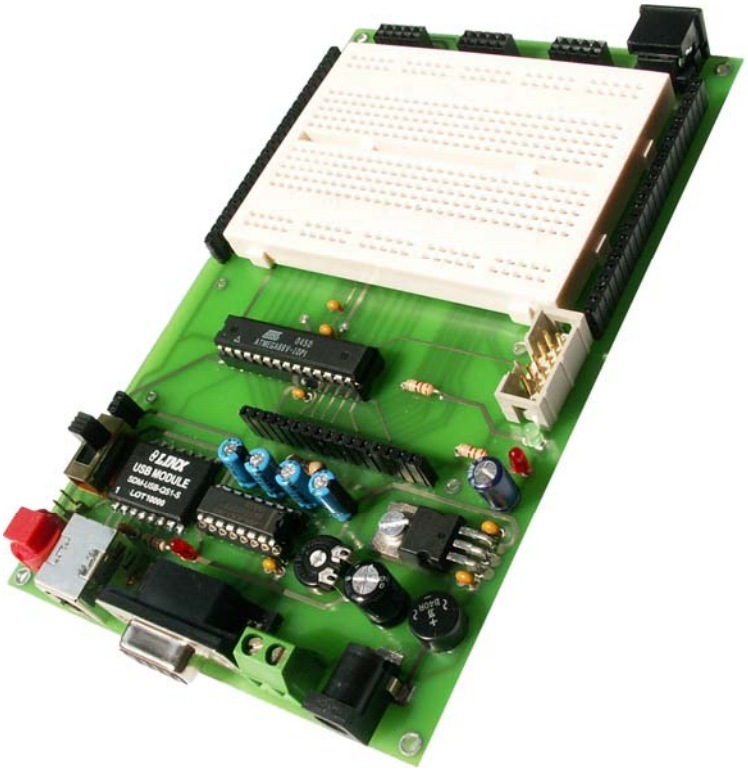
Version

0.4

MCS ELECTRONICS

Making Things Easy



Lehr- und Experimentierbaukasten (EDB) Merkmale::

- LCD
- PS/2 Tastatur/Maus Schnittstelle
- Steckbrett
- USB Interface
- RS232 Schnittstelle
- ISP Verbindung

- Wird mit ausführlichem Handbuch geliefert

Lehr- und Experimentierbaukasten (EDB) Handbuch

FÜR VERFÜGBARE UPDATES BESUCHEN SIE

www.mcselec.com/edb

MCS ELECTRONICS

Lehr- und Experimentierbaukasten (EDB)

Barry de Graaff

www.mcselec.com

Inhaltsverzeichnis

Inhaltsverzeichnis *(Fortsetzung)*

Vorwort

Ende der 80-iger Jahre des letzten Jahrhunderts waren Mikrocontroller relativ teuer und fanden ausschliesslich in Computern und Consumer Electronic (Fernsehgeräte und Stereoanlagen) Verwendung. Aufgrund der grossen Nachfrage nach einfacher und flexibler elektronischen Geräten wurden Mikrocontroller immer erschwinglicher. Heutzutage sind Mikrocontroller für Jeden verfügbar, der ein Interesse an der Entwicklung elektronischer Baugruppen hat.

Der Lehr- und Experimentierbaukasten zeigt Ihnen einen Weg in die Welt der Elektronik und Mikrocontroller. Die Platine in Verbindung mit dem Handbuch werden Ihnen die Elemente der Elektronik und Mikrocontoller näher bringen. Sie werden die Bestandteile anhand von Experimenten kennenlernen. Die Experimente bauen logisch aufeinander auf und werden Sie zu neuen Versuchen anspornen.

Bei den Experimenten liegt der Schwerpunkt in der Verbindung einfacher Bauteile, wie Widerstände oder Leuchtdioden mit Mikrocontrollern. Wo benötigt wird eine einfache Erklärung der Bauteile oder der theoretischen Grundlagen der Controller gegeben.

Verfallen Sie nicht Panik, wenn Sie den Controller in den ersten paar Versuchen zunächst nicht verstehen. In den ersten Versuchen behandeln wir den Controller als „Black Box“. Im weiteren werden wir Ihnen die theoretischen Grundlagen zu Controllern und deren Links ins Internet vermitteln.

Bei dieser Gelegenheit möchte ich auch noch auf die mit dem Handbuch gelieferte CD verweisen, auf der sich die Source Codes für die Experimente, die Lösungen der Übungsaufgaben ebenso wie Visual Basic Beispiele, Datenblätter und Applikationsberichte befinden. Eine Übersicht hierrüber finden sie im Hauptverzeichnis der beigefügten CD „Contents of CD.pdf“.

Wir von MCS Elektonic wünschen Ihnen viel Vergnügen bei den Experimenten und freuen uns auf Ihre Kommentare zu diesem und unseren anderen Produkten. Für zusätzliche Informationen oder Kontakt zu uns besuchen Sie bitte unsere Webseite www.mcselec.com.

Kapitel

1

1. Die Praxis

Dieses Kapitel beschreibt, welche Grundkenntnisse für den Start mit dem Lehr- und Experimentierbaukasten (EDB) benötigt werden.

Der Lehr- und Experimentierbaukasten ist für Leute gedacht, die wenig oder gar keine Erfahrung im Umgang mit Mikrocontrollern haben. Damit Sie Ihre eigene Experimentierplatine aufbauen können, sollten über etwas Erfahrung im Zusammenbau und Löten elektronischer Bausätze verfügen. Hierzu wird Computerbasiswissen vorausgesetzt.

1.1 Hardware Voraussetzungen

Systemanforderungen Windows 95, 98, NT, 2000 oder XP. In diesem Handbuch benützen wir die serielle Schnittstelle zur Programmierung des Chips. Wenn Sie über keine serielle Schnittstelle verfügen, können Sie den STK200/300 Dongle oder den Wiazania USB Programmer benutzen. In Kapitel 4 wird Ihnen die Benutzung des STK200/300 Dongles und des Wiazania USB Programmers gezeigt.

1.2 Vorsicht bei unprogrammierten Bauteilen

Der von MCS Electronics mitgelieferte Mikrocontroller ATMega88 wurde bereits programmiert, um die serielle Programmierung zu ermöglichen. Falls Sie (Ausversehen) den Chip gelöscht haben, oder sich einen neuen gekauft haben, benötigen Sie zur (Re-) Programmierung den STK200/300 Dongle oder den Wiazania USB Programmer. Die Anleitung hierzu finden Sie in Kapitel 4.

Kapitel

2

2. Inbetriebnahme

Dieses Kapitel zeigt Ihnen die Inbetriebnahme des Lehr- und Experimentierbaukastens

Dieses Kapitel zeigt Ihnen die Inbetriebnahme des Lehr- und Experimentierbaukastens

Bitte befolgen Sie die Anweisungen in Kapitel 2 Schritt für Schritt. Auf diese Art und Weise ermöglichen Sie sich einen reibungslosen Weg zu den Experimenten in Kaptitel 3 ohne unliebsame Überraschungen.

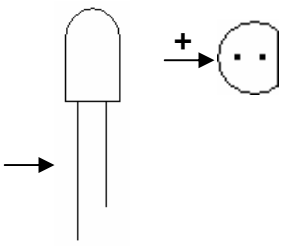
2.1 Montage Ihrer Platine

Kapitel 2.1 führt Sie durch die Montage Ihres EDB`s. Am besten löten Sie die Bauteile analog zur Stückliste, die Sie auf auf Seite 18 finden.

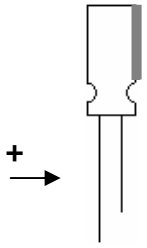
Auf der Platine finden Sie Lötauge wie diese: 

Bei IC´s markiert das quadratische Lötauge Pin Nummer 1. Bei Leuchtdioden und Kondensatoren bedeutet das quadratische Lötauge den Pluspol. Nachfolgend finden Sie einige Zeichnungen, die Ihnen helfen werden, die Positionen der Bauteile zu finden.

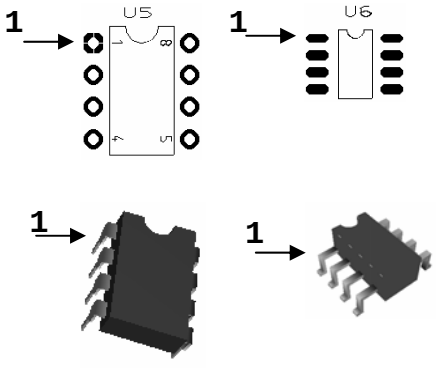
LED



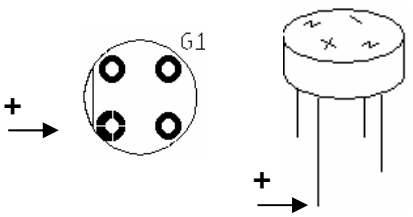
Kondensator



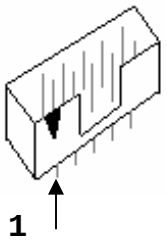
IC



Gleichrichter



Steckverbinder



Widerstände

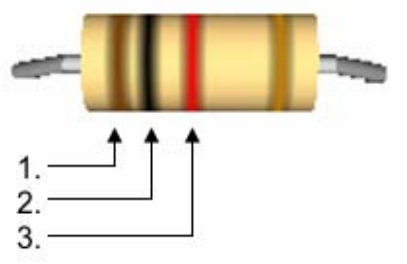
Widerstände sind nicht polarisiert, daher ist es gleichgültig, welcher Pin mit welchem Pad verbunden wird.

Jedoch ist zu beachten, dass der richtige Widerstandswert an die passende Position auf die Platine gelötet wird.

Um den richtigen Widerstandswert zu bestimmen kann man entweder ein Ohm-Meter benutzen, oder ihn mittels des aufgedruckten Farbcodes bestimmen.

Farbcode	Wert Ring Eins oder Zwei	Wert Ring Drei Multiplikationsfaktor
Black	0	x1
Brown	1	x10
Red	2	x100
Orange	3	x1000
Yellow	4	
Green	5	
Blue	6	
Violet	7	
Grey	8	
White	9	

Beispiel für den Farbcode:



1. Brain = 1
2. Schwarz = 0
3. Rot = x100

Somit 1000 Ohm = 1kOhm

Der vierte Ring gibt die Toleranz an. Gold bedeutet eine Toleranz von 5%.

Stückliste

Bauteil	Beschreibung	Wert
R1, R5	Widerstand ¼ W	330Ω
R4	Widerstand ¼ W	220Ω
C3, C4, C9, C10, C11, C12, C13	Keramikkondensator	100nF
R2	Widerstand 2 W	47Ω
U1	IC-Sockel 28 DIL	Für U1
U3	IC-Sockel 16 DIL	Für U3
U4	Spannungsregler	7805
R3	Potenziometer	10kΩ
G1	Brückengleichrichter	
D1, D2, D3	LED Leuchtdiode	3mm
X15	Pfostenleiste weiblich	1x2
X5, X6, X7	Pfostenleiste weiblich	HDR2x5
X3, X4	Pfostenleiste weiblich	HRD1x24
X12	Pfostenleiste LCD	HDR1x16
X13	USB Stecker	Typ B weiblich
X9	Pfostenstecker ISP	10 polig
X1	Stromversorgung	Stromversorgungsstecker
C5, C6, C7, C8	Elko radial	1µF
C2	Elko radial	100µF
C1	Elko radial	220µF
Reset	Reset Taste	SW_SPST
X16	Stecker SubD-9 weiblich	DB9FL
X2	Stromversorgung	Schraubklemme
X10	PS2 Tastatur	MiniDin6
S1	USB/UART Umschalter	Schalter
-	Lochrasterplatte	7x7cm

Diese Bauteile nicht löten:
1x ATMega88 (U1), 1x USB Modul SDM-USB-QS1-S (U2)

Ratschlag
Wir raten Ihnen, die IC Sockel und nicht die IC`s direkt in die Leiterplatte einzulöten.
Für den ATMega88 empfehlen wir Ihnen einen einfachen IC Sockel zu benutzen, da dieser ein leichteres Austauschen gegenüber gedrehten Qualitätssockeln ermöglicht.

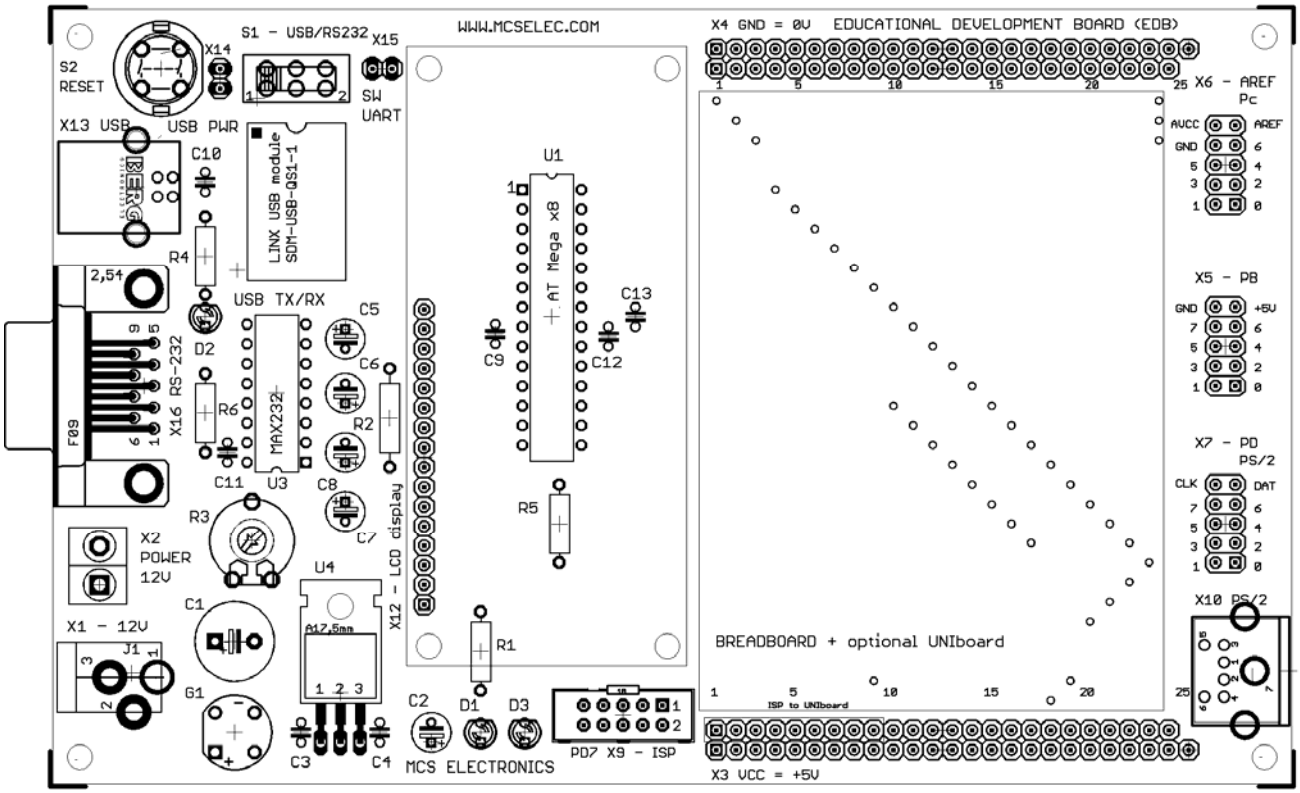
Das USB Modul kann nur direkt aufgelötet werden. **Bitte löten Sie es jetzt noch nicht ein.**

Ebenso die IC`s und das Display zum jetzigen Zeitpunkt noch nicht einsetzen.

Stromversorgungsoptionen
Sie haben zwei Möglichkeiten. Entweder eine externe Stromversorgung an X2 anzubringen, oder einen Stromversorungsadapter an X1 zu benutzen.
Sie können das Entwicklungsboard auch über den USB-Bus versorgen. Dazu müssen Sie eine Kabelbrücke an X14 einlöten.
Vorsicht, bei Benutzung der Stromversorgung via USB, darf der Spannungsregler (U4) LM7805 nicht eingelötet werden. Ferner darf keine Stromversorgung an X1 oder X2 angeschlossen sein.

Vorsicht, der USB-Bus darf nicht zu stark belastet werden (MCS Electronic kann für mögliche Schäden am PC oder USB-Bus nicht haftbar gemacht werden).

Platinenlayout
Um die richtigen Positionen der Bauteile zu finden, benützen Sie die abgebildete Zeichnung als Vorlage. Diese ist auch auf der Leiterplatte aufgedruckt und Sie finden sie in vergrösserter Form in Anlage 4.



Bevor Sie die Stromversorgung einschalten

Wenn Sie alle Bauteile eingelötet haben, überprüfen Sie die Platine nach Lötrückständen und beseitigen Sie diese.

Messen Sie den Widerstand zwischen folgenden Pins:
Zwischen X2.1 und 2.2 sollte er ca. 18 MΩbetragen.
Zwischen X2 und X4 sollten er ca. 2,6kΩ sein.

Sollte der Widerstand nahe 0Ω sein, haben Sie irgendwo einen Kurzschluss. Auf keinen Fall die Stromversorgung einschalten! Überprüfen Sie erneut die Lötpunkte und prüfen Sie, ob U4 und G1 richtig herum eingelötet sind.

Sollten Sie einen unendlich hohen Widerstand messen, sind möglicherweise die Steckverbinder X2, X3 oder X4 unzureichend eingelötet. Löten Sie diese nach.

Für den Fall dass die Werte etwas abweichen, vergewissern Sie sich, dass diese innerhalb einer Grenze von +/- 10% der oben genannten Werte sind.

Wenn alles in Ordnung ist, verbinden Sie die Stromversorgung (oder USB-Kabel). Die externe Stromversorgung sollte zwischen 9 und 15 Volt Gleichspannung betragen. Die Polarität ist nicht entscheidend, solange der Brückengleichrichter benützt wird.

Die Power LED sollte jetzt leuchten. Messen Sie die Spannung zwischen Pin 2 und 3 an U4 (ca. 5 Volt).

Entspricht die o.g. Spannung nicht diesem Wert, unterbrechen Sie die Stromversorgung und überprüfen Sie die Platine erneut.

Ist die Spannung ok, trennen Sie die Stromversorgung und stecken Sie dann den MAX232 sowie den ATMega88 in die vorgesehenen Sockel. Zuletzt löten Sie das USB Modul ein.

Löten Sie den einreihigen Steckverbinder in das LCD Modul. Stecken Sie das LCD Modul in den Steckverbinder X12 ein.

2.2 Installation von Bascom AVR

Für die Programmierung des ATMega88 µ-Controllers mit dem EDB wird Bascom AVR benötigt. Sie können zwischen der kostenlosen Demoversion, die 4kByte Code compilieren kann, oder der beziehbaren Vollversion wählen. Beide Versionen sind auf der Seite http://www.mcselec.comverfügbar.

Installationsprozedur der Vollversion
Die Vollversion wird als CD geliefert. Normalerweise wird die CD nach Einlegen ins Laufwerk automatisch gestartet. Falls nicht, selektieren Sie das CD ROM Laufwerk und doppelklicken Sie auf Setup Exe. Fahren Sie auf Seite 19 fort.

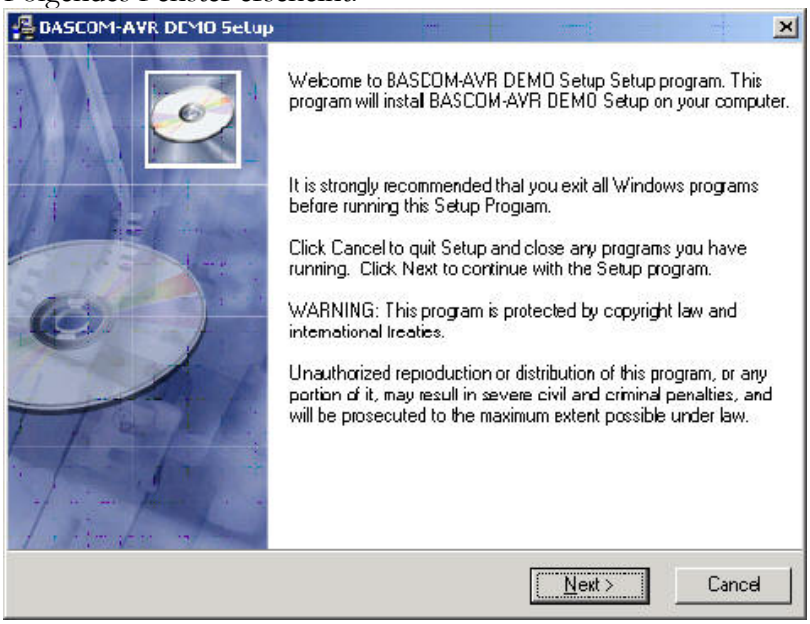
Installationsprozedur der Demoversion
- Downloaden Sie das Installationsfile (BCAVRD.zip) von www.mcselec.com.

- Mit Windows XP können Sie das ZIP File direkt öffnen. Stellen Sie sicher, dass Sie alle Files im gleichen Verzeichnis extrahieren. Falls Sie die Zip Files nicht öffnen können, installieren Sie Win ZIP www.winzip.com)
  Starten Sie setup.exe nicht aus dem Zip Verzeichnis!

Nach der Extraktion erhalten Sie einen Ordner, der das setup.exe File beinhaltet.
- Zur Installation starten Sie setup.exe.

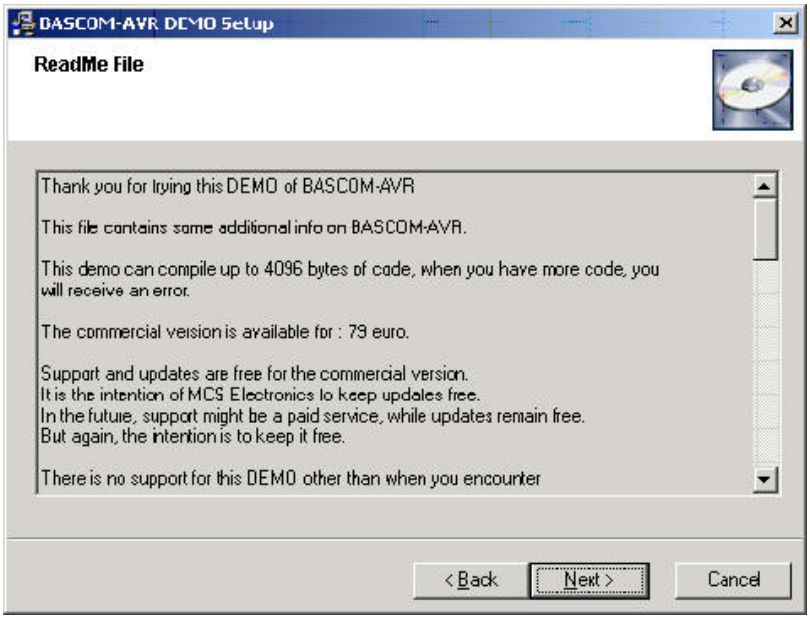
Fahren Sie auf der nächsten Seite fort.

Folgendes Fenster erscheint:



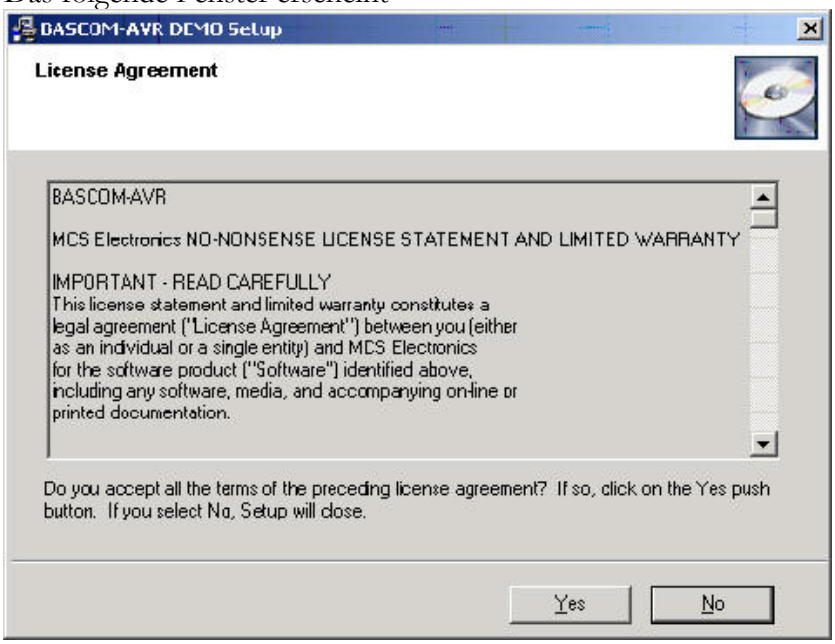
Weiter mit „next“

Das folgende Fenster erscheint:



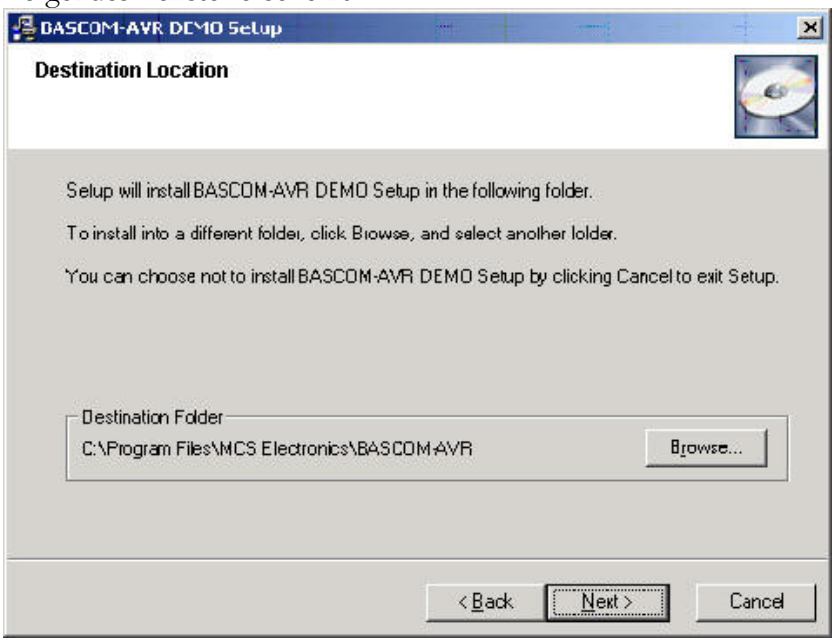
Weiter mit „next“

Das folgende Fenster erscheint



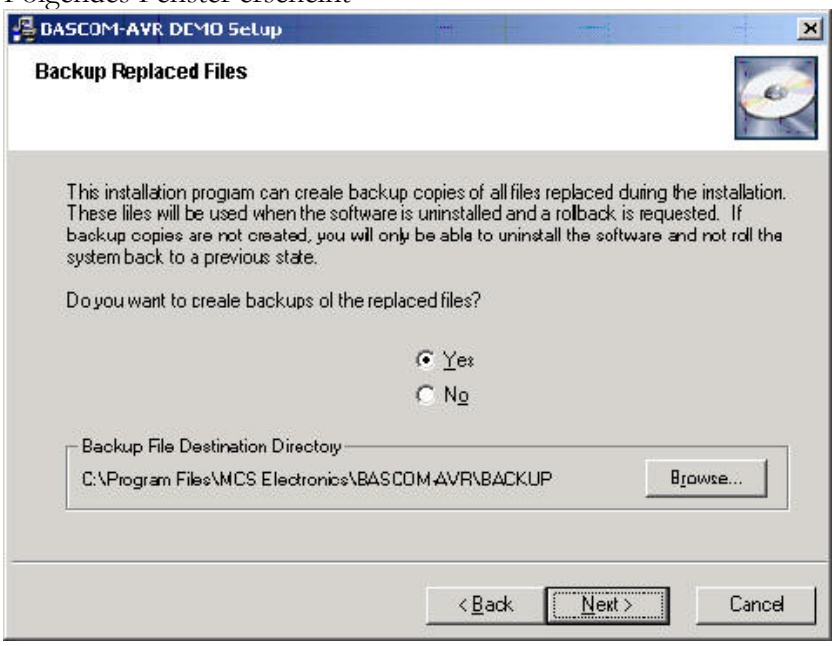
Weiter mit „Yes“

Folgendes Fenster erscheint

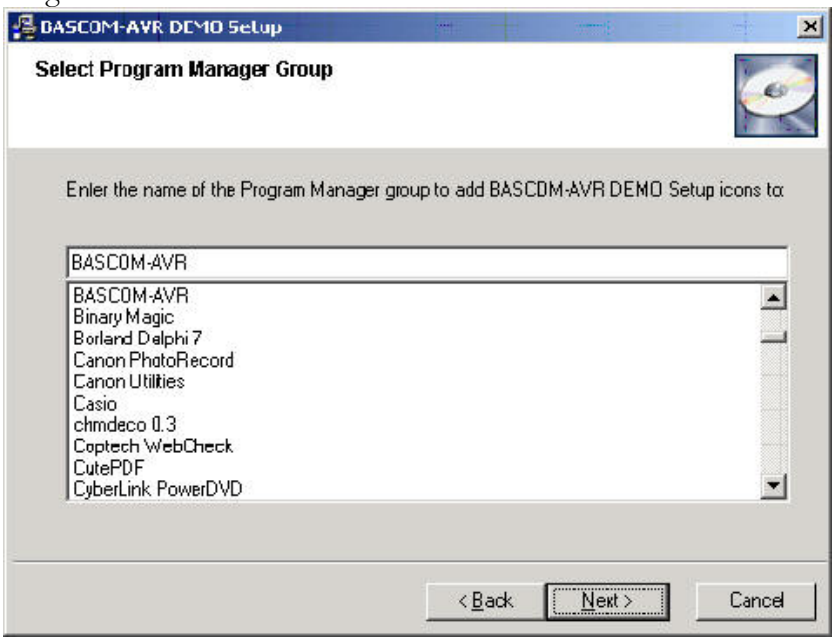


Sie können das Laufwerk und das Verzeichnis selektieren. Nachdem Sie das Zielverzeichnis selektiert haben, weiter mit „next“.

Folgendes Fenster erscheint

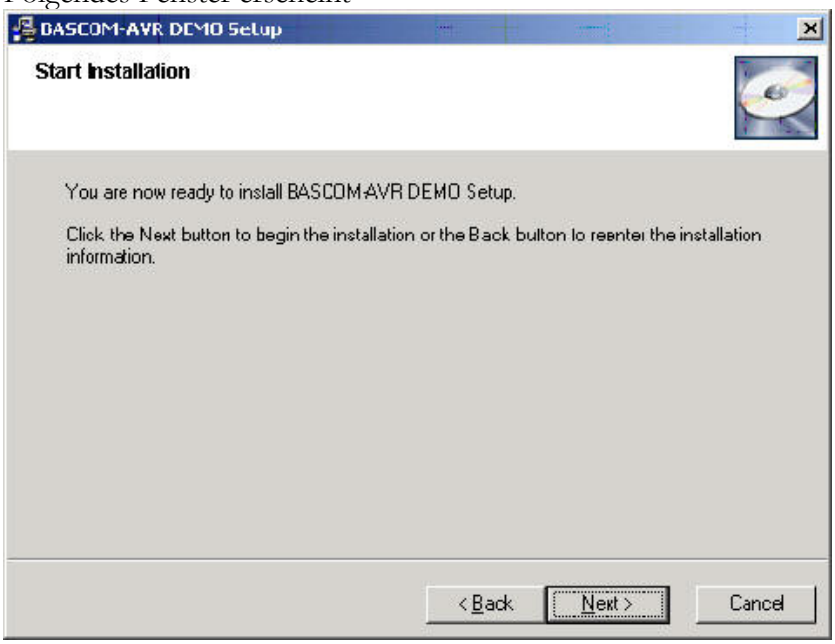


In diesem Fenster können auswählen, ob sie ein Backup von allen ersetzten Files erzeugen wollen. Normalerweise werden keine Dateien ersetzt, solange sie noch nicht existieren.
Selektieren Sie „yes“ und klicken Sie auf „next“.
Folgendes Fenster erscheint

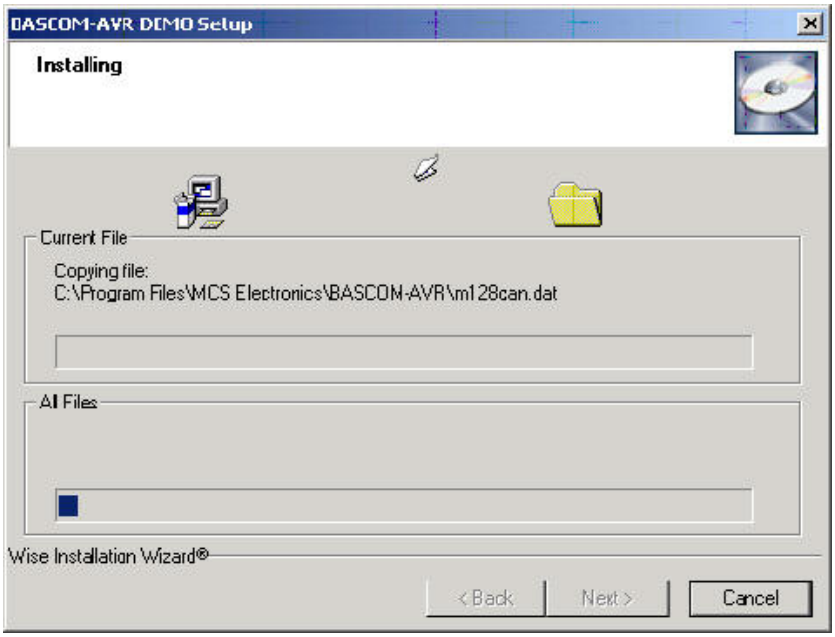


Sie können auswählen, in welcher Programmgruppe Bascom installiert werden soll. Klicken Sie auf „next“ nachdem Sie eine Programmgruppe selektiert haben.

Folgendes Fenster erscheint



Das Setup-Programm hat nun alle erforderlichen Angaben für die Installation. Drücken Sie „next“, um die Installation zu starten.



Während der Installation erscheint das oben gezeigte Fenster.

Bascom AVR kann nun benutzt werden.

Für die Installation von Bascom unter Windows NT, Windows 2000 oder XP benötigen Sie Administratorrechte. Wenn Sie Bascom das erste Mal ausführen möchten, benötigen Sie auch Administratorrechte. Danach können Sie Bascom mit jedem User benützen.

2.3 Progammieren des Controllers mit der seriellen Schnittstelle

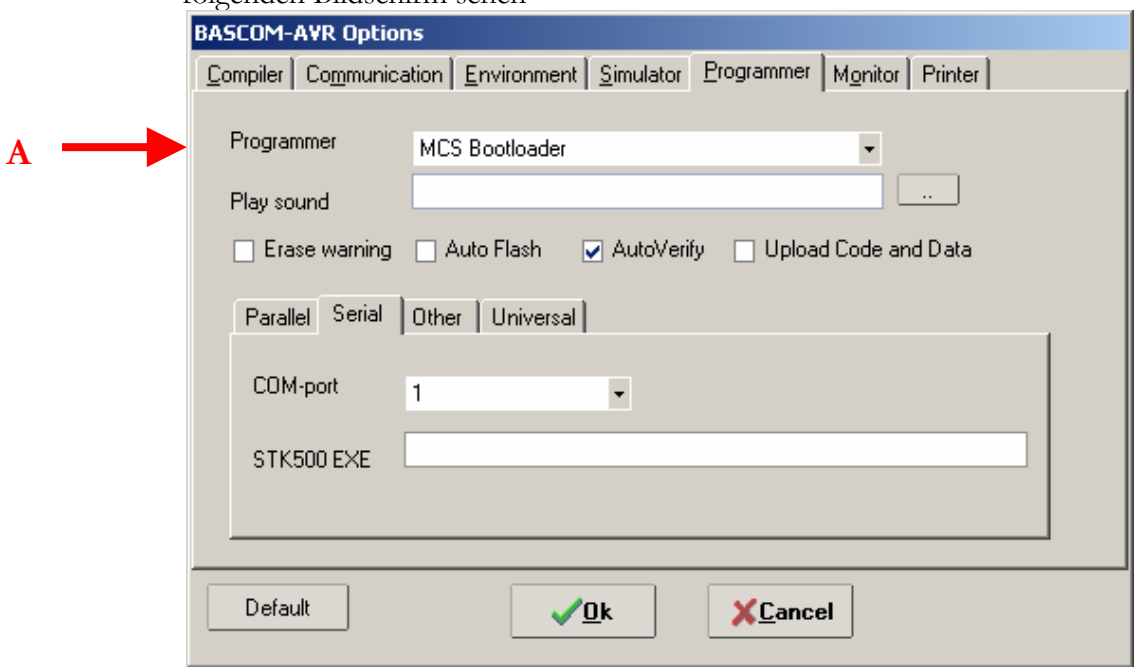
Dieses Kapitel beschreibt, wie sie Bascom konfigurieren, um den µ-Controller mittels der seriellen Schnittstelle programmieren können.

Die Methode, um den Controller mittels der seriellen Schrittstelle zu programmieren nennt sich „Bootloader“. Ein Bootloader ist ein Stück Software, dass Ihre Applikation in den Speicher des Controllers lädt. (Auch der Bootloader selber wird im Controller gespeichert, mcselectronic hat dies bereits veranlasst)

Benützen Sie das mit dem Bausatz mitgelieferte serielle Kabel für die Progammierung des EDB.
Sie können jedes ungekreuzte Kabel mit folgendem Pinout benützen: 2->2, 3->3, 4->4, 5->5.

Verbinden Sie das serielle Kabel mit dem PC und stecken Sie das andere Ende auf X16 (DB-9) mit der Bezeichnung „RS-232“ auf dem EDB.

In Bascom klicken Sie auf „Optionen“ und „Programmer“ . Nun sollten Sie folgenden Bildschirm sehen



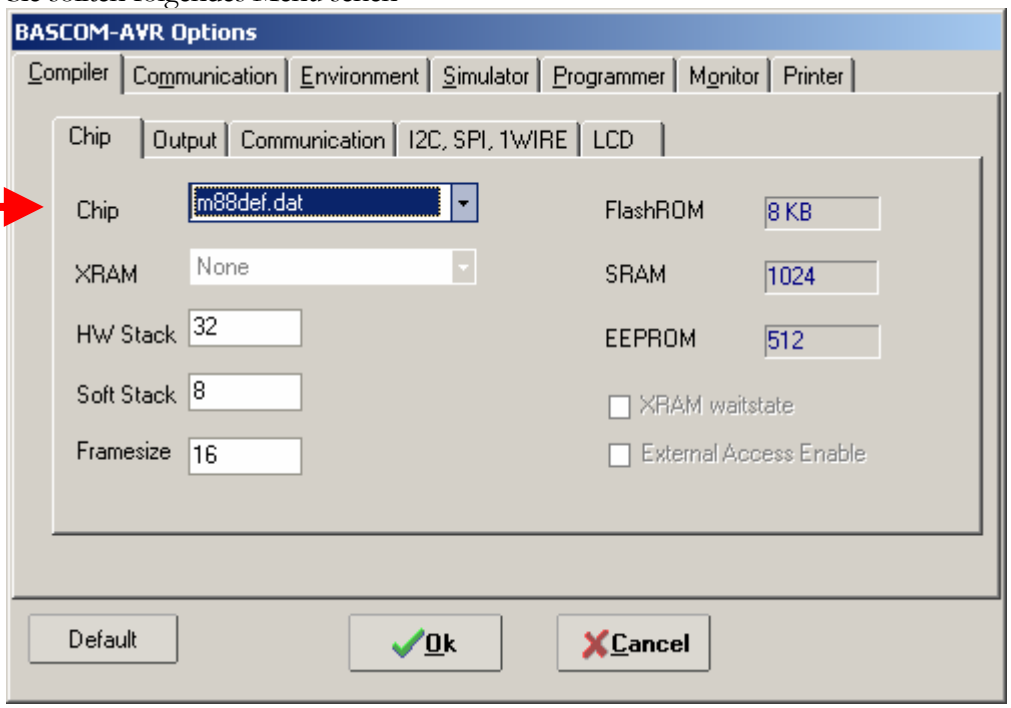
- Selektieren Sie „MCS-Bootloader“ im Auswahlmenü (A).
- Wählen Sie den zugehörigen Com-Port und klicken „ok“..

„Flashprogrammierung“

Bevor Sie den Chip programmieren können, müssen Sie zuerst das Testprogramm öffnen und kompilieren.
Das Testprogramm finden Sie auf der EDB-CD. Öffnen Sie EDBtest.Bas mit Bascom („File“ > „open“)

Öffnen Sie nun das Chip-Optionsmenü („Options“ > „Compiler“ > „Chip“)

Sie sollten folgendes Menü sehen



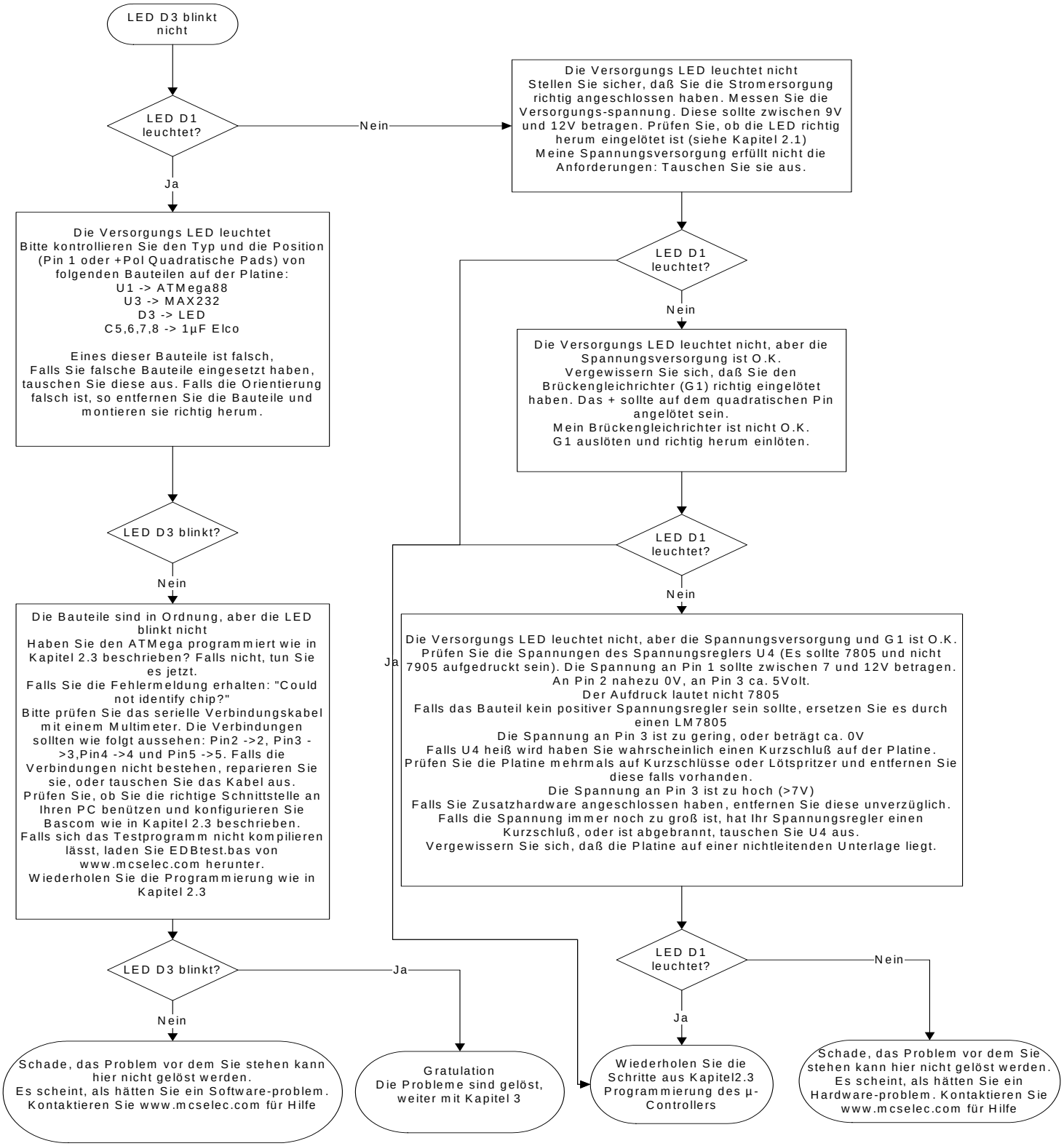
- Selektieren Sie m88def.dat im „Chip“ Auswahlmenü (A)
- Lassen Sie die anderen Optionen unverändert und klicken „ok“
- Mit F7 kompilieren Sie nun das Testprogramm
- Drücken Sie nun F4, um das Programm auf den Chip herunter zu laden

Wenn alles gut geht, sollten Sie nun eine blinkende LED sehen.

Falls Sie den ATMega88 nicht bei MCS Electronic erworben haben, oder die „fuse bits“ verändert haben sollten, lesen Sie in Kapitel 4.3 weiter, bevor Sie unseren Support kontaktieren.

● Blinkt die LED D3 (PD7)?
Falls ja, Gratulation! Weiter mit Kapitel 3.
Falls nicht, keine Sorge....weiter auf der nächsten Seite.

2.4 Fehlerbehebung



Kapitel

3

3. Versuche

In diesem Kapitel werden Ihnen die Grundlagen der Elektronik und Mikrocontrollertechnik vermittelt..

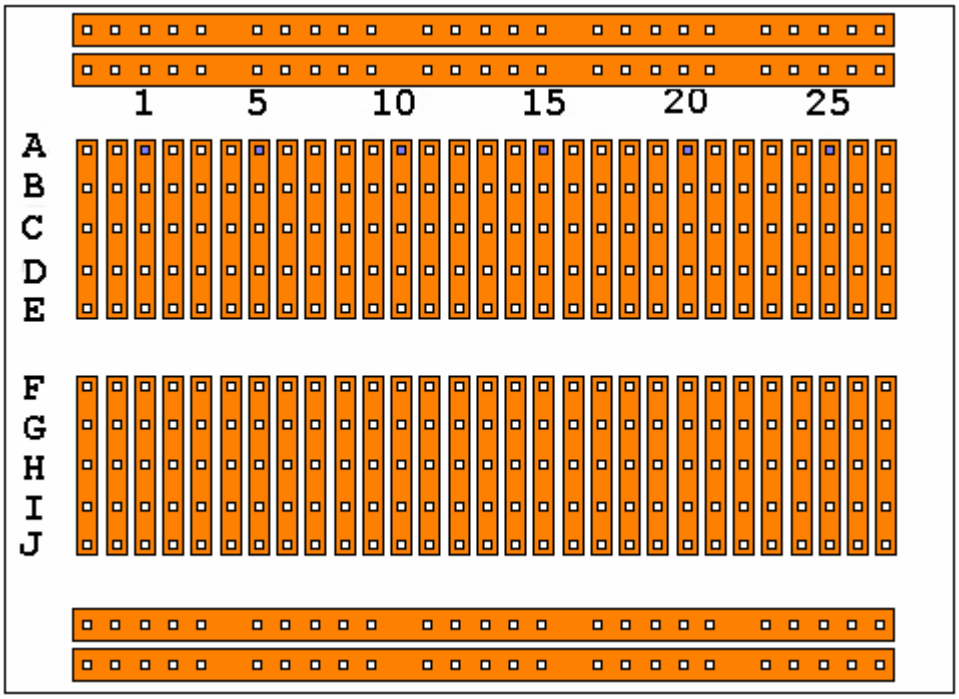
Jetzt haben Sie Ihr EDB-Board zusammen gebaut und getestet und können nun mit den Experimenten starten. Die Experimente folgen logisch, aufeinander aufbauend und werden Sie zu weiteren Versuchen anspornen.

Die Experimente sind wie folgt unterteilt:,

- Grundlagen Input/Output
- Serielle Kommunikation mit dem UART
- Displayausgabe (LCD, sowie 7-Segment)
- Tastatureingabe (PS/2, Matrixtastatur, RC5 Fernbedienung)
- Erweiterte Input/Output-Befehle
- Analog/Digital-Konvertierung
- Weitere Controller Features
- Weitere Interfaces (I^2C, USB)
- Motoren

STECKBRETT

Das Steckbrett wird benötigt, um lötfreie Verbindungen zwischen Bauteilen und Verdrahtungen in einem Experimentierstadium herzustellen.
Falls Sie solch ein Experimentalfeld noch nie benützt haben, betrachten Sie die unten dargestellte Zeichnung, die Ihnen die internen Verbindungen zeigt.



Die **kupferfarbenen** Gebiete zeigen die internen Verbindungen auf. Sie können alle Bauteile mit einem Rasterabstand von 2,54 mm einsetzen. Die Drahtstärke darf 0,3 bis 0,8 mm im Durchmesser betragen. Bauteile oder Drähte niemals mit Gewalt einsetzen.

ACHTUNG ! GEFAHR VON SCHWEREN VERLETZUNGEN ORDER TOD !

Das Steckbrett ist nicht geeignet fur Verbindungen mit dem elektrischen Netz.

Niemals verbindungen zwischen dem elektrischen netz und dem Steckbrett herstellen

3.1 Grundlegende Ein-/Ausgangsversuche I/O

I/O bedeutet Eingang und Ausgang. Eingänge werden gewöhnlicherweise mit den Eingängen des Mikrocontrollers verbunden. Ausgänge hingegen werden gewöhnlicherweise mit den Ausgängen des Controllers verbunden. (Ein bidirektionaler Pin des µC kann sowohl als Ein – wie als Ausgang konfiguriert werden). I/Os präsentieren die verschiedensten Anwendungen. Die einfachste Form ist ein LED-Ausgang und ein Schaltereingang.

Experiment 1

3.1.1 LED-Ausgang, Schalter als Eingang

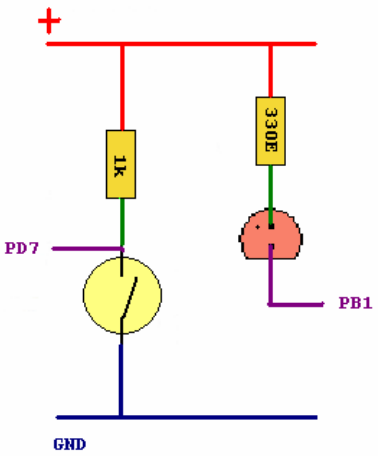
Stückliste

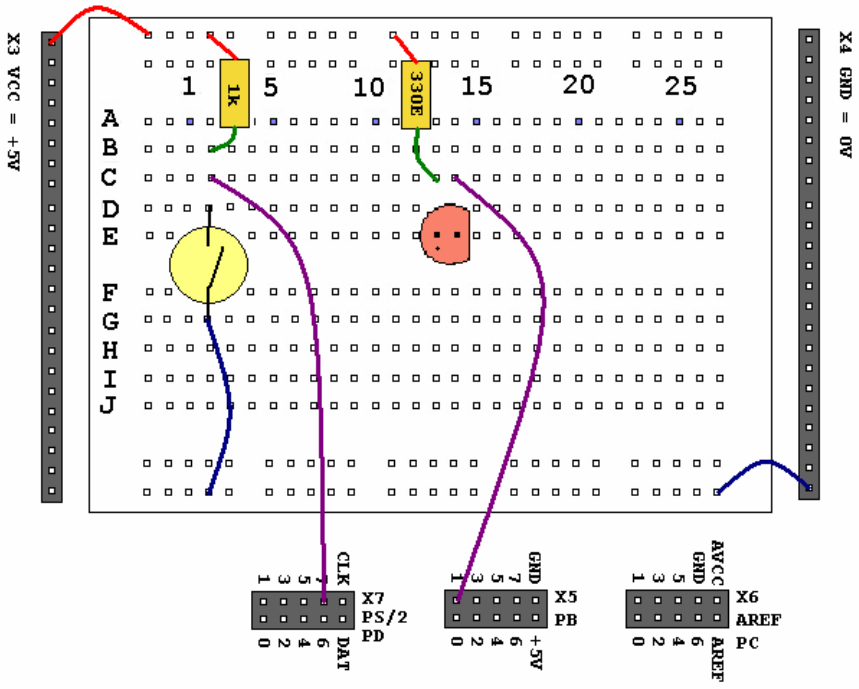
1x Widerstand 1k
1x Widerstand 330E
Ausgabe
1x LED
1x Schalter oder Draht
6x Verbindungsdrähte

Ziele

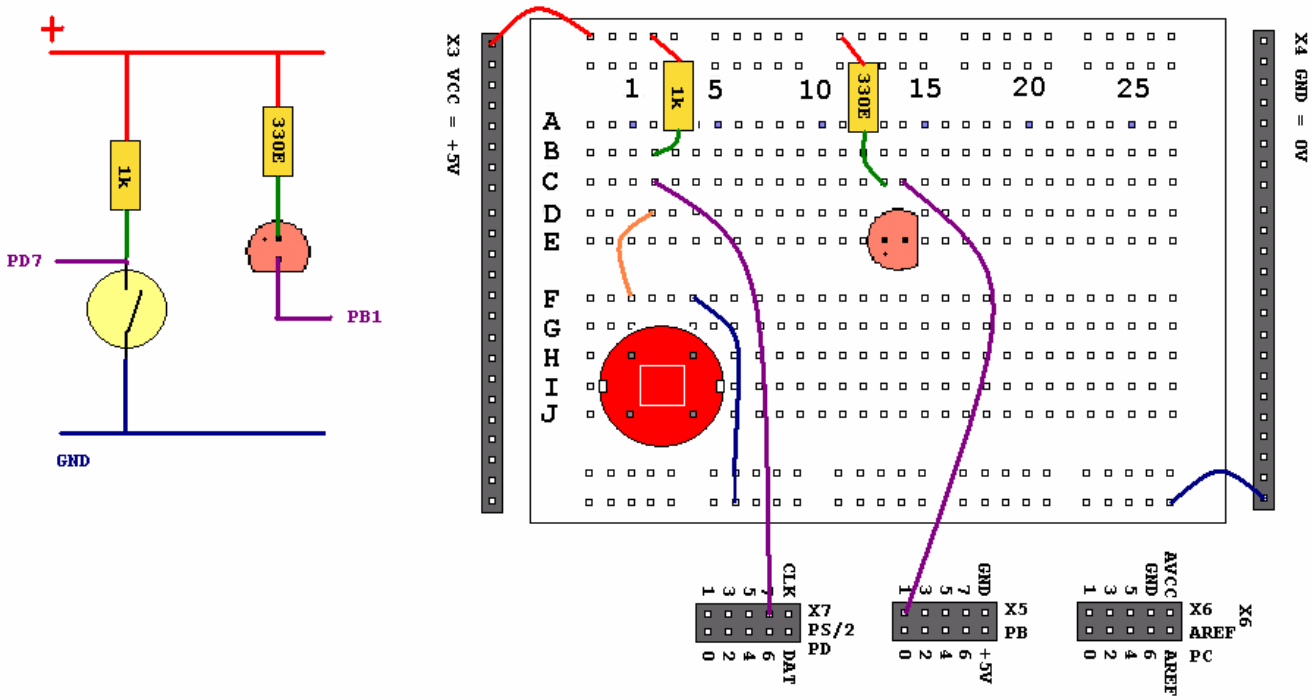
- Grundwissen Ein- /

Bauen Sie die unten gezeigte Schaltung auf. Es wird Ihnen gezeigt, wie Sie einen Schalter und eine LED mit dem Mikrocontroller verbinden. Falls Sie keinen Schalter haben, können Sie auch einen abisolierten Draht verwenden. Siehe auch vorgeschlagene Verdrahtung des Steckbrettes.





Sie können auch den durch MCS Electronic vertriebenen Taster verwenden. Dann sieht das Board wie folgt aus:



Öffnen Sie nun die EDBexperiment1.bas Datei von der EBD-CD. Es sieht folgendermaßen aus:

```
'-----------------------------------------------------------------
'                       EDBexperiment1.bas
'       Experiment 1 for the Educational Development Board
'                  (c) 1995-2005, MCS Electronics
'                       Fileversion 1.0
'-----------------------------------------------------------------
'
'Purpose:
'This program assigns the value of Pind.7 to Pinb.1.
'
'Conclusions:
'You should be able to switch the led with a key press.

$regfile = "m88def.dat"                                    'To tell Bascom which chip we use

Config Pind.7 = Input                                      'Configure Pin D7 as input
Config Pinb.1 = Output                                     'Configure Pin B1 as output


Do                                                         'Do...loop will loop forever
   Portb.1 = Pind.7                                        'Assign the output with the input value
Loop

End
```

Laden Sie diesen Code in den Controller, wie Sie es mit dem Testprogramm aus Kapitel 2.3 (unter Flashprogrammierung) gelernt haben. Testen Sie die Funktionalität des Experimentes durch Betätigen des Tasters. (Die LED leuchtet wenn man den Taster gedrückt hält).

Experiment

2

3.1.2 Lauflicht mit LEDs

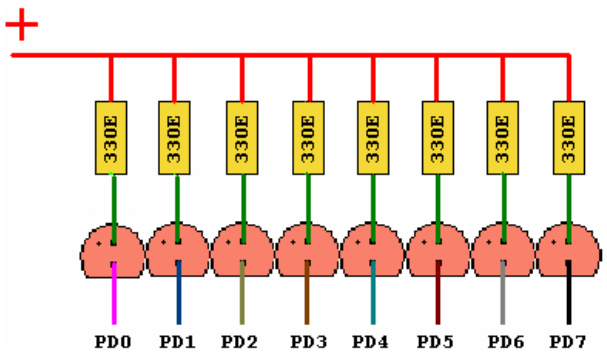
Stückliste

8x Widerstand 330E
8x LED
9x Verbindungsdrähte

Ziele

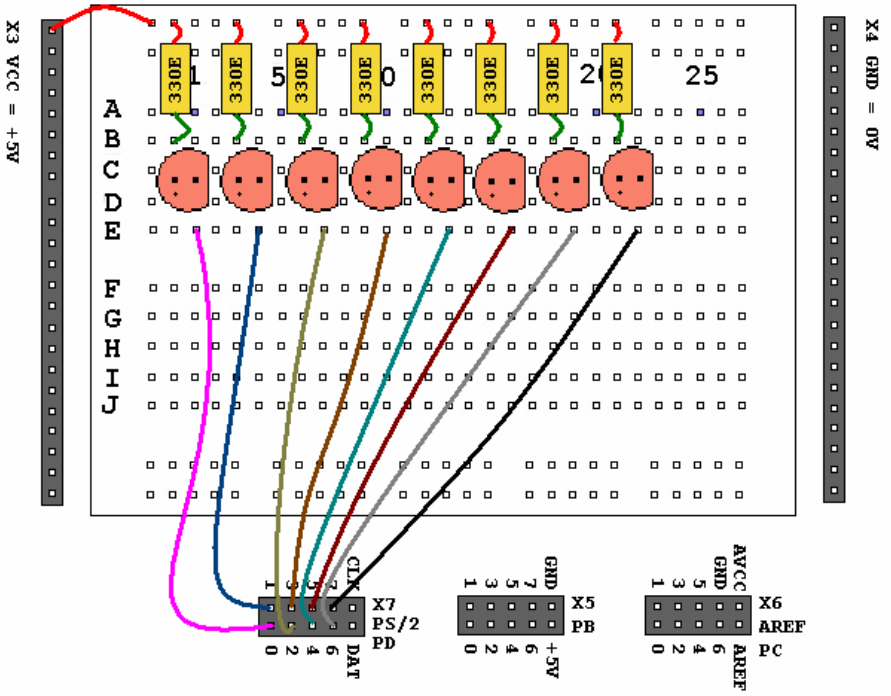
- Erlernen des Rotate Befehles
- Erlernen der Basis Input/Output

Ein weiteres Beispiel für grundlegende Input/Output Programmierung, der Rotate Befehl



Sie dürfen keine Verbindung zu GND herstellen. Die GND Versorgung erfolgt durch den Controller.

Vorschlag zum Layout des Steckbrett



Öffnen Sie nun das File EDBexperiment2.bas der EDB-CD

Es sieht wie folgt aus:

```
'-----------------------------------------------------------
'                       EDBexperiment2.bas
'       Experiment 2 for the Educational Development Board
'                  (c) 1995-2005, MCS Electronics
'                       Fileversion 1.0
'-----------------------------------------------------------
'
'Purpose:
'This program 'rotates' the LED's on port D.
'
'Conclusions:
'You should be able to see a moving LED effect.

$regfile = "m88def.dat"                                    'To tell Bascom which chip we use

Config Portd = Output                                      'Configure Pin D7 as output

Portd = &B1111110                                         'We start by making pd0 low (led on)

Do
   Rotate Portd , Right , 1                                'Move the 'bits' one place to the right
                                                           'Now PD1 will light and PD0 goes out.
   Waitms 100                                              'Wait otherwise we cannot see te rotate
Loop

End
```

Laden Sie diesen Code in den Mikrocontroller, genauso wie Sie es mit dem Testprogramm aus Kapitel 2.3 (*unter Flashprogrammierung*) getan haben. Sie sehen nun ein Lauflicht von links nach rechts.

Ändern Sie die Zeile

```
Rotate Portd , Right , 1
```
into:
```
Rotate Portd , Right , 2
```

Können Sie den Unterschied beschreiben?

Finden Sie nun selbst heraus, wie Sie die Blinkrichtung des Lauflichtes von rechts nach links ändern können. Hilfe finden Sie durch Bewegen des Cursors auf den Rotate Befehl und Drücken der F1 Taste.

Experiment

3a

3.1.3a Ausgang mit Transistor oder FET

Stückliste

1x BC547 NPN transistor
1x LED
5x Verbindungsdrähte
Transistoren und
1x Widerstand 330, 47k ohm

Ziele

- Erlernen der elektrischen Baugruppen des Controllers
- Verwendung von

FET's

Theorie

Experiment 1 und 2 verwenden s.g. „direkte IO". Das bedeutet das direkte Anschliessen der Bauteile (Schalter, LED) an den Controller. Eine einfache Lösung, die aber Nachteile hat. Erstens: es kann nur ein geringer Strom vom Controller getrieben werden. Zweitens: „direkte IO" birgt zudem die Gefahr, dass der Controller durch elektrostatische Entladung (ESD) zerstört werden kann. (ESD wird ausführlich in Kapitel 3.1.5 behandelt).

Der getriebene Strom am Pin des Controllers darf einen spezifischen Wert nicht überschreiten. Dieser Wert kann im Datenblatt des ATMega88 nachgeschlagen werden www.atmel.com Der zulässige Strom kann unter „**Electrical Characteristics**" gefunden werden. Einen Auszug hieraus finden Sie auf der nächsten Seite.

Zunächst betrachten wir die Tabelle „**Absolute Maximum Ratings**". Diese Werte beschreiben die absoluten Maximalwerte, das heisst bleibt man unterhalb dieser Werte wird der Controller auf keinen Fall zerstört. Dies sagt nichts über die Funktionalität ihrer Anwendung aus (Wird ein Strom von 40mA am Portausgang gestellt, garantiert Atmel, dass der Baustein nicht zerstört wird. Atmel garantiert nicht die Funktion des Bausteins während 40mA getrieben werden.)

Welcher Strom kann getrieben werden??
Wenn Sie einen Strom an einem Pin bei 0 Volt treiben:

1. Der Gesamtstrom der I/O Pins am Port C0...C5, ADC7, ADC6 darf 100mA nicht überschreiten.
2. Der Gesamtstrom der Port Pins B0...B5, D5...D7, XTAL1, XTAL2 darf 100mA nicht überschreiten.
3. Der Gesamtstrom der Port Pins D0...D4, Reset darf 100 mA nicht überschreiten.

Wenn Sie einen Strom an einem Pin bei 5 Volt treiben:

1. Der Gesamtstrom der I/O Pins C0...C5, D0...D4, ADC7, Reset darf 150 mA nicht überschreiten.
2. Der Gesamtstrom der I/O Pins B0...B5, D5...D7, ADC6, XTAL1, XTAL2 darf 150 mA nicht überschreiten.

All dies gilt bei einer Gesamstromaufnahme des Controllers von 200mA (absolutes Maximum). Für weitere Informationen zur Bauteilecharakteristik ziehen Sie bitte das neueste Datenblatt zu Rate. Die oben genannten Werte beziehen sich auf ein Vorankündigungsdatenblatt.

27. Electrical Characteristics

27.1 Absolute Maximum Ratings*

Operating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
Voltage on any Pin except $\overline{\text{RESET}}$ with respect to Ground	-0.5V to V_{CC}+0.5V
Voltage on $\overline{\text{RESET}}$ with respect to Ground	-0.5V to +13.0V
Maximum Operating Voltage	6.0V
DC Current per I/O Pin	40.0 mA
DC Current V_{CC} and GND Pins	200.0 mA

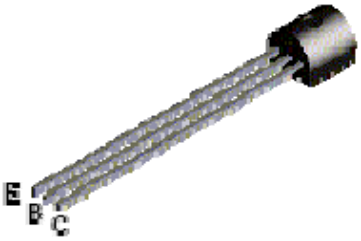
*NOTICE: Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

27.2 DC Characteristics ATmega48/88/168*

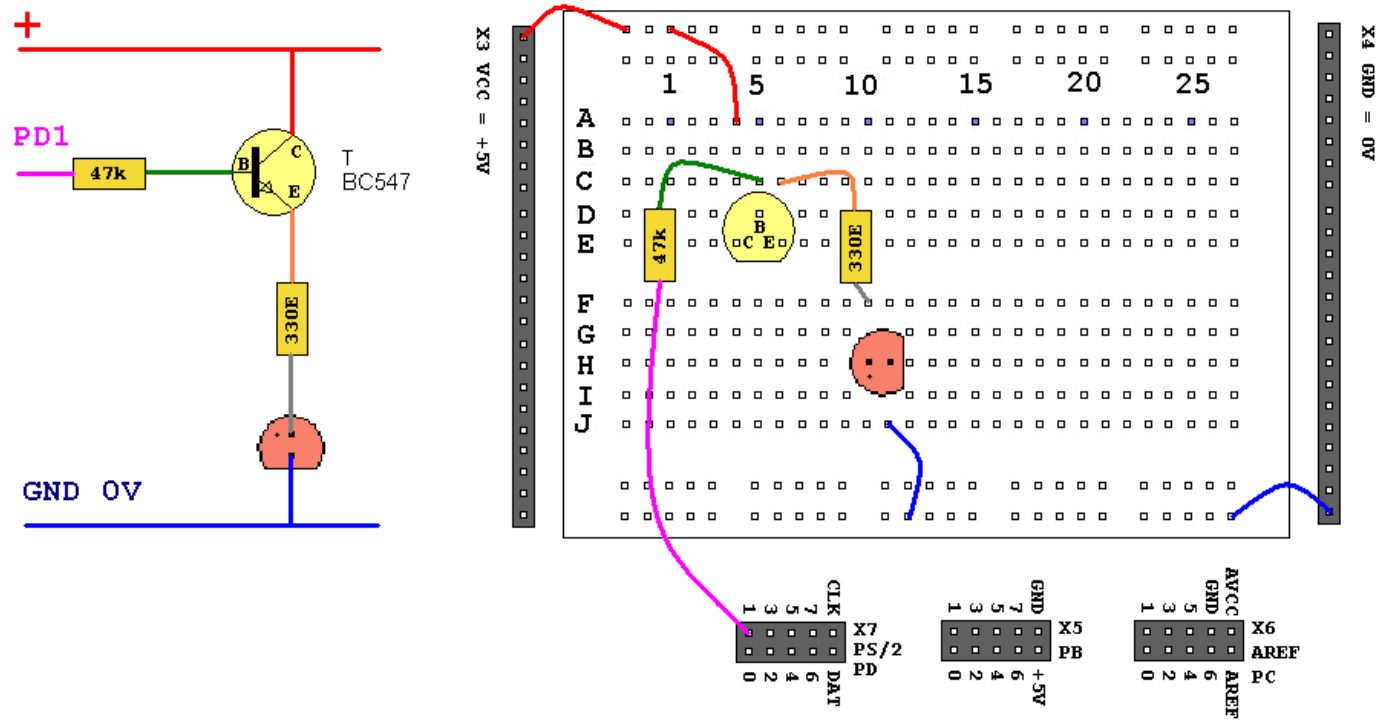
* Note: The following table contains preliminary data for the ATmega88 and ATmega168.

T_A = -40°C to 85°C, V_{CC} = 1.8V to 5.5V (unless otherwise noted)

Symbol	Parameter	Condition	Min.(5)	Typ.	Max.(5)	Units
V_{IL}	Input Low Voltage, except XTAL1 and $\overline{\text{RESET}}$ pin	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	-0.5 -0.5		$0.2V_{CC}$(1) $0.3V_{CC}$(1)	V
V_{IH}	Input High Voltage, except XTAL1 and $\overline{\text{RESET}}$ pins	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	$0.7V_{CC}$(2) $0.6V_{CC}$(2)		V_{CC} + 0.5 V_{CC} + 0.5	V
V_{IL1}	Input Low Voltage, XTAL1 pin	V_{CC} = 1.8V - 5.5V	-0.5		$0.1V_{CC}$(1)	V
V_{IH1}	Input High Voltage, XTAL1 pin	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	$0.8V_{CC}$(2) $0.7V_{CC}$(2)		V_{CC} + 0.5 V_{CC} + 0.5	V
V_{IL2}	Input Low Voltage, $\overline{\text{RESET}}$ pin	V_{CC} = 1.8V - 5.5V	-0.5		$0.2V_{CC}$(1)	V
V_{IH2}	Input High Voltage, $\overline{\text{RESET}}$ pin	V_{CC} = 1.8V - 5.5V	$0.9V_{CC}$(2)		V_{CC} + 0.5	V
V_{IL3}	Input Low Voltage, $\overline{\text{RESET}}$ pin as I/O	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	-0.5 -0.5		$0.2V_{CC}$(1) $0.3V_{CC}$(1)	V
V_{IH3}	Input High Voltage, $\overline{\text{RESET}}$ pin as I/O	V_{CC} = 1.8V - 2.4V V_{CC} = 2.4V - 5.5V	$0.7V_{CC}$(2) $0.6V_{CC}$(2)		V_{CC} + 0.5 V_{CC} + 0.5	V
V_{OL}	Output Low Voltage(3), $\overline{\text{RESET}}$ pin as I/O	I_{OL} = 20mA, V_{CC} = 5V I_{OL} = 6mA, V_{CC} = 3V			0.7 0.5	V
V_{OH}	Output High Voltage(4), $\overline{\text{RESET}}$ pin as I/O	I_{OH} = -20mA, V_{CC} = 5V I_{OH} = -10mA, V_{CC} = 3V	4.2 2.3			V
V_{OL3}	Output Low Voltage(3), $\overline{\text{RESET}}$ pin as I/O	TBD			TBD	V
V_{OH3}	Output High Voltage(4), $\overline{\text{RESET}}$ pin as I/O	TBD	TBD			V



Sie können einen Transistor benützen, wenn Sie einen grösseren als den durch den Controller zulässigen Strom benötigen. Die Zeichnung unten zeigt, wie man einen Transistor als Schalter verwendet.



Öffnen Sie das File EDBexperiment3.bas der EDB-CD und laden Sie das Programm in den ATMega88.

Online Datenquelle:
BC547: http://www.fairchildsemi.com/pf/BC/BC547.html
ATMega88:
http://www.atmel.com/dyn/products/product_card.asp?part_id=3302

Experiment

3b

3.1.3b Augang mit FET

Stückliste

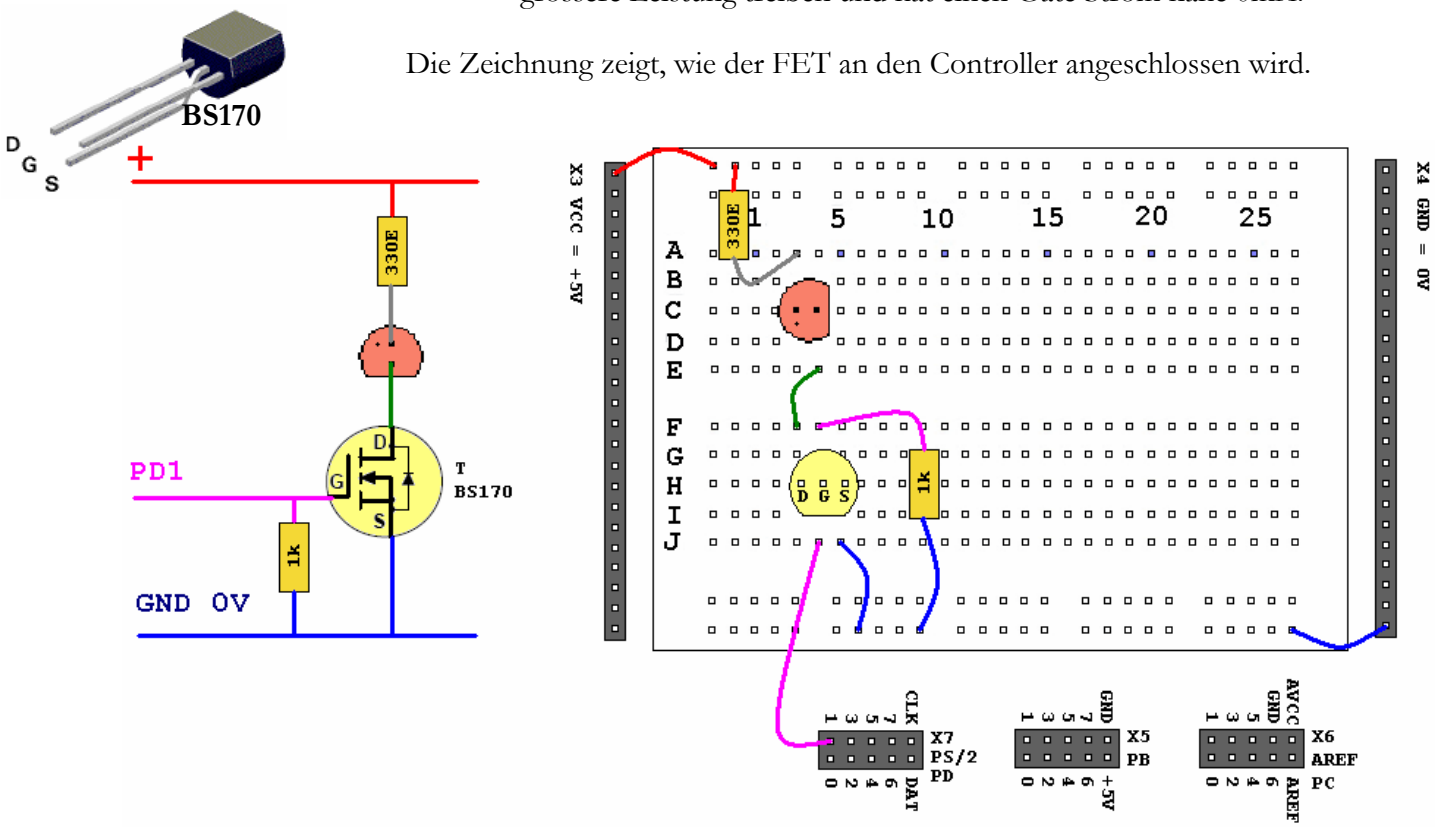
1x BS170 N-channel enh. FET
1x LED
5x Verbindungsdrähte
1x Widerstand 330, 47k ohm

Ziele

- Verwendung von FET's

Die Verwendung eines MOSFET anstelle eines „klassischen" BC547 Transistors hat einige Vorteile. Ein MOSFET kann eine grössere Leistung treiben und hat einen Gate Strom nahe 0mA.

Die Zeichnung zeigt, wie der FET an den Controller angeschlossen wird.



Sie können das Programm aus Versuch 3a weiter verwenden. Falls Sie es nicht haben, öffnen Sie das File EDBexperiment.3.bas von der EDB-CD und laden Sie es in den ATMega88. Belassen Sie die Bauteile auf dem Steckbrett.

Online Datenquelle:

BS170: http://www.fairchildsemi.com/pf/BS/BS170.html

Versuch 3c

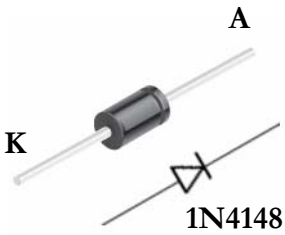
3.1.3c Ausgang mit Relais

Stückliste

1x BS170 N-Kanal FET
wirden
1x Ralais (Schrack PE014-005)
7x Verbindungsdraht
1x Widerstand 1kΩ
1x Diode 1N4148

Ziele

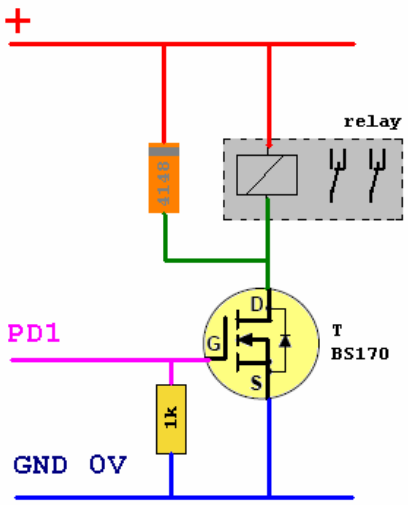
- Lernen wie/warum Relais benützt

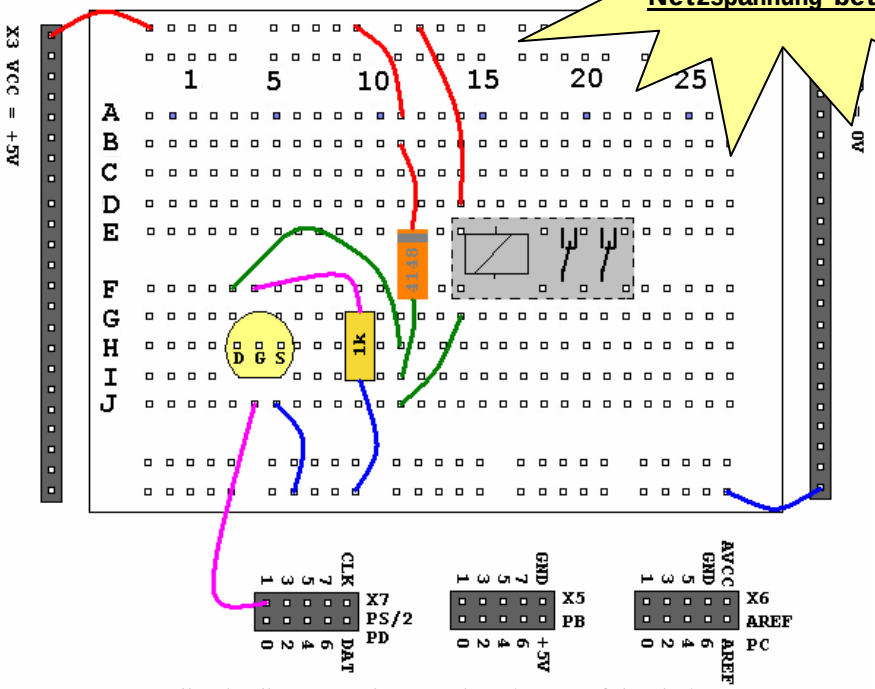


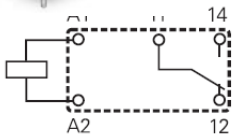
Mund Transistoren sind beide Halbleiter. Die in den letzten Versuchen verwendeten können nur Gleichspannung schalten. Ein Relais kann man als potenzialfreien Kontakt verwenden. Dieses gibt Ihnen die Möglichkeit, damit Wechselspannungen, Netzspannung zu schalten..

Sie benötigen einen FET, um das Relais zu schalten....

Bitte NIEMALS das Versuchsbrett mit Netzspannung betreiben







Falls Sie die Versuche 3a oder 3b ausgeführt haben, brauchen Sie den ATMega88 nicht reprogrammieren. Falls nicht, öffnen Sie das File EDBexperiment3.bas von der EDB-CD und laden es in den ATMega88.

Online Datenquelle

Relay: http://relays.tycoelectronics.com/

Versuch
4

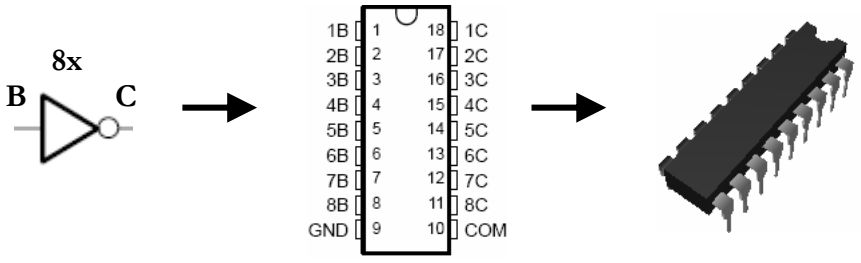
3.1.4 Ausgang mit ULN2803

Stückliste
1x ULN2803 oder Vergleichstyp
2x LED
7x Verbindungsdraht
2x Widerstand 330Ω

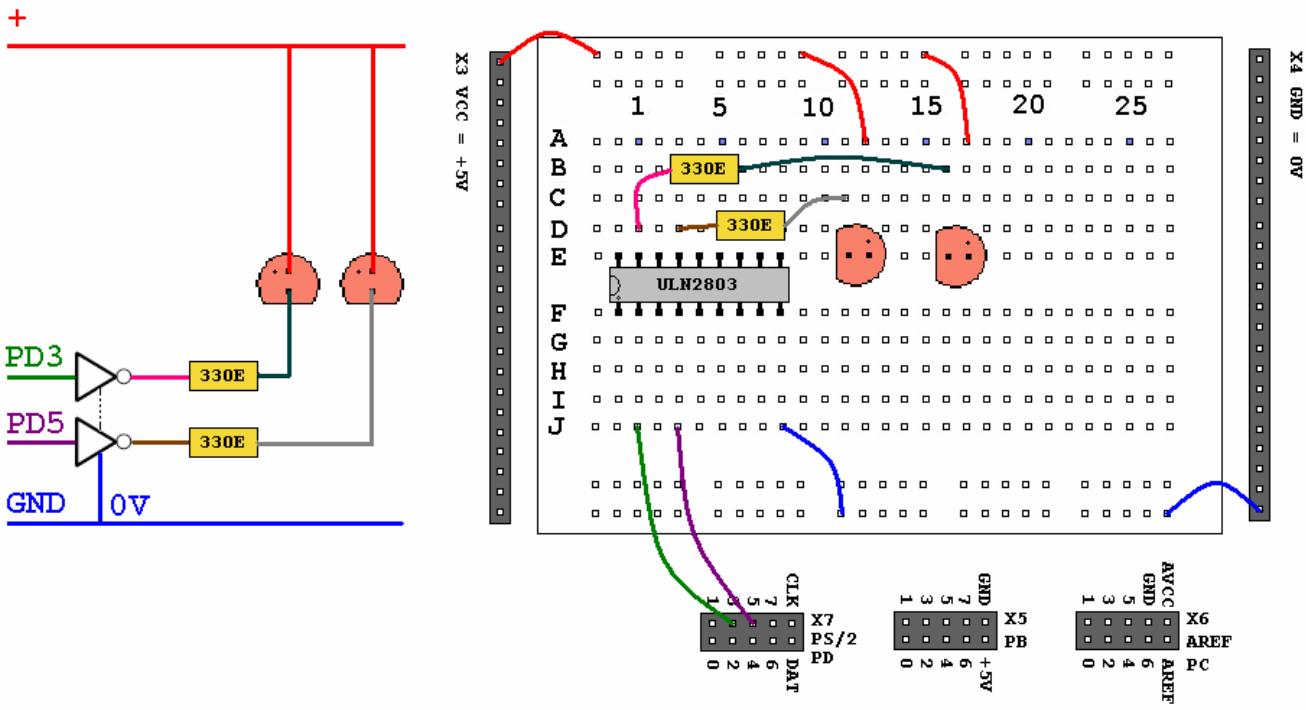
Ziele
- Umgang mit dem ULN2803

In vielen Anwendungen reicht ein einfacher Ausgang nicht aus. Stellen Sie sich die grosse Menge an Bauteilen vor, wenn Sie 7 oder 8 Transistorausgänge benützen wollen! (siehe Kapitel 3.1.3a) Dies wäre nicht nur teuer, sondern auch unsicher.

Um dieses Problem zu lösen, kann man ein Transistor-Array im Dual-In-Line Gehäuse kaufen. Der UNL2803 hat 8 eingebaute „Ports“ von denen jeder 30mA treiben kann



Bauten Sie unten gezeigtes Layout auf:



Öffnen Sie EDBexperiment4.bas von der EDB-CD und laden Sie das Programm in den ATMega88.

In diesem Versuch haben nur zwei Ports des UNL2803 verbunden. Natürlich wäre es möglich, 8 LEDs anzuschliessen. Sie können auch Relais anschliessen, vergewisserns Sie sich, dass sie nicht zuviel Strom ziehen. Mehr Informationen hierzu finden Sie im Datenblatt.

Online Datenquelle

ULN2803: http://focus.ti.com/docs/prod/folders/print/uln2803a.htm

Versuch
5

3.1.5 I/O mit Optokoppler

Stückliste

1x PC817 oder SFH601 oder Vergleichstyp Optokopplern
1x LED
7x Verbindungsdraht
1x Widerstand 330Ω, 820Ω

Ziele

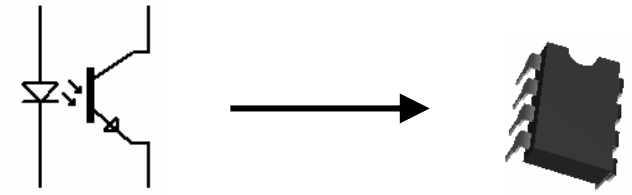
- Umgang mit ESD
- Was versteht man unter

Theorie
ESD

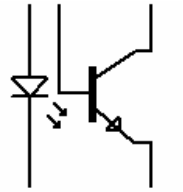
Haben Sie schon einmal einen elektrischen Schlag bekommen, wenn Sie etwas oder jemanden berührt haben? Dann haben Sie Erfahrung mit elektrostatischer Aufladung (ESD). Halbleiter sind sehr empfindlich und können durch ESD zerstört werden. ESD wird nicht nur durch Menschen erzeugt, und kann nicht immer gesehen, oder gefühlt werden.

ESD sichere Elektronikschaltungen zu entwerfen ist nicht einfach und es liegt weit hinter dem Fokus dieses Handbuches. Wir werden Ihnen jedoch zeigen, wie man eine galvanische Trennung mit einem Optokoppler erzielt.

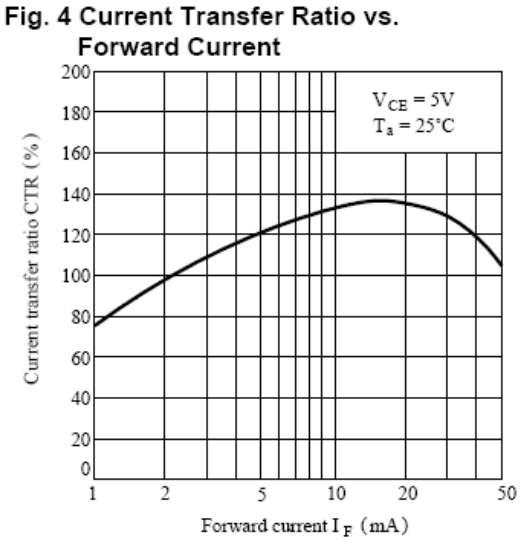
Ein Optokoppler ist ein spezieller Transistor mit einer eingebauten LED. Der Transistor braucht keinen Basisanschluss. Wenn ein Strom durch die LED fliesst wird Licht emittiert, dieses Licht erzeugt den Basisstrom an dem speziellen Transistor. Dies wird im folgendem gezeigt:

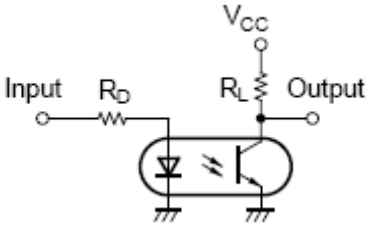


Es gibt auch Typen, die einen Basis Pin haben. In diesem Fall kann der Transistor mittels der LED oder des Basisanschlusses betrieben werden



Für diesen Versuch wird der PC817, oder der SFH601 oder ein anderer Optokoppler verwendet. Angenommen, Sie verwenden den PC817, sehen Sie im Datenblatt nach, dass Sie auf der EDB-CD finden. Im Bild Nummer 4 wird gezeigt, welcher Strom angeschlossen werden muss.

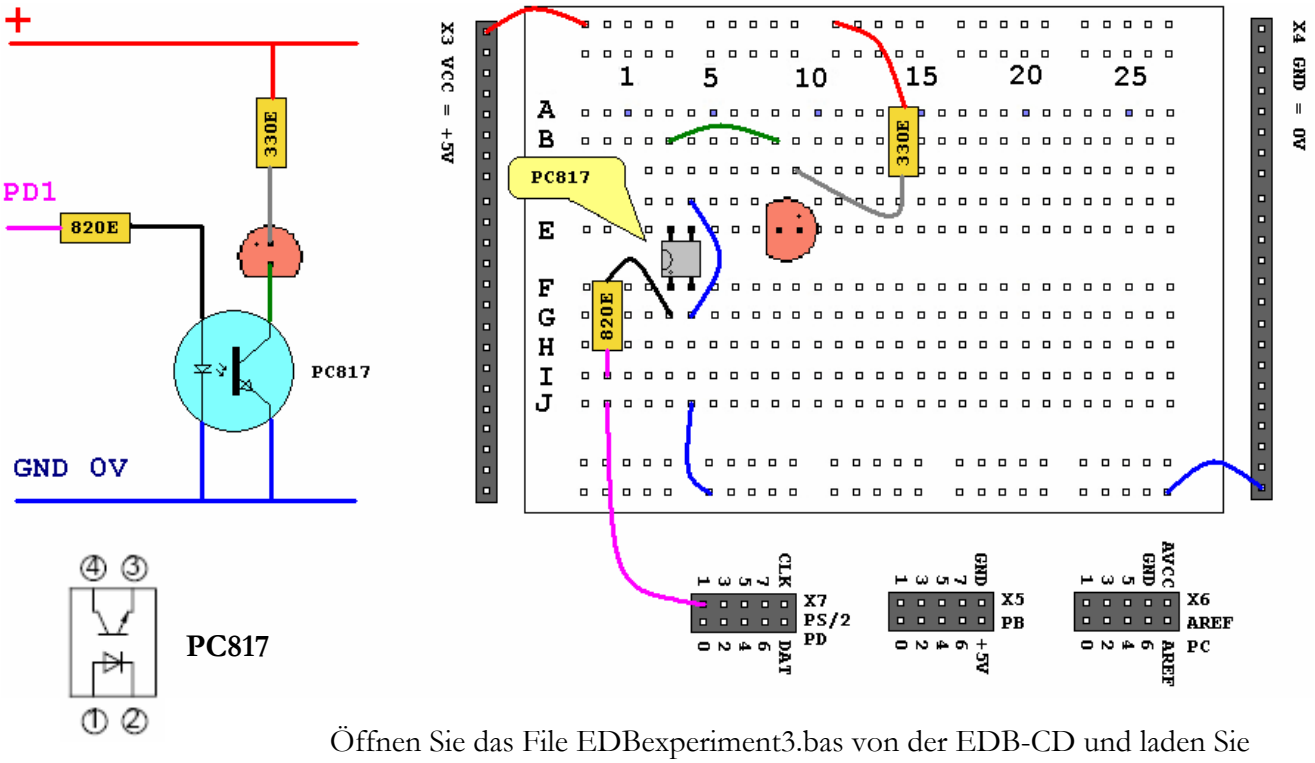




Um auf der sicheren Seite zu sein, ist es besser, einen Strom > 5mA fliessen zu lassen. Nehmen wir an, der Spannungsabfall über die LED beträgt 1,3V (Vorwärtsspannung) und der Versorgungsspannungspegel betrage 5V. Der Spannungsabfall an Rd beträgt demnach 5 – 1,3 = 3,7V. Der Widerstand Rd berechnet sich wie folgt: 3,7V / 5mA = 740Ω.

Ein Widerstand von 740Ω findet sich nicht in der E24 Normreihe. Der nähest mögliche Widerstand beträgt 820Ω, was einem Strom von 4,5mA entspricht, oder 120%. Das ist so in Ordnung.

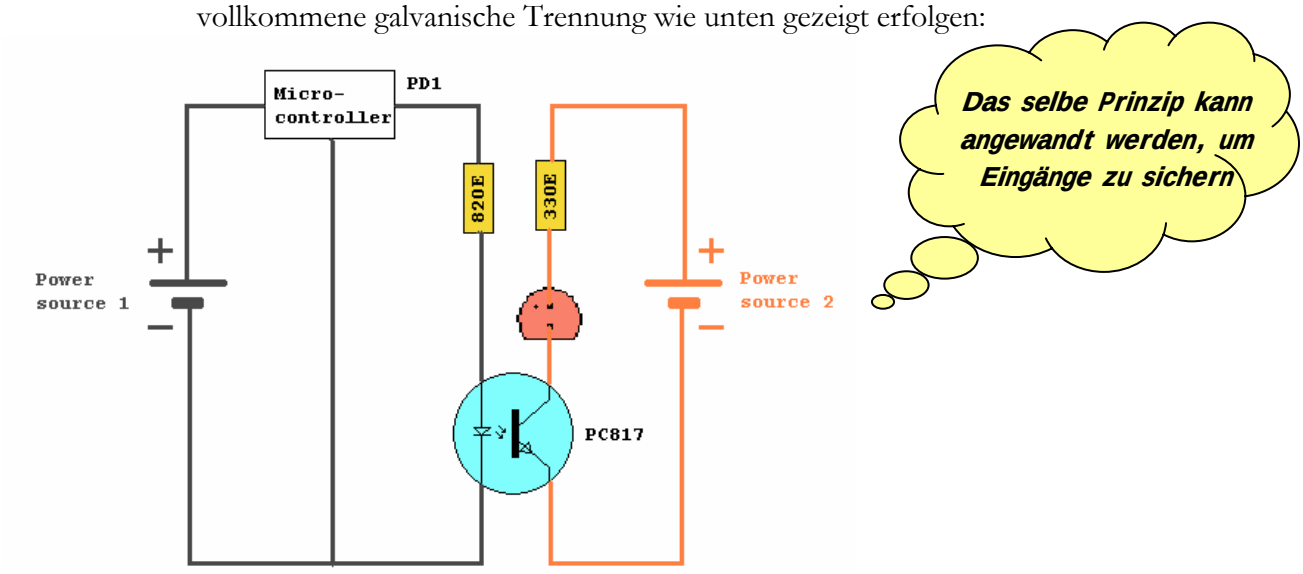
Online Datenquelle:
SFH601: http://www.vishay.com/optocouplers/list/product-83663/



PC817

Öffnen Sie das File EDBexperiment3.bas von der EDB-CD und laden Sie das Programm in den ATMega88.

Wie Sie sehen, blinkt die LED wieder. ESD kann an allen elektrisch leitendem Material auftreten. Somit ist der oben gezeichnete Pin PD1 ESD sicher. Aber ESD kann auch über die Versorgungsleitungen (GND, VCC) auftreten und den Controller zerstören. Um diesem vorzubeugen muss eine vollkommene galvanische Trennung wie unten gezeigt erfolgen:



Betrachten Sie den Schaltplan, bauen Sie ihn aber nicht nach.

3.2 Serielle asynchrone Datenkommunikation (UART)

Wie haben uns nun die Grundlagen der Ein- und Ausgabe angeschaut, mit der wir eine Menge an einfachen Problemen lösen können. Wir können durch den Anwender einen Knopf drücken lassen und dies mit einer LED bestätigen. Aber was, wenn ein Benutzer seinen Namen eingeben und diesen dann am Display lesen möchte?

In diesem Kapitel wird Ihnen der populärste Weg der Datenkommunikation in der Welt der Mikrocontroller vorgestellt.

Versuch
6a

3.2.1 Der UART

Stückliste
1x RS232 Kabel
Datenverbindung
1x PC mit serieller Schnittstelle

Ziele
- Benützung der seriellen

UART bedeutet universeller, asynchroner Empfänger und Sender. Sie können den UART dazu benützen, um Daten zwischen dem Ihrem PC und dem EDB zu senden und empfangen.

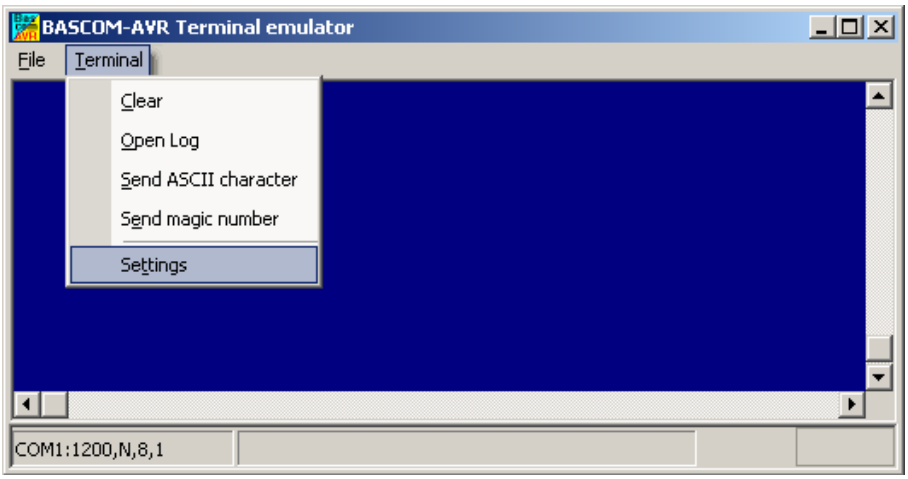
Ein UART sieht sehr ähnlich aus, wie der die serielle Schnittstelle auf der Rückseite Ihres PCs. Der grosse Unterschied ist, dass Ihre serielle Schnittstelle RS232/V24 Signalpegel (+15 und –15V), während der UART TTL-Pegel (5V) oder LVTTL (3V oder weniger) benützt. Um den UART mit dem PC betreiben zu können benützen wir einen MAX232 Pegelwandler (U3). Der MAX232 ist ein integrierter Schaltkreis, der Spannungen erzeugt, um sie zur Kommunikation mit der seriellen Schnittstelle der PCs zu nutzen.

Für dieses Experiment benötigen Sie ein ungekreuztes, serielles Datenkabel zwischen X16 (der SUB-D9 Steckverbinder, bezeichnet als „RS-232“) auf dem EDB und der seriellen Schnittstelle an Ihrem PC.

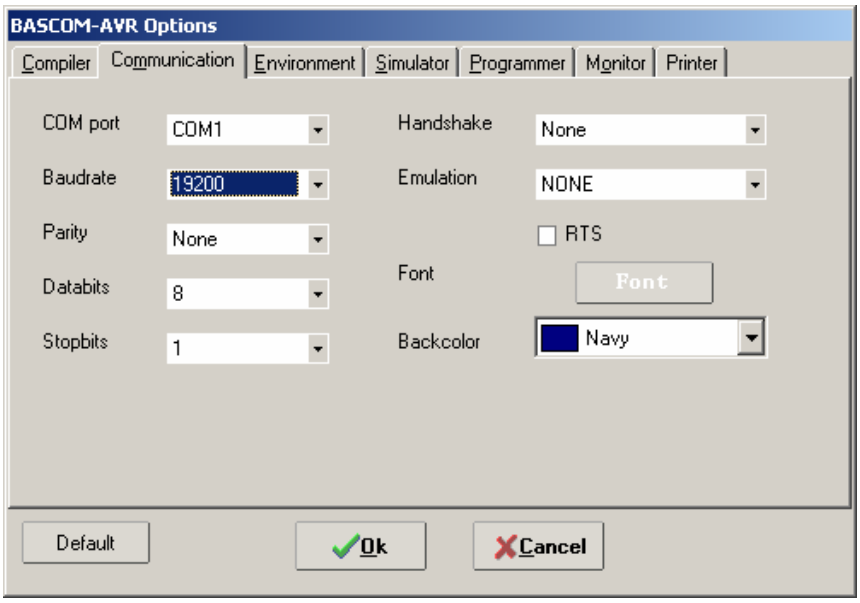
Das Kabel sollte folgende Verbindungen haben:
Pin2 nach Pin2
Pin3 nach Pin3
Pin4 nach Pin4 und
Pin5 nach Pin5
Vergewissern Sie sich, dass der Schalter „S1 USB/RS232“ auf dem EDB Board sich in Stellung RS232 befindet.
Nachdem Sie die Kabelverbindung hergestellt haben, starten Sie BASCOM

und öffnen Sie den integrierten Terminalemulator durch auf  zu Klicken.

Nun sollten Sie folgendes sehen::



Klicken Sie auf „Terminal“ und „Settings“



Wählen Sie den COM Anschluss, den Sie an Ihrem PC benützen
Wenn Sie einen PC ohne COM Anschluss haben, können Sie einen USB nach Seriell- Adapter kaufen, und ihn als einen virtuellen COM Anschluss benü1tzen.

Schließen Sie das Terminal Emulator Fenster.

Wie schon gesagt, benutzen wir einen UART, um Daten zwischen dem Mikrocontroller und dem PC zu senden und zu empfangen. Lassen Sie uns die Richtung der Daten wie folgt definieren:

Vom EDB aus gesehen:
Daten senden bedeutet: Daten vom EDB zum PC und
Daten empfangen bedeutet: Daten vom PC zum EDB.

Grundsätzlich sehen die Daten gleich aus, wir müssen sie jedoch spezifizieren, damit Sie verstehen können, was mir meinen. Die Daten können Variablen, Registerwerte, Konstanten (Text oder Zahlen) oder Rechenergebnisse sein. Darüber hinaus können Sie den Controller veranlassen eine Zeile zu drucken, die anzeigt welcher Teil Ihres Programmes in der debug Phase gerade ausgeführt wird.

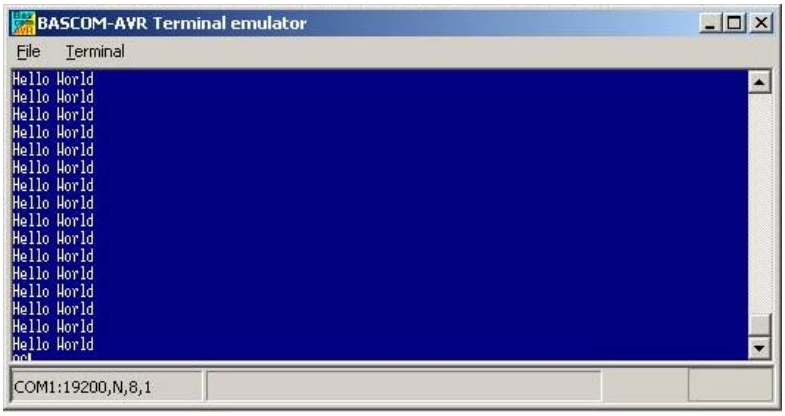
Daten können auch von der Tastatur gesendet werden, die es dem Endbenutzer erlauben eine Mitteilung einzugeben, oder ein Menu benützen zu lassen. Wir werden das später erklären.

Versuch 6a

Öffnen Sie die Datei EDBexperiment6a.bas von der EDB CD und laden Sie diese in den ATMega88. Bitte lesen Sie die Anmerkungen, die Sie in der EDBexperiment6a.bas finden. Dort erfahren Sie, wie Sie den UART initialisieren und benützen.

Nachdem Sie den Controller programmiert, und das serielle Kabel angeschlossen haben öffnen Sie den ?Terminal Emulator, in dem Sie auf den Knopf im Bascom klicken  im Bascom.

Das EDBexperiment6a.bas Programm zeigt nun auf dem Bildschirm Ihres PCs „Hello World“ an, wie in der nachfolgenden Abbildung gezeigt.



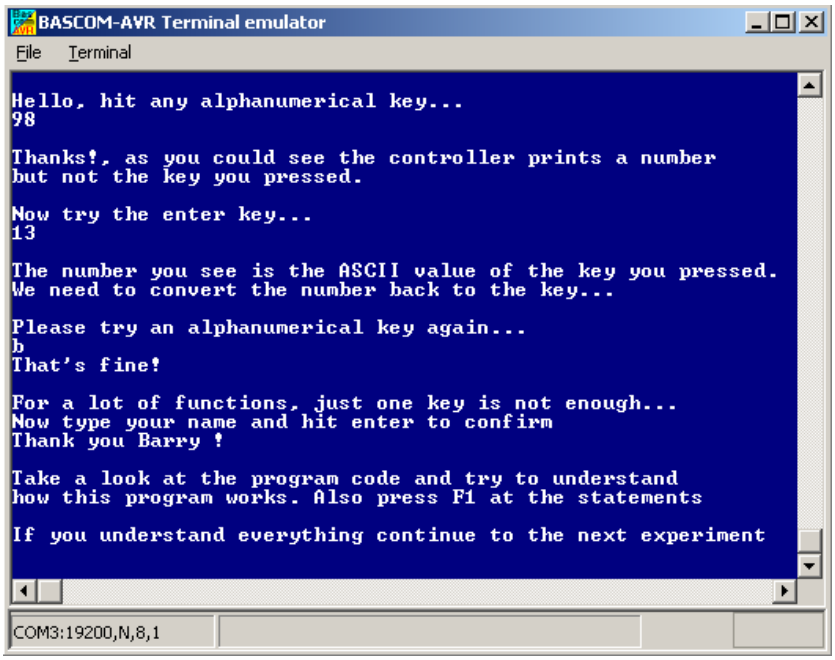
Versuch
6b

Versuch 6b

Stellen Sie sicher, dass der Schalter „S1 USB/RS232“ auf dem EDB auf RS232 gestellt ist.

Öffnen Sie die Datei EDBexperiment6b.bas von der EDB CD und laden Sie diese in den ATMega 88. Nachdem Sie den Controller programmiert, und das serielle Kabel angeschlossen haben öffnen Sie den ?Terminal Emulator, in dem Sie auf den  Knopf im Bascom klicken.

Folgen Sie den Anweisungen, die Ihnen vom EDBexperiment6b.bas Programm auf dem Bildschirm vorgegeben werden. :



Anmerkung: Über RS232 können auch zwei Pcs oder zwei Mikrocontroller miteinander verbunden werden.

Auf der nächsten Seite finden Sie noch mehr Information hierrüber.

Online Datenquelle:
UART Hardware
MAX232: http://www.maxim-ic.com/quick_view2.cfm/qv_pk/1798

ASCII

Wie Sie der Datei EDBexperiment6b.bas sehen konnten, benutzen wir den „PRINT“ Befehl, um etwas zum UART zu senden. Eigentlich senden wir nicht einfach Text. Wir senden ASCII Zeichen. ASCII ist die Abkürzung von American Standard Code for Information Interchange. Grundsätzlich besteht der ASCII Zeichensatz aus 127 Zeichen.

```
ASCII Tabelle (nicht komplett)
Decimal  Hex    Binary     Value
-------  ---    ------     -----
  000    000   00000000      NUL    (Null char.)
  008    008   00001000       BS    (Backspace)
  009    009   00001001       HT    (Horizontal Tab)
  010    00A   00001010       LF    (Line Feed)
  012    00C   00001100       FF    (Form Feed)
  013    00D   00001101       CR    (Carriage Return)
  048    030   00110000        0
  049    031   00110001        1
  065    041   01000001        A
  066    042   01000010        B
```

Sie finden eine komplette ASCII-Tabelle im Anhang 2.

Carriage Return (CR, Wagenrücklauf) und Line Feed (LF, Zeilenvorschub)

Wie Sie der Datei EDBexperiment6b.bas sehen konnten, benutzen wir den „PRINT“ Befehl, um etwas zum UART zu senden. Sie konnten auch sehen, dass ein zweiter PRINT-Befehl immer den angezeigten Text in die darauffolgende Zeile ausgibt. Das liegt darin begründet, dass der PRINT-Befehl immer ein CR- und ein LF-Zeichen hinzufügt.

Wenn wir den Befehl
```
Print "ABC"
```
Schreiben, passiert grundsätzlich folgendes:
Wir senden 65 66 67 13 10 zum UART (Im Binärformat)

Das Carriage Return-Zeichen (13) bewegt den Cursor zur Spaltenposition 0 der aktuellen Zeile. Das Line-Feed-Zeichen (10) bewegt den Cursor in die nächste Zeile

```
Print "ABC" ;
```
Wenn Sie ein Semikolon (;) am Ende der Zeile schreiben...
Sendet Bascom kein Carriage Return/Line Feed. Sie können einen anderen Text nach dem ABC in der selben Zeile anzeigen lassen.

```
Print "ABC" ; Chr(13) ;
```
Dies würde lediglich ABC CR senden. Die nächste Anzeige würde ABC überschreiben.

Versuch

6c

3.2.2 Der Software UART

Stückliste

1x RS232 Kabel
1x PC mit serieller Schnittstelle

Ziele

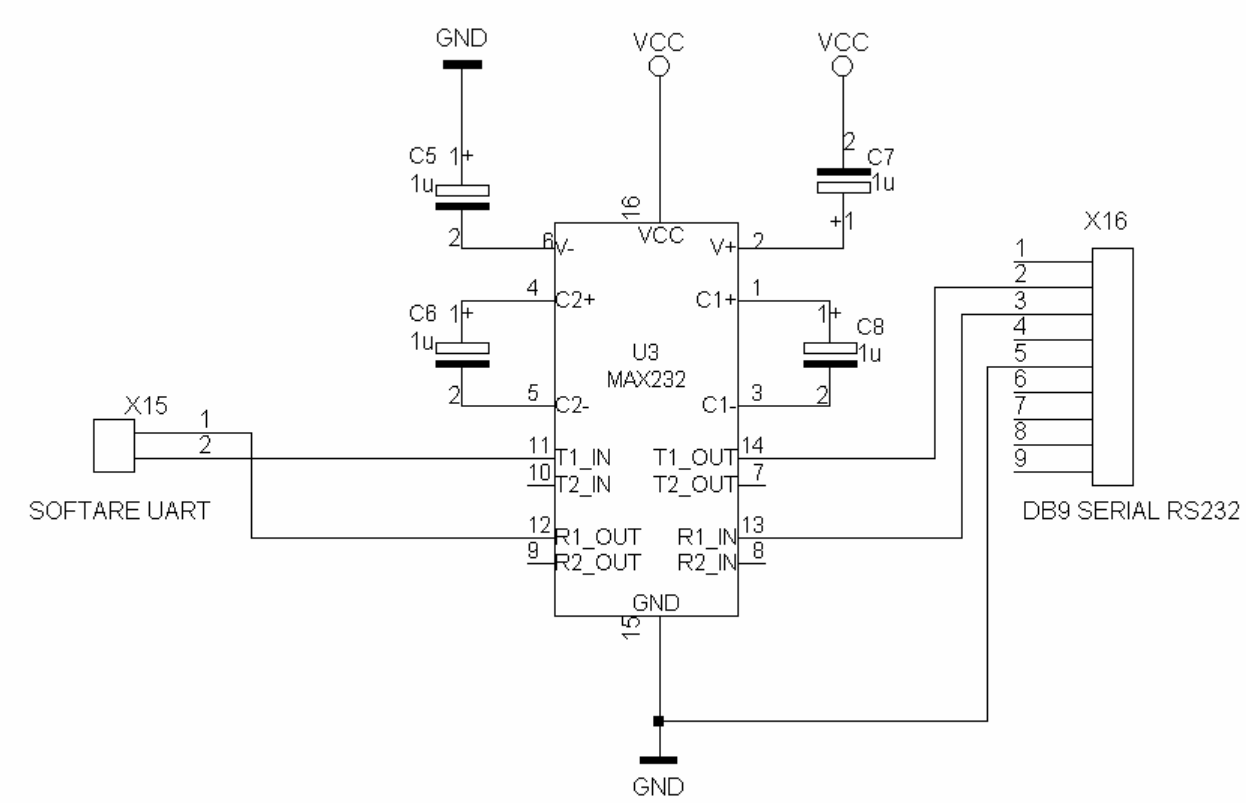
- Benützung des Software UART Datenverbindung

Die vorhergehenden Beispiele nutzten den Hardware UART. Das bedeutet, der Computer benutzt die internen UART Register und die interne Hardware (portd.0 und portd.1) des ATMega88. Manchmal möchte man jedoch mehr als einen UART oder USB und RS232 gleichzeitig nutzen (USB wird in Kapitel 3.8.2 ff erklärt).

Der Bascom Compiler macht es uns einfach, weitere UARTs zu „erstellen" Bascom kann Software-UARTs an buchstäblich jedem einzelnen Anschluss-Pin erstellen.

Stellen Sie sicher, dass der Schalter „S1 USB/RS232" auf dem EDB auf USB gestellt ist.
Wir werden jetzt nicht das USB-Interface nutzen, aber wenn der Schalter auf USB gestelt ist, werden die Hardware-UART-pins portd.0 und portd.1 vom MAX232 Pegelwandler getrennt.

Mit auf USB gestelltem S1-Schalter, stellt sich der Pegelwandler wie folgt dar:



Was wir nun tun müssen ist, die Port Pins welche wir als Software-UART verwenden wollen, mit X15 zu verbinden.

Verbinden Sie den X15-Pin der sich am nächsten zu S1 befindet mit portc.1
Verbinden Sie den anderen Pin von X15 mit portc.2

Anmerkung: wenn Sie den Bootloader nutzen, um Ihr EDB zu laden, müssen Sie Schalter S1 (USB/RS232) während der Programmierung auf RS232 setzen. Um den Software-UART zu testen müssen Sie ihn zurück auf USB setzen. (Drücken Sie den Reset Knopf nachdem Sie auf USB geschaltet haben. Auf diese Weise verpassen Sie keine Meldung im Terminalfenster)

Öffnen Sie die Datei EDBexperiment6c.bas von der EDB CD und laden Sie diese in den ATMega 88. Lesen Sie bitte die Kommentare in der Datei EDBexperiment6c.bas. Sie erläutern die Initialisierung und Nutzung des Software-UART.

Nachdem Sie den Controller programmiert und das serielle Kabel angeschlossen haben, öffnen Sie den Terminal Emulator, in dem Sie auf den  Knopf im Bascom klicken.

Sie sollten nun sehen, wie das Programm EDBexperiment6c.bas auf eine alphanumerischen Eingabe wartet. Ihre Eingabe sollte wieder als Ausgabe ins Terminalfenster zurückgeschrieben werden.

3.3 Ausgabe auf ein Display

Nicht immer will man einen PC nutzen, um etwas anzeigen zu lassen. Sagen wir mal, Sie möchten lediglich ein paar Zeilen anzeigen. Zu diesem Zweck haben wir eine bessere Lösung, eine LCD-Anzeige (Display).

3.3.1 HD44780 LCD

Versuch
7

Stückliste
1x HD44780 LCD
1x PC mit serieller Schnittstelle

Ziele
- Benützung eines LCD-Displays

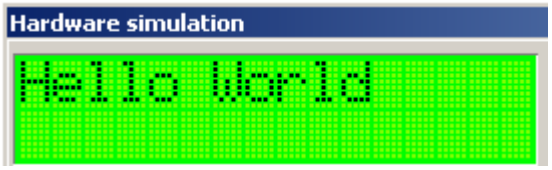
Öffnen Sie die Datei EDBexperiment7.bas von der EDB CD und laden Sie diese in den ATMega 88. Lesen Sie bitte die Anmerkungen in der Datei EDBexperiment7.bas. Sie erläutern die Initialisierung und Nutzung des LCD-Displays.

Stellen Sie sicher dass Sie das LCD mit dem Anschluss X12 des EDB verbunden haben.

Nach dem Anschluss und der Programmierung des Chips sollten Sie nun „Hello World“ auf dem LCD-Display angezeigt bekommen.

Probieren Sie auch für dieses Programm den Hardware-Simulator aus. Stellen Sie sicher, dass die Datei EDBexperiment7.bas noch geöffnet ist. Klicken Sie auf

Anschliessend klicken Sie  im AVR Simulator Fenster. Nun klicken Sie ein paar mal den „Play“ und den „Stepp-into-code“ Knopf. Sie sollten Folgendes sehen:



Online Datenquelle:
Hitachi Semi is now Renesas
LCD manufacturer: http://www.renesas.com/
LCD background: http://home.iae.nl/users/pouweha/lcd/lcd.shtml

Ubung

Ubung
1

Der beste Weg um einen Controller kennen zu lernen, ist es selbst zu tun. Erstellen Sie ein Programm welches Sie via UART nach Ihrem Namen fragt. Lassen Sie den Namen dann jedoch statt auf dem Computerbildschirm auf dem LCD ausgeben. (Die Lösung finden Sie auf der EDB-CD, Datei Exercise1.bas)

Versuch
8

3.3.2 Die 7-Segment-Anzeige

Stückliste

1x 7-Segment-Anzeige
(Gem. Anode SL-119-OSS16HWA)
Conrad 146536, oder ähnliche
8x Verbindungsdrähte

Ziele

Benützung einer 7-Segment-Anzeige

Die Benutzung von LCD-Displays hat 2 Nachteile. Zum einen können sie sehr teuer sein, zum anderen sind sie nicht so robust wie die gute alte LED. Wenn Sie nur ein Zeichen darstellen müssen, können Sie eine 7-Segment-Anzeige nutzen.

Eine 7-Segment-Anzeige ist nichts weiter als ein paar LEDs, die in einem Plastikgehäuse so angeordnet sind, dass sie ein Zeichen darstellen wenn man sie ein- oder ausschaltet.

Generell gibt es 2 verschiedene Typen, gemeinsame Anode und gemeinsame Kathode.

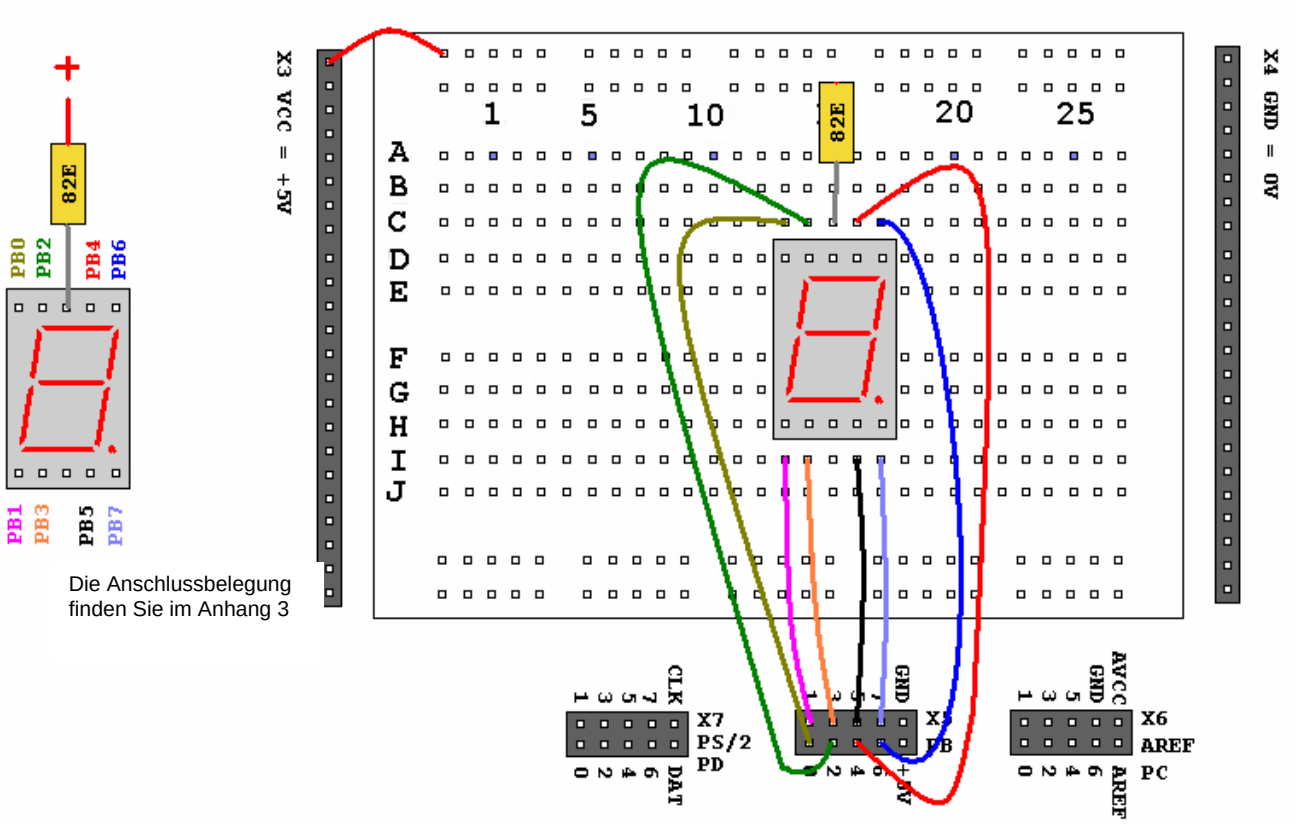
Common Anode Displays haben einen gemeinsamen POSITIVEN Pol. Das bedeutet, Sie müssen VCC am gemeinsamen Pin anschließen. Und Ihr ?Controller muss ein 0V/ GND-Signal anlegen, um die LED zum leuchten zu bringen.

Displays mit gemeinsamer Kathode haben einen gemeinsamen NEGATIVEN Pol. Das bedeutet, Sie müssen 0V/GND am gemeinsamen Pin anschließen. Und Ihr ?Controller muss ein +5V/VCC-Signal anlegen, um die LED zum leuchten zu bringen.

Zumeist benutzt man gemeinsame Anode, weil Mikrocontroller mehr Strom zu Ground als zu VCC schalten können.

Online Datenquelle:
Application note for multiple 7 segment displays:

http://www.maxim-ic.com/appnotes.cfm/appnote_number/3210
Bauen Sie diese Zeichnung nach



Die Anschlussbelegung finden Sie im Anhang 3

Öffnen Sie die Datei EDBexperiment8.bas von der EDB CD und laden Sie diese in den ATMega 88. Nach der Programmierung sollten Sie einen hexadezimalen Zähler auf dem Display sehen (0...9 A...F).

Lesen Sie bitte die Anmerkungen in der Datei EDBexperiment8.bas. Prüfen Sie, ob Sie die „Data und Read“ Befehle verstanden haben.

3.4 Tastatureingabe

Es wäre schön, wenn wir eine PS2 oder eine AT- Tastatur direkt mit unserem Mikrocontroller verbinden könnten, anstatt sie mit unserem PC zu verbinden. (und dann die Tasten direkt durch den UART auslesen könnten) In diesem Kapitel sehen wir, wie einfach es ist, eine Tastatur mit dem EDB und Bascom zu verbinden.

3.4.1 PS2 oder AT Tastatur

Versuch
9

Stückliste

1x HD44780 LCD
Tastatur
1x PS2 oder AT Tastatur
1x optional einen PC mit COM Anschluss

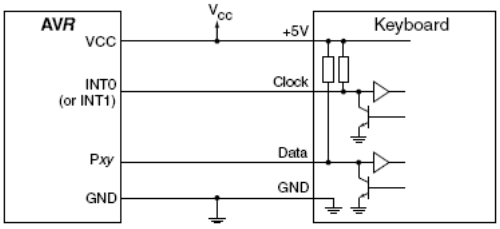
Ziele

- Anschließen einer PS2 oder AT

Das EDB wird schon mit einem Standard PS2 Interface geliefert. Das heißt, Sie können eine Standard PS2 Tastatur direkt an X10 (markiert mit PS2) am EDB anschließen. Sie können auch eine ältere AT Tastatur anschließen, aber wenn Sie sich dazu entschließen, werden Sie einen Adapter benötigen, der Ihren DIN42524 in einen Mini DIN PS2 Stecker umwandelt.

Die unten stehende Zeichnung zeigt Ihnen das Interface und die Pin Ausgänge der DIN Anschlüsse, die sich an Ihrer Tastatur befinden.

Tastatur Interface

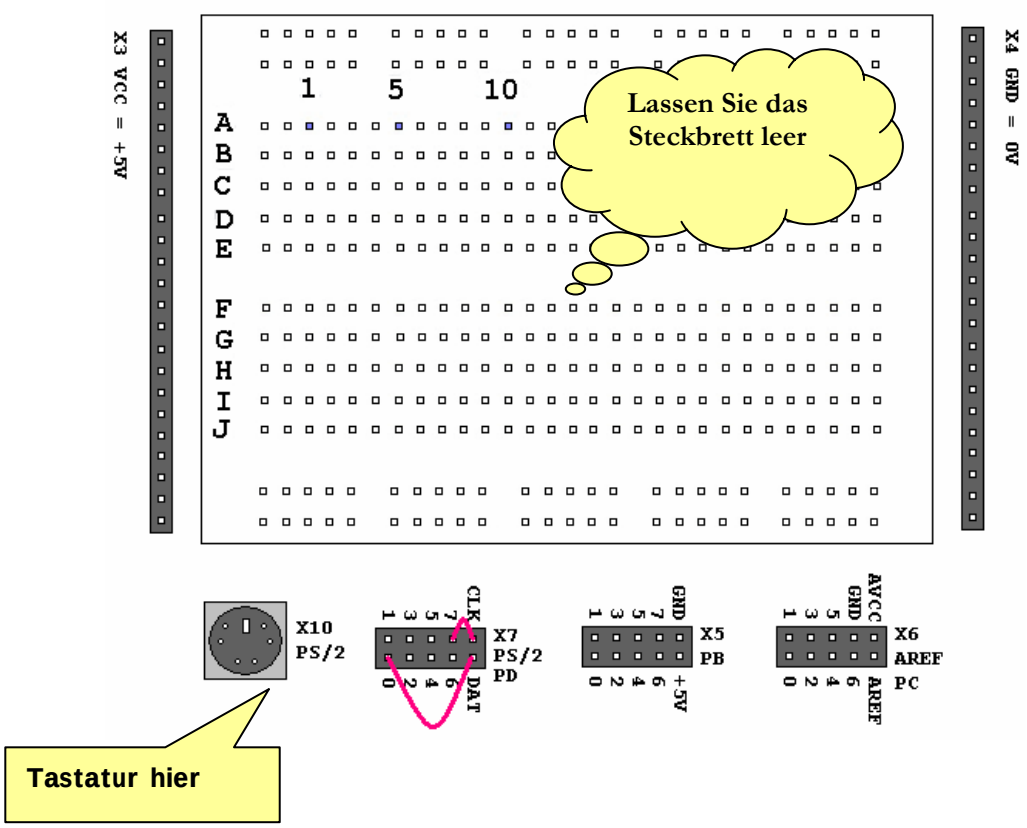


Pin Ausgänge

AT Computer		
Signals	DIN41524, Female at Computer, 5-pin DIN 180°	6-pin Mini DIN PS2 Style Female at Computer
Clock	1	5
Data	2	1
nc	3	2,6
GND	4	3
+5V	5	4
Shield	Shell	Shell

Quelle: Atmel Anwendungshinweis
AVR313: interfacing the PC AT Keyboard“

Online Datenquelle:
Atmel Application notes:
http://www.atmel.com/dyn/products/app_notes.asp?family_id=607



Schliessen Sie Ihre Tastatur an,und schauen Sie sich die obige Zeichnung an. Verbinden Sie nun CLK mit PD.7 und DAT mit PD.0. Dann öffnen und laden Sie die Datei EDBexperiment9.bas in den µC.

Schreiben Sie nun einen beliebigen Text auf der Tastatur und drücken Sie die Enter Taste. Ihr Text sollte nun auf dem Bildschirm zu sehen sein.

3.4.2 Matrix Tastatur

Versuch
10

Stückliste

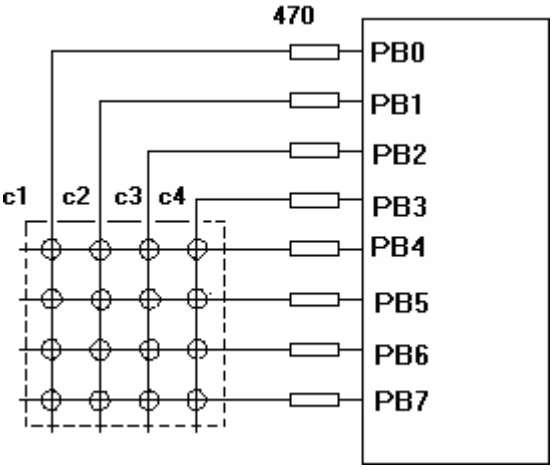
1x Matrix Tastatur
Conrad 709840
1x PC mit COM Anschluss
8x Widerstand 470E

Ziele

- Anschluss einer Matrix Tastatur

Theorie

Der ATMega88 AVR ist (zum Zeitpunkt, als dieses Handbuch geschrieben wurde) einer der neuesten Bausteine aus der AVR Familie. Zusätzlich zu den regulären Möglichkeiten der AVR Bausteine hat er mehr Interrupt Quellen,als seine Vorgänger. Wie Sie im Verlauf dieses Experiments sehen werden, ist eine Matrix Tastatur eine Tastatur mit 16 oder 9 Tasten. Sie wird an den AVR wie folgt angeschlossen:



Source: Bascom Help

Wie Sie sehen können, benötigen Sie nur 8 I/O Anschlüsse, um 16 Tasten zu erkennen. Es gibt 2 Arten um zu erkennen, ob der Nutzer eine Taste gedrückt hat. Die erste Möglichkeit ist das Abfragen. Das bedeutet wir lassen sie Software ununterbrochen überprüfen, ob eine Taste gedrückt wurde. (Eine do...loop Struktur) Das Problem hierbei ist, dass Ihr Controller nicht viel zusätzliche Aufgaben wahrnehmen kann während er die Matrix abfragt.

Die zweite Möglichkeit ist das Benutzen von Interrupts. Wie schon gesagt, kann nahezu jeder Pin des ATMega88 als Interrupt-Quelle gewählt werden. Wenn Sie Interrupts benutzen wird die „Matrix lesen“ Routine nur ausgeführt, wenn der Nutzer eine Taste gedrückt hat. (Typischerweise ist die „Matrix lesen“ Routine innerhalb der Interrupt Routine die gleiche, als wenn wir Abfragen benützen. Nur bei der Benutzung von Interrupts wird die „Matrix lesen“ Routine einmalig pro Interrupt/Tastendruck ausgeführt)

Auf der nächsten Seite finden Sie eine Kurzbeschreibung der Interrupts.

INTERRUPTS

Ein Interrupt ist ein Ereignis, welches die Ausführung Ihres Programmes stopt und die Ausführung einer bestimmten Interrupt Routine veranlasst. Interrupts können von I/O Pins ausgelöst werden. Allerdings kann auch der UART RxD Pin und die Timer/Counter einen Interrupt generieren.

Nachdem der Interrupt behandelt wurde, wird das Programm an der Stelle fortgesetzt, an der der Interrupt das Programm zuvor stopte.

Sie können Interrupts dafür benützen, um die Priorität von bestimmten Prozessen in Ihrem System zu erhöhen.

Für dieses Experiment haben wir keine Vorgabe wie das Steckbrett aufgebaut werden sollte, weil die Matrix sich nicht auf dem Steckbrett installieren lässt. Stellen Sie die Verbindung so her, wie Sie sie auf der vorherigen Seite sehen können.

Verbinden Sie Pin1 der Matrix über einen 470Ω Widerstand mit portb.0, verbinden Sie Pin2 der Matrix über einen 470Ω Widerstand mit portb.1, usw. usw...

Programmieren Sie nun den ATMega88 mit EDBexperiment10.bas und verbinden EDB und Ihren PC mit dem seriellen Kabel.

Sie sollten nun den Wert 16 auf Ihrem Bildschirm sehen. Wenn Sie eine Taste drücken sollte sich dieser Wert in einen Wert <16 verändern.

Lesen Sie die Anmerkungen in der Datei EDBexperiment10.bas und sehen Sie, ob Sie den Getkbd Befehl und die Art und Weise, wie wir die Interrupts initialisieren und nutzen verstehen.

Dieser Versuch kann nur ausgeführt werden, wenn Sie Bascom Version 1.11.8.1 oder höher verwenden.

Schauen Sie sich auch die Datei EDBexperiment10polling.bas an, welche auch die Matrix Tastatur ausliest, allerdings ohne Interrupts. Prüfen Sie, ob Sie den Unterschied zwischen beiden Verfahren verstanden haben.

Wenn Sie eine ältere Version von Bascom benützen, können Sie eine neuere Demoversion von www.mcselec.com herunterladen. Oder Sie können das Beispiel für die Matrix Abfrage EBDexeriment10polling.bas verwenden.

Versuch
11

3.4.3 Fernbedienung mit RC5 Protokoll

Stückliste

1x TSOP1736 oder equivalent Fernbedienung
1x PC mit COM Anschluss
1x RC5 Fernbedienung

Ziele

- Benutzen einer RC5

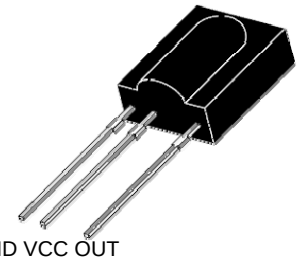
Man kann darüber diskutieren, ob ein Kapitel über „Fernbedienungen“ in den Bereich „Tastatureingabe“ gehört. Jedenfalls können wir eine Fernbedienung als ein Eingabegerät benützen.

Heutzutage hat jeder Haushalt unzählige Fernbedienungen. Einige dieser Fernbedienungen benützen das RC5 Protokoll. Das RC5 Protokoll ist in Bascom AVR integriert. Daher kann die Fernbedienungsfunktionalität sehr einfach zum EDB hinzugefügt werden.

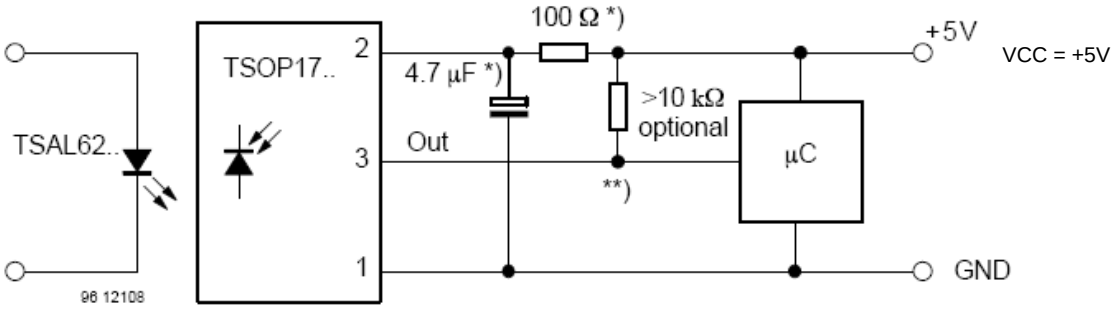
Wenn Sie mehr über das RC5 erfahren möchten, schauen Sie sich folgende von Philipps herausgegebene Veröffentlichung an:

http://www.semiconductors.philips.com/acrobat_download/applicationnotes/AN10210_2.pdf

Das TSOP1736 Modul ist ein Infrarotempfänger. Die folgende Zeichnung zeigt Ihnen ein Standardinterface für das TSOP1736.



GND VCC OUT
TSOP1736

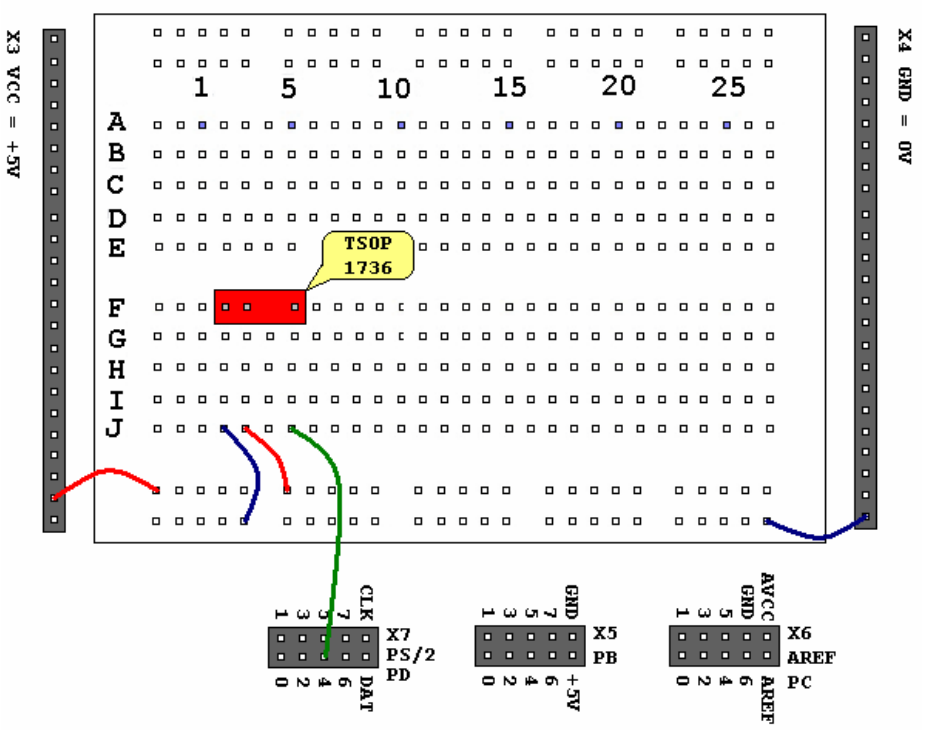


*) recommended to suppress power supply disturbances
**) The output voltage should not be hold continuously at a voltage below 3.3V by the external circuit.

Source: Vishay Semiconductors

(Die TSAL Diode befindet sich in der Fernbedienung)
Sie können auch ein anderes Infrarotempfängermodul benützen (wie zum Beispiel das SFH506 oder das SFH5110-36). Wenn Sie ein anderes benützen, stellen Sie sicher, dass Sie die Pin Ausgänge überprüfen, bevor Sie das Modul an das EDB anschliessen.

Empfohlener Aufbau für diesen Versuch:



Was wir jetzt tun müssen, ist nach einer RC5 Fernbedienung zu suchen. Die meisten Philipps Fernbedienungen nutzen das RC5 Protokoll, aber auch die meisten Universalbedienungen können dieses Protokoll benutzen.

Nachdem Sie eine RC5 Fernbedienung gefunden haben, oder wenn Sie Zweifel haben öffnen und programmieren Sie die Datei EDBexperiment11.bas. Verbinden Sie Ihr EDB und den PC mit dem seriellen Kabel. Dann zielen Sie mit Ihrer Fernbedienung auf die gewölbte Seite Ihres Infrarotempfängermoduls. Wenn Sie nun eine Taste auf Ihrer Fernbedienung drücken, sollten Sie Zahlen auf Ihrem PC Bildschirm sehen (Terminal Fenster).

Dieser Versuch kann nur ausgeführt werden, wenn Sie Bascom Version 1.11.8.1 oder höher verwenden.

Lesen Sie die Anmerkungen in der Datei EDBexperiment11.bas und sehen Sie, ob Sie verstehen, was passiert.

Online Datenquelle:
Background on RC5: http://www.clearwater.com.au/rc5/
http://www.epanorama.net/links/irremote.html

3.5 Erweiterte I/O

Wir möchten nun über fortgeschrittenere I/O Fragen diskutieren, welche nicht einfach zu verstehen sind wenn wir keinen UART benützt haben. (Daher die Reihenfolge der Kapitel) In Kapitel 3.5.1 werden wir zuerst feststellen, dass Schalter nicht ideal sind. Im weiteren Verlauf werden wir sehen, wie man die Position eines Potentiometers feststellt und wie man ein LDR verwendet, um Helligkeitsgrade zu erkennen. Wir werden dieses Kapitel damit beenden, Töne mit einem Lautsprecher zu erzeugen.

Versuch
12

3.5.1 Zähler mit entprelltem Eingang

Stückliste

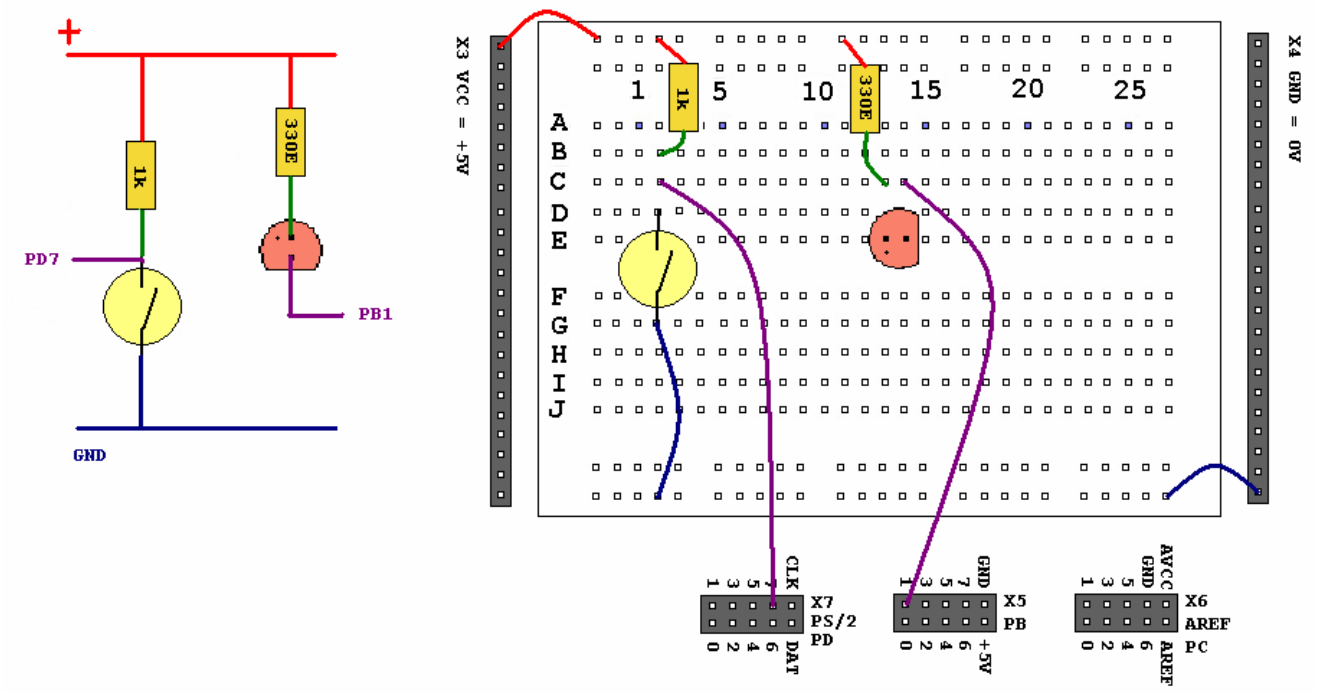
1x LED
1x Schalter oder Draht
6x Steckbrett Kabel
1x Widerstand 330E, 1k

Ziele

- Etwas über nicht ideale Schalter erfahren
- Entprellen mit Hilfe von Software

Schalter sind nicht ideal. Das bedeutet, wenn Sie einen Schalter an einen Mikrocontroller anschliessen, wie sie es im Versuch 1 taten...könnten Sie ein ungenaues Programm erhalten. Das ist kein grosses Problem wenn Sie eine LED anschliessen, aber es kann zu einem Problem werden, wenn Sie einen Zähler aufbauen wollen.

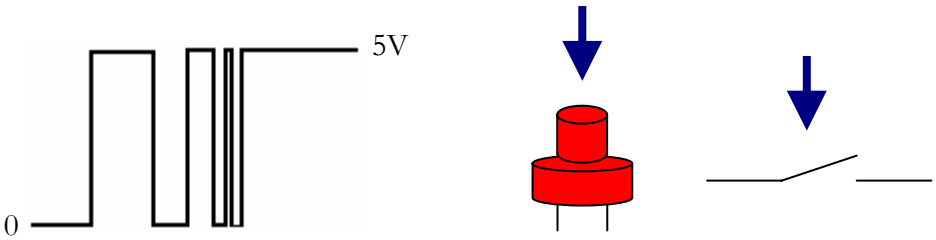
Lassen Sie uns noch einmal den Versuch 1 aufbauen:



Das Programm EDBexperiment12a.bas „zählt“ wie oft Sie mit dem Schalter den Kontakt schließen. Sie können die Anzahl in Ihrem Terminal Fenster sehen. Wenn Sie Glück haben sehen Sie, dass das Programm jedes Mal, wenn Sie den Kontakt schließen lediglich einmal zählt.

Nun ersetzen Sie den Schalter durch einen einfachen Draht. Dann schliessen und öffnen Sie mehrmals den Kontakt (Draht) und beobachten Sie den Zähler. Wie Sie sehen können ergibt dies ein mangelhaftes Zählergebnis. Die Zählung erfolgt korrekt, manchmal allerdings zählt es mehrmals, wenn Sie den Kontakt schließen.

Was passiert ist folgendes ; Jedes Mal wenn Sie den Kontakt schließen ist dieser nicht direkt stabil. Mit anderen Worten, er verhält sich wie folgt:



Dieser Effekt wird als „prellen“ bezeichnet. Die Kontaktgabe wird zwar innerhalb von Millisekunden stabil, aber der Mikrocontroller läuft mit über 8MHz. Dies eröffnet dem Mikrocontroller die Möglichkeit, fallende und steigende Flanken innerhalb von Mikrosekunden zu erkennen. Wenn sich der Schalter wie im obigen Diagramm verhält, würde der Controller 4x, anstatt nur 1x zählen.

Die Lösung ist eine Zeitverzögerung „dead time“ zu erstellen, in welcher der Controller keine Spannungsveränderung bemerkt. Die Zeitverzögerung kann einfach programmiert werden, indem man ein „Waitms 100“ direkt nach der Zeile einfügt, in welcher der Controller die Eingabe abfragt.

Nun können Sie die Datei EDBexperiment12b.bas in den Mikrocontroller laden und die Anmerkungen lesen. Beachten Sie bitte die Zeile „Waitms 100“. Wenn Ihr Controller immer noch ungenau zählt, können Sie den Wert des Waitms-Befehls auf 250 ändern.

Das Einfügen des „Waitms“ Befehls um genaue Zählwerte zu erhalten nennt man „Entprellen per Software“.

Versuch
13

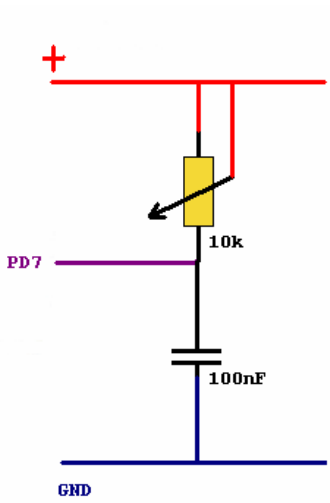
3.5.2 Der GetRC Befehl

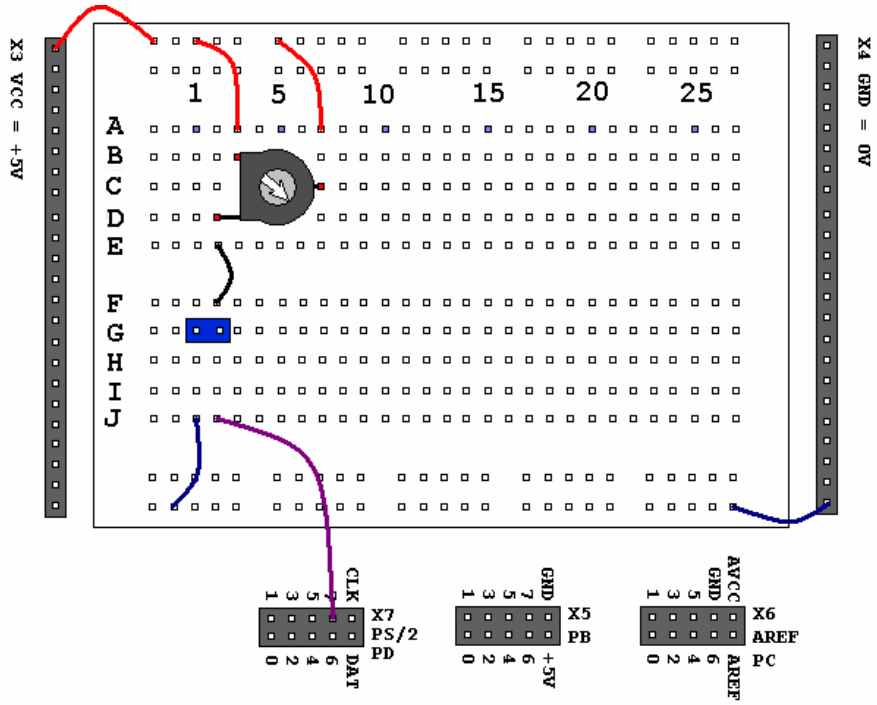
Stückliste
1x Keramik Kondensator 100nF
1x Potentiometer 10k
7x Verbindungsdrähte

Ziele
Beherrschen des GetRC Befehls

Der GetRC Befehl im Bascom gibt uns die Möglichkeit die Position des Schleifer-Abgriffs eines Potentiometers auszulesen. Dies kann nur benutzt werden, wenn keine besondere Genauigkeit notwendig ist.

Bauen Sie die unten gezeigte Schaltung auf.





Der Text auf dem Kondensator entspricht dem Wert in pico Farad. 100 nF = 100000pF. Die letzte Ziffer auf dem Kondensator ist die Anzahl der Nullen. (so bedeutet 104 auf dem Kondensator = 100000pF = 100nF)

Das Potentiometer passt nicht auf jedes Steckbrett. Wenn das passiert müssen Sie Drähte mit 0,5mm Durchmesser an das Potentiometer löten, bevor Sie es anschließen können.

Programmieren Sie den ATMega88 mit EDBexperiment13.bas und verbinden Sie das EDB mit Ihrem PC über das serielle Kabel. Sie sollten nun Werte auf Ihrem Bildschirm sehen. Diese Werte sollten Sie durch Drehen am Potentiometer verändern können. Lesen Sie die Anmerkungen in der Datei EDBexperiment13.bas und sehen Sie, ob Sie verstehen, was passiert.

Versuch
14

3.5.3 Sirene mit SOUND Befehl

Stückliste

1x Lautsprecher RS267–6968
oder äquivalent
1x Elko = 100µF
1x Widerstand 120Ω
7x Verbindungsdrähte

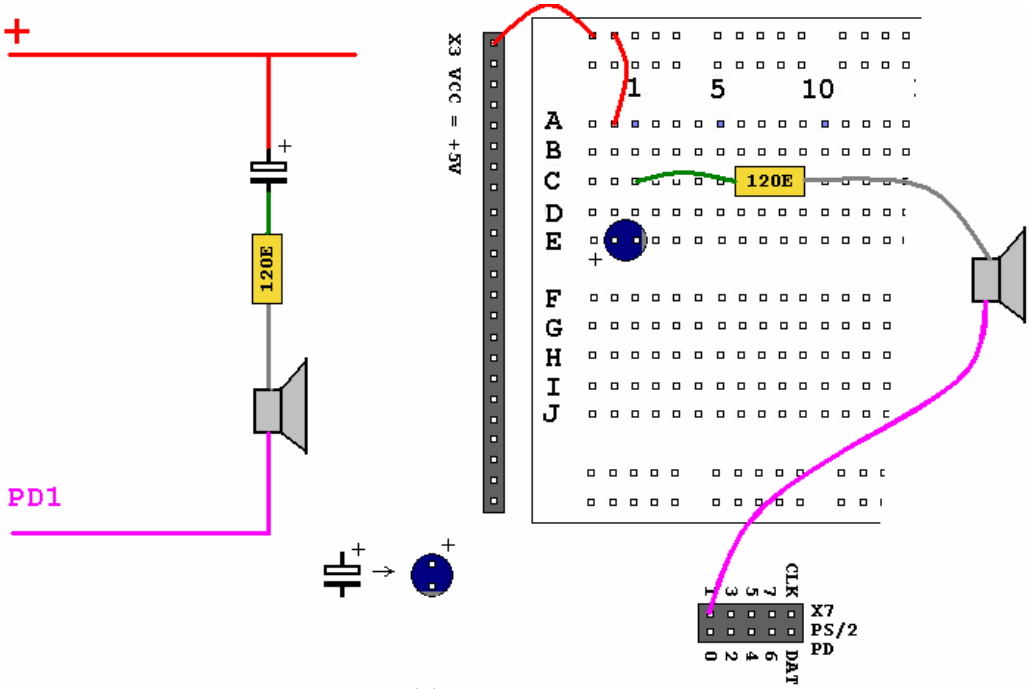
Ziele

- Beherrschen des SOUND Befehls

Der SOUND Befehl gibt Ihnen die Möglichkeit verschiedene Töne mit einem Lautsprecher zu erzeugen. Er ist ideal, wenn Sie eine Anwendung haben, in welcher Sie den Benutzer auf spezifische Situationen hinweisen wollen (OK oder ERROR Warnton). Der SOUND Befehl ist nicht dafür gedacht, um genaue Frequenzen zu erzeugen. Wenn Sie das möchten, nutzen Sie einen Zeitgeber (Timer) (Kapitel 3.7.3).

Für diesen Versuch sollten Sie einen Elko von ungefähr 100µF benützen (aber 47µF, 220µF gehen ebenso gut). Sie können einen alten PC Lautsprecher verwenden, oder einen RS 267-6968 bestellen. In den meisten Fällen kann der Lautsprecher nicht auf das Steckbrett montiert werden. Deshalb müssen Sie einen Draht an den Lautsprecher löten. Benutzen Sie einen massiven Kupferdraht von ungefähr 0,5mm Durchmesser.

Bauen Sie die folgende Versuchsanordnung nach:



Programmieren Sie den ATMega88 mit EDBexperiment14.bas und schauen Sie, ob Sie den Programmcode verstehen. Sie sollten nun einen Sirenenalarm hören.

3.6 Analog/Digital (AD) und Digital/Analog (DA) Wandlung

Obwohl Mikrocontroller digital sind, können Sie auch eine Analog/Digital (AD) und Digital/Analog (DA) Wandlung durchführen. Dies eröffnet Ihnen die Möglichkeit analoge Signale zu messen. Aber auch „analoge“ Ausgaben zu erzeugen. Dieses Kapitel zeigt Ihnen, wie man analoge Signale umwandelt oder erzeugt.

Versuch
15a

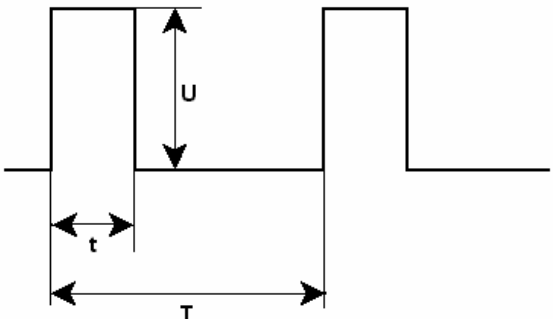
3.6.1 Pulsweitenmodulierter (PWM) Ausgang

Stückliste
1x MOSFET IRF520 oder äquivalent
1x Glühbirne 6V
6x Verbindungsdrähte

Ziele
- Beherrschen von PWM

PWM ist eine Form der digitalen Ausgabe. Trotzdem können wir es als anloge Ausgabe verwenden.

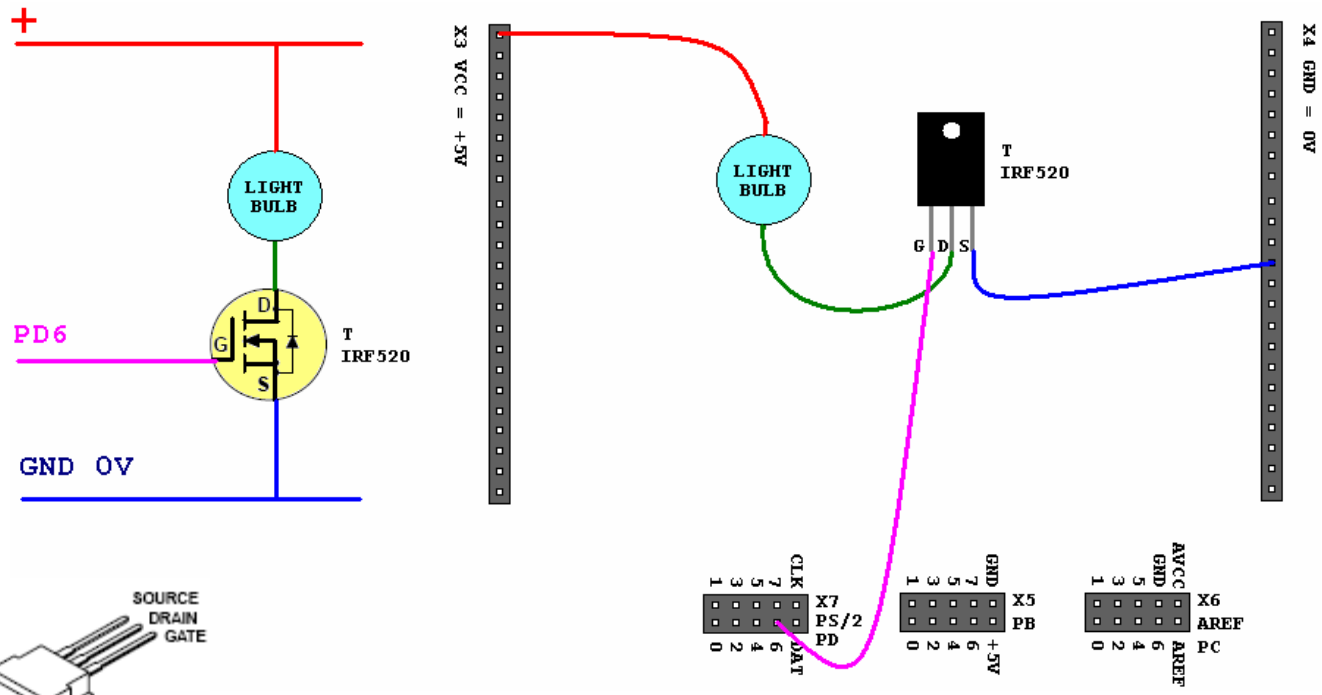
PWM Signale sind digitale Gleichstromsignale, die wie folgt aussehen:

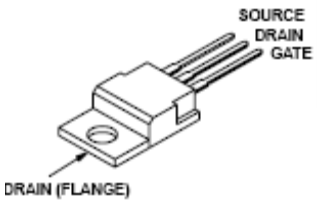


Stellen Sie sich vor, wir speisen eine Glühbirne mit einem derartigen Signal. Wenn wir nun die Zeit t erhöhen, erhöht sich auch die Durchschnittsspannung an der Glühbirne. Wir wählen die Zeit T derart, dass wir kein Lichtflackern mehr wahrnehmen. So sieht es aus, als wäre die Glühbirne analog angesteuert.

Wenn Sie etwas anderes wie eine Glühbirne verwenden, reicht normalerweise das Anschliessen eines Kondensators an den PWM Ausgang aus, um ein analoges Signal zu erzeugen.

Das oben beschriebene System wird Pulsweitenmodulation (PWM) genannt. PWM ist in den meisten AVR Bausteinen integriert und wird von Bascom unterstützt.
Dieser Versuch zeigt Ihnen ein PWM Beispiel.
Bauen Sie folgende Versuchsanordnung:





Die Komponenten passen nicht auf das Steckbrett. Deshalb werden Sie einige Drähte anlöten müssen.

Programmieren Sie den ATMega88 mit EDBexperiment15a.bas und verbinden Sie Ihr EDB und den PC mit dem seriellen Kabel.

Das von der Glühbirne erzeugte Licht sollte sich verändern und sie sollten die PWM Werte auf Ihrem Bildschirm ablesen können.

Lesen Sie die Kommentare in der Datei EDBexperiment15a.bas und sehen Sie, ob Sie verstehen was passiert. Das Programm EDBexperiment15a.bas nutzt einen Zeitgeber (Timer). In Kapitel 3.7.3 erfahren Sie hierrüber mehr.

Online Datenquelle:
PWM Background:
http://www.netrino.com/Publications/Glossary/PWM.html
http://www.4qdtec.com/pwm-01.html

IRF520: http://ec.irf.com/v6/en/US/adirect/ir?cmd=catProductDetailFrame&productID=IRF520

Versuch
15b

3.6.2 AD Wandlung mit lichtempfindlichem Widerstand (LDR)

Stückliste

1x LDR NSL19-MS51
RS596-141
1x PC mit COM Anschluss
2x Widerstand 2k2

Ziele

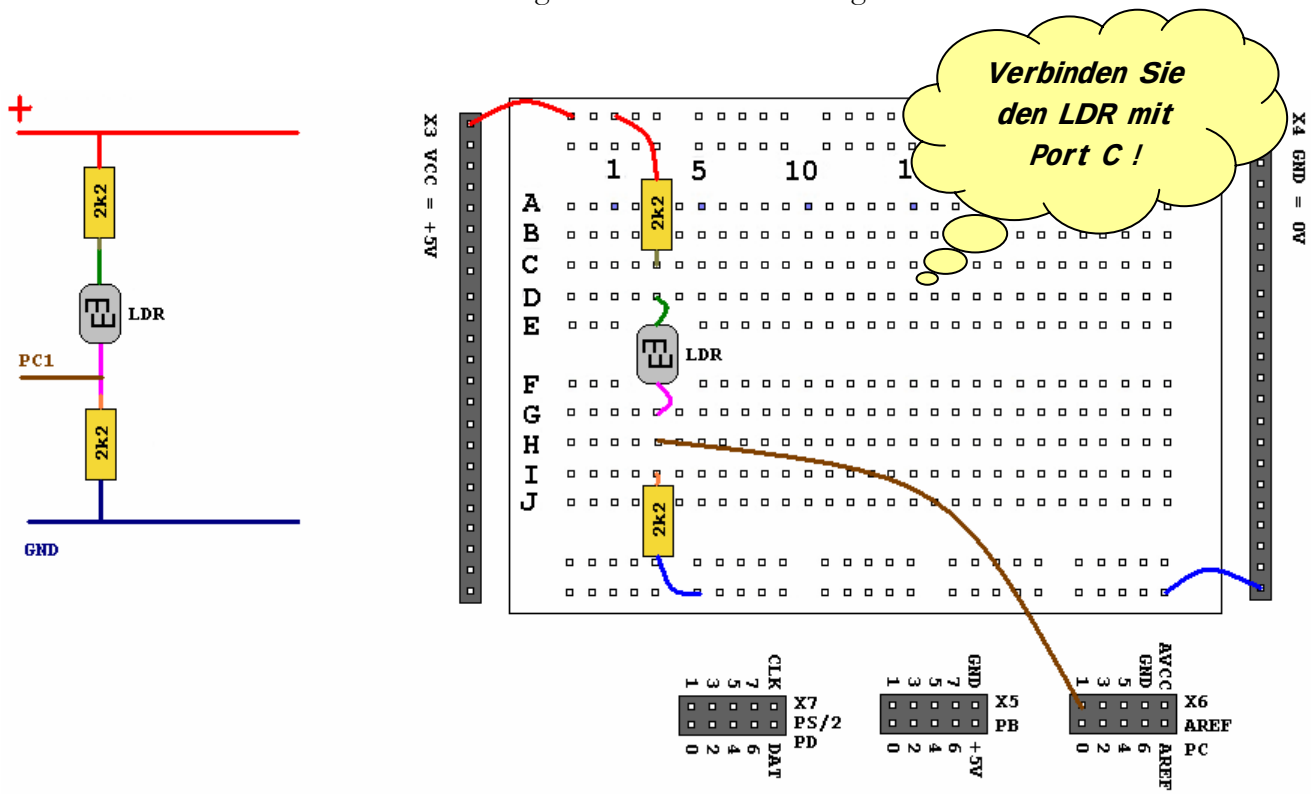
- Benutzen eines AD Wandlers
- Beherrschen von LDRs

Dieses Kapitel zeigt eine AD Wandler Anwendung und bereitet Sie auf Übung 2 „der LDR Lichtschalter“ vor.

LDRs (Light Dependant Resistor)sind lichtempfindliche Widerstände. Das bedeutet, dass der Wert des Widerstandes von der Lichtstärke abhängt. Für diesen Versuch können die meisten LDRs genutzt werden.

Der LDR NSL19-MS51 hat einen Widerstand von 5k bei normalen Tageslicht (ohne Sonneneinstahlung). Einen Widerstand von 1M2 in der Dämmerung und ein Widerstand von >35M in kompletter Dunkelheit. Für diesen Versuch können die meisten Typen von LDRs verwendet werden. Wenn Sie einen anderen Typ benützen, werden Sie andere Werte auf Ihrem Bildschirm sehen können.

Zuerst bauen Sie folgende Versuchsanordnung



Programmieren Sie nun den ATMega88 mit EDBexperiment15b.bas und verbinden Sie Ihr EDB mit dem PC über das serielle Kabel.

Sie sollten nun Werte auf Ihrem Bildschirm ablesen können. Diese Werte sollten Sie ändern können, indem Sie Ihre Hand über den LDR halten oder ihn mit einer Taschenlampe anleuchten.

Lesen Sie die Anmerkungen in der Datei EDBexperiment15b.bas und sehen Sie, ob Sie verstehen was passiert.

Verbinden Sie PC1 mit Masse und beobachten Sie. Dann verbinden Sie PC1 mit VCC = +5V und beobachten Sie.

Online Datenquelle:
LDR Background: http://www.radio-electronics.com/info/data/resistor/ldr/light_dependent_resistor.php

http://www.technologystudent.com/elec1/ldr1.htm

Ubung 2, LDR Lichtschalter

Ubung
2

Testen Sie Ihr Wissen über die vorherigen Kapitel indem Sie dies Übungsbeispiel bauen.

Bauen Sie einen LDR Lichtschalter,

Verbinden Sie eine LED oder eine Glühbirne (wie Sie sie beispielsweise in Ihrem Fahrrad finden können) mit Ihrem Mikrocontroller. Stellen Sie sicher, dass Sie sie nicht direkt anschliessen! Nutzen Sie einen BS170 um den relativ grossen Strom zu schalten.
Wenn Sie Ihre Glühbirne angeschlossen haben, prüfen Sie, ob Sie sie mit einem einfachen Programm, wie Sie es beispielsweise in Versuch 1 gesehen haben, ein- und ausschalten können.
Nun benützen Sie den LDR aus dem vorigen Kapitel und schalten Sie die Glühbirne an wenn es dunkel wird und schalten Sie sie aus, wenn es hell ist.
Wenn Sie Erfolg hatten Herzlichen Glückwunsch!! Wenn nicht, werfen Sie einen Blick auf die Lösung dieser Aufgabe. Diese finden Sie auf der EDB CD. Dateiname: Exercise2.bas

Versuch
16

3.6.3 Preiswerter Spannungsmesser

Stückliste

1x PC mit COM Anschluss
1x Widerstand 1k2, 3k9
Anwendung
1x Potentiometer 10k
1x Spannungsmesser

Ziele

- Benutzen eines AD Wandlers
- Beherrschen einer AD

Wenn Sie die AD Werte auf Ihrem Bildschirm in Versuch 15 beobachtet haben, haben Sie wahrscheinlich herausgefunden, dass eine Spannung von 0V einen AD Wert von ungefähr 0 und eine Spannung von 5V einen Wert von ungefähr 1024 ergibt. Tatsächlich wird der AD Wert 1024 bereits von einer Spannung von 1,1V erreicht.

Lassen Sie uns nun einen Spannungsmesser von Spannungen zwischen 0 und ungefähr 5V mit einer Auflösung von 0,1V bauen.

Weil wir nur 1,1V messen können müssen wir ein paar Widerstände als Spannungsteiler hinzufügen. Auf diese Weise repräsentiert eine vom Controller gemessene 1V Spannung eine tatsächliche 5V Eingangsspannung.

Die Zeichnung auf der linken Seite zeigt welche Widerstände Sie für den Spannungsteiler verwenden könnten.

V	AD Value
1,1	1024
1	931
0,9	838
0,8	745
0,7	652
0,6	559
0,5	465
0,4	372
0,3	279
0,2	186
0,1	93
0	0

Das Fazit hieraus ist, dass wir 1024 durch 11 teilen müssen. Das Ergebnis ist 93,09. Es wird vorzugsweise mit ganzen Zahlen (integers) gearbeitet, weil Mikrocontroller nicht sehr gut teilen können. (Das heisst, Controller können teilen, aber mit Rest, 1024 durch 11 ergibt 93 Rest 1)

Wir runden 93,09 ab auf 93. Die Ergebnisse der Berechnung für jedes Volt sehen wir in der Tabelle oben rechts.

Legen wir jetzt beispielsweise an den Port C.1 Pin an und der AD Wert ist 190. 190 ist nicht ein Wert, der in der Tabelle vorhanden ist....deshalb müssen wir irgendwie abrunden.

Sagen wir zum Beispiel alle Werte unter 93 sind 0,0V.
Werte von 93 bis 185 sind 0,1V, Werte von 186 bis 278 sind 0,2V, usw.....
Das einzige was wir nun tun müssen, ist ein Programm zu schreiben, welches testet, ob die AD Werte innerhalb eines bestimmten Bereiches liegen....wie folgt:

```
Test_value = 93  '(The initial value "11" because everything below 11 is 0,0V)
 AD_value = 190
 Voltage = 00

 Not_ready:
   If AD_value ≤ Test_value then goto ready
   Test_value = Test_value + 93
   Voltage = Voltage + 1
 goto not_ready

 Ready:
 Print Voltage
```

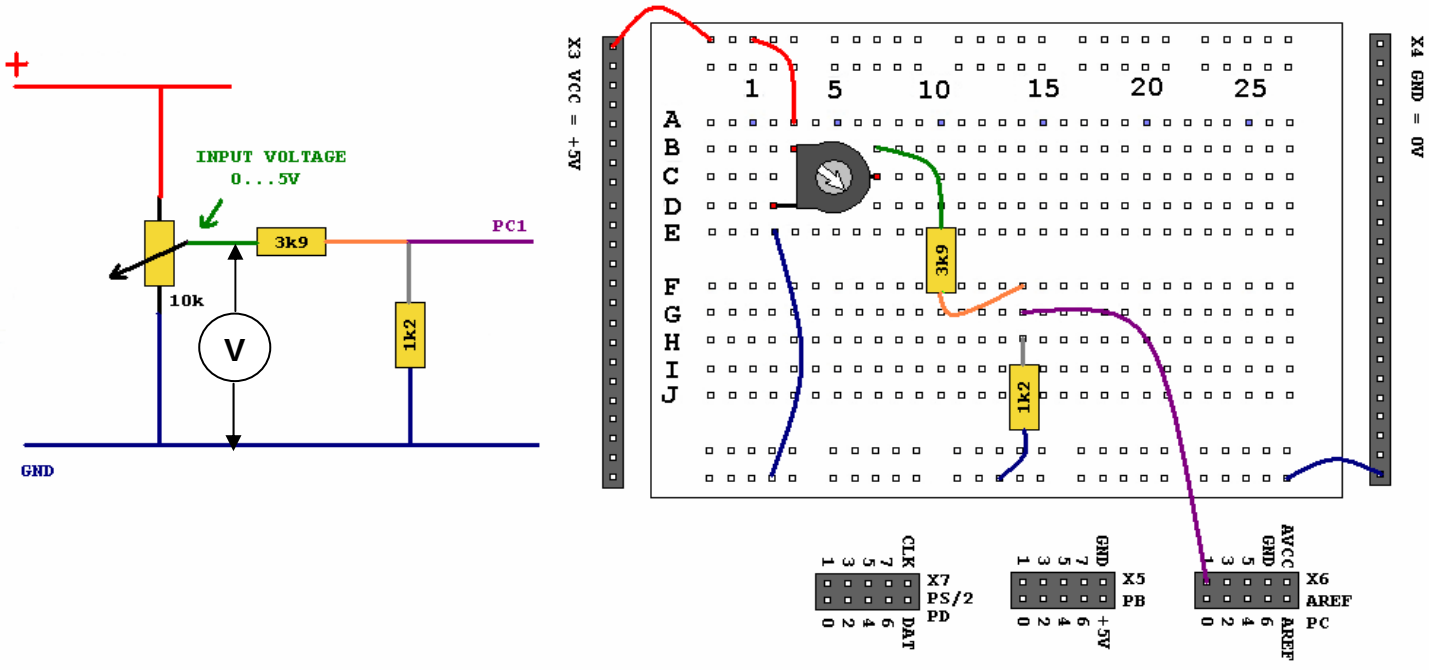
In unserem Beispiel war AD_value 190. Der Anfangswert für Test_value war 93. Wenn nun der IF-Befehl FALSE...ergibt, addieren wir 93 zum Testwert und 1 zur Spannung. (Wir haben nun festgestellt, daß die gemessene Spannung nicht 0,0V ist).

Test_value ist nun 186, der IF Befehl noch FALSE. Folglich ist die gemessene Spannung nicht 0,1V. Wie addieren jetzt wiederrum 93 zum Testwert und 1 zur Spannung.

Test_value ist jetzt 279, der IF Befehl ergibt TRUE! Nun springen wir zur Marke „ready". Wenn wir jetzt die Spannungsvariable anzeigen, würden wir 0,2V sehen. Wir haben nun erfolgreich den AD Wert ohne die Division zu nutzen interpretiert.

Wenn wir das Programm wie oben vorgeschlagen benützen, ist die Ausgabe von 0 bis 1,1V. Teilen wir alle Werte durch 4 erhalten wir einen Wert, der der vollen Skala entspricht.

Programmieren Sie EDBexperiment16.bas in den Controller. Schliessen Sie das serielle Kabel an und bauen Sie folgende Schaltung. **Verifizieren** Sie, ob der Controller entsprechend der obigen Theorie arbeitet, indem Sie am Potentiometer drehen und die Spannung zwischen GND und PC1 mit einem Spannungsmesser messen.



Übung
3

Übung 3, Spannungsmesser mit Balkendiagrammausgabe

Testen Sie Ihr Wissen über die vorhergehenden Kapitel, indem Sie dieses Übungsbeispiel aufbauen.

Bauen Sie eine Balkendiagrammausgabe für den Spannungsmesser im vorigen Versuch.

Schließen Sie 8 LEDs an Port B an. Wenn die Spannung = 5V sollen alle LEDs leuchten, wenn die Spannung 2,5V beträgt, sollen die LEDs an PD0...4 leuchten, usw....

Hinweis:

```
Select Case AD_value
  Case Is < 128 : Portb = &B11111110
  Case 128 To 256 : Portb = &B11111100
  Case 256 To 384 : Portb = &B11111000
  Case 384 To 512 : Portb = &B11110000
  Case 512 To 640 : Portb = &B11100000
  Case 640 To 768 : Portb = &B11000000
  Case 768 To 896 : Portb = &B10000000
  Case Is > 896 : Portb = &B00000000
End Select
```

Schliessen Sie auch ein LCD an und lassen Sie darauf die Spannung anzeigen.

Wenn Sie Erfolg hatten Herzlichen Glückwunsch!! Wenn nicht, werfen Sie einen Blick auf die Lösung dieser Aufgabe. Diese finden Sie auf der EDB CD. Dateiname: Exercise3.bas.

Exercise 4, A variable light source

Ubung
4

Testen Sie Ihr Wissen über die vorherigen Kapitel, indem Sie dieses Übungsbeispiel aufbauen.

Eine veränderliche Lichtquelle aufbauen....Genauer: Sie sollten die Lichtstärke einer Glühbirne verändern können indem Sie am Potentiometer drehen.

- Verbinden Sie eine Glühbirne (wie Sie sie beispielsweise in Ihrem Fahrrad finden können) mit Ihrem Mikrocontroller. Stellen Sie sicher, dass Sie sie nicht direkt anschliessen! Nutzen Sie einen BS170 um den relativ grossen Strom zu schalten.

- Wenn Sie Ihre Glühbirne angeschlossen haben, prüfen Sie, ob Sie sie mit einem einfachen Programm, wie Sie es beispielsweise in Versuch 1 gesehen haben, ein- und ausschalten können.

- Nutzen Sie die PWM Funktionalität, um die Lichtstärke der Glühbirne zu regeln.

- Nutzen Sie ein Potentiometer in Verbindung mit der AD Wandlung aus dem vorherigen Versuch, um die Lichtstärke der Glühbirne zu regeln.

Wenn Sie Erfolg hatten Herzlichen Glückwunsch!! Wenn nicht, werfen Sie einen Blick auf die Lösung dieser Aufgabe. Diese finden Sie auf der EDB CD. Dateiname: **Exercise4.bas**.

3.7 Andere Controller Funktionen

In diesem Kapitel werden wir einige andere Fähigkeiten des Controllers behandeln, die oft benützt werden, aber nicht in einem der anderen Versuche auftauchten. Sie werden sehen, wie man den Speicher, den „Watchdog“ und die internen Zeitgeber (Timer) des Controllers nutzt.

3.7.1 Speicherarten

Stückliste	Ziele
-keine-	- Kennen der Speicherarten

Theorie

Der ATMega88 Mikrocontroller hat 3 Speicherarten.

FLASH -> Bootloader FLASH
Anwendungs FLASH
RAM
EEPROM

Der FLASH Speicher (8kByte) kann nur mit einem externen Programmiergerät wie zum Beispiel dem STK200/300 Dongle oder dem Wiazania USB STK (Kapitel 4) programmiert werden.

Es gibt jedoch eine Ausnahme, den Bootloader. Wenn Sie die Bootloader Möglichkeit nutzen, teilen Sie den FLASH Speicher in zwei Abschnitte. Der Bootloader selbst befindet sich im Bootloader FLASH und wird mit dem STK programmiert. Der Anwendungs FLASH kann dann über das serielle Kabel programmiert werden (Kapitel 2.3).

Im Anwendungs FLASH speichern wir das Programm/die Firmware, welche wir ausführen wollen. Wenn Sie den Bootloader nicht benützen, ist der gesamte FLASH für den Programmcode verfügbar. Der FLASH ist nicht flüchtig. Das bedeutet, dass er erhalten bleibt, wenn Stromzufuhr zum Chip getrennt wird.

Während FLASH Ihr(en) Programm/Anwendungscode speichert, wird RAM Speicher dazu benutzt, die Eingaben und Resultate von arithmetischen und logischen Operationen zu speichern. Des weiteren werden Bytes, die vom UART gesendet oder empfangen werden müssen, zuerst in den RAM geladen. Auch die Rücksprungadressen von JUMP (Sprung) Befehlen werden von Bascom im RAM abgelegt.

RAM (Random Access) Speicher ist eine Form der temporären

Speicherung. Nach Beendigung der Rechenoperationen müssen Sie die Daten entweder speichern oder löschen. RAM ist flüchtig. Das bedeutet, dass sein Inhalt verloren geht, wenn die Stromzufuhr unterbrochen wird.

Eine spezielle Art von Speicher ist der EEPROM. (Electrically Erasable/Programmable Read Only Memory/ Elektrisch löschbarer/programmierbarer nur-lese-Speicher). Lassen Sie sich von dieser Bezeichnung nicht in die Irre führen. Früher konnte EEPROM nur gelesen werden, dann wurde er ROM Speicher genannt und konnte NIEMALS vom Controller selbst gelöscht werden. Man konnte den ROM nur mit einem ROM Programmiergerät programmieren und man konnte den Chip nur mit Ultraviolettlicht löschen (Die Chips hatten ein transparentes Fenster an der Oberseite).

Heutzutage kann der Controller den EEPROM elektrisch löschen und programmieren. Dies eröffnet Ihnen die Möglichkeit, Daten zu speichern, selbst wenn die Stromversorgung getrennt wird. Das ist die Bedeutung von „nicht flüchtig".

EEPROM ist besonders handlich, wenn Ihre Anwendung vom Nutzer programmierbare Einstellungen hat. Stellen Sie sich vor, was passieren würde, wenn Sie Ihre TV Kanäle jedes Mal neu einstellen müssten, sobald Sie Ihren Fernseher einschalten...Aber Sie müssen es nicht, weil ein EEPROM in Ihrem Fernseher sitzt.

Versuch 17

Verbinden Sie Ihr EDB mit dem PC über das serielle Kabel, programmieren Sie die Datei EDBexperiment17.bas und starten Sie das Bascom Terminal.

Dieses Programm fordert Sie auf, ein alphanumerisches Zeichen einzugeben und es speichert die Daten...Folgen Sie den Anweisungen auf dem Bildschirm und beobachten Sie was passiert nachdem Sie die Stromquelle getrennt und wieder angeschlossen haben.

Haben Sie gesehen, dass die Werte des EEPROM erhalten geblieben sind?

Online Datenquelle:
Atmel AVR family:
http://www.atmel.com/dyn/products/param_table.asp?family_id=607

3.7.2 Watchdog

Stückliste
-keine-

Ziele
- Kennenlernen des Watchdog

Ein Watchdog Zeitgeber (Timer) ist ein Sicherheitsmerkmal, welches verhindert, dass Software außer Kontrolle gerät. Wenn Ihre Software korrekt läuft, wird dieser Timer von Ihnen periodisch zurück gesetzt.

Theorie

Wenn sich nun der Controller innerhalb eines Stückes sinnlosen Codes aufhängt (nach einem elektrischen Problem) oder wenn externe Hardware nicht mehr oder nicht ordnungsgemäß arbeitet, wird der Watchdog Timer nicht wieder zurückgesetzt. Wenn das passiert läuft der Watchdog über und resettet Ihren Controller. Der Mikrocontroller wird dann neu starten und beginnt damit, Ihren Code von Anfang an auszuführen.

Eine typische Watchdog Implementation sieht wie folgt aus:

```
Config Watchdog = 2048                    'reset after 2048 mSec
Start Watchdog                            'start the watchdog timer

   Do
   'Print "Hello"
   Reset Watchdog
   'Your program code goes here, repeat the Reset Watchdog
statement
   Loop
End
```

Wenn Sie möchten, können Sie mit dem Watchdog experimentieren. Sie finden die Datei Watchdog.bas auf der EDB CD. Entfernen Sie den Kommentar in der Zeile „Print Hello“ und fügen Sie in der Zeile „reset watchdog“ ein Kommentarzeichen ein. Sie werden den „reset“ in Ihrem Bildschirmfenster bemerken.

Die maximale Watchdog Zeit für den ATMega88 beträgt 8 Sekunden. Für die meisten Anwendungen ist es empfehlenswert, eine kürzere Zeit zu verwenden. Wählen Sie statt dessen 2048 oder 1024.

Versuch
18

3.7.3 Verwendung von Timern

Stückliste
-keine-

Ziele
- Benutzung des Timers

Der ATMega88 hat einen eingebauten Timer. Sie können ihn benutzen, um Zeitintervalle zu messen oder Ihre eigene Warteschleife zu programmieren. Sie werden nun ein Beispiel eines Sekunden Timers sehen.

Verbinden Sie Ihr EDB mit dem PC über das serielle Kabel, laden Sie die Datei EDBexperiment18.bas in den μC und starten Sie das Bascom Terminal.

Sie sehen nun einen Sekundenzähler auf dem Bildschirm. Lesen Sie die Anmerkungen in der Datei EDBexperiment18.bas und sehen Sie, ob Sie verstehen was passiert.

Anmerkung:
Das EDBexperiment18.bas Programm ist für ungefähr 10 Minuten genau. Es nutzt den internen RC Oszillator des ATMega88. Wenn Sie eine genauere Zeit benötigen, nutzen Sie besser eine Echtzeituhr, wie zum Beispiel den Maxim Dallas DS1307 Real Time Clock IC. Der DS1307 hat eine I²C Schnittstelle. Über die Nutzung von I²C werden Sie im nächsten Kapitel mehr erfahren.

Überblick über verfügbare Timer

Der ATMega88 hat folgende eingebaute Timer:

8-bit Timer/Counter0 with PWM
16-bit Timer/Counter1 with PWM
8-bit Timer/Counter2 with PWM and Asynchronous Operation

In der Bascom Hilfe finden Sie eine Beschreibung wie man diese Timer initialisiert und welche Interrupt Quellen sie nutzen. Drücken Sie die Taste F1 in dem Bascom AVR IDE Fenster und suchen Sie nach „Config Timer“ im Index.

3.8 Andere Schnittstellen

Versuch

19

Außer dem UART gibt es noch andere gebräuchliche Schnittstellen. In diesem Kapitel werden Sie sehen wie man das I²C-Protokoll benutzt und wir werden sehen wie man das USB-Modul nutzen kann, welches für das Ausbildungs- und Entwicklungsboard verfügbar ist.

3.8.1 Der I²C bus

Teileliste:
1x DS1624
2x Widerstand 4k7
1x Computer mit COM-Anschluss

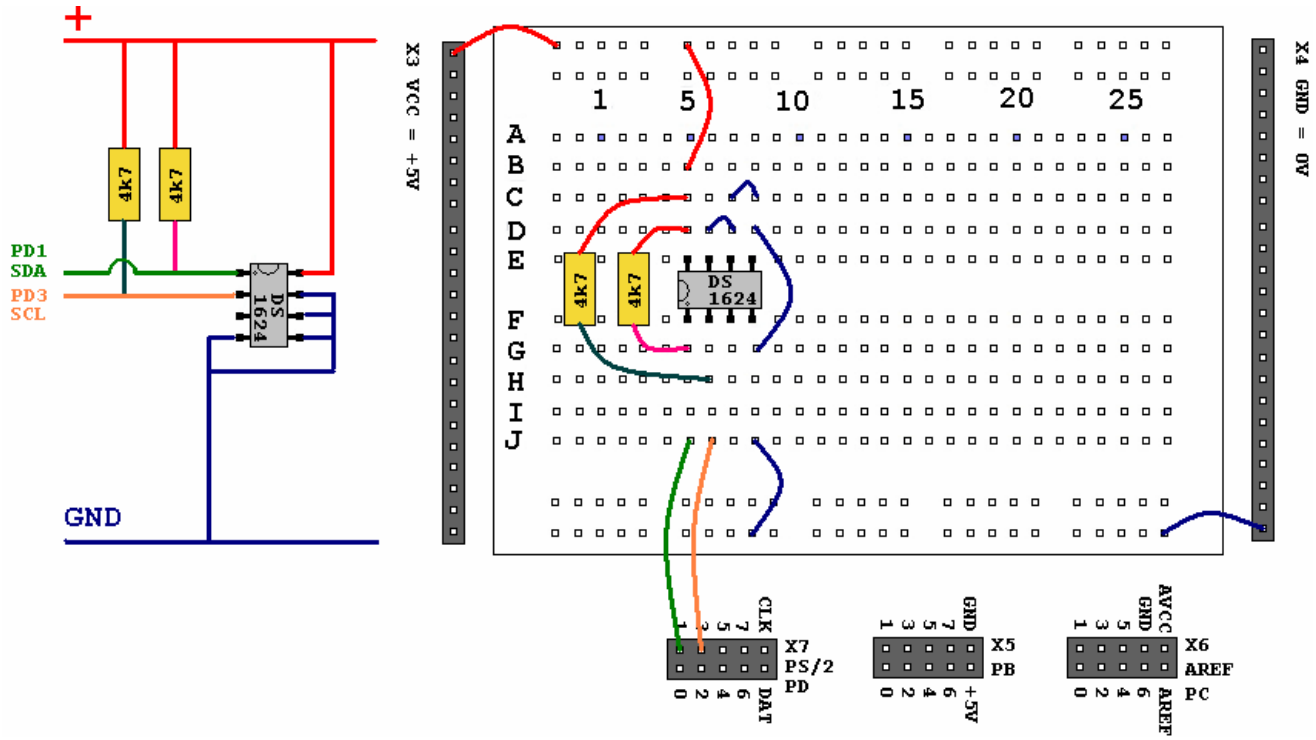
Ziele
- Benutzung von I²C

Theorie

Das DS1624 ist ein digitales Thermometer mit einem Messbereich von –55 Grad Celsius bis +125 Grad Celsius. Es wird mit Hilfe des I2C Bus angeschlossen. Der I²C-Bus hat ein serielles 2-Draht-Protokoll. Die 2 Drähte sind Serial DAta (SDA,Serielle Daten) und Serial CLock (SCL, Serieller Takt).

In I2C wird der Mikrocontroller als MASTER (Meister) bezeichnet, das DS1624 ist ein slave (Sklave), Der Master ist das Gerät welches den Seriellen Takt (SCL) erzeugt. Man kann mehrere slaves anschliessen, jedoch nur einen Master.

Bitte bauen Sie folgende Schaltung auf:



I²C (2-wire serial data bus) Theorie

Wenn Sie mehr über die Theorie des I²C-Busses erfahren möchten, schlagen wir Ihnen vor das DS1307 Datenblatt von Maxim / Dallas semiconductor zu lesen. Der DS1307 ist zwar nicht der gleiche Chip wie der in unserem Experiment benutzte, aber beide nutzen das gleiche Protokoll. Das Datenblatt des DS1307 ist besser geeignet um den I²C-Bus zu verstehen.

Das DS1307-Datenblatt befindet sich auf der EDB-CD oder kann von www.maxim-ic.com geladen werden. Bitte beginnen Sie auf Seite 5 bei Abschnitt „2-WIRE SERIAL DATA BUS“ und lesen Sie nicht weiter als bis Seite 9. (Denken Sie daran, wir benutzen einen anderen Chip. Das Prinzip ist das Gleiche, aber die Adress- und Befehls-Bytes sind verschieden)

Programmieren Sie den ATMega88 mit EDBexperiment19.bas und verbinden Sie das Board und Ihren PC mit dem seriellen Kabel.

Sie sollten nun in der Lage sein, die Temperatur in ihrem Terminal abzulesen.

Lesen Sie die Anmerkungen in der Datei EDBexperiment19.bas und sehen Sie ob Sie verstehen was passiert.

Online Datenquelle:

I2C Background:
http://www.semiconductors.philips.com/markets/mms/protocols/i2c/

I2C Interface Support in Windows:
http://www.microsoft.com/whdc/archive/i2c.mspx

3.8.2 Die USB-Schnittstelle

Obwohl heutzutage die meisten Computer mit USB-Anschlüssen ausgestattet sind, gehen die Entwickler nicht oft den Schritt und implementieren USB in ihr System. Dieses Kapitel gibt Ihnen eine Einführung in die USB-Entwicklung.

Als Erstes sollten Sie wissen, dass Sie 3 Möglichkeiten haben, Ihre eigene Hardware mit USB auszurüsten:

1. **Das USB-Gerät als virtuellen COM-Anschluss**
   Das ist die einfachste Möglichkeit, wenn Sie USB möchten, Sie verbinden Ihr Gerät mit dem USB-Anschluss des Computers und das Betriebssystem behandelt Ihr Gerät als wenn es ein COM-Anschluss wäre.

   Vorteile:
   - einfach
   - kompatibel mit Terminalprogrammen und existierender Software

   Nachteile:
   - der Nutzer sieht den COM-Anschluss, was nicht sehr professionell ist
   - der Com-Anschluss kann falsch konfiguriert werden
   - der Nutzer muss 2 Treiber installieren

2. **Das USB-Gerät mit vorgegebener VID&PID und Treiber**

   Im allgemeinen: Anwendungen greifen nicht auf USB-Anschlüsse zu, sie greifen auf Geräte mit einer bestimmten VID & PID (Vendor & Part ID, Hersteller- und Teile-Identifikationsnummer) auf dem Bus zu.

   Bei dieser zweiten Möglichkeit können Sie die von Linx Technologies (www.linxtechnologies.com) vertriebenen Treiber nutzen, um auf Ihr Gerät zuzugreifen.

   Vorteile:
   - relativ einfach zu benutzen

   Nachteile:
   - Sie müssen die von Linx registrierte VID&PID benutzen

3. Das USB-Gerät mit Ihrem eigenen VID&PID und Treiber

Bei dieser Möglichkeit können Sie Ihr eigenes Gerät entwerfen. Wenn Sie Ihre PID von MCS Electronics kaufen, erhalten Sie zusätzlich ein Programm, welches die .inf Installationsdatei für Sie schreibt. Wenn Sie sich dafür entscheiden, ihre PID nicht bei MCS zu kaufen, bedenken Sie, daß Sie selbst die *.inf-Datei schreiben müssen.

Vorteile:
- Sie können Ihre eigene VID&PID benutzen

Nachteile:
- Sie müssen Ihre VID von MCS oder dem USB-Consortium kaufen
- Sie müssen Ihre eigene *.inf-Datei schreiben

USB-Geräte werden primär dazu verwendet, mit Computern zu interagieren, deshalb müssen Sie fast immer die Anwendungen für den Computer zusätzlich zu Ihrem Mikrocontroller-Code schreiben. Das und die Kosten des USB-Moduls (und der optionalen VID) kann hinderlich für die Implementation von USB sein. In der Zukunft werden COM-Anschlüsse für Computersysteme nicht mehr verfügbar sein und die USB-Module werden erschwinglicher werden. Aus diesem Grunde haben wir das Kapitel über USB in das EDB-Handbuch aufgenommen.

Eine umfangreiche Lektüre über den USB-Bus liegt außerhalb des Zieles dieses Handbuchs. (www.xyz) Wir werden beschreiben wie man Das EDB wie bei Möglichkeit 1 und 2 beschrieben anschließt. Wir werden weiterhin einige Beispiele in Visual Basic 6 und Windows Scripting aufzeigen.

Die 3. Möglichkeit wird nicht weiter beschrieben, wir werden nur ein Werkzeug vorstellen, um die EEPROMS des USB-Moduls zu beschreiben und so die VID&PID zu ändern.

Online Datenquelle:

USB module: http://www.linxtechnologies.com/
USB chip inside Linx module:
http://www.ftdichip.com/Products/FT232BM.htm
USB general : www.usb.org
Get your own product ID: www.mcselec.com

3.8.3 Installationsprozedur

Wenn Sie Möglichkeit 1 nutzen möchten (virtueller COM-Anschluß) folgen Sie bitte der installation wie in Kapitel 3.8.3.1 beschrieben. Für die anderen Möglichkeiten folgen Sie den Anweisungen des Kapitels 3.8.3.2.

Wenn Sie von Möglichkeit 1 (virtueller COM) zu Möglichkeit 2 oder 3 wechseln möchten, bzw. von Möglichkeit 2 oder 3 zu 1, müssen Sie als erstes Ihren alten Gerätetreiber entfernen. Die notwendigen Anweisungen hierfür finden Sie in Kapitel 3.8.3.3.

Vergewissern Sie sich, dass Ihr Computer über einen funktionsfähigen USB-Anschluss verfügt. USB-Anschlüsse funktionieren nicht unter Windows NT und einigen Windows-95 Versionen. Wir raten dazu, immer den gleichen USB-Anschluss zu verwenden wenn Sie Ihr EDB anschliessen, auf diese Weise vermeiden Sie es, den Treiber neu installieren zu müssen.

3.8.3.1 Windows-Installation des virtuellen COM-Anschlusses

A type

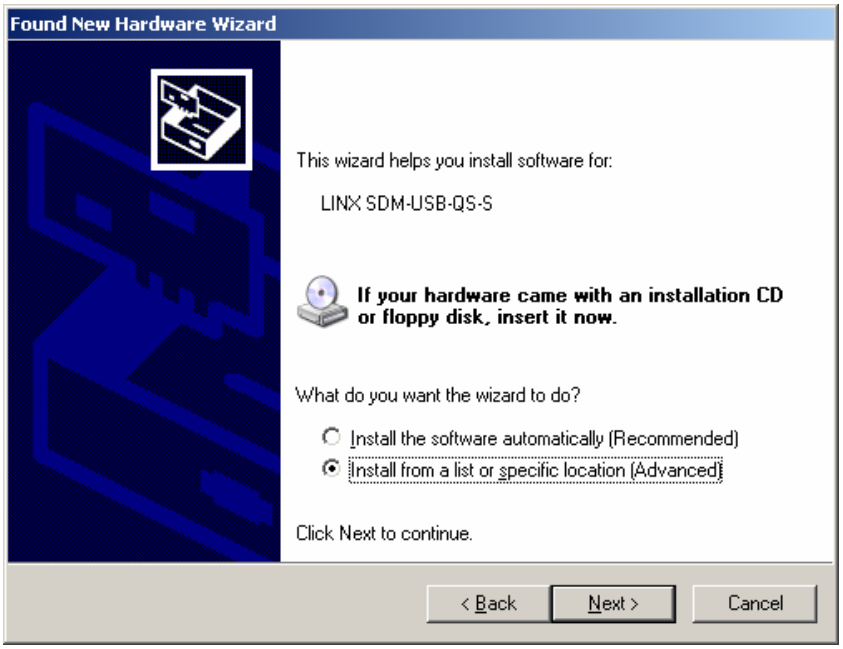
B type

Verbinden Sie das EDB mit der Stromversorgung und das USB-A zu USB-B-Kabel mit dem EDB und dem Computer.

Sie sollten nun folgende Meldung sehen:

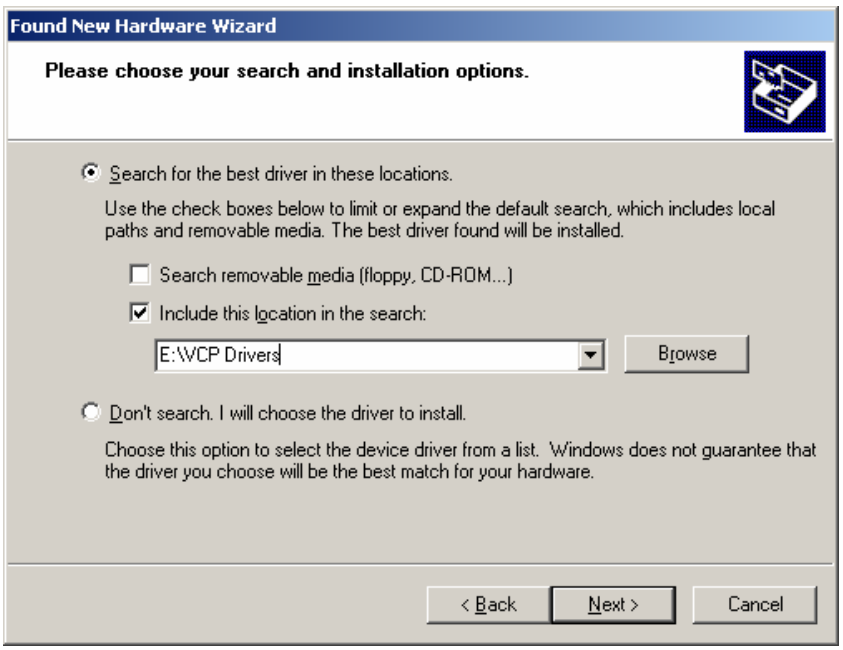
Wählen Sie „Nein, diesmal nicht“ und klicken Sie auf „Weiter“

(nach einem Moment) sollten Sie folgende Meldung sehen:

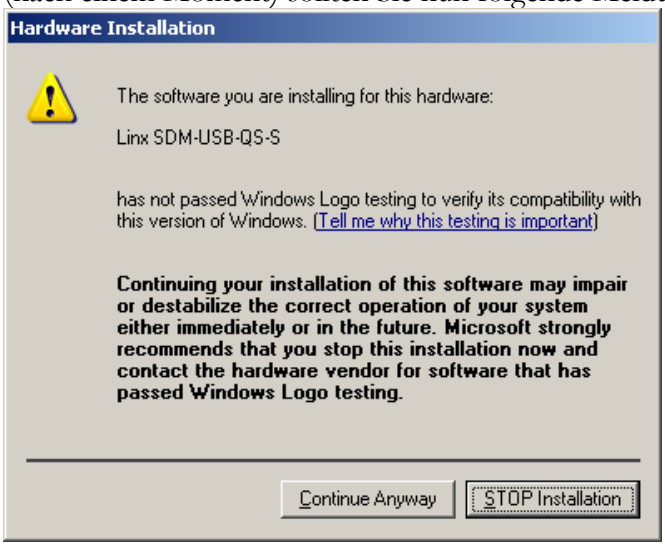


Wählen Sie „Installieren aus einer Liste oder einem bestimmten platz“ und klicken Sie „Weiter“

(nach einem Moment) sollten Sie folgende Meldung sehen:



Legen Sie nun die EDB-CD ein und wählen Sie „Durchsuchen“ um folgenden Ordner einzustellen:
IhrCDLaufwerk:\USB_Module\Virtual_COM_Drivers\
Und klicken Sie auf “Weiter”
(nach einem Moment) sollten Sie nun folgende Meldung sehen:



Dieses Fenster ist einfach nur eine Warnung dass der Treiber nicht den Zertifikationsprozess von Microsoft absolviert hat und möglicherweise ein problem für das System erzeugen könnte. Der mit dem USB-Modul ausgelieferte Treiber wurde unabhängig getestet und sollte keinerlei Probleme auslösen, solange er nicht vom Nutzer modifiziert wird. Klicken Sie auf „Trotzdem fortfahren“

Nach einem Moment erscheint ein Fenster mit der Mitteilung „Der Wizzard hat die Installation der Software für Linx SDM-USB-QS-S beendet“. Klicken Sie auf „Beenden“

Ziehen Sie jetzt nicht den USB- oder Stromversorgungs-Stecker, die Installation ist noch nicht beendet.

Der Computer führt nun erneut eine Hardwareerkennung durch. Normalerweise findet er nun den virtuellen COM-Port. Um den virtuellen COM-Port zu installieren, müssen Sie nun alle aufgezeigten Schritte auf den vorhergegangenen Seiten wiederholen (beginnend von 3.8.3.1 „Found new hardware wizzard“)

Der virtuelle COM-Port ist nun installiert

Fahren Sie bei Kapitel 3.8.4 fort

3.8.3.2 Windows-installation des USB-Gerätes

A type

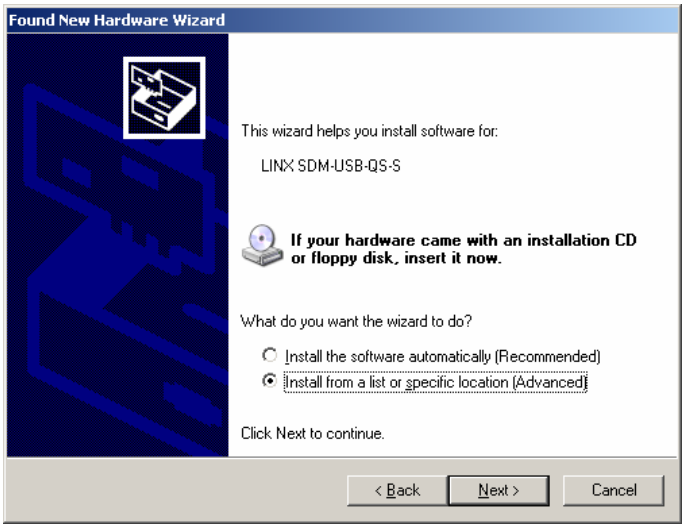
B type

Verbinden Sie das EDB mit der Stromversorgung und das USB-A zu USB-B-Kabel mit dem EDB und dem Computer.

Sie sollten nun folgende Meldung sehen:

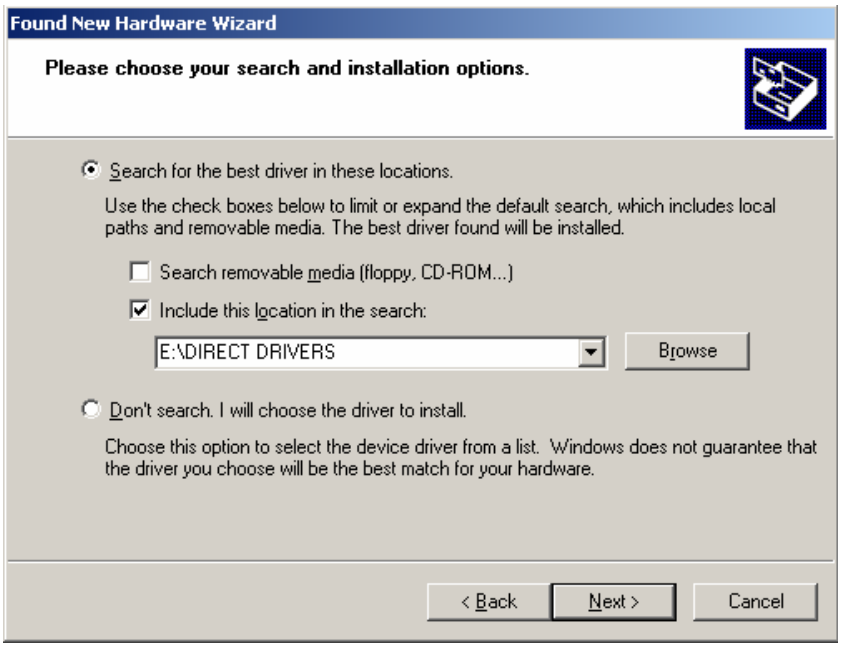
Wählen Sie „Nein, diesmal nicht“ und klicken Sie auf „Weiter“

(nach einem Moment) sollten Sie folgende Meldung sehen:



Wählen Sie „Installieren aus einer Liste oder einem bestimmten platz“ und klicken Sie „Weiter“

(nach einem Moment) sollten Sie folgende Meldung sehen:



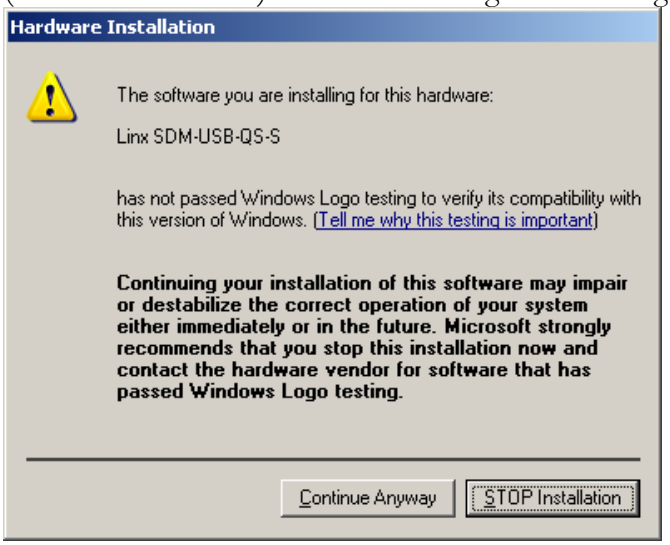
Legen Sie nun die EDB-CD ein und wählen Sie „Durchsuchen“ um folgenden Ordner einzustellen:

Ihr CD Laufwerk: \USB_Module\Direct_Drivers\

Und klicken Sie auf “Weiter”

Wenn Sie Ihre eigene VID&PID programmiert haben, müssen Sie ihren selbstgeschriebenen Treiber hier angeben

(nach einem Moment) sollten Sie nun folgende Meldung sehen



Dieses Fenster ist einfach nur eine Warnung dass der Treiber nicht den Zertifikationsprozess von Microsoft absolviert hat und möglicherweise ein

problem für das System erzeugen könnte. Der mit dem USB-Modul ausgelieferte Treiber wurde unabhängig getestet und sollte keinerlei Probleme auslösen, solange er nicht vom Nutzer modifiziert wird. Klicken Sie auf „Trotzdem fortfahren“

Nach einem Moment erscheint ein Fenster mit der Mitteilung „Der Wizard hat die installation der Software für Linx SDM-USB-QS-S beendet“. Klicken Sie auf „Beenden“

Das USB-Gerät ist nun installiert

Fahren Sie bei Kapitel 3.8.5 fort

3.8.3.3 Treiber Deinstallation

Wenn Sie von Möglichkeit 1 (virtueller COM) nach Möglichkeit 2 oder 3 wechseln wollen, oder von 2 oder 3 nach 1, müssen Sie zuerst Ihren alten Gerätetreiber entfernen. Dieses Kapitel sagt Ihnen wie.

Legen Sie zuerst Ihre EDB CD ein.

Wenn Sie den virtuellen COM Treiber installiert haben
Nutzen Sie den Dateimanager um zu Ihrem CD ROM Laufwerk zu gelangen. Öffnen Sie den „VCP Drivers folder" und öffnen Sie „FTDIUNIN.exe"

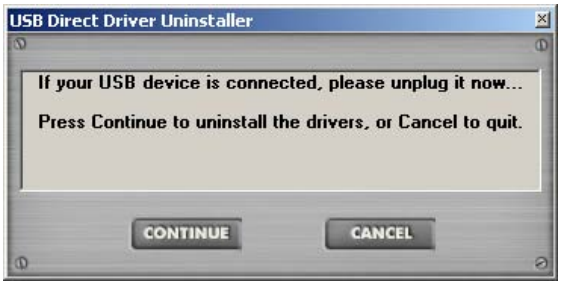
USB_Module\Virtual_COM_Drivers\FTDIUNIN.exe

Sie sollten folgendes sehen:

Klicken Sie auf „Continue" und nach einem Augenblick auf „Finish". Wenn Sie den „direct driver" installieren möchten, schlagen Sie bitte in Kapitel 3.8.3.2 nach.

Wenn Sie den „Direct driver" installiert haben
Nutzen Sie den Dateimanager um zu Ihrem CD ROM Laufwerk zu gelangen. Öffnen Sie „Direct Drivers" und öffnen Sie „FTD2XXUN.exe"

USB_Module\Direct_Drivers\FTD2XXUN.exe Sie sollten folgendes sehen:



Klicken Sie auf „Continue" und nach einem Augenblick auf „Finish". Wenn Sie den „virtuellen COM Anschluss Treiber" installieren möchten, schlagen Sie bitte in Kapitel 3.8.3.1 nach.

Versuch
20

3.8.4 USB Schnittstelle als virtueller COM Port

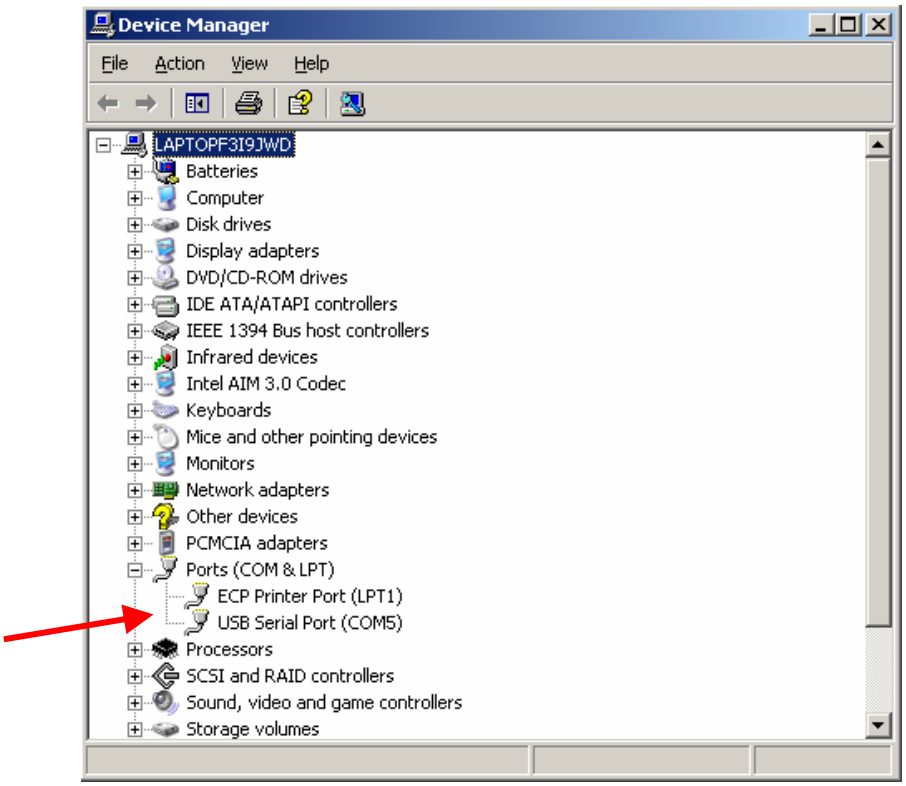
Stückliste
1x USB-A zu USB-B Kabel
1x PC mit USB Anschluss

Ziele
- Ein Gerät als virtuellen COM Port betreiben

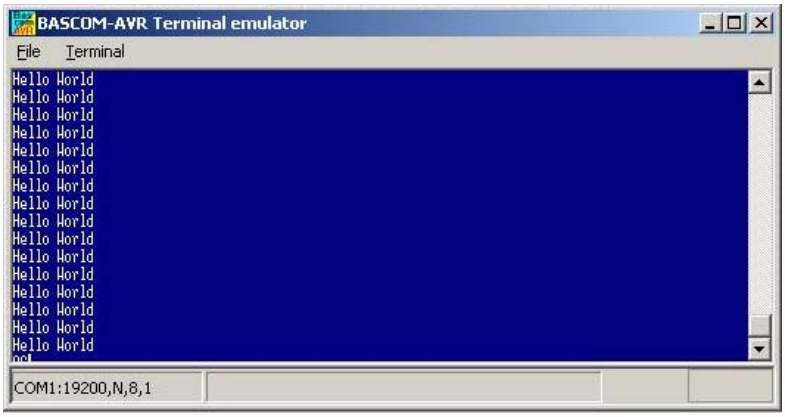
Dieser Versuch kann nur durchgeführt werden, wenn Sie die Treiber wie in Kapitel 3.8.3.1 beschrieben installiert haben. Stellen Sie sicher, daß der Schalter „S1 USB/RS232“ auf USB gesetzt ist.

Öffnen Sie die Datei EDBexperiment6a.bas von der EDB CD und laden Sie diese in den ATMega88. Wir benutzen das gleiche Programm wie in Versuch 6a, um den virtuellen COM Anschluss zu testen. Nachdem Sie den Controller programmiert und das USB Kabel angeschlossen haben, öffnen Sie den Terminal Emulator, indem Sie auf diesen Knopf im Bascom clicken. 

Die Einstellungen sollten lauten: 19200 baud, no parity, 8 data bits, 1 stop bit. Stellen Sie sicher, daß Sie den entsprechenden (virtuellen) COM Anschluss verwenden. Sie finden den entsprechenden COM Anschluss in Ihrer Windows Konfiguration (System, Hardware, Gerätemanager).



In diesem Fall sollten Sie COM5 benutzen. In Ihrem System könnte es jedoch ein anderer Anschluss sein
Sie sollten jetzt sehen, wie das EDBexperiment6a.bas Programm „Hello World“ auf Ihrem Computer Bildschirm wie folgt anzeigt:



Es besteht die Möglichkeit, daß Sie Ihre eigene Anwendungssoftware für Windows/Linux schreiben. Wenn Sie Windows benützen, können Sie Visual Basic verwenden (Basic wie im Bascom AVR). Sie können auch eine Programmiersprache benützen, welche andere Betriebssysteme unterstützt, zum Beispiel C++ Borland Builder, Pascal oder jede andere Programmiersprache, welche die Benutzung von ActiveX Komponenten unterstützt.

Wir benutzen eine ActiveX (Software-) Komponente, um Daten vom COM Anschluss an die Anwendung in Windows/Linux zu senden oder zu empfangen. Ich empfehle Ihnen die Benutzung der „royalty free OCX“ Komponente (mcscomm.ocx), welche von MCS Electronics bereit gestellt wird. (Microsoft stellt mscomm32.ocx bereit, aber diese Komponente funktioniert für diese Anwendung nicht)

Sie können Beispielprogramme im Ordner „Visual Basic“ auf der EDB CD finden. Die mcscomm.ocx Datei befindet sich ebenfalls in diesem Ordner.

Versuch
21

3.8.5 USB Schnittstelle mit standardmässigem PID&VID

Stückliste

1x USB-A zu USB-B Kabel
1x PC mit USB Anschluss

Ziele

- Ein Gerät mitstandard-PID &VID betreiben

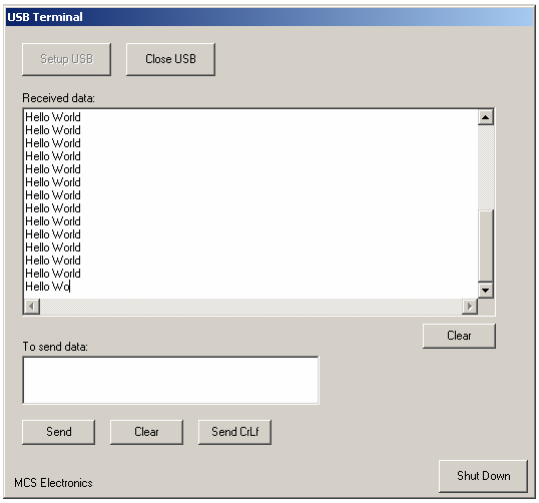
Dieser Versuch kann nur durchgeführt werden, wenn die Treiber wie in Kapitel 3.8.3.2 beschrieben installiert sind.

Stellen Sie sicher, daß der Schalter „S1 USB/RS232“ auf dem EDB auf USB gesetzt ist.

Öffnen Sie die Datei EDBexperiment6a.bas von der EDB CD und programmieren Sie diese in den ATMega88. Wir benutzen das gleiche Programm wie in Versuch 6a um die USB Schnittstelle mit PID & VID zu testen.

Öffnen Sie nun das „USB Terminal“ Programm aus dem Ordner „Visual_Basic\USB_Terminal“ auf der EDB CD.

Klicken Sie auf „setup USB“. Sie sollten nun sehen, wie das Programm „Hello World“ auf dem Computerbildschirm anzeigt. Versuchen Sie auch EDBexperiment6b.bas.



Es besteht die Möglichkeit, daß Sie Ihre eigene Anwendungssoftware für Windows/Linux schreiben. Wenn Sie Windows benützen, können Sie Visual Basic verwenden (Basic wie im Bascom AVR). Sie können auch eine Programmiersprache benützen, welche andere Betriebssysteme unterstützt, zum Beispiel C++ Borland Builder, Pascal oder jede andere Programmiersprache, die die Benutzung von DLL Komponenten unterstützt.

Wir benutzen eine DLL (Software-) Komponente, um Daten vom USB Port an eine Anwendung in Windows/Linux usw. zu senden bzw. zu empfangen.

Sie können Beispielprogramme im Ordner „Visual Basic“ auf der EDB CD finden. Die DLL Datei befindet sich auch in diesem Ordner. Weitere Beispiele in anderen Programmiersprachen finden Sie auf der FTDI Webseite. http://www.ftdichip.com/Projects/CodeExamples.htm.

3.8.6 USB Schnittstelle mit eigenem PID&VID

Dieses Kapitel ist nur für fortgeschrittene Entwickler gedacht. Das Kapitel gibt nur einen kurzen Überblick über die erweiterten Möglichkeiten. Es ist kein Schritt-für-Schritt Ratgeber welcher Ihnen alle Schritte, welche zum Erstellen Ihres eigenen PID-Gerätes notwendig sind, aufzeigt.

WARNUNG: Das Verändern des EEPROM (PID&VID) des USB-Moduls kann dazu führen, dass Sie nicht mehr auf den Chip zugreifen können. Änderungen des EEPROM und des Treibers geschehen auf Ihre eigene Verantwortung und Ihr eigenes Risiko. MCS Electronics gibt keine Unterstützung für Anfragen, welche aufgrund von EEPROM Änderungen oder modifizierten Treibern entstehen.

Die Programmierung des EEPROM kann nur durchgeführt werden, wenn Sie die Treiber wie in Kapitel 3.8.3.2 beschrieben installiert haben.

Das USB Interface ist mit einem SDM-QS-S1-S USB Modul von Linx technologies bestückt, dessen Hauptbestandteil ein FT232xx Chip von FTDI ist.

Sie finden die Linx Webseite unter http://www.linxtechnologies.com/
Sie finden die FTDI Webseite unter http://www.ftdichip.com/

Den EEPROM des USB SDM-QS-S1-S USB Moduls können Sie sowohl lesen als auch beschreiben. Das eröffnet ihnen die Möglichkeit, die PID&VID des Moduls zu ändern.

Wir empfehlen Ihnen eindringlich, die oben angegebenen Webseiten zu besuchen und die Handbücher der Hersteller zu lesen, bevor Sie jedwede EEPROM Programmierung beginnen.
Sie finden die Anwendung zur Programmierung der EEPROM im Ordner „USB Module“ auf der EDB CD, aber auch auf den Webseiten der Hersteller.

Bevor Sie Ihr eigenes PID Gerät mit Windows/Linux usw. nutzen können, müssen Sie zuerst die *.inf Gerätetreiber-Dateien modifizieren. Nehmen Sie die *.inf-Dateien, welche sich auf der EDB CD befinden als Beispiele.

Online Datenquelle:
Get your own product ID: www.mcselec.com

Versuch
22

3.9 Motoren

Außer der Mensch-Maschine-Schnittstelle und Datenkommunikation, stellen Mikrocontroller eine Schnittstelle für Stellmotoren bereit. Die gebräuchlichsten Stellmotoren sind Gleichstrom- und Schrittmotoren. Dieses Kapitel zeigt Ihnen, wie Sie Gleichstrom- und Schrittmotoren verwenden.

3.9.1 Schrittmotoren

Stückliste
1x Bi-polarer Schrittmotor
1x Quasar Schrittmotor-Steuerung

Ziele
- Benutzung eines Schrittmotors

Für diesen Versuch nutzen wir die Quasar Schrittmotor Steuerung 3158 für Bi-polare Schrittmotoren. Weitere Informationen finden Sie auf der Quasar Webseite www.quasarelectronics.com/kit-files/3000/3158.pdf.

Falls Sie keinen Schrittmotor besitzen, können Sie einen Schrittmotor aus einem Diskettenlaufwerk benützen. Diese sind meist nicht Bi-polar, deswegen sind sie nur zum experimentieren geeignet (in Verbindung mit dieser Steuerung), aber nicht für die Nutzung in einer realen Anwendung.

Verbinden Sie den Schrittmotor mit dem „TO MOTOR“ Anschlüssen auf dem Steuerungsplatine.

Verbinden Sie Portd.0 mit dem „STEP +“ Pin,
verbinden Sie Portd.1 mit dem „DIR +“ Pin,
verbinden Sie GND vom EDB mit „STEP –„ und „DIR –„

Schliessen Sie 8....30V an beide Stromanschlüsse der Steuerungsplatine an. Beginnen Sie mit einer geringen Spannung falls Sie die Spezifikationen des Motors nicht besitzen (eine galvanische Trennung ist möglich, für Details schauen Sie in die 3158 pdf Datei).

Wenn Sie den Schrittmotor eines Diskettenlaufwerks benutzen, haben Sie möglicherweise nur 3 Drähte, um den Motor anzuschliessen. In diesem Fall lassen Sie den „TO MOTOR“ Anschluss Pin frei.

Programmieren Sie den ATMega88 mit EDBexperiment22.bas. Sie sollten nun sehen, wie der Motor sich zuerst in die eine, dann in die andere Richtung bewegt.

Lesen Sie die Anmerkungen in der Datei EDBexperiment22.bas und sehen Sie, ob Sie verstehen, was passiert.

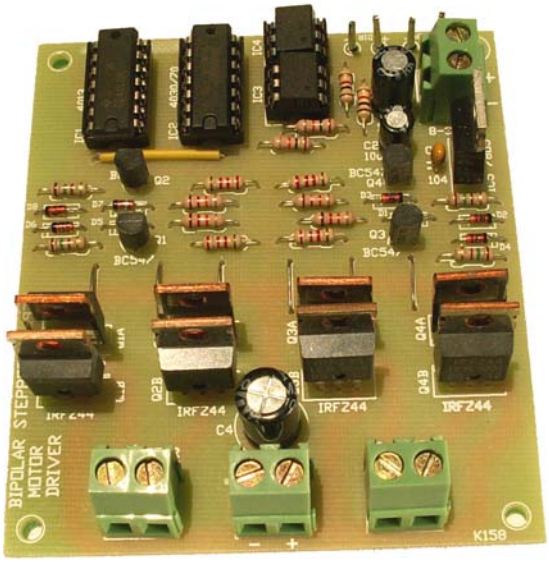
Versuch
23

3.9.2 PWM gesteuerter Gleichspannungsmotor

Stückliste
1x Gleichspannungsmotor
Gleichspannungsmotors
1x Quasar Schrittmotor-Steuerung

Ziele
- Benutzen eines

Für diesen Versuch nutzen wir die Quasar Schrittmotor Steuerung 3158 für Bi-polare Schrittmotoren. Weitere Informationen finden Sie auf der Quasar Webseite www.quasarelectronics.com/kit-files/3000/3158.pdf.



Falls Sie keinen Gleichstrommotor besitzen, können Sie einen Gleichstrommotor aus einem Kasettendeck verwenden.

Verbinden Sie den Gleichstrommotor mit einem der „TO MOTOR“ Anschlüssen auf der Steuerung.

Verbinden Sie Portd.0 mit dem „STEP +“ Pin,
verbinden Sie Portd.1 mit dem „DIR +“ Pin,
verbinden Sie GND vom EDB mit „STEP –„ und „DIR – „

Schliessen Sie 8....30V an beide Stromanschlüsse der Steuerungsplatine an. Beginnen Sie mit einer geringen Spannung falls Sie die Spezifikationen des Motors nicht besitzen (eine galvanische Trennung ist möglich, für Details schauen Sie in die 3158 pdf Datei)

Programmieren Sie den ATMega88 mit EDBexperiment22.bas. Sie sollten nun sehen, wie der Motor sich zuerst in die eine, dann in die andere Richtung bewegt.

Lesen Sie die Anmerkungen in der Datei EDBexperiment22.bas und sehen Sie, ob Sie verstehen, was passiert.

Online Datenquelle:
Motor driver:
http://www.quasarelectronics.com/motor_controllers_drivers.htm

Kapitel

4

4. Andere Programmiermethoden

Dieses Kapitel erklärt, wie man das ATMega88 mit Hilfe des parallelen oder USB-Anschlusses programmiert.

Sie können das ATMega88 auch mit Hilfe des parallelen oder des USB-Anschlusses programmieren.

Es gibt 3 Gründe dafür, das zu tun:

1. Sie haben keinen seriellen Anschluss
2. Sie wollen einen leeren / gelöschten Microcontroller programmieren
3. Sie wollen den gesamten Speicher programmieren.
   Das ist mit einem bootloader nicht möglich, weil der Bootloader selbst schon Speicher auf dem Controller belegt.

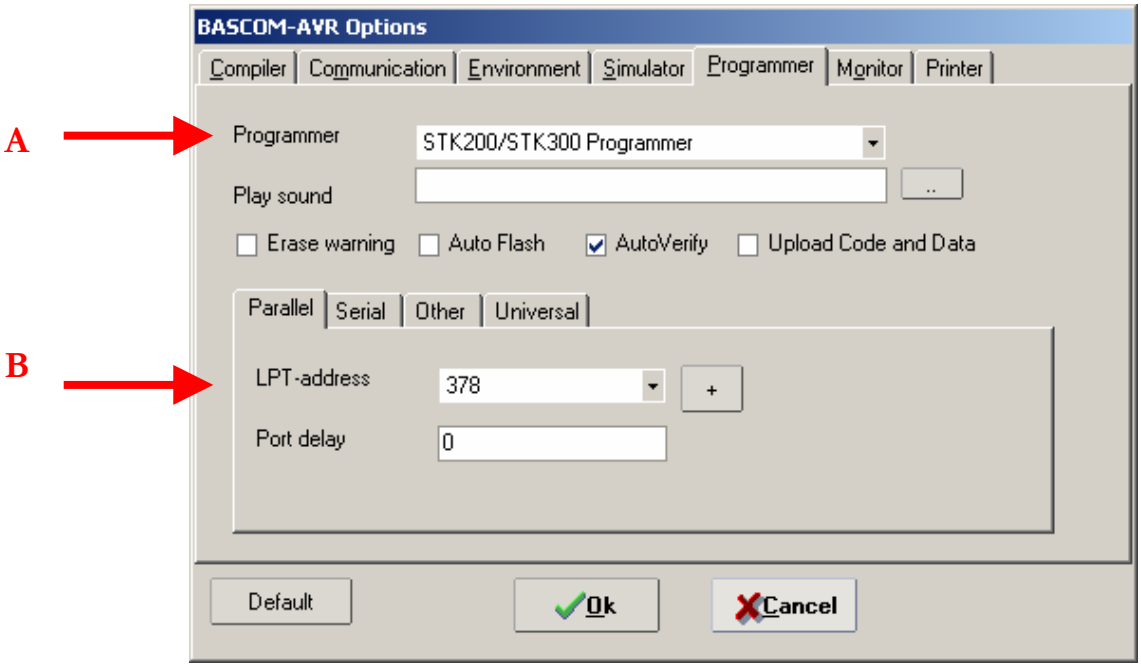
Kapitel 4.3 beschreibt, wie man den Bootloader umprogrammiert.

4.1 Programmieren mit dem STK200/300 Dongle

Sie können einen STK200/300 Dongle von MCS Electronics erwerben, Sie können sich aber auch ihren eigenen bauen. Der Schaltplan des STK Dongle befindet sich im Anhang 1.

Bevor Sie fortfahren, vergewissern Sie sich bitte, dass im BIOS der parallele Anschluss auf ECP Modus eingestellt ist. Der Dongle arbeitet nicht in anderen Modi, beispielsweise im bi-direktionalen Modus.

Verbinden Sie nun den STK200/300 Dongle mit dem parallelen Anschluss Ihres Computers und das flache Kabel des STK200/300 mit dem Anschluss X9 (mit ISP gekennzeichnet) am Ausbildungs- und Entwicklungsboard.
Im Bascom klicken Sie auf „Options“ und „Programmer“. Nun sollten Sie folgendes auf dem Bildschirm sehen:



- Wählen Sie zuerst den „STK200/300 Programmer“ im Drop-Down-Feld (A)
- Dann wählen Sie Ihre LPT-Adresse. Wenn Sie LPT1 benutzen ist diese standardmäßig „378“ und für LPT2 wählen Sie „278“
- Drücken Sie nun auf OK

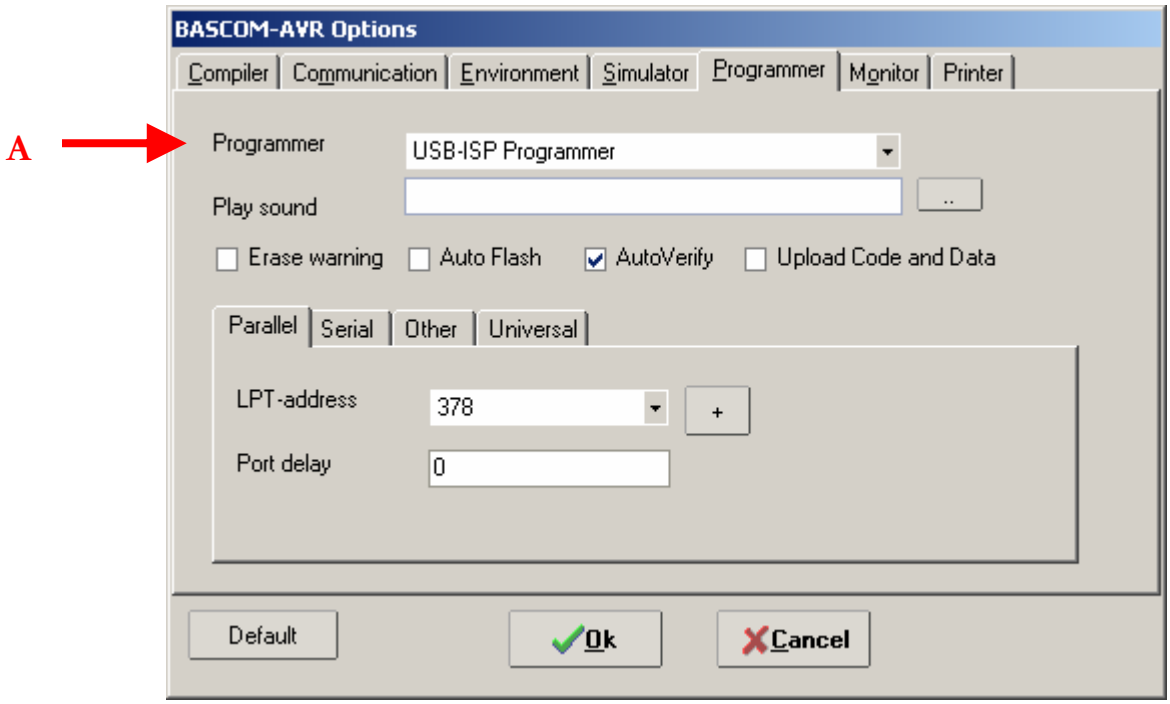
Wenn Sie den Bootloader umprogrammieren wollen fahren Sie bitte bei Kapitel 4.3 fort, anderenfalls können Sie Ihren Code wie im Kapitel 2.3 beschrieben programmieren, beginnend bei Zeile **„Flashprogrammierung“**

4.2 Programmieren mit dem Wiazania USP ISP

Sie können einen Wiazania USB ISP Programmier-Dongle von MCS-Electronics erwerben. Der USB ISP Programmier-Dongle hat die gleichen Möglichkeiten wir der STK200/300 Dongle, er nutzt nur statt des Parallelanschlusses den USB-Anschluss.

Verbinden Sie den USB-ISP Programmierer mit einem des USB-Anschlüsse Ihres PCs, verbinden Sie dann das Flachkabel des USB ISP mit dem Anschluss X9 (mit ISP gekennzeichnet) am Ausbildungs- und Entwicklungsboard.

Im Bascom klicken Sie auf „Options“ und „Programmer“. Nun sollten Sie folgendes auf dem Bildschirm sehen:



- Wählen Sie „USB-ISP Programmer“ im dem Drop-Down-Feld (A)
- Drücken Sie auf OK

Wenn Sie den Bootloader umprogrammieren wollen fahren Sie bitte bei kapitel 4.3 fort, anderenfalls können Sie Ihren Code wie im Kapitel 2.3 beschrieben programmieren, beginnend bei Zeile **„Flashprogrammierung“**

4.3 Umprogrammieren des Bootloaders

Unter Nutzung der Programmiermethoden der Kapitel 4.1 oder 4.2

- Öffnen Sie die Datei Bootloader.bas (EDB-CD) im Bascom AVR
- Kompilieren Sie den Bootloader (F7 drücken)
- Vergewisseren Sie sich, dass Sie einen der STK-Dongles - beschrieben in Kapitel 4.1 und 4.2 – nutzen und programmieren Sie mit Taste F4 den chip
- Programmieren Sie die Sicherungs-Bits des ATMega88 wie unten beschrieben
- Zuletzt wählen Sie „MCS Bootloader“ im Feld „Programmer“ wie in Kapitel 2.3 beschrieben

Programmieren der Sicherungs-Bits

Die ATMega AVR-Familie nutzt Sicherungs-Bits um erweiterte Features des Chips zu setzen.
Unter normalen Umständen werden diese Bits nur einmal programmiert, obwohl das auch viele Male möglich ist. Sie können diese Sicherungs-Bits mit Hilfe der in Bascom eingebauten Programmier-routine programmieren.

Sie müssen die folgenden zwei Schritte auf dieser Seite nur ausführen, wenn eine der folgenden Bedingungen zutrifft:

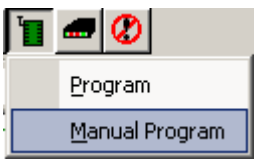
1. Sie benutzen einen nicht von MCS Electronics vertriebenen ATMega88, oder Sie haben die Sperr-Bits an einem von MCS Electronics vertriebenen Chip geändert
2. Das Programmieren des Chips wie auf der vorhergehenden Seite beschrieben hat nicht funktioniert

Sie benötigen einen STK200/300 Dongle oder einen Wiazania USB Programmierer um die Sicherungs-Bits zu programmieren.

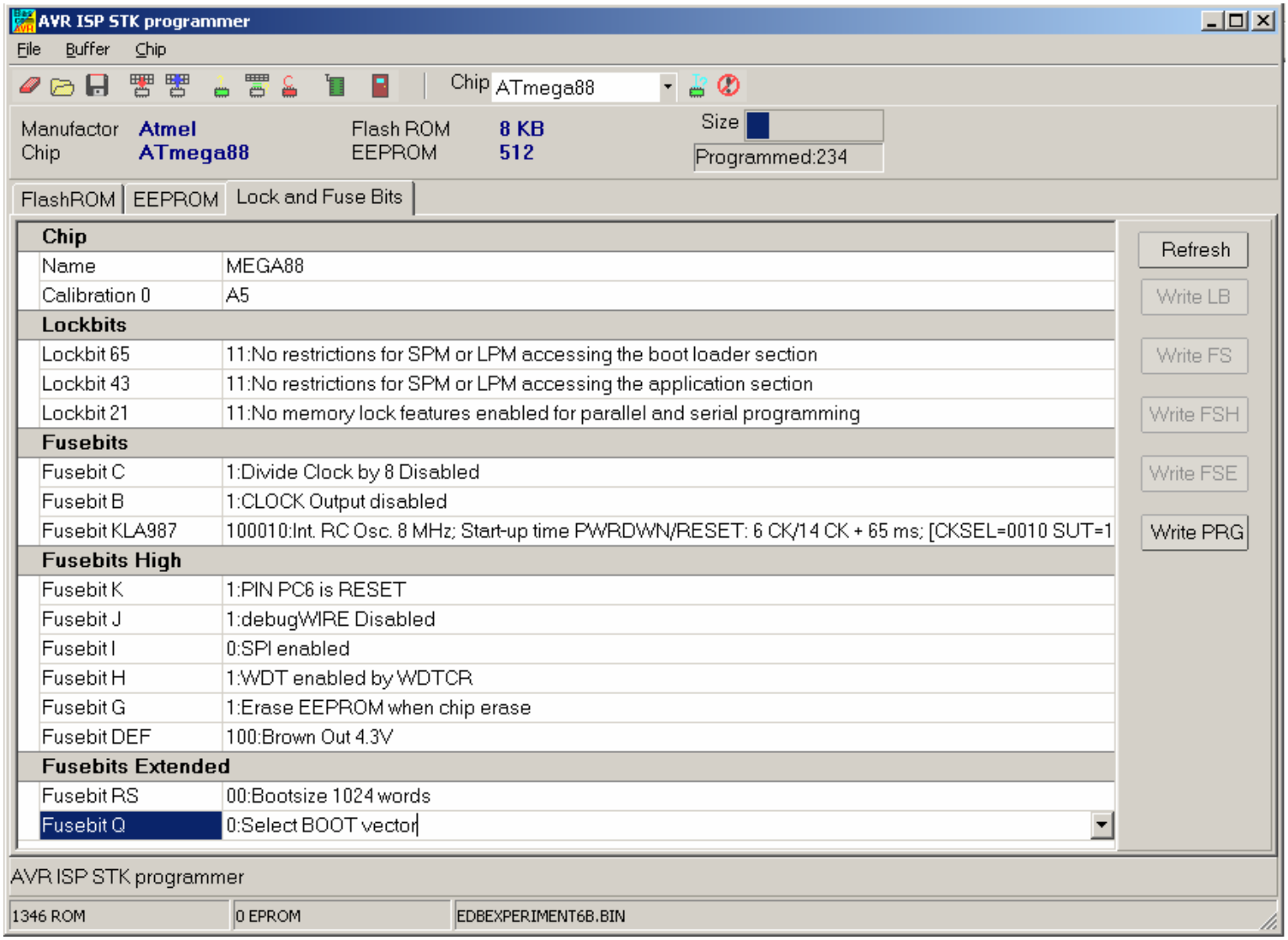
Diese Schritte werden die Sicherungs-Bits für die Nutzung des ATMega88 auf dem Ausbildungs- und Entwicklungs-Board zurücksetzen.

Öffnen Sie das Bascom DIE, dann öffnen Sie eine *.bas oder eine leere Datei

Öffnen Sie den Programmierer wie folgt:



Wählen Sie den „Lock and Fuse Bits“ Reiter und maximieren Sie das Programmierer-Fenster:



Vergewissern Sie sich, dass die Sicherungs- und Sperrbits wie oben zu sehen gesetzt sind, drücken Sie „Refresh“ um die aktuellen Einstellungen zu sehen. Sicherungsbit Q sollte auf BOOT gesetzt sein.

WARNUNG, richten Sie Ihre Aufmerksamkeit auf Sicherungsbit KLA987, welches die Optionen für den Oszillator setzt, es sollte den Wert „100010: Int. RC Osc... SUT = 1“ haben. Wenn Sie diese Opion verändern, wird das Gerät nicht mehr funktionieren und Sie können es nicht ohne einen externen Oszillator (Frequenzgenerator) zurücksetzen. Vergewissern Sie sich, dass ausschliesslich 100010 eingestellt ist.

Sie können die Bits mit dem „Write XXX“ Knopf schreiben, der erst verfügbar wird, nachdem Sie eine Option (Drop-Down-Feld) geändert haben und eine andere fuse/lock bits sektion angeklickt haben.

Atmel AVR In System Programming Dongle

Atmel offers a software package called the Atmel AVR ISP which allows the programming of AVR Microcontrollers in circuit with a simple dongle which is attached to the Parallel Port. This dongle is detailed below. It can be built cheaply, making it an ideal starting point for developing with ATMEL AVR micros.

The current release of the software is Version 2.65 which can be downloaded at ftp://www.atmel.com/pub/atmel/avr_isp.zip (875KB)

If you are seeking a Serial/RS232 Version, have a look at Atmel's Application Note AVR910 which details a serial programmer using at AT90S1200.

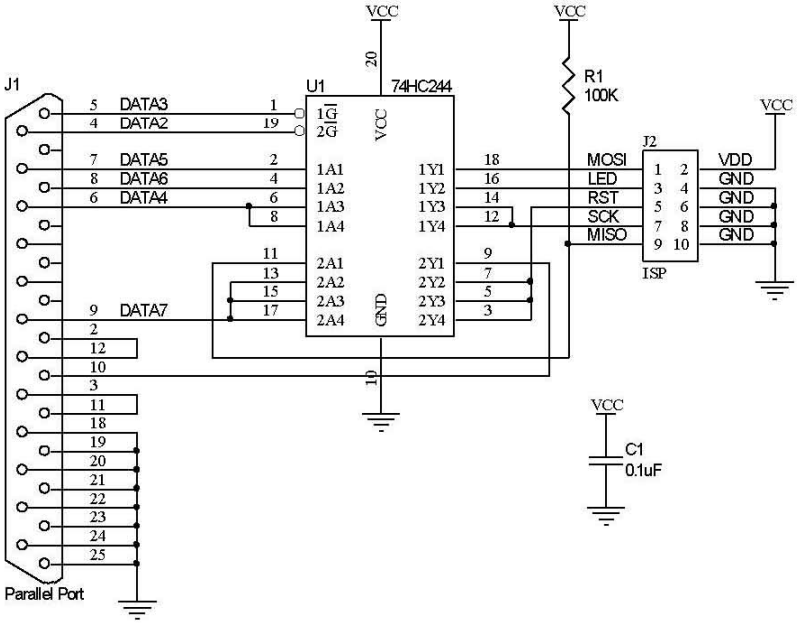


Figure 1 : Schematic for the ATMEL ISP Value Added Dongle

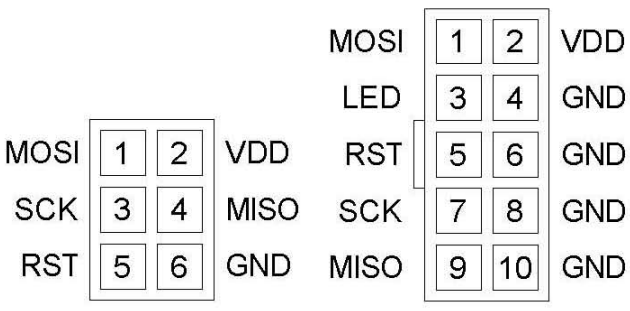
Using a 74HC244 Tri-State buffer as the main component, operation is extremely simple. The two loopback connections, pin 2 to 12 and 3 and 11 is used to identify the dongle. With both links in place the dongle is identified as a Value Added Pack Dongle. With only pins 2 and 12 links, it is reported as a STK300 or AVR ISP Dongle. With only 3 and 11 the dongle is reported as an STK200 or old Kanda ISP Dongle.

DATA2 and DATA3 of the Parallel Port Drive the TriState Outputs. A low will allow the passing of the serial clock and data during programming. MOSI, LED, SCK and Reset being outputs are buffered from the Parallel Port's DATA5, DATA6, DATA4 and DATA7 Respectively. The only input, MISO is fed into nACK, a status input of the Parallel Port.

There are two standard ISP Connectors for Atmel AVR Microcontroller ISP Programming. One standard is the 10 pin version using a DIL 5x5 header of 0.1" Pitch, shown in the above schematic. This is used on the ATMEL STK Kits. The other is a more compact 6 pin version, once again using a DIL 3x3 header of 0.1" pitch. This 6 pin version is the standard connector for ATMEL ISP Programmers.

The main advantage of the 10 pin header is the clean and easy use of 10 pin IDC crimp headers.

Name	Function	Description
MOSI	Master Out - Slave In	Data being transmitted to the part being programmed is sent on this pin
LED	Program LED	Optional Programming LED
RST	Target MCU Reset	Connects to Target AVR. Target AVR is programmed while in Reset State.
SCK	Shift Clock	Serial Clock Generated by the Programmer
MISO	Master In – Slave Out	Data received from the part being programmed is sent on this pin
VCC	ISP Power	Power Supply for the ISP. ISP Header must supply power to the dongle.
GND	Ground	Common Ground



6 Pin Header 10 Pin Header

Annex 2 ASCII Table

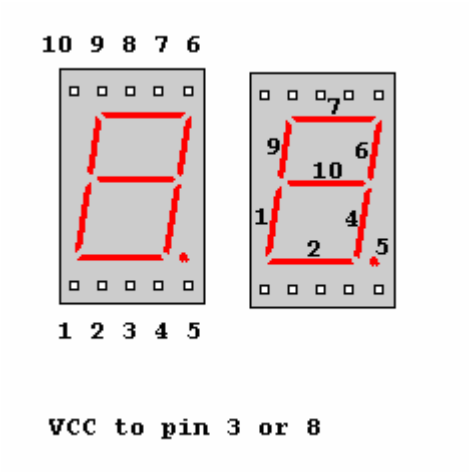
Decimal	Hex	Binary	Value
000	000	00000000	NUL (Null char.)
001	001	00000001	SOH (Start of Header)
002	002	00000010	STX (Start of Text)
003	003	00000011	ETX (End of Text)
004	004	00000100	EOT (End of Transmission)
005	005	00000101	ENQ (Enquiry)
006	006	00000110	ACK (Acknowledgment)
007	007	00000111	BEL (Bell)
008	008	00001000	BS (Backspace)
009	009	00001001	HT (Horizontal Tab)
010	00A	00001010	LF (Line Feed)
011	00B	00001011	VT (Vertical Tab)
012	00C	00001100	FF (Form Feed)
013	00D	00001101	CR (Carriage Return)
014	00E	00001110	SO (Shift Out)
015	00F	00001111	SI (Shift In)
016	010	00010000	DLE (Data Link Escape)
017	011	00010001	DC1 (XON) (Device Control 1)
018	012	00010010	DC2 (Device Control 2)
019	013	00010011	DC3 (XOFF)(Device Control 3)
020	014	00010100	DC4 (Device Control 4)
021	015	00010101	NAK (Negative Acknowledgement)
022	016	00010110	SYN (Synchronous Idle)
023	017	00010111	ETB (End of Trans. Block)
024	018	00011000	CAN (Cancel)
025	019	00011001	EM (End of Medium)
026	01A	00011010	SUB (Substitute)
027	01B	00011011	ESC (Escape)
028	01C	00011100	FS (File Separator)
029	01D	00011101	GS (Group Separator)
030	01E	00011110	RS (Request to Send)(Record Separator)
031	01F	00011111	US (Unit Separator)
032	020	00100000	SP (Space)
033	021	00100001	! (exclamation mark)
034	022	00100010	" (double quote)
035	023	00100011	# (number sign)
036	024	00100100	$ (dollar sign)
037	025	00100101	% (percent)
038	026	00100110	& (ampersand)
039	027	00100111	' (single quote)
040	028	00101000	((left/opening parenthesis)
041	029	00101001	) (right/closing parenthesis)
042	02A	00101010	* (asterisk)
043	02B	00101011	+ (plus)
044	02C	00101100	, (comma)
045	02D	00101101	- (minus or dash)
046	02E	00101110	. (dot)
047	02F	00101111	/ (forward slash)
048	030	00110000	0
049	031	00110001	1
050	032	00110010	2

Decimal	Hex	Binary	Value	
051	033	00110011	3	
052	034	00110100	4	
053	035	00110101	5	
054	036	00110110	6	
055	037	00110111	7	
056	038	00111000	8	
057	039	00111001	9	
058	03A	00111010	:	(colon)
059	03B	00111011	;	(semi-colon)
060	03C	00111100	<	(less than)
061	03D	00111101	=	(equal sign)
062	03E	00111110	>	(greater than)
063	03F	00111111	?	(question mark)
064	040	01000000	@	(AT symbol)
065	041	01000001	A	
066	042	01000010	B	
067	043	01000011	C	
068	044	01000100	D	
069	045	01000101	E	
070	046	01000110	F	
071	047	01000111	G	
072	048	01001000	H	
073	049	01001001	I	
074	04A	01001010	J	
075	04B	01001011	K	
076	04C	01001100	L	
077	04D	01001101	M	
078	04E	01001110	N	
079	04F	01001111	O	
080	050	01010000	P	
081	051	01010001	Q	
082	052	01010010	R	
083	053	01010011	S	
084	054	01010100	T	
085	055	01010101	U	
086	056	01010110	V	
087	057	01010111	W	
088	058	01011000	X	
089	059	01011001	Y	
090	05A	01011010	Z	
091	05B	01011011	[	(left/opening bracket)
092	05C	01011100	\	(back slash)
093	05D	01011101	]	(right/closing bracket)
094	05E	01011110	^	(caret/circumflex)
095	05F	01011111	_	(underscore)
096	060	01100000	`	
097	061	01100001	a	
098	062	01100010	b	
099	063	01100011	c	
100	064	01100100	d	
101	065	01100101	e	
102	066	01100110	f	
103	067	01100111	g	
104	068	01101000	h	

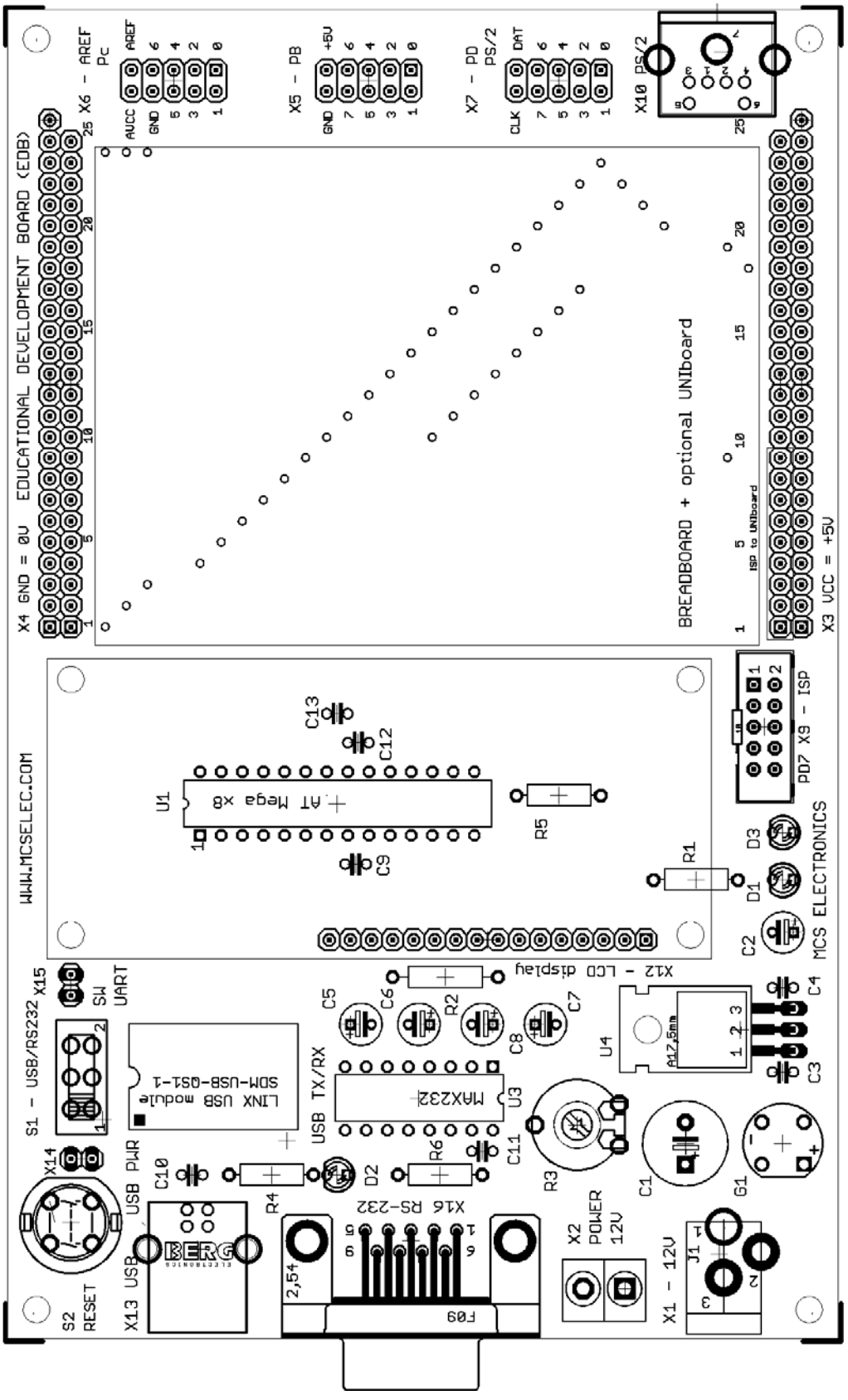
Decimal	Hex	Binary	Value	
105	069	01101001	i	
106	06A	01101010	j	
107	06B	01101011	k	
108	06C	01101100	l	
109	06D	01101101	m	
110	06E	01101110	n	
111	06F	01101111	o	
112	070	01110000	p	
113	071	01110001	q	
114	072	01110010	r	
115	073	01110011	s	
116	074	01110100	t	
117	075	01110101	u	
118	076	01110110	v	
119	077	01110111	w	
120	078	01111000	x	
121	079	01111001	y	
122	07A	01111010	z	
123	07B	01111011	{	(left/opening brace)
124	07C	01111100	|	(vertical bar)
125	07D	01111101	}	(right/closing brace)
126	07E	01111110	~	(tilde)
127	07F	01111111	DEL	(delete)

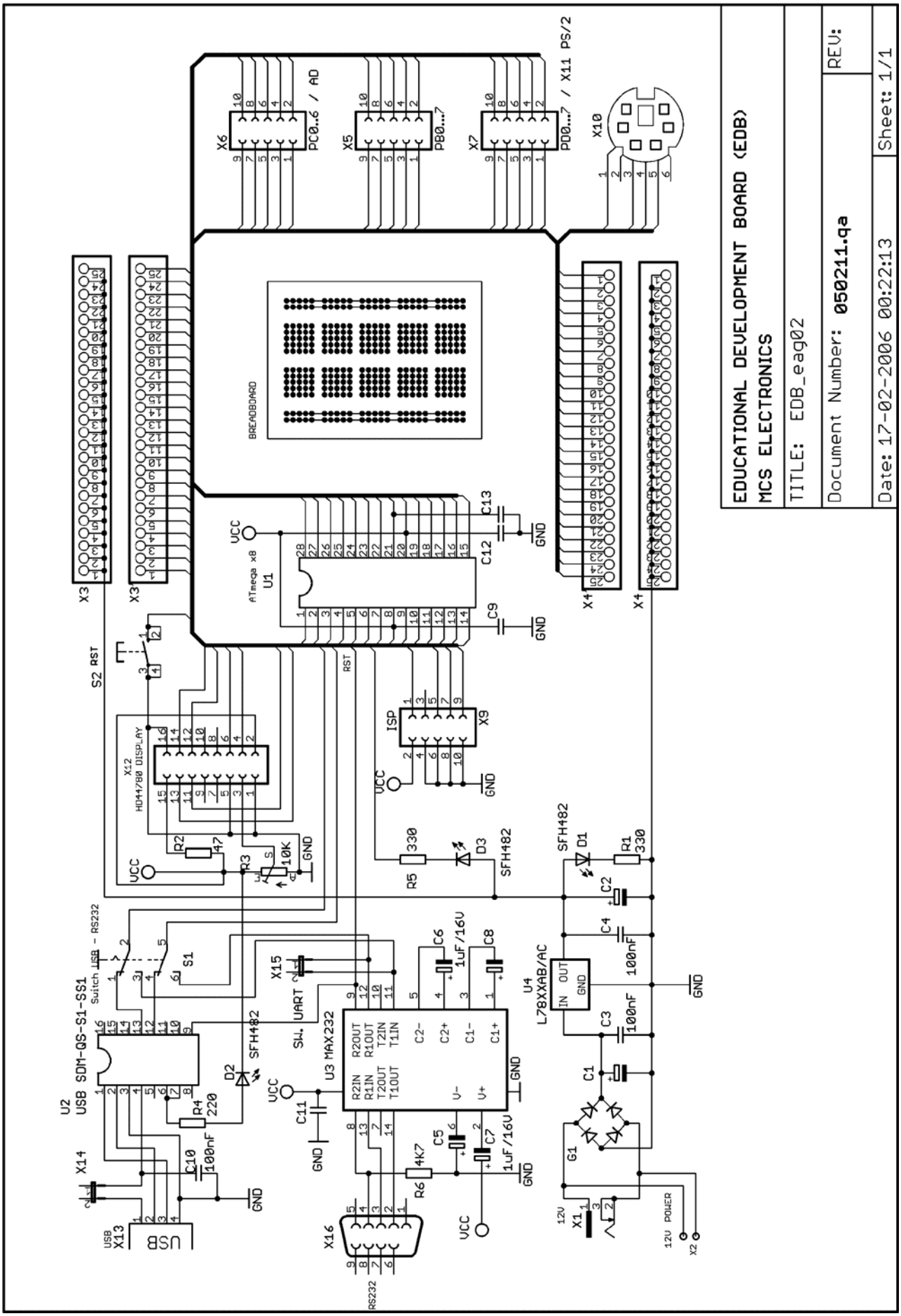
Annex 3 Anschlussbelegung 7-Segment-Anzeige

Pinbelegung für das SL119 oder OS516HWA 7-Segment-Display



Annex 4 EDB Schaltpläne




EDUCATIONAL DEVELOPMENT BOARD (EDB)
MCS ELECTRONICS
TITLE: EDB_eag02
Document Number: 050211.qa
REV:
Date: 17-02-2006 00:22:13
Sheet: 1/1
BREADBOARD
ATmega x8
U1
X6
PC0..6 / AD
X5
PB0...7
X7
PD0...7 / X11 PS/2
X10
X3
X4
C13
C12
C9
GND
VCC
S2 RST
RST
X12
HD44780 DISPLAY
ISP
X9
R2
47
R3
10K
R5
330
D3
SFH482
D1
R1
330
C2
C4
100nF
C3
100nF
U4
L78XXAB/AC
IN
OUT
GND
G1
C1
12V
X1
12V POWER
X2
U2
USB SDM-QS-S1-SS1
Switch USB - RS232
S1
X15
SW. UART
U3 MAX232
R2IN
R1IN
T2OUT
T1OUT
R2OUT
R1OUT
T2IN
T1IN
C2-
C2+
C1-
C1+
V-
V+
C6
1uF/16V
C8
C5
C7
1uF/16V
C11
D2
SFH482
R4
220
C10
100nF
X14
USB
X13
X16
RS232
R6
4K7

Copyright Hinweise

Der Inhalt dieses Dokumentes betrifft nationale und internationale Copyrightgesetze auf denen das Copyright von MCS Electronics © 2005 Basiert, mit Ausnahme von:

Paragraph 3.1.3a “ATMega88 Datasheet” (Atmel corporation),
Paragraph 3.1.5 “Current Transfer Ratio” (Sharp Semiconductor),
Paragraph 3.4.1 “AN AVR313” (Atmel corporation),
Paragraph 3.4.3 “TSOP Interface” (Vishay Semiconductor),
Annex 1 “AVR ISP Dongle” (Beyondlogic.org).

MCS Electronics erteilt hiermit die Erlaubnis, dieses Dokument für persönliche/schulische oder nicht-kommerzielle Zwecke zu verwenden. In allen anderen Fällen darf dieses Dokument nicht reproduziert, kopiert, gespeichert oder verändert werden sowie nicht als Ganzes oder Teil als Bestandteil eines abgeleiteten Werkes verwendet werden, ohne die schriftliche Erlaubnis von MCS Electronics.

Ausdrücke und Produktnamen in diesem Dokument können Warenzeichen Dritter sein.

Hinweise zur Gewährleistung und Zuverlässigkeit

Die Informationen in diesem Dokument werden ohne jegliche ausdrückliche oder implizite Gewährleistung gegeben. Die betrifft ebenso Handelsangelegenheiten, Titel, Rechte an Geistigem Eigentum oder ausdrückliche Eignung für jeglichen speziellen Zweck sowie Richtigkeit und Volltändigkeit.

Sie erklären sich damit einverstanden, dass in keinem Fall MCS Electronics für irgendwelche Schäden verantwortlich ist, die durch den Gebrauch dieses Dokumentes verursacht werden könnten. Dies betrifft insbesondere jegliche Schäden, Einnahmeverluste und Produktionsunterbrechuungen,,auch wenn MCS Electronics auf deren Möglichkeit hingeiwesen wurde.
Die Verwendung dieser Dokumentation und der EDB-Hardware, einschliesslich der Leiterplatte und deren Komponenten, geschieht ausdrücklich auf eigenes Risiko.

MCS Electronics garantiert nicht die Fehlerfreiheit und Vollständigkeit der Informationen, Texte, Grafiken, Links oder anderer Dinge in diesem Dokument. MCS Electronics kann diese Informationen jederzeit ohne Vorankündigung ändern. MCS Electronics unterliegt keiner Verpflichtung
solche Informationen auf dem neuesten Stand zu halten.

Aus diesem Dokument könen keinerlei Rechte abgeleitet werden