AVR-Programmierung unter Mac OSX

im Studiengang BEL3 Lehrveranstaltung Embedded Systems

Tutorial

ausgeführt von: Jürgen Hausladen A-2460 Bruck/Leitha, Obere Neugasse 6

Wien 01.02.2011



Inhaltsverzeichnis

1	Einleitung	3
2	Grundlegendes	3
3	CrossPack for AVR Development	3
4	Eclipse	3
4.	1 Installation	3
4.	2 Konfiguration	4
4.	3 Konfiguration eines AVR-Programmers (optional)	5
5	dfu-programmer	8
5.	1 USB-Bibliothek	8
5.	2 dfu-programmer installieren	8
6	Erstellen eines AVR-Projektes	9
6.	1 Projektkonfiguration	9
6.	2 Anlegen der C-Source Datei	12
6.	3 Auswahl des Programmers (optional)	
6.	4 Kompilieren	15
6.	5 Programmieren	15
6.	6 Flashen	15
Ab	bildungsverzeichnis	17

1 Einleitung

Für den Abschluss des 3. Semesters an der FH-Technikum Wien, war im Fach Embedded Systems ein Mikrocontroller-Projekt zu realisieren. Hierfür wurde der Mikrocontroller AT90USB162 verwendet. Da jedoch die Entwicklungswerkzeuge von ATMEL nur bedingt für andere Plattformen als Microsoft Windows verfügbar sind, bzw. nahezu keine Anleitungen für die Installation bzw. Konfiguration vorhanden sind, habe ich mir zum Ziel gesetzt, diese unter Mac OSX nativ zu betreiben. Ein weiteres Ziel war, die Verwendung der Entwicklungsumgebung Eclipse für die Programmierung, da diese plattformunabhängig erhältlich ist, und zahlreiche Features bzw. Erweiterungen besitzt. Das komplette Tutorial wurde unter Mac OSX 10.6 Snow Leopard getestet.

2 Grundlegendes

Zu Beginn muss, um unter Mac OSX überhaupt Code erzeugen zu können XCode installiert sein. Das Installationspaket hierfür befindet sich auf der "Mac OSX Install DVD" unter "Optionale Installationspakete".

3 CrossPack for AVR Development

"CrossPack for AVR Development" ist vergleichbar mit dem Programm WinAVR unter Windows. Es beinhaltet die komplette GNU Compiler Suite, C-Bibliotheken für die AVR-Programmierung, AVRDUDE zum Programmieren des Controllers und noch viele weitere Features. Es ist mit diesem Package auch das Debuggen und Simulieren möglich. Das Installationspaket hierfür ist unter "http://www.obdev.at/products/crosspack/download.html" erhältlich.

4 Eclipse

4.1 Installation

Die Entwicklungsumgebung "Eclipse" ist eine sehr verbreitete Entwicklungsumgebung und ist für die verschiedensten Programmiersprachen erhältlich. Der große Vorteil liegt hier darin, dass die Entwicklungsumgebung plattformunabhängig ist und man sich daher nicht jedes Mal bei einem Systemwechsel eine andere Entwicklungsumgebung anlernen muss. Eclipse ist unter "http://www.eclipse.org/downloads/" erhältlich. Die benötigte Version trägt den Namen "Eclipse IDE for C/C++ Developers". Hierbei ist zu beachten, dass bei Verwendung von Mac OSX 10.6 (Snow Leopard) die 64-Bit Version verwendet werden sollte, da Mac OSX 10.6 ein 64-Bit Betriebssystem ist und daher nicht jedes Mal beim Start die 32-Bit Bibliotheken geladen werden müssen. Das eben geladene Archiv kann dann anschließend in einen Ordner entpackt werden und über Drag & Drop in den Programmordner gezogen werden. Ausgeführt wird das Programm über die Eclipse App im Hauptverzeichnis.



Abb.1: Eclipse Programmordner

Für Entwickler, welche Eclipse auch für die Java-Programmierung einsetzen, empfiehlt es sich beide Java Apps getrennt zu installieren, um auch beide Apps gleichzeitig ausführen zu können.

4.2 Konfiguration

Da nun Eclipse fertig installiert ist müssen nun noch die entsprechenden Plugins für die AVR-Programmierung installiert werden. Dafür wird Eclipse gestartet und der Plugin-Manager über das Menü "Help" und "Install New Software" aufgerufen.



Abb.2: Eclipse Plugin-Manager

Anschließend daran wird in das Feld "Work with:" die URL zum AVR-Plugin eingegeben ("http://avr-eclipse.sourceforge.net/updatesite/"). Nun muss nur noch der Plugin in dem darunterliegenden Feld ausgewählt werden und über die Schaltfläche "Next" installiert werden.

$\Theta \bigcirc \Theta$	Install
Available Software Check the items that you wish to install.	
Work with: Http://avr-eclipse.sourceforge.net/update	nd more software by working with the <u>"Available Software Sites"</u> preferences.
type filter text	
Name	Version
VIII CDT Optional Features	
AVR Eclipse Plugin	2.3.4.20100807PRD
Select All Deselect All 1 item selecte	ed
Details	
Plugin for the development of applications for the Atme	el AVR series of embedded microprocessors.
	More
Show only the latest versions of available software	Hide items that are already installed
Sroup items by category	What is <u>already installed</u> ?
Contact all update sites during install to find required	l software
?	< Back Next > Cancel Finish

Abb.3: Eclipse AVR-Plugin Installation

Falls Eclipse nach der Installation neu starten möchte, so sollte dies auch zugelassen werden, um die korrekte Funktion des Plugins zu gewährleisten.

4.3 Konfiguration eines AVR-Programmers (optional)

Da der e-Stick nicht über einen entsprechenden Programmer programmiert wird, sondern über einen Bootloader geflasht wird, ist dieser Schritt nicht zwingend erforderlich.

Um einen AVR-Programmer für die Programmierung des Mikrocontrollers einzurichten, muss dieser im Plugin "AVRDude" definiert sein. Dieser Plugin ist unter "Eclipse", "Einstellungen", "AVR", "AVRDude" zu erreichen.





0 0	Preferences	
type filter text 🛛 🕄	AVRDude	(⇒ ∓ ⇒ - ▼
►General ▼AVR	AVRDude Global Settings	
AVRDude	Log internal AVRDude output to console	
Paths ►C/C++ ►Help	Use custom configuration file for AVRDude	
▶Install/Update ▶Run/Debug	AVRDude config file	Browse
► Tasks	Programmer configurations	
► Usage Data Collector	Configuration Description	(bbb
	Configuration usb Default AVRDude Programmer Config	
		Edit
		Remove
	Restore Defaults	Apply
?	Cancel	ОК

Abb.5: AVRDude

Um die entsprechende Hardware hinzuzufügen, muss zunächst der Button "Add" gedrückt werden, wodurch eine große Auswahl an Programmern aufgelistet wird.

0 0	Edit AVRDude Programmer	Configuration New Configuration			
Configuration name	New Configuration				
Description	Default AVRDude Programmer Configuration. Modi	fy as required for your setup.			
Programmer Hardware (-c)					
Arduino Arduino Arduino Arduino Arduino Armei AppNote AVR109 Boot Loader Armei AppNote AVR119 Boot Loader Armei AVR Dragon in HVSP mode Armei AVR Dragon in JSP mode Armei AVR Dragon in JSP mode Armei AVR Dragon in PD mode Armei AVR Dragon in PD mode Armei AVR ISP Armei AVR ISP Armei AVR ISP Atmei AVR ISP mode Armei JTAG ICE mkil Armei JTAG ICE mkil in debugWire mode Armei JTAG ICE mkil in isP mode Armei JTAG ICE mkil in isP mode Armei JTAG ICE mkil in isP mode Armei JTAG ICE mkil in mede		Programmer details from [/usr/local/CrossPack-AVR-20100115/etc/avrdude.conf:373] Id = "stk500v2"; desc = "Atmel STK500 Version 2.x firmware"; type = stk500v2;			
Override default port (-P)					
Override default baudrate (-b)					
State of Parallel Port lines after A	VRDude exit				
/Reset Line		Vcc Lines			
• restore to previous state		• restore to previous state			
 activated (-E reset) 		O activated (-E vcc)			
deactivated (-E noreset)		O deactivated (-E novcc)			
Delay between avrdude invocations Command line preview	milliseconds avrdude -cstk500v2 [part specific options]				
(?		Cancel OK			

Abb.6: AVR-Programmer

Hier muss nun nur noch der richtige Programmer ausgewählt, eventuell die Baud-Rate, der Port des Programmers bzw. der Name der Konfiguration festgelegt werden. Zum Abschluss sollte noch unter "Paths" die Pfade zum Compiler überprüft werden.

0 0		Preferences	
type filter text	Paths		⇔ - ⇔ - ▼
▶General ▼AVR AVRDude Paths ▶C/C++ ▶Help	Path Settings for the AVF Disable search for system Note: If disabled, a mar	R Eclipse Plugin stem paths at startup nual rescan may be required when a new avr-gcc toolchain has been in	stalled.
▶Install/Update ▶Run/Debug ▶Tasks ▶Team ▶Usage Data Collector	Path to AVR-GCC GNU make AVR Header Files AVRDude (a) Atmel Part Description	Source Current value System /usr/local/CrossPack-AVR-20100115/bin System /usr/local/CrossPack-AVR-20100115/bin System /usr/local/CrossPack-AVR-20100115/bin System /usr/local/CrossPack-AVR-20100115/bin on Files	Edit Rescan
		(Restore Defaults)	Apply
?		Cancel	ОК

Abb.7: Paths

5 dfu-programmer

5.1 USB-Bibliothek

Damit der dfu-programmer den angeschlossenen Mikrocontroller richtig erkennt, sind zwingend die entsprechenden USB-Bibliotheken für den entsprechenden Mikrocontroller notwendig. Diese können direkt von der Seite "http://www.ellert.se/twain-sane/" geladen werden. Für eine schnelle Installation ohne jeglichen Aufwand sollte hier je nach System das binary package von "libusb" geladen werden. Erfahrene Nutzer können sich natürlich hier auch die Sourcefiles downloaden und die Bibliothek selbst kompilieren.

5.2 dfu-programmer installieren

Der dfu-programmer ist ein USB-Flash Tool für AVR Mikrocontroller, welche einen Bootloader besitzen. Dieses Programm ist vergleichbar mit dem FH-Tool oder ATMEL Flip. Dieses Tool muss als Einziges leider von Hand kompiliert werden. Dazu werden die Sourcefiles von "http://sourceforge.net/projects/dfu-programmer/" geladen, in einen beliebigen Ordner entpackt und anschließend über "Programme", "Dienstprogramme", "Terminal" konfiguriert und kompiliert. Ich empfehle hier den dfu-programmer in der Version 0.5.2 zu verwenden, da die Version 0.5.4 Probleme bei der Konfiguration unter Mac OSX 10.6 macht. Zur Konfiguration und Kompilierung wechselt man im Terminal in das entpackte Verzeichnis über den Befehl "cd Beispielverzeichnis/.../dfu-programmer-0.5.2"("..." steht für die Verzeichnisse bis zu dem entpacken dfu-programmer Verzeichnis). Anschließend wird das Skript bootstrap.sh ausgeführt. Dies geschieht über den Befehl "./bootstrap.sh".



Im nächsten Schritt konfigurieren wir das Projekt für die Kompilierung über den Befehl "./configure". Ist dies geschehen, wird das Projekt über den Befehl "make" kompiliert. Zum Installieren muss nun nur noch der Befehl "make install" ausgeführt werden. Sollte hier ein Fehler auftreten, so wurde höchstwahrscheinlich aufgrund der fehlenden administrativen Rechte, der Zugriff für die Installation verweigert. Für diesen Fall lautet der Befehl wie folgt "sudo make install". Dabei wird für die Authentifizierung als root vor der Installation nach dem Benutzer-Passwort gefragt.

Wurden all diese Schritte erfolgreich durchgeführt, so kann nun über die folgenden Befehle der Mikrocontroller über den Terminal geflasht werden:

Syntax: dfu-programmer <Mikrocontrollertyp> <Befehl>

dfu-programmer at90usb162 erase	Löscht den kompletten Mikrocontroller
dfu-programmer at90usb162 flash main.hex	Flasht das Programm "main.hex" auf
	den Mikrocontroller
dfu-programmer at90usb162 start	Startet das eben geflashte Programm

6 Erstellen eines AVR-Projektes

6.1 Projektkonfiguration

Zum Erstellen eines AVR-Projektes wird im Menü "File" der Punkt "New" und im weiteren der Punkt "Project" angeklickt.

🗯 Eclipse	File	Edit	Source	Refactor	Navigate	Search	Project	AVR	Run	Window	Help			
Image: Constraint of the second se	Nev Ope Clo Clo	v en File se se All			C第N ▶ 第₩ 合業₩	i Mal i C+ i C P i C P i Pro	kefile Proj + Project roject ject	ject wit	h Exist	ing Code	nts/wor			
i bluetoothtest ► 🥵 eStick_Blueto	Rev	Save Save A Save A vert	s 		쁐S 슙쁐S	Convert to a C/C++ Project Source Folder Folder Source File					Convert to a C/C++ Project Source Folder Folder			
	Mov Ren @ I Cor	ve ame Refresl ivert L	n ine Delim	iiters To	F2 F5	h Hea ↑ File Ca ↑ Tas	ader File from Ter ss k	nplate						
	ا 👜	Print			жP	📑 Oth	ner			Ħ	8N			
	Swi Res	tch Wo tart	orkspace		Þ		_			_				
	ès Z	lmport Export												
	Pro	perties	;		жI									

Abb.9: Projekt erstellen

Anschließend öffnet sich ein Fenster mit der Auswahl der Programmiersprache. Hier wird der Punkt "C Project" im Menü "C/C++" ausgewählt.

00	Ne	w Project		
Select a wizard				
Create a new C p	project			
Wizards:				
type filter text)
V 🗁 General				
Project				
C Proje	:t			
C++ Pr	oject			
Makefile	e Project with Existing Coo	e		
(?)	(< Back)	lext >	Cancel	(Finish
	40. Dec en en en			

Abb.10: Programmiersprachenauswahl

Ist der Eintrag ausgewählt, wird über den Button "Next" der nächste Konfigurationsschritt eingeleitet. Hier spezifiziert man unter "Project Name" den Projektnamen und wählt darunter "AVR Cross Target Application" und "Empty Project" aus.

\varTheta 🔿 🔿 СР	roject		
C Project Create C project of selected type			
Project name: Beispielprojekt Image: Beispielprojekt	space/Beispielprojekt Browse		
Project type:	Toolchains:		
V Construction AVR Cross Target Application AVR Cross Target Static Library AVR-CCC Toolchain			
Show project types and toolchains only if	they are supported on the platform		
(?) (< Back Nex	t > Cancel Finish		

Abb.11: AVR Projekt Auswahl

Durch Bestätigen mit "Next" wird wieder das nächste Menü angezeigt. Hier kann zwischen den Build-Konfigurationen gewählt werden. Da mit dem e-Stick nicht gedebuggt werden kann, wird hier nur der Punkt "Release" angewählt.

\varTheta 🔿 🕙 C Project	
Select Configurations Select platforms and configurations you wish to deploy on	
Project type: AVR Cross Target Application Toolchains: AVR-GCC Toolchain Configurations: Configuration	
■ 🗱 Debug ☑ 🛞 Release	Select all Deselect all
Use "Advanced settings" button to edit project's properties. Additional configurations can be added after project creation.	Advanced settings
Use "Manage configurations" buttons either on toolbar or on pr (?) < Back	incel Finish

Abb.12: Build- Konfiguration

Nun muss nur noch der Button "Finish" gedrückt werden um das Projekt anzulegen. Das Projekt ist jederzeit im Project Explorer einsehbar.

6.2 Anlegen der C-Source Datei

Um nun das C-Sourcefile für den späteren Programmcode anzulegen, muss über einen Rechtsklick auf das Projekt über "New" und "Source File" eine neue C Datei angelegt werden. Dieser Prozess ist bei H-Files analog.



Abb.13: C-Sourcefile anlegen

In dem nun aufgegangenen Fenster muss nun nur noch der Name der C-Datei eingegeben und mit "Finish" abgeschlossen werden.

00	New Source File	
Source File		
Create a new s	ource file.	C
Source folder:	Beispielprojekt	Browse
Source file:	Beispielprojekt.c	
Template:	Default C source template	Configure
?	Cancel	Finish

Abb.14: Source-File Konfiguration

6.3 Auswahl des Programmers (optional)

Diese Option würde benötigt werden, wenn der Mikrocontroller über einen externen Programmer programmiert werden würde. Um diesen für das Projekt einzurichten, muss dieser über AVRDude ausgewählt werden. Dazu wählt man durch einen Rechtsklick auf das Projekt den Punkt "Properties".



Abb.15: Projekteigenschaften

Anschließend daran geht ein Menü auf, wodurch der Eintrag "AVR" und weiters der Eintrag "AVRDude" gewählt werden kann. Dieser erlaubt die Auswahl des Programmers. Jedoch muss dieser vorher wie in Kapitel 4.3 beschrieben konfiguriert worden sein.

00	Properties for Beispielprojekt
type filter text	AVRDude 🗘 🗘 🗸 🚽
► Resource	
▼AVR AVRDude Target Hardware	Configuration: Release [Active]
Builders ►C/C++ Build	
C/C++ General	Programmer Flash / EEPROM Fuses Lockbits Advanced Other
Run/Debug Settings	Programmer configuration
Task Repository WikiText	Configuration usbtiny
	JTAG ICE BitClock
	Specify the bit clock period in microseconds for the JTAG interface or the ISP clock (JTAG ICE only). Set this to > 1.0 for target MCUs running with less than 4MHz on a JTAG ICE. Leave the field empty to use the preset bit clock period of the selected Programmer.
	JTAG ICE bitclock
	BitBang Programmer Bit State Change Delay
	Specify the delay in microseconds for each bit change on bitbang-type programmers. Set this when the the host system is very fast, or the target runs off a slow clock Leave the field empty to run the 15P connection at max speed.
	Bit state change delay
	- AVIDude command line previou
	avrdude -pm16 -cstk600pp
	-Uflash:w:Beispielprojekt.hex:a
	Copy Project Settings Restore Defaults Apply
?	Cancel OK

Abb.16: Programmer Auswahl

Weiters ist es anzuraten die FUSE-Bits im Tab "Fuses" zu setzen. Diese können auch über ein Icon neben dem Eintrag "direct hex values" vom Mikrocontroller ausgelesen werden. Zum Abschluss muss noch über den Unterpunkt "Target Hardware" im AVR Menüpunkt, der Mikrocontroller ausgewählt und die Clockfrequenz gesetzt werden. Diese Werte können ebenfalls auch automatisch über "Load from MCU" ausgelesen werden.

00	Properties for Beispielprojekt	
type filter text	Target Hardware	\$
Resource ♥AVR AVRDude Builders ▶C/C++ Build ♥C/C++ Build Project References Run/Debug Settings ₱Task Repository WikiText	Configuration: Release [Active]	(Manage Configurations)
	MCU Type ATmega16 CLoad from MCU	
	MCU Clock Frequency 1000000	
	Сору	Project Settings (Restore Defaults) (Apply)
?		Cancel OK

Abb.17: Target Hardware

6.4 Kompilieren

Das erstellte Projekt kann wenn alle Schritte befolgt wurden über das Hammer-Symbol kompiliert werden. Achtung: Auf die Richtige Konfiguration achten (Debug bzw. Release).



Abb. 10. Komplitere

6.5 Programmieren

Um das Projekt bei angeschlossenen Programmer auf den AVR zu laden, muss lediglich der grüne AVR Upload Button gedrückt werden. Diese Option ist beim e-Stick leider nicht verfügbar.



Abb.19: Programmieren des Mikrocontrollers

6.6 Flashen

Zum Flashen des Mikrocontrollers wird wie in Kapitel 5.2 beschrieben über "Programme", "Dienstprogramme", "Terminal" ein Terminalfenster aufgerufen, über welches mit dem Befehl "cd Beispielverzeichnis/.../Hexfileordner" zu dem Ordner navigiert wird in dem sich das erstellte Hexfile befindet. Unter Eclipse befindet sich dieses Verzeichnis standardmäßig im Hauptprojektverzeichnis und trägt den Namen "Release" bzw. "Debug". Anschließend daran wird der e-Stick über den Befehl "dfu-programmer at90usb162 erase" gelöscht, mit dem Befehl "dfu-programmer at90usb162 flash filename.hex" geflasht und das Programm über den Befehl "dfu-programmer at90usb162 start" gestartet.

```
■ Terminal — bash — 80×24
Last login: Wed Feb 2 17:43:43 on ttys000
Jurgen-Hausladens-MacBook-Pro:~ Juergen$ cd documents/workspace/eStick_Bluetooth
/Release
Jurgen-Hausladens-MacBook-Pro:Release Juergen$ dfu-programmer at90usb162 erase
Jurgen-Hausladens-MacBook-Pro:Release Juergen$ dfu-programmer at90usb162 flash e
Stick_Bluetooth.hex
Validating...
2950 bytes used (24.01%)
Jurgen-Hausladens-MacBook-Pro:Release Juergen$ dfu-programmer at90usb162 start
Jurgen-Hausladens-MacBook-Pro:Release Juergen$ dfu-programmer at90usb162 start
Jurgen-Hausladens-MacBook-Pro:Release Juergen$ dfu-programmer at90usb162 start
```

Abb.20: dfu-programmer Flashvorgang

Abbildungsverzeichnis

Abb.1:	Eclipse Programmordner4
Abb.2:	Eclipse Plugin-Manager4
Abb.3:	Eclipse AVR-Plugin Installation5
Abb.4:	Einstellungen6
Abb.5:	AVRDude
Abb.6:	AVR-Programmer
Abb.7:	Paths7
Abb.8:	dfu-programmer konfigurieren & installieren8
Abb.9:	Projekt erstellen9
Abb.10:	Programmiersprachenauswahl10
Abb.11:	AVR Projekt Auswahl10
Abb.12:	Build- Konfiguration11
Abb.13:	C-Sourcefile anlegen12
Abb.14:	Source-File Konfiguration12
Abb.15:	Projekteigenschaften13
Abb.16:	Programmer Auswahl14
Abb.17:	Target Hardware
Abb.18:	Kompilieren
Abb.19:	Programmieren des Mikrocontrollers15
Abb.20:	dfu-programmer Flashvorgang16