Valve3P: 3-point actuator control for akYtec PR controllers

User guide, ticket #3011

akYtec GmbH, version 1.0, November 2026

1 Purpose and scope 1
2 Safety requirements 1
3 Interface 2
3.1 Control inputs 2
3.2 Settings inputs 2
3.3 Outputs 3
4 Function 3
4.1 Position estimate 3
4.2 Normal operation 3
4.3 Reversal pause 3
4.4 End positions 4
4.5 Synchronisation 4
4.6 Disable and fault 4
5 Installation and commissioning 4
5.1 Wiring 4
5.2 Using the block in a project 4
5.3 Choosing the settings 4
6 Limitations 5
7 Test cases 5
7.1 Simulation results 5
7.2 Check in the ALP simulator or on hardware 6

1 Purpose and scope

Valve3P turns a position demand of 0 to 100 % into Open and Close pulses for two relay outputs. It drives 230 V AC 3-point actuators of mixing valves, typically from the output of the ALP PID block, which only provides a linear 0 to 100 % signal.

The actuator has no position feedback. Valve3P estimates the position from the length of each pulse and the travel time of the actuator. All actuator-specific values are settings inputs, so one block suits different actuators. Two or more instances can run in one project.

Supported controllers: PR200 (all variants with relay outputs). The block uses only standard ALP functions and runs on PR100 and PR205 as well. The test project was built for the PR200.24.2(4).0.

2 Safety requirements

Read this section before installing the block. Valve3P is a control function, not a safety function. It cannot detect faults in the actuator, the relays or the wiring.

WARNING: requirements for every installation

Actuator with end-position protection. The actuator must switch itself off at both end positions (internal limit switches), or it must be rated as stall-proof (slip clutch or electronic stall protection). The

position estimate drifts over time. The block can therefore command a movement while the valve already sits at its end stop. An actuator without such protection can overheat or be damaged. Do not use it with this block.

Hardware relay interlock. The rule that Open and Close are never on together is implemented in software only. A welded relay contact keeps the motor running regardless of the program, and the program cannot detect it. Wire each direction through a normally-closed contact of the opposite direction (for example with interposing relays or interlocked contactors), so both windings can never receive power at the same time.

Independent overtemperature protection. The position estimate can be wrong. It must never be the only thing that prevents a dangerous flow temperature. For underfloor heating, install a hardwired safety thermostat on the flow line that switches off the circulation pump independently of the PLC, as required by the heating system manufacturer and the applicable standards.

Settings inputs: constants only

Every settings input (travel times, overdrive, deadband, minimum pulse, sample period, reversal pause, start-up sync, re-sync interval) must be fed by a constant block on the ALP canvas.

Never connect a settings input to a network variable, a Modbus register, a variable written by SCADA or any other remotely writable source. Take the values from the actuator datasheet.

SCADA and other remote systems may only read the outputs of the block.

The block adds these protections of its own:

- **Interlock:** both relay outputs are derived from one internal state (stopped, opening, closing). They are never on in the same program cycle. A reversal pause is enforced between opposite movements.
- **Disable:** with Enable = 0 both outputs switch off in the same cycle. The valve holds its position.
- **Input validation:** every settings input is checked in every cycle. A value outside its valid range switches both outputs off and sets Fault. Demand is clamped to 0 to 100 %.
- **Safe defaults:** no overdrive into the end stops, no start-up sync and no periodic re-sync.
- **Relay wear:** deadband, minimum pulse and sample period limit the number of switching operations.

3 Interface

3.1 Control inputs

Pin	Type	Function
xEnable	BOOL	1 = control active. 0 = both relays off at once, the valve holds its position.
rDemand	REAL	Position demand in %, usually the PID output. Clamped to 0 to 100 %.

3.2 Settings inputs

All settings inputs have a default value. A value outside the valid range sets Fault and switches both relays off. Fault clears by itself as soon as all values are valid again.

Pin	Setting	Default	Valid range	Meaning
rTravelOpen	Travel time open	120 s	10 to 3600 s	Running time from fully closed to fully open (datasheet).
rTravelClose	Travel time close	120 s	10 to 3600 s	Running time from fully open to fully closed (datasheet).
rOverdrive	End overdrive	0 s	0 to 600 s	Extra run time added when demand reaches 0 % or 100 % and to sync runs.
rDeadband	Deadband	2 %	0.5 to 20 %	No pulse while |demand - estimate| is inside this band.

Pin	Setting	Default	Valid range	Meaning
rMinPulse	Minimum pulse	0.5 s	0.1 to 5 s	Calculated pulses shorter than this are skipped.
rSamplePeriod	Sample period	10 s	0.5 to 600 s	Interval between two decisions. See section 4.2.
rReversalPause	Reversal pause	2 s	0.2 to 60 s	Minimum pause between the end of one movement and the start of the opposite one.
udStartSync	Start-up sync	0	0, 1, 2	0 = none, 1 = run to closed, 2 = run to open after power-up.
udResyncHours	Periodic re-sync	0 h	0 to 720 h	0 = off. Otherwise a sync run to the nearer end every n hours.

3.3 Outputs

Pin	Type	Function
xOpen	BOOL	Relay OPEN.
xClose	BOOL	Relay CLOSE.
rPosition	REAL	Estimated valve position in %.
xSynchronized	BOOL	1 after a run of at least the full travel time has reached an end position. The estimate is then referenced to a real end position.
xMoving	BOOL	1 while a pulse is active.
xFault	BOOL	1 while a settings input is outside its valid range. Both relays are off.

4 Function

4.1 Position estimate

The block has no feedback from the valve. During every pulse it adds or subtracts the travelled distance:

```
change in % = pulse time / travel time of that direction x 100
```

The estimate is limited to 0 to 100 %. After power-up it starts at 0 %. It is not stored through a power failure. Small errors from relay timing, motor start-up and differences between datasheet and real travel time add up over time. The estimate is corrected only when the valve is driven to an end position (sections 4.4 and 4.5).

4.2 Normal operation

Once per sample period, and only while no pulse is running, the block compares the demand with the estimate:

1. error = demand - position estimate
2. If |error| is inside the deadband, nothing happens.
3. Otherwise pulse time = |error| / 100 x travel time of the required direction.
4. If the pulse time is shorter than the minimum pulse, it is skipped.
5. Otherwise the relay for that direction switches on for the pulse time. The next decision follows one sample period after the pulse has ended.

Demand changes during a running pulse are taken into account at the next decision. The sample period limits how often the relays switch. A short period follows the PID closely but switches more often. A long period protects the relays but adds a delay to the control loop. For a heating mixing circuit with a 120 s actuator, 5 to 15 s is a typical value.

4.3 Reversal pause

After a movement ends, a movement in the opposite direction waits until the reversal pause has passed. The request is kept and starts as soon as the pause is over. A movement in the same direction does not wait for the pause.

4.4 End positions

When the demand reaches 100 % (or 0 %), the block runs the relay for the remaining travel according to the estimate plus the end overdrive. Afterwards the estimate is set to exactly 100 % (or 0 %), and no further pulses follow while the demand stays at the limit. If the remaining time is shorter than the minimum pulse, the estimate is set to the end position without a pulse.

With the default overdrive of 0 s the valve only reaches its end if the estimate is correct. An overdrive of 5 to 10 % of the travel time makes sure the end is reached even after drift. Use it only with actuators that have end-position protection (section 2).

4.5 Synchronisation

- **Start-up sync** (udStartSync = 1 or 2): after power-up, as soon as Enable = 1, the actuator runs for the full travel time plus overdrive to the selected end. The estimate is then 0 % or 100 %, and Synchronized = 1. This briefly closes or opens the valve fully.
- **Periodic re-sync** (udResyncHours > 0): every n hours the actuator runs for the full travel time plus overdrive to the end nearer to the current estimate. This disturbs the process for up to one travel time plus overdrive.
- **Synchronized** also becomes 1 when a normal end run lasted at least the full travel time.

Both sync modes deliberately run the actuator against its end stop. They are off by default and require an actuator with end-position protection.

4.6 Disable and fault

Enable = 0 or Fault = 1 switches both relays off in the same program cycle. A running pulse is aborted, and the estimate keeps the distance travelled so far. When Enable returns to 1 and Fault is cleared, control continues from the current estimate. A pending start-up sync waits until Enable = 1.

5 Installation and commissioning

5.1 Wiring

- Connect the OPEN winding of the actuator to the relay wired to xOpen, and the CLOSE winding to the relay wired to xClose. In the test project these are Q1 and Q2.
- On the PR200, Q1 and Q2 share one common terminal. This suits a 3-point actuator, because both windings use the same 230 V phase.
- Add the hardware interlock described in section 2.
- Check the direction: with xOpen = 1 the valve must open towards the hot side. Swap the two windings if it does not.

5.2 Using the block in a project

1. Place one block (or macro instance) per actuator.
2. Connect rDemand to the PID output and xEnable to the release condition of the heating circuit.
3. Connect every settings input to a constant block with the value from the actuator datasheet. Leave a setting at its default only if the default suits the actuator.
4. Connect xOpen and xClose to the relay outputs.
5. Optionally publish rPosition, xSynchronized, xMoving and xFault to SCADA (read only). Use xFault for an alarm.
6. For a second actuator, repeat with a second instance and its own relays and constants.

5.3 Choosing the settings

Setting	Recommendation
Travel times	From the datasheet. If open and close differ, enter both. Check with a stopwatch on site if possible.
Deadband	1 to 3 %. Smaller values give more small pulses.
Minimum pulse	The shortest pulse the actuator follows reliably, usually 0.3 to 1 s.

Setting	Recommendation
Sample period	5 to 15 s for heating circuits. Should be clearly shorter than the response time of the loop.
Reversal pause	From the datasheet, otherwise 1 to 3 s.
End overdrive	0 s unless the actuator has end-position protection. Then 5 to 10 % of the travel time.
Start-up sync	1 (closed) is the safe choice for heating, if the actuator has end-position protection. Otherwise 0.
Periodic re-sync	Off, unless the circuit rarely reaches 0 % or 100 % and the actuator has end-position protection. Then 24 to 168 h.

6 Limitations

- Position is estimated, not measured. Accuracy depends on the travel time setting and on how often the valve reaches an end position.
- The estimate is not retained through a power failure. After power-up it starts at 0 % or runs the start-up sync.
- A running pulse is completed even if the demand changes. Only Enable = 0 or a fault aborts it.
- The block cannot detect a jammed valve, a broken wire or a welded relay contact.
- The block is not a safety function and cannot replace the protections listed in section 2.

7 Test cases

The logic was tested in a cycle-by-cycle simulation of the ST code (10 ms cycle, 32-bit REAL and UDINT arithmetic as on the controller). Default settings unless stated. Result: 21 of 21 checks passed. A final check in the ALP simulator or on hardware is still required (section 7.2).

7.1 Simulation results

Acceptance test	Procedure	Result
Step changes	Demand 0 to 50 %, then 50 to 20 %.	One OPEN pulse of 60.0 s, then one CLOSE pulse of 36.0 s. Estimate 50 % and 20 %.
Deadband	Estimate 50 %, demand 51.5 %, deadband 2 %.	No pulse.
Minimum pulse	Error 3.5 % = 4.2 s, minimum pulse 5 s.	No pulse.
End position 100 %	From 80 %, demand 100 %, overdrive 10 s.	OPEN 34.0 s, estimate 100 %, no further pulses.
End position 0 %	From 100 %, demand 0 %, overdrive 10 s.	CLOSE 130.0 s, estimate 0 %, Synchronized = 1.
Reversal pause	Demand reverses during an OPEN pulse, pause 5 s.	CLOSE starts exactly 5.0 s after OPEN ends.
No overlap	About 1 h of random demand and Enable changes.	171 pulses, OPEN and CLOSE never on together, no pause violated.
Disable during pulse	Enable = 0 after 30 s of a 60 s OPEN pulse.	Both relays off in the same cycle, estimate stays at 25 %.
Start-up sync	Modes 1, 2 and 0. Mode 1 with Enable delayed by 20 s.	Closed: CLOSE 130 s (overdrive 10 s), 0 %. Open: OPEN 120 s, 100 %. None: no pulse. Sync starts when Enable = 1.

Acceptance test	Procedure	Result
Invalid settings	Each of the 9 settings set out of range during a pulse, and one from power-up.	Fault = 1 and both relays off in the same cycle. No pulse from power-up. Fault clears when corrected.
Periodic re-sync	Re-sync 1 h, demand 70 %.	After 1 h: OPEN 120 s to 100 %, then back to 70 %.
Timer overflow	Pulse across the GET_TIME overflow (49.7 days).	Pulse length correct (60.0 s).

7.2 Check in the ALP simulator or on hardware

1. Set Enable = 1 and Demand = 50 %. About one sample period later the OPEN relay switches on for about 60 s (travel time 120 s). Position shows 50 %.
2. Set Demand = 51 %. No relay switches.
3. Set Demand = 20 %. After the reversal pause and the next sample, the CLOSE relay switches on for about 36 s.
4. Start a pulse and set Enable = 0. Both relays switch off at once.
5. Set travel time open to 0. Fault = 1, both relays off. Restore the value. Fault = 0.
6. Set Demand = 100 %. The OPEN relay runs for the remaining travel plus overdrive, then stays off.