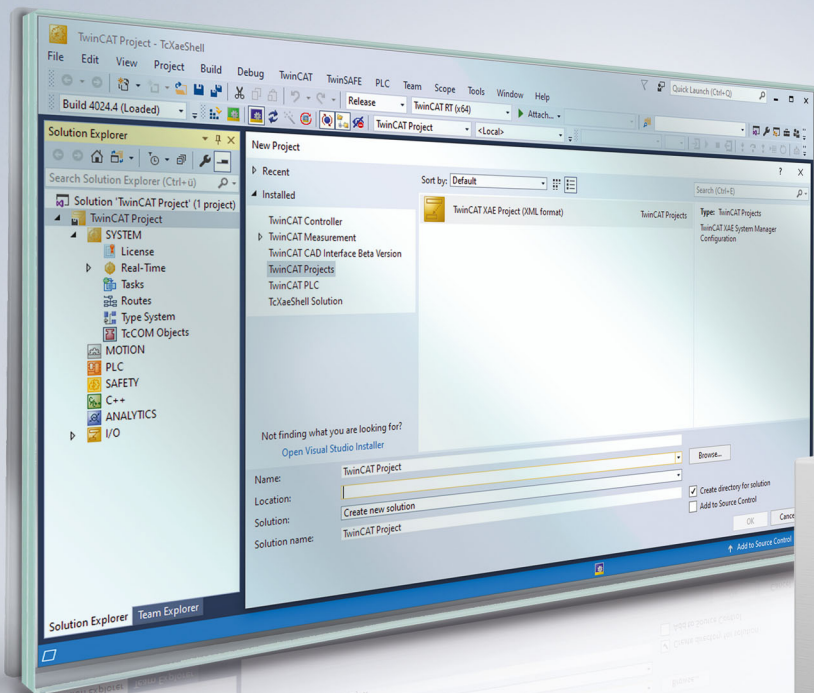


# BECKHOFF New Automation Technology

Manual | EN

# TF5130

TwinCAT 3 | Robotics uniVAL PLC





# Table of Contents

<b>1</b>	<b>Foreword</b> .....	<b>5</b>
1.1	Notes on the documentation .....	5
1.2	Safety instructions .....	6
1.3	Notes on information security.....	7
<b>2</b>	<b>Introduction</b> .....	<b>8</b>
<b>3</b>	<b>System Requirements</b> .....	<b>9</b>
<b>4</b>	<b>Licensing</b> .....	<b>10</b>
<b>5</b>	<b>PLC API</b> .....	<b>13</b>
5.1	Function blocks .....	13
5.1.1	Database.....	13
5.1.2	Internal .....	34
5.1.3	MC_GroupContinue .....	35
5.1.4	MC_GroupInterrupt .....	36
5.1.5	MC_GroupJog.....	37
5.1.6	MC_GroupPower .....	39
5.1.7	MC_GroupReadActualPosition .....	40
5.1.8	MC_GroupReset .....	41
5.1.9	MC_GroupStop .....	42
5.1.10	MC_MoveAxisAbsolute .....	43
5.1.11	MC_MoveCircularAbsolute .....	45
5.1.12	MC_MoveDirectAbsolute .....	47
5.1.13	MC_MoveLinearAbsolute.....	49
5.1.14	VAL_ComputeFrame .....	51
5.1.15	VAL_GetForwardTransformation .....	53
5.1.16	VAL_GetInverseTransformation.....	54
5.1.17	VAL_GetMotorTemperature.....	55
5.1.18	VAL_MoveSafeReference.....	57
5.1.19	VAL_ReadAxesGroup.....	58
5.1.20	VAL_ReadSafeReference.....	59
5.1.21	VAL_ReadWriteArmIO .....	60
5.1.22	VAL_ReadWriteIO.....	62
5.1.23	VAL_SafeBrakeTest.....	64
5.1.24	VAL_SetMovementID.....	65
5.1.25	VAL_ShiftPoint .....	66
5.1.26	VAL_TouchProbe.....	67
5.1.27	VAL_WriteAxesGroup .....	68
5.2	Functions.....	69
5.2.1	VAL_DefineCartesianPos .....	69
5.2.2	VAL_DefineJointPos .....	70
5.2.3	VAL_DefineTrsf.....	71
5.2.4	VAL_EnableManualMove.....	72
5.3	Types .....	73
5.3.1	Internal .....	73

5.3.2	MC_TransitionParameter .....	73
5.3.3	T_CS9SftyCmd .....	74
5.3.4	T_CS9SftyFbk.....	74
5.3.5	T_CartesianPos .....	75
5.3.6	T_Command .....	75
5.3.7	T_JointPos .....	76
5.3.8	T_Mdesc .....	76
5.3.9	T_StaeubliRobot .....	77
5.3.10	T_Status.....	78
5.3.11	T_Trsf.....	79
5.4	enums .....	80
5.4.1	eBlending .....	80
5.4.2	eConfigElbow .....	80
5.4.3	eConfigShoulder .....	81
5.4.4	eConfigWrist.....	81
5.4.5	eMC_BUFFER_MODE .....	82
5.4.6	eMC_TRANSITION_MODE .....	83
5.4.7	eOperationMode .....	83
5.4.8	eStateMachine .....	84
5.5	unival PLC Error Codes .....	85
5.5.1	CS8C_ErrorMessage .....	85
5.5.2	CS9_ErrorMessages.....	88
<b>6</b>	<b>Support and Service .....</b>	<b>98</b>

# 1 Foreword

## 1.1 Notes on the documentation

This description is only intended for the use of trained specialists in control and automation engineering who are familiar with applicable national standards.

It is essential that the documentation and the following notes and explanations are followed when installing and commissioning the components.

It is the duty of the technical personnel to use the documentation published at the respective time of each installation and commissioning.

The responsible staff must ensure that the application or use of the products described satisfy all the requirements for safety, including all the relevant laws, regulations, guidelines and standards.

### Disclaimer

The documentation has been prepared with care. The products described are, however, constantly under development.

We reserve the right to revise and change the documentation at any time and without prior announcement. No claims for the modification of products that have already been supplied may be made on the basis of the data, diagrams and descriptions in this documentation.

### Trademarks

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered trademarks of and licensed by Beckhoff Automation GmbH.

Other designations used in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owners.

### Patent Pending

The EtherCAT Technology is covered, including but not limited to the following patent applications and patents:

EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702

with corresponding applications or registrations in various other countries.



EtherCAT® is a registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany

### Copyright

© Beckhoff Automation GmbH & Co. KG, Germany.

The reproduction, distribution and utilization of this document as well as the communication of its contents to others without express authorization are prohibited.

Offenders will be held liable for the payment of damages. All rights reserved in the event of the grant of a patent, utility model or design.

## 1.2 Safety instructions

### Safety regulations

Please note the following safety instructions and explanations!  
Product-specific safety instructions can be found on following pages or in the areas mounting, wiring, commissioning etc.

### Exclusion of liability

All the components are supplied in particular hardware and software configurations appropriate for the application. Modifications to hardware or software configurations other than those described in the documentation are not permitted, and nullify the liability of Beckhoff Automation GmbH & Co. KG.

### Personnel qualification

This description is only intended for trained specialists in control, automation and drive engineering who are familiar with the applicable national standards.

### Description of symbols

In this documentation the following symbols are used with an accompanying safety instruction or note. The safety instructions must be read carefully and followed without fail!

#### **DANGER**

##### **Serious risk of injury!**

Failure to follow the safety instructions associated with this symbol directly endangers the life and health of persons.

#### **WARNING**

##### **Risk of injury!**

Failure to follow the safety instructions associated with this symbol endangers the life and health of persons.

#### **CAUTION**

##### **Personal injuries!**

Failure to follow the safety instructions associated with this symbol can lead to injuries to persons.

#### **NOTE**

##### **Damage to the environment or devices**

Failure to follow the instructions associated with this symbol can lead to damage to the environment or equipment.



##### **Tip or pointer**

This symbol indicates information that contributes to better understanding.

## 1.3 Notes on information security

The products of Beckhoff Automation GmbH & Co. KG (Beckhoff), insofar as they can be accessed online, are equipped with security functions that support the secure operation of plants, systems, machines and networks. Despite the security functions, the creation, implementation and constant updating of a holistic security concept for the operation are necessary to protect the respective plant, system, machine and networks against cyber threats. The products sold by Beckhoff are only part of the overall security concept. The customer is responsible for preventing unauthorized access by third parties to its equipment, systems, machines and networks. The latter should be connected to the corporate network or the Internet only if appropriate protective measures have been set up.

In addition, the recommendations from Beckhoff regarding appropriate protective measures should be observed. Further information regarding information security and industrial security can be found in our <https://www.beckhoff.com/secguide>.

Beckhoff products and solutions undergo continuous further development. This also applies to security functions. In light of this continuous further development, Beckhoff expressly recommends that the products are kept up to date at all times and that updates are installed for the products once they have been made available. Using outdated or unsupported product versions can increase the risk of cyber threats.

To stay informed about information security for Beckhoff products, subscribe to the RSS feed at <https://www.beckhoff.com/secinfo>.

## 2 Introduction

The TC3 Robotics uniVAL PLC allows direct communication between the PLC and the robotics controller from Stäubli via a common interface. The robot's movements can be programmed directly in the PLC and compared with the robot's actual values in real-time. The TC3 Robotics uniVAL PLC combines PLC control and robotics on a single platform and enables programming from a single system without having to know a special robot programming language.

The communication takes place via EtherCAT, with TwinCAT and the robotics controller from Stäubli exchanging the data as master and slave, respectively.

TwinCAT sends the motion commands to the robot via EtherCAT. Thanks to this efficient communication, commands can be sent from the PLC to the robot at high speed. In addition, the PLC programmer has real-time access to the robot's position data at all times. Other motion programs which are located in the robot controller's database can also be activated via this interface.



### 3 System Requirements

Technical data	TF5130
Required	TC1200 TwinCAT 3 PLC
Target operating system	Windows 7, Windows 10, Windows CE, TwinCAT/BSD
TwinCAT version	<b>3.1.4022</b> and later versions

TwinCAT library	TwinCAT version	uniVAL PLC server	Stäubli Robot Control (SRC)
Tc3_uniValPlc_v4_6	3.1.4024.43	s4.6	s7.10.2 (cs8) s8.10(cs9)
Tc3_uniValPlc_v4_5	3.1.4022.35 & 3.1.4024.15	s4.5	s7.10.2 (cs8) s8.10(cs9)
Tc3_uniValPlc_v4_4	3.1.4022.32 & 3.1.4024.9	s4.4	s7.10.2 (cs8) s8.12.3 (cs9)
Tc3_uniValPlc_v4_3	3.1.4022.30 & 3.1.4024.0	s4.2 & s4.3	s7.10.2 (cs8) s8.6.2 (cs9)
Tc3_uniValPlc_v4	3.1.4022.28	s4.1	s7.10.1 (cs8) s8.5.1 (cs9)
Tc3_uniValPlc	3.1.4022.28	3.1	s7.10.0 (cs8)

## 4 Licensing

The TwinCAT 3 function can be activated as a full version or as a 7-day test version. Both license types can be activated via the TwinCAT 3 development environment (XAE).

### Licensing the full version of a TwinCAT 3 Function

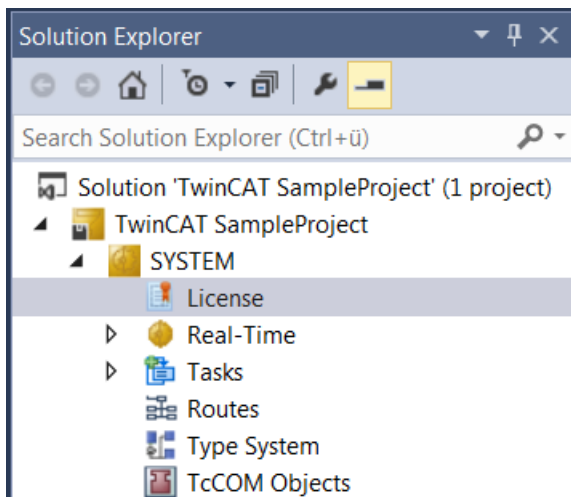
A description of the procedure to license a full version can be found in the Beckhoff Information System in the documentation "[TwinCAT 3 Licensing](#)".

### Licensing the 7-day test version of a TwinCAT 3 Function



A 7-day test version cannot be enabled for a [TwinCAT 3 license dongle](#).

1. Start the TwinCAT 3 development environment (XAE).
2. Open an existing TwinCAT 3 project or create a new project.
3. If you want to activate the license for a remote device, set the desired target system. To do this, select the target system from the **Choose Target System** drop-down list in the toolbar.
  - ⇒ The licensing settings always refer to the selected target system. When the project is activated on the target system, the corresponding TwinCAT 3 licenses are automatically copied to this system.
4. In the **Solution Explorer**, double-click **License** in the **SYSTEM** subtree.



⇒ The TwinCAT 3 license manager opens.

- Open the **Manage Licenses** tab. In the **Add License** column, check the check box for the license you want to add to your project (e.g. "TF4100 TC3 Controller Toolbox").

Order No	License	Add License
TF3601	TC3 Condition Monitoring Level 2	<input type="checkbox"/> cpu license
TF3650	TC3 Power Monitoring	<input type="checkbox"/> cpu license
TF3680	TC3 Filter	<input type="checkbox"/> cpu license
TF3800	TC3 Machine Learning Inference Engine	<input type="checkbox"/> cpu license
TF3810	TC3 Neural Network Inference Engine	<input type="checkbox"/> cpu license
TF3900	TC3 Solar-Position-Algorithm	<input type="checkbox"/> cpu license
TF4100	TC3 Controller Toolbox	<input checked="" type="checkbox"/> cpu license
TF4110	TC3 Temperature-Controller	<input type="checkbox"/> cpu license
TF4500	TC3 Speech	<input type="checkbox"/> cpu license

- Open the **Order Information (Runtime)** tab.
  - ⇒ In the tabular overview of licenses, the previously selected license is displayed with the status "missing".
- Click **7-Day Trial License...** to activate the 7-day trial license.

- ⇒ A dialog box opens, prompting you to enter the security code displayed in the dialog.

- Enter the code exactly as it is displayed and confirm the entry.
- Confirm the subsequent dialog, which indicates the successful activation.
  - ⇒ In the tabular overview of licenses, the license status now indicates the expiry date of the license.

10. Restart the TwinCAT system.

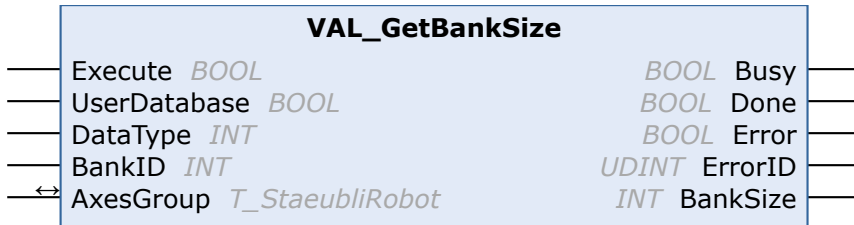
⇒ The 7-day trial version is enabled.

## 5 PLC API

### 5.1 Function blocks

#### 5.1.1 Database

##### 5.1.1.1 VAL\_GetBankSize



FUNCTION\_BLOCK VAL\_GetBankSize

This function block returns the size of the specified bank of data from the selected database.

#### Syntax

##### Definition:

```
FUNCTION_BLOCK VAL_GetBankSize
VAR_INPUT
    Execute      : BOOL := FALSE;
    UserDatabase : BOOL := FALSE;
    DataType     : INT  := 0;
    BankID       : INT  := -1;
END_VAR
VAR_OUTPUT
    Busy        : BOOL := FALSE;
    Done        : BOOL := FALSE;
    Error       : BOOL := FALSE;
    ErrorID     : UDINT := 0;
    BankSize    : INT  := 0;
END_VAR
VAR_IN_OUT
    AxesGroup   : T_StaeubliRobot;
END_VAR
```

#### Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
UserDatabase	BOOL	True=Use user database; False=Use plc database
DataType	INT	Type of data to resize (1 = point, 2 = joint, 3 = tool, 4 = motion descriptor)
BankID	INT	Number of the bank of point to resize. This parameter is effective only when DataType parameter = 1.

#### Outputs

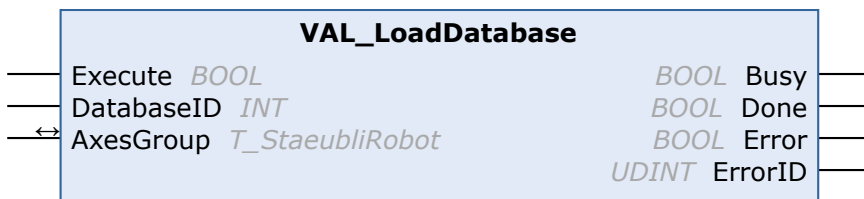
Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success

Name	Type	Description
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
BankSize	INT	Size of the selected bank of data

 **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaeubliRobot [ <a href="#">▶ 77</a> ]	Data block for a robot

### 5.1.1.2 VAL\_LoadDatabase



FUNCTION\_BLOCK VAL\_LoadDatabase

Load a database identified by the ID given as input parameter.

Parameter Name	Value	Comment
ErrorID	0	No error
	[110..142]	See VAL3 error code of libLoad instruction

#### Syntax

##### Definition:

```
FUNCTION_BLOCK VAL_LoadDatabase
VAR_INPUT
    Execute      : BOOL := FALSE;
    DatabaseID   : INT  := -1;
END_VAR
VAR_OUTPUT
    Busy        : BOOL := FALSE;
    Done        : BOOL := FALSE;
    Error       : BOOL := FALSE;
    ErrorID     : UDINT := 0;
END_VAR
VAR_IN_OUT
    AxesGroup   : T_StaeubliRobot;
END_VAR
```

 **Inputs**

Name	Type	Description
Execute	BOOL	Rising edge starts function execution.
DatabaseID	INT	Identifier of the database to be loaded.

 **Outputs**

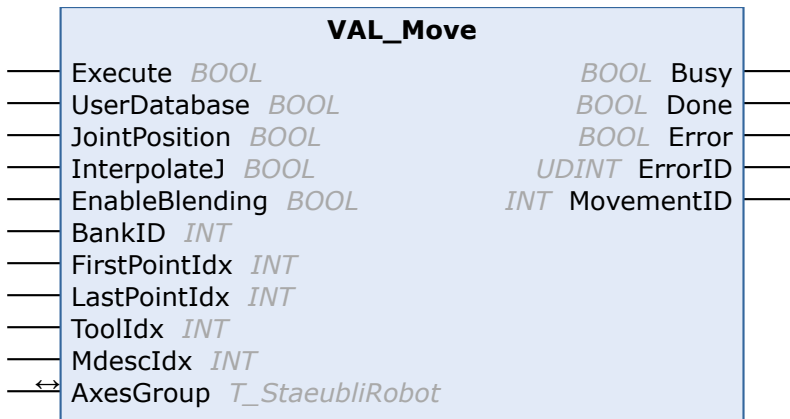
Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success

Name	Type	Description
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code

 Inputs/Outputs

Name	Type	Description
AxesGroup	T_StaebliRobot [▶ 77]	Data block for a robot

### 5.1.1.3 VAL\_Move



#### FUNCTION\_BLOCK VAL\_Move

This function block allows to command a motion to the specified end position located inside a database on robot side.

When FirstPointIdx and LastPointIdx parameters are different, the robot will move along the path defined by the points located in the bank between these 2 indexes. If LastPointIdx is not specified or equal to FirstPointIdx, the function block command a single movement to the point index defined by FirstPointIdx parameter.

Example: One supposes that there are 6 points defined in the bank number 2 of a database  
 If FirstPointIdx is equal 0 and LastPointIdx is equal 4, the robot will move along a path defined by the points with indexes 0-1-2-3-4  
 If FirstPointIdx is equal 5 and LastPointIdx is equal 0, the robot will move along a path defined by the points with indexes 5-4-3-2-1-0

#### Syntax

##### Definition:

```

FUNCTION_BLOCK VAL_Move
VAR_INPUT
    Execute          : BOOL := FALSE;
    UserDatabase     : BOOL := FALSE;
    JointPosition    : BOOL := FALSE;
    InterpolateJ     : BOOL := FALSE;
    EnableBlending   : BOOL := FALSE;
    BankID           : INT := -1;
    FirstPointIdx    : INT := -1;
    LastPointIdx     : INT := -1;
    ToolIdx          : INT := -1;
    MdescIdx         : INT := -1;
END_VAR
VAR_OUTPUT
    Busy            : BOOL := FALSE;
    Done            : BOOL := FALSE;
    Error           : BOOL := FALSE;
    ErrorID         : UDINT := 0;
    MovementID     : INT := -1;
    
```

```

END_VAR
VAR_IN_OUT
  AxesGroup : T_StaeubliRobot;
END_VAR

```

### Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
UserDatabase	BOOL	True=Use user database; False=Use plc database
JointPosition	BOOL	True=move to joint position; False= move to a cartesian position
InterpolateJ	BOOL	True=Joint interpolated movement; False=Linear movement
EnableBlending	BOOL	True=use blending parameter of motion descriptor; False= robot stops at destination.
BankID	INT	Number of the bank in which the point is located. Relevant only when moving on Cartesian position
FirstPointIdx	INT	Index of the first Joint or Point data in the selected bank
LastPointIdx	INT	Index of the last Joint or Point data in the selected bank.
ToolIdx	INT	Index of the tool position in the bank
MdescIdx	INT	Index of the motion descriptor in the bank.

### Outputs

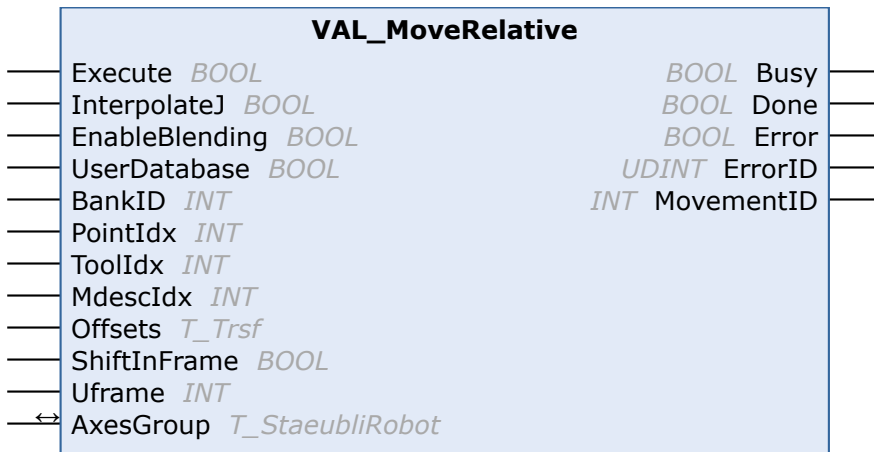
Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
MovementID	INT	Identifier for this commanded movement

### / Inputs/Outputs

Name	Type	Description
AxesGroup	T_StaeubliRobot [ <a href="#">▶ 77</a> ]	Data block for a robot



### 5.1.1.4 VAL\_MoveRelative



#### FUNCTION\_BLOCK VAL\_MoveRelative

This function block allows to commands a movement to a position calculated with specified offsets from a point located in a bank of a database.

The offset might be applied in the following coordinate system:

- Tool
- User Frame

#### Syntax

##### Definition:

```
FUNCTION_BLOCK VAL_MoveRelative
VAR_INPUT
    Execute           : BOOL := FALSE;
    InterpolateJ      : BOOL := FALSE;
    EnableBlending    : BOOL := FALSE;
    UserDatabase      : BOOL := FALSE;
    BankID            : INT := -1;
    PointIdx          : INT := -1;
    ToolIdx           : INT := -1;
    MdescIdx          : INT := -1;
    Offsets           : T_Trsf;
    ShiftInFrame      : BOOL := FALSE;
    Uframe            : INT := -1;
END_VAR
VAR_OUTPUT
    Busy              : BOOL := FALSE;
    Done              : BOOL := FALSE;
    Error             : BOOL := FALSE;
    ErrorID           : UDINT := 0;
    MovementID       : INT := -1;
END_VAR
VAR_IN_OUT
    AxesGroup        : T_StaebliRobot;
END_VAR
```

#### 🔴 Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
InterpolateJ	BOOL	True=Joint interpolated movement; False=Linear movement
EnableBlending	BOOL	True=use blending parameter of motion descriptor; False= robot stops at destination.
UserDatabase	BOOL	True=Use user database; False=Use plc database
BankID	INT	Number of the bank in which the point is located

Name	Type	Description
PointIdx	INT	Index of the Point data in the bank
ToolIdx	INT	Index of the Tool in the bank to use for the commanded movement
MdescIdx	INT	Index of the motion descriptor in the bank.
Offsets	T_Trspf [▶ 79]	Relative distances in each dimensions for approaching the end-position.
ShiftInFrame	BOOL	True= Offset are applied in user frame. False= Offset are applied in Tool (default)
Uframe	INT	Index of the user frame in which offsets are applied. This parameter is effective when parameter ShiftInFrame is True. If not specified, the offsets are applied in the user frame associated to the specified bankID

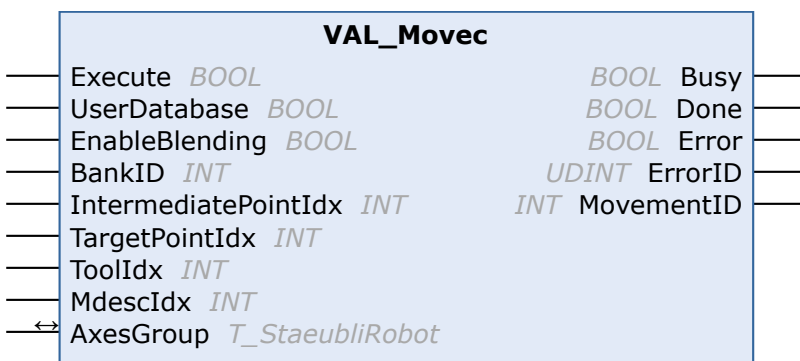
 **Outputs**

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
MovementID	INT	Identifier for this commanded movement

 /  **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaeubliRobot [▶ 77]	Data block for a robot

**5.1.1.5 VAL\_Movec**



FUNCTION\_BLOCK VAL\_Movec

This function commands a circular interpolated movement to a point located inside a database on robot side.

**Syntax**

**Definition:**

```
FUNCTION_BLOCK VAL_Movec
VAR_INPUT
    Execute          : BOOL := FALSE;
    UserDatabase     : BOOL := FALSE;
    EnableBlending   : BOOL := FALSE;
```

```

    BankID      : INT := -1;
    IntermediatePointIdx : INT := -1;
    TargetPointIdx : INT := -1;
    ToolIdx     : INT := -1;
    MdescIdx    : INT := -1;
END_VAR
VAR_OUTPUT
    Busy      : BOOL := FALSE;
    Done      : BOOL := FALSE;
    Error     : BOOL := FALSE;
    ErrorID   : UDINT := 0;
    MovementID : INT := -1;
END_VAR
VAR_IN_OUT
    AxesGroup : T_StaeubliRobot;
END_VAR

```

 **Inputs**

Name	Type	Description
Execute	BOOL	// Rising edge starts function execution
UserDatabase	BOOL	True=Data of the user database will be used
EnableBlending	BOOL	True=use blending parameter of motion descriptor; False= robot stops at destination.
BankID	INT	Number of the bank in which the point is located. Relevant only when moving on Cartesian position
IntermediatePointIdx	INT	Intermediate point passing during the movement
TargetPointIdx	INT	Target point of the circle
ToolIdx	INT	Index of the tool position in the bank
MdescIdx	INT	Index of the motion descriptor in the bank.

 **Outputs**

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
MovementID	INT	Identifier for this commanded movement

 **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaeubliRobot <a href="#">▶ 77</a>	Data block for a robot

**5.1.1.6 VAL\_Position**



## FUNCTION\_BLOCK VAL\_Position

This function block returns the coordinates of the position in the user frame associated with the destination BankID

**Syntax****Definition:**

```
FUNCTION_BLOCK VAL_Position
VAR_INPUT
    Execute      : BOOL := FALSE;
    UserDatabase : BOOL := FALSE;
    BankID       : INT := -1;
    PointIdx     : INT := -1;
    BankIDDest   : INT := -1;
END_VAR
VAR_OUTPUT
    Busy         : BOOL := FALSE;
    Done         : BOOL := FALSE;
    Error        : BOOL := FALSE;
    ErrorID      : UDINT := 0;
    PositionDest : T_CartesianPos;
END_VAR
VAR_IN_OUT
    AxesGroup    : T_StaeubliRobot;
END_VAR
```

 **Inputs**

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
UserDatabase	BOOL	True=Data of the user database will be used
BankID	INT	Number of the bank in which the point is located.
PointIdx	INT	Index of the point in the bank
BankIDDest	INT	Number of the bank in which the coordinates of the position are calculated

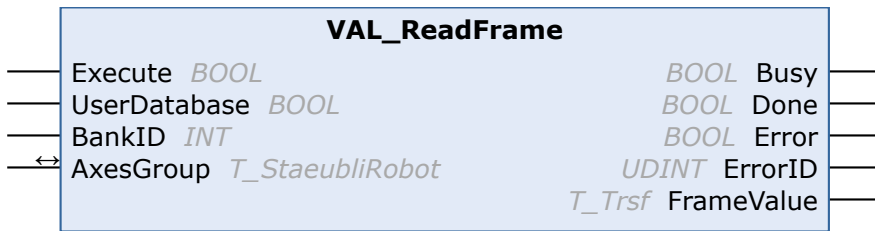
 **Outputs**

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
PositionDest	T_CartesianPos [ <a href="#">▶ 75</a> ]	New Position in the destination coordsystem

 **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaeubliRobot [ <a href="#">▶ 77</a> ]	Data block for a robot

### 5.1.1.7 VAL\_ReadFrame



#### FUNCTION\_BLOCK VAL\_ReadFrame

This function block returns the value of the user frame associated to a bank of point.

#### Syntax

##### Definition:

```
FUNCTION_BLOCK VAL_ReadFrame
VAR_INPUT
    Execute      : BOOL := FALSE;
    UserDatabase : BOOL := FALSE;
    BankID       : INT  := -1;
END_VAR
VAR_OUTPUT
    Busy        : BOOL := FALSE;
    Done        : BOOL := FALSE;
    Error       : BOOL := FALSE;
    ErrorID     : UDINT := 0;
    FrameValue  : T_Trssf;
END_VAR
VAR_IN_OUT
    AxesGroup   : T_StaeubliRobot;
END_VAR
```

#### Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
UserDatabase	BOOL	True= Data of the user database will be used
BankID	INT	Number of the bank for which coordinate system (frame) is returned.

#### Outputs

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
FrameValue	<a href="#">T_Trssf [▶ 79]</a>	Coordinates of the requested user frame

#### Inputs/Outputs

Name	Type	Description
AxesGroup	<a href="#">T_StaeubliRobot [▶ 77]</a>	Data block for a robot

### 5.1.1.8 VAL\_ReadJoint



FUNCTION\_BLOCK VAL\_ReadJoint

This function block returns the coordinates of a joint position in the selected database.

#### Syntax

##### Definition:

```
FUNCTION_BLOCK VAL_ReadJoint
VAR_INPUT
    Execute      : BOOL := FALSE;
    UserDatabase : BOOL := FALSE;
    JointIdx     : INT  := -1;
END_VAR
VAR_OUTPUT
    Busy       : BOOL := FALSE;
    Done       : BOOL := FALSE;
    Error      : BOOL := FALSE;
    ErrorID    : UDINT := 0;
    JointValue : T_JointPos;
END_VAR
VAR_IN_OUT
    AxesGroup : T_StaeubliRobot;
END_VAR
```

#### Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
UserDatabase	BOOL	True=Data of the user database will be used
JointIdx	INT	Index of the joint variable in the bank

#### Outputs

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
JointValue	T_JointPos [▶ 76]	Coordinates of joint position

#### Inputs/Outputs

Name	Type	Description
AxesGroup	T_StaeubliRobot [▶ 77]	Data block for a robot

### 5.1.1.9 VAL\_ReadMdesc



FUNCTION\_BLOCK VAL\_ReadMdesc

This function returns the current value of a motion descriptor in the selected database.

#### Syntax

##### Definition:

```
FUNCTION_BLOCK VAL_ReadMdesc
VAR_INPUT
    Execute      : BOOL := FALSE;
    UserDatabase : BOOL := FALSE;
    MdescIdx     : INT  := -1;
END_VAR
VAR_OUTPUT
    Busy       : BOOL := FALSE;
    Done       : BOOL := FALSE;
    Error      : BOOL := FALSE;
    ErrorID    : UDINT := 0;
    MdescValue : T_Mdesc;
END_VAR
VAR_IN_OUT
    AxesGroup : T_StaeubliRobot;
END_VAR
```

#### Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
UserDatabase	BOOL	True=Data of the user database will be used
MdescIdx	INT	Index of the motion descriptor in the bank

#### Outputs

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
MdescValue	T_Mdesc [▶ 76]	Values for the motion descriptor

#### Inputs/Outputs

Name	Type	Description
AxesGroup	T_StaeubliRobot [▶ 77]	Data block for a robot

### 5.1.1.10 VAL\_ReadPoint



FUNCTION\_BLOCK VAL\_ReadPoint

This function returns the coordinates of a cartesian point located in a bank of a the database.

#### Syntax

##### Definition:

```
FUNCTION_BLOCK VAL_ReadPoint
VAR_INPUT
    Execute      : BOOL := FALSE;
    UserDatabase : BOOL := FALSE;
    BankID       : INT  := -1;
    PointIdx     : INT  := -1;
END_VAR
VAR_OUTPUT
    Busy        : BOOL := FALSE;
    Done        : BOOL := FALSE;
    Error       : BOOL := FALSE;
    ErrorID     : UDINT := 0;
    PointValue  : T_CartesianPos;
END_VAR
VAR_IN_OUT
    AxesGroup   : T_StaebliRobot;
END_VAR
```

#### Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
UserDatabase	BOOL	True=Data of the user database will be used
BankID	INT	Number of the bank in which the point is located.
PointIdx	INT	Index of the point in the bank

#### Outputs

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
PointValue	T_CartesianPos [▶ 75]	Coordinates of the requested point

#### Inputs/Outputs

Name	Type	Description
AxesGroup	T_StaebliRobot [▶ 77]	Data block for a robot



5.1.1.11 VAL\_ReadTool



FUNCTION\_BLOCK VAL\_ReadTool

This function block returns the coordinates of a Tool Center Point located in a database.

Syntax

Definition:

```
FUNCTION_BLOCK VAL_ReadTool
VAR_INPUT
    Execute      : BOOL := FALSE;
    UserDatabase : BOOL := FALSE;
    ToolIdx      : INT := -1;
END_VAR
VAR_OUTPUT
    Busy       : BOOL := FALSE;
    Done       : BOOL := FALSE;
    Error      : BOOL := FALSE;
    ErrorID    : UDINT := 0;
    ToolValue  : T_Trssf;
END_VAR
VAR_IN_OUT
    AxesGroup  : T_StaeubliRobot;
END_VAR
```

Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
UserDatabase	BOOL	True=Data of the user database will be used
ToolIdx	INT	Index of the Tool in the bank of tool located in the database

Outputs

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
ToolValue	T_Trssf [▶ 79]	Coordinates of the requested Tool

Inputs/Outputs

Name	Type	Description
AxesGroup	T_StaeubliRobot [▶ 77]	Data block for a robot

### 5.1.1.12 VAL\_SetBankSize



FUNCTION\_BLOCK VAL\_SetBankSize

This function blocks resizes the bank of the specified type in user or plc database

#### Syntax

##### Definition:

```
FUNCTION_BLOCK VAL_SetBankSize
VAR_INPUT
    Execute      : BOOL := FALSE;
    UserDatabase : BOOL := FALSE;
    DataType     : INT  := 0;
    DataSize    : INT  := 0;
    BankID      : INT  := -1;
END_VAR
VAR_OUTPUT
    Busy       : BOOL := FALSE;
    Done       : BOOL := FALSE;
    Error      : BOOL := FALSE;
    ErrorID    : UDINT := 0;
END_VAR
VAR_IN_OUT
    AxesGroup  : T_StaebliRobot;
END_VAR
```


#### Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
UserDatabase	BOOL	True=Use user database; False=Use plc database
DataType	INT	Type of data to resize (1 = point, 2 = joint, 3 = tool, 4 = motion descriptor)
DataSize	INT	New size for the selected bank
BankID	INT	Number of the bank of point to resize. This parameter is effective only when DataType parameter = 1.

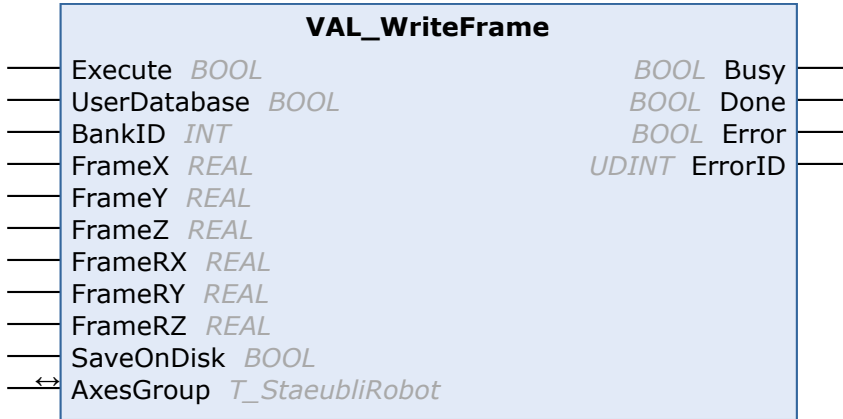
#### Outputs

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code

 /  Inputs/Outputs

Name	Type	Description
AxesGroup	T_StaeubliRobot [  77]	Data block for a robot

5.1.1.13 VAL\_WriteFrame



FUNCTION\_BLOCK VAL\_WriteFrame

This function writes the coordinates of a user frame associated to a bank of point located in a database.

Syntax

Definition:

```
FUNCTION_BLOCK VAL_WriteFrame
VAR_INPUT
    Execute      : BOOL := FALSE;
    UserDatabase : BOOL := FALSE;
    BankID       : INT := -1;
    FrameX       : REAL := 0;
    FrameY       : REAL := 0;
    FrameZ       : REAL := 0;
    FrameRX      : REAL := 0;
    FrameRY      : REAL := 0;
    FrameRZ      : REAL := 0;
    SaveOnDisk   : BOOL := TRUE;
END_VAR
VAR_OUTPUT
    Busy      : BOOL := FALSE;
    Done      : BOOL := FALSE;
    Error     : BOOL := FALSE;
    ErrorID   : UDINT := 0;
END_VAR
VAR_IN_OUT
    AxesGroup : T_StaeubliRobot;
END_VAR
```

 Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
UserDatabase	BOOL	True=Data of the user database will be used
BankID	INT	Number of the bank for which user frame is returned.
FrameX	REAL	Value for X coordinate
FrameY	REAL	Value for Y coordinate
FrameZ	REAL	Value for Z coordinate
FrameRX	REAL	Value for RX coordinate
FrameRY	REAL	Value for RY coordinate

Name	Type	Description
FrameRZ	REAL	Value for RZ coordinate
SaveOnDisk	BOOL	True=Tool data stored onto the controller's flashdisk. False=Data not stored--> faster FB execution

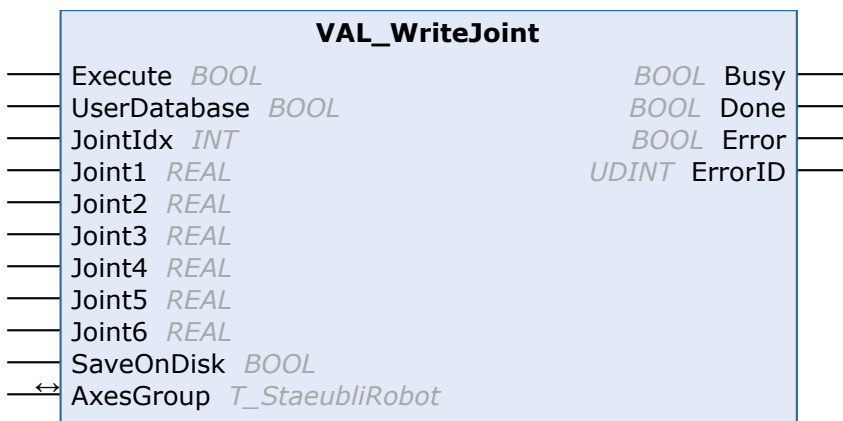
 **Outputs**

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code

 **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaeubliRobot  ▶ 77	Data block for a robot

**5.1.1.14 VAL\_WriteJoint**



FUNCTION\_BLOCK VAL\_WriteJoint

This function block allows to initialize a joint position inside a database

**Syntax**

**Definition:**

```
FUNCTION_BLOCK VAL_WriteJoint
VAR_INPUT
    Execute      : BOOL := FALSE;
    UserDatabase : BOOL := FALSE;
    JointIdx     : INT  := -1;
    Joint1      : REAL := 0;
    Joint2      : REAL := 0;
    Joint3      : REAL := 0;
    Joint4      : REAL := 0;
    Joint5      : REAL := 0;
    Joint6      : REAL := 0;
    SaveOnDisk  : BOOL := TRUE;
END_VAR
VAR_OUTPUT
    Busy       : BOOL := FALSE;
    Done       : BOOL := FALSE;
    Error      : BOOL := FALSE;
    ErrorID    : UDINT := 0;
```

```

END_VAR
VAR_IN_OUT
  AxesGroup : T_StaeubliRobot;
END_VAR
    
```

 **Inputs**

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
UserDatabase	BOOL	True=Data of the user database will be used; False=Use data of the plc database
JointIdx	INT	Index of the joint position in the bank
Joint1	REAL	Coordinate for axis 1
Joint2	REAL	Coordinate for axis 2
Joint3	REAL	Coordinate for axis 3
Joint4	REAL	Coordinate for axis 4
Joint5	REAL	Coordinate for axis 5 - not applicable for 4 axis robots-
Joint6	REAL	Coordinate for axis 6 - not applicable for 4 axis robots-
SaveOnDisk	BOOL	True=Tool data stored onto the controller's flashdisk. False=Data not stored--> faster FB execution

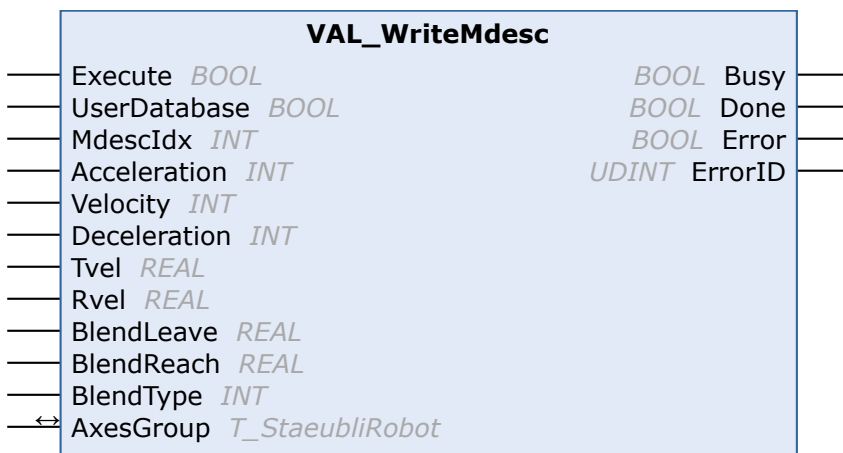
 **Outputs**

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code

 **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaeubliRobot [▶ 77]	Data block for a robot

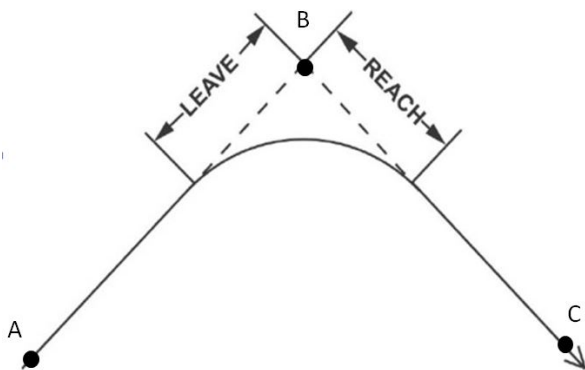
**5.1.1.15 VAL\_WriteMdesc**



FUNCTION\_BLOCK VAL\_WriteMdesc

This function block allows to set the properties of a motion descriptor located in a database. The following picture describes the definition of blending distances

Definition of the distances: 'leave' / 'reach'



**Syntax**

**Definition:**

```

FUNCTION_BLOCK VAL_WriteMdesc
VAR_INPUT
    Execute      : BOOL := FALSE;
    UserDatabase : BOOL := FALSE;
    MdescIdx     : INT  := -1;
    Acceleration : INT  := 100;
    Velocity     : INT  := 100;
    Deceleration : INT  := 100;
    Tvel         : REAL := 99999;
    Rvel         : REAL := 99999;
    BlendLeave    : REAL := 10;
    BlendReach   : REAL := 10;
    BlendType    : INT  := 0;
END_VAR
VAR_OUTPUT
    Busy      : BOOL := FALSE;
    Done      : BOOL := FALSE;
    Error     : BOOL := FALSE;
    ErrorID   : UDINT := 0;
END_VAR
VAR_IN_OUT
    AxesGroup : T_StaebliRobot;
END_VAR
    
```

 **Inputs**

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
UserDatabase	BOOL	True=Data of the user database will be used
MdescIdx	INT	Index of the motion descriptor in the bank.
Acceleration	INT	Joint acceleration. Percentage of the nominal acceleration of the robot
Velocity	INT	Joint velocity. Percentage of the nominal speed of the robot
Deceleration	INT	Joint deceleration. Percentage of the nominal deceleration of the robot
Tvel	REAL	Maximum cartesian velocity (mm/sec or inches/sec). Unit depends on settings on robot controller
Rvel	REAL	Maximum rotational velocity (mm/sec or inches/sec). Unit depends on settings on robot controller

Name	Type	Description
BlendLeave	REAL	Distance from the destination point at which the nominal trajectory is left
BlendReach	REAL	Distance from the destination point at which the nominal trajectory is joined again.
BlendType	INT	1=Off; 2=Joint; 3=Cartesian

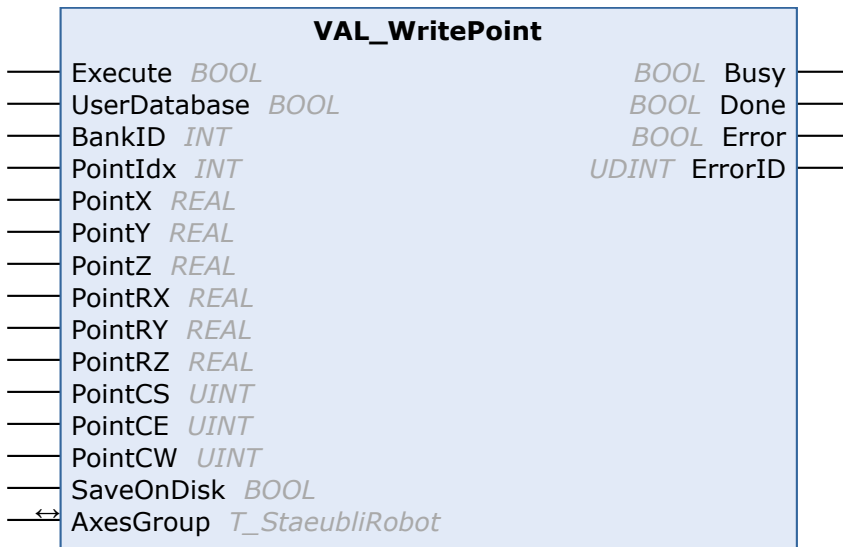
 **Outputs**

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code

 **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaubliRobot [▶ 77]	Data block for a robot

**5.1.1.16 VAL\_WritePoint**



FUNCTION\_BLOCK VAL\_WritePoint

This function block allows to modify the coordinates of a cartesian position located in a database.

**Syntax**

**Definition:**

```
FUNCTION_BLOCK VAL_WritePoint
VAR_INPUT
    Execute          : BOOL := FALSE;
    UserDatabase     : BOOL := FALSE;
    BankID           : INT := -1;
    PointIdx         : INT := -1;
    PointX           : REAL := 0;
    PointY           : REAL := 0;
    PointZ           : REAL := 0;
```

```

    PointRX      : REAL := 0;
    PointRY      : REAL := 0;
    PointRZ      : REAL := 0;
    PointCS      : UINT := 2;
    PointCE      : UINT := 2;
    PointCW      : UINT := 2;
    SaveOnDisk   : BOOL := TRUE;
END_VAR
VAR_OUTPUT
    Busy        : BOOL := FALSE;
    Done        : BOOL := FALSE;
    Error       : BOOL := FALSE;
    ErrorID     : UDINT := 0;
END_VAR
VAR_IN_OUT
    AxesGroup   : T_StaubliRobot;
END_VAR

```

## Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
UserDatabase	BOOL	True=Data of the user database will be used
BankID	INT	Number of the bank in which the point is located.
PointIdx	INT	Index of the cartesian point in the bank
PointX	REAL	Value for X coordinate
PointY	REAL	Value for Y coordinate
PointZ	REAL	Value for Z coordinate
PointRX	REAL	Value for RX coordinate
PointRY	REAL	Value for RY coordinate
PointRZ	REAL	Value for RZ coordinate
PointCS	UINT	Value for shoulder configuration (1=> sfree; 2=> ssame; 3=> righty; 4=> lefty)
PointCE	UINT	Value for elbow configuration (1=> sfree; 2=> ssame; 3=> epositive; 4=> enegative)
PointCW	UINT	Value for wrist configuration (1=> sfree; 2=> ssame; 3=> wpositive; 4=> wnegative)
SaveOnDisk	BOOL	True=Tool data stored onto the controller's flashdisk. False=Data not stored--> faster FB execution

## Outputs

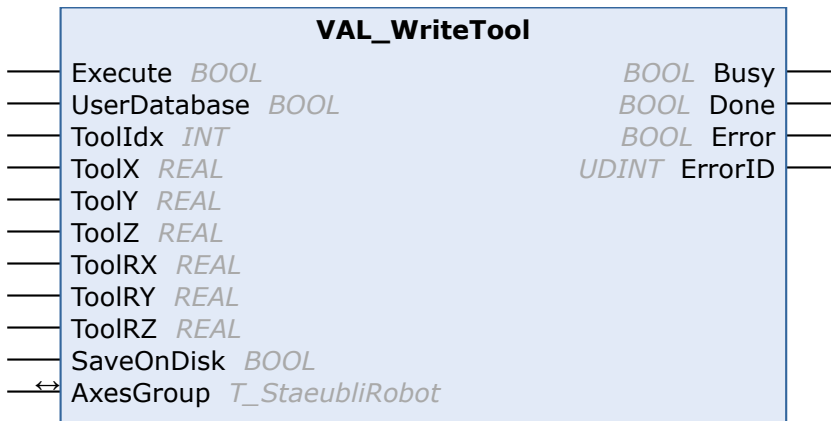
Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code

## Inputs/Outputs

Name	Type	Description
AxesGroup	T_StaubliRobot [ <a href="#">▶ 77</a> ]	Data block for a robot



5.1.1.17 VAL\_WriteTool



FUNCTION\_BLOCK VAL\_WriteTool

This function block allows to modify the coordinates of a tool located in a database.

Syntax

Definition:

```

FUNCTION_BLOCK VAL_WriteTool
VAR_INPUT
    Execute      : BOOL := FALSE;
    UserDatabase : BOOL := FALSE;
    ToolIdx      : INT := -1;
    ToolX        : REAL := 0;
    ToolY        : REAL := 0;
    ToolZ        : REAL := 0;
    ToolRX       : REAL := 0;
    ToolRY       : REAL := 0;
    ToolRZ       : REAL := 0;
    SaveOnDisk   : BOOL := TRUE;
END_VAR
VAR_OUTPUT
    Busy      : BOOL := FALSE;
    Done      : BOOL := FALSE;
    Error     : BOOL := FALSE;
    ErrorID   : UDINT := 0;
END_VAR
VAR_IN_OUT
    AxesGroup : T_StaebliRobot;
END_VAR
    
```

📌 Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
UserDatabase	BOOL	True=Data of the user database will be used
ToolIdx	INT	Index of the tool position in the bank
ToolX	REAL	Value for X coordinate
ToolY	REAL	Value for Y coordinate
ToolZ	REAL	Value for Z coordinate
ToolRX	REAL	Value for RX coordinate
ToolRY	REAL	Value for RY coordinate
ToolRZ	REAL	Value for RZ coordinate
SaveOnDisk	BOOL	True=Tool data stored onto the controller's flashdisk. False=Data not stored--> faster FB execution

 **Outputs**

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code

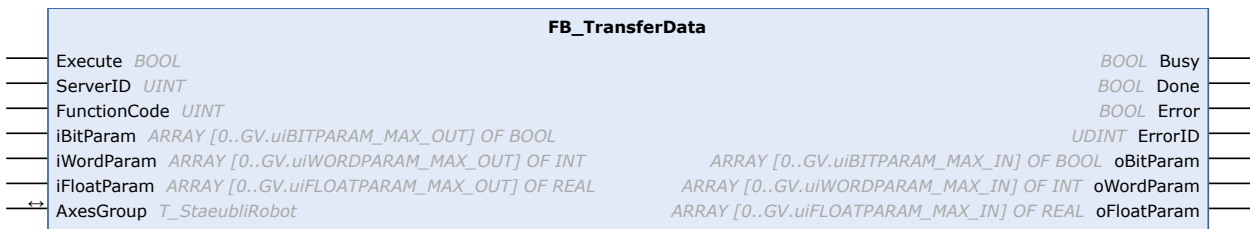
 **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaeubliRobot <a href="#">▶ 77</a>	Data block for a robot

## 5.1.2 Internal

The item listed in this folder are for internal use only. The functions / function blocs are not intended to be called by user application

### 5.1.2.1 FB\_TransferData



#### FUNCTION\_BLOCK FB\_TransferData

This function block manages the communication with uniVAL plc server installed on the robot side

#### Syntax

##### Definition:

```

FUNCTION_BLOCK FB_TransferData
VAR_INPUT
    Execute      : BOOL := FALSE;
    ServerID     : UINT := 0;
    FunctionCode : UINT := 0;
    iBitParam    : ARRAY [0..GV.uiBITPARAM_MAX_OUT] OF BOOL;
    iWordParam   : ARRAY [0..GV.uiWORDPARAM_MAX_OUT] OF INT;
    iFloatParam  : ARRAY [0..GV.uiFLOATPARAM_MAX_OUT] OF REAL;
END_VAR
VAR_OUTPUT
    Busy        : BOOL := FALSE;
    Done        : BOOL := FALSE;
    Error       : BOOL := FALSE;
    ErrorID     : UDINT := 0;
    oBitParam   : ARRAY [0..GV.uiBITPARAM_MAX_IN] OF BOOL;
    oWordParam  : ARRAY [0..GV.uiWORDPARAM_MAX_IN] OF INT;
    oFloatParam : ARRAY [0..GV.uiFLOATPARAM_MAX_IN] OF REAL;
END_VAR
VAR_IN_OUT
    AxesGroup   : T_StaeubliRobot;
END_VAR
    
```

 Inputs

Name	Type	Description
Execute	BOOL	Rising edge starts function execution
ServerID	UINT	Number of the server to address
FunctionCode	UINT	Code of the function to address
iBitParam	ARRAY [0..GV.uiBITPARAM_MAX_OUT] OF BOOL	
iWordParam	ARRAY [0..GV.uiWORDPARAM_MAX_OUT] OF INT	
iFloatParam	ARRAY [0..GV.uiFLOATPARAM_MAX_OUT] OF REAL	

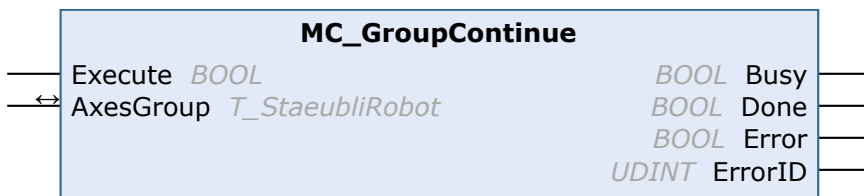
 Outputs

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
oBitParam	ARRAY [0..GV.uiBITPARAM_MAX_IN] OF BOOL	
oWordParam	ARRAY [0..GV.uiWORDPARAM_MAX_IN] OF INT	
oFloatParam	ARRAY [0..GV.uiFLOATPARAM_MAX_IN] OF REAL	

 Inputs/Outputs

Name	Type	Description
AxesGroup	T_StaubliRobot <a href="#">[▶ 77]</a>	Data block for a robot

### 5.1.3 MC\_GroupContinue



FUNCTION\_BLOCK MC\_GroupContinue

This function blocks allows to continue the movement that has been previously stopped by the MC\_Interrupt function block.

**Syntax**

**Definition:**

```
FUNCTION_BLOCK MC_GroupContinue
VAR_INPUT
    Execute : BOOL := FALSE;
END_VAR
VAR_OUTPUT
    Busy : BOOL := FALSE;
    Done : BOOL := FALSE;
    Error : BOOL := FALSE;
    ErrorID : UDINT := 0;
END_VAR
VAR_IN_OUT
    AxesGroup : T_StaeubliRobot;
END_VAR
```

 **Inputs**

Name	Type	Description
Execute	BOOL	Rising edge starts function execution

 **Outputs**

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code

 **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaeubliRobot [ <a href="#">▶ 77</a> ]	Data block for a robot

**5.1.4 MC\_GroupInterrupt**



FUNCTION\_BLOCK MC\_GroupInterrupt

This function block commands a controlled stop of the robot along the commanded trajectory. The robot power remains enabled. When velocity zero is reached the DONE output set. Unlike MC\_GroupStop, all commanded movements are not cancelled, this means the robot can continue its trajectory as soon as MC\_GroupContinue is executed. At the interrupted FB the output CommandAborted will not be set, Busy is still high and Active is reset.

**Syntax**

**Definition:**

```
FUNCTION_BLOCK MC_GroupInterrupt
VAR_INPUT
    Execute : BOOL := FALSE;
```

```

END_VAR
VAR_OUTPUT
  Busy      : BOOL := FALSE;
  Done      : BOOL := FALSE;
  Error     : BOOL := FALSE;
  ErrorID   : UDINT := 0;
END_VAR
VAR_IN_OUT
  AxesGroup : T_StaeubliRobot;
END_VAR
    
```

 **Inputs**

Name	Type	Description
Execute	BOOL	Rising edge starts function execution

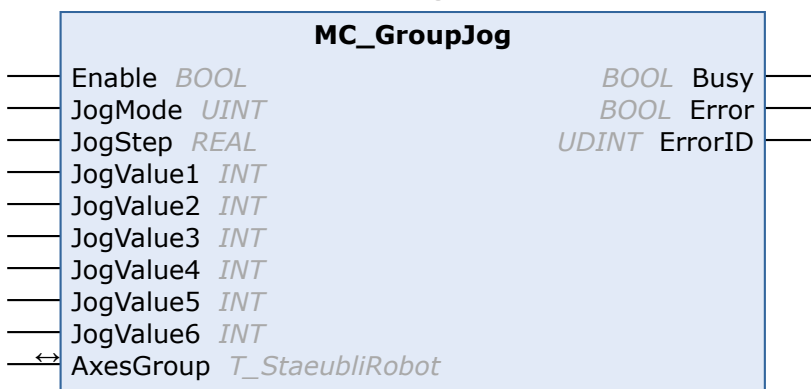
 **Outputs**

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code

 /  **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaeubliRobot [ <a href="#">▶ 77</a> ]	Data block for a robot

### 5.1.5 MC\_GroupJog



FUNCTION\_BLOCK MC\_GroupJog

This function block allows to move the robot from a user-supplied teach pendant in manual mode only.

- The robot can be moved in speed or in position mode depending on the value of the JogStep input parameter.
- Several axes can be moved simultaneously when their dedicated input are set.
- The robot speed still depends on the override value.
- If JogMode input is Frame(2) or Tool(3) the robot will move along Frame and Tool coordinate systems selected by Staeubli\_Robot.Command.CoordSystemCmd and Staeubli\_Robot.Command.ToolCmd

Examples

- If you set JogStep value to 0 and JogValue\_xx to 50% and the override value is 10%, the selected axes will move with 5% of the nominal speed.
- If you set JogStep value to 3 and JogValue\_xx to 50%, the selected axes will move:
  - 1.5 degrees or millimeters when JogMode is Joint (1).
  - 1.5 millimeters in direction X,Y,Z when JogMode is Frame(2) or Tool(3).
  - 1.5 degrees in direction RX,RY,RZ when JogMode is Frame(2) or Tool(3).



This function is effective only when Stäubli Teach Pendant is removed and replaced by the delivered dummy plug.  
 The controller must be configured to have user-supplied teach pendant in use (file iomap.cfx on controller).  
 Manual operation mode can be selected either from Stäubli Working Mode Selector Box or from user-supplied teach pendant

**Syntax**

**Definition:**

```

FUNCTION_BLOCK MC_GroupJog
VAR_INPUT
    Enable      : BOOL := FALSE;
    JogMode     : UINT := 0;
    JogStep     : REAL := 0;
    JogValue1   : INT  := 0;
    JogValue2   : INT  := 0;
    JogValue3   : INT  := 0;
    JogValue4   : INT  := 0;
    JogValue5   : INT  := 0;
    JogValue6   : INT  := 0;
END_VAR
VAR_OUTPUT
    Busy       : BOOL := FALSE;
    Error      : BOOL := FALSE;
    ErrorID    : UDINT := 0;
END_VAR
VAR_IN_OUT
    AxesGroup  : T_StaebliRobot;
END_VAR
    
```

**Inputs**

Name	Type	Description
Enable	BOOL	True= content of the function block is executed
JogMode	UINT	Select jog mode. 1=Joint ; 2=Frame ; 3=Tool
JogStep	REAL	When this value is non zero, the robot is jogged in step mode.  When a jogValue is set, the robot moves until the specified distance is reached.  The JogValue shall be reset and set again to trigger the next movement with the specified distance.
JogValue1	INT	Commanded speed for Axis1 / X direction [-100%..100%]
JogValue2	INT	Commanded speed for Axis2 / Y direction [-100%..100%]
JogValue3	INT	Commanded speed for Axis3 / Z direction [-100%..100%]
JogValue4	INT	Commanded speed for Axis4 / RX direction [-100%..100%]
JogValue5	INT	Commanded speed for Axis5 / RY direction [-100%..100%]
JogValue6	INT	Commanded speed for Axis6 / RZ direction [-100%..100%]

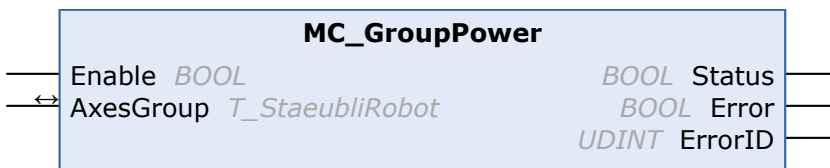
 **Outputs**

Name	Type	Description
Busy	BOOL	A valid jog command is sent to the robot
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code

 **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaeubliRobot [ <a href="#">▶ 77</a> ]	Data block for a robot

### 5.1.6 MC\_GroupPower



FUNCTION\_BLOCK MC\_GroupPower

This function block manages the arm power ON/OFF sequence. Switching arm power ON/OFF is possible only if remote automatic operation mode is selected. The operation mode is selected with the Working Mode Selector box. The arm power ON / arm power OFF sequences on the robot side can take up to 1 second until robot is effectively powered ON/OFF. During that time Enable input and Status output have different values.

- Enable=TRUE : Command to switch robot power ON.
- Enable=FALSE: Command to switch robot power OFF.



If an error occurs, the Enable input must go low and high once again to re-execute the function block. However, the pending error(s) must be acknowledged using the MC\_GroupReset function block.

When a user-supplied teach pendant is in use, this function block allows to power ON/OFF the robot in any operation mode.

**Syntax**

**Definition:**

```
FUNCTION_BLOCK MC_GroupPower
VAR_INPUT
  Enable : BOOL := FALSE;
END_VAR
VAR_OUTPUT
  Status : BOOL := FALSE;
  Error : BOOL := FALSE;
  ErrorID : UDINT := 0;
END_VAR
VAR_IN_OUT
  AxesGroup : T_StaeubliRobot;
END_VAR
```

 **Inputs**

Name	Type	Description
Enable	BOOL	TRUE= Command to switch power on ; FALSE= Command to switch power off.

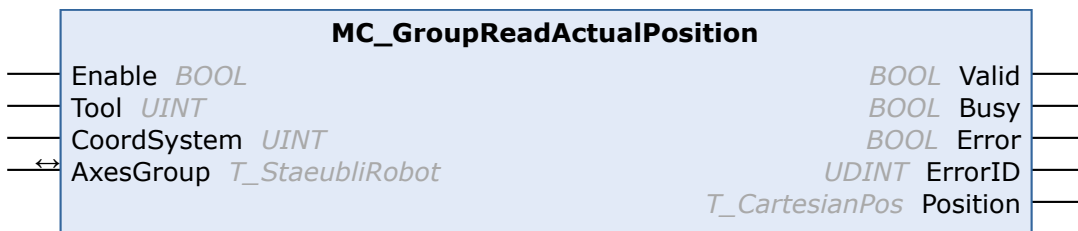
 **Outputs**

Name	Type	Description
Status	BOOL	Effective state of the robot power.
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code

 /  **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaebliRobot [ <a href="#">▶ 77</a> ]	Data block for a robot

### 5.1.7 MC\_GroupReadActualPosition



FUNCTION\_BLOCK MC\_GroupReadActualPosition

This function block provides a convenient way of reading the current position of the selected Tool Center Point in the selected user frame.



It shall be noticed that this function block may change briefly the value of the following members of the robot data block  
 T\_StaebliRobot.Command.ToolCmd  
 T\_StaebliRobot.Command.CoordSystemCmd until the position of the robot is effectively read.  
 This may impact your application if it continuously read the position of the robot directly from the robot data block.

**Syntax**

**Definition:**

```
FUNCTION_BLOCK MC_GroupReadActualPosition
VAR_INPUT
    Enable      : BOOL := FALSE;
    Tool        : UINT := 0;
    CoordSystem : UINT := 0;
END_VAR
VAR_OUTPUT
    Valid       : BOOL := FALSE;
    Busy        : BOOL := FALSE;
    Error       : BOOL := FALSE;
    ErrorID     : UDINT := 0;
    Position    : T_CartesianPos;
END_VAR
VAR_IN_OUT
    AxesGroup   : T_StaebliRobot;
END_VAR
```

 **Inputs**

Name	Type	Description
Enable	BOOL	TRUE = Enable the execution of the function block
Tool	UINT	Number of the Tool for which position is reported



Name	Type	Description
CoordSystem	UINT	Number of the user frame in which the position of selected Tool Center Point is reported

 **Outputs**

Name	Type	Description
Valid	BOOL	TRUE = A valid set of data are available.
Busy	BOOL	TRUE = function block is proceeding...
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
Position	T_CartesianPos [ <a href="#">▶ 75</a> ]	Current cartesian position of the selected Tool in the selected user frame

 /  **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaeubliRobot [ <a href="#">▶ 77</a> ]	Data block for a robot

### 5.1.8 MC\_GroupReset



FUNCTION\_BLOCK MC\_GroupReset

This function blocks allows to reset all pending errors on robot side. Further actions might be executed by the MC\_GroupReset function block. Telegrams transmitted between PLC and robot controller may end in a deadlock (e.g. one device is restarted during data exchange). In this situation, the function block resets the communication protocol. After this, new telegrams can be exchanged.

**● For CS8C controller ONLY**

**i** If the state of one dual channel contact in the CS8C E-Stop chain is not consistent, a hardware fault is registered and robot operation is not possible until this hardware fault is repaired, tested, and acknowledged. (See CS8C instruction manual for further details)  
Once the hardware fault is tested, the MC\_GroupReset function block performs the required acknowledgment.

**Syntax**

**Definition:**

```

FUNCTION_BLOCK MC_GroupReset
VAR_INPUT
    Execute : BOOL := FALSE;
END_VAR
VAR_OUTPUT
    Busy : BOOL := FALSE;
    Done : BOOL := FALSE;
    Error : BOOL := FALSE;
    ErrorID : UDINT := 0;
END_VAR
VAR_IN_OUT
    AxesGroup : T_StaeubliRobot;
END_VAR
  
```

 **Inputs**

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution

 **Outputs**

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code

 **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaeubliRobot [ <a href="#">▶ 77</a> ]	Data block for a robot

### 5.1.9 MC\_GroupStop



FUNCTION\_BLOCK MC\_GroupStop

This function block commands a controlled motion stop. Moreover, this function cancels any movement that have been commanded and buffered on robot side.

#### Syntax

**Definition:**

```
FUNCTION_BLOCK MC_GroupStop
VAR_INPUT
    Execute : BOOL := FALSE;
END_VAR
VAR_OUTPUT
    Busy : BOOL := FALSE;
    Done : BOOL := FALSE;
    Error : BOOL := FALSE;
    ErrorID : UDINT := 0;
END_VAR
VAR_IN_OUT
    AxesGroup : T_StaeubliRobot;
END_VAR
```

 **Inputs**

Name	Type	Description
Execute	BOOL	Rising edge starts function execution

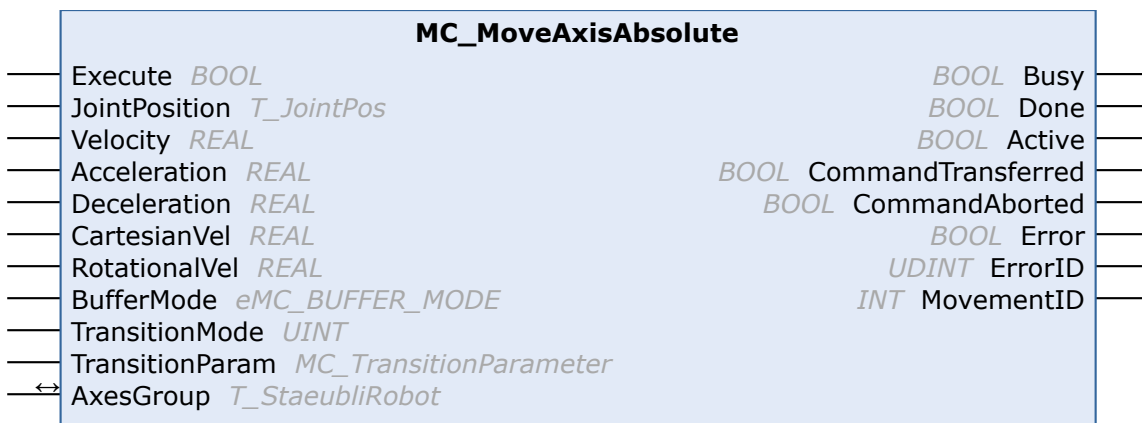
 **Outputs**

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code

 /  **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaubliRobot [ <a href="#">▶ 77</a> ]	Data block for a robot

### 5.1.10 MC\_MoveAxisAbsolute



FUNCTION\_BLOCK MC\_MoveAxisAbsolute

This function block commands a joint interpolated movement to an end-position. The values specified for the end-position are interpreted as joint values. They represent the position of each axis in degrees.

The speed of the robot is calculated with both

- Parameters related to axes of the robot (Velocity / Acceleration / Deceleration)
- Parameter related to the Tool Center Point (CartesianVel / RotationalVel)

The final robot speed is set by the most limiting of these parameters. When the speed of the object, carried by the robot, does not need to be handled with special care, parameters related to Tool Center Point can be ignored (kept to default values). The speed along the trajectory may change depending on the robot position in the space.

On the contrary, if a constant speed is required for the Tool Center Point, it shall be set with CartesianVel parameter.

**Example: The speed of the object carried by the robot shall be 100mm/s.**

Set the CartesianVel parameter to 100. However, axes related speed/acceleration parameters are still active. If you set the Velocity parameter too low (e.g. 10%), the object may not reach 100mm/s.

**Syntax**

**Definition:**

```

FUNCTION_BLOCK MC_MoveAxisAbsolute
VAR_INPUT
  Execute          : BOOL := FALSE;
  JointPosition    : T_JointPos;
  Velocity         : REAL := REAL#100;
  Acceleration     : REAL := REAL#100;
  Deceleration     : REAL := REAL#100;
  CartesianVel     : REAL := REAL#99999;
  RotationalVel    : REAL := REAL#99999;
  BufferMode       : eMC_BUFFER_MODE := Buffered;
  TransitionMode   : UINT := 0;
  TransitionParam  : MC_TransitionParameter;
END_VAR
VAR_OUTPUT
  Busy            : BOOL := FALSE;
  Done            : BOOL := FALSE;
  Active         : BOOL := FALSE;
  CommandTransferred : BOOL := FALSE;
  CommandAborted : BOOL := FALSE;
  Error          : BOOL := FALSE;
  ErrorID        : UDINT := 0;
  MovementID     : INT := -1;
END_VAR
VAR_IN_OUT
  AxesGroup : T_StaeubliRobot;
END_VAR

```

### Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
JointPosition	T_JointPos <a href="#">[▶ 76]</a>	Absolute end position for each axis
Velocity	REAL	Joint velocity. Percentage of the nominal speed of the robot. Range [0.01 .. 500]
Acceleration	REAL	Joint acceleration. Percentage of the nominal acceleration of the robot. Range [0.01 .. 500]
Deceleration	REAL	Joint deceleration. Percentage of the nominal deceleration of the robot. Range [0.01 .. 500]
CartesianVel	REAL	Optional: Maximum cartesian velocity at Tool Center Point in mm/s
RotationalVel	REAL	Optional: Maximum rotational velocity at Tool Center Point in degree/s
BufferMode	eMC_BUFFER_MODE <a href="#">[▶ 82]</a>	0 = aborting / 1= buffered / 6-7 = blending
TransitionMode	UINT	Defines the profile of the robot trajectory nearby the set points. 0 = None / 3 = Corner distance / 10 = blending distances
TransitionParam	MC_TransitionParameter <a href="#">[▶ 73]</a>	

### Outputs

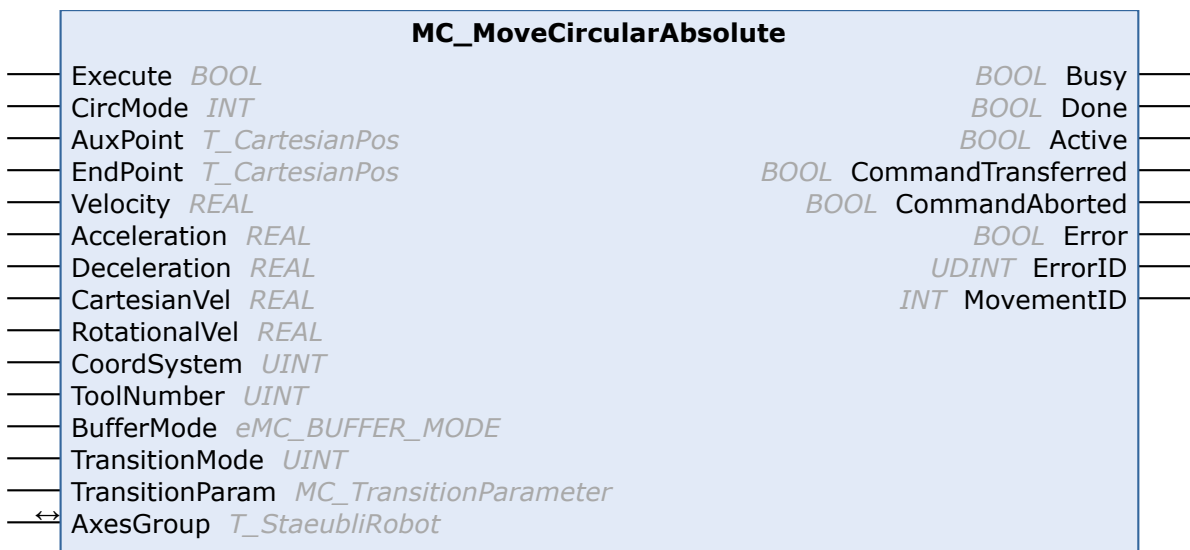
Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Active	BOOL	Set when commanded movement is currently executed by the robot
CommandTransferred	BOOL	Commanded motion is successfully buffered inside robot controller
CommandAborted	BOOL	Command is aborted by another command

Name	Type	Description
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
MovementID	INT	Identifier for this commanded movement

 Inputs/Outputs

Name	Type	Description
AxesGroup	T_StaebliRobot [▶ 77]	Data block for a robot

### 5.1.11 MC\_MoveCircularAbsolute



FUNCTION\_BLOCK MC\_MoveCircularAbsolute

This function block commands a circular interpolated movement starting from the actual position of the TCP. The end point as well as the auxiliary point are defined absolutely in the specified coordinate system.

The speed of the robot is calculated with both

- Parameters related to axes of the robot (Velocity / Acceleration / Deceleration)
- Parameter related to the Tool Center Point (CartesianVel / RotationalVel)

The final robot speed is set by the most limiting of these parameters. When the speed of the object, carried by the robot, does not need to be handled with special care, parameters related to Tool Center Point can be ignored (kept to default values). The speed along the trajectory may change depending on the robot position in the space.

On the contrary, if a constant speed is required for the Tool Center Point, it shall be set with CartesianVel parameter.

**Example: The speed of the object carried by the robot shall be 100mm/s.**

Set the CartesianVel parameter to 100. However, axes related speed/acceleration parameters are still active. If you set the Velocity parameter too low (e.g. 10%), the object may not reach 100mm/s.

**Syntax**

**Definition:**

```
FUNCTION_BLOCK MC_MoveCircularAbsolute
VAR_INPUT
    Execute          : BOOL := FALSE;
    CircMode         : INT  := 3;
```

```

AuxPoint      : T_CartesianPos;
EndPoint      : T_CartesianPos;
Velocity      : REAL := REAL#100;
Acceleration  : REAL := REAL#100;
Deceleration  : REAL := REAL#100;
CartesianVel  : REAL := REAL#99999;
RotationalVel : REAL := REAL#99999;
CoordSystem   : UINT := 0;
ToolNumber    : UINT := 0;
BufferMode    : eMC_BUFFER_MODE := Buffered;
TransitionMode : UINT := 0;
TransitionParam : MC_TransitionParameter;
END_VAR
VAR_OUTPUT
  Busy          : BOOL := FALSE;
  Done          : BOOL := FALSE;
  Active        : BOOL := FALSE;
  CommandTransferred : BOOL := FALSE;
  CommandAborted : BOOL := FALSE;
  Error         : BOOL := FALSE;
  ErrorID       : UDINT := 0;
  MovementID    : INT := -1;
END_VAR
VAR_IN_OUT
  AxesGroup : T_StaeubliRobot;
END_VAR

```

 **Inputs**

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
CircMode	INT	Specifies the meaning of the input signals 'AuxPoint' and 'CircDirection'. Only 'BORDER' mode is supported.
AuxPoint	<u>T_CartesianPos</u> <a href="#">▶ 75</a>	Coordinates of the intermediate position in the specified coordinate system.
EndPoint	<u>T_CartesianPos</u> <a href="#">▶ 75</a>	Coordinates of the end point in the specified coordinate system.
Velocity	REAL	Joint velocity. Percentage of the nominal speed of the robot. Range [0.01 .. 500]
Acceleration	REAL	Joint acceleration. Percentage of the nominal acceleration of the robot. Range [0.01 .. 500]
Deceleration	REAL	Joint deceleration. Percentage of the nominal deceleration of the robot. Range [0.01 .. 500]
CartesianVel	REAL	Optional: Maximum cartesian velocity at Tool Center Point in mm/s
RotationalVel	REAL	Optional: Maximum rotational velocity at Tool Center Point in degree/s
CoordSystem	UINT	Number of the coordinate system (located in the database) in which the AuxPoint and EndPoint variables are applied
ToolNumber	UINT	Index of the Tool in the bank that shall be used for the movement.
BufferMode	<u>eMC_BUFFER_MODE</u> <a href="#">▶ 82</a>	0 = aborting / 1= buffered / 6-7 = blending
TransitionMode	UINT	Defines the profile of the robot trajectory nearby the set points. 0 = None / 3 = Corner distance / 10 = blending distances
TransitionParam	<u>MC_TransitionParameter</u> <a href="#">▶ 73</a>	

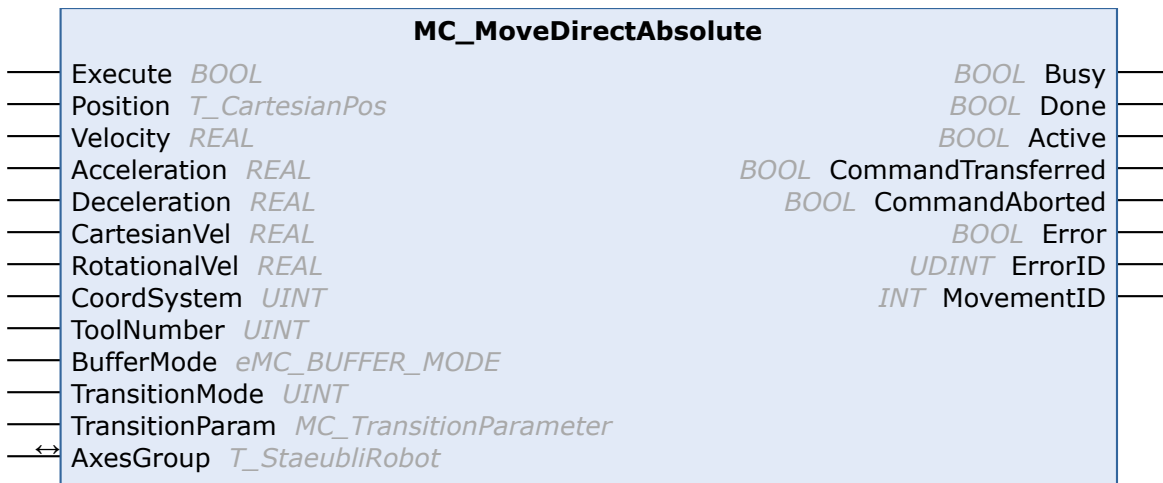
 **Outputs**

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Active	BOOL	Set when commanded movement is currently executed by the robot
CommandTransferred	BOOL	Commanded motion is successfully buffered inside robot controller
CommandAborted	BOOL	Command is aborted by another command
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
MovementID	INT	Identifier for this commanded movement

 **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaeubliRobot  ▶ 77	Data block for a robot

### 5.1.12 MC\_MoveDirectAbsolute



FUNCTION\_BLOCK MC\_MoveDirectAbsolute

This function block commands a joint interpolated movement from the current robot position to the specified absolute position in the specified coordinate system.

The speed of the robot is calculated with both

- Parameters related to axes of the robot (Velocity / Acceleration / Deceleration)
- Parameter related to the Tool Center Point (CartesianVel / RotationalVel)

The final robot speed is set by the most limiting of these parameters. When the speed of the object, carried by the robot, does not need to be handled with special care, parameters related to Tool Center Point can be ignored (kept to default values). The speed along the trajectory may be changed depending on the robot position in the space.

On the contrary, if a constant speed is required for the Tool Center Point, it shall be set with CartesianVel parameter.

**Example: The speed of the object carried by the robot shall be 100mm/s.**

Set the CartesianVel parameter to 100. However, axes related speed/acceleration parameters are still active. If you set the Velocity parameter too low (e.g. 10%), the object may not reach 100mm/s.

## Syntax

### Definition:

```
FUNCTION_BLOCK MC_MoveDirectAbsolute
VAR_INPUT
    Execute          : BOOL := FALSE;
    Position         : T_CartesianPos;
    Velocity         : REAL := REAL#100;
    Acceleration     : REAL := REAL#100;
    Deceleration     : REAL := REAL#100;
    CartesianVel     : REAL := REAL#99999;
    RotationalVel    : REAL := REAL#99999;
    CoordSystem      : UINT := 0;
    ToolNumber       : UINT := 0;
    BufferMode        : eMC_BUFFER_MODE := Buffered;
    TransitionMode   : UINT := 0;
    TransitionParam  : MC_TransitionParameter;
END_VAR
VAR_OUTPUT
    Busy             : BOOL := FALSE;
    Done             : BOOL := FALSE;
    Active           : BOOL := FALSE;
    CommandTransferred : BOOL := FALSE;
    CommandAborted  : BOOL := FALSE;
    Error            : BOOL := FALSE;
    ErrorID          : UDINT := 0;
    MovementID      : INT := -1;
END_VAR
VAR_IN_OUT
    AxesGroup       : T_StaeubliRobot;
END_VAR
```

### Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
Position	T_CartesianPos <a href="#">[► 75]</a>	Absolute end positions for each dimension in the specified coordinate system.
Velocity	REAL	Joint velocity. Percentage of the nominal speed of the robot. Range [0.01 .. 500]
Acceleration	REAL	Joint acceleration. Percentage of the nominal acceleration of the robot. Range [0.01 .. 500]
Deceleration	REAL	Joint deceleration. Percentage of the nominal deceleration of the robot. Range [0.01 .. 500]
CartesianVel	REAL	Optional: Maximum cartesian velocity at Tool Center Point in mm/s
RotationalVel	REAL	Optional: Maximum rotational velocity at Tool Center Point in degree/s
CoordSystem	UINT	Number of the coordinate system (located in a database on robot side) in which the Position input parameter is applied
ToolNumber	UINT	Index of the Tool Center Point (located in a database on robot side) that shall be used for the movement.
BufferMode	eMC_BUFFER_MODE <a href="#">[► 82]</a>	0 = aborting / 1 = buffered / 6-7 = blending
TransitionMode	UINT	Defines the profile of the robot trajectory nearby the set points. 0 = None / 3 = Corner distance / 10 = blending distances



Name	Type	Description
TransitionParam	MC_TransitionParameter [▶ 73]	

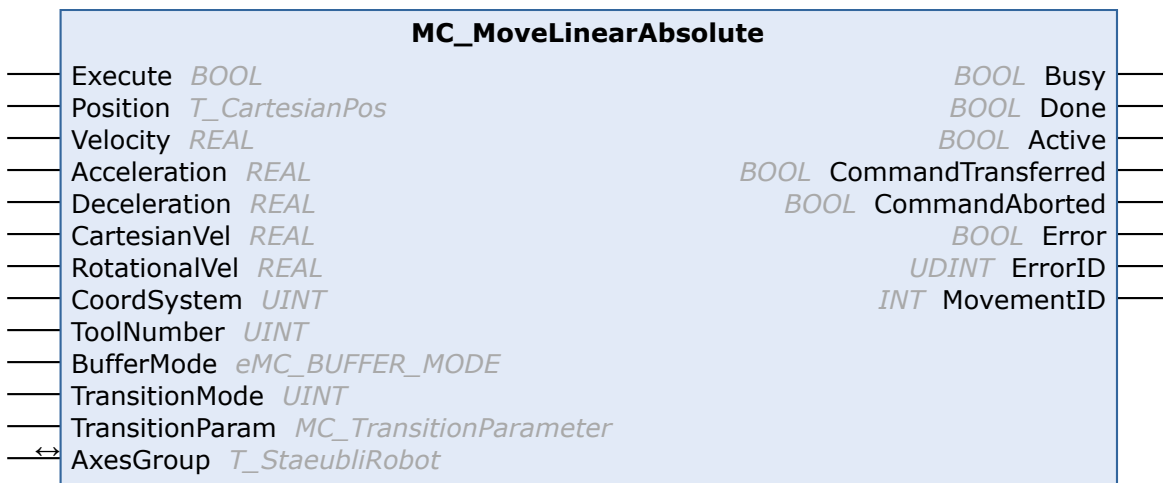
 **Outputs**

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Active	BOOL	Set when commanded movement is currently executed by the robot
CommandTransferred	BOOL	Commanded motion is successfully buffered inside robot controller
CommandAborted	BOOL	Command is aborted by another command
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
MovementID	INT	Identifier for this commanded movement

 **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaebliRobot [▶ 77]	Data block for a robot

### 5.1.13 MC\_MoveLinearAbsolute



FUNCTION\_BLOCK MC\_MoveLinearAbsolute

This function block commands a linear interpolated movement from the current robot position to the specified absolute position in the specified coordinate system.

The speed of the robot is calculated with both

- Parameters related to axes of the robot (Velocity / Acceleration / Deceleration)
- Parameter related to the Tool Center Point (CartesianVel / RotationalVel)

The final robot speed is set by the most limiting of these parameters. When the speed of the object, carried by the robot, does not need to be handled with special care, parameters related to Tool Center Point can be ignored (kept to default values). The speed along the trajectory may be changed depending on the robot position in the space.

On the contrary, if a constant speed is required for the Tool Center Point, it shall be set with CartesianVel parameter.

**Example: The speed of the object carried by the robot shall be 100mm/s.**

Set the CartesianVel parameter to 100. However, axes related speed/acceleration parameters are still active. If you set the Velocity parameter too low (e.g. 10%), the object may not reach 100mm/s.

## Syntax

### Definition:

```
FUNCTION_BLOCK MC_MoveLinearAbsolute
VAR_INPUT
    Execute          : BOOL := FALSE;
    Position         : T_CartesianPos;
    Velocity         : REAL := REAL#100;
    Acceleration     : REAL := REAL#100;
    Deceleration     : REAL := REAL#100;
    CartesianVel     : REAL := REAL#99999;
    RotationalVel    : REAL := REAL#99999;
    CoordSystem      : UINT := 0;
    ToolNumber       : UINT := 0;
    BufferMode        : eMC_BUFFER_MODE := Buffered;
    TransitionMode   : UINT := 0;
    TransitionParam  : MC_TransitionParameter;
END_VAR
VAR_OUTPUT
    Busy             : BOOL := FALSE;
    Done             : BOOL := FALSE;
    Active           : BOOL := FALSE;
    CommandTransferred : BOOL := FALSE;
    CommandAborted  : BOOL := FALSE;
    Error            : BOOL := FALSE;
    ErrorID          : UDINT := 0;
    MovementID      : INT := -1;
END_VAR
VAR_IN_OUT
    AxesGroup       : T_StaeubliRobot;
END_VAR
```

### Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
Position	T_CartesianPos [ <a href="#">▶ 75</a> ]	Absolute end positions for each dimension in the specified coordinate system.
Velocity	REAL	Joint velocity. Percentage of the nominal speed of the robot. Range [0.01 .. 500]
Acceleration	REAL	Joint acceleration. Percentage of the nominal acceleration of the robot. Range [0.01 .. 500]
Deceleration	REAL	Joint deceleration. Percentage of the nominal deceleration of the robot. Range [0.01 .. 500]
CartesianVel	REAL	Optional: Maximum cartesian velocity at Tool Center Point in mm/s
RotationalVel	REAL	Optional: Maximum rotational velocity at Tool Center Point in degree/s
CoordSystem	UINT	Number of the coordinate system (located in a database on robot side) in which the Position input parameter is applied

Name	Type	Description
ToolNumber	UINT	Index of the Tool in the bank that shall be used for the movement.
BufferMode	eMC_BUFFER_MODE [▶ 82]	0 = aborting / 1= buffered / 6-7 = blending
TransitionMode	UINT	Defines the profile of the robot trajectory nearby the set points. 0 = None / 3 = Corner distance / 10 = blending distances
TransitionParam	MC_TransitionParameter [▶ 73]	

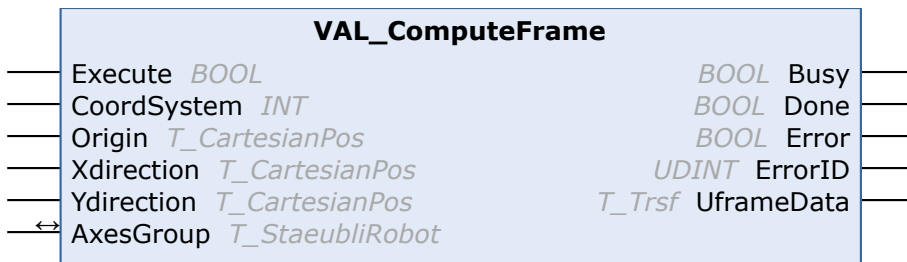
 **Outputs**

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Active	BOOL	Set when commanded movement is currently executed by the robot
CommandTransferred	BOOL	Commanded motion is successfully buffered inside robot controller
CommandAborted	BOOL	Command is aborted by another command
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
MovementID	INT	Identifier for this commanded movement

 **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaebliRobot [▶ 77]	Data block for a robot

### 5.1.14 VAL\_ComputeFrame



FUNCTION\_BLOCK VAL\_ComputeFrame

This functions blocks computes the coordinates of a the user frame associated to a bank of cartesian point in the plc database.

A user frame is computed with 3 cartesian positions that shall be taught as shown below.

To achieve maximal accuracy, it is recommend to teach Ox vector along the longest side of the object for which a user frame will be defined.

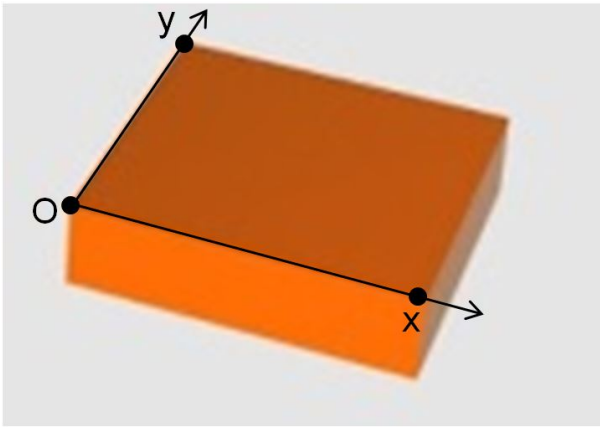


Fig. 1: Frame definition with 3 points

## Syntax

### Definition:

```

FUNCTION_BLOCK VAL_ComputeFrame
VAR_INPUT
    Execute      : BOOL := FALSE;
    CoordSystem  : INT := -1;
    Origin       : T_CartesianPos;
    Xdirection   : T_CartesianPos;
    Ydirection   : T_CartesianPos;
END_VAR
VAR_OUTPUT
    Busy        : BOOL := FALSE;
    Done        : BOOL := FALSE;
    Error       : BOOL := FALSE;
    ErrorID     : UDINT := 0;
    UframeData  : T_Trnsf;
END_VAR
VAR_IN_OUT
    AxesGroup   : T_StaeubliRobot;
END_VAR

```

### Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
CoordSystem	INT	Number of the user frame to compute. The user frame is stored in the plc database located on robot side.
Origin	<a href="#">T_CartesianPos [▶ 75]</a>	Coordinates of the origin of the user frame
Xdirection	<a href="#">T_CartesianPos [▶ 75]</a>	Coordinates of the point defining Ox direction
Ydirection	<a href="#">T_CartesianPos [▶ 75]</a>	Coordinates of the point defining Oxy plane of the user frame.

### Outputs

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
UframeData	<a href="#">T_Trnsf [▶ 79]</a>	Coordinates of the computed user frame

 Inputs/Outputs

Name	Type	Description
AxesGroup	T_StaebliRobot [ <a href="#">▶ 77</a> ]	Data block for a robot

### 5.1.15 VAL\_GetForwardTransformation



FUNCTION\_BLOCK VAL\_GetForwardTransformation

This functions blocks converts a joint position into a cartesian position.

#### Syntax

#### Definition:

```
FUNCTION_BLOCK VAL_GetForwardTransformation
VAR_INPUT
    Execute      : BOOL := FALSE;
    JointPosition : T_JointPos;
    ToolNumber   : UINT := 0;
    CoordSystem  : UINT := 0;
END_VAR
VAR_OUTPUT
    Busy        : BOOL := FALSE;
    Done        : BOOL := FALSE;
    Error       : BOOL := FALSE;
    ErrorID     : UDINT := 0;
    CartesianPos : T_CartesianPos;
END_VAR
VAR_IN_OUT
    AxesGroup : T_StaebliRobot;
END_VAR
```

 Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
JointPosition	T_JointPos [ <a href="#">▶ 76</a> ]	Position of each axis.
ToolNumber	UINT	Number of the Tool Center Point for which the cartesian position shall be returned. The Tool Center Point is located in a plc dedicated database on robot side
CoordSystem	UINT	Number of the user frame in which the cartesian position is returned. The user frame is stored in a plc dedicated database located on robot side.

 Outputs

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success

Name	Type	Description
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
CartesianPos	T_CartesianPos [▶ 75]	Coordinates of the computed position

 **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaeubliRobot [▶ 77]	Data block for a robot

### 5.1.16 VAL\_GetInverseTransformation



FUNCTION\_BLOCK VAL\_GetInverseTransformation

This functions blocks converts a cartesian position into a joint position. The computed joint position is computed with following rules:

- If configuration fields of the cartesian position (CS CE CW) is 1, robot's final configuration is not easily predictable. For this reason, it is not recommended to have CS CE or CW set to 1.
- If configuration fields of the cartesian position (CS CE CW) is 2, robot's final configuration is the same than this defined by JointReference parameter.
- If configuration fields of the cartesian position (CS CE CW) are 3 or 4, robot's final configuration is this of Cartesian input position.

#### Syntax

**Definition:**

```
FUNCTION_BLOCK VAL_GetInverseTransformation
VAR_INPUT
    Execute      : BOOL := FALSE;
    Position     : T_CartesianPos;
    ToolNumber   : UINT := 0;
    CoordSystem  : UINT := 0;
    JointReference : T_JointPos;
END_VAR
VAR_OUTPUT
    Busy        : BOOL := FALSE;
    Done        : BOOL := FALSE;
    Error       : BOOL := FALSE;
    ErrorID     : UDINT := 0;
    JointPos    : T_JointPos;
END_VAR
VAR_IN_OUT
    AxesGroup   : T_StaeubliRobot;
END_VAR
```

 **Inputs**

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
Position	T_CartesianPos [▶ 75]	Cartesian position to convert into a joint position

Name	Type	Description
ToolNumber	UINT	Number of the Tool Center Point used for the computation. The Tool Center Point is located in a plc dedicated database on robot side
CoordSystem	UINT	Number of the user frame to which the cartesian position relates. The user frame is stored in a plc dedicated database located on robot side.
JointReference	T_JointPos [▶ 76]	Joint position used as a reference if configuration fields of input position are 1 or 2

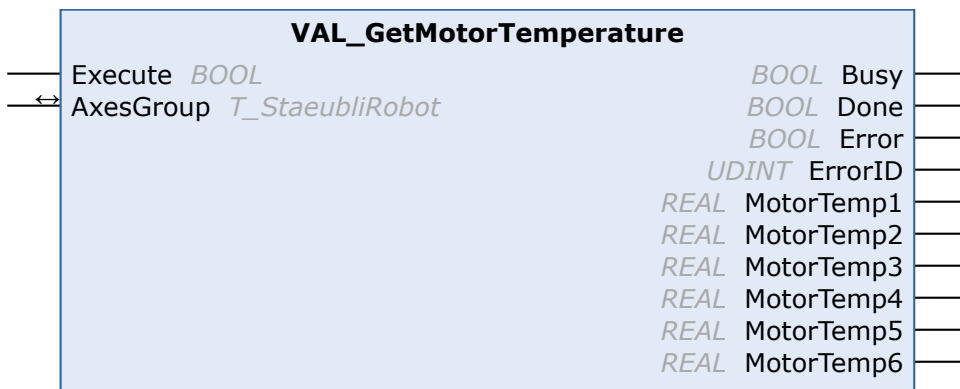
 **Outputs**

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
JointPos	T_JointPos [▶ 76]	Coordinates of the computed joint position

 **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaebliRobot [▶ 77]	Data block for a robot

### 5.1.17 VAL\_GetMotorTemperature



FUNCTION\_BLOCK VAL\_GetMotorTemperature

This function block returns the temperature of each motor of the robot. This function is available for CS9 controller ONLY.

The parameter ErrorID shows the axes for which the temperature could not be gathered and shall be interpreted as a bitfield:

Bit No	Description
0	Temperature motor 1 could not be gathered
1	Temperature motor 2 could not be gathered
2	Temperature motor 3 could not be gathered
3	Temperature motor 4 could not be gathered
4	Temperature motor 5 could not be gathered

Bit No	Description
5	Temperature motor 6 could not be gathered

## Syntax

### Definition:

```

FUNCTION_BLOCK VAL_GetMotorTemperature
VAR_INPUT
    Execute : BOOL := FALSE;
END_VAR
VAR_OUTPUT
    Busy      : BOOL := FALSE;
    Done      : BOOL := FALSE;
    Error     : BOOL := FALSE;
    ErrorID   : UDINT := 0;
    MotorTemp1 : REAL := 0;
    MotorTemp2 : REAL := 0;
    MotorTemp3 : REAL := 0;
    MotorTemp4 : REAL := 0;
    MotorTemp5 : REAL := 0;
    MotorTemp6 : REAL := 0;
END_VAR
VAR_IN_OUT
    AxesGroup : T_StaeubliRobot;
END_VAR

```

### Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution

### Outputs

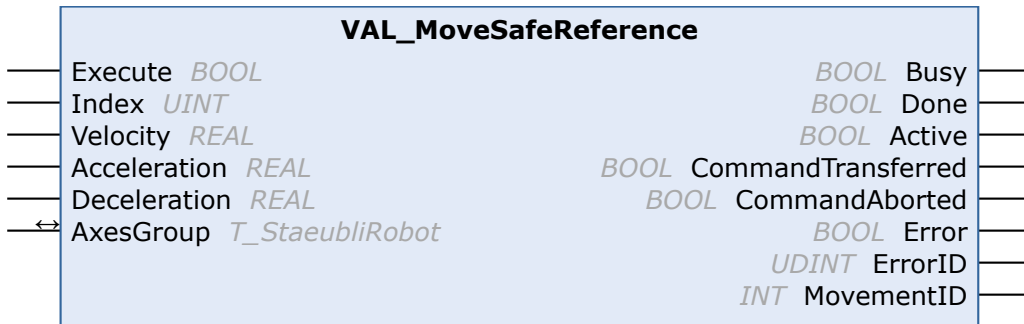
Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
MotorTemp1	REAL	Temperature of motor 1
MotorTemp2	REAL	Temperature of motor 2
MotorTemp3	REAL	Temperature of motor 3
MotorTemp4	REAL	Temperature of motor 4
MotorTemp5	REAL	Temperature of motor 5 - Not applicable for 4 axis robots
MotorTemp6	REAL	Temperature of motor 6 - Not applicable for 4 axis robots

### Inputs/Outputs

Name	Type	Description
AxesGroup	T_StaeubliRobot <a href="#">▶ 77</a>	Data block for a robot



### 5.1.18 VAL\_MoveSafeReference



FUNCTION\_BLOCK VAL\_MoveSafeReference



Applicable for CS9 controller only

This function block commands a joint interpolated movement to an end-position. The values specified for the end-position are interpreted as joint values. They represent the position of each axis in degrees.

#### Syntax

##### Definition:

```
FUNCTION_BLOCK VAL_MoveSafeReference
VAR_INPUT
    Execute      : BOOL := FALSE;
    Index        : UINT;
    Velocity     : REAL := REAL#100;
    Acceleration : REAL := REAL#100;
    Deceleration : REAL := REAL#100;
END_VAR
VAR_OUTPUT
    Busy          : BOOL := FALSE;
    Done          : BOOL := FALSE;
    Active        : BOOL := FALSE;
    CommandTransferred : BOOL := FALSE;
    CommandAborted   : BOOL := FALSE;
    Error         : BOOL := FALSE;
    ErrorID       : UDINT := 0;
    MovementID    : INT := -1;
END_VAR
VAR_IN_OUT
    AxesGroup    : T_StaebliRobot;
END_VAR
```

#### 📌 Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
Index	UINT	Number of the safe position to reach
Velocity	REAL	Joint velocity. Percentage of the nominal speed of the robot
Acceleration	REAL	Joint acceleration. Percentage of the nominal acceleration of the robot
Deceleration	REAL	Joint deceleration. Percentage of the nominal deceleration of the robot

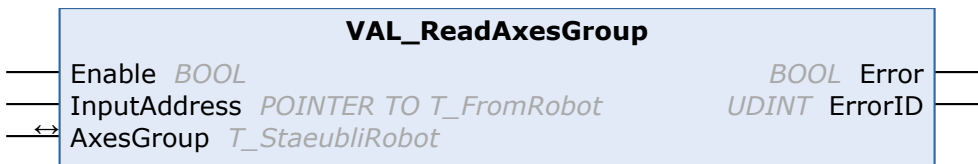
 **Outputs**

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Active	BOOL	Set when commanded movement is currently executed by the robot
CommandTransferred	BOOL	Commanded motion is successfully buffered inside robot controler
CommandAborted	BOOL	Command is aborted by another command
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
MovementID	INT	Identifier for this commanded movement

 **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaeubliRobot <a href="#">[▶ 77]</a>	Data block for a robot

### 5.1.19 VAL\_ReadAxesGroup



FUNCTION\_BLOCK VAL\_ReadAxesGroup

This function block initializes the values of the interface data block by reading the inputs of the communication interface. It provides the data for the other blocks of the library. It must be called as the first network of a robot's program.

**Syntax**

**Definition:**

```
FUNCTION_BLOCK VAL_ReadAxesGroup
VAR_INPUT
    Enable      : BOOL := FALSE;
    InputAddress : POINTER TO T_FromRobot;
END_VAR
VAR_OUTPUT
    Error      : BOOL := FALSE;
    ErrorID    : UDINT := 0;
END_VAR
VAR_IN_OUT
    AxesGroup  : T_StaeubliRobot;
END_VAR
```

 **Inputs**

Name	Type	Description
Enable	BOOL	TRUE = Content of the FB is executed

Name	Type	Description
InputAddress	POINTER TO T_FromRobot [▶ 73]	Base address of the Inputs within the IO interface with a robot. Use 'ADR' operator for setting this input of the function block. Typical syntax is ADR(%IWxx) where xx depends on the hardware configuration of the PLC.

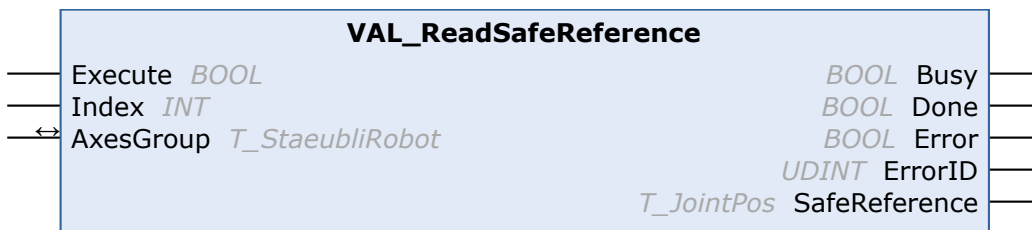
 **Outputs**

Name	Type	Description
Error	BOOL	Error flag
ErrorID	UDINT	Error code

 /  **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaeubliRobot [▶ 77]	Data block for a robot

### 5.1.20 VAL\_ReadSafeReference



FUNCTION\_BLOCK VAL\_ReadSafeReference



Applicable for CS9 controller only

This function block returns the coordinates of a the safe reference position. For further details about CS9 safety features, please consult 'CS9 Safety Manual'

**Syntax**

**Definition:**

```
FUNCTION_BLOCK VAL_ReadSafeReference
VAR_INPUT
    Execute    : BOOL := FALSE;
    Index      : INT  := -1;
END_VAR
VAR_OUTPUT
    Busy       : BOOL := FALSE;
    Done       : BOOL := FALSE;
    Error      : BOOL := FALSE;
    ErrorID    : UDINT := 0;
    SafeReference : T_JointPos;
END_VAR
VAR_IN_OUT
    AxesGroup  : T_StaeubliRobot;
END_VAR
```

 **Inputs**

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution

Name	Type	Description
Index	INT	Number of the safe reference. Range [1..2]

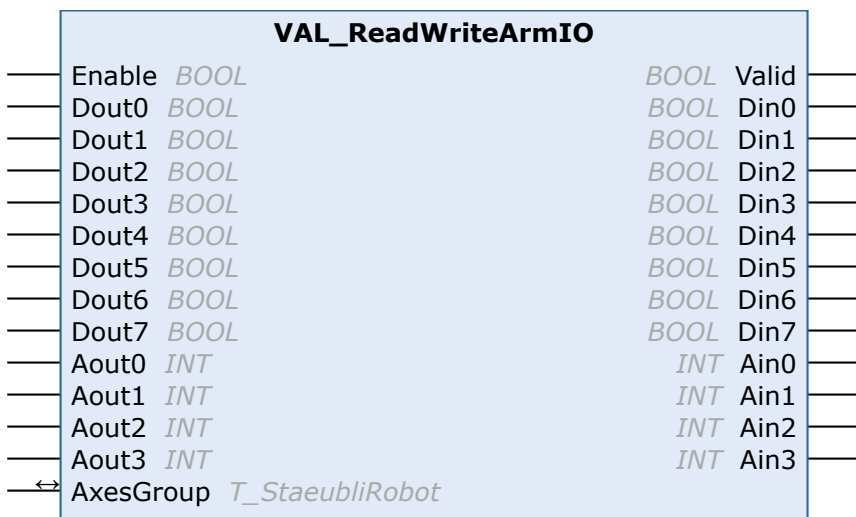
 **Outputs**

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
SafeReference	T_JointPos [▶ 76]	Coordinates of the selected safe reference position

 /  **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaeubliRobot [▶ 77]	Data block for a robot

### 5.1.21 VAL\_ReadWriteArmIO



FUNCTION\_BLOCK VAL\_ReadWriteArmIO

**Syntax**

**Definition:**

```
FUNCTION_BLOCK VAL_ReadWriteArmIO
VAR_INPUT
  Enable : BOOL := FALSE;
  Dout0 : BOOL := FALSE;
  Dout1 : BOOL := FALSE;
  Dout2 : BOOL := FALSE;
  Dout3 : BOOL := FALSE;
  Dout4 : BOOL := FALSE;
  Dout5 : BOOL := FALSE;
  Dout6 : BOOL := FALSE;
  Dout7 : BOOL := FALSE;
  Aout0 : INT := 0;
  Aout1 : INT := 0;
  Aout2 : INT := 0;
  Aout3 : INT := 0;
END_VAR
VAR_OUTPUT
```

```

Valid : BOOL := FALSE;
Din0  : BOOL := FALSE;
Din1  : BOOL := FALSE;
Din2  : BOOL := FALSE;
Din3  : BOOL := FALSE;
Din4  : BOOL := FALSE;
Din5  : BOOL := FALSE;
Din6  : BOOL := FALSE;
Din7  : BOOL := FALSE;
Ain0  : INT := 0;
Ain1  : INT := 0;
Ain2  : INT := 0;
Ain3  : INT := 0;
END_VAR
VAR_IN_OUT
  AxesGroup : T_StaeubliRobot;
END_VAR

```

 **Inputs**

Name	Type	Description
Enable	BOOL	Function block content is executed as long as this input is set TRUE
Dout0	BOOL	Command state of digital output channel 0 of the ArmIO board
Dout1	BOOL	Command state of digital output channel 1 of the ArmIO board
Dout2	BOOL	Command state of digital output channel 2 of the ArmIO board
Dout3	BOOL	Command state of digital output channel 3 of the ArmIO board
Dout4	BOOL	Command state of digital output channel 4 of the ArmIO board
Dout5	BOOL	Command state of digital output channel 5 of the ArmIO board
Dout6	BOOL	Command state of digital output channel 6 of the ArmIO board
Dout7	BOOL	Command state of digital output channel 7 of the ArmIO board
Aout0	INT	Command state of analog output channel 0 of the armIO board. Unit is millivolt
Aout1	INT	Command state of analog output channel 1 of the armIO board. Unit is millivolt
Aout2	INT	Command state of analog output channel 2 of the armIO board. Unit is millivolt
Aout3	INT	Command state of analog output channel 3 of the armIO board. Unit is millivolt

 **Outputs**

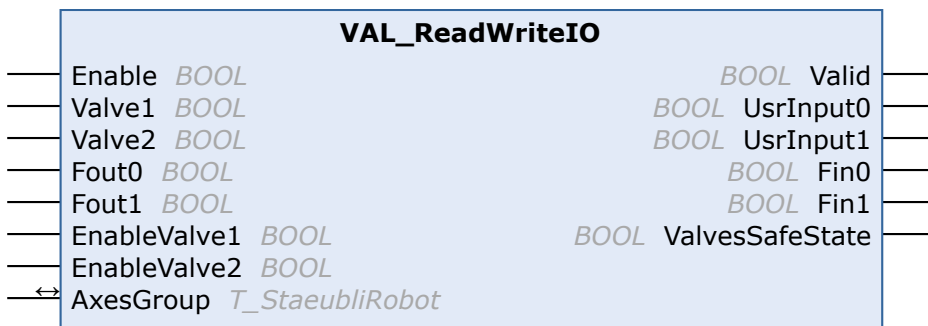
Name	Type	Description
Valid	BOOL	Set when function block is executing. Reset when Done or Error is set
Din0	BOOL	Command state of digital input channel 0 of the ArmIO board
Din1	BOOL	Command state of digital input channel 1 of the ArmIO board
Din2	BOOL	Command state of digital input channel 2 of the ArmIO board

Name	Type	Description
Din3	BOOL	Command state of digital input channel 3 of the ArmIO board
Din4	BOOL	Command state of digital input channel 4 of the ArmIO board
Din5	BOOL	Command state of digital input channel 5 of the ArmIO board
Din6	BOOL	Command state of digital input channel 6 of the ArmIO board
Din7	BOOL	Command state of digital input channel 7 of the ArmIO board
Ain0	INT	Command state of analog input channel 0 of the armIO board. Unit is millivolt
Ain1	INT	Command state of analog input channel 1 of the armIO board. Unit is millivolt
Ain2	INT	Command state of analog input channel 2 of the armIO board. Unit is millivolt
Ain3	INT	Command state of analog input channel 3 of the armIO board. Unit is millivolt

 Inputs/Outputs

Name	Type	Description
AxesGroup	T_StaubliRobot [▶ 77]	Data block for a robot

### 5.1.22 VAL\_ReadWriteIO



FUNCTION\_BLOCK VAL\_ReadWriteIO

Update the digital outputs of the controller with values passed as parameter. Return the state of the digital inputs available on the controller.



The mentioned inputs Valve1 and Valve2 are not available with TS Scara robot (CS8C controller).



Please be aware of the following signal combination for CS9 controller controlling robot equipped with 5-3 ways solenoid valves

Valvexx	EnableValvexx	Air flow on outlet (robot forearm)
0 or 1	0	NO air
0	1	Axx
1	1	Bxx

<b>Valvexx</b>	<b>EnableValvexx</b>	<b>Air flow on outlet (robot forearm)</b>
Comment: xx = 1 or 2 depending on robot model		

**Syntax**

**Definition:**

```

FUNCTION_BLOCK VAL_ReadWriteIO
VAR_INPUT
    Enable      : BOOL := FALSE;
    Valve1     : BOOL := FALSE;
    Valve2     : BOOL := FALSE;
    Fout0      : BOOL := FALSE;
    Fout1      : BOOL := FALSE;
    EnableValve1 : BOOL := FALSE;
    EnableValve2 : BOOL := FALSE;
END_VAR
VAR_OUTPUT
    Valid      : BOOL := FALSE;
    UsrInput0  : BOOL := FALSE;
    UsrInput1  : BOOL := FALSE;
    Fin0       : BOOL := FALSE;
    Fin1       : BOOL := FALSE;
    ValvesSafeState : BOOL := FALSE;
END_VAR
VAR_IN_OUT
    AxesGroup : T_StaeubliRobot;
END_VAR
    
```

 **Inputs**

Name	Type	Description
Enable	BOOL	Function block content is executed as long as this input is set TRUE
Valve1	BOOL	Command the state of the internal valve connected to A1&B1 outlets on the forearm of 6 axis robot
Valve2	BOOL	Command the state of the internal valve connected to A2&B2 outlets on the forearm of 6 axis robot
Fout0	BOOL	Command Fast output provided on J111-9 / J111-12 connector for CS8C controller OR J212-4 / J212-9 connector for CS9 controller
Fout1	BOOL	Command Fast output provided on J111-9 / J111-12 connector for CS8C controller OR J212-1 / J212-5 connector for CS9 controller
EnableValve1	BOOL	Apply to CS9 with 6 axes robot equipped with 5-3 ways solenoid valves ONLY. TRUE=Valve1 can be operated. FALSE=Valve1 is in (closed) middle Position
EnableValve2	BOOL	Apply to CS9 with 6 axes robot equipped with 5-3 ways solenoid valves ONLY. TRUE=Valve2 can be operated. FALSE=Valve2 is in (closed) middle Position

 **Outputs**

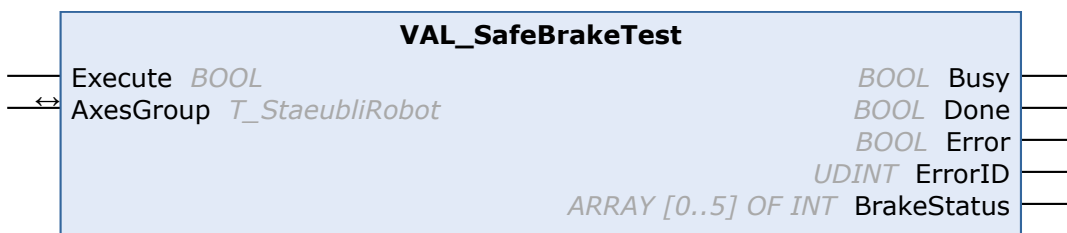
Name	Type	Description
Valid	BOOL	Set when function block is executing. Reset when Done or Error is set
UsrInput0	BOOL	State of the digital user input 0 provided on J109-11 / J109-30 connector of the CS8C controller
UsrInput1	BOOL	State of the digital user input 1 provided on J109-16 / J109-35 connector of the CS8C controller

Name	Type	Description
Fin0	BOOL	State of the fast digital input provided on J111-2 / J111-7 connector of the CS8C controller
Fin1	BOOL	State of the fast digital input provided on J111-3 / J111-8 connector of the CS8C controller
ValvesSafeState	BOOL	CS9 controlling 5-3 ways valves ONLY. The valves are in Safe State. Valve position may not match the commanded state. Refer to CS9 safety manual for details

 **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaebliRobot [▶ 77]	Data block for a robot

### 5.1.23 VAL\_SafeBrakeTest



FUNCTION\_BLOCK VAL\_SafeBrakeTest



Applicable for CS9 controller only

This function blocks commands a test of all the brakes of the connected robot.

For further details about CS9 safety features, please consult 'CS9 Safety Manual'

#### Syntax

##### Definition:

```
FUNCTION_BLOCK VAL_SafeBrakeTest
VAR_INPUT
    Execute : BOOL := FALSE;
END_VAR
VAR_OUTPUT
    Busy      : BOOL := FALSE;
    Done      : BOOL := FALSE;
    Error     : BOOL := FALSE;
    ErrorID   : UDINT := 0;
    BrakeStatus : ARRAY [0..5] OF INT;
END_VAR
VAR_IN_OUT
    AxesGroup : T_StaebliRobot;
END_VAR
```

 **Inputs**

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution



 **Outputs**

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
BrakeStatus	ARRAY [0..5] OF INT	Result of the brake test for each axis

 **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaeubliRobot [ <a href="#">▶ 77</a> ]	Data block for a robot

### 5.1.24 VAL\_SetMovementID



FUNCTION\_BLOCK VAL\_SetMovementID

This instruction changes the movement ID for the next commanded movement. It is useful to arrange so that the same path always uses the same movement ID values.

After execution of MC\_GroupStop(), the movement ID is automatically reset to 0.

After executing this function block, the relation between the movement ID set and the ID of the movement in progress may become uncertain:

Several commanded movements may then have the same movement id. This function block should therefore not give a value that is also the movement ID of a pending move command.

To finish, this is highly recommended to execute this function when robot is NOT moving

#### Syntax

##### Definition:

```
FUNCTION_BLOCK VAL_SetMovementID
VAR_INPUT
    Execute      : BOOL := FALSE;
    MovementID  : INT  := -1;
END_VAR
VAR_OUTPUT
    Busy        : BOOL := FALSE;
    Done        : BOOL := FALSE;
    Error       : BOOL := FALSE;
    ErrorID     : UDINT := 0;
END_VAR
VAR_IN_OUT
    AxesGroup   : T_StaeubliRobot;
END_VAR
```

 **Inputs**

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution

Name	Type	Description
MovementID	INT	Value for setting movement ID on robot side

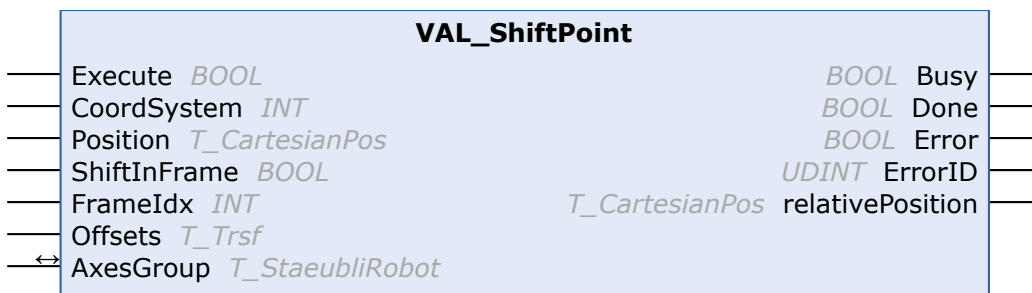
**🔌 Outputs**

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	s when function block has terminated with error
ErrorID	UDINT	Error code

**🔌 / 🔌 Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaebliRobot [▶ 77]	Data block for a robot

### 5.1.25 VAL\_ShiftPoint



FUNCTION\_BLOCK VAL\_ShiftPoint

This function block computes a new cartesian position by applying a geometrical transformation. The geometrical transformation specified by 'Offsets' parameter can be applied relative to different coordinate systems :

- User frames ( Equivalent to native compose() function)
- Tool ( Equivalent to native appro() function )

**Syntax**

**Definition:**

```

FUNCTION_BLOCK VAL_ShiftPoint
VAR_INPUT
    Execute      : BOOL := FALSE;
    CoordSystem  : INT := -1;
    Position     : T_CartesianPos;
    ShiftInFrame : BOOL := FALSE;
    FrameIdx     : INT := -1;
    Offsets      : T_Trnsf := STRUCT(X := 0, Y := 0, Z := 0, RX := 0, RY := 0, RZ := 0);
END_VAR
VAR_OUTPUT
    Busy          : BOOL := FALSE;
    Done          : BOOL := FALSE;
    Error         : BOOL := FALSE;
    ErrorID       : UDINT := 0;
    relativePosition : T_CartesianPos;
END_VAR
VAR_IN_OUT
    AxesGroup    : T_StaebliRobot;
END_VAR
    
```

 Inputs

Name	Type	Description
Execute	BOOL	Rising edge triggers function execution
CoordSystem	INT	Index of the user frame (starting with 0)
Position	T_CartesianPos [ <a href="#">▶ 75</a> ]	Absolute end positions for each dimension in the specified coordinate system.
ShiftInFrame	BOOL	True=Position will be shifted in specified user frame. False=shift in according to orientation of the Position parameter
FrameIdx	INT	Index of the user frame in which Position will be shifted - Applicable only if ShiftInFrame is true
Offsets	T_Trnsf [ <a href="#">▶ 79</a> ]	Offsets in all directions to apply to the Position

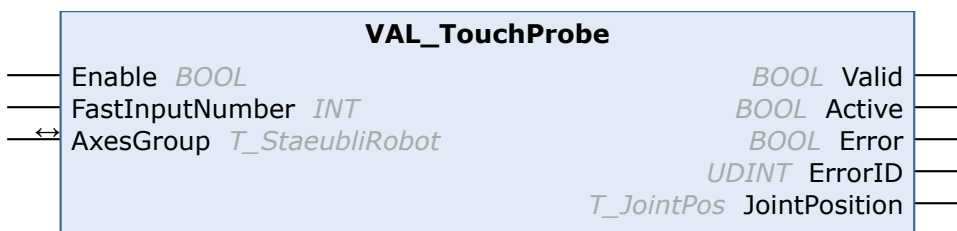
 Outputs

Name	Type	Description
Busy	BOOL	Set when function block is executing. Reset when Done or Error is set
Done	BOOL	This output is set when function block has terminated with success
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
relativePosition	T_CartesianPos [ <a href="#">▶ 75</a> ]	New Position after the offsets have been applied

 /  Inputs/Outputs

Name	Type	Description
AxesGroup	T_StaeubliRobot [ <a href="#">▶ 77</a> ]	Data block for a robot

### 5.1.26 VAL\_TouchProbe



FUNCTION\_BLOCK VAL\_TouchProbe

This function block enables high accurate ‘on the fly’ capture of the robot position. A sensor shall be connected to a dedicated fast digital input of the robot controller. The position capture occurs on a rising edge of this dedicated sensor.

Fast input number	Description	Wiring CS8C	Wiring CS9
0	Fast input 0	J111-2(+)/ J111-7(-)	J212-2(+)/ J212-7 (-)
1	Fast input 1	J111-3(+)/ J111-8(-)	J212-3(+)/ J212-8 (-)

**Syntax**

**Definition:**

```

FUNCTION_BLOCK VAL_TouchProbe
VAR_INPUT
    Enable          : BOOL := FALSE;
    FastInputNumber : INT  := -1;
END_VAR
VAR_OUTPUT
    Valid          : BOOL := FALSE;
    Active         : BOOL := FALSE;
    Error          : BOOL := FALSE;
    ErrorID       : UDINT := 0;
    JointPosition : T_JointPos;
END_VAR
VAR_IN_OUT
    AxesGroup : T_StaeubliRobot;
END_VAR
    
```

 **Inputs**

Name	Type	Description
Enable	BOOL	Rising edge triggers function execution
FastInputNumber	INT	Index of the fast input which is used to capture robot position.

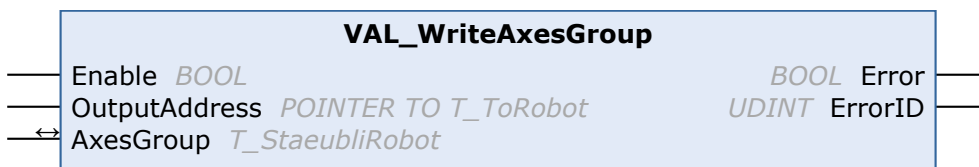
 **Outputs**

Name	Type	Description
Valid	BOOL	This output is set when function block has terminated with success
Active	BOOL	TRUE = Fast robot position capture is activated on robot side. FALSE = Fast robot position capture is disabled.
Error	BOOL	This output is set when function block has terminated with error
ErrorID	UDINT	Error code
JointPosition	T_JointPos <a href="#">▶ 76</a>	Position of each robot axis when rising edge occurred on input

 **Inputs/Outputs**

Name	Type	Description
AxesGroup	T_StaeubliRobot <a href="#">▶ 77</a>	Data block for a robot

### 5.1.27 VAL\_WriteAxesGroup



```
FUNCTION_BLOCK VAL_WriteAxesGroup
```

**Syntax**

**Definition:**

```

FUNCTION_BLOCK VAL_WriteAxesGroup
VAR_INPUT
    Enable          : BOOL := FALSE;
    OutputAddress   : POINTER TO T_ToRobot;
END_VAR
VAR_OUTPUT
    
```

```

Error      : BOOL := FALSE;
ErrorID    : UDINT := 0;
END_VAR
VAR_IN_OUT
  AxesGroup : T_StaeubliRobot;
END_VAR
    
```

 **Inputs**

Name	Type	Description
Enable	BOOL	TRUE = Content of the FB is executed
OutputAddress	POINTER TO <u>T_ToRobot</u> [▶ <u>73</u> ]	Base address of the Outputs within the IO interface with a robot. Use 'ADR' operator for setting this input of the function block. Typical syntax is ADR(%QWxx) where xx depends on the hardware configuration of the PLC.

 **Outputs**

Name	Type	Description
Error	BOOL	Error flag
ErrorID	UDINT	Error code

 /  **Inputs/Outputs**

Name	Type	Description
AxesGroup	<u>T_StaeubliRobot</u> [▶ <u>77</u> ]	Data block for a robot

## 5.2 Functions

### 5.2.1 VAL\_DefineCartesianPos



FUNCTION VAL\_DefineCartesianPos : T\_CartesianPos [▶ 75]

This function might be used to initialize a cartesian position. It provides a way to have a more compact code. The following code below

```

myPoint := VAL_DefineCartesianPos(10,-13.125,115.10,45.5,25.3,-90.25,2,2,2);
    
```

is equivalent to

```

myPoint.X := 10;
myPoint.Y := -13.125;
myPoint.Z := 115.10;
myPoint.RX := 45.5;
myPoint.RY := 25.3;
myPoint.RZ := -90.25;
    
```

```
myPoint.CS := 2;
myPoint.CE := 2;
myPoint.CW := 2;
```

**Syntax**

**Definition:**

```
FUNCTION VAL_DefineCartesianPos : T_CartesianPos
VAR_INPUT
  X   : REAL := 0;
  Y   : REAL := 0;
  Z   : REAL := 0;
  RX  : REAL := 0;
  RY  : REAL := 0;
  RZ  : REAL := 0;
  CS  : UINT := 2;
  CE  : UINT := 2;
  CW  : UINT := 2;
END_VAR
```

 **Inputs**

Name	Type	Description
X	REAL	Value for X coordinate
Y	REAL	Value for Y coordinate
Z	REAL	Value for Z coordinate
RX	REAL	Value for RX coordinate
RY	REAL	Value for RY coordinate
RZ	REAL	Value for RZ coordinate
CS	UINT	Value for shoulder configuration (1=> sfree; 2=> ssame; 3=> righty; 4=> lefty)
CE	UINT	Value for elbow configuration (1=> sfree; 2=> ssame; 3=> righty; 4=> lefty)
CW	UINT	Value for wrist configuration (1=> sfree; 2=> ssame; 3=> righty; 4=> lefty)

 **Return value**

Name	Type	Description
VAL_DefineCartesianPos	T_CartesianPos [ <a href="#">▶ 75</a> ]	

**5.2.2 VAL\_DefineJointPos**



```
FUNCTION VAL_DefineJointPos : T_JointPos [▶ 76]
```

This function might be used to initialize a joint position. It provides a way to have a more compact code. The following below

```
myJoint := VAL_DefineJointPos(10, -13.125, 115.10, 45.5, 25.3, -90.25);
```

is equivalent to

```
myJoint.J1 := 10;
myJoint.J2 := -13.125;
myJoint.J3 := 115.10;
myJoint.J4 := 45.5;
myJoint.J5 := 25.3;
myJoint.J6 := -90.25;
```

**Syntax**

**Definition:**

```
FUNCTION VAL_DefineJointPos : T_JointPos
VAR_INPUT
  J1 : REAL := 0;
  J2 : REAL := 0;
  J3 : REAL := 0;
  J4 : REAL := 0;
  J5 : REAL := 0;
  J6 : REAL := 0;
END_VAR
```

 **Inputs**

Name	Type	Description
J1	REAL	Coordinate for axis 1
J2	REAL	Coordinate for axis 2
J3	REAL	Coordinate for axis 3
J4	REAL	Coordinate for axis 4
J5	REAL	Coordinate for axis 5 - not applicable for 4 axis robots-
J6	REAL	Coordinate for axis 6 - not applicable for 4 axis robots-

 **Return value**

Name	Type	Description
VAL_DefineJointPos	T_JointPos [ <a href="#">▶ 76</a> ]	

**5.2.3 VAL\_DefineTrsf**



```
FUNCTION VAL_DefineTrsf : T_Trsl [▶ 79]
```

This function might be used to initialize a T\_Trsl variable. It provides a way to have a more compact code. The following code below:

```
myTrsf := VAL_DefineTrsf(10,-13.125,115.10,45.5,25.3,-90.25);
```

is equivalent to

```
myTrsf.X := 10;
myTrsf.Y := -13.125;
myTrsf.Z := 115.10;
myPoint.RX := 45.5;
myTrsf.RY := 25.3;
myTrsf.RZ := -90.25;
```

**Syntax**

**Definition:**

```
FUNCTION VAL_DefineTrsf : T_Trsf
VAR_INPUT
  X   : REAL := 0;
  Y   : REAL := 0;
  Z   : REAL := 0;
  RX  : REAL := 0;
  RY  : REAL := 0;
  RZ  : REAL := 0;
END_VAR
```

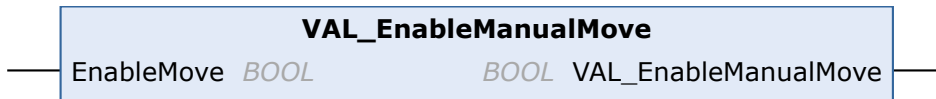
 **Inputs**

Name	Type	Description
X	REAL	Value for X coordinate
Y	REAL	Value for Y coordinate
Z	REAL	Value for Z coordinate
RX	REAL	Value for RX coordinate
RY	REAL	Value for RY coordinate
RZ	REAL	Value for RZ coordinate

 **Return value**

Name	Type	Description
VAL_DefineTrsf	T_Trsf [ <a href="#">▶ 79</a> ]	

**5.2.4 VAL\_EnableManualMove**



FUNCTION VAL\_EnableManualMove : BOOL

This function enables movement of the robot when manual mode is selected. It provides the hold-to-run function described hereafter:

Although robot speed is limited and user shall activate the 3-position switch, an additional action is required as per ISO 10218-1:2011 - Robots and robotic devices -- Safety requirements for industrial robots - Chapter 5.7.3:

- Manual control of the robot from inside the safeguarded space shall be performed with a reduced speed in conjunction with either of the following:
  - hold-to-run controls in conjunction with an enabling device in accordance with 5.8 [Pendant controls]
  - for program verification only, a start/stop control in conjunction with an enabling device in accordance with 5.8 [Pendant controls]
- This function is effective only when Stäubli Teach Pendant is removed and replaced by the delivered shorting plug.
- The controller must be configured to have user-supplied teach pendant in use (file iomap.cfx on controller).
- Manual operation mode must be selected from user-supplied teach pendant

**Syntax**

**Definition:**



```
FUNCTION VAL_EnableManualMove : BOOL
VAR_INPUT
    EnableMove : BOOL := FALSE;
END_VAR
```

**Inputs**

Name	Type	Description
EnableMove	BOOL	If this input is TRUE, the robot movement are enabled. When released, the robot stops on the programmed trajectory

**Return value**

Name	Type	Description
VAL_EnableManualMove	BOOL	

## 5.3 Types

### 5.3.1 Internal

#### 5.3.1.1 T\_FromRobot

```
TYPE T_FromRobot : STRUCT
```

Structure defining the inputs (PLC side) for the interface with a robot. This datatype is for internal use only. It is deliberately not documented.

#### 5.3.1.2 T\_ToRobot

```
TYPE T_ToRobot : STRUCT
```

Structure defining the outputs (PLC side) for the interface with a robot. This datatype is for internal use only. It is deliberately not documented.

### 5.3.2 MC\_TransitionParameter

```
TYPE MC_TransitionParameter : STRUCT
```

**Syntax**

**Definition:**

```
TYPE MC_TransitionParameter :
STRUCT
    leave : REAL;
    reach : REAL;
END_STRUCT
END_TYPE
```

**Parameters**

Name	Type	Description
leave	REAL	Distance from the destination point at which the nominal trajectory is left.
reach	REAL	Distance from the destination point at which the nominal trajectory is joined again.

### 5.3.3 T\_CS9SftyCmd

TYPE T\_CS9SftyCmd : STRUCT

Structure defining information related to CS9 safety featuresApplicable for CS9 controller only.For futher details about CS9 safety features, please consult 'CS9 Safety Manual'Restart control is the safety function to maintain the robot in safe state after a safety stop, or after machine start-up, until an explicit restart acknowledge (manual reset) signal is received.

#### Syntax

##### Definition:

```
TYPE T_CS9SftyCmd :
STRUCT
  RestartSafety      : BOOL := FALSE;
  ResetReferencing  : BOOL := FALSE;
  ForceReferencingCheck : BOOL := FALSE;
  SafeRefSensor     : BOOL := FALSE;
END_STRUCT
END_TYPE
```

#### Parameters

Name	Type	Initial	Description
RestartSafety	BOOL	FALSE	Command a safety restart/acknowledgment of the CS9 internal safety PLC program. Safety acknowledge can be disabled on the robot side, when restart control is fully implemented within an external PLC program
ResetReferencing	BOOL	FALSE	Effective when robot power is OFF. Reset the safe referencing procedure. It shall be completely performed again.
ForceReferencingCheck	BOOL	FALSE	When set, robot moves at slow speed only. A movement to second referencing position is required.
SafeRefSensor	BOOL	FALSE	State of the safe referencing sensor

### 5.3.4 T\_CS9SftyFbk

TYPE T\_CS9SftyFbk : STRUCT

Structure defining information related to CS9 safety featuresApplicable for CS9 controller only.For futher details about CS9 safety features, please consult 'CS9 Safety Manual'Restart control is the safety function to maintain the robot in safe state after a safety stop, or after machine start-up, until an explicit restart acknowledge (manual reset) signal is received.

#### Syntax

##### Definition:

```
TYPE T_CS9SftyFbk :
STRUCT
  WaitForRestart    : BOOL := FALSE;
  BrakeTestWarning  : BOOL := FALSE;
  BrakeTestTimeout  : BOOL := FALSE;
  SafeReferencingStatus : UINT := 7;
END_STRUCT
END_TYPE
```

#### Parameters

Name	Type	Initial	Description
WaitForRestart	BOOL	FALSE	The CS9 internal safety PLC program is waiting for a safety restart/acknowledgment. This feature can be disabled

Name	Type	Initial	Description
BrakeTestWarning	BOOL	FALSE	TRUE= Time interval for performing brake test is about to elapse. Robot brakes shall be tested as soon as possible
BrakeTestTimeout	BOOL	FALSE	TRUE= Time interval for performing brake test is up. Robot will move wil low speed until Brake test procedure is successfully executed.
SafeReferencingStatus	UINT	7	Self explaining...

### 5.3.5 T\_CartesianPos

TYPE T\_CartesianPos : STRUCT

This data type represents a cartesian position.A Cartesian position defines the position and orientation of a point in the environment of the robot.

#### Syntax

#### Definition:

```

TYPE T_CartesianPos :
STRUCT
  X   : REAL := 0;
  Y   : REAL := 0;
  Z   : REAL := 0;
  RX  : REAL := 0;
  RY  : REAL := 0;
  RZ  : REAL := 0;
  CS  : UINT := 2;
  CE  : UINT := 2;
  CW  : UINT := 2;
END_STRUCT
END_TYPE
    
```

#### Parameters

Name	Type	Initial	Description
X	REAL	0	X coordinate
Y	REAL	0	Y coordinate
Z	REAL	0	Z coordinate
RX	REAL	0	RX coordinate
RY	REAL	0	RY coordinate
RZ	REAL	0	RZ coordinate
CS	UINT	2	Configuration shoulder (1=> sfree; 2=> ssame; 3=> righty; 4=> lefty)
CE	UINT	2	Configuration elbow (1=> efree; 2=> esame; 3=> epositive; 4=> enegative)
CW	UINT	2	Configuration wrist (1=> wfree; 2=> wsame; 3=> wpositive; 4=> wnegative)

### 5.3.6 T\_Command

TYPE T\_Command : STRUCT

Structure defining the command information defined in robot data block.

#### Syntax

#### Definition:

```

TYPE T_Command :
STRUCT
  EnableProfiAlarms : BOOL := FALSE;
  EnableVerbose     : BOOL := FALSE;
  OverrideCmd       : UINT := 0;
  OperationModeCmd  : UINT := 0;
  ToolCmd           : UINT := 0;
  CoordSystemCmd    : UINT := 0;
END_STRUCT
    
```

```

    CS9Safety      : T_CS9SftyCmd;
    LifebitPeriod  : TIME := TIME#200ms;
END_STRUCT
END_TYPE

```

**Parameters**

Name	Type	Initial	Description
EnableProfiAlarms	BOOL	FALSE	TRUE=Error messages are reported on Profinet network.
EnableVerbose	BOOL	FALSE	TRUE=Enable tracing activity of the server. It is strongly recommended to enable trace for debugging purpose only.
OverrideCmd	UINT	0	Commanded Override [1..100] (monitor speed)
OperationModeCmd	UINT	0	Commanded operation mode 0= invalid , 1= Manu, 3=Auto, 4 remote(extAut)
ToolCmd	UINT	0	Select the Tool used for movement. Starting with 0
CoordSystemCmd	UINT	0	Select the coordinate system used for movement. Starting with 0
CS9Safety	T_CS9SftyCmd [▶ 74]		CS9 Only - Safety features
LifebitPeriod	TIME	TIME#200ms	Select the period of the internal lifebit (100ms<Period<1000ms)

### 5.3.7 T\_JointPos

TYPE T\_JointPos : STRUCT

This data type represents a joint position. A joint position is defined by the angular position of each revolute axis and the linear position of each linear axis. The fields are expressed in degrees for the rotary axes and in millimeter or inches for the linear axes. The origin of each axis is defined according to the type of robot used.

**Syntax**

**Definition:**

```

TYPE T_JointPos :
STRUCT
    J1 : REAL := 0;
    J2 : REAL := 0;
    J3 : REAL := 0;
    J4 : REAL := 0;
    J5 : REAL := 0;
    J6 : REAL := 0;
END_STRUCT
END_TYPE

```

**Parameters**

Name	Type	Initial	Description
J1	REAL	0	Position of axis 1
J2	REAL	0	Position of axis 2
J3	REAL	0	Position of axis 3
J4	REAL	0	Position of axis 4
J5	REAL	0	Position of axis 5 if applicable. For 4 axis robots, this value is 0
J6	REAL	0	Position of axis 6 if applicable. For 4 axis robots, this value is 0

### 5.3.8 T\_Mdesc

TYPE T\_Mdesc : STRUCT

This data type describes a motion descriptor. A motion descriptor is a variable used to define :

- The dynamic parameters of a commanded movement (Speed, Acceleration, Deceleration)
- The geometry of the trajectory at way point (Blending distances)

**Syntax**

**Definition:**

```

TYPE T_Mdesc :
STRUCT
  Accel      : INT := 0;
  Vel        : INT := 0;
  Decel      : INT := 0;
  Tvel       : REAL := 0;
  Rvel       : REAL := 0;
  Leave      : REAL := 0;
  Reach      : REAL := 0;
  eBlendingType : eBlending := 0;
END_STRUCT
END_TYPE
    
```

**Parameters**

Name	Type	Initial	Description
Accel	INT	0	Maximum permitted joint acceleration as a % of the nominal acceleration of the robot.
Vel	INT	0	Maximum permitted joint speed as a % of the nominal speed of the robot.
Decel	INT	0	Maximum permitted joint deceleration as a % of the nominal deceleration of the robot.
Tvel	REAL	0	Maximum permitted translational speed of the tool center point, in mm/s or inches/s depending on the unit of length defined in robot's controller
Rvel	REAL	0	Maximum permitted rotational speed of the tool center point, in degrees per second.
Leave	REAL	0	Distance from the destination point at which the nominal trajectory is left.
Reach	REAL	0	Distance from the destination point at which the nominal trajectory is joined again.
eBlendingType	eBlending <a href="#">[► 80]</a>	0	Blending type 0: NotDefined, 1: Off, 2: Joint, 3: Cartesian

**5.3.9 T\_StaeubliRobot**

TYPE T\_StaeubliRobot : STRUCT

Data structure for server commands/status

**Syntax**

**Definition:**

```

TYPE T_StaeubliRobot :
STRUCT
  Status      : T_Status;
  Command     : T_Command;
END_STRUCT
END_TYPE
    
```

**Parameters**

Name	Type
Status	T_Status <a href="#">[► 78]</a>

Name	Type
Command	T_Command [▶ 75]

### 5.3.10 T\_Status

TYPE T\_Status : STRUCT

Structure defining the status information defined in robot data block

#### Syntax

#### Definition:

```

TYPE T_Status :
STRUCT
  Initialized           : BOOL;
  Online               : BOOL;
  ErrorPending         : BOOL;
  IsMoving             : BOOL;
  EStopActive          : BOOL;
  DummyPlug            : BOOL;
  ExternalWMS          : BOOL;
  ExternalMCP          : BOOL;
  ExternalMcp_Wms      : BOOL;
  ActualSpeed          : REAL;
  ActualOverride       : REAL;
  ActualErrorNumber    : UINT;
  ActualOperationMode : UINT;
  ActualErrorID        : T_PendingErrors;
  CartesianPos         : T_CartesianPos;
  JointPos             : T_JointPos;
  JointForces          : ARRAY [0..5] OF REAL;
  ActualCoordSystem    : UINT := 0;
  ActualTool           : UINT := 0;
  RobotStateMachine   : UINT;
  MovementID          : INT;
  MovementProgress     : INT;
  RobotModel           : STRING(10);
  ControllerModel      : UINT;
  CS9Safety            : T_CS9SftyFbk;
  Heartbeat            : UINT;
  ServerMajorVersion   : UINT;
  ServerMinorVersion   : UINT;
  ServerEdit           : UINT;
  ClientMajorVersion   : UINT;
  ClientMinorVersion   : UINT;
  ClientEdit           : UINT;
END_STRUCT
END_TYPE

```

#### Parameters

Name	Type	Initial	Description
Initialized	BOOL		True = The library is initialized
Online	BOOL		True = robot can receive commands
ErrorPending	BOOL		True = an error is pending
IsMoving	BOOL		True = robot is moving
EStopActive	BOOL		True = E-stop is pending
DummyPlug	BOOL		True = Staubli teach pendant replaced by dummy plug
ExternalWMS	BOOL		True = Remote WMS option activated. Operation mode selection is done from user-supplied Working Mode Selector device. Staubli WMS Box is replaced by J113 dummy plug. False = Staubli WMS box is used OR externalWMS option NOT configured properly (see iomap.cfx file on robot side)

Name	Type	Initial	Description
ExternalMCP	BOOL		True = Remote MCP option activated. User-supplied Teach Pendant is used for jogging robot in manual mode.False = Stäubli SP1 Teach pendant is in used or Remote WMS option NOT configured properly (see iomap.cfx file on robot side)
ExternalMcp_Wms	BOOL		True = Robot is properly configured to use both user- supplied WMS and Teach Pendant.
ActualSpeed	REAL		Cartesian speed of the current TCP
ActualOverride	REAL		Current override value [0.01 .. 100]
ActualErrorNumber	UINT		Total number of pending errors in robot
ActualOperationMode	UINT		Actual working mode 0= invalid , 1= manu, 3=Auto, 4 remote(extaut)
ActualErrorID	T_PendingErrors		List of pending error on server side
CartesianPos	T_CartesianPositions [▶ 75]		UDT for a robot position
JointPos	T_JointPositions [▶ 76]		
JointForces	ARRAY [0..5] OF REAL		
ActualCoordSystem	UINT	0	Number of the user frame in which the position of the Tool Center Point is reported.
ActualTool	UINT	0	Number of the Tool Center Point for which the position is reported.
RobotStateMachine	UINT		State Machine of the robot
MovementID	INT		Identifier of motion currently executed
MovementProgress	INT		Percentage of actual movement that has been completed
RobotModel	STRING(10)		Robot model connected to controller
ControllerModel	UINT		Model of connected Stäubli controller 8=CS8C / 9=CS9
CS9Safety	T_CS9SftyFbk [▶ 74]		CS9 Only - Safety features
Heartbeat	UINT		Counter incremented at every cycle of the server's communication task
ServerMajorVersion	UINT		Major version of unival PLC server
ServerMinorVersion	UINT		Minor version of unival PLC server
ServerEdit	UINT		Edit of the uniVAL plc server
ClientMajorVersion	UINT		Major version of unival PLC client library
ClientMinorVersion	UINT		Minor version of unival PLC client library
ClientEdit	UINT		Edit of the unival PLC client library

### 5.3.11 T\_Trnsf

TYPE T\_Trnsf : STRUCT

This data type represents a geometrical transformation

#### Syntax

#### Definition:

```

TYPE T_Trnsf :
STRUCT
  X   : REAL := 0;
  Y   : REAL := 0;
  Z   : REAL := 0;
  RX  : REAL := 0;

```

```

    RY : REAL := 0;
    RZ : REAL := 0;
END_STRUCT
END_TYPE

```

### Parameters

Name	Type	Initial	Description
X	REAL	0	X coordinate
Y	REAL	0	Y coordinate
Z	REAL	0	Z coordinate
RX	REAL	0	RX coordinate
RY	REAL	0	RY coordinate
RZ	REAL	0	RZ coordinate

## 5.4 enums

### 5.4.1 eBlending

TYPE eBlending :

Define rounding solutions. The members of this enumeration are matching the 'blend' field of VAL3 motion descriptor.

#### Syntax

##### Definition:

```

TYPE eBlending :
(
    NOTDEF      := 0,
    OFF         := 1,
    JOINT       := 2,
    CARTESIAN   := 3
);
END_TYPE

```

### Parameters

Name	Initial
NOTDEF	0
OFF	1
JOINT	2
CARTESIAN	3

### 5.4.2 eConfigElbow

TYPE eConfigElbow :

Define robot's elbow configuration. The members of this enumeration are matching VAL3 description of robot's elbow configuration.

#### Syntax

##### Definition:

```

TYPE eConfigElbow :
(
    eNotDefined := 0,
    eFree       := 1,
    eSame       := 2,
    ePositive   := 3,

```



```

    enegative      := 4
);
END_TYPE

```

**Parameters**

Name	Initial
eNotDefined	0
efree	1
esame	2
epositive	3
enegative	4

### 5.4.3 eConfigShoulder

TYPE eConfigShoulder :

Define robot's shoulder configuration The members of this enumeration are matching VAL3 description of robot's shoulder configuration

**Syntax**

**Definition:**

```

TYPE eConfigShoulder :
(
    sNotDefined    := 0,
    sfree          := 1,
    ssame          := 2,
    righty         := 3,
    lefty          := 4
);
END_TYPE

```

**Parameters**

Name	Initial
sNotDefined	0
sfree	1
ssame	2
righty	3
lefty	4

### 5.4.4 eConfigWrist

TYPE eConfigWrist :

Define robot's wrist configuration The members of this enumeration are matching VAL3 description of robot's wrist configuration

**Syntax**

**Definition:**

```

TYPE eConfigWrist :
(
    eNotDefined    := 0,
    wfree          := 1,
    wsame          := 2,
    wpositive      := 3,
    wnegative      := 4
);
END_TYPE

```

**Parameters**

Name	Initial
eNotDefined	0
wfree	1
wsame	2
wpositive	3
wnegative	4

**5.4.5 eMC\_BUFFER\_MODE**

TYPE eMC\_BUFFER\_MODE :

A fundamental part of interpolated motion control is blending of (buffered) consecutive motion commands on an axes group. This enumeration lists the supported buffer modes.

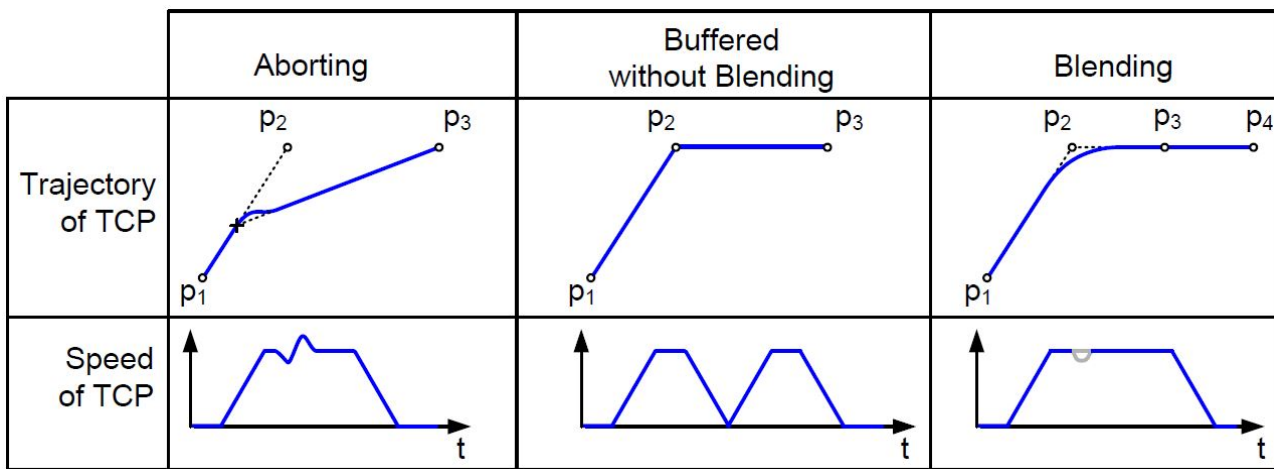


Fig. 2: Trajectory of the TCP of two consecutive motion commands in three modes

Source: PLCopen - Technical Committee 2 – Task Force / Document: Function blocks for motion control version 2.0

**Syntax**

**Definition:**

```

TYPE eMC_BUFFER_MODE :
(
  Aborting           := 0,
  Buffered           := 1,
  BlendingJoint      := 6,
  BlendingCartesian := 7
);
END_TYPE
    
```

**Parameters**

Name	Initial	Description
Aborting	0	A FB with buffer mode “Aborting” aborts any ongoing motion and starts the new motion immediately
Buffered	1	The next FB affects the axes group as soon as the previous motion is fully completed (Robot has reached commanded destination).
BlendingJoint	6	The current and the next motion FBs are blended, so the axes group will not stop between the motions. During blending, the robot is performing a joint interpolated motion

Name	Initial	Description
BlendingCartesian	7	The current and the next motion FBs are blended, so the axes group will not stop between the motions. The shape of the robot's motion during the blending is a Bezier curve

### 5.4.6 eMC\_TRANSITION\_MODE

TYPE eMC\_TRANSITION\_MODE :

This enumeration defines the profile of the robot trajectory nearby the set points. Full description for this feature can be found in the document edited by PLCopen organization

Source: PLCopen - Technical Committee 2 – Task Force / Document: Function blocks for motion control version 2.0.



uniVALplc client library defines MC\_TransitionParameter data type for specifying mentioned distances.

#### Syntax

##### Definition:

```

TYPE eMC_TRANSITION_MODE :
(
  None           := 0,
  CornerDistance := 3,
  BlendingDistances := 10
);
END_TYPE
    
```

#### Parameters

Name	Initial	Description
None	0	The motion blocks are not modified, and no transition curve is inserted.
CornerDistance	3	This mode is like 'Blending distances' except that leave and reach distances are equal.
BlendingDistances	10	If the position where the original contour can be left is known, the user can specify: <ul style="list-style-type: none"> <li>• The distance from the arrival point at which the nominal trajectory is left.</li> <li>• The distance from the arrival point at which the nominal trajectory is rejoined</li> </ul>

### 5.4.7 eOperationMode

TYPE eOperationMode :

Define the working mode of the robot. The working mode is selected with a dedicated key switch located on the WMS.

The picture below shows the Working Mode Selector currently abbreviated WMS.



Fig. 3: WMS Box

**Syntax**

**Definition:**

```
TYPE eOperationMode :
(
  wrkMode_INVALID   := 0,
  wrkMode_MANUAL    := 1,
  wrkMode_TEST      := 2,
  wrkMode_AUTO      := 3,
  wrkMode_REMOTE    := 4
);
END_TYPE
```

**Parameters**

Name	Initial	Description
wrkMode_INVALID	0	
wrkMode_MANUAL	1	
wrkMode_TEST	2	Not implemented
wrkMode_AUTO	3	
wrkMode_REMOTE	4	

**5.4.8 eStateMachine**

TYPE eStateMachine :

Define steps of the robot's state machine

**Syntax**

**Definition:**

```
TYPE eStateMachine :
(
  SM_POWEROFF       := 0,
  SM_STANDBY        := 1,
  SM_INTERRUPT      := 2,
  SM_HOLD           := 6,
  SM_MOVING         := 3,
  SM_CONNECT        := 4,
  SM_JOGGING        := 5
);
END_TYPE
```

**Parameters**

Name	Initial	Description
SM_POWEROFF	0	Robot power is OFF

Name	Initial	Description
SM_STANDBY	1	Robot settled with no pending movement
SM_INTERRUPT	2	Robot settled with pending movement
SM_HOLD	6	Robot stopped because of Move/Hold key
SM_MOVING	3	Robot is moving
SM_CONNECT	4	Robot is moving along a connection move
SM_JOGGING	5	Robot is jogged from teach pendant

## 5.5 unival PLC Error Codes

### 5.5.1 CS8C\_ErrorMessage

Id	Description
0	No Error
1	uniVAL plc server license not installed
2	uniVAL plc server demo period has expired. Reboot is needed
3	Lifebit Error. The lifebit period seems to be not stable.
5	Try to load a non-existing server.
6	Error while loading server. check event logger for more details
7	ArmlIO error: IO signal not responding
8	Controller IO error: IO signal not responding (UsrIn, Valves, Fout, Fin)
9	unknown server Key or server not ready
10	Robot model could not be identified. Check event logger for more information
20	Invalid function code
21	BankID not defined in user database or user frame not defined in plc database (range: [0..31])
22	Command not supported for this controller.
23	TCP not defined
24	Mdesc not defined
25	Joint not defined
26	Point not defined
27	Invalid value for VEL
28	Invalid value for ACCEL
29	Invalid value for DECEL
30	Invalid value for TVEL
31	Invalid value for RVEL
32	Invalid value for LEAVE
33	Invalid value for REACH
34	Invalid value for BLEND. Range is [1..3]
35	Userframe cannot be computed. Check the position of the 3 construction points
36	Invalid point index for Userframe computation range is: [1..3]
37	Invalid configuration for a point
38	Try to access a datatype that does not exist. (1=Point 2=Joint 3=TCP 4=Mdesc)
39	Invalid size for a bank
40	Position not reachable.
41	Motion stack is locked because a previous motion has failed.
43	Change in configuration detected - Shoulder
44	Change in configuration detected - Elbow

Id	Description
45	Change in configuration detected - Wrist
46	Robot powering ON/OFF sequence has failed.
48	Cannot execute movement, no database has been loaded.
50	Cannot change monitor speed. Robot not in 'Remote' working mode.
51	Invalid frame number. The position of the current active Tool is reported relatively to a defined userFrame. There are 32 userframes available on robot side. This error is reported when you select a userFrame number that is out of range [0..31]
52	Invalid Tool number. A total number of 32 Tools are available on robot side. You shall select the number of the Tool which position is reported by uniVAL plc server. This error is reported when you select a Tool number that is out of range [0..31]
60	uniVALplc client library initialization failed.
61	Parameter AxesGroup not in range.
62	Index parameter not in range.
63	Invalid Buffer mode (0='Aborted' 1='Buffered' 6='Blending Joint' 7='Blending Cartesian')
64	Invalid Transition Mode (0='None' 3='Corner distance' 10='Blending distances).
65	Communication reset during transaction. (Communication protocol aborted)
66	Data of the profinet module cannot be read at specified address.
67	Access to an invalid input or output number
68	Operation timeout. Server acknowledgment missed.
90	RemoteMCP mode not activated. Dummy plug not detected. Check that dummy plug is installed on J110.
92	Invalid Jog mode selected. (1=Joint 2=Frame 3=Tool)
93	Cannot jog: Manual mode requested
94	Cannot jog: Robot power is OFF
110	Internal error #10 while loading a data base. Please contact your local Stäubli business unit
111	Internal error #11 while loading a data base. Please contact your local Stäubli business unit
112	Internal error #12 while loading a data base. Please contact your local Stäubli business unit
113	Internal error #13 while loading a data base. Please contact your local Stäubli business unit
114	Internal error #14 while loading a data base. Please contact your local Stäubli business unit
120	Internal error #20 while loading a data base. Please contact your local Stäubli business unit
121	Internal error #21 while loading a data base .Please contact your local Stäubli business unit
122	Internal error #22 while loading a data base. Please contact your local Stäubli business unit
123	Internal error #23 while loading a data base. Please contact your local Stäubli business unit
130	Internal error #30 while loading a data base. Please contact your local Stäubli business unit
131	Internal error #31 while loading a data base. Please contact your local Stäubli business unit
132	Cannot load database, Device not found.
133	Cannot load database, Device not ready
134	Cannot load database, Device timeout
139	Cannot load database, File not found
142	Cannot load database, No answer from FTP server.
200	Envelope error.(0x1503)
201	Cannot settle arm power: working mode change. (0x1605)
202	Joint limit electrical switch activated. (0x160d)
203	Drive error SAFETY-DriveFault. (0x160e)
204	Cannot settle arm power (0x1626)
205	Arm power shutdown: working mode change. (0x1627)
206	Teach pendant emergency stop activated. (0x1709)
216	System event SAFETY-ShortOpen.(0x1736)
217	System event SAFETY-Watchdog.(0x1737)
218	System event SAFETY-ThermoOpened.(0x1738)

Id	Description
219	Internal power error (fuse F3 on USBI safety board ?).(0x1739)
220	Internal power error (fuse F2 on RSI board ?).(0x173a)
221	Teach pendant emergency stop activated.(0x173b)
222	Remote emergency stop activated.(0x173c)
223	Cell emergency stop chain open (User ES1-2)
224	Deadman switch released or emergency stop chain for manual mode open (User EN)
225	Emergency stop chain DOOR open
226	cell emergency stop chain open (User ESB1-2).(0x1740)
227	brake select switch turned from normal (0) position.(0x1741)
228	brake release button pressed.(0x1742)
229	system event ROBOT-SafetyInit.(0x1743)
230	system event SAFETY-UnstableSignal.(0x1744)
231	system event SAFETY-NoSignal.(0x1745)
232	system event SAFETY-Transient.(0x1746)
233	phasing signal is enabled.(0x1747)
234	system event SAFETY-EstopSoft.(0x1748)
235	system event ROBOT-Safety.(0x1749)
238	Use of WMS is not allowed.(0x180A)
239	The robot is not calibrated: only joint jog move is possible.
240	Cannot move: destination is out of software axis limits.(0x3b05)
241	Cannot complete move: destination is out of reach.(out of software joint limits).0x3b06)
242	Cannot move: cannot cross singularity in this case.(0x3b07)
243	Cannot apply arm power: arm power is still in disabling phase.(0x3f04)
244	Cannot apply arm power: press and hold the deadman switch. (0x3f08)
245	Cannot apply arm power: release & press deadman to enable arm power request. (0x3f12)
246	Jog mode cannot be selected: first enable arm power.(0x3f15)
247	Turn off arm power or switch to manual mode. Jogging is not possible in this mode. (0x3f16)
248	Jog move impossible: first select a jog mode.(0x3f19)
249	Cannot apply arm power: park the pendant on the cradle.(0x3f22)
251	Invalid working mode. Try to perform an action which is not allowed in the current working mode. e.g. switch power ON in remote workingmode.(0x4001)
252	Cannot apply arm power: after an emergency stop, eStop acknowledge signal (as configured in iomap.cf) must be validated.(0x4002)
800	Unexpected application runtime error. Check error logger
810	Invalid numerical calculation (division by zero).
811	Invalid numerical calculation (e.g.ln(-1))
820	Access to an array with an index that is larger than the array size.
821	Access to an array with negative index or resize to zero.
829	Invalid task name. See taskCreate() instruction.
830	The specified name does not correspond to any VAL 3 task.
831	A task with the same name already exists. See taskCreate instruction.
832	Only 2 different periods for synchronous tasks are supported. Change scheduling period.
840	Not enough memory space available.
841	Not enough memory space to run the task. See the run memory size.
860	Maximum instruction run time exceeded.
861	Internal VAL 3 interpreter error
870	Invalid instruction parameter. See the corresponding instruction.
880	Uses data or a program from a library not loaded in the memory.

Id	Description
881	Incompatible kinematic: Use of a point/joint/config that is not compatible with the arm kinematic.
882	The reference frame or tool of a variable belongs to a library and is not accessible.
890	The task cannot resume from the location specified. See taskResume() instruction.
900	The speed specified in the motion descriptor is invalid (negative or too great).
901	The acceleration specified in the motion descriptor is invalid (negative or too great).
902	The deceleration specified in the motion descriptor is invalid (negative or too great).
903	The translation velocity specified in the motion descriptor is invalid (negative or too great).
904	The rotation velocity specified in the motion descriptor is invalid (negative or too great).
905	The reach parameter specified in the movement descriptor is invalid (negative).
906	The leave parameter specified in the movement descriptor is invalid (negative).
922	Attempt to write in a system input.
923	Use of a dio, aio or sio input/output not connected to a system input/output.
924	Attempt to access a protected system input/output
925	Read or write error on a dio, aio or sio (field bus error)
950	Cannot run this movement instruction: a previous movement request could not be completed (point out of reach, singularity, configuration problem, etc.)
953	Movement command not supported
954	Invalid movement instruction: target out of reach, or check the movement descriptor.
960	Invalid flange tool coordinates
961	Invalid world tool coordinates
962	Use of a point without a reference frame. See Definition.
963	Use of a frame without a reference frame. See Definition.
964	Use of a tool without reference tool. See Definition.
965	Invalid frame or reference tool (global variable linked to a local variable)
999	Unexpected error. Please check event logger of Stäubli controller.

## 5.5.2 CS9\_ErrorMessages

Id	Description
0	No Error
1	uniVAL plc server license not installed
2	uniVAL plc server demo period has expired. Reboot is needed
3	Lifebit Error. The lifebit period seems to be not stable.
4	Not defined
5	Try to load a non-existing server.
6	Error while loading server. check event logger for more details
7	ArmIO error: IO signal not responding
8	Controller IO error: IO signal not responding (UsrIn, Valves, Fout, Fin)
9	unknown server Key or server not ready
10	Robot model could not be identified. Check event logger for more information
20	Invalid function code
21	BankID not defined in user database or user frame not defined in plc database (range: [0..31]).
22	Command not supported for this controller.
23	TCP not defined
24	Mdesc not defined
25	Joint not defined



Id	Description
26	Point not defined
27	Invalid value for VEL
28	Invalid value for ACCEL
29	Invalid value for DECEL
30	Invalid value for TVEL
31	Invalid value for RVEL
32	Invalid value for LEAVE
33	Invalid value for REACH
34	Invalid value for BLEND. Range is [1..3]
35	Userframe cannot be computed. Check the position of the 3 construction points
36	Invalid point index for Userframe computation range is: [1..3]
37	Invalid configuration for a point
38	Try to access a datatype that does not exist. (1=Point 2=Joint 3=TCP 4=Mdesc)
39	Invalid size for a bank
40	Position not reachable.
41	Motion stack is locked because a previous motion has failed.
43	Change in configuration detected - Shoulder
44	Change in configuration detected - Elbow
45	Change in configuration detected - Wrist
46	Robot powering ON/OFF sequence has failed.
47	
48	Cannot execute movement, no database has been loaded.
49	
50	Cannot change monitor speed. Robot not in 'Remote' working mode.
51	User-frame number is out of range [0..31]
52	Tool number is out of range [0..31]
60	uniVALplc client library initialization failed.
61	Parameter AxesGroup not in range.
62	Index parameter not in range.
63	Invalid Buffer mode (0='Aborted' 1='Buferrred' 6='Blending Joint' 7='Blending Cartesian')
64	Invalid Transition Mode (0='None' 3='Corner distance' 10='Blending distances).
65	Communication reset during transaction. (Communication protocol aborted)
66	Data of the profinet module cannot be read at specified address.
67	Invalid inpt/output signal number.
68	Operation timeout. Server acknowledgment missed.
90	RemoteMCP mode not activated. Dummy plug not detected. Check that dummy plug is installed on J110.
92	Invalid Jog mode selected. (1=Joint 2=Frame 3=Tool)
93	Cannot jog: Manual mode requested
94	Cannot jog: Robot power is OFF
110	Internal error #10 while loading a data base. Please contact your local Stäubli business unit
111	Internal error #11 while loading a data base. Please contact your local Stäubli business unit
112	Internal error #12 while loading a data base. Please contact your local Stäubli business unit
113	Internal error #13 while loading a data base. Please contact your local Stäubli business unit
114	Internal error #14 while loading a data base. Please contact your local Stäubli business unit
120	Internal error #20 while loading a data base. Please contact your local Stäubli business unit
121	Internal error #21 while loading a data base .Please contact your local Stäubli business unit
122	Internal error #22 while loading a data base. Please contact your local Stäubli business unit
123	Internal error #23 while loading a data base. Please contact your local Stäubli business unit

Id	Description
130	Internal error #30 while loading a data base. Please contact your local Stäubli business unit
131	Internal error #31 while loading a data base. Please contact your local Stäubli business unit
132	Cannot load database, Device not found.
133	Cannot load database, Device not ready
134	Cannot load database, Device timeout
139	Cannot load database, File not found
142	Cannot load database, No answer from FTP server.
199	Not response from RSI board.
200	Envelope error.(0x1503)
201	internal DSI board voltage error
202	redundancy check error in DSI board
203	internal DSI error
204	communication error with one encoder
205	shortcut on RBR signal enable1 (J1204)
206	shortcut on RBR signal enable2 (J1204)
207	shortcut on RBR signal mode1 (J1204)
208	shortcut on RBR signal mode2 (J1204)
209	shortcut on RBR signal select1 (J1204)
210	shortcut on RBR signal select2 (J1204)
211	shortcut on RBR signal select3 (J1204)
212	shortcut on RBR signal select4 (J1204)
213	invalid voltage on a RBR input
214	open wire on brake 1
215	open wire on brake 2
216	open wire on brake 3
217	open wire on brake 4
218	open wire on brake 5
219	open wire on brake 6
220	DSI board alarm: unexpected encoder feedback on axis 1
221	DSI board alarm: unexpected encoder feedback on axis 1
222	DSI board alarm: unexpected encoder feedback on axis 2
223	DSI board alarm: unexpected encoder feedback on axis 2
224	DSI board alarm: unexpected encoder feedback on axis 3
225	DSI board alarm: unexpected encoder feedback on axis 3
226	DSI board alarm: unexpected encoder feedback on axis 4
227	DSI board alarm: unexpected encoder feedback on axis 4
228	DSI board alarm: unexpected encoder feedback on axis 5
229	DSI board alarm: unexpected encoder feedback on axis 5
230	DSI board alarm: unexpected encoder feedback on axis 6
231	DSI board alarm: unexpected encoder feedback on axis 6
232	cut wire, disconnected or shortcut on valve A1
233	cut wire, disconnected or shortcut on valve A2
234	cut wire, disconnected or shortcut on valve B1
235	cut wire, disconnected or shortcut on valve B2
236	cut wire, disconnected or shortcut on brake 1
237	cut wire, disconnected or shortcut on brake 2
238	cut wire, disconnected or shortcut on brake 3
239	cut wire, disconnected or shortcut on brake 4

Id	Description
240	cut wire, disconnected or shortcut on brake 5
241	cut wire, disconnected or shortcut on brake 6
242	shortcut on a valve signal
243	shortcut on a brake signal
244	shortcut on a RBR signal
245	DSI temperature is too high
246	invalid brake power supply
247	unexpected voltage on encoder 1 power supply
248	unexpected voltage on encoder 2 power supply
249	unexpected voltage on encoder 3 power supply
250	unexpected voltage on encoder 4 power supply
251	unexpected voltage on encoder 5 power supply
252	unexpected voltage on encoder 6 power supply
253	DSI in standalone mode: please check if RBR is not activated or press update button
254	An encoder change detected. Encoder validation procedure is required (calibration page).
255	safety alarm: reboot is required after validation of the faulty safety signal
256	unexpected STO feedback from the drives: STO1 OFF
257	unexpected STO feedback from the drives: STO1 ON
258	unexpected STO feedback from the drives: STO2 OFF
259	unexpected STO feedback from the drives: STO2 ON
260	diagnostic is missing. Check safety board I/O on control panel
261	diagnostic is missing. Check safety board I/O on control panel
262	inconsistent safety stop signal from the MCP (MCPES1 off)
263	inconsistent safety stop signal from the MCP (MCPES2 off)
264	inconsistent safety stop signal from user stop USIA (USIA1 (J100-1, J101-7) off)
265	inconsistent safety stop signal from user stop USIA (USIA2 (J100-2, J101-8) off)
266	inconsistent safety stop signal from user stop USIB (USIB1 (J100-5) off)
267	inconsistent safety stop signal from user stop USIB (USIB2 (J100-6) off)
268	inconsistent safety stop signal from user stop USIC (USIC1 (J100-9) off)
269	inconsistent safety stop signal from user stop USIC (USIC2 (J100-10) off)
270	inconsistent safety stop signal from user stop USID (USID1 (J100-13) off)
271	inconsistent safety stop signal from user stop USID (USID2 (J100-14) off)
272	communication lost between the CPU and the safety board
273	drive error
274	interruption or too much delay in the communication between the MCP and the safety board
275	safety stop request from MCP
276	safety stop request from MCP
277	safety stop request from MCP
278	ESTOP on WMS (J101 5-7,6-8) or safety stop request from user USIA (J100 1-3,2-4)
279	user stop USIA1 (J100-1)
280	user stop USIA2 (J100-2)
281	user stop USIB (J100-5/6)
282	user stop USIB1 (J100-5)
283	user stop USIB2 (J100-6)
284	user stop USIC (J100-9/10)
285	user stop USIC1 (J100-9)
286	user stop USIC2 (J100-10)
287	user stop USID (J100-13/14)

Id	Description
288	user stop USID1 (J100-13)
289	user stop USID2 (J100-14)
290	no working mode request from the CPU (safety need a WMS and there is no WMS connected)
291	SS1 safety stop condition from external safety PLC
292	SS2 safety stop condition from external safety PLC
293	diagnostic is missing. Check safety board I/O on control panel
294	diagnostic is missing. Check safety board I/O on control panel
295	WMS stop (WMS signal WMS-Local (J101-1))
296	WMS stop (WMS signal WMS-Remote (J101-2))
297	WMS stop (WMS signal WMS-Manu (J101-3))
298	working mode change
299	working mode change
300	working mode change
301	arm power is not allowed in brake release mode
302	diagnostic is missing. Check safety board I/O on control panel
303	brake test timeout
304	activation of brake test mode
305	the MCP plug was not placed in time after the MCP removal request
306	a safety restart is needed
307	safety reset signal activated
308	the enabling device is released
309	release the enabling device and press it again
310	safe velocity required but disabled
311	safe referencing required but not defined
312	safety overspeed
313	safe joint limits overpassed
314	movement detected in Safe Operating Stop mode
315	the enabling device was not released 1s after activation of service auto
316	safety stop request from Zone1
317	safety stop request from Zone2
318	diagnostic is missing. Check safety board I/O on control panel
319	diagnostic is missing. Check safety board I/O on control panel
320	diagnostic is missing. Check safety board I/O on control panel
321	diagnostic is missing. Check safety board I/O on control panel
322	diagnostic is missing. Check safety board I/O on control panel
323	diagnostic is missing. Check safety board I/O on control panel
324	diagnostic is missing. Check safety board I/O on control panel
325	manual slow working mode expected
326	brake release mode expected to open brakes without arm power
327	a brake is still in closing phase
328	no brake is selected
329	several brakes are selected
330	no communication between CPU and safety board RSI9
331	diagnostic is missing. Check safety board I/O on control panel
332	diagnostic is missing. Check safety board I/O on control panel
333	diagnostic is missing. Check safety board I/O on control panel
334	diagnostic is missing. Check safety board I/O on control panel
335	diagnostic is missing. Check safety board I/O on control panel

Id	Description
336	diagnostic is missing. Check safety board I/O on control panel
337	diagnostic is missing. Check safety board I/O on control panel
338	diagnostic is missing. Check safety board I/O on control panel
339	diagnostic is missing. Check safety board I/O on control panel
340	temperature out of range
341	internal voltage out of range
342	internal error
343	the safety application contains faults
344	safety configuration is corrupted
345	the safety configuration does not match the safety application
346	the format of the safety configuration is not supported
347	internal communication error
348	test output fault
349	input deactivation fault
350	signal on USIA1 (J100-1) is not the expected test signal TDO0 (J100-3/8/11, J101-5, J103-7)
351	signal on USIA2 (J100-2) is not the expected test signal TDO1 (J100-4, J101-6, J103-12)
352	signal on USIB1 (J100-5) is not the expected test signal TDO2 (J100-7/16, J103-5)
353	signal on USIB2 (J100-6) is not the expected test signal TDO0 (J100-3/8/11, J101-5, J103-7)
354	signal on USIC1 (J100-9) is not the expected test signal TDO0 (J100-3/8/11, J101-5, J103-7)
355	signal on USIC2 (J100-10) is not the expected test signal TDO3 (J100-12/15, J103-3)
356	signal on USID1 (J100-13) is not the expected test signal TDO3 (J100-12/15, J103-3)
357	signal on USID2 (J100-14) is not the expected test signal TDO2 (J100-7/16, J103-5)
358	signal on WMS-Local (J101-1) is not the expected test signal TDO0 (J100-3/8/11, J101-5, J103-7)
359	signal on WMS-Remote (J101-2) is not the expected test signal TDO0 (J100-3/8/11, J101-5, J103-7)
360	signal on WMS-Manu (J101-3) is not the expected test signal TDO0 (J100-3/8/11, J101-5, J103-7)
361	signal on WMS-Reset (J101-4) is not the expected test signal TDO1 (J100-4, J101-6, J103-12)
362	signal on Enabling1 (J103-8) is not the expected test signal TDO0 (J100-3/8/11, J101-5, J103-7)
363	signal on Enabling2 (J103-17) is not the expected test signal TDO1 (J100-4, J101-6, J103-12)
364	signal on Estop1 (J103-6) is not the expected test signal TDO2 (J100-7/16, J103-5)
365	signal on Estop2 (J103-4) is not the expected test signal TDO3 (J100-12/15, J103-3)
366	signal on USOA1 (J102-1) is not as expected
367	signal on USOA2 (J102-2) is not as expected
368	signal on USOB1 (J102-5) is not as expected
369	signal on USOB2 (J102-6) is not as expected
370	signal on USOC1 (J102-9) is not as expected
371	signal on USOC2 (J102-10) is not as expected
372	internal signal on STO1 (J222-10) is not as expected
373	internal signal on STO2 (J222-10) is not as expected
374	shortcut to 24V on USOA1 (J102-1) or USOA2 (J102-2)
375	shortcut to 24V on USOB1 (J102-5) or USOB2 (J102-6)
376	shortcut to 24V on USOC1 (J102-9) or USOC2 (J102-10)
377	shortcut to 24V on STO1 or STO2
378	shortcut to 24V on STO1 or STO2
379	safety overrun. Reduce the number/type of monitoring points
380	diagnostic is missing. Check safety board I/O on control panel
381	diagnostic is missing. Check safety board I/O on control panel
382	diagnostic is missing. Check safety board I/O on control panel
383	diagnostic is missing. Check safety board I/O on control panel

Id	Description
384	unreferenced error on DSI board
385	safety license missing or over
386	RSI temperature exceeds threshold
387	Encoder on Axis1 exceeds a temperature threshold
388	Encoder on Axis2 exceeds a temperature threshold
389	Encoder on Axis3 exceeds a temperature threshold
390	Encoder on Axis4 exceeds a temperature threshold
391	Encoder on Axis5 exceeds a temperature threshold
392	Encoder on Axis6 exceeds a temperature threshold
393	Safe referencing (manu or auto) need to be activated in safety program because of SEL, SLS or SZM configuration
394	continuous axis license not compatible with safety version lower than 200.004
395	maintenance mode is not allowed by the current user profile
396	current working mode is not allowed by the current user profile
397	Safe referencing inconsistency: positions of 'safe position 1' and 'safe position 2' should be different for all joint
398	SS2 is unsupported in current safety version (requested from external safety PLC?)
399	Invalid working mode. Try to perform an action which is not allowed in the current working mode.
400	Cannot settle arm power: error reading DSI temperature.
401	Cannot settle arm power: DSI temperature must be above 0 degrees Celsius.
403	Arm power shutdown: internal power error (24V_OK in ARPS module)
404	Arm power shutdown: controller power supply shutdown detected (MAIN_POWER_OK signal)
405	Arm power shutdown: due to prepareCpuShutdown VAL3 function call
406	Cannot apply arm power: controller not yet initialized.
407	Cannot apply arm power: simultaneous power off request.
408	Cannot apply arm power: arm power is still in disabling phase.
410	Overspeed on joint. (0x0209)
411	Cannot move: cannot cross singularity in this case.(0x021c)
412	Translation or Rotational velocity too high (0x021e) (0x021f)
413	Cartesian position unreachable with the specified configuration. (0x0215)
414	Jog mode cannot be selected: first enable arm power.(0x3f15)
415	Jog mode cannot be selected: first select manual working mode, then enable arm power. (0x3f16)
416	Jog move impossible: first select a jog mode.(0x3f19)
417	Invalid move axis or direction. (0x3f1A)
418	Cannot apply arm power: park the pendant on the cradle.(0x3f22)
420	
421	
438	Safety board communication error. Please check event logger of Stäubli controller.
440	system event CPU-BatteryOff. CPU board battery is not present or dead.
441	system event CPU-Overcurrent. Overcurrent detected on CPU.
442	system event CPU-CpuReadingFailed. CPU temperature reading failed.
443	system event CPU-BBReadingFailed. Baseboard temperature reading failed.
444	system event CPU-FanReadingFailed. CPU board fan speed reading error.
445	system event CPU-BatteryReadingFailed. Battery state reading failed.
446	system event CPU-OvercurrentReadingFailed. CPU board overcurrent state reading error.
447	system event CPU-CpuTemperatureLow. CPU temperature is lower than minimum threshold).
448	system event CPU-CpuTemperatureWarning. CPU temperature exceeded the warning threshold.
449	system event CPU-CpuTemperatureLimit. CPU temperature exceeded the error threshold. Behaviour of CPU board is uncertain.

Id	Description
450	system event CPU-BBTemperatureLow. Baseboard temperature is lower than minimum threshold.
451	system event CPU-BBTemperatureWarning. Baseboard temperature exceeded the warning threshold.
452	system event CPU-BBTemperatureLimit. Baseboard temperature exceeded the error threshold Behaviour of CPU board is uncertain.
453	
454	system event CPU-FanHigh. CPU board fan speed is higher than limit.
455	CPU fan control failed.
456	Safety board error 'Not ready after a minute': Failed to initialize. Please reboot the controller.
457	Safety board error: Not ready anymore.
458	Safety alarm.
459	Safety error . A reboot is required.
460	Safety board error communication error. Please check event logger of Stäubli controller and Staubli support service.
461	
470	Invalid safe referencing position.
471	Safety restart failed during safe referencing procedure.
472	The number of the safe reference position does not match the status of safe referencing procedure.
473	The reference position is out of range.
479	The test brake procedure failed.
480	No
481	Hibernate function not supported by controller
482	Timeout waiting for confirmation of the safety board.
483	Robot must be disabled to activate power save.
484	
485	
486	
487	Wake up function not supported by controller
488	Timeout waiting for encoder interface ready state
489	Timeout waiting for safety board ready state
513	Internal motion generator error.
514	Invalid parameter value.
515	Invalid motion generator state.
520	Joint position out of software limits.
521	Position out of software limits on joints: %s.
522	Overspeed on joint %u.
525	Arm position not allowed.
526	Translation velocity too high.
527	Rotation velocity too high.
530	No joint solution found (no convergence). Cartesian position probably out of reach.
531	Cartesian position out of joint limits.
532	Cartesian position out of workspace.
533	Cartesian position unreachable with the specified configuration.
534	Orientation unreachable with this arm.
535	The arm position found is in a forbidden area.
536	The arm positions found are either outside joint limits or inside forbidden areas.
540	Cannot cross singularity in this case.
541	Internal error: Cartesian path discontinuity.

Id	Description
542	Configuration discontinuity: the joint position reached does not match the arm configuration requested for next movement.
543	Robot not at trajectory start position.
544	The Cartesian path is leading a multi-turn joint out of its range.
550	Circular arc: the angle between start and intermediate points exceeds 180 deg.
551	Circular arc: the angle between intermediate and end points exceeds 180 deg.
552	Circular arc: start, intermediate and end points are too close.
553	Circular arc: the orientation change between start and intermediate, or intermediate and end points is exactly 180 deg. The direction of the move is
554	Cartesian blending: the rotation is too large. Move direction maybe ambiguous.
560	No blending supported with compliant moves.
561	No blending supported with movejSync.
562	'joint' blending not supported with tracking movements. Use 'Cartesian' instead.
563	No blending supported with trajMove.
564	No blending supported with splines.
565	Invalid blending in mdesc : 'Cartesian' blending is not possible when switching between {external axes+arm} and {arm only} kinematics.
570	Invalid data for this arm (wrong number of joints or wrong kinematic).
571	Invalid kinematic chain part.
572	Part mounting error: a resetMotion() is required to switch to carried part trajectory mode.
573	Part mounting error: a resetMotion() is required to switch to carried tool trajectory mode.
574	Part mounting error: carried part mode is not supported for this move.
580	A 'trackOn' movement is required to start a conveyor tracking sequence.
581	A 'trackOff' movement is required to end a conveyor tracking sequence.
582	The moving frame is different from the previous one. Use a 'trackon' movement to switch between moving frames.
583	The tracking movements stopped: %s
584	Tracking iterations failed: the conveyor may be out of reach or too fast.
585	Internal tracking error.
586	Conveyor too fast to be tracked.
590	Alter: invalid status, command ignored.
591	Alter: alteration of the path starting.
592	Alter: alteration of the path stopping.
800	Unexpected application runtime error. Check error logger
810	Invalid numerical calculation (division by zero).
811	Invalid numerical calculation (e.g.ln(-1))
820	Access to an array with an index that is larger than the array size.
821	Access to an array with negative index or resize to zero.
829	Invalid task name. See taskCreate() instruction.
830	The specified name does not correspond to any VAL 3 task.
831	A task with the same name already exists. See taskCreate instruction.
832	Only 2 different periods for synchronous tasks are supported. Change scheduling period.
840	Not enough memory space available.
841	Not enough memory space to run the task. See the run memory size.
860	Maximum instruction run time exceeded.
861	Internal VAL 3 interpreter error
870	Invalid instruction parameter. See the corresponding instruction.
880	Uses data or a program from a library not loaded in the memory.
881	Incompatible kinematic: Use of a point/joint/config that is not compatible with the arm kinematic.



Id	Description
882	The reference frame or tool of a variable belongs to a library and is not accessible.
890	The task cannot resume from the location specified. See taskResume() instruction.
900	The speed specified in the motion descriptor is invalid (negative or too great).
901	The acceleration specified in the motion descriptor is invalid (negative or too great).
902	The deceleration specified in the motion descriptor is invalid (negative or too great).
903	The translation velocity specified in the motion descriptor is invalid (negative or too great).
904	The rotation velocity specified in the motion descriptor is invalid (negative or too great).
905	The reach parameter specified in the movement descriptor is invalid (negative).
906	The leave parameter specified in the movement descriptor is invalid (negative).
922	Attempt to write in a system input.
923	Use of a dio, aio or sio input/output not connected to a system input/output.
924	Attempt to access a protected system input/output
925	Read or write error on a dio, aio or sio (field bus error)
950	Cannot run this movement instruction: a previous movement request could not be completed (point out of reach, singularity, configuration problem, etc.)
953	Movement command not supported
954	Invalid movement instruction: target out of reach, or check the movement descriptor.
960	Invalid flange tool coordinates
961	Invalid world tool coordinates
962	Use of a point without a reference frame. See Definition.
963	Use of a frame without a reference frame. See Definition.
964	Use of a tool without reference tool. See Definition.
965	Invalid frame or reference tool (global variable linked to a local variable)
999	Unexpected error. Please check event logger of Stäubli controller.

## 6 Support and Service

Beckhoff and their partners around the world offer comprehensive support and service, making available fast and competent assistance with all questions related to Beckhoff products and system solutions.

### Download finder

Our [download finder](#) contains all the files that we offer you for downloading. You will find application reports, technical documentation, technical drawings, configuration files and much more.

The downloads are available in various formats.

### Beckhoff's branch offices and representatives

Please contact your Beckhoff branch office or representative for [local support and service](#) on Beckhoff products!

The addresses of Beckhoff's branch offices and representatives round the world can be found on our internet page: [www.beckhoff.com](http://www.beckhoff.com)

You will also find further documentation for Beckhoff components there.

### Beckhoff Support

Support offers you comprehensive technical assistance, helping you not only with the application of individual Beckhoff products, but also with other, wide-ranging services:

- support
- design, programming and commissioning of complex automation systems
- and extensive training program for Beckhoff system components

Hotline: +49 5246 963-157  
e-mail: [support@beckhoff.com](mailto:support@beckhoff.com)

### Beckhoff Service

The Beckhoff Service Center supports you in all matters of after-sales service:

- on-site service
- repair service
- spare parts service
- hotline service

Hotline: +49 5246 963-460  
e-mail: [service@beckhoff.com](mailto:service@beckhoff.com)

### Beckhoff Headquarters

Beckhoff Automation GmbH & Co. KG

Huelshorstweg 20  
33415 Verl  
Germany

Phone: +49 5246 963-0  
e-mail: [info@beckhoff.com](mailto:info@beckhoff.com)  
web: [www.beckhoff.com](http://www.beckhoff.com)



More Information:  
**[www.beckhoff.com/tf5130](http://www.beckhoff.com/tf5130)**

Beckhoff Automation GmbH & Co. KG  
Hülshorstweg 20  
33415 Verl  
Germany  
Phone: +49 5246 9630  
[info@beckhoff.com](mailto:info@beckhoff.com)  
[www.beckhoff.com](http://www.beckhoff.com)

