Programmierung des ROBO TX Controllers

Teil 1: PC-Programmierung

Package und Dokumentation
„PC_Programmierung_RoboTXC“

V1.5 vom 24.04.2012

Referenzen:

Bezeichnung	Version	Datum
ROBO TX Controller Firmware	1.30	19.03.2012
Library ftMscLib.dll	1.5.11	19.02.2012
ROBOPro	3.1.3	26.03.2012

MSC Vertriebs GmbH
Design Center Aachen
Pascalstr. 21
52076 Aachen

Inhalt

1 Einführung

Dieses Dokument beschreibt die Programmierung des fischertechnik ROBO TX Controllers vom PC aus, insbesondere für PC-Programmierer, die nicht mit der fischertechnik Programmieroberfläche ROBOPro arbeiten, sondern eigene PC-Programme für den ROBO TX Controller erstellen möchten.

Als Schnittstellen zum PC stehen USB oder Bluetooth zur Verfügung. In beiden Fällen steht zur Kommunikation ein virtueller COM-Port auf dem PC zur Verfügung. Die USB-Schnittstelle bildet die USB-Standardklasse *Communication Device Class (CDC)* ab. Die Bluetooth-Schnittstelle folgt dem *Serial Port Profile (SPP)*.

1.1 Systemvoraussetzungen

Die PC-Programmierung des ROBO TX Controllers, soweit in diesem Dokument beschrieben, und soweit von der mitgelieferten Library unterstützt, setzt ein Microsoft Windows Betriebssystem (2000, XP, Vista) voraus. Ebenso wurden Standard Microsoft Entwicklungswerkzeuge wie Microsoft Visual C++ oder Microsoft Visual Studio verwendet, um die mitgelieferten Beispielprojekte zu erstellen.

Jenseits der Programmierung unter Microsoft Windows ist grundsätzlich eine Programmierung des ROBO TX Controllers aber auch mit anderen PC-Betriebssystemen möglich auf denen die verwendeten Standard-Schnittstellen (USB-CDC, Bluetooth-SPP) unterstützt werden.

Weitere Systemvoraussetzungen sind ein USB-Fullspeed-Host-Port auf dem PC, sowie eine Bluetooth-Schnittstelle (im PC eingebaut oder über externen Adapter, z.B. ein Bluetooth-USB-Stick).

1.2 Begriffsdefinitionen

ROBO TX Controller heißt der fischertechnik Robotik-Controller selbst. Er wird auch abkürzend mit **TX-C** bezeichnet.

Firmware ist die Software, die auf dem ROBO TX Controller, also das Betriebssystem, Gerätetreiber, Protokolle und der Boot Loader.

Robo-Programm bezeichnet den Teil der Software, der als ladbare, ausführbare Robo-Applikation auf dem ROBO TX Controller laufen kann. Voraussetzung ist auf dem TX-C natürlich eine vorhandene und lauffähige Firmware.

2 Software-Struktur auf dem PC

Der ROBO TX Controller bietet 2 physische Schnittstellen zum PC hin: USB und Bluetooth. In beiden Fällen werden COM-Port-Treiber verwendet, die aus Sicht der PC-Software wie serielle Standard-Schnittstellen (RS232) verwendet werden können. Es kann aber zu jedem Zeitpunkt immer nur eine von beiden Schnittstellen aktiv sein: entweder USB oder Bluetooth.

Die Low-Level-Schnittstelle ist der jeweilige COM-Port selbst. Unter Verwendung der Library ftMscLib.dll steht alternativ eine High-Level-API zur Verfügung, die die Funktionalität des ROBO TX Controllers in komfortablen Funktionsaufrufen wiedergibt. Das nachfolgende Bild zeigt dies auch noch mal grafisch:

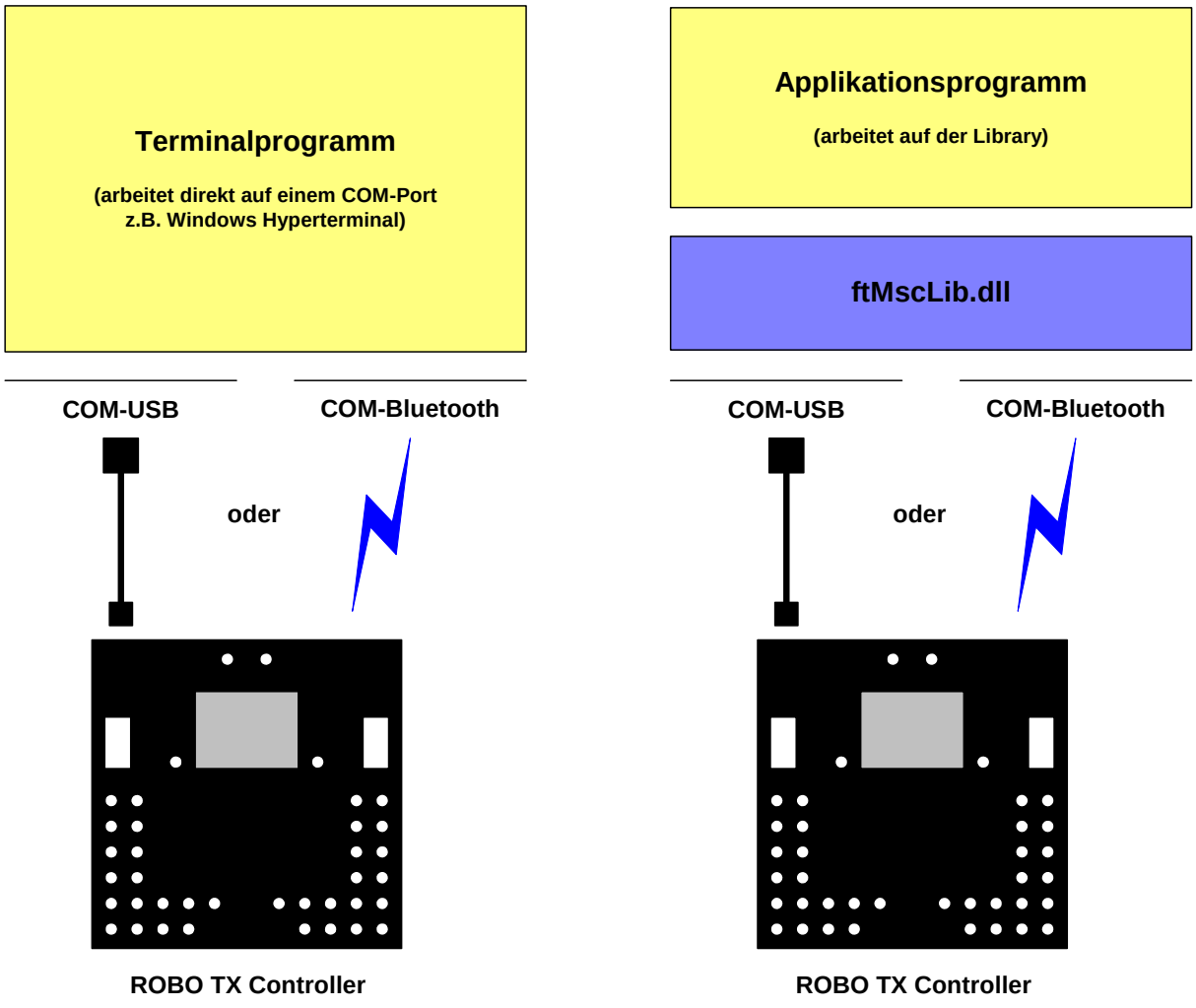


Abb. 1: Schnittstellen vom ROBO TX Controller zum PC

2.1 Low-Level-API (COM-Port Level)

Die Low-Level-API ist der COM-Port selbst. In diesem Falle wird die Library ftMscLib.dll nicht verwendet, sondern direkt über den COM-Port auf den ROBO TX Controller zugegriffen. Hierbei gibt es 3 Unterschnittstellen:

- Remote Shell
- File-Transfer-Interface (X-Modem)
- X1-Protokoll-Schnittstelle

Die Firmware auf dem ROBO TX Controller erkennt automatisch welche Schnittstelle verwendet wird. Wenn die einlaufenden Daten eine bestimmte Struktur haben (X-Modem oder X1), dann wird der entsprechende Dienst verwendet. Alle anderen eingehenden Daten, werden als Remote Shell Kommandos interpretiert. Mit einem Standard-Terminalprogramm, wie z.B. dem Windows Hyperterminal, können bereits die Remote Shell und das File Transfer Interface bedient werden.

Eine genauere Beschreibung der verschiedenen Unterschittstellen der Low-Level-API befindet sich in Kapitel 13.

2.2 High-Level-API (Library Level)

Die High-Level-API wird durch die Funktionen der Library ftMscLib.dll dargestellt. Letztere verwendet die verschiedenen Unterschnittstellen der Low-Level-API intern, um Ihrerseits in zusammengefassten Funktionsaufrufen die Programmierung des ROBO TX Controllers zu erleichtern. Insbesondere das relativ komplexe X1-Protokoll verschwindet so unter einer wesentlich einfacheren Oberfläche.

Eine Beschreibung der Library (i.e. der High-Level-API) befindet sich in Kapitel 5 sowie ausführlich mit allen verfügbaren API-Funktionen im separaten Dokument **`Windows_Library_ftMscLib.pdf`**.

3 Treiber-Installation auf dem PC

3.1 Installation des USB-Treibers

Das laufende Board wird per USB-Kabel im Betrieb an den PC angeschlossen (ggf. nochmals aus- und wieder einstecken). Windows wird anschließend das Vorhandensein eines neuen USB-Devices melden und nach einem Treiber suchen. Wählen Sie im automatisch startenden Installationsassistenten auf die Frage nach „Windows-Update“ die Auswahl „Nein, diesmal nicht“ und im nächsten Fenster „Software von einer bestimmten Liste oder Quelle installieren“ und verweisen Sie auf die mitgelieferte Datei **`\USB_Driver\ROBO_TX_Controller.inf`**. Anschließend sollte im Geräte-Manager ein zusätzliches COM-Port-Device mit der Bezeichnung “fischertechnik USB ROBO TX Controller” auftauchen. Notieren Sie am besten die COM-Port-Nummer, die von Windows automatisch vergeben wurde (siehe Geräte-Manager).

3.2 Bluetooth-Installation

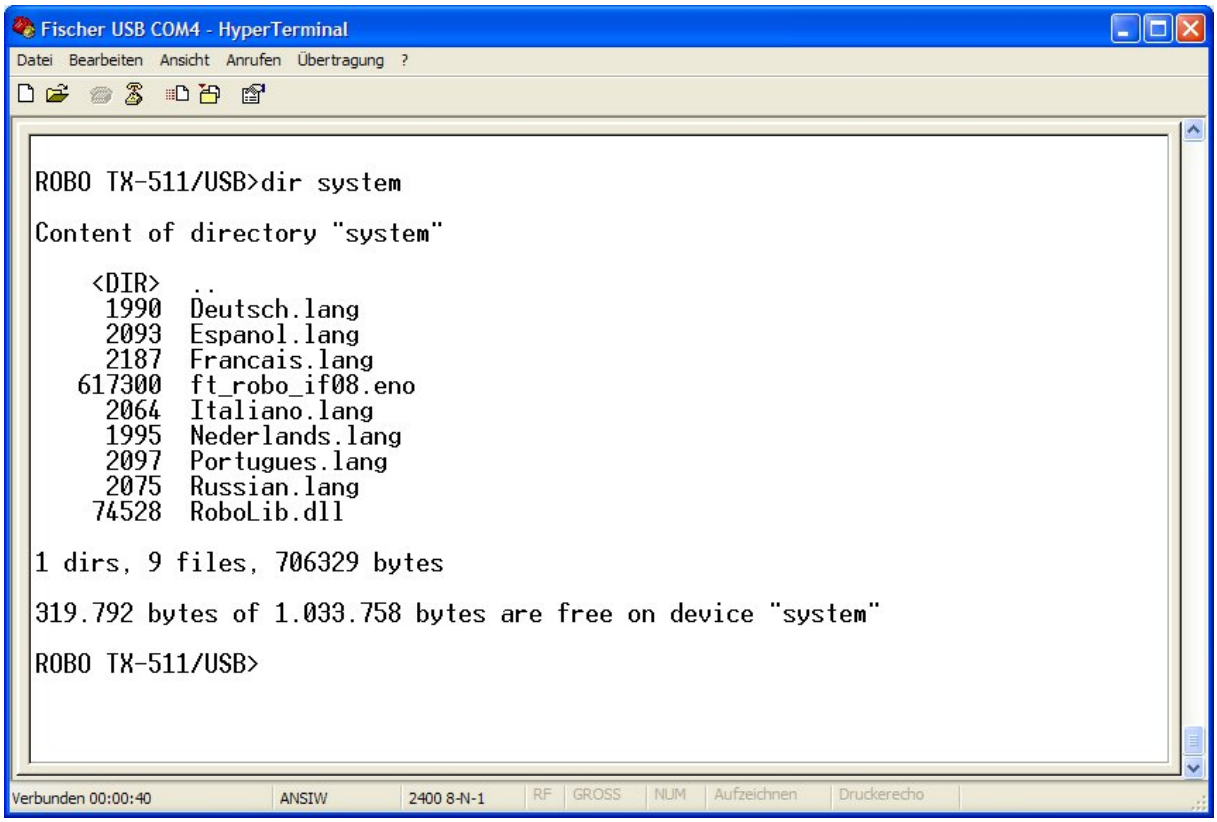
Der laufende ROBO TX Controller kann jederzeit vom PC-Bluetooth-Dienst erkannt werden, sofern diese Eigenschaft („Bluetooth Gerät sichtbar“) in den entsprechenden Einstellungen nicht ausgeschaltet ist. Es erscheint im Suchfenster auf dem PC mit seinem Host-Namen, der auch auf dem Display angezeigt wird (z.B. „ROBO TX-511“). Eine Kopplung mit dem PC erfolgt unter Verwendung des Hauptschlüssels (PIN) „1234“. Von dem Moment an ist dem ROBO TX Controller ein fester COM-Port zugeteilt und die Bluetooth-Verbindung wird vom PC automatisch aufgebaut, sobald auf den entsprechenden COM-Port zugegriffen wird.

Je nachdem, welcher Bluetooth-Protokoll-Stapel und welche Bluetooth-Hardware auf dem PC verwendet werden, kann sich die Installation und die Kopplung des PCs mit dem TX-C im Ablauf unterscheiden. Hierbei sind in jedem Fall die Installationsanweisungen des jeweiligen Herstellers zu befolgen.

3.3 Erster Test von USB und Bluetooth

Ein erster Test der USB-Kommunikation bzw. der Bluetooth-Kommunikation mit dem Board kann mit einem einfachen Terminal-Programm (z.B. HyperTerminal, ZOC oder TeraTerm) erfolgen. Wählen Sie die richtige COM-Port-Nummer des USB- bzw. Bluetooth-Treibers aus und konfigurieren Sie eine beliebige Baurate (spielt keine Rolle).

Wenn die Verbindung aufgebaut werden konnte, erhalten Sie im Terminal-Programm die Monitor-Konsole des Betriebssystems 4NetOS, das auf dem ROBO TX Controller läuft. Mit jedem Drücken von <ENTER> sollte nun ein Kommandozeilen-Prompt erscheinen. Durch Eingabe von „dir“ können Sie sich weiterhin den Inhalt der virtuellen Laufwerke „System“, „Flash“ oder „Ramdisk“ anzeigen lassen:



Weitere Befehle erhalten Sie durch Eingabe von „?“ oder „help“. Diese sind in Kapitel 13.2 näher beschrieben.

4 Test mit RoboTxTest.exe

4.1 Installation

Mit dem Windows-basierten Programm **RoboTxTest.exe** steht ein einfaches Programm zur Diagnose und zum Testen der Funktionalität des ROBO TX Controllers zur Verfügung. Die Kommunikation von **RoboTxTest** mit dem Controller basiert auf der gleichen PC-Library **ftMscLib.dll**, die auch von der grafischen Benutzeroberfläche ROBOPro eingesetzt wird. Eine ähnliche Funktionalität mit einem geringeren Umfang bietet auch das ROBOPro mit dem integrierten Testfenster „Interface-Test“ an. Insbesondere ist bei **RoboTxTest** die implementierte Remote-Shell-Schnittstelle zum ROBO TX Controller zu erwähnen, die von ROBOPro Software nicht angeboten wird.

Eine spezielle Installation für das Diagnosetool und der Library ist nicht erforderlich, wenn alle Dateien von der CD bereits auf den PC kopiert wurden. Das ausführbare Programm **RoboTxTest.exe**, die dazugehörige DLL **ftMscLib.dll** müssen lediglich zusammen im gleichen Verzeichnis (hier: **\RoboTxTest**) liegen. Das Diagnosetool **RoboTxTest.exe** (nicht die DLL) wurde als .NET-Programm erstellt und erfordert daher eine Installation des .NET Framework 2.0 oder eine aktuellere Version auf dem jeweiligen Windows PC.

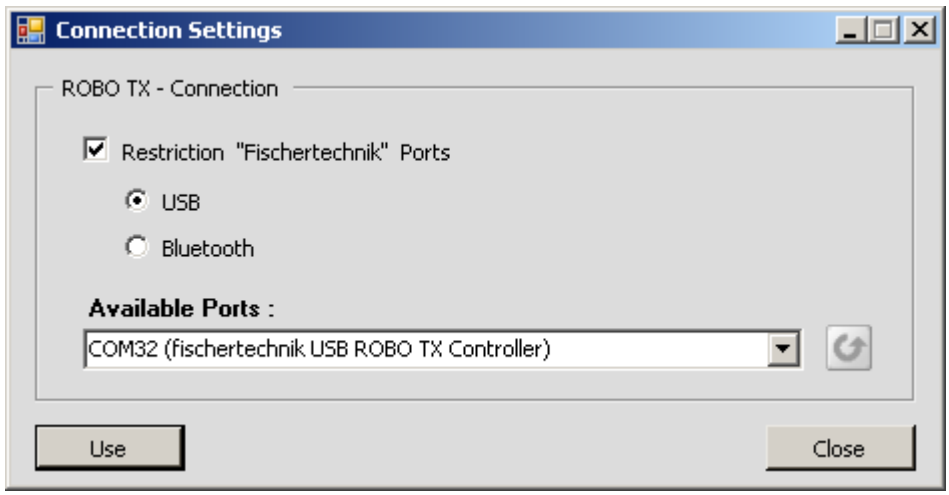
Insofern noch keine .NET-Umgebung auf Ihrem Rechner vorhanden ist, wird die Installation derselben beim ersten Aufruf von **RoboTxTest.exe** automatisch angeboten. Hierzu ist eine Verbindung zum Internet erforderlich.

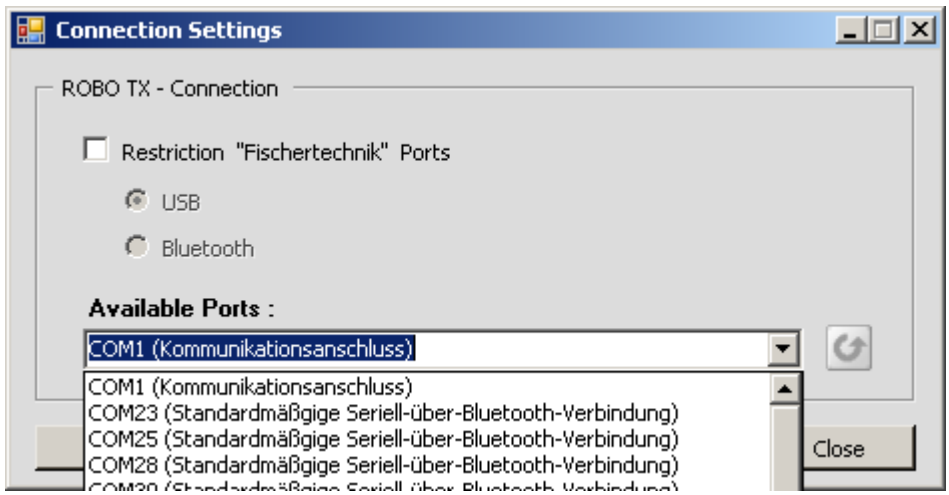
Mit jedem Aufruf des Programms **RoboTxTest.exe** wird zudem noch eine Logging-Datei **ftlib.log** angelegt, mit welcher eine Fehlerdiagnose erfolgen kann, wenn etwas nicht auf Anhieb funktioniert. Eine bereits vorhandene Loggingdatei wird in die Datei **ftLib.bak** umbenannt.

4.2 Verbindung zum ROBO TX Controller

Mit dem Programm **RoboTxTest.exe** kann der Benutzer eine Verbindung zum ROBO TX Controller über einen auszuwählenden (virtuellen) COM-Port öffnen (die COM-Port-Nummer ist nicht eingeschränkt), der wiederum über den entsprechenden Windows-Gerätetreiber die USB- oder die Bluetooth-Schnittstelle nutzt. Die Auswahl des COM-Ports erfolgt entweder über das Menü [**Connection**] aus der Menüleiste oder durch einen Click auf das untere Textfeld der COM-Port Beschreibung. Danach öffnet sich ein neues Fenster, welches durch Festlegung der Verbindungsart eine Auswahl der zur Verfügung stehenden COM-Ports anbietet. Wird die Einschränkung für 'Fischertechnik Ports' aufgehoben, werden alle verfügbaren COM-Ports aufgelistet. Die Liste der COM-Ports wird dabei entsprechend der Auswahlkriterien automatisch generiert und durch Anklicken der Schaltfläche angezeigt.

Auswahl eines COM-Ports:



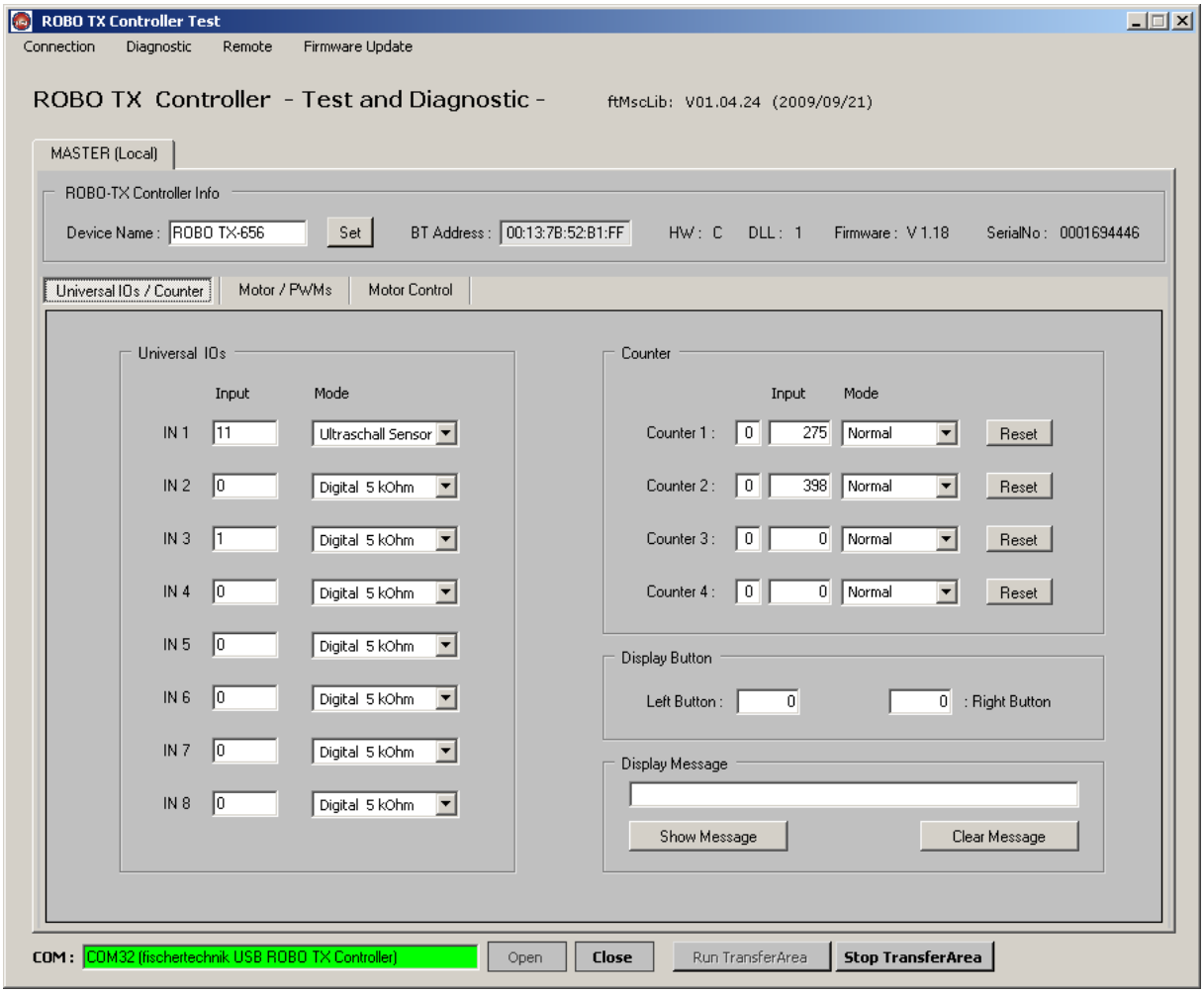


Nach der Auswahl eines COM-Ports und der Bestätigung über den Button [**Use**] wird der ausgewählte Port in der Textzeile des Hauptfensters angezeigt und kann anschließend geöffnet werden. Button [**Close**] schließt die Verbindung wieder.

4.3 Diagnose und Tests mit dem ROBO TX Controller

Nach einem erfolgreichen Verbindungsaufbau wird das Textfeld grün angezeigt. Die notwendigen Informationen des ROBO TX Controller werden ausgelesen und im oberen

Bereich 'ROBO-TX Controller Info' ausgegeben. Zusätzlich wird anhand der ausgelesenen Informationen, die entsprechende Anzahl von angeschlossenen Slave-Interfaces, sprich Extension-Boards, als weitere Registerkarte angezeigt. Sind keine Slaves angeschlossen, wird nur das lokale Interface (Master) als Registerkarte angezeigt. Zum Betrieb mehrere ROBO TX Controller lesen Sie bitte auch Kapitel 11.



Über die Aktivierung des Buttons [**Run TransferArea**] wird die Kommunikation mit dem ROBO TX Controller gestartet und die aktuell anliegenden Werte werden angezeigt.

Folgende Informationen, bzw. Zustände des Controllers werden auf der Registerkarte '**Universal IO's / Counter**' angezeigt:

- Werte die, entsprechend der Konfiguration des Eingangs, an den 8 Universal Inputs anliegen (IN1 ... IN8)
- aktueller Zählerstand der 4 Zählereingänge (C1 ... C4)
- Zeitmessung beim Drücken der beiden Auswahltaster auf dem Controller (ms)

Anzeigen der Universaleingänge: Die anliegenden Werte können durch die Auswahl des entsprechenden Anzeigemodus beeinflusst werden. Es werden die Werte einer Spannungs- oder einer Widerstandsmessung für jeden Universal-Eingang separat angezeigt. Optional

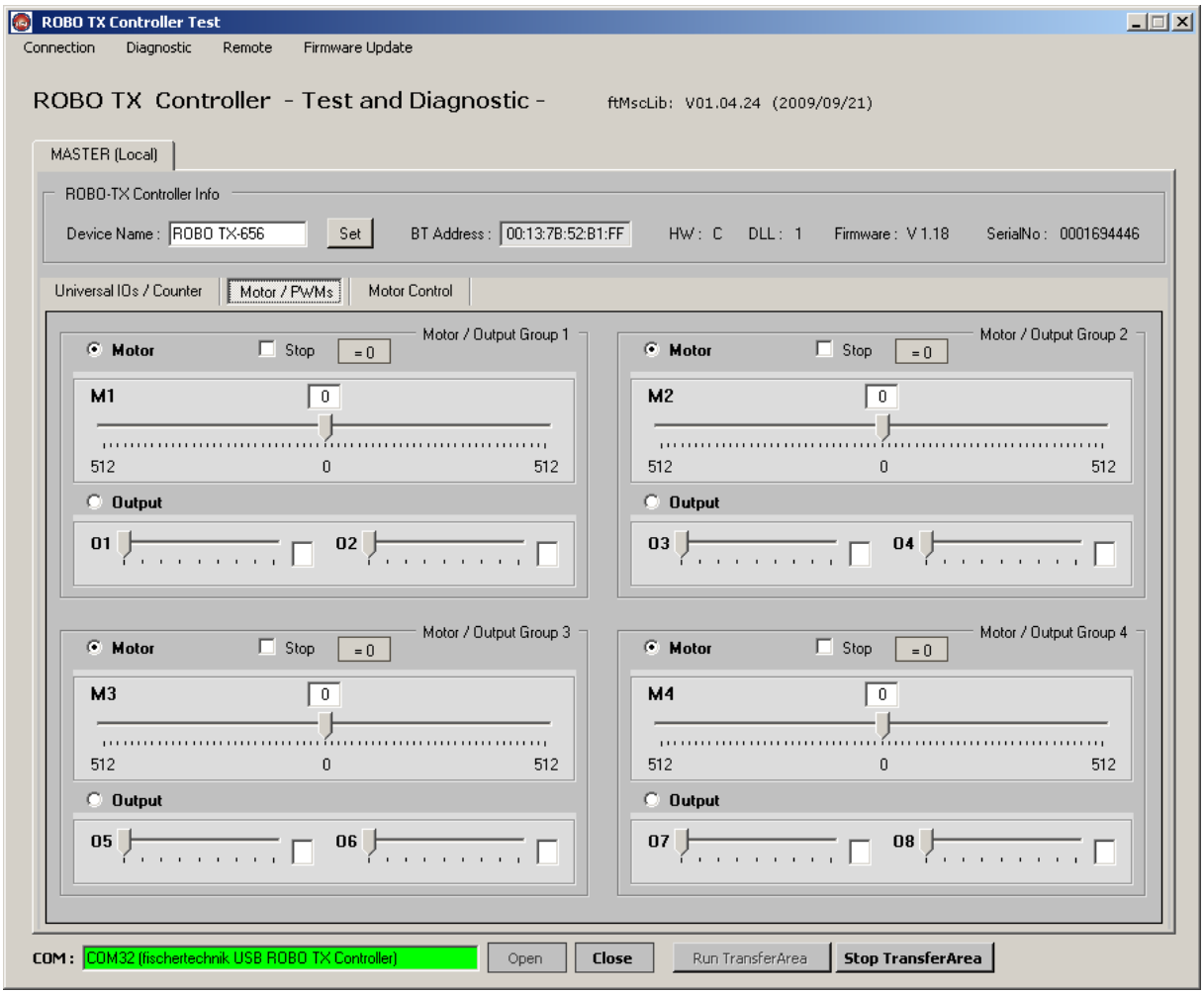
kann die Ausgabe digital (1/0) angezeigt werden. Liegt ein Werteüberlauf an einem Universal-Eingang vor, wird im Ausgabefeld die Zeichenfolge "*overrun*" angezeigt.

Anzeige der Zählereingänge: Für die 4 digitalen Zählereingänge (Counter 1 bis 4) wird der aktuelle Zählerstand in den Feldern angezeigt. Den Zähler kann durch Anklicken des Button [**Reset**] wieder zurückgesetzt werden. Ein entsprechender Reset-Befehl wir dabei zum ROBO TX Controller gesendet.

Anzeige Zeitmessung der gedrückten Auswahltasten: Für die beiden Auswahltasten auf dem ROBO TX Controller wird hier die Zeit ausgegeben, wie lange diese Taster auf dem Controller gedrückt bleiben. Die Anzeige der Zeit wird in Millisekunden (ms) ausgegeben.

Mit der Option '**Display Message**' kann eine Meldung mit max. 64 Zeichen auf dem Display des ROBO TX Controllers ausgegeben werden. Dazu editiert man eine Meldung in der vorgegebenen Zeile und aktiviert den Button [**Show Message**]. Die Meldung wird damit zum Controller gesendet und auf dem Display angezeigt. Es können mehrere Meldungen hintereinander gesendet werden, angezeigt wird jedoch immer die zuletzt gesendete. Durch Betätigen der rechten Auswahltaste wird die letzte aktuelle Meldung wieder gelöscht. Der Button [**Clear Message**] löscht jedoch alle Meldungen auf dem Controller.

Wählt man die Ansicht '**Motor/PWMs**', so erhält man folgende Anzeigeoptionen:

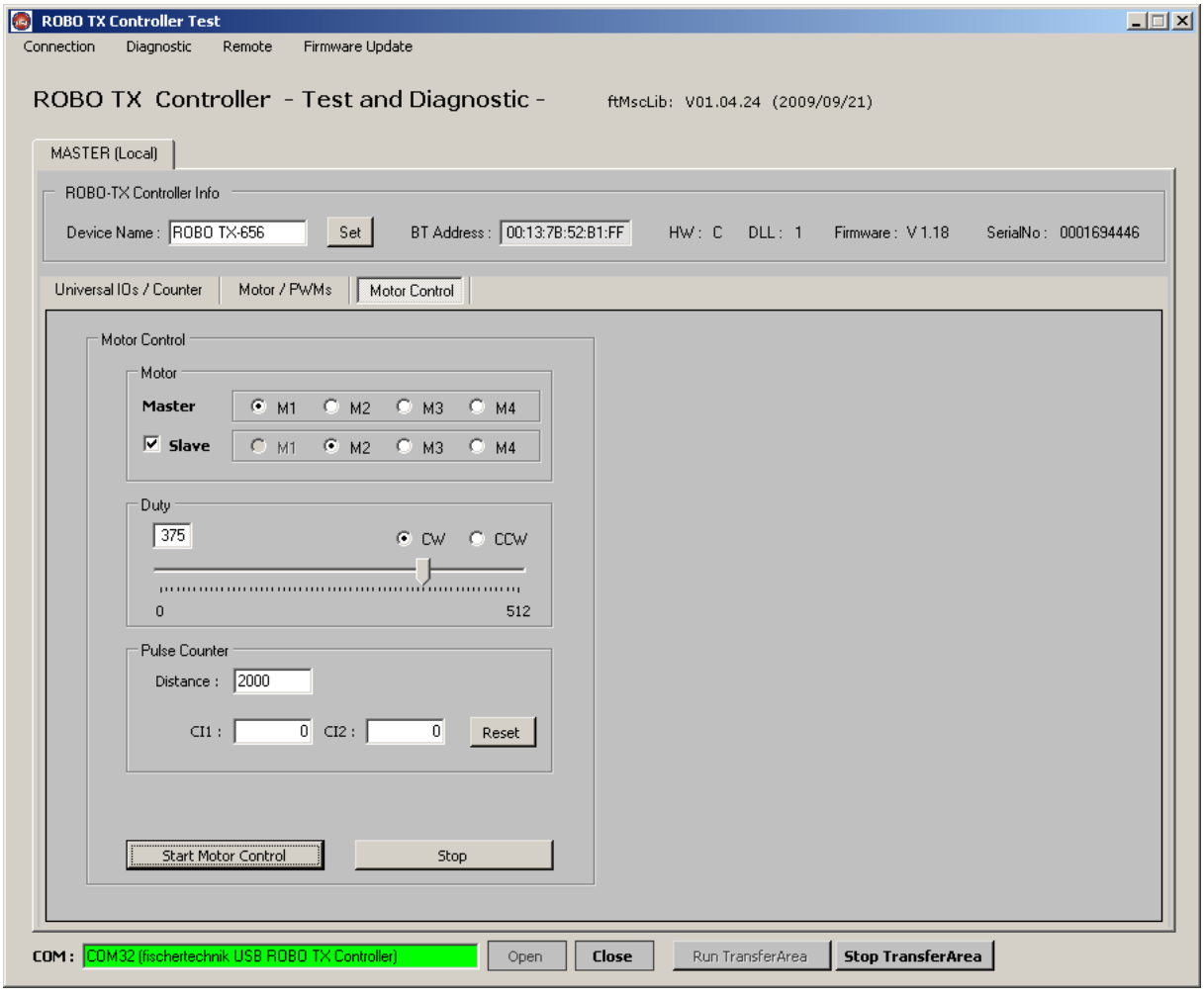


Hier können die 4 Motoren oder die 8 PWM-Ausgänge einzeln gesteuert werden. Entweder wird ein Motor oder das entsprechende Paar von PWM-Ausgängen durch unabhängige Schieberegler gesteuert.

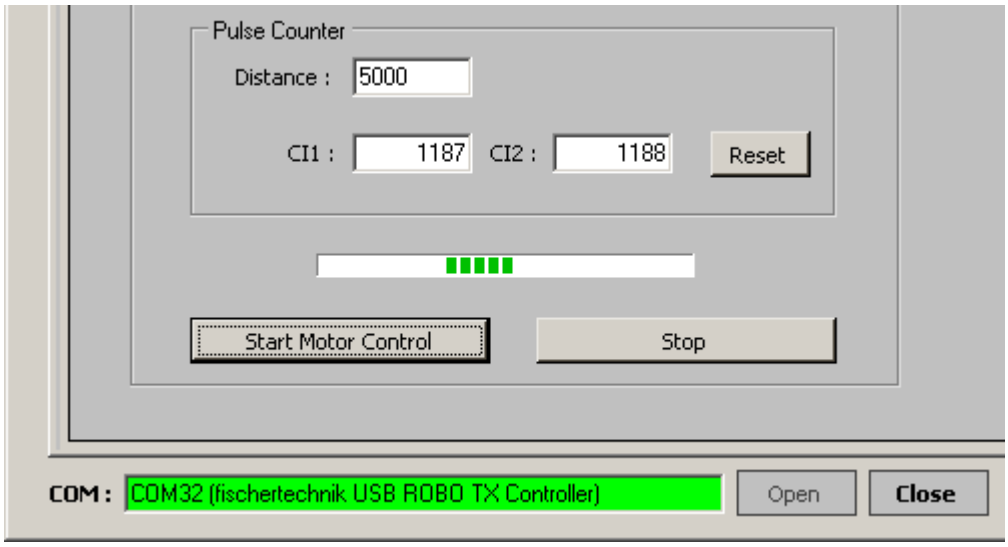
Motoransteuerung: Für die 4 Motoren M1 bis M4 kann die Drehrichtung und die Motorgeschwindigkeit durch Bewegen des Schiebereglers nach links oder rechts gesteuert werden. Die Geschwindigkeit kann in feinen PWM-Schritten von 0 bis 512 eingestellt werden. Die Option [**Stop**] erlaubt es einen Motor augenblicklich zu stoppen unter Beibehaltung der Geschwindigkeitseinstellung. Die Option "=0" fährt den Motor augenblicklich und präzise auf die Geschwindigkeit 0 wieder zurück (Stillstand).

Wechselt man von der Option '**Motor**' auf die Option '**Output**', wird automatisch eine Konfigurationsänderung zum Interface gesendet, welche die Motoren deaktiviert und somit die beiden Ausgänge einer Motorgruppe als unabhängige Outputs behandelt. Die Ansteuerung der jeweiligen PWM-Ausgänge einer Motorgruppe erfolgt nun über die separaten Schieberegler in Schritten von 0 bis 8. Der Schieberegler für den Motor einer Gruppe ist dabei deaktiviert.

Eine Motorsynchronisierung kann über folgende Einstellungen unter der Registerkarte '**Motor Control**' getestet und durchgeführt werden.

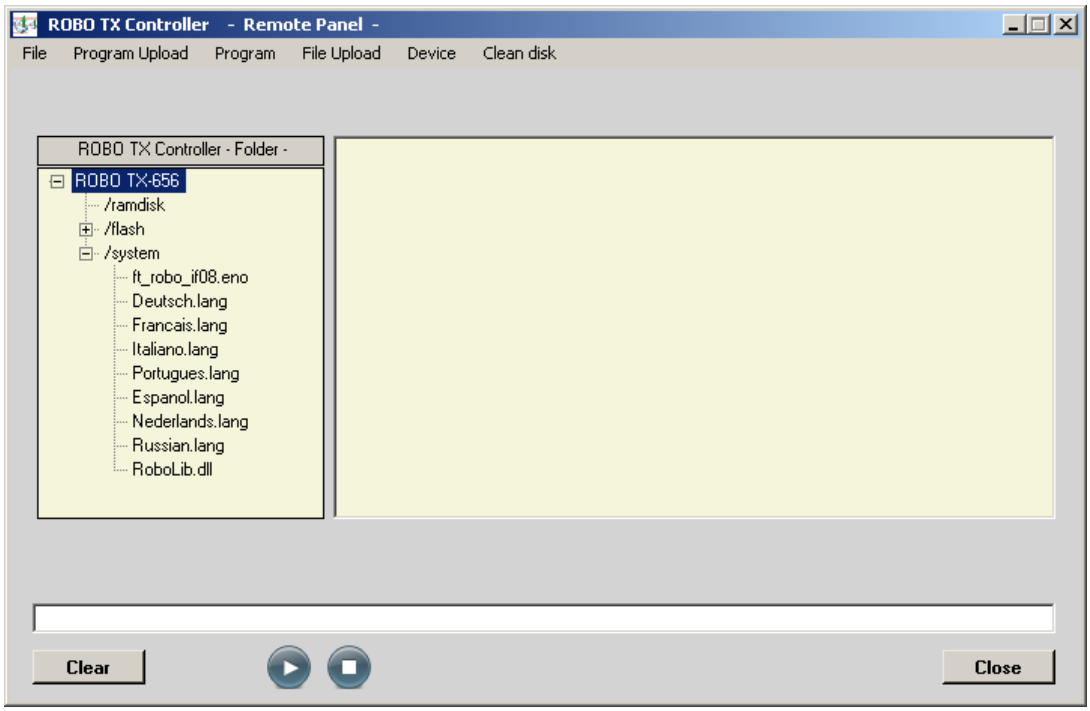


Man wählt zunächst einen Master-Motor aus und weist diesem optional einen Slave-Motor zu. Innerhalb der Einstellung '**Duty**' werden die zu fahrende Geschwindigkeit, sowie die Drehrichtung der Motoren eingestellt. Über die Eingabe der Distanz wird der Anwendung mitgeteilt, wann das Ziel des Motorsynchronisationslaufs erreicht ist. Die beiden Zähleingänge dokumentieren den augenblicklichen Zählerstand während des Synchronlaufs. Mit dem Button [**Reset**] können diese Zähler auch während des laufenden Betriebs einer Motorsynchronisation zurückgesetzt werden. Das Ziel wird dadurch natürlich etwas später erreicht. Der Button [**Start Motor Control**] startet dabei eine aktive Motorsynchronisation, Button [**Stop**] stoppt zwischenzeitlich den Synchronlauf, bis dieser wieder mit [**Start Motor Control**] angestoßen wird. Ein Starten des Synchronlaufs bewirkt immer eine Zurücksetzung der Zählerstände. Haben die Motoren oder der Motor das vorgegebene Ziel erreicht, so werden die Motoren augenblicklich gestoppt. Während eines Synchronlaufs wird ein umlaufender Fortschrittsbalken angezeigt, der die aktive Synchronisation symbolisiert.



4.4 File-Upload und Remote Shell

Über das Menü [**Remote**] kommt man zu folgendem Zusatzfenster:

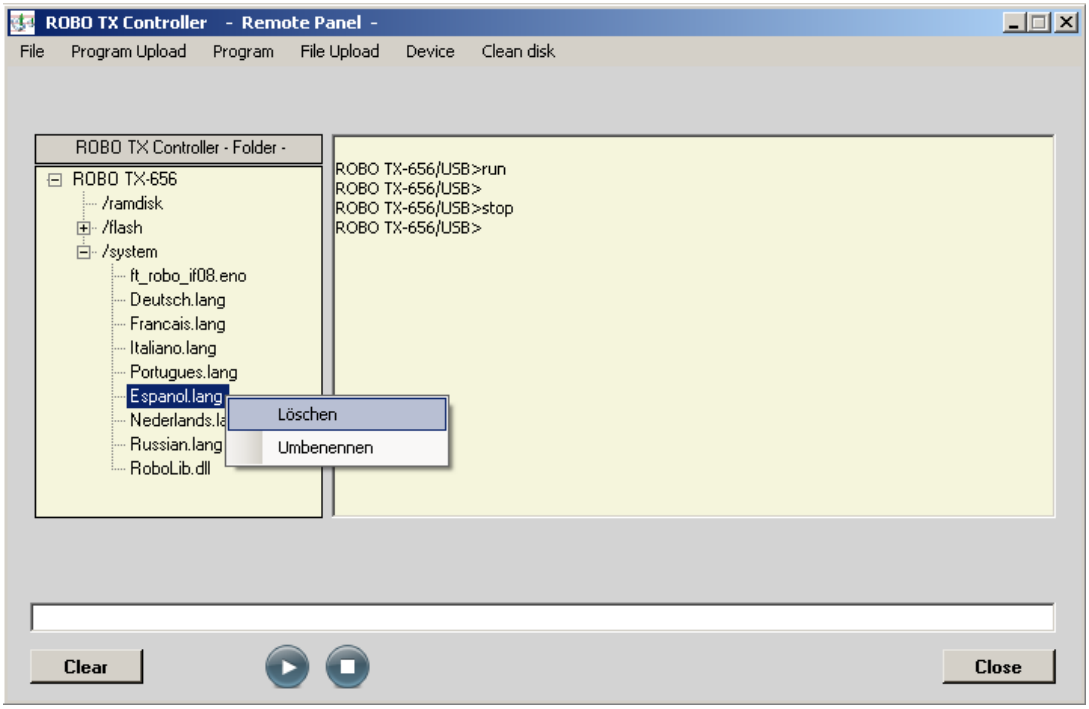


Das Fenster ist unterteilt in einem Bereich 'ROBO TX Controller Folder' und einem Bereich für die Konsolenausgabe (Remote Shell). Im linken Bereich wird der Inhalt der verfügbaren Laufwerke '/ramdisk', '/flash' und '/system' in einer hierarchischen Struktur ausgegeben. Die Laufwerke lassen sich bei einem Klick auf das "+"-Zeichen öffnen und zeigen die dort momentan gespeicherten Dateien an.

Über die Menüauswahl [**File**] kann eine beliebige Datei, die zum Interface geladen werden soll, ausgewählt werden. Die getätigte Auswahl eines Directorys wird dabei in der Ini-Datei "robotest.ini" gespeichert und steht beim nächsten Aufruf wieder zur Verfügung. Nach Auswahl einer Datei werden die Dateiinformationen im Konsolenfenster ausgegeben. Über das Menü [**File Upload**] wird das Upload der ausgewählten Datei in das Dateisystem des ROBO TX Controller aktiviert. Es stehen als „Ziellaufwerk“ alle verfügbaren Laufwerke zur Verfügung. Nur die Flashdisk speichert Dateien dauerhaft, so dass diese beim nächsten Einschalten des Controllers wieder zur Verfügung stehen.

Über das Menü [**Program Upload**] können nur lauffähige Programme in das entsprechende Ziellaufwerk geladen werden. Programme können danach mit den Kommandos [**Run**] und [**Stop**] aus der Eingabezeile heraus oder über das Menü [**Program**] gestartet oder gestoppt werden.

Dateien auf den Laufwerken können entweder über den Konsolenbefehl '**del**' oder aber durch das Aktivieren des Kontextmenüs "**Löschen**" (rechte Maustaste) auf dem Dateinamen im linken Fenster gelöscht werden, ebenso wird das Umbenennen einer Datei als Kontextmenü angeboten.



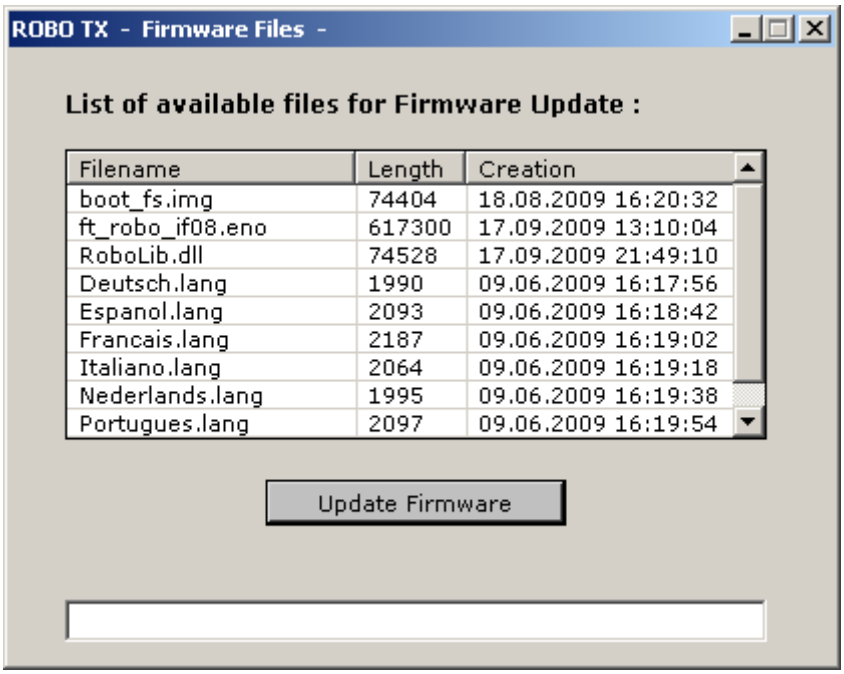
Mit Hilfe des Menüs [**Clean disk**] kann der Inhalt der beiden Laufwerke /ramdisk und /flash vollständig gelöscht werden. Ein entsprechender Konsolenbefehl "**clean_disk /ramdisk**" oder "**clean_disk /flash**", eingegeben in der Kommandozeile, führt die gleiche Aktion aus.

Die untere Eingabezeile dient der Eingabe von Konsolenbefehlen. Die Ausgabe des ROBO TX Controllers auf einen abgesetzten Befehl wird im oberen rechten Fenster ausgegeben. Mit dem Befehl "**?**" kann man sich alle zur Verfügung stehenden Konsolenbefehle des ROBO TX Controller ausgeben lassen. Button [**Clear**] löscht den Inhalt des rechten Ausgabefensters.

Die Konsole „Remote Panel“ entspricht dem, was man auch mit einem Terminalprogramm (z.B. Windows HyperTerminal) im Direktzugriff auf den ROBO TX Controller über einen COM-Port erhalten bzw. sehen würde.

4.5 Firmware-Update

Über das Menü [**Firmware Update**] ist das Update der Firmware inklusive der benötigten Sprachdateien aus einem beliebigen Verzeichnis möglich. Nach der Auswahl eines Verzeichnisses, welches die notwendigen Dateien zum Firmware-Update beinhaltet (hier: Unterverzeichnis **`/Firmware`**), erscheint folgendes Info-Fenster mit einer Liste der Dateien, die für ein Firmware-Update zur Verfügung stehen.



Durch die Aktivierung des Buttons [**Update Firmware**] wird das Firmware-Update gestartet. Über einen danach eingeblendeten Fortschrittsbalken wird der Verlauf zeitlich dargestellt. In der unteren Ausgabezeile werden die durchgeführten Schritte, sprich die Befehle, während des Firmware-Updates ausgegeben. Zwischendurch kommt es zu einem scheinbaren kurzen Stillstand. Dann werden nämlich die Befehle '**flush_disk flash**' und '**flush_disk system**' auf dem ROBO TX Controller ausgeführt, was eine gewisse Zeit in Anspruch nimmt. Wenn das Firmware-Update erfolgreich beendet wurde, wird unten in der Ausgabezeile die Meldung "Firmware Update finished..." ausgegeben. Das Fenster kann danach wieder geschlossen werden und im ROBO TX Controller befindet sich nach einem Aus- und wieder Einschalten die zuvor gesendete Firmware-Version.

5 Beschreibung der Library ftMscLib.dll

Die Windows Library "ftMscLib“ unterstützt, analog zur Library "FtLib" des ROBO Interface (Firma Knobloch), die Kommunikation mit dem neuen ROBO TX Controller im Online-Betrieb.

Dabei wurden möglichst äquivalente Funktionsaufrufe definiert, um ein Minimum an Aufwand zur Implementierung der neuen Library zu haben. Als mögliche Returnwerte wurden die Error-Codes aus der bisherigen ftLib.h benutzt und um neue erweitert.

Die Windows Library wurde mit Microsoft Visual C++ Studio 6.0 erstellt und steht in folgenden Versionen zur Verfügung:

Als statische Library (WIN32-Bibliothek .lib)

- `FtMscLib_Static_LIBCMT_Release.lib`	Multithreaded (Linkoption /MT)
- `FtMscLib_Static_LIBCMTD_Debug.lib`	Multithreaded debuggen (Linkoption /MTd)

Als dynamische Library (.dll):

- `ftMscLib.dll, ftMscLib.lib`	Multithreaded DLL (Linkoption /MD)
- `ftMscLib_debug.dll, ftMscLib_debug.lib`	Multithreaded DLL debuggen (Linkopt. /MDd)

Für die MultiMedia-Timer Aufrufe innerhalb der Library wurde das WIN32 Bibliothek-Modul "Winmm.lib" dazugelinkt.

Die Weitere Beschreibung der Library und eine vollständige Auflistung aller Library-Aufrufe (High-Level-API) findet sich im Dokument `Windows_Library_ftMscLib.pdf`.

6 Bluetooth Messaging API

Neben der Möglichkeit einer PC-Host-Verbindung über Bluetooth können die ROBO TX Controller auch programmgesteuert per Bluetooth kurze Textnachrichten untereinander austauschen. Hierüber können von einem Controller zum anderen Aktionen ausgelöst werden oder sogar Programme auf mehreren Controllern im laufenden Betrieb über Funk synchronisiert werden.

Um dies zu ermöglichen gibt es eine Anzahl API-Funktionen die den Verbindungsaufbau (aktiv oder passiv) von einem Controller zu einem (oder mehreren) anderen regeln und die den Nachrichtenversand und -empfang ermöglichen.

6.1 Netzwerktopolgie, aktiver- und passiver Verbindungsaufbau

Wenn z.B. zwei ROBO TX Controller Nachrichten untereinander austauschen sollen, so ist einer immer der MASTER und der andere der SLAVE. Der MASTER ist derjenige, der den Verbindungsaufbau initiiert (aktiv, connect) und der SLAVE ist derjenige, der darauf wartet von dem MASTER verbunden zu werden (passiv, listen):

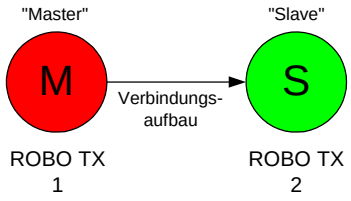


Abb. 6-1

Sollte mal ein Verbindungsabbruch passieren, etwa weil die 2 Controller sich jeweils auf fahrenden Modellen befinden und diese sich zu weit voneinander entfernt haben (außer Funkreichweite, wenn mehr als ca. 10 m dazwischen liegen), so obliegt es wieder dem MASTER die Verbindung wieder herzustellen. Der SLAVE, der einen Verbindungsabbruch registriert, tut weiter nichts außer das erneute Verbinden des MASTERs abzuwarten.

Hat man mehr als 2 Controller im Bluetooth-Netzwerk-Verbund, so ist wiederum genau einer der MASTER und alle anderen SLAVEs:

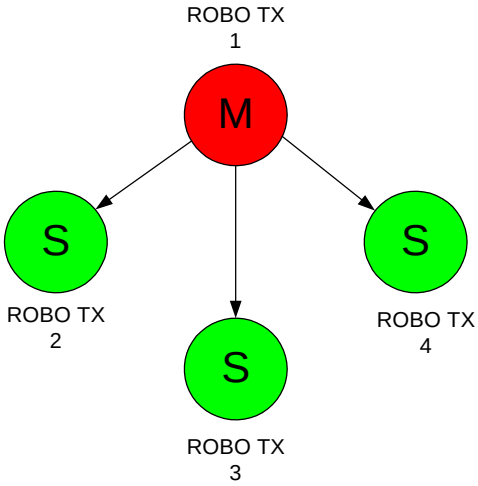


Abb. 6-2

In diesem Beispiel würde der Nachrichtenversand von ROBO TX 2 zu ROBO TX 3 über den MASTER (ROBO TX 1) weitergeleitet (geroutet). Hierfür ist im Applikationsprogramm zu sorgen.

6.2 Bluetooth-Messaging im Online Mode

Wenn man das Bluetooth Messaging im Online Mode betreiben will, gibt es ein paar Dinge zu beachten. Die Verbindung vom Host PC zum MASTER des Bluetooth Messagings sollte nicht gleichzeitig eine Bluetooth-Verbindung sein, weil sonst der Host-PC versucht die Rolle des MASTERs zu übernehmen. Der Controller, der die Rolle des MASTERs spielt, sollte vom Host PC also immer über eine USB-Verbindung angebunden sein. Eine Verbindung vom Host-PC zu einem Controller, der für das Bluetooth Messaging eine SLAVE Rolle spielt, ist dagegen problemlos möglich, da hierdurch kein Rollenwechsel antizipiert wird. Ein SLAVE kann auch Verbindungen von verschiedenen MASTERn annehmen, solange er nicht selbst versucht Bluetooth-Verbindungen aufzubauen (wodurch er zum MASTER würde).

Nachfolgend ein paar Beispiele:

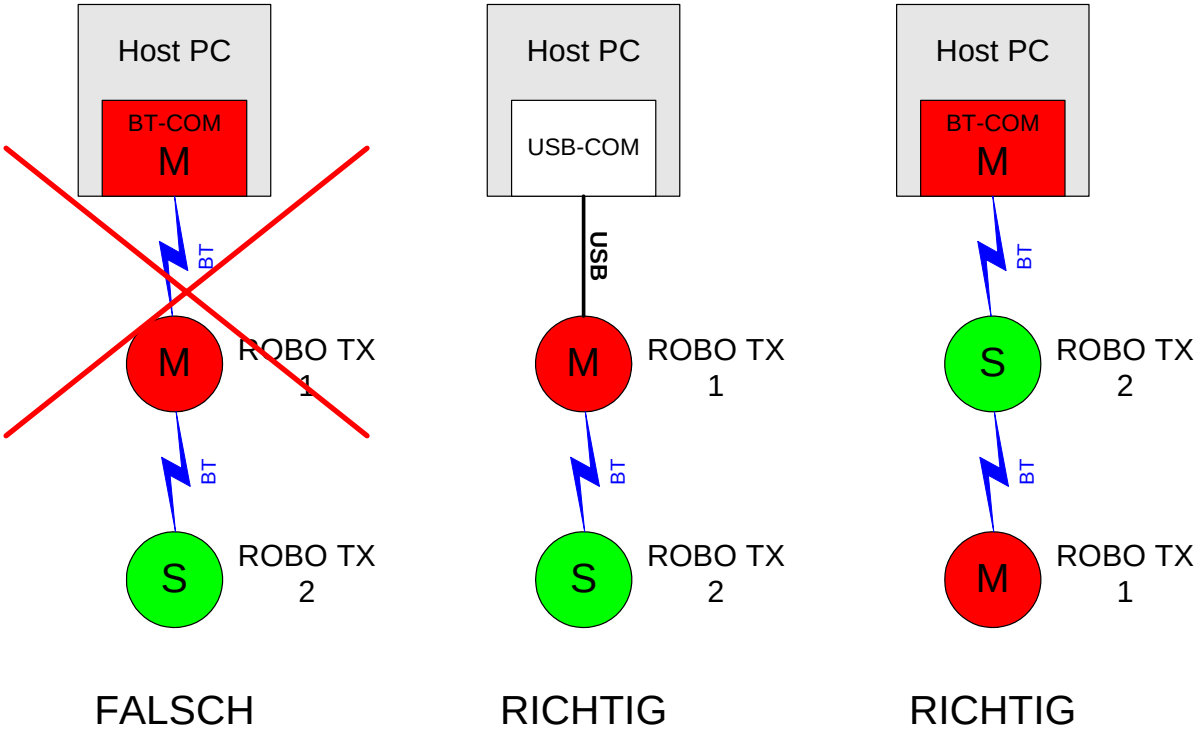


Abb. 6-3

Diese Einschränkung bezüglich des Verbindungsausbaus kommt aus der Spezifikation eines Bluetooth *Piconets*, das als Topologie mit einem MASTER und 1-n SLAVEs definiert ist. Eine Netzwerkverbund hingegen, in welchem mehrere *Piconets* aneinander gekoppelt sind, nennt man *Scatternet*. Hierbei müssen einige Teilnehmer gleichzeitig sowohl MASTER als auch SLAVE sein. Dies wird jedoch von der Firmware im ROBO TX Controller nicht unterstützt.

WICHTIGER HINWEIS
Kommt es versehentlich doch mal zum Master-Wechsel oder zu Doppelrollen (MASTER und SLAVE gleichzeitig) eines Controllers so kann durchaus der Verbindungsaufbau noch klappen. Es sind aber mit hoher Wahrscheinlichkeit Probleme mit dem Versand und Empfang von Nachrichten zu erwarten. Einige Richtungen werden nicht funktionieren.

Möchte man mehrere ROBO TX Controller gleichzeitig im Online Mode betreiben, dann wären von einem PC aus unter Einhaltung des Master/Slave-Prinzips (Piconet-Regeln) auch solche Konstellationen denkbar:

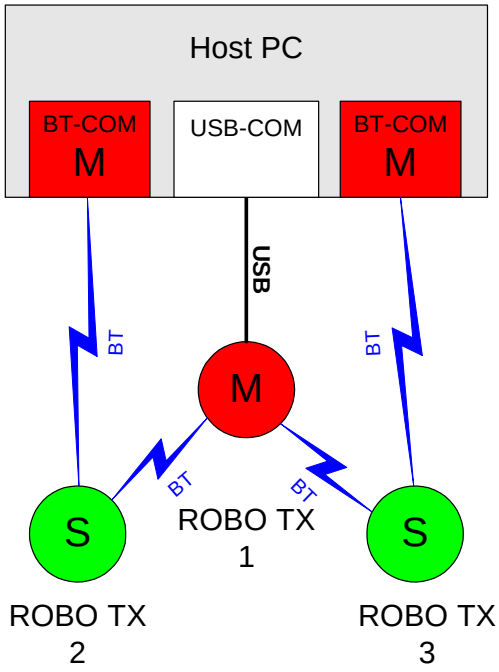


Abb. 6-4

Die Möglichkeit speziell im Online Mode mit mehreren PCs zu arbeiten, steht natürlich ebenfalls offen (es kann nur eine USB-Verbindung pro PC betrieben werden). Diese Variante ist insbesondere zum Einstieg bzw. Anfängern empfohlen, da man sonst in komplexen Szenarien leicht den Überblick über Bluetooth-Verbindungen, Bluetooth-Adressen und COM-Ports verlieren kann:

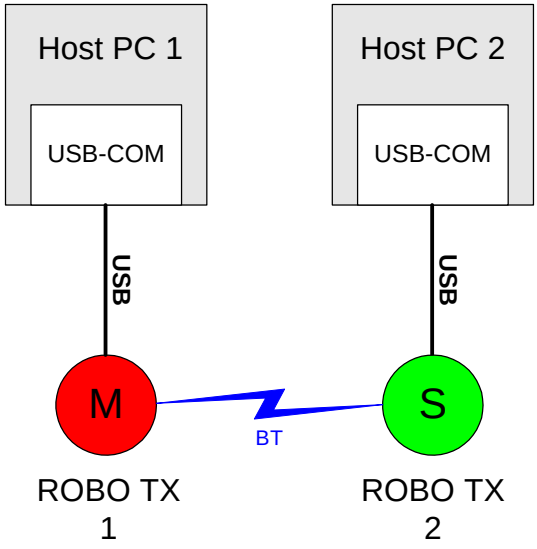


Abb. 6-5

7 I²C-Schnittstelle

Der I²C-Bus (auch IIC oder TWI genannt) ist ein einfacher 2-Draht-Bus zur Inter-Chip-Kommunikation, der von Philips Semiconductor eingeführt wurde. Er erlaubt den einfachen und kostengünstigen Austausch kleiner bis mittlerer Datenmengen auf kurzen Distanzen.

Auf dem I²C-Bus werden Datenbits synchron zu einem Clock-Signal übertragen. Es werden standardmäßig 2 Clock-Geschwindigkeiten angeboten:

„Standard“ 100 KHz Clock
„Fast“ 400 KHz Clock

Die meisten I²C-Geräte arbeiten heutzutage mit 400 kHz. Über die vorhandene Funktions-API (siehe separates Dokument **`Windows_Library_ftMscLib.pdf`**) können I²C-Schreib- und Lese-Operationen ausgeführt werden, wobei die Clock-Geschwindigkeit für jedes Device eingestellt werden kann.

Ein I²C-Gerät kann den Taktzyklus verlängern (Clock Streching). Dies wird von der Firmware automatisch erkannt und unterstützt. Die Callback-Funktionen kehren in diesem Fall ggf. später zurück.

7.1 I²C Hardware-Anschluss

Der ROBO TX Controller sieht eine I²C-Master-Schnittstelle auf dem Extension-Port EXT2 vor. Auf diesem Stecker wird auch ein 5V-Ausgang zur Verfügung gestellt, über die maximal 150mA entnommen werden können. Wird versucht mehr Strom zu entnehmen, werden die 5V durch eine Schutzschaltung abgeschaltet.

Die Daten- und die Clock-Leitungen haben intern im ROBO TX Controller einen 2,2 KOhm Pull-Up-Widerstand.

Die interne Schaltung der ROBO TX Controllers ist hier bildlich dargestellt:

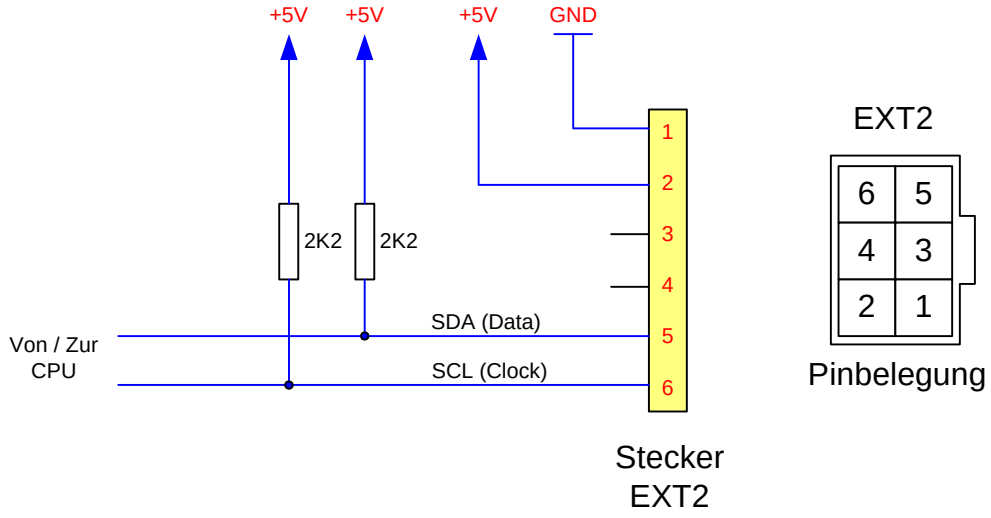


Abb. 7-1: Interne Verdrahtung I²C im ROBO TX Controller

Da die I²C-Schnittstelle eine Bus-Struktur hat können auch mehrere I²C-Geräte gleichzeitig an demselben Bus angeschlossen werden. Die Adressierung erfolgt über die jeweiligen I²C-Geräte-Adressen, die sich in diesem Fall unterscheiden müssen.

8 Beispiel-Anwendungen in C

Im Unterverzeichnis \Demo_Static_Lib_C befinden sich Source-Code-Beispiele für C Programme unter Verwendung der statischen Version der Library. Damit ist es möglich die Library-Funktionen des ROBO TX Controllers in ein und dasselbe ausführbare Programm (executable) zu integrieren. Die Beispiele stehen jeweils als MS Visual C++ V 6.0 Projekt zur Verfügung.

Die Beispiele zeigen einfache Anwendungen zur Steuerung der ROBO TX Controllers:

LightRun Beispiel zur Verwendung der Einzelausgänge O1 bis O8, etwa zur Ansteuerung von Lampen. Das Programm beendet sich nach kurzem Ablauf selbstständig.
MotorRun Einfache Motoransteuerung (verschiedene Geschwindigkeiten) auf Ausgang M1. Das Programm beendet sich nach kurzem Ablauf selbstständig.
StopGo Beispielprogramm zur Ansteuerung eines Encoder-Motors. Dabei werden Zählimpulse gelesen und bei einem Zählerstand von >= 1000 wird das Programm beendet. Der Motor wird über den Digitaleingang I8 gestartet (=1) oder gestoppt (=0). Das Programm beendet sich nach Erreichen des Zählerstandes 1000 selbstständig.
TrafficLights Einfache Ampelsteuerung auf O1 – O3. Das Programm beendet sich nach kurzem Ablauf selbstständig.

Alle Beispiele greifen beim Erstellen auf Dateien in den gemeinsam verwendeten Unterverzeichnissen \Common\Lib und \Common \Inc zu.

Die erstellten ausführbaren Programme (Executables) werden auf der Kommandozeile gestartet.

Beispiel für den Aufruf:

Starten des Programms LightRun durch Aufruf von RUN.BAT im Verzeichnis
`\Demo_Static_Lib_C\LightRun\Release`

Der vom ROBO TX Controller verwendete USB-COM-Port wird jeweils automatisch ermittelt.

9 Beispiel-Anwendungen in C++

Im Unterverzeichnis \Demo_Dll_C++ befinden sich Source-Code-Beispiele für C++ Programme, die die dynamische Library ftMscLib.dll verwenden. Die Beispiele stehen jeweils als MS Visual C++ V 6.0 Projekt zur Verfügung.

Die Beispiele zeigen einfache Anwendungen zur Steuerung der ROBO TX Controllers:

LightRun – Lampen Beispiel zur Verwendung der Einzelausgänge O1 bis O8, etwa zur Ansteuerung von Lampen. Das Programm beendet sich nach kurzem Ablauf selbstständig.
MotorStop – Ultraschallsensor und Encoder-Motor Beispielprogramm zur Ansteuerung eines Encoder-Motors auf M1 (Encoder-Ausgang auf C1 respektive). Dabei wird der Wert eines Ultraschallsensors auf I1 ausgewertet. Ohne „gesehene" Hindernisse dreht der Motor M1. Unterschreitet der gemessene Wert die Grenze von 15, so wird der Motor gestoppt. Bei jedem Stopp des Motors wird der erreichte Zählerstand vom Zählereingang C1 nach dem Start ausgegeben. Der Zählerstand des Zählimpulses wird danach wieder auf 0 (Reset) gesetzt, für die nächste Messung. Das Programm wird mit <Ctrl>+<C> im PC-Fenster beendet.
WarningLight – Ultraschallsensor und Lampe Beispielprogramm zur Ansteuerung einer Lampe auf O8 mit Distanzmessung über einen Ultraschallsensor auf I1. Die Lampe blinkt dabei in unterschiedlichen schnellen Intervallen, je kleiner der gemessene Abstand über den Ultraschallsensor ist. Das Programm wird mit <Ctrl>+<C> im PC-Fenster beendet.
MotorEx_2M_Master – 2 Encoder-Motoren synchronisieren Beispielprogramm zur Ansteuerung zweier Encoder-Motoren auf M1 und M2 (Encoder-Ausgänge auf C1 und C2 respektive) und zum synchronen Betrieb der beiden Motoren. Beide Motoren fahren zyklisch je 200 Schritte hin und her. Bremst man einen Motor während des Laufens (z.B. durch einen mechanischen Widerstand) ab, so passt der jeweils andere Motor seine Geschwindigkeit entsprechend an. Das Programm wird mit <Ctrl>+<C> im PC-Fenster beendet.
MotorEx_Ext1 – Encoder-Motor auf Extension-Controller betreiben Beispielprogramm zur Ansteuerung eines Encoder-Motors auf M1 (Encoder-Ausgang auf C1 respektive) auf einem Extension-Module (Slave Extension 1). Der Motor fährt zyklisch je 200 Schritte hin und her. Das Master-Modul ist das jenige, was am PC angeschlossen ist. Das Programm wird mit <Ctrl>+<C> im PC-Fenster beendet.
StopGoBtButtonPart und **StopGoBtMotorPart** – Bluetooth-Messaging Siehe detaillierte Beschreibung weiter unten in Kapitel 9.1.

I2cTemp – I²C-Temperatursensor DS1631

Beispielprogramm zur Ansteuerung eines externen Temperatursensors DS1631 mit I²C-Schnittstelle (Conrad Electronic Best.-Nr. 19 82 98).

Nach einer kurzen Initialisierungssequenz wird der aktuelle Temperaturwert alle 0,5 Sekunden ausgelesen und am PC-Bildschirm ausgegeben.

Das Programm wird mit einem beliebigen Tastendruck beendet.

Hinweis: andere Conrad-Sensoren aus der C-Control-Reihe können ebenfalls verwendet und direkt an den ROBO TX Controller angeschlossen werden. Der I²C-Anschluss ist kompatibel.

I2cTpa81 – I²C-Infrarot-Temperatursensor-Zeile TPA81

Beispielprogramm zur Ansteuerung einer externen Temperatursensor-Zeile TPA81 mit I²C-Schnittstelle (erhältlich z.B. über www.roboter-teile.de). Dieser erlaubt eine berührungslose Temperaturmessung und über seine Zeilenstruktur (8 Pixel) auch ein einfache Ortsbestimmung von Objekten die Wärme abstrahlen.

Nach einer kurzen Initialisierungssequenz wird der Wert der Umgebungstemperatur sowie aller 8 Messstellen (Pixel) in einer Zeile am PC-Bildschirm ausgegeben. Bewegt man eine Wärmequelle vor der Linse so kann ein Anstieg des Wertes, dessen Pixel die Wärmequelle im Fokus hat, beobachtet werden. Die Anzeige wird alle 0,5 Sekunden aktualisiert.

Das Programm wird mit einem beliebigen Tastendruck beendet.

Alle Beispiele greifen beim Erstellen auf Dateien in den gemeinsam verwendeten Unterverzeichnissen \Common\Lib und \Common \Inc zu.

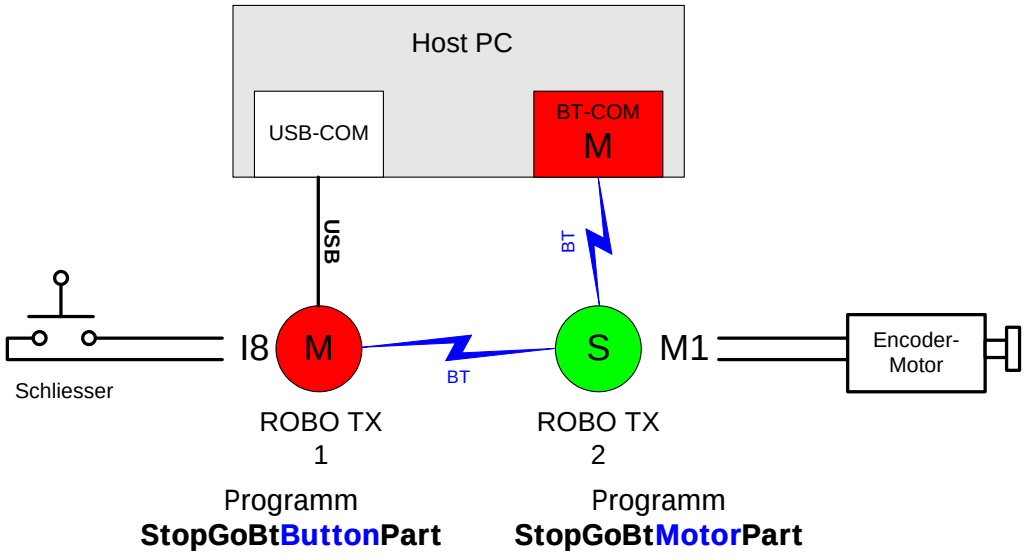
Die erstellten ausführbaren Programme (Executables) werden auf der Kommandozeile gestartet. Beispiel für den Aufruf:

Starten des Programms LightRun durch Aufruf von RUN.BAT im Verzeichnis
`\Demo_Dll_C#\LightRun\Release`

Der vom ROBO TX Controller verwendete USB-COM-Port wird jeweils automatisch ermittelt.

9.1 Beispiel mit 2 Controllern und Bluetooth Messaging

Im nachfolgenden Beispiel wird das Austauschen von Bluetooth-Nachrichten zwischen zwei Controllern mit C-Programmen demonstriert. Dazu ist z.B. folgender Aufbau erforderlich:



Der MASTER-Controller (ROBO TX 1, auf dem Bild links) ist per USB an den PC angebunden, der SLAVE-Controller (ROBO TX 2, auf dem Bild rechts) hingegen per Bluetooth. Dazu muss der SLAVE-Controller so auf dem PC eingerichtet werden, dass er einer COM-Port-Nummer auf dem PC zugeordnet wird. Diese wird in RUN_BT.BAT im Verzeichnis `Demo_Dll_C++\StopGoBtMotorPart\Release\` eingetragen (z.B. COM111). Damit kann der SLAVE-Controller über Bluetooth im Online-Mode arbeiten.

Auf dem MASTER-Controller, der die Bluetooth-Verbindung aktiv aufbaut, läuft das Programm StopGoBtButtonPart, auf dem SLAVE-Controller, der darauf wartet eine Bluettooth-Verbindung anzunehmen, läuft das Programm StopGoBtMotorPart.

Vor dem Start der Programme sind die Bluetooth-Adressen der Controller in die Datei BT_addr.c im Verzeichnis `Common\C\` wie folgt einzutragen (hier beispielhaft):

```
UCHAR8 bt_address_table[BT_CNT_MAX][BT_ADDR_LEN] =
{
   {0x00, 0x13, 0x7B, 0x53, 0x10, 0xE7}, // Bluetooth-Adresse von ROBO TX 1 (MASTER)
   {0x00, 0x13, 0x7B, 0x52, 0xB2, 0x11}, // Bluetooth-Adresse von ROBO TX 2 (SLAVE)
};
```

Anschließend sind die Programme neu zu erzeugen, damit die richtigen Bluetooth-Adressen in diese übernommen werden.

Gestartet werden die Programme dann wie folgt:

Zuerst das Programm StopGoBtMotorPart im Verzeichnis
`Demo_Dll_C++\ StopGoBt MotorPart\Release\`
durch Aufruf von RUN_BT.BAT.

Anschließend das Programm StopGoBtMotorPart im Verzeichnis
`Demo_Dll_C++\StopGoBtButtonPart\Release\`
durch Aufruf von RUN.BAT.

Betätigt man nun den Taster I8 auf ROBO TX 1 (MASTER), dann schickt das Programm StopGoBtButtonPart eine Bluetooth-Nachricht zu ROBO TX 2 (SLAVE) und der Motor auf diesem beginnt sich zu drehen. Lässt man den Taster wieder los, so stoppt der Motor. Das Programm StopGoBtMotorPart auf ROBO TX 2 schickt wiederum die Position des Encoder-Motors per Bluetooth-Nachricht an ROBO TX 1 zurück. Ist eine Position von 1000 erreicht so stoppt das Programm StopGoBtButtonPart und baut die Bluetooth-Verbindung zwischen ROBO TX 1 und ROBO TX 2 wieder ab.

10 Beispiel-Anwendungen in C#

Im Unterverzeichnis \Demo_Dll_C# befinden sich ein Source-Code-Beispiel für C# Programme, das die dynamische Library ftMscLib.dll verwendet. Das Beispiel steht als MS Visual Studio 2008 Projekt zur Verfügung.

Das Beispiel zeigt eine einfache Anwendung zur Steuerung der ROBO TX Controllers:

MotorStop Ultraschallsensor, Encoder-Motor und Menütasten

Beispielprogramm zur Ansteuerung eines Encoder-Motors auf M1. Dabei wird der Wert eines Ultraschallsensors auf I1 ausgewertet. Ohne „gesehene“ Hindernisse dreht der Motor M1. Unterschreitet der gemessene Wert die Grenze von 15 cm, so wird der Motor gestoppt. Bei jedem Stopp des Motors wird der erreichte Zählerstand vom Zählereingang C1 nach dem Start ausgegeben. Der Zählerstand des Zählimpulses wird danach wieder auf 0 (Reset) gesetzt, für die nächste Messung.

Das Programm wird mit <Ctrl>+<C> im PC-Fenster oder durch einen beliebigen Tastendruck auf dem ROBO TX Controller beendet.

Starten des Programms durch Aufruf von RUN.BAT im Verzeichnis
`\Demo_Dll_C#\MotorStop\MotorStop\bin\Release\`

Der vom ROBO TX Controller verwendete USB-COM-Port wird automatisch ermittelt.

11 Multi-Controller-Betrieb

Um größere Modelle zu bauen ist es u.U. erforderlich mehrere ROBO TX Controller zu koppeln. Dies ist über die sogenannten „Extension-Ports“ möglich. Sowohl physisch als auch auf der Befehlsebene wird dies durch eine Master/Slave-Struktur realisiert. In einer Kette von gekoppelten Interfaces gibt es immer einen Master und bis zu 8 Slaves, die vom Master ferngesteuert werden.

Jedes TX-C besitzt 2 Extension Ports, die einen seriellen Erweiterungsbus darstellen. Um die Interfaces zu koppeln ist lediglich mit einem 6-poligen Flachbandkabel (1:1 durchverbunden) von einem Interface zum jeweils benachbarten Interface eine Verbindung herzustellen. Dadurch wird eine Datenverbindung vom Master zu jedem der Slaves realisiert, wie nachfolgend dargestellt.

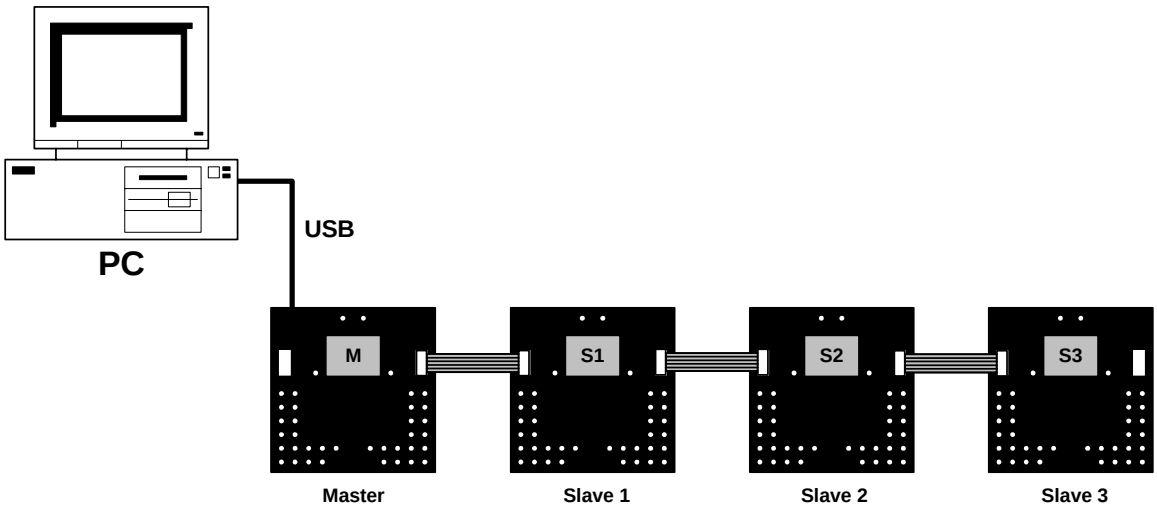


Abb. 2: Kopplung von Interfaces für den Multi-Interface-Betrieb

Bei dem Erweiterungsbus handelt es sich um eine RS485-Schnittstelle mit Abgriffen auf jedem Interface. Dadurch, dass jedes Interface 2 physische Anschlüsse hat und der Erweiterungsbus zwischen beiden Anschlüssen auf dem Interface durchverbunden ist, ist eine einfache Verkabelung benachbarter Interfaces möglich, ohne dass es T-Busverbinder bedarf, und dennoch wird eine klassische Busstruktur erreicht.

Die Kommunikation auf dem Erweiterungsbus ist ebenfalls Master/Slave-orientiert. Der Master pollt dabei alle angeschlossenen Slaves zyklisch ab und kann so I/O-Information austauschen. Damit jeder Slave vom Master individuell und eindeutig angesteuert werden kann, benötigt jeder Slave eine eigene Adresse. Der Modus (Master oder Slave) so wie die Slave-Adresse kann über das Konfigurationsmenü („Einstellungen“) eingestellt werden.

Die Slaves werden vom Master im Online-Modus so gesteuert, wie ein einzelnes Board vom PC aus im Online-Modus gesteuert wird. Der Master ist hierbei der einzige, der mit dem PC kommunizieren kann. Erfolgt im Online-Modus eine Anfrage oder ein Befehl vom PC an einen der Slaves, so wird diese Anfrage vom Master an den entsprechenden Slave weitergeleitet.

12 Update der ROBO TX Controller Firmware

Die Firmware des ROBO TX Controllers kann auf mehrere Arten ein Update erfahren:

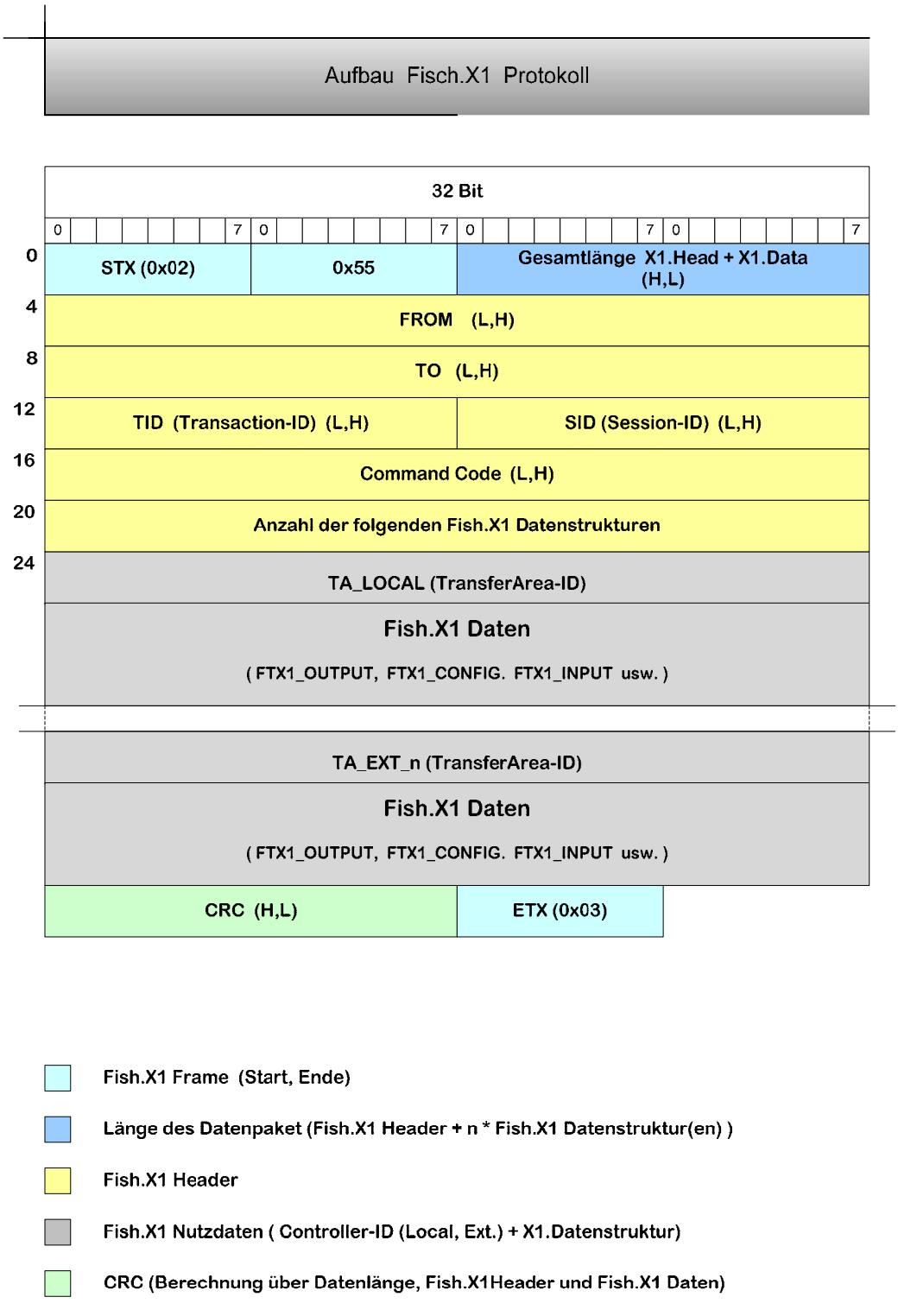
- Über ROBOPro (wird i.d.R. automatisch beim ersten Verbindungsaufbau zum TX-C angeboten, soweit ein Update erforderlich ist)
- Über die Funktion „Firmware Update“ in RoboTxTest (siehe Kapitel 4.5). Hierbei ist auf das in diesem Packet mitgelieferte Unterverzeichnis **`/Firmware`** zu verweisen.
- Aus einer selbstgeschriebenen PC-Applikation heraus mit Hilfe der Library-Funktion „RoboTxFwUpdate“ (s. auch Dokument **`Windows_Library_ftMscLib.pdf`**).
- Mit dem Repair-Tool (siehe separates Paket mit entsprechender Bezeichnung)

13 Low-Level-API (COM Port Level)

Die Low-Level-API verwendet den jeweiligen COM-Port des ROBO TX Controllers selbst (USB oder Bluetooth). Die 3 Unterschnittstellen sind nachfolgend beschrieben.

13.1 Fish.X1-Schnittstellenprotokoll

Nachfolgend wir der Aufbau und der Inhalt eines Fish.X1 Datenpakets zwischen der PC-Library und dem ROBO-TX Controller beschrieben.



Beschreibung und Inhalt eines Datenpakets

Offset	Länge	Inhalt und Bedeutung
0	1	STX (0x02) Markiert mit dem nachfolgenden Byte den Anfang eines Fish.X1 Datenpakets
1	1	0x55
2	2	Länge (H,L) des Datenpakets Setzt sich aus der Länge des Headers (20 Byte) und der Länge der Nutzdaten zusammen.
4	4	Bus-Adresse, 4 Byte (L,H), <FROM> (Sender des Datenpakets) Mögliche Werte: BUS_ADR_WILDCARD=1, BUS_ADR_MASTER=2, BUS_ADR_SLAVE1=3 ... BUS_ADR_SLAVE8=10
8	4	Bus-Adresse, 4 Byte (L,H) <TO> (Empfänger des Datenpakets) Mögliche Werte s.o.
12	2	**T**ransaction-**ID** (TID), 2 Byte (L,H) Jedes Datenpaket wird vom Sender mit einer fortlaufenden Nummer markiert, die Antwort auf einen Request wird vom Empfänger der Nachricht mit der gleichen TID zurückgesendet.
14	2	**S**ession-**ID** (SID), 2 Byte (L,H) Das ROBO-TX Interface verwaltet intern eine Session-ID, die bei jedem Taskwechsel von Lokal auf Remote, bzw. von Remote nach Lokal neu vergeben wird. Z.B. wird die Session-ID von der Firmware neu vergeben, falls es bei der Kommunikation zu einem Timeout kommt. Der Sender kann dann entsprechend reagieren.
16	4	Fish.X1 Command Code (Tele-Code), 4 Byte (L,H) Definiert einen eindeutigen Request- bzw. Reply-Code. Der Command-Code beschreibt damit die mitgelieferte Datenstruktur in den Nutzdaten.
20	4	Anzahl der folgenden Fish.X1 Datenstrukturen, 4 Byte (L,H). (= n)
24	4	**TransferArea-ID**, 4 Byte (L,H) Die ID legt fest, für welchen ROBO TX Controller, und damit eine adressierte TransferArea aus der gesamten TransferArea, die nachfolgende Datenstruktur bestimmt ist. Die über den Command Code festgelegte Datenstruktur aus der individuellen TransferArea wird aktualisiert. Die Interfaces können als Master (immer lokal) oder als Slaves (an einem Master) konfiguriert sein. Eine entsprechende Kennung wird hier über die TransferArea-ID angegeben: TA_LOCAL=0, TA_EXT_1=1, TA_EXT_2= 2 usw. Der Master (TA_LOCAL) verwaltet die IO-Werte der einzelnen Slaves in seiner TransferArea und kann somit die aktuellen IO-Daten anhand der ShareMemory-ID zuordnen.
28	m	Nutzdaten der Nachricht, abhängig von der gesendeten Datenstruktur. Zusammen mit der ShareMemory-ID können die Nutzdaten auch eine Länge 0 haben, üblicherweise eine Anfrage oder ein Reply auf einen Request. Als Nutzdaten werden die in der Headerdatei ROBO_TX_FW.H definierten Strukturen eingesetzt. Der Command-Code identifiziert die mitgelieferte Datenstruktur.
24 + n*(4+m)	2	CRC, 2 Byte (H,L) Der CRC wird ab Byte 2 (Datenlänge) bis zum Ende der Nutzdaten berechnet.
24 + 2 n*(4+m)	1	ETX (0x03) Markiert das Ende einer Fish.X1 Nachricht

13.2 Remote Shell

Die Remote Shell (oder Remote Console) ermöglicht es Dateioperationen im Dateisystem des ROBO TX Controllers auszuführen, sowie andere Informationen abzurufen. Die Befehle der Remote Shell sind alle ASCII basiert und können direkt von einem Terminalprogramm wie Windows HyperTerminal eingegeben werden. In der Regel gibt es zu jedem Befehl eine Antwort.

Besteht eine aktive Verbindung zum ROBO TX Controller erscheint auf Drücken der <RETURN>-Taste eine Eingabeaufforderung (Prompt), die den Hostnamen des angesprochenen ROBO TX Controllers und die Schnittstelle anzeigt.

Beispiel: **ROBO TX-511\USB>** für USB
oder: **ROBO TX-511\Bluetooth>** für Bluetooth

Der Befehlsvorrat der Remote Shell ist wie folgt:

Kommando	Beschreibung
?	Gibt die Liste der gültigen Befehle aus
help	Gibt die Liste der gültigen Befehle aus
version	Gibt die aktuelle Versionsnummer der Firmware aus
get_ser_num	Gibt die Seriennummer des ROBO TX Controllers aus
get_board_ver	Gibt die Kennung der Hardware-Revision aus
get_eeprom	Gibt eine strukturierte Darstellung der Board-Daten aus
clean_disk	Bereinigt das angegebene Laufwerk von nicht zwingend benötigten Dateien
erase_disk	Löscht das angegebene Laufwerk des ROBO TX Controllers[1)]
erase_flashdisks	Löscht alle Flash-Laufwerke des ROBO TX Controllers[1)]
erase_boot	Löscht den Urlader des ROBO TX Controllers[1)]
erase_phase0	Löscht den Bootloader des ROBO TX Controllers[1)]
dir	Zeigt den Verzeichnisinhalt an
type	Gibt den Inhalt einer Textdatei aus
del	Löscht die angegebene Datei
copy	Kopiert die angegebene Datei
format	Formatiert das angegebene Laufwerk
ren	Benennt die angegebene Datei um
xrecv	Löst den Dateiempfangsvorgang auf dem ROBO TX Controller aus
load	Lädt ein ROBO-Programm von einem der Laufwerke in den Programmspeicher, damit dieses ausgeführt werden kann.
run	Startet das aktuelle im Programmspeicher befindliche ROBO-Programm.
stop	Beendet das aktuell laufende ROBO-Programm.

1) Diese Befehle sind nur zu verwenden, wenn man genau weiß, was man tut. Sie können andernfalls dazu führen, dass der ROBO TX Controller nicht mehr startet. In diesem Fall ist die Firmware mit dem „Repair-Kit“ wieder herzustellen.

13.3 X-Modem File-Transfer

Mit Hilfe der X-Modem File-Transfer-Schnittstelle können Dateien (u.a. ROBO-Programme) vom PC auf den ROBO TX Controller übertragen werden. Die andere Richtung (Übertragung vom TX-C zum PC) ist nicht vorgesehen.

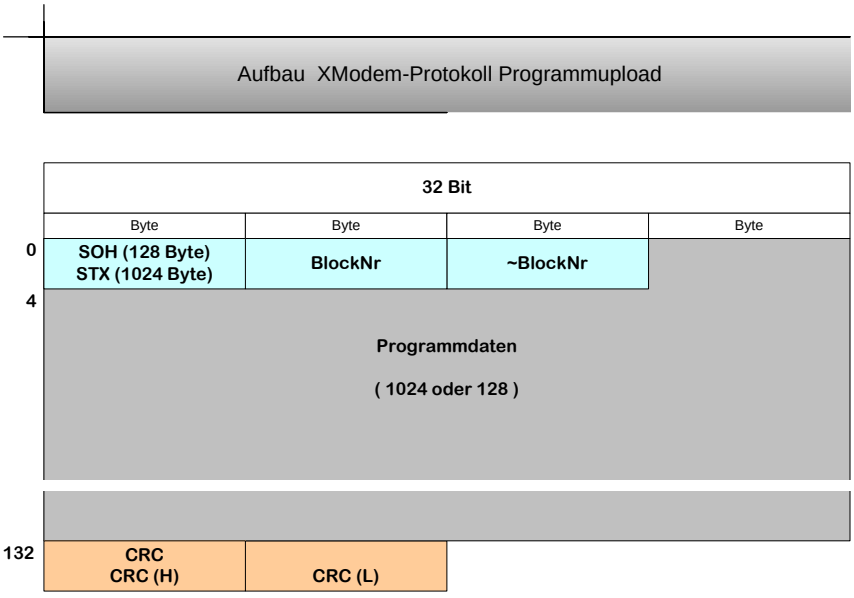
Für einen Dateidownload wird das X-Modem-Protokoll mit einer festen Paketlänge verwendet, welches eine gesicherte Datenübertragung regelt. Das X-Modem-Protokoll arbeitet dabei blockorientiert, wobei die zu übertragenden Daten in gleich große Einheiten (Blöcke) aufgeteilt werden. Die Blöcke haben immer eine Größe von 1029, bzw. 1028 oder 133, bzw. 132, je nachdem wie die Länge des CRC festgelegt wurde und werden gegebenenfalls mit beliebigen Zeichen aufgefüllt. Quittungen auf gesendete Datenblöcke bestehen aus einem einzigen Zeichen.

Die Übertragung (Download) wird durch den Empfänger (ROBO-TX Controller) angestoßen, indem dieser ein NAK oder ein 'C' auf die Anfrage

xrecv /<device>/"<file>" <length>

sendet. Bei einem NAK wird die Länge des CRC mit einem Byte festgelegt, sendet der Empfänger ein 'C' als Bestätigung, wird die CRC-Länge mit 2 Byte vereinbart. Eine fehlerfreie empfangene Checksumme wird mit einem ACK (Acknowledge) vom Empfänger bestätigt. Bei einem aufgetretenen Fehler wird der gesendete Block mit NAK (Negative Acknowledge) abgelehnt und dann bis zu zehnmal neu versendet. Das Ende einer Übertragung wird vom Sender mit EOT (End of Transmission) angezeigt. Auch dieses muss mit einem ACK bestätigt werden. Die Prüfsumme ist die arithmetische Summe der Datenbytes modulo 256.

Der in der Anfrage anzugebende Dateiname ist mit dem Zeichen " zu begrenzen und kann somit auch Leerzeichen enthalten. Weiterhin ist die Länge der Datei in der Anfrage mit anzugeben. Mit der Angabe <device> wird festgelegt, auf welchem Ziellaufwerk die Datei gespeichert werden soll. Als Ziellaufwerk stehen die Ramdisk und die Flashdisk zur Verfügung, anzugeben ist dabei die Zeichenfolge "ramdisk" oder "flash".



Beschreibung und Inhalt eines Datenpakets beim Upload (XModem-Protokoll)

Offset	Länge	Inhalt und Bedeutung
0	1	SOH (0x01), Start Of Heading Wird eingetragen, wenn die Blockgröße 128 Byte beträgt STX (0x02), Start Of Text Wird eingetragen, wenn der Datenblock 1024 Byte enthält
1	1	Laufende Blocknummer, beginnend mit 0
2	1	Einer-Komplement der Blocknummer
3	128 oder 1024	Datenblock, bestehend aus den Daten der zu übertragenen Datei oder des Programms
131	1	CRC, arithmetische Prüfsumme über die Datenbytes oder CRC (H) High-Wert der Prüfsumme
132	1	CRC (L), Low-Wert der Prüfsumme

Zusätzlich zu den Quittungen NAK und ACK kann der Empfänger auf einen Datenblock auch mit einem CAN (= Cancel) antworten. Die Datenübertragung ist daraufhin von dem Sender zu unterbrechen und mit einem abschließenden EOT zu beenden.

14 Versionshistorie dieses Dokument

Version	Datum	Autor	Änderungen Bemerkungen
1.0	02.10.2009	Gesamtes Team	Erste Fassung
1.1	09.10.2009	Peter Duchemin	Kapitel 4.5 und Kapitel 10: Hinweis auf mitgeliefertes Unterverzeichnis /Firmware Kapitel 11.1, Kommentar in Strukturbild eingefügt. Fehlendes Feld „Anzahl“ (Offset 20) in Tabelle eingefügt. Kapitel 11.2, Fußnote unter Tabelle eingefügt
1.2	11.12.2009	Peter Duchemin	Nur Versionsnummern auf Deckblatt angepasst
1.4	23.12.2011	Peter Duchemin	Kapitel zur Bluetooth Message API hinzugefügt, Anzahl der Beispiele um eine Bluetooth-Messaging-Demo erhöht.
1.5	24.04.2012	Peter Duchemin	Kapitel zur I^2C-Schnittstelle hinzugefügt, Anzahl der Beispiele um zwei Demos mit I^2C-Sensoren erhöht