Deutscher Amateur Radio Club e.V
Distrikt Württemberg

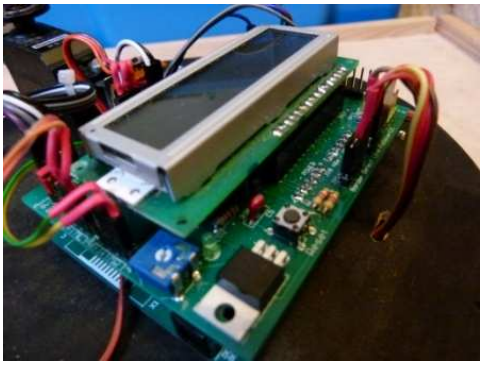
Referat für Jugend-Ausbildung-Weiterbildung
Axel Tüner (DF9VI)

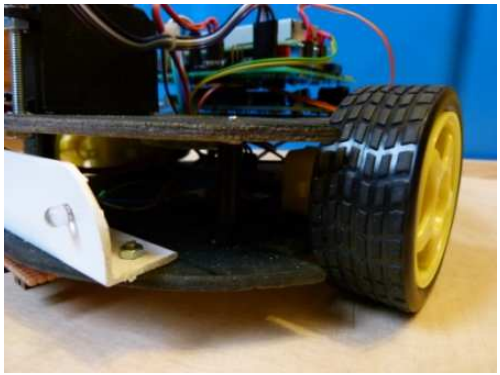
ROBBY III, der Arduino-Roboter

Robby ist ein kleiner, von einem Arduino gesteuerter fahrbarer Roboter. Robby kann ferngesteuert werden, kann mit Dir sprechen, und sich allein zurechtfinden. Ein LCD gibt weitere Informationen. Auch auf Handy-Signale reagiert Robby. Er misst z.B. die Lufttemperatur, Helligkeit und weitere Umweltparameter und kommuniziert mit Deinem PC. Robby wird über eine USB-Schnittstelle programmiert. Die Programmier-sprache ist „C“. Die Programmieroberfläche ist recht einfach gestaltet und für Kinder ab ca. 12 Jahren geeignet.

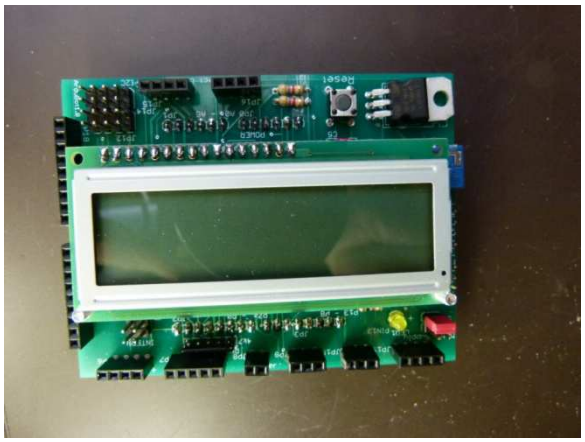
Robby besteht aus folgenden Komponenten:
- Hardwaresatz mit Getriebemotoren, Halterungen, Rädern, Schrauben usw.
- Arduino Bausatz
- Bausatz Multifunktionsplatine
- Lichtleistensatz, Tastenleiste
- Bluetooth-Modul
- RF-Modul
- Ping-Modul
- Line-Follower-Modul
- Kompass-Modul
- Lage-Modul
- Infrarot-Fernsteuerung

Robby gibt es nur als einzelne Komponenten. Das Projekt ist modular aufgebaut und kann auch noch fast beliebig erweitert werden. Es stehen über die Erweiterungsmöglickeiten viele Schnittstellen zur Verfügung.



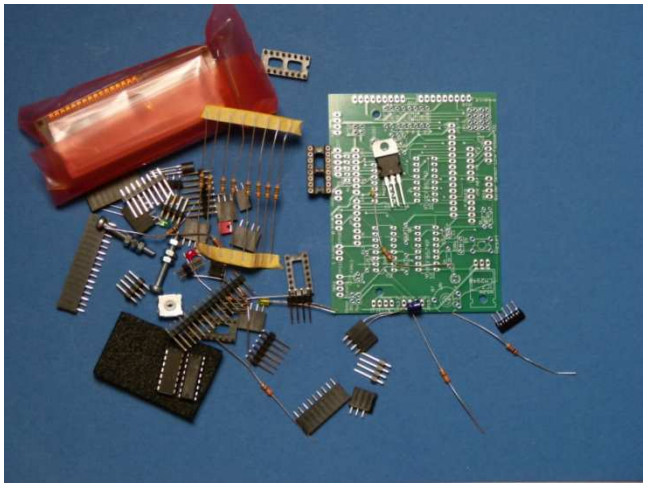


Die Robby-Multifunktions-Platine



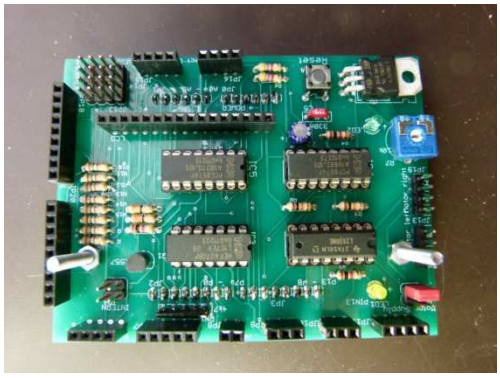
Die Motorplatine dient zu Steuerung der beiden Motoren von Robby III, aber nicht nur das. Die Motorplatine erfüllt gleich mehrere Funktionen, um nicht allzu viele Zusatz-platinen zu haben und doch die wesentlichen Funktionen des Mikrokontrollers, aber auch von mechatronischen System zu demons-trieren. Dazu sind die entsprechenden Schnittstellen und Anschlüsse auf der Platine mit untergebracht. Die Platine wird auf den Arduino Uno oder kompatible Version aufgesteckt.

Im Folgenden sind die Ports mit den zugehörigen Funktionen aufgelistet:
– JP6 (Wireless) Hier wird das RF-Modul (Bluetooth) aufgesteckt
– JP7 der Odometer-Anschluss
– JP8 Lautsprecheranschluss
– JP9, JP10 hier können zwei Servos (#0, #1) angeschlossen, JP9 (Servo 0) trägt den Ultraschall-
Sensor. JP10 ist außerdem für den Infrarot-Sensor vorgesehen.
– JP11 (Ping) hier wird das Ultraschallmodul angeschlossen
– JP13 Anschluss Motor links, bzw. Servomotor links
– JP19 Anschluss Motor rechts, bzw. Servomotor rechts
– JP12 Anschluss für den I2C-Bus
– JP15 Anschluss für den „Line Follower“-Sensor
– JP16 Anschluss für den „Border Detect“-Sensor
– JP17 LCD-Display, hier wird ein 16x2 LCD Charakter-Display angeschlossen
– JP18 8Bit-Extension-Port
– JP20 LED-Port, die Anschlüsse von JP18 sind über Widerstände herausgeführt



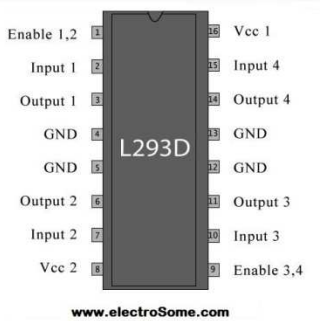
Ein 16x2 Charakter-Display kann direkt auf die Platine gesteckt werden. Für weitere Ergänzungen zur Ansteuerung über I2C-Bus ist der Port JP12 vorgesehen. Die benötigten Abschlusswiderstände befinden sich schon auf der Platine.
Über JP7 kann das Odometer zugeschaltet werden, dazu müssen zwei Unterbrechungsroutinen (Interrupt-Serviceroutinen) programmiert werden, die die Interrupts bedienen. In den nachfolgenden Kapiteln werden jeweils die Funktionen mit den an diesen Ports angeschlossenen Modulen oder Sensoren näher erklärt.

Übersicht einzelner Bauteile und Funktionen



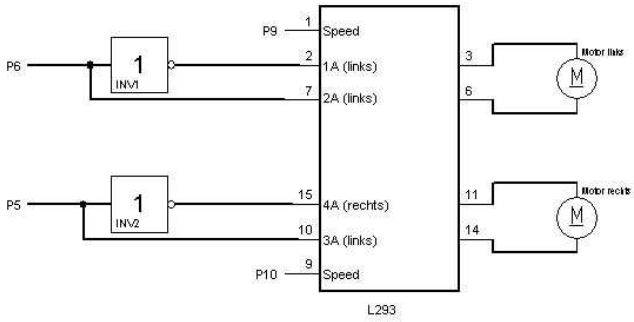
Die gelbe LED ist direkt mit PIN 13 des Arduino verbunden („Blink-LED“).
Die grüne LED zeigt die Betriebsspannung nach dem Spannungsregler an (5V).
Der Taster hat die Funktion des Arduino-Reset-Tasters. Seine Position ermöglicht einen Reset auch bei Sandwichbetrieb.
Der Transistor kann die Beleuchtung des LCD schalten. Dazu wird die Bibliothek Liquid Crystal benötigt.
lcd.setBacklight(HIGH); Einschalten des LCD
lcd.setBacklight(LOW); Ausschalten des LCD

Anschluss der Getriebemotoren



JP13 und JP 19 dienen zum Anschluss der Getriebemotoren. Angesteuert werden die Motoren über den Halbbrückentreiber L293. Dadurch ist eine Drehrichtungsänderung ganz einfach möglich. Hier ein einfaches Programm zur Erläuterung der Funktionsweise.
Rechter Motor an P 5, linker Motor an P 6
Mit HIGH oder LOW an diesen Pins wird die jeweilige Drehrichtung vorgegeben.

Ansteuerung der Motoren



Vorwärts:
digitalWrite (6,HIGH)
digitalWrite (5,HIGH);

Rückwärts
digitalWrite (6,LOW);
digitalWrite (5,LOW);

Die Geschwindigkeit wird über die Pins 9 und 10 eingestellt.
Es sind Werte von 0 bis 255 möglich.
Beispiel: analogWrite (9, 200);

Mit diesem Programm fährt Robby 3 Sekunden vorwärts und danach 3 Sekunden Rückwärts.

```
int RM = 5;
int LM = 6;
int speedRight = 10;
int speedLeft = 9;

void setup()

{
  pinMode (RM, OUTPUT);
  pinMode (LM, OUTPUT);
  pinMode (speedRight, OUTPUT);
  pinMode (speedLeft, OUTPUT);
}

void Forward () {
  digitalWrite (LM,HIGH);
  digitalWrite (RM,HIGH);
  digitalWrite (speedRight, HIGH);
  digitalWrite (speedLeft, HIGH);

}
void Backward () {
  digitalWrite (LM,LOW);
  digitalWrite (RM,LOW);
  digitalWrite (speedRight, HIGH);
  digitalWrite (speedLeft, HIGH);

}

void loop ()

{
  Forward();
  delay(3000);
  Backward();
  delay(3000);
}
```

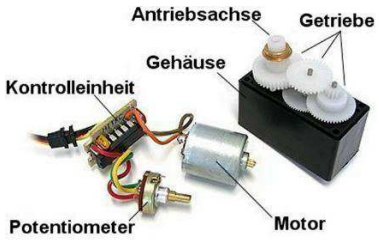
Das Lenken geschieht mit unterschiedlichen Werten für speedLeft und speedRight
Beispiel für eine Rechtskurve:
digitalWrite (speedRight, 100);
digitalWrite (speedLeft, 200);

Über den Jumper „Motor Supply“ kann die Spannung für die Getriebemotoren von 9V auf 5V umgeschaltet werden. Die Position des Jumpers ist abhängig von den verwendeten Motoren. Umgebaute Servos Typ RS2 vertragen auch 9V problemlos. 9V bedeutet Batteriespannung ohne Spannungsregler. Die Spannung hängt also vom verwendeten Akku-Pack ab,

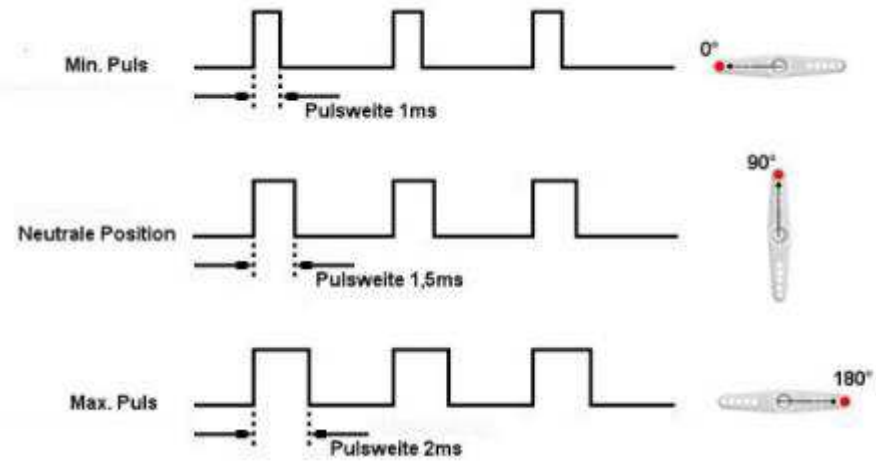
Servos

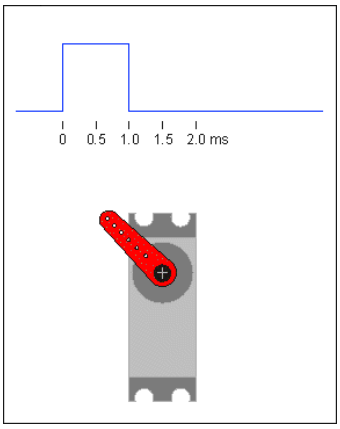
Servo bedeutet nichts anderes als ein "stellbarer Motor". Dies kann sowohl ein Wechselspannungs- sowie ein Gleichspannungsmotor sein. Der Motor dient nur dazu den Hebel des Servos in Position zu fahren. Eine Auswerteelektronik überwacht dann nur noch die Stellung des Hebels.
In unserem Fall, also bei den RC-Servos übernimmt diese Aufgabe ein direkt auf der Hebelchase befestigtes Potentiometer (einstellbarer Widerstand).
Da das Potentiometer immer mechanisch mit dem Hebel verbunden ist, kann die Elektronik so den SOLL und IST Wert bestimmen.
Der Drehwinkel des Hebels ist somit NUR durch den Drehwinkel des Potentiometers UND einem mechanischen Hindernis bestimmt.

Servos werden gesteuert, indem man ihnen einen Impuls in variabler Breite schickt. Dazu wird die Steuerleitung benutzt. Die Parameter für diesen Impuls sind: Ein minimaler Impuls, ein maximaler Impuls und die Wiederholungsrate.
Die neutrale Stellung legt die Position fest, in die der Servo die gleiche Bewegung in einer Richtung zurücklegt, sowie auch in die Gegenrichtung.
Es ist wichtig zu wissen, daß unterschiedliche Servos unterschiedliche Begrenzungen haben, aber alle gemeinsam haben eine Mittelstellung in der Position 1,5ms.



Anschlüsse:
schwarz: GND (Minus)
Rot: + 5V
Gelb: Signal (Pin 2)

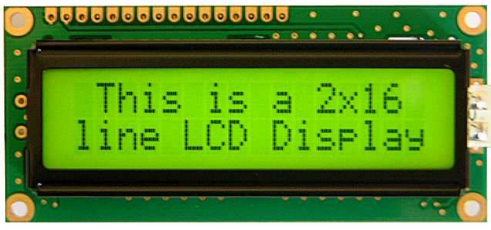




Bei diesem Programm folgt der Servo der Einstellung des Potenziometers

```
#include <Servo.h>
Servo myservo;                    // Objekt erzeugen
int potipin = 0;                  // Poti an A0
int val;
Voit setup()
 {
myservo.attach(2);                // Zugriff zum Servo an Pin 2
}
void loop()
{
val = analogRead(potipin);        // liest Potentiometer (Werte zwischen 0 and 1023)
val = map(val, 0, 1023, 0, 179);  // Skalieren
myservo.write(val);               // Servo stellen
delay(15);
}
```

LCD Display



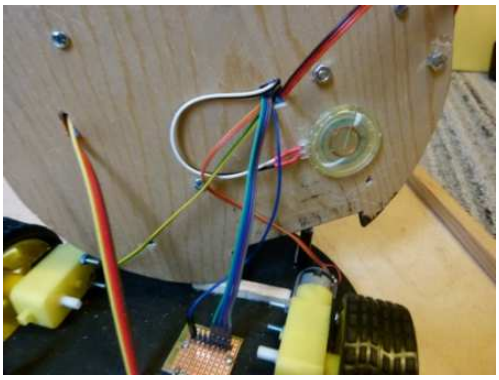
Auf das Roby III kann ein LCD-Display Typ 16x2 direkt aufgesteckt werden
Das Display wird über die so genannte I2C-Schnittstelle angesprochen. Diese Schnittstelle ist eine serielle Schnittstelle und benötigt daher nur 2 Leitungen: Takt und Daten.
In der Software wird eine Library benötigt. Die I2C-Adresse lautet 0x20. Mit dem Kommando setBacklight wird die Hintergrundbeleuchtung eingeschaltet. Init intialisiert das Display einmalig.

```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x20,16,2);
lcd.setBacklight(HIGH);
   lcd.init();
   lcd.print("Ich bin Robby");
```

6 Der Lautsprecher

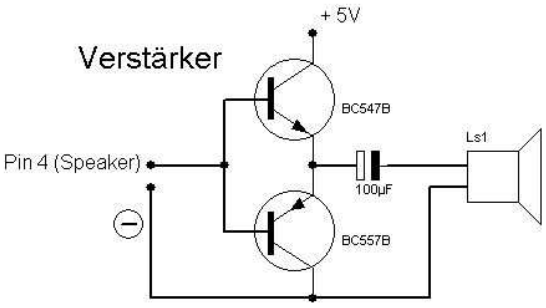
Der Roby III erhält auch einen Lautsprecher. Auf dem Motorshield ist dazu schon ein Vorwiderstand vorgesehen. Der Vorwiderstand verhindert, dass der Ausgang des Mikrokontrollers überlastet wird. Der Lautsprecher wird direktüber den Widerstand an den **Ausgang P4** des Arduino angeschlossen, dafür ist auf der Motorplatine die Stiftreihe JP8 (Speaker) vorgesehen. Mit dem Arduino ist es dann möglich, Töne zu erzeugen. Er kann damit sowohl Morsezeichen senden, wie auch Melodien erzeugen.

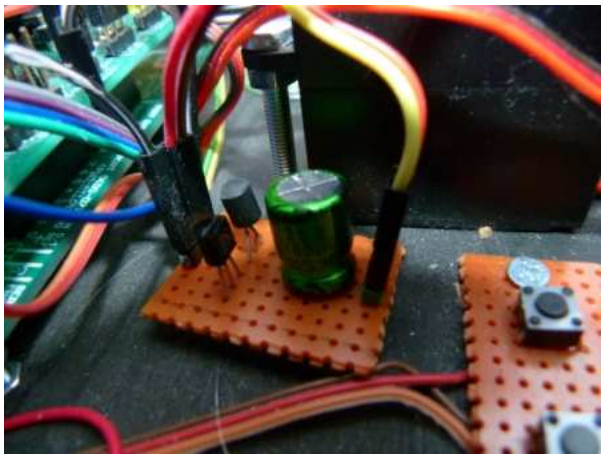
Erzeugen eines Tons von 1000 Hz:
```
Tone(4,1000);
```
Abschalten des Tons:
```
noTone(4)
```



Lautsprecher-Verstärker

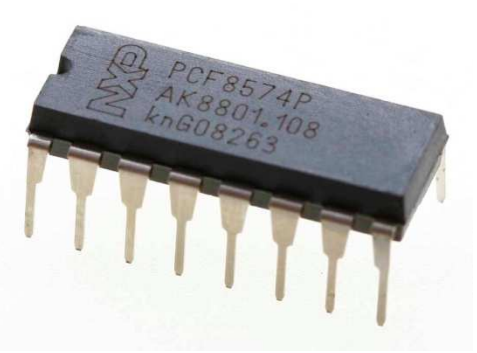
Bei Bedarf kann ein kleiner Verstärker die Lautstärke beträchtlich erhöhen. Er besteht aus nur 2 Transistoren und einem Elko.





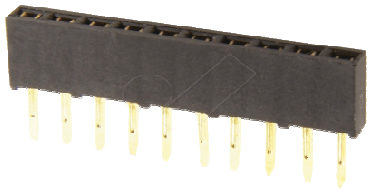
Eine kleine Lochrasterplatine nimmt die wenigen Bauteile auf.
Zum Verbinden eignen sich besonders die Jumper Wires (Dupont), die über Ebay bezogen werden können.

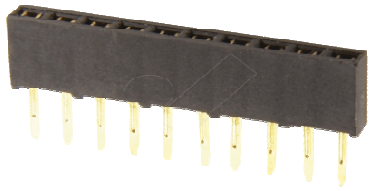
Porterweiterung

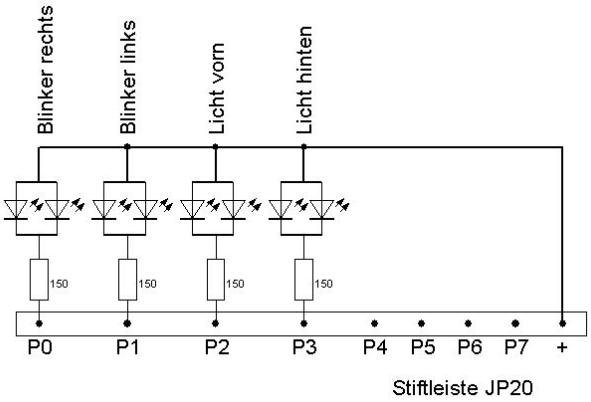


Die Stiftreihen JP18 und JP20 sind eine Porterweiterung über I2C. Dadruch haben wir die Möglichkeit, mehr Funktionen zu realisieren. Die beiden Stiftreihen werden gleich angesteuert, nur besitz Stiftreihe JP20 vorgeschaltete 150 Ohm-Widerstände. Der Widerstandswert 150 ermöglicht den Anschluß von 2 parallel geschalteten LED´s. Soll nur eine LED pro Port angeschlossen werden, muß der Widerstandswert auf ca. 330 Ohm erhöht werden. Die Stiftreihen bilden einen 8-Bit-Port ab.
Der äußere Pin von JP 18 ist Gnd, der äußere Pin von JP20 ist +5V. So ist es möglich, Erweiterungen mit LED´s oder Tasten über einen Stecker anzuschließen. Beispiel für den Anschluß von LED´s weiter unten.
Zur Ansteuerung benötigten wir die Library „Wire“. Die Adresse des I2C-Porterweiterungschips PCF8574 lautet 0x21.
Mit dem Kommando Wire.write(B0000001); wird der Ausgang „0“ auf „HIGH“ gesetzt.

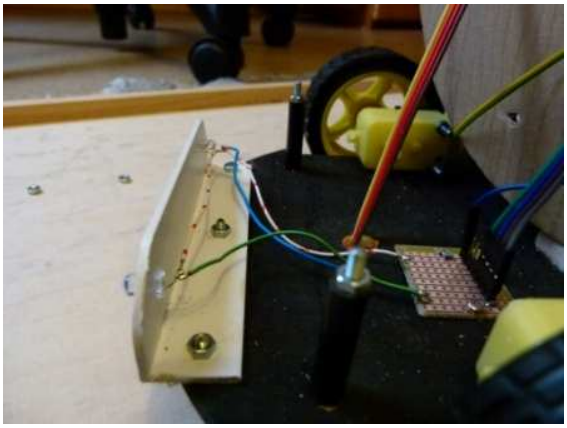
```
#include <Wire.h>
#define expander 0x21   //Adresse 2.PCF

void setup() {
   Wire.begin();
}

void loop() {
  Wire.beginTransmission(expander);
  Wire.write(B0000001);
}
```

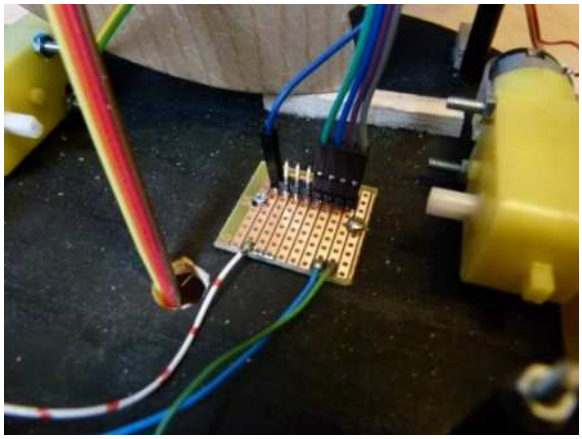






Beschaltung der 8-Bit-Porterweiterung mit LED´s für den Robby





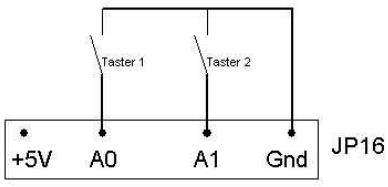
Lichtleiste vorn mit 2 hellen weißen LED´s.

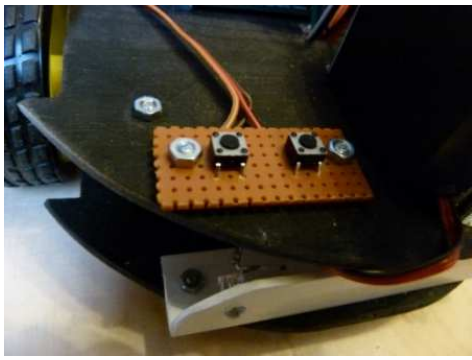
Angeschlossen an eine Verteilerplatine

Wahlweise kann die Lichtleiste erweitert werden, z.B. mit Blink-LED´s für die Richtungsanzeige und mit einer Lichtleiste für hinten (rote-Led´s).

Taster

Mit den Tastern kann man Robby Kommandos geben oder Einstellungen verändern. Es sind erst einmal 2 Taster vorgesehen. Die Taster werden an JP 16 (Border Detect) angeschlossen.
Die Funktion Border Detect wird bei Roby III nicht verwendet.
Der Pin auf der rechten Seite der Buchsenleiste JP16 ist GND. Wir schalten die Taster natürlich an GND und verwenden die internen Pull-Up-Widerstände



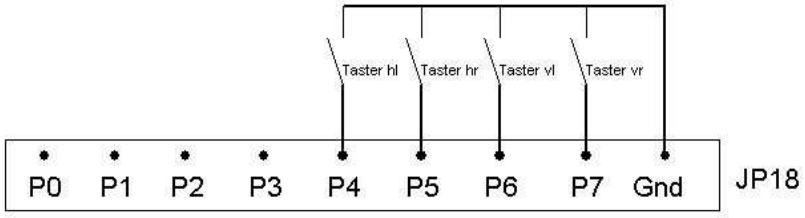


Beispiel für das Einlesen der Taster an A0 und A1

```
  pinMode (A0,INPUT);
  pinMode (A1,INPUT);
  digitalWrite (A0,HIGH);
  digitalWrite (A1,HIGH);

if (digitalRead(A0) == 0){lcd.clear();lcd.print("Taster 1"); delay(1000);break;}
if (digitalRead(A1) == 0){lcd.clear();lcd.print("Taster 2"); delay(1000);break;}
```

Berührungstaster

Mit 4 Berührungstastern („Fühlern“) kann Robby Hindernisse erkennen und darauf reagieren. Die 4 Taster befinden sich vorne links(vl), vorne rechts(vr), hinten links(hl) und hinten rechts(hr).
Die Taster werden an JP 18 angeschlossen, wie auf dem Bild zu sehen ist. Geschaltet werden sie nach Gnd.



Eingelesen werden die Taster über den I2C-Port-Baustein PCF 8574P

Beispielprogramm für das Einlesen der 4 Taster:

```
#include <Wire.h>

    Wire.requestFrom(0x21, 1);
    byte switchbyte = Wire.read();
    boolean vr = bitRead(switchbyte,7);
    boolean vl = bitRead(switchbyte,6);
    boolean hr = bitRead(switchbyte,5);
    boolean hl = bitRead(switchbyte,4);

if (vl == 0) {lcd.clear();lcd.print("Taster VL"); delay(1000);break;}
usw…
```

Ping-Modul

Verwendung finden kann der Ultraschall-Distanz-Sensor HC-SR04. Das Modul kann direkt aufgesteckt oder über ein 4-ploiges Verbindungskabel angeschlossen werden.
Funktion des Moduls:
Ein 10 us langer Impuls am Trigger-Pin (Trig) löst 8 40kHz Ultraschall Impulse aus. Die gemessene Entfernung ist proportional zur Echo-Puls-Weite am Signal Pin und kann und kann durch die folgende Formel berechnet werden.
Distanz = Pulsweite Signal-PIN / 29 /2
(Die Schallschwindigkeit beträgt ca. 29us/cm und das Signal muss den doppelten Weg zum Hindernis und zurück laufen).

Hier ein einfaches Testprogramm, das die Entfernung zum Hindernis in cm auf dem Seriellen Monitor ausgibt:

```
int pingPin = 12;
int inPin = 11;

void setup() {
Serial.begin(9600);
}

void loop()
{
    long duration, inches, cm;
    pinMode(pingPin, OUTPUT);
    digitalWrite(pingPin, LOW);
    delayMicroseconds(2);
    digitalWrite(pingPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(pingPin, LOW);
    pinMode(inPin, INPUT);
    duration = pulseIn(inPin, HIGH);
    inches = microsecondsToInches(duration);
    cm = microsecondsToCentimeters(duration);
    Serial.println(cm, DEC);
    delay(100);
}

  long microsecondsToInches(long microseconds) {
  return microseconds / 74 / 2;
  }

  long microsecondsToCentimeters(long microseconds){
  return microseconds / 29 / 2;
  }
```

Bluetooth-Modul

Das Bluetooth-Modul kann direkt aufgesteckt werden (Stiftleiste „Wireless“).

Es belegt die Pins P0 und P1, also genau die serielle Schnittstelle. Achtung: bei aufgesteckten Modul kann der Arduino nicht programmiert werden, eben weil das Modul die serielle Schnittstelle belegt. Also immer erst Modul bei abgeschaltetet Versorgungsspannung (!!!) abziehen.

Das Modul wird wie eine serielle Schnittstelle angesprochen. Sendet ein Handy eine Buchstabenfolge, so wird diese mit folgendem Programm auf dem seriellen Monitor angezeigt:
Einlesen von der seriellen Schnittstelle mit Ausgabe auf den seriellen Monitor.

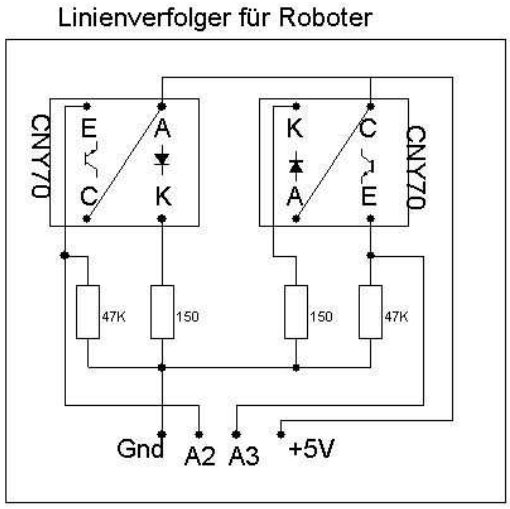
```
void setup()
{
  Serial.begin(9600);
}

void loop()
{char inByte;
  if (Serial.available() > 0) {
    inByte = Serial.read();
    Serial.print(inByte);
  }
}
```

Nur wie kann ein Smartphone-Handy Buchstaben senden?
Dazu benötigt man eine APP, z.B. den ITEAD BT Debugging Assistant. Die APP bekommt man z.B. hier: http://blog.iteadstudio.com/make-arduino-talk-with-android-by-bluetooth/

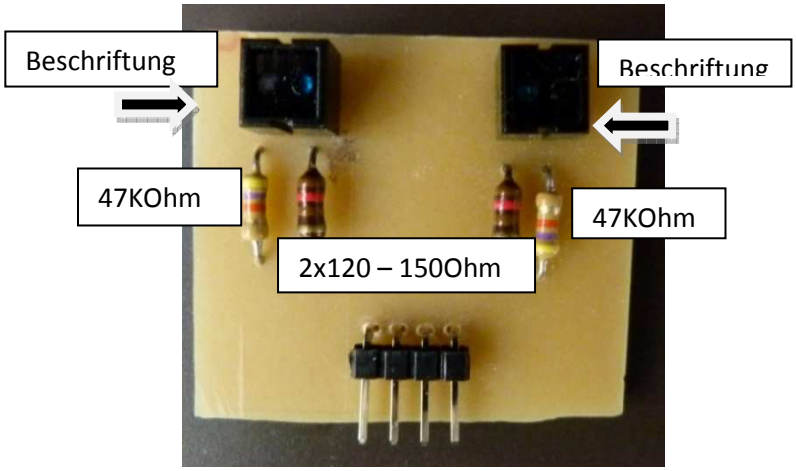
Nach dem Starten der APP sucht das Programm einen Bluetooth-Sender (Search Device). Danach den gefundenen Eintrag (z.B. „linvor“) anklicken und in das Feld den Text eingeben. Der Text erscheint über die serielle Schnittstelle auf dem Monitor.

Linienverfolger

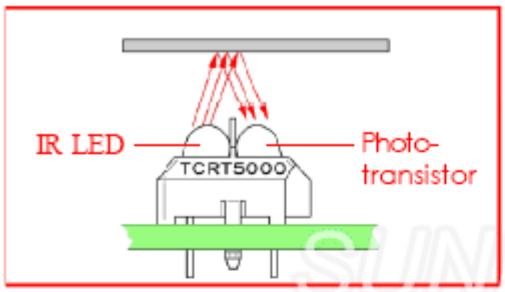
Mit diesem Programm folgt Robby einer
ca. 2cm breiten schwarzen Linie.

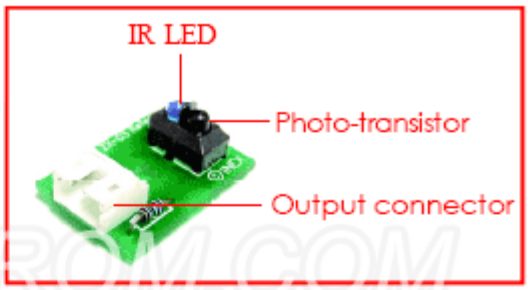



```
int SpeedRight= 10;
int SpeedLeft = 9;
int RM = 5;
int LM = 6;
int irLeftPin = A2;
int irRightPin = A3;
int Left, Right;
int Ref = 100;   /Grenzwert Schwarzerkennung

void setup() {
  pinMode (RM, OUTPUT);
  pinMode (LM, OUTPUT);
  pinMode (SpeedRight, OUTPUT);
  pinMode (SpeedLeft, OUTPUT);

}

void loop() {
    Right = analogRead(irRightPin);
    Left = analogRead(irLeftPin);

    if (Left < Ref && Right > Ref) {          // Linker Sensor weiß, rechter Sensor schwarz
     digitalWrite (LM,HIGH);
     digitalWrite (RM,LOW);
     analogWrite (SpeedLeft, 255);
     analogWrite (SpeedRight, 0);
  }
    if (Left > Ref && Right < Ref) {          // Rechter Sensor weiß, linker Sensor schwarz
     digitalWrite (RM,HIGH);
     digitalWrite (LM,LOW);
     analogWrite (SpeedLeft, 0);
     analogWrite (SpeedRight, 255);
  }
 }
```

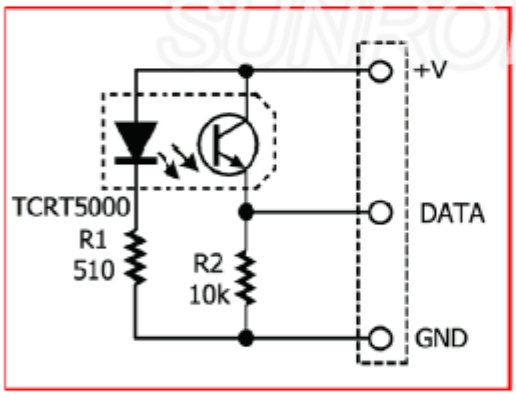


The Infrared Reflector consist of 2 parts, the Emitter and the Detector. This simple circuit shows how its functions. It shoots out an LED light and bounces off the surface which in turns returns back to the detector processing and then giving a digital data value back to the i-BOX it is connected to.

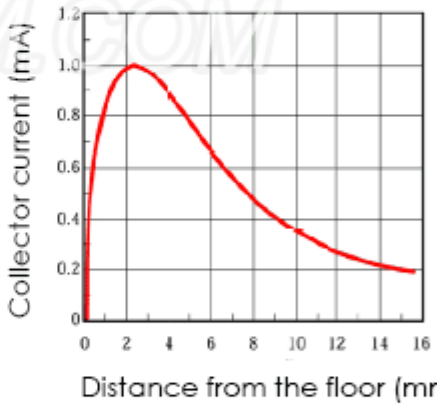




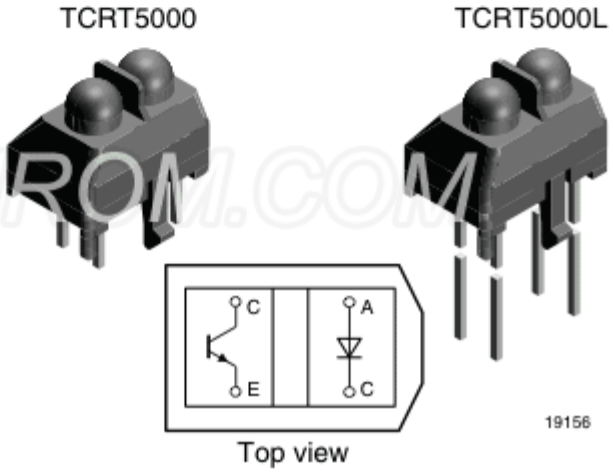
The Infrared Reflector can act Analog sensor and Digital sensor. The suitable range detection is during 2 to 10mm. The best point is 3mm. (See the performance graph). The heart of this sensor is **TCRT5000** Infrared reflector device. Output of this sensor can define 2 types. *One is analog in DC voltage term. Range is 0 to 5V.* If the photo-transistor within TCRT5000 can detect more infrared light density, it can conduct more. Output voltage go high. In the other hand, it can detect low light density. Output voltage will be low. *Another type is Digital signal output. If sensor can detect more light density, It can send logic "1" to output and send logic "0" in case detect low light density.*



Infrared Reflector sensor schematic

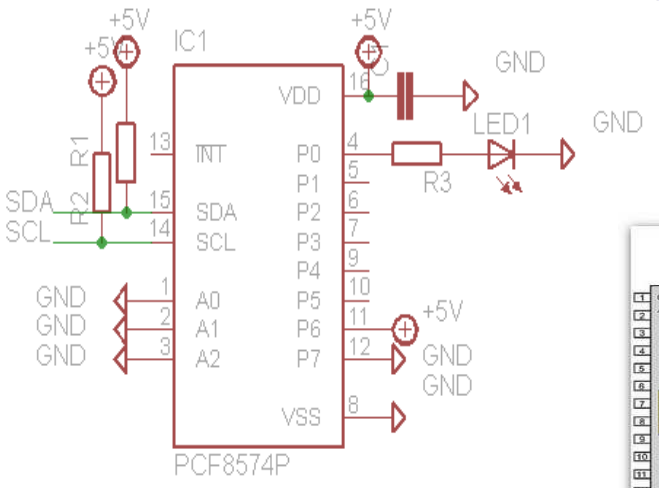


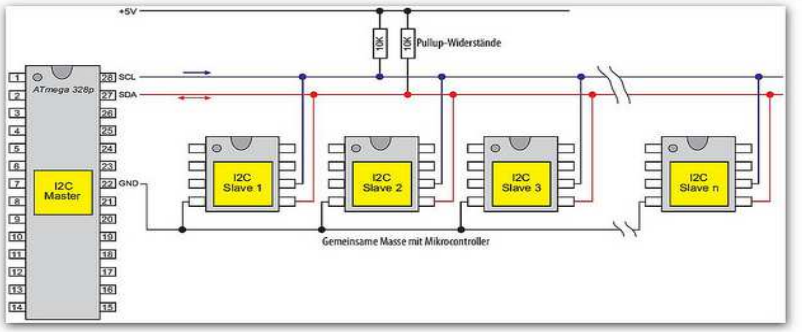
Performance graph of IR Reflector



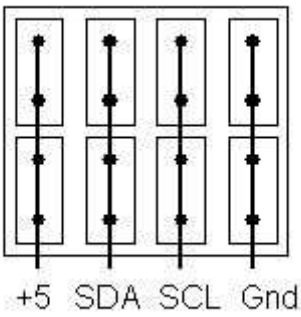
I2C-Bus

I²C ist ein synchroner serieller Zweidraht-Bus (eine Daten- und eine Taktleitung), der für die Kommunikation zwischen ICs über kleine Distanzen geeignet ist. Entwickelt wurde er Anfang der 80er Jahre von Philips. Gesprochen "I quadrat C" steht für IIC = Inter IC Bus.

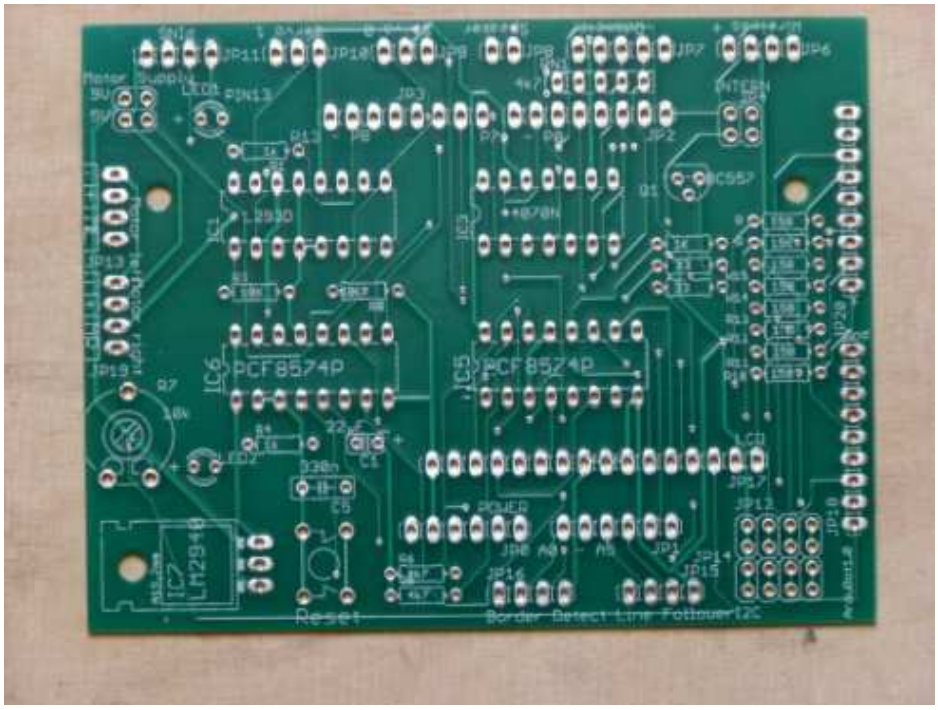




I2C-Bus mit Master und 4 Slaves
Beim Multifunktionsboard sind die beiden Pull-Up´s auf dem Board aktiv.



Steckerbelegung
4 parallele Steckmöglichkeiten
Es können 4 Kabel (4-polig) aufgesteckt werden.

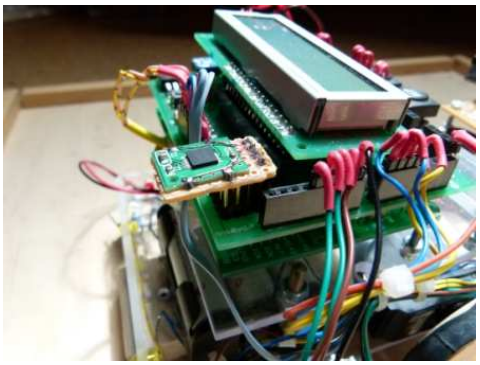


Das Kompassmodul

Wir verwenden das HDMM01 von Pollin. Auf dem Modul befindet sich ein MMC2120 von MEMSIC. Der Sensor misst in zwei Achsen mit einer Auflösung von 12 Bit zwischen ±2 Gauss und verfügt über eine I²C Schnittstelle, so dass er sich leicht an einen Mikrocontroller anbinden lässt. Natürlich hab ich mir gleich eins von den Modulen geordert und ein wenig damit rumgespielt.

Anschluss | Der Anschluss des Moduls an einen AVR ist sehr einfach. Zwei Pins des Moduls dienen der Stromversorgung (der Sensor mag lt. Datenblatt alles zwischen 2.7 und 5.25 V), die anderen beiden Pins sind SDA und SCL und werden dementsprechend direkt mit dem µC verbunden. Die I2C-Adresse des Moduls ist H30.

Der Sensor löst die Messung für die x- und y-Dimension jeweils mit zwölf Bit auf. Dementsprechend ist ein Messwert auf zwei Bytes verteilt. Das erste gelesene Byte ist das Kontrollregister, also beginnen die Messdaten ab dem zweiten Byte. Das zweite gelesene Byte ist high-Byte des x-Wertes, das dritte gelesene Byte dessen low-Byte. Das vierte Byte ist high-Byte des y-Wertes und das fünfte Byte low-Byte des y-Wertes. Das ungenutzte obere Nibble der high-Bytes soll man laut Datenblatt maskieren. Man braucht jetzt nur noch jeweils high- und low-Byte verrechnen, um an den entsprechenden Datenpunkt zu kommen.



Adresse	Inhalt des Leseregisters
0x00	Statusregister
0x01	MSB X-Messwert
0x02	LSB X-Messwert
0x03	MSB Y-Messwert
0x04	LSB Y-Messwert

Statusregister

0 TM (TakeMeasurements)

1 SET (Set Coil)

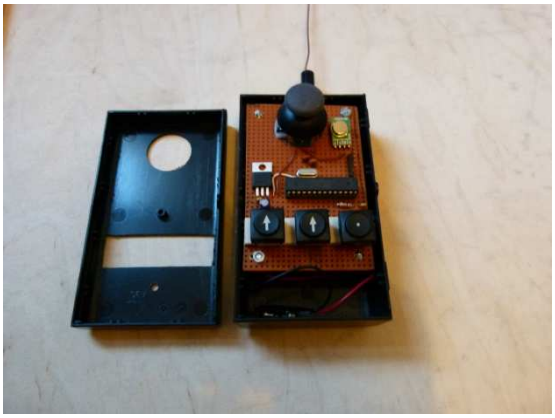
2 RESET (Reset Coil)

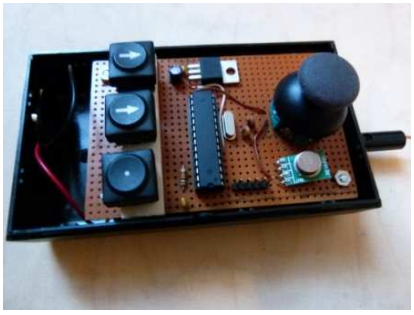
Beispielprogramm Ausgabe in Grad auf dem LCD

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#define  I2ADDR       0x30
#define  TakeMeasure  0x01
LiquidCrystal_I2C lcd(0x20,16,2);   //Adresse 0x20 für I2C
void setup(){
Wire.begin(30);
 lcd.setBacklight(HIGH);
 lcd.init();
 lcd.setCursor(0,0);lcd.print("Kompass");
}
void loop(){
byte MsbX,LsbX,MsbY,LsbY;
int x,y;
char line[80];
Wire.beginTransmission(I2ADDR); // Pollin HDMM01
Wire.write(byte(0));   // Adressfeld ist hier nicht wichtig
Wire.write(byte(TakeMeasure));
Wire.endTransmission();
delay(20);
Wire.beginTransmission(I2ADDR);
Wire.write(byte(0x01));
Wire.requestFrom(I2ADDR, 4);


while(Wire.available()<4);
MsbX  =Wire.read();         // obere  4 Bit X
LsbX  =Wire.read();         // untere 8 Bit X
MsbY  =Wire.read();         // obere  4 Bit Y
LsbY  =Wire.read();         // untere 8 Bit Y
Wire.endTransmission();         // stop transmitting
x=((MsbX&0x0f)*256)+(LsbX);
y=((MsbY&0x0f)*256)+(LsbY);
x = map(x, 1900, 2188, -180, 180);
y = map(y, 1910, 2193, -180, 180);
double mygrad = atan2(x, y)*180/3.1415927410;
if (mygrad < 0)    mygrad = mygrad +360;
// Ausgabe von X, Y und Grad

lcd.setCursor(0,1);
lcd.print(mygrad);lcd.print("  Grad");
delay(200);
lcd.print("              ");
}
```

Der Fernsteuersender

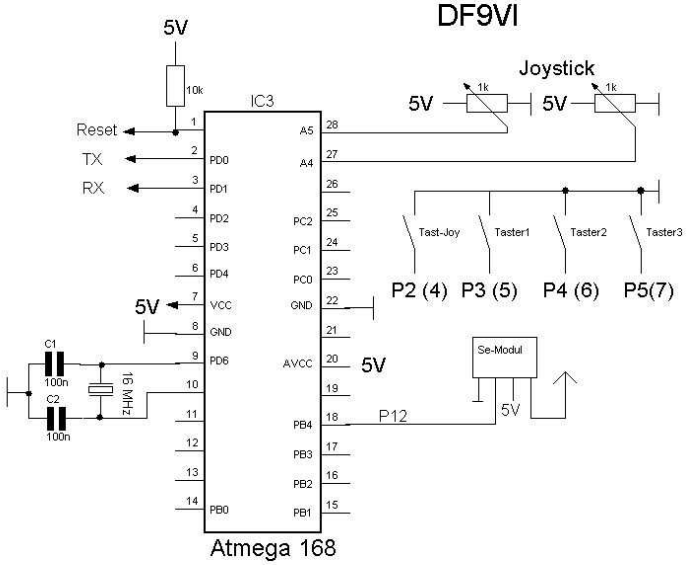
Hiermit läßt der Roboter sich fernsteuern. Das Modul arbeitet auf 433 MHz. Über den Joystick ist vor, zurück, rechts und links möglich. Beim Drücken ertönt die Hupe. 3 Taster ermöglichen weitere Funktionen, z.B. Licht ein und aus usw. Der Sender wird auf Lochraster aufgebaut, eine Leiterplattenversion kommt
eventuell später.

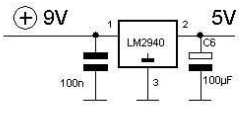


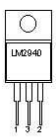


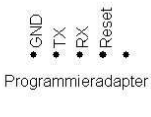
Fernsteuersender

Deutscher Amateur Radio Club e.V.
DF9VI







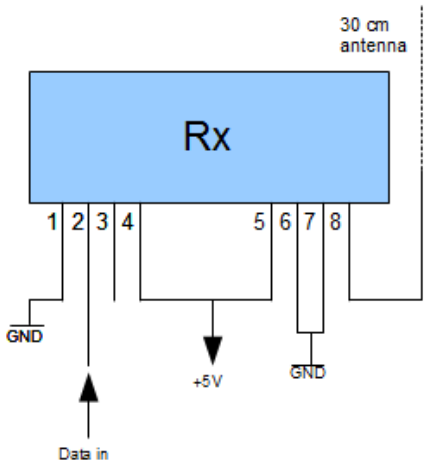


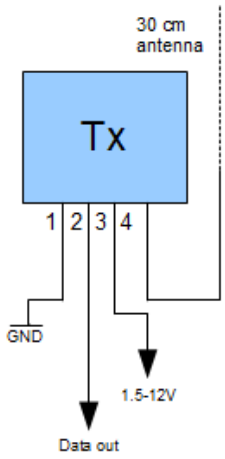
Der Fernsteuerempfänger

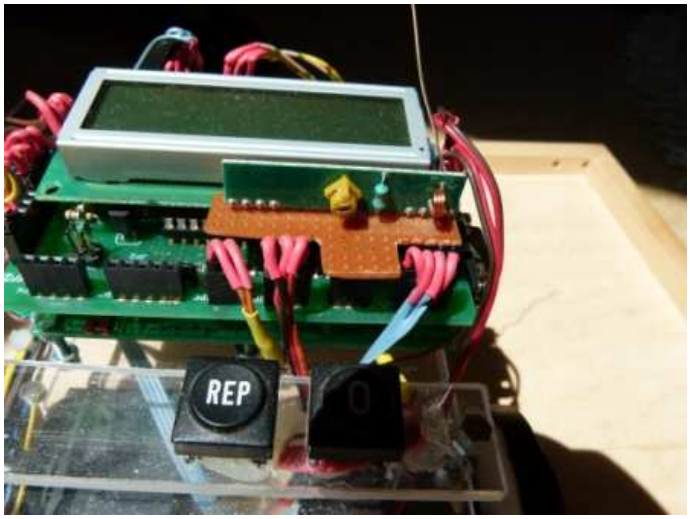
RF-Link Sender und Empfänger von Watterott

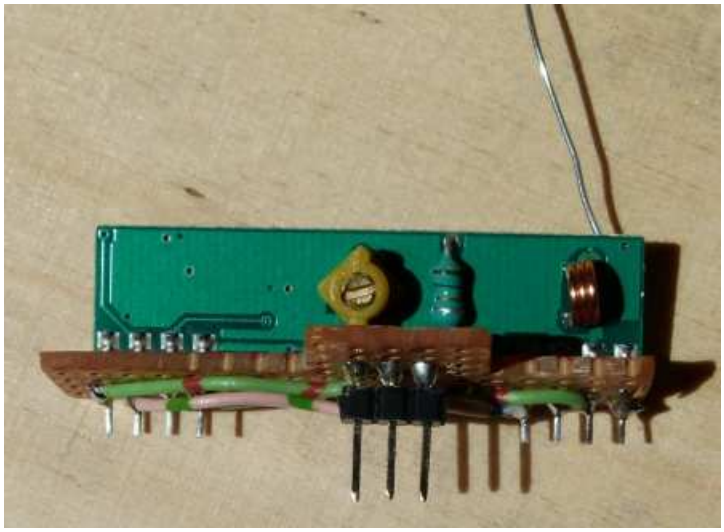
Library: http://www.open.com.au/mikem/arduino/

(VirtualWire)



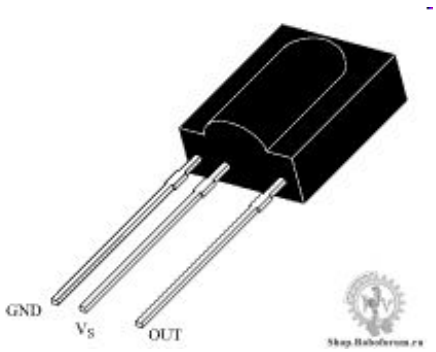




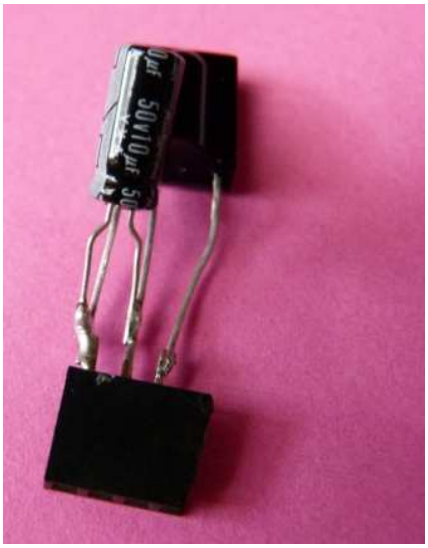


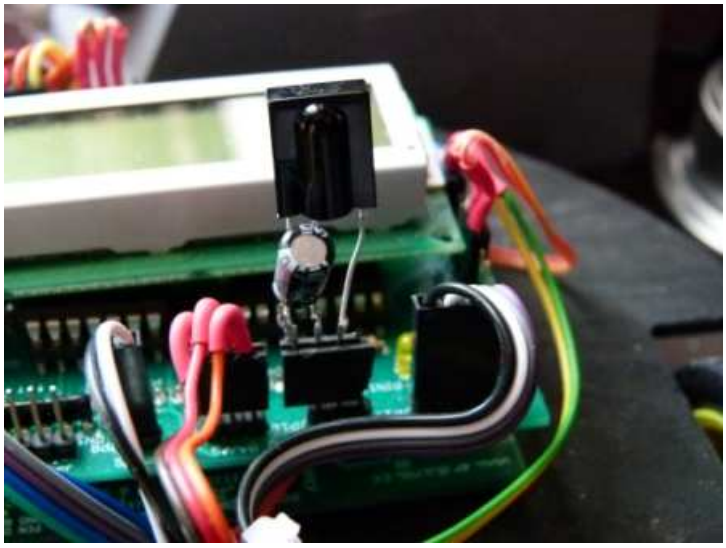
Infrarot - Fernsteuerung

Den Robby mit einer Universal-Fernbedienung fernsteuern! Dazu benötigen wir eine passende Fernsteuerung. Ausprobieren, nicht alle funktionieren! Als Empfänger fungiert der Baustein
TSOP 1136. Er beinhaltet eine Infrarot-Empfangsdiode und e einen passenden Verstärker. Der Baustein wird ganz einfach mit einem Pin des Arduino verbunden. Benötigt wird noch die Library IrRemote.h .



Zwischen die Stromversogungspins des TSOP 1136 muss ein Kondensator von ca. 10uF geschaltet werden, um Störsignale, durch die Motoren verursacht, zu blockieren. Out wird mit Pin 8 des Arduino verbunden. Das Modul wird wie auf den Bildern zu sehen ist, auf den Anschluss Servo 1 gesteckt.
Der Empfänger ist sehr richtungsempfindlich, also immer von vorn (wo die „Beule“ ist):





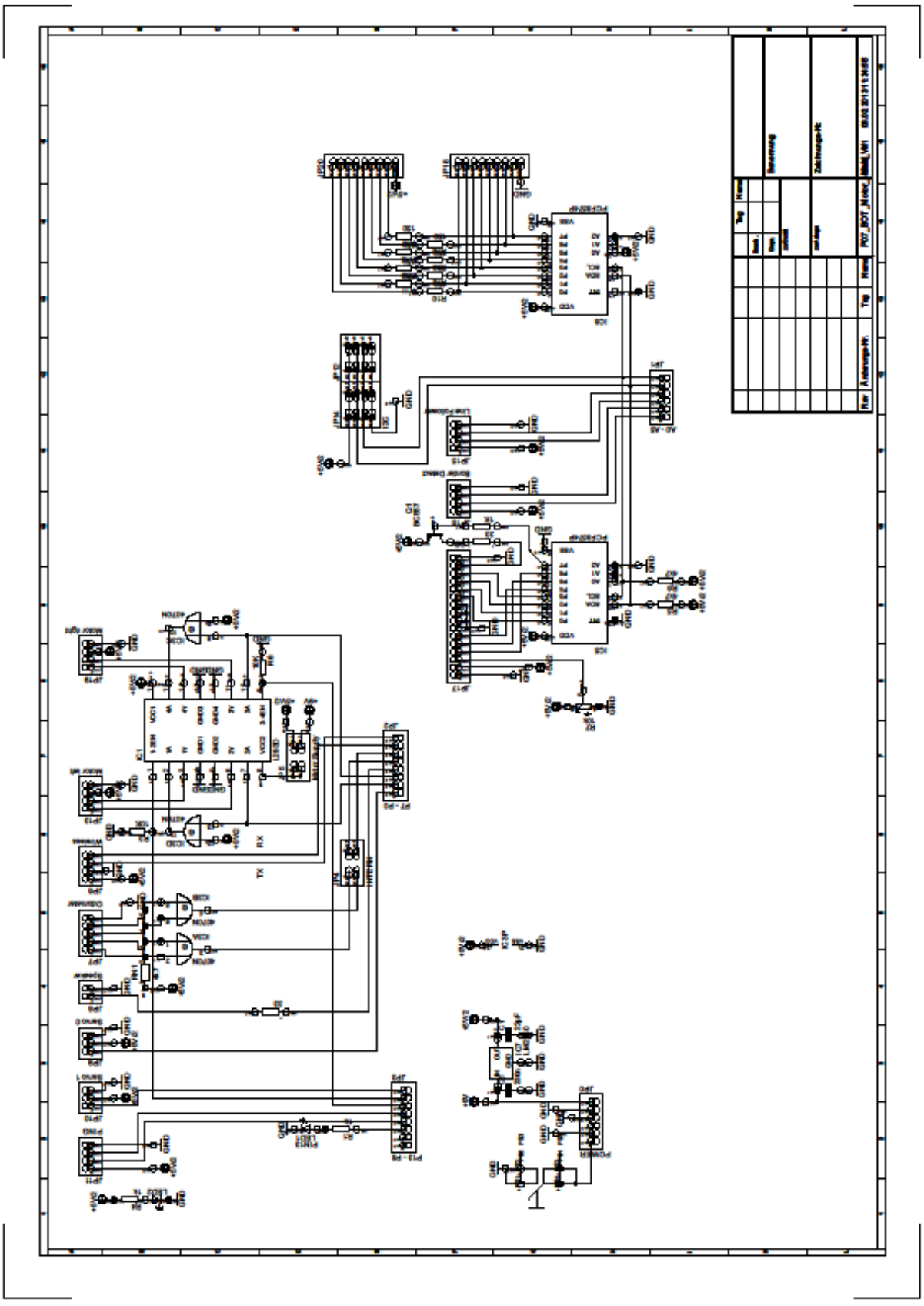
Beispielprogramm

```
#include <IRremote.h>    //Benötigte Library
int RECV_PIN = 8;
IRrecv irrecv(RECV_PIN);
decode_results results;
void setup()
{
  Serial.begin(9600);
  irrecv.enableIRIn();
}
void loop()
{
  if (irrecv.decode(&results))
     {
     Serial.println(results.value, HEX);
     irrecv.resume();                    // Receive the next value
     }
}
```

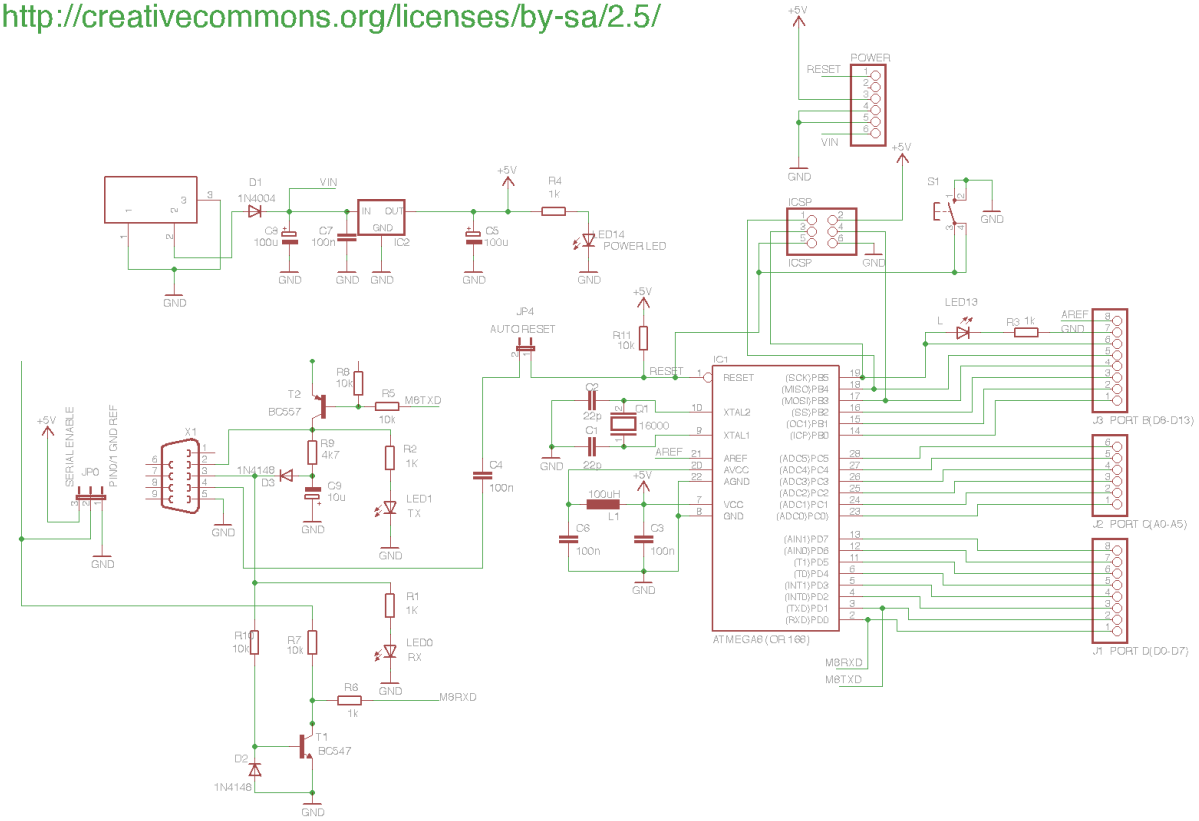
So kann man Robby steuern

```
void loop()
{

if (irrecv.decode(&results))
 {
   byte val = (results.value);
   irrecv.resume();                // Receive the next value
     switch(val)
        {
         case 2:
                  Forward(255);
                  break;
usw….
```

Schaltplan

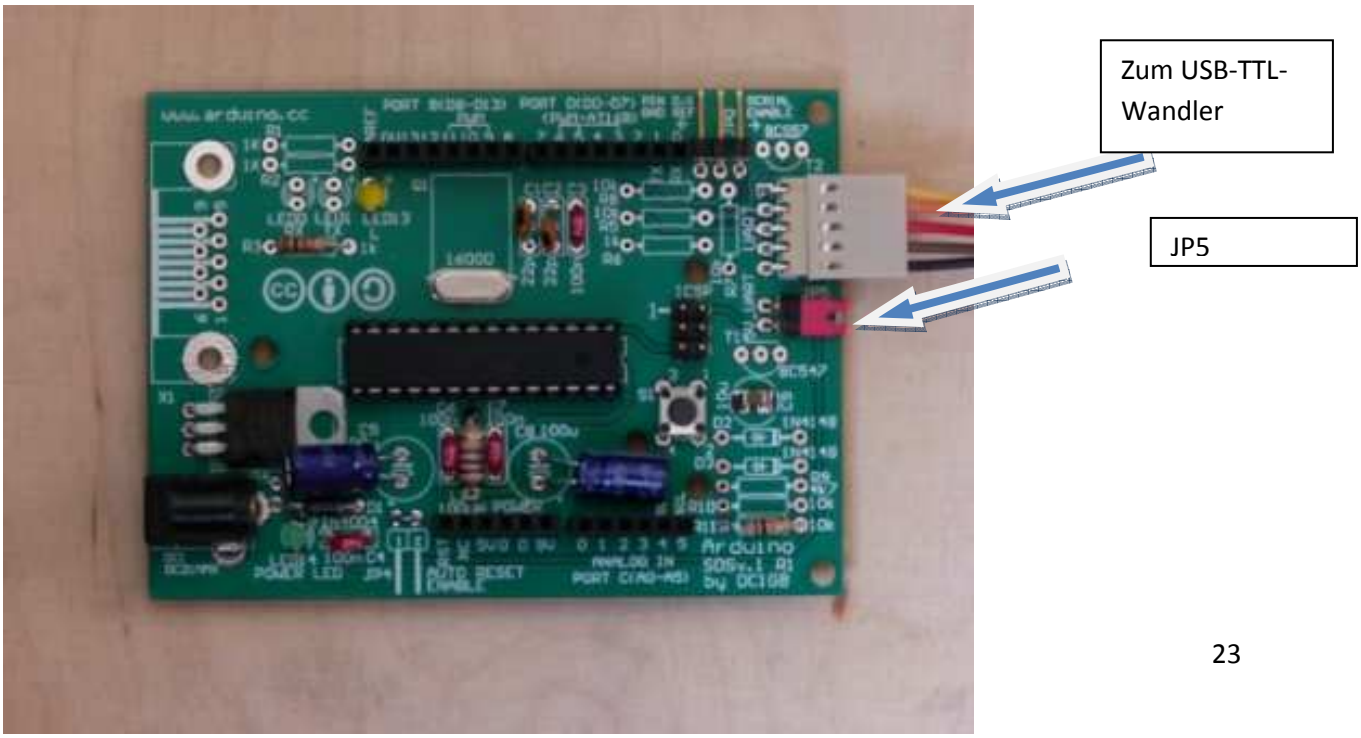


08.03.2013 21:46:36 f=0.66 C:\Users\Axel-\Desktop\eagle-Ardubot\Robby\P07_BOT_Motor_shield_V1.sch (Sheet: 1/1)



Modifizierte Version des Arduino-Single-Sided

Zugefügt wurde eine 5-poliger Stecker zum direkten Anschließen eines USB auf TTL-Wandlers. Die Leiterplatte muss bei Verwendung dieser Variante nicht vollständig bestückt werden. Über den Jumper JP5 kann der Arduino vom PC aus mit Strom versorgt werden



USB-TTL-Wandler

Je nach Ausführung muss eine Modifikation vorgenommen werden. Der Arduino benötigt die Signale RX, TX und DTR. Die Treiber lassen sich problemlos auf allen Windows-Versionen installieren

Wir programmieren mittlerweile alle Module über diese Wandler. Dazu wird ein 5-poliger Stecker benötigt.