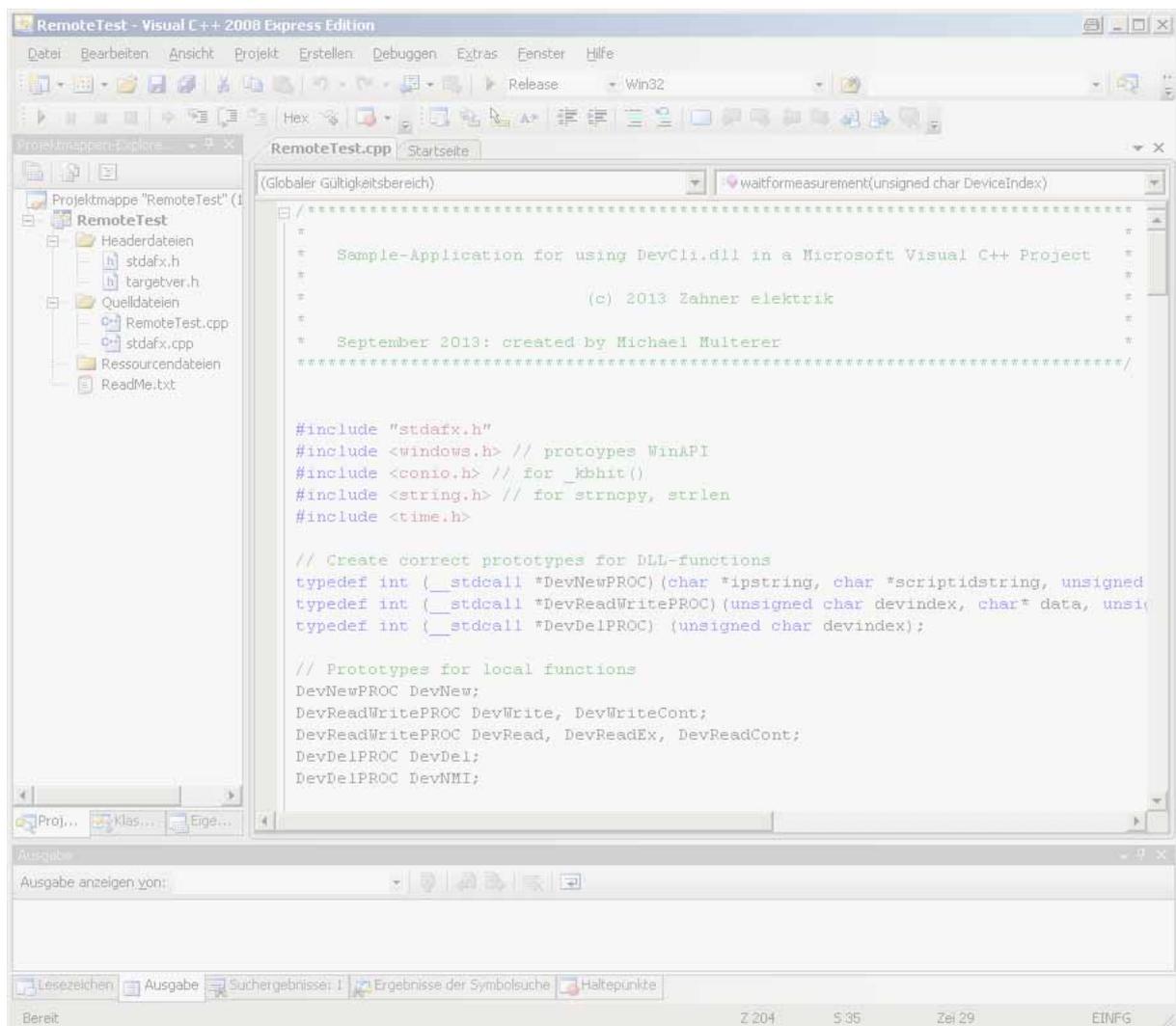


# DevCli.dll

## Programmer's Reference



```
RemoteTest - Visual C++ 2008 Express Edition
Datei Bearbeiten Ansicht Projekt Erstellen Debuggen Extras Fenster Hilfe
Release Win32
Projektmappe "RemoteTest" (1)
  RemoteTest
    Headerdateien
      stdafx.h
      targetver.h
    Quelldateien
      RemoteTest.cpp
      stdafx.cpp
    Ressourcendateien
      ReadMe.txt
RemoteTest.cpp
(Globaler Gültigkeitsbereich)
waitformeasurement(unsigned char DeviceIndex)
Sample-Application for using DevCli.dll in a Microsoft Visual C++ Project
(c) 2013 Zahner elektrik
September 2013: created by Michael Multerer
#include "stdafx.h"
#include <windows.h> // prototypes WinAPI
#include <conio.h> // for _kbhit()
#include <string.h> // for strlen, strcpy, strncpy, strlen
#include <time.h>
// Create correct prototypes for DLL-functions
typedef int (__stdcall *DevNewPROC)(char *ipstring, char *scriptidstring, unsigned
typedef int (__stdcall *DevReadWritePROC)(unsigned char devindex, char* data, unsi
typedef int (__stdcall *DevDelPROC)(unsigned char devindex);
// Prototypes for local functions
DevNewPROC DevNew;
DevReadWritePROC DevWrite, DevWriteCont;
DevReadWritePROC DevRead, DevReadEX, DevReadCont;
DevDelPROC DevDel;
DevDelPROC DevNMI;
```



<b>1. Introduction</b>	<b>4</b>
<b>2. Function Overview</b>	<b>4</b>
<b>3. Administrative Functions</b>	<b>5</b>
<b>4. Read/Write Functions</b>	<b>6</b>

---

# 1. Introduction

---

Thales offers a software interface to extend its connectivity towards third party software. As this interface is realized by TCP/IP socket communication the third party software can even run on a separate computer and communicate with Thales over network. The Windows<sup>®</sup> dynamic link library DevCli.dll provides all functions necessary for communication with the Thales window over TCP/IP. It is used for several tasks like [Signal Acquisition with Net VIs](#) and [Remote Control](#) of the IM6/Zennium. This manual describes the common low level communication functions used whereas details of the applications are given in the manuals of [Signal Acquisition](#) and [Remote Control](#). The LabVIEW<sup>™</sup> sample applications of Net VI and Remote Control also use the functions of DevCli.dll encapsulated in a LabVIEW<sup>™</sup> library.

---

# 2. Function Overview

---

The complete communication between third party software and Thales is realized by the functions exported by c:\flink\DevCli.dll. Although the functions are presented in Delphi syntax, their use is not limited to a dedicated development platform. They can be used in every software which is able to call functions of a Windows<sup>®</sup> Dynamic Link Library (DLL).

The following functions are available:

```
function DevNew(IPAddr,DevName:PChar;Size:SmallInt):Integer;stdcall;  
function DevNewEx(IPAddr,DevName:PChar;Size:SmallInt):Integer;stdcall;  
function DevDel(DevIdx:Byte):Integer;stdcall;  
function DevNMI(DevIdx:Byte):Integer;stdcall;
```

```
function DevWrite(DevIdx:Byte;var Data;N:Integer):Integer;stdcall;  
function DevRead(DevIdx:Byte;var Data;N:Integer):Integer;stdcall;  
function DevWriteEx(DevIdx:Byte;var Data;N:Integer):Integer;stdcall;  
function DevReadEx(DevIdx:Byte;var Data;N:Integer):Integer;stdcall;  
function DevWriteCont(DevIdx:Byte;var Data;N:Integer):Integer;stdcall;  
function DevReadCont(DevIdx:Byte;var Data;N:Integer):Integer;stdcall;
```

## 3. Administrative Functions

### **DevNew(IPAddr,DevName:PChar;Size:SmallInt):Integer;stdcall;**

This function initialises the TCP/IP communication with the Thales window and has to be called before using any other function.

#### **Parameters**

IPAddr pointer to string containing IP-address or computer name, e.g. "127.0.0.1"  
 DevName pointer to string containing name of device, choose a descriptive name for the Net VI, use "ScriptRemote" for remote control  
 Size integer (16 bit) setting the buffer size used by Thales for communication in bytes, e.g. 256

#### **Return Value**

-1, 0 setup of communication to device failed  
 > 0 device index (DevIdx) for further communication

If communication with the Thales window fails, the following message box is displayed. In case the message box is unwanted for a silent retry use DevNewEx.



### **DevNewEx(IPAddr,DevName:PChar;Size:SmallInt):Integer;stdcall;**

Same as DevNew, but no message box is displayed if communication with the Thales window fails.

### **DevDel(DevIdx:Byte):Integer;stdcall;**

This function should be called to terminate the TCP/IP communication properly when the Client is shutting down.

#### **Parameters**

DevIdx device index obtained from DevNew

#### **Return Value**

-1 error, e.g. invalid device index  
 0 success

### **DevNMI(DevIdx:Byte):Integer;stdcall;**

Initiate an NMI (non maskable interrupt) on the IM6/Zennium, can be used to boot an IM6/Zennium remote.

#### **Parameters**

DevIdx device index obtained from DevNew

#### **Return value**

-1 error, e.g. invalid device index  
 0 success

## 4. Read/Write Functions

---

Remark on read/write functions:

There are three pairs of read/write functions called DevRead/DevWrite, DevReadEx/DevWriteEx and DevReadCont/DevWriteCont which use different data buffers and are therefore not exchangeable. Watch out which function is used by Thales in each special case and use the appropriate counterpart.

### **DevWrite(DevIdx:Byte;var Data;N:Integer):Integer;stdcall;**

Write to Thales, overwrite last data in buffer.

#### **Parameters**

DevIdx     device index obtained from DevNew  
Data        pointer to write data buffer  
N            number of bytes to write, 32 bit integer

#### **Return Value**

-1            error, e.g. invalid device index  
> -1         number of bytes written

### **DevRead(DevIdx:Byte;var Data;N:Integer):Integer;stdcall;**

Read data from Thales, clear source buffer.

#### **Parameters**

DevIdx     device index obtained from DevNew  
Data        pointer to buffer for received data  
N            not used

#### **Return value**

-1            error, e.g. invalid device index  
> -1         number of read characters (including 0)

### **DevWriteEx(DevIdx:Byte;var Data;N:Integer):Integer;stdcall;**

Write data to Thales, overwrite last data in buffer.

#### **Parameters**

DevIdx     device index obtained from DevNew  
Data        pointer to write data  
N            number of bytes to write, 32 bit integer

#### **Return Value**

-1            error, e.g. invalid device index  
> -1         number of bytes written

### **DevReadEx(DevIdx:Byte;var Data;N:Integer):Integer;stdcall;**

Read data from Thales, keep data in buffer. Subsequent calls of DevReadEx return the same result until it is overwritten by Thales.

#### **Parameters**

DevIdx     device index obtained from DevNew  
Data        pointer to buffer for received data  
N            not used

#### **Return value**

-1            error, e.g. invalid device index  
> -1         number of read characters (including 0)

## **DevWriteCont(DevIdx:Byte;var Data;N:Integer):Integer;stdcall;**

Write to Thales using a FIFO buffer. This function is used by Remote Control of Thales Z2.21 and later.

### **Parameters**

DevIdx     device index obtained from DevNew  
Data       pointer to write data buffer  
N           number of bytes to write, 32 bit integer

### **Return Value**

-1           error, e.g. invalid device index  
> -1         number of bytes written

## **DevReadCont(DevIdx:Byte;var Data;N:Integer):Integer;stdcall;**

Read data from Thales using a FIFO buffer. This function is used by Remote Control of Thales Z2.21 and later.

### **Parameters**

DevIdx     device index obtained from DevNew  
Data       pointer to buffer for received data  
N           not used

### **Return value**

-1           error, e.g. invalid device index  
> -1         number of read characters (including 0)