RP6 ROBOT SYSTEM

RP6v2 CONTROL M256 WiFi

Erweiterungsmodul

AREXX Engineering

RP6v2-M256-WIFI

www.arexx.com

RP6v2 CONTROL M256 WiFi

Bedienungsanleitung

- Deutsch (German) -

Version RP6M256-DE-20120711

WICHTIGE INFORMATION!
Bitte unbedingt lesen!

Bevor Sie dieses RP6 Erweiterungsmodul in Betrieb nehmen, lesen Sie bitte diese Anleitung UND die RP6 ROBOT SYSTEM Bedienungsanleitung vollständig durch! Sie erläutert Ihnen die korrekte Verwendung und weist auf mögliche Gefahren hin! Weiterhin enthält sie wichtige Informationen, die für viele Anwender keineswegs offensichtlich sein dürften. Die RP6v2-M256-WIFI Anleitung ist nur eine Ergänzung!

Bei Nichtbeachtung dieser Anleitung und der RP6 ROBOT SYSTEM Anleitung erlischt jeglicher Garantieanspruch! Weiterhin übernimmt AREXX Engineering keinerlei Haftung für Schäden jeglicher Art, die aus Nichtbeachtung dieser Anleitung resultieren!

Bitte beachten Sie vor allem den Abschnitt "Sicherheitshinweise" in der RP6 ROBOT SYSTEM Bedienungsanleitung!

Impressum

©2012 AREXX Engineering

Nervistraat 16
8013 RS Zwolle
The Netherlands

Tel.: +31 (0) 38 454 2028
Fax.: +31 (0) 38 452 4482

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/

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!

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.
AREXX Engineering garantiert die Funktionalität der mitgelieferten Applikationsbeispiele unter Einhaltung der in den technischen Daten spezifizierten Bedingungen. Sollte sich der Roboter oder die PC-Software darüber hinaus als fehlerhaft oder unzureichend erweisen, so übernimmt der Kunde alle entstehenden Kosten für Service, Reparatur oder Korrektur.
Bitte beachten Sie auch die entsprechenden Lizenzvereinbarungen auf der CD-ROM!

Symbole

Im Handbuch werden folgende Symbole verwendet:

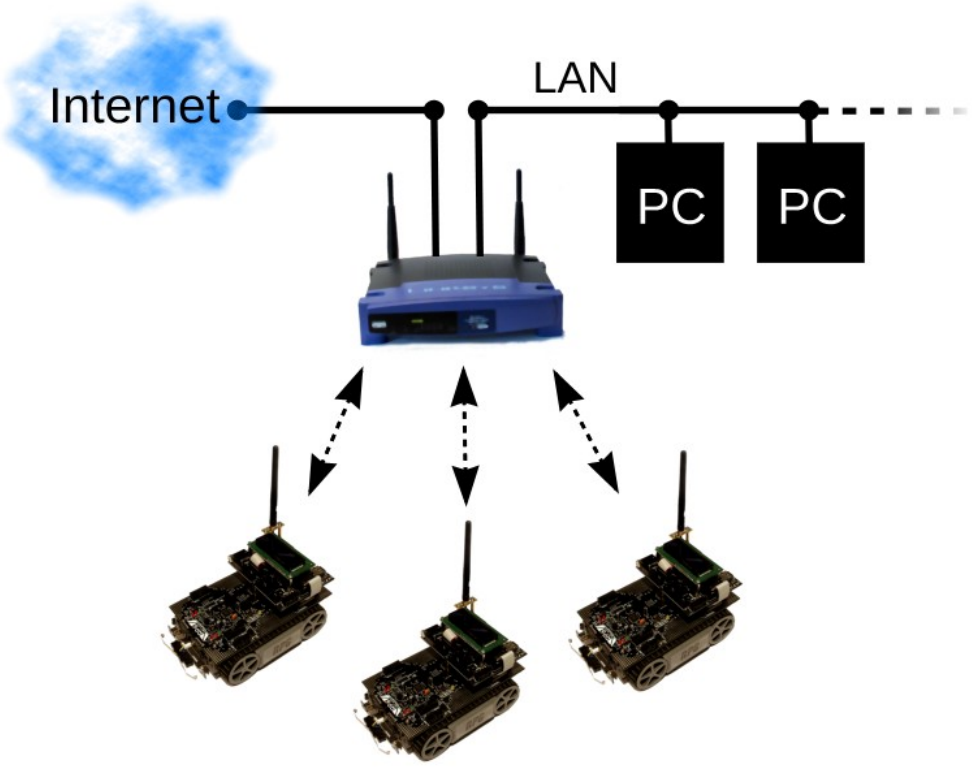
Das "Achtung!" Symbol weist auf besonders wichtige Abschnitte hin, die sorgfältig beachtet werden müssen. Wenn Sie hier Fehler machen, könnte dies ggf. zur Zerstörung des Roboters oder seinem Zubehör führen und sogar Ihre eigene oder die Gesundheit anderer gefährden!

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.

Inhaltsverzeichnis

1. Das RP6v2 M256 WIFI Erweiterungsmodul

Mit dem RP6v2 Control M256 WIFI Erweiterungsmodul (Teilenummer RP6v2-M256-WIFI oder auch kurz RP6-M256) ist es möglich einen oder mehrere RP6 und RP6v2 Roboter in ein drahtloses Computernetzwerk einzubinden, sie darüber fernzusteuern und Telemetriedaten zu übertragen. Der Roboter kann auch selbst bei Bedarf Daten aus dem Internet abrufen.

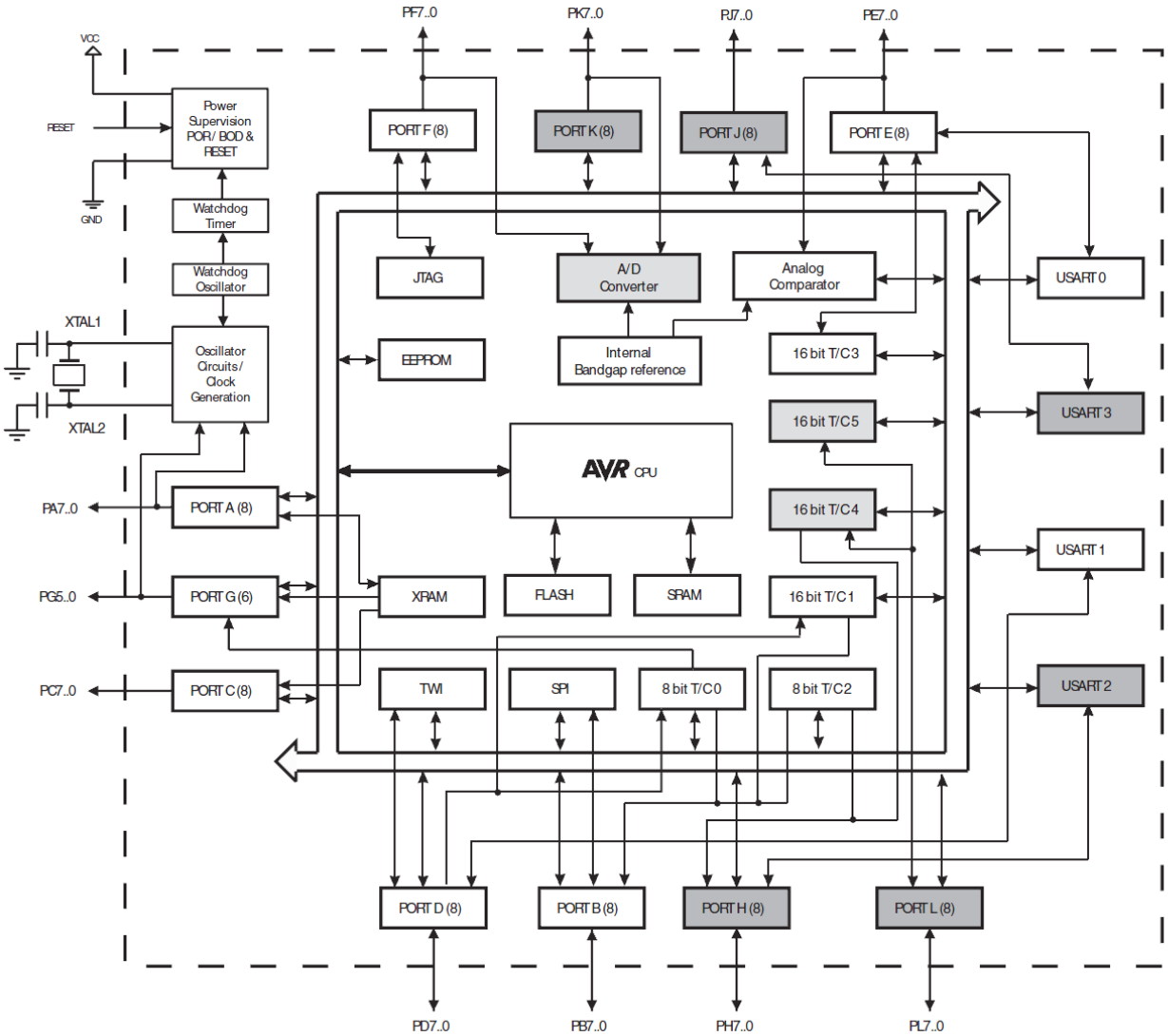


Das Modul enthält den größten Mikrocontroller der ATMEGA Serie der Firma Atmel, den ATMEGA2560, der mit 256KB Flash ROM (davon 8KB für den Bootloader reserviert) und 8KB SRAM mehr als genügend Speicher für sehr komplexe Anwenderprogramme zur Verfügung stellt. Die einfach anzuwendende Softwarebibliothek ist kompatibel zu den anderen AVR Prozessoren des RP6, man muss sich also nicht erst neu einarbeiten.

Eine Besonderheit des Moduls ist das auf der Platine enthaltene sehr energieeffiziente 802.11b/g WLAN Modul mit integriertem Prozessor. Der interne Prozessor kümmert sich um alle WLAN Funktionen, den TCP/IP Stack und die AES/WPA2 Verschlüsselung. Dies entlastet den AVR Hauptprozessor und vereinfacht die Ansteuerung für den Anwender (einfache Text Befehle, Ansteuerung über serielle Schnittstelle). Das Modul kann sich direkt nach dem Einschalten automatisch in einer Sekunde mit einem WLAN Netzwerk verbinden und mit Datentransfers beginnen.
Der Energiebedarf des gesamten Moduls ist sehr gering, bei aktivem 500kBit/s WLAN Datentransfer, maximaler Sendeleistung und laufendem Programm im ATMEGA2560 Prozessor benötigt das RP6v2-M256-WiFi Modul insgesamt weniger als 460mW. Damit ist es ideal für Batterie betriebene Anwendungen wie mobile Roboter.

Der ATMEGA2560 ist doppelt so schnell (16MHz) getaktet wie der Controller auf dem RP6 Mainboard und auch sonst steht dem Anwender auf dem RP6-M256 mehr Rechenzeit zur Verfügung, da der Controller nicht auch noch mit Motorregelung, ACS, IR-COMM, etc. beschäftigt ist.

Ein vereinfachtes Block Diagramm mit den Periphereie Einheiten des ATMEGA2560 Controllers ist auf dieser Seite abgebildet. Unter anderen erkennt man hier den AVR Kern in der Mitte mit Flash ROM und SRAM Speicher sowie Busanbindung zu den Peripherieeinheiten. Rundherum sind die zahlreichen 8 Bit I/O Ports, Timer, 4 USARTs (serielle Schnittstellen bzw. zwei dual SPI / UART Ports,), SPI, I²C Bus (TWI), A/D Wandler, Komparator, EEPROM, Taktgeber und Reset Einheit zu erkennen, alle über einen Bus miteinander verbunden.



Zusätzlich gibt es direkte Verbindungen von den speziellen Peripherie Einheiten wie den Timern oder den seriellen Schnittstellen zu den I/O Ports, damit diese bei Bedarf mit der Aussenwelt verbunden werden können. Bei den Timern also z.B. die PWM Ausgänge oder bei den seriellen Schnittstellen die Sende und Empfangssignale.

Das RP6-M256 Modul stellt über 60 freie 5V I/O Pins (+6 auf dem XBUS Steckverbinder und +2 auf dem UART Steckverbinder, d.h. bis zu 68 insgesamt) auf vielen einfach zu verwendenden 2.54mm Raster Steckverbindern zur Verfügung. Davon 16x A/D Wandler Kanäle, 12x Hardware PWM Ausgänge (z.B. für Servos, dimmbare LED Scheinwerfer, Piezo Signalgeber o.ä.), ein 8 Bit Display Port, 2x USART/SPI Ports, 22x externe Interrupts, 4 Timer Capture Eingänge, einen Analog Comparator, I2C Bus, USART und einen ISP Programmieranschluss (geteilt mit microSD Kartenslot).
Dazu noch 4x A/D Wandler Kanäle vom WLAN Modul. Zudem gibt es einen MicroSD Karten Slot für Flash Karten als Massenspeicher, zwei Taster und sieben Status LEDs (3x WLAN Modul, 4x Mikrocontroller).

Wie bei den anderen RP6 Erweiterungsmodulen (RP6-M32 und RP6-M128) kann ein LC-Display angeschlossen werden um den aktuellen Status anzuzeigen. Über die zahlreichen I/O Ports können Bedienelemente nachgerüstet werden.

Eine große 10cm 2.4GHz Antenne mit 2dBi Gewinn, 15cm u.FL zu RP-SMA Pigtail und Montagematerial runden die Ausstattung ab. Mit dem Montagematerial ist es für erfahrene Anwender möglich, die Antenne auch an anderen Positionen zu verwenden.

Neue Software lässt sich dank des internen Bootloaders über die WLAN Verbindung sehr bequem drahtlos in das Modul laden. Einmal passend eingerichtet ist nur ein Mausklick notwendig um ein neues Programm drahtlos in den ATMEGA2560 zu laden. Die Übertragungsgeschwindigkeit ist (bei guten Umgebungsbedingungen / wenig Störquellen) sehr ähnlich wie über die kabelgebundene USB Verbindung.

Das WLAN Modul kann über die RobotLoader Software vom Host PC aus über das RP6 USB Interface eingerichtet werden. Dies ist sowohl über eine Benutzeroberfläche oder per Text Terminal möglich. Das Mikrocontrollerprogramm muss sich bei passender Konfiguration nur wenig um das WLAN Modul kümmern und kann einfach Textnachrichten über die serielle Schnittstelle senden, so als wäre es per Kabel mit dem PC verbunden (natürlich können im Vergleich zur Kabelgebundenen Übertragung größere Verzögerungen entstehen oder die Verbindung abrupt abbrechen).

1.1. Sicherheitshinweise

Bevor Sie mit dem RP6-M256 loslegen, sollten Sie sich unbedingt mit dem Roboter ansich vertraut machen und alle Beispielprogramme des Roboters OHNE montiertes RP6-M256 Erweiterungsmodul ausprobieren! Diese Anleitung versteht sich als kleine Ergänzung zur großen RP6 Bedienungsanleitung. Lesen Sie diese auf jeden Fall bevor Sie mit dem RP6-M256 anfangen! **Die Sicherheitshinweise in der RP6 Anleitung gelten zusätzlich zu den hier gegebenen Hinweisen, insbesondere die Hinweise zu ESD und verletzungsgefahr durch scharfe Pins auf der Platine.**

Ein wichtiger Hinweis für Anfänger:

Für das RP6-M256 geschriebene Programme laufen natürlich NICHT korrekt auf dem Mikrocontroller der Basiseinheit und umgekehrt (komplett andere Pinbelegung und Taktfrequenz und völlig andere Hardwareausstattung)! Das gleiche gilt für alle anderen Erweiterungsmodule.

ACHTUNG:

Wenn Sie ein Programm in den falschen Controller laden, könnte dieser oder die gesteuerten Schaltungen durchaus beschädigt werden! Beispielsweise wenn ein I/O Pin normalerweise als Eingang verwendet, im Programm für den falschen Controller aber als Ausgang geschaltet und dann aufgrund der angeschlossenen Schaltung überlastet wird!

Normalerweise passiert nichts schlimmes wenn Sie sich mal vertun sollten, aber es kann hier für nichts garantiert werden! Der RobotLoader kann nicht unterscheiden für welchen Controller die Programme bestimmt sind, da die Hexfiles alle gleich aufgebaut sind. Er verhindert also nicht, dass Sie Programme in den falschen Controller laden! **Nutzen Sie die Funktion des RobotLoaders, verschiedene Kategorien anzulegen! Für jedes Erweiterungsmodul eine eigene Kategorie!** Für das RP6-M256 gibt es natürlich zusätzlich die separate Hexdatei Liste im WIFI Loader Tab des RobotLoaders.

ACHTUNG:

Das Antennenkabel sollte nur wenn unbedingt notwendig von dem kleinen Steckverbinder auf der Platine (u.FL) abgesteckt werden! Aufgrund der kleinen Abmessungen ist dieser Steckverbinder nur für wenige Steckzyklen ausgelegt. Der Stecker ist mit Sorgfalt zu behandeln!

Der große RP-SMA Steckverbinder am anderen Ende des Kabels ist deutlich robuster! Hier kann die Antenne problemlos dutzende male ab- und wieder angeschraubt werden. Allerdings sollte auch dies nur dann wenn nötig getan werden, da sich auch hier die Kontakte bei größerer Zyklenzahl abnutzen und sich somit die Signalqualität verschlechtern kann.

1.2. Technischer Support

Bei Fragen oder Problemen erreichen Sie unser Support Team wie folgt über das Internet (Bevor Sie uns kontaktieren **lesen Sie aber bitte diese Bedienungsanleitung vollständig durch!** Erfahrungsgemäß erledigen sich viele Fragen so bereits von selbst!):

- über unser Forum: `http://www.arexx.com/forum/`

- per E-Mail: `info@arexx.nl`

Unsere Postanschrift finden Sie im Impressum dieses Handbuchs. Aktuellere Kontaktinformation, Softwareupdates und weitere Informationen gibt es auf unserer Homepage:

http://www.arexx.com/

und auf der Homepage des RP6 Roboters:

http://www.arexx.com/rp6

Auch dem Roboternetz, der größten deutschsprachigen Robotik Community, kann man auf jeden Fall mal einen Besuch abstatten:

http://www.roboternetz.de/

Es gibt dort ein Forum für den RP6!

1.3. Lieferumfang

Folgende Dinge sollten Sie in Ihrem RP6v2 CONTROL M256 WIFI Karton vorfinden:

- Fertig aufgebautes RP6-M256 Modul
- 1 Stück 10cm 2.4GHz WLAN Antenne
- 1 Stück 15cm Pigtail Kabel, u.FL → RP-SMA
- Montagematerial für Antenne
  Die Antennenhalterung sollte bereits vormontiert sein.
  Das Antennenkabel ist bereits passend angeschlossen.
  Es muss nur noch die Antenne aufgeschraubt werden.
- 4 Stück 25mm M3 Distanzbolzen
- 4 Stück M3 Schrauben
- 4 Stück M3 Muttern
- 2 Stück Adapterstücke mit 2x M3 Bohrung für Montage von Displays
- 1 Stück 14 pol Flachbandkabel
- 1 Stück CD-ROM mit Software und Bedienungsanleitung

Aktualisierte Versionen der Software und dieser Anleitung gibt es auf unserer Homepage. Es wird empfohlen die Software und Dokumentation stets von dort herunterzuladen da die CD-ROM nur sehr selten aktualisiert werden kann!

1.4. Features und technische Daten

Dieser Abschnitt gibt einen Überblick darüber, was das RP6 CONTROL M256 zu bieten hat und dient gleichzeitig der Einführung einiger Begriffe und Bezeichnungen von Komponenten des Moduls.

Features, Komponenten und technische Daten des RP6 CONTROL M256 WIFI:

- **Leistungsfähiger Atmel ATMEGA2560 8-Bit Mikrocontroller**
  - ◇ Geschwindigkeit 16 MIPS (=16 Millionen Instruktionen pro Sekunde) bei 16MHz Takt, also doppelt so schnell getaktet wie der Controller auf dem RP6 Mainboard.
  - ◇ Speicher: 256KB Flash ROM, 8KB SRAM, 4KB EEPROM
  - ◇ Frei in C programmierbar (mit WinAVR / avr-gcc)!
  - ◇ 6 Hardware Timer (2x 8 Bit, 4x 16 Bit)
  - ◇ Software Bibliothek ist kompatibel mit den anderen RP6 Modulen
  - ◇ ... und vieles mehr (s. Datenblatt)!
- **Energieeffizientes 802.11b/g 2.4GHz WLAN Funkmodul**
  - ◇ Typ: Roving Networks RN-171 (für komplette Spezifikation siehe Datenblatt!)
  - ◇ eigener Prozessor mit komplettem TCP/IP Stack und WPA2-AES Verschlüsselung, einfache Ansteuerung über serielle Schnittstelle
  - ◇ verbindet sich innerhalb einer Sekunde automatisch mit dem WLAN Accesspoint, danach können sofort Daten übertragen werden
  - ◇ große 10cm 2.4GHz Antenne mit 15cm Pigtail Kabel
  - ◇ Reichweite max. 100m ohne Hindernisse (variiert je nach Störpegel)
  - ◇ Datenrate über serielle Schnittstelle 500kBit/s (identisch mit der Bitrate die für den Bootloader über die USB Schnittstelle im „HighSpeed" Modus verwendet wird). Für Telemetrieübertragung ist diese Geschwindigkeit mehr als ausreichend.
  - ◇ Datenrate über WLAN: max. 54MBit/s (in dieser Anwendung natürlich nicht real erreichbar), Standardeinstellung zur Erhöhung der Reichweite 6MBit/s.
  - ◇ Datenübertragung erfolgt transparent per TCP (oder UDP).
  - ◇ Die Firmware unterstützt zahlreiche weitere Protokolle (NTP, FTP Client, HTTP Client usw.), mehr dazu finden Sie in der Dokumentation des WLAN Moduls
  - ◇ Adhoc Modus per Jumper aktivierbar (Hersteller Konfiguration wird wiederhergestellt)
  - ◇ Firmware des WLAN Moduls kann per Internet Verbindung (FTP) aktualisiert werden
  - ◇ Levelshifter für 3.3V ↔ 5V sind auf der Platine enthalten. Daher kann der Mikrocontroller normal mit 5V laufen und ist zum restlichen RP6 und zahlreichen normalen Robotiksensoren / Aktoren kompatibel. 3.3V Sensorik kann natürlich über weitere Levelshifter angebunden werden.
- **Energiebedarf des gesamten RP6-M256 Moduls an 5V nur ca. 460mW mit aktivem 500kBit/s Datentransfer** (je nach Umgebungstemperatur, Peripherie Einsatz und Software kann dieser Wert abweichen)

- **Über 60 freie I/O 5V Ports** auf einfach zu verwendenden 2.54mm Raster Steckverbindern**, davon alternative Funktionen:**
  - ◊ 16x 10 Bit A/D Wandler Kanäle
  - ◊ 12x 16 Bit Hardware PWM Kanäle
  - ◊ 3x 8 Bit Hardware PWM Kanäle
  - ◊ 4x Timer Capture Eingänge
  - ◊ 8 Bit Display Port, auch als normale I/O Ports verwendbar
  - ◊ 2x USART/SPI Ports, können entweder als Master SPI oder als UART verwendet werden
  - ◊ 22x externe Interrupts auf diversen I/O Ports
  - ◊ I2C Bus
  - ◊ USART als Programmieranschluss, kann auch für andere Zwecke verwendet werden, insbesondere wenn das WLAN Modul für den Software Upload verwendet wird.
  - ◊ ISP Programmieranschluss (für fortgeschrittene Anwender)
  - ◊ JTAG Interface (über Adapterkabel nutzbar)
- **4 A/D Wandler Kanäle und 3.3V Spannung vom WLAN Modul** sind auf einem 6 Pin Steckverbinder zugänglich. Spannungsteiler vorgeschaltet um Spannungsbereich zu erweitern. Normal wird nur ein Messbereich 0 bis 0.4V, 0.5V Schwelle für wakeup Trigger und 1.2V maximale Spannung unterstützt. Die externen Spannungsteiler erweitern dies auf 0 bis 3.12V Messbereich, 3.9V Wakeup trigger, 9V maximale Spannung (empfohlen max. 5V wie auch bei den anderen A/D Wandlern). Diese können unabhängig von Mikrocontroller ausgelesen werden und auch zum aufwecken des WLAN Moduls verwendet werden. Spannungsteiler Faktor: 0.12821
- **microSD Kartenslot für Speicherkarten**
  - ◊ Ideal für Datenlogging, Missionsdaten oder ggf. Ablage von statischen Daten die per WLAN abgerufen werden sollen (Hinweis: die Datenrate ist natürlich eingeschränkt, dies taugt nicht als Fileserver!)
  - ◊ kompatibel mit SD und SDHC Karten die den SPI Modus unterstützen (sollte bei fast allen Karten der Fall sein)
  - ◊ SDHC ist aufwändiger, benötigt mehr Programmspeicher und Rechenleistung, daher sind Karten bis max. 2GByte sehr zu empfehlen. Größere Karten funktionieren allerdings auch. Im Programmcode ist dies konfigurierbar.
- **I²C-Bus Erweiterungsanschlüsse**
  - ◊ Kann beliebige I²C Bus Slaves ansteuern.
  - ◊ Der MEGA2560 auf dem Modul kann als Master oder Slave verwendet werden. Sinnvollerweise sollte er als Master verwendet werden und den Roboter komplett steuern (der Controller auf dem Mainboard übernimmt allerdings die Regelung der Motorgeschwindigkeit, ACS, IRCOMM, Akku Überwachung etc. selbstständig und entlastet so den Controller auf dem Erweiterungsmodul).
- **4 Status LEDs am Mikrocontroller**
- **3 Status LEDs am WLAN Modul (Verbindung, Aktivität, Fehlerstatus)**
- **2 Eingabetaster**

- **LC-Display Port**
  - ◊ Zum Anschluss eines 16x2 Zeichen LC-Displays. Auch andere LC-Displays lassen sich hier anschließen, z.B. 16x4 o.ä., allerdings lassen sich diese dann nur mit zwei Distanzbolzen befestigen und könnten an einer Seite überstehen... Am besten vor dem Kauf genau ausmessen und passendes Montagematerial mitbestellen! Ausserdem müssen Sie bei anderen Display Formaten als 16x2 evtl. ein paar Anpassungen in der Library vornehmen, damit es korrekt funktioniert (Hauptsächlich die Initialisierung des Displays und evtl. die Cursor Positionierung). Die Beispielprogramme sind alle auf ein 16x2 Zeichen Display zugeschnitten.
  - ◊ Das Display kann z.B. Textmeldungen/Menüs, Programmzustände oder Sensorwerte anzeigen.
  - ◊ 8 Bit Display Port mit allen Steuerleitungen. Auch als normale I/O Ports verwendbar, Doppelbelegung mit einem Teil des XMEM Interface (AD0-7, ALE, RD, WR, zusätzlich A8 und A9 auf weiterer 3 pol. Stiftleiste verfügbar).
- **Alle 4 externen Interrupts am XBUS Anschluss lassen sich verwenden.**
- **Anschluss für das USB PC Interface**
  - ◊ Für Programmupload. Dieser läuft genau wie beim Roboter selbst, schnell und einfach über das USB Interface und die komfortable RobotLoader Software.
  - ◊ Alternativ bzw. bei diesem Modul sogar primär kann auch das WLAN Modul zum laden von neuen Programmen verwendet werden. Die serielle Schnittstelle ist dann frei für andere Anwendungen (der Bootloader erzeugt allerdings beim Booten Ausgaben mit 38400 Baud auf der seriellen Schnittstelle, dies ist nicht änderbar).
  - ◊ Konfiguration des WLAN Moduls per RobotLoader Software
- **Bootloader vorinstalliert,** dadurch kein ISP Programmiergerät erforderlich.
  - ◊ Programme können über das RP6 USB Interface in den Mikrocontroller geladen werden.
  - ◊ Programme können ebenfalls drahtlos per WLAN in den Mikrocontroller geladen werden. Mit vergleichbarer Geschwindigkeit wie bei der Kabelverbindung.
  - ◊ Zahlreiche Konfigurationsmöglichkeiten: Boot Delay, LCD an/aus, WLAN Baudrate, WLAN Boot deaktivieren, Sprungvektoren einstellbar, automatischer Programmstart oder Start per Tastendruck. Nachträgliche Erweiterung des Bootloaders um zusätzliche Funktionen ist möglich (z.B. andere Protokolle oder andere Displays). Dies benötigt natürlich zusätzlichen Flash Speicher.
  - ◊ 8KB Flash ROM fest für Bootloader reserviert.
- **Abmessungen (LxB):** 112mm x 90mm, zahlreiche M3 Montagebohrungen

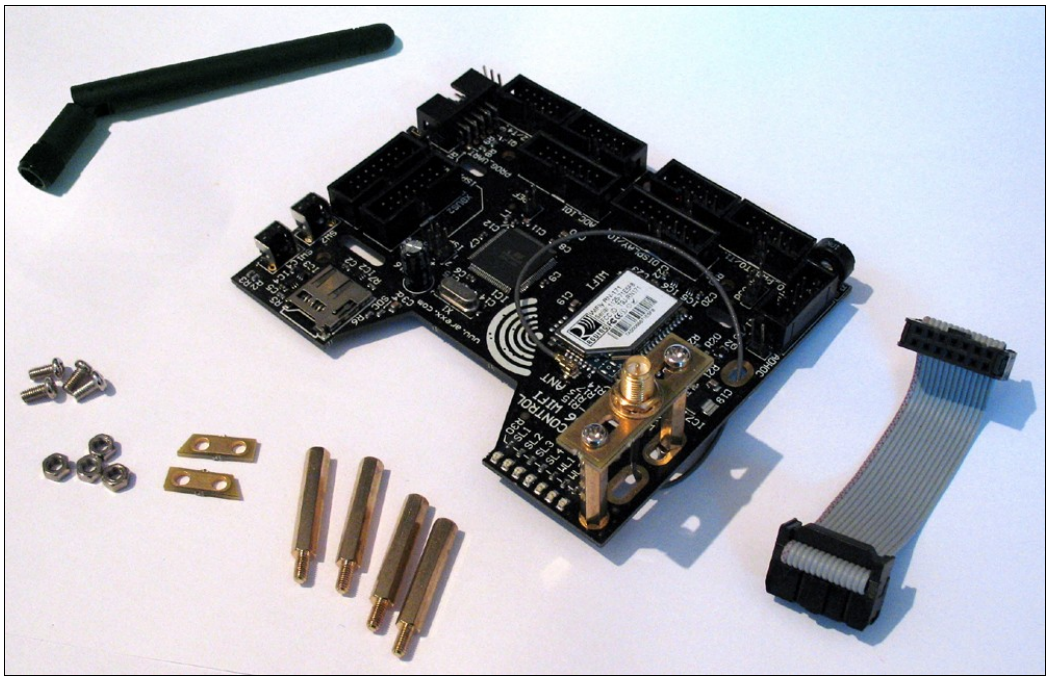
Weiterhin werden wie beim Roboter einige C Beispielprogramme sowie eine Funktionsbibliothek mitgeliefert, welche die Programmierung stark erleichtert.

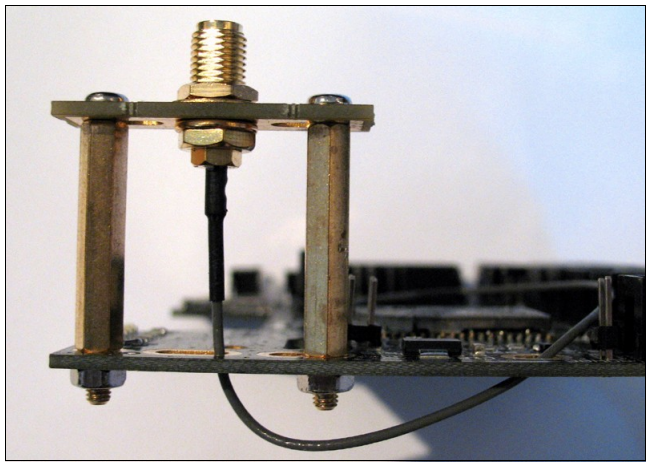
Auf der Website zum Roboter werden demnächst weitere Programme und Updates für den Roboter und seine Erweiterungsmodule zur Verfügung stehen. Sie können natürlich auch gerne Ihre eigenen Programme über das Internet mit anderen RP6 Anwendern austauschen! Die RP6M256Library und die Beispielprogramme stehen unter der Open Source Lizenz GPL!

2. Montage des Erweiterungsmoduls

Wie Sie das Modul auf dem Roboter befestigen, hängt natürlich davon ab, welche anderen Erweiterungsmodule Sie evtl. bereits auf dem Roboter montiert haben.

Folgendes Montagematerial, Antenne und Anschlusskabel sollte in der Packung enthalten sein:

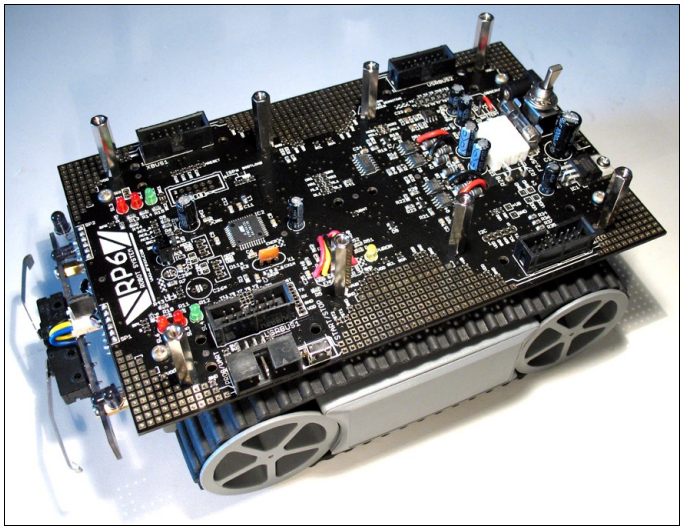




Wie man auf dem Bild erkennt, ist die Antennenhalterung mit dem goldenen RP-SMA (Reverse Polarity SMA) Steckverbinder **bereits vormontiert.**

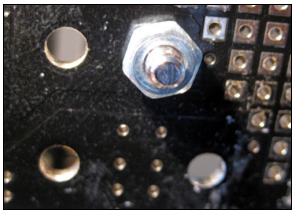
Bitte überprüfen Sie bei Lieferung ob dies der Fall ist. Weiterhin sollte überprüft werden, ob das Antennenkabel wie auf den Fotos verlegt wurde. Falls dies nicht so ist, können Sie das natürlich auch selbst korrigieren.

Der RP-SMA Steverbinder sollte fest in der Antennenhalterung sitzen. Falls nicht, können Sie es mit zwei Zangen oder einer Zange und einem 8mm Steckschlüssel vorsichtig festziehen (eine Zange hält die untere Mutter, die andere bzw. der Steckschlüssel dreht die obere). **Dabei nicht abrutschen und das Gewinde oder das Antennenkabel beschädigen!** Wie die Abflachung orientiert ist, spielt keine Rolle die Antenne kann später beliebig rotiert werden. Da das u.FL Kabel mit der Drehung komplett rotiert und somit auch der kleine Stecker am anderen Ende belastet wird, wurde auf eine Fixierung durch die Abflachung absichtlich verzichtet. So wird die Montage an anderen Orten vereinfacht (der Steckverbinder kann dann so rotiert werden, dass der kleine u.FL Verbinder auf der Platine möglichst wenig belastet wird).



Um das Modul auf dem Roboter zu montieren, müssen Sie zunächst die vier Schrauben vom Mainboard lösen. Eventuell können Sie auch den kleinen Stecker der Bumperplatine vorsichtig lösen, damit Sie das Mainboard komplett anheben können. Das ist aber nicht unbedingt erforderlich wenn Sie mit den Fingern auch so unter das Mainboard greifen können um die Distanzbolzen mit den M3 Muttern festzuschrauben.

Achtung: Beim wieder Anstecken des Kabels der Bumperplatine unbedingt mit einem Finger hinter der Sensorplatine gegenhalten damit diese nicht zu stark nach hinten gedrückt wird! Alternativ kann man auch die zwei Schrauben der Bumperplatine lösen und das Kabel dran lassen...

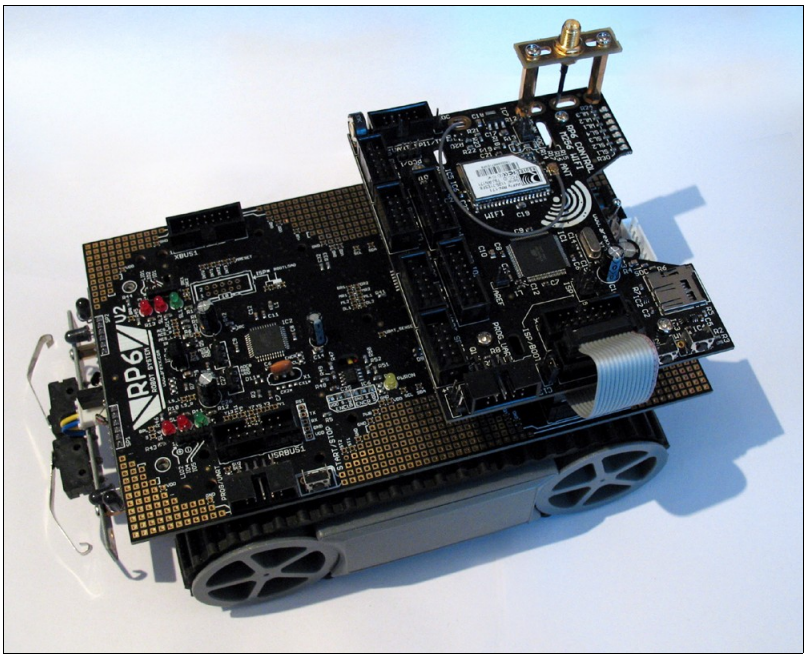


Dann können Sie die vier 25mm M3 Distanzbolzen nacheinander mit M3 Muttern in den Befestigungslöchern im Mainboard festschrauben wie auf dem Foto gezeigt.

Auf dem Bild oben sind alle 8 Distanzbolzen angeschraubt, auch die vom Lochraster Erweiterungsmodul!

Anschließend setzten Sie das Erweiterungsmodul oben auf die Distanzbolzen und Schrauben es mit den vier M3 Schrauben fest. Andersherum geht es natürlich auch: erst die Distanzbolzen an das Erweiterungsmodul anschrauben und danach mit Muttern am Mainboard befestigen.

Jetzt muss noch das 14 polige XBUS Flachbandkabel angesteckt werden – fertig.

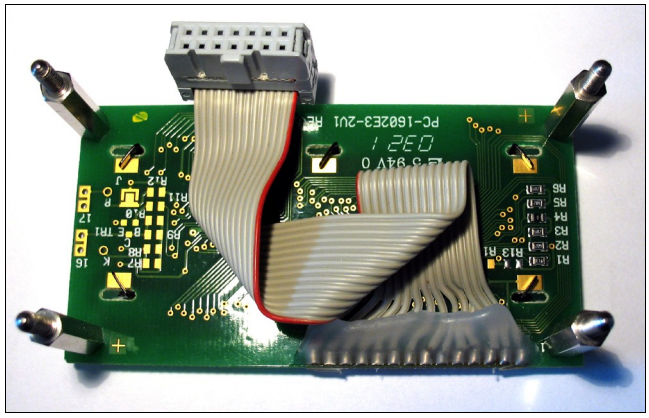


Anders als bei anderen RP6 Erweiterungsmodulen gibt es hier KEINEN USRBUS Anschluss. Auf der Platine war an der Stelle nicht genügend Platz.

Wir empfehlen das RP6-M256 auf dem hinteren Erweiterungsstapel auf dem Roboter zu montieren – als oberstes Modul, denn so sind die Antenne und das optionale Display optimal montierbar. Dann sind auch beide Programmieranschlüsse auf der gleichen Seite des Roboters zugänglich. Vorne können Sie das Experimentiermodul RP6-EXP anbringen (s. Foto auf der nächsten Seite für eine Beispielkonfiguration).

Wenn Sie zusätzlich das 16x2 Zeichen LC-Display besitzen, sollten Sie es VOR der Montage auf dem Roboter an das Erweiterungsmodul anschließen und montieren.

Das 14 polige Flachbandkabel des Displays ist sehr flexibel und kann problemlos gefaltet werden. Damit das Kabel gut unter das Display passt, sollten Sie es für das RP6-M256 Erweiterungsmodul in etwa wie auf dem Foto gezeigt falten (kann je nach Display Modell abweichen!).

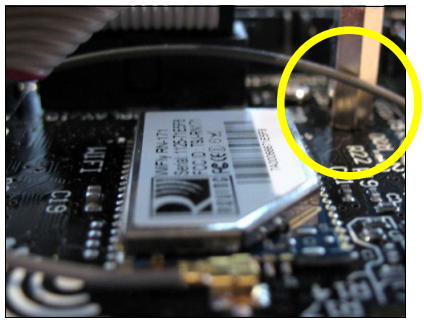


Dann kann das Display z.B. mit 20mm oder 25mm Distanzbolzen, passenden Muttern und Schrauben am Erweiterungsmodul befestigt werden.

Sie können auch ein beliebiges anderes Text LC-Display mit HD44780 kompatiblem Controller verwenden. Sie müssen dazu nur ein eigenes Kabel an das Display anlöten. *Dabei bitte unbedingt auf die richtige Anschlussbelegung achten! Hinweis: Die Initialisierung die der Bootloader vornimmt ist allerdings fest auf ein 16x2 Zeichen Display eingerichtet. Die Statusmeldungen können auch auf größeren Displays erscheinen, dann aber evtl. an anderer Stelle als gedacht.*

Es werden nicht unbedingt 4 Distanzbolzen benötigt wie auf dem Foto! Zwei Stück (<u>beide</u> vorne *oder* hinten) reichen bereits aus, um das Display zu befestigen.

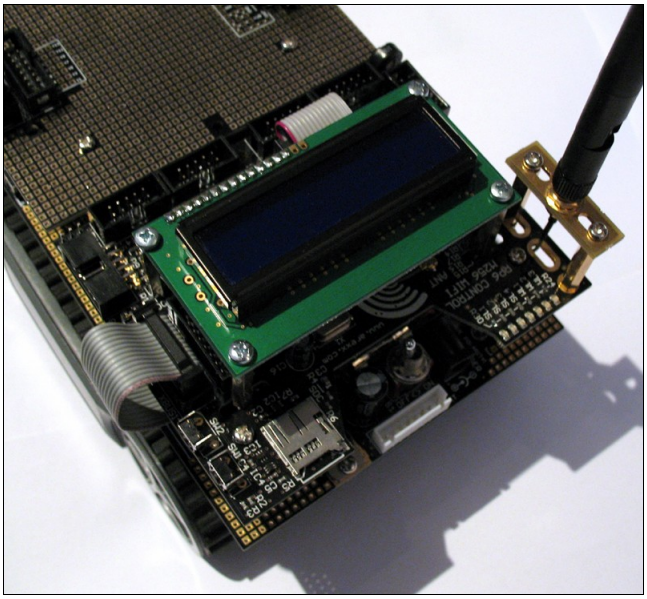
ACHTUNG: Bei der Montage des Displays aufpassen das nicht das dünne Antennenkabel <u>unter</u> den Distanzbolzen eingeklemmt wird! Dies kann an einer Stelle an der das Kabel entlang verlegt wurde recht leicht passieren, also vorsichtig montieren und das Kabel beiseite drücken wenn der Distanzbolzen festgeschraubt wird!



Tipp:

Es werden zwei kleine Platinenstücke mit zwei M3 Bohrungen beim RP6-M256 mitgeliefert. Diese sind OPTIONAL (d.h. werden normalerweise nicht benötigt) und können als Adapter für LCDs verwendet werden bei denen die Montagebohrungen nicht ganz exakt zu den Bohrungen auf der Platine passen. Sie müssen nur noch ggf. die Höhe der Distanzbolzen mit Muttern oder Unterlegscheiben anpassen.

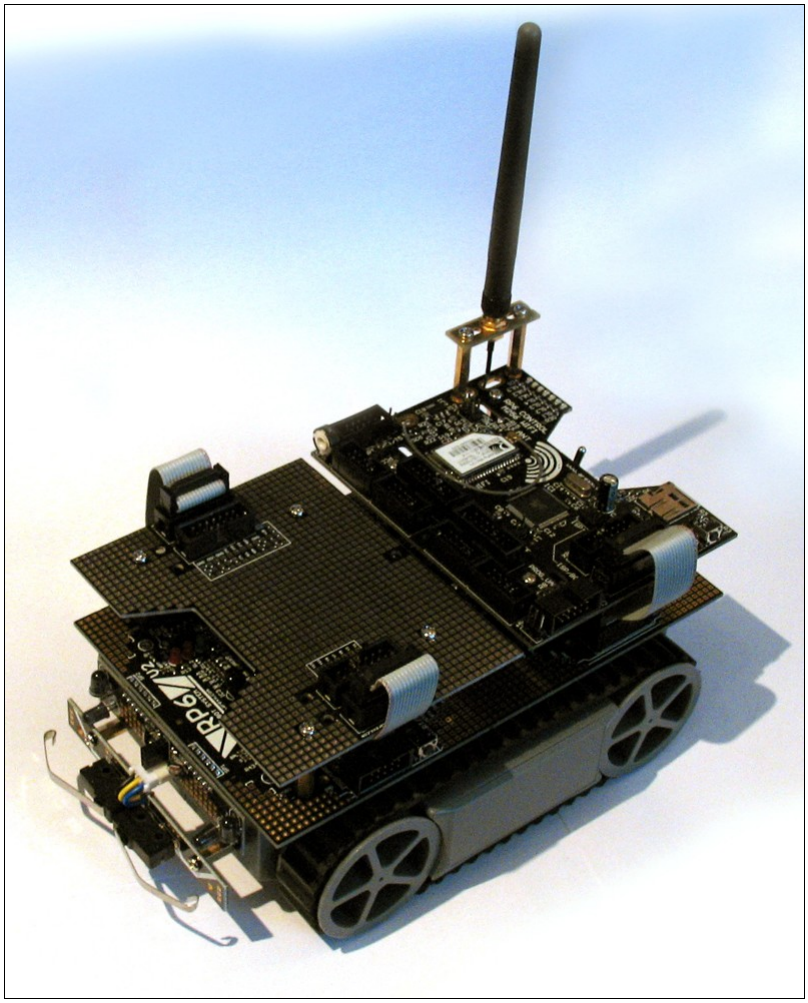
Die Platinenstücke können auch für andere Montagezwecke verwendet werden, z.B. um die Antennenhalterung an einer anderen Stelle zu montieren (mit nur knapp unpassenden Bohrungen). Auch hier werden ggf. zwei zusätzliche M3 Schrauben und Muttern benötigt.



Fertig montiert könnte das Display dann wie auf der Abbildung rechts aussehen.

Die Erweiterungsstecker mit den freien I/O Ports kann man einfach über kleine 10 und 14 polige Flachbandkabel mit einem oder mehreren RP6-EXP Erweiterungsmodulen verbinden. Die I/Os und ADCs können dort zur Auswertung von Sensoren o.ä. verwendet werden. Ein einziges Modul wird nur schwer ausreichend sein um die zahlreichen I/O Ports sinnvoll verwenden zu können, eine flexiblere Ausbaustufe wäre es, zwei RP6-EXP Module auf der ersten Ebene nebeneinander zu montieren und dann zusätzlich darüber auf einer zweiten Ebene eine weitere RP6-EXP vorne und das RP6-M256 hinten.

Ebenfalls denkbar ist eine Kombination mit dem RP6-M32 Modul. Eine Kombination mit dem RP6-M128 ist nur bedingt sinnvoll, da die C-Control Software leider den I2C Slave Modus nicht unterstützt. Das RP6-M256 Modul sollte aber stets der Busmaster sein. Denkbar wäre alternativ, dass C-Control Modul über eine der sekundären seriellen Schnittstellen anzusteuern. Dies erfordert natürlich Software Anpassungen.



Die Flachbandkabel passen gut durch die Lücke in der Mitte zwischen den zwei Modulstapeln.

Für fortgeschrittene Anwender ist es prinzipiell auch möglich hier das Kabel der WLAN Antenne hindurch zu verlegen um die Antenne auf einer anderen Ebene bzw. einem selbstgebauten Aufbau zu montieren.

Das RP6-M256 Modul kann prinzipiell sogar in völlig anderen Anwendungen als dem RP6 verwendet werden, auch hier ist die flexible Montage der Antenne von Vorteil (z.B. Montage in einem Gehäuse)

3. RobotLoader 2

Die neue RobotLoader Version 2.x (bis Version 1.4 auch RP6Loader genannt) bietet einige neue Funktionen, natürlich insbesondere Unterstützung für Netzwerk Verbindungen mit dem WLAN Modul. Die Neuerungen werden im Folgenden kurz erläutert.

3.1. Überblick

Genau wie beim RP6Loader und RobotLoader, muss der RobotLoader 2 nicht extra installiert werden, einfach in einen beliebigen Ordner auf der Festplatte entpacken und die enthaltene RobotLoader_start.exe oder unter Linux das Startscript robotloader_linux.sh bzw. robotloader_linux_x64.sh verwenden.

Wie in der Abbildung auf dieser Seite zu sehen, kann mit dem neuen WiFi Loader ähnlich wie mit der seriellen USB Verbindung Software in den Mikrocontroller geladen werden. Im oberen Bereich ist nun eine Liste mit IP Adressen im Format <IP>:<PORT> angezeigt, über die kleinen Buttons

+ *hinzufügen*, () *ändern* und **x *löschen***

kann die Liste manuell genau wie die Hexfile Liste editiert werden. Eingabe von IP und Port muss ebenfalls im Format <IP>:<PORT> erfolgen, also z.B. 192.168.10.171:2000. Der normale Port zur Kommunikation ist standardmäßig 2000, kann aber in den Einstellungen des WLAN Moduls geändert werden.

Prinzipiell reicht es dann die gewünschte Adresse auszuwählen, die Hexdatei wie vom normalen „Serial Loader" Tab bekannt auszuwählen und auf „Upload" zu klicken. Die Geschwindigkeit mit der dies geschieht ist bei guter Verbindung sehr ähnlich.

Wenn man auf Connect klickt, wird eine Verbindung aufgebaut und Informationen vom RP6-M256 abgefragt. Dies wird im unteren Bereich in der Box „Status" dargestellt. Direkt darunter befindet sich der RSSI Balken (Received Signal Strength Indicator), dieser zeigt die aktuell empfangene Signalstärke in dBm an (deziBel mit Bezugsgröße 1mW, z.B. 0dBm = 1mW, -30dBm = 0.001mW). Der Balken wird bei jedem klick auf „Connect" aktualisiert und zusätzlich auch über einen UDP Broadcast der automatisch alle paar Sekunden (einstellbar) vom WLAN Modul gesendet wird. Solange dieser Balken am rechten Ende gerade noch grün gefärbt ist, ist die Signalstärke im brauchbaren Bereich (alles besser als -65dBm). Färbt sich der Balken komplett rot ist die Signalstärke gering und die Verbindung könnte abbrechen,

wenn sich der Roboter noch ein paar Meter weiter vom Accesspoint entfernt.

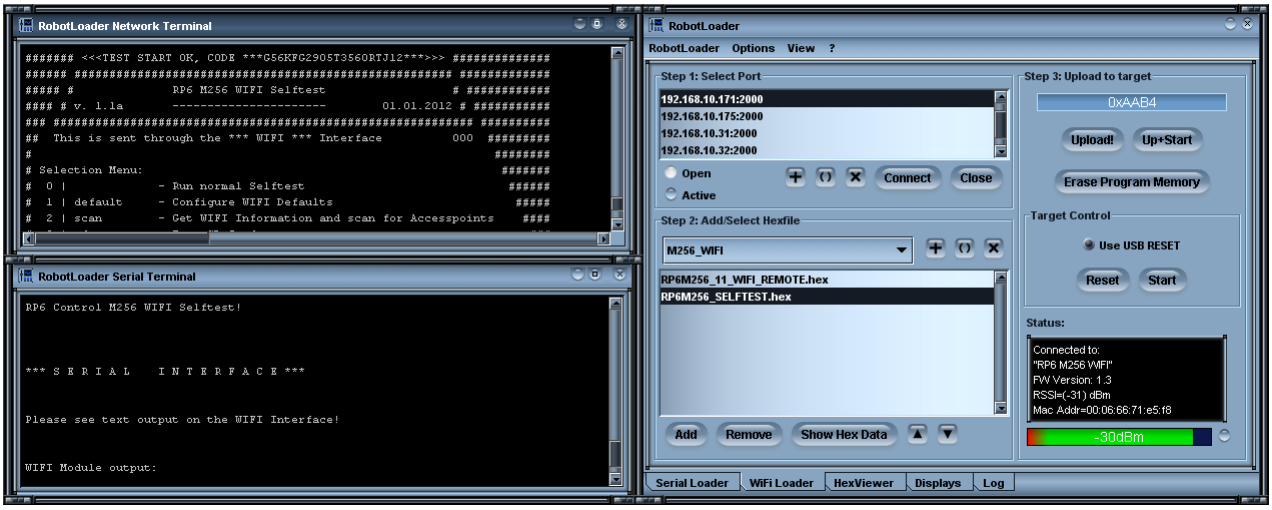
Aktzeptabel: 

Schlecht: 

Die Reichweite des WLANs ist stark von der Umgebung abhängig. Alle Hindernisse können je nach Material die Reichweite stark einschränken (insbesondere Metall und Wasser). Nachbar WLAN Netze, 2.4GHz Videotransmitter und Mikrowellenherde können das eigene WLAN ebenfalls stören.

3.2. Neue Terminals

Das serielle Terminal allein reicht nun natürlich nicht mehr aus und es ist ein Netzwerk / WIFI Terminal hinzugekommen. Prinzipiell verhält es sich sehr ähnlich wie das normale serielle Terminal. Da man nun sicherlich häufiger zwischen den Tabs hin und her wechseln muss, ist es nun möglich die Terminals in separate Fenster „abzudocken" und beliebig zu positionieren. Das kann dann z.B. wie in der folgenden Abbildung aussehen:



(Die Fenstergröße kann beliebig angepasst werden)

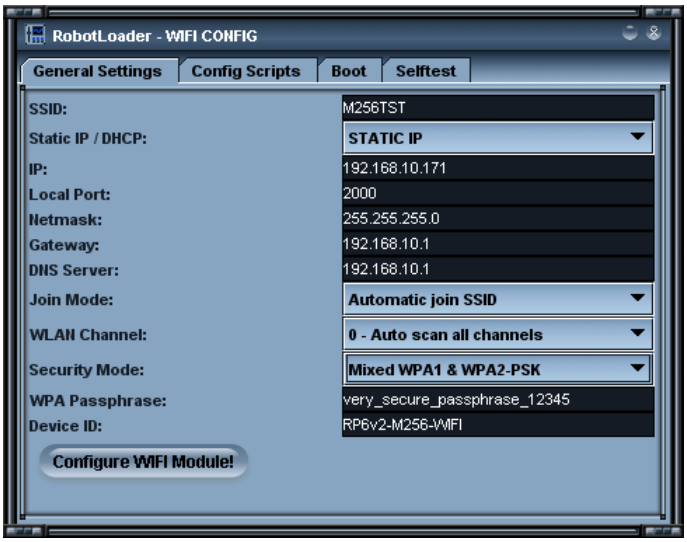
Nun kann man also serielle Schnittstelle und WIFI Terminal gleichzeitig für Ausgaben des Programms verwenden, was während der Entwicklungsphase von neuer Software für den RP6 sehr hilfreich sein kann - z.B. Ausgaben des Controllers auf dem Mainboard normal per serieller Schnittstelle und Ausgaben des RP6-M256 per WLAN gleichzeitig anzeigen lassen, oder auch zwei verschiedene Ausgaben vom RP6-M256 erzeugen.

Tipp:

Es ist auch möglich andere Terminalsoftware wie PuTTY oder Tera Term zu verwenden (schnellere Darstellung bei hohen Übertragungsraten). Diese kann man auf Telnet einstellen dann sollte eine Verbindung mit dem WLAN Modul möglich sein. Ggf. noch den Zeilenumbruch passend einstellen (CR oder CR+LF wird vom WLAN Modul Interpreter verwendet, LF oder CR+LF von den Beispielprogrammen → CR+LF wählen)

Das geladene Programm wird über WLAN anders als im seriellen Modus übrigens nicht mehr mit einem einfachen „s" gestartet sondern mit dem Befehl „start_program" (dies ist zuverlässiger über die WLAN Verbindung auch wegen diverser Statusmeldungen des WLAN Moduls).

3.3. WLAN Einstellungen



Die Einstellungen des WLAN Moduls können mit einem Konfigurationsdialog eingestellt werden. Dazu muss das USB Serial Interface mit dem RP6-M256 Modul verbunden sein. Die einmal vorgenommenen Einstellungen werden automatisch im WLAN Modul gespeichert und bleiben auch nach dem Abschalten erhalten.

Der Dialog unterstützt zur Zeit nur die Änderung der Einstellungen wenn die USB Kabelverbindung besteht. Über das Netzwerk Terminal lassen sich die Einstellungen auch über einen im WLAN Modul integrierten Kommandozeilen Interpreter konfigurieren. Dazu gibt es vom Hersteller des Moduls eine umfangreiche (englischsprachige) Dokumentation. Für einfache Anwendungen reicht es aber völlig, das Modul einmal zu konfigurieren, die Einstellungen müssen danach nicht mehr geändert werden. Das Modul verhält sich aus Sicht des Mikrocontrollers wie eine serielle Schnittstelle – mit dem Unterschied das auf dem PC eine normale TCP/IP Verbindung genutzt werden kann. Dies kann mit jeder gängigen Programmiersprache einfach genutzt werden und sogar (eingeschränkter) Zugriff per Webbrowser ist möglich.

Alle üblichen Parameter des verwendeten WLAN Accesspoints müssen in dem Dialog eingetragen werden. Abschließend genügt ein Klick auf „Configure WIFI Module" (zuvor bitte einmal im Serial Loader Tab den passenden Port auswählen).

Der RobotLoader wechselt dann automatisch auf die „Log" Ansicht und zeigt welche Kommandos ausgeführt werden und ob es erfolgreich war (grüne „OK!" Meldungen sollten nach den Kommandos erscheinen!). Dies kann auch im seriellen Terminal beobachtet werden. Alles was dort ausgeführt wird, kann man auch selbst manuell über Textkommandos erledigen, mit der Benutzeroberfläche ist es allerdings deutlich komfortabler.

Im „Configure Scripts" Tab des Konfigurationsdialogs können Scripte zur Konfiguration aufgerufen werden die man in Textform im RobotLoader Verzeichnis ablegen kann. Dies ist nützlich wenn man öfters mal den WLAN Accesspoint wechseln muss (wenn der Roboter an verschiedenen Standorten eingesetzt wird).

Nach diesem Allgemeinen Überblick, wird nun schrittweise die Konfiguration erläutert.

3.3.1. Accesspoint Konfigurieren

Die Konfiguration des WLAN Accesspoints ist von Modell zu Modell verschieden und kann der zugehörigen Anleitung entnommen werden. Falls möglich sollten Sie feste IP Adressen verwenden (DHCP deaktiviert), dies erleichtert es das WLAN Modul anzusprechen. Die RobotLoader Software kann das WLAN Modul auch automatisch finden, es gibt dazu den Dialog Options->Discover WIFI Devices. Dort werden die automatisch vom WLAN Modul gesendeten UDP Statusmeldungen in einer Liste dargestellt. Es können somit beliebig viele der RP6 WLAN Module aufgelistet werden. Über den dortigen „Add" Button kann die IP direkt der Liste im WIFI Loader Tab hinzugefügt werden.

Dies ist insbesondere bei Verwendung von DHCP äusserst nützlich. Für eigene Anwendungen ist es allerdings einfacher mit statischen IP Adressen zu arbeiten.

Als Verschlüsselung sollte auf jeden Fall WPA gewählt werden, auf gar keinen Fall WEP verwenden! Diese Verschlüsselung wurde schon vor langer Zeit ausgehebelt und ist somit kein ernsthaftes Hindernis mehr. Als WPA Passphrase natürlich nicht die Standardeinstellungen des Routers verwenden sondern selbst etwas ausdenken (die in der Abbildung zu sehende Passphrase ist übrigens kein besonders gutes Beispiel ;-)).

Die restlichen Einstellungen sind je nach Netzwerk zu wählen. Sie können verschiedene WLAN Kanäle durchprobieren falls einzelne durch Nachbarnetzwerke oder sonstige Störquellen stark belastet sind. Kanal 1, 6 oder 11 sind im 2.4GHz Band nicht überlappend.

Tipp:

Auf einem Notebook mit WLAN Karte kann man z.B. inSSIDer

http://www.metageek.net/products/inssider/

verwenden (für Linux mittlerweile als BETA Version verfügbar) um die Kanalbelegung darzustellen und einen möglichst wenig belegten Kanal zu wählen.

Weiterer Tipp:

Wenn der Accesspoint nur für die Robotersteuerung verwendet wird, kann auch der langsamere 802.11b Modus verwendet werden. Das hat den Vorteil das sich die Reichweite etwas erhöht. Dazu muss manuell das RN-171 Modul etwas umkonfiguriert werden. Der Accesspoint kann auch auf b/g (bzw. b/g/n) mixed Modus eingestellt werden. Allerdings beeinträchtigt dieser Modus die Übertragungsgeschwindigkeit von Geräten die im schnelleren 802.11g Modus arbeiten!

Auf der WLAN Kommandozeile (bzw. im Konfigurationsscript) aktiviert

```
set wlan rate 8
```

den normalen 6MBit/s 802.11g Modus (6MBit/s ist die langsamste Rate im 802.11g Modus, bessere Reichweite) und

```
set wlan rate 1
```

schaltet auf 2MBit/s 802.11b um.

Bei besseren Dualband Routern mit zwei Funkmodulen kann man die schnelleren Geräte auch auf einen separaten Kanal legen.

3.3.2. PC konfigurieren

Der Computer auf dem RobotLoader laufen soll, muss passend zum Accesspoint / Router konfiguriert werden. Am besten statische IP Adressen verwenden. Subnetzmaske und ggf. Gateway (IP des Routers, das muss nicht unbedingt der Accesspoint sein sondern ggf. ein zweiter separater Router für die Internet Verbindung) passend konfigurieren. Dies kann hier nicht für alle Betriebssysteme und Netzwerk Konfigurationen erläutert werden, ist aber i.d.R. sehr einfach und gut dokumentiert.

Dies kann so wie in der Abbildung des WLAN Konfigurationsdialogs dargestellt aussehen, z.B. Netzmaske 255.255.255.0, IP 192.168.10.x (x = 1 bis 254, aber nicht 255, denn das wäre die Broadcast Adresse!), ...

Testen Sie falls möglich die WLAN Verbindung zunächst mit einem anderen Gerät bevor Sie das RP6-M256 in Betrieb nehmen. Notebooks, Tablets oder Smartphones mit WLAN Funktion bieten sich hier an. Wenn es damit funktioniert, kann man das RP6-M256 passend einrichten.

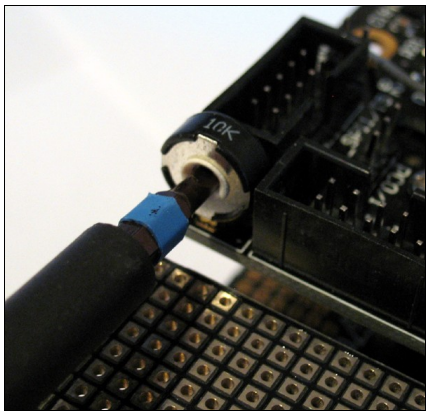
3.3.3. RP6-M256 USB Verbindung und Display testen

Nun kann das RP6-M256 getestet und konfiguriert werden.

Prüfen Sie zunächst ob der ISP / BOOT Jumper (kleine Steckbrücke direkt zwischen den XBUS und PROG / UART Steckverbindern) in der Position BOOT gesteckt ist (also die Position am Platinenrand). Wenn das nicht der Fall ist, wird es nicht funktionieren da kein Reset ausgelöst werden kann.

Nun das am PC angeschlossene USB Interface über das 10 polige Flachbandkabel mit dem PROG/UART Anschluss auf dem RP6-M256 verbinden und den RobotLoader starten. Dann den Roboter anschalten – auf dem LC-Display (sofern vorhanden) sollte eine Textnachricht erscheinen. Die Status LEDs sollten blinken (WL1 – 3 sind vom WLAN Modul und SL1 – 4 vom Mikrocontroller gesteuert – d.H. es sollten einige rote, grüne und gelbe LEDs direkt nach dem Anschalten mehrmals blinken). Hat das geklappt, klicken Sie im RobotLoader im Serial Loader Tab bitte auf Verbinden – es sollte in der Statusbox eine Meldung wie „Connected to RP6 Control" erscheinen.

Achtung: Nicht verwechseln! Hier für diesen Test muss der SERIAL Loader verwendet werden, NICHT der WiFi Loader! Mit dem WiFi Loader kann es noch nicht funktionieren, da das WLAN Modul noch nicht konfiguriert wurde.



Wenn das Display nichts oder nur zwei Reihen komplett dunkle Kästchen anzeigen sollte, aber die LEDs blinken und das Verbinden mit dem RobotLoader klappt, müssen Sie den Kontrast des Displays einstellen (oder Sie haben ein anderes Display verwendet und die Anschlussbelegung ist falsch... in dem Fall sofort den Roboter abschalten und die Verbindung prüfen). Der Kontrast kann mit Poti R1 auf der Platine eingestellt werden. Das Poti können Sie mit einem kleinen Schlitzschraubendreher verstellen. Kreuzschlitz geht auch, klappt dann aber mit vielen Schraubendrehern nicht, weil diese keinen guten Halt im Poti finden.

Das Flachandkabel des ggf. montierten Experimentiermoduls können Sie zum Einstellen entfernen damit man leichter an das Poti herankommt.

3.3.4. WLAN Modul konfigurieren und testen

Wenn der erste Test erledigt ist, kann sofort der WLAN Konfigurationsdialog gestartet werden, Menü `Options → WiFi SERIAL Config → Configure WiFi Settings`.

Zuvor noch kurz auf den WIFI Loader Tab im RobotLoader Hauptfenster wechseln und sicherstellen das die Checkbox `Use USB Reset` aktiviert ist. Dies ist wichtig, da bei angeschlossenem USB Interface das WLAN Modul den AVR Mikrocontroller nicht zurücksetzen kann. Dazu wird dann die USB Reset Leitung verwendet. Dies ist ebenfalls notwendig wenn das USB Interface an einem anderen Mikrocontroller angeschlossen ist (gemeinsame Reset Leitung, das USB Interface hat immer die Kontrolle darüber wenn es angeschlossen ist).

Dann im Konfigurationsdialog alle Einstellungen nochmals kontrollieren und auf den „Configure WIFI Module" Button klicken. Der Rest läuft dann wie weiter oben beschrieben ab. *Sollte es hier Probleme geben*, könnte sich das WLAN Modul evtl. noch im Auslieferungszustand des Herstellers befinden. In diesem Fall muss einmal ein Initialisierungsscript im „Configure Scripts" Tab ausgeführt werden, hier den Button

`Run Initial WIFI config (BAUDRATE IS 9600)` anklicken.

Sollte auch das fehlschlagen, überprüfen Sie die Stromversorgung und ob die WLAN LEDs tatsächlich blinken (eine der gelben LEDs und die grüne sollten blinken, das bedeutet das versucht wird eine Verbindung zum voreingestellten WLAN aufzubauen, was natürlich fehlschlagen wird). Bei Problemen sollten Sie den Support kontaktieren.

Wenn alles geklappt hat und der WLAN Accesspoint angeschaltet und in Reichweite ist, sollte keine der gelben LEDs mehr leuchten, stattdessen sollte nur die grüne LED langsam blinken. Das bedeutet, dass das WLAN Modul eine Verbindung aufgebaut hat.

Docken Sie nun im Menü über `View->Detach Network Terminal` bitte das Netzwerk Terminal ab und positionieren es neben dem RobotLoader Fenster. Dies erleichtert die Bedienung und man kann die Kommunikation mit dem WLAN Modul besser beobachten.

Nun kann die Verbindung mit dem RobotLoader getestet werden. Dazu im RobotLoader WiFi Loader Tab die passende IP Adresse in die Liste hinzufügen, z.B.

`192.168.10.171:2000`

Dann auf Connect klicken – in der Statusbox sollten nun Informationen zum Modul angezeigt werden (MAC Adresse, Firmware Version) sowie die Empfangsstärke dargestellt werden. Im Terminal sieht man einige Textausgaben. Die grüne LED hört nun auf zu blinken und leuchtet stattdessen dauerhaft (Verbindung offen).

Hinweis:

Wenn „Timeout" oder ähnliche Fehlermeldungen erscheinen und auch nach den automatischen weiteren Versuchen keine Verbindung aufgebaut werden kann, bitte einfach nochmal probieren auf connect zu klicken! Dies sollte nur selten passieren, kann aber je nach Netzwerk durchaus ab und zu vorkommen. Bei Verbindungsproblemen bitte versuchen zunächst auf „CLOSE" bzw. „Schließen" zu klicken und es dann nochmal probieren.

Wenn das geklappt hat, können Sie das Selftest Programm über die WLAN Verbindung auf den Mikrocontroller laden und starten. Bitte wirklich die WLAN Verbindung verwenden um zu testen ob alles funktioniert – d.H. in diesem Fall nicht den SERIAL Loader Tab verwenden!

Klicken Sie nach erfolgreichem Upload bitte auf den Button „Start".

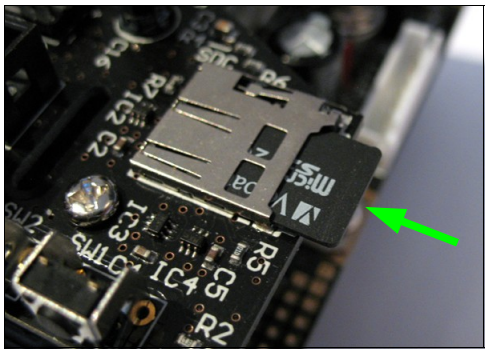
Wird das Programm korrekt ausgeführt? Es sollte ein Textmenü im Netzwerk Terminal erscheinen. Bitte beachten Sie, dass dieses Menü NICHT im normalen seriellen Terminal erscheint also nicht die beiden verschiedenen Terminals verwechseln!

3.3.5. MicroSD Karte

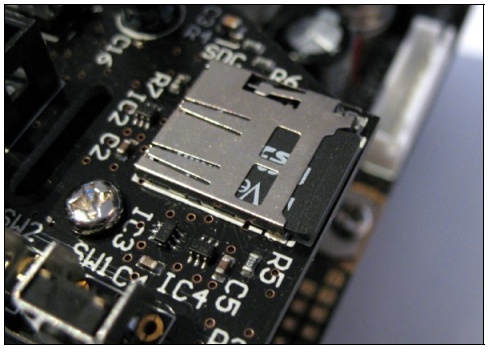
Um den Selbsttest vollständig durchführen zu können, benötigen Sie eine microSD Karte. Die SD Karte muss mit FAT32 formatiert sein, NTFS oder Linux ext3 wird nicht unterstützt. Wenn Sie keine microSD Karte besitzen, können Sie diesen Schritt auch überspringen dann wird ggf. zum Schluss eine Fehlermeldung angezeigt da keine SD Karte gefunden wurde, das kann dann natürlich ignoriert werden!

Es wird KEINE GARANTIE für jeglichen Datenverlust übernommen! Es wird direkt ohne Betriebssystem auf die Karte zugegriffen. Es kann auch ohne Rücksicht auf das Dateisystem auf die Karte geschrieben werden, wodurch alle Daten verloren gehen können. Zudem können Programmierfehler leicht zu Datenverlust führen. Also bitte eine Karte verwenden auf der keinerlei wichtige Daten enthalten sind! MicroSD Karten sind sehr billig, daher sollte es auch kein Problem sein eine eigene kleine 1 oder 2GB Karte nur für das RP6-M256 Modul zu besorgen.

Für den SD Karten Test müssen Sie mit einem passenden Kartenleser vom PC aus eine Testdatei ins Hauptverzeichnis der SD Karte kopieren. Die Datei finden Sie im Verzeichnis des Selftest Programms, sie heisst „M256_SELFTEST_TESTFILE.txt" und enthält etwas ASCII Text der vom Programm gelesen und über die WLAN Verbindung an den PC gesendet wird. Es wird auch ein Teststring am Anfang der Datei überprüft, nur wenn der Text korrekt gelesen wurde wird der Test als bestanden gewertet.



Die SD Karte kann wie dargestellt in den Kartenslot eingesetzt werden. Es handelt sich um einen sog. Push-Push Slot, d.h. man muss die Karte leicht in den Slot drücken um sie einzusetzen und auch einmal kurz leicht drücken und danach ziehen um sie wieder zu entfernen. Dabei ensteht ein leises Klick Geräusch. Den Roboter sicherheitshalber bitte VORHER abschalten und erst NACH dem Einsetzen der Karte wieder anschalten!



ACHTUNG:

Der microSD Kartenslot ist NICHT dafür geeignet die Karte im laufenden Betrieb zu wechseln! Das kann zwar funktionieren, es wird aber keine Garantie dafür übernommen!

Es wird allgemein keine Garantie für jeglichen Datenverlust übernommen! Am besten eine leere Karte verwenden.

Nun muss noch das RP6Base_I2CSlave.hex Programm in den Mikrocontroller auf dem RP6 Mainboard geladen werden! Dies ist notwendig um einen passenden I2C Bus Teilnehmer zur Verfügung zu haben mit dem die Bus Kommunikation getestet werden kann. Nachdem das Programm auf den MEGA32 auf der RP6Base geladen wurde, bitte das USB Interface wieder umstecken und mit dem RP6-M256 verbinden.

Dann kann der Selbsttest gestartet werden. Dazu bitte sobald das Auswahlmenü erscheint eine 0 eintippen und Enter drücken. Es werden nun Ausgaben auf der seriellen Schnittstelle und über die Netzwerkverbindung geschickt. Weiterhin sollten Ausgaben auf dem evtl. vorhandenen LC Display erscheinen. Folgen Sie den Anweisungen des Programms.

Zuerst müssen die beiden Taster SW1 und SW2 gedrückt werden. Direkt danach werden die roten LEDs getestet – also darauf achten ob alle leuchten (es werden nur die roten LEDs direkt vom Mikrocontroller gesteuert, die anderen LEDs sind mit dem WLAN Modul verbunden)!

Der I2C Test lässt einmal kurz ein kleines Lauflicht auf den LEDs des RP6 Mainboards laufen und prüft die Kommunikation.

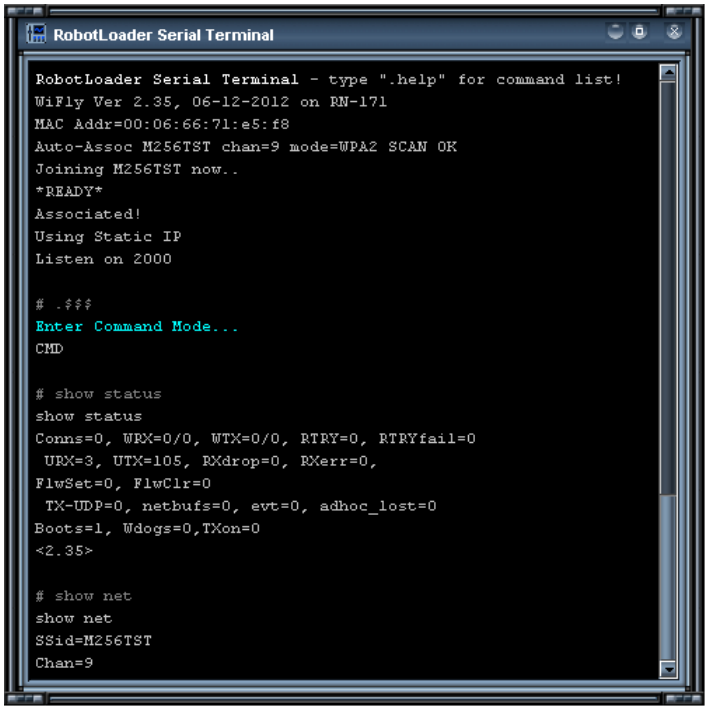
Zum Schluss des Tests wird der Text von der SD Karte gelesen und ausgegeben. **Wenn keine microSD Karte eingelegt ist, erscheint eine Fehlermeldung. Diese kann in diesem Fall natürlich ignoriert werden.**

Wenn dies alles funktioniert, können Sie nachdem Sie die Anleitung zuende gelesen haben mit den Beispielprogramm anfangen!

3.4. WLAN Kommandozeile

Über die Terminals kann auf zwei verschiedene Arten auf das WLAN Modul zugegriffen werden. Einmal natürlich per Netzwerk – das geht allerdings nur wenn die Verbindung ordnungsgemäß funktioniert. Weiterhin über die serielle Schnittstelle (das USB Interface). Dazu wird der Mikrocontroller in einen „Passthrough" Modus geschaltet in dem er alles vom PC an das WLAN Modul weiterleitet und umgekehrt. Dadurch kann das WLAN Modul sehr komfortabel konfiguriert werden. Der Mikrocontroller selbst muss dadurch prinzipiell auch nichts weiter selbst im Programm konfigurieren, da die Einstellungen des WLAN Moduls gespeichert werden können.

Um den Passthrough Modus zu aktivieren kann man einfach im RobotLoader Menü `Options → WiFi SERIAL Config → Enable Passthrough Mode` anklicken. Danach wird alles was im seriellen Terminal eingetippt wird an das WLAN Modul weitergeleitet (in den Optionen sicherstellen das CR oder CR+LF aktiviert ist!).



Durch eintippen von:

`.$$$`

wird in den Kommandomodus gewechselt (der Punkt am Anfang ist nur im RobotLoader Terminal notwendig und sorgt dafür, das davor und danach passend gewartet wird und insbesondere KEIN CR+LF gesendet wird – anders geht es im RobotLoader Terminal nicht, da stets nur bei Druck auf die Enter Taste gesendet wird und dann automatisch das Trennerzeichen hinzugefügt wird).

Im Kommandomodus hat man zahlreiche Möglichkeiten das WLAN Modul zu konfigurieren und den aktuellen Status abzufragen. Die auf der nächsten Seite folgende Tabelle listet kurz einige nützliche Befehle auf – ist aber bei weitem nicht vollständig. Hier muss auf die ausführliche Dokumentation des Modulherstellers verwiesen werden.

`ver`	Die Firmware Version angezeigen.
`show rssi`	Die aktuelle Signalstärke anzeigen.
`scan`	Nach Accesspoints scannen
`show status`	Aktuellen Status anzeigen
`show net`	Aktuellen Netzwerk Status anzeigen
`get everything`	Alle Einstellungen anzeigen
`get wlan`	WLAN Einstellungen anzeigen
`get ip`	IP Einstellungen anzeigen
`get comm`	Kommunikationsparameter anzeigen
`set ip adr 192.168.10.171`	IP Adresse einstellen
`set ip mask 255.255.255.0`	Netzmaske einstellen
`set ip gateway 192.168.10.1`	Gateway einstellen
`set ip dhcp 0`	DHCP deaktivieren
`set wlan join 1`	Automatisch mit Accesspoint verbinden
`set wlan ssid M256TST`	Accesspoint mit SSID M256TST verwenden
`set wlan auth 4`	WPA2 Accesspoints verwenden bei automatischer Verbindung zum nächsten Accesspoint
`set wlan passphrase <PASS>`	WPA Passphrase einstellen
`set wlan hide 1`	WPA Passphrase verbergen (für get Befehl)
`set wlan linkmon 5`	Alle 5 Sekunden UDP Broadcast auf Port 55555 senden und Verbindung zum Accesspoint prüfen.
`set comm size 1024`	Paketgröße einstellen. Hier: wenn 1024 Zeichen im Puffer sind, Netzwerkpaket senden.
`set comm time 10`	Timeout für Netzwerkpaket senden einstellen. Hier: wenn für 10ms keine neuen Daten über die serielle Schnittstelle eingetroffen sind, Netzwerkpaket senden
`save`	Einstellungen im Flash ROM speichern
`exit`	Kommandomodus verlassen
`reboot`	Modul neustarten

Wie gesagt ist die Tabelle bei weitem nicht vollständig und gibt nur einen kleinen Überblick. Dies muss man alles NICHT selbst erledigen, da der RobotLoader diese Einstellungen vorhnehmen kann.

Bitte ändern Sie nicht ohne guten Grund die I/O Port Konfiguration, dies ist passend eingestellt um den Mikrocontroller mit dem WLAN Modul zurücksetzen zu können, was für den Bootloader absolut notwendig ist.

Man kann den Passthrough Modus auch über die beiden Taster am Modul aktivieren. Hier ist es auch möglich notfalls in den 9600 Baud Modus umzuschalten wenn die Werkseinstellungen des WLAN Moduls wiederhergestellt wurden (Roboter auschalten, SW1 gedrückt halten, Roboter einschalten → Passthrough Modus 9600 Baud ist aktiv, Modul kann konfiguriert werden. Roboter anschalten und warten bis der Bootloader Timeout durch ist dann SW2 drücken → 500kBaud Passthrough Modus aktiv).

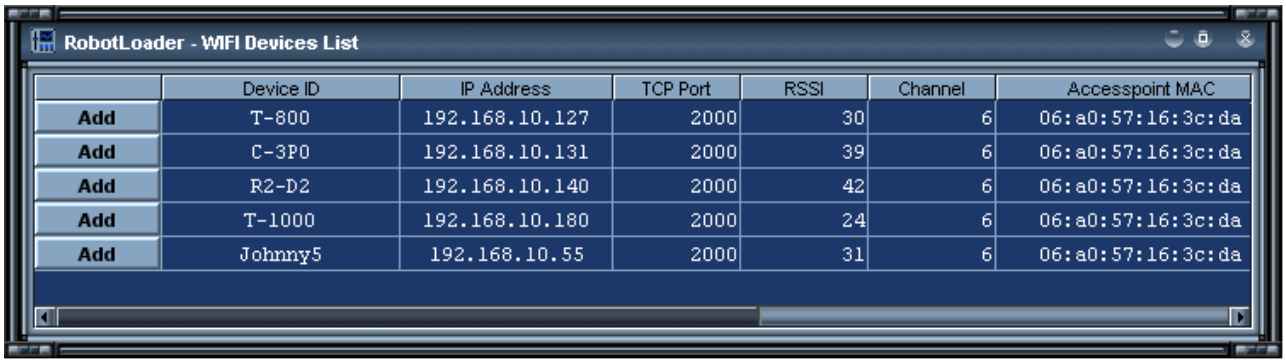
Das ist aber normalerweise nicht nötig, da RobotLoader dies auch ohne Tastendruck aktivieren kann.

Den Kommandomodus erreicht man auch über die Netzwerkverbindung. Dort funktioniert es sehr ähnlich per Eingabe von .$$$

Die Kommandos sind identisch, man muss aber beachten das die Verbindung abbricht wenn man die Einstellungen ändert und das Modul neu startet.

3.5. WLAN IP Adresse herausfinden

Ist die IP Adresse des Moduls nicht bekannt oder wird dynamisch per DHCP zugewiesen, kann man diese im „Discover WIFI Devices" Dialog herausfinden.



	Device ID	IP Address	TCP Port	RSSI	Channel	Accesspoint MAC
Add	T-800	192.168.10.127	2000	30	6	06:a0:57:16:3c:da
Add	C-3PO	192.168.10.131	2000	39	6	06:a0:57:16:3c:da
Add	R2-D2	192.168.10.140	2000	42	6	06:a0:57:16:3c:da
Add	T-1000	192.168.10.180	2000	24	6	06:a0:57:16:3c:da
Add	Johnny5	192.168.10.55	2000	31	6	06:a0:57:16:3c:da

Das WLAN Modul kann alle paar Sekunden eine Statusmeldung per UDP Broadcast an Port 55555 senden. Neue IP Adressen werden dann in der Liste und im Log des RobotLoaders angezeigt. Die hier angezeigte Device ID kann man übrigens in den WLAN Einstellungen verändern. Dabei allerdings keine Leerzeichen verwenden!

Es kann nach Einschalten der Versorgungsspannung bis zu 10 Sekunden dauern bevor die Statusmeldung ankommt. Es muss dazu keine Verbindung aufgebaut werden. Kontrollieren Sie aber ggf. vorher die Firewall Einstellungen um UDP Daten auf Port 55555 zu erlauben.

Per Klick auf den Button „Add" kann man die IP der Liste im WiFi Loader Tab hinzufügen.

3.6. WLAN Verbindungsprobleme beheben

Sollten Sie Probleme mit der Verbindung zum WLAN Modul haben, kann dies diverse Ursachen haben. Einige davon werden hier nun kurz dargestellt.

Das wichtigste: Der WLAN Accesspoint muss passend konfiguriert sein! Wenn das WLAN schon mit anderen Geräten nicht läuft, wird es vermutlich auch mit dem RP6-M256 nicht funktionieren. Ebenso müssen die Einstellungen des PCs und des WLAN Moduls dazu passen. Wenn das nicht der Fall ist, kann die Verbindung nicht funktionieren.

Wenn Sie sicher sind, dass das WLAN ordnungsgemäß arbeitet, müssen Sie zunächst feststellen wo es bei der Kommunikation mit dem RP6-M256 genau hakt.

Im RobotLoader können Sie den Mikrocontroller in den „Passthrough" Modus versetzen und direkt Befehle an das WLAN Modul senden wie weiter oben beschrieben .

Dort gibt es diverse Diagnose Funktionen (s. Anhang B für weitere Details). Prüfen Sie ob sich das WLAN Modul mit dem Accesspoint verbinden kann (die gelbe LED WL1 wird blinken wenn keine Verbindung zum Accesspoint aufgebaut werden konnte). Funktioniert die WLAN Verbindung, liegt das Problem evtl. an den Einstellungen des PCs.

Stellen Sie auch sicher, dass die Kommunikation nicht durch eine Firewall blockiert wird! Auf Windows PCs läuft i.d.R. mindestens die Firewall von Microsoft. Beim ersten Start des neuen RobotLoaders kann eine Sicherheitsabfrage erscheinen, da der RobotLoader sofort nach dem Start am UDP Port 55555 lauschen möchte. Das sollte erlaubt werden! Genauso wie TCP Datenverkehr mit dem WLAN Modul erlaubt werden muss.

Stellen Sie weiterhin sicher, dass das WLAN Modul eine IP Addresse verwendet, die nicht bereits ein anderer Rechner im LAN benutzt!

Alle Rechner müssen sich im selben IP Subnetz befinden, also bei Netzmaske 255.255.255.0 müssen die ersten drei Zahlen der IP Addresse übereinstimmen.

Also z.B. müssen dann alle IP Adressen so anfangen: 192.168.10.x und die letzte Zahl an der Stelle x kann von 1 bis 254 festgelegt werden.

Sind die Rechner hingegen im Subnetz 192.168.2.x oder ähnlich, kann nicht direkt mit dem WLAN Modul kommuniziert werden wenn seine IP 192.168.10.x ist.

Wenn Sie DHCP verwenden wollen, stellen Sie sicher, dass der DHCP Server ordnungsgemäß arbeitet und der DHCP Modus im WLAN Modul aktiviert ist.

Sollten Sie hier ihr spezielles Problem nicht gefunden haben – hier nochmals der Hinweis vom Anfang:

Bei Problemen können Sie uns per E-Mail oder über unser Forum sehr einfach und schnell Fragen stellen:

http://www.arexx.com/forum/

Aktualisierte Dokumentation, Softwareupdates und weitere Informationen gibt es auf der Homepage des Roboters:

http://www.arexx.com/rp6

4. RP6 CONTROL M256 WIFI Library

Wie für den Roboter selbst, gibt es auch für das RP6-M256 eine Funktionsbibliothek mit vielen hilfreichen Funktionen, die es Einsteigern bedeutend einfacher machen. Die Bibliothek heisst RP6 CONTROL M256 WIFI Library oder etwas kürzer RP6M256Lib. Die Stopwatch, Delay, UART und I²C-Bus Funktionen sind identisch mit denen der normalen RP6Library. Es werden für den I²C-Bus sogar die identischen Dateien verwendet. Diese liegen in der RP6Library im Unterordner RP6common. Die UART Library ist getrennt, da der interne Aufbau des ATMEGA2560 etwas abweicht und die UART Routinen teilweise für das WLAN Modul etwas optimiert wurden. Wir werden diese Funktionen hier nicht nochmal beschreiben da die Funktionsweise prinzipiell identisch ist. **Schauen Sie sich bitte das entsprechende Kapitel in der RP6 Bedienungsanleitung und die Beispielprogramme an!** Hier beschreiben wir nur Funktionen, die es nur für das RP6-M256 gibt, oder die etwas anders funktionieren als in der RP6Lib.

Hinweis:

Trotz der vorgefertigten Funktionen, ist die RP6M256Lib nur eine Grundlage! Perfekt ist diese Library bei weitem nicht. Man kann noch vieles verbessern und hinzufügen. Hier sind Ihre eigenen Programmierfähigkeiten gefordert!

Die Library liegt aktuell in einer frühen Version vor und wird noch erweitert. Neue Versionen werden auf der Webseite veröffentlicht. Dies gilt auch für die Beispielprogramme.

4.1.1. Mikrocontroller initialisieren

```
void initRP6M256(void)
```

Wie schon von der RP6Lib bekannt, muss diese Funktion IMMER als erstes in der Main Funktion aufgerufen werden! Sie heisst hier nur etwas anders.

Die Funktion initialisiert die Hardwaremodule des Mikrocontrollers auf dem RP6-M256. Nur wenn Sie diese Funktion als erstes aufrufen, wird der Mikrocontroller korrekt arbeiten! Ein Teil wird zwar schon vom Bootloader initialisiert, aber nicht alles.

Beispiel:

```
1  #include "RP6M256Lib.h"
2
3  int main(void)
4  {
5        initRP6M256();              // Initialisierung
6                                    // IMMER ALS ERSTES AUFRUFEN!
7  // [...] Programmcode...
8
9        while(true);      // Endlosschleife
10       return 0;
11 }
```

Jedes Programm für das RP6 CONTROL M256 muss mindestens so ausschauen! Die Endlosschleife in Zeile 9 ist notwendig, um ein definiertes Ende des Programms zu garantieren!

Ohne diese Schleife könnte sich das Programm anders verhalten wie erwartet! *Genau wie mit dem Controller auf dem Mainboard!*

Normalerweise wird in dieser Endlosschleife natürlich irgendetwas sinnvolles getan. Ideen für die generelle Struktur von Programmen sind in den Beispielprogrammen zu finden!

4.1.2. Status LEDs

Die LEDs lassen sich ähnlich wie auf dem Mainboard steuern, allerdings sind nur vier LEDs verfügbar.

Auch für das RP6-M256 heisst die Funktion „setLEDs“:

```
void setLEDs(uint8_t leds)
```

Beispiel:

```
setLEDs(0b0000); // Dieser Befehl schaltet alle LEDs aus.
setLEDs(0b0001); // Und dieser schaltet LED1 an und alle anderen aus.
setLEDs(0b0010); // LED2
setLEDs(0b0100); // LED3
setLEDs(0b1010); // LED4 und LED2
```

Die LEDs sind direkt mit I/O Ports des Mikrocontrollers verbunden (anders als z.B. beim RP6-M32 wo diese und das LC-Display an einem externen Schieberegister angeschlossen sind). Es gibt auch 4 Funktionen welche die LEDs einzeln setzen:

```
void setLED1(uint8_t led),  void setLED2(uint8_t led),
void setLED3(uint8_t led),  void setLED4(uint8_t led)
```

```
setLED1(1); // LED1 an
setLED2(0); // LED2 aus
```

4.1.3. Taster

Die zwei seitlichen Taster können mit den folgenden Routinen abgefragt werden.

```
uint8_t getPressedKeyNumber(void)
```

Diese Funktion ermittelt welcher Taster gerade gedrückt ist und gibt dann 1 oder 2 zurück. Ist kein Taster gedrückt wird 0 zurückgegeben.

```
uint8_t checkPressedKeyEvent(void)
```

Diese Funktion prüft ob ein Taster gedrückt worden ist und gibt dann ein einziges mal die Tastennummer zurück – im Gegensatz zu getPressedKeyNumber, wo die Tastennummer ständig zurückgegeben wird. Das ist nützlich um in der Hauptschleife die Tasten abzufragen ohne den Programmfluss zu unterbrechen.

Ähnlich ist es auch mit der Funktion:

```
uint8_t checkReleasedKeyEvent(void)
```

Hier wird allerdings der Tastenwert erst dann genau einmal zurückgegeben, wenn die Taste gedrückt und danach auch wieder losgelassen wurde. Auch diese Funktion blockiert den normalen Programmfluss nicht – man muss nicht mit einer Schleife darauf warten bis die Taste wieder losgelassen wurde.

4.1.4. LC-Display

Das LC-Display ist ideal um einfache Sensorwerte und Statusmeldungen auszugeben, während der Roboter nicht mit dem PC verbunden ist. Die Ausgabe auf das LC-Display funktioniert ähnlich wie bei der seriellen Schnittstelle – aber natürlich gibt es ein paar kleine Besonderheiten. Schauen Sie sich am besten die Beispielprogramme an, dann wird schnell klar wie man das LCD verwenden kann.

```
void initLCD(void)
```

Diese Funktion muss immer zu Beginn des Programms aufgerufen werden, um das LCD zu initialisieren. Wenn Sie statt des LCDs andere Peripherie an die I/O Ports angeschlossen haben, können Sie diese und andere LCD Funktionen einfach aus dem jeweiligen Programm entfernen.

```
void setLCDD(uint8_t lcdd)
```

Diese Funktion benötigen Sie normalerweise nicht – wir beschreiben Sie hier nur um kurz zu erläutern wie das Display angesteuert wird.

Das LCD kann im 8-Bit Modus betrieben werden. Es werden 8 Datenleitungen und drei Steuerleitungen verwendet (Enable (EN) und Register Select (RS), Read/Write (R/W), also anders als beim RP6-M32 und RP6-M128 werden alle Signale benutzt! Dies ist für Text Displays nicht sonderlich wichtig, kann aber z.B. für größere Punktmatrix Displays aus Performance Gründen wichtig sein). Diese Funktion setzt auch kurz das Enable Signal, damit das LCD die Daten übernimmt.

```
void writeLCDCommand(uint8_t cmd)
```

Diese Funktion ruft setLCDD auf, setzt allerdings die RS Leitung auf low, um einen Befehl an das LCD zu senden.

```
void clearLCD(void)
```

Sendet den Befehl zum Löschen des Display Inhalts an das LCD.

```
void clearPosLCD(uint8_t line, uint8_t pos, uint8_t length)
```

Löscht einen bestimmten Bereich des Displays. Die Parameter sind: Zeile, Startposition in der Zeile und Länge des zu löschenden Bereichs.

Beispiel:

```
clearPosLCD(0,10,5);   // löscht in der ersten Zeile des Displays
                       // die letzten 5 Zeichen!
```

```
void setCursorPosLCD(uint8_t line, uint8_t pos)
```

Setzt den Textcursor an eine bestimmte Position auf dem Display. Der Parameter line kann 0 für die obere, oder 1 für die untere Zeile sein. Der Paramter pos darf für 2x16er LCDs im Bereich von 0 bis 15 liegen.

```
void writeCharLCD(uint8_t ch)
```

Sendet ein einzelnes Zeichen an das LCD – das funktioniert analog zur writeChar Funktion für die serielle Schnittstelle. Allerdings muss man hier zunächst sicherstellen das der Cursor des Displays an der richtigen Position ist, denn sonst sieht man den Text nicht!

Beispiel:

```
setCursorPosLCD(1,5);  // positioniere Cursor in der zweiten Zeile, Zeichen 5.
writeCharLCD('R');     // jetzt wird „RP6" ausgeben, und zwar
writeCharLCD('P');     // beginnend an der Cursorposition!
writeCharLCD('6');
```

void writeStringLCD(char *string)

Analog zur entsprechenden Funktion für die serielle Schnittstelle, sendet writeStringLCD eine nullterminierte Zeichenkette aus dem SRAM an das LCD. Also sollten Sie diese Funktion nur verwenden, wenn der Text auch wirklich im RAM liegt und nicht nur fest vordefiniert ist. Dazu ist das Makro:

writeStringLCD_P(STRING)

besser geeignet, da hier der Text direkt aus dem Flashspeicher gelesen wird, ohne den Umweg über den Arbeitsspeicher.

void writeStringLengthLCD(char *string, uint8_t length, uint8_t offset)

Mit dieser Funktion kann ein Text mit einer bestimmten Länge auf dem LCD ausgegeben werden. Die Paramter sind identisch zu denen der entsprechenden Funktion für die serielle Schnittstelle.

showScreenLCD(LINE1,LINE2)

Um die Textausgabe auf dem LCD etwas zu vereinfachen, kann man mit dieser Funktion beide Zeilen des LCDs mit nur einem Aufruf beschreiben. Der Cursor wird automatisch richtig platziert und der Inhalt des Displays vorher gelöscht.

Beispiel:

```
showScreenLCD("LCD Zeile 1", "LCD Zeile 2");
```

void writeIntegerLCD(int16_t number, uint8_t base)

Die schon von der seriellen Schnittstelle bekannte Funktion um Zahlen in den Formaten BIN, OCT, DEC oder HEX auf dem LCD auszugeben.

void writeIntegerLengthLCD(int16_t number, uint8_t base, uint8_t length)

Auch writeIntegerLengthLCD ist bis auf den Namen identisch zur bereits bekannten Funktion für die seriellen Schnittstelle.

4.1.5. SPI Bus

Die SPI Funktionen sollten NICHT verwendet werden wenn die microSD Karte verwendet wird. Diese sind nur aus der RP6-M32 Library übernommen worden und werden hier eigentlich nicht benötigt. Man kann aber prinzipiell am ISP Anschluss weitere SPI Peripherie anschließen. Nur einen Chipselect Pin muss man sich noch von einem der anderen Pins organisieren und dabei beachten das hier die Levelshifter der SD Karte mit auf dem Bus liegen (über Vorwiderstände)! Besser ist es jedoch, die UART / SPI Ports für weitere SPI Peripherie zu verwenden, entsprechende Funktionen werden eventuell in Zukunft noch in die Library mit aufgenommen, bis dahin muss auf die AVR Dokumentation verwiesen werden.

```
void writeSPI(uint8_t data)
```

Überträgt ein Datenbyte über den SPI Bus.

```
writeWordSPI(uint16_t data)
```

Überträgt zwei Datenbytes die in einer 16 Bit Variablen übergeben werden über den SPI Bus, wobei das High Byte zuerst gesendet wird.

```
void writeBufferSPI(uint8_t *buffer, uint8_t length)
```

Überträgt bis zu 255 Bytes über den SPI Bus aus einem entsprechend großen Array. Die Anzahl der zu übertragenden Bytes im Array „buffer" wird mit dem Parameter „length" angegeben.

```
uint8_t readSPI(void)
```

Liest ein Datenbyte vom SPI Bus.

```
uint16_t readWordSPI(void)
```

Liest zwei Bytes vom SPI Bus und gibt diese als 16 Bit Variable zurück. Das zuerst gelesene Byte ist dabei das High Byte.

```
void readBufferSPI(uint8_t *buffer, uint8_t length)
```

Liest bis zu 255 Bytes vom SPI Bus in ein entsprechend großes Array.

4.1.6. ADCs

Alle ADC Kanäle lassen sich mit der schon von der RP6Lib bekannten Funktion:

```
uint16_t readADC(uint8_t channel)
```

auslesen. Eine automatische Variante die der Reihe nach die ADC Kanäle im Hintergrund ausliest gibt es für das RP6-M256 (noch) nicht, kann aber bei Bedarf auch selbst hinzugefügt werden.

Die Kanäle sind natürlich anders bezeichnet als in der RP6Lib – 16 Kanäle sind verfügbar: `ADC_15, ADC_14, … ADC_2, ADC_1 und ADC_0`

4.1.7. I/O Ports

Da auf dem RP6 CONTROL M256 insgesamt 60 freie I/O Pins verfügbar sind, beschreiben wir hier wie man allgemein auf I/O Ports eines AVRs zugreifen kann. Es werden NICHT alle Details zu den Hardwaremodulen wie Timer Capture, Output Modulator und ähnlichem beschrieben, dazu bitte das Datenblatt konsultieren.

Der ATMEGA2560 hat 9 I/O Ports zu je 8 Bit und einen mit 6 Bit. Jeder Port wird über 3 Register gesteuert. Ein Register für die „Richtung" der I/O Pins (DDRx), also ob ein Pin als Eingang oder Ausgang geschaltet ist, ein Register zum Schreiben (PORTx) und ein Register zum Lesen (PINx).

Wenn man einen I/O Pin als Ausgang verwenden will, um z.B. eine LED zu schalten, muss man das entsprechende Bit im DDRx Register auf 1 setzen. Das x steht hierbei für den Namen des jeweiligen Ports. A, B, C, … bis L (PORTI gibt es nicht).

```
Beispiel:
DDRL |= IO_PL5_OC5C; // PL5 ist nun Ausgang
DDRL = IO_PL5_OC5C | IO_PL4_OC5B | IO_PL0_ICP4; // PL5, PC4, PC0 sind nun Aus-
gänge, alle anderen Pins des PortL sind Eingänge!


DDRE |= IO_PE2_XCK0_AIN0 | IO_PE4_OC3B_I4; // PE2, PE4 sind nun Ausgänge
```

Dann kann man über das PORTx Register den Ausgang auf high oder low Pegel schalten.

```
Beispiel:
PORTL |= IO_PL5_OC5C;  // High
PORTL &= ~IO_PL5_OC5C ; // Low
```

Ist ein Bit im DDRx Register 0, so ist der zugehörige Pin als Eingang konfiguriert.

```
Beispiel:
DDRD &= ~IO_PD7_T2;  // PD7 ist nun Eingang
```

Dann kann man über das PINx Register den Zustand des Pins auslesen, also ob high oder low Pegel am Pin anliegt.

```
if(PINC & IO_PC6)
      writeString_P("PC6 is HIGH!\n");
else
      writeString_P("PC6 is LOW!\n");
```

Man kann weiterhin die im Mikrocontroller integrierten Pullup Widerstände aktivieren indem man die Bits im PORTx Register setzt (gilt natürlich nur wenn der Port als Eingang konfiguriert ist).

Die ADC Kanäle können ebenfalls als I/O Pins verwendet werden! Bitte beachten Sie die unterschiedlichen Schreibweisen verglichen mit den Bezeichnungen oben (ADC_7 vs. ADC7)!

```
IO_ADC7, IO_ADC6, …
```

Die Definition aller I/O Pins finden Sie in der Header Datei RP6M256.h der RP6M256-Lib. Es können prinzipiell auch einfach direkt Zahlen verwendet werden, allerdings sind Namen oftmals übersichtlicher.

Anstelle von z.B. IO_PE6_T3_I6 kann man übrigens auch (1 << PINE6) schreiben. Das ist identisch und so nur im Header RP6M256.h der Library definiert.

Der Bootloader konfiguriert übrigens nach einem Reset alle I/O Ports auf den Erweiterungssteckverbindern als EINGANG mit Pullup Widerstand. Ausnahme sind die Ports des LC-Display Steckverbinders. Diese sind nur dann Eingänge wenn das Display im Bootloader deaktiviert wurde. Achtung: In der Library sind diese dennoch fest als Ausgänge definiert – dies muss ggf. geändert werden!

Hinweis:

Die einzelnen I/O Pins sind für einen maximalen Strom von 20mA ausgelegt. Insgesamt sollte ein 8 Bit Port nicht mit mehr als 100mA belastet werden! Die Gesamtlast an allen I/O Ports darf 400mA nicht überschreiten. Wenn Sie also größere Verbraucher schalten wollen, müssen Sie dies über externe Transistoren tun! Alle größeren Verbraucher wie LED Scheinwerfer und Motoren benötigen definitiv externe Leistungstreiber, diese können NICHT direkt an den I/O Ports angeschlossen werden (Modellbau Servomotoren haben den Leistungstreiber mit Regelung bereits integriert).

Für genauere Informationen müssen wir hier auf das Datenblatt des MEGA2560 verweisen, das auf der RP6 Webseite / CD-ROM zu finden ist.

4.1.8. Internes EEPROM

Das im Mikrocontroller integrierte 4096 Byte EEPROM kann mit zwei kleinen Hilfsfunktionen angesteuert werden. Es ist nützlich um Informationen dauerhaft abzuspeichern, da diese auch nach dem Abschalten der Stromversorgung erhalten bleiben.

Mit

```
uint8_t readINTEE(uint8_t adr)
```

kann ein Byte von einer bestimmten Adresse gelesen werden. Ein Byte an eine bestimmte Adresse schreiben erfolgt mit:

```
void writeINTEE(uint8_t adr, uint8_t data)
```

Der Adressbereich umfasst wie gesagt 4096 Bytes, d.h. 0 bis 4095 sind die möglichen Adressen.

ACHTUNG:

Die ersten 32 Bytes sind für die Einstellungen des Bootloaders reserviert! Überschreiben Sie die ersten 32 Bytes (Adressen 0 bis 31) also bitte nicht versehentlich, sonst gehen die Einstellungen verloren. Dieser Datenbereich ist per Checksumme gesichert, d.h. der Bootloader erkennt fehlerhafte Daten und überschreibt diese bei Fehlern mit standard Werten!

Zusätzlich der Hinweis:

Das EEPROM hat eine typische Lebensdauer von einigen Millionen Schreibzyklen. Dies ist im Datenblatt genauer spezifiziert. Das EEPROM sollte daher nur zur Speicherung von Einstellungen und ähnlichem verwendet werden und die Zyklenzahl soweit wie möglich reduziert werden (also besser nicht 10x pro Sekunde eine Variable im EEPROM speichern). Größere Datenmengen können besser auf der SD Karte aufgezeichnet werden!

Hinweis:

Der Prozessor verfügt noch über einige weitere Hardwaremodule die nicht extra eigene Funktionen in der Library erhalten haben.

Dies kann zu einem späteren Zeitpunkt noch nachgeholt werden, ist bei einigen spezielleren Modulen jedoch unwahrscheinlich (Modulator, Komparator, …). Hier müssen Sie (wie üblich) selbst das entsprechende Kapitel im Datenblatt studieren und eine eigene Funktion für die Ansteuerung schreiben. Dies ist meistens auch sinnvoller um es passend für die eigene Anwendung zu gestalten.

4.1.9. microSD Karte

Der microSD Kartenslot kann über eine zusätzliche Library angesteuert werden, die im RP6M256Lib Verzeichnis enthalten ist. Diese bietet zahlreiche Funktionen für den Zugriff auf Partitionen und FAT16/32 Dateisysteme.

Die Library enthält eine ausführliche Dokumentation im HTML Format im Verzeichnis:

RP6Lib/RP6control_M256_WIFI/sdc/doc/html/

In den Beispielprogrammen wird die Verwendung kurz demonstriert. Im Beispielprogramm 4 wird gezeigt wie allgemeine Infos über die microSD Karte abgerufen werden können und eine Datei gelesen werden kann.

In Beispiel 14 werden Sensordaten einmal pro Sekunde in eine Datei geloggt. Mit dem Webserver Code aus Beispiel 13, kann der Inhalt der Logdatei über einen Web Browser abgerufen werden.

ACHTUNG:

Der microSD Kartenslot ist NICHT dafür geeignet die Karte im laufenden Betrieb zu wechseln! Das kann zwar funktionieren, es wird aber keine Garantie dafür übernommen!

Es wird allgemein keine Garantie für jeglichen Datenverlust übernommen! Am besten eine leere Karte verwenden. Auf keinen Fall eine Karte mit wichtigen Daten verwenden!

Hinweis:

Es kann passieren, dass das Dateisystem / die Partitionstabelle auf der Karte beschädigt wird, wenn ein Programmfehler auftritt oder man abrupt Abschaltet/Reset auslöst während auf die Karte geschrieben wird. Dann muss die Karte ggf. neu formatiert werden, z.B. mit dem Linux Tool mkfs.vfat oder im Windows Explorer rechts auf den (richtigen) Datenträger klicken und formatieren wählen. Schlimmstenfalls könnte es auch notwendig sein eine neue Partitionstabelle auf der Karte z.B. unter Linux mit fdisk zu erstellen oder in Windows mit der Datenträgerveraltung. Alle Daten auf der Karte gehen dabei natürlich verloren.

Es ist ratsam eine Funktion ins eigene Programm einzubauen, welche das Schreiben sicher beendet – z.B. wenn einer der Taster gedrückt wird, ein bestimmter Befehl gesendet wird oder die Akkuspannung unter einen gewissen Schwellwert fällt. Das sollte man dann auch stets auslösen bevor der Roboter abgeschaltet wird.

Das ist eine normale Einschränkung bei Datenträgern mit Dateisystem. Mit dem PC könnte das Dateisystem ebenfalls beschädigt werden, wenn man die Karte während eines Schreibvorgangs einfach aus dem Kartenleser zieht. Auf dem AVR könnte man alternativ auch im Rohdaten Modus arbeiten, also ganz ohne Dateisystem. Dann kann die Karte allerdings nicht mehr ohne weiteres am PC genutzt werden...

4.1.10. WIFI Library

Die WIFI Library bietet einige Hilfsfunktionen um mit den WLAN Modul zu kommunizieren.

4.1.10.1. Daten Kommunikation

Grundsätzlich funktioniert das Senden von Daten fast genau wie beim normalen UART, die jeweiligen Funktionen haben einfach nur ein _WIFI hinten angefügt. Also in der Form:

```
void writeChar_WIFI(char ch)
void writeString_WIFI(char *string)
void writeStringLength_WIFI(char *data, uint8_t length, uint8_t offset)
void writeInteger_WIFI(int16_t number, uint8_t base)
void writeIntegerLength_WIFI(int16_t number, uint8_t base, uint8_t length)
```

zum Schreiben und

```
void clearReceptionBuffer_WIFI(void)
uint16_t getBufferLength_WIFI(void)
char readChar_WIFI(void)
uint16_t readChars_WIFI(char *buf, uint16_t numberOfChars)
```

zum Lesen.

Ein kleines Detail ist bei der writeChar_WIFI Funktion zu beachten (und somit auch bei allen anderen Funktionen die diese Funktion verwenden): Diese Funktion unterstützt Flusskontrolle und kann somit auch blockieren wenn das WLAN Modul mal länger braucht um ein Paket per Netzwerk zu versenden. Dies muss ggf. im eigenen Programm berücksichtigt werden.

Die Funktionsweise ist ansonsten identisch mit den bereits vom RP6 bekannten Funktionen, daher werden hier nur die zusätzlichen Funktionen genauer erläutert.

Ein kleines Beispiel soll an dieser Stelle genügen:

```
writeString_P_WIFI("Hallo Echo!\n");
```

Sendet den Konstanten Text „Hallo Echo!" mit Zeilenumbruch an das WLAN Modul, welches diesen wiederum per WLAN weiterleitet. Der Text kann dann z.B. im Terminal des RobotLoaders erscheinen.

```
writeChar_WIFI('A');
```

Sendet das ASCII Zeichen A.

```
uint8_t test = 42;
writeInteger_WIFI(test,DEC);
```

Sendet den numerischen Wert der Variable test im Dezimalformat als ASCII Text, in diesem Fall also 42.

Viele weitere Details finden Sie in den Beispielprogrammen. Der Empfang von Text wird ebenfalls in den Beispielprogrammen gezeigt und ist auch in den anderen RP6 Beispielprogrammen erläutert.

4.1.10.2. Kommandomodus

Für die eigentliche Datenkommunikation mit dem PC sind die folgenden Funktionen nicht nötig. Wenn das Modul einmal passend mit dem RobotLoader eingestellt wurde, läuft es nahezu transparent und kann für den Mikrocontroller wie eine serielle Schnittstelle verwendet werden. Für einige weitergehende Funktionen die das WLAN Modul bereitstellt werden demnächst in einer neuen Version der Library zusätzliche Funktionen bereitgestellt.

Der Kommandomodus des WLAN Moduls wird mit folgender Funktion aktiviert:

```
void enter_cmd_mode_WIFI(void)
```

Nachdem diese Funktion ausgeführt wurde, befindet sich das WLAN Modul im Kommandomodus. Je nachdem welche Firmware Version auf dem WLAN Modul läuft, geht der Wechsel in den Kommandomodus entweder über Eingabe von $$$ und eine Wartezeit oder alternativ (viel schneller) auch über einen I/O Port. Ab Firmware Version 2.32 ist der GPIO Modus verfügbar und kann im Quellcode der Library aktiviert werden (das Modul muss entsprechend konfiguriert sein, ist normal deaktiviert). Die neueste Version der Library stellt sich automatisch über die Konfigurationsdaten die der RobotLoader im EEPROM ablegt ein. Daher kann einfach im RobotLoader im WLAN Konfigurationsdialog die Verwendung des GPIO14 Pins aktiviert werden!

Mit

```
void leave_cmd_mode_WIFI(void)
```

kann der Kommandomodus wieder verlassen werden.

Man kann dann prinzipiell mit den normalen writeString / Char _WIFI Funktionen Kommandos an das Modul senden – diese müssen stets mit \r (=CR, Carriage Return) abgeschlossen werden.

Das Modul gibt sofort jedes empfangene Zeichen als Echo zurück. Dies kann benutzt werden um die korrekte Übermittlung zu gewährleisten und ist auch nützlich um zu warten bis das Kommando vollständig vom Modul empfangen wurde.

Möchte man ein Kommando an das WLAN Modul senden und das Echo direkt mit prüfen kann die Funktion

```
int8_t writeCommand_WIFI(char * cmd)
```

verwendet werden. Hier wird allerdings nur geprüft ob das Echo des Kommandos selbst zurückempfangen wird. Es wird jedes einzelne Zeichen vom WLAN Modul sofort zurückgesendet und überprüft. Somit ist sichergestellt, dass das Kommando nach verlassen der Funktion vollständig übertragen wurde, allerdings wird nicht weiter auf eine Bestätigungsnachricht oder sonstige Ausgaben gewartet. Rückgabewert ist 1 bei Erfolg und 0 bei Misserfolg.

Wenn man auf eine Antwort des WLAN Moduls warten möchte, kann man eine der beiden folgenden Funktionen verwenden. Auf ein einzelnes Zeichen (char response) kann man mit

```
int8_t waitCharResponse_WIFI(char response, uint32_t timeout)
```

warten. Auf eine komplette Zeichenkette kann mit

```
int8_t waitResponse_WIFI(char * response, uint32_t timeout)
```

gewartet werden. Bei beiden Funktionen kann ein timeout angegeben werden (ist momentan nur ein Zähler ohne festen Zeitbezug um die Schleifendurchläufe im Normalfall nicht zu verlangsamen).

Kommandos kann man auch mit

```
int8_t issueCMD_WIFI(char * cmd, char * response)
```

an das WLAN Modul senden. Hier wird einiges automatisch Mithilfe der oben genannten Funktionen erledigt. Das Kommando wird gesendet und auf die angegebene Antwort gewartet. Wenn diese ausbleibt wird bis zu 3 mal versucht das Kommando nochmals zu senden. Rückgabewert 1 bei Erfolg, 0 bei Misserfolg.

Die Funktionen sind alle blockierend, man muss sie daher mit Bedacht einsetzen! Schlägt ein kommando fehl (z.B. weil das WLAN Modul gerade mit anderem beschäftigt war) kann es zu langen Verzögerungen kommen.

Bei komplexeren Programmen könnten nicht blockierende Varianten erforderlich sein. Hier muss man manuell mit den normalen writeChar_WIFI und writeString_WIFI Funktion arbeiten und es ins eigene Programm integrieren.

Hinweis:

Man kann das WLAN Modul sowohl über die Netzwerk Verbindung, als auch über die serielle Schnittstelle in den Kommandomodus versetzen. Aber nur EINE der beiden Schnittstellen darf jeweils aktiv sein, nicht beide gleichzeitig.

Der RobotLoader nutzt die Netzwerk Verbindung um Befehle an das Modul zu senden (z.B. Reset für Mikrocontroller auslösen und Signalstärke abfragen).

Man sollte die eigenen Programme also so auslegen, das immer nur kurz in den Kommandomodus gewechselt wird und diesen so bald wie möglich wieder verlassen. Sonst kann es sein, dass man ein paar Versuche starten muss bis der RobotLoader wieder die Kontrolle über das Modul erlangt oder man sogar manuell den Reset Knopf betätigen muss.

Um die Text Antworten vom WLAN Modul leichter verarbeiten zu können kann man mit folgender Funktion einzelne Zeilen erkennen und auswerten:

```
uint8_t parseLine_WIFI(uint8_t data)
```

Dazu muss Sie in einer Schleife aufgerufen und jedes einzelne Zeichen übergeben werden. Die Rückgabewerte der Funktion geben an ob eine komplette Zeile eingelesen wurde – diese liegt dann in einem Buffer (=Array) und kann weiterverarbeitet werden.

Der Rückgabewert ist:
0 wenn keine neue Zeile erkannt wurde,
1 wenn ein Trennzeichen erkannt wurde und
2 wenn der Buffer zu klein war (alles was reingepasst hat kann dennoch ausgelesen werden). Standardmäßig sind es 254 Zeichen.

Der Buffer ist als

```
char receiveBuffer_WIFI[256]
```

definiert und kann ganz normal im eigenen Programm verwendet werden.

Folgendes einfaches Beispiel gibt alle vom WLAN Modul empfangen Textzeilen wieder auf der seriellen Schnittstelle aus:

```
while(true)
{
	if(getBufferLength_WIFI()) // Irgendwelche Daten empfangen?
	{
		if(parseLine_WIFI(readChar_WIFI())) // Ganze Zeile?
		{
			writeString("\nHabe folgenden Text empfangen: ");
			writeString(receiveBuffer_WIFI);
		}
	}
}
```

Baut man nun eine Verbindung zum Modul auf, kann man im Netzwerk Terminal Text eintippen und senden. Dieser sollte dann im seriellen Terminal erscheinen. Natürlich muss auch die serielle Schnittstelle zunächst geöffnet werden.

Hinweis:

Diese wenigen einfach zu verwendenden Funktionen reichen schon aus um mit dem WLAN Modul arbeiten zu können.

Prinzipiell erfolgt die Ansteuerung aller Sonderfunktionen über Text Kommandos und entsprechende Reaktionen des WLAN Moduls. Dafür reichen die gegebenen Funktionen also bereits aus und deswegen wurde bislang auch nichts weiteres in die zentrale Library aufgenommen. Die RP6M256Lib konzentriert sich hier auf die grundlegende Kommunikation, alles andere ist Anwendungsspezifisch. Einiges wird in den Beispielprogrammen demonstriert, aber natürlich bei weitem nicht alle möglichen Funktionen.

Kleines Beispiel. IP Adresse, Netzmaske und Gateway ändern geht im Programmcode folgendermaßen:

```
enter_cmd_mode_WIFI();
issueCMD_WIFI("set ip address 192.168.10.180","AOK");
issueCMD_WIFI("set ip netmask 255.255.255.0","AOK");
issueCMD_WIFI("set ip gateway 192.168.10.1","AOK");
leave_cmd_mode_WIFI();
```

Hier wird stets auf „AOK" gewartet was die Bestätigungsnachricht des WLAN Moduls ist. Ähnlich funktioniert es auch mit den meisten anderen Befehlen.

Die Grundlegende Konfiguration wird allerdings schon vom RobotLoader erledigt und muss nicht im eigenen Programm vorgenommen werden. Das spart Platz und Zeit. Zudem können die Einstellungen geändert werden, ohne das Programm nochmals neu compilieren zu müssen.

5. Beispielprogramme

Auf der CD / Webseite finden Sie einige ausführlich kommentierte Beispielprogramme. Diese Beispielprogramme demonstrieren die grundlegenden Funktionen des RP6 CONTROL M256 WIFI. Genau wie schon beim Roboter stellen sie keinesfalls die optimale Lösung dar und verstehen sich als Ausgangspunkt für eigene Programme. Das ist absichtlich so, damit Ihnen auch noch etwas zu tun bleibt – wäre ja langweilig einfach nur vorgefertige Programme auszuprobieren...

Es werden in den Beispielprogrammen bei weitem nicht alle Möglichkeiten des Moduls genutzt! Daher gibt es noch sehr viel Spielraum für eigene Erweiterungen und deutlich komplexere Programme. Das größte Programm, der Selbsttest, benötigt weniger als 45KByte Speicher (ein Großteil davon belegt die SDCard Library). Abzüglich des Bootloaders bleiben also noch gut 200KByte Programmspeicher übrig was für sehr komplexe Programme und auch für kleine Realtime Betriebssysteme und einfache Webserver ausreicht. Größere Datenmengen können zudem auf die SD Karte ausgelagert werden.

Sie können Ihre eigenen Programme selbstverständlich mit anderen Anwendern über das Internet austauschen. Die RP6M256Lib und alle Beispielprogramme stehen unter der Open Source Lizenz „GPL" (General Public License) und daher sind Sie berechtigt, die Programme unter den Bedingungen der GPL zu modifizieren, zu veröffentlichen und anderen Anwendern zur Verfügung zu stellen.

Allgemein gibt es für die ATMEGA Mikrocontroller Familie schon sehr viele Bespielprogramme im Internet, da diese Controller bei Hobby Anwendern sehr beliebt sind. Allerdings muss man hier immer darauf achten, andere Beispielprogramme auch an die Hardware des RP6 CONTROL M256 und die RP6M256Lib anzupassen – sonst wird es oft nicht funktionieren (die offensichtlichsten Probleme sind andere Pinbelegungen, Verwendung von bereits anderweitig verwendeten Hardwaremodulen wie Timern, andere Taktfrequenz, etc. pp.)!

Die Beispiele sind in Englisch kommentiert, damit nicht zwei Sprachversionen gepflegt werden müssen.

Im Folgenden wird die Funktion der Beispielprogramme kurz auch auf Deutsch erläutert.

Beispiel 1: „Hello World"-Programm mit LCD Textausgabe und LED Lauflicht
Verzeichnis: <RP6Examples>\RP6_M256_WIFI_EXAMPLES\Example_01_LCD
Datei: RP6M256_LCD.c

Das Programm erzeugt Ausgaben auf der seriellen Schnittstelle

UND sendet Ausgaben über die WLAN Verbindung!

Sie sollten den Roboter also an den PC anschließen und sich die Ausgaben im Terminal der RobotLoader Software ansehen! Optional können Sie auch das LC-Display anschließen!

Der Roboter bewegt sich in diesem Beispielprogramm *nicht – sofern Sie nur das I²C-Bus Slave Programm in den Controller auf dem Mainboard geladen haben*! Sie können ihn also z.B. auf einen Tisch neben dem Computer stellen.

Dieses Beispielprogramm gibt einen kurzen „Hello World" Text über die serielle Schnittstelle und über das WLAN Modul / die offene Netzwerk Verbindung aus. Sie können sich die Ausgabe also in beiden Terminals anschauen!

Anschließend wird ein Lauflicht ausgeführt. Zusätzlich wird auf dem LCD (sofern vorhanden) zunächst ein statischer Text ausgegeben und später ein sich bewegender „Hello World" Text – die beiden Wörter „HELLO" und „WORLD" bewegen sich langsam hin und her. Nach etwa 16 Sekunden wird eine kurze Pause gemacht und dies auch über die Netzwerkverbindung ausgegeben. Nach 8 Sekunden wird weitergemacht.

Beispiel 2: Taster und LC Display
Verzeichnis: <RP6Examples>\RP6_M256_WIFI_EXAMPLES\Example_02_Buttons
Datei: RP6M256_Buttons.c

Das Programm erzeugt Ausgaben auf der seriellen Schnittstelle, der WLAN Verbindung und auf dem LC-Display!

Der Roboter bewegt sich in diesem Beispielprogramm *nicht!*

Dieses Beispielprogramm demonstriert die Verwendung der 2 Taster auf dem RP6 CONTROL M256. Bei jedem Tastendruck wird die Tastennummer im Display angezeigt und über die serielle Schnittstelle und der WLAN Verbindung ausgegeben.

Beispiel 3: Analog Digital Wandler und I/O Ports
Verzeichnis: <RP6Examples>\RP6_M256_WIFI_EXAMPLES\Example_03_ADC_IO1
Datei: RP6M256_IO_ADC.c

Das Programm erzeugt Ausgaben auf der seriellen Schnittstelle, der WLAN Verbindung und auf dem LC-Display!

Der Roboter bewegt sich in diesem Beispielprogramm *nicht!*

Es funktioniert zwar eigentlich genau wie auf dem Roboter – aber weil es vor allem bei diesem Erweiterungsmodul sehr häufig benötigt wird, demonstriert dieses Beispielprogramm wie die freien ADCs und I/Os ausgewertet werden können. Das Programm zeigt auch wie einfach man periodisch Telemetriedaten an einen Host Rechner senden kann.

Alle 100ms wird eine sehr lange Textnachricht per WLAN übertragen, die absichtlich völlig unnötig vergrößert wurde. Dies soll demonstrieren, dass auch größere Nachrichten übertragen werden können. Die Bandbreite kann natürlich deutlich effizienter ge-

nutzt werden, z.B. wenn ein Programm ständig alle Sensordaten empfangen und anzeigen soll.

Es werden in der Textnachricht alle I/O Port Zustände, alle Messwerte der ADC Kanäle und alle Stopwatch Zählerstände übertragen. Zusätzlich noch ein paar mal ein 16 Bit Zähler in Dezimal und Binär (d.h. es werden 16 ASCII Zeichen ausgegeben – obwohl zwei Bytes schon reichen würden – oder alternativ 4 Zeichen bei hexadezimaler Darstellung. Wie gesagt, nicht gerade effizient, nur zum vergrößern der Nachricht).

Beispiel 4: microSD Karte
Verzeichnis: <RP6Examples>\RP6_M256_WIFI_EXAMPLES\Example_04_SDCARD
Datei: RP6M256_04_SDCARD.c

Das Programm erzeugt Ausgaben auf der seriellen Schnittstelle, der WLAN Verbindung und auf dem LC-Display!

Eine passende microSD Karte sollte im Kartenslot stecken!

Der Roboter bewegt sich in diesem Beispielprogramm *nicht!*

Dieses Programm gibt den Inhalt der Test Textdatei aus, die auf der microSD Karte im Hauptverzeichnis für das Selbsttest Program abgelegt wurde. Dabei wird geprüft ob ein gewisser Textteil korrekt ausgelesen wird.

Sie müssen die Datei M256_SELFTEST_TESTFILE.txt in dem Beispielverzeichnis auf eine geeignete microSD Karte kopieren. Die microSD Karte kann dann in den Kartenslot des RP6-M256 gesteckt werden, dabei muss der Roboter ausgeschaltet sein! Dann einschalten und das Programm hochladen+starten. Es werden nun einige Informationen über die SD Karte abgerufen und anschließend der Inhalt der Textdatei auf serieller Schnittstelle und per WLAN ausgegeben.

Beispiel 5: WLAN Kommandos
Verzeichnis: <RP6Examples>\RP6_M256_WIFI_EXAMPLES\Example_05_WLAN_CMD
Datei: RP6M256_WLAN_CMD.c

Das Programm erzeugt Ausgaben auf der seriellen Schnittstelle, der WLAN Verbindung und auf dem LC-Display!

Der Roboter bewegt sich in diesem Beispielprogramm *nicht!*

Für einige Anwendungen kann es notwendig sein, dem WLAN Modul Befehle zu senden bzw. einige Statusinformationen abzufragen wie beispielsweise die aktuelle Signalstärke.

Es ist auch möglich verschiedene in der Modul Firmware enthaltene Client Anwendungen für diverse Protokolle zu verwenden (HTTP Client, FTP Client...) - dies wird hier nicht im Detail beschrieben. In der Dokumentation zum WLAN Modul sind weitere Informationen dazu zu finden.

In diesem Beispielprogramm wird regelmäßig ein Scan nach Accesspoints ausgelöst und die Ergebnisse ausgegeben. Die wird sowohl über die serielle als auch über die WLAN Verbindung ausgegeben.

Beispiel 6: I²C Bus Interface – Master Modus
Verzeichnis: <RP6Examples>\RP6_M256_WIFI_EXAMPLES\Example_06_I2CMaster
Datei: RP6M256_06_I2CMaster.c

Dieses Programm demonstriert wie man den Master Modus des I²C Busses verwenden kann. Der Controller auf dem Roboter Mainboard muss das I2C Slave Beispielprogramm (RP6Base_I2CSlave.hex) geladen haben!

Dieses Programm zeigt, wie man den Controller auf dem Mainboard im Slave Modus steuern kann. Das klappt natürlich nur dann, wenn auch das I²C Slave Beispielprogramm aus den RP6Base Beispielen in den Controller auf dem Mainboard geladen ist.

Der Zugriff auf den I²C Bus funktioniert fast genau wie auf dem Controller auf dem Mainboard auch – es sind dieselben Funktionen.

Es werden hier alle Register die das Slave Beispiel zur Verfügung stellt ausgelesen und per WLAN übertragen – nur per WLAN, nicht über die serielle Schnittstelle! Zusätzlich wird noch ein Zählerstand mit den LEDs auf dem Mainboard angezeigt.

Beispiel 7: I²C Bus Interface – Master Modus - auf Interrupts reagieren
Verzeichnis: <RP6Examples>\RP6_M256_WIFI_EXAMPLES\Example_07_I2CMaster
Datei: RP6M256_07_I2CMaster.c

Dieses Programm demonstriert wie man den Master Modus des I²C Busses verwenden kann. Der Controller auf dem Roboter Mainboard muss das I2C Slave Beispielprogramm geladen haben!

Vielleicht sind Ihnen schon die Interrupt Signale auf den XBUS Anschlüssen des RP6 aufgefallen? Diese kann man nutzen um ohne ständiges Abfragen der Slaves auf Sensoränderungen zu reagieren. Jeder Buszugriff kostet schließlich Zeit.

Ein gutes Beispiel dafür ist das ACS auf dem Roboter. Der Sensorzustand ändert sich hier nur *relativ* selten und es wäre nicht besonders effektiv, ständig über den Bus abzufragen ob sich etwas geändert hat. Sobald sich der Status des ACS ändert, wird vom Slave Beispielprogramm das INT1 Signal auf High Pegel gesetzt. Da INT1 an einen der Interrupt Eingänge vom MEGA2560 auf dem RP6 CONTROL M256 gelegt wurde, kann der Controller direkt auf dieses Ereignis reagieren und den Status des Controllers auf dem Mainboard abfragen.

Wir nutzen in den Beispielprogrammen allerdings KEINE Interrupt Routinen um auf das Ereignis zu reagieren, sondern fragen den Zustand des Pins bei jedem Durchlauf der Hauptschleife ab. Da die I2C Bus Transfers selbst auch Interruptgesteuert ablaufen, könnte innerhalb der Interrupt Routine keine neue Übertragung stattfinden. Esmuss in der Hauptschleife immer die task_I2CTWI() Funktion aufgerufen werden, die den Ablauf der I2C Transfers regelt. Daher hätte es keinen großen Vorteil eine Interrupt Routine zu verwenden. Um genau zu sein, würde es sogar Probleme bereiten, denn so könnten bereits laufende I2C Bus Übertragungen unterbrochen werden. Also wird nur die Funktion task_checkINT() verwendet um ständig das Interrupt Signal auszuwerten und ggf. eine Abfrage zu starten. Sobald das Statusregister 0 des Slaves gelesen wird, wird das Interrupt Signal zurückgesetzt. Über die ersten drei Register des Slaves kann man herausfinden, was den Interrupt ausgelöst hat.

Genau das macht dieses Beispielprogramm auch. Resultat ist, dass der aktuelle Zustand des ACS über die 4 LEDs, das LCD und der WLAN Verbindung ausgegeben wird.

Zu Beginn des Programms wird noch die Sendeleistung des ACS über den I²C Bus eingestellt. Neben dem ACS reagiert das Programm auch auf die Bumper und auf eventuell gesendete RC5 Übertragungen von einer Fernbedienung oder anderen Robotern.

Das kann man so ähnlich natürlich auch mit allen anderen Sensoren des Roboters handhaben.

Ein weiteres Detail des Programms ist die „Heartbeat" Anzeige. Also „Herzschlag" Anzeige. Die task_LCDHeartbeat() Funktion sorgt dafür, dass auf dem LCD ständig das Zeichen '*' mit einer Frequenz von 1Hz blinkt. Das ist sehr hilfreich um festzustellen ob sich das Programm komplett aufgehangen hat, oder ob nur ein bestimmter Teil der Software fehlerhaft ist. Wenn Sie eigene Programme schreiben und sich das Programm dann anscheinend komplett aufhängt, kann das sehr hilfreich sein um die Fehlerquelle einzugrenzen. Das sich das Programm aufhängt, kann während der Entwicklung durchaus mal passieren.

Beispiel 8: I²C Bus Interface – RP6 Master Library
Verzeichnis: <RP6Examples>\RP6_M256_WIFI_EXAMPLES\Example_08_I2CMaster
Datei: RP6M256_08_I2CMaster.c

Dieses Programm demonstriert wie man den Master Modus des I²C Busses verwenden kann. Der Controller auf dem Roboter Mainboard muss das I2C Slave Beispielprogramm geladen haben!

Da Programme schnell etwas unübersichtlich werden können, wenn man soviel in eine Datei reinpackt, wird jetzt die fertige RP6 Master Library verwendet. Diese wurde weitgehend vom RP6-M32 Modul übernommen, in den dortigen Beispielprogrammen ist der Aufbau etwas genauer beschrieben. Beim Entwurf wurde es gleich so angelegt, dass man eine kleine Bibliothek zur Steuerung des Roboters über den I²C Bus hat, die sich fast wie die normale RP6Lib für den Controller auf dem Mainboard verwenden lässt. Viele Funktionen und Variablen sind identisch zur RP6Lib benannt. Das vereinfacht es, Teile von Programmen für die RP6Lib auch mit der RP6M256Lib weiterzuverwenden. Auch die bekannten Event Handler für ACS, IRCOMM und Bumper sind wieder verfügbar.

Im I²C Slave Beispielprogramm für den Controller auf dem Mainboard ist auch eine „Software Watchdog" Funktion eingebaut - was ähnlich zum Heartbeat funktioniert aber über den I²C Bus. Reagiert der Master innerhalb einer festgelegten Zeit nicht auf ein Interrupt Ereignis (indem Register 0 gelesen wird), werden alle Systeme der Roboter Basiseinheit ausgeschaltet und das Programm gestoppt. Vor allem werden die Motoren ausgeschaltet! Denn sollte sich der Master Controller aufhängen, aber vorher noch den Befehl mit 10cm/s vorwärts zu fahren senden, fährt der Roboter ungebremst vor das nächste Hindernis und stoppt auch nicht bei einer Kollision...

Der Software Watchdog Timer ist zunächst deaktiviert. Man muss vorher noch einen Befehl über den I²C Bus senden um den Watchdog zu aktivieren. Es ist auch möglich den Watchdog so zu konfigurieren, dass dieser alle 500ms ein Interrupt Ereignis auslöst um zu überprüfen ob der Mastercontroller noch darauf reagiert. Das werden wir im nächsten Beispiel verwenden.

Es gibt auch Event Handler für Watchdog Requests und zusätzlich für niedrigen Akkuladezustand.

Das Programm ist ansonsten ähnlich zu Beispiel 7. Neu hinzugekommen sind nur wie schon erwähnt der weitere Watchdog Timer dessen Requests auch auf dem LC-Display dargestellt werden und es werden alle Sensorregister ausgelesen und über die serielle Schnittstelle ausgegeben. Wie gehabt werden auch weiterhin ACS, Bumper und RC5 Events dargestellt.

Beispiel 9: I²C Bus Interface – Bewegungsfunktionen
Verzeichnis: <RP6Examples>\RP6_M256_WIFI_EXAMPLES\Example_09_Move
Datei: RP6M256_09_Move.c

Dieses Programm demonstriert wie man den Master Modus des I²C Busses verwenden kann. Der Controller auf dem Roboter Mainboard muss das I²C Slave Beispielprogramm geladen haben!

ACHTUNG: Der Roboter bewegt sich in diesem Beispielprogramm!

Jetzt testen wir einige der bereits aus der RP6Lib bekannten Bewegungsfunktionen, die nun über den I2C Bus quasi ferngesteuert werden: move, rotate, moveAtSpeed, changeDirection und stop. Die Funktionen können identisch zu denen in der RP6Lib benutzt werden. In diesem Beispiel haben wir alles aus den vorherigen Beispielen bis auf die Anzeige für den Watchdog wieder entfernt, damit es übersichtlicher ist und ausserdem der blockierende Modus der Bewegungsfunktionen verwendet wird (dann würden ein paar Sachen wie die Heartbeat Anzeige ohnehin nicht funktionieren). Der Roboter bewegt sich in diesem Beispiel hin und her und dreht sich dabei um etwa 180° - genau wie in Beispiel 7 zur RP6Lib („RP6Base_Move_02.c").

Beispiel 10: I²C Bus Interface – Verhaltensbasierter Roboter
Verzeichnis: <RP6Examples>\RP6_M256_WIFI_EXAMPLES\Example_10_Move
Datei: RP6M256_10_Move.c

Dieses Programm demonstriert wie man den Master Modus des I²C Busses verwenden kann. Der Controller auf dem Roboter Mainboard muss das I2C Slave Beispielprogramm geladen haben!

ACHTUNG: Der Roboter bewegt sich in diesem Beispielprogramm!

Mit der neuen Library kann man fast 1:1 die Beispielprogramme zum Verhaltensbasierten Roboter von der RP6Base übernehmen. Genau das wurde hier mit Beispielprogramm RP6Base_05_Move_05 getan. Es waren nur kleinere Änderungen notwendig, u.a. müssen die LEDs auf dem Mainboard über die Funktion setRP6LEDs gesteuert werden, da setLEDs schon für die LEDs auf dem RP6-M256 reserviert ist...

Ansonsten ist das Programm nahezu identisch zum bereits bekannten Programm – der Roboter fährt herum und weicht dabei Hindernissen aus. Nur das er diesmal vom RP6-M256 Modul gesteuert wird.

Neu hinzugekommen ist die Darstellung des aktuell aktiven Verhaltens auf dem LC-Display und mit den LEDs. So kann man direkt sehen welches Verhalten gerade aktiv ist. Dazu gibt es eine kleine Hilfsfunktion die sicherstellt, dass der Text immer nur einmal an das LC-Display gesendet wird. Sonst würde der Text auf dem Display flim-

mern. Während das Verhalten „Cruise" aktiv ist, wird mit den 4 roten Status LEDs ein kleines Lauflicht ausgeführt.

Es wird auch der Akkuzustand überwacht. Wenn der Ladezustand sehr niedrig ist, wird der Roboter angehalten. Das dauert bei frisch aufgeladenen Akkus allerdings eine Weile...

Beispiel 11: WLAN Fernsteuerung 1
Verzeichnis: <RP6Examples>\RP6_M256_WIFI_EXAMPLES\Example_11_WIFI_REMOTE_1
Datei: RP6M256_11_WIFI_REMOTE_1.c

Dieses Programm demonstriert wie man den Master Modus des I²C Busses verwenden kann. Der Controller auf dem Roboter Mainboard muss das I2C Slave Beispielprogramm geladen haben!

Zusätzlich wird hier über die WLAN Verbindung der Roboter ferngesteuert (Textbefehle).

ACHTUNG: Der Roboter bewegt sich in diesem Beispielprogramm!

Die WLAN Verbindung bietet sich natürlich an, um den Roboter darüber fernzusteuern. Dies wird in diesem Programm demonstriert.

Es ist ein simpler Kommandointerpreter, der auf bestimmte Textkommandos reagieren kann. Der Kommandointerpreter wird hier allerdings gleich in das Verhaltensbasierte Modell aus den anderen Beispielen eingebaut damit das Programm später leicht um autonome Verhalten ergänzt werden kann.

In diesem Beispiel ist behaviour_wifiControl das einzige aktive Verhalten.

Die kleine Bedienungsanleitung des Programms wird direkt am Anfang per WLAN ausgegeben. Man tippt zunächst „cmd" ein, um den Interpreter zu aktivieren. Aktivieren heisst hier die Steuerung des Roboters aktivieren bzw. das Verhalten aktiv zu schalten. Der Interpreter wartet immer auf Kommandos. Ist er „aktiv" kann man den Roboter darüber steuern.

Vorbelegt sind die Tasten w-a-s-d und q, e zur Richtungssteuerung. Für jedes Kommando muss man die Enter Taste drücken. Also z.B.:

```
60 + Enter  … der Roboter fährt vorwärts

d + Enter … der Roboter rotiert nach rechts, der Kommandointerpreter reduziert automatisch die Geschwindigkeit beim Rotieren da dies sonst im Vergleich zur Vorwärtsbewegung sehr schnell sein würde.

s + Enter … Rückwärts.

120 + Enter … schneller

w + Enter … wieder vorwärts

q + Enter … Kurve nach links

Enter … Stop!   (alternativ 0 + Enter)

z + Enter … Interpreter beenden
```

Während der Interpreter läuft, werden zahlreiche Sensorwerte per WLAN übertragen. In diesem Fall alle 100ms. So etwas könnte man natürlich nutzen um es in einer graphischen Oberfläche darzustellen. Auch die Textbefehle könnte man hinter Schaltflächen in einer graphischen Benutzeroberfläche verpacken oder auch z.B. einen Joystick oder ein Gamepad zur Steuerung verwenden.

Beispiel 12: WLAN Fernsteuerung 2
Verzeichnis: <RP6Examples>\RP6_M256_WIFI_EXAMPLES\Example_12_WIFI_REMOTE_2
Datei: RP6M256_12_WIFI_REMOTE_2.c

Dieses Programm demonstriert wie man den Master Modus des I²C Busses verwenden kann. Der Controller auf dem Roboter Mainboard muss das I2C Slave Beispielprogramm geladen haben!

Zusätzlich wird hier über die WLAN Verbindung der Roboter ferngesteuert (Textbefehle).

ACHTUNG: Der Roboter bewegt sich in diesem Beispielprogramm!

Hier wird das autonome Verhalten der anderen Programme wieder hinzugefügt – man kann nun die Verhaltensweisen über die WLAN Verbindung aktivieren und deaktivieren. Und auch weiterhin den Roboter manuell steuern.

Stets aktiv ist das Escape Verhalten, denn sollte der Roboter mit einem Hindernis kollidieren ist es immer besser wenn er zurücksetzt und ausweicht anstatt weiterzufahren. Allerdings ist der Kommandomodus noch immer höher priorisiert. Das heisst man kann den Roboter absichtlich z.B. eine Kiste schieben lassen, ohne dass das Escape Verhalten aktiv wird.

Avoid und Cruise kann man aktivieren und deaktivieren. Hier sollte man Cruise als letztes aktivieren, denn dann fährt der Roboter einfach drauflos ohne auf Hindernisse zu achten. Ist Cruise deaktiviert, reagiert der Roboter aber trotzdem auf Hindernisse (mal die Hand vor dem Roboter bewegen oder auf die Bumper drücken).

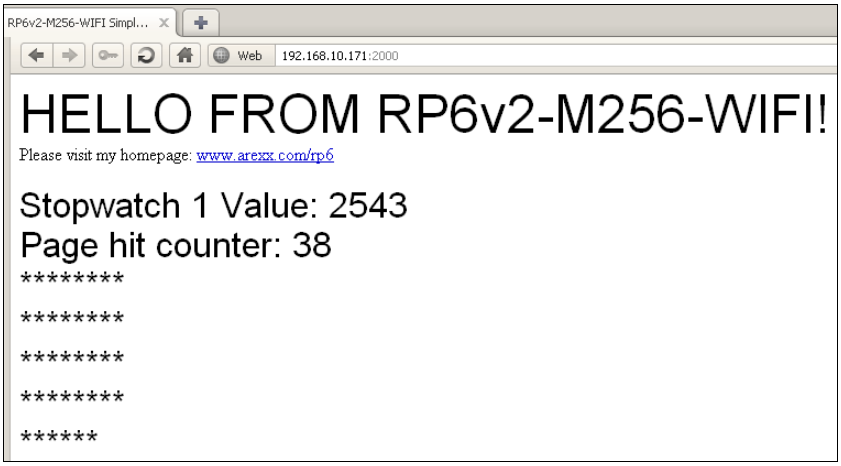
Sobald irgendein Verhalten ausser dem „Stop" Verhalten aktiv ist, werden alle Sensorwerte des Roboters mit 10Hz per WLAN übertragen.

Beispiel 13: Sehr einfacher Webserver
Verzeichnis: <RP6Examples>\RP6_M256_WIFI_EXAMPLES\Example_13_Simple_Webserver
Datei: RP6M256_13_Simple_Webserver.c

Das Programm erzeugt Ausgaben auf der seriellen Schnittstelle, der WLAN Verbindung und auf dem LC-Display!

Der Roboter bewegt sich in diesem Beispielprogramm *nicht!*

Die Netzwerkdatenverbindung kann man auch für Anwendungen wie Webserver verwenden. Dieses Beispiel demonstriert dies auf sehr einfache Weise. Es wird hier ein extrem minimalistischer Webserver implementiert. Der Code für den Server ist effektiv nur etwa 25 Zeilen kurz (ohne Inhalt und Debug Ausgaben mitzurechnen).



Der Webserver kann nur eine einzige Sache, nämlich die auf dem Screenshot zu sehende Webseite an den anfragenden Browser senden.

Die Seite ist allerdings immerhin dynamisch, d.h. Der Inhalt kann vom Programm verändert werden. Es wird hier einfach der Wert einer der Stopwatches angezeigt (wird nach 10 Sekunden wieder auf 0 gesetzt) und bei jedem Seitenaufruf ein Zähler hochgezählt und dazu passend viele * ausgegeben. Es ist nur eine einzige Verbindung gleichzeitig möglich. Daher darf man die Seite auch nicht zu schnell aktualisieren oder mit zwei Browsern gleichzeitig darauf zugreifen. Der Webserver reagiert immer mit der gleichen Antwort auf die GET Anfrage. Er kann keine weiteren Dateien wie Bilder o.ä. nachliefern. Dazu müsste man schon etwas mehr Aufwand treiben, es ist aber durchaus möglich. Bei großen Daten wie Bildern muss man allerdings die begrenzte Übertragungsgeschwindigkeit beachten.

Das Beispiel kann prinzipiell auch um Eingabemöglichkeiten über Formulare erweitert werden, wodurch man den Roboter dann per Webbrowser steuern könnte.

Beispiel 14: Datenlogging auf SD Karte + Webserver
Verzeichnis: <RP6Examples>\RP6_M256_WIFI_EXAMPLES\Example_14_SDCARD_logging
Datei: RP6M256_14_SDCARD_logging.c

Das Programm erzeugt Ausgaben auf der seriellen Schnittstelle, der WLAN Verbindung und auf dem LC-Display!

Der Roboter bewegt sich in diesem Beispielprogramm *nicht!*

Nun wird die SD Karte verwendet um Daten aufzuzeichnen. Es werden 1x pro Sekunde alle Sensorwerte vom Roboter abgerufen und in eine Textdatei auf der Karte geschrieben. Dazu werden u.a. ein paar Standard C Funktionen für die Verarbeitung von Zeichenketten verwendet.

Zusätzlich wurde der kleine Webserver auf dem vorherigen Beispiel mit ins Programm integriert um die aufgezeichneten Daten per Webbrowser abrufen zu können.

Es sind die bereits weiter oben genannten Sicherheitshinweise zur microSD Karte zu beachten! Die Schreiboperationen müssen durch Druck auf den Taster SW1 beendet werden (Lauflicht stoppt und auf dem LCD wird eine entsprechende Meldung ausgegeben) bevor der Roboter ausgeschaltet oder ein Reset (auch per WLAN Verbindung) ausgelöst werden darf.

In Zukunft wird dies ggf. noch etwas komfortabler gelöst und per Webserver und/oder den RobotLoader einstellbar sein.

Damit sind wir am Ende dieser Zusatzanleitung angelangt. Die Beispielprogramme sind nur als Grundlage / Ausgangspunkt gedacht. Jetzt können Sie Ihre eigene Kreativität walten lassen, neue Programme schreiben und neue Sensoren am RP6 anbringen, die Sie mit dem RP6-M256 steuern können - oder etwas ganz anderes damit anstellen.

ANHANG

A – Anschlussbelegungen

In diesem Abschnitt finden Sie die Anschlussbelegungen der wichtigsten Stecker mit den I/O Ports.

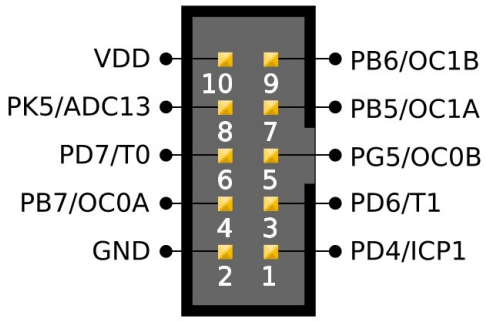
Der Anschluss der seriellen Schnittstelle hat genau die gleiche Pinbelegung wie auf dem Mainboard. Das gilt natürlich auch für die beiden XBUS Anschlüsse!

I/O Ports:

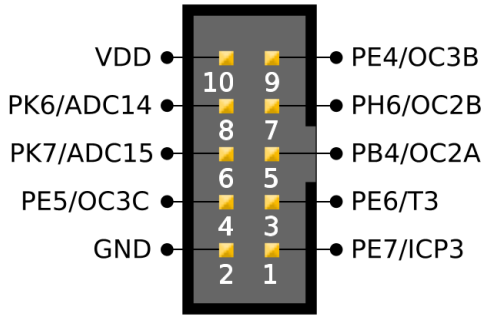
Alle 10 poligen I/O Port Anschlüsse (und der 10 polige ADC Port, s. nächste Seite) sind grob pinkompatibel ausgelegt. Das bedeutet die 5V Versorgungsspannung und I/Os liegen auf den selben Pins, allerdings sind die Zusatzfunktionen natürlich unterschiedlich. Es wurde jedoch darauf geachtet, die alternativen Funktionen möglichst ähnlich zu verteilen. So sind z.B. die beiden SPI / UART Ports auf den gleichen Pins. Die meisten PWM und Timer/Capture Kanäle sind ebenso auf den gleichen Pins. Allerdings durchaus auch von zwei unterschiedlichen Timern (Hinweis: OC0B und OC2A/B sind 8 Bit Timer, die anderen 16 Bit. OC0A ist mit OC1C doppelt belegt).

Somit kann man Beispielsweise mehrere pinkompatible Sensorik- und Aktorikplatinen erstellen.

IO_PWM/T0/T1



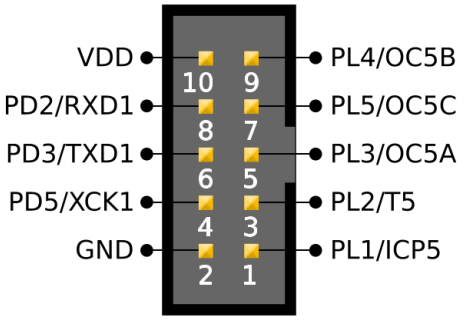
IO_PWM/T2/T3



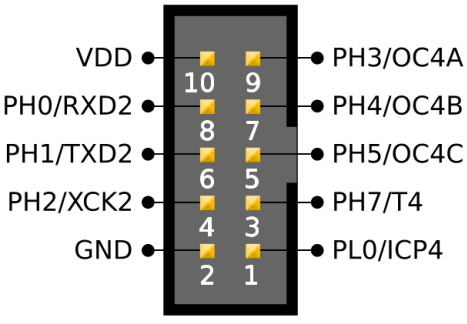
ACHTUNG: Verbinden Sie lieber nicht die Betriebsspannungspins eines anderen Erweiterungsmoduls (z.B. eines Experimentiermoduls) mit den Betriebsspannungspins an diesen Steckern.

Die Versorgungspins sind nur eingeschränkt für den direkten Betrieb von größeren Verbrauchern geeignet! Kleine Servos sind mit zusätzlichen Kondensatoren möglich. Mehrere größere Servos bitte direkt vom RP6 Mainboard, aus separatem Akku oder mind. mit eigenem 5V Spannungsregler versorgen.

UART_SPI1/T5



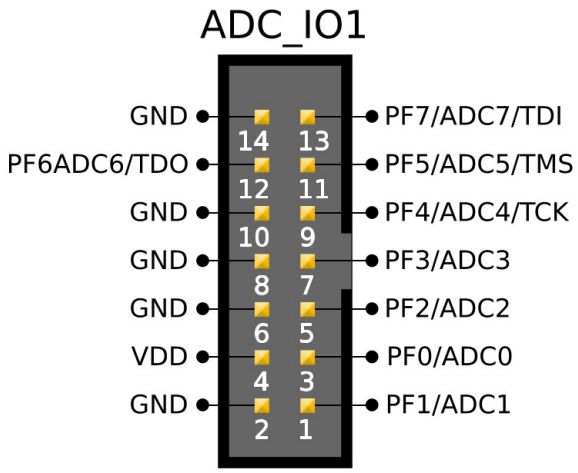
UART_SPI2/T4

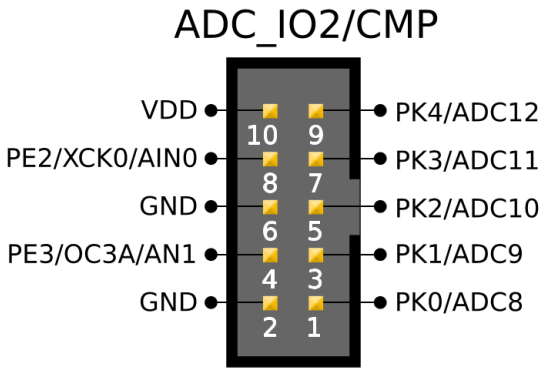


ADC Kanäle:

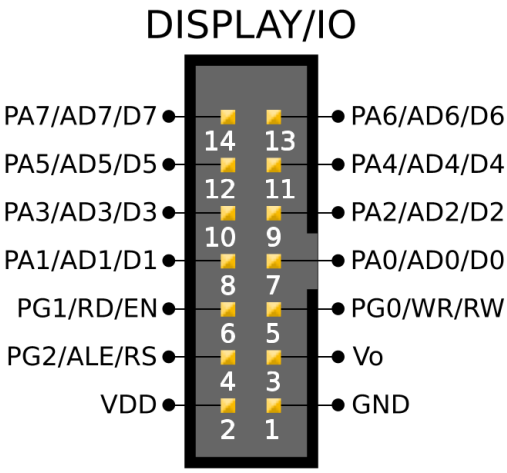
Von den 16 ADC Kanälen (die natürlich auch als I/O Pins verwendbar sind) sind 8 auf einem 14 poligen und 5 auf einem 10 poligen Stecker verfügbar. Ebenfalls zusammen mit der 5V Betriebsspannung. Die restlichen 3 Kanäle sind auf andere 10 polige Steckverbinder verteilt (s. vorherige Seite).

Auf dem 10 poligen Steckverbinder sind zusätzlich zwei normale I/O Ports mit Analog Comparator und OC3A PWM Kanal alternativ Funktionen verfügbar. Auf dem 14 poligen Steckverbinder ist auch das JTAG Interface verfügbar (abgesehen vom Reset Signal, s. ISP Stecker).





LCD Anschluss:



Wenn Sie nicht das Standard LCD einsetzen möchten, können Sie anhand der nebenstehenden Anschlussbelegung ein eigenes Kabel für das LCD zusammenbauen.

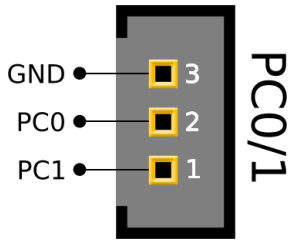
Anders als beim RP6-M32 und RP6-M128 Modul kann das Display hier im 8 Bit Modus angesteuert werden.

Alternativ lassen sich die Pins auch als normale I/O Ports verwenden. Zusätzlich ist hier ein Teil des XMEM Speicherinterfaces herausgeführt was sich ggf. mit zusätzlichen I/Os für Grafik Displays und andere spezielle Anwendungen verwenden lassen könnte (nicht getestet).

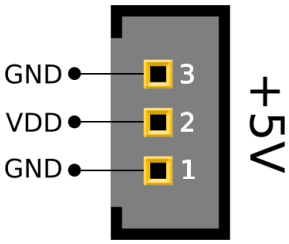
Achten Sie unbedingt auf korrekte Anschlussbelegung und darauf den Stecker nicht spiegelverkehrt anzuschließen!

Die Bezeichnungen der einzelnen Pins können je nach Display Hersteller anders sein (die Bezeichnung nach dem letzten / ist hier gemeint. Die Bezeichnungen davor sind die Portnamen des ATMEGA), aber normalerweise sind die Bezeichnungen mit denen hier verwendeten identisch und Sie können die Pins 1:1 mit dem Display verbinden!

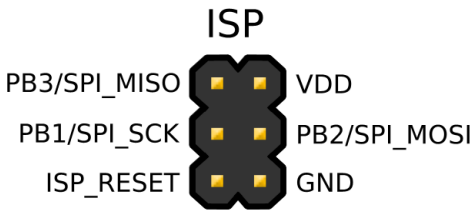
Weitere Steckverbinder:



PC0/A8 und PC1/A9 sind auf einer kleinen 3 poligen Stiftleiste verfügbar. Die anderen Adresspins des XMEM Interfaces sind anderweitig verwendet.



Auf einer weiteren 3 poligen Stiftleiste ist die 5V Versorgungsspannung verfügbar um weitere externe Verbraucher anzuschließen, z.B. für Displays mit anderer Hintergrundbeleuchtung (ACHTUNG: ggf. Vorwiderstand oder Stromquelle erforderlich!). Wenn das RP6-M256 Modul ohne den RP6 verwendet wird, kann hier auch das gesamte Modul an eine externe Versorgungsspannung angesschlossen werden ohne dazu den XBUS Steckverbinder benutzen zu müssen.



Der 6 polige ISP Anschluss (In System Programming) ist für fortgeschrittene Anwender gedacht.

Für Anfänger gilt: NICHT VERWENDEN (jedenfalls nicht ohne guten Grund)!

Er ist parallel zum microSD Karten Slot geschaltet. Die I/O Ports (und damit auch ISP Adapter) sollten daher nur verwenden werden wenn keine SD Karte eingesetzt ist. Allerdings sind dennoch 3.3V Levelshifter an den Pins (über Vorwiderstände, s. Schaltplan). Dies sollte beachtet werden wenn eigene Schaltungen daran angeschlossen werden (welche Pins sind Eingang/Ausgang).

Auf einem weiteren 6 poligen Anschluss direkt neben dem WLAN Modul sind 4 A/D Wandler Kanäle vom WLAN Modul verfügbar. Diese haben verschiedene Anwendungsmöglichkeiten und können vom WLAN Modul unabhängig vom Mikrocontroller ausgelesen werden. Es kann darüber aus dem Standby aufgeweckt werden und die Messwerte können im Kommandomodus per Netzwerk abgefragt werden. Die A/D Kanäle sind über einen Spannungsteiler mit dem Steckverbinder verbunden um den Messbereich zu erweitern (Normal: 1.2V max, 0..400mV Messbereich, 500mV Wakeup Trigger ; Spannungsteiler Faktor: 0.12821 ; daraus ergibt sich am Steckverbinder: 9V max, 0..3.2V Messbereich, 3.9V Wakeup Trigger).

Zusätzlich kann an dem Steckverbinder die 3.3V Versorgungsspannung abgegriffen werden. Auch dieser Anschluss ist eher für fortgeschrittene Anwender gedacht. Für normale Anwendungen sollten die A/D Kanäle des Mikrocontrollers verwendet werden.

Die beiden Jumper sind auf der Platine ausreichend beschriftet. Der ISP/BOOT Jumper dient dazu den Reset Pin zwischen dem ISP Steckverbinder und der normalen RP6 Reset Leitung umzuschalten. Der ADHOC Jumper kann das WLAN Modul in die Werkseinstellungen zurückversetzen wenn er vor dem Einschalten gebrückt wird und aktiviert den ADHOC Modus (es wird keine Brücke mitgeliefert da dieser nur kurzzeitig gebrückt werden muss → z.B. einen Schraubendreher verwenden!). Am AREF Steckverbinder kann bei Bedarf eine externe Referenzspannung für den A/D Wandler angschlossen werden (der AREF Pin ist wo es auf der Platine steht, der andere ist GND).

B - Tipps zum WLAN Modul

1. WiFi Modul Passthrough Modus

Wie in Kapitel 3.4 bereits erläutert, kann man mit dem Menüpunkt

`Options → WiFi SERIAL Config → Enable Passthrough Mode`

den Passthrough Modus aktivieren. Das USB Interface muss dazu angeschlossen und der korrekte Port im Serial Loader ausgewählt sein. Danach wird alles was im seriellen Terminal eingetippt wird an das WLAN Modul weitergeleitet. In den Optionen sicherstellen das CR oder CR+LF aktiviert ist! Das WLAN Modul benötigt mindestens ein CR nach jedem Kommando, mit LF alleine funktioniert es nicht!

Man kann auch die Menüoption

`Options → WiFi SERIAL Config → Reset WIFI + Enable Passthrough`

verwenden. Dies löst einen Hardware Reset des WLAN Moduls aus und man kann die komplette Boot Ausgabe sehen. Die Ausgabe im seriellen Terminal sollte in etwa so ausschauen wenn des Modul korrekt konfiguiert wurde und sich sofort mit dem Accesspoint verbinden kann:

```
WIFI:
WiFly Ver 2.32, 02-13-2012 on RN-171
MAC Addr=00:06:66:71:e6:05
Auto-Assoc M256TST chan=11 mode=WPA2 SCAN OK
Joining M256TST now..
*READY*
Associated!
Using Static IP
Listen on 2000
```

Hat man etwas nicht korrekt konfiguriert oder kann der Accesspoint nicht erreicht werden, schaut es hingegen etwa so aus:

```
WIFI:
WiFly Ver 2.32, 02-13-2012 on RN-171
MAC Addr=00:06:66:71:e6:05
Auto-Assoc M256TST chan=8 mode=NONE FAILED
*READY*
Auto-Assoc M256TST chan=8 mode=NONE FAILED
Auto-Assoc M256TST chan=8 mode=NONE FAILED
```

Durch eintippen von:

`.$$$`

Kann man in den Kommandomodus wechseln.

Es bietet sich an hier zunächst den Befehl

```
scan
```

auszuführen. Dieser listet alle Accesspoints in Reichweite auf und zeigt einige der Konfigurationsparameter. Wird der eigene Accesspoint hier nicht aufgelistet ist er evtl. zu weit entfernt oder der Empfang durch etwas anderes gestört (Antenne prüfen!). Es kann natürlich auch an einer fehlerhaften Konfiguration des Accesspoints liegen.

2. Firmware Update und Werkseinstellungen wiederherstellen

Die Firmware auf dem WLAN Modul kann aktualisiert werden.

Dazu stellt der Hersteller des Moduls einen FTP Server bereit, von dem sich das Modul selbst die neue Software herunterladen kann. Die alte Firmware bleibt dabei im Flash Speicher erhalten und kann bei Problemen leicht wieder aktiviert werden. Ein Firmware Update ist prinzipiell sogar mit dem RobotLoader möglich, dieser hat für die Test Routine einen kleinen FTP Server integriert – der sich aber nur kurz während des Tests aktiviert. Diese Funktion soll später auch allgemein nutzbar gemacht werden.

Um die Firmware bis diese Funktion implementiert ist zu aktualisieren muss das Modul passend konfiguriert sein (Gateway!) und eine Internet Verbindung bestehen. Dann im Kommandomodus zunächst prüfen welche Version installiert ist.

```
Ver
```

Mit

```
ls
```

werden die aktuell im Flash abgelegten Firmware Images angezeigt.

```
FL# SIZ FLAGS
 11  19   3 WiFly_EZX-2.23
 30   1  10 config
 51  20   3 WiFly_EZX-2.32
185 Free, Boot=11, Backup=51
```

Sollten Sie z.B. ein Modul mit Firmware 2.23 erhalten haben, können Sie es auf die zur Zeit aktuelle Version 2.32 updaten diese hat eine neue nützliche Funktion (Kommandomodus per GPIO aktivieren – dies ist schneller).

Dazu

```
ftp update wifly7-232.img
```

eingeben (oder einfach nur `ftp update`, dann wird das aktuellste Image heruntergeladen). Das Modul lädt nun die Firmware herunter und aktiviert das boot image.

Wenn dies erfolgreich abgeschlossen wurde

```
reboot
```

eingeben!

Nach dem Reboot sollte man einen Werksreset durchzuführen und die Einstellungen nochmals komplett neu erledigen – das kann man mit der RobotLoader GUI recht komfortabel erledigen.

Dazu zunächst im Tab Configure Scripts auf

```
Run Factory RESET!
```

klicken. Warten bis das durchgelaufen ist. Danach „Run initial WIFI config (BAUDRATE IS 9600)!" anklicken und warten. Die Ausgaben im log Tab beobachten. Auf einige der Kommandos sollte mit einem grün gefärbten „OK!" geantwortet werden. Hier wird insbesondere die normale Baudrate von 9600 Baud wieder auf 500000 Baud erhöht.

Hat das geklappt, kann man im General Settings Tab die Konfiguration wiederholen – nicht vergessen die WPA Passphrase einzugeben!

Sollte es Probleme mit der neuen Version geben, kann man mit

```
boot image XX
```

ein altes Image wieder aktivieren. XX steht hierbei für die Zahl die das „ls" Kommando in der Spalte #FL vor dem Dateinamen anzeigt. Also nach obigem Beispiel

```
boot image 11
```

für die alte 2.23 Firmware und

```
boot image 51
```

für die neuere 2.32.

Danach reboot und Factory RESET ausführen.

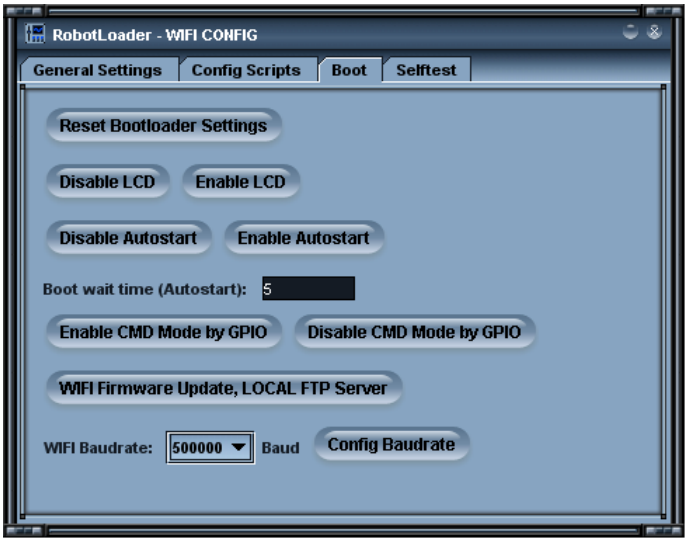
3. GPIO für Kommandomodus verwenden

Mit der Firmware Version 2.32 kann man einen GPIO verwenden um in den Kommandomodus zu wechseln. Dies ist deutlich schneller als 250ms zu warten, $$$ zu senden, wieder 250ms zu warten und zu prüfen ob der Kommandomodus aktiviert wurde.

Dazu muss man

```
set uart cmd_GPIO 14
save
reboot
```

ausführen (Groß- und Kleinschreibung beachten!). GPIO14 kann vom Mikrocontroller aus gesteuert werden.



In neueren RobotLoader Versionen als 2.3b kann das auch direkt im WLAN Konfigurationsdialog im Tab „Boot" erledigt werden. Dort auf den Button „Enable CMD Mode by GPIO" klicken, dann wird dies aktiviert.

Die Library liest automatisch die Einstellung auf dem EEPROM im Mikrocontroller aus und entscheidet anhand dessen welcher Modus aktiv ist. So muss nichts an der Library verändert werden.

In dem genannten Dialog kann übrigens auch eingestellt werden, dass das Programm automatisch (nach einer einstellbaren Wartezeit) startet sobald die Stromversorgung anliegt und das LCD kann deaktiviert werden. Weiterhin kann die Baudrate des WLAN Moduls konfiguriert werden.

4. ADHOC Modus und Modul in den Auslieferungszustand zurücksetzen

Auf dem Modul gibt es einen mit „ADHOC" beschrifteten Jumper zwischen dem UART_SPI1/T5 Steckverbinder und der Antenne.

Wird dieser WÄHREND (bzw. vorher) der Roboter eingeschaltet wird überbrückt, aktiviert sich der ADHOC Modus und die meisten Einstellungen werden zurückgesetzt. Detail Infos finden Sie in der Hersteller Dokumentation. Dieser Modus kann verwendet werden um das Modul ohne serielle Verbindung zu konfigurieren, allerdings wird dies nicht vom RobotLoader unterstüzt – man muss hier direkt auf der Kommandozeile arbeiten. Sich mit dem Modul im ADHOC Modus zu verbinden ist mit einem normalen Notebook auch nicht ganz einfach – je nach verwendeter WLAN Software und je nach Betriebssystem. Dies wird hier daher nicht weiter beschrieben.

Brückt man den Jumper direkt nach dem Einschalten und tastet dann 1x pro Sekunde 5x hintereinander wird das Modul komplett in den Auslieferungszustand zurückversetzt. Das Blinken der LEDs verändert sich wenn dies erfolgreich war.

Danach kann man im RobotLoader wieder die initial Konfiguration durchführen.

C – Entsorgungs- und Sicherheitshinweise

Entsorgungshinweise

Der RP6 und alle Erweiterungsmodule dürfen nicht im normalen Hausmüll entsorgt werden! Diese Komponenten müssen wie alle Elektrogeräte beim Wertstoffhof oder an einer anderen Elektro-Altgeräte Sammelstelle abgegeben werden!

Falls Sie Fragen dazu haben, wenden Sie sich an Ihren Händler.

Sicherheitshinweise für Akkus und Batterien

Akkus und Batterien gehören nicht in Kinderhände! Lassen Sie Batterien/Akkus nicht offen herumliegen, es besteht die Gefahr, dass diese von Kindern oder Haustieren verschluckt werden. Suchen Sie im Falle eines Verschluckens sofort einen Arzt auf!

Ausgelaufene oder beschädigte Batterien können bei Berührung mit der Haut Verätzungen verursachen, benutzen Sie deshalb in diesem Fall geeignete Schutzhandschuhe. Achten Sie darauf, dass die Batterien/Akkus nicht kurzgeschlossen oder ins Feuer geworfen werden. Normale Batterien dürfen außerdem nicht aufgeladen werden! Es besteht Explosionsgefahr! Nur darauf ausgelegte, wiederaufladbare Akkumulatoren wie z.B. NiMH Akkus dürfen mit einem passenden Ladegerät geladen werden!

Entsorgungshinweise für Akkus und Batterien

Akkus und Batterien gehören genausowenig in den Hausmüll wie der Roboter selbst! Sie als Endverbraucher sind gesetzlich (Batterieverordnung) zur Rückgabe aller gebrauchten Batterien und Akkus verpflichtet. Eine Entsorgung über den Hausmüll ist untersagt!

Geben Sie defekte/alte Akkus bzw. leere Batterien deshalb nur bei Ihrem Händler, oder einer Batteriesammelstelle Ihrer Gemeinde ab! Sie können gebrauchte Akkus und Batterien auch überall dort abgeben, wo Batterien und Akkus verkauft werden.

Sie erfüllen damit die gesetzlichen Verpflichtungen und leisten Ihren Beitrag zum Umweltschutz!