

## Introduction

This document specifies a new ASCII command line protocol to control the 2<sup>nd</sup> generation 5-phase, 2-phase, and encoded motor controller.

The controller shall respond to simple commands sent over the serial or USB connections. The protocol is a simple terminal style command line protocol. All commands shall be terminated with a line feed. The controller shall respond to all commands with a readable ASCII response message terminated with a line feed, and followed by a command prompt. The command prompt shall be the “\$” character, followed by a single space character.

The basic command line structure is defined as follows:

<command> [<args>]

where <command> is any one of the commands define below, followed optionally by zero or more arguments which are command specific. The command and arguments shall be separated by whitespace.

The [<args>] are specific to the <command> and are described with each command below.

The controller shall respond to unrecognized or ill-formatted commands with an appropriate error message.

Serial Port Settings for USB/RS-232:

Baud: 38400

Data bits: 8

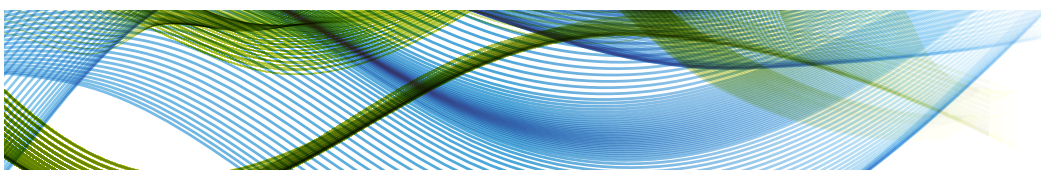
Stop bits: 1

Parity: None

Flow control XON/XOFF

## Supported Commands

| <b>Command</b>  | <b>Purpose</b>                                 |
|-----------------|--|
| read            | Request the value of a single register         |
| write           | Overwrite the value of a single register       |
| savesetup       | Save all setup registers to NV storage         |
| stopall         | Stop all motor motion                          |
| defaultsetup    | Load all setup registers with factory defaults |
| programfirmware | Enter bootloader for firmware upgrade          |
| help            | Request a description of the protocol usage    |



## read

The read command is used to request the value of a single controller register. The command format is:

read <regname|regnumber>

where <regname> is the symbolic name of the register and <regnumber> is the register number in either decimal or hexadecimal format.

If the controller recognizes the specified register, it shall respond with the current value of the register. If the specified register is not recognized, the controller shall respond with an appropriate error message.

## write

The write command is used to overwrite the current value of a single controller register. The command format is:

write <regname|regnumber> <value>

where <regname> is the symbolic name of the register, <regnumber> is the register number in either decimal or hexadecimal format, and <value> is the new value to be written to the register in either decimal or hexadecimal format.

If the controller recognizes the specified register, and the new value is acceptable, the controller shall respond with the new value of the register. If the specified register is not recognized, or the specified value is not acceptable the controller shall respond with an appropriate error message.

## savesetup

The savesetup command is used to save all current setup register values to the controller Non-Volatile storage. The command takes no arguments. The controller will respond with the command prompt.

## stopall

The stopall command is used to request that all motor motion stop. The command takes not arguments. The controller shall immediately stop all motion and respond with the command prompt.

## defaultsetup

The defaultsetup command is used to load all setup register values with factory default values. The command takes no arguments. The controller will respond with the command prompt.

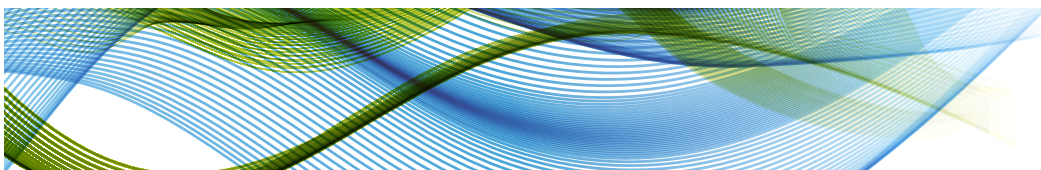
NOTE: This command does not save the setup register values to non-volatile storage. Thus, returning the controller to its factory default state requires executing the defaultsetup command followed by the savesetup command.

## programfirmware

The programfirmware command is used to request firmware upgrade mode. The command takes no arguments. The controller will respond by stopping all motor motion and entering the bootloader to await a firmware upgrade.

## help

The help command is used to request a summary of the command protocol usage, and has no arguments. The controller shall respond to the help command with a typical command usage description.

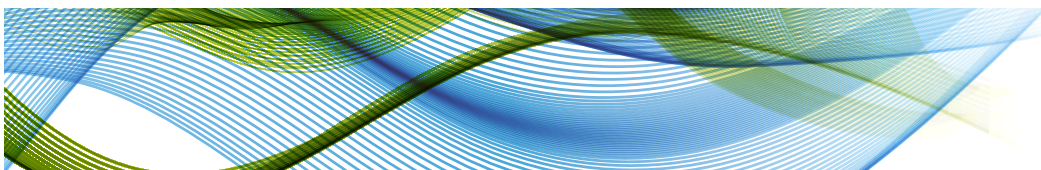


## Device Register Descriptions

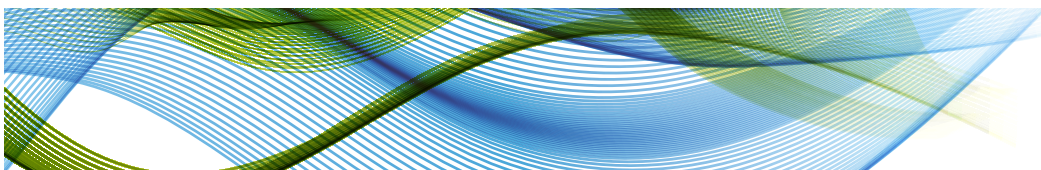
The following table includes all device registers and their definitions. Register names that begin with “setup\_” are saved to controller Non-Volatile storage when the savesetup command is issued.

Additional register definitions may be added over time as needed for controller product variants and feature enhancements.

| <regname>          | <regnumber> | values  | read/<br>write | description                                   |
|--------------------|-------------|---|----------------|---|
| productid          | 0x01        | Product -specific   | r              | The controller product ID code                |
| versionhw          | 0x02        |   | r              | The controller hardware version number        |
| versiondate        | 0x03        |   | r              | The controller hardware version date          |
| versionsw          | 0x04        |   | r              | The controller firmware version number        |
| productid_subclass | 0x05        | 1: 2 phase stepper<br>2: 5 phase stepper<br>3: DC Encoder<br>4: LED | r              | The controller sub class ID.                  |
| product_serialnum  | 0x06        |   | r              | The controller factory assigned serial number |
| target_1           | 0x10        |   | r/w            | Motor 1 target position                       |
| target_2           | 0x20        |   | r/w            | Motor 2 target position                       |
| increment_1        | 0x11        |   | r/w            | Motor 1 position delta from current position  |
| increment_2        | 0x21        |   | r/w            | Motor 2 position delta from current position  |
| current_1          | 0x12        |   | r              | Motor 1 current position                      |
| current_2          | 0x22        |   | r              | Motor 2 current position                      |
| limit_1            | 0x13        | 0: Home<br>1: Far Limit<br>2: Abort all motion                      | r/w            | Motor 1 limit seek request                    |



| <regname>     | <regnumber> | values   | read/<br>write | description                |
|---------------|-------------|--|----------------|----------------------------|
| limit_2       | 0x23        | 0: Home<br>1: Far Limit<br>2: Abort all motion   | r/w            | Motor 2 limit seek request |
| status_1      | 0x14        | Bits [0..7]<br>0: idle<br>1: driving to home<br>2: coming off home<br>3: driving to limit<br>4: seeking forward<br>5: decel forward<br>6: forward backlash<br>7: seeking reverse<br>8: decel reverse<br>9: reverse backlash<br>11: for ward decel during abort<br>12: reverse decel during abort<br>Bit 8<br>Motor at home<br>Bit 9<br>Motor at limit<br>Bit 10<br>LED 1 On<br>Bit 11<br>LED 2 On<br>Bit 12<br>Insufficient voltage<br>(+24V rail)<br>Bits[13..14]<br>0: Motor Type #1<br>1: Motor Type #2<br>2: Motor Type #3<br>3: Motor Type #4<br>(Controller specific.<br>For 2-Phase:<br>#1: Unknown<br>#2: AM0820<br>#3: AM1524<br>#4: Unknown,<br>possible short or<br>open coil | r              | Motor 1 status             |
| status_2      | 0x24        | Same as status_1   | r              | Motor 2 status             |
| setup_accel_1 | 0x15        |  | r/w            | Motor 1 acceleration       |
| setup_accel_2 | 0x25        |  | r/w            | Motor 2 acceleration       |
| setup_initv_1 | 0x16        |  | r/w            | Motor 1 initial velocity   |



|                     |      |  |     |                                |
|---------------------|------|--|-----|--------------------------------|
| setup_initv_2       | 0x26 |  | r/w | Motor 2 initial velocity       |
| setup_maxv_1        | 0x17 |  | r/w | Motor 1 maximum velocity       |
| setup_maxv_2        | 0x27 |  | r/w | Motor 2 maximum velocity       |
| setup_revbacklash_1 | 0x18 |  | r/w | Motor 1 reverse backlash value |
| setup_revbacklash_2 | 0x28 |  | r/w | Motor 2 reverse backlash value |
| setup_fwdbacklash_1 | 0x19 |  | r/w | Motor 1 forward backlash value |
| setup_fwdbacklash_2 | 0x29 |  | r/w | Motor 2 forward backlash value |
| setup_config_1      | 0x1b | Bit 0:<br>0 = near is home<br>1 = far is home<br>Bit 1:<br>0 == reverse seek direct<br>1 = reverse seek through h ome<br>Bit 2:<br>0 = Axis 1 enabled<br>1 = Axis 1 disabled | r/w | Motor 1 sensor configuration   |
| setup_config_2      | 0x2b | Bit 0:<br>0 = near is home<br>1 = far is home<br>Bit 1:<br>0 == reverse seek direct<br>1 = reverse seek through home<br>Bit 2:<br>0 = Axis 2 enabled<br>1 = Axis 2 disabled  | r/w | Motor 2 sensor configuration   |
| setup_limit_1       | 0x1c |  | r   | Motor 1 limit position value   |
| setup_limit_2       | 0x2c |  | r   | Motor 2 limit position value   |

