

Bending machine 24V

Data modules

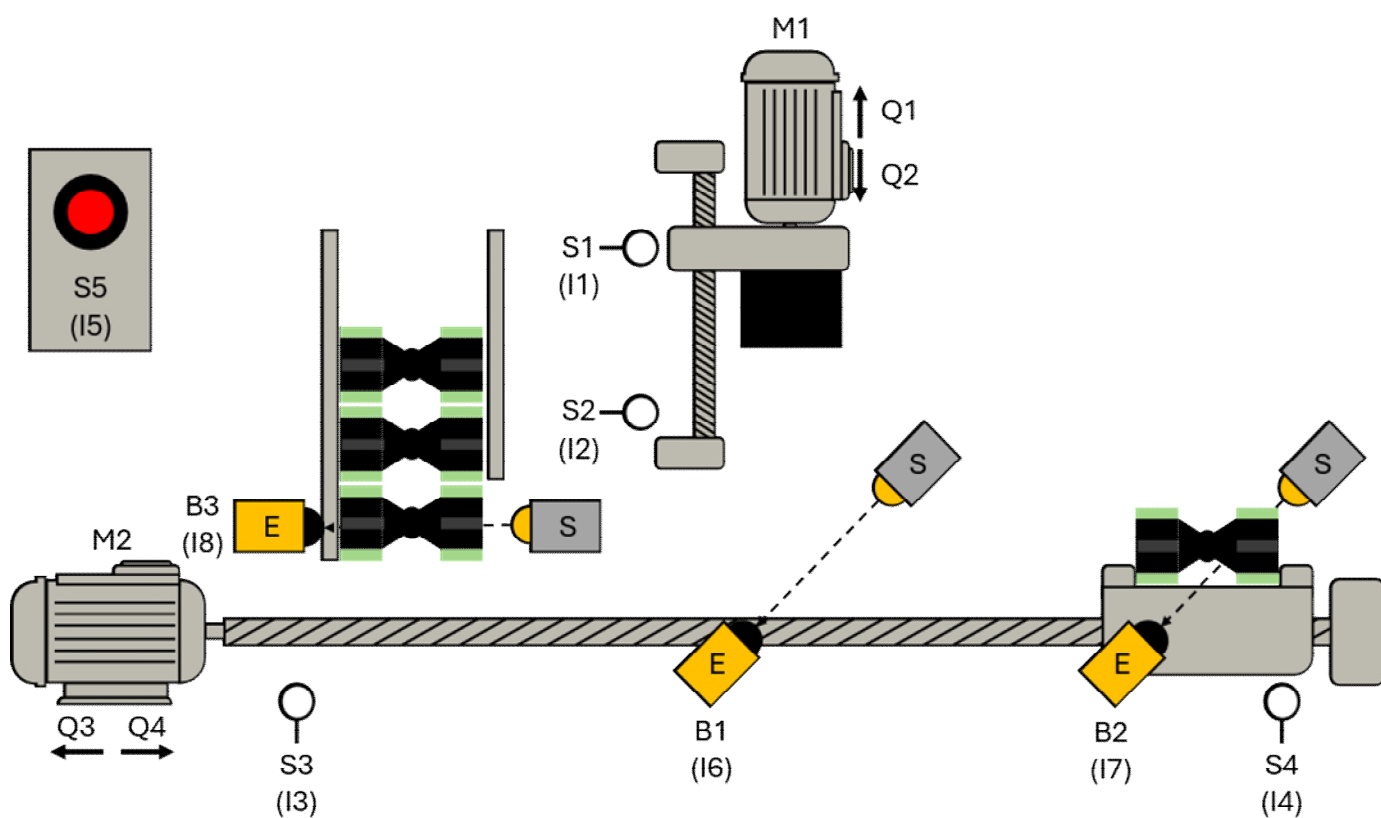


Table of contents

6 Data modules..... 1

6.1 Introduction 1

6.2 Global Data Building Blocks..... 3

6.3 Schematic structure of a data module..... 4

6.4 Instance Data Module..... 6

6.5 Testing Data Modules 8

6.5.1 Observing in Data Module..... 8

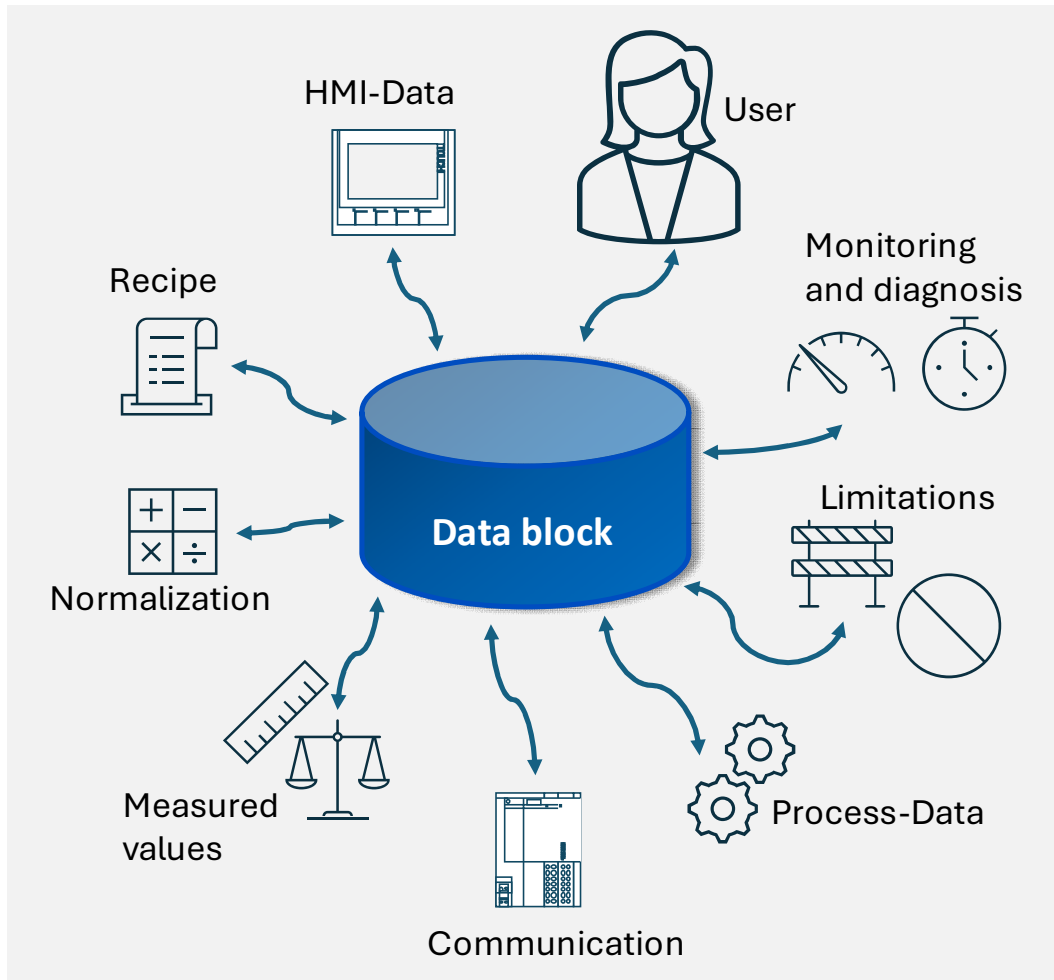
6.5.2 Controlling Operands in Data Module 9

6.5.3 Observing in an observation table..... 10

6 Data modules

6.1 Introduction

Data modules can be used in the TIA project to store data. Unlike Merker, data can be stored in the data module in a structured way and made available for processing. Variables of different data types can be stored in any order. The following image shows examples of data to be filed:



Picture 1 Structured data storage

Unlike functions and function modules, data modules do not contain any program code information, i.e. they do not contain an executable program.

They offer the following advantages:

- Order and structure in the project
- Better overview of plant data
- Faster access in the PLC due to optimized access (no absolute addressing)
- Marker variables can be dispensed with
- Reusability in different projects without conflicts with addresses
- DB variables can be pre-assigned with start values
- Monitoring of variables directly in the DB possible
- Simple backup of current values

Types of data modules

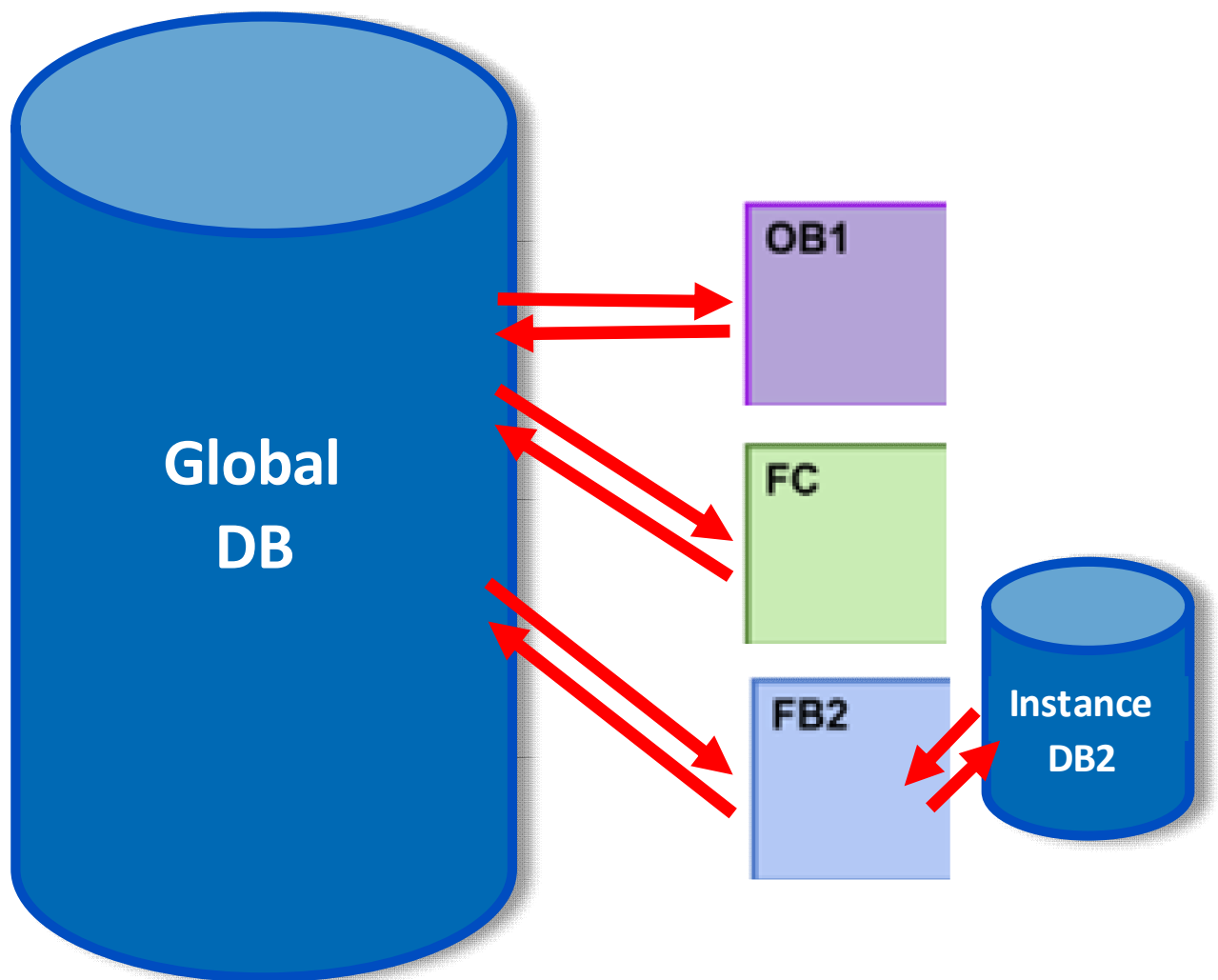
Basically, a distinction is made between two DB types.

Global Data Modules

- The content and the data structure are determined by the user.
- The data can be addressed globally by the user.

Instance Data Modules

- The content and the data structure are defined by the calling function module through its interface.
- The instance data module is automatically generated when FB is called.
- The FB instance (call) accesses the data of its instance DB directly via local interfaces.



Picture 2 Data module types

6.2 Global data building blocks

Global data modules are used to record user data (variables) that can be used by all code modules. The data structure within a global data building block is defined by the programmer.

Global data modules in the user program

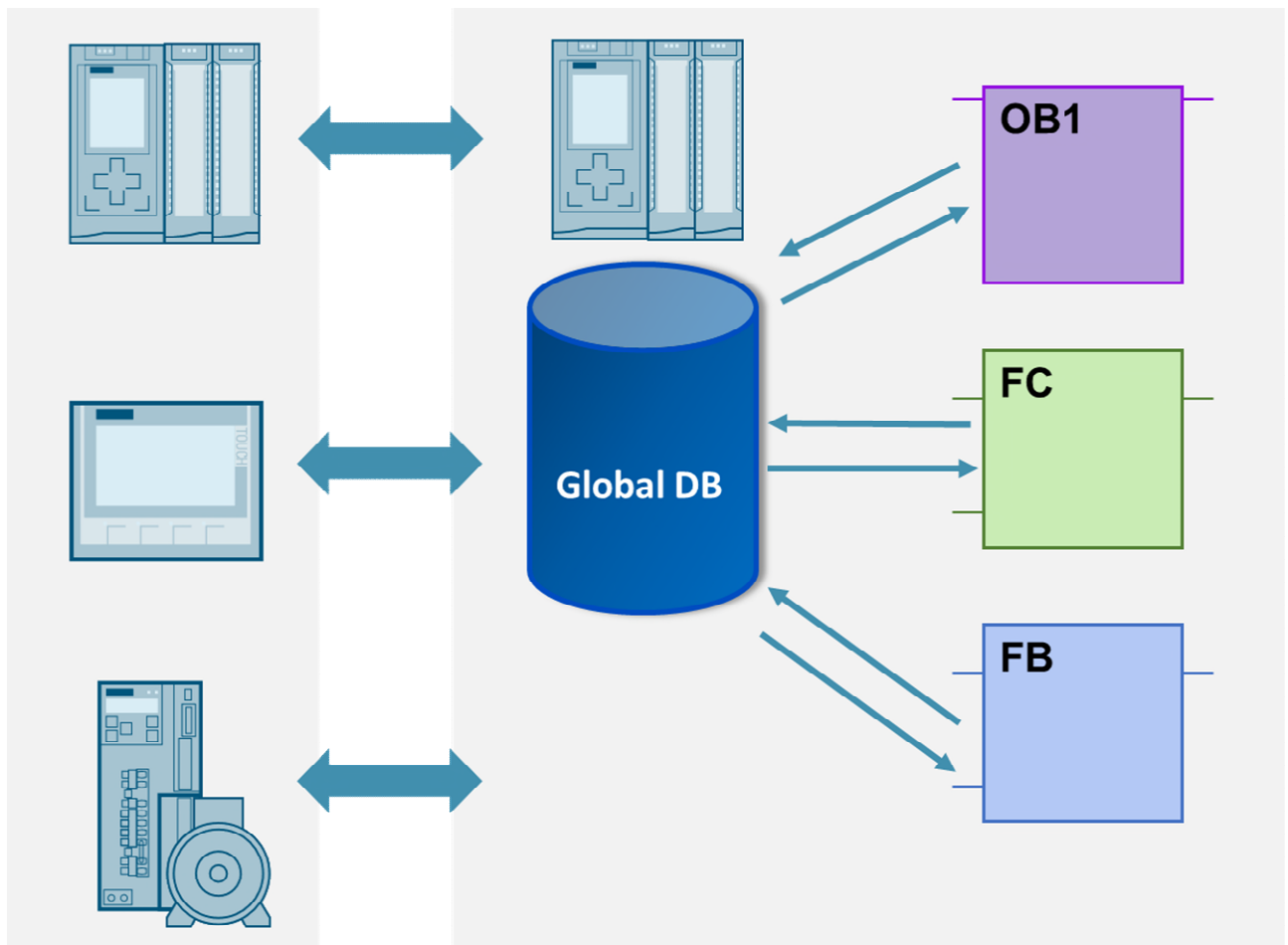
Any FB, FC or OB can read the data from a global DB or write data to a global DB.

Examples of data to be stored are:

- Target or parameter values
- Data area for error messages
- Interface data
- Material tracking data
- Timer Building Block Timing

The use of data modules is recommended for data traffic between different systems:

- Frequency converter control
- CPU-CPU Kommunikation
- HMI Connection



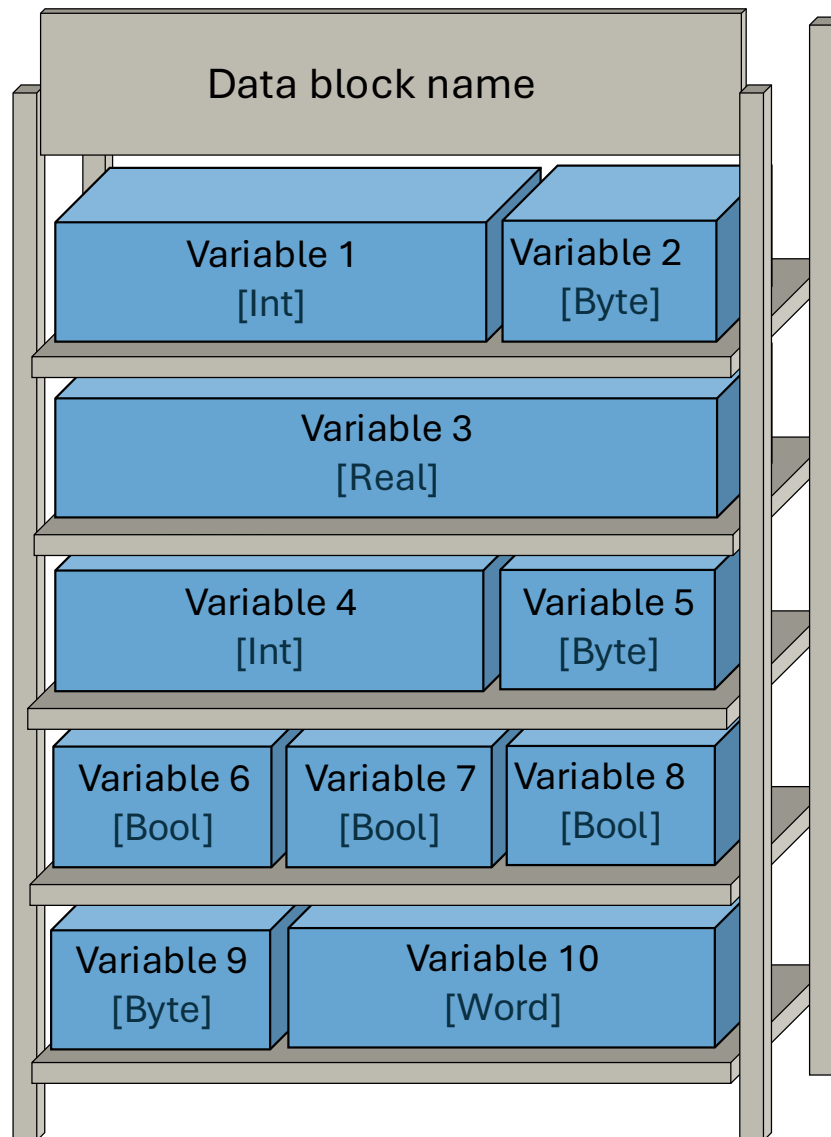
Picture 3 Access to data from a global DB

6.3 Schematic structure of a data module

A data module can be thought of as a shelf. The shelf has a name, the data block name. On this shelf you can create compartments (variables) of different sizes (data type). The size is defined by the data type. Each subject is also given a name.

The data is then accessed via:

Shelf name and technical name → "Data module name". Variable name



Picture 4: Schematic structure of a data module

The length of a data module can be up to 16 MB.

In the standard definition, the data is automatically stored by the system in the data module. It is an optimized data module.

The maximum number of data modules is limited by the memory of the PLC.

Structure of the data module in the editor

The following image shows the structure of a data module in the declaration view.

	Name	Data type	Start value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Comment
1	Static			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
2	resultVolume	Real	0.0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Current volume from Calculate block
3	value1	Dint	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Value 1
4	value2	Dint	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Value 2
5	resultCalculation	Dint	0	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	Result
6	<Add new>			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Picture 5 Structure of a data module

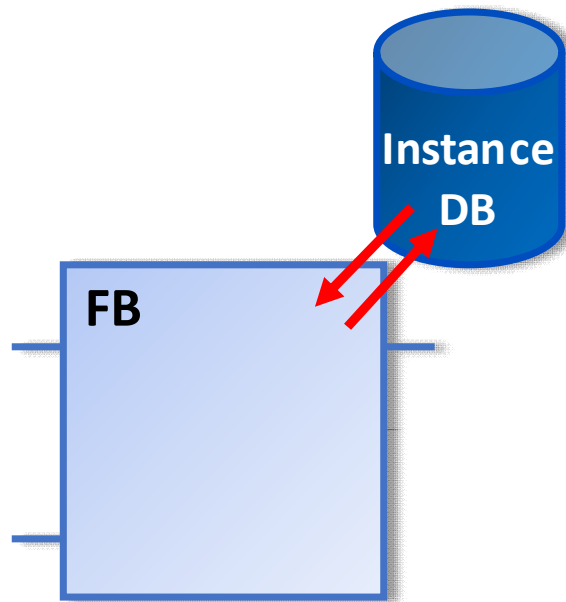
The meaning of each column is shown in the following table:

Column	Meaning
Name	Name of the variable
Data type	Variable data type
Seed	Value that the variable should take on during PLC start-up
Remanence	Marks the variables as remanent.
Reachable from HMI/OPC UA	Indicates whether these variables can be accessed at runtime of HMI/OPC UA.
Writable from HMI/OPC UA	Indicates whether the variable can be described at runtime by HMI/OPC UA.
Visible in HMI Engineering	Indicates whether the variable in the operand selection is visible by default from the HMI.
Setting value	Values that will probably have to be fine-tuned during commissioning. After commissioning, the current values can be adopted as starting values in.
Comment	Kommentar zur Variablen

Picture 6 Columns in the declaration view

6.4 Instance Data Module

Instance data modules are data modules that are automatically generated when a system function module or a function module is called.



Picture 7 Instance Data Module

The data of the building block is stored in these instance data modules. They are directly assigned to the building block. For each call to a function module, a new instance data module must be created.

The generated instance modules are stored under System Modules → Program Resources or Program Module General.

The data structure depends on the programming of the function module used. The following image shows an instance data module of the system function module of a counter.

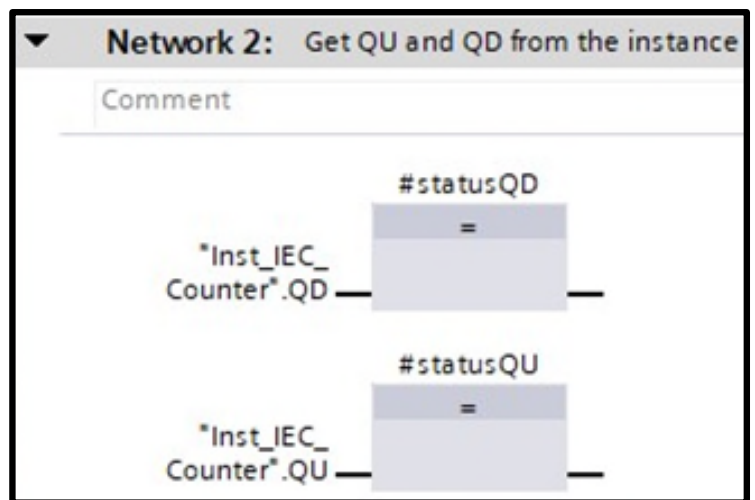
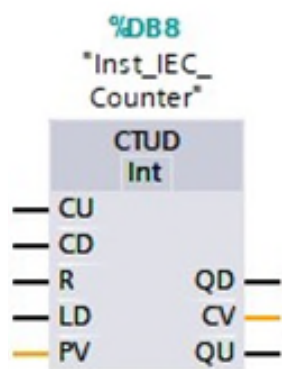
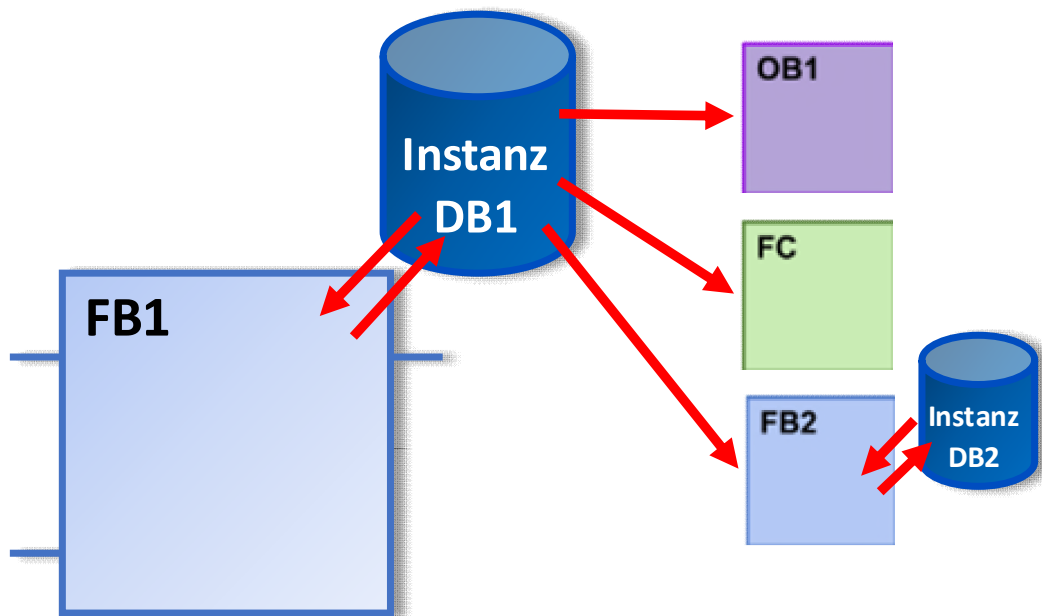
		Inst_IEC_Counter			
		Name	Data type	Start value	Monitor value
	1	Static			
	2	CU	Bool	false	FALSE
	3	CD	Bool	false	FALSE
	4	R	Bool	false	FALSE
	5	LD	Bool	false	FALSE
	6	QU	Bool	false	TRUE
	7	QD	Bool	false	FALSE
	8	PV	Int	0	6
	9	CV	Int	0	8

Picture 8 Instance data module of an IEC counter



Instance data modules can also be created manually using the function "Add new module → Data module" and selection of the corresponding function module.

Variables of an instance DB can be accessed by FC, FB, and OB both read-and-write.
 The following image shows access to the QU and QD of a counter.



Picture 9 Access to Instance Data

Only read access is recommended, **not** writing. The program can become confusing, especially due to write accesses, as no cross-references are displayed.

6.5 Testing of data modules

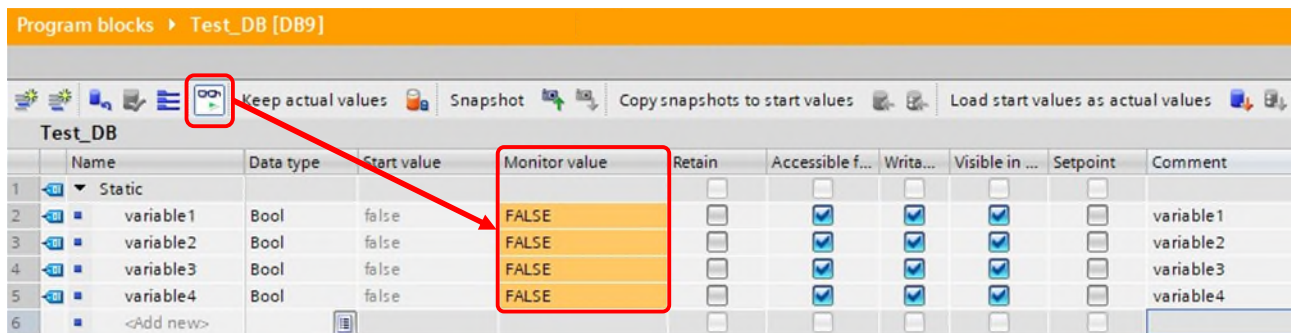
To test the values of a data module, you can observe and control the variables via an existing online connection directly in the open data module or via an observation table.

6.5.1 Observing in the data module

To do this, activate the "Watch All" button in the function bar of the declaration table. A new column "Observation value" is inserted in the declaration table, in which the current actual value can be observed.

A picture that contains table. Auto-generated description

All variables are represented with the current value of the PLC as an "observation value" and are constantly updated.



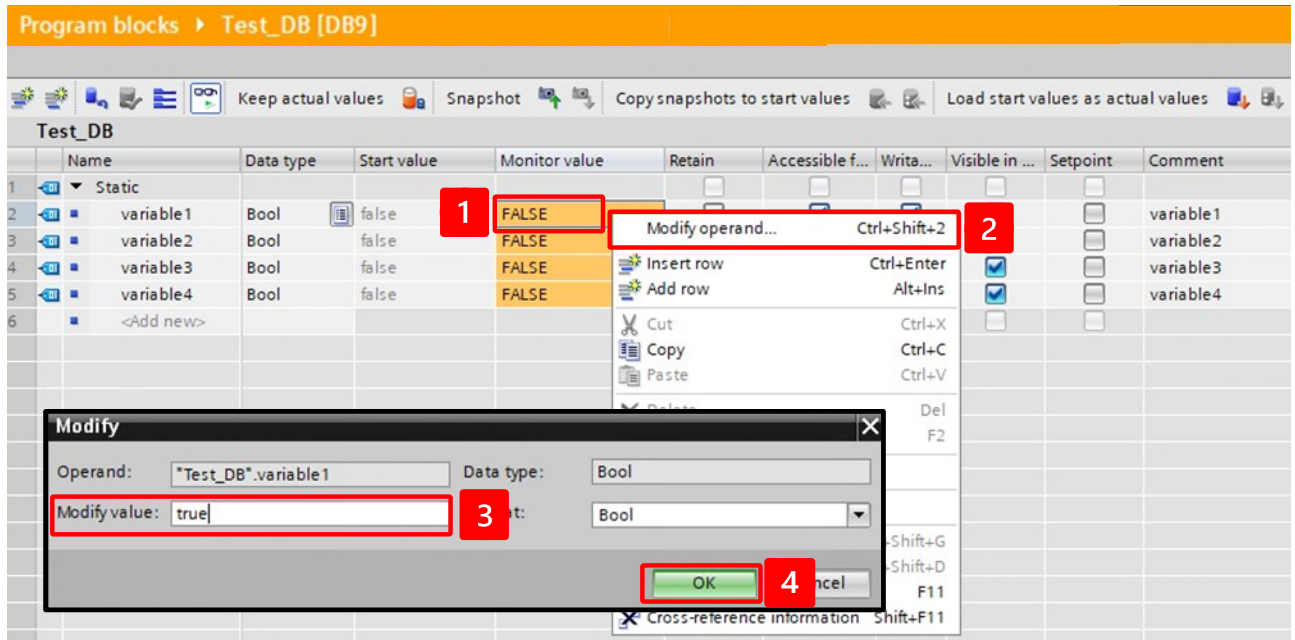
	Name	Data type	Start value	Monitor value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint	Comment
1	Static									
2	variable1	Bool	false	FALSE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	variable1
3	variable2	Bool	false	FALSE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	variable2
4	variable3	Bool	false	FALSE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	variable3
5	variable4	Bool	false	FALSE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	variable4
6	<Add new>									

Picture 10 Data module in the observation function

6.5.2 Controlling operands in the data module

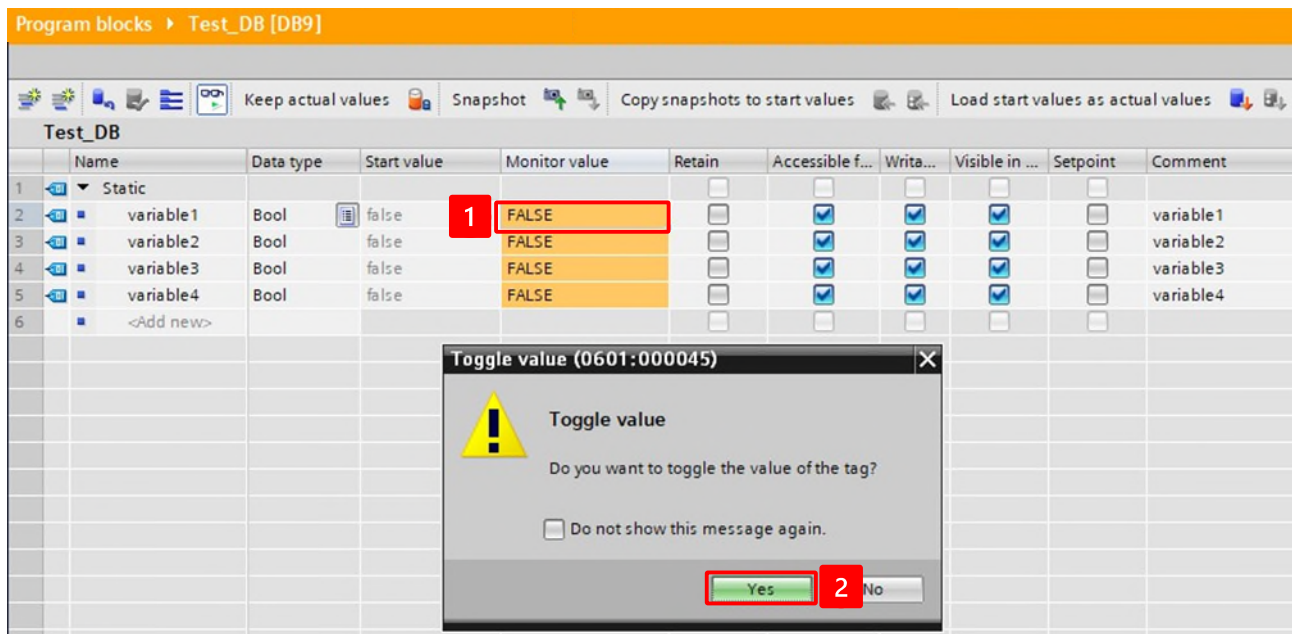
In the data module, you have the option of right-clicking on a variable with the function "Control Operand..." to change the observation value.

In the following "Taxes" window, specify the tax value. This is adopted as the new observation value by clicking on "OK", the starting value remains unchanged.



Picture 11 Data module Controlling value

Controlling binary variables is also possible in this way. Boolean variables can also be switched directly by double-clicking on their observation value.



Picture 12 Toggle Boolean Variable Data Module

6.5.3 Observing in an observation table

You can enter and observe the variables of the data module in an observation table. From here, you can also change the value of the variable using the tax value.

	Name	Address	Display format	Monitor value	Modify value		Comment
1	"Test_DB".variable1		Bool	<input checked="" type="checkbox"/> TRUE	<input type="text" value="TRUE"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
2	"Test_DB".variable2		Bool	<input type="checkbox"/> FALSE		<input type="checkbox"/>	<input type="checkbox"/>
3	"Test_DB".variable3		Bool	<input type="checkbox"/> FALSE		<input type="checkbox"/>	<input type="checkbox"/>
4	"Test_DB".variable4		Bool	<input type="checkbox"/> FALSE		<input type="checkbox"/>	<input type="checkbox"/>
5		<Add new>				<input type="checkbox"/>	<input type="checkbox"/>

Picture 13 Observation table

You can open a data module, copy out the desired variables via the Windows clipboard and paste them into the observation table.