

# XD-C Controller manual – 3.0

The XD-C is a versatile single-axis piezo stage controller, supporting a wide variety of use cases and communication options for our stages. Designed for OEM and scientific applications, easy single-axis use is supported. A Windows GUI is included for plug-and-play testing.



Image for illustration only. May not be an exact representation of the product.



PCB-only version

## 1. Key specifications

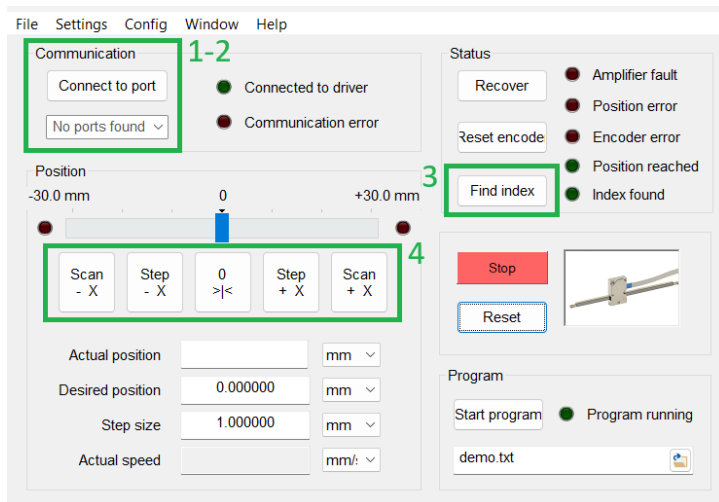
Compatible products	XLS series, XVS series, XRT-U series, XVP series
Number of axes	Single-axis
Power supply	48 VDC (screw terminal), 5W
Temperature range	-30°C to +70°C
Dimensions	80 x 54 x 23 mm
Dimensions (PCB-only version)	50 x 50 x 30 mm
Connector to stage	15-pin D-Sub HD female
Control types	Closed loop, open loop, hybrid
Key communication options	USB, UART, GPIO, I2C, SPI, ...
Software support	Windows GUI, C++, Python, LABView
Motion profile	Point-to-point trapezoidal
Encoder compatibility	Optical, incremental
Supported resolutions	Up to 78 nm (linear), up to 3 $\mu$ rad (rotary)
Error detection	Overcurrent, thermal, short-circuit, ...

## 2. Content

XD-C Controller manual – 3.0 .....	1
1. Key specifications .....	1
2. Content .....	2
3. Getting started .....	2
4. Control of a stage .....	3
5. Communication using USB / UART .....	3
5.1. Format for instructions .....	3
5.2. List of commands .....	5
5.3. Feedback from controller .....	10
6. Communication using GPIO .....	12
7. Communication using I2C / SPI .....	13
8. Connections / pin layouts .....	14
9. Tuning .....	17
10. Windows GUI .....	18
10.1. Required files .....	18
10.2. Commands for the Windows GUI demo program .....	19
11. Python, MATLAB, C++ and LabVIEW .....	21
12. Support .....	21

## 3. Getting started

All controllers are delivered with a power adapter, USB cable and USB stick with software to control stages for easy plug-and-play use. To get started, plug the stage connector into the XD-C. Connect the XD-C to a computer with the provided USB cable and open up the Windows GUI, press SCAN or MOVE and the stage will start moving.



A quick overview of the Windows GUI:

To control a stage:

1. Connect the correct COM port.
  2. Load *settings\_default.txt* from USB stick.
  3. Press “Find Index”.
- The stage will start moving and look for the encoder index.
4. You can start controlling the stage:
    - a. SCAN: move with continuous speed.
    - b. STEP: move with specified step size.
    - c. 0: move to home position.

Using the XD-C controller, a Xeryon stage can be used to create a wide variety of motion patterns:

- Short- or long-range motion, high- or low-speed motion
- Repetitive or dynamic motion, smooth continuous motion
- Set-and-forget applications, controlled damping (soft landing) motion
- A combination of the above, and more

### Tuning

Xeryon stages work based on ultrasonic piezo technology. This technology is frequency-based, and differs slightly from both a) electromagnetic motors and b) other piezo systems. As a result, motion characteristics may differ from conventional motion systems. Several parameters can be used for finetuning motion behaviour. Motion trajectory is split into two zones: bulk move (or ‘scanning’) and final positioning. When experiencing unexpected behaviour, the following simple tuning procedures can be followed (Settings -> Edit). In-depth tuning info can be found in chapter 9.

<b>Behaviour</b>	<b>Final positioning (zone 1)</b>	<b>Bulk move / scanning (zone 2)</b>
<i>Vibration</i>	Lower proportional factor	Lower proportional factor
<i>Noise</i>	Adjust frequency	Adjust frequency
<i>Overshoot</i>	Lower deacceleration	Lower deacceleration
<i>Force too low</i>	-	Lower frequency
<i>Inaccurate landing</i>	Lower positioning tolerance	-
<i>Slow landing</i>	Adjust proportional factor	-

Note: for vertical or inclined use with higher payloads, advanced tuning may be necessary. For assistance, please contact [support@xeryon.com](mailto:support@xeryon.com). More information about the Windows GUI: ch. 9.

## 4. Control of a stage

---

The controllers can be controlled using various methods:

- Using USB / UART see [section 5](#)
- Using GPIO see [section 6](#)
- Using I2C / SPI see [section 7](#)

## 5. Communication using USB / UART

---

- A host computer or controller can communicate with the XD-C via the USB configured as a virtual COM port. The baud rate is automatically detected by the controller and can be up to 115200. The protocol uses 8 data bits, 1 stop bit, no parity bit, no handshaking. This section consists of three parts:**Fout! Verwijzingsbron niet gevonden.**
- 5.2 List of **commands**
- 5.3 Feedback from controller

**Note:** To enable UART, send UART=9600 command and use the following pins:

Ground (pin 2), 5V reference (5), Tx (12), Rx (18).

### 5.1. Format for instructions

A command line consists of maximum 16 characters followed by a ‘new line’ character (ASCII code 10). The command has the following fixed format:

X:DPOS=-12345678

- 1 character defining the axis, followed by a colon.
- 4 characters for the command.
- '=' sign separating the command from the corresponding value.
- Optional sign.
- Decimal value up to 8 decimal places (9 if the sign is omitted).
- Maximum total of 16 characters.

The characters have to be sent from left to right, in the example above starting with 'X' and ending with '8'. The command tags are in upper case. The instruction should be terminated with a 'new line' character (ASCII code 10). The driver processes the instruction immediately after receiving this 'new line' character.

Some instructions such as 'ZERO' and 'RSET' require no value. In that case, it is sufficient to send only the command itself, e.g. 'ZERO' followed by the 'new line' character.

### Value range

There are 9 characters reserved for the value including its sign. For signed values 8 decimal places are available, giving a range from -99 999 999 to +99 999 999. For positive numbers, the '+' sign can be omitted, increasing the positive range to 999 999 999. No spaces, commas or periods should be added to the numbers. Only integers are allowed.

- X:DPOS=-99999999
- X:DPOS=+99999999
- X:DPOS=99999999

### Request a value

To request the value of a certain setting, put '=' after the parameter for which you want to know the value, e.g. EPOS=? gives the controller a request for the encoder position. FREQ=? asks the controller for the current excitation frequency. This works best with INFO=0, otherwise the reply disappears in the constant flood of feedback data.

### Units are as follows:

Type	XLS / XVP series	XRT series	Resolution
Time, delays	ms		1 ms
Target position, step size	encoder units		1 encoder increment
Speed	µm/s	deg/s (*)	1 µm/s or 0.01 deg/s
Frequency	Hz		1 Hz

(\*) Conversion factor of 100 required: e.g. enter SSPD=10000 for 100 deg/s.

## 5.2. List of commands

This is a list of all possible commands that can be sent and their use. This part is divided into:

- I. [Motion](#)
- II. [Settings handling](#)
- III. [Communication](#)
- IV. [Tuning](#)
- V. [Signal \(advanced\)](#)
- VI. [Directional settings \(advanced\)](#)
- VII. [Trigger outputs \(advanced\)](#)

I. Motion commands		
Command	Range	Explanation
<b>INDX</b>	0, 1	Find the index. A value of 0 or 1 indicates the initial search direction. The controller sets off in the specified direction to search. When a mechanical limit is reached (detected by position error > ILIM) it reverses the search direction. After finding index, stage is positioned at the index position.
<b>INDA</b>	1 bit	Automatic detection of encoder index.
<b>HOME</b>	-	Go to the home position. This equals DPOS=0.
<b>DPOS</b>	26 bits	Set target position. Closed-loop control is used to reach and maintain the new position. The position is expressed in encoder units. Positive and negative values are allowed within the range of the stage.
<b>STEP</b>	26 bits	Move relative to the current position, over a specified distance. When already in closed loop, the current desired position is used as a reference. When before in open loop, the actual position (encoder value) is used as a reference. The command value specifies the step size in encoder increments. Positive values send the stage towards higher encoder values, negative values send the stage towards lower encoder values. Closed-loop control is used to reach and maintain the new position.
<b>SCAN</b>	-1,0,1	Continuously move with fixed speed. The speed is maintained by closed-loop control. A positive number sends the stage towards increasing encoder values, a negative number sends the stage towards decreasing encoder values. A zero value stops the stage.
<b>SSPD</b>	24 bits	Set speed. Used as scanning speed (SCAN command) and as target speed towards the next target position (DPOS and STEP). Unit is 1 $\mu\text{m/s}$ or 0.01 deg/s. Default: 10000 (10 mm/s or 100 deg/s).
<b>ISPD</b>	24 bits	Set the speed which is used while searching the index. Unit is 1 $\mu\text{m/s}$ or 0.01 deg/s.
<b>ACCE</b>	16 bits	Set acceleration for speed profile. Expressed in $\text{m/s}^2$ . Default value: 255.
<b>DECE</b>	16 bits	Set deceleration for speed profile, when approaching target position. Default / maximum value: 255.
<b>LLIM</b>	26 bits	Set low-side soft end stop. Expressed in encoder units.

<b>HLIM</b>	26 bits	Set high-side soft end stop. Expressed in encoder units.
<b>MOVE</b>	-1,0,1	Continuously move in open loop. Phase and amplitude influence the speed, but speed is not controlled. A positive number sends the stage towards increasing encoder values, a negative number sends the stage towards decreasing encoder values. A zero value stops the stage.
<b>PHAS</b>	16 bits	Set the phase offset between the excitation signals. Can be used to control the speed in open loop. Input values 0-65535 correspond to a phase shift of 0-360°. Around 16384 the phase corresponds with a MOVE=1 direction, around 49152 (= -16384) it corresponds to a MOVE=-1 direction.
<b>AMPL</b>	16 bits	Set amplitude for open-loop piezo excitation signals.
<b>STOP</b>	-	Stop the stage.
<b>CONT</b>	-	Continue movement after a stop command.
<b>ENBL</b>	1 bit	Enable drive. Or: If an actuator is in error mode, you can recover from that by sending "ENBL=1". This enables the actuator again.
<b>ENBR</b>	1 bit	Enable drive on power-up automatically.
<b>FILE</b>		Filters fast variations from the encoder values (noise). Default value: 0, sets filter to a minimum.
<b>FILG</b>		Extracts the spikes from the raw encoder inputs. Defines the minimum pulse width that may be responded to. Default value: 0
<b>INTF</b>		Controller integration factor. Default value: 1. Minimum = 1
<b>DTIM</b>		Determines how long (in ms) the motor must be turned off before it can be turned on again. Ensures that the tage is not over-excited when landing by switching it on and off too quickly. Default value: 0

II. Manage settings		
Command	Range	Explanation
<b>RSET</b>	-	Reset the driver. All piezo signals go to zero and settings are reset to their saved value.
<b>LOAD</b>	-	Load settings from memory.
<b>SAVE</b>	-	Save settings to memory.
<b>BLCK</b>	0-1	Blocked mode. When blocking mode is enabled (1), the ENBL command should be send to the controller if an error occurs. If blocking mode is disabled (0), the controllers executes the next command even when an error occurs.
<b>TEST</b>	0-1	Test LED indicators. TEST=1 switches all indicators on. TEST=0 brings them back to their function.

III. Communication																		
Command	Range	Explanation																
<b>INFO</b>	4 bits	<p>Select type of info to be transmitted from the controller to the master (PC).</p> <p>0: Stop broadcasting info automatically.            1: SRNO, SOFT, MODEL, STAT, SYNC            2: SRNO, SOFT, MODEL, STAT, FREQ, SYNC, EPOS, DPOS, REQUESTED PARAMETER*, TIME            3: EPOS, DPOS, STAT            4: EPOS, STAT, DPOS, TIME            5: STAT, FREQ, EPOS, DPOS, REQUESTED PARAMETER*, TIME            6: REQUESTED PARAMETER*            7: EPOS, STAT</p> <p>Default: 2</p> <p>e.g. INFO=7 will alternatingly send EPOS &amp; STAT values.</p> <p>* REQUESTED VALUE, when a parameter value is requested (e.g. by sending FREQ=?) the value for the parameter will be returned here.</p>																
<b>UART</b>	0-76800	<p>Set UART baud rate. To switch UART off: UART=0. To set the baud rate to 9600: UART=9600. The maximum baud rate is 76800. When UART is off, the UART can only be restarted by sending the UART command via USB.</p>																
<b>GPIO</b>	0-13	<p>Select the preferred input mode using the GPIO command. For more details see section 5.</p> <table border="1" data-bbox="475 1149 1385 1653"> <tbody> <tr> <td>GPIO=0</td> <td>Control via IO pins switched off. The controller will only react to text commands sent through USB or UART.</td> </tr> <tr> <td>GPIO=2</td> <td>Pulse and direction mode, with direction &amp; enable pins.</td> </tr> <tr> <td>GPIO=3</td> <td>Pulse and direction mode, with forward &amp; backward pins.</td> </tr> <tr> <td>GPIO=4</td> <td>A quad B input mode.</td> </tr> <tr> <td>GPIO=8</td> <td>PWM control, with direction &amp; enable pins.</td> </tr> <tr> <td>GPIO=9</td> <td>PWM control, with forward &amp; backward pins.</td> </tr> <tr> <td>GPIO=12</td> <td>Analog control, with direction &amp; enable pins.</td> </tr> <tr> <td>GPIO=13</td> <td>Analog control, with forward &amp; backward pins.</td> </tr> </tbody> </table>	GPIO=0	Control via IO pins switched off. The controller will only react to text commands sent through USB or UART.	GPIO=2	Pulse and direction mode, with direction & enable pins.	GPIO=3	Pulse and direction mode, with forward & backward pins.	GPIO=4	A quad B input mode.	GPIO=8	PWM control, with direction & enable pins.	GPIO=9	PWM control, with forward & backward pins.	GPIO=12	Analog control, with direction & enable pins.	GPIO=13	Analog control, with forward & backward pins.
GPIO=0	Control via IO pins switched off. The controller will only react to text commands sent through USB or UART.																	
GPIO=2	Pulse and direction mode, with direction & enable pins.																	
GPIO=3	Pulse and direction mode, with forward & backward pins.																	
GPIO=4	A quad B input mode.																	
GPIO=8	PWM control, with direction & enable pins.																	
GPIO=9	PWM control, with forward & backward pins.																	
GPIO=12	Analog control, with direction & enable pins.																	
GPIO=13	Analog control, with forward & backward pins.																	
<b>STPS</b>	16 bits	<p>Select the step size for each STEP pulse. The step size is expressed in encoder units. Default: 1 encoder unit.</p>																
<b>POLI</b>	1-65535	<p>Set polling interval. Specifies the time between data updates. Also defines the polling interval of the analog IO. The interval is expressed in milliseconds. The default value is 97 (97 ms).</p>																
<b>DLAY</b>	16 bit	<p>Sets the delay between the moment the stage reaches its target position and the moment the 'position reached' flag is raised. Expressed in milliseconds. Default: 100 (ms).</p>																

IV. Tuning		
Command	Range	Explanation
<b>FREQ</b>	24 bits	Set the frequency of the excitation signals for zone 1. Unit is Hz.
<b>FRQ2</b>	24 bits	Set the frequency of the excitation signals for zone 2. Unit is Hz. Also used for scanning.
<b>PROP</b>	16 bits	Proportional control factor for zone 1.
<b>PRO2</b>	16 bits	Proportional control factor for zone 2.
<b>ZON1</b>	26 bits	Width of zone 1: Expressed in mm.
<b>ZON2</b>	26 bits	Width of zone 2: Expressed in mm.
<b>DUCO</b>	1 bit	Amplitude is used in closed loop if set to 1. If set to 0, a fixed amplitude of 50% is used. Default: 1.
<b>ELIM</b>	20 bits	<p>ELIM (error limit) sets the maximum following error. When the following error exceeds the value set by ELIM, then the controller goes in safe mode and the motor signals are switched off. Recovery: RSET or ENBL=1.</p> <p>This error may be triggered when trying to move beyond the physical limits of the stages, or by setting too high a speed. Do not forget to first find the index position (INDX command) to avoid that the stages travels beyond the end stops and triggers this error. Default: 10000.</p>
<b>ILIM</b>	26 bits	Sets the following error at which the index search algorithm reverses direction. This influences the time the targe stalls at the end position during an index search.
<b>SLIM</b>	26 bits	Error saturation limit
<b>PTOL</b>	16 bits	<p>Position tolerance. When the stage is within +/- position tolerance of the desired position, the control is switched off and the 'position reached' flag is raised.</p> <p>Values are expressed in encoder units and should be in the range 0 – 65535. The range is applied symmetrically with respect to positive and negative position errors. e.g. PTOL=2 allows s errors between -2 and +2 encoder units. Default: 2. See also TOUT and PTO2.</p>
<b>PTO2</b>	16 bits	Second position tolerance, similar to PTOL. Comes into action if first position tolerance PTOL fails within a timeout time TOUT. The default value is 10.
<b>TOUT</b>	16 bits	<p>Set timeout time. To avoid that the stage keeps vibrating indefinitely around the desired position without 'landing', a timeout time can be set. The timer starts when the stage is near the desired position, within a distance of +/- PTO2. After passing the timeout time, PTO2 becomes the new position tolerance.</p> <p>The time is expressed in milliseconds. The default value is 1000 (ms).</p>
<b>TOU2</b>	16 bits	This defines a safety timeout. When the motor is on for a time longer that the value set by TOU2, then the controller goes in safe mode and the motor



		signals are switched off. Recovery: RSET or ENBL=1 depending on BLCK. Status bit #18 goes up when this timeout is triggered. TOU2=0 disables this timeout. Any other value sets the timeout time in seconds. Maximum value: 65535 seconds.
<b>TOU3</b>	16 bits	This defines another safety timeout. When the stage is trying to “land” to a specific position for longer than the value set by TOU3, then the controller goes in safe mode. The motor signals are switched off.
<b>ENCR</b>	1 bit	Reset the encoder by sending “ENCR=1”.

<b>V. Signal shaping (advanced use)</b>		
Command	Range	Explanation
<b>ENBL</b>	0-3	Enable amplifiers. Bit 0 is for piezo signal 1, bit 1 for piezo signal 2. ENBL=3 enables both amplifiers, ENBL=0 disables both amplifiers. ENBL=1 enables only amplifier 1, ENBL=2 enables only amplifier 2.
<b>ENBR</b>	0-1	Enable amplifiers automatically upon controller start-up.
<b>ZERO</b>	-	Force the piezo signals to zero volt.
<b>MAMP</b>	16 bits	Set maximum amplitude. The piezo excitation signals are limited to the corresponding voltages. MAMP=65535 sets them to the maximum voltage of 45 V. MAMP=36400 sets the maximum to 25 V. The relation is linear.
<b>MIMP</b>	16 bits	Set minimum amplitude for piezo excitation signals. See MAMP for values.
<b>PHAC</b>	16 bit	Phase correction. Corrects an imbalance in the motor. Such imbalance may cause a rattling or scratching noise when the stage moves at low speed. Practical values are in the range of a few 1000, positive or negative. Default: 0 (no correction)
<b>OFSA</b>	12 bits	Offset on the piezo signals on piezo phase 1. OFSA=4095 corresponds to full scale (45 V), OFSA=0 produces no offset. The relation is linear.
<b>OFSB</b>	12 bits	Similar to OFSB, but for piezo phase 2.
<b>FILP</b>	8 bits	Filter speed for phase of piezo excitation signals. Default value: 1. Max. value: 255.
<b>FILA</b>	8 bits	Filter speed for amplitude of piezo excitation signals. Default value: 1. Max. value: 255.

<b>VI. Directional settings (advanced use)</b>		
Command	Range	Explanation
<b>ENCD</b>	1 bit	Set the encoder direction. Set the counting direction with respect to the A/B signals or sin/cos signals of the encoder. Flip this bit to swap left and right, or clockwise and counter-clockwise. Default value is 0.

<b>ENCO</b>	32 bits	Sets the encoder offset: distance between the index position and the desired zero position. In encoder units. Default value is 0.
<b>ACTD</b>	1 bit	Set the actuation direction. If not set correctly, the stage will move away from the desired position. Default value is 0.
<b>PATH</b>	1 bit	For rotation stages only. Selects whether the stage will follow the shortest path (PATH=1) to the target position or follow a linear approach, respecting high to low or low to high (PATH=0). Default: 1 for rotation stages, 0 for linear stages.

<b>VII. Trigger outputs (advanced use)</b>		
Command	Range	Explanation
<b>TRGS</b>	26 bits	Start of the trigger pulses, expressed in encoder units.
<b>TRGW</b>	26 bits	Width of the trigger pulses, expressed in encoder units. This (positive) value should be lower than the pitch (TRGP).
<b>TRGP</b>	26 bits	Pitch of the trigger pulses, expressed in encoder units. The pitch can't be negative. To trigger pulses in the negative direction: set the TRGS at the most negative point and use a positive pitch.
<b>TRGN</b>	26 bits	Number of trigger pulses.

### 5.3. Feedback from controller

<b>Status LEDs (Only for XD-C PCB)</b>	
<b>Green LED</b>	Power on for controller board
<b>Orange LED</b>	Motor on
<b>Red LED</b>	Disabled (ENBL=1 to enable) OR error (position failure, amplifier fault, timeout)

Information is sent back from the Xeryon controller to the master in ASCII format. The format is as follows:

1. Tag: Four characters describing the type of information
2. '=' sign separating the command from the corresponding value
3. Signed value associated with that information (sign + 8 decimal places). The message is terminated with a 'new line' character (ASCII code 10).

e.g. EPOS=12345678

#### **Different types of information:**

The command INFO determines which information is sent back. See the example below (INFO=2).

Tag	Explanation
SRNO	Serial number of the driver (hardware)
SOFT	Software version installed on the driver. e.g. 20103 → 2.1.3

[TYPE]	Type of motion device (XLS, XRT, XVP) and its resolution e.g. XLS1=312
STAT	<a href="#">Status (see below)</a>
FREQ	Excitation frequency currently in use
SYNC	Fixed value "12345678". Can be used for debugging communication issues.
EPOS	Encoder position
DPOS	Desired position
TIME	Time stamp: resolution 0,1 ms

### Meaning of STAT(US):

The Status contains 24 bits:

Status bit	Name	Explanation
0	Amplifiers enabled	XRTA only: Amplifiers for phase 1 and 2 enabled
1	End stop	Stage stopped by end stop
2	Thermal protection 1	Amplifier for phase 1 or 3 in thermal protection.
3	Thermal protection 2	Amplifier for phase 2 or 4 in thermal protection.
4	Force zero	Motor signals are currently forced to zero.
5	Motor on	The piezo motor is on.
6	Closed loop	The stage is currently in closed loop control.
7	Encoder at index	Indicates whether the stage is positioned exactly at the encoder index.
8	Encoder valid	Indicates whether the encoder index has been passed and therefore the encoder value reflects the absolute position, not the relative position with respect to the startup position.
9	Searching index	Indicates whether the stage is currently searching the index position.
10	Position reached	Indicates whether the target position is reached (within tolerance limits).
11	Error compensation	Error compensation is on.
12	Encoder error	Indicates an error produced by the encoder.
13	Scanning	Indicates whether the stage is in a scanning mode.
14	Left end stop	Indicates that the left end stop is passed.
15	Right end stop	Indicates that the right end stop is passed.
16	Error limit	Indicates that the position error has reached the limit set by ELIM. This can indicate a collision or mechanical limit (end of stroke).

17	Searching optimal frequency	The driver is searching for the optimal excitation frequency of the piezo motor.
18	Safety timeout triggered	If this is set to “1”, then the safety timeout was triggered. See the explanation of the command “TOU2” several pages back.
19	EtherCAT acknowledge	Only used when control via EtherCAT.
20	Emergency stop	Not used.
21	Position fail	If this is set to “1”, then the safety timeout TOU3 was triggered.

## 6. Communication using GPIO

There are 4 different configurations for the use of the digital and analog IO pins.

1. Control the position of the stage using [pulses](#). Each pulse does a step (size) in a certain direction.
2. Control the position of the stage using an [encoder-like signal](#).
3. Control the speed of the stage using a [PWM input signal](#).
4. Control the speed of the stage using an [analog input](#).

All of these methods are described more in detail below. To select a method, the command “GPIO” is used. This can be found in section. This command can be sent over USB or UART. **Corresponding pins** can be found in section 6.

The GPIO settings can be stored in memory using the “SAVE” command. That way, the GPIO mode will become active at power-up without having to first send the GPIO command via USB, UART or any other method.

The GPIO input and commands can be used together: the controller will react to both GPIO inputs and text commands. You don’t have to go back to GPIO=0 to send text commands. You can send text commands also in the GPIO=2, 3, 4, ... modes. But when going from GPIO input to text commands, first send a STOP command. Recommended maximum pulsing frequency: 20 kHz.

### 5.1 Pulse and direction mode

This mode is activated by the command GPIO=2 or 3.

GPIO=2 enables the input signals: ground (pin 19), pulse (23), direction (25), enable (27), index (30).

GPIO=3 enables the input signals: ground (pin 19), pulse (23), forward (25), backward (27), index (30).

On each positive edge of the PULSE signal, the target position is incremented or decremented with a particular step size. The default step size is 1 (1 encoder unit). This step size can be changed with the command STPS, e.g. STPS=10.

The DIRECTION signal determines the direction: incrementing or decrementing the target position. The ENABLE signal enables the input, it has to be high in order to start moving.

An alternative to the DIRECTION and ENABLE signal is the FORWARD and BACKWARD signal. By setting one of them high, it’s possible to select the direction of moving.

The INDEX signal is an analog input used as digital input. When going high (positive edge) the controller searches the index (3.3 V is sufficient, max. 10 V).

### 5.2 Encoder-like signal (A quad B input)

This mode is activated by the command GPIO=4

Input signals: ground (pin 19), A (23), B (25), enable (27), index (30).

Encoder-like A quad B signal used as input to adapt the target position.

### 5.3 PWM control

This mode is activated by the command GPIO=8 or 9

GPIO=8 enables the input signals: ground (pin 19), PWM (23), direction (25), enable (27).

GPIO=9 enables the input signals: ground (pin 19), PWM (23), forward (25), backward (27).

The frequency of the PWM signal can be set by the command PWMF. PWMF=1000 sets the PWM frequency to 1000 Hz.

The speed is proportional to the pulse width. When the pulse width is 50 %, speed is set to 50 % of the speed set by the SSPD command. When the signal is all the time high, then 100 % of SSPD is selected. Speed can be controlled from 0 to 100 %.

The use of the direction, enable, forward and backward signals is the same as in pulse and direction mode.

### 5.4 Analog controls

This mode is activated by the command GPIO=12 or 13

GPIO=12 enables the input signals: ground (pin 19), speed (29), direction (25), enable (27).

GPIO=13 enables the input signals: ground (pin 19), speed (29), forward (25), backward (27).

The speed input is proportional to the voltage applied to the speed input pin. 10 V corresponds to 100 % of the speed set by the SSPD command.

The use of the direction, enable, forward and backward signals is the same as in pulse and direction mode.

## 7. Communication using I2C / SPI

---

This controller supports both I<sup>2</sup>C (Inter-Integrated Circuit) and SPI (Serial Peripheral Interface) communication protocols for flexible integration with external devices.

**I<sup>2</sup>C:** A two-wire communication protocol (SCL and SDA) suitable for connecting multiple devices using a shared bus. Ensure proper pull-up resistors are included for reliable operation.

**SPI:** A high-speed, four-wire protocol (SCK, MOSI, MISO, CS) designed for point-to-point or master-slave configurations. Multiple slave devices can be managed using independent Chip Select (CS) lines.

Refer to the pinout diagram and specifications for wiring details and compatible voltage levels.

## 8. Connections / pin layouts

### 8.1. Connection to stage

Stage connector: 15-pin D-Sub HD Female

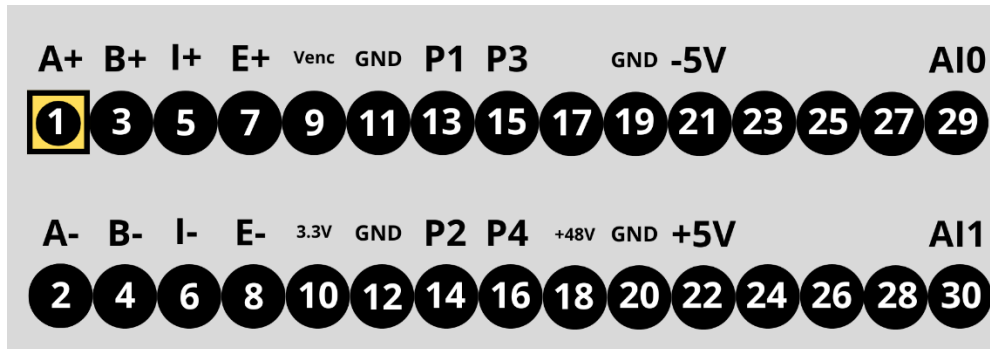
Pin 1 is engraved.



PIN #	SIGNAL	FUNCTION	IN/OUT	PIN #	SIGNAL	FUNCTION	IN/OUT
1	NC	Not Connected	-	9	CLK+/E+	Clock / Encoder +	OUT
2	ENC PWR	Encoder Power	OUT	10	GND P1 / GND P2 / P3	Phase / Ground	OUT
3	ENC GND	Encoder Ground	OUT	11	ENC-	Encoder Index -	IN
4	P2	Piezo Phase 2	OUT	12	ENC A-	Encoder A-	IN
5	P1	Piezo Phase 1	OUT	13	ENC B-	Encoder B-	IN
6	ENC+	Encoder Index +	IN	14	NC	Not Connected	-
7	ENC A+	Encoder A+	IN	15	P4	Piezo Phase 4	OUT
8	ENC B+	Encoder B+	IN				

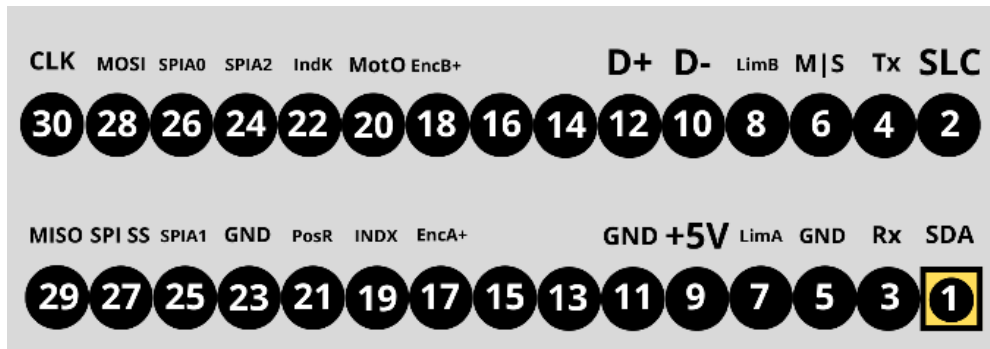
## 8.2. Connection to GPIO banks 1 & 2

IO pins (1.17 mm pitch, 2 rows, 30 contacts)



Pin layout bank 1

BANK 1 (buffered)									
Pin #	Symbol	Function	In/Out	V	Pin #	Symbol	Function	In/Out	V
1	A+	Positive encoder signal A	In/Out	5V	2	A-	Negative encoder signal A	In/Out	5V
3	B+	Positive encoder signal B	In/Out	5V	4	B-	Negative encoder signal B	In/Out	5V
5	I+	Positive encoder index	In/Out	5V	6	I-	Negative encoder index	In/Out	5V
7	E+	Positive encoder error	In/Out	5V	8	E-	Negative encoder error	In/Out	5V
9	Venc	Encoder supply out	Out	5V	10	3.3 V	3.3 V out	Out	3.3V
11	GND	Ground	In		12	GND	Ground	In	
13	P1	Piezo phase 1	Out	48V	14	P2	Piezo phase 2	Out	48V
15	P3	Piezo phase 3	Out	48V	16	P4	Piezo phase 4	Out	48V
17	-	Not connected	-		18	48 V	Power supply	In	48V
19	GND	Ground	In		20	GND	Ground	In	
21	-5 V	-5 V out	Out	-5V	22	5 V	5 V out	Out	5V
23	DI0	Pulse / A / PWM*	In	3.3V	24	DO0	Trigger pulse (see <a href="#">TRGS command</a> )	Out	3.3V
25	DI1	Direction / forward / B*	In	3.3V	26	DO1	Encoder valid signal	Out	3.3V
27	DI2	Enable / backward*	In	3.3V	28	DO2	Position reached signal	Out	3.3V
29	AI0	Analog speed input*	In	3.3V	30	AI1	Find index*	In	3.3V



Pin layout bank 2

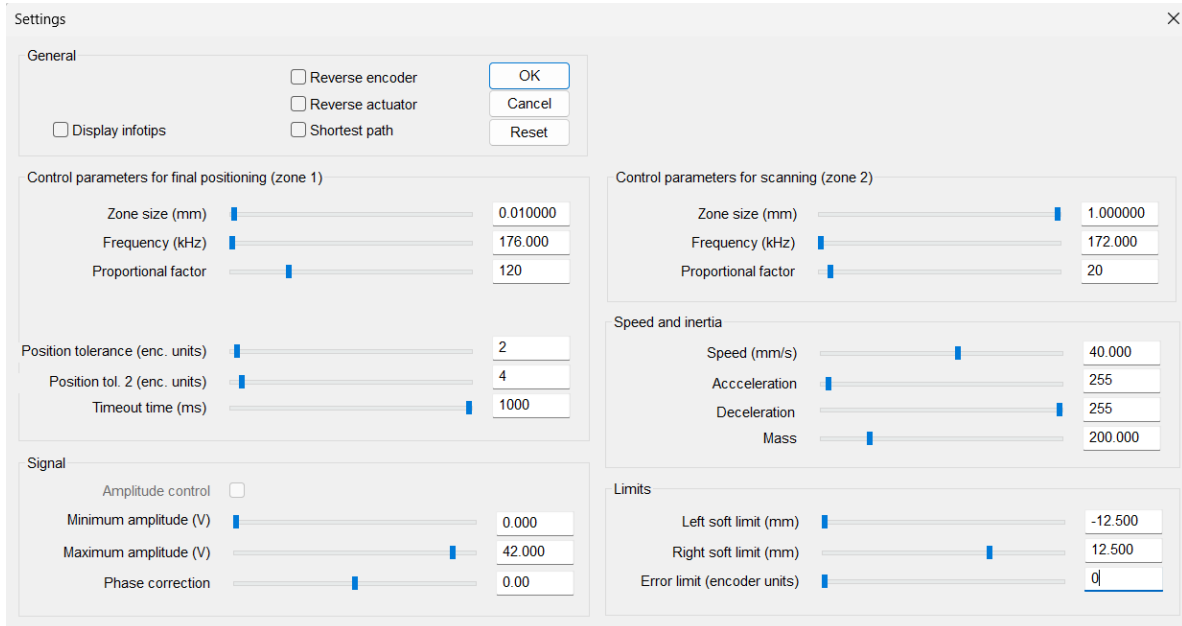
BANK 2									
Pin #	Symbol	Function	In/Out	V	Pin #	Symbol	Function	In/Out	V
1	SDA	I2C data	In/Out	3.3V	2	SLC	I2C clock	In/Out	3.3V
3	Rx	UART Rx*	In	3.3V	4	Tx	UART Tx*	Out	3.3V
5	GND	Ground	In		6	M S	Set as master or slave	In	3.3V
7	Limit A	Passed position limit A	Out	3.3V	8	Limit B	Passed position limit B	Out	3.3V
9	Error	Error, fault	In	3.3V	10	D-	USB - data signal	In/Out	5V
11	GND		In		12	D+	USB + data signal	In/Out	5V
13	Extra 0		In		14	Extra 1		In	
15	Extra 2		In		16	Extra 3		In	
17	Extra 4	A+ encoder (3.3V)	Out	3.3V	18	Extra 5	B+ encoder (3.3V)	Out	3.3V
19	Extra 6	Index flag	Out	3.3V	20	Extra 7	Motor on flag	Out	3.3V
21	Extra 8	Position reached flag	Out	3.3V	22	Extra 9	Index known flag	Out	3.3V
23	GND	Ground	In		24	SPI A2	SPI slave address bit 2	Out	3.3V
25	SPI A1	SPI slave address bit 1	Out	3.3V	26	SPI A0	SPI slave address bit 0	Out	3.3V
27	SPI SS	SPI slave select	In/Out	3.3V	28	MOSI	SPI master out, slave in	In/Out	3.3V
29	MISO	SPI master in, slave out	In/Out	3.3V	30	SPI CLK	SPI clock	In/Out	3.3V



## 9. Tuning

Xeryon's stages are primarily intended for closed-loop control and for this reason already have a position sensor integrated. A control algorithm is implemented on our controller and this algorithm uses multiple control parameters. These control parameters have to be set according to the type of stage, load and customer-specific motion requirement. These parameters are pre-set by Xeryon upon delivery of the stage and controller. In most cases, you don't need to adapt the settings right out of the box. Nevertheless, it may be required that the user modifies these parameters for optimal performance.

Below you can find the corresponding commands for the most common settings:



The screenshot shows a 'Settings' window with the following sections and values:

- General:**
  - Display infotips
  - Reverse encoder
  - Reverse actuator
  - Shortest path
- Control parameters for final positioning (zone 1):**
  - Zone size (mm): 0.010000
  - Frequency (kHz): 176.000
  - Proportional factor: 120
  - Position tolerance (enc. units): 2
  - Position tol. 2 (enc. units): 4
  - Timeout time (ms): 1000
- Control parameters for scanning (zone 2):**
  - Zone size (mm): 1.000000
  - Frequency (kHz): 172.000
  - Proportional factor: 20
- Speed and inertia:**
  - Speed (mm/s): 40.000
  - Acceleration: 255
  - Deceleration: 255
  - Mass: 200.000
- Signal:**
  - Amplitude control
  - Minimum amplitude (V): 0.000
  - Maximum amplitude (V): 42.000
  - Phase correction: 0.00
- Limits:**
  - Left soft limit (mm): -12.500
  - Right soft limit (mm): 12.500
  - Error limit (encoder units): 0

- Reverse encoder/reverse stage: stage specific, don't change.
- Shortest path: PATH
- Zone size, zone 1: ZON1
- Frequency, zone 1: FREQ
- Proportional factor, zone 1: PROP
- Position tolerance: PTOL
- Position tolerance 2: PTO2
- Timeout time: TOUT
- Amplitude control: DUCO
- Min/max amplitude: MIMP/MAMP
- Phase correction: PHAS
- Zone size, zone 2: ZON2
- Frequency, zone 2: FRQ2
- Proportional factor, zone 2: PRO2
- Speed: SSPD
- Acceleration/deceleration: ACCE/DECE
- Mass: MASS
- Left soft limit: LLIM
- Right soft limit: HLIM
- Error limit: ELIM

### Frequencies/proportional values:

Typically, in zone 1 (closest to the target position), the frequency and proportional factor are both chosen higher. This gives a better 'landing' on the target position. For zone 2 (further away from the target position) the frequency is chosen lower to increase speed. At the same time the proportional factor for zone 2 typically has to be chosen lower to avoid instability. Be aware that outside a certain frequency range, the motor will have very limited force (frequency too high) or feature unstable behaviour (frequency too low). A typical frequency difference between FREQ and FRQ2 is between 1

and 3 kHz for a 1N motor and between 0.5 and kHz for a 3N, 5N or 10N motor. The proportional factors in zone 1 (PROP) are typically 2-3 times the value of the proportional factors in zone 2.

When the stage does not want to land on the target position, despite optimizing frequency and proportional factor for zone 1, try to increase the positioning tolerances PTOL and PTO2. See the instruction set for more information.

To obtain the maximum power out of the stage, the frequency in zone 2 (FRQ2) should be set as low as possible without drawing too much current, which results an amplifier fault (status bit 2 & 3)

### **Adding mass to the stage**

As soon as adding a mass of 100g or more, the MASS parameter can be adapted. For every 100g of mass you add, increase the MASS parameter with 100. Finding the optimal MASS parameters is usually obtained by trial-and-error. This parameter is mostly used for heavy payloads on horizontal stages.

### **Changing the dynamics**

Would you like your stage to react faster or slower, the following parameters can be adapted to achieve this.

- SSPD is used to set the maximum speed the stage. Please note, if the frequency in zone 2 is set too low, the speed can't always be reached.
- ACCE defines the acceleration of the stage to the set speed in SSPD. The default value of 65500 means no acceleration limitation is taking into account (i.e. full acceleration).
- DECE defines the deceleration of the stage upon reaching its target. The default value of 65500 means no deceleration limitation is taken into account (i.e. full deceleration).
- PROP & PRO2 are proportional factors used for the closed feedback loop. Both parameters are used for the two different zones (ZON1 and ZON2). Typically PROP is higher than PRO2. Increase them to get faster positioning times.

When the stage needs to move with less overshoot, decrease the above mentioned parameters. Higher proportional factors let the controller react stronger and reduce positioning errors, but can also lead to instability or noisy operation when chosen too high.

## **10. Windows GUI**

---

To provide the user with a quick way to interact with the controlled and the connected stage, a Windows GUI is supplied with every controller. The use is simple and self-explanatory. It can be used for manual input and to run simple scripts.

### **10.1. Required files**

The Windows GUI makes use of the following files:

- The executable of the Windows GUI: Xeryon\_Dialog.exe
- A configuration file named "config.txt". This file should not be edited by the user.
- A default settings file named "settings\_default.txt". The GUI reads this file for initial settings at start-up. Replace or modify this file to alter the default settings. After saving, these values are stored on the driver.
- A settings file named "settings\_user.txt". This file will be created by the program after pressing the "save to file" button.
- A demo program. The file dialog window presents "demo.txt" as default filename.

**Remark:** config.txt has to be in the same folder as Xeryon\_Dialog.exe.

## 10.2. Commands for the Windows GUI demo program

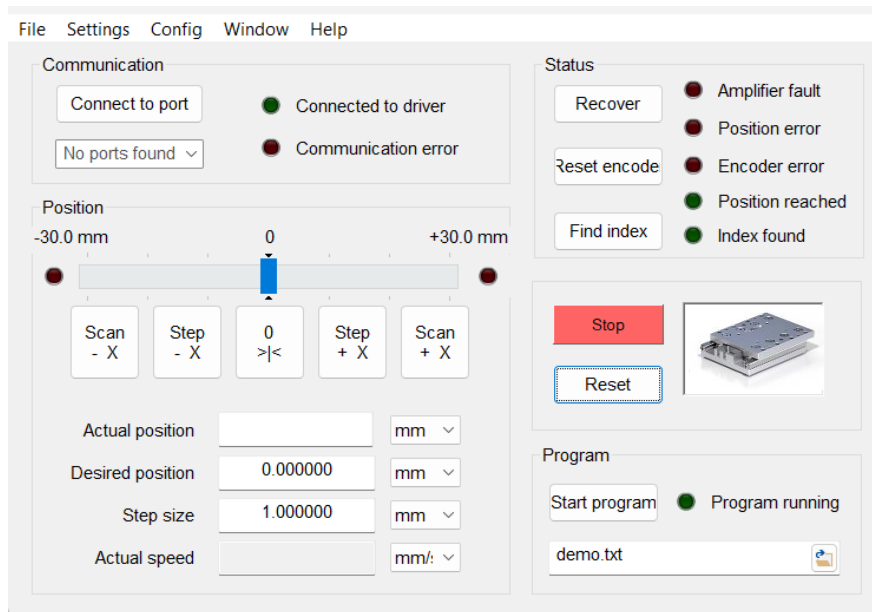
Command	Explanation
BAUD	Set the baud rate for communication.
DPOL	Delay used when polling for a 'position reached' signal after a new target position is set. When DPOL is too small, the Windows GUI may trigger on the 'position reached' status flag of the previous target position due to communication delay. In that case, a succeeding WAIT command will start the timer at the start of the movement instead of after the target has been reached.
HELP	Switch help on or off. HELP=1 switches the info tips on. HELP=0 switches the info tips off.
HALT	Stop the program. (Not to be confused by the STOP command for the driver.)
LABL	Label in the program used by REPT.
LOG	Start or stop logging of data. LOG=1 switches logging on. LOG=0 switched logging off. Data is stored in datalog.csv. When datalog.csv already exists, new data is appended.
MASS	Specifies the mass/inertia of the load on the stage. The Windows GUI calculates the optimal control parameters to obtain stable operation.
MMAS	Maximum mass that can be selected in the Windows GUI.
MPRO	Maximum proportional factor that can be selected in the Windows GUI.
MSPD	Maximum speed that can be selected in the Windows GUI.
PORT	Default port number to appear in the Windows GUI.
REPT	Repeat the above program a specified number of times. The first argument specifies the number of loops. The second argument specifies the label to jump to (label range 0 – 99). If the label does not exist, then the program jumps back to the first line. The REPT command should be placed at the end of the block to be repeated. Nesting of REPT blocks is allowed. Example: REPT=10 2 does 10 loops starting from label 2.
WAIT	Wait a specified time before proceeding to the next command. Time expressed in milliseconds. When WAIT follows a STEP or DPOS command, the timer is started when reaching the target position.

### Example program file (demo.txt)

```

SSPD=100    % Set speed to 100 mm/s or 100 degrees/s
DPOS=0      % Go to position 0
WAIT=100    % Wait 100 ms after arrival at position
LABL=2      % Set label 2
DPOS=60     % Go to position 60 mm or 60 degrees
WAIT=100    % Wait 100 ms after arrival at position
SSPD=10     % Set speed to 10 mm/s or 10 degrees/s
SCAN=1      % Move with constant speed in positive direction
WAIT=2000   % Wait for 2 s (while scan goes on)
REPT=3 2    % Repeat 3 times the code above, starting from label 2
STOP        % Stop stage
DPOS=0      % Finish in the center

```



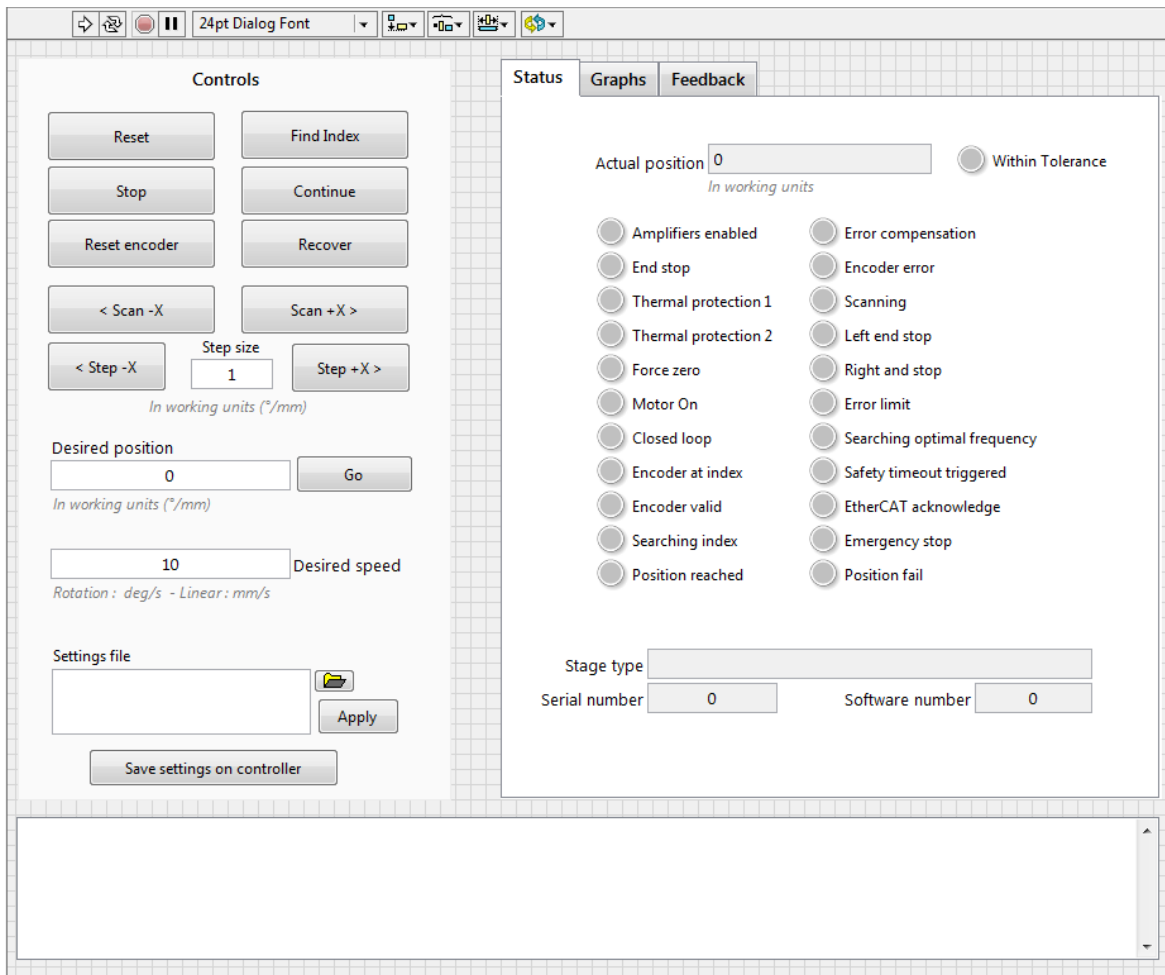
### Function of buttons and status displays:

- 1 Settings button:
  - Load from file: select a file to load your own user settings.
  - Edit: open the settings window (see below).
  - Save to file: save the current settings to a new file.
  - Save on driver: save the current to the controller.
- 1 Save to file: save the settings to a file you select
- 2 Connect to port: when the port is selected, connect to the port and select settings file
- 3 Port: Choose COM port of XD-C controller.
- 4 Connected to driver: successfully connected to the controller.
- 5 Communication error: communication error between the computer and controller.
- 6 Recover: recover the controller after a fault/error.
- 7 Reset encoder: reset encoder count. Index will need to be found again.
- 8 Find index: start indexing procedure.
- 9 Amplifier fault: overcurrent, undervoltage or short-circuit error.
- 10 Position error: position couldn't be reached within set timeout.
- 11 Encoder error: encoder not working or damaged. (commonly caused by short-circuit)
- 12 Position reached: desired position reached.
- 13 Index found: index or home position found.
- 14 Stop: stop the motor.
- 15 Reset: reset the settings back to settings\_default.txt and stop the motor
- 16 Start program: start the program selected below
- 17 Demo.txt: select the demo program
- 18 Blue slider: use the slider to change the desired position.
- 19 Scan + X: the stage will continuously move in the + X direction.
- 20 Step + X: the stage will take a step in the + X direction.
- 21 0 button: move to the middle/index position.
- 22 Actual position: current position of the stage. (can't be changed here)
- 23 Desired position: change the position of the stage here.
- 24 Step size: choose the size of the steps.
- 25 Actual speed: calculated speed of the stage. (will always be lower than actual speed due to polling delay)

## 11. Python, MATLAB, C++ and LabVIEW

Interaction with the XD-C controller and connected stage is also possible through Python, MATLAB, LabVIEW, C++ or any other programming language that supports sending ASCII commands over a (virtual) COM port. Demo programs available for the programming languages mentioned above. The demo's include functions for advanced motion commands, error statuses, settings control and more.

All libraries are available from our website under [Downloads](#).



LabVIEW example

## 12. Support

- Contact: [support@xeryon.com](mailto:support@xeryon.com)
- Address: Interleuvenlaan 21, 3001 Leuven, Belgium
- Phone: +32 (0)16 903 904
- Website: [xeryon.com](http://xeryon.com)