

LCP-USB Inclinator sensor

DLL Interface library description

Description

The LCP-USB sensor connects to a USB host (PC) with a standard 4 pin USB A connector. It is USB 2.0 compatible.

The sensor samples the two inclination axes, digital filters them and outputs the readings calibrated in degrees.

A Windows dll is supplied to interface between the user application and the USB sensor. The functions available using the library are detailed in this document.

The sensor has a maximum USB message rate of 100 messages per second.

USB driver

The LCP-USB has a HID USB interface., and is supported by two Level Developments supplied dlls. These are HidLibrary.dll and LCPLibrary.dll.

HidLibrary.dll and LCPLibrary.dll must be installed in the application directory.

The user must reference LCPLibrary.dll in his project.

LCPLibrary.dll functions

The library has a single class called LCPDevice. In the users c# code for example, this is declared like this:

```
private static LCPDevice sensor = new LCPDevice();
```

And thereafter the functions are called using *sensor.Function()*;

For example, to get the X axis value:-

```
double XAngle;  
int XVal;  
  
XVal = sensor.Get_X();           // get scaled int from sensor  
  
if (XVal == -1000000)  
    // there is an error, do something..  
  
else    // value is good, rescale to floating point degrees  
    XAngle = (double)( XVal / 1000.0);
```

All functions which return an int, return the value -1000000 in the case of error. This indicates no response or timeout to the request sent to the sensor. Usually this is caused by the sensor not being connected.

List of functions

Bool Initialise(int VendorId, int ProductId)

Sets up the library and connects to the sensor using the Vendor ID and Product ID. For the LCP_USB, the Vendor ID is 0x04D8 and the Product ID is 0xEC73. If the operation is successful, true is returned, otherwise false.

Void Dispose()

Finishes operation with the sensor and closes the connection to the sensor tidily. This must be called when closing the user app.

int Get_Product

Reads the product code from the sensor. The LCP-USB sensor is code 2. This is not adjustable

int Get_Serial()

Reads the serial number from the sensor. Set at the factory, not adjustable.

int Get_X()

Reads the X Axis angle from the sensor in degrees, scaled by 1000. i.e. a value of 11234 equals 11.234 degrees

int Get_Y()

Reads the Y Axis angle from the sensor in degrees, scaled by 1000. i.e. a value of 11234 equals 11.234 degrees.

int Get_360()

Reads the 360 degree angle from the sensor in degrees, scaled by 1000. i.e. a value of 11234 equals 11.234 degrees.

int Get_Filter()

Reads the filter setting from the sensor.

Index	Frequency
0	0.125Hz
1	0.25Hz
2	0.5Hz
3	1.0Hz
4	2Hz
5	4Hz
6	8Hz
7	16Hz
8	32Hz

int Get_X_Direction()

Reads the X direction setting.

0 = normal

1 = reversed

int Get_Y_Direction()

Reads the Y direction setting.

0 = normal
1 = reversed

int Get_360_Direction()

Reads the 360 direction setting.

0 = normal
1 = reversed

int Get_Axes()

Reads the setting number of axes from the sensor.

1 = single 360 degree axis
2 = separate X and Y axes

void Set_Filter(byte index)

Sets the filter in the sensor.

Index	Frequency
0	0.125Hz
1	0.25Hz
2	0.5Hz
3	1.0Hz
4	2Hz
5	4Hz
6	8Hz
7	16Hz
8	32Hz

void Set_X_Direction(byte direction)

Sets the X axis direction

0 = normal
1 = reversed

void Set_Y_Direction(byte direction)

Sets the Y axis direction

0 = normal
1 = reversed

void Set_360_Direction(byte direction)

Sets the 360 axis direction

0 = normal
1 = reversed

void Set_Axes(byte n)

Sets the number of axes in the sensor

1 = single 360 degree axis

2 = separate X and Y axes

void Set_Zero(byte n)

Sets the zero point in the sensor

0 = absolute

1 - relative

Revision History:

1. Chris Wedgwood
Level Developments
4.9.2014

2. Jesper Sotikov
Level Developments
16.12.2019
Updated Vendor and Product ID's