NXP-Robo-System

Lehrerworkshop Januar 2009

Shared Technology Infrastructure
Lab Service
NXP Germany GmbH

Herzlich Willkommen

Shared Technology Infrastructure STI

- Entwicklungszentrum Hausbruch
- ~400 Mitarbeiter
- Hauptwerk in Lokstedt
- ~2000 Mitarbeiter
- Entwicklung von Chips für:
  Autoradios
  TV Sets, Set-Top-Boxen
  Geldkarte
  Reisepass
  Warenlabel
  Wegfahrsperre
  Autotür-Systeme

Vorstellung

Shared Technology Infrastructure STI

Björn Jereczek
LA060 Tel. 3692

Wolfgang Günther
LA054 Tel. 2025

Detlef Dwenger
LA054 Tel. 3763

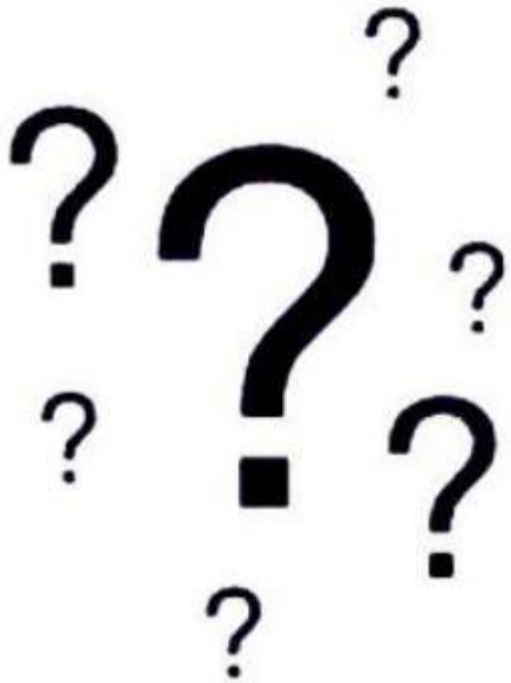
Tobias Schnardthorst
LA070 Tel. 2456

Thorsten Wolthausen
LA058 Tel. 1909

Ablauf

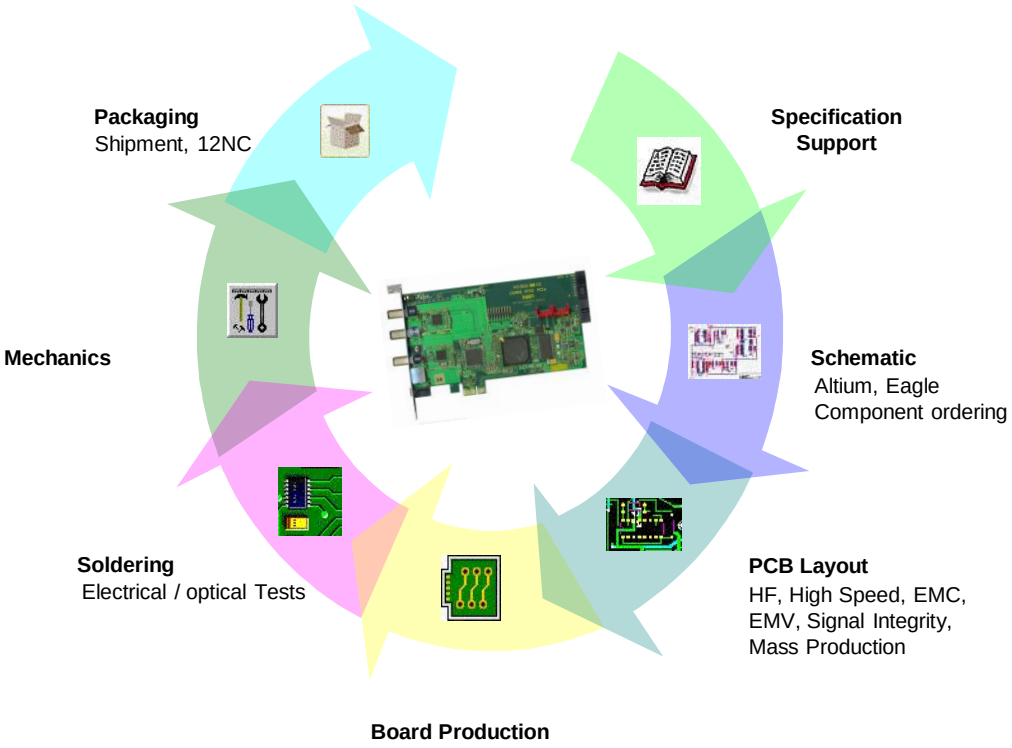
9:00	Begrüßung
	Theorie I
	Einführung
	Hardware
10:30	Pause
10:45	Theorie II
	Hardware Programmierung
11:45	Elektrisch Mechanischer Aufbau
	Einweisung Hardware aufbauen
12:15	Mittag
13:00	Hardware aufbauen
15:00	Pause
15:15	Inbetriebnahme und Fehlersuche
	Testprogramme
16:15	Software
	Roboter mit MOPS Funktion
17:15	Fragen
	Schlussbetrachtung und Feedback
18:00	Ende

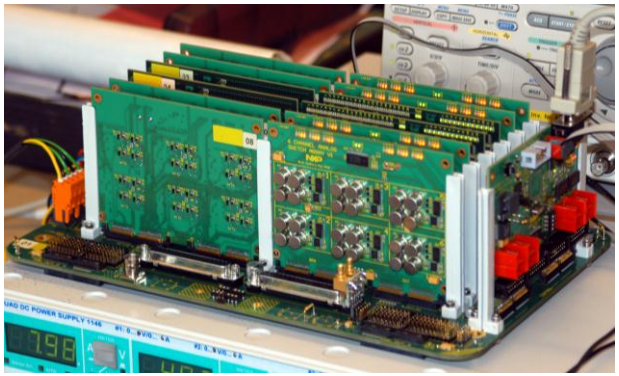
Fragen erwünscht

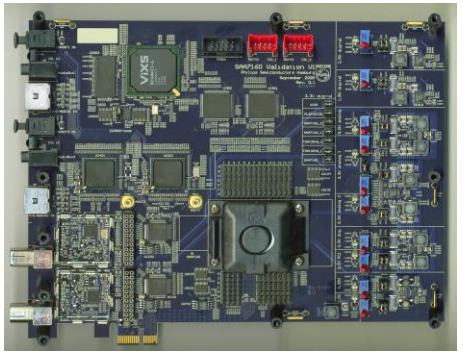


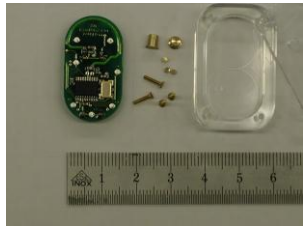
STI Lab Service

Shared Technology Infrastructure STI







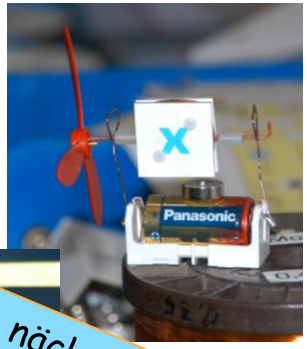


Girlsday

Shared Technology Infrastructure STI

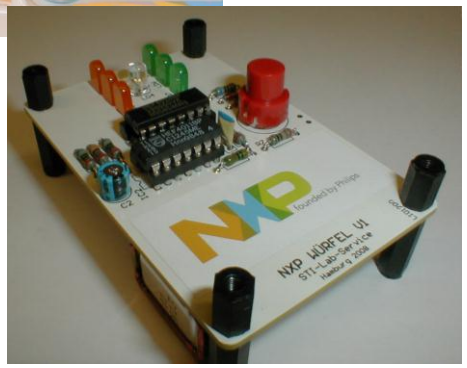
Am Anfang war es noch schwer mit dem Löten, …

Das Wort löten hatte ich noch nie gehört – jetzt weiß ich was das ist.



Wir möchten nächstes Jahr auf jeden Fall wieder kommen.

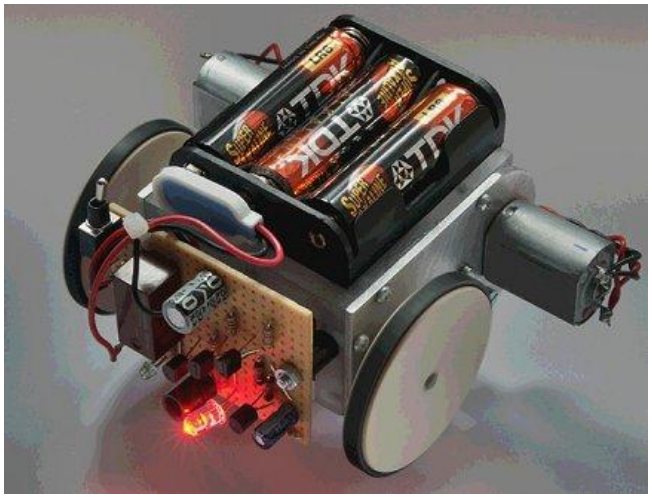
Girls'Day
Mädchen-Zukunftstag

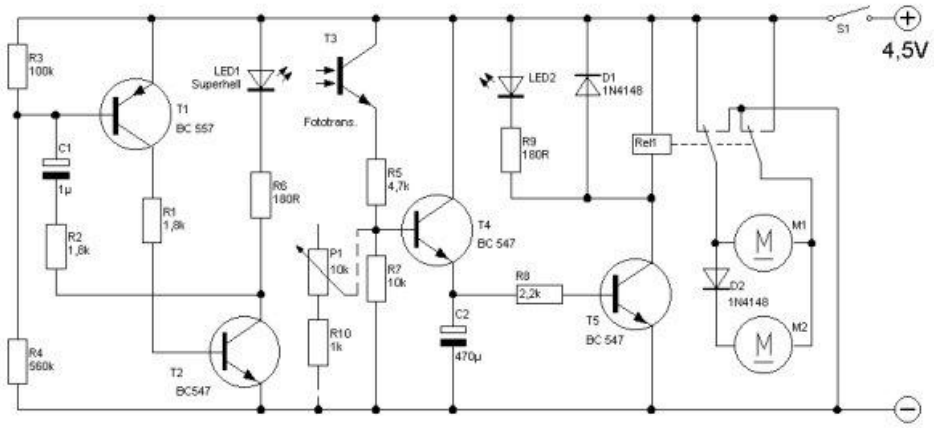


Interessant waren die Labore –
von der Fliege unter dem Elektronenrastermikroskop,
der Vergleich eines Menschenhaares mit einem Bonddraht
unter dem Mikroskop bis zum Chemielabor.

Mops “Vater”

- Markus Bindhammer
- Mops-Projekt: http://www.elexs.de/robo1.htm

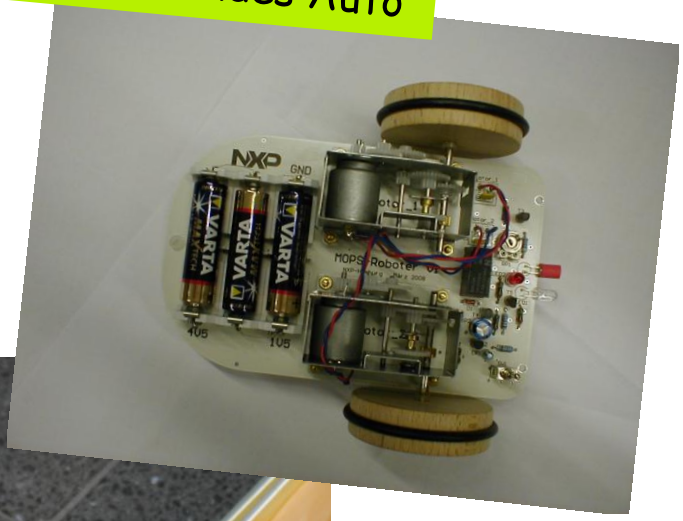


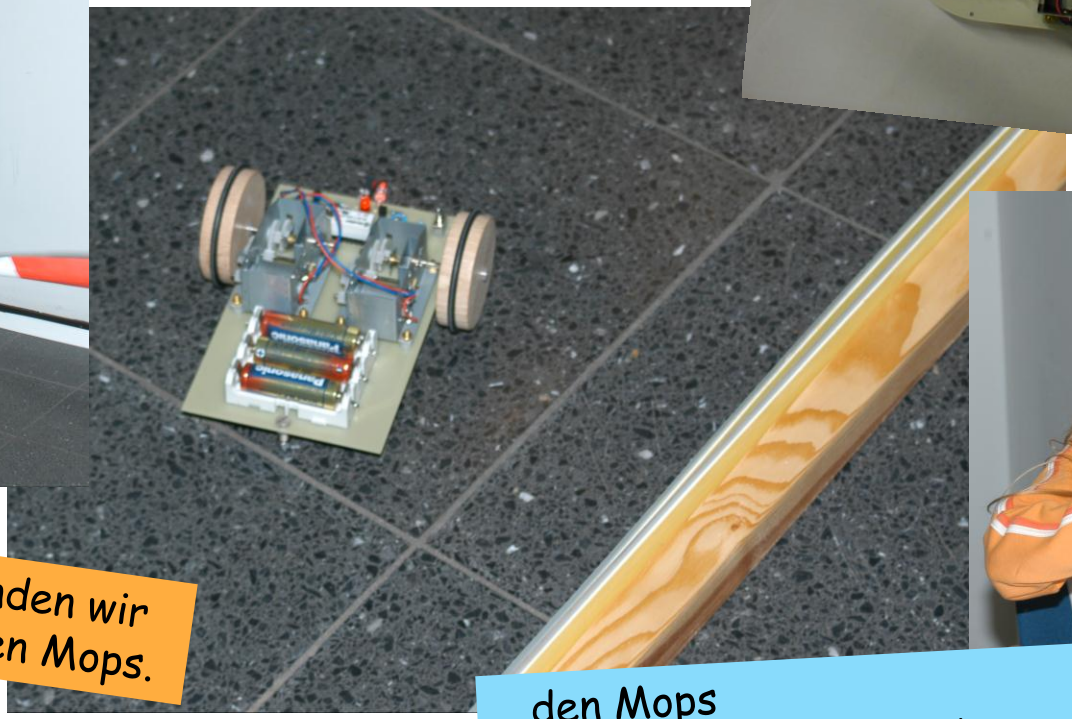


MOPS

Shared Technology Infrastructure STI

Ein fast von allein denkendes Auto



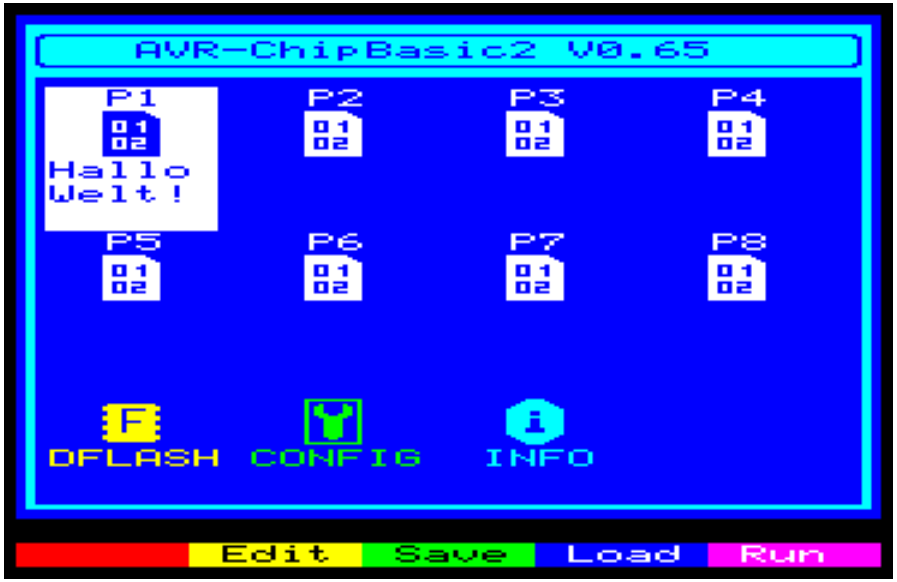


Besonders toll fanden wir den Würfel und den Mops.

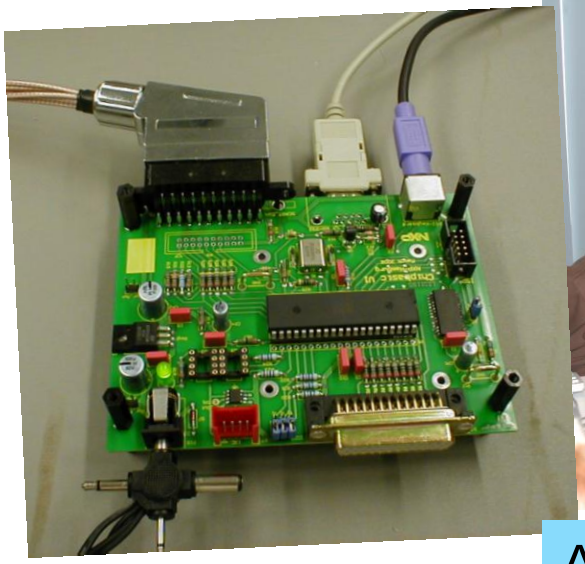
.. den Mops
(ein Roboter der Wänden ausweicht)

AVR ChipBasic2 “Vater”

- Jörg Wolfram
- AVR-Projekt: http://www.jcwolfram.de/projekte/avr/chipbasic2/main.php



AVR



AVR Minicomputer Workshop

... das Programmieren hat auch Spaß gemacht ...

... ein X-trem gutes Thema ...

AVR Übersicht


Computer


Serielle Schnittstelle


ISP Programmer

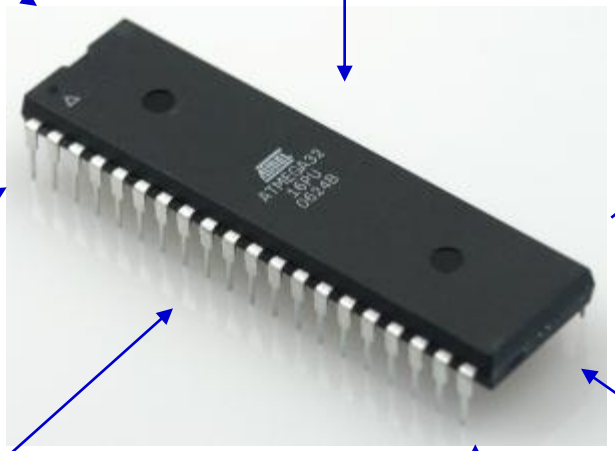

Daten Speicher


I2C Schnittstelle


Temperatur Sensor

Fernseher


Scart



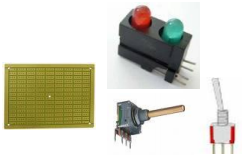

Dataflash Speicher


ISP Schnittstelle


Stecker Netzteil


Drucker


Parallele Schnittstelle

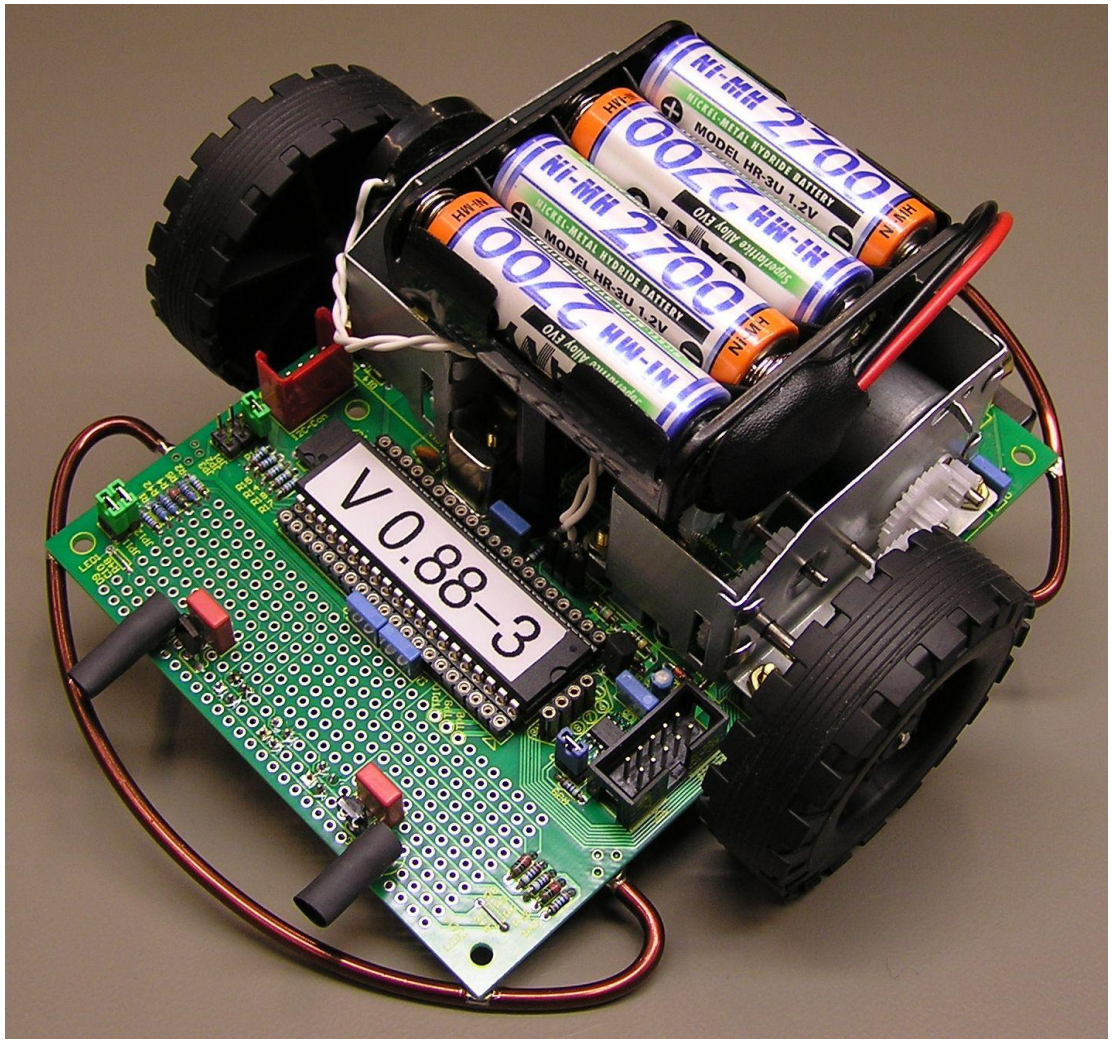

Erweiterungen


Konfiguration


PS2

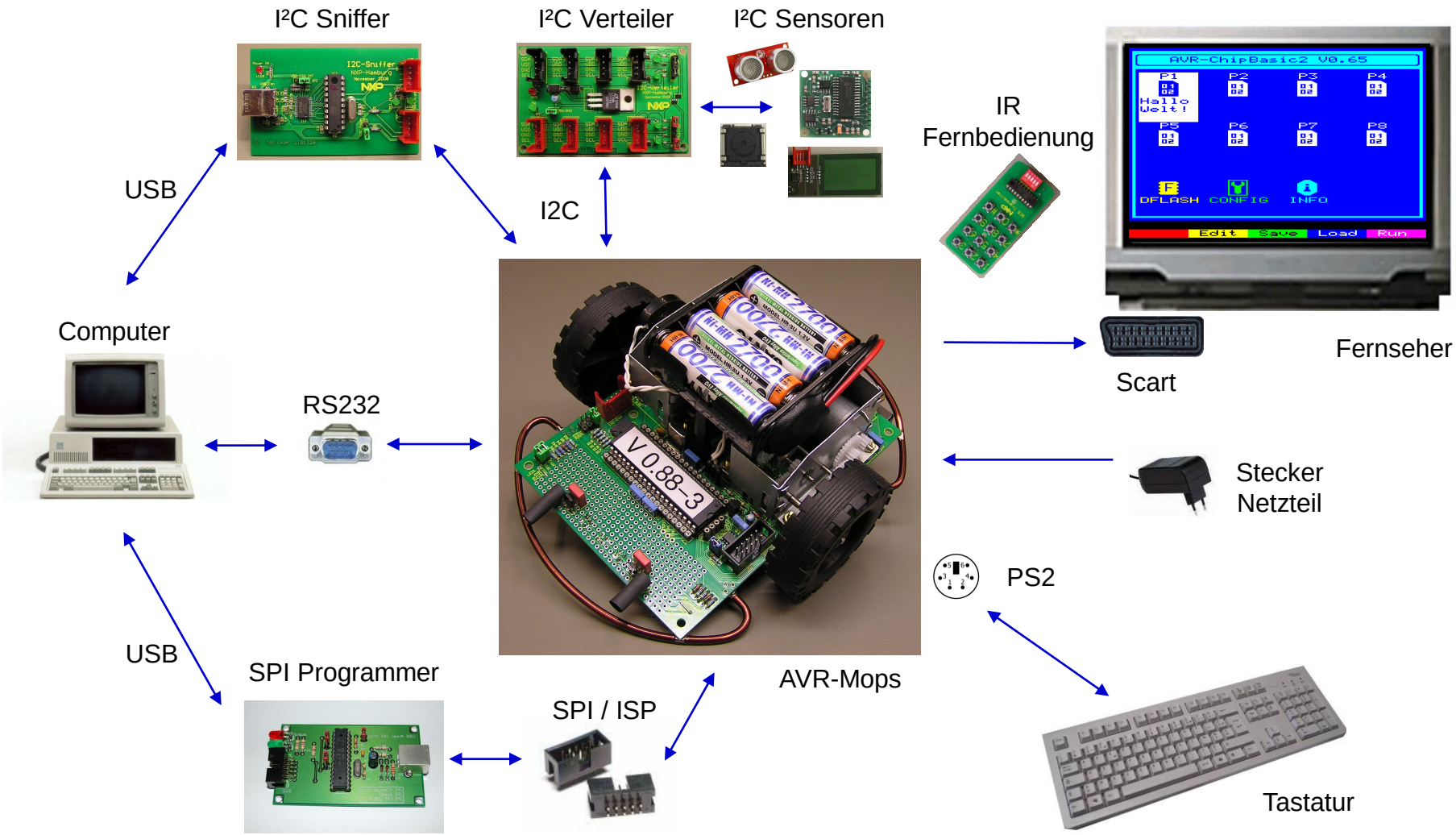
Tastatur

AVR-Mops



NXP-Robo-System

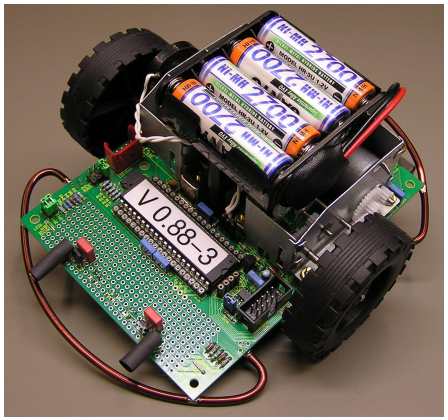
Shared Technology Infrastructure STI



Inhalt

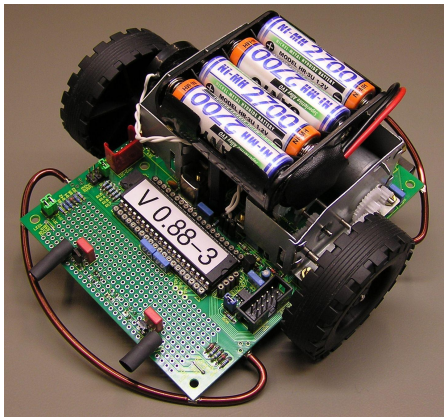
- AVR-Mops-System-Überblick
- Hardware
  - Sensoren
  - Motoren
  - Speicher
  - Interfaces
- Hardware Programmierung
- Elektrischer & mechanischer Aufbau

- Inbetriebnahme
- Programme
- Anregungen / Erweiterungen
- Unterrichtsideen
- Anhang



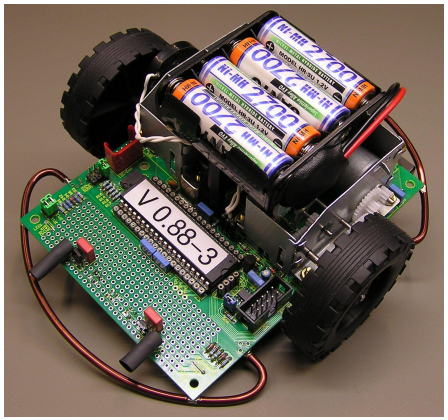
AVR-Mops System Überblick

- Hardware
- Blockschaltbild
- Platinenlayout
- Software



Hardware I

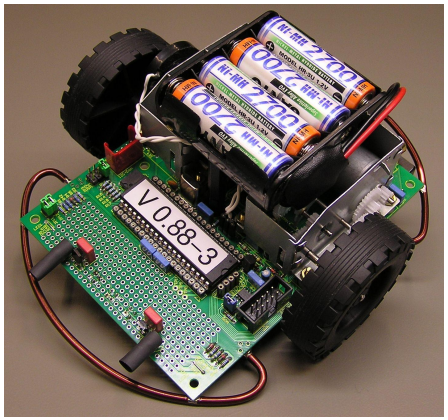
- CPU ATMega 644-20PO
- Sensoren
  - 3 Liniensensoren CNY70
  - 2 Abstandssensoren IS471F / LD274
  - Infrarotsensor TSOP3438 für RC5 Empfänger
- Motoren
  - 2 Motoren mit Getriebeblöcke
  - Motoransteuerung mit PWM PCA9533 und Leistungstreiber TDA7073



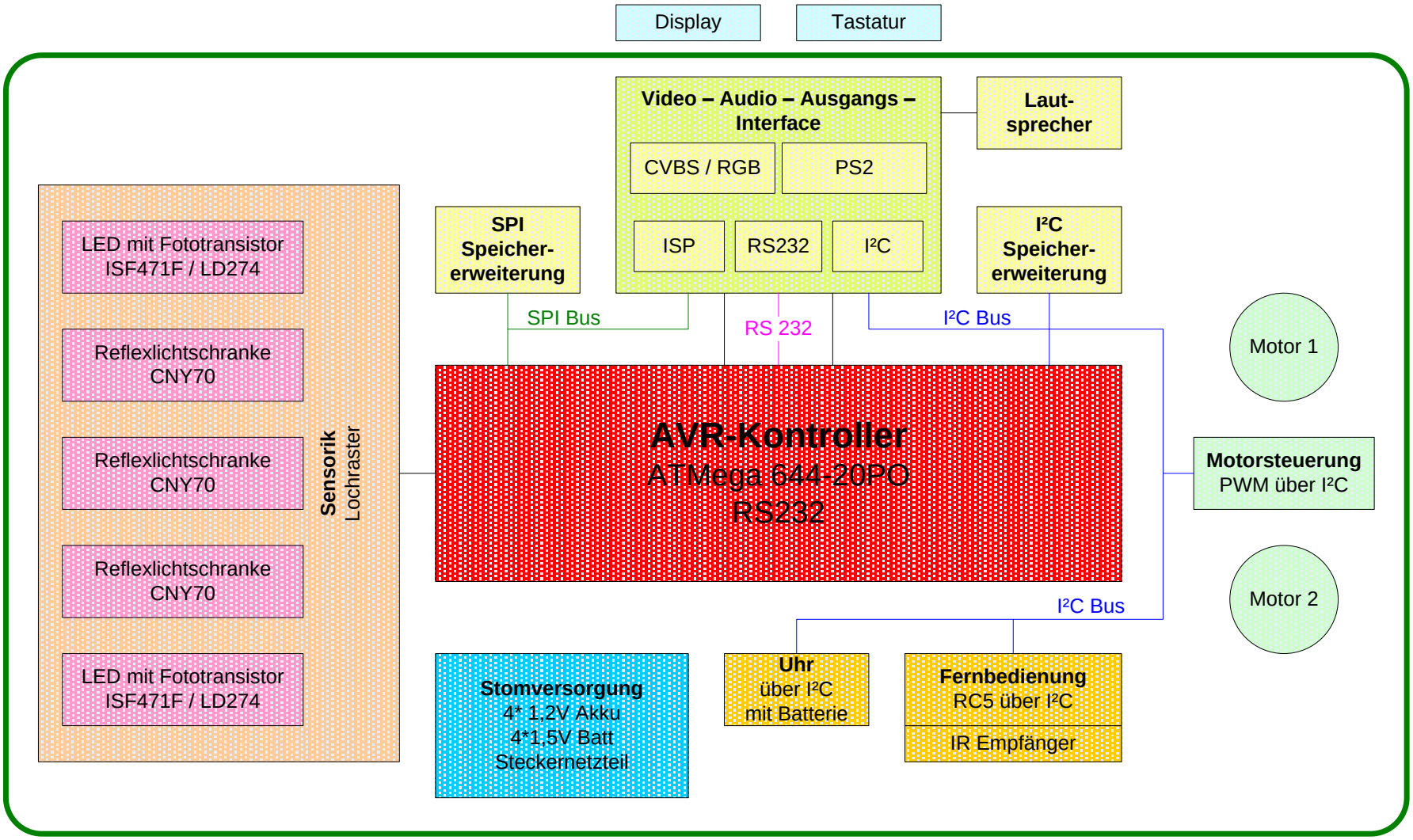
Hardware II

Shared Technology Infrastructure STI

- Speicher
  - Internen Programmspeicher in der CPU für 8 Programme
  - Externen Programmspeicher über SPI Bus AT45DB081
  - I²C Datenspeicher ST24C64
  - I²C RTC Uhr mit Datenspeicher PCF2128
- Interfaces
  - 5V Stromversorgung mit Akkus oder Netzteil
  - RS232 Anschluss
  - I²C Bus Anschluss
  - SPI BUS Anschluss
  - PS2 Tastatur Anschluss
  - Video Anschluss für Fernseher
  - Audio Anschluss für Lautsprecher

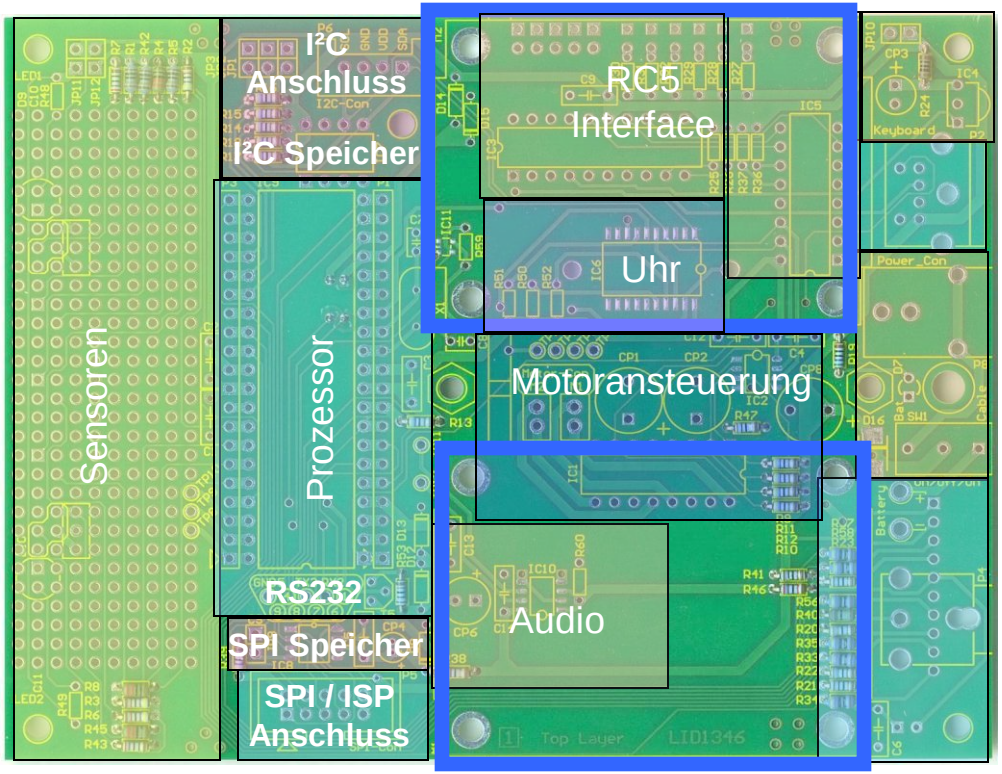


Blockschaltbild



Platinenlayout

Shared Technology Infrastructure STI



RC5 Empfänger

PS2 Tastatur Anschluss

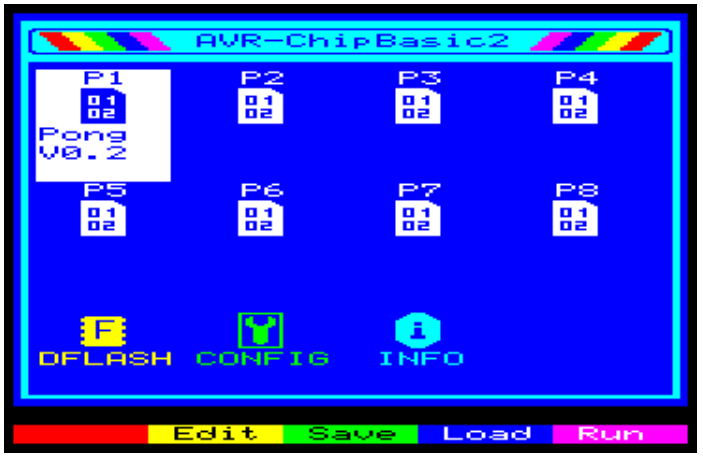
Power Anschluss

Video / Audio Anschluss

Software

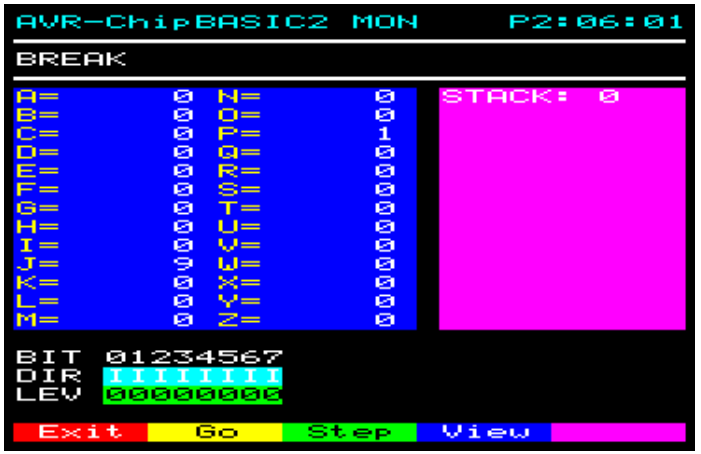
- Siehe Doku Jörg Wolfram
- Keine kommerzielle Software
- Aktuelle Version 0.89
- Bugs
- Autor kontaktieren

Grafische Oberfläche



Editor

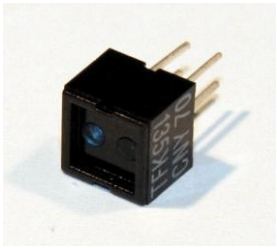
Debugger



Fragen

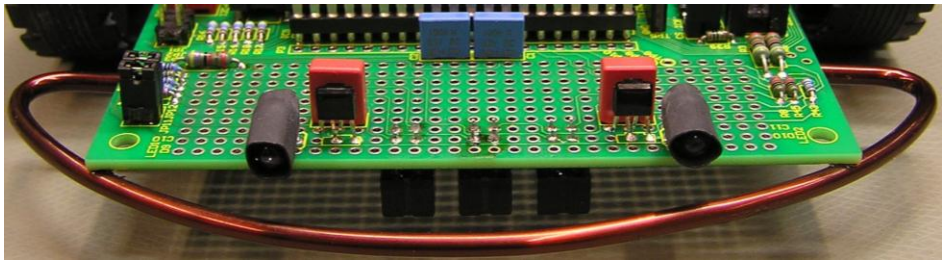
Sensoren

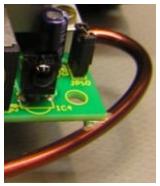
- Sensorbereiche des AVR-Mops
- Liniensensor CNY70
- Abstandssensor IS471F
- Infrarotsensor TSOP3438
- Anschlussmöglichkeiten am AVR
- Inter-Integrated-Circuit (IIC, I2C, I²C)
- WWW Links Sensoren



Sensorbereich des AVR-Mops

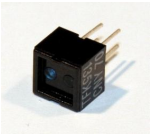
- Sensorbereiche vorne und hinten
- Abstands-, Linien- und Infrarot Sensor
- Spare Fläche für weitere Sensoren

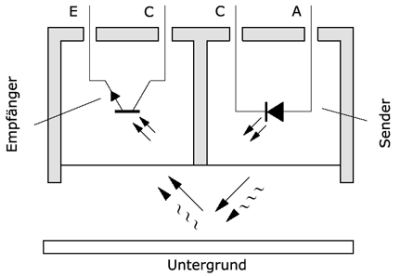


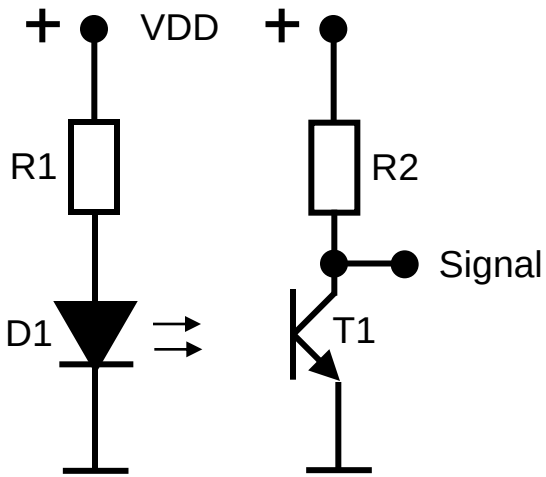


Liniensensor CNY70

- Reflex -Lichtschranke mit LED und Fototransistor
- Strom fließt durch R1 / D1 und D1 erzeugt Licht
- Licht trifft auf die Basis von T1
- Kollektor – Emitter Strecke von T1 schaltet durch
- Signal = 0V
- Hindernis zwischen D1 und T1
- Kein Licht auf die Basis von T1
- Kollektor – Emitter Strecke schaltet nicht durch
- Signal = VDD
- Empfindlich auf Fremdlicht !!!

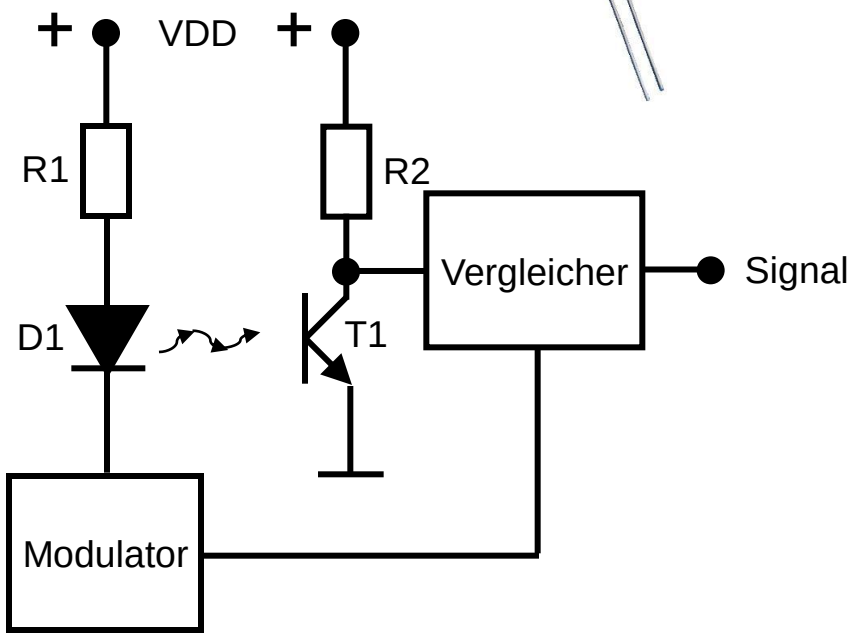






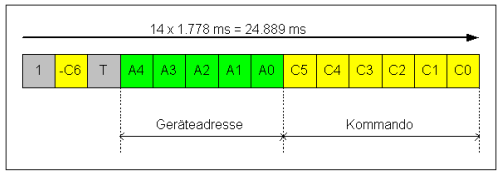
Abstandssensor IS471F / LD274

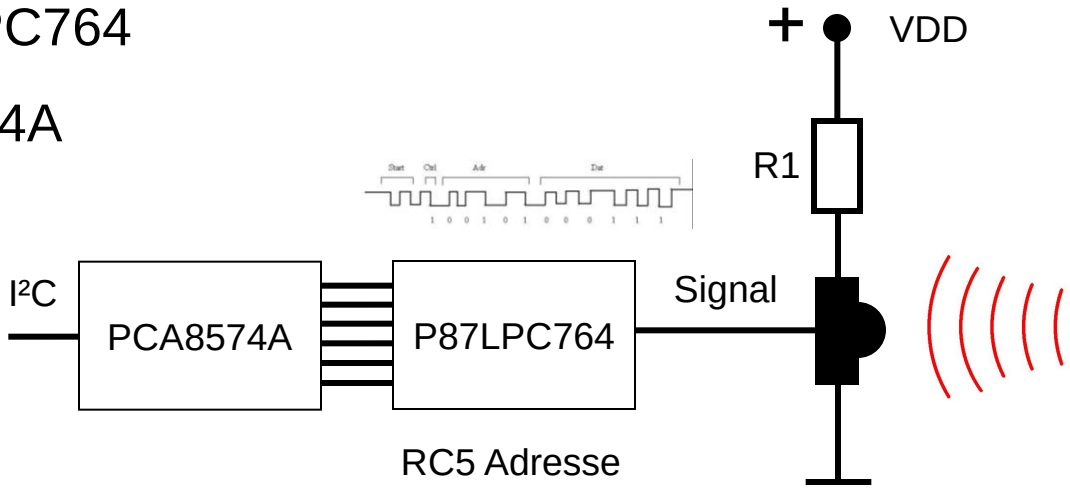
- Lichtschrankenprinzip wie in vorheriger Folie nur …
- D1 sendet moduliertes Licht aus
- Kollektorsignal wird mit dem Modulatorsignal verglichen
- Beide gleich => Signal = 0V
- Ungleich => Signal = VDD
- Unempfindlich für Fremdlicht !!!
- LD274 strahlt auch nach hinten !!!



Infrarotsensor TSOP3438

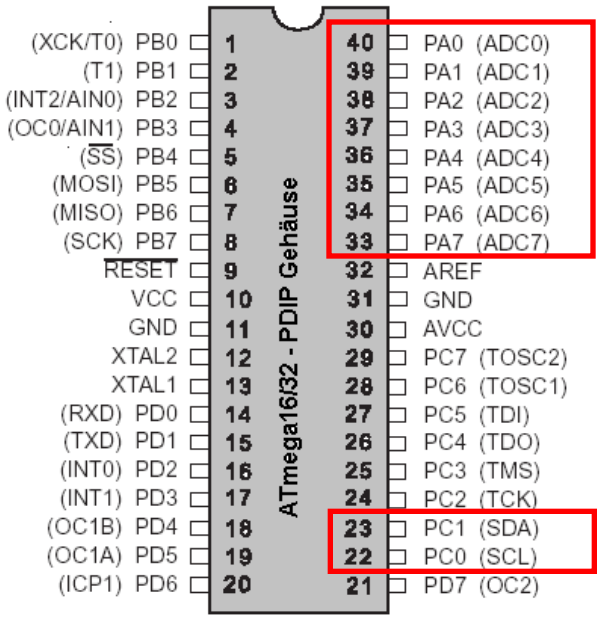
- Umwandlung vom modulierten Signal in ein digitales
- Trägerfrequenz 38KHz
- RC5 Format
- Geräteadresse und Kommando wird übertragen
- Fernbedienung für Audio / Video Equipment
- Dekodierung mit µC P87LPC764
- Auslesen über I²C PCA8574A





Anschlussmöglichkeiten am AVR

- 8 Port Bits (PA0..7) für
  - Input / Output / ADC
  - PA0 frei
  - PA1 frei
  - PA2 frei
  - PA3 = CNY70 Rechts
  - PA4 = CNY70 Links
  - PA5 = CNY70 Mitte
  - PA6 = IS471F Links
  - PA7 = IS471F Rechts
- I²C Bus

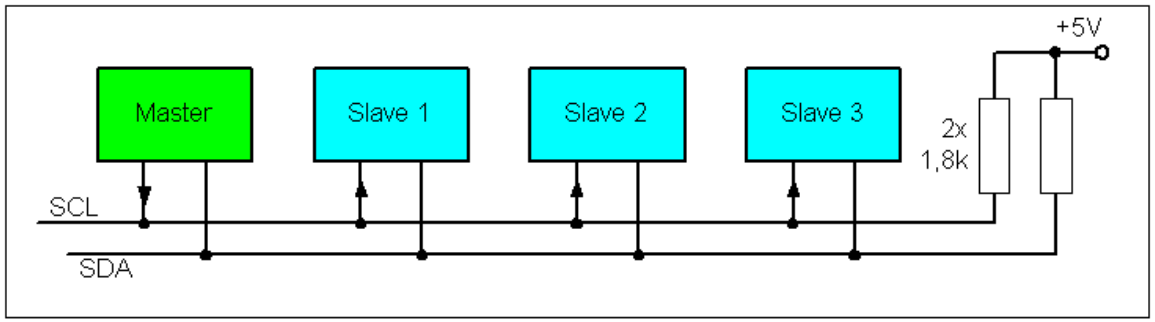


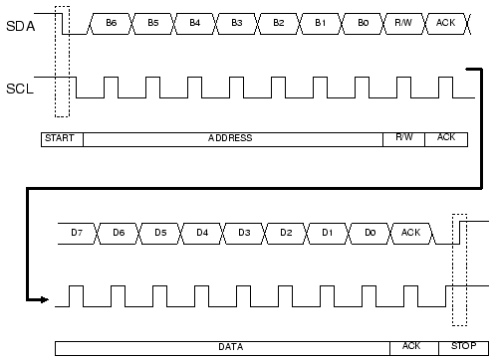
Inter-Integrated-Circuit - I²C Bus

- I²C = IIC = Inter-Integrated-Circuit
- In den 80er Jahren von der Firma Philips entwickelt
- Kommunikation zwischen einzelnen IC's
- Unterhaltungselektronik z.B. Fernsehgeräte, DVD-Player, etc.
- Synchrone serielle 2-Draht-Verbindung
- 5V / 3,3V Bus !!!
- Master (z.B. ein µController) und einer beliebigen Anzahl von Slaves
- Leitungen: SCL = Taktübertragung, SDA = Datenübertragung
- Daten vom Master zum Slave (schreiben)
- Daten vom Slave zum Master (lesen)

Inter-Integrated-Circuit - I²C Bus

- Jeder Slave hat eine 8Bit Adresse
- Bit0 der Adresse: 0=schreiben, 1=lesen (z.B. 0xA0=schreiben)
- Bus Ruhezustand, SCL und SDA auf High,1 (5Volt)
- Pullup Widerstände für SCL und SDA





WWW Links Sensoren

- Allgemein:
- http://www.roboternetz.de/wissen/index.php/Sensorarten
- http://www.kreatives-chaos.com/artikel/liniensensor-mit-cny70
- http://www.kreatives-chaos.com/artikel/abstandssensor-mit-is471
- RC5:
- http://www.roboternetz.de/wissen/index.php/RC5-Code
- http://www.sprut.de/electronic/ir/rc5.htm
- I²C:
- http://www.sprut.de/electronic/pic/grund/i2c.htm
- http://www.roboternetz.de/wissen/index.php/I2C
- http://www.elektronik-magazin.de/page/der-i2c-bus-was-ist-das-21

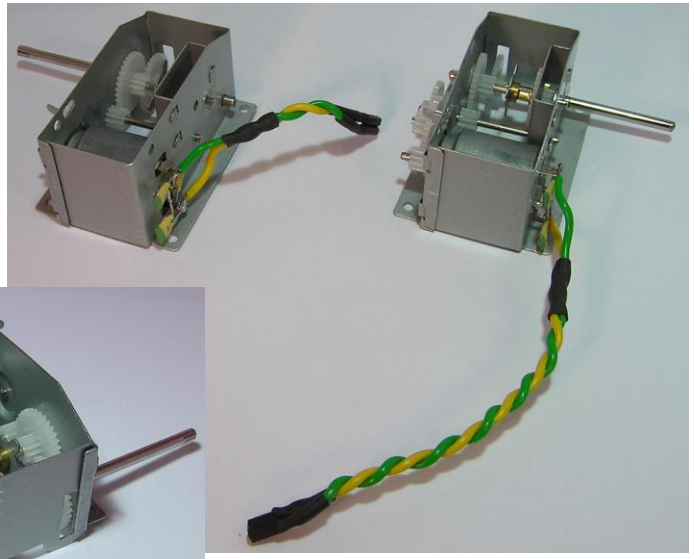
Fragen

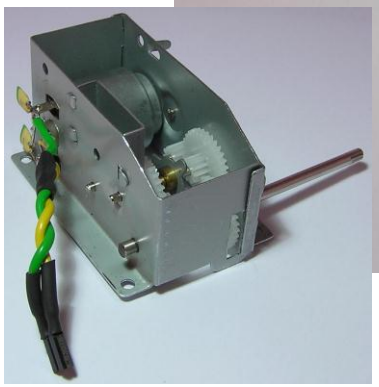
Motoren

- Motor- bzw. Getriebeblock
- Pulsweiten Modulation (PWM)
- Motoransteuerung
- Motorentstörung
- WWW Links Motoren

Motor- bzw. Getriebeblock

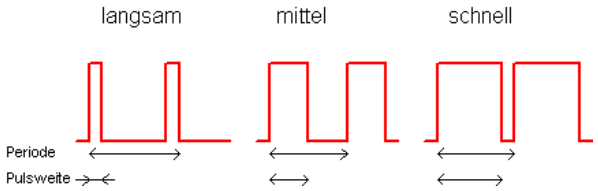
- Metallgehäuse
- Hohes Drehmoment und niedrige Achsdrehzahl
- 1,5 – 5Volt
- ~160 UPM bei 4,5V
- Günstig
- Modifikationen:
- Kurbelantrieb entfernen
- Achsen müssen gekürzt werden

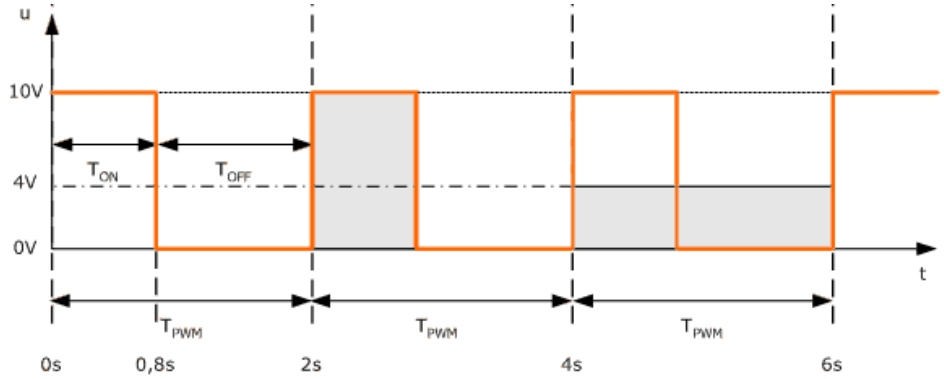


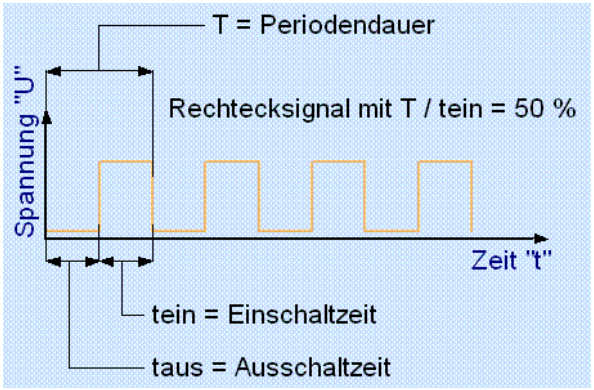


Pulsweiten Modulation PWM

- Ansteuerung von größeren Lasten (z.B. Motoren)
- Impulse mit voller Spannung, aber variabler Breite
- Rechtecksignal mit konstanter Frequenz
- Variables Tastverhältnis (duty cycle)
- Microcontroller haben spezielle Ausgänge
- Vorteil: Motor bekommt immer volle Spannung

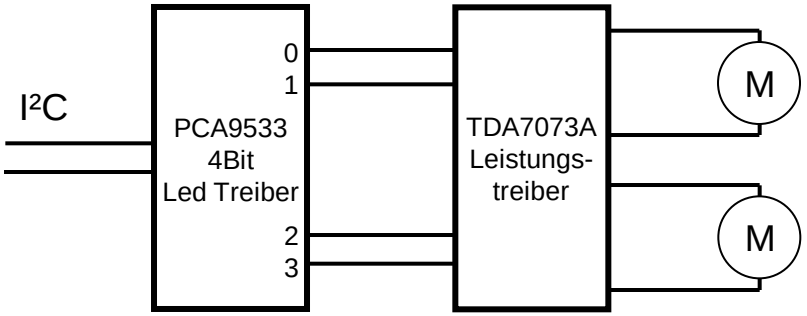






Motoransteuerung

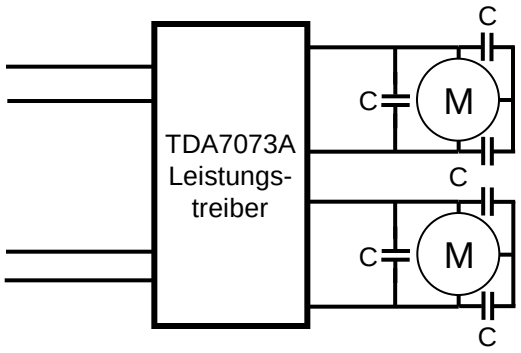
- Pulsweiten Modulation
- I²C Steuerung
- PCA9533 4Bit Led Treiber
- TDA7073 Leistungtreiber
- Geschwindigkeitssteuerung
- Sanftanlauf



	Fahrtrichtung		
Leitung	rechts	links	stop
0,2	PWM	(HiZ) 1	(HiZ) 1
1,3	(HiZ) 1	PWM	(HiZ) 1

Motorentstörung

- Motoren erzeugen Störimpulse
- **Abhilfe:**
- Wegfiltern, sonst auf der Versorgungspannung
- Kurze, verdrillte Kabel als Zuleitung
- Kondensatoren C=22nF
- Frisch geladene Akku‘s
- Software:
- Langsames Anfahren
- vorwärts – stopp – rückwärts



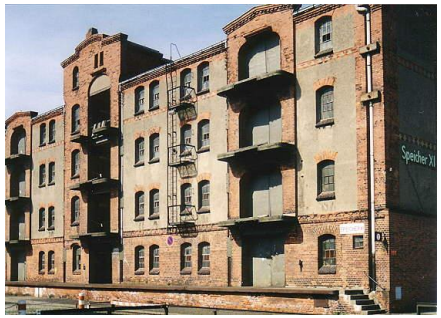
WWW Links Motoren

- PWM:
- http://www.strippenstrolch.de/1-3-13-pwm-pulsweitenmodulation.html
- http://www.mikrocontroller.net/articles/Pulsweitenmodulation
- Motoren:
- http://winkler.turbo.at/shop_de/product_info.php?cPath=1100_1197&products_id=8031
- http://www.roboternetz.de/wissen/index.php/Getriebemotoren_Ansteuerung

Fragen

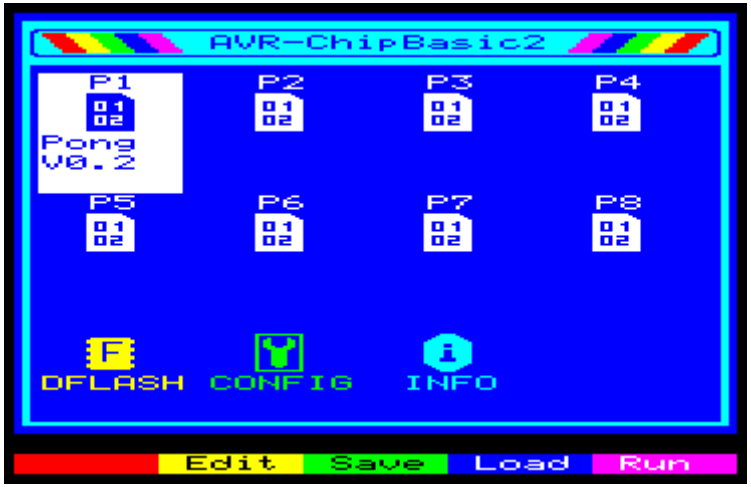
Speicher

- Interner Programmspeicher
- SPI Speicher AT45DB081D
- Serial Peripheral Interface (SPI Bus)
- I²C EEPROM 24C64
- I²C RTC mit Datenspeicher PCF2128



Interner Programmspeicher

- ATMEGA644
- 8 Programme a 95 Zeilen x 35 Zeichen
- Behält die Daten nach dem Ausschalten

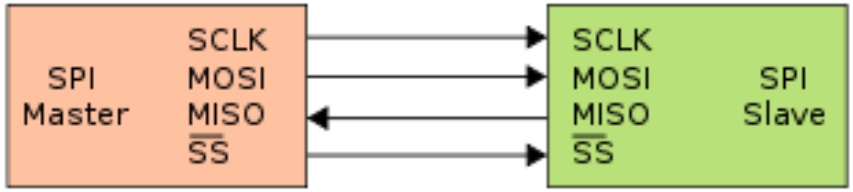


SPI Speicher AT45DB081D

- Programm- und Datenspeicher 4096 Pages a 256 Bytes
- Wird über den SPI Bus addressiert
- Wird durch das Betriebssystem formatiert
- Externer Programmspeicher
- Behält die Daten nach dem Ausschalten
- Chip select mit JP9 (gesteckt)

Serial Peripheral Interface – SPI Bus

- Motorola
- Serieller Bus
- Master / Slave



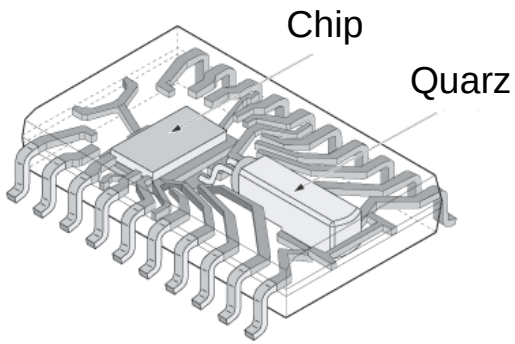
- SPI:
- http://de.wikipedia.org/wiki/Serial_Peripheral_Interface

I²C EEPROM 24C64

- Datenspeicher 8192 x 8Bit
- Wird über I²C adressiert (Adresse 0xA0)
- Behält die Daten nach dem Ausschalten

I²C RTC mit Datenspeicher PCF2128

- Real Time Clock
- Uhr mit Kalender und Speicher
- Alarm, Countdown Zähler, Timestamp … Funktionen
- Wird über den I²C Bus addressiert (Adresse 0xA2)
- Datenspeicher 512 Bytes
- Behält die Daten **nicht** nach dem Ausschalten!!!
- Backup Batterie



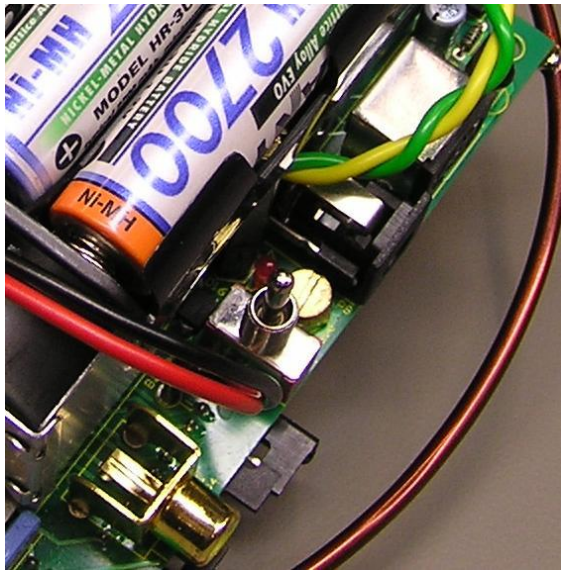
Fragen

Interfaces

- Spannungsversorgung
- RS232 serielle Schnittstelle
- I²C Bus Buchse
- SPI Buchse
- PS2 Tastatur Buchse
- Video Buchse

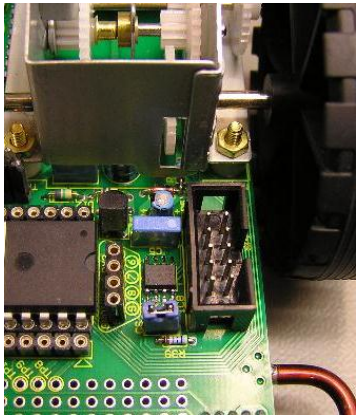
Stromversorgung

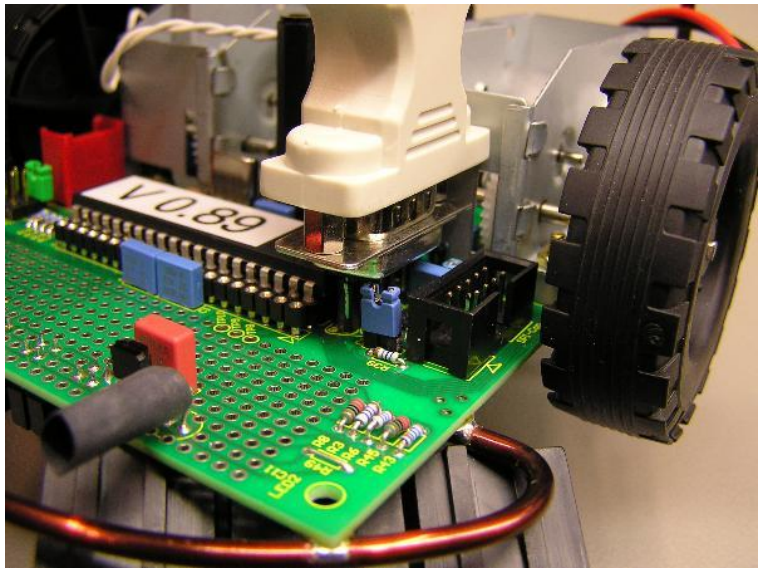
- 5 Volt System
- Netzteil: 5Volt / ~2Ampere Gleichspannung
- Akkubetrieb: 4 x 1,2V 2700mAh
- Einstellung über den Schalter
  - nach vorne Akkubetrieb
  - nach hinten Netzteilbetrieb
  - Mittelstellung Aus
- Akkus werden im Netzteilbetrieb nicht geladen



RS232 serielle Schnittstelle

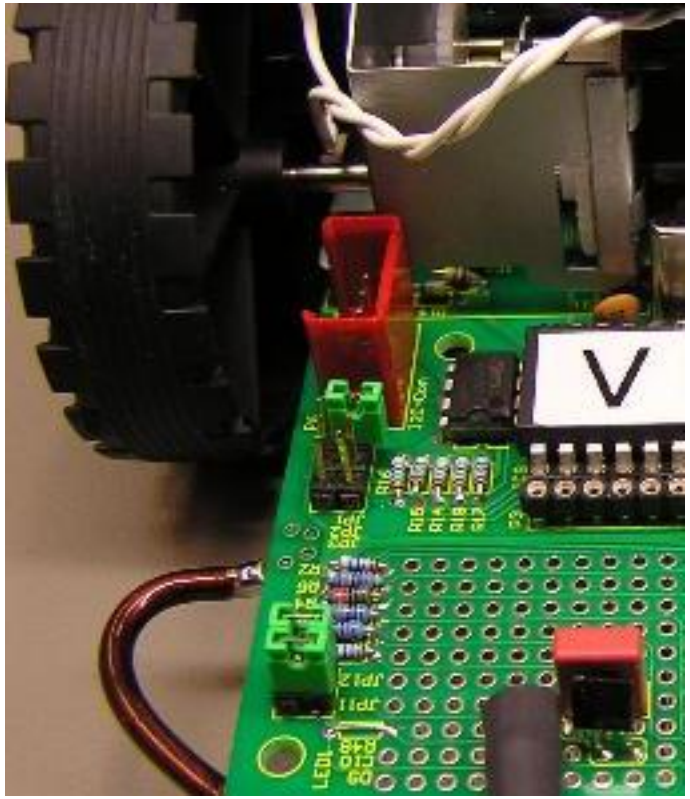
- Übertragung von Daten AVR-Mops ⇔ PC
- 1200, 2400 Baud
- 8 Bit
- No parity
- 1 Stop Bit
- Einstellung im Config Menu
- 3Pins neben dem Prozessor





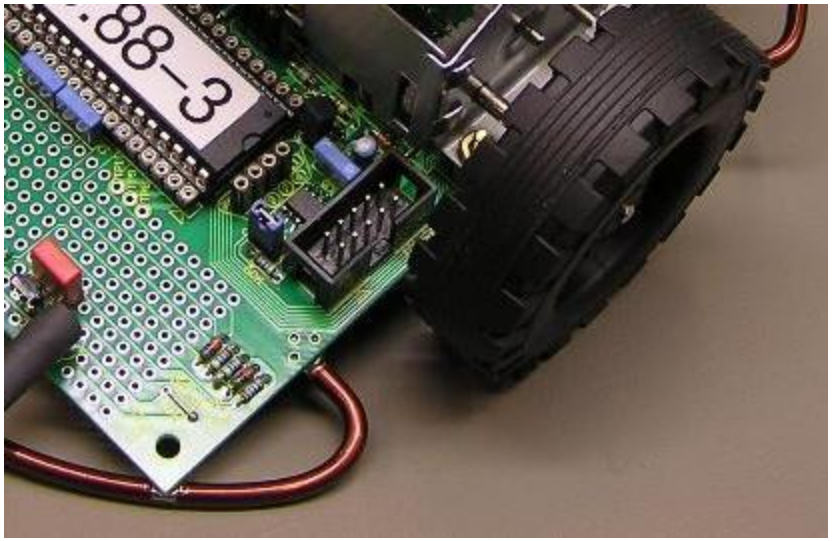
I²C Bus Buchse

- 4 pol. rote Stoko Buchse
- 5Volt Bus
- 100 / 400KHz Bustakt
- Einstellung im Config Menu



SPI Bus Buchse

- 10 pol. schwarze Wannenbuchse
- 156KHz / 5 MHz Bustakt
- Einstellung im Config Menu
- Clonen

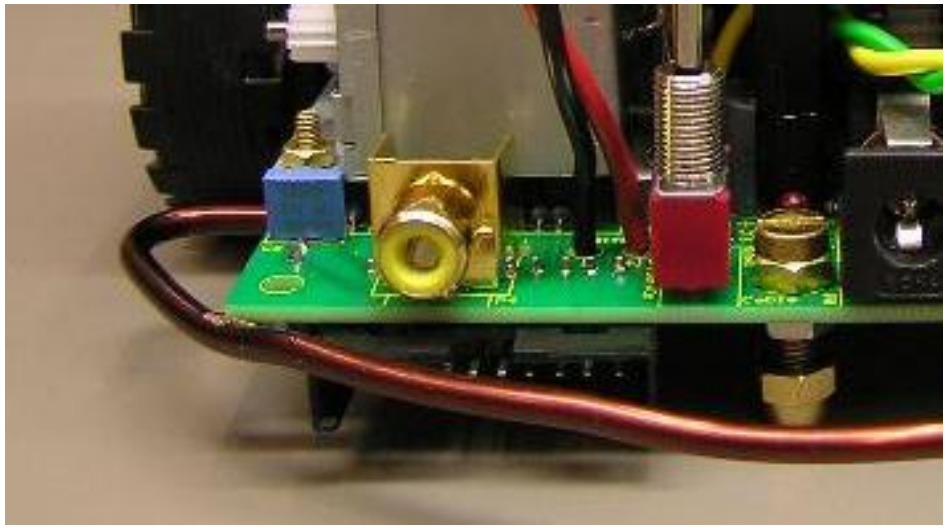


PS2 Tastatur Buchse

- MiniDin Buchse
- Deutsches oder US Tastaturlayout
- Einstellung im Config Menu

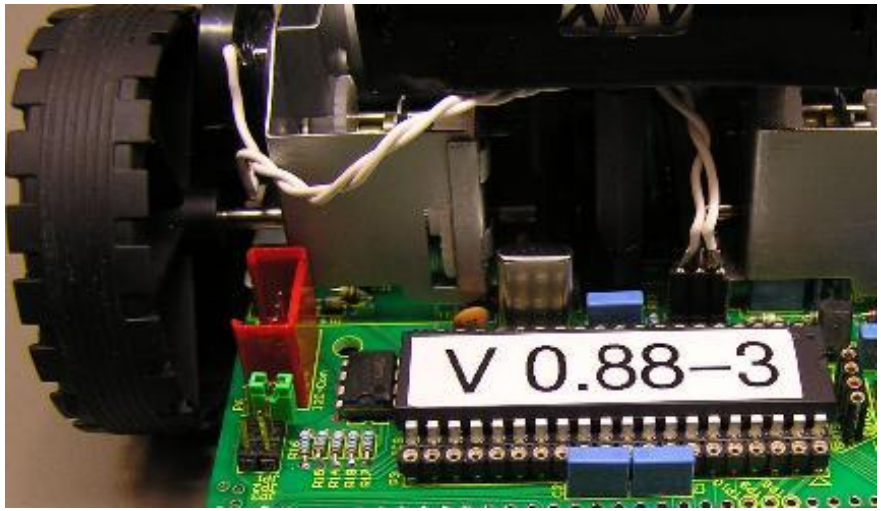
Video Buchse

- Bildschirmausgabe
- Gelbe Cinchbuchse CVBS (Composite Video Burst Sync) in schwarz/weiss
- 10 pol. Stocko Buchse TV RGB + Sync (nicht für VGA RGB)



Lautsprecher Buchse

- Tonausgabe
  - Lautsprecher
  - 10 polige AV Buchse

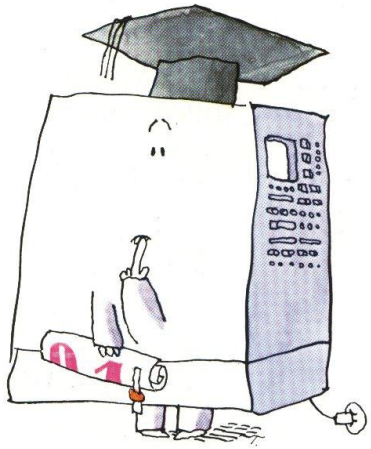


Fragen

Pause

Hardware Programmierung

- Programmiersprache - Befehle
- Sensoren Programmierung
- I²C Adressen der IC's
- Motor Programmierung
- RC5 Programmierung
- Uhr Programmierung



Programmiersprache - Befehle

- Variable: **A .. Z**
- Array: **AR(0 .. 1024)**
- Data: **DA 0, A, 5** Start, Wert1, Wert2
- Zuweisung: **L=42**
- Funktion: **ADC(0..7)** (Portpin)
- I²C Übertragung: **IC($A0, 0, 5)** (I²C Adresse, Start, Anzahl)
  - Schreiben DA ….. : IC(…)
  - Lesen IC(…) : AR(…)
- Befehl: **NOTE 10**
- Ganzer Umfang von Tiny Basic siehe Manual

Ton Programmierung

- NOTE n
- n = 0 .. 63 Dunkler Sound
- n = 64 .. 127 Heller Sound
- n = 255 Rauschen

- 10 NOTE 20

Sensor Programmierung

- Abstandssensor: 2 x IS471F — JP11 gesteckt!!!
  - Rechts = PA7, Links = PA6
- 10 R=ADC(7):L=ADC(6) — Ergebnis in Variable speichern
- Liniensensor: 3 x CNY70 — JP12 gesteckt!!!
  - Rechts = PA3, Mitte = PA5, Links = PA4
- 10 R =ADC(3):M=ADC(5):L=ADC(4) — Ergebnis in Variable speichern

I²C Adresstabelle

Motoransteuerung	PCA9533D01	0xC4
oder	PCA9533D02	0xC6
Speichererweiterung	EEPROM 24C64	0xA0
Uhr	PCF2128	0xA2
RC5 Fernbedienung	PCA8574AN	0x38
oder	PCA8574	0x40

Motor Programmierung

- Beide Motoren sollen vorwärts fahren
- Programmierung des PCA9533D01 I²C Adr = 0xC4

  oder PCA9533D02 I²C Adr = 0xC6
  - Control Register = 0x11 Auto increment flag und Register 1
  - PSC0 = 0 Frequenz Vorteiler 0 = 1/152
  - PWM0 = 0x80 Tastverhältnis 0 = 128/256
  - PSC1 = 0 Frequenz Vorteiler 1 = 1/152
  - PWM1 = 0x80 Tastverhältnis 1 = 128/256
  - LS0 = 0x33 Led0,Led2 = PWM, Led1,Led3 = (HiZ) 1

- 10 DA 0,$11,0,$80,0,$80,0,$33 Daten festlegen
- 20 IC $C6,0,7 7 Bytes übertragen

RC5 Programmierung

- RC5 Kommando auslesen
- Programmierung des PCA8574A I²C Adr = 0x38
- Programmierung des PCA8574 I²C Adr = 0x40

- 10 IC $39,0,1 1 Byte lesen vom I²C Baustein PCA8574A
- oder
- 10 IC $41,0,1 1 Byte lesen vom I²C Baustein PCA8574
- 20 K=AR(0) Ergebnis in Variable speichern

Uhr Programmierung

- Uhr stellen
- Programmierung des PCF2128 I^2C Adr = 0xA2

- 10 DA 0,0 — Subadresse = 0
- 11 DA 1,0,0,0 — Ctrl1=0, Ctrl2=0, Ctrl3=0
- 12 DA 4,0,$15,$11 — 11.15.00 Uhr einstellen

```
                  ^-----------------  Stunde
              ^---------------------  Minuten
          ^-----------------------------  Sekunden
```

- 13 DA 7,$17,$1,$11;$8 — 17 November 2008, Montag einstellen

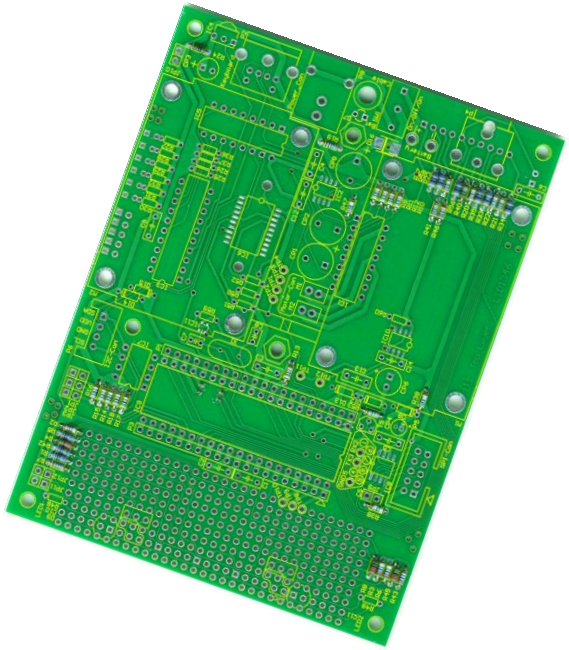
```
                        ^------  Jahr
                  ^--------------  Monat
              ^----------------------  Wochentag
          ^------------------------------  Tag
```

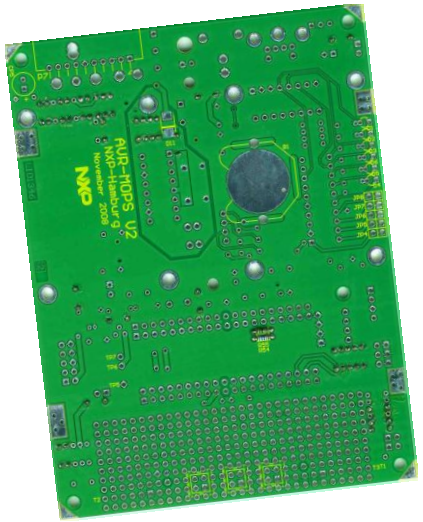
- 20 IC $A2,0,11 — 11 Bytes übertragen

Fragen

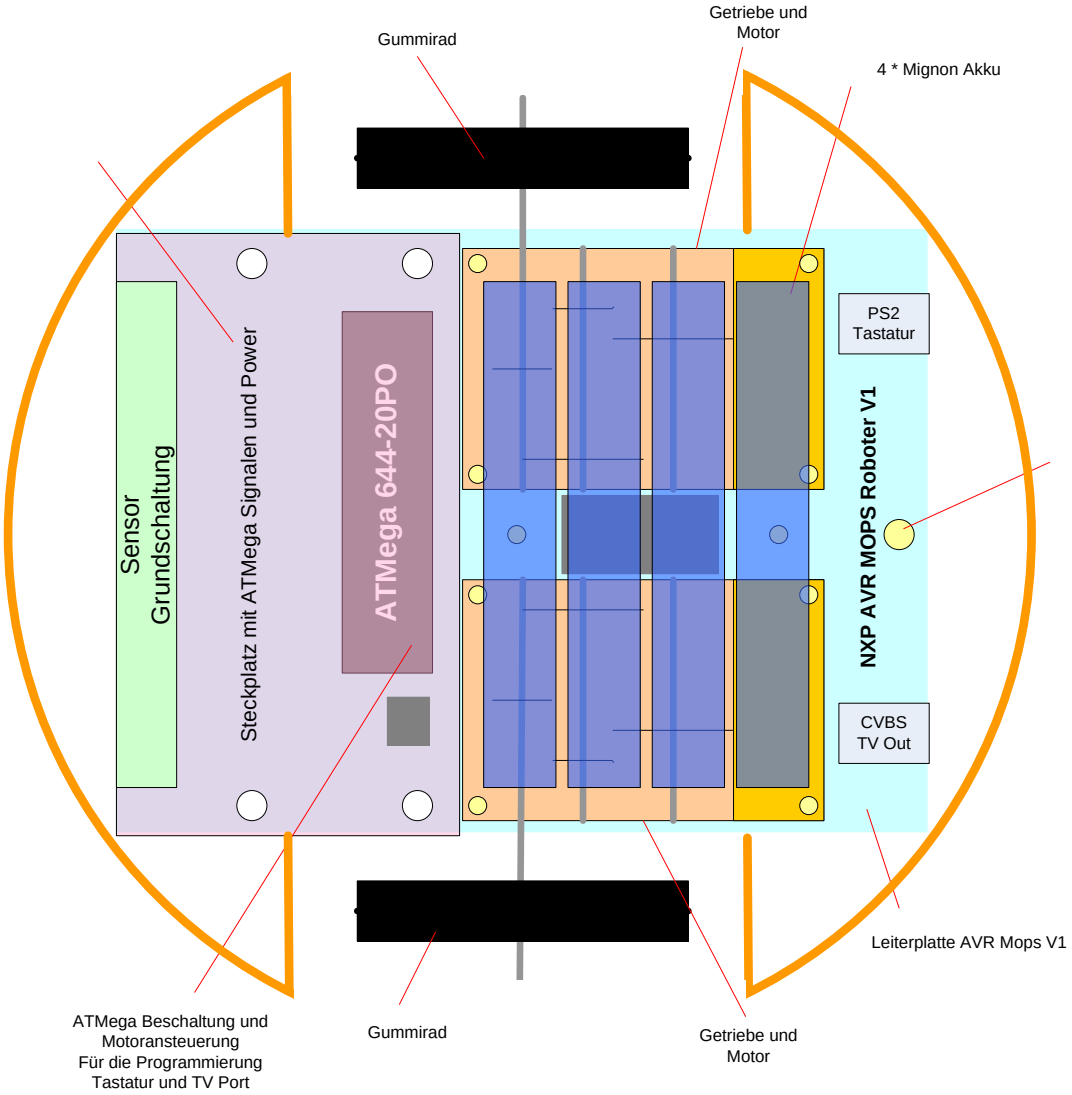
Elektrischer & mechanischer Aufbau

- Schematischer Aufbau
- Ansichten des Roboters



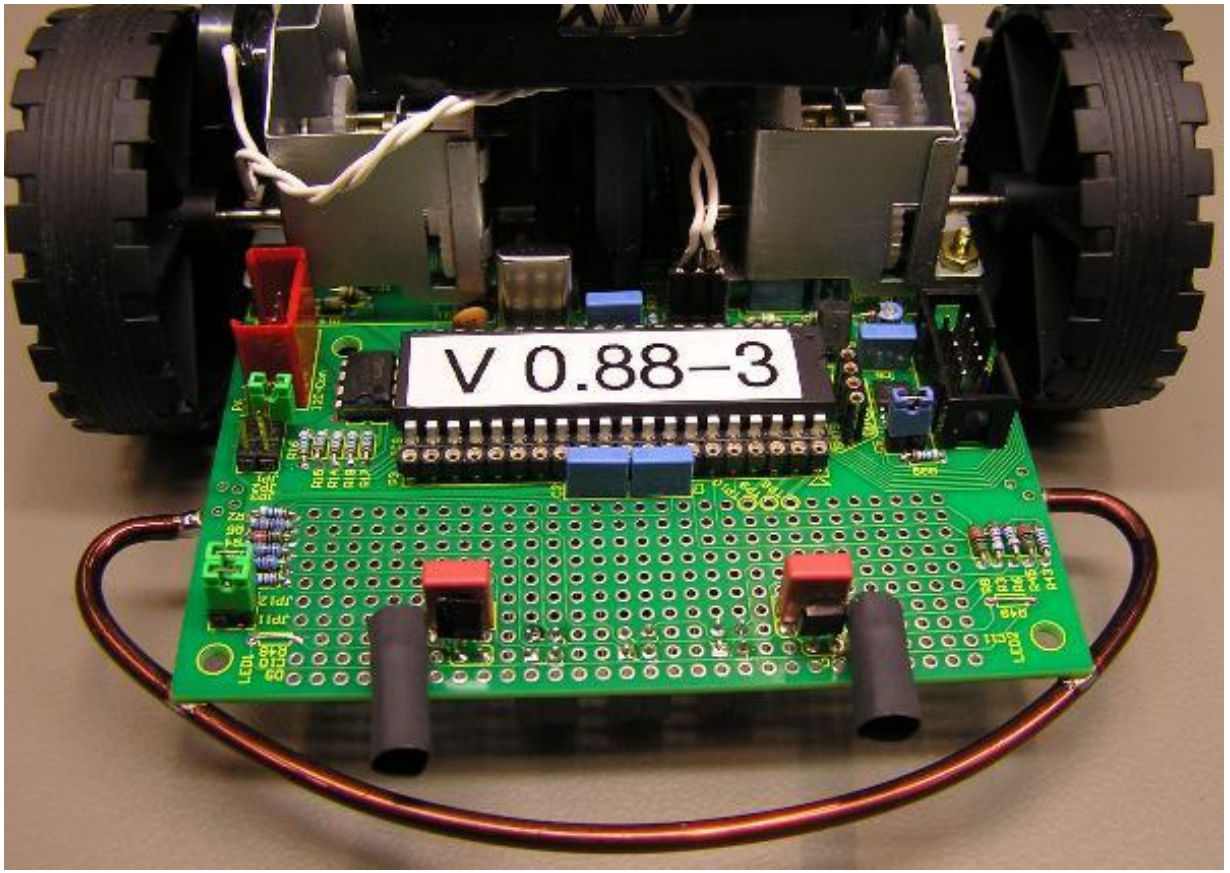


Schematischer Aufbau



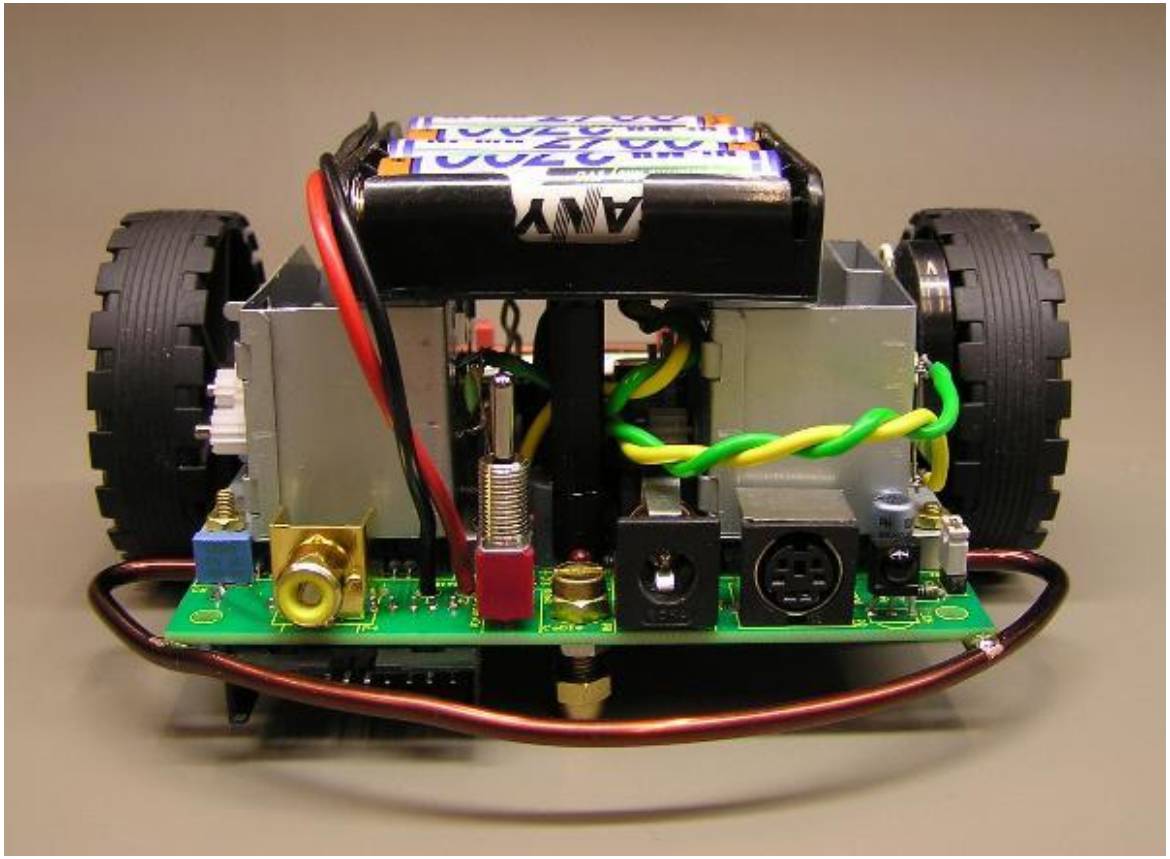
Ansicht - vorne

- Sensor Bereich
- Mikrocontroller
- I²C Buchse
- I²C EEPROM
- ISP / SPI Buchse
- SPI Speicher



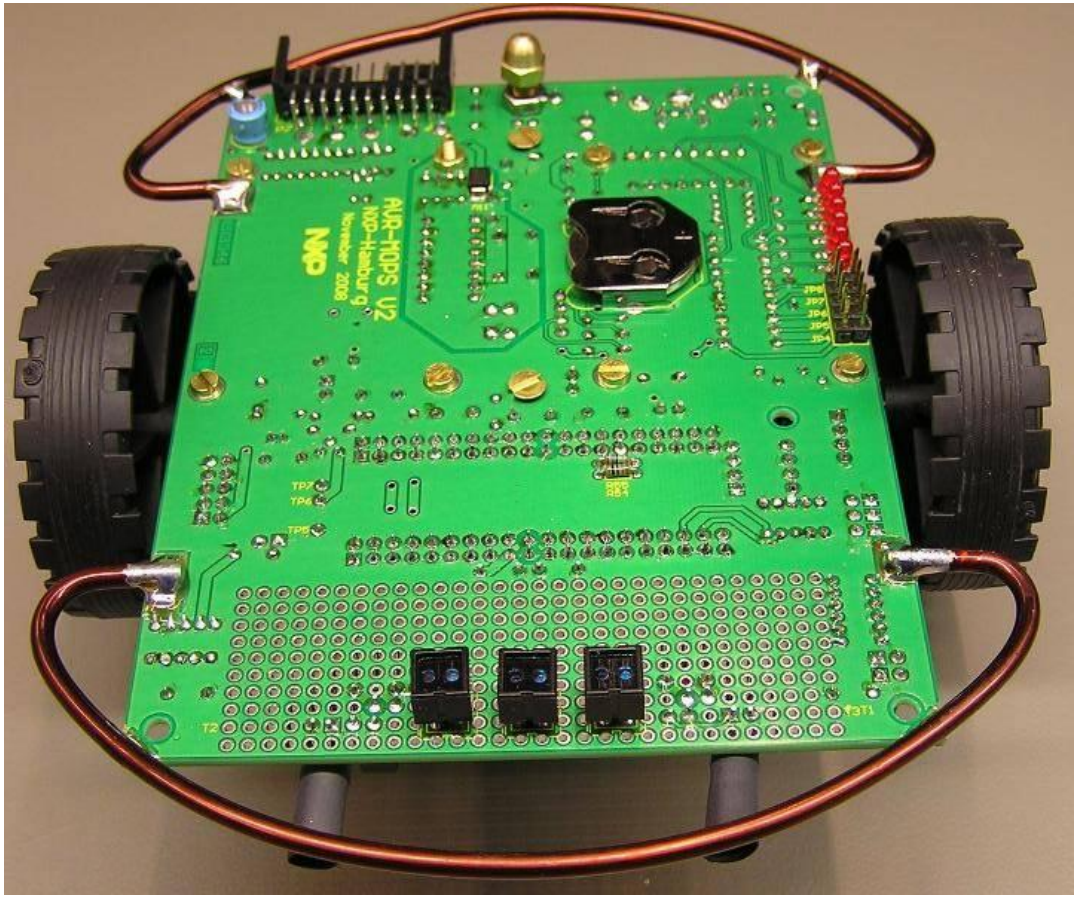
Ansicht - hinten

- Power Buchse
- Video Buchsen
- Tastatur Buchse
- RC5 Empfänger



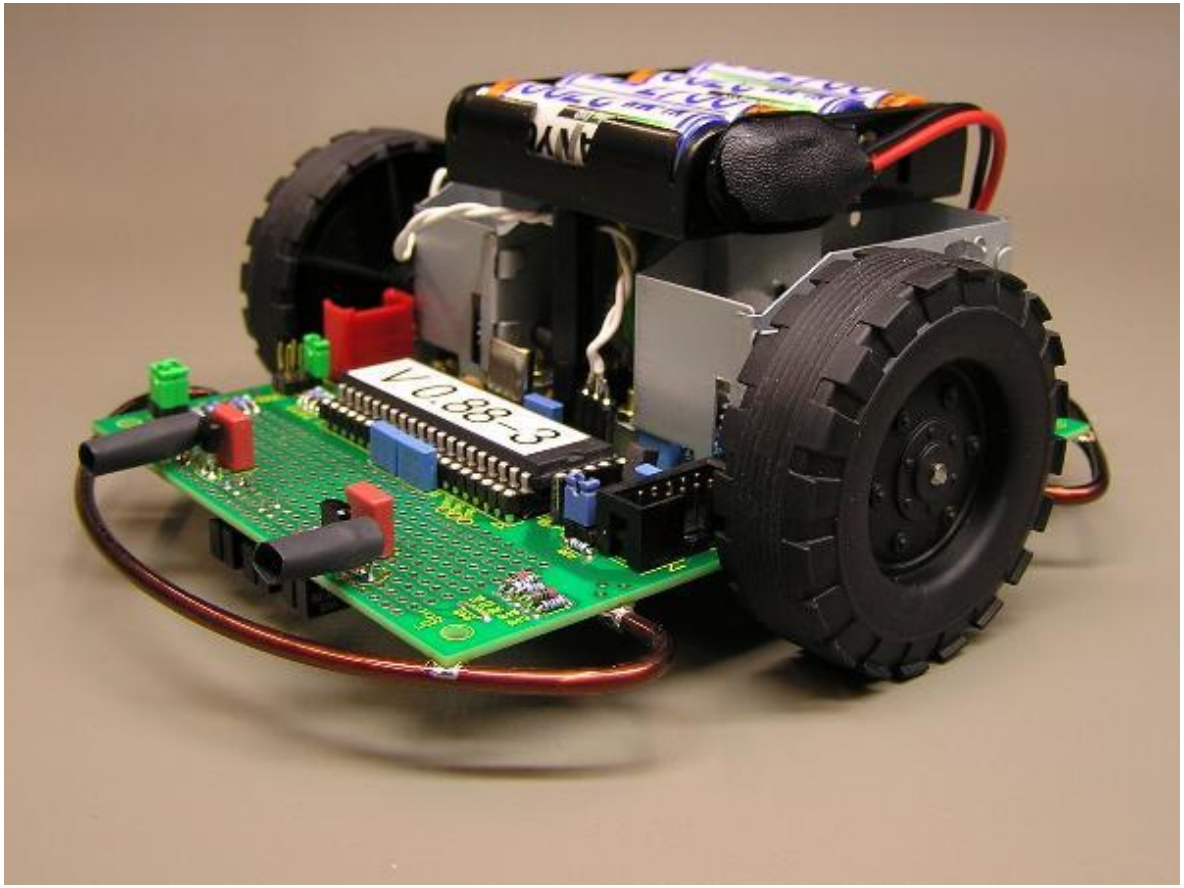
Ansicht - unten

- Backup Batterie
- RC5 Leds
- RC5 Adressen Jumper
- CNY70



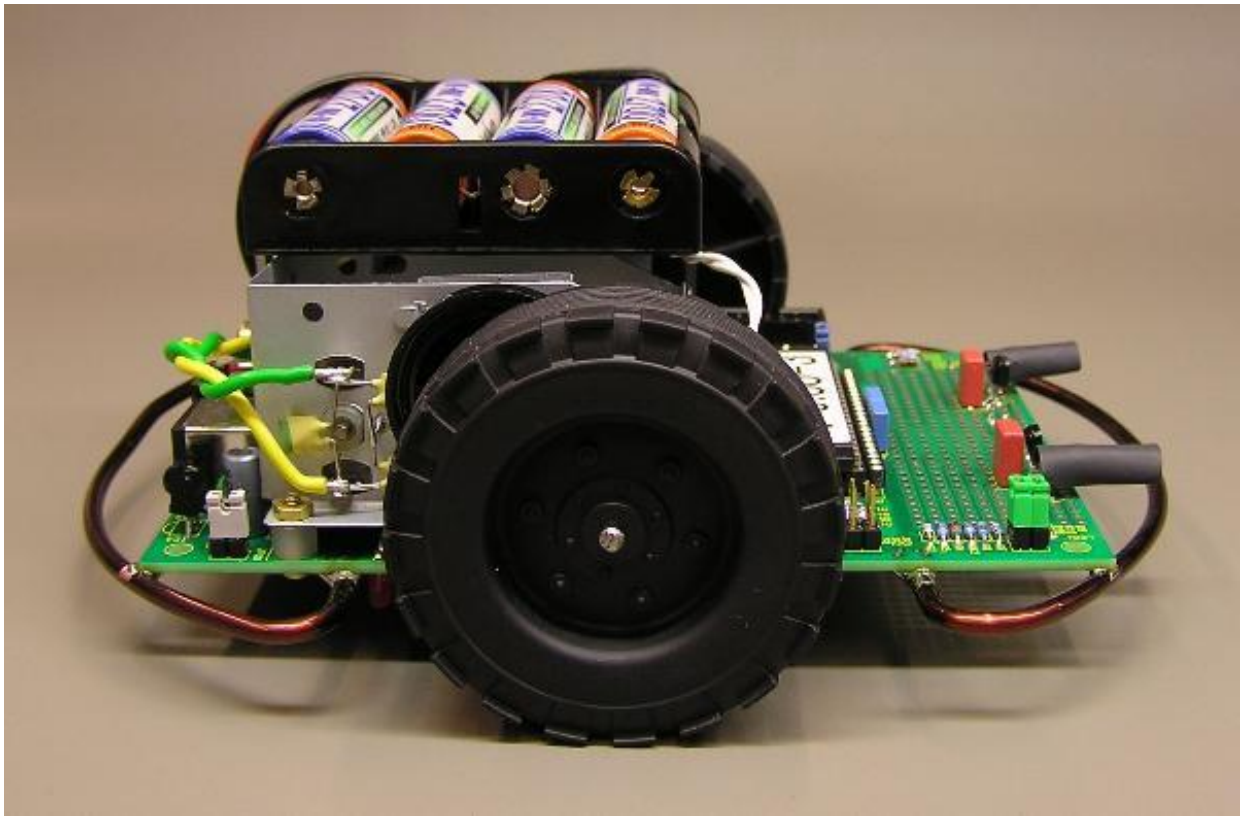
Ansicht - links

- Motor



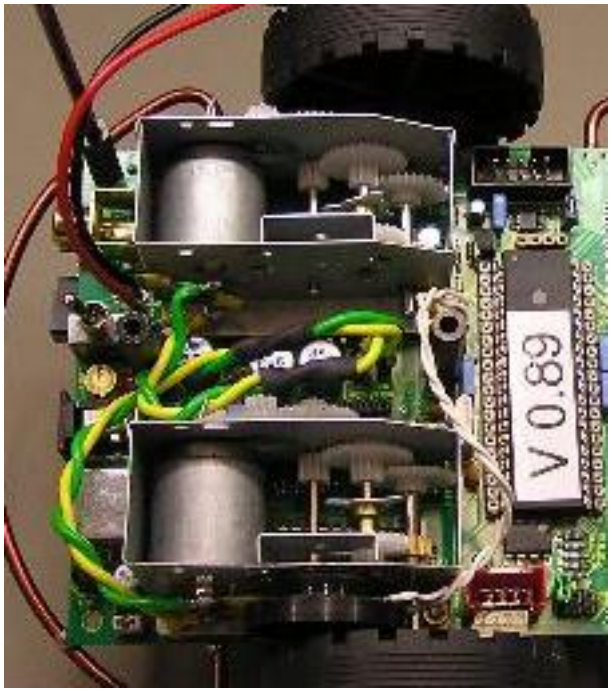
Ansicht - rechts

- Motor
- Entstörung



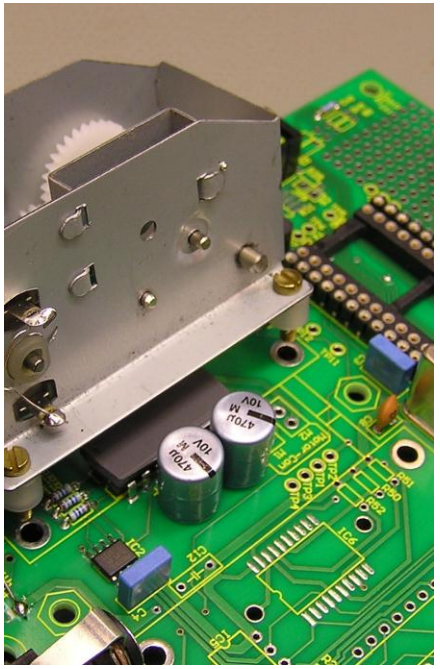
Ansicht - Motoren

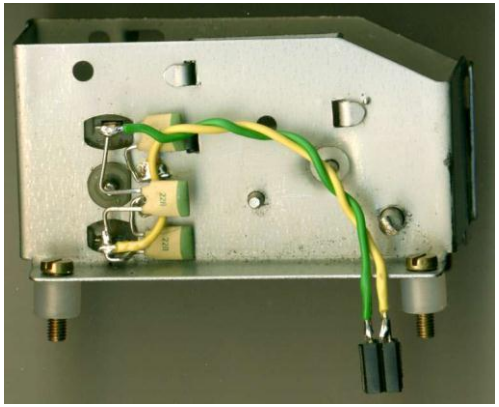
- Motoren
- Lautsprecher

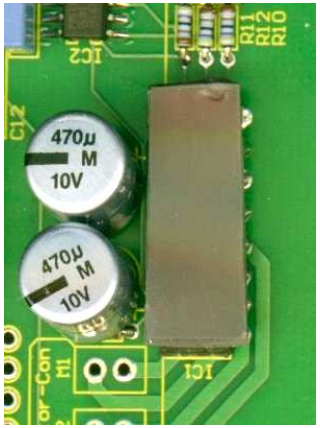


Ansicht – Motor Ansteuerung

- Leistungstreiber IC
- Wärmeleitfolie
- Entstörung

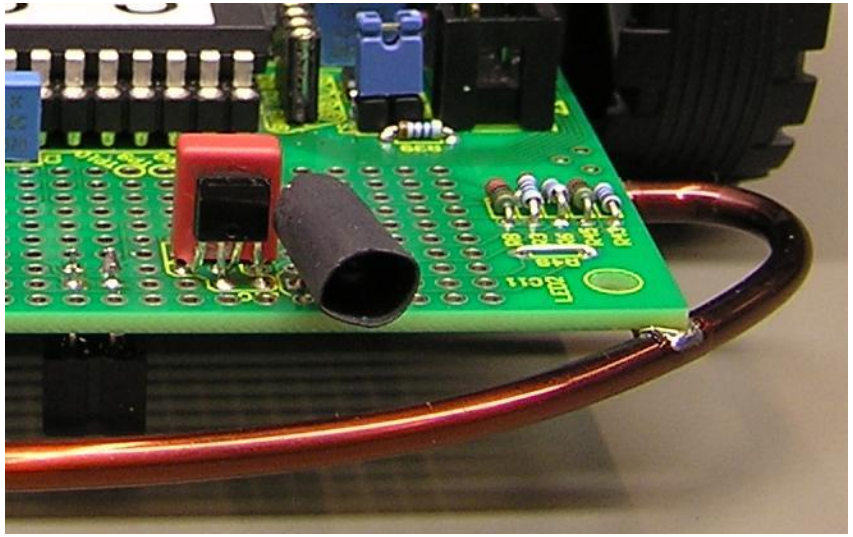


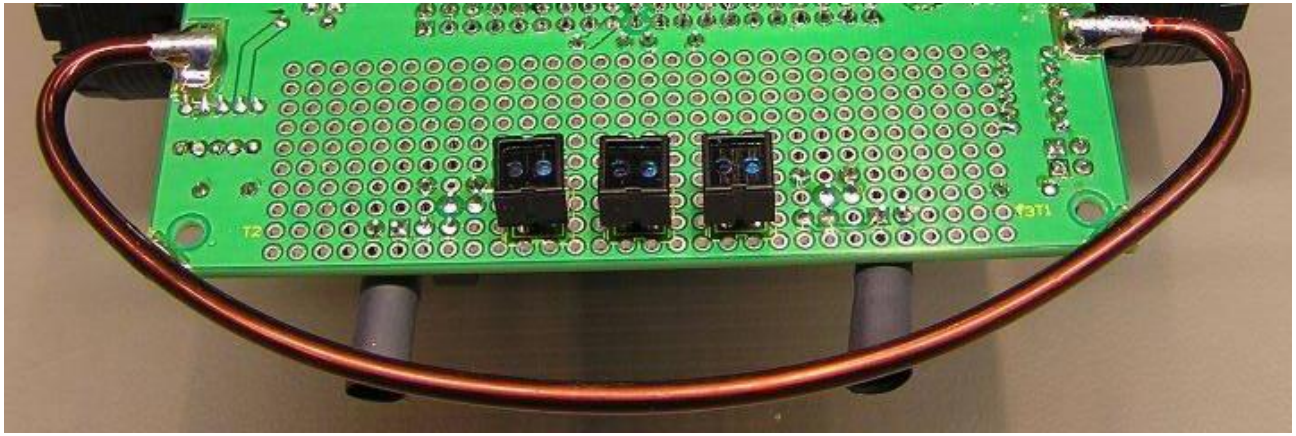




Ansicht - Sensoren

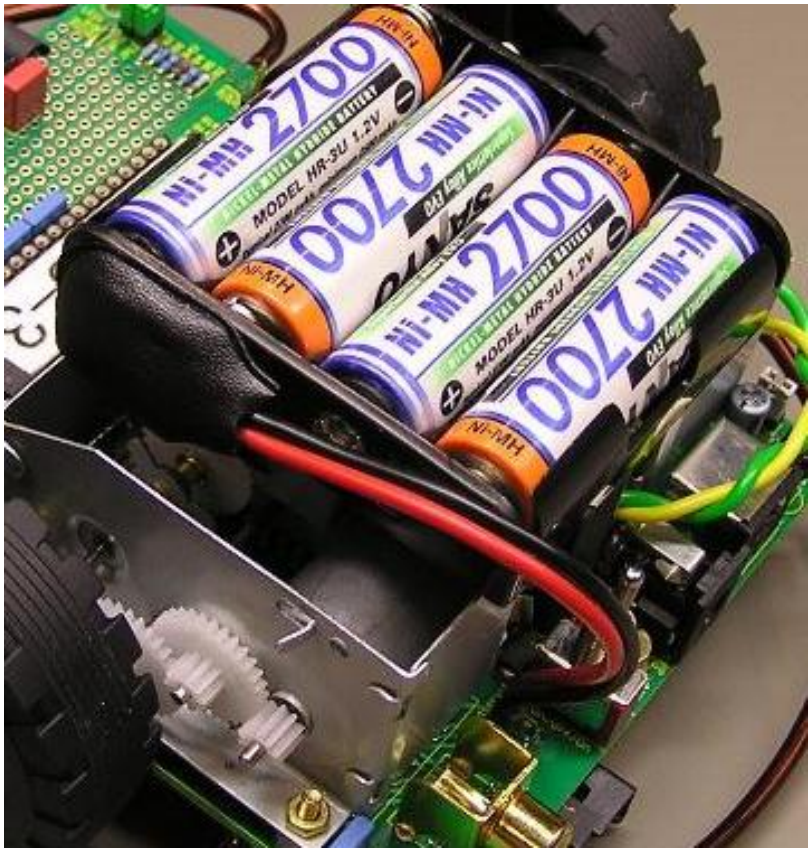
- IS471F
- LD274
- CNY70
- Stossstangen
- Schutz der Sensoren

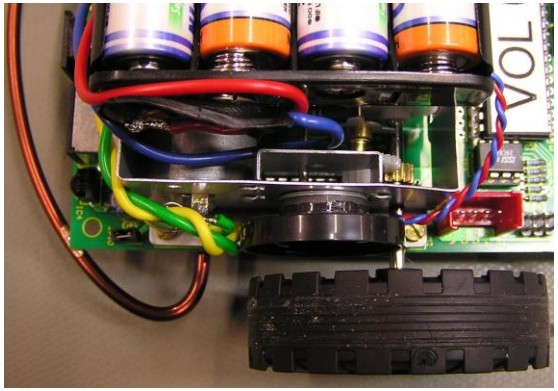




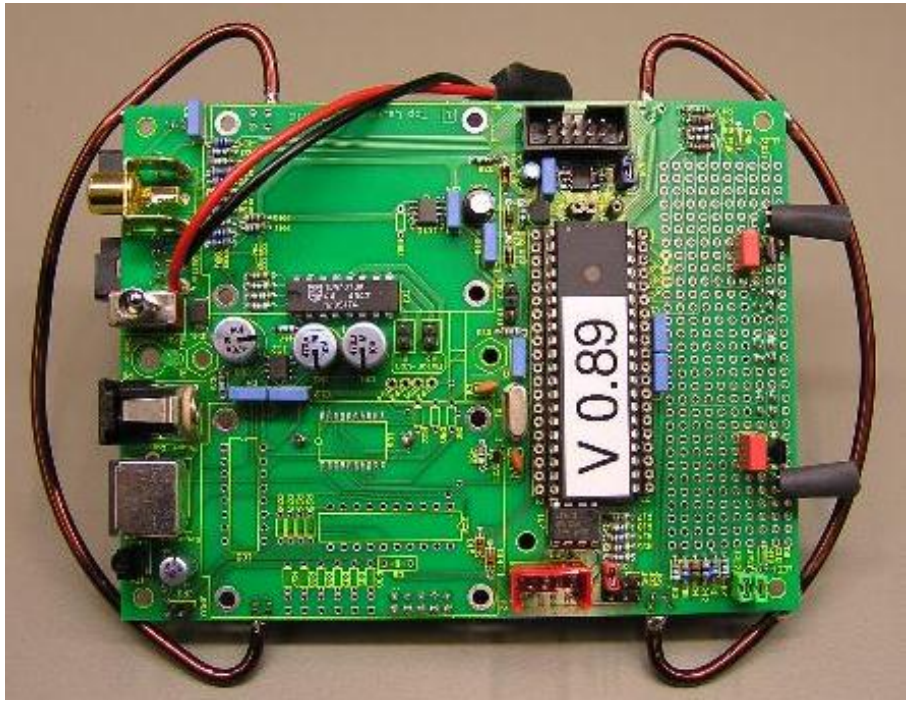
Ansicht – Batteriepack & Lautsprecher

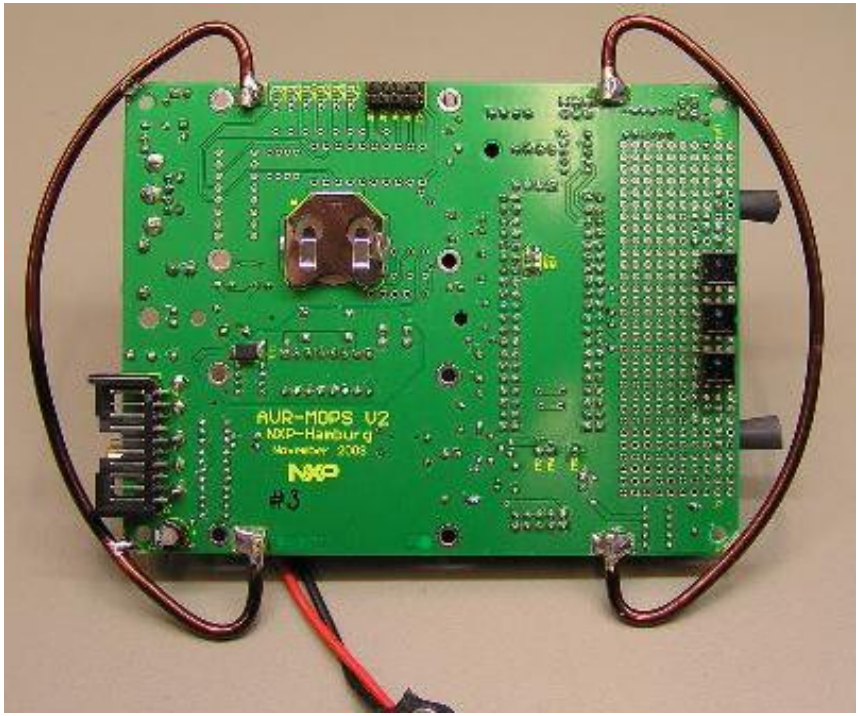
- Batterie Pack
- Kurze, dicke Leitungen
- Lautsprecher





Vorgefertigter Roboter





Fragen

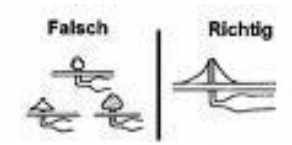
Mittag

Inbetriebnahme

- Optische Kontrolle
- AVR Test
- Betriebssystem aufspielen
- Serielle Schnittstelle Test
- Sensor Test
- Tone Test
- I²C Scan Test
- I²C EEPROM Test
- I²C Uhr Test
- I²C Motor Test
- I²C RC5 Test

Optische Kontrolle

- Bauteile
  - Richtige Ausrichtung der Bauteile?
    - Dioden, Led‘s, Elko‘s, IC‘s
  - Werte kontrollieren
  - Motoren richtig verdrahtet?
  - Batterieanschluss richtig? Keine Akku‘s einlegen !!!
- Lötstellen
  - Alle Lötstellen gelötet?
  - Kurzschlüsse?
  - Saubere Lötstellen?
- Jumper
  - Sind die Jumper richtig gesetzt?
    - DFLASH vorhanden? = JP9 setzen
    - RC5 vorhanden? = JP10 setzen
    - IS471F vorhanden? = JP11 setzen
    - CNY70 vorhanden? = JP12 setzen



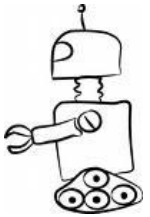
AVR Test

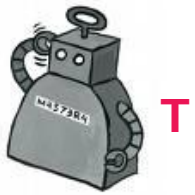
- AVR Test
  - Netzgerät auf 5V und 1A stellen
  - AVR-Mops und Netzteil mit Versorgungskabel verbinden
  - Umschalter auf Powerbuchse stellen
  - Power Led leuchtet
  - Strom am Netzteil ablesen (0,1- 0,17A)

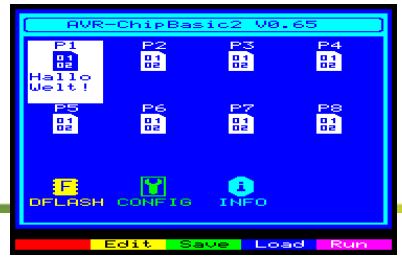
  - Versorgungskabel lösen
  - Geladene Akkus einlegen (Richtung beachten)!!!
  - Spannung kontrollieren (~ 5V an dem Batteriepack)
  - Umschalter auf Batterie stellen
  - Power Led leuchtet

Betriebssystem aufspielen

- AVR Mastersystem (Host) mit Netzteil versorgen
- Neues AVR System (Target) mit Überspielkabel verbinden
  - Überspielkabel ist mit H und T gekennzeichnet
  - Neues AVR System nicht mit Spannung versorgen!!!
- Im CONFIG Menue die ***CLONE Funktion*** starten
- Neues AVR System wird programmiert
- Überspielkabel lösen
- Neues AVR System mit Spannung versorgen
- Auf dem Bildschirm erscheint der Start Bildschirm
- !!!**Achtung**!!! ATmega644 <> ATmega644P







Serielle Schnittstelle Test I

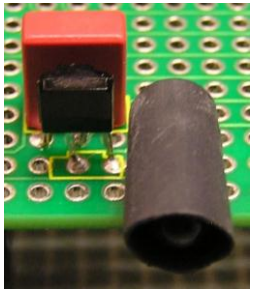
- AVR-Mops mit dem PC verbinden
  - 9 pol. Buchse mit Stiften auf den Mops stecken
  - 9 pol. Verbindungskabel (gekreuzt) Buchse – Buchse
  - Wenn der PC keine serielle Schnittstelle mehr hat =>
    USB – serielle Schnittstelle Konverter verwenden!
- Auf dem PC das Programm *Hyperterminal* starten
  - Einstellung: 2400Baud – 8Bit – no parity – 1 Stopbit – no flow control
  - ***Text aufzeichnen*** starten
- Auf dem AVR-Mops zu speicherndes Programm auswählen
- Editor (F1) aufrufen
- Übertragung zum PC mit CTRL und F2 starten

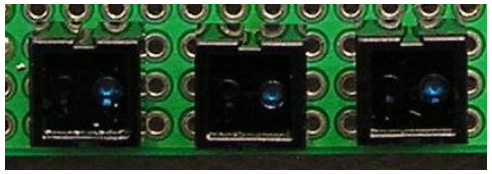
Serielle Schnittstelle Test II

- AVR-Mops mit dem PC verbinden
  - 9 pol. Buchse mit Stiften auf den Mops stecken
  - 9 pol. Verbindungskabel (gekreuzt) Buchse – Buchse
  - Wenn der PC keine serielle Schnittstelle mehr hat => USB – serielle Schnittstelle Konverter verwenden!
- Auf dem AVR-Mops zu ladendes Programm auswählen
- Editor (F1) aufrufen
- AVR-Mops mit CTRL und F3 vorbereiten
- Auf dem PC das Programm ***Hyperterminal*** starten
  - Einstellung: 2400Baud – 8Bit – no parity – 1 Stopbit – no flow control
  - ***Text senden*** starten
  - Programm wird übertragen

Sensor Test

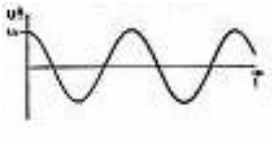
- Auf dem AVR-Mops müssen JP11 und JP12 gesteckt sein!
- ***Sensor Test*** starten
- Werte für die Sensoren werden ausgeben 0 - 1024
- Abstandssensor IS471F / LD274 reagiert auf Hindernisse
- Liniensensoren CNY70 reagieren auf schwarz / weiß Änderungen





Tone Test

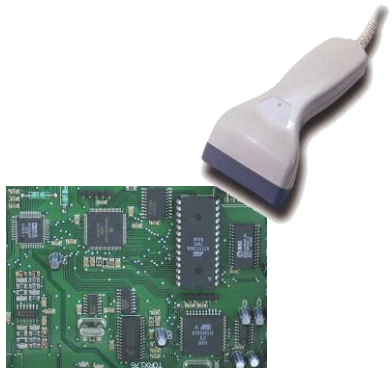
- Auf dem AVR-Mops muss ein Lautsprecher angeschlossen sein
- ***Tone Test*** starten
- Es werden 2 x 64 Töne und 1 x Rauschen ausgegeben



I²C Scan Test

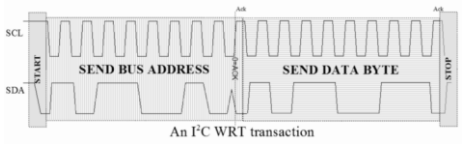
- ***I2C Scan*** starten
- Anzeige aller I²C Devices auf dem Mops
- Ausgabe der I²C Read Adressen
- Bei voller Bestückung werden folgende Adressen angezeigt:

0x38 (0x40), 0xA0, 0xA2, 0xC4 (0xC6)



I²C EEPROM Test

- Auf dem AVR-Mops muss das IC7 bestückt sein
- I²C Adresse = 0xA0
- *Xpeek*, *Xpoke* funktionieren nur auf I²C Adresse 0xA2 !!! (V 0.75)
- In der Version 0.88 kann man über *Config* die Adresse 0 einstellen
- ***EEPROM Test*** starten
- 256 Bytes werden geschrieben
- 256 Bytes werden gelesen und verglichen



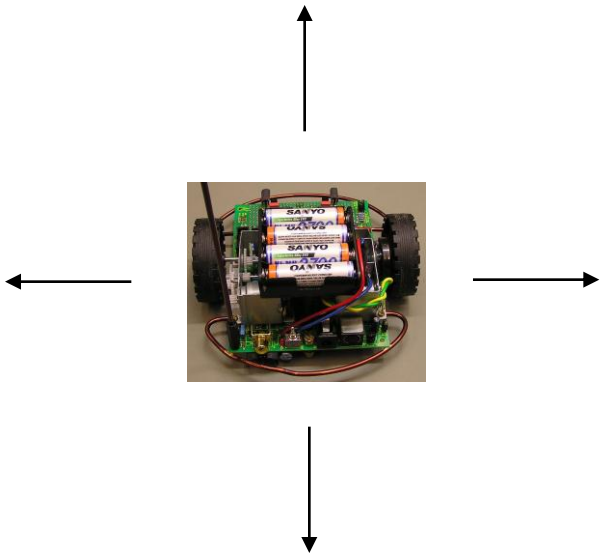
An I²C WRT transaction

I²C Uhr Test

- IC6 und die Batterie auf der Unterseite müssen bestückt sein
- I²C Adresse = 0xA2
- ***Uhr Test*** starten
- Menu mit: s = Uhrzeit aus Programmkode übernehmen
  u = Uhrzeit anzeigen
  r = RAM Test
  e = Ende
- Stellen der Uhr über setzen der Variablen im Programm
- Uhrzeit, Datum, Wochentag und Timestamp werden ausgegeben
- Timestamp Funktion wird durch TP2 ausgelöst (TP2 = GND)

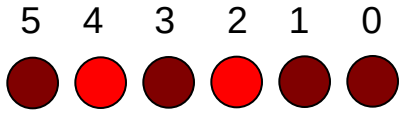
I²C Motor Test

- Motoren richtig angeschlossen?
- Kondensatoren am Motor angelötet?
- ***Motor Test*** starten
- Cursor Tasten steuern den AVR-Mops
- Page up / down steuert die Geschwindigkeit
- S Taste stoppt den AVR-Mops
- I²C Adresse richtig gesetzt !



I²C RC5 Test

- Auf dem AVR-Mops muss JP10 gesteckt sein! Led Anzeige
- AVR-Mops Geräte Adresse einstellen JP4 .. JP8 = 5Bit
- Geräte Adresse 0 = TV
- Geräte Adresse auf dem IR Fernbedienungssender einstellen
- Beide Adressen müssen gleich sein!!!
- ***RC5 Test*** starten
- RC5 Kommando wird angezeigt = 6Bit



Fragen

Programme

- RC5 Motor
- Mops Test
- Linie
- Tipps & Tricks

mops.bas

```
PROGRAM 5: Mops  Test
01
02 GOS 40
03 GOS 20
04 L=ADC(6):R=ADC(7)
05 IF (L<500)#(R<500) GOS 50
06
07 P=KEY(0):IF P<>$ED GO 3
08 GOS 35
09 END
10
11
12
13
14
15
16
17
18
19
20 'Motoren vorwaerts
21 DA 0,$15,$33
22 IC $C6,0,2
23 RET
24
25 'Robo links zurueck
26 DA 0,$15,$C0
27 IC $C6,0,2
28 RET
29
30
31
32
33
34
35 'Motoren stop
36 DA 0,$15,0
37 IC $C6,0,2
38 RET
39
40 'PCA9533 init
41 DA 0,$11,0,$FF,0,$FF,0
42 IC $C6,0,6
43 RET
```


RC5 Motor Programm

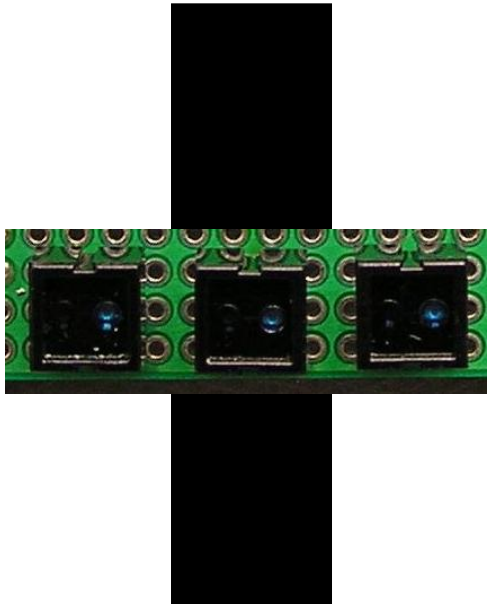
- Der AVR-Mops wird über die Fernbedienung gesteuert
- ***RC5 Motor*** starten
- Taste 2 = vorwärts
- Taste 8 = rückwärts
- Taste 5 = stopp
- Taste 6 = rechts
- Taste 4 = links
- Taste * = schneller
- Taste # = langsamer
- Taste 0 = Ende

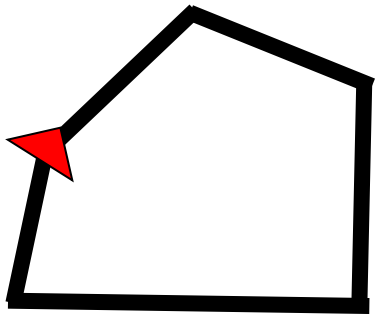
Mops Programm

- Auf dem AVR-Mops muss JP11 gesteckt sein!
- ***Mops Test*** starten
- Der AVR-Mops fährt vorwärts bis er auf ein Hindernis trifft
- Dann stoppen beide Motoren
- Der Linke dreht dann einige Umdrehungen rückwärts
- Damit weicht er dem Hindernis aus
- Der Motor wird gestoppt
- Nun drehen beide Motoren wieder vorwärts

Linie Programm

- Auf dem AVR-Mops muss JP12 gesteckt sein!
- ***Linie*** starten
- Der AVR-Mops fährt vorwärts eine schwarzen Linie nach





Tipps & Tricks

Shared Technology Infrastructure STI

- Zeilen ausnutzen — 10 Befehl1:Befehl2:Befehl3 …
Platz für BREAK lassen
- Befehle abkürzen — GOS für GOSUB
- Routinen schreiben — Routinen nur einmal schreiben
Aufruf mit GOS Programmnr. , Zeilennr.

- **!!! Vorsicht !!!**
- Einfügen in einer Zeile — Zeichen an Position 35 wird rausgeschoben
- Alt+Einfg — Zeile 95 wird rausgeschoben
- Alt+Einfg, Alt+Entf — GO, GOS Zeilennummern werden nicht geändert

Fragen

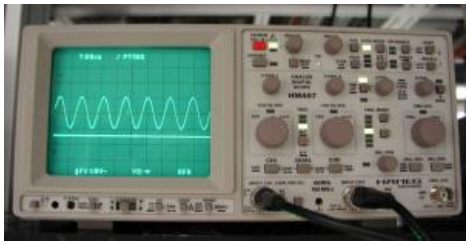
Pause

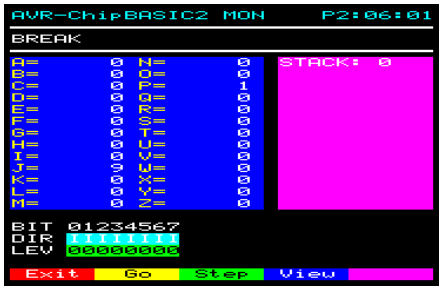
Anregungen / Erweiterungen

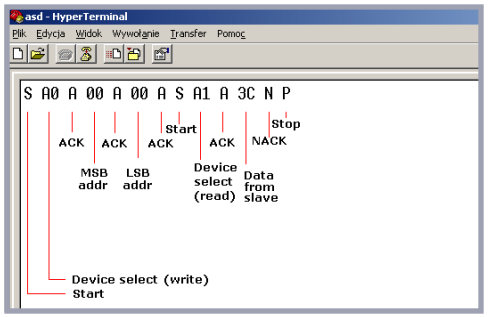
- Debug
- I²C Sniffer
- I²C Sniffer Installation
- I²C Verteiler
- Screendump erzeugen
- Neues Betriebssystem aufspielen
- Burn-o-mat
- Fuses einstellen
- USBASP
- AVR-Mops Erweiterungen

Debug

- Hardware
  - Multimeter
    - Spannung richtig ?
    - Zu viel Strom ?
    - Kurzschluß ?
    - Lötbrücke ?
  - Oszilloskop
    - Kurven richtig ?
    - Frequenzen richtig ?
- Software
  - BREAK Kommando in der Software
  - RC5 LED‘s
    - IR Kommando richtig ?
  - I²C Sniffer
    - I²C Daten richtig ?

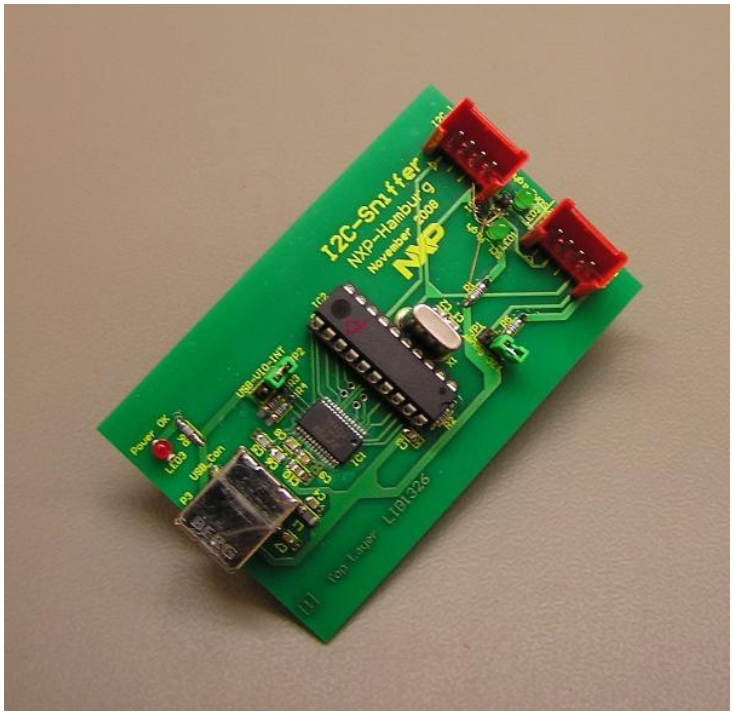






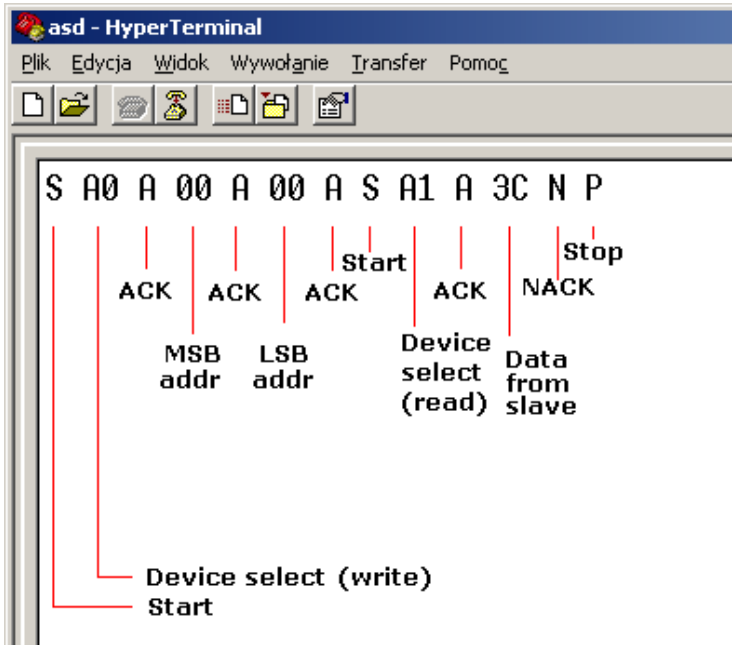
I²C Sniffer

- I²C Bus Logger
- Radosław Kwiecień http.//en.radzio.dxp.pl
- Anzeige der SDA / SCL Leitung



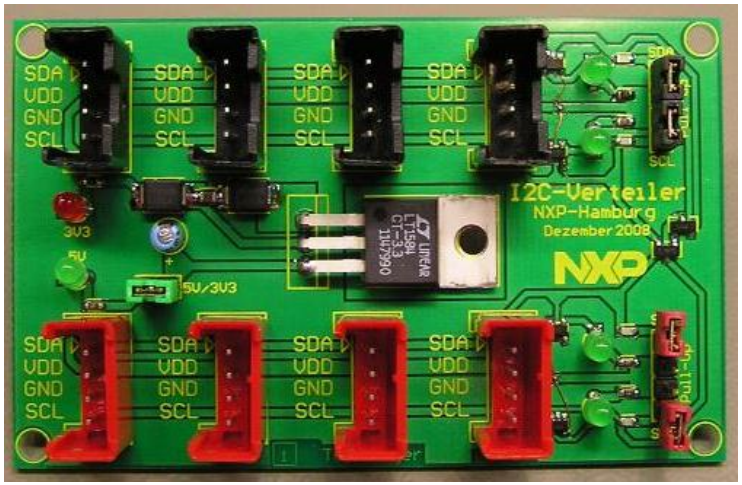
I²C Sniffer Installation

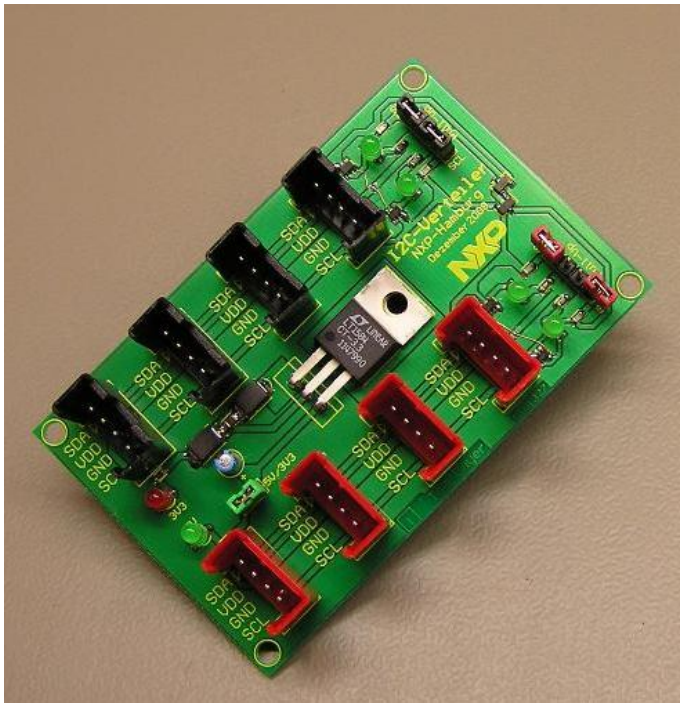
- USB Kabel PC ⇔ Sniffer
- Softwaretreiber installieren
  - Treiber unter /I2C Sniffer/Treiber
- Neue COM Schnittstelle
  - Comport Nummer unter Systemsteuerung/Device Manager /Ports nachsehen
- ***Hyperterminal*** starten
- Einstellungen
  - Connect to Connect using COMx
  - 11520 Baud / 8N1 / Flow control „None“
- Siehe Anzeige



I²C Verteiler

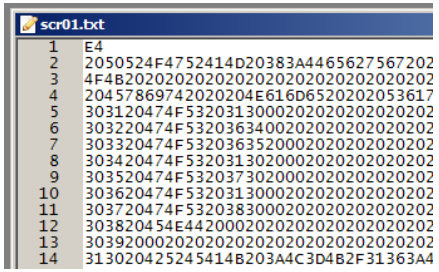
- Anschlussmöglichkeit weiterer Sensoren
- 5V / 3,3V Bus Verteiler
- Anzeige der SDA / SCL Leitung
- Schaltbare Pullup Widerstände
- 3,3V Erzeugung

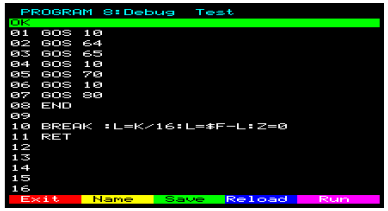




Screendump erzeugen

- Software installieren:
  - Zu finden unter /AVR_Mops/AVR/Tools
  - Perl für Windows http://strawberryperl.com/
    - Perl testen im cmd-Fenster mit „perl –v“
  - ImageMagick http://www.imagemagick.org/script/index.php
- PC:
  - AVR-Mops mit seriellem Kabel am COM Port anschliessen
  - ***Hyperterminal*** starten
    - Einstellungen:
      Connect to Connect using COMx
      11520 Baud / 8N1 / Flow control „None“
  - Textdatei aufzeichnen z.B. scr01.txt im Verzeichnis von konvert.pl
- AVR-Mops:
  - Rechte STRG + P drücken
  - Screendump wird übertragen
- PC:
  - ctab_0.dat mit ins Verzeichnis kopieren wo konvert.pl liegt
  - Konvert.pl editieren (Verzeichnis von ctab_0.dat)
  - Cmd Shell öffnen und in das Verzeichnis von konvert.pl gehen
  - Perl konvert.pl scr01.txt
  - Convert temp.xpm scr01.png



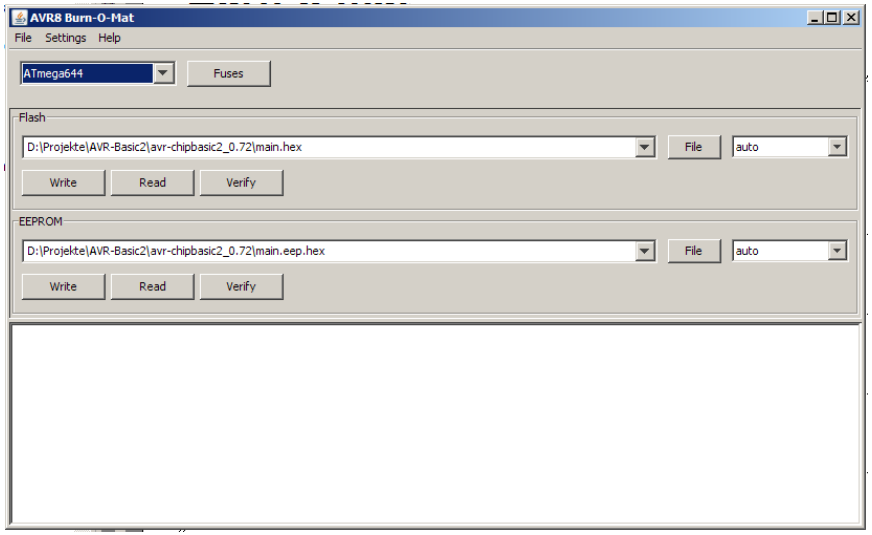


Neues Betriebssystem aufspielen

- USB2ASP am USB Port anschliessen
  - Treiber installieren (/USB2ASP/Treiber 0.1.12…)
  - Eintrag im Device Manager unter „LibUSB Win32 Devices“ prüfen
  - USB2ASP vom USB entfernen und erneut anschließen
- Burn-o-mat installieren / starten
  - Falls nicht vorhanden Java installieren (\Burn-O-Mat\AVRDude\jre…)
  - \AVR8_Burn-O-Mat\AVR8_Burn_O_Mat.jar ausführen
- USBASP mit AVR-MOPS über ISP Schnittstelle mit 1:1 Kabel verbinden

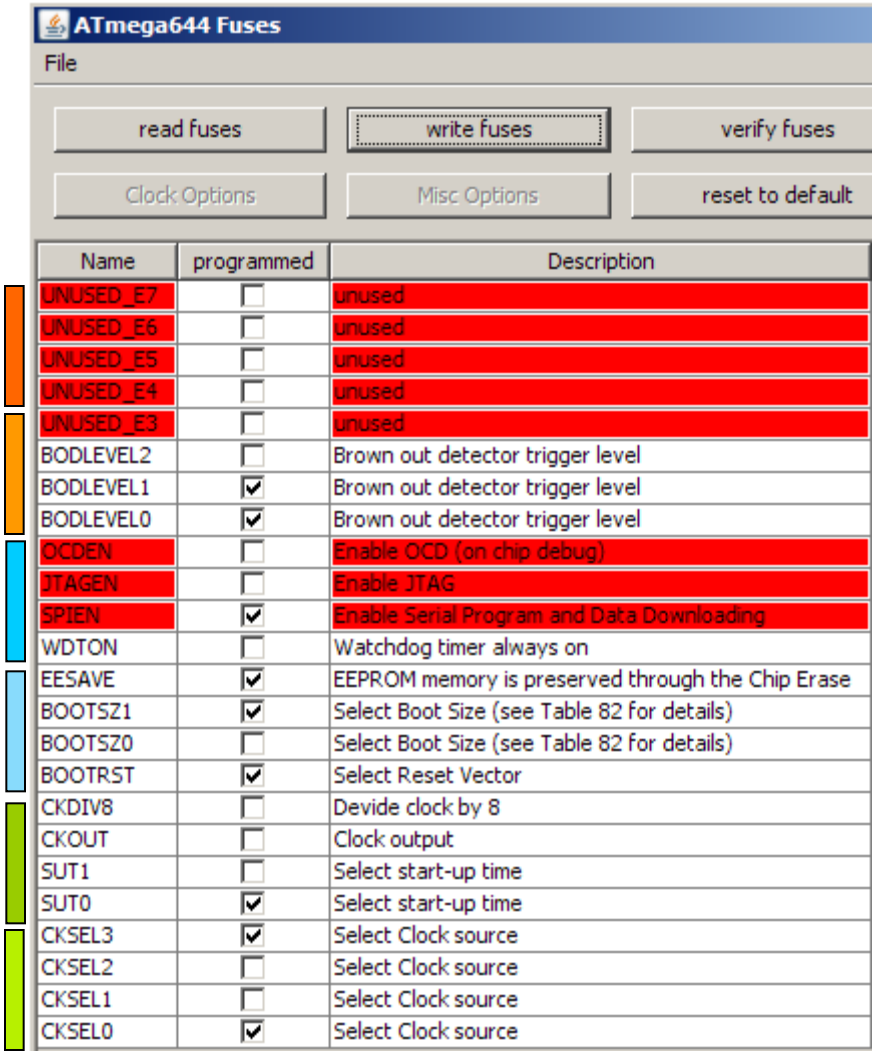
Burn-o-mat

- Einstellungen
  - AVR Type ATmega644 einstellen
  - Pfade für AVR Dude.exe und .config einstellen
  - Als Programmer usbasp… auswählen
  - AVR8_Burn_O_Mat.jar schließen und neu starten
- Datei Pfade für Hex File einstellen



Fuses einstellen

- 3 Bytes (Low, High und Ext Byte)
- Low=0xE6, High=0xD2, Ext=0xFC
- Invertierung ergibt die Haken
- Write und Write Fuses ausführen

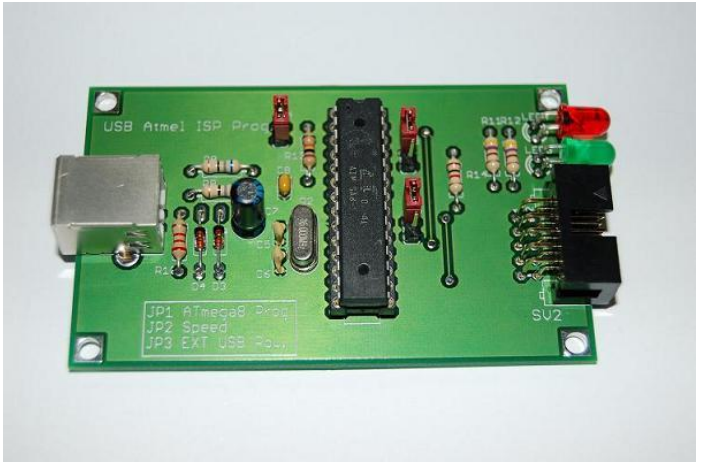


Name	programmed	Description
UNUSED_E7	☐	unused
UNUSED_E6	☐	unused
UNUSED_E5	☐	unused
UNUSED_E4	☐	unused
UNUSED_E3	☐	unused
BODLEVEL2	☐	Brown out detector trigger level
BODLEVEL1	☑	Brown out detector trigger level
BODLEVEL0	☑	Brown out detector trigger level
OCDEN	☐	Enable OCD (on chip debug)
JTAGEN	☐	Enable JTAG
SPIEN	☑	Enable Serial Program and Data Downloading
WDTON	☐	Watchdog timer always on
EESAVE	☑	EEPROM memory is preserved through the Chip Erase
BOOTSZ1	☑	Select Boot Size (see Table 82 for details)
BOOTSZ0	☐	Select Boot Size (see Table 82 for details)
BOOTRST	☑	Select Reset Vector
CKDIV8	☐	Devide clock by 8
CKOUT	☐	Clock output
SUT1	☐	Select start-up time
SUT0	☑	Select start-up time
CKSEL3	☑	Select Clock source
CKSEL2	☐	Select Clock source
CKSEL1	☐	Select Clock source
CKSEL0	☑	Select Clock source

USBASP

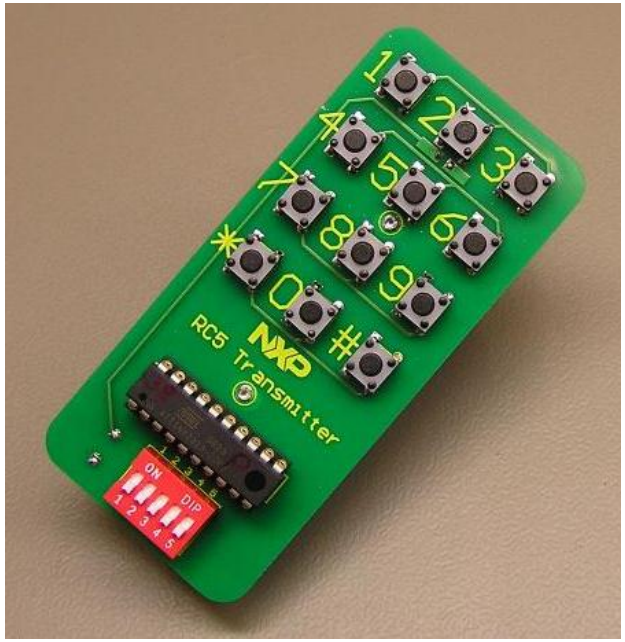
- USB Atmel SPI Programmer
- Ulrich Radig http://www.ulrichradig.de

- Neuen Prozessor programmieren:
  - JP2 gesteckt
  - Write fuses
  - JP2 öffnen
- Prozessor programmieren:
  - Read fuses
  - Write Flash
  - Write EEPROM
  - Write fuses



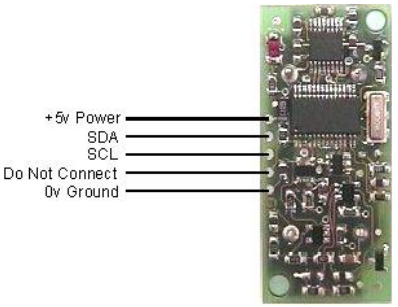
RC5 Fernbedienung

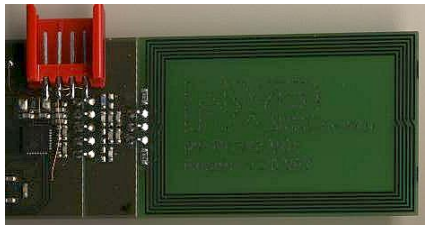
- Steuerung AVR-Mops
- Adresse Fernbedienung = Adresse AVR-Mops

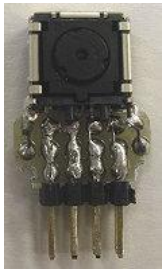


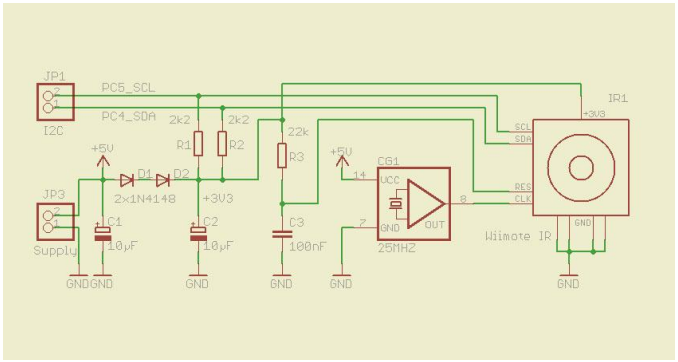
AVR-Mops Erweiterungen

- Ultraschall Sensor SRF08
- RFID Sensor RC232
- Compass Sensor CMPS03
- WII Sensor
- LM75 Sensor

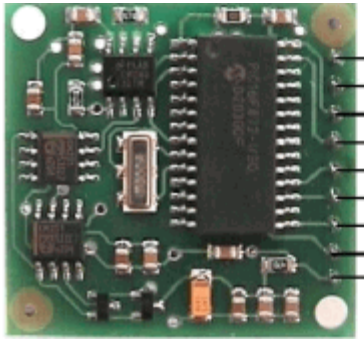






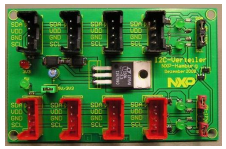


Norden

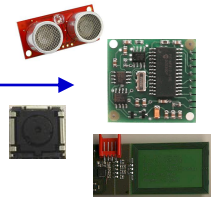


NXP Robo System

I²C Sniffer



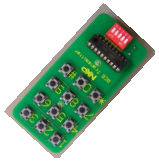
I²C Verteiler



I²C Sensoren

USB

I2C

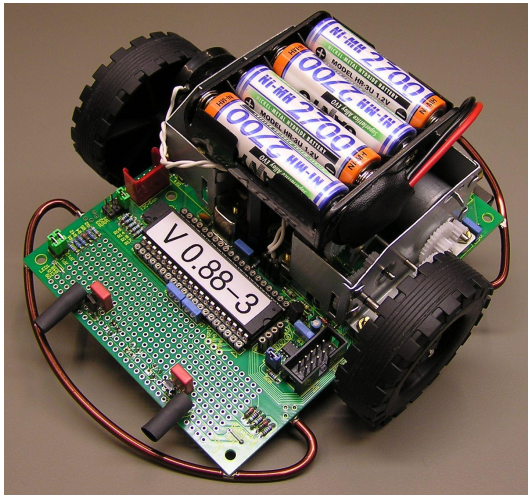


IR Fernbedienung

Fernseher

Computer

RS232



AVR-Mops

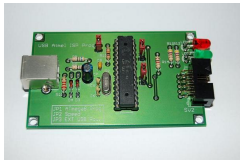
Scart

Stecker Netzteil

USB

PS2

SPI Programmer



SPI / ISP

Tastatur

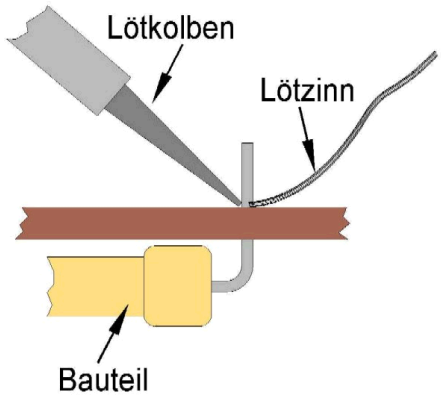
Fragen

Unterrichtsideen

- AVR-Mops löten
- AVR-Mops im Unterricht
- AVR-Mops Design Wettbewerb

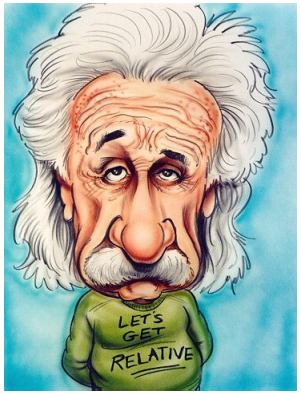
AVR-Mops löten

- Einführung in die Löttechnik
- Lötkurs „Basics“
- Lötkurs „AVR-Mops“



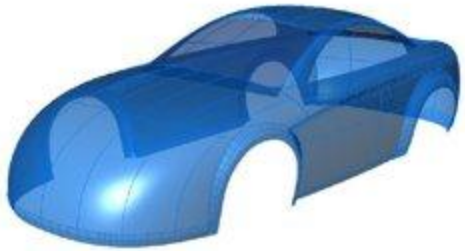
AVR-Mops im Unterricht

- Schwerpunkt Elektronik => „Elektronik zum Anfassen“
- Mathematik
  - Mathematische Grundlagen, Berechnungen
- Physik
  - Elektronik z.B. Regelung
  - Erdmagnetfeld (Compass Sensor)
  - Ultraschall
  - RFID
- Informatik
  - Übertragungsverfahren
  - Security
  - Programmierung
- Kunst
  - Design



AVR-Mops Design Wettbewerb

- Robo Karosserie
- Kunst und Design
- Pappmaschee
- Jury, Preise
- Befestigungslöcher vorhanden



Fragen

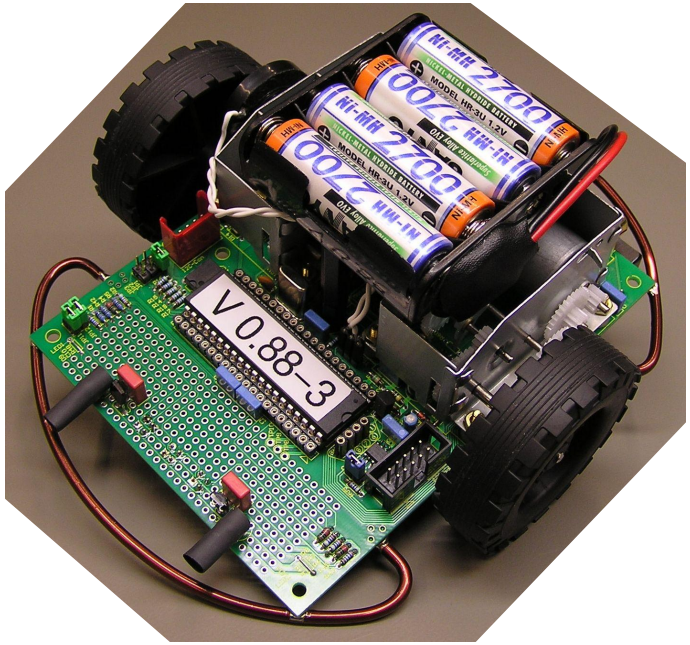
Componenten von NXP

Shared Technology Infrastructure STI

Temperatur
LM75

Real Time Clock
PCF2128

Kompass
KMZ51 /52



I²C Transmitter
PCA8574AN

RFID
RC523

Audio
TDA7050

Leistungstreiber
TDA7073A

RC5 Controller
P87LPC764

4bit LED Treiber
PCA9533

Vielen Dank

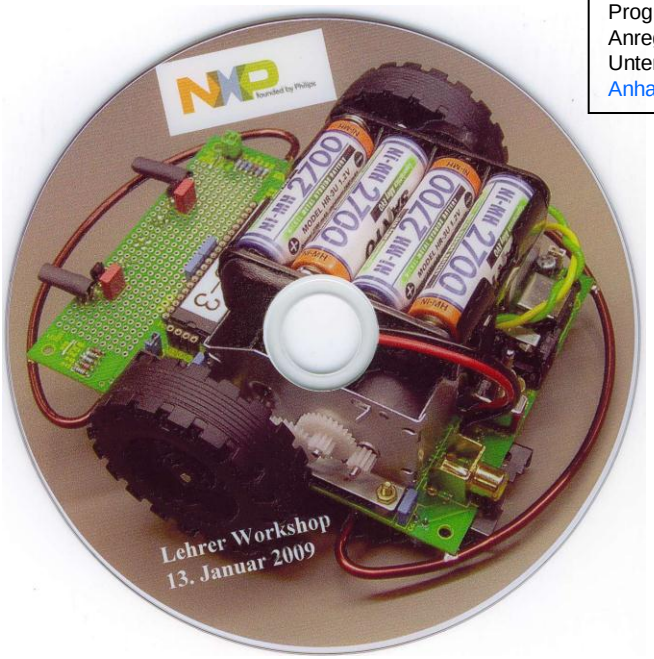
- Vielen Dank für ihr Kommen
- Technische Unterstützung
- Besuch in ihrer Schule
- Lötunterstützung
- Bauteile Beschaffung

- Kiste / Schule
- DVD

ANHANG

- Jumper
- WWW Links Software
- Anschlüsse
- Atmel Übersicht

- DVD Inhalt:
  - Datenblätter
  - Schaltpläne
  - Programme
  - Bilder
  - usw.



Jumper

- JP1 Auto Run aktiv (P1 wird automatisch gestartet)
- JP2 immer offen (keine Funktion)
- JP3 NTSC aktiv (USA Fernsehnorm)
- JP4..JP8 RC5 Adresse
- JP9 DFLASH AT45DB041D / 081D aktiv
- JP10 RC5 Led Anzeige aktiv
- JP11 IS471F aktiv
- JP12 CNY70 aktiv

- Aktiv = Jumper gesteckt!!!

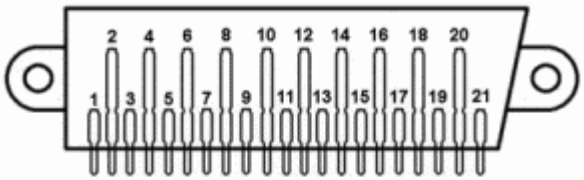
WWW Links Software

AVR	http://www.jcwolfram.de/projekte/avr/chipbasic2/main.php
USBASP	http://www.ulrichradig.de/home/index.php/avr/usb-avr-prog http://www.fischl.de/usbasp/
AVRDUDE	http://savannah.nongnu.org/projects/avrdude/
Burn-o-mat	http://avr8-burn-o-mat.aaabbb.de/
Java 1.5	http://java.sun.com/javase/downloads
Perl für Windows	http://strawberryperl.com/
ImageMagick	http://www.imagemagick.org/script/index.php
Programmer‘s Notepad	http://www.pnotepad.org/

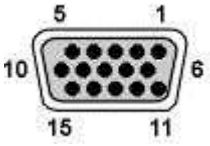
Anschlüsse

Shared Technology Infrastructure STI

Scart connections
(View from solder side of connector)



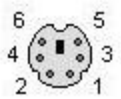
Pin 01: Audio output R
Pin 02: Audio input R
Pin 03: Audio output L
Pin 04: Audio ground
Pin 05: RGB Blue ground
Pin 06: Audio input L
Pin 07: RGB Blue input
Pin 08: AV mode switching
Pin 09: RGB Green ground
Pin 10: Data/RGB vertical sync
Pin 11: RGB Green input
Pin 12: Data/RGB horizontal sync
Pin 13: RGB Red ground
Pin 14: Data ground
Pin 15: S-video chrominance/RGB Red input
Pin 16: Fast blanking
Pin 17: Composite video output ground
Pin 18: Composite video/S-video luminance input ground
Pin 19: Composite video output
Pin 20: Composite video/S-video luminance input
Pin 21: Ground (shield)



VGA Stecker
SUB D15

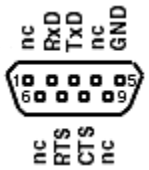
Pin1: Rot
Pin2: Grün
Pin3: Blau
Pin4: ID2 oder VESA-DCC Reserved
Pin5: NC oder VESA Masse
Pin6: Rot Masse
Pin7: Grün Masse
Pin8: Blau Masse
Pin9: Kodierung oder VESA-DCC Ausgang
Pin10: Horizontal - Vertikal Syncronisation Masse
Pin11: ID0
Pin12: ID1 oder VESA-DCC SDA (Serial Data Line)
Pin13: Horizontal Syncronisation
Pin14: Vertikal Syncronisation
Pin15: NC oder VESA-DCC SCL (Serial Data Clock Line)

PS2-Buchse von vorne

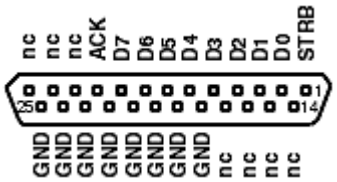


PIN	Beschreibung
1	DAT
2	NC
3	GND
4	+5V
5	CLK
6	NC

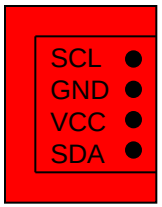
Seriell Port Stecker
SUB D9



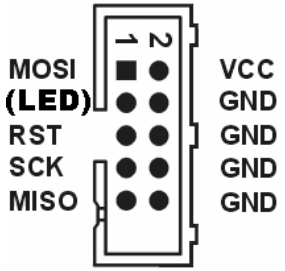
Parallel Port Buchse
SUB D25



I2C NXP Belegung
4-poliger Stecker
von oben



ISP Belegung
10-poliger Wannenstecker



Atmel Übersicht

Chip	Flash kb	Eeprom kb	Sram kb	I/O Pin	Uarts	Timer 16b	Timer 8b	10b-A/D	PWM	RTC
ATtiny2313	2	0,128	0,128	18	1	1	1	-	-	-
ATmega8	8	0,5	1	23	1	1	2	8	3	y
ATmega16	16	0,5	1	32	1	1	2	8	3	y
ATmega32	32	1	2	32	1	1	2	8	4	y
ATmega64	64	2	4	53	2	2	2	8	8	y
ATmega128	128	4	4	53	2	2	2	8	8	y

ATmega8 - PDIP Gehäuse