fischertechnik ROBO TX Controller

umFish50.DLL

Übersicht von Programmiermöglichkeiten auf Basis der ftMscLib.DLL - umFish50.DLL - FishFaceTX.DLL

Ulrich Müller

Inhaltsverzeichnis

Übersichten

ROBO TX Controller

Allgemein

Ein **Master**, 0 - 8 **Extensions** (baugleich mit Master). Zwei Ext.-Anschlüsse
Text-Display, mehrzeilig + **2 Bedienknöpfe** zur Status-Anzeige und Einstellung, Auswahl von Programmen. Zusätzlich von einer Anwendung (im Download-Modus) nutzbar.
Anschluß über **USB** (Mini-Stecker) an Rechner.
Alternativer Anschluß über **Bluetooth** (soll auch zum Nachrichtenaustausch zwischen Controllern genutzt werden können).
Camera-Anschluß vorh., Nutzung offen.

Sensoranschlüsse

Der TX Controller besitzt 8 Universal-Eingänge, die für den Verwendungzweck konfiguriert werden müssen. Außerdem sind 4 Zählereingänge vorhanden, die wahlweise als "schnelle" Zähler (EncoderMotor) oder als Zähler mit D5K-Sensoren eingesetzt werden können, zusätzlich können sie als einfacher D5K-Eingang genutzt werden.

Universaleingänge :

D5K : Digital 5 kOhm (Taster, Reedkontakt, PhotoTransitor)

D10V : Digital 10 V (SpurSensor)

A5K : Analog 5 kOhm (NTC, Photowiderstand, Potentiometer)

A10V : Analog 10 V (FarbSensor, Spannung allgem.)

Aktorenanschlüsse

Der TX Controller besitzt 8 einpolige Ausgänge (O-Ausgänge), davon können jeweils zwei zu M-Ausgängen zusammengeschlossen werden. Sie ermöglichen dann einen Motorbetrieb

Lieferbare Sensoren und Motoren und deren Anschluß

Sensoren

D5K : Taster

D5K : Reedkontakt

D5K : Phototransistor

D10V : **Spursensor**

Anschluß Kabel rot/grün an Plus und Masse, gelb / blau an zwei verschiedene Universaleingänge

A5K : NTC

A5K : Photowiderstand

A5K : Potentiometer

A10V : **Farbsensor**

Anschluß Kabel rot/grün an Plus und Masse, schwarz an Universaleingang

Dist : Ultraschallsensor

Aktoren

Motoren : XS, XM, Encoder, Power

Lampen : normal, Linse, Hupe, Magnet

Installation

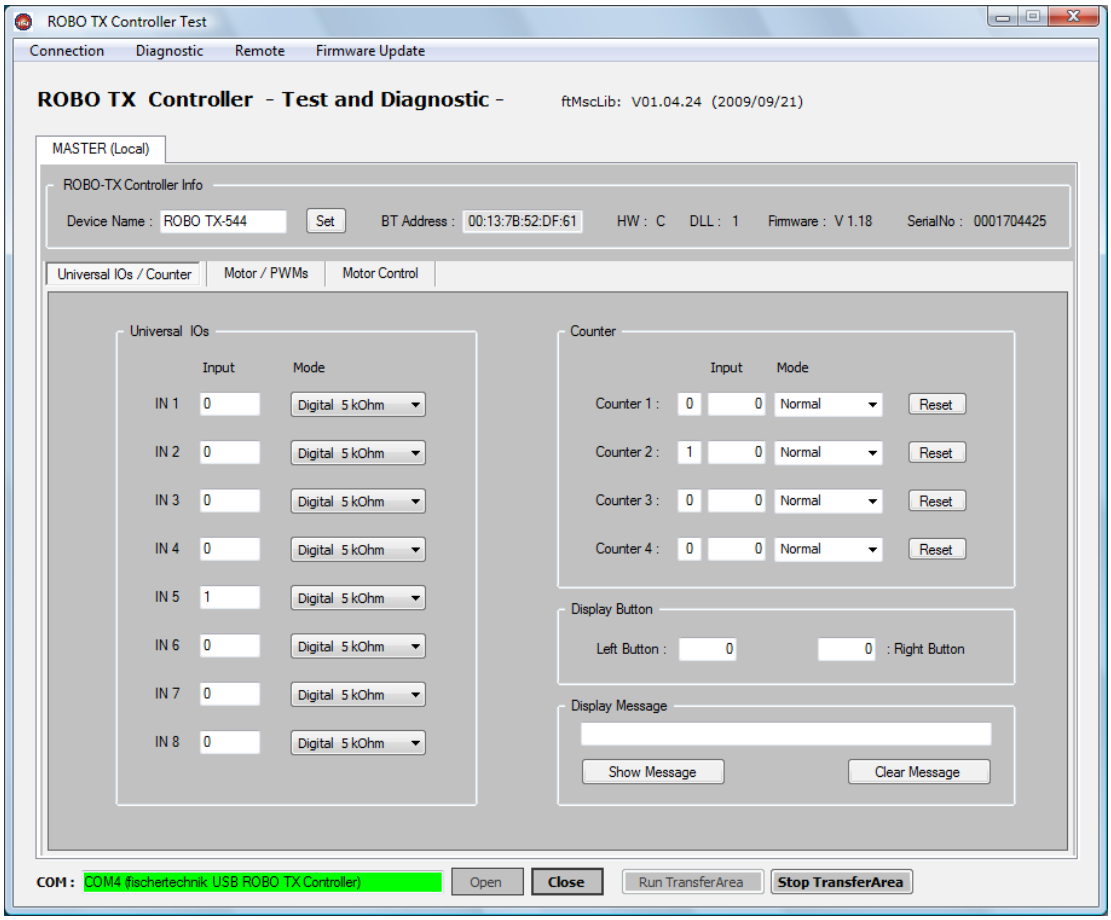
Benötigt werden das Päckchen "**PC Programming ROBO-TXC 1.2**" von www.fischertechnik.de | Computing | Downloads. Dort enthalten ist der benötigte **USB-Treiber** für den TX-Controller(Installtionsbeschreibung in der beiliegenden Dokumentation), die ebenfalls benötigte **ftMscLib.DLL** und das TestTool **RoboTxTest.exe**.

Das Päckchen **umFish50.ZIP** mit der **umFish50.DLL** und den **Deklarationen** zur Nutzung von umFish50.DLL aus verschiedenen Programmiersprachen.

Die im nachfolgenden Text angesprochene Klasse **FishFaceTX** sollte bei Bedarf über eine passendes Päckchen von www.ftcomputing.de/fishdllstx.htm geladen werden.

ftMscLib.DLL und umFish.DLL müssen an einem zentralen Ort (z.B. **\Windows\System32**) plaziert werden.

ROBO TX Test Panel


ROBO TX Controller Test
Connection
Diagnostic
Remote
Firmware Update
ROBO TX Controller - Test and Diagnostic -
ftMscLib: V01.04.24 (2009/09/21)
MASTER (Local)
ROBO-TX Controller Info
Device Name : ROBO TX-544
Set
BT Address : 00:13:7B:52:DF:61
HW : C
DLL : 1
Firmware : V 1.18
SerialNo : 0001704425
Universal IOs / Counter
Motor / PWMs
Motor Control
Universal IOs
Input
Mode
IN 1 0 Digital 5 kOhm
IN 2 0 Digital 5 kOhm
IN 3 0 Digital 5 kOhm
IN 4 0 Digital 5 kOhm
IN 5 1 Digital 5 kOhm
IN 6 0 Digital 5 kOhm
IN 7 0 Digital 5 kOhm
IN 8 0 Digital 5 kOhm
Counter
Input
Mode
Counter 1 : 0 0 Normal Reset
Counter 2 : 1 0 Normal Reset
Counter 3 : 0 0 Normal Reset
Counter 4 : 0 0 Normal Reset
Display Button
Left Button : 0
0 : Right Button
Display Message
Show Message
Clear Message
COM : COM4 (fischertechnik USB ROBO TX Controller)
Open
Close
Run TransferArea
Stop TransferArea

RoboTxTest.EXE TestPanel aus dem ftMscLib.DLL-Paket. Stand alone Programm

Belegung für Testzwecke :

M1 Motor mit 4er Impulsrad
C1 Taster als Impulszähler
I1 Endtaster

M2 Motor mit 8 Schritten
C2 Taster als Impulszähler
I2 Endtaster

O5 Lampe rot
O6 Lampe gelb
O7 Lampe grün

I3 Taster
I4 Farbsensor
I5 Phototransistor
I6 Photowiderstand
I7 Spursensor links
I8 Spursensor rechts

Bei den Motoren können alternativ auch EncoderMotoren mit gleicher Anschlußbelegung eingesetzt werden. Beim Test ist dann die unterschiedliche Impulszahl zu beachten.

Graphische Programmiersprache ROBO Pro

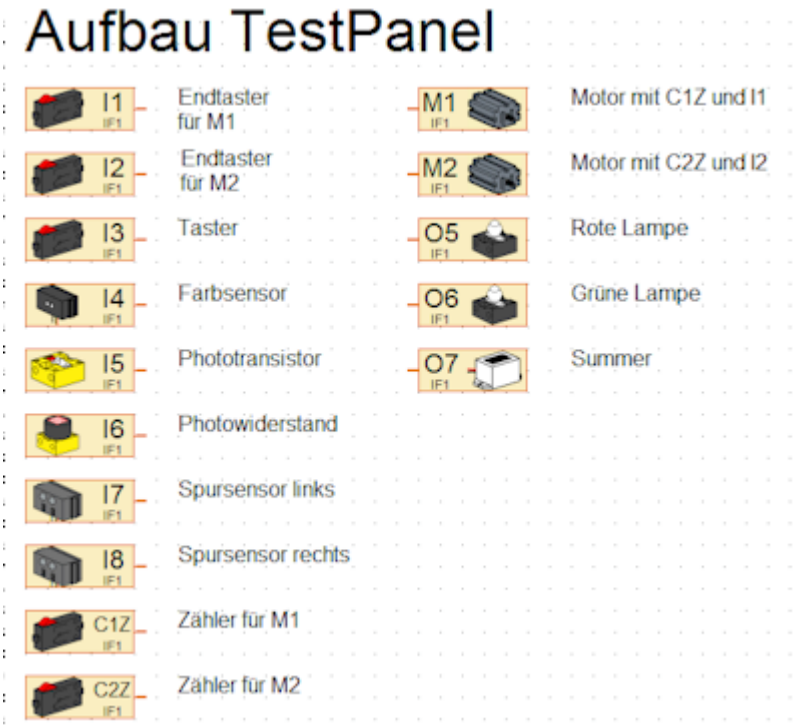
Anschlußumsetzung ROBO Interface -> ROBO TX Controller

In der IDE von ROBO Pro wird unter der Oberfläche TX Controller eine halbautomatische Konvertierung von ROBO Interface Programmen in ROBO TX Controller Programme angeboten. Dazu wird die Anschlußbelegung wie unten angeführt umgestellt :

ROBO Interface	**ROBO TX Controller**
D1 Ultraschall	**I1** Ultraschall, aber nur neuer Sensor
A1 analog 10V	**I2** A10V Farbsensor
AX analog 5 kOhm	**I3** A5K NTC, Photowiderstand, Poti
AY analog 5 KOhm	**I4** A5K
I1 - I4 digital	**I5 - I8**
I5 - I8 digital	**C1D - C4D**
M1 - M4	**M1 - M4** wie bisher
O1 - O8	**O1 - O8** alternativ

ROBO Pro Programm zur Einstellung der Anschlüsse

Bei Arbeiten mit ROBO Pro wird man auch mit dem eingebauten Testpanel arbeiten, dann kann man dessen Belegung schnell mit dem kleinen Programm unten herstellen.



ftMscLib.DLL - genutzte Funktionen

Primär zum Einsatz mit C/C++ zur Entwicklung von DLLs und Tools

Auskunftsfunktionen

ftxGetLibVersionStr : Auslesen der Version der DLL

GetAvailableComPorts : Erzeugen einer internen Liste verfügbarer COM-Verbindungen

EnumComPorts : Auslesen eines Elements der Liste

Verbindungsaufbau TX Controller

ftxInitLib : Initialisierung der Lib

ftxCloseLib : Schließen der Lib

ftxOpenComDevice : Herstellen einer Verbindung zum TX Controller

ftxCloseComDevice : Schließen einer Verbindung zum TX Controller

ftxStartTransferArea : Starten des Austauschbereiches für Controllerdaten

ftxStopTransferArea : Schließen des Austauschbereiches

Universaleingänge

SetFtUniConfig : Konfigurierung der Universaleingänge

GetInIOValue : Auslesen der Werte eines Eingangs

Countereingänge

SetFtCntConfig : Konfigurierung der Countereingänge

GetInCounterValue : Auslesen eines Eingangs

SetOutCounterReset : Zurücksetzen eines Einganges

M / O - Ausgänge

SetOutPwmValues : Setzen (Schalten) der Parameter eines O-Ausganges

SetFtMotorConfig : Konfigurierung eines M-Ausganges

SetOutMotorValues : Setzen (Schalten) der Parameter eines M-Ausganges

umFish50.DLL - List der Funktionen

umFish50.DLL ist vor allem für die Erstellung eigener sprachspezifischer Libraries - wie z.B. FishFache.DLL vorgesehen. umFish50.DLL unterstützt den ROBO TX Controller (und nur diesen) über einen USB-Anschluß oder über Bluetooth im "online" Modus (das Programm läuft auf dem PC).

umFish50.DLL faßt die ftMscLib Funktion mit dem Ziel einer einfacheren Nutzung zusammen:

- MultiMediaTimer
    - Kontrolle von M-Output und C-Inputs. Stop eines Motors, wenn er sein Ziel (Counterwert) oder den Endtaster erreicht hat (RobMotor).
    - Anhalten laufender Motoren über die ESC-Taste.
- Zusammenfassung von ftMscLib.DLL Funktionen zu Sensor/Aktor orientierten Methoden.
- Konfiguration der Universal-Eingänge bei der ersten Nutzung.
- Einfügen von Sleeps in zeitkritische Programmabschnitte.

Alle Methoden geben bei Fehler den gleichen FehlercodetxError zurück.

Parameters

ctrId : Controller ID (0 - 8)

inpNr : Nummer eines Universal-Einganges (0 - 7 (11))

cntNr : Nummer eines Zähler-Einganges (0 - 3)

outNr : Nummer eines O-Ausganges (0 - 7)

motNr : Nummer eines M-Ausganges (0 - 3)

OnOff : Power für einen O-Ausgang (0 - 512)

speed : Speed für einen M-Ausgang (0 - 512)

Common Constants

ctxError Return-Wert : Methode ist fehlgeschlagen

ctxOff, ctxOn, ctxLeft, ctxRight : Schalten der M / O - Ausgänge

ctxFull 512, ctxHalf 444 : Power mit der der Ausgang betrieben werden soll

ctxMain, ctxExt1 : Name des Controllers: Master / Extension (0 / 8)

ctxI1 … : Number of a Universal Input (0 - 8, - 11, if C Inputs are used as switches)

ctxC1 … : Nummer eines Zähler-Einganges (0 - 3)

ctxM1 … : Nummer eines M-Ausganges (0 - 3)

ctxO1 … ; Nummer eines O-Ausganges (0 - 7)

Verbindung zum TX Controller

int **txOpenController**(ComNr)
Herstellen einer Verbindung zum genutzen Master TX Controller
ComNr : Nummer der COM-Verbindung. Kann über RoboTxTest.EXE ermittelt werden. Ebenso über den Geräte-Manager von Windows. Dazu muß der TX Controller angeschlossen sein, die COM-Nummer kann auch über den Geräte-Manager geändert werden

int **txCloseController**()
Close the existing connection to the TX Controller

Universal Eingänge

Die Universal-Eingänge werden in Übereinstimmung mit der ersten sie nutzenden Methode konfiguriert (nach einem txOpenController).

int **txGetAnalog**(ctrId, inpNr)
Lesen des aktuellen Analog-Wertes von einem Universal-Eingang der als A5K konfiguriert wurde (NTC, Photowiderstand, Potentiometer)

int **txGetDistance**(ctrId, inpNr)
Lesen des aktuellen Distanzwertes von einem Universal-Eingang der als Dist (Ultraschall-Sensor) konfiguriert wurde

int **txGetInput**(ctrId, inpNr)
Lesen des aktuellen digitalen Zustandes eines Universal-Einganges der als D5K (Taster, Reedkontakt, Phototransistor) konfiguriert wurde.

int **txGetTrack**(ctrId, inpNr)
Lesen des Status einer "Halb"spur eines Spursensors(D10V). TRUE(1) heißt hier

int **txGetVoltage**(ctrId, inpNr)
Lesen des aktuellen Spannungswertes von dem adressierten Universal-Einganges, der als A10V(Farbsensor, Spannung allgem.) konfiguriert wurde.

Zähler-Eingänge

int **txGetCounter**(ctrId, cntNr)
Lesen des aktuellen Zählerstandes des adressierten Zählers

int **txClearCounter**(ctrId, cntNr)
Zurücksetzen des adressierten Zählers auf Null.

M / O Outputs

Ein RobMotor ist ein Encoder-Motor mit zugeordnetem Taster an einem Zähler-Eingang mit der gleichen Nummer wie der entsprechende Motor-Ausgang. Alternativ ist ein Motor plus Impulstaster (Nummer wie gehabt). Beiden Motorarten ist zusätzlich ein Endtaster zugeordnet, der bei Linksdrehung ausgewertet wird.

int **txSetLamp**(ctrId, outNr, OnOff)
Schalten eines O-Ausganges auf den Powerwert von OnOff(0 (off) to 512)

int **txSetMotor**(ctrId, motNr, direction, speed)
Schalten eines M-Ausganges auf die Drehrichtung left, right, off und einen Speed-Wert (0 - 512)

int **txStartMotor**(ctrId, motNr, direction, speed, iCount, robMode)
Schalten eines RobMotors auf die Drehrichtung left, right, off und einen Speed-Wert (0 - 512) für die Dauer von iCount Impulsen. Der Motor wird asynchron abgeschaltet, wenn die Impulszahl oder der Endtaster erreicht wurden.

int **txIsMotorReady**(ctrId, motNr)
Status eines Motors, der mit txStartMotor gestartet wurde.
Rückgabewert -1, wenn der Motor noch läuft, sonst die Anzahl der über die Vorgabe hinaus gefahrenen Impulse (0 - …).

System Funktionen

void **txSleep**(mSek)
Anhalten des aktuellen Threads für die angegebene Anzahl von Millisekunden.

int **txEsc**()
Status der ESC-Taste (0 : nicht gedrückt)

int **txGetTickCount**()
Lesen der aktuellen TickCounts (in Millisekunden) nach Mitternacht.

javaFish50.DLL - Liste der Funktionen

Primär zum Einsatz zur Erstellung javaspezifischer Klassenbibliotheken

javaFish50.DLL ist eine Wrapper.DLL speziell für Java (JNI Konvention), die umFish50.DLL 1:1 kapselt.

Verbindungsaufbau TX Controller

jxOpenController : Herstellen einer Verbindung zum TX Controller

jxCloseController : Schließen einer bestehenden Verbindung zum TX Controller

Universaleingänge

Auslesen des aktuellen Eingangswertes, beim ersten Zugriff wird der Anschluß auf die dem Zugriffsbefehl entsprechende Eingangsart fliegend konfiguriert. Ab da gilt dann diese Konfigurierung und die weiteren Zugriffe laufen dann deutlich schneller

jxGetAnalog : A5K-Eingang (NTC, Photowiderstand, Potentiometer)

jxGetDistance : Dist-Eingang (UltraschallSensor)

jxGetInput : D5K-Eingang (Taster, Reedkontakt, PhotoTransistor)

jxGetTrack : D10V-Eingang (SpurSensor)

jxGetVoltage : A10V-Eingang (FarbSensor, Spannung allgem.)

Countereingänge

jxGetCounter : Auslesen des aktuellen Counterstandes

jxClearCounter : Zurücksetzen des Counters

M / O - Ausgänge

jxSetLamp : Schalten eines O-Ausganges

jxSetMotor : Schalten eines M-Ausganges

jxStartMotor : Starten eines "Encoder"-Motors für die angegebene Impulszahl, der Motor wird bei Erreichen des Ziels asynchron abgeschaltet.

jxIsMotorReady : Abfrage, ob Motorziel erreicht ist (läuft noch : -1, 0 und größer abgeschaltet mit Angabe der zusätzlich gefahrenen Impulse)

Systemfunktionen

sleep : Warten für MilliSekunden
escape : Abfrage, ob ESC-Taste gedrückt wurde
getTickCount : Verstrichene Zeit in Ticks seit Mitternacht.

FishFaceTX.DLL - Klasse FishFace

Zum Einsatz in der Anwendungsprogrammierung. Es sind mehrere Sprachversionen mit gleichem Funktionsangebot vorgesehen. Sie bauen auf den Funktionen der umFish50.DLL auf. Hinzugekommen ist die Unterbrechbarkeit von Windows-Programmen durch Einfügen von DoEvents Befehlen, die Abbrechbarkeit länger laufender Methoden ist durch Einfügen von Abfragen auf NotHalt bzw. die ESC-Taste gegeben. Durch den TX Controller erkannte Fehlersituationen werden durch Auslösen von Exceptions angezeigt. Die Parameter werden in Form von enum's übergeben. Für die Schreibvereinfachung gibt es eine Reihe von Überladungen. So ist der Parameter für den Controller stets optional (es gilt dann Ctr.Main)

Implementierung z.Zt. in C# 2005 für VB.Net und C# 2005 / 2008 und VB 2005 / 2008. Sowie in Java 6 in der Eclipse 3.4 Umgebung und für Python 3.1.1

Die jeweilig erforderliche FishFaceTX sollte von www.ftcomputing.de/fishdllstx.htm als extra Päckchen geladen werden.

Verbindungsaufbau TX Controller

OpenController(ComName) : Herstellen einer Verbindung zum TX Controller

CloseController() : Beenden einer Verbindung zum TX Controller

Hilfsfunktionen

Finish([InputName]) : Feststellen eines Abbruchwunsches

Pause(mSek) : Anhalten des Programmablaufs

Universaleingänge

InputName : enum Unv (I1 … I8)

GetAnalog(InputName) :	A5K-Eingang (NTC, Photowiderstand, Potentiometer)
GetDistance(InputName) :	Dist-Eingang (UltraschallSensor)
GetInput(InputName) :	D5K-Eingang (Taster, Reedkontakt, PhotoTransistor)
GetTrack(InputName) :	D10V-Eingang (SpurSensor)
GetVoltage(InputName) :	A10V-Eingang (FarbSensor, Spannung allgem.)

Countereingänge

CounterName : enum Cnt (C1 … C4)

GetCounter(CounterName) :	Auslesen des aktuellen Counterstandes
ClearCounter(CounterName) :	Zurücksetzen des Counters

M / O - Ausgänge

LampName : enum Out (O1…O8)
MotorName : enum Mot(M1…M4)
OnOff / **Richtung** : enum Dir (On, Off / Left, Right)
Power / **Speed** : int 0 - 512
iCount : Zählerwert
"Encoder"-Motor : echter Encoder oder Kombination Motor/Impuls"Taster". Bei RobMotor auch noch Zuordnung eines EndTasters bei LinksLauf. Dem Motorausgang sind fest Impuls und Endeingänge zugeordnet (gleiche Nr bei I- und C-Eingang).

SetLamp(LampName, OnOff, Power) : Schalten eines O-Ausganges

SetMotor(MotorName, Richtung, Speed) : Schalten eines M-Ausganges

StartMotor(MotorName, Richtung, Speed, iCount) : Starten eines "Encoder"-Motors für die angegebene Impulszahl, der Motor wird bei Erreichen des Ziels asynchron abgeschaltet.

StartRobMotor(MotorName, Richtung, Speed, iCount) Starten eines "Encoder"-Motors für die angegebene Impulszahl, der Motor wird bei Erreichen des Ziels oder des zugehörenden "End"-Tasters (bei LinksLauf) asynchron abgeschaltet.

Warten auf Digitaleingang

WaitForInput(InputName [,true/false]) : Warten auf Erreichen des vorgegebenen Zustandes

WaitForHigh(InputName) : Warten auf einen false/true-Durchgang am Eingang

WaitForLow(InputName) : Warten auf einen true/false-Durchgang am Eingang

Warten auf Motor

Der oder die Motoren müssen durch StartMotor / StartRobMotor gestartet werden.

WaitForMotor(MotorName) : Warten auf Erreichen des Zieles für den angebenen Motor. Es wird eine evtl. Abweichung vom Ziel zurückgegeben.

WaitForMotors(MotorName…) : Warten auf Erreichen der Ziele aller angegebenen Motoren der Liste.

Angegeben wurde die C# Version, bei Java beginnts natürlich mit kleinen Buchstaben

FishFaceTX.DLL : Device-Klassen

Umfangreiche Erweiterung der Assembly FishFaceTX.DLL um Device-Klassen, die zusätzlich zu ihren Methoden auch eine Reihe von Ereignissen bieten. Details siehe FishFaTXCSdev.PDF.

Device-Klassen

DeviceBase

- AnalogInput
  - A10VInput
    - ColorSensor
  - A5KInput
    - NTC
    - PhotoResistor
    - PotentioMeter
  - DistanceSensor
- BinaryInput
  - D5KInput
    - PhotoTransistor
    - PushButton
    - ReedContact
  - HalfTrack
- CounterInput
- DualOutput
  - Motor
- MonoOutput
  - Buzzer
  - Lamp
  - Magnet
- --- Combined Devices ---
- Drive2NXT (2x EncoderMotor)
- EncoderMotor (Motor, CounterInput)
- LightBarrier (Lamp, PhotoTransitor)
- Lights (nx Lamp)
- LimitedMotor (Motor, 2x BinaryInput)
- RobMotor (Motor, D5KInput, CounterInput)
- RobMotors (nx RobMotor)
- TrailSensor (2x HalfTrack)
- TwinMotors (2x Motor)

Programmbeispiele

ftMscLib.DLL

C# 2005 / 2008

HelloFtMscLib

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Threading;
using TX = ftMscLibHello.ftMscLib;
using cs = System.Console;

namespace ftMscLibHello {
  class Program {
    static uint ftHandle;
    static uint errCode;
    const  uint txOK       = 0x00000000;
    const  int  txWarten   = 44;
    const  int  ctxMain    = 0;
    const  int  Taster     = 2;
    const  int  LampeRot   = 4;
    const  int  LampeGruen = 5;

static void Main(string[] args) {
  short wert = 0;
  bool ok = false;
  cs.WriteLine("HelloftMscLib-Test an COM4 gestartet");

  // --- Herstellen der Verbindung ---
     TX.ftxInitLib();
     ftHandle = TX.ftxOpenComDevice("COM4", 38400, ref errCode);
     if (errCode != txOK) {
       cs.WriteLine("ControllerProblem.Open"); cs.Read();
       return;
     }
     TX.ftxStartTransferArea(ftHandle);
  // --- Konfigurieren der Universal-Eingänge ---
     TX.SetFtUniConfig(ftHandle, ctxMain, Taster, 1, true);
     Thread.Sleep(txWarten);

  cs.WriteLine("--- Zum Blinken : I3-Taster ---");
  do {
    TX.GetInIOValue(ftHandle, ctxMain, Taster, ref wert, ref ok);
    Thread.Sleep(txWarten);
  } while (wert == 0);
```

```
  cs.WriteLine("Bei der Arbeit");
  for (int i = 0; i < 10; i++) {
    TX.SetOutPwmValues(ftHandle, ctxMain, LampeRot, 512);
    TX.SetOutPwmValues(ftHandle, ctxMain, LampeGruen, 0);
    Thread.Sleep(444);
    TX.SetOutPwmValues(ftHandle, ctxMain, LampeRot, 0);
    TX.SetOutPwmValues(ftHandle, ctxMain, LampeGruen, 512);
    Thread.Sleep(333);
  }
  TX.SetOutPwmValues(ftHandle, ctxMain, LampeGruen, 0);
  cs.WriteLine("Das war's");

  // --- Schließen der Verbindung ---
     TX.ftxStopTransferArea(ftHandle);
     TX.ftxCloseDevice(ftHandle);
     TX.ftxCloseLib();
     cs.WriteLine("--- FINIS : Enter-Taste ---");
     cs.Read();
}}}
```

umFish50.DLL

VC++ 6.0

HelloFish

```
#include <Windows.h>
#include < iostream.h>
#include "umFish50VC.h"

void main() {
   char Ende;
   int res, i;

   cout << "--- HelloFish50-Test an COM4 gestartet ---" << endl;
   res = txOpenController(4);
  if (res == ctxError) {
    cout << "OpenError" << endl, cin.get(Ende);
    return;
  }
   cout << "--- Zum Blinken : I3 Taster ---" << endl;
   while (txGetInput(ftxMain, ftxI3) == 0) txSleep(12);
  cout << "--- Bei der Arbeit ---" << endl;
   for (i=0; i<10; i++) {
     txSetLamp(ftxMain, ftxO5, ftxFull);
     txSetLamp(ftxMain, ftxO6, ftxOff);
     Sleep(444);
     txSetLamp(ftxMain, ftxO5, ftxOff);
     txSetLamp(ftxMain, ftxO6, ftxFull);
     Sleep(333);
   }
   txSetLamp(ctxMain, ctxO6, ctxOff);
   cout << "Das war's" << endl;
   txCloseController();
   cout << "--- FINIS : Enter-Taste ---" << endl;
   cin.get(Ende);
}
```

Herstellen der Verbindung zum TX Controller an Anschluß COM4 (ggf. Ändern) durch

```
   res = txOpenController(4);
  if (res == ctxError) {
    cout << "OpenError" << endl, cin.get(Ende);
    return;
  }
```

Hier wird der Rückgabeparameter auf den glücklichen Verlauf der Operation abgefragt. Später wird darauf verzichtet.

Java (über javaFish50.DLL)

HelloFish

```
import ftcomputing.roboTX.*;
public class MainProbe {
  public static void main(String[] args) {
    MainProbe mp = new MainProbe();
    System.out.println("Hallo Meister");
    mp.Action();
  }

  private void Action(){
     JavaFish ft = new JavaFish();
     ft.jxOpenController(4);
     ft.jxSetMotor(0, 0, 1, 444);
     JavaFish.sleep(1234);
     ft.jxSetMotor(0, 0, 0, 0);
        do{
       ft.jxSetLamp(0, 4, 512);
       ft.jxSetLamp(0, 5, 0);
       JavaFish.sleep(444);
       ft.jxSetLamp(0, 4, 0);
       ft.jxSetLamp(0, 5, 512);
       JavaFish.sleep(333);
     } while((ft.jxGetInput(0,2) == 0) && (JavaFish.escape() == 0));
     ft.jxSetLamp(0, 4, 0);
     ft.jxSetLamp(0, 5, 0);
     ft.jxCloseController();
     System.out.println("Ja, Meister, woll!");
  }
}
```

Hier wird die Klasse MainProbe erst instanziert. Die eigentliche Funktion läuft dann im Unterprogramm Action ab. Hier wurde außerdem symbolische Konstanten verzichtet (Lampe an O5 hat die Ausgangsnummer 4). OpenController greift auf den TX Controller über COM4 zu (ggf. anpassen)

C# 2005 / 2008

HelloFish50

```
using System;
using System.Collections.Generic;
using System.Text;
using um = umFish50.umFish50;

namespace HelloFish50CS {
  class Program {
    static void Main(string[] args) {
      int res = 0;
      Console.WriteLine("HelloFish50-Test an COM4 gestartet");
      res = um.txOpenController(4);
      if ((uint)res == um.ctxError) {
        Console.WriteLine("OpenError");
        Console.Read();
        return;
      }
      Console.WriteLine("--- Zum Blinken : I3-Taster");
      while (um.txGetInput(um.ctxMain, um.ctxI3) == 0)
        um.txSleep(44);
      Console.WriteLine("Bei der Arbeit");
      for (int i = 0; i < 10; i++) {
        um.txSetLamp(um.ctxMain, um.ctxO5, um.ctxFull);
        um.txSetLamp(um.ctxMain, um.ctxO6, um.ctxOff);
        um.txSleep(444);
        um.txSetLamp(um.ctxMain, um.ctxO5, um.ctxOff);
        um.txSetLamp(um.ctxMain, um.ctxO6, um.ctxFull);
        um.txSleep(333);
      }
      um.txSetLamp(um.ctxMain, um.ctxO6, um.ctxOff);
      Console.WriteLine("Das war's");
      um.txCloseController();
      Console.WriteLine("--- FINIS : Enter-Taste ---");
      Console.Read();
    }
  }
}
```

Console Projekt. Die Elemente der Klasse umFish50 sind static, deshalb keine Instantiierung sondern Qualifizierung durch Namensraum und Klassennamen abgekürzt durch

```
using um = umFish50.umFish50;
```

Herstellen der Verbindung zum TX Controller an Anschluß COM4 (ggf. Ändern) durch

```
      res = um.txOpenController(4);
      if ((uint)res == um.ctxError) {
        Console.WriteLine("OpenError");
        Console.Read();
        return;
      }
```

Hier wird der Rückgabeparameter auf den glücklichen Verlauf der Operation abgefragt. Später wird darauf verzichtet.

VB2005 / 2008

HelloFish

```
Imports cs = System.Console
Imports um = HelloFish50VB.umFish50

Module HelloFish50

  Sub Main()
    Dim res As Int32 = 0
    cs.WriteLine("HelloFish50-Test an COM4 gestartet")
    res = um.txOpenController(4)
    If res = um.ctxError Then
      cs.WriteLine("OpenError")
      cs.Read()
      Return
    End If
    cs.WriteLine("--- Zum Blinken : I3-Taster")
    Do
      um.txSleep(44)
    Loop While um.txGetInput(um.ctxMain, um.ctxI3) = 0
    cs.WriteLine("Bei der Arbeit")
    For i As Integer = 1 To 10
      cs.WriteLine("Runde : " & i)
      um.txSetLamp(um.ctxMain, um.ctxO5, um.ctxFull)
      um.txSetLamp(um.ctxMain, um.ctxO6, um.ctxOff)
      um.txSleep(444)
      um.txSetLamp(um.ctxMain, um.ctxO5, um.ctxOff)
      um.txSetLamp(um.ctxMain, um.ctxO6, um.ctxFull)
      um.txSleep(333)
    Next
    um.txSetLamp(um.ctxMain, um.ctxO6, um.ctxOff)
    cs.WriteLine("Das war's")
    um.txCloseController()
    cs.WriteLine("--- FINIS : Enter-Taste ---")
    cs.Read()
  End Sub

End Module
```

Console Projekt. Die Elemente der Klasse umFish50 sind static, deshalb keine Instantiierung sondern Qualifizierung durch Namensraum und Klassennamen abgekürzt durch

```
Imports um = HelloFish50VB.umFish50
```

Herstellen der Verbindung zum TX Controller an Anschluß COM4 (ggf. Ändern) durch

```
    res = um.txOpenController(4)
    If res = um.ctxError Then
      cs.WriteLine("OpenError")
      cs.Read()
      Return
    End If
```

Hier wird der Rückgabeparameter auf den glücklichen Verlauf der Operation abgefragt. Später wird darauf verzichtet.

Delphi4

```
program HalloDelphi50;

uses
  Windows,
  SysUtils,
  umFish50 in 'umFish50.PAS';
var
  tx, i: LongInt;
begin
  tx := txOpenController(4);
  if tx = ftiFehler then begin
    WriteLn('Hier stimmt etwas nicht : ENDE (Enter-Taste)');
    ReadLn;
    exit;
  end
  else WriteLn('HalloDelphi in Action');
  WriteLn('--- Zum Blinken : I3-Taster');
  while txGetInput(ftiMain, ftiI3) = 0 do Sleep(123);
  WriteLn('Bei der Arbeit');
  for i := 1 to 10 do begin
    WriteLn('Runde : ' + IntToStr(i));
    txSetLamp(ftiMain, ftiO5, ftiFull);
    txSetLamp(ftiMain, ftiO6, ftiOff);
    Sleep(444);
    txSetLamp(ftiMain, ftiO5, ftiOff);
    txSetLamp(ftiMain, ftiO6, ftiFull);
    Sleep(333);
 end;
  txSetLamp(ftiMain, ftiO6, ftiOff);
  txCloseController();
  WriteLn('HalloDelphi beendet'); ReadLn;
end.
```

Console Projekt. umFish50 in 'umFish50.PAS'; Einbinden der Deklarationen.

Herstellen der Verbindung zum TX Controller an Anschluß COM4 (ggf. Ändern) durch

```
  tx := txOpenController(4);
  if tx = ftiFehler then begin
    WriteLn('Hier stimmt etwas nicht : ENDE (Enter-Taste)');
    ReadLn;
    exit;
  end
```

Hier wird der Rückgabeparameter auf den glücklichen Verlauf der Operation abgefragt. Später wird darauf verzichtet.

FishFaceTX.DLL

C# 2005/2008

HelloFish Console

```
using FishFaceTX;

namespace HelloFishTXConsoleCS {
  class Program {
    static FishFace tx = new FishFace();

    static void Main(string[] args) {
      try {
        Console.WriteLine("--- HelloFish gestartet     ---");
        tx.OpenController("COM4");
        Console.WriteLine("--- Beenden : Escape-Taste  ---");
        Console.WriteLine("--- Zum Blinken : I3-Taster ---");
        tx.WaitForInput(Unv.I3);
        Console.WriteLine("--- Bei der Arbeit          ---");
        do {
          tx.SetLamp(Out.O5, Dir.On);
          tx.SetLamp(Out.O6, Dir.Off);
          tx.Pause(444);
          tx.SetLamp(Out.O5, Dir.Off);
          tx.SetLamp(Out.O6, Dir.On);
          tx.Pause(444);
        } while (!tx.Finish());
      }
      catch (FishFaceException txe) {
        Console.WriteLine(txe.Message);
      }
      finally {
        tx.CloseController();
        Console.WriteLine("--- RETURN Taste drücken    ---");
        Console.ReadLine();
      }
    }
  }
}
```

HelloFish Windows - Nutzung Template FishTXWindows

```
using FishFaceTX;

namespace HelloFishTXCS {
  public partial class frmMain : Form {
    FishFace tx    = new FishFace();
    Unv Starter    = Unv.I3;
    Out LampeRot   = Out.O5;
    Out LampeGruen = Out.O6;

    private void Action() {
      lblStatus.Text = "Zum Blinken : I3-Taster";
      tx.WaitForInput(Starter);
      lblStatus.Text = "Bei der Arbeit";
      do {
        tx.SetLamp(LampeRot,   Dir.On);
        tx.SetLamp(LampeGruen, Dir.Off);
        tx.Pause(444);
        tx.SetLamp(LampeRot,   Dir.Off);
        tx.SetLamp(LampeGruen, Dir.On);
        tx.Pause(444);
      } while (!tx.Finish());
    }

    #region --- Programm-Kontrolle ---
  }
}
```

Download siehe www.ftcomputing.de/fishdllstx.htm, Abschnitt C#2005/2008

VB2005/2008

HelloFish Console

```
Imports FishFaceTX
Module HelloFishTXConsoleVB
   Dim tx As New FishFace()
   Sub Action()
      Do
         tx.SetLamp(Out.O5, Dir.On)
         tx.SetLamp(Out.O6, Dir.Off)
         tx.Pause(444)
         tx.SetLamp(Out.O5, Dir.Off)
         tx.SetLamp(Out.O6, Dir.On)
         tx.Pause(333)
      Loop Until tx.Finish()
   End Sub
```

```
#Region "--- ProgramControl ---"
  Sub Main()
    Try
       tx.OpenController("COM4") ' Den COM-Namen anpassen
      Console.WriteLine("Bei der Arbeit, Abbruch ESC-Taste")
            Action()
   Catch txe As FishFaceException
     Console.WriteLine(txe.Message)
   Finally
     tx.CloseController()
    Console.WriteLine("--- FINITO : RETURN drücken ---")
    Console.Read()
   End Try
  End Sub
#End Region
End Module
```

HelloFish Windows

```
Imports FishFaceTX

Public Class HelloFishTXWinVB
  Dim tx As New FishFace()
  Const Starter As Unv = Unv.I3
  Const LampeRot As Out = Out.O5
  Const LampeGruen As Out = Out.O6

    Private Sub Action()
     lblStatus.Text = "Zum Blinken : I3-Taster"
     tx.WaitForInput(Starter)
     lblStatus.Text = "Bei der Arbeit"
     Do
       tx.SetLamp(LampeRot, Dir.On)
       tx.SetLamp(LampeGruen, Dir.Off)
       tx.Pause(444)
       tx.SetLamp(LampeRot, Dir.Off)
       tx.SetLamp(LampeGruen, Dir.On)
       tx.Pause(333)
     Loop Until tx.Finish()
    End Sub

#Region "--- ProgramControl ---"
End Class
```

Download siehe www.ftcomputing.de/fishdllstx.htm, Abschnitt VB2005/2008

Java 6 und Eclipse 3.4 mit der Java-Klasse FishFaceTX

HelloFish Console

```
import ftcomputing.roboTX.*;
public class TXmain {
   FishFaceTX tx = new FishFaceTX();

   public static void main(String[] args) {
      TXmain mf = new TXmain();
      System.out.println("TXmain gestartet");
      System.out.println("FishFace-Version : " +
                                          FishFaceTX.Version());
      try { mf.Action(333); }
      catch(FishFaceException eft) { System.out.println(eft);}
      finally { System.out.println("Finito");}

   }
   private void Action(int Dauer) {
       tx.openController("COM4");
       int Runde = 0;
       tx.startMotor(Mot.M1, Dir.Left, 444, 12);
       tx.waitForMotor(Mot.M1);
       do {
           System.out.println("Runde    : " + ++Runde);
           tx.setLamp(Out.O5, 512);
           tx.pause(Dauer);
           tx.setLamp(Out.O6, 512);
           tx.pause(Dauer);
           tx.setLamp(Out.O7, 512);
           tx.pause(Dauer);
           tx.setLamp(Out.O5, 0);
           tx.setLamp(Out.O6, 0);
           tx.setLamp(Out.O7, 0);
           tx.pause(Dauer * 2);
       } while(!tx.finish());
       tx.closeController();
       System.out.println("--- FINITO ---");
   }
}
```

Download siehe www.ftcomputing.de/fishdllstx.htm, Abschnitt Java

Python 3.1.1 und FishFaTX.py

Console Anwendung Blinker

```
# ---------- BlinkerTest50c.py ----------------------------------


from FishFaTX import *

print ('------ Blinker wird gestartet ------')
ft = FishFace()
try:
  ft.OpenController(4)
  print ('Zum Starten des Blinkers I3 Druecken')

  ft.WaitForInput(0, 3)
  Runde = 1

  while not ft.Finish(0, 1):
    print ('Runde : ', Runde)
    ft.SetLamp(0, 5, 512)
    ft.Pause(555)
    ft.SetLamp(0, 5, 0)
    ft.Pause(333)
    Runde += 1
except FishFaceException:
  print ('FishFace Fehler')
print ('------- Blinker wird beendet -------')
ft.CloseController()
```

Download siehe www.ftcomputing.de/fishdllstx.htm Abschnitt Python