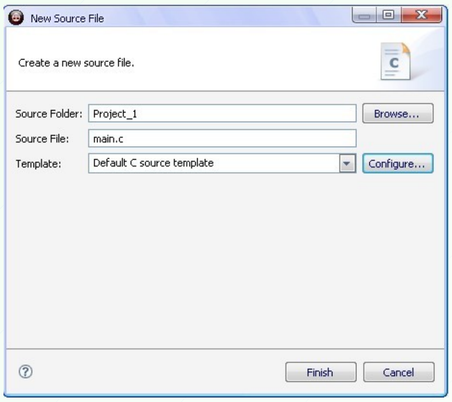
Project type: AVR32 Standalone Executable

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 MCU*, e.g. for AVR32 module **UC3A1512**, and in the field *Project type* **AVR32 Standalone 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.*




New Source File

Create a new source file.

Source Folder: Project_1

Source File: main.c

Template: Default C source template

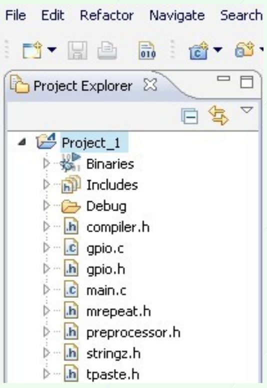
Give a name in the field *Source File*, e.g. **main.c** and click the button "Finish".

Now we will write a small program in the new created source file. In this program we change periodically the level of the pin 2 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_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
  }
}
```

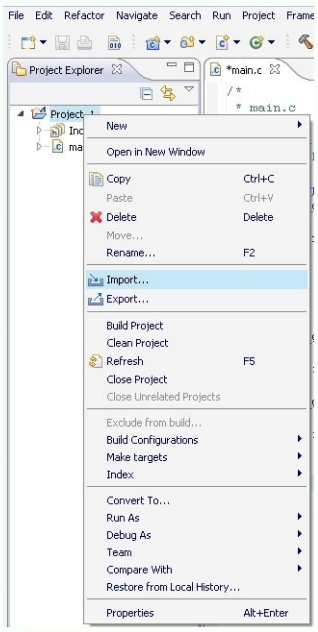
Take over the source code illustrated on top in main.c.

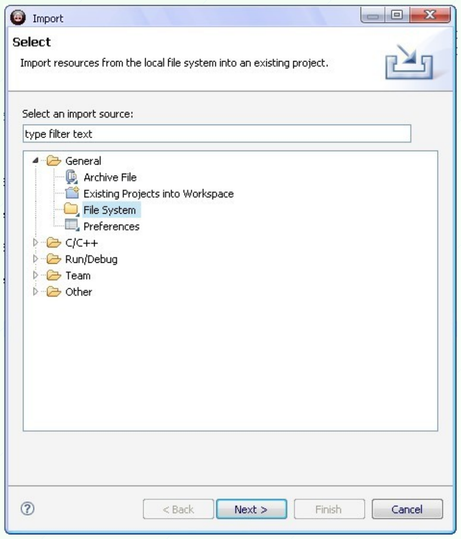
4.3. Adding the Library



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:\AT32UC3A-1.4.0

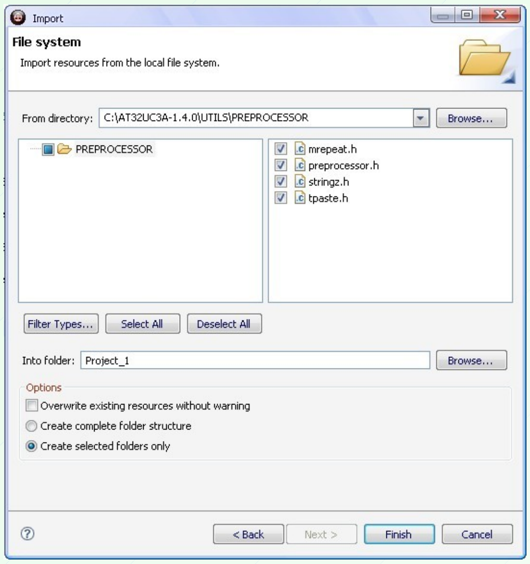
Click with the right mouse button on the project name and choose *Import...* or on the *File*➔*Import...*





Import resources from the local file system into an existing project.

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


Import

File system

Import resources from the local file system.

From directory: C:\AT32UC3A-1.4.0\UTILS\PREPROCESSOR

PREPROCESSOR

[✓] mrepeat.h
[✓] preprocessor.h
[✓] stringz.h
[✓] tpaste.h

Into folder: Project_1

Options

[ ] Overwrite existing resources without warning
○ Create complete folder structure
● Create selected folders only

In this window "Import" we can add the libraries to our project. Enter in the field *From directory:* the library folder, e.g. C:\AT32UC3A-1.4.0\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 we must open in our case three times the window "Import".

You will find the required libraries in the following folders:
- compiler.h ➔ C:\AT32UC3A-1.4.0\UTILS
- gpio.h and gpio.c ➔ C:\AT32UC3A-1.4.0\DRIVERS\GPIO
- mrepeat.h, preprocessor.h, stringz.h and tpaste.h ➔ C:\AT32UC3A-1.4.0\UTILS\PREPROCESSOR

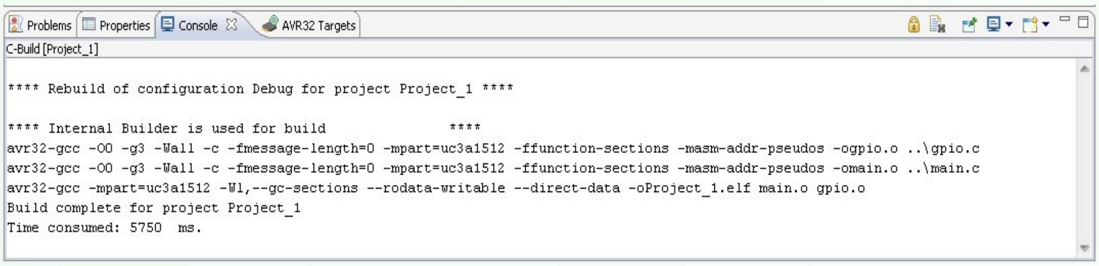
4.4. Compilation of the Project

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

Compile the project:

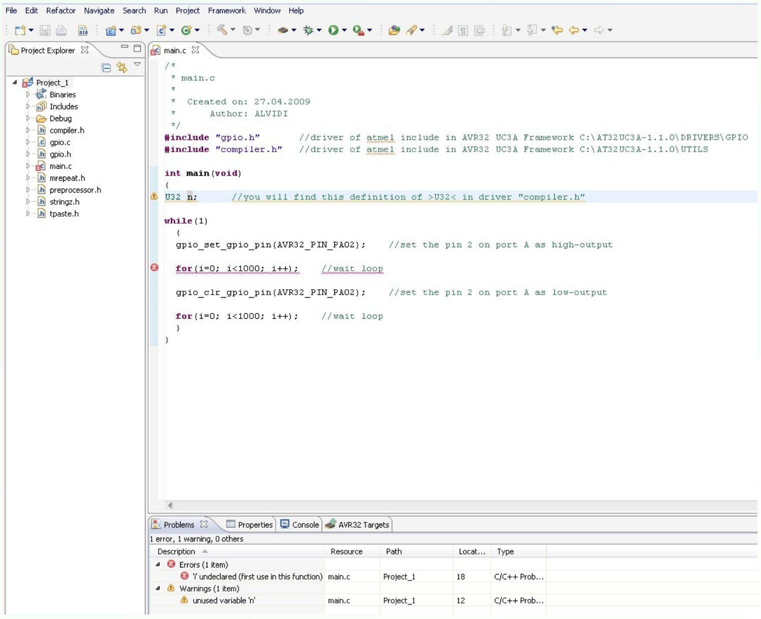
- *Project*➔*Build ALL* **or**
- key sequence [*Strg*]+[*B*] **or**
- right mouse button on your project ➔ *Build Project*

You will see the result of the compilaton 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
 *
 *  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 n;      //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
  }
}
```

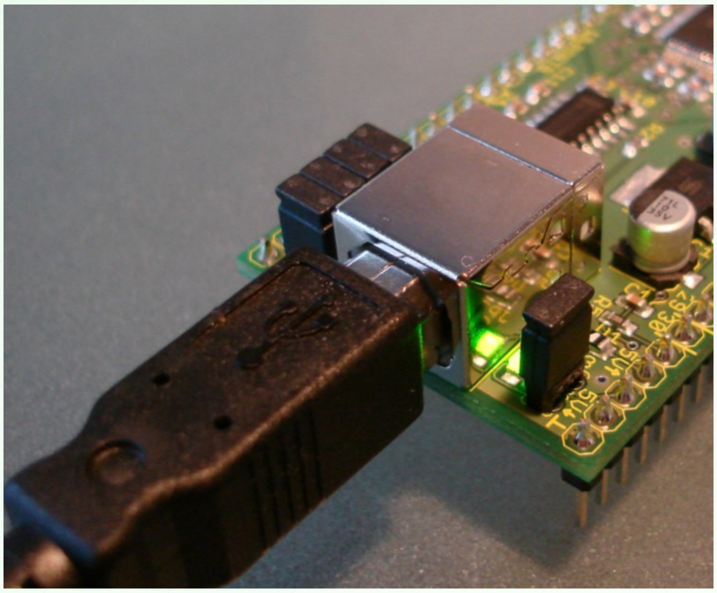
1 error, 1 warning, 0 others

Description	Resource	Path	Locat...	Type
Errors (1 item)				
'i' undeclared (first use in this function)	main.c	Project_1	18	C/C++ Prob...
Warnings (1 item)				
unused variable 'n'	main.c	Project_1	12	C/C++ Prob...

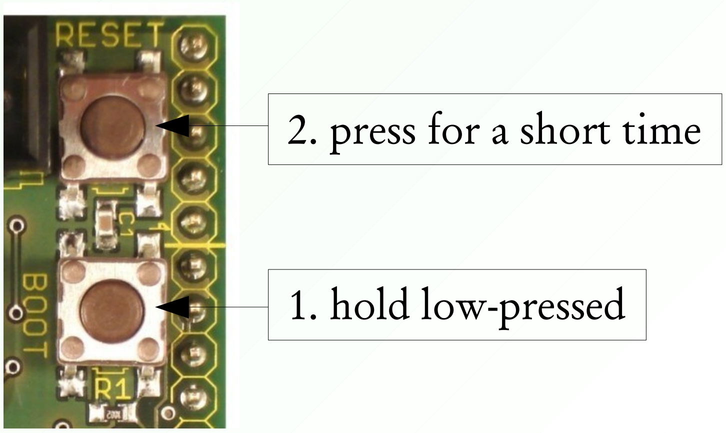
5. Hardware

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.

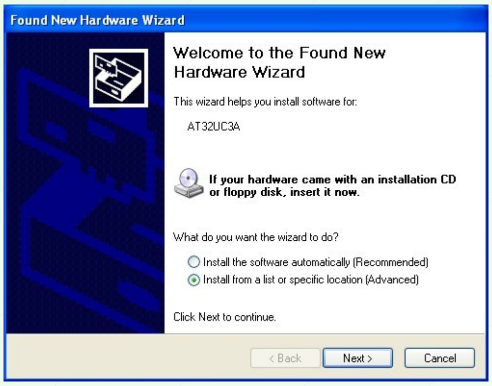


Hold the "BOOT"-key low-pressed and press for a short time the "RESET"-key. Therefore you start the boot loader.



5.2. Installation of the USB-Driver

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


Found New Hardware Wizard

Welcome to the Found New Hardware Wizard

This wizard helps you install software for:

AT32UC3A

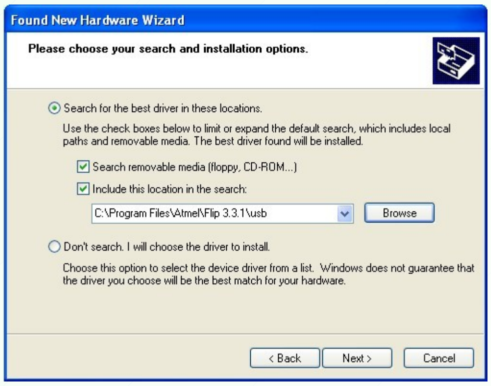
If your hardware came with an installation CD or floppy disk, insert it now.

What do you want the wizard to do?

- ○ Install the software automatically (Recommended)
- [✓ Install from a list or specific location (Advanced)]

Click Next to continue.

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


Found New Hardware Wizard

Please choose your search and installation options.

[✓ Search for the best driver in these locations.]

Use the check boxes below to limit or expand the default search, which includes local paths and removable media. The best driver found will be installed.

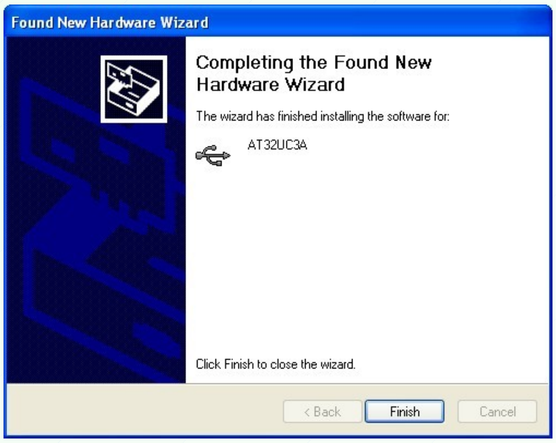
- ✓ Search removable media (floppy, CD-ROM...)
- ✓ Include this location in the search:

C:\Program Files\Atmel\Flip 3.3.1\usb

○ Don't search. I will choose the driver to install.

Choose this option to select the device driver from a list. Windows does not guarantee that the driver you choose will be the best match for your hardware.

Enter in white field the folder of the driver C:\Program Files\Atmel\Flip 3.3.1\usb and click the button „Next >“.

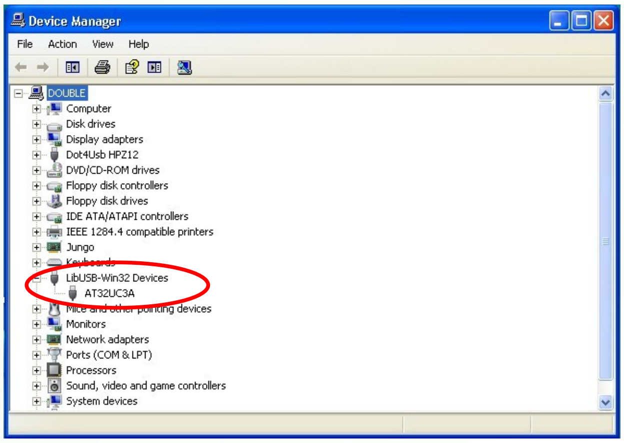

Found New Hardware Wizard

Completing the Found New Hardware Wizard

The wizard has finished installing the software for:

AT32UC3A

Click Finish to close the wizard.


Device Manager

DOUBLE
- Computer
- Disk drives
- Display adapters
- Dot4Usb HPZ12
- DVD/CD-ROM drives
- Floppy disk controllers
- Floppy disk drives
- IDE ATA/ATAPI controllers
- IEEE 1284.4 compatible printers
- Jungo
- Keyboards
- LibUSB-Win32 Devices
  - AT32UC3A
- Mice and other pointing devices
- Monitors
- Network adapters
- Ports (COM & LPT)
- Processors
- Sound, video and game controllers
- System devices

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.S ➔ C:\1.3.0-AT32UC3A\SERVICES\USB\CLASS\DFU\EXAMPLES\ISP\BOOT
- conf_isp.h ➔ C:\1.3.0-AT32UC3A\SERVICES\USB\CLASS\DFU\EXAMPLES\ISP\CONF

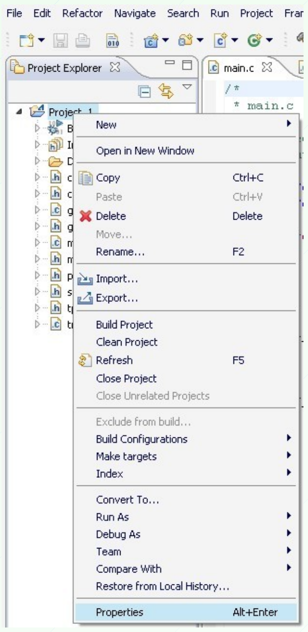
To include library to the project, see chapter *4.3 Adding the Library*.

Open the Assembler-File *trampoline.S* and change the line

```
#include "../CONF/conf_isp.h"
```

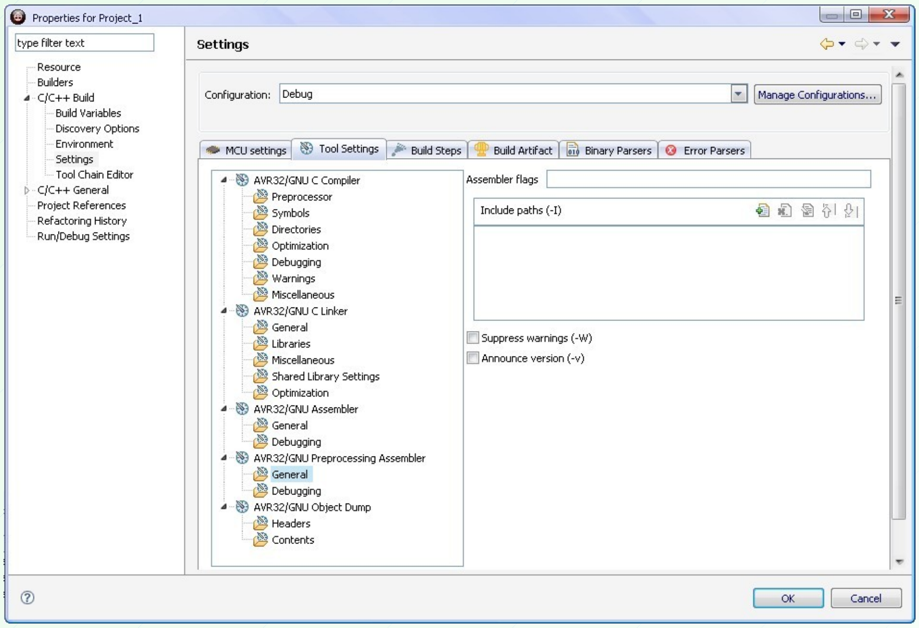
to

```
#include "conf_isp.h"
```



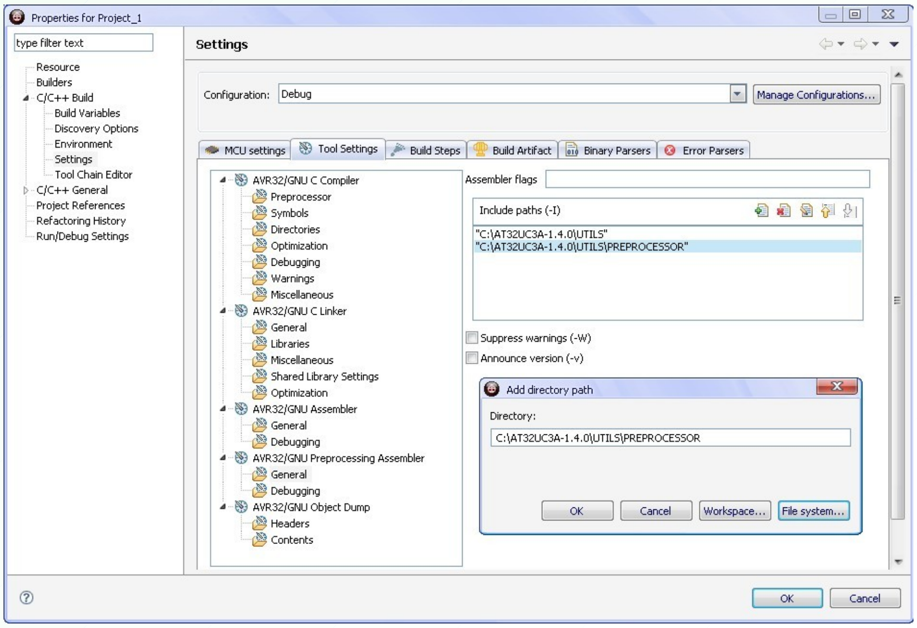
In the next step we have to include two paths to our project.
Click with the right mouse button on the project name and choose *Properties* or on *File➔Properties*

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



Configuration: Debug

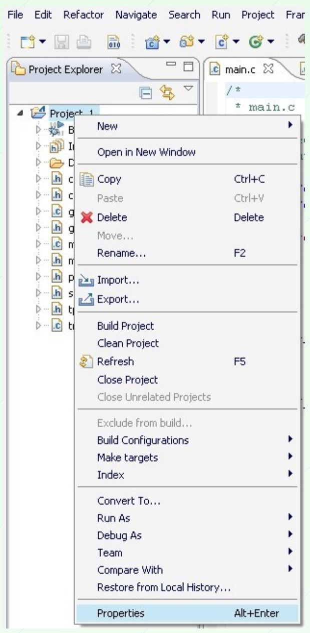
In the right part of this window we are seeing the field *Include paths (-I)*. In this field we have to include two folders. Click the white leaf with green cross and include one by one following two floders:
[C:\AT32UCA-1.4.0\UTILS] und
[C:\AT32UCA-1.4.0\UTILS\PREPROCESSOR]



Configuration: Debug

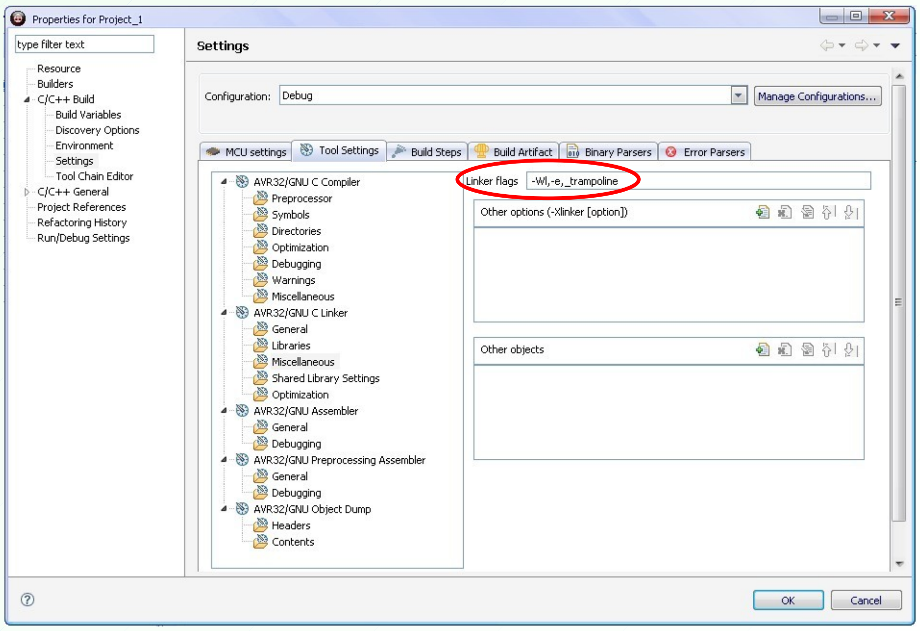
Include paths (-I):
"C:\AT32UC3A-1.4.0\UTILS"
"C:\AT32UC3A-1.4.0\UTILS\PREPROCESSOR"

Directory: C:\AT32UC3A-1.4.0\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,".


Properties for Project_1

Settings

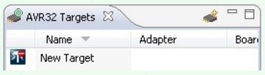
Configuration: Debug

Linker flags: -Wl,-e,_trampoline

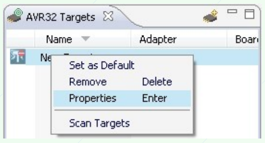
6.2. Installation of the Programmer

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

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

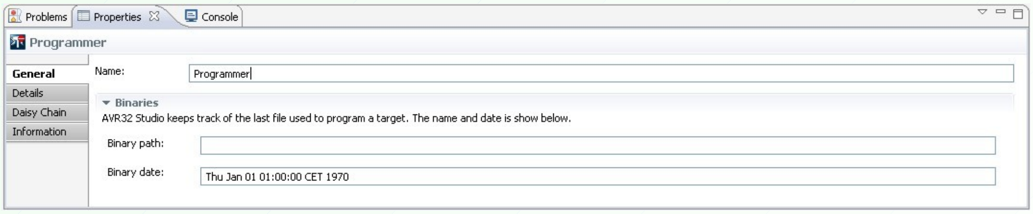


Click on the symbol "Creates a new target" in the gray field. Afterwards appears in the white field *New Target*.



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.



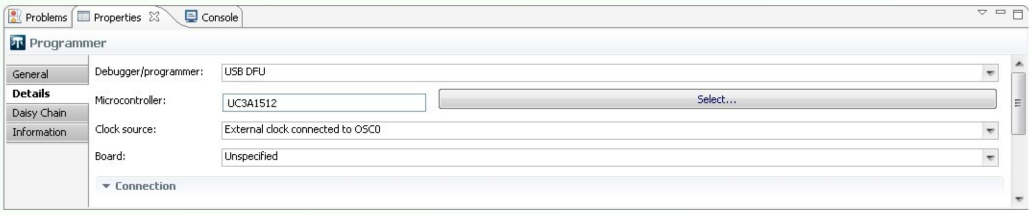
Name: Programmer

AVR32 Studio keeps track of the last file used to program a target. The name and date is show below.

Binary path:

Binary date: Thu Jan 01 01:00:00 CET 1970

In the window *Properties*➔ *Details* will be set the following hardware parameters:
* - ***Debugger/programmer***: USB DFU
* - ***Microcontroller***: UC3A1512
* - ***Clock source***: Internal RC oscillator <u>**or**</u> External clock connected to OSC0



Debugger/programmer: USB DFU

Microcontroller: UC3A1512

Clock source: External clock connected to OSC0

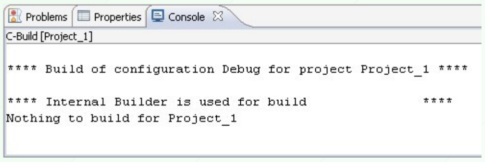
Board: Unspecified

6.2. 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.

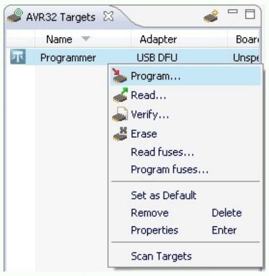
1. Start the boot loader (see chapter *5.1. Start the Boot Loader*).
2. Save the complete project *File*➔ *Save All*
3. Compile the project (see chapter *4.4. Compilation of the Project*)

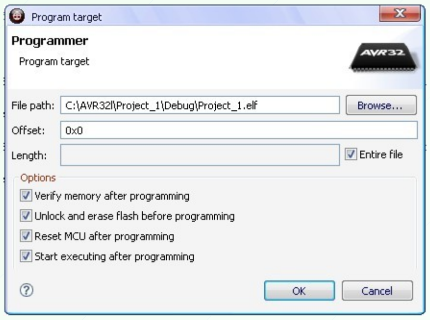
If you get the following message, it means that since the last compilation nothing has been changed in the program.



```
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...*



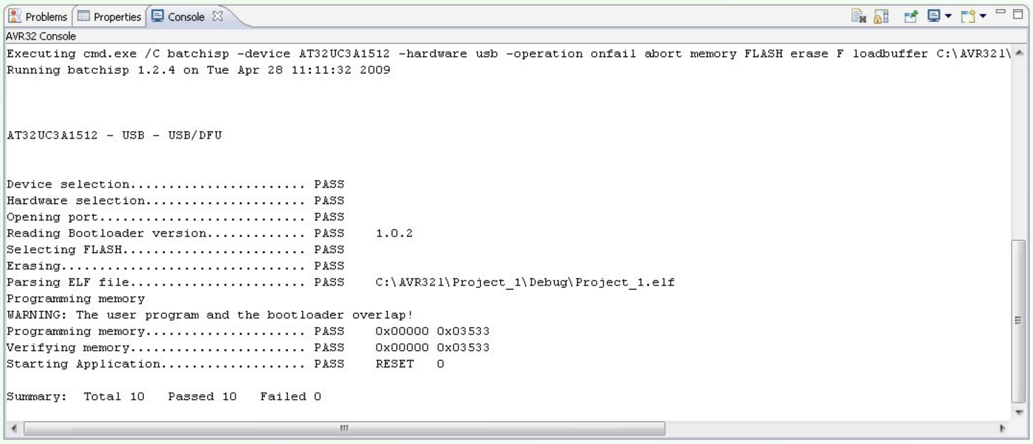


Program target
Programmer
Program target
File path: C:\AVR32l\Project_1\Debug\Project_1.elf
Offset: 0x0
Length: [✓] Entire file
Options
[✓] Verify memory after programming
[✓] Unlock and erase flash before programming
[✓] Reset MCU after programming
[✓] Start executing after programming

5. Enter in the field *File path:* compiled ELF-file. In our case it is the following address: C:\AVR32_Projects\Project_1\Debug\Project_1.elf

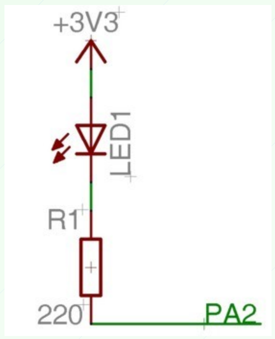
Important!
Choose **all** option fields as it is shown in the left picture and click afterwards the button "OK,".

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



```
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 2 in a row resistor 220 Ω 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:\AT32UC3A-1.4.0\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

- **AVR32 UC3 USB DFU Bootloader**
  http://www.atmel.com/dyn/resources/prod_documents/doc7745.pdf

- **AT32UC3A Series: AT32UC3A0512, AT32UC3A0256, AT32UC3A0128, AT32UC3A1512, AT32UC3A1256, AT32UC3A1128 Preliminary (803 pages, revision C, updated 10/07)**
  http://www.atmel.com/dyn/products/datasheets.asp?family_id=682