




[Help](#)


DCS800
DC Drives



**CoDeSys
Application
development
G562e_a_b Part 10**

 **eLearning**





© Copyright 11/8/2021 ABB. All rights reserved.
CODESYS_10R0101

Welcome to the CoDeSys 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.

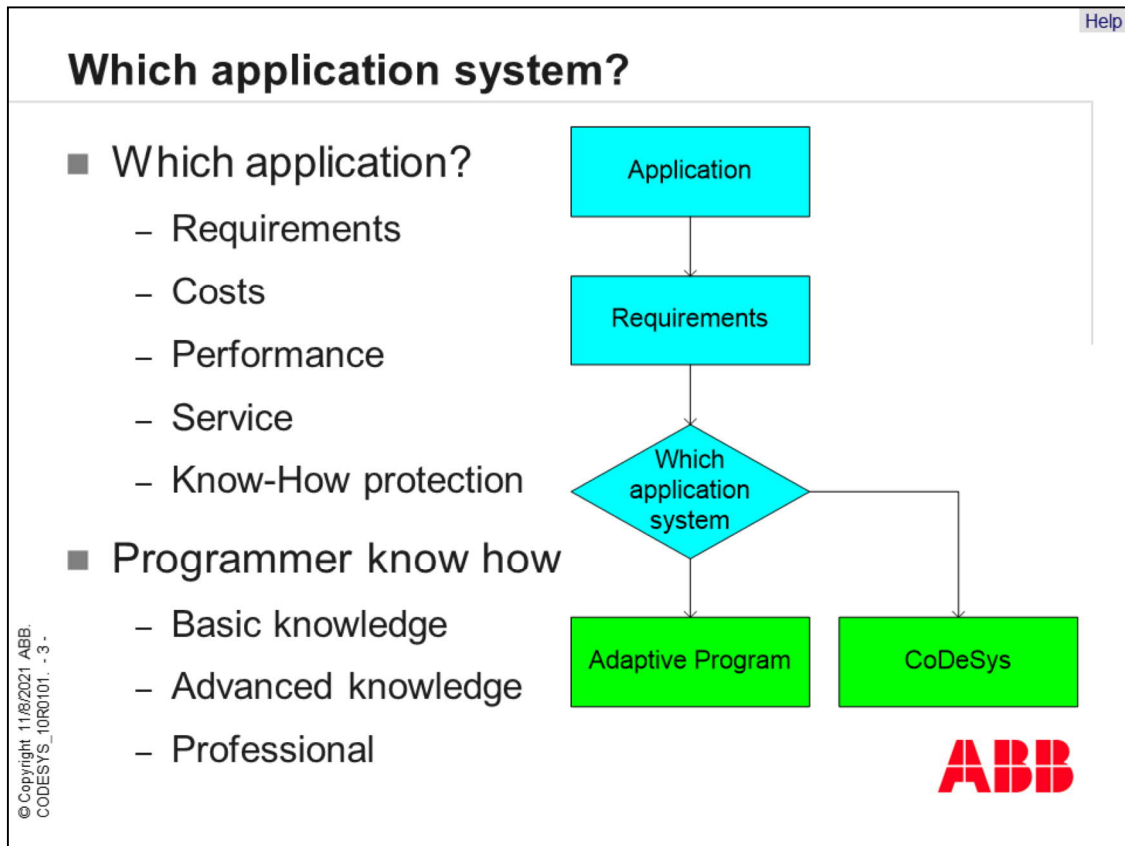
Objectives

After completing this module, you will know about:

- The way to create applications
- The possibilities to control the drive
- Visualizations
- Arithmetic functions
- Software timing
- Fault tracing
- Documentation

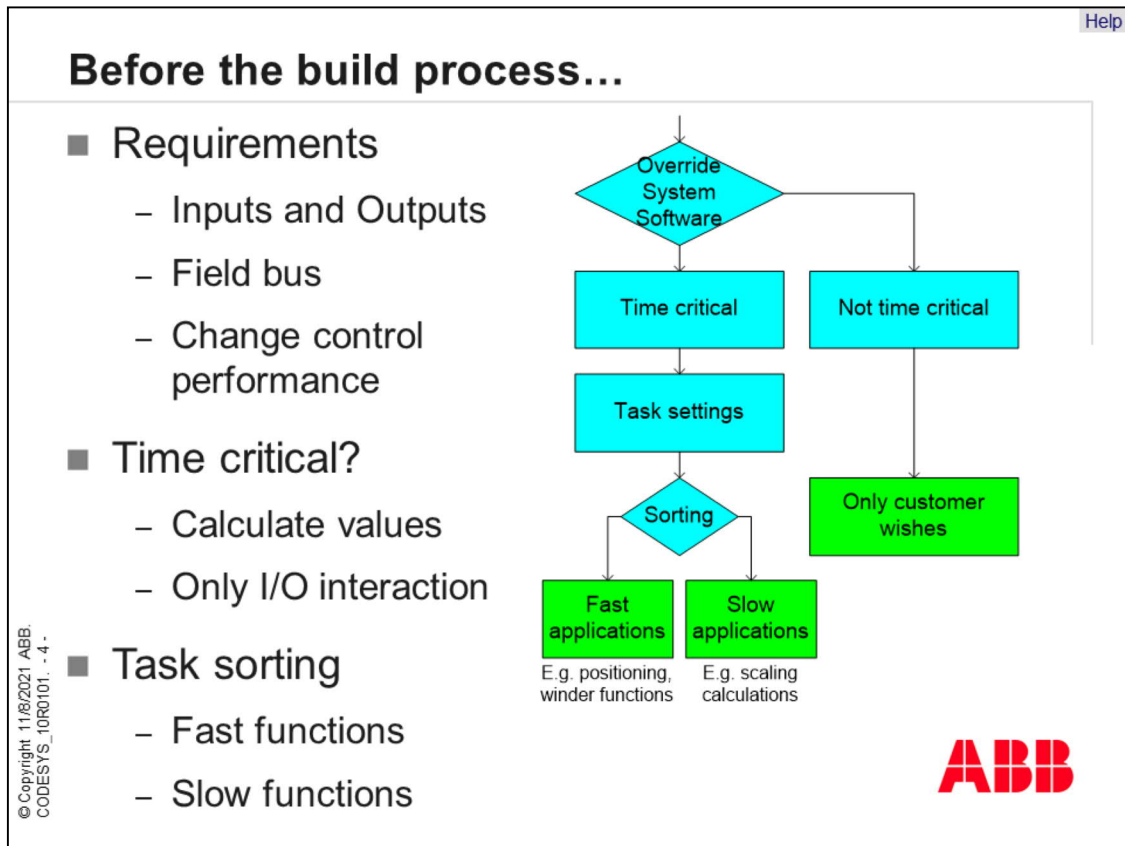


After completing this module, you will know how to create applications and the possibilities of controlling the drive. You will learn to create visualizations and how to work safely with arithmetic functions. Furthermore, this module discusses software timing, fault tracing and documentation of applications.



If an application program is needed, the first step is to choose which application system should be used. The next step is to check the requirements for the application. Another important step is to determine the costs of the tool, the engineering costs, the required performance and the approach in service cases. In certain cases, the know-how protection could be a point of interest for customers. You also have to take into consideration the knowledge of the programmer. The programmer must be able to solve application problems properly. So, it is important to know if the programmer has basic, advanced or professional knowledge of the software tools.

The last step is to decide between "Adaptive Program" and "CoDeSys".



Before starting to program, calculate what the requirements of the application are.

Important questions to be answered are:

- Which inputs and outputs are necessary?
- Is there a field bus connection required? and
- Should the control performance be changed?

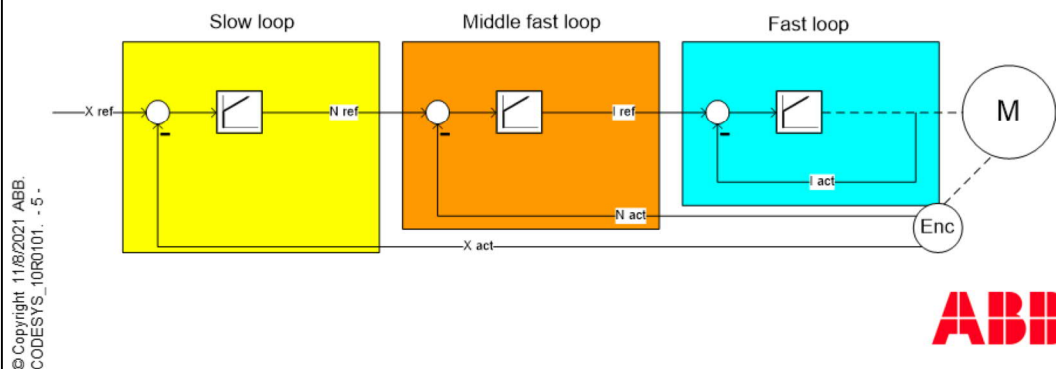
Another important issue is evaluating the application performance. If the application is a time critical application, then it is necessary to perform calculations in fast task cycles. If the application is not time critical, then only customer wishes must be implemented.

The processor load of the DCS800 converter is limited, so it is required to sort the programs in the correct task. It is recommended to sort fast and slow functions to split the processor load.

Example: Position control (time critical)

■ Cascade control

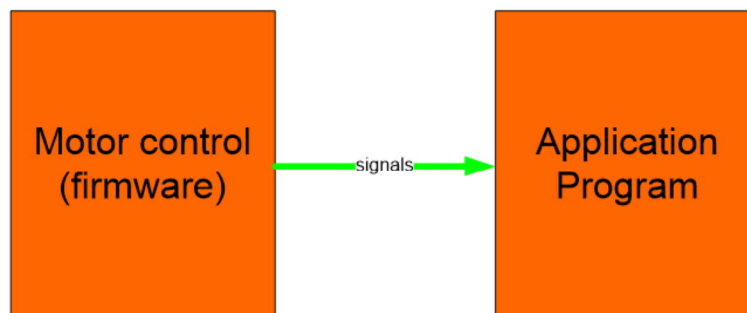
- Current controller (DCS800: 3,3 ms)
- Speed controller (DCS800: 3,3 ms)
- Position controller (DCS800: fastest 5 ms)



An example of a time critical application is a position control. To generate this application the current and speed controller are used from the standard firmware. The execution time of these control loops is 3,3 milliseconds. The position controller has to work in the fastest task cycle in CoDeSys with 5 milliseconds. Slower execution cycles would cause to problems for the entire control performance.

Example: Scaling for panel

- Convert units for the panel
 - Independent from the control structure
 - Not time critical



Another example is the calculation of internal values which should be displayed on the panel in the physical unit. This calculation is independent from the control structure of the drive and is not time critical. In most cases it is enough to calculate with 100 or 500 milliseconds.

Help

Possibilities to control the drive

- **Local I/O control**
 - Used for mixed control
 - Settings in group 10
 - MCW bit 11...15 (main control word)
 - ACW bit 12...15 (auxiliary control word)
- **Main Control Word**
 - Set P 7.01 (MCW)
 - Write with CoDeSys to parameter 7.01 (MCW)

```

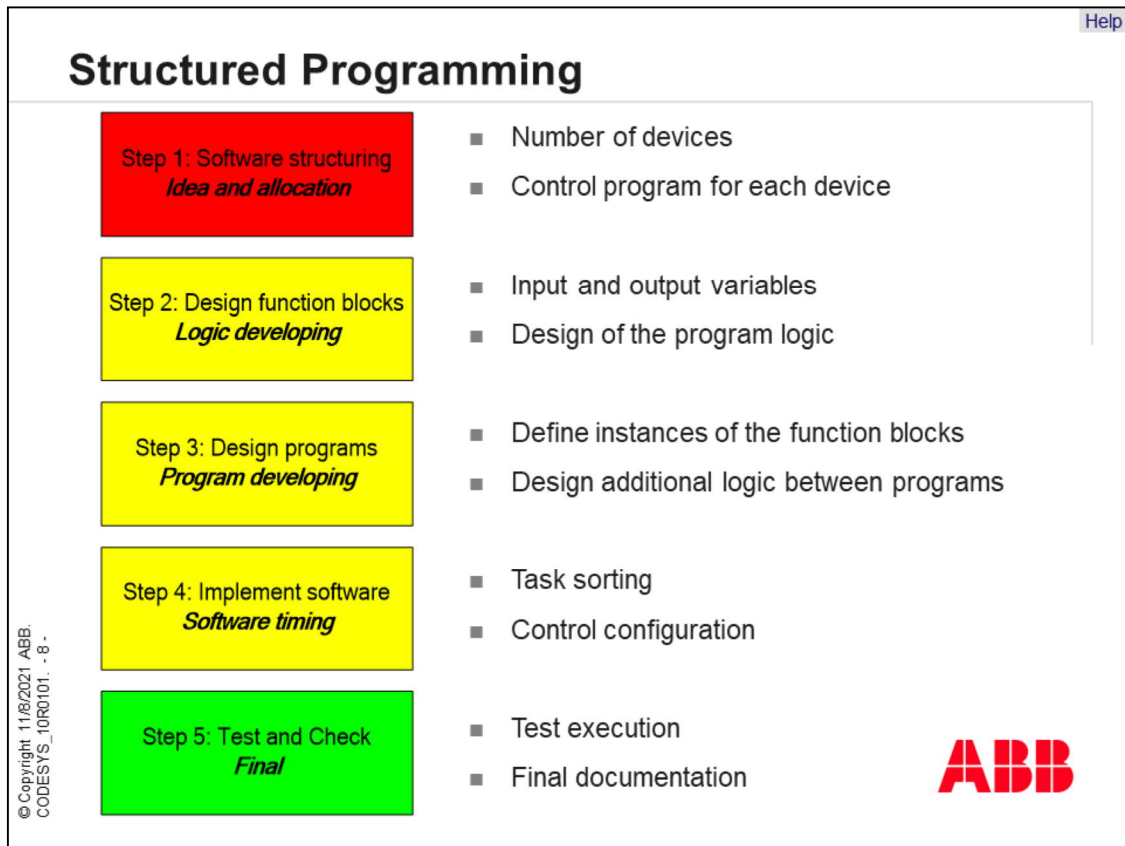
graph TD
    LIOLocal[Local I/O] --> DI[Digital Inputs]
    LIOLocal --> BMCW[Bits of MCW, ACW]
    MCWMain[Main Control Word] --> P701[P 7.01 (MCW)]
    DI --> DC[Drive Control]
    BMCW --> DC
    P701 --> DC
    
```

© Copyright 11/8/2021 ABB.
 CODESYS_10R0101. - 7 -

There are several possibilities of controlling a drive. The first control mode is the "Local I/O control" mode. In this mode the start commands are transferred via the terminal block as digital signals. If the commands should be connected with the application program, it is possible to use bit 11 to 15 in the "main control word" or bit 12 to 15 in the "auxiliary control word". This is a mixture between local and application control.

The other way is a "main control word" control strategy. This mode is often used when using a field bus connection to an overriding control system. In this control mode it is recommended to write directly to parameter 7.01 "main control word".

Both opportunities cause a connection between the system software and the application program.



Efficient working is only possible in a structured programming style. This example shows the way to create applications from the conception of the idea to the final test.

The first step is the idea and allocation of the problem.

Important questions needed to be answered are:

- How many devices are needed, or should each drive have its own application program?
- Which actors and sensors are needed?
- What is the functionality of the application program?

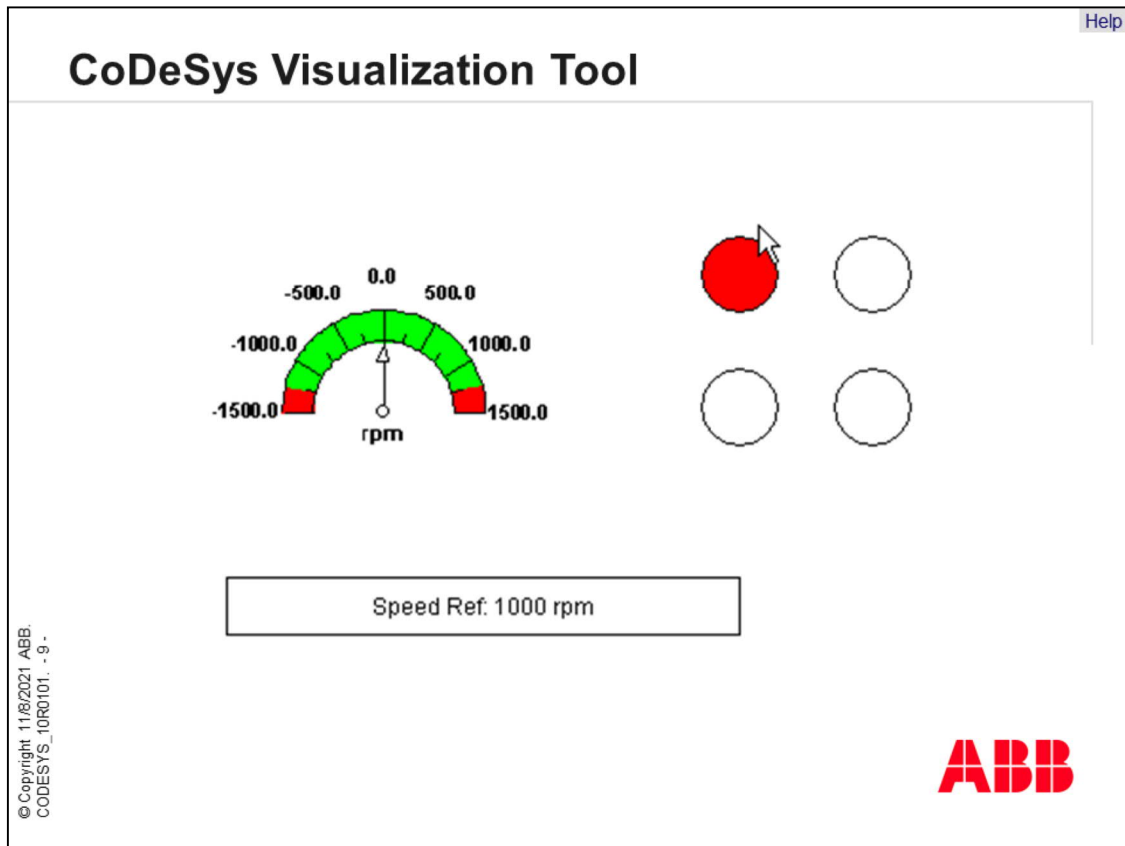
All these questions are important for solving the first step.

In the second step, the function blocks with the input and output variables should be designed. Inside these blocks, the application logic has to be designed.

In step three the whole application program between the function blocks must be developed.

Step four handles software implementation and timing. Each program part must be sorted to a task cycle to decrease the processor load.

The last step is to test the software. All software functions must be tested and checked for overflow or limitations. If the test is successful, a final documentation can be written and archived.

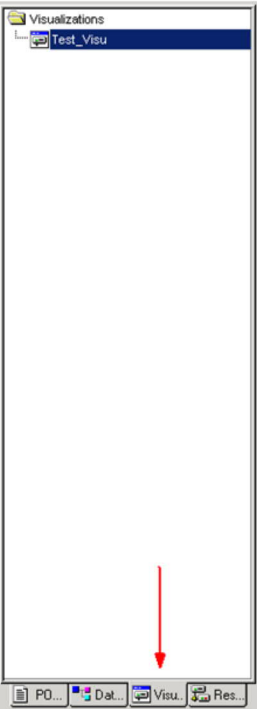


CoDeSys includes a visualization tool which can be used to connect global variables of the program. The visualization tool offers several buttons and fields to get status signals or to set reference values.

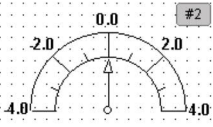


A meter is also available which can be used to display actual values of the program. The status of a Boolean signal is displayed by several color buttons.

Help

Visualization



- Visualization inside CoDeSys
- Connection with global variables
- Functionality
 - Switches
 - Diagrams
 - Displays

ABB


Inside CoDeSys a visualization can be implemented in the same manner as programs.

To open the visualization tool, click on "Visualization" in the "window-changer-bar". All global variables of the CoDeSys program can be connected to the visualization tool. The visualization offers the functionality of switches, diagrams or displays.

The visualization tool is usable as long as the CoDeSys program is online and connected with the drive. This tool could provide valuable help during the commissioning phase.


Help

Edit the visualization



- Above the editor window you can see the icon bar
- With this icons you can build the visualization
- The following tools are available:
 - Button
 - Rectangle, Ellipse
 - Table
 - Trend Line
 - Meter

© Copyright 11/8/2021 ABB.
 CODESYS_10R0101. - 11 -



If the visualization tool is opened, the control desk can be edited.

Above the editor window you will find the icon bar with all opportunities for editing the visualization window. Click on the icons and place them in the window. Some of the buttons can be resized. Buttons are available as a rectangle, a circle, an ellipse or a triangle. These figures can be used to set values or display values coming from the CoDeSys application.

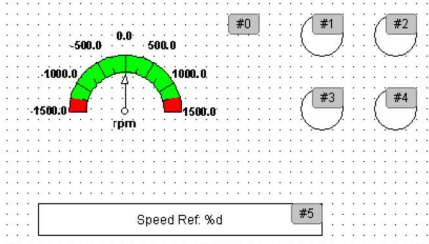
Another type of button is the "OK-button", which is used to select a state. This button is connected with a Boolean variable. Additional symbols are tables, trend lines or meters. A description of these symbols can be found in the CoDeSys help!


Help

Visualization example

- All tools are connected with global variables
- The global variables can also be actual values of the drive, if connected with drive signals
- Example:
 - The meter shows the actual speed of the drive
 - The rectangle is used to type in a speed reference
 - The ellipses show the drives status

© Copyright 11/8/2021 ABB
 CODESYS_10R0101 - 12 -





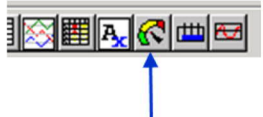
This is a visualization example for the DCS800. As a rule all tools in the visualization window are connected with global variables. When working with the DCS800 converter, the variables could also be actual values of the drive, if connected directly with drive signals via special interface function blocks.

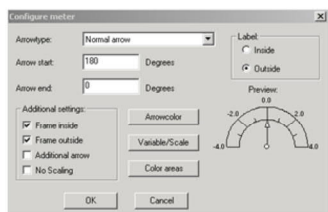
The visualization example shows a meter, 4 circles and a rectangle. The meter is connected with a global variable which is directly connected with the actual speed of the drive. The status of the drive is shown in the four ellipses. It is connected with the "main status word". With the rectangle it is possible to set a speed reference which is transmitted to the drive.

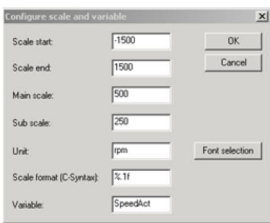
Help


Visualization example

- Configuration of the meter
 - Scaling
 - Color
 - Design
- Connection and format
 - Global variable
 - Unit









© Copyright 11/8/2021 ABB
 CODESYS_10R0101 - 13 -

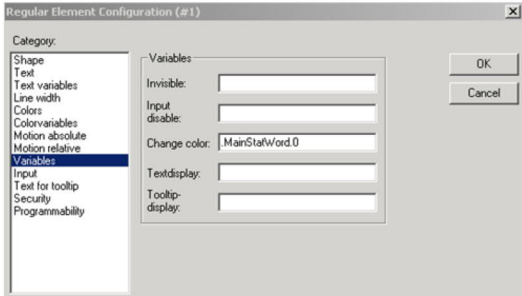
All devices used in the visualization window must be configured to show the correct values. The first device is the meter which should be used to display the actual speed of the drive. Important settings for the meter are scaling, color and design.

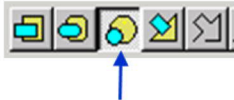
The next step is to connect the device with the global variable. In the configuration window the minimum and maximum value, as well as the unit have to be set. If all settings are set properly, the meter should show the actual speed of the drive with the correct scaling.

Help

Visualization example

- Circles can only show Boolean values
- Actual state will be shown with several colors
- Easy connection of "Word" variables
 - Select a bit





Note: Select a bit in a word

MainStatWord.0

↑

ABB

© Copyright 11/8/2021 ABB
 CODESYS_10R0101 - 14 -

The next device is the circle symbol. It can show Boolean values by changing colors.

To implement circle symbols, click on the circle icon and draw a circle in the visualization window. A double click on the drawing opens the configuration window. In this window all necessary settings can be set. An important setting is the selection of the variable in Boolean or Word format.

Note that a bit in a "WORD variable" can be selected, if a dot is set after the variable and the required bit number.

Help

Visualization example

- Input data must be configured
 - Presentation with the unit
 - Position inside the box
 - %d is the data source

Regular Element Configuration (#5)

Category:

- Shape
- Text**
- Text variables
- Line width
- Colors
- Colorvariables
- Motion absolute
- Motion relative
- Variables
- Input
- Text for tooltip
- Security
- Programmability

Text

Content: Speed Ref: %d rpm

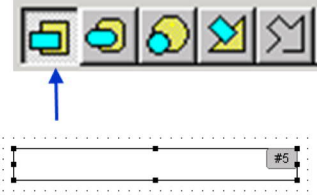
Horizontal

☐ Left ☒ Center ☐ Right

Vertical

☐ Top ☒ Center ☐ Bottom

Font... Standard-Font



© Copyright 11/8/2021 ABB
CODESYS_10R0101 - 15 -

ABB

A rectangle can also be used as an input field. Start with the implementation of the device by clicking the rectangle icon and draw the symbol in the visualization window. A double click opens the configuration window in which the settings for the input field can be set.

There are several configuration categories for the field with which the unit, the position of the text in the box and the data source can be selected. Further settings include text font and size.

[Help](#)

Visualization example

■ Keypad is the input source

- Minimum: -1500
- Maximum: 1500
- Dialog title: Speed Reference

© Copyright 11/8/2021 ABB
CODESYS_10R0101 - 16 -

- Motion absolute
- Motion relative
- Variables
- Input**
- Text for tooltip
- Security
- Programmability

In CoDeSys there is a keypad that can be used to enter input values. First select the menu "Input", then the input source "Keypad" and type in the minimum and maximum values allowed for the variable. The dialog title also has to be set. Later the title will be visible in the field.

When the application program is online, double clicking the rectangle field on the last page will open the keypad. There you can type in the values to be transmitted to the global variable in CoDeSys.

Help

Software timing

- System software
 - Drive control circuit
 - Digital I/O
 - Analog I/O
- Application software
 - Read parameters
 - Calculation
 - Write parameters

```
graph TD; C1[Calculation 1] --> C2[Calculation 2]; C2 --> C3[Calculation 3]; C3 --> C4[Calculation 4]; C4 --> C1;
```

© Copyright 11/8/2021 ABB.
CODESYS_10R0101. - 17 -

The term software timing simply means the execution order of calculations. The DCS800 distinguishes between calculations of the system software and the application software. The timing of the system software is fixed in the drive and cannot be changed. It handles the control circuit of the drive and the digital and analog inputs and outputs. The updating time of the parameters and hardware connections can be found in the firmware description.


The timing of the application software can be freely selected by the user. A list of predefined cycle times provides the option to sort each program part to an appropriate cycle time. This eases the sorting of programs!

Help

Calculation time

- Task cycle time
 - Cycle in which the task is triggered
- Execution time
 - From task start until the end of the program code
- It is essential:
 - **Execution time << task cycle time**
- Attention:
 - Execution time > task cycle time
 - Calculation isn't completed correctly!
 - After 3 stalled tasks → Fault message

© Copyright 11/8/2021 ABB
CODESYS_10R0101 - 18 -



In the “digital world” there are a lot of terms for which it is important to know and understand the difference. The first item is the "task cycle time". This time defines in which time interval a program part will be started with the execution. The term "task triggering" is often also used to describe this item. A task cycle of 20 milliseconds means that the program will be executed every 20 milliseconds.

Another interesting item is the "execution time". It defines the time which the processor needs to perform all the calculations and commands.

It is essential that the execution time is smaller than the cycle time of the program. Otherwise the calculation will not be completed correctly.

If the task continues to be blocked after three calls of a stalled task, then the DCS800 generates a fault message.

Help

See the actual status...

- CoDeSys provides an overview about the execution time

Task configuration

- Task100ms [DEBUG]
 - Inputs_Outputs()
 - Logique();
 - Supervision();
- Task5ms
 - Speed_Current_I

Task execution

Reset Scaling (µs/Pixel): 1

All used task cycles

Task100ms

Min: 487µs Last: 542µs Max: 4986µs Cycle: 0µs Jitter: 100µs

Task5ms

Min: 456µs Last: 923µs Max: 1063µs Cycle: 0µs Jitter: 5452µs

Actual values

ABB

© Copyright 11/8/2021 ABB
 CODESYS_10R0101 - 19 -

Information about the actual load of each task is provided in a window in the task configuration. A meter that shows the actual load is available in CoDeSys if the drive is online and shows if the execution time is okay or if an overload will occur.

Note: It is not recommended to overload a task cycle and there is no command in existence which stops the application program in such cases. This is the task of the programmer!

Help

Software timing

- 1st step
 - Read values which are needed for a calculation
- 2nd step
 - Calculate
- 3rd step
 - Write the calculated values to the output
- Back to the 1st step!

```
graph TD; A[Read values] --> B[Calculation]; B --> C[Write values]; C --> A;
```

© Copyright 11/6/2021 ABB
CODESYS_10R0101 - 20 -

For calculations with high accuracy, it is important to execute a program in the correct sequence. PLC's do this in the following order:

- The first step is to read and collect the values which are needed for a calculation
- The next step is to calculate the values from the same time level
- And finally, the results are written if the calculation is completed.

Why documentation?

- Applications have to be documented
 - Paper form
 - PC file
- Types of documents
 - Customer documentation
 - Developer documentation
- Reasons to document
 - Quality control
 - Easier to understand for others

Why is it important to document applications? This question is essential for good application developers because the quality of a complicated application is identifiable by the documentation.

Basically, all applications have to be documented. There is no difference in the documentation between small and big application programs. It is recommended to document in paper form and as a pc file.

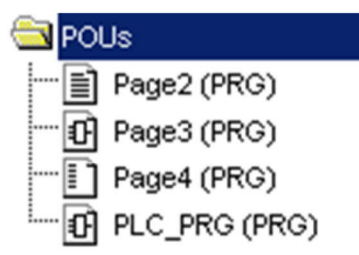
The types of documents are customer and developer documentation. The difference lies in the perception of the problem which has to be solved. A customer wants to have the settings in writing which must be performed, providing the customer with a correct functionality. The developer documentation is more detailed and explains the application functionality.

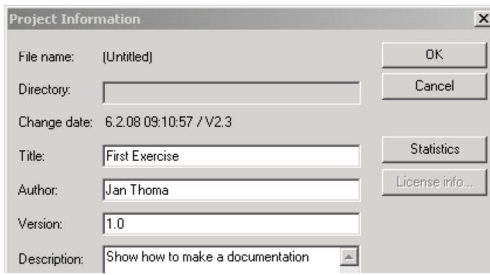
Reasons to document include quality control and making everything easier to understand for other people.

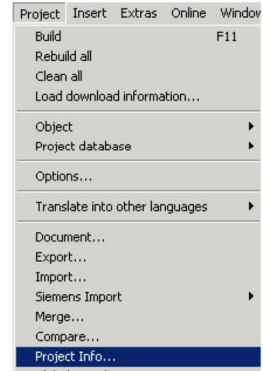
Help


Example: Documentation of a project

- Project with 4 POU's
- Click on *Project Info* in the *Project* menu
- Type in important project data







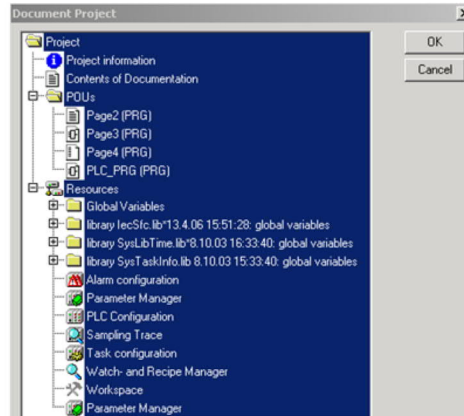
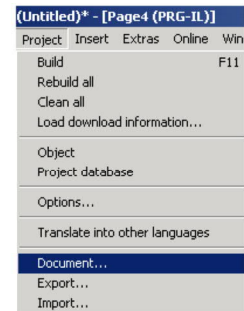


© Copyright 11/8/2021 ABB
 CODESYS_10R0101 - 22 -

The documentation of a project should be explained in a small example. The starting point is a project with 4 POU's. Click on "Project Info" in the menu "Project" to open the information window. Type in the important project data (this could be the title, author, version or a small description of the functionality). Two of these fields are displayed later in the firmware parameters.

Documentation form

- Click on *Document* in the main menu *Project*
- Select the data which has to be printed



© Copyright 11/8/2021 ABB
CODESYS_10R0101 - 23 -



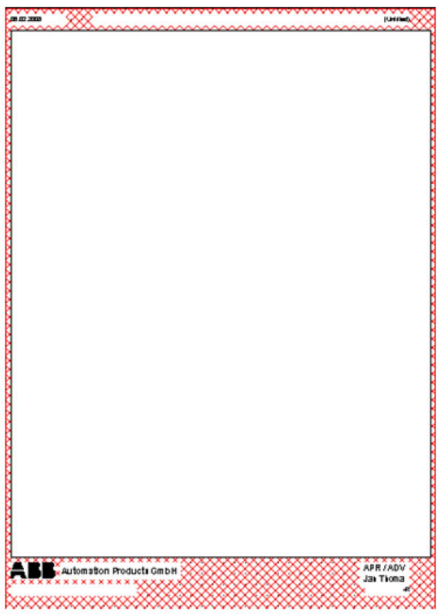
CoDeSys offers a documentation form which can be used to print the complete project. Select the documentation form by clicking "Document" in the menu "Project". The next step is to select the things which have to be printed.


[Help](#)

Create own layout

- Click on *Printer Setup* in the menu *File*
- Click on *Edit*
- The “layout editor” opens
- Set the fields and printing area
- Save these settings for all documents

© Copyright 11/6/2021 ABB
 CODESYS_10R0101 - 24 -





It is often helpful to create a customized layout which includes, for example, the company logo or to consider using a special paper format.

Configure the layout of the form by clicking "Printer Setup" in the menu "File". Click on the button "Edit" to open the layout editor. All settings can be performed in the layout editor. Set the "printing area" fields to mark the useable area for the printer. Note that in the editor window of CoDeSys the selected "printing area" will be displayed in all subsequent projects.

The last step is to save these settings for all documents which are created after the first configuration.

Help

Printer setup

- Select *Printer Setup* in *File* menu
- Click on button *Printer Setup*
- Set your “normal” printer and page settings

Printer and properties

Page orientation

Paper size

© Copyright 11/8/2021 ABB
 CODESYS_10R0101 - 25 -

Now the configuration of the documentation is ready, but CoDeSys offers a second setting which allows the user to change between several printer formats. So, the layout is independent of the paper format.

Select "Printer Setup" in the menu "File" to open the setup window. Click on the "Printer Setup" button to choose the printer and the paper format.

Attention: After specifying these settings, please check your printing area in the editor window. Otherwise, there can be function blocks outside the printing area!

Summary

Key points of this module

- Creating applications
- Options of controlling the drive
- Visualizations
- Arithmetic functions
- Software timing
- Fault tracing
- Documentation

In this module you should have learned about the following:

- Creating applications,
- Options of controlling the drive,
- Visualizations,
- Arithmetic functions,
- Software timing,
- Fault tracing, and
- Documentation.

Additional information

■ Links to related information

- 3S-software.com
- DC-Drive-News (Intranet)

■ Additional references

- Application Manual (3ADW 000199)
- Firmware Manual (3ADW 000193)
- Hardware Manual (3ADW 000194)
- 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**
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.