<Operationmanual_Basic Gripping_S7_V1_21(DE).docx>

topic: <FB Basic Gripping>

version: <1.21>

History

Author	Reason for change/changes made	Release	Date
Armbruster	Creation	1.0	
Nock	Display parameter change with output bit Automatic reset of the direction flags	1.21	18.03.2020

Content

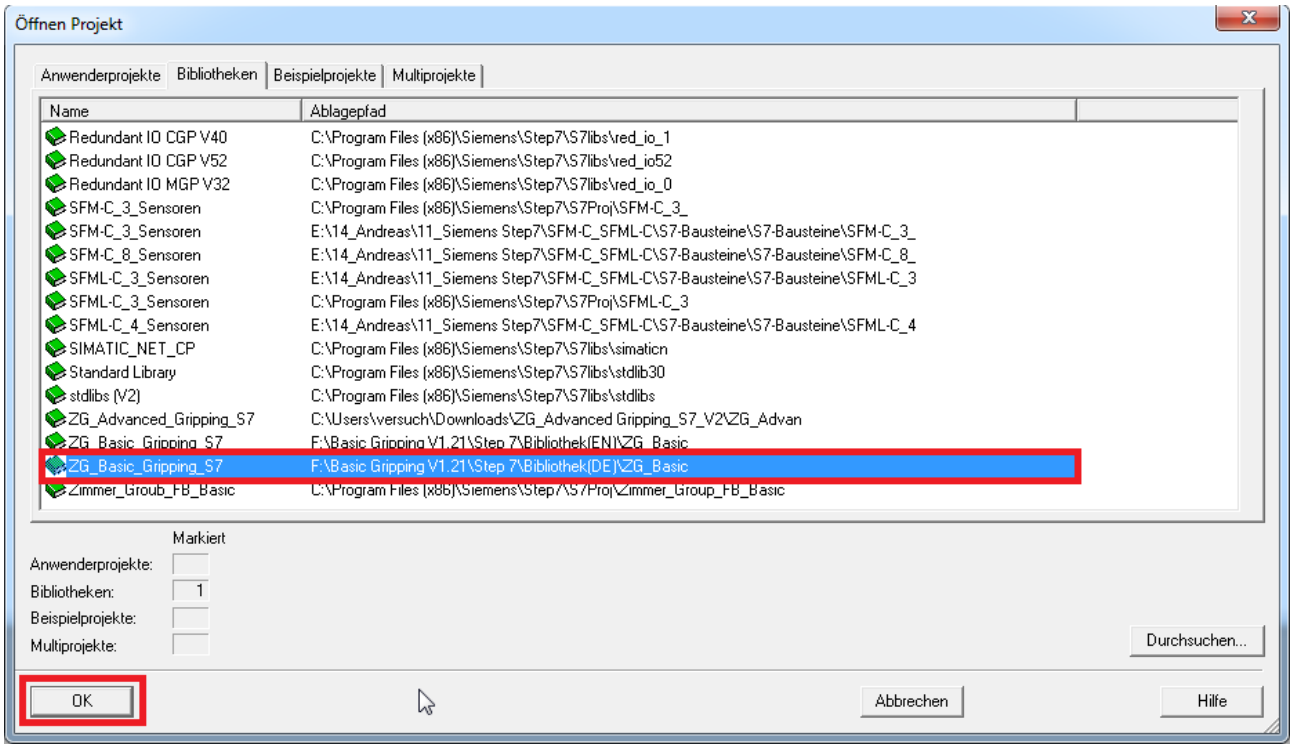
Basic Gripping

1 Foreword

To use the example program, a correct hardware configuration must first be created. In this example a Siemens CPU314C-2 PN/DP with a Balluff IO Link master is used. After the hardware settings the function block can be implemented. To do this, go through the following steps.

2 Integrating the library

The function block is provided as a library. To open it, please select the "File" -> "Open..." menu. In the now visible window you can select and open the desired library.
Please select the global library " **ZG_Basic_Gripping_S7**".



Now the Zimmer library is ready for use and can be used in your project.

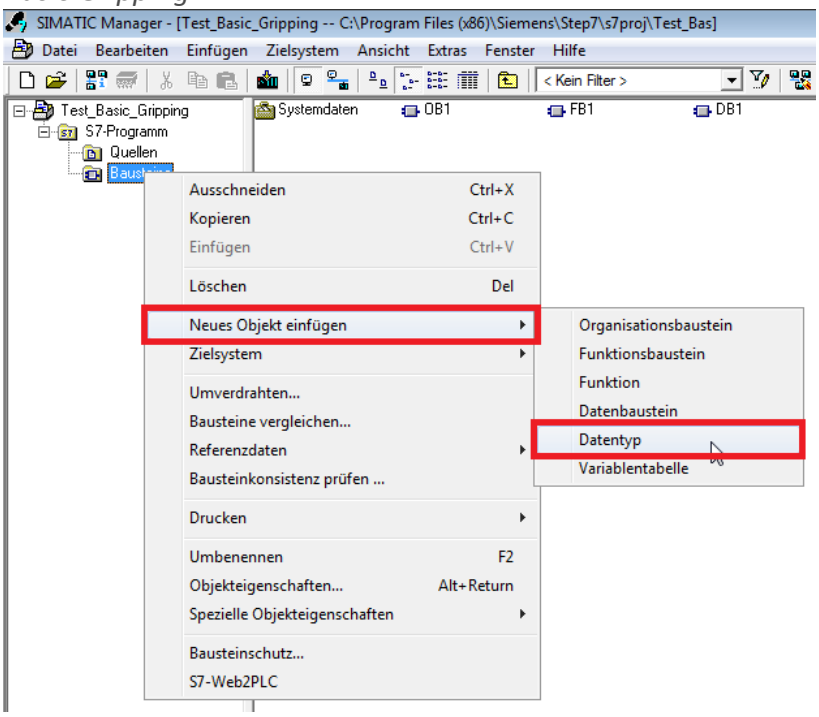
3 Declare data types

The input and output data of the gripper - or also called "process data" - are transferred to the function block via specific data types (structures or UDT). The following data types must be created for communication between the function block and the gripper:

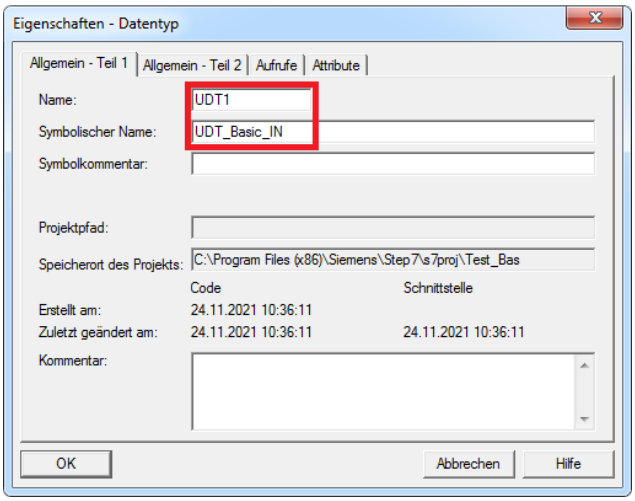
Name	Symbolic name	Includes
UDT1	UDT_Basic_IN	6 bytes input data
UDT2	UDT_Basic_OUT	8 bytes output data

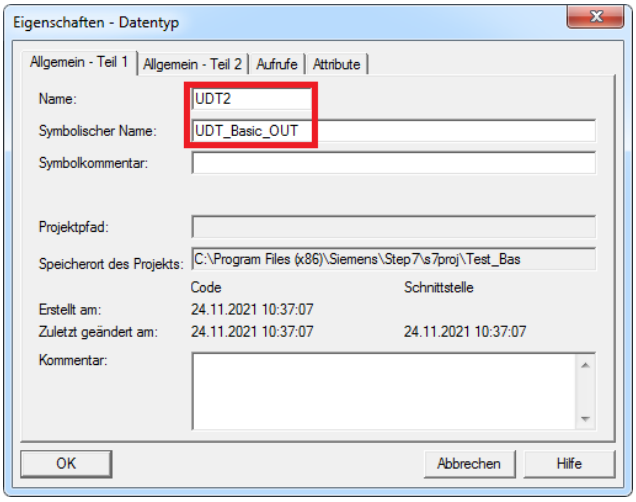
A new data type can be inserted into the project by right-clicking on "Blocks" -> "Insert new object" -> "Data type".

Basic Gripping



The "Symbolic name" of the UDT is defined in the window that opens. Please enter the given name.





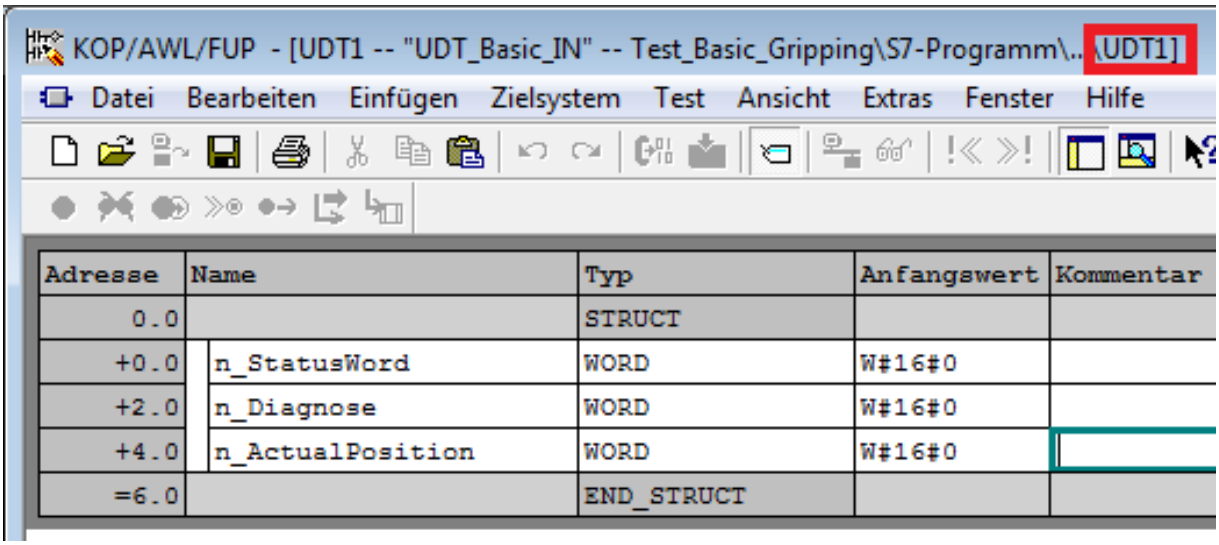
Please open the corresponding data type and define it as follows. The sequence should be followed.

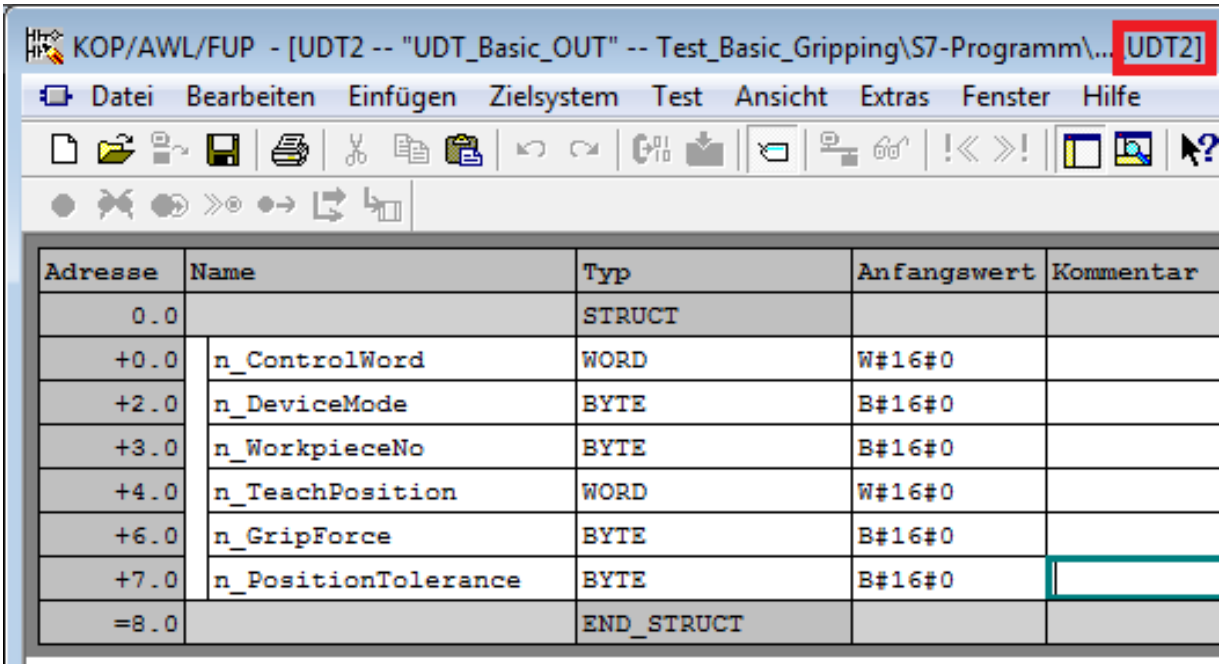
UDT1:

Name	Type	Initial value
n_StatusWord	WORD	W#16#0
n_Diagnose	WORD	W#16#0
n_ActualPosition	WORD	W#16#0

Basic Gripping
UDT2:

Name	Type	Initial value
n_ControlWord	WORD	W#16#0
n_DeviceMode	BYTE	B#16#0
n_WorkpieceNo	BYTE	B#16#0
n_TeachPosition	WORD	W#16#0
n_GripForce	BYTE	B#16#0
n_PositionTolerance	BYTE	B#16#0





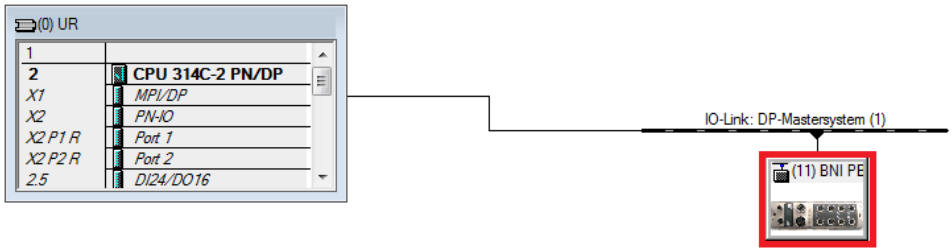
4 Declare global structures in the OB

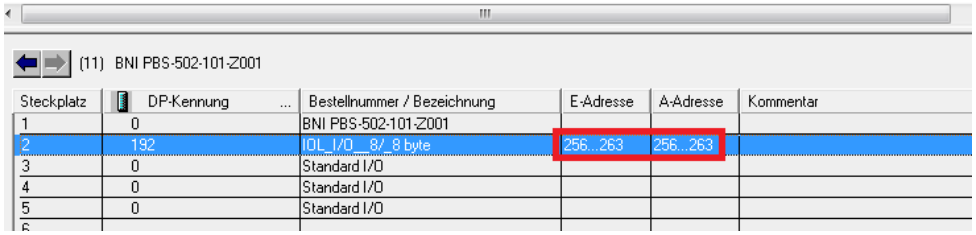
In the used operation block global variables must be created from the created structures to link them to the block. Please create the following variables in the OB in the declaration table:

Name	Datentyp	Adresse	Kommentar
OB1_EV_CLASS	Byte	0.0	Bits 0-3 = 1 (Coming event), Bits 4-7 = 1 (Event class 1)
OB1_SCAN_1	Byte	1.0	1 (Cold restart scan 1 of OB 1), 3 (Scan 2-n of OB 1)
OB1_PRIORITY	Byte	2.0	Priority of OB Execution
OB1_OB_NUMBR	Byte	3.0	1 (Organization block 1, OB1)
OB1_RESERVED_1	Byte	4.0	Reserved for system
OB1_RESERVED_2	Byte	5.0	Reserved for system
OB1_PREV_CYCLE	Int	6.0	Cycle time of previous OB1 scan (milliseconds)
OB1_MIN_CYCLE	Int	8.0	Minimum cycle time of OB1 (milliseconds)
OB1_MAX_CYCLE	Int	10.0	Maximum cycle time of OB1 (milliseconds)
OB1_DATE_TIME	Date_And_Time	12.0	Date and time OB1 started
g_UDT_Gripper1_IN	UDT_Basic_IN	20.0	
g_UDT_Gripper1_OUT	UDT_Basic_OUT	26.0	

5 Linking process data with peripherals

In the hardware configuration, address ranges for the input and output data have been assigned to the IO-Link port to which the gripper is connected. In this example, the range starts at peripheral address 256.



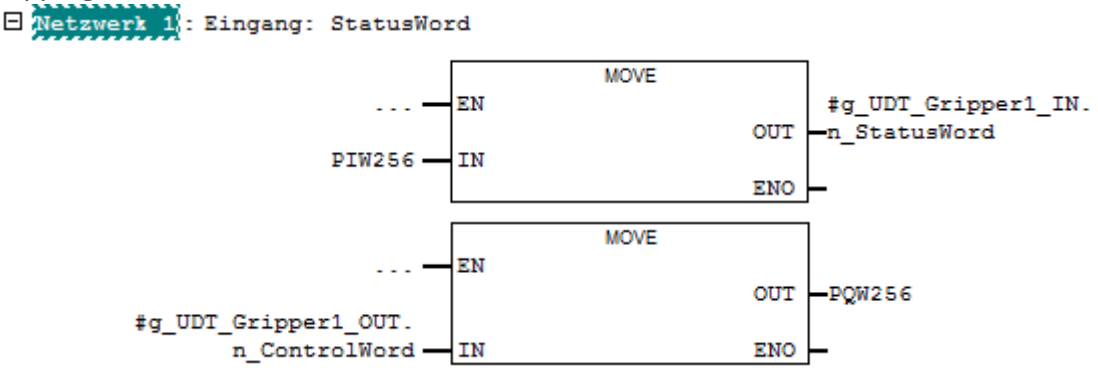


In the operation block, the peripheral addresses must be linked with the global structures created. In our example "MOVE" blocks were used for the assignments. The input data should be linked in the OB before the FB and the output data after the FB. When assigning the addresses, the sequence of the global structure must be observed.

In this example the assignments - starting with address 256 - look as follows:

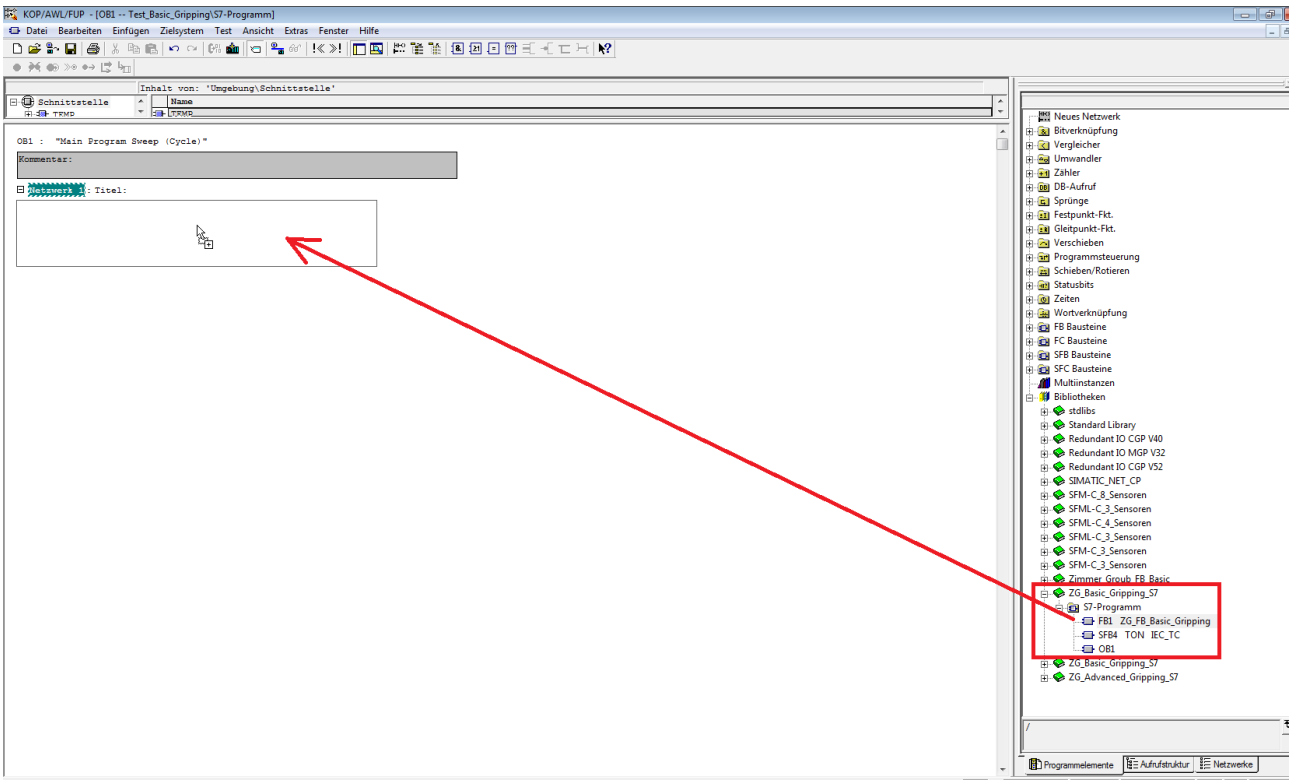
Assignment of …	**Assignment to …**
PIW256	g_UDT_Gripper1_IN.n_StatusWord
PIW258	g_UDT_Gripper1_IN.n_Diagnose
PIW260	g_UDT_Gripper1_IN.n_ActualPosition
g_UDT_Gripper1_OUT.n_ControlWord	PQW256
g_UDT_Gripper_1OUT.n_DeviceMode	PQB258
g_UDT_Gripper_1OUT.n_WorkpieceNo	PQB259
g_UDT_Gripper_1OUT.n_TeachPosition	PQW260
g_UDT_Gripper_1OUT.n_GripForce	PQB262
g_UDT_Gripper_1OUT.n_PositionTolerance	PQB263

Basic Gripping



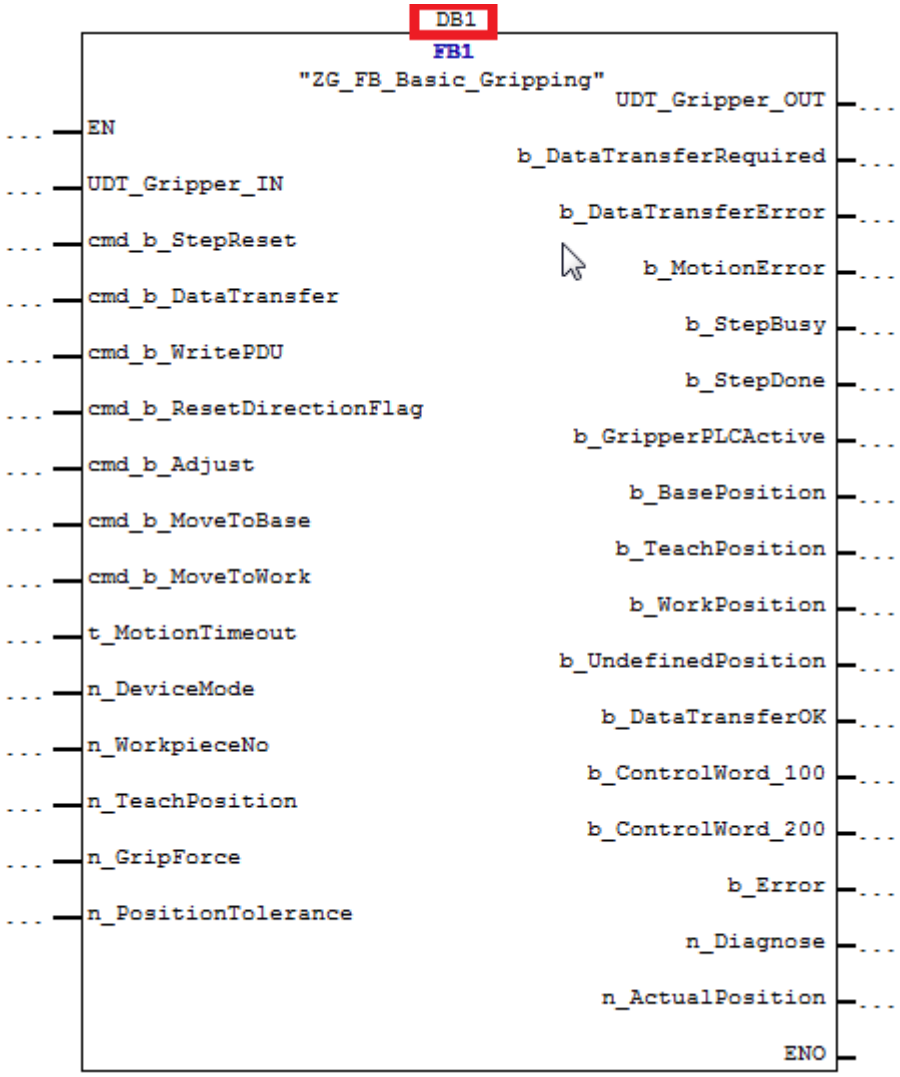
6 Insert function block

To be able to use the "Basic-Gripping" block, it must be called in the desired operation block. The "ZG_FB_Basic_Gripping" block can be copied from the already integrated library to a free network using drag & drop. The standard block "SFB4 (TON)" is also present in this library, since it is required in the "Basic-Gripping" block.



Basic Gripping

Since this is a function block, an instance data block is required. In this example, a new data block DB1 was generated.



Variables which are declared with „b_“ are binary signals.

Variables which are declared with „cmd_“ are command inputs. They can be controlled with a push button switch for example.

Variables which are declared with „n_“ are input and output words or bytes. They are needed for transfering of the several positions and functions.

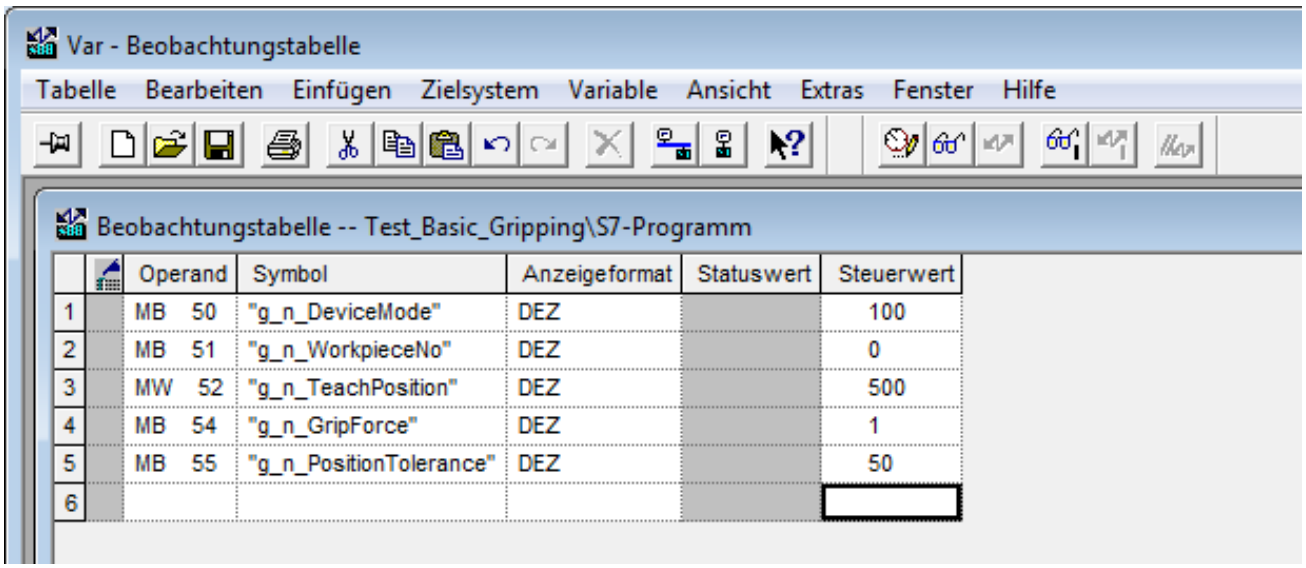
7 Using the function block

On the inserted function block in chapter 6 are some inputs and outputs which must be wired. The input „UDT_Gripper_IN“ must be connected now with the belonging variable in chapter 4. The same must be done with the output variable „UDT_Gripper_OUT“. Now the function block can read the several states and positions of the gripper and can handle them. The output wiring of the gripper can also be parameterized.

For moving the gripper the several position data and options (drive profiles) have to be transfered. For standard values the following values in the table can be used. These are exemplary and can vary in different projects. You can set these parameters like in in our example on the block as constants or you can also use variables with the correct length that the wiring is flexible. If there is no wiring, the variables are preinitialised with the standard values.

n_DeviceMode	100 (1 at GEP/GED5000IL)
n_WorkpieceNo	0
n_TeachPosition	500
n_GripForce	1
n_PositionTolerance	50

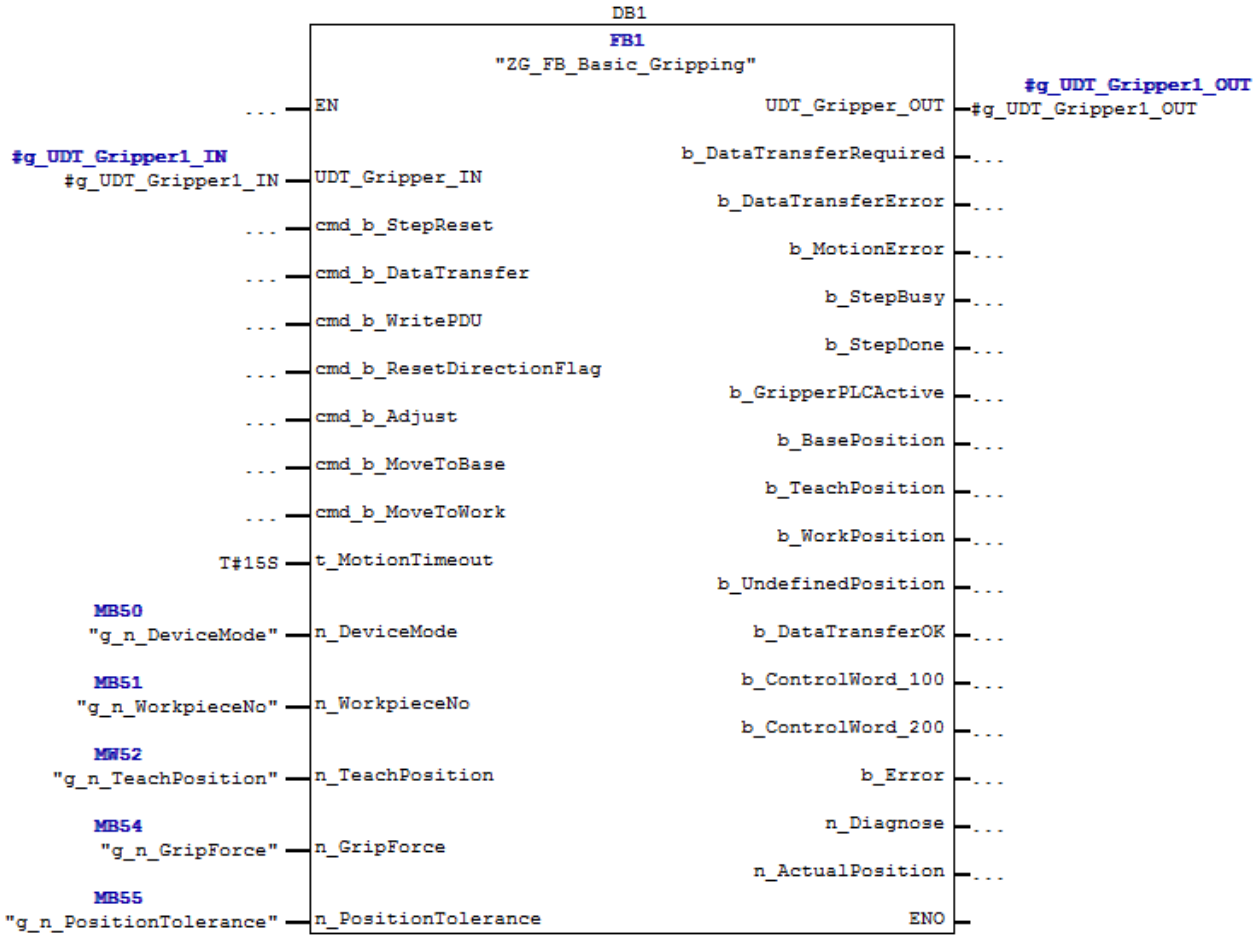
The variable n_DeviceMode corresponds to the drive profile of the gripper. These drive profiles can be found in the operation manual of the gripper. In this example DeviceMode 100 (at GEP2000IL or GPP5000IL) or 1 (at GEP/GED5000IL) was selected which corresponds to the drive profile „Universal operation“ and can be used as a standard value.



	Operand	Symbol	Anzeigeformat	Statuswert	Steuerwert
1	MB 50	"g_n_DeviceMode"	DEZ		100
2	MB 51	"g_n_WorkpieceNo"	DEZ		0
3	MW 52	"g_n_TeachPosition"	DEZ		500
4	MB 54	"g_n_GripForce"	DEZ		1
5	MB 55	"g_n_PositionTolerance"	DEZ		50
6					

Basic Gripping

The completed function block should now look like in the following picture:



At last the settings have to be compiled and transfered to the device.

8 Functions of the function block

Depending on the wiring of the function block, several functions are carried out. You can find more information in the header of the block.

8.1 Resetting the step sequence „cmd_b_StepReset“ (BOOL)

The input variable „cmd_b_StepReset“ resets the step sequence in this function block. It doesn’t depend on in which step the function block is at that moment. When the function block puts the error „b_DataTransferError“ or „b_MotionError“ out, it only can be resetted with this input.

8.2 Transfering data with handshake „cmd_b_DataTransfer“ (BOOL)

After each change of a process parameter (except "ControlWord") or during a cold start of the gripper, the parameters must be accepted with a data transfer. If the output variable "b_DataTransferRequired" is "TRUE", the gripper hasn’t worked with the currently set parameters yet. In this case the input "cmd_b_DataTransfer" must be triggered that the process parameters are transferred. Then the variable "b_DataTransferRequired" changes to "FALSE". Thereby the "ControlWord" is set to value 1 and bit 12 of the "StatusWord" is waited for. Bit 12 becomes "TRUE" as soon as the data transfer is finished. Then the "ControlWord" is set to 0 again and waited until bit 12 becomes "FALSE". This procedure is a handshake and should be used for flawless data transmission.

8.3 Saving workpiece recipes „cmd_b_WritePDU“ (BOOL)

When this input is set to „TRUE“, the actual written process parameters at the input side of the function block are saved into the selected „WorkpieceNo“. This function sets the „ControlWord“ to the value 2 and waits for the bit 12 of the „StatusWord“. This procedure can last up to 30 seconds. The parameters are saved power failure safe in the gripper and they can be selected again with writing the „WorkpieceNo“. Up to 32 recipes can be safed in the gripper.

8.4 Resetting the direction flags „cmd_b_ResetDirectionFlag“ (BOOL)

When a gripper was moved to WorkPosition for example, the bit 14 of the „StatisWord“ is set. This signal keeps alive til a movement into the other direction or a new startup of the gripper. When a gripper must be driven to the same direction more than one time, this bit must be resetted before. This can be done with the input „cmd_b_ResetDirectionFlag“. This function sets the „ControlWord“ to the value 4 and waits for bit 13 and bit 14 of the „StatusWord“ becoming „FALSE“. After that it can be moved again into the same direction. Since the version 1.21 of the function block, this procedure has been carried out automatically before a movement of the gripper.

8.5 Drive to BasePosition „cmd_b_MoveToBase“ (BOOL)

When this input is set to „TRUE“, the gripper fingers move with the setted drive profile and grip force to the „BasePosition“. This function sets the „ControlWord“ to the value 256.

8.6 Drive to WorkPosition „cmd_b_MoveToWork“ (BOOL)

When this input is set to „TRUE“, the gripper fingers move with the setted drive profile and grip force to the „WorkPosition“. This function sets the „ControlWord“ to the value 512.

8.7 Limitting of the motion time „t_MotionTimeout“ (TIME) and „b_MotionError“ (BOOL)

If the gripper can‘t carry out a movement or can’t reach the required destination, the step sequence will stop and the function block will be blocked for further commands. To avoid this struggle, the time „t_MotionTimeout“ at the input can be defined. It is the maximum time which is allowed for the gripper’s movement til arriving the position. It depends on the input parameters and have to be adjusted for your applikation. If the gripper doesn’t reach its required destination in the setted time, the step sequence jumps into a error step. The output „b_MotionError“ is set to „TRUE“ and only can be resetted again with the input „cmd_b_StepReset“.

8.8 Data transfer is required „b_DataTransferRequired“ (BOOL)

The variable „b_DataTransferError“ is active when at least on of the output variables which are sent to the gripper has been changed. As long as this variable is active, the gripper hasn’t confirmed the changed values yet. For transfering the data the input variable „cmd_b_DataTransfer“ must be triggered. Then the variable „b_DataTransferRequired“ changes to „FALSE“ and the gripper uses the actual set parameters.

8.9 Error in the DataTransfer „b_DataTransferError“ (BOOL)

The output „b_DataTransferError“ is set to „TRUE“ when the data transfer („ControlWord“ = 1) couldn’t be carried out successfully and the feedback of the gripper wasn’t sent in the first second. There can be several reasons for this. An error code can be taken from the output „n_Diagnose“. All the error codes are described in detail in the operation manual. This error can be resetted with setting the input „cmd_b_StepReset“.

8.10 Function block is busy „b_StepBusy“ (BOOL)

If the function block handles a command and is not in the intial step, this output is active and shows, that it is blocked for further commands.

8.11 Ready for commands „b_StepDone“ (BOOL)

If the function block is in the initial step and is ready for commands, this output is set to „TRUE“.

8.12 Bit 6 of the StatusWord „b_GripperPLCActive“ (BOOL)

This signal shows that the controller inside of the gripper is ready for operation. When the gripper is plugged in again or it is restarted after a voltage breakdown, the controller can only recieve data when this signal is set again.

8.13 Bit 8 of the StatusWord „b_BasePosition“ (BOOL)

When the gripper reaches its defined „BasePosition“, this signal is activated. The size of the area is defined with the „PositionTolerance“.

8.14 Bit 9 of the StatusWord „b_TeachPosition“ (BOOL)

When the gripper reaches its defined „TeachPosition“, this signal is activated. The size of the area is defined with the „PositionTolerance“.

8.15 Bit 10 of the StatusWord „b_WorkPosition“ (BOOL)

When the gripper reaches its defined „WorkPosition“, this signal is activated. The size of the area is defined with the „PositionTolerance“.

8.16 Bit 11 of the StatusWord „b_UndefinedPosition“ (BOOL)

When the gripper stands still and is not on „BasePosition“, „TeachPosition“ or „WorkPosition“, this signal is „TRUE“.

8.17 Bit 12 of the StatusWord „b_DataTransferOK“ (BOOL)

With this bit the gripper gives feedback that a data transfer („ControlWord“ = 1) was carried out successfully. That’s why it is used at a handshake procedure.

8.18 Bit 13 of the StatusWord „b_ControlWord_100“ (BOOL)

This direction flag turns to „TRUE“ when der gripper got a „MoveToBase“ command. The gripper can’t execute a further „MoveToBase“ command in this state. The flag is set to „FALSE“ again when the gripper gets a „MoveToWork“ command or a reset is done with „cmd_b_ResetDirectionFlag“ (see 8.4).

8.19 Bit 14 of the StatusWord „b_ControlWord_200“ (BOOL)

This direction flag turns to „TRUE“ when der gripper got a „MoveToWork“ command. The gripper can’t execute a further „MoveToWork“ command in this state. The flag is set to „FALSE“ again when the gripper gets a „MoveToBase“ command or a reset is done with „cmd_b_ResetDirectionFlag“ (see 8.4).

8.20 Bit 15 of the StatusWord „b_Error“ (BOOL) and „n_Diagnose“ (WORD)

When the diagnoe value of the gripper is not 0, this bit is set. The error code is put out in the data word „n_Diagnose“. The descriptions to the error codes can be found in the operation manual.

8.21 n_ActualPosition (WORD)

This data word shows the actual position of the gripper fingers.