FH | JOANNEUM
Elektronik und
Computer Engineering

Crazy Car Controller
Arduino V2.0
Manual

DI (FH) Markus Krenn, MSc

Hardware Version: 2.0
Dokument Version: 1.0
Stand: 09.11.2016

Änderungsverzeichnis

Vers. Nr.	Datum	Änderung	Ersteller
1.0	28.10.2016	Erstellen des Dokumentes	Markus Krenn
1.0	09.11.2016	Überprüft von Karl Engelbogen Dokument freigegeben	Markus Krenn

Inhalt

Abbildungsverzeichnis

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

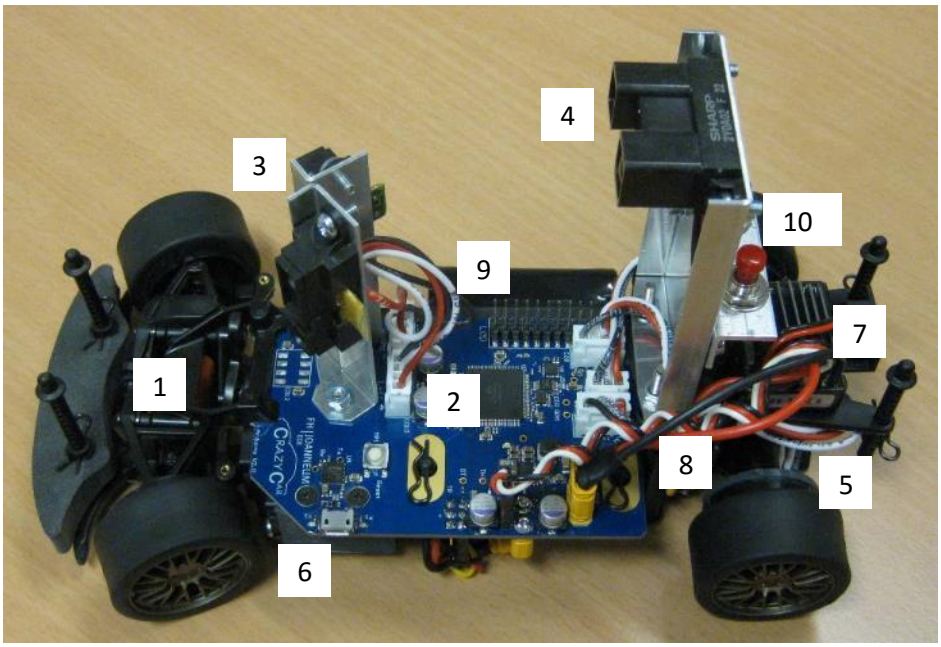

Abbildung 1: Crazy Car Komponenten.

1. HPI RS4 Micro
2. Crazy Car Controller Arduino V2.0
3. SHARP Abstandssensor GP2Y0A02YK0F
4. SHARP Abstandssensoren GP2Y0A21YK0F
5. Drehzahlsensor
6. Lenk-Servo (Unterseite)
7. Electronic Speed Control – Fahrtenregler
8. Motor
9. Akkumulator
10. Start- und Stop Taste

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 Lenk-Servo. 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.

weiß…	Servo Signal (kann auch gelb/blau sein)
rot…	BEC Spannung vom ESC
schwarz…	Masse (Akku Minus)

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

Das Servo Signal ist ein Pulsweiten Moduliertes Signal mit einer Frequenz von etwa 50Hz.
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 Lenk-Servo bzw. Fahrzeug angepasst werden. Hier muss drauf geachtet werden, dass z.B. das Lenk-Servo nicht weiter dreht als es die Lenkung bzw. das Chassis 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 bzw. das Chassis mechanisch zulässt, ansonsten kommt es bei dauerhaften Betrieb zu Getriebeschäden im Servo oder zu Schäden am Chassis

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. rückwärtsfahren. Zusätzlich zu diesen beiden Funktionen gibt es noch eine „Bremse“, bei welcher der Motor „blockiert“. Mit der Crazy Car Controller Arduino 2.0 Version wird eine modifizierte Variante des ESCs ausgeliefert. Die Ansteuerung und Signalfolge für diesen ESC, kann aus dem ESC Manual entnommen werden. In der Auslieferungskonfiguration hat der ESC 50% seiner maximalen Vorwärts/Rückwärtsdrehzahl und 100% seiner Bremswirkung. Diese Konfiguration kann im UART Mode geändert werden, siehe ESC Manual.

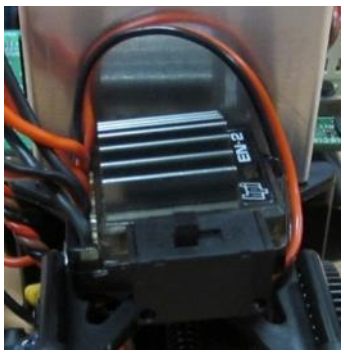

Abbildung 4: Fahrtenregler.

2.4. Akku

Das Fahrzeug beinhaltet im Chassis Inneren einen 6V NiMh Akku mit 1200mAh. Dieser wird an der Unterseite des CC Controllers befestigt und mittles XT30 Steckers verbunden.

Abbildung 5: Akku im Fahrzeug und ausgebaut.

3. Crazy Car Controller

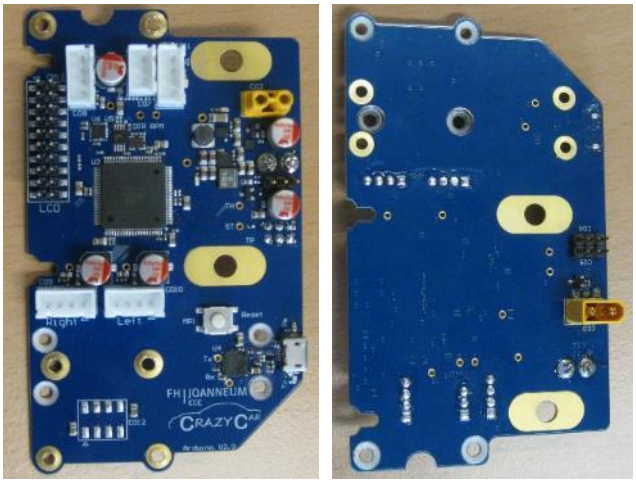

Abbildung 6: Crazy Car Controller Arduino.

3.1. Mikrokontroller

Der Mikrokontroller von der Firma ATMEL ist ein ATMEGA2560-16AU mit 8-Bit Prozessorkern, 8kB RAM und 256kB 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 MEGA 2560 Modul.

3.2. Display

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

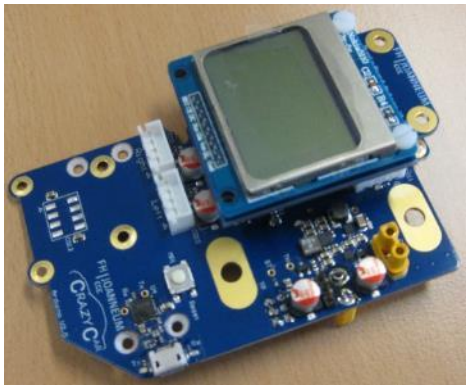

Abbildung 7: Crazy Car Controller mit aufgestecktem 84x48 Pixel Grafikdisplay Board.

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 dem Chassis 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. Line Follower / Universalstecker

Am vorderen Ende des Crazy Car Controllers kann ein zusätzlicher Stecker aufgelötet werden. Dieser dient in erster Linie um eine Line Follower Funktionalität umzusetzen (nicht im Set enthalten). Diese Pins sind direkt mit dem Mikrokontroller verbunden und können für allgemeine Steuerungsaufgaben herangezogen werden. Die dafür vorgesehene Stiftliste kann nachbestückt werden.

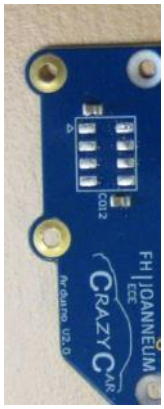

Abbildung 9: CO12 Anschlusspins für Line Follower.

3.5. Display/Universal Stecker

Um den Crazy Car Controller erweiterbar zu gestalten wurde eine allgemein zugängliche Schnittstelle ausgeführt. Diese dient in erster Linie um den Mikrokontroller mit dem Arduino Bootloader über das ICSP Interface zu programmieren und das Displayboard aufzustecken. Es können mit dieser Schnittstelle eigene Erweiterungen der Basis Hardware durchgeführt werden. Vor der Entwicklung eines eigenen Aufsteckboards wird empfohlen sich über den Stecker und die Spannungsversorgungen genauestens beim Crazy Car Controller Entwicklungsteam zu erkundigen.

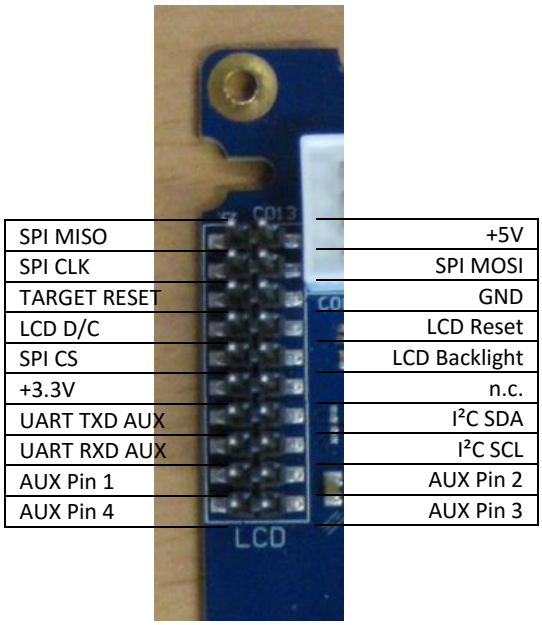

Abbildung 10: Display/Universal Schnittstelle.

3.6. Pin Liste

ATMEGA2560	Arduino	Crazy Car Funktion
Display/Universal Schnittstelle		
PB2	D51	LCD/SPI MOSI (ICSP MOSI)
PB3	D50	SPI MISO (ICSP MISO)
PB1	D52	LCD/SPI Clock (ICSP Clock)
PB0	D53	LCD/SPI Chip Select
PC7	D30	LCD Backlight
PC6	D31	LCD Reset
PC5	D32	LCD Data/Cmd
PD0	D21	I^2C SCL – Bewegungssensor SCL
PD1	D20	I^2C SDA – Bewegungssensor SDA
PJ0	D15	UART TxD – UART 3
PJ1	D14	UART RxD – UART 3
PE6	??	Auxilary Pin 1
PE5	D3	Auxilary Pin 4
PH4	D7	Auxilary Pin 2
PH3	D6	Auxilary Pin 3
Start/Stop Taste		
PB6	D12	Start Taste
PB7	D13	Stop Taste
Aktoren		
PE3	D5	Lenkung – PWM Signal
PE4	D2	Speed Control – PWM Signal
Sensoren		
PF0	A0	Analog – Batterie Spannungsmessung
PF1	A1	Analog – Sensor Links
PK0	A8	Enable Analog Sensor Links digital (nicht benutzt)
PF2	A2	Analog – Sensor Rechts
PK1	A9	Enable Analog Sensor Rechts digital (nicht benutzt)
PF3	A3	Analog – Sensor Vorne
PD7	D38	Enable Analog Sensor Vorne digital (nicht benutzt)
PL1	D48	RPM Sensor Richtung
PL0	D49	RPM Sensor Drehzahl
PD3	D18	RPM Sensor Drehzahl
PD0	D21	I^2C SCL – Bewegungssensor SCL
PD1	D20	I^2C SDA – Bewegungssensor SDA
PD2	D19	Bewegungssensor Interrupt
PK3	A11	Line Follower 1
PK4	A12	Line Follower 2
PK5	A13	Line Follower 3
PK6	A14	Line Follower 4
PK7	A15	Line Follower 5
Allgemein		
PE0	D0	Debug UART RxD
PE1	D1	Debug UART TxD

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 – 80cm bzw. 15 – 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 nichtlinear*. Die Spannungsversorgung der Sensoren ist meinem 330µF Kondensator gestützt um die Stromspitze der Sensoren beim Aussenden des Infrarotstrahls auszugleichen. Das analoge Signal ist zusätzlich über ein Tiefpassfilter gefiltert. Die detaillierte Information über den Sensor entnehmen sie bitte dem Datenblatt.

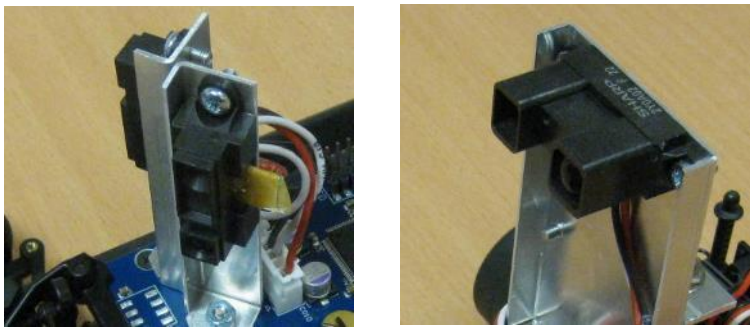

Abbildung 11: SHARP Abstandssensor montiert.

4… GND
3… Analog Ausgang Sensor
2… Enable
1… +5.0V (rechts)

Abbildung 12: SHARP Abstandssensor Pinbelegung.

4.2. Drehzahlsensor

Um die Achsdrehzahl des Fahrzeugs messen zu können, ist auf der Hinterachse eine Scheibe mit 22 Neodym Magneten mit abwechselnder Polarität angebracht. Die Drehbewegung wird über einen Hallsensor ausgewertet und als Rechtecksignal an den Mikrokontroller weitergegeben. Zusätzlich kann über einen weiteren Pin die Drehrichtung der Achse bestimmt werden. Ob der Sensor die Magneten richtig detektiert, wird optisch über eine grüne LED angezeigt.

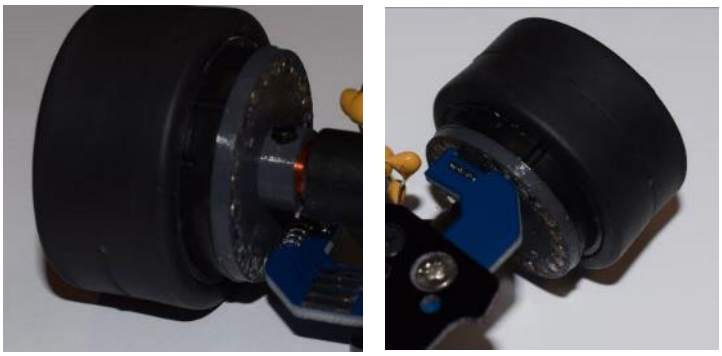

Abbildung 13: Hall Sensor auf hinterer Antriebsachse.

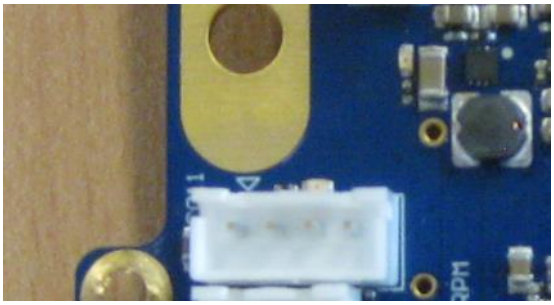

Abbildung 14: Hall Sensor Stecker + LED.

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 der Arduino Lib https://github.com/Snowda/MPU9250 ausgelesen und benutzt werden.

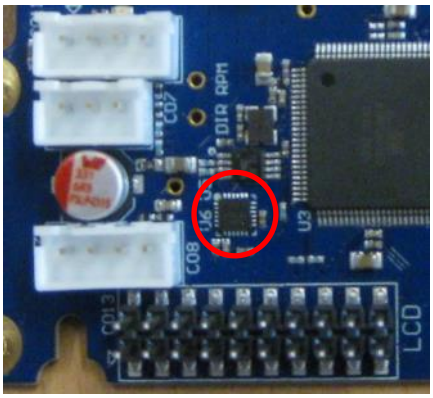

Abbildung 15: 9-Achsen Bewegungssensor.

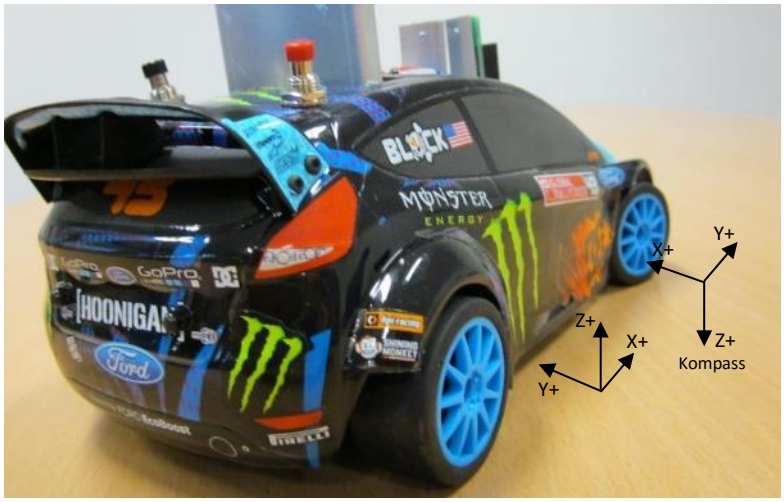

Abbildung 16: 9-Achsen Bewegungssensor – Achsenzuordnung.

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 Akku 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 nicht versorgt werden. Der gesamte Kontrollerkern und die Sensorik wird mit +5V versorgt. Lediglich der 9-Achsen Bewegungssensor benötigt eine 3.3V Versorgungspannung.

5.2. Programmierung

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

Abbildung 17: Beispiel COM14.

Der FDTI Treiber für den „USB Serial Port“ kann hier heruntergeladen werden:
http://www.ftdichip.com/FTDrivers.htm

Dieser muss dann entsprechend in der Arduino Entwicklungsumgebung unter *Tools / Port* eingestellt werden. Um den Kontroller programmieren zu können, muss auch das entsprechende Board und der Prozessor ausgewählt werden

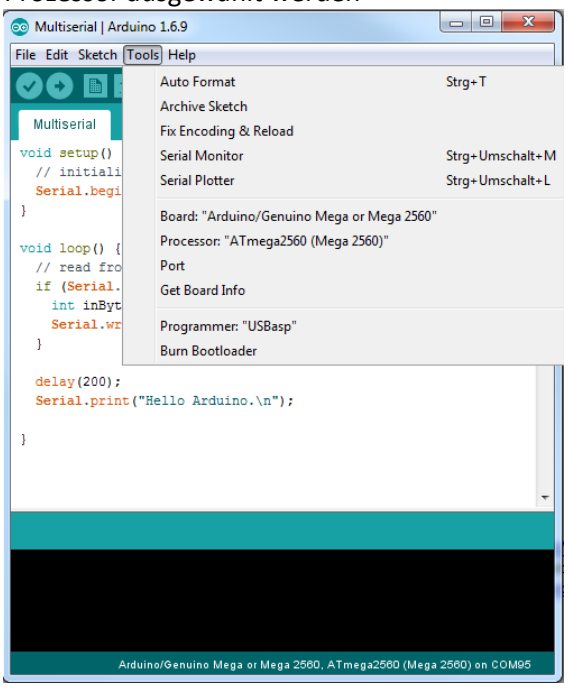

Abbildung 18: Beispiel Arduino IDE COM-Port und Prozessor Konfiguration.

Eine Demosoftware steht auf der Crazy Car Homepage https://fh-joanneum.at/projekt/crazycar/ zur Verfügung. Die Entwicklungsumgebung, den Befehlssyntax sowie die Anleitungen für die Arduino Entwicklungsumgebung 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 MPU9250 in das Installationsverzeichnis unter libraries kopiert werden. z.B. D:\Arduino1.6.9\libraries