Motorized Focus Lens Series

Rev 2.1

The Motorized focus lens controller unit provides precise control over focus for compatible lenses. The unit simplifies lens operation through a user-friendly interface and eliminates the need for manual adjustments. The motorized focus lens controller offers a seamless integration with external applications via a dedicated Python API, RS-232 protocol or with the provided Windows application.

Technical specifications

Power requirements	24Vdc 1A	
Communication	RS-232 / TTL	
Housing	Metal	
Operating environment	Temperature	0~40 °C
	Humidity	20~85 %
Certification	CE / RoHS	
Included	Focus controller Motorized lens (Cable 0.1 meter) RS-232 9-Pin Sub-D cable (1.4 meter) Lens motor extension cable (1.3 meter)	

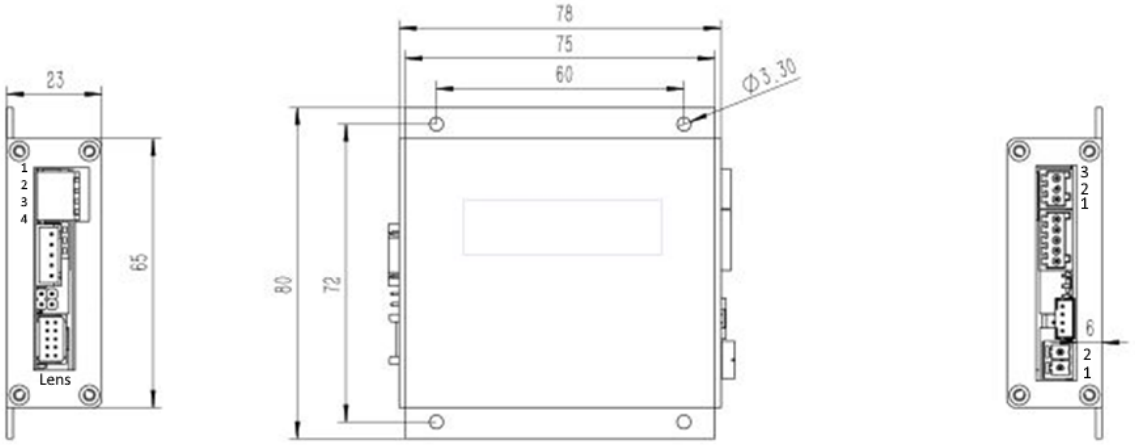
Safety

The lens motor works on high voltage, please read the following safety rules:

- Do not disconnect the lens motor when the controller is still powered
- Do not cut/shorten the lens motor cables

Technical drawing

Controller



Usage

The focus lens has an integrated position sensor that has a resolution of 16384 steps per 360 degrees rotation of the motor. Between the begin and end position of the lens are 2.93 rotations of the motor, approximately 48000 steps. The sensor in the lens only outputs values from 0 to 16384. The controller counts these values to generate a full range of 48000 for this lens. When power is applied to the lens, the current position value of the lens will be reset to 0, even if the position is not at the focus far position. Focusing far will decrement the position value, meaning the position value can become negative. Focusing nearer will increment the position value.

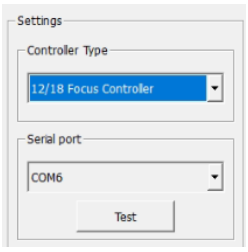
Force Scan

After a new motor is connected, the controller needs to scan for the new motor. This is a one-time process. Before scanning, the focus position should be set to the middle of the range. The process can be started by setting DIP switch 4 to ON (down). The green LED starts flashing. The motor starts turning and searching for the lens' end positions. The LED turns off when scanning is finished. This process is also started after a power-cycle. Scanning is only necessary once and the scanning parameters are stored in non-volatile memory. Turn this switch to OFF (up) after scanning has been done. Scanning can also be initiated using serial commands.

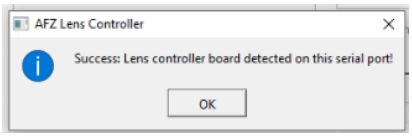
Windows Application

A. Connecting to the controller

1. Connect the controller with a dedicated RS-232 port or a USB > RS-232 cable. Start the application and select the Controller Type = 12/18 Focus Controller, Serial port = COMx. Where x is the port assigned to the board by windows.



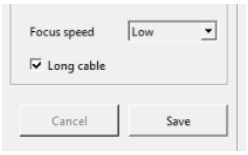
Click on “Test”, when the board is successfully connected the following message is displayed:



When an error message is displayed try the following:

- Try selecting another Serial port from the list
- Check the controller has power connected
-

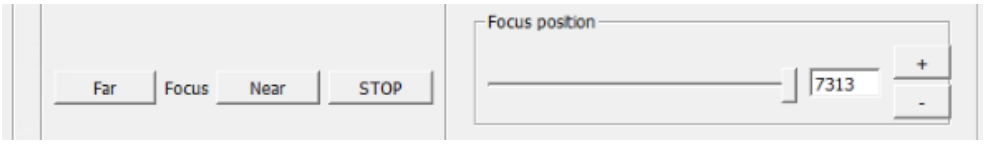
2. After connecting to the board, selecting the connected lens:



3. Select the focus speed; A higher settings will increase the time to adjust but will be less accurate.
4. Press save to store the settings to the controller and enable control.

B. Controlling the lens.

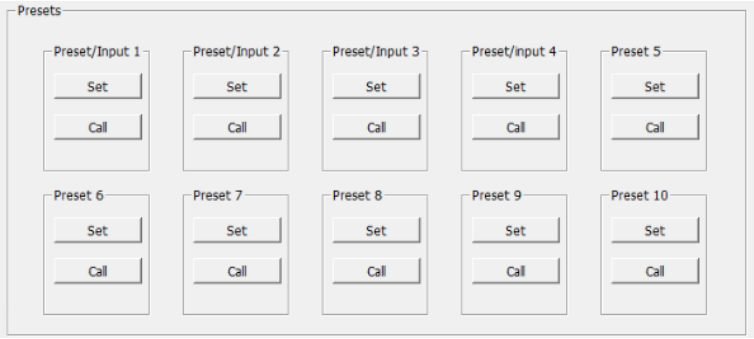
1. The lens can be controlled with the section below.



2. The calibration sequence can also be stared with the button “Calibrate all”
3. The current position can be read from the controller by pressing “Refresh”

C. Using presets

1. 10 Positions can be saved and stored in the software by using the panel below:



2. These positions are not stored in the controller but saved locally on the computer.
3. The absolute position values can be later called when using the API.

API Communication

The controller communicates over a serial connection. DIP-switch 1 determines the voltage level, either RS-232 or TTL-level.

Use the following settings for the serial port:

Parameter	Value
Baud Rate	115200
Data bits	8
Stop bits	1
Parity	No
Flow Control	Off

The controller implements a custom protocol that consists of 16 Bytes

1 Byte	1 Byte	1 Byte	11 Bytes	1 Byte	1 Byte
Address	Command Class	Operand	Parameter	Modbus CRC low Byte	Modbus CRC high Byte

Byte	Description
Address	Not Used; Always 0x01
Command Class	Controller command
Operand	Specific value per command
Parameter	Parameter depending on command
Modbus CRC	16-bit Modbus CRC

The controller supports the following commands

Command	Command class	Operand
Rotate	0x64	0x01
Read motor status	0x65	0x00
Set Debug Mode	0x6A	0x03
Start motor scan	0x6A	0x00

Rotate
The rotate command is shown below

1 Byte	1 Byte	1 Byte	11 Byte			2 Bytes
Addr.	Com.	Op.	Speed (4 Bytes)	Steps (4 Bytes)	Reserved (3 Byte)	CRC
0x01	0x64	0x01	v	0xXXXXXXXX	0x00	0xLLHH

Speed (v) is a positive value defined by the following formula:

$$v = \frac{RPM * 16384}{6000}$$

The amount of steps is a 2's-complement value. The direction is determined by the sign (negative or positive).
The response to this command is an echo of the command.

Read Motor Position
The command to read the motor position is given below,

1 Byte	1 Byte	1 Byte	11 Byte	2 Bytes
Addr.	Com.	Op.	Parameter	CRC
0x01	0x65	0x01	0x000000000000000000	0xLLHH

The returned data is shown below,

1 Byte	1 Byte	1 Byte	1 Byte	4 Bytes	4 Bytes	2 Bytes	2 Bytes
Addr.	Com.	Op.	Motor Status	1-lap position (uint32)	Absolute Position (int32)	Reserved	CRC
0xXX	0x65	0x00	0xXX	0xXXXXXXXX	0xXXXXXXXX	0x000	0xLLHH

Motor status

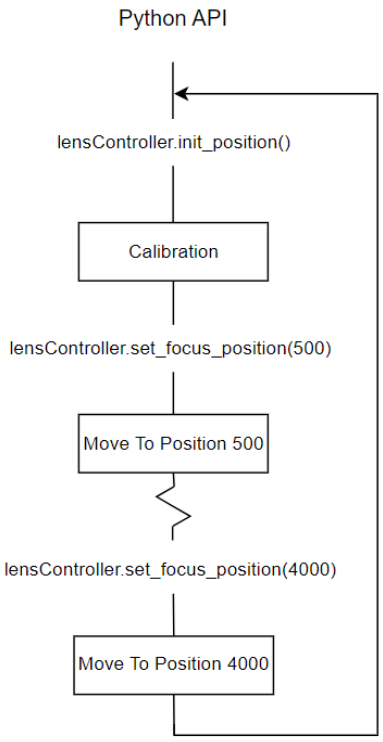
Value	Description
0x00	Stopped
0x01	Positive rotation
0xFF	Negative rotation

1-Lap Position
This is the raw value of the motor sensor. It can output a value of 0-16383 steps, which represents one rotation of the motor. This value is not applicable for use with the focus lens.

Absolute position
The controller uses the value of the sensor in the motor and the number of rotations between the begin and end position of the lens to determine the current position of the motor. After a power cycle, the current position value will be set to 0, even if the current position is not at the focus far position. This means the Absolute position value can become negative.

Python API
See the Python API documentation for more information.

Example usage



Connections

Power

Pin	Function
1	+24V
2	GND

DIP Switch

DIP	Function
1	Serial port levels: ON(down): RS-232; OFF(up): TTL-Level
2	Multiplier for pulse, Leave OFF
3	Multiplier for pulse, Leave OFF
4	Scan enable

Jumpers

Jumper	Function
1	No Jumper; Internal use
2	Jumper placed; internal use

RS-232 / TTL

Pin	Function
1	GND
2	Rx
3	Tx