

Rundenzähler und Ampel für eine Carrerabahn

Thies Hansen, Stefan Büll, G-13

Bearbeitungszeitraum:

03.02.2009 - 30.06.2009

Inhaltsverzeichnis

1 Information und Planung.....	1
1.1 Lastenheft.....	1
1.2 Pflichtenheft.....	1
1.3 Lösungsansatz.....	2
1.3.1 Beschreibung des Lösungsansatzes im Überblick.....	2
1.3.2 Technologieschema.....	2
1.3.3 Blockschaltplan.....	3
1.4 Arbeitsplanung.....	3
1.5 Kostenplanung.....	3
1.5.1 Entwicklungskosten.....	4
1.5.2 Fertigungskosten.....	4
2 Durchführung.....	4
2.1 Hardware.....	4
2.1.1 Schaltungsbeschreibung.....	4
2.1.2 Schaltplan.....	5
2.1.3 Bestückungsplan.....	6
2.1.4 Bauteilliste.....	7
2.1.5 Platinenlayout.....	8
2.2 Software.....	9
2.2.1 Entwicklungsumgebung / Programmiersprache.....	9
2.2.2 Programmbeschreibung.....	9
2.2.3 Programmablaufplan / Struktogramm.....	10
2.2.4 Quelltext.....	11
3 Kontrolle und Dokumentation.....	20
3.1 Inbetriebnahme, Mess- und Prüfprotokolle.....	20
3.2 Bedienungsanleitung.....	20
3.3 Fotos.....	20
3.3.1 Gesamtaufbau.....	20
3.4 Kritik der eigenen Projektplanung und - durchführung.....	21
3.4.1 Zeitplanung.....	21
3.4.2 Kostenplanung, tatsächliche Kosten.....	21
4 Quellenverzeichnis.....	22
4.1 Datenblätter.....	22
4.2 Beispielprojekte.....	22
4.3 Fachliteratur.....	22

Information und Planung

1.1 Lastenheft

Zielbestimmung:

Es soll ein Rundenzähler für eine 2 spurige Carreraautorennbahn mit einer Ampel realisiert werden.

Produkteinsatz:

Carrerarennbahn,

Der Rundenzähler soll speziell auf die Carrerabahn anwendbar sein und eine Ampel wie auf einer richtigen Rennstrecke über der Rennbahn sein.

Produktfunktion:

Der Zähler muss extern über einen Taster Resetbar sein. Die Startphase muss über Taster einleitbar sein. Die Anzahl der Runden werden ständig auf dem Display für beide Bahnen angezeigt.

Produktdaten:

Die Spannungsversorgung kann über den Rennbahntravo entnommen werden oder es kann an zwei Bannanbuchsen eingespeist werden.

Der Zähler soll ohne mechanische Veränderung am Auto geschehen (kein Magnet im Auto oder mechanischen Schalter in der Schiene)

Zusätzlich soll eine Ampel an der Strecke für den Start montiert werden

Start über Taster (3 rote, 3, orange, 3 grüne Lämpchen).

Der Zähler muss pro Bahn angezeigt werden.

unter anderem:

- Erstellung eines Prototypen auf Lochrasterplatinen
- Ein 6-poliger ISP-Adapter ist vorzusehen, sodass der Atmega in der Schaltung programmiert werden kann

1.2 Pflichtenheft

Zielbestimmung:

Das Projekt realisiert zwei Rundenzähler, der die Runden für eine Carrerabahn zählt. Außerdem wird noch eine Ampelsteuerung hergestellt, die die Startphase für die Autos einleitet.

Produkteinsatz:

Es wird für jede Bahn ein Zähler benötigt. Die Ampel wird im Start und Zielbereich eingesetzt.

Produktfunktion:

Es werden zwei Lichtschranken an einer Brücke montiert um die Runden zählen zu können. Diese Brücke wird im Start und Zielbereich angebracht. In der Brücke befinden sich die LED's zur Anzeige des Starts. Die Stromversorgung erfolgt über den Travo der Carrerabahn oder beliebig über Bannanenbuchsen.

Produktdaten:

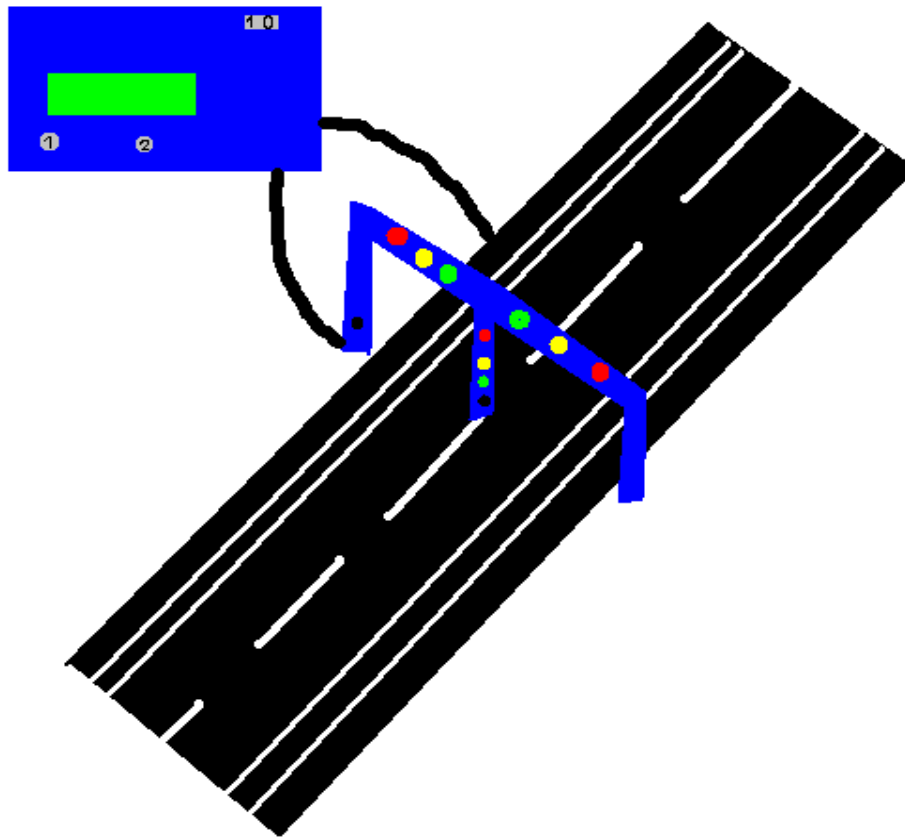
Verwendet wird ein standard zweizeiliges Display mit je 16 Zeichen. Die Bahn wird mit 6,8 – 8,0 Volt betrieben. Das Programm der Schaltung kann jederzeit über die ICSP-Schnittstelle im Gehäuse geändert werden. Die Lichtschranken sind mit zwei Infrarot LEDs und zwei Fototransistoren realisiert.

1.3 Lösungsansatz

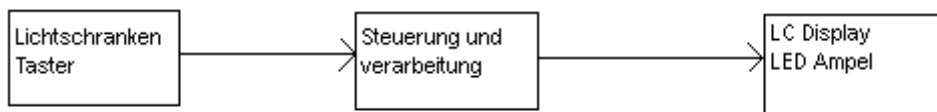
1.3.1 Beschreibung des Lösungsansatzes im Überblick

Die Lichtschranken und die Ampel wird in eine Brücke über die Bahn eingebaut. Die Steuerung und das LCD-Modul befinden sich in einem externen Gehäuse.

1.3.2 Technologieschema



1.3.3 Blockschartplan



1.4 Arbeitsplanung

Zuerst schreiben wir mit dem Kunden das Lastenheft woraus wir das Pflichtenheft erstellen.

Danach wird aus dem Pflichtenheft ein Schaltplan entwickelt, die Platine layoutet und gelötet, anschließend wird noch das Programm für den Atmega geschrieben.

Sobald alles fertig zusammengebaut ist, testen wir das Gerät, gleichen es ab und machen eine BGV A3 Prüfung.

Die genaue Arbeitsplanung ist in einem OpenWorkbench Projekt verfasst.

1.5 Kostenplanung

Pos.	Anzahl	Beschreibung	Einzelpreis	Gesamtpreis
1	3	LED rot 5mm	0,05 €	0,15 €
2	3	LED gelb 5mm	0,05 €	0,15 €
3	3	LED grün 5mm	0,05 €	0,15 €
4	1	Bananenbuchse rot	0,26 €	0,26 €
5	1	Bananenbuchse sw	0,33 €	0,33 €
6	1	Lochrasterplatine	1,85 €	1,85 €
7	1	IC Sockel	0,08 €	0,08 €
8	1	Zwillingslitze	0,70 €	0,70 €
9	1	Schrumpfschlauch	0,49 €	0,49 €
10	2	Fototransistor	0,24 €	0,48 €
11	2	IR LED 5mm	0,12 €	0,24 €
12	1	Gehäuse BL	3,10 €	3,10 €
13	1	Flachbandkabel	0,50 €	0,50 €
14	2	Taster	0,20 €	0,40 €
15	1	Schalter	0,23 €	0,23 €
16	13	LED Fassungen	0,01 €	0,13 €
17	1	Kabelkanal	3,39 €	3,39 €
18	1	LC Display	4,95 €	4,95 €
19	1	Atmega 48	1,20 €	1,20 €
20	3	BC 108	0,24 €	0,72 €
21	9	Widerstände (versch. Größen)	0,01 €	0,09 €
22	1	Poti 5K	0,19 €	0,19 €
23	1	7805	0,17 €	0,17 €
24	1	8 pol. Wannenstecker	0,07 €	0,07 €
25	2	16 pol. Wannenstecker	0,07 €	0,14 €
26	2	Kondensatoren 100n	0,07 €	0,14 €
			Gesamtpreis:	20,31 €
			enthltende MwSt.	3,86 €

1.5.1 Entwicklungskosten

Entwicklungskosten für die Stoppuhr

Pos.	Anzahl	Beschreibung	Einzelpreis	Gesamtpreis
1	32	Stunden Gesellenlohn	45,99 €	1.471,68 €
			Gesamtpreis:	1.471,68 €
			enthltende MwSt.	279,62 €

1.5.2 Fertigungskosten

Fertigungskosten für die Stoppuhr

Pos.	Anzahl	Beschreibung	Einzelpreis	Gesamtpreis
1	16	Stunden Gesellenlohn	45,99 €	735,84 €
			Gesamtpreis:	735,84 €
			enthltende MwSt.	139,81 €

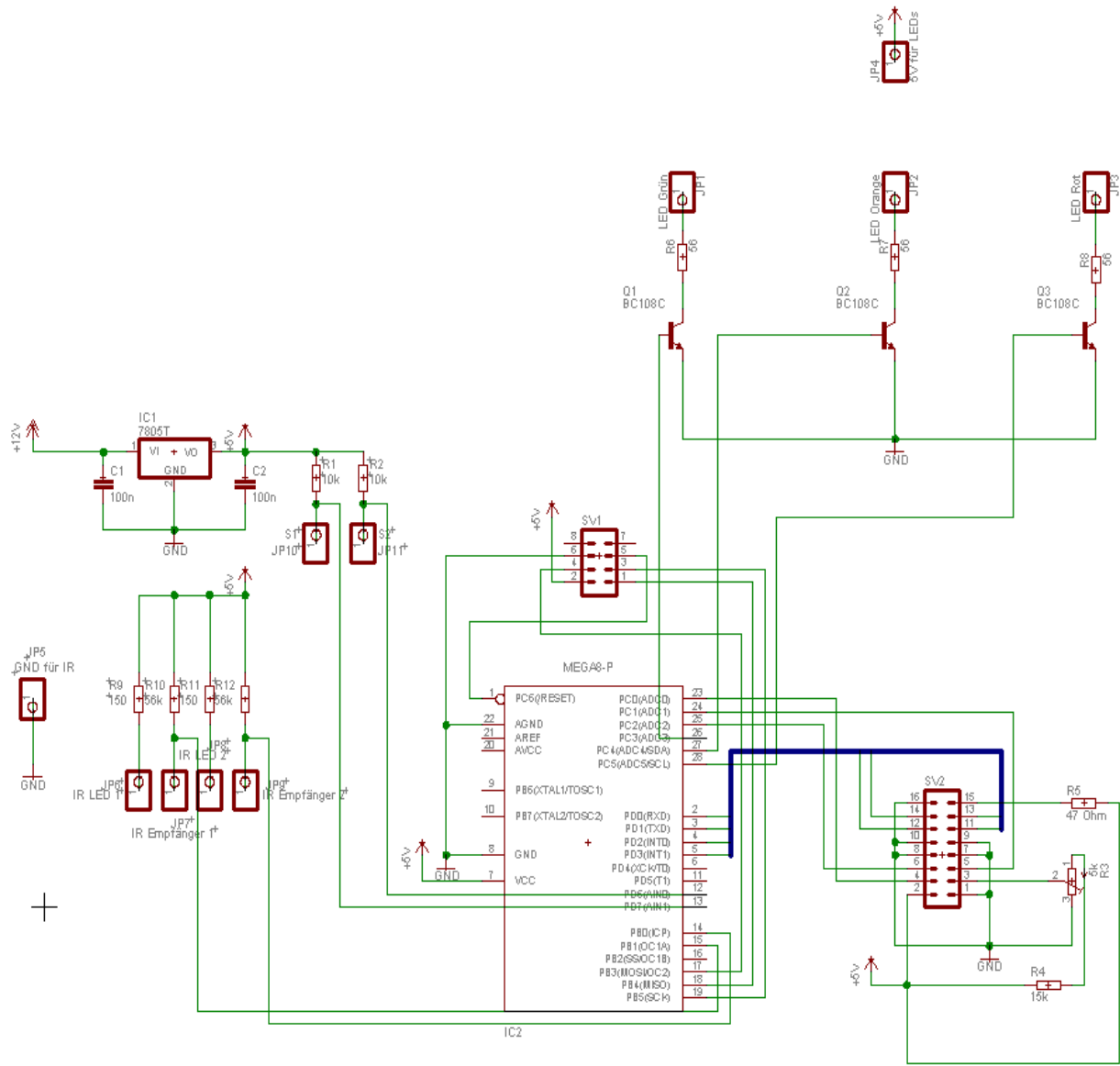
2 Durchführung

2.1 Hardware

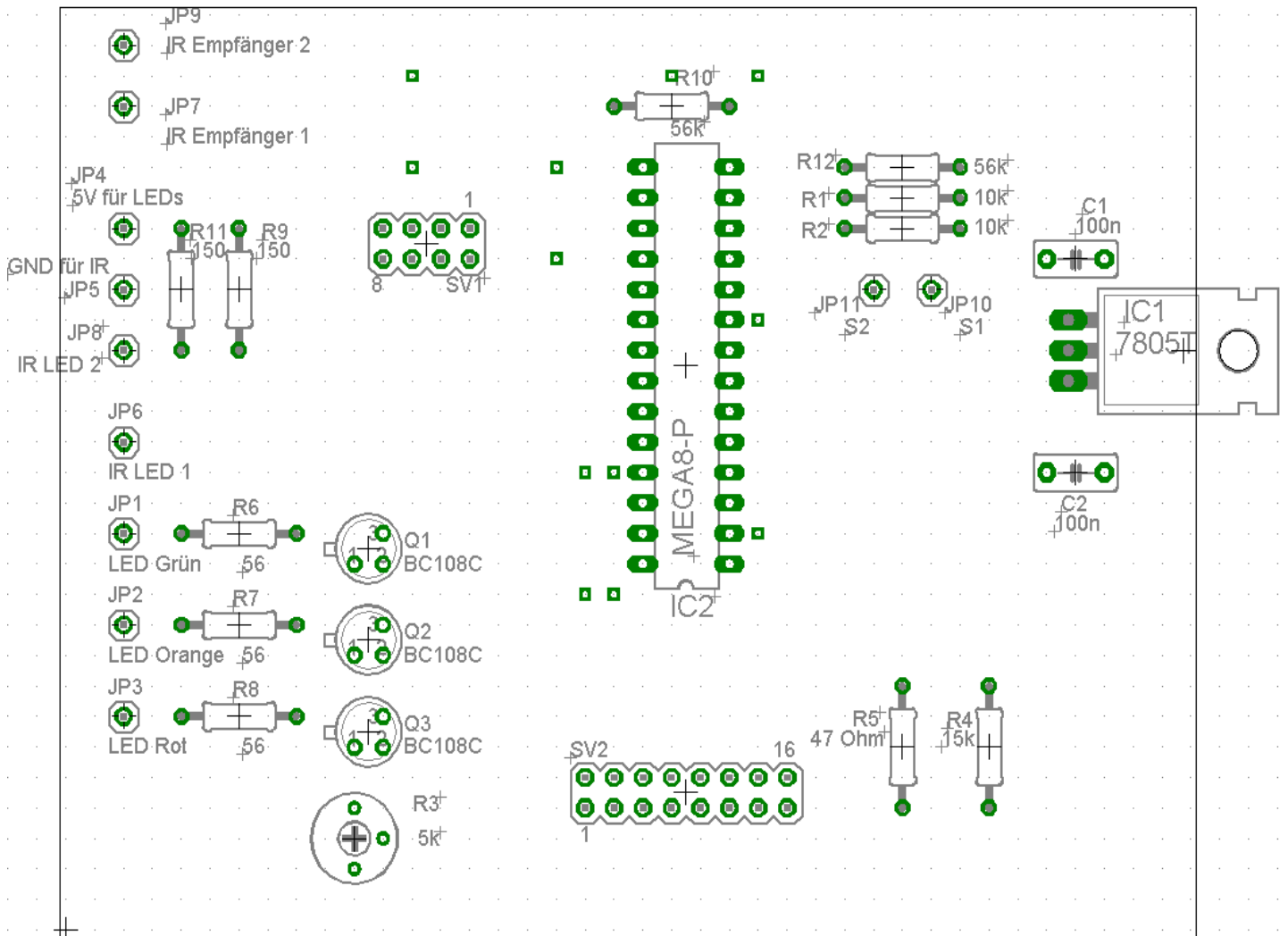
Die Ampel und die Lichtschranken befinden sich in einer Brücke die aus Kabelkanal gebaut ist. Diese Brücke wird über die Carrerabahn gestellt. Von der Brücke geht ein mehrpoliges Kabel zu einem externen Gehäuse. Dort befindet sich die Steuerung und die Anzeige.

2.1.1 Schaltungsbeschreibung

Die Schaltung wird über die Betriebsspannung der Carrerabahn versorgt, von dort wird die Spannung auf 5V Betriebsspannung für den Atemga stabilisiert. Die Taster, LEDs und die Lichtschranken sind an den Microcontroller angeschlossen. Der Microcontroller wird so programmiert, dass er die Runden durch die Lichtschrankenimpulse zählen kann . Der Ablauf einer Ampel, die durch die LEDs angezeigt wird, wird durch Taster gestartet. Über ein Taster wird die Schaltung Resetet.



2.1.2 Bestückungsplan

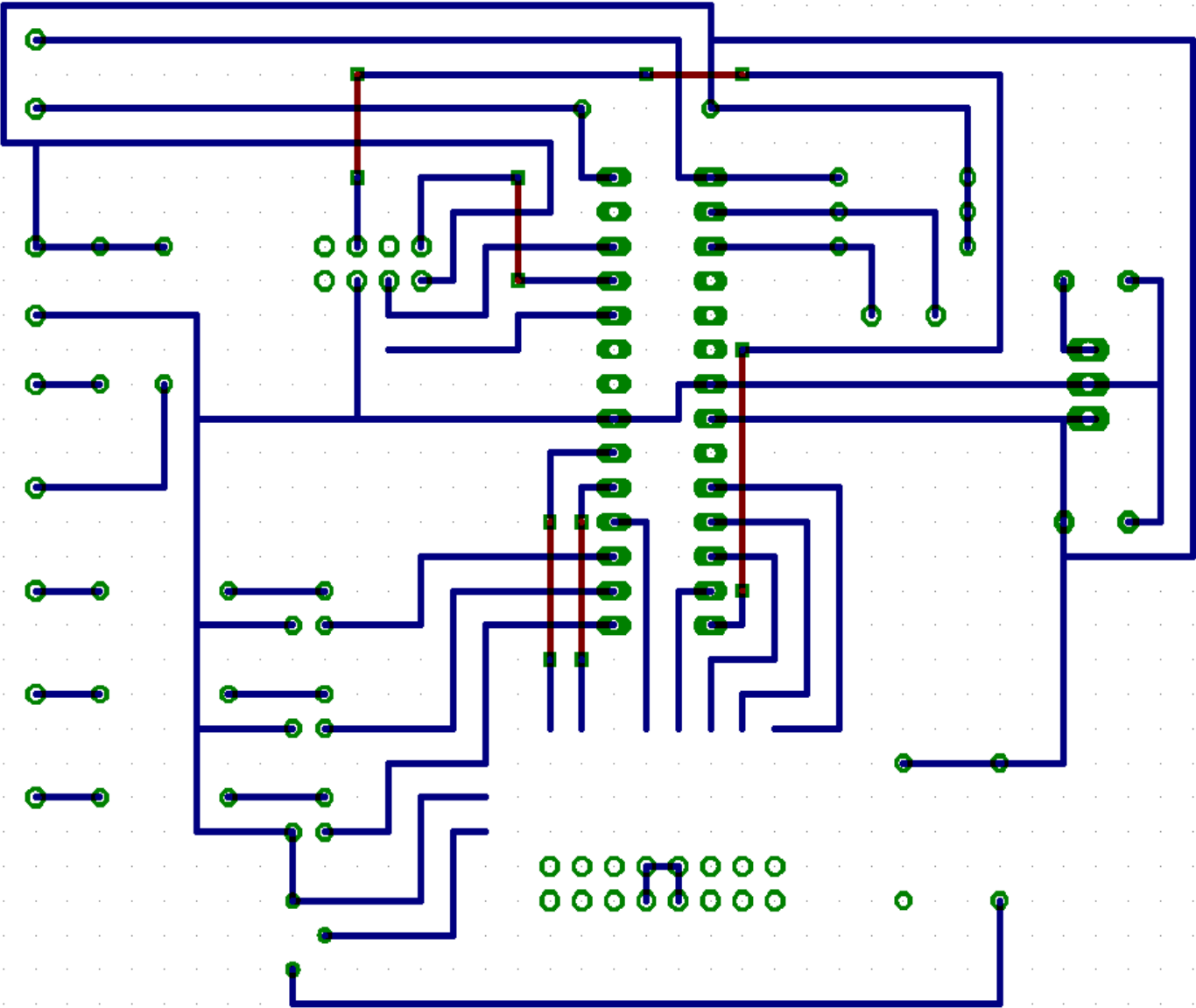


2.1.3 Bauteilliste

Anzahl Beschreibung

3	LED rot 5mm
3	LED gelb 5mm
3	LED grün 5mm
1	Bananenbuchse rot
1	Bananenbuchse sw
1	Lochrasterplatine
1	IC Sockel
1	Zwillingslitze
1	Schrumpfschlauch
2	Fototransistor
2	IR LED 5mm
1	Gehäuse BL
2	Taster
1	Schalter
13	LED Fassungen
1	Kabelkanal
1	LC Display
1	Atmega 48
3	BC 108
9	Widerstände (versch. Größen)
1	Poti 5K
1	7805
1	8 pol. Wannenstecker
2	16 pol. Wannenstecker
2	Kondensatoren 100n
2	Flachbandkabel
1	mehrpoliges Kabel
1	Div. Befestigungsmaterial

2.1.4 Platinenlayout



2.2 Software

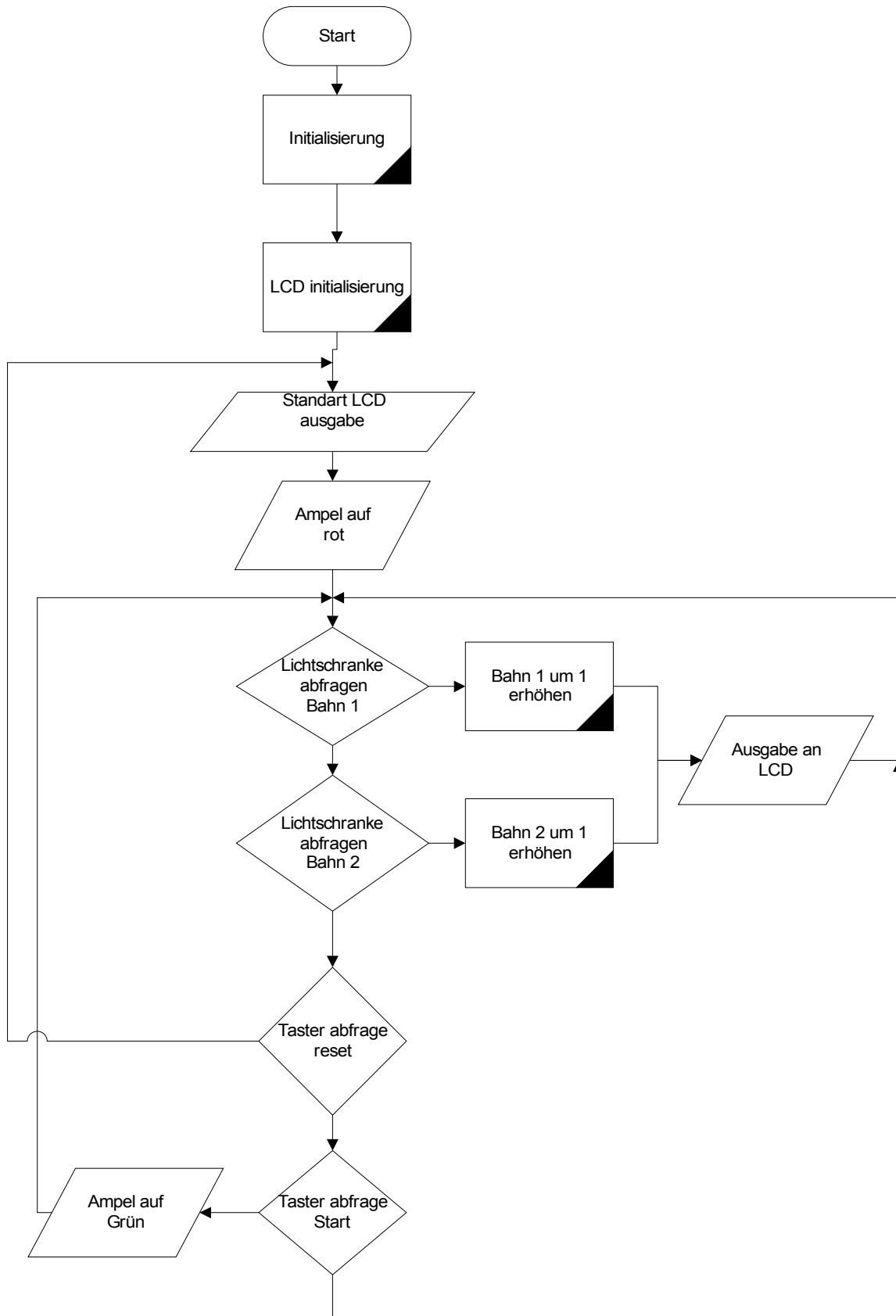
2.2.1 Entwicklungsumgebung / Programmiersprache

Das Programm ist in C geschrieben. Es wurde mit AVR-Studio 4 erstellt und getestet.

2.2.2 Programmbeschreibung

Das Programm enthält als erstes die Variablen deklaration, anschließend findet eine Initialisierung des Displays statt und die Standartanzeige wird ausgegeben. Die rote Ampel leuchtet. Nachdem die Initialisierung abgeschlossen ist, kommt das Hauptprogramm mit einer Endlosschleife. In dieser Endlosschleife werden die Lichtschranken und die Taster abgefragt. Sollte einer der Lichtschranken ein Signal haben, geht es in ein Unterprogramm. Dort wird dann gezählt und der neue Wert auf das Display ausgegeben. Wenn der Reset-Taster Signal bekommt, wird zurück zum Anfang vom Hauptprogramm gesprungen und die Ampel und der Zähler werden zurückgesetzt. Sollte der Taster für die Ampelsteuerung ein Signal bekommen, wird zum Ampelunterprogramm gesprungen. Dort wird dann die gelbe LED dazugeschaltet, nach der Phase gehen die roten und gelben LEDs aus und die grünen LEDs werden eingeschaltet. Danach wird wieder zurück und die Endlosschleifegesprungen.

2.2.3 Programmablaufplan / Struktogramm



2.2.4 Quelltext

```
#include <avr/io.h> //Registernamen einbinden
```

```
/*
```

```
Programm made by Stefan Büll und Thies Hansen
```

```
16.06.09 Ampelsteuerung und Rundenzähler für Carrerabahn
```

Pin Belegung

```
PB0 = IR Empfänger 1
```

```
PB1 = IR Empfänger 2
```

```
PB2 = -----
```

```
PB3 = ICSP
```

```
PB4 = ICSP
```

```
PB5 = ICSP
```

```
PB6 = Taster 1
```

```
PB7 = LED Grün
```

```
PC0 = LCD DB 4
```

```
PC1 = LCD DB 5
```

```
PC2 = LCD DB 6
```

```
PC3 = LCD DB 7
```

```
PC4 = -----
```

```
PC5 = -----
```

```
PC6 = ICSP
```

```
PC7 = -----
```

```
PD0 = LED Orange
```

```
PD1 = LED Rot
```

```
PD2 = LCD R/W
```

```
PD3 = LCD E
```

```
PD4 = LCD R/S
```

```
PD5 = -----
```

```
PD6 = -----
```

```
PD7 = -----
```

```
Reset= Taster 2
```

Variablen vergeben

```
*/
```

```
void rot(void);
```

```
void orange(void);
```

```
void gruen(void);
```

```
void ampelabfrage(void);
```

```
void delay_ms(unsigned int anz_milisekunden);
```

```
void delay_100us(unsigned int anz);
```

```
void init_ports(void);
```

```

void lcd_init_4bit(void);
void lcd_write_data(unsigned char data);
void lcd_write_command(unsigned char command);
void lcd_write_command8(unsigned char command);
void lcd_enable(void);
void lcd_write_text(char * textstring);
void Links(void);
void Rechts(void);

int  bahn1;
int  bahn2;

char ausgabertext[20]="          ";

int main(void)                                //Hauptprogramm
{

init_ports();                                //Initialisierung der Ports
lcd_init_4bit();                              //Initialisierung des Displays
bahn1 = 0;                                    //Zähler der Bahnen auf null
bahn2 = 0;

lcd_write_command(0x80);                      //Standart LCD ausgabe
lcd_write_text("Bahn 1: ");
sprintf(ausgabertext,16,"%-2d",bahn1); //Dezimalwert in String drucken, max. 16 Zeichen
lcd_write_text(ausgabertext);                //String ausgeben
lcd_write_text("Runden");
lcd_write_command(0xC0);
lcd_write_text("Bahn 2: ");
sprintf(ausgabertext,16,"%-2d",bahn2); //Dezimalwert in String drucken, max. 16 Zeichen
lcd_write_text(ausgabertext);                //String ausgeben
lcd_write_text("Runden");
rot();                                       //rote Ampel an

while (1)                                    //Endlosschleife im Hauptprogramm
{
    if((PINB & 0x01) == 0x01)                //Abfrage Lichtschranke erste Bahn
    {
        if((PINB & 0x01) == 0x00)
        {
            Links();
        }
    }
}

```

```

        if((PINB & 0x02) == 0x02)           //Abfrage Lichtschranke zweite Bahn
        {
            if((PINB & 0x02) == 0x00)
            {
                Rechts();
            }
        }
// TASTERABFRAGE
if((PINB & 0b01000000) == 0x00)
// Abfrage für die Ampel
    {
        ampelabfrage();
    }
}                                           //Ende Schleifenblock

}                                           //Ende Hauptprogrammblock

////////////////////////////////////
////////////////////////////////////

void Links(void)                           //Unterprogramm für die 1. Bahn
{
    bahn1 = bahn1 + 1;                       //Zähler + 1 zählen
    lcd_write_command(0x80);                 //Auswählen der ersten Zeile
    lcd_write_text("Bahn 1: ");
    sprintf(ausgabetext,"%-2d",bahn1); //Dezimalwert in String drucken, max. 16 Zeichen
    lcd_write_text(ausgabetext);           //String ausgeben
    lcd_write_text("Runden");
}

void Rechts(void)                           //Unterprogramm für die 2. Bahn
{
    bahn2 = bahn2 + 1;                       //Zähler + 1 zählen
    lcd_write_command(0xC0);                 //Auswählen der 2. Zeile
    lcd_write_text("Bahn 2: ");
    sprintf(ausgabetext,"%-2d",bahn2); //Dezimalwert in String drucken, max. 16 Zeichen
    lcd_write_text(ausgabetext);           //String ausgeben
    lcd_write_text("Runden");
}

```

```

/////////////////////////////////////////////////////////////////
/////////////////////////////////////////////////////////////////
//-----Unterprogramm für die Ampel-----
void ampelabfrage(void)
{
if (PINB6==1);
{
    rot();                // rote Ampel an
    delay_ms(800);        // Warteschleife
    orange();             // orange Ampel an
    delay_ms(3000);       // Warteschleife
    PORTD=PORTD & 0b11111100; // rote und orange Ampel aus
    gruen();              // grüne Ampel an
}
};

void rot(void)
{
    PORTD=PORTD | 0b00000010;
}

void orange(void)
{
    PORTD=PORTD | 0b00000001;
}

void gruen(void)
{
    PORTB=PORTB | 0b10000000;
}

/////////////////////////////////////////////////////////////////
/////////////////////////////////////////////////////////////////
void init_ports(void)
{
    DDRB=0b10111000;      // Port initialisierung
    DDRC=0b11111111;
    DDRD=0b11111111;

    PORTB=0x00;           //alle Ausgänge auf null
    PORTC=0x00;

```

```
PORTD=0x00;
```

```
}
```

```
////////////////////////////////////////////////////////////////  
////////////////////////////////////////////////////////////////
```

```
void lcd_enable(void)
```

```
{
```

```
int i;
```

```
PORTD = PORTD | 0b00001000;           //Enable für LCD ausgeben
```

```
i=0; //Warten
```

```
PORTD = PORTD & 0b11110111;
```

```
return;
```

```
}
```

```
////////////////////////////////////////////////////////////////  
////////////////////////////////////////////////////////////////
```

```
void lcd_write_data8(unsigned char data)
```

```
{
```

```
PORTD = PORTD & 0b11110111; // R/W auf 0 für Write
```

```
PORTD = PORTD | 0b00010000; // RS auf 1 für Datenregister
```

```
PORTC=data;
```

```
lcd_enable();
```

```
delay_ms(2);           // Viel Sicherheit: Manche brauchen 40 us, andere 1,6ms
```

```
}
```

```
//-----oberen und unteren 4 Datenbits auf LCD übertragen-----
```

```
void lcd_write_data(unsigned char data)
```

```
{
```

```
unsigned char nibble_h, nibble_l;
```

```
PORTD = PORTD & 0b11110111;
```

```
PORTD = PORTD | 0b00010000;
```

```
nibble_h = (data >>4) & 0b00001111;
```

```
nibble_l = data & 0b00001111;
```

```
PORTC = nibble_h;
```

```
lcd_enable();
```

```

PORTC = nibble_l;
lcd_enable();

delay_ms(2);           // Viel Sicherheit: Manche brauchen 40 us, andere 1,6ms

}

```

```

/////////////////////////////////////////////////////////////////
/////////////////////////////////////////////////////////////////
void lcd_write_command8(unsigned char command)
{
PORTD = PORTD & 0b11111011; // R/W auf 0 für Write
PORTD = PORTD & 0b11101111; // RS  auf 0 für Befehlsregister
PORTC = command;

```

```

lcd_enable();
delay_ms(2);           // Viel Sicherheit: Manche brauchen 40 us, andere 1,6ms

}

```

```

//-----oberen und unteren 4 Kommandobits übertragen-----
void lcd_write_command(unsigned char command)
{
unsigned char nibble_h, nibble_l;

PORTD = PORTD & 0b11111011; // R/W auf 0 für Write
PORTD = PORTD & 0b11101111; // RS  auf 0 für Befehlsregister

nibble_h = (command >>4) & 0b00001111;
nibble_l = command & 0b00001111;

```

```

PORTC = nibble_h;
lcd_enable();
PORTC = nibble_l;
lcd_enable();

delay_ms(2);           // Viel Sicherheit: Manche brauchen 40 us, andere 1,6ms

```



```

void delay_ms(unsigned int anz_milisekunden)
{
unsigned int i,k,l;

for(i=0;i<anz_milisekunden; i=i+1)
    {
        for(k=0;k<39; k=k+1)
            {
                l=0xff; //Irgendeine sinnlose Anweisung zur Zeitverschwendung
            }
        l=l+1;        //s.o.
        l=11;        //s.o.
    }
}

```

```

void delay_100us(unsigned int anz)
{
unsigned int i,l;

l=l+1;        //Zeit vergeuden
l=l+1;        //Zeit vergeuden
l=l+1;        //Zeit vergeuden
l=l+1;        //Zeit vergeuden
for(i=1;i<anz; i=i+1)
    {
        l=l+1;        //Zeit vergeuden
        l=l+1;        //Zeit vergeuden
        l=l+1;        //Zeit vergeuden
        l=l+1;        //Zeit vergeuden
        l=l+1;        //Zeit vergeuden
    }
}

```

3 Kontrolle und Dokumentation

3.1 Inbetriebnahme, Mess- und Prüfprotokolle

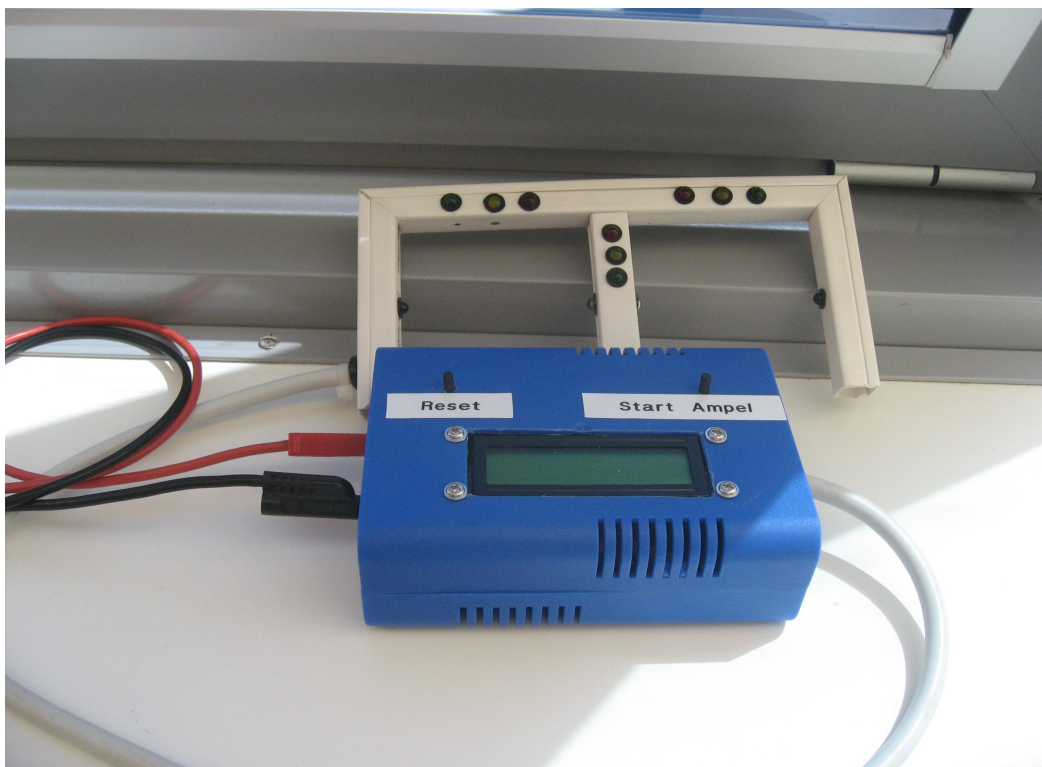
1. Überprüfen der internen Betriebsspannung am 7805. Sie sollte 5,0 Volt betragen.
2. Überprüfen der LCD Anzeige (Kontrast mit Hilfe des Trimmers einstellen)
3. Startphase einleiten um LEDs zu Testen.
4. Reset der Schaltung mit dem Resettaster durchführen.

3.2 Bedienungsanleitung

Das Gerät kann wahlweise mit 6,5 – 8 Volt betrieben werden. Die Spannung wird einfach an die Bannanbuchsen an der Seite angelegt. Mit dem Schalter an der Seite wird das Gerät eingeschaltet. Die Ampel sollte jetzt rot leuchten (sollte dies nicht der Fall sein, überprüfen Sie nocheinmal die Spannung, ansonsten wenden Sie sich bitte an den Kundendienst). Auf dem Display erfolgt jetzt eine Anzeige. Um die Startphase einzuleiten, betätigen Sie bitte den Taster rechts. Die Ampel sollte jetzt Rot/Orange leuchten und danach auf Grün umschalten. Bei jeder Durchfahrt der Autos, wird auf dem Display für die jeweilige Bahn gezählt. Für den Reset der Schaltung betätigen Sie bitte den linken Taster.

3.3 Fotos

3.3.1 Gesamtaufbau





3.4 Kritik der eigenen Projektplanung und - durchführung

3.4.1 Zeitplanung

Wir haben uns ein klein Wenig in der Bauphase verkalkuliert, aber die Zeit haben wir bei dem Programmieren wieder aufgeholt und sind somit schneller fertig geworden, als wir geplant haben.

3.4.2 Kostenplanung, tatsächliche Kosten

Unsere Kostenplanung ist auch sehr gut hingekommen. Wir mussten teilweise Bauteile austauschen, aber da hat sich am Preis nicht viel geändert.

4 Quellenverzeichnis

4.1 Datenblätter

- Display.pdf
- Atmega48.pdf
- BC 108.pdf

4.2 Beispielprojekte

- Kein ähnliches Projekt gefunden.

4.3 Fachliteratur

- Tabellenbuch
- Internet