# Protocol of Serial Port

## Serial Port Configuration

The controller’ s serial communication port is set as follows:

- 115200 bits/s
- 8-bit data
- 1 Start bit
- 1 Stop bit
- No Parity

## Communication Protocol Description

1.Protocol command start from blank, end of enter.

2.For commands where no reply is expected, such as speed setting, the controller will issue a “plus” character (+) followed by a Carriage Return after every command as an acknowledgment

3.Read command:~or？ + command, Ex:Read control mode setting : ~emod or?emod

Change setting command need send“%eesav”; Note:Change the configuration must be send save instructions.(except speed command)

The controller uses a simple communication protocol based on ASCII characters. Commands are not case sensitive. ?a is the same as ?A. Commands are terminated by carriage

return (Hex 0x0d, ‘\r’ ).

## Command Acknowledgement

The controller will acknowledge commands in one of the two ways:

For commands that cause a reply, such as a configuration read or a speed or amps queries,

the reply to the query must be considered as the command acknowledgement.

For commands where no reply is expected, such as speed setting, the controller will issue a “plus” character (+) followed by a Carriage Return after every command as an acknowledgment.

## Command Error

If a command or query has been received, but is not recognized or accepted for any reason, the controller will issue a “minus” character (-) to indicate the error.

If the controller issues the “-” character, it should be assumed that the command was not recognized or lost and that it should be repeated.

## Runtime Command

| Setting command in Running | parameter | description |
|---|---|---|
| ^MAC | Channel Acceleration | Set Acceleration |
| !D0 | Variable Number Value | Set Individual Digital Out bits |
| !D1 | Variable Number Value | Set Individual Digital Out bits |
| ^MDEC | Channel Deceleration | Channel Deceleration |

| | | |
|---|---|---|
| %EESAV | None | Save Configuration in EEPROM |
| !EX | None | Emergency Shutdown |
| !MG | None | Release Shutdown |
| !M | Value | Set Motor Command |
| !P | Value | Next Go to Absolute Desired Position |
| ^MVEL | Value | Set position mode speed |
| **Command** | **Arguments** | **Description** |
| ?C | None | Set position mode counter |
| ?A | InputNbr | Read Motor Amps |
| ?AI | InputNbr | Read Analog Inputs |
| ?AIC | InputNbr | Read Analog Input after Conversion |
| ?BA | InputNbr | Read Battery Amps |
| ?BS | None | Read BL Motor Speed in RPM |
| ?CIA | Channel | Read Internal Analog Command |
| ?CIP | Channel | Read Internal Pulse Command |
| ?CIS | Channel | Read Internal Serial Command |
| ?D | InputNbr | Read All Digital Inputs |
| ?DI | InputNbr | Read Individual Digital Inputs |
| ?DO | None | Read Current Digital Outputs |
| ?E | None | Read Closed Loop Error |
| ?F | None | Read Feedback |
| ?FF | None | Read Fault Flags |
| ?FS | None | Read Status Flags |
| ?LK | None | Read Lock status |
| ?M | Channel | Read Actual Motor Command |
| ?P | Channel | Read Applied Power Level |
| ?PI | InputNbr | Read Pulse Inputs |
| ?PIC | Channel | Read Pulse Input after Conversion |
| ?S | Channel | Read Encoder Motor Speed in RPM |
| ?T | Sensor Number | Read Case & Internal Temperatures |
| ?V | Sensor Number | Read Internal Voltages |

### ^MAC - Motor Acceleration Rate

Set the rate of speed change during acceleration for a motor channel. This command is identical to the AC realtime command. Acceleration value is in 0.1*RPM per second. When using controllers fitted with encoder, the speed and acceleration value are actual RPMs.

Brushless motor controllers use the hall sensor for measuring actual speed and acceleration will also be in actual RPM/s.

When using the controller without speed sensor, the acceleration value is relative to the Max RPM configuration parameter, which itself is a user-provide number for the speed normally expected speed at full power. Assuming that the Max RPM parameter is set to 1000, and acceleration value of 10000 means that the motor will go from 0 to full speed in exactly 1 second, regardless of the actual motor speed.

Syntax: ^MAC cc nn

~MAC [cc]

Where: cc = Motor channel

nn = Acceleration time in 0.1 RPM per seconds

Allowed Range: 100 to 32000

**!D0 - Reset Individual Digital Out bits**

The D0 command will turn off the single digital output selected by the number that follows.

Syntax: !D0 nn

Where: nn = output number

Examples: !D0 2: will turn output 2 to 0

**D1 - Set Individual Digital Out bits**

The D1 command will activate the single digital output that is selected by the parameter

that follows.

Syntax: !D1 nn

Where: nn = output number

Examples: !D1 1: will turn ON output 1

**MDEC - Motor Deceleration Rate**

This parameter sets the motor deceleration. It is the same as MACC but for when the motor goes from a high speed to a lower speed.

Syntax: ^MDEC cc nn

~MDEC [cc]

Where: cc = Motor channel

nn = Deceleration time in 0.1 RPM per second

Allowed Range: 100 to 32000

**DS - Set all Digital Out bits**

The D command will turn ON or OFF one or many digital outputs at the same time. The number can be a value from 0 to 255 and binary representation of that number has 1bit affected to its respective output pin.

Syntax: !DS nn

Where: nn = bit pattern to be applied to all output lines at once

Examples: !DS 03: will turn ON outputs 1 and 2. All others are off

**EESAV - Save Configuration in EEPROM**

Controller configuration that have been changed using any Configuration Command can then be saved in EEPROM. Once in EEPROM, it will be loaded automatically in the controller every time the unit is powered on. If the EESAV command is not called after changing a configuration, the configuration will remain in RAM and active only until the controller is turned off. When powered on again, the previous configuration that was in the EEPROM is loaded. This command uses no parameters

Syntax: %EESAV

Notes: Do not save configuration while motors are running. Saving to

EEPROM takes several milliseconds, during which the control loop is suspended.

**EX - Emergency Stop**

The EX command will cause the controller to enter an emergency stop in the same way as
if hardware emergency stop was detected on an input pin. The emergency stop condition will remain until controller is reset or until the MG release command is received.

Syntax: !EX

**MG - Emergency Stop Release**

The MG command will release the emergency stop condition and allow the controller to
return to normal operation.

Syntax: !MG

!M—Set Motor Speed

```
   Syntax：   !M nn mm
   Where:：    nn= Channel 1
                  Speed value 0 to 1000 is Forward
                  Speed value 0 to -1000 is Reverse

                mm= Channel 2
                  Speed value 0 to 1000 is Forward
                  Speed value 0 to -1000 is Reverse
```

If the controller is single channel, please send !M nn

**P - Go to Absolute Desired Position**

This command is used in the Position Count mode to make the motor move to a specified encoder count value.Commands Reference

Syntax: !P [nn] mm

Where: nn = motor channel

mm = absolute count destination (encoder ppr * 4 = 1 circle, set range is ±2147483647)

Example: !P 1 10000: make motor go to absolute count value 10000

**C - Read Encoder Counter Absolute**

Returns the encoder value as an absolute number. The counter is a 32-bit counter with a range of +/- 2000000000 counts.

Syntax: ?C [cc]

Reply: C=nn

Where: cc = channel number

nn = absolute counter value

?C------Read counter counter value

!MS 1    Stop Ch1 to Run

Allow accept pulse max value -2147483647

**MVEL - Default Position Velocity**

This parameter is the default speed at which the motor moves while in position mode. Values are in RPMs. To change velocity while the controller is in operation, use the !S runtime
command.
Syntax: ^MVEL [cc] nn
~MVEL [cc]
Where: cc = Motor Channel. May be omitted in single channel controllers
nn = Velocity value in RPM

**A - Read Motor Amps**

Measures and reports the motor Amps for all operating channels. Note that the current flowing through the motors is often higher than this flowing through the battery.
Syntax: ?A [cc]
Reply: A = aa
Where: cc = motor channel
aa = Amps *10 for each channel
Examples: Q: ?A
R: A=100:200
Q: ?A 2
R: A=200
Notes: Single channel controllers will report a single value. Sepex controllers
report the motor Amps and the Field excitation Amps.
Some power board units measure the Motor Amps and calculate the
Battery Amps, while other models measure the Battery Amps and calculate the Motor Amps. The measured Amps is always more precise than
the calculated Amps. See controller datasheet to find which Amps is
measured by your particular model.

**AI - Read Analog Input**

Reports the raw value in mV of each of the analog inputs that are enabled. Input that is disabled will report 0.
Syntax: ?AI [cc]
Reply: AI=nn
Where: cc = Analog Input number
nn = millivolt for each channel
Allowed Range: 0 to 5000mV

**AIC - Read Analog Input after Conversion**

Returns value of an Analog input after all the adjustments are performed to convert it to a
command or feedback value (Min/Max/Center/Deadband/Linearity). If an input is disabled,
the query returns 0.
Syntax: ?AIC
Reply: AIC=nn

Where: nn = Converted analog input value +/-1000 range

**BA - Read Battery Amps**

Measures and reports the Amps flowing from the battery. Battery Amps are often lower than motor Amps.

Syntax: ?BA [cc]

Reply: BA=aa

Where: cc = motor channel

aa = Amps *10 for each channel

Examples: Q: ?BA

R: BA=100:200

Notes: Single channel controllers will report a single value. Sepex controllers report a single value with the battery current for both the Armature and Field excitation.

Some power board units measure the Motor Amps and Calculate the Battery Amps, while other models measure the Battery Amps and calculate the Motor Amps. The measured Amps is always more precise than the calculated Amps. See controller datasheet to find which Amps is measured by your particular model.

**BS - Read BL Motor Speed in RPM**

On brushless motor controllers, reports the actual speed measured using the motor's Hall sensors as the actual RPM value.

Syntax: ?BS

Reply: BS=nn

Where: nn = speed in RPM

Notes: To report RPM accurately, the correct number of motor poles must be loaded in the BLPOL configuration parameter.

**CIA - Read Internal Analog Command**

Returns the motor command value that is computed from the Analog inputs whether or not the command is actually applied to the motor. This query can be used, for example, to read the command joystick from within a MicroBasic script or from an external microcomputer, even though the controller may be currently responding to RS232 or Pulse command because of a higher priority setting. The returned value is the raw Analog input value with all the adjustments performed to convert it to a command (Min/Max/Center/Deadband/Linearity).

Syntax: ?CIA

Reply: CIA=nn

Where: nn = command value in +/-1000 range

**CIP - Read Internal Pulse Command**

Returns the motor command value that is computed from the Pulse inputs whether or not the command is actually applied to the motor. This query can be used, for example, to read the command joystick from within a MicroBasic script or from an external microcomputer even though the controller may be currently responding to RS232 or Analog command because of a higher priority setting. The returned value is the raw Pulse input value with all the adjustments performed to convert it to a command (Min/Max/Center/Deadband/Linearity).

Syntax: ?CIP

Reply: CIP=nn

Where: nn = command value in +/-1000 range

**CIS - Read Internal Serial Command**

Returns the motor command value that is issued from the serial input or from a MicroBasic script whether or not the command is actually applied to the motor. This query can be used, for example, to read from an external microcomputer the command generated inside MicroBasic script, even though the controller may be currently responding to a Pulse or Analog command because of a higher priority setting.

Syntax: ?CIS

Reply: CIS=nn

Where: nn = command value in +/-1000 range

**D - Read Digital Inputs**

Reports the status of each of the available digital inputs. The query response is a single digital number which must be converted to binary and gives the status of each of the inputs.

Syntax: ?D [cc]

Reply: D=nn

Where: cc = Digital Input number

nn = b1 + b2*2 + b3*4 + ... +bn*2n-1

Examples: Q: ?D

R: D=17 : Inputs 1 and 5 active, all others inactive

**DI - Read Individual Digital Inputs**

Reports the status of an individual Digital Input. The query response is a boolean value (0 or 1).

Syntax: ?DI [cc]

Reply: DI=nn

Where: cc = Digital Input number

nn = 0 or 1 state for each input

Examples: Q: ?DI

R: DI=1:0:1:0:1:0

Q: ?DI 1

R: DI=0

**DO - Read Digital Output Status**

Reads the actual state of all digital outputs. The response to that query is a single number which must be converted into binary in order to read the status of the individual output bits.

Syntax: ?DO [cc]

Reply: DO=nn

Where: cc = Digital Input number

nn = d1 + d2*2 + d3*4 + ... + dn * 2n-1

Examples: Q: ?DO

R: DO=17 : Outputs 1 and 5 active, all others inactive

Q: ?DO 1

R: DO=1 : Queried output 1 is active

**E - Read Closed Loop Error**

In closed-loop modes (Speed or Position), returns the difference between the desired speed or position and the measured feedback. This query can be used to detect when the motor has reached the desired speed or position. In open loop mode, this query returns 0.

Syntax: ?E

Reply: E=nn

Where: nn = error

**F - Read Feedback In**

Reports the value of the feedback sensors that are associated to each of the channels in closed-loop modes. The feedback source can be Encoder, Analog or Pulse. Selecting the feedback source is done in the encoder, pulse or analog configuration parameters. This query is useful for verifying that the correct feedback source is used by the channel in the closed-loop mode and that its value is in range with expectations.

Syntax: ?F [cc]

Reply: F=nn

Where: cc =channel number

nn = feedback values

**FF - Read Fault Flag**

Reports the status of the controller fault conditions that can occur during operation. The response to that query is a single number which must be converted into binary in order to evaluate each of the individual status bits that compose it.

Syntax: ?FF [cc]

Reply: FF = f1 + f2*2 + f3*4 + ... + fn*2n-1

Where: f1 = overheat

f2 = overvoltage

f3 = undervoltage

f4 = short circuit

f5 = emergency stop

f6 = Sepex excitation fault

f7 = MOSFET failure

f8 = startup configuration fault

**FS - Read Status Flag**

Report the state of status flags used by the controller to indicate a number of internal conditions during normal operation. The response to this query is the single number for all status flags. The status of individual flags is read by converting this number to binary and look at various bits of that number.

Syntax: ?FS

Reply: FS = f1 + f2*2 + f3*4 + ... + fn*2n-1

Where: f1 = Serial mode

f2 = Pulse mode

f3 = Analog mode

f4 = Power stage off

f5 = Stall detected

f6 = At limit

f7 = Unused

f8 = Micro Basic script running

**LK - Read Lock Status**

Returns the status of the lock flag. If the configuration is locked, then it will not be possible to read any configuration parameters until the lock is removed or until the parameters are reset to factory default. This feature is useful to protect the controller configuration from being copied by unauthorized people.

Syntax: ?LK

Reply: LK=ff

Where: ff = 0 : unlocked

1 : locked

**M - Read Motor Command Applied**

Reports the command value that is being used by the controller. The number that is reported will be depending on which mode is selected at the time. In the RS232 mode, the reported value will be the command that is entered in via the

RS232 or USB port and to which an optional exponential correction is applied.

In the Analog and Pulse modes, this query will report the Analog or Pulse input after it is being converted using the min, max, center, deadband, and linearity corrections.

This query is useful for viewing which command is actually being used and the effect of the correction that is being applied to the raw input.

Syntax: ?M [cc]

Reply: M=nn

Where: cc = channel number

nn = command value used for each motor. 0 to ±1000 range

Examples: Q: ?M

R: M=800:-1000

Q: ?M 1

R: M=800

**P - Read Motor Power Output Applied**

Reports the actual power that is being applied to the motor at the power output stage. This value takes into account all the internal corrections and any limiting resulting from temperature or over current.

Syntax: ?P [cc]

Reply: P=p1:p2

Where: cc = motor channel
p1, p2 = 0 to ±1000 power level
Examples: Q: ?P 1
R: P=800
Notes: For Sepex controllers this query will report the applied power on the
Armature and Field excitation.

**PI - Read Pulse Input**

Reports the value of each of the enabled pulse input captures. The value is the raw number in microseconds when configured in Pulse Width mode. In Frequency mode, the returned value is in Hertz. In Duty Cycle mode, the reported value ranges between 0 and 4095 when the pulse duty cycle is 0% and 100% respectively.

Syntax: ?PI [cc]
Reply: PI=nn
Where: cc = Pulse capture channel number
nn = value *each channel
Allowed Range: 0 to 65000μs

**PIC - Read Pulse Input after Conversion**

Returns value of a Pulse input after all the adjustments were performed to convert it to a command or feedback value (Min/Max/Center/Deadband/Linearity). If an input is disabled, the query returns 0.

Syntax: ?PIC
Reply: PIC=nn
Where: nn = Converted analog input value +/-1000 range

**S - Read Encoder Speed RPM**

Reports the actual speed measured by the encoders as the actual RPM value.
Syntax: ?S [cc]
Reply: S =vv:vv
Where: cc = channel number
vv = speed in RPM
Notes: To report RPM accurately, the correct Pulses per Revolution (PPR) must
be stored in the encoder configuration

**T - Read Temperature**

Reports the temperature at each of the Heatsink sides and on the internal silicon chips. The reported value is in degrees C with a one degree resolution.

Syntax: ?T [cc]
Reply: T=tm:t1:t2
Where: cc = temperature channel
tm = internal ICs
t1 = channel1 side
t2 = channel2 side

**V - Read Volts**

Reports the voltages measured inside the controller at three locations: the main battery voltage, the internal voltage at the motor driver stage, and the voltage that is available on the 5V output on the DSUB 15 or 25 front connector. For safe operation, the driver stage voltage must be above 12V. The 5V output will typically show the controller ’ s internal regulated 5V minus the drop of a diode that is used for protection and will be in the 4.7V range. The battery voltage is monitored for detecting the undervoltage or overvoltage conditions.

Syntax: ?V [cc]

Reply: V=vdr:vmot:v5out

Where: vdr = internal voltage in Volts *10

vmot = main battery voltage in Volts *10

v5out = 5V output on DSub connector in millivolts

Examples: Q: ?V

R:V=135:246:4730

Q: ?V 3

R:V=4730