Hanging Man

Version 1.0

Geocaching Spiel
Hanging Man bzw. Galgenmännle

Lonsee im Juli 2014

Markus Fulde
Finkenweg 3
D-89173 Lonsee
Telefon +49 (7336) 92 11 89
Fax +49 (7336) 92 10 68
Mobil +49 (160) 84 54 314

Email Markus.Fulde@t-online.de

1 Inhaltsverzeichnis

2 Abbildungsverzeichnis

3 Tabellenverzeichnis

4 Schaltbilderverzeichnis

5 Softwareverzeichnis

6 Historie

Datum	*Entscheidung*
01.05.2014	Beginn der Projektarbeit und Dokumenterstellung
09.06.2014	Um auch auf die Tastereingabe via Interrupt reagieren zu können wir die Controllerbelegung für DDS2 PD3 INT1 und PD4 getauscht.
09.07.2014	Fertigstellung der HD, Beauftragung der PCB's bei Leiton, Fertigstellung der Projektdokumentation und der Projektsoftware
29.07.2014	Fertigstellung der ersten kompletten Dokumentversion zum Projekt Hanging Man

Tabelle 1: Historie

7 Allgemeines

7.1 Die Entwicklungsumgebung

Die Entwicklungsumgebung des Projekts Hanging Man:

Abbildung 1: Entwicklungsumgebung Hanging Man

7.2 Allgemeine Beschreibung und Idee zur Realisierung

Die Wikipedia schreibt zum Hanging Man bzw. Galgemännchen folgendes:

Galgenmännchen, Galgenraten, Galgenbaum oder auch Galgenmann, Hängemann, Hängemännchen oder einfach auch Galgen (englisch auch hangman) ist ein einfaches Buchstabenspiel.

Geschichte

Tony Augarde, Autor des Werkes The Oxford Guide to Word Games (Oxford University Press) sagt im Kapitel über dieses Spiel: „Die Ursprünge von Galgenmann liegen im Dunkeln, es scheint jedoch in Viktorianischer Zeit entstanden zu sein." 1894 ist es in Alice Bertha Gommes Buch Traditional Games unter dem Namen Vögel, Tiere und Fische (engl.:Birds, Beasts and Fishes) erwähnt. Die Regeln waren den heutigen Versionen ähnlich.

Spielverlauf

… ein t bitte …" „… ja, an achter und elfter Stelle ist ein T …"

Benötigt werden dazu Papier und Stift. Die Anzahl der Mitspieler ist variabel, häufig sind es jedoch nur zwei. Der Beginner überlegt sich nun ein längeres Wort, von dem er jedoch lediglich den Anfangsbuchstaben hinschreibt. Alle weiteren Buchstaben des ausgedachten Wortes werden durch Striche markiert. Der Rate-Spieler nennt nun in beliebiger Reihenfolge nacheinander einzelne Buchstaben des Alphabets. Der Gegner muss nun jeweils ansagen, wie oft und an welcher Stelle des Lösungswortes der Buchstabe vorkommt. So ergibt sich nach und nach das gesuchte Wort. Kommt ein genannter Buchstabe darin jedoch nicht vor oder hat der Löser gar das falsche Wort geraten, so beginnt der erste Spieler damit, einen Galgen mit einem Gehängten zu zeichnen. Dies geschieht in mehreren Etappen (bei jeder Fehlfrage kommt ein Teilstrich dazu), so dass der Rätsellöser je nach gespieltem Schwierigkeitsgrad etwa 10 bis 15 Fehlversuche hat. Hat er dann das Wort noch nicht herausgefunden, so hat er verloren und hängt symbolisch am Galgen.

Varianten

Es existieren zahlreiche Varianten dieses Spiels, so dass es ratsam erscheint, sich vor Spielbeginn darüber zu verständigen. So gibt es zum Beispiel:

- Umlaute werden ae, oe oder ue geschrieben.
- die Anzahl der Fragen wird beschränkt (der Galgen ist schneller fertig).
- Bei zwei Mitspielern schreiben beide gleichzeitig ein Wort auf und die Zettel werden anschließend getauscht und es wird abwechselnd geraten.
- Bei mehreren Mitspielern kommt abwechselnd jeder mit dem Raten dran und gewonnen hat derjenige, der das Wort als erster benennt.
- Anstatt des Galgens zeichnet man andere Gegenstände.
- Der Anfangsbuchstabe muss ebenfalls geraten werden.

Lerneffekte

Das Spiel ist gerade bei Pädagogen, die in der Schule tätig sind, sehr verbreitet. Sie spielen oder lassen es spielen, um in der Grundschule die deutsche Rechtschreibung zu festigen und später das fremdsprachige Vokabular zu üben. Auch in anderen Fächern ist es zum Erlernen von Fachbegriffen (Länder, Hauptstädte etc.) nützlich. Pädagogische Kritik gab und gibt es an der Darstellung eines Galgens. Deshalb sind manche dazu übergegangen, anstatt des martialisch anmutenden mittelalterlichen Hinrichtungsinstruments einfach ein Tier (kleines Schweinchen, Elefant etc.) oder andere harmlose Dinge, wie zum Beispiel eine Blume, zu zeichnen.

In diesem Geocaching-Projekt Hanging Man wird das Spielkonzept dazu benutzt, eine kleine Elektronik zu entwickeln welche im Gelände versteckt wird, die nach jedem erfolgreichem Spiel die finalen Geokoordinaten der finalen Dose anzeigt.

Hierbei wird immer per Zufallszahlengenerator aus einem Satz vordefinierter Worte ausgewählt.

7.3 Leistungsumfang

im Folgenden wird der Leistungsumfang und die Teilfunktionalität beschrieben welche Hanging Man besitzt und dem Benutzer zur Verfügung stellt:

- Hauptschalter
- LCD-Display
  - Bedienung des Spiels
  - Statusmeldungen
- Drück-Dreh-Schalter
  - Steuerung des Spiels
  - Auswahl und Bestätigung der zu suchenden Buchstaben
- Schnittstellen
  - RS232 Schnittstelle zum Tracing und zur Datenübertragung der Logging-Daten an PC
  - ISP Schnittstelle zur direkten Programmierung des Target
- Akustische Signalisierungen über Piezo-Schallgeber

7.4 Funktionskomponenten

Das Projekt Hanging Man verfügt über die folgenden einzelnen Funktions- / Teilkomponenten:

- Spannungsversorgung via 9V Batterie und Erzeugung von 5V über Festspannungsregler SUP
- Mikrocontroller ATmega8L (inkl. ISP, RS232 und Reset) µC
- LCD-Display 2x16 EA DOG-M 162 von Electronic Assembly LCD
  - Die Helligkeit des Display soll über PWM einstellbar sein
  - Bei Innaktivität soll das Display abgeschaltet werden können
- Hauptschalter für Spannungsversorgung SW
- Drück-Dreh-Geber (inkrementeller Drehgeber mit Tasterfunktion) DDS
- LED-Anzeigen für Betriebsanzeige und Low-Power Batterie-Überwachung LED
  - Low-Current LED Grün zur Anzeige des Betriebs (Allive LED)
  - Low-Current LED Rot zur Überwachung der Batteriespannung
- Gehäuse mit Batteriefach und Displayfenster BOX
- Akustisches Feedback über Piezo Schallwandler SOUND

8 Elektronische Grundlagen

8.1 Mikrokontroller ATmega8L

Im Projekt wird der Mikrokontroller ATmega8L von ATMEL mit externem Takt von 8MHz eingesetzt. Hierzu wird ein externer Quarz beschaltet.

Erste Inbetriebnahmen und Versuche bzgl. Projektumsetzung werden mit dem ATmega8L auf dem STK500 von ATMEL realisiert.

PINOut ATmega8L:

PINOut ATmega8LPDIP

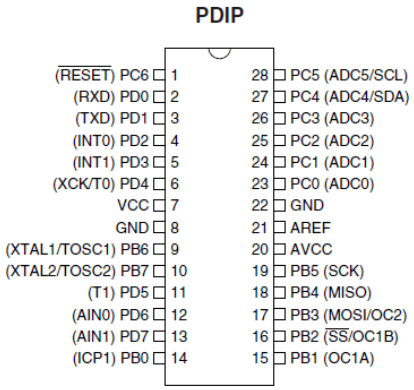


Abbildung 2: PINOut ATmega8L PDIP

PINOut ATmega8LTQFP

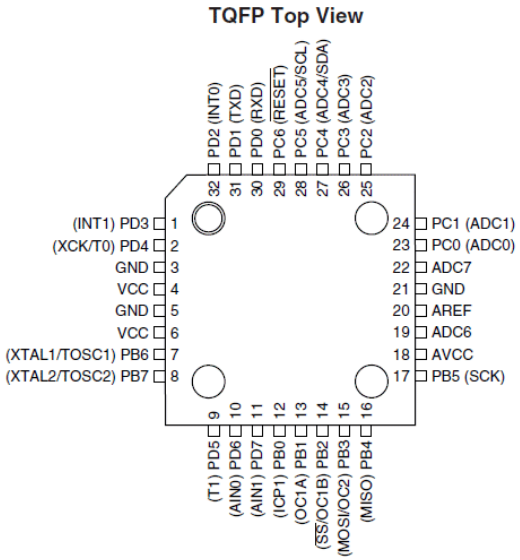


Abbildung 3: PINOut ATmega8L TQFP

Bauteile:

Stückliste: CPU-Unit ATmega8L			
Sonstiges		*Halbleiter*	
G1	Quarz 8,00 MHz	IC1	ATMEL ATmega8L RISC CPU

Tabelle 2: Stückliste CPU-Unit ATmega8L

8.2 Ressourcenzuordnung ATmega8L im Projekt Hanging Man

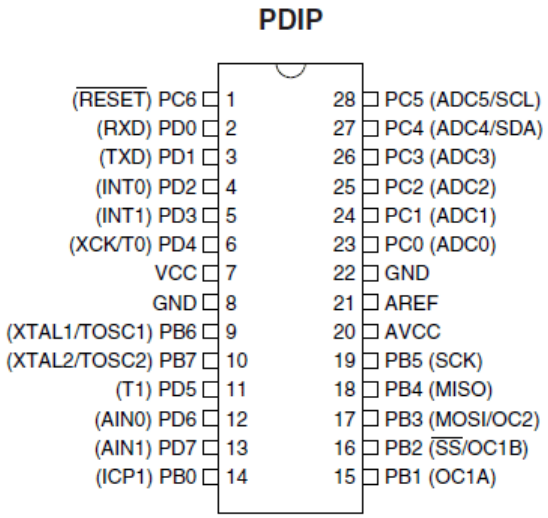


Abbildung 4: PINOut ATmega8L PDIP

PIN	*Port*	*Funktion*	*Used*	*Beschreibung*	*Definition*
1	PC6	RESET	J	Externer Reset Eingang	RESET
2	PD0	RXD	J	RS232 Schnittstelle – Receive Data	RXD
3	PD1	TXD	J	RS232 Schnittstelle – Transmit Data	TXD
4	PD2	INT0 / PD2	J	Impulsdrehgeber / Rchtungserkennung	DDS1
5	PD3	INT1 / PD3	J	Taster	SELECT
6	PD4	PD4	J	Impulsdrehgeber / Richtungserkennung	DDS2
7	VCC	VCC	J	Spannungsversorgung +5V	VCC
8	GND	GND	J	Ground GND	GND
9	PB6	XTAL1	J	Externer 16 MHz Quarz	XTAL1
10	PB7	XTAL2	J	Externer 16 MHz Quarz	XTAL2
11	PD5	n.c.	N	n.c.	n.c.
12	PD6	PD6	J	LCD Datenbus E	LCD_E
13	PD7	PD7	J	LCD Datenbus RS	LCD_RS
14	PB0	PB0	J	Alive-LED	ALIVE
15	PB1	PB1	J	Spannungsüberwachung-LED	PWRLED
16	PB2	PB2	J	Soundausgabe über Piezo-Summer	SOUND
17	PB3	MOSI OC2	J	ISP Programmierinterface und PWM für Displayhelligkeit	MOSI_PWM
18	PB4	MISO	J	ISP Programmierinterface	MISO
19	PB5	SCK	J	ISP Programmierinterface	SCK
20	AVCC	AVCC	J	Versorgungsspannung ADC	AVCC
21	AREF	AREF	J	Externe Referenzspannung	AREF
22	GND	AGND	J	Ground GND für ADC	AGND
23	PC0	ADC0	J	Analogeingang Spannungsüberwachung	PWR_CHECK
24	PC1	n.c.	N	n.c.	n.c.
25	PC2	PC2	J	LCD Datenbus Bit 0	LCD_D0
26	PC3	PC3	J	LCD Datenbus Bit 1	LCD_D1
27	PC4	PC4	J	LCD Datenbus Bit 2	LCD_D2
28	PC5	PC5	J	LCD Datenbus Bit 3	LCD_D3

Tabelle 3: Ressourcenzuordnung ATmega8L

	Externe Betaktung und Reset
	Spannungsversorgungen GND und VCC
	Serielle Schnittstelle RS232
	ISP Programmierinterface
	Zu den Analog-Digital-Wandlern gehörend
	Zu Timer2 gehörende Pin's
	GPIO Verwendung

Die unter „Definition" vergebenen Namen definieren die Namensgebung der Signalleitungen im Schaltplan und ggf. in der Software.

8.3 Interrupt-Vektor-Tabelle ATmega8L

Reset and Interrupt Vectors

Vector No.	Program Address(2)	Source	Interrupt Definition
1	0x000(1)	RESET	External Pin, Power-on Reset, Brown-out Reset, and Watchdog Reset
2	0x001	INT0	External Interrupt Request 0
3	0x002	INT1	External Interrupt Request 1
4	0x003	TIMER2 COMP	Timer/Counter2 Compare Match
5	0x004	TIMER2 OVF	Timer/Counter2 Overflow
6	0x005	TIMER1 CAPT	Timer/Counter1 Capture Event
7	0x006	TIMER1 COMPA	Timer/Counter1 Compare Match A
8	0x007	TIMER1 COMPB	Timer/Counter1 Compare Match B
9	0x008	TIMER1 OVF	Timer/Counter1 Overflow
10	0x009	TIMER0 OVF	Timer/Counter0 Overflow
11	0x00A	SPI, STC	Serial Transfer Complete
12	0x00B	USART, RXC	USART, Rx Complete
13	0x00C	USART, UDRE	USART Data Register Empty
14	0x00D	USART, TXC	USART, Tx Complete
15	0x00E	ADC	ADC Conversion Complete
16	0x00F	EE_RDY	EEPROM Ready
17	0x010	ANA_COMP	Analog Comparator
18	0x011	TWI	Two-wire Serial Interface
19	0x012	SPM_RDY	Store Program Memory Ready

Notes: 1. When the BOOTRST Fuse is programmed, the device will jump to the Boot Loader address at reset, see "Boot Loader Support – Read-While-Write Self-Programming" on page 209.
2. When the IVSEL bit in GICR is set, Interrupt Vectors will be moved to the start of the boot Flash section. The address of each Interrupt Vector will then be the address in this table added to the start address of the boot Flash section.

Table 19 shows reset and Interrupt Vectors placement for the various combinations of BOOTRST and IVSEL settings. If the program never enables an interrupt source, the Interrupt Vectors are not used, and regular program code can be placed at these locations. This is also the case if the Reset Vector is in the Application section while the Interrupt Vectors are in the boot section or vice versa.

Tabelle 4: Interrupt-Vektor-Tabelle ATmega8L

Reset and Interrupt Vectors Placement

BOOTRST[1]	IVSEL	Reset Address	Interrupt Vectors Start Address
1	0	0x000	0x001
1	1	0x000	Boot Reset Address + 0x001
0	0	Boot Reset Address	0x001
0	1	Boot Reset Address	Boot Reset Address + 0x001

Note: 1. The Boot Reset Address is shown in Table 82 on page 220. For the BOOTRST Fuse "1" means unprogrammed while "0" means programmed.

Tabelle 5: Interrupt-Vektor-Tabelle ATmega8L

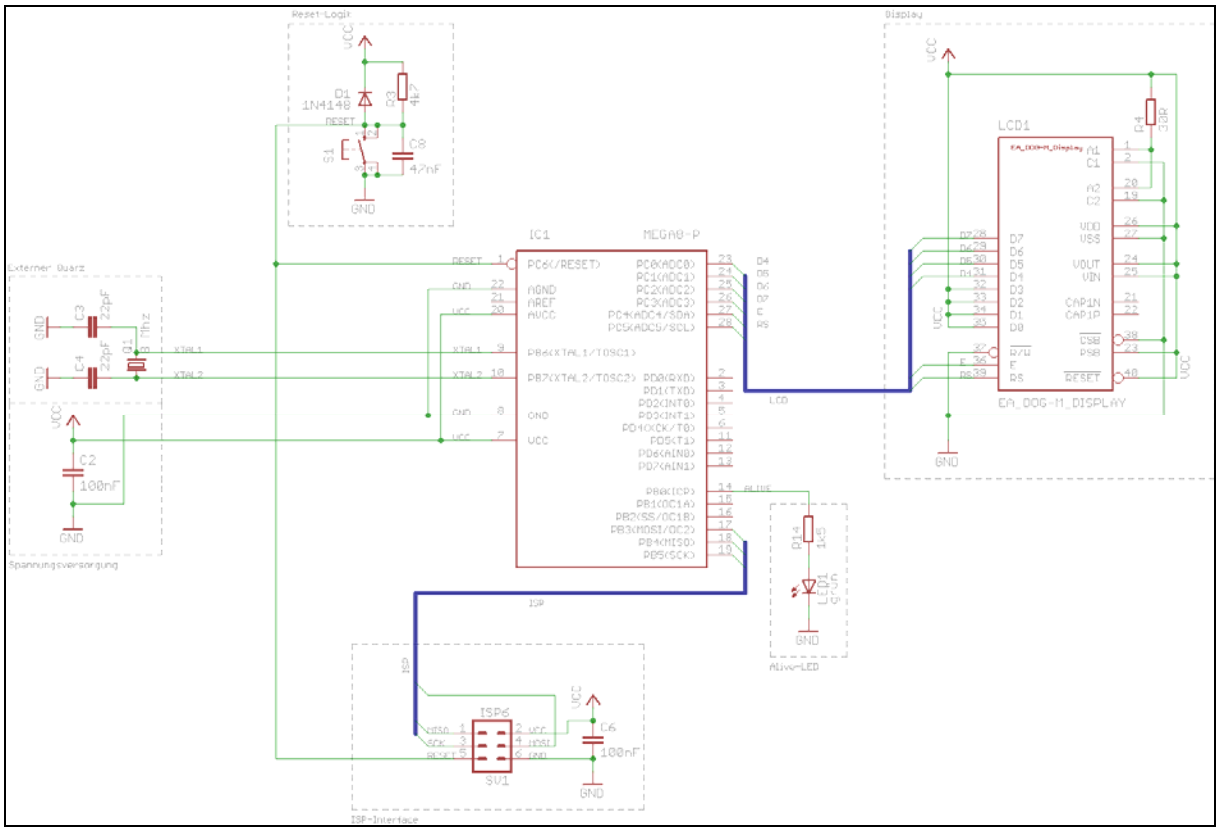
Zur Erfassung und Auswertung des Drehgebersignals wird de externe Interrupt 0 verwendet.

8.4 Basisbeschaltung eines ATmega8L inkl. Display

Die im Folgenden dargestellte Schaltung bildet die Basisbeschaltung eines AVR ATmega. Zum Einsatz kommt ein externer Quarz mit 8.000.000 Hz (8 MHz). Weiter ist im Schaltplan der Anschluss eines externen RESET-Tasters vorgesehen.

Die wesentlichen Bestandteile sind vorhanden, welche wären:

- Reset-Logik
- ISP-Interface
- Spannungsversorgung
- Geräuschreduktion für ADC
- Externer Quarz
- Display



Schaltbild 1: Basisbeschaltung ATmega 8L inklusive Display

Bauteile:

Stückliste: Basisbeschaltung ATmega8L			
Widerstände		***Halbleiter***	
R3	Metallschichtwiderstand 4k7 Ω	IC1	ATMEL AVR ATmega8L-P
R4	Metallschichtwiderstand 40 Ω	D1	Diode 1N4148
R14	Metallschichtwiderstand 1k5 Ω	LCD1	LCD-Display EA DOGM162
		LED1	Low-Current LED, grün
Kondensatoren		***Sonstiges***	
C3, C4	Keramikkondensator 22 pF	S1	Kurzhubtaster
C8	Keramikkondensator 47 nF	Q1	Quarz 8.000.000 Hz
C2, C6	MP-Kondensatoren 100nF	SV1	Federleiste MA03-2 (ISP)

Tabelle 6: Stückliste Basisbeschaltung ATmega8L

Reset-Taster:

Als Reset-Taster wir ein Print-Kurzhubtaster verwendet welcher auf der Display- und Tastatur-PCB für die Frontblende untergebracht ist. Der Reset-Taster ist über eine Bohrung im Frontdeckel mit einem dünnen Gegenstand (z.B. Büroklammer) erreichbar und zu betätigen.

Kurzhubtaster 6x6mm, Höhe: 4,3mm, 12V, vertikal
Bestellnummer Reichelt elektronik: *TASTER 3301*

Tabelle 7: Kurzhubtaster für RESET

8.5 ISP-Schnittstelle / ISP-Programmierung

Zur In-System-Programmierung wird die ATMEL ISP-Schnittstelle umgesetzt. Die Programmierung im Projekt erfolgt direkt über BASCOM mit Hilfe von STK500 oder der USB-Programmieradapter AVRISPMKII von ATMEL.

Grundsätzlich ist der Mikrocontroller mit jedem Atmel – ISP – Kompatiblen Programmiergerät programmierbar.

ISP Programmierinterface von ATMEL:

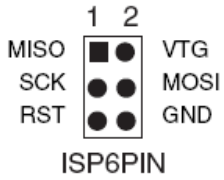
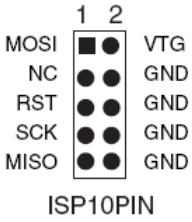
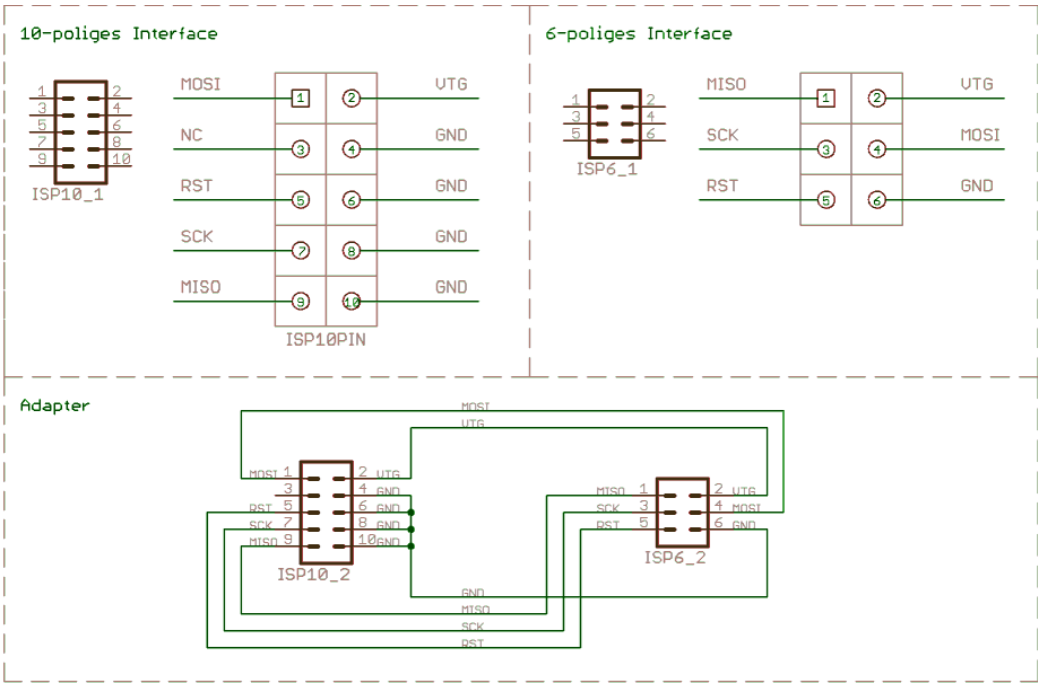
	
6-pin ISP connection Pinout	10-pin ISP connection Pinout

Tabelle 8: ISP connection Pinout

Für das Projekt Hanging Man wird das 6-polige Interface in der folgenden Form umgesetzt:



Schaltbild 2: 10-Pin ISP connection Pinout

Der Adapter und das 10-polige Interface werden nicht umgesetzt. Das 6-polige ISP-Interface befindet sich auf allen Boards und kann direkt mit dem AVRISP mkII verbunden werden.

8.6 RS232-Traceschnittstelle

Abkürzung	*Name*	*Beschreibung*	*Pin-Nr. 25-pol.*	*Pin-Nr. 9-pol.*	*In/Out*
--	Common Ground	Gemeinsame Abschirmmasse (nicht Datenmasse)	Pin 1	—	—
TxD,TX,TD	Transmit Data	Leitung für ausgehende (gesendete) Daten.	Pin 2	Pin 3	Out
RxD,RX,RD	Receive Data	Leitung für den Empfang von Daten.	Pin 3	Pin 2	In
RTS	Request to Send	„Sendeanforderung"; Eine logische Null an diesem Ausgang signalisiert der Gegenstelle, dass sie Daten Senden kann	Pin 4	Pin 7	Out
CTS	Clear to Send	Eine logische Null an diesem Eingang ist ein Signal der Gegenstelle, dass sie Daten entgegennehmen kann	Pin 5	Pin 8	In
DSR	Dataset Ready	Ein angeschlossenes Gerät signalisiert dem Computer, dass es einsatzbereit (nicht notwendigerweise empfangsbereit) ist, wenn eine logische Null auf dieser Leitung anliegt.	Pin 6	Pin 6	In
GND	Ground	Signalmasse. Die Signalspannungen werden gegen diese Leitung gemessen.	Pin 7	Pin 5	—
DCD,CD	(Data) Carrier Detect	Ein Gerät signalisiert dem Computer, dass es einlaufende Daten auf der Leitung erkennt	Pin 8	Pin 1	In
DTR	Data Terminal Ready	Über diese Leitung signalisiert der PC dem Gerät, dass er betriebsbereit ist. Damit kann ein Gerät eingeschaltet oder zurückgesetzt werden. (Üblicherweise schaltet ein Gerät z.B. Modem diese Leitung auf DSR durch, wenn es einsatzbereit ist)	Pin 20	Pin 4	Out
RI	Ring Indicator	Das Gerät zeigt dem PC an, dass ein Anruf ankommt ("ring" ist engl. für "klingeln"; besonders bei Modems)	Pin 22	Pin 9	In

Tabelle 9: PIN-Belegung des seriellen RS232-Ports

In/Out wird auch Sicht vom PC aus gesehen.

RS232 Buchse 9-polig:

RS232 Stecker 9-polig:

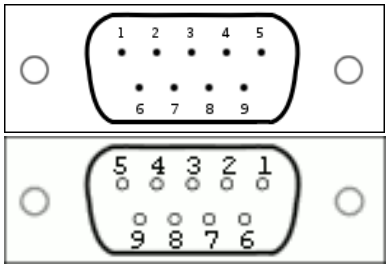
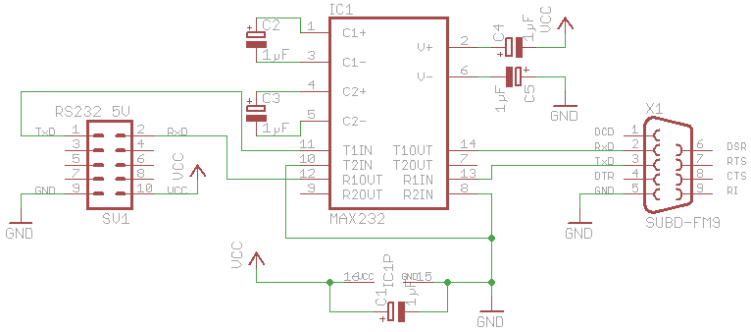


Tabelle 10: PIN-Belegung der 9-poligen RS232 Stecker/Buchse

8.6.1 Adapter RS232 PIN-Header / D-SUB9



Schaltbild 3: Adapter zwischen PIN-Header und D-SUB9 Buchse

Bauteile:

Stückliste: RS232-Adapter zwischen PIN-Header und D-SUB9 Buchse PC			
Sonstiges		***Halbleiter***	
PIN-HEADER	PFL10 Pfostensteckverbinder	IC1	Maxim MAX232 CPE
D-SUB_9POLIG	SUB-D-Buchse 9-polig.		
Kondensatoren			
C1, C2, C3, C4, C5	Elko 1µF/16V		

Tabelle 11: Stückliste RS232-Adapter zwischen PIN-Header und D-SUB9 Buchse PC

PIN-Belegung Pfostenstecker:

PIN	*Funktion*
1	TxD
2	Rxd
3	n.c.
4	n.c.
5	n.c.
6	n.c.
7	n.c.
8	n.c.
9	GND
10	Spannungsversorgung Target (Vcc)

Tabelle 12: PIN-Belegung RS232-Pfostensteckers

Ressourcenzuordnung zum ATmega8L:

Nummer	*Schaltbild*	*Ressource ATmega8L*
1	RXD	PortD.0 [PD0] (RXD)
2	TXD	PortD.1 [PD1] (TXD)

Tabelle 13: Ressourcenzuordnung SW-RS232 für den ATmega8L

8.6.2 Trace-Adapter RS232 Prototyp auf Lochraster

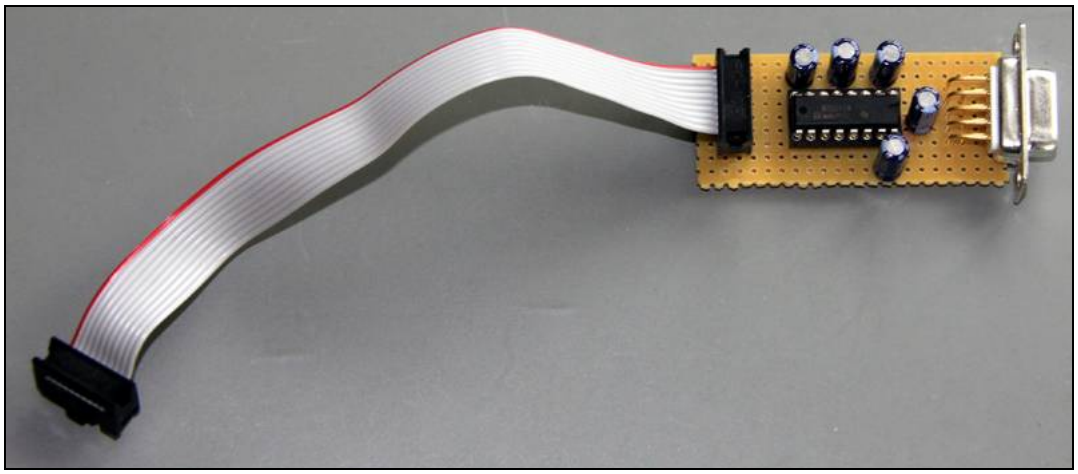


Abbildung 5: RS232-Traceadapter

8.7 AVR Fusebits Tutorial

Einleitung - Was sind Fusebits?

Fusebits sind im Grunde genommen nichts anderes als Speicherzellen die man Programmieren und Löschen kann. Sie dienen jedoch nicht zur Speicherung von Daten sondern mit ihrer Hilfe kann das Verhalten des AVR beeinflusst werden. Zum Beispiel können bestimmte Funktionen aktiviert und deaktiviert werden.

Was ist zu beachten?

Bevor man zum ersten mal die Fusebits eines AVR verändert, sollte man wissen das man den Controller damit nicht zerstören kann. Es ist jedoch möglich den Controller so einzustellen das man mit den normalen Werkzeugen nicht mehr darauf zugreifen kann. Grundsätzlich kann ein "verfuseter" Controller mit dem richtigen Werkzeug aber wieder repariert werden. Ob dies wirtschaftlich jedoch sinnvoll ist steht auf einem anderen Blatt.

Was anfänglich Probleme verursacht ist die "invertierte Logik" der Fusebits. Ein programmiertes Fusebit ist nicht wie man annehmen könnte auf "1" gesetzt sondern auf "0". Eine unprogrammierte Fuse ist "1". Manche Programme stellen programmierte Fusebits mit einem gesetzten Häkchen dar (z.B. PonyProg), welches dann als eine 0 (=programmed) interpretiert wird. Möchte man das Fusebit "setzen" oder "programmieren" muss man das Häkchen setzen.

Handelt es sich um ein Enable Fusebit bedeutet ein programmiertes Fusebit (0), das die Funktion eingeschaltet ist. Ist es ein Disable Fusebit bedeutet ein programmiertes Fusebit (0), das die Funkiton ausgeschaltet ist.

Was braucht man?

Die Fusebits lassen sich über ISP, JTAG oder parallel programmieren. Man kann hierfür die gleiche Hardware und Software verwenden wie zum Programmieren der des Flash Speichers oder des Eeproms, zum Beispiel PonyProg oder das AVR Studio.

Die Fusebytes

Je nach Controller stehen bis zu drei Fusebytes zur Verfügung: Fuse Low Byte, Fuse High Byte und Extended Fuse Byte. Einige der älteren Controller haben nur ein Fuse Byte und sehr wenige Fusebits.

Die Fusebits

Die folgende Beschreibung listet alle Fusebits auf die es bei den AVR Controllern gibt. Es gibt jedoch keinen Controller in dem alle Fusebits gleichzeitig zu finden sind. Ältere Controller vom Typ AT90S kennen teilweise nur 2 Fusebits.

CKSEL

Die wohl am häufigsten geänderten Fusebits sind CKSEL0 bis CKSEL3 (Select Clock Source). Mit ihrer Hilfe wählt man die Taktquelle aus der der Controller seinen Takt erhält. Hier ist etwas Vorsicht geboten da eine falsche Einstellung den Controller lähmen kann. Eine falsche Einstellung lässt sich jedoch relativ leicht beheben. Die genauen Parameter können zwischen den einzelnen Typen variieren
Default: Interner RC Oszillator mit 1MHz (bzw 8MHz bei Typen mit Vorteiler)

```
CKSEL0 : 0 (programmiert)
CKSEL1 : 1 (unprogrammiert)
CKSEL2 : 1 (unprogrammiert)
CKSEL3 : 1 (unprogrammiert)
```

SUT

Mit SUT0 und SUT1 lässt sich die Zeit einstellen wie lange der Reset Impuls nach einem Reset oder Power Up verzögert wird. Je nach Umgebungsbedinung kann die Reset Zeit verlängert oder verkürzt werden. Zusammen mit der Brown Out Detection wird eine externe Resetschaltung (bis auf den üblichen 10kOhm PullUp Widerstand) meist überflüssig.

Default:

```
SUT0 : 0 (programmiert)
SUT1 : 1 (unprogrammiert)
```

CKDIV8

Divide Clock by 8 ist etwas irreführend. Wenn dieses Fusebit gesetzt ist wird ein Vorteiler aktiviert, der den Takt für den Controller durch 8 teilt. Es ist jedoch möglich diesen Vorteiler auf einen anderen Wert einzustellen. Dies ist dann sinnvoll wenn der Controller aus einer externen Taktquelle gespeist werden soll, die Frequenz aber zu hoch ist. Details dazu im Artikel Taktquelle.

Default:

```
CKDIV8 : 0 (programmiert)
```

CKOUT

Wird diese Fuse programmiert wird der CPU Takt an dem entsprechenden CLKO Pin ausgegeben.

Default:

```
CKOUT : 1 (unprogrammiert)
```

CKOPT

CKOPT kommt zum Einsatz wenn der AVR von einem externen Quarz getaktet wird. Wird CKOPT programmiert (0) schwingt der Oszillator mit der maximalen Amplitude. Dies kann notwendig werden wenn der AVR in einer Umgebung mit vielen Störsignalen betrieben werden soll. Ist CKOPT unprogrammiert (1) schwingt der Oszillator mit einer niedrigeren Amplitude. Dadurch verringert sich die Stromaufnahme und die Störabstrahlung.

Default:

```
CKOPT : 1 (unprogrammiert)
```

RSTDISBL

Dieses Fuse Bit steuert die Funktion des Reset Pin. Wird es programmiert kann man den Reset Pin als normalen IO Pin verwenden.

Achtung: Wird dieses Bit programmiert kann der Controller nicht mehr über die ISP Schnittstelle erreicht werden

Default:

```
RSTDISBL : 1 (unprogrammiert)
```

SPIEN

Mit SPIEN kann die ISP Schnittstelle abgeschaltet werden. Dieses Fusebit lässt sich nur über die parallele Programmierung ändern. Ist die ISP Schnittstelle einmal abgeschaltet kann der Controller nicht mehr über ISP erreicht werden.

Default:

```
SPIEN : 0 (programmiert)
```

JTAGEN

JTAGEN aktiviert/deaktiviert die JTAG Schnittstelle.

Default:

JTAGEN : 0 (programmiert)

DWEN

DWEN aktiviert/deaktiviert die debugWire Schnittstelle.

Default:

```
DWEN : 1 (unprogrammiert)
```

OCDEN

OCDEN aktiviert/deaktiviert das On-Chip Debug System. Das On-Chip Debug System kann unabhängig von der JTAG Schnittstelle deaktiviert werden. Bei abgeschaltetem OCD kann der Controller über JTAG nur programmiert werden.

Default:

```
OCDEN : 1 (unprogrammiert)
```

EESAVE

Wird EESAVE programmiert wird das EEprom bei einem Chip Erase vor dem Löschen geschützt. Ein Chip Erase löscht normalerweise den kompletten Speicher.

Default:

```
EESAVE : 1 (unprogrammiert)
```

BODEN

BODEN aktiviert/deaktiviert die Brown Out Detection. Bei manchen Controllern wird diese Funktion durch die BODLEVEL Fusebits übernommen.

Default:

```
BODEN : 1 (unprogrammiert)
```

BODLEVEL

Mit BODLEVEL kann der Spannungswert eingestellt werden bei dem der Unterspannungsschutz aktiv werden soll. Ältere Controller (zb ATmega128) haben nur zwei Schwellwerte. Mit BODLEVEL kann zwischen den Werten gewechselt werden, mit BODEN wird die Funktion komplett deaktiviert. Neuere Controller (zb ATmega168) haben 3 BODLEVEL Fusebits mit denen mehrere Schwellwerte eingestellt werden können bzw die gesamte Funktion deaktiviert wird. Ab Werk ist bei allen Typen die BOD Funktion abgeschaltet.

Default:

```
BODLEVEL : 1 (unprogrammed)
```

WDTON

Mit WDTON kann der Watchdog Timer permanent aktiviert werden. Ist dieses Fusebit nicht programmiert (1) kann der Watchdog per Software gesteuert werden.

Default:

```
WDTON : 1 (unprogrammiert)
```

BOOTRST

BOOTRST bestimmt an welche Adresse nach einem Reset gesprungen wird. Unprogrammiert (1) springt der Controller nach einem Reset an Adresse $0000. Wird das Fusebit programmiert springt der Controller nach einem Reset an den Beginn des Bootloaders. Die Adresse hängt vom Controller und von den Einstellungen der BOOTSZ Fusebits ab.

Default:

```
BOOTRST : 1 (unprogrammiert)
```

BOOTSZ

Mit BOOTSZ wird die Größe des Speicherbereiches bestimmt der für den Bootloader reserviert wird. Die Größe ist abhängig vom Controllertyp. Dieser Speicherbereich befindet sich immer am Ende des Flash Adressraumes..

Default:

```
BOOTSZ : siehe Datenblatt
```

Compatibility Bits

Viele Controller haben ein Compatibility Bit. Mit diesem Bit lässt sich der Controller in einen Modus versetzen in dem er sich exakt so verhält wie sein Vorgänger. Beim ATmega128 gibt es zB das M103C Bit. Der ATmega128 verhält sich also wie ein ATmega103.

Ob das Compatibility Bit ab Werk programmiert ist oder nicht hängt vom Controller ab.

SELFPRGEN

SELFPRGEN aktiviert/deaktiviert die Self Programming Funktion.

Default:

```
SELFPRGEN : 1 (unprogrammiert)
```

HWBEN

HWBEN aktiviert/deaktiviert die Hardware Boot Funktion
Default:
`HWBEN : 0 (programmiert)`

Sollte ich ein Fusebit vergessen haben oder neue dazukommen bitte ergänzen.

Fusebits mit dem AVR Studio programmieren:

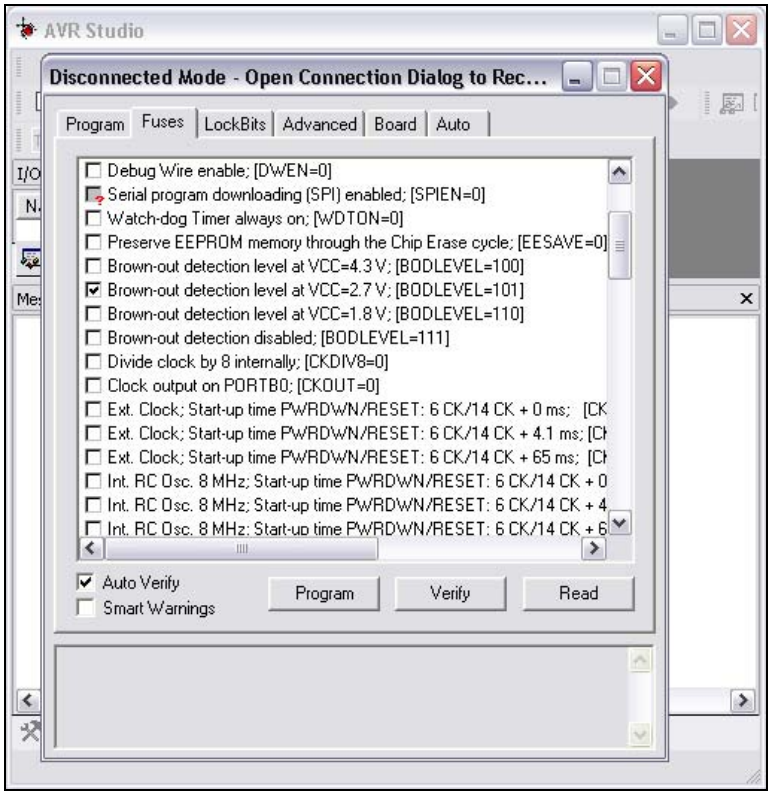


Abbildung 6: Fusebits im AVR Studio

Am einfachsten und intuitivsten lassen sich die Fusebits mit dem AVR Studio programmieren. Für jedes Fusebit gibt es eine kurze Beschreibung und den Default Wert. Gibt es für eine Funktion mehrere Fusebits wird für jede Kombination ein Häkchen mit Beschreibung und Bitkombination angezeigt.

In diesem Beispiel (ATmega168) sieht man zb das die Brown Out Detection auf 2,7V eingestellt wurde.

8.8 AVR Fuse Konfiguration ATmega8L

Der ATmega8L besitzt zwei Fuse-Bytes.

Fuse Bytes des ATmega8L:

The ATmega8 has two fuse bytes. Table 87 and Table 88 describe briefly the functionality of all the fuses and how they are mapped into the fuse bytes. Note that the fuses are read as logical zero, "0", if they are programmed.

Table 87. Fuse High Byte

Fuse High Byte	Bit No.	Description	Default Value
RSTDISBL[4]	7	Select if PC6 is I/O pin or RESET pin	1 (unprogrammed, PC6 is RESET-pin)
WDTON	6	WDT always on	1 (unprogrammed, WDT enabled by WDTCR)
SPIEN[1]	5	Enable Serial Program and Data Downloading	0 (programmed, SPI prog. enabled)
CKOPT[2]	4	Oscillator options	1 (unprogrammed)
EESAVE	3	EEPROM memory is preserved through the Chip Erase	1 (unprogrammed, EEPROM not preserved)
BOOTSZ1	2	Select Boot Size (see Table 82 for details)	0 (programmed)[3]
BOOTSZ0	1	Select Boot Size (see Table 82 for details)	0 (programmed)[3]
BOOTRST	0	Select Reset Vector	1 (unprogrammed)

Notes:
1. The SPIEN Fuse is not accessible in Serial Programming mode.
2. The CKOPT Fuse functionality depends on the setting of the CKSEL bits, see "Clock Sources" on page 26 for details.
3. The default value of BOOTSZ1..0 results in maximum Boot Size. See Table 82 on page 220.
4. When programming the RSTDISBL Fuse Parallel Programming has to be used to change fuses or perform further programming.

Table 88. Fuse Low Byte

Fuse Low Byte	Bit No.	Description	Default Value
BODLEVEL	7	Brown out detector trigger level	1 (unprogrammed)
BODEN	6	Brown out detector enable	1 (unprogrammed, BOD disabled)
SUT1	5	Select start-up time	1 (unprogrammed)[1]
SUT0	4	Select start-up time	0 (programmed)[1]
CKSEL3	3	Select Clock source	0 (programmed)[2]
CKSEL2	2	Select Clock source	0 (programmed)[2]
CKSEL1	1	Select Clock source	0 (programmed)[2]
CKSEL0	0	Select Clock source	1 (unprogrammed)[2]

Notes:
1. The default value of SUT1..0 results in maximum start-up time. SeeTable 10 on page 30 for details.
2. The default setting of CKSEL3..0 results in internal RC Oscillator @ 1MHz. See Table 2 on page 26 for details.

The status of the Fuse Bits is not affected by Chip Erase. Note that the Fuse Bits are locked if lock bit1 (LB1) is programmed. Program the Fuse Bits before programming the Lock Bits.

The fuse values are latched when the device enters Programming mode and changes of the fuse values will have no effect until the part leaves Programming mode. This does not apply to the EESAVE Fuse which will take effect once it is programmed. The fuses are also latched on Power-up in Normal mode.

Festlegung der Fuse-Bits für das Projekt Hanging Man:

Fuse High Byte $D1:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RSTDISBL	WDTON	SPIEN	CKOPT	EESAVE	BOOTSZ1	BOOTSZ0	BOOTRST
1	1	0	1	0	0	0	1

Tabelle 14: Fuse High Byte ATmega8L

Fuse Low Byte - $FF:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
BODLEVEL	BODEN	SUT1	SUT0	CKSEL3	CKSEL2	CKSEL1	CKSEL0
1	1	1	1	1	1	1	1

Tabelle 15: Extended Fuse Byte ATmega8L

Bemerkung:
0 = programmiert (!! negative Logik !!)
1 = unprogrammiert (!! negative Logik !!)

	Default (keine Programmierung)
	Wichtige Programmierung (abweichend von Default)
	Sonstige Systemeinstellung abweichend von Default)

CKSEL3-0 ist im Originalzustand laut Datenblatt auf 0001. Das entspricht dem internen Oszillator mit 1MHz. Uns interessiert aber der Zustand mit externem Quarz mit 8 MHz. Dazu muss CKSEL3-1 auf 111 gesetzt werden. Die Start-Up-Time wird durch CKSEL0 und SUT bestimmt. Eine 1 auf CKSEL0 bedeutet zusammen mit 11 auf SUT1-0 die längstmögliche Startverzögerung. So hat der Quarz die meiste Zeit zum Anschwingen. Wie beim Atmega üblich, bedeutet eine Null ein Häkchen und bei einer Eins bleibt das Feld leer.

8.8.1 Die Fuse-Konfiguration von Hanging Man

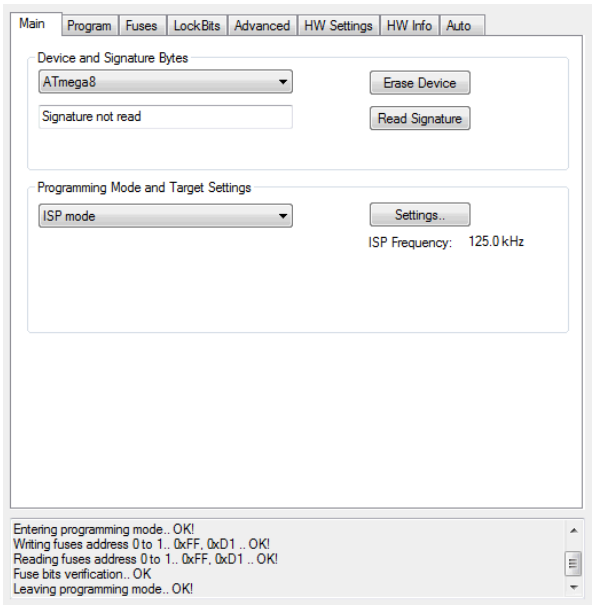


Tabelle 16: AVR-Studio - Main

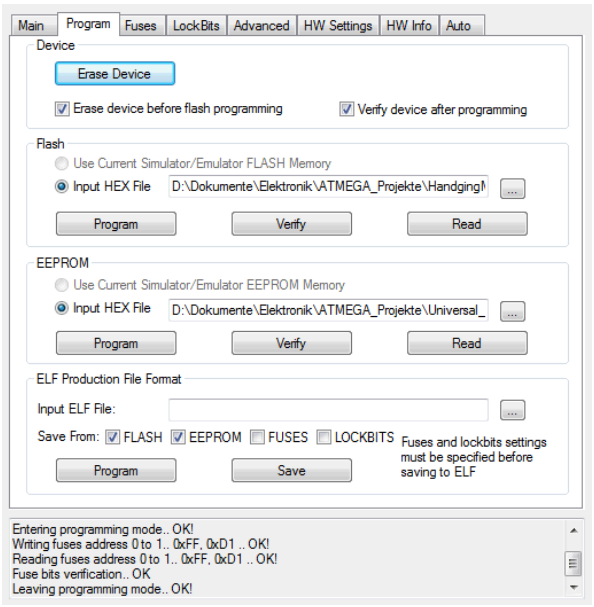


Tabelle 17: AVR-Studio - Program

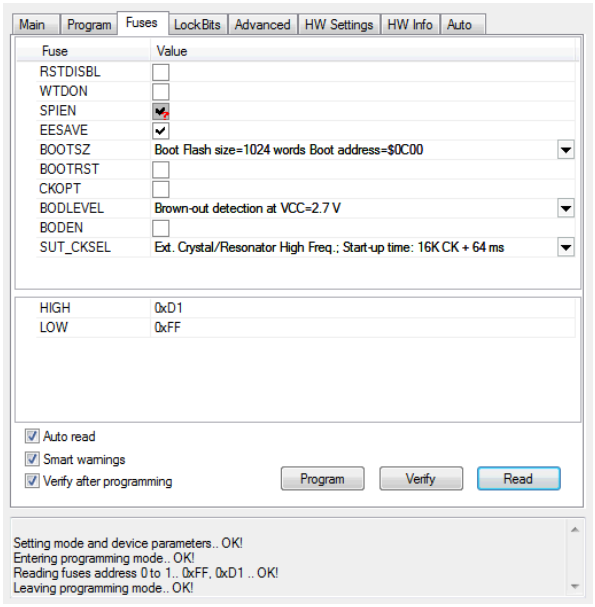


Tabelle 18: AVR-Studio - Fuses

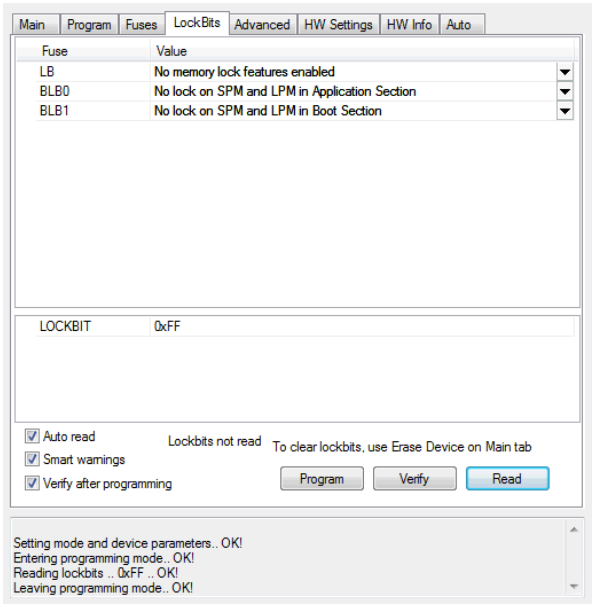


Tabelle 19: AVR-Studio - LockBits

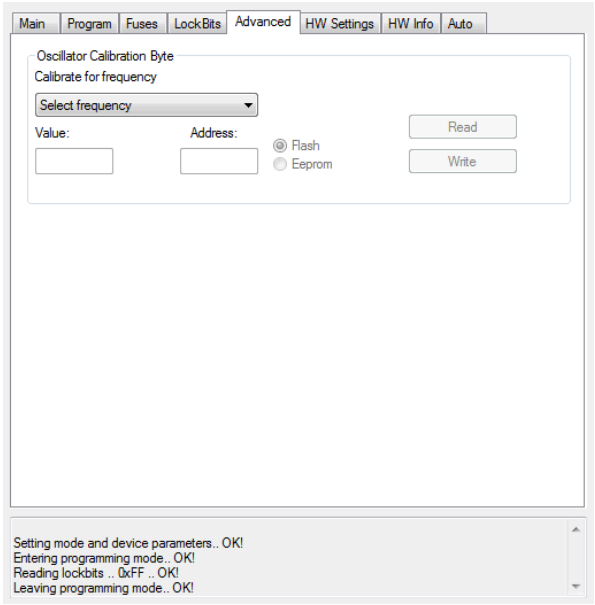


Tabelle 20: AVR-Studio - Advanced

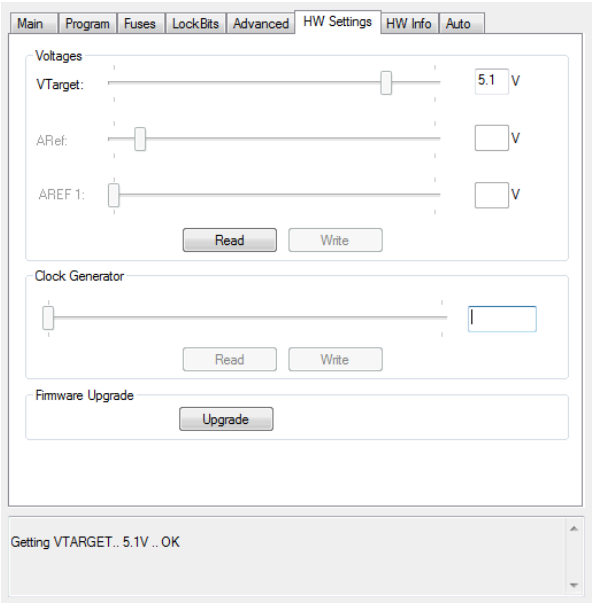


Tabelle 21: AVR-Studio – MainHW Settings

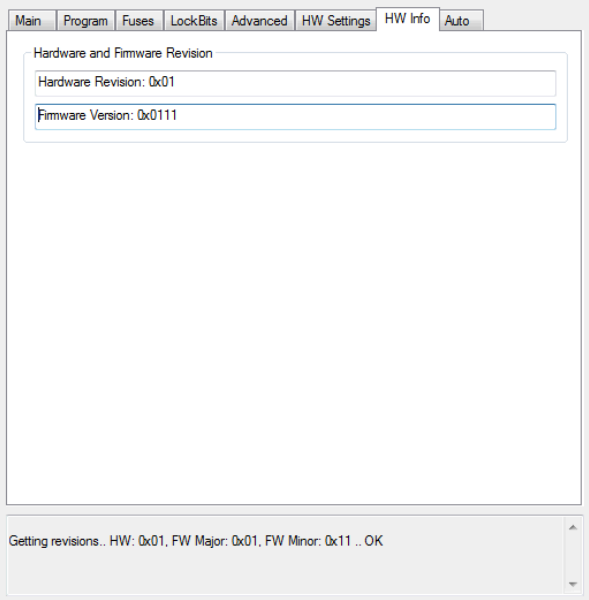


Tabelle 22: AVR-Studio – HW Info

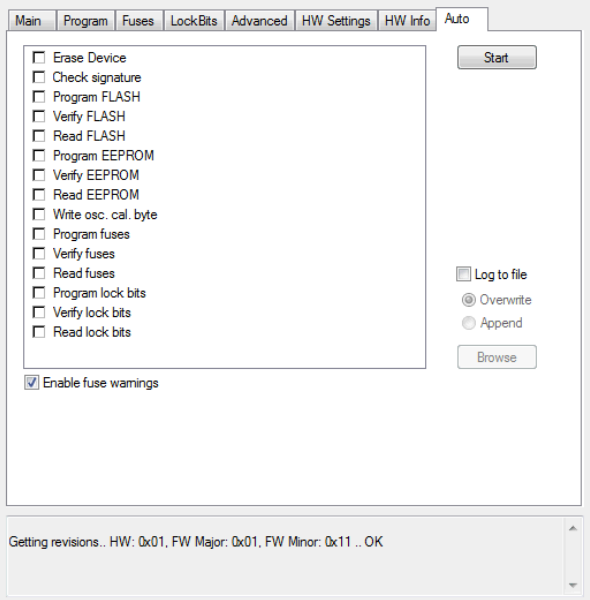


Tabelle 23: AVR-Studio - Auto

8.9 Grundlagen zur Spannung 5V, Vcc und VDD

Für die Spannungsversorgung des Projekts HangingMan wird die Spannung +5V und GND bereitgestellt.

Unterschiedliche Technologien haben unterschiedliche Bezeichnungen für die notwendigen Spannungsversorgungen.

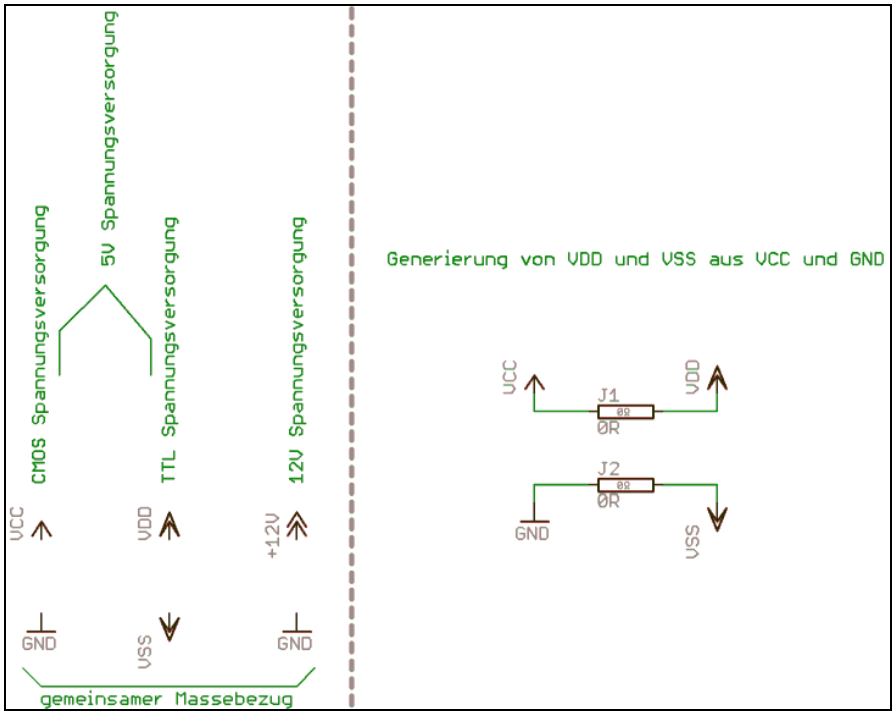
Für CMOS Bausteine gelten die Bezeichnungen VCC und GND.

TTL-Logik verwendet die Bezeichnungen VDD und VSS. VDD und VSS werden im Projekt durch 0Ω-Widerstände (Drahtbrücken) erzeugt um EAGLE die korrekte Behandlung der Spannungen zu ermöglichen.

Für das Projekt gelten die Folgenden Bezeichnungen in den Netzen:

Spannungspotential	*Verwendeter Name*
GND (Ground) = MASSE = 0V	GND VSS
+5V Spannung	VCC VDD
+12V	+12V

Tabelle 24: Vorgeschriebene Namensgebung für Spannungsversorgungen



Schaltbild 4: Symbolfestlegung für Spannungsversorgungen

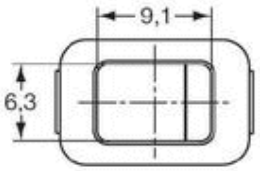
9 Elektronische Teilkomponenten

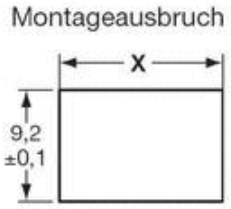
9.1 Hauptschalter

Das Spiel kann durch einen kleinen Hauptschalter ein- und ausgeschaltet werden. Folgender Hauptschalter wurde ausgewählt:

Einpoliger Subminiatur-Wippenschalter 250 V/AC 3 A

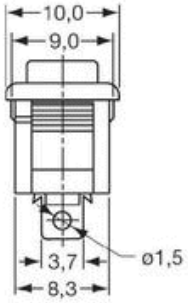
Conrad:
Bestell-Nr.: 700039 - 62
EAN: 2050000213762





X	Länge Ausschnitt	Frontplattenstärke
A	13,6 $^{+0,00}_{-0,10}$	0,75~1,25
B	13,7 $^{+0,00}_{-0,10}$	1,25~2,00
C	13,8 $^{+0,00}_{-0,10}$	2,00~2,50*

*: 2,50mm = maximale Plattenstärke



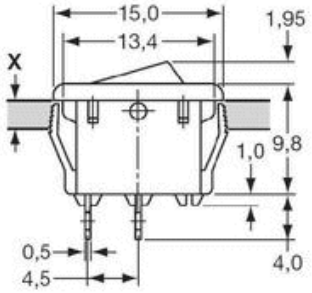


Abbildung 7: Hauptschalter

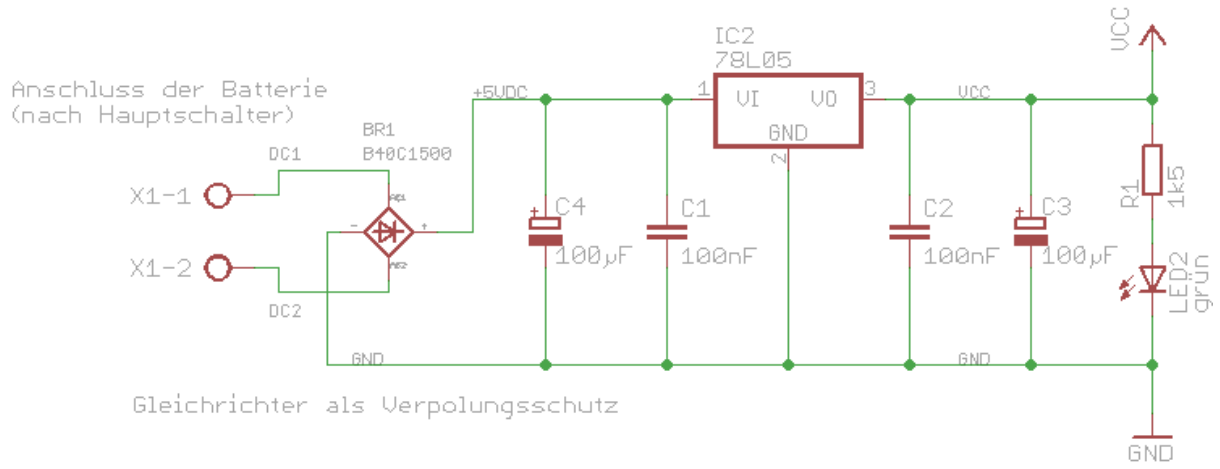
9.2 Spannungsversorgung

Die Spannungsversorgung des Spiels erfolgt mittels 9V Blockbatterie. Für die Aufnahme der Batterie im Gehäuse ist ein Batteriefach vorhanden.

Die Elektronik des Spiels arbeitet mit +5V. Diese Spannung wird mittels Festspannungsregler erzeugt.

Für die Batteriespannung ist eine Überwachungslogik vorgesehen. Hierzu wird die Spannung vom Mikrocontroller überwacht. Sinkt die Spannung auf eine zuvor festgelegte Grenze so wird dies über eine rote LED angezeigt und die Batterie muss gewechselt werden.

9.2.1 Beschaltung des Festspannungsreglers:



Schaltbild 5: Beschaltung des Festspannungsreglers

Bauteile:

Stückliste: Beschaltung Spannungsversorgung			
Widerstände		***Halbleiter***	
R1	1k5 Ω	IC2	5V Festspannungsregler 78L05
		BR1	Brückengleichrichter B40C1500
Konensatoren		LED2	LowCurrent 3mm LED grün
C1, C2	Folienkondensator 100nF		
C3, C4	Elektrolytkondensato 100µF		

Tabelle 25: Stückliste LCD Beschaltung

Anstelle des Brückengleichrichters kann auch ein Diodenarray bestehend aus 4 x 1N4001 Universaldioden verwendet werden. Die Aufgabe des Brückengleichrichters / Diodenarrays besteht lediglich darin, die Elektronik gegen (kurzzeitige) Verpolung der 9V Batterie zu schützen.

9.3 LCD-Display

Allgemeines:

Als Anzeigedisplay wird ein LCD-Modul der Firma ELECTRONIC ASSEMBLY der DOG_Serie 3,3V EA DOG-M Super Flach / 55x27 mm inkl. Kontroller ST7036 für 4-/8-BIT SPI (4-Draht) eingesetzt.

Als Hintergrundbeleuchtung wird LED-Beleuchtung Weiß eingesetzt.

Zum Einsatz kommt das folgende Display:

LCD-Modul 2x16 – 5,57 mm	EA DOGM162B-A
LED Hintergrundbeleuchtung Weiß	EA LED55X31W

EA DOGM162B-A: Buchstabenhöhe 3,6 mm

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6
1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6

Abbildung 8: LCD 2x16

Für das LCD-Display wird eine weiße LED-Hintergrundbeleuchtung eingesetzt.

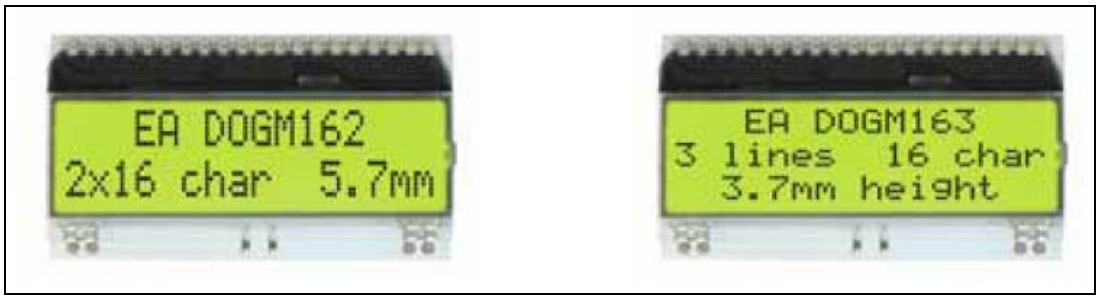
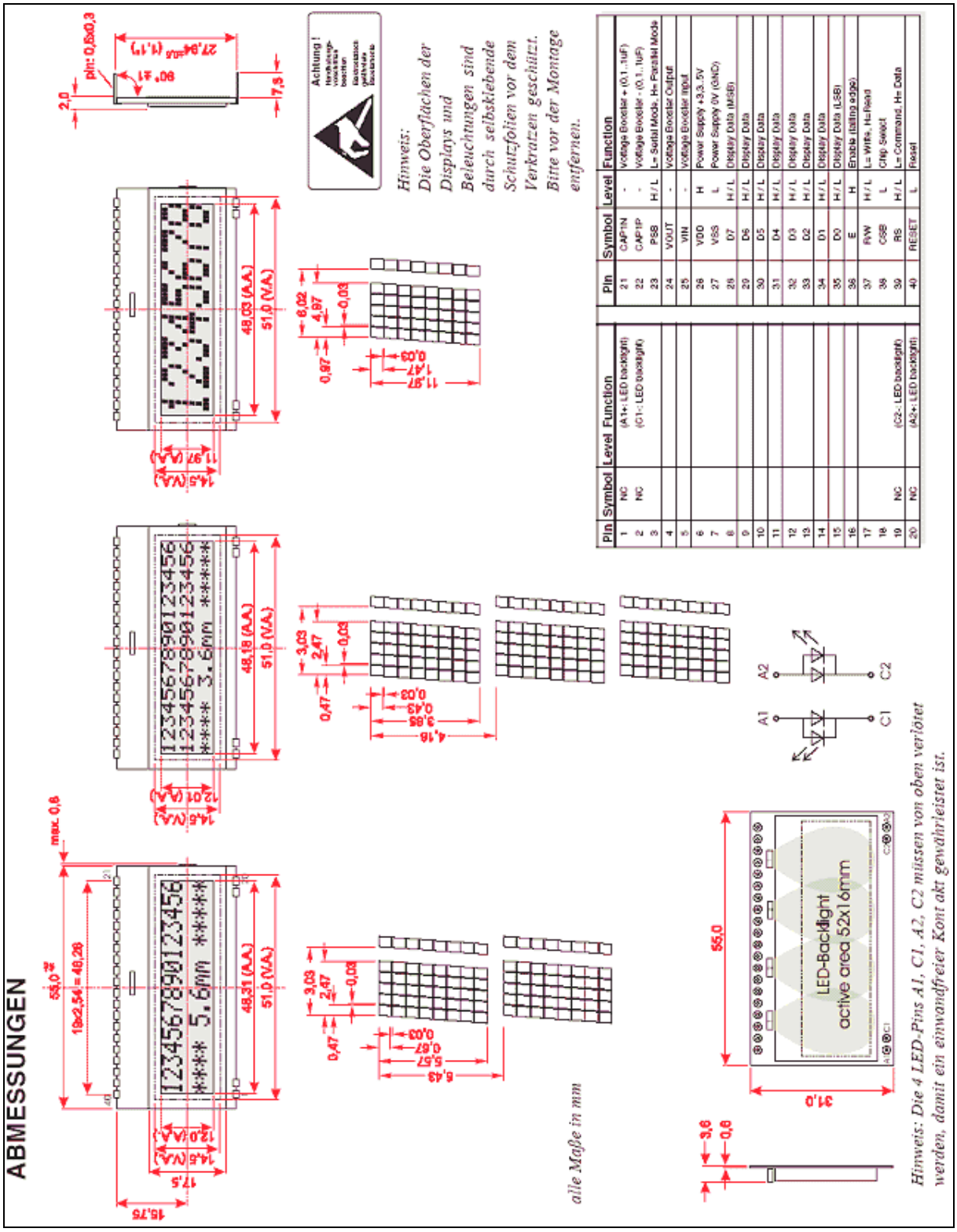


Abbildung 9: LCD-Display EA DOG-M

Technische Daten:

- Kontrastreiche LCD-Supertwist Anzeige
- Optionale LED-Beleuchtungskörper in verschiedenen Farben
- 1x8, 2x16 und 3x16 Zeichen mit 12,0 mm / 5,6 mm und 3,6 mm Schrift
- Kontroller ST 7036 für 4-BIT, 8-BIT und SPI (4-DRAHT) Interface
- Spannungsversorgung +3,3V oder +5V single supply (typ. 250µA)
- Keine zus. Spannungen erforderlich
- Betriebstemperaturbereich -20..+70°C
- LED-Hintergrundbeleuchtung 3..80mA@3,3V oder 2..40mA@5V
- Keine Montage erforderlich: einfach nur in PCB einlöten.

Abmessungen:



Pin	Symbol	Level	Function
1	NC		(A1+: LED backlight)
2	NC		(C1-: LED backlight)
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			
17			
18			
19	NC		(C2-: LED backlight)
20	NC		(A2+: LED backlight)

Pin	Symbol	Level	Function
21	CAP1N	-	Voltage Booster + (0.1..1uF)
22	CAP1P	-	Voltage Booster - (0.1..1uF)
23	PS8	H/L	L= Sutial Mode, H= Parallel Mode
24	VOUT	-	Voltage Booster Output
25	VIN	-	Voltage Booster Input
26	VDD	H	Power Supply +3,3..5V
27	VSS	L	Power Supply 0V (GND)
28	D7	H/L	Display Data (MSB)
29	D6	H/L	Display Data
30	D5	H/L	Display Data
31	D4	H/L	Display Data
32	D3	H/L	Display Data
33	D2	H/L	Display Data
34	D1	H/L	Display Data
35	D0	H/L	Display Data (LSB)
36	E	H	Enable (falling edge)
37	R/W	H/L	L= Write, H=Read
38	CSB	L	Chip Select
39	RS	H/L	L= Command, H= Data
40	RESET	L	Reset

Abbildung 10: Abmessungen und technische Daten DOGM LCD-Display

Kontrasteinstellung:

Für alle Displays der EA DOG- Serie ist der Kontrast per Befehl einstellbar. Dies erfolgt über die Bits C0..C5 in den Befehlen "Contrast Set" und "Power/Icon Control/Contrast Set". In der Regel wird der Kontrast einmalig eingestellt und dann - dank integrierter Temperaturkompensation - über den gesamten Betriebstemperaturbereich (-20..+70°C) konstant gehalten.

Insgesamt benötigen die Displays selbst im 3,3V Betrieb keine zusätzliche negative Spannung!

LED-Hintergrundbeleuchtung für LCD-Display:

white EA LED55x31-W	Forward voltage	Current max.	Limiting resistor	
			@ 3,3 V	@ 5 V
Connected in parallel	3,2 V	60 mA	1,6 ohm	30 ohm
Connected in series	6,4 V	30 mA	-	-

Für die Beschaltung der Hintergrundbeleuchtung wird ein Vorwiderstand von 30 Ohm benötigt. Eine Serienschaltung der LED's ist nicht möglich weil hierfür eine Forward-Spannung von 6,4V benötigt wird, das Display aber nur mit 5V betrieben wird. Somit kommt nur Parallelschaltung in Frage!

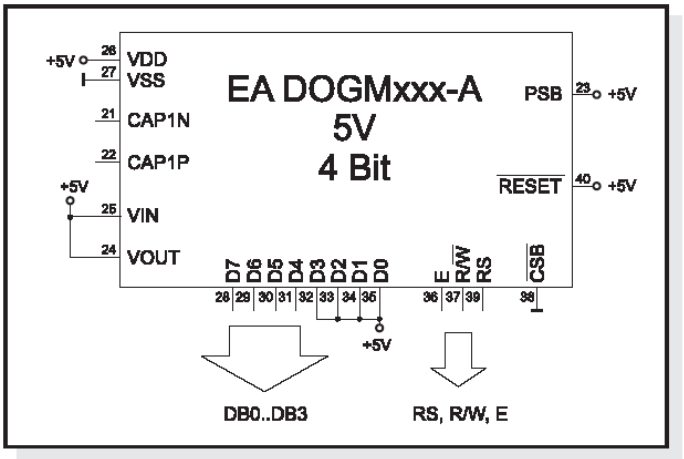
Tabelle 26: LED-Hintergrundbeleuchtung für LCD-Display

Für die Hintergrundbeleuchtung wird eine PWM über den ATmega realisiert mit Hilfe derer die Beleuchtungshelligkeit eingestellt werden kann. Siehe hierzu Beschaltung des LCD-Displays und separate Transistor-Stufe.

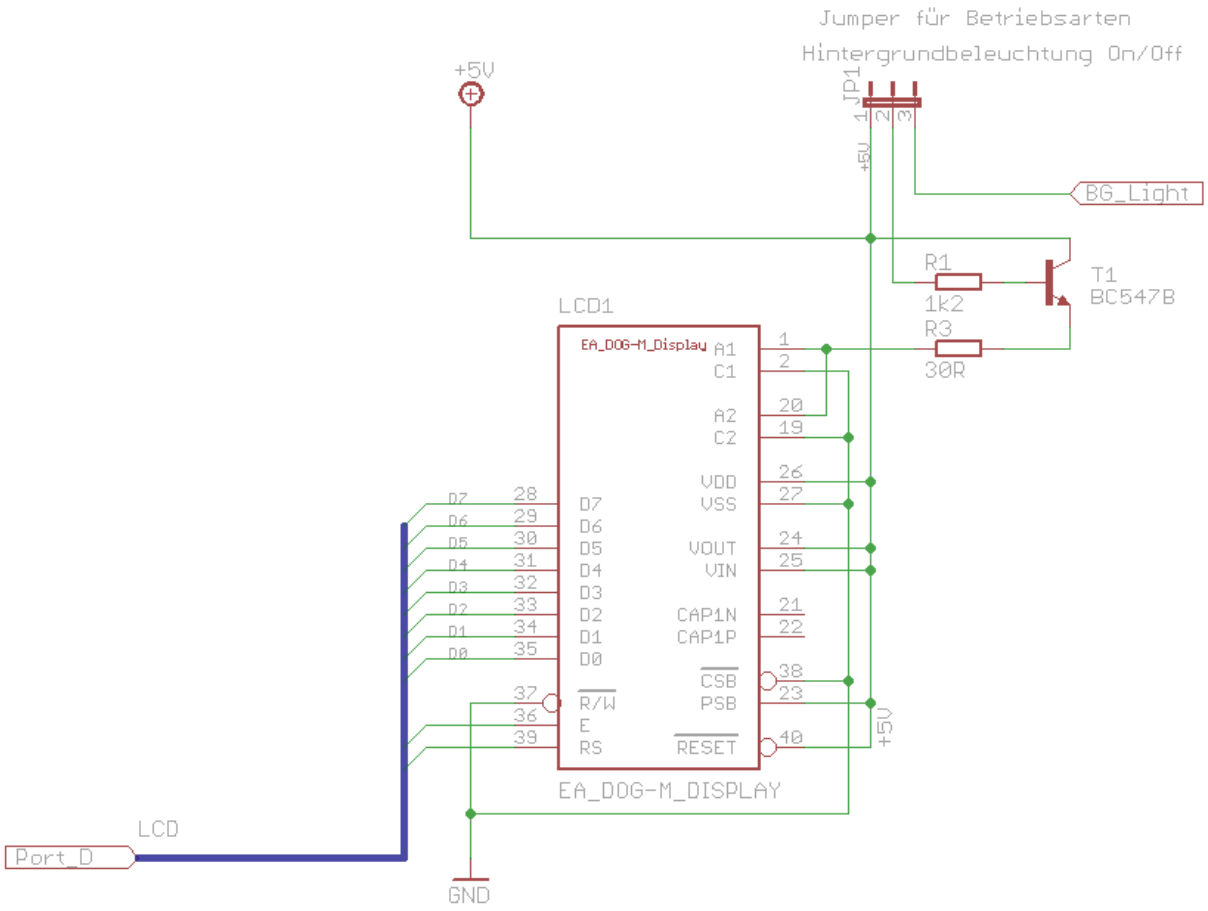
Das LCD-Display muss vor dem verlöten oder stecken auf die Beleuchtungseinheit gelötet werden, damit die LEDs der Beleuchtung über die Kontakte des LCDs Strom bekommen. Es ist ratsam, alle Pins der Beleuchtungseinheit anzulöten, da sich so der Druck beim Einsetzten des Displays in die Fassung besser verteilt. Elektrisch ist dies nicht notwendig. Bitte sehr sparsam mit dem Lötzinn umgehen, da es sonst an den Beinchen herunter läuft und somit das Display nicht in die Fassung passt.

9.3.1 Beschaltung des LCD-Display

Die Beschaltung des LCD-Display an den ATmega erfolgt wie im folgenden Schaltbild aus dem Datenblatt des LCD-Displays gefordert:



Schaltbild 6: Beschaltung LCD-Display gemäß Datenblatt



Schaltbild 7: Beschaltung LCD-Display im Projekt Hanging Man

Bemerkung:

Auf der Zielplatine des Projekts wird auf den Jumper verzichtet da die Helligkeitseinstelung nur über PWM realisiert wird. Hierzu dient die PWM des Timers 2 welcher sich bei OC2 den Pin mit MOSI teilt!
Auf der Zielplatine wird das LCD-Display im 4-Bit-Modus angesprochen und nicht im 8-Bit-Modus!

Bauteile:

Stückliste: LCD Beschaltung			
Widerstände		***Halbleiter***	
R1	1k2 Ω	LCD1	LCD Display EA DOGM162B-A
R3	30 Ω	T1	Transistor BC547B
Sonstiges			
JP1	Jumper 3-polig, 2,54mm Raster		

Tabelle 27: Stückliste LCD Beschaltung

Bemerkung zur LCD-Beschaltung:

- Über den Jumper JP1 kann eingestellt werden, ob die Beleuchtung mit konstanter Spannung +5V und gleichbleibender Helligkeit betrieben wird oder ob eine Ankopplung an den ATmega zur PWM erfolgt.

Ressourcenzuordnung zum ATmega8L:

Nummer	*Schaltbild*		*Ressource ATmega8L*		
1	MOSI_PWM	[PWM]	PortB.3	[OC2]	(Timer2)
2	Out_Port LCD:D4	[LCD_D0]	PortC.2	[PC2]	(GPIO)
3	Out_Port LCD:D5	[LCD_D1]	PortC.3	[PC3]	(GPIO)
4	Out_Port LCD:D6	[LCD_D2]	PortC.4	[PC4]	(GPIO)
5	Out_Port LCD:D7	[LCD_D3]	PortC.5	[PC5]	(GPIO)
6	Out_Port LCD:E	[LCD_E]	PortD.6	[PD6]	(GPIO)
7	Out_Port LCD:RS	[LCD_RS]	PortD.7	[PD7]	(GPIO)

Tabelle 28: Ressourcenzuordnung ATmega8L für LCD-Display

9.3.2 Zeichensatz und Befehlstabellen

Der unten abgebildete Zeichensatz ist integriert. Zusätzlich können 8 eigene Zeichen frei definiert werden.

Tabelle 29: Zeichensatz des LCD-Displays

Befehlstabellen des Display:

Instruction	Instruction Code										Description	Instruction Execution Time		
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		OSC= 380kHz	OSC= 540kHz	OSC= 700kHz
Clear Display	0	0	0	0	0	0	0	0	0	1	Write "20H" to DDRAM. and set DDRAM address to "00H" from AC	1.08 ms	0.76 ms	0.59 ms
Return Home	0	0	0	0	0	0	0	0	1	x	Set DDRAM address to "00H" from AC and return cursor to its original position if shifted. The contents of DDRAM are not changed.	1.08 ms	0.76 ms	0.59 ms
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	Sets cursor move direction and specifies display shift. These operations are performed during data write and read.	26.3 µs	18.5 µs	14.3 µs
Display ON/OFF	0	0	0	0	0	0	1	D	C	B	D=1:entire display on C=1:cursor on B=1:cursor position on	26.3 µs	18.5 µs	14.3 µs
Function Set	0	0	0	0	1	DL	N	DH	IS2	IS1	DL: interface data is 8/4 bits N: number of line is 2/1 DH: double height font IS[2:1]: instruction table select	26.3 µs	18.5 µs	14.3 µs
Set DDRAM Address	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Set DDRAM address in address counter	26.3 µs	18.5 µs	14.3 µs
Read Busy Flag and Address	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Whether during internal operation or not can be known by reading BF. The contents of address counter can also be read.	0	0	0
Write Data to RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0	Write data into internal RAM (DDRAM/CGRAM/ICONRAM)	26.3 µs	18.5 µs	14.3 µs
Read Data from RAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0	Read data from internal RAM (DDRAM/CGRAM/ICONRAM)	26.3 µs	18.5 µs	14.3 µs

Tabelle 30: LCD-Display EA DOGM Instruction Code

Instruction table 0(IS[2:1]=[0,0])														
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	X	X	S/C and R/L: Set cursor moving and display shift control bit, and the direction, without changing DDRAM data.	26.3 µs	18.5 µs	14.3 µs
Set CGRAM	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	Set CGRAM address in address counter	26.3 µs	18.5 µs	14.3 µs

Tabelle 31: LCD-Display EA DOGM Instruction table 0

Instruction table 1(IS[2:1]=[0,1])														
Bias Set	0	0	0	0	0	1	BS	1	0	FX	BS=1:1/4 bias BS=0:1/5 bias FX: fixed on high in 3-line application and fixed on low in other applications.	26.3 µs	18.5 µs	14.3 µs
Set ICON Address	0	0	0	1	0	0	AC3	AC2	AC1	AC0	Set ICON address in address counter.	26.3 µs	18.5 µs	14.3 µs
Power/ICON Control/ Contrast Set	0	0	0	1	0	1	Ion	Bon	C5	C4	Ion: ICON display on/off Bon: set booster circuit on/off C5,C4: Contrast set for internal follower mode.	26.3 µs	18.5 µs	14.3 µs
Follower Control	0	0	0	1	1	0	Fon	Rab 2	Rab 1	Rab 0	Fon: set follower circuit on/off Rab2~0: select follower amplified ratio.	26.3 µs	18.5 µs	14.3 µs
Contrast Set	0	0	0	1	1	1	C3	C2	C1	C0	Contrast set for internal follower mode.	26.3 µs	18.5 µs	14.3 µs

Tabelle 32: LCD-Display EA DOGM Instruction table 1

Instruction table 2(IS[2:1]=[1,0])														
Double Height Position Select	0	0	0	0	0	1	UD	X	x	x	UD: Double height position select	26.3 µs	18.5 µs	14.3 µs
Reserved	0	0	0	1	X	X	X	X	X	X	Do not use (reserved for test)	26.3 µs	18.5 µs	14.3 µs

Tabelle 33: LCD-Display EA DOGM Instruction table 2

Eine detaillierte Beschreibung des hier integrierten Kontrollers ST7036 finden Sie im Internet unter
http://www.lcd-module.de/eng/pdf/zubehoer/st7036.pdf

Initialisierungsbeispiel für EA DOGM162 bei 8-Bit und 5V:

INITIALISIERUNGSBEISPIEL FÜR 8 Bit / 5V EA DOGM162												
Befehl	**RS**	**R/W**	**DB7**	**DB6**	**DB5**	**DB4**	**DB3**	**DB2**	**DB1**	**DB0**	**Hex**	**Bemerkung**
Function Set	0	0	0	0	1	1	1	0	0	1	$39	8-Bit Datenlänge, 2 Zeilen, Instruction table 1
Bias Set	0	0	0	0	0	1	1	1	0	0	$1C	BS: 1/4, 2-zeiliges LCD
Power Control	0	0	0	1	0	1	0	0	1	0	$52	Booster aus, Kontrast C5, C4setzen
Follower Control	0	0	0	1	1	0	1	0	0	1	$69	Spannungsfolger und Verstärkung setzen
Contrast Set	0	0	0	1	1	1	0	1	0	0	$74	Kontrast C3, C2, C1 setzen
Display ON/OFF	0	0	0	0	0	0	1	1	1	1	$0F	Display ein, Cursor ein, Cursor blinken
Clear Display	0	0	0	0	0	0	0	0	0	1	$01	Display löschen, Cursor Home
Entry Mode Set	0	0	0	0	0	0	0	1	1	0	$06	Cursor Auto-Increment

Tabelle 34: LCD-Display EA DOGM162 Initialisierungsbeispiel

Initialisierungsbeispiel für EA DOGM163 bei 8-Bit und 5V:

INITIALISIERUNGSBEISPIEL FÜR 8 Bit / 5V EA DOGM163												
Befehl	**RS**	**R/W**	**DB7**	**DB6**	**DB5**	**DB4**	**DB3**	**DB2**	**DB1**	**DB0**	**Hex**	**Bemerkung**
Function Set	0	0	0	0	1	1	1	0	0	1	$39	8-Bit Datenlänge, 2 Zeilen, Instruction table 1
Bias Set	0	0	0	0	0	1	1	1	0	1	$1D	BS: 1/4, 3-zeiliges LCD
Power Control	0	0	0	1	0	1	0	0	0	0	$50	Booster aus, Kontrast C5, C4setzen
Follower Control	0	0	0	1	1	0	1	1	0	0	$6C	Spannungsfolger und Verstärkung setzen
Contrast Set	0	0	0	1	1	1	1	1	0	0	$7C	Kontrast C3, C2, C1 setzen
Display ON/OFF	0	0	0	0	0	0	1	1	1	1	$0F	Display ein, Cursor ein, Cursor blinken
Clear Display	0	0	0	0	0	0	0	0	0	1	$01	Display löschen, Cursor Home
Entry Mode Set	0	0	0	0	0	0	0	1	1	0	$06	Cursor Auto-Increment

Tabelle 35: LCD-Display EA DOGM163 Initialisierungsbeispiel

9.3.3 PWM-Einstellungen für Hintergrundbeleuchtung

Die Hintergrundbeleuchtung des LCD Display wird mittels vom ATmega erzeugten PWM eingestellt.

Der Helligkeitswert des Display kann aber zur Spielzeit nicht geändert werden sondern er ist empirisch ermittelt und softwareseitig festgelegt.

Der Quarz für den Prozessor arbeitet mit 8 MHz. Für die Prescaler-Einstellung wird der Wert 128 festgelegt. Damit ergibt sich für die PWM-Frequenz folgender Wert:

Quarz = 8 MHz ; Prescaler = 128 ; Timer = 8 Bit ergibt: (8000000Hz / 128) / (256*2) = **122,07 Hz**

9.3.4 BASCOM Beispielcode für die Displayansteuerung (Funktionsbibliothek)

```
'#################################################################################
' LCD-Display.BAS                                                Stand 03.05.2014
' ---------------                                                (C) Markus Fulde
'
' Testprogramm zur Inbetriebnahme des Display DOGM163
' Die Inbetriebnahme erfolgt mit dem Eva-Board STK500 + STK501
' Die für die LCD-Display-Ansteuerung notwendigen Routinen sind
' als solche gekennzeichnet und bereits zur späteren Verwendung
' ausgelagert.
'#################################################################################

'--------------------------------------------------------------------------------
' Compilerinstruktionen und Compilerdirektiven
'--------------------------------------------------------------------------------
$regfile = "m8def.dat"                                    ' Definitionsdatei für ATmega8 laden
$crystal = 8000000                                        ' Quarzfrequenz für 16 MHz festlegen

$baud = 19200                                             ' Baudrate für RS232 Traceausgabe defi-
nieren


'--------------------------------------------------------------------------------
' Allgemeine Zusatzinformatonen zu Programmbeginn
'--------------------------------------------------------------------------------


'--------------------------------------------------------------------------------
' Definition von Ressourcen
'--------------------------------------------------------------------------------

' ----- LED's -----
Alive_pin Alias PinB.0                                    ' GPIO für Alive-LED (für DDR oder In-
put)
```

```
Alive Alias PortB.0                                     ' GPIO für Alive-LED (für Output oder
Pullup)

Pwrled_pin Alias PINB.1                                 ' GPIO für Power-LED (für DDR oder In-
put)
Pwrled Alias PORTB.1                                    ' GPIO für Power-LED (für Output oder
Pullup)

' ----- LCD-Display -----
' LCD-Display
Db4_pin Alias PortC.2                                   ' GPIO für LCD Pin4
Db5_pin Alias PortC.3                                   ' GPIO für LCD Pin5
Db6_pin Alias PortC.4                                   ' GPIO für LCD Pin6
Db7_pin Alias PortC.5                                   ' GPIO für LCD Pin7
E_pin Alias PortD.6                                     ' GPIO für LCD E
Rs_pin Alias PortD.7                                    ' GPIO für LCD RS

' ----- Test: Tasten zu Testzwecken -----
Key1_pin Alias PIND.2                                   ' GPIO für Key 1 (für DDR oder Input)
Key1 Alias PORTD.2                                      ' GPIO für Key 1 (für Output oder Pul-
lup)

Key2_pin Alias PIND.3                                   ' GPIO für Key 2 (für DDR oder Input)
Key2 Alias PORTD.3                                      ' GPIO für Key 2 (für Output oder Pul-
lup)


'-------------------------------------------------------------------------------
' Definition von Konstaten
'-------------------------------------------------------------------------------

' ----- Für Testumgebung bzw. Traceausgaben -----
Const Main_testmodus = 1                                ' Flag für Testmodus Allgemeinsystem
Const Lcd_testmodus = 1                                 ' Flag


' ----- Allgemeine Systemkonstanten -----

' Tatsächliches Allgemeines
' Const Led_aus = 0
' Const Led_ein = 1

Const Led_aus = 1 ' Achtung !! bei STK500 ist Logik gedreht!!
Const Led_ein = 0 ' Achtung !! bei STK500 ist Logik gedreht!!

Const False = 0
Const True = 1

Const Pullup_aus = 0
Const Pullup_ein = 1

' Zeitvorgabe für Sekunden-Timer
Const Timervorgabe = 34286                              ' Timer von 1 Sekunden (SekundenTick)


' ----- LCD -----

' Anmerkung: Wert von 35 ist der beste Wert welcher durch Versuche ermittelt wurde!!
Const Lcd_kontrast_default = 35                         ' LCD-Default-Kontrast
' Anmerkung: Wert von 70 ist der beste Wert welcher durch Versuche ermittelt wurde!!
Const Lcd_helligkeit_default = 70                       ' LCD-Default-Helligkeit
Const Lcd_helligkeit_aus = 0                            ' Wert für Display aus


'-------------------------------------------------------------------------------
' Definition von Variablen und Datentypen
'-------------------------------------------------------------------------------

' ----- Temporäre Hilfsvariablen -----
Dim Temp_byte_1 As Byte                                 ' Temporäre Byte Variable 1
Dim Temp_byte_2 As Byte                                 ' Temporäre Byte Variable 2

' ----- Arbeitsvariablen für Spieleablauf -----
Dim Game_hangman_status As Byte                         ' Arbeitsvariable für Hangman-Status

' ----- Variablen für LCD-Display -----
Dim Lcd_kontrastwert As Byte                            ' Arbeitsvariable für Kontrastwert
Dim Lcd_helligkeit As Byte                              ' Arbeitsvariable für Displayhelligkeit
```

```
'-------------------------------------------------------------------------------
' Prototyping
'-------------------------------------------------------------------------------

' ----- LCD und Print -----
Declare Sub Lcd_print_hangingman(byval Value As Byte)       ' Funktion zum schrittweisen Aufbau des
Hanging Man


'-------------------------------------------------------------------------------
' Konfiguration und Basiseinstellungen (Projekt und Testumgebung)
'-------------------------------------------------------------------------------

' --------------------------------- CONFIG -------------------------------------

' ----- Timer -----

' Konfiguration eines Timers für 1 Sekunden Timer-Tick (Scheduler und Alive)
Config Timer1 = Timer , Prescale = 256                      ' Timer 1 verwenden
On Timer1 Sekunden_tick                                     ' Interrupt Routine
Timer1 = Timervorgabe
Enable Timer1                                               ' Interrupt für Sekunden-Tack

' Konfiguration Timer 2 für Hardware-PWM an OC2 (D.7)
Config Timer2 = Pwm , Prescale = 128 , Compare Pwm = Clear Up
Enable TIMER2

' ----- LCD Display -----

' Konfiguration LCD Display
Config Lcdpin = Pin , Db4 = Db4_pin , Db5 = Db5_pin , Db6 = Db6_pin , Db7 = Db7_pin , E = E_pin , Rs
= Rs_pin
Config Lcd = 16 * 2 , Chipset = Dogm162v5                   ' DOG-M Treiber laden
Config Lcdbus = 4                                           ' LCD arbeitet über 4-Bit
Initlcd                                                     ' LCD initialisieren
Waitms 100                                                  ' 100ms nach Init warten
Cursor Off Noblink                                          ' Blinkenden Cursor abschalten

' Definition benutzerdefinierter Zeichen
' LCD-Zeichen linker Sockel des Galgens
Deflcdchar 0 , 16 , 16 , 16 , 16 , 16 , 16 , 30 , 30
' LCD-Zeichen linke obere Ecke des Galgens
Deflcdchar 1 , 15 , 9 , 10 , 12 , 8 , 8 , 8 , 8
' LCD-Zeichen rechter Galgen ohne Männchen
Deflcdchar 2 , 28 , 4 , 32 , 32 , 32 , 32 , 32 , 32
' LCD-Zeichen rechter Galgen mit Kopf
Deflcdchar 3 , 28 , 4 , 4 , 14 , 17 , 17 , 14 , 4
' LCD-Zeichen Bauch
Deflcdchar 4 , 4 , 4 , 4 , 4 , 32 , 32 , 32 , 32
' LCD-Zeichen Bauch mit linkem Beine
Deflcdchar 5 , 4 , 4 , 4 , 4 , 8 , 16 , 16 , 32
' LCD-Zeichen Bauch mit beiden Beinen
Deflcdchar 6 , 4 , 4 , 4 , 4 , 10 , 17 , 17 , 32
' LCD-Zeichen kompletter Männchen-Körper
Deflcdchar 7 , 14 , 21 , 21 , 4 , 10 , 17 , 17 , 32

Cls                                                         ' Clear Screen


' ------------------------------ Port's und Pin's -----------------------------

' ----- LED-Konfigurationen -----
Config Alive_pin = Output
Config Pwrled_pin = Output

' ----- Test: Tasten-Konfiguration ------
Config Key1_pin = Input
Config Key2_pin = Input

' ------------------------------ Variablen und Werte --------------------------

' ----- LED-Konfigurationen -----
Alive = Led_aus                                             ' Alive-LED aus
Pwrled = Led_aus                                            ' LED für Spannungsüberwachung

' ----- LCD-Display -----
Lcd_kontrastwert = Lcd_kontrast_default                     ' Kontrastwert
Lcd_helligkeit = Lcd_helligkeit_default                     ' Displayhelligkleit
```

```
' ----- Spielesteuerung -----
Game_hangman_status = 0


'--------------------------------------------------------------------------------
' Und los gehts, hier noch die Restarbeiten
'--------------------------------------------------------------------------------

' ----- Freigabe aller Interrupts ----
Enable Interrupts                                       ' Damit auch Empfang von Daten über
Buffer

' ---------------------------------- Gosub's ----------------------------------

Gosub Lcd_kontrast_set                                  ' LCD Kontrast einstellen
Gosub Lcd_helligkeit_set                                ' Displayhelligkeit einstellen


' ################################################################################
'
'                             Hauptprogramm ConvCtrl
'
' ################################################################################

' Kleiner LCD-Test zu Beginn
#if Lcd_testmodus
   ' Mit kleiner Schleife gesamtes Display mit * füllen
   For Temp_byte_1=1 to 2 Step 1
      For Temp_byte_2=1 to 16 Step 1
         Locate Temp_byte_1, Temp_byte_2
         Lcd "*"
         waitms 50
      Next Temp_byte_2
   Next Temp_byte_1

   ' Mit kleiner Schleife gesamtes Display leeren
   For Temp_byte_1=2 to 1 Step -1
      For Temp_byte_2=16 to 1 Step -1
         Locate Temp_byte_1, Temp_byte_2
         Lcd " "
         waitms 50
      Next Temp_byte_2
   Next Temp_byte_1

   ' Noch auf beide Zeilen die Positionszahlen ausgeben und gut
   Locate 1 , 1 : Lcd "1234567890123456"
   Locate 2 , 1 : Lcd "1234567890123456"

#endif


' ------------------------------------------------
' ----- Hier ist die Programmhauptschleife -----
' ------------------------------------------------

' In Abhängigkeit der Konstante Traceausgabe schreiben
#if Main_testmodus
   Print "** OK, let's GO **"
#endif

Do                                                      ' Hauptschleife

   ' Test für Kontrasteinstellung
   ' Debounce Key1_pin , 0 , Lcd_inccontrast , Sub            ' Erhöhe den Kontrast
   ' Debounce Key2_pin , 0 , Lcd_deccontrast , Sub            ' Verringere den Kontrast

   ' Test für Helligkeitseinstellung
   ' Debounce Key1_pin , 0 , Lcd_inchelligkeit , Sub          ' Erhöhe den Helligkeit
   ' Debounce Key2_pin , 0 , Lcd_dechelligkeit , Sub          ' Verringere den Helligkeit

   ' Test für Ausschalten / Einschalten Hintergrundbeleuchtung
   ' Debounce Key1_pin , 0 , Lcd_beleuchtung_ein , Sub        ' Beleuchtung ein
   ' Debounce Key2_pin , 0 , Lcd_beleuchtung_aus , Sub        ' Beleuchtung aus

   ' Test für Galgenmännchen
   Debounce Key1_pin , 0 , Lcd_print_hangingman_all , Sub   ' Galgenmännchen zeichnen

Loop                                                    ' Hauptschleife
```

```
'## End Hauptprogramm ######################################################

End


'*****************************************************************************
' Interruptroutinen
'*****************************************************************************

'-----------------------------------------------------------------------------
' Interrupt-Service-Routine (Timer1): Sekunden_tick
' Routine zur Auswertung des Timer Interrupts
'-----------------------------------------------------------------------------
Sekunden_tick:

   ' ----- Programmcode -----

   Timer1 = Timervorgabe                                    ' Timer neu laden
   Toggle Alive                                             ' Alive-LED toggeln lassen

Return
'-- End Sekunden_tick  ---------------------------------------------------


'*****************************************************************************
' Subroutinen
'*****************************************************************************

' ***************
' * LCD-Display *
' ***************

'-----------------------------------------------------------------------------
' LCD - Subroutine: Lcd_print_hangingman
'
' Subroutine schreibt je nach uebergebenem Status den Zustand des Hanging Man
' auf das Display
'    Status 0: Galgen ohne Maennchen
'    Status 1: Galgen mit Kopf
'    Status 2: Galgen mit Kopf und Rumpf
'    Status 3: Galgen mit Kopf, Rumpf und linken Bein
'    Status 4: Galgen mit Kopf, Rumpf, linken und rechten Bein
'    Status 5: Galgen mit kompletten Körper
'
' Parameter:    Value = Step n des Hanging-Man
' Rückgabewert: keine
'
' Globale Variable:
' --
'-----------------------------------------------------------------------------
Sub Lcd_print_hangingman(byval Value As Byte)

   ' ----- Programmcode -----

   ' Galgen zeichnen
   Locate 1 , 1 : Lcd Chr(1)
   Locate 2 , 1 : Lcd Chr(0)

   Select Case Value
      Case 0 :
         Locate 1 , 2 : Lcd Chr(2)
      Case 1 :
         Locate 1 , 2 : Lcd Chr(3)
      Case 2 :
         Locate 1 , 2 : Lcd Chr(3)
         Locate 2 , 2 : Lcd Chr(4)
      Case 3 :
         Locate 1 , 2 : Lcd Chr(3)
         Locate 2 , 2 : Lcd Chr(5)
      Case 4 :
         Locate 1 , 2 : Lcd Chr(3)
         Locate 2 , 2 : Lcd Chr(6)
      Case 5 :
         Locate 1 , 2 : Lcd Chr(3)
         Locate 2 , 2 : Lcd Chr(7)
   End Select

End Sub
'-- End Lcd_Print_hanginman ---------------------------------------------------
```

```
'*******************************************************************************
' GOSubroutinen
'*******************************************************************************

' ***************
' * LCD-Dsiplay *
' ***************

'-------------------------------------------------------------------------------
' LCD - Gosub-Routine: Lcd_contrast_set
' Routine berechnen neue Kontrastwerte und steuert direkt den
' Kontroller des Display an.
' Aufgrund von 6 Bit sind nur Kontrastwerte zwischen 0 und 63 möglich!
'-------------------------------------------------------------------------------
Lcd_kontrast_set:                                    ' Kontrasteinstellung Display

   ' ----- Programmcode -----

   ' Verarbeitung des Kontrastwertes für High-Byte und Low-Byte
   Temp_byte_1 = Lcd_kontrastwert And &B00001111
   Temp_byte_1 = Temp_byte_1 + &B01110000

   Temp_byte_2 = Lcd_kontrastwert
   Shift Temp_byte_2 , Right , 4
   Temp_byte_2 = Temp_byte_2 And &B00000011
   Temp_byte_2 = Temp_byte_2 + &B01010100

   ' Instruction Table 1 einstellen [0,1]
   _temp1 = &B00101001
   !rCall _Lcd_control

   ' Tempvar_1 = &B0111xxxx für Kontrast Set Instruction Table 1 - Low Byte
   _temp1 = Temp_byte_1
   !rCall _Lcd_control

   ' Temovar_2 = &B010101xx für Kontrast Set Instruction Table 1 - High Byte
   _temp1 = Temp_byte_2
   !rCall _Lcd_control

   ' Zurückschalten auf Instruction Table 0 [0,0]
   _temp1 = &B00101000
   !rCall _Lcd_control

Return
'-- End Lcd_kontrast_set -------------------------------------------------------


'-------------------------------------------------------------------------------
' LCD - Gosub-Routine: Lcd_IncContrast
' Routine erhöht bei Einsprung den Kontrastwert um 1
'-------------------------------------------------------------------------------
Lcd_inccontrast:

   ' ----- Programmcode -----

   ' Aufgrund von 6 Bit sind nur Kontrastwerte zwischen 0 und 63 möglich!
   If Lcd_kontrastwert < 63 Then
      Incr Lcd_kontrastwert
   end if
   Gosub Lcd_kontrast_set
   Print "Kontrast: " ; Lcd_kontrastwert

   Return
'-- End Lcd_IncContrast ----------------------------------------------------


'-------------------------------------------------------------------------------
' LCD - Gosub-Routine: Lcd_DecContrast
' Routine verringert bei Einsprung den Kontrastwert um 1
'-------------------------------------------------------------------------------
Lcd_deccontrast:

   ' ----- Programmcode -----

   If Lcd_kontrastwert > 1 Then
      Decr Lcd_kontrastwert
   End If
   Gosub Lcd_kontrast_set
```

```
   Print "Kontrast: " ; Lcd_kontrastwert

Return
'-- End Lcd_DecContrast ------------------------------------------------------


'--------------------------------------------------------------------------------
' LCD - Gosub-Routine: Lcd_helligkeit_set
'
' Routine setzt den Helligkeitswert aus der globalen Variable Lcd_helligkeit in
' das Controllregister
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' Lcd_helligkeit = Helligkeitswert für PWM
'--------------------------------------------------------------------------------
Lcd_helligkeit_set:

   ' ----- Programmcode -----

   ' Variable in Timer-Count-Register laden
   Ocr2 = Lcd_helligkeit

Return
'-- End Ir_set_helligkeit -----------------------------------------------------


'--------------------------------------------------------------------------------
' LCD - Gosub-Routine: Lcd_IncHelligkeit
' Routine erhöht bei Einsprung die Helligkeit um 1
'--------------------------------------------------------------------------------
Lcd_inchelligkeit:

   ' ----- Programmcode -----

   If Lcd_helligkeit < 250 Then
      Lcd_helligkeit = Lcd_helligkeit + 10
   end if
   Gosub Lcd_helligkeit_set
   #if Lcd_testmodus
      Print "Helligkeit: " ; Lcd_helligkeit
   #endif

Return
'-- End Lcd_IncHelligkeit -------------------------------------------------------


'--------------------------------------------------------------------------------
' LCD - Gosub-Routine: Lcd_DecHelligkeit
' Routine verringert bei Einsprung die Helligkeit um 1
'--------------------------------------------------------------------------------
Lcd_dechelligkeit:

   ' ----- Programmcode -----

   If Lcd_helligkeit > 10 Then
      Lcd_helligkeit = Lcd_helligkeit - 10
   End If
   Gosub Lcd_helligkeit_set
   #if Lcd_testmodus
      Print "Helligkeit: " ; Lcd_helligkeit
   #endif

Return
'-- End Lcd_Dechelligkeit --------------------------------------------------------


'--------------------------------------------------------------------------------
' LCD - Gosub-Routine: Lcd_beleuchtung_aus
'
' Routine schaltet die LCD-Hintergrundbeleuchtung aus
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' --
'--------------------------------------------------------------------------------
```

```
Lcd_beleuchtung_aus:

   ' ----- Programmcode -----

   Lcd_helligkeit = Lcd_helligkeit_aus
   Gosub Lcd_helligkeit_set

   #if Lcd_testmodus
      Print "Beleuchtung aus"
   #endif

Return
'-- End Lcd_beleuchtung_aus ---------------------------------------------------


'--------------------------------------------------------------------------------
' LCD - Gosub-Routine: Lcd_beleuchtung_ein
'
' Routine schaltet die LCD-Hintergrundbeleuchtung ein
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' --
'--------------------------------------------------------------------------------
Lcd_beleuchtung_ein:

   ' ----- Programmcode -----

   Lcd_helligkeit = Lcd_helligkeit_default
   Gosub Lcd_helligkeit_set

   #if Lcd_testmodus
      Print "Beleuchtung ein"
   #endif

Return
'-- End Lcd_beleuchtung_ein ----------------------------------------------------


'--------------------------------------------------------------------------------
' LCD - Gosub-Routine: Lcd_Print_hangingman
'
' Routine zeichnet Galgenmaennchen auf dem Display
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' --
'--------------------------------------------------------------------------------
Lcd_print_hangingman_all:

   ' ----- Programmcode -----

   Cls                                                  ' Bildschirm löschen

   Call Lcd_print_hangingman(game_hangman_status )      ' Maennchen gemaess Status zeichnen
   Incr Game_hangman_status                             ' Statuscounter erhoehen

   ' Pruefen ob Statuscounter noch valide und ggf. zuruecksetzen
   If Game_hangman_status > 5 Then
      Game_hangman_status = 0
   End If

Return
'-- End Lcd_beleuchtung_ein ----------------------------------------------------



'--------------------------------------------------------------------------------
' Devices schließend und ggf. "Terminate Programm execution"
'--------------------------------------------------------------------------------

' System halt
End                                                  'end program

'--------------------------------------------------------------------------------
' Definition von globalen Konstantenfeldern
```

```
'-------------------------------------------------------------------------------


' -----------------------------------------------------------------------------


'#################################### END ######################################


'################################### Historie ##################################
' 03.05.2014 : Version x.x
'              Beginn der Implementierungen für Hanging Man
' 03.05.2014 : Versions 1.0
'              Fertigstellung Inbetriebnahme LCD-Display und wichtigster
'              Funktionen
'###############################################################################
```

Software 1: Code zur Ansteuerung des LCD-Displays

9.3.5 Prototyp LCD-Display-Ansteuerung

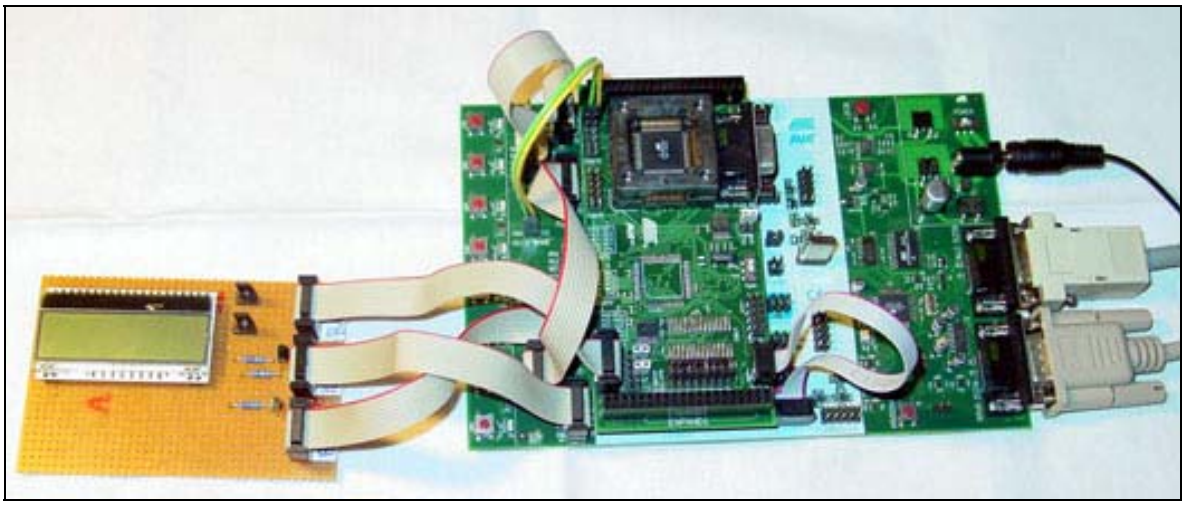


Abbildung 11: Prototyp LCD-Display-Ansteuerung mit STK

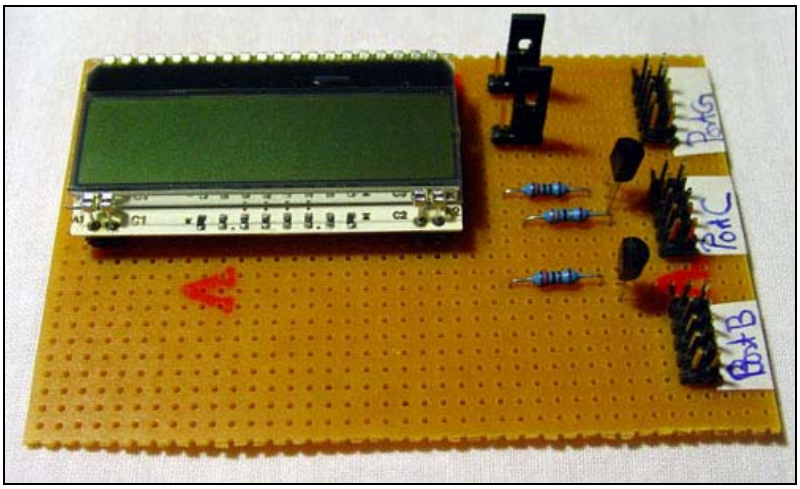


Abbildung 12: Prototyp LCD-Display-Ansteuerung PCB

9.4 Drück-Dreh-Geber (DDS) / Inkrementaldrehgeber

Allgemeines:

Um im Spiel die Buchstaben auszuwählen wird ein inkrementeller Drehgeber mit Tasterfunktion verwendet. Drehen nach rechts wechselt im Spiel die Buchstaben in Richtung A bis Z, drehen nach links von Z bis A. Betätigung des Tasters wählt den entsprechenden Buchstaben aus.

Zum Einsatz kommt dabei ein Drehimpulsgeber des folgenden Typs:

ALPS STEC11B Drehimpulsg., 20/20, vert., MT

Typ:	Drehimpulsegeber
Aufbau:	20 Impulse / 20 Rastungen
Nennspannung:	5 VDC
Nennlast:	0,01 W
Hersteller:	ALPS
Verpackungsgewicht:	0,005 kg

Reichelt Bestellnummer: *STEC11B13*

Abbildung 13: Drehimpulsgeber

Technische Details:

Wie der folgenden Darstellung zu entnehmen ist kann über die Abfolge der fallenden bzw. steigenden Flanken die Drehrichtung erkannt werden.

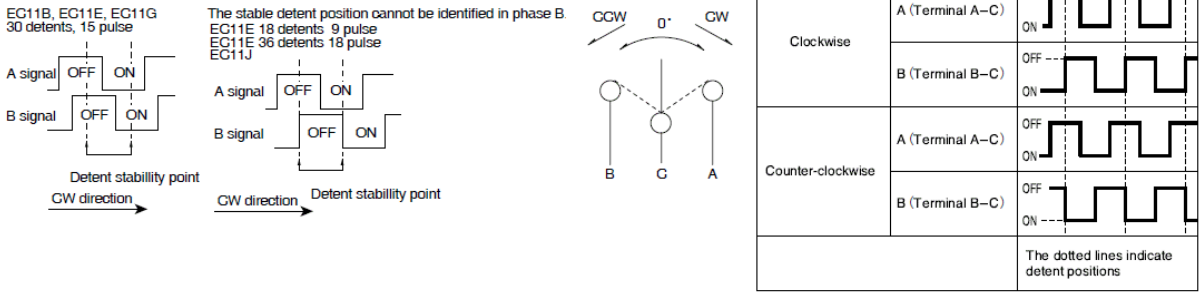


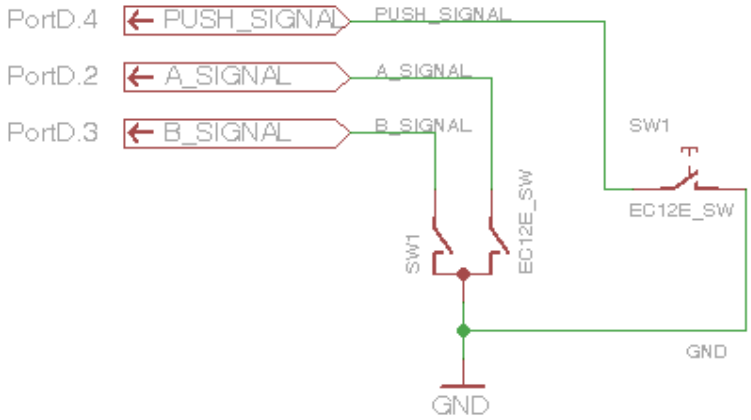
Abbildung 14: Drehimpulsgeber Beschaltung und Auswertung

Die Umsetzung im Projekt erfolgt über einen Interrupt. Hierbei wird Terminal A mittels Interrupt erkannt. In der Interrupt-Service-Routine wird Terminal B ausgelesen. Über den Pegel von B wird die Drehrichtung ermittelt.

Auf eine externe Beschaltung mit PullUp- bzw. PulLDown-Widerstände wird verzichtet. Terminal C wird mit GND beschaltet. Für die beiden GPIO-Pins an denen Termeinal A und B am Controller aufgeschaltet werden, werden die Controler internen PullUp-Widerstände aktiviert. Somit entstehen wir im obigen Diagramm dargestellt beim Betätigen der Schalter fallende Signalflanken.

9.4.1 Beschaltung des Drehimpulsgebers

Die Beschaltung des Drehimpulsgebers erfolgt wie im folgenden Schaltbild dargestellt:



Schaltbild 8: Beschaltung Drehimpulsgeber im Projekt Hanging Man

Bauteile:

Stückliste: Beschaltung Drehimpulsgeber		
Sonstiges		
SW1	Drehimpulsgeber / Encoder	

Tabelle 36: Stückliste Beschaltung Drehimpulsgeber

Ressourcenzuordnung zum ATmega8L:

Nummer	*Schaltbild*		*Ressource ATmega8L*		
1	Terminal A_SIGNAL	[A_SIGNAL]	PortD.2	[PD2]	(GPIO bzw. INT0)
2	Terminal B_SIGNAL	[B_SIGNAL]	PortD.4	[PD4]	(GPIO)
3	Taster	[PUSH_SIGNAL]	PortD.3	[PD3]	(GPIO bzw. INT1)

Tabelle 37: Ressourcenzuordnung ATmega8L für Drehimpulsgeber

Anmerkung:
Die Beschaltung von B_SIGNAL und Taster wurde wärend des Projekt getauscht, damit der Taster auch über Interrupt 1 (INT1) abgefragt werden kann. Die Beschreibungen in diesem Kapitel beziehen sich aber noch auf den Originalzustand.

9.4.2 BASCOM Beispielcode für den Drehimpulsgeber inkl. Interruptsteuerung

```
'###############################################################################
' Drehgeber.BAS                                              Stand 04.05.2014
' -------------                                              (C) Markus Fulde
'
' Testprogramm zur Inbetriebnahme des Drück-Dreh-Gebers (Inkrementaldrehgeber)
' Die Inbetriebnahme erfolgt mit dem Eva-Board STK500 + STK501
' Die für den DDS notwendigen Routinen sind als solche gekennzeichnet
' und bereits zur späteren Verwendung ausgelagert.
'###############################################################################

'-------------------------------------------------------------------------------
' Compilerinstruktionen und Compilerdirektiven
'-------------------------------------------------------------------------------
$regfile = "m8def.dat"                                    ' Definitionsdatei für ATmega8L laden
$crystal = 8000000                                        ' Quarzfrequenz für 16 MHz festlegen
```

```
$baud = 19200                                          ' Baudrate für RS232 Traceausgabe defi-
nieren


'-----------------------------------------------------------------------------
' Allgemeine Zusatzinformatonen zu Programmbeginn
'-----------------------------------------------------------------------------


'-----------------------------------------------------------------------------
' Definition von Ressourcen
'-----------------------------------------------------------------------------

' ----- LED's -----
Alive_pin Alias PinB.0                                 ' GPIO für Alive-LED (für DDR oder In-
put)
Alive Alias PortB.0                                    ' GPIO für Alive-LED (für Output oder
Pullup)

' ----- Test: Tasten zu Testzwecken -----
A_signal_pin Alias Pind.2                              ' GPIO für DDS Signal A (für DDR oder
Input)
A_signal Alias Portd.2                                 ' GPIO für DDS Signal A (für Output oder
Pullup)

B_signal_pin Alias Pind.3                              ' GPIO für DDS Signal B (für DDR oder
Input)
B_signal Alias Portd.3                                 ' GPIO für DDS Signal B (für Output oder
Pullup)

Push_signal_pin Alias Pind.4                           ' GPIO für DDS Push (für DDR oder Input)
Push_signal Alias Portd.4                              ' GPIO für DDS Push (für Output oder
Pullup)


'-----------------------------------------------------------------------------
' Definition von Konstaten
'-----------------------------------------------------------------------------

' ----- Für Testumgebung bzw. Traceausgaben -----
Const Main_testmodus = 1                               ' Flag für Testmodus Allgemeinsystem


' ----- Allgemeine Systemkonstanten -----

' Tatsächliches Allgemeines
' Const Led_aus = 0
' Const Led_ein = 1

Const Led_aus = 1 ' Achtung !! bei STK500 ist Logik gedreht!!
Const Led_ein = 0 ' Achtung !! bei STK500 ist Logik gedreht!!

Const False = 0
Const True = 1

Const Pullup_aus = 0
Const Pullup_ein = 1

' Zeitvorgabe für Sekunden-Timer
Const Timervorgabe = 34286                             ' Timer von 1 Sekunden (SekundenTick)


' ----- DDS -----
Const Dds_none = 0                                     ' Keine Betätigung des DDS
Const Dds_links = 1                                    ' DDS wurde nach links gedreht
Const Dds_rechts = 2                                   ' DDS wurde nach rechts gedreht


'-----------------------------------------------------------------------------
' Definition von Variablen und Datentypen
'-----------------------------------------------------------------------------

' ----- Temporäre Hilfsvariablen -----
Dim Temp_byte_1 As Byte                                ' Temporäre Byte Variable 1
Dim Temp_byte_2 As Byte                                ' Temporäre Byte Variable 2

' ----- DDS -----
Dim Dds_flag As Byte                                   ' Arbeitsvariable zur Behandlung des DDS
```

```
'-------------------------------------------------------------------------------
' Prototyping
'-------------------------------------------------------------------------------


'-------------------------------------------------------------------------------
' Konfiguration und Basiseinstellungen (Projekt und Testumgebung)
'-------------------------------------------------------------------------------

' ---------------------------------- CONFIG -----------------------------------

' ----- Timer -----

' Konfiguration eines Timers für 1 Sekunden Timer-Tick (Scheduler und Alive)
Config Timer1 = Timer , Prescale = 256                      ' Timer 1 verwenden
On Timer1 Sekunden_tick                                     ' Interrupt Routine
Timer1 = Timervorgabe
Enable Timer1                                               ' Interrupt für Sekunden-Tack


' ----- DDS -----

' Int 0 für Signal A so konfigurieren dass Interrupt bei Pegeländerung angesprungen wird
Config Int0 = Falling
On Int0 Dds_interrupt                                       ' Interrupt-Routine
Enable Int0                                                 ' Interrupt freigeben

' ------------------------------ Port's und Pin's -----------------------------

' ----- LED-Konfigurationen -----
Config Alive_pin = Output


' ----- DDS ------
Config A_signal_pin = Input
Config B_signal_pin = Input
Config Push_signal_pin = Input


' ------------------------------ Variablen und Werte --------------------------

' ----- LED - Konfigurationen -----
Alive = Led_aus                                             ' Alive-LED aus

' ----- DDS - Konfiguration -----

' Interne Pull-Ups für die Schalter aktivieren
A_signal = Pullup_ein
B_signal = Pullup_ein
Push_signal = Pullup_ein

' Arbeitsvariable vorinitialisieren
Dds_flag = Dds_none


'-------------------------------------------------------------------------------
' Und los gehts, hier noch die Restarbeiten
'-------------------------------------------------------------------------------

' ----- Freigabe aller Interrupts ----
Enable Interrupts                                           ' Damit auch Empfang von Daten über
Buffer

' ---------------------------------- Gosub's ----------------------------------


' ###############################################################################
'
'                              Hauptprogramm ConvCtrl
'
' ###############################################################################


' ------------------------------------------------
' ----- Hier ist die Programmhauptschleife -----
' ------------------------------------------------

' In Abhängigkeit der Konstante Traceausgabe schreiben
```

```
#if Main_testmodus
   Print "** OK, let's GO **"
#endif

Do                                                          ' Hauptschleife

   ' Testroutinen um zu prüfen ob die Schalter überhaupt schalten
   Debounce Push_signal_pin , 0 , Dds_taste_erkannt , Sub
   ' Debounce A_signal_pin , 0 , Dds_taste_erkannt , Sub
   ' Debounce B_signal_pin , 0 , Dds_b_taste_erkannt , Sub

   ' ----- DDS auswerten -----
   Select Case Dds_flag
      Case Dds_links : Gosub Dds_links_verarbeiten
      Case Dds_rechts : Gosub Dds_rechts_verarbeiten
   End Select

Loop                                                        ' Hauptschleife

'## End Hauptprogramm ###############################################################

End


'*****************************************************************************
' Interruptroutinen
'*****************************************************************************

'-----------------------------------------------------------------------------
' Interrupt-Service-Routine (Timer1): Sekunden_tick
' Routine zur Auswertung des Timer Interrupts
'-----------------------------------------------------------------------------
Sekunden_tick:

   ' ----- Programmcode -----

   Timer1 = Timervorgabe                                    ' Timer neu laden
   Toggle Alive                                             ' Alive-LED toggeln lassen

Return
'-- End Sekunden_tick  ---------------------------------------------------


'-----------------------------------------------------------------------------
' Interrupt-Service-Routine (External Interrupt 0): Dds_interrupt
' Routine zur Auswertung des INT0 Interrupts für DDS-Bewegung
'-----------------------------------------------------------------------------
Dds_interrupt:

   ' ----- Programmcode -----
   If A_signal_pin = B_signal_pin Then

      ' Links herum
      Dds_flag = Dds_links

   Else

      ' Rechts herum
      Dds_flag = Dds_rechts

   End If

Return
'-- End Dds_interrupt -------------------------------------------------------


'*****************************************************************************
' Subroutinen
'*****************************************************************************


'*****************************************************************************
' GOSubroutinen
'*****************************************************************************

'-----------------------------------------------------------------------------
' DDS - Gosub-Routine: Dds_Taste_erkannt
' Routine gibt nur Traceausgabe aus, dass Taste erkannt wurde
'-----------------------------------------------------------------------------
```

```
Dds_taste_erkannt:

   ' ----- Programmcode -----

   Print "DDS gedueckt"

   Return
'-- End Dds_Taste_erkannt ------------------------------------------------------


'-------------------------------------------------------------------------------
' DDS - Gosub-Routine: Dds_links_verarbeiten
' Routine verarbeitet das Drehen des DDS nach links
'-------------------------------------------------------------------------------
Dds_links_verarbeiten:

   ' ----- Programmcode -----

   Print "DDS links"
   Dds_flag = Dds_none

Return
'-- End Dds_links_verarbeiten ---------------------------------------------


'-------------------------------------------------------------------------------
' DDS - Gosub-Routine: Dds_TDds_rechts_verarbeiten
' Routine verarbeitet das Drehen des DDS nach links
'-------------------------------------------------------------------------------
Dds_rechts_verarbeiten:

   ' ----- Programmcode -----

   Print "DDS rechts"
   Dds_flag = Dds_none

Return
'-- End Dds_rechts_verarbeiten --------------------------------------------


'-------------------------------------------------------------------------------
' Devices schließend und ggf. "Terminate Programm execution"
'-------------------------------------------------------------------------------

' System halt
End                                                    'end program

'-------------------------------------------------------------------------------
' Definition von globalen Konstantenfeldern
'-------------------------------------------------------------------------------


' ------------------------------------------------------------------------


'###################################### END ####################################


'################################### Historie ##################################
' 04.05.2014 : Version x.x
'              Beginn der Implementierungen für Hanging Man
' 22.05.2014 : Versions 1.0
'              Fertigstellung Inbetriebnahme und Test Drehgeber
'###############################################################################
```

Software 2: Code zur Ansteuerung des Drehimpulsgebers

9.4.3 Prototyp des Drehimpulsgebers

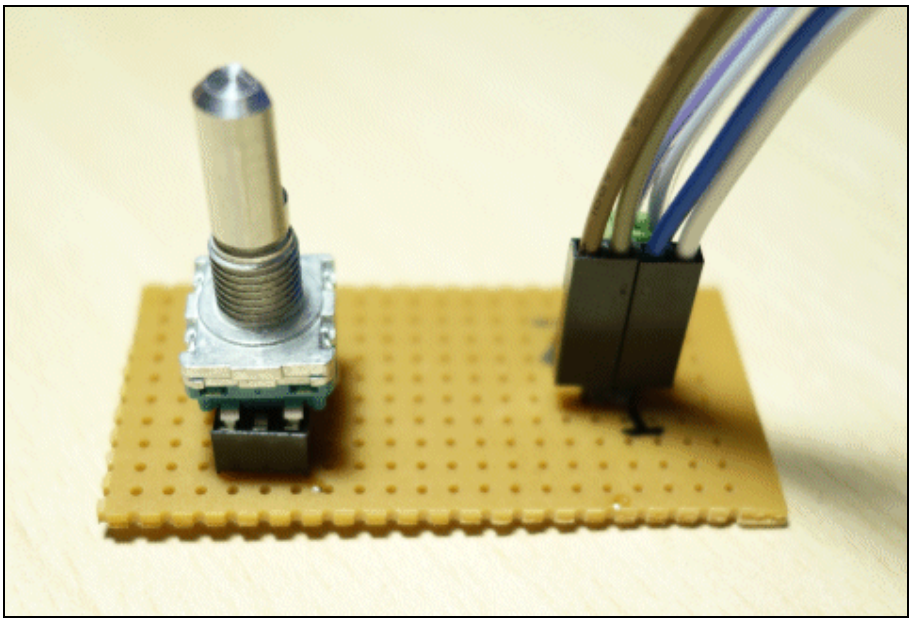


Abbildung 15: Prototyp Drehimpulsgeber

9.5 Batteriespannungsüberwachung

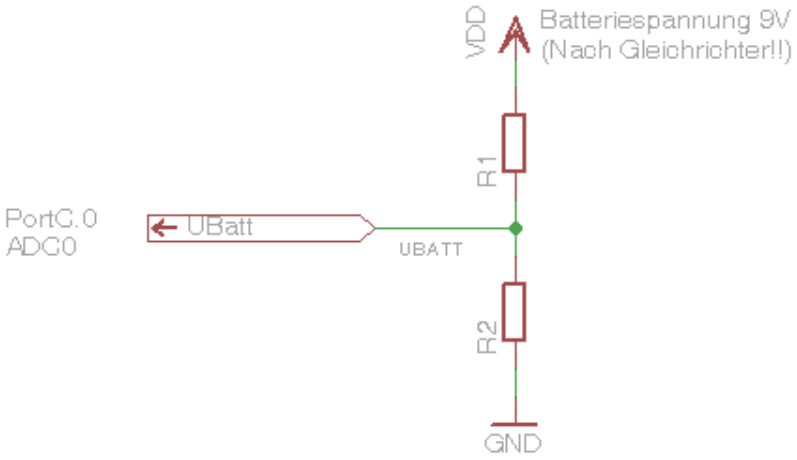
Damit rechtzeitig das Austauschen der batterie angezeigt werden kann wird mittels internem ADC-Wandler eine Batteriespannungsüberwachung implementiert.

Hierbei wird über einen 50/50-Spannungsteiler direkt die Batteriespannung der 9V-Block-Batterie abgegriffen und auf den ADC0-Eingang des ATmega8 geleitet.

Fällt die Spannung unter eine in der SW eingestellte Schwelle so wird dies bei Systemstart erkannt und eine rote Low-Current-LED eingeschaltet die optisch das Wechseln der Batterie anzeigt.

9.5.1 Beschaltung des ADC zur Batteriespannungsüberwachung

Die Beschaltung des ADC erfolgt wie im folgenden Schaltbild dargestellt:



Schaltbild 9: Beschaltung ADC0 im Projekt Hanging Man

Bauteile:

Stückliste: Beschaltung ADC für Batteriespannungsüberwachung		
Widerstände		
R1, R2	Metallschichtwiderstände 1,00 MΩ	

Tabelle 38: Stückliste Beschaltung ADC für Batteriespannungsüberwachung

Ressourcenzuordnung zum ATmega8L:

Nummer	*Schaltbild*	*Ressource ATmega8L*
1	UBatt [UBATT]	PortC.0 [PC0] (ADC0)

Tabelle 39: Ressourcenzuordnung ATmega8L für Batteriespannungsüberwachung

9.5.2 BASCOM Beispielcode für den Drehimpulsgeber inkl. Interruptsteuerung

```
'#################################################################################
' ADC-Batterieueberwachung.BAS                                    Stand 24.05.2014
' ----------------------------                                    (C) Markus Fulde
'
' Testprogramm zur Inbetriebnahme des ADC's zur Batteriespannungsueberwachung
' Die Inbetriebnahme erfolgt mit dem Eva-Board STK500 + STK501
' Die für den ADC notwendigen Routinen sind als solche gekennzeichnet und
' bereits zur späteren Verwendung ausgelagert.
'#################################################################################

'--------------------------------------------------------------------------------
' Compilerinstruktionen und Compilerdirektiven
'--------------------------------------------------------------------------------
$regfile = "m8def.dat"                                  ' Definitionsdatei für ATmega8 laden
$crystal = 8000000                                      ' Quarzfrequenz für 16 MHz festlegen

$baud = 19200                                           ' Baudrate für RS232 Traceausgabe defi-
nieren


'--------------------------------------------------------------------------------
' Allgemeine Zusatzinformatonen zu Programmbeginn
'--------------------------------------------------------------------------------


'--------------------------------------------------------------------------------
' Definition von Ressourcen
'--------------------------------------------------------------------------------

' ----- LED's -----
Alive_pin Alias PinB.0                                  ' GPIO für Alive-LED (für DDR oder In-
put)
Alive Alias PortB.0                                     ' GPIO für Alive-LED (für Output oder
Pullup)

Pwrled_pin Alias PINB.1                                 ' GPIO für Power-LED (für DDR oder In-
put)
Pwrled Alias PORTB.1                                    ' GPIO für Power-LED (für Output oder
Pullup)


'--------------------------------------------------------------------------------
' Definition von Konstaten
'--------------------------------------------------------------------------------

' ----- Für Testumgebung bzw. Traceausgaben -----
Const Main_testmodus = 1                                ' Flag für Testmodus Allgemeinsystem
Const Adc_testmodus = 1                                 ' Flag für Testmodus des ADC


' ----- Allgemeine Systemkonstanten -----
```

```
' Tatsächliches Allgemeines
' Const Led_aus = 0
' Const Led_ein = 1

Const Led_aus = 1 ' Achtung !! bei STK500 ist Logik gedreht!!
Const Led_ein = 0 ' Achtung !! bei STK500 ist Logik gedreht!!

Const False = 0
Const True = 1

Const Pullup_aus = 0
Const Pullup_ein = 1

' Zeitvorgabe für Sekunden-Timer
Const Timervorgabe = 34286                                  ' Timer von 1 Sekunden (SekundenTick)

' ----- ADC: Batterieüberwachung -----
Const Ubatt_grenzwert = 700                                 ' Grenzwert für Batterieüberwachung


'-------------------------------------------------------------------------------
' Definition von Variablen und Datentypen
'-------------------------------------------------------------------------------

' ----- Temporäre Hilfsvariablen -----
Dim Temp_byte_1 As Byte                                     ' Temporäre Byte Variable 1
Dim Temp_byte_2 As Byte                                     ' Temporäre Byte Variable 2

Dim Temp_word_1 As Word                                     ' Temporäre Word Arbeitsvariable

' ----- ADC: Batteriespannungsueberwachung -----
Dim Adc_channel As Byte                                     ' Arbeitsvariable für ADC-Kanalauswahl


'-------------------------------------------------------------------------------
' Prototyping
'-------------------------------------------------------------------------------


'-------------------------------------------------------------------------------
' Konfiguration und Basiseinstellungen (Projekt und Testumgebung)
'-------------------------------------------------------------------------------

' --------------------------------- CONFIG -----------------------------------

' ----- Timer -----

' Konfiguration eines Timers für 1 Sekunden Timer-Tick (Scheduler und Alive)
Config Timer1 = Timer , Prescale = 256                      ' Timer 1 verwenden
On Timer1 Sekunden_tick                                     ' Interrupt Routine
Timer1 = Timervorgabe
Enable Timer1                                               ' Interrupt für Sekunden-Tack


' ------------------------------ Port's und Pin's ----------------------------

' ----- LED-Konfigurationen -----
Config Alive_pin = Output
Config Pwrled_pin = Output

' ----- ADC: Batterieueberwachung -----
' Konfiguration ADC Single-Mode und automatische Prescaler Setting
' Der Single-Mode wir bei BASCOM in Verbindung mit der Funktion GETADC() verwendet
' Der Prescaler teilt den internen Tackt durch 2, 4, 8,16,32,64 or 128 da der ADC
' einen Takt zwischen 50-200 kHz benötigt.
' Das AUTO Feature von BASCOM, setzt automatisch die höchste mögliche Taktrate
Config Adc = Single , Prescaler = Auto , Reference = Avcc
Start Adc


' ------------------------------ Variablen und Werte -------------------------

' ----- LED-Konfigurationen -----
Alive = Led_aus                                             ' Alive-LED aus
Pwrled = Led_aus                                            ' LED für Spannungsüberwachung

' ----- ADC_ Batteriespannungsueberwachung -----
Adc_channel = 0                                             ' Spannungsteiler zur
Batterispannungsmessung hängt an ADC0
```

```
'-------------------------------------------------------------------------------
' Und los gehts, hier noch die Restarbeiten
'-------------------------------------------------------------------------------

' ----- Freigabe aller Interrupts ----
Enable Interrupts                                       ' Damit auch Empfang von Daten über
Buffer


' --------------------------------- Gosub's ---------------------------------


' ###############################################################################
'
'                             Hauptprogramm ConvCtrl
'
' ###############################################################################


' ---------------------------------------------
' ----- Hier ist die Programmhauptschleife -----
' ---------------------------------------------

' In Abhängigkeit der Konstante Traceausgabe schreiben
#if Main_testmodus
   Print "** OK, let's GO **"
#endif

Do                                                      ' Hauptschleife

   Temp_word_1 = Getadc(adc_channel)

   ' Prüfen auf Unterspannung
   If Temp_word_1 < Ubatt_grenzwert Then

      ' Wenn Unterspannung erkannt dann LED dauerhaft einschalten
      Pwrled = Led_ein

   End If

   Print "Channel " ; Adc_channel ; " value " ; Temp_word_1
   Wait 2

Loop                                                    ' Hauptschleife

'## End Hauptprogramm ##########################################################

End


'*******************************************************************************
' Interruptroutinen
'*******************************************************************************

'-------------------------------------------------------------------------------
' Interrupt-Service-Routine (Timer1): Sekunden_tick
' Routine zur Auswertung des Timer Interrupts
'-------------------------------------------------------------------------------
Sekunden_tick:

   ' ----- Programmcode -----

   Timer1 = Timervorgabe                                ' Timer neu laden
   Toggle Alive                                         ' Alive-LED toggeln lassen

Return
'-- End Sekunden_tick  ------------------------------------------------------


'*******************************************************************************
' Subroutinen
'*******************************************************************************


'*******************************************************************************
' GOSubroutinen
'*******************************************************************************
```

```
'-------------------------------------------------------------------------------
' Devices schließend und ggf. "Terminate Programm execution"
'-------------------------------------------------------------------------------

' System halt
End                                                     'end program

'-------------------------------------------------------------------------------
' Definition von globalen Konstantenfeldern
'-------------------------------------------------------------------------------


' ------------------------------------------------------------------------


'#################################### END ##################################


'################################### Historie ##############################
' 24.05.2014 : Version x.x
'              Beginn der Implementierungen für Hanging Man
' 25.05.2014 : Versions 1.0
'              Fertigstellung Inbetriebnahme Batteriespannungsueberwachung
'###########################################################################
```

Software 3: Code zur Ansteuerung des ADC Batteriespannungsüberwachung

9.6 Sound

Die Soundausgabe im Projekt HangingMan erfolgt mittels Piezo-Summer.

Zum Einsatz kommt der Piezo-Summer CPM 121 von Reichelt.
Reichelt Bestellnummer: *SUMMER CPM 121*

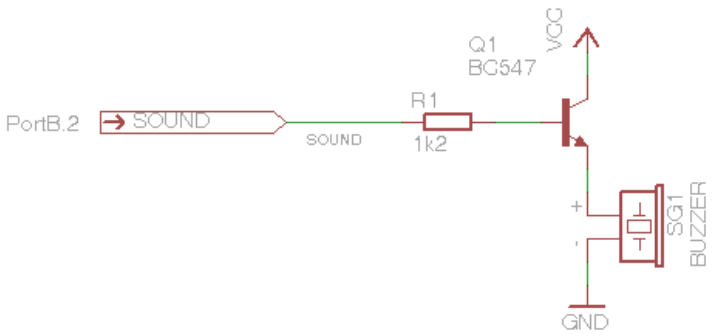
Der Piezo-Signalgeber ist für Printmontage geeignet und hat eine niedrige Stromaufnahme.

Typ	Summer
Ausführung	Piezo
Spannungsversorgung	3 - 15 V DC
Frequenz	4,1 kHz
Rastermaß	7,6 mm
Ø	13 mm
Geräuschpegel	83 dB
Bestellnummer:	*SUMMER CPM 121*

Tabelle 40: Piezo-Schallgeber

Der Piezo wird über eine Transistorstufe vom Controller aus angesteuert.

Beschaltung:



Schaltbild 10: Soundgeber

Bauteile:

Stückliste: Soundgeber			
Sonstiges		*Halbleiter*	
SG1	Piezo-Summer CPM 121	Q1	NPN Transistor BC 547B
Widerstand			
R1	Metallschichtwiderstand 1,5 kΩ		

Tabelle 41: Bauteile für Soundgeber

Ressourcenzuordnung zum ATmega8L:

Nummer	*Schaltbild*		*Ressource ATmega8*		
1	SOUND	SOUND	PortB.2	[BS2]	(GPIO Output)

Tabelle 42: Ressourcenzuordnung für 5V Soundgeber

9.6.1 BASCOM Beispielcode Zur Soundausgabe

```
'###############################################################################
' PiezoSound.BAS                                          Stand 27.05.2014
' --------------                                          (C) Markus Fulde
'
' Testprogramm zur Inbetriebnahme des Piezo Summers
' Die Inbetriebnahme erfolgt mit dem Eva-Board STK500 + STK501
' Die für den Summer notwendigen Routinen sind als solche gekennzeichnet und
' bereits zur späteren Verwendung ausgelagert.
'###############################################################################

'-------------------------------------------------------------------------------
' Compilerinstruktionen und Compilerdirektiven
'-------------------------------------------------------------------------------
$regfile = "m8def.dat"                                  ' Definitionsdatei für ATmega8 laden
$crystal = 8000000                                      ' Quarzfrequenz für 16 MHz festlegen

$baud = 19200                                           ' Baudrate für RS232 Traceausgabe defi-
nieren


'-------------------------------------------------------------------------------
' Allgemeine Zusatzinformatonen zu Programmbeginn
'-------------------------------------------------------------------------------


'-------------------------------------------------------------------------------
' Definition von Ressourcen
'-------------------------------------------------------------------------------
```

```
' ----- LED's -----
Alive_pin Alias PinB.0                                  ' GPIO für Alive-LED (für DDR oder In-
put)
Alive Alias PortB.0                                     ' GPIO für Alive-LED (für Output oder
Pullup)

Pwrled_pin Alias PINB.1                                 ' GPIO für Power-LED (für DDR oder In-
put)
Pwrled Alias PORTB.1                                    ' GPIO für Power-LED (für Output oder
Pullup)

Piezosound_pin Alias Pinb.2                             ' GPIO für Soundausgabesteuerung (Tran-
sistorstufe)
Piezosound Alias Portb.2                                ' GPIO für Sound (für Output oder
Pullup)


'-------------------------------------------------------------------------------
' Definition von Konstaten
'-------------------------------------------------------------------------------

' ----- Für Testumgebung bzw. Traceausgaben -----
Const Main_testmodus = 1                                ' Flag für Testmodus Allgemeinsystem


' ----- Allgemeine Systemkonstanten -----

' Tatsächliches Allgemeines
' Const Led_aus = 0
' Const Led_ein = 1

Const Led_aus = 1 ' Achtung !! bei STK500 ist Logik gedreht!!
Const Led_ein = 0 ' Achtung !! bei STK500 ist Logik gedreht!!

Const False = 0
Const True = 1

Const Pullup_aus = 0
Const Pullup_ein = 1

' Zeitvorgabe für Sekunden-Timer
Const Timervorgabe = 34286                              ' Timer von 1 Sekunden (SekundenTick)

' ----- Sound -----
Const Sound_ein = 1
Const Sound_aus = 0


'-------------------------------------------------------------------------------
' Definition von Variablen und Datentypen
'-------------------------------------------------------------------------------

' ----- Temporäre Hilfsvariablen -----
Dim Temp_byte_1 As Byte                                 ' Temporäre Byte Variable 1
Dim Temp_byte_2 As Byte                                 ' Temporäre Byte Variable 2

Dim Temp_word_1 As Word                                 ' Temporäre Word Arbeitsvariable 1
Dim Temp_word_2 As Word                                 ' Temporäre Word Arbeitsvariable 2


'-------------------------------------------------------------------------------
' Prototyping
'-------------------------------------------------------------------------------


'-------------------------------------------------------------------------------
' Konfiguration und Basiseinstellungen (Projekt und Testumgebung)
'-------------------------------------------------------------------------------

' --------------------------------- CONFIG ---------------------------------

' ----- Timer -----

' Konfiguration eines Timers für 1 Sekunden Timer-Tick (Scheduler und Alive)
Config Timer1 = Timer , Prescale = 256                  ' Timer 1 verwenden
On Timer1 Sekunden_tick                                 ' Interrupt Routine
Timer1 = Timervorgabe
Enable Timer1                                           ' Interrupt für Sekunden-Tack
```

```
' ------------------------------ Port's und Pin's ---------------------------

' ----- LED-Konfigurationen -----
Config Alive_pin = Output                                   ' GPIO-Pin auf Ausgang schalten
Config Pwrled_pin = Output                                  ' GPIO-Pin auf Ausgang schalten

' ----- Sound ------
Config Piezosound_pin = Output                              ' GPIO-Pin auf Ausgang schalten


' ------------------------------ Variablen und Werte -------------------------

' ----- LED-Konfigurationen -----
Alive = Led_aus                                             ' Alive-LED aus
Pwrled = Led_aus                                            ' LED für Spannungsüberwachung

' ----- Sound -----
Piezosound = Sound_aus                                      ' Sound ausschalten


'-------------------------------------------------------------------------------
' Und los gehts, hier noch die Restarbeiten
'-------------------------------------------------------------------------------

' ----- Freigabe aller Interrupts ----
Enable Interrupts                                           ' Damit auch Empfang von Daten über
Buffer


' --------------------------------- Gosub's ---------------------------------


' ###############################################################################
'
'                             Hauptprogramm ConvCtrl
'
' ###############################################################################


' ---------------------------------------------
' ----- Hier ist die Programmhauptschleife -----
' ---------------------------------------------

' In Abhängigkeit der Konstante Traceausgabe schreiben
#if Main_testmodus
   Print "** OK, let's GO **"
#endif

Do                                                          ' Hauptschleife

   Piezosound = Sound_ein
   Waitms 200
   Piezosound = Sound_aus
   Waitms 200

Loop                                                        ' Hauptschleife

'## End Hauptprogramm ########################################################

End


'*******************************************************************************
' Interruptroutinen
'*******************************************************************************

'-------------------------------------------------------------------------------
' Interrupt-Service-Routine (Timer1): Sekunden_tick
' Routine zur Auswertung des Timer Interrupts
'-------------------------------------------------------------------------------
Sekunden_tick:

   ' ----- Programmcode -----

   Timer1 = Timervorgabe                                    ' Timer neu laden
   Toggle Alive                                             ' Alive-LED toggeln lassen
```

```
Return
'-- End Sekunden_tick  ------------------------------------------------------


'*******************************************************************************
' Subroutinen
'*******************************************************************************


'*******************************************************************************
' GOSubroutinen
'*******************************************************************************


'-------------------------------------------------------------------------------
' Devices schließend und ggf. "Terminate Programm execution"
'-------------------------------------------------------------------------------

' System halt
End                                                     'end program

'-------------------------------------------------------------------------------
' Definition von globalen Konstantenfeldern
'-------------------------------------------------------------------------------


' ------------------------------------------------------------------------


'###################################### END ######################################


'################################### Historie ###################################
' 27.05.2014 : Version x.x
'              Beginn der Implementierungen für Hanging Man
' 27.05.2014 : Versions 1.0
'              Fertigstellung Inbetriebnahme Piezo-Soundausgabe
'################################################################################
```

Software 4: Code zur Ansteuerung des Summers

9.7 LED-Zuordnung Schaltplan

Im Projekt Hangingman werden insgesamt zwei 3mm Low-Current-LED's verwendet welche ohne Transistortufe mittels Metallschichtvorwiderstand direkt an den GPIO-Pin's des ATmega8L-Controllerst betrieben werden.

In der folgenden Tabelle befinden sich alle LED's des Projekt beschrieben und den HW-Rssourcen im Schaltplan inkl. Funktion zugeordnet:

LED	*Funktion*	*Abkürzung*	*Farbe*
1	Alive LED	LED 1	GRÜN
2	Power LED	LED 2	ROT

Tabelle 43: LED-Zuordnung Schaltplan

Die grüne Alive LED zeigt an, dass der Controller bzw. das Spiel normal arbeitet und der Hauptschalter eingeschaltet ist. Dabei blinkt die LED im Sekundenrhythmus.

Die rote LED dient der Batteriespannungsüberwachung. Sinkt die Batteriespannung über eine in der SW eingestellte Schwelle so leuchtet die rote LED und die Batterie sollte mittelfristig getauscht werden.

Bauteile:

Stückliste: LED-Ansteuerung			
Widerstände		*Halbleiter*	
R1, R2	Metallschichtwiderstand 1k2 Ω	LED1	Low Current LED, 3mm, green
		LED2	Low Current LED, 3mm, red

Tabelle 44: Bauteile für LED-Ansteuerung

Ressourcenzuordnung zum ATmega8L:

Nummer	*Schaltbild*		*Ressource ATmega8*		
1	ALIVE	ALIVE	PortB.0	[PB0]	(GPIO Output)
2	PWRLED	PWRLED	PortB.1	[PB1]	(GPIO Output)

Tabelle 45: Ressourcenzuordnung für LED-Asnsteuerung

Die Gehäusedurchführung / Gehäusefassung für die 3mm LED's wird als LED-Fassung aus Polyamid ausgeführt.

LED-Fassung Gummi Passend für LED 3 mm Donau 3 SC

Passend für LED 3 mm
Einbau-Ø 5.7 mm

Bestellnummer Conrad: 184802 - 62

10 Mechanik

Die Schwierigkeit bestand darin, ein formschönes Gehäuse zu finden welches nicht zu groß ist sondern das auch nicht gut in der Hand gehalten werden kann und in dem sowohl das ausgewählt Display als auch der 9V-Batterieblock Platz findet.

Nach langer Suche wird nun das folgende Gehäuse eingesetzt:

BOPLA BOS 750

Digitales Handgehäuse

Die Tastenfläche ist vertieft zur Aufnahme einer Folientastatur. Batteriefach für 9V-Block, mit Panoramascheibe (muss von außen geklebt werden) Tastenfeld: 70 x 86mm

Hersteller : BOPLA
Artikelnummer des Herstellers : 34750000
Verpackungsgewicht : 0.109 kg
RoHS : konform

Bestllnummer Reichelt: *BOPLA BOS 750*

Typ	Handgehäuse
Farbe	schwarz
Ausführung	mit Panoramascheibe
Länge	157 mm
Breite	84 mm
Höhe	30 mm

Tabelle 46: Gehäuse

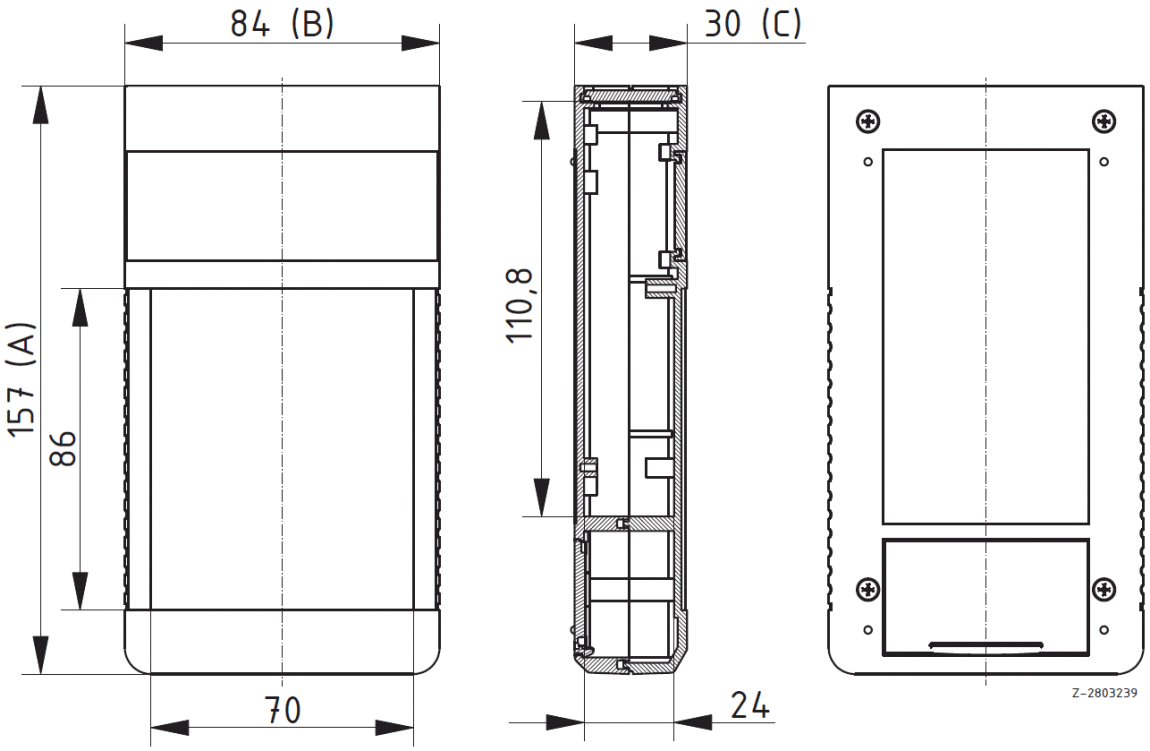


Abbildung 16: Gehäusemaße

11 Bauteile und Bauteilbeschaffung

Für das Prototyping und die Inbetriebnahme der im obigen Kapitel „Grundlagen" beschriebene Einzelthemen werden die Einzelteile in Verbindung mit dem STK500 und Labor-Lochrasterplatinen aufgebaut und in Betrieb genommen.

Die folgenden Stücklisten geben einen Überblick über die benötigten Bauelemente und ergeben gleichzeitig die Bestellisten von Conrad Electronic, Reichelt Electronic und Farnell.

Bauelemente Reichelt Electronik:

Stk.	*Beschreibung*	*Bestellnummer*	*Einzelpreis*	*Gesamtpreis*
		Kondensatoren		
2	Keramikkondensator 22pF 50 VDC	KERKO 22P	0,06	0,12
1	Keramikkondensator 47nF 50 VDC	KERKO 47N	0,093	0,093
7	Keramikkondensator 100nF 50 VDC	KERKO 100N	0,063	0,441
2	Elektrolytkondensator 100µF / 35V	RAD 100/35	0,04	0,08
		Widerstände		
2	Metallschichtwiderständ 1M Ohm	METALL 1,00M	0,082	0,162
4	Metallschichtwiderständ 1k2 Ohm	METALL 1,20K	0,082	0,328
1	Metallschichtwiderständ 4k7 Ohm	METALL 4,70K	0,082	0,082
1	Metallschichtwiderständ 30 Ohm	METALL 30,0	0,082	0,082
		Halbleiter		
1	Diode 1N4148	1N 4148	0,02	0,02
1	Spannungsregler 2A positiv, TO-220	µA 78S05	0,35	0,35
1	DIL Brückengleichrichter B80C800DIP	B80C800DIP	0,17	0,17
2	BC 547B Transistor NPN TO-92 45V 0,1A 0,5W	BC 547B	0,04	0,08
1	DOG LCD-Modul 16x2, Hintergrund schwarz	EA DOGM162S-A	9,05	9,05
1	Led-Beleuchtung für EA DOGL..Farbe: weiss	EA LED55X31-W	7,30	7,30
1	LED 3mm, low-Current, grün	LED 3MM 2MA GN	0,08	0,08
1	LED 3mm, low-Current, rot	LED 3MM 2MA RT	0,08	0,08
1	ATMega AVR-RISC-Controller, S-DIL-28	ATMEGA 8L8 DIP	1,85	1,85
		Sonstiges		
1	Standardquarz, Grundton, 8,0 MHz	8,0000-HC49U-S	0,17	0,17
1	Kurzhubtaster 6x6mm, Höhe: 4,3mm, 12V, vertikal	TASTER 3301	0,11	0,11
1	Piezosummer	SUMMER CPM 121	2,20	2,20
1	Drosselspule 10µH INDUCTOR0207/12	SMCC 10µ	0,16	0,16
1	MA03-2 für ISP06	SL 2X50G 2,54	0,71	0,71
1	MA06-02 für RS232	SL 2X50G 2,54	s.o.	s.o.
1	ALPS STEC11B Drehimpulsg., 20/20, vert., MT	STEC11B13	3,95	3,95
1	Gehäuse	BOPLA BOS 750	14,40	14,40
1	Batterieclip für 9-Volt-Block, High-Quality, vertikal	CLIP HQ9V	0,36	0,36
1	Drehknopf für Rundachse 6mm	KNOPF 13-164B	0,40	0,40
1	IC-Sockel, 28-polig, schmal	GS 28P-S	0,43	0,43

Tabelle 47: Bauelemente Reichelt Electronik

Bauelemente Conrad Electronic:

Stk.	*Beschreibung*	*Bestellnummer*	*Einzelpreis*	*Gesamtpreis*
Sonstiges				
1	9 V Block-Batterie Alkali-Mangan Varta Longlife 6LR61 9 V	650709 - 62	3,99	3,99
2	LED-Fassung Gummi Passend für LED 3 mm Donau 3 SC	184802 - 62	0,14	0,28
1	Wippschalter 250 V/AC 3 A 1 x Aus/Ein Eledis MR519-0F522 rastend	700039 – 62	2,23	2,23
1	Trockenmittelbeutel 5 g/Beutel Transparent Material Kieselgel	181896 – 62	1,52	1,52

Tabelle 48: Bauelemente Conrad Electronic

Die vorausberechneten Materialkosten belaufen sich auf ca. 50,848 Euro.

12 Hardware

12.1 Festlegung von Netzklassen im Projekt

Für die Umsetzung der PCB in EAGLE werden die folgenden Netzklassen definiert:

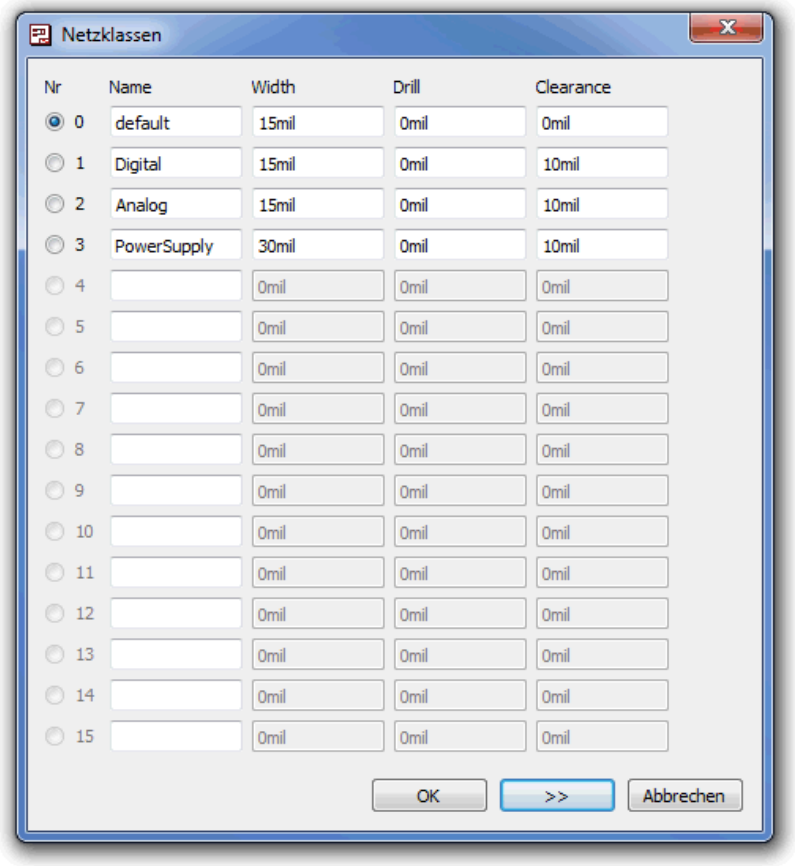
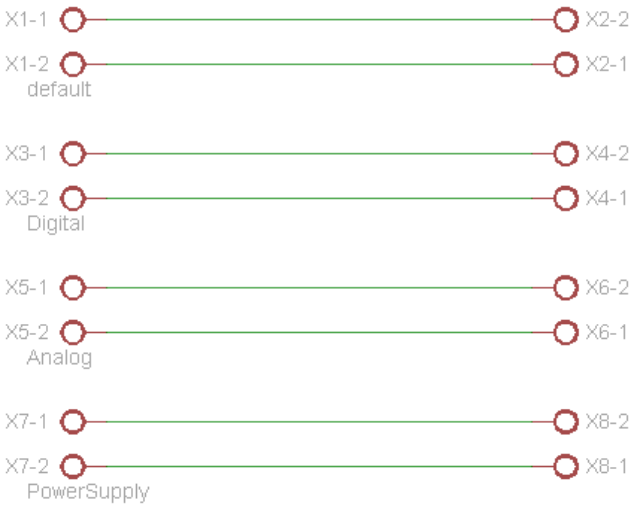


Abbildung 17: Definition der Netzklassen

Diese Vorgaben führen bei einem einfachen Schaltbild zur folgenden Umsetzung auf dem Board:



Schaltbild 11: Schaltbild für Definition von Netzklassen

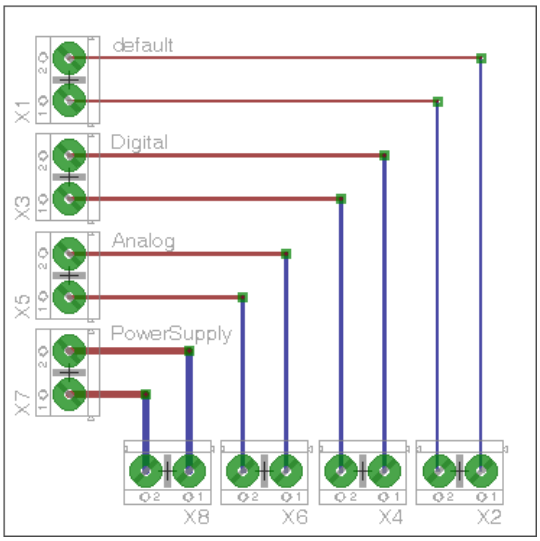
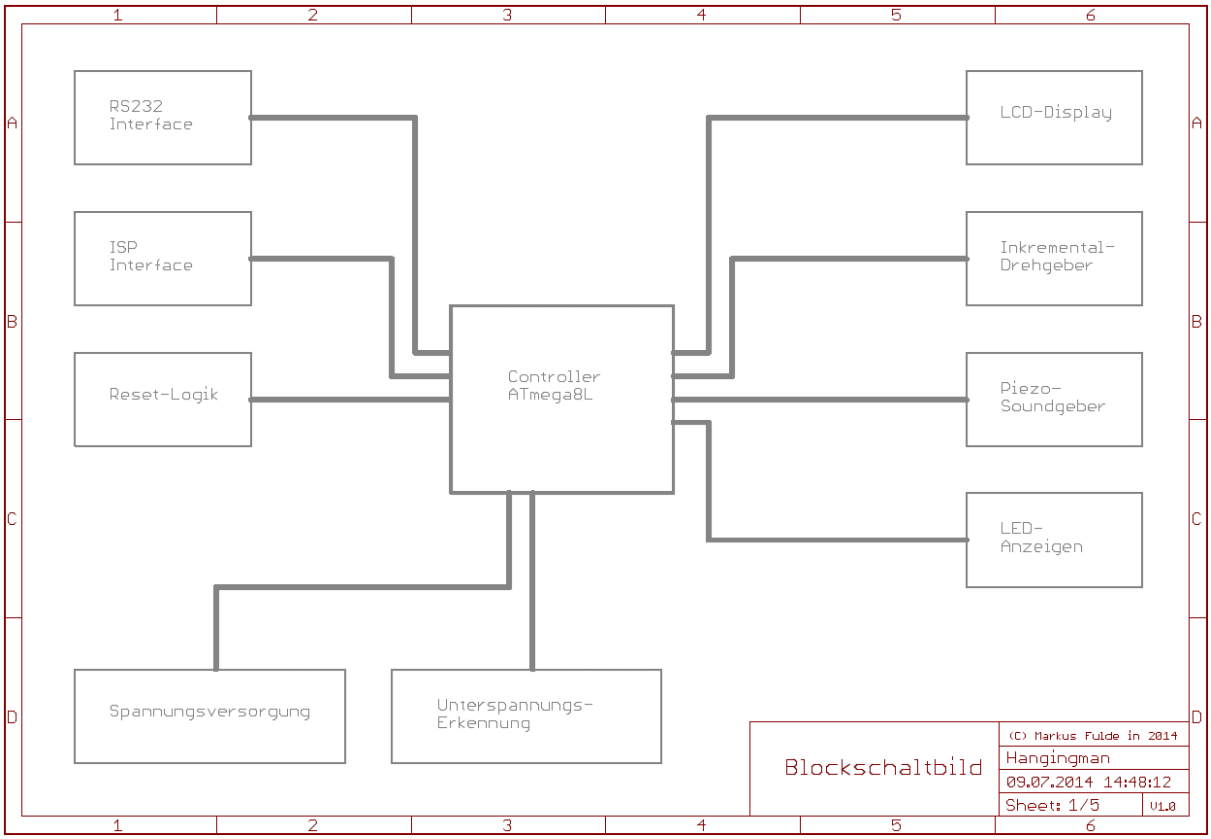


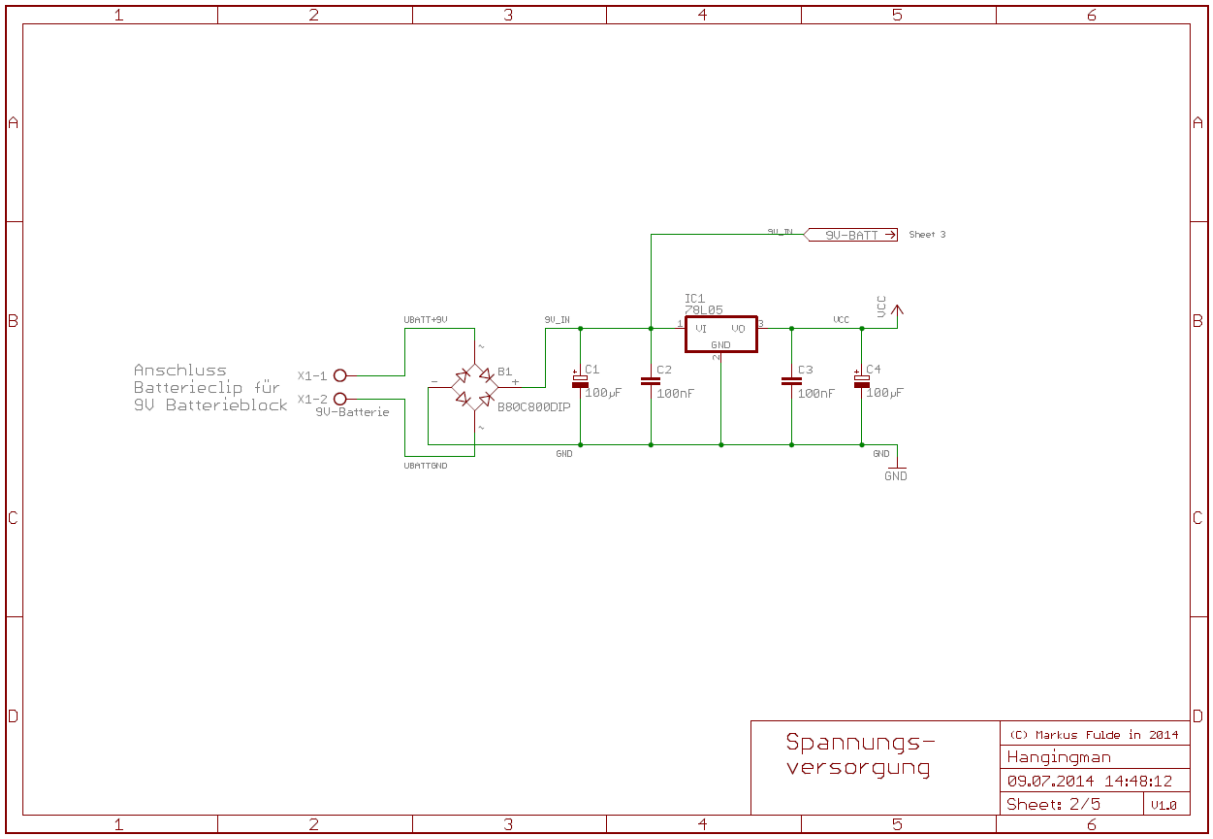
Abbildung 18: Demoboard N etzklassen

12.2 Die PCB zum Hanging Man

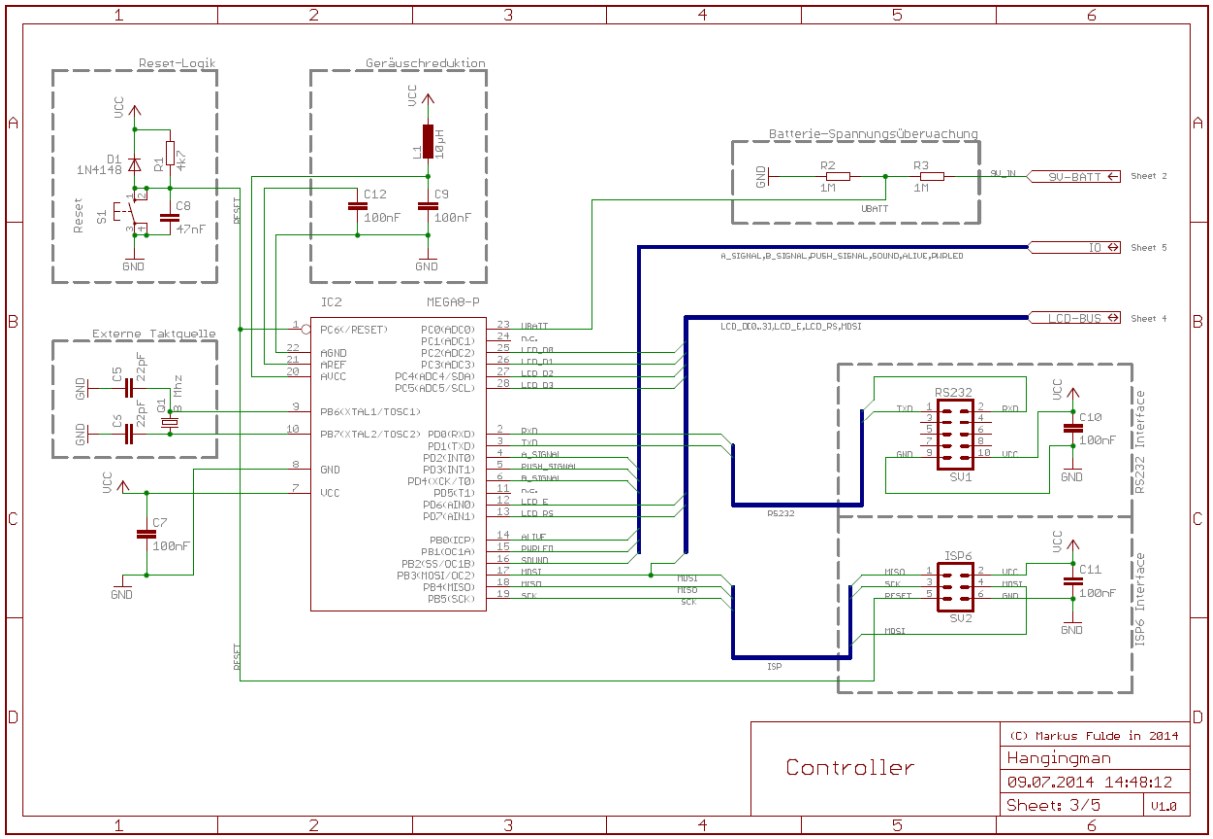
12.2.1 Schematic



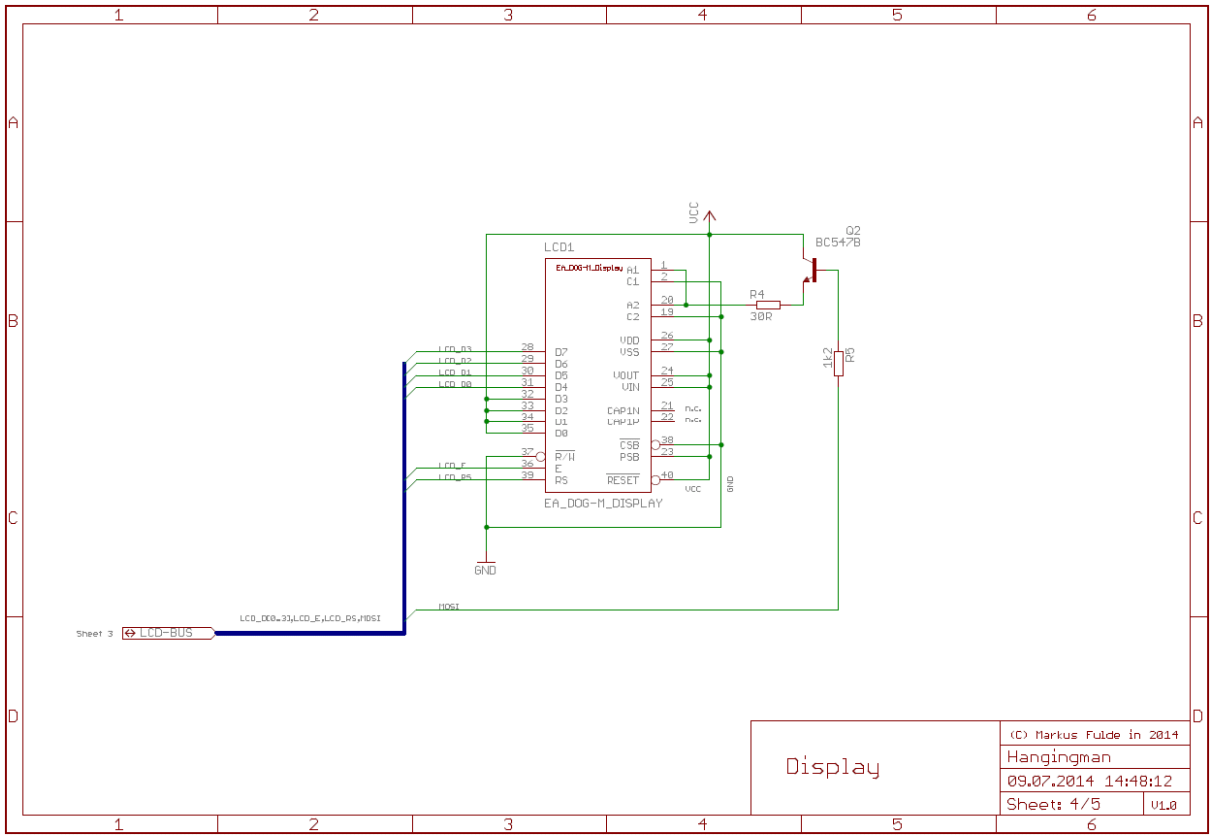
Schaltbild 12: Schaltbild HangingMan - Sheet 1



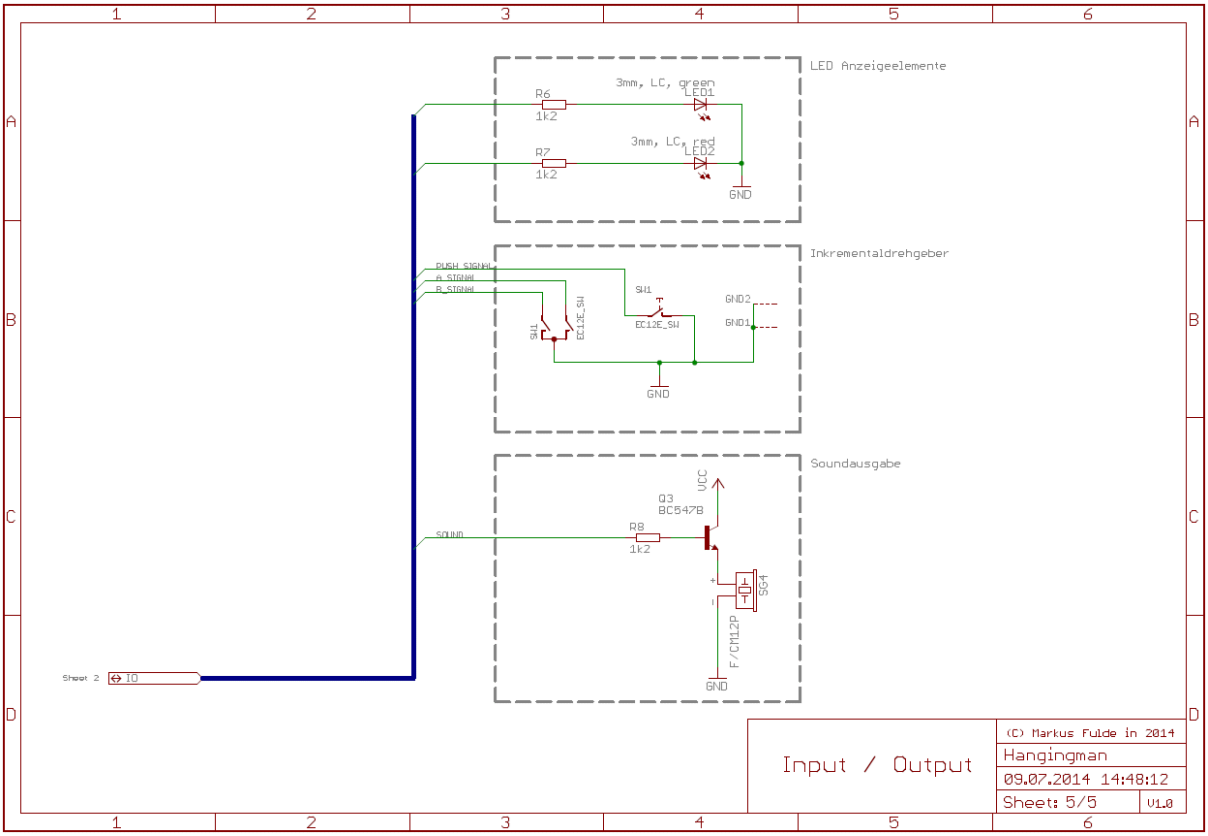
Schaltbild 13: Schaltbild HangingMan - Sheet 2



Schaltbild 14: Schaltbild HangingMan - Sheet 3



Schaltbild 15: Schaltbild HangingMan - Sheet 4



Schaltbild 16: Schaltbild HangingMan - Sheet 5

12.2.2 Layout, Layer und Bestückung

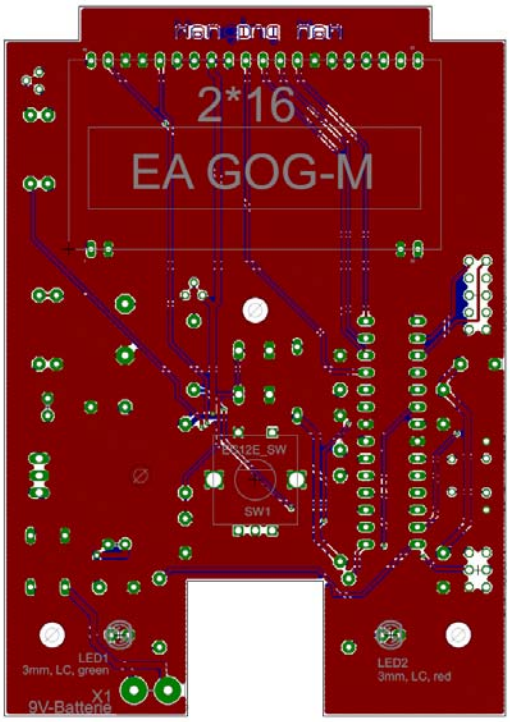


Abbildung 19: PCB Hangingman – Layout gesamt

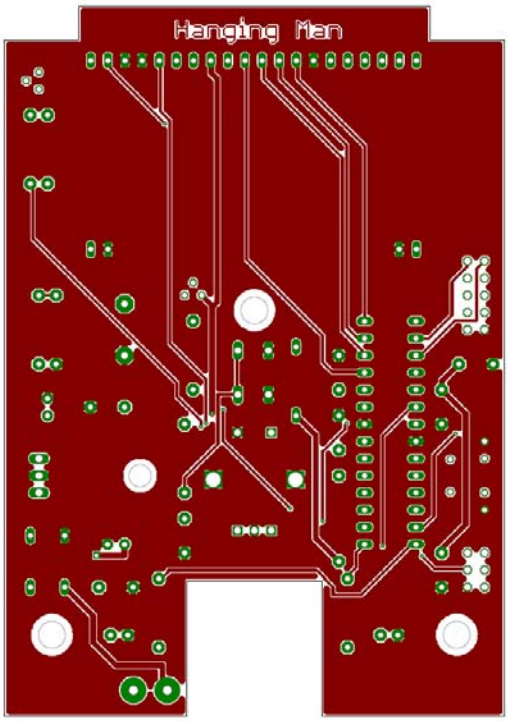


Abbildung 20: PCB Hangingman – Top Layer

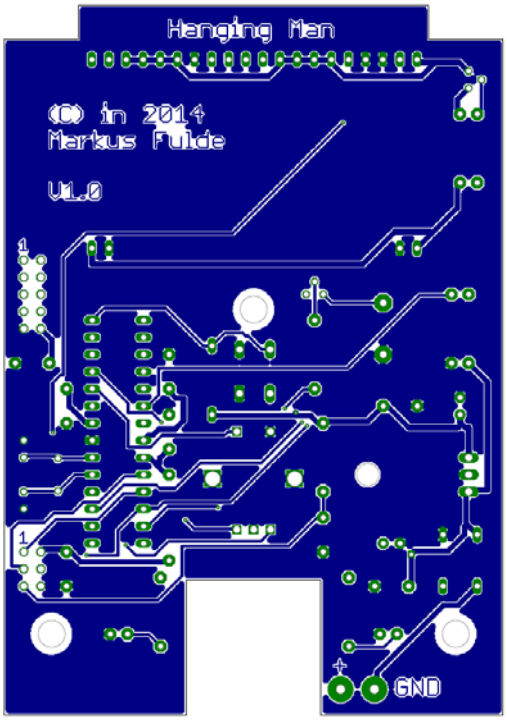


Abbildung 21: PCB Hangingman – Bottom Layer

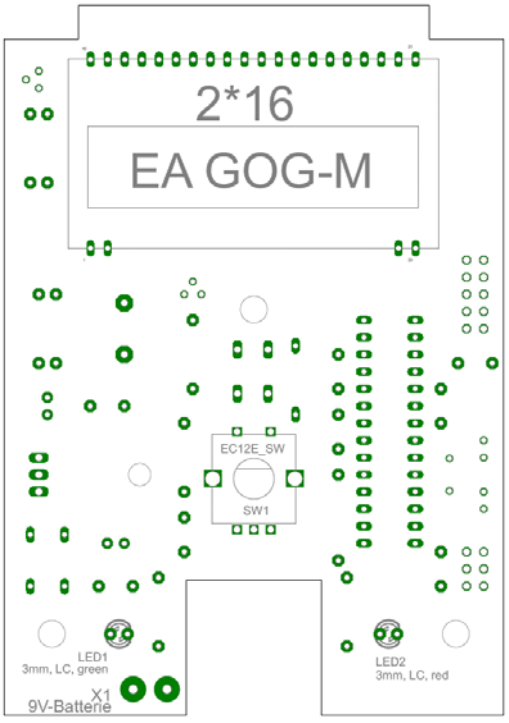


Abbildung 22: PCB Hangingman – Bestückung Top Layer

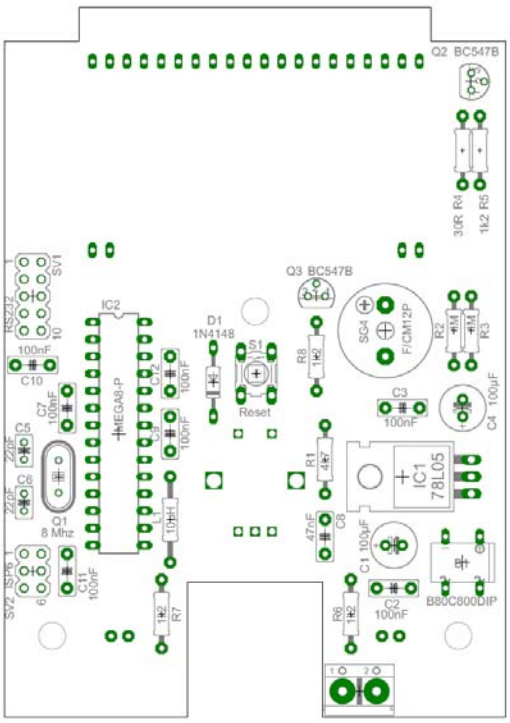


Abbildung 23: PCB Hangingman – Bestückung Bottom Layer

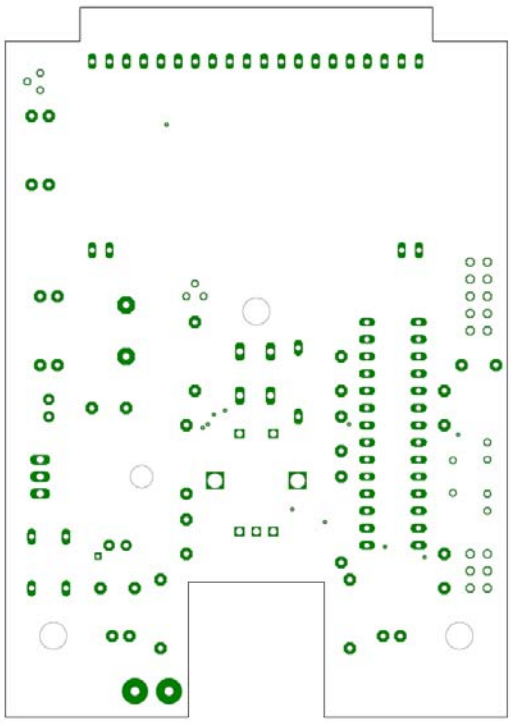


Abbildung 24: PCB Hangingman – Pads und Vias

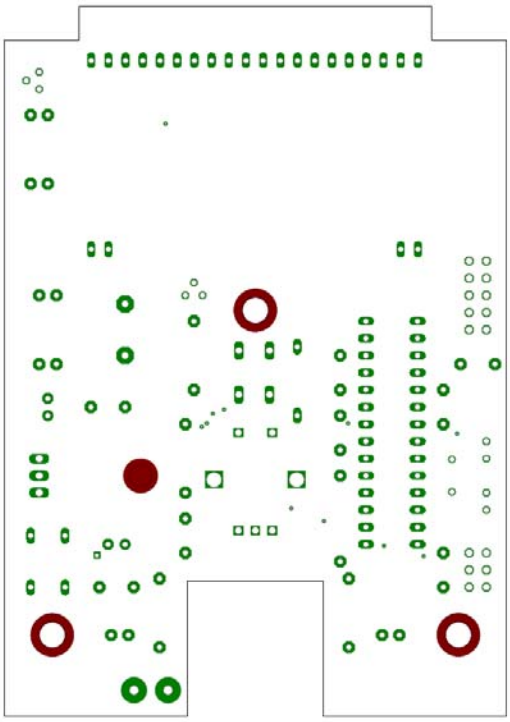


Abbildung 25: PCB Hangingman – Restricted Areas

12.2.3 Eagle-BOM

Pos.	Bauteile	Menge	Wert	Device
1.	D1	1	1N4148	1N4148
2.	IC1	1	78L05	7805T
3.	X1	1	9V-Batterie	AKL055-02
4.	B1	1	B80C800DIP	B250C1000
5.	Q2, Q3	2	BC547B	BC547B
6.	C5, C6	2	22pF	CAPACITOR025-025X050
7.	C8	1	47nF	CAPACITOR050-025X075
8.	C2, C3, C7, C9, C10, C11, C12	7	100nF	CAPACITOR050-025X075
9.	Q1	1	8 Mhz	CRYTALHC49U-V
10.	LCD1	1	EA_DOG-M_DISPLAY	EA_DOG-M_DISPLAY
11.	SW1	1	EC12E_SW	EC12E_SW
12.	SG4	1	F/CM12P	F/CM12P
13.	L1	1	10µH	INDUCTOR0207/12
14.	LED1	1	3mm, LC, green	LED3MM
15.	LED2	1	3mm, LC, red	LED3MM
16.	SV2	1	ISP6	MA03-2
17.	SV1	1	RS232	MA05-2
18.	IC2	1	MEGA8-P	MEGA8-P
19.	S1	1	Reset	MJTP1230
20.	C1, C4	2	100µF	POLARIZEDCAPACITORE2.5-7
21.	R2, R3	2	1M	RESISTOR0207/10
22.	R5, R6, R7, R8	4	1k2	RESISTOR0207/10
23.	R1	1	4k7	RESISTOR0207/10
24.	R4	1	30R	RESISTOR0207/10

Tabelle 49: Eagle BOM für das Projekt Hanging Man

Folgende Bauteile werden noch außerhalb von Eagle benötigt:

Pos.	Bauteile	Menge	Wert	Device
1.	--	1	Gehäuse	BOPLA BOS 750
2.	--	1	9V Batterieclip	--
3.	--	1	9V Blockbatterie	--
4.	--	1	LED Gehäusedurchführungen 3mm	--
5.	--	1	LED Display Hintergrundbeleuchtung	EA LED55X31W
6.	--	1	Miniatur Wippschalter	--
7.	--	1	Drehknopf für DDS	--
8.	--	1	Trockenbeutel 5 Gramm	--

Tabelle 50: Weitere Bauteile für das Projekt Hanging Man

12.2.4 Das Board

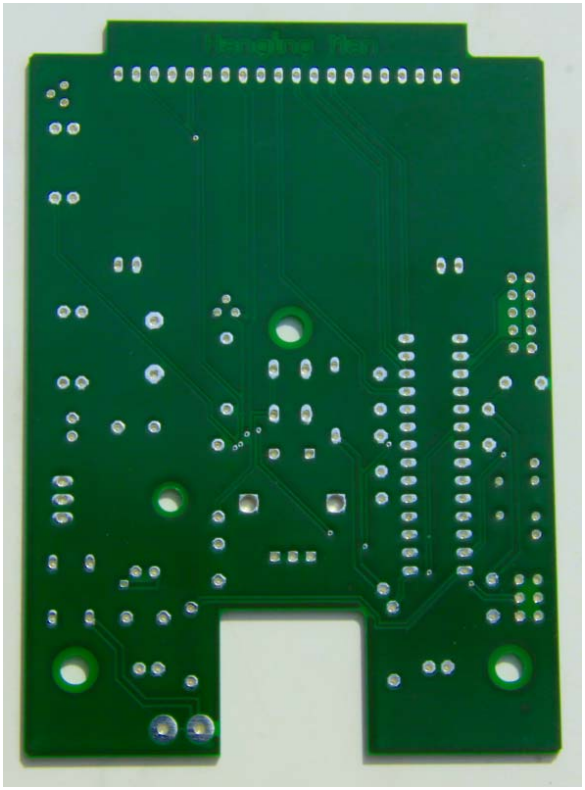


Abbildung 26: PCB Hangingman TOP

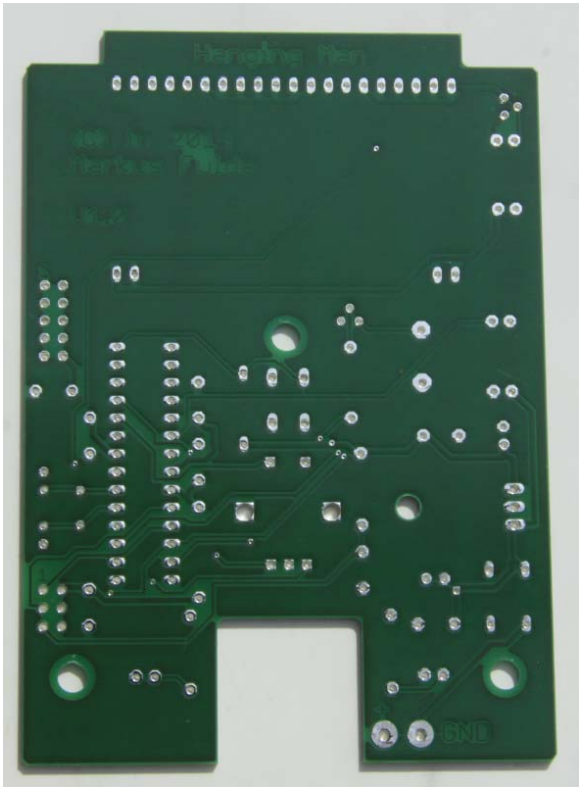


Abbildung 27: PCB Hangingman BOTTOM

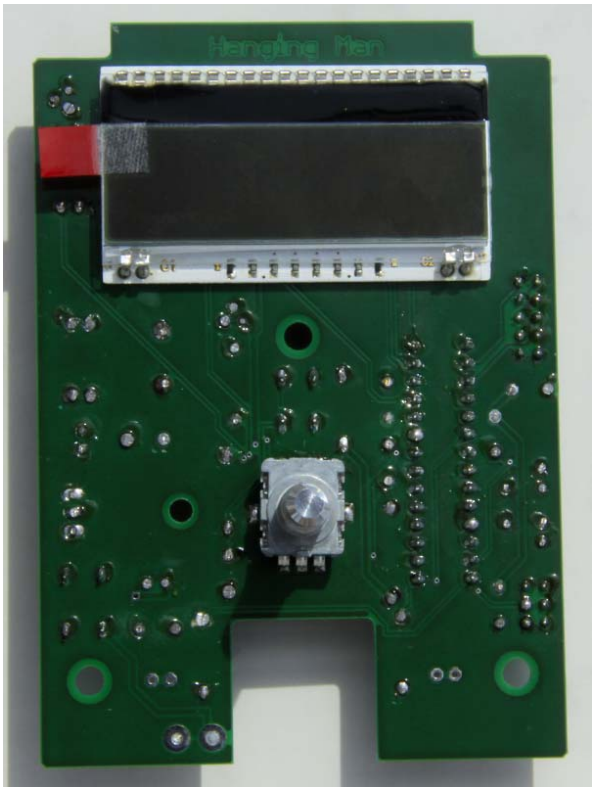


Abbildung 28: PCB TOP fertig bestückt

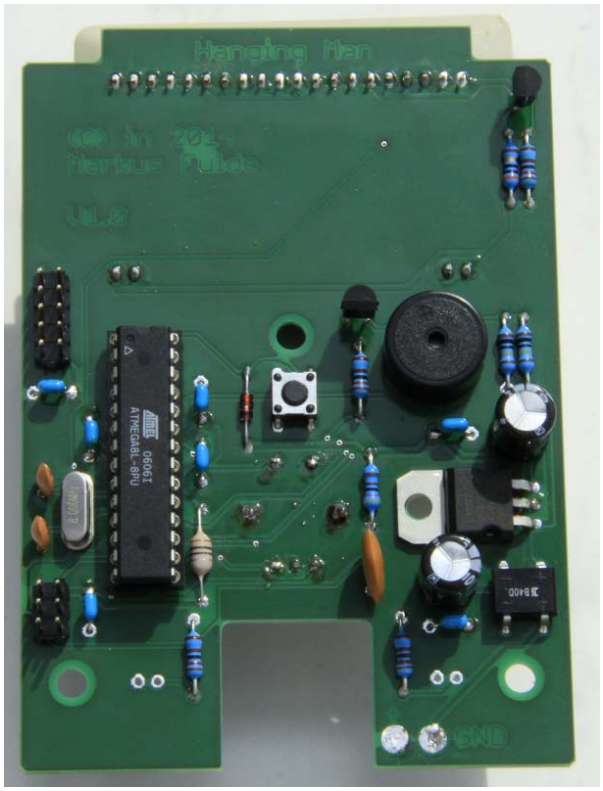


Abbildung 29: PCB BOTTOM fertig bestückt

12.3 Die fertige Hardware

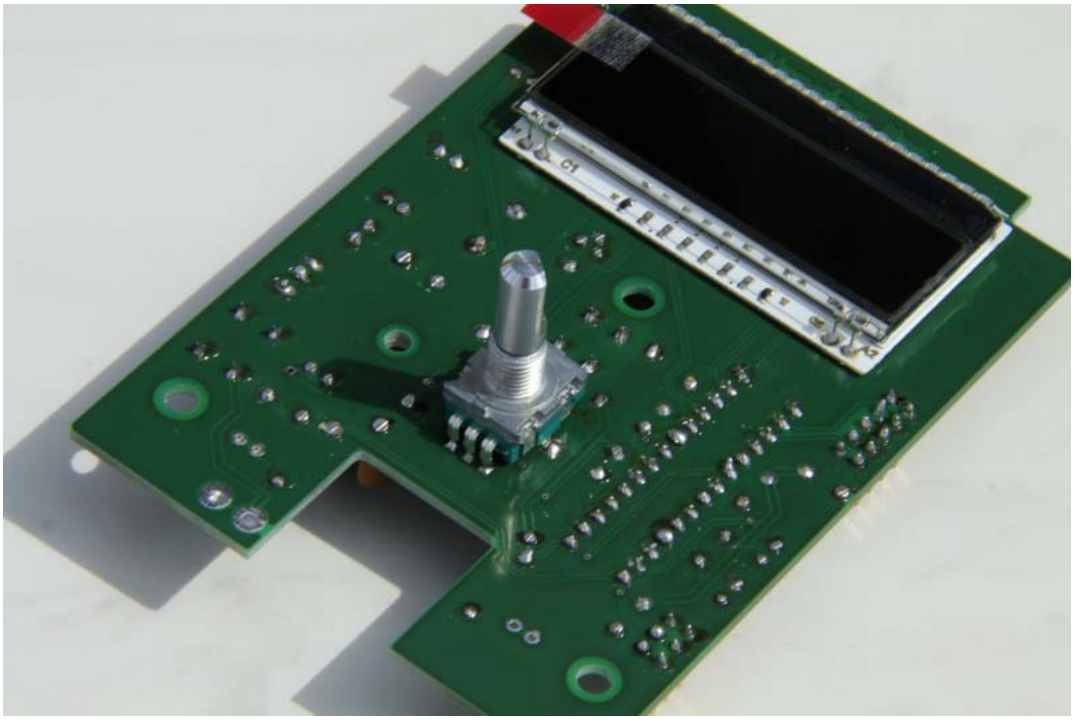


Abbildung 30: Die fertige Platine

Abbildung 31: Das fertige Gerät im Gehäuse montiert

Abbildung 32: Galgenmännchen in Betrieb

13 Software

13.1 Systemfestlegungen und Definitionen

13.1.1 Timerfestlegungen

Die Timer im Gesamtsystem haben der Priorität nach die folgende Reihenfolge:

1. Timer0 8-Bit Timer
2. Timer1 16-Bit Timer
3. Timer2 8-Bit Timer

Timer0:

Nicht verwendet!

Timer1:

Der Timer1 ist im ATmega8L der Timer mit der mittleren Priorität. Mit seiner Hilfe wird ein SW-Timer aufgebaut. Der Timer1 versorgt das Gesamtsystem mit einem 1-Sekunden-Timertick und sorgt für das Toggeln der BetriebsLED.

Timer2:

Mit dem Timer 2 wird die PWM für die Hintergrundbeleuchtung realisiert. Dieser Time rist ein 8 Bit Timer.

13.2 KnowHow: PWM-Signale mit Bascom erzeugen

13.2.1 Grundbegriffe

Bei der Puls-Weiten-Modulation (PWM) wird ein digitales Ausgangssignal erzeugt, dessen Tastverhältnis moduliert wird.
Das Tastverhältnis gibt das Verhältnis der Länge des eingeschalteten Zustands zur Periodendauer an. Dabei bleiben die Frequenz und der Pegel des Signals immer gleich! Es ändert sich nur die Länge von High zu Low.

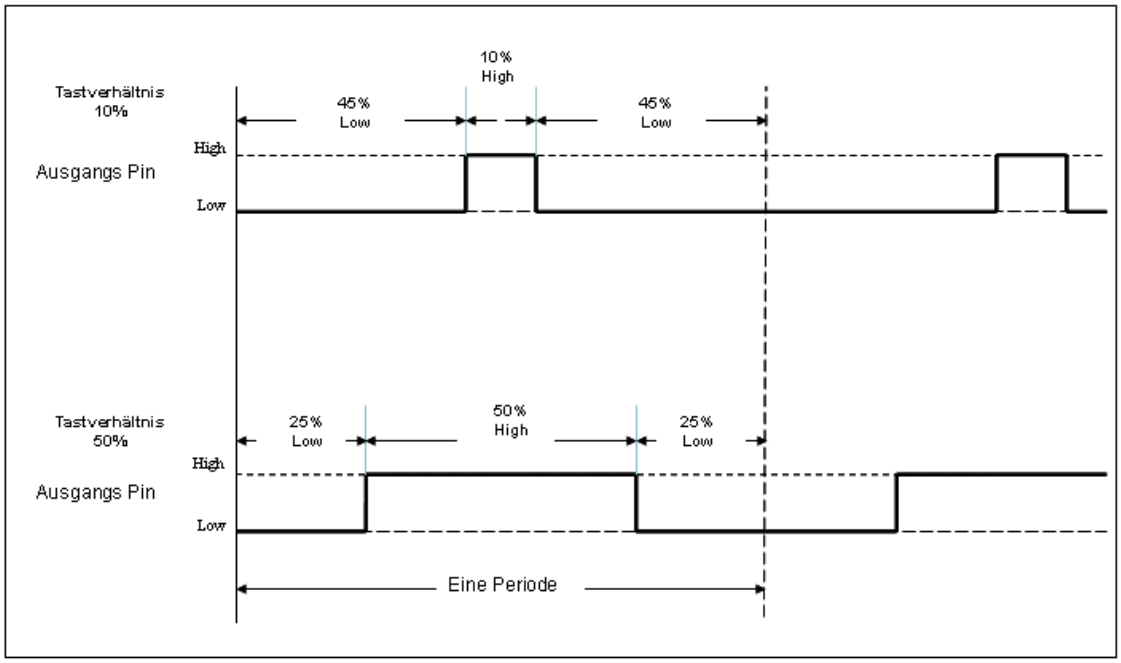


Abbildung 33: PWM Tastverhältnis einmal von 10% und einmal von 50%

Man könnte das in etwa mit einem Wasserhahn vergleichen, den man z.B. alle Minuten betätigt. Wenn man ihn in jeder Minute nur kurz aufdreht und dann gleich wieder zumacht, kommt in Summe nur wenig Wasser raus. Wenn man ihn aber in jeder dieser Minuten länger offen lässt, kommt mehr Wasser raus.

Der Rhythmus bleibt immer gleich, aber es ändert sich in Summe die Wassermenge, die raus kommt.

Mit dem PWM-Signal kann man nun tolle Sachen machen.
Zum Beispiel:

- eine LED (oder über einen Transistor auch eine Lampe) an den PWM-Ausgang anschliessen und mit der Länge des PWM-Signal's die Helligkeit der LED verändern.
- einen Motor in der Geschwindigkeit regeln.
- mittels nachgeschaltetem RC-Filter, welcher das PWM-Signal glättet, kann eine Gleichspannung erzeugt werden die zwischen 0V und 5V geregelt werden kann.

13.2.2 PWM-Arten

Es gibt zwei Arten PWM-Signale in Bascom zu erzeugen:

Software PWM

Vorteile:

- es kann (fast) jeder Ausgabe-Pin des AVR benutzt werden.
- unter zuhilfename eines (freien) Timers können sogar mehrere verschiedene PWM-Signale auf verschiedene Pins erzeugt werden.

Nachteil:

- Etwas grösserer Programmaufwand, da der PortPin per Software verändert werden muss.

Hardware PWM

Vorteile:

- Sehr schnell (Maximal die Quarzfrequenz / Periode)
- unabhängig vom Programmablauf des AVR

Nachteile:

- Je nach AVR können nur bestimmte Timer mit bestimmten Ausgangspins dafür verwendet werden.
- belegt den Timer, der für keine weiteren funktionen verwendet werden kann.

Beim einem ATMega8 stehen drei Hardware-PWM-Ausgänge verteilt auf zwei Timer zur Verfügung.

Mit Timer1 können zwei PWM Signale erzeugt werden (Compare A => OC1A - Pin 15 und Compare B => OC1B - Pin 16).

Die Auflösung kann auf 8, 9 und 10 Bit eingestellt werden, also max. 1024 Abstufungen.

Timer2 kann ein PWM-Signal mit einer Auflösung von 8 Bit erzeugen (Compare Register => OC2 - Pin 17)

13.2.3 PWM-Ablauf

Das folgende Bild zeigt den Ablauf bei Timer1. Als Taktquelle dient die CPU-Frequenz, dessen Frequenz im Prescaler (Vorteiler) nochmal verkleinert werden kann. Je nach eingestelltem Wert in den Output Compare Registern wird der Status des Ausgangs-Pin entsprechend oft umgeschaltet, und erzeugt somit das PWM-Signal.

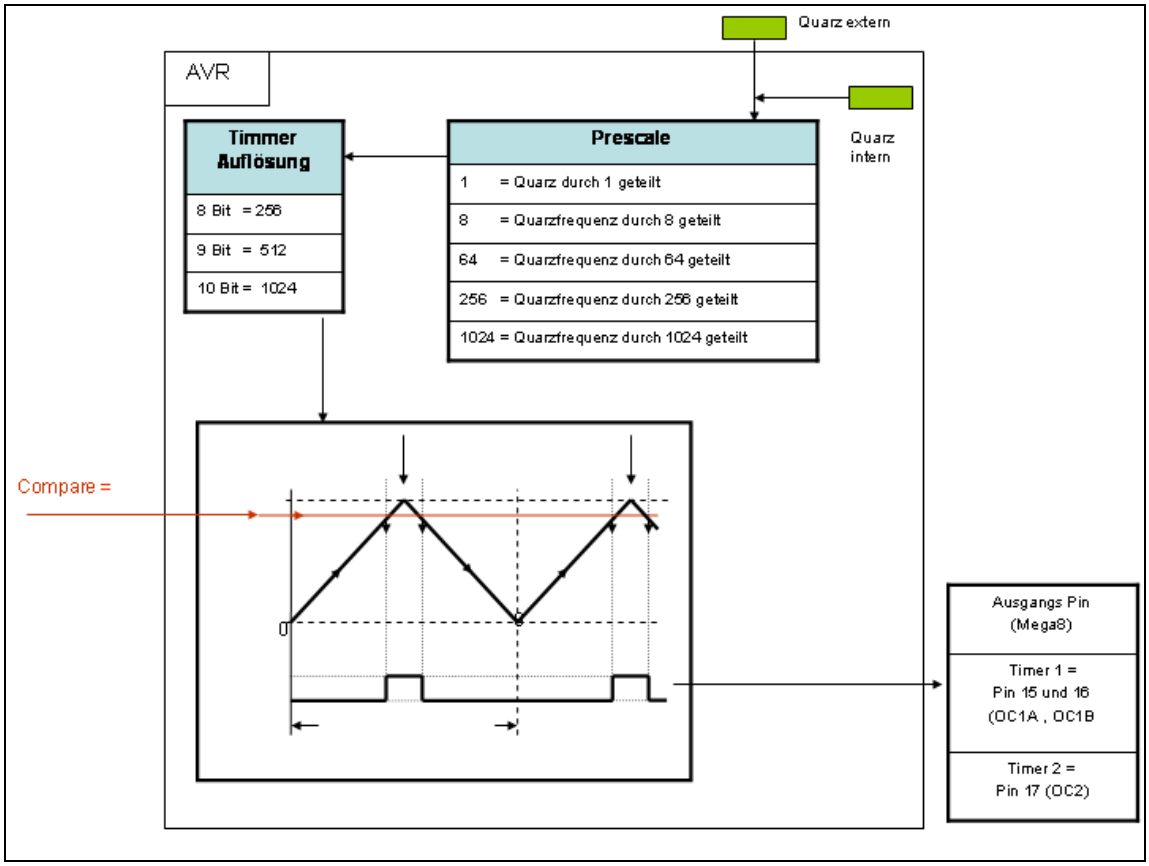


Abbildung 34: PWM Konfiguration des ATmega

13.2.4 Genauere Erklärung

Der Prescaler teilt die Frequenz die vom Quarz kommt! Bei Teilung 1 geht die vollständige Frequenz zum Timer. Bei Teilung 8 nur ein Achtel der Quarzfrequenz. (Also bei 8 MHz Quarz kommen zum Timer dann nur 1 MHz) Bei Teilung 1024 sind es dann z.B. 8 MHz/1024 = 7,8125kHz

Der Timer ist hier der Zähler für die PWM. Er zählt mit der Frequenz, die vom Prescaler kommt, einmal von 0 bis zu der eingestellten Timerauflösung rauf, dann wieder auf 0 zurück. (dann wieder von 0 auf Timerauflösung

u.s.w.) Einmal rauf- und runterzählen, ergibt ein Periode. Die Periode ist gleich die Ausgangsfrequenz des PWM-Signals. Ausgangsfrequenz = (Quarzfrequenz/Prescale) /(Timerauflösung*2)

z.B.: Quarz = 8 MHz ; Prescaler = 1 ; Timer = 8 Bit ergibt: (8000000Hz/1) / (256*2) = 15,625 kHz

oder: Quarz = 8 MHz ; Prescaler = 8 ; Timer = 10 Bit ergibt: (8000000Hz/8) / (1024*2) = 244,14 Hz

Mit dem Compare Register definiert man nun das Tastverhältnis! Überall, wo nun der Timer diese Compare Linie schneidet, schaltet der Ausgang! Beim raufzählen des Timers auf EIN, beim runterzählen auf AUS.

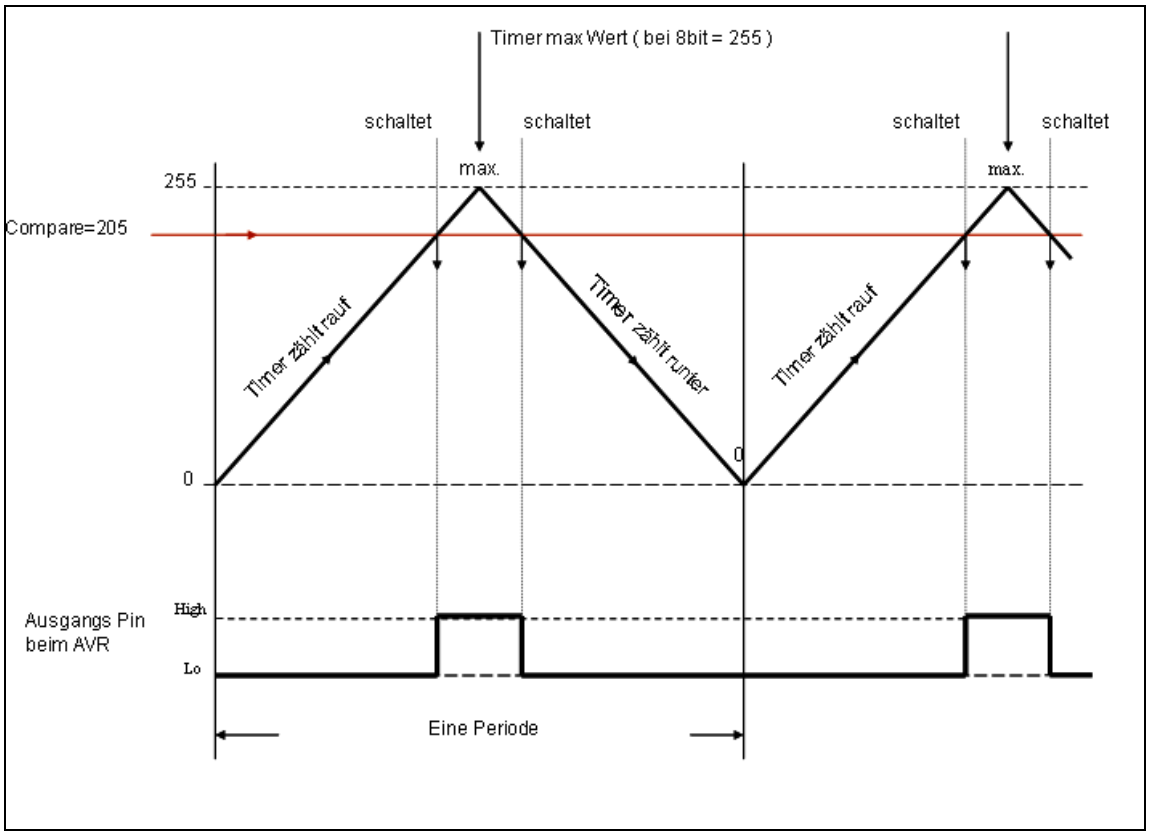


Abbildung 35: PWM mit einem Tastverhältnis 20%

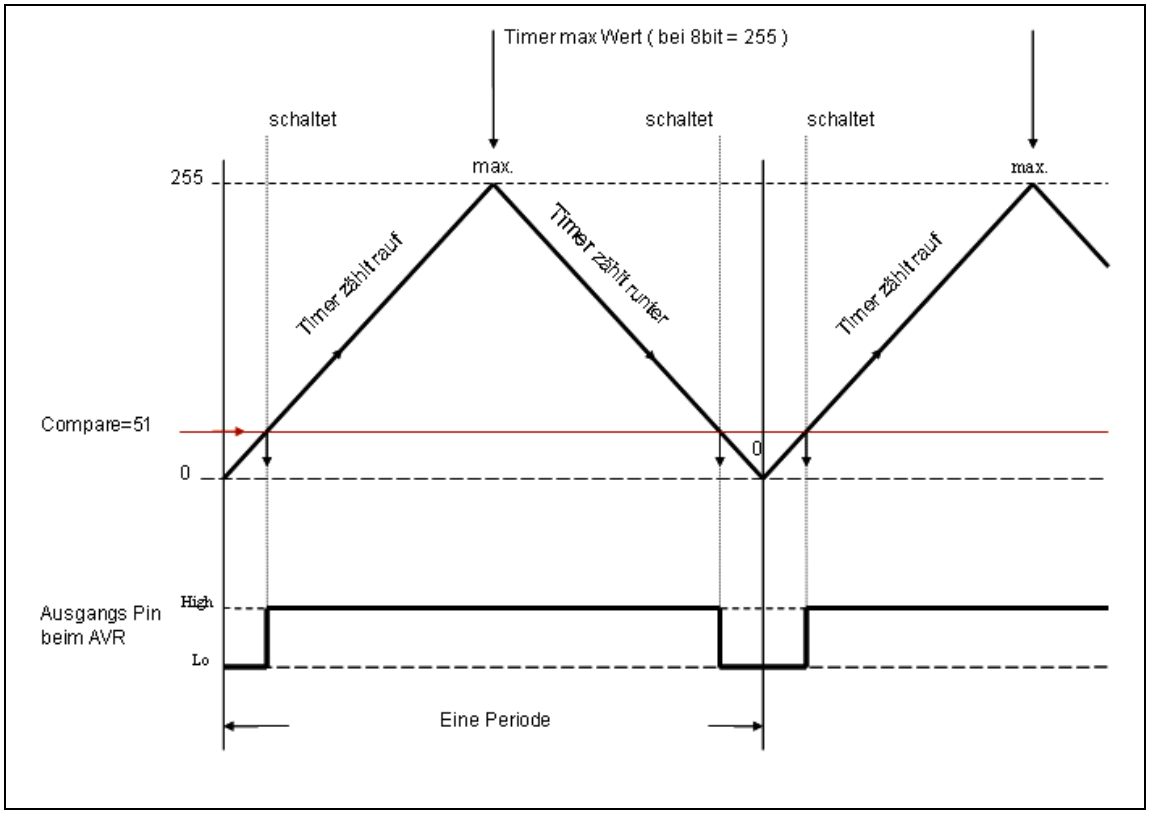


Abbildung 36: PWM mit einem Tastverhältnis von 80%

Hier sieht man, wie die Signale auf einem Oszilloskop ausschauen. Oben das Signal von Pin15 (Compare A), unten das von Pin 16 (Compare B)

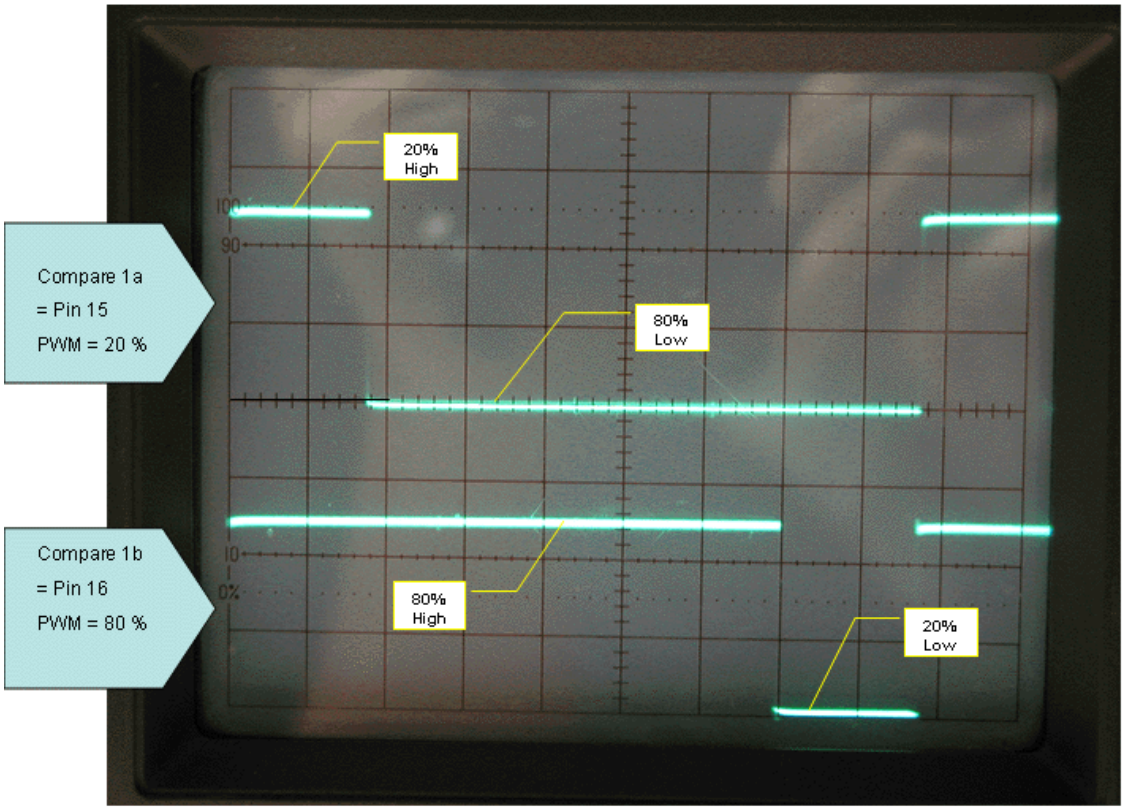


Abbildung 37: Oszillogramm der PWM mit Time 1a und 1b

13.2.5 Grundprogramm

Hier nun ein Grundprogramm für die Ausgabe von zwei PWM Signalen mit dem Timer1
' Hardware PWM mit Timer1

```
$regfile = "m8def.dat"
$crystal = 4000000

Config Portb.1 = Output
Config Portb.2 = Output

Config Timer1 = Pwm , Pwm = 8 , Compare A Pwm = Clear Up , Compare B Pwm =
Clear Up , Prescale = 1

Do
 Compare1a = 205
 Compare1b = 51
Loop

End
```

Erklärung:

```
$regfile = "m8def.dat"
$crystal = 8000000
```

Definiert den Mega8 und den 8MHz Quarz

```
Config Portb.1 = Output
Config Portb.2 = Output
```

Definiert die zwei Ausgänge von Timer1 auf Ausgabe.
Portb.1 = für Compare1a (= Compare A) = Pin 15
Portb.2 = für Compare1b (= Compare B) = Pin 16

```
Config Timer1 = Pwm , Pwm = 8 , Compare A Pwm = Clear Up , Compare B Pwm =
Clear Up , Prescale = 1
```

Config Timer1 = Pwm
: Timer1 auf PWM einstellen

Pwm = 8
: Timer Auflösung = 8 Bit einstellen

Compare A Pwm = Clear Up
: Definiert, wie der Compare A schalten soll. Bei „Clear Up" schaltet der Ausgang beim erreichen des Compare-Wertes zuerst auf High und dann auf Low. Bei „Clear Down", umgekehrt.

Compare B Pwm = Clear Up
: Das gleiche noch mal mit Compare B

Prescale = 1
: Hier wird der Prescaler auf 1 eingestellt.
(Wert 1 heißt, direkte Frequenz vom Quarz zum Timer.)
Weitere Teilungen, wie z.B.: 8, 64, 256 und 1024 sind möglich.

```
Compare1a = 205
Compare1b = 51
```

Hier kann man nun die Werte für das Tastverhältnis, in dem Register Compare1a und Compare1b übergeben. Oder man kann, statt Compare1a und 1b, auch die Bezeichnungen Pwm1a und Pwm1b verwenden, Bascom nimmt beides.

Mit diesem kurzen Programm, hat man nun zwei PWM Signale erzeugt, bei dem eines ein Tastverhältnis von 20% (Compare1a) und das andere 80 % hat. :-)

13.3 Verwendete SW

Zur Erstellung dieses Projekts kam folgende Software zum Einsatz:

- Workstation DELL XPS420: Betriebsystem Windows 7 Ultimate 64 Bit
- Notebook DELL Inspiron 17R SW: Betriebsystem Windows 8.1 Profesional 64 Bit

- BASCOM-AVR Basic Compiler BASCOM 2.0.7.7
  MCS Electronics
- EAGLE 6.6.0 Standard NON-PROFIT
- ATMEL AVR Studio 4.0 und ATMEL AVR Studio 6.0

Zur Erstellung dieses Projekts kam folgende HW-Umgebung und SDK's zum Einsatz:

- ATMEL STK500
- ATMEL ISP-Programmer AVRISP mkII
- Eigenes Prototyping auf Lochraster
- Fertiges PlatinenLayout mit Hilfe von Eagle und Herstellung durch Leiton Berlin

13.4 Bedienungsstruktur und Spielablauf

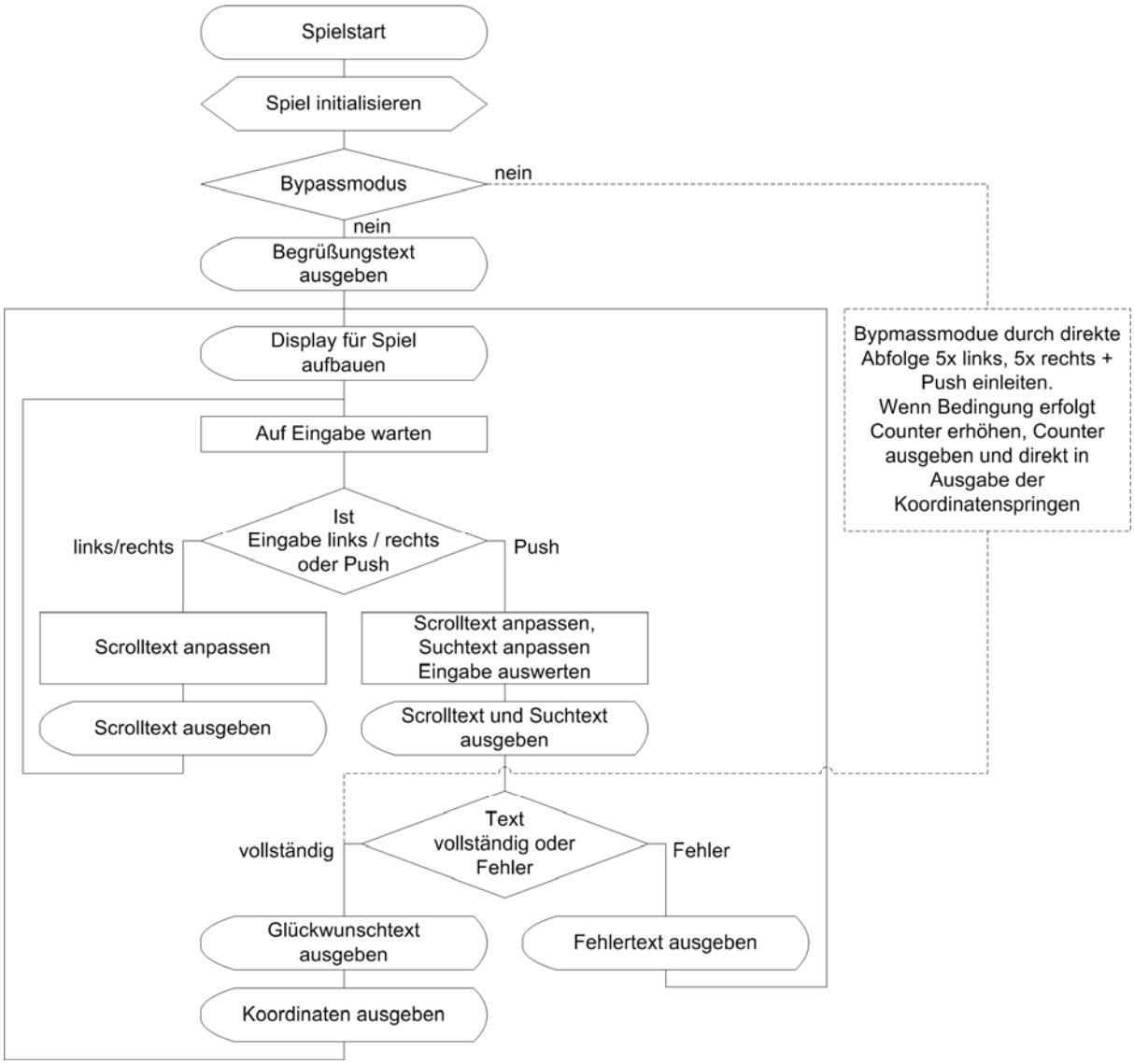


Abbildung 38: Spielablauf

13.5 Kurzanleitung

Einschalten
Das Spiel wird durch den kleinen Hauptschalter auf der Gehäuseoberseite eingeschaltet. Danach läuft die Begrüßung automatisch ab und es erfolgt der automatische Start des Spiels.

Allgemeines
Das Display besteht aus 2 Zeilen a 16 Zeichen. Auf der linken Seite werden mit jeweils 2 Zeichen in Zeile 1 und Zeile 2 der Galgen und ggf. das hängende Galgenmännchen abgebildet.

In Zeile 1 befindet sich außerdem ein Buchstaben-Auswahlbalken über den die gesuchten Buchstaben für den 13-stelligen Begriff ausgewählt werden können.

In Zeile 2 befindet sich das gesuchte 13-stellige Wort das zunächst nur durch Unterstriche verdeckt dargestellt wird. Bei Auswahl der Buchstaben wird nach und nach das Wort – sollte es sich um den richtigen Buchstaben handeln – dargestellt.

Bedienung
- Das Spiel wird mittels Drück- Drehsteller (DDS), auch Inkrementaldrehgeber genannt, gesteuert.
- Durch drehen nach links scrollt der Buchstaben-Auswahlbalken der sich in der 1. Textzeile gefindet nach links.
- Durch drehen nach rechts scrollt der Buchstaben-Auswahlbalken der sich in der 1. Textzeile gefindet nach rechts.
- Durch drücken des Drehknopfes wird der ausgewählte Buchstabe bestätigt.

Spielverlauf
Es geht darum ein 13-stelliges Nomen der deutschen Sprache zu erraten welches zu Beginn beim Spielstart automatisch aus einer Datenbank von insgesamt 100 Wörtern ermittelt wurde.

Bei jedem Spielstart und nach jedem erfolgreichen oder misslungenen Versuch wird ein neues Wort ermittelt.

Ist der ausgewählte Buchstabe korrekt ertönt ein kurzer Doppelpiep und der ausgewählt Buchstabe wird in dem gesuchten aber noch ausgeblendeten Wort in der 2. Textzeile angezeigt. Der Ausgewählte Buchstabe wird dann im Buchstaben-Auswahlbalken durch ein Leerzeichen ersetzt und somit ist er nicht mehr erneut anwählbar.

Ist der ausgewählte Buchstabe falsch so erklingt ein langezogener Piep und das Galgenmännchen beginnt am Galgen zu baumeln. Der Spieler hat so 5 Versuche das Wort richtig herauszufinden.

Werden alle 13 Buchstaben des Wortes richtig erraten ertont ein dreifacher Piep und das Spiel gibt die Geokoordinaten für das Versteck des Caches preis. Hierbei zeigt das Display repetierend die Koordinaten und den Hinweistext an.

Dieser Modus kann durch erneutes Betätigen oder Drehen des DDS verlassen werden. Danach startet das Spiel erneut.

Hängt das Galgenmännchen vollständig am Galgen so ist nach 5 Versuchen das Spiel automatisch beendet. Nach einem kurzen Trosttext startet das Spiel erneut mit einem neuen Begriff.

Bypassmodus
Für Hilfen und Telefonjoker ist im Spiel noch ein Bypassmodus implementiert. Hierzu muss beim Einschalten des Spiels der DDS-Knopf gedrückt werden. Beim ersten Pieps muss dieser losgelassen werden. Nun betätigt man 5x DDS links, 5x DDS rechts und drück sofort nochmals den DDS Knopf.

Danach wird ein Zähler ausgegeben wie oft der Bypassmodus bereits ausgeführt wurde. Dieser Counter wird persistent im EEPROM abgespeichert so dass der Owner Kontroller darüber hat ob jemand den Bypassmodus entdeckt oder weitergegeben hat und wie oft das schon passiert ist.

Nach der Ausgabe des Counter wechselt das Spiel automatisch in die Ausgabe von Koordinaten und Hinweistext wie im Spielverlauf beschrieben.

Weicht man von der Sequenz zum erreichen des Bypassmodus ab so befindet man sich sofort ohne optische Veränderung direkt im Spiel und der Bypassmodus kann nicht aktiviert werden.

13.6 Der Source-Code zum Projekt Hanging Man

```
'#################################################################################
'
' HangingMan.BAS                                                 Stand 09.07.2014
' --------------                                                 (C) Markus Fulde
'
' Gesamtsoftware Steuerprogramm für das Geocaching-Spiel Hanging Man
'
' Das Projekt verfüg über die folgenden Funktions- und Teilkomponenten:
' - Spannungsversorgung via 9V Batterie und 5V über Festspannungsregler
' - Mikrocontroller ATmega8L (inkl. ISP, RS232 und Reset)
' - LCD-Display 2x16 EA DOG-M 162 von Electronic Assembly
' - Die Helligkeit des Display soll über PWM einstellbar sein
' - Bei Innaktivität soll das Display abgeschaltet werden können
' - Hauptschalter für Spannungsversorgung
' - Drück-Dreh-Geber (inkrementeller Drehgeber mit Tasterfunktion)
' - LED-Anzeigen für Betriebsanzeige und Low-Power Batterie-Überwachung
' - Gehäuse mit Batteriefach und Displayfenster
' - Akustisches Feedback über Piezo Schallwandler
'
'#################################################################################


'--------------------------------------------------------------------------------
' Allgemeines zum Spiel:
'--------------------------------------------------------------------------------
' Galgenmännchen, Galgenraten, Galgenbaum oder auch Galgenmann, Hängemann,
' Hängemännchen oder ein-fach auch Galgen (englisch auch hangman) ist ein
' einfaches Buchstabenspiel.
'
' Spielverlauf
' Der Spieler spielt alleine mit Hilfe der Elektronik. Am Ende bei erfolgreichem
' Spiel wird eine Geokoordinaten für das Finale ausgegeben.
' Der Computer überlegt sich nun ein Wort 13 Buchstaben. Alle Buchstaben des
' ausgedachten Wortes werden durch Striche markiert. Der Geocacher hat nun die
' Aufgabe, die Buchstaben zu erraten. Hierzu wählt er in beliebiger Reihenfolge
' nacheinander einzelne Buchstaben des Alphabets mit dem DDS aus. Der Computer
' prüft die Eingabe und zeigt an, wie oft und an welcher Stelle des Lösungswortes
' der Buchstabe vorkommt. So ergibt sich nach und nach das gesuchte Wort.
' Kommt ein genannter Buchstabe darin jedoch nicht vor beginnt der Computer damit,
' einen Galgen mit einem Gehängten zu zeichnen. Dies geschieht in 5 Etappen
' (bei jeder Fehlfrage kommt ein Teilstrich dazu), so dass der Geocacher 5
' Versuche hat das Wort zu erraten. Hat er dann das Wort noch nicht herausgefunden,
' so hat er verloren und hängt symbolisch am Galgen.
'
' Umlaute werden ae, oe oder ue geschrieben.
' Der Anfangsbuchstabe muss ebenfalls geraten werden.
'--------------------------------------------------------------------------------


'--------------------------------------------------------------------------------
' Anmerkungen zum Programm:
'--------------------------------------------------------------------------------

' [1] Zufallszahlengenerator in BASCOM
' Die Rnd() Funktion Arbeitet Nicht Wirklich Zufällig. Hierzu Sagt Mcv Folgendes:
' The Rnd() Function Returns An Integer / Word And Needs An Internal Storage Of 2 Bytes.
' (___rseed). Each New Call To Rnd() Will Give A New Positive Random Number.
' notice Notice that it is a software based generated number.
' And each time you will restart your program the same sequence will be created.
'
' You can use a different SEED value by dimensioning and assigning ___RSEED yourself:
' Dim ___rseed as word : ___rseed = 10234
' Dim I as word : I = rnd(10)
'
' When your application uses a timer you can assign ___RSEED with the timer value.
' This will give a better random number.
'
' Aus diesem Grund wird hier in der Software beim Auslesen der Batteriespannung
' die SEED-Variable des Zufallszahlengenerators neu gesetzt um mehr
' Zufälligkeit zu erzeugen.
'
' [2] Bypassmodus
' Um das Spiel nicht spielen zu müssen wird quasi als Hintertür ein Bypassmodus
' implementiert der wie folgt erreicht werden kann:
'    - Spiel ausschalten
'    - DDS Drücken / gedrückt halten und Spiel einschalten
```

```
'    - DDS loslassen
'    - DDS 5x links drehen
'    - DDS 5x rechts drehen
'    - DDS erneut kurz betätigen
' Danach wird direkt die finale Geokoordinate an der sich der Cache befindet
' angezeigt. Abweichungen zur beschriebenen Bedienung führen zum sofortigen
' Abbruch der Bypasssequenz und es muss normal gespielt werden.
' Zur Kontrolle ob ein Cacher den Bypassmode findet und dies weiterkommuniziert
' wird ein Bypasscounter geführt der im EEPROM abgelegt wird. Dieser Couner
' wird kurz vor dem Anzeigen der finalen Koordinaten angezeigt. Somit hat man
' die Kontrolle darüber ob jemand dieses Hintertürchen gefunden hat.
' Das Hintertürchen ist als Telefonjoker für Cacher gedacht die erfolglos
' versucht haben das Spiel zu spielen und den Owner kontaktieren.

'-----------------------------------------------------------------------------
' Compilerinstruktionen und Compilerdirektiven
'-----------------------------------------------------------------------------
$regfile = "m8def.dat"                                      ' Definitionsdatei für ATmega128 laden
$crystal = 8000000                                          ' Quarzfrequenz für 16 MHz festlegen

$hwstack = 128                                              ' HW-Stack auf 128 Bytes erweitern
$swstack = 64                                               ' SW-Stack auf 64 Bytes erweitern
$framesize = 80                                             ' Framesize auf 80 Byte festlegen

$baud = 19200                                               ' Baudrate für RS232 Traceausgabe defi-
nieren


'-----------------------------------------------------------------------------
' Allgemeine Zusatzinformatonen zu Programmbeginn
'-----------------------------------------------------------------------------


'-----------------------------------------------------------------------------
' Definition von Ressourcen
'-----------------------------------------------------------------------------

' ----- LED's -----
Alive_pin Alias PinB.0                                      ' GPIO für Alive-LED (für DDR oder In-
put)
Alive Alias PortB.0                                         ' GPIO für Alive-LED (für Output oder
Pullup)

Pwrled_pin Alias PINB.1                                     ' GPIO für Power-LED (für DDR oder In-
put)
Pwrled Alias PORTB.1                                        ' GPIO für Power-LED (für Output oder
Pullup)

' ----- LCD-Display -----
' LCD-Display
Db4_pin Alias PortC.2                                       ' GPIO für LCD Pin4
Db5_pin Alias PortC.3                                       ' GPIO für LCD Pin5
Db6_pin Alias PortC.4                                       ' GPIO für LCD Pin6
Db7_pin Alias PortC.5                                       ' GPIO für LCD Pin7
E_pin Alias PortD.6                                         ' GPIO für LCD E
Rs_pin Alias PortD.7                                        ' GPIO für LCD RS

' ----- SOUND -----
Piezosound_pin Alias Pinb.2                                 ' GPIO für Soundausgabesteuerung (Tran-
sistorstufe)
Piezosound Alias Portb.2                                    ' GPIO für Sound (für Output oder Pul-
lup)

' ----- DDS -----
A_signal_pin Alias Pind.2                                   ' GPIO für DDS Signal A (für DDR oder
Input)
A_signal Alias Portd.2                                      ' GPIO für DDS Signal A (für Output oder
Pullup)

B_signal_pin Alias Pind.4                                   ' GPIO für DDS Signal B (für DDR oder
Input)
B_signal Alias Portd.4                                      ' GPIO für DDS Signal B (für Output oder
Pullup)

Push_signal_pin Alias Pind.3                                ' GPIO für DDS Push (für DDR oder Input)
Push_signal Alias Portd.3                                   ' GPIO für DDS Push (für Output oder
Pullup)
```

```
'--------------------------------------------------------------------------------
' Definition von Konstaten
'--------------------------------------------------------------------------------

' ----- Für Testumgebung bzw. Traceausgaben -----
Const Main_testmodus = 0                                ' Flag für Testmodus Allgemeinsystem
Const Lcd_testmodus = 0                                 ' Flag für Testmodus rund um das LCD-
Display
Const Game_testmodus = 0                                ' Flag für Testmodus der Spielsteuerung
Const Dds_testmodus = 0                                 ' Flag für Testmodus des Drückdrehstel-
lers
Const Bypass_testmodus = 0                              ' Flag für Testmodus des Bypass-Modes
Const Eeprom_testmodus = 0                              ' Flag für Testmodus der Eeprom-Daten


' ----- Allgemeine Systemkonstanten -----

' Tatsächliches Allgemeines
' Const Led_aus = 0
' Const Led_ein = 1

Const Led_aus = 1                                       ' Achtung !! bei STK500 ist Logik ge-
dreht!!
Const Led_ein = 0                                       ' Achtung !! bei STK500 ist Logik ge-
dreht!!

Const False = 0
Const True = 1

Const Pullup_aus = 0
Const Pullup_ein = 1

' Zeitvorgabe für Sekunden-Timer
Const Timervorgabe = 34286                              ' Timer von 1 Sekunden (SekundenTick)


' ----- LCD -----

' Anmerkung: Wert von 35 ist der beste Wert welcher durch Versuche ermittelt wurde!!
Const Lcd_kontrast_default = 35                         ' LCD-Default-Kontrast

' Anmerkung: Wert von 70 ist der beste Wert welcher durch Versuche ermittelt wurde!!
Const Lcd_helligkeit_default = 70                       ' LCD-Default-Helligkeit

Const Lcd_helligkeit_aus = 0                            ' Wert für Display aus
Const Lcd_abschaltzeit = 10                             ' Defaultwert auf 10 Sekunden festgelegt

Const Lcd_beleuchtung_status_ein = 1                    ' Statuswert für LCD Beleuchtung ein
Const Lcd_beleuchtung_status_aus = 0                    ' Statuswert für LCD Beleuchtung aus


' ----- ADC: Batterieüberwachung -----
Const Ubatt_grenzwert = 700                             ' Grenzwert für Batterieüberwachung


' ----- Sound -----
Const Sound_ein = 1                                     ' Wert für GPIO-Wert um Sound einzu-
schalten
Const Sound_aus = 0                                     ' Wert für GPIO-Wert um Sound auszu-
schalten


' ----- DDS -----
Const Dds_none = 0                                      ' Keine Betätigung des DDS
Const Dds_links = 1                                     ' DDS wurde nach links gedreht
Const Dds_rechts = 2                                    ' DDS wurde nach rechts gedreht


' ----- GAME -----
Const Game_anzahl_text = 100                            ' Anzahl der Text im DATA Bereich
Const Game_textmaske = "_____________"

'                                    1         2         3
'                          123456789012345678901234567890123456789
Const Game_auswahltext_default = "uvwxyzabcdefghijklmnopqrstuvwxyzabcdef"
'                                123456789012345678901234567
'                                          1         2

Const Game_startposition_default = 1                    ' Startposition für Buchstabenauswahl
```

```
Const Game_fuellzeichen = 32                                ' Leerzeichen ist Fuellzeichen
' Const Game_fuellzeichen = 95                               ' Unterstrich ist Fuellzeichen

Const Game_cursorzeichen = &B00101010                       ' Zeichen aus Displayzeichentabelle für
Cursorzeichen (Stern)

Const Game_leftcount_max = 5                                ' Anzahl von Linkstix für Bypassmode
Const Game_rightcount_max = 5                               ' Anzahl von Rechtstix für Bypassmode


' ----- EEPROM -----
Const Ee_default_version_value = 1                          ' Version für persistente Daten


'-------------------------------------------------------------------------------
' Definition von Variablen und Datentypen
'-------------------------------------------------------------------------------

' ----- Temporäre Hilfsvariablen -----
Dim Temp_byte_1 As Byte                                     ' Temporäre Byte Variable 1
Dim Temp_byte_2 As Byte                                     ' Temporäre Byte Variable 2

Dim Temp_word_1 As Word                                     ' Temporäre Word Variable 1
Dim Temp_word_2 As Word                                     ' Temporäre Word Variable 2


' ----- System -----
Dim ___rseed As Word                                        ' Manipulation des Zufallszahlengenera-
tors
Dim Sectick_counter As Word                                 ' Globaler Sekundenzähler


' ----- Arbeitsvariablen für Spieleablauf -----
Dim Game_hangman_status As Byte                             ' Arbeitsvariable für Hangman-Status
Dim Game_ablaufstatus As Byte                               ' Arbeitsvariable für System-Status

Dim Game_suchtext As String * 13                            ' Stringvariable für den Suchtext
Dim Game_suchtext_zeichen(14) As Byte At Game_suchtext Overlay

Dim Game_scrolltext As String * 13                          ' Stringvariable für Scrolltext in Zeile
1
Dim Game_scrolltext_zeichen(14) As Byte At Game_scrolltext Overlay       ' Zeichenweise Zugriff auf
Scrolltext

Dim Game_raetseltext As String * 13                         ' Stringvariable für zu suchenden Text
in Zeile 2
Dim Game_raetseltext_zeichen(14) As Byte At Game_raetseltext Overlay       ' Zeichenweise Zugriff
auf Raetseltext

Dim Game_auswahltext As String * 38                         ' Stringvariable für scrollbaren Aus-
wahltext
Dim Game_auswahltext_zeichen(39) As Byte At Game_auswahltext Overlay       ' Zeichenweise Zugriff
auf Auswahltext

Dim Game_buchstaben_index As Byte                           ' Arbeitsvariable für aktuelle Buchsta-
benauswahl

Dim Game_bypassmode_flag As Byte                            ' Arbeitsvariable für Einstieg in
Bypassmodus
Dim Game_bypassmode_leftcounter As Byte                     ' Arbeitsvariable für Bypass Linkscounts
Dim Game_bypassmode_rightcounter As Byte                    ' Arbeitsvariable für Rechtscount


' ----- Variablen für LCD-Display -----
Dim Lcd_kontrastwert As Byte                                ' Arbeitsvariable für Kontrastwert
Dim Lcd_helligkeit As Byte                                  ' Arbeitsvariable für Displayhelligkeit
Dim Lcd_abschaltcounter As Word                             ' Variable für Abschaltzeitpunkt
Dim Lcd_beleuchtung_status As Byte                          ' Arbeitsvariable für Zustand der
Hintergrundbeleuchtugn


' ----- ADC: Batteriespannungsueberwachung -----
Dim Adc_channel As Byte                                     ' Arbeitsvariable für ADC-Kanalauswahl


' ----- DDS -----
Dim Dds_flag As Byte                                        ' Arbeitsvariable zur Behandlung des DDS
Dim Push_flag As Byte                                       ' Arbeitsvariable zur Behandlung des
```

```
Tasters


' ----- EEPROM -----
Dim Ee_data_version_value As Eram Byte At &H0000            ' EEPROM Datenstruktur internes EEPROM -
Init-Index / Version
Dim Ee_data_bypass_counter As Eram Word At &H0001           ' EEPROM Datenstruktur internes EEPROM -
Counter
Dim Ee_version_value As Byte                                ' BYTE Arbeitsvariable für Daten-Version
Dim Ee_bypass_counter As Word                               ' WORD Arbeitsvariable für Bypasscounter


'-------------------------------------------------------------------------------
' Prototyping
'-------------------------------------------------------------------------------

' ----- LCD und Print -----
Declare Sub Lcd_print_hangingman(byval Value As Byte)       ' Funktion zum schrittweisen Aufbau des
Hanging Man


'-------------------------------------------------------------------------------
' Konfiguration und Basiseinstellungen (Projekt und Testumgebung)
'-------------------------------------------------------------------------------

' --------------------------------- CONFIG -------------------------------------

' ----- Timer -----

' Konfiguration eines Timers für 1 Sekunden Timer-Tick (Scheduler und Alive)
Config Timer1 = Timer , Prescale = 256                      ' Timer 1 verwenden
On Timer1 Sekunden_tick                                     ' Interrupt Routine
Timer1 = Timervorgabe
Enable Timer1                                               ' Interrupt für Sekunden-Tack freigeben

' Konfiguration Timer 2 für Hardware-PWM an OC2 (D.7)
Config Timer2 = Pwm , Prescale = 128 , Compare Pwm = Clear Up       ' Timer 2 verwenden
Enable Timer2                                               ' Interrupt für PWM Timer 2 freigeben


' ----- LCD Display -----

' Konfiguration LCD Display
Config Lcdpin = Pin , Db4 = Db4_pin , Db5 = Db5_pin , Db6 = Db6_pin , Db7 = Db7_pin , E = E_pin , Rs
= Rs_pin
Config Lcd = 16 * 2 , Chipset = Dogm162v5                   ' DOG-M Treiber laden
Config Lcdbus = 4                                           ' LCD arbeitet über 4-Bit
Initlcd                                                     ' LCD initialisieren
Waitms 100                                                  ' 100ms nach Init warten
Cursor Off Noblink                                          ' Blinkenden Cursor abschalten

' Definition benutzerdefinierter Zeichen
Deflcdchar 0 , 16 , 16 , 16 , 16 , 16 , 16 , 30 , 30       ' LCD-Zeichen linker Sockel des Galgens
Deflcdchar 1 , 15 , 9 , 10 , 12 , 8 , 8 , 8 , 8            ' LCD-Zeichen linke obere Ecke des Gal-
gens
Deflcdchar 2 , 28 , 4 , 32 , 32 , 32 , 32 , 32 , 32        ' LCD-Zeichen rechter Galgen ohne Männ-
chen
Deflcdchar 3 , 28 , 4 , 4 , 14 , 17 , 17 , 14 , 4          ' LCD-Zeichen rechter Galgen mit Kopf
Deflcdchar 4 , 4 , 4 , 4 , 4 , 32 , 32 , 32 , 32           ' LCD-Zeichen Bauch
Deflcdchar 5 , 4 , 4 , 4 , 4 , 8 , 16 , 16 , 32            ' LCD-Zeichen Bauch mit linkem Beine
Deflcdchar 6 , 4 , 4 , 4 , 4 , 10 , 17 , 17 , 32           ' LCD-Zeichen Bauch mit beiden Beinen
Deflcdchar 7 , 14 , 21 , 21 , 4 , 10 , 17 , 17 , 32        ' LCD-Zeichen kompletter Männchen-Körper

Cls                                                         ' Clear Screen
' Anmerkung: CLS muss gemäß Datenblatt sein um selbstdefinierte Zeichen in den Controller zu über-
nehmen ......


' ----- ADC: Batterieueberwachung -----
' Konfiguration ADC Single-Mode und automatische Prescaler Setting
' Der Single-Mode wir bei BASCOM in Verbindung mit der Funktion GETADC() verwendet
' Der Prescaler teilt den internen Tackt durch 2, 4, 8,16,32,64 or 128 da der ADC
' einen Takt zwischen 50-200 kHz benötigt.
' Das AUTO Feature von BASCOM, setzt automatisch die höchste mögliche Taktrate
Config Adc = Single , Prescaler = Auto , Reference = Avcc
Start Adc


' ------------------------------ Port's und Pin's ---------------------------
```

```
' ----- LED-Konfigurationen -----
Config Alive_pin = Output                                   ' GPIO für Alive-LED ist Output
Config Pwrled_pin = Output                                  ' GPIO für Power-LED ist Output


' ----- Sound ------
Config Piezosound_pin = Output                              ' GPIO für Sound ist Output


' ----- DDS ------
Config A_signal_pin = Input                                 ' GPIO für DDS-A ist Input, PullUp wird
später geschaltet
Config B_signal_pin = Input                                 ' GPIO für DDS-B ist Input, PullUp wird
später geschaltet
Config Push_signal_pin = Input                              ' GPIO für Taster ist Input, PullUp wird
später geschaltet


' ----- DDS -----

' Int 0 für Signal A so konfigurieren dass Interrupt bei fallender Flanke angesprungen wird
Config Int0 = Falling                                       ' Schalter zieht PullUp-Spannung auf GND
On Int0 Dds_interrupt                                       ' Interrupt-Routine
Enable Int0                                                 ' Interrupt freigeben

' Int 1 für den Taster so konfigurieren, dass Interrupt bei fallender Flanke angesprungen wird
' Via Default ist Interrupt aber deaktiviert. Er wird nur zur Abfrage des Tasters bei der Anzeige
' der Geokoordinaten aktiviert.
Config Int1 = Falling                                        ' Schalter zieht PullUp-Spannung auf
GND
On Int1 Push_interrupt                                       ' Interrupt-Routine
Disable Int1                                                 ' Interrupt freigeben


' ------------------------------ Variablen und Werte -------------------------

' ----- System -----
Sectick_counter = 0                                         ' Globaler Sekundenzähler initialisieren


' ----- LED-Konfigurationen -----
Alive = Led_aus                                             ' Alive-LED aus
Pwrled = Led_aus                                            ' LED für Spannungsüberwachung


' ----- Sound -----
Piezosound = Sound_aus                                      ' Sound ausschalten


' ----- LCD-Display -----
Lcd_kontrastwert = Lcd_kontrast_default                     ' Kontrastwert
Lcd_helligkeit = Lcd_helligkeit_default                     ' Displayhelligkleit


' ----- Spielesteuerung -----
Game_hangman_status = 0                                     ' Statusvariable für gezeichneten
HangingMan
Game_ablaufstatus = 0                                       ' Steuervariable für Spielablaufsteue-
rung

Game_bypassmode_leftcounter = 0                             ' Linkscounter initialisieren
Game_bypassmode_rightcounter = 0                            ' Rechtscounter initialisieren


' ----- ADC_ Batteriespannungsueberwachung -----
Adc_channel = 0                                             ' Spannungsteiler zur
Batterispannungsmessung hängt an ADC0


' ----- DDS - Konfiguration -----

' Interne Pull-Ups für die Schalter aktivieren
A_signal = Pullup_ein
B_signal = Pullup_ein
Push_signal = Pullup_ein

' ----- Bypassmodus -----

' An dieser Stelle wird sofort der Status für den Bypassmodus abgefragt.
' Der Bypassmodus kann erreicht werden indem PTT direkt beim Einschalten betätigt wird
```

```
' Danach muss man 5x links drehen, 5x rechts drehen und erneut PTT betätigen

' Ggf Trace schreiben
#if Bypass_testmodus
   Print "Pin-Pegel = " ; Push_signal_pin
#endif

' Wurde bei Systemstart PTT als Triggersignal für den Einstieg in den Bypassmode erkannt?
If Push_signal_pin = True Then

   ' PTT nicht betätigt -> Bypassmodus nicht schalten
   Game_bypassmode_flag = False

   ' Ggf Trace schreiben
   #if Bypass_testmodus
      Print "Bypassmode False"
   #endif

Else                                                    ' Wurde Bypassmode getriggert?

   ' PTT betätigt -> Bypassmodus schalten
   Game_bypassmode_flag = True

   ' Ggf Trace schreiben
   #if Bypass_testmodus
      Print "Bypassmode True"
   #endif

End If                                                  ' Wurde Bypassmode getriggert?


' Arbeitsvariable vorinitialisieren
Dds_flag = Dds_none                                     ' ... steuert DDS-Drehung erkannt
Push_flag = False                                       ' ... steuert Push-Taster erkannt


'-------------------------------------------------------------------------------
' Und los gehts, hier noch die Restarbeiten
'-------------------------------------------------------------------------------

' ----- Freigabe aller Interrupts ----
Enable Interrupts                                       ' Damit auch Empfang von Daten über
Buffer

' --------------------------------- Gosub's ---------------------------------

Gosub Lcd_kontrast_set                                  ' LCD Kontrast einstellen

Gosub Lcd_helligkeit_set                                ' Displayhelligkeit einstellen
Lcd_beleuchtung_status = Lcd_beleuchtung_status_ein     ' Hintergrundbeleuchtung ist EIN

Gosub Adc_check_batterie                                ' Batteriespannung überprüfen

Gosub Ee_init                                           ' Persistente Daten aus EEPROM auslesen


' ###############################################################################
'
'                            Hauptprogramm ConvCtrl
'
' ###############################################################################


' In Abhängigkeit der Konstante Traceausgabe schreiben
#if Main_testmodus
   Print "** OK, let's GO **"
#endif


' Begrüßungstext 1 ausgeben
'                   1234567890123456
Locate 1 , 1 : Lcd " Willkommen zum"                    ' Text Zeile 1
Locate 2 , 1 : Lcd "Geocaching Spiel"                   ' Text Zeile 2

Gosub Sound_piep_400ms                                  ' 400 ms Piepton ausgeben

Wait 3                                                  ' Wartezeit bis nächste Anzeige
Cls                                                     ' Bildschirm löschen

'                   1234567890123456
```

```
Locate 1 , 1 : Lcd " Galgenm" ; Chr(132) ; "nnchen"          ' Text Zeile 1
Locate 2 , 1 : Lcd "     GC585BM    "                        ' Text Zeile 2

Gosub Sound_piep_400ms                                       ' 400 ms Piepton ausgeben

Wait 3                                                       ' Wartezeit bis nächste Anzeige

' Logik für Abschaltung Hintergrundbeleuchtung "aufziehen"
Lcd_abschaltcounter = Sectick_counter + Lcd_abschaltzeit


' ---------------------------------------------
' ----- Hier ist die Programmhauptschleife -----
' ---------------------------------------------

' Erklärung Spielstati - Game_ablaufstatus:
' 0 = Lösche Bildschirm, zeichne Basisgalgen, erzeuge Zufallstext, lade Auswahltext,
'     lade Textinhalte in Arbeitsstring und zeige diese auf dem Display an
' 1 = Normales Spiels
' 2 = Verloren-Behandlung, Verlorentexte ausgeben, Spielstati zurücksetzen
' 3 = Jippi, gewonnen, Geokoordinate ausgeben
'
' Achtung: Status 1 ist normaler Status der im Wesentlichen den DDS auswertet
'          Hier läuft die Do-While-Schleife die meiste Zeit!!
'

Do                                                           ' Hauptschleife

   ' ------------------------------ Systemaufgaben ----------------------------

   ' Prüfen ob Sekundencounter für Hintergrundbeleuchtung abgelaufen ist
   If Lcd_abschaltcounter = Sectick_counter Then

      ' JA -> Hintergrundbeleuchtung abschalten und Strom sparen
      Gosub Lcd_beleuchtung_aus
      Lcd_beleuchtung_status = Lcd_beleuchtung_status_aus    ' Hintergrundbeleuchtung ist AUS

      ' Durch Verringerung des Counters wird verhindert dass Funktion mehrfach angesprungen wird.
      Decr Lcd_abschaltcounter

   End If                                                    ' LCD Abschaltzeitpunkt erreicht

   ' Prüfen ob der Bildschirm abgeschaltet ist und der DDS betätigt wurden
   If Lcd_beleuchtung_status = Lcd_beleuchtung_status_aus Then

      ' JA -> wurde Drehung erkannt?
      If Dds_flag > Dds_none Then

         'JA -> Beleuchtung einschalten und DDS ignorieren
         Gosub Lcd_beleuchtung_ein                           ' Hintergrundbeleuchtung einschalten
         Lcd_beleuchtung_status = Lcd_beleuchtung_status_ein       ' Hintergrundbeleuchtung ist EIN
         Dds_flag = Dds_none                                 ' Flag korrigieren

        ' Logik für Abschaltung Hintergrundbeleuchtung erneut "aufziehen"
        Lcd_abschaltcounter = Sectick_counter + Lcd_abschaltzeit

      End If                                                 ' DDS betätigt

   End If                                                    ' Hintergrundbeleuchtung abgeschaltet


   ' -------------------------- Spielsteuerung ------------------------------

   ' ==== Spiel Vorbereitung ====

   ' Prüfen ob Spiel neu gestartet werden muss und Anzeige neu aufsetzt
   If Game_ablaufstatus = 0 Then
      ' JA -> Bildschirm löschen und Hanginman neu aufsetzen
      Cls
      Call Lcd_print_hangingman(game_hangman_status )        ' Maennchen gemaess Status zeichnen

      ' Zufallstext ermitteln
      Gosub Game_suchtext_select

      ' Auswahltext in Arbeitsvariable laden (gesamter Zeichenvorrat für Zeile 1)
      Game_auswahltext = Game_auswahltext_default

      ' Ratseltext zurücksetzen und Text auf Display anzeigen (Zeile 2)
      Game_raetseltext = Game_textmaske
      Gosub Lcd_print_raetseltext
```

```
    ' Scrollzeile für Buchstaben vorbereiten (Zeile 1)
    Game_buchstaben_index = Game_startposition_default
    Gosub Game_update_scrolltext
    Gosub Lcd_print_scrolltext

    ' Spielstatus weiterschalten
    Game_ablaufstatus = 1

End If                                                  ' Spiel Vorbereitung


' ==== normales Spiel => Auswertung des DDS ====

If Game_ablaufstatus = 1 Then

    ' ----- DDS auswerten -----
    Select Case Dds_flag

       Case Dds_links : Gosub Dds_links_verarbeiten
       Case Dds_rechts : Gosub Dds_rechts_verarbeiten

    End Select

End If                                                  ' normales Spiel


' ==== Leider verloren ====

If Game_ablaufstatus = 2 Then

    ' ----- Fehlerszenario -----

    Wait 1

    Gosub Sound_piep_1s
    Waitms 500
    Gosub Sound_piep_1s
    Waitms 500
    Gosub Sound_piep_1s
    Waitms 500

    '                   1234567890123456
    Locate 1 , 1 : Lcd "!!!  Schade  !!!"
    Locate 2 , 1 : Lcd "Leider  verloren"

    Wait 3

    '                   1234567890123456
    Locate 1 , 1 : Lcd "Probiere es doch"
    Locate 2 , 1 : Lcd "einfach nochmals"

    Wait 3

    Game_ablaufstatus = 0                               ' Gameablaufsteuerung zurücksetzen
    Game_hangman_status = 0                             ' Hangmanstatus zurücksetzen

    ' Logik für Abschaltung Hintergrundbeleuchtung erneut "aufziehen"
    Lcd_abschaltcounter = Sectick_counter + Lcd_abschaltzeit

End If                                                  ' Leider verloren


' ==== Jippi gewonnen ====

If Game_ablaufstatus = 3 Then

    Wait 1

    Gosub Sound_piep_1s
    Waitms 500
    Gosub Sound_piep_1s
    Waitms 500
    Gosub Sound_piep_1s
    Waitms 500

    '                   1234567890123456
    Locate 1 , 1 : Lcd "!! Herzlichen !!"
    '                  "   Glückwunsch
    Locate 2 , 1 : Lcd "!  Gl" ; Chr(129) ; "ckwunsch !"
```

```
      Wait 3

      '                   1234567890123456
      Locate 1 , 1 : Lcd "Du hast gewonnen"
      Locate 2 , 1 : Lcd "und findest den "

      Wait 3

      '                   1234567890123456
      Locate 1 , 1 : Lcd "Cache bei fol-  "
      Locate 2 , 1 : Lcd "genden Koords.. "

      Temp_byte_1 = True                                    ' Arbeitsvaraible für Schleifenabbruch-
kriterium Hinweistexte

      Wait 3

      ' Ab hier auch Interrupt 1 für Taster zulassen.
      ' Interrupt wird wieder in der ISR disabled
      Push_flag = False                                     ' Flag für Taster zurücksetzen
      Gifr.intf1 = 1                                        ' Anstehender Interrupt löschen
      Enable Int1                                           ' Interrupts freigeben

      Do

         '                   1234567890123456
         Locate 1 , 1 : Lcd "N48" ; Chr(223) ; " 99.999     "
         Locate 2 , 1 : Lcd "E009" ; Chr(223) ; " 99.999    "

         Wait 3

         '                   1234567890123456
         Locate 1 , 1 : Lcd "Hinweistext.1111"
         Locate 2 , 1 : Lcd "Hinweistext.2222"
         Wait 3

         ' Prüfe ob Schleife für Hinweistext verlassen werden kann
         If Dds_flag <> Dds_none Or Push_flag = True Then

            ' JA, alle Flags zurücksetzen

            Dds_flag = Dds_none
            Push_flag = False
            Temp_byte_1 = False

         End If                                             ' Abbruchbedingung für Hinweistext

      Loop Until Temp_byte_1 = False

      ' Spielstatus weiterschalten, Hagmanstatus zurückschalten und nochmals kurz Piepsen
      Game_hangman_status = 0
      Game_ablaufstatus = 4
      Gosub Sound_piep_400ms                                ' 400 ms Piepton ausgeben


      ' Logik für Abschaltung Hintergrundbeleuchtung erneut "aufziehen"
      Lcd_abschaltcounter = Sectick_counter + Lcd_abschaltzeit

   End If                                                   ' gewonnen


   ' ==== OK, Du magst nochmals spielen ====

   If Game_ablaufstatus = 4 Then

      '                   1234567890123456
      Locate 1 , 1 : Lcd "Ok Du magst also"
      Locate 2 , 1 : Lcd "nochmal spielen!"

      Gosub Sound_piep_400ms

      Waitms 2600

      '                   1234567890123456
      Locate 1 , 1 : Lcd "Na dann los und "
      Locate 2 , 1 : Lcd "viel Gl" ; Chr(129) ; "ck!!!!!!"

      Wait 3
```

```
      ' Spielstatus zurückschalten
      Game_ablaufstatus = 0

      ' Logik für Abschaltung Hintergrundbeleuchtung erneut "aufziehen"
      Lcd_abschaltcounter = Sectick_counter + Lcd_abschaltzeit

   End If                                                  ' OK nochmals


   ' PTT-Taste des DDS auswerten
   Debounce Push_signal_pin , 0 , Dds_taste_erkannt , Sub

Loop                                                       ' Hauptschleife

'## End Hauptprogramm ###############################################################

End


'*********************************************************************************
' Interruptroutinen
'*********************************************************************************

'---------------------------------------------------------------------------------
' Interrupt-Service-Routine (Timer1): Sekunden_tick
'
' Routine zur Auswertung des Timer Interrupts
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' Sectick_counter = Variable für den Sekundencounter
'---------------------------------------------------------------------------------
Sekunden_tick:

   ' ----- Programmcode -----

   Timer1 = Timervorgabe                                   ' Timer neu laden
   Toggle Alive                                            ' Alive-LED toggeln lassen

   ' Timervariable erhöhen
   Incr Sectick_counter

Return
'-- End Sekunden_tick  ---------------------------------------------------------


'---------------------------------------------------------------------------------
' Interrupt-Service-Routine (External Interrupt 0): Dds_interrupt
'
' Routine zur Auswertung des INT0 Interrupts für DDS-Bewegung
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' Dds_flag = Variable für den Zustand (Drehrichtung) des DDS
'---------------------------------------------------------------------------------
Dds_interrupt:

   ' ----- Programmcode -----

   ' Erkennung der Drehrichtung
   If A_signal_pin = B_signal_pin Then

      ' Links herum
      Dds_flag = Dds_links

   Else                                                    ' Erkennung der Drehrichtung

      ' Rechts herum
      Dds_flag = Dds_rechts

   End If                                                  ' Erkennung der Drehrichtung

Return
'-- End Dds_interrupt -----------------------------------------------------------
```

```
'--------------------------------------------------------------------------------
' Interrupt-Service-Routine (External Interrupt 1): Push_interrupt
' Routine zur Auswertung des INT0 Interrupts für DDS-Bewegung
'--------------------------------------------------------------------------------
Push_interrupt:

   ' ----- Programmcode -----

   Push_flag = True
   Disable Int1

Return
'-- End Push_interrupt -------------------------------------------------------


'********************************************************************************
' Subroutinen
'********************************************************************************

' ***************
' * LCD-Display *
' ***************

'--------------------------------------------------------------------------------
' LCD - Subroutine: Lcd_print_hangingman
'
' Subroutine schreibt je nach uebergebenem Status den Zustand des Hanging Man
' auf das Display
'    Status 0: Galgen ohne Maennchen
'    Status 1: Galgen mit Kopf
'    Status 2: Galgen mit Kopf und Rumpf
'    Status 3: Galgen mit Kopf, Rumpf und linken Bein
'    Status 4: Galgen mit Kopf, Rumpf, linken und rechten Bein
'    Status 5: Galgen mit kompletten Körper
'
' Parameter:    Value = Step n des Hanging-Man
' Rückgabewert: keine
'
' Globale Variable:
' --
'--------------------------------------------------------------------------------
Sub Lcd_print_hangingman(byval Value As Byte)

   ' ----- Programmcode -----

   ' Galgen zeichnen
   Locate 1 , 1 : Lcd Chr(1)
   Locate 2 , 1 : Lcd Chr(0)

   Select Case Value
      Case 0 :
         Locate 1 , 2 : Lcd Chr(2)
      Case 1 :
         Locate 1 , 2 : Lcd Chr(3)
      Case 2 :
         Locate 1 , 2 : Lcd Chr(3)
         Locate 2 , 2 : Lcd Chr(4)
      Case 3 :
         Locate 1 , 2 : Lcd Chr(3)
         Locate 2 , 2 : Lcd Chr(5)
      Case 4 :
         Locate 1 , 2 : Lcd Chr(3)
         Locate 2 , 2 : Lcd Chr(6)
      Case 5 :
         Locate 1 , 2 : Lcd Chr(3)
         Locate 2 , 2 : Lcd Chr(7)
   End Select                                              ' Case Value

End Sub
'-- End Lcd_Print_hanginman -----------------------------------------------


'********************************************************************************
' GOSubroutinen
'********************************************************************************

' ***************
' * LCD-Dsiplay *
' ***************
```

```
'-------------------------------------------------------------------------------
' LCD - Gosub-Routine: Lcd_contrast_set
'
' Routine berechnen neue Kontrastwerte und steuert direkt den
' Kontroller des Display an.
' Aufgrund von 6 Bit sind nur Kontrastwerte zwischen 0 und 63 möglich!
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' Lcd_kontrastwert = Kontrastwert für Display
'-------------------------------------------------------------------------------
Lcd_kontrast_set:                                       ' Kontrasteinstellung Display

   ' ----- Programmcode -----

   ' Verarbeitung des Kontrastwertes für High-Byte und Low-Byte
   Temp_byte_1 = Lcd_kontrastwert And &B00001111
   Temp_byte_1 = Temp_byte_1 + &B01110000

   Temp_byte_2 = Lcd_kontrastwert
   Shift Temp_byte_2 , Right , 4
   Temp_byte_2 = Temp_byte_2 And &B00000011
   Temp_byte_2 = Temp_byte_2 + &B01010100

   ' Instruction Table 1 einstellen [0,1]
   _temp1 = &B00101001
   !rCall _Lcd_control

   ' Tempvar_1 = &B0111xxxx für Kontrast Set Instruction Table 1 - Low Byte
   _temp1 = Temp_byte_1
   !rCall _Lcd_control

   ' Temovar_2 = &B010101xx für Kontrast Set Instruction Table 1 - High Byte
   _temp1 = Temp_byte_2
   !rCall _Lcd_control

   ' Zurückschalten auf Instruction Table 0 [0,0]
   _temp1 = &B00101000
   !rCall _Lcd_control

Return
'-- End Lcd_kontrast_set -------------------------------------------------------


'-------------------------------------------------------------------------------
' LCD - Gosub-Routine: Lcd_helligkeit_set
'
' Routine setzt den Helligkeitswert aus der globalen Variable Lcd_helligkeit in
' das Controllregister
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' Lcd_helligkeit = Helligkeitswert für PWM
'-------------------------------------------------------------------------------
Lcd_helligkeit_set:

   ' ----- Programmcode -----

   ' Variable in Timer-Count-Register laden
   Ocr2 = Lcd_helligkeit

Return
'-- End Ir_set_helligkeit -------------------------------------------------------


'-------------------------------------------------------------------------------
' LCD - Gosub-Routine: Lcd_beleuchtung_aus
'
' Routine schaltet die LCD-Hintergrundbeleuchtung aus
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' --
'-------------------------------------------------------------------------------
```

```
Lcd_beleuchtung_aus:

   ' ----- Programmcode -----

   Lcd_helligkeit = Lcd_helligkeit_aus
   Gosub Lcd_helligkeit_set

   #if Lcd_testmodus
      Print "Beleuchtung aus"
   #endif

Return
'-- End Lcd_beleuchtung_aus ------------------------------------------------


'--------------------------------------------------------------------------------
' LCD - Gosub-Routine: Lcd_beleuchtung_ein
'
' Routine schaltet die LCD-Hintergrundbeleuchtung ein
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' --
'--------------------------------------------------------------------------------
Lcd_beleuchtung_ein:

   ' ----- Programmcode -----

   Lcd_helligkeit = Lcd_helligkeit_default
   Gosub Lcd_helligkeit_set

   #if Lcd_testmodus
      Print "Beleuchtung ein"
   #endif

   ' Kurzes Piep ausgeben
   Gosub Sound_piep_100ms

Return
'-- End Lcd_beleuchtung_ein -------------------------------------------------


'--------------------------------------------------------------------------------
' LCD - Gosub-Routine: Lcd_print_raetseltext
'
' Routine gibt den bereits enträtselten Text in Zeile 2 mit 13 Zeichen aus
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' Game_raetseltext = Textstring für Raetseltext
'--------------------------------------------------------------------------------
Lcd_print_raetseltext:

   ' ----- Programmcode -----

   ' 1234567890123456
   '    xxxxxxxxxxxxx
   Locate 2 , 4 : Lcd Game_raetseltext

Return
'-- End Lcd_print_raetseltext -----------------------------------------------


'--------------------------------------------------------------------------------
' LCD - Gosub-Routine: Lcd_print_scrolltext
'
' Routine gibt den Scroll-Text der Buchstaben in Zeile 1 mit 13 Zeichen aus
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' Game_scrolltext  = Textstring gescrollte Buchstaben
'--------------------------------------------------------------------------------
Lcd_print_scrolltext:
```

```
   ' ----- Programmcode -----

   ' 1234567890123456
   '    xxxxxxxxxxxx
   Locate 1 , 4 : Lcd Game_scrolltext

Return
'-- End Lcd_print_raetseltext ------------------------------------------------


' **********************
' * ADC-Batteriestatus *
' **********************

'-------------------------------------------------------------------------------
' ADC - Gosub-Routine: Adc_check_batterie
'
' Routine liest den Spannungswert aus dem AD-Wandler, prüft gegen eine
' voreingestellte Grenze und schaltet ggf. die Power-LED als Warnsignal ein.
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' Temp_word_1     = temporäre WORD Arbeitsvariable
' adc_channel     = Variable beinhaltet Kanalnummer des ADC
' Ubatt_grenzwert = Konstante für Grenzwert
'-------------------------------------------------------------------------------
Adc_check_batterie:

   ' ----- Programmcode -----

   ' ADC-Wert lesen
   Temp_word_1 = Getadc(adc_channel)

   ' Prüfen auf Unterspannung
   If Temp_word_1 < Ubatt_grenzwert Then

      ' Wenn Unterspannung erkannt dann LED dauerhaft einschalten
      Pwrled = Led_ein

   End If                                                  ' Unterspannung?

   ' Zufallszahlengenerator "verdrehen" :-) Erklärung siehe ganz oben in der Einleitung!!
   ___rseed = Temp_word_1

Return
'-- End Adc_check_batterie -----------------------------------------------------


' ***********************
' DDS: Drückdrehsteller *
' ***********************

'-------------------------------------------------------------------------------
' DDS - Gosub-Routine: Dds_links_verarbeiten
'
' Routine verarbeitet das Drehen des DDS nach links
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' Viele ... (zu viele um hier einzeln beschrieben zu erden!)
'-------------------------------------------------------------------------------
Dds_links_verarbeiten:

   ' ----- Programmcode -----

   ' Bearbeitungsflag zurücksetzen
   Dds_flag = Dds_none

   ' Buchstabenzeiger verringern und Grenzwerte überprüfen
   Decr Game_buchstaben_index
   If Game_buchstaben_index = 0 Then

      Game_buchstaben_index = 26

   End If                                                  ' Grenzwert erreicht
```

```
   ' Scrollbaren Buchstabenasuwahltext auf LCD-Display aktualisieren
   Gosub Game_update_scrolltext
   Gosub Lcd_print_scrolltext

   ' Logik für Abschaltung Hintergrundbeleuchtung erneut "aufziehen"
   Lcd_abschaltcounter = Sectick_counter + Lcd_abschaltzeit

   ' Ggf. Debugausgaben schreiben
   #if Dds_testmodus
      Print "DDS links - Index = " ; Game_buchstaben_index
   #endif

   ' Prüfung ob bypass aktiv, wenn ja verarbeiten
   If Game_bypassmode_flag = True Then

      ' JA, Bypassmode aktiv
      Incr Game_bypassmode_leftcounter

      ' Prüfen ob Linkscounter zu hoch, wenn ja Bypassmodus deaktivieren
      If Game_bypassmode_leftcounter > Game_leftcount_max Then

         ' Ja, Bypassmodus verlassen
         Game_bypassmode_leftcounter = 0
         Game_bypassmode_flag = False

         ' ggf. Trace ausgeben
         #if Bypass_testmodus
            Print "Bypass zuruecksetzen"
         #endif

      End If                                                ' Prüfung Linkscounter

      ' ggf. Trace ausgeben
      #if Bypass_testmodus
         Print "Linkscounter = " ; Game_bypassmode_leftcounter
      #endif

   End If                                                   ' Behandlung Bypassmode

Return
'-- End Dds_links_verarbeiten ------------------------------------------------


'--------------------------------------------------------------------------------
' DDS - Gosub-Routine: Dds_Dds_rechts_verarbeiten
'
' Routine verarbeitet das Drehen des DDS nach links
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' Viele ... (zu viele um hier einzeln beschrieben zu erden!)
'--------------------------------------------------------------------------------
Dds_rechts_verarbeiten:

   ' ----- Programmcode -----

   ' Bearbeitungsflag zurücksetzen
   Dds_flag = Dds_none

   ' Buchstabenzeiger erhöhen und Grenzwerte überprüfen
   Incr Game_buchstaben_index
   If Game_buchstaben_index = 27 Then

      Game_buchstaben_index = 1

   End If                                                   ' Grenzwert erreicht?

   ' Scrollbaren Buchstabenasuwahltext auf LCD-Display aktualisieren
   Gosub Game_update_scrolltext
   Gosub Lcd_print_scrolltext

   ' Logik für Abschaltung Hintergrundbeleuchtung erneut "aufziehen"
   Lcd_abschaltcounter = Sectick_counter + Lcd_abschaltzeit

   ' Ggf. Debugausgaben schreiben
   #if Dds_testmodus
      Print "DDS rechts - Index = " ; Game_buchstaben_index
   #endif
```

```
   ' Prüfung ob bypass aktiv UND Linkscounter auf Max-Value, wenn ja verarbeiten
   If Game_bypassmode_flag = True And Game_bypassmode_leftcounter = Game_leftcount_max Then

      ' JA, Bypassmode aktiv
      Incr Game_bypassmode_rightcounter

      ' Prüfen ob Rightcounter zu hoch, wenn ja Bypassmodus deaktivieren
      If Game_bypassmode_rightcounter > Game_rightcount_max Then

         ' Ja, Bypassmodus verlassen
         Game_bypassmode_rightcounter = 0
         Game_bypassmode_flag = False

         ' ggf. Trace ausgeben
         #if Bypass_testmodus
            Print "Bypass zuruecksetzen"
         #endif

      End If                                                ' Prüfung Linkscounter

      ' ggf. Trace ausgeben
      #if Bypass_testmodus
         Print "Rechtscounter = " ; Game_bypassmode_rightcounter
      #endif

   Else                                                     ' Behandlung Bypassmode

      ' Bypassmodus verlassen
      Game_bypassmode_rightcounter = 0
      Game_bypassmode_flag = False

      ' ggf. Trace ausgeben
      #if Bypass_testmodus
         Print "Bypass zuruecksetzen"
      #endif

   End If                                                   ' Behandlung Bypassmode

Return
'-- End Dds_rechts_verarbeiten ------------------------------------------------


'--------------------------------------------------------------------------------
' DDS - Gosub-Routine: Dds_Taste_erkannt
'
' Routine gibt nur Traceausgabe aus, dass Taste erkannt wurde
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' Viele ... (zu viele um hier einzeln beschrieben zu erden!)
'--------------------------------------------------------------------------------
Dds_taste_erkannt:

   ' ----- Programmcode -----

   ' Ggf. Debugausgaben schreiben
   #if Dds_testmodus
      Print "DDS gedueckt"
   #endif

   ' Prüfen ob der Bildschirm abgeschaltet ist und der DDS betätigt wurden
   If Lcd_beleuchtung_status = Lcd_beleuchtung_status_aus Then

      'JA -> Beleuchtung einschalten und DDS ignorieren
      Gosub Lcd_beleuchtung_ein                         ' Hintergrundbeleuchtung einschalten
      Lcd_beleuchtung_status = Lcd_beleuchtung_status_ein       ' Hintergrundbeleuchtung ist EIN

   Else                                                 ' Bildschirm abgeschaltet?

      ' Prüfen ob Bypassmodus aktiv und Triggerbedingungen erfüllt
      If Game_bypassmode_flag = True Then

         ' Prüfen ob Bypassbedingungen erfüllt sind
         If Game_bypassmode_leftcounter = Game_leftcount_max And Game_bypassmode_rightcounter =
Game_rightcount_max Then

            ' JA, Bedingungen erfüllt, nun können Koordinaten auch ohne Spiel angezeigt werden
```

```
            ' ggf. Trace ausgeben
            #if Bypass_testmodus
               Print "Bypass ausloesen, Koordinaten anzeigen"
            #endif

            ' Gamestatus setzen und fertig
            Game_ablaufstatus = 3

            ' Counter für Bypassmode erhöhen und in EEPROM speichern
            Incr Ee_bypass_counter
            Ee_data_bypass_counter = Ee_bypass_counter

            ' Aktuelle Counter Value auf Display ausgeben
            '                  1234567890123456
            Cls
            Locate 1 , 1 : Lcd "Bypasscounter:"
            Locate 2 , 1 : Lcd Ee_bypass_counter

         End If                                              ' Bypassbedingungen erfüllt?

         ' In jedem Fall Counter und Flags wieder zurücksetzen
         Game_bypassmode_flag = False
         Game_bypassmode_leftcounter = 0
         Game_bypassmode_rightcounter = 0

         ' ggf. Trace ausgeben
         #if Bypass_testmodus
            Print "Bypassflags und Counter zuruecksetzen"
         #endif

      Else                                                   ' Bypassmode?

         ' NEIN -> Normale PTT Verarbeitung und Auswertung der Eingabe

         ' Ggf. Trace schreiben
         #if Game_testmodus
            Print "Auswertung PTT"
         #endif

         ' Prüfen ob Zeichen schon ausgewählt wurde und Buchstabe mit Unterstrich ausge-x-t
         ' If Game_scrolltext_zeichen(7) <> Game_fuellzeichen Then
         If Game_scrolltext_zeichen(7) <> Game_cursorzeichen Then

            ' JA -> Zeichen ist gültig

            #if Game_testmodus
               Print "Zeichen ist gueltig : " ; Chr(game_scrolltext_zeichen(7) )
               Print "Game_suchtext   : " ; Game_suchtext
               Print "Game_scrolltext : " ; Game_scrolltext
            #endif

            ' Nun im Rätseltext nach dem Buchstaben suchen und ggf. Buchstaben in Suchtext eintragen

            Temp_byte_2 = False

            For Temp_byte_1 = 1 To 13 Step 1

               If Game_suchtext_zeichen(temp_byte_1) = Game_scrolltext_zeichen(7) Then

                  ' Zeichen gefunden
                  Temp_byte_2 = True
                  Game_raetseltext_zeichen(temp_byte_1 ) = Game_scrolltext_zeichen(7)

               End If                                        ' Suchtextzeichen = Scrolltextzeichen?

            Next                                             ' Schleife über alle Zeichen

            ' Nun prüfen ob passende Zeichen gefunden wurden

            If Temp_byte_2 = True Then

               ' JA

               Gosub Lcd_print_raetseltext

               ' Kurzes Piep ausgeben
               Gosub Sound_piep_100ms
               Waitms 100
               Gosub Sound_piep_100ms
```

```
            Else                                                ' Zeichen gefunden

               ' nein

               Incr Game_hangman_status
               #if Game_testmodus
                  Print "Fehler Hangman status : " ; Game_hangman_status
               #endif

               Call Lcd_print_hangingman(game_hangman_status ) ' Maennchen gemaess Status zeichnen

               ' Prüfen ob Spiel schon zuende

               If Game_hangman_status = 5 Then

                  ' Spielstatus weiterschalten
                  Game_ablaufstatus = 2

                  ' Rätseltext übernehmen und anzeigen da Spiel eh zuende
                  Game_raetseltext = Game_suchtext
                  Gosub Lcd_print_raetseltext

               End If                                           ' Spiel zuende

               ' Piepston ausgeben und warten
               Gosub Sound_piep_1s
               ' Wait 1

            End If                                              ' Fehler gemacht

            ' Gesamten Auswahltext nach dem ausgewählten Zeichen durchsuchen und mit Füllzeichen
besetzen
            ' ==> Zeichen kann nicht mehr ausgewählt werden.

            ' Gesuchtes Zeichen ist kleingeschrieben, daher muss scrolltext_zeichen angepasst werden
            Temp_byte_2 = Game_scrolltext_zeichen(7) + 32

            ' Ggf. Debugausgaben schreiben
            #if Dds_testmodus
               Print "Rücksetzzeichen = " ; Chr(temp_byte_2)
            #endif

            For Temp_byte_1 = 1 To 38 Step 1

               If Game_auswahltext_zeichen(temp_byte_1 ) = Temp_byte_2 Then

                  ' Zeichen gefunden
                  Game_auswahltext_zeichen(temp_byte_1 ) = Game_fuellzeichen

                  ' Ggf. Debugausgaben schreiben
                  #if Dds_testmodus
                     Print "Zeichen gefunden"
                  #endif

               End If                                       ' Textposition entspricht Füllzeichen

            Next                                            ' Schleife über alle Zeichen

            ' Scrolltext updaten !!! nicht vergessen
            Gosub Game_update_scrolltext
            Gosub Lcd_print_scrolltext

            #if Game_testmodus
               Print "Game_raetseltext : " ; Game_raetseltext
            #endif

            ' Nun prüfen ob bisheriger Text dem gesuchen Wort entspricht

            If Game_raetseltext = Game_suchtext And Game_hangman_status < 5 Then

               ' JA

               #if Game_testmodus
                  Print "Jippi gewonnen"
               #endif

               ' Spielstatus weiterschalten
               Game_ablaufstatus = 3
```

```
            End If                                              ' Rätseltext=Suchtext & Status<5

         End If                                                 ' Ungültiges Zeichen

      End If                                                    ' Bypassmode?

      ' Logik für Abschaltung Hintergrundbeleuchtung erneut "aufziehen"
      Lcd_abschaltcounter = Sectick_counter + Lcd_abschaltzeit

   End If                                                       ' Bypassmode=True & leftcoun-
ter=leftcount_max

   Return
'-- End Dds_Taste_erkannt ------------------------------------------------------


' ********
' * GAME *
' ********

'-------------------------------------------------------------------------------
' GAME - Gosub-Routine: Game_suchtext_select
'
' Routine ermittelt via Zufallszahl den im Spiel zu suchenden Text aus dem
' DATA Bereich.
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' Game_suchtext = Arbeitsvariable in dem Suchtext zur Verfügung gestellt wird.
'-------------------------------------------------------------------------------
Game_suchtext_select:

   ' ----- Programmcode -----

   ' Zufallszahl ermitteln
   Temp_byte_1 = Rnd(game_anzahl_text )                         ' Random bei z.B. 40 = 0-39

   ' Suchtext auslesen
   Game_suchtext = Lookupstr(temp_byte_1 , Worte_data)

   ' Falls Flag für Debuging gesetzt debugausgaben ausgeben
   #if Game_testmodus
      Print "Index : " ; Temp_byte_1
      Print "Text  : " ; Game_suchtext
   #endif

Return
'-- End Game_suchtext_select ---------------------------------------------------


'-------------------------------------------------------------------------------
' GAME - Gosub-Routine: Game_update_scrolltext
'
' Routine überträgt aus dem Gesamtstring game_auswahltext den für die Ausgabe
' auf dem Display relevanten Teil in die globale Variable game_scrolltext.
' Dabei der der mittlerste Buchstabe welche ausgewählt werden kann ggf.
' als Großbuchstabe dargestellt.
' Wurde der Buchstabe bereits ausgewählt und ist er als Leerzeichen gekenn-
' zeichnet wird der Buchstabe ignoriert.
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' Game_scrolltext       = Teilstring für die Ausgabe auf dem Display
' Game_auswahltext      = Gesamtstring auf dem gearbeitet wird
' Game_buchstaben_index = Index auf aktuelle A-Z Buchstabenposition
'-------------------------------------------------------------------------------
Game_update_scrolltext:

   ' ----- Programmcode -----

   '          1         2         3
   ' 12345678901234567890123456789012345678
   ' uvwxyzabcdefghijklmnopqrstuvwxyzabcdef
   '       abcdefghijklmnopqrstuvwxyz
   '       12345678901234567890123456
   '                1         2
```

```
   ' Zeile 1 ist Scrolltext
   ' Zeile 2 ist Raetseltext

   Game_scrolltext = Mid(game_auswahltext , Game_buchstaben_index , 13 )

   ' Überprüfen ob Zeichen ein NULL-gesetztes Zeichen ist, also dass der Buchstabe
   ' schon mal ausgewählt wurde. Wenn NEIN Buchstaben GROSS setzen
   If Game_scrolltext_zeichen(7) <> Game_fuellzeichen Then

      ' NEIN -> Buchstaben GROSS schreiben
      Game_scrolltext_zeichen(7) = Game_scrolltext_zeichen(7) - 32

   Else                                                    ' Null gesetztes Zeichen ?

      ' JA -> An nicht mehr gültiger Cursorstelle ein Cursorzeichen ausgeben
      Game_scrolltext_zeichen(7) = Game_cursorzeichen

   End If                                                  ' Null gesetztes Zeichen ?

Return
'-- End Game_update_scrolltext ------------------------------------------------


' *********
' * SOUND *
' *********

'--------------------------------------------------------------------------------
' SOUND - Gosub-Routine: Sound_piep_100ms
'
' Routine piepst für 100 ms
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' --
'--------------------------------------------------------------------------------
Sound_piep_100ms:

   ' ----- Programmcode -----

   Piezosound = Sound_ein
   Waitms 100
   Piezosound = Sound_aus

Return
'-- End Sound_piep_100ms -------------------------------------------------------


'--------------------------------------------------------------------------------
' SOUND - Gosub-Routine: Sound_piep_400ms
'
' Routine piepst für 400 ms
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
' --
'--------------------------------------------------------------------------------
Sound_piep_400ms:

   ' ----- Programmcode -----

   Piezosound = Sound_ein
   Waitms 400
   Piezosound = Sound_aus

Return
'-- End Sound_piep_400ms -------------------------------------------------------


'--------------------------------------------------------------------------------
' SOUND - Gosub-Routine: Sound_piep_1s
'
' Routine piepst für 1s
'
' Parameter: keine
' Rückgabe:  keine
```

```
'
' Globale Variablen:
' --
'-------------------------------------------------------------------------------
Sound_piep_1s:

   ' ----- Programmcode -----

   Piezosound = Sound_ein
   Wait 1
   Piezosound = Sound_aus

Return
'-- End Sound_piep_1s ------------------------------------------------------


' **********
' * EEPROM *
' **********

'-------------------------------------------------------------------------------
' EEPROM - Gosub-Routine: EE_init
'
' Routine prüft ob EEPROM schon initialisiert ist / war, holt ggf. dieses nach
' und liest aktuellen Bytepasscounter aus.
'
' Parameter: keine
' Rückgabe:  keine
'
' Globale Variablen:
'    Ee_bypass_counter      = Arbeitsvariable für Bypasscounter
'    Ee_data_bypass_counter = Flash-variable für Bypasscounter
'    Ee_data_version_value  = Flash-Variable für Version
'    Ee_version_value       = Arbeitsvariable für Version
'-------------------------------------------------------------------------------
Ee_init:

   ' ----- Programmcode -----

   ' Anmerkung:
   ' Es wird davon ausgegangen dass nicht initialisierte / neue EEPROMS
   ' bzw. gelöschte EEPROMS den Wert 0xFF enthalten.

   ' Version aus Speicher auslesen
   Ee_version_value = Ee_data_version_value

   ' Prüfen ob Eeprom schon initialisiert ist, wenn NEIN tue dies
   If Ee_version_value <> Ee_default_version_value Then

      ' NEIN, noch nicht initialisiert

      ' Arbeitsvariablen vorbelegen
      Ee_version_value = Ee_default_version_value
      Ee_bypass_counter = 0

      ' Arbeitsvariablen in EEPROM schreiben
      Ee_data_bypass_counter = Ee_bypass_counter
      Ee_data_version_value = Ee_version_value

      ' Ggf. Trace ausgeben
      #if Eeprom_testmodus
         Print "EEPROM und Arbeitsvariablen wurden neu initialisiert"
      #endif

   Else                                                  ' Eeprom schon Initialisiert

      ' JA, Speicher ist initialisiert

      ' Counter in Arbeitsvariable auslesen
      Ee_bypass_counter = Ee_data_bypass_counter

      ' Ggf. Trace ausgeben
      #if Eeprom_testmodus
         Print "EEPROM ist gueltig. Bypasscounter = " ; Ee_bypass_counter
      #endif

   End If                                                ' Eeprom schon Initialisiert

Return
'-- End Ee_init ------------------------------------------------------------
```

```
'--------------------------------------------------------------------------------
' Devices schließend und ggf. "Terminate Programm execution"
'--------------------------------------------------------------------------------

' System halt
End                                                     'end program


'--------------------------------------------------------------------------------
' Definition von globalen Konstantenfeldern
'--------------------------------------------------------------------------------

Worte_data:
Data "WELTRAUMFAHRT"                                    ' 1
Data "ABENTEUERLICH"                                    ' 2
Data "ABENTEUERLUST"                                    ' 3
Data "ANLAGEOBJEKTE"                                    ' 4
Data "ASTROGEOLOGIE"                                    ' 5
Data "ASTROBIOLOGIE"                                    ' 6
Data "BERGKRISTALLE"                                    ' 7
Data "BERGSTATIONEN"                                    ' 8
Data "BERGSTEIGEREI"                                    ' 9
Data "BERGWANDERUNG"                                    ' 10
Data "MYSTERYCACHES"                                    ' 11
Data "HAUSAPOTHEKEN"                                    ' 12
Data "HAUSVERWALTER"                                    ' 13
Data "HAUSHALTSWARE"                                    ' 14
Data "FESTSPIELHAUS"                                    ' 15
Data "FALSCHAUSSAGE"                                    ' 16
Data "WOCHENENDHAUS"                                    ' 17
Data "HAUSBESETZUNG"                                    ' 18
Data "WACKELKONTAKT"                                    ' 19
Data "WACHTELSCHLAG"                                    ' 20
Data "FAHRRADFAHREN"                                    ' 21
Data "DURCHGELAUFEN"                                    ' 22
Data "ASPHALTCOWBOY"                                    ' 23
Data "ARBEITSZIMMER"                                    ' 24
Data "FEUERWEHRAUTO"                                    ' 25
Data "ABSCHIEDSKUSS"                                    ' 26
Data "BUNDESKANZLER"                                    ' 27
Data "ZWIEBELSCHALE"                                    ' 28
Data "BANKGEHEIMNIS"                                    ' 29
Data "ANSICHTSKARTE"                                    ' 30
Data "ASTROPHYSIKER"                                    ' 31
Data "FEUERWEHRMANN"                                    ' 32
Data "KALENDERMONAT"                                    ' 33
Data "HAFENARBEITER"                                    ' 34
Data "COMPUTERFIRMA"                                    ' 35
Data "KNOBLAUCHZEHE"                                    ' 36
Data "LEUCHTREKLAME"                                    ' 37
Data "MINUTENZEIGER"                                    ' 38
Data "VERSTECKSPIEL"                                    ' 39
Data "AMEISENHAUFEN"                                    ' 40
Data "KAPITALANLAGE"                                    ' 41
Data "GANGSCHALTUNG"                                    ' 42
Data "MUTTERSPRACHE"                                    ' 43
Data "BERUFSBERATER"                                    ' 44
Data "NEBELSCHWADEN"                                    ' 45
Data "OBERINSPEKTOR"                                    ' 46
Data "WASCHMASCHINE"                                    ' 47
Data "BLUMENSCHMUCK"                                    ' 48
Data "ORIENTTEPPICH"                                    ' 49
Data "DAMPFMASCHINE"                                    ' 50
Data "PERSONENWAGEN"                                    ' 51
Data "POLIZEISCHUTZ"                                    ' 52
Data "FAMILIENNAMEN"                                    ' 53
Data "RASIERAPPARAT"                                    ' 54
Data "SATTELTASCHEN"                                    ' 55
Data "ZAPPELPHILIPP"                                    ' 56
Data "TASCHENMESSER"                                    ' 57
Data "IDEENREICHTUM"                                    ' 58
Data "TODESKANDIDAT"                                    ' 59
Data "KANONENFUTTER"                                    ' 60
Data "GEISTERSTUNDE"                                    ' 61
Data "UMSTANDSKLEID"                                    ' 62
Data "INNENMINISTER"                                    ' 63
Data "RAUMFORSCHUNG"                                    ' 64
Data "UNTEROFFIZIER"                                    ' 65
```

```
Data "COMPUTERSPIEL"                                    ' 66
Data "VANILLESCHOTE"                                    ' 67
Data "POLTERGEISTER"                                    ' 68
Data "VERSUCHSREIHE"                                    ' 69
Data "ELEKTROMAGNET"                                    ' 70
Data "WALNUSSKUCHEN"                                    ' 71
Data "SCHATTENBOXEN"                                    ' 72
Data "HAUPTSCHALTER"                                    ' 73
Data "ZIMMERANTENNE"                                    ' 74
Data "WASSERLEITUNG"                                    ' 75
Data "ERDUMLAUFBAHN"                                    ' 76
Data "ZUGVERBINDUNG"                                    ' 77
Data "EINTRITTSGELD"                                    ' 78
Data "WARTESCHLANGE"                                    ' 79
Data "ZUCKERMELONEN"                                    ' 80
Data "ENERGIEQUELLE"                                    ' 81
Data "WASSERSPIEGEL"                                    ' 82
Data "VERKEHRSCHAOS"                                    ' 83
Data "ENERGIEBEDARF"                                    ' 84
Data "SCHMERZMITTEL"                                    ' 85
Data "VERBOTSSCHILD"                                    ' 86
Data "NIEDRIGWASSER"                                    ' 87
Data "FAHRRADREIFEN"                                    ' 88
Data "TROMMELBREMSE"                                    ' 89
Data "SCHAUKELSTUHL"                                    ' 90
Data "INFRASTRUKTUR"                                    ' 91
Data "RECHENAUFGABE"                                    ' 92
Data "POLIZEIWACHEN"                                    ' 93
Data "ZAHLENSCHLOSS"                                    ' 94
Data "MODEINDUSTRIE"                                    ' 95
Data "LICHTSCHALTER"                                    ' 96
Data "BRIEFUMSCHLAG"                                    ' 97
Data "KLASSENZIMMER"                                    ' 98
Data "HAUTAUSSCHLAG"                                    ' 99
Data "INSEKTENPLAGE"                                    ' 100

' ------------------------------------------------------------------------------


'#################################### END ######################################


'################################### Historie ##################################
'
' 30.05.2014 : Version x.x
'              Beginn der Implementierungen für Hanging Man.
'              Fertigstellung Inbetriebnahme LCD-Display und wichtigster
'              Funktionen.
' ------------------------------------------------------------------------------
' 09.06.2014 : B_SIGNAL und PUSH_SIGNAL wegen Erfassung über Interrupt 1
'              getauscht.
' ------------------------------------------------------------------------------
' 02.07.2014 : Restarbeiten und Begriffe auf 100 Stück ergänzt.
' ------------------------------------------------------------------------------
' 04.07.2014 : Spiel-Bypass eingebaut um Koordinaten ohne Spiel anzeigen
'              zu können :-)
' ------------------------------------------------------------------------------
' 09.07.2014 : Bypasscounter implementiert und Ablage Counter in EEPROM
' ------------------------------------------------------------------------------
' 09.07.2014 : Versions 1.0 - Fertigstellung erste Produktivversion
'
'###############################################################################
```

Software 5: Source-Code des Projekts Hanging Man

14 Interessantes und wichtige Links

14.1 Bücher und Literatur

- myAVR Lehrbuch Mikrocontroller-Programmierung
  Laser & Co. Solutions GmbH
- LCD Lehrheft
  Laser & Co. Solutions GmbH
- Projekt „myTWI“
  Laser & Co. Solutions GmbH
- Mikrocomputertechnik mit Controllern der Atmel AVR-RISC-Familie
  G. Schmitt, Oldenburgverlag, ISBN 3-486-58016-7
- Leiterplattendesign mit EAGLE
  André Ketler, Marc Neujahr, mitp Verlag, ISBN 978-3-8266-1340-1
- Messen, Steuern und Regeln mit AVR_Controllern
  Wolfgang Trampert, Franzis Verlag, ISBN 3-7723-4298-1
- Programmierung der AVR RISC Mikrocontroller mit BASCOM-AVR
  Claus Kühnel, Books on Demand, ISBN 3907857046

14.2 Internet

14.2.1 Firmen und Foren

SW

MCS Electronics
URL: http://www.mcselec.com/
- BASCOM-AVR BASIC Compiler und Forum

HW

Bauelemente:

Conrad Electronic
URL: http://www.conrad.de
- Bauelemente und Zubehör (zum Teil aber sehr teuer)

Pollin Electronic
URL: http://www.pollin.de/shop/shop.php
- Bauelemente
- Atmega Evaluationsboard

Reichelt elektronik
URL: http://www.reichelt.de/
- Günstige Bauelemente
- STK500 von ATMEL

Farnell Deutschland
URL: http://de.farnell.com/jsp/home/homepage.jsp
- Baulelemente

ELV

URL: http://shop.elv.de/output/controller.aspx
- Günstige Bauelemente
- LCD-Displays

RS Components GmbH
URL: http://www.rsonline.de
- Große Auswahl an Bauelementen zu guten Preisen

Hersteller und Spezialitäten:

ATMEL Corporation
URL: http://www.atmel.com/
- ATmega
- STK500

Sensirion
URL: http://www.sensirion.com/
- Halbleiterhersteller für Temperatursensoren und Feuchtigkeitssensoren

TAOS
URL: http://www.taosinc.com
- Halbleiterhersteller für Lichtsensoren

Riesen + Kern GmbH
URL: http://www.driesen-kern.de
- Deutscher Distributor für Sensirion Halbleiter

rb-Messtechnik Reinhardt
URL: http://www.rb-messtechnik.de
- Windgeber

DatasheetCatalog.COM
URL: http://www.datasheetcatalog.com
- Datenblätter zu fast allen bekannten elektr. Bauelementen

Worls Of Electronic – Elektronikprojekte
URL: http://www.woe.onlinehome.de/projekte.htm
- AVR JTAG Emulator

Plantinenservice und Hersteller:

GS Electronic
Sven Schult
Spillbähnstraße 19a
53844 Troisdorf

Tel. 02241-3010465
Fax 02241-3010469

eMail gselectronic@gsel.de
URL: http://www.gsel.com/
- Platinenservice
- Einzelstücke
- Kleinserien

LeitOn GmbH
Gottlieb-Dunkel-Str. 47-48
12099 Berlin

Tel.: +49-(0)30-701 73 49-0

Fax: +49-(0)30-701 73 49-19

E-Mail: kontakt@leiton.de
URL: http://www.leiton.de
- Platinenservice
- Einzelstücke
- Kleinserien
- Sehr günstige Preise
- Leiton stellt Download für Eagle DesignRules zur Verfügung

14.2.2 ATmega SW und HW-Lösungen

Laser & Co. Solutions GmbH
URL: http://www.myavr.de
- Bausätze zum ATmega8
- SW-Lösungen zum Selbststudium
- Dokumente

14.2.3 Foren

RoboterNetz.de
URL: http://www.roboternetz.de
- Großer Portal für Robotik, Elektronik und Mikrocontroller

MCS Electronics
URL: http://www.mcselec.com/
- Forum rund um BASCOM-AVR

AVR feaks
URL: http://www.avrfreaks.net/
- Forum AVRFREAKS.NET

Pony-Prog Tutorial
URL: http://www.mikrocontroller.net/articles/Pony-Prog_Tutorial
- Pony-Prog Tutorial

QSLnet
URL: http://www.qsl.net
URL: http://www.qsl.net/pa3ckr/index.html
URL: http://www.qsl.net/pa3ckr/bascom%20and%20avr/arrays%20and%20data/index.html
- Forum für Elektronik und SW-Lösungen (entstanden aus Radio Amateur Community)
- Zusammenfassung BASCOM und AVR Lösungen (Arrays usw.)

AVR_Praxis
URL: http://www.avr-praxis.de/index.php
AVR-PRAXiS ist ein Forum, das ausschließlich für einen Gedankenaustausch und als Diskussionsplattform für Interessierte bereitstelle, welche sich privat, durch das Studium oder beruflich mit der AVR-Mikrocontrollerfamilie beschäftigen wollen oder müssen.

Microcontroller.net
URL: http://www.mikrocontroller.net
Großes Portal mit Forum und Chat

BASCOM-Forum
URL: http://www.bascom-forum.de
Forum für Projekte, Hardware und Diskussionen

Infos rund um den ATmega:
URL: http://www.dieelektronikerseite.de/uC%20Ecke/Module/Ports%20-%20Wenn%20der%20AVR%20steuert.htm
URL: http://www.kreatives-chaos.com/artikel/avr-grundschaltungen

Informationen zum TWI / I²C-Bus:
URL: http://www.roboternetz.de/wissen/index.php/I2C
URL: http://www.roboternetz.de/wissen/index.php/TWI
URL: http://www.roboternetz.de/wissen/index.php/TWI_Praxis
URL: http://www.roboternetz.de/wissen/index.php/TWI_Praxis_Multimaster
URL: http://www.roboternetz.de/wissen/index.php/Bascom_I2C_Master
URL: http://www.roboternetz.de/wissen/index.php/Bascom_und_USI-Kommunikation
URL: http://www.roboternetz.de/wissen/index.php/Bascom_Soft-I2c_Library
URL: http://www.roboternetz.de/wissen/index.php/Bascom_Inside-Code

14.3 Das STK500 und STK501

14.3.1 Das STK500 mit STK501 (ATMEL)

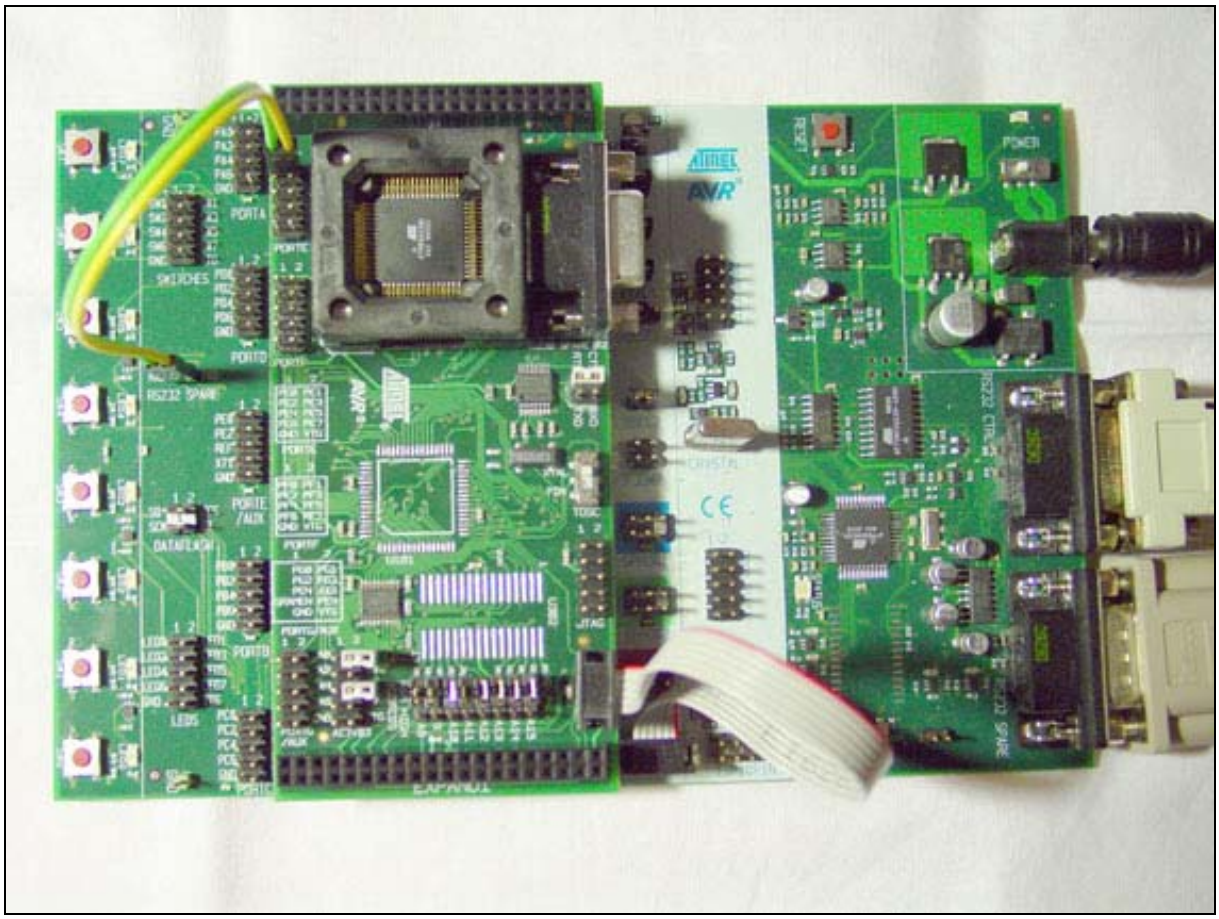


Abbildung 39: Das STK500 mit STK501 (ATMEL)

14.3.2 STK501 anschließen für RS232-Kommunikation

Der 9-polige Sub-D Stecker mit der Beschriftung „RS232 Spare“ wird mit einem freien COM-Port des PC verbunden.

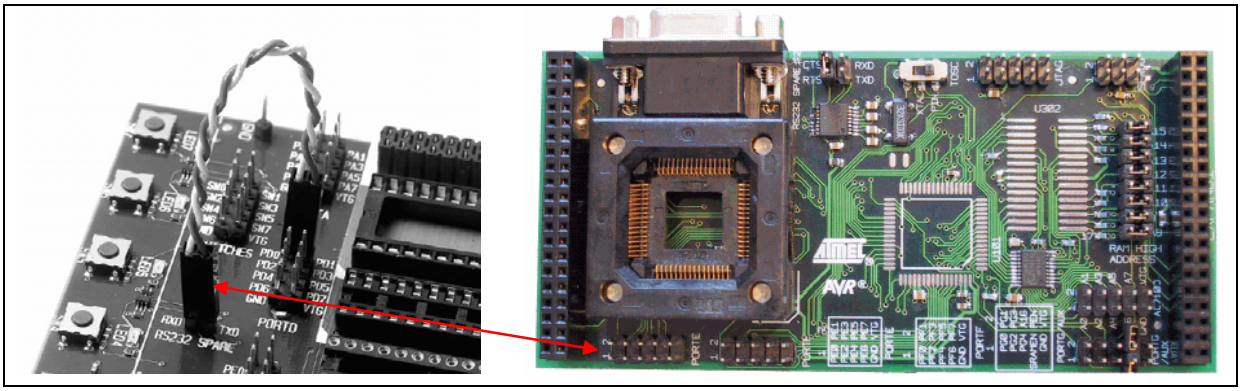


Abbildung 40: RS232-Verbindung STK500 und STK501

Der 9pol. RS232 Spare Stecker ist über einen Pegelwandler mit den beiden Pins im IO-Bereich verbunden (heist auch RS232 Spare). Diese zwei Pins (RxD und TxD) müssen mit dem mitgelieferten 2pol Kabel mit den Pins PE0 und PE1 (PortE) auf dem STK501 verbunden werden.

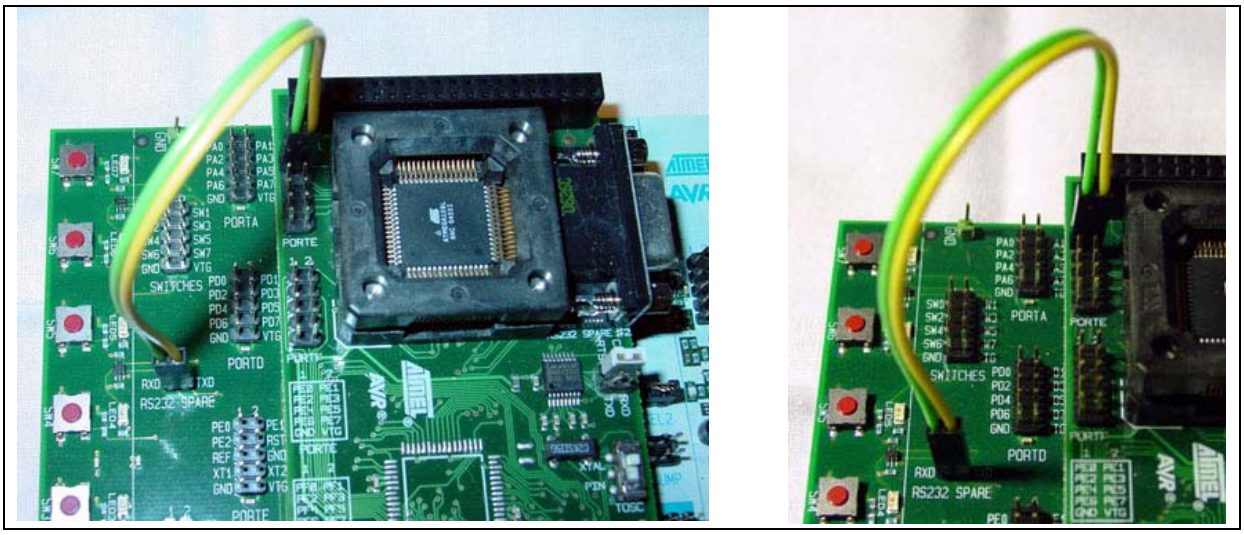


Abbildung 41: RS232-Verbindung STK500 mit STK501

14.4 Inbetriebnahme AVR ISP mkII

Für die Inbetriebnahme des STK500 + STK501 muss der RESET-Jumper auf dem STK500-Board entfernt werden (Dieser Hinweis stammt aus der Dokumentation des mkII).

Alle weiteren Einstellungen zeigen die folgenden Dialog-Boxen des AVR Studios 4.0:

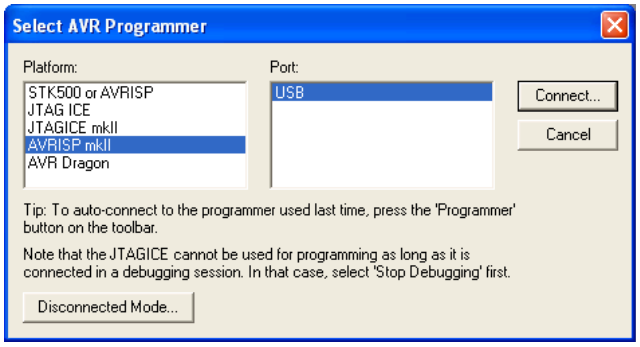


Abbildung 42: AVR Studio Programmer Selection

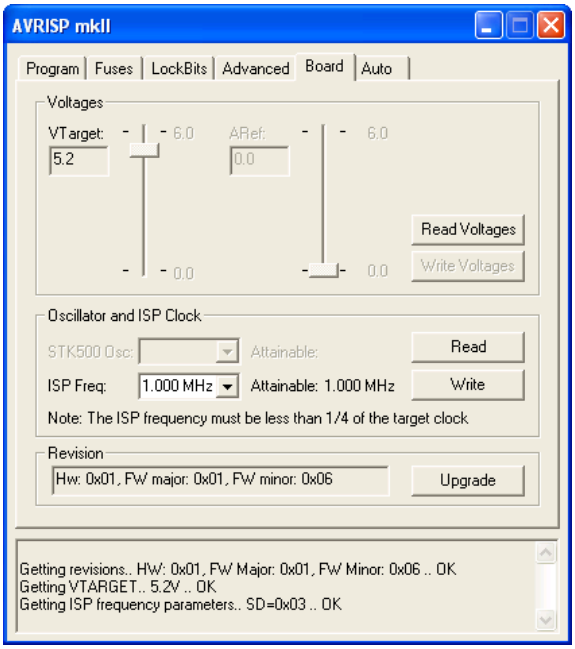


Abbildung 43: AVR Studio Board-Settings

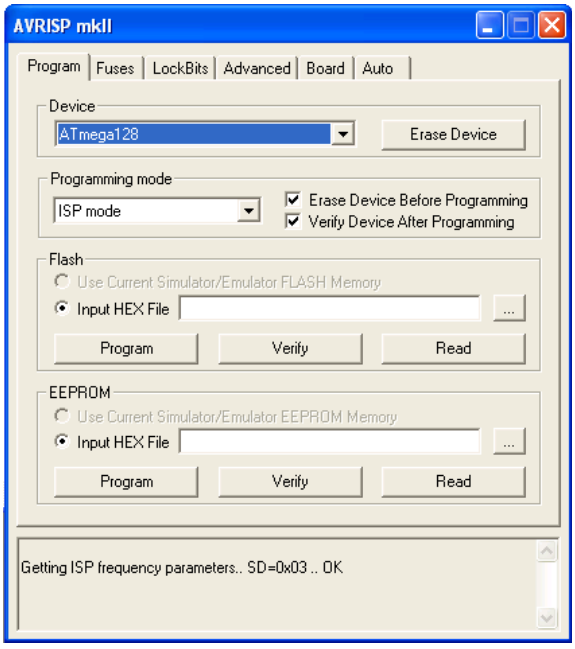


Abbildung 44: AVR Studio Program Settings

Hinweis: Die ISO Frequenz muss 1/4 bzw. 1/6 der Taktfrequenz des ATmegas betragen. Bei 16 MhZ des ATmega128 wurde der ISP Prommer erfolgreich mit 1,000 MHz in Betrieb genommen.

15 Entwicklungsbegleitende Notizen und Informationen

15.1 Projektcheckliste für AVR Systemdesigns

Diese Checkliste beinhaltet eingige grundlegende Regeln beim Design mit AVR Mikrokontrollern.

http://www.mikrocontroller.net/articles/AVR_Checkliste

Dies sind zusammengefasst in Kürze:

15.1.1 Abblockkondensator(en) ordnungsgemäß installiert?

Abblockkondensatoren ("Bunker-Kondensatoren") dienen dazu, sehr kurze Versorgungsspannungseinbrüche, die durch Schaltvorgänge verursacht werden können, zu kompensieren. Diesen Zweck erfüllen sie optimal, wenn folgende Regeln eingehalten werden:

- Ein Abblockkondensator sollte möglichst dicht am IC sitzen.
- Jedes IC in einer Schaltung sollte einen Abblockkondensator besitzen.
- Bei ICs mit mehreren Anschlüssen für VCC und GND sollte jedes VCC-GND-Paar mit einem eigenen Abblockkondensator beschaltet werden (z. B. AVRs in SMD-Bauform wie dem ATmega16A also mit vier Kondensatoren).
- Es sollten keramische Kondensatoren mit einer Kapazität von 100 nF verwendet werden. Größere Kondensatoren, etwa 10 µF-Elkos, sind für diese Aufgabe *nicht* geeignet, weil sie "zu langsam" sind!

15.1.2 Spannungsversorgung richtig angeschlossen?

Der AVCC-Pin ist der Versorgungsanschluss für den AD-Wandler und den zugehörigen Port. Er ist nicht an allen AVRs vorhanden; wenn er aber vorhanden ist, so muss er auf jeden Fall angeschlossen sein, auch wenn der AD-Wandler nicht benutzt wird. Wird der AD-Wandler verwendet, sollte zur Verbesserung der Genauigkeit der AVCC-Pin über einen Lowpass-Filter angeschlossen werden (siehe Datenblatt). Oft funktioniert die Programmierung des Controllers auch, wenn Vcc oder GND nicht richtig angeschlossen ist. Zur Sicherheit kann man mit einem Messgerät direkt an den Anschlüssen des AVRs kontrollieren (VCC-GND, AVCC-GND) prüfen, ob die Verbindungen korrekt sind. Es empfiehlt sich, vor dem Einsetzen bzw. Einlöten des Controllers die Versorgungsanschlüsse nochmals zu prüfen, um sicherzustellen, dass man den IC nicht durch eine zu hohe Spannung aufgrund eines Fehlers in der Versorgung zerstört.

15.1.3 Reset-Pin korrekt beschaltet?

Der Reset-Anschluss am AVR ist 'active-low', d. h. wenn man den Pin mit GND (Masse) verbindet, wird der Controller resettet. Zwar haben AVRs einen internen Pullup-Widerstand, der den Reset-Pin gegen VCC "zieht", dieser ist jedoch relativ hochohmig (ca. 50 kOhm, vgl. Datenblatt) und reicht unter Umständen nicht aus, um den Reset-Pin sicher "hochzuhalten". Als Mindestbeschaltung empfiehlt sich dringend, einen externen Pullup-Widerstand vorzusehen (typisch 10 kOhm), der den Reset-Pin mit VCC verbindet. Er sollte nicht kleiner als 4,7 kOhm sein, da der Programmieradapter sonst eventuell den Reset-Pin während des Programmiervorgangs nicht sicher auf "low" ziehen kann. Zusätzlich sollte man auch noch einen Kondensator 47 nF oder 100 nF zwischen Reset-Pin und GND anordnen. Dieses RC-Glied sorgt dafür, dass der Controller beim Einschalten der Versorgungsspannung für eine definierte Zeitspanne im Reset gehalten wird. Im laufenden Betrieb sorgt der Kondensator dafür, dass der Reseteingang unempfindlich gegenüber Spikes und Glitches wird. Er sollte deshalb unmittelbar in Pin-Nähe beim Prozessor untergebracht werden. Dieser Kondensator darf jedoch nicht verwendet werden, wenn DebugWire möglich sein soll.

Atmel empfiehlt zusätzlich noch zum Schutz vor Überspannungen eine externe Diode nach VCC ("Clamp-Diode"), da für den Reset-Pin keine interne vorhanden ist. Diese Diode bereitet jedoch bei manchen Programmieradaptern Schwierigkeiten.

15.1.4 Alle Ground-Anschlüsse beschaltet?

Bei AVRs mit mehreren Ground-Anschlüssen müssen alle Anschlüsse beschaltet werden.
Siehe

http://www.mikrocontroller.net/forum/read-1-107259.html

15.2 Datenblätter

ALPS - STEC11B Drehimpulsgeber 11 mm mit Taster
Produktbezeichnung Reichelt: STEC11B13

Encoder
11mm Size Metal Shaft Type
EC11 Series

Car Use RoHS applicable

Compact and highly reliable type available in many varieties.

Features

- Compact and highly accurate sliding contact type encoder.
- 1.5mm-travel push switch with compliance to 4.5mm height.
- Incremental type.
- Operation life 1,000,000cycles can be supported (EC11J).

Applications

- For controlling volume in car AV equipment, volume and menu selection in car navigation systems and various in-car components where less mounting surface is required
- For various types of controlling in DVD players/recorders, mini component stereos, CD players, portable audio players and various AV equipment
- For home appliances (microwave ovens)

Typical Specifications

Items	Specifications
Rating	10mA 5V DC
Operating life	15,000cycles EC11J:100,000cycles / 500,000cycles / 1,000,000cycles

Product Line

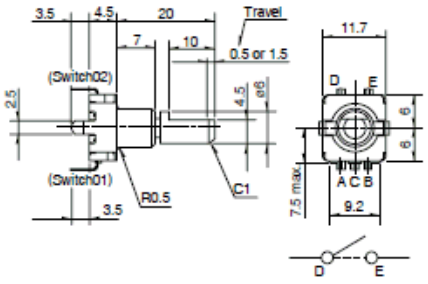
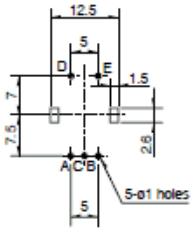
Structure	Actuator configuration	Actuator length (mm)	Torque (mN·m)	Number of detent	Number of pulse	Push-on switch	Travel of push-on switch (mm)	Operating life (cycles)	Minimum packing unit (pcs.)	Product No.	Drawing No.
Horizontal	Flat	20	12±7	30	15	Without	——	15,000	700	EC11B15202AA	1
						With	0.5			EC11B15242AE	2
							1.5			EC11B15242AF	3
Vertical			10±7	18	9	Without	——		1,200	EC11E09204A4	4
				30	15					EC11E15204A3	
			7^{+3}_{-4}	Without						EC11E1530401	
			10±7	36	18					EC11E1820402	
			7^{+3}_{-4}	Without						EC11E1830401	
			10±7	18	9	With	0.5			EC11E09244BS	5
				30	15					EC11E15244G1	
			7^{+3}_{-4}	Without						EC11E153440D	
			10±7	36	18					EC11E18244AU	
			7^{+3}_{-4}	Without						EC11E183440C	
			10±7	18	9		1.5			EC11E09244AQ	
				30	15					EC11E15244B2	
			7^{+3}_{-4}	Without						EC11E1534408	
			10±7	36	18					EC11E18244A5	
			7^{+3}_{-4}	Without						EC11E1834403	

Product Line

Structure	Actuator configuration	Actuator length (mm)	Torque (mN·m)	Number of detent	Number of pulse	Push-on switch	Travel of push-on switch (mm)	Operating life (cycles)	Minimum packing unit (pcs.)	Product No.	Drawing No.
Less shaft wobble	20-tooth serration	18	10±7	30	15	With	1.5	15,000	1,200	EC11G1524402	6
			7^{+3}_{-4}	Without						EC11G1534403	
Reflow	Flat	20	12±5	18	9	Without	—	100,000	300	EC11J0920401	7
				30	15					EC11J1520401	
				18	9	With	0.5			EC11J0924401	8
				30	15					EC11J1524401	
				18	9		1.5			EC11J0925401	
				30	15					EC11J1525401	
				18	9	Without	—	500,000		EC11J0920601	7
				30	15					EC11J1520601	
				18	9	With	0.5			EC11J924601	8
				30	15					EC11J1524601	
				18	9		1.5			EC11J0925601	
				30	15					EC11J1525601	
				18	9	Without	—	1,000,000		EC11J0920801	7
				30	15					EC11J1520801	
				18	9	With	0.5			EC11J092481	8
				30	15					EC11J1524801	
				18	9		1.5			EC11J0925801	
				30	15					EC11J1525801	
Push lock	20-tooth serration	25	10±7	30	15	Without	—	15,000	1,000	EC11E152T409	9
		26.4				With	8		800	EC11E152U402	10
Self-return switch	Flat	20	3 to 30	Without	Self-return switch	Without	—		1,200	EC1110120001	11
						With	0.5			EC111012010H	12
							1.5			EC1110120201	
Dual-shaft	Flat	Inner-shaft=25	10± 7	30	15		0.5			EC11EBB24C03	13
	Slotted	Outer-shaft=15				Without	—				
	Flat	Inner-shaft=25				With	1.5		700	EC11E0B2LB01	14
	Slotted	Outer-shaft=15	3 to 30	Without	Self-return switch	Without	—				

Note

Products other than those listed in the above chart are also available. Please contact us for details.

5	**Vertical with push-on switch (travel 0.5/1.5mm)** 		

Product Varieties

Shaft Dimensions

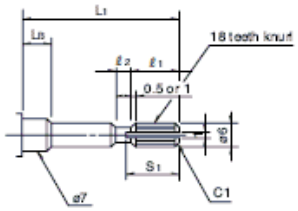
1. Single-shaft Type

1) Knurled Type

Unit:mm

Configuration (Shaft diameter : ϕ6)

Not applicable for EC11E and EC11G with push-lock mechanism



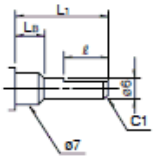
Detailed dimensions

L1	LB	ℓ1	ℓ2	S1
20	7	6	1	7
25	7	10	2	11

2) Flat Type

Unit:mm

Configuration (Shaft diameter : ϕ6)



Detailed dimensions

	L1	LB	ℓ
※	15	5	7
	15	7	5 (6)
	20	7	10 (12)
	25	7	12

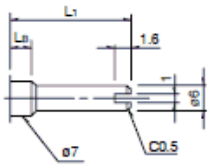
※Does not comply with EC111

Dimensions marked with ( ) apply to products without push-on switches.

3) Slotted Type

Unit:mm

Configuration (Shaft diameter : ϕ6)



Detailed dimensions

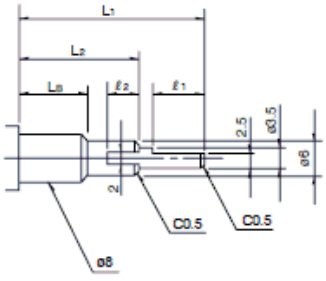
L1	LB
15	7
20	7
25	7

2. Standard Dimensions and configuration of Dual-shaft Type

1) Flat Type

Unit:mm

Configuration (Inner-shaft : ϕ3.5 Outer-shaft : ϕ6)



Detailed dimensions

L1	L2	LB	ℓ1	ℓ2
20	10	5	7	4
25	15	7	7	5
30	20	7	7	5

Notes

1. The highlighted figures in shaft types refer to Product Specifications in P.194 and P.195.
2. Products other than those listed in the above chart are also available. Please contact us for details

Product Specifications

Items			EC11B	EC11E/EC11G	EC11J	EC111	EC20A
Operating temperature range			–30℃ to +85℃		–40℃ to +85℃	–30℃ to +85℃	–30℃ to +80℃
Maximum operating current (Resistive load)			10mA				0.5mA
Electrical performance	Rating		10mA 5V DC				0.5mA 5V DC
	Output signal		Output of A and B signals, proportionate to phase difference			Self-return switch	Output of A and B signals, proportionate to phase difference
	Insulation resistance		100MΩ min. 250V DC				10MΩ min. 50V DC
	Voltage proof		300V AC for 1minute				50V AC for 1min.
Mechanical performance	Rotational torque (without click feeling)		——	7^{+3}_{-4}mN·m	——	3 to 30mN·m	——
	Detent torque		12±7mN·m	10±7mN·m	12±5mN·m	——	40±20mN·m
	Push-pull strength		100N				
	Resistance to soldering heat	Manual soldering	350℃ max. 3s max.				
		Dip soldering	260±5℃, 5±1s		——	260±5℃, 5±1s	
		Reflow soldering	————		Please see P.235	————	
Durability	Rotational life		15,000cycles		100,000cycles	15,000cycles	30,000cycles
Environmental performance	Cold		–40±3℃ for 240h				
	Dry heat		85±3℃ for 240h				
	Damp heat		60±2℃, 90 to 95%RH for 240h				

Push-on Switch Specifications

Items	EC11B		EC11E/EC111/EC11G		EC11J	EC20A	
Circuit・number of contacts	Single pole and single throw (Push-on)						
Travel (mm)	$0.5^{+0.4}_{-0.3}$	1.5±0.5	0.5±0.3	1.5±0.5 8±0.8 ※	0.5±0.3	1.5±0.5	
Operating force (N)	6±3	5±2	$6^{+2.5}_{-2}$	4±2 8 max. ※	5±2	4±2	
Rating	3A 16V DC (10mA 16V DC min. ratings)		0.5A 16V DC (1mA 16V DC min. ratings) 0.5A 12V DC ※		0.1A 5V DC		0.5A 16V DC (1mA 16V DC min. ratings)
Contact resistance	100mΩ for initial period; 200mΩ after rotational life						
Operating life	25,000 times min.	20,000times min. 10,000times min.※			1,000,000 times min.	100,000 times min.	20,000times min.

Note

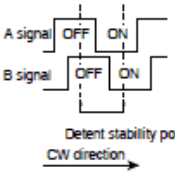
※marked specification is only applicable to EC11E152U402.

Product Specifications

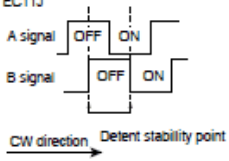
Output Wave

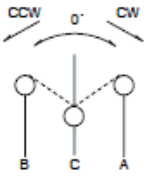
EC11B	EC11E/EC11G/EC11J	EC111	EC20A

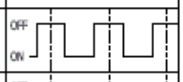
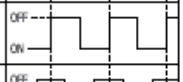
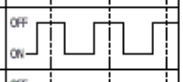
EC11B, EC11E, EC11G
30 detents, 15 pulse



The stable detent position cannot be identified in phase B.
EC11E 18 detents 9 pulse
EC11E 36 detents 18 pulse
EC11J



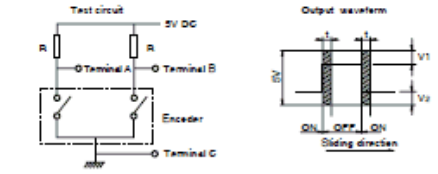


Shaft rotational Direction	Signal	Output
Clockwise	A (Terminal A–C)	
	B (Terminal B–C)	
Counter-clockwise	A (Terminal A–C)	
	B (Terminal B–C)	
		The dotted lines indicate detent positions

Sliding Noise

EC11B	EC11E/EC11G/EC11J	EC111	EC20A

$V_1 = V_2 = 1.5V$ max.



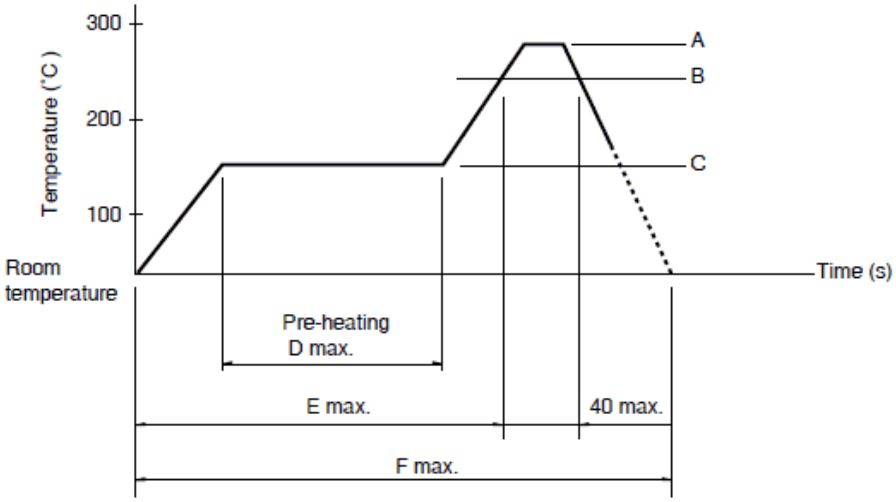
Measurement condition : Rotation speed 360°/s t : Masking time to avoid chattering

EC11B	EC11E/EC11G/EC11J	EC111	EC20A
At R = 5kΩ Chattering : 2ms max. Bounce : 2ms max.	At R = 5kΩ Chattering : 3ms max. Bounce : 2ms max.	——	At R = 5kΩ Chattering : 8ms max. Bounce : 5ms max.

Soldering Conditions

Example of Reflow Soldering Condition

1. Heating method: Double heating method with infrared heater.
2. Temperature measurement: Thermocouple 0.1 to 0.2 ϕ CA (K) or CC (T) at soldering portion (copper foil surface). A heat resisting tape should be used for fixed measurement.
3. Temperature profile



Series(Reflow type)	A(℃) 3s max.	B(℃)	C(℃)	D(s)	E(s)	F(s)
EC11J	260	230	150	120	———	240

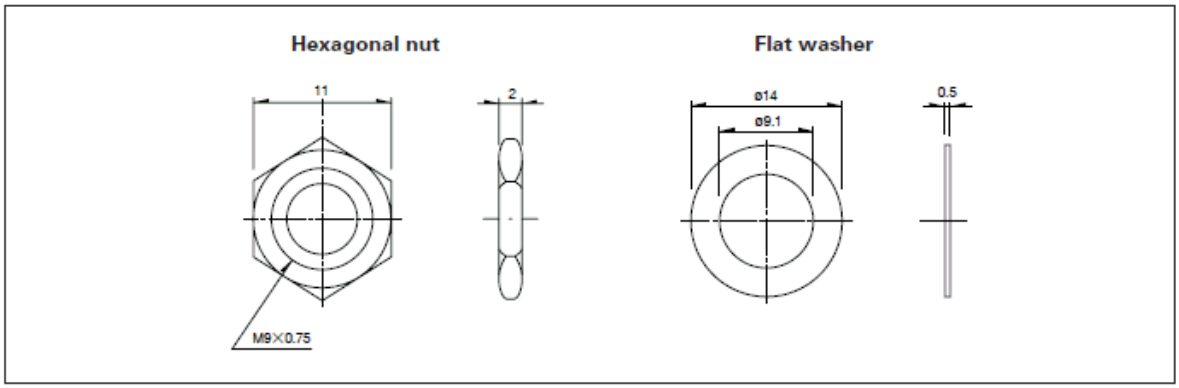
Notes

1. The condition mentioned above is the temperature on the mounting surface of a PC board. There are cases where the PC board's temperature greatly differs from that of the switch, depending on the PC board's material, size, thickness, etc. The above-stated conditions shall also apply to switch surface temperatures.
2. 2. Soldering conditions differ depending on reflow soldering machines. Prior verification of soldering condition is highly recommended.

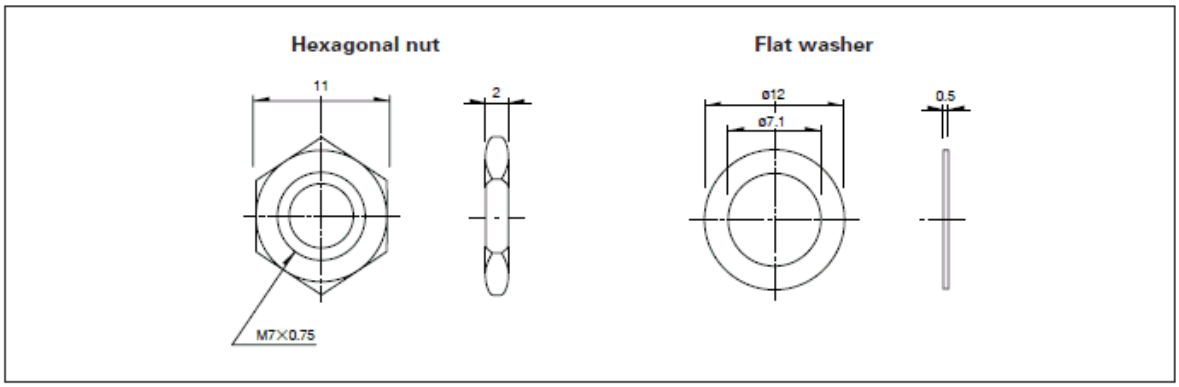
Attached Parts

The following parts are included with the product.

SRGH Series Unit:mm



RK097, EC11B Series Unit:mm



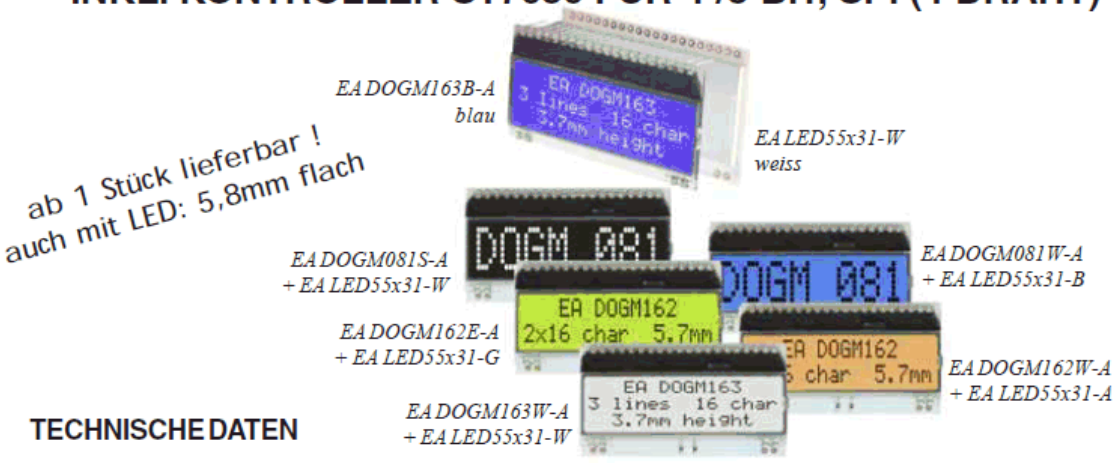
LCD Display
Electronic Assembly EA DOG162B-A

6.2005

EA DOG-M

DOG-SERIE 3,3V
SUPER FLACH / 55x27mm
INKL. KONTROLLER ST7036 FÜR 4-/8-BIT, SPI (4-DRAHT)

ab 1 Stück lieferbar !
auch mit LED: 5,8mm flach



TECHNISCHE DATEN

* KONTRASTREICHE LCD-SUPERTWIST ANZEIGE
* OPTIONALE LED-BELEUCHTUNGSKÖRPER INVERSCHIEDENEN FARBEN
* 1x8, 2x16 UND 3x16 ZEICHEN MIT 12,0 mm / 5,6 mm UND 3,6 mm SCHRIFT
* KONTROLLER ST 7036 FÜR 4-BIT, 8-BIT UND SPI (4-DRAHT) INTERFACE
* SPANNUNGSVERSORGUNG+3,3V ODER +5V SINGLE SUPPLY (typ. 250µA)
* KEINE ZUS. SPANNUNGEN ERFORDERLICH
* BETRIEBSTEMPERATURBEREICH -20..+70°C
* LED-HINTERGRUNDBELEUCHTUNG3..80mA@3,3V oder 2..40mA@5V
* KEINE MONTAGE ERFORDERLICH: EINFACH NUR IN PCB EINLÖTEN

BESTELLBEZEICHNUNG

LCD-MODUL 1x8 - 11,97mm	**EA DOGM081x-A**
LCD-MODUL 2x16 - 5,57mm	**EA DOGM162x-A**
LCD-MODUL 3x16 - 3,65mm	**EA DOGM163x-A**

x: B = blauer Hintergrund
E = Gelb/grüner Hintergrund
L = Gelb/grüner Hintergrund (nicht beleuchtbar)
S = schwarzer Hintergrund
W = weisser Hintergrund

LED-BELEUCHTUNG WEISS	**EA LED55X31-W**
LED-BELEUCHTUNG GELB/GRÜN	**EA LED55X31-G**
LED-BELEUCHTUNG BLAU	**EA LED55X31-B**
LED-BELEUCHTUNG ROT	**EA LED55X31-R**
LED-BELEUCHTUNG AMBER	**EA LED55X31-A**
BUCHSENLEISTE 4,8mm HOCH (2 STÜCK ERFORDERLICH)	**EA FL-20P**

ELECTRONIC ASSEMBLY GMBH
making things easy
LOCHHAMER SCHLAG 17 · D-82166 GRÄFELFING
TEL 089/8541991 · FAX 089/8541721 · http://www.lcd-module.de

EA DOG-M

ELECTRONIC ASSEMBLY

EA DOG SERIE

Mit der EA DOG-Serie präsentiert ELECTRONIC ASSEMBLY die weltweite 1. Displayserie, welche ohne zusätzlicher Hilfsspanung an 3,3V Systemen lauffähig sind. Selbstverständlich ist auch der Betrieb an einem herkömmlichen 5V System möglich.
Anders als bei üblichen LCD-Modulen bestellen Sie hier die Anzeige und die entsprechende Beleuchtung separat. Dadurch ergeben sich mannigfaltige Kombinationsmöglichkeiten.
Konzipiert für kompakte Handgeräte bietet diese moderne LCD-Serie mit und ohne Beleuchtung eine Reihe echter Vorteile:

* extrem kompakt mit 55x31mm bei marktüblicher Schriftgröße von 5,57mm (2x16) !
* superfach mit 2,0mm unbeleuchtet bzw. 5,8mm inkl. LED-Beleuchtung
* 4-Bit, 8-Bit und SPI Interface (4-Draht)
* nur typ. 250µA Stromverbrauch in vollem Betrieb (LED-Beleuchtung weiss ab 3mA)
* simple Montage durch einfaches Einlöten
* verschiedenste (63) Designvarianten ab 1 Stück lieferbar

KONTRASTEINSTELLUNG

Für alle Displays der EA DOG- Serie ist der Kontrast per Befehl einstellbar. Dies erfolgt über die Bits C0..C5 in den Befehlen "Contrast Set" und "Power/Icon Control/Contrast Set". In der Regel wird der Kontrast einmalig eingestellt und dann - dank integrierter Temperaturkompensation - über den gesamten Betriebstemperaturbereich (-20..+70°C) konstant gehalten.
Insgesamt benötigen die Displays selbst im 3,3V Betrieb keine zusätzliche negative Spannung !

LED-BELEUCHTUNGEN

Zur individuellen Hintergrundbeleuchtung sind 5 verschiedene Varianten erhältlich: weiss, gelb/grün, blau, rot und amber.

Es stehen jeweils 2 separate LED-Pfade zur Verfügung, welche zur optimalen Anpassung an die Systemspannung parallel oder in Serie geschaltet werden können. Dadurch sind alle Beleuchtungen alternativ mit 5V oder auch mit 3,3V zu betreiben!

Der Betrieb der Hintergrundbeleuchtung erfordert einen externen Vorwiderstand zur Strombegrenzung. Dieser errechnet sich aus R=U/I; die Werte entnehmen Sie aus den Tabellen nebenan. Für eine optimale Lebensdauer empfehlen wir den Einsatz einer Stromquelle.

Die Lebensdauer der gelb/grünen, roten und amber-farbigen Beleuchtung beträgt 100.000 Stunden, die der weißen und blauen Beleuchtung deutlich darunter.

Achtung: Betreiben Sie die Beleuchtung nie direkt an 5V/3,3V; das kann zur sofortigen Zerstörung der LED's führen!
Beachten Sie unbedingt ein Derating bei Temperaturen >25°C.

yellow/green EA LED55x31-G	Forward voltage	Current max.	Limiting resistor @ 3,3 V	Limiting resistor @ 5 V
Connected in parallel	2,2 V	80 mA	14 ohm	35 ohm
Connected in series	4,4 V	40 mA	-	7,5 ohm

white EA LED55x31-W	Forward voltage	Current max.	Limiting resistor @ 3,3 V	Limiting resistor @ 5 V
Connected in parallel	3,2 V	60 mA	1,6 ohm	30 ohm
Connected in series	6,4 V	30 mA	-	-

blue EA LED55x31-B	Forward voltage	Current max.	Limiting resistor @ 3,3 V	Limiting resistor @ 5 V
Connected in parallel	3,2 V	60 mA	1,6 ohm	30 ohm
Connected in series	6,4 V	30 mA	-	-

amber EA LED55x31-A	Forward voltage	Current max.	Limiting resistor @ 3,3 V	Limiting resistor @ 5 V
Connected in parallel	2,4 V	80 mA	11 ohm	32 ohm
Connected in series	4,8 V	40 mA	-	5 ohm

red EA LED55x31-R	Forward voltage	Current max.	Limiting resistor @ 3,3 V	Limiting resistor @ 5 V
Connected in parallel	2,1 V	80 mA	15 ohm	36 ohm
Connected in series	4,2 V	40 mA	-	20 ohm

MONTAGE

Zuerst werden das Display und der jeweilige Beleuchtungskörper aufeinandergesteckt. Dann wird die gesamte Einheit einfach in eine Platine gesteckt und dort verlötet. Bitte beachten Sie, dass die 4 Pins für die Beleuchtung auch von oben verlötet werden müssen.
Achtung: Auf dem Display befinden sich 2 Schutzfolien und auf der Beleuchtung jeweils eine. Diese sollen während oder nach der Fertigung entfernt werden.

Technische Änderung vorbehalten. Wir übernehmen keine Haftung für Druckfehler und Applikationsbeispiele.

EA DOG-M

ELECTRONIC ASSEMBLY

5 VERSCHIEDENEN TECHNOLOGIEN

Als Standard sind 5 verschiedene Technologien in STN und FSTN lieferbar:

Displaytyp	Technologie	optionale Beleuchtung	Lesbarkeit	Displayfarbe unbeleuchtet	Displayfarbe mit Beleuchtung	empfohlene Beleuchtung
	FSTN pos. transflektiv	mit und ohne Beleuchtungskörper zu verwenden	auch bei abgeschalteter Bel. lesbar	schwarz auf weiß	schwarz auf Beleuchtungsfarbe	weiß, blau
	STN pos. gelb/grün transmissiv	Beleuchtungskörper erforderlich	auch bei abgeschalteter Bel. lesbar	dunkelgrün auf gelb/grün	schwarz auf gelb/grün	gelb/grün
	STN neg. blau transmissiv	nur beleuchtet zu verwenden	---	---	Beleuchtungsfarbe auf blauem Hintergrund	weiß, gelb/grün
	FSTN neg. transmissiv	nur beleuchtet zu verwenden	---	---	Beleuchtungsfarbe auf schwarzem Hintergrund	weiß
	STN pos. gelb/grün reflektiv	keine Beleuchtung möglich	ohne Beleuchtung bestens lesbar	dunkelgrün auf gelb/grün	---	---

3 VERSCHIEDENE DISPLAYS

Alle 3 Displays sind in den oben genannten Technologien erhältlich:

5 VERSCHIEDENE BELEUCHTUNGEN

Zur Anpassung an unterschiedlichste Designs stehen 5 verschiedene Beleuchtungsfarben zur Auswahl. Die effektivste und gleichzeitig hellste Beleuchtung ist die weiße EA LED55x31-W.

EA LED55x31-W
Weiß

EA LED55x31-G
Gelb/Grün

EA LED55x31-B
Blau

EA LED55x31-R
Rot

EA LED55x31-A
Amber

Wenn Sie auf dieser Seite nur schwarz/weiß Darstellungen sehen: das farbige Datenblatt finden Sie im Internet unter http://www.lcd-module.de/deu/pdf/doma/dogm.pdf

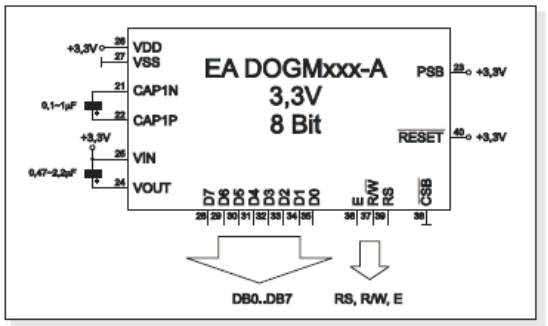
3

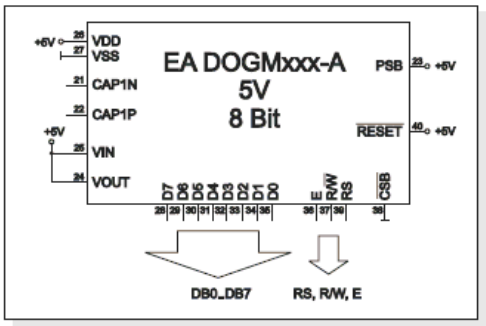
EA DOG-M

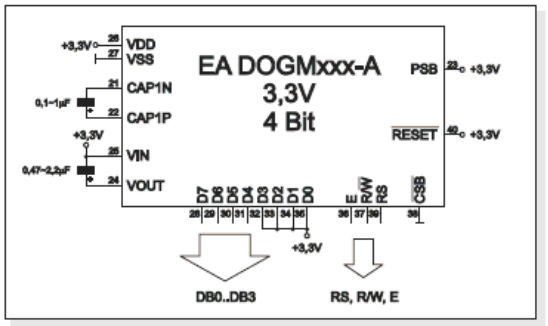
ELECTRONIC ASSEMBLY

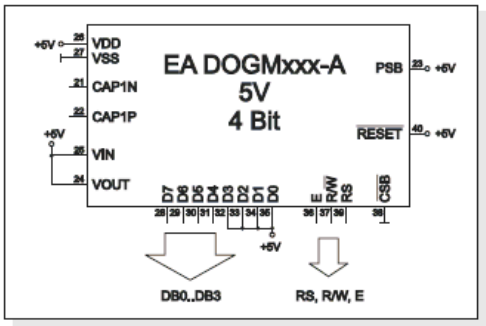
APPLIKATIONSBEISPIELE

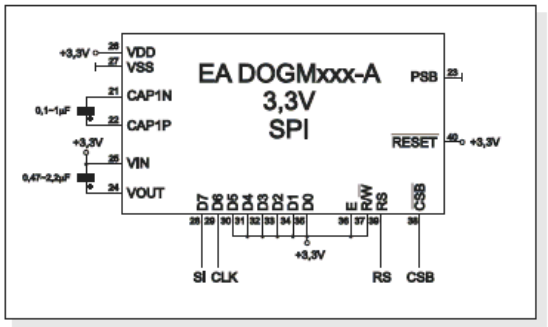
Je nach Interface und Versorgungsspannung ist eine unterschiedliche Beschaltung zu wählen. Im 3,3V Betrieb sind 2 zusätzliche Kondensatoren erforderlich.
Bitte beachten Sie, dass aufgrund der COG-Technik die Strombelastbarkeit der Ausgänge begrenzt ist. Es kann dadurch bei größerer Buslast zu Signalverschleifungen und unsauberen Pegeln kommen. Im Zweifelsfall sind zusätzliche Pull-Down Widerstände (8051) erforderlich, oder es müssen zusätzliche Waits/NOP's eingefügt werden.

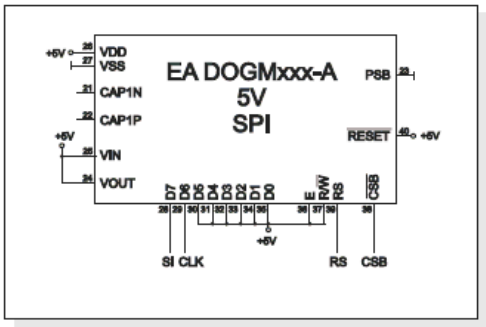












EA DOG-M

ELECTRONIC ASSEMBLY

ZEICHENSATZ

Der unten abgebildete Zeichensatz ist integriert. Zusätzlich können 8 eigene Zeichen frei definiert werden.

Eine detaillierte Beschreibung des hier integrierten Kontrollers ST7036 finden Sie im Internet unter http://www.lcd-module.de/eng/pdf/zubehoer/st7036.pdf

5

EA DOG-M

ELECTRONIC ASSEMBLY

BEFEHLSTABELLEN

Instruction	Instruction Code										Description	Instruction Execution Time		
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		OSC= 380kHz	OSC= 540kHz	OSC= 700kHz
Clear Display	0	0	0	0	0	0	0	0	0	1	Write "20H" to DDRAM. and set DDRAM address to "00H" from AC	1.08 ms	0.76 ms	0.59 ms
Return Home	0	0	0	0	0	0	0	0	1	x	Set DDRAM address to "00H" from AC and return cursor to its original position if shifted. The contents of DDRAM are not changed.	1.08 ms	0.76 ms	0.59 ms
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	Sets cursor move direction and specifies display shift. These operations are performed during data write and read.	26.3 µs	18.5 µs	14.3 µs
Display ON/OFF	0	0	0	0	0	0	1	D	C	B	D=1:entire display on C=1:cursor on B=1:cursor position on	26.3 µs	18.5 µs	14.3 µs
Function Set	0	0	0	0	1	DL	N	DH	IS2	IS1	DL: interface data is 8/4 bits N: number of line is 2/1 DH: double height font IS[2:1]: instruction table select	26.3 µs	18.5 µs	14.3 µs
Set DDRAM Address	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Set DDRAM address in address counter	26.3 µs	18.5 µs	14.3 µs
Read Busy Flag and Address	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Whether during internal operation or not can be known by reading BF. The contents of address counter can also be read.	0	0	0
Write Data to RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0	Write data into internal RAM (DDRAM/CGRAM/ICONRAM)	26.3 µs	18.5 µs	14.3 µs
Read Data from RAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0	Read data from internal RAM (DDRAM/CGRAM/ICONRAM)	26.3 µs	18.5 µs	14.3 µs
Instruction table 0(IS[2:1]=[0,0])														
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	X	X	S/C and R/L: Set cursor moving and display shift control bit, and the direction, without changing DDRAM data.	26.3 µs	18.5 µs	14.3 µs
Set CGRAM	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	Set CGRAM address in address counter	26.3 µs	18.5 µs	14.3 µs
Instruction table 1(IS[2:1]=[0,1])														
Bias Set	0	0	0	0	0	1	BS	1	0	FX	BS=1:1/4 bias BS=0:1/5 bias FX: fixed on high in 3-line application and fixed on low in other applications.	26.3 µs	18.5 µs	14.3 µs
Set ICON Address	0	0	0	1	0	0	AC3	AC2	AC1	AC0	Set ICON address in address counter.	26.3 µs	18.5 µs	14.3 µs
Power/ICON Control/ Contrast Set	0	0	0	1	0	1	Ion	Bon	C5	C4	Ion: ICON display on/off Bon: set booster circuit on/off C5,C4: Contrast set for internal follower mode.	26.3 µs	18.5 µs	14.3 µs
Follower Control	0	0	0	1	1	0	Fon	Rab 2	Rab 1	Rab 0	Fon: set follower circuit on/off Rab2~0: select follower amplified ratio.	26.3 µs	18.5 µs	14.3 µs
Contrast Set	0	0	0	1	1	1	C3	C2	C1	C0	Contrast set for internal follower mode.	26.3 µs	18.5 µs	14.3 µs
Instruction table 2(IS[2:1]=[1,0])														
Double Height Position Select	0	0	0	0	0	1	UD	X	x	x	UD: Double height position select	26.3 µs	18.5 µs	14.3 µs
Reserved	0	0	0	1	X	X	X	X	X	X	Do not use (reserved for test)	26.3 µs	18.5 µs	14.3 µs

Eine detaillierte Beschreibung des hier integrierten Kontrollers ST7036 finden Sie im Internet unter http://www.lcd-module.de/eng/pdf/zubehoer/st7036.pdf

EA DOG-M

ELECTRONIC ASSEMBLY

INITIALISIERUNGSBEISPIELE

EA DOGM081

INITIALISIERUNGSBEISPIEL FÜR 8 Bit / 3.3V
EA DOGM081

Befehl	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Hex	Bemerkung
Function Set	0	0	0	0	1	1	0	0	0	1	$31	8-Bit Datenlänge, 1 Zeile, Instruction table 1
Bias Set	0	0	0	0	0	1	0	1	0	0	$14	BS: 1/5, 1-zeiliges LCD
Power Control	0	0	0	1	0	1	0	1	0	1	$55	Booster ein, Kontrast C5, C4setzen
Follower Control	0	0	0	1	1	0	1	1	0	1	$6D	Spannungsfolger und Verstärkung setzen
Contrast Set	0	0	0	1	1	1	1	1	0	0	$7C	Kontrast C3, C2, C1 setzen
Display ON/OFF	0	0	0	0	0	0	1	1	1	1	$0F	Display ein, Cursor ein, Cursor blinken
Clear Display	0	0	0	0	0	0	0	0	0	1	$01	Display löschen, Cursor Home
Entry Mode Set	0	0	0	0	0	0	0	1	1	0	$06	Cursor Auto-increment

Initialisierung für 3,3V

INITIALISIERUNGSBEISPIEL FÜR 8 Bit / 5V
EA DOGM081

Befehl	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Hex	Bemerkung
Function Set	0	0	0	0	1	1	0	0	0	1	$31	8-Bit Datenlänge, 1 Zeile, Instruction table 1
Bias Set	0	0	0	0	0	1	1	1	0	0	$1C	BS: 1/4, 1-zeiliges LCD
Power Control	0	0	0	1	0	1	0	0	0	1	$51	Booster aus, Kontrast C5, C4setzen
Follower Control	0	0	0	1	1	0	1	0	1	0	$6A	Spannungsfolger und Verstärkung setzen
Contrast Set	0	0	0	1	1	1	0	1	0	0	$74	Kontrast C3, C2, C1 setzen
Display ON/OFF	0	0	0	0	0	0	1	1	1	1	$0F	Display ein, Cursor ein, Cursor blinken
Clear Display	0	0	0	0	0	0	0	0	0	1	$01	Display löschen, Cursor Home
Entry Mode Set	0	0	0	0	0	0	0	1	1	0	$06	Cursor Auto-increment

Initialisierung für 5V

EA DOGM162

INITIALISIERUNGSBEISPIEL FÜR 8 Bit / 3.3V
EA DOGM162

Befehl	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Hex	Bemerkung
Function Set	0	0	0	0	1	1	1	0	0	1	$39	8-Bit Datenlänge, 2 Zeilen, Instruction table 1
Bias Set	0	0	0	0	0	1	0	1	0	0	$14	BS: 1/5, 2-zeiliges LCD
Power Control	0	0	0	1	0	1	0	1	0	1	$55	Booster ein, Kontrast C5, C4setzen
Follower Control	0	0	0	1	1	0	1	1	0	1	$6D	Spannungsfolger und Verstärkung setzen
Contrast Set	0	0	0	1	1	1	1	0	0	0	$78	Kontrast C3, C2, C1 setzen
Display ON/OFF	0	0	0	0	0	0	1	1	1	1	$0F	Display ein, Cursor ein, Cursor blinken
Clear Display	0	0	0	0	0	0	0	0	0	1	$01	Display löschen, Cursor Home
Entry Mode Set	0	0	0	0	0	0	0	1	1	0	$06	Cursor Auto-increment

Initialisierung für 3,3V

INITIALISIERUNGSBEISPIEL FÜR 8 Bit / 5V
EA DOGM162

Befehl	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Hex	Bemerkung
Function Set	0	0	0	0	1	1	1	0	0	1	$39	8-Bit Datenlänge, 2 Zeilen, Instruction table 1
Bias Set	0	0	0	0	0	1	1	1	0	0	$1C	BS: 1/4, 2-zeiliges LCD
Power Control	0	0	0	1	0	1	0	0	1	0	$52	Booster aus, Kontrast C5, C4setzen
Follower Control	0	0	0	1	1	0	1	0	0	1	$69	Spannungsfolger und Verstärkung setzen
Contrast Set	0	0	0	1	1	1	0	1	0	0	$74	Kontrast C3, C2, C1 setzen
Display ON/OFF	0	0	0	0	0	0	1	1	1	1	$0F	Display ein, Cursor ein, Cursor blinken
Clear Display	0	0	0	0	0	0	0	0	0	1	$01	Display löschen, Cursor Home
Entry Mode Set	0	0	0	0	0	0	0	1	1	0	$06	Cursor Auto-increment

Initialisierung für 5V

EA DOGM163

INITIALISIERUNGSBEISPIEL FÜR 8 Bit / 3.3V
EA DOGM163

Befehl	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Hex	Bemerkung
Function Set	0	0	0	0	1	1	1	0	0	1	$39	8-Bit Datenlänge, 2 Zeilen, Instruction table 1
Bias Set	0	0	0	0	0	1	0	1	0	1	$15	BS: 1/5, 3-zeiliges LCD
Power Control	0	0	0	1	0	1	0	1	0	1	$55	Booster ein, Kontrast C5, C4setzen
Follower Control	0	0	0	1	1	0	1	1	1	0	$6E	Spannungsfolger und Verstärkung setzen
Contrast Set	0	0	0	1	1	1	0	0	1	0	$72	Kontrast C3, C2, C1 setzen
Display ON/OFF	0	0	0	0	0	0	1	1	1	1	$0F	Display ein, Cursor ein, Cursor blinken
Clear Display	0	0	0	0	0	0	0	0	0	1	$01	Display löschen, Cursor Home
Entry Mode Set	0	0	0	0	0	0	0	1	1	0	$06	Cursor Auto-increment

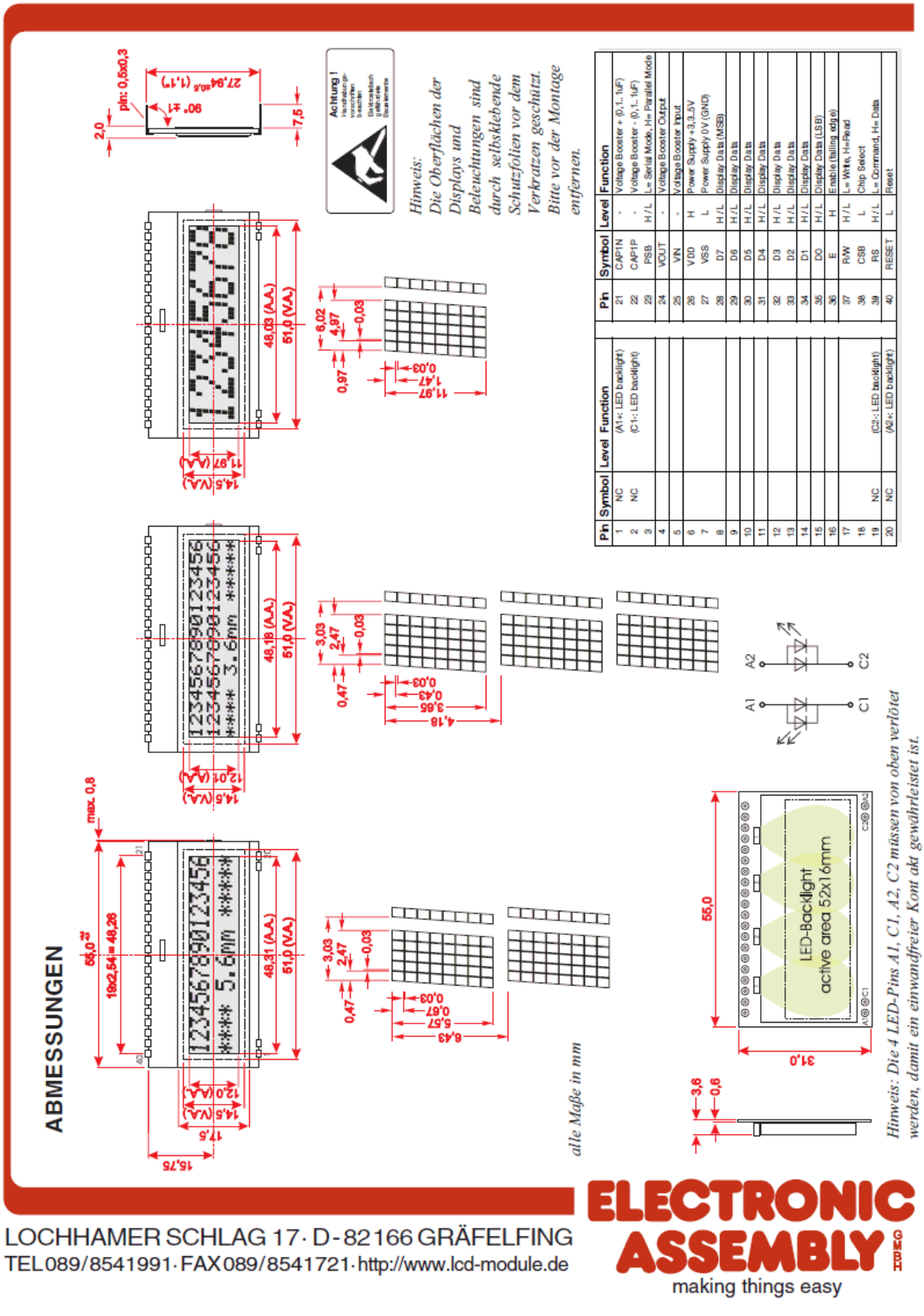
Initialisierung für 3,3V

INITIALISIERUNGSBEISPIEL FÜR 8 Bit / 5V
EA DOGM163

Befehl	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Hex	Bemerkung
Function Set	0	0	0	0	1	1	1	0	0	1	$39	8-Bit Datenlänge, 2 Zeilen, Instruction table 1
Bias Set	0	0	0	0	0	1	1	1	0	1	$1D	BS: 1/4, 3-zeiliges LCD
Power Control	0	0	0	1	0	1	0	0	0	0	$50	Booster aus, Kontrast C5, C4setzen
Follower Control	0	0	0	1	1	0	1	1	0	0	$6C	Spannungsfolger und Verstärkung setzen
Contrast Set	0	0	0	1	1	1	1	1	0	0	$7C	Kontrast C3, C2, C1 setzen
Display ON/OFF	0	0	0	0	0	0	1	1	1	1	$0F	Display ein, Cursor ein, Cursor blinken
Clear Display	0	0	0	0	0	0	0	0	0	1	$01	Display löschen, Cursor Home
Entry Mode Set	0	0	0	0	0	0	0	1	1	0	$06	Cursor Auto-increment

Initialisierung für 5V

EA DOG-M



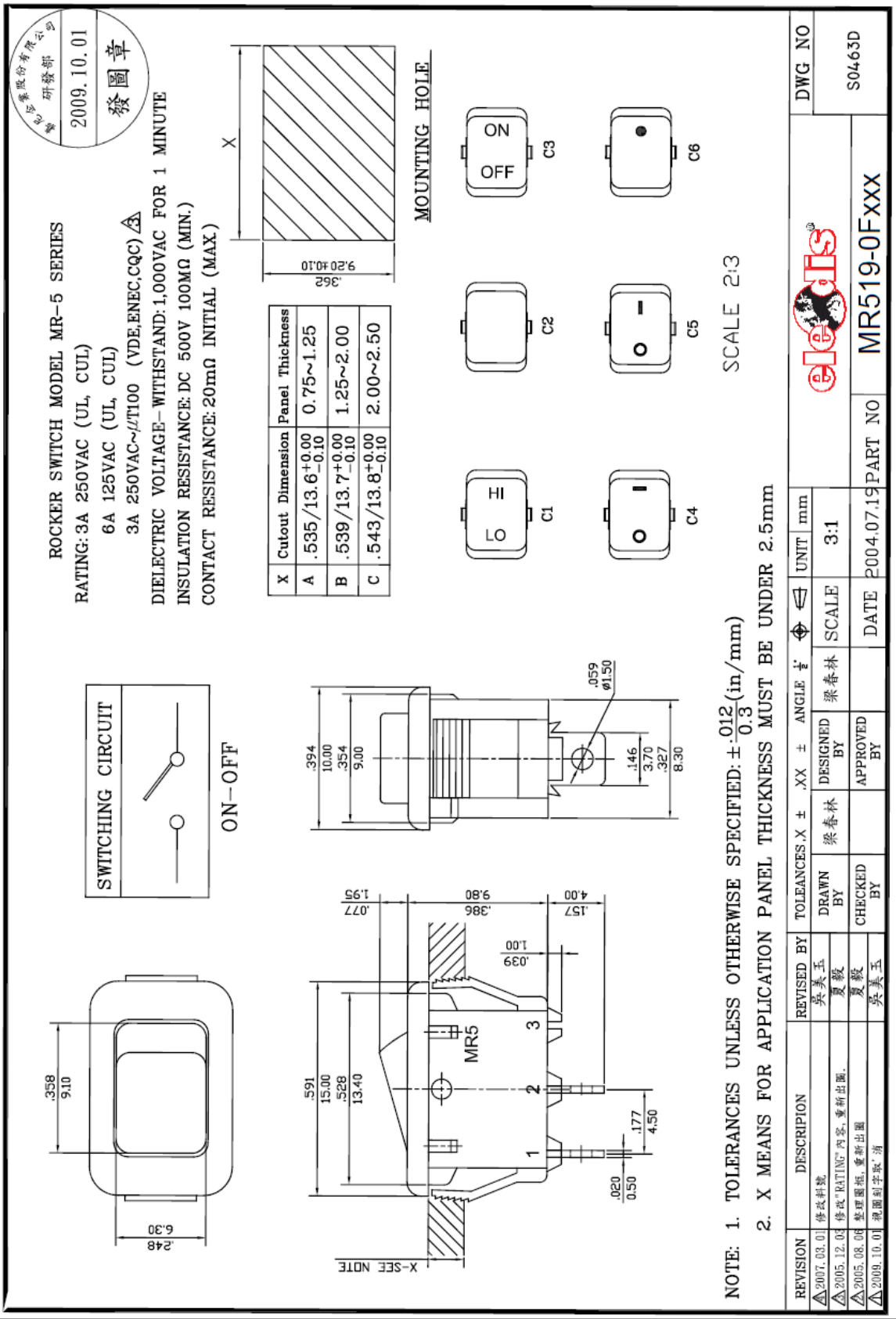
Pin	Symbol	Level	Function
1	NC		(A1+: LED backlight)
2	NC		(C1-: LED backlight)
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			
17			
18			
19	NC		(C2-: LED backlight)
20	NC		(A2+: LED backlight)

Pin	Symbol	Level	Function
21	CAP1N	-	Voltage Booster + (0,1..1uF)
22	CAP1P	-	Voltage Booster - (0,1..1uF)
23	PSB	H/L	L= Serial Mode, H= Parallel Mode
24	VOUT	-	Voltage Booster Output
25	VIN	-	Voltage Booster Input
26	VDD	H	Power Supply +3,3..5V
27	VSS	L	Power Supply 0V (GND)
28	D7	H/L	Display Data (MSB)
29	D6	H/L	Display Data
30	D5	H/L	Display Data
31	D4	H/L	Display Data
32	D3	H/L	Display Data
33	D2	H/L	Display Data
34	D1	H/L	Display Data
35	D0	H/L	Display Data (LSB)
36	E	H	Enable (falling edge)
37	R/W	H/L	L= Write, H=Read
38	CSB	L	Chip Select
39	RS	H/L	L= Command, H= Data
40	RESET	L	Reset

LOCHHAMER SCHLAG 17 · D-82166 GRÄFELFING
TEL 089/8541991 · FAX 089/8541721 · http://www.lcd-module.de

ELECTRONIC ASSEMBLY GMBH
making things easy

Hauptschalter:



Gehäuse: