rexroth
A Bosch Company

Quick Start Guide

Software module H4U.app Position Force

H4U.app xF - TwinCat 3

RE01939-01-Z Edition 2026-01

Disclaimer

The data specified above only serve to describe the product. As our products are constantly being further developed, no statements concerning a certain condition or suitability for a certain application can be derived from our information. The information given does not release the user from the obligation of own judgment and verification. It must be remembered that our products are subject to a natural process of wear and aging.

Table of contents

1 About this documentation

1.1 About this documentation

This documentation is valid for the following products:

- **H4U**.app Position Force

This Quick Start Guide is valid only in conjunction with the application manual “Software module **H4U**.app Position Force" (R.01939-FK).

Read the commissioning instructions and, in particular, the above document completely before working with the software module.

This document describes the software module for a TwinCat 3 system. This software module is therefore referred to as TC3_H4Uapp_xFAxis<Valve/Pump> in the following description.

2 Description

2.1 Brief description

The software module **H4U**.app xF controls the position (x) and force (F) of a hydraulic axis. It is used to activate valves and pump systems (fixed or variable displacement pump), which control the hydraulic axis.

The software module can be integrated directly in the PLC application of the existing machine control.

The **H4U**.app xF supports various system topologies for valve-controlled and pump-controlled axes. In the case of valve-controlled axes, the software monitors and compensates system pressures for an optimized velocity feedforward control. For pump-based control concepts the app determines limiting values for the control on the basis of the performance data of the pump and the motor. Operating state monitoring ensures that the pump is operated within the admissible operating limits.

To allow fast parameterization pre-defined data sets for Bosch Rexroth pumps and valves are available with technical parameters and characteristic curves. These data sets are directly used as data structure in the PLC application

2.2 Interface description

The figure and the table below show the input and output data of software module *“TC3_H4Uapp_xFAxisValve“*. The interface for software module *“TC3_H4Uapp_xFAxisPump“* merely differs in the structured data types used, but not in the names of the interface.

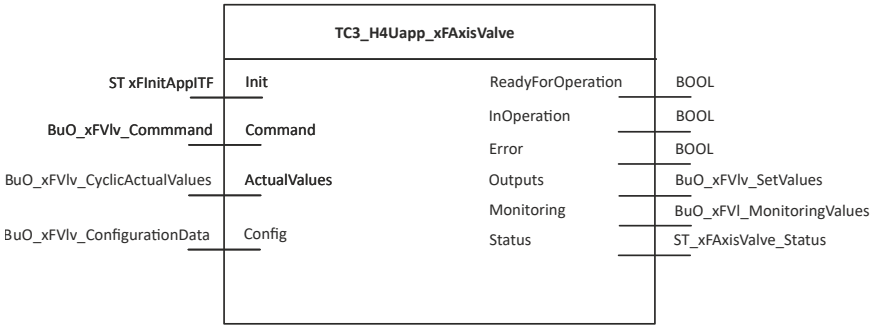

Fig. 1: I/O assignment of the software module.

The structures used and the functionality are described in detail in the application manual “Software module **H4U**.app Position Force" (R.01939-FK).

VAR_INPUT						
Name	**Type**	**Min**	**Max**	**Def**	**Unit**	**Comment**
Init	ST_xFInitAppITF	-	-	-	[-]	Interface for controlling the initialization of the app (for details, see application manual)
Command	BuO_xFVlv_Command	-	-	-	[-]	Interface for commanding the axis and axis control (for details, see application manual)
ActualValues	BuO_xFVlv_CyclicActualValues	-	-	-	[-]	Actual values (for details, see application manual)
Config	BuO_xFVlv_ConfigurationData	-	-	-	[-]	Interface of the configuration data of the axis and axis control (for details, see application manual)
VAR_OUTPUT						
ReadyForOperation	BOOL	False	True	False	[-]	The license and component check has been completed and the software module is ready for operation.
InOperation	BOOL	False	True	False	[-]	The software module executes the controller.
Error	BOOL	False	True	False	[-]	Indicates that a module error has occurred.
SetValues	BuO_xFVlv_SetValues	-	-	-	[-]	Controller output variables (for details, see application manual)
Monitoring	BuO_xFVlv_MonitoringValues	-	-	-	[-]	Internal process data and values of the axis (for details, see application manual).
Status	ST_xFAxisValve_Status	-	-	-	[-]	Status of functions of the software module (for details, see application manual)

2.3 Functional description

The function block *"TC3_H4Uapp_xFAxis<Valve/Pump>"* offers functions for the closed-loop control of position and force control as well as for axis commanding of hydraulic axes. Before the controller functionality and axis commanding can be used, initializing with license and component database check has to be executed. To this end, the path to the license file has to be written at input *LicenseFilePath* and the component data at input *Config.Valve.ValveData* or *Config.Pump.PumpData*, respectively. The license and component data check is started with a rising edge at input *Init.Execute*. Once the check is completed, the output switches to *ReadyForOperation* = TRUE.

After successful initialization the controller functionality can be utilized. The function block is parameterized and commanded using inputs *Config* and *Command*. The required actual and command values are applied to input *ActualValues*.

As long as the axis remains activated *(InOperation)*, license enabling is stored. For a change in licensing, a new license file has to be handed over and initializing has to be restarted with *Init.Execute*. Then, the change is only accepted after a power-off and then another power-on command.

NOTICE	PowerOff may only be initiated when the system is in a safe state.

Reloading of the component data at runtime can be triggered with the input *Config.Valve.ReloadComponentData* or *Config.Pump.ReloadComponentData*, respectively. During reloading of component data, the output *ReadyForOperation* remains TRUE.

In the event of an error, output *Error* changes from FALSE to TRUE and the cause of error can be determined from the structure at output *Status*. Invalid component data will not result in an error (for details, see application manual).

2.4 Integrating the library

To install the **H4U**.app xF in the library manager and then to integrate it in an application project proceed as follows:

1. Download the installation file for library managers of the **H4U**.app xF from the Bosch Rexroth homepage.
2. Activate the license in the Bosch Rexroth licensing portal (for this you require the serial number of the control unit) and download the license file: ➦ https://licensing.boschrexroth.com/.
3. Install the **H4U**.app in the library manager of TwinCat 3: ➦ https://infosys.beckhoff.com/content/1031/tc3_plc_intro/41891384434359666059.html
4. Store the license file in the file system of the control. The path to the license file then has to be handed over to the **H4U**.app.

   ℹ
   - For some systems, a suitable card reader is required for the memory card used for storing the license file.
   - The file name and the license name can be adapted if you wish to use, for example, a common file path for several projects.

5. Integrate the software module in the application project as illustrated in the implementation example. It is recommended that the software module be globally instantiated. Initializing should be carried out in a task with low priority, which can be blocked until initializing is completed. The controller functionality has to be executed in a fast task that meets hard real-time requirements.

6. Configuring the **H4U**.app: To this end, import the pre-configuration from the available XML, adapt it according to the application at hand and pass it to the input *Config*. Details can be found in the application manual “Software module H4U.app Position Force“ (R.01939-FK).

   Alternative: Hand over own component data, for details, see application manual “Software module **H4U**.app Position Force" (R.01939-FK).

7. Component data for Bosch Rexroth valves/pumps are made available in the form of XML files. Select the correct component and add it to the application project by importing it from the PLC OpenXML to easily pass the component data to the input *Config.Valve.ValveData* or *Config.Pump.PumpData*, respectively. (Alternatively, hand over own pump data)

8. Link values of the input structure *ActualValues* according to the system topology. For details, see application manual “Software module **H4U**.app Position Force" (R.01939-FK).

For controller optimization, diagnostics and troubleshooting: Details can be found in the application manual “Software module **H4U**.app Position Force" (R.01939-FK).

2.5 Implementation example

The software module can be integrated as follows:

```
// Definition of global instances

VAR_GLOBAL
    fbH4Uapp_xFAxisValve    : TC3_H4Uapp_xFAxisValve;
    fbH4U_HydDriveSim       : TC3_H4U_HydraulicDriveSim_Valve_Sample;
END_VAR


// ------------ initialisation program ------------
// H4U.app xF Axis Valve and Plant xF Valve initialisation and license check

PROGRAM prg_H4U_init
VAR
    bInitOnce   : BOOL  := TRUE;
END_VAR

// The initialisation and the license check is only to be executed once
IF bInitOnce = TRUE THEN
    // Load default configuration
    fbH4Uapp_xFAxisValve.Config := H4Uapp_xFAxisValve_CustomConfig;

    // Load ValveData
    fbH4Uapp_xFAxisValve.Config.Valve.ValveData := Vlv_4WRPEH6C5B40L_3X;

    // Load path to license file
    fbH4Uapp_xFAxisValve.Init.LicenseFilePath := '!ENTER LICENSE PATH HERE!';

    // Tell the H4U.app.xF Axis Valve to execute initialisation and license check
    fbH4Uapp_xFAxisValve.Init.Execute := TRUE;
```

```
    // Enable simulation model
    fbH4U_HydDriveSim.Enable := TRUE;

    bInitOnce := FALSE;
END_IF

// Once the initialisation is done and the H4U.app.xF Axis Valve is ready for operation
// set execute initialisation to false
IF fbH4Uapp_xFAxisValve.ReadyForOperation = TRUE AND fbH4Uapp_xFAxisValve.Status.Init.Done
= TRUE THEN
    fbH4Uapp_xFAxisValve.Init.Execute := FALSE;
ELSE
    // Run Initialisation Action as long as H4U.app.xF Axis Valve is not ready for operation
    fbH4Uapp_xFAxisValve.aInit();
END_IF


// ------------ motion program ------------
// Controls the motion commands for the set value generation

PROGRAM prg_H4U_motion
VAR
    bStopAxis       : BOOL              := FALSE;
    uiMotionStep    : UINT              := MotionStep.Start;

    // Variables for time measurement
    TIC             : GETCPUCOUNTER;
    TIC_HW          : UDINT;
    TIC_LW          : UDINT;
    TOC             : GETCPUCOUNTER;
    TOC_HW          : UDINT;
    TOC_LW          : UDINT;
    ExecTimeMotion  : LREAL;
END_VAR

// STOP
IF bStopAxis = TRUE THEN
    uiMotionStep  := MotionStep.Stop;
END_IF

// State machine - press cycle
CASE uiMotionStep OF
    MotionStep.Stop:
        // STOP - command
        fbH4Uapp_xFAxisValve.Command.Mode := H4U_CommandMode.Stop;
        IF bStopAxis = FALSE THEN
            uiMotionStep := MotionStep.Start;
        END_IF

    MotionStep.Start:
        // Wait until H4U.app xF Axis Valve is ready for operation
        IF fbH4Uapp_xFAxisValve.ReadyForOperation = TRUE THEN
            uiMotionStep := MotionStep.PowerOn;
        END_IF

    MotionStep.PowerOn:
        // Power on - command
        fbH4Uapp_xFAxisValve.Command.Mode := H4U_CommandMode.PowerOn;
        uiMotionStep := MotionStep.WaitUntilAxisInOp;

    MotionStep.WaitUntilAxisInOp:
        //  Wait until axis is in operation
        IF fbH4Uapp_xFAxisValve.Status.Axis.AxisInOperation = TRUE THEN
            uiMotionStep := MotionStep.MoveToStartPos;
        END_IF
```

```
    MotionStep.MoveToStartPos:
        // Move to start position - set command parameters
        fbH4Uapp_xFAxisValve.Command.Mode                       :=
H4U_CommandMode.MoveAbsolute;
        fbH4Uapp_xFAxisValve.Command.MoveAbsolute.Position     := 200;
        fbH4Uapp_xFAxisValve.Command.MoveAbsolute.Velocity     := 200;
        fbH4Uapp_xFAxisValve.Command.MoveAbsolute.Acceleration := 1000;
        fbH4Uapp_xFAxisValve.Command.MoveAbsolute.Deceleration := 1000;
        fbH4Uapp_xFAxisValve.Command.MoveAbsolute.Jerk         := 10000;

        uiMotionStep := MotionStep.WaitUntilMoveToStartPosDone;

    MotionStep.WaitUntilMoveToStartPosDone:
        // Wait until move to start position is done and change into next MotionStep
        IF fbH4Uapp_xFAxisValve.Status.Axis.CommandInterface.MoveAbsolute.Done = TRUE THEN
            uiMotionStep := MotionStep.MoveOutFast;
        END_IF

    MotionStep.MoveOutFast:
        // Move out fast - set command parameters
        fbH4Uapp_xFAxisValve.Command.Mode := H4U_CommandMode.MoveAbsoluteVelRes;
        fbH4Uapp_xFAxisValve.Command.MoveAbsoluteVelRes.Position         := 700;
        // Set residual velocity for a seamless transition to the next MotionStep
        fbH4Uapp_xFAxisValve.Command.MoveAbsoluteVelRes.VelocityResidual := 100;
        fbH4Uapp_xFAxisValve.Command.MoveAbsoluteVelRes.Velocity         := 200;
        fbH4Uapp_xFAxisValve.Command.MoveAbsoluteVelRes.Acceleration     := 1000;
        fbH4Uapp_xFAxisValve.Command.MoveAbsoluteVelRes.Deceleration     := 1000;
        fbH4Uapp_xFAxisValve.Command.MoveAbsoluteVelRes.Jerk             := 10000;

        uiMotionStep := MotionStep.WaitUntilMoveOutFastDone;

    MotionStep.WaitUntilMoveOutFastDone:
        // Wait until move out fast is done and change into next MotionStep
        IF fbH4Uapp_xFAxisValve.Status.Axis.CommandInterface.MoveAbsoluteVelRes.Done = TRUE
THEN
            uiMotionStep := MotionStep.MoveOutSlowAndPress;
        END_IF

    MotionStep.MoveOutSlowAndPress:
        // Move out slow and press - set command parameters
        fbH4Uapp_xFAxisValve.Command.Mode := H4U_CommandMode.MoveAbsoluteForceLimit;
        fbH4Uapp_xFAxisValve.Command.MoveAbsoluteForceLimit.Position                    :=
900;
        fbH4Uapp_xFAxisValve.Command.MoveAbsoluteForceLimit.Velocity                    :=
100;
        fbH4Uapp_xFAxisValve.Command.MoveAbsoluteForceLimit.Acceleration                :=
1000;
        fbH4Uapp_xFAxisValve.Command.MoveAbsoluteForceLimit.Deceleration                :=
1000;
        fbH4Uapp_xFAxisValve.Command.MoveAbsoluteForceLimit.Jerk                        :=
10000;
        fbH4Uapp_xFAxisValve.Command.MoveAbsoluteForceLimit.ForcePositiveDirection      :=
20;
        fbH4Uapp_xFAxisValve.Command.MoveAbsoluteForceLimit.ForceGradientPosDirection   :=
600;
        fbH4Uapp_xFAxisValve.Command.MoveAbsoluteForceLimit.ModeForceLimitation         :=
1;

        uiMotionStep := MotionStep.WaitUntilMoveOutSlowAndPressDone;

    MotionStep.WaitUntilMoveOutSlowAndPressDone:
        // Wait until move out slow and press is done and change into next MotionStep
        IF fbH4Uapp_xFAxisValve.Status.Axis.CommandInterface.MoveAbsoluteForceLimit.Done =
TRUE THEN
            uiMotionStep := MotionStep.MoveInFast;
        END_IF
```

```
    MotionStep.MoveInFast:
        // Move in fast - set command parameters
        fbH4Uapp_xFAxisValve.Command.Mode := H4U_CommandMode.MoveAbsolute;
        fbH4Uapp_xFAxisValve.Command.MoveAbsolute.Position     := 200.0;
        fbH4Uapp_xFAxisValve.Command.MoveAbsolute.Velocity     := 200.0;
        fbH4Uapp_xFAxisValve.Command.MoveAbsolute.Acceleration := 1000;
        fbH4Uapp_xFAxisValve.Command.MoveAbsolute.Deceleration := 1000;
        fbH4Uapp_xFAxisValve.Command.MoveAbsolute.Jerk         := 10000;

        uiMotionStep := MotionStep.WaitUntilMoveInFastDone;

    MotioNStep.WaitUntilMoveInFastDone:
        //Wait until move in fast is done and change into next MotionStep
        IF fbH4Uapp_xFAxisValve.Status.Axis.CommandInterface.MoveAbsolute.Done = TRUE THEN
            uiMotionStep := MotionStep.MoveOutFast;
        END_IF

ELSE
    // Stop the axis if the MotionStep is invalid
    fbH4Uapp_xFAxisValve.Command.Mode := H4U_CommandMode.Stop;
END_CASE

// Time stamp begin
TIC(cpuCntHiDW => TIC_HW, cpuCntLoDW => TIC_LW);

// Execute motion action
fbH4Uapp_xFAxisValve.aMotion();

// Time stamp end and calculation of the execution time in us
TOC(cpuCntHiDW => TOC_HW, cpuCntLoDW => TOC_LW);
ExecTimeMotion := UDINT_TO_LREAL(TOC_LW - TIC_LW)*0.1;//Rechenzeit in µs


// ------------ control program ------------
// Calls the H4U.app xF Axis Valve and the plant model

PROGRAM prg_H4U_control

VAR
    // Variables for time measurement
    TIC             : GETCPUCOUNTER;
    TIC_HW          : UDINT;
    TIC_LW          : UDINT;
    TOC             : GETCPUCOUNTER;
    TOC_HW          : UDINT;
    TOC_LW          : UDINT;
    ExecTimeControl : LREAL;
END_VAR

// The control action and the simulation are only executed when the H4U.app.xF Axis Valve
// is ready for operation (initialisation and license check are done)
IF fbH4Uapp_xFAxisValve.ReadyForOperation = TRUE THEN
    // Time stamp begin
    TIC(cpuCntHiDW => TIC_HW, cpuCntLoDW => TIC_LW);

    // Execute control action
    fbH4Uapp_xFAxisValve.aControl();

    // Time stamp end and calculation of the execution time in us
    TOC(cpuCntHiDW => TOC_HW, cpuCntLoDW => TOC_LW);
    ExecTimeControl := UDINT_TO_LREAL(TOC_LW - TIC_LW)*0.1;

    // This code should be commented if you connect your real IO
        // Write ValveCmd into simulation input and execute simulation
        fbH4U_HydDriveSim.ValveLiftCmd                          :=
```

```
fbH4Uapp_xFAxisValve.Outputs.ValveCmd;
        fbH4U_HydDriveSim();

        // Write actual simulation values into H4U.app.xF input
        fbH4Uapp_xFAxisValve.ActualValues.Force                 :=
fbH4U_HydDriveSim.ActualValues.Force;
        fbH4Uapp_xFAxisValve.ActualValues.PressureSupply2       :=
fbH4U_HydDriveSim.MonitoringValues.PressureSupply.pS2;
        fbH4Uapp_xFAxisValve.ActualValues.Position              :=
fbH4U_HydDriveSim.ActualValues.Position;
        fbH4Uapp_xFAxisValve.ActualValues.Velocity              :=
fbH4U_HydDriveSim.ActualValues.Velocity;
        fbH4Uapp_xFAxisValve.ActualValues.Acceleration          :=
fbH4U_HydDriveSim.ActualValues.Acceleration;
        fbH4Uapp_xFAxisValve.ActualValues.PressureA             :=
fbH4U_HydDriveSim.ActualValues.PressureA;
        fbH4Uapp_xFAxisValve.ActualValues.PressureB             :=
fbH4U_HydDriveSim.ActualValues.PressureB;
        fbH4Uapp_xFAxisValve.ActualValues.PressureSupply1       :=
fbH4U_HydDriveSim.MonitoringValues.PressureSupply.pS1;
        fbH4Uapp_xFAxisValve.ActualValues.PressureTank          :=
fbH4U_HydDriveSim.MonitoringValues.PressureSupply.pT;
        fbH4Uapp_xFAxisValve.ActualValues.ValveSpoolPosition    :=
fbH4U_HydDriveSim.ActualValues.ValveLift;

    // Example how you can connect your real IO to the H4U.app.xF
        (*
        OutputToRealIO...ValveCmd                               :=
fbH4Uapp_xFAxisValve.Outputs.ValveCmd;

        fbH4Uapp_xFAxisValve.ActualValues.Force                 := InputFromRealIO...Force;
        fbH4Uapp_xFAxisValve.ActualValues.PressureSupply2       := InputFromRealIO...pS2;
        ...
        *)
END_IF
```

3 Service and support

Contact

Bosch Rexroth AG

Service Industriehydraulik

Bürgermeister-Dr.-Nebel-Straße 8

97816 Lohr am Main

Phone +49 (0) 9352 405060

Germany

Email: service@boschrexroth.com

Internet: ➦ www.boschrexroth.com/service

For questions about the product

Bosch Rexroth AG

Phone +49 (0) 9352 403020

Email: my.support@boschrexroth.com

Preparation of information

We can help you in a fast and efficient way if you keep the following information available:

- Details about the product concerned, particularly the type code and serial number.
- Your contact details (phone number, e-mail address)

Index

Bosch Rexroth AG
Zum Eisengießer 1
97816 Lohr a.Main
Germany
Tel. +49 9352 18-0
www.boschrexroth.com

RE01939-01-Z