

Hausarbeit Embedded-Systems

Sommersemester 2011

**Geschwindigkeitserfassung
am Kicker (Tischfußballtisch)**

Nils Feldkämper

Betreuer: Prof. Dr.-Ing. Jürgen Wübbelmann

Inhaltsverzeichnis

1	Aufgabenstellung	5
2	Einführung.....	5
2.1	Wo für steht „Arduino“	6
3	Hardware im Detail.....	7
3.1	Mikrokontroller Board	8
3.1.1	Details:	8
3.3	Hauptplatine	9
3.3.1	Spannungsversorgung	9
3.3.2	Mikrokontroller mit Sockel für Lichtschranken, LCD und Tastatur	10
3.3.3	Materialkosten Hauptplatine.....	10
3.3.4	Bilder der Hauptplatine	11
3.4	LCD	12
3.4.1	LCD Verdrahtungsplan	13
3.4.2	LCD Bild im Betrieb	13
3.4.3	Materialkosten LCD.....	13
3.5	Tastatur	14
3.5.1	Materialkosten Tastatur	14
3.5.2	Tastatur Verdrahtungsplan.....	15
3.5.3	Tastatur Detail Bilder	15
3.6	Lichtschranken	17
3.6.1	Verteiler	17
3.6.2	Sender	18
3.6.3	Materialkosten Sender	19
3.6.4	Empfänger.....	21
3.6.5	Materialkosten Empfänger	22
3.7	Spannungsversorgung (extern).....	24
3.8	Materialkosten gesamte Hardware	24
4	Software im Detail	25
4.1	Softwarerestruktur.....	25
4.1.1	Includes	25
4.1.2	Definitionen	25
4.1.3	Globale Variablen.....	25
4.1.4	Globale Objekte	26
4.1.5	Prototypen	26
4.1.6	Nützliche Funktionen.....	26
4.1.7	Funktionen zum Initialisieren der Lichtschranken.....	26
4.1.8	Background Funktionen.....	26
4.1.9	Kontrollfunktionen.....	27
4.1.10	Menüfunktionen	27
4.1.11	Interruptfunktionen	27
4.1.12	Setup	27
4.1.13	Loop	27
4.2	Speicherabschätzung im SRAM	28
4.2.1	Globale Variablen.....	28
4.2.2	globale Objekte	28
4.2.3	Variablen in Funktionen.....	28

4.2.4	Gesamt	28
4.3	Speicherbedarf im Flash	29
4.4	Menüführung	29
4.5	Werte verändern	32
4.6	Lichtschranken	36
4.6.1	Geschwindigkeitsmessung	36
4.6.2	Kalibrierung	38
4.7	Bestenliste	42
4.8	Software im Anhang	42
5	Messungen	43
5.1	Messung der Geschwindigkeit mit nur einer Lichtschranke	43
5.1.1	Seite 1 des Test Programms	44
5.1.2	Seite 2 des Test Programms	45
5.2	Praktische Messung im Kicker	46
5.2.1	Versuchsaufbau mit einer Lichtschranke	47
5.2.2	Versuchsaufbau mit zwei Lichtschranken	49
5.2.3	Hilfsmittel	50
5.2.4	Durchführung der Messungen	51
6	Ergebnisse	52
6.1	Messungen mit dem Frequenzgenerator	52
6.1.1	Ergebnisse und Verbesserungen	52
6.2	Praktische Messungen im Kicker Tisch	54
7	Zeitaufwand	55
8	Interpretation	55
9	Quellen	55

Abbildungsverzeichnis

Abbildung 1:	Übersicht Hardware	7
Abbildung 2:	Arduino Mini Pro	8
Abbildung 3:	Verdrahtungsplan Hauptplatine	9
Abbildung 4:	Hauptplatine Bild 1 ohne Mikrokontroller	11
Abbildung 5:	Hauptplatine Bild 2 mit Mikrokontroller	11
Abbildung 6:	20x4 LCD HD44780 Standard	12
Abbildung 7:	Verdrahtungsplan LCD-Platine	13
Abbildung 8:	LCD Bild 1	13
Abbildung 9:	Schaltplan Tastatur	15
Abbildung 10:	Tastatur Seitenansicht	15
Abbildung 11:	Tastatur Draufsicht	16
Abbildung 12:	Tastatur fertig	16
Abbildung 13:	Verdrahtungsplan Verteiler Platine	17
Abbildung 14:	Verteilerplatine Bild 1	17
Abbildung 15:	Verdrahtungsplan Sender	18
Abbildung 16:	Senderplatine Versuch 1 Bild 1	19

Abbildung 17: Senderplatine Versuch 1 Bild 2.....	19
Abbildung 18: Senderplatine Versuch 2 Bild 1.....	20
Abbildung 19: Senderplatine Versuch 2 Bild 2 (links)	20
Abbildung 20: Senderplatine Versuch 2 Bild 3 (rechts)	20
Abbildung 21: Verdrahtungsplan Empfänger	21
Abbildung 22: Empfängerplatine Bild 1	22
Abbildung 23: Empfängerplatine Bild 2	23
Abbildung 24: Empfängerplatine Bild 3	23
Abbildung 25: Empfängerplatine Bild 4	23
Abbildung 26: externe Stromversorgung.....	24
Abbildung 27: Ebene 1 im Menü.....	29
Abbildung 28: Menüeben 2 oben	30
Abbildung 29: Menüeben 2 unten	31
Abbildung 30: Werte verändern Bild 1	32
Abbildung 31: Objektlänge verstellen Bild 1.....	33
Abbildung 32: Objektlänge verstellen Bild 2.....	34
Abbildung 33: Objektlänge verstellen Bild 3.....	35
Abbildung 34: Messung mit einer Lichtschranke.....	37
Abbildung 35: Kalibrierung, nichts gefunden	38
Abbildung 36: Kalibrierung L1 ok	39
Abbildung 37: Teil 2 Kalibrierung.....	40
Abbildung 38: Teil 2 der Kalibrierung mit Zähler	41
Abbildung 39: Bestenliste	42
Abbildung 40: praktische Messung im Kicker Makierungen	46
Abbildung 43: Sender im Kicker 1 Lichtschranke	47
Abbildung 44: Empfänger im Kicker 1 Lichtschranke	47
Abbildung 41: Aufbau des Kickers mit nur einer Lichtschranke	48
Abbildung 45: Draufsicht im Kicker 1 Lichtschranke	48
Abbildung 46: Zweilichtschranke Blickrichtung im Kicker	49
Abbildung 48: Zweilichtschranken Aufbau im Kicker	49
Abbildung 42: Digitaler Messschieber	50
Abbildung 47: Zweilichtschranken Abstandmessung	50

1 Aufgabenstellung

Es soll eine Geschwindigkeitsanzeige für einen Kicker (Tischfußball) entwickelt werden. Zudem sollen verschiedene Werte verändert werden können.

2 Einführung

Diese Aufgabenstellung klingt am Anfang sehr einfach. Es ist einem überlassen, wie tief man einsteigt, um dieses Projekt umzusetzen. Bei allen Komponenten hat man die freie Wahl. Daher ist es wichtig am Anfang festzulegen, welche Hardware verwendet werden sollte.

Als Erstes wird überlegt, welche Komponenten für eine Geschwindigkeitsanzeige in Betracht gezogen werden sollten.

Als Plattform wird der Arduino Pro Mini verwendet. Dieser ermöglicht einen leichten Umgang mit der Hardware des Mikrocontrollers. Zur Ausgabe wird ein LCD angeschlossen, welches die Einstellungen und die Geschwindigkeit anzeigen. Um Werte auf dem LCD verändern zu können, wird eine Art Tastatur benötigt.

Da die Geschwindigkeit gemessen werden soll, werden zwei Lichtschranken verwendet. Wenn der Ball im Kicker beide nacheinander unterbricht, wird über die gemessene Zeitdifferenz und den vorher eingestellten Abstand zwischen L1 und L2 die Geschwindigkeit in km/h berechnet und anschließend ausgegeben.

Während der Entwicklung ist die Idee entstanden, mit nur einer Lichtschranke über die Größe des Objektes zu messen. Diese Messung misst die Zeitdifferenz, wie lange die Lichtschranke unterbrochen ist.

Nach der groben Festlegung der Hardware wird eine Übersicht über die Softwarestruktur erstellt. Die Grundidee hierzu war es, auf eine im Sommer 2010 für ein anderes Projekte entwickelte Menüklasse zurückzugreifen. Mit dieser Klasse kann eine Menüstruktur mit beliebig vielen Unterebenen auf Basis des Nested-Set-Models erstellt werden. Jeder Menüpunkt kann über die Navigation mit einem Tastenfeld ausgewählt werden.

Da es wichtig ist, die Menüstruktur von Anfang an festzulegen, wurde folgende gewählt:

1. Start
2. Bestenliste
3. Information
4. Einstellungen
 - 4.1. Objekt Länge
 - 4.2. Abstand Lichtschranke 1 zu Lichtschranke 2 (L1<->L2)
 - 4.3. Kalibrierung
 - 4.4. Bestenliste zurücksetzen.
 - 4.5. Werkseinstellungen

Das Ziel ist es, die Einstellungsmöglichkeiten möglichst übersichtlich zu halten. Unter „Start“ befindet sich die Geschwindigkeitsmessung.

2.1 Wo für steht „Arduino“

Arduino ist eine auf Processing basierende Software, die es ermöglicht, Mikrocontroller zu programmieren. Viele Funktionen, die benötigt werden, um Ports zu setzen oder Timer zu initialisieren, sind vorgegeben. Es werden die Programmiersprachen C/C++ verwendet.

Arduino steht aber nicht nur für die Softwareumgebung. Unter dem Begriff Arduino lässt sich auch Hardware finden. Deren Name lautet zum Beispiel „Arduino Pro Mini“. Diese Hardware besteht meist aus dem ATMega 328, der auf einem Board mit ein paar Bauteilen verbaut wird. Es gibt auch andere Kontroller, die verwendet werden können.

Im Sommersemester 2009 haben 3 Studenten (Sascha Ebelt, Insa Maßmann) und ich eine Einführung über den Arduino geschrieben. Diese befindet sich im Verzeichnis „[anhang/arduino_einführung.pdf](#)“. In diesem Dokument sind alle relevanten Details ersichtlich, sodass hier nicht näher auf den Arduino eingegangen wird.

3 Hardware im Detail

Als Beginn ist es sinnvoll, eine Übersicht über die gesamte Hardware zu zeigen. Anschließend wird auf die einzelnen Komponenten näher eingegangen.

Zur folgenden Übersicht lässt sich sagen, dass alle bau miteinander verbundenen Komponenten während des Betriebs an und abgesteckt werden können, ohne dass der Mikrokontroller neu gestartet werden muss. Die rot markierte Leitung ist für das Display. Dieses wird einmal beim Start initialisiert und kann daher nicht abgezogen werden. Die grünen Leitungen sind fest angeschraubt. Bei dem orangenen Pfeil handelt es sich um die Lichtschrankenstrecke.

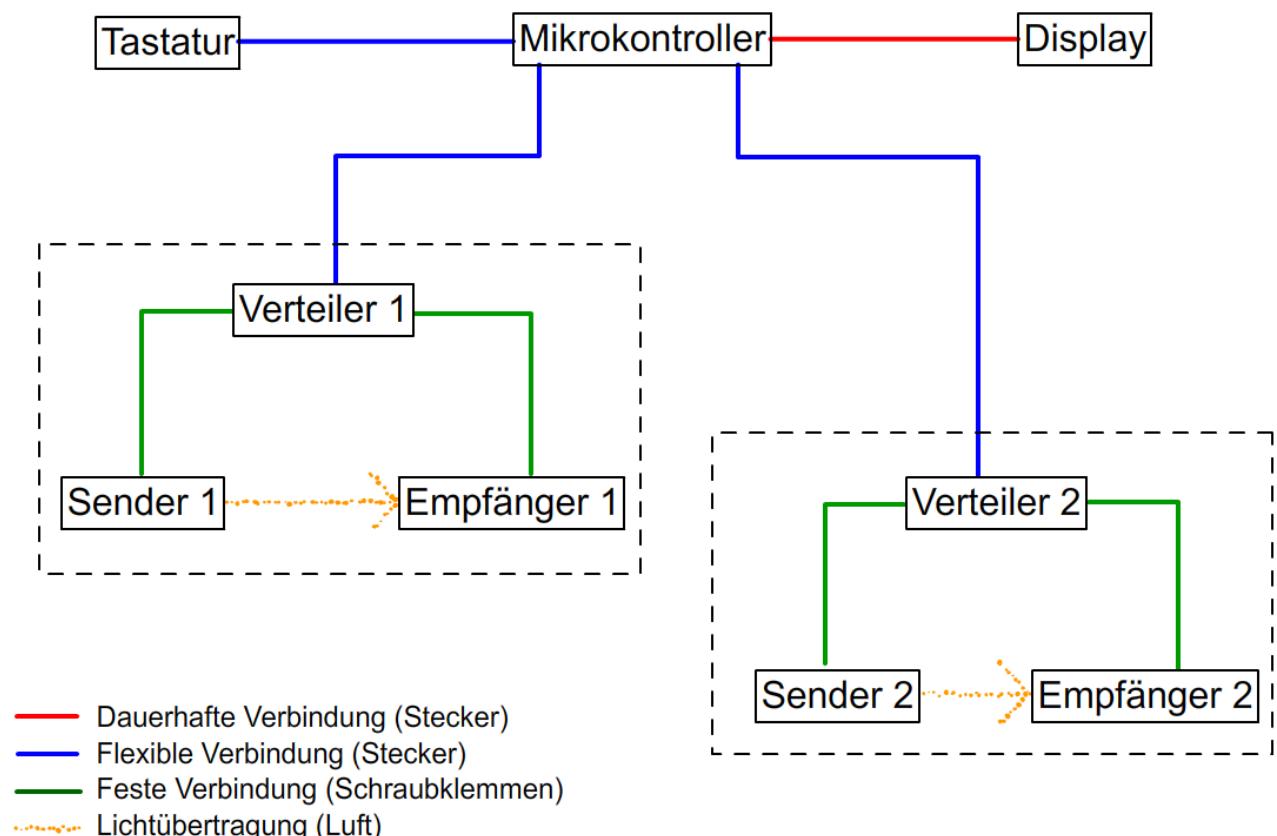


Abbildung 1: Übersicht Hardware

Bei allen folgenden Verdrahtungsplänen gelten diese Regeln:

- | | |
|---|--|
| - Rote Linien | = Leiterbahnen unterhalb der Platine |
| - Grüne Linien | = Brücken oberhalb der Platine |
| - Braune Komponenten | = Bauteile |
| - Schwarze Komponenten | = Sockel, Halterungen |
| - Schwarze Kreise ohne Füllung | = mögliche Sockel, die nicht bestückt sind |
| - Schwarze Kreise mit Füllung (schwarz) | = Steckleiste für Mikrokontroller, LCD |
| - Schwarze Kreise mit Füllung (blau) | = Verbindungen von Sockeln und Halterungen |

Im Hintergrund aller Verdrahtungspläne befindet sich ein graues Muster der Lochrasterplatten.

3.1 Mikrokontroller Board

Als Mikrokontroller wird der ATMega328 verwendet. Dieser ist auf einem Arduino Board verlötet. Dieses Board nennt sich „Arduino Pro Mini“ und sieht wie folgt aus:

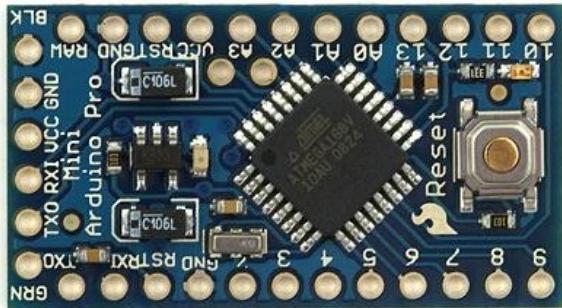


Abbildung 2: Arduino Mini Pro

Das Board wurde von der Firma „Sparkfun“ designet und produziert. Kaufen kann man es zum Beispiel bei www.watterott.com.

3.1.1 Details:

Mikrocontroller	ATmega328
Spannung	5 V
Eingangsspannung	5 - 12 V
Digital I/O Pins	14 (6 PWM)
Analog Input Pins	6
Strom pro Pin	40 mA
Flash Memory	32 KB (2 KB werden vom Bootloader verwendet)
SRAM	2 KB
EEPROM	1024 bytes
Taktrate	16 MHz

Der Mikrokontroller besitzt zudem zwei Interrupts, die für die Lichtschranken verwendet werden. Der 16 Bit Timer auf dem Board wird zur Frequenzgenerierung verwendet.

3.3 Hauptplatine

Für die Hauptplatine ist eine Schaltung entworfen worden, in der der Arduino Pro Mini als Breakout Board gesteckt wird. Auf der Platine befindet sich zudem eine kleine Spannungsversorgung, die eine Ausgangsspannung von 5V liefert. Die Schaltung wird auf einer Lochrasterplatine aufgebaut.

Als Erstes wird der Verdrahtungsplan vorgestellt, um dann näher auf die verwendeten Komponenten einzugehen.

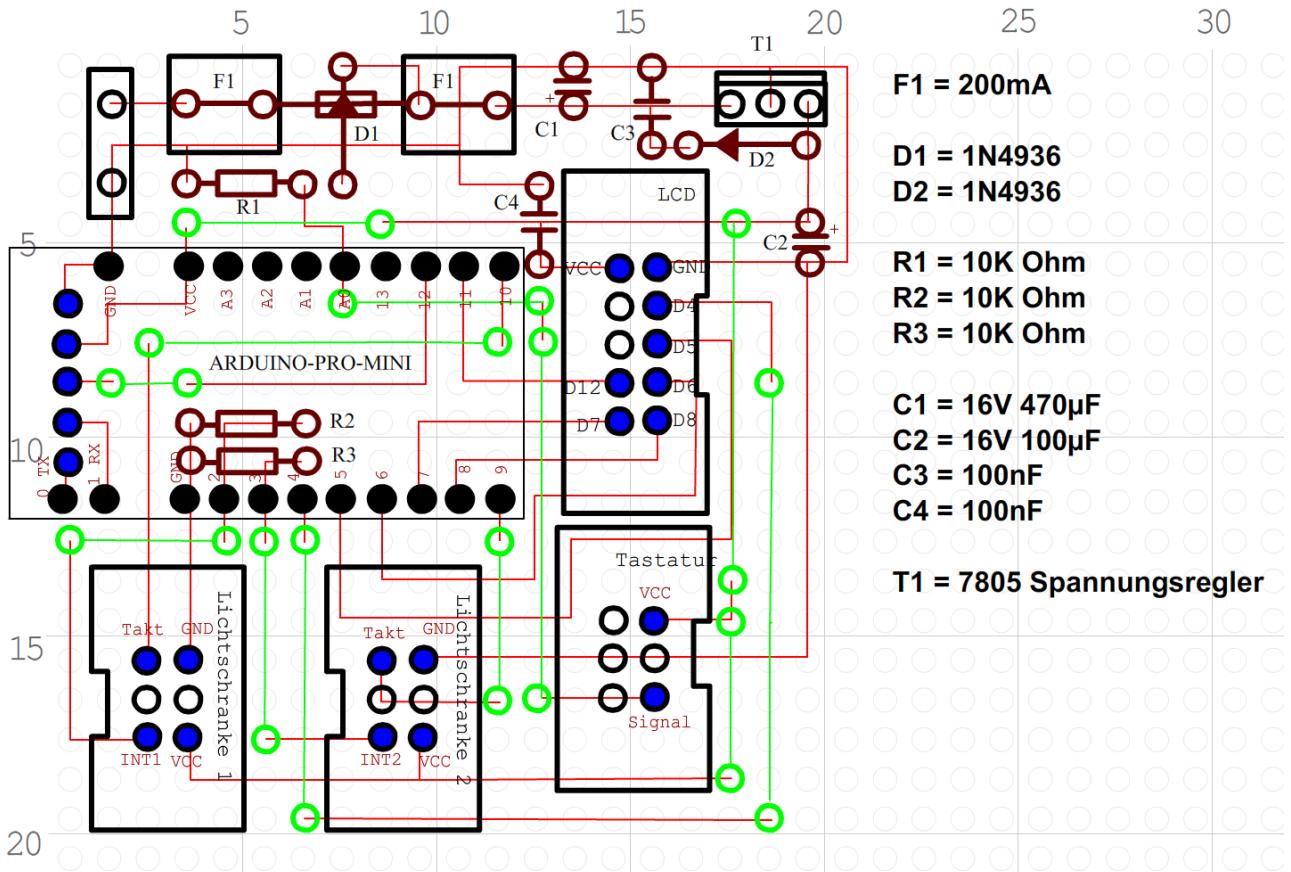


Abbildung 3: Verdrahtungsplan Hauptplatine

3.3.1 Spannungsversorgung

Im oberen Teil des Verdrahtungsplans liegt die Spannungsversorgung. Diese gliedert sich wiederum in zwei Teile:

- Schutzschaltung
- Spannungsversorgung

Die Schutzschaltung besteht aus der Sicherung (F1) und der Diode (D1). Die Sicherung ist so gewählt, dass ein Kurzschluss oder eine Überbelastung der Schaltung die Sicherung auslöst. Die Diode D1 sorgt dafür, dass die Klemmen am Eingang nicht vertauscht werden können. Sobald diese vertauscht werden, entsteht ein Kurzschluss und die Sicherung löst aus.

Die Spannungsversorgung wird durch einen 7805 Spannungswandler realisiert. Zur Stabilisierung der Spannung befinden sich jeweils zwei Kondensatoren vor dem 7805 (C1, C3) und dahinter (C2, C4). Diese gäten die Spannung, sobald Spitzen auftreten. Die Diode D2 leitet Überspannungen die in der Schaltung entstehen zurück ins Netz (externe Stromversorgung). Damit wird verhindert, dass der Mikrocontroller durch eine Überspannung in der Schaltung zerstört wird.

Am Spannungswandler sind kleine Kühlkörper aus Kupfer angeklebt worden, damit dieses Bauteil nicht überhitzt. Somit wird gewährleistet, dass die Schaltung stundenlang betrieben werden kann.

3.3.2 Mikrokontroller mit Sockel für Lichtschranken, LCD und Tastatur

Im unteren Teil des Verdrahtungsplans sieht man die Sockel für verschiedene Komponenten:

- LCD
- Tastatur
- Lichtschranke 1
- Lichtschranke 2
- Mikrokontroller
- 5 ausgeführte Ports des Mikrokontrollers (GND, VCC, D12, D1 (rx), D0 (tx))

Der Widerstand R1 dient als Pull-down-Widerstand für die Tastatur. Somit wird gewährleistet, dass der Sockel und die Leiterbahn nicht als Antenne fungieren und Störsignale empfangen, wenn keine Tastatur angeschlossen ist. Dass gleiche gilt für die Widerstände R2 und R3, die die Interrupts 1 und 2 für die Lichtschranken schützen.

Die Tastatur sowie die beiden Lichtschranken können während des Betriebes ab- und angesteckt werden.

3.3.3 Materialkosten Hauptplatine

Sicherung	= 0,50 €	* 1	= 0,50 €
Arduino Pro Mini 5V	= 14,95 €	* 1	= 14,95 €
7805	= 0,29 €	* 1	= 0,29 €
Kondensator C1	= 0,24 €	* 1	= 0,24 €
Kondensator C2	= 0,15 €	* 1	= 0,15 €
Kondensatoren C3,C4	= 0,12 €	* 2	= 0,24 €
Sockel	= 0,20 €	* 4	= 0,80 €
Klemme	= 0,15 €	* 1	= 0,15 €
Dioden	= 0,07 €	* 2	= 0,14 €
Widerstand	= 0,07 €	* 3	= 0,21 €
Pfostensteckersockel	= 0,20 €	* 1	= 0,20 €
Platine und Zubehör	= 1,50 €	* 1	= 1,50 €
<hr/>			
Gesamt:			= 19,37 €

3.3.4 Bilder der Hauptplatine

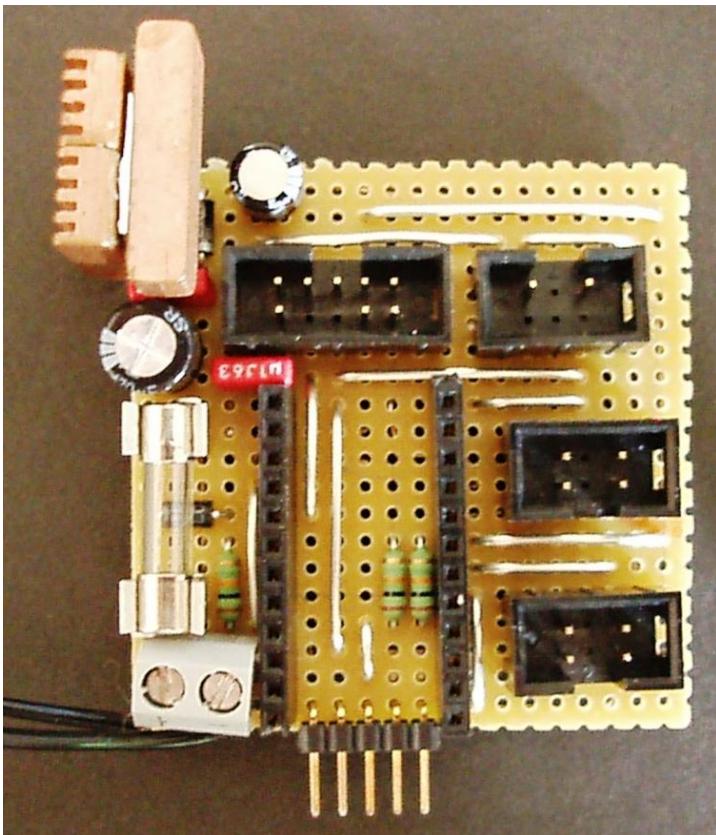


Abbildung 4: Hauptplatine Bild 1 ohne Mikrokontroller

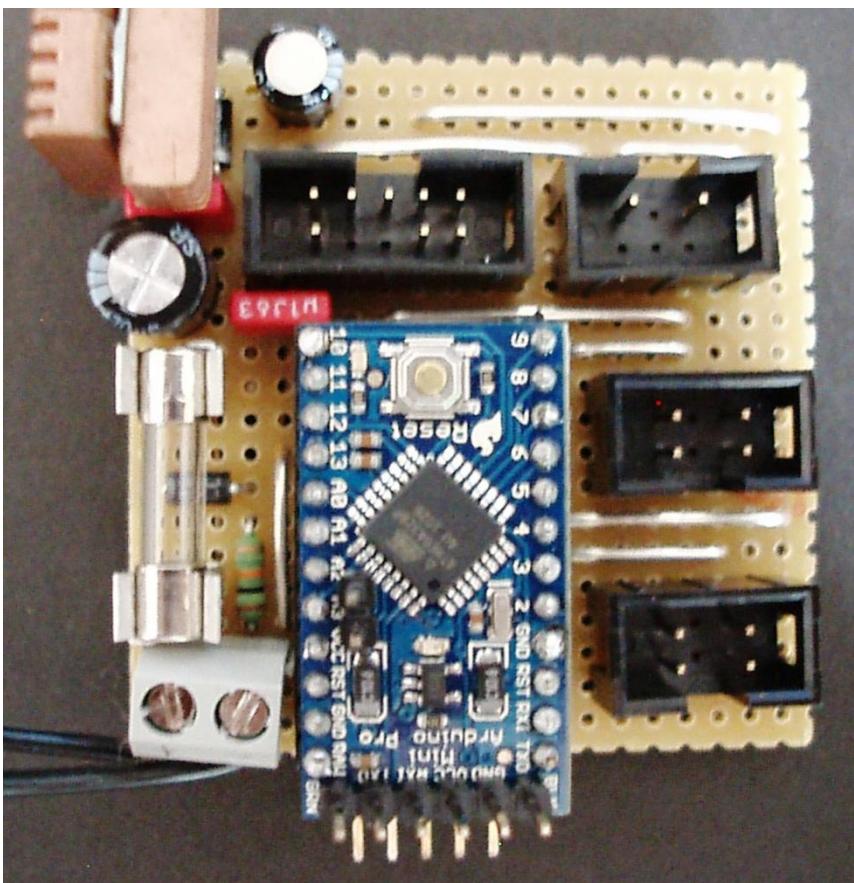


Abbildung 5: Hauptplatine Bild 2 mit Mikrokontroller

3.4 LCD

Als LCD wird ein 20 Zeichen, vier Zeilen (20x4) Display mit blauer Hintergrundbeleuchtung sowie weißer Schrift verwendet. Das LCD wird im 4Bit – Modus betrieben damit weniger I/O Pins benötigt werden. Es unterstützt den Standard HD 47780 und kann acht selbst definierte Zeichen verwenden.



i

Abbildung 6: 20x4 LCD HD44780 Standard

Das LCD hat folgende Eigenschaften:

- 4 Zeilen mit 20 Zeichen
- Blaue Hintergrundbeleuchtung
- Weiße Schrift
- HD 44780 Standard wird unterstützt.
- 8 bit / 4 bit Betrieb möglich
- 8 selbst generierte Zeichen können dargestellt werden

3.4.1 LCD Verdrahtungsplan

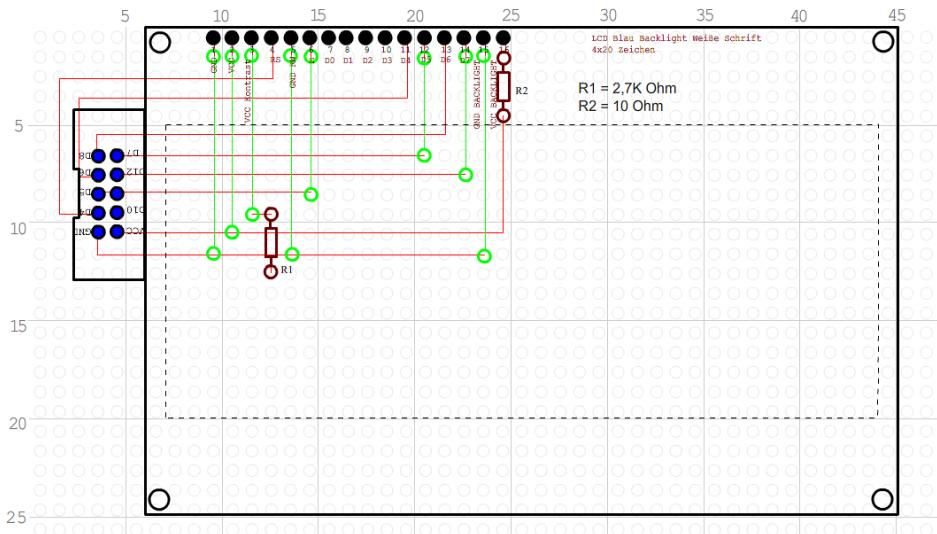


Abbildung 7: Verdrahtungsplan LCD-Platine

3.4.2 LCD Bild im Betrieb



Abbildung 8: LCD Bild 1

3.4.3 Materialkosten LCD

LCD	= 19,99 €	* 1	= 19,99 €
Widerstand	= 0,07 €	* 2	= 0,14 €
Pfostensteckersockel	= 0,20 €	* 1	= 0,20 €
Platine und Zubehör	= 0,50 €	* 1	= 0,50 €
<hr/>			
Gesamt:		= 20,83 €	

3.5 Tastatur

Als Tastatur werden sechs Taster bezeichnet. Diese sind so verschaltet, dass sich für jeden Tastendruck ein anderer Spannungswert am analogen Pin des Mikrocontrollers ergibt. Dies wird durch verschiedene große Widerstände realisiert.

Der Analog-Digital Wandler wandelt eine Spannung zwischen 0 - 5V in einen digitalen Wert zwischen 0-1023 um. Das heißt, jeder Schritt im Digitalen entspricht 0,00488 V.

Da nicht alle Widerstandswerte günstig bereitgestellt werden können, wurde durch Ausprobieren und geschicktes Abwägen der digitalen Werte folgende Widerstandstabelle aufgestellt:

Widerstand	Wert	Digitaler Wert
R1	330 Ohm	991
R2	1500 Ohm	891
R3	3900 Ohm	737
R4	5600 Ohm	657
R5	8200 Ohm	560
R6	12000 Ohm	463

Der Abstand sollte mit den Widerständen der E12 Reihe mit zehn Prozent Toleranz realisiert werden. Diese sind besonders günstig, haben aber den Nachteil einer Schwankung des Wertes um plus minus zehn Prozent. Aus dem Grund wurde ein Bereich um jeden digitalen Wert von (x-35, x+35) gelegt. Durch Ausprobieren wurde festgestellt, dass diese Werte sehr gut funktionieren.

Somit ergibt sich folgende Digitale-/Spannungstabelle:

Widerstand	digital	min	Max	analog	min	Max
R1	991	956	1023	4,836 V	4,665 V	5,000 V
R2	891	856	926	4,348 V	4,177 V	4,519 V
R3	737	702	772	3,597 V	3,426 V	3,767 V
R4	657	622	692	3,206 V	3,035 V	3,377 V
R5	560	525	595	2,733 V	2,562 V	2,904 V
R6	463	428	498	2,259 V	2,089 V	2,430 V

Diese Bereiche decken zudem kleine Spannungsschwankungen, die durch die Versorgung auftreten können, ab.

Beim Bau der Tastatur musste zwischen den Tastern die Lücken mit Füllmaterial gestopft werden, damit beim Drucken der Taster die Papierabdeckung nicht zerstört wird.

3.5.1 Materialkosten Tastatur

Taster	= 0,08 €	* 6	= 0,48 €
Widerstand	= 0,07 €	* 6	= 0,42 €
Pfostensteckersockel	= 0,20 €	* 1	= 0,20 €
Platine und Zubehör	= 0,50 €	* 1	= 0,50 €
<hr/>			
Gesamt:			= 1,60 €

3.5.2 Tastatur Verdrahtungsplan

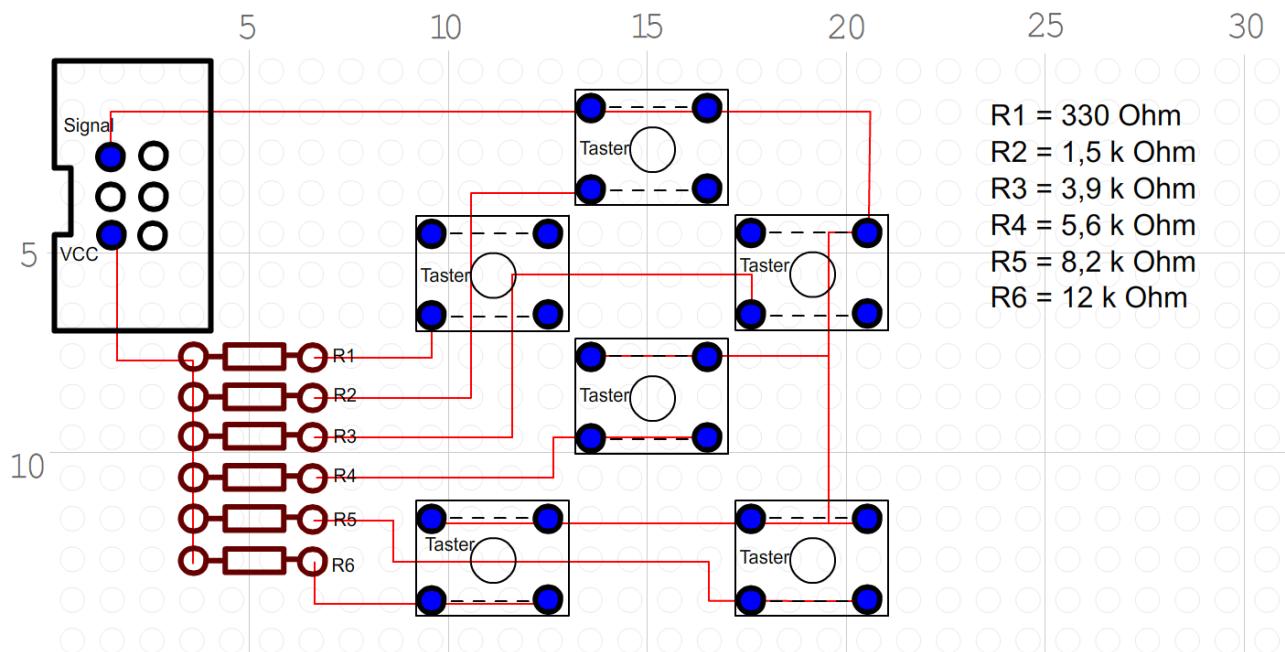


Abbildung 9: Schaltplan Tastatur

3.5.3 Tastatur Detail Bilder

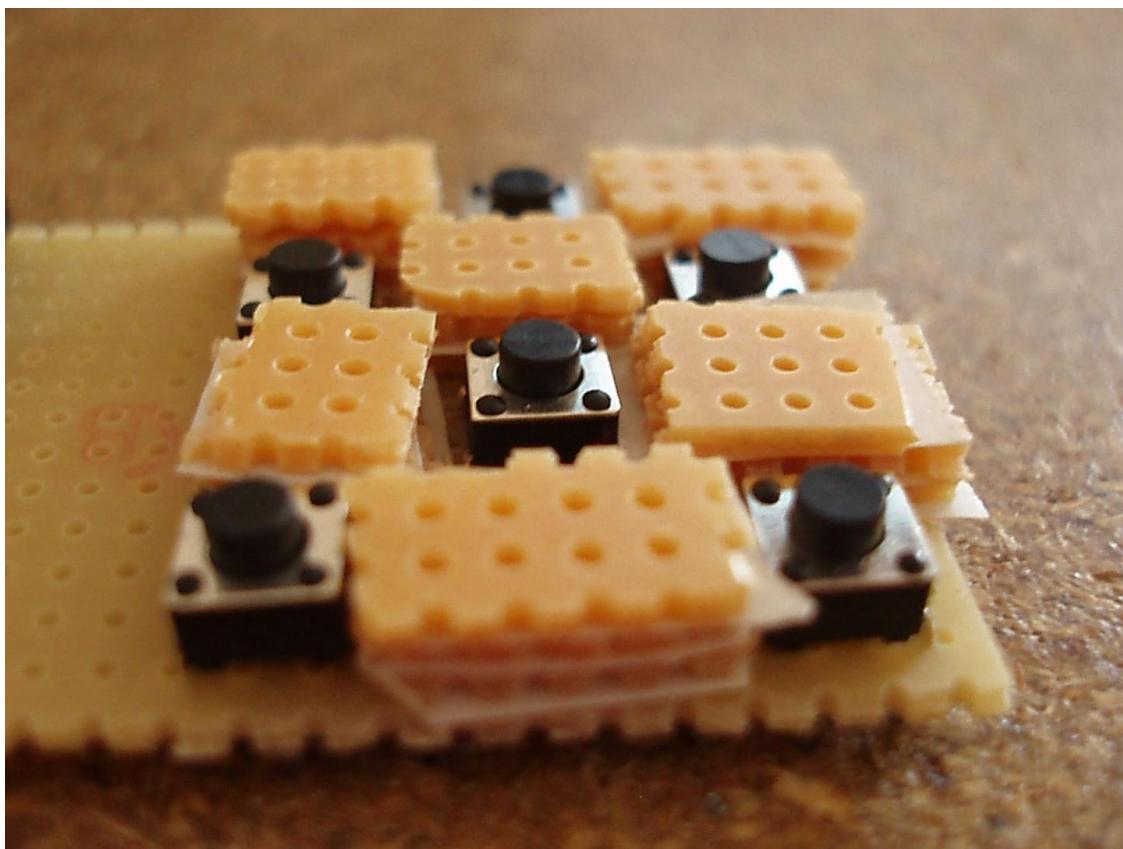


Abbildung 10: Tastatur Seitenansicht

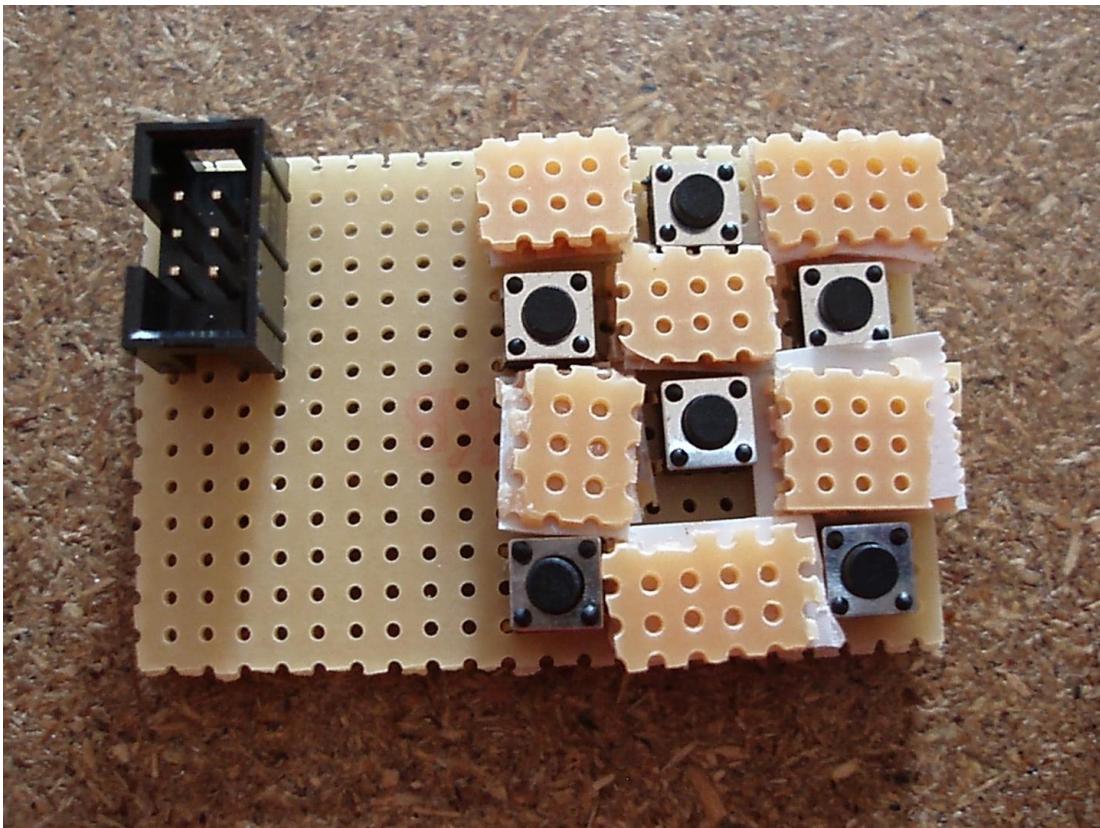


Abbildung 11: Tastatur Draufsicht

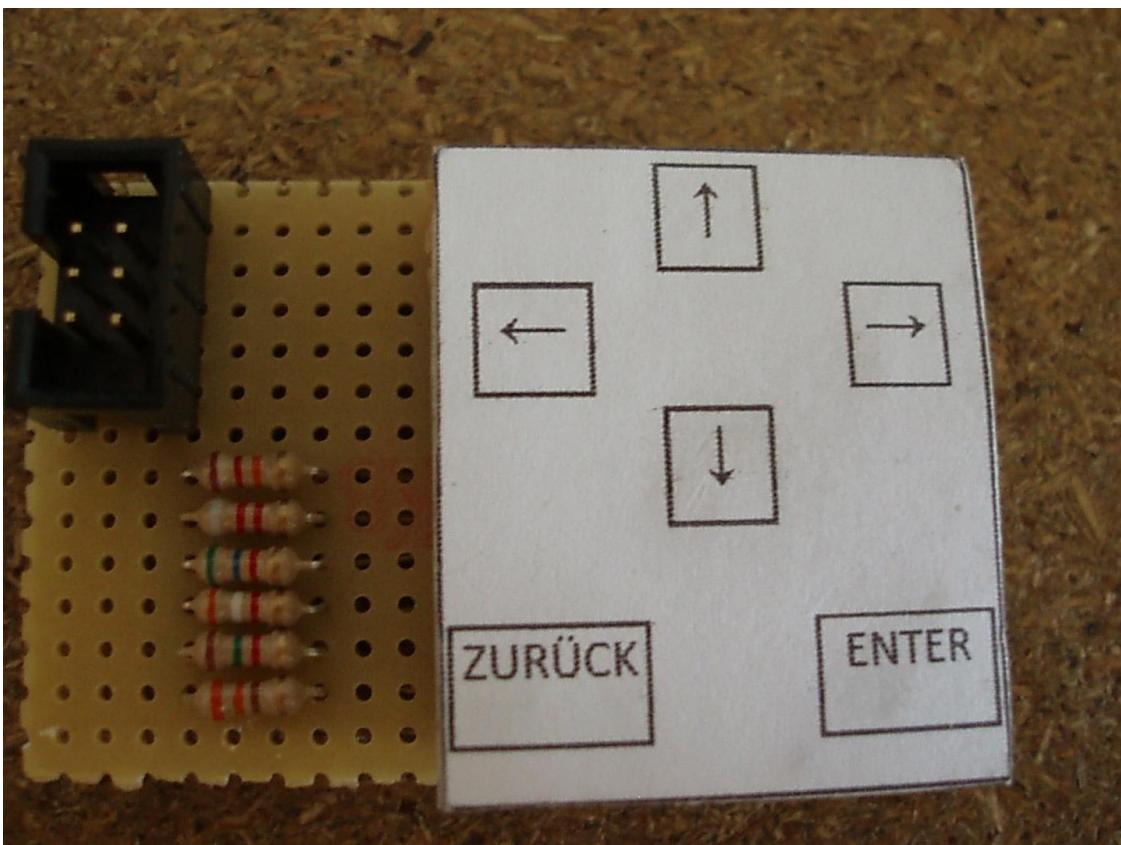


Abbildung 12: Tastatur fertig

3.6 Lichtschranken

Jede Lichtschranke besteht aus drei Modulen.

3.6.1 Verteiler

Der Verteiler stellt eine Verbindung zwischen Hauptplatine, Sender und Empfänger dar. Auf ihm werden die Leitungen des Senders und Empfängers gebündelt und zur Hauptplatine weitergeleitet.

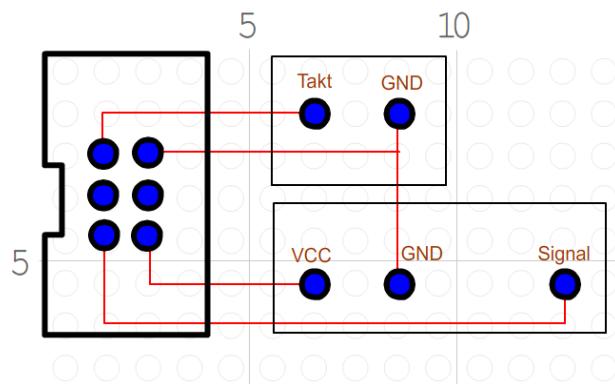


Abbildung 13: Verdrahtungsplan Verteiler Platine

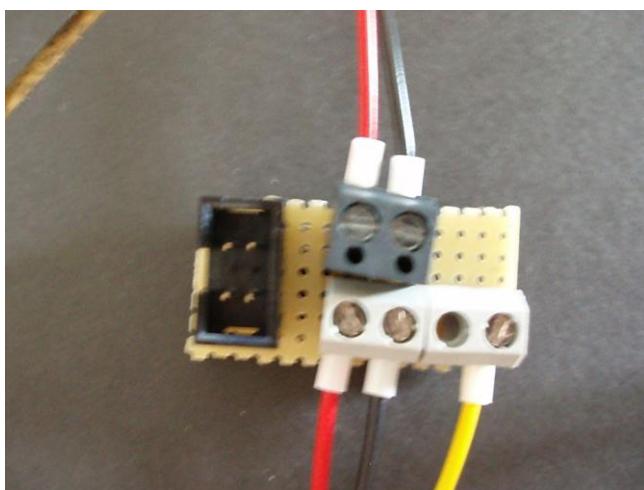


Abbildung 14: Verteilerplatine Bild 1

3.6.1.1 Materalkosten Verteiler

Klemmen	= 0,15 €	* 3	= 0,45 €
Pfostensteckersockel	= 0,20 €	* 1	= 0,20 €
Platine und Zubehör	= 0,20 €	* 1	= 0,20 €
<hr/>			
Gesamt:			= 0,95 €

3.6.2 Sender

Der Sender besteht aus einem Widerstand R1 und einer Infrarot Diode D1.

Die Diode wird mit einem Takt von 38 kHz betrieben und verträgt maximal einen Durchlassstrom von 60 mA. Da der Ausgang des Mikrocontroller maximal pro Pin 40 mA liefern kann, wird die Diode mit <40mA betrieben. Der Strom wird über den Vorwiderstand R1 eingestellt. Die Diode hat einen Spannungsfall von 1,2V, somit ergibt sich folgende Rechnung:

$$U = 5V, U_D = 1,2V, I = 40mA$$

$$U_R = U - U_D = 5V - 1,2V = 3,8V \quad (1)$$

$$R = \frac{U_R}{I} = \frac{3,8V}{40mA} = 90 \Omega \quad (2)$$

Da in der E12 Widerstandsreihe der Widerstand mit 90 Ohm nicht existiert, wird der nächstgrößere mit 100 Ohm gewählt. Einen kleineren kann man nicht nehmen, da dieser einen Strom >40mA zur Folge hätte.

$$I = \frac{U}{R} = \frac{3,8V}{100 \Omega} = 38mA \quad (3)$$

Durch die Diode fließt somit ein Strom von 38mA.

Im folgenden Verdrahtungsplan steht ebenfalls, dass ein Strom von 35mA fließt. Dieser bezieht sich auf die Diode, die beim Versuchsaufbau 1 verwendet wurde.

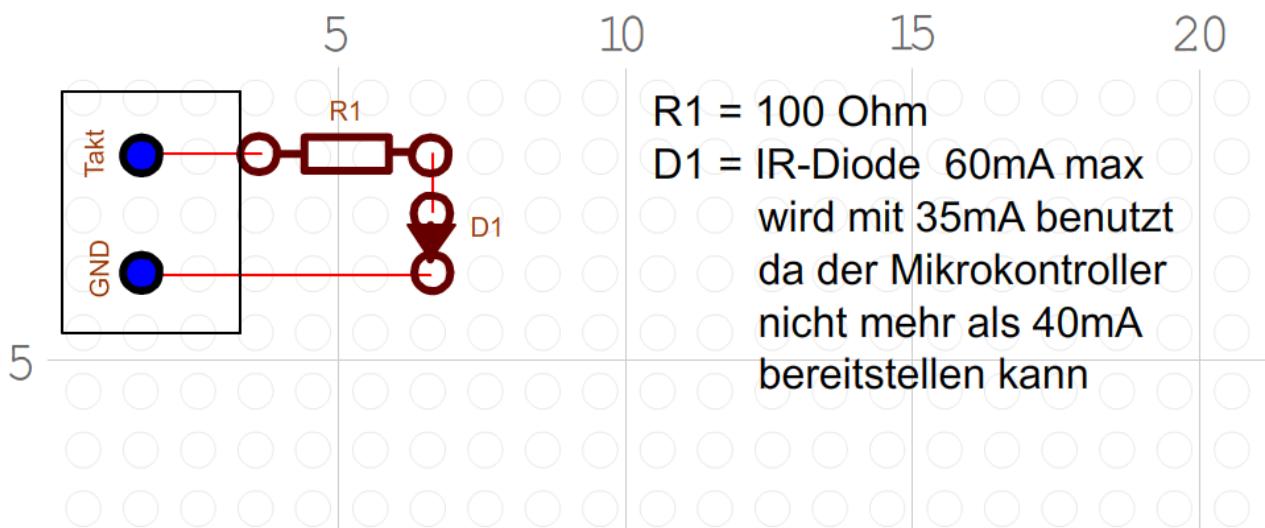


Abbildung 15: Verdrahtungsplan Sender

3.6.3 Materialkosten Sender

IR-Diode	= 0,03 €	* 1	= 0,04 €
Widerstand	= 0,07 €	* 1	= 0,07 €
Klemme	= 0,15 €	* 1	= 0,15 €
Platine und Zubehör	= 0,20 €	* 1	= 0,20 €
<hr/>			
	Gesamt:		= 0,46 €

3.6.3.1 Versuch 1

Beim ersten Versuch die Lichtschranke aufzubauen, wurde eine einfache Abschirmung gegen Streuung verwendet (siehe Bild 2). Es hat sich jedoch herausgestellt, dass die IR Diode zu stark gestreut hat und die Abschirmung nicht optimal war. Aus dem Grund wurde dieser Versuch verworfen.

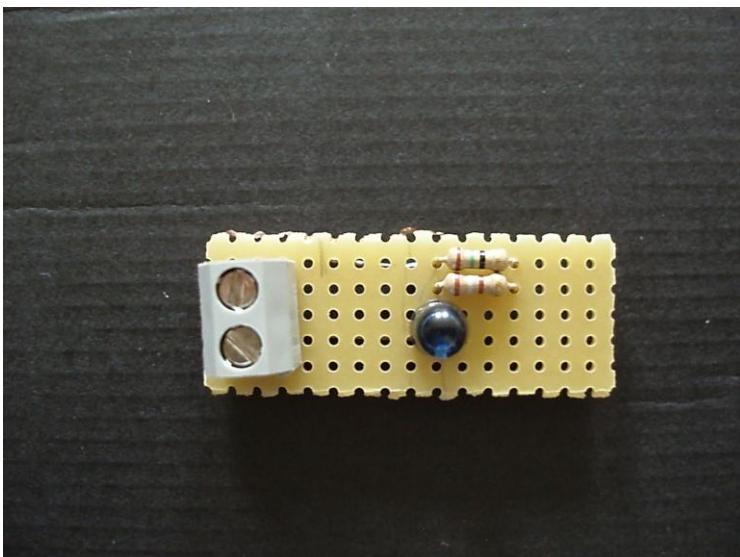


Abbildung 16: Senderplatine Versuch 1 Bild 1



Abbildung 17: Senderplatine Versuch 1 Bild 2

3.6.3.2 Versuch 2

Beim zweiten Versuch wurde eine andere IR Diode verwendet. Diese besitzt eine Linse zum Bündeln der Austrittsstrahlen. Außerdem hat diese Diode einen anderen Betriebsstrom, der fließt. Die Rechnung, die in Formel (3) verwendet wird, bezieht sich auf diese Diode.

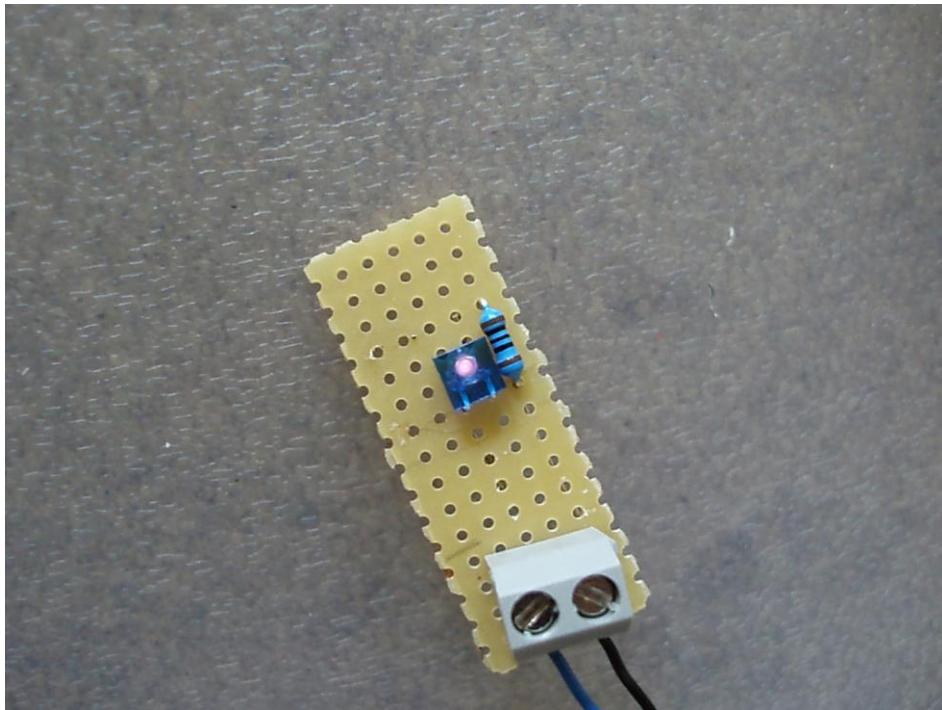


Abbildung 18: Senderplatine Versuch 2 Bild 1

Für die Abschirmung wurde ein Gehäuse aus Pappe gebaut, in welches die Platine eingelassen wird. Auf die Öffnung vorne können verschiedene Lochblenden gesetzt werden, sodass die Streuung weiter minimiert wird. Es hat sich zudem herausgestellt, dass je kleiner die Öffnung ist, umso schwerer die Einstellung der Lichtschranken wird. Dazu mehr im Software Teil unter Kalibrierung.

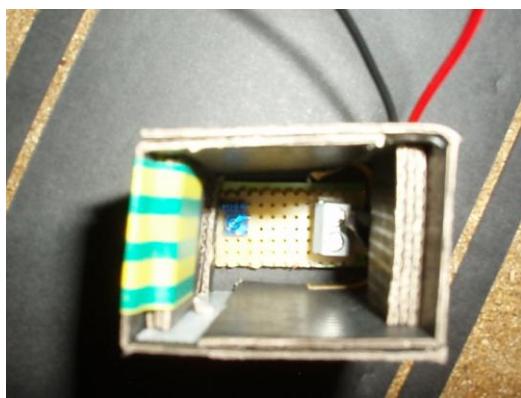


Abbildung 19: Senderplatine Versuch 2 Bild 2 (links)

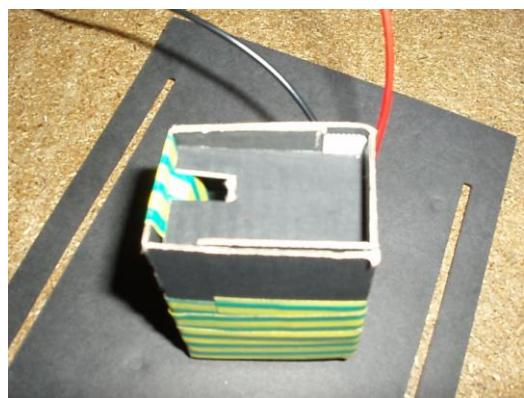


Abbildung 20: Senderplatine Versuch 2 Bild 3 (rechts)

3.6.4 Empfänger

Der Infrarot Empfänger wird durch einen TSOP 31238 realisiert. Dieser Baustein wird eigentlich beim Übertragen von Infrarotsignalen zwischen Fernsehfernbedienung und Fernseher eingesetzt. Er lässt sich aber auch gut für schnelle günstige Lichtschranken verwenden. Die Reaktionszeit liegt im Nanosekundenbereich.

Der IR-Empfänger braucht ein Signal, das mit 38 kHz getaktet ist. Ansonsten wird kein gültiger TTL Pegel ausgegeben.

Die Bauteile, die auf dem Verdrahtungsplan zu sehen sind, dienen einerseits zur Absicherung gegen Störung (C1, R2), wie es im Datenblatt beschrieben ist, anderseits zur Auffrischung des Ausgangs (R1, D1). Die Ausgangsspannung muss aufgefrischt werden, da auf der Hauptplatine ein Pull-down-Widerstand verbaut ist. Beim Testen der Schaltung ist aufgefallen, dass das Ausgangssignal S nicht 5V als High Pegel ausgibt, sondern etwas weniger. Der Ausgangsstrom ist so gering, dass über den Pull-down-Widerstand ein Spannungsfall auftritt. Da auf dem Ausgangssignal auch leichte Störung zu messen waren, wurde ab und zu der Interrupt am Mikrokontroller ausgelöst, ohne das ein Ereignis an der Lichtschranke aufgetreten war. Aus diesem Grund wird das Ausgangssignal über den Widerstand R1 und die Diode D1 auf einen höheren Pegel gezogen.

Für den Empfänger wurde eine Papphülle gebaut, die mit austauschbaren Blenden die Feineinstellung verbessert.

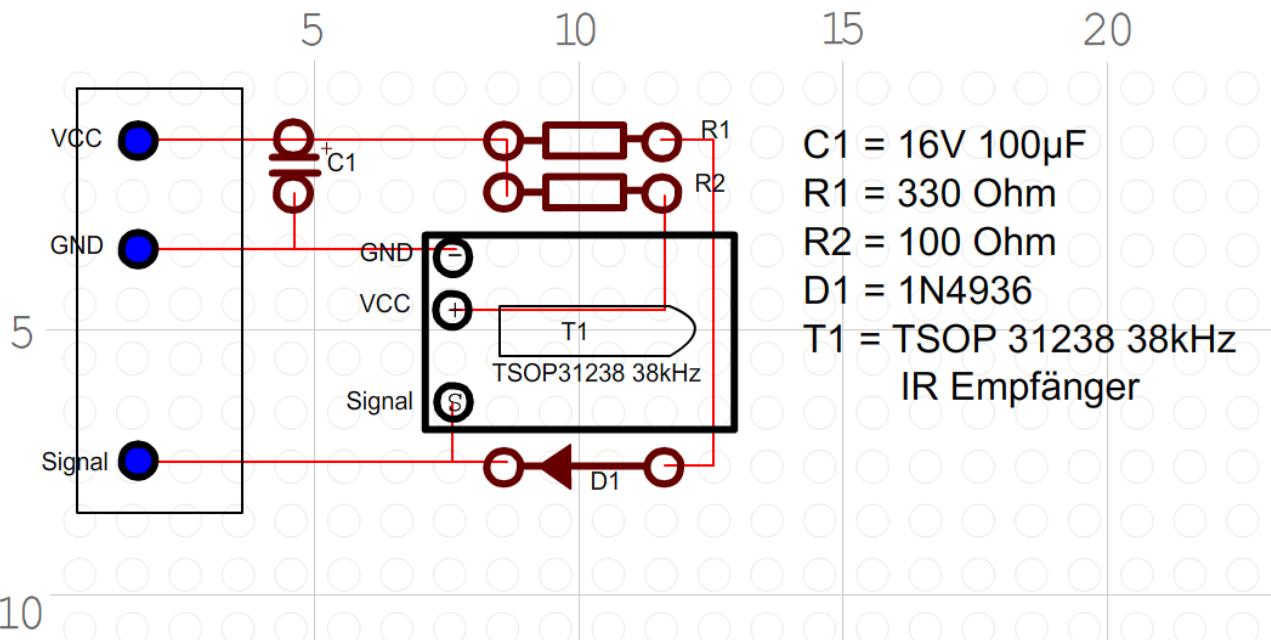


Abbildung 21: Verdrahtungsplan Empfänger

3.6.5 Materialkosten Empfänger

TSOP 31238	= 0,82 €	* 1	= 0,82 €
Kondensator	= 0,15 €	* 1	= 0,15 €
Diode	= 0,07 €	* 1	= 0,07 €
Widerstand	= 0,07 €	* 2	= 0,14 €
Klemmen	= 0,15 €	* 2	= 0,30 €
Platine und Zubehör	= 0,20 €	* 1	= 0,20 €
<hr/>			
	Gesamt:		= 1,68 €

3.6.5.1 Bilder des Empfängers

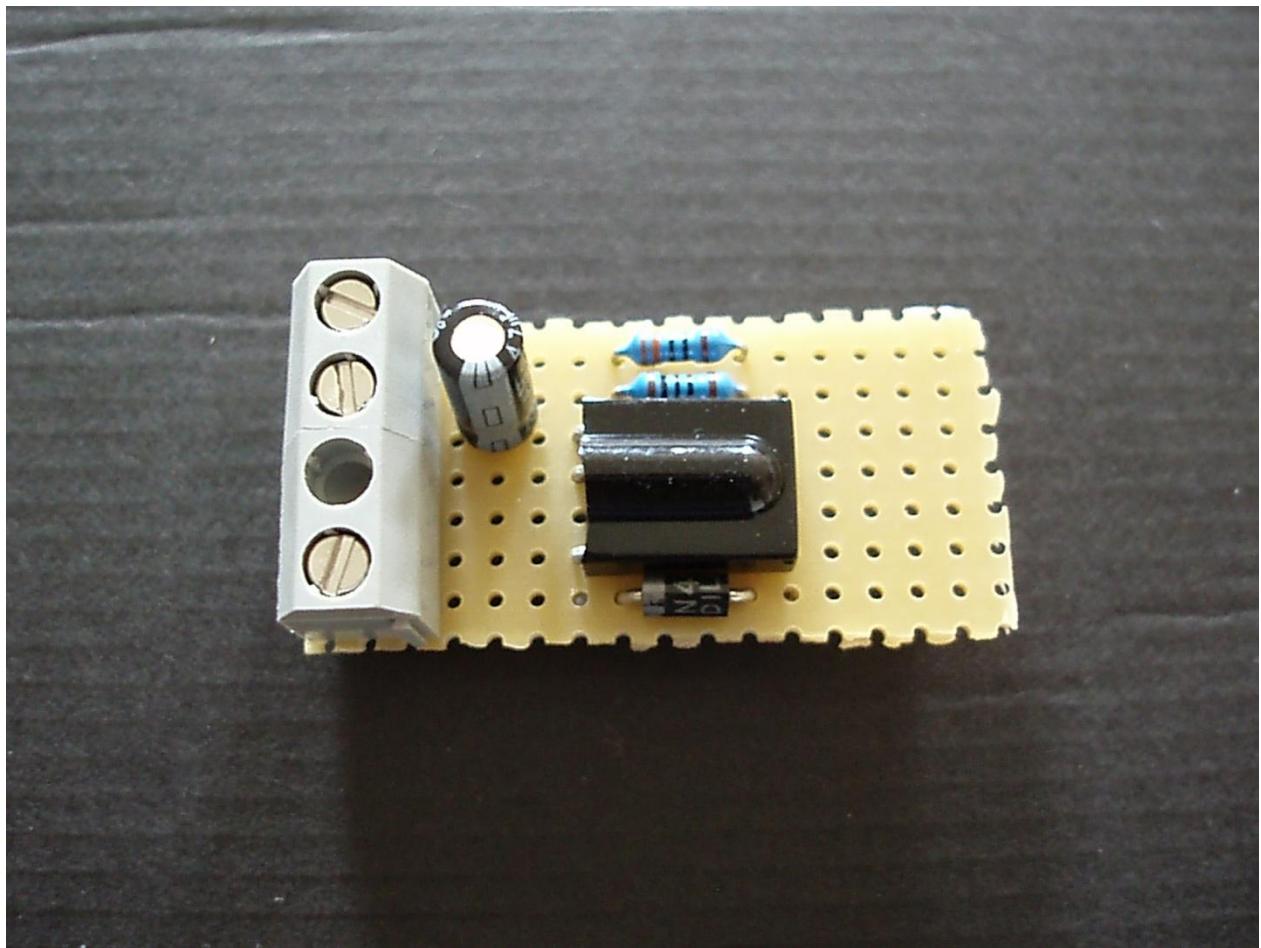


Abbildung 22: Empfängerplatine Bild 1

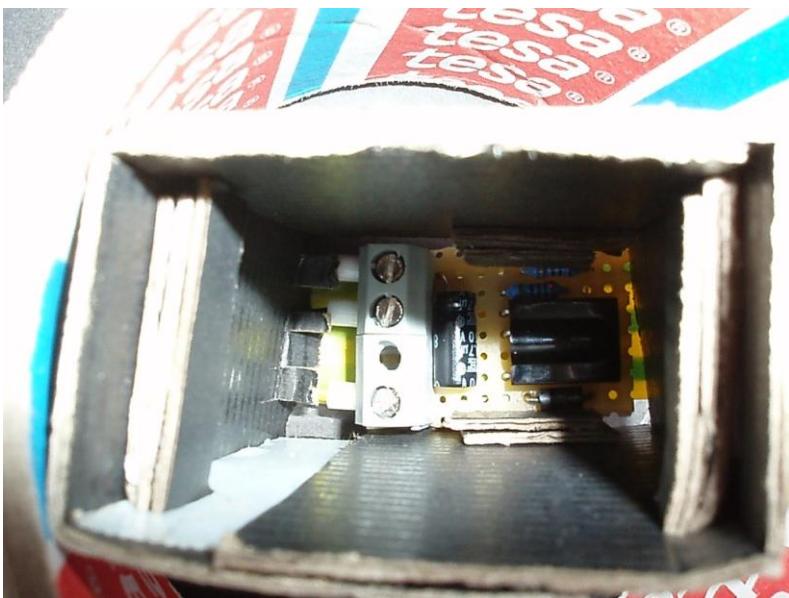


Abbildung 23: Empfängerplatine Bild 2

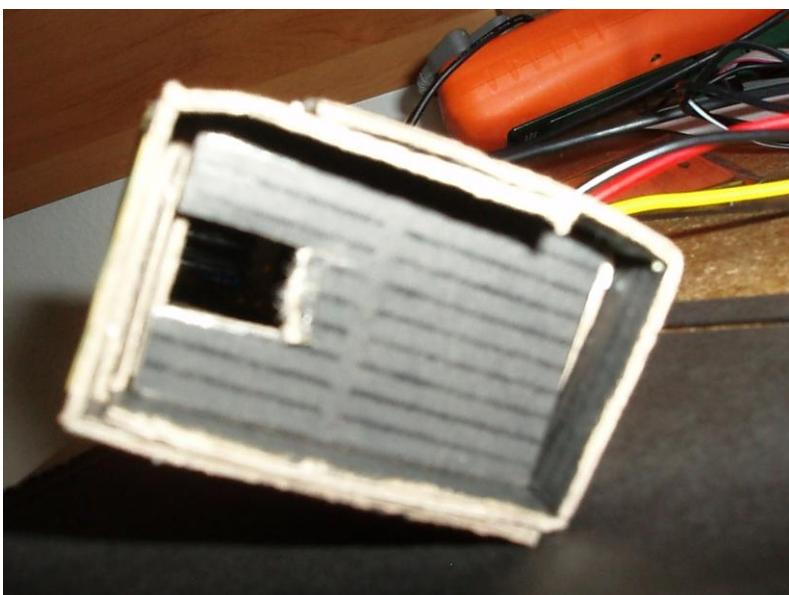


Abbildung 24: Empfängerplatine Bild 3

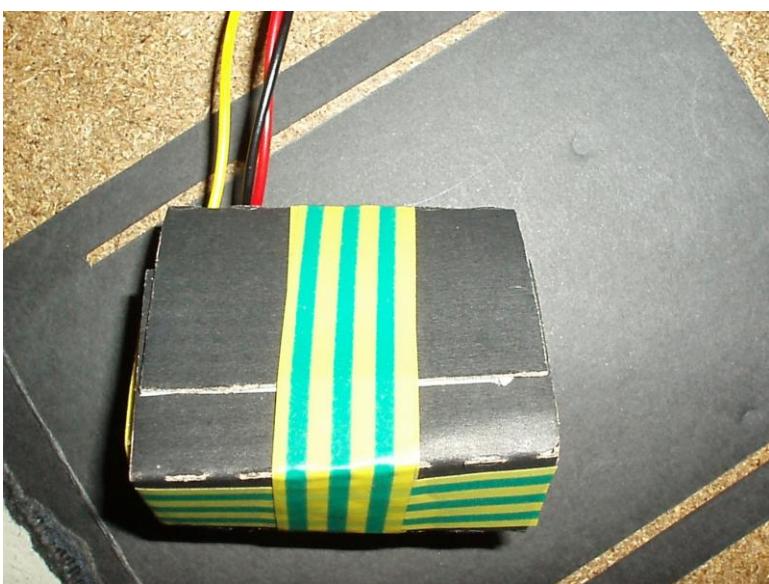


Abbildung 25: Empfängerplatine Bild 4

3.7 Spannungsversorgung (extern)

Als externe Spannungsversorgung wird ein Steckernetzteil verwendet. Dieses wird in der Kostenliste für die Hardware nicht aufgeführt, da es viele Möglichkeiten gibt, die Platine mit Spannung zu versorgen.



Abbildung 26: externe Stromversorgung

3.8 Materialkosten gesamte Hardware

Hauptplatine	= 19,37 €	* 1 = 19,37 €
LCD-Platine	= 20,83 €	* 1 = 20,83 €
Tastatur	= 1,60 €	* 1 = 1,60 €
Verteiler	= 0,95 €	* 2 = 1,90 €
Sender	= 0,46 €	* 2 = 0,92 €
Empfänger	= 1,68 €	* 2 = 3,36 €
Versandkosten ca.	= 10 €	* 1 = 10 €
Kabelverbindungen	= 5 €	* 1 = 5 €
<hr/>		
Gesamt 62,98 €		

4 Software im Detail

Die Software ist so geschrieben, dass sie an möglichst wenigen Stellen durch ein sleep (Wartezeit) unterbrochen wird. Dadurch ist es möglich, regelmäßig mit „polling“ die Tastatur abzufragen und Backgroundfunktionen auszuführen. Die Backgroundfunktionen und ihre Bedeutung sind unter Softwarestruktur näher erklärt.

Die Softwarestruktur wird ohne Quellcode erklärt, da dieser zu lang ist. Wenn das Dokument geöffnet ist, ist es möglich, Stück für Stück anhand der Struktur die Quellcode zu verstehen.

4.1 Softwarestruktur

Da die Arduino Softwareumgebung den Umgang mit mehreren Dateien sehr schwer macht, ist die komplette Software in einer Datei erstellt worden. Da diese sehr lang ist, wird unter „Softwarestruktur“ der grobe Aufbau erklärt.

4.1.1 Includes

Als Erstes kommen die „Includes“. Diese binden externe Klassen ein. Die Klassen liegen nicht mit im Projektverzeichnis, sondern sind in der Entwicklungsumgebung unter „./libraries“ zu finden. Alle Klassen, die nicht von der Arduino Umgebung bereitgestellt werden, befinden sich im Anhang unter ([anhang/klassen_geschwindigkeitsanzeige_kicker.rar](#)) und müssen zur Software in das oben genannte Verzeichnis hinzugefügt werden. Die Klassen „TimerOne“, „LiquidCrystal“ und „Menu“ wurden nicht von mir erstellt. Diese kann man im Arduino Playground finden. Der Playground ist eine Arduino Plattform auf der Klassen bereitgestellt werden (<http://www.arduino.cc/playground/>).

4.1.2 Definitionen

Alles, was statisch vorliegt, sollte definiert werden. In der Software sind viele Texte, die im Programm verwendet werden, an einer Stelle definiert. Somit ist es leichter, Veränderungen vorzunehmen oder aber die Software in die englische Sprache zu übersetzen. Zudem sind alle Pins, die vom Mikrokontroller verwendet werden, definiert.

Außerdem werden viele Parameter und Einstellungen für die Tastatur definiert.

4.1.3 Globale Variablen

Da es zeitlich nicht mehr möglich war, alle Funktionen in Klassen zu gliedern, gibt es viele globale Variablen. Bei den Variablen gibt es hauptsächlich den Datentyp „unsigned char“, der Werte in einem Bereich zwischen 0-255 aufnehmen kann. Variablen dieses Types benötigen 1 Byte im Speicher. Dieser Type wurde gewählt, da möglichst wenig Speicher verbraucht werden soll.

Alle globalen Variablen die die Zeiten beinhalten, haben den Datentyp „unsigned long“. Dieser ist 4 Byte groß und kann die Werte zwischen 0 – 4294967295 annehmen. Die Zeiten werden in Millisekunden gespeichert. Somit tritt der Speicherüberlauf erst nach 50 Tagen auf. Solange sollte die Geschwindigkeitsanzeige nicht am Stück laufen.

4.1.4 Globale Objekte

Es werden einige globale Objekte verwendet. Dazu zählen:

- | | |
|-----------------------------------|-----------|
| - LCD | 1 Objekt |
| - LCDMenu | 1 Objekt |
| - Nested Set Mode (Menu Struktur) | 9 Objekte |

Durch die Arduino Umgebung kommen weitere Objekte hinzu:

- EEPROM
- SERIAL

Diese belegen nur Speicher, wenn sie verwendet werden.

4.1.5 Prototypen

Prototypen sind Funktionen bei denen nur der Funktionskopf an einer Stelle oberhalb der anderen Funktionen gesetzt wird. Das Verwenden von Protoptypen macht erst Sinn, wenn Funktionen andere Funktionen benötigen, die unterhalb ihrer Definition stehen. Da das Programm von oben nach unten abläuft, kommt dies in größeren Programmen häufiger vor.

4.1.6 Nützliche Funktionen

Unter nützliche Funktionen steht eine Funktion, die einen float Wert in einen zwei int Werte umwandelt.

4.1.7 Funktionen zum Initialisieren der Lichtschranken

Hier werden die Funktionen aufgeführt, die die Lichtschranken wieder zurücksetzen, bzw. freigeben. Wenn der Interrupt der Lichtschranke auslöst, wird eine globale Variable gesetzt, die den Interrupt sperrt. Diese Sperre kann mit den Funktionen wieder rückgängig gemacht werden.

4.1.8 Background Funktionen

Backgroundfunktionen werden durch die Loop-Schleife immer wieder regelmäßig aufgerufen. In der Software dienen diese dazu, die Tastatureingabe abzufragen und die Funktionen, die im Menü laufen, kontinuierlich aufzurufen.

Die Backgroundfunktionen unterteilen sich in eine Funktion, die Menüfunktionen aufruft, „BACK_SelectMenuFunction()“ und eine weitere, die sich um die Verarbeitung der Tastatureingaben kümmert „BACK_button()“.

BACK_SelectMenuFunction() ist so ausgelegt, dass hier schnell Änderungen vorgenommen werden können, falls sich das Menü verändern sollte.

In BACK_button() werden die Tastatureingaben verarbeitet. Hier ist es möglich, verschiedene Modi auszuwerten, je nachdem wie man die Tastatur konfiguriert hat:

- 1 analoger Eingang mit Widerstandsmessungen
- 6 I/O Pins, für jeden Button einen eigenen.
- Steuerung über a,s,d,w,q,e über den seriellen Monitor

4.1.9 Kontrollfunktionen

Die Kontrollfunktionen dienen dazu, den Ablauf der Menü Funktionen zu kontrollieren. Hier sind folgende Optionen möglich:

- Menüfunktionen immer wieder aufrufen (Schleifen Modus)
- Menüfunktion einmal aufrufen, danach zurück zum Menü
- Menüfunktionen gezielt beenden
- Warteschleifen, die das Programm nicht anhalten
- Initialisierung der Buttons
- Initialisierung des Menüs

In der Software sind folgende Namen für diese Funktionen vergeben:

- CONTROL_set_func_active()
- CONTROL_set_func_end()
- CONTROL_refresh_timer_func()
- CONTROL_refresh_time()
- CONTROL_init_button()
- CONTROL_menu_init()

4.1.10 Menüfunktionen

Die Menüfunktionen werden durch die Backgroundfunktionen aufgerufen und durch die Kontrollfunktionen überwacht. Jeder Menüpunkt, der keine Untermenüpunkte besitzt, erhält eine Funktion. Für die Geschwindigkeitsanzeige wurden folgende Funktionen erstellt.

- FUNC_start_speed();
- FUNC_show_highscore();
- FUNC_show_information();
- FUNC_setting_object_length();
- FUNC_setting_distance();
- FUNC_setting_calibration();
- FUNC_setting_highscore_reset();
- FUNC_setting_system_reset();

4.1.11 Interruptfunktionen

Die zwei Interruptfunktionen sind in der Setup Funktion so initialisiert, dass sie bei ansteigender sowie abfallender Flanke (Mode: „changing“) zwei Funktionen aufrufen, die die Zeitdifferenz nach dem letzten Auslösen messen.

4.1.12 Setup

Die Setup-Funktion wird von der Arduino Umgebung benötigt. Sie wird beim Einschalten oder nach einem Reset des Mikrocontrollers einmal aufgerufen. In dieser sollten alle Hardwarekomponenten initialisiert werden.

4.1.13 Loop

Die Loop Schleife ist das gleiche wie eine „while(1)“ Schleife und wird die ganze Zeit durchgängig aufgerufen. Von dieser Schleife sollte das Hauptprogramm laufen.

4.2 Speicherabschätzung im SRAM

4.2.1 Globale Variablen

Variablen mit:

- „unsigned char“ 1 Byte
 $31 * 1 \text{ Byte} = 31 \text{ Byte}$
- „char“ 1 Byte pro Zeichen (maximal 18 Zeichen)
 $18 * 1 \text{ Byte} = 18 \text{ Byte}$
- „unsigned long“ 4 Byte
 $11 * 4 \text{ Byte} = 44 \text{ Byte}$

Gesamt: ca. 60 Byte

4.2.2 globale Objekte

Bei den globalen Objekten lässt sich der Speicherbereich nur schwer einschätzen, da die Klassen, die im Hintergrund liegen, analysiert werden müssen. Daher kann man hier nur eine grobe Übersicht machen:

- LCD Klasse:
 Beinhaltet hauptsächlich Integer Variablen
 ca. 50 Byte
- Menu Klasse:
 Beinhaltet das Nested Set Model
 ca. $9 * 20 \text{ Byte} + \text{interner Verbrauch} = 250 \text{ Byte}$
- LCD Menu Klasse:
 Arbeite mit dem Menu und LCD Objekten
 ca. 100 Byte
- Arduino Klassen die im Hintergrund laufen:
 ca. 100 Byte

Gesamt: ca. 500 Byte

4.2.3 Variablen in Funktionen

Da Variablen in Funktionen immer nur solange gültig sind, bis die Funktion beendet ist, wurde die Funktion mit dem größten Speicherbedarf als Platzhalter für alle anderen genommen.

Die größte ist: „FUNC_start_speed()“. Diese Funktion hat einen Speicherbedarf von:

- Funktionsname „char“ (max 18 Zeichen)
 $18 * 1 \text{ Byte} = 18 \text{ Byte}$
- Verschiedene Puffer Variablen in float
 $2 * 4 \text{ Byte} = 8 \text{ Byte}$
- Integervariablen
 $2 * 2 \text{ Byte} = 4 \text{ Byte}$

Gesamt: ca. 30 Byte

4.2.4 Gesamt

Da der SRAM des Mikrocontrollers 2000 Byte groß ist, ist eine grobe Speicherabschätzung interessant. Das Programm benötigt zusammen genommen ca. 590 Byte SRAM. Somit ist man da auf der sicheren Seite.

4.3 Speicherbedarf im Flash

Beim Kompilieren des Programms wird angezeigt, dass der Code 16042 Byte der 32000 Byte belegt. Der Bootloader und die vereinfachten Funktionen der Arduino Software belegen ca. 2000 Byte.

4.4 Menüführung

Die Menüführung in der Software ist einfach gehalten. In der ersten Ebene gibt es vier Auswahlpunkte. Die ersten drei können direkt ausgeführt werden. Hinter dem Vierten liegt die Menüebene zwei mit fünf weiteren Auswahlmöglichkeiten.

- Start
- Bestenliste
- Information
- Einstellungen
 - o Objekt Laenge
 - o Abstand L1<->L2
 - o Kalibrierung L1 L2
 - o Reset Highscore
 - o Werkseinstellungen

Im folgenden Bild sieht man sehr gut, wie das Menü aussieht.



Abbildung 27: Ebene 1 im Menü

Der Cursor steht immer vor der Reihe, die man auswählt. Wenn mehr Auswahlmöglichkeiten als Zeilen im Display vorhanden sind, wird ein kleiner Scrollbalken angezeigt. Diesen sieht man im zweiten Bild unten rechts in der Ecke (der Pfeil nach unten). Dieser zeigt an, dass es noch weitere Auswahlmöglichkeiten unterhalb der aktuellen Anzeige gibt.

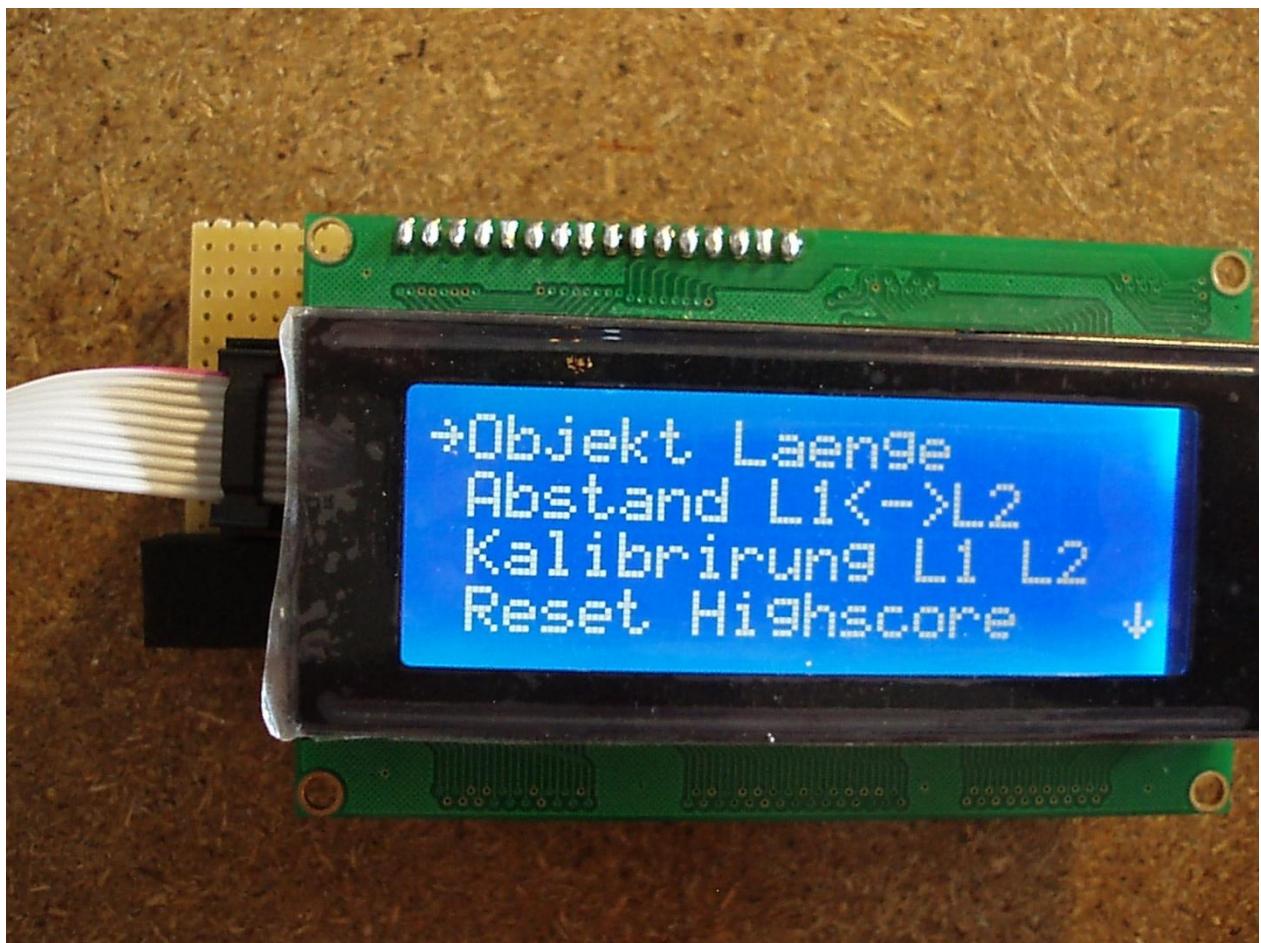


Abbildung 28: Menüeben 2 oben

In dem oben gezeigten Bild sieht man die zweite Menüebene, in der sich 5 Auswahlmöglichkeiten befinden.

Das nächste Bild bezieht sich auch noch auf den Punkt Menüführung. Man sieht, dass der letzte Menüpunkt in Ebene zwei ausgewählt wurde. Der Cursor steht unten und der Scrollbalken zeigt an, dass es noch einen Wert oberhalb der aktuellen Anzeige gibt.

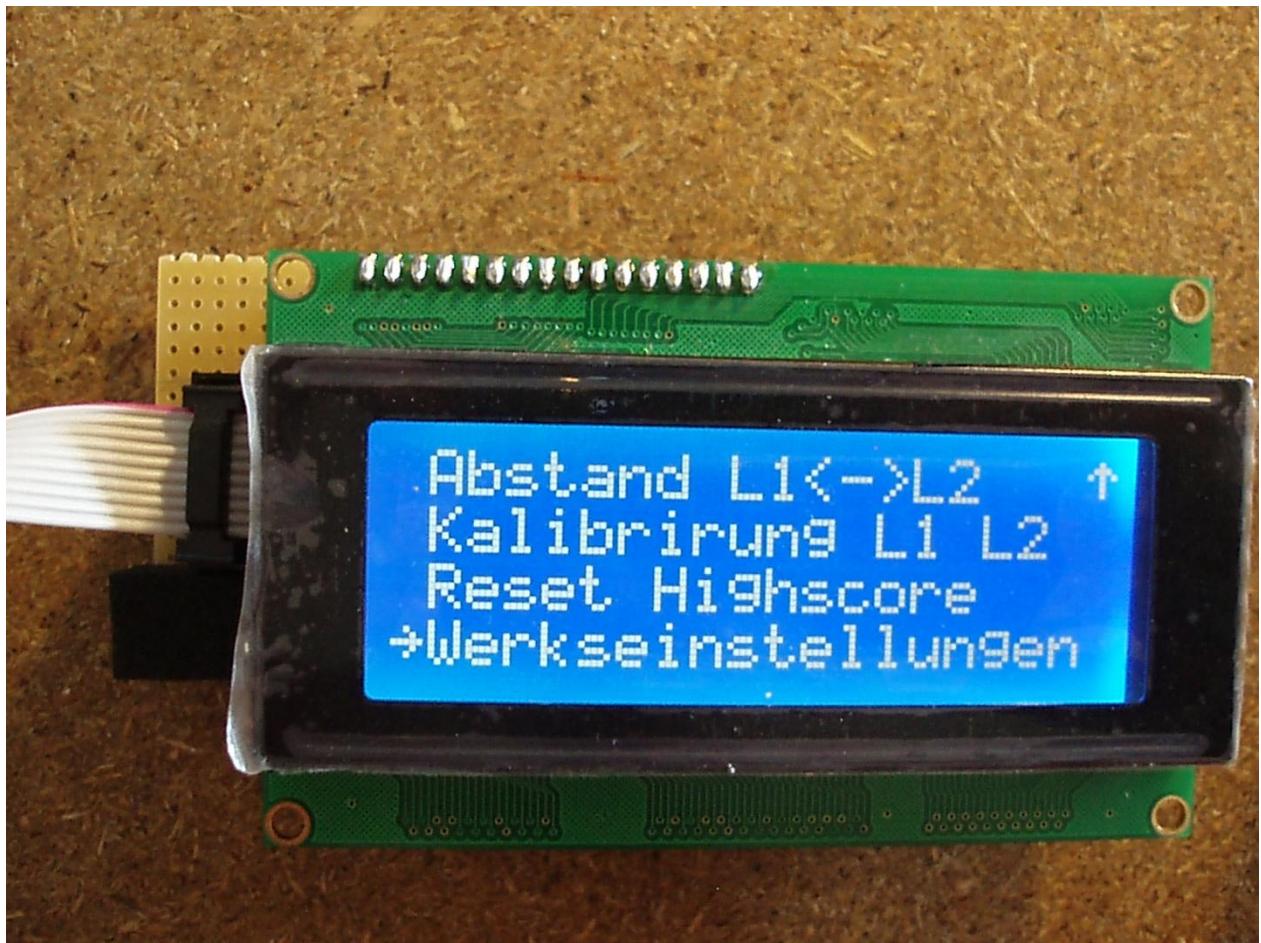


Abbildung 29: Menüeben 2 unten

Das Menü bedient man auf der Tastatur mit zwei der Pfeiltasten. Mit „Pfeile nach oben“ wandert man im Menü nach oben, mit „Pfeil nach unten“ geht es nach unten. Wenn man am Ende ankommt, kann man nicht weiter scrollen. Einen Menüpunkt öffnet man mit der Taste „Enter“ und zurück geht es mit „Zurück“ auf der Tastatur.

Die „Zurück-Taste“ funktioniert in allen Funktionen. Mit „Zurück“ kommt man immer zum Ursprung zurück.

4.5 Werte verändern

In der Aufgabenstellung war es gefordert, Werte verändern zu können. In den nächsten Bildern ist erklärt, wie man das im Detail macht.

Wenn in der zweiten Menüebene der Punkt „Objekt Laenge“ ausgewählt wird und man mit „Enter“ bestätigt, wird das folgende Bild angezeigt.



Abbildung 30: Werte verändern Bild 1

Da bei der deutschen Beschreibung von „Links, Rechts, Hoch, Runter“ der Buchstabe „R“ zweimal vorkommt, habe ich mich hier für die englische Bezeichnung der Pfeiltasten entschieden.

L = left, R = right, U = up, D = down.

Mit L und R kann man den Wert verstellen und mit U und D lässt sich die Zeile verstellen. Diese Text, der die Bedienung erklärt, kommt immer einmalig vor den Einstellungen. Mit der Taste „Enter“ geht es weiter zu den Einstellungen. Mit „Zurück“ kommt man immer, egal wo man ist, zurück zum Menü.

Im nächsten Bild sieht man, wie es möglich ist, die Objekt Länge zu verstellen. Am Anfang, wenn die Software sich im Auslieferungszustand befindet oder in die Werkseinstellungen zurück gesetzt wurde, stehen alle Werte auf „0“. Der Cursor ist auf den ersten Wert gerichtet. Mit „U“ und „D“ lässt sich der Cursor verschieben. Mit „L“ und „R“ kann man den Wert verstetlen.



Abbildung 31: Objektlänge verstetlen Bild 1

Die Anzeige der Zeile wird immer angepasst. Erreicht man das Minimum oder Maximum werden die Pfeile „<“ und „>“ passend ein-/ausgeblendet. Dieses Verhalten sieht man gut, wenn man das Bild oben und das nächste Bild vergleicht.



Abbildung 32: Objektlänge verstellen Bild 2

Im Vergleich zum letzten Bild ist hier nun der „<“ vor dem Wert hinzugekommen.

Im folgenden Bild sieht man die Veränderungen, wenn mehrere Werte verstellt wurden und der Cursor nicht mehr auf der ersten Position steht. Des Weiteren kann man erkennen, dass bei Zahlen, die mehr als eine Stelle besitzen, der Cursor hinter der Zahl an die passende Stelle verschoben wird.

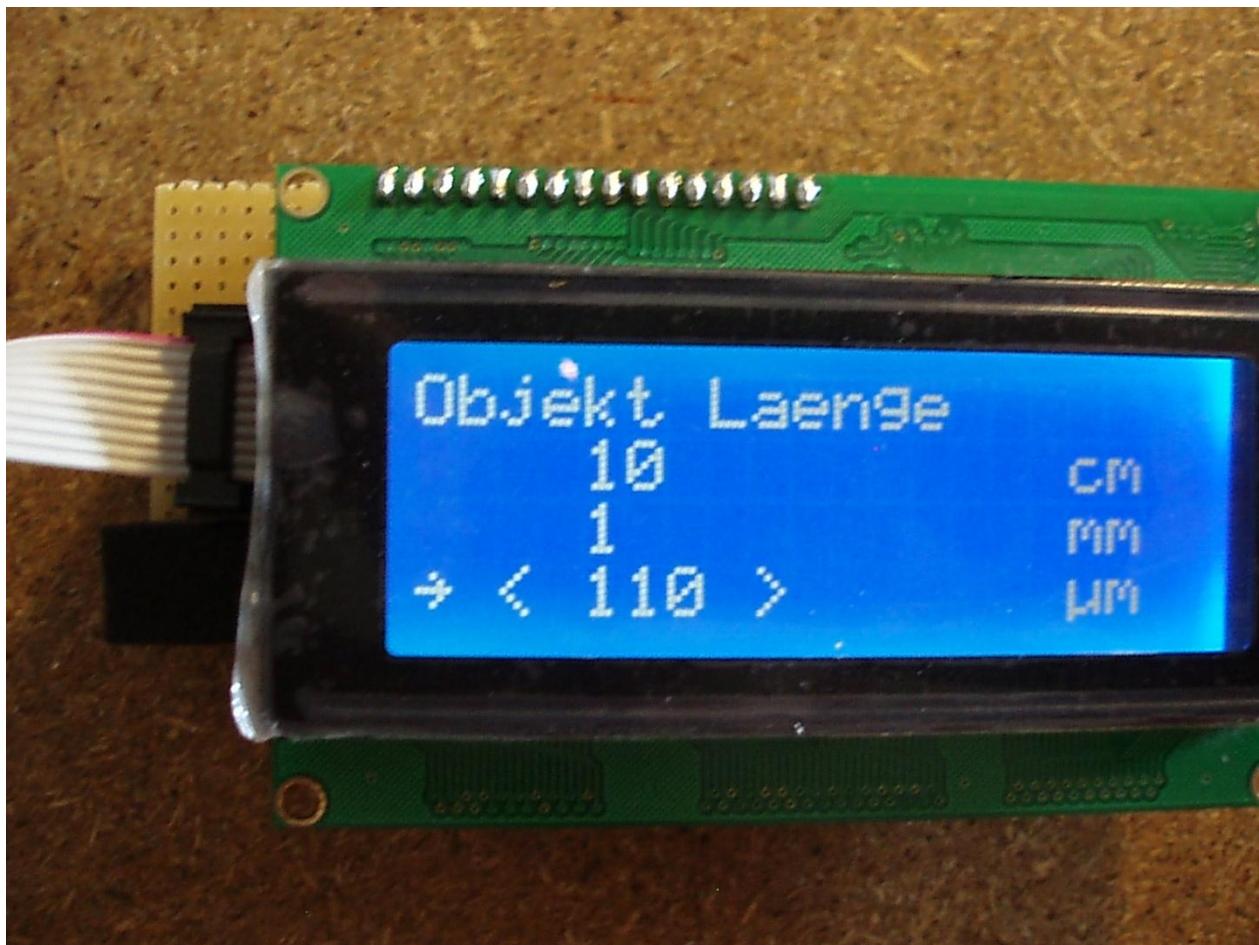


Abbildung 33: Objektlänge verstellen Bild 3

Die jetzige Einstellung die man auf dem LCD sehen kann, stellt einen Wert von 10,1110 cm ein. Die Wertebereiche für die verschiedenen Einstellungen cm, mm, μm sind so definiert:

- cm 0 – 99 Schrittweite 1
- mm 0 – 9 Schrittweite 1
- μm 0 – 990 Schrittweite 10

Durch diese Bereiche lässt sich ein Maximaler Wert von 99 cm, 9mm 990 μm einstellen (99,999 cm). Die Bereiche von 0 – 99 wurden so gewählt, da diese Werte leicht im Speicher (EEPROM) abgelegt werden können und eine einheitliche Einstellung der Werte möglich ist.

Mit der Taste „Enter“ werden die Werte gespeichert. Mit der Taste „Zurück“ kommt man zurück zum Menü ohne zu speichern.

Die gleichen Einstellungen kann man auch unter „Abstand L1<->L2“ vornehmen. Nur der Title der Überschrift lautet dann anders und ein anderer Speicherbereich wird beschrieben. Die Werte sind, da sie im EEPROM gespeichert werden, nach einem Neustart weiterhin vorhanden.

4.6 Lichtschranken

Die Auswertung der Lichtschranken in der Software geschieht in zwei unterschiedlichen Bereichen. Einmal wird die Messung für die Geschwindigkeit genutzt, zum anderen für die Einstellung und das Testen der Lichtschranken.

Sobald der Mikrokontroller eingeschaltet wird, läuft im Hintergrund eine Messung, die überprüft, ob keine, eine, oder zwei Lichtschranken angeschlossen sind. Diese Messung läuft die ganze Zeit über, solange keine Geschwindigkeitsmessung gestartet wird. Diese Messung wird durch das ständige Stoppen des Timers erreicht. Der Timer wird alle 200 ms ausgeschaltet und 200 ms später wieder eingeschaltet.

Der Timer erzeugt eine Frequenz von 38,42 kHz. Dieses Signal wird auf die Infrarot Diode gegeben. Der Empfänger braucht 38 kHz, um einen TTL Pegel auszugeben. Der Pegel geht hardwaremäßig direkt auf einen Interrupt, der mit dem Modus „changing“ konfiguriert ist (auslösen bei fallender oder steigender Flanke). Durch das ständige ausschalten des Timers wird so in einem Abstand von 200 ms ein Rechtecksignal gegeben. Wenn an einem Interrupt kein Signal ankommt, ist dieser Ausgang nicht belegt und keine Lichtschranke ist angeschlossen.

4.6.1 Geschwindigkeitsmessung

Wenn die Geschwindigkeitsmessung gestartet wird, unterbricht eine Variable (Run_Speed = 1) die Überprüfung der Lichtschranken. Der Timer wird nun durchgehen aktiviert, da eine Unterbrechung nur durch ein Ereignis der Lichtschranke ausgelöst werden soll.

Als nächstes wird bei der Geschwindigkeitsmessung überprüft, ob eine Lichtschranke angeschlossen ist. Wenn dies nicht der Fall ist, wird für 3 Sekunden eine Meldung auf dem LCD angezeigt. Diese 3 Sekunde Wartezeit lassen sich mit keinem Button abbrechen. Anschließend wird wieder das Hauptmenü angezeigt.

Nun wird erklärt, wie die Lichtschranken messen, um später Fragen gleich zu beantworten.

Wenn nur eine Lichtschranke angeschlossen ist, wird die Zeit gemessen wie lange die Lichtschranke unterbrochen ist. Hierfür werden die Werte der „Objekt Länge“ benötigt.

Wenn zwei Lichtschranken angeschlossen sind, gibt es drei verschiedene Modi, wie man zu einer Geschwindigkeitsmessung kommt:

- Messung nur Anhand der Objekt Länge. Das liefert zwei Geschwindigkeiten die dann gemittelt werden.
- Messung indem die Objekt Länge auf 0 gestellt wird und der Abstand zwischen den Lichtschranken eingestellt sein muss. Diese Messung liefert 1 Ergebnis.
- Messung mit Objekt Länge und Lichtschranken abstand. Diese Messung liefert 3 Messwerte die dann auch gemittelt werden.

Alle 3 Modi sind über die Objekt Länge / Abstand zwischen L1 und L2 einstellbar. Sie sind in der Software implementiert und liefern gute Messergebnisse. Mehr dazu im Kapitel Messung.

Wenn zwei Lichtschranken angeschlossen sind, wird überprüft, ob Werte für die Objekt Länge oder für den Abstand zwischen L1 und L2 eingestellt sind. Ansonsten kommt eine Meldung, die drei Sekunden anhält und danach eine Rückkehr zum Hauptmenü erzwingt. Um diese Meldung zu vermeiden, müssen erst Werte eingestellt werden.

Wenn alle Werte gültig sind und die Lichtschranken gefunden wurden, sieht das Display bei Messungen mit einer Lichtschranke wie folgt aus:



Abbildung 34: Messung mit einer Lichtschranke

Aktuell wird gewartet, dass ein Ereignis am Interrupt auftritt und die Geschwindigkeit angezeigt werden kann.

In der zweiten Zeile sieht man wie viele Lichtschranken zur Messung verwendet wurden und in welchem Modi die Messung durchgeführt wird. Als letztes sieht man unten den Highscore. Das ist der größte Wert, der jemals erreicht wurde.

Die Geschwindigkeit wurde auf einen Wert von 99,99 km/h begrenzt. Es können also Werte zwischen 0,00 und 99,99 km/h angezeigt werden. Zur Genauigkeit der Messung wird später noch mehr erklärt.

4.6.2 Kalibrierung

Der weitere Teil, der mit den Lichtschranke zu tun hat, ist die Kalibrierung. Am Anfang war es sehr schwer, die Infrarot Lichtschranken genau einzustellen, daher wurde eine Funktion geschrieben, die dies erleichtert.

Infrarotlicht ist für das menschliche Auge nicht sichtbar. Es lässt sich nur über Umwege sichtbar machen. Zum Beispiel mit Hilfe einer Digitalkamera oder Handykamera.

Da die Lichtschranken aber sehr genau eingestellt werden müssen, ist es mit Infrarotlicht ziemlich schwer.

Die Funktion zeigt an, ob die Lichtstrecke zwischen den Lichtschranken in Ordnung ist. Im nächsten Bild sieht man wie es aussehen kann, wenn keine Lichtschranke gefunden wurde, oder aber die Lichtstrecke zwischen den Lichtschranken unterbrochen ist.

Der Rechtschreibfehler, der auf den nächsten Bilder im Titel „Kalibrierung“ zusehen ist, wurde mittlerweile aus der Software behoben. Aus Zeitgründen wurden die Bilder nicht neu erstellt.



Abbildung 35: Kalibrierung, nichts gefunden

Lichtschranken, die nicht gefunden werden, werden mit „Lx nicht gefunden“ markiert. Die Lichtschranke, die erkannt wird, wird mit „Lx ok“ dargestellt.

Im folgenden Bild wurde, im Gegensatz zu dem oberen Bild, eine Lichtschranke erkannt und mit „ok“ gekennzeichnet.



Abbildung 36: Kalibrierung L1 ok

Im nächsten Teil der Kalibrierung kann die Genauigkeit der Lichtschranken eingestellt werden. Aus der ersten Messung der Kalibrierung geht hervor, ob eine Lichtschranke gefunden wurde. Ob die Lichtstrecke gut ist oder nur auf Reflexion basiert, ist nicht zu erkennen. Das Bild auf der nächsten Seite zeigt Teil 2 der Kalibrierung.

Beim zweiten Teil der Kalibrierung muss man die Lichtschranke mit Objekten unterbrechen, die sich erst langsam, dann schneller und am ende sehr schnell bewegen. Hinter der „0“ verbirgt sich ein Zähler, der anzeigt, dass ein Impuls am Interrupt eingegangen ist. Dieser Wert muss sich bei jeder Unterbrechung erhöhen. Falls Sprünge auftreten, ist das nicht schlimm, dann wurde ein Objekt mehrmals zu schnell erkannt. Bei hohen Geschwindigkeiten kommt das selten vor. Erst wenn sehr schnelle Objekte erkannt werden, ist die Lichtschrank richtig eingestellt. Verändert sich der Wert nicht, sollte man die Kalibrierung neu starten und den ersten Test wiederholen bis dauerhaft ein „ok“ angezeigt wird. Wenn der Wert beim ersten Test schwankt, ist die Lichtstrecke nicht gut.



Abbildung 37: Teil 2 Kalibrierung

Auf diesem Bild sieht man, dass sich der Wert des Zählers verändert hat. Der Wert des Zählers kann nicht überlaufen, da die Variable dahinter vom Datentyp „long“ ist.

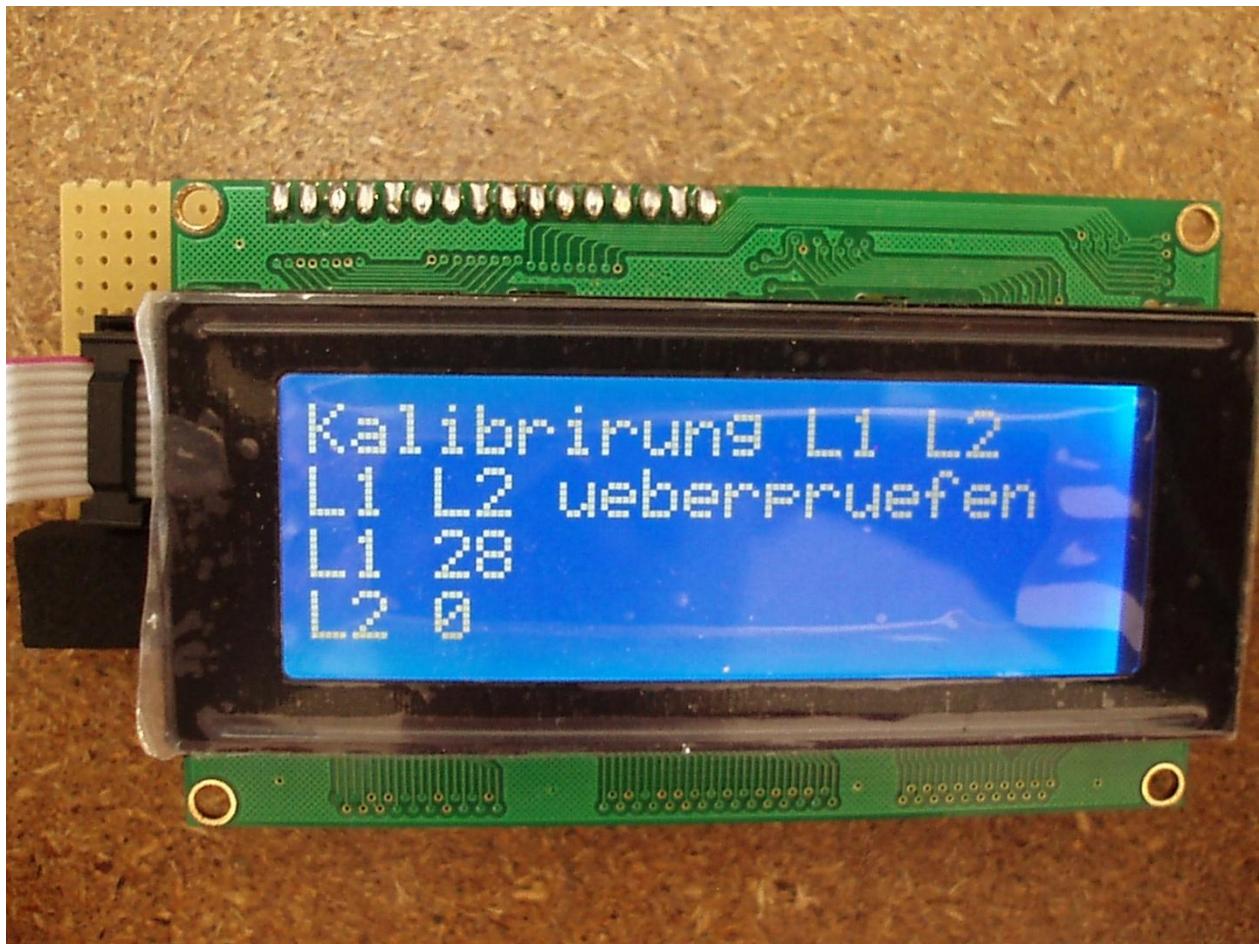


Abbildung 38: Teil 2 der Kalibrierung mit Zähler

4.7 Bestenliste

Der letzte Teil der Software bezieht sich auf die Bestenliste. Hier werden die drei besten Zeiten gespeichert. Kommt eine neue Zeit hinzu, werden die Werte passend nach unten geschoben.



Abbildung 39: Bestenliste

4.8 Software im Anhang

Die Software befindet sich im Anhang. Durch die Arduino Umgebung hat die Datei die Endung *.pde. Die Arduino Umgebungs Software kann man kostenlos herunterladen.

<http://arduino.cc/en/Main/Software>

5 Messungen

5.1 Messung der Geschwindigkeit mit nur einer Lichtschranke

In einem ersten Testversuch (Idealfall) soll folgende Messwerte aufgenommen werden:

Berechnung der Zeitdifferenz zwischen zwei Ereignissen, wobei die Zeit in der die Ereignisse ausgelöst werden in folgenden Abschnitten untersucht wird:

1 Hz bis 9 Hz	pro Schritt 1 Hz	(10 mal gemessen)
10 Hz bis 90 Hz	pro Schritt 10 Hz	(100 mal gemessen)
100 Hz bis 900 Hz	pro Schritt 100 Hz	(1000 mal gemessen)
1000 Hz bis 9000 Hz	pro Schritt 1000 Hz	(10000 mal gemessen)

Das Ergebnis wird in Prozent angegeben.

- Ermitteln der minimalen und maximalen Abweichung vom gemessenen Wert (Dauerhafter Fehler).
- Ermitteln der minimal und maximal möglichen Geschwindigkeit aus den Ergebnissen der beiden vorangegangenen Punkte. Die Geschwindigkeit bezieht sich auf eine Ballgröße von 34mm. Diese Größe entspricht der Norm beim Tischfußball.

Mit diesen Messwerten soll ermittelt werden, ob es sich lohnt, Korrekturfunktionen einzusetzen und wie diese aussehen könnten.

Zum Auslösen der Ereignisse am Interrupt wurde ein Funktionsgenerator angeschlossen (Rechteckbetrieb). An diesem wurden die oben aufgelisteten Frequenzen eingestellt.

Im Mikrokontroller wurde zum Test folgender Programmcode verwendet. Im Code wurde zu jedem Frequenzbereich der Wert, wie oft die Messung wiederholt werden sollte, angepasst.

5.1.1 Seite 1 des Test Programms

```
/* Variablen definieren die für die Zeitmessung gebraucht werden */
unsigned long Light_1_Time          = 0;
byte Light_1_State                  = 0;

/* Variablen definieren die für den Test gebraucht werden. Diese werden später im Programm nicht mehr verwendet */
byte test_run_cnt                  = 0;
unsigned long test_run_value = 0;
unsigned long Light_1_min           = 4000000;
unsigned long Light_1_max           = 0;

/* Die Setup Funktion beschreibt die Funktion in der alle Einstellungen vorgenommen werden sollen */
void setup()
{
    /* Serielle Übertragung zur Fehler Ausgabe initialisieren */
    Serial.begin(9600);

    /* Interrupt Definieren 0 => Interrupt 0, Light_1_getTime => Interrupt Funktion, CHANGE => Interrupt Modus */
    attachInterrupt(0, Light_1_getTime, CHANGE);
}

/* Interrupt Funktion */
void Light_1_getTime()
{
    /* Wenn die Zeit gemessen wurde, wir der Interrupt über die Variable deaktiviert, global bleibt er aktiv */
    if(Light_1_State == 0) {
        if(Light_1_Time == 0) {
            /* Zeitmessung starten */
            Light_1_Time = micros();
        } else {
            /* Zeitmessung beenden */
            Light_1_Time = micros()-Light_1_Time;
            Light_1_State = 1;
        }
    }
}

/* Werte von Light_1 zurücksetzen */
void Light_1_Init()
{
    Light_1_Time = 0;
    Light_1_State = 0;
}
```

5.1.2 Seite 2 des Test Programms

```
/* Loop ist die Funktion in der das eigentliche Programm abläuft */
void loop()
{
    /* Nur wenn eine Zeit gemessen wurde, soll diese Zeit zur Statistik hinzugefügt werden */
    if(Light_1_State != 0) {
        /* Test Counter inkrementieren */

        test_run_cnt++;

        /* Multiplikation mit 2, da die Periodendauer wichtig ist und im Moment nur die reine "AN" Zeit gemessen wird */
        Light_1_Time = Light_1_Time*2;

        /* Ergebnis aufaddieren */

        test_run_value += Light_1_Time;

        /* Minimal Wert ermitteln */

        if(Light_1_Time < Light_1_min) {
            Light_1_min = Light_1_Time;
        }

        /* Maximal Wert ermitteln */

        if(Light_1_Time > Light_1_max) {
            Light_1_max = Light_1_Time;
        }

        /* Aktuell gemessene Zeit zurücksetzen */

        Light_1_Init();
    }

    /* Wenn der Testlauf beendet ist (in diesem Fall nach 10 Durchläufen) werden die ermittelten Werte ausgegeben und 2 Sekunden
    gewartet bevor der neue Testlauf beginnt. */

    if(test_run_cnt >= 10) {
        /* Ausgabe der Zeitdifferenz zwischen zwei Ereignissen am Interrupt */

        Serial.println("Zeit in Mikrosekunden");
        Serial.println(test_run_value/test_run_cnt);

        /* Maximale Abweichung */

        Serial.print("Maximale Differenz: ");
        Serial.println(Light_1_max - test_run_value/test_run_cnt);

        /* Minimale Abweichung */

        Serial.print("Minimale Differenz: ");
        Serial.println(test_run_value/test_run_cnt - Light_1_min);

        Serial.println();

        /* Test Variablen zurück setzen */

        test_run_cnt = 0;
        test_run_value = 0;
        Light_1_min = 4000000;
        Light_1_max = 0;

        /* nach 2 Sekunden Wartezeit beginnt der Test von neuem */

        delay(2000);
    }
}
```

5.2 Praktische Messung im Kicker

Bei der praktischen Messung im Kicker wurden auf der Kickerfläche Klebestreifen angebraucht, die die Reflexion des Lichtes über die Bodenplatte verhindern. Die schwarzen Markierungen sind nur als Hilfestellung gedacht, damit die Lichtschranken genau gerade zu platzieren sind.

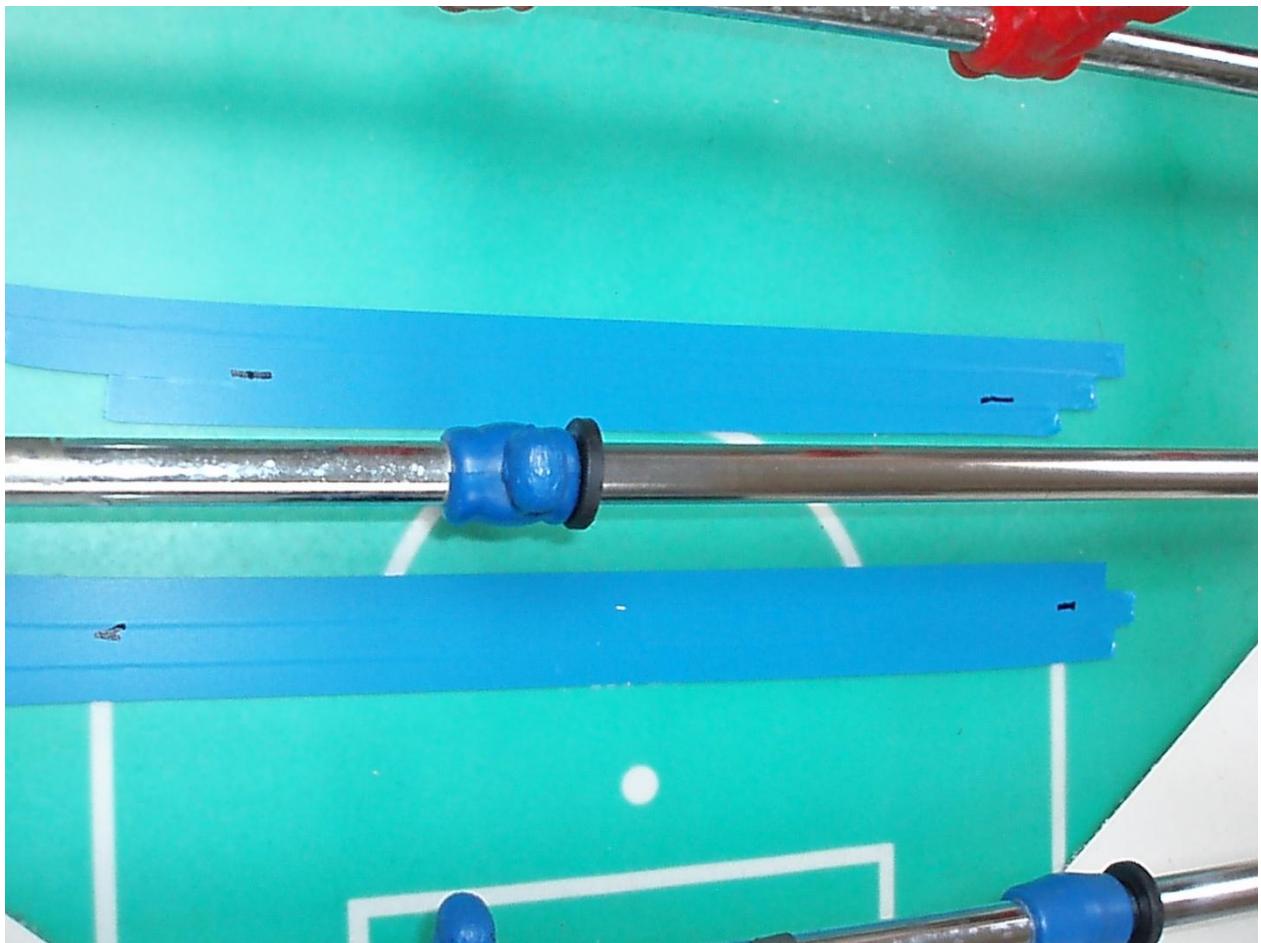


Abbildung 40: praktische Messung im Kicker Makierungen

In den folgenden Bildern sieht man die Lichtschranken flach auf dem Kickerboden liegen. Damit diese sich nicht drehen oder verschieben, wurden die Lichtschranken mit Gewichten versehen. Die Gewichte bestehen aus Metallplatten, Schraubenboxen oder schweren Behältern.

5.2.1 Versuchsaufbau mit einer Lichtschranke

Die praktische Messung mit einer Lichtschranke sah wie folgt aus. Die Lichtstrecke wurde möglichst gerade anhand vorhandener Linien im Kicker aufgebaut.

Im folgenden Bild sieht man den Sender.



Abbildung 41: Sender im Kicker 1 Lichtschranke

In diesem Bild sieht man den Empfänger, der durch eine schwere Kiste auf seiner Position festgehalten wird.

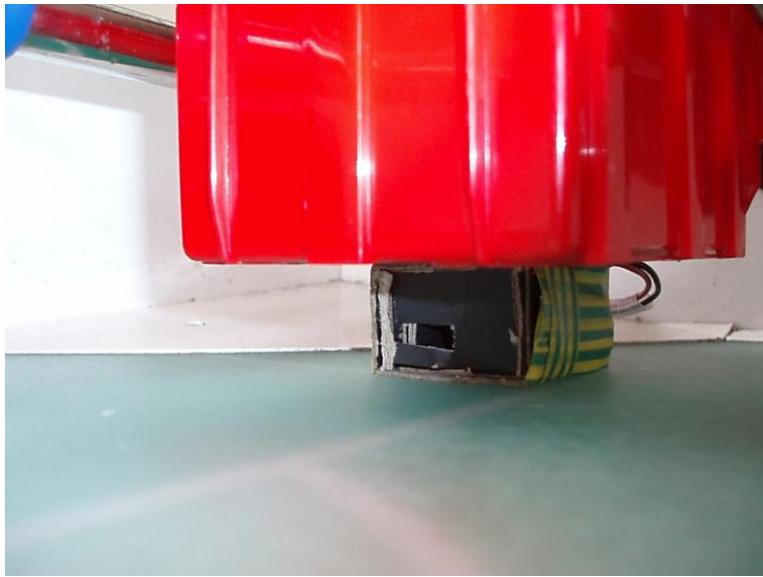


Abbildung 42: Empfänger im Kicker 1 Lichtschranke

In diesem Bild ist der Aufbau mit nur einer Lichtschranke abgebildet.



Abbildung 43: Aufbau des Kickers mit nur einer Lichtschranke

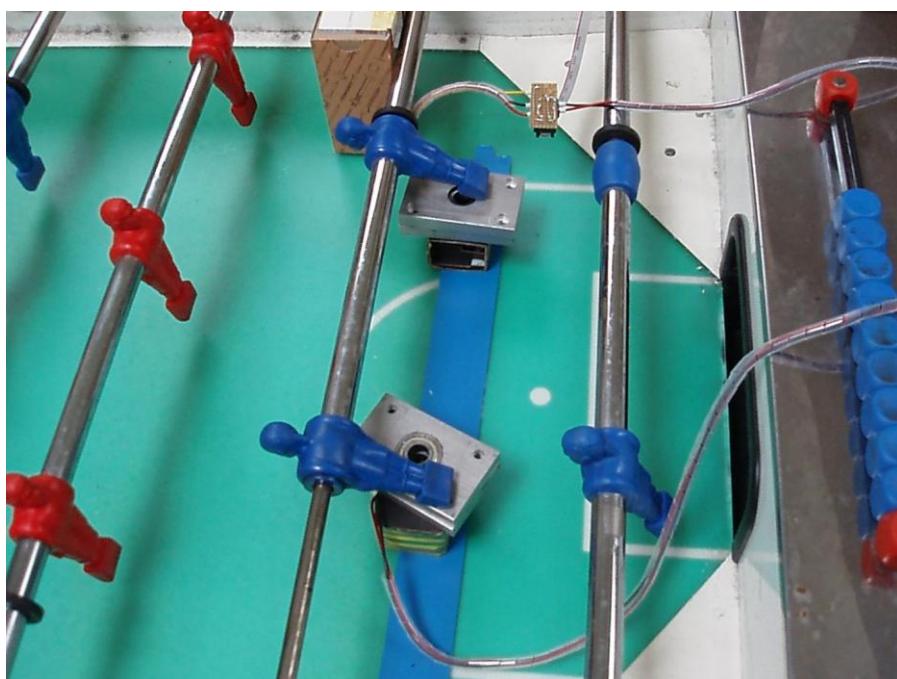


Abbildung 44: Draufsicht im Kicker 1 Lichtschranke

5.2.2 Versuchsaufbau mit zwei Lichtschranken

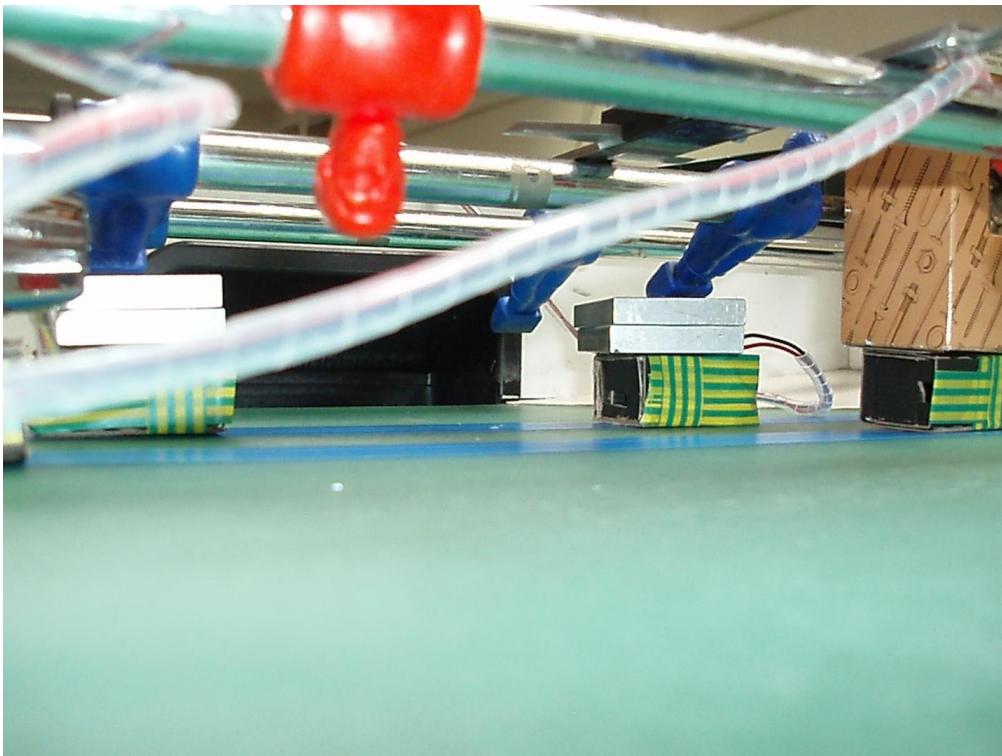


Abbildung 45: Zweilichtschranke Blickrichtung im Kicker

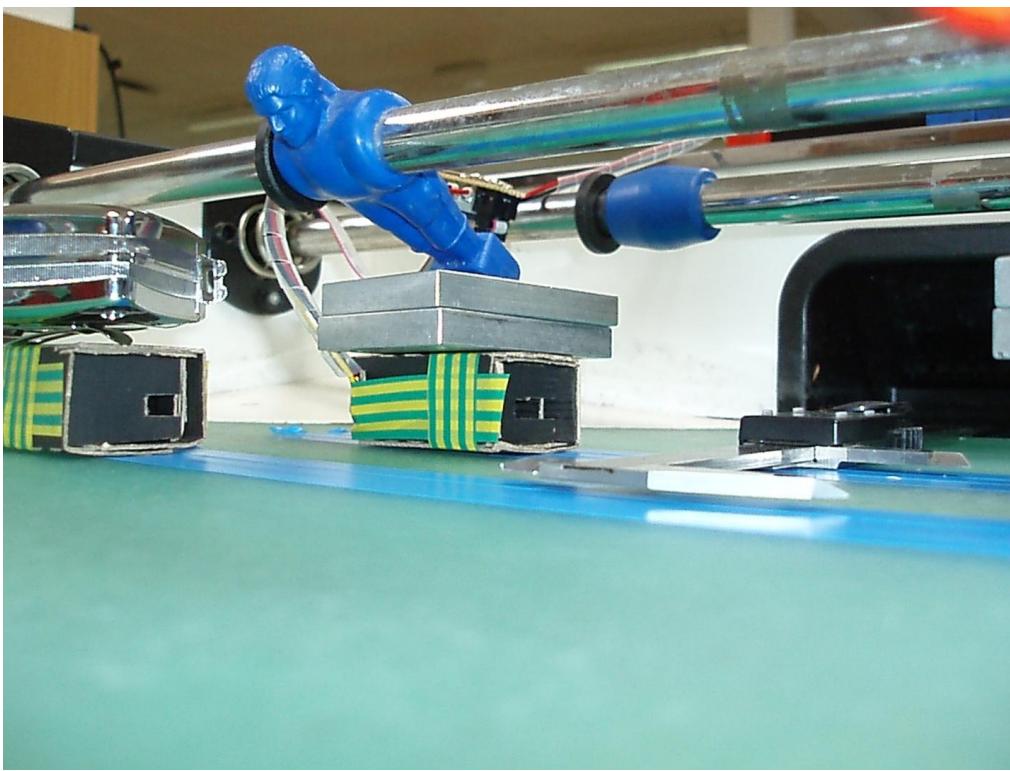


Abbildung 46: Zweilichtschranken Aufbau im Kicker

5.2.3 Hilfsmittel

Für die Messungen im Kicker um den Abstand zwischen den einzelnen Lichtschranken zu ermitteln, wurde ein digitaler Messschieber verwendet.



Abbildung 47: Digitaler Messschieber

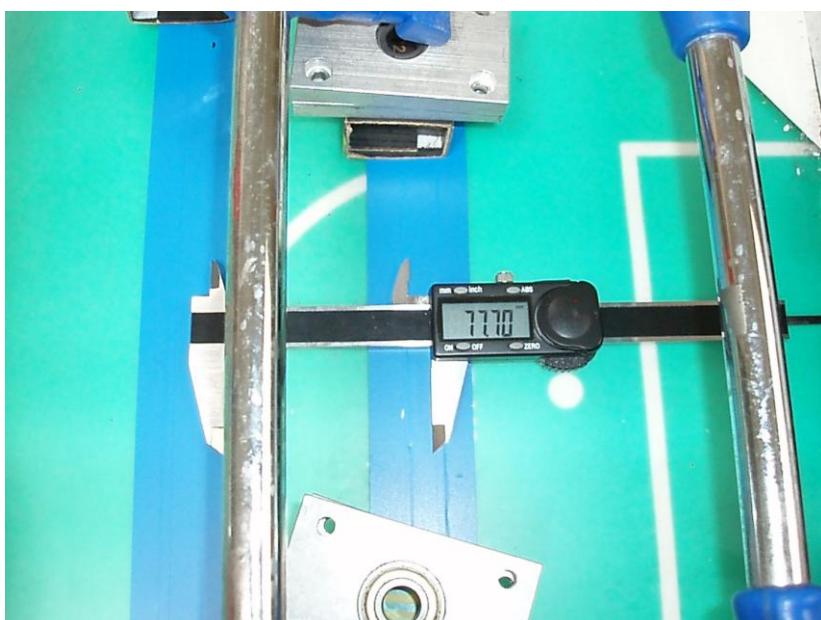


Abbildung 48: Zweilichtschranken Abstandmessung

5.2.4 Durchführung der Messungen

Der Test bezieht sich auf die Messung mit einer und zwei Lichtschranken.

Mit dem Messschieber wurde für den Abstand zwischen den Lichtschranken eine Wert von 77,7 mm ermittelt. Der Ball hat eine Größe von 33,34 mm. Diese Werte wurden abwechselnd für die verschiedenen Modi eingestellt.

Der Ball wurde mit der Reihe, die im Mittelfeld steht (fünf Spieler), durch die Lichtschranken geschossen. Dabei wurde darauf geachtet, dass die Schussrichtung exakt gerade Richtung Tor verläuft.

Zu jedem Modus, der eingestellt werden kann, wurden mehrere Schüsse abgegeben.

6 Ergebnisse

6.1 Messungen mit dem Frequenzgenerator

6.1.1 Ergebnisse und Verbesserungen

Bei der Auswertung des Tests wird Bezug zu den drei Punkten, die in der Tabelle stehen, genommen. Jeder Punkt hat in der unten aufgeführten Tabelle seine eigene dicke Spalte (durch schwarze Balken getrennt). Die oben aufgeführten Frequenzbereiche sind in der Tabelle durch blaue Spalten in vier kleinere Bereiche unterteilt.

Ergebnis in Tabellen Form: **Punkt 1**

Punkt 2

Punkt 3

Frequenz [Hz]	Zeit [μs]	gemessene Zeit im Kontroller [μs]	Abweichung [%]	Maximale Abweichun g nach Unten [μs]	Maximale Abweichun g nach Oben [μs]	Maximaler Fehler (plus/minus) [%]	Speed [m/s]	Speed [km/h]	Abweichung in Bezug auf den maximalen Fehler [km/h]
(10 Werte gemittelt)									
1	1000000,0	1022525	-2,25	4	4	2,25	0,02	0,06	0,00
2	500000,0	505248	-1,05	8	16	1,05	0,03	0,12	0,00
3	333333,3	335518	-0,66	7	9	0,66	0,05	0,18	0,00
4	250000,0	251144	-0,46	7	9	0,46	0,07	0,24	0,00
5	200000,0	200864	-0,43	8	8	0,44	0,08	0,30	0,00
6	166666,7	167106	-0,26	8	8	0,27	0,10	0,37	0,00
7	142857,1	143153	-0,21	7	1	0,22	0,12	0,43	0,00
8	125000,0	125204	-0,16	12	4	0,18	0,14	0,49	0,00
9	111111,1	111259	-0,13	4	12	0,15	0,15	0,55	0,00
(100 Werte gemittelt)									
10	100000,0	100107	-0,11	5	11	0,12	0,17	0,61	0,00
20	50000,0	50053	-0,11	10	14	0,14	0,34	1,22	0,00
30	33333,3	33343	-0,03	9	15	0,08	0,51	1,84	0,00
40	25000,0	25011	-0,04	4	4	0,11	0,68	2,45	0,00
50	20000,0	20002	-0,01	13	11	0,09	0,85	3,06	0,00
60	16666,7	16671	-0,03	17	15	0,12	1,02	3,67	0,00
70	14285,7	14287	-0,01	9	15	0,12	1,19	4,28	0,01
80	12500,0	12502	-0,02	11	13	0,15	1,36	4,90	0,01
90	11111,1	11112	-0,01	8	8	0,16	1,53	5,51	0,01
(1000 Werte gemittelt)									
100	10000,0	10001	-0,01	14	10	0,18	1,70	6,12	0,01
200	5000,0	4999	0,02	9	7	0,35	3,40	12,24	0,04
300	3333,3	3333	0,01	11	5	0,51	5,10	18,36	0,09
400	2500,0	2500	0,00	12	4	0,66	6,80	24,48	0,16
500	2000,0	2000	0,00	9	15	0,83	8,50	30,60	0,25
600	1666,7	1666	0,04	14	10	1,03	10,20	36,73	0,38
700	1428,6	1428	0,04	12	12	1,20	11,90	42,86	0,51
800	1250,0	1249	0,08	14	10	1,40	13,61	49,00	0,69
900	1111,1	1111	0,01	9	15	1,50	15,30	55,09	0,82
(10000 Werte gemittelt)									
1000	1000,0	1000	0,00	8	16	1,65	17,00	61,20	1,01
2000	500,0	500	0,00	12	12	3,30	34,00	122,40	4,04
3000	333,3	333	0,10	13	11	5,05	51,05	183,78	9,28
4000	250,0	249	0,40	10	14	7,00	68,27	245,78	17,20
5000	200,0	200	0,00	8	8	8,25	85,00	306,00	25,25
6000	166,7	167	-0,20	14	10	10,10	101,80	366,47	37,01
7000	142,9	143	-0,10	14	10	11,65	118,88	427,97	49,86
8000	125,0	125	0,00	13	11	13,20	136,00	489,60	64,63
9000	111,1	111	0,10	7	9	14,95	153,15	551,35	82,43

Tabelle 1: Excel Tabelle, Messung mit Frequenzgenerator

6.1.1.1 Zu Punkt 1:

Feststellung:

Beim Berechnen der Zeitdifferenz mit der prozentualen Abweichung fällt auf, dass bei niedrigen Frequenzen die Abweichung stark ansteigt, bei Höheren spielt dies kaum noch eine Rolle. Diese Abweichung lässt sich regelmäßig feststellen und sie schwankt kaum.

Verbesserung:

Diese Werte könnte man durch Korrekturfaktoren verbessern. Je nach Abhängigkeit der Geschwindigkeit die in einem Frequenzbereich zwischen 1Hz – 20Hz auftritt.

6.1.1.2 Zu Punkt 2:

Feststellung: Der Maximalfehler berechnet sich wie folgt:

$$\frac{50}{(\text{errechnete Zeit}) \cdot (\text{MAX}(\text{der minimalen Abweichung}) + \text{MAX}(\text{der maximalen Abweichung}))} + \text{Betrag(Fehler aus Punkt 1)}$$

Die minimale und maximale Abweichung treten bei allen Messwerten gleichermaßen auf. Dadurch lässt sich erkennen, dass bei großen Frequenzen (kleinen Messzeiten) der Fehler stark ansteigt. In der Tabelle ist die Abweichung deutlich zu sehen. Im vierten Frequenzbereich 1000Hz bis 9000Hz überschreitet der Wert die +-5 Prozent. Danach steigt die Fehlerentwicklung deutlich an.

Verbesserung:

In diesem Fall würde nur ein schneller Mikrocontroller helfen, dadurch wäre die Auflösung auch großen Frequenzbereichen genauer.

6.1.1.3 Zu Punkt 3:

Feststellung:

Wenn man die Korrektur unter Punkt 1 durchführt, lassen sich beliebig kleine Frequenzen messen. Bzw. die minimale sinnvolle Geschwindigkeit liegt bei 1km/h. Alles was drunter liegt, ist nicht mehr wichtig, könnte aber ausgewertet werden.

Bei der maximalen Geschwindigkeit spielt die unter Punkt 2 ermittelte Fehlerwahrscheinlichkeit von +- X Prozent eine große Rolle. Ab +- 5 Prozent wird der Fehler zu groß, um noch gute Werte liefern zu können.

Bei ca. +-3 Prozent Abweichung erreicht man ca. 120km/h. Diese Geschwindigkeit wird vermutlich niemand schießen können. Die Bälle, die von guten Spielern geschossen werden, liegen meistens bei einer Geschwindigkeit von 50km/h. (Ermittlung mit einer Hochgeschwindigkeitskamera an der HS-Osnabrück).

6.2 Praktische Messungen im Kicker Tisch

Da bei der praktischen Messung kein Ball mit der gleichen Geschwindigkeit geschossen werden kann und keine Einrichtung bereit stand, um eine gleichmäßige Geschwindigkeit zu simulieren, wurde nur ausprobiert, ob die Messungen in etwa an die Werte herankommen, die mit der Hochgeschwindigkeitskamera erzielt wurde.

Der schnellste Schuss am Kicker ergab eine Geschwindigkeit von 52,38 km/h. Dieser Wert wurde mit einer Lichtschranke gemessen. Die Werte mit zwei Lichtschranken liegen leicht unterhalb dieses Wertes.

Mit der Hochgeschwindigkeitskamera wurde eine maximale Geschwindigkeit von 55 km/h festgestellt. Daraus lässt sich schließen, dass die Messung bis auf minimale Abweichungen die reale Geschwindigkeit anzeigt.

Wie sich die praktischen Messungen verhalten, wenn der Ball schräg durch die Lichtschranken fliegt oder aber durch andere Dinge beeinflusst wird, lässt sich mit der groben Infrarotlichtschranke nicht ermitteln.

7 Zeitaufwand

Für das komplette Projekt wurde die vorgegeben Zeit von 14 Tagen minimal überschritten, wenn für jeden Tag 8 Stunden angesetzt sind.

Überlegungen zur Hardware und Software:	1 Tag
Erstellung der Hardware:	2 Tage
Erstellung der Software:	2 Tage
Genaue Einstellung der Lichtschranken:	5 Tage
Probleme in der Hardware und Software beheben:	2 Tage
Schreiben der Hausarbeit:	4 Tage

Zeitaufwand: 16 Tage

8 Interpretation

Die Messergebnisse zeigen, dass es mit wenig Hardwareaufwand möglich ist, eine Geschwindigkeitsanzeige für den Kicker zu erstellen. Ob die angezeigten Werte nun exakt den physikalischen Geschwindigkeiten entsprechen, konnte nicht ermittelt werden. Hierfür werden eine Referenzmessung sowie die Möglichkeit, einen Ball für jede Messung mit identischer Geschwindigkeit zu schießen, benötigt.

Durch die vier unterschiedlichen Modi, die Geschwindigkeit zu bestimmen, kann man für jeden Einsatzzweck die richtige Wahl treffen.

lässt sich anhand der praktischen Erfahrung beurteilen, dass die Lichtschranken mit Infrarotlicht äußerst schwer einzustellen sind. Würde man hier auf eine Messung mittels Laser umsteigen, wären vermutlich bessere Messergebnisse möglich. Erstens streut der Strahl des Lasers nicht so sehr und zweitens ist die Justierung leichter, da das Licht gesehen werden kann.

9 Quellen

- www.arduino.cc Software, Reference, Forum, Playground

10 Anhang

- Software (./anhang/software_geschwindigkeitsanzeige_kicker.rar)
- Arduino Einführung (./anhang/arduino_einführung.pdf)
- Klassen (./anhang/klassen_geschwindigkeitsanzeige_kicker.rar)
- Verdrahtungspläne (./anhang/verdrahtungspläne.rar)

ⁱ Quelle LCD Bild: http://www.watercooling.de/catalog/popup_tm.php?pID=36314&imagenr=1