MPU6050_Library

Functions

void	**MPU6050_Initialize** ()
bool	**MPU6050_TestConnection** ()
uint8_t	**MPU6050_GetDeviceID** ()
void	**MPU6050_SetClockSource** (uint8_t source)
void	**MPU6050_SetFullScaleGyroRange** (uint8_t range)
uint8_t	**MPU6050_GetFullScaleGyroRange** ()
uint8_t	**MPU6050_GetFullScaleAccelRange** ()
void	**MPU6050_SetFullScaleAccelRange** (uint8_t range)
bool	**MPU6050_GetSleepModeStatus** ()
void	**MPU6050_SetSleepModeStatus** (FunctionalState NewState)
void	**MPU6050_GetRawAccelGyro** (s16 *AccelGyro)
void	**MPU6050_WriteBits** (uint8_t slaveAddr, uint8_t regAddr, uint8_t bitStart, uint8_t length, uint8_t data)
void	**MPU6050_WriteBit** (uint8_t slaveAddr, uint8_t regAddr, uint8_t bitNum, uint8_t data)
void	**MPU6050_ReadBits** (uint8_t slaveAddr, uint8_t regAddr, uint8_t bitStart, uint8_t length, uint8_t *data)
void	**MPU6050_ReadBit** (uint8_t slaveAddr, uint8_t regAddr, uint8_t bitNum, uint8_t *data)
void	**MPU6050_I2C_Init** () Initializes the I2C peripheral used to drive the MPU6050.
void	**MPU6050_I2C_ByteWrite** (u8 slaveAddr, u8 *pBuffer, u8 writeAddr) Writes one byte to the MPU6050.
void	**MPU6050_I2C_BufferRead** (u8 slaveAddr, u8 *pBuffer, u8 readAddr, u16 NumByteToRead) Reads a block of data from the MPU6050.

Detailed Description

Function Documentation

uint8_t MPU6050_GetDeviceID ( )

Get Device ID. This register is used to verify the identity of the device (0b110100).

Returns:

Device ID (should be 0x68, 104 dec, 150 oct)

See also:

MPU6050_RA_WHO_AM_I

MPU6050_WHO_AM_I_BIT

MPU6050_WHO_AM_I_LENGTH

uint8_t MPU6050_GetFullScaleAccelRange ( )

Get full-scale accelerometer range. The FS_SEL parameter allows setting the full-scale range of the accelerometer sensors, as described in the table below.

```
0 = +/- 2g
1 = +/- 4g
2 = +/- 8g
3 = +/- 16g
```

Returns:

Current full-scale accelerometer range setting

See also:

MPU6050_ACCEL_FS_2

MPU6050_RA_ACCEL_CONFIG

MPU6050_ACONFIG_AFS_SEL_BIT

MPU6050_ACONFIG_AFS_SEL_LENGTH

uint8_t MPU6050_GetFullScaleGyroRange ( )

Get full-scale gyroscope range. The FS_SEL parameter allows setting the full-scale range of the gyro sensors, as described in the table below.

```
0 = +/- 250 degrees/sec
1 = +/- 500 degrees/sec
2 = +/- 1000 degrees/sec
3 = +/- 2000 degrees/sec
```

Returns:

Current full-scale gyroscope range setting

See also:

MPU6050_GYRO_FS_250

MPU6050_RA_GYRO_CONFIG

MPU6050_GCONFIG_FS_SEL_BIT

MPU6050_GCONFIG_FS_SEL_LENGTH

void MPU6050_GetRawAccelGyro (s16 * AccelGyro)

Get raw 6-axis motion sensor readings (accel/gyro). Retrieves all currently available motion sensor values.

Parameters:

AccelGyro 16-bit signed integer array of length 6

See also:

MPU6050_RA_ACCEL_XOUT_H

bool MPU6050_GetSleepModeStatus ( )

Get sleep mode status. Setting the SLEEP bit in the register puts the device into very low power sleep mode. In this mode, only the serial interface and internal registers remain active, allowing for a very low standby current. Clearing this bit puts the device back into normal mode. To save power, the individual standby selections for each of the gyros should be used if any gyro axis is not used by the application.

Returns:

Current sleep mode enabled status

See also:

MPU6050_RA_PWR_MGMT_1

MPU6050_PWR1_SLEEP_BIT

void MPU6050_I2C_BufferRead (u8 slaveAddr,
u8 * pBuffer,
u8 readAddr,
u16 NumByteToRead
)

Reads a block of data from the MPU6050.

Parameters:

slaveAddr	: slave address MPU6050_DEFAULT_ADDRESS
pBuffer	: pointer to the buffer that receives the data read from the MPU6050.
readAddr	: MPU6050's internal address to read from.
NumByteToRead	: number of bytes to read from the MPU6050 (NumByteToRead >1 only for the Mgnetometer readinf).

Returns:

None

void MPU6050_I2C_ByteWrite (u8 slaveAddr,
u8 * pBuffer,
u8 writeAddr
)

Writes one byte to the MPU6050.

Parameters:

slaveAddr : slave address MPU6050_DEFAULT_ADDRESS
pBuffer : pointer to the buffer containing the data to be written to the MPU6050.
writeAddr : address of the register in which the data will be written

Returns:

None

void MPU6050_I2C_Init ( )

Initializes the I2C peripheral used to drive the MPU6050.

Parameters:

None

Returns:

None

void MPU6050_Initialize ( )

Power on and prepare for general usage. This will activate the device and take it out of sleep mode (which must be done after start-up). This function also sets both the accelerometer and the gyroscope to their most sensitive settings, namely +/- 2g and +/- 250 degrees/sec, and sets the clock source to use the X Gyro for reference, which is slightly better than the default internal clock source.

```
void MPU6050_ReadBit ( uint8_t    slaveAddr,
                       uint8_t    regAddr,
                       uint8_t    bitNum,
                       uint8_t *  data
                     )
```

Read a single bit from an 8-bit device register.

Parameters:

slaveAddr	I2C slave device address
regAddr	Register regAddr to read from
bitNum	Bit position to read (0-7)
data	Container for single bit value
timeout	Optional read timeout in milliseconds (0 to disable, leave off to use default class value in readTimeout)

```
void MPU6050_ReadBits ( uint8_t    slaveAddr,
                        uint8_t    regAddr,
                        uint8_t    bitStart,
                        uint8_t    length,
                        uint8_t *  data
                      )
```

Read multiple bits from an 8-bit device register.

Parameters:

slaveAddr	I2C slave device address
regAddr	Register regAddr to read from
bitStart	First bit position to read (0-7)
length	Number of bits to read (not more than 8)
data	Container for right-aligned value (i.e. '101' read from any bitStart position will equal 0x05)
timeout	Optional read timeout in milliseconds (0 to disable, leave off to use default class value in readTimeout)

void MPU6050_SetClockSource (uint8_t source)

Set clock source setting. An internal 8MHz oscillator, gyroscope based clock, or external sources can be selected as the MPU-60X0 clock source. When the internal 8 MHz oscillator or an external source is chosen as the clock source, the MPU-60X0 can operate in low power modes with the gyroscopes disabled.

Upon power up, the MPU-60X0 clock source defaults to the internal oscillator. However, it is highly recommended that the device be configured to use one of the gyroscopes (or an external clock source) as the clock reference for improved stability. The clock source can be selected according to the following table:

```
CLK_SEL | Clock Source
——-+——————————
0       | Internal oscillator
1       | PLL with X Gyro reference
2       | PLL with Y Gyro reference
3       | PLL with Z Gyro reference
4       | PLL with external 32.768kHz reference
5       | PLL with external 19.2MHz reference
6       | Reserved
7       | Stops the clock and keeps the timing generator in reset
```

Parameters:

source New clock source setting

See also:

MPU6050_GetClockSource()

MPU6050_RA_PWR_MGMT_1

MPU6050_PWR1_CLKSEL_BIT

MPU6050_PWR1_CLKSEL_LENGTH

void MPU6050_SetFullScaleAccelRange (uint8_t range)

Set full-scale accelerometer range.

Parameters:

range New full-scale accelerometer range setting

See also:

MPU6050_GetFullScaleAccelRange()

void MPU6050_SetFullScaleGyroRange (uint8_t range)

Set full-scale gyroscope range.

Parameters:

range New full-scale gyroscope range value

See also:

MPU6050_GetFullScaleGyroRange()

MPU6050_GYRO_FS_250

MPU6050_RA_GYRO_CONFIG

MPU6050_GCONFIG_FS_SEL_BIT

MPU6050_GCONFIG_FS_SEL_LENGTH

void MPU6050_SetSleepModeStatus (FunctionalState NewState)

Set sleep mode status.

Parameters:

enabled New sleep mode enabled status

See also:

MPU6050_GetSleepModeStatus()

MPU6050_RA_PWR_MGMT_1

MPU6050_PWR1_SLEEP_BIT

bool MPU6050_TestConnection ( )

Verify the I2C connection. Make sure the device is connected and responds as expected.

Returns:

True if connection is valid, FALSE otherwise

```
void MPU6050_WriteBit ( uint8_t slaveAddr,
                        uint8_t regAddr,
                        uint8_t bitNum,
                        uint8_t data
                      )
```

write a single bit in an 8-bit device register.

Parameters:

slaveAddr	I2C slave device address
regAddr	Register regAddr to write to
bitNum	Bit position to write (0-7)
value	New bit value to write

```
void MPU6050_WriteBits ( uint8_t slaveAddr,
                         uint8_t regAddr,
                         uint8_t bitStart,
                         uint8_t length,
                         uint8_t data
                       )
```

Write multiple bits in an 8-bit device register.

Parameters:

slaveAddr	I2C slave device address
regAddr	Register regAddr to write to
bitStart	First bit position to write (0-7)
length	Number of bits to write (not more than 8)
data	Right-aligned value to write