Das BASTLI AVR-Demoboard
Anleitung und Dokumentation

Mario Mauerer

25. Mai 2009

Inhaltsverzeichnis

1 Übersicht **3**
1.1 Die einzelnen Funktionen im Detail . . . 5
1.1.1 Taster . . . 5
1.1.2 DIP-Schalter . . . 5
1.1.3 7-Segment Anzeige . . . 5
1.1.4 Temperatursensor (NTC-Widerstand) . . . 6
1.1.5 IR-Decoder . . . 6
1.1.6 Serielle Schnittstelle . . . 7
1.1.7 Spannungsversorgung . . . 7
1.1.8 Reset-Beschaltung . . . 8
1.1.9 In-System-Programming-Anschluss . . . 8
1.1.10 Jumper . . . 8
1.1.11 Stiftleiste für externe Versorgung . . . 8
1.2 Spezifikationen . . . 9

2 Aufbauanleitung **10**
2.1 Teileliste . . . 10
2.2 Aufbau des AVR-Demoboards . . . 11
2.3 Aufbauhinweise . . . 11
2.3.1 Elektrolytkondensatoren . . . 11
2.3.2 Gleichrichter . . . 12
2.3.3 Leuchtdioden . . . 12
2.3.4 Siebensegment-Anzeige . . . 12
2.3.5 Widerstandsnetzwerk . . . 13
2.3.6 Diode am Spannungsregler . . . 13
2.3.7 ICs . . . 13
2.3.8 Drahtbrücken . . . 13
2.4 Inbetriebnahme . . . 13

3 Programmierung **14**
3.1 Allgemeines zur Programmierung . . . 14
3.2 Programmierung - HowTo unter Windows . . . 14
3.2.1 Installation der Programme . . . 14
3.2.2 Einstellungen der Fuses . . . 16
3.2.3 Das erste Programm . . . 16
3.3 Pinbelegung des AVR . . . 17

4 Linksammlung **18**

1 Übersicht

Das Bastli AVR-Demoboard haben wir dazu entworfen, Anfängern die Möglichkeiten eines Atmel Mikrocontrollers (AVR ATMega32) zu zeigen.
Natürlich kann es auch einfach als ,,Development-Board" verwendet werden, da nahezu alle IO-Pins des AVR mittels Stiftleisten kontaktierbar sind und so gute Erweiterungsmöglichkeiten bietet.

Mit der sich auf dem Board befindenden Hardware lassen sich sehr viele Features des AVR ausreizen, was dem Anfänger viele Erkundungsmöglichkeiten bietet. Ausserdem haben wir Wert darauf gelegt, dass das Board leicht für den Anfänger selbst aufzubauen ist, damit man rasch und unkompliziert zu ersten Ergebnissen beim Mikrocontroller-Einstieg gelangt.
Diese Anleitung beschreibt den Aufbau und die Inbetriebnahme des AVR-Demoboardes. Falls Fragen oder Probleme auftauchen, kann der Bastli unter www.bastli.ethz.ch oder direkt während der Öffnungszeiten im Shop kontaktiert werden.

Das Board besitzt folgende Features:

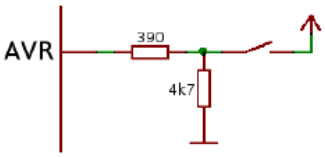
- AVR ATmega32:
  - 16MHz Taktfrequenz per Quarz oder interner Oszillator
  - 32kiB Flash
  - 1kiB EEPROM
  - 2kiB SRAM
  - 2 8-Bit-Counter
  - 1 16-Bit-Counter
  - 4 Hardware-PWM-Kanäle
  - 8-Kanal A/D-Wandler (10Bit Auflösung)
  - UART
  - Watchdog
  - etc.
- 10-Pin ISP-Anschluss zum Programmieren des AVR
- 4 Taster
- 8-Bit DIP-Schalter
- 7-Segment-Anzeige
- 5 LEDs
- Temperatursensor (NTC-Widerstand)
- IR-Decoder (TSOP1736)
- Serielle Schnittstelle, RS232 kompatibel
- On-Board-Spannungsregelung

1.1 Die einzelnen Funktionen im Detail

1.1.1 Taster

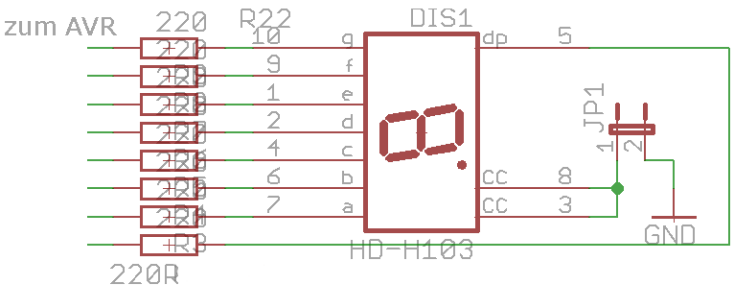
Die Taster sind in negativer Logik an PortA 0-3 angeschlossen. Das heisst, dass sie bei Betätigung den Pin des AVR auf GND ziehen. Um klare Pegel zu erhalten, müssen die internen PullUp-Widerstände des AVR eingeschaltet sein, da auf dem Board keine Pullup-Widerstände vorhanden sind.
Bei den Tastern ist etwas Vorsicht geboten. Anders als beim DIP-Switch sind hier keine Schutzwiderstände vorhanden. Würde z.B. der entsprechende Pin des AVR als Ausgang konfiguriert und auf High gesetzt, käme es beim Drücken des Tasters zum Kurzschluss!

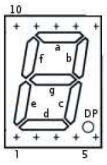
1.1.2 DIP-Schalter



Der 8Bit DIP-Schalter ist in positiver Logik an PortB angeschlossen. Ist der Schiebeschalter also in der ,,ON"-Position, wird der AVR-Pin auf High/5V gezogen. Die Die internen PullUp-Widerstände des AVR müssen **aus**geschaltet sein. Hier sind nämlich externe PullDown-Widerstände vorhanden (Widerstandsnetzwerk, je 4.7k Ohm) Die 390 Ohm Widerstände verhindern, dass zu hohe Ströme fliessen, falls der AVR-Pin als Output geschaltet ist und auf GND liegt.

1.1.3 7-Segment Anzeige



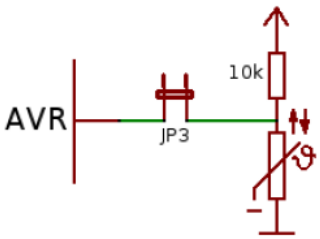


PORTCx	LED
0	dp
1	a
2	b
3	c
4	d
5	e
6	f
7	g

Die 7-Segment Anzeige ist am PortC angeschlossen. Sie wird in positiver Logik angesteuert.
Der PortC wird auch für das JTAG Interface gebraucht. Wenn sich die 7-Segment Anzeige merkwürdig verhält, liegt das wahrscheinlich daran, dass das JTAG Interface aktiviert ist. Dieses kann durch eine Fuse ausgeschalten werden.

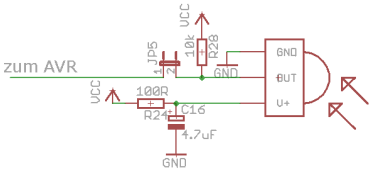
Über den LEDs fallen 1.8V ab. Bei einem Vorwiderstand von 220Ω fliessen somit pro Diode $\frac{5V-1.8V}{220\Omega} = 14.5mA$. Wenn alle LEDs leuchten verbraucht die Anzeige bereits 116mA. Dies muss beim Anschliessen eigener Hardware beachtet werden, da der Vcc Pin des AVR nur 200mA liefern kann.

1.1.4 Temperatursensor (NTC-Widerstand)



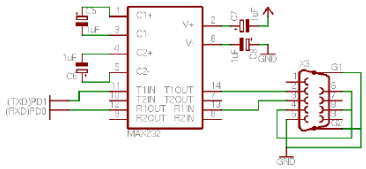
Der NTC hat einen Nennwert von 15k Ohm und ist über einen Spannungsteiler an den A/D-Wandler des AVR angeschlossen. Da der A/D-Wandler die Spannungen gegen GND misst, ist der NTC auch gegen GND geschaltet. Mit dieser Anordnung lässt sich der Widerstand des NTC, welcher stark temperaturabhängig ist, bestimmen.

1.1.5 IR-Decoder



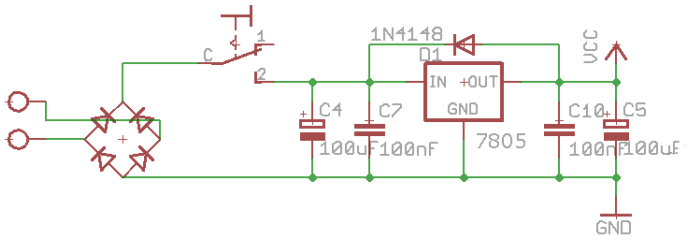
Auf dem Demoboard befindet sich ebenfalls ein IR-Decoder TSOP1736. Mit ihm ist es möglich, modulierte IR-Signale zu demodulieren. Das IC ist am empfindlichsten für IR-Signale, welche mit einem 36kHz-Träger moduliert wurden. Dies findet man z.B. bei allen gängigen Fernbedienungen (TV, Radio etc.) Der TSOP macht nichts anderes, als dieses Signal zu demodulieren und als eine Folge von High- und Low-Pegeln auszugeben. Die Auswertung des Signales müsste also vom AVR erledigt werden.

1.1.6 Serielle Schnittstelle



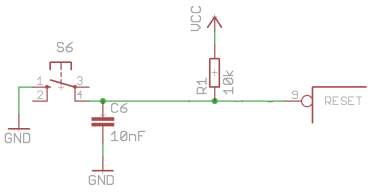
Der ATMega32 besitzt, wie die meisten anderen AVR, ein internes U(S)ART. Dies ist ein Protokoll, welches eine serielle Datenschnittstelle spezifiziert. Mit dieser Schnittstelle ist es möglich, die RS232-Schnittstelle eines PC anzusprechen. Alles, was dazu gebraucht wird, ist ein Pegelwandler (MAX232), da die RS232 nur mit anderen Pegeln arbeitet. Das Protokoll ist dasselbe.
Die TXD und RXD Pins des AVR werden über den Pegelwandler an eine DSUB09 (RS232) Buchse geführt. Damit wird eine sehr einfache Schnittstelle ermöglicht, welche nicht voll ausgenutzt wird. (man könnte noch zusätzliche Steuerleitungen verwenden, was hier aber nicht gemacht wird und im Allgemeinen auch nicht benötigt wird). Mit dieser Schnittstelle und einem Terminalprogramm (findet sich z.B. auf der Bastli-AVR-Demoboard-Website) könnt ihr also Daten an den PC senden und anzeigen.

1.1.7 Spannungsversorgung



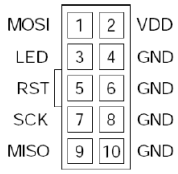
Das Demoboard enthält einen Festspannungsregler, welcher stabile 5V generiert. Der Anschluss erfolgt über einen Gleichrichter, die Polarität der ans Demoboard angelegten Spannung spielt also keine Rolle und man kann auch Wechselspannung anlegen.
Da der 7805 ein Linearregler ist, entwickelt er eine relativ hohe Verlustleistung, er wird also z.B. beim gleichzeitigen Betrieb aller LED auf dem Board recht warm. Beim Anschluss einer externen Schaltung muss besonders auf diese Verlustleistung geachtet werden, damit der 7805 nicht zu heiss wird. Alternativ kann man ihm einen kleinen Aufsteckkühlkörper spendieren oder die Eingangsspannung etwas verringern.

1.1.8 Reset-Beschaltung



Die Reset Schaltung ermöglicht es dem ISP-Programmierer, den Reset Pin des AVR mit dem Reset-Taster auf GND zu ziehen und den AVR damit zu resetten. Der 10k Ohm Widerstand ist zwingend notwendig, er verhindert, dass der Reset-Pin im Betrieb auf Low gehen kann. (=Reset). Der 10nF-Kondensator dient demselben Zweck, er filtert zusammen mit den 10k den Reset-Eingang und schützt ihn so vor Störungen, welche den AVR ungewollt resetten würden.

1.1.9 In-System-Programming-Anschluss



Auf dem Board ist ein standard 10-Pin AVR-ISP Anschluss. Das Demoboard kann mit jedem AVR-ISP-Programmiergerät programmiert werden.

1.1.10 Jumper

Auf dem Board befinden sich 3 Jumper. Mit diesen könnt ihr gewisse Funktionen des Boardes abschalten, um die AVR-Pins für eure eigenen Zwecke zu benutzen. Mit den Jumpern lassen sich die 7-Segmentanzeige, der NTC und der IR-Decoder vom AVR entkoppeln. Genaueres siehe im Schaltplan, welchen ihr auf der Bastli Homepage findet.

1.1.11 Stiftleiste für externe Versorgung

Unter dem AVR befindet sich die Stiftleiste ,,JP3". Hier liegen am linken Pin 5V, am rechten Pin GND an, sodass ihr eine externe Schaltung, welche vielleicht vom ATmega32 gesteuert wird, versorgen könnt. Der mittlere Pin ist nicht belegt. (Verhindert einen Kurzschluss durch einen Jumper) Beachtet aber, dass ihr diesen Pins nicht viel Strom entnehmen könnt. Der kritische Faktor ist die Verlustleistung am 7805 Spannungsregler! Beachtet deshalb unbedingt die Temperatur des Spannungsreglers, falls ihr zusätzlich Strom entnehmt. Wird sie zu hoch (Regler nicht mehr anfassbar), so muss dieser entweder gekühlt werden, oder die Eingangsspannung des AVR-Demoboardes muss reduziert werden. (Siehe ,,Spezifikationen - Spannungsversorgung" im nächsten Kapitel)

1.2 Spezifikationen

Die folgenden Hinweise sollen Tipps für einen sicheren und zerstörungsfreien Betrieb des Demoboardes liefern.

Spannungsversorgung:
Das Board verfügt über einen Gleichrichter und einen Standard 7805 Spannungsregler. Das Board wird über die Schraubklemme ,,Input" oben links auf dem Board mit Spannung versorgt. Die Polarität der angelegten Spannung spielt wegen dem integrierten Gleichrichter keine Rolle.
Die Maximale Eingangsspannung beträgt 12V DC oder 8V AC.
Achtet besonders bei der Speisung einer externen Schaltung mit dem 7805 des Demoboards darauf, dass dieser aufgrund der Verlustleistung nicht zu heiss wird. Ansonsten kann der Regler und/oder die ICs zerstört werden!. Spendiert dem 7805 daher bei Bedarf einen Aufsteck-Kühlkörper und geht mit der Eingangsspannung des AVR-Demoboardes runter. (Dies reduziert die Verlustleistung im 7805).

Strombelastung des AVR:
Der ATmega32 kann maximal 20mA pro Pin liefern. Daher ist bei der Benützung einer externen Schaltung, welche über die Stiftleisten mit dem AVR verbunden wird, darauf zu achten, dass die Pins des AVR nicht zu stark belastet werden, ansonsten können sie zerstört werden. In Summe kann der AVR nicht mehr als 200mA pro Versorgungsschiene liefern/schlucken. Sprich: Summiert man alle Ströme aller Pins auf, welche z.B. von VCC durch den AVR an die Pins nach GND fliessen, darf diese Summe 200mA nicht überschreiten. Umgekehrt genauso: der AVR kann summasummarum nicht mehr als 200mA in seinen eigenen GND ableiten.

2 Aufbauanleitung

In diesem Abschnitt soll besonders Anfängern der Aufbau des AVR-Demoboardes erleichtert werden.

2.1 Teileliste

Folgende Bauteile sollten mit dem Demoboard geliefert worden sein:

Anzahl	Bauteil
1	ATmega32
1	MAX232
1	TSOP1736 (IR-Receiver)
1	Sockel 40Pin
1	Quarz 16MHz
5	Kurzhubtaster
1	DIP-Schater
1	NTC 15k
1	7-Segment-Anzeige SC52-11GWA
6	LED 3mm Rot
1	Gleichrichter B380C1500 (1.5A)
1	L7805CV Spannungsregler
1	Schraubklemme 2Pol
1	SubD-Buchse
1	Schiebeschalter
1	Widerstandsnetzwerk 4.7k
1	Widerstand 100R
14	Widerstand 220R
8	Widerstand 390R
2	Widerstand 10k
1	Elko 4.7uF
2	Elko 100uF
4	Elko 1uF
6	Keramikkondensator 100nF
2	Keramikkondensator 22pF
1	Keramikkondensator 10nF
1	Stiftleiste 2x5 (ISP-Anschluss)
3	Stiftleiste 1x2 (div. Jumper)
1	Stiftlieste 1x3
3	Stiftleiste 1x8
1	Stiftleiste 1x6
3	Jumper
1	Diode 1N4148

2.2 Aufbau des AVR-Demoboards

Die Platine des Demoboardes ist bereits industriell vorgefertigt. Ihr müsst lediglich die Bauteile mit der Platine verlöten.
Beim Board wurde Wert auf einfache Lötbarkeit gelegt. Der Aufbau sollte deshalb keine grossen Schwierigkeiten bereiten. Bei den Links finden sich auch Seiten mit Tipps zum Löten.
Für den Aufbau benötigt man einen Elektronik-Lötkolben mit ca. 40-80W Heizleistung (Dachrinnen-Löt-Monster mit 200W eignen sich nicht!...), Lötzinn, Seitenschneider und sonstiges Standard-Werkzeug. Ansonsten kann der Bastli gerne für Tipps, Material(Messplatz) und Support kontaktiert werden.

Für den Aufbau bewährt es sich, den Schaltplan und das Layout zur Hand zu haben, damit man sieht, welches Bauteil wo und wie plaziert werden muss. Auf der Homepage des Bastli finden sich ausserdem Fotos der aufgebauten Platinen, welche ebenfalls als Referenz verwendet werden können.
Die Bauteile werden durch die Platine gesteckt und auf der Unterseite verlötet. Der ATMega32 wird gesockelt, damit man ihn bei einem allfälligen Defekt einfach austauschen kann.
Die Bauteile sind alle relativ robust, eine Zerstörung durch zu langes ,,braten" an den Bauteilen mit dem Lötkolben ist nicht sehr wahrscheinlich. Dennoch sollte die Löttemperatur nicht zu hoch (ca. 325-350°C) gewählt werden.

2.3 Aufbauhinweise

Beginnt am Besten mit den kleineren, nicht sehr hohen Bauteilen wie den Widerständen, Drahtbrücken oder Tastern. Zuletzt lötet man die hohen, klobigen Bauteile wie den Spannungsregler, die Elektrolytkondensatoren oder die Subd-Buchse ein.
Die Bauteile werden gemäss Layout/Bestückungsdruck/Photos in die Platine gesteckt und auf der Unterseite verlötet. Die Seite der Platine mit dem AMIV-Logo und dem übrigen Bestückungsdruck ist die Oberseite, hier kommen die Bauteile hin und werden auf der anderen Platinenseite verlötet.

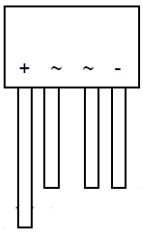
Es folgen nun einige spezielle Aufbauhinweise, da gewisse Bauteile mit einer vorgegebenen Orientierung eingebaut werden müssen.

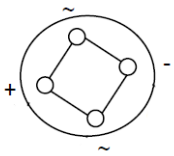
2.3.1 Elektrolytkondensatoren

Die Elkos haben eine Polung, sie können nicht beliebig eingelötet werden. Beachtet das Layout und den Bestückungsdruck. Die Seite des Elkos mit dem weissen Streifen ist der negative Anschluss und muss ein kleineres Potential als der andere

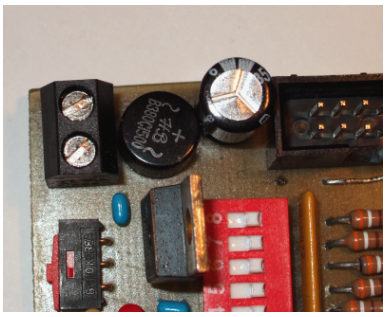
Pin führen. Ansonsten wird der Kondensator beschädigt und kann platzen. Auf dem Bestückungsdruck hat es ein kleines „+". An diese Seite muss also diejenige Seite des Elkos, die keinen weissen Streifen hat.

2.3.2 Gleichrichter





Beim Gleichrichter ist auf die Einbaurichtung zu achten. Der Bestückungsdruck und das nebenstehende Foto sollten die nötigen Infos für den richtigen Einbau liefern.



2.3.3 Leuchtdioden

Die LEDs haben ebenfalls eine Polung und können nicht beliebig eingelötet werden. Der kurze Anschlussdraht der LED ist die Kathode. (Eselsbrücke: **k**urz für **K**athode). Die Kathoden der 5 LEDs sind über die Vorwiderstände mit den AVR-Pins verbunden, die Kathoden zeigen also in Richtung AVR. Die Anoden hängen direkt an VCC(positive 5V-Versorgung). Bei der LED am Spannungsregler zeigt die Kathode in Richtung Platinenrand/Vorwiderstand.

2.3.4 Siebensegment-Anzeige

Der Punkt der Anzeige muss unten rechts und nicht oben links sein.

2.3.5 Widerstandsnetzwerk

Zwischen dem DIP-Schalter und den 390-Ohm-Widerständen ist das Widerstandsnetzwerk, welches die 4.7kPullDown-Widerstände enthält, einzulöten. Der gemeinsame Pin, welcher mit einem kleinen Punkt gekennzeichnet ist, muss auf der Seite Richtung ISP-Header sein. Der neunte Widerstand am Netzwerk wird nicht gebraucht und dessen Pin muss mit einer Zange abgetrennt werden, da keine Bohrung dafür vorhanden ist.

2.3.6 Diode am Spannungsregler

Am 7805 befindet sich eine Diode, welcher den Spannungsregler beim Ausschalten schützt, da dann die Ausgangsspannung aufgrund der Kondensatoren grösser als die Eingangsspannung am Regler ist. Die Kathode der Diode ist mit einem schwarzen Ring gekennzeichnet, sie zeigt in Richtung Gleichrichter.

2.3.7 ICs

Die Pins des ATmega32 und des MAX232 müssen, bevor man sie in den Sockel stecken kann, etwas nach innen gebogen werden. Ansonst passen die ICs nicht in den Sockel. Dies macht man am Besten, indem man das IC mit der Breitseite auf den Tisch stellt und dann gleichzeitig alle Pins einer Seite etwas nach innen drückt, indem man das IC nach unten auf den Tisch drückt.

2.3.8 Drahtbrücken

Auf der Platine müssen 3 Drahtbrücken eingelötet werden. Dazu verwendet man am Besten ein abgeknipstes Stück Draht von einem Widerstand o.Ä., biegt es zurecht, sodass es in die Bohrungen passt und verlötet es.

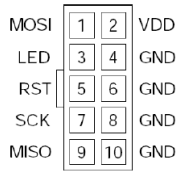
2.4 Inbetriebnahme

Nachdem ihr alle Bauteile verlötet habt, wird es Zeit, das Board in Betrieb zu nehmen. Nehmt den ATmega32 für den ersten Test wieder aus dem Sockel, damit er bei einem allfälligen Defekt in der Spannungsversorgung nicht ,,gebraten" wird. Legt nun eine Spannung zwischen 7 und 12V DC (oder bis zu 8V AC) an der Eingangsklemme an und schaltet den Hauptschalter ein (rote LED leuchtet). Messt nun mit einem Multimeter die Spannung an der Stiftleiste JP3. Sie muss bei ca. 5V liegen. Ist alles OK, könnt ihr den AVR wieder in den Sockel stecken und mit der Programmierung fortfahren.

3 Programmierung

3.1 Allgemeines zur Programmierung

Den AVR kann man mit einem geeigneten In-System-Programmer programmieren. Das Demoboard besitzt den von Atmel genormten 10Pin-ISP-Anschluss:



Nun benötigt man lediglich einen passenden Programmieradapter, um das Programm vom PC auf den AVR zu übertragen.
Der Bastli hat zum Beispiel einen solchen im Angebot: den USBASP 2.0. Dieser kann mit avrdude angesprochen werden.

3.2 Programmierung - HowTo unter Windows

Dieses kleine Tutorial soll Anfängern einen Überblick über das Vorgehen geben, wie man unter Windows eine Toolchain zur Entwicklung und Programmierung von AVR-Code einrichtet.
Es wird hier als Beispiel WinAVR, das AVRstudio, der AVR Burn-O-Mat und der USBASP 2.0 vom Bastli verwendet.
WinAVR erlaubt euch die Benutzung des Gnu C Compilers unter Windows. Das AVRstudio ist eine Entwicklungsumgebung für AVRs mit integriertem Simulator, welcher noch ganz praktisch ist. Eigentlich wäre AVRstudio für die Assemblerprogrammierung gedacht, doch es kann WinAVR als Plugin nutzen und so kann man auch in C programmieren und alle Vorzüge des AVRstudios nutzen. Verwendet man einen mit dem AVRstudio kompatiblen Programmer(z.B. den MKII von Atmel), so kann man auch direkt aus dem AVRstudion den AVR flashen. Andernfalls braucht man eine externe Software, welche das vom AVRstudio erzeugte HexFile in den AVR brennt. Dies wäre in diesem Fall der avrdude, welcher auch den Bastli USBASP 2.0 ansprechen kann und schon mit WinAVR mitgeliefert wird.

3.2.1 Installation der Programme

Es werden die folgenden Programme/Sammlungen benötigt:

1. WinAVR
   Dies ist die Toolchain, welche grundsätzlich den C-Compiler und noch ein paar andere Tools enthält, darunter auch den avrdude.

2. AVRstudio
   Das AVRstudio ist die Entwicklungsumgebung, hier werden die Projekte verwaltet, sowie der Code programmiert und compiliert. Er kann dann je nach Wunsch im Simulator simuliert werden.

3. AVR Burn-O-Mat
   Dieses Programm ist lediglich eine Java-basierte graphische Oberfläche für den avrdude. Der avrdude ist ein kommandozeilenbasierter Programmer. Da der geneigte Windows-User aber nicht gerne Kommandozeilen quält, wird eine hübsche Oberfläche verwendet. Ausserdem können die Fuses des AVR viel bequemer programmiert werden als in der Kommandozeile.

Begonnen wird mit der Installation von WinAVR. Ihr findet die aktuellste Version unter http://winavr.sourceforge.net/
Installiert es am Besten unter C:\Programme. Das Programmers Notepad müsst ihr nicht installieren (Das ist ein Editor). Mit WinAVR habt ihr avrdude auch schon auf eurer Festplatte.
Die Inhalte vom WinAVR-Paket werden eigentlich nur im Hintergrund benutzt, ihr braucht euch da gar nicht gross darum zu kümmern.

Nun ist das AVRStudio an der Reihe.
Ihr findet die aktuellste Version auf der Atmel-Homepage: www.atmel.com
Für den Download ist eine Registrierung erforderlich.
Das AVRstudio merkt selbstständig, dass WinAVR installiert ist und bindet die nötigen Ressourcen automatisch ein.

Nun hat man eine vollständige Entwicklungsumgebung in C für AVR-Controller installiert. Das AVRstudio erzeugt euch beim compilieren ein Intel-Hex-File. Dieses File muss nun noch in den AVR gebrannt werden. Dazu verwenden wir AVR Burn-O-Mat.

Der AVR Burn-O-Mat kann hier bezogen werden: http://avr8-burn-o-mat.aaabbb.de/
Installiert ihn auf eurer Festplatte.
Nach der Installation startet ihr den Burn-O-Mat und geht nach "Settings", "AVRDUDE".
Hier muss nun der Pfad vom avrdude angegeben werden. Das file avrdude.exe findet ihr im WinAVR-Ordner unter \bin\. Das configuration-File avrdude.conf findet sich im selben Ordner.
In diesem Tutorial verwenden wir als Programmer den USBASP 2.0 vom Bastli. Wählt deshalb unter "Programmer" den usbasp und unter "Port" usb aus. Der Rest kann so bleiben, wie es ist.
Es wird angenommen, dass der USBASP 2.0 im System bereits installiert wurde (Siehe Doku zum USBASP). Nun haben wir alles fertig installiert und konfiguriert.

3.2.2 Einstellungen der Fuses

Alle AVRs haben einen Satz Konfigurierungs-Bits, Fuses genannt. Mit diesen werden einige grundlegende Operationen des AVRs wie z.B. die Taktquelle festgelegt. Mit ihnen kann man sich aber auch aus dem AVR ,,aussperren", indem man z.B. eine Taktquelle wählt, welche nicht am AVR angeschlossen ist, deshalb ist beim setzen der Fuses etwas Vorsicht geboten.

Der ATmega32 hat werksseitig das JTAG-Interface aktiviert, welches ein paar Pins benötigt. Dieses wollen wir nun ausschalten, da wir es nicht benötigen.

Schliesst also das mit Spannung versorgte Demoboard über den USBASP 2.0 an den Computer an und startet den AVR Burn-O-Mat. Stellt sicher, dass am USBASP 2.0 der ,,slow sck" jumper gesetzt ist.
Wählt bei ,,AVR Type" den ATmega32 aus und klickt auf ,,Fuses", ein neues Fenster öffnet sich, in welchem ihr die Fuses des ATmega32 sieht.
Klickt nun auf ,,read Fuses". Der Burn-O-Mat sollte ein erfolgreiches Auslesen der Fuses bestätigen. Wählt nun beim dropdown-Menü oben rechts den expertmode.
Entfernt nun das Häckchen beim ,,JTAGEN"-Fuse und klickt auf ,,write Fuses". Wiederum bestätigt das Programm ein erfolgreiches Schreiben der Fuses und der AVR hat nun das JTAG-Interface abgeschaltet. Tut man dies nicht, funktioniert die 7-Segmentanzeige auf dem Demoboard nicht richtig, da die nötigen Pins vom JTAG-Interface belegt werden.
Alle anderen Fusebits können noch unangetastet bleiben.

3.2.3 Das erste Programm

Nun wollen wir ein Programm in den AVR laden. Dazu verwenden wir das ,,AVR_Demoboard_Testprogramm", welches ihr auf der Bastli-Homepage herunterladen könnt. Das Programm testet alle Funktionen des AVR-Demoboardes und erlaubt euch daher eine Überprüfung eurere Lötarbeit.
Schliesst das mit Spannung versorgte Demoboard mit dem USBASP 2.0 an den PC an und startet den AVR Burn-O-Mat. Stellt sicher, dass beim USBASP 2.0 der ,,slow sck" jumper gesetzt ist.
Werksseitig läuft der AVR mit dem internen 8 MHz RC-Oszillator. Für das Beispielprogramm wird der externe 16MHz Quarz benötigt, wir müssen also die Fuses neu setzen.
Klickt daher im Burn-O-Mat auf ,,Fuses" und lest die aktuelle Fusebit-konfiguration ein. Wechselt nun in den Tab ,,Oszillator/Clock options" und dann in den Tab ,,External crystal or ceramic resonator". Wählt ,,crystal", ,,8-16MHz" und ,,slowly rising power". Geht nun zurück zum Tab ,,Fuse Editor". Nun ist beim ,,CKOPT"

Fuse ein Häckchen gesetzt und bei keinem CKSEL und SUT Fuse ein Häckchen. Klickt nun auf ,,write fuses". Falls alles geklappt hat, bestätigt der Burn-O-Mat das erfolgreiche Schreiben der Fuses und der AVR lässt sich weiterhin ansprechen, indem man z.B. auf ,,read fuses" klickt und die fuses erfolgreich ausgelesen werden. Schliesst nun das Fuse-Fenster. Der ,,slow-sck" jumper am USBASP 2.0 kann nun wieder entfernt werden.

Das Beispielprogramm kommt direkt mit einem fertig erzeugten Intel-hex-file daher. Wählt unter ,,Flash" beim Burn-O-Mat dieses Hexfile aus. Es findet sich im Ordner ,,default" des Beispielprogrammes. Klickt nun auf ,,Write" - der Burn-O-Mat bestätigt ein erfolgreiches Schreiben des AVR.
Das Programm nutzt ebenfalls die serielle Schnittstelle, diese ist wie folgt konfiguriert: 9600 Baud, 8 Data bits, no parity, 1 Stop bit, no handshake. Auf der Bastli-Homepage findet ihr ein Terminalprogramm, welches euch die auf der seriellen Schnittstelle empfangenen Daten anzeigt.
Nun sollten die folgenden Funktionen am Demoboard ersichtlich sein:

- Die 7-Segmentanzeige zählt hoch
- Der Punkt der 7-Segmentanzeige blinkt
- Die LEDs bilden ein kleines Lauflicht
- Wenn ihr mit einer Fernbedienung (egal welcher Typ, hauptsache IR mit 36kHz moduliert, was die Meisten machen) auf den TSOP 1736 zielt und eine Taste an der Fernbedienung drückt, sollten alle LED-Anzeigen stillstehen und beim Loslassen der Taste weiterlaufen.
- Wenn ihr den Taster S5 drückt, wird der ADC-Wert des NTC per UART an den PC geschickt und laufend angezeigt. Je wärmer der NTC, umso kleiner ist dieser Wert.
- Wenn ihr die Taster S2, S3 oder S4 drückt, wird per UART die Meldung an den PC geschickt, welcher Taster gerade gedrückt wird.
- Stellt ihr beim DIL-Switch diese Konfiguration ein: ,,00011000", so wird dies ebenfalls mit einer Meldung an die serielle Schnittstelle quittiert.

Mit diesem Programm konntet ihr also alle Funktionen eures Demoboardes testen.

3.3 Pinbelegung des AVR

Die folgenden Tabellen zeigen, an welchem AVR-Pin welche Funktion des Demoboardes angeschlossen ist. (Diese Informationen könnte man auch aus dem Schaltplan entnehmen).

	Nr.	Pin
DIL-Switch	1	PB0
	2	PB1
	3	PB2
	4	PB3
	5	PB4
	6	PB5
	7	PB6
	8	PB7

	Segment	Pin
7-Seg-Anzeige	dot	PC0
	a	PC1
	b	PC2
	c	PC3
	d	PC4
	e	PC5
	f	PC6
	g	PC7

	Nr.	Pin
LEDs	LED1	PD7
	LED2	PD6
	LED3	PD5
	LED4	PD4
	LED5	PD3

	Typ	Pin
Diverse	TSOP Input	PD2(INT0)
	NTC Input	PA7 (ADC 7)
	Taster S2	PA0
	Saster S3	PA1
	Taster S4	PA2
	Taster S5	PA3

4 Linksammlung

Die folgenden Links bieten zusätzliche Infos zum Löten und Programmieren:

Bastli Homepage: http://www.bastli.ethz.ch
Umfangreiches Tutorial zum AVR-GCC: http://www.mikrocontroller.net/articles/AVR-GCC-Tutorial
Tutorial zum Löten: http://www.roboternetz.de/wissen/index.php/L%C3%B6t-Tutorial
WinAVR-GCC: http://winavr.sourceforge.net/
AVR Studio: http://www.atmel.com/dyn/Products/tools_card.asp?tool_id=2725