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.
Attention, this exercise is much more complex than the other parts. It’s a complete project which demands that the user have experience with the commissioning of DCS800 and the local control of a drive.

It is necessary that the participant knows the approach to implement an application and configuration.
After completing this module, you will be able to create a complete project with CoDeSys.
The exercise includes several jobs which must be solved to get a whole project. So we design a new function block and integrate them into a program. Also reading and writing parameters from the standard firmware is a part of this project. If the program is ready and implemented it is necessary to test it.
The project is a start logic for a motor. The motor logic controls the speed reference and the actual speed of the motor. Idea of this logic is that the release of the drive is only possible if the speed reference is lower than minimum speed (new parameter). That means the drive will be controlled over speed reference potentiometer.
This is the overview about the whole project.

In step 1 the new program must be created. Step 2 deal with the design of the new function block called “MotorLogic”. After this the other function blocks must be implemented and connected. Step 4 and 5 gives you information about new user events and parameters.

Commissioning step by step is the next step in our program. Here it is important to find faults and errors in the software. Last step is the test and documentation of the project.
Start a new project and select the DCS800 target. For this program we use the programming language “Function Block Diagram”. This “POU” is used to build the program.

But we need a second “POU” for the new function block called “Motor Logic”.

Step 1: Add a POU

- Start a new program
- Select DCS800 target
- Choose type Program and language FBD
- POU name: PLC_PRG
- Click with right mouse button to POU
- Select Add Object
- A new window opens
Select the type “Function Block” for this “POU” and programming language “Function Block Diagram”. All “POUs” of the project are shown in the root directory on the left side. In CoDeSys it is possible to define many “POUs” which can be programs, function blocks or functions.
Now we need a third “POU” which is used to work with user events. It’s from type program which work with programming language “Function Block Diagram”. So, the project is split into 2 parts with a defined function block.
If you will define new function blocks it is an important question which inputs and outputs are necessary. You can see after the variables the type of the variable. This is also an important information for the design of the block.

Define all input and output variables like in the picture below. For good documentation, add a comment, please.
If you type in variables, you get the declaration window in which the class and the type has to be defined. The class “VAR” accord to local variables which are only available in the function block. Class “VAR INPUT and OUTPUT” defines connection variables on the function block.
Next step is to create the function block “Motor Logic”. Insert all function blocks in network 1 which are shown on the picture. Start with the last function block ease the approach to build the program.
The new function block controls the motor. If the actual speed “InAct” of the motor is lower than a defined minimum value. This function is connected with an AND-element. If the speed reference “InRef” is greater than minimum value. The motor start if the condition is fulfilled. That means the motor will be controlled via analog input. If the motor is standing still a special delayed signal will be generated. The other inputs are features.
After development of a new block, it can be used in the main program PLC_PRG. The name *MotorLogic* is the name of this block and must be called in PLC_PRG. In PLC_PRG you can see the function block in the visualization with inputs and outputs.

After development of a new block, it can be used in the main program “PLC_PRG”. The name “Motor Logic” is the name of this block and must be called for using in main program “PLC_PRG”. If you use a graphical programming language the new function block is shown as box with inputs and outputs. The function of this block is hidden inside.
Next step is to build the main program “PLC_PRG”. Put in the function blocks like in the picture and define the variables. We need global variables for the variables speed reference, speed actual and “REL” because they should be connected later to another POU. The functionality of this three blocks are the directly reading of parameters.
Now the main program must be enlarged with the new function block “Motor Logic”. Connect the the variables like in the picture and write comments in the several networks. Network 5 shows a conversion between a boolean and a word type. With this operation it is possible to set a bit in a word with a boolean value. In this exercise we set bit 11 in main control word.
The picture on the top explained again the conversion between bits and words. Another way to do this is the possibility to use a function block from the special library which is available from the CD.

This approach is from a boolean value to word. But with CoDeSys it is possible to do this in the inverse direction. Take a word variable and write it to a boolean connection. Select the needed bit after the dot.

The picture on the bottom shows the connection of variable “REL” to function block “Motor Logic”. We need for this application only bit 7. So we have to set the notation “REL dot 7” to input “REL” of function block “Motor Logic”.

Step 3: Bit selection

- Select a Bit of a Word:
  - This is a typical conversion in CoDeSys
  - A Boolean value set a bit in a word (bit 11 in mcw)

- The other way:
  - REL is a word variable
  - Bit 7 of this word should be set input bRel of FB MotorLogic
Now we have to complete the main program with network 6. This network includes function block parameter set because we will write to parameter 7.01, the main control word of the drive.

The picture shows also the declaration window with the documentation. Correct your application that it looks like the picture. If you have done this the main program is ready and we can go to the next step.
Our next job is to define the POU “Events”. Here we will define the trigger signals for all events and the initialization.

First function block in network 1 is the event initialization. With this block the event texts are allocated to the events. In this exercise we save the event messages in parameters 60.06, 60.07 and 60.08.

Network 2 gives information about the trigger signals. Event 310 should be set if the actual speed of the drive is greater than the limit. The same procedure is in network 3. There event 610 will be triggered.
Last event is the notice message 810. Check the declaration window so that it looks like the picture.

Now the event program is ready.
Before we defined the events. Now we must set the message texts for this project. Message texts have a own data type in CoDeSys. So, we must change to item “Data types” and add new objects. Each object is a new data type.
We need a name for the new data types, and it will be avoided to choose a name which associate with the event. So, we use names like “Fault, Alarm and Notice”. In this new data types, we must set the event messages.

Delete the structure and write a message text into brackets. Note, only 12 characters are allowed! If you need more than 1 text, set a comma between the texts in the bracket. The first string in the bracket accord to the first event number.
Global variables are used to connect programs and parameters. So, we defined before parameters as global ones which are shown in the list. The last 3 rows shows the new data types of the events which are set 1 slide before. Fill in the global variables for this events so that they can be connected to parameters. This variables are needed later in the parameter manager.

Now the variable definitions are complete, and we can do the next step.
Change to the parameter manager in folder resources and insert a new list. Select variables and set the name “Application Parameters” in a short notation to the list. A new list is created!
There are several fields to declare parameters. Fill in the variables and numbers from the picture on the bottom. The field “Name” is the parameter name and “Var” the connection to the variable in CoDeSys. Range of the data is given by the minimum and maximum fields. All fields are parameters that means the tick in field “Read Only” isn’t set.
Add a second list named “Application Signals” and add lines to the list. Fill in the variables and numbers like in the picture. Set a tick in the fields “Read Only” because these are signals, which aren’t writeable.

Now the application is ready and we can go to the next step.
An important thing in CoDeSys is fault tracing. In bigger projects it is normal that they are any errors in program. CoDeSys shows the user all errors with the applied line or network. Build your program and look to the message window. If you see the message “0 errors and 0 warnings” you can be happy. Otherwise, you must find the errors in the program.
Find the errors when you click to the error line and the editor window jump to the applied line or network.

If there are many errors check the correct declaration and the several characters and types to find the errors.
Errors in the parameter list are dangerous, because incorrect parameters will not be generated. Check the maximum and minimum of the selected variables. Boolean variables get the character 0 and 1. In Enum variables it is necessary to select the first entry.
If the program is ready, we must configure the task configuration. In this application we need two different tasks. The main program PLC_PRG work with the 5 ms cycle. The event program needs the 100 ms task.
It is necessary that the drive is in “Local Mode”. Change parameter settings with the panel, DriveWindow and DriveWindow light.
Select your communication channel and download the program. Click “Run” and select “Create Boot Project” to save the program on the Memory Card. When the program is saved correctly you get the message that the program is saved and the user parameters are generated.
Now we have to test the program with the hardware. Close the main contactors with digital input 7. Digital input 8 must be active. An inactive digital input 8 disables the motor logic and close a bypass.

Turn the potentiometer until the motor is running. Increase the speed reference until you see the alarm message. If the motor speed increased more and more until a fault is activated. Check also the notice if the drive is standing still. Change the new parameters and play with the application.
For all applications it is necessary to document it for other users and for your own protection. You can find “Project Information” in menu Project. The field “Title” appears in parameter 4.03 and “Version” in parameter 4.12.
Key point of this module is learning to create a complete project in CoDeSys.
Additional information

- **Further information:**
  - Application Manual (3ADW000 199)
  - Firmware Manual (3ADW000 193)
  - Hardware Manual (3ADW000 194)

- **CoDeSys support:**
  - [www.3s-software.com](http://www.3s-software.com)

- **Drive support and libraries:**
  - ABB DC-Supportline
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
Thank you for your attention. You may now go ahead and move on to the next unit.