<Operationmanual_Basic Gripping_TC2_V1_21(EN).docx>
topic: <FB Basic Gripping>

version: <1>
status: <21>

# History

| Author | Reason for change/changes made | Release | Status | Date |
|---|---|---|---|---|
| Armbruster | Basic version | 1 | 00 | 07.03.2018 |
| Nock | Displaying parameter change with output bit<br>Automatic reset of the direction flags | 1 | 21 | 01.04.2020 |

# Content

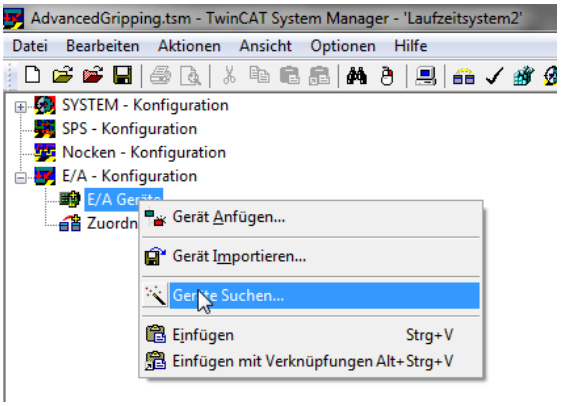
1 Foreword ..... 4
2 Creating of a system configuration with the system manager ..... 4
3 PLC Control ..... 6
4 Using the function block ..... 8
5 Functions of the function block ..... 9
5.1 Resetting the step sequence „cmd_b_StepReset“ (BOOL) ..... 9
5.2 Transfering data with handshake „cmd_b_DataTransfer“ (BOOL) ..... 9
5.3 Saving workpiece recipes „cmd_b_WritePDU“ (BOOL) ..... 9
5.4 Resetting the direction flags „cmd_b_ResetDirectionFlag“ (BOOL) ..... 9
5.5 Drive to BasePosition „cmd_b_MoveToBase“ (BOOL) ..... 9
5.6 Drive to WorkPosition „cmd_b_MoveToWork“ (BOOL) ..... 10
5.7 Limitting of the motion time „t_MotionTimeout“ (TIME) and „b_MotionError“ (BOOL) ..... 10
5.8 Data transfer is required „b_DataTransferRequired“ (BOOL) ..... 10
5.9 Error in the DataTransfer „b_DataTransferError“ (BOOL) ..... 10
5.10 Function block is busy „b_StepBusy“ (BOOL) ..... 10
5.11 Ready for commands „b_StepDone“ (BOOL) ..... 10
5.12 Bit 6 of the StatusWord „b_GripperPLCActive“ (BOOL) ..... 10
5.13 Bit 8 of the StatusWord „b_BasePosition“ (BOOL) ..... 10
5.14 Bit 9 of the StatusWord „b_TeachPosition“ (BOOL) ..... 10
5.15 Bit 10 of the StatusWord „b_WorkPosition“ (BOOL) ..... 10
5.16 Bit 11 of the StatusWord „b_UndefinedPosition“ (BOOL) ..... 11
5.17 Bit 12 of the StatusWord „b_DataTransferOK“ (BOOL) ..... 11
5.18 Bit 13 of the StatusWord „b_ControlWord_100“ (BOOL) ..... 11
5.19 Bit 14 of the StatusWord „b_ControlWord_200“ (BOOL) ..... 11
5.20 Bit 15 of the StatusWord „b_Error“ (BOOL) and „n_Diagnose“ (UINT) ..... 11
5.21 n_ActualPosition (UINT) ..... 11

# 1 Foreword

For using the example program at first a correct hardware configuration must be done. In this example a Beckhoff CX9020 with a Beckhoff IO Link master are used. After the hardware settings the example project can be implemented. Please pass the following steps for that.

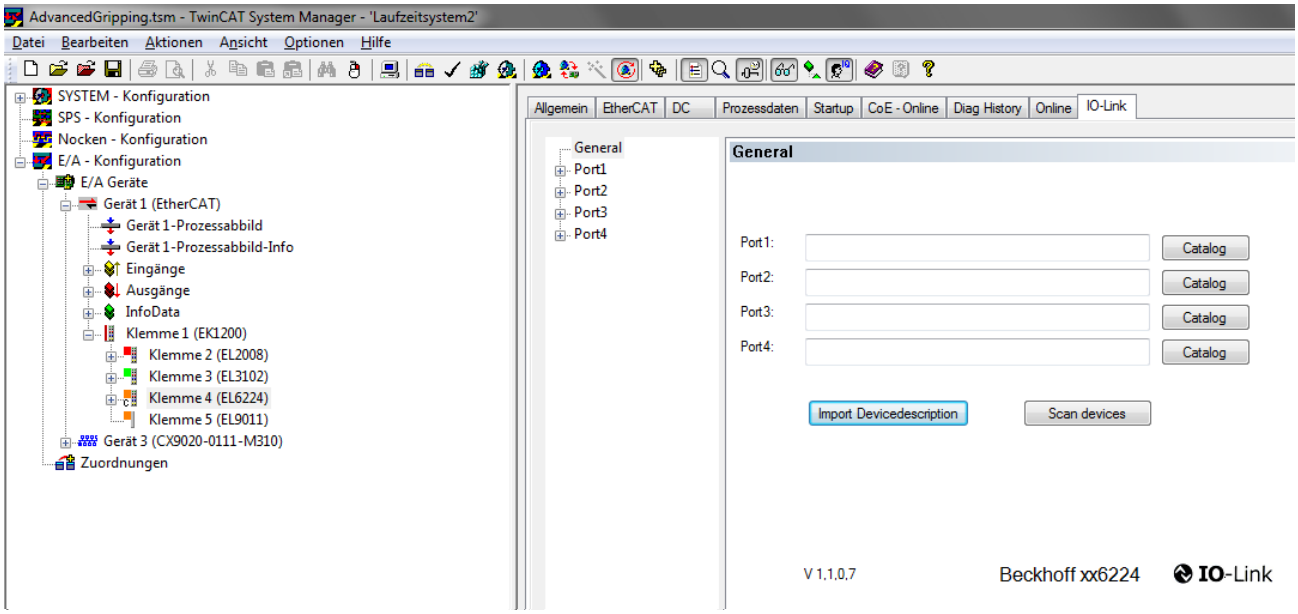
# 2 Creating of a system configuration with the system manager

When the connected PLC is turned to the ConfigMode, you can look for devices.



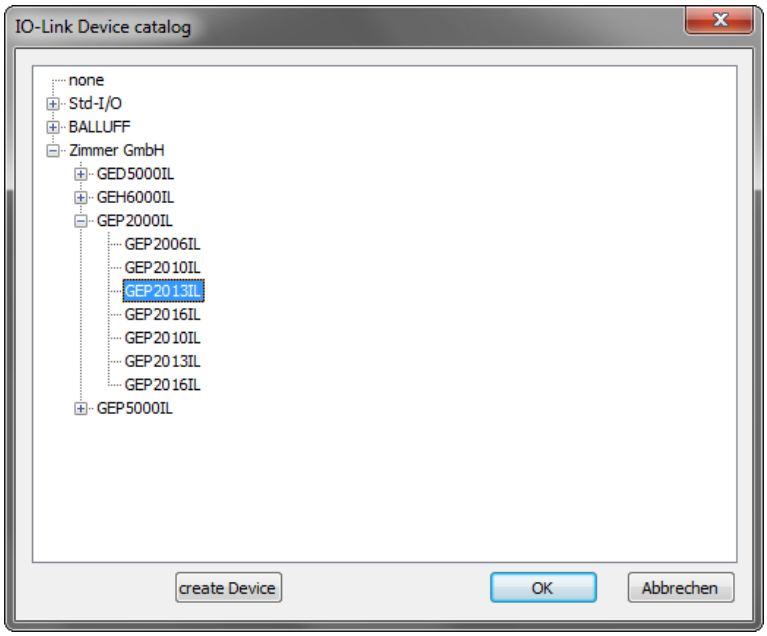

Now the connected devices should be found and listed. In our example the IO Link master (conductor 4 / EL6224) is located on the first module under the conductor 1.

The IO Link master offers the function to identify the gripper. For that you must click on „Scan devices“ in the equestrian “IO Link” in the menu of the IO Link master. After that a further window is opened which shows a list on which port which gripper is connected.



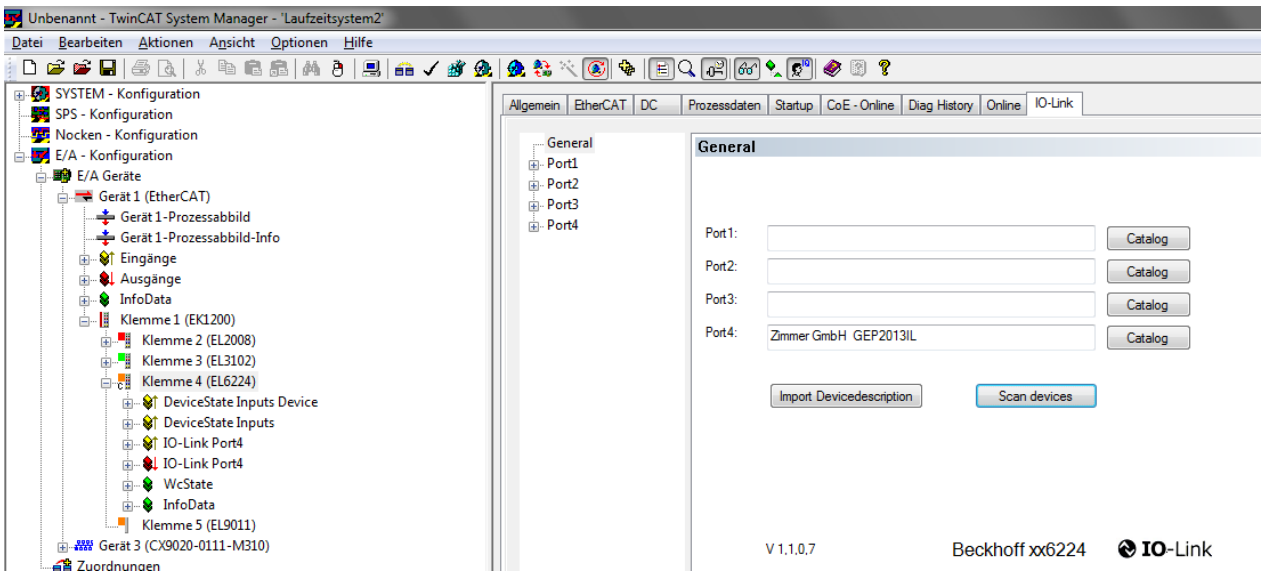

If not the correct gripper was identified or you would like to do the system configuration without the connected gripper, the gripper also can be inserted manually. For that please click on the button „Catalog“ next to the requiered port.

Now the „IO-Link Device catalog“ is opened where the several grippers are listed.



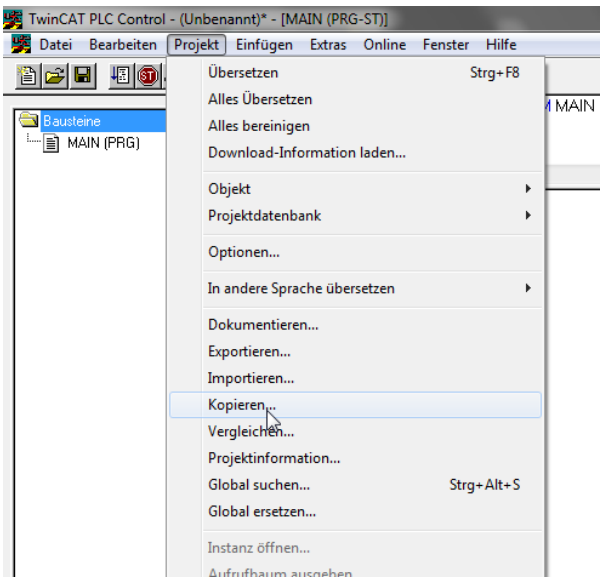

If the required gripper is not in the list, it can be inserted with a click on the button „Import Devicedescription“. All IODD data for our grippers can be downloaded from our hompage.

When the required gripper was inserted on the correct port, the gripper type is shown at the configurated port.



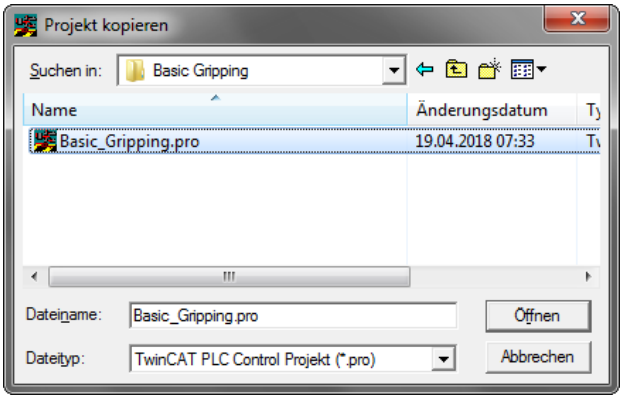

In the parameter tree the inputs and outputs can be shown. After the integration of a PLC configuration they can be linked with the variables of the program.

# 3 PLC Control

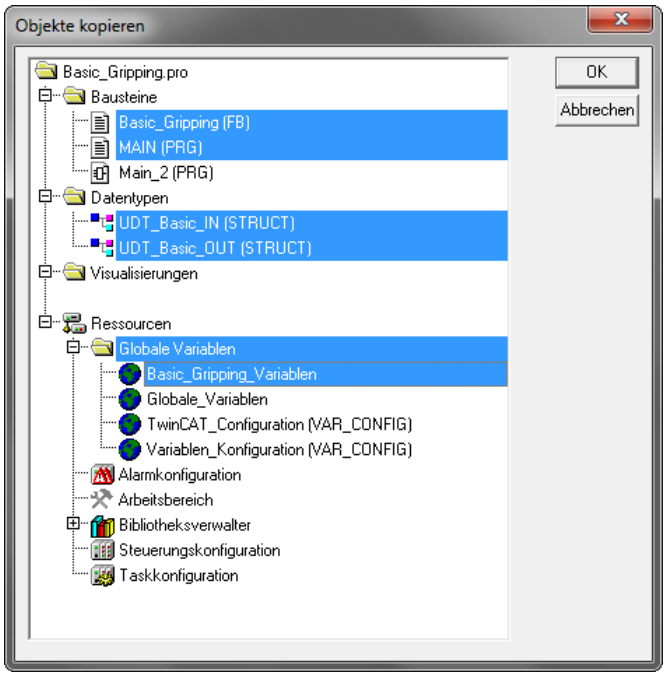



We created an example program to make the controlling of a gripper easier. In this example project is shown how a gripper can be controlled and how the several signals must be handled.

You have to copy the required blocks from our example project for using them in your application. For that you must select in your project the command „Copy …“ in the menu „Project“.




In the opened window you must select now our example project.



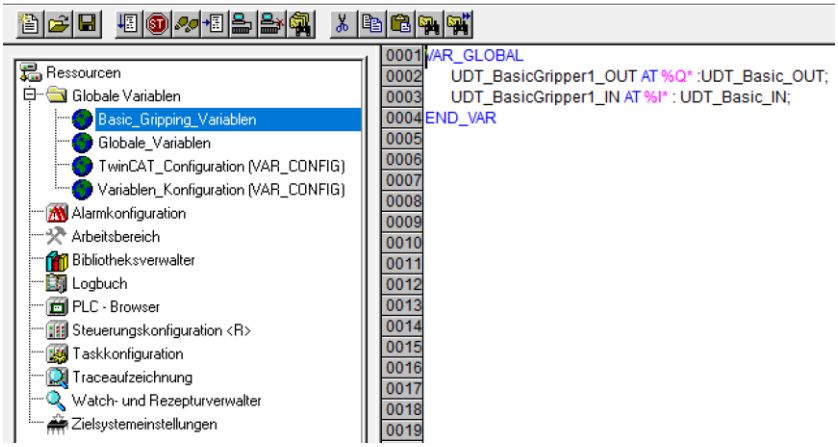

In the opened window „Copy objects“ the required components of the project can be copied. The function block „Basic Gripping“ includes the program sequence to control the gripper correctly. The data types „UDT_Basic_IN“ and „UDT_Basic_OUT“ and the global variable data „Basic_Gripping_Variablen“ are needed that the block works correctly. You can select all of them and insert them in your consisting project with the button „OK“. If you created a new empty project you also can copy the function block „MAIN“ to your project. In this block the program call for the function block „ZG_FB_Basic_Gripping“ is located.

```
UDT_Basic_OUT
0001 TYPE UDT_Basic_OUT :
0002 STRUCT
0003     n_ControlWord : WORD;
0004     n_DeviceMode : BYTE;
0005     n_WorkpieceNo : BYTE;
0006     n_TeachPosition :WORD;
0007     n_GripForce : BYTE;
0008     n_PositionTolerance : BYTE;
0009 END_STRUCT
0010 END_TYPE
```

```
UDT_Basic_IN
0001 TYPE UDT_Basic_IN :
0002 STRUCT
0003     n_StatusWord : WORD;
0004     n_Diagnose: WORD;
0005     n_ActualPosition: WORD;
0006 END_STRUCT
0007 END_TYPE
```

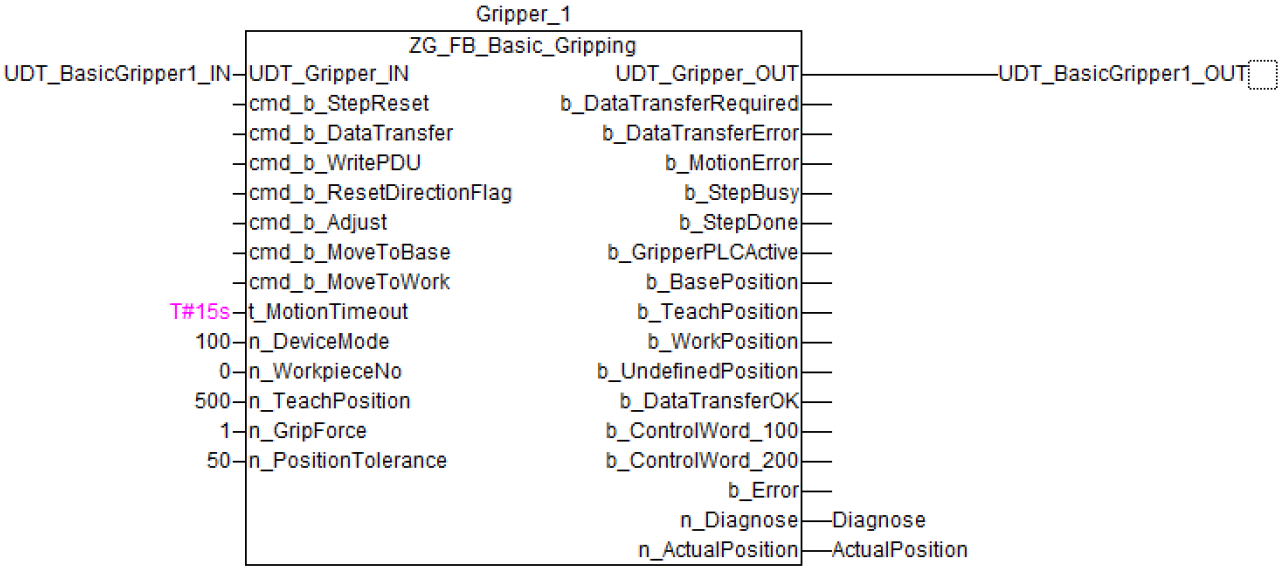
The definitions of the data types „UDT_Basic_OUT“ and „UDT_Basic_IN“ are an exact image of the input and output variables of the gripper which you want to control. The sequence and data size must be identical with the specification in the description of the gripper.

If they were changed, the data areas wouldn’t have been synchrone anymore and the gripper wouldn’t have been able to handle the data correctly. A properly controlling of the gripper wouldn’t be possible anymore.




In the added variable table the variable „UDT_BasicGripper1_OUT“ is assigned with the data type „UDT_Basic_OUT“ and the variable „UDT_BasicGripper1_IN“ with the data type „UDT_Basic_IN“. These both variables must be added on the function block.

For the simpler legibilty of the function block it was inserted in the programming language FUP.




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

Variables which are declared with „cmd_“ are command inputs. The 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.

## 4 Using the function block

On the inserted function block in chapter 3 are some inputs and outputs which must be wired. The input „UDT_Gripper_IN“ must be connected now with the belonging variable in chapter 3. 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 bei 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.

The completed block should now look like in the picture above. At last the settings must be transfered to the PLC. Please pass the steps which are necessary for Beckhoff:

- Compile
- 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“ (UINT)

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 (UINT)

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