PIEZOMOTION

INSTRUCTION MANUAL

Rotary Piezoelectric Motor
Evaluation Kit

RBS Series Motor
with Factory-Fitted Encoder
and Piezo Motion Closed Loop Control

Discover **affordable precision** with piezoelectric **innovation**.

Table of Contents

1.0 Introduction

This user guide covers the software installation, operation, and control of Piezo Motion's rotary piezomotor, which has been supplied with an integrated encoder and driver assembly and programmed with Piezo Motion's proprietary closed loop control software.

The RBS Series of rotary piezo motors represents a quantum leap in design of small size high-performance DC motors. Injection-molded using extremely durable, but light weight engineered reinforced thermoplastics, the RBS series provide low cost with superior precision and ultrafast response/start-stop characteristics. Highly energy efficient, the RBS series consume zero power in hold position while still providing significant force. Available in a variety of configurations (including non-magnetic) the RBS series is the ideal choice for high volume demanding OEM applications where superior performance and economical unit cost are important factors.

The RBS series combine high performance and excellent quality with an affordable low cost. The main body of the piezo motor is molded using modern reinforced engineered thermoplastics.

The contents of this evaluation kit are intended to be used as an evaluation tool for engineers interested in learning more about the performance and operation of Piezo Motion's RBS series rotary piezoelectric motors.

The electronic driver PCB for the piezo motor is included in the kit, together with cables and a 12 VDC power supply.

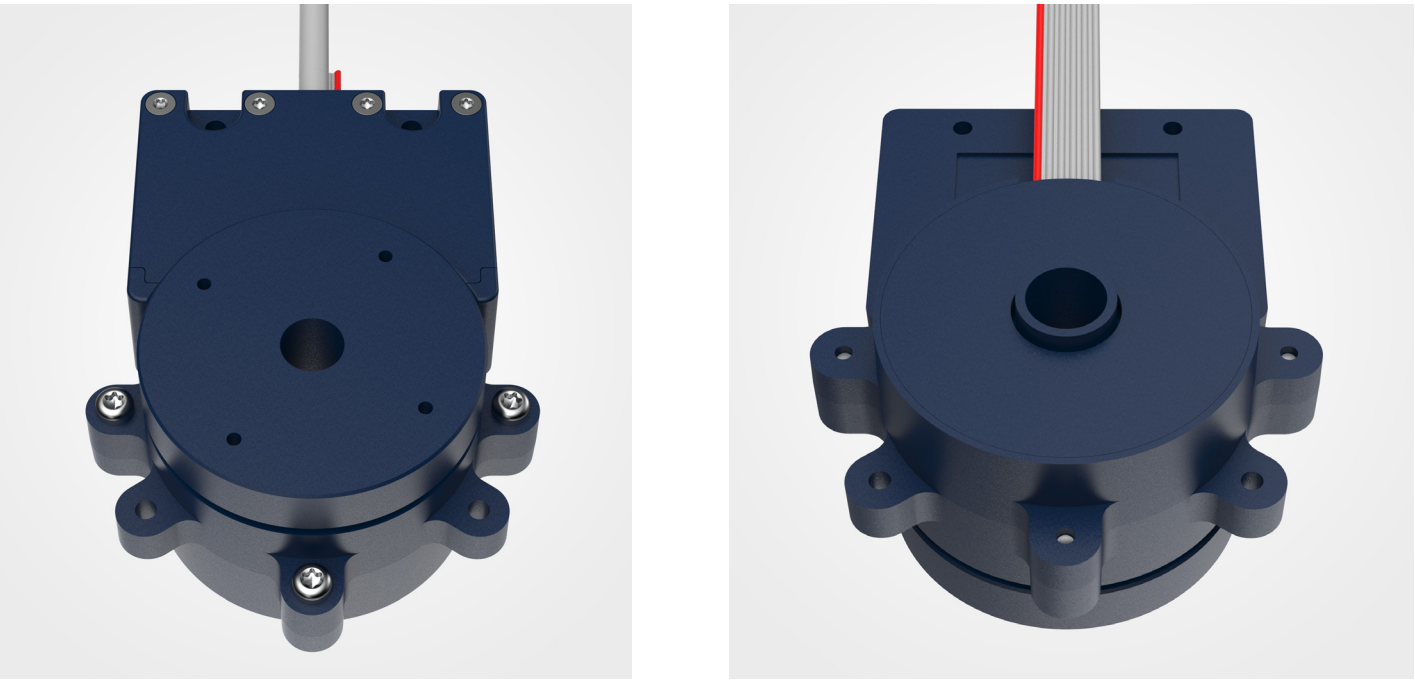


Figure 1. RBS Series Motors with Factory-Fitted Encoder - (Hollow Shaft, front & rear)

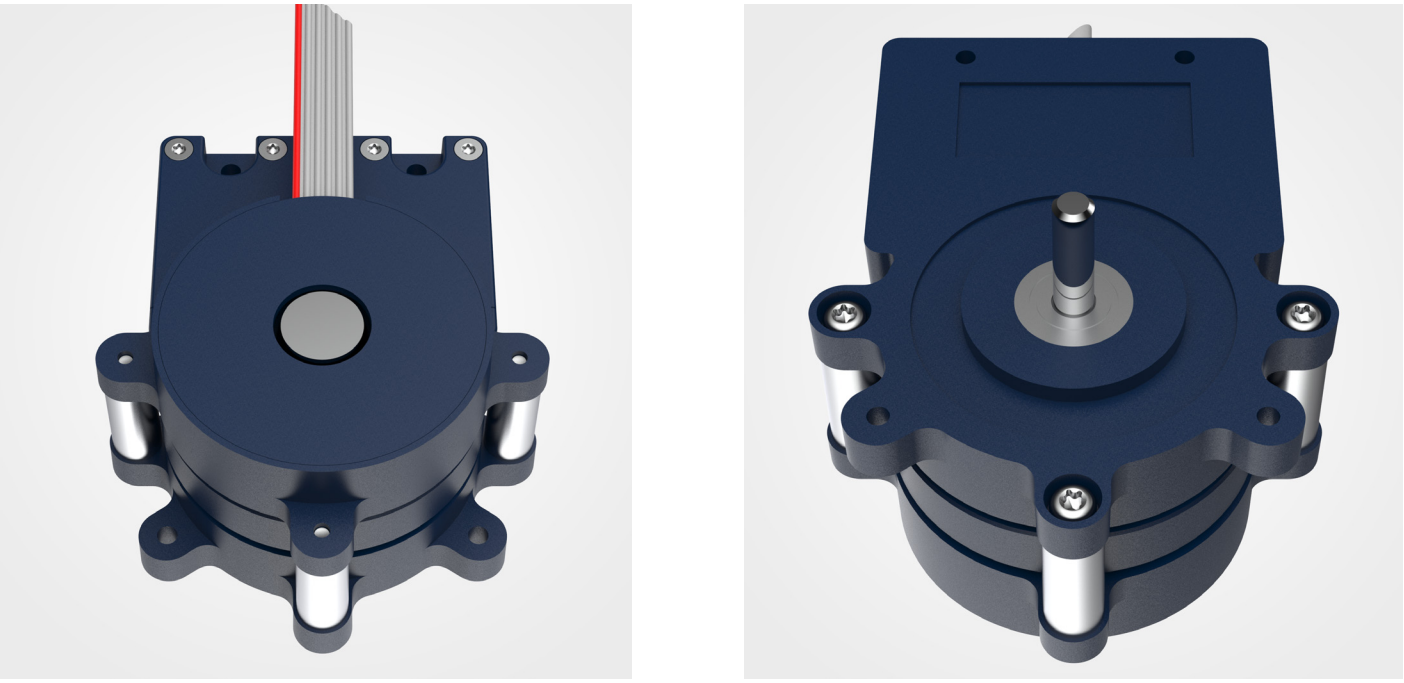
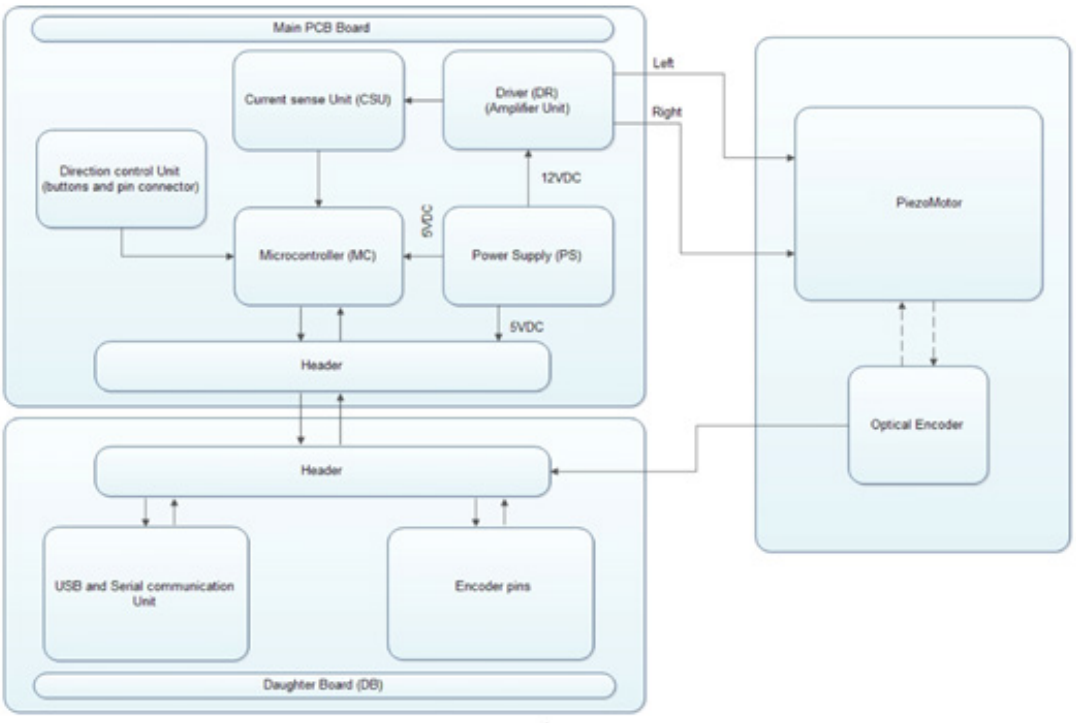


Figure 2. RBS Series Motors with Factory-Fitted Encoder - (Solid Shaft, front & rear)



Block Diagram of Driver Board (with optional daughter board and encoder)

The Direction Control Unit (DCU), includes manual (push button) directional control signal inputs to the microcontroller (MC) for continuous piezo motor operation. This is implemented through active TTL low inputs to the microcontroller. The Current Sense Unit (CSU) monitors current during motor operation.

The Microcontroller (MC), provides software-based control of motor motion in response to directional control inputs. When directional control signals are received, the microcontroller generates enable control output signals proportional to the control signals, and current feedback (via CSU) to the Driver (DR). In PWM mode of operation, the pulse width of the driver enable signal determines the amplitude of motion. A current negative feedback input is used by software to determine the optimal excitation frequency of the piezo motor to maintain the required current.

The Driver element (DR) is comprised of two gate driver ICs with FETs (to provide drive current) and step up voltage transformers. The Driver section uses the supply 12 V DC. The enabled gate driver amplifies the 5V TTL phase signals to a 12 V gate drive signal that switches on the FETs. When the FETs are active, the transformer steps up the ultrasonic signal voltage to the level required for excitation of the piezo motor (which can be between 30 V to 120 V depending on piezo motor model). Channel drive current is also detected within the Driver element, where it is amplified then integrated to provide an analog signal proportional to the channel drive current. This current sense feedback is used to optimize motor control and performance. Activation of motor motion in a specific direction is performed by command from the microcontroller.

2.0 Unpacking and Preparation

After carefully unpacking the piezo motor verify that all parts are present:

- Piezo motor with integrated encoder assembly
- Installation and control software
- Electronic driver PCB with encoder daughter board
- Interconnect cables (incl. USB cable)
- Power Supply - 12V DC

3.0 Mechanical Drawings of RBS Series

3.1 Specification for RBS Series Rotary motors

Weight	76 g
Dimensions	66x52x31 mm
Minimum Controlled Linear	196.0 µrad
Nominal Outside Diameter	32 mm
Minimum Controlled Angular Step	196.0 µrad
Uni-directional Repeatability	±1 arc-min
Torque	>30 mN.m
Rotational Speed	>100 rpm
Driver Supply Voltage	12 V
Current Options	360 mA | 360 mA
Driver Output Voltage	60 V
Response Time Range	30 µs to 50 µs
Minimum Angular Step	< 10 µrad
Angular Hysteresis at Direction Change	< 10 arc.min
Operating Temperature Range	-20 to 80°C
Shaft Type	Solid Shaft
Shaft Length	15 mm
Shaft Style	Plain
Connection Type	Molex connector
Wire Length	30 cm
Encoder	Optical

Table 1 – RBS Series specifications

4.0 Electronic Driver Overview

Piezo Motion's electronic driver PCB has been designed to provide an economical user-control interface compatible with all Piezo Motion piezo motors. Each driver PCB is supplied pre-programmed for the specific motor model and is software configurable to provide optimization of drive signals and integrated controls. The primary purpose of the driver PCB is the formation of electrical pulses with specific frequency and amplitude for excitation of the piezo motor.

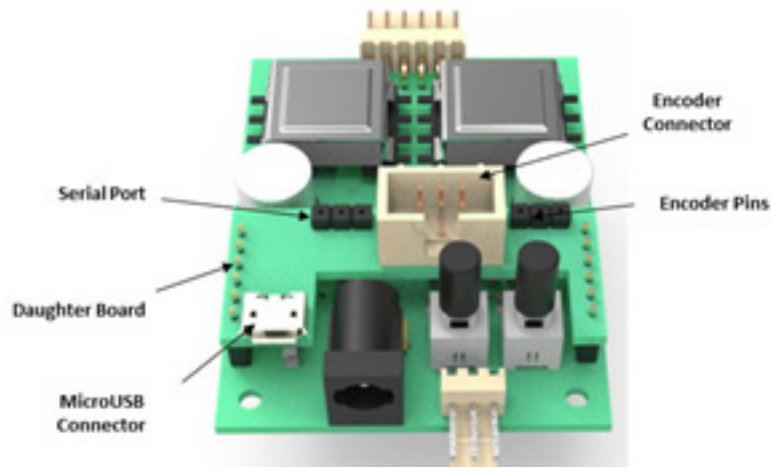


Figure 3. Driver PCB with installed daughter board for closed loop control

4.1 Main Driver PCB

Important note:

The RBS motor with integrated encoder and driver-daughter board assembly has been pre-programmed to operate in closed loop mode only using proprietary Piezo Motion control software. Manual control of motor motion can be performed by pressing either of the two Manual Control Buttons (if present) located on the driver PCB.

The electronic driver PCB enables precision motion control of the piezo motor via a microcontroller based 12 V DC digital system, which also allows for user generated inputs for motion control.

The driver assembly (Main PCB) is comprised of five main sections as shown in the block diagram. The Power Supply (PS), accepts a 12 V DC input through a DC power jack with a 2.0 mm center positive pin. The 12 V is filtered then regulated to 5 V DC and filtered again to provide the board operational voltage.

4.2 Motion Control Closed-Loop (Feedback Control) using Proprietary Piezo Motion software

In closed-loop control (feedback control) mode, an additional daughter PCB is mounted on driver PCB. Feedback from an external optical encoder, mounted on the piezo motor, is fed to the daughter board and used to close the loop. The position and speed of the motor can then be controlled through an elaborate set of commands via either a USB port (through Piezo Motion's GUI) or serial (RS 232) port commands.

The daughter board performs two key functions. Firstly, it enables the communication between the optical encoder installed on piezo motor and the driver PCB microprocessor, which provides for precision rotary or rotational closed loop control. Secondly, the daughter board's communications unit allows piezo motor motion control via external devices using either USB or Serial Port (RS 232) interface.

During installation of the daughter board, the microcontroller is factory-programmed with proprietary encoder motion control algorithms. Once the daughter board is installed the driver PCB will no longer work as a standalone driver. However, manual control of motor movement can still be achieved by pushing the Manual Control Buttons of the driver PCB.

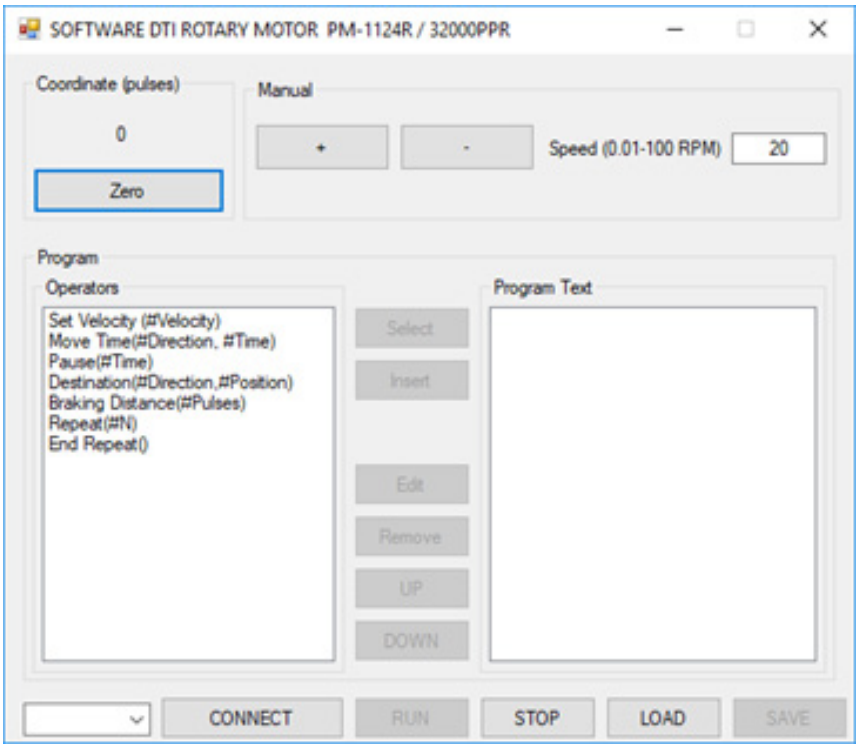
The rotary encoder has a resolution of 196 µrad (32,000 PPR) after interpolation and quadrature detection.

Two output signals from the encoder (channel A and channel B, with phase difference of 90°) can be monitored on the pins of the Encoder Output connector.

4.3 PLC & Serial Port Control Overview

Note: API control of the motor using Python is available as well and it is described in Section 6

Pre-programmed motion control algorithms enable implementation of several operators/commands for specific motion control. The key commands are for defining of speed (“Set Velocity”) and for movement to a defined position (“Destination”). These commands are resident within a library which can be accessed using either Piezo Motion’s optional programmable logic control (PLC) software, or via the serial (RS-232) port, or via Piezo Motion’s Python API library (Section 6).



The algorithms, which have been optimized based on the specific dynamic characteristics of the motor, analyze encoder feedback signaling and perform real-time noise filtration and temporal processing. This provides the following advantages:

- Increase in the range of speed control range during continuous mode operation.
- Reduction in ultrasonic vibrational noise during speed control.
- Substantial increase in the accuracy of speed control (speed stabilization) with external load changes.
- Dramatic increase in system response time during speed stabilization.

Set Velocity Command

Four different algorithms have been designed to control piezo motor speed, a brief description of these is provided here:

Continuous-frequency algorithm – Medium to high speeds are regulated by varying the excitation frequency along the resonance characteristic of the piezo motor within its medium-frequency region during medium to high speed motion.

Hysteresis algorithm – Low speeds are regulated by varying the excitation frequency along the resonance characteristic of the piezo motor within its high-frequency region during lower speed motion.

Modulation algorithm – Slow speeds are regulated by the on formation of train excitation packets with specific fixed repetition rates. The packets are internally frequency modulated during slow speed motion.

Frequency modulation algorithm – Very slow speeds are regulated by the formation of train excitation packets (similar to modulation algorithm) but with a varying repetition rate during very slow speed motion.

An example of the four different speed control algorithms for a rotary piezo motor is provided here.

- Medium to high speed: 100-2 rpm (1 rpm speed increment)
- Low Speed: 2-1 rpm (0.1 rpm speed increment);
- Slow Speed: 1-0.2 rpm (0.1 rpm speed increment);
- Very Slow Speed: 0.2-0.01 rpm (0.01 rpm speed increment).

In this example, control over the entire range of speeds is performed using a root algorithm. The root algorithm is based on the interval principle, where each range of speed uses its own algorithm and its own starting frequency point. Depending on the set speed (which is specified by the “Velocity” command), the program selects the optimum algorithm to ensure stabilization of the required speed.

These algorithms, in contrast to traditional PWM algorithms, enable the user to significantly extend the range of speed control of the piezo motor, while simultaneously reducing the associated noise and parasitic ultrasonic vibration.

Set Destination Command

The main command responsible for movement and positioning is the “Destination” command. Movement using this command is specified in pulses from the encoder; where the resolution of the encoder is 196 µrad (40.5 arc.sec).

The underlying algorithm has been developed to optimize motion control by analyzing the required speed of movement and specific dynamic characteristics of the motor (i.e. the inertia of the rotor/carriage/loan and self-braking torque/force). The algorithm also analyses the braking distance when approaching the desired target coordinate.

Since the value of the external inertia load is initially unknown, the system is initially programmed for a fixed breaking distance and fixed speed of braking based on the follow assumptions:

- For rotary piezo motor RBS - moment of inertia of the rotor is Io = 70 g*cm2 and maximum programmed speed is ωo = 60 rpm;
- For a rotary piezo motor, taking into account the additional external inertial load I, the maximum programmed speed ω in a given coordinate can be calculated as:

$$\omega = \omega_o \sqrt{Io/(Io + I)}$$

Note: If the programmed speed exceeds the calculated above maximum speeds, the destination position can be overshot.

An example of how the Destination command implements these algorithms for a rotary piezo motor is provided here:

- The Destination command analyzes the programmed speed of rotation ω, which is set using the Set Velocity command. If the speed exceeds the fixed speed of braking, then it algorithm decelerates the motor in the span of certain number of encoder pulses (default value) in order to achieve the fixed speed of braking.
- If the speed of rotation ω (set by Set Velocity command) is equal to or less than the fixed speed of
- braking, then positioning is performed at the programmed speed ω.
- If the programmed speed of rotation ω is greater than the fixed speed of braking and the travel range (determined by Destination command) is smaller than the deceleration distance (default value), then the movement is carried out at the fixed speed of braking.

These positioning algorithms can significantly improve the accuracy of positioning and can bring the positioning resolution of movement to within the actual resolution of the encoder.

An additional Command “Braking Distance” can be implemented for custom braking distance (the default value is 500 pulses). In this case the command needs to be placed before the Destination command. The Destination command will then be based on the new custom value for the braking distance. If the custom value for the braking distance is set at zero (“Braking Distance(0)”) the piezo motor will work without any deceleration when approaching the target coordinate. In this case, the risk for overshooting increases.

Note: The user must choose the correct value for the braking distance in order to achieve a positioning resolution equal to the maximum achievable resolution of the encoder.

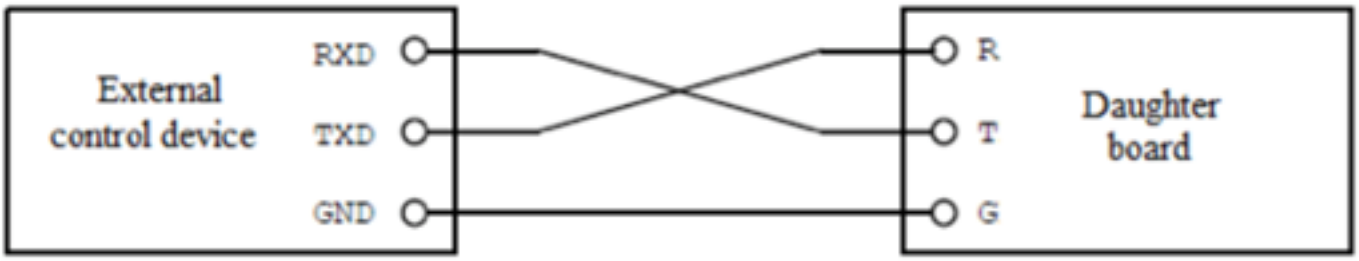
To implement motor control via the serial port, a 3-pin connector is used, which is located on the daughter board of the driver, where:

contact G - common (GND);

contact R - data reception (Receiver or RXD);

contact T - transmission (Transmitter or TXD).

The connection diagram of the control device to the Daughter board of the motor driver is shown below.



5.0 Mechanical Drawings of RBS Series

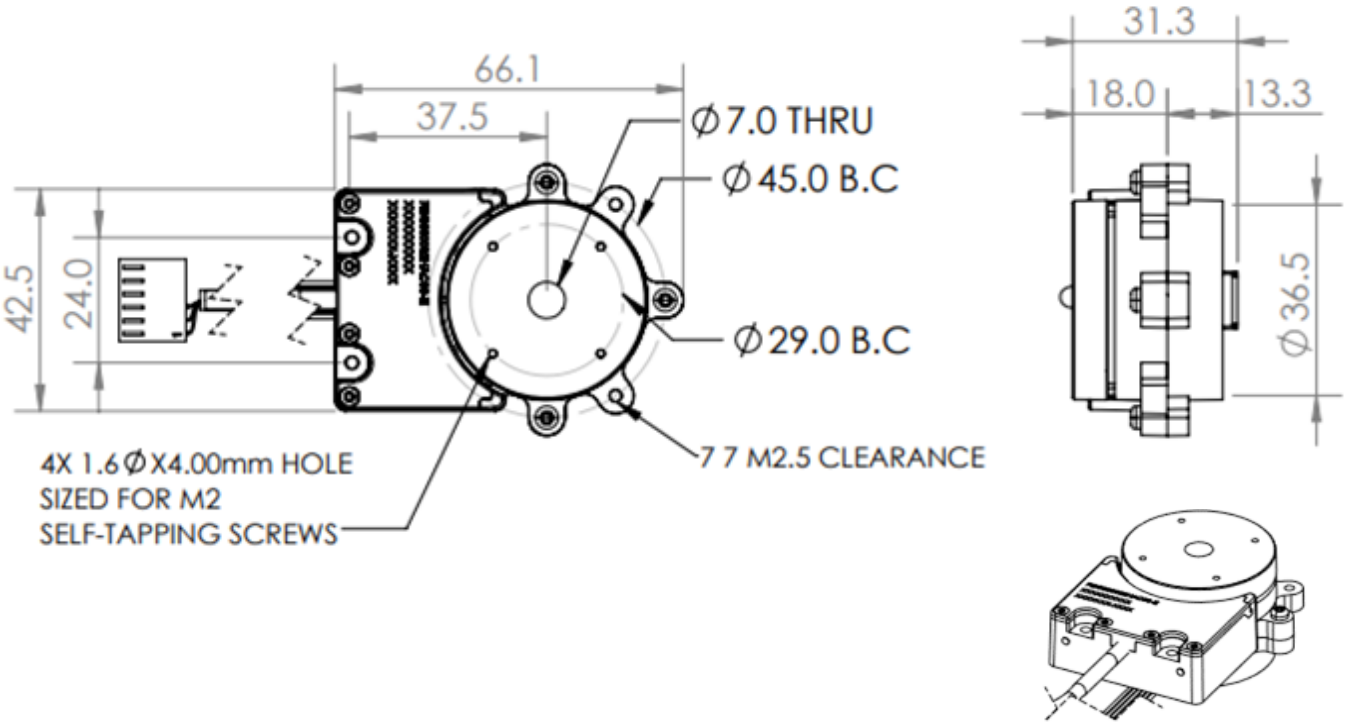


Figure 4. Mechanical Drawing (mm) of RBS Rotary Motor with Encoder

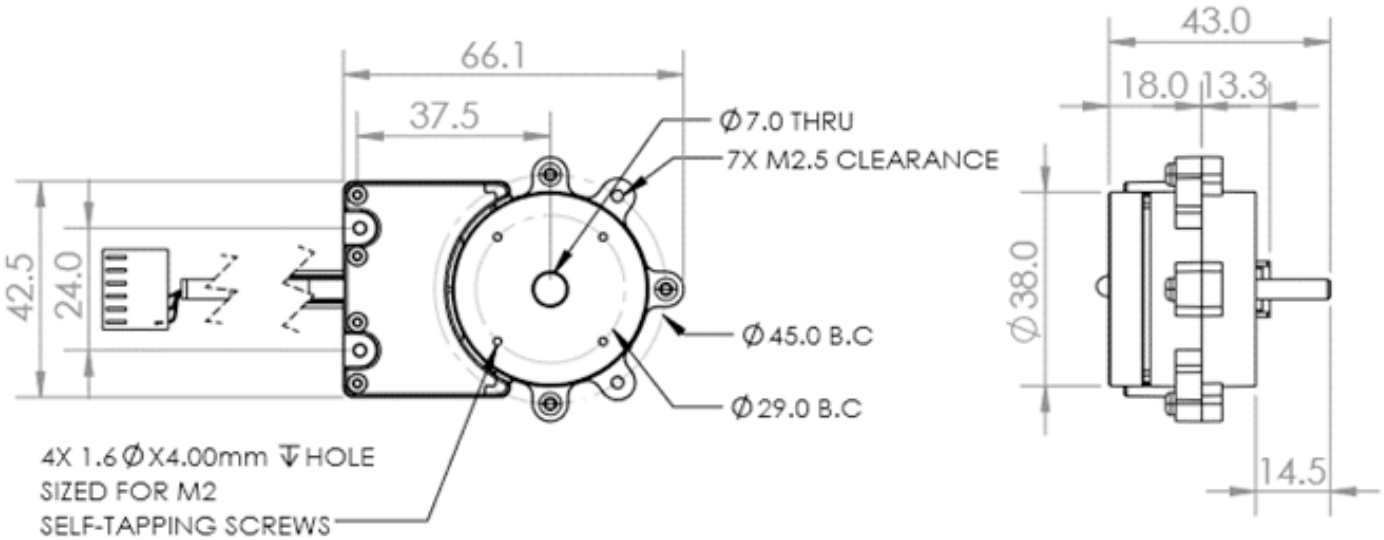


Figure 5. Mechanical Drawing (mm) of RBS Solid Shaft Rotary Motor with Encoder

6.0 Piezo motor system set up

6.1 Software Installation

Note: Before connecting the driver to a power supply, complete the software installation steps as described below.

i. Windows® PC with Windows 10 operating system is required

ii. Open the main folder “PIEZO MOTION ROTARY motor with encoder installation software V1”, located on the external storage device supplied with the motor and copy the folder into a location on your computer hard drive. This folder contains all installation programs.

iii. Open the executable file “GUI SOFTWARE PIEZO MOTION ROTARY MOTOR with ENCODER V1”. If an error occurs, you must install additional software, namely the .NET Framework version 4 or higher. To do this, run the executable file “dotNetFx40_Full_x86_x64”, which is in the root directory of the main folder or download the updated version of the .NET Framework from the official Microsoft website.

iv. If the file “ GUI SOFTWARE PIEZO MOTION ROTARY MOTOR with ENCODER V1” is successfully opened, open one of the folders labelled “CP210x_Windows_XP” (for Windows XP) or “CP210x_Windows_7_8_8.1_10” (for Windows 7,8,8,1,10), both folders are located in the root directory of the main folder.

v. Run the driver installation. To do this, run the file “CP210xVCPInstaller_x64” if you have a 64 bit OS or “CP210xVCPInstaller_x86” if you have a 32 bit OS. These files are located in the root directory of the corresponding folder for your OS.

vi. Connect the power supply to the motor board.

vii. Connect the motor board to the PC using the USB cable supplied.

viii. Open “Device Manager”, which is located in the “Control panel”.

ix. In the “Ports (COM and LPT)” section, the line “Silicon Labs CP210x USB to UART Bridge” should appear.

6.2 Connecting the Power Supply

Connect the 12 VDC Power Supply to the Power Supply Connector located on the electronic driver PCB. Connect the other end of the power supply cable to a wall power socket

6.3 Connecting the Driver Board and Encoder

The piezo motor connects to the driver board by a connector on the end of the motor wire. This connector mates with the corresponding connector on the electronic driver PCB. The connectors can only be joined in one possible orientation. Press the connector gently in place so that it is flush with the edges of the receptacle on the driver PCB.

Connect the encoder ribbon cable (which is attached to the encoder motor assembly) to the 6-PIN connector located on the top of the Daughter Board (RBS). Press the connector gently in place so that it is flush with the edges of the receptacle on the Daughter Board.

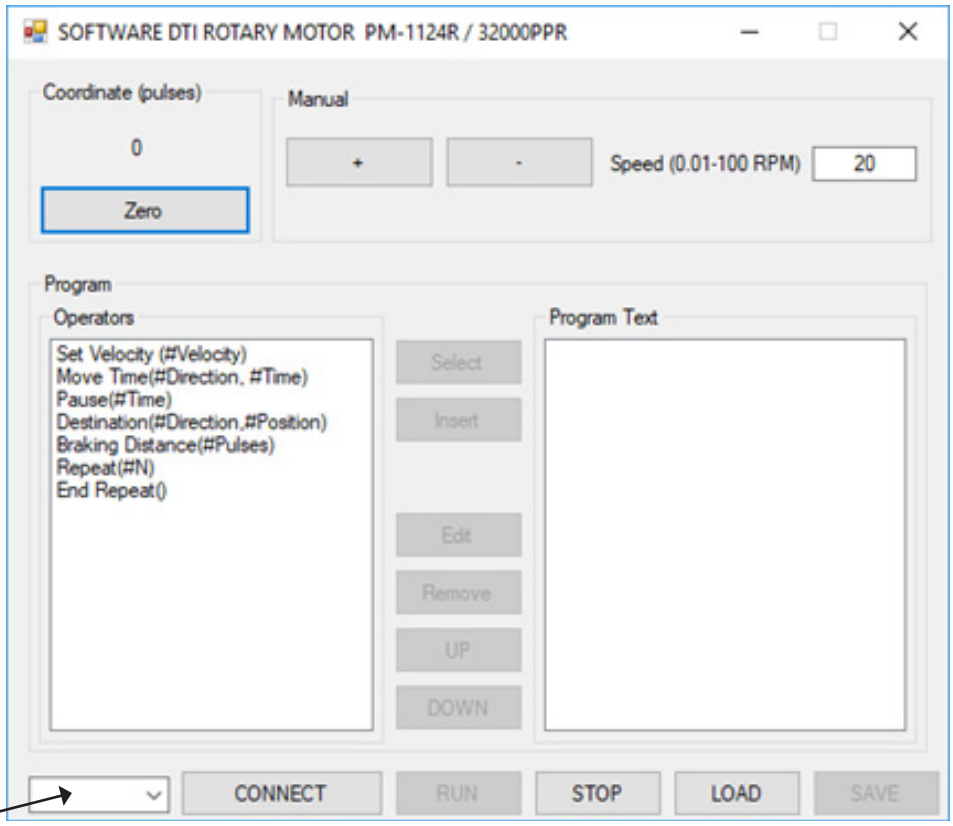
6.4 Software Operation

Open the file “GUI SOFTWARE PIEZO MOTION ROTARY MOTOR with ENCODER V1 “.

In the bottom left corner of the pop-up list, select the desired COM port (see figure) and press the “CONNECT” button, the name of the button will change to “DISCONNECT”.

NOTE: Make sure that the COM port is connected correctly by activating the motor motion with the “+” or “-” buttons from the “Manual” control section of the window.

The device is ready to use.



6.5 Software Description

The software program is presented as a standard Windows Application. The Main input window (as shown) is titled "SOFTWARE PIEZO MOTION ROTARY MOTOR PM-1124R / 32000PPR ". This window contains three main fields:

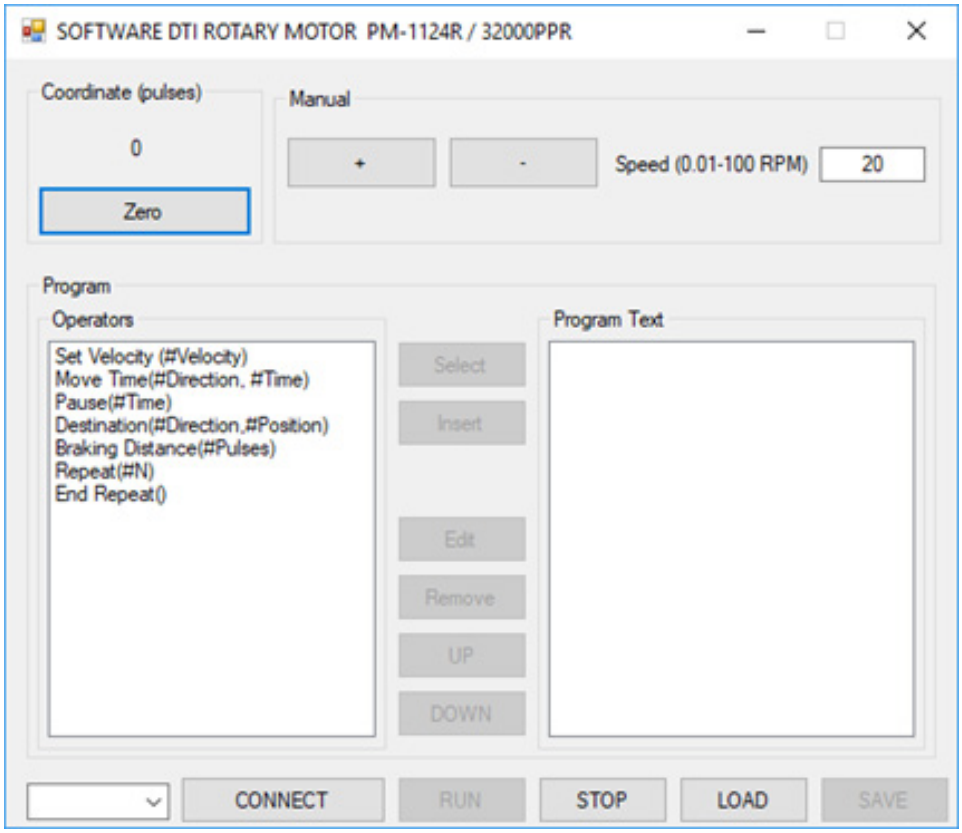
"Manual" - control field for manual operation activated by pressing the "**+**" or "-" button, which moves the motor in opposite directions. The speed of movement is entered in the "**Speed (0.01-100 RPM)**" field.

"**Coordinate (pulses)**" - field where the number of encoder pulses is displayed. The "**Zero**" button resets this field to zero.

NOTE: "Coordinate" field shows the travel coordinate only in manual mode. After the "Zero" button is pressed it shows the "absolute" coordinate in respect to the "zero" position. This window is not functional in "Program" mode.

"**Program**" - field for programming user set parameters.

Positioning of the motor is implemented by counting encoder pulses. For the rotary motor model RBS this is 32,000 PPR.



The "**Program**" panel is designed to create, view and edit the motor control program. It contains the subpanels - "**Operators**" and "**Program Text**" and the buttons for creating and editing programs - "**Select**"; "**Insert**"; "**Edit**"; "**Remove**"; "**UP**";"**DOWN**"

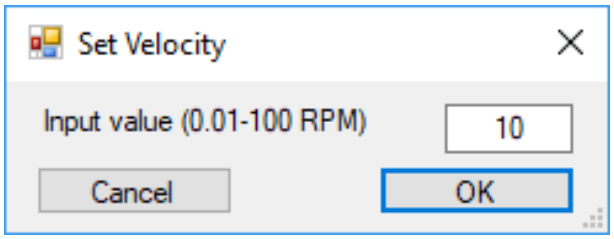
The functions of the subpanels and the buttons are discussed below.

"**Operators**" - contains all available action operators. These are "**Set Velocity (#Velocity)"; "*Move Time (#Direction, #Time)*"; "Pause (#Time); Destination (#Direction, #Position)"; "Repeat (#N); End Repeat ()", "Braking Distance (# Pulses)"** . The functions of these operators are described below.

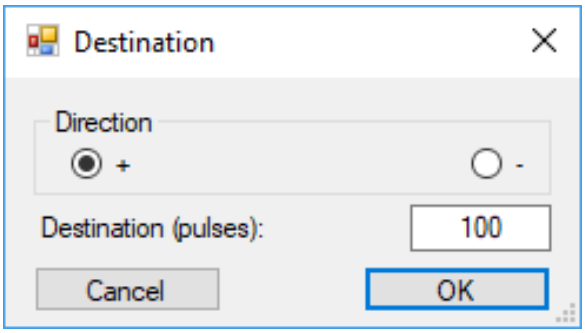
The user can select any operator by clicking on the "**Select**" button and add the operator to the "**Program Text**" field on the right. Depending on the selected operator, the user is prompted to enter the value(s) of the required parameters.

6.6 Operator Functions

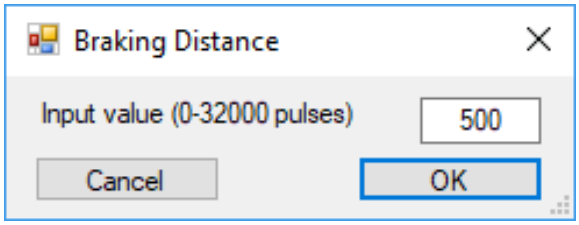
Set Velocity (#Velocity) – This operator is used for setting motor speed. When selected, a dialog box for entering speed value appears. The user needs to enter the required speed in the input value field (0.01-100 RPM).



Destination (# Direction, # Position) – This operator is used for moving to a specified coordinate. The user needs to specify the direction of movement (positive direction (+) or negative (-)). The absolute value of the travel in the selected direction is entered in the Destination field in pulses.

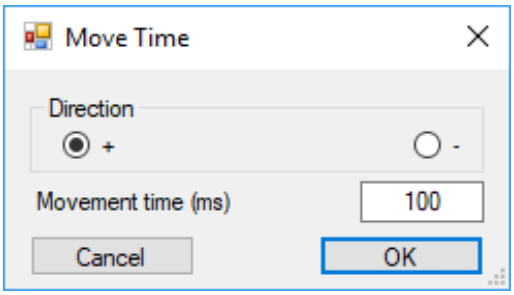


Braking Distance (# Pulses) - is used to program the breaking distance before reaching the target coordinate. Normally, it is placed before the "*Destination*" operator. The users need to enter the breaking distance (in pulses) in the operator window.

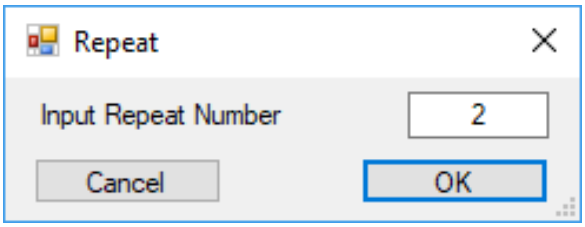


NOTE: "Braking Distance (# Pulses)" operator improves accuracy of positioning, when approaching target coordinate. It provides the flexibility of changing the braking distance, depending on the load. This function helps to eliminate unwanted effects, like target "overshooting" and "hunting" mode, which are typical for servo systems based on electromagnetic motors.

Move Time (#Direction, #Time) – This operator is used for programming a specified time of movement with speed defined by **Set Velocity** operator placed before this operator. The user needs to specify the direction of movement and enter the movement time in milliseconds in the field "**Movement time (ms)**").



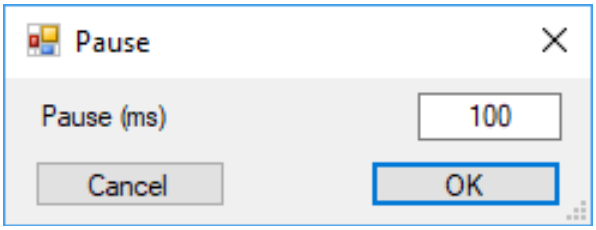
Repeat (#N) – This operator is used for loop/cycling. The command is inserted in the beginning of each loop/cycle. The user must enter the number of repetitions of the cycle by completing the "I**nput Repeat Number**" field.



Note: The maximum number of cycles allowed in the "Repeat (#N)" operator is 255. The proper use of this operator requires that operator "Pause (#Time);" with minimum duration of 10 ms is placed in the body of the cycle between the operators for movement and the end of the cycle. "Repeat (#N)" operator inside the cycle of another "Repeat (#N)" operator is not allowed.

End Repeat () – This operator used to end all operations. The command is placed at the end of all operators in the loop/cycle. It requires "**Repeat (#N)**" operator before it.

Pause (#Time) – This operator is used to determine the period (in ms) of any desired pause.



To execute/start the program, click the "**Run**" button. To stop the program, press the "**Stop**" button.

6.7 Program Edit Operators Functions

"Edit" - edits the corresponding operator of the program. Select an operator in the working program and press the "Edit" key;

“Remove” - deletes the corresponding operator. Select an operator in the working program and press the “Remove” key.

Note: Before loading a new program please remove all lines of the old program starting from the last one. Currently, there is not an option to clear the whole program.

“UP” – moves one position up the selected operator in the text of the program.

“DOWN” – moves one position down the selected operator in the text of the program.

“Insert” - introduction of a new operator in specific place of the program.

“Operators” panel; select the operator in the work program, above which the new operator will be inserted; press the “Insert” key; enter the corresponding parameters into the new operator window.

6.8 Saving and loading work programs

To save work program to a file, use the “SAVE” button. To open an already saved program, use the “LOAD” button.

Note: when writing a program please be aware of the following recommendations:

- Include a “Pause” operator (with a minimum time parameter of 10 ms) after each “Destination” operator. This will ensure appropriate synchronization between the implementation of consecutive operators.
- The operator “Destination” contains a complex algorithm for precise positioning of the motor, without overshooting. For proper implementation of this operator, the “Set Velocity” operator must first have a parameter defined at less than 60 RPM.

6.9 Factory installed Demonstration Test Program

A factory installed demonstration program called, “RBS Closed Loop Test Program” is included with the software. This program can be found in the folder called “Test Programs”. The demonstration program is designed to show the capabilities of the motor over a range of programmed settings.

To load the demonstration program, first follow the software installation steps outlined at the start of this Section.

On the main program operator screen, click the “LOAD” button and locate the program file called, “RBS Closed Loop Test Program” from the pop-up menu that appears. This program can be found in the folder called “Test Programs”.

Click “Open” to load the demonstration program. The program commands should appear on the righthand side of the operator window under “Program Text”.

Click “RUN” to execute the program. The motor will start to move until the program is finished, the software will then indicate, “The Program is Executed”.

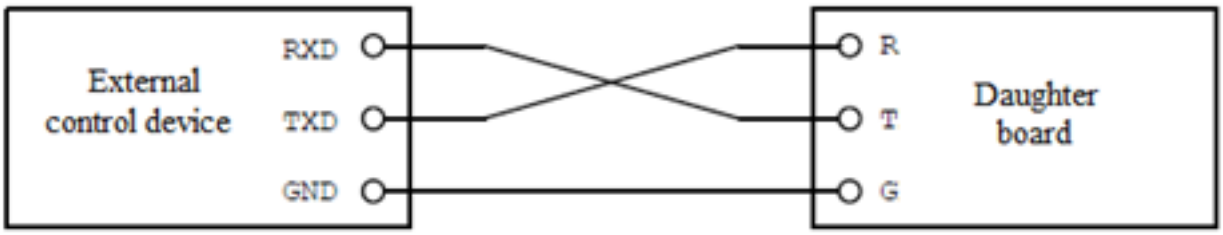
7.0 Controlling the motor with commands through serial port

7.1 Connection

To implement the motor control via the serial port, a 3-pin connector is used, which is located on the daughter board of the driver, where:

contact G - common (GND);
contact R - data reception (Receiver or RXD);
contact T - transmission (Transmitter or TXD).

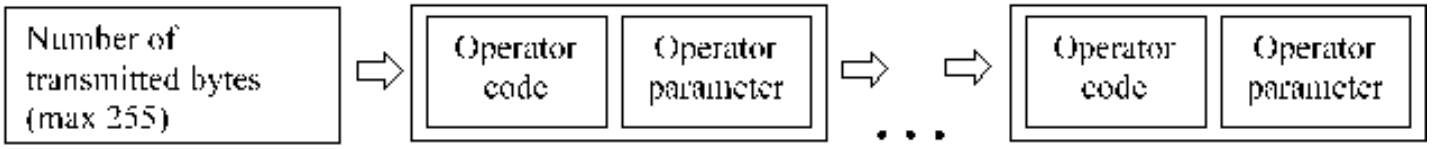
The connection diagram of the control device to the Daughter board of the motor driver is shown below.



7.2 Packaging of transmitted data for control

To start a transfer of control package to motor driver, the code “5” needs to be sent, which prepares the microprocessor for receiving the package. Each package includes three main fragments:

- number of transmitted bytes;
- code of the operator;
- parameters of the operator.



After executing all commands, the microprocessor sends back the code “5” (which indicates the completion of all commands), as well as the value of the number of counted pulses (4 bytes), when the last command is executed.

In order to stop the execution of the current command, code “10” must be sent. This code stops the implementation of the program.

7.3 Instruction Set

“Move Time(#Direction, #Time)”

- Command number: 1.

Parameter size: 1byte.

- Parameter “Direction” sets the direction of movement.

Parameter size: 1byte.

When “Direction” equals: “1” - move to the right; any number in the range “2-255” move to the left.

- Parameter “Time” sets motor running time in milliseconds.

Parameter size: 4bytes.

Range of admissible values: [1...2^{32}].

“Pause(#Time)”

- Command number: 2.

Parameter size: 1byte.

- Parameter “Pause” sets motor idle time in milliseconds.

Parameter size: 4bytes.

Range of admissible values: [1...2^{32}].

“Set Velocity (#Velocity)”

- Command number: 3.

Parameter size: 1byte.

- Parameter “Set Velocity” sets motor speed.

Parameter size: 2bytes.

Calculation of the parameter:

Velocity= 100*Speed [RPM];

Range of admissible values: [1...10000].

“Continuous movement(#Direction)”

- Command number: 4.

Parameter size: 1byte.

- Parameter “Direction” sets the direction of movement.

Parameter size: 1byte.

When “Direction” equals: “1” - move to the right; any number in the range “2-255” move to the left.

“Stop”

- Command number: 5.
- Parameter size: 1byte.

After this command is implemented, the piezo motor stops and the microprocessor sends back through the serial port number “5” and immediately after that the value of the pulse counter (4 bytes). The command “Stop” is used together with the command “***Continuous movement***”.

“Destination(#Direction, #Position)”

- Command number: 6.

Parameter size: 1byte.

- Parameter “Direction” sets the direction of movement.

Parameter size: 1byte.
When “Direction” equals: “1” - move to the right; any number in the range “2-255” move to the left.

- Parameter “Position” sets number of encoder pulses.

Parameter size: 3bytes.
Range of admissible values: [1...2^{24}].

“Home(#Direction)”
Note: The operator “Home” can be used only if there is a mechanical limit/stop installed in the system.

- Command number: 11.
- Parameter “Direction” sets the direction of movement.
- Parameter size: 1byte.

When “Direction” equals: “1” - move to the right; any number move to the left.

“Braking Distance (#Pulses)”

- Command number: 9.
- Parameter “Pulses” sets number of encoder pulses.
- Parameter size: 2bytes.

Range of admissible values: [1...2^{16}].

5.4 Serial Port configuration

-baud rate: 9600
-data bits: 8
-parity: none
-stop bits: 1.

Example

Implementing command "***Destination***" 500 to the right

5→5→6→1→244→1→0 (all decimal, arrows are not transmitted)

5 - start of transmission;
next 5 - number of bytes transferred;
6 - command number (in this case - 6 for "***Destination***" command) ;
1 - first parameter of "***Destination***" command - direction: 1 is for right direction;
next 3 bytes - number of pulses, low significant bytes forward (the pulses number must be divided into 3 bytes. In this case 500 = 0 1 244.

Note: It is necessary to use a terminal that DOES NOT convert the entered values using ASCII table. An example of such terminal is when the user enters number 5 in the terminal, the program sends 53 (according to the ASCII table). A terminal, which sends number 5 unchanged should be used.

8.0 Controlling the motor with commands through Python

The API and examples are available on our GitHub.com account - piezomotion.com/w98j

9.0 Recommended settings to avoid overheating

Piezo Motion's range of piezomotors are designed for precise control applications using a duty cycle. They are not designed for prolonged operation in Continuous (non-stepping) Mode, which can lead to overheating of the motor and possible internal damage not protected under warranty.
To avoid overheating of the motor please follow the guidelines in the table below and ensure that motion control settings for Continuous Mode and/or Stepping (PWM) Mode are within the limits specified in the table below.
For applications requirements exceeding the recommended guideline, please contact Piezo Motion's Technical Support.

Model/Series #	Max. Operating Power (W)	Max. Operating Power (W)	Recommended PWM Duty Cycle	Maximum Duration in Continuous Mode
RBS Series	4.32 W	1.8W	40%	30s
LBS004	4.2 W	1.7W	40%	20s
LCS004	4.2 W	1.7W	40%	20s
LCS010 (10N)	19.2 W	5.8W	30%	15s
RAS(300mA)	1.5 W	0.5 W	30%	10 sec.
RAS(100mA)	0.5 W	0.25 W	50%	30 sec.
RAS(50mA)	0.25 W	0.13 W	50%	60 sec.
LAS(100mA)	0.5 W	0.25 W	50%	30 sec.

Table 2 – Recommended Guidelines for Motion Control.

10.0 Appendix

10.1 Determination of Windows®Operation System

To determine if your computer is running a 32-Bit or 64-Bit version of Windows, in Windows XP, you need to:

1.Open the Start menu.
2. Right click on “My Computer” and select “Properties”.
3. Select the “General” tab. If the computer runs under 64-Bit OS it will be stated under the line stating Microsoft Windows XP. If a number is not specified but only the Windows edition is mentioned, eg. Version 2002 Service pack 3, then your computer runs on 32-Bit OS.

To determine if your computer is running a 32-Bit or 64-Bit version of Windows, in Windows 7,8,10, you need to:
1. Open the Start menu.
2. Under “Settings” click “About”.
3. The information will be listed in “Device specifications”.

11.0 Technical Support

Technical support is available from 9 AM to 5.30 PM U.S. Eastern Time. Please refer to contact information at end of manual.

12.0 Warranty

All sales and deliveries are made exclusively on the basis of our general Terms and Conditions of Business. These are available to view and download on the Piezo Motion homepage at http://piezomotion.com/terms-and-conditions/

Piezo Motion Corp
6700 Professional Parkway
Sarasota, Florida 34240
USA

Tel: (941) 907- 4444
Email: info@piezomotion.com
piezomotion.com

PIEZOMOTION