

AcceleroMMA7361

Generated by Doxygen 1.7.6.1

Tue Jul 3 2012 16:25:31

Contents

1	Class Index	1
1.1	Class List	1
2	Class Documentation	1
2.1	AcceleroMMA7361 Class Reference	1
2.1.1	Constructor & Destructor Documentation	2
2.1.2	Member Function Documentation	2

1 Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

AcceleroMMA7361	1
---------------------------------	---

2 Class Documentation

2.1 AcceleroMMA7361 Class Reference

Public Member Functions

- [AcceleroMMA7361](#) ()
constructor
- void [begin](#) ()
- void [begin](#) (int sleepPin, int selfTestPin, int zeroGPin, int gSelectPin, int xPin, int yPin, int zPin)
- int [getXRaw](#) ()
[getXRaw\(\)](#): Returns the raw data from the X-axis analog I/O port of the Arduino as an integer
- int [getYRaw](#) ()
[getYRaw\(\)](#): Returns the raw data from the Y-axis analog I/O port of the Arduino as an integer
- int [getZRaw](#) ()
[getZRaw\(\)](#): Returns the raw data from the Z-axis analog I/O port of the Arduino as an integer
- int [getXVolt](#) ()
[getXVolt\(\)](#): Returns the voltage in mV from the X-axis analog I/O port of the Arduino as a integer
- int [getYVolt](#) ()

- getYVolt(): Returns the voltage in mV from the Y-axis analog I/O port of the Arduino as a integer*
- int [getZVolt \(\)](#)
 - getZVolt(): Returns the voltage in mV from the Z-axis analog I/O port of the Arduino as a integer*
- int [getXAccel \(\)](#)
 - getXAccel(): Returns the acceleration of the X-axis as a int (1 G = 100.00)*
- int [getYAccel \(\)](#)
 - getYAccel(): Returns the acceleration of the Y-axis as a int (1 G = 100.00)*
- int [getZAccel \(\)](#)
 - getZAccel(): Returns the acceleration of the Z-axis as a int (1 G = 100.00)*
- void [getAccelXYZ](#) (int *_XAxis, int *_YAxis, int *_ZAxis)
 - getAccelXYZ(int *_XAxis, int *_YAxis, int *_ZAxis) returns all axis at once as pointers*
- int [getTotalVector \(\)](#)
 - getTotalVector returns the magnitude of the total acceleration vector as an integer*
- void [setOffSets](#) (int xOffsetSet, int yOffsetSet, int zOffsetSet)
- void [calibrate](#) ()
- void [setARefVoltage](#) (double _refV)
- void [setAveraging](#) (int avg)
 - setAveraging(int avg): Sets how many samples have to be averaged in getAccel default is 10.*
- int [getOrientation](#) ()
- void [setSensitivity](#) (boolean sensi)
 - setSensitivity sets the sensitivity to +/-1.5 G (HIGH) or +/-6 G (LOW) using a boolean HIGH (1.5 G) or LOW (6 G)*
- void [sleep](#) ()
 - sleep lets the device sleep (when device is sleeping already this does nothing)*
- void [wake](#) ()
 - wake enables the device after sleep (when device is not sleeping this does nothing) there is a 2 ms delay, due to enable response time (datasheet: typ 0.5 ms, max 2 ms)*

2.1.1 Constructor & Destructor Documentation

2.1.1.1 AcceleroMMA7361::AcceleroMMA7361 ()

constructor

acceleroMMA7361.cpp - Library for retrieving data from the MMA7361 accelerometer.
For more information: variable declaration, changelog,... see [AcceleroMMA7361.h](#)

2.1.2 Member Function Documentation

2.1.2.1 void AcceleroMMA7361::begin ()

begin function to set pins: sleepPin = 13, selfTestPin = 12, zeroGPIn = 11, gSelectPin = 10, xPin = A0, yPin = A1, zPin = A2. When you use [begin\(\)](#) with an empty parameter list, these standard values are used

2.1.2.2 void **AcceleroMMA7361::begin** (int *sleepPin*, int *selfTestPin*, int *zeroGPin*, int *gSelectPin*, int *xPin*, int *yPin*, int *zPin*)

begin variables

- int sleepPin: number indicating to which pin the sleep port is attached. DIGITAL OUT
- int selfTestPin: number indicating to which pin the selftest port is attached. DIGITAL OUT
- int zeroGPin: number indicating to which pin the ZeroGpin is connected to. DIGITAL IN
- int gSelectPin: number indication to which pin the Gselect is connected to. DIGITAL OUT
- int xPin: number indicating to which pin the x-axis pin is connected to. ANALOG IN
- int yPin: number indicating to which pin the y-axis pin is connected to. ANALOG IN
- int zPin: number indicating to which pin the z-axis pin is connected to. ANALOG IN
- int offset: array indicating the G offset on the x,y and z-axis When you use [begin\(\)](#) without variables standard values are loaded: A0,A1,A2 as input for X,Y,Z and digital pins 13,12,11,10 for sleep, selftest, zeroG and gSelect

2.1.2.3 void **AcceleroMMA7361::calibrate** ()

[calibrate\(\)](#): Sets X and Y values via setOffsets to zero. The Z axis will be set to 100 = 1G WARNING WHEN CALIBRATED YOU HAVE TO MAKE SURE THE Z-AXIS IS PERPENDICULAR WITH THE EARTHS SURFACE

2.1.2.4 int **AcceleroMMA7361::getOrientation** ()

getOrientation returns which axis perpendicular with the earths surface x=1,y=2,z=3 is positive or negative depending on which side of the axis is pointing downwards

2.1.2.5 void **AcceleroMMA7361::setARefVoltage** (double *refV*)

[setARefVoltage\(double _refV\)](#): Sets the AREF voltage to external, (now only takes 3.3 or 5 as parameter) default is 5 when no AREF is used. When you want to use 3.3 AREF, put a wire between the AREF pin and the 3.3 V VCC pin. This increases accuracy

2.1.2.6 void **AcceleroMMA7361::setOffSets** (int *xOffSet*, int *yOffSet*, int *zOffSet*)

[setOffSets\(int offSetX, int offSetY, int offSetZ\)](#): Sets the offset values for the x,y,z axis. The parameters are the offsets expressed in G-force (100 = 1 G) Offsets are added to the raw datafunctions