# BDC-COMM Application

# User's Guide

BDC-COMM-UG-02

# Copyright

Texas Instruments
108 Wild Basin, Suite 350
Austin, TX 78746
http://www.ti.com/stellaris

# Table of Contents

# B D C - C O M M   A P P L I C A T I O N

## Introduction

The BDC-COMM application is a command line or graphical user interface (GUI) application designed to help you control and monitor the MDL-BDC24. Using your computer's COM port and a special cable, the application interfaces to the MDL-BDC24 using simple serial/UART communication. With an MDL-BDC24 as the main interface, the application allows connectivity to a network of MDL-BDC24 and legacy MDL-BDC devices that are connected using the built-in CAN interface of the boards.

### System Requirements

BDC-COMM is built using the Fast Light Tool Kit (FLTK) cross-platform GUI development software allowing the application to run on both Windows and Linux systems.

**NOTE:** Because the BDC-COMM application sends out a periodic heartbeat to keep the board alive, slower systems may have trouble keeping up. The heartbeat is sent out every 50 ms, so if you have an older or slow system, the heartbeats may not always get out in time. Missing a heartbeat makes the LED on the MDL-BDC24/MDL-BDC flash instead of remain solid. It also causes the motor to stop spinning if it is currently running.

## Hardware Requirements

To use the application, connect the MDL-BDC24 to a PC. The MDL-BDC24 must be the first board in a chain of motor controllers. The legacy MDL-BDC does not have the hardware support needed to communicate with the PC over the serial/UART link.

Spotting the difference between a MDL-BDC24 and MDL-BDC is easy. The newer MDL-BDC24 comes in black plastic with a red Texas Instruments logo on the fan grill. The legacy MDL-BDC comes in grey plastic with an orange Luminary Micro logo on the fan grill.

#### Building the Interface Cable

See the *MDL-BDC24 Getting Started Guide* for information about building the required interface cable.

## BDC-COMM Basics

The BDC-COMM application is capable of running in both GUI mode (see "Running BDC-COMM in GUI Mode" on page 5) and command line mode (see "Running BDC-COMM in Command Line Mode" on page 5).

### Running BDC-COMM in GUI Mode

To launch the application in GUI mode, simply double-click on the bdc-comm.exe file. When running in Windows, a console window opens briefly before the GUI appears, which is typical behavior for Windows executables. The console window disappears on its own.

Launch the GUI from the command line by typing "bdc-comm.exe" in the console window. No additional input arguments are needed.

### Running BDC-COMM in Command Line Mode

The BDC-COMM application defaults to command line mode when input arguments are passed to it when launching from the console window. For example, typing:

```
> bdc-comm.exe –c 1
```

(which tells the application to open using COM1), launches the application in command line mode. When the application is running, your console appears with a hash (# sign). For example:

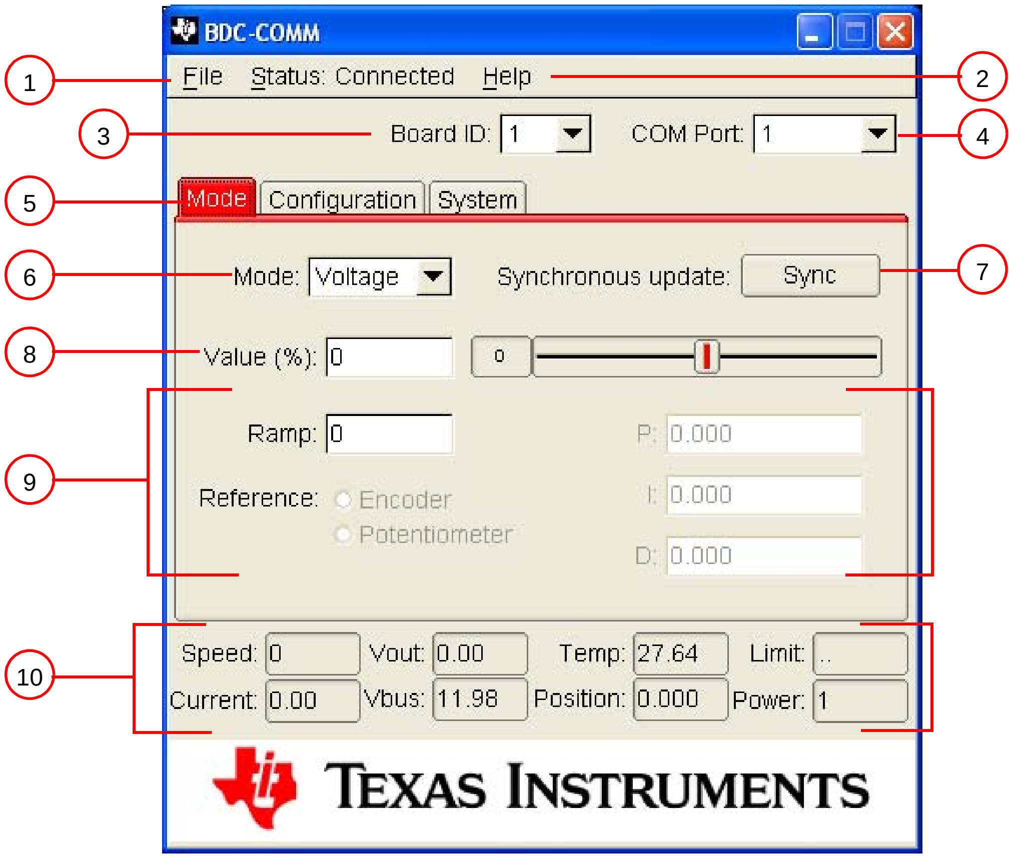
```
> bdc-comm.exe –c 1
#
```

When the hash symbol appears, you can begin typing commands. If you need a list of the available commands, type “help”.

## Using the GUI

Figure 1-1 shows the main GUI window and Figure 1-2 and Figure 1-3 show the Configuration and System tabs. More details on the interface are provided in Table 1-1 on page 8. When the application launches, it tries to open a connection to the MDL-BDC24 through COM1. If COM1 is unavailable or not connected to the board, it does not show any real-time status information. If no board is connected to the application, the status indicator in the application menu bar shows “Disconnected” and the Board ID drop-down lists “- -“ as the current board ID.

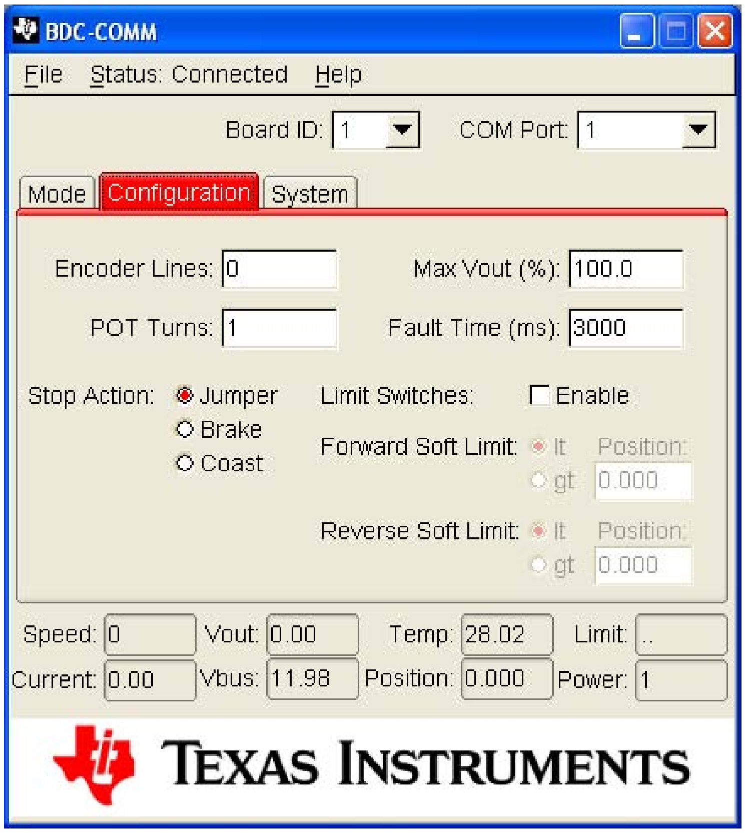
**Figure 1-1. BDC-COMM Main GUI Window**



Board ID: 1
COM Port: 1
Mode: Voltage
Synchronous update:
Value (%): 0
Ramp: 0
Reference:
P: 0.000
I: 0.000
D: 0.000
Speed: 0
Vout: 0.00
Temp: 27.64
Limit: ..
Current: 0.00
Vbus: 11.98
Position: 0.000
Power: 1

**Figure 1-2. Configuration Tab**

11



Board ID: 1 COM Port: 1

Encoder Lines: 0
Max Vout (%): 100.0
POT Turns: 1
Fault Time (ms): 3000

Stop Action: [✓ Jumper] Brake Coast

Limit Switches: ☐ Enable

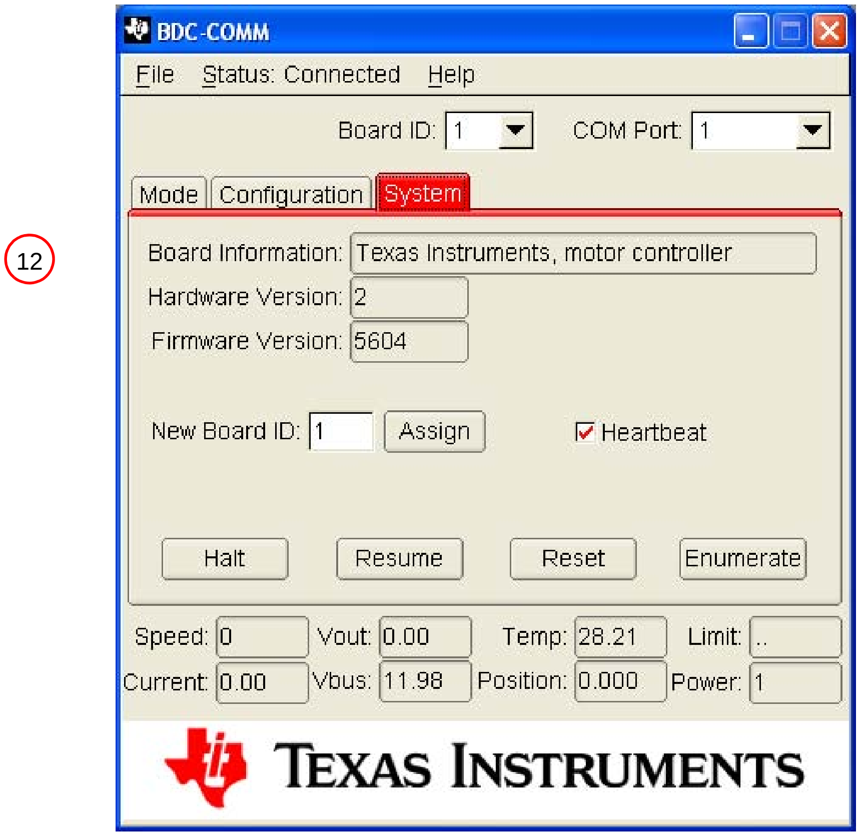
Forward Soft Limit: [✓ lt] gt Position: 0.000

Reverse Soft Limit: [✓ lt] gt Position: 0.000

Speed: 0 Vout: 0.00 Temp: 28.02 Limit: ..

Current: 0.00 Vbus: 11.98 Position: 0.000 Power: 1

**Figure 1-3. System Tab**



Board Information: Texas Instruments, motor controller
Hardware Version: 2
Firmware Version: 5604
New Board ID: 1
☑ Heartbeat
Speed: 0
Vout: 0.00
Temp: 28.21
Limit: ..
Current: 0.00
Vbus: 11.98
Position: 0.000
Power: 1

**Table 1-1. Description of GUI Window Controls**

| Item Number | Name | Description |
|---|---|---|
| 1 | **Menu Bar** | |
| | **File Menu** | **Update Firmware**<br>Displays firmware update dialog window. Firmware update requires a binary file (*.bin). Invoke this option using the keyboard shortcut "Ctrl+U". |
| | | **Exit**<br>Exits/Quits the application. Keyboard shortcut "Ctrl+Q" |
| | **Status Indicator** | **Connect/Disconnect**<br>The text of the status indicator shows the current connection status. Clicking this option drops down a connect or disconnect menu item, depending on the current status. |
| 2 | **Help** | Displays help for the BDC-COMM application. |
| 3 | **Board ID** | Displays the current active board. Selecting a different board from the drop-down list changes the current active board. |
| 4 | **COM Port** | Displays the current active COM port. Selecting a different COM port from the drop-down list disconnects the active COM port connection and attempts to open up a connection to the new COM port. The Status Indicator (item number 2) is updated if a board is found. |

**Table 1-1. Description of GUI Window Controls (Continued)**

| Item Number | Name | Description |
|---|---|---|
| 5 | **Command Grouping Tabs** | The commands are grouped into operation types: Mode (items related to Current Operating mode), Configuration (items related to the configuration of the control system), and System (board information, board ID assignment, basic system level commands). |
| 6 | **Mode** | The Mode drop-down controls the active control mode used for the motor controller. The board defaults to Voltage Control mode out of reset. Changing the control mode enables or disables options (item number 9) that are relevant to the selected mode. |
| 7 | **Synchronous Update** | When the ‘Sync’ button is clicked, it turns red to indicate that Synchronous Update mode is enabled. Synchronous Update mode allows you to change the values in the Mode tab without having them apply immediately. Clicking the ‘Sync’ button a second time applies the newly selected values and disables Synchronous Update Mode. |
| 8 | **Value** | This field sets the value for the active control mode. For example, voltage in Voltage Mode, current in Current Mode. The value can be set in the box or using the slider control, and setting one automatically updates the other. You can also “click and drag” the value box to set the value instead of manually typing in the value. |
| 9 | **Control Mode Options** | The control mode options change with each control mode. The options that are not applicable to the active mode are grayed out. If the control mode is changed and an option becomes active, the target is polled to get the current value on the board. |
| 10 | **Real-Time Status** | These boxes contain the real-time status information of the active board. Some fields are not updated in every control mode, so they may not change. The status information is updated every 500 ms. |
| 11 | **Configuration Tab** | |
| | **Encoder Lines** | Selects the number of encoder lines, if present. This option only applies when the ‘Reference’ option in the Mode tab is set to ‘Encoder’. |
| | **POT Turns** | Selects the number of POT turns, if present. This option only applies when the ‘Reference’ option in the Mode Tab is set to ‘Potentiometer’. |
| | **Max Vout** | Changes the maximum output voltage that the motor can operate. |
| | **Fault Time** | Indicates the amount of time that the fault indicator is active. |
| | **Stop Action** | Selects what to do when the motor is stopped.<br>Jumper – The jumper on the board indicates the action.<br>Brake – The motor stops immediately.<br>Coast – the motor coasts to a stop, and no braking is applied. |
| | **Limit Switches** | Disables or enables the use of the hardware (jumper) limit switches using the checkbox.<br>Forward Software Limit – If the jumpers are not used, sets the limit position and whether it is less than (lt) or greater than (gt) the set position.<br>Reverse Software Limit – If the jumpers are not used, sets the limit position and whether it is less than (lt) or greater than (gt) the set position. |

**Table 1-1. Description of GUI Window Controls (Continued)**

| Item Number | Name | Description |
|---|---|---|
| 12 | **System Tab** | |
| | **Board/Hardware Information** | Shows information about the motor controller, including details of the hardware and firmware version. |
| | **Board ID Assignment** | Changes the ID of the active board. Once the new ID is chosen and entered in the box, click the ‘Assign’ button. A 5-second countdown begins, during which time you must press the user button on the motor controller. If you press the button, the board attempts to blink the LED X times, where X is the new ID. Note that after 5 seconds, the heartbeat resumes and depending on the length of the ID, it may be cut off by the heartbeat.<br><br>If you miss the 5-second window to press the button on the MDL-BDC24, any device on the network that had the new ID already assigned has its ID erased. This condition appears as a fast blinking pattern on the LED. In this state, you can still assign an ID to the board. |
| | **Heartbeat** | Enables/disables the heartbeat. |
| | **Halt** | Halts the system. If the system is halted, all motors are put into the neutral state and continue to drive until the Resume command has been issued. |
| | **Resume** | Enables the system from a halt state. |
| | **Reset** | Resets the system over the network. |
| | **Enumerate** | Re-enumerates the network. |

# Using the Command Line

The BDC-COMM application can also be run from the command line, as described in “Running BDC-COMM in Command Line Mode” on page 5. Users who are more comfortable with a command line utility can use the BDC-COMM application without the GUI and can perform advanced tasks such as scripting.

## Commands

The commands that BDC-COMM provides are identical to the command list described in the CAN Interface section of the *RDK-BDC Software User’s Guide* (document name SW-RDK-BDC-UG-xxxx.pdf). For a more thorough description of each command, see that document.

Type “help,” “h,” or “?” at any time on the command line to view the full list of available commands. This displays the full list of commands along with a brief description of what the commands do. For example:

```
> bdc-comm.exe –c 1
# help
   help        - display a list of commands
   h           - alias for help
   ?           - alias for help
   id          - set the target ID
   heartbeat   - start/stop the heartbeat
   volt        - voltage control mode commands
   cur         - current control mode commands
   speed       - speed control mode commands
   pos         - position control mode commands
   stat        - status commands
   config      - configuration commands
   system      - system commands
   update      - update the firmware
   boot        - wait for boot loader to request update
   exit        - exit the program
   quit        - alias for exit
   q           - alias for exit
#
```

### ID Command

Sets a new active board ID.

***Format:***

```
# id <num>
```

Where <num> is any value between 1 and 63.

### Heartbeat Command

Enables or disables the heartbeat.

***Format:***

```
# heartbeat [on|off]
```

## Voltage Control Mode

The "volt" command has many sub-commands that control the various aspects of Voltage Control mode. Type "volt" on the command line to see a list of the valid options:

```
# volt
volt [en|dis|set|ramp]
```

### Voltage Mode Enable

Voltage mode is enabled using the "en" sub-command. This command allows the other voltage related commands to control the voltage applied to the output of the MDL-BDC24.

***Format:***

```
# volt [en]
```

### Voltage Mode Disable

Voltage mode is disabled using the "dis" sub-command. Issuing this command disables all other voltage commands from affecting the output voltage of the MDL-BDC24.

***Format:***

```
# volt dis
```

### Voltage Mode Set (Value)

Sets the voltage when Voltage Control mode is enabled. If Voltage Control mode is not enabled, using this command does not affect the output voltage on the MDL-BDC24.

***Format:***

```
# volt [set] <val> <sync_group>
```

Where <val> is a 16-bit 8.8 signed fixed-point number. The optional 8-bit < sync_group > field specifies the group number for a synchronous update.

### Voltage Mode Set Ramp

Sets the ramp rate over an extended period of time.

***Format:***

```
# volt ramp <val>
```

Where <val> is a 16-bit 8.8 unsigned fixed point number that indicates the maximum rate of change for the voltage.

## Current Control Mode

The "cur" command has many sub-commands that control the various aspects of Current Control mode. Type "cur" on the command line to see a list of the valid options:

```
# cur
cur [en|dis|set|p|i|d]
```

### Current Mode Enable

Current mode is enabled using the "en" sub-command. This command allows the other current related commands to control the amount of current applied to the output of the MDL-BDC24.

***Format:***

```
# cur [en]
```

### Current Mode Disable

Current control mode is disabled using the "dis" sub-command. Issuing this command disables all other current control commands from affecting the output of the MDL-BDC24.

***Format:***

```
# cur dis
```

### Current Mode Set (Value)

Sets the current when Current Control mode is enabled. If Current Control mode is not enabled, using this command does not affect the output voltage on the MDL-BDC24.

***Format:***

```
# cur [set] <val> <sync_group>
```

Where <val> is a 32-bit 16.16 signed fixed point number. The optional 8-bit < sync_group > field specifies the group number for a synchronous update.

### Current Mode P, I and D

Sets the P (proportional), I (integral) and D (differential) constants for the PID control algorithm.

***Format:***

```
# current [p|i|d] <val>
```

Where <val> is a 32-bit 16.16 signed fixed point number.

## Speed Control Mode

The "speed" command has many sub-commands that control the various aspects of Speed Control mode. Type "speed" on the command line to see a list of the valid options:

```
# speed
speed [en|dis|set|p|i|d|ref]
```

### Speed Mode Enable

Speed mode is enabled using the "en" sub-command. This command allows the other speed control related commands to affect the speed of the motor attached to the MDL-BDC24.

***Format:***

```
# speed [en]
```

### Speed Mode Disable

Speed control mode is disabled using the "dis" sub-command. Issuing this command disables all other speed control commands from affecting the output of the MDL-BDC24.

***Format:***

```
# speed dis
```

### Speed Mode Set (Value)

Sets the speed when Speed Control mode is enabled. If Speed Control mode is not enabled, using this command does not affect the output voltage on the MDL-BDC24.

***Format:***

```
# speed [set] <val> <sync_group>
```

Where <val> is a 32-bit 16.16 signed fixed point number. The optional 8-bit <sync_group> field specifies the group number for a synchronous update.

### Speed Mode P, I and D

Sets the P (proportional), I (integral) and D (differential) constants for the PID control algorithm.

***Format:***

```
# speed [p|i|d] <val>
```

Where <val> is a 32-bit 16.16 signed fixed point number.

### Speed Mode Reference

Sets the reference used for measuring the current speed of the motor. Currently, the only supported speed reference is an encoder.

***Format:***

```
# speed ref <val>
```

Where <val> is a 8-bit unsigned number.

## Position Control Mode

The "pos" command has many sub-commands that control the various aspects of position control mode. Type "pos" on the command line to see a list of the valid options:

```
# pos
pos [en|dis|set|p|i|d|ref]
```

### Position Mode Enable

Position mode is enabled using the "en" sub-command. This command allows the other position control related commands to affect the position of the motor attached to the MDL-BDC24.

***Format:***

```
# pos [en]
```

### Position Mode Disable

Position control mode is disabled using the "dis" sub-command. Issuing this command disables all other position control commands from affecting the output of the MDL-BDC24.

***Format:***

# pos dis

### Position Mode Set (Value)

Sets the position when position control mode is enabled. If position control mode is not enabled, using this command does not affect the output voltage on the MDL-BDC24.

***Format:***

```
# pos [set] <val> <sync_group>
```

Where <val> is the position represented as a 32-bit value. The optional 8-bit <sync_group> field specifies the group number for a synchronous update.

### Position Mode P, I and D

Sets the P (proportional), I (integral) and D (differential) constants for the PID control algorithm.

***Format:***

```
# pos [p|i|d] <val>
```

Where <val> is a 32-bit 16.16 signed fixed point number.

### Position Mode Reference

Sets the reference used for measuring the current speed of the motor. Currently, the only supported speed reference is an encoder.

***Format:***

```
# pos ref <val>
```

Where <val> is a 8-bit unsigned number.

## Board Status

The "stat" command has many sub-commands that report information about the current status of the board. Type "stat" on the command line to see a list of the valid options:

```
# stat
stat [vout|vbus|fault|cur|temp|pos|speed|limit|power]
```

### Vout Status

Obtains the current voltage being applied to the motor.

***Format:***

```
# stat vout
```

**Return:**

```
stat vout (board_id) = <val>
```

Where board_id is the current selected board, and <val> is the most recent value obtained from the motor controller.

### Vbus Status

Obtains the current bus voltage.

***Format:***

```
# stat vbus
```

**Return:**

```
stat vbus (board_id) = <val> (real_value)
```

Where board_id is the current selected board, and <val> is the most recent value obtained from the motor controller. The real value of the bus voltage is also displayed in parenthesis (for example, 12.15).

### Fault Status

Obtains the current fault status.

***Format:***

```
# stat fault
```

**Return:**

```
stat fault (board_id) = <val>
```

Where board_id is the current selected board, and <val> is the most recent fault value obtained from the motor controller.

### Current Status

Obtains the most recent current value (Current mode).

***Format:***

```
# stat cur
```

**Return:**

```
stat cur (board_id) = <val> (real_value)
```

Where board_id is the current selected board, and <val> is the most recent value obtained from the motor controller. The real value of the current is also displayed in parenthesis (for example, 1.25).

### Temperature Status

Obtains the current temperature reading.

***Format:***

```
# stat temp
```

**Return:**

```
stat temp (board_id) = <val> (real_value)
```

Where board_id is the current selected board, and <val> is the most recent value obtained from the motor controller. The real value of the temperature is also displayed in parenthesis (for example, 25.65).

### Position Status

Obtains the current position (Position mode).

***Format:***

```
# stat pos
```

**Return:**

```
stat pos (board_id) = <val> (real_value)
```

Where board_id is the current selected board, and <val> is the most recent value obtained from the motor controller. The real value of the position is also displayed in parenthesis (for example, 12.15).

**Speed Status**

Obtains the current speed (Speed mode).

***Format:***

```
# stat speed
```

**Return:**

```
stat speed (board_id) = <val> (real_value)
```

Where board_id is the current selected board, and <val> is the most recent value obtained from the motor controller. The real value of the speed is also displayed in parenthesis (for example, 12.15).

**Limit Status**

Obtains the values of the limit switches.

***Format:***

```
# stat limit
```

**Return:**

```
stat limit (board_id) = <info>
```

Where board_id is the current selected board, and <info> is the fitting of the jumpers. If the forward or reverse limit switches are set, an "F" or "R" appears. Otherwise, a "." appears in that position. (for example, if forward is fitted and reverse is not, it displays "F.".

**Power Status**

Obtains the power.

***Format:***

```
# stat power
```

**Return:**

```
stat power (board_id) = <val>
```

Where board_id is the current selected board, and <val> is the most recent value obtained from the motor controller.

## Board Configuration

The "config" command has many sub-commands that set configuration value of the board. Type "config" on the command line to see a list of the valid options:

```
# config
config [lines|turns|brake|limit|fwd|rev|maxvout|faulttime]
```

### Encoder Lines

Sets the number of encoder lines (if encoder present).

***Format:***

```
# config lines <val>
```

Where <val> is the number of encoder lines, represented as a 16-bit unsigned value.

### Potentiometer Turns

Sets the number of potentiometer turns (if POT present).

***Format:***

```
# config turns <val>
```

Where <val> is the number of potentiometer turns, represented as a 16-bit unsigned value.

### Braking

Sets the braking mode for the motor.

***Format:***

```
# config brake [jumper|brake|coast]
```

Where jumper lets the jumper configuration choose, brake stops the motor immediately, and coast applies no braking.

### Limit Switches

Configure the use of limit switches (hardware versus software).

***Format:***

```
# config limit [on|off]
```

If set to "on", the motor controller uses the hardware limit switches. If "off", it uses software limits.

### Forward Soft Limit

Configure the use of software limit in the forward direction.

***Format:***

```
# config fwd <pos> [lt|gt]
```

Where <pos> is the software-determined position limit, represented as a 16.16 fixed point value. The "lt" and "gt" specify whether the limit is less than (lt) or greater than (gt) the value of <pos>.

### Reverse Soft Limit

Configure the use of software limit in the reverse direction.

***Format:***

```
# config rev <pos> [lt|gt]
```

Where <pos> is the software-determined position limit, represented as a 16.16 fixed point value. The "lt" and "gt" specify whether the limit is less than (lt) or greater than (gt) the value of <pos>.

### Maximum Vout

Sets the maximum output voltage.

***Format:***

```
# config maxvout <val>
```

Where <val> is the maximum output voltage specified as a 16-bit 8.8 unsigned fixed point number.

### Fault Time

Configure the amount of time a fault is active for.

***Format:***

```
# config faulttime <val>
```

Where <val> is the amount of time (in milliseconds) that the fault is active. This value is represented as a 16-bit unsigned value.

## System Commands

The "system" command has many sub-commands that perform system level operations. Type "system" on the command line to see a list of the valid options:

```
# system
system [halt|resume|reset|enum|assign|query|sync|version|hwver]
```

### Halt

Halt the system (all boards). When halt is issued, all motor controllers stop driving the motor, go to a neutral state. All motors remain stopped until a resume or reset command has been received.

***Format:***

```
# system halt
```

### Resume

Resume the system from the halted state. If the system is halted, a resume must be sent before the MDL-BDC24 will allow any commands to affect the output and allow the motor to move.

***Format:***

```
# system resume
```

### Reset

Upon receiving this message the motor controller stops driving the motor, goes to a neutral state, and resets internal settings to their boot settings.

***Format:***

```
# system reset
```

### Enumerate

This command causes the motor controller to send out a response to indicate that device is present on the CAN network. In order to prevent all devices from responding at once, the motor controllers wait for (device number) * 1ms after the enumerate command before responding. A list of connected devices is returned.

***Format:***

```
# system enum
```

### Assign ID

This command causes the motor controller to enter the assignment state. Upon reception of this command with the new ID, the controller waits 5 seconds for the user push button to be pressed. Once pressed, the controller accepts the new ID and blinks the LED to indicate the new assigned value (that is, blinks 5 times for an ID of 5).

***Format:***

```
# system assign <val>
```

Where <val> is the new ID, ranging from 1 to 63. ID 1 is typically reserved for the first device in the network.

### Query

Query the active device for basic information. It should return some information such as “Texas Instruments Motor Controller”.

***Format:***

```
# system query
```

### Synchronous Update

Allows up to 8 groups of devices to be simultaneously updated with a single command.

***Format:***

```
# system sync <val>
```

Where <val> is a bitmask of the groups that are to be updated. It is represented as a 8-bit unsigned value.

### Version

Request the current version of firmware of the motor controller. The motor controller returns the value.

***Format:***

```
# system version
```

### Hardware Version

Request the current hardware version of the motor controller. The motor controller returns the value. A value of 1 indicates the device connected to the given ID is an MDL-BDC and a value of 2 is an MDL-BDC24.

***Format:***

```
# system hwver
```

## Firmware Update

The "update" command allows you to update the firmware of your board using the embedded bootloader application.

***Format:***

```
# update <file_name>
```

Where <file_name> is the name of the binary file (*.bin) that you would like to program to the board. Please note that if you program an invalid binary file, you may render the board inoperable. See the "Wait for Bootloader" section for updating a device with an incorrect image.

## Wait for Bootloader

Wait for the bootloader to request an update. Once the bootloader requests the update, the firmware update begins using the specified file. The MDL-BDC24 boot loader checks if the button is pressed on reset and issue a command to request a firmware update. If nothing responds to this request, the MDL-BDC24 continues to boot normally. The "boot" command is run to respond to this request and start an update automatically using the <file_name> provided. This command should be issued while holding down the button on the MDL-BDC24 that is being updated. Once the update starts, the button can be released and the firmware update continues normally. The "boot" command should only be used when the "update" cannot successfully update a controller and only works on MDL-BDC24 controllers.

***Format:***

```
# boot <file_name>
```

Where <file_name> is the name of the binary file (*.bin) that you would like to program to the board.

| **Products** | | **Applications** | |
|---|---|---|---|
| Amplifiers | amplifier.ti.com | Audio | www.ti.com/audio |
| Data Converters | dataconverter.ti.com | Automotive | www.ti.com/automotive |
| DLP® Products | www.dlp.com | Communications and Telecom | www.ti.com/communications |
| DSP | dsp.ti.com | Computers and Peripherals | www.ti.com/computers |
| Clocks and Timers | www.ti.com/clocks | Consumer Electronics | www.ti.com/consumer-apps |
| Interface | interface.ti.com | Energy | www.ti.com/energy |
| Logic | logic.ti.com | Industrial | www.ti.com/industrial |
| Power Mgmt | power.ti.com | Medical | www.ti.com/medical |
| Microcontrollers | microcontroller.ti.com | Security | www.ti.com/security |
| RFID | www.ti-rfid.com | Space, Avionics & Defense | www.ti.com/space-avionics-defense |
| RF/IF and ZigBee® Solutions | www.ti.com/lprf | Video and Imaging | www.ti.com/video |
| | | Wireless | www.ti.com/wireless-apps |