



TR Mark III SDK

Quick Start Guide

Raytech GmbH
Oberebenestrasse 11
CH 5620 Bremgarten
Switzerland

Version 1.01/ZF

Document Review			
Version	Amendment	By	Date
1.00	Initial	Z. Filipovic	17.06.2011
1.01	Code examples changed to match .NET project update.	Z. Filipovic	02.09.2011



Contents

1 About this Guide.....	3
2 What is TR Mark III SDK.....	3
3 Prerequisites.....	3
4 Installing TR Mark III native USB Driver.....	3
4.1 Windows XP.....	3
4.2 Windows Vista, 7, 2003 and 2008.....	3
5 Getting started.....	5
5.1 Creating project.....	5
5.2 Namespaces.....	6
5.3 TR Mark III Driver instance.....	6
5.4 Connect and Disconnect.....	7
5.5 Get Connected USB Devices.....	7
5.6 SendAndReceive method.....	7
5.7 Send Method.....	8
5.8 Start and Stop measuring.....	8
5.9 Retrieving Measuring results, asynchronous events.....	8
5.10 Receive all Data, asynchronous events.....	8
6 Appendix.....	9



1 About this Guide

This quick reference guide provides a short introduction to TR Mark III SDK library. This manual clarifies the various possible applications of the TR Mark III SDK package, and provides a systematic approach for TR Mark III customization projects.

2 What is TR Mark III SDK

TR Mark III SDK is device management software. It is essentially a software layer (or driver) that resides between the Operating System (OS) IO system, custom Windows Application and TR Mark III device. TR Mark III SDK with native Raytech USB Driver for Windows provides the OS with full device functionality, appearing to OS as Raytech USB device. TR Mark III SDK can also be used without native Raytech USB drivers to control the device over standard RS 232 Serial port.

3 Prerequisites

- TR Mark III minimum acceptable firmware version 3.0086
- PC (Windows XP SP 3, Vista, 7, 2003, 2008 operating systems)
- .NET compatible development kit (e.g. Visual Studio 2010, SharpDevelop 4.x)
- .NET 4.0
- TR Mark III SDK
- In order to use USB functionality, native TR Mark III USB Driver must be installed on the system.

4 Installing TR Mark III native USB Driver

USB Driver installation is not necessary in order to use RS 232 serial port for communication with device.

4.1 Windows XP

Connect the TR Mark III with your PC using USB cable. In the Driver Installation Dialog choose a folder containing *TR-Mark_III.INF* file and click "Next" Button. Ignore unsigned driver installation warning.

4.2 Windows Vista, 7, 2003 and 2008

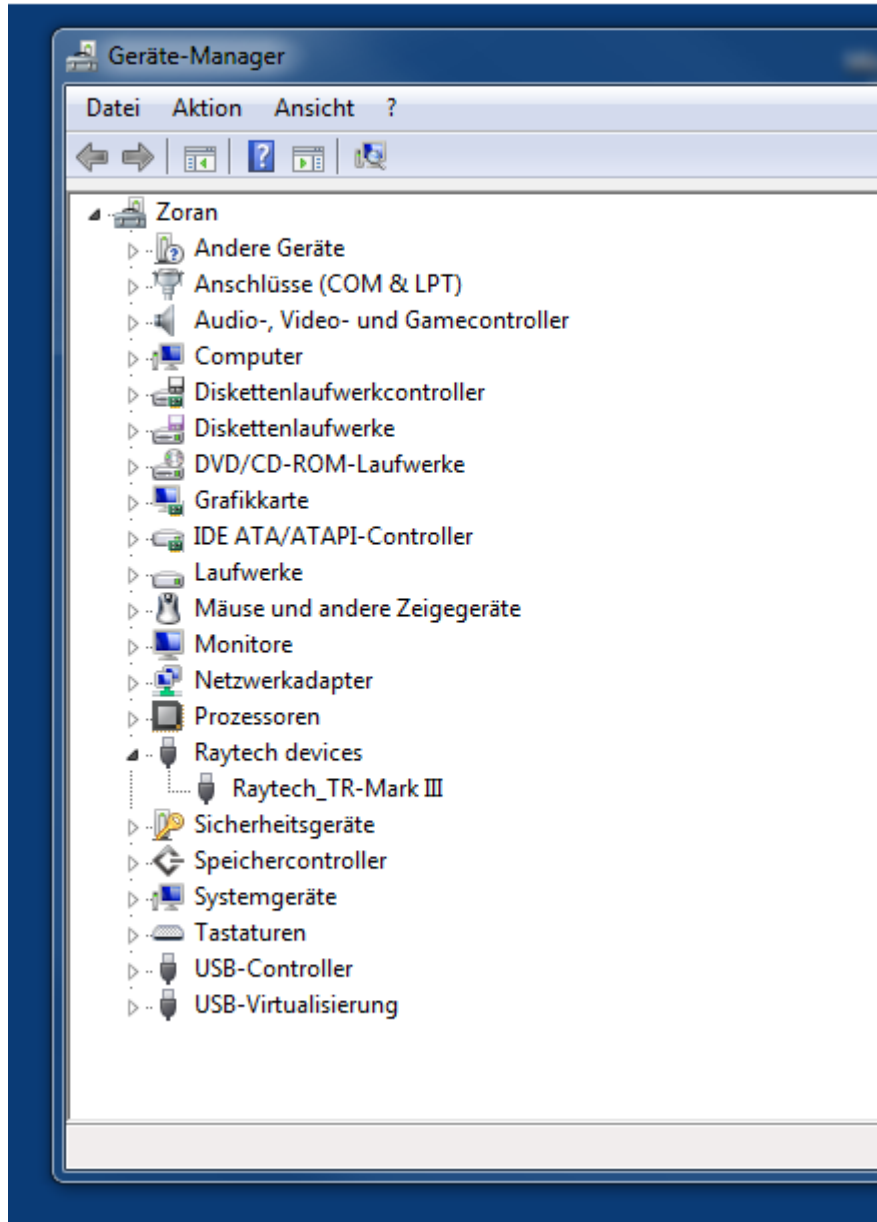
Open the folder containing TR Mark III USB drivers. Double click on:

- *install_x86.exe*, automatic driver installer for 32 bit OS
- *install_x64.exe*, automatic driver installer for 64 bit OS
- *install_ia64.exe*, automatic driver installer for Itanium based systems

Ignore unsigned driver installation warning in order to complete installation.



Windows Device Manager after successfully driver installation. Notice new device-group entry “Raytech devices” with “Raytech_TR-Mark III” as child.



NOTE

⇒ In order to use native Raytech USB drivers, TR Mark III firmware must be reconfigured. This will be achieved by typing service code 2001 into the device setup.



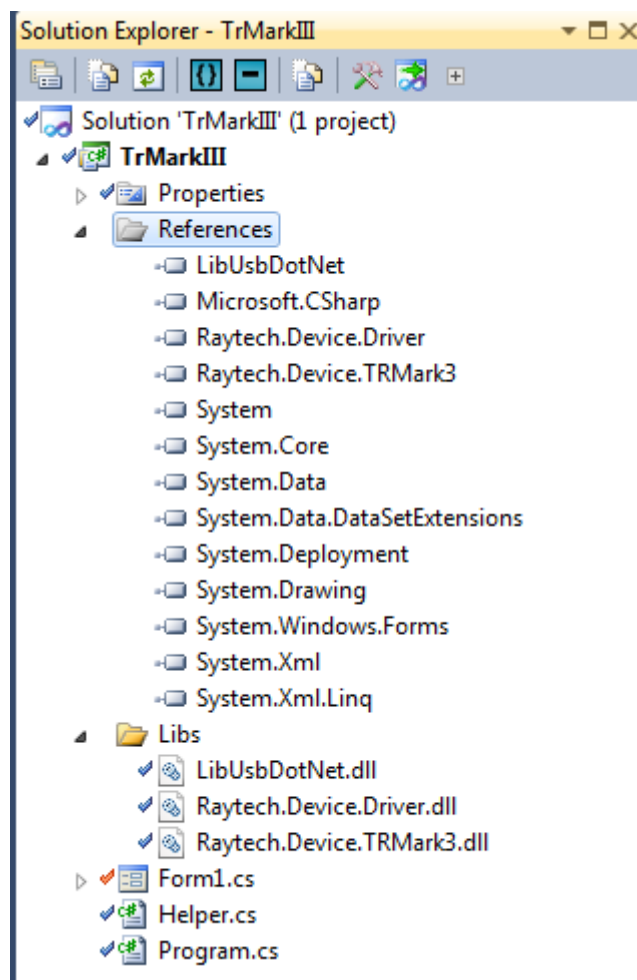
5 Getting started

5.1 Creating project

First of all, create new project in integrated development environment of your choice and add references to the following libraries:

- **LibUsbDotNet.dll** (.NET C# USB wrapper library for WinUSB, LibUsb-Win32 and libusb-1.0)
- **Raytech.Device.Driver.dll** (Base library for TR Mark III and other Raytech device software)
- **Raytech.Device.TRMark3.dll** (Software library for interaction with TR Mark III)

It is recommended to copy the library files in some folder of the project.





5.2 Namespaces

In order to use full potential of the SDK, object from following namespaces can be used:

```
using Raytech.Device.TRMark3;
using Raytech.Device.Driver.Common;
using Raytech.Device.Driver.USB;
using Raytech.Device.Driver.Helpers;
using Raytech.Device.Driver.InformationHandling;
```

5.3 TR Mark III Driver instance

There are several ways to use instance driver in your application.

1. Direct(simple)

```
TrMk3Device driver = new TrMk3Device();
```

2. Using IDriverBase interface (recommended):

```
IDriverBase driver = new TrMk3Device();
```

3. As module (plugin) in MEF (Managed Extensibility Framework) based application (advanced):

```
[ImportMany(typeof(IDriver))]
public List<Lazy<IDriver, IDriverMetadata>> DeviceApplications {
    get; private set;
}

private void LoadDeviceControlApplications() {
    try {
        DirectoryCatalog directoryCatalog = new DirectoryCatalog(
            Environment.CurrentDirectory + @"\Devices");

        this.compositionContainer = new CompositionContainer(directoryCatalog);

        this.compositionContainer.ComposeParts(this);
    } catch (Exception ex) {
        Messenger.Default.Send(new ErrorDialogMessage(ex.Message, null));
    }
}

public IDriverBase GetDriver() {
    return DeviceApplications.Value.DeviceControlDriver;
}
```



5.4 Connect and Disconnect

Serial connection mode:

```
this.driver.SerialPortName = (sender as ComboBox).Text;
```

```
driver.SetSerialCommunicationMode();  
driver.Connect();
```

USB connection mode.

```
//USB device selection change event  
this.cmbUsbDevice.SelectedIndexChanged += (sender, e) => {  
  
    if (!string.IsNullOrEmpty(this.cmbUsbDevice.Text)) {  
        //Find selected USB device  
        var query = from d in this.usbDeviceHandler.GetConnectedDevices()  
                    where this.cmbUsbDevice.Text == d.GetProductString()  
                    select d;  
  
        //Expose USB device to the driver  
        if (query.Count() > 0)  
            this.driver.UsbDevice = query.First();  
    }  
};
```

```
driver.SetUsbCommunicationMode();  
driver.Connect();
```

Disconnect and reset:

```
driver.Disconnect();
```

5.5 Get Connected USB Devices

Class UsbDeviceHandler is USB information source:

```
UsbDeviceHandler usbDeviceHandler = new UsbDeviceHandler();  
  
var devices = this.usbDeviceHandler.GetConnectedDevices();
```

React on device plugged in and out:

```
this.usbDeviceHandler.OnUsbNotify += delegate {  
    //Do something  
};
```

5.6 SendAndReceive method

The *SendCommand* method sends a *command* to the *device* and retrieves the results synchronously.

```
int timeout = 1000;  
ExecutionData answer = driver.SendAndReceive("GV", timeout, ReceiveMode.Once);
```

If more than one answer is expected, *ReceiveMode* must be set to *Multiple*.



5.7 Send Method

The *Send* method sends a *command* to the *device* and retrieves no results.

```
driver.Send("GV");
```

5.8 Start and Stop measuring

```
driver.StartMeasure<TrMk3MeasurementMode>(
    TrMk3MeasurementMode.SinglePhaseA, IntermediateResultDisplay.Show);
```

```
driver.Send("STOP");
```

5.9 [Retrieving Measuring results](#), asynchronous events

```
this.driver.OnSendResult += (sender, e) => {
    this.DisplayResults(e.Result as Measurement);
};
```

Measurement is specific TR Mark III object.

5.10 Receive all Data, asynchronous events

Organized as ExecutionData...

```
this.driver.OnSendExecutionData += (sender, e) => {
    // implementation
};
```

Or row data as strings...

```
this.driver.OnSendDataRow += (sender, e) => {
    //implementation
};
```




6 Appendix

For the full reference list of the TR Mark III commands, please refer to the “*Command Set TRMarkIII.pdf*” delivered with this package, or call the Raytech GmbH technical support.