Openmag User's Guide

Sean D'Epagnier

1

Published by Sean D'Epagnier

Table of Contents

1 Software

1.1 Installation

1. Windows OS – There is a binary distribution for windows. This is a zip (openmag_win32.zip) which contains all programs and drivers. The device can only function as one of: mouse, joystick, or cdc device. This limitation is because windows is broken. The best I can offer is to choose one device at a time. The cdc device is used for configuration, to force this mode, hold the second (right) button down when plugging in the device. This is only for the pointer model, the survey model is always in cdc mode. Windows will prompt for a driver the first time. Manually specify the downloaded folder containing the file OpenMag.inf. Once the driver is installed, you will be able to communicate via serial to the device.

   Hyperterminal Setup – Normally you will use the dataclient, but if you want to use hyperterminal for some reason: go to File->Properties, click on Settings, then ASCII Setup. Check "Send line ends with line feeds" and "Echo typed characters locally"
2. Other OS – Most common OS's support usb mouse, joystick, and cdc devices. This means you do not need to do anything special and no special driver is needed. For the serial device you should get a virtual comm port device, on linux it is /dev/ttyACM0 (the kernel module is cdc-acm), on freebsd it is /dev/ttyU0 (the kernel module is umodem). You will need Qt 4.3.0 or better to run DataViewer.

1.2 DataClient

Once configured the device will show up as a comm port. This means you can communicate to the device with any terminal program (eg hyperterminal)

The dataclient is the console version of the interface to device. The gui version is dataviewer in the next section. The dataclient program provides a more interactive user interface than a basic serial terminal. The main features are lineediting and completion via readline, data and reply separation (stdout and stderr), and the ability to change directories using cd, as well as see the directory via pwd.

The datainterface that the dataclient provides direct access to has two important concepts:

- operators – commands that operate on accessors, eg: get, set, clear, ls, ops ...
- accessors – data on the device which can be accessed eg: softwareversion

1.2.1 Example Session

To connect to a device, fifo or file:

```
./dataclient <file>
```

To connect to a server

```
./dataclient <host:port>
```

Once successfully connected you should see the prompt

```
$->
```

To autodetect the device, run the dataclient with no arguements.

It is now possible to execute commands and query data. Many of the commands are similar to unix commands. To list all possible commands issue “allops”

```
$-> allops
type ops ls allops mem get set values clear
```

Note: operators may be added or removed depending on software version

To list the accessors, issue “ls”

```
$-> ls
mouse/ joystick/ softwareversion settings/ stats/ calc/ calibration/ sensors/
```

Items with a trailing / are directories. To list their contents, you may either cd into them, or ls them

```
$-> ls stats
freeram runtime mainloopfreq watchdog_resets
```

It should be apparent that “stats” is an accessor as is “stats/runtime”. For example you may “get stats/runtime”.

```
$-> get stats/runtime
51.81s
$-> cd stats/
stats/ $-> get runtime
54.54s
```

Not all accessors support all operators. To see which operators are supported, use the ops operator which is always supported

```
$-> ops softwareversion
type mem ops get
```

This means it is valid to replace “ops” with any of the valid operators in the above command. Notice that you cannot “set” softwareversion.

Whenever the sensors are set up to automatically output data, this streaming data appears on stderr from dataclient.

For example, you may log data

```
sean@sun ~ $ ./dataclient 2> log
$-> set sensors/accel/outputrate 10
$-> [ctrl-D to exit]
sean@sun ~ $ cat log
accel: 201 58 1273
accel: 204 65 1273
accel: 202 65 1271
...
```

This interface includes additional host side operators which work with the dataclient program, they will not work with a console program.

1.2.2 Host Side Commands

1. cd – change to a given directory if it exists, relative paths are supported, "cd ../../info" Lastdir is supported as well eg: "cd -"
2. pwd – display the current directory

It is recommended to run “dataclient --help” to show all the capabilites of dataclient.

1.3 DataViewer

The DataViewer is a graphical application used to query and interact with the device while it is running. When it is first run, you should see a tree view, and below it a console and an output window.

Hit populate to automatically completely query the device. This may take a few seconds. Now it is possible to view all of the data stored on the device. If you press “Get Values”, it will re-request just the values. This is useful because many of the values update continuously. You may also check certain values and only request those values. It is possible to modify certain values. The ones marked “write only” or “read/write” can be modified. For values with only certain possible settings, a dropdown is provided.

The Console window displays the actual data being sent and received to the device. The dataclient is actually running and doing all of the communication with the device, the dataviewer communicates with the dataclient. There may be operations that can only be performed from the console, for this you will have to run the dataclient.

The Output window displays streaming data comming from the device.

2 DeviceData

This chapter describes all of the data available on the device. It assumes use of the data-client. It should be obvious how to use the DataViewer as well. Each section begins with the name of the directory.

The openmag project operates an accelerometer, magnetometer, and temperature sensor. If the device is to be used while moving, the calculated pitch, roll, and yaw are invalid due to lack of gyro compensation. If the down vector is not known, the magnetic vector output is still useful as it is not affected by acceleration, but yaw calculation is impossible.

The accelerometers are in sensor coordinates, the calibrated magnetometer is aligned to the accelerometer, so both calibrated outputs are in sensor coordinates. Box coordinates are another coordinate frame which can be defined by the user, there is a rotation to get from sensor to box coordinates. There is also a rotation to get from sensor to laser coordinates (only on the survey device)

On some devices not all of these accessors will be visible, for example the pointer device does not include the survey directory because it cannot take surveys, and the survey device does not include the mouse or joystick directory.

Do not be alarmed if you do not have all of the accessors listed, or you have accessors which are not listed here.

2.1 root

This directory contains all of the other directories. It contains the "softwareversion" accessor for determining what software version you are using. When you first connect to the device, you are in the root directory.

2.2 mouse

- enableshakewheel – if set, then shaking the mouse can scroll up and down
- orientation – set which axis is used for movement, default is x
- horizontalsensitivity – the sensitivity for horizontal movement, set to negative to reverse
- vertical;sensitivity – the sensitivity for vertical movement, set to negative to reverse
- scrollsensitivity – sensitivity for scrolling

2.3 joystick

The device can function as a 6 axis 3 button joystick.

- rotation

  The rotation allows for an arbitrary rotation to be entered by the user for joystick data. If you enter 0 for the angle, then there is no rotation. This allows the joystick to work in arbitrary coordinates.

  - angle – angle in degrees to rotate coordinates
  - x – x component of vector to rotate around
  - y – y component of vector to rotate around
  - z – z component of vector to rotate around

- axis0-axis5 – the axis of joystick to setup
  - input – what this axis gets input from, can be roll, pitch, yaw, x, y, or z. Pitch is the angle between the x axis and the currently measured data. Roll is the angle rotated around the x axis. Yaw uses the other sensor to tilt-compensate an angle.
  - sensor – which sensor to use? accel, mag, or none to disable
  - sensitivity – increase to make more sensitive, make negative to reverse direction.

2.4 settings

- usbcdcenabled – the usb device acts like a virtual com port cdc device
- usbmouseenabled – the usb device acts like a mouse
- usbjoystickenabled – the usb device acts like a joystick
- uartenabled – the uart works like a serial port for data
- uartbaudrate – the baud rate of the uart, (default 38400)
- button0outputsinfo – if set the pitch, roll, yaw, dip, incline, and asmith are reported when the first button is pressed
- truenorth – if set the declination is subtracted from the yaw

Note: Do not enable more than one of usbcdcenabled, usbmouseenabled, or usbjoystickenabled if you are connected to a computer running windows.

Enabling mouse and especially joystick makes the device run slower which will slow down the autocalibration algorithms as well as reduce sensor input rates. Disable these modes if you are not using them.

2.5 stats

The accessors relevant:

- freeram - unused ram from the total of 8192 bytes, useful for debugging.
- runtime - how long the device has been running since powerup in seconds
- mainloopfreq - how fast the main loop is running in hz, very useful for profiling.
- watchdog_resets - If the device crashes or locks up, after a timeout it automatically resets, but this counter gets incremented. If this value is ever non-zero, then a software bug has occured.

2.6 power

- battery
  - voltage - the battery voltage
  - chargedvoltage - the voltage of the battery when it was last charged
  - dischargedvoltage - the voltage of the battery when it was last discharged
  - state - the current state of the battery
- stillsleeptype - sleep type to use when the device is still
- stillsleeptimeout - time the device must be still before sleeping
- buttonidlesleeptype - sleep type to use when no buttons are pressed

- buttonidletimeout - time the device must not have a button pressed before sleeping
- wakeonmotion - if set to true, the device will wake on motion when it sleeps from being still. It will not wake from motion from a button idle timeout.
- lastsleeptype - the last type of sleep the device was in, used for debugging

Note: the device does not sleep when plugged in via USB.

The sleep types supported:

- none - sleep disabled
- idle - sensors and processor stopped, device consumes about 15% of normal power. Can wake up from the most different ways.
- powersave - shutdown everything possible while still keeping the LCD and backlight on (if available)
- powerdown - shutdown everything to lowest possible power consumption. Uses about 0.5% of normal operating power.

Note: If wakeonmotion is set, because extra power is needed to be able to wake from motion, and the device sleeps from being still, the device will consume about 1.5% of normal operating power in powerdown mode rather than 0.5%.

2.7 calc

- pitch - the angle the box x-axis is pointed up or down, from -90 to 90.
- roll - the rotation around the box x-axis, from -180 to 180.
- yaw - the heading of the box x-axis, from 0-360. 0 is north, 180 is south.
- dip - the measured inclination angle between the magnetic vector and acceleration vector. This is related to your geographic magnitic latitude and should be relatively constant when well calibrated and the device is not moving.
- incline - the angle between the laser axis and horizontal.
- asmith - the direction the laser axis is pointed, like yaw.
- boxalignedaccel - the calibrated accelerometer with the boxalignment applied, this is used to calculate pitch and roll
- boxalignedmag - the calibrated magnetomtermeter with the boxalignment applied, this along with boxaligned accel is used to calculate yaw and dip

2.8 calibration

- inclination – the inclination, can be set by the user to speed up calibration
- declination – the angle between magnetic and true north
- fieldstrength – the strength of the field in gauss, used to make the calibrated magnetometer output correct in gauss
- boxalignment – the rotation from sensor coordinates to box coordinates
- accel – Computes biases and scale factor for the accelerometer
- mag – Computes biases, scale factors, and cross-coupling coefficients

- magalign – computes dip angle as well as the rotation from magnetometer to accelerometer coordinates. The rotation is displayed as a vector to rotate around, where the magnitude is the angle to rotate.
- magfast – fast mag calibration which only computes bias and scale factor. This is used by mouse mode. The calibration is calculated quickly without being as precise and does not require the device to stop moving.

There are 4 types of autocalibration:

- accel - used for accelerometer, need accurate calibration to compute pitch and roll
- mag - magnetometer, needed for accurate magnetometer readings
- magalign - this requires both accel and mag to calculate, it is the misalignment of the magnetic sensors in relation to the accelerometers. It corrects errors in yaw calculation by as much as 5 degrees.
- magfast are - this just computes biases for the magnetometer quickly, it is used for mouse movement because the more precise magnetometer calibration updates slower, can be used when relative movement is better than absolute. Each type is a directory with:
- calibration - the current calibration, you may clear it with the clear operator
- autocalenabled - specifies if true, calibration can continously update
- debugging - enable this to automatically output calibration when it is updated

2.9 sensors

The mag also provides

- range - The ADC which reads from the magnetic sensors supports variable gain. This means that with reduced precision, you can measure over a larger range. The range for the magnetometer, can be one of 6.4, 3.2, 1.6, or 0.8. This is the approximate range in gauss of the magnetic field that can be measured, the actual range that can be measured without saturation varies from unit to unit depending on the bias of the sensor.
- saturated - gives status on which channels (if any) of the magnetometer saturated.
- bandwidth - The magnetometer can be read at different speeds. This is apparent when looking at the inputrate and noise of the mag after setting the bandwidth. The faster data is read, the higher the noise is. Currently there are 3 settings: fast (64hz), normal (13hz), and slow (4.5hz)

The temperature sensor provides

- outputtype - can be set to F, C, or raw. This specifies the units to use for temperature.
- output - get this to read the temperature
- autooutput - if set to true the temperature will continously output

2.10 survey

The device can store various measurements

- stats - get for statistics about the measurements
- measurements - get to dump all of the measurements

3 Calibration

One of the key features is autocalibration. The device may not come pre-calibrated, so for precise measurements, the user should understand how to verify calibration. If you are interested in how the calibration works in more detail, see the Calibration document. The device computes the unknown calibration coefficients needed to deliver useful data. There are 4 types of calibration performed:

3.1 StillPoints

The device heavily uses stillpoints for calibration. What this means, is it can detect when it is not moving by looking at sensor noise levels. Once the unit is "still" the raw sensor data can be used in a meaningful way to calibrate the sensors. It is essential to place the device in various orientations and hold it perfectly still for 1-3 seconds. It is important to cover all possible orientations. To make it easier to make sure you do it correctly, I recommend each of the 6 sides in 2-3 rotations on each side. 10 positions is minimum, 12-15 positions will give better results.

3.2 Verifying Calibration

The accelerometer only estimates 4 unknowns (or 7 if calibration over temperature is enabled) This means that it automatically calibrates as soon as you hold the device still for a second or so in a few orientations. Here is an example session:

```
$-> set calibration/accel/debugging true
$-> clear calibration/accel/calibration
$-> accel bias: (-1.428 18.66 3.059)
accel magnitude 1.33e+03
accel bias: (132.2 65.65 -202.8)
accel magnitude 1.33e+03
accel bias: (260.5 64.23 -109.1)
accel magnitude 1.33e+03
accel bias: (295.7 68.98 -86.69)
accel magnitude 1.33e+03
accel bias: (302.5 70.23 -87.32)
accel magnitude 1.33e+03
accel bias: (287.9 77.5 -111.9)
accel magnitude 1.33e+03
accel bias: (289.2 79.17 -112.6)
accel magnitude 1.33e+03
accel bias: (290.3 82.19 -111.2)
accel magnitude 1.33e+03
accel bias: (290.4 82.08 -115.1)
accel magnitude 1.33e+03
accel bias: (291 81.71 -116.6)
accel magnitude 1.33e+03
accel bias: (290 81.95 -118.3)
accel magnitude 1.33e+03
```

As you can see, the bias changes less as it approaches the true value. The calibration updates when you move the device to a new orientation and hold it still.

The magnetometer is more complicated to calibrate. It estimates 9 unknowns, as well as the alignment between the magnetometer and the accelerometer (4 more unknowns). For best results, change the bandwidth of the magnetometer to either slow or medium. When you turn mag debugging on:

```
$-> set calibration/mag/debugging true
$-> clear calibration/mag/calibration
```

There are two updates, the first gives you Residules

```
mag residule: -0.814864
mag bias: (5.76e+03 -1.91e+04 1.61e+04)
mag magnitude ratios: [1 1]
mag cross coupling: {0.00111 -0.0011 0.00363}
mag magnitude: 3.68e+04
```

The closer the residule is to zero, the more correct the calibration is.

The other type of update is the Still update. You need a minimum of 10 good readings before you will get this update. Once you get a still update, the deviation should be much better, and calibration is more complete.

```
mag still dev: 0.010449
mag bias: (3.39e+04 -2.03e+04 1.38e+04)
mag magnitude ratios: [0.97 1.07]
mag cross coupling: {-0.109 -0.00378 0.0655}
mag magnitude: 4.52e+04
```

A deviation of .01 makes the device accurate to roughly .5 degrees for yaw. You can typically get this with 11-14 orientations. This deviation is the same units as the residule for the normal update. As you can see the deviation is very low once a still update occurs. Notice that the magnitude ratios are well estimated, this typically doesn't occur until a still update. It is important to get a still update before using the device for accurate measurements.

If you have calibrated in a distortion-free area, and want to lock the calibration so that moving through areas with distortions will not alter the calibration, disable autocalibration:

```
$-> set calibration/mag/autocalenabled false
```

You can monitor the magalign and magfast calibration the same way as the above. The accel must have decent calibration for pitch and roll to be accurate. The accel, mag, and magalign calibrations all must have decent calibration before the yaw calculation is accurate.

If the device moves in or out of a distortion, it may take it a while to realize what happened, eventually it should recover.

3.3 Alignment

Once calibration for the sensors is performed, accelerometer and magnetometer vectors are available in the sensor coordinate system. This coordinate system typically has some arbitrary rotation away from the desired coordinate system.

There are two alignments performed. One rotates from sensor coordinates into box coordinates. Box coordinates are used to calculate pitch, roll, yaw and Dip (the current measured inclination). The other rotation rotates into laser coordinates. Unlike box coordinates, laser coordinates do not havfe a concept of roll since the alignment is to an axis (laser or sight). Laser coordinates provide incline and asmith which are like pitch and yaw except they use laser coordinates not box coordinates.

You may perform alignments external to the device with your own software, in this case you might as well read from the sensors directly.

To set the box alignment there are two options.

- if you know magnetic north, place the device level and facing magnetic north:

  ```
  set boxalignment 1
  ```

  Now the boxalignment is set
- If you do not know magnetic north, then place the device level:

  ```
  set boxalignment 2
  ```

  Next place the device pointed straight up (against a wall should work):

  ```
  set boxalignment 3
  ```

If you want to reset boxalignment to no rotation.

```
clear boxalignment
```

To set the laser alignment, you have to take shots around the laser axis. To take a shot, align through the laser axis to a known point and:

```
set laseralignment 1
```

repeat this command for at least 5 shots, or until the error is low enough.

If you make a bad shot, you will need to reset and start over. To reset the laser alignment:

```
clear laseralignment
```

4 Menu

The menu runs directly only on the survey device model. It can be run under Emulation for all devices, see the Chapter 5 [Emulation], page 13 chapter for details.

The menu on the device is simple, press the key of the number letter or symbol which next to the word or icon you want to perform. The main menu allows you to select a function for the device. It is always possible to press 'D' repeatedly to eventually return to the Main Menu. It is usually possible to press 'A' to read information about the current page.

4.1 Compass

The compass shows a 3d compass as well as other statistics.

4.2 Survey

The survey system allows you to take measurements of incline and asmith using the laser. These measurements may be recorded. Typically there is enough storage for at least 500 shots. It is also possible to enter attributes to go with each measurement (such as distance between survey points) but not required.

Data required to store a shot:

- 5 bytes - incline and asmith
- 5 bytes - name and number (only needed if the name changes or the numbers are not sequential)
- 2 bytes - distance
- 4 bytes - lrud

Name	**incline**	**asmith**	**distance**	**lrud**	**bytes used**
AFF-001	+42.2	135.3			10
AFF-002	-26.7	167.4			5
AFF-003	+35.2	302.7	23.4		7
AFF-005	-42.1	253.9		1 1 2 3	14

The first measurement uses 5 extra bytes because there is no measurement before it. The fourth measurement uses 5 bytes normally, 4 more for lrud, and 5 more because the number is not sequential.

The measurements can be downloaded to a computer, or viewed on the device.

4.3 Pedometer

The pedometer uses a simple algorithm to detect steps by sensing verticle motion. It counts steps with a given tolerance.

4.4 Gaussmeter

The gaussmeter allows you to measure magnetic fields in gauss. There is support for absolute, relative and vector magnetometer modes.

4.5 Calibration

Proper calibration is extremely important for accurate measurements. The device can be re-calibrated at any time without any additional equipment. The on-screen help system explains what each step of calibration does. See the Chapter 3 [Calibration], page 8 chapter for more details.

4.6 Settings

There are a few settings that can be modified on the device, there are far more settings that can be modified via the dataclient.

5 Emulation

Emulation allows you to run the calibration algorithms, and the menu system on a pc computer rather than the device. If you had an embedded system running a regular OS talking to the device, it would be possible to run the algorithms on that system therefore offloading the floating point computations.

Note: Currently Emulation does not compile on win32.

The program “calibrationhost” can be run which runs a tcp server. Connect to this server using the dataclient.

```
./calibrationhost -q /dev/ttyACM0 &
[1] 12351
Listening on port 7029 for connections from telnet or dataclient
./dataclient localhost:7029
$-> ls
calc/ sensors/
```

The emulation provides the exact same interface. The device (/dev/ttyACM0) should previously be configured to output accel, mag, and temperature data. There is a script “runcalibrationhost.sh” which runs the calibrationhost program and sets up the device to output raw data correctly.

It is also possible to relay the data as a server:

```
./dataclient -p 3000 -q
Listening on port 3000 for connections from telnet or dataclient
$->
```

You may then connect to this dataclient with other dataclients from remote hosts on port 3000. These dataclients may in turn run as servers as well.

It is also possible to run the menu interface if you have opengl. This program “menuhost” automatically runs the calibration algorithms, so it is equivilant to calibrationhost, with the addition of the menu system.