RHINO MOTION CONTROLS

RMCS-220X

High-Torque Encoder DC Servo Motor and Driver

UART, I2C, PPM and Analog input interface (Max. 15Vdc and 7A)

RHINO

Installation Manual and Datasheet

Key Features

- Zero-Backlash DC Servo Motor Performance with 0.2deg encoder resolution
- 10RPM,60RPM,100RPM,200RPM,300RPM,600RPM,900RPM High-Torque DC Servo Motor
- Metal Gearbox and Gears with 18000RPM base motor
- 0.2deg resolution Quadrature Optical Encoder on output shaft
- High-Current DC Servo motor driver integrated with the motor
- Absolute (32bit) Motor position control interface via UART, I2C, PPM signal and analog input
- Accurate Motor speed control interface via UART, I2C, PPM signal and analog input
- Industrial Grade Aluminum housing for motor and drive
- Compatible with NEMA 23 mounting setup
- Speed/Position can be controlled using a terminal or MCU via simple UART/I2C commands
- I2C master device can control multiple RMCS-220x via simple I2C command structures
- An RC receiver or any PPM source can directly control the speed of the motor
- An analog signal or fixed analog voltage from a potentiometer can directly control the speed of the motor
- Max-speed, Damping, P-Gain, I-Gain and Speed Feedback settings are adjustable

Description

Thank you for purchasing RMCS-220X, High-Torque Encoder DC Servo Motor and Driver. RMCS-220X is Rhino Motion Controls introductory Encoder DC Servo motor control solution designed for easy installation and operation with multiple different interfaces.

The RMCS-220X integrates a High-Torque DC motor with 18000RPM base motor and Metal Gearbox and Gears for 10RPM, 60RPM, 100RPM, 200RPM, 300RPM, 600RPM and 900RPM options. It houses a 0.2deg resolution quadrature optical encoder on its output shaft that allows for superior position and speed control with zero back-lash at the output.

The RMCS-220X offers speed and position control via UART, I2C, PPM input signal and a simple analog voltage input.

Technical Specifications

Specification	Min	Max	Units	Comments
Supply Voltage	11	15	Volts DC	Between V+ and GND
Current	0.5	7	Amps	No-load to stalled condition
Input Signal High Voltage	4	6	Volts DC	With respect to GND
Input Signal Low Voltage	0	1	Volts DC	With respect to GND
UART Baud Rate	—	9600	bps	For UART interface
I2C clock freq.	10	200	kHz	For I2C interface
Ambient Temp.	0	70	Celsius	Operating Temperature
Humidity	0	95%		Non condensing
Analog Input Voltage	0	5	Volts DC	For Analog Voltage interface
PPM Pulse Width	600	2400	usec	For PPM signal interface

Mechanical Specifications

Specification	Details
Dimensions (L * W * H)	120mm * 60mm *65mm
Weight	350gms

Encoder Specifications

Specification	Details
Counts per Rotation	1800 counts
Degrees per count on output shaft	0.2deg per count

Caution

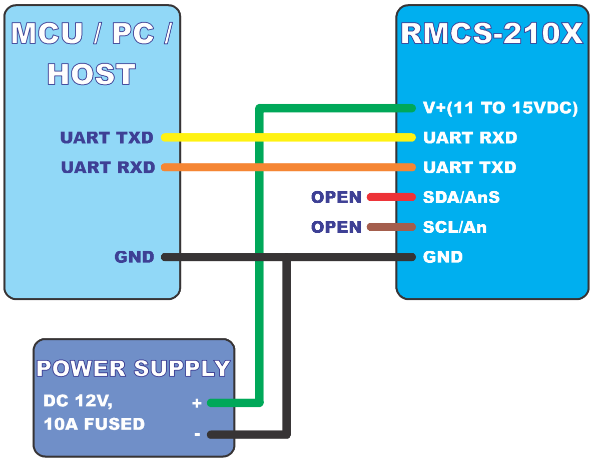
- Read this document carefully before installing and using this product
- Inputs voltage to the drive must not exceed the maximum of 15VDC or it may damage the drive
- Reversing polarity power supplied to the drive will damage the drive or power supply
- Excess humidity or condensation on the drive may damage the drive
- Voltage in excess of 7V on the input terminals may damage the speed controller
- Reverse voltage in excess of 7V between the input terminals may damage the controller
- Keep the motor and drive in a ventilated or cool temp.
- Make sure the supply is well regulated and there is minimal voltage ripple
- Disconnection of PPM, UART, I2C or analog interface while the motor is in motion will not stop the motor. It will continue to rotate at the last speed specified by either interface

Power and Input Terminal Assignments

Terminal No.	Terminal Name	Wire Color	Description
Terminal 1	GND	BLACK	Ground should be connected to negative of supply of battery
Terminal 2	SCL/PPM/Analog	BROWN	I2C clock / PPM input signal / Analog Voltage Input
Terminal 3	SDA/Analog Sense	RED	I2C Data / Analog Input Sense
Terminal 4	UART TXD	ORANGE	UART Data Transmit of speed controller, connect to RXD of host
Terminal 5	UART RXD	YELLOW	UART Data Receive of speed controller, connect to TXD of host
Terminal 6	V+	GREEN	V+ should be connected to positive of supply or battery

Motion Control Signal Connection UART

To control the speed of RMCS-220x via UART from a PC, MCU or Host device refer to the connection diagram below. In case of PC a RS232 level convertor must be used as UART works on TTL. The TXD line from the RMCS-220x must be connected to the RXD line from the Host device or controller and vice versa for the TXD line.



Motion Control using UART

The ideal way to use the UART interface of the RMCS-220x is with a terminal software like hyper-terminal, putty, etc. The UART interface works at a fixed baud rate of 9600bps. The UART signals must be TTL logic compatible.

The UART interface on the RMCS-220x prompts the user for a command variable and decimal value string. To set a value of a variable the user must provide an integer decimal value following the command code. To read/display the value of a variable the user must give the command character and immediately follow it by line feed and carriage return. The UART command processor will return the value of the variable. The command list and value range are as follows.

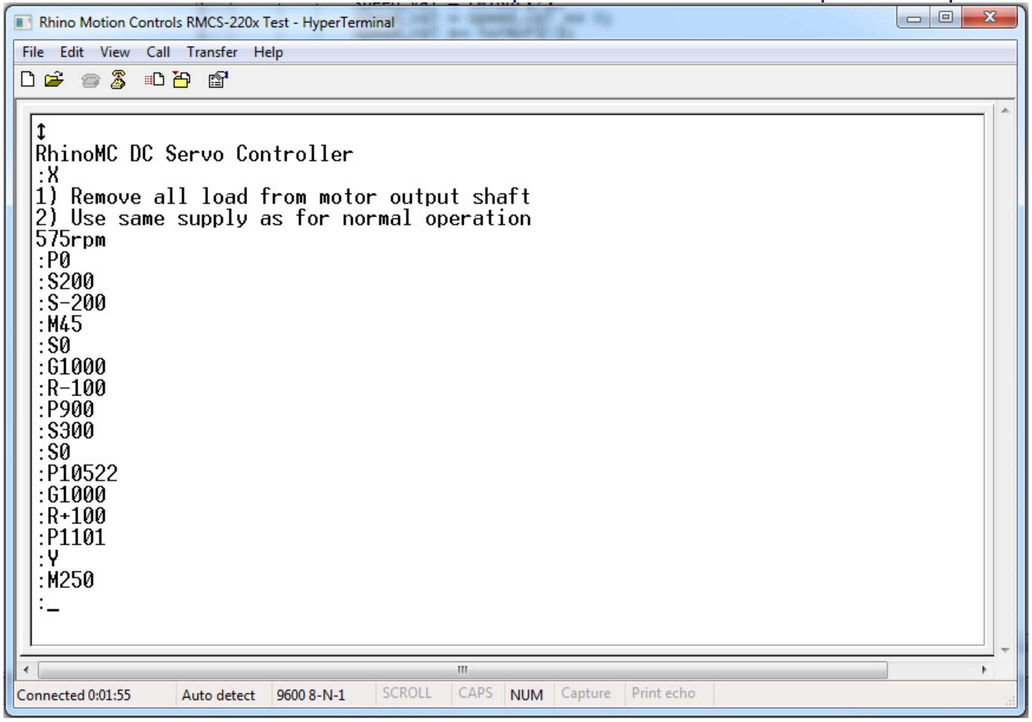
Command	Description	Value Minimum	Value Maximum
‘S’	Read/Write Motor Speed and Direction	-255	+255
‘M’	Read/Write Motor Max Speed	0	255
‘D’	Read/Write Speed Damping	0	255
‘E’	Read/Write I2C address	0	127
‘Y’	Load Default Values of all settings and gains	-	-
‘P’	Read/Write Encoder Position	-2147483648	2147483647
‘G’	Read/Write Go to Position Command	-2147483648	2147483647
‘R’	Write Relative Go to Position Command	-2147483648	2147483647
‘A’	Read/Write Speed-Feedback Gain term	0	32767
‘B’	Read/Write P-Gain term	0	32767
‘C’	Read/Write I-Gain term	0	32767
‘X’	Auto-calibrate Speed-Feedback Gain term	-	-

The motor rotation direction is specified by the '+' and '-'characters before the speed value. "-255" and "+255" are maximums in either direction while the motor will remain at stand still at '0'. The following snapshot depicts normal usage of UART commands to control the speed and motion of the motor. Here the 'S' speed command is first used to read the speed which is initially '0' at startup. Then the motor is commanded to move at a speed of '100' in the forward direction and then at the same speed but in the reverse direction by the value of '-100'. The usage of the 'M', 'D' and 'E' are similar.

The servo-motor's position is represented in number of counts the output shaft has traversed in either forward or reverse direction. Again here the forward direction being represented as '+' and reverse direction as '-'. The encoder position is always reset on start-up to zero. The encoder's counter is a 32bit signed long integer than can be read or written to using the 'P' command. Reading using the 'P' command will return the encoders absolute position in counts, where each count is a 0.2deg of rotation on the output shaft. Writing a value to the encoder counter using the 'P' command will set the current encoder counter to exactly the value that is specified.

The servo-motor can be commanded to rotate to an absolute encoder position by using the 'G' command. On correctly issuing this command with a valid value the servo will immediately start to accelerate to the max-speed setting as per the 'M' command and will rotate at the max-speed set until it arrives at the absolute position specified in the value.

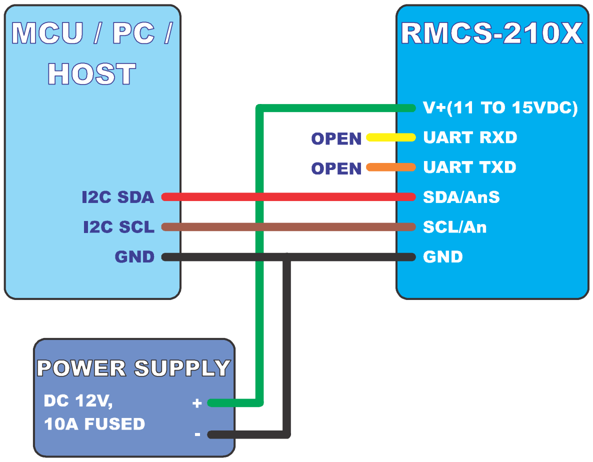
The servo-motor can be commanded to rotate to a encoder position relative to the current encoder position by using the 'R' command. On correctly issuing this command with a valid value the servo will immediately start to accelerate to the max-speed setting as per the 'M' command and will rotate at the max-speed set until it arrives at the current encoder position + the relative position specified in the value.



```
↕
RhinoMC DC Servo Controller
:X
1) Remove all load from motor output shaft
2) Use same supply as for normal operation
575rpm
:P0
:S200
:S-200
:M45
:S0
:G1000
:R-100
:P900
:S300
:S0
:P10522
:G1000
:R+100
:P1101
:Y
:M250
:_
```

Motion Control Signal Connection I2C

RMCS-220x is and I2C slave device with default slave device address of 0x10 (hex) or 16 is decimal. The I2C clock and data lines must be connected to the I2C master line on a host device or controller. The I2C master device must pull-up the I2C lines with 4.7kOhm resistors.



Motion Control using I2C

RMCS-220x is an I2C slave device with a default I2C address of 0x10 (hex), decimal ‘16’. I2C is the easiest communication technique if multiple RMCS-220x or i2C devices are to be controlled from the same I2Ccontrol master.

The I2C interface on the RMCS-220x will receive the command variable number followed by the value that should be written to it. Value format is a 2-byte signed integer or 4-byte signed integer representation and it follows little-Endian byte ordering. The value can be read out immediately after performing a write command using the repeated start command or by the next read command. During the I2C write command if required number of bytes are not received after receiving the command variable the RMCS-220x I2C interface will not write that value to the variable. This is useful to read the value of a variable without having to update its value. The command list and value range are as follows.

Command Byte	Description	Value Minimum	Value Maximum	Format
0	Read/Write Motor Max Speed	0	255	2-byte signed
1	Read/Write Motor Speed and Direction	-255	+255	2-byte signed
2	Read/Write Speed Damping	0	255	2-byte signed
3	Read/Write Encoder Position	-2147483648	2147483647	4-byte signed
4	Read/Write Go to Position Command	-2147483648	2147483647	4-byte signed
5	Read/Write Speed-Feedback Gain term	0	32767	2-byte signed
6	Read/Write P-Gain term	0	32767	2-byte signed
7	Read/Write I-Gain term	0	32767	2-byte signed
8	Write Relative Go to Position Command	-2147483648	2147483647	4-byte signed

Here is an example usage of updating the speed variable on the RMCS-220x to forward 255. We will assume the default slave address of 16.

```
I2C_Start(0x10 + 0);        // send the slave address of the RMCS-220x and write bit 0
I2C_Write(1);               // send the command variable for speed
I2C_Write(255);             // send LSB of 255
I2C_Write(0);               // send MSB of 0 to and so Speed of forward 255
I2C_Stop();                 // send I2C stop
```

Here is an example usage of updating the speed variable on the RMCS-220x to reverse 255 and reading it back. We will assume the default slave address of 16.

```
I2C_Start(0x10 + 0);        // send the slave address of the RMCS-220x and write bit 0
I2C_Write(1);               // send the command variable for speed
I2C_Write(1);               // send LSB of 1
I2C_Write(255);             // send MSB of 255 to and so Speed of backward 255
I2C_Rep_Start(0x10 + 1);            // send I2C address with rep start and 1 to read
speed = I2C_Read_Ack();             // read speed LSB byte and ack
speed = I2C_Read_Nak();             // read speed MSB byte and don’t ack
I2C_Stop();                 // send I2C stop
```

Here is an example usage of updating encoder counter variable on the RMCS-220x to zero. We will assume the default slave address of 16.

```
I2C_Start(0x10 + 0);        // send the slave address of the RMCS-220x and write bit 0
I2C_Write(3);               // send the command variable for encoder counter
I2C_Write(0);               // send 1st -byte to 0
I2C_Write(0);               // send 2nd -byte to 0
I2C_Write(0);               // send 3rd -byte to 0
I2C_Write(0);               // send 4th –byte to 0
I2C_Stop();                 // send I2C stop
```

Here is an example usage of rotating the motor exactly 180deg from the current position on the RMCS-220x to zero. We will assume the default slave address of 16. 180deg is 900 counts which is 0x384 in hex

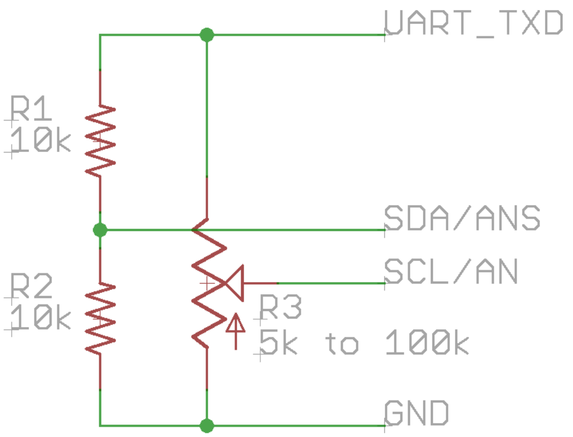
```
I2C_Start(0x10 + 0);        // send the slave address of the RMCS-220x and write bit 0
I2C_Write(8);               // send the command variable for relative go to position command
I2C_Write(0x84);            // send 1st -byte to 0
I2C_Write(0x03);            // send 2nd -byte to 0
I2C_Write(0);               // send 3rd -byte to 0
I2C_Write(0);               // send 4th –byte to 0
I2C_Stop();                 // send I2C stop
```

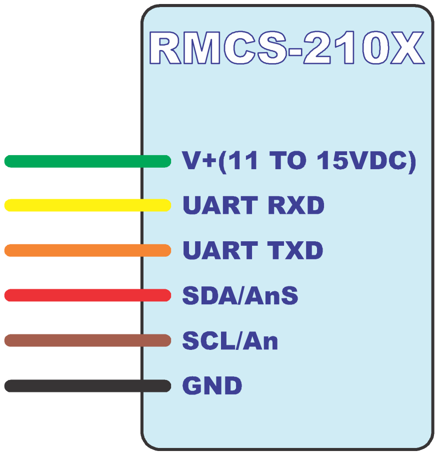
Speed Control Signal Connection for Analog Input

The speed of the RMCS-220x motor can be controlled using via an analog voltage from a supply, Digital to Analog Converter, a fixed resistor divider or a potentiometer.
To enable the analog signal input to the RMCS-220x the SDA/Ans analog input sense must be at 2.5V +/- 0.5V. This can be produced by connecting a resistor divider between UART TXD and GND.

An analog signal for speed control can be produced by putting a potentiometer between UART TXD and GND and connecting the center terminal of the potentiometer to the SCL/An analog input line of the RMCS-220x.

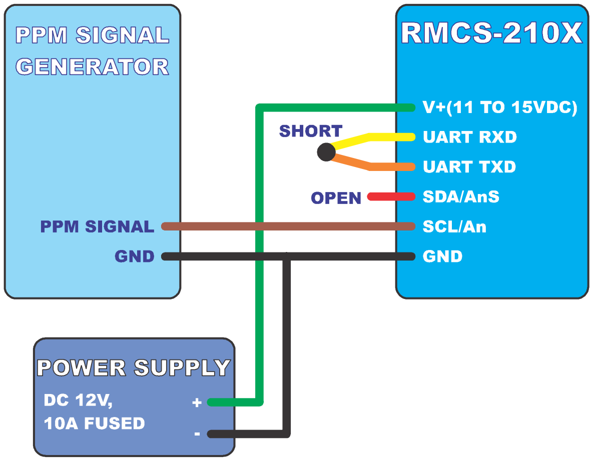
The analog voltage range for speed control is from 0V to 5V DC. The motor speed will be zero at the center voltage of 2.5V.





Speed Control Signal Connection PPM Signal

RMCS-220x motor speed can be controlled using via a PPM signal from any PPM generator like a wireless PPM receiver, Servo controller or a micro-controller I/O pin.



To enable the PPM signal input to the RMCS-220x the UART TXD and UART RXD lines of the RMCS-220x must be shorted together. Remember to power – off and power – on RMCS-220x after shorting RXD and TXD.

The PPM signal pulse width must range from 600us to 2.4ms. The motor speed will be zero at PPM pulse width of 1.5ms.

Maxspeed and Damping

Maxspeed sets the maximum speed at which the motor will rotate in forward or reverse direction. The speed will drop off linearly to 0 from the max speed. For example a max speed setting of 100 and speed setting of 100 will return a true speed of 39.

Damping variable sets a limit on how quickly the true speed and change based on its current value. It allows for smooth ramp up and down for speeds and removes jerks and clicks in the system.

Please note that maxspeed and damping can only be modified via the I2C and UART interfaces. But, they will affect the motion in both the analog and PPM interfaces.

Speed value to RPM conversion

Servo Speed [rpm] = (speed value * max-speed value * motor rated RPM) / 65000

Please note that this is only value when system is auto-calibrated using ‘X’ command via UART communication interface.

Speed range for different interfaces

Interface	Reverse Max Speed	Forward Max Speed	Motor Stand-Still	Input Units
UART	-255	255	0	Ascii Values
I2C	-255	255	0	Signed integer 2's-complent
PPM	0.6	2.4	1.5	ms
Analog	0	5	2.5	Volts

Guide to General Problems

Problem Symptom	Possible Reasons and Solutions
Motor is not rotating	Drive is not powered up Correct inputs are not being provided to update the motor speed
Erratic Motion on Motor or Drive Resets	Power supply voltage not stable or regulated Power supply not able to supply enough current to change the speed or direction Control input signals are not connected properly or not adequate Control signal interference due to power supply or environmental noise
Excessive Motor or Drive Heating	Load on the motor is excessive or irregular Drive is damaged Power supply voltage is too high Not enough cooling or ventilation for motor or drive

Power Supply Selection

A high-torque DC motor requires high current during startup and during high load or irregular load conditions. The general rule of thumb to make sure your power supply is adequate for a DC motor is to make sure it can supply the maximum current required by the motor during stall condition. For RMCS-220X this is 7 Amperes. This doesn't necessarily have to be its continuous current capability but it should be able to provide a pulse of 7 amperes during startup of the motor. It is also good practice to have sufficient low –ESR decoupling capacitors on the output of the supply before you connect it to a DC motor drive. This is to make sure that the motor driver does not reset or suffer from variations in speed due to an insufficient or unregulated supply.

Auto-Calibration of Speed-Feedback Gain

The speed-feedback gain term is auto-calibrated and set at the time of shipping. It is adjusted to give the full range of speeds capable by the servo-motor while powered with a 12V supply. This auto-calibrated value is also reset when using the load defaults command 'Y' in UART communication mode. Please note that the behavior of the motor can become erratic if this gain is not set correctly.

Service and Support

Service and support for this product are available from the Rhino Motion Controls Web site (http://www.rhinomc.com) and our customer service email: info@rhinomc.com

Six-Month Warranty

Rhino Motion Controls (rhinomc.com) warrants its products against defects in materials and workmanship for a period of 6 months from shipment delivery. During the warranty period, Rhino Motion Controls will either, at its option, repair or replace products which proved to be defective.

Exclusions

The above warranty does not extend to any product damaged by reasons of improper or inadequate handlings by customer, improper or inadequate customer wirings, unauthorized modification or misuse, or operation beyond the electrical specifications of the product and/or operation beyond environmental specifications for the product.

Obtaining Warranty Service

To obtain warranty service, please contact our customer service department at info@rhinomc.com before returning product for service. Please make sure that you have gone through this entire installation manual and datasheet before deciding that your product is liable for replacement or repair under this 6-month warranty Customer shall prepay shipping charges for products returned to Rhino Motion Controls for warranty service, and Rhino Motion Controls shall pay for return of products to customer.

Warranty Limitations

Rhino Motion Controls makes no other warranty, either expressed or implied, with respect to the product. Rhino Motion Controls specifically disclaims the implied warranties of merchantability and fitness for a particular purpose. Some jurisdictions do not allow limitations on how long and implied warranty lasts, so the above limitation or exclusion may not apply to you. However, any implied warranty of merchantability or fitness is limited to the 6-month duration of this written warranty.

Disclaimer

This product and its documentation are supplied on an as-is basis and no warranty as to their suitability for any particular purpose is either made or implied.

This document provides preliminary information that may be subject to change without notice.