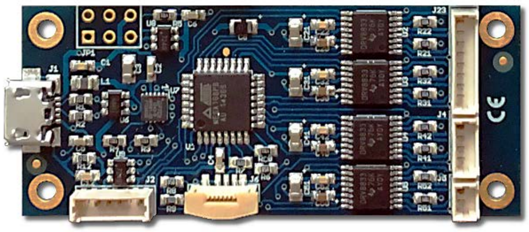
Lens motor control
specification and instructions

MCR600

Theia lens motor controller instructions

This lens control board can be used with Theia Technologies® motorized lenses that use a P-iris.



Supported lenses

TL936P R4
TL936P R6
TL1250P R4
TL1250P R6
TL410P R4
TL410P R6

Two versions of the board are available: one supports the photo interrupter limit switches of the -R6 lenses, the other version does not.

Precautions and disclaimers

- Do not short circuit any part of the board
- Keep the controller board dry
- The controller board contains fuses but proper ESD protection procedures should be followed
- If the controller board becomes too hot disconnect it

- The controller is not designed for use in life sustaining or lifesaving applications.
- The controller is not designed for use in applications where their failure may lead to significant loss of property or health of humans or animals.
- The Information given herein is believed to be accurate and reliable. However, users should independently evaluate the suitability of and test each product selected for their own applications.
- The products described herein are intended for integration. Users must get all required certifications and registrations for the resulting assembly including for the products described herein.
- We reserve the right to make changes to the products described herein without notification. A new specification replaces any previous ones.
- See Theia's standard terms and conditions for warrantee and other information.

Software

Installation

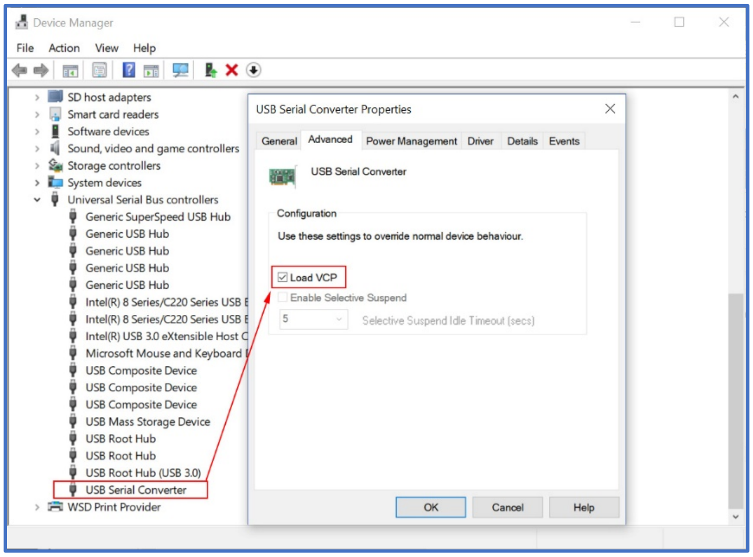
Theia provides a simple control program for interacting with the control board. This application is intended to be used as a test and demonstration program for controlling the lens motors. It can be downloaded from https://www.theiatech.com/lenscontroller/

This software (installation and use) and the driver board are intended to be used by a skilled person in a laboratory or industrial setting. This is not intended to be a fool-proof consumer electronics ready controller.

Download and unzip the motor control program to a folder that is not system protected such as C:\Users\yourAccount\AppData\Local\Theia. Be sure to copy a short cut to your desktop for easy access to the test program.

For initial setup the serial port controller may need to be configured with these following steps. For Windows:

1. Connect USB cable between the lens control board and computer with Windows 7/10
2. Open the "Device Manager"
3. Expand "Universal Serial Bus controllers" and open "USB Serial Converter"
4. In "Advanced" tab, check "Load VCP" (Checkbox must be checked)



Configuration

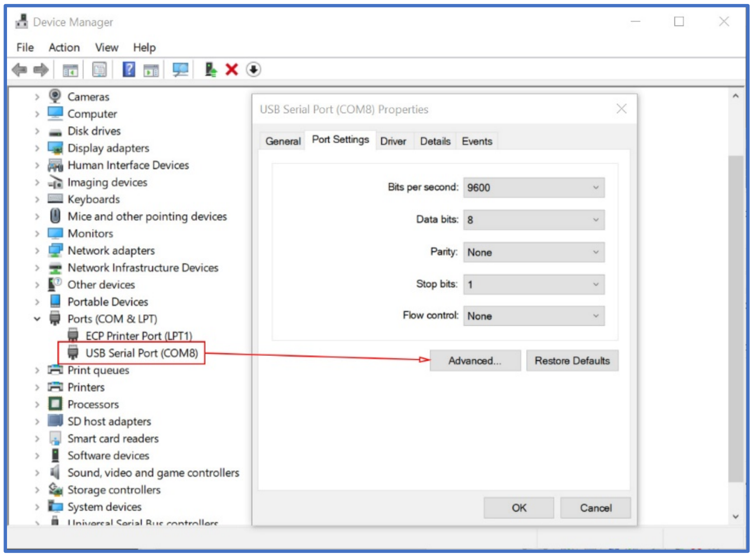
Use these settings to override normal device behaviour.

☑ Load VCP

□ Enable Selective Suspend

5 Selective Suspend Idle Timeout (secs)

5. Click “OK” to save the settings
6. Disconnect the USB cable then reconnect the USB cable.
7. Expand "Ports (COM & LPT)" and open "USB Serial Port (COMx)"
   The COM port for your controller board may have a different number. If the COM port is greater than 9 continue with the following steps.



Bits per second: 9600

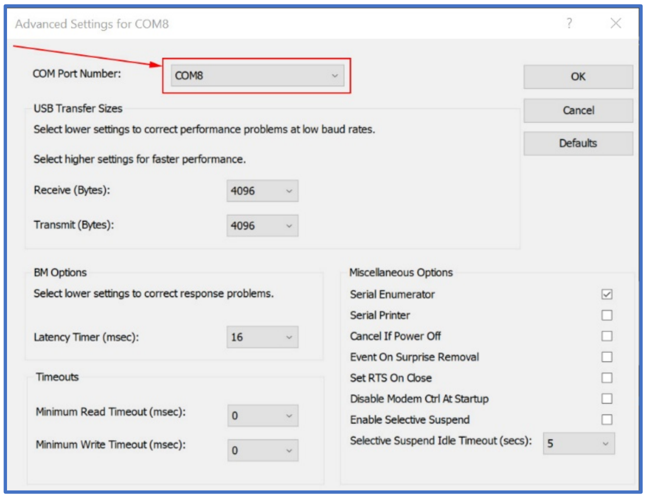
Data bits: 8

Parity: None

Stop bits: 1

Flow control: None

8. In "Port Settings" tab, chose "Advanced .."
9. Set the COM Port below or equal COM9



Advanced Settings for COM8

COM Port Number: COM8

USB Transfer Sizes

Select lower settings to correct performance problems at low baud rates.

Select higher settings for faster performance.

Receive (Bytes): 4096

Transmit (Bytes): 4096

BM Options

Select lower settings to correct response problems.

Latency Timer (msec): 16

Timeouts

Minimum Read Timeout (msec): 0

Minimum Write Timeout (msec): 0

Miscellaneous Options

Serial Enumerator ☑

Serial Printer □

Cancel If Power Off □

Event On Surprise Removal □

Set RTS On Close □

Disable Modem Ctrl At Startup □

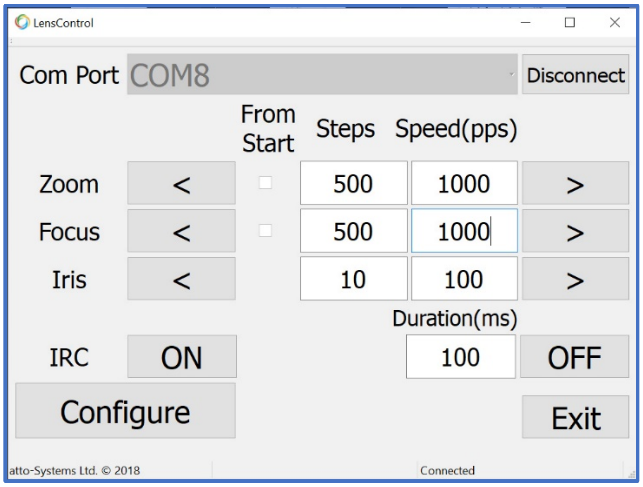
Enable Selective Suspend □

Selective Suspend Idle Timeout (secs): 5

10. Click “OK” and “OK” again. Close the device manager. Your control board should now be setup with the COM port that was selected.

Run the demonstration controller software

Start the LensControl software. Select the correct COM port from the drop down list and “connect” to the motor control board.



LensControl

Com Port COM8

		From Start	Steps	Speed(pps)	
Zoom	<	□	500	1000	>
Focus	<	□	500	1000	>
Iris	<		10	100	>

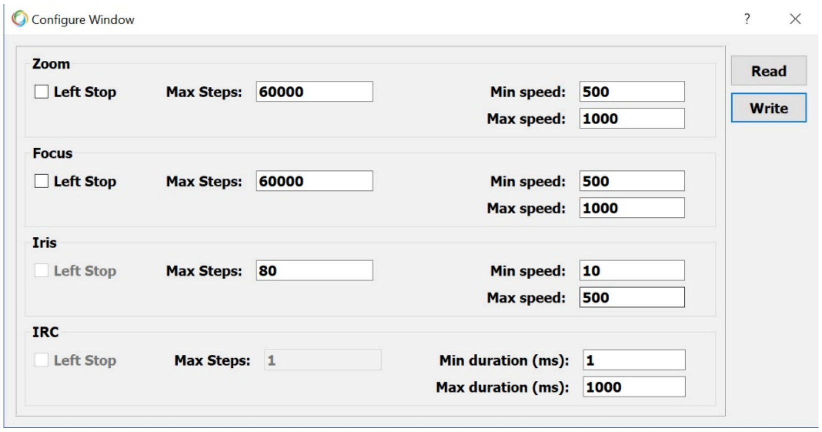
Duration(ms)

IRC ON 100 OFF

atto-Systems Ltd. © 2018

Connected

The configuration settings may need to be adjusted for your lens. Click on “configure” to bring up the parameter limits.



Configure Window

Zoom
□ Left Stop   Max Steps: 60000   Min speed: 500
Max speed: 1000

Focus
□ Left Stop   Max Steps: 60000   Min speed: 500
Max speed: 1000

Iris
□ Left Stop   Max Steps: 80   Min speed: 10
Max speed: 500

IRC
□ Left Stop   Max Steps: 1   Min duration (ms): 1
Max duration (ms): 1000

Check the datasheet of the lens for maximum speeds for zoom and focus. In general, these maximums should be about 1000pps. Be sure to click “write” to store the changed values before closing the configuration window.

Hardware specification

1. Features:

- supports several Theia Technologies'® lenses
- controls P-iris, focus, zoom and IR cut filter motors
- reads photo interrupter limit switches for zoom and focus
- supports USB, I2C, SPI and LV-TTL-UART
- small size of only about 2.4" x 1" (60x25 mm²)
- over-current protection
- over-temperature protection
- single 5V supply (via USB or pin connector)

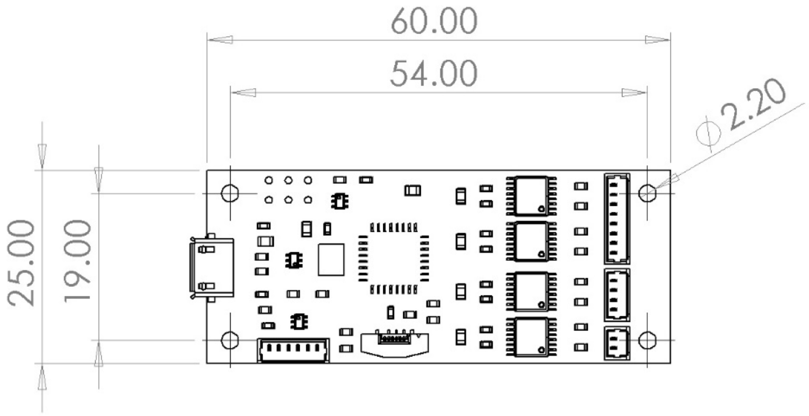
2. Ratings and parameters

2.1 Absolute maximum ratings:

Maximum Operating Voltage	5.5 V
Maximum current draw	<500mA
Storage Temperature	-40 °C … 100 °C
Operating Temperature	-40 °C … 85 °C

2.2 Mechanical:

Length	60mm
Width	25mm
Height (without cables)	10mm
Weight	10g
Mounting holes	4x M2

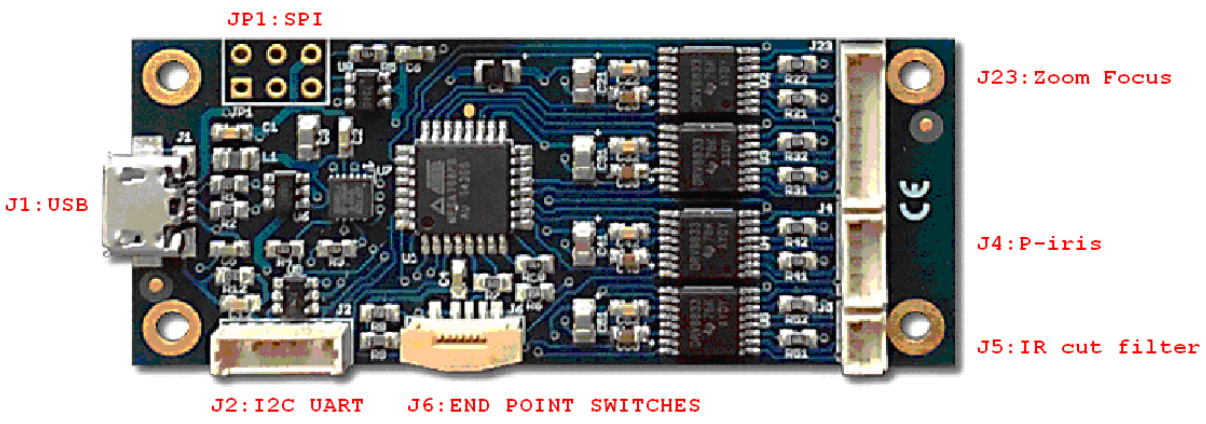


2.3 Operating parameters:

Operating Voltage	5.0V
Operating Current (while stepping)	<150mA
Operating Current (while idle)	<50mA

Note: Power consumption depends on many parameters of the operating mode.

3. Connectors



3.1. USB (J1)

Micro USB connector able to supply and control the board via a single cable. This connector interfaces with a FTDI®* FT200 USB to I2C bridge chip and uses drivers available for that chip. That way this interface appears as a serial port under some major operating systems. General compatibility is USB 2.0 but please refer to the datasheet of that chip to get full information about its features and compatibility.

The I2C side of that bridge chip is also connected to the I2C lines accessible at connector J2. Configuration options accessible through software need to be used to decide which interface (USB or I2C) to use. By default, the USB interface is active.

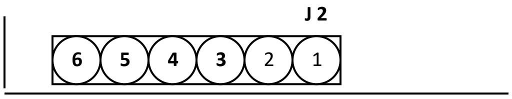
The USB interface is protected by a Texas Instruments®** TPD3S014-Q1 which limits the supply current and provides ESD protection. Please refer to the datasheet of that chip for the details. Additional protection and filtering comes from passive elements, the FT200 and the controller chip.

*/ FTDI is a trademark of Future Technology Devices International Ltd, Glasgow, United Kingdom
**/ Texas Instruments is a trademark of Texas Instruments Incorporated, Dallas, Texas, USA

3.2. I2C (J2)

J2 is a Molex®* 53047-0610 or compatible connector. The fitting cable connector might be the Molex®* 51021-0600 or compatible.

J2 has a total of 6 circuits (pins) of which the I2C interface uses two for its signals. To function the board would additionally need a ground reference (GND) and a 5V power supply. Both are available on J2 as well.



3 – I2C SCL
4 – I2C SDA
5 – Power GND
6 – Power +5 V

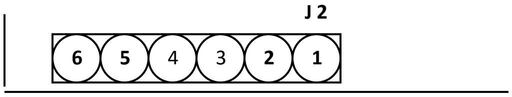
The I2C signals have 4.7 kOhm pull-up resistors on board. Speeds up to 400 kBaud are supported. ESD protection is provided to these signals by Littelfuse®** SP724AHTG clamping diode arrays. Please see its datasheet for details. Similar boards with I2C voltage level translation to 1.8V or 3.3V are available.

*/ Molex is a trademark of Molex LLC, USA
**/ Littelfuse is a trademark of Littelfuse Inc., USA

3.3. UART (J2)

J2 is a Molex®* 53047-0610 or compatible connector. The fitting cable connector might be the Molex®* 51021-0600 or compatible.

J2 has a total of 6 circuits (pins) of which the UART interface uses two for its signals. To function the board would additionally need a ground reference (GND) and a 5V power supply. Both are available on J2 as well.



1 – UART TxD
2 – UART RxD
5 – Power GND
6 – Power +5 V

The UART signals run at (5V CMOS) TTL levels at 19200 Baud and are 8N1 encoded. Please ask if you need that to be changed to something else. Similar boards with RS232 signal levels are available.

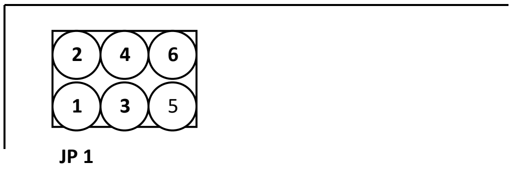
ESD protection is provided to these signals by Littelfuse®** SP724AHTG clamping diode arrays. Please see its datasheet for details.

*/ Molex is a trademark of Molex LLC, USA
**/ Littelfuse is a trademark of Littelfuse Inc., USA

3.4. SPI (JP1)

JP1 is 3x2 pin header with 2.54 mm (0.1”) spacing. This header is normally used for the initial firmware upload to the controller, but this happens through an SPI bus which could be made available to clients as a means to communicate with the board. At the moment that support is not yet implemented. Please ask if it is needed.

JP1 has a total of 6 circuits (pins) of which the SPI interface uses three for its signals. To function the board would additionally need a ground reference (GND) and a 5V power supply. Both are available on JP1 as well.



1 – SPI MISO
2 – Power +5 V
3 – SPI SCK
4 – SPI MOSI
5 – Reset
6 – Power GND

ESD protection is provided to these signals by Littelfuse®* SP724AHTG clamping diode arrays. Please see its datasheet for details.

3.5. Reset (JP1)

Pin 5 of JP1 (see above) is an active-low reset input to the board. It has a pull-up resistor of 4.7 kOhm. ESD protection is provided to it by a Littelfuse®* SP724AHTG clamping diode array. Please see its datasheet for details.

*/ Littelfuse is a trademark of Littelfuse Inc., USA

3.6. General Purpose I/O

Some Pins on some connectors could be reprogrammed into GPIO if certain interfaces are not needed. They could then be used to drive certain periphery like LEDs or read input. Please ask if such special features are needed.

3.7. Lens Connectors (J23, J4, J5, J6)

These connectors are intended for cables coming from the lens. More details can be found in the lens datasheet.

J23	Zoom and Focus stepper motor coil connections
J4	P-iris stepper motor coil connections
J5	IR cut filter coil connections
J6	Zoom and Focus end point switches

Those functions work independently from each other. So for instance manual lenses with a P-iris may also be interfaced.

The end point switches are optional. The board needs a software setting to work with them – and would fail to work properly with that setting if the switches were missing.

3.7. Power Supply (J1, JP1, J2)

As mentioned above in the sections covering the connectors J1, JP1 and J2 there are several ways to supply power to the board. It is important to note that all the power pins of those connectors are directly connected – all three GND pins are connected and all three +5 Volt pins are connected (note that this includes USB).

DO NOT use more than one power supply at the same time because there are always small differences in their voltages and high cross currents would most probably run through the board – leading to energy loss, heat and may even end up in overheating and ignition.

The board is also not intended to be used for power distribution. It may be tempting to receive the power via USB and then serve it to periphery connected to one of the connectors mentioned here. However we discourage to use such a topology. The user would be responsible to check the current capability of the connectors and monitor the heat of the PCB traces under worst case conditions in order to verify how much energy can be distributed safely that way. He would also have to make the correct setting to the register telling the USB bus what the maximum power requirement is – adding the stepper motor current of the lens to that going to the additional periphery (see below). Depending on the device supplying energy to the USB bus that request may be denied if such a high current could not be supplied. That would disable the stepper motor operation.

The common +5 Volt supply line goes through a Texas Instruments®* TPD3S014-Q1 which limits the supply current to the board – both its maximum level and its rush-in rate on power-up. Please refer to the datasheet of that chip for the details.

*/ Texas Instruments is a trademark of Texas Instruments Incorporated, Dallas, Texas, USA

4. Software

4.1. Overview

A simple control software example is provided for testing and demonstration. That software is given "as is" in order to help with getting started and testing.

The user can further integrate communication with the board into your own program. In addition to basic stepping commands there are commands which set device parameters by writing values to a permanent memory on the board.

There is the firmware update mechanism. It may be activated during power-up and in that case a new firmware version can be uploaded, verified and saved to the device memory. Please contact us if there is any questions about firmware upgrades.

4.2. Different bus behaviors

With exception of the SPI bus the boards have already been used with different serial buses, namely USB, UART and I2C. In all these cases the commands to a board come from an external source and the board executes them and answers. But there are limitations depending on the type of bus used. The board may be unable to answer while a stepping command is being executed and/or the board may be unable to actually process any new command arriving while currently executing a stepping command. The later most notably disables the usability of the STOP command which would normally interrupt the execution of a current stepping sequence.

The current firmware only supports the USB bus. We will soon have a version that simultaneously supports all buses, but for now we will limit the description to USB and add the features of the other buses as soon as they are integrated.

The USB bus as it is currently handled on those boards include both limitations listed – unable to answer or process new commands while executing the current one. Commands arriving at such a time will remain in the 512 byte input buffer until they can be processed. That buffer may overflow so we recommend not to send new commands until execution of the previous command has finished. In the case of an overflow a partial command will most probably remain in the buffer. To clean this a byte containing "Carriage Return" (0x0D) needs to be sent.

If commands are being discarded (like a STOP command when no movement is currently executed) the board will not send the related confirmation. Since via USB any arriving command is being delayed until the end of any movement this will always happen so the STOP commands becomes useless and "invisible" via USB. Such behavior may also concern other commands.

4.3. Serial Protocol (version 5.2)

The current protocol is very simple. That makes it suitable for fast commands coming from low-power hardware.

Each command starts with its ID and ends with carriage return CR(0x0D).

The answers coming from the board also end with carriage return CR(0x0D).

4.3.1. Move Stepper Motor

Bytes:

Command ID	Motor ID	Steps (MSB)	Steps (LSB)	Start/Stop	Speed (MSB)	Speed (LSB)	CR

Command ID:	0x66	Rotate forward	count steps relative to current position
	0x62	Rotate backward	count steps relative to current position
	0x73	Rotate forward	count steps absolute from the left end point switch (zero) (only for motors 0x01 and 0x02 when switches enabled) (execution always starts with going back to the switch)
Motor ID:	0x01	Focus	
	0x02	Zoom	
	0x03	Iris	
	0x04	IR-cut filter	not a stepper motor (see explanations below)
Steps:	###	Number of steps to do:	(256 * MSB) + LSB For motors of type 0x01 this parameter works differently. Please see the explanations below.
Start / Stop:	0x01	Start	
	0x00	Stop	set the number of steps to zero for this (Stop will not work on USB – no parsing while moving)
Speed:	###	Steps per second:	(256 * MSB) + LSB **Must** be between min./max. speed of the motor setup. For motors of type 0x01 this parameter works differently. Please see the explanations below.
CR:	0x0D	End of Command marker	

Answer after the command was executed successfully:

Bytes:

Answer ID	Status	CR

Answer ID:	0x74	End of Rotation
Status:	0x00	Success
CR:	0x0D	End of Answer marker

Note: The step count and speed parameters work differently for motors configured as DC motors (motor type 0x01). Please read the details on the next page. Motor number four (**IR cut filter**) is set as a DC motor by default.

4.3.1.1. Motion Control for DC motors

There is a command to write the setup of each motor (see below). There are stepper and DC motors. DC motors are powered for [Number of steps] times of a pulse width. The pulse width is set by 1 / [Steps per second]. Setting [Steps per second] to 500 gives a measuring unit of 2 ms and then setting [Number of steps] to 300 gives a single drive of 600 ms. [Steps per second] **must** be between min./max. speed of the motor setup.

4.3.2. Read Firmware Version

Bytes:

Command ID	CR

Command ID: 0x76 Read Firmware Version

CR: 0x0D End of Command marker

Answer <u>after the command was executed</u> successfully:
Bytes:

Answer ID	Version 1	Version 2	Version 3	Version 4	Version 5	CR

Answer ID: 0x76 Firmware Version

Version [1..5]: ### Values identifying the installed firmware version

CR: 0x0D End of Answer marker

4.3.3. Read Serial Number of the board

Bytes:

Command ID	CR

Command ID: 0x79 Read Serial Number

CR: 0x0D End of Command marker

Answer <u>after the command was executed</u> successfully:
Bytes:

Answer ID	SN 1	SN 2	SN 3	SN 4	SN 5	SN 6	CR

Answer ID: 0x79 Serial Number

SN [1..6]: ### Values identifying the serial number of the board

CR: 0x0D End of Answer marker

4.3.4. Read Motor Setup

Bytes:

Command ID	Motor ID	CR

Command ID: 0x67 Read Motor Setup

Motor ID: 0x01 Focus
0x02 Zoom
0x03 Iris
0x04 IR-cut filter

CR: 0x0D End of Command marker

Answer <u>after the command was executed</u> successfully:
Bytes:

Answer ID	Motor ID	Motor type	Use left stop	Use right stop	Steps (MSB)	Steps (LSB)	Min speed (MSB)	Min speed (LSB)	Max speed (MSB)	Max speed (LSB)	CR

Answer ID:	0x67	Motor Setup	
Motor ID:	0x01	Focus	
	0x02	Zoom	
	0x03	Iris	
	0x04	IR-cut filter	
Motor Type:	0x00	Stepper motor	
	0x01	DC motor	
Use left stop:	0x00	No	
	0x01	Yes	If this is 0x00 command 0x73 will not work.
Use right stop:	0x00	No	
	~~0x01~~	~~Yes~~	Currently only the left end point switch is supported.
Steps:	###	Max. number of steps:	(256 * MSB) + LSB (The board does NOT keep track of the position)
Min. Speed:	###	Min. steps per second:	(256 * MSB) + LSB
Max. Speed:	###	Max. steps per second:	(256 * MSB) + LSB
CR:	0x0D	End of Answer marker	

Note: For an invalid motor ID the return will be:

0x67 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0xFF 0x0D

4.3.5. Write Motor Setup

Bytes:

Command ID	Motor ID	Motor type	Use left stop	Use right stop	Steps (MSB)	Steps (LSB)	Min speed (MSB)	Min speed (LSB)	Max speed (MSB)	Max speed (LSB)	CR

Command ID:	0x63	Write Motor Setup	
Motor ID:	0x01	Focus	
	0x02	Zoom	
	0x03	Iris	
	0x04	IR-cut filter	
Motor Type:	0x00	Stepper motor	use this for motors number one to three (Focus, Zoom, Iris)
	0x01	DC motor	use this for motor number four (IR cut filter)
Use left stop:	0x00	No	
	0x01	Yes	If this is 0x00 command 0x73 will not work.
Use right stop:	0x00	No	
	~~0x01~~	~~Yes~~	Currently only the left end point switch is supported.
Steps:	###	Max. number of steps:	(256 * MSB) + LSB (The board does NOT keep track of the position)

Min. Speed:	###	Min. steps per second:	(256 * MSB) + LSB
Max. Speed:	###	Max. steps per second:	(256 * MSB) + LSB
CR:	0x0D	End of Command marker	

Answer after the command was executed successfully:
Bytes:

Answer ID	Status	CR

Answer ID:	0x63	Motor Setup Written
Status:	0x00	Success
	0x01	Wrong Motor ID
CR:	0x0D	End of Answer marker