

## SoftMotion: DriveInterface: Analog

Last update: 14.03.2008

Hardware interface	I/Os from standard PLC configuration
Supported drives	any +/-10V frequency converters or drive, which is controlled via a set velocity value with a actual position value
Runtimes	any
Author	Hilmar Panzer
Components	Analogdrive.lib
Version	1.9.4.1

### CONTENT

<b>1</b>	<b>PARAMETERS IN PLC CONFIG</b>	<b>2</b>
1.1	BusInterface .....	2
1.2	AxisGroup .....	2
1.3	supported Drive.wControlType.....	2
1.4	Additional structure <i>Analog_AXIS_REF</i> .....	2
1.5	Configuration data.....	2
1.5.1	Cyclic Inputs .....	2
1.5.2	Cyclic Outputs.....	2
<b>2</b>	<b>FEATURES</b>	<b>4</b>
2.1	Additional function blocks.....	4

## 1 Parameters in PLC config

### 1.1 BusInterface

wParam1	Not used
wParam2	Not used
dwParam1	Not used
dwParam2	Not used

### 1.2 AxisGroup

wParam1	Not used
wParam2	Not used
wParam3	Not used
wParam4	Not used
dwParam1	Not used
dwParam2	Not used
dwParam3	Not used
dwParam4	Not used

### 1.3 supported Drive.wControlType

T / - no	V/V yes	V/P yes	P/P yes	PV/PV yes	V/- no	CONF yes
----------	---------	---------	---------	-----------	--------	----------

The cyclically sent data must consist of: fSetPosition and/or fActPosition.

The received data must consist of: fActPosition and/or fActVelocity.

### 1.4 Additional structure *Analog\_AXIS\_REF*

name	Type	
in	Analog_AXIS_REF	internal use: input structure of cyclic data
out	AnalogOutput	internal use: input structure of cyclic data
controller	AnalogPositionController	internal use
hes	SMC_HardwareEndSwitches	internal use
dwIncperTurn	DWORD	number of increments per turn
dwEncoderCounterModulo	DWORD	Modulo value of Encoder input
dwMaxPositionDiff	DWORD	maximum position lag (0: don't watch)
dwOldActPosition	DWORD	internal use
dwActPosition	DWORD	internal use
dwPosOffset	DWORD	internal use
bOldCaptureOccured	BOOL	internal use
bInvertDirection	BOOL	FALSE: dwActPosition and fSetVelocity

run in same direction

## 1.5 Configuration data

During the first IEC cycle the following parameters must be configured:

Drive_MS.dwMaxPositionDiff	DWORD	Maximum position lag in increments (0: switched off)
Drive_MS.dwEncoderCounterModulo	DWORD	Modulo value of encoder input (dwActPosition)
Drive_MS.bInvertDirection	BOOL	FALSE: dwActPosition and fSetVelocity run in same direction
Drive_MS.out.fMaxVelocityPos	LREAL	The maximum velocity value (technical units) in positive direction, with the corresponding output value iMaxVelocityPos
Drive_MS.out.iMaxVelocityPos	INT	
Drive_MS.out.iZeroVelocity	INT	output value which produces standstill at drive
Drive_MS.out.fMaxVelocityNeg	LREAL	The maximum velocity value (<0; technical units) in negative direction, with the corresponding output value iMinVelocityPos
Drive_MS.out.iMaxVelocityNeg	INT	
Drive_MS.controller.fDeadTime	LREAL	Deadtime due to communication and system delay in cycles
Drive_MS.controller.fKP	LREAL	P-factor of the position controller

### 1.5.1 Cyclic Inputs

The following values must be received and updated cyclically in the application:

Name	Type	
Drive_MS.in.bEndLimitPos	BOOL	limit switch in positive direction (if not used: TRUE)
Drive_MS.in.bEndLimitNeg	BOOL	limit switch in negative direction (if not used: TRUE)
Drive_MS.in.dwActPosition	DWORD	Encoder Position
Drive_MS.in.bLatchOccured	BOOL	Optional: rising edge indicates that a latch has taken place
Drive_MS.in.dwLatchPosition	DWORD	Optional: Latched position

### 1.5.2 Cyclic Outputs

The following values must be transmitted cyclically from the application to the drive system:

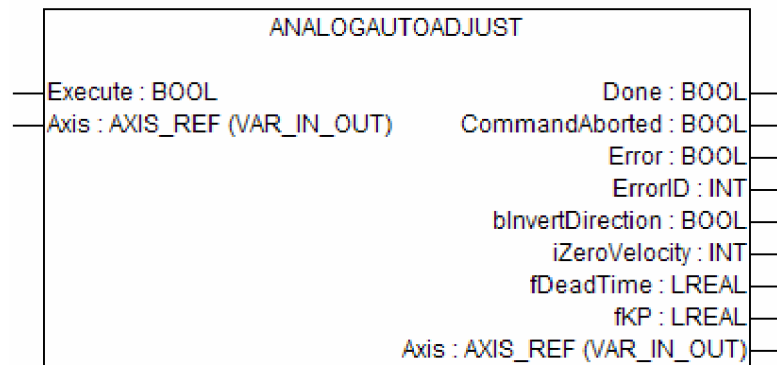
name	Type	
bEnable	BOOL	put power on axis
iSetVelocity	INT	set velocity value
bEnableLatch	BOOL	TRUE: enable latch mechanism

## 2 Features

- **RegulatorOn**
- Detecting and acknowledging lag **errors**
- any **gearing factors** (dwRatioTechUnitsDenom/iRatioTechUnitsNum)
- **linear/rotary axes**
- **controlling modes**: position (0/3), velocity (2): set param 1091 byControllerMode
- **limit switches** generate an error, when not TRUE.
- **homing** via SMC\_Homing
- **latching** (use MC\_TouchProbe with iTriggerNumber:=1)
- **setting and acknowledging drive errors** (use AXIS\_REF.bError to indicate an error and reset the drive's error, when AXIS\_REF.bErrorAckn is set)

### 2.1 Additional function blocks

#### AnalogAutoAdjust



This function block runs the motor and automatically adjusts the controller parameters bInvertDirection, iZeroVelocity, fDeadTime, fKp. It must be handled with care as the drive may act instable during this procedure and move quickly.

It may only be used, when at least the following parameters are already set:

Drive_MS.dwIncPerTurn	DWORD	Increments per turn (encoder resolution)
Drive_MS.dwMaxPositionDiff	DWORD	Maximum position lag in increments (0: switched off)
Drive_MS.dwEncoderCounterModulo	DWORD	Modulo value of encoder input (dwActPosition)
Drive_MS.out.fMaxVelocityPos	LREAL	The maximum velocity value (technical units) in positive direction, with the corresponding output value iMaxVelocityPos
Drive_MS.out.iMaxVelocityPos	INT	
Drive_MS.out.iZeroVelocity	INT	output value which produces standstill at drive
Drive_MS.out.fMinVelocityNeg	LREAL	The maximum velocity value (<0; technical units) in negative direction, with the corresponding output value iMinVelocityPos
Drive_MS.out.iMinVelocityNeg	INT	

Before starting this FB, power must be set on the drive. Therefore it is useful to first set Drive\_MS.controller.fKp to 0 (no position control loop), then use MC\_Power to enable the drive, and finally start this FB.

This FB shouldn't be called during every startup, but just during commissioning. It's output shall be read out, stored and set during the first IEC cycle of the following application starts.