

Application Notes

ServoOne / junior / Safety / LeviOne PROFINET with SIEMENS TIA Portal Example without Technology Objects (TO)

KeDrive

Disclaimer

The recommendations for action on which these Application Notes are based have been developed within the framework of tests under the ambient conditions specified in the operating instructions. The user is responsible for compliance with and verification of these environmental conditions in the specific application.

These Application Notes are intended for qualified personnel who commission and maintain drive and automation components. According to IEC 60364 or CENELEC HD 384, qualified personnel are persons who have the appropriate qualifications and are familiar with the installation, assembly, commissioning and operation of KEBA products (electrical devices) and who are familiar with all accident prevention regulations, directives and laws applicable at the place of use.

The safety instructions contained in the device documentation of the respective device must be observed.

The screenshots shown in these Application Notes are only examples to illustrate the individual steps.

Please note that KEBA products may contain software that is licensed as Open Source Software (OSS) or Free Software (FOSS). The license conditions of the OSS and / or FOSS contained / used in the products are available on the DevAdmin Service webpage on the controller. These must be complied with.

All information is subject to change at any time. Liability for correctness and completeness is excluded.

Exported from KeWiki on 15 January, 2025

Version info

Version	TIA V17
PLC	SIEMENS PLC compatible with TIA Portal (created with 1511-1PN, 1515F-2PN)
Slave	<ul style="list-style-type: none">• ServoOne/ServoOne Junior FW 4.30-00 or higher• LeviOne 51LJ008 FW 31.40-94 or higher
Service Tools	DriveManager 5.24.0
Device Description	GSDML-V2.34-LTI-ServoOne-20180530.xml (also for LeviOne)
Optional	HILSCHER NL 51N-DPL for LeviOne PROFIBUS to PROFINET Gateway

 **Download additional required data**

- on DataPortal

Table of Contents

- [1 Description](#)
- [2 ServoOne Operation Modes Standard telegram \(STD\) vs. PPO](#)
 - [2.1 Acyclic Communication via PKW-Channel](#)
 - [2.2 Acyclic Communication via DP-V1 service](#)
 - [2.3 Mapping](#)
 - [2.4 LeviOne Operation Modes](#)
 - [2.5 KEBA Example Projects](#)
 - [2.6 Hardware Configuration](#)
 - [2.7 Organization Blocks](#)
 - [2.8 Example Function Blocks](#)
 - [2.9 Watch List](#)
 - [2.10 Accessing the Example Function Blocks](#)
 - [2.11 Program Selectors](#)
- [3 Example Function Blocks](#)
 - [3.1 SO_Power](#)
 - [3.1.1 PCON: Switching between position and velocity mode](#)
 - [3.2 SO_PLCTablePos](#)

- [3.3 SO_MoveAbsolute](#)
 - [3.4 SO_ReadDPV1](#)
 - [3.5 SO_ReadPKW](#)
 - [3.6 SO_WriteDPV1](#)
 - [3.7 SO_WritePKW](#)
 - [3.8 SO_ReadActError](#)
 - [3.9 SO_ReadPKWString](#)
 - [3.10 SO_WritePKWString](#)
 - [3.11 SO_Service](#)
 - [3.12 TCP Read / Write functions](#)
 - [3.13 LowPass1_PT1](#)
 - [3.14 LowPass2_PT1](#)
 - [4 Known Issues](#)
-

1 Description

The delivered examples are designed for TIA Portal Tool by company SIEMENS. It contains example code for ServoOne and LeviOne slave devices. With this example, you are able to do first steps like switch on drive, read actual error, run homing, set new value (and table positioning), read/write integers and strings via PKW channel or DP-V1 acyclic service. For LeviOne with PROFIBUS you can also use a PROFIBUS to PROFINET Gateway.

Notes:

- This example projects handles no Technology Objects. The slaves are handled as IO-Devices (RT), means soft real time behaviour.
 - This project is based on a 1511-1PN CPU and 1515F-2PN. If using other CPU types like S7-1200 / S7-300 / S120 or else, it can be possible that some data types are not supported any more and with it some functions or function blocks. Then, you have to edit the project.
-

2 ServoOne Operation Modes Standard telegram (STD) vs. PPO

It doesn't matter if using PROFIBUS or PROFINET: Depending on your application, you have to define your operation mode for the drive. The ServoOne acts as PROFIBUS/PROFINET - I/O slave device and delivers the following modes:

1. **Position Mode (PCON)**
 - a. Point-to-Point movements
 - b. Target Velocity (endless positioning)
 - c. Traversing blocks
 - d. Jog-Mode (endless positioning)
2. **Velocity Mode (SCON)**
 - a. Target velocity
 - b. Jog-Mode

3. **V/f Mode (VFCON)**
 - a. Target velocity
4. **IRT Positioning (PCON) only with PROFINET**
 - a. Interpolated position mode

The modes 1 - 3 are done by the internal profile generator (PG) in the drive. Mode 4 is used for coordinated axis movements (real-time). For all modes you will need a process data mapping for a cyclic communication PLC $\leftarrow \rightarrow$ Drive. For simple applications using mode 1 ... 3 the PROFIdrive profile defines so called standard telegrams (**STD**). With it, the mapping is fixed. If you need a user defined mapping you have to use the parameter process object (**PPO**). The following STD telegrams are supported by ServoOne drives (*see also manual*):

- **STD1:** Velocity Control (VCON)
- **STD7:** Traversing Blocks (PCON)
- **STD8:** Interpolated Positioning (PCON IRT)
- **STD9:** Positioning/Velocity (endless positioning)

2.1 Acyclic Communication via PKW-Channel

Some PPO telegrams deliver an additional PKW channel (Parameter Configuration Channel). This channel/service is used to set or read parameters in the drives. The communication is acyclic means the processing time varies (no real time). It is a confirmed service. When you want to use the PKW channel beside an cyclic communication you have to map the right telegram which supports both.

2.2 Acyclic Communication via DP-V1 service

DP-V1 is an extension to the PROFINET standard and allows to access drive parameters without any PKW channel. It means that this service is independent of the chosen telegram type. A similar mechanism with request and response is used like PKW. The communication is also acyclic, means the processing time varies (no real time).

2.3 Mapping

When you are using the standard telegrams the master will write down the telegram selection value in P922[0]. Depending on that value the drive will map the necessary process data automatically in P915/P916. You can't change this mapping - it is fix.

Instead, when you are using a user defined mapping the master will write down a telegram value > 100 in P922. With it you have to map your parameters in P915/P916 manually! P915 means "*Write from PLC to drive*" and P916 means "*Read from drive to PLC*". You can find the same sequence in your PLC as input/output data.

Each entry represents a 16 bit value. If a 32 bit value (parameter) should be transmit, it has to be mapped twice in a row:

Tab. 1: Example: User Defined Mapping in the Drive

<div> <div>915</div> <div>COM_DP_PZDSelectionWrite</div> </div>				<div> <div>916</div> <div>COM_DP_PZDSelectionRead</div> </div>			
	915 0	COM_DP_PZDSelectionWrite	967		916 0	COM_DP_PZDSelectionRead	968
	915 1	COM_DP_PZDSelectionWrite	1275		916 1	COM_DP_PZDSelectionRead	1276
	915 2	COM_DP_PZDSelectionWrite	1275		916 2	COM_DP_PZDSelectionRead	1276

Example: Write to drive:

P967 = 16-Bit value = 1x Entry

P1275 = 32-Bit value = 2x Entry

...

Example: Read from drive:

P968 = 16-Bit value = 1x Entry

P1276 = 32-Bit value = 2x Entry

...

You can find the map able parameters for P915/916 in the signal list P1284 and P1285. The parameters P967 (*controlword*) and P968 (*statusword*) are mostly needed and at first position.

In *Tab. 1* you can find an example for a user defined mapping using a PPO telegram. With this mapping table you can estimate the minimum amount of process data read/write (in *Tab. 2*). Depending on this calculation, you have to select the telegram type.

Tab. 1: Example for User Defined Mapping in this TIA Example Project with PCON

P915 with PCON (from PLC)				P916 with PCON (to PLC)			
Object	Parameter	Datatype	Row Entry	Object	Parameter	Datatype	Row Entry
Controlword	P967[0]	uint16	1x	Statusword	P968[0]	uint16	1x
Target Position [user units]	P1275[0]	int32	2x	Act. Position [user units]	P1276[0]	int32	2x
Target Velocity [user units]	P1277[0]	int32	2x	Act. Velocity [user units]	P1271[0]	int32	2x
Acceleration [user units]	P1278[0]	uint16	1x	Act. current [A]	P700[0]	float32	2x
Deceleration [user units]	P1279[0]	uint16	1x	Act. Torque [Nm]	P419[0]	float32	2x
Torque Scale [Nm]	P1286[0]	uint16	1x				
Torque max. positive dir.	P1287[0]	uint16	1x				
Torque max. negative dir.	P1288[0]	uint16	1x				

Σ [Bit]	160		Σ [Bit]	144	
Σ [Words]	10		Σ [Words]	9	
Σ [Bytes]	20		Σ [Bytes]	18	

Example: For this example mapping with 20 Byte (10 words) I/O data you can use PPO5 telegram in your hardware configuration (20 Byte I/O + PKW channel).

You have to import the GSDML device file into the TIA portal. After that, the ServoOne/LeviOne is known in the hardware catalogue with all supported telegrams. The TIA Portal will handle the ServoOne/LeviOne as a periphery so it will generate a start/end address when using this device (*Figure 1*).

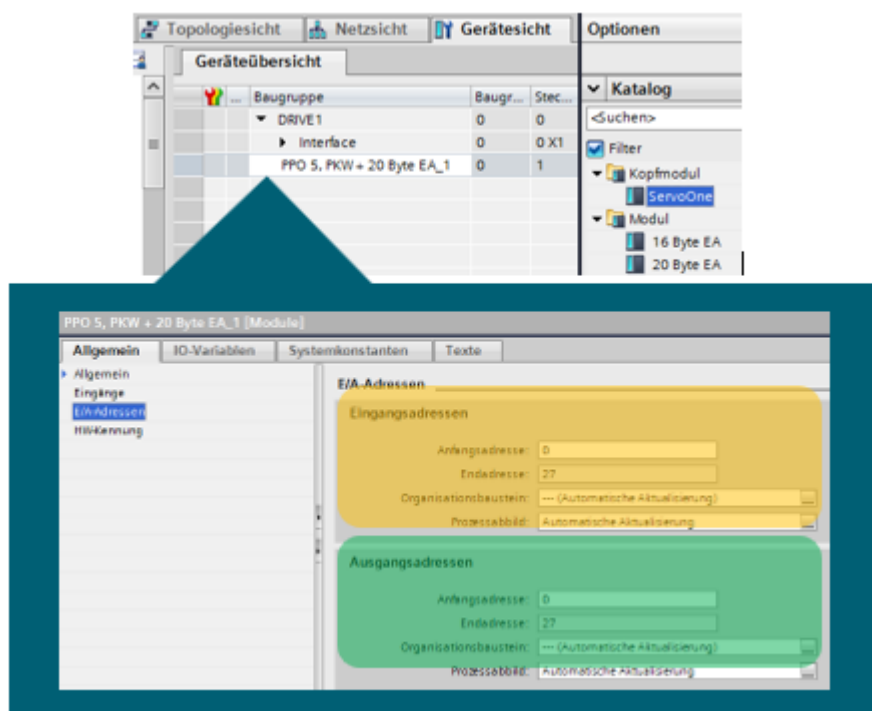
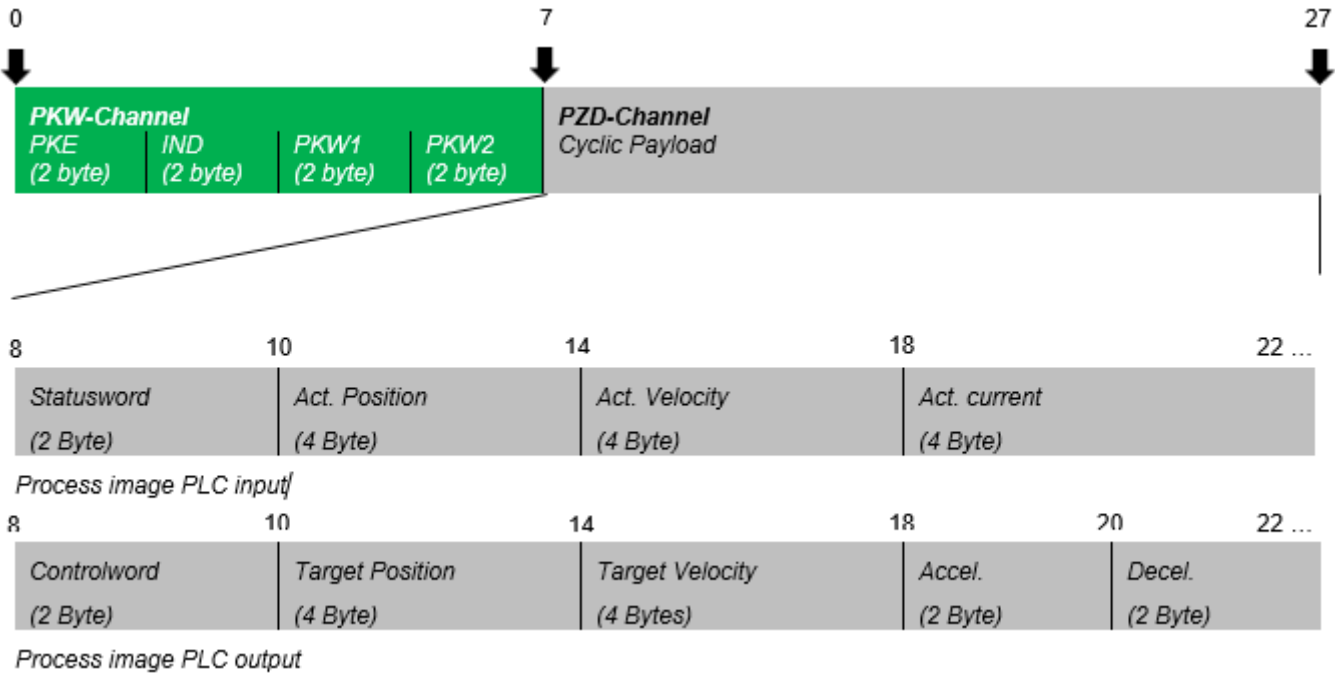


Figure 1.1: Hardware configuration with start/end address

You can find your process data in the input/output process image of the PLC. The process data starts at the configured address and have the same order like in the ServoOne mapping. As default, the SIEMENS S7 PLC handles the data with little endian bit order.



If you want to use DP-V1 acyclic service, you will only need the HardwareID of the depending drive. It is assigned by the TIA Portal. You can find this HW-ID in the Device Configuration → Drive → Properties → Hardware Identifier or since TIA15.1 in the System Constants:

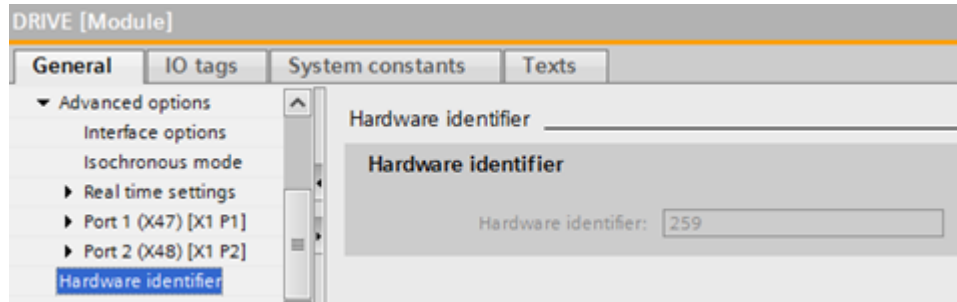


Figure 1.2: Hardware-ID of the slave

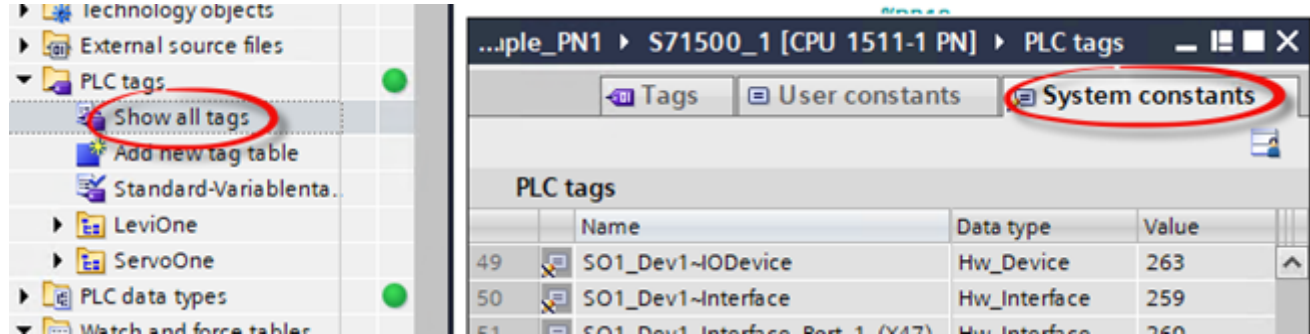


Figure 1.2: Hardware-ID of the slave (since TIA15.1)

2.4 LeviOne Operation Modes

This example contains code for using the 6D functionality of LeviOne magnet bearing for LeviSpin spindle. The operation modes are defined as following:

- **Vibration Mode:** In vibration mode, a one-dimensional vibration of a given basic frequency (in Hz determined by *FREQV* and the amplitude values in *SOLLF* in nm) is generated.
- **Orbital Mode:** In orbital motion mode, a circular reference trajectory for radial directions [μm] is generated. When enabling orbital, a “spiral” motion from center position to a circular orbit is performed; when orbital motion is disabled, a spiral motion from circular orbit to center position is performed.
- **Tool Change Mode:** When the pneumatic tool change mechanism releases the tool holder, it exerts a large force in axial direction that the axial magnetic bearing is unable to counteract. Therefore, the magnetic bearing controller has to change into a special mode for tool change

More information and restrictions to each mode you can get from the user manual.

When using PROFIBUS/PROFINET fieldbus interface with LeviOne, you have to choose a matching telegram and map the needed process data, as well as ServoOne. In this example, the telegram *112 = PKW + 28 Byte IO* (with PKW channel) was used:

Tab. 3: Example for LeviOne 6D Handling

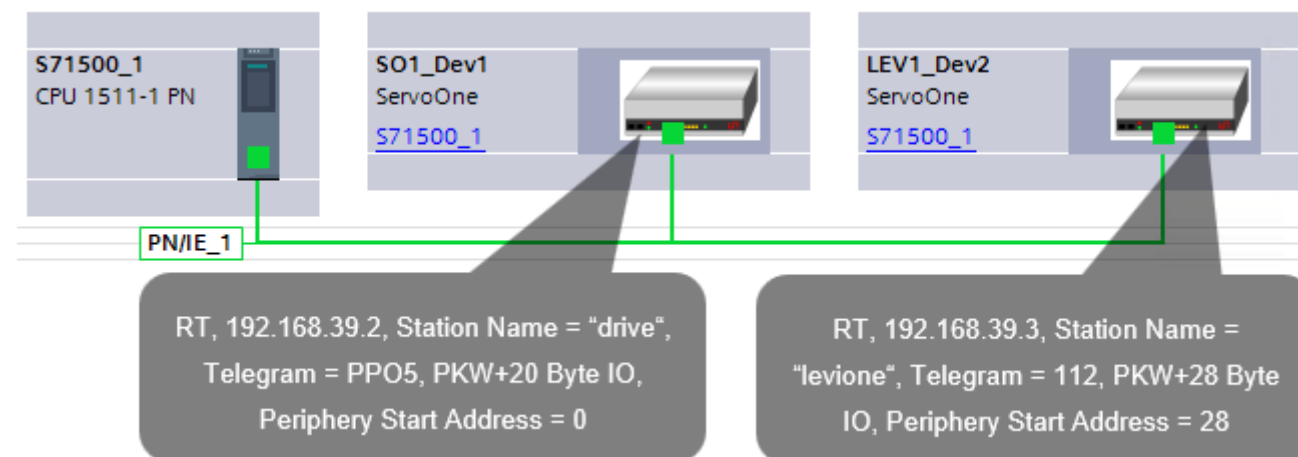
P915 with PCON (to slave)			P916 with PCON (from slave)		
<i>Object</i>	<i>Parameter</i>	<i>Datatype</i>	<i>Object</i>	<i>Parameter</i>	<i>Datatype</i>
Controlword Control	P967[0]	uint16	Statusword Control	P968	uint16
STEU2 (6D)	P3811[0]	uint16	STAT2 (6D)	P3810	uint16
FREQV [Hz]	P3407[0]	float32			
SOLF2 [nm]	P3832[0]	int32			
DIR_X	P3815[0]	float32			
DIR_Y	P3816[0]	float32			
DIR_Z	P3817[0]	float32			
Σ [Bit]		192	Σ [Bit]		32
Σ [Words]		12	Σ [Words]		2
Σ [Bytes]		24	Σ [Bytes]		4

When using a telegram with PKW channel, the process data are shifted by 8 bytes from the start address, same at ServoOne.

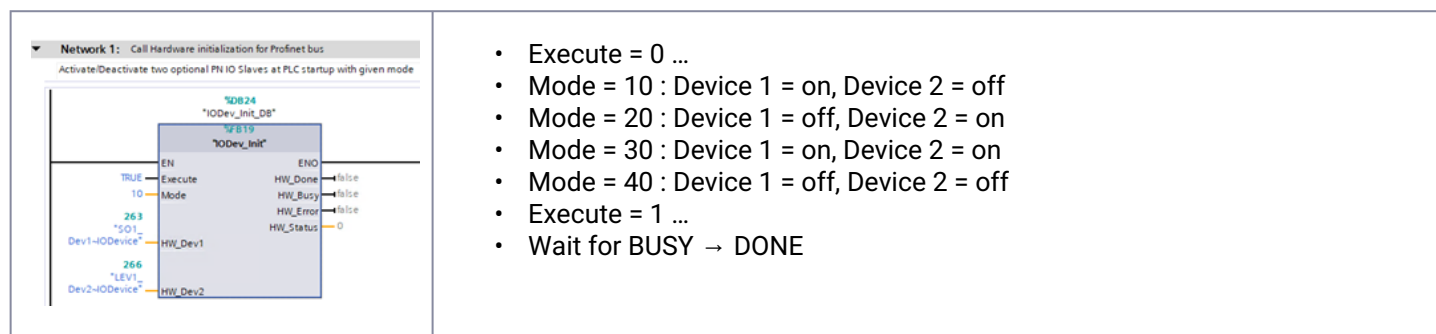
2.5 KEBA Example Projects

This project was built up with a PROFINET connection but it is also useable for PROFIBUS – The mechanism is the same. The programs are programmed in language FBD/ST. The basic interfaces for the KEBA function blocks are always the PN/DP Status-/Controlword and/or the PKW In/Out PKE + IND + Payload which are directly addressed via Periphery I/O address (PROFINET and PROFIBUS) or SFC14/15 (only with PROFIBUS).

2.6 Hardware Configuration



Mostly, this example project will be used with ServoOne drive(s). Because of that, the LeviOne is an optional device. Initial, only the ServoOne will be activated at the PLC startup process, the LeviOne will be initially deactivated by the function block “**IODev_Init**” in OB1. This FB is using the system function “**D_ACT_DP**”. When a PN slave is deactivated, the PLC will not access its process data any more. This FB can be used freely to deactivate and activate two devices by giving a mode and hardware IDs:



When changing the mode, the port wiring (Topology View) has to be edit (ports connected or not). As device Input *HW_Dev1/2*, you can use system constants or directly the *HW_DEVICE ID*. If you don't want to use the slave selection, just delete this FB and only use on slave device. The activation of slaves is time consuming. It takes some seconds until BUSY will changing to DONE.

2.7 Organization Blocks

This example contains the following organization blocks, which calls the variant function blocks.

Tab. 5: Overview of implemented OBs

<u>Program</u>	<u>Description</u>
<i>SO_Movement [OB1]</i>	Contains Function Blocks for Handling ServoOne (Power On/Off, Operation Mode, Homing, (Table)Positioning, ..). All FBs here can communicate via handshake by each other.
<i>SO_Communication [OB123]</i>	ServoOne: Read and write parameter via PKW, DPV1 service and TCP/IP, read and write string via PKW, error handling, service handling
<i>LEV_Function [OB124]</i>	LeviOne: Handling of the Magnetic Bearing / 6D Functionality, Read and write parameter via PKW, DPV1

2.8 Example Function Blocks

The function blocks in this project for TIA Portal are programmed in language SCL. Each instance need a data block, which here are organized in groups. The function blocks are only examples to show, how it can be handled.

KEBA recommend to build up your own project with your own solution without adopt this examples.

2.9 Watch List

The example program contains two watch tables “*WatchList1_SO*” and “*WatchList2_LEV*”:

- Accessing function blocks
- Handle the variant function blocks
- Show actual values of the function blocks

Name	Address	Display format	Monitor value	Monitor with trig...	Modify with trigge...
// Function Selector for LeviOne					
"LEV_Selector_Function"	%MB7	DEC+/-	0	Permanent	Permanent
// LeviOne Function handling					
"LEV_Control_DB".PNDP_Statusword		DEC	0	Permanent	Permanent
"LEV_Control_DB".ToolChangeProcessDone		Bool	FALSE	Permanent	Permanent
"LEV_Control_DB".Mode_Orbital		Bool	FALSE	Permanent	Permanent
"LEV_Control_DB".Mode_ToolChange		Bool	FALSE	Permanent	Permanent
"LEV_Control_DB".Mode_Vibration		Bool	FALSE	Permanent	Permanent
"LEV_Control_DB".EnableControl		Bool	FALSE	Permanent	Permanent
"LEV_Control_DB".VibrationFrequency		Floating-point nu...	0.0	Permanent	Permanent
"LEV_Control_DB".VibrationAmplitude		DEC+/-	0	Permanent	Permanent
"LEV_Control_DB".Direction_X		Floating-point nu...	0.0	Permanent	Permanent
"LEV_Control_DB".Direction_Y		Floating-point nu...	0.0	Permanent	Permanent

Name	Ad...	Display format	Monitor value	Monitor...	Mo...	Modify value
// ServoOne Function Handling						
// Inputs						
"SO_Power_DB".HMD		DEC	259	Perman...	Per...	
"SO_Power_DB".ControlMode		DEC	0	Perman...	Per...	0
"SO_Power_DB".TScale		DEC	1000	Perman...	Per...	1000
"SO_Power_DB".TMaxPositive		DEC	1000	Perman...	Per...	1000
"SO_Power_DB".TMaxNegative		DEC	1000	Perman...	Per...	1000
"SO_Power_DB".IQuickstop		Bool	TRUE	Perman...	Per...	TRUE
"SO_Power_DB".EnablePower		Bool	FALSE	Perman...	Per...	FALSE
"SO_Power_DB".ExecuteHoming		Bool	FALSE	Perman...	Per...	FALSE
"SO_Power_DB".SpeedModePCON		Bool	FALSE	Perman...	Per...	
"SO_Power_DB".SetNewValue		Bool	FALSE	Perman...	Per...	
"SO_Power_DB".AxisErrorReset		Bool	FALSE	Perman...	Per...	FALSE

Figure 2: Prepared watch lists for ServoOne and LeviOne

2.10 Accessing the Example Function Blocks

If you want to use more instance of function blocks in an OB which are modifying the output process data (controlword, target values, PKW-I/O, ...), you have to handle the access of this function blocks in a consistent way. Otherwise, each function block will modify the same output data in the same time or maybe accessing the same parameter. In this example project, you have to handle the access manually through selector variables for each OBs.

2.11 Program Selectors

With variable "SO_Selector_Movement" in the watch list "WatchList1_SO", you can handle the function block access in the OB "SO_Movement":

- Selector = 0: Use SO_PLCTablePos
- Selector = 1: Use SO_MoveAbsolute

With variable "SO_Selector_Communication" in the watch list "WatchList1_SO", you can handle the function block access in the OB "SO_Communication":

- Selector = 0: Disabled
- Selector = 1: SO Read Integer/Float Parameter via PKW

- Selector = 2: SO Write Integer/Float Parameter via PKW
- Selector = 3: SO Read String Parameter via PKW
- Selector = 4: SO Write String Parameter via PKW
- Selector = 5: SO Service Functions
- Selector = 6: Read Integer/Float Parameter via DP-V1
- Selector = 7: Write Integer/Float Parameter via DP-V1
- Selector = 8: Read Integer/Float Parameter via TCP/IP X3 Interface
- Selector = 9: Write Integer/Float Parameter via TCP/IP X3 Interface
- Selector = 10: Read String Parameter via TCP/IP X3 Interface
- Selector = 11: Write String Parameter via TCP/IP X3 Interface

With variable "*LEV_Selector_Function*" in the watch list "*WatchList2_LEV*", you can handle the function block access in the OB "*LEV_Function*":

- Selector = 0: Disabled
- Selector = 1: Read via PKW channel
- Selector = 2: Write via PKW channel
- Selector = 3: Read parameter value via DPV1
- Selector = 4: Write parameter value via DPV1

Info:

- *DPV1/PKW access can be done in parallel for both device due to different IO addresses / HW-IDs*
- When using the *SO_ReadActError* function block, make sure that only this FB has access to PKW I/O and no other PKW Read/Write function block. For that, use the "*FBLock*" output for locking following function blocks.
- Make also sure that you have no parallel access of cyclic process data and PKW Write to the same parameter number otherwise the value of this parameter will toggle.

3 Example Function Blocks

3.1 SO_Power

Overview:

General axis handling for one axis. You can use this FB for single movements or in combination with "SO_PLCTablePos" or "SO_MoveAbsolute".

- PCON, SCON, VFCON, TCON **will be set by this FB**
- FB checks at startup, if PLC mode at drive side is active
- Cyclic and Acyclic communication
- Uses *SO_ReadDPV1* function
- Switch on a single axis
- Reset axis error (rising edge)
- Handle quick stop
- Handle axis jog
- Set torque scale (online)
- Set control mode (online)
- Run homing procedure
- Set new target value
- Feedback outputs
- According to Flow Chart 1

Compatibility / Use with:

- ServoOne (*Tested*)
- ServoOne Safety (*Tested*)
- ServoOne junior
- ServoOne with TwinSync

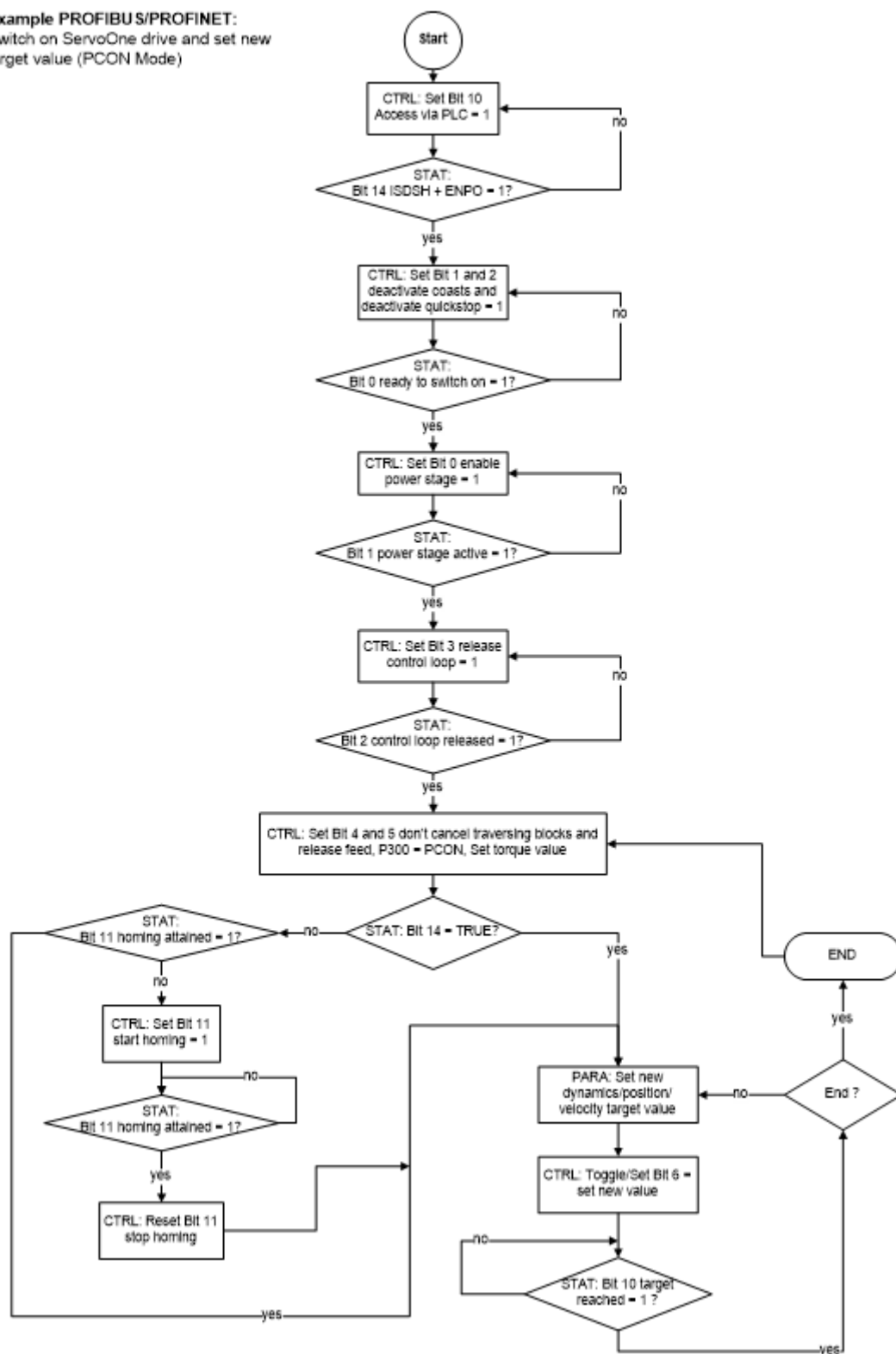
Inputs	Datatype	Description
HWID	HW_IO	Hardware-ID (UINT), given by TIA system configuration for the PROFINET Salve, value 0 is not allowed
PNDP_Statusword	UINT	Statusword 1 of the drive, read by process data input (direct address (%IWx, PEW) or System Function (SFC14)

ControlMode	USINT	<p>Set control mode numeric value:</p> <ul style="list-style-type: none"> • 0 = PCON • 1 = SCON • 2 = VFCON • 3 = TCON <p>The mode will be written to the drive via DPV1 acyclic service. This is always possible, when FB is generally active (also when power stage is active)</p>
EnablePower	BOOL	When TRUE the FB will run the power on sequence. After that, all other functions are possible (homing, set new value). When FALSE, all feedback outputs are FALSE and axis power is disabled
AxisErrorReset	BOOL	Reset Axis Error, always possible when FB is generally enabled (EN)
ExecuteHoming	BOOL	When TRUE, start homing. The configured homing method in the ServoOne will be used. Only possible in Mode PCON.*
SpeedModePCON	BOOL	When FALSE the drive works in standard positioning mode. Otherwise, the drive works in velocity mode (endless positioning). Then the target velocity will be used as target. Only possible in mode PCON.*
SetNewValue	BOOL	PCON: Toggle this bit when target is reached to set new value. SCON/VFCON: TRUE = activate reference
JogPositive	BOOL	Jog axis in positive direction when enabled (with interlock), only <i>JogSpeed1</i> in P1268[0] used
JogNegative	BOOL	Jog axis in negative direction when enabled (with interlock), only $-1 * \text{JogSpeed1}$ in P1268[0] used
/Quickstop	BOOL	Low active: 0 = Quickstop active, 1 = Quickstop not active, <u>only possible while FB is enabled</u>
SetImmediately	BOOL	TRUE = take new set point immediately while moving
TScale	UINT	Torque online scale [thousandth] of nominal motor torque, online changeable (0 ... 1000), influences P332[0] <i>T_MaxScale</i> (online)
TMaxPositive	UINT	Torque positive direction limit [thousandth] of nominal motor torque (0 ... 1000), influences P331[0] <i>T_MaxPos</i> (online)
TMaxNegative	UINT	Torque negative direction limit [thousandth] of nominal motor torque (0 ... 1000), influences P330[0] <i>T_MaxNeg</i> (online)

Outputs	Datatype	Description
PNDP_Controlword	UINT	Controlword for the drive, written by process data output (direct address (%QWx, PEW) or System Function (SFC15))
ActiveControlMode	STRING	Indication, Active control mode in the drive
TTScale	UINT	Target Torque value to drive
TTMaxPositive	UINT	Target Torque value positive direction to drive
TTMaxNegative	UINT	Target Torque value negative direction to drive
PwrBusy	BOOL	When TRUE, running power on sequence
PwrStatus	BOOL	When TRUE, the axis is switched on
HomeBusy	BOOL	When TRUE, homing procedure is running
HomeDone	BOOL	When TRUE, homing procedure was done. The homing state of the drive can be read via P151 Bit 15.
AxisError	BOOL	When TRUE, an axis error occurred
TargetReached	BOOL	When TRUE, target position/velocity reached
MotionAckn	BOOL	When toggling, a new request was acknowledged by the drive
FBError	BOOL	When TRUE an error occurred in the Function Block
ErrorID	STRING	Contains some internal Error messages

To switch on power stage of ServoOne, you have to run through a routine like in the following flow chart (according to PROFIdrive):

Example PROFIBUS/PROFINET:
Switch on ServoOne drive and set new target value (PCON Mode)



Flow Chart 1: Example switch on routine for PCON

3.1.1 PCON: Switching between position and velocity mode

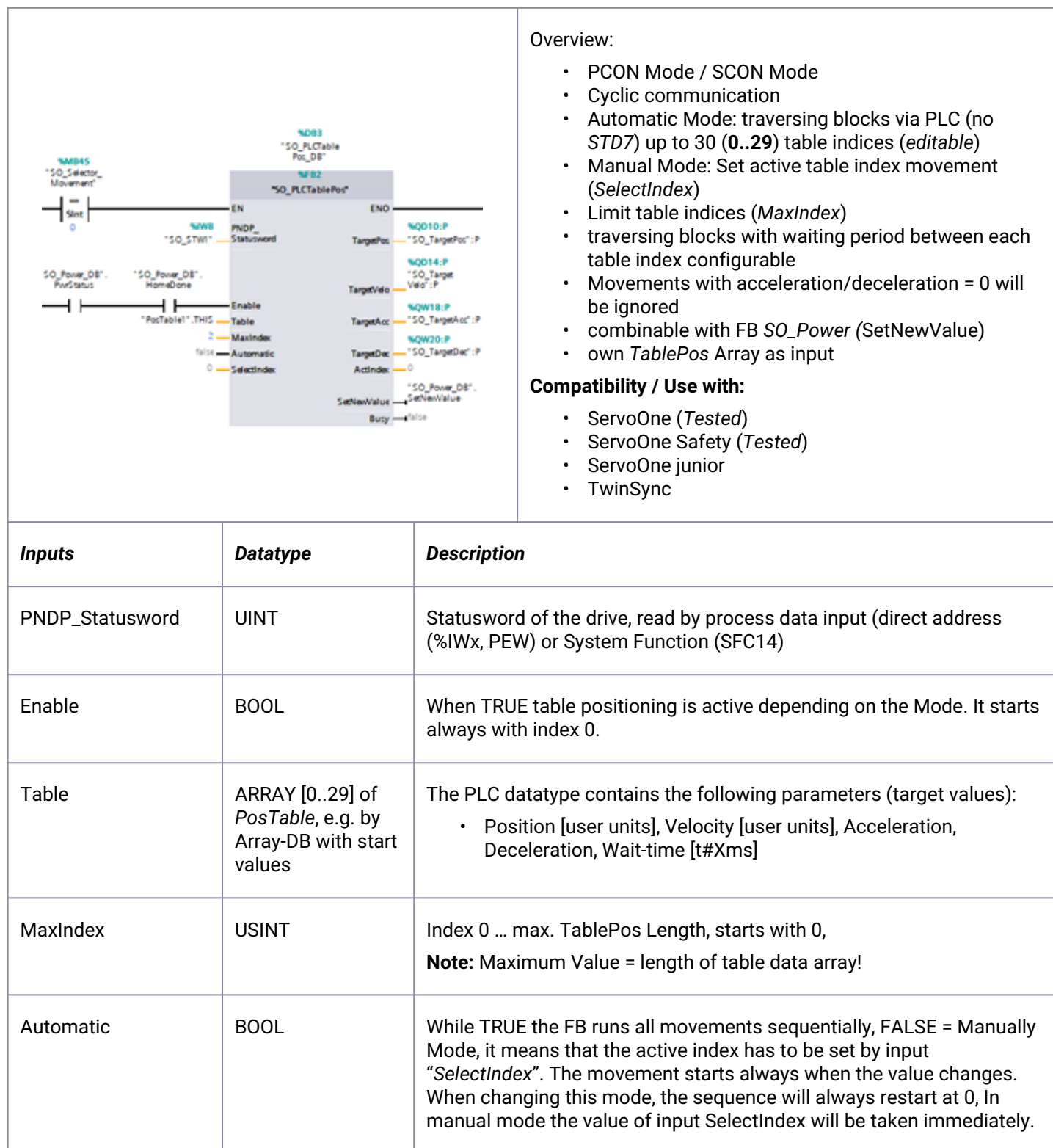
With *SO_Power* function block it is possible to switch between raw position mode (PCON) and PCON velocity mode (endless positioning). For that, the drive remains in PCON position mode all the time. The following example shows, how to do it with *SO_Power* (and additional *SO_MoveAbsolute* or *SO_PLCTablePos*)

Note: If you are using this speed mode in combination with software limit switches, make sure that the endless positioning is possible between this end positions (or deactivate with 0), otherwise the drive will stop or not start.

Tab. 5: Operation mode switching process

Step	Procedure
1	Set the control mode 'PCON' with <code>Input ControlMode</code> and check with output <code>ActiveControlMode</code> , if the drive activates this mode (<code>P300[0]</code>).
2	<i>Optional:</i> Set the torque values. They can be changed also online every time
3	Enable power stage with input <code>EnablePower := TRUE</code> and wait for output <code>PwrStatus = TRUE</code> means power stage enabled
4	<i>Optional:</i> Start homing procedure with rising edge <code>ExecuteHoming := TRUE</code> and wait for homing done with <code>HomeDone = TRUE</code>
5	<i>Optional:</i> Finish homing when attained with <code>ExecuteHoming := FALSE</code>
<i>Decide if Velocity Mode or Position Mode ...</i>	
PCON Velocity Mode:	
6	Set target velocity via <code>P1277</code> , acceleration via <code>P1278</code> and deceleration via <code>P1279</code> (can also be handled by function block <code>SO_PLCTablePos</code> or <code>SO_PLCTablePos</code>)
7	<code>SpeedModePCON := TRUE</code> will run a velocity by using the position mode (PCON) Execute input (rising and falling edge) <code>SetNewValue</code> (can also be handled by function block <code>SO_PLCTablePos</code> or <code>SO_PLCTablePos</code>) → Result: PN Controlword Bit 6 will be toggled
8	Wait for output <code>TargetReached = TRUE</code> , indicates that the target velocity is reached
9	You can set a new target velocity <code>P1277</code> while moving by executing input <code>SetNewValue</code> (rising and falling edge, can also be handled by function block <code>SO_PLCTablePos</code> or <code>SO_PLCTablePos</code>)
PCON Position Mode:	
10	While moving, <code>SpeedModePCON := FALSE</code> means standard positioning mode active
11	Set new dynamic values and execute input (rising and falling edge) <code>SetNewValue</code> (can also be handled by function block <code>SO_PLCTablePos</code> or <code>SO_PLCTablePos</code>) → Result: PN Controlword Bit 6 will be toggled
12	Wait for output <code>TargetReached = TRUE</code> , indicates that the target position is reached

3.2 SO PLCTablePos



SelectIndex	USINT	Index <= maxIndex <= max. TablePos array (here: 29), otherwise value will be set to 0!, Movements with acceleration/deceleration = 0 will be ignored
Outputs	Datatype	Description
TargetPos	DINT	Process data output for target position (cyclic data)
TargetVelo	DINT	Process data output for target velocity (cyclic data)
TragetAcc	UINT	Process data output for acceleration (cyclic data)
TargetDec	UINT	Process data output for deceleration (cyclic data)
ActIndex	USINT	Actual table index
SetNewValue	BOOL	This bit indicates that a new target position can be set. When using SCON Mode, this output is not needed; target velocities will be run directly without start signal when power stage is enabled!
Busy	BOOL	When TRUE, the FB is running

Depending on the operation mode, the table can be filled up with the needed target values. If using SCON mode, only velocity, acceleration and deceleration is needed. When using PCON mode, an additional target position is needed:

PosTable1				
		Name	Data type	Start value
1		PosTable1	Array[0..29] of " ... "	
2		PosTable1[0]	"PosTable"	
3		Positon	DInt	0
4		Velocity	DInt	600
5		Acceleration	UInt	600
6		Deceleration	UInt	600
7		Wait	Time	t# 5s
8		PosTable1[1]	"PosTable"	
9		Positon	DInt	0
10		Velocity	DInt	10
11		Acceleration	UInt	600
12		Deceleration	UInt	600
13		Wait	Time	t# 2s

Figure 2.1: Example Table for SCON Mode and Wait-Time

In every case, the wait time is optionally. If Wait = t#0s, the target values for this entry will be run immediately. In this case, take care about your PLC cycle time, target acceleration / deceleration and target reached window.

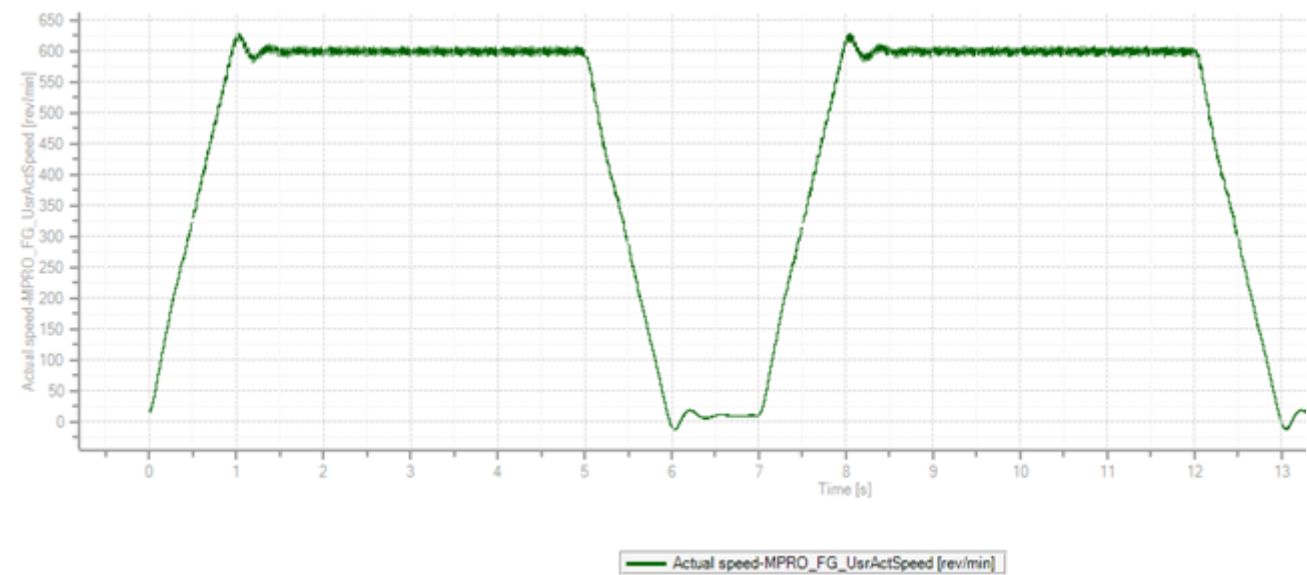


Figure 2.2: SCON Motion Profile Result with Wait-Time

PosTable1			
	Name	Data type	Start value
1	▼ PosTable1	Array[0..29] of *Pos...	
2	▼ PosTable1[0]	*PosTable"	
3	Positon	DInt	10000
4	Velocity	DInt	50
5	Acceleration	UInt	500
6	Deceleration	UInt	300
7	Wait	Time	T#1s
8	▼ PosTable1[1]	*PosTable"	
9	Positon	DInt	-100000
10	Velocity	DInt	50
11	Acceleration	UInt	300
12	Deceleration	UInt	300
13	Wait	Time	T#2s
14	▼ PosTable1[2]	*PosTable"	
15	Positon	DInt	50000
16	Velocity	DInt	50
17	Acceleration	UInt	300
18	Deceleration	UInt	300
19	Wait	Time	T#3S

Figure 3.1: Example Table for PCON Mode and Wait-Time

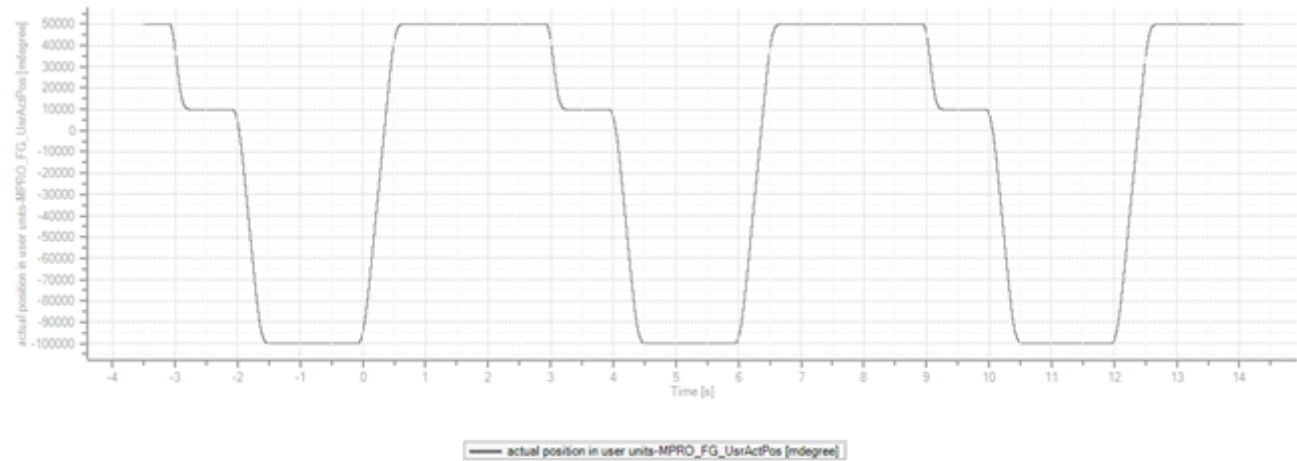


Figure 3.2: PCON Motion Profile Result with Wait-Time

3.3 SO_MoveAbsolute

Overview:

- Optional: use DP-V1 communication (*SO_ReadDPV1*)
- Interacts with *SO_Power*
- Partially edge controlled
- Aborting running movement
- No PLCopen conformity!
- PCON Mode
- Cyclic communication
- Single axis movement

Compatibility / Use with:

- ServoOne (*Tested*)
- ServoOne Safety (*Tested*)
- ServoOne junior

Inputs	Datatype	Description
Execute	BOOL	Starts a new movement with every rising edge with given target and dynamic values, blocked when Abort = TRUE
Abort	BOOL	Always possible when this FB is active, without ramp, rising TRUE = PCON: abort a running positioning command, SCON: Reset ramp generator. When a position command was aborted, a new can be executed.
HWID	HW_IO	Hardware-ID (UINT), given by TIA system configuration for the PROFINET Salve.

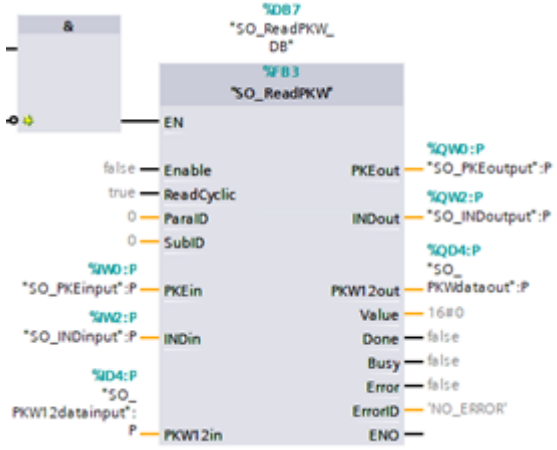
UseDPV1	BOOL	When true, the next execute uses DP-V1 write function for P1278/1279 Acceleration/Deceleration. If this parameters are already mapped, the cyclic communication will always have higher priority (overwriting)
PNDP_Statusword_In	UINT	Statusword 1 of the drive, read by process data input (direct address (%IWx, PEW) or System Function (SFC14), cyclic data
PNDP_Controlword_In	UINT	Controlword 1 for the drive, written by process data output (direct address (%QWx, PEW) or System Function (SFC15) as input for some checks in the FB, cyclic data
Position	DINT	Target position in user units
Velocity	DINT	Target velocity in user units
Acceleration	UINT	Target acceleration in user units
Deceleration	UINT	Target deceleration in user units
Outputs	Datatype	Description
PNDP_Controlword_Out	WORD	Controlword 1 output to drive (cyclic data)
Position_Out	WORD	Commanded position for cyclic process data (cyclic data)
Velocity_Out	WORD	Commanded velocity for cyclic process data (cyclic data)
Acceleration_Out	DWORD	Commanded acceleration for cyclic process data if DP-V1 is not in use
Deceleration_Out	BOOL	Commanded deceleration for cyclic process data if DP-V1 is not in use
SetNewValue	BOOL	Start Command for SO_Power / run new value
Busy	BOOL	When TRUE the FB/positioning is busy
Error	BOOL	When TRUE an error occurred
ErrorID	STRING	Short description of the error when Error = TRUE

3.4 SO_ReadDPV1

			<p>Overview:</p> <ul style="list-style-type: none"> • Acyclic communication DP-V1 • Structed Parameters are not supported! • Only single parameters possible • It doesn't matter which operation mode is configured. • A telegram with PKW channel is not necessary • Use only one time per drive or add an access control. • Read value cyclic or once with every rising edge • Supports WORD, DWORD and REAL à you have to convert the output value to your target data type. • Needed: Import also PLC datatypes "BMPAReqRd" and "BMPAResRd" • Only possible datatypes: WORD, DWORD, REAL, BYTE, in case of other datatypes (e.g. UINT) note the possible value range of the target parameter (overflow) <p>Compatibility / Use with:</p> <ul style="list-style-type: none"> • ServoOne (Tested) • ServoOne Safety (Tested) • ServoOne junior • LeviOne junior (Tested) 		
Inputs	Datatype	Description			
Read	BOOL	With every rising edge, the value will be read out one time, FALSE = off / reset			
ReadCyclic	BOOL	While TRUE, the value will be read out cyclically, default = FALSE			
HWID	HW_IO	Hardware-ID (UINT), given by TIA system configuration for the PROFINET Salve.			
ParaID	UINT	Parameter Number (like in the DriveManager 5), this value will be checked by the FB; will be taken into account with next rising edge			
SubID	BYTE	Sub index (like in the DriveManager 5), Default = 0; can be changed while FB is busy, will be taken into account with next rising edge			
Datatype	USINT	Has always to be fit before read: Datatype to be read: 0 = WORD (default), 1 = DWORD, 2 = REAL, 3 = BYTE			
Outputs	Datatype	Description			

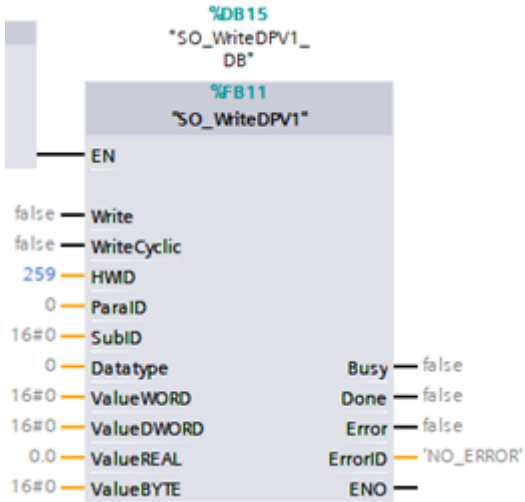
ValueWORD	WORD	Output value WORD when Datatype = 0
ValueDWORD	WORD	Output value DWORD when Datatype = 1
ValueREAL	DWORD	Output value REAL when Datatype = 2
ValueBYTE	BYTE	Output value BYTE when Datatype = 3
Busy	BOOL	While TRUE the FB is busy
Done	BOOL	While TRUE, the read process is done
Error	BOOL	While TRUE, the FB has an error
ErrorID	STRING	Short description of the error when Error = TRUE

3.5 SO_ReadPKW

			<p>Overview:</p> <ul style="list-style-type: none"> • Acyclic communication • Single Parameters or Array-Parameters possible • It doesn't matter which operation mode is configured. • A telegram with PKW channel is necessary in the S7 hardware configuration • Use only one time per drive or add an access control. • Parameter-ID and Sub-ID are online changeable while FB is enabled. With this you are able to build a ring buffer for more than one parameter read access. • You have to convert the output value to your target data type. <p>Compatibility / Use with:</p> <ul style="list-style-type: none"> • ServoOne (Tested) • ServoOne Safety (Tested) • ServoOne junior • LeviOne (Tested)
Inputs	Datatype	Description	
Enable	BOOL	With every rising edge, the value will be read out one time, FALSE = off / reset	

ReadCyclic	BOOL	While TRUE, the value will be read out cyclically, default = TRUE (keeps conformity to older versions)
ParaID	UINT	ServoOne Parameter which have to be read out, Range: 0 ... 4095, otherwise Error = True
SubID	UINT	Array sub index; the FB automatically compensates the offset automatically (+1). Range: 0 ... 254 otherwise wrong access.
PKEin	WORD	Input address of identifier PKE (2 bytes)
INDin	WORD	Input address of sub index value IND (2 bytes)
PKW12in	DWORD	Input address of payload PKW1+PKW2 (4 bytes)
Outputs	Datatype	Description
PKEout	WORD	Output address of identifier PKE (2 bytes)
INDout	WORD	Output address of sub index value IND (2 bytes)
PKW12out	DWORD	Output address of payload PKW1+PKW2 (4 bytes)
Value	DWORD	Value read from drive
Valid	BOOL	While TRUE, the read cycle is done
Busy	BOOL	While TRUE, the FB is busy
Error	BOOL	When TRUE an error occurred
ErrorID	STRING	Short description of the error when Error = TRUE

3.6 SO_WriteDPV1

			<p>Overview:</p> <ul style="list-style-type: none"> • Acyclic communication DPV1 • Structed Parameters are not supported! • Only single parameters possible • It doesn't matter which operation mode is configured. • A telegram with PKW channel is not necessary • Use only one time per drive or add an access control. • Read value once time with every rising edge • Supports BYTE/WORD, DWORD and REAL • Needed: Import also PLC datatypes "BMPAReqWr" and "BMPAResWr" • Only possible datatypes: WORD, DWORD, REAL, BYTE, in case of other datatypes (e.g. UINT) note the possible value range of the target parameter (overflow) <p>Compatibility / Use with:</p> <ul style="list-style-type: none"> • ServoOne (Tested) • ServoOne Safety (Tested) • ServoOne junior • LeviOne (Tested) 		
Inputs	Datatype	Description			
Write	BOOL	With every rising edge, the value will be written one time, FALSE = off / reset			
WriteCyclic	BOOL	While TRUE, the value will be written cyclically, default = FALSE			
HWID	HW_IO	Hardware-ID (UINT), given by TIA system configuration for the PROFINET Salve.			
ParaID	UINT	Parameter Number (like in the DriveManager 5), this value will be checked by the FB; will be taken into account with next rising edge of "Write"			
SubID	BYTE	Sub index (like in the DriveManager 5), Default = 0; can be changed while FB is busy, will be taken into account with next rising edge of "Write"			
Datatype	USINT	Has always to be fit: Datatype to write: 0 (default) = BYTE/WORD, 1 = DWORD, 2 = REAL, 3 = BYTE			
ValueWORD	WORD	Target value when "Datatype" = 0			

ValueDWORD	DWORD	Target value when "Datatype" = 1
ValueREAL	REAL	Target value when "Datatype" = 2
ValueBYTE	BYTE	Target value when "Datatype" = 3
Outputs	Datatype	Description
Busy	BOOL	While TRUE the FB is busy
Done	BOOL	While TRUE, the read process is done
Error	BOOL	While TRUE, the FB has an error
ErrorID	STRING	Short description of the error when Error = TRUE

3.7 SO_WritePKW

			<p>Overview:</p> <ul style="list-style-type: none"> • Acyclic communication • Single Parameters or Array-Parameters possible • It doesn't matter which operation mode is configured. • A telegram with PKW channel is necessary in the S7 hardware configuration • Use only one time per drive or add an access control. • Parameter-ID and Sub-ID are online changeable while FB is enabled. With this you are able to build a ring buffer for more than one parameter write access. • You have to convert the input value to your target data type (e.g. REAL). <p>Compatibility / Use with:</p> <ul style="list-style-type: none"> • ServoOne (Tested) • ServoOne Safety (Tested) • ServoOne junior • LeviOne (Tested)
Inputs	Datatype	Description	
Enable	BOOL	With every rising edge, the value will be written one time, FALSE = off / reset	

WriteCyclic	BOOL	While TRUE, the value will be written cyclically, default = TRUE (keeps conformity to older versions)
Value	DWORD	Send value to drive. If target datatype is REAL you have to convert this value first to IEEE754 format.
ParaID	UINT	ServoOne Parameter which have to be read out, Range: 0 ... 4095, otherwise Error = True
SubID	UINT	Array sub index; the FB automatically compensates the offset automatically (+1). Range: 0 ... 254 otherwise wrong access.
PKEin	WORD	Input address of identifier PKE (2 bytes)
INDin	WORD	Input address of sub index value IND (2 bytes)
PKW12in	DWORD	Input address of payload PKW1+PKW2 (4 bytes)
Outputs	Datatype	Description
PKEout	WORD	Output address of identifier PKE (2 bytes)
INDout	WORD	Output address of sub index value IND (2 bytes)
PKW12out	DWORD	Output address of payload PKW1+PKW2 (4 bytes)
Valid	BOOL	While TRUE, the write cycle is done
Busy	BOOL	While TRUE, the FB is busy
Error	BOOL	When TRUE an error occurred
ErrorID	STRING	Short description of the error when Error = TRUE

3.8 SO_ReadActError

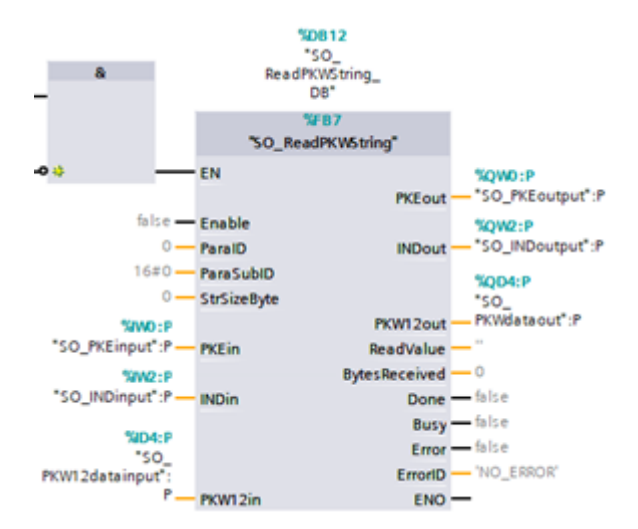
<div></div>			<div>Overview:</div> <ul style="list-style-type: none">• This FB uses <i>SO_PKWRead</i> function block• It doesn't matter which operation mode is configured.• A telegram with PKW channel is necessary in the S7 hardware configuration• If the drive error is confirmed the error values are reset to zero automatically (blocking mode = TRUE)• output "FBLOCK" for PKW access handle when using several PKW FBs <div>Compatibility / Use with:</div> <ul style="list-style-type: none">• ServoOne (<i>Tested</i>)• ServoOne Safety (<i>Tested</i>)• ServoOne junior (<i>Tested</i>)• LeviOne (<i>Tested</i>)		
Inputs		Datatype	Description		
Enable		BOOL	While TRUE (rising edge), the latest error will be read out when Bit 3 of PNDP_Statusword is true, otherwise outputs are inactive (values are zero); note the blocking mode		
Cyclic		BOOL	While TRUE = blocking mode, the FB is then cyclically monitoring the error register of the slave, the PKW then is blocked for other FBs à FBLOCK output; When FALSE = non-blocking mode, the FB is reading the error register one time with every rising edge of "Enable" input; this input is changeable online		
PNDP_Statusword		UINT	Statusword 1 of the drive, read by process data input (direct address (%IWx, PEW) or System Function (SFC14)		
PKEin		WORD	Input address of identifier PKE (2 bytes)		
INDin		WORD	Input address of sub index value IND (2 bytes)		
PKW12in		DWORD	Input address of payload PKW1+PKW2 (4 bytes)		
Outputs		Datatype	Description		

ActErrorText	String[254]	Latest Error Text from device
ActErrorID	DWORD	Latest Error ID from device
ActErrorLocation	DWORD	Latest error location from device
ActErrorTime	DWORD	Latest timestamp [s] of error from device
ActSafetyCode	DWORD	Latest alarm/error code from Safe PLC if available; will be automatically detected, if latest error is an safety error
PKEout	WORD	Output address of identifier PKE (2 bytes)
INDout	WORD	Output address of sub index value IND (2 bytes)
PKW12out	DWORD	Output address of payload PKW1+PKW2 (4 bytes)
Busy	BOOL	While TRUE, the FB is polling the failure bit / reading error objects via PKW
Done	BOOL	TRUE = read process is done, FB is finished, no new errors will be detected!
FBLock	BOOL	TRUE when Bit 3 = TRUE = error in drive; with this output make sure that other PKW Read/Write function blocks are locked so they have no access to the PKW channel.

***Recommendation:**

- You can use the error location and the ID as index for an error list which is stored in a PLC (file / data block ...)
- This FB will take some seconds to read out all data, indicated by output “Busy” and “Done”

3.9 SO_ReadPKWString

		<p>Overview:</p> <ul style="list-style-type: none">• It doesn't matter which operation mode is configured.• A telegram with PKW channel is necessary in the S7 hardware configuration• Edge controlled• Contains a 200ms wait time between every 4 byte access <p>Compatibility / Use with:</p> <ul style="list-style-type: none">• ServoOne (<i>Tested</i>)• ServoOne Safety (<i>Tested</i>)• ServoOne junior• LeviOne (<i>Tested</i>)
Inputs	Datatype	Description
Enable	BOOL	Rising edge (true) means read out string one time from device. While false, the output values will be reset.
ParaID	UINT	ServoOne Parameter which has to be read out, Range: 0 ... 4095
ParaSubID	BYTE	Array sub index, the FB automatically compensates the offset (+1). Range: 0 ... 254
StrSizeByte	BYTE	Amount of string data which have to be read out (1 ... 254 bytes). Data will be read out by 4 byte blocks. The requested string size can be longer than the real string when the size is not known.
PKEin	WORD	Input address of identifier PKE (2 bytes)
INDin	WORD	Input address of sub index value IND (2 bytes)
PKW12in	DWORD	Input address of payload PKW1+PKW2 (4 bytes)
Outputs	Datatype	Description
PKEout	WORD	Output address of identifier PKE (2 bytes)

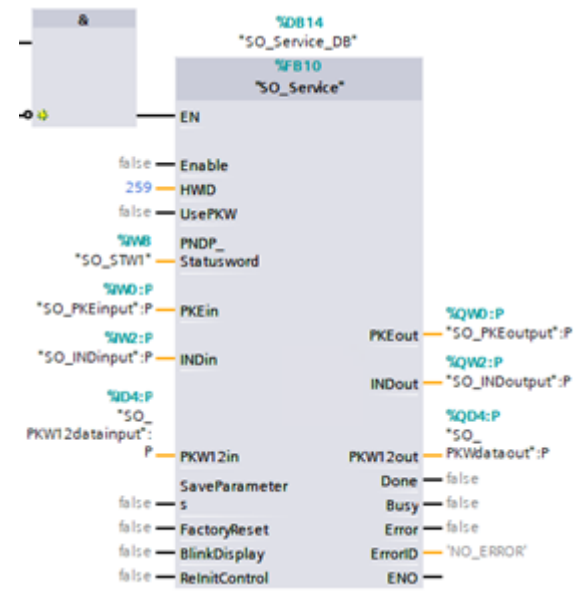
INDout	WORD	Output address of sub index value IND (2 bytes)
PKW12out	DWORD	Output address of payload PKW1+PKW2 (4 bytes)
ReadValue	STRING[254]	Read out value with max. 254 signs
BytesReceived	USINT	Number of received bytes / characters from device
Done	BOOL	While TRUE, the read process is done
Busy	BOOL	While TRUE, the FB is busy
Error	BOOL	When TRUE, an error occurred
ErrorID	STRING	Short description of the error when Error = TRUE

3.10 SO_WritePKWString

			<p>Overview:</p> <ul style="list-style-type: none">• It doesn't matter which operation mode is configured.• A telegram with PKW channel is necessary in the S7 hardware configuration• Edge controlled• Contains a 500ms wait time between every 4 byte access <p>Compatibility / Use with:</p> <ul style="list-style-type: none">• ServoOne (<i>Tested</i>)• ServoOne Safety (<i>Tested</i>)• ServoOne junior• LeviOne (<i>Tested</i>)		
Inputs	Datatype	Description			
Enable	BOOL	Rising edge (true) means write string one time to device. While false the output values will be reset.			
ParaID	UINT	ServoOne Parameter which has to be written, Range: 0 ... 4095			

ParaSubID	BYTE	Array sub index, the FB automatically compensates the offset (+1). Range: 0 ... 254
WriteValue	STRING	String value which has to be written (max. 255 characters). The FB supports the standard ASCII table. Special German letters (Umlaute) are not supported. They will be replaced by "?". If there is no input string, nothing will happen. The input string will be terminated automatically with zero.
PKEin	WORD	Input address of identifier PKE (2 bytes)
INDin	WORD	Input address of sub index value IND (2 bytes)
PKW12in	DWORD	Input address of payload PKW1+PKW2 (4 bytes)
Outputs	Datatype	Description
PKEout	WORD	Output address of sub index value IND (2 bytes)
INDout	WORD	Output address of payload PKW1+PKW2 (4 bytes)
PKW12out	DWORD	Output address of payload PKW1+PKW2 (4 bytes)
Done	BOOL	While TRUE, the write cycle is done
Busy	BOOL	While TRUE, the FB is busy.
Error	BOOL	When TRUE an error occurred
ErrorID	STRING	Short description of the error when Error = TRUE

3.11 SO_Service

		<p>Overview:</p> <ul style="list-style-type: none">• Can use DPV1 and PKW communication• It doesn't matter which operation mode is configured.• A telegram with PKW channel is necessary in the S7 hardware configuration• Partially edge controlled• Uses <i>SO_ReadPKW</i> and <i>SO_WritePKW</i> function• Uses <i>SO_ReadDPV1</i> and <i>SO_WriteDPV1</i> function• Always one input is allowed, otherwise the FB will block with Error = TRUE / ErrorID, Reset when all service inputs = FALSE <p>Compatibility / Use with:</p> <ul style="list-style-type: none">• ServoOne• ServoOne Safety (<i>Tested</i>)• ServoOne junior (<i>Tested</i>)
Inputs	Datatype	Description
Enable	BOOL	While TRUE the function block is active, all functions can be used, While FALSE output values are frozen.
HWID	HW_IO	Hardware-ID (UINT), given by TIA system configuration for the PROFINET Salve.
UsePKW	BOOL	0 = Use acyclic DPV1 service (default), 1 = Use acyclic PKW channel
PNDP_Statusword	UINT	Statusword 1 of the drive, read by process data input (direct address (%IWx, PEW) or System Function (SFC14)
PKEin	WORD	Input address of identifier PKE (2 bytes)
INDin	WORD	Input address of sub index value IND (2 bytes)
PKW12in	DWORD	Input address of payload PKW1+PKW2 (4 bytes)

SaveParameters	BOOL	Only possible while power stage is not enabled; when TRUE (rising edge) the parameters will be saved permanently in the memory of the drive; Done = TRUE = successfully finished; Possible with PKW and DP-V1
FactoryReset	BOOL	Only possible while power stage is not enabled; While TRUE (rising edge) the device will be configured with factory settings without parameter save; Done = TRUE = successfully finished; Possible with PKW and DP-V1
BlinkDisplay	BOOL	While TRUE (rising edge) the 7-segment display will blink for 10 seconds; Done = TRUE = successfully finished; Possible with PKW and DP-V1
RelnitControl	BOOL	Only possible while power stage is not enabled; When TRUE (riding edge) the control system will be re-initialized; Done = TRUE = successfully finished; Possible with PKW and DP-V1
Outputs	Datatype	Description
PKEout	WORD	Output address of identifier PKE (2 bytes)
INDout	WORD	Output address of sub index value IND (2 bytes)
PKW12out	DWORD	Output address of payload PKW1+PKW2 (4 bytes)
Done	BOOL	While TRUE the function was completed successfully.
Busy	BOOL	When TRUE the FB is busy
Error	BOOL	When TRUE an error occurred
ErrorID	STRING	Short description of the error when Error = TRUE

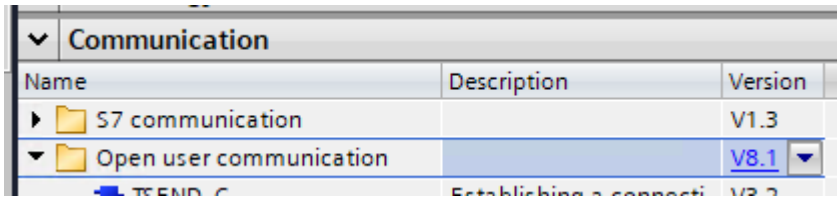
3.12 TCP Read / Write functions

This example project includes TCP Read / Write functions for accessing drive parameters acyclic via drive service interface X3. The same way like DriveManager communication protocol is used here. The following value types are possible:

- (UD)Integer8...32/Float32 values
- String

The Function Blocks can be used for ServoOne and D3 device types. Notice, that each device type can only handle a limited number of parallel opened socket connections. The Function Blocks (new version) are described in the [D3-DA TIA](#)

Example Project. This TCP/IP example communication function blocks need TIA Open User Communication FBs version ≥ V 8.1:



3.13 LowPass1_PT1

<div></div>			<div>Overview:</div> <ul style="list-style-type: none">• Low Pass Filter (first order)• No Execute/Enable input needed• Online calculation• 64 Bit LREAL values• Sample time will be taken from current cycle time <div>Compatibility / Use with:</div> <ul style="list-style-type: none">• Device independent
Inputs	Datatype	Description	
X	LREAL	Raw input value	
f_cut	LREAL	Cut off frequency [Hz], default value = 1 Hz, When value = 0 à Y = X, filter activated when value > 0	
Outputs	Datatype	Description	
Y	LREAL	Filtered output value	

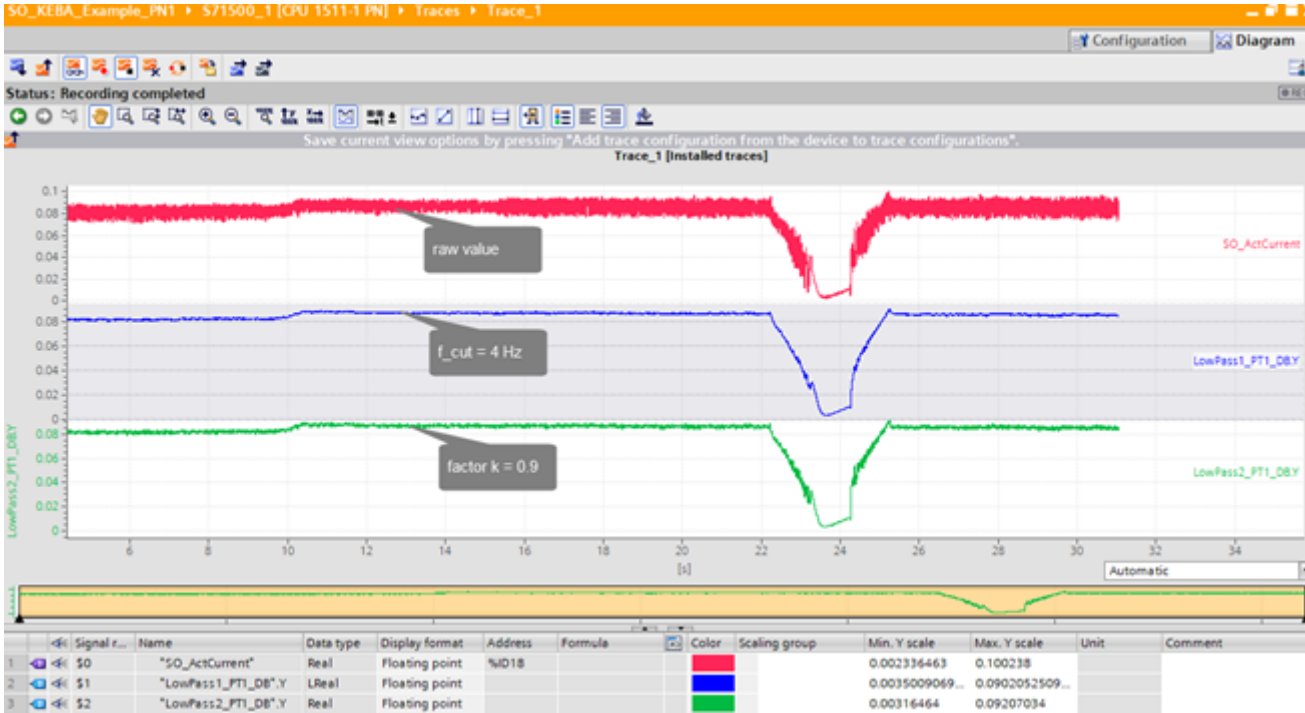


Figure 4: Example filter effect with Filter 1 and 2

3.14 LowPass2_PT1

```
graph LR
    EN[EN] --> FB21
    X["%ID18 'SO_ActCurrent'"] --> FB21
    k["0.5 k"] --> FB21
    FB21 --> Y["Y 0.0"]
```

Overview:

- Low Pass Filter (first order)
- No Execute/Enable input needed
- Online calculation
- 32 Bit REAL values

Compatibility / Use with:

- Device independent

Inputs	Datatype	Description
X	REAL	Raw input value
k	REAL	Filter factor, default value = 0.5 ; 0 > k > 0.9x = active, else Y = X
Outputs	Datatype	Description
Y	REAL	Filtered output value

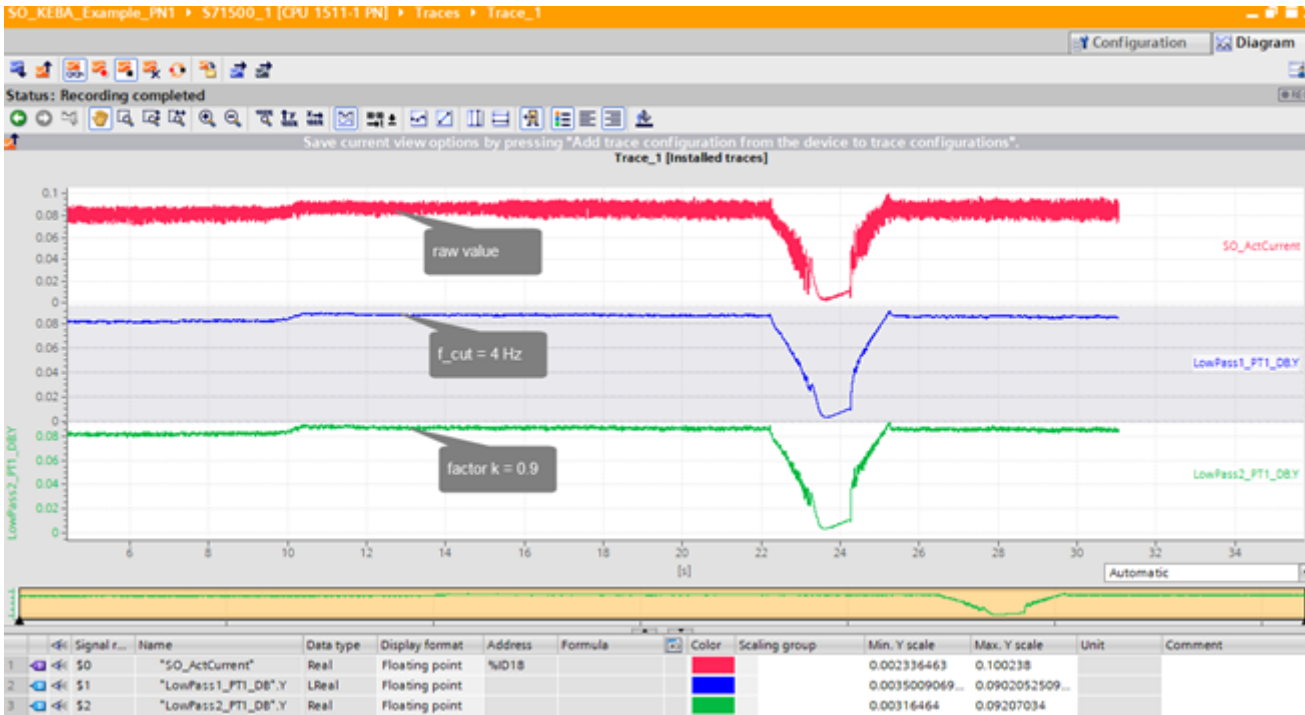


Figure 5: Example filter effect with Filter 1 and 2

4 Known Issues