



Laufroboter **YETI ARDUINO**

BAUANLEITUNG: Modell YT-5000



© AREXX - DIE NIEDERLANDE V0915

Inhaltsverzeichnis

1.	Produktbeschreibung YETI	5
2.	YETI Allgemeine Info	6
3.	ARDUINO Allgemeine Info	15
4.	Wie laufen wir eigentlich?	20
5.	Hardware	22
6.	Aufbau Elektronik	17
7.	Teileliste Mechanik	33
7.	Bauanleitung für die Mechanikteile	36
8.	YETI Akkus aufladen	48
9.	Software	49
10.	Installation der Software	52
11.	Inbetriebnahme und Test	56
12.	Kalibrierung des YETIs	57
13.	YETI Programmieren	60
14.	Erweiterungen	71
xx.	APPENDIX	93
	A. Übersicht der YETI-Funktionen	94
	B. Schaltbild YETI	97
	C. Schaltbild Display Module	98
	D. Schaltbild US Module	99
	F. Schaltbild USB IR	100
	G. Flachband Anschlußbelegung	101
	H. Fehlersuche	102
	I. Montage Erweiterungskits	104
	J. Tastatur-Kontrollmodus und Testprogramm	105
	K. ADC Messwert Akkusspannung	106

AREXX und YETI sind registrierte Warenzeichen von AREXX Engineering - HOLLAND.

© Deutsche Übersetzung/German translation (September 2015): AREXX Engineering (NL).

Diese Beschreibung ist urheberrechtlich geschützt. Der Inhalt darf auch nicht teilweise kopiert oder übernommen werden ohne schriftlicher Zustimmung des europäischen Importeurs:

AREXX Engineering - Zwolle (NL).

Hersteller und Vertreiber sind nicht haftbar oder verantwortlich für die Folgen unsachgemäßer Behandlung, Einbaufehler und oder Bedienung dieses Produkts bei Mißachtung der Bauanleitung.
Der Inhalt dieser Gebrauchsanleitung kann ohne vorheriger Ankündigung unsererseits geändert werden.



Fabrikant:
AREXX Engineering
JAMA Oriental



Europäischer Importeur:
AREXX Engineering
ZWOLLE Die Niederlande

Technische Unterstützung beim Bauen
des Roboters:

WWW.AREXX.COM
WWW.ROBOTERNETZ.DE

© AREXX Holland und JAMA Taiwan

© Deutsche Übersetzung: AREXX - Die Niederlande

Impressum

©2007 AREXX Engineering

Nervistraat 16
8013 RS Zwolle
The Netherlands

Tel.: +31 (0) 38 454 2028

Fax.: +31 (0) 38 452 4482

E-Mail: Info@arexx.nl

Diese Bedienungsanleitung ist urheberrechtlich geschützt. Der Inhalt darf ohne vorherige schriftliche Zustimmung des Herausgebers auch nicht teilweise kopiert oder übernommen werden! Änderungen an Produktspezifikationen und Lieferumfang vorbehalten. Der Inhalt dieser Bedienungsanleitung kann jederzeit ohne vorherige Ankündigung geändert werden. Neue Versionen dieser Anleitung erhalten Sie kostenlos auf <http://www.arexx.com/>

"YETI LAUFROBOTER" sind eingetragenes Warenzeichen von AREXX Engineering. Alle anderen Warenzeichen stehen im Besitz ihrer jeweiligen Eigentümer. Wir sind nicht verantwortlich für den Inhalt von externen Webseiten, auf die in dieser Anleitung verlinkt wird!

Hinweise zur beschränkten Garantie und Haftung

Die Gewährleistung von AREXX Engineering beschränkt sich auf Austausch oder Reparatur des Roboters und seines Zubehörs innerhalb der gesetzlichen Gewährleistungsfrist bei nachweislichen Produktionsfehlern, wie mechanischer Beschädigung und fehlender oder falscher Bestückung elektronischer Bauteile, ausgenommen aller über Steckverbinder/Sockel angeschlossenen Komponenten. Es besteht keine Haftbarkeit für Schäden, die unmittelbar durch, oder in Folge der Anwendung des Roboters entstehen. Unberührt davon bleiben Ansprüche, die auf unabdingbaren gesetzlichen Vorschriften zur Produkthaftung beruhen. Sobald Sie irreversible Veränderungen (z.B. Anlöten von weiteren Bauteilen, Bohren von Löchern etc.) am Roboter oder seinem Zubehör vornehmen oder der Roboter Schaden infolge von Nichtbeachtung dieser Anleitung nimmt, erlischt jeglicher Garantieanspruch!

The warranty is not applicable in case of disrespect of this manual! In addition, AREXX Engineering is not responsible for damages of all kinds resulting from the disrespect of this manual! Please adhere above all to the „Safety recommendations“.

Es kann nicht garantiert werden, dass die mitgelieferte Software individuellen Ansprüchen genügt oder komplett unterbrechungs- und fehlerfrei arbeiten kann. Weiterhin ist die Software beliebig veränderbar und wird vom Anwender in das Gerät geladen. Daher trägt der Anwender das gesamte Risiko bezüglich der Qualität und der Leistungsfähigkeit des Gerätes inklusive aller Software. Bitte beachten Sie auch die entsprechenden Lizenzvereinbarungen auf der CD-ROM!

Wichtig

Vor dem ersten Gebrauch dieses Robot Arms lesen Sie bitte zuerst die Gebrauchsanleitung aufmerksam durch! Sie erklärt die richtige Handhabung und informiert Sie über mögliche Gefahren. Zudem enthält sie wichtige Informationen, die nicht allen Benutzern bekannt sein dürften.

Symbole

Im Handbuch werden folgende Symbole verwendet:

	<p><i>Das "Achtung!" Symbol weist auf besonders wichtige Abschnitte hin, die sorgfältig beachtet werden müssen. Wenn Sie hier Fehler machen, könnte dies ggf. zur Zerstörung des Roboters oder seines Zubehörs führen und sogar Ihre eigene oder die Gesundheit anderer gefährden!</i></p>
	<p><i>Das "Information" Symbol weist auf Abschnitte hin, die nützliche Tipps und Tricks oder Hintergrundinformationen enthalten. Hier ist es nicht immer essentiell alles zu verstehen, aber meist sehr nützlich.</i></p>

Sicherheitshinweise

- Sobald die Verpackungen mit Teilen geöffnet sind, erlischt das Recht auf Rückkehr.
- Lesen Sie vor dem Bauen zuerst die Gebrauchsanleitung aufmerksam durch.
- Seien Sie beim Hantieren mit den Werkzeugen vorsichtig.
- Arbeiten Sie nicht im Beisein kleiner Kinder. Die Kinder können sich an den Werkzeugen verletzen oder kleine Komponenten und Teile in den Mund stecken.
- Prüfen Sie die Polung der Batterien und/oder Netzadapter.
- Halten Sie die Elektronik stets trocken. Wenn das Gerät einmal nass geworden ist, entfernen Sie sofort die Batterien oder die Stromversorgung.
- Bei längerem Nichtgebrauch die Batterien entfernen bzw. die Stromversorgung trennen.
- Wenn Sie meinen, dass das Gerät nicht länger sicher betrieben werden kann, trennen Sie es sofort von der Stromversorgung und stellen Sie sicher, dass es nicht unsichtlich durch kleiner Kinder benutzt werden kann.
- Das Modul besitzt hochempfindliche Bauteile. Elektronische Bauteile sind sehr gegen statische Elektrizität empfindlich. Fassen Sie das Modul nur an den Rändern an und vermeiden Sie direkten Kontakt mit den Bauteilen auf der Platine. Kinder unter 14 Jahre sollen den Roboter nur mit Hilfe eines Erwachsenen bauen.

Normaler Gebrauch

Dieses Produkt wurde als Experimentierplattform für alle an Robotik interessierten Personen entwickelt. Das Hauptziel besteht darin zu lernen, wie man das Gerät in ARDUINO und C++ Sprache programmieren kann. Das Gerät ist kein Spielzeug! Es eignet sich nicht für Kinder unter 14 Jahren.

Das Gerät ist nur für Innengebrauch bestimmt. Es darf weder feucht noch nass werden. Bitte achten Sie auch auf Kondenswasser, das sich bei einem Wechsel von einem kalten in einen warmen Raum entwickeln kann. Warten Sie eine Weile, bis sich das Gerät an die neuen Umgebungsbedingungen angepasst hat, bevor Sie es in Betrieb nehmen.

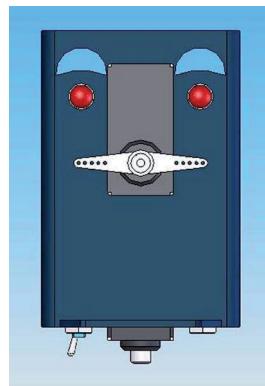
Jede andere Einsatzart als oben beschrieben kann zu Schäden und Risiken wie Kurzschluss, Brand, Stromschlag usw. führen .

1. PRODUKTBESCHREIBUNG YETI

1.1. Zu welcher Roboterfamilie gehört YETI?

YETI ist ein gehender Roboter. Sein Name stammt vom "Schnee-Menschen", einem vermeintlich im Himalaja lebenden, behaarten Riesen. Genauso wie dieses legendäre Ungeheuer bewegt sich unser YETI auf zwei riesigen Füßen, aber darin erschöpft sich auch schon die Ähnlichkeit dieser Namensvetter.

Der Antrieb für die Beine und Füße unseres YETIs erfolgt durch Servomotoren, und wird gesteuert von einem Mikroprozessor. Dieser führt ein Computerprogramm aus, das wir zuvor im Gehirn des YETI-Roboters gespeichert haben.



1.2. Spezifikationen:

Motoren	2 Servomotoren (5 Volt)
Prozessortyp	ATmega328P
Programmiersprache	ARDUINO und C++ (*Wiring)
Spannung	4 St. AA Akku
	4,8 - 6 Volt
Strom	Min. 10 mA
	Max. 600 mA
Kommunikation	Infrarot und I2C Bus
Erweiterung	Erweiterungen möglich mittels Flachbandkabel und Stapelbau
Höhe	278 mm
Breite	155 mm
Tief	100 mm

*) Wiring ist eine dünne Überschicht oberhalb C++, der das Einfügen einer eingebetteten Software vereinfacht.(The Wiring Language: A thin layer on top of C++ which simplifies the process of writing embedded programs. - What is the difference between Wiring and C++? - Quora)

2. YETI ALLGEMEINE INFO

2.1. Wer oder was ist ein YETI?

Wie bereits am Anfang beschrieben stammt sein Name vom “Schneemann”, einem vermeintlich im Himalaja lebenden behaarten Riesen. Mit seinem Riesenkörper und seinen großen Füßen kann er sich nur ziemlich schwerfällig bewegen.

Unser YETI ist ein hochgewachsener Roboter, der ebenfalls auf großen Füßen läuft. Er kann vorwärts und rückwärts gehen und sich sogar links oder rechts herum drehen.

Bei jedem Schritt vorwärts oder rückwärts muß YETI sich zuerst auf einem Fuß abstützen und dann den anderen Fuß versetzen. Dazu benutzt er zwei Servomotoren. Ein Servomotor verwendet ein Getriebe und ist deshalb sehr kräftig. Außerdem enthält der Servomotor eine impuls gesteuerte, elektronische Regelung. Die Elektronik erlaubt dem Servo nur eine genau bestimmte Schwenkung durchzuführen.

Der YETI verwendet einen Servo an der Vorderseite und einen Servo an der Unterseite. Der Servo an der Vorderseite zieht die Füße zum Verschieben hoch und wird Fußservo genannt. Der Servo an der Unterseite versetzt die Beine (und damit auch die Füße) eins nach dem anderen und wird Beinservo genannt.

2.2. Was können wir mit dem YETI anfangen?

- Neue (Beispiel-) Programme in den YETI übertragen.
- Selbsterstellte Programme in den YETI übertragen.
- YETI mit gebrauchsfertigen Erweiterungsmodulen erweitern, so dass der YETI zum Beispiel Gegenständen ausweichen oder Entfernungen messen kann.
- YETI mit selbstgebauten Erweiterungsmodulen erweitern.
- YETI mittels Infrarotsignalen mit dem PC kommunizieren lassen.
- YETI mittels Infrarotsignalen aus dem PC oder aus der Fernbedienung eines Fernsehers steuern.
- YETI Melodien oder Geräusche produzieren lassen
- Seine "Augen"-LEDs ein- und ausschalten.
- Den Körper des YETIs erweitern, zum Beispiel um ein Display oder einen LED-Mund.
- Der Einbau selbsterstellter Erweiterungsmodulen
- YETI mittels verdrahteter oder Infrarot Fernsteuerung kommunizieren lassen mit Ihrem PC
- YETI mittels verdrahteter oder Infrarot Fernsteuerung aus ihrem PC oder Fernbedienung ihres Fernsehers fernsteuern.

2.3. YETI wird mit drei Handgriffen zum Leben erweckt:

1. Setzen Sie zuerst die Mechanik- und Elektronikmodule des YETIs mit Hilfe der Bauanleitung zusammen.
2. Laden Sie ggf. die Akkus auf.
3. Schalten Sie den YETI mit dem Hauptschalter auf der Unterseite des Geräts ein.

Nach einigen Sekunden wird YETI seine Beine und Füße strecken und anschließend (mittels eines Standard-Beispielprogramms im Prozessorhirn) ein Beispiel seines Könnens geben.

Nun, das war zunächst gar nicht schwierig, und es sieht aus, als ob man jetzt bereits fertig wäre.

Jetzt aber fängt die Arbeit erst an !

Wir werden uns jetzt mit dem Entwurf und Schreiben eigener Programme beschäftigen und können so in kreativer Weise das Verhalten unseres YETIs anpassen.

2.4. Übertragung eines (Beispiel-) Programms in den YETI

2.4.1. Übertragung mittels USB-IR-Transceiver

Um ein Beispielprogramm in den YETI-Speicher zu übertragen werden wir harmlose unsichtbare Infrarotlichtstrahlen benutzen. Das Infrarotlicht wird dazu im mitgelieferten Infrarot USB-Adapter erzeugt, der an Ihrem PC an einem USB-Port eingesteckt werden muss.

Ihr YETI-Roboter ist ausgestattet mit einem eingebauten Infrarotempfänger, der sich hinter den kleinen Öffnungen auf dem Rücken des Roboters befindet. Der ASURO-Roboter, ein weiterer Roboter aus unserer Roboterfamilie, verwendet übrigens das gleiche Infrarotübertragungssystem.

YETI muss Sichtkontakt (ca. 5cm bis 50cm Abstand zwischen IR-Transceiver-Stick und YETI, beide Bestückungsseiten zeigen zueinander und nichts ist im Lichtweg) zum IR-Transceiver haben. YETI wird jetzt eingeschaltet (S1 auf ON), und UPLOAD starten ins Arduino Software wann die Kontaktaufnahme nicht geklappt haben, einfach, erneut UPLOAD drücken.



2.4.2. USB-Infrarot-Tranceiver-Stick

Diese Inbetriebnahme gilt nur für den USB-Infrarot-Transceiver-Stick. Der USB-Transceiver-Stick wird mit dem USB-Kabel an einer freien USB-Buchse angeschlossen.

**Es erscheint die Meldung: "Neue Hardware gefunden:
AREXX ASURO USB-IR" Danach den USB-Treiber von der
YETI-CD installieren.**

TEST MIT TERMINAL PROGRAMM

Sollte der Treiber nicht automatisch gefunden werden, bitte die neue FTDI driver von www.arexx.com oder die FTDI Website Downloaden. Eventuell sind für Installieren Administratorrechte erforderlich. Dann abmelden und als Administrator erneut anmelden. Es wird nun ein Treiber installiert, damit man unter Windows den USB-Transceiver wie eine normale serielle Schnittstelle ansprechen kann.

Hat dies Fehlerfrei geklappt, so startet man zum Ausprobieren auch hier das Terminalprogramm (downloaden von Internet) z.B "HTerm", gibt ASURO USB für den Verbindungsnamen an und wählt ein beliebiges Symbol aus.

Beim nächsten Fenster "Verbinden über:" wählt man die letzte verfügbare COM-Schnittstelle aus. Nach Drücken auf "OK" wählt man:

- Bits pro Sekunde: 2400
- Datenbits: 8
- Parität: keine
- Stoppbits: 1
- Flussteuerung: kein

Danach wieder bestätigen mit "OK".



Im Gegensatz zum bisherigen RS232-IR-Transceiver und zum USB-IR-Transceiver V1.0 unterdrückt der neue USB-IR-Transceiver sein eigenes Echo. **Dies bietet deutliche Vorteile bei der Datenübertragung, allerdings funktioniert der Selbsttest über Reflexion im Hyperterminal damit nicht mehr.**

Die korrekte Installation des Treibers kann aber dadurch verifiziert werden, **dass die grüne Sendeleuchtdiode bei der Datenübertragung oder beim Tippen im Hyperterminal flackert**. Der USB-IR-Transceiver wurde werkseitig auf Funktion überprüft, ein Defekt ist daher sehr unwahrscheinlich.

Sollte trotzdem ein Test gewünscht sein, eignet sich das Selbsttestprogramm von YETI, das bei Sichtverbindung zum USB-IR-Transceiver gestartet wird. Die Ausgaben des Selbsttests können dann im Terminalprogramm beobachtet werden. Erreicht der Selbsttest den Test des IR-Transceivers (LED am YETI flackert gelblich) können vom Terminalprogramm aus auch Zeichen gesendet werden, die dann vom Roboter aus mit dem im Alphabet folgenden Zeichen beantwortet werden."

Beim Senden blinkt die gelbe LED und beim Empfang leuchtet die grüne LED. Der Transceiver liest sein eigenes Echo nicht und daher brauchen Sie den YETI, um diesen Test während des Selbsttests durchzuführen.

USB-Infrarot-Transceiver funktioniert nicht

Ist der Treiber ordnungsgemäß installiert? Teilweise werden andere COM-Port-Nummern zugewiesen als die letzte. Hier hilft im Hyperterminal auch mal andere Ports einzustellen und den Test zu wiederholen. Ggf. kann man auch in der Systemsteuerung nachsehen, welcher Port zugewiesen wurde.

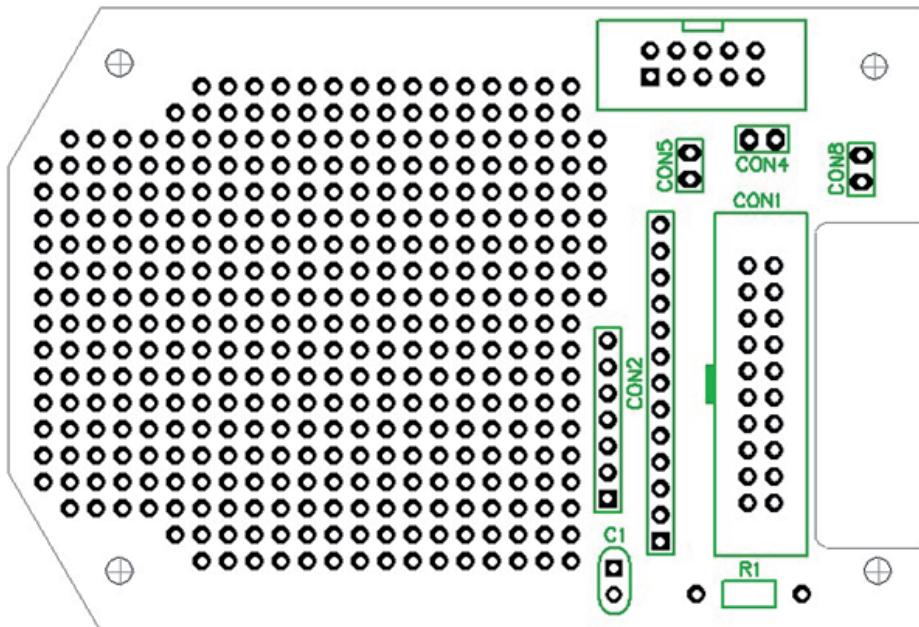
2.4.3. Übertragung mittels USB-Kabel

Auch mit einem Erweiterungsmodul und USB-Kabel kann man Programme in den YETI Roboter übertragen. Der Transfer wird sogar wesentlich schneller ablaufen.

Die Übertragung eines Programms in den YETI-Speicher wird das existierende Programm im YETI-Speicher überschreiben. Das im YETI-Speicher vorhandene Beispielprogramm wird dabei gelöscht. Das ist jedoch kein Problem, da wir dieses Programm jederzeit wieder aus dem PC in den Roboterspeicher übertragen können.

YETI Programmierung-Erweiterungsmodul YT-PRG

Die YT-PRG ist ein Optionelle erweiterungs Satz für die YETI ARDUINO Laufroboter. Siehe Kapitel 14.11



2.5. Laden eines Programms in den YETI

2.5.1. Kabelgebundene Datenübertragung

- * Verbinde den USB Adapter zum PC.
- * Starte die Arduino IDE (Entwicklungsumgebung)²⁾
- * Wähle im Arduino Programm am PC den COM-port zum USB-port Adapter.
- * Wähle im Arduino IDE ein Yeti Programm, im Menübereich File -> sketchbook
- * Verbinde die verkabelte Erweiterungsplatine mit dem YETI indem Sie es entweder auf dem Display oder am Flachbandkabel einstecken (falls die Platine bereits angeschlossen war, muss die Verbindung nicht jedes Mal für diesen Schritt gelöst werden).
- * Verbinde mit dem kleineren, aber längeren, 10-Pins-Flachbandkabel den USB-Adapter mit dem YETI.
- * Schalte den YETI ein.
- * Klicke den "Upload" Knopf auf dem Arduino IDE.
- * Das Programm wird jetzt kompiliert und in den YETI-Speicher übertragen.
- * Warte bis die Arduino IDE den Datentransfer abgeschlossen hat.
- * Der YETI wird 5 Sekunden nach Übertragungsende das Programm starten.

2.5.2. Drahtlose Datenübertragung

- * Verbinde den USB Adapter zum PC.
- * Starte die Arduino IDE (Entwicklungsumgebung)
- * Wähle im Arduino Programm am PC den COM-port zum USB-port Adapter.
- * Wähle im Arduino IDE ein Yeti Programm, im Menübereich File -> sketchbook Sehe zu, dass die Öffnungen am YETI's Rücken Sicht kontakt zur Oberseite des COM-Port Adapters haben.
- * Wähle im Arduino IDE ein Yeti Programm, im Menübereich File -> sketchbook
- * Sehe zu, dass die Öffnungen am YETI's Rücken Sichtkontakt zur Oberseite des COM-Port Adapters haben.
- * Schalte den YETI-Hauptschalter aus.
- * Klicke den "Upload" Knopf auf dem Arduino IDE.

²⁾ IDE = Integrierte Entwicklungsumgebung (IDE = Abkürzung für Integrated Development Environment)

- * Schalte den YETI-Hauptschalter ein (falls die Kompilierung sehr lange dauert, versuche dann den Transfer nochmals zu starten, wobei sie den YETI einschalten sobald die Statuszeile “uploading” meldet).
- * Das Programm wird jetzt in den YETI-Speicher übertragen.
- * Warte bis das YETI Programm übertragen worden ist.
- * Der YETI wird 5 Sekunden nach Übertragungsende das Programm starten.
- * Falls ein Prozess hängenbleibt oder ein Fehler meldet, beachten Sie bitte die Information am Ende dieser Seite.

Falls alles in Ordnung ist, wird der YETI nun das neu-geladene Programm starten.

2.7. Die Kommunikation zwischen dem YETI und ihrem PC

Nach dem Anklicken der “Upload”-Taste in der Arduino IDE Entwicklungs-umgebung wird die Arduino Software versuchen mit dem YETI Roboter Kontakt aufzunehmen. Falls der YETI antwortet wird das ausgewählte YETI-Programm übertragen. Der YETI wird jedoch nur antworten falls die Kontaktaufnahme innerhalb dem 5 Sekunden Intervall nach dem Einschalten des YETIs versucht wird. Falls der YETI Roboter nicht innerhalb dieser 5 Sekundenperiode kontaktiert werden kann wird der Roboter fortfahren mit dem Start des im Roboterspeicher vorhandenen Programms.

Falls YETI nicht mit einer Antwort reagiert wird das Arduino Programm eine Fehlermeldung erzeugen. Aus ASURO-Benutzerkreisen stammen Berichte über möglichen Problemfällen im Datenübertragungsbereich. Solche Probleme können vermieden werden indem man folgendes sicherstellt:

- einen guten Sichtkontakt zwischen Infrarot-Sender und Infrarot-Empfänger.
- die Benutzung der aktuellsten Flash Software Version.
- die Benutzung von gut gefüllten und intakten Akkus / Batterien.
- die Abschirmung der Systemen von Kunstlichtquellen (speziell Leuchtstoffröhren und Sparlampen) und Sonneneinstrahlung.

2.6. Erweiterungsmodule (Kits)

Sie können YETI mit zusätzlichen (nicht im Lieferumfang enthaltenen) Baugruppen erweitern, welche die Leistungsfähigkeit des Roboters erheblich steigern. So können Sie YETI mit einem Ultraschall-Sender/Empfänger ausstatten, der ihm ermöglicht mit Schallwellenechos die Distanz zu entfernten Gegenstände zu messen und diesen auszuweichen.

Auch können Sie YETI mit einem Display ausstatten, um darauf Daten oder Nachrichten anzeigen. Ein Erweiterungsmodul besteht aus einer kleinen Leiterplatte, die wahlweise mit oder ohne Bauelementen geliefert wird. Die Erweiterungsmodule passen genau in YETIs Kopf und werden sozusagen in seiner Schädeldecke genau oberhalb der Augen festgeschraubt.

Sie können aber auch Ihre eigenen Module mit Experimentierleiterplatten entwerfen. Diese werden auf YETIs Kopf anstelle des kleinen Vordachs platziert.

Mit einem Flachbandkabel verbinden Sie die Erweiterungsmodule über eine I2C-Bus-Schnittstelle mit der Hauptplatine (und somit automatisch auch mit dem I2C Bus des Mikroprozessors im YETIs Kopf).

3. ARDUINO ALLGEMEINE INFO

3.1. Wer oder was ist ARDUINO?

Arduino ist ein open source-Einplatinen Mikrocontroller, der insbesondere Künstlern, Designern, Bastlern und anderen Interessierten den Zugang zur Programmierung und zu Mikrocontrollern und die Projektarbeit an interaktiven Objekten erleichtern soll.

Die Arduino-Plattform basiert auf einer Atmega 168 oder Atmega 328 Mikrocontroller von Atmel. Das System stellt dem Anwender sowohl digitale Ein- und Ausgänge als analoge Eingänge zur Verfügung. Damit kann das Arduino-System aus der Umgebung Signale empfangen und anschließend darauf reagieren.

Es werden verschiedene Arduino-Platten auf dem Markt angeboten, wie zum Beispiel Arduino Uno, Arduino LilyPad und Arduino Mega 2560. Da jede Arduino-Platine individuell seine spezifischen Eigenschaften aufweist kann man für wohl jedes Projekt die ideale Arduino-Baugruppe auswählen.

Eingangssignale können zum Beispiel durch Schaltern, Lichtsensoren, Bewegungssensoren, Abstandssensoren und Temperatursensoren geliefert werden. Auch kann man Kommandos aus dem Internetbereich als Eingangssignale zuliefern. Ausgangssignale wiederum können Motoren, Lämpchen, Pumpen und Bildschirme ansteuern.

Zur Programmierung verfügt das System über einem Compiler für eine standardisierte Programmiersprache und einem Bootloader. Die Programmiersprache basiert auf die Wiring-Programmiersprache, die mit C++ übereinstimmt.

Arduino wurde zunächst in 2005 als Projekt gestartet in Ivrea, Italien. Erklärtes Ziel war ursprünglich die Idee Studenten in Projektarbeiten zu unterstützen, wobei die Prototyp-Erstellung deutlich preiswerter sein sollte als bei vergleichbaren herkömmlichen Methoden. Die Entwicklergruppe um Massimo Banzi und David Cuartielles benannte das Projekt nach einer historischen Gestalt mit dem Namen 'Arduin von Ivrea'. Das Wort 'Arduino' bedeutet 'Kräftiger Freund'.

3.2 Mikrocontroller!

3.2.1. Anwendungen

Ein Mikrocontroller (manchmal in der verkürzten Schreibweise auch µC, uC oder MCU genannt) ist ein kleiner Computer in einer integrierten Einzelschaltung, der den Prozessorkern, Speicher und einen Satz programmierbarer Ein-/Ausgangsanschlüsse enthält.

Programmspeicher und ein kleiner Datenspeicher, RAM (beziehungsweise Random Access Memory) gehören oft ebenfalls zur Ausstattung des Chips. Mikrocontroller werden in automatisch gesteuerten Anlagen und Systemen eingesetzt, wie zum Beispiel in der Motorensteuerung, Implantaten, Fernsteuerungen, Bürosystemen, Spielzeug und Hochleistungswerkzeugen.

Die Gewichtseinsparung und Kosteneinsparung durch Integration des Mikroprozessors, Speicher und Ein-/Ausgangsanschlüsse auf einem Einzelchip führt dazu dass die Digitalsteuerung für immer mehr Anwendungsbereiche noch wirtschaftlicher wird.

Ein typischer Haushalt in einer fortschrittlichen Wohngegend verfügt über vier allgemeinen Mikroprozessoren und drei Dutzend Mikrocontroller Schaltungen. Ein Durchschnittstyp eines Mittelklassen-PKW verwendet 30 oder mehr Mikrocontroller. Auch findet man diese Bauteile in vielen elektrischen Maschinen wie Waschmaschinen, Mikrowellenherden und Telefonen.

3.3. Leistungsverbrauch und Geschwindigkeit

Manche Mikrocontrollers arbeiten bei einer niedrigen Taktfrequenz von 4 kHz und weisen einen geringen Leistungsverbrauch im Milliwatt- oder Mikrowatt-Bereich auf. Sie können üblicherweise sofort auf einem Knopfdruck oder Interrupt-Prozess aktiviert werden. Der Leistungsverbrauch liegt beim Warten (bei abgeschaltetem CPU-Clockgenerator sowie fast der kompletten Zusatzbeschaltung) im Nanowattbereich, was eine Langlebigkeit der Batterien sicherstellt.

Andere Mikrocontroller werden eher im Hochleistungsbereich verwendet, wo man sie zum Beispiel als Digitale Signalprozessoren (DSP) mit höheren Taktraten und höherem Leistungsverbrauch einsetzt.

Das Arduino-System arbeitet mit einem leistungsfähigen Atmel 328P-AU Single-Chip, der mit einem 8-bit Mikrocontroller (getaktet mit 20MHz) und 32K Bytes In-System programmierbarem Flash-Speicher ausgestattet ist. Die Stromversorgung ist recht flexibel im Bereich von DC7-12V, gewählt worden, wodurch man auf stabilen und ordentlich abgesicherten Arbeitsbedingungen für den Chip und abgetrennten Leistungsleitungen bis 2A für die Motorenversorgung aufbauen kann.

3.4 Mikrocontroller Programme

Die Software für Mikrocontroller muss im auf dem Chip verfügbaren Speicherplatz passen, denn ein externer Zusatzspeicher wäre zu kostspielig. Die Compiler und Assembler sind optimiert für die Umsetzung der Hochsprache und Assemblercodes in kompakte Maschinenbefehle, die in den Speicher des Mikrocontrollers abgelegt werden.

Je nach Chihtyp kann das Programm in einem permanenten ROM³-Speicher, der nur während der Fertigung des Chips beschrieben werden kann, oder in einem immer wieder beschreibbaren Flash – oder mehrfach beschreibbarem ROM-Speicher gespeichert werden.

Ursprünglich hat man Mikrocontroller nur in Assemblersprache programmiert, aber zur Zeit sind auch verschiedene höheren Programmiersprachen für die Programmierung der Mikrocontroller verfügbar. Diese Sprachen sind entweder Spezialsprachen oder Varianten der allgemeinen Hochsprachen wie zum Beispiel C. Verkäufer der Mikrocontroller bieten oft gratis Werkzeuge an um die Implementierung der Hardware zu vereinfachen. Das Arduino-System stellt uns etwa 16 kBytes Flash-Speicher für die sogenannten Sketch-Programme zur Verfügung, die in C programmiert werden können.

3.5 Schnittstellenarchitektur

Mikrocontroller verfügen in der Regel über verschiedenen bis Dutzenden von frei programmierbaren Ein-/Ausgangspins (GPIO). Die GPIO-Anschlüsse sind mittels Softwarekommandos programmierbar als Eingangs- beziehungsweise Ausgangspins. Bei Programmierung als Eingangspins werden sie oft zum Ablesen der Sensoren oder externen Signalen verwendet. Programmiert man sie als Ausgangspins kann man sie zum Beispiel zur LED- oder Motoransteuerung verwenden.

Eine Vielzahl an eingebetteten Systemen benötigt die Ablesemöglichkeit der Analogsignalen von den Sensoren. Zu diesem Zweck werden Analog/Digitalwandler (A/D-Wandler oder in Englisch ADC⁴) eingesetzt.

Da die Prozessoren speziell zur Verarbeitung von Digitaldaten, das heißt Nullen und Einsen, entwickelt worden sind, können sie mit den Analogsignalen der Sensoren an sich nichts anfangen. Deshalb werden die A/D-Wandler eingesetzt um die eintreffenden Daten in eine für den Prozessor lesbaren Form zu verwandeln.

³ ROM = Read Only Memory (nur lesbarer Speicher)

⁴ Analog-to-digital converter (ADC)

Ein weniger üblichen Form der Datenverwandlung in Mikrocontroller ist die Digital zu Analog-Umwandlung (im D/A-Wandler oder DAC⁵) womit der Prozessor Analogsignale oder Spannungspegel erzeugen kann.

Zusätzlich zu den Konvertern verfügen einige eingebettete Systeme auch noch über Zeituhr-Schaltungen (Timers). Einer der gängigen Zeituhrtypen ist die Programmierbare Intervall-Zeituhr (PIT⁶), der von einem beliebigen Anfangswert herabzählt nach Null. Nachdem es die Null erreicht hat sendet die Uhr einen Interrupt an den Prozessor als Signal, dass die eingestellte Zeit abgelaufen ist. Das ist eine nützliche Funktion für zum Beispiel Thermostaten, die regelmäßig die Umgebungstemperatur registrieren und prüfen sollen, um - falls notwendig – die Klimaanlage beziehungsweise Heizung ein zu schalten.

Das universelle asynchrone UART⁷-Transceivermodul erlaubt uns ohne allzu viel Prozessoraufwand Kommunikationsdaten über einer seriellen Leitung mit anderen Systemen aus zu tauschen (das heißt zu versenden und zu empfangen).

Spezielle integrierte Hardware im Chip ermöglicht oft auch die Kommunikation mit anderen Chips in Digitalformaten wie I2C und Serial Peripheral Interface (SPI).

Das Arduinosystem stellt uns 14 digitale I/O-Anschlüsse und 7 analoge I/O-Anschlüsse zur Verfügung (I/O- das heisst: Eingang/Ausgangs-).

⁵ Digital-to-analog converter (DAC)

⁶ Programmable Interval Timer (PIT)

⁷ Universal Asynchronous Receiver/Transmitter (UART)

4. WIE LAUFEN WIR EIGENTLICH?

3.1. Zusammenfassung

YETI balanciert auf einem Fuß und versetzt anschließend den anderen Fuß. Dazu zieht YETI die Außenseite seines abstützenden Fußes hoch und drückt zur gleichen Zeit die Außenseite seines zu versetzenden Fußes herunter. Dabei neigt sich YETI zum abstützenden Fuß herüber und verlagert den Großteil seines Gewichts auf diesen abstützenden Fuß. Anschließend schiebt YETI seinen zu versetzenden Fuß nach vorne und schließt somit den Schritt ab. Der abstützende Fuß wird nun der zu versetzende Fuß und umgekehrt.

Dieser Bewegungsablauf muß sich beim Gehen natürlich ständig wiederholen.

3.2. Ausführliche Erläuterung

Sobald der YETI aus der Stillstandposition laufen soll, dreht er seinen Fußservo von vorne betrachtet im Uhrzeigersinn. Dabei passieren zwei Dinge gleichzeitig.

Zum einen hebt sich die rechte Seite des Fußservos und zieht die Außenseite des rechten Fußes ein wenig hoch. Er könnte seinen Körper nun nach rechts bewegen, tut das aber nicht, weil ein Großteil des Gewichts sich immer noch auf der Innenseite des rechten Beins befindet.

Zweitens senkt sich die linke Seite des Fußservos und drückt gleichzeitig die Außenseite des linken Fußes ein wenig herunter. Dabei wird das linke Bein hochgedrückt. Beide Bewegungen zusammen bewegen den Körper nach rechts und verlagern das Gewicht von den linken auf den rechten Fuß.

Falls YETI seinen rechten Fuß etwas zu hoch anheben würde, drückt der linke Fuß den Körper des YETIs über die Gleichgewichtsposition des rechten Fußes hinweg. Der Roboter verliert dann sein Gleichgewicht und würde nach Rechts umkippen. YETI soll deshalb seinen rechten Fuß gerade soweit hochheben, dass er, wenn sein linkes Bein ihn nach Rechts drückt, gerade nicht seinen rechten Gleichgewichtspunkt erreicht. Je dichter sich der Körper dem Gleichgewichtspunkt annähert, desto größer ist der Anteil des Gesamtgewichts auf dem rechten Fuß und desto leichter lässt sich der linke Fuß versetzen.

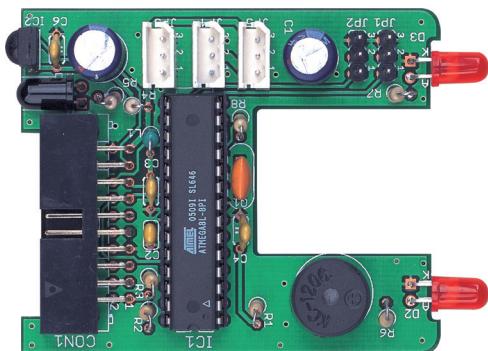
Jetzt müssen sich noch die Beinservos bewegen. Mit dem Beinservo versetzt YETI seinen linken Fuß möglichst weit nach vorne. Beim Versetzen des linken Fußes von hinten nach vorne bewegt sich auch der Körper bis zur halben Strecke des linken Fußes, aber immer noch in der Spur des rechten Fußes. Er stützt sich also immer noch überwiegend auf den rechten Fuß. Wenn sich nun der linke Fuß weit nach vorne gestreckt hat, zieht er die Außenseite seines linken Fußes hoch und die Außenseite seines rechten Fußes herunter. Dadurch verschiebt er seinen Körper, sein Körergewicht und damit auch den Schwerpunkt des Körpers vom rechten Fuß zum linken Fuß. Ab diesem Zeitpunkt ruht das Gewicht hauptsächlich auf den linken Fuß. Auch hier gilt die Regel, daß YETI nicht zu weit nach links balancieren darf, damit er nicht nach links umkippt.

Beim nächsten Schritt wiederholt sich die Prozedur mit dem gegenüberliegenden Fuß.

5. HARDWARE

5.1. YETI Basisleiterplatte

Die Hauptleiterplatte des YETIs enthält einen ATmega328-P Mikrocontroller Chip. Dieser Mikrocontroller ist verbunden mit dem großen, runden Piepser und mit den beiden roten LEDs an der Vorderseite des Roboters. Zahlreiche andere Mikrocontrolleranschlüsse, z. B. der I2C Bus, werden direkt zum 20-Pins Steckeranschluß auf der Rückseite weitergeleitet. Dieser Steckverbinder erlaubt eine Systemerweiterung mit weiterer Hardware über Flachbandkabel. Die beiden schwarzen Steckverbinder neben der roten LED versorgen den Anschluss der beiden Servos. Die drei weißen Steckverbinder versorgen den Ein/Aus-Schalter und die beiden Akkupacks.



5.2. Flachbandanschluß

Vier der zwanzig Flachbandadern sind für die Stromversorgung reserviert. Pin 19 für VCC und Pin 7, 8 und 20 für GND.

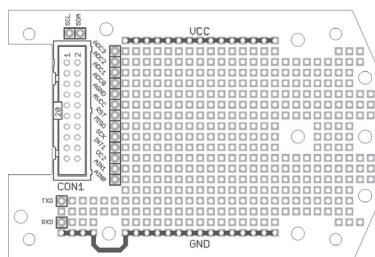
Alle übrigen Flachbandanschlüsse sind direkt mit festgelegten Anschlüssen des Mikrocontrollers verbunden. Direkt neben den Anschlußnummern sind die Mikrocontrollerpinfunktionen spezifiziert. Ein Mikrocontrollerpin kann je nach Anwenderprogramm auf verschiedenste Art verwendet werden. Überprüfen Sie bitte im Datenblatt bzw. Handbuch zum Mikrocontroller die genaue Definition der Pinfunktion.

5.3. Erweiterungsanschlussbelegung

Alle Erweiterungsmodule werden grundsätzlich mit dem gleichen 20-adrigen Flachbandkabel auf der YETI-Hauptplatine angeschlossen. Das Flachbandkabel enthält auch die Stromversorgungsanschlüsse der Erweiterungsmodule und den I2C Bus von und zu den Erweiterungsmodulen.

Info Flachbandanschluss: (Mehr Infos siehe Seite 15)

Pin 1 PC5 (ADC5/SCL)
Pin 2 PC4 (ADC4/SDA)
Pin 3 PC3(ADC3)
Pin 4 PC2(ADC2)
Pin 5 PC1(ADC1)
Pin 6 PC0(ADC0)
Pin 7 GND
Pin 8 GND
Pin 9 AVCC
Pin 10 PC6(RESET)
Pin 11 PB5(SCK)
Pin 12 PB4(MISO)
Pin 13 PB3(MOSI/OC2)
Pin 14 PD3(INT1)
Pin 15 PD6(AIN0)
Pin 16 D7(AIN1)
Pin 17 PD0(RXD)
Pin 18 PD1(TXD)
Pin 19 VCC
Pin 20 GND



5.4. YETI Experimentiermodul

Im Experimentiermodul können Sie einen eigenen Entwurf einer Elektronikbaugruppe aufbauen und dieses Modul auf unterschiedlichste Art aus dem Mikrocontroller ansteuern.

5.5. Flachbandanschlussbelegung

Vier der zwanzig Flachbandadern sind für die Stromversorgung reserviert. Pin 19 für VCC und Pin 7, 8 und 20 für GND.

Alle übrigen Flachbandanschlüsse sind direkt mit festgelegten Anschlüssen des Mikrocontrollers verbunden.

Direkt neben den Anschlußnummern sind die Mikrocontroller-pinfunktionen spezifiziert. Ein Mikrocontrollerpin kann je nach Anwendungsprogramm auf verschiedenste Art verwendet werden. Überprüfen Sie bitte im Datenblatt bzw. Handbuch zum Mikrocontroller die genaue Definition der Pinfunktion.

Pin 1	PC5 (ADC5/SCL)	Serial Clock (für die I2C Datenkommunikation)
Pin 2	PC4 (ADC4/SDA)	Serial Data (für die I2C Datenkommunikation)
Pin 3	PC3(ADC3)	Digitaler Eingang/Ausgang oder analoger Messeingang
Pin 4	PC2(ADC2)	Digitaler Eingang/Ausgang oder analoger Messeingang
Pin 5	PC1(ADC1)	Digitaler Eingang/Ausgang oder analoger Messeingang
Pin 6	PC0(ADC0)	Digitaler Eingang/Ausgang oder analoger Messeingang
Pin 7	GND	GND (Mehrere Anschlüsse zur Entstörung vorgesehen)
Pin 8	GND	GND (Mehrere Anschlüsse zur Entstörung vorgesehen)
Pin 9	AVCC	Betriebsspannung AD-Konverter
Pin 10	PC6(RESET)	Mikrocontroller Reset Anschluss
Pin 11	PB5(SCK)	Digitaler Eingang/Ausgang
Pin 12	PB4(MISO)	Digitaler Eingang/Ausgang oder I2C Funktion Pin
Pin 13	PB3(MOSI/OC2)	Digitaler Eingang/Ausgang oder I2C function pin oder Timer2 Pin
Pin 14	PD3(INT1)	Digitaler Eingang/Ausgang oder externer Interrupt
Pin 15	PD6(AIN0)	Digitaler Eingang/Ausgang oder analoger Testeingang
Pin 16	D7(AIN1)	Digitaler Eingang/Ausgang oder analoger Testeingang
Pin 17	PD0(RXD)	Digitaler Eingang/Ausgang oder RS232 Eingang
Pin 18	PD1(TXD)	Digitaler Eingang/Ausgang oder RS232 Ausgang
Pin 19	VCC	VCC
Pin 20	GND	GND (Mehrere Anschlüsse zur Entstörung vorgesehen)

Belegung des Flachbandkabels in einer Grafik siehe Seite 16

ACHTUNG:

Der AVCC-Anschluss ist wegen der Filterung über die Spule nicht so stark belastbar wie der VCC-Anschluss!

Die Belegung des Flachbandkabels auf der YETI-Erweiterungsplatine

erstellt von Tester uwegw, keine Gewähr für Richtigkeit!

YETI Stecker Lötsseite		KABEL									
Erweiterungen	Nummer	Display/I2C	3	5	7	9	11	13	15	17	19
Signal	PC5/SCL	PC2/ADC3	PC1/ADC1	AGND	AVCC	PB5/SCK	PB3/OC2	PD6/AIN0	PDI/TXD	VCC	
Signal	PC4/SDA	PC2/ADC2	PC0/ADC0	AGND	RESET	PBA/MISO	PDB3/INT1	PDI/TXD	GND		
Erweiterungen	2	4	6	8	10	12	14	16	18	20	
Erweiterungen	Display/I2C	US	US	US	US	US	US	US	US	US	PLATINE

YETI Stecker Bestückungsseite		PLATINE									
Erweiterungen	Nummer	Display/I2C	US	6	8	10	12	14	16	18	20
Signal	PC5/SDA	PC2/ADC2	PC0/ADC0	AGND	RESET	PB4	PB3/INT0	PD6/AIN0	PDI/TXD	GND	
Signal	PC5/SCL	PC3/ADC3	PC1/ADC1	AGND	AVCC	PB5	PB3/OC2	PDI/TXD	VCC		
Erweiterungen	1	3	5	7	9	11	13	15	17	19	
Erweiterungen	Display/I2C	US	US	US	US	US	US	US	US	US	KABEL

Legende:

VCC	frei
GND	belegt
Erweiterungen, sonst frei	

Die Steckerleisten auf den Erweiterungsplatinen

Pin	entspricht Flachband Nr.	Pin	entspricht Flachband Nr.	Pin	entspricht Flachband Nr.
1	2	3	4	5	6
15	16	13	14	11	12
US	PD6/AIN0	PD3/AIN1	PD3/OC2	PD3/INT1	RESET

Pin entspricht Flachband Nr.
Signal
Erweiterungen

Pin	entspricht Flachband Nr.	Pin	entspricht Flachband Nr.	Pin	entspricht Flachband Nr.
1	2	1	2	1	2
1	2	17	18	9	7/8
US	PC5/SCL	PC4/SDA	PD0/RXD	AGND	PC0/ADC0

Pin entspricht Flachband Nr.
Signal
Erweiterungen

6. AUFBAU ELEKTRONIK

Überprüfen Sie zunächst anhand der Teileliste der Elektronik-Komponenten, ob alle Bauteile vorhanden sind.

6.1. Die Lötarbeit

Die Beschriftung der Leiterplatte zeigt genau, wo Sie die Bauteile bestücken müssen. Sollten Sie sich die Position genauer ansehen wollen, stehen Ihnen ein Bild der fertigen Platine und der Bestückungsplan zur Verfügung, siehe Seite 20 und 21.

Beim Bestücken einer Leiterplatte beginnen wir vorzugsweise mit den niedrigen Bauteilen. Das sind normalerweise die Widerstände. Schneiden Sie die Drahtenden kurz nach dem Einlöten ab, so daß Sie immer genug Platz zum Arbeiten haben.

Ehe Sie mit dem Löten beginnen, sollten Sie die integrierten Schaltungen probeweise kurz auf der Leiterplatte einsetzen und notfalls die Beinchen mit einer Flachzange genau ausrichten. Meistens sind die Beinchen etwas zu weit angewinkelt. Zum Schluß montieren Sie das IC in den IC-Sockel. ICs dürfen auf keine Fall direkt eingelötet werden, sondern müssen immer in den Sockel montiert werden!

WICHTIG

Der Elko und der IC-Sockel müssen mit korrekter Orientierung eingelötet werden!

TIP

Die IC-Beinchen können Sie einfach auf einer harten Tischfläche ausrichten! Legen Sie dazu die Beinchen auf die Tischfläche und drücken Sie diese vorsichtig in die richtige Position.

Bei technischen Fragen oder Problemen können Sie in folgenden Foren um Hilfe fragen:

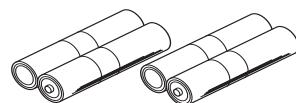
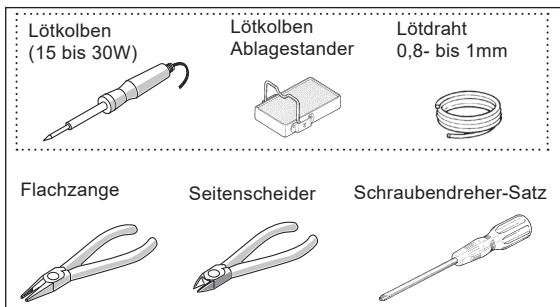
www.arexx.com --> Forum

www.roboternetz.de --> Forum

Warnungen

- * Mit dem Öffnen der Plastikbeutel mit Komponenten und Teilen erlischt das Rückgaberecht.
- * Lese vor dem Bauen zuerst die Gebrauchsanleitung aufmerksam durch.
- * Sei vorsichtig beim Hantieren mit den Werkzeugen.
- * Baue nicht im Beisein kleiner Kinder. Die Kinder können sich an den Werkzeugen verletzen oder kleine Komponenten und Teile in den Mund stecken.
- * Achte auf die Polung der Batterien.
- * Sorge dafür, daß die Batterien und die Batteriehalter trocken bleiben. Falls der YETI naß wird, entferne dann die Batterien und trockne alle Teile, so gut es geht.
- * Entferne die Batterien, wenn der YETI mehr als eine Woche ruht..

6.2. Notwendige Werkzeuge für den Zusammenbau



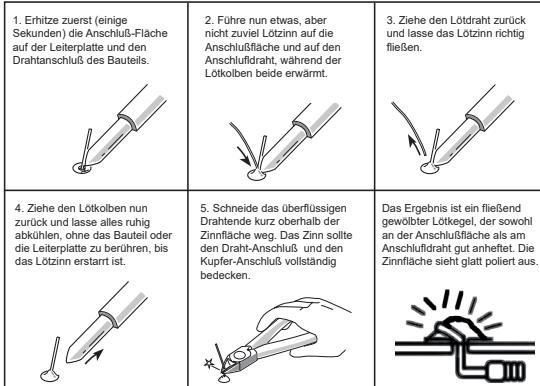
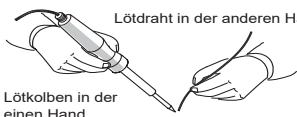
Benötigte Batterien: AA Akku, 4 Stück
(nicht im Bausatz enthalten)

6.3. Löten der Komponenten:

Benutze nur das von uns empfohlene Lötzinn, das ein spezielles Flußmittel für Elektronik-Bauteile enthält.



Die korrekte Haltung zum professionellen Löten:



Lötfehler orten und reparieren:

Kalte Lötstelle



Zinn am Bauteilanschluß,
aber nicht auf der
Anschlußfläche.



Zu wenig Zinn



Zinn ist nicht gut
geflossen.



Lötschluß



Zwei Anschluß-Flächen durch
Zinnbrücke mit einander ver-
bunden.



Gute Lötverbindung



Die Zinnfläche sieht
glatt poliert aus.



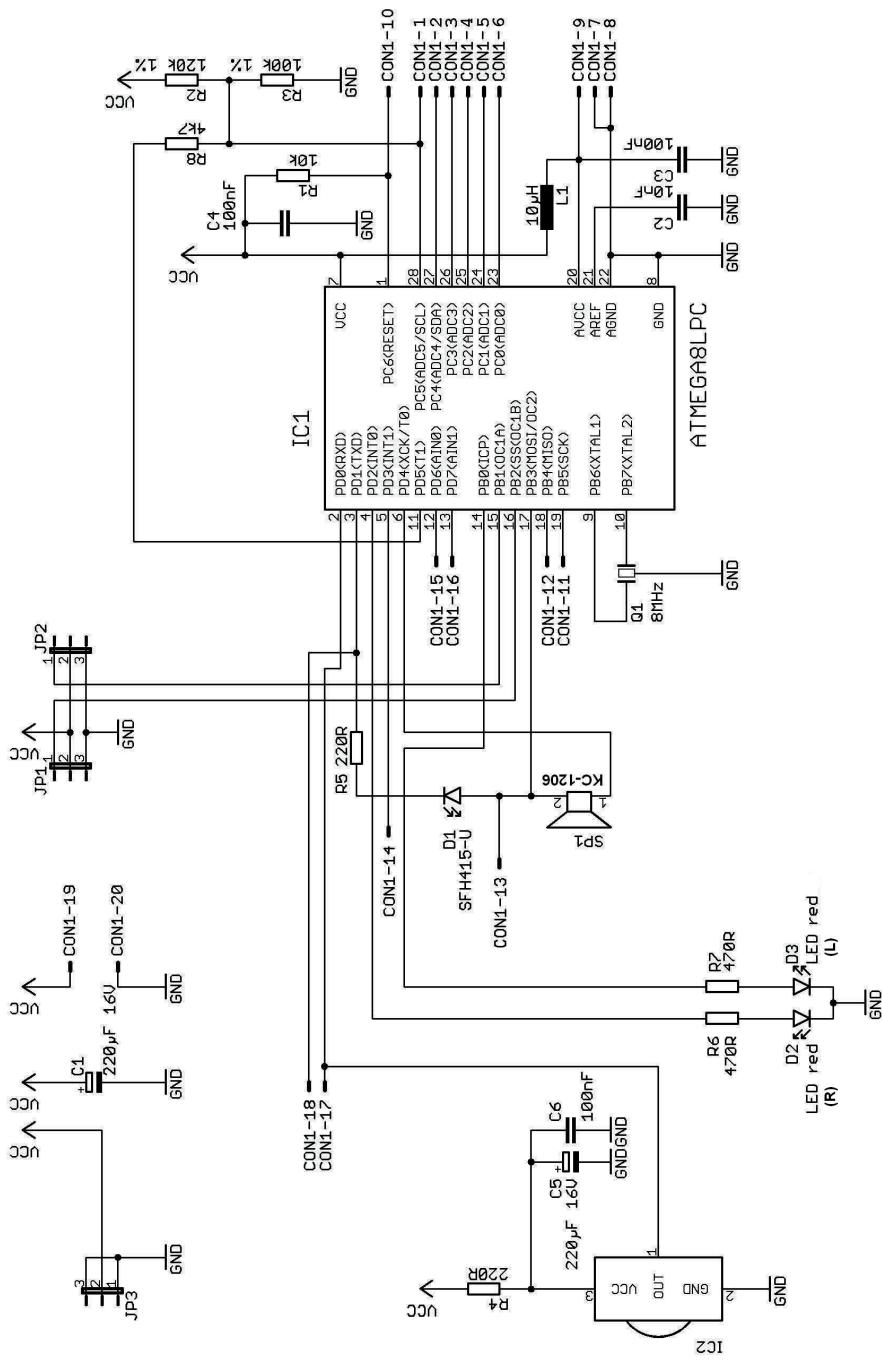
6.4. Bestückung Hauptplatine

WICHTIG! Für die Montage der Teile siehe 5.5 (Schaltplan) und 5.6 (Bilder)

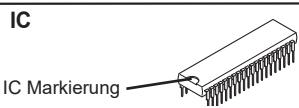
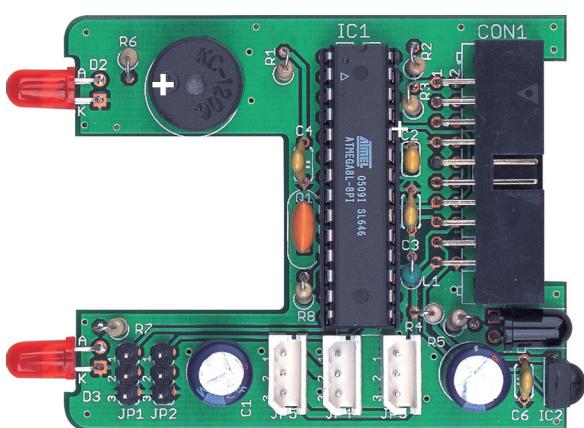
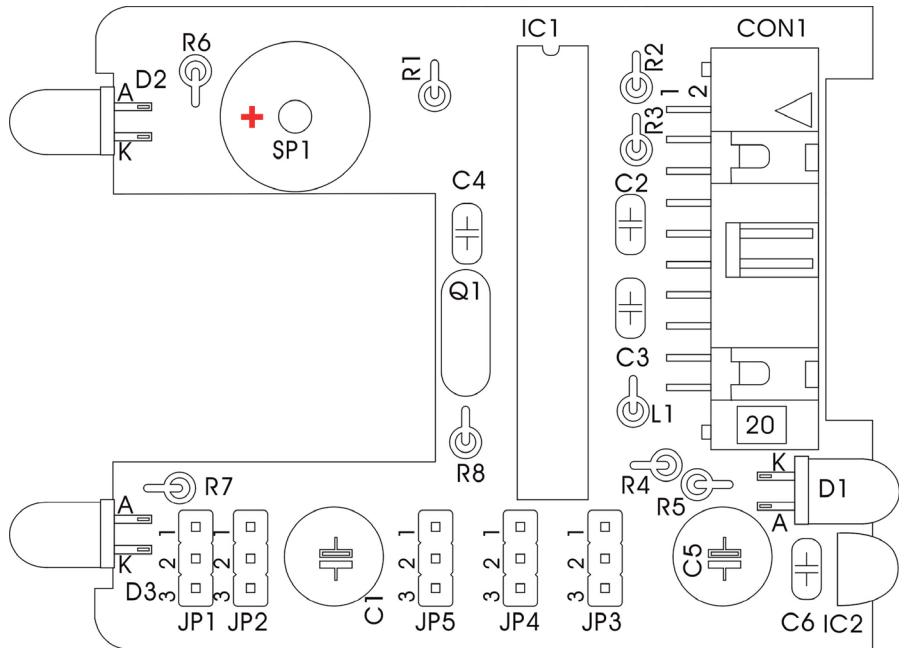
YETI Teileliste

Nr.	Name	St.
PCB1	Platine	1
IC1	ATmega328-P (auf richtige Polung achten)	1
IC2	SFH5110 IR-Empfänger-IC (auf richtige Polung achten)	1
R1	10K / 0.25W / 5% (braun, schwarz, orange, gold)	1
R2	120K / 0.25W / 1% (braun, rot, schwarz, orange, braun)	1
R3	100K / 0.25W / 1% (braun, schwarz, schwarz, orange, braun)	1
R4	220R / 0.25W / 5% (rot, rot, braun, gold)	1
R5	220R / 0.25W / 5% (rot, rot, braun, gold)	1
R6	470R / 0.25W / 5% (gelb, violett, braun, gold)	1
R7	470R / 0.25W / 5% (gelb, violett, braun, gold)	1
R8	4K7 / 0.25W / 5% (gelb, violett, rot, gold)	1
L1	10uH (braun, schwarz, schwarz, silber) ist 0 Ohm	1
C1	220uF/16V (auf richtige Polung achten)	1
C2	10nF Aufdruck: 103	1
C3	100nF Aufdruck: 104	1
C4	100nF Aufdruck: 104	1
C5	220uF/16V (auf richtige Polung achten)	1
C6	100nF Aufdruck: 104	1
D1	SFH415-U IR-LED (auf richtige Polung achten)	1
D2	LED Rot, 5 mm (auf richtige Polung achten, sehe 5.6)	1
D3	LED Rot, 5 mm (auf richtige Polung achten, sehe 5.6)	1
Q1	Quarz, 8Mhz / 3 PIN	1
SP1	Piepser, 5V, (KC1206) (auf richtige Polung achten)	1
IC socket	28 PIN, IC Fuß (auf richtige Polung achten)	1
JP1	3 PIN, Platinensteckverbinder, schwarz	1
JP2	3 PIN, Platinensteckverbinder, schwarz	1
JP3	3 PIN, Platinensteckverbinder, weiß	1
JP4	3 PIN, Platinensteckverbinder, weiß	1
JP5	3 PIN, Platinensteckverbinder, weiß	1
CON1_PCB	Konnektor, male, 20 pins, für Flachband/ 90° gewinkelt	1
Klett-Band	(male) (bereits vormontiert)	2
Klett-Band	(female) (bereits vormontiert)	2

6.5. Schaltbild YETI



6.6. Hauptplatine



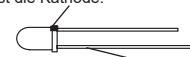
Widerstand & Drossel

Die Werte werden mit einem Farbcodemarkiert.



LED & IR-LED

Die Seite mit einer flachen Markierung ist die Kathode.



Kondensator

Kein Polung



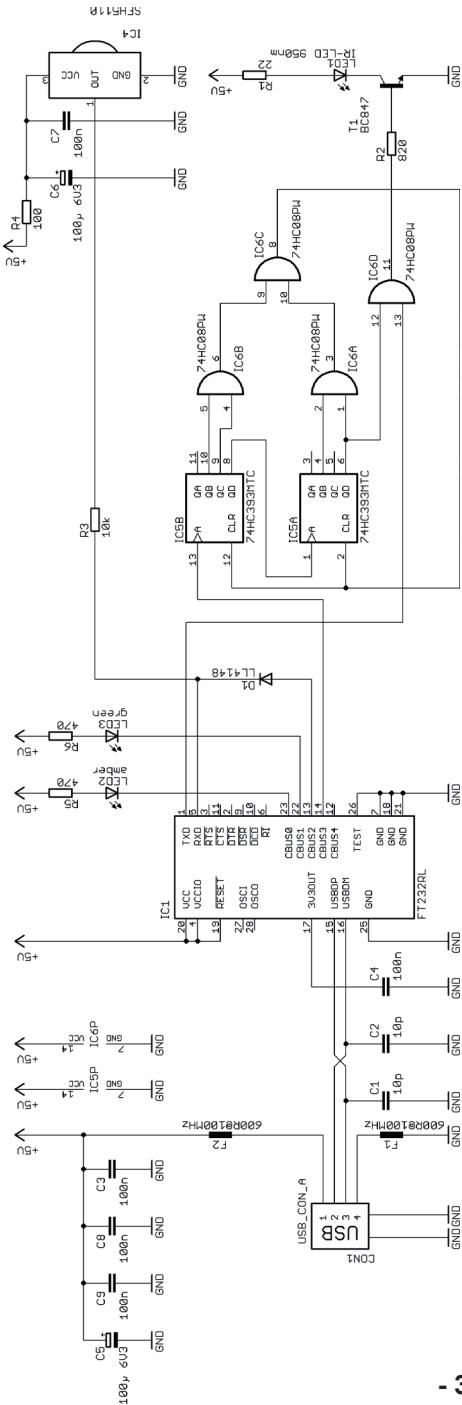
ELKO (Elektrolyt Kondensator)

Der längere Anschluss ist die (+) Seite.



Die Seite mit einer weißen Markierungslinie ist die (-)

6.7. Schaltbild USB-IR-Transceiver



6.8. Layout USB-Infraread-Transceiver PCB

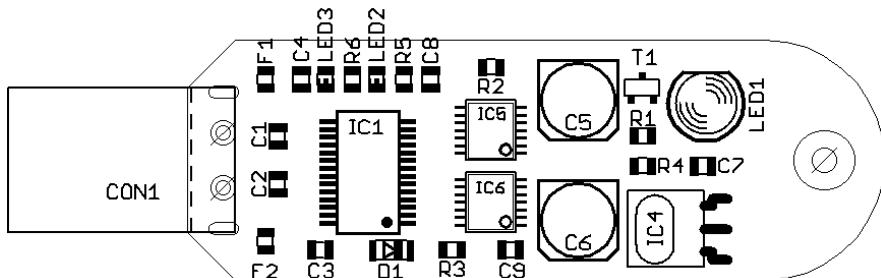


Fig. 5.1.: Layout USB- Infrared-Transceiver

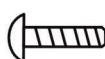


7. TEILELISTE MECHANIK

Mutter M3



Linsenkopf-
schraube
M3x8mm



O 8 St.

Stellring
Groß



O 4 St.

Stellring
Klein



O 4 St.

Maden-
Schraube



O 10 St.

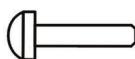
Mutter M2



Distanzrollen



Niete



O 4 St.

Gestänge-
anschluss



O 4 St.

Servo
Schraube



O 2 St.

O 2 St.

Kugelkopf-
schraube



Gelenkkopf mit
Innengewinde



Kugelkopf-
mutter



Sechskant-Winkel-
schraubendreher



O 4 St.

O 4 St.

O 4 St.

O 1 St.

Drahtachse lang



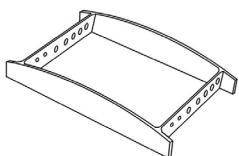
O 2 St.

Drahtachse kurz



O 4 St.

Fuß



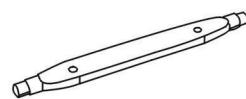
O 2 St.

Achse
5x80mm



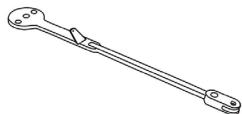
O 2 St.

Fuß Kuppelstange



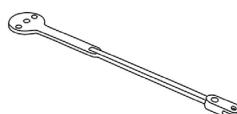
O 2 St.

Hinterbein



O 2 St.

Vorderbein



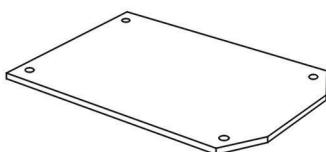
O 2 St.

Stelldraht



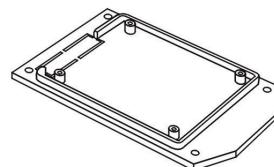
O 2 St.

Oberdeckel (meist oben)



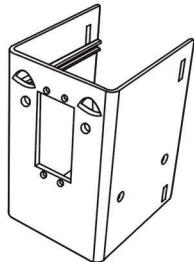
O 1 St.

Oberteilsocke (unteres Teil)



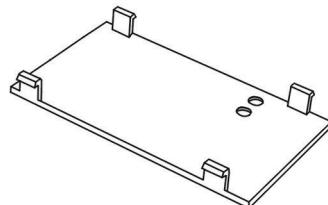
O 1 St.

Kopfteil



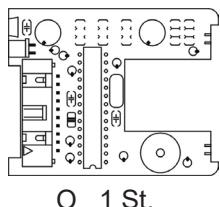
O 1 St.

Hintere Abdeckung



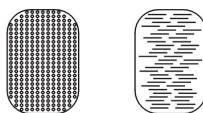
O 1 St.

YETI Hauptplatine
mit Bestückung



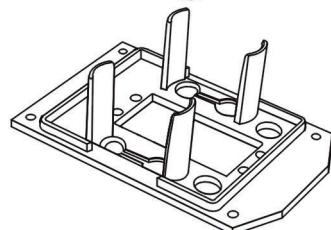
O 1 St.

Klett-Band
vormontiert

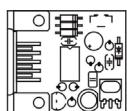


O 2 St. Male
O 2 St. Female

Bodenteil

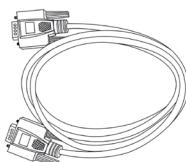


IR/Transceiverplatine
mit Bestückung



O 1 St.

RS-232 Kabel

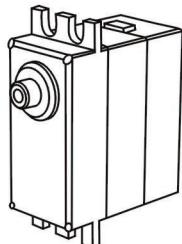


O 1 St.

O 1 St.

7.1. Wichtige Elektronikteile

Servo Motor



O 2 St.

Servo Achse



O 1 St. 1mm
O 1 St. 2mm

Mit Kabelsatz vormontiert

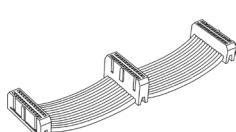


O 1 St.



O 1 St.

Flachbandkabel



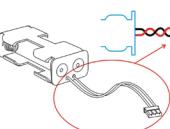
O 1 St.

Kabelsatz vormontiert



O 1 St.

Batteriehalter



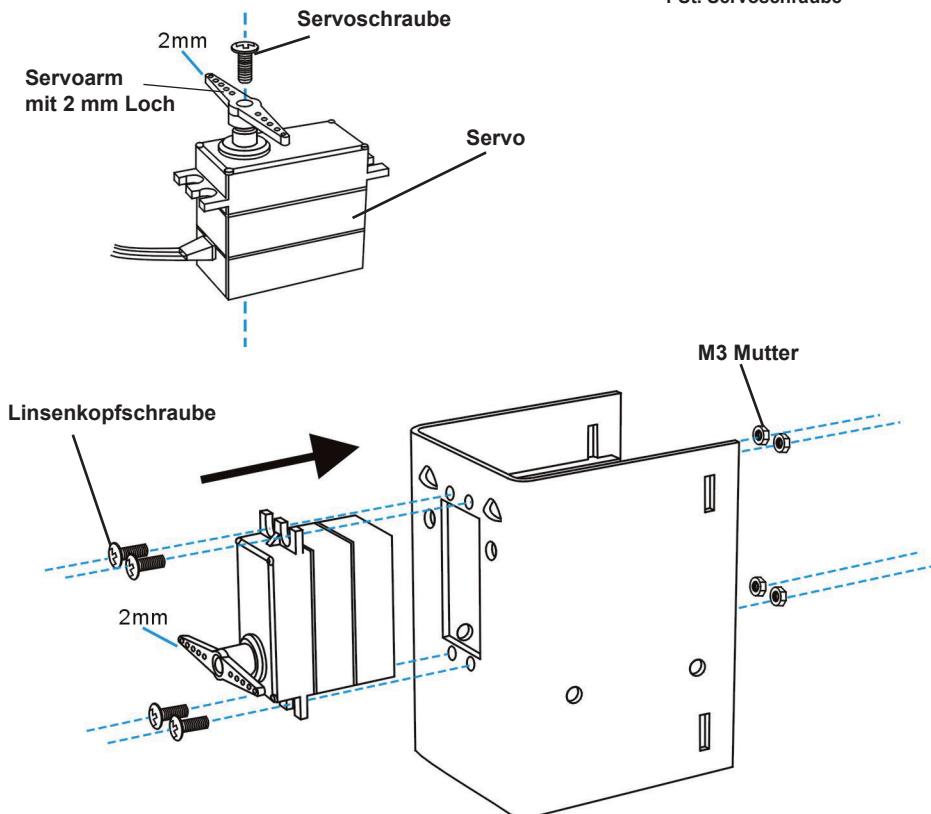
O 2 St.

7.2. Bauanleitung für die Mechanikteile

Montage des Kopfservos:

Zur Montage des Vorderservos wird folgendes benötigt;

- 1 St. Kopfteil
- 1 St. Servo
- 4 St. Linsenkopfschraube
- 4 St. Mutter M3
- 1 St. Servoarm mit 2mm Loch
- 1 St. Servoschraube



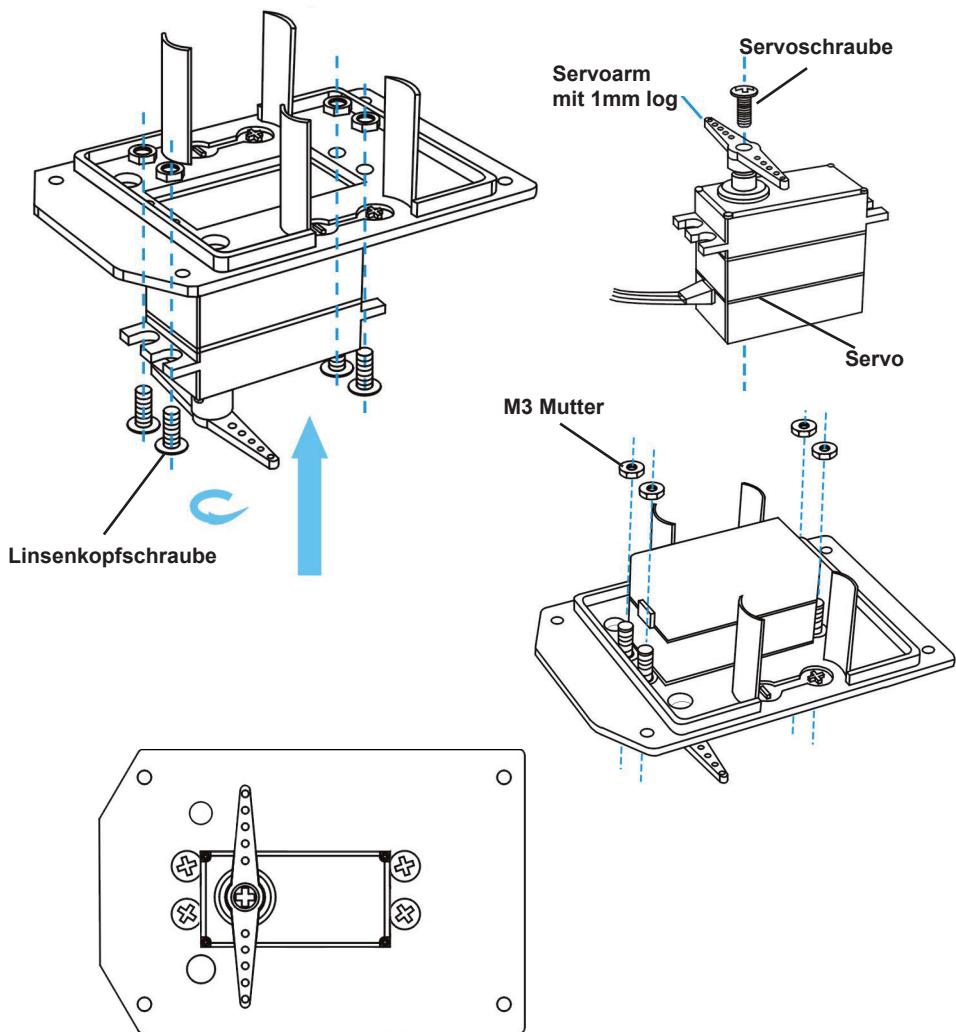
Befestigen Sie den Servo genau so, wie es in den Zeichnungen beschrieben wird.

Montieren Sie die Servoachse auf dem Servo, beachten Sie dabei bitte die kleine Detailskizze!

Montage des Bodenservos:

Zur Montage des Bodenservos wird folgendes benötigt;

1 St. Bodenteil
1 St. Servo
4 St. Linsenkopfschraube
4 St. Mutter M3
1 St. Servoarm mit 1 mm Loch
1 St. Servoschraube



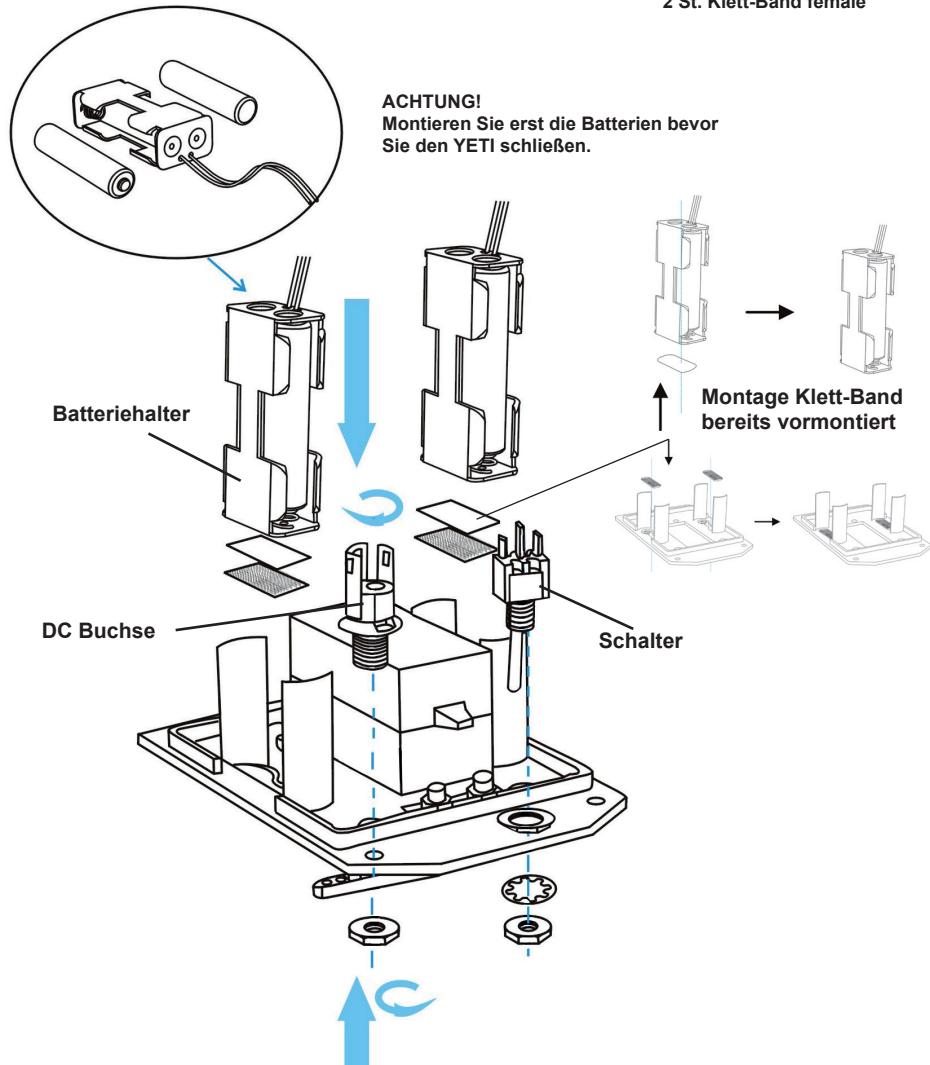
Befestigen Sie den Servo genau so, wie es in den Zeichnungen beschrieben wird.

Montieren Sie die Servoachse auf dem Servo, beachten Sie dabei bitte die kleine Detailskizze!

Endmontage der Bodenplatte:

Zur Endmontage des Bodenservos wird folgendes;

- 1 St. Montiertes Bodenteil
- 1 St. Schalter
- 1 St. DC Buchse
- 2 St. Batteriehalter
- 4 St. AA Akku
- 2 St. Klett-Band male
- 2 St. Klett-Band female

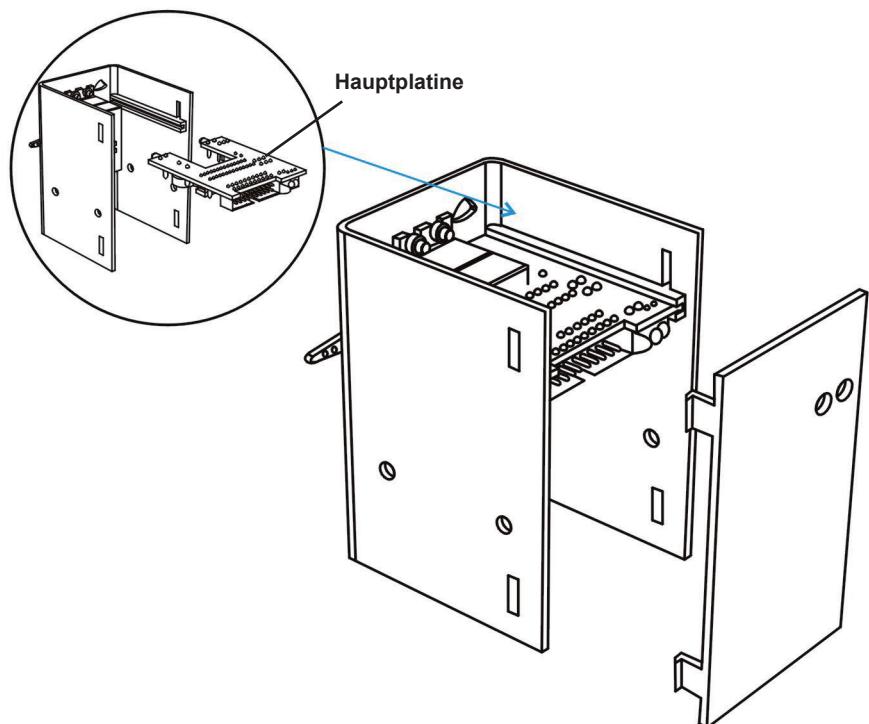


Montieren Sie die Bodenplatte genauso, wie es in der Zeichnung beschrieben wird.

Montage des YETI Kopfes:

Zur Montage des Kopfes wird folgendes benötigt;

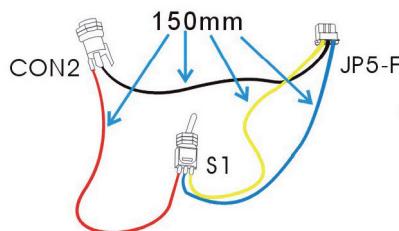
1 St. Montiertes Bodenteil
1 St. Montiertes Kopfteil
1 St. Montierte Hauptplatine
1 St. Hinterabdeckung



ACHTUNG!

Montieren Sie erst die komplette Verkabelung bevor Sie die Hinterabdeckung schließen.

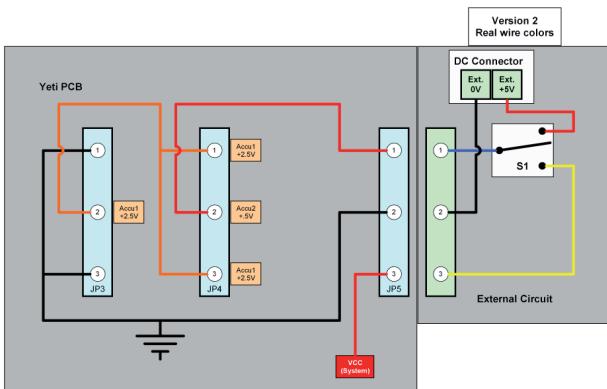
Den Schaltplan für den Kabelanschluß finden Sie auf den Seiten 33 und 34.



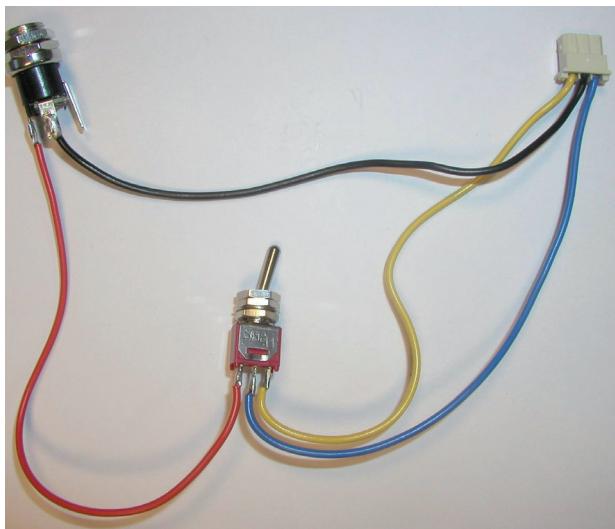
Kabel Endmontage:

Zur Endmontage der Kabel wird folgendes benötigt;

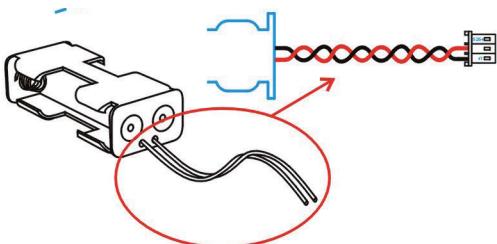
- 1 St. Montiertes Bodenteil
- 1 St. Montiertes Kopfteil
- 1 St. Montierte Hauptplatine
- Montierter Kabelsatz mit;
 - 1 St. Kabel schwarz
 - 1 St. Kabel blau
 - 1 St. Kabel gelb
 - 1 St. Kabel rot



Montieren Sie zunächst die Verkabelung genau so wie es in die Zeichnung beschrieben wird.

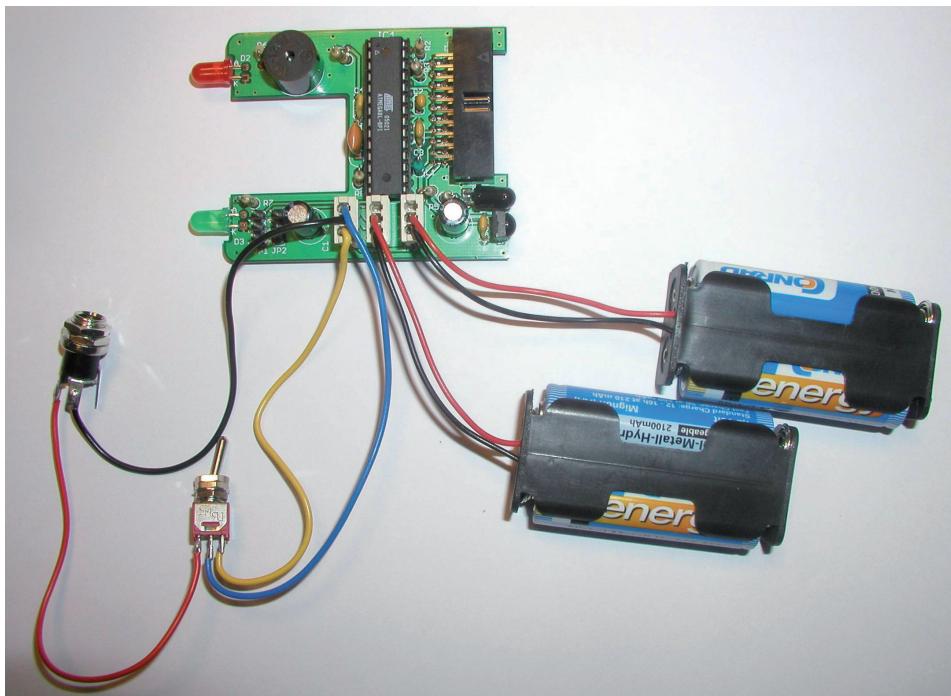
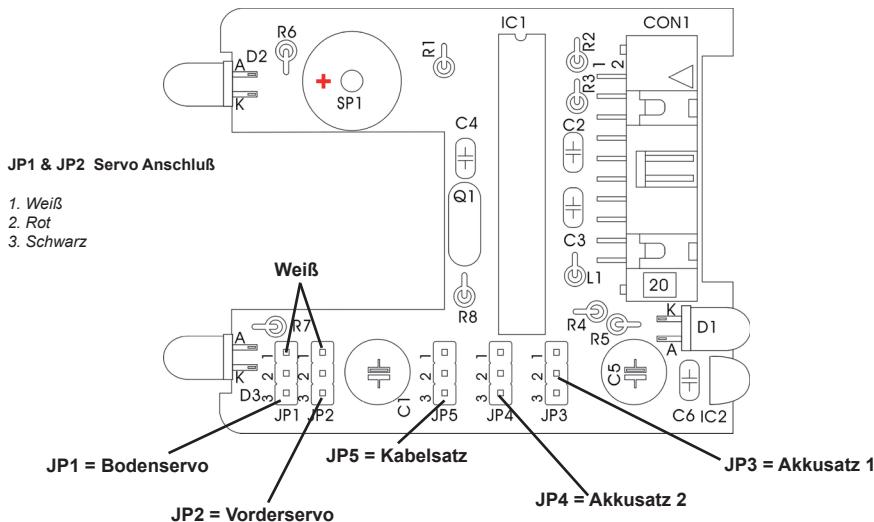


Fertig montierter Kabelsatz



Montierter Akku Kabelsatz

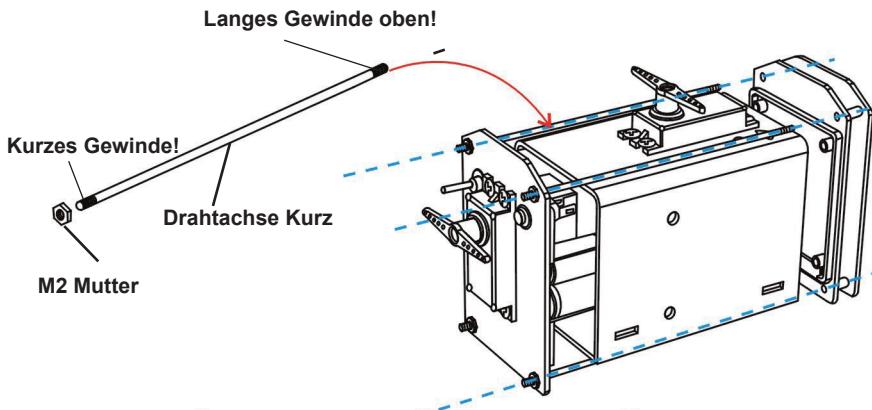
Anschlussbelegung der Hauptplatine



Endmontage des YETI Kopfs:

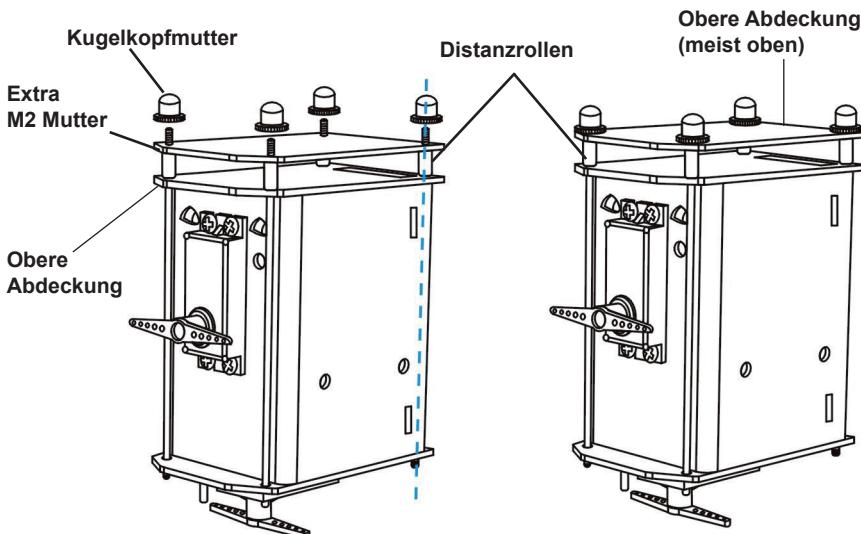
Zur Endmontage des Kopfs wird folgendes benötigt;

- 1 St. YETI Kopf
- 1 St. Obere Abdeckung (meist oben)
- 1 St. Obere Abdeckung
- 4 St. Drahtachse Kurz
- 4 St. M2 Mutter
- 4 St. Kugelkopfmutter
- 4 St. Distanzrollen



TIP!

Benutze die extra M2 Muttern aus den Ersatzteile für die Befestigung der oberen Abdeckung.

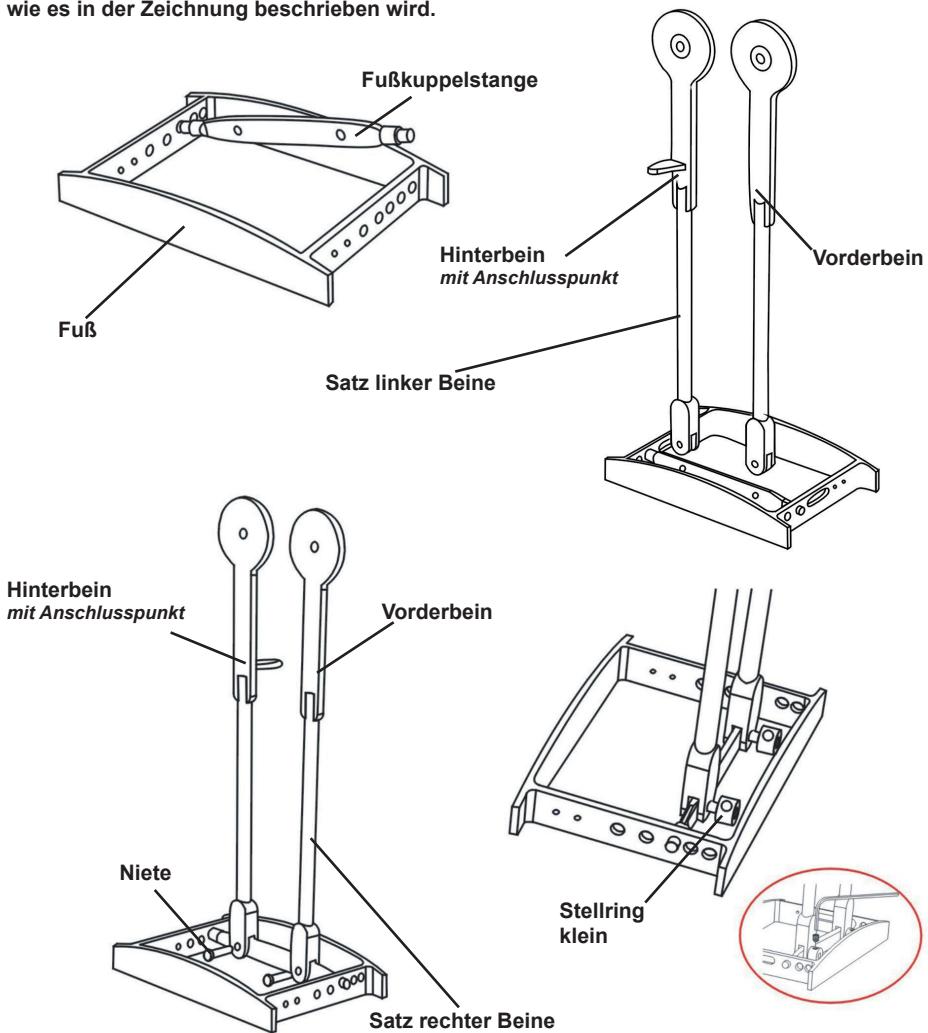


Bein- und Fußmontage:

Zur Bein- und Fußmontage wird folgendes benötigt;

2 St. Fuß
2 St. Vorderbein
2 St. Hinterbein
2 St. Fußkuppelstange
2 St. Niete
2 St. Stellring klein

Montieren Sie Beine und Fuß genauso,
wie es in der Zeichnung beschrieben wird.

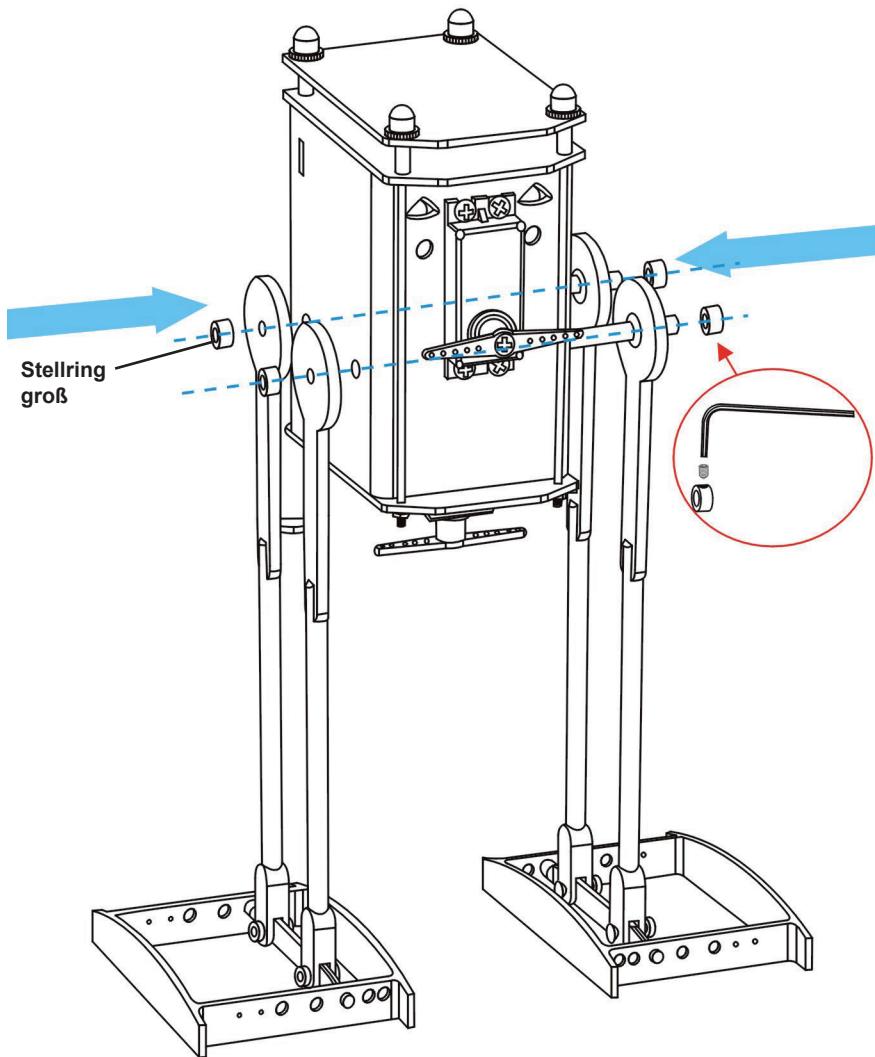


Montage der Beine I:

Zum ersten Teil der Montage der Beine wird folgendes benötigt;

1 St. Montiertes Chassis
1 St. Montierte Beinesatz Links
1 St. Montierte Beinesatz Rechts
2 St. Achse 5 x 80mm
4 St. Stellring 5 mm

Montieren Sie die Beine an am Chassis genau so, wie es in der Zeichnung beschrieben wird.

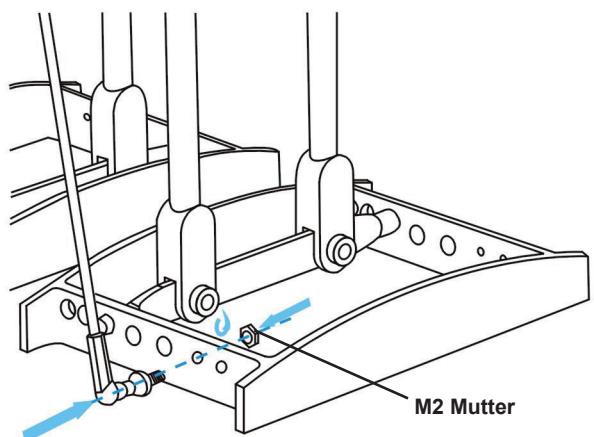
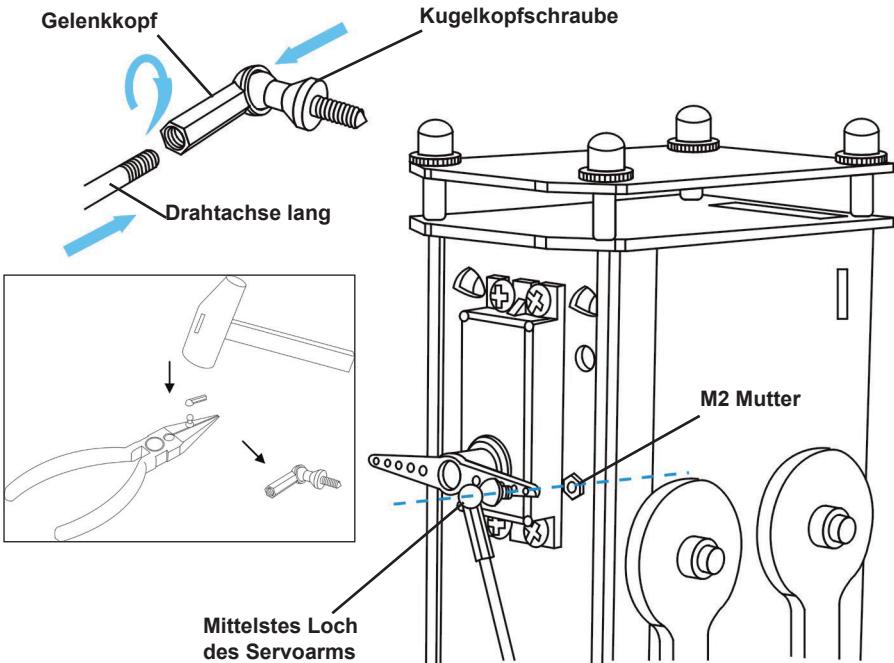


Montage der Beine II:

Zum zweiten Teil der Montage der Beine wird folgendes benötigt;

1 St. Montiertes Chassis
4 St. Gelenkkopf
4 St. Kugelkopfschraube
2 St. Drahtachse lang
4 St. Mutter M2

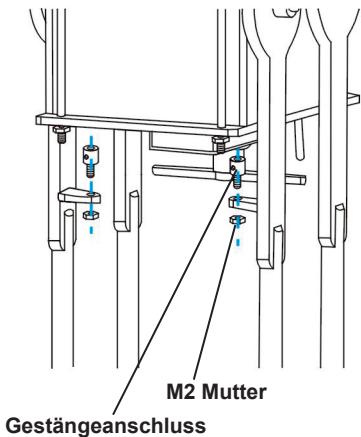
Montiere das Drahtende mit Kugelkopf genau so, wie es in der Zeichnung beschrieben wird.



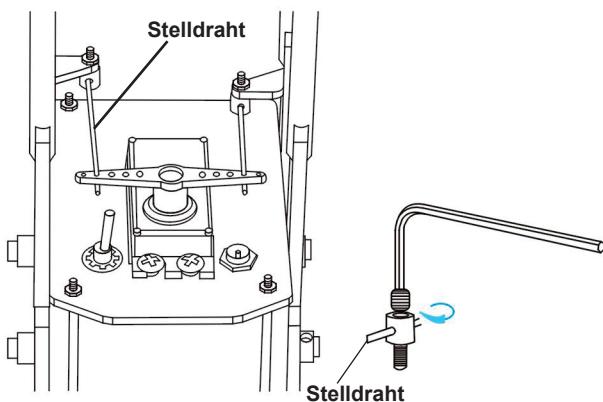
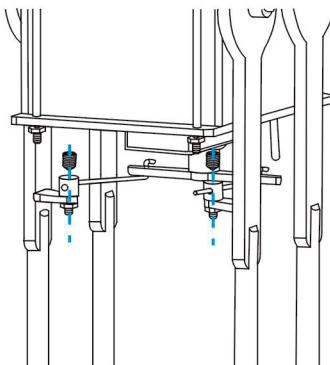
Endmontage der YETI Beine:

Zur Endmontage der YETI Beine wird folgendes benötigt;

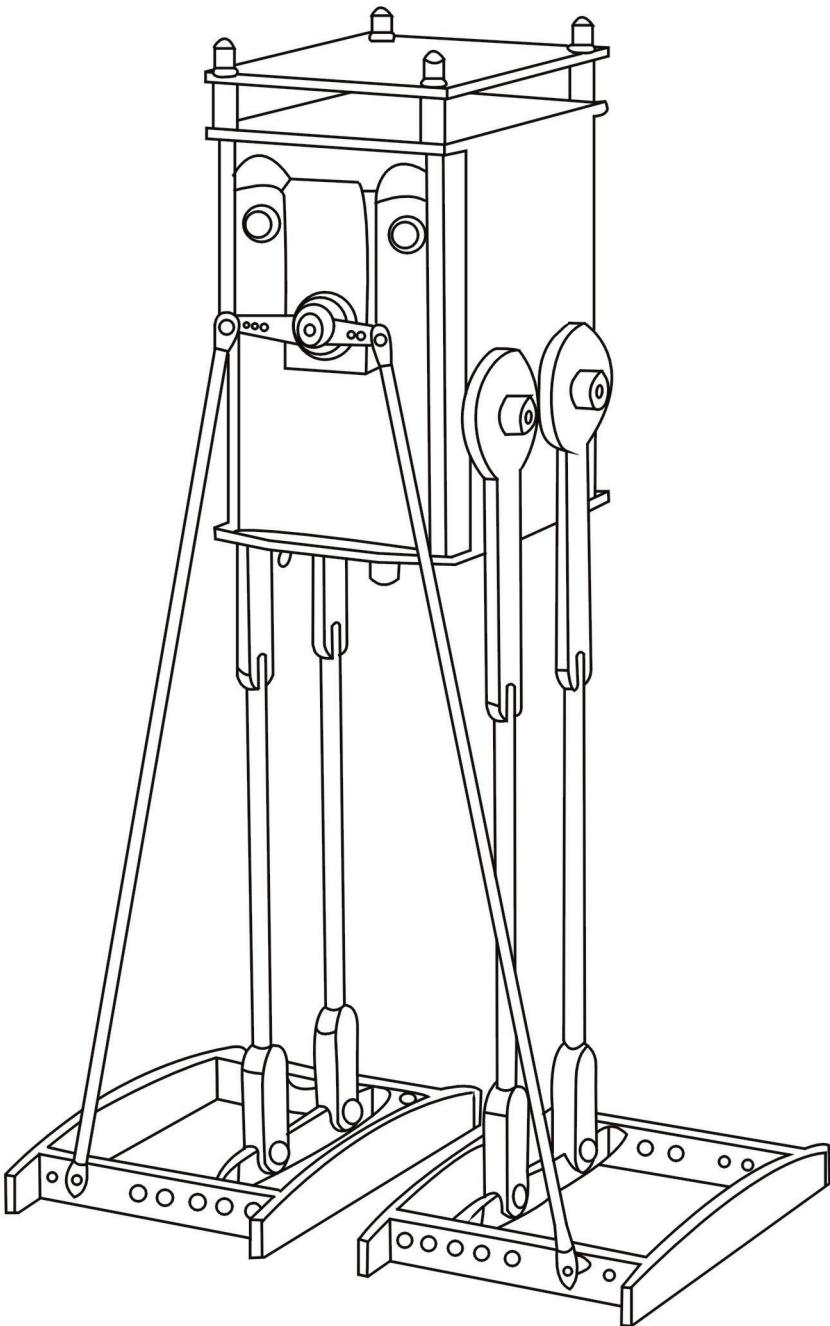
- 1 St. Montiertes Chassis
- 2 St. Mutter M2
- 2 St. Gestängeanschluss
- 2 St. Stelldraht



Montieren Sie die Beine genau so an den Servo, wie es in der Zeichnung beschrieben wird.



DER FERTIGE YETI !



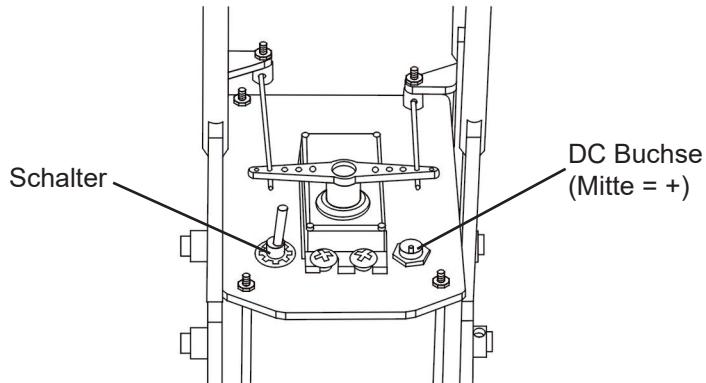
8. YETIS AKKUS AUFLADEN

Als Versorgungsspannung ist eine Gleichspannung von 4,8 Volt vorgesehen, die mit 4 NiMH (1,2V) Akkumulatoren erzeugt wird.

ACHTUNG: Die Akkus sind nicht durch einen Widerstand oder eine Sicherung geschützt!

Verwenden Sie zum Laden nur geprüfte Ladegeräte, am besten ein Mikroprozessorgesteuertes!

Alternativ funktioniert auch ein Steckernetzgerät mit sehr niedrigem Ladestrom, z.B. 5 Volt / 300 mA.



Akkus laden:

1. Den Schalter auf Aus stellen!
2. Ladegerät an die DC-Buchse anschließen.

SIEHE APPENDIX für extra Info!

9. SOFTWARE

9.1. Schreiben Sie ein eigenes YETI-Programm

Diese Kapitel unterstützt alle unerfahrene Programmierer sofern zu mindest etwas Hintergrundwissen und ein wenig Programmierbasis vorhanden ist.

Das Kapitel ist relativ schwierig und enthält eine Anzahl neuer Definitionen und Wortbegriffe, aber das positive Ergebnis ist die Mühe wert. Ein Basiswissen hilft Ihnen beim Nachfragen, bei der Studie von Internetdokumenten in Benutzerkreisen und bei der Beratungssuche nach Unterstützung oder Informationen.

Selbstverständlich werden Sie in der Lage sein ein selbstgeschriebenes Programm in den YETI-Roboter zu übertragen, aber wie macht man das?

- Sie schreiben ein Programm in einer Sprache, die man “C++” nennt (siehe zum Beispiel das Programm “demonstration.ino“, das sich im Beispielbereich der YETI-Bibliothek befindet).
- Anschließend übertragen Sie das Programm in den YETI.

Tatsächlich benötigen Sie nur zwei Schritte um das Programm zu erstellen. Diese einfache Arbeitsweise wird auch regelmäßig von erfahrenen Programmierern benutzt. Zunächst werden wir aber die Anfänger helfen indem wir die Methode schrittweise erklären.

9.2. Stufe 1 Die Erstellung eines “C++”-Programms.

Normalerweise werden Sie ein YETI Programm für eine spezielle Programmiersprache entwerfen und schreiben. In unserem Fall wählen wir dazu eine populäre Programmiersprache mit dem Namen “C++”. Zum Schreiben eines YETI Programms werden Sie einen speziellen Texteditor benötigen. Der von uns praktizierte Texteditor ist Bestandteil des Arduino-Programms.

Natürlich könnten Sie auch Programme im „Notepad“ schreiben oder einen gängigen Textverarbeiter wie MS Word verwenden, aber wir empfehlen Ihnen dringend solche allgemeine Textverarbeiter nicht dazu zu benutzen, weil die Textverarbeitung dadurch ziemlich erschwert wird.

Tatsächlich bietet das Arduino-Programm einige Vorteile, die Ihnen beim Editieren helfen. Hinzu kommen die Hilfsmittel zur Fehlersuche in verschiedenen Programmiersprachen. Die Arduino Entwicklungsumgebung (IDE) wird Ihnen helfen beim Hervorheben aller Spezialkommandos, Kommentaren und Variablen, die farblich leicht unterscheidbar sofort erkannt werden. Diese Hilfsmittel werden Ihnen beim Programmieren ein wertvolle Unterstützung bieten.

Stufe 2 Das Compilieren eines “YETI”-Programms

Nun folgt die einfache Phase: das hochladen des Programms in den YETI. In einer früheren Yeti-Version gab es noch einen Zwischenschritt ehe wir das Programm übertragen konnten, aber in der vorliegenden Version haben den Prozess beträchtlich vereinfachen können.

Der YETI verfügt intern über 32kiloBytes Flash Speicherplatz, in dem das Programm gespeichert wird. Für uns stehen davon etwa 30kiloBytes Platz zur freien Verfügung. Der Rest wird vom YETI benötigt für ein verstecktes Programmteil, das man den “Bootloader” nennt und uns das Hochladen eine Programms stark vereinfacht. Mit normalen Benutzerkommandos kann man die “Bootloader” Software weder editieren, noch lesen oder löschen.

Nach dem Einschalten wird der YETI immer zuerst den Bootloader starten ehe das System das vom Benutzer erstelltes Programm startet. Dazu bleibt der Roboter etwa 5 Sekunden im “Bootloader”-Modus. In diesen 5 Sekunden beobachtet YETI den seriellen Bus um festzustellen ob ein Benutzer vielleicht versucht ein Programm hochzuladen. Das Hochladen kann wahlweise kabelgebunden oder drahtlos stattfinden. Je nachdem ein Benutzer die Programmiererweiterung und ein Stecker auf dieser Leiterplatte als Brückenschalter aufgesteckt worden ist, wird das System auf höchste oder auf geringere Übertragungsgeschwindigkeit geschaltet. Diese Geschwindigkeitswahl ist notwendig weil die drahtlose, serielle Datenübertragung keinen Hochgeschwindigkeitsmodus zulässt.

Nachdem YETI festgestellt hat, dass ein Benutzer versucht ein Programm hochzuladen liest das System die Daten von seriellen Bus und überträgt diese in den internen Flash-Speicher.

10. SOFTWAREINSTALLATION UND INITIALE KOMMANDOS

Legen Sie den YETI CD in Ihrem PC. Sofern alles stimmt wird die CD automatisch starten. Andernfalls öffnen Sie die CD mit dem Windows Explorer. Nach der Wahl der Sprache finden Sie alle benötigten Programme im Software-Menü.

Ehe Sie mit den Programmen arbeiten können müssen Sie diese auf ihre Festplatte kopieren. Um Programme auf eine Festplatte zu kopieren benötigen Sie eine Administratorberechtigung am PC. Falls Sie nicht als Administrator eingeloggt sind, sollten Sie sich ausloggen und danach wieder als Administrator einloggen. Beim Kopieren der Software geschieht folgendes:

1. Kopiere den Bereich mit Namen „Arduino“ nach C:\Program Files\

10.1 Windows

10.1.1 ARDUINO IDE

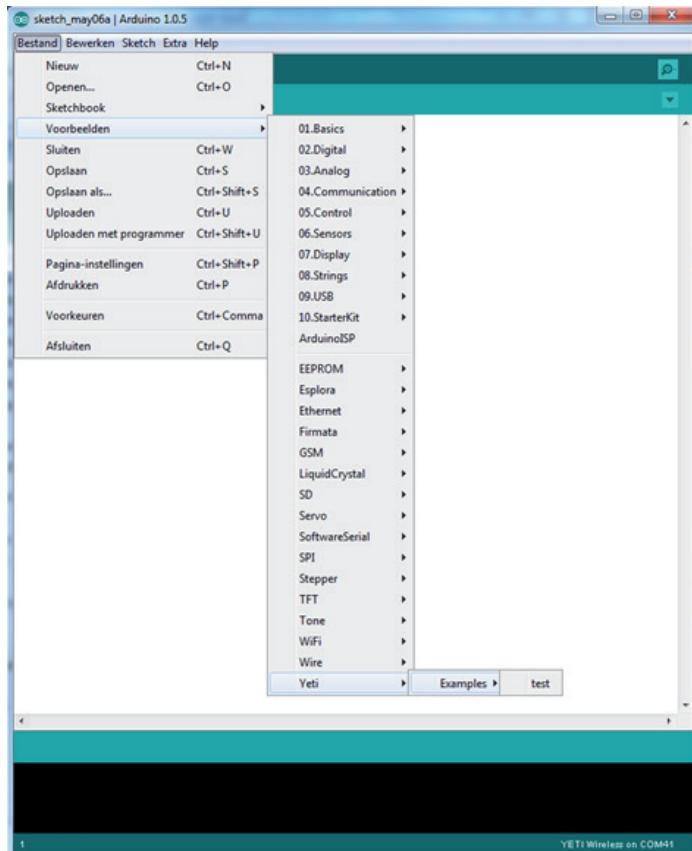
Im Arduino Installationsbereich (C:\Program Files\Arduino) klicken Sie mit der rechten Maustaste auf die ausführbare (Executable) Arduino-Datei und legen dazu eine Verknüpfung auf Ihrem Desktop an. Klicken Sie danach auf diese Desktop-Verknüpfung um Arduino zu starten.

Nachdem die Arduino IDE (Entwicklungsumgebung) gestartet wurde, was beim ersten Mal etwas mehr Zeit kosten kann, klicken Sie auf den Menüreiter “Extra”. Wählen Sie jetzt den Reiter “Boards”.

Falls Sie die drahtlose Kommunikationsverbindung benutzen müssen Sie YETI Wireless (drahtlos) wählen, sonst wählen Sie YETI Wired High-Speed (Hochgeschwindigkeitskabelverbindung).

Zusätzliche Information über die Arduino Entwicklungsumgebung steht Ihnen zur Verfügung auf der Webseite: <http://www.arduino.cc/en/Guide/HomePage>

Um nun eines der Beispielprogrammen zu testen können Sie den “File” -Reiter anklicken und “Examples” (Beispiele) wählen. Weiter in diesem Menü wählen Sie dann “Yeti” und klicken “Examples” (siehe nachfolgende Abbildung).



Im „Extra“-Reiter selektieren Sie auch den korrekten COM-Port, Falls es nur einen Eintrag gibt, wählen Sie den gemeldeten COM-Port. (Dabei sollte der Yeti Programmierer [kabelgebunden/ drahtlos] eingesteckt sein).

Falls mehrere COM-Ports zur Auswahl vorhanden sind, wechseln Sie zum Windows Startmenü. Dort klicken Sie mit der rechten Maustaste auf “Computer” und wählen “Manage”. Dann wählen Sie den Device Manager und klicken auf “Ports (COM & LPT)”. Dort suchen Sie nach Namen wie “USB Serial Port(COM xx)” oder “YETI Programmer”.

Die Menüknöpfe im obersten Fensterbereich entsprechen den nachfolgenden Kommandos:



Verify (Überprüfen)

Fehlerprüfung ihres Programmentwurfs.



Upload (Hochladen)

Compiliert den Programmtext und überträgt den Programmcode zum Arduino I/O board. Siehe den untenstehenden Informationspunkt zum Hochladen.



New (Neu)

Erzeugt einen neuen Programmentwurf (Englisch Sketch; Deutsch Skizze).



Open (Öffnen)

Auflistung aller Programmentwürfen (Sketches) in ihrem Entwurfsbereich (Sketchbook). Das Durch Anklicken eines Programms wird die Entwicklungsumgebung den Text im aktuellen Fenster öffnen.

Hinweis: aufgrund eines Javafehlers erlaubt diese Menü kein Blättern. Falls Sie ein Programm anklicken weiter unten in der Liste anklicken wollen, können Sie alternativ dazu das Menü File --> Sketchbook benutzen.



Save (Speichern)

speichert ihren Programmentwurf.



Serial Monitor (Serieller Monitor)

Öffnet den seriellen Monitor. Diese Option steht zur Verfügung um Informationen aus ihrem Programm zurück an den PC zurück zu schicken.

Falls Sie noch kein spezielles Beispielprogramm gewählt hatten, können Sie jetzt eins anklicken. Als erstes Programm empfehlen wie Hello World. Verbinden Sie wahlweise den drahtlosen Yeti Adapter mit Ihrem PC oder die Kabelgebundene Programmierer sowohl zum PC und Yeti.

Um den Yeti zu programmieren klicken Sie den Upload-Knopf. Nachdem Sie diesen angeklickt haben schalten Sie den Yeti ein. Warte aber nicht zu lange mit dem Einschalten des Yetis. Falls Sie eine Drahtverbindung benutzen kann Yeti auch vor dem Anklicken des Upload-Knopfes eingeschaltet werden und wird das System dann automatisch durch ein Reset zurücksetzen.

Falls Sie eine Fehlermeldung erhalten, sollten Sie die USB-Verbindung überprüfen, ob auch der korrekte COM-Port gewählt worden ist, und ob die Batterien des Yetis vollgeladen und die Kabelverbindungen vollständig intakt sind. Falls Sie drahtlos Datenübertragung verwenden kann auch der Abstand zwischen Yeti und Programmierer zu groß gewählt worden sein oder die Datenkommunikation durch zu viel Umgebungslicht gestört werden. Die drahtlose Programmierung kann durch Sonnenlicht gestört werden, das vielleicht abgeschirmt werden muss.

Zum Betrieb der kabelgebundenen Programmierung müssen Sie den Bootloader auf kabelgebundene Kommunikation umschalten und überprüfen ob CON8 auf dem Programmiererboard obenauf dem YETI mit einer Schaltbrücke bestückt (verbunden) worden ist. CON8 ist der zuständige Schalter für die Übertragungsgeschwindigkeit. Falls die Brücke vorhanden ist, wird die Übertragungsgeschwindigkeit auf 57600 Baud geschaltet. Ohne Programmierboard oder ohne CON8-Brückenschalter wird die Übertragungsgeschwindigkeit auf 2400 Baud herabgesetzt.

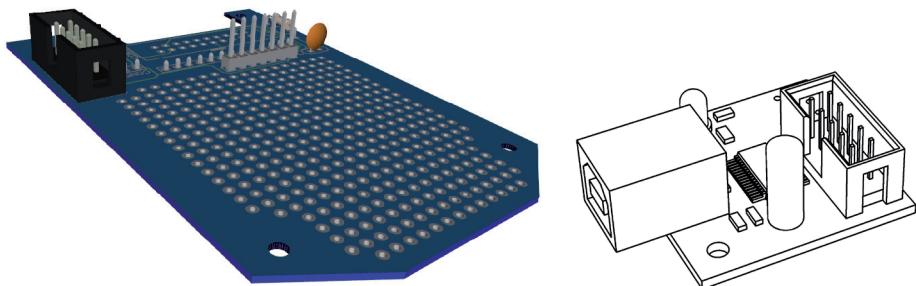
11. TEST UND BETRIEB

Nach der Zusammenbauphase werden wir den Roboter in Bewegung bringen. Zuerst werden wir jedoch wie man Fehler findet und beseitigt, die wir möglicherweise in den vorhergehenden Phasen gemacht haben. Diese Fehlersuche und deren Beseitigung soll selbstverständlich ohne Beschädigung der Bauteilen stattfinden.

11.1. Der kabelgebundene „USB nach Serielles Signal“-Konverter

Die Datenübertragung mit der kabelgebundenen „USB nach Serielles Signal“-Konverter ist die zuverlässigste und schnellste Methode um Programme in den Yeti zu übertragen. Dieses Verfahren erlaubt Ihnen auch Rückmeldungen aus dem Yeti mit Hilfe des Seriellen Port-Viewer im Arduino IDE am Bildschirmfenster zu beobachten. Der Knopf für den Seriellen Port-Viewer befindet sich im oberen, rechten Bereich (siehe dazu die Symboldeutung auf der vorherigen Seite. In Ihrem Yeti-Programm können Sie mit dem Kommando „Serial.println() function“ Informationen an ihren PC zurückschicken. Diese Option ist sehr hilfreich bei der Fehlersuche in Programmen.

Beachten Sie bitte dass die Datenübertragungsgeschwindigkeit des Seriellen Port-Viewers übereinstimmt mit der Geschwindigkeit, die Sie im Programm definiert haben. Im Seriellen Port-Viewer kann die Datenübertragungsgeschwindigkeit in der unteren rechten Winkel editiert werden. Die Voreinstellung des Arduino Systems lautet 9600 Baud, aber das Yeti System erlaubt Datentransfers bis zu 57600 Baud.



11.2. Wireless INFRA RED TRANSCEIVER

In der Standardkonfiguration wird das Yeti-System ausgestattet mit dem drahtlosen infrarot Transceiver für die „USB nach Serielles Signal“-Übertragung. Diese Arbeitsweise vereinfacht die Verbindung, weil keine lästige Kabel uns die Arbeit erschweren. Der drahtlose Kommunikation fehlt jedoch momentan die Rückverbindung zum Seriellen Port-Viewer, aber ein verzierter Programmierer kann diese Option in Gang setzen. Siehe im Anhang nach, wie die Option funktioniert, wenn Sie die drahtlose infrarot Transceiver-Option in Betrieb nehmen wollen.



12. KALIBRIERUNG

Um den Yeti Roboter zu einem stabilen Laufschritt zu bewegen benötigen wir ein sauberes und sorgfältig ausgearbeitetes Kalibrierungsprogramm, was man nur beim ersten Gehversuch des Yetis oder nach Änderungen am Körper, Beinen oder Füßen benötigt. Diese Softwarekalibrierung (oder Abgleich) wird benötigt damit der ATmega Prozessor lernt wo die Endpunkte und Mittelpunkte für jeden Servomotor sich befinden. Die Endpunktlage variiert abhängig vom Zusammenbau des Roboters. Werksseitig werden Endpunkte im Yeti-Speicher festgelegt, aber angesichts der Streuung dieser Werten empfehlen wir nachdrücklich nachfolgende Abgleichprozedur.

12.1. Hardware calibration

Ehe man den Softwareteil der Kalibrierung startet ist es wichtig die Mitte der Servomotoren zu eichen. Sobald das Kalibrierungsprogramm im Yeti geladen ist, wird es automatisch jeden Servomotor zentrieren (in die Mittelposition bringen). Deshalb sollten Sie die Schrauben der Servomotorarme entfernen und die Servomotorarme entfernen. Nachdem das Programm die Servomotoren zentriert hat können Sie die Arme wieder befestigen, weil Sie wissen dass das die Servomotoren nun zentriert sind.

Dieses gilt sowohl für die Servomotoren der Vorderseite als für den unteren Antrieb. Nach dieser Zentrierung können Sie die Softwarekalibrierung starten.

12.2. Software calibration

Um das Yeti-System zu kalibrieren haben wir ein Programm entwickelt, das Ihnen die Inbetriebnahme des Yetis vereinfacht. Nach der softwaregestützten Kalibrierung wird Yeti automatisch die von Ihnen gewählten Werte in seinem EEPROM-Speicher ablegen. Ein EEPROM ist ein Speichertyp das abgelegte Wertangaben auch bei Verlust der Stromversorgung aufbewahrt. Da der Yeti ja nicht immer eingeschaltet ist, kann man das als sehr hilfreiche Option betrachten.

Zum Starten der Kalibrierung navigieren Sie im Arduino IDE zum Bereich

example folder->Yeti->Examples->Yeti_Calibration.

Laden Sie (wahlweise kabelgebunden oder drahtlos) dieses Programm in den Yeti. Die Kalibrierung ist für beide Methoden (kabelgebunden oder drahtlos) identisch, aber bei drahtloser Übertragung ist es wichtig das drahtlose USB-Modul zum Rücken des Yeti ausgerichtet zu halten. Falls die Kommunikationsverbindung abreißt während der Yeti gerade Daten sendet, müssen den Yeti erneut starten.

Nach dem Starten des Seriellen Port Viewers sollten Sie am Bildschirm folgenden Text lesen nachdem der Yeti gestartet wird "Yeti calibration procedure started Are you sure you want to calibrate me? Type Y or N and press enter"

In Deutscher Übersetzung:

"Yeti Kalibrierungsprozedur wird gestartet. Sind Sie sicher dass Sie mich kalibrieren wollen? Antworten Sie bitte mit Y oder N und klicken dann die „Enter“-Taste"

Falls Sie diesen Text nicht erhalten, muss:

- entweder der Yeti noch gestartet werden,
- oder ist die Ladeprozedur des Programms fehlgeschlagen (→ Sie können das Hochladen des Programms einfach wiederholen.)
- oder das drahtlose Übertragungsmodul wurde nicht sauber auf den Rücken des Yeti ausgerichtet
- oder das drahtlose Übertragungsmodul wird in der Übertragung durch Fremdlichtquellen gestört.

Zur Beantwortung der Frage klicken Sie ein „Y“ an. Beim ersten Eichpunkt muss der Körper rechts nach vorne beugen.

Wichtig ist nun, dass das linke Bein gerade über dem Boden schweben soll ehe Sie den „S“ anklicken. Sie sollten den Yeti nicht zu weit nach Rechts beugen lassen, denn dort besteht das Risiko dass der Roboter beim Marschieren nach vorne hinfällt. Das linke Bein sollte gerade über dem Boden schweben. Klicken Sie + oder – und drücken Sie die Enter-Taste um die Position zu editieren. Sie werden viele Klicks + und – benötigen weil der Yeti ziemlich empfindlich reagiert.

Um den Roboter auch nur ein wenig zu verbeugen müssen Sie vielleicht 20 bis 30 Mal das Minuszeichen anklicken. Anschließend klicken Sie „S“ und die Enter-Taste wenn Sie soweit fertig sind.

Nach der Rechtsbeugung kehren wir zurück in die Mitte. Das gilt als Kontrolle. Klicken Sie + und dann etwa 20 bis 30 mal die Entertaste bis Yeti wieder gerade steht. Anschließend klicken Sie „S“ und die Enter-Taste wenn der Yeti gerade steht..

Wir machen jetzt weiter mit der Beugung nach links vorne. Dazu wird die gleiche Prozedur angewandt. Jetzt müssen Sie sicherstellen, dass das rechte Bein gerade in der Luft schwebt, aber der Yeti darf nicht zu weit nach links beugen. Anschließend klicken Sie „S“ und die Enter-Taste wenn Sie soweit fertig sind.

Der Yeti wird sich nun wieder gerade hinstellen und dann weitergehen mit dem rechten Fuß nach vorne. Die Rückseite des rechten Beines sollte recht nah an der Mitte des linken Beines heranreichen.

Siehe die Abbildung zur Verdeutlichung.

Für die Stabilität ist es wichtig, dass das rechte Bein nicht zu weit nach vorne ragt. Anschließend klicken Sie „S“ und die Enter-Taste wenn Sie soweit fertig sind.

Nach der Rechtsbeugung kehren wir zurück in die Mitte. Das gilt als Kontrolle. Klicken Sie + und dann etwa 20 bis 30 mal die Entertaste bis Yeti wieder gerade steht. Anschließend klicken Sie „S“ und die Enter-Taste wenn der Yeti gerade steht.



Der letzte Kalibrierungspunkt betrifft der Punkt links vorne. Auch hier wird wiederum die gleiche Prozedur wie für rechts vorne angewandt. Sorgen Sie dafür, dass das Bein nicht zu weit nach vorne ragt, damit beim Gehen die Stabilität erhalten bleibt. Anschließend klicken Sie „S“ und die Enter-Taste wenn Sie soweit fertig sind.

Sobald der Serielle Port Viewer meldet dass die Bewegung links vorne abgeschlossen ist und Sie jetzt fertig sind (Meldung: “Left forward saved, we are done now”) ist die Kalibrierung abgeschlossen. Das Yeti-Programm hat dabei die Endpunktswerte im eigenen Yeti-Speicher gesichert. Sie können jetzt ein willkürliches anderes Programm hochladen. Wenn Sie ein Programm geladen haben, wobei der Yeti marschieren sollte, aber ständig nach vorne kippt, sollten Sie die Kalibrierungsprozedur wiederholen. Wählen Sie dabei die Endpunkte der Beinen etwas kleiner. Dadurch wird die Stabilität beim Gehen verbessert.

13. YETI PROGRAMMIEREN

Und jetzt?

So, unser YETI ist betriebsbereit und funktionsfähig. Das war's wohl, oder?

Vergiß es, die wirkliche Arbeit fängt jetzt erst an!

Erfahrene ARDUINO-Programmierer können nun direkt mit der Programmierung beginnen. Anfänger sollten jedoch lieber zuerst dieses Kapitel lesen, auch wenn das eine oder andere Thema Ihnen vielleicht bekannt vorkommt.

Ein Minicomputer

Wir fangen mal mit einer einfachen Einführung an. Die Hauptplatine des YETIs enthält einen Minicomputer. Einen solchen Kleinstrechner nennt man auch 'Mikrocontroller' und wird in einem IC eingebaut. Ein 'IC' ist ein Integrated Circuit (Integrierter Schaltkreis) und wird manchmal auch kurzweg Chip genannt. Unser 'Mikrocontroller' befindet sich nun in der kleinen schwarzen Box mit den 28 Beinchen, welche die elektrische Anschlüsse zu den roten Lämpchen, zum Lautsprecher, und zu den Antriebsmotoren in den YETI-Muskeln und Infrarot Sender/Empfänger des YETIs versorgen.

Diese Kurzbeschreibung reicht zur Einführung, wird jedoch in den nachfolgenden Kapiteln für die Anfängergruppe Schritt für Schritt erläutert. Beim Lesen werden Sie feststellen, dass die Lektüre langsam anspruchsvoller wird und Sie unbemerkt mit der Programmierung angefangen haben.

Motoren

Der YETI verfügt über zwei Spezialmotoren. Diese Motoren befinden sich zusammen mit einigen Zahnrädern und mit einiger Steuerelektronik jeweils in einem kleinen Gehäuse. Ein solches Antriebsmodul nennt man auch 'Servomotor'. Die Zahnräder bilden ein Getriebe. In der Praxis setzt das Servomotorgetriebe die Drehzahl herab und das Ausgangszahnrad dreht sich erheblich langsamer als der Motor selbst. Tatsächlich kann das Ausgangszahnrad nicht einmal eine volle Umdrehung durchführen und erreicht nur eine Dreivierteldrehung von etwa 220 Grad. Die Herabsetzung der Drehzahl führt jedoch zu einer erheblichen Drehkraft.

Der Mikrocontroller und sein Befehlssatz

Der Befehlssatz für den Mikrocontroller umfaßt 120 einzelne Basiskommandos, die in der englischen Fachsprache der Programmierer auch 'Instructions' genannt werden. Der Mikrocontroller kann jedoch Hunderte von Basiskommandokombinationen verarbeiten. Eine Reihe von Basiskommandos wird ein Programm genannt. Zur Ausführung eines Programms muß der Mikrocontroller diese Reihe in seinen Speicher laden, nimmt dann ein einzelnes Basiskommando aus dem Speicher und führt es aus. Dann nimmt er das nächste Kommando der Reihe und führt auch dieses aus. Der Prozessor wiederholt diese Prozedur ununterbrochen.

Bei einem normalen Computer müssen Sie immer zuerst ein Programm, wie zum Beispiel ein Spielprogramm, starten, damit Sie es benutzen können. Beim Starten kopiert der Computer das Programm von der Festplatte oder CD in den Arbeitsspeicher des Computers. Falls Sie den Computer danach abschalten, verschwindet das Programm aus dem Arbeitsspeicher und Sie müssen das Programm zum Benutzen wiederum starten.

'Flash'-Speicher

In einem Mikrocontroller ist das Programm in einem Spezialspeicher (im sog. 'Flash'-Speicher) abgelegt. Beim Stromausfall und Abschalten bleibt das Programm im 'Flash'-Speicher erhalten. Um ein solchermaßen gespeichertes Programm zu löschen, muß der Mikrocontroller zuerst eingeschaltet werden.

Im Mikrocontroller ist der 'Flash'-Speicher sozusagen der Arbeitsspeicher, der ständig bereit das betriebsbereite Programm enthält und es beim Abschalten auch nicht verliert. Sie können diesen Speicher als Kombination eines Festplattenspeichers und eines Arbeitsspeichers betrachten.

Programmieren

Wie aber erstellen wir jetzt ein Programm?

Programmierer haben sich angewöhnt die Programmentwicklung als das „Schreiben“ eines Programms zu bezeichnen, weil ein Programm zunächst als einfacher Text betrachtet werden kann und manchmal als einfacher Brief oder Kurzgeschichte angesehen werden kann.

Für das Schreiben eines Programms werden wir einen speziellen Texteditor verwenden, der ‘ARDUINO’ genannt wird. Um ein Yeti-Programm zu schreiben könnte man auch irgendeinen anderen Editor verwenden, wie zum Beispiel Notepad oder Microsoft Word, aber wir empfehlen Ihnen dringend die ARDUINO IDE zu verwenden, weil dieses Programm Ihnen als Spezialeditor für die Programmierarbeit mit einer Darstellung in Spezialfarben sehr wirksam unterstützt

Ein Programm besteht aus einer Anzahl einfachen Textzeilen, die natürlich einige Randbedingungen erfüllen muss. Zum Beispiel muss der Text übersetzbar sein in Prozessorkommandos. Ein anderer Name für die einfache Textzeilen ist Quellcode (Englisch: ‘source code’).

Die Randbedingungen und Regeln für die Übersetzung in Prozessorkommandos bilden zusammen die „Programmiersprache“. Und analog an den Wörterbüchern in den verschiedenen menschlichen Sprachen verwenden wir eine spezielle Übersetzersoftware um die simple Textdatei, die unseren „Quelltext“ enthält, in eine ‘Hex’-Datei mit den Prozessorkommandos für den Mikrocontroller zu übersetzen.

Wir können aus verschiedenen Programmiersprachen eine geeignete Sprache wählen, aber die populärste Sprache im Bereich der Mikroprozessorsoftware ist eine Sprache mit einem kurzen Namen: ‘C++’. Das ist auch die Programmiersprache die wir zur Programmierung des Yeti-Software ausgewählt haben.

Das Übersetzen eines Programms

Ein Mikrocontroller kann mit einem in der ‘C++’-Sprache geschriebene Textdatei nichts anfangen. Zum Übersetzen einer in der ‘C++’-Sprache geschriebenen Textquelle in eine Datei mit HEX-Kommandos für den Mikrocontroller benötigen wir ein Übersetzungsprogramm. Ein solches Übersetzungsprogramm ist zunächst ein ganz normales Computerprogramm, das in Fachkreisen üblicherweise ‘Compiler’ genannt wird. Für den YETI benötigen wir daher einen ‘C++’-Compiler.

Somit benötigen wir einen ‘C++’-Compiler um die Yeti-Programme zu compilieren. Ein solcher Compiler ist Bestandteil der Arduino Entwicklungsumgebung IDE. Um nun ein Yeti-Programm ohne Programmübertragung zu compilieren kann man in der IDE einfach die Compile-Taste anklicken. Die Arduino IDE wird dann die ganze schwere Arbeit für Sie erledigen.

Das Laden eines Programms

Wie können wir ein Computerprogramm in den Flashspeicher eines Mikrocontrollers übertragen?

Angenommen der Compiler hat gerade die Übersetzung eines ‘C++’-Programms in die Datei ‘test.hex’ mit den Mikrocontroller-Codezeilen geschrieben.

Als letzten Schritt können Sie dann die Upload-Taste im Arduino IDE betätigen um diese Ergebnisdatei mit Hilfe des Infrarotübertragungsmoduls in den Yeti Mikrocontroller zu übertragen.

Dieser letzte Schritt, in dem das ARDUINO IDE Programm die Ergebnisdatei in den Yeti-Roboter überträgt wird das Hochladen (Englisch: „Upload“) eines Programms genannt. Sobald Sie in der Arduino Entwicklungsumgebung IDE „Upload“ anklicken wird die Arduino IDE automatisch das Programm compilieren. Deshalb ist die „Compile“-Taste in der Praxis nur sinnvoll zur Fehlersuche in den Programmen.

Hauptprogrammstruktur

Eine 'C++'-Hauptprogramm hat eigentlich immer nachfolgende Grundstruktur:

```
int main(void){  
    return 0;  
}
```

'int'	beschreibt den Typ der Hauptfunktion
'main'	ist der Name der Hauptfunktion
'void'	bedeutet 'leer' oder 'unbesetzt'
'return 0'	bedeutet: lieferne eine Zahl Null zurück an die Variable, womit die Funktion 'main' aufgerufen wurde.

Grundsätzlich sollte jede 'C++'-Kommandozeile immer mit einem Semikolon, d.h. ';', abgeschlossen werden. Ausgenommen sind nur Zeilen, die mit einer Funktionsklammer ')' enden.

Der Arduino IDE wird die Programmquelle in fest vorgegebenen Farben darstellen, die nach bestimmten Kategorien zugeordnet werden. Syntax-Teile oder spezifische 'C++'-Funktionen werden in hell orangenen Farben, Objekte in fett orangenen Farben und Kommentare in Grau dargestellt.

Man kann fast überall Kommentar einfügen, nach '//' oder zwischen '/*' und '*/'

```
int main/*Beliebiger Kommentartext*/(void){ /*Irgendein Kommentar*/  
    //Kommentarzeile  
    /* Textzeilen  
     */  
    return 0; // Beliebige Textzeile  
/*  
weiterer Text  
*/  
}
```

#Die ‘C++’-Sprache setzt immer eine und nur eine Hauptfunktion mit dem Namen ‘main’ voraus und daneben ggf. noch eine oder mehrere Zusatzfunktionen mit beliebigen Namen.

#Nachfolgendes Programm enthält nur eine Hauptfunktion mit dem Namen ‘main’, führt jedoch keine Kommandos aus und bildet sozusagen nur ein Gerüst:

```
int main(void){          //Beginn der main-Funktion
    return 0;            //verlasse die Hauptfunktion unter Rück-
    gabe des Werts 0
}
                                //Ende der main-Funktion
```

‘C++’-Kurs

Wie bereits gesagt, wäre es nicht schlecht an dieser Stelle einen kompletten ‘C++’-Programmierkurs ein zu fügen. Auf diesem Bereich sind jedoch viele Handbücher im Handel und ausführliche Webseiten verfügbar, die ‘C++’ sehr detailliert dokumentieren. Deshalb werden wir in diesem Manuskript nur die für den YETI benötigten ‘C++’-Begriffe und Funktionen beschreiben.

Nachdem wir die Grundbegriffe verstanden haben, fangen wir die Programmierung mal vorsichtig an. Wir werden dazu keinen kompletten ‘C++’-Programmierkurs anbieten, aber die auf der CD mitgelieferten Beispielprogramme beschreiben.

OK, es soll also kein ‘C++’-Programmierkurs werden, denn diese gibt es in Hülle und Fülle und Information zum ‘C++’-Programmieren gibt es auch in Büchern und im Internet. Unsere Information bildet jedoch einen guten Grundstock für eine Wiederholung des Wesentlichen und ein Leitfaden für Anfänger mit Praxisbeispielen.

WICHTIG

Der Buchstaben v am Anfang der YETI-Funktionen ist ein Hilfsmittel des Programmierers! Der Buchstaben v (void) am Anfang der YETI-Funktionen bedeutet, dass die Funktion keinen Wert zurückliefert.

YETI schaltet sein rechtes ‘Auge’ ein

#Dieses Programm schaltet das rechte YETI-‘Auge’ ein

```
#include "Servo.h"
#include "Wire.h"
#include "yeti.h" //Lade Definitionen und Funktionen

yeti robot; //create a yeti object

int rightLED = 8; //pin connected to the right led

void setup()
{
    robot.initYeti(); //initialize the yeti program
    pinMode(rightLED, OUTPUT); //set the pin to output
    digitalWrite(rightLED, HIGH); //switch the led on
}

void loop()
{
```

#include “yeti.h”, “Wire.h” and “Servo.h”

An der Stelle wo Sie diese Zeile im Programm aufführen wird der Compiler die spezifizierte Datei oder Dateien einfügen. In diesem Fall werden drei Dateien eingefügt. Die Datei yeti.h enthält alle Funktionen, die den Yeti Arbeitskommandos ausführen lässt. Die beiden anderen Dateien werden benötigt um das Programm yeti.h funktionieren zu lassen.

yeti robot;

Diese Funktion erzeugt ein Yeti-Objekt, dass „robot“ genannt wird. Alle Yeti-Funktionen können aufgerufen werden indem man „robot“ vor der Funktion platziert. Die Erschaffung eines „robot“-Objekts bildet teilweise auch das Gehirn des Yetis.

int rightLED = 8;

Diese Definition legt eine Variable für die rechtsseitige LED, der an Digitalpin 8 des Mikroprozessors angeschlossen wird. Die linksseitige LED wird an Digitalpin 2 des Mikroprozessors angeschlossen.

robot.initYeti();

Dieses Kommando legt die übrigen Basisfunktionen fest, die nicht einfach mit dem „yeti robot;“-Kommando gestartet werden können (wie zum Beispiel das Starten der Servomotoren).

pinMode(rightLED, OUTPUT);

Dieses Kommando schaltet (im Programm mit dem Wert OUTPUT = 8) den Digitalpin 8 auf Ausgabemodus, so dass dieser Mikroprozessoranschluss Objekte wie die angeschlossene LED ansteuern kann. Ohne dieser Festlegung wird der betreffende Digitalpin automatisch in Eingabemodus betrieben, was natürlich zum Ansteuern einer LED weniger geeignet ist.

digitalWrite(rightLED, HIGH);

Dieses Kommando schaltet (im Programm mit dem Wert OUTPUT = 8) die Ausgangsspannung des Digitalpins 8 auf 5V, während LOW einen Wert von 0V entspricht. Indem wir den Anschlusspin auf „hoch“ (Englisch: HIGH) setzen schalten wir die an diesem Anschlusspin angeschlossenen LED ein.

Der „void loop()“ ist in diesem Programm eine leere Funktion, da der Prozessor keine Aufgaben mehrfach erledigen muss. Die Aufgabe besteht hier lediglich darin die LED nur einmalig einzuschalten sodass es keinen Sinn macht das X-Tausendfach Mal pro Sekunde zu tun.

Der Programmteil innerhalb den geschweiften ‘for’-Klammern ‘{’ und ‘}’ heißt ‘Schleife’ oder ‘Loop’.

```
for(...){  
}
```

Weiteres Beispiel:

```
#include <Servo.h>  
#include <Wire.h>  
#include <yeti.h>          //include the yeti library  
  
yeti robot;                //create a yeti object  
int rightLED = 8;           //pin connected to the right led  
  
void setup()  
{  
    robot.initYeti();         //initialize the yeti program  
    pinMode(rightLED, OUTPUT); //set the pin to output  
    digitalWrite(rightLED, HIGH); //switch the led on  
    robot.beep(5000, 1000);    //make the beeper beep for 1  
    second  
    delay(1000);  
    robot.beep(2200, 1000);    //make the beeper sound for ano-  
    other second  
}  
  
void loop()  
{  
}
```

robot.beep(5000, 1000);

Diese Funktion aus der Yeti-Bibliothek sorgt dafür, dass der auf der Yeti-Hauptcontroller Leiterplatte eingebaute Piepser während 1 Sekunde, das heißt 1000 Millisekunden, einen Ton von 5000 Hertz erzeugt. Diese Funktion ist nicht-blockierend, das heißt der Prozessor beauftragt diese Aufgabe sofort und arbeitet weiter, während die Tonerzeugung im Hintergrund abläuft. Wir müssen deshalb aufpassen wenn wir zwei Töne mit unterschiedlicher Frequenz direkt nacheinander erzeugen wollen.

delay(1000);

In der letzten Funktion habe wir das Merkmal „nicht-blockierend“ erwähnt und jetzt haben wir ein Problem falls wir zwei Töne direkt nacheinander mit unterschiedlicher Frequenz erzeugen wollen. Für diesen Zweck benötigen wir eine Funktion „Verzögerungszeit“ (Englisch „delay“), die das Programm für die spezifizierte Zeitlänge anhält. In unserem Beispiel beträgt die Verzögerung 1000 Millisekunden oder 1 Sekunde. Die „delay“-Funktion stoppt das Programm bis die Wartezeit vorbei ist. Die im Hintergrund ablaufende Prozesse werden dabei nicht angehalten, sodass der Piepser einfach weiter tönt und das gilt auch während die „delay“-Funktion aktiv ist.

Lass den Yeti einen Freudentanz machen

```
#include <Servo.h>
#include <Wire.h>
#include <yeti.h>           //include the yeti library

yeti robot;                //create a yeti object

void setup()
{
    robot.initYeti();        //initialize the yeti program

    robot.moveForwardX(0);
    delay(300);
    robot.moveBody(3);       //3 stands for swing right
    delay(400);
    robot.moveBody(4);       //4 stands for swing left
    delay(150);
    robot.moveBody(5);       //5 stands for level
}

void loop()
{
```

robot.moveForwardX(0);

Diese Funktion wird den Roboter zum Vorwärtsgehen veranlassen. In diesem Fall wählen wir 0 Schritte weil wir lediglich die Servomotoren zentrieren wollen ehe wir den Yeti von einer Seite zur anderen marschieren lassen. Falls Sie einen anderen Wert eintragen, wie zum Beispiel 2, wird der Yeti 2 Schritte vorwärtsgehen und sich dann im Gleichgewicht aufrichten.

Robot.moveBody(3);

Diese Funktion wird den Roboter in eine bestimmte Richtung laufen oder sich aufrichten lassen (wobei nur der vordere Servomotor aktiviert wird! Falls Sie beide Servomotoren zentrieren wollen müssen Sie robot.moveForwardX(0) benutzen). Falls Sie als Parameter 3 zwischen den Klammern setzen, wendet sich der Yeti nach Rechts. Wenn Sie anstelle einer 3 eine 4 einsetzen, wendet sich Yeti nach Links. Eine 5 gibt das Kommando sich senkrecht auszubalancieren beziehungsweise aufzurichten indem der Servomotor zentriert wird. Wenn Sie Yeti nur gehen lassen wollen sollten Sie dazu die Funktion "moveForwardX()" benutzen.

Lasse den Yeti spazieren!

Folgendes Programm wird den Yeti 10 Vorwärtsschritte machen lassen.

```
#include <Servo.h>
#include <Wire.h>
#include <yeti.h>           //include the yeti library

yeti robot;                //create a yeti object

void setup()
{
    robot.initYeti();        //initialize the yeti program
    robot.moveForwardX(10);  //make Yeti walk 10 steps forward
}

void loop()
{
}

robot.moveForwardX(10);
```

14. ERWEITERUNGEN

Erweiterungsmodule (Kits)

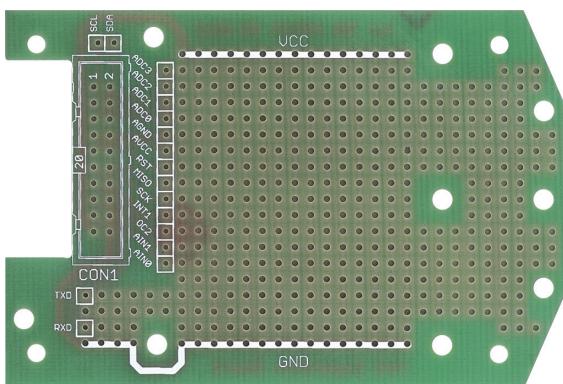
Sie können YETI erweitern mit zusätzlichen (nicht im Lieferumfang enthaltenen) Baugruppen, welche die Leistungsfähigkeit des Roboters erheblich steigern. So können Sie YETI mit einem Ultraschall-Sender/Empfänger ausstatten, der ihm ermöglicht mit Schallwellenechos die Distanz zu entfernten Gegenständen zu messen und diesen beim Gehen auszuweichen.

14.1.2. YETI Experimentiermodul YT-EXP1

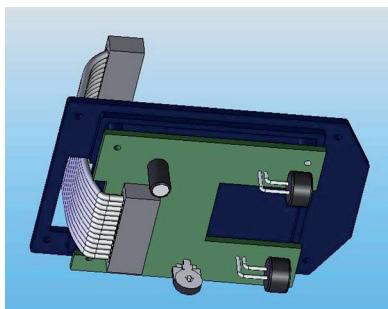
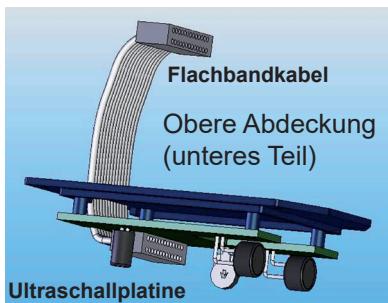
Im Experimentiermodul können Sie einen eigenen Entwurf einer Elektronikbaugruppe aufbauen und dieses Modul auf unterschiedlichste Art mit dem Mikrocontroller ansteuern.

14.1.2. Teilliste Experimentiermodul YT-EXP1

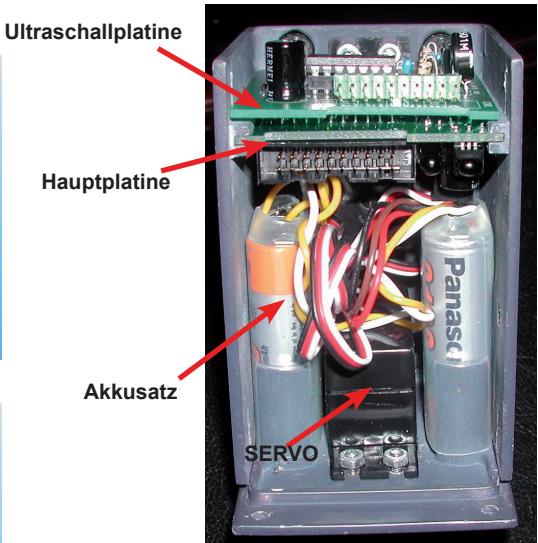
PCB-DSP	YETI Experimentierplatine
CON1-PCB	Platinenstecker, male, 20 Pins, für Flachbandkabel
CON1-FC (2 St.)	Flachbandstecker, 20 Pins
F1	Flachbandkabel, 20 Adern, 10cm



14.1.3. Montage Erweiterungskits



Montage Ultraschallplatine



Montage
Display Platine



Montage Experimentierplatine



14.2. YETI Display Erweiterungsmodul YT-DSP2

Allgemeine Beschreibung

Das Display Erweiterungsmodul enthält 4 Stück 8-Segmentanzeigeböcke. Ein ebenfalls im Erweiterungsmodul vorhandener 24 Pin I2C Treiberchip erledigt die Ansteuerung des Displays.

Der YETI Mikrocontroller steuert seinerseits den I2C Treiberchip. I2C (buchstabiert als I-Quadrat-C) ist eine offizielle Standardmethode für die gegenseitige Kommunikation zwischen Elektronikmodule mit nur zwei Anschlussadern SCL (Serial Clock) und SDA (Serial Data). Wir benötigen für dieses Kommunikationsprotokoll deshalb nur 2 der 20 Adern des Flachbandkabels.

Das Display bietet eine Anzeigeoption für diverse Nachrichten oder Daten. Die einfache Ansteuerung des I2C Chips erlaubt die Darstellung vielerlei selbstdefinierten Symbole auf dem Display.

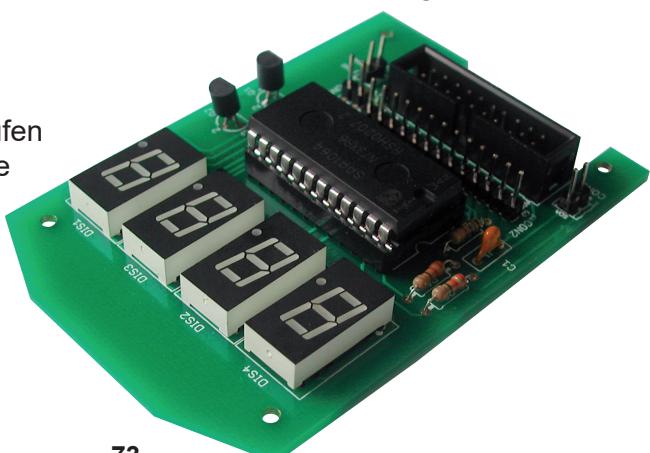
Hardware Beschreibung

Das Displaymodul enthält vier 8-Segment Displays, einen I2C Displaychip und einige Hilfskomponenten.

Der Chip kommuniziert über das I2C-Interface mit einem Steuerungsmodul, in der Regel einem Mikrokontroller. Der Chip liefert selbstständig alle Steuerungssignale zur Ansteuerung der Displays.

Der Anwender braucht sich um diese Details nicht mehr zu kümmern und muss nur festlegen welches Symbol in jedem einzelnen Zeichen angezeigt werden soll. Dazu wird selbstverständlich der I2C-Bus verwendet.

Auch die Helligkeit der Anzeige lässt sich in 8 Stufen regeln. Zudem verfügt jede Displayeinheit über 8 Anschlüsse für die Einzelsegmente: sieben Datensignale für die Einzelsegmente und ein Dezimalpunkt.



Jedes Segment besteht aus einer Leuchtdiode oder LED. Außerdem wird noch ein gemeinsamer Pol für den Stromversorgungsanschluß der Leuchtdioden in jedem Displayelement benötigt. Der Hersteller hat dazu vier Segmente jeweils an einem Anschluss verknüpft und die Anschlüsse 3 und 14 intern zu einen gemeinsamen Stromversorgungsanschluß zusammengelegt.

Zur Ansteuerung der vier Displays mit jeweils $8+2=10$ Anschlässen würde man theoretisch ein 40-poliges IC benötigen. Diese Verbindungsfülle können wir jedoch mit dem Multiplexertrick beseitigen. Der Chip benötigt dazu 2 Sätze 8-Segment-Anschlüsse: P1-P8 und P9-P16.

Zunächst betrachten wir den Satz P1-P8, die sowohl mit Display 1 als mit Display 2 verbunden sind. Falls wir nun eine bestimmten Bitkombination auf P1-P8 schalten, dann werden beide angeschlossenen Displays das gleiche Symbol anzeigen. Der Trick besteht nun darin, mit den Transistoren Q1 und Q2 Display 1 ein- und Display 2 abzuschalten. Mit dieser Schaltung wird zuerst nur Display 1 das Symbol anzeigen. Nun jedoch schalten wir Display 1 mit Transistor Q2 aus, erzeugen auf den Datenleitungen P1-P8 eine neue Bitkombination für ein weiteres Symbol und schalten Display 2 mit Transistor Q1 ein. Das zweite Symbol wird nun auf Display 2 angezeigt. Dieses Multiplexerprinzip wird auch für Display 3 und 4 angewandt. Falls wir nun den Multiplexertrick mit höheren Schaltgeschwindigkeit betreiben, wird das menschliche Auge nicht registrieren, daß die Displays eigentlich die Hälfte der Zeit gar nicht aufleuchten.

Wir können zwei der vier Displays auch ohne Multiplexermodul ansteuern. Diese beiden Displays sind ständig eingeschaltet, falls ein Display (z.B. Display 1) auf P1-P8 und das zweite Display (Display 3) auf P9-P16 angeschlossen wird. Die Beispielsoftware basiert auf die Randbedingung, daß die Displays folgendermaßen geschaltet werden:

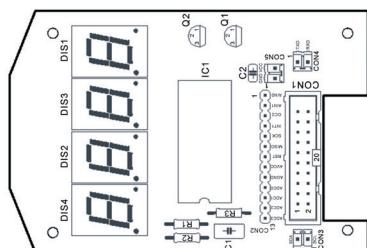
Displayschaltung:

Display 4 Display 2 Display 3 Display 1

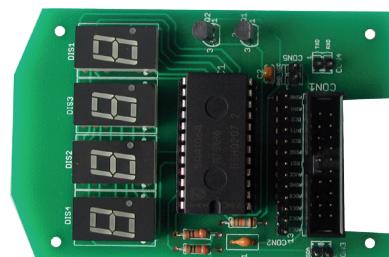
Diese Schaltung basiert auf die grundlegende Idee, daß zum Betrieb zweier Displays (Display 1 und 3) im statischen Modus diese Module nebeneinander stehen sollten..

14.2.1. Teilleiste YETI Displaysatz YT-DSP2 KIT

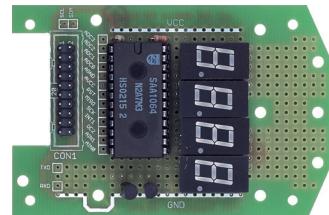
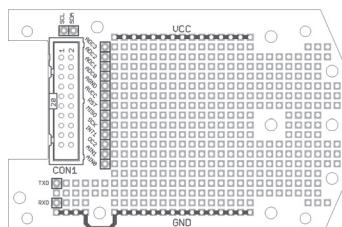
PCB-DSP	YETI Display platine
R1	330R (orange, orange, braun, gold)
R2	330R (orange, orange, braun, gold)
R3	18K (braun, grau, orange, gold)
C1	2,7nF (Aufdruck: 272)
C2	100nF (Aufdruck: 104)
Q1	BC547B/C (auf richtige Polung achten)
Q2	BC547B/C (auf richtige Polung achten)
D1	8-Segment display, common anode
D2	8-Segment display, common anode
D3	8-Segment display, common anode
D4	8-Segment display, common anode
IC1	SAA1064 (auf richtige Polung achten)
S1	IC-Fuß, 24-pins, 600mil (auf richtige Polung achten)
CON2	Kontaktleiste 13-polig, Platinenmontage
CON, 3, 4 und 5 (3 St.)	Kontaktleiste 2-polig, Platinenmontage
CON1-PCB	Platinenstecker, male, 20 polig, für Flachbandkabel
CON1-FC (2 St.)	Flachbandstecker, 20-polig
F1	Flachbandkabel, 20 Adern, 10cm

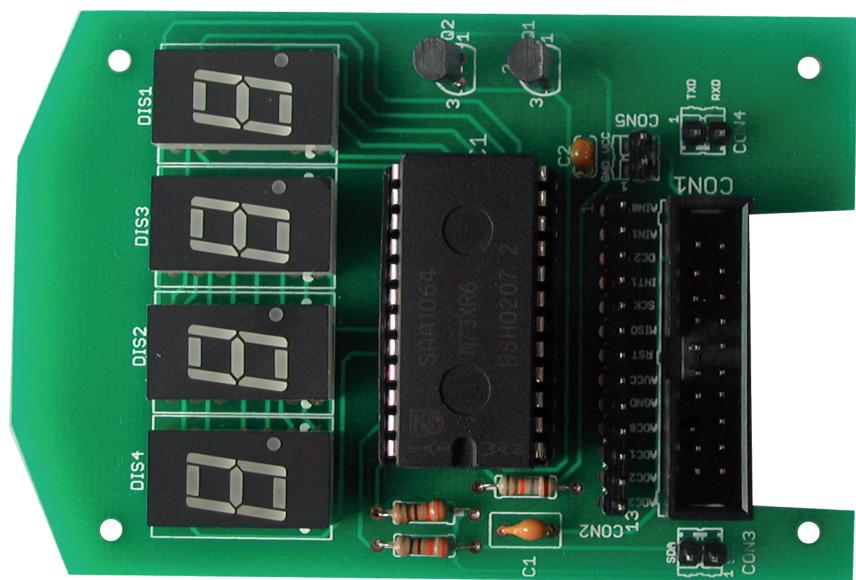
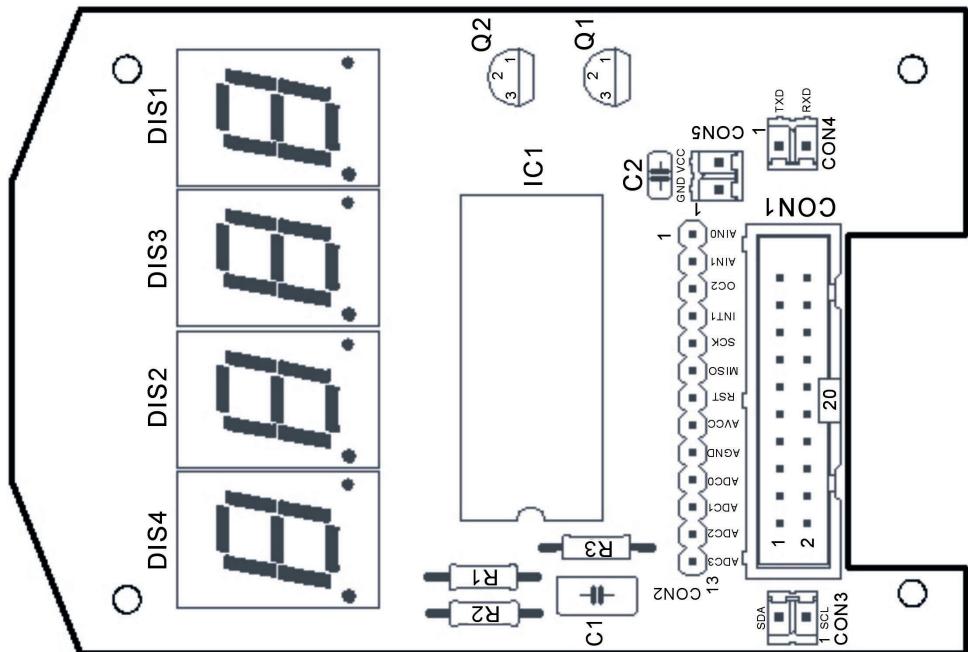


Display auf original Display Platine YT-DSP2 KIT

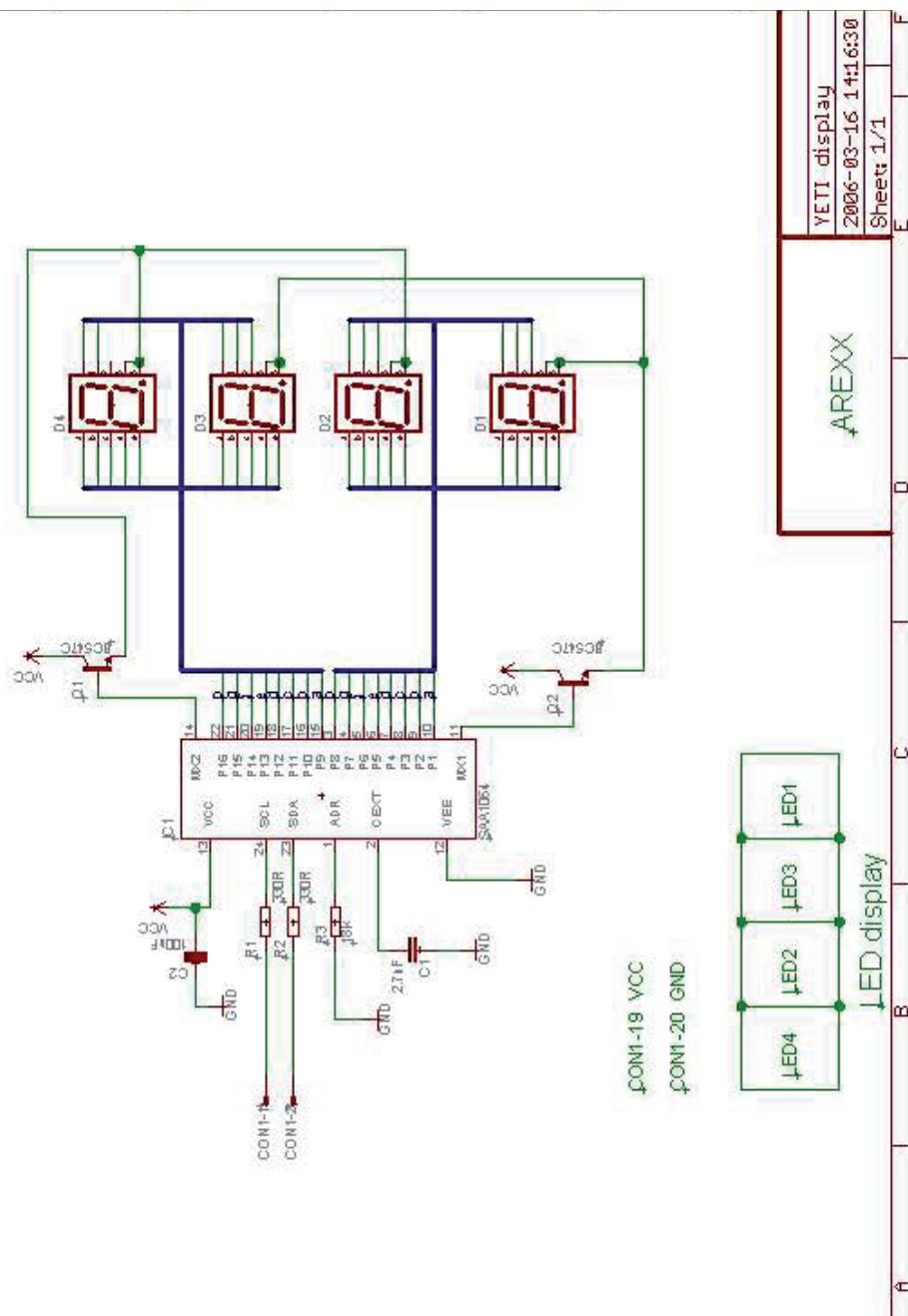


Display auf Experimentierplatine





14.2.2. Display Modul Schaltplan



14.3. YETI Ultraschall Erweiterungsmodul YT-ULT3

Der Ultraschall Erweiterungssatz enthält einen Ultraschallsender und einen Ultraschallempfänger. Ultraschall bedeutet: Schallwellen mit einer so hohen Frequenz, dass Menschen diesen Ton nicht hören können.

Fledermäuse senden zum Beispiel Ultraschallwellen aus, damit sie im Dunkeln fliegen und jagen können. Men nennt dieses Verfahren Echo-Ortung. Gegenstände reflektieren die Schallwellen, die dann im Ohr empfangen werden.

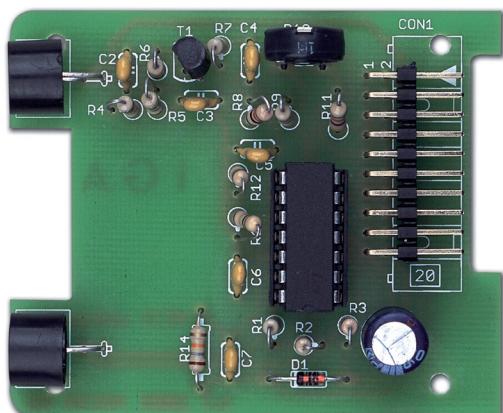
Beim YETI wird ein Mikrofon zum Schallwellenempfang benutzt.

Der Sender erzeugt Schallwellen mit einer Frequenz von 40.000Hz. Der Empfänger registriert die eventuell von einem Gegenstand reflektierten Signale und auch die Zeitverzögerung zwischen Sende- und Empfangsimpuls.

Aus der Zeitverzögerung zwischen Sende- und Empfangsimpuls können wir den Abstand zum Gegenstand berechnen. Das Ultraschallmodul verwandelt die Zeitverzögerung in eine elektrische Spannung. Das Flachbandkabel führt diese elektrische Spannung zur Analog/Digitalwandlereingang des Mikroprozessors im YETI.

Ein im YETI verfügbares Programm kann die Spannung messen und aufgrund der Daten eine Aktion ausführen.

Im YETI ist zur Aufnahme des Ultraschallmoduls bereits ein Platz am Deckel zum Innenraum des Kopfes reserviert. Hinter den beiden "Augenbrauen"-Öffnungen in der Stirn befinden sich der Sender, beziehungsweise Empfänger.



14.3.1. Hardware Beschreibung

Das Ultraschallmodul besteht aus 5 Komponenten:

1. Der Sender
2. Der Empfänger
3. Der Empfängerverstärker
4. Eine konstante Referenzspannung
5. Eine variable Referenzspannung

Der Mikrocontroller generiert das benötigte Ultraschallsignal.

Ein Senderlautsprecher (TX) erzeugt das akustische Signal und ein Empfängermikrofon (RX) empfängt die reflektierte Schallwelle. Der Empfängerverstärker erhöht den schwachen Empfangspegel, wobei der Verstärkungsfaktor manuell mit R10 einstellbar ist. Die konstante Referenzspannung liefert eine Spannung, die auf genau 50% der Stromversorgungsspannung eingestellt und zur Ansteuerung des Senders und zur Erzeugung der variablen Referenzspannung benötigt wird. Ein spezieller Regelkreis lässt mit Hilfe der variablen Referenzspannung den Mikrocontroller je nach Entfernung der Reflexionsquelle nach jeder Schallwelle immer genauer zuhören. Diese Regelung ist sehr wichtig, denn bei großer Entfernung der Reflexionsquellen nimmt der Empfangspegel rapide ab.

Der Mikrocontroller erzeugt die benötigten Ultraschallwellen und führt diese Signale über CON1-13 in das Ultraschallmodul. Das reflektierte und im Ultraschallmikrofon empfangene Signal wird über CON1-6 in den Mikrocontroller zurückgeführt. Anschluß CON1-15 liefert nach jedem Senderimpuls ein abnehmendes Spannungssignal an den Mikrocontroller.

Das im Mikrocontroller erzeugte ‘Ultraschall’-Signal ist zunächst natürlich keine Schallwelle, sondern ein elektrisches Signal. Genau genommen darf man diese Signale erst ‘Ultraschallwellen’ nennen, nachdem der Lautsprecher das Signal in Schallwellen verwandelt hat.

Dieses im Mikrocontroller generierte ‘Ultraschall’-Signal wird zunächst über CON1-13 und über Widerstand R3 in den Sender eingespeist. Der Sender besteht aus 2 (von 4) Verstärker, die einzelnes IC (Integrated Circuit = Chip) mit dem Namen IC1 zur Verfügung stellt. Fachleute nennen diese Verstärker oft Opamp, d.h. Operational Amplifier oder Operationsverstärker. Es handelt sich dabei um Differenzverstärker mit zwei Eingängen: einen Plus- und einen Minuseingang. Wie der Name Differenzverstärker andeutet, verstärkt das Verstärkermodul die Differenzspannung zwischen den beiden Eingängen.

Das Ultraschallsignal erreicht nun einen Eingang eines der beiden Opamps. Die übrigen Eingänge der beiden Opamps werden mit einer festen Referenzspannung in Höhe von 50% der Stromversorgungsspannung verbunden. Beide Opamps haben zur Aufgabe den Lautsprecher mit Energie zu versorgen und das vom Mikrocontroller geringfügig geschwächt und verzerrt aufbereitete Signal zu regenerieren. Das Ultraschallmikrofon (RX) empfängt das reflektierte Signal und verwandelt es in ein elektrisches Signal. Nach einem einfachen Filter erreicht das Signal den Empfangsverstärker Opamp IC1B, dessen Verstärkungsfaktor mit einem Regelwiderstand einstellbar ist.

Nach jedem Senderimpuls steigt die Spannung auf CON1-15 sprunghaft auf 50% der Stromversorgungsspannung VCC. Dazu wird das elektrische ‘Ultraschallsignal’ nicht nur über R3 an den Verstärker, aber auch über Diode D1 an Kondensator C7 weitergeleitet. Nachdem diese Spannung C7 zunächst sofort aufgeladen hat, entlädt der Kondensator sich langsam über Widerstand R14.

Der Mikrocontroller enthält einen analogen Komparator, d.h. einen Vergleichsmodul, das zwei elektrische Spannungspegel vergleicht. Sowohl der Empfangspegel als auch diese Entladungsspannung werden nun auf CON1-6 beziehungsweise CON1-15 dem analogen Komparator zum Vergleich angeboten. Der Mikrocontroller vergleicht den Empfangspegel und die Entladungsspannung. Falls der Empfangspegel die Entladungsspannung übersteigt, betrachtet der Mikrocontroller den Empfangspegel als ein zuverlässiges Empfangssignal. Unmittelbar nach dem Senderimpuls muß ein Empfangspegel daher schon relativ hoch sein, um als reflektiertes Signal akzeptiert zu werden. Je länger der Senderimpuls jedoch zurückliegt, desto geringer darf der Empfangspegel sein, um als reflektiertes Signal akzeptiert zu werden.

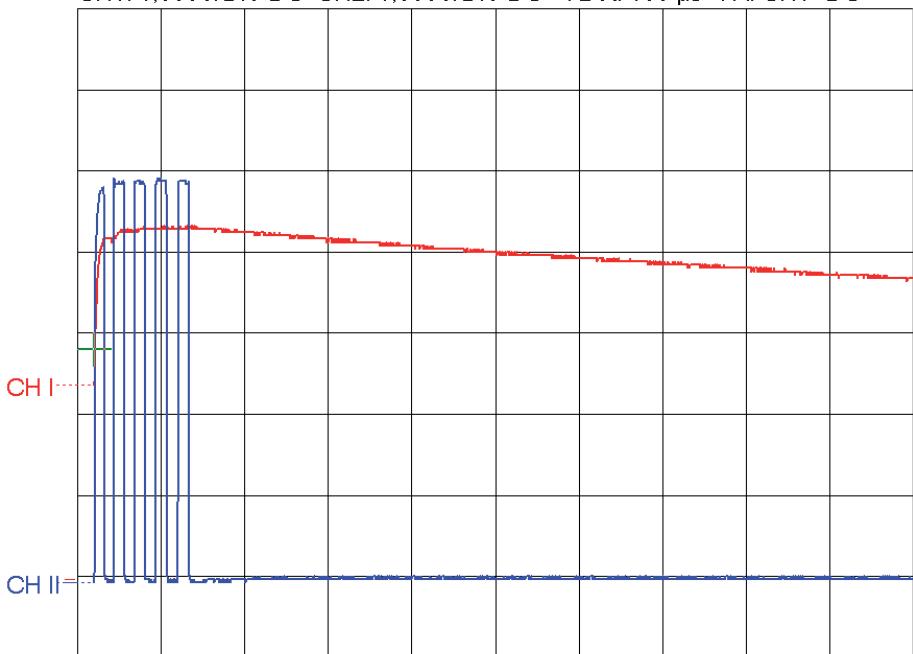
14.3.2. Abgleich Ultraschall-Zusatzmodul

Die Ultraschallplatine ist in YETIs Kopf eingebaut. Allerdings wird das Ultraschallmodul in dieser Position durch folgende Dingen negativ beeinflußt:

1. Unerwünschte Reflektionen des Ultraschalltons innerhalb des YETI Chassis.
2. Unerwünschte Reflektionen des Ultraschalltons durch die Aussenhülle von YETI.

In folgender Grafik sehen Sie ein Oszillosrogramm der Referenzspannung (rot, gemessen an CON1-15) und des Ultraschallsignals des Senders (blau, CON1-13, 5 Impulse von etwa 4.5V, vom Mikrocontroller erzeugt):

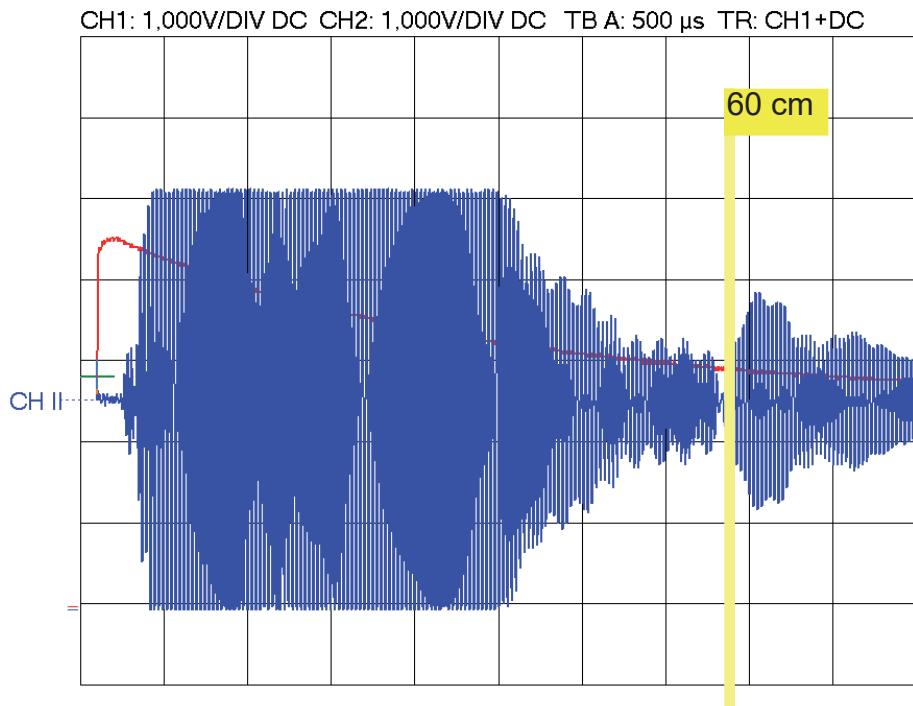
CH1: 1,000V/DIV DC CH2: 1,000V/DIV DC TB A: 100 μ s TR: CH1+DC



Das erste Signal (CON1-13) zeigt kurz 5 Pulse von etwa 4.5 Volt.

Das Signal kommt sofort vom Mikroprozessor und geht zu der Ultraschallplatine.

Die rote Kurve an CON1-15 ist eine Entladekurve, welche als Referenzsignal dient. Diese wird auf der Ultraschallplatine erzeugt und zurück zum Mikrocontroller geleitet. Erzeugt der Empfängerteil des Ultraschallsensors eine Spannung die höher ist als das Referenzsignal, wird dies vom Mikrocontroller als Reflektion an einem Objekt erkannt.



Nun eine weitere Messung, diesmal jedoch neben der Referenzspannung (rot) auch das empfangene Echo (blau, CON1-6 - wird zum Mikrocontroller geleitet).

Wie man leicht erkennen kann gibt es zu Beginn eine sehr starke Fehlmessung! Das Objekt welches das Signal eigentlich reflektieren sollte befindet sich bei der Markierung in 60 cm Entfernung. Dies wird durch Schallreflektionen innerhalb von YETI verursacht.

Im Folgenden werden wir beschreiben wie man dies stark verbessern kann!

Wie man aus der Beschreibung des Messverfahrens unseres Ultraschallsensors herauslesen kann, reagiert der Empfänger sehr empfindlich auf jegliche Arten von Reflektion des Ultraschallsignals. So auch auf Reflektionen innerhalb von YETI!

Um diesen zu verringern, müssen wir das Innere von YETI unterhalb des Ultraschallsensors vollständig mit Watte auffüllen (s. Abbildung 1)!

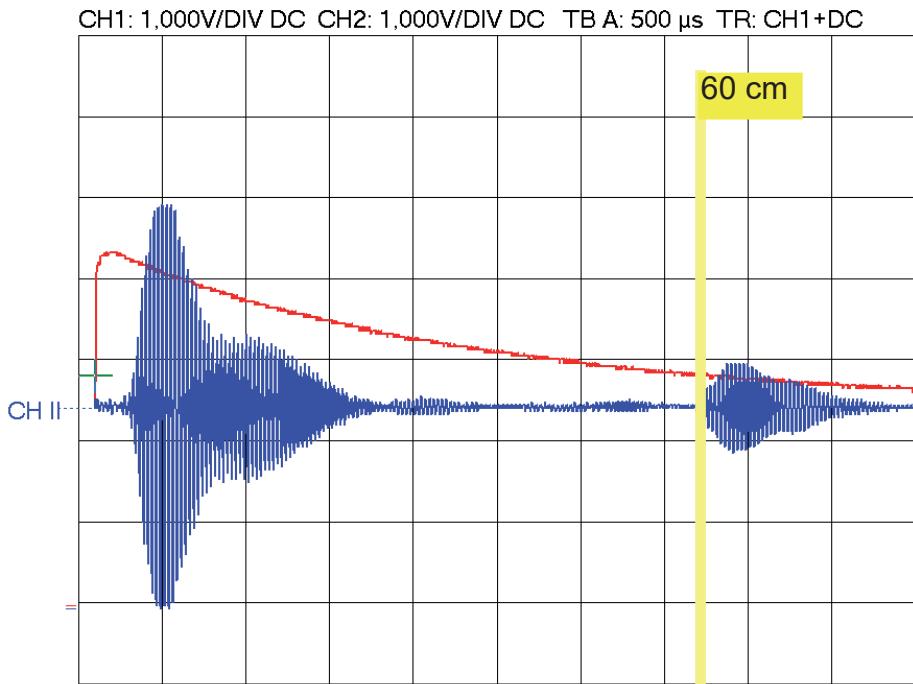
Abbildung 1. YETI vollständig mit Watte ausgefüllt.



Schritt 1.

Abdämmen der unerwünschten Schallreflektionen innerhalb von YETI.

Folgendes Oszilloskopogramm zeigt die neue Situation
(YETI vollständig mit Watte aufgefüllt):



Es ist leicht zu erkennen das es nun deutlich geringere unerwünschte Reflektionen gibt!

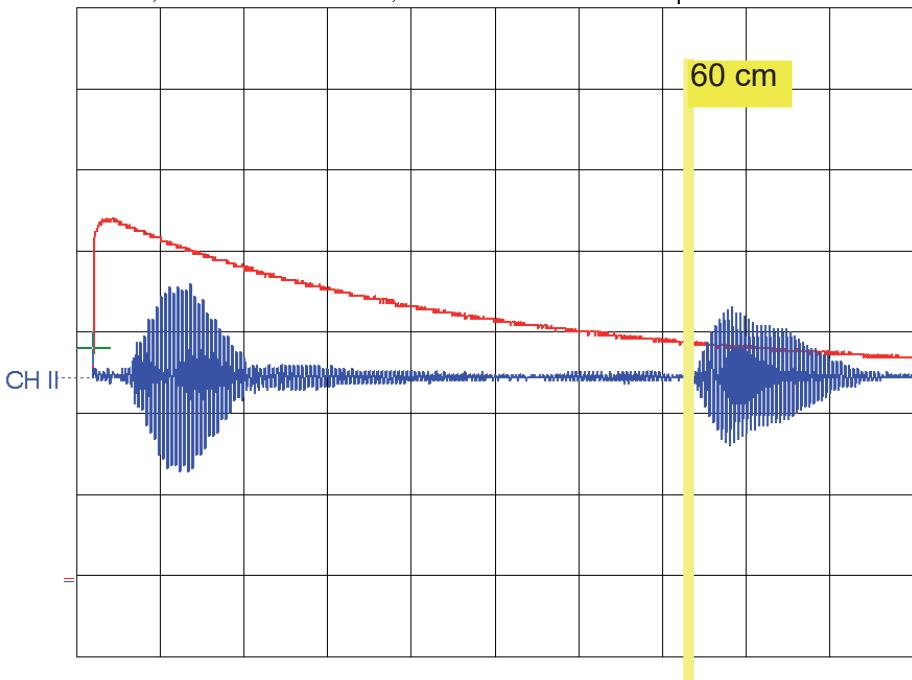
Doch noch immer gibt es einen kleinen Bereich zu Beginn der Messung, der höher als das Referenzsignal liegt. Dies wird durch Reflektionen direkt vor YETIs Kopf verursacht.



Schritt 2.

Dies lässt sich mit einem weiteren Stück Watte (oder ähnlichem, evtl. etwas festeres Material verwenden!) vermeiden, welches so ausserhalb von YETI angebracht wird, wie es auf dem nebenstehenden Bild zu sehen ist!

CH1: 1,000V/DIV DC CH2: 1,000V/DIV DC TB A: 500 μ s TR: CH1+DC



Auf dem Oszilloskopogramm erkennt man das Ergebnis unserer Bemühungen: Alle zu Beginn der Messung empfangenen Signale bleiben unterhalb der Referenzspannung! Erst das tatsächlich zu detektierende Objekt erzeugt eine höhere Spannung am Empfängerausgang!

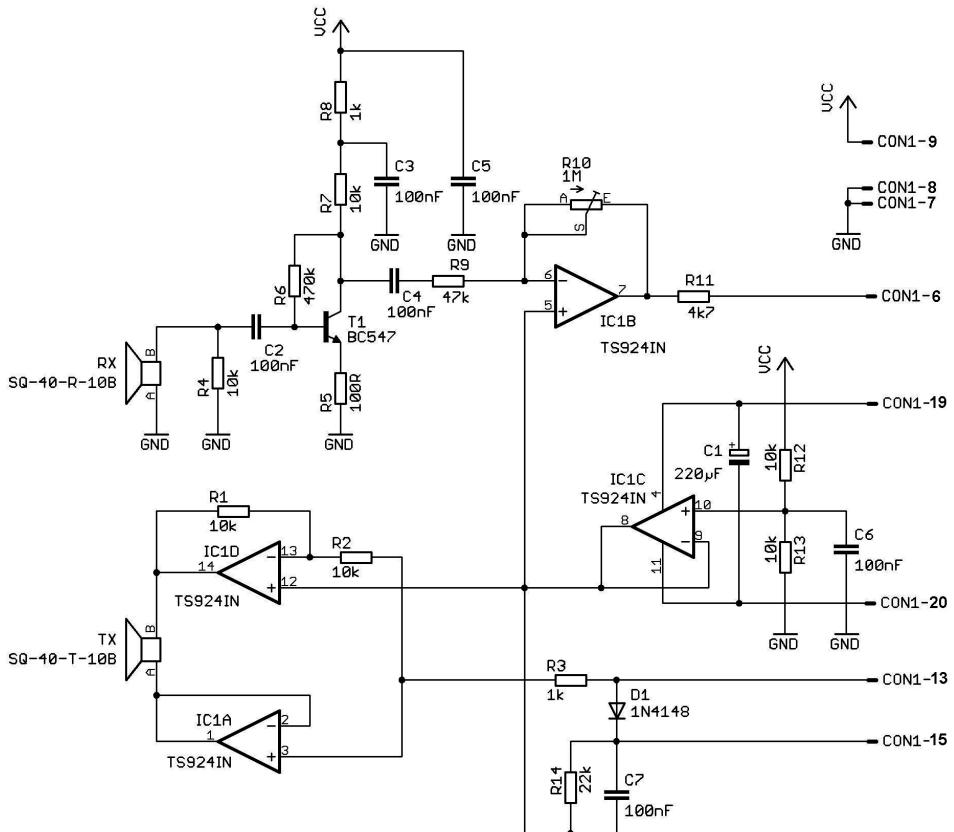
Die Zeitspanne zwischen den 5 übertragenen Ultraschallpulsen und dem Reflektierten Signal wird vom Mikrocontroller gemessen und in den Abstand umgerechnet. Diesen Abstand kann man dann z.B. auf dem YETI Display anzeigen!

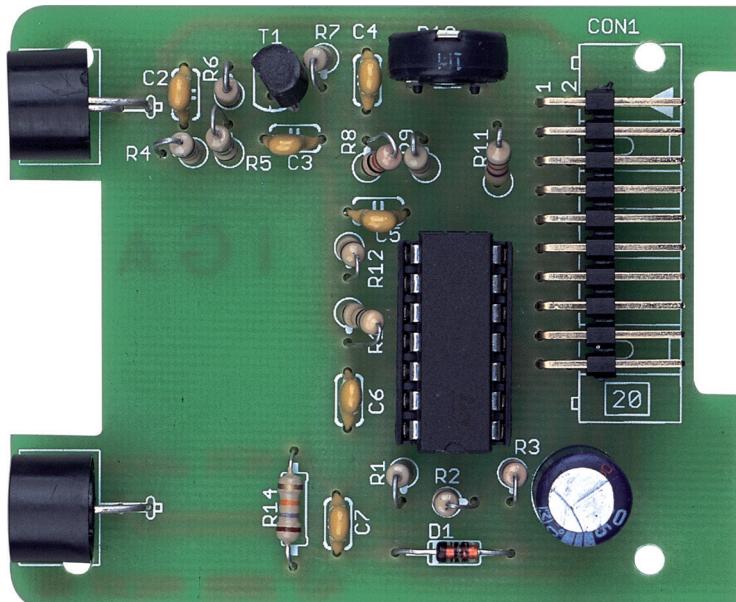
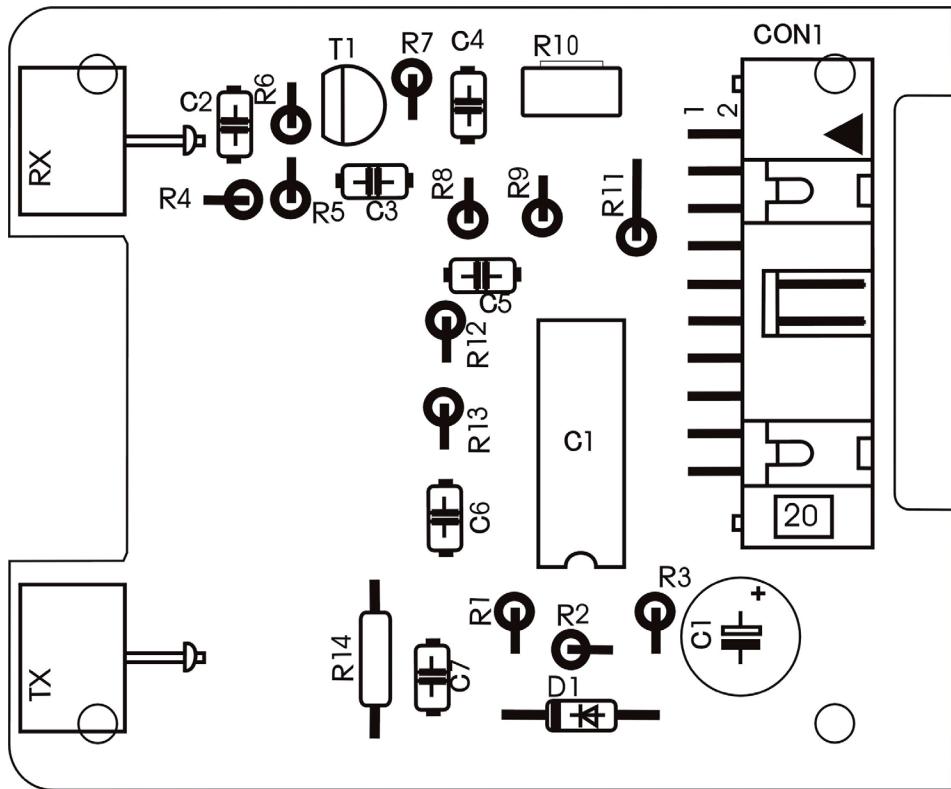
Es ist sehr wichtig, dass Sie beim Abgleich (14.3.2) dafür sorgen das die Reflektion zu Beginn der Messung unterhalb der Referenzspannung bleiben.

14.3.3. Teileliste YETI Ultrasschallsatz YT-ULT3

PCB-UTS	Ultraschall Platine	
R1	10K	(braun, schwarz, orange, gold)
R2	10K	(braun, schwarz, orange, gold)
R3	1K	(braun, schwarz, rot, gold)
R4	10K	(braun, schwarz, orange, gold)
R5	100R	(braun, schwarz, braun, gold)
R6	470K	(gelb, violett, gelb, gold)
R7	10K	(braun, schwarz, orange, gold)
R8	1K	(braun, schwarz, rot, gold)
R9	47K	(gelb, violett, orange, gold)
R10 TRIMMER	1M	Trimmer
R11	4K7	(gelb, violett, rot, gold)
R12	10K	(braun, schwarz, orange, gold)
R13	10K	(braun, schwarz, orange, gold)
R14	22K	(rot, rot, orange, gold)
C1	220uF	(auf richtige Polung achten)
C2	100nF	(Aufdruck: 104)
C3	100nF	(Aufdruck: 104)
C4	100nF	(Aufdruck: 104)
C5	100nF	(Aufdruck: 104)
C6	100nF	(Aufdruck: 104)
C7	100nF	(Aufdruck: 104)
IC1	TS924IN (Quad Opamp)	(auf richtige Polung achten)
S1	IC-Sockel, 14-Pins	(auf richtige Polung achten)
TX	400ST100	(Ultraschall Sender)
RX	400SR100	(Ultraschall Empfänger)
T1	BC547B oder BC547C	(auf richtige Polung achten)
D1	1N4148	(auf richtige Polung achten)
CON1-PCB	PCB Stecker, male, 20 Pins, für Flachbandkabel	
CON1-FC (2 St.)	Flachbandkabelstecker, 20 Pins	
F1	Flachbandkabel, 20-adrig, etwa10cm	

14.3.4. Schaltbild YETI Ultraschallsatz





14.11. YETI Programmierung-Erweiterungsmodul YT-PRG

Wir haben das Programmierung-Erweiterungsmodul entworfen um den Arduino Yeti mit einer kabelgebundenen Programmieroption auszustatten. Obwohl diese Option nicht unbedingt aussieht wie ein hilfreiche Zusatzoption bietet die Kabelverbindung tatsächlich erheblich mehr Komfort als die drahtlose Datenübertragung. Die Kabelgebundene Kommunikation verläuft bis zu 24-mal schneller als die drahtlose Datenübertragung, die außerdem empfindlich reagiert auf störende Lichtquellen. Außerdem erlaubt die Kabelverbindung eine beschleunigte Rückmeldung vom Yeti-Programm zur seriellen Datenmonitor.

Das Programmierung-Erweiterungsmodul bietet außerdem viel Platz für experimentelle Schaltungen. Für Sie stehen 389 freie Anschlussfelder zur Verfügung, worauf Sie experimentelle Baugruppen befestigen oder mit eigenen Entwürfen bestücken.

14.11.2 Stückliste YETI Programmierung-Erweiterungsmodul

PCB-PRG Leiterplatte für das YETI Programmierung-Erweiterungsmodul
R1 10K

C1 100nF

CON1-PCB PCB Steckverbinder, männlich, 20 Pins, für Flachbandkabel

CON2 Pin Anschlussleiste 13 – Pins Leiterplattenmontage

CON4, 5,6 Pin Anschlussleiste 2 – Pins Leiterplattenmontage

CON6 Pin Anschlussleiste 10 – Pins Leiterplattenmontage

CON7 Pin Anschlussleiste 7 –Pins Leiterplattenmontage

CON1-FC (2 Stück) Flachbandkabel Steckverbinder, 20-pins

F1 Flachbandkabel, 20-adrig, 10cm

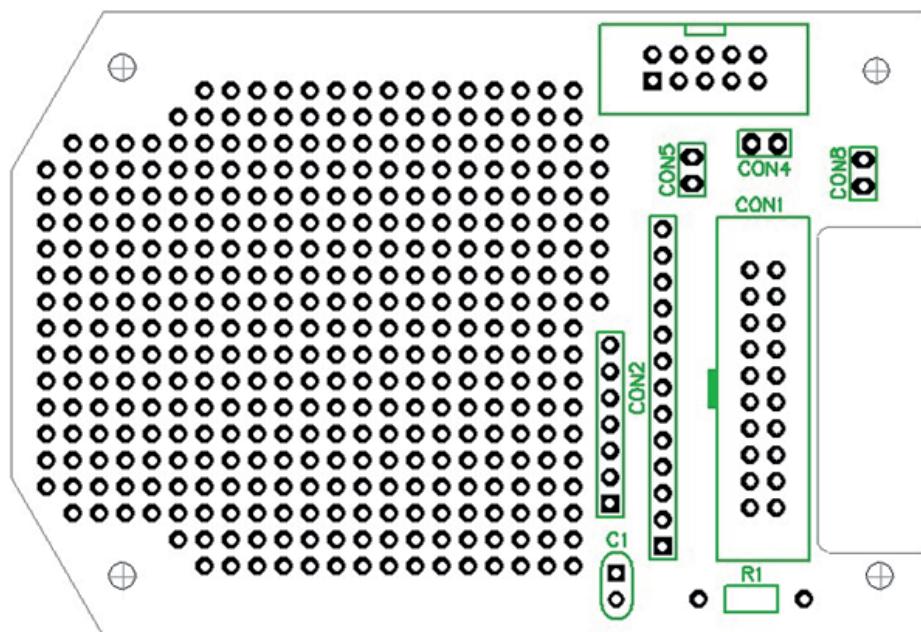
14.11.3 Betrieb des Programmierung-Erweiterungsmoduls

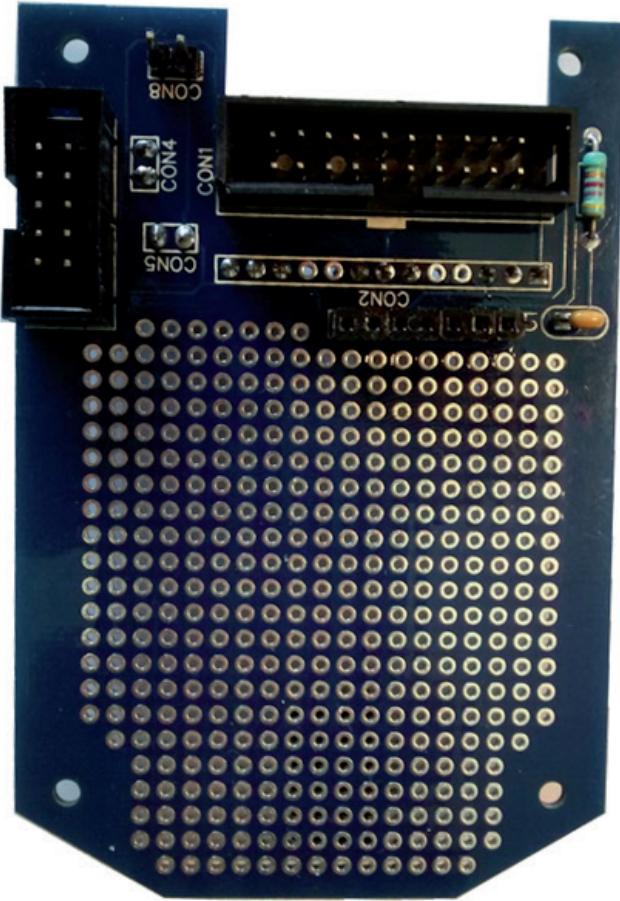
Man kann das Programmierung-Erweiterungsmodul auf zwei Weisen benutzen. Falls Sie das Anzeigemodul ebenfalls installiert haben können Sie das Programmierung-Erweiterungsmodul auf dieses Modul befestigen. Andernfalls können Sie das Programmierung-Erweiterungsmodul als eigenständiges Modul einsetzen.

Falls Sie das Anzeigemodul auf Ihrem Yeti montiert haben müssen Sie einige Steckverbinder auf dieser Leiterplatte (Steckverbinder 2, 4 und 5) auslöten und die kurze Steckerleisten durch längeren ersetzen. Auf dem Programmierung-Erweiterungsmodul müssen Sie die weiblichen Steckverbinder 2, 4 und 5 auf der Leiterplattenunterseite einlöten, so dass diese in das Anzeigemodul eingesteckt werden können. Beachten Sie bitte beim Löten an der Leiterplattenunterseite dass die Stifte genau senkrecht montiert werden, sonst werden sie nicht in die Steckerleisten des Anzeigemoduls passen. Die übrigen Steckverbinder werden einfach auf der Leiterplattenoberseite eingelötet.

Falls Sie das Anzeigemodul nicht auf Ihrem Yeti montiert haben benutzen Sie das Programmierung-Erweiterungsmodul autarke Baugruppe. Wenn Sie das Modul als selbständige Baugruppe verwenden sollten Sie keine Steckverbinder auf der Leiterplattenunterseite festlöten, weil das beim Installieren die Yeti-Oberseite nicht freihält Sie sollten nur Steckverbinder 1 und das mitgelieferte Flachbandkabel benutzen.

Steckverbinder 8 kann als Überbrückungsverbindung geschaltet werden falls Sie die schnelle kabelgebundene Datenverbindung beim eingesteckten Leiterplattenmodul benutzen wollen. Falls Sie die Hochladegeschwindigkeit beim installierten Leiterplattenmodul anpassen wollen löten Sie eine Steckerleiste auf diesen Lötaugen und benutzen eine Schaltbrücke um zwischen 2400 Baud und 57600 Baud Hochladegeschwindigkeit umzuschalten.





YETI Programmierung-Erweiterungsmodul YT-PRG Assembliert

Zum Abschluss

Wir hoffen, dass unsere Roboter ASURO und YETI Ihnen auf den Weg in die Roboterwelt geholfen haben! Wie unsere japanischen Freunde glauben auch wir, dass Roboter nach den Computern und Mobiltelefonen die nächste technologische Revolution bilden werden. Diese Revolution wird auch neue wirtschaftliche Impulse auslösen. Leider haben Japan, andere asiatische Länder und auch die USA, Europa dabei längst überholt. Im Gegensatz zu Europa beginnt der Technikunterricht in Asien bereits in der Grundschule und ist ein wichtiger Bestandteil der Ausbildung. Als Zielsetzung bei der Entwicklung der Roboter ASURO und YETI haben wir deshalb gewählt:

TO TRAIN A SCIENTIFIC MIND



APPENDIX

A. ÜBERSICHT DER YETI FUNKTIONEN

Grundfunktionen

initYeti();

Initialisierungsfunktion für die Basisfunktionen der Yeti-Modulen. Diese Initialisierungsfunktion initialisiert den Mikroprozessor nicht für die Anzeigefunktionen oder für den Betrieb der Ultraschallmodule.

wirelessSerialInit();

Initialisierungsfunktion für die drahtlose serielle Modulen (USB seriell nach Infrarot)

IRSerialprint(int TXData) / IRSerialprint(String TXData);

Überträgt entweder eine Integer-Zahl oder eine Zeichenkette (via Infrarotverbindung und serieller Port-Anschluss) zum PC oder (theoretisch) ein zusätzlicher Yeti.

IRSerialprintln(int TXData) / IRSerialprintln(String TXData);

baut die gleiche Kommunikationsverbindung auf wie oben, aber wechselt nach jedem Datensatz auf eine neuen Zeile, was für die Benutzung des Seriellen Port Viewers am PC übersichtlicher wirkt. Die Namen dieser und der vorherigen Funktion unterscheiden sich durch ein „In“ am Ende der Funktionsnamen, was als Abkürzung für „newline“ (Englisch für „neue Zeile“) verwendet wird.

IRSerialread();

Die Funktion liefert Daten aus dem seriellen Port zurück falls dort Daten vorliegen. Bitte vorher das Vorhandensein der Daten überprüfen mit der Funktion Serial.available(); Falls das Ergebnis 0 ist sind im Puffer keine seriellen Daten zum Ablesen vorhanden.

moveForwardX(int numberOfSteps);

das Kommando um den Yeti eine spezifizierte Anzahl Schritte vorwärts gehen zu lassen. Es ist die sicherste Methode den Yeti vorwärts laufen zu lassen..

moveForwardXNC();

Grundsätzlich die gleiche Funktionalität wie die obenstehende Funktion, aber mit geringfügigen Unterschieden. Diese Funktion führt keine Zentrierung durch, was die Spazierbewegung bei mehrfachen Wiederholungen eleganter wirken lässt. Sie können auch keine Anzahl der Schrittbewegungen angeben. Bei jedem Aufruf macht der Roboter nur einen Schritt. Die Funktion wurde speziell für das RF-Modul optimiert.

moveBackwardXNC();

Die Funktion verhält sich wie die vorherige, aber bewegt den Roboter in die andere Richtung: rückwärts anstatt vorwärts.

turnRight(int angle);

bewegt den Yeti nach rechts – in kleinen Schritten nach Rechts, falls der Winkel 0 beträgt. Bei einem Winkel größer als Null werden größeren Schritten gemacht.

turnLeft(int angle);

bewegt den Yeti nach links – in kleinen Schritten nach links, falls der Winkel 0 beträgt. Bei einem Winkel größer als Null werden größeren Schritten gemacht.

moveBody(int leanDirection);

bewegt den Körper wahlweise nach rechts oder nach links oder aufwärts. Die Funktion ist zum Beispiel praktisch, wenn Sie den Yeti rückwärts steuern wollen, oder wenn Sie ihn Kreisbewegungen durchführen lassen wollen.

Falls Sie die Funktion benutzen wollen, ist es sinnvoll im obersten Programmbereich folgende Definitionen einzufügen: 3 bedeutet eine Rechtsbeugung, 4 eine Linksbeugung und 5 eine Hoch aufgerichtete Position.

moveLegs(int walkDirection)

bewegt die Yeti-Beine in die gewünschte Position. Als Optionen gelten: links vorwärts, rechts vorwärts, und mittig. Diese Funktion ist gut kombinierbar mit der zuvor erwähnten Funktion, zum Beispiel um den Yeti kreisen zu lassen.

Falls Sie die Funktion benutzen wollen, ist es sinnvoll im obersten Programmbereich folgende Definitionen einzufügen: 6 bedeutet eine links vorwärts, 7 rechts vorwärts und 8 mittig.

beep(int frequency, int duration);

Diese Funktion lässt den Piepser mit der angegebenen Frequenz und für die spezifizierten Zeitdauer ertönen. Die Funktion wirkt nicht blockierend, so dass das Programm nach dem Aufruf ohne das Ende abzuwarten weitermacht. Die Frequenz wird in Hertz und die Zeitdauer in Millisekunden. 1 Sekunde = 1000 Millisekunde.

initDisplay();

initialisiert das Anzeigemodul des Yetis und startet den Treiber für die Anzeige.

displayDigit(int displayNumber);

zeigt eine Einzelzahl an von 0 bis 9999, wobei die Leerstellen (wie zum Beispiel die 3 Leerstellen in der Darstellung der Zahl „1“) mit Nullen aufgefüllt wird.

displayDigit(int firstDigit, int secondDigit, int thirdDigit, int forthDigit);

zeigt vier Einzelziffern, die jeweils von 0 bis 9 festgelegt werden können.

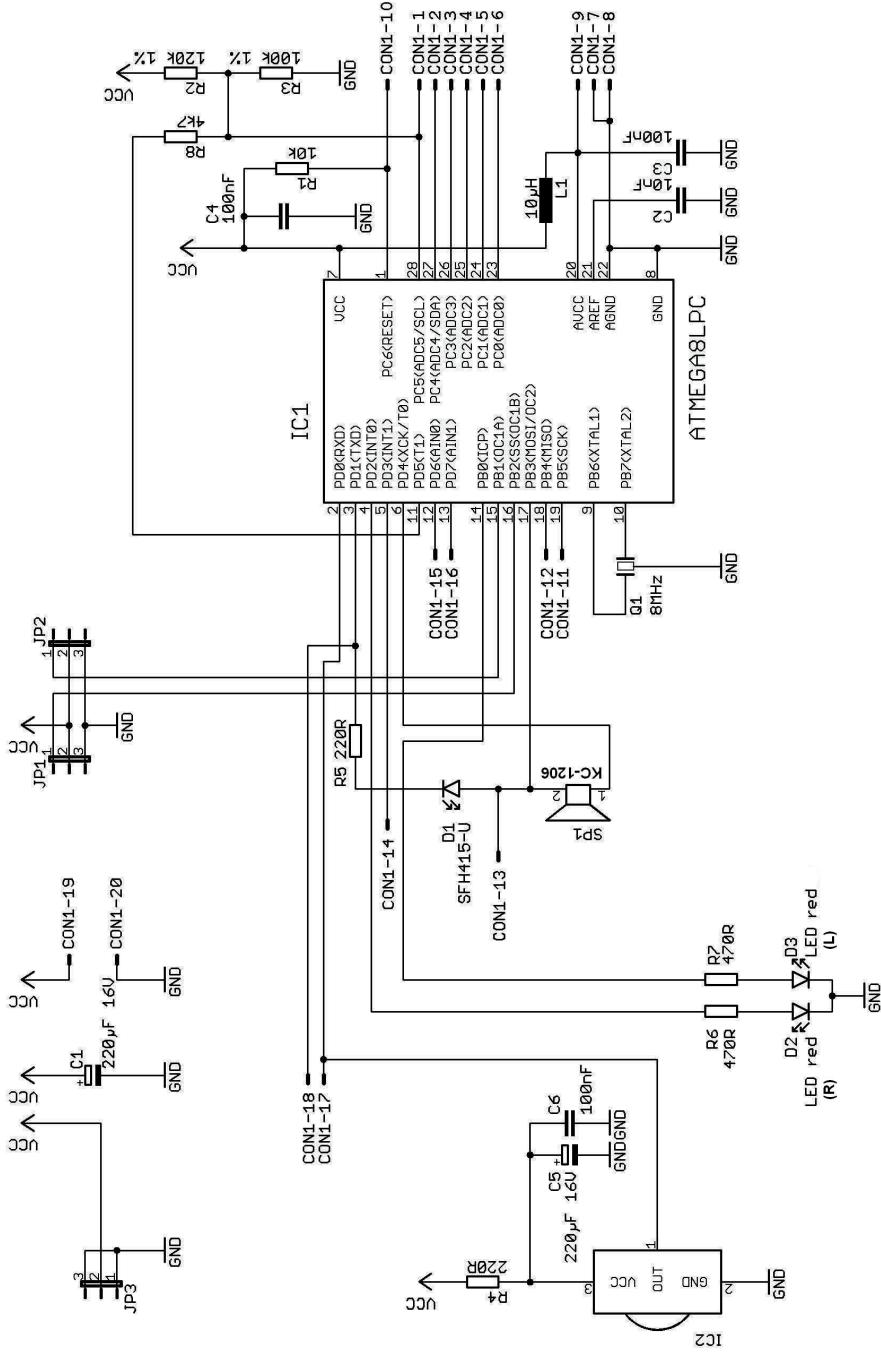
initPing();

initialisiert die ping-Module für den Yeti zur Einstellung der Zeituhren usw.

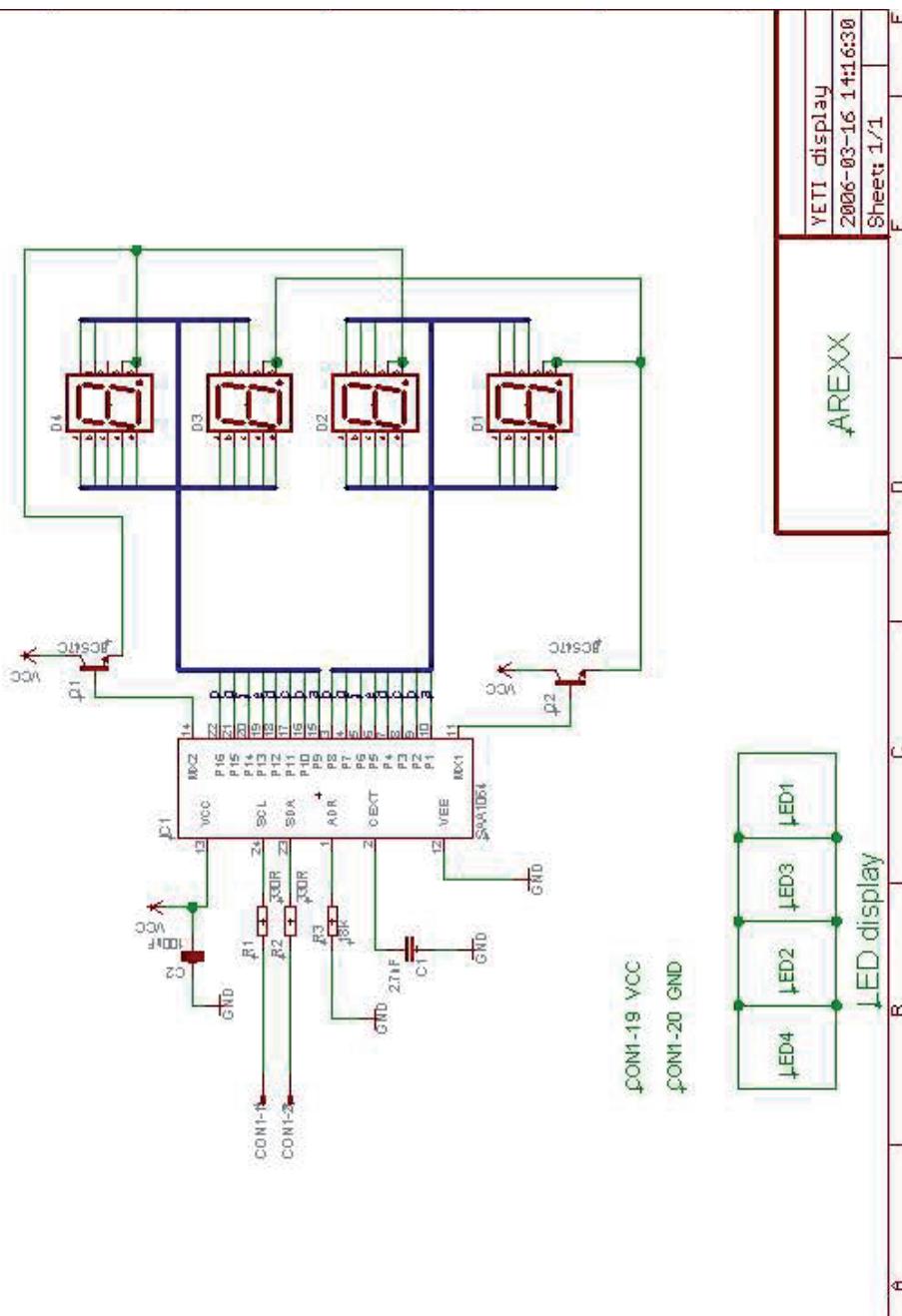
ping();

Abfrage der zur Zeit gültigen des Ping-Sensors. Dabei werden zuerst einige Zeituhren. Während der Aktivphasen der Ping-Funktion kann der Piepser eventuell nicht korrekt funktionieren. Dieser sendet 10 Impulse, wobei insbesondere junge Personen einen schwachen Klick wahrnehmen können. Die Funktion liefert einen Wert zurück, der die momentane Distanz zu einem Objekt in Zentimeter angibt. Falls kein Objekt gefunden wird liefert die Funktion eine Null zurück.

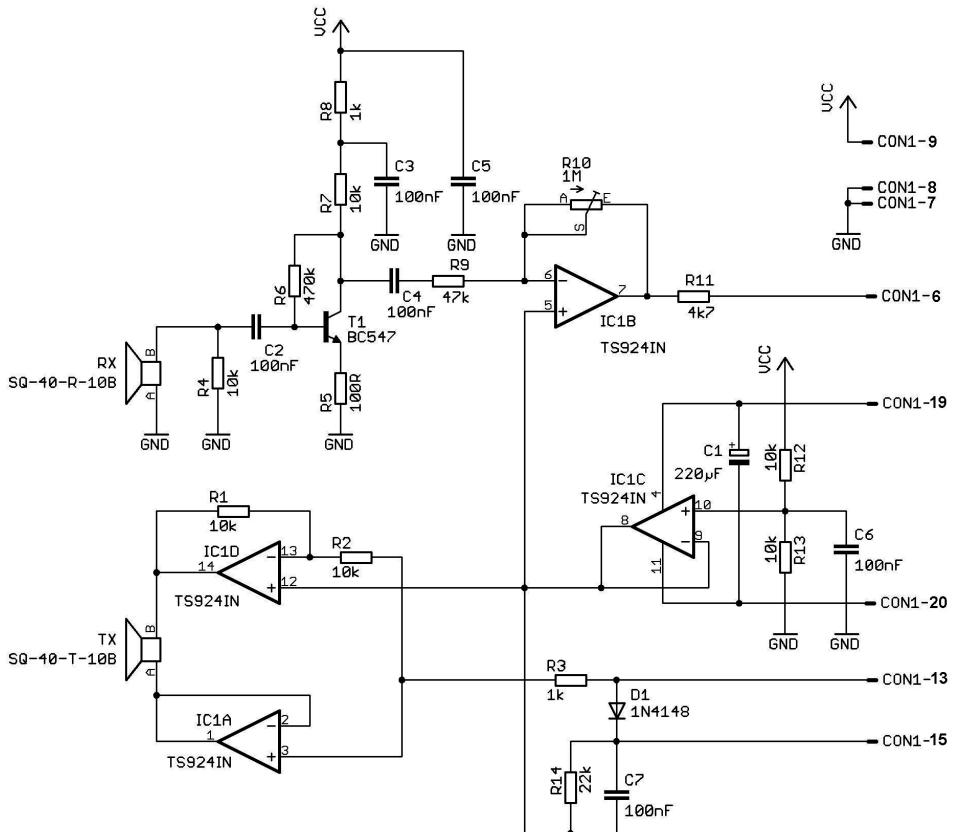
B. SCHALTBILD YETI



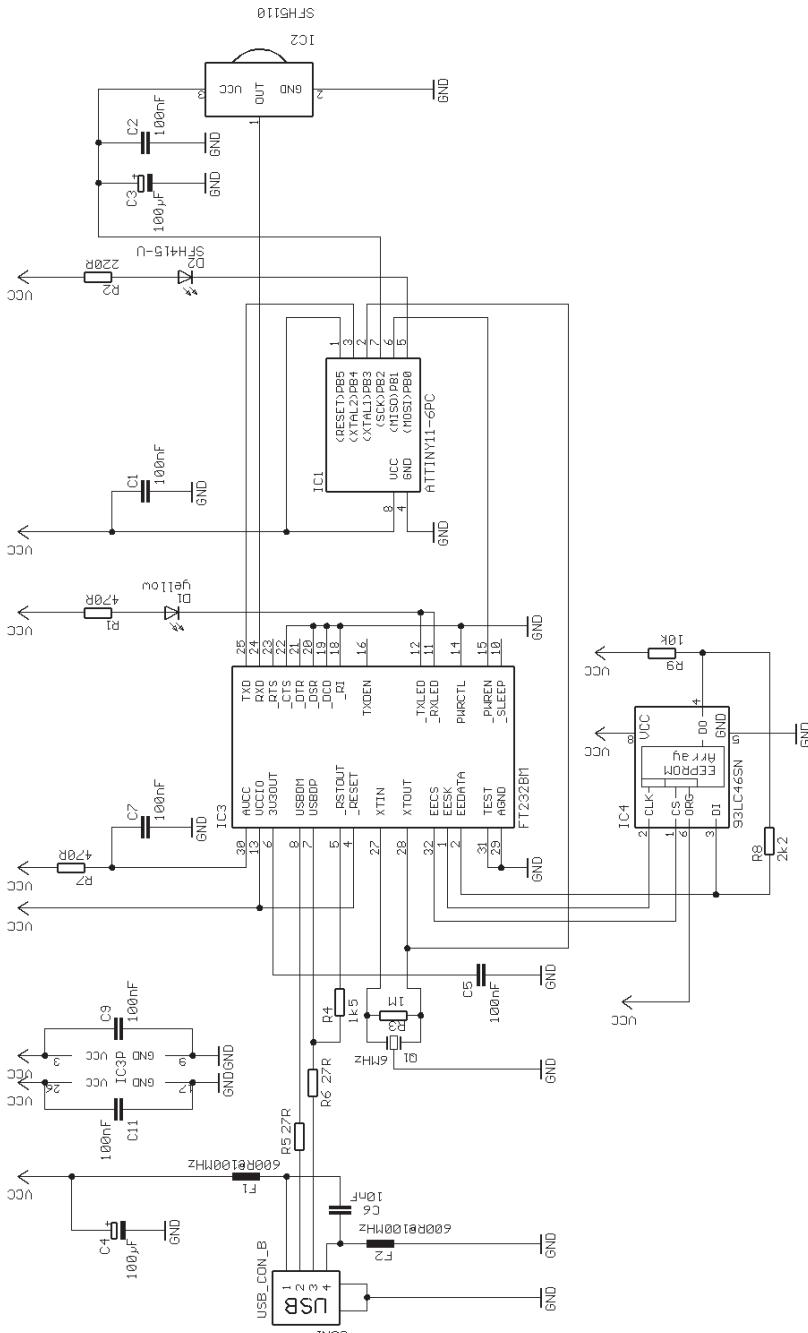
C. SCHALTBILD DISPLAY MODUL



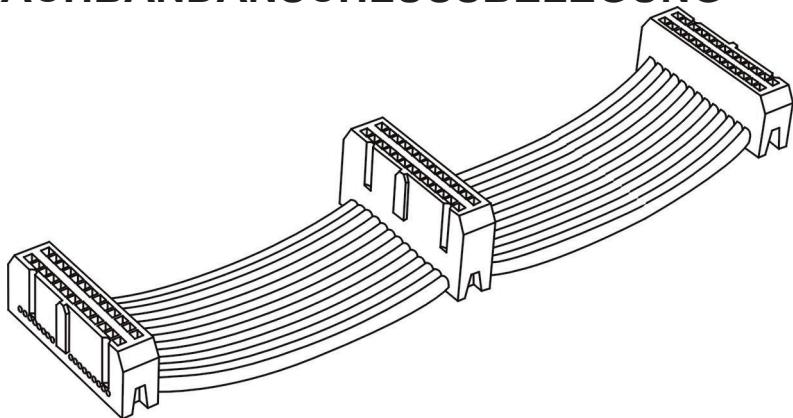
D. SCHALTBILD US MODUL



F. SCHALTBILD USB IR-TRANSCEIVER



G. FLACHBANDANSCHLUSSBELEGUNG



Pin 1	SCL	Serial Clock (für die I2C Datenkommunikation)
Pin 2	SDA	Serial Data (für die I2C Datenkommunikation)
Pin 3	PC3(ADC3)	Digitaler Eingang/Ausgang oder analoger Messeingang
Pin 4	PC2(ADC2)	Digitaler Eingang/Ausgang oder analoger Messeingang
Pin 5	PC1(ADC1)	Digitaler Eingang/Ausgang oder analoger Messeingang
Pin 6	PC0(ADC0)	Digitaler Eingang/Ausgang oder analoger Messeingang
Pin 7	GND	GND (Mehrere Anschlüsse zur Entstörung vorgesehen)
Pin 8	GND	GND (Mehrere Anschlüsse zur Entstörung vorgesehen)
Pin 9	AVCC	Betriebsspannung für den AD-Konverter
Pin 10	PC6(RESET)	Mikrocontroller Reset Anschluss
Pin 11	PB5(SCK)	Digitaler Eingang/Ausgang
Pin 12	PB4(MISO)	Digitaler Eingang/Ausgang oder I2C Funktion Pin
Pin 13	PB3(MOSI/OC2)	Digitaler Eingang/Ausgang oder I2C Funktionspin oder Timer2 Pin
Pin 14	PD3(INT1)	Digitaler Eingang/Ausgang oder externer Interrupt
Pin 15	PD6(AIN0)	Digitaler Eingang/Ausgang oder analoger Testeingang
Pin 16	D7(AIN1)	Digitaler Eingang/Ausgang oder analoger Testeingang
Pin 17	PD0(RXD)	Digitaler Eingang/Ausgang oder RS232 Eingang
Pin 18	PD1(TXD)	Digitaler Eingang/Ausgang oder RS232 Ausgang
Pin 19	VCC	VCC
Pin 20	GND	GND (Mehrere Anschlüsse zur Entstörung vorgesehen)

H. Fehlersuche

H1. ALLGEMEIN

Alle Bauteile auf richtige Einbaulage (Polung) und korrekten Wert prüfen. Lötstellen auf Kurzschlüsse bzw. kalte Lötstellen nachsehen. Hat sich irgendwo ein Lötauge gelöst? Sind diese Kontrollen durchgeführt ohne einen Fehler zu finden, muss mit Hilfe des Schaltplanes und einem geeigneten Messgerät (Multimeter bzw. Oszilloskop) das defekte Bauteil gesucht werden. ICs, Transistoren, Dioden und LEDs sind oft die wahrscheinlichsten Kandidaten für einen Defekt.

H.2. Störung der ARDUINO IDE Kommunikation

- Überprüfen Sie bitte ob der korrekte COM-Port gewählt worden ist
- Überprüfen Sie bitte ob Sie das korrekten Arduino-System beziehungsweise den richtigen Arduino-Prozessor gewählt haben.

H4. IR-Schnittstelle

H.4.1. YETI sendet keine Zeichen

SFH 415-U, IR-LED richtig eingebaut?

Polung der IR-LED prüfen.

Widerstand R7 (RS-232 IR-transceiver) und Widerstand R5 (YETI) richtig?

220 Ω (ro,ro,br,gld)

H.4.2. YETI empfängt keine Zeichen

Zwischen IR-Transceiver und YETI muss Sichtverbindung bestehen (Abstand ca. 50cm) und der IR-Transceiver muss voll funktionsfähig sein.

SFH5110 IR-Empfänger-IC richtig eingebaut in YETI und RS-232 IR-Transceiver?

Transistor Q1, Diode D4, Widerstand R3 und C4 im RS-232 Transceiver und

Widerstand R4 und C6 auf YETIs Hauptplatine überprüfen !

470 Ω (br,sw,br,gld)

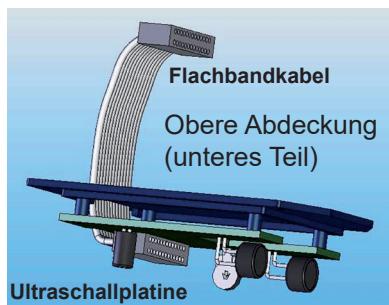
100nF (Aufdruck104)

Wer den Fehler bisher noch nicht gefunden hat, der möge sich überlegen, ob er das SFH5110 IR-Empfänger-IC eingelötet oder "eingeschweißt" hat. SFH5110 IR-Empfänger-IC ist ein wenig hitzeempfindlich und ist evtl. beim Einbauen kaputt gegangen, dann neues IC (SFH 5110-36) besorgen und einbauen.

H.4.3. Es geht immer noch nicht so richtig

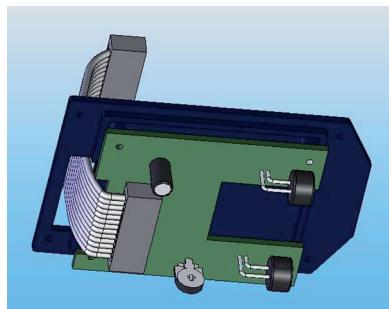
Kommt es bei der Übertragung von Daten vom PC zu YETI immer wieder zu Schwierigkeiten, so muss ein wenig am Trimmer TR1 des Transceivers gedreht werden.

I. MONTAGE ERWEITERUNGSKITS



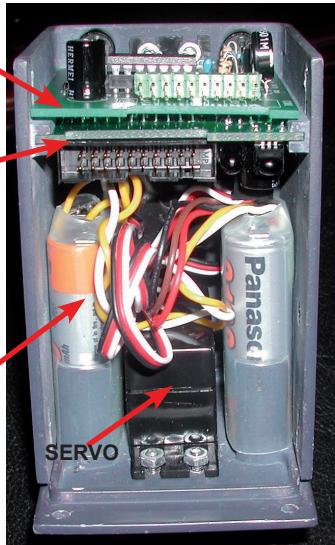
Ultrasonic Board

Main Board



Assembly Main Board

Main Board



Assembly
Display Board



Assembly Experimental Board



K. ADC MESSWERT AKKUSPANNUNG

Umrechnungstabelle ADC-Messwert in Akkuspannung

Autor: YETI-Tester uwegw

Diese File können Sie downloaden auf WWW.AREXX.COM

Im blau markierten Feld kann die an Pin21 (AREF) des Mega8 gemessene interne Referenzspannung eingetragen werden. Sie sollte etwa 2,56V betragen, kann aber von Chip zu Chip etwas abweichen, z.B. auf 2,7V. Aus dieser Referenzspannung errechnet die Tabelle den zu jedem Spannungswert gehörigen ADC-Wert.

AREF: 2,71 <An Aref gemessene Referenzspannung

Überspannung	Akkuspannung in Volt	ADC-Sp.	ADC-Wert	Spannungsbereich
	6	2,73	1031	In diesem Bereich wird die zulässige Betriebsspannung einiger Komponenten überschritten!
	5,9	2,68	1013	Es drohen Schäden an Bauteilen!
	5,8	2,64	996	
	5,7	2,59	979	
	5,6	2,55	962	
	5,5	2,50	945	
Akku voll	5,4	2,45	927	In diesem Bereich sollten alle Komponenten einwandfrei funktionieren.
	5,3	2,41	910	
	5,2	2,36	893	
	5,1	2,32	876	
	5	2,27	859	
	4,9	2,23	842	
	4,8	2,18	824	
	4,7	2,14	807	
Akku halbvoll	4,6	2,09	790	In diesem Bereich wird die minimale Spannung für den Infrarotempfänger unterschritten.
	4,5	2,05	773	Er könnte hier bereits Funktionsstörungen zeigen, ebenso andere Komponenten,
	4,4	2,00	756	die mehr als 4,5V brauchen. Der Mega8 hat noch genügend Spannung.
	4,3	1,95	739	
Akku bald leer	4,2	1,91	721	In diesem Bereich funktioniert der Mega8 noch, sonstige Komponenten könnten aber
	4,1	1,86	704	sich ausfallen. Der Akku ist bald leer und sollte nachgeladen werden.
	4	1,82	687	
	3,9	1,77	670	
	3,8	1,73	653	
	3,7	1,68	635	In diesem Bereich wird die empfohlene Endladeschlussspannung der Akkus von 0,9V pro
	3,6	1,64	618	Zelle erreicht. Wenn die Akkus noch weiter entladen werden, können sie beschädigt werden!
	3,5	1,59	601	
	3,4	1,55	584	
	3,3	1,50	567	
	3,2	1,45	550	
	3,1	1,41	532	
	3	1,36	515	Hier wird die Grenze der Betriebsspannung für den Mega8 erreicht.