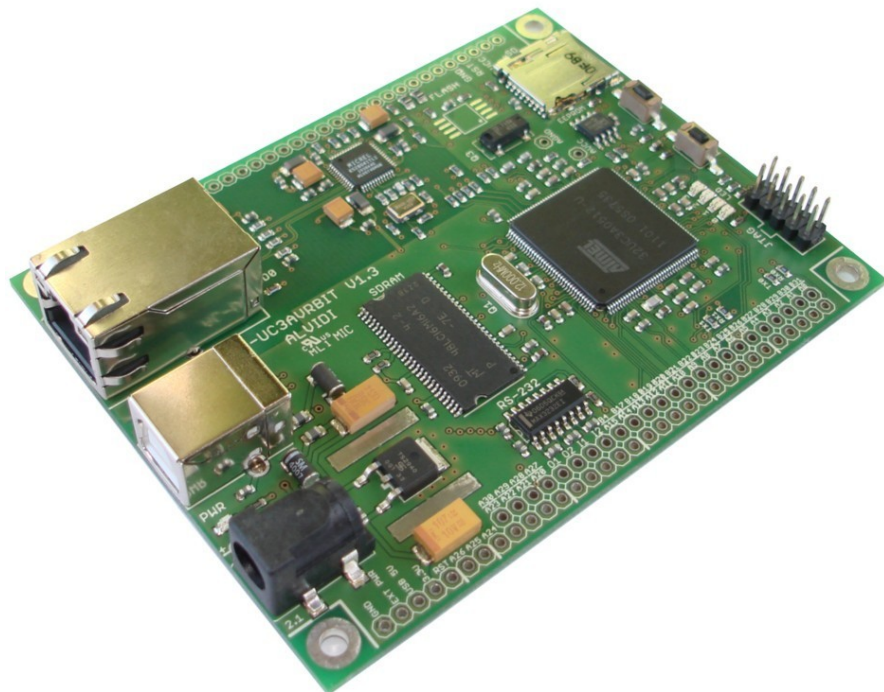


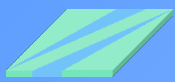
**ALVIDI**

# **Erster Schritt**

***Vom Herunterladen bis  
zum Programmieren***

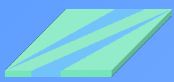
**Revision vom 27.07.2012**





## Verzeichnis

|           |                               |           |
|-----------|-------------------------------|-----------|
| <b>1.</b> | <b>Einleitung</b>             | <b>3</b>  |
| <b>2.</b> | <b>Herunterladen</b>          | <b>4</b>  |
| <b>3.</b> | <b>Installation</b>           | <b>5</b>  |
| <b>4.</b> | <b>Atmel Studio 6</b>         | <b>6</b>  |
| 4.1.      | Neues Projekt                 | 7         |
| 4.2.      | Hinzufügen der Bibliotheken   | 8         |
| 4.3.      | Compilieren des Projekts      | 11        |
| <b>5.</b> | <b>Hardware</b>               | <b>12</b> |
| 5.1.      | Starten des Bootloaders       | 12        |
| 5.2.      | Installieren des USB-Treibers | 13        |
| <b>6.</b> | <b>Programmieren</b>          | <b>14</b> |
| 6.1.      | Erweiterung des Programms     | 14        |
| 6.2.      | Installation des Programmers  | 15        |
| 6.3.      | Controller Programmieren      | 16        |
| <b>7.</b> | <b>Quellen</b>                | <b>17</b> |



## 1. Einleitung

Die ähnliche Information, die Sie hier sehen, finden Sie auch auf der Webseite von Atmel. Wir stellen Ihnen mit Hilfe unseres Produktes, AVR32 Board, unsere Erfahrungen mit AVR32 in diesem Dokument vor.

Dieses Dokument ermöglicht Ihnen einen schnellen Einstieg in die AVR32-Controller-Familie. Es führt Sie vom Herunterladen kostenloser Software und Tools von Atmel bis zum Programmieren des Controllers.

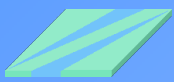
Es ist bekannt, dass der erste Schritt sehr schwer ist. Wir haben diesen Schritt gemacht und präsentieren ihm in diesem Dokument.

Um sich die Bekanntschaft mit AVR32 zu machen, benötigen Sie folgendes:

- Software
  - Atmel Studio 6.0 (build 1843) Installer - Full
  - FLIP 3.4.5 oder höher
- Hardware
  - AVR32-Board AL-UC3AVRBIT mit AT32UC3A0512 Controller oder
  - ein anderer AVR32 Produkt
- JTAG-Programmer
  - AVR JTAGICE MKII oder
  - AVR Dragon
- USB-Programmer
  - USB-Bootloader (auf jedem Controller UC3-Serie vorinstalliert)

Zum Programmieren stehen Ihnen zwei Möglichkeiten JTAG & USB Programmer zur Verfügung. Als kosten günstigen externe Programmer empfehlen wir AVR Dragon. Vorteil diesen JTAG Programmer ist die Debug-Funktion. Als Zweite steht der kostenlose USB-Bootloader zur Wahl. Wie der Name schon verrät, benötigt er nur eine USB-Verbindung mit dem Computer.

Jeder Controller der UC3-Serie ist mit dem USB-Bootloader vorprogrammiert, deswegen können alle drei oben genannten Hardware mit Hilfe eines USB-Kabels programmiert werden.



## 2. Herunterladen

Das komplette Software finden Sie auf der Webseite von Atmel.

- **Atmel Studio 6.0 (build 1843) Installer - Full**

<http://www.atmel.com/tools/ATMELSTUDIO.aspx>

dieses File enthält die Entwicklungsumgebung für AVR32 - Controller. Vorerst müssen Sie sich registrieren. Nachdem Sie alle (\*) Felder ausgefüllt haben, erscheint das Paket zum Runterladen.

- **Atmel Software Framework 3.3.0**

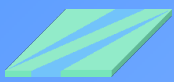
<http://www.atmel.com/tools/AVRSOFTWAREFRAMEWORK.aspx>

dieses File enthält zahlreiche Beispiele, Treiber, Quellcodes, fertige Projekte, HTML-Dokumentation, Softwareservices ...

- **FLIP 3.4.5 for Windows (Java Runtime Environement 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) unterstützt In-System Programmierung von Flash Geräten durch RS232, USB oder CAN. Hier befindet sich unter anderen der Treiber für USB-Bootloader.



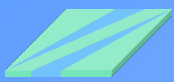
## 3. Installation

1. Installieren Sie als Erstes **as6installer-6.0.1843.exe**



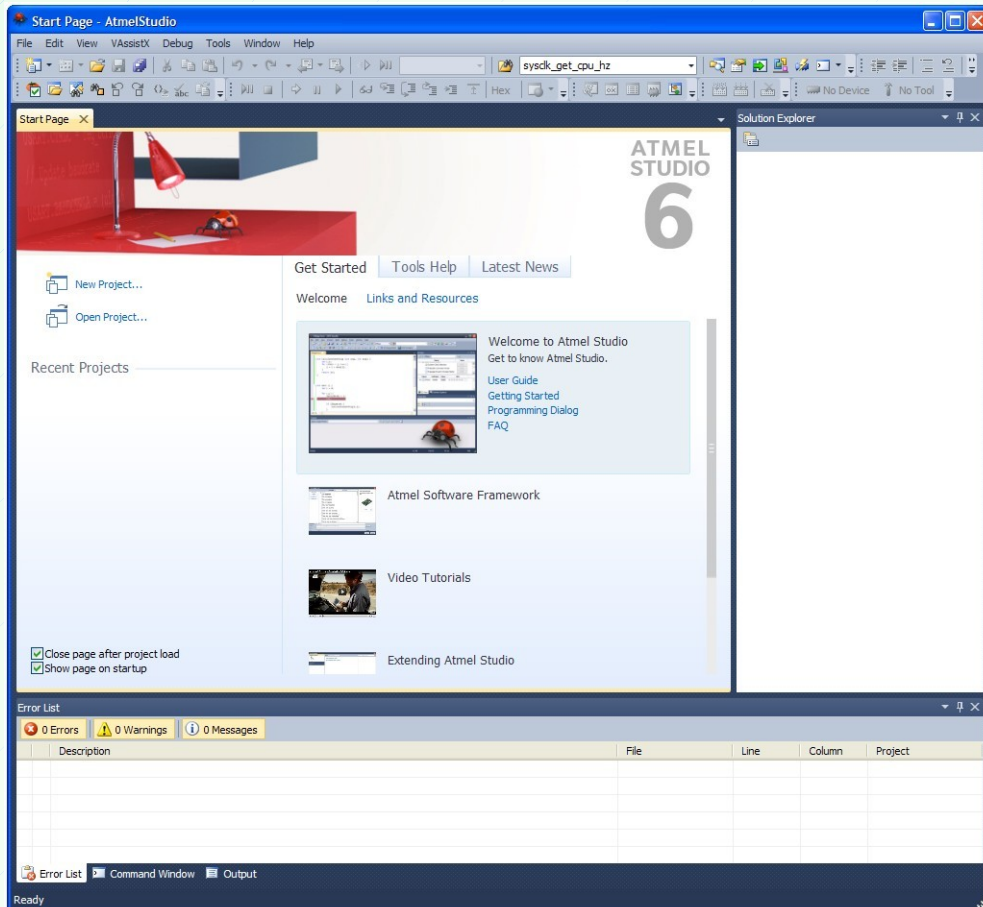
Folgen Sie den Anweisungen im Fenster und installieren Sie Atmel Studio 6. Die Installation wird einige Minuten in Anspruch nehmen. Bitte achten Sie darauf, dass Ihre Firewall die komplette Installation zulässt.

2. Entpacken Sie **asf-standalone-archive-3.3.0.zip**, z.B. auf Partition C:\
3. Installieren Sie **Flip Installer - 3.4.5.106.exe**

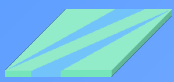


## 4. Atmel Studio 6

Starten Sie Atmel Studio 6 *Start* → *Programmen* → *Atmel* → *Atmel Studio 6.0*

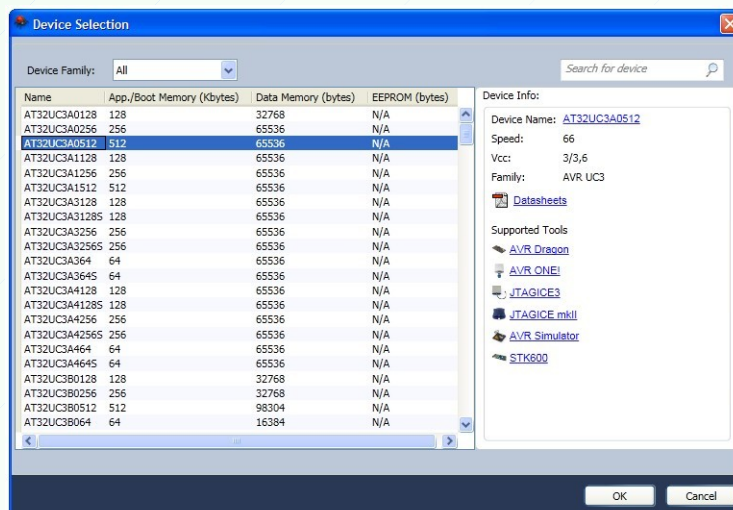
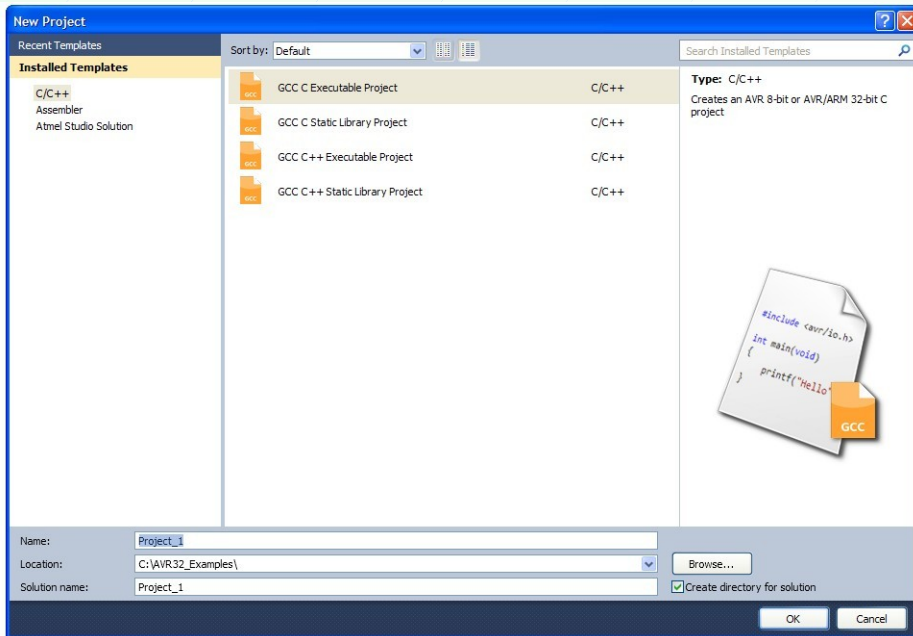


Sie begrüßt jetzt das Fenster „Start Page“. Hier können Sie mehr Informationen über die Software, Hardware und Programmer bekommen.

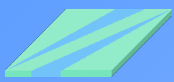


## 4.1. Neues Projekt

Starten Sie ein neues Projekt *File* → *New* → *Project...* Im mittleren Fenster wählen Sie „GCC C Executable Project“. Geben Sie einen Namen dem Projekt im Feld *Name*, z.B. **Project\_1** und als Speicherort im Feld *Location* z.B. **C:\AVR32\_Examples\** . Klicken Sie anschließend auf die Schaltfläche „OK“. Im nächsten Schritt wählen Sie zutreffenden Controller. In unserem Fall ist es AT32UC3A0512. Mit klicken auf „OK“ schließt sich dieses Fenster und öffnet sich ein leeres Fenster „Project\_1.c“ mit einer Startmaske auf.







Schreiben wir ein kleines Programm. In diesem Programm verändern wir zeitweise den Ausgangszustand des Pins 12 vom Port X. Bei unserem AVR32 Board ist dieses Pin mit LED0 verbunden.

```
#include <avr32/io.h>
#include "gpio.h"
#include "sysclk.h"
#include "delay.h"

int main(void)
{
    sysclk_init();
    delay_init(sysclk_get_cpu_hz());

    while(1)
    {
        gpio_tgl_gpio_pin(AVR32_PIN_PX12);
        delay_ms(500);
    }
}
```

Übernehmen Sie den oben abgebildeten Quellcode in *Project\_1.c*

## 4.2. Hinzufügen der Bibliotheken

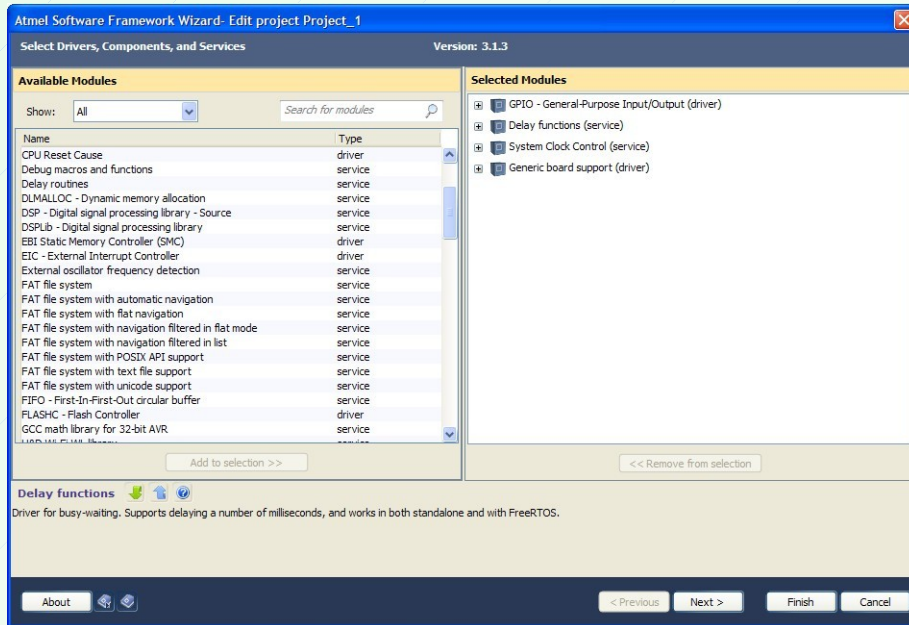
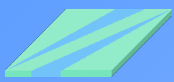
Nachdem wir das Programm geschrieben haben, müssen wir noch ein Paar Bibliotheken zu unserem Projekt hinzufügen. Diese Bibliotheken findet man in *Project* → *ASF Wizard*

Wählen Sie im Linken Feld folgende Bibliotheken aus:

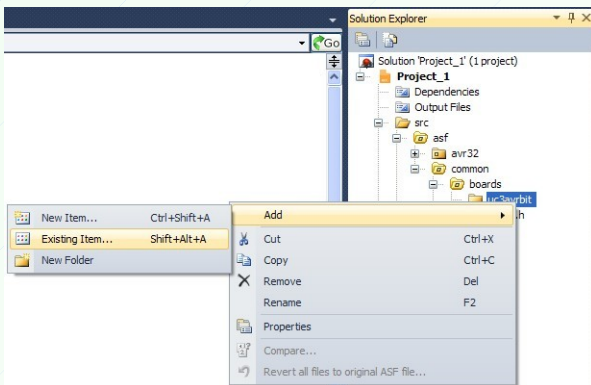
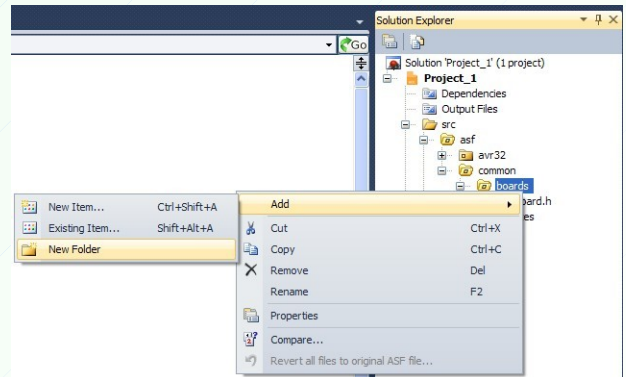
- GPIO – General-Purpose Input/Output
- Generic board support
- System Clock Control
- Delay functions

Mit der Schaltfläche „Add to selection >>“ übertragen Sie den obigen Auswahl in den rechten Fenster. Mit der Schaltfläche „Finish“ importieren Sie die ausgewählten Bibliotheken in das Projekt.





Erstellen Sie den Ordner uc3avrbit im Fenster **Solution Explorer** unter **Project\_1** → **src\as\common\boards**. Indem Sie mit der rechten Maustaste auf Ordner **boards** → **Add** → **New Folder** klicken.



Mit der rechten Maustaste auf Ordner uc3a-vrbit → **Add** → **Existing Item...** fügen Sie folgende File hinzu:

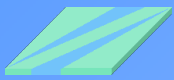
[http://alvidi.de/data\\_sheets/AL\\_LIB\\_UC3.zip](http://alvidi.de/data_sheets/AL_LIB_UC3.zip) →

AL\_LIB\_UC3 → UC3AVRBIT:

[uc3avrbit.h](#)

[led\\_func.c](#)

[led\\_func.h](#)



In der Bibliothek **board.h** unter *src\asf\common\boards* ändern Sie die Zeilen:

```
#define USER_BOARD          99  //!< User-reserved board (if any).
#define DUMMY_BOARD         100  //!< Dummy board to support board-independent
```

wie folgt:

```
#define USER_BOARD          99  //!< User-reserved board (if any).
#define DUMMY_BOARD         100  //!< Dummy board to support board-independent
#define UC3AVRBIT           101  //!< ALVIDI AL-UC3AVRBIT board with AT32UC3A0512
```

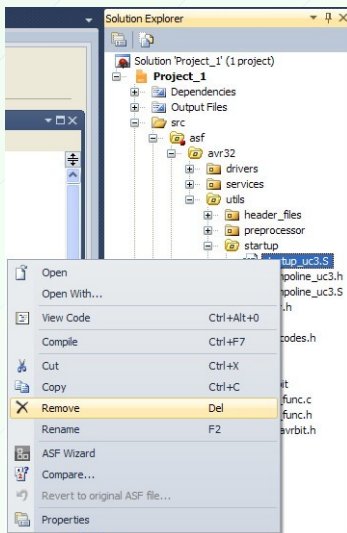
```
#define BOARD                UC3AVRBIT
```

und

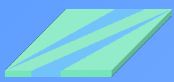
```
#if BOARD == EVK1100
#include "evk1100/evk1100.h"
```

durch

```
#if BOARD == UC3AVRBIT
#include "uc3avrbit/uc3avrbit.h"
#elif BOARD == EVK1100
#include "evk1100/evk1100.h"
```



Jetzt steht uns die letzte Änderung bevor. Wir müssen die Bibliothek **startup\_uc3.S** unter *src\asf\avr32\utils\startup* löschen.



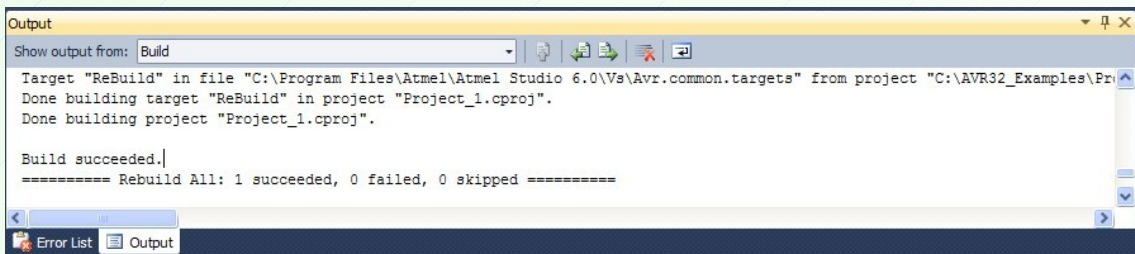
## 4.3. Compilieren des Projekts

Als erstes muss das gesamte Projekt gespeichert werden, *File* → *Save ALL*.

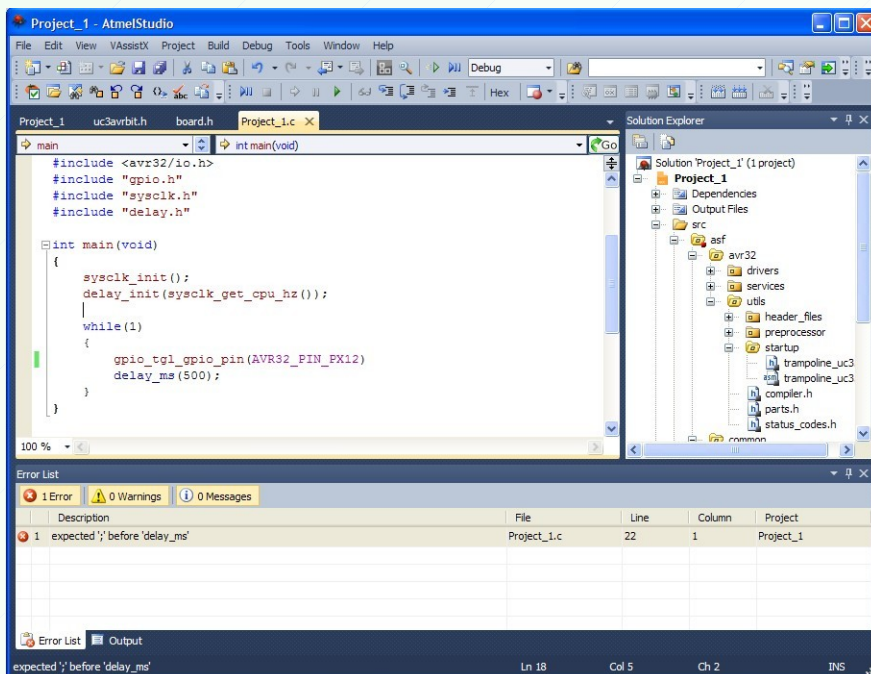
Compilieren Sie das Projekt:

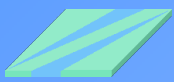
- *Build* → *Rebuild Solution* oder
- Tastenkombination **[Strg]+[Alt]+[F7]** oder
- rechte Maustaste auf **Solution 'Project\_1'** → *Rebuild Solution*

Das Ergebnis des Compilierens werden Sie im Fenster „Output“ sehen, wie es in unserem Abbild dargestellt ist.



Falls während des Compilierens Fehler im Programm gefunden werden, dann erscheinen sie im Fenster „Error List“. Die untere Abbildung stellt diesen Fall vor.

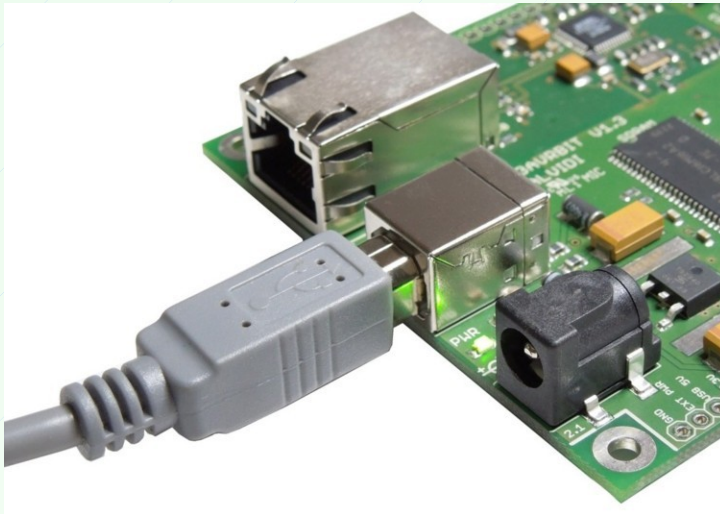




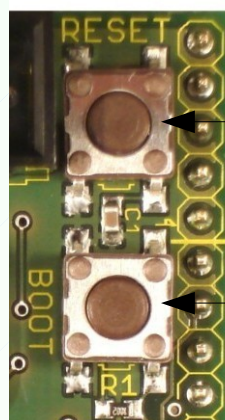
## 5. Hardware

### 5.1. Starten des Bootloaders

Stellen Sie eine USB-Verbindung zwischen Ihrem Computer und dem Hardware Gerät, z.B. AVR32-Board. Sobald die USB mit dem Board verbunden, leuchtet die grüne *Power LED*.

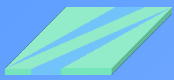


Halten Sie die „BOOT“-Taste gedrückt und drücken Sie kurzzeitig die „RESET“-Taste. Somit starten Sie den Bootloader



2. kurzzeitig drücken

1. gedrückt halten



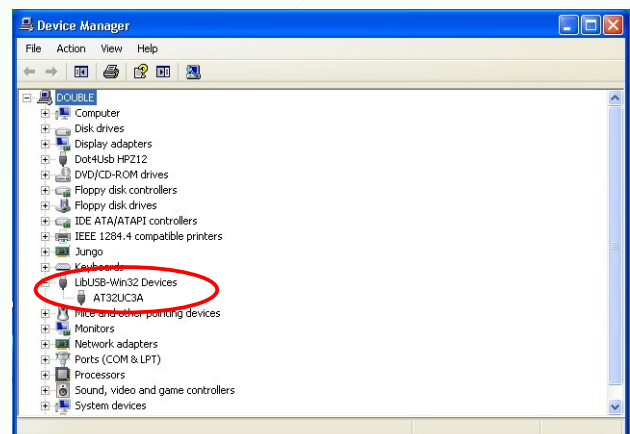
## 5.2. Installieren des USB-Treibers

Bei den bestehender USB-Verbindung und gestartetem Bootloader erscheint die untere Abbildung.



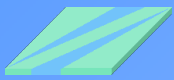
Wählen Sie *Install from a list or location (Advanced)* und klicken Sie auf die Schaltfläche „Next >“.

Geben Sie in weißem Feld den Zielordner des Treibers [C:\Program Files\Atmel\FliP 3.4.5\usb](#) ein und klicken Sie auf die Schaltfläche „Next >“.



Nach einer erfolgreichen Installation erscheint die linke obere Abbildung. Im rechten Abbild *Device Manager* wird das angeschlossene Gerät unter **LibUSB-Win32 Devices**→**AT32UC3A** sichtbar.

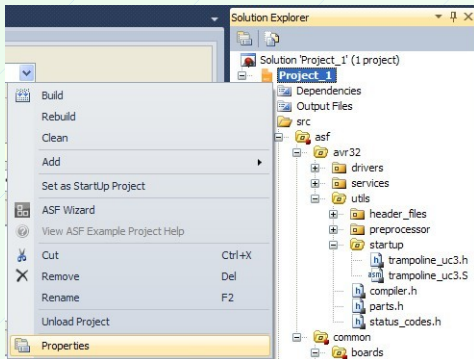




## 6. Programmieren

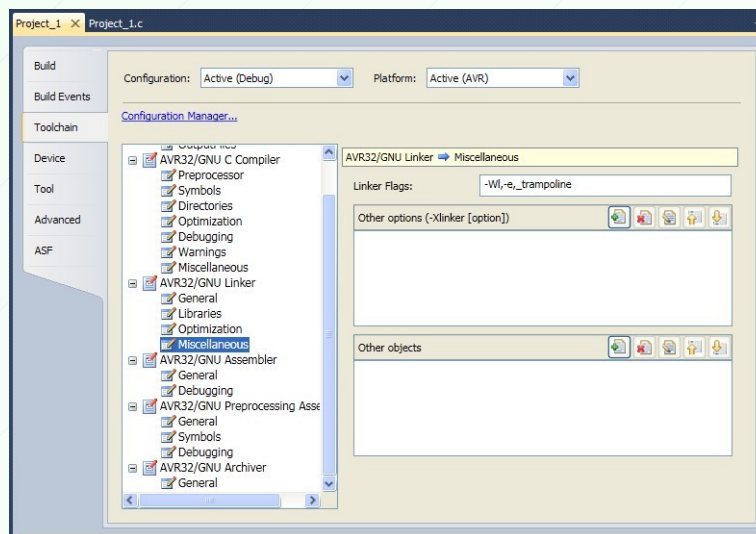
### 6.1. Erweiterung des Programms

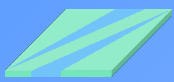
Zum Programmieren mit USB-Bootloader müssen wir die Eigenschaften unseres Project\_1 mit einer Zeile erweitern.



Klicken Sie mit der rechten Maustaste im Fenster *Solution Explorer* auf **Project\_1** und wählen Sie die Option **Properties**.

Im Feld **Linker Flags** unter *Project\_1* → *Toolchain* → *AVR32/GNU Linker* → *Miscellaneous* tragen Sie die Zeile `-Wl,-e,_trampoline` ein.





## 6.2. Installation des Programmers

Wie man mit dem Bootloader programmiert wird im pdf-File *AVR32 UC3 USB DFU Bootloader* von Atmel erklärt. Siehe: <http://www.atmel.com/Images/doc32166.pdf>

Aus diesem Dokument ist ersichtlich, dass man zum Programmieren Windows Programm „Eingabeaufforderung“ benötigt. Dieses Programm findet man unter *Start → Zubehör → Eingabeaufforderung*.

Mit einem Befehl z.B.

```
cmd.exe /C batchisp -device AT32UC3A0512 -hardware usb -operation onfail abort  
memory FLASH erase F
```

löscht man den gesamten Speicher außer Bootloaderbereich und mit dieser Kommandozeile

```
cmd.exe /C batchisp -device AT32UC3A0512 -hardware usb -operation onfail abort  
memory FLASH erase F loadbuffer
```

```
C:\AVR32_Examples\Project_1\Project_1\Debug\Project_1.hex program verify  
start reset 0
```

wird den gesamten Speicher außer Bootloaderbereich gelöscht, mit dem File *Project\_1.hex* beschrieben, verglichen und durch internen Reset das Programm gestartet.

D.h. um das Programm mit Hilfe von USB Bootloader in den Controller einzuspielen, müssen wir jedes Mal im Fenster *Eingabeaufforderung* den obigen Befehl neu einfügen. Wir können die Sache deutlich vereinfachen in dem wir eine selbst ausführende Datei erstellen. Dazu benötigen wir ein „Editor“. Dieses Programm findet man ebenfalls unter *Start → Zubehör → Eingabeaufforderung*. In weiteren Schritten wird es erklärt, wie man ein solches File zusammenstellt.

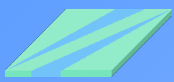
1. Schritt: öffnen Sie „Editor“ und fügen Sie diese Zeile ein:

```
cmd.exe /C batchisp -device AT32UC3A0512 -hardware usb -operation onfail abort  
memory FLASH erase F loadbuffer
```

```
C:\AVR32_Examples\Project_1\Project_1\Debug\Project_1.hex program verify  
start reset 0
```

2. Schritt: speichern Sie diesen File z.B. unter *C:\AVR32\_Examples\Project\_1* **nicht** als txt-Datei **sondern** als bat-Datei. Als Bezeichnung nehmen wir **PROGRAM.bat**

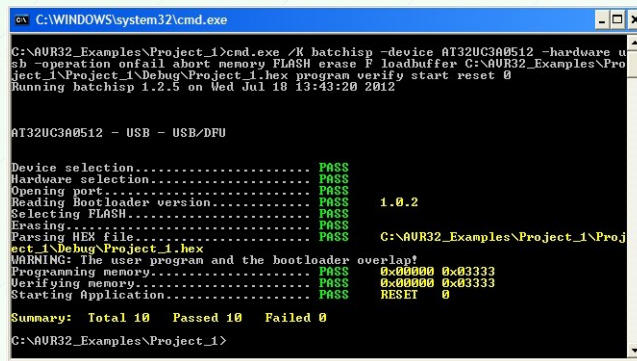




## 6.3. Controller Programmieren

Um AVR32 Board zu programmieren, muss zu erst Bootloader gestartet werden. Siehe Kapitel 5.1. *Starten des Bootloaders*. In 2 Schritten aus letztem Kapitel haben wir ein bat-File erstellt. **PROGRAM.bat** wird jedes Mal aufgerufen, wenn wir ein Update einspielen. Für ein anderes Projekt muss nur den Link (C:\AVR32\_Examples\Project\_1\Project\_1\Debug\Project\_1.hex) im Schritt 1 angepasst werden.

Das Fenster schließt sich nach dem Ausführen der Befehle zu. Damit sich das Fenster nach dem Programmieren geöffnet bleibt, müssen wir **cmd.exe /C** durch **cmd.exe /K** ersetzen. Das Ergebnis des Programmieren wird im Fenster *Eingabeaufforderung* detailliert dargestellt (siehe untere Abbildung) .

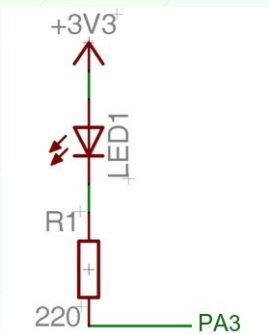


```
C:\WINDOWS\system32\cmd.exe
C:\AVR32_Examples\Project_1>cmd.exe /K batchisp -device AT32UC3A0512 -hardware u...
Running batchisp 1.2.5 on Wed Jul 18 13:43:20 2012

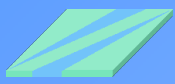
AT32UC3A0512 - USB - USB/DFU

Device selection..... PASS
Hardware selection..... PASS
Opening port..... PASS
Reading bootloader version..... PASS 1.0.2
Selecting FLASH..... PASS
Erasing..... PASS C:\AVR32_Examples\Project_1\Proj
Erasing HEX file..... PASS
WARNING: The user program and the bootloader overlap!
Programming memory..... PASS 0x000000 0x033333
Verifying memory..... PASS 0x000000 0x033333
Starting Application..... PASS RESET 0
Summary: Total 10 Passed 10 Failed 0
C:\AVR32_Examples\Project_1>
```

Jetzt müssen wir uns vergewissern, dass das Programm richtig funktioniert. Falls keine LED auf dem Board vorhanden ist, schließen Sie an Port X Pin 12 in einer Reihe Widerstand 220  $\Omega$  und ein LED, wie im unteren Schaltplan. Wenn die LED blinkt, dann haben Sie alles richtig gemacht.



Die weiteren Beispiele für AVR32 Controller Serie UC3 finden Sie im AVR Software Framework. Im Ordner [C:\asf-3.3.0\avr32\drivers](#) befinden sich die Treiber und Quellcodebeispiele für ADC, PWM, RTC, USART, ...



## 7. Quellen

- **AVR32015: Atmel Studio 5 Video Beispiele**  
[http://www.atmel.com/microsite/avr\\_studio\\_5/](http://www.atmel.com/microsite/avr_studio_5/)
- **AVR32 UC3 USB DFU Bootloader**  
<http://www.atmel.com/Images/doc32166.pdf>