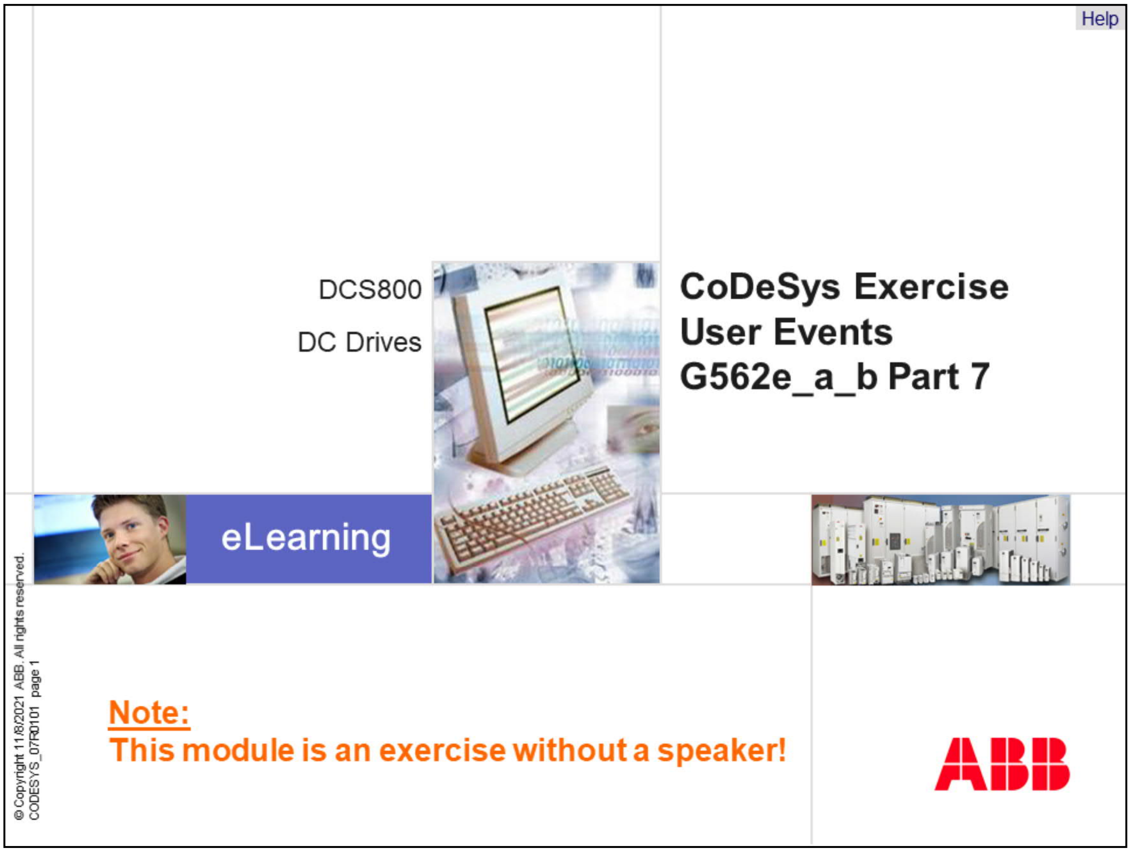


Help

DCS800

DC Drives

CoDeSys Exercise User Events G562e_a_b Part 7

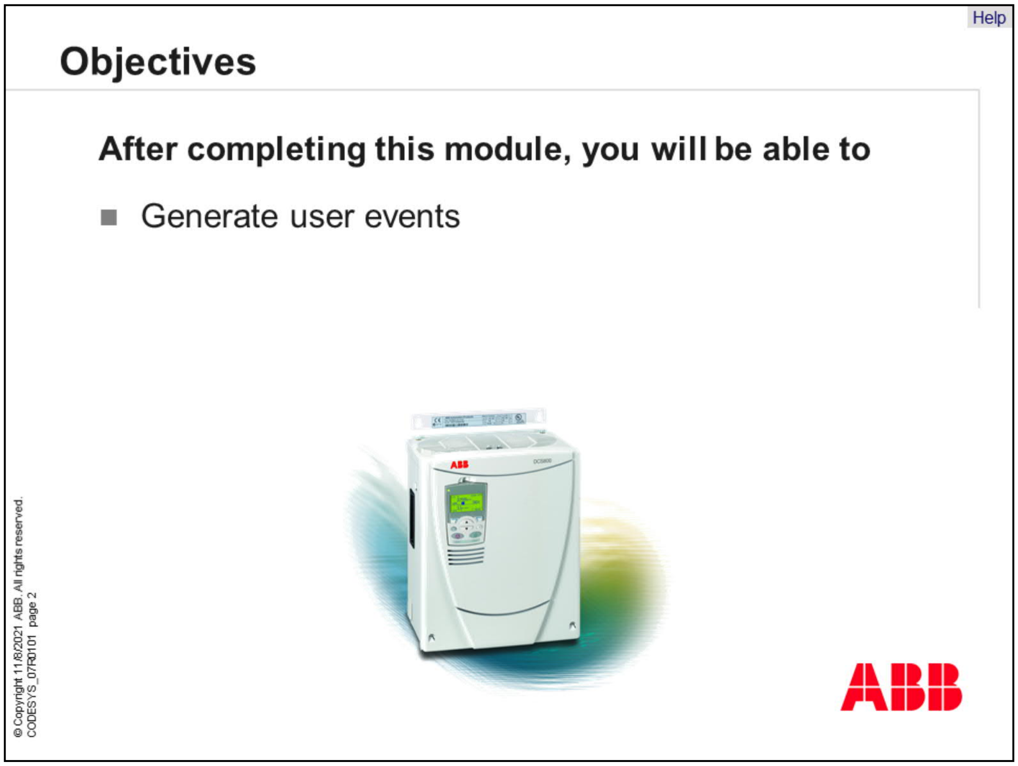
eLearning

Note:

This module is an exercise without a speaker!

Welcome to the [Module title] training module for the [Product name], ABB [Product family].

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.



Help

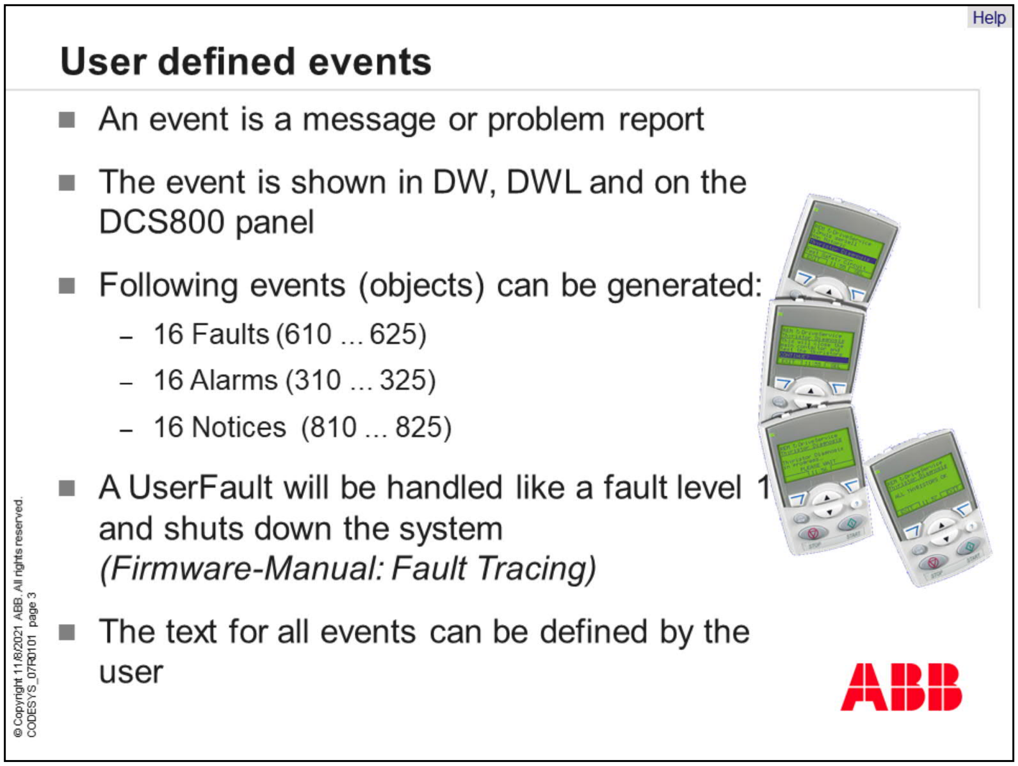
Objectives

After completing this module, you will be able to

- Generate user events

After completing of this module, you will be able

- To generate user events

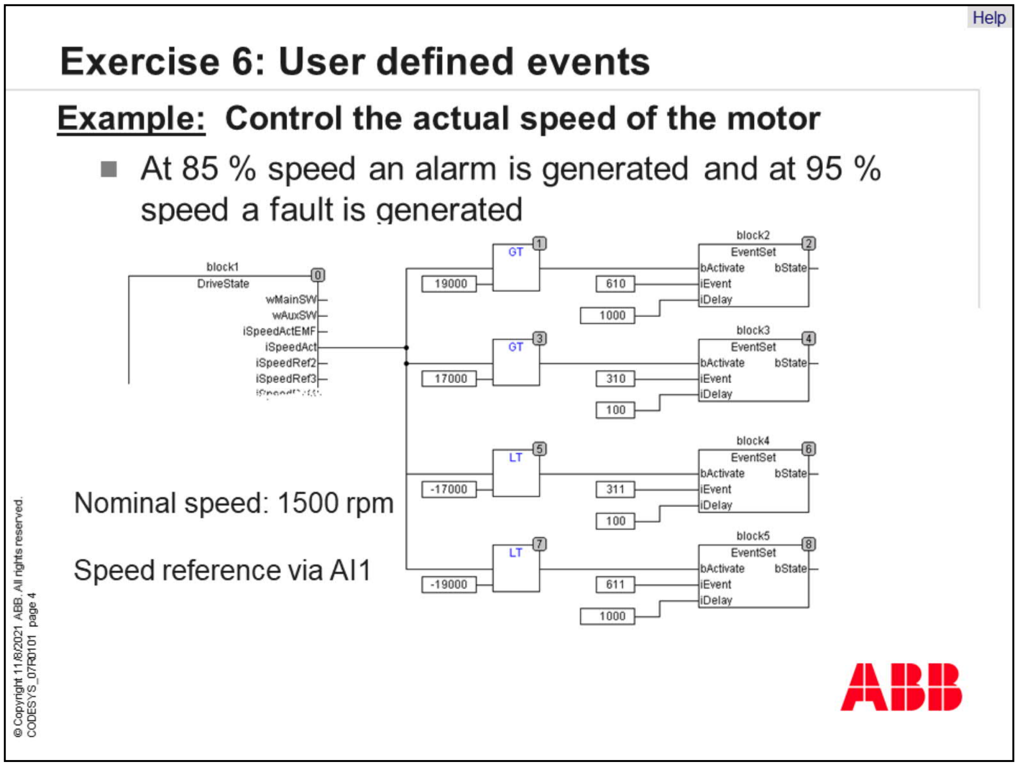


User defined events

- An event is a message or problem report
- The event is shown in DW, DWL and on the DCS800 panel
- Following events (objects) can be generated:
  - 16 Faults (610 … 625)
  - 16 Alarms (310 … 325)
  - 16 Notices (810 … 825)
- A UserFault will be handled like a fault level 1 and shuts down the system *(Firmware-Manual: Fault Tracing)*
- The text for all events can be defined by the user

An event is a message or a problem report. The event is shown in DriveWindow, DriveWindow light and on the DCS800 panel. There are 16 Faults, 16 Alarms and 16 Notices available. The definition of this events is happened in CoDeSys.

“User Faults” will be handled like fault level 1 and shuts down the drive.



Exercise 6: User defined events

Example: Control the actual speed of the motor

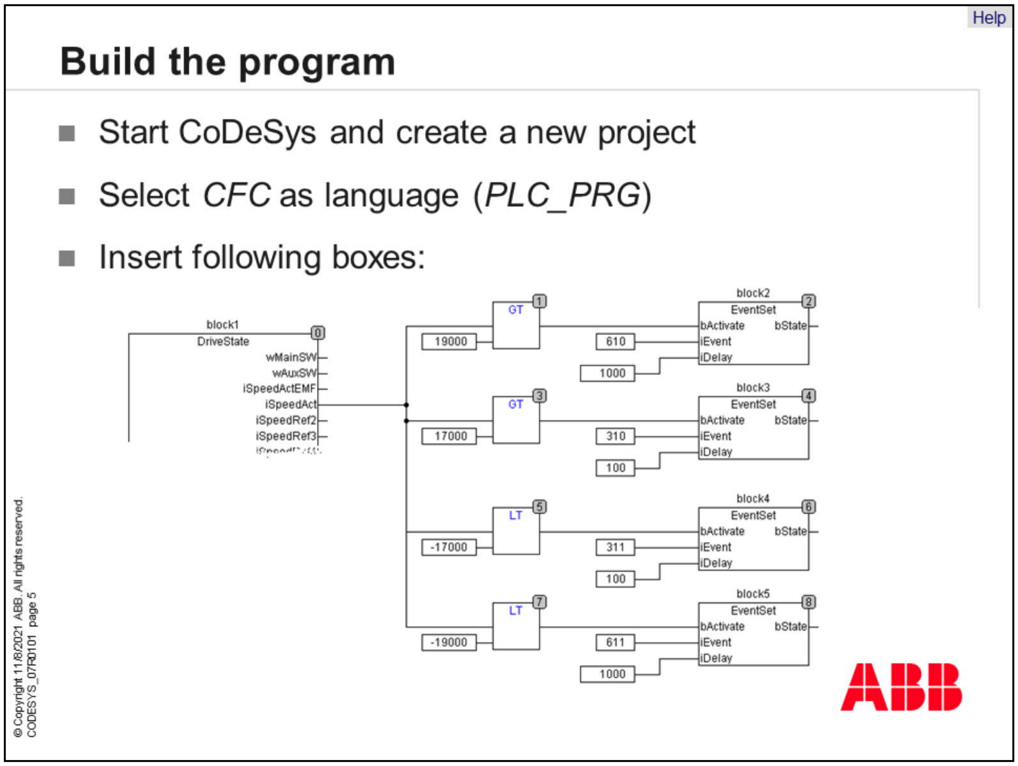
- At 85 % speed an alarm is generated and at 95 % speed a fault is generated

Nominal speed: 1500 rpm

Speed reference via AI1

Let’s do a little exercise. We would like to control the actual speed of a motor. At 85% of nominal motor speed there should be generated an alarm and with 95% a fault. The application should have the functionality for both motor directions.

In this application we work with internal values of speed measurement. So, we must scale the physical speed from 1500 rpm into internal scaling. The speed reference is given with analog input 1.



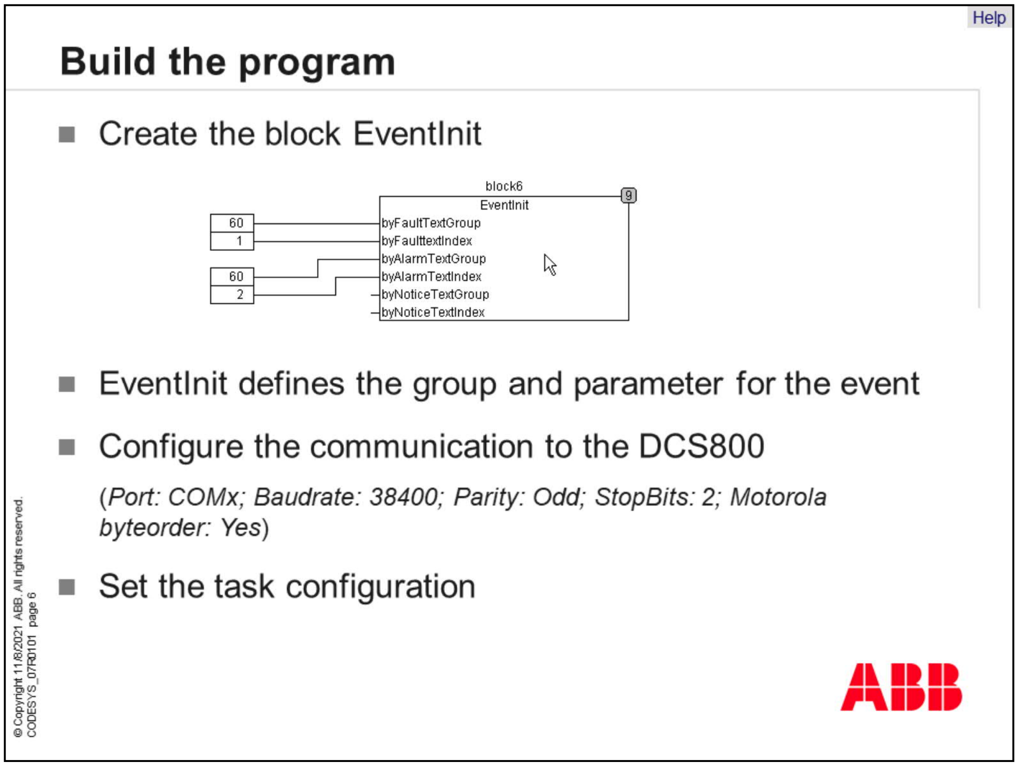
Help

Build the program

- Start CoDeSys and create a new project
- Select *CFC* as language (*PLC_PRG*)
- Insert following boxes:

First start CoDeSys and create a new project. We use programming language "CFC" and the program name "PLC_PRG".

If the settings are done correctly, insert the boxes like in the picture and connect the function blocks.


Help

Build the program

- Create the block EventInit

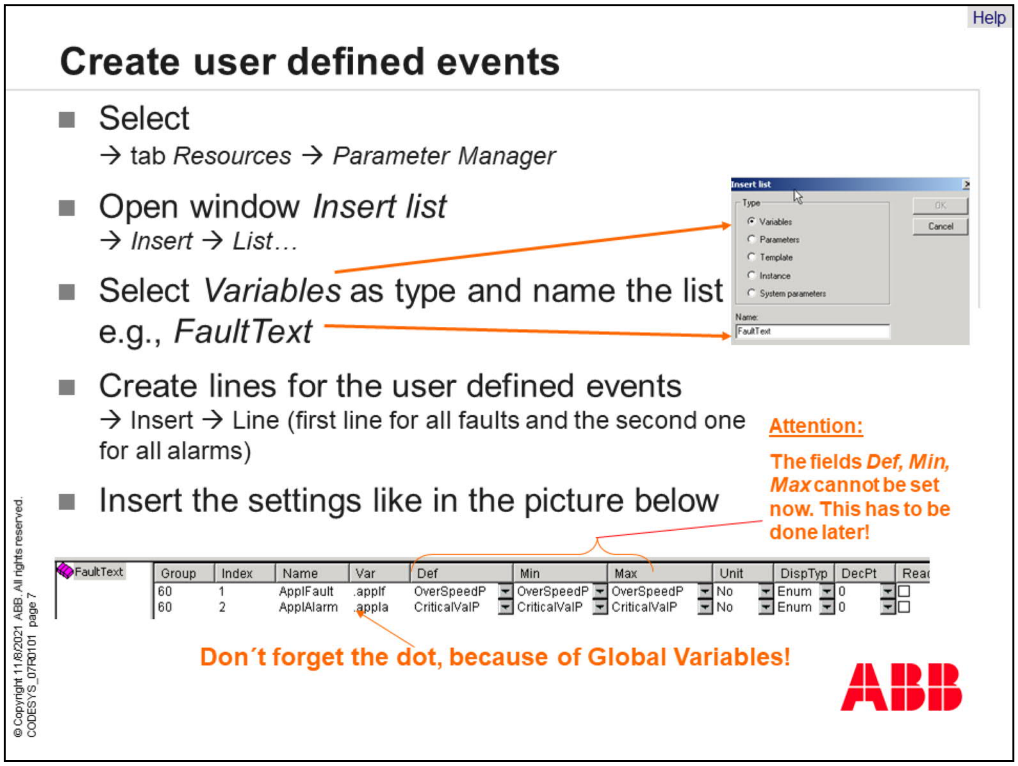
block6
EventInit
9
60
1
60
2
byFaultTextGroup
byFaulttextIndex
byAlarmTextGroup
byAlarmTextIndex
byNoticeTextGroup
byNoticeTextIndex

- EventInit defines the group and parameter for the event
- Configure the communication to the DCS800

  (*Port: COMx; Baudrate: 38400; Parity: Odd; StopBits: 2; Motorola byteorder: Yes*)
- Set the task configuration

Now we need several texts for the alarms and faults. The event messages are saved in parameters which are defined in the “Parameter Manager”. We use function block “Event Init” to allocate the texts for the right event. In this exercise we are saving the fault messages in user parameter 60.01 and the alarms in 60.02.

Then set the communication and task parameters to get a working communication between drive and pc.



Create user defined events

- Select
  → tab *Resources* → *Parameter Manager*
- Open window *Insert list*
  → *Insert* → *List…*
- Select *Variables* as type and name the list e.g., *FaultText*
- Create lines for the user defined events
  → Insert → Line (first line for all faults and the second one for all alarms)
- Insert the settings like in the picture below

● Variables
○ Parameters
○ Template
○ Instance
○ System parameters

Name: FaultText

Attention:

The fields *Def, Min, Max* cannot be set now. This has to be done later!

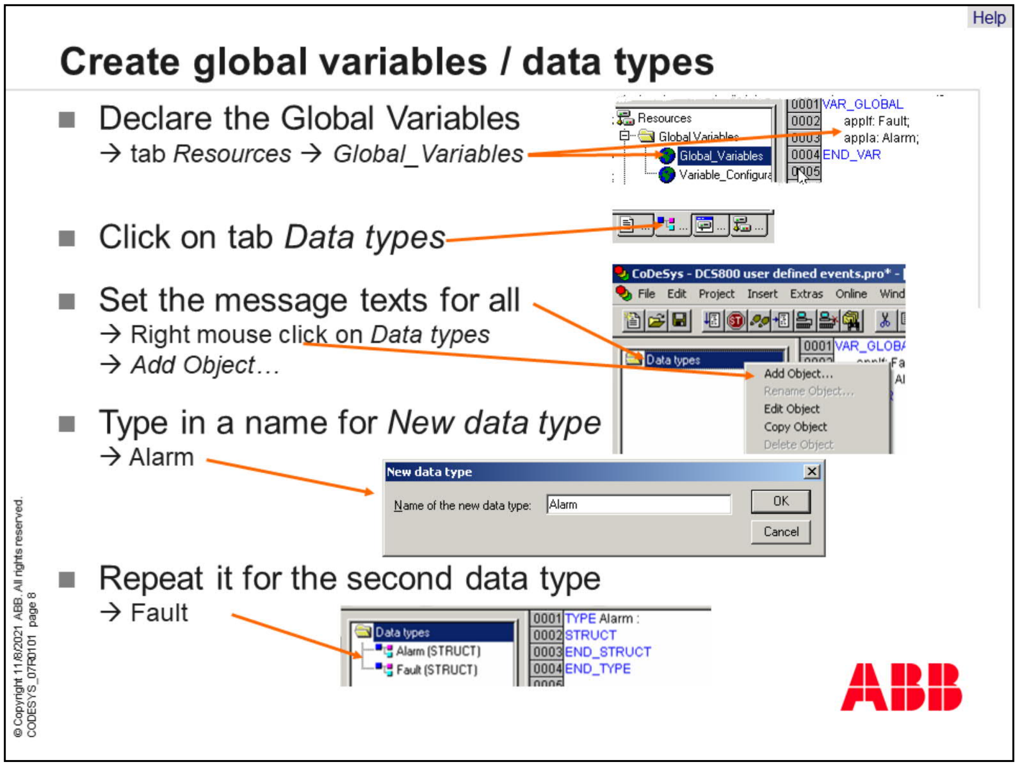
FaultText

Group	Index	Name	Var	Def	Min	Max	Unit	DispTyp	DecPt	Rea
60	1	ApplFault	.applf	OverSpeedP	OverSpeedP	OverSpeedP	No	Enum	0	☐
60	2	ApplAlarm	.appla	CriticalValP	CriticalValP	CriticalValP	No	Enum	0	☐

Don´t forget the dot, because of Global Variables!

Now we change to the "Parameter Manager" and define new user parameters for the message texts. Field Variable is the interface between CoDeSys and the parameter. It must be a global variable which has a dot before the variable. Data type of this parameters are an "Enumeration".

What you can see later in the parameter list is the name of the new parameter group and the name of the parameter index which is defined in field "Name".



Create global variables / data types

- Declare the Global Variables
  → tab *Resources* → *Global_Variables*
- Click on tab *Data types*
- Set the message texts for all
  → Right mouse click on *Data types*
  → *Add Object…*
- Type in a name for *New data type*
  → Alarm
- Repeat it for the second data type
  → Fault

```
VAR_GLOBAL
    applf: Fault;
    appla: Alarm;
END_VAR
```

Name of the new data type: Alarm

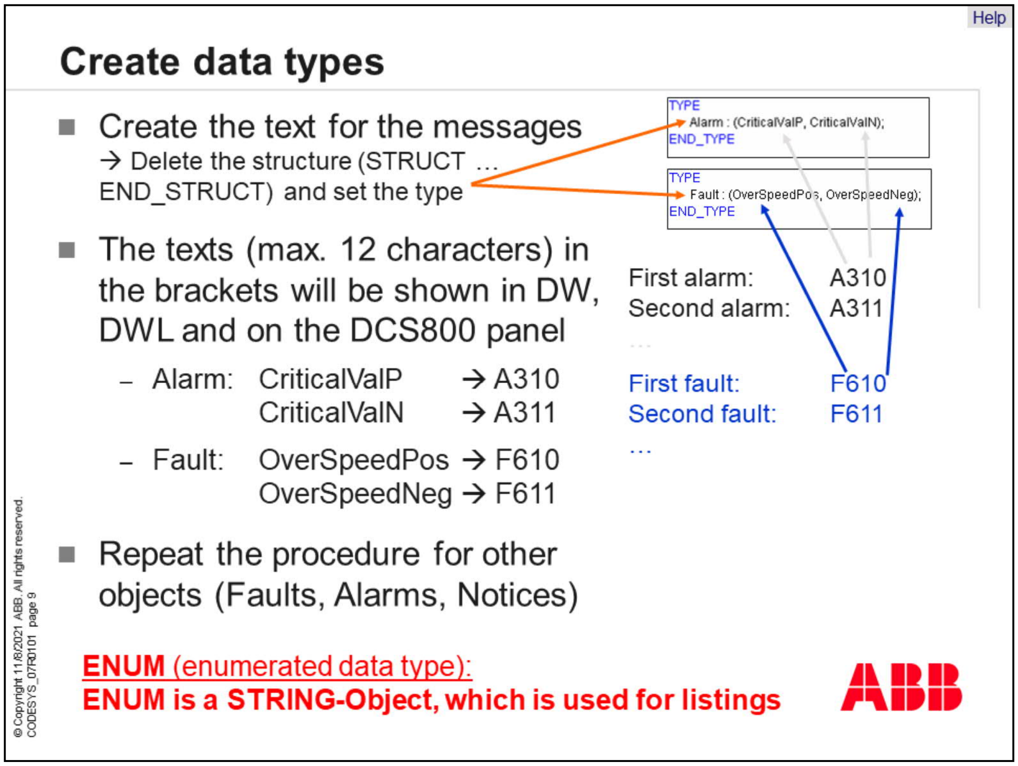
```
TYPE Alarm :
STRUCT
END_STRUCT
END_TYPE
```

CODESYS_07R0101 page 8

The slide before shows that we need global variables to connect CoDeSys with the parameters. So, we must define 2 global ones.

Next step is to define the alarm and fault texts. This can be done in the window “Data Types”. Add a new data type and select a name for the new data type. In this exercise we define the names Alarm and Fault.

Now the 2 new data types are added to the list!



Create data types

- Create the text for the messages
  → Delete the structure (STRUCT … END_STRUCT) and set the type
- The texts (max. 12 characters) in the brackets will be shown in DW, DWL and on the DCS800 panel
  - Alarm: CriticalValP → A310
    CriticalValN → A311
  - Fault: OverSpeedPos → F610
    OverSpeedNeg → F611
- Repeat the procedure for other objects (Faults, Alarms, Notices)

```
TYPE
  Alarm : (CriticalValP, CriticalValN);
END_TYPE
```

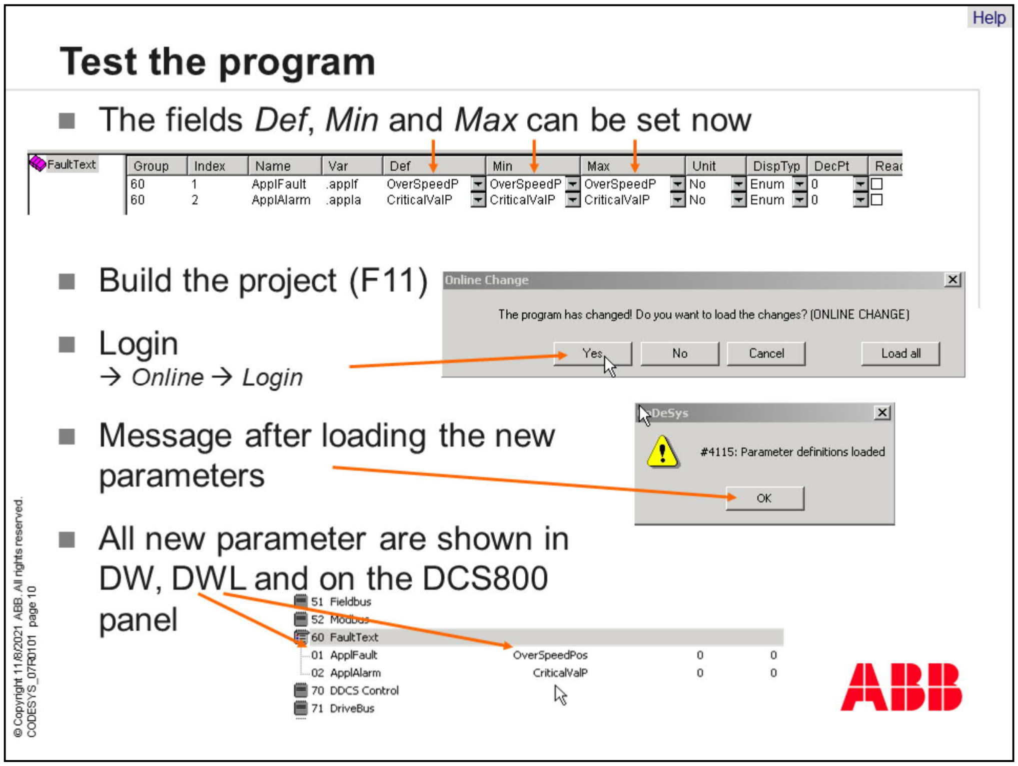
```
TYPE
  Fault : (OverSpeedPos, OverSpeedNeg);
END_TYPE
```

First alarm: A310
Second alarm: A311
…
First fault: F610
Second fault: F611
…

ENUM (enumerated data type):
ENUM is a STRING-Object, which is used for listings

This slide shows how we add texts to the several data types. The message text is in brackets split with a comma. First one in the bracket is allocated to the smallest event number. See also the picture on the right side.

There are in maximum 12 characters allowed!



Test the program

- The fields *Def*, *Min* and *Max* can be set now

Group	Index	Name	Var	Def	Min	Max	Unit	DispTyp	DecPt	Reac
60	1	ApplFault	.applf	OverSpeedP	OverSpeedP	OverSpeedP	No	Enum	0	☐
60	2	ApplAlarm	.appla	CriticalValP	CriticalValP	CriticalValP	No	Enum	0	☐

- Build the project (F11)
- Login
  → *Online* → *Login*

The program has changed! Do you want to load the changes? (ONLINE CHANGE)

- Message after loading the new parameters

#4115: Parameter definitions loaded

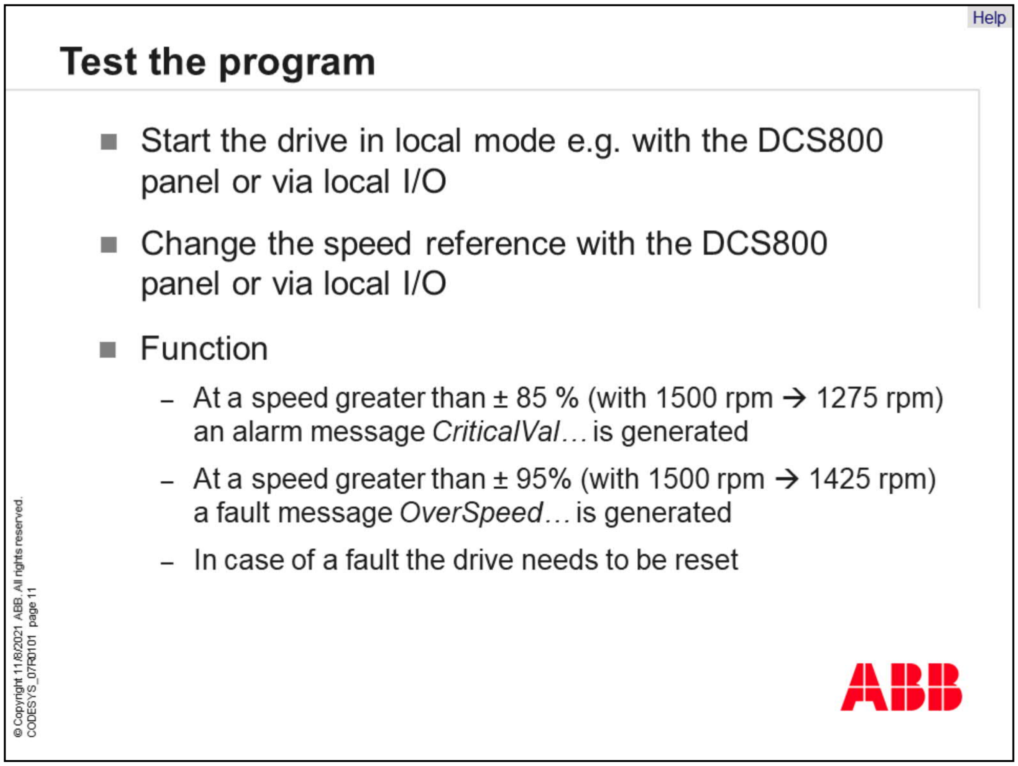
- All new parameter are shown in DW, DWL and on the DCS800 panel

51 Fieldbus			
52 Modbus			
60 FaultText			
01 ApplFault	OverSpeedPos	0	0
02 ApplAlarm	CriticalValP	0	0
70 DDCS Control			
71 DriveBus			

Now the event messages are defined, and we can set the values in the “Parameter Manager”. Normally, we set the first text in all fields because we don’t need a selection for this parameter. It’s only to store data.

If the program is ready, we can download it into DCS800 and test the functionality.

The 2 new parameters will be shown in the parameter list with the default text.



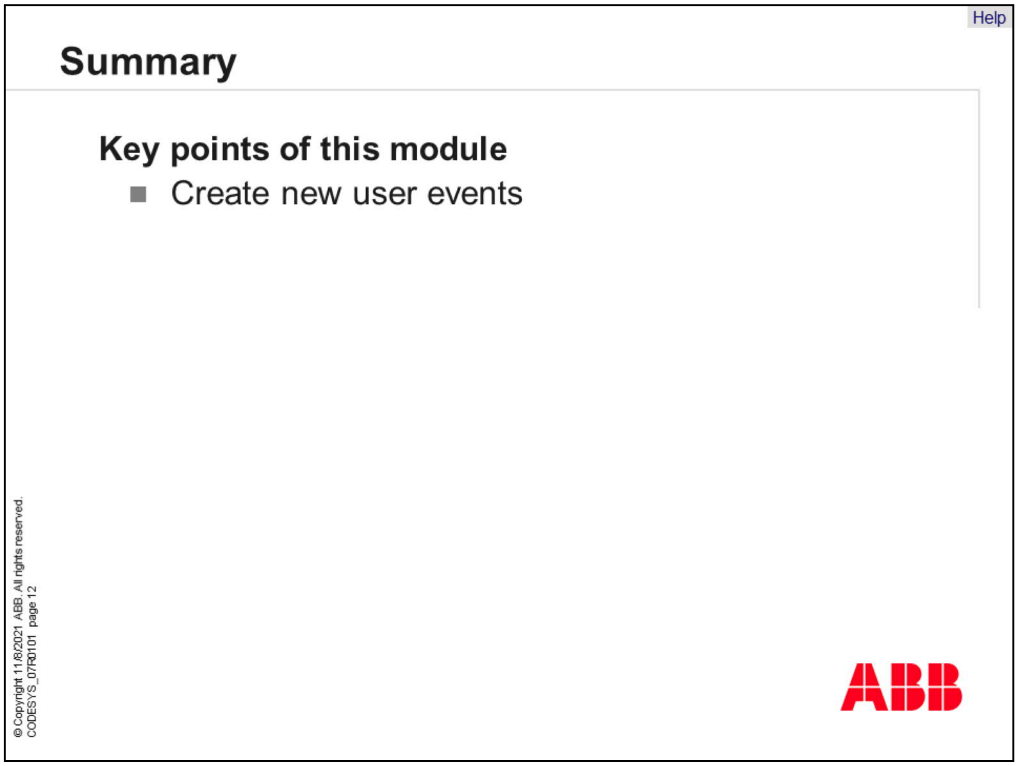
Help

Test the program

- Start the drive in local mode e.g. with the DCS800 panel or via local I/O
- Change the speed reference with the DCS800 panel or via local I/O
- Function
  - At a speed greater than ± 85 % (with 1500 rpm → 1275 rpm) an alarm message *CriticalVal…* is generated
  - At a speed greater than ± 95% (with 1500 rpm → 1425 rpm) a fault message *OverSpeed…* is generated
  - In case of a fault the drive needs to be reset

Before we can do a test with this new application, it necessary to configure the drive first. The easiest way to check the application is to play with the unit in local mode with the switches.

Start the drive and change the speed reference up and down. If the drive is faster than 1275 rpm you get an alarm message. A motor speed faster than 1425 rpm generate a fault and the drive switch off.

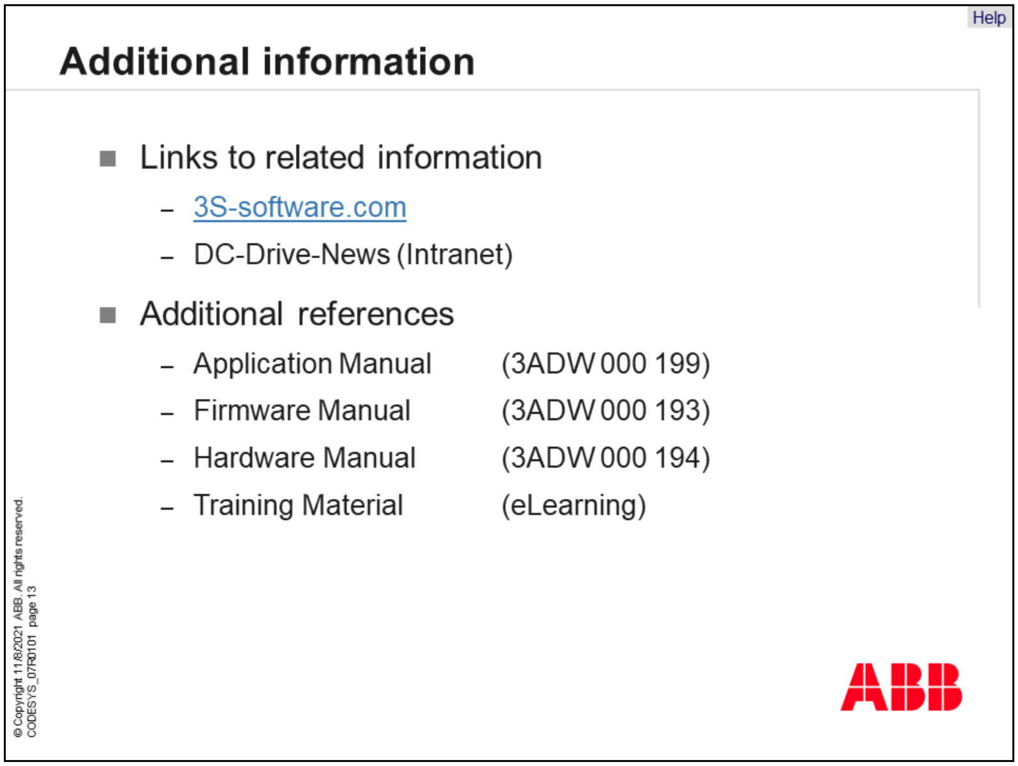


Summary

Key points of this module

- Create new user events

Key points of this module was to create a little application with user events.



Additional information

- Links to related information
  - 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	(eLearning)

Help

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

ABB

Power and productivity
for a better world™

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