Programmer’s Manual

Communication with the robot through the USB port

The robot has a USB port next to the power jack at the rear of the robot. You will need a USB 5-pin mini cable to connect to the robot. You will also need to install a device driver to talk with the robot. The best way to install the device driver is to install it from the Software Update Page.

Communication to a robot through a terminal emulator

Commands can be sent to a real robot via a terminal emulator. The robot is running an API command interpreter on the USB port. Connection to the robot is made through a USB cable. The driver converts the USB port connection to a Com port connection. This allows you to use any terminal emulator program (such as Hyperterm or TeraTerm) to communicate with the robot. Use the following procedure:

1. Turn on the robot and connect the USB 5 pin mini cable between your computer and your robot.

2. Bring up the Terminal Emulator program.

3. Find the COM port that the robot is connected to. This is usually the last one on the list. The communication parameters (Baud, start/stop bits, parity, etc.) are unimportant. They apply only to a real COM port.

4. Type GetVersion. If you are connected to the robot, then it will echo back each character that you type. Hit the enter key. The robot will now process the command and supply a response.

5. Now type help to get a list of available commands.

Command Syntax

Commands are matched with a case insensitive method. Partial word match is supported. You only need to type in enough letters of the command to make it unique. Commands have a flexible format. In the strictly non-sequenced format, the syntax consists of only 3 elements:

- **Cmd** – This is the command name.
- **Flag** – Flags are always initialized to zero. Specifying a flag on the command line sets the flag to one.
- **ParamName ParamValue** – The ParamName specifies what parameter the next argument (ParamValue) is.

Flags and Param(Name/Value) pairs can be specified on the command line in any order. All Flags and ParamNames are matched with a case insensitive method. Partial word match is supported. You only need to type in enough letters of the Flag or ParamName to make it unique. In a less verbose format, the user can omit the ParamName from the Param(Name/Value) pairs. However, this format requires that all ParamValues are specified in the correct sequence. This sequence is the sequence shown in the Usage command. Unnamed ParamValues are assigned in sequence to the earliest ParamName in the sequence that does not already have a value assigned.

Response Syntax

All responses will have a control-Z (^Z) at the end of the response string. Command responses are in the form of CSV spread sheets. E.G. GetVersion returns:

- Component,Major,Minor,Build
- Product Model,XV-11,,
- Serial Number,AAAnnnnAA,0000000,D
- Software,6,1,13328
- LDS Software,V1.0.0,,
- LDS Serial,XXX-YYY,,
- MainBoard Vendor ID,1,,
MainBoard Serial Number,99,,
MainBoard Version,0,8,
Chassis Version,-1,,
UIPanel Version,-1,,

The first line of text is the column labels. Each line afterwards is composed of a row label, a comma, and then data associated with that label. The order and number of the rows and columns is not guaranteed to stay the same from release to release. The labels used will not be changed, but new rows and columns may be added anywhere. The application should parse the full response into rows and columns, and look up the particular value by matching its row and column names in the table.
## Table of Robot Application Commands

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clean</td>
<td>Starts a cleaning by simulating press of start button.</td>
</tr>
<tr>
<td>DiagTest</td>
<td>Executes different test modes. Once set, press Start button to engage. (Test modes are mutually exclusive.)</td>
</tr>
<tr>
<td>GetAccel</td>
<td>Get the Accelerometer readings.</td>
</tr>
<tr>
<td>GetAnalogSensors</td>
<td>Get the A2D readings for the analog sensors.</td>
</tr>
<tr>
<td>GetButtons</td>
<td>Get the state of the UI Buttons.</td>
</tr>
<tr>
<td>GetCalInfo</td>
<td>Prints out the cal info from the System Control Block.</td>
</tr>
<tr>
<td>GetCharger</td>
<td>Get the diagnostic data for the charging system.</td>
</tr>
<tr>
<td>GetDigitalSensors</td>
<td>Get the state of the digital sensors.</td>
</tr>
<tr>
<td>GetErr</td>
<td>Get Error Message.</td>
</tr>
<tr>
<td>GetLDSScan</td>
<td>Get scan packet from LDS.</td>
</tr>
<tr>
<td>GetLifeStatLog</td>
<td>Get All Life Stat Logs.</td>
</tr>
<tr>
<td>GetMotors</td>
<td>Get the diagnostic data for the motors.</td>
</tr>
<tr>
<td>GetSchedule</td>
<td>Get the Cleaning Schedule. (24 hour clock format)</td>
</tr>
<tr>
<td>GetSysLog</td>
<td>Get System Log data.</td>
</tr>
<tr>
<td>GetTime</td>
<td>Get Current Scheduler Time.</td>
</tr>
<tr>
<td>GetVersion</td>
<td>Get the version information for the system software and hardware.</td>
</tr>
<tr>
<td>GetWarranty</td>
<td>Get the warranty validation codes.</td>
</tr>
<tr>
<td>Help</td>
<td>Without any argument, this prints a list of all possible cmds. With a command name, it prints the help for that particular command</td>
</tr>
<tr>
<td>PlaySound</td>
<td>Play the specified sound in the robot.</td>
</tr>
<tr>
<td>RestoreDefaults</td>
<td>Restore user settings to default.</td>
</tr>
<tr>
<td>SetDistanceCal</td>
<td>Set distance sensor calibration values for min and max distances.</td>
</tr>
<tr>
<td>SetFuelGauge</td>
<td>Set Fuel Gauge Level.</td>
</tr>
<tr>
<td>SetLCD</td>
<td>Sets the LCD to the specified display. (TestMode Only)</td>
</tr>
<tr>
<td>SetLDSRotation</td>
<td>Sets LDS rotation on or off. Can only be run in TestMode.</td>
</tr>
<tr>
<td>SetLED</td>
<td>Sets the specified LED to on, off, blink, or dim. (TestMode Only)</td>
</tr>
<tr>
<td>SetMotor</td>
<td>Sets the specified motor to run in a direction at a requested speed. (TestMode Only)</td>
</tr>
<tr>
<td>SetSchedule</td>
<td>Modify Cleaning Schedule.</td>
</tr>
<tr>
<td>SetSystemMode</td>
<td>Set the operation mode of the robot. (TestMode Only)</td>
</tr>
<tr>
<td>SetTime</td>
<td>Sets the current day, hour, and minute for the scheduler clock.</td>
</tr>
<tr>
<td>SetWallFollower</td>
<td>Enables/Disables wall follower</td>
</tr>
<tr>
<td>TestMode</td>
<td>Sets TestMode on or off. Some commands can only be run in TestMode.</td>
</tr>
<tr>
<td>Upload</td>
<td>Uploads new program to the robot.</td>
</tr>
</tbody>
</table>
Detailed Command Descriptions

Command: **Clean**

**Description:** Starts a cleaning by simulating press of start button.

**Usage:** Clean [House] [Spot] [Stop]

**Options:**

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>House</td>
<td>(Optional) Equivalent to pressing ‘Start’ button once. Starts a house cleaning. (House cleaning mode is the default cleaning mode.)</td>
</tr>
<tr>
<td>Spot</td>
<td>(Optional) Starts a spot clean.</td>
</tr>
<tr>
<td>Stop</td>
<td>Stop Cleaning,</td>
</tr>
</tbody>
</table>

**Return Format:** None

---

Command: **DiagTest**

**Description:** Executes different test modes. Once set, press Start button to engage. (Test modes are mutually exclusive.)

**Usage:** DiagTest [TestsOff] [DrivePath] [DriveForever] [MoveAndBump] [DropTest] [AutoCycle] [OneShot] [BrushOn] [VacuumOn] [LDSOn] [AllMotorsOn] [DisablePickupDetect] [DrivePathDist <DrivePathDist_value>] [DriveForeverLeftDist <DriveForeverLeftDist_value>] [DriveForeverRightDist <DriveForeverRightDist_value>] [DriveForeverSpeed <DriveForeverSpeed_value>] [Speed <Speed_value>] [BrushSpeed <BrushSpeed_value>]

**Options:**

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TestsOff</td>
<td>Stop Diagnostic Test and clears all diagnostic test modes.</td>
</tr>
<tr>
<td>DrivePath</td>
<td>Sets DrivePath TestMode. Press start button to start. Robot travels straight by commanded distance as path. Mutually exclusive with other diagtest modes. Use ‘TestsOff’ option to stop.</td>
</tr>
<tr>
<td>MoveAndBump</td>
<td>Sets Move and Bump TestMode. Press start button to start. Executes canned series of motions, but will react to bumps. Mutually exclusive with other diagtest modes.</td>
</tr>
<tr>
<td>DropTest</td>
<td>Enables DropTest. Robot drives forward until a drop is detected. Mutually exclusive with other diagtest modes.</td>
</tr>
<tr>
<td>AutoCycle</td>
<td>DropTest argument to enable automatic restart of the test. The robot will drive backwards and then forward until a drop is detected until the test is over.</td>
</tr>
<tr>
<td>OneShot</td>
<td>Only executes test once.</td>
</tr>
<tr>
<td>BrushOn</td>
<td>Turns on brush during test. May conflict with motor commands of test so use carefully!</td>
</tr>
<tr>
<td>VacuumOn</td>
<td>Turns on vacuum during test. May conflict with motor commands of test so use carefully!</td>
</tr>
<tr>
<td>LDSOn</td>
<td>Turns on LDS during test. May conflict with motor commands of test so use carefully!</td>
</tr>
<tr>
<td>------------</td>
<td>-------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>AllMotorsOn</td>
<td>Turns on brush, vacuum, and lds during test. May conflict with motor commands of test so use carefully!</td>
</tr>
<tr>
<td>DisablePickupDetect</td>
<td>Ignores pickup (wheel suspension). By default, pickup detect is enabled and stops the test.</td>
</tr>
<tr>
<td>DrivePathDist</td>
<td>Distance in mm</td>
</tr>
<tr>
<td>DriveForeverLeftDist</td>
<td>Use next arg to set left wheel dist for DriveForever test. Requires DriveForeverRightDist as well. The ratio of this value to DriveForeverRightDist determines turn radius.</td>
</tr>
<tr>
<td>DriveForeverRightDist</td>
<td>Use next arg to set right wheel dist for DriveForever test. Requires DriveForeverLeftDist as well. The ratio of this value to DriveForeverLeftDist determines turn radius.</td>
</tr>
<tr>
<td>DriveForeverSpeed</td>
<td>Use next arg to set turn speed of outer wheel for DriveForever test in mm/s.</td>
</tr>
<tr>
<td>Speed</td>
<td>DropTest argument to set the robot speed in mm/s.</td>
</tr>
<tr>
<td>BrushSpeed</td>
<td>DropTest argument to set the speed of the brush in rpm.</td>
</tr>
</tbody>
</table>

**Return Format**: None

---

**Command**: GetAccel  
**Description**: Get the Accelerometer readings.  
**Usage**: GetAccel  
**Options**: None  
**Return Format**:

```
Label,Value PitchInDegrees, 0.00 RollInDegrees, 0.00 XInG, 0.000 YInG, 0.000 ZInG, 0.000 SumInG, 0.000
```

---

**Command**: GetAnalogSensors  
**Description**: Get the A2D readings for the analog sensors.  
**Usage**: GetAnalogSensors [raw] [stats]  
**Options**:

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
</table>
| raw  | Return raw analog sensor values as milliVolts.  
(DefaultValue in native units of what they measure.) |
| stats | Return stats (avg,max,min,dev,cnt) of raw analog sensor values as milliVolts.  
(Implies 'raw' option) |

**Return Format**:

```
'GetAnalogSensors' produces:  SensorName,Value WallSensorInMM,34585 BatteryVoltageInmV,19761 LeftDropInMM,60 RightDropInMM,60 RightMagSensor,0 LeftMagSensor,0 XTemp0InC,28 XTemp1InC,28 VacuumCurrentInmA,1342 ChargeVoltInmV,0 NotConnected1,0 BatteryTemp1InC,20 NotConnected2,0 CurrentInmA,3493 NotConnected3,0
```
**Command: GetButtons**

**Description:** Get the state of the UI Buttons.

**Usage:** GetButtons

**Options:** None

**Return Format:**

Button Name,Pressed BTN_SOFT_KEY,0 BTN_SCROLL_UP,0 BTN_START,0 BTN_BACK,0 BTN_SCROLL_DOWN,0

---

**Command: GetCalInfo**

**Description:** Prints out the cal info from the System Control Block.

**Usage:** GetCalInfo

**Options:** None

**Return Format:**

Parameter, Value
- LDSOffset, 0
- XAccel, 0
- YAccel, 0
- ZAccel, 0
- RTCOffset, 0
- LCDContrast, 43
- RDropMin, -1
- RDropMid, -1
- RDropMax, -1
- LDropMin, -1
- LDropMid, -1
- LDropMax, -1
- WallMin, -1
- WallMid, -1
- WallMax, -1

---

**Command: GetCharger**

**Description:** Get the diagnostic data for the charging system.

**Usage:** GetCharger

**Options:**

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
</table>

**Return Format:**

Charger Variable Name, Value Label, Value Label
- FuelPercent, 100
- BatteryOverTemp, 0
- ChargingActive, 0
- ChargingEnabled, 0
- ConfidentOnFuel, 0
- OnReservedFuel, 0
- EmptyFuel, 0
- BatteryFailure, 0
- ExtPwrPresent, 0
- ThermistorPresent[0], 0
- ThermistorPresent[1], 0
- BattTempCAvg[0], 103
- BattTempCAvg[1], 103
- VBatt, 0.21
- VExtV, 0.00
- Charger_mAH, 0
- MaxPWM, 65536
- PWM, -858993460
Command: **GetDigitalSensors**

**Description:** Get the state of the digital sensors.

**Usage:** GetDigitalSensors

**Options:** None

**Return Format:**

```
Digital Sensor Name, Value
SNSR_DC_JACK_CONNECT,0
SNSR_DUSTBIN_IS_IN,1
SNSR_LEFT_WHEEL_EXTENDED,0
SNSR_RIGHT_WHEEL_EXTENDED,0
LSIDEBIT,0
LFRONTBIT,0
RSIDEBIT,0
RFRONTBIT,0
```

Command: **GetErr**

**Description:** Get Error Message.

**Usage:** GetErr [Clear]

**Options**

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clear</td>
<td>Dismiss the reported error.</td>
</tr>
</tbody>
</table>

**Return Format:**

If an error currently exists, then report the error message. Otherwise no msg.
Possible Error Msgs are:
- 1 - WDT
- 2 - SSEG LED
- 3 - BTN LED
- 4 - BACK LED
- 5 - FLASH
- 10 - BattNominal
- 11 - BattOverVolt
- 12 - BattUnderVolt
- 13 - BattOverTemp
- 14 - BattShutdownTemp
- 15 - BattUnderCurrent
- 16 - BattTimeout
- 17 - BattTempPeak
- 18 - BattFastCapacity
- 19 - BattMACapacity
- 20 - BattOnReserve
- 21 - BattEmpty
- 22 - BattMismatch
- 23 - BattLithiumAdapterFailure
- 207 - I had to reset my system. Please press START to clean
- 217 - Please unplug my Power Cable when you want me to clean.
- 218 - Please set schedule to ON first.
- 220 - Please set my clock first.
- 222 - Please put my Dirt Bin back in.
- 223 - Please check my Dirt Bin and Filter. Empty them as needed.
- 224 - My Brush is overheated. Please wait while I cool down.
- 225 - My Battery is overheated. Please wait while I cool down.
- 226 - I am unable to navigate. Please clear my path.
- 227 - Please return me to my base.
- 228 - My Bumper is stuck. Please free it.
- 229 - Please put me down on the floor.
- 230 - My Left Wheel is stuck. Please free it from debris.
- 232 - My Right Wheel is stuck. Please free it from debris.
- 233 - I have an RPS error. Please visit web support.
- 234 - My Brush is stuck. Please free it from debris.
- 235 - My Brush is overloaded. Please free it from debris.
- 236 - My Vacuum is stuck. Please visit web support.
- 237 - Please Check my filter and Dirt Bin.
- 238 - My Battery has a critical error. Please visit web support.
- 239 - My Brush has a critical error. Please visit web support.
- 240 - My Schedule is now OFF.
- 241 - I can’t shut down while I am connected to power.
- 243 - A Software update is available. Please visit web support.
- 244 - My SCB was corrupted. I reinitialized it. Please visit web support.
- 245 - Please Dust me off so that I can see.

Command: **GetLDSScan**

**Description:** Get scan packet from LDS.

**Usage:** GetLDSScan

**Options:**

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
</table>

**Return Format:**

- Get scan packet from LDS.
360 output lines of LDS Scan Angle, Distance code in MM, normalized spot intensity, and error code. Followed by 2 status variable pairs. Example:

```
AngleInDegrees,DistInMM,Intensity,ErrorCode
HEX 0,221,1400,0
1,223,1396,0
2,228,1273,0
. . .
359,220,1421,0
```

**ROTATION_SPEED (in Hz, Floating Point), 5.00**

---

**Command:** **GetLifeStatLog**

**Description:** Get All Life Stat Logs.

**Usage:** GetLifeStatLog

**Options:** None

**Return Format:**

Multiple LifeStat logs are output, from the oldest to the newest. Note that only the non-zero entries are printed.

```
runID,statID,count,Min,Max,Sum,SumV*2
0,LS_RunDate,3,-3,0,0,0,0x0,0x0
0,LS_A2D0,3,-2,2,0,0,0x0,0x0
0,LS_A2D1,3,-1,4,0,0,0x0,0x0
0,LS_A2D2,3,0,6,0,0,0x0,0x1b
0,LS_A2D3,3,1,8,0,0,0x6,0xc
0,LS_A2D4,3,2,10,0,0,0xf,0x4b
0,LS_A2D5,3,3,12,0,0,0x12,0x6c
0,LS_A2D6,3,4,14,0,0,0x15,0x93
0,LS_A2D7,3,5,16,0,0,0x18,0xc0
0,LS_A2D8,3,6,18,0,0,0x1b,0xf3
0,LS_A2D9,3,7,20,0,0,0x1e,0x30
0,LS_A2D10,3,8,22,0,0,0x21,0x16b
0,LS_A2D11,3,9,24,0,0,0x24,0x1fb
0,LS_A2D12,3,10,26,0,0,0x27,0x1fb
0,LS_A2D13,3,11,28,0,0,0x2a,0x24c
0,LS_A2D14,3,12,30,0,0,0x2d,0x2a3
0,LS_A2D15,3,13,32,0,0,0x30,0x300
0,LS_LDROP_MM,3,14,34,0,0,0x33,0x363
0,LS_RDROP_MM,3,15,36,0,0,0x36,0x3cc
0,LS_CLEANTYPE,3,16,38,0,0,0x39,0x43b
0,LS_ERROR_BRUSH_OVERTEMP,3,17,40,0,0,0x3c,0x4b0
0,LS_ERROR_BATTERY_OVERTEMP,3,18,42,0,0,0x3f,0x52b
0,LS_ERROR_LWHEEL_STUCK,3,19,44,0,0,0x42,0x5ac
0,LS_ERROR_RWHEEL_STUCK,3,20,46,0,0,0x45,0x633
0,LS_ERROR_LDS_JAMMED,3,21,48,0,0,0x48,0x6c0
0,LS_ERROR_BRUSH_STUCK,3,22,50,0,0,0x4b,0x753
0,LS_ERROR_VACUUM_STUCK,3,23,52,0,0,0x4e,0x7ec
0,LS_ERROR_BATTERY_CRITICAL,3,24,54,0,0,0x51,0x88b
0,LS_ERROR_LDS_JAMMED,3,25,56,0,0,0x54,0x930
0,LS_ERROR_BATTERY_CRITICAL,3,26,58,0,0,0x57,0x92b
0,LS_ERROR_BATTERY_CRITICAL,3,27,60,0,0,0x5a,0x9dc
0,LS_ERROR_BATTERY_CRITICAL,3,28,62,0,0,0x5d,0x9ef
0,LS_ERROR_BATTERY_CRITICAL,3,29,64,0,0,0x60,0xc00
0,LS_ERROR_BATTERY_CRITICAL,3,30,66,0,0,0x63,0xc3b
0,LS_ERROR_BATTERY_CRITICAL,3,31,68,0,0,0x66,0xc6b
0,LS_ERROR_BATTERY_CRITICAL,3,32,70,0,0,0x69,0xc9b
0,LS_ERROR_BATTERY_CRITICAL,3,33,72,0,0,0x6c,0xcb0
0,LS_ERROR_BATTERY_CRITICAL,3,34,74,0,0,0x6f,0x10b
0,LS_ERROR_BATTERY_CRITICAL,3,35,76,0,0,0x72,0x10ec
0,LS_ERROR_BATTERY_CRITICAL,3,36,78,0,0,0x75,0x11d3
0,LS_ERROR_BATTERY_CRITICAL,3,37,80,0,0,0x78,0x12c0
0,LS_ERROR_BATTERY_CRITICAL,3,38,82,0,0,0x7b,0x15ab
0,LS_ERROR_BATTERY_CRITICAL,3,39,84,0,0,0x7e,0x15ac
0,LS_ERROR_BATTERY_CRITICAL,3,40,86,0,0,0x80,0x16b3
0,LS_ERROR_BATTERY_CRITICAL,3,41,88,0,0,0x84,0x17bb
0,LS_ERROR_BATTERY_CRITICAL,3,42,90,0,0,0x87,0x17cc
0,LS_ERROR_BATTERY_CRITICAL,3,43,92,0,0,0x8a,0x18cc
0,LS_ERROR_BATTERY_CRITICAL,3,44,94,0,0,0x8d,0x19e3
0,LS_ERROR_BATTERY_CRITICAL,3,45,96,0,0,0x90,0x1b00
0,LS_ERROR_BATTERY_CRITICAL,3,46,98,0,0,0x93,0x1c23
0,LS_ERROR_BATTERY_CRITICAL,3,47,100,0,0,0x96,0x1d4c
0,LS_ERROR_BATTERY_CRITICAL,3,48,102,0,0,0x99,0x1e7b
0,LS_ERROR_BATTERY_CRITICAL,3,49,104,0,0,0x9c,0x1fb0
0,LS_ERROR_BATTERY_CRITICAL,3,50,106,0,0,0x9f,0x20eb
0,LS_ERROR_BATTERY_CRITICAL,3,51,108,0,0,0xa2,0x222c
0,LS_ERROR_BATTERY_CRITICAL,3,52,110,0,0,0xa5,0x2373
0,LS_ERROR_BATTERY_CRITICAL,3,53,112,0,0,0xa8,0x24c0
0,LS_ERROR_BATTERY_CRITICAL,3,54,114,0,0,0xab,0x2613
0,LS_ERROR_BATTERY_CRITICAL,3,55,116,0,0,0xae,0x276c
0,LS_ERROR_BATTERY_CRITICAL,3,56,118,0,0,0xb1,0x28cb
0,LS_ERROR_BATTERY_CRITICAL,3,57,120,0,0,0xb4,0x2a30
0,LS_ERROR_BATTERY_CRITICAL,3,58,122,0,0,0xb7,0x2b9b
0,LS_ERROR_BATTERY_CRITICAL,3,59,124,0,0,0xb8,0x2d0c
0,LS_ERROR_BATTERY_CRITICAL,3,60,126,0,0,0xbd,0x2e83
0,LS_ERROR_BATTERY_CRITICAL,3,61,128,0,0,0xc0,0x3000
0,LS_ERROR_BATTERY_CRITICAL,3,62,130,0,0,0xc3,0x3183
0,LS_ERROR_BATTERY_CRITICAL,3,63,132,0,0,0xc6,0x330c
0,LS_ERROR_BATTERY_CRITICAL,3,64,134,0,0,0xc9,0x349b
0,LS_ERROR_BATTERY_CRITICAL,3,65,136,0,0,0xd6,0x3630
0,LS_ERROR_BATTERY_CRITICAL,3,66,138,0,0,0xcf,0x37cb
0,LS_ERROR_BATTERY_CRITICAL,3,67,140,0,0,0xd2,0x396c
0,LS_ERROR_BATTERY_CRITICAL,3,68,142,0,0,0xd5,0x3b13
0,LS_ERROR_BATTERY_CRITICAL,3,69,144,0,0,0xd8,0x3cc0
0,LS_ERROR_BATTERY_CRITICAL,3,70,146,0,0,0xe0,0x402c
0,LS_ERROR_BATTERY_CRITICAL,3,71,148,0,0,0xe1,0x41eb
0,LS_ERROR_BATTERY_CRITICAL,3,72,150,0,0,0xe2,0x43b0
0,LS_ERROR_BATTERY_CRITICAL,3,73,152,0,0,0xe4,0x43b0
```
### Command: GetMotors

**Description:** Get the diagnostic data for the motors.

**Usage:** GetMotors [Brush] [Vacuum] [LeftWheel] [RightWheel] [Laser] [Charger]

**Options:**

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brush</td>
<td>Return Brush Motor stats.</td>
</tr>
<tr>
<td>Vacuum</td>
<td>Return Vacuum Motor stats.</td>
</tr>
<tr>
<td>LeftWheel</td>
<td>Return LeftWheel Motor stats.</td>
</tr>
<tr>
<td>RightWheel</td>
<td>Return RightWheel Motor stats.</td>
</tr>
<tr>
<td>Laser</td>
<td>Return LDS Motor stats.</td>
</tr>
<tr>
<td>Charger</td>
<td>Return Battery Charger stats.</td>
</tr>
</tbody>
</table>

**Return Format:**

If no flags are specified, then all motors are reported on. Parameter,Value

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brush MaxPWM,65536</td>
<td>Brush_PWM,-858993460</td>
</tr>
<tr>
<td>Brush mVolts,1310</td>
<td>Brush_mVolts,1310</td>
</tr>
<tr>
<td>Brush Encoder,0</td>
<td>Brush_Encoder,0</td>
</tr>
<tr>
<td>Brush RPM,-858993460</td>
<td>Brush_RPM,-858993460</td>
</tr>
<tr>
<td>Vacuum MaxPWM,65536</td>
<td>Vacuum_PWM,-858993460</td>
</tr>
<tr>
<td>Vacuum mVolts,1310</td>
<td>Vacuum_mVolts,1310</td>
</tr>
<tr>
<td>Vacuum Encoder,0</td>
<td>Vacuum_Encoder,0</td>
</tr>
<tr>
<td>Vacuum RPM,52428</td>
<td>Vacuum_RPM,52428</td>
</tr>
<tr>
<td>LeftWheel MaxPWM,65536</td>
<td>LeftWheel_PWM,-858993460</td>
</tr>
<tr>
<td>LeftWheel mVolts,1310</td>
<td>LeftWheel_mVolts,1310</td>
</tr>
<tr>
<td>LeftWheel Encoder,0</td>
<td>LeftWheel_Encoder,0</td>
</tr>
<tr>
<td>LeftWheel PositionInMM,0</td>
<td>LeftWheel_PositionInMM,0</td>
</tr>
<tr>
<td>LeftWheel RPM,-13108</td>
<td>LeftWheel_RPM,-13108</td>
</tr>
<tr>
<td>RightWheel MaxPWM,65536</td>
<td>RightWheel_PWM,-858993460</td>
</tr>
<tr>
<td>RightWheel mVolts,1310</td>
<td>RightWheel_mVolts,1310</td>
</tr>
<tr>
<td>RightWheel Encoder,0</td>
<td>RightWheel_Encoder,0</td>
</tr>
<tr>
<td>RightWheel PositionInMM,0</td>
<td>RightWheel_PositionInMM,0</td>
</tr>
<tr>
<td>RightWheel RPM,-13108</td>
<td>RightWheel_RPM,-13108</td>
</tr>
<tr>
<td>Laser MaxPWM,65536</td>
<td>Laser_PWM,-858993460</td>
</tr>
<tr>
<td>Laser mVolts,1310</td>
<td>Laser_mVolts,1310</td>
</tr>
<tr>
<td>Laser Encoder,0</td>
<td>Laser_Encoder,0</td>
</tr>
<tr>
<td>Laser RPM,52428</td>
<td>Laser_RPM,52428</td>
</tr>
<tr>
<td>Charger MaxPWM,65536</td>
<td>Charger_PWM,-858993460</td>
</tr>
<tr>
<td>Charger mAH,52428</td>
<td>Charger_mAH,52428</td>
</tr>
</tbody>
</table>

### Command: GetSchedule

**Description:** Get the Cleaning Schedule. (24 hour clock format)

**Usage:** GetSchedule [Day <Day_value>]

**Options:**

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>


Day | Day of the week to get schedule for. Sun=0, Sat=6. If not specified, then all days are given.
--- | ---
Return Format:
Schedule is Enabled Sun 00:00 - None - Mon 00:00 - None - Tue 00:00 R Wed 00:00 R Thu 00:00 R Fri 00:00 H Sat 00:00 H

Command: **GetSysLog**
Description: Get System Log data.
Usage: GetSysLog
Options: None
Return Format:
(Unimplemented) Sys Log Entries: Run, Stat, Min, Max, Sum, Count, Time(ms)

Command: **GetTime**
Description: Get Current Scheduler Time.
Usage: GetTime
Options: None
Return Format:
DayOfWeek HourOf24:Min:Sec Example: Sunday 13:57:09

Command: **GetVersion**
Description: Get the version information for the system software and hardware.
Usage: GetVersion
Options: None
Return Format:
Component, Major, Minor, Build ModelID, 0, XV11, ConfigID, 1,, Serial Number, AAannonnAA, 0000000, D Software, 2, 1, 15499 BatteryType, 1, NIMH_12CELL, BlowerType, 1, BLOWER ORIG, BrushSpeed, 0,, BrushMotorType, 1, BRUSH_MOTOR ORIG, SideBrushType, 1, SIDE_BRUSH_NONE, WheelPodType, 1, WHEEL POD ORIG, DropSensorType, 1, DROP_SENSOR ORIG, MagSensorType, 1, MAG_SENSOR ORIG, WallSensorType, 1, WALL_SENSOR ORIG, Locale, 1, LOCALE_USA, LDS Software, V1.0.0,, LDS Serial, XXX-YY,, LDS CPU, F2802x/cd00,, MainBoard Vendor ID, 1,, MainBoard Serial Number, 99,, MainBoard Version, 15, 0, ChassisRev, -1,, UIPanelRev, -1,

Command: **GetWarranty**
Description: Get the warranty validation codes.
Usage: GetWarranty
Options: None
Return Format:
### Command: **Help**

**Description:** Without any argument, this prints a list of all possible cmds. With a command name, it prints the help for that particular command

**Usage:** Help [Cmd <Cmd_value>]

**Options:**

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cmd</td>
<td>(Optional) Next argument is command to show help for. If Cmd option not used, help gives list of all commands.</td>
</tr>
</tbody>
</table>

**Return Format:**

'Help' Generates: Help - Without any argument, this prints a list of all possible cmds. With a command name, it prints the help for that particular command Clean - Starts a cleaning by simulating press of start button. DiagTest - Executes different test modes. Once set, press Start button to engage. (Test modes are mutually exclusive.) GetAccel - Get the Accelerometer readings. GetAnalogSensors - Get the A2D readings for the analog sensors. GetButtons - Get the state of the UI Buttons. GetCalInfo - Prints out the cal info from the System Control Block. GetCharger - Get the diagnostic data for the charging system. GetDigitalSensors - Get the state of the digital sensors. GetErr - Get Error Message. GetLDSScan - Get scan packet from LDS. GetLifeStatLog - Get All Life Stat Logs. GetMotors - Get the diagnostic data for the motors. GetSchedule - Get the Cleaning Schedule. (24 hour clock format) GetSysLog - Get System Log data. GetTime - Get Current Scheduler Time. GetVersion - Get the version information for the system software and hardware. GetWarranty - Get the warranty validation codes. PlaySound - Play the specified sound in the robot. RestoreDefaults - Restore user settings to default. SetDistanceCal - Set distance sensor calibration values for min and max distances. SetFuelGauge - Set Fuel Gauge Level. SetMotor - Sets the specified motor to run in a direction at a requested speed. (TestMode Only) SetTime - Sets the current day, hour, and minute for the scheduler clock. SetLED - Sets the specified LED to on, off, blink, or dim. (TestMode Only) SetLCD - Sets the LCD to the specified display. (TestMode Only) SetLDSRotation - Sets LDS rotation on or off. Can only be run in TestMode. SetSchedule - Modify Cleaning Schedule. SetSystemMode - Set the operation mode of the robot. (TestMode Only) SetWallFollower - Enables/Disables wall follower TestMode - Sets TestMode on or off. Some commands can only be run in TestMode. Upload - Uploads new program to the robot.  'Help Clean' Generates: Clean - Starts a cleaning by simulating press of start button. House - (Optional) Equivalent to pressing 'Start' button once. Starts a house cleaning. (House cleaning mode is the default cleaning mode.) Spot - (Optional) Starts a spot clean. Stop - Stop Cleaning.

---

### Command: **PlaySound**

**Description:** Play the specified sound in the robot.

**Usage:** PlaySound [SoundID <SoundID_value>] [Stop]

**Options:**

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SoundID</td>
<td>Play the sound library entry specified by the number in the next argument. Legal values are:</td>
</tr>
</tbody>
</table>
0 – Waking Up
1 – Starting Cleaning
2 – Cleaning Completed
3 – Attention Needed
4 – Backing up into base station
5 – Base Station Docking Completed
6 – Test Sound 1
7 – Test Sound 2
8 – Test Sound 3
9 – Test Sound 4
10 – Test Sound 5
11 – Exploring
12 – ShutDown
13 – Picked Up
14 – Going to sleep
15 – Returning Home
16 – User Canceled Cleaning
17 – User Terminated Cleaning
18 – Slipped Off Base While Charging
19 – Alert
20 – Thank You

Stop
Stop playing sound.

Return Format: None

Command: RestoreDefaults
Description: Restore user settings to default.
Usage: RestoreDefaults
Options: None
Return Format: None

Command: SetDistanceCal
Description: Set distance sensor calibration values for min and max distances.
Usage: SetDistanceCal [DropMinimum] [DropMiddle] [DropMaximum] [WallMinimum] [WallMiddle] [WallMaximum]
Options:

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DropMinimum</td>
<td>Take minimum distance drop sensor readings. Mutually exclusive of DropMiddle and DropMax.</td>
</tr>
<tr>
<td>DropMiddle</td>
<td>Take middle distance drop sensor readings. Mutually exclusive of DropMinimum and DropMax.</td>
</tr>
<tr>
<td>DropMaximum</td>
<td>Take maximum distance drop sensor readings. Mutually exclusive of DropMinimum and DropMiddle.</td>
</tr>
<tr>
<td>WallMinimum</td>
<td>Take minimum distance wall sensor readings. Mutually exclusive of WallMiddle and WallMax.</td>
</tr>
<tr>
<td>WallMiddle</td>
<td>Take middle distance wall sensor readings. Mutually exclusive of WallMiddle and WallMax.</td>
</tr>
</tbody>
</table>
WallMaximum | WallMinimum and WallMax.  
| | Take maximum distance wall sensor readings. Mutually exclusive of WallMinimum and WallMiddle.

**Return Format:**

```
Label,Value RDropCalA2DMin,-1 RDropCalA2DMid,-1 RDropCalA2DMax,-1 LDropCalA2DMin,-1 LDropCalA2DMid,-1 LDropCalA2DMax,-1 WallCalA2DMin,-1 WallCalA2DMid,-1 WallCalA2DMax,-1
```

---

**Command:** **SetFuelGauge**

**Description:** Set Fuel Gauge Level.

**Usage:** 

```
SetFuelGauge [Percent <Percent_value>]
```

**Options:**

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percent</td>
<td>Fuel Gauge percent from 0 to 100</td>
</tr>
</tbody>
</table>

**Return Format:** None

---

**Command:** **SetLCD**

**Description:** Sets the LCD to the specified display. (TestMode Only)

**Usage:** 

```
SetLCD [BGWhite] [BGBlack] [HLine <HLine_value>] [VLine <VLine_value>] [HBars] [VBars] [FGWhite] [FGBlack] [Contrast <Contrast_value>]
```

**Options:**

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BGWhite</td>
<td>Fill LCD background with White</td>
</tr>
<tr>
<td>BGBBlack</td>
<td>Fill LCD background with Black</td>
</tr>
<tr>
<td>HLine</td>
<td>Draw a horizontal line (in foreground color) at the following row.</td>
</tr>
<tr>
<td>VLine</td>
<td>Draw a vertical line (in foreground color) at the following column.</td>
</tr>
<tr>
<td>HBars</td>
<td>Draw alternating horizontal lines (FG,BG,FG,BG,...), across the whole screen.</td>
</tr>
<tr>
<td>VBars</td>
<td>Draw alternating vertical lines (FG,BG,FG,BG,...), across the whole screen.</td>
</tr>
<tr>
<td>FGWhite</td>
<td>Use White as Foreground (line) color</td>
</tr>
<tr>
<td>FGBBlack</td>
<td>Use Black as Foreground (line) color</td>
</tr>
<tr>
<td>Contrast</td>
<td>Set the following value as the LCD Contrast value into NAND. 0..63</td>
</tr>
</tbody>
</table>

**Return Format:** None

---

**Command:** **SetLDSRotation**

**Description:** Sets LDS rotation on or off. Can only be run in TestMode.

**Usage:** 

```
SetLDSRotation [On] [Off]
```

**Options:**
## Command: SetLED

**Description:** Sets the specified LED to on, off, blink, or dim. (TestMode Only)

**Usage:**
```
SetLED [BacklightOn] [BacklightOff] [ButtonAmber] [ButtonGreen] [LEDRed] [LEDGreen] [ButtonAmberDim] [ButtonGreenDim] [ButtonOff]
```

**Options:**

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BacklightOn</td>
<td>LCD Backlight On (mutually exclusive of BacklightOff)</td>
</tr>
<tr>
<td>BacklightOff</td>
<td>LCD Backlight Off (mutually exclusive of BacklightOn)</td>
</tr>
<tr>
<td>ButtonAmber</td>
<td>Start Button Amber (mutually exclusive of other Button options)</td>
</tr>
<tr>
<td>ButtonGreen</td>
<td>Start Button Green (mutually exclusive of other Button options)</td>
</tr>
<tr>
<td>LEDRed</td>
<td>Start Red LED (mutually exclusive of other Button options)</td>
</tr>
<tr>
<td>LEDGreen</td>
<td>Start Green LED (mutually exclusive of other Button options)</td>
</tr>
<tr>
<td>ButtonAmberDim</td>
<td>Start Button Amber Dim (mutually exclusive of other Button options)</td>
</tr>
<tr>
<td>ButtonGreenDim</td>
<td>Start Button Green Dim (mutually exclusive of other Button options)</td>
</tr>
<tr>
<td>ButtonOff</td>
<td>Start Button Off</td>
</tr>
</tbody>
</table>

**Return Format:** None

## Command: SetMotor

**Description:** Sets the specified motor to run in a direction at a requested speed. (TestMode Only)

**Usage:**
```
SetMotor [LWheelDist <LWheelDist_value>] [RWheelDist <RWheelDist_value>] [Speed <Speed_value>] [Accel <Accel_value>] [RPM <RPM_value>] [Brush] [VacuumOn] [VacuumOff] [VacuumSpeed <VacuumSpeed_value>] [RWheelDisable] [LWheelDisable] [BrushDisable] [RWheelEnable] [LWheelEnable] [BrushEnable]
```

**Options:**

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LWheelDist</td>
<td>Distance in millimeters to drive Left wheel. (Pos = forward, neg = backward)</td>
</tr>
<tr>
<td>RWheelDist</td>
<td>Distance in millimeters to drive Right wheel. (Pos = forward, neg = backward)</td>
</tr>
<tr>
<td>Speed</td>
<td>Speed in millimeters/second. (Required only for wheel movements)</td>
</tr>
<tr>
<td>Accel</td>
<td>Acceleration in millimeters/second. (Used only for wheel movements. Defaults to 'Speed'.)</td>
</tr>
<tr>
<td>RPM</td>
<td>Next argument is the RPM of the motor. Not used for wheels, but applied to all other motors specified in the command line.</td>
</tr>
<tr>
<td>Brush</td>
<td>Brush motor forward (Mutually exclusive with wheels and vacuum.)</td>
</tr>
</tbody>
</table>
VacuumOn | Vacuum motor on (Mutually exclusive with VacuumOff)
VacuumOff | Vacuum motor off (Mutually exclusive with VacuumOn)
VacuumSpeed | Vacuum speed in percent (1-100).
RWheelDisable | Disable Right Wheel motor
LWheelDisable | Disable Left Wheel motor
BrushDisable | Disable Brush motor
RWheelEnable | Enable Right Wheel motor
LWheelEnable | Enable Left Wheel motor
BrushEnable | Enable Brush motor

**Return Format:** None

---

**Command:** SetSchedule

**Description:** Modify Cleaning Schedule.

**Usage:**

```
SetSchedule [Day <Day_value>] [Hour <Hour_value>] [Min <Min_value>] [House] [None] [ON] [OFF]
```

**Options:**

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Day</td>
<td>Day of the week to schedule cleaning for. Sun=0, Sat=6. (required)</td>
</tr>
<tr>
<td>Hour</td>
<td>Hour value 0..23 (required)</td>
</tr>
<tr>
<td>Min</td>
<td>Minutes value 0..59 (required)</td>
</tr>
<tr>
<td>House</td>
<td>Schedule to Clean whole house (default) (Mutually exclusive with None)</td>
</tr>
<tr>
<td>None</td>
<td>Remove Scheduled Cleaning for specified day. Time is ignored. (Mutually exclusive with House)</td>
</tr>
<tr>
<td>ON</td>
<td>Enable Scheduled cleanings (Mutually exclusive with OFF)</td>
</tr>
<tr>
<td>OFF</td>
<td>Disable Scheduled cleanings (Mutually exclusive with ON)</td>
</tr>
</tbody>
</table>

**Return Format:**

```
Schedule is Enabled Sun 00:00 - None - Mon 00:00 - None - Tue 00:00 R Wed 00:00 R Thu 00:00 R Fri 00:00 H Sat 00:00 H
```
**Standby**
Start standby operation. (mutually exclusive of other options)

**Return Format:** None

---

**Command:** **SetTime**

**Description:** Sets the current day, hour, and minute for the scheduler clock.

**Usage:** `SetTime [Day <Day_value>] [Hour <Hour_value>] [Min <Min_value>] [Sec <Sec_value>]`

**Options:**

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Day</td>
<td>Day of week value Sunday=0, Monday=1, ... (required)</td>
</tr>
<tr>
<td>Hour</td>
<td>Hour value 0..23 (required)</td>
</tr>
<tr>
<td>Min</td>
<td>Minutes value 0..59 (required)</td>
</tr>
<tr>
<td>Sec</td>
<td>Seconds value 0..59 (Optional, defaults to 0)</td>
</tr>
</tbody>
</table>

**Return Format:** None

---

**Command:** **SetWallFollower**

**Description:** Enables/Disables wall follower

**Usage:** `SetWallFollower [Enable] [Disable]`

**Options:**

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable</td>
<td>Enable wall follower</td>
</tr>
<tr>
<td>Disable</td>
<td>Disable wall follower</td>
</tr>
</tbody>
</table>

**Return Format:** None

---

**Command:** **TestMode**

**Description:** Sets TestMode on or off. Some commands can only be run in TestMode.

**Usage:** `TestMode [On] [Off]`

**Options:**

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>On</td>
<td>Turns Testmode on. Mutually exclusive with Off.</td>
</tr>
<tr>
<td>Off</td>
<td>Turns Testmode off. Mutually exclusive with On.</td>
</tr>
</tbody>
</table>

**Return Format:** None

---

**Command:** **Upload**

**Description:** Uploads new program to the robot.

**Usage:** `Upload [dump] [code] [sound] [LDS] [xmodem] [size <size_value>] [noburn] [readflash] [reboot]`

**Options:**

<table>
<thead>
<tr>
<th>Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dump</td>
<td>Dump the contents of the upload save area.</td>
</tr>
<tr>
<td>code</td>
<td>Upload file is the main application. (Mutually exclusive with sound, LDS)</td>
</tr>
<tr>
<td>-------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>sound</td>
<td>Upload file is a sound module. (Mutually exclusive with code, LDS)</td>
</tr>
<tr>
<td>LDS</td>
<td>Upload file is an LDS module. (Mutually exclusive with sound, code)</td>
</tr>
<tr>
<td>xmodem</td>
<td>Use xmodem protocol when transmitting upload module</td>
</tr>
<tr>
<td>size</td>
<td>data size to upload to device.</td>
</tr>
<tr>
<td>noburn</td>
<td>test option -- do NOT burn the flash after the upload.</td>
</tr>
<tr>
<td>readflash</td>
<td>test option -- read the flash at the current region.</td>
</tr>
<tr>
<td>reboot</td>
<td>Reset the robot after performing the upload.</td>
</tr>
</tbody>
</table>

**Return Format:** None