

Crazy Car - Arduino V1.0

Manual



DI (FH) Markus Krenn, MSc.

HW Vers.: 1.0
Doc. Vers.: 1.0
Stand: 03.11.2015

Änderungsverzeichnis

Vers. Nr.	Datum	Änderung	Ersteller
1.0	15.10.2015	Erstellen des Dokumentes	Markus Krenn
1.0	16.10.2015	Korrekturlesen 5. Inbetriebnahme	Karl Engelbogen, HTBL Kapfenberg
1.0	03.11.2015	Korrektur 5. Inbetriebnahme	Markus Krenn

Inhalt

1.	Allgemein.....	5
1.1.	Zweck.....	5
1.2.	Komponenten.....	5
2.	Das Fahrzeug	6
2.1.	Allgemein.....	6
2.2.	Servo.....	7
2.3.	Fahrtenregler.....	7
2.4.	Akku	8
3.	Crazy Car Controller.....	9
3.1.	Mikrokontroller	9
3.2.	Display	9
3.3.	Start/Stop Taste.....	10
3.4.	Lichtsteuerung.....	10
3.5.	Zugängliche Stecker.....	11
3.6.	Pin Liste	12
4.	Sensorik	13
4.1.	Abstandssensor	13
4.2.	Drehzahlsensor.....	14
4.3.	9-Achsen Bewegungssensor	15
4.4.	Messung der Batteriespannung	15
5.	Inbetriebnahme.....	16
5.1.	Spannungsversorgung	16
5.2.	Programmierung	16
5.3.	Funktionsbibliotheken.....	17

Abbildungsverzeichnis

Abbildung 1: Crazy Car Komponenten.	5
Abbildung 2: Standard 3pol. Servo/BEC Stecker.	6
Abbildung 3: Standard Modellbau Servo.	7
Abbildung 4: Fahrtenregler.	7
Abbildung 5: Akku im Fahrzeug und ausgebaut.	8
Abbildung 6: Crazy Car Controller Arduino.	9
Abbildung 7: 84x48 Pixel Grafikdisplay.	9
Abbildung 8: Start und Stop Taste.	10
Abbildung 9: Anschlusspins für Lichtsteuerung.	10
Abbildung 10: SPI Schnittstelle.	11
Abbildung 11: I ² C Schnittstelle.	11
Abbildung 12: ICSP Schnittstelle.	11
Abbildung 13: SHARP Abstandssensor montiert.	13
Abbildung 14: SHARP Abstandssensor Pinbelegung.	13
Abbildung 15: Hall Sensor auf Antriebsachse.	14
Abbildung 16: Hall Sensor Stecker + LED.	14
Abbildung 17: 9-Achsen Bewegungssensor.	15
Abbildung 18: 9-Achsen Bewegungssensor – Achsenzuordnung.	15
Abbildung 19: Beispiel COM14.	16
Abbildung 20: Beispiel Arduino IDE COM-Port Konfiguration.	16

1. Allgemein

1.1. Zweck

Das Crazy Car ist ein autonom fahrendes Modellfahrzeug im Maßstab 1:18. Die Steuerung des Fahrzeugs übernimmt ein ATMEL Mikrokontroller, auf welchem der Bootloader für das Arduino Framework programmiert ist. Das Fahrzeug ist mit entsprechender Sensorik und Aktorik ausgestattet um autonom einen speziell präparierten Kurs abzufahren.

1.2. Komponenten

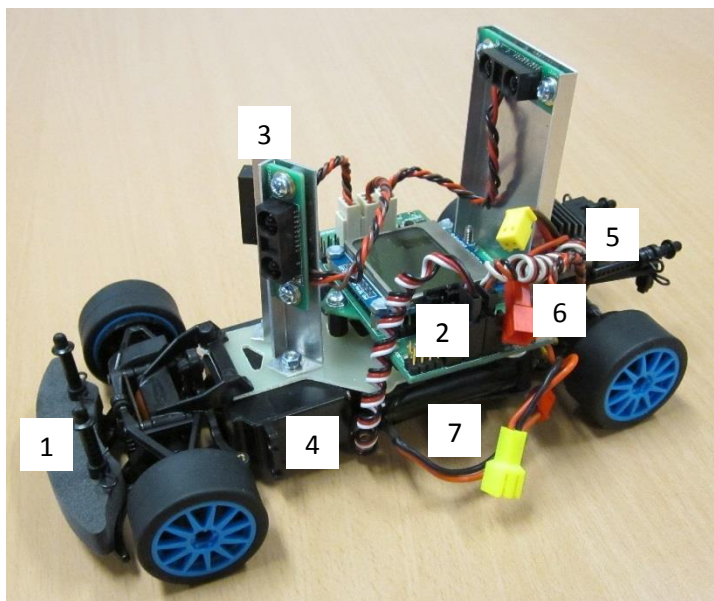


Abbildung 1: Crazy Car Komponenten.

1. HPI RS4 Micro
2. Crazy Car Controller Arduino V1.0
3. SHARP Abstandssensoren GP2Y0A60SZLF
4. Lenkservo
5. Electronic Speed Control – Fahrtenregler
6. Motor
7. Akkumulator

2. Das Fahrzeug

2.1. Allgemein

Das Chassis ist ein herkömmliches Modellfahrzeug im Maßstab 1:18 produziert von der Firma HPI und trägt die Bezeichnung RS4 Micro. Das Chassis ist mit unterschiedlichen Lexan Karosserien erhältlich und trägt alle mechanischen und elektronischen Teile des Crazy Cars.

Die Standardkomponenten die jedes Modellauto beinhaltet sind die ESC und das Lenkservo. Beide werden mit einem 3pol. Kabel am Crazy Car Controller angeschlossen. Diese 3 Kabel beinhaltet die Spannungsversorgung, welche vom Fahrtenregler entnommen wird, BEC, sowie das Signal mit welchem man die Komponenten ansteuert.



Abbildung 2: Standard 3pol. Servo/BEC Stecker.

Das Servo Signal ist ein Pulsweiten Moduliertes Signal mit einer Frequenz zwischen 50 & 60Hz. Die Pulsbreite muss in den Bereichen 1ms – 2ms variieren:

Faustformel

1.0ms	100% Ausschlag links/rechts bzw. 100% vorwärts/rückwärts
1.5ms	Mittelstellung
2.0ms	100% Ausschlag links/rechts bzw. 100% vorwärts/rückwärts

Da dieses Signal nicht Standardisiert ist, können die realen Werte abweichen und müssen für jedes Fahrzeug angepasst werden. Hier muss drauf geachtet werden, dass z.B. das Lenk-Servo nicht weiter dreht als es die Lenkung mechanisch zulässt.

2.2. Servo

Das Servo ist zuständig für die Betätigung der Lenkung. Der Ausschlagwinkel des Servo-Hebels folgt der jeweiligen Pulsbreite des Servo-Signals.

VORSICHT: Das Servo darf keinesfalls weiter drehen als es die Lenkung mechanisch zulässt, ansonsten kommt es bei dauerhaften Betrieb zu Getriebebeschäden im Servo.

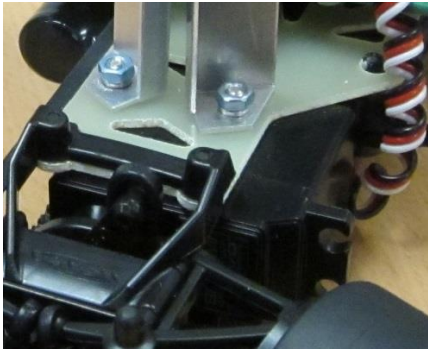


Abbildung 3: Standard Modellbau Servo.

2.3. Fahrtenregler

Der Fahrtenregler steuert den Antriebsmotor des Fahrzeuges an. Über das angelegte PWM Signal kann das Fahrzeug vorwärts bzw. rückwärts fahren. Zusätzlich zu diesen beiden Funktionen gibt es noch eine „Bremse“, bei welcher der Motor „blockiert“. Damit der Fahrtenregler funktioniert, muss nach dem Einschalten das Neutral Signal, 1.5ms, min. 1.5 Sekunden anliegen. Danach ertönt ein Piep-Ton und der Regler ist einsatzbereit.



Abbildung 4: Fahrtenregler.

Signalfolgen

Signalwechsel Mittelstellung auf Vorwärts => Fahrzeug fährt Vorwärts

Signalwechsel Vorwärts auf Rückwärts => Fahrzeug bremst

Signalwechsel Vorwärts auf Rückwärts auf Mittstellung auf Rückwärts => Fahrzeug fährt Rückwärts

Signalwechsel Mittelstellung auf Rückwärts => Fahrzeug fährt Rückwärts

2.4. Akku

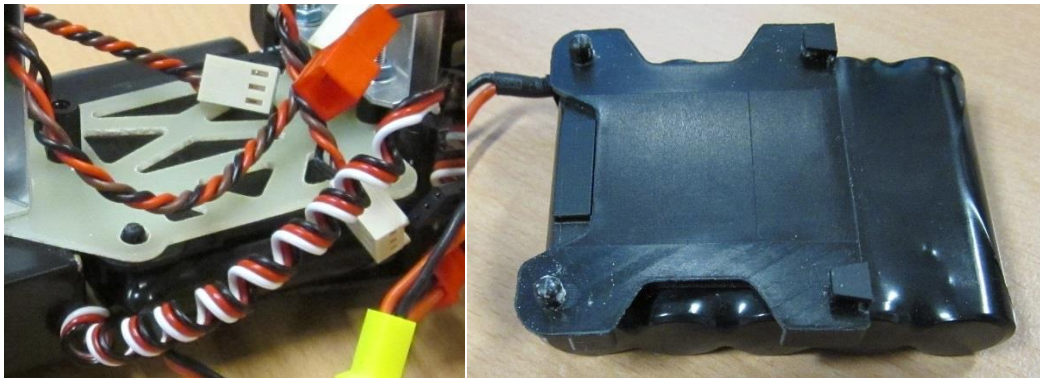


Abbildung 5: Akku im Fahrzeug und ausgebaut.

Das Fahrzeug beinhaltet im Chassis Inneren einen 6V NiMh Akku mit 1200mAh.

3. Crazy Car Controller



Abbildung 6: Crazy Car Controller Arduino.

3.1. Mikrokontroller

Der Mikrokontroller von der Firma ATMEL ist ein ATMEGA328P-AU mit 8-Bit Prozessorkern, 2kB RAM und 32kB Flash Programmspeicher. Dieser ist mit dem Arduino Framework programmiert um die volle Funktionalität des Arduino Frameworks benutzen zu können. Die Schaltung basiert auf dem Arduino Nano Modul.

3.2. Display

Das Display ist ein 84x48 Pixel Grafikdisplay mit dem Displaycontroller PCD8544 und blauer Hintergrundbeleuchtung. Dieses ist am SPI Interface des Mikrokontrollers angeschlossen und kann Mithilfe der u8gLib Library (<https://code.google.com/p/u8glib/>) im Arduino Framework benutzt und angesteuert werden.



Abbildung 7: 84x48 Pixel Grafikdisplay.

3.3. Start/Stop Taste

Das Crazy Car muss laut Reglement mit einer Taste gestartet bzw. mit einer anderen Taste gestoppt werden können. Dazu sind 2 Taster mit Schließer Funktion auf der Karosserie montiert welche direkt an 2 digitalen Pins des Mikrokontrollers angeschlossen sind.

Start-Taste = schwarz

Stop-Taste = rot

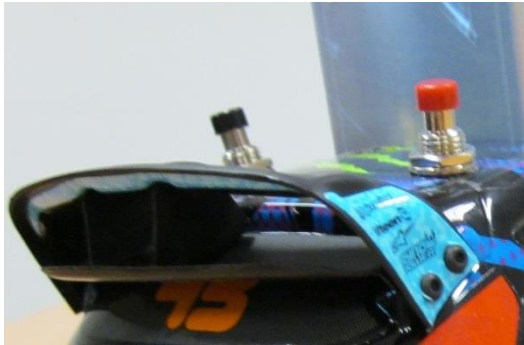


Abbildung 8: Start und Stop Taste.

3.4. Lichtsteuerung

Um etwaige Beleuchtungseffekte oder sonstige Steuersignal am Crazy Car umzusetzen, wurde der I²C Bus um einen I/O Expander PCF8574 erweitert (Tipp über die Arduino lib <http://playground.arduino.cc/Main/PCF8574Class> kann man den Portexpander ansprechen). Dieser kann als Ein-/Ausgang verwendet werden und benutzt den Logikpegel von 3.3V bzw. 0V.

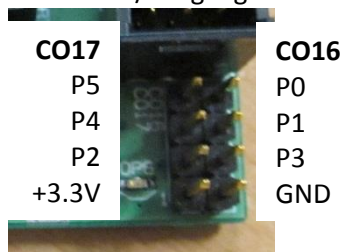


Abbildung 9: Anschlusspins für Lichtsteuerung.

P6 ist nicht belegt und auf P7 ist eine LED angeschlossen die bereits auf der Platine aufgelötet ist.

3.5. Zugängliche Stecker

Um den Crazy Car Controller erweiterbar zu gestalten bzw. um die Signal zu messen wurden der I²C Bus und die SPI Schnittstelle auf Pin-Header ausgeführt. Die ICSP Schnittstelle wird benutzt, um den Mikrokontroller mit dem Arduino Bootloader zu programmieren.

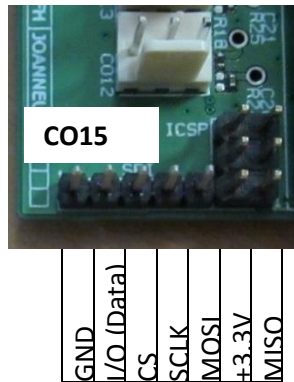


Abbildung 10: SPI Schnittstelle.



Abbildung 11: I²C Schnittstelle.



Abbildung 12: ICSP Schnittstelle.

3.6. Pin Liste

ATMEGA	Arduino	Crazy Car Funktion
PB0	D8	LCD Reset
PB1	D9	LCD Data/Cmd
PB2	D10	SPI Chip Select
PB3	D11	SPI MOSI
PB4	D12	SPI MISO
PB5	D13	SPI SCLK
PC0	A0	Analog Eingang Sensor 1
PC1	A1	Analog Eingang Sensor 2
PC2	A2	Analog Eingang Sensor 3
PC3	A3	Analog Eingang Batterie Spannungsmessung
PC4	A4	I ² C SDA
PC5	A5	I ² C SCL
PD0	D0	UART RXD
PD1	D1	UART TXD
PD2	D2	Signal Drehzahlmessung
PD3	D3	Stop Taste
PD4	D4	Start Taste
PD5	D5	Speed Control (PWM Signal)
PD6	D6	Lenkung (PWM Signal)
PD7	D7	LCD Hintergrundbeleuchtung
PC6	RESET	Reset Signal
PB6 & PB7		16 MHz Quarz
AREF	AREF	Referenzspannung 2.048V

4. Sensorik

4.1. Abstandssensor

Auf dem Fahrzeug sind insgesamt 3 Abstandssensoren der Firma SHARP montiert, welche die Abstände nach vorne, nach links und nach rechts messen. Damit kann sich das Fahrzeug innerhalb der Streckenbegrenzung orientieren.

Der Sensor hat einen Messbereich von 10cm – 150cm (lt. Datenblatt) und gibt je nach Entfernung einen Spannungswert aus, welcher mit dem Mikrokontroller über den Analog Digital Wandler eingelesen werden kann (*Tipp: Der Zusammenhang zwischen Entfernung und Spannung ist nicht linear*). Zusätzlich ist eine genaue Referenzspannungsquelle am Mikrokontroller angeschlossen und der Spannungswert des Sensors über ein Tiefpassfilter gefiltert. Die detaillierte Information über den Sensor entnehmen sie bitte dem Datenblatt.

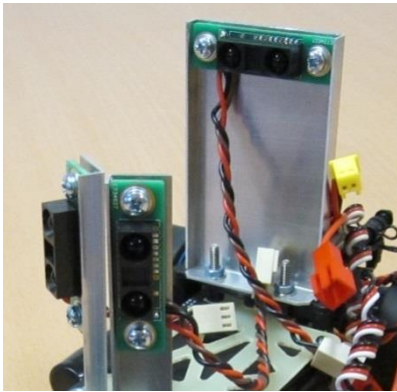


Abbildung 13: SHARP Abstandssensor montiert.



- 1... +3.3V (links)
- 2... Enable (pulled-up high 10k)
- 3... Analog Output
- 4... GND

Abbildung 14: SHARP Abstandssensor Pinbelegung.

4.2. Drehzahlsensor

Um die Achsdrehzahl des Fahrzeugs messen zu können, sind auf der Achse 2 Neodym Magnete angebracht die über einen Hallsensor ausgewertet und als Rechtecksignal an den Mikrokontroller weitergegeben werden. Dies bedeutet, dass pro Umdrehung der Achse 2 Impulse ausgegeben werden. Ob der Sensor die Magnete richtig erkennt, wird optisch über eine LED angezeigt.

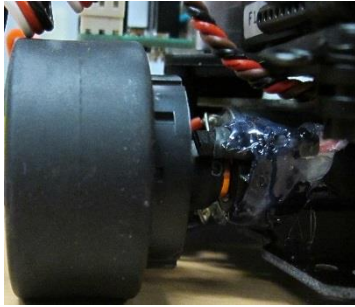


Abbildung 15: Hall Sensor auf Antriebsachse.

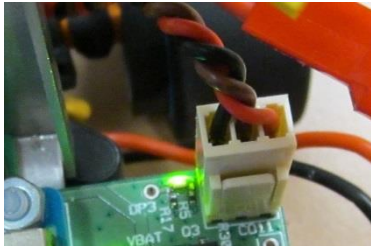


Abbildung 16: Hall Sensor Stecker + LED.

4.3. 9-Achsen Bewegungssensor

Für ausgefeiltere Navigationssaufgaben kann der 9-Achsen Bewegungssensor benutzt werden. Dieser beinhaltet ein 3-Achsen Gyroskop, einen 3-Achsen Beschleunigungssensor sowie einen 3 Achsen Kompass. Der Sensor ist am I²C Bus angeschlossen und kann Mithilfe Arduino Lib <https://github.com/Snowda/MPU9250> ausgelesen und benutzt werden.

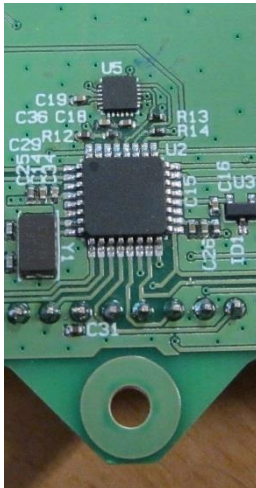


Abbildung 17: 9-Achsen Bewegungssensor.

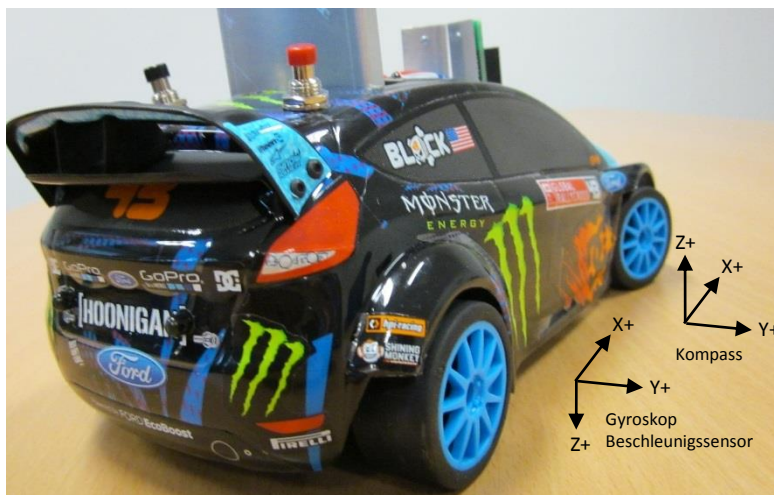


Abbildung 18: 9-Achsen Bewegungssensor – Achsenzuordnung.

4.4. Messung der Batteriespannung

Um den Ladezustand der Batterie überwachen zu können, ist Mithilfe eines Spannungsteilers eine Spannungsmessung realisiert. Der Spannungsteiler teilt die Spannung so weit herunter, dass der Mikrokontroller diese verarbeiten kann. Die Spannung des Akkus kann mit dem Analog Digital Wandler eingelesen und überwacht werden. Dies spielt vor allem eine Rolle wenn ein Lithium-Polymer statt dem NiMh Akku benutzt wird.

5. Inbetriebnahme

5.1. Spannungsversorgung

Das Fahrzeug und die Kontrollereinheit kann von dem 6V NiMh Akku oder von der USB Schnittstelle betrieben werden. Wird das Fahrzeug vom 6V Akku versorgt, muss der Fahrtenregler eingeschalten und an der Kontrollereinheit angesteckt sein. Wird das USB Kabel angeschlossen, ist die Kontrollereinheit automatisch aktiv jedoch kann die Aktorik, Lenkung und Motor, von der USB Schnittstelle allein nicht versorgt werden.

5.2. Programmierung

Die Programmierung des Mikrocontrollers erfolgt über die micro USB Schnittstelle. Nach dem Anschließen des Mikrocontrollers über das USB Kabel am PC sollte im Gerätemanager im Bereich Anschlüsse (COM & LPT) ein neuer COM-Port zu finden sein



Abbildung 19: Beispiel COM14.

Dieser muss dann entsprechend in der Arduino Entwicklungsumgebung unter *Tools / Port* eingestellt werden.

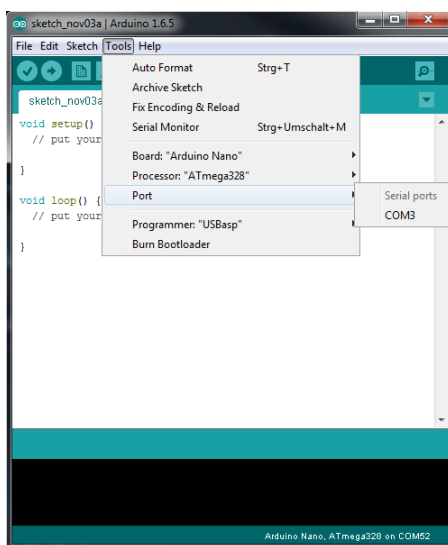


Abbildung 20: Beispiel Arduino IDE COM-Port Konfiguration.

Eine Demosoftware steht auf der CrazyCar Homepage <http://www.fh-joanneum.at/crazycar> unter *Support / Downloads* zur Verfügung. Die Entwicklungsumgebung, den Befehlssyntax sowie die Anleitungen für die Arduino IDE finden sie auf der Arduino Homepage <https://www.arduino.cc/>.

5.3. Funktionsbibliotheken

Um die zusätzlichen Funktionen auf der Hardware nutzen zu können, müssen die entsprechenden Bibliotheken: U8glib, I2Cdev und pcf8574 in Installationsverzeichnis unter libraries kopiert werden.
z.B. D:\Arduino1.6.5\libraries