

Help

DCS800  
DC Drives

**CoDeSys Exercise  
Function Blocks &  
Libraries  
G562e\_a\_b Part 8**

eLearning

**Note:**  
**This module is an exercise without a speaker!**

**ABB**

© Copyright 11/08/2021 ABB. All rights reserved.  
CODESYS\_08F0101 page 1



Welcome to the CoDeSys Function Blocks and Libraries training module for the DCS800, ABB DC Drives. If you need help navigating this module, please click the Help button in the top right-hand corner. To view the presenter notes as text, please click the Notes button in the bottom right corner.

## Objectives

**After completing this module, you will be able to**

- Create new function blocks
- Implement function blocks into libraries



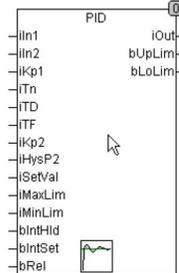
After completing of this module, you will be able to

- Create new function blocks and
- Implement function blocks into libraries

Help

## Create own function blocks

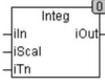
### Overview about created blocks



PID-Controller

Inputs: -In1, -In2, -Kp1, -Tn, -ITD, -ITF, -Kp2, -IHysP2, -iSetVal, -iMaxLim, -iMinLim, -bIntHld, -bIntSet, -bRel

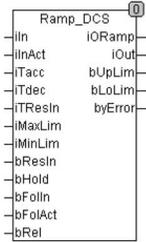
Outputs: iOut, bUpLim, bLoLim



Integrator

Inputs: -In, -iScal, -ITn

Output: iOut



Ramp-generator

Inputs: -In, -InAct, -ITacc, -ITdec, -ITResIn, -iMaxLim, -iMinLim, -bResIn, -bHold, -bFolln, -bFolAct, -bRel

Outputs: iORamp, iOut, bUpLim, bLoLim, byError

© Copyright 11/6/2021 ABB. All rights reserved.  
CODESYS\_Development page 3

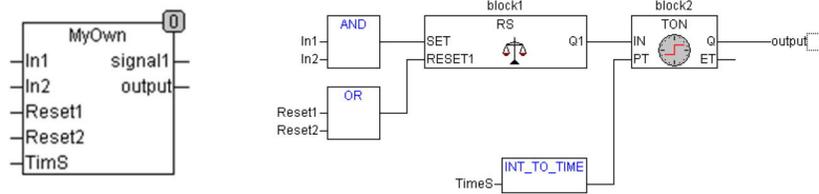


There can be an idea to put several functions into function blocks to hide know how. In CoDeSys it is possible to create own function blocks and put them into libraries. Further the user can define pictures for this blocks, like PID-controller. This function blocks are integrated in a library and can protected with a password. In this case the program code isn't visible for other users.

## Exercise 7: Create own function blocks

- With CoDeSys it is possible to create function blocks and insert them into libraries
- The function blocks can be protected by password
- Inputs and outputs can be selected and their data types can be chosen

### Example:



© Copyright 11/6/2021 ABB. All rights reserved.  
CODESYS\_Development page 4

**ABB**

Let's do a little exercise. In the next exercise we build a new function block called "MyOwn". This block includes function blocks and operators from the CoDeSys standard library. The difference between programs and function blocks are that the inputs and outputs must be defined.

## Create own function blocks

- Create a new project
- Open a *PLC\_PRG* in *CFC* language

1. Create a new Object  
→ Right mouse click on *POU*
2. Select  
→ *Add Object*
3. Choose  
→ *Function Block*
4. Language  
→ *FBD*
5. Fill out *Name of the new POU* press *OK*

The screenshot shows two windows. The top window is a context menu for 'POUs' with 'Add Object...' selected. The bottom window is the 'New POU' dialog box with 'Name of the new POU:' set to 'MyOwn', 'Type of POU' set to 'Function Block', and 'Language of the POU' set to 'FBD'.



© Copyright 11/6/2021 ABB. All rights reserved.  
CODESYS\_Development page 5

Next steps to build the new project are to add a new object and select the “Program Organization Unit” “Function Block”. In this exercise we use programming language “Function Block Diagram”.

## Build the program

- Now insert the blocks
- Open a new Network
  - Right mouse click on *Network 1*, select *Network (after)*
- Insert the other blocks in Network 2 and connect them

0001	FUNCTION_BLOCK MyOwn
0002	VAR_INPUT
0003	END_VAR
0004	VAR_OUTPUT
0005	END_VAR
0006	VAR
0007	block1: RS;
0008	END_VAR

Cut	Ctrl+X
Copy	Ctrl+C
Paste	Ctrl+V
Delete	Del
Network (before)	
Network (after)	Ctrl+T
Input	
Ctrl+H	

0001

AND

OR

block1

RS

SET

RESET

Q1

0002

INT\_TO\_TIME

block2

TON

IN

PT

Q

ET

© Copyright 11/6/2021 ABB. All rights reserved. CODESYS\_Development page 6

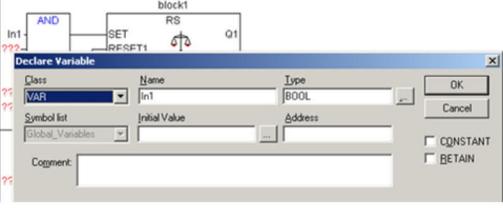
ABB

When the new network is designed, we have to put in the several function blocks. If Network 1 is ready, we add a new network after the first one and put in the other blocks. It will be avoided to split the whole program in several networks. So, it is easier to document the networks and get an overview about the whole program.

## Help

# Input variables

- Name the variables (e.g. *In1*)
- Click *Enter* (↵) and the *Declare Variable* Window is opened
- Declare the variable  
*Class: VAR\_INPUT* (FB input)  
*Name: In1*  
*Type: BOOL*
- When the settings are correct, click *OK*
- Do the same with all the other inputs



Class	Name	Type
VAR_INPUT	In1	BOOL
VAR_INPUT	In2	BOOL
VAR_INPUT	Reset1	BOOL
VAR_INPUT	Reset2	BOOL
VAR_INPUT	TimeS	Int



© Copyright 11/05/2021 ABB. All rights reserved.  
 CODESYS\_Development page 7

Next step is to declare all variables. This point is important because it must be defined which variables are inputs, outputs or internal variables. For our example we declare the variables like in the table.

## Output variables

- To declare outputs, mark the output and create an assign
- At the output appear question marks

The image displays two screenshots of the ABB ladder logic editor. The top screenshot shows a function block 'block1' with an output 'Q1'. An orange arrow points from the 'Assign' button in the top right corner of the editor to the 'Q1' output. The bottom screenshot shows the same function block with a question mark '?' next to the output 'Q1', indicating that the output has been declared but not yet assigned.

© Copyright 11/6/2021 ABB. All rights reserved.  
CODESYS\_Development page 8

**ABB**

In programming language “Function Block Diagram” the first output at a function block has no connecting point. So we have to add a “Assign” and question marks are shown on the output. Now a variable for the output can be defined.

Help

## Output variables

- Declare all outputs like in the table:

Class	Name	Type
VAR_OUTPUT	output	BOOL
VAR	signal1	BOOL

- Finish the program like in picture on the right!

```

0001FUNCTION_BLOCK MyOwn
0002VAR_INPUT
0003  In1: BOOL;
0004  In2: BOOL;
0005  Reset1: BOOL;
0006  Reset2: BOOL;
0007  TimeS: INT;
0008END_VAR
0009VAR_OUTPUT
0010  output: BOOL;
0011END_VAR
0012VAR
0013  block2: TON;
0014  block1: RS;
0015  signal1: BOOL;
0016END_VAR

```

© Copyright 11/6/2021 ABB. All rights reserved.  
 CODESYS\_Development page 9

We have 1 output of the new function block. Now this variable must be declared as output variable. The picture on the right side shows the several variables for the function block. We have input variables, output variables and internal variables.

Help

## Call the own function block

- Change to *PLC\_PRG*
- Insert the following three blocks:

The diagram illustrates a ladder logic network with three main components:

- Block 1 (in Digin):** A system block with eight digital input terminals labeled bDI1 through bDI8.
- Block 2 (block MyOwn):** A custom function block with five inputs: In1, In2, Reset1, Reset2, and TimeS. A constant value of 10000 is connected to the TimeS input.
- Block 3 (out DigOut):** A system block with two outputs: bDOut and wChannel. A constant value of 1 is connected to the wChannel output.

Connections: The outputs of the 'in Digin' block are connected to the In1, In2, Reset1, and Reset2 inputs of the 'MyOwn' block. The output of the 'MyOwn' block is connected to the bDOut output of the 'out DigOut' block.

- The box in the middle is the own function block
- The other blocks are system blocks for DCS800

© Copyright 11/6/2021 ABB. All rights reserved.  
CODESYS\_Development page 10

**ABB**

When the new function block called “My Own” is ready we can work with them. So, we change to the POU “PLC\_PRG” and build an application like in the picture. Now the project is ready, and we can test it.

Help

## Test the program

- Build your project (F11)
- Configure the communication to the DCS800
- Set the task configuration
- Login and test your program

**Correct function:**

If **DI1** AND **DI2** are **TRUE**, then after 10 seconds the output is **TRUE**  
 With **DI3** OR **DI4** you can Reset the output (**DO1 = FALSE**)

**ABB**

© Copyright 11/6/2021 ABB. All rights reserved.  
 CODESYS\_Development page 11

Build the project and configure the communication channel to DCS800. Set the task configuration and login with the program.

The correct function of this application is a delay of 10 seconds if digital input 1 and 2 are set.

## Exercise 8: Libraries

- The own function blocks can only used in the program in which they were written
- To use them in other programs, they must be imported into libraries
- It is possible to protect libraries with passwords
- Libraries use the extension \*.lib

The next exercise deal with library handling. Own function blocks can only used in the program in which they are written. To use them in other programs, they must be imported to libraries. It is possible to protect libraries with passwords.

## Export function blocks

- The own function blocks must be exported if they should be available in different programs
- Export  
→ *Project* → *Export...*
- Choose the function block (FB) to be exported and click *OK*

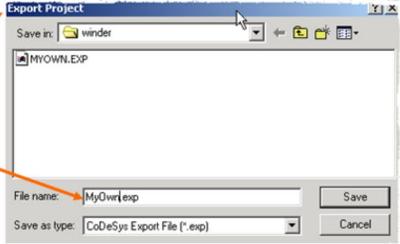
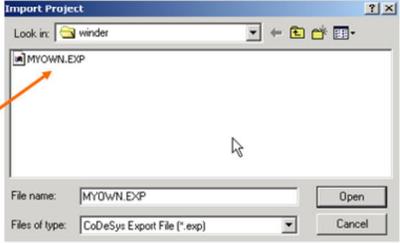
© Copyright 11/6/2021 ABB. All rights reserved.  
CODESYS\_Development page 13

There can be a need to use this function block in other projects. This is possible when the function block is exported to the other project.

## Help

# Export / import function blocks

- The window *Export Project* is opened
- Choose a *File name*
- Click *Save*
- Open another program
- Import the function block  
→ *Project* → *Import...*
- Choose the file to import
- Click *Open*
- The function block is imported

© Copyright 11/6/2021 ABB. All rights reserved. CODESYS\_Development page 14

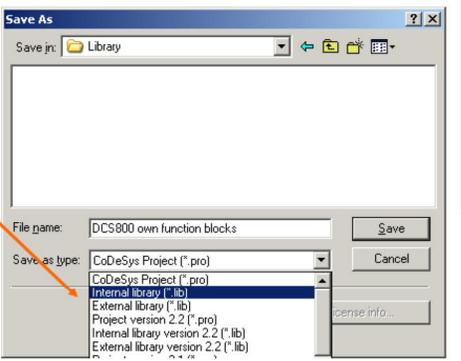


To export function blocks, define a name for the export file, please. Import of function blocks which are saved into other projects is very easy.

Open another program and click import to add the function block.

## Create libraries

- Create a new library  
→ *File* → *Save as*
- Select library type  
→ *Internal library*
- Chose a *File name* for the library
- Save the library



**Note:**

Internal library: libraries which are written with the five IEC languages

External library: libraries which used other programming languages (e.g. C)

© Copyright 11/6/2021 ABB. All rights reserved.  
CODESYS\_Development page 15



If a new library should be created click to menu “File” the item “Save As” and select “Internal library”. Now all function blocks which are included in the project, will be saved in a new library. It isn’t possible to save programs in a library structure. Only function blocks are inside a library.

## Help

# Insert libraries

- Select  
→ tab *Resources* → *Library Manager*
- Available libraries are shown
- Insert a new library  
→ *Insert* → *Additional Library...*
- Choose the directory in which the new library is located
- Mark the library and click *Open*

© Copyright 11/6/2021 ABB. All rights reserved.  
CODESYS\_Development page 16



The next question is how libraries can be embedded to projects. For this case you can find the “Library Manager” in CoDeSys. Insert libraries with “Library Manager” when you click to “Additional Library”.

The library manager shows you all available libraries like the “DCS800 Library” or the “Standard Library”.

Help

## Insert libraries

- Check if the new library is installed

© Copyright 11/6/2021 ABB. All rights reserved.  
CODESYS\_06/2021 page 17

**ABB**

If the new library is correctly installed, you can find the library name in the directory and see the available blocks.

Help

## Summary

### Key points of this module

- Create new function blocks
- Implement function blocks into libraries

© Copyright 11/6/2021 ABB. All rights reserved.  
CODESYS\_Development page 18



Key points of this module are to create new function blocks and implement them into libraries.

## Additional information

- Links to related information
  - [3S-software.com](http://3S-software.com)
  - DC-Drive-News (Intranet)
- Additional references
  - Application Manual (3ADW 000 199)
  - Firmware Manual (3ADW 000 193)
  - Hardware Manual (3ADW 000 194)
  - Training Material



## Glossary

- **CoDeSys**  
Controller Development System (software tool)
- **Memory Card**  
Flash memory
- **DriveWindow Light**  
Software Tool for commissioning and maintenance using AC/DC
- **Target**  
Interface between Drive and CoDeSys tool
- **Control Builder**  
Whole system with software and hardware
- **PLC\_PRG**  
Main program which is used in all applications
- **POU**  
Program Organization Unit
- **Library**  
It includes function blocks which are given or designed by other users





Power and productivity  
for a better world™

Thank you for your attention. You may now go ahead and move on to the next unit.