<Operationmanual_Basic Gripping_TC3_V1_21(EN).docx>
topic: <FB Basic Gripping>

version: <1>
status: <21>

History

Author	Reason for change/changes made	Release	Status	Date
Nock	Displaying parameter change with output bit Automatic reset of the direction flags	1	21	01.04.2020

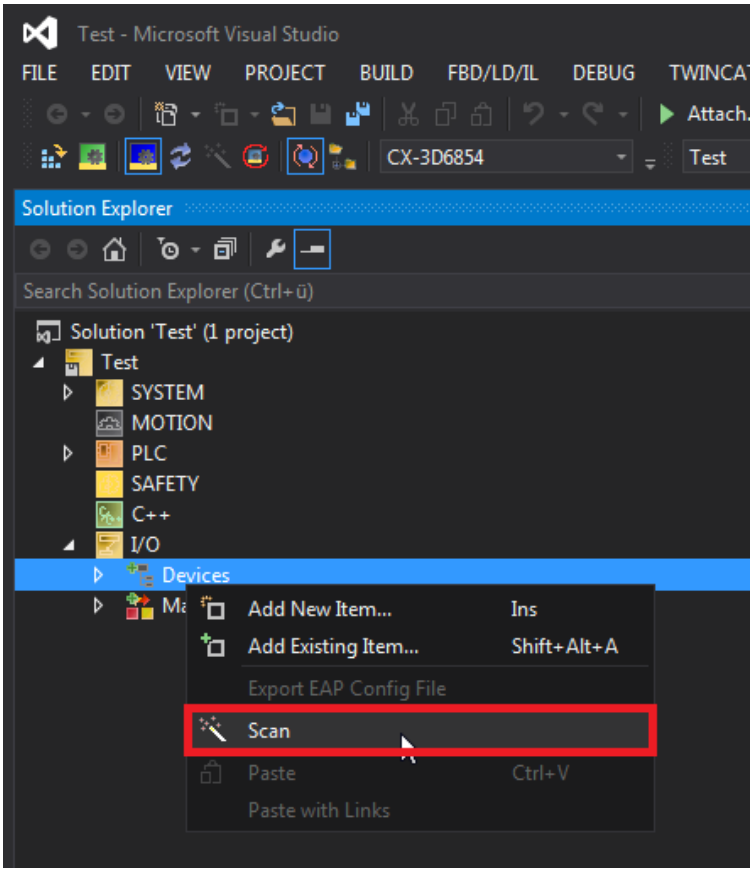
Content

1 Foreword

To use the sample program, a correct hardware configuration must first be created. In this example a Beckhoff CX5120-1 with a Beckhoff IO Link Master is used. After the hardware settings have been made, the sample project can be implemented. To do this, go through the following steps.

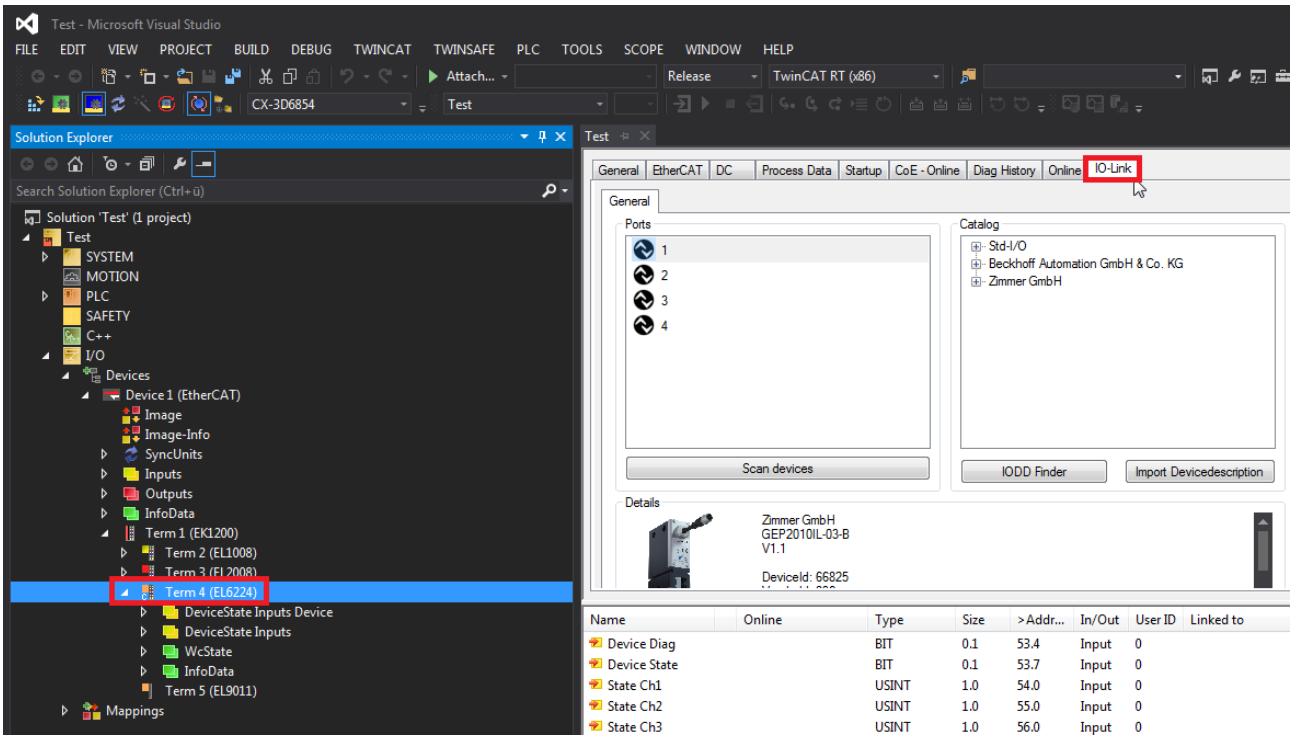
2 Creating a system configuration in the „Solution Explorer”

As soon as the connected controller has been set to "Config Mode", it is possible to search for devices.



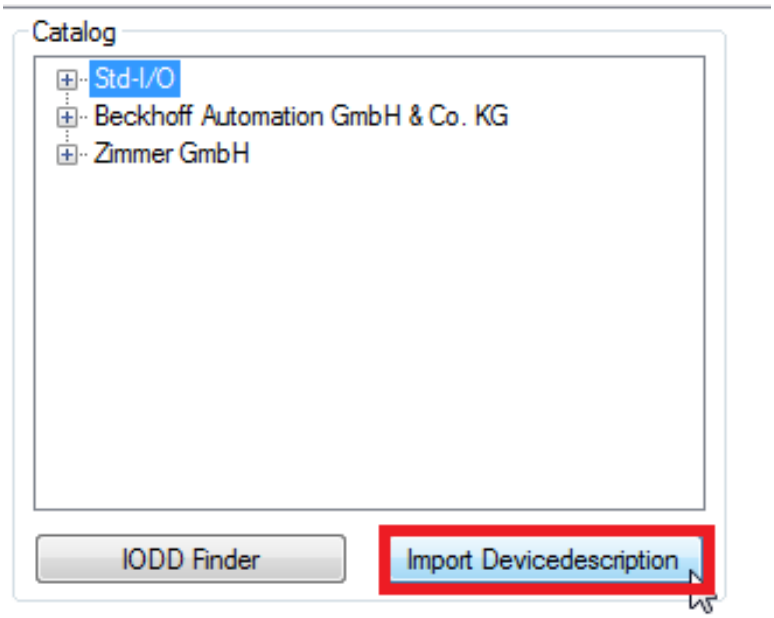
The connected devices should now be found and listed. In our example, the IO Link Master (terminal 4 / EL6224) is located on the first module under terminal 1.

The IO Link Master offers the function to identify the connected gripper. To do this, click on the "Scan devices" button in the "IO-Link" tab in the menu of the IO Link Master. A further window then opens, in which it is listed to which port which gripper has been connected.

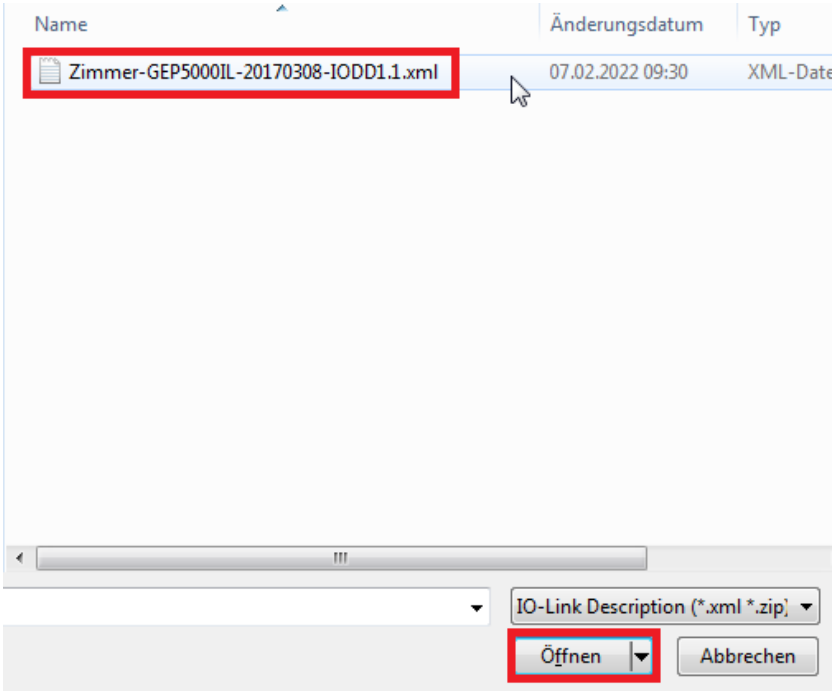


If the correct gripper was not recognized or you want to carry out the system configuration without a gripper connected, the gripper can also be inserted manually. For this purpose, the "Catalog" on the right-hand side is available, in which various third-party devices, including the IO Link gripper, are listed.

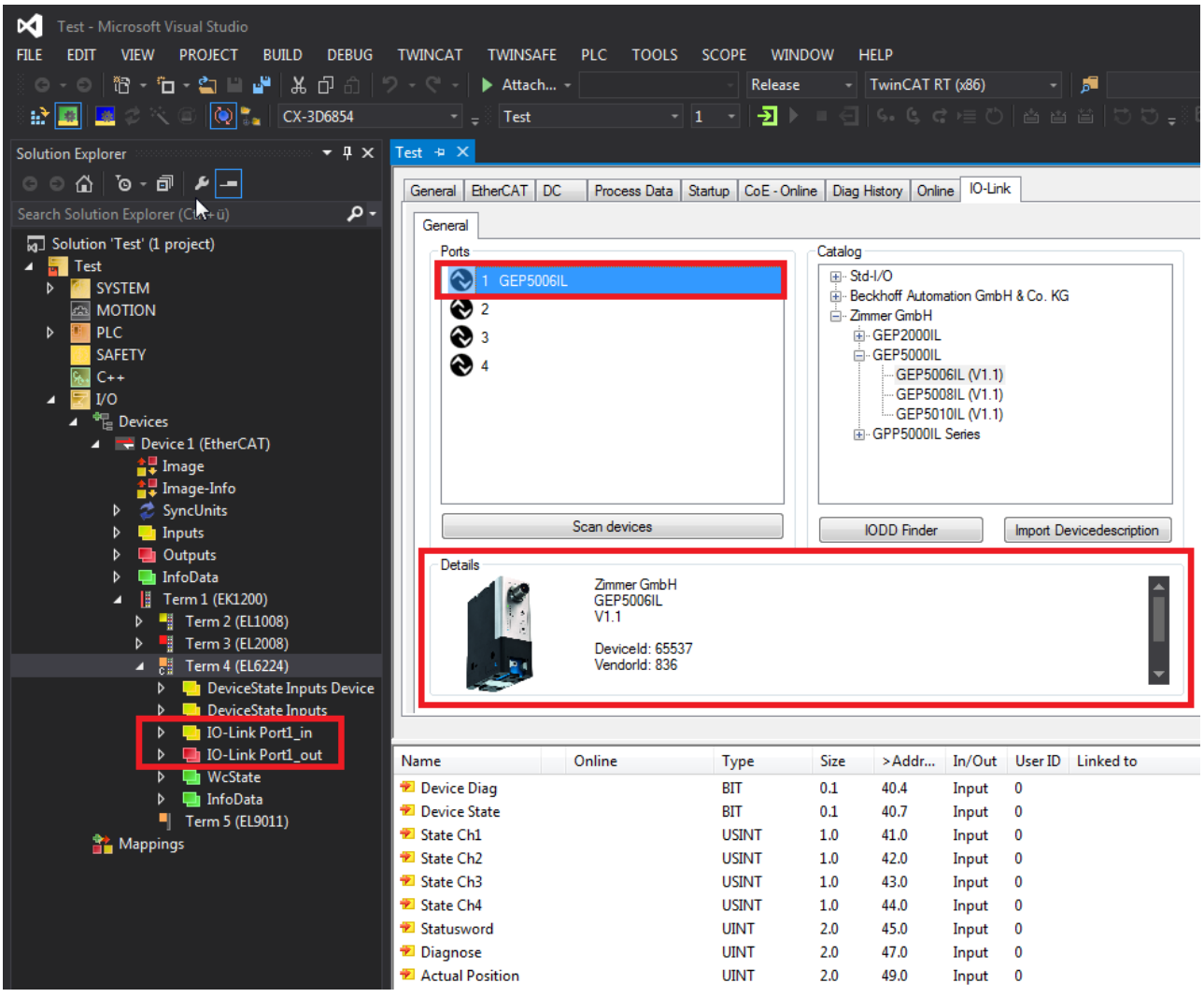
If the required gripper is not in the listing, it can be inserted by clicking the "Import Device Description" button.



All IODD files for our grippers can be downloaded from our homepage. After the download the desired IODD can be selected.



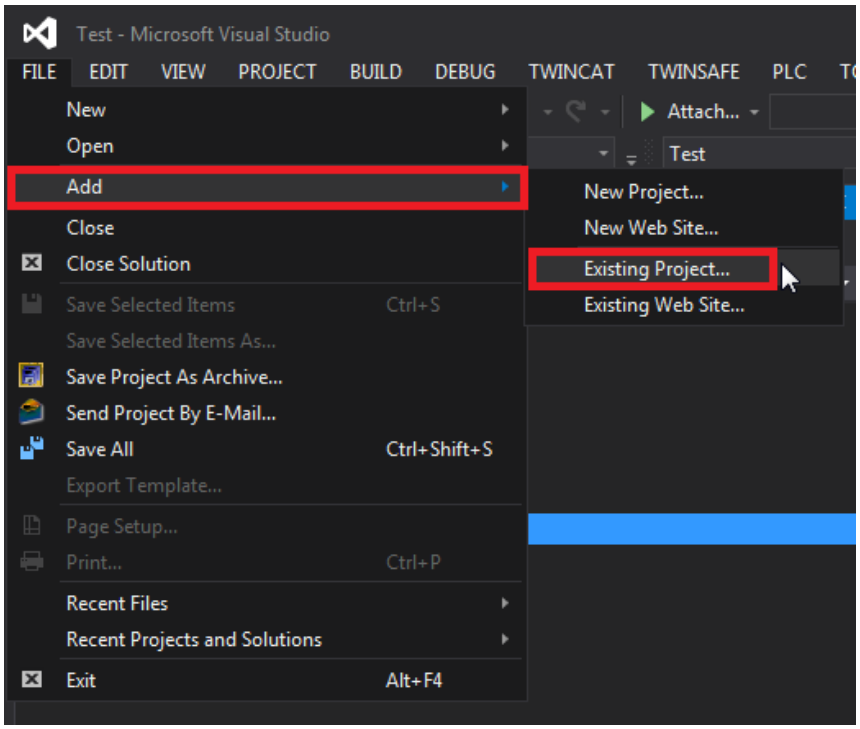
Once the desired gripper has been inserted on the correct port, the gripper type is displayed under the configured port.



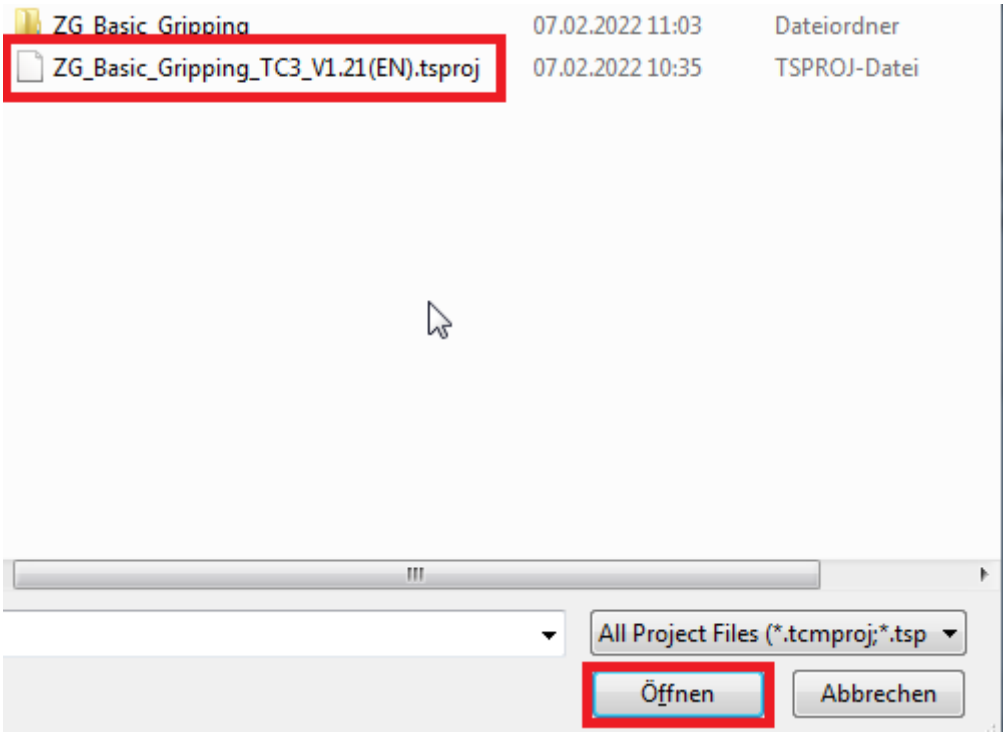
The inputs and outputs ("IO-Link Port1_in" and "IO-Link Port1_out") of the gripper are now displayed in the parameter tree and can be linked to the variables of the program after including a PLC configuration.

3 Copy and include PLC objects

To be able to use Zimmer's sample objects in your project, you must first load our sample project into your project. To do this, you must click on the "Existing Project..." function under the "File" tab in the "Add" menu.

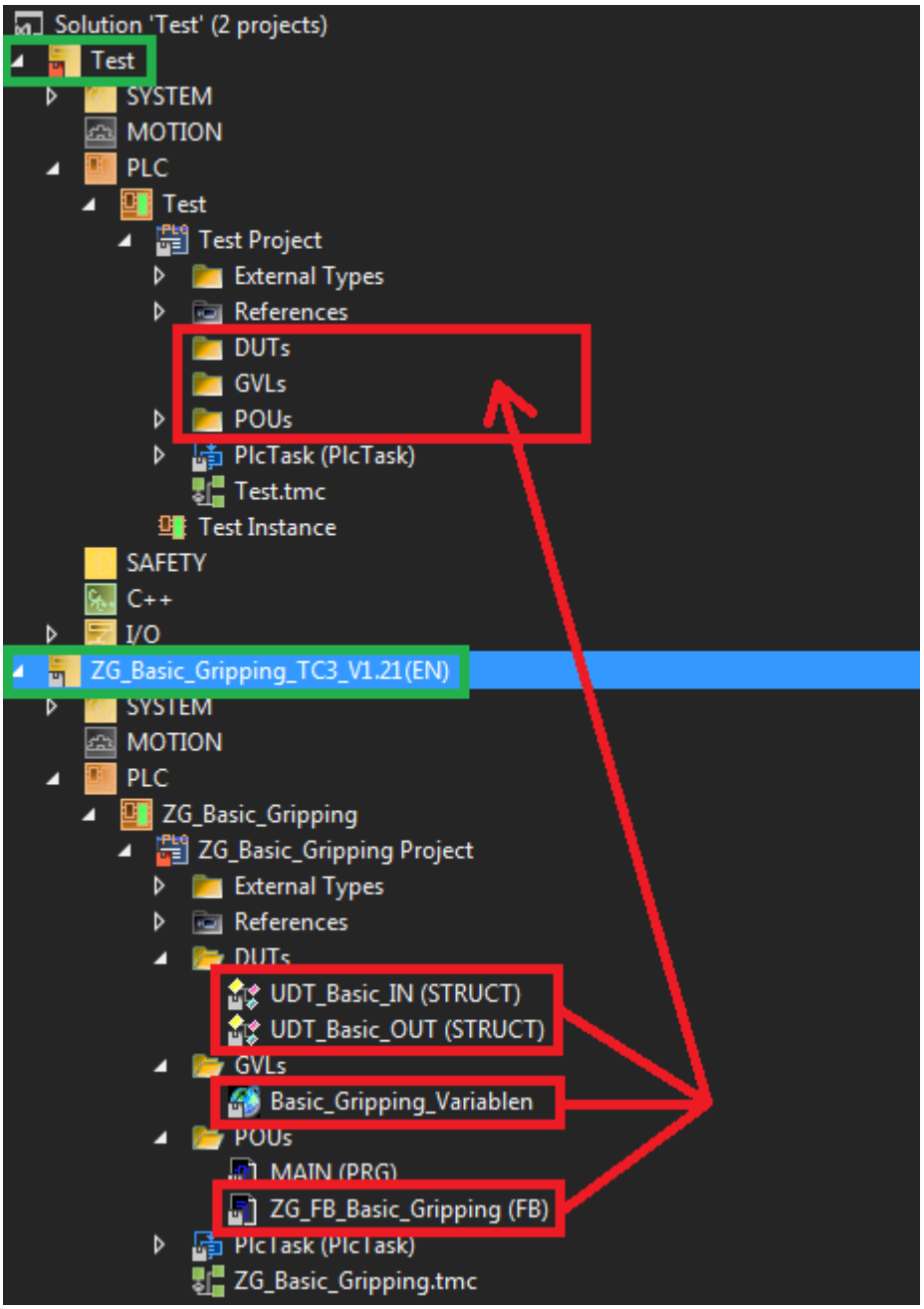


In the window that opens, you must now select our sample project.



The sample project now appears in the project tree under your project. By dragging and dropping, the desired objects can be copied into the corresponding folder of your project.

The "ZG_FB_Basic_Gripping" function block contains the control sequences that the gripper requires to execute commands. The data types "UDT_Basic_IN" and "UDT_Basic_OUT" as well as the global variable file "Basic_Gripping_Variables" are required for the communication interface between hardware and module. If your project and the main program are still empty, you can optionally also copy the "MAIN" program object into your project. This already contains the function block call of the "ZG_FB_Basic_Gripping" including the wiring of the inputs and outputs.



The data types "UDT_Basic_IN" and "UDT_Basic_OUT" are an exact image of the input and output parameters of the gripper. The image of the process parameters can also be taken from the operating manual of the gripper. The sequence and the variable size must be identical to the specification from the operating instructions of the gripper. If the data types are changed manually, proper operation of the function block would no longer be possible.

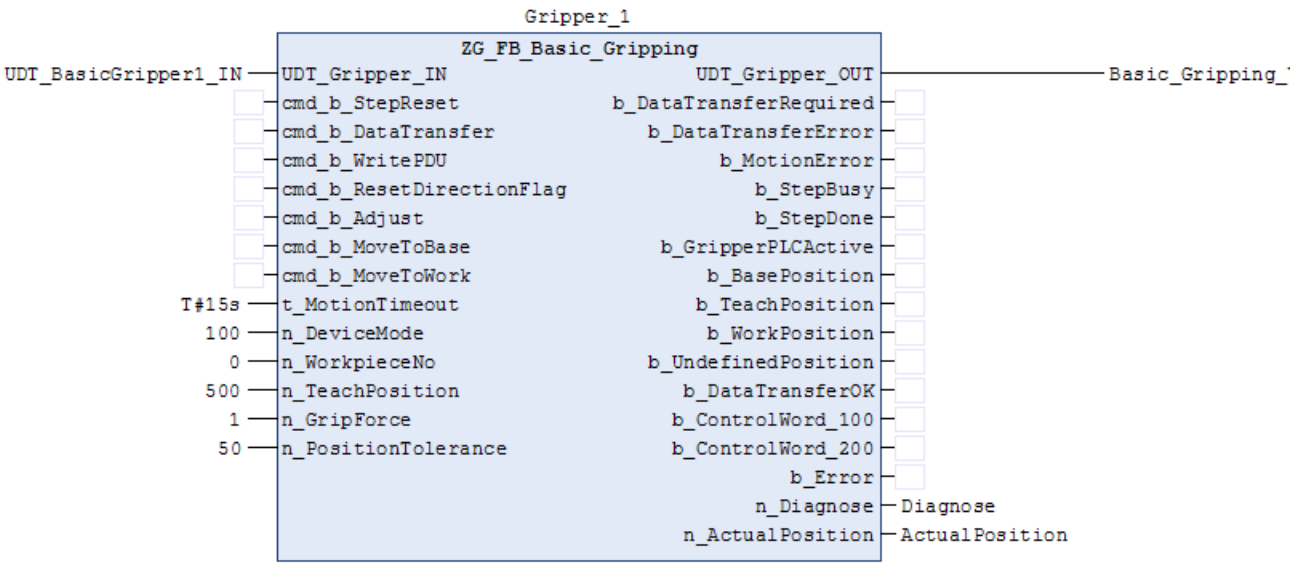
```
UDT_Basic_IN
1  TYPE UDT_Basic_IN :
2  STRUCT
3      n_StatusWord         : WORD;
4      n_Diagnose           : WORD;
5      n_ActualPosition     : WORD;
6  END_STRUCT
7  END_TYPE
```

```
UDT_Basic_OUT
1  TYPE UDT_Basic_OUT :
2  STRUCT
3      n_ControlWord        : WORD;
4      n_DeviceMode         : BYTE;
5      n_WorkpieceNo        : BYTE;
6      n_TeachPosition      : WORD;
7      n_GripForce          : BYTE;
8      n_PositionTolerance  : BYTE;
9  END_STRUCT
10 END_TYPE
```

In the enclosed global variable file "Basic_Gripping_Variables" the variable "UDT_BasicGripper1_OUT" is declared with the data type "UDT_Basic_OUT" and the variable "UDT_BasicGripper1_IN" with the data type "UDT_Basic_IN". These variables are intended for connection to the function block as a communication interface.

```
Basic_Gripping_Variablen
1  {attribute 'qualified_only'}
2  VAR_GLOBAL
3      UDT_BasicGripper1_OUT AT %Q* : UDT_Basic_OUT;
4      UDT_BasicGripper1_IN AT %I*  : UDT_Basic_IN;
5  END_VAR
```

In the enclosed program file "MAIN" the function block has been inserted in the programming language FBD for easier readability.



Variables declared with "b_" represent binary signals.

Variables declared with "cmd_" represent command inputs. These can be controlled with a key, for example.

Variables declared with "n_" are input or output bytes or words. These are needed for the transmission of the individual positions and functions.

4 Using the function block

Several inputs and outputs are now to be connected to the function block inserted under point 3. The "UDT_Gripper_IN" input on the function block must now be connected to the associated variable from point 3. The same must be done with the output variable "UDT_Gripper_OUT". Now the function block can read out the individual states and positions of the gripper and process them in the function block. Furthermore, the gripper can be parameterized by the output circuit.

In order for the gripper to be driven, the various position data and options (drive profiles) must be transferred to it. The values listed in the following table can be used as default values. These are exemplary and can vary depending on the project. You can enter these parameters as constants on the module, as in this example, or you can use variables of the appropriate length so that the circuitry is flexible. When not connected, the variables are pre-initialized with the default 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 travel profile of the gripper. These travel profiles can be found in the mounting instructions of the gripper. In this example, DeviceMode 100 (for GEP2000IL or GPP5000IL) or 1 (for GEP/GED5000IL) was selected, which corresponds to the "universal drive" travel profile. This can be used as default value.

The finished module should now correspond to the above figure. Finally, the settings must be transferred to the PLC. To do this, go through the steps necessary for Beckhoff:

- Translate
- Log in

5 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.

5.1 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.

5.2 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.

5.3 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.

5.4 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.

5.5 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.

5.6 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.

5.7 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“.

5.8 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.

5.9 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“.

5.10 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.

5.11 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“.

5.12 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.

5.13 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“.

5.14 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“.

5.15 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“.

5.16 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“.

5.17 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.

5.18 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 5.4).

5.19 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 5.4).

5.20 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.

5.21 n_ActualPosition (WORD)

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