Documentation coscom.dll Version 1.2.9

Released 24.01.2006

coscom.dll build 1.2.9 / 24.01.2006 - 9:48 h

Changes made

The function „coscomgetStatus" returns now the coscom function S00 value. In version 1.0.0 the coscomgetStatus is different from the original coscom-protocol.

There is no limit for the port number („coscomopenPort"). In version 1.0.0 the coscomgetStatus is different from the original coscom-protocol.

The dll can be more simply integrated into programs (refer to chapter 0).

The function coscomgetDuration returns the actual time of the coscom function T00 in milliseconds. In version 1.0.0. coscomgetDuration did not work.

For the MFC compatibility new include files for explicit binding was added. See chapter E.

The function "coscomclosePort" checks the port number correct.

Integrating coscom.dll into the program

A General rules for function calls

All functions except „coscomopenPort“, „coscomclosePort“, „isvalidCoscomDevice“ call inside the function „isvalidCoscomDevice" so it isn't necessary to call this function extra.

B Return values of the coscom.dll functions

B1 Delphi

```
FLastError := coscomget...(FPortNumber,parameter ...);
     case FLastError of
     0: FErrorText := 'connection is off';
     1: do somthing
     1: FErrorText := 'function is unsupported';
     2: FErrorText := 'connection is interrupted';
     3: FErrorText := 'an exception or an other error occured'
     4: FErrorText := 'wrong portnumber'
     end;
```

B2 C / C++

```
int FLastError = coscomget...(FPortNumber,parameter ...);
     switch( FLastError of){
     case 0: FErrorText = "connection is off"; break;
     case 1: do something break;
     case -1: FErrorText = ""function is unsupported""; break;
     case -2: FErrorText = "connection is interrupted""; break;
     case -3: FErrorText = "an exception or an other error occured"; break;
     case -4: FErrorText = "wrong portnumber"; break;
     }
```

C Delphi:

- Copy "coscom.dll" in the program directory.
- Add the file " coscom1_2.pas to the project.
- Add suitable unit the uses entry coscom1_2.
- Add to the functions in the program the desired function calls.

Example:

```
uses
      coscom1_2
;

procedure TForm.ActionOpenDeviceExecute(Sender: TObject);
var
   aportnumber : Integer;
```

```
begin
    closeDevice;
    try
       aportnumber := StrToInt(Edit1.Text);
    except
          exit;
    end;
    if coscomopenPort(aportnumber) > 0 then        FPortNumber := aportnumber;
end;

procedure TForm.closeDevice;
begin
    if (FPortNumber > 0) and (coscomclosePort(FPortNumber) > 0) then
      FPortNumber := 0;
end;

procedure TForm.Button5MouseDown(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
    coscomKeyPressStart(FPortNumber,1);
end;

procedure TForm.Button5MouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
    coscomKeyPressStart(FPortNumber,0);
end;

procedure TForm.ActionSetSpeedExecute(Sender: TObject);
var
   myValue : Double;
begin
      try
         myValue := StrToFloat('20');
         if coscomsetSpeed(FPortNumber,myValue) > 0 then
            show('Speed Ok');
         else
             show('Speed failed');
         except
              show('Speed failed');
         end;

end;
```

D C / C ++

- Copy "coscom.dll" in the program directory.
- Add to the project the file "coscom.lib".
- Add to the source code the include instruction #include "coscom1_2.h".
- Add to the functions in the program the desired function calls.

Example:

```
#include "coscom1_2.h"

void ActionOpenDeviceExecute(int aportnumber)
{
    closeDevice();
    try
    {
      aportnumber := StrToInt(Edit1.Text);
    }
    catch(...)
    {
         return;
    }
    if (coscomopenPort(aportnumber) > 0 )
       FPortNumber = aportnumber;
}

void closeDevice()
{
    if ( (FPortNumber > 0) && (coscomclosePort(FPortNumber) > 0) )
     FPortNumber = 0;
}

void Button5MouseDown()
{
    coscomKeyPressStart(FPortNumber,1);
}

void Button5MouseUp()
{
    coscomKeyPressStart(FPortNumber,0);
}

void ActionSetSpeedExecute()
{
  double myValue;
  try
  {
     //myValue = 20;
    if (coscomsetSpeed(FPortNumber,myValue) > 0 )
          show("Speed Ok");
    else
           show("Speed failed);
  }
  catch(...)
  {
    show("Speed failed);
  }
}

void ActionGetSpeedExecute()
{
  double myvalue;
  if (coscomgetSpeed(FPortNumber,&myvalue) > 0)
  {

    show(FormatFloat("0.0",myvalue));
  }
  else
    show("get speed error");
}
```

E C / C ++ MFC

- In order to use the „coscom.dll“ with the MFC you have to include the dll with the explicit binding mechanism.
- Copy "coscom.dll" in the program directory (debug path etc.).
- Add to the project the files"coscom_vc_1_2.h" and coscom_vc_1_2.cpp.
- Add to the source code the include instruction #include "coscom_vc_1_2.h".
- Add to the overloaded main function CWinApp ::InitInstance() the function vc_loadcoscomDLL() and to the main function CWinApp::ExitInstance() the function vc_unloadcoscomDLL().
- Add to the functions in the program the desired function calls.

Example:

```
#include "stdafx.h"
#include "visualcoscomtest.h"
#include "visualcoscomtestDlg.h"
#include "coscom_vc_1_2.h"

CVCApp theApp;

///////////////////////////////////////////////////////////////////////////
// CVCApp Initialisation

BOOL CVCApp::InitInstance()
{
      if(!vc_loadcoscomDLL())
```

```
    {
        //do something
        return FALSE;
    }

    CVCcoscomdlltestDlg dlg;
    m_pMainWnd = &dlg;
    int nResponse = dlg.DoModal();
    if (nResponse == IDOK)
    {
    }
    else if (nResponse == IDCANCEL)
    {
    }

    return FALSE;  or TRUE if you uses no modal dialog
}

int CVCApp::ExitInstance()
{
    vc_unloadcoscomDLL();
    return CWinApp::ExitInstance();
}


void CVCcoscomdlltestDlg::OnOpenPort()
{
    coscomopenPort(1);
}

void CVCcoscomdlltestDlg::OnClose()
{
    coscomclosePort(1);
}

void CVCcoscomdlltestDlg::OnSetSpeedAccel()
{
    coscomsetAcceleration(1, 4)
    coscomsetSpeed(1, 1.27);
    //or
    //      coscomsetSpeedAccel(1,1.27,4);
}

void CVCcoscomdlltestDlg::OnDevStart()
{
    coscomKeyPressStart(1,1);
    coscomKeyPressStart(1,0);
}
```

```
void CVCcoscomdlltestDlg::OnDevStop()
{
	coscomKeyPressStop(1,1);
	coscomKeyPressStop(1,0);
}
```

Function reference

A Function (coscomopenPort)

opens port with port number 1 .. x
return value -4 means invalid port number
return value 0 means an error occurred
return value 1 means successful call of function

```
extern "C" int __stdcall coscomopenPort(int portnumber);
```

B Function (coscomclosePort)

closes port with port number 1 .. x
return value -4 means invalid port number
return value 0 means an error occurred
return value 1 means successful call of function

```
extern "C" int __stdcall coscomclosePort(int portnumber);
```

C Function (isvalidCoscomDevice)

checks if the device is ready for function call with port number 1 .. x
return value -4 means invalid port number
return value 0 means the port isn't open
return value -2 means the port is opened but the device can't be controlled
return value 1 means the device can controlled by user

```
extern "C" int __stdcall isvalidCoscomDevice(int portnumber);
```

D Function (coscomgetProtocol)

returns the device hardware protocol status with port number 1 .. x
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user
returned value how;
0 == off
1 == on

```
extern "C" int __stdcall coscomgetProtocol(int portnumber, int *how);
```

E Function (coscomsetProtocol)

this function enables/disables the hardware protocol device with port number 1 .. x
parameter how
0 == off
1 == on
remark: if switches the protocol on, the dll automatically creates a directory
under the program directory named "Protocol". in this directory an file is opened
with an name with the following format "Protocolyyyymmdd-n.TXT"
yyyy == year
mm == month
dd == day
n == ticks from start of the windows session
a valid example is "Protocol20051214-3910265.TXT"

```
extern "C" int __stdcall coscomsetProtocol(int portnumber, int how);
```

F Function (coscomgetVersion)

get the device eprom software version with port number 1 .. x
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user
returned parameter values:
versionchars = see coscom.org function V00

```
extern "C" int __stdcall coscomgetVersion(int portnumber, char * versionchars);
```

G Function (coscomgetStatus)

checks the device control status with port number 1 .. x
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user
returned parameter values:
returned value value:
0 == Stop
1 == Run
2 == Pause
3 == Stopping
4 == Detecting
5 == Off

```
extern "C" int __stdcall coscomgetStatus(int portnumber, int *value);
```

H Function (coscomgetDuration)

get the device runtime duration 1 .. x
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user
returned parameter values:
returned value value:
value = duration in ms
- this duration is the time returned by the coscom function T00

```
extern "C" int __stdcall coscomgetDuration(int portnumber, int *value);
```

I Function (coscomgetSpeed)

get the device actual speed with port number 1 .. x
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user
returned parameter values:
returned value value:
value = speed in m/s for treadmills m/min for ladder ergometers
- this speed is the value returned by the coscom function S01

```
extern "C" int __stdcall coscomgetSpeed(int portnumber, double * value);
```

J Function (coscomgetAcceleration)

get the device actual acceleration index with port number 1 .. x
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user
returned parameter values:
returned value value:
value = acceleration index 0..7
- this acceleration index is the value returned by the coscom function A01

```
extern "C" int __stdcall coscomgetAcceleration(int portnumber, int *value);
```

K Function (coscomgetElevation)

get the device actual elevation with port number 1 .. x
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user
returned parameter values:
returned value value:
value = elevation in % - (height / length) * 100%
- this elevation is the value returned by the coscom function E01

```
extern "C" int __stdcall coscomgetElevation(int portnumber, double * value);
```

L Function (coscomgetHeartrate)

get the device actual heart rate with port number 1 .. x
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user
returned parameter values:
returned value value:
value = heart rate in beats per minute
0 == invalid heart rate value
- this elevation is the value returned by the coscom function P01

```
extern "C" int __stdcall coscomgetHeartrate(int portnumber, int *value);
```

M Function (coscomgetDistance)

get the device actual distance with port number 1 .. x
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user
returned parameter values:
returned value value:
value = distance in meters for treadmills / decimetre for ladder ergometer
- this distance is the value returned by the coscom function D00

```
extern "C" int __stdcall coscomgetDistance(int portnumber, int *value);
```

N Function (coscomgetError)

get the device error status with port number 1 .. x
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user
returned parameter values:
returned value value:
value = error numbers
- this error numbers are the values returned by the coscom function Z00

```
extern "C" int __stdcall coscomgetError(int portnumber, int *value);
```

O Function (coscomgetFailSafe)

get the device fail safe time with port number 1 .. x
remark: this value is the last value set by the function coscomsetFailSafe
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user
returned parameter values:
returned value value:
value = 0..25
- this fail safe time is the value set by the coscom function F00

```
extern "C" int __stdcall coscomgetFailSafe(int portnumber, int *value);
```

P Function (coscomgetWeight)

get the device Weight with port number 1 .. x
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user
returned parameter values:
returned value value:
value = 0..
- this weight is the value returned by the coscom function J00

```
extern "C" int __stdcall coscomgetWeight(int portnumber, int *value);
```

Q Function (coscomgetEnergy)

get the device Energy with port number 1 .. x
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user
returned parameter values:
returned value value:
value =
- this energy is the value returned by the coscom function J01

```
extern "C" int __stdcall coscomgetEnergy(int portnumber, double * value);
```

R Function (coscomgetPower)

get the device Power with port number 1 .. x
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user
returned parameter values:
returned value value:
value =
- this power is the value returned by the coscom function J02

```
extern "C" int __stdcall coscomgetPower(int portnumber, double * value);
```

S Function (coscomsetFailSafe)

get the device fail safe time with port number 1 .. x
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user
returned parameter values:
returned value value:
value = 0..25
- this fail safe time is the value set by the coscom function F00

```
extern "C" int __stdcall coscomsetFailSafe(int portnumber, int value);
```

T Function (coscomsetSpeed)

sets the device program speed with port number 1 .. x
parameter value:
value = speed in m/s for treadmills m/min for ladder ergometers
- this speed is the value set by the coscom function S02
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user

```
extern "C" int __stdcall coscomsetSpeed(int portnumber, double value);
```

U Function (coscomsetAcceleration)

sets the device program acceleration index with port number 1 .. x
parameter value:
value = 0..7
- this acceleration index is the value set by the coscom function A01
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user

```
extern "C" int __stdcall coscomsetAcceleration(int portnumber, int value);
```

V Function (coscomsetSpeedAccel)

this function is the concatenation of the functions coscomsetAcceleration + coscomsetSpeed

```
extern "C" int __stdcall coscomsetSpeedAccel(int portnumber, double value, int accelerationindex);
```

W Function (coscomsetElevation)

sets the device program elevation with port number 1 .. x
parameter value:
value = elevation in % - (height / length) * 100%
- this elevation is the value set by the coscom function E03
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user

```
extern "C" int __stdcall coscomsetElevation(int portnumber, double value);
```

X Function (coscomKeyPressStart)

sends device key press start command with port number 1 .. x
parameter how:
0 == released
1 == pressed
- this key press is the value send by the coscom function U10
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user

```
extern "C" int __stdcall coscomKeyPressStart(int portnumber, int how);
```

Y Function (coscomKeyPressStop)

sends device key press stop command with port number 1 .. x
parameter how:
0 == released
1 == pressed
- this key press is the value send by the coscom function U10
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user

```
extern "C" int __stdcall coscomKeyPressStop(int portnumber, int how);
```

Z Function (coscomKeyPressPlus)

sends device key press + command with port number 1 .. x
parameter how:
0 == released
1 == pressed
- this key press is the value send by the coscom function U10
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user

```
extern "C" int __stdcall coscomKeyPressPlus(int portnumber, int how);
```

AA Function (coscomKeyPressMinus)

sends device key press - command with port number 1 .. x
parameter how:
0 == released
1 == pressed
- this key press is the value send by the coscom function U10
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user

```
extern "C" int __stdcall coscomKeyPressMinus(int portnumber, int how);
```

BB Function (coscomKeyPressDown)

sends device key press down command with port number 1 .. x
parameter how:
0 == released
1 == pressed
- this key press is the value send by the coscom function U10
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user

```
extern "C" int __stdcall coscomKeyPressDown(int portnumber, int how);
```

CC Function (coscomKeyPressUp)

sends device key press up command with port number 1 .. x
parameter how:
0 == released
1 == pressed
- this key press is the value send by the coscom function U10
return value -4 means invalid port number
return value -2 means the port is opened but the device can't be controlled
return value -3 means an error occurred during function call
return value 0 means the port isn't open
return value 1 means the device can controlled by user

```
extern "C" int __stdcall coscomKeyPressUp(int portnumber, int how);
```