Legal information

Qualified Personnel

This function must only be handled by qualified personnel whose specialization matches the requirements. A qualified person is someone with relevant experience and the ability to identify hazards and risks associated with the operation. Improper handling by unqualified individuals can lead to operational failures and safety risks.

Disclaimer of Liability

We have thoroughly reviewed the contents of this publication to ensure that it aligns with the described hardware and software. However, as some variations may occur, we cannot guarantee complete consistency. The information in this publication is regularly reviewed, and any necessary updates or corrections will be included in future editions.

Proper use of Halow-Tech products

Products should only be utilized for the specific applications outlined in the accompanying catalog and related technical documents. If incorporating products or components from other manufacturers, they must be either recommended or authorized by the relevant company. To ensure safe and trouble-free operation, proper handling during transport, storage, installation, assembly, commissioning, operation, and maintenance is essential. It is important to adhere to the permissible environmental conditions and to follow the guidelines provided in the associated documentation.

HydroXGuard

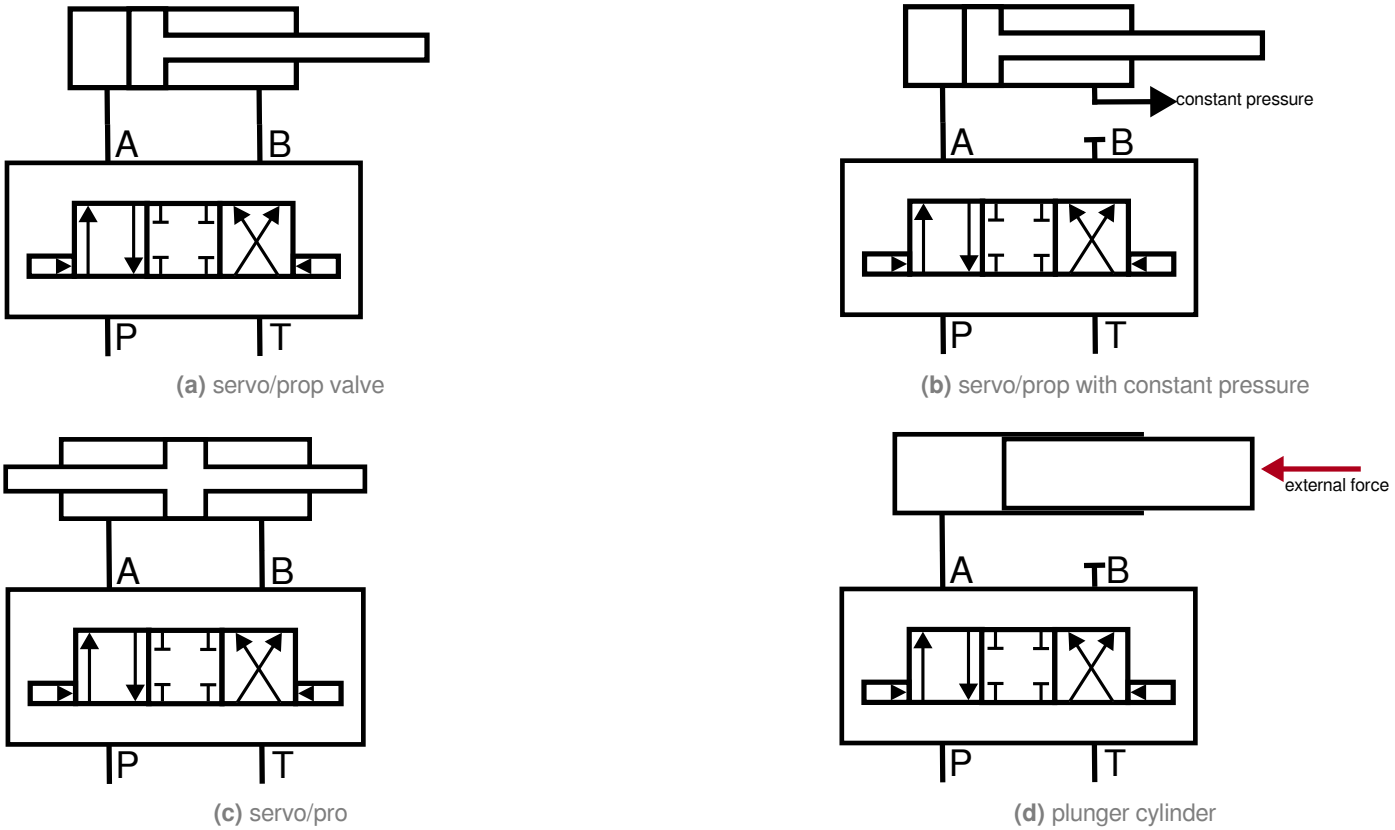
Single Axis Full-featured, with advanced Control Capabilities for Hydraulic Axes

Contents

Description

This function block has been developed to facilitate position control of different hydraulic axes, including:

- Single acting cylinder
- Plunger cylinder
- Double acting cylinder
- Double acting double rod cylinder



(a) servo/prop valve

(b) servo/prop with constant pressure

(c) servo/pro

(d) plunger cylinder

Figure 1: Position control for the hydraulic setup shown

The HydroXGuard PLC Technology function block represents a state-of-the-art solution for advanced hydraulic motion and force control. Depending on its configuration settings, it acts either as an advanced motion controller or an advanced force regulator for a hydraulic axis. A notable feature of this block is its dedicated input for synchronizing multiple axes, either in terms of position or velocity, enhancing the coordination and efficiency of complex hydraulic systems.

At the heart of the HydroXGuard function block is an internal trajectory generator, with trapezoidal acceleration profile. This generator crafts precise position/force trajectories, along with velocity and acceleration paths, adaptable to the dynamics of the controlled system through input parameters. This flexibility allows for the customization of motion profiles to fit specific application needs.

The control algorithm embedded within the block leverages these trajectories in both closed and open loop control configurations. This ensures that the hydraulic axis faithfully follows the intended path with exceptional precision. This integration allows for rapid, precise, and stable adjustments in position and force control, significantly enhancing the performance and reliability of hydraulic systems.

Users have the capability to define detailed parameters, including target position, velocity, acceleration, and jerk for axis motion, as well as target force, force rate, and force acceleration for force control. This level of customization facilitates the tailoring of the system response to meet specific operational requirements, ensuring optimal performance under diverse conditions.

When operating in position control mode, the block intelligently monitors the force exerted by the hydraulic axis. It regulates both the position and speed to ensure that a predetermined maximum force threshold is never exceeded, thanks to an integrated superimposed overload controller. This feature is crucial for preventing damage to the hydraulic system and the load it manipulates, ensuring safe and reliable operation.

The following diagram illustrates how the HydroXGuard function integrates into the overall control system, highlighting its role in achieving precise and reliable position control of the hydraulic axis.

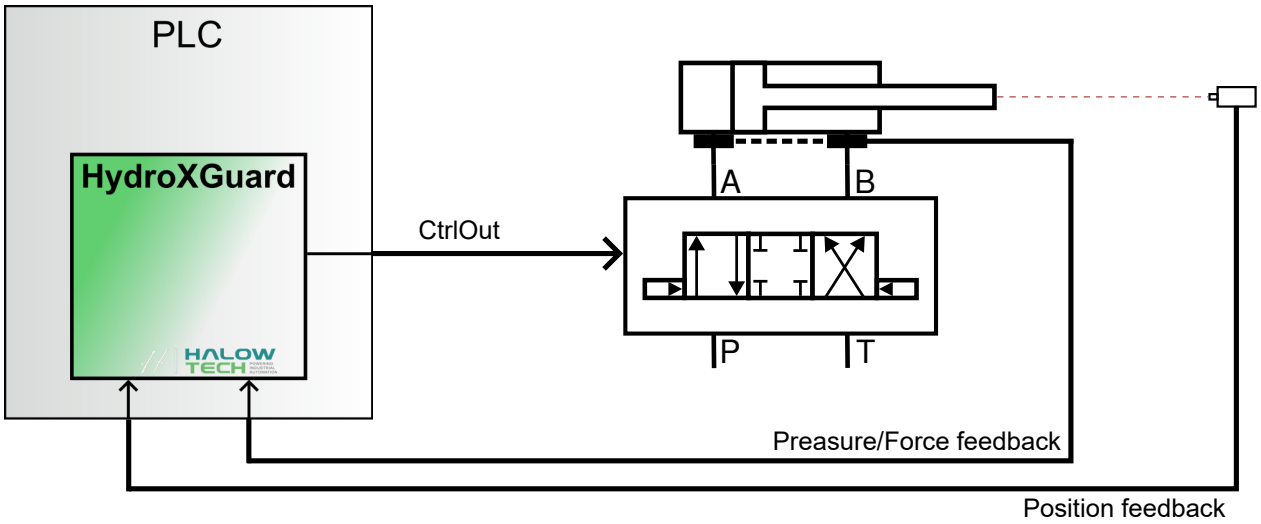


Figure 2: Integration of the function into the overall control system.

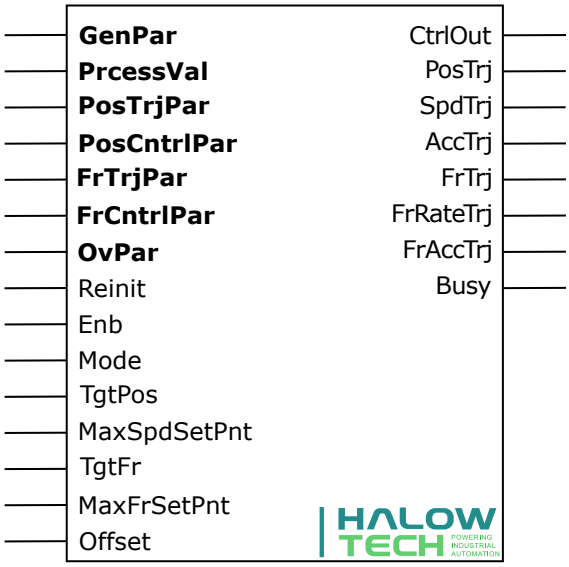
Compatible with various PLC platforms like Siemens S7, Siemens TIA Portal, Rockwell Studio AIO, Rockwell RSLogix 5000, Rexroth IndraWorks, B&R Automation Studio, and Beckhoff TwinCAT, FlexiMotion provides the same high performance on any platform.

Needed PLC Functions[1]

Mainfunction:	HydPosControl	HaTe_HydPosControlV1a
Subfunction:	SubFunktion1	HaTe_SubFunktion1V3c

[1] In addition to the main function, various subfunctions may be required. These need to be imported into the PLC program as well. If you are already using other functions from Halow-Tech that utilize the same subfunction, it does not need to be imported again.

Block diagram



Inputs

GenPar		
	SampleTime	<REAL>
	MaxOut	<REAL>
	MinOut	<REAL>
	SupImpCtrl	<REAL>

GenPar → SampleTime - Calling frequency of the controller, <REAL>

The sample time, in second, of the cyclic interrupt task of the plc at which the function is running. A higher sampling frequency allows for more precise control.

GenPar → MaxOut - Maximum Output, <REAL>

Is used to define the upper limit of the *CtrlOut* to a specific value. We recommend to set this limit at 100. This setting essentially represents the highest level of opening for the servo or proportional valve, expressed as a percentage in one direction

GenPar → MinOut - Minimum Output, <REAL>

Is used to define the lower limit of the *CtrlOut* to a specific value. We recommend to set this limit at -100. This setting essentially represents the highest level of opening for the servo or proportional valve, expressed as a percentage in another direction.
Recommendation: When using a linear servo or proportional valve, it's advised to restrict the control signal to a range between -100 and 100. This range signifies the valve's opening percentage in either direction. Subsequently, these percentage values must be translated to the valve's physical input, which might be represented in current or voltage terms. In figure 3 and 4 are common mappings from output to valve input shown.

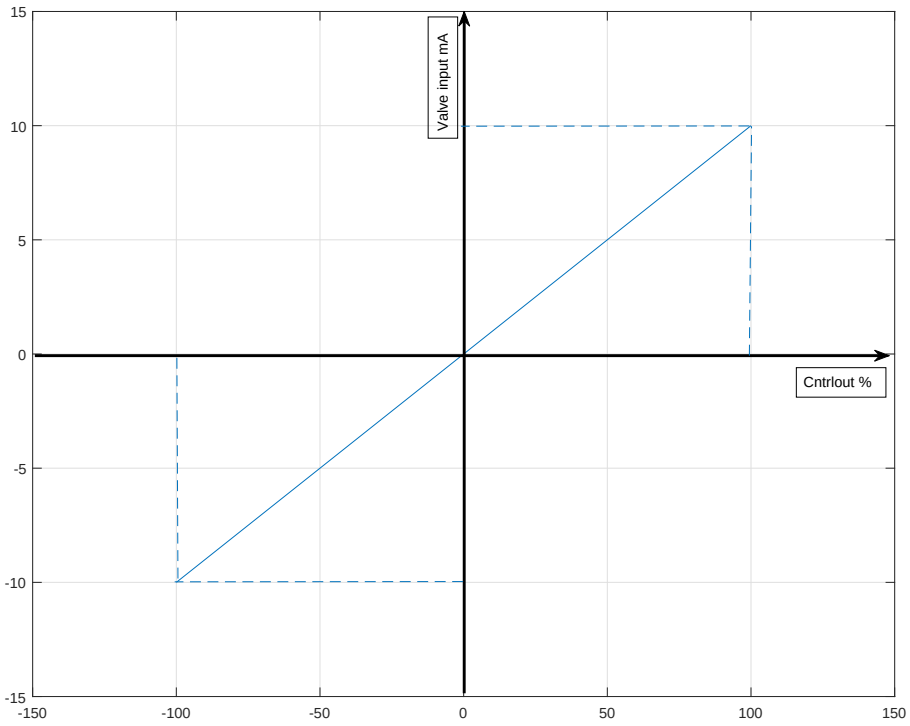


Figure 3: *CtrlOut* against valve input -10mA to 10mA

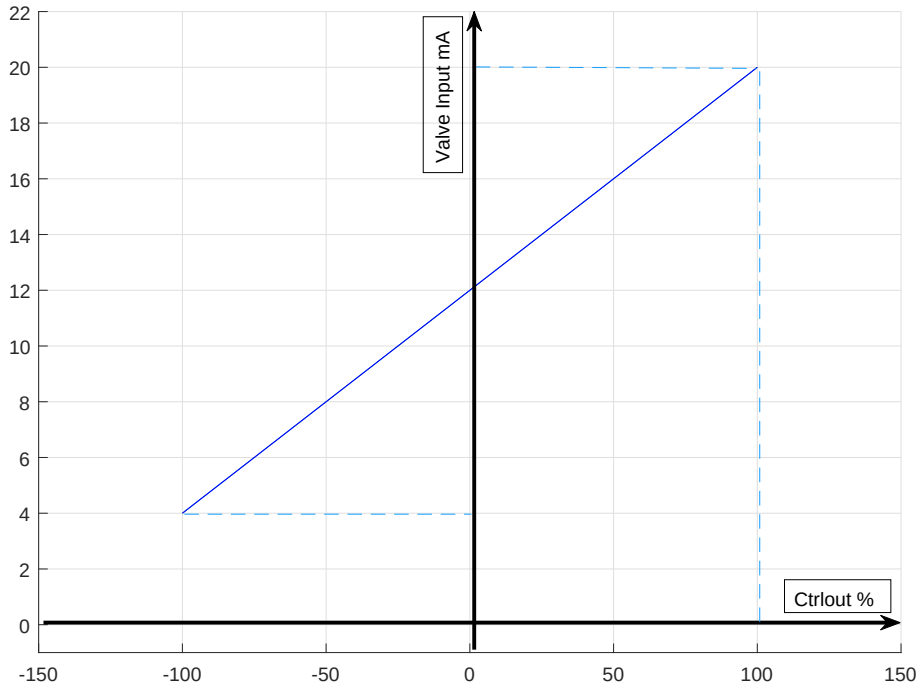


Figure 4: *CtrlOut* against valve input 4mA to 20mA

GenPar → SupImpCtrl - Superimposed Controller input, <REAL>

Every target position value entered through the *TgtPos* input on the function block passes through the internal trajectory generator, and the resulting position trajectory is then fed to the position controller. If you wish to apply a target position value directly to the controller without generating a trajectory or manipulate the position controller with an external superimposed controller, you should use the *SupImpCtrl* input.

First Note: The use of this input should be approached with caution, as it requires a solid understanding of the system's dynamics and control engineering principles.

Second Note: When synchronizing multiple cylinders in position, this input can be used in conjunction with the Position Synchron Controller, which is available as additional Halow-Tech function block. Connecting *SupImpCtrl* to the output of the Synchron controller enables accurate synchronization of the cylinders.

PrcessVal		
	ActPos	<REAL>
	ActFr	<REAL>
	ActPr	<REAL>
	HighPr	<REAL>
	LowPr	<REAL>

PrcessVal → ActPos - Cylinder Actual Position, <REAL>

The input represents the current measured position of the hydraulic cylinder. It is used in the error signal calculation of the internal controller. It should be provided in the same unit as *TgtPos* .

The measured position value provided to *ActPos* must be sampled at a rate at least as fast as the cycle time of the Cycle Interrupt Task in which the function block operates. This requirement is critical for real-time position control.

PrcessVal → ActFr - Cylinder Actual Force, <REAL>

The *ActFr* input is designated for the measured force of the hydraulic axis. It is imperative that the force is measured and inputted in the same unit as the *TgtFr* input. This function block uses the force value for both monitoring and controlling the force exerted by the hydraulic axis.
The measured force value provided to *ActFr* must be sampled at a rate at least as fast as the cycle time of the Cycle Interrupt Task in which the function block operates. This requirement is critical for real-time force control and monitoring.

PrcessVal → ActPr - Actual pressure, <REAL>

Actual Pressure value measured in bar on the piston side of the hydraulic cylinder. It is used to adapt the dynamics of the position controller to the load borne by the hydraulic axis. This ensures that the hydraulic axis exhibits consistent control behavior regardless of the load it carries.
Important: If the hydraulic axis's pressure is not being measured, enter a default value of -1 in the *ActPr* input. This action will effectively deactivate the pressure-dependent dynamic calculation of controller.

PrcessVal → HighPr - High Pressure Line Value, <REAL>

Input the high-pressure line value of the valve in [bar]. This is not a measured value but a fixed constant.

PrcessVal → LowPr - Low Pressure Line Value, <REAL>

Input the Low-pressure line value of the valve in [bar]. This is not a measured value but a fixed constant.

PosTrjPar		
	MaxAccSetPnt	<REAL>
	JerkSetPnt	<REAL>

PosTrjPar → MaxAccSetPnt - Maximum Acceleration Setpoint, <REAL>

This input determines the maximum allowable rate of change in velocity for the position trajectory. It is measured in the same unit as *TgtPos* per second squared. This parameter needs to be customized based on the specific system requirements. *MaxAccSetPnt* plays a critical role in regulating the acceleration of the hydraulic cylinder as it follows the trajectory towards the target position.

PosTrjPar → JerkSetPnt - Maximum Jerk Setpoint, <REAL>

JerkSetPnt controls the abruptness or smoothness of motion by regulating the rate of change of acceleration. It is measured in the same unit as *TgtPos* per second cubed. Adjusting *JerkSetPnt* allows users to customize the motion profile according to their desired level of abruptness or smoothness. For example, if the maximum acceleration needs to be achieved within half a second, the *JerkSetPnt* value should be set to double the maximum acceleration value. See figure 5.

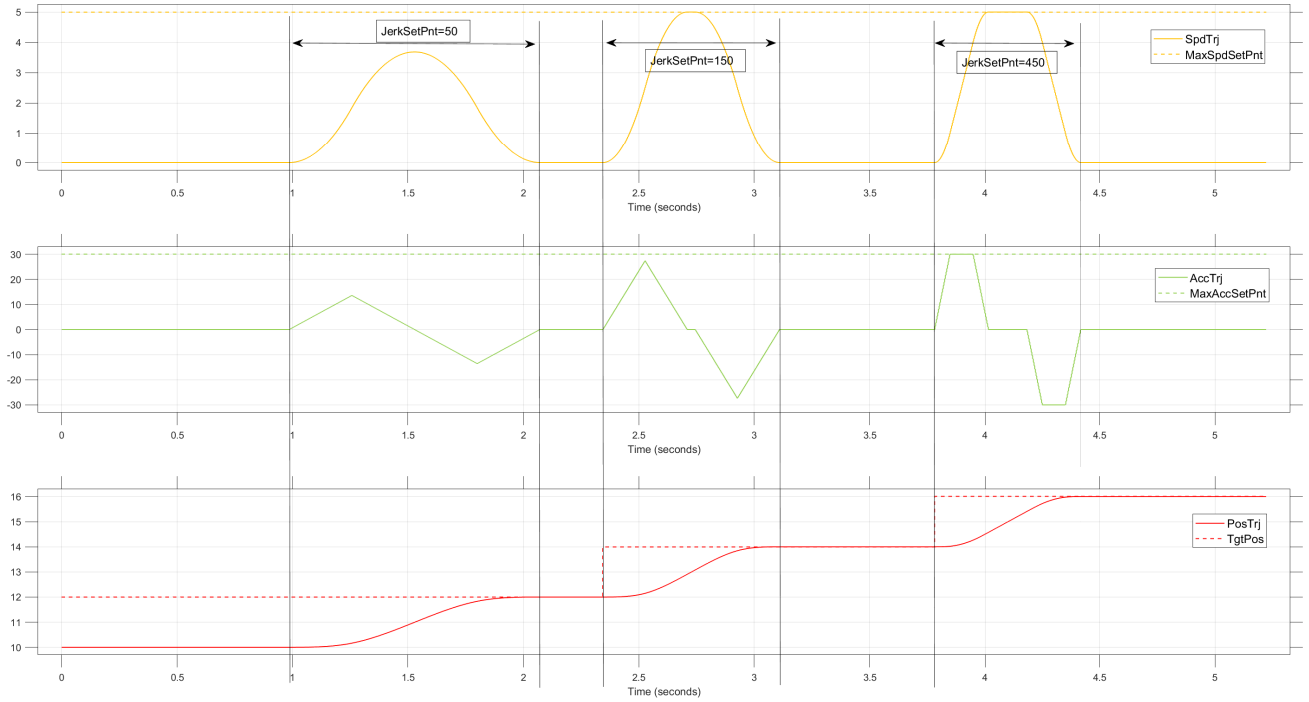


Figure 5: General behavior of the input *JerkSetPnt*

PosCntrlPar		
	KpPos	<REAL>
	MaxKpAmp	<REAL>
	KiPos	<REAL>
	ModeIntCntrl	<BOOL>
	GainFwdSpdCtrl	<REAL>
	GainBwdSpdCtrl	<REAL>

PosCntrlPar → KpPos - Controller P-Factor, <REAL>

It represents the amplification factor of the P-controller within the position controller. A higher *KpPos* value can enhance control accuracy. However, caution is advised: setting the *KpPos* value excessively high may render the closed-loop system unstable. For optimal results and to maintain system stability, it's recommended to initiate with a modest *KpPos* value and incrementally adjust upwards to achieve the desired precision. See example 1 for optimal adjustment.

PosCntrlPar → MaxKpAmp - Maximum Kp Amplification, <REAL>

It relates to the P-controller (Position and Force controller) within the function block, which is capable of dynamically altering the constant *KpPos* and *KpFr* factor. This adjustment ensures that the hydraulic cylinder exhibits consistent control behavior regardless of the load it bears. The dynamic value of the *KpPos* and *KpFr* factors is calculated based on the pressure on the piston side of the cylinder. If the cylinder lacks a pressure sensor, the controller operates with a constant *Kp* value. Through the *MaxKpAmp* input, users can set the upper limit of the amplification for this constant *Kp* value. We recommend setting it between 4 and 5. **Important:** If the hydraulic cylinder's pressure is not being measured, enter a default value of -1 in the *ActPr* input. This action will effectively deactivate the pressure-dependent dynamic calculation of *Kp* .

PosCntrlPar → KiPos - Controller I-Factor, <REAL>

KiPos stands for the amplification factor of the I-controller within the position controller. A *KiPos* value set to zero effectively deactivates the I-controller. While a higher *KiPos* value results in swifter error integration, enhancing overall accuracy, it can also introduce increased oscillations in the closed-loop system. For best performance, it's advisable to commence with a low *KiPos* value and gradually increase it until the desired accuracy level is reached, balancing precision with system stability. See example 1 for optimal adjustment.

PosCntrlPar → ModeIntCntrl - Mode Selection for internal Integral Controller, <BOOL>

This input determines the behavior of the I-Controller. When set to False , the I-Controller remains continuously active. If set to True , the I-Controller operates only in the steady state, deactivating during the hydraulic axis movement. We advise setting this input to true, as allowing the I-Controller to function solely in the steady state often provides superior stability and accuracy for hydraulic axes. This approach typically minimizes, if not eliminates, significant overshoots and, by enabling an increase in the *Ki* factor, greatly enhances accuracy.

PosCntrlPar → GainFwdSpdCtrl - Positive Speed Feedforward, <REAL>

GainFwdSpdCtrl is a system parameter that adds a constant value for positive velocities to the control signal, improving trajectory tracking. It reduces the error signal and facilitates smoother following of the desired trajectory. The optimal positive feedforward value depends on the hardware and system dynamics, which can be determined through measurement and analysis, as illustrated in example 1.

PosCntrlPar → GainBwdSpdCtrl - Negative Speed Feedforward, <REAL>

GainBwdSpdCtrl is a system parameter that adds a constant value for negative velocities to the control signal, improving trajectory tracking. It reduces the error signal and facilitates smoother following of the desired trajectory. The optimal positive feedforward value depends on the hardware and system dynamics, which can be determined through measurement and analysis, as illustrated in example 1.

FrTrjPar		
	FrMaxAccSetPnt	<REAL>
	FrJerkSetPnt	<REAL>

FrTrjPar → FrMaxAccSetPnt - Maximum force acceleration setpoint, <REAL>

This input sets the maximum allowable acceleration of the force, effectively shaping the force trajectory. It's measured in the same units as *TgtFr* , but on a per second squared basis, underscoring its role in adjusting the system's dynamics to match particular requirements. Importantly, the force rate, measured in N/s, represents the first derivative of force, indicating how quickly the force changes over time. Meanwhile, force acceleration, denoted in N/s^2, is the second derivative of force, reflecting the rate of change of the force rate itself. *FrMaxAccSetPnt* is crucial in managing how rapidly the hydraulic cylinder's force accelerates, facilitating a controlled and precise progression towards the target force.

FrTrjPar → FrJerkSetPnt - Maximum force jerk setpoint, <REAL>

This parameter modulates the transition dynamics of force application, dictating the rapidity or gradualness by controlling the rate at which force acceleration changes. Expressed in the same units as *TgtFr* but per cubic second, adjusting the *FrJerkSetPnt* enables users to finely tune the force trajectory to achieve the preferred level of abruptness or smoothness in force adjustments. This flexibility allows for precise control over the force application, ensuring both responsiveness and smooth operation. See figure 6.

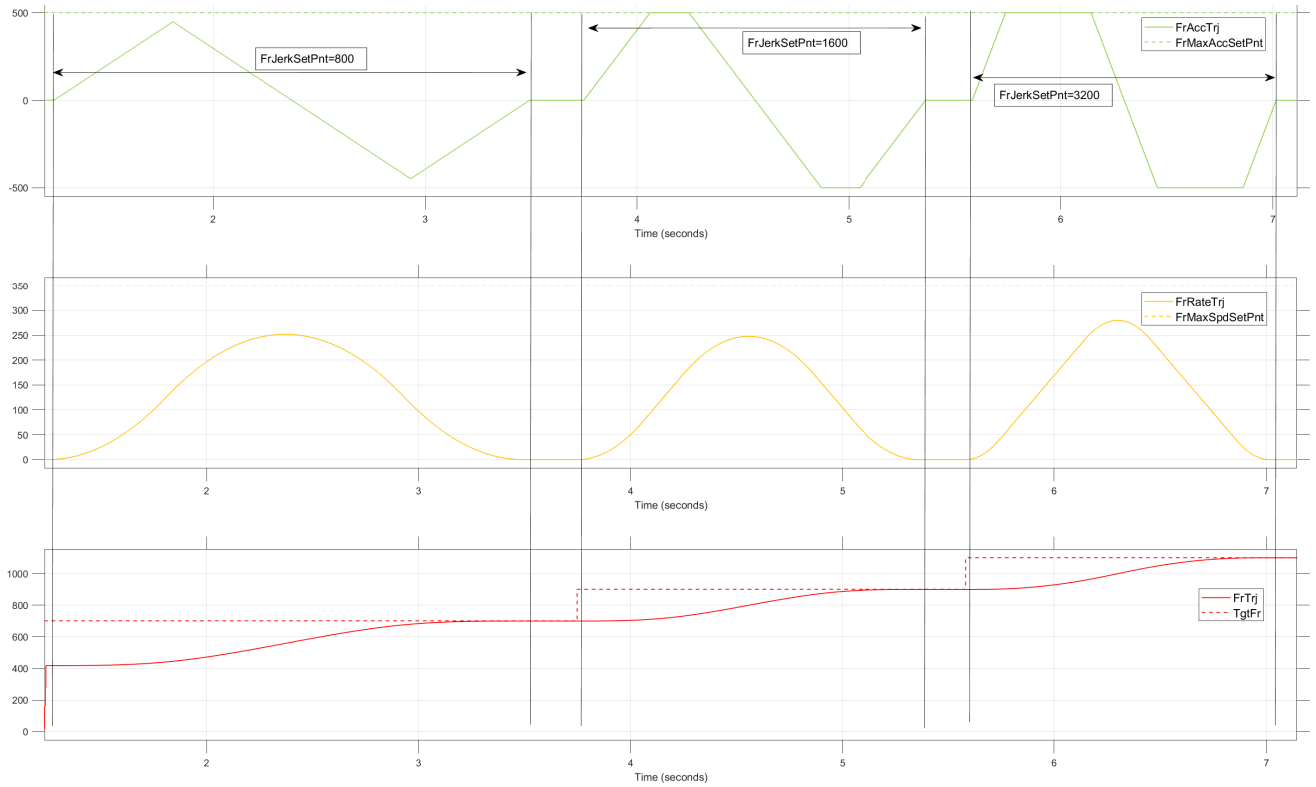


Figure 6: General behavior of the input *FrJerkSetPnt*

FrCntrlPar		
	SwitchPos2Fr	<REAL>
	KpFr	<REAL>
	MaxKpAmp	<REAL>
	KiFr	<REAL>
	ModeIntCntrl	<BOOL>
	FrGainFwdSpdCtrl	<REAL>
	FrGainBwdSpdCtrl	<REAL>

FrCntrlPar → SwitchPos2Fr - Change from position to force controll, <REAL>

Refer to the description of input *Mode* . See figure 7: Between $T = 0.0$ and $T = 1.24s$, *ActFr* $<$ *SwitchPos2Fr* . So the function block is in position control. After $T = 1.24s$, *ActFr* $>$ *SwitchPos2Fr* so the function block switches automatically to force control.

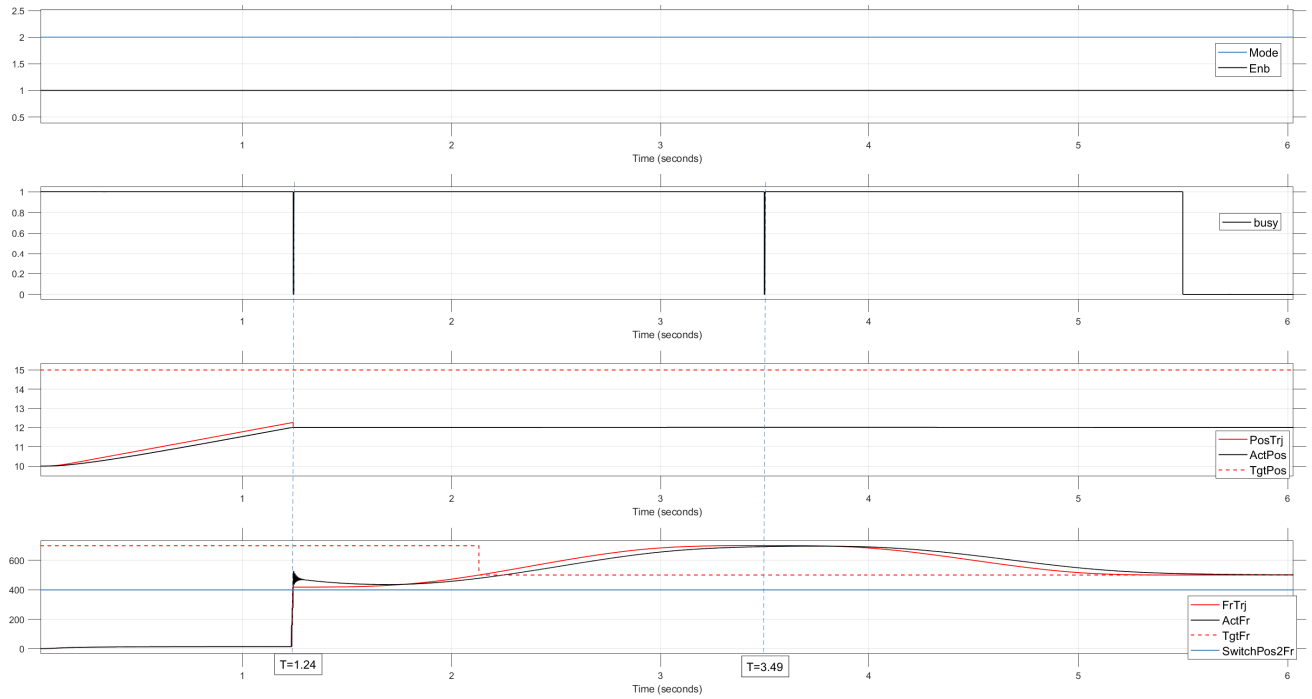


Figure 7: Switch from position to force control mode

FrCntrlPar → KpFr - Force controller P-Factor, <REAL>

This input represents the amplification factor of the P-controller within the force controller. A higher *KpFr* value can enhance control accuracy. However, caution is advised: setting the *KpFr* value excessively high may render the closed-loop system unstable. For optimal results and to maintain system stability, it's recommended to initiate with a modest value and incrementally adjust upwards to achieve the desired precision. See example 2 for optimal adjustment.

FrCntrlPar → MaxKpAmp - Maximum Kp Amplification, <REAL>

It relates to the P-controller (Force controller) within the function block, which is capable of dynamically altering the constant KpFr factor. This adjustment ensures that the hydraulic cylinder exhibits consistent control behavior regardless of the load it bears. The dynamic value of the *KpFr* factors is calculated based on the pressure on the piston side of the cylinder. If the cylinder lacks a pressure sensor, the controller operates with a constant *KpFr* value (in this case set *ActPr* to -1). Through the *MaxKpAmp* input, users can set the upper limit of the amplification for this constant *KpFr* value. We recommend setting it between 4 and 5.

FrCntrlPar → KiFr - Force controller I-Factor, <REAL>

This input represents the amplification factor of the I-controller within the force controller. A higher *KiFr* value can enhance control accuracy. However, caution is advised: setting the *KiFr* value excessively high may render the closed-loop system unstable. For optimal results and to maintain system stability, it's recommended to initiate with a modest value and incrementally adjust upwards to achieve the desired precision. See example 2 for optimal adjustment.

FrCntrlPar → ModeIntCntrl - Mode Selection for internal Integral Controller, <BOOL>

This input determines the behavior of the I-Controller. When set to False , the I-Controller remains continuously active. If set to True , the I-Controller operates only in the steady state, deactivating during the hydraulic axis movement. We advise setting this input to true, as allowing the I-Controller to function solely in the steady state often provides superior stability and accuracy for hydraulic axes. This approach typically minimizes, if not eliminates, significant overshoots and, by enabling an increase in the *KiFr* factor, greatly enhances accuracy.

FrCntrlPar → FrGainFwdSpdCtrl - Force Positive Speed Feedforward, <REAL>

It serves as a feedforward control factor during force control. This factor introduces a static component into the control signal during the build-up of force. It enhances the responsiveness and accuracy of the force control process. See example 2 for optimal adjustment.

FrCntrlPar → FrGainBwdSpdCtrl - Force Positive Speed Feedforward, <REAL>

It serves as a feedforward control factor during force control. This factor introduces a static component into the control signal during the reduction of force. It enhances the responsiveness and accuracy of the force control process. See example 2 for optimal adjustment.

OvPar		
	OvEnb	<BOOL>
	OvPeakFilter	<REAL>
	OvCntrlLim	<REAL>
	OvMaxFr	<REAL>
	OvKi	<REAL>
	OvAntiWindup	<REAL>
	OvKd	<REAL>
	OvKdN	<REAL>

OvPar → OvEnb - Enable force overload protection in position mode, <BOOL>

The OvEnb input enables the internal superimposed force controller. Its important to note that this controller is only effective in position control mode $Mode = 1$. When operating in this mode and the *OvEnb* input is set to True , the actual force *ActFr* is continuously monitored. If the force exceeds the value specified by the *OvMaxFr* input, the superimposed controller calculates a dynamic additional setpoint value for the position controller. This adjustment ensures that the force remains regulated at the *OvMaxFr* value, effectively preventing overload. This feature is crucial for ensuring the system operates safely within its designated parameters, particularly in position control $Mode = 1$.

Explicitly, in the event of an overload, the superimposed controller triggers the cylinder to retract. This mechanism is a safeguard that ensures the hydraulic system remains within safe operating lim its, avoiding damage to the cylinder or associated machinery by retracting the cylinder to alleviate the excessive force.

Should an overload occur while the cylinder is extending, the controller will immediately terminate the movement. The cylinder's position will then be regulated to ensure that the actual force ActFr re mains controlled at the *OvMaxFr* level. Please refer to figure 8 for a visual representation of this process.

In the event of an overload while the cylinder is stationary (holding position), the overload controller will adjust the cylinder's opening such that the actual force ActFr is regulated to remain at the OvMaxFr value. Please refer to figure 9 for a visual representation of this process.

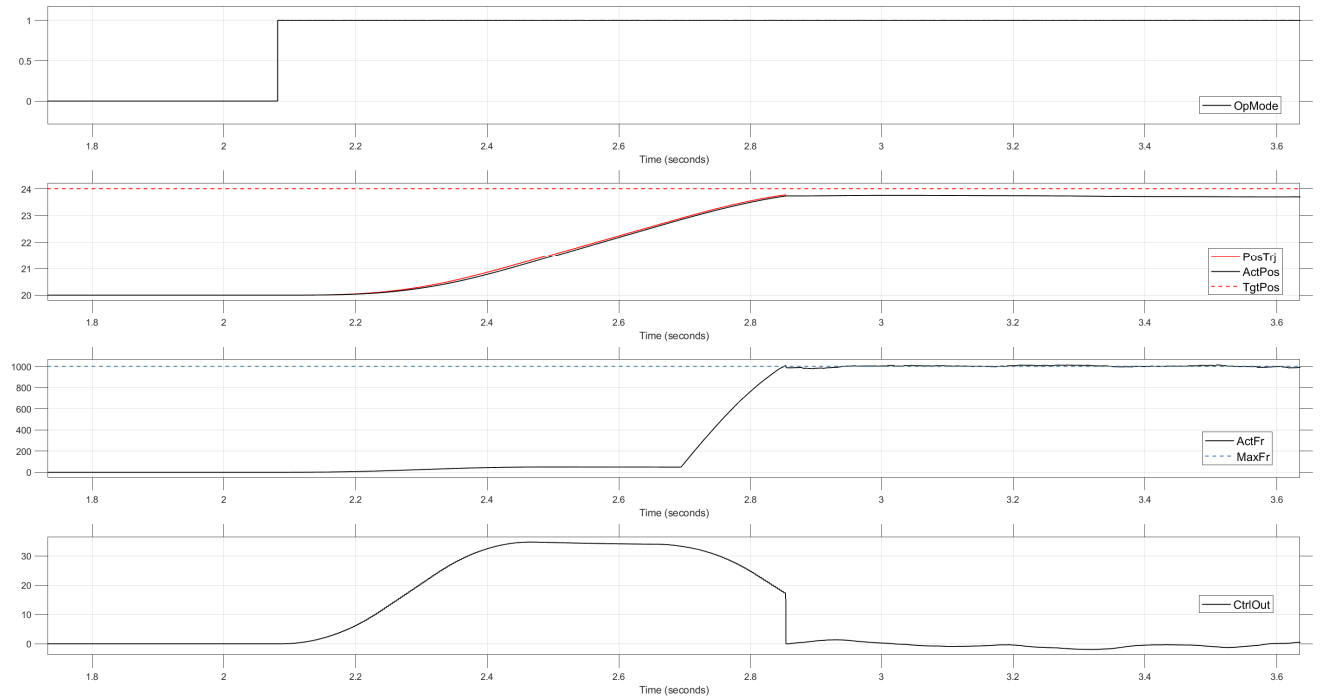


Figure 8: Behaviour of the overload protection. *ActFr* remaining at *OvMaxFr* after the cylinder reaching an obstacle while movement.

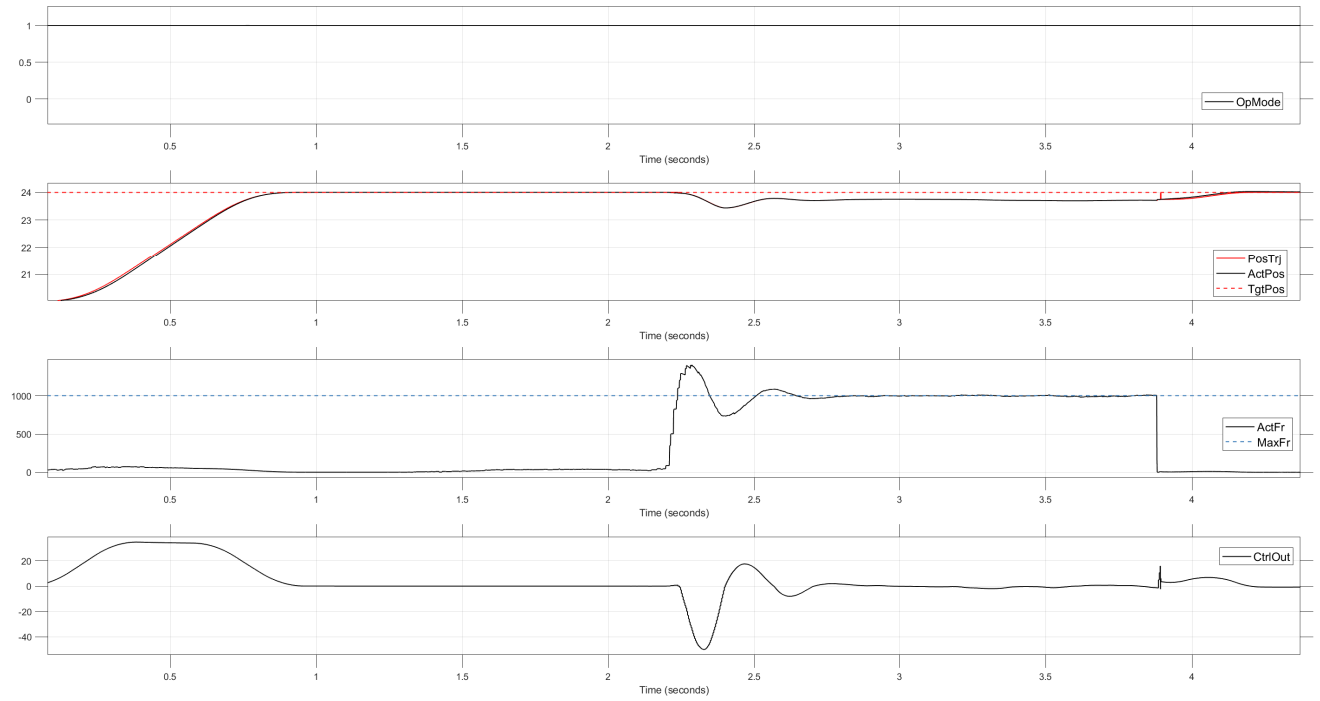


Figure 9: Behaviour of the overload protection. *ActFr* remaining at *OvMaxFr* after an external force appears in steady state.

OvPar → OvPeakFilter - Delay time for filtering peak force impulses, <REAL>

This input serves as a delay timer for the overload controller within the system. When the actual force *ActFr* exceeds the predefined threshold *OvMaxFr* , the overload controller initiates a waiting period of *OvPeakFilter* seconds. If, after this duration, the force remains above *OvMaxFr* , the overload controller is then activated. This feature ensures that transient peaks in force do not trigger unnecessary adjustments, allowing for a brief period to assess whether the overload condition persists before taking corrective action.

OvPar → OvCntrlLim - Maximum deviation of the position, <REAL>

The maximum permissible retraction of the cylinder during overload. It must be in the same unit as *ActPos* .

OvPar → OvMaxFr - Maximum permitted force for overload protection, <REAL>

Refer to the description of *OvEnb* .

OvPar → OvKi - Overload protection controller I-Factor, <REAL>

The *OvKi* input represents the integral gain factor for the overload control. A value of zero effectively deactivates the controller. The higher this value, the more dynamic but also more prone to fluctuations the controller becomes. It is recommended to start with a very small value, in the range of 0.000001, as the initial setting. The optimal value needs to be determined separately for each application and system, ensuring tailored control performance to meet specific operational requirements.

OvPar → OvAntiWindup - Windup protection of the I-Factor, <REAL>

The *OvAntiWindup* input is specifically designed to mitigate the windup effect observed in the integral component of the overload controller. Windup can arise when the integral controller reaches saturation due to the *OvCntrlLim* limits, resulting in an accumulation of integral error. This accumulation can subsequently lead to overshoot and instability once the actuator is capable of responding. By properly adjusting the *OvAntiWindup* value, it's possible to maintain controller responsiveness and stability, even under conditions where the integral controller is saturated. Fine-tuning this parameter enhances the system's ability to respond to overload conditions more effectively, thereby improving the reliability and efficiency of the overload protection mechanism. We suggest an initial value of 2 for the *OvAntiWindup* input.

OvPar → OvKd - Overload protection controller D-Factor, <REAL>

The *OvKd* input specifies the derivative gain factor for the overload controller's D-Factor. This parameter enhances the controller's ability to predict and react to changes in the system's overload conditions by considering the rate of change of the overload error. A higher *OvKd* value means the controller will respond more aggressively to changes in overload conditions, potentially improving system responsiveness and stability by preempting overshoots or compensating for anticipated load changes. However, setting this value too high can lead to increased sensitivity to noise and rapid changes, possibly causing instability. It's crucial to balance the *OvKd* setting to achieve optimal control performance, making adjustments based on the specific dynamics and requirements of the application.

We recommend initially configuring the overload controller using only the integral gain *OvKi* , setting the derivative gain *OvKd* to 0. If the controller does not achieve the desired behavior with *OvKi* alone, then begin to adjust the derivative component *OvKd* . Always start with a small value for to carefully assess its impact on the system's performance and gradually refine the controller's response to meet the desired specifications.

OvPar → OvKdN - Filtercoefficient of the controller D-Component, <REAL>

The input *OvKdN* in the D-component of the Overload Controller represents the derivative filter coefficient. This coefficient is crucial as it determines the effect of the derivative term within the controller by filtering out high-frequency noise that can lead to instability. It effectively shapes the frequency response of the derivative action, smoothing the controller's output to prevent erratic behavior.

Adjusting *OvKdN* allows for the tuning of the derivative action's sensitivity to changes in the system's error. A Small *OvKdN* results in a less sensitive derivative action, which can be beneficial in systems where measurement noise is present, as it reduces the impact of the noise on the control signal. Conversely, a high *OvKdN* makes the derivative action more sensitive to the error change rate, which can improve the controller's responsiveness but may also amplify the noise.

In practice, the value of *OvKdN* should be chosen to balance the trade-off between responsiveness and noise sensitivity, with the aim of enhancing the overall performance of the overload controller without compromising stability. We suggest an initial value of 15.

Enb - Turn controller on/off, <BOOL>

This input controls the activation or deactivation of the function. When the input signal is turned off (False), the function no longer responds to changes in the setpoints and does not generate any trajectory. The monitoring signal *PosTrj* reflects the current position and *FrTrj* reflects the current force, while the other monitoring signals are set to zero. In addition, when the function is deactivated, the *CtrlOut* is set to zero, indicating that there is no valve movement caused by the controller, as shown in table 1.

Mode - Mode select input, <INT>

The function block can operate in either position control or force control mode, depending on the *Mode* input setting. This versatility allows the block to adapt its behavior to meet specific application requirements.
Mode=1: In this mode, the function block operates in position control. There is no monitoring of force values; the system focuses solely on achieving and maintaining the target position.
Mode=2: Initially, the function block operates in position control mode, similar to Mode 1. However, in Mode 2, the current force value at the *ActFr* input is continuously monitored. If the actual force exceeds the threshold specified by the *SwitchPos2Fr* input, the function block automatically switches from position control to force control. This transition ensures that the system can adapt to changing load conditions without manual intervention as shown in table 1.

Important Notes:
In Mode 1, there is no force monitoring. The system does not react to changes in force; it remains dedicated to position control regardless of any force applied.
In Mode 2, the switch from position to force control occurs only once when the actual force surpasses the *SwitchPos2Fr* threshold. It's crucial to understand that if the actual force falls below the *SwitchPos2Fr* threshold after switching to force control, the function block does not revert back to position control. This behavior is designed to maintain stability and prevent constant switching between modes under fluctuating force conditions.

	CtrlOut	*PosTrj*	*SpdTrj*	*AccTrj*	*FrTrj*	*FrRateTrj*	*FrAccTrj*
Enb =True *Mode* =1	calculated based by the position control algorithm	calculated by position trajectory generator	calculated by speed trajectory generator	calculated by acceleration trajectory generator	reflects the current force	0	0
Enb =True *Mode* =2 *ActFr < SwitchPos2Fr*	calculated based by the position control algorithm	calculated by position trajectory generator	calculated by speed trajectory generator	calculated by acceleration trajectory generator	reflects the current force	0	0
Enb =True *Mode* =2 *ActFr > SwitchPos2Fr*	calculated based by the force control algorithm	reflects current position	0	0	calculated by force trajectory generator	calculated by force trajectory generator	calculated by force trajectory generator
Enb =False	0	reflects current position	0	0	reflects the current force	0	0

Table 1: Output signals with the different modes of the controller

TgtPos - Target Position setpoint, <REAL>

The input allows users to specify the desired target position for the hydraulic cylinder. It can be defined in any unit appropriate for the application (e.g., meters, centimeters, millimeters, micrometer). When *TgtPos* recognizes a change, the function generates a trajectory from the current position to the specified target. The internal position controller ensures that the cylinder follows this trajectory accurately.
Important: It's essential to understand that any changes to the *TgtPos* , *MaxSpdSetPnt* , *MaxAccSetPnt* and *JerkSetPnt* during axis movement are internally ignored by the block. The block responds to changes only when *PosTrj* matches *TgtPos* . The *Busy* output serves as an indicator of the block's readiness: When the *Busy* output of the block is logicaly True , it indicates that the cylinder is in motion, and the controller will not respond to new setpoints. Conversely, if it is logically False , the cylinder is in a steady state, allowing the controller to react to new setpoints. See figure 10.

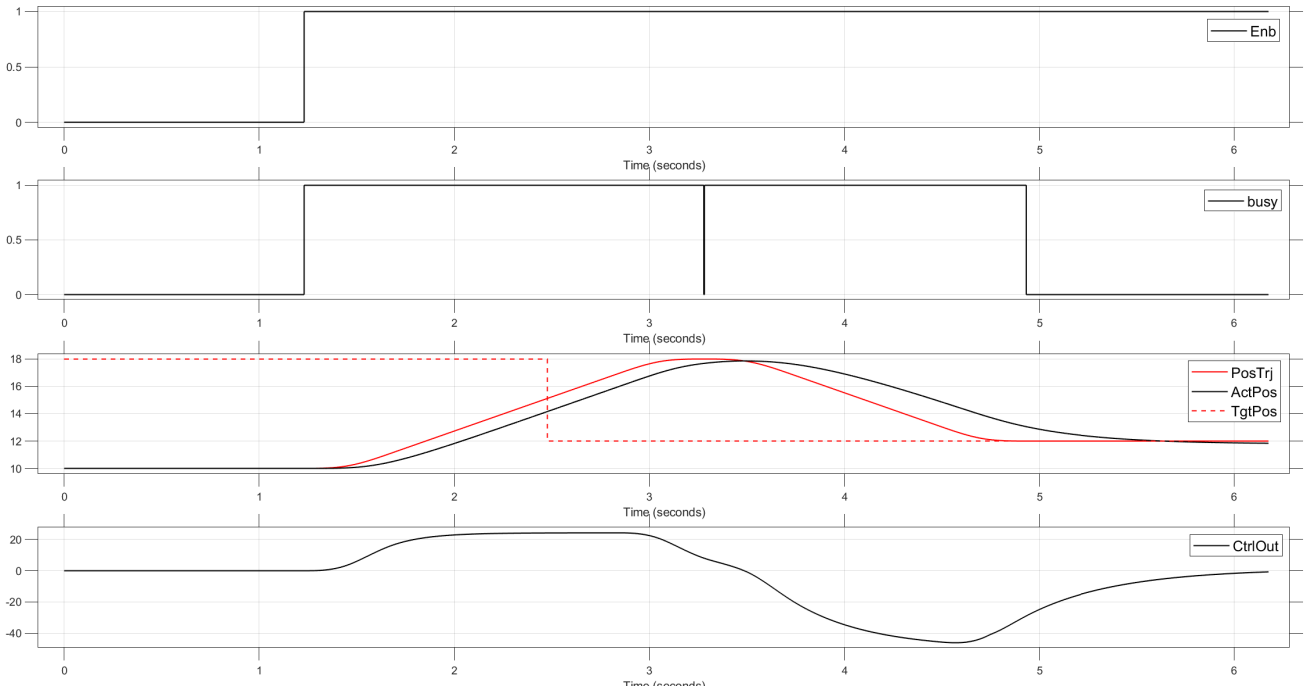


Figure 10: General behavior of the position controller

MaxSpdSetPnt - Maximum speed setpoint, <REAL>

This input sets the maximum speed allowed for the position trajectory generated by the internal generator. It is defined in the same unit as the *TgtPos* input, per second. This parameter ensures that the hydraulic cylinder moves at a controlled speed while reaching the target position defined by *TgtPos* .This sets the top speed for the hydraulic axis during movement. However, factors like distance, acceleration *MaxAccSetPnt* , and jerk *JerkSetPnt* might make it go slower than this speed. See figure 11.

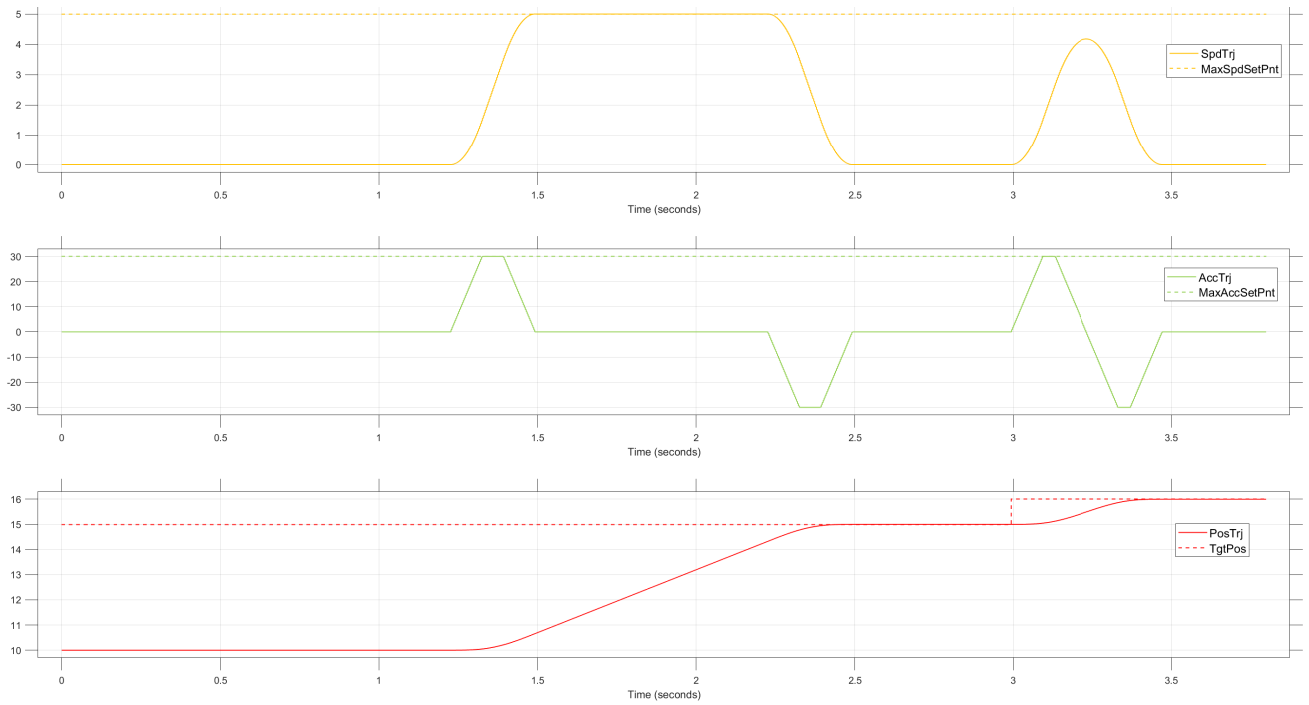


Figure 11: General behavior of the input *MaxSpdSetPnt*

TgtFr - Target Force setpoint, <REAL>

The input allows users to specify the desired target force for the hydraulic cylinder. It can be defined in any unit appropriate for the application (e.g., N, kN, MN). When *TgtFr* recognizes a change and the function is in force control mode, the function generates a trajectory from the current force to the specified target. The internal controller ensures that the current force follows this trajectory accurately. When the function block is deactivated, changes to this input is without function.
Important: It is essential to understand that any changes to the *TgtFr* , *FrMaxSpdSetPnt* , *FrMaxAccSetPnt* and *FrJerkSetPnt* during force adjusment are internally ignored by the block. The block responds to changes only when *FrTrj* matches *TgtFr* . The Busy output serves as an indicator of the block's readiness: When the Busy output of the block is logicaly True , it indicates that the cylinder force is in transition, and the controller will not respond to new setpoints. Conversely, if it is logically False , the cylinder is in a steady state, allowing the controller to react to new setpoints. See figure 12: at $T = 3.23s$, *FrTrj* exactly equals *FrTrj* $= 700N$. Exactly at this time point the *Busy* output set to false and the *FrTrj* goes to the new target *TgtFr* $= 500N$.

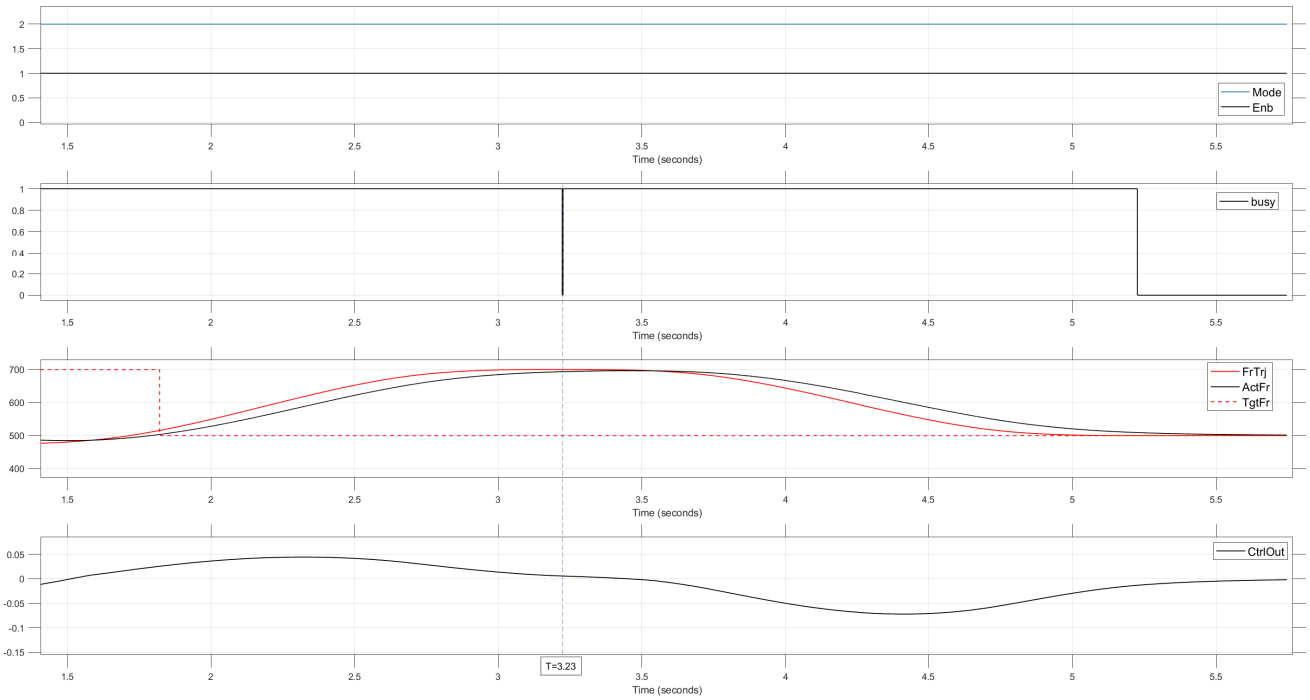


Figure 12: Reaction of the Force controller with a new setpoint while Force transition.

FrMaxSpdSetPnt - Maximum force rate setpoint, <REAL>

This parameter defines the maximum rate at which the force can change, as determined by the internal trajectory generator, and is measured in the same units per second as the *TgtFr* input. It governs the pace at which the hydraulic axis approaches the designated target force, ensuring a smooth and controlled force adjustment. Although this setting establishes the upper limit of the force change rate, the actual rate may be influenced by additional factors such as the total force adjustment required, *FrMaxAccSetPnt* , and *FrJerkSetPnt* , potentially resulting in a slower adjustment than the maximum specified rate.

Offset - Displacement of ControlSignal, <REAL>

This input serves as a means of adjusting the control signal in both the activated and deactivated states of the controller. When the controller is enabled, the value is added to the control signal, allowing for an additional offset. This enables manual control of the cylinder's position in conjunction with the controller's output. In the deactivated state, where the controller output is fixed at zero, the input can be used as a direct means of manual control, independently influencing the cylinder's behavior. It provides a convenient way to switch between automatic and manual operation of the cylinder.

Outputs

CtrlOut - Control Signal, <REAL>

This is the primary output of the block, indicating the valve's opening value. When confined within the range of -100 and $+100$, this value denotes the valve's opening percentage in one or another direction. It is essential to map this percentage to the valve's physical input, which may be quantified in terms of current or voltage. See figure 3 and figure 4.

PosTrj - Monitoring Signal: Position Trajectory, <REAL>

The *PosTrj* output provides information about the desired position trajectory that the hydraulic cylinder should follow. It represents the path from the current position (*ActPos*) to the target position (*TgtPos*). This output is mainly used for monitoring and visualization purposes, allowing users to observe the planned trajectory of the cylinder's movement.
Note: If the *Enb* input is set to false, the *PosTrj* will match the current position *ActPos* of the hydraulic axis. See table 1.

SpdTrj - Monitoring Signal: Speed Trajectory, <REAL>

The *SpdTrj* output represents the velocity trajectory which the hydraulic cylinder should follow. It starts and ends at zero and is limited by the *MaxSpdSetPnt* input. This output is primarily used for monitoring purposes.
Note: If the *Enb* input is set to false, the *SpdTrj* will default to zero. See table 1.

AccTrj - Monitoring Signal: Acceleration Trajectory, <REAL>

The *AccTrj* output represents the velocity trajectory which the hydraulic cylinder should follow. It starts and ends at zero and is limited by the *MaxAccSetPnt* input. This output is primarily used for monitoring purposes.
Note: If the Enb input is set to false, the AccTrj will default to zero. See table 1.

FrTrj - Monitoring Signal: Force Trajectory, <REAL>

It represents the force trajectory, a dynamic profile detailing the planned progression of force over time. This output is critical for visualizing and understanding how the force is intended to evolve throughout the operation, based on the current input parameters and desired end goal. This output is mainly used for monitoring and visualization purposes, allowing users to observe the planned trajectory of the force exerted by the hydraulic axis.
Note: If the *Enb* input is set to false, the *FrTrj* will match the current force *ActFr* of the hydraulic axis. See table 1.

FrRateTrj - Monitoring Signal: Force Rate Trajectory, <REAL>

It delineates the trajectory of the force rate, illustrating the speed at which the force is programmed to change over time within the force controller. It provides a clear representation of how rapidly or slowly the force is expected to evolve, aiding in the optimization of force control strategies for enhanced performance and responsiveness. It starts and ends at zero and is limited by the *FrMaxSpdSetPnt* input. This output is primarily used for monitoring purposes.
Note: If the *Enb* input is set to false, the *FrRateTrj* will default to zero. See table 1.

FrAccTrj - Monitoring Signal: Force Acceleration Trajectory, <REAL>

It indicates the force acceleration trajectory, providing a time-based profile of the expected acceleration rates of force. It starts and ends at zero and is limited by the *MaxAccSetPnt* input. This output is primarily used for monitoring purposes.
Note: If the *Enb* input is set to false, the *FrAccTrj* will default to zero. See table 1.

OpMode	Controller Mode
0	Controller is inactive
1	Position Control
2	Force Control

Table 2: *OpMode* with the different modes of the controller

Busy - Monitoring Signal: Busy signal, <BOOL>

This output reflects the block's responsiveness to new setpoints. If the hydraulic cylinder is moving or under force transition, the *Busy* output becomes True , indicating that the block is not responding to changes in setpoints. The block resumes responsiveness when *PosTrj* aligns with *TgtPos* in position control or *FrTrj* aligns with *TgtFr* in force control, at which point the *Busy* output switches to False , signaling readiness to accept setpoint modifications. See figure 13.

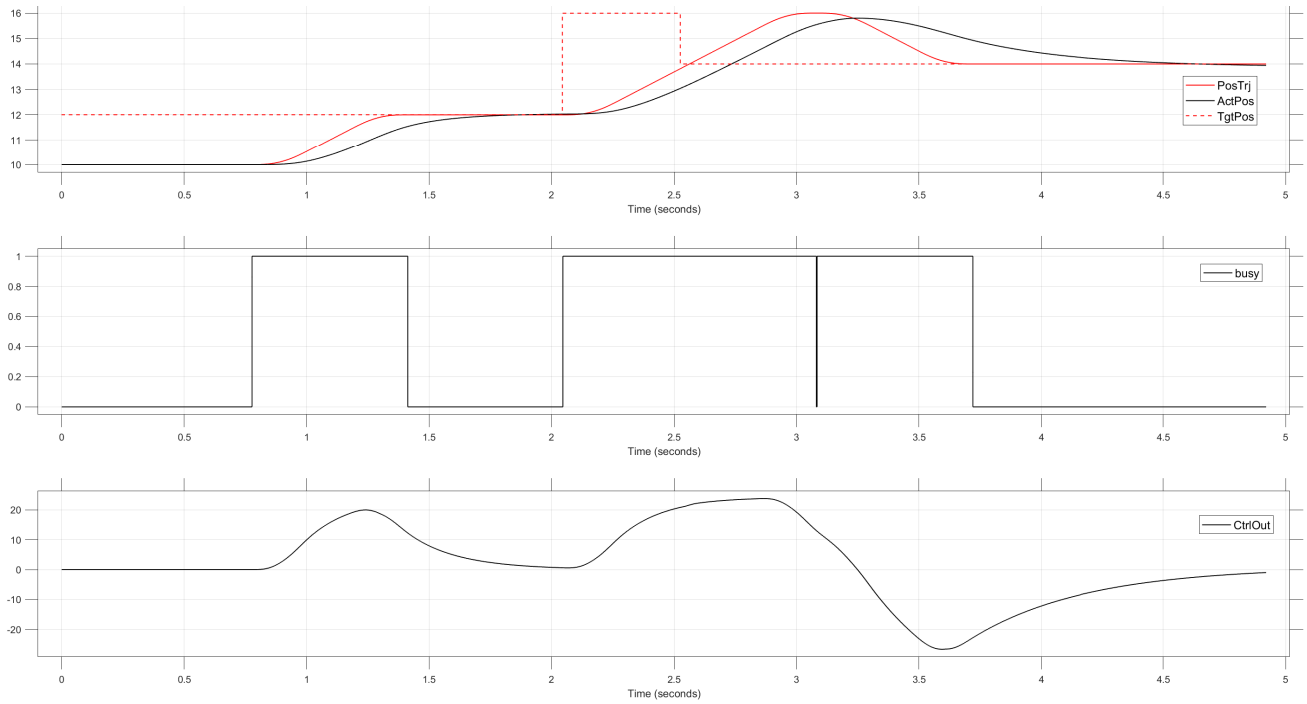


Figure 13: Reaction of output *Busy* while performing an movement

FAQ

How do I stop the hydraulic cylinder during its movement?

To halt the axis while it's moving, simply set the *Enb* Input to logical false. Doing this makes the *CtrlOut* immediately drop to zero, leading to the valve closing and consequently stopping the movement.

Can I input a new position target while the axis is still in motion?

If you wish to assign a new target position (*TgtPos*) during the axis's operation, you must first halt the movement using the *Enb* input. After stopping, wait for a specific delay (a few tens to hundreds of milliseconds depending on the specific axis) to ensure the axis has fully settled. Once stationary, you can then input the new *TgtPos* .

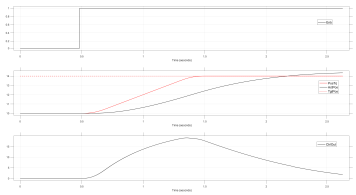
How can I effectively adjust the parameter KpPos , KiPos , GainFwdSpdCtrl and GainBwdSpdCtrl ?

For a detailed description, see example 1.

How can I effectively adjust the parameter KpFr , KiFr , FrGainFwdSpdCtrl and FrGainBwdSpdCtrl ?

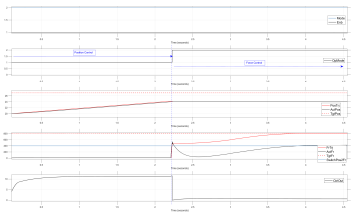
For a detailed description, see example 2.

Examples



Example 1: Setting the position control and feedforward parameters

This example shows how you can adjust the position control and feed-forward parameters experimentaly.



Example 2: Setting the force control and feedforward parameters

This example shows how you can adjust the force control and feed-forward parameters experimentaly.

Important for all examples: These values are provided solely as examples to illustrate the approach for setting control parameters. Under no circumstances should these values be directly applied to your machine, not even as initial starting points. The control parameters must be explicitly adjusted for each machine individually.

Setting the position control and feedforward parameters

Begin by initializing all parameters *KpPos* , *KiPos* , *GainFwdSpdCtrl* and *GainBwdSpdCtrl* to zero. Start the tuning process with the *KpPos* value by setting it to a minimal level. Then, specify a position target using the *TgtPos* and enable the controller. Ideally, adjust *KpPos* so that, during the movement the actual position runs parallel with the position trajectory output (*PosTrj*) at certain moments. If *KpPos* is too low, the actual position might never gets paralleled with the position trajectory.

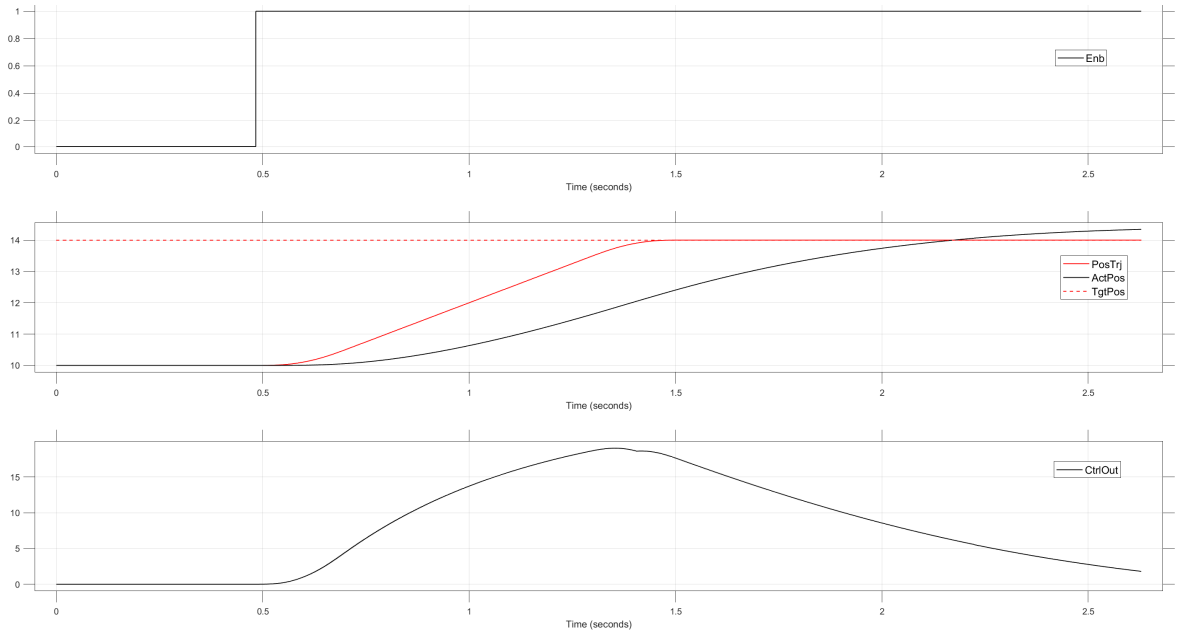


Figure 14: Reaction of the controler with the parameters *KpPos* $= 10$; *KiPos* $= 0$; *GainFwdSpdCtrl* $= 0$ and *GainBwdSpdCtrl* $= 0$.

IIn Figure 14, it is observable that during the movement, the *ActPos* and *PosTrj* are never parallel. This indicates that the *KpPos* value is too small.

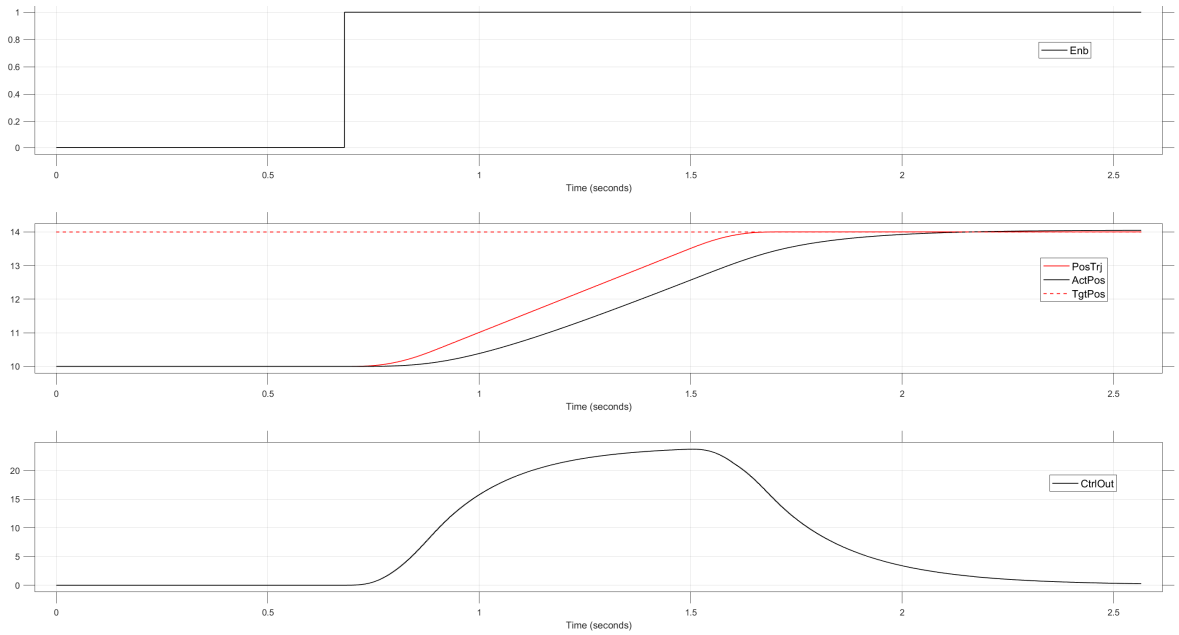


Figure 15: Reaction of the controler with the parameters *KpPos* $= 25$; *KiPos* $= 0$; *GainFwdSpdCtrl* $= 0$; *GainBwdSpdCtrl* $= 0$ and *GainAccCtrl* $= 0$.

AAccording Figure 15 when the *KpPos* increased to 25, the *PosTrj* and *ActPos* now run parallel during movement, indicating that the *KpPos* has been optimally adjusted for the current conditions.
Setting *GainFwdSpdCtrl* and *GainBwdSpdCtrl* : After setting the *KpPos* value, read the value of *CtrlOut* at any point where *ActPos* and *PosTrj* run parallel during movement and divide it by *MaxSpdSetPnt* . Enter the resulting value into the *GainFwdSpdCtrl* input. See figure 16.

$$GainFwdSpdCtrl = \frac{CtrlOut_{t=2.26s}}{MaxSpdSetPnt_{t=2.26s}} = \frac{21.85}{5} = 4.37$$

Repeat the same process when retracting the cylinder, without changing the *KpPos* value. Divide *CtrlOut* by *MaxSpdSetPnt* and enter this result into the *GainBwdSpdCtrl* input. See figure 16.

$$GainBwdSpdCtrl = \frac{CtrlOut_{t=5.04s}}{MaxSpdSetPnt_{t=5.04s}} = \frac{34.15}{5} = 6.83$$

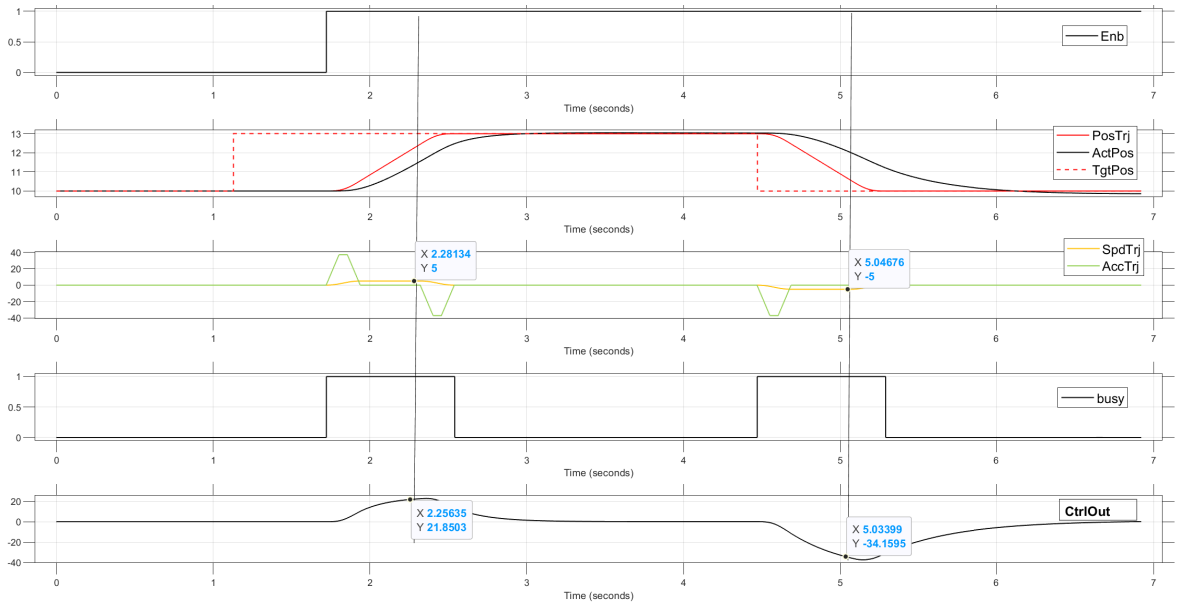


Figure 16: Adjusting the feedforward parameters based on the system's response to a control input.

SSetting *KiPos* : For setting *KiPos* , we suggest setting the *ModeIntCntrl* input to True , ensuring that the I-controller is active only in steady states. Then, observe the persistent control deviation of the position and incrementally increase *KiPos* until you are satisfied with the controller's accuracy.

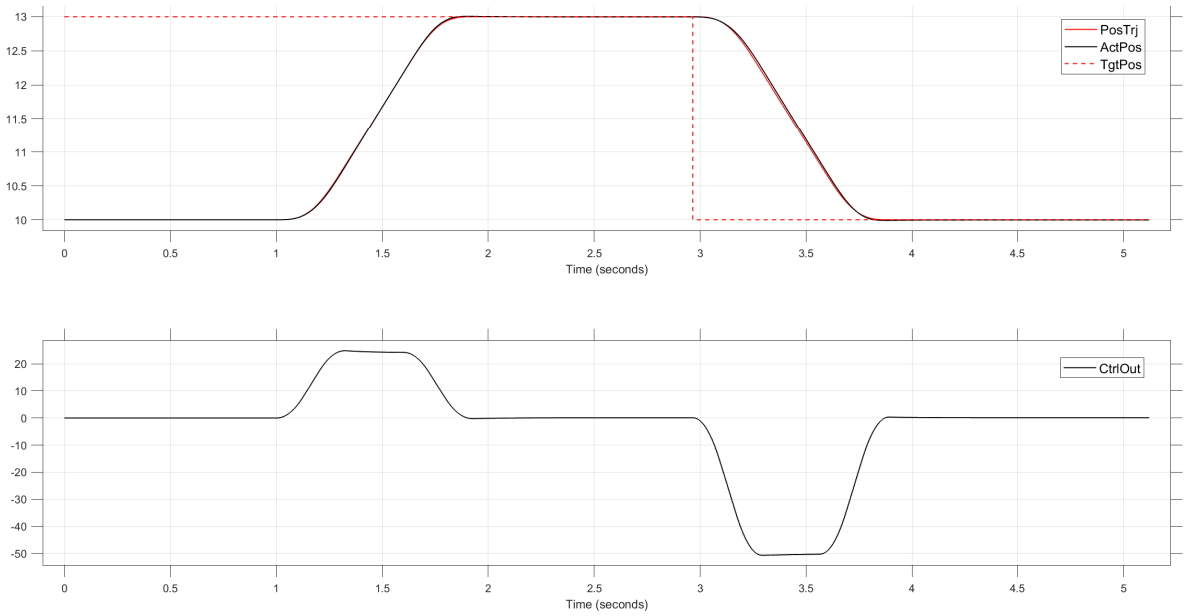


Figure 17: System behavior with well-adjusted parameters.

WWhen all parameters are adjusted as described, the system should be able to closely follow the desired trajectory, as illustrated in Figure 17.

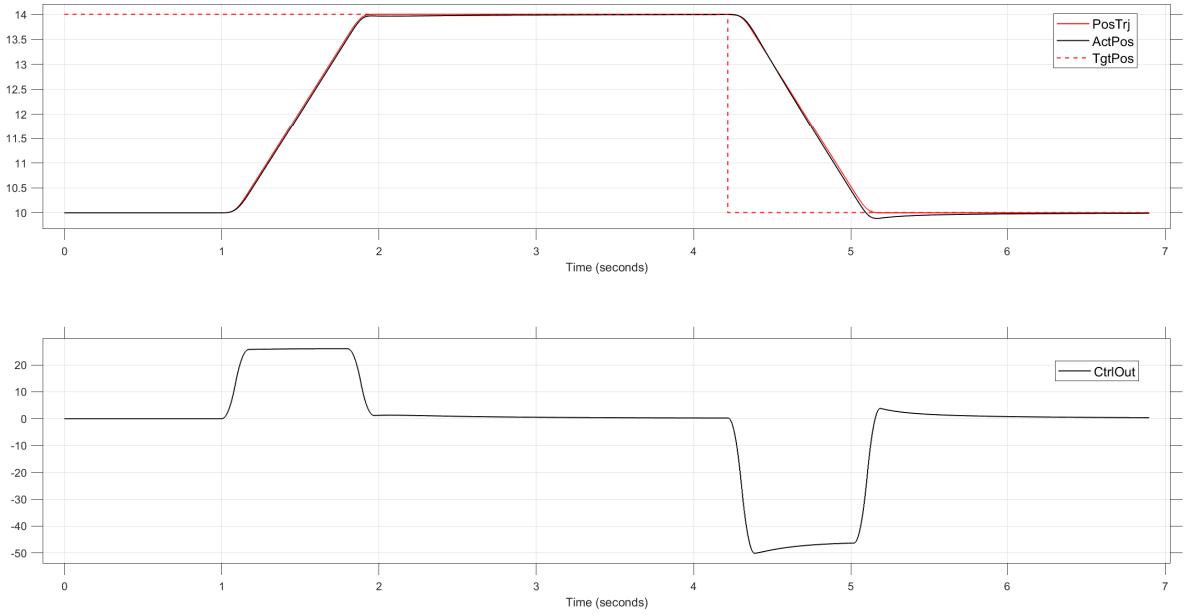


Figure 18: Overshoot despite proper parameter adjustment.

Note: If, despite proper parameter adjustment, overshooting is observed in the control behavior (as shown in figure 18), it is an indication that the dynamics of the Position Trajectory are faster than your machine's maximum capability. Therefore, you should reduce the *MaxAccSetPnt* and *JerkSetPnt* parameters.

Setting the force control and feedforward parameters

Begin by initializing all parameters *KpFr* , *FrGainFwdSpdCtrl* , *FrGainBwdSpdCtrl* and *KiFr* to zero. Start the tuning process with the *KpFr* value by setting it to a minimal level. Then, specify a force target using the *TgtFr* and enable the controller. Ideally, adjust *KpFr* so that, during the force transition the actual force runs parallel with the force trajectory output *FrTrj* at certain moments. If *KpFr* is too low, the actual force might never gets paralleled with the position trajectory. In figure 19 the value of *KpFr* is to low.

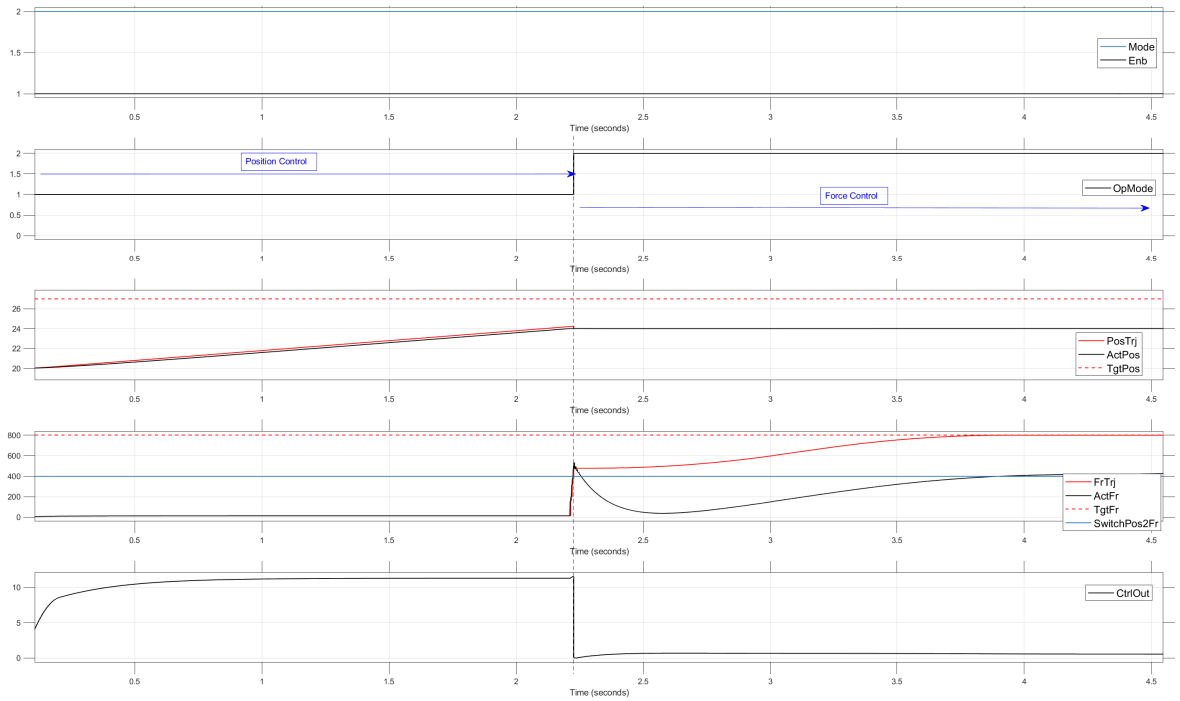


Figure 19: Reaction of the controller with the parameters $KpFr = 0.002$; $KiFr = 0$; $FrGainFwdSpdCtrl = 0$ and $FrGainBwdSpdCtrl = 0$.

AA suitable value for *KpFr* is shown in figure 20.

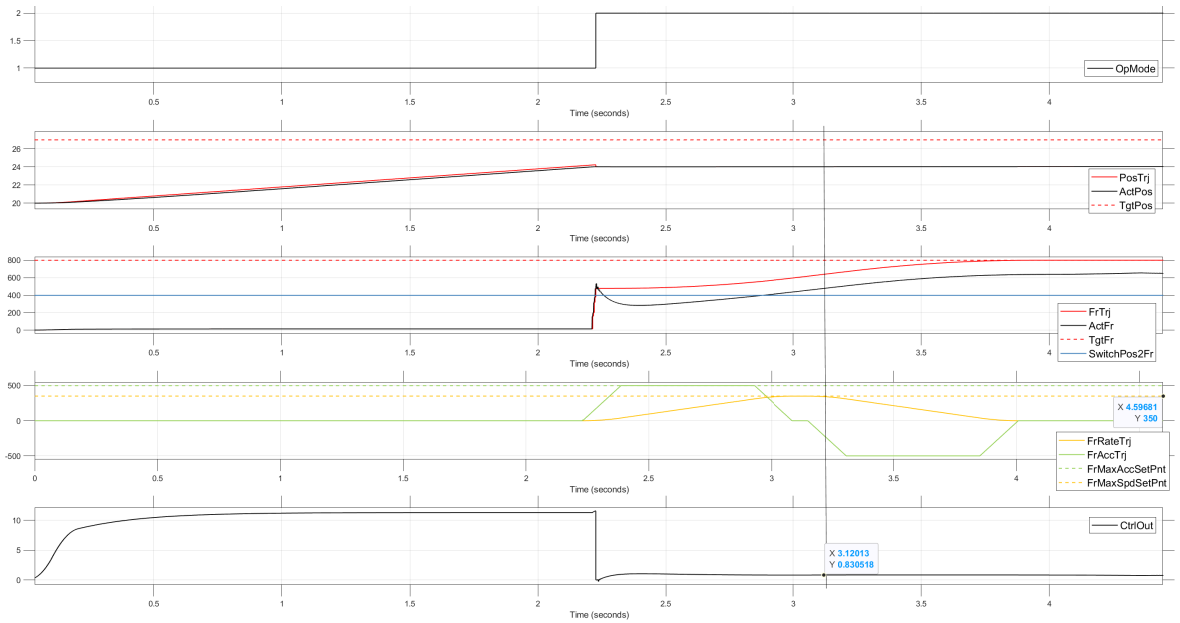


Figure 20: Reaction of the controller with the parameters $KpFr = 0.007$; $KiFr = 0$; $FrGainFwdSpdCtrl = 0$ and $FrGainBwdSpdCtrl = 0$.

SSetting *FrGainFwdSpdCtrl* and *FrGainBwdSpdCtrl* : After setting the *KpFr* value, read the value of *CtrlOut* at any point where *ActFr* and *FrTrj* run parallel during force transition and divide it by *FrMaxSpdSetPnt* . Enter the resulting value into the *GainFwdSpdCtrl* input. Also shown in figure 20.

$$\mathit{FrGainFwdSpdCtrl} = \frac{\mathit{CtrlOut}_{\,t=3.12s}}{\mathit{MaxSpdSetPnt}_{\,t=3.12s}} = \frac{0.83}{350} = 0.0023$$

Repeat the same process when reducing the force, without changing the *KpFr* value. Divide *CtrlOut* by *FrMaxSpdSetPnt* and enter this result into the *FrGainBwdSpdCtrl* input. After setting *FrGainFwdSpdCtrl* and *FrGainBwdSpdCtrl* the force control behavior should be much better, as seen in figure 21.

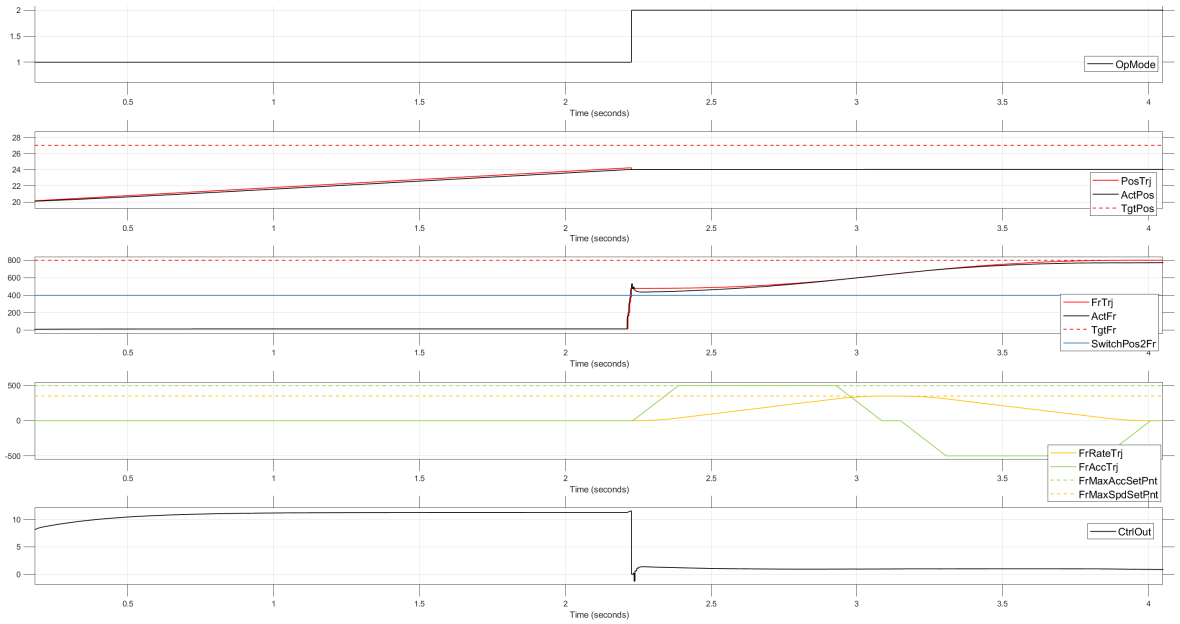


Figure 21: Reaction of the controller with the parameters $\mathit{KpFr} = 0.007$; $\mathit{KiFr} = 0$; $\mathit{FrGainFwdSpdCtrl} = 0.0023$ and $\mathit{FrGainBwdSpdCtrl} = 0.0023$.

SSetting *KiFr* : For setting *KiFr* , we suggest setting the *ModeIntCntrl* input to one, ensuring that the I-controller is active only in steady states. Then, observe the persistent control deviation of the force and incrementally increase *KiFr* until you are satisfied with the controller's accuracy.

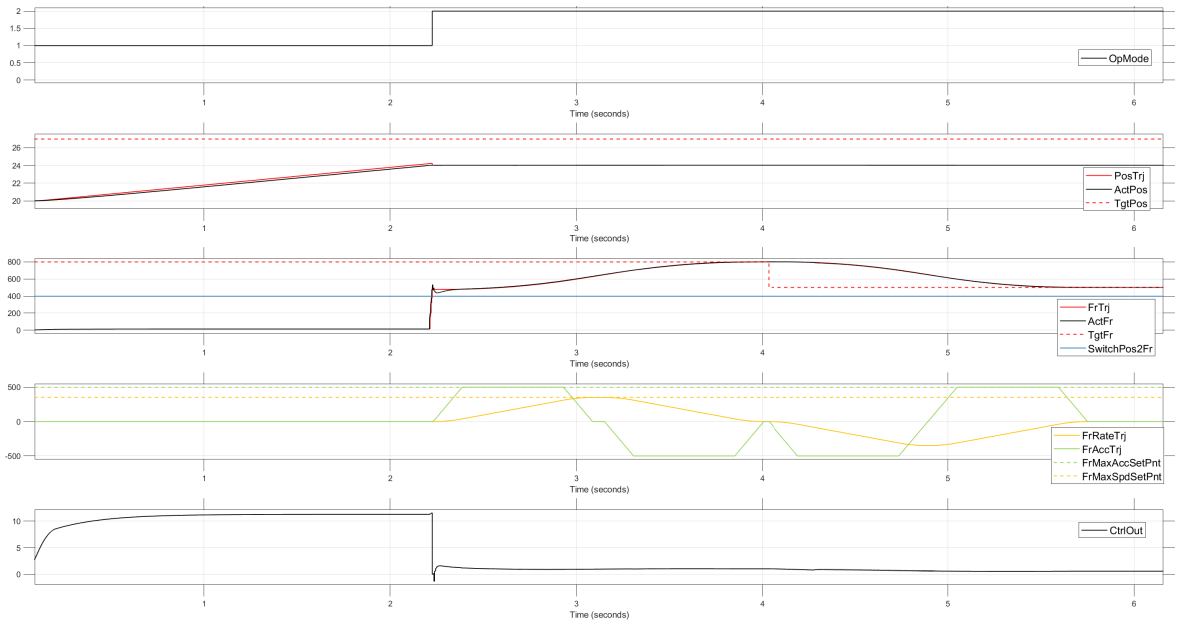


Figure 22: Reaction of the controller with the parameters *KpFr* $= 0.007$; *KiFr* $= 0.5$; *FrGainFwdSpdCtrl* $= 0.0023$ and *FrGainBwdSpdCtrl* $= 0.0023$.

WWhen all parameters are adjusted as described, the system should be able to closely follow the desired force trajectory, as illustrated in figure 22.