



*-power in control*

# Advanced Controller Platform AxC 500



## External C CoDeSys libraries

DEIF A/S · Frisenborgvej 33 · DK-7800 Skive · Tel.: +45 9614 9614 · Fax: +45 9614 9615 · [info@deif.com](mailto:info@deif.com) · [www.deif.com](http://www.deif.com)

Document no.: 4189340767

**WIND**

## Revision

Revision	Author	Date	Description
A	CSK	2012-10-02	For version 1.0.0rc1.
B	CSK	2012-10-23	Updated to 1.0.0rc2.
C	CSK	2013-03-19	Updated to 1.0.0rc3. Bug fixes.

# Contents

<b>Revision</b>	<b>i</b>
<b>Contents</b>	<b>ii</b>
<b>1 AxC 500 External C CoDeSys Libraries</b>	<b>1</b>
1.1 Definitions . . . . .	1
1.2 Purpose . . . . .	1
1.3 Installation instructions . . . . .	1
1.4 Making the CoDeSys 3.x library . . . . .	7
1.5 Generating the m4 and c stub files . . . . .	12
1.6 Example: Implementing you own C algorithms . . . . .	13
1.7 Using C several source files . . . . .	15
1.8 Building the external library . . . . .	15
1.9 Using the command prompt to build . . . . .	17
1.10 Installing the library on AxC 500 . . . . .	18
1.11 Using the external library . . . . .	19

# 1 AxC 500 External C CoDeSys Libraries

## 1.1 Definitions

AxC 500 is a platform acronym for both DEIF AWC 500 and AMC 500 controllers.

## 1.2 Purpose

The AxC 500 External C CoDeSys Lib Development package enables easy compilation of C stub files exported by CoDeSys, with the users added implementations, which are used for creating external CoDeSys libraries. External libraries are written in C and run on native target rather than the usual libraries written in the IEC 61131-3 languages and run by the CoDeSys runtime. This can be e.g. control algorithms written in C programming language:

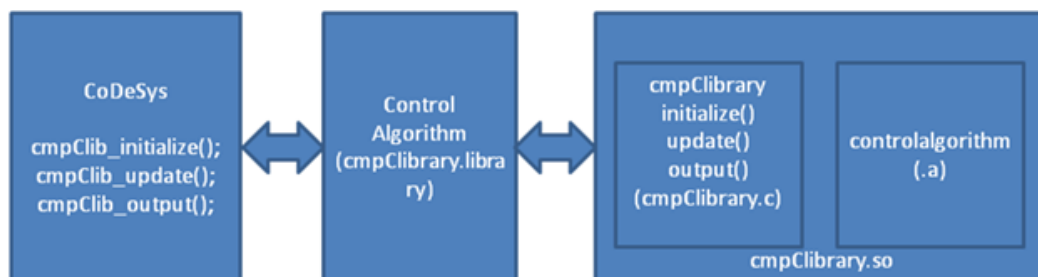


Figure 1.1: Interface between CoDeSys and External C libraries( compiled to linux.so files)

The C file(s) are compiled to the AxC 500 processor and linked into a .so file suitable for the Linux OS. The package will also archive the .so file together with an install script in a .dupdate file, which then can be easily installed on each AxC 500 in the production. Finally the package can help uploading the .dupdate file to the AxC 500 by having the user provide IP, username and password.

## 1.3 Installation instructions

Login to <ftp://support.deif.com>, download the installer (DEIF\_AxC\_500\_external\_c\_lib\_dev\_package\_vx.y.z.exe) which is approximately 200 MB:

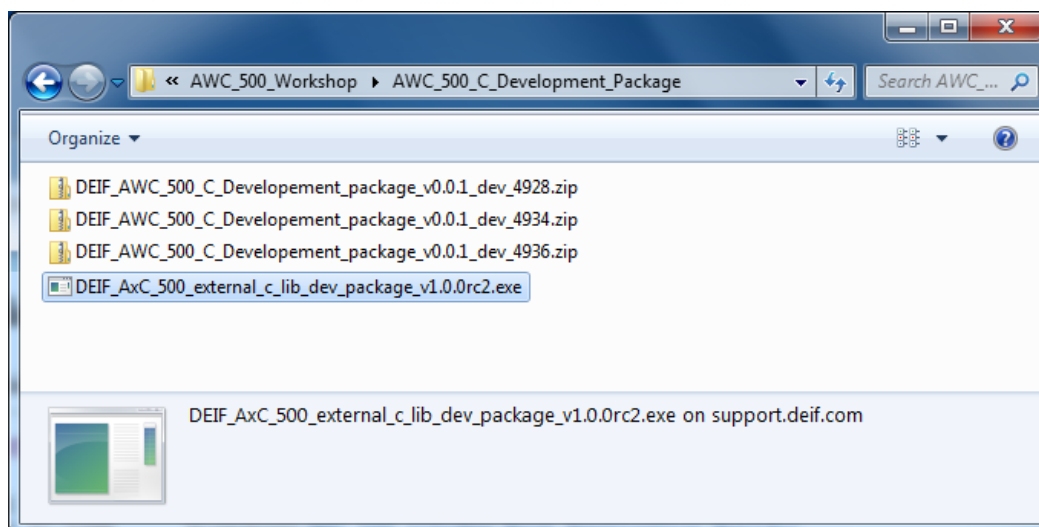


Figure 1.2: AxC 500 support site to download the package

Then run the Installer. If prompted, click "Yes" to accept elevated access:

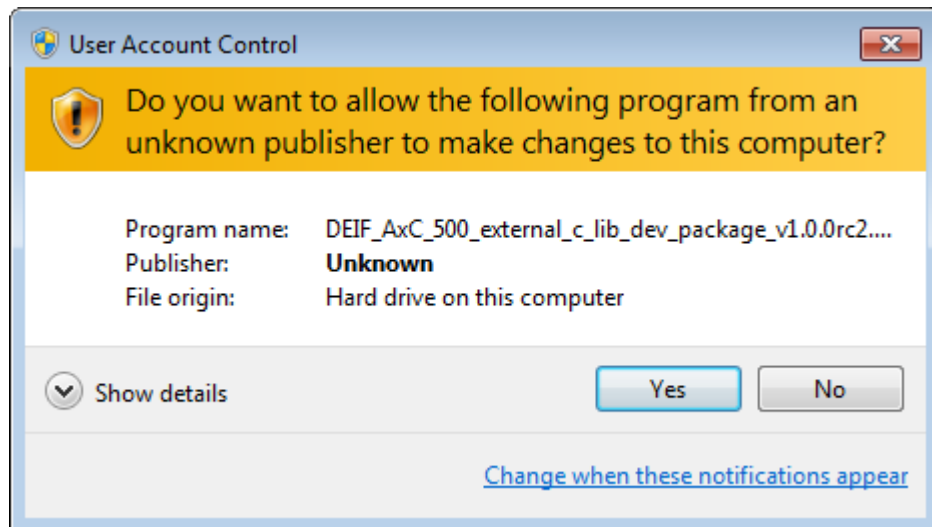


Figure 1.3: Running the Installation package

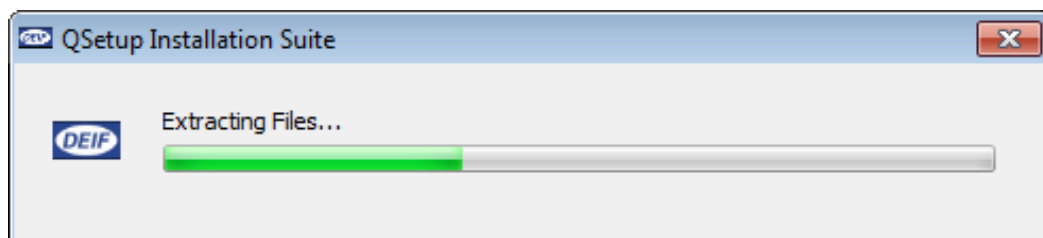


Figure 1.4: Running the Installation package

Press "Next" at the welcome screen and follow the installer guidelines:



Figure 1.5: Running the Installation package

Press "Next":

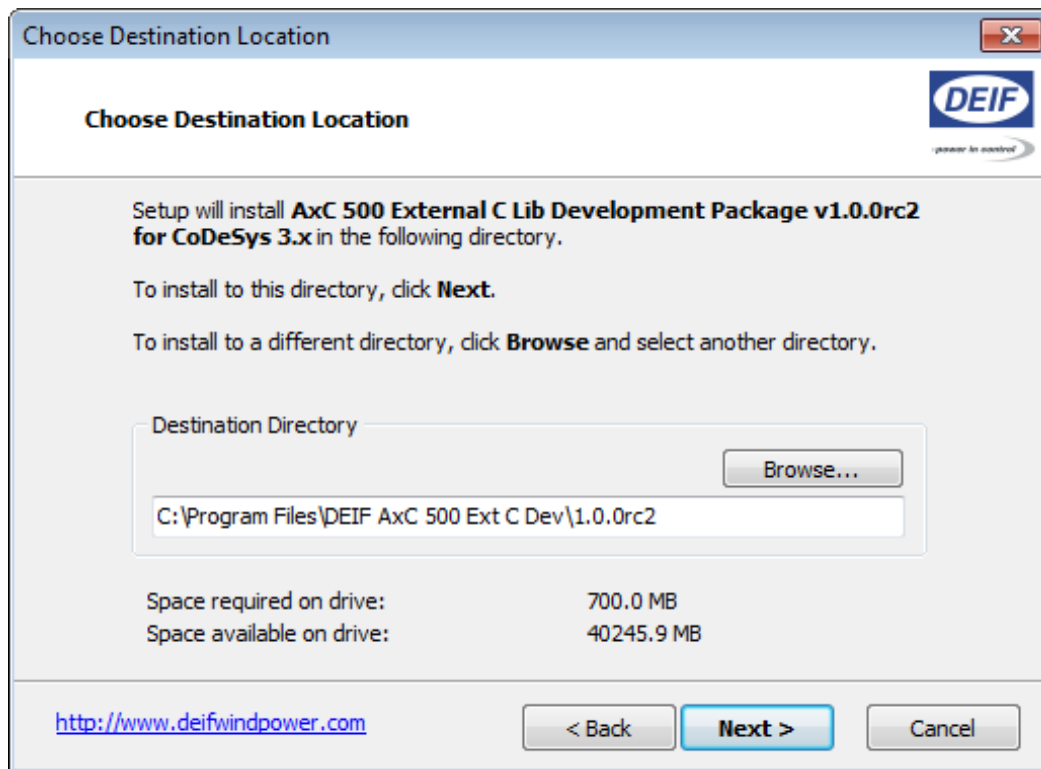


Figure 1.6: Chose destination.

Press "Next":

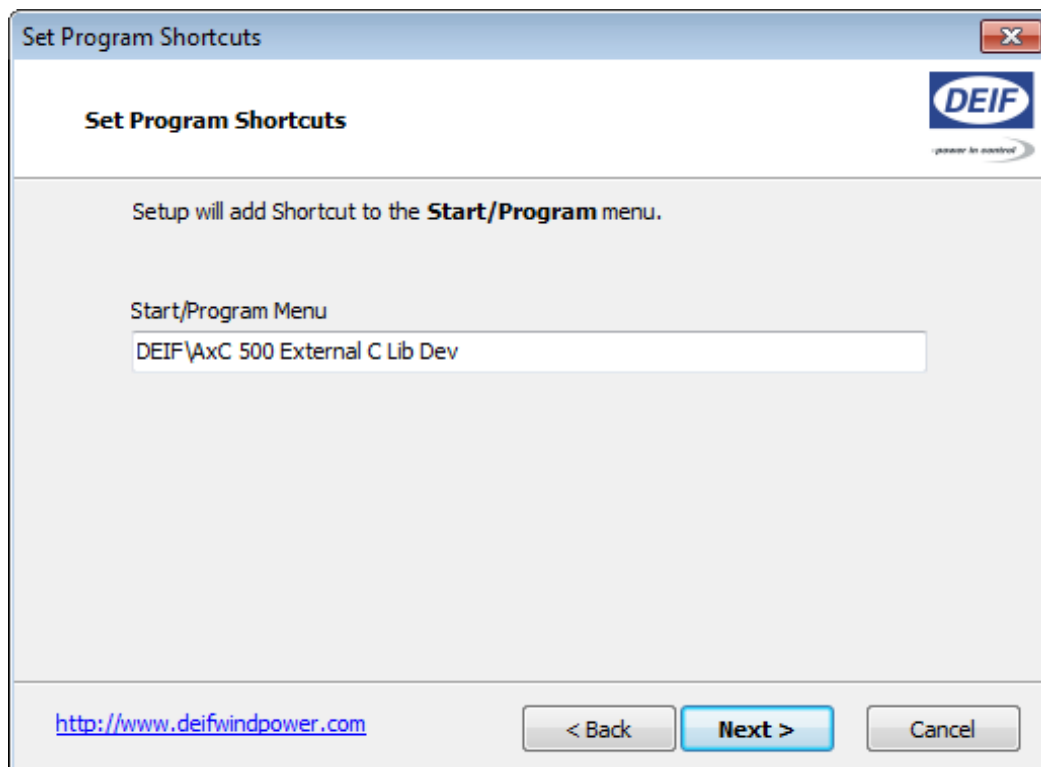


Figure 1.7

Press "Next":

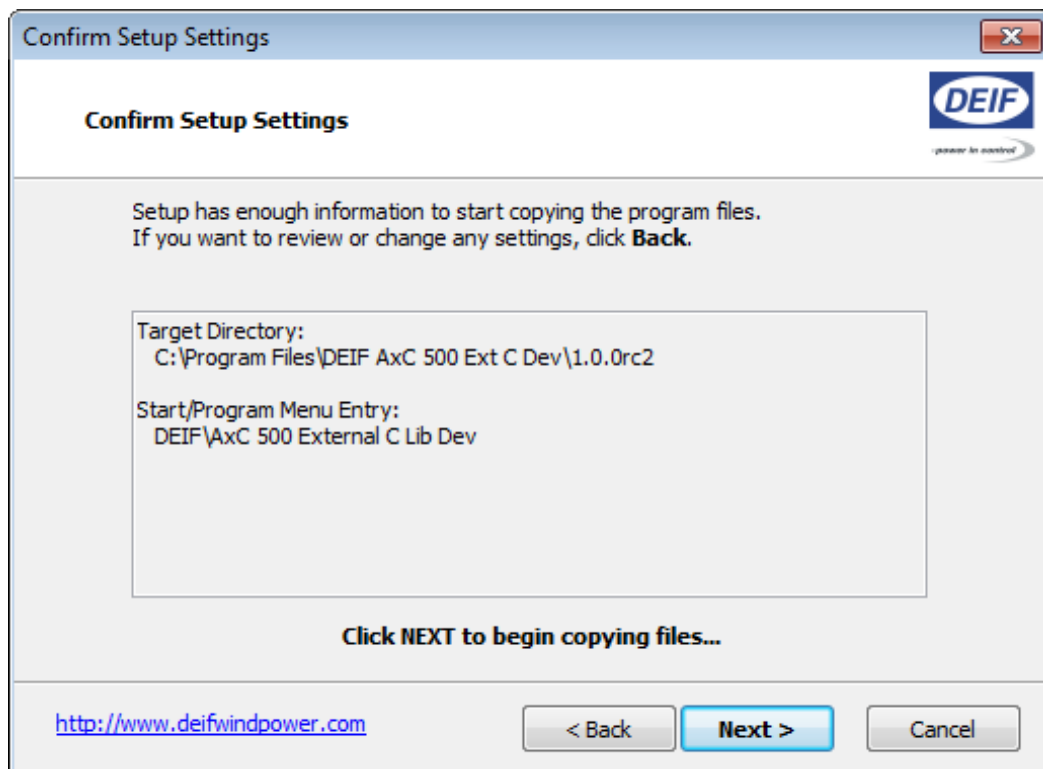


Figure 1.8

The installer will then copy the files to the PC:

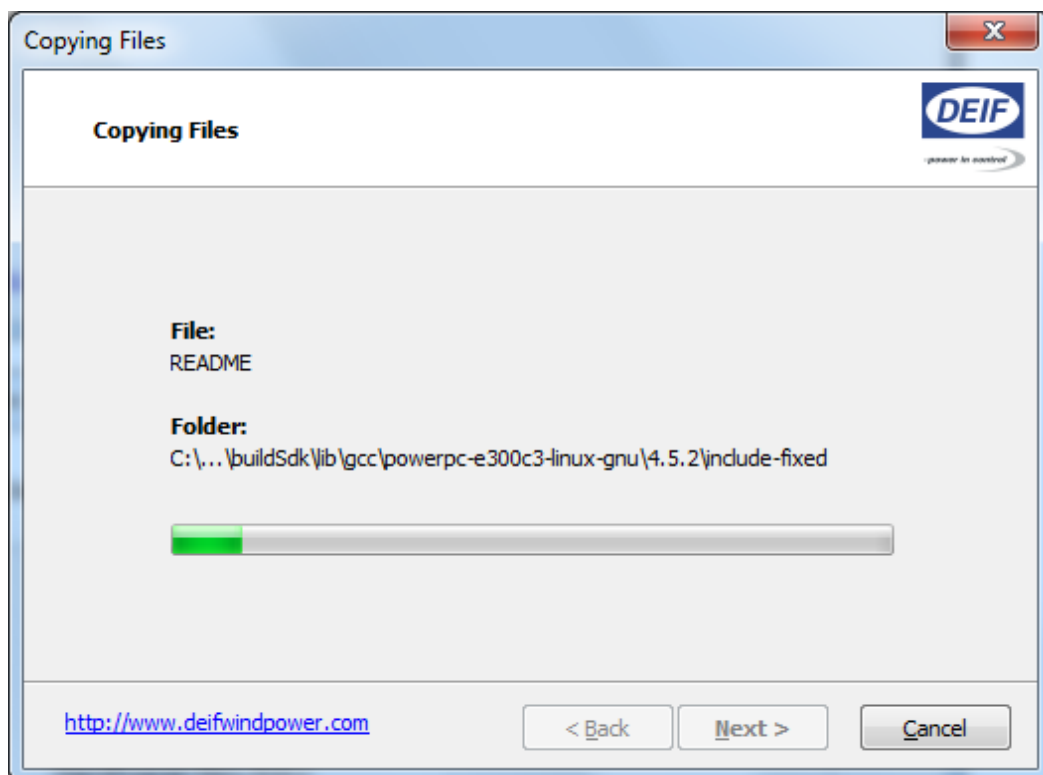


Figure 1.9

You will be prompted to install Cygwin. If you already have installed Cygwin, this step can be skipped, but it is recommended to click "Yes". If you clicked "No" you may be prompted to locate the Cygwin path. If no Cygwin was found or if some Cygwin parts are missing, you should run the installer again and click "Yes".

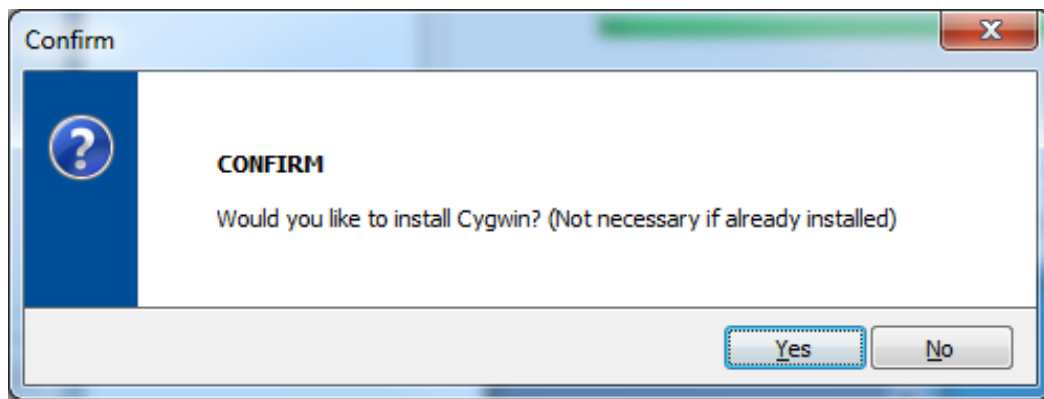


Figure 1.10

The Cygwin installer will open in a separate window and will run automatically. Only the components necessary will be installed and only the parts not already installed:

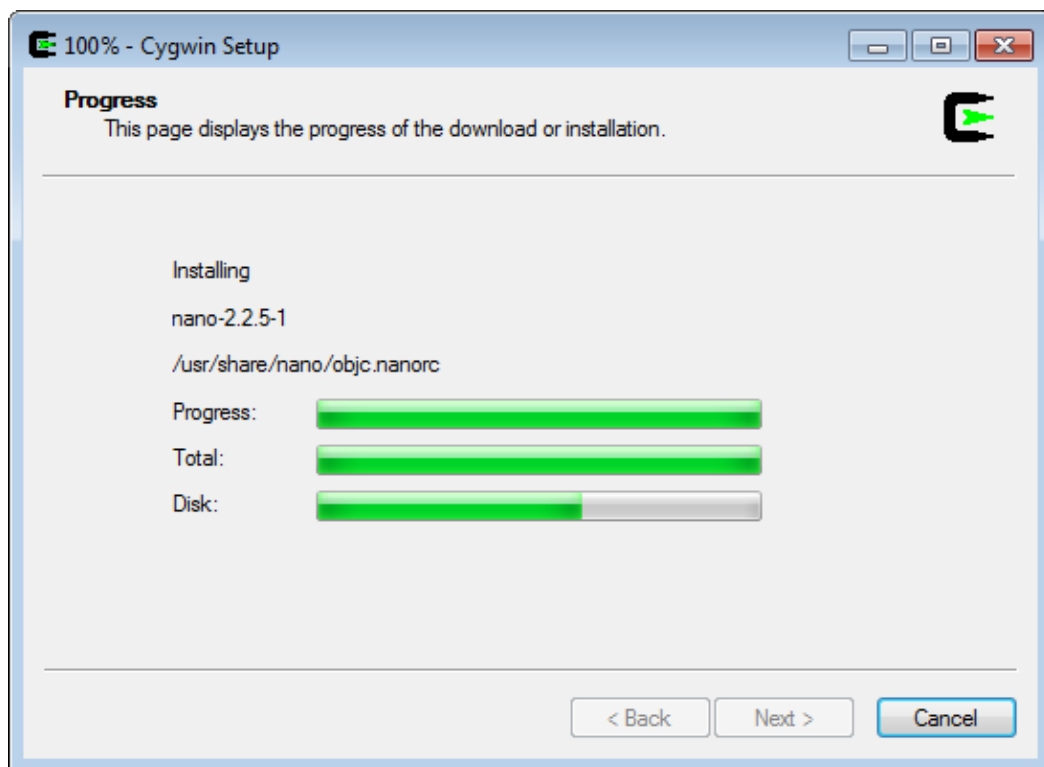


Figure 1.11

Here after the setup is finished:



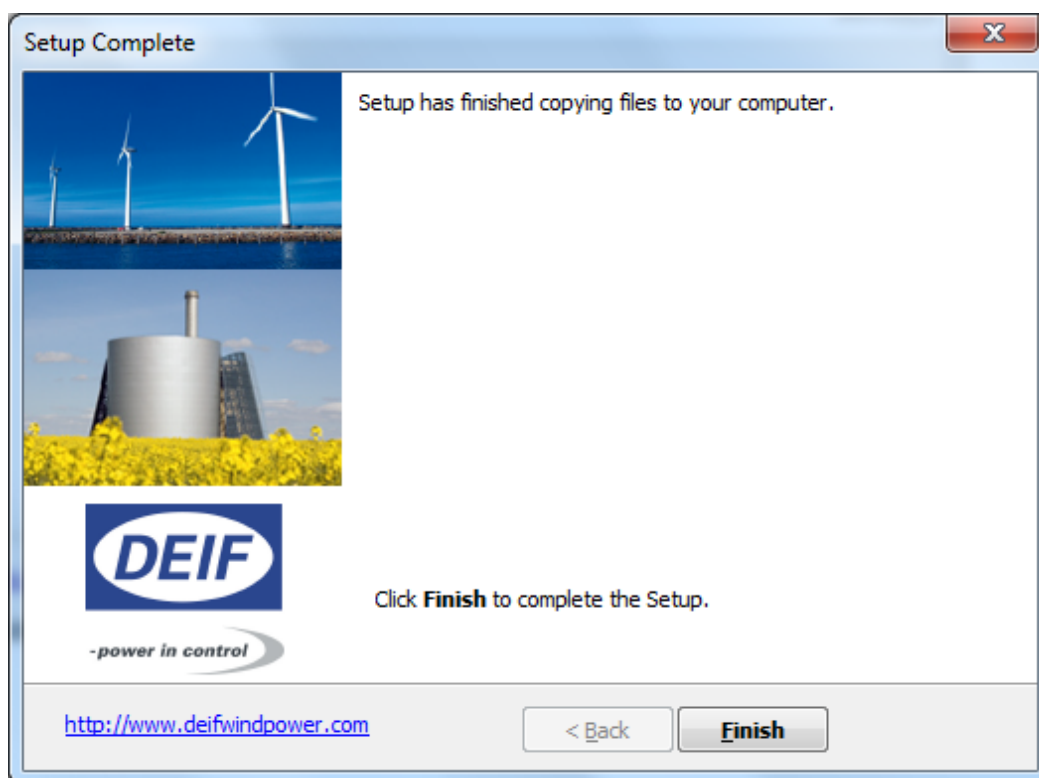


Figure 1.12

## 1.4 Making the CoDeSys 3.x library

First open CoDeSys and make a new library. This documentation will be based on the example library "TestLib":

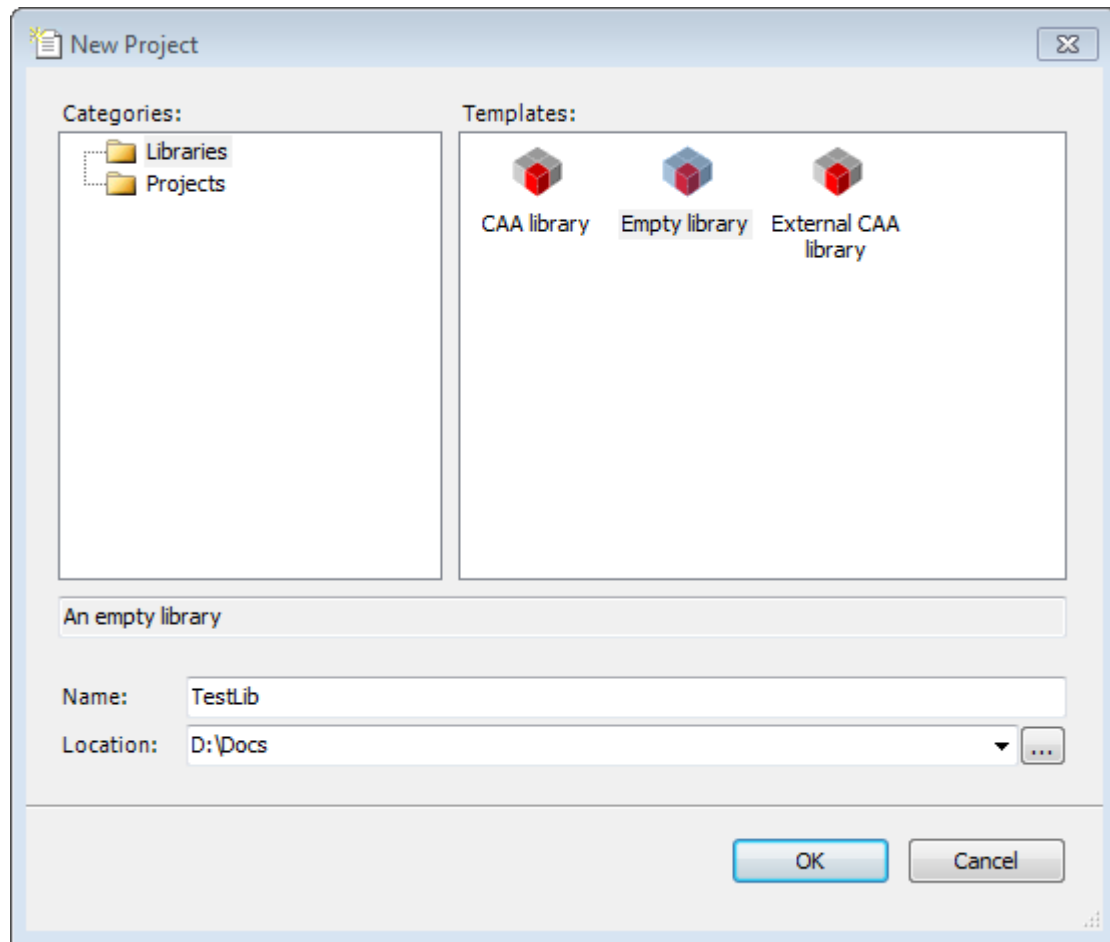


Figure 1.13

Add and implement the interface (IN/OUT variables) for all needed POU's:

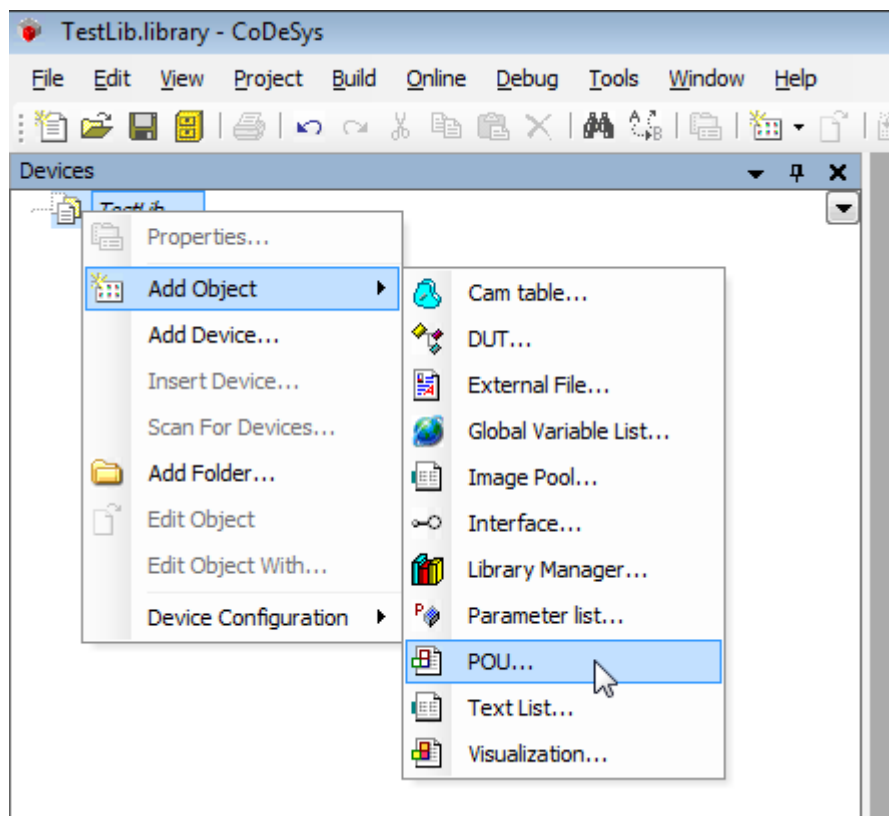


Figure 1.14

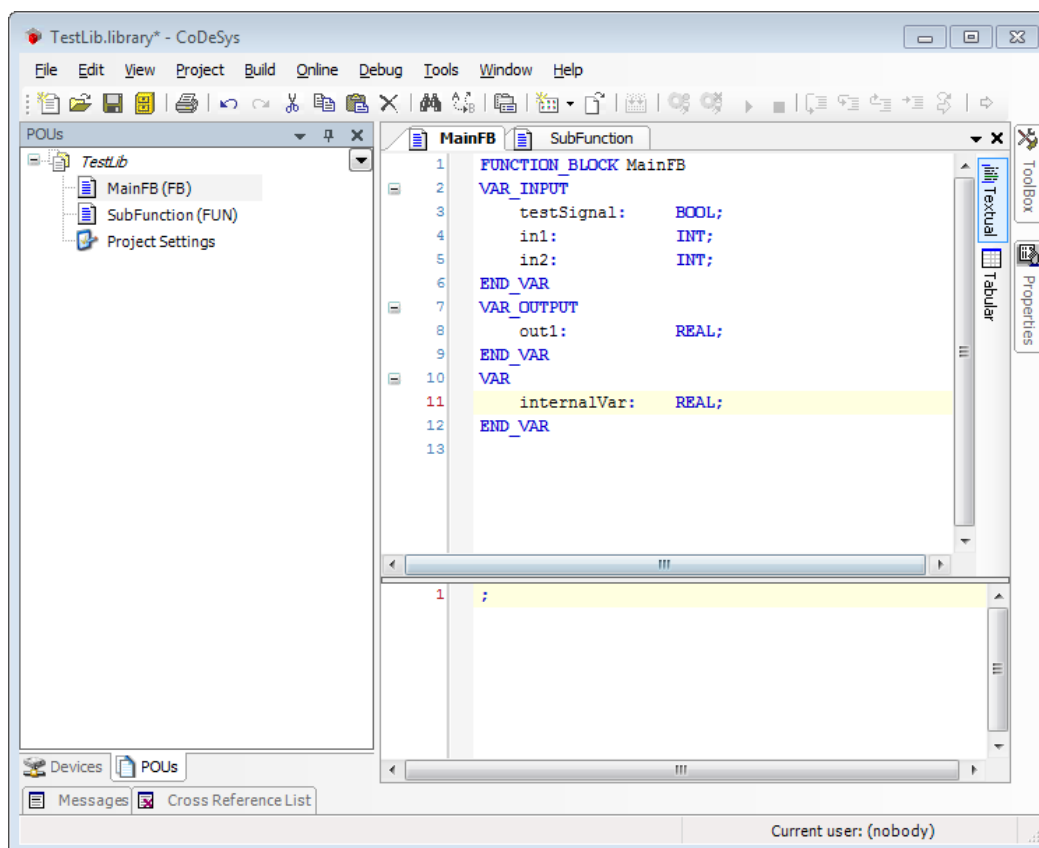


Figure 1.15

Now go to "Project Information" via the "Project" menu:

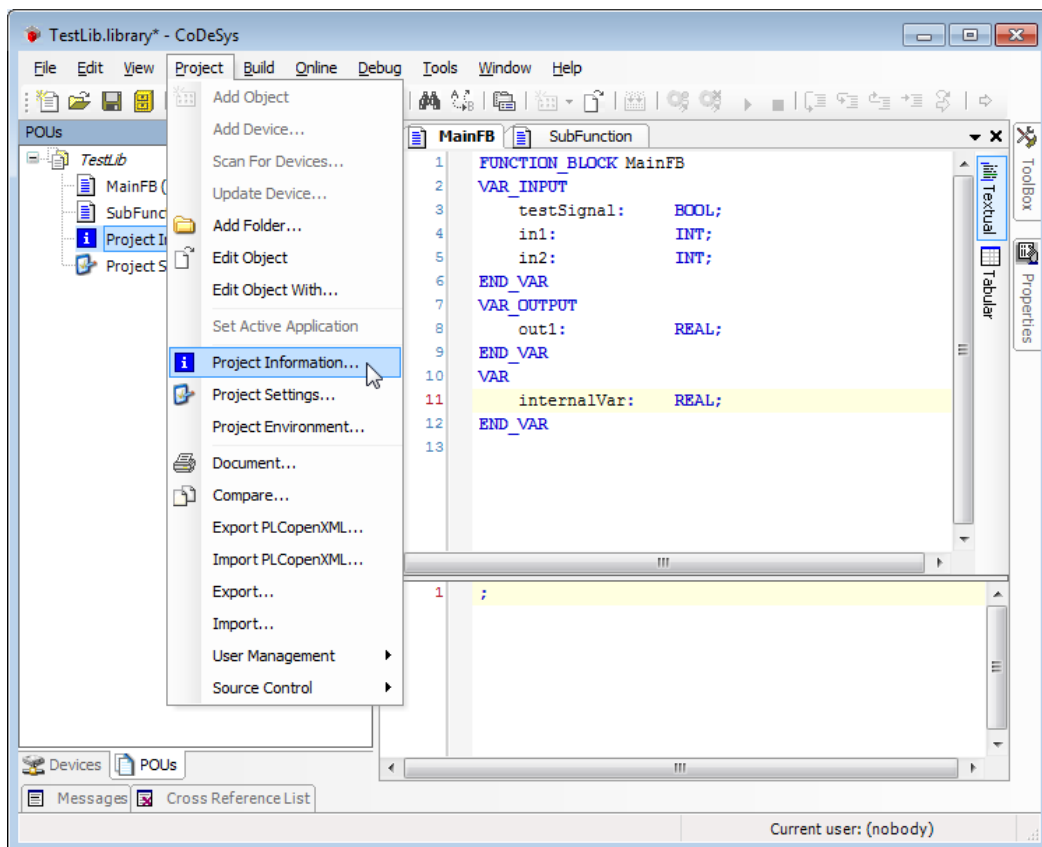


Figure 1.16

Fill out the required info as a minimum:

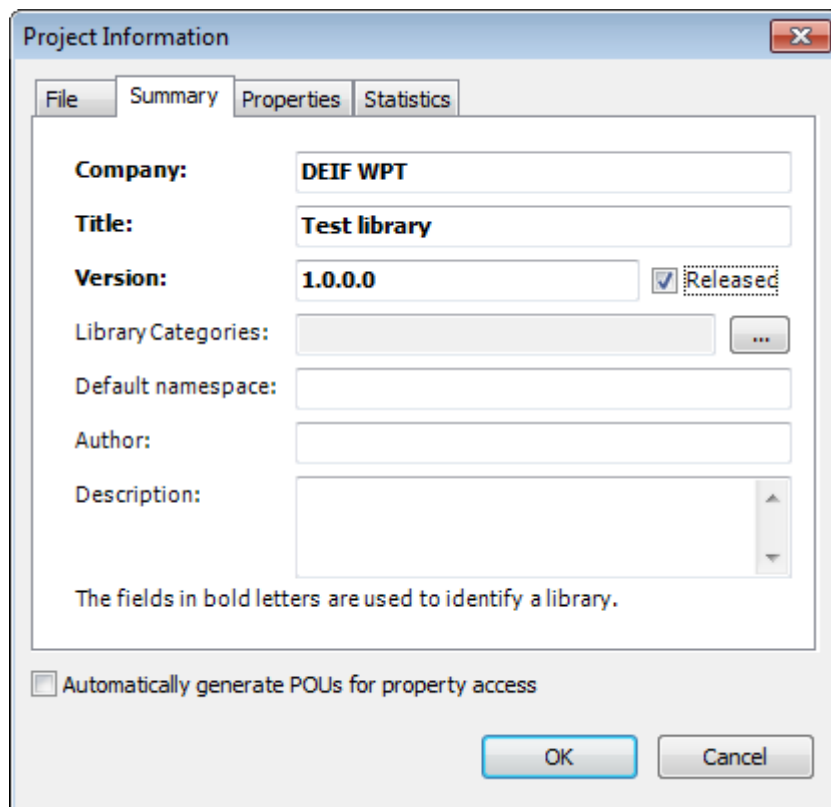


Figure 1.17

Now go to properties for each POU and set the "External implementation" flag at the "Build" pane:

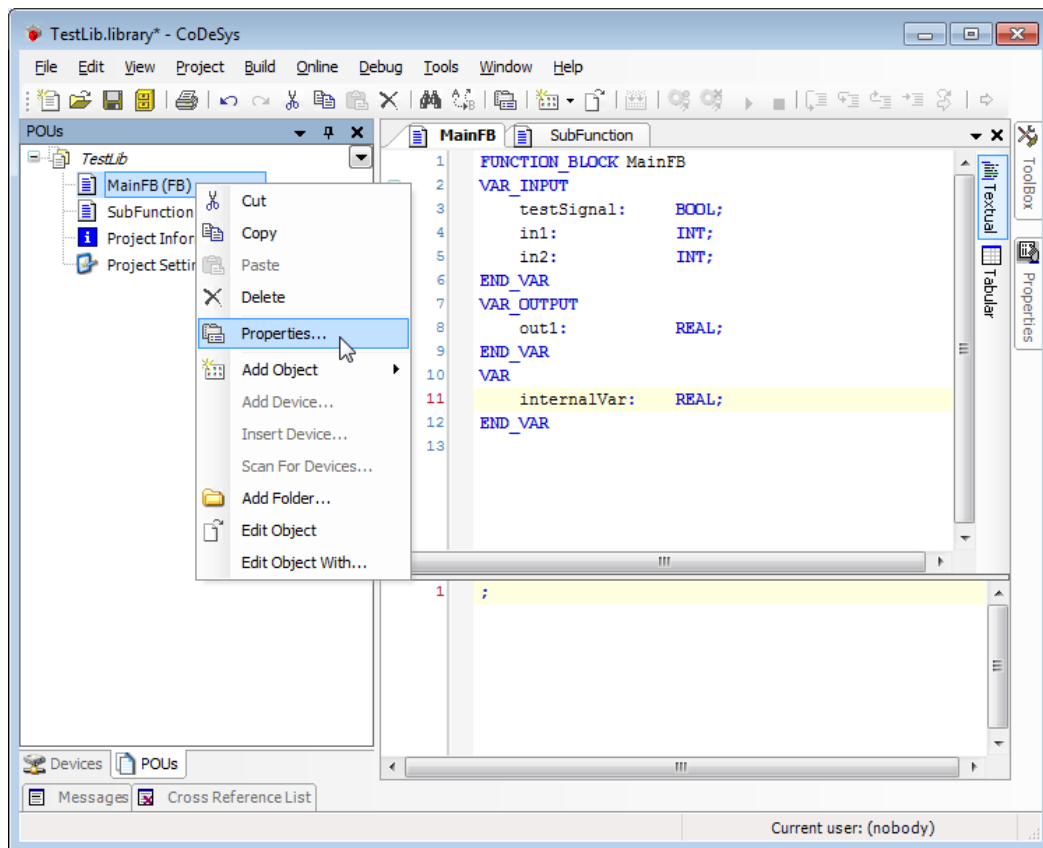


Figure 1.18

Set "External implementation":

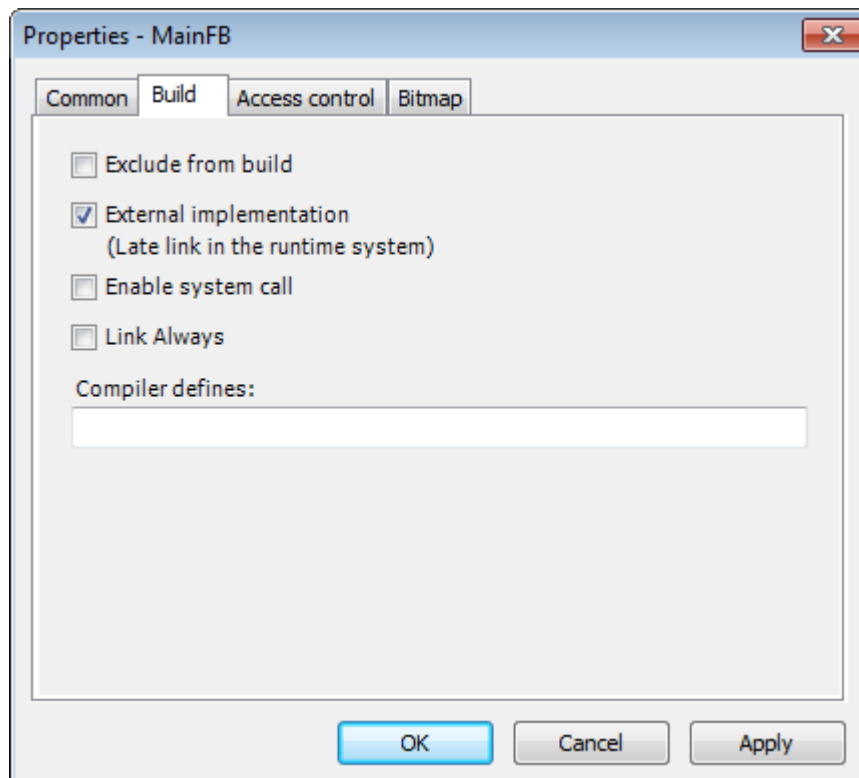



Figure 1.19

Finally "Save Project And Install Into Library Repository" found in the "File" menu or via the icon:  .

## 1.5 Generating the m4 and c stub files

CoDeSys can help providing code stubs for the POUs which should be used to add their implementations to. To generate these files click "Build runtime system files..." from the "Build" menu:

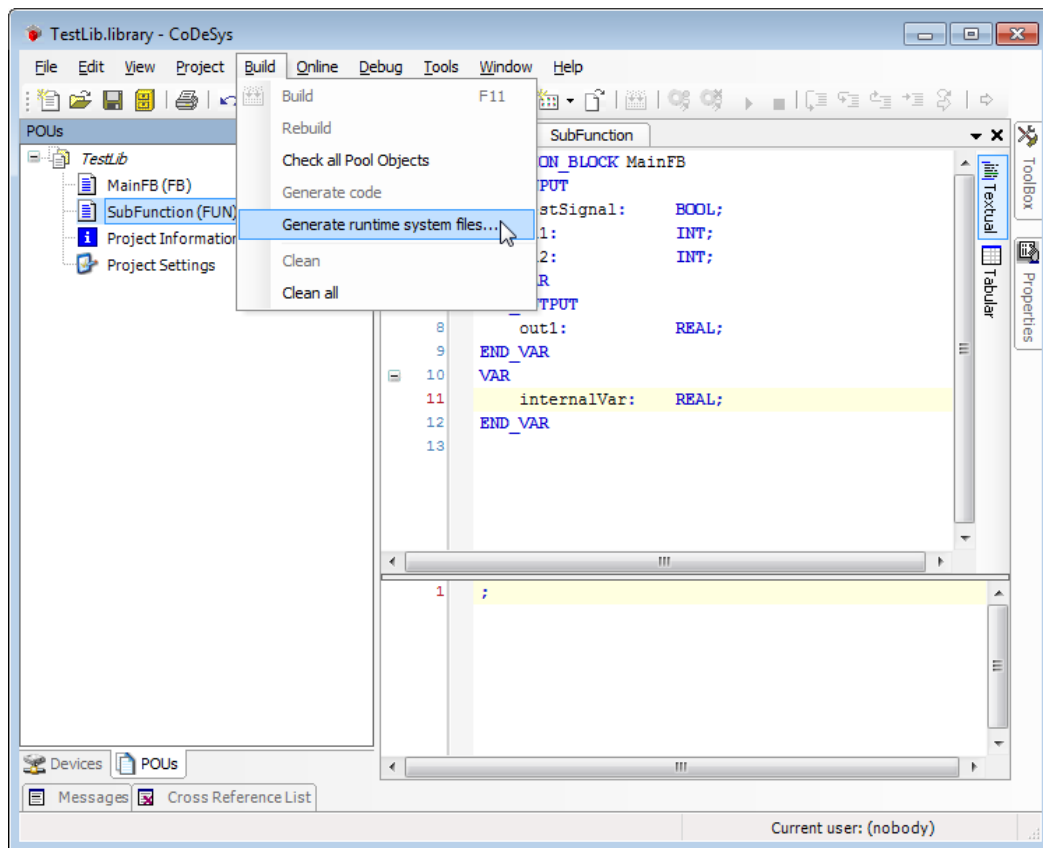


Figure 1.20

Set the output directory and make sure both files are checked:

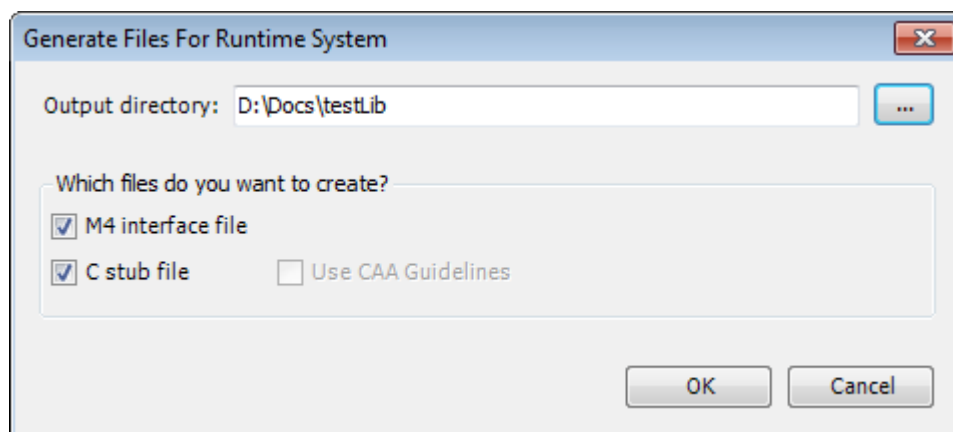
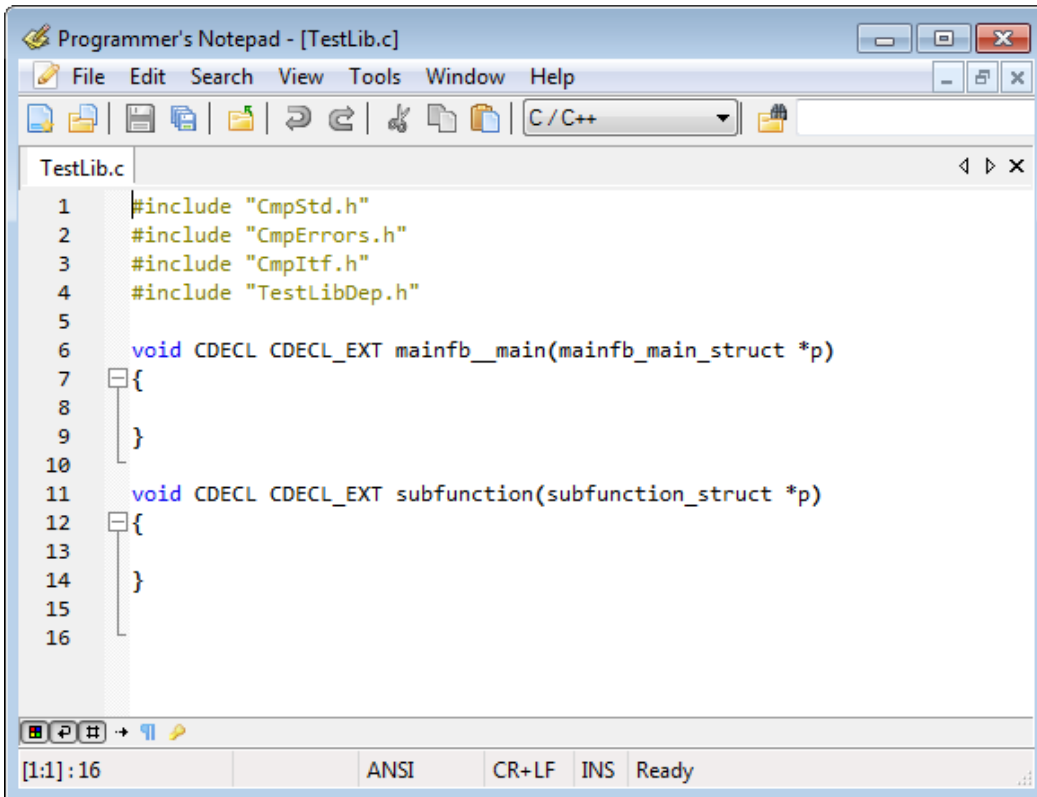


Figure 1.21

Now the C stub file is ready to be modified manually with implementation of the library functions or interface and integrate existing c files.

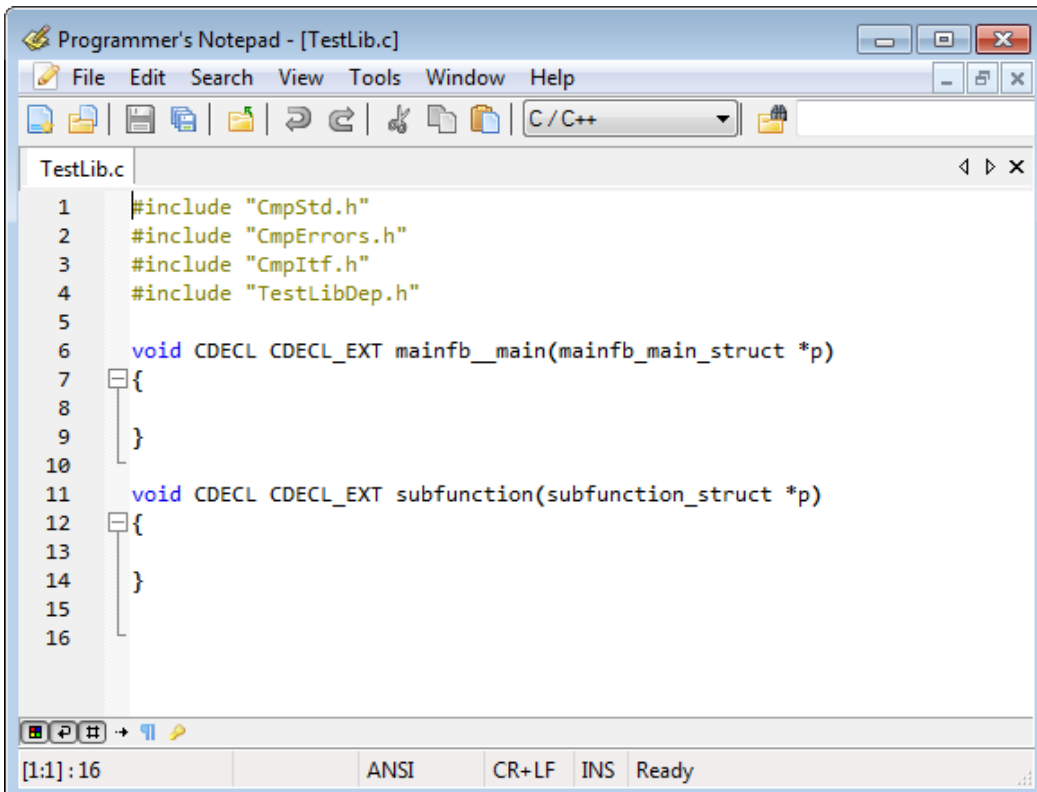
## 1.6 Example: Implementing you own C algorithms

The C file generated by CoDeSys should be modified to implement the functionality desired. Each program, function block or function has its own function skeleton in the c stub:



```
Programmer's Notepad - [TestLib.c]
File Edit Search View Tools Window Help
TestLib.c
1  #include "CmpStd.h"
2  #include "CmpErrors.h"
3  #include "CmpItf.h"
4  #include "TestLibDep.h"
5
6  void CDECL CDECL_EXT mainfb__main(mainfb_main_struct *p)
7  {
8
9  }
10
11 void CDECL CDECL_EXT subfunction(subfunction_struct *p)
12 {
13
14 }
15
16
[1:1]: 16 ANSI CR+LF INS Ready
```

Figure 1.22



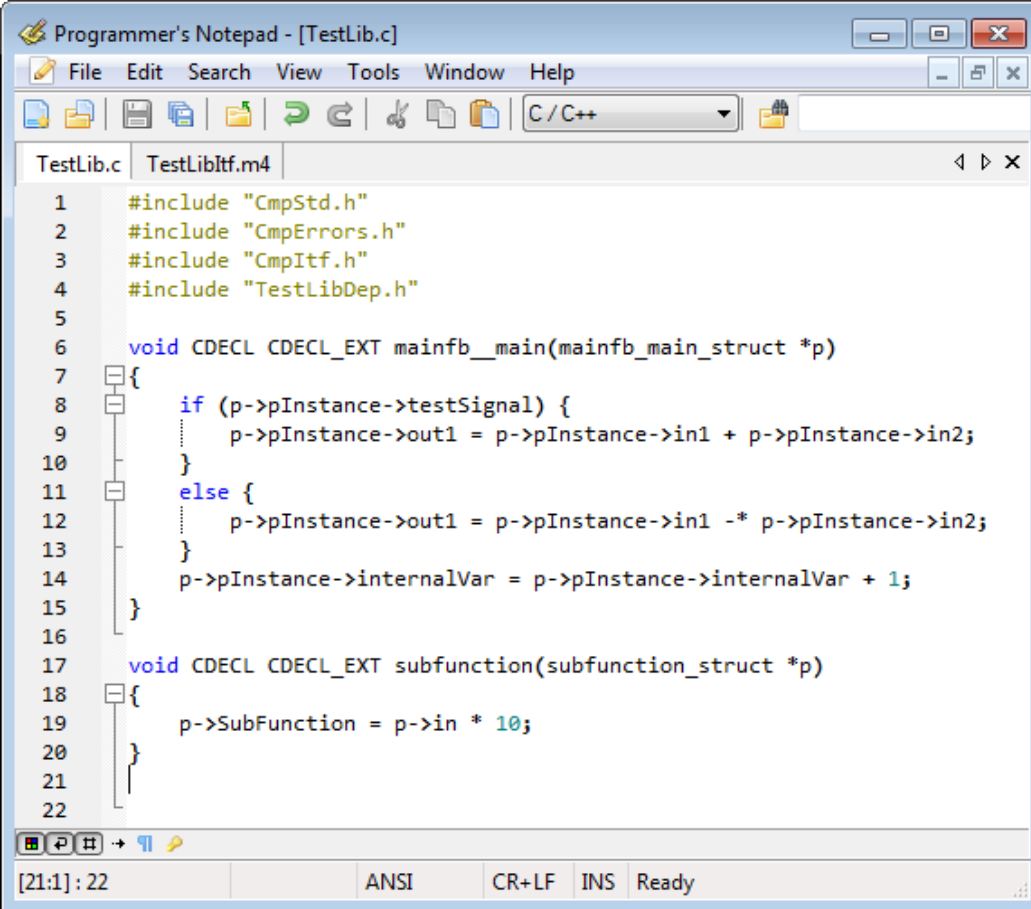
```
Programmer's Notepad - [TestLib.c]
File Edit Search View Tools Window Help
TestLib.c
1  #include "CmpStd.h"
2  #include "CmpErrors.h"
3  #include "CmpItf.h"
4  #include "TestLibDep.h"
5
6  void CDECL CDECL_EXT mainfb__main(mainfb_main_struct *p)
7  {
8
9  }
10
11 void CDECL CDECL_EXT subfunction(subfunction_struct *p)
12 {
13
14 }
15
16
[1:1]: 16 ANSI CR+LF INS Ready
```

Figure 1.23



Implement each function with the desired code. The inputs, outputs and internal variables can be found in "p" which points to a structure containing all variables. See the ".m4" file for the structure.

In our example the implementation is the following:



```
1  #include "CmpStd.h"
2  #include "CmpErrors.h"
3  #include "CmpItf.h"
4  #include "TestLibDep.h"
5
6  void CDECL CDECL_EXT mainfb__main(mainfb_main_struct *p)
7  {
8      if (p->pInstance->testSignal) {
9          .....
10         p->pInstance->out1 = p->pInstance->in1 + p->pInstance->in2;
11     }
12     else {
13         .....
14         p->pInstance->out1 = p->pInstance->in1 - * p->pInstance->in2;
15     }
16     p->pInstance->internalVar = p->pInstance->internalVar + 1;
17 }
18
19 void CDECL CDECL_EXT subfunction(subfunction_struct *p)
20 {
21     p->SubFunction = p->in * 10;
22 }
```

[21:1] : 22      ANSI      CR+LF      INS      Ready

Figure 1.24

## 1.7 Using C several source files

It is possible to have several source files. Just make sure to include the header and keep all the sources (h and c files) in the same directory as the c stub and m4 files. Then the tool chain will automatically compile and link them.

## 1.8 Building the external library

The building process is initialized by clicking "Build External CoDeSys C library" from the start menu: Start->DEIF->AxC 500 External C lib Dev:

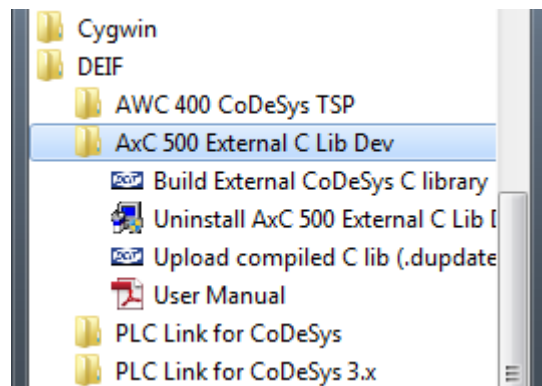


Figure 1.25

This will initiate a tool chain which will ask for the location of the c stub file previously implemented:

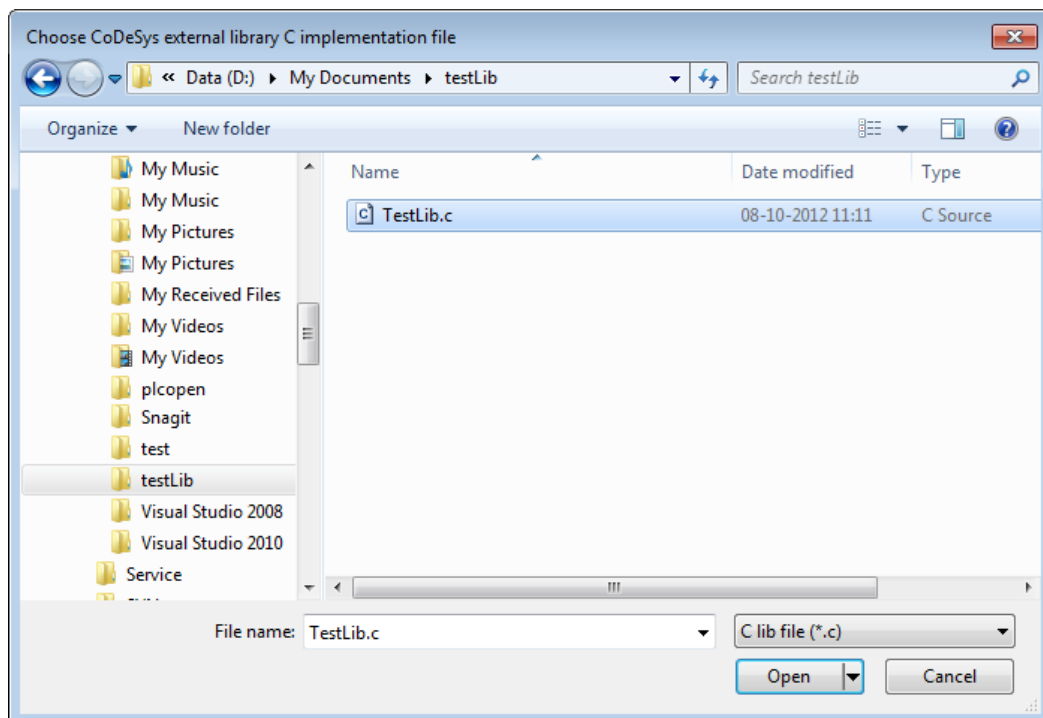


Figure 1.26

The tool chain will now begin compiling your library. Any errors occurring will be shown in the command window, e.g. if you have made syntax errors in the code or if some parts of Cygwin are missing:

```

Build External CoDeSys C library
#####
Building *Dep.m4
### Generating Dep file "TestLibDep.m4"
### Generating C include file "TestLibInclude.c"
### Generating make file "Makefile"
#####
Building *Dep.m4 finished successfully
#####
### Generating m4 files and headers
1 file(s) copied.
Dos2Unix: Processing file C:\Program Files\DEIF AWC 500 Ext C Dev\0.0.3 dev\buildTools\tempfiles\TestLibItf_...
Dos2Unix: Processing file C:\Program Files\DEIF AWC 500 Ext C Dev\0.0.3 dev\buildTools\tempfiles\TestLibItf.h ...
Unix2Dos: Processing file C:\Program Files\DEIF AWC 500 Ext C Dev\0.0.3 dev\buildTools\tempfiles\TestLibItf.h ...
includes -I/M4Defs -I.../Components -I"C:\Program Files\DEIF AWC 500 Ext C Dev\0.0.3 dev\buildTools\tempfiles"
1 file(s) copied.
Dos2Unix: Processing file C:\Program Files\DEIF AWC 500 Ext C Dev\0.0.3 dev\buildTools\tempfiles\TestLibDep_...
Dos2Unix: Processing file C:\Program Files\DEIF AWC 500 Ext C Dev\0.0.3 dev\buildTools\tempfiles\TestLibDep.h ...
Unix2Dos: Processing file C:\Program Files\DEIF AWC 500 Ext C Dev\0.0.3 dev\buildTools\tempfiles\TestLibDep.h ...
Dos2Unix: Processing file C:\Program Files\DEIF AWC 500 Ext C Dev\0.0.3 dev\buildTools\tempfiles\TestLibDep.cpp ...
rm -f TestLib.o TestLibInclude.o *~ core .depend TestLib.tar.bz2 libTestLib.so.0.1 libTestLib.so libTestLib.so.0
powerpc-e300c3-linux-gnu-cc -DTRG_PPC -DALIGNATTRIB= -DCDECL= -DHANDLE_WIN32_PRAGMA -DDL_DECL= -D_REENTRANT= -DHUGEPT=
Program Files\DEIF AWC 500 Ext C Dev\0.0.3 dev\codesysRITK\Components" -I"C:\Program Files\DEIF AWC 500 Ext C Dev\0.0.3
x" -DLINUX -g -DCAALib -DCAALIB_ENDIAN -DMOTOROLA_BYTE_ORDER -M TestLib.c TestLibInclude.c > .depend
make: Circular TestLib.o < TestLib.o afhangighed opgive.
powerpc-e300c3-linux-gnu-cc -fPIC -c -DTRG_PPC -DALIGNATTRIB= -DCDECL= -DHANDLE_WIN32_PRAGMA -DDL_DECL= -D_REENTRANT= -
ed -I"C:\Program Files\DEIF AWC 500 Ext C Dev\0.0.3 dev\codesysRITK\Components" -I"C:\Program Files\DEIF AWC 500 Ext C D
orms\Linux" -DLINUX -g -DCAALib -DCAALIB_ENDIAN -DMOTOROLA_BYTE_ORDER -o TestLibInclude.o TestLibInclude.c
powerpc-e300c3-linux-gnu-cc -fPIC -c -DTRG_PPC -DALIGNATTRIB= -DCDECL= -DHANDLE_WIN32_PRAGMA -DDL_DECL= -D_REENTRANT= -
ed -I"C:\Program Files\DEIF AWC 500 Ext C Dev\0.0.3 dev\codesysRITK\Components" -I"C:\Program Files\DEIF AWC 500 Ext C D
orms\Linux" -DLINUX -g -DCAALib -DCAALIB_ENDIAN -DMOTOROLA_BYTE_ORDER -o TestLib.o TestLib.c
TestLib.c: In function 'mainfb_main':
TestLib.c:12:43: error: invalid type argument of unary '*' have 'int'
make: *** [TestLib.o] Fejl 1
#####
Building m4 header and cpp files failed
#####
Making .so file failed
#####
Compiling library and making .dupdate file failed
#####
Press any key to continue . . .

```

Figure 1.27

You will need to repeat the process after correcting the error. If the process went fine you will be prompted a dialog asking to download the compiled library to an AxC 500 :

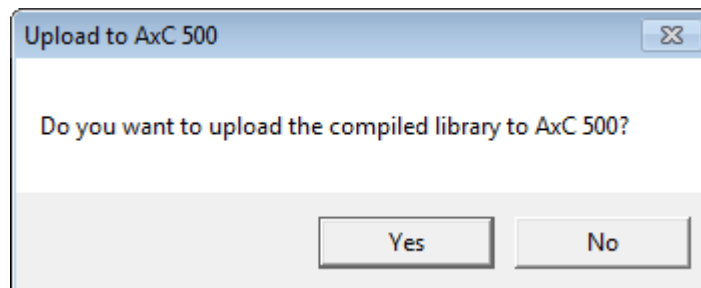


Figure 1.28

At this time the compilation has finished and the compiled library has been copied to the source folder. The compiled file has the library name, version and ".dupdate" as extension. The dUpdate file is an archive which can be easily installed on the AxC 500 just by uploading it to tmpfwupdates.

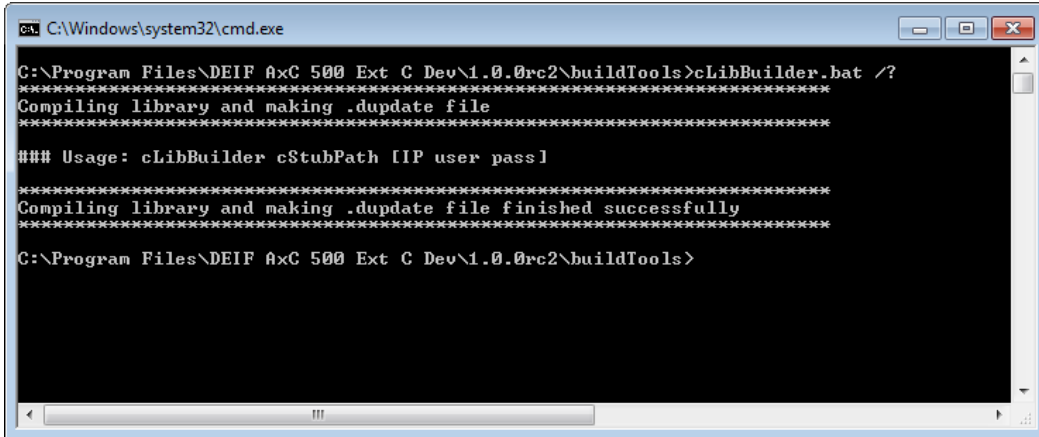


**When uploading a library to the AxC 500 , the CoDeSys runtime will be restarted, so make sure all operation has been stopped prior to uploading.**

Clicking "Yes" will initiate an uploading session. See next section. Clicking "No" will leave the .dupdate for manual or later uploading.

## 1.9 Using the command prompt to build

It is possible to avoid the user prompting of the source files and whether to upload to the AxC 500 by calling the tool chain from the command with arguments. The usage is: `cLibBuilder "path to c stub file" [IP username password]` If the last part in square brackets is omitted, the dUpdate archive is not uploaded to the AxC 500. Go to the install folder and navigate further to "buildTools" to find "cLibBuilder.bat". To display the usage type "cLibBuilder ?":



```
C:\Windows\system32\cmd.exe
C:\Program Files\DEIF AxC 500 Ext C Dev\1.0.0rc2\buildTools>cLibBuilder.bat /?
*****
Compiling library and making .dupdate file
*****
### Usage: cLibBuilder cStubPath [IP user pass]
*****
Compiling library and making .dupdate file finished successfully
*****
C:\Program Files\DEIF AxC 500 Ext C Dev\1.0.0rc2\buildTools>
```

Figure 1.29

## 1.10 Installing the library on AxC 500

Installing the library can be done automatically after a compilation or manually by the start menu Start->DEIF->AxC 500 External C lib Dev->Upload compiled C lib (.dupdate) to AxC 500. When using the manual approach you will be prompted a file dialog for choosing the .dupdate file. Next you will be asked the IP, username and password of the AxC 500 :

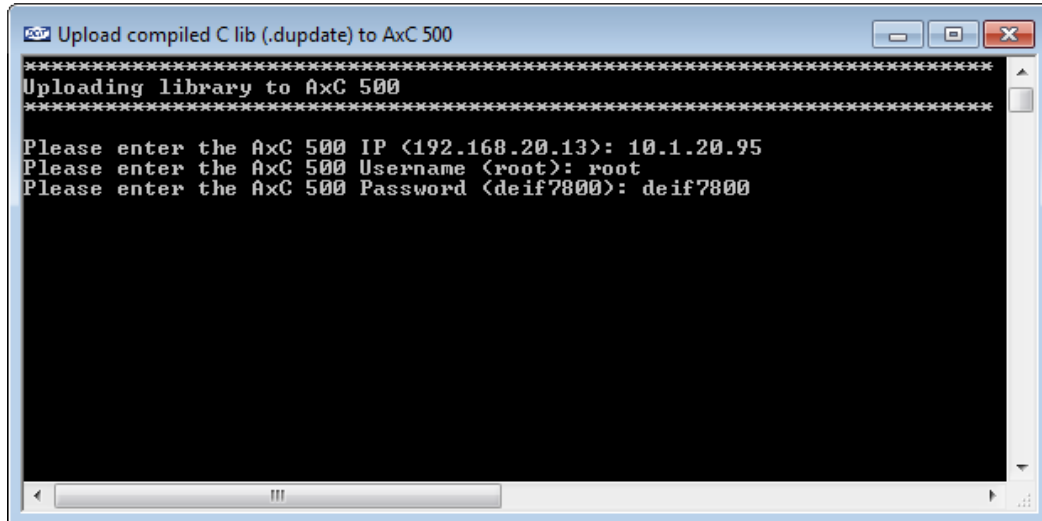


Figure 1.30

Next the tool will upload the dUpdate file and it will be installed automatically:

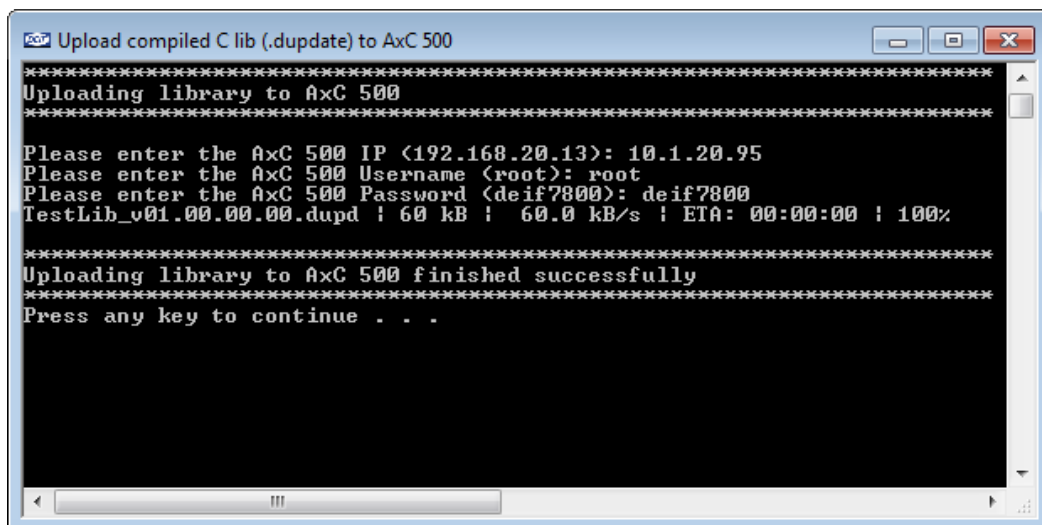


Figure 1.31

The library is now ready to be used in CoDeSys and on AxC 500 .

## 1.11 Using the external library

Add and use the library in a new or existing AxC 500 project:

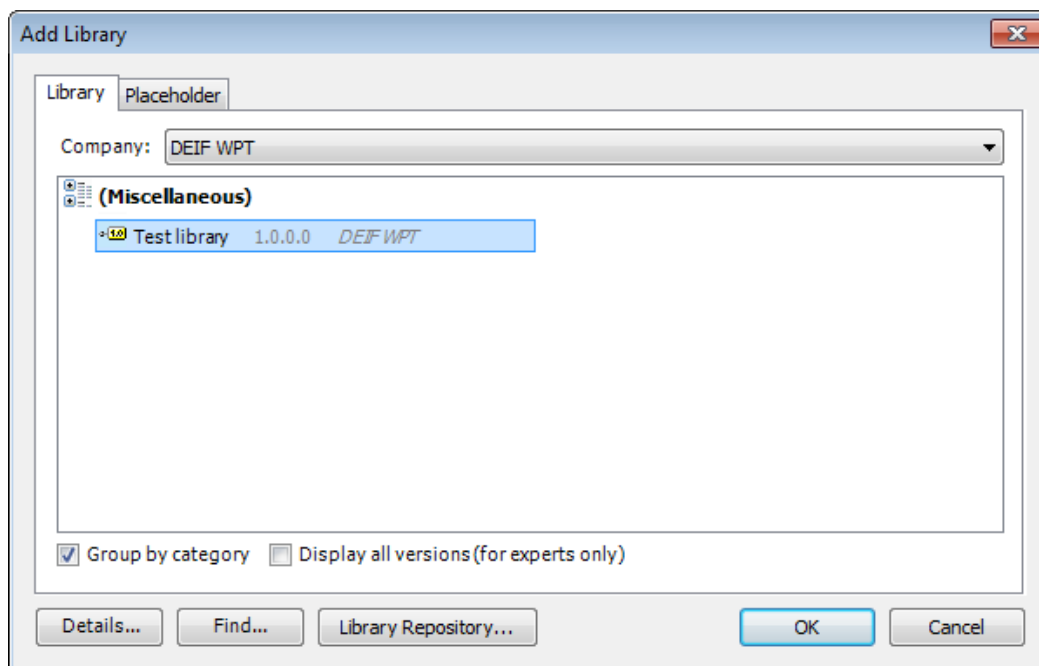


Figure 1.32

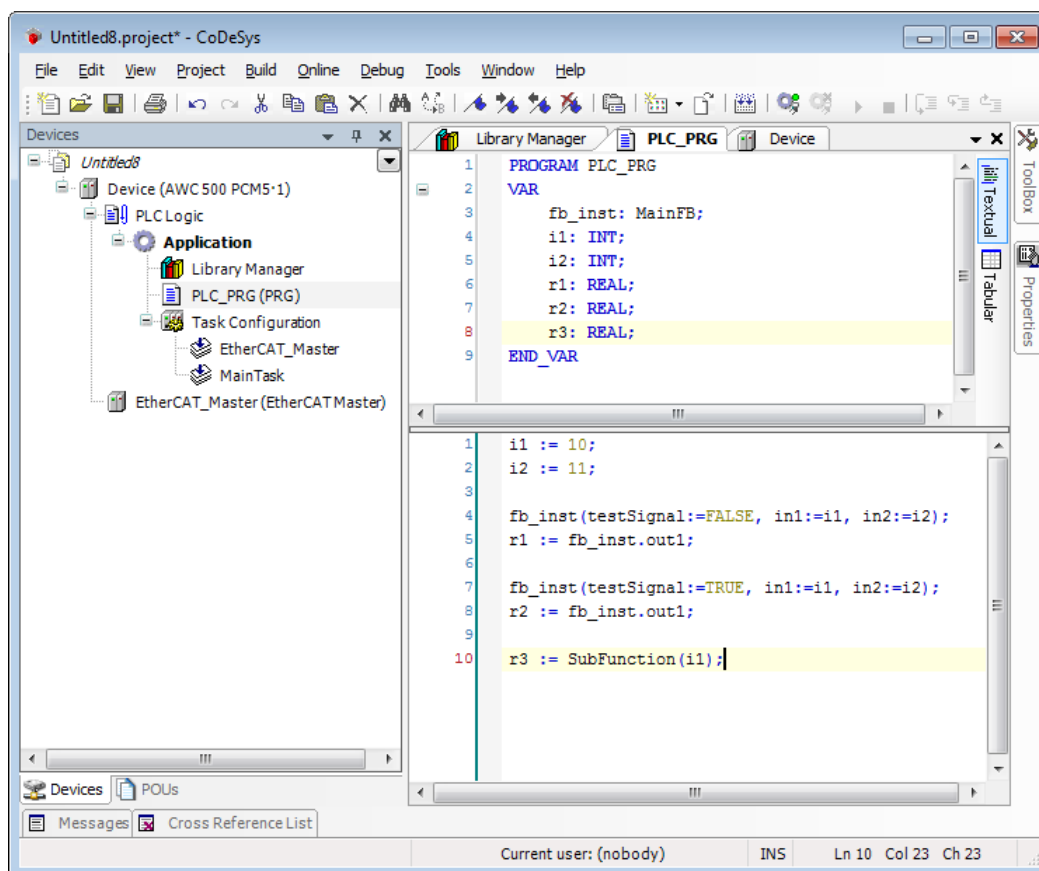


Figure 1.33

Download and run the application to see the result:

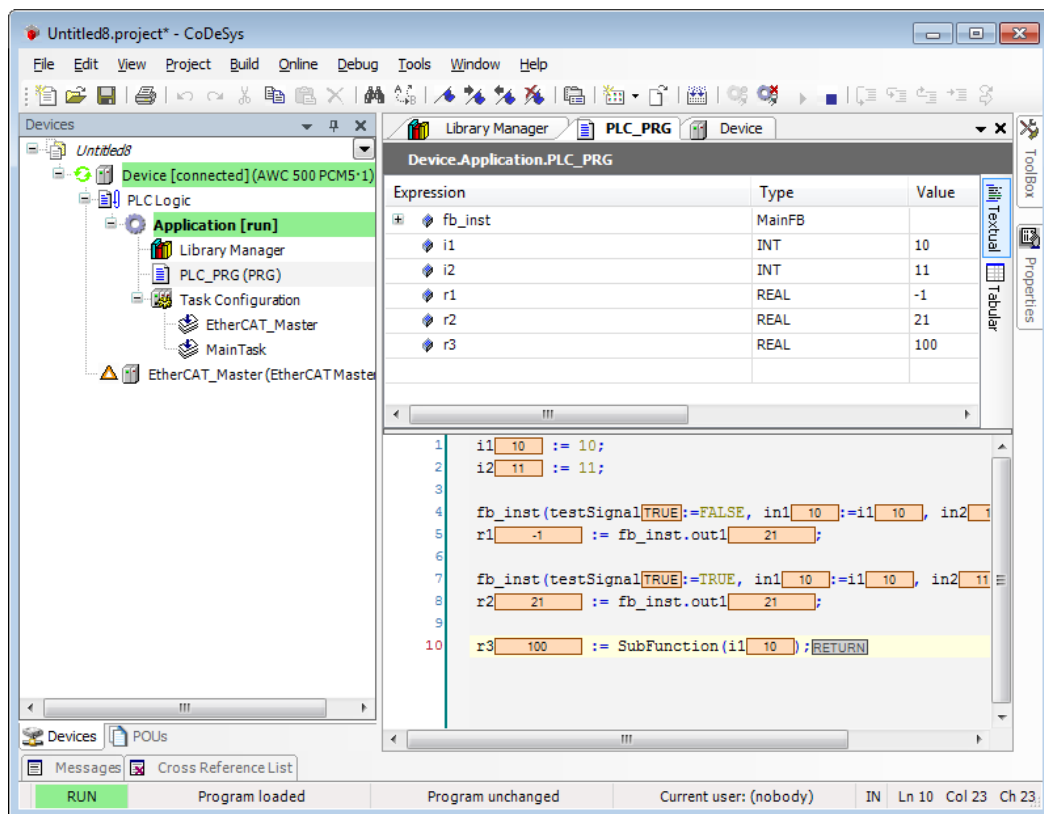


Figure 1.34

DEIF A/S reserves the right to change any of the above.