Infineon Arduino Library Documentation

Author: Infineon Technologies AG
Date: February 12, 2019

Contents

1 TLE5012B-Angle-Sensor

Library of Infineon's highly sensitive [TLE5012B] 360° magnetic angle sensor(https://www.infineon.com/cms/en/product/sensor/magnetic-position-sensor/angle-sensor/tle5012b-e1000/) for Arduino.

Summary

The TLE5012B is a 360° angle sensor that detects the orientation of a magnetic field. This is achieved by measuring sine and cosine angle components with monolithic integrated Giant Magneto Resistance (iGMR) elements. These raw signals (sine and cosine) are digitally processed internally to calculate the angle orientation of the magnetic field (magnet). The TLE5012B is a pre-calibrated sensor. The calibration parameters are stored in laser fuses. At start-up the values of the fuses are written into flip-flops, where these values can be changed by the application-specific parameters. Further precision of the angle measurement over a wide temperature range and a long lifetime can be improved by enabling an optional internal autocalibration algorithm. Data communications are accomplished with a bi-directional Synchronous Serial Communication (SSC) that is SPI-compatible. The sensor configuration is stored in registers, which are accessible by the SSC interface. Additionally four other interfaces are available with the TLE5012B: Pulse-Width-Modulation (PWM) Protocol, Short-PWM-Code (SPC) Protocol, Hall Switch Mode (HSM) and Incremental Interface (IIF). These interfaces can be used in parallel with SSC or alone. Pre-configured sensor derivates with different interface settings are available.

Key Features and Benefits

- Giant Magneto Resistance (GMR)-based principle
- Integrated magnetic field sensing for angle measurement
- 360° angle measurement with revolution counter and angle speed measurement
- Two separate highly accurate single bit SD-ADC
- 15 bit representation of absolute angle value on the output (resolution of 0.01°)
- 16 bit representation of sine / cosine values on the interface
- Max. 1.0° angle error over lifetime and temperature-range with activated auto-calibration
- Bi-directional SSC Interface up to 8 Mbit/s
- Interfaces: SSC, PWM, Incremental Interface (IIF), Hall Switch Mode (HSM), Short PWM Code (SPC, based on SENT protocol defined in SAE J2716)
- Output pins can be configured (programmed or pre-configured) as push-pull or open-drain
- Bus mode operation of multiple sensors on one line is possible with SSC or SPC interface

Hardware

Please find the datasheet of the TLE5012B here. It depends on the evaluation board which you are using or the respective configuration of the sensor on your PCB which communication protocol as well as addresses you need to use for communicating with the sensor. This library only works with the SPI compatible Synchronous Serial Communication (SSC) interface of the TLE5012B.

Installation

Integration of Library

Please download this repository from GitHub by clicking on the following field in the latest release of this repository:

To install the TLE5012B angle sensor library in the Arduino IDE, please go now to **Sketch** > **Include Library** > **Add .ZIP Library...** in the Arduino IDE and navigate to the downloaded .ZIP file of this repository. The library will be installed in your Arduino sketch folder in libraries and you can select as well as include this one to your project under **Sketch** > **Include Library** > **TLE5012B**.

Usage

Please follow the example sketches in the /examples directory in this library to learn more about the usage of the library. Especially, take care of the SPI and I²C configuration of the sensor. For more information, please consult the datasheet here.

2 License Summary for Repository

```
Important Notice:
Changes, suggestions and commits in this repository may only be done following each
license of the respective file and putting it under the same license (except
stated otherwise by the license).
All rights of the respective copyright holders shall be reserved.
Brands and product names are trademarks of their respective owners.
Referred and linked files/pages are out-of-scope of this repository and underly
their respective licenses.
```

In case of deviating information in this repository, license information in files is decisive except the above 'Important Notice' section which shall be valid in the scope of the repository (not if statement here is not applicable).

License

- WHETHER IN CONTRACT, STRICT LIABILITY,OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) A↩ RISING IN ANY WAY OUT OF THE USE
- OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
-
- To improve the quality of the software, users are encouraged to share modifications, enhancements or bug fixes with
- Infineon Technologies AG.

3 Data Structure Index

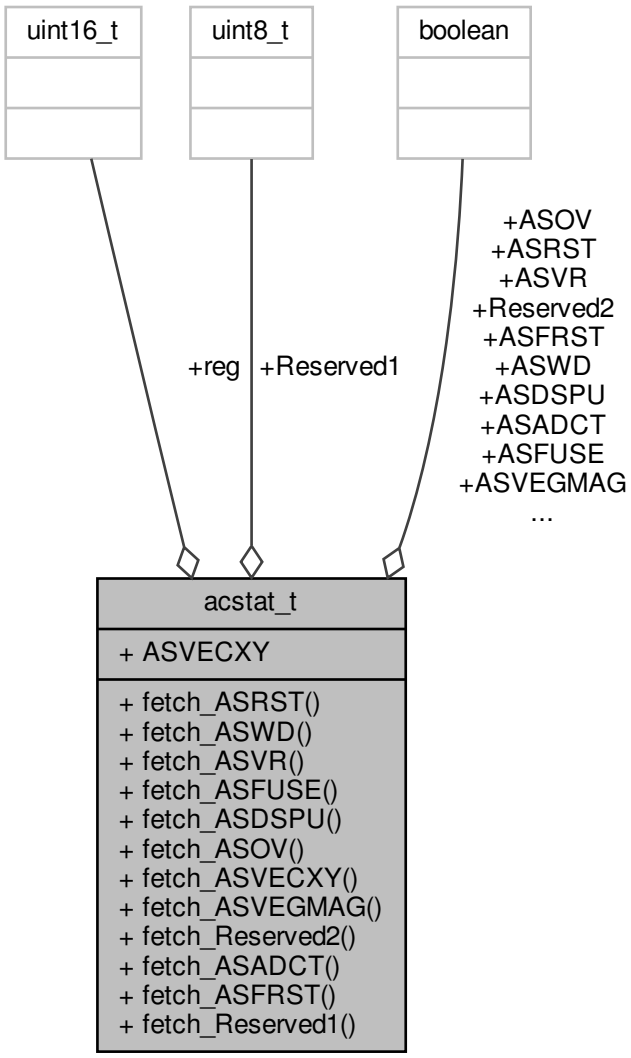
3.1 Data Structures

Here are the data structures with brief descriptions:

4 Data Structure Documentation

4.1 acstat_t Struct Reference

Collaboration diagram for acstat_t:



Public Member Functions

- boolean fetch_ASRST (uint16_t reg)

  the register value
- boolean **fetch_ASWD** (uint16_t reg)
- boolean **fetch_ASVR** (uint16_t reg)

- boolean **fetch_ASFUSE** (uint16_t reg)
- boolean **fetch_ASDSPU** (uint16_t reg)
- boolean **fetch_ASOV** (uint16_t reg)
- boolean **fetch_ASVECXY** (uint16_t reg)
- boolean **fetch_ASVEGMAG** (uint16_t reg)
- uint8_t **fetch_Reserved2** (uint16_t reg)
- boolean **fetch_ASADCT** (uint16_t reg)
- boolean **fetch_ASFRST** (uint16_t reg)
- boolean **fetch_Reserved1** (uint16_t reg)

Data Fields

- uint8_t Reserved1

  Activation Status register offset 0x01
- boolean ASFRST

  bits 15:11
- boolean ASADCT

  bits 10:10 Activation of Firmware Reset
- boolean Reserved2

  bits 9:9 Enable ADC Test vector Check
- boolean ASVEGMAG

  bits 8:8
- boolean ASVECXY

  bits 7:7 Activation of Magnitude Check
- boolean ASOV

  bits 6:6 Activation of X,Y Out of Limit-Check
- boolean ASDSPU

  bits 5:5 Enable of DSPU Overflow Check
- boolean ASFUSE

  bits 4:4 Activation DSPU BIST
- boolean ASVR

  bits 3:3 Activation Fuse CRC
- boolean ASWD

  bits 2:2 Enable Voltage regulator Check
- boolean ASRST

  bits 1:1 Enable DSPU Watchdog
- uint16_t reg

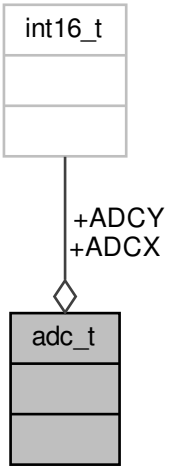
  bits 0:0 Activation of Hardware Reset

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.2 adc_t Struct Reference

Collaboration diagram for adc_t:



Data Fields

- int16_t ADCX

  ADC_X offset 0x10, ADC_Y offset 0x11
- int16_t ADCY

  bits 15:0 ADC value of X-GMR

4.2.1 Field Documentation

4.2.1.1 ADCX

`int16_t adc_t::ADCX`

Referenced by Tle5012b::readSensorType().

4.2.1.2 ADCY

`int16_t adc_t::ADCY`

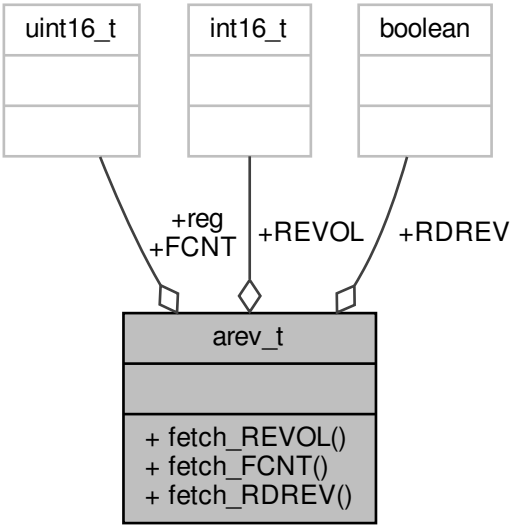
Referenced by Tle5012b::readSensorType().

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.3 arev_t Struct Reference

Collaboration diagram for arev_t:



Public Member Functions

- int16_t fetch_REVOL (uint16_t reg)

  the register value
- uint16_t **fetch_FCNT** (uint16_t reg)
- boolean **fetch_RDREV** (uint16_t reg)

Data Fields

- int16_t REVOL

  Angle Revolution register offset 0x04
- uint16_t FCNT

  bits 8:0 Revolution counter. Increments for every full rotation in counter-clockwise direction
- boolean RDREV

  bits 14:9 Internal frame counter. Increments every update period
- uint16_t reg

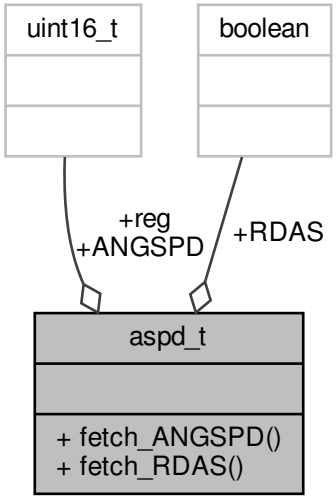
  bits 15:15 Read Status, Revolution

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.4 aspd_t Struct Reference

Collaboration diagram for aspd_t:



Public Member Functions

- boolean fetch_ANGSPD (uint16_t reg)

  the register value
- uint16_t **fetch_RDAS** (uint16_t reg)

Data Fields

- uint16_t ANGSPD

  Angle Speed register offset 0x03
- boolean RDAS

  bits 14:0 Signed value, where the sign bit [14] indicates the direction of the rotation.
- uint16_t reg

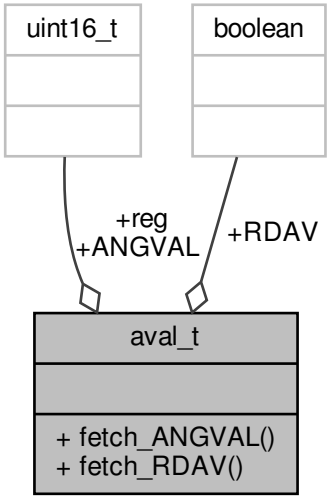
  bits 15:15 Read Status, Angle Speed

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.5 aval_t Struct Reference

Collaboration diagram for aval_t:



Public Member Functions

- boolean fetch_ANGVAL (uint16_t reg)

  the register value
- uint16_t **fetch_RDAV** (uint16_t reg)

Data Fields

- uint16_t ANGVAL

  Angle Value register offset 0x02
- boolean RDAV

  bits 14:0 Calculated Angle Value (signed 15-bit)
- uint16_t reg

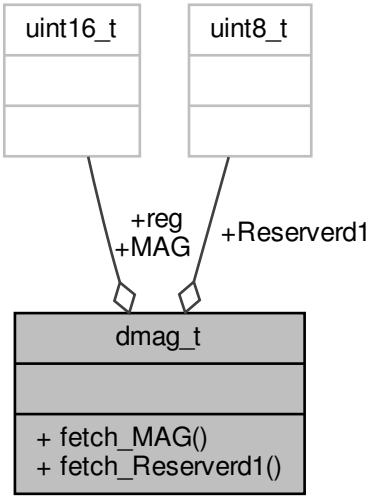
  bits 15:15 Read Status, Angle Value

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.6 dmag_t Struct Reference

Collaboration diagram for dmag_t:



Public Member Functions

- uint16_t fetch_MAG (uint16_t reg)

  the register value
- uint8_t **fetch_Reserverd1** (uint16_t reg)

Data Fields

- uint8_t Reserverd1

  D_Mag vector magnitude offset 0x14
- uint16_t MAG

  bits 15:10
- uint16_t reg

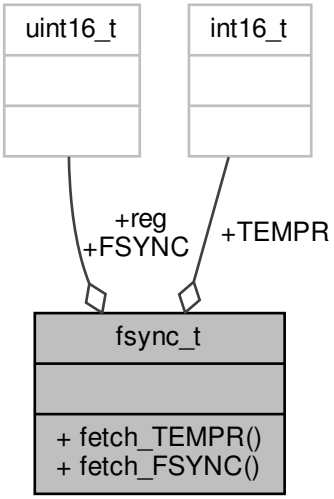
  bits 9:0 Unsigned Angle Vector Magnitude after X, Y error compensation (due to temperature).

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.7 fsync_t Struct Reference

Collaboration diagram for fsync_t:



Public Member Functions

- int16_t fetch_TEMPR (uint16_t reg)

  the register value
- uint16_t **fetch_FSYNC** (uint16_t reg)

Data Fields

- uint16_t FSYNC

  Frame Synchronization register offset 0x05
- int16_t TEMPR

  bits 15:9 Frame Synchronization Counter Value
- uint16_t reg

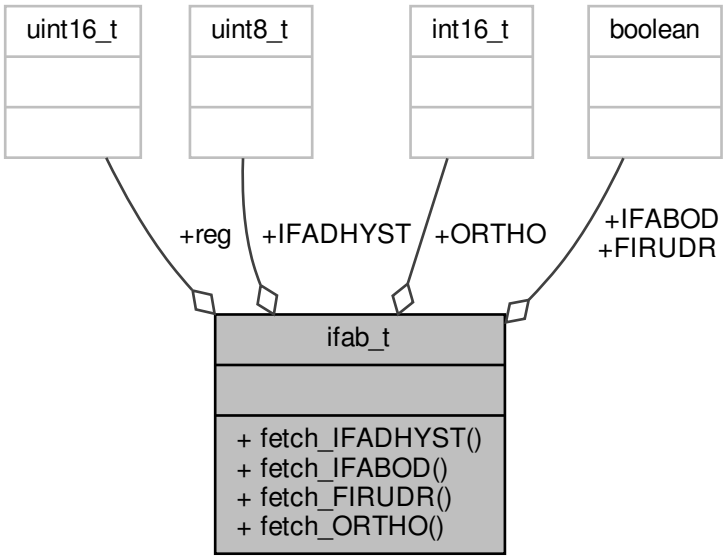
  bits 8:0 Signed offset compensated temperature value.

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.8 ifab_t Struct Reference

Collaboration diagram for ifab_t:



Public Member Functions

- uint8_t fetch_IFADHYST (uint16_t reg)

  the register value
- boolean **fetch_IFABOD** (uint16_t reg)
- boolean **fetch_FIRUDR** (uint16_t reg)
- uint16_t **fetch_ORTHO** (uint16_t reg)

Data Fields

- int16_t ORTHO

  IFAB register offset 0x0d
- boolean FIRUDR

  bits 15:4 Orthogonality Correction of X and Y Components
- boolean IFABOD

  bits 3:3 Initial filter update rate (FIR)
- uint8_t IFADHYST

  bits 2:2 IFA,IFB,IFC Output Mode
- uint16_t reg

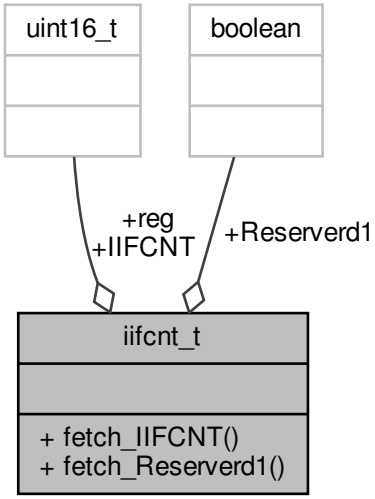
  bits 1:0 Hysteresis (multi-purpose)

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.9 iifcnt_t Struct Reference

Collaboration diagram for iifcnt_t:



Public Member Functions

- uint16_t fetch_IIFCNT (uint16_t reg)

  the register value
- boolean **fetch_Reserverd1** (uint16_t reg)

Data Fields

- boolean Reserverd1

  IIF counter value offset 0x20
- uint16_t IIFCNT

  bits 15:14
- uint16_t reg

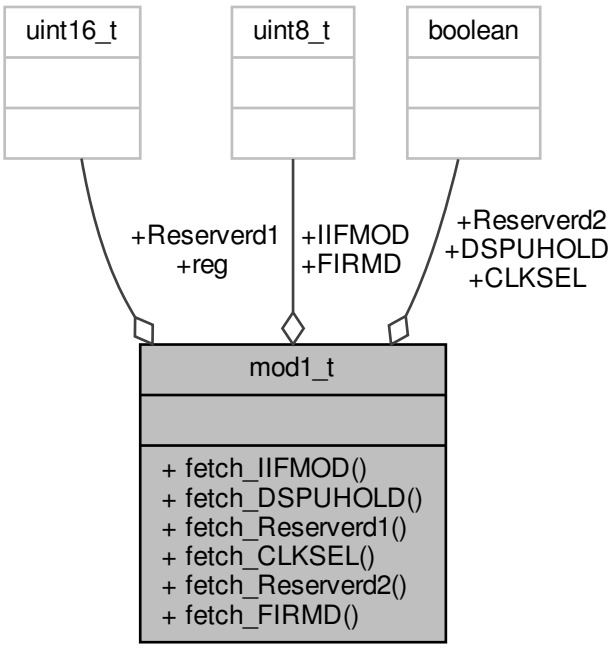
  bits 14:0 14 bit counter value of IIF increments

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.10 mod1_t Struct Reference

Collaboration diagram for mod1_t:



Public Member Functions

- uint8_t fetch_IIFMOD (uint16_t reg)

  the register value
- boolean **fetch_DSPUHOLD** (uint16_t reg)
- uint16_t **fetch_Reserverd1** (uint16_t reg)
- boolean **fetch_CLKSEL** (uint16_t reg)
- boolean **fetch_Reserverd2** (uint16_t reg)
- uint8_t **fetch_FIRMD** (uint16_t reg)

Data Fields

- uint8_t FIRMD

  MOD_1 Interface Mode1 register offset 0x06
- uint16_t Reserverd1

  bits 15:14 Update Rate Setting
- boolean CLKSEL

  bits 13:5
- boolean Reserverd2

  bits 4:4 Switch to external clock at start-up only.

- boolean DSPUHOLD

  bits 3:3
- uint8_t IIFMOD

  bits 2:2 If DSPU is on hold, no watchdog reset is performed by DSPU
- uint16_t reg

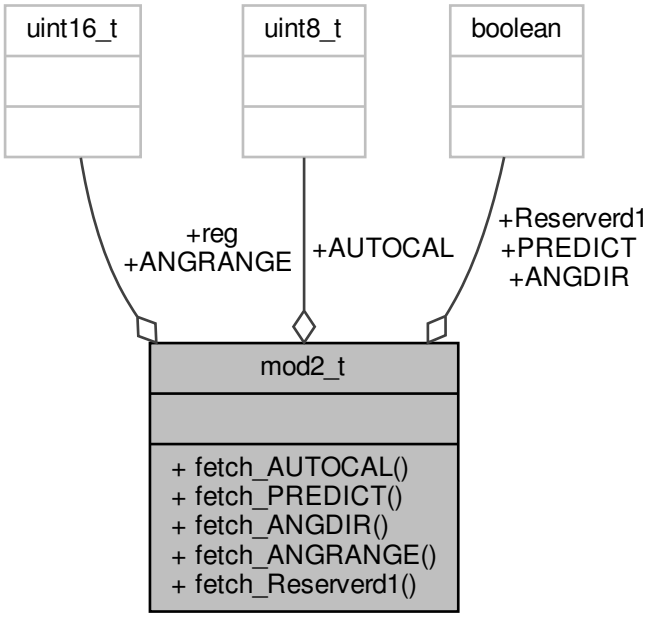
  bits 1:0 Incremental Interface Mode

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.11 mod2_t Struct Reference

Collaboration diagram for mod2_t:



Public Member Functions

- uint8_t fetch_AUTOCAL (uint16_t reg)

  the register value
- boolean **fetch_PREDICT** (uint16_t reg)
- boolean **fetch_ANGDIR** (uint16_t reg)
- uint16_t **fetch_ANGRANGE** (uint16_t reg)
- boolean **fetch_Reserverd1** (uint16_t reg)

Data Fields

- boolean Reserverd1

  MOD_2 Interface Mode2 register offset 0x08
- uint16_t ANGRANGE

  bits 15:15
- boolean ANGDIR

  bits 14:4 Changes the representation of the angle output by multiplying the output with a factor ANG_RAN↩ GE/128.
- boolean PREDICT

  bits 3:3 Inverts angle and angle speed values and revolution counter behaviour.
- uint8_t AUTOCAL

  bits 2:2 Prediction of angle value based on current angle speed
- uint16_t reg

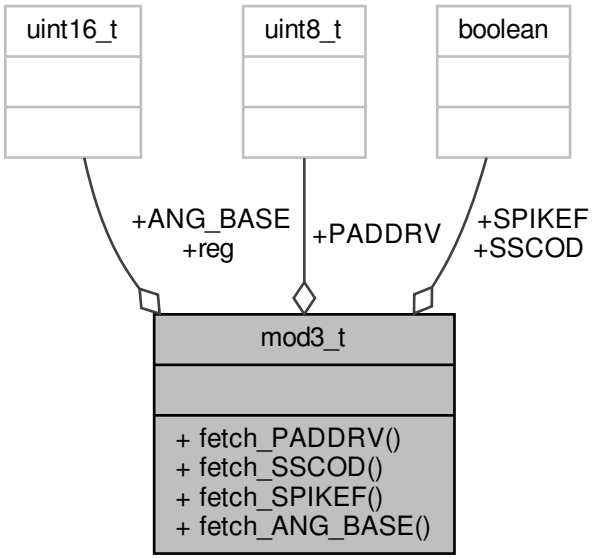
  bits 1:0 Automatic calibration of offset and amplitude synchronicity for applications with full-turn.

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.12 mod3_t Struct Reference

Collaboration diagram for mod3_t:



Public Member Functions

- uint8_t fetch_PADDRV (uint16_t reg)

  the register value
- boolean **fetch_SSCOD** (uint16_t reg)
- boolean **fetch_SPIKEF** (uint16_t reg)
- uint16_t **fetch_ANG_BASE** (uint16_t reg)

Data Fields

- uint16_t ANG_BASE

  MOD_3 Interface Mode3 register offset 0x09
- boolean SPIKEF

  bits 15:4 Sets the 0° angle position (12 bit value). Angle base is factory-calibrated to make the 0° direction parallel to the edge of the chip.
- boolean SSCOD

  bits 3:3 Filters voltage spikes on input pads (IFC, SCK and CSQ).
- uint8_t PADDRV

  bits 2:2 SSC-Interface Data Pin Output Mode
- uint16_t reg

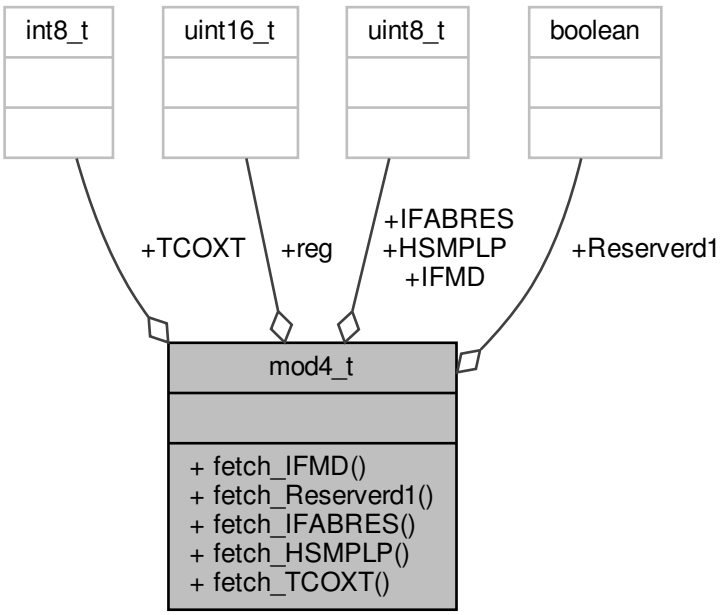
  bits 1;0 Configuration of Pad-Driver

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.13 mod4_t Struct Reference

Collaboration diagram for mod4_t:



Public Member Functions

- uint8_t fetch_IFMD (uint16_t reg)

  the register value
- boolean **fetch_Reserverd1** (uint16_t reg)
- uint8_t **fetch_IFABRES** (uint16_t reg)
- uint8_t **fetch_HSMPLP** (uint16_t reg)
- int8_t **fetch_TCOXT** (uint16_t reg)

Data Fields

- int8_t TCOXT

  MOD_4 Interface Mode4 register offset 0x0e
- uint8_t HSMPLP

  bits 15:9 7-bit signed integer value of X-offset temperature coefficient.
- uint8_t IFABRES

  bits 8:5 Hall Switch mode (multi-purpose)
- boolean Reserverd1

  bits 4:3 IIF resolution (multi-purpose)
- uint8_t IFMD

bits 2:2

- uint16_t reg

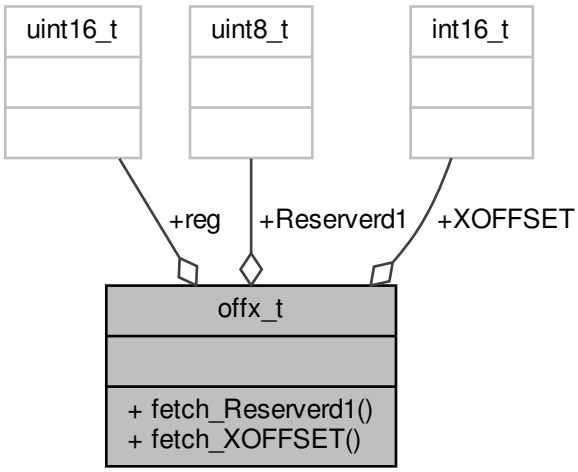
  bits 1:0 Interface Mode on IFA,IFB,IFC

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.14 offx_t Struct Reference

Collaboration diagram for offx_t:



Public Member Functions

- uint8_t fetch_Reserverd1 (uint16_t reg)

  the register value
- uint16_t **fetch_XOFFSET** (uint16_t reg)

Data Fields

- int16_t XOFFSET

  Offset X offset 0x0a
- uint8_t Reserverd1

  bits 15:4 12-bit signed integer value of raw X-signal offset correction at 25 ℃.
- uint16_t reg

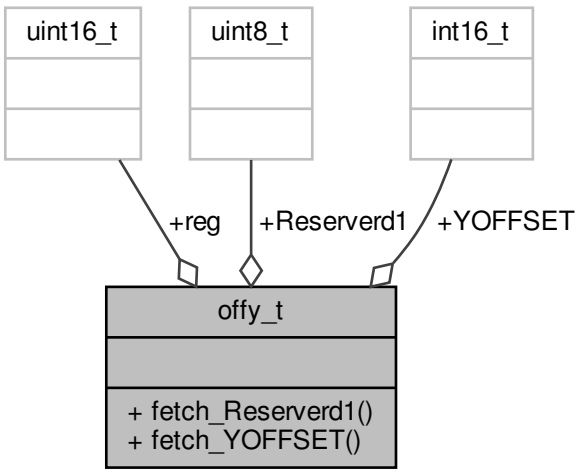
  bits 3:0

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.15 offy_t Struct Reference

Collaboration diagram for offy_t:



Public Member Functions

- uint8_t fetch_Reserverd1 (uint16_t reg)

  the register value
- uint16_t **fetch_YOFFSET** (uint16_t reg)

Data Fields

- int16_t YOFFSET

  Offset Y offset 0x0b
- uint8_t Reserverd1

  bits 15:4 12-bit signed integer value of raw Y-signal offset correction at 25°C.
- uint16_t reg

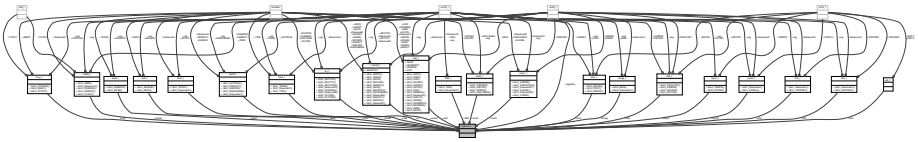
  bits 3:0

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.16 regSensor_t Struct Reference

Collaboration diagram for regSensor_t:



Data Fields

- uint16_t **registers** [MAX_NUM_REG]
- struct stat_t **stat**
- struct acstat_t **acstat**
- struct aval_t **aval**
- struct aspd_t **aspd**
- struct arev_t **arev**
- struct fsync_t **fsync**
- struct mod1_t **mod1**
- struct sil_t **sil**
- struct mod2_t **mod2**
- struct mod3_t **mod3**
- struct offx_t **offx**
- struct offy_t **offy**
- struct synch_t **synch**
- struct ifab_t **ifab**
- struct mod4_t **mod4**
- struct tcoy_t **tcoy**
- struct adc_t **adc**
- struct dmag_t **dmag**
- struct traw_t **traw**
- struct iifcnt_t **iifcnt**
- struct t250_t **t250**

4.16.1 Field Documentation

4.16.1.1 registers

```
uint16_t regSensor_t::registers[MAX_NUM_REG]
```

4.16.1.2 stat

```
struct stat_t regSensor_t::stat
```

4.16.1.3 acstat

`struct acstat_t regSensor_t::acstat`

4.16.1.4 aval

`struct aval_t regSensor_t::aval`

4.16.1.5 aspd

`struct aspd_t regSensor_t::aspd`

4.16.1.6 arev

`struct arev_t regSensor_t::arev`

4.16.1.7 fsync

`struct fsync_t regSensor_t::fsync`

4.16.1.8 mod1

`struct mod1_t regSensor_t::mod1`

4.16.1.9 sil

`struct sil_t regSensor_t::sil`

4.16.1.10 mod2

`struct mod2_t regSensor_t::mod2`

4.16.1.11 mod3

`struct mod3_t regSensor_t::mod3`

4.16.1.12 offx

`struct offx_t regSensor_t::offx`

4.16.1.13 offy

struct offy_t regSensor_t::offy

4.16.1.14 synch

struct synch_t regSensor_t::synch

4.16.1.15 ifab

struct ifab_t regSensor_t::ifab

4.16.1.16 mod4

struct mod4_t regSensor_t::mod4

4.16.1.17 tcoy

struct tcoy_t regSensor_t::tcoy

4.16.1.18 adc

struct adc_t regSensor_t::adc

4.16.1.19 dmag

struct dmag_t regSensor_t::dmag

4.16.1.20 traw

struct traw_t regSensor_t::traw

4.16.1.21 iifcnt

struct iifcnt_t regSensor_t::iifcnt

4.16.1.22 t250

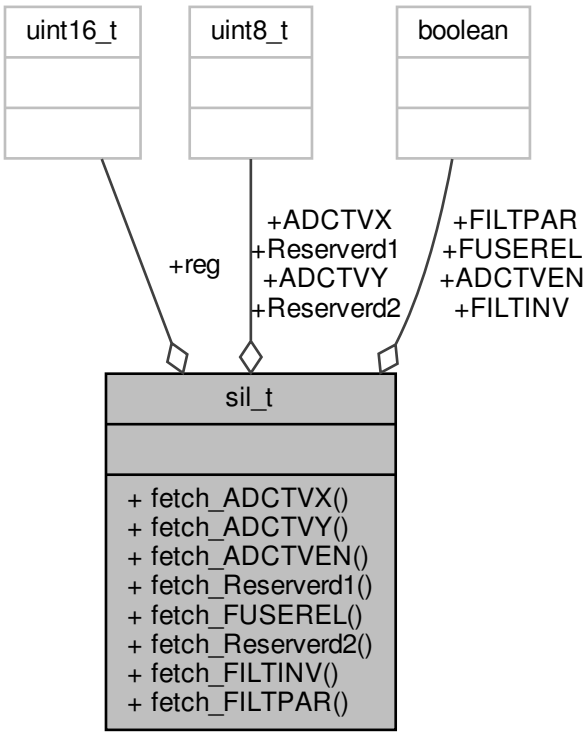
struct t250_t regSensor_t::t250

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.17 sil_t Struct Reference

Collaboration diagram for sil_t:



Public Member Functions

- uint8_t fetch_ADCTVX (uint16_t reg)

  the register value
- uint8_t **fetch_ADCTVY** (uint16_t reg)
- boolean **fetch_ADCTVEN** (uint16_t reg)
- uint8_t **fetch_Reserverd1** (uint16_t reg)
- boolean **fetch_FUSEREL** (uint16_t reg)
- uint8_t **fetch_Reserverd2** (uint16_t reg)
- boolean **fetch_FILTINV** (uint16_t reg)
- boolean **fetch_FILTPAR** (uint16_t reg)

Data Fields

- boolean FILTPAR

  SIL register offset 0x07
- boolean FILTINV

  bits 15:15

- uint8_t Reserverd1

  bits 14:14 The raw X-signal is routed also to the raw Y-signal input of the filter so SIN and COS signal should be identical.
- boolean FUSEREL

  bits 13:11 The X- and Y-signals are inverted. The angle output is then shifted by 180°.
- uint8_t Reserverd2

  bits 10:10 Triggers reload of default values from laser fuses into configuration registers.
- boolean ADCTVEN

  bits 9:7
- uint8_t ADCTVY

  bits 6:6 Sensor elements are internally disconnected and test voltages are connected to ADCs.
- uint8_t ADCTVX

  bits 5:3 Test vector X
- uint16_t reg

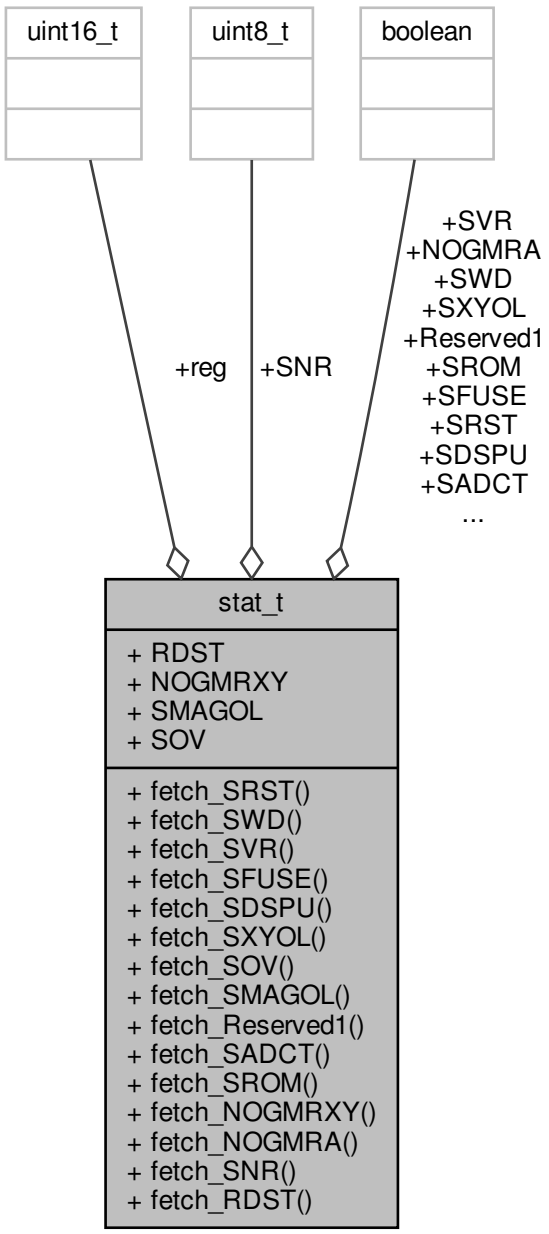
  bits 2:0 Test vector Y

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.18 stat_t Struct Reference

Collaboration diagram for stat_t:



Public Member Functions

- boolean fetch_SRST (uint16_t reg)

  the register value
- boolean **fetch_SWD** (uint16_t reg)

- boolean **fetch_SVR** (uint16_t reg)
- boolean **fetch_SFUSE** (uint16_t reg)
- boolean **fetch_SDSPU** (uint16_t reg)
- boolean **fetch_SXYOL** (uint16_t reg)
- boolean **fetch_SOV** (uint16_t reg)
- boolean **fetch_SMAGOL** (uint16_t reg)
- boolean **fetch_Reserved1** (uint16_t reg)
- boolean **fetch_SADCT** (uint16_t reg)
- boolean **fetch_SROM** (uint16_t reg)
- boolean **fetch_NOGMRXY** (uint16_t reg)
- boolean **fetch_NOGMRA** (uint16_t reg)
- uint8_t **fetch_SNR** (uint16_t reg)
- boolean **fetch_RDST** (uint16_t reg)

Data Fields

- boolean RDST

  Status register 0x00
- uint8_t SNR

  bits 15:15 Read status
- boolean NOGMRA

  bits 14:13 Slave number
- boolean NOGMRXY

  bits 12:12 No valid GMR angle value
- boolean SROM

  bits 11:11 No valid GMR XY values
- boolean SADCT

  bits 10:10 Status ROM
- boolean Reserved1

  bits 9:9 Status ADC Test
- boolean SMAGOL

  bits 8:8
- boolean SXYOL

  bits 7:7 Status magnitude out of Limit
- boolean SOV

  bits 6:6 Status X,Y Data out of Limit
- boolean SDSPU

  bits 5:5 Status overflow
- boolean SFUSE

  bits 4:4 Status digital signal processing unit
- boolean SVR

  bits 3:3 Status fuse CRC
- boolean SWD

  bits 2:2 Status voltage regulator
- boolean SRST

bits 1:1 Status Watchdog

- uint16_t reg

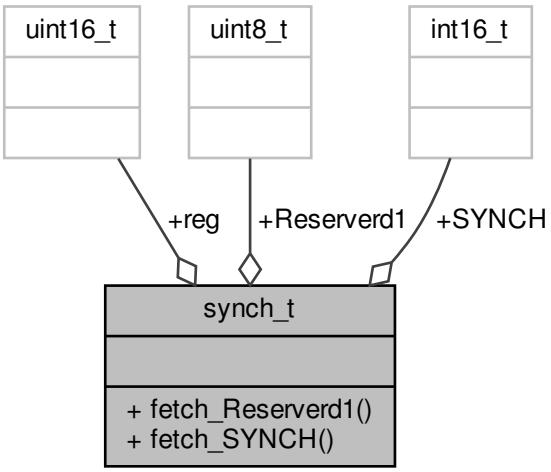
  bits 0:0 Status Reset

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.19 synch_t Struct Reference

Collaboration diagram for synch_t:



Public Member Functions

- uint8_t fetch_Reserverd1 (uint16_t reg)

  the register value
- uint16_t **fetch_SYNCH** (uint16_t reg)

Data Fields

- int16_t SYNCH

  Synchronicity offset 0x0c
- uint8_t Reserverd1

  bits 15:4 12-bit signed integer value of amplitude synchronicity
- uint16_t reg

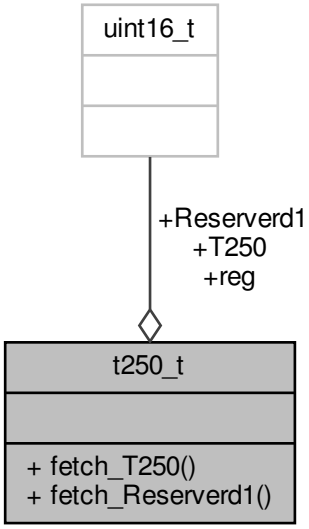
  bits 3:0

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.20 t250_t Struct Reference

Collaboration diagram for t250_t:



Public Member Functions

- uint16_t fetch_T250 (uint16_t reg)

  the register value
- uint16_t **fetch_Reserverd1** (uint16_t reg)

Data Fields

- uint16_t T250

  register T250 offset 0x30
- uint16_t Reserverd1

  bits 15:9 Signed offset value at 25℃ temperature; 1dig=0.36℃.
- uint16_t reg

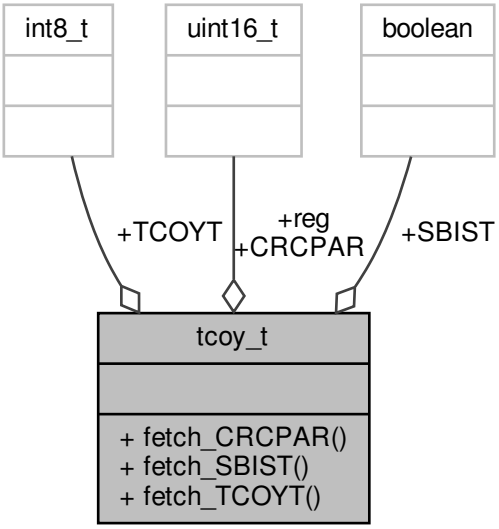
  bit 8:0

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.21 tcoy_t Struct Reference

Collaboration diagram for tcoy_t:



Public Member Functions

- uint16_t fetch_CRCPAR (uint16_t reg)

  the register value
- boolean **fetch_SBIST** (uint16_t reg)
- int8_t **fetch_TCOYT** (uint16_t reg)

Data Fields

- int8_t TCOYT

  TCO_Y Temperature Coefficient register offset 0x0f
- boolean SBIST

  bits 15:9 7-bit signed integer value of Y-offset temperature coefficient.
- uint16_t CRCPAR

  bits 8:8 Startup-BIST
- uint16_t reg

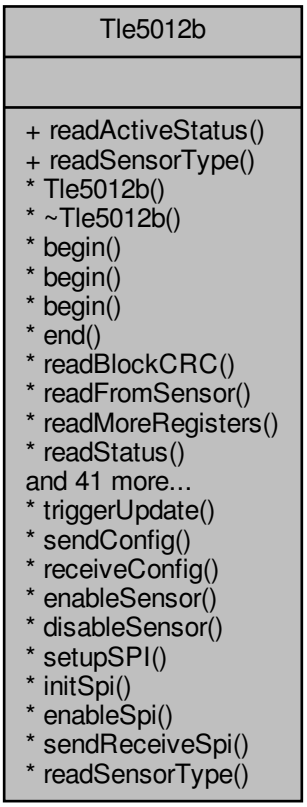
  bits 7:0 CRC of Parameters

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

4.22 Tle5012b Class Reference

Collaboration diagram for Tle5012b:



Public Member Functions

- errorTypes readActiveStatus (uint16_t &data)

Tle5012b_SPI.h - Library for Arduino for the TLE5012B angle sensor.

GMR-based angle sensor for angular position sensing in automotive applications

Author

Infineon Technologies AG

Copyright

Infineon Technologies AG

Version

1.0.1

This library include the register read and bit separation function.

- errorTypes readSensorType (uint16_t reg[ ])

Tle5012b_SPI.h - Library for Arduino for the TLE5012B angle sensor.

GMR-based angle sensor for angular position sensing in automotive applications

Author

Infineon Technologies AG

Copyright

Infineon Technologies AG

Version

1.0.1

The TLE5012B is a 360° angle sensor that detects the orientation of a magnetic field. This is achieved by measuring sine and cosine angle components with monolithic integrated Giant Magneto Resistance (iGMR) elements. These raw signals (sine and cosine) are digitally processed internally to calculate the angle orientation of the magnetic field (magnet). The TLE5012B is a pre-calibrated sensor. The calibration parameters are stored in laser fuses. At start-up the values of the fuses are written into flip-flops, where these values can be changed by the application-specific parameters. Further precision of the angle measurement over a wide temperature range and a long lifetime can be improved by enabling an optional internal autocalibration algorithm. Data communications are accomplished with a bi-directional Synchronous Serial Communication (SSC) that is SPI-compatible. The sensor configuration is stored in registers, which are accessible by the SSC interface. Additionally four other interfaces are available with the TLE5012B: Pulse-Width-Modulation (PWM) Protocol, Short-PWM-Code (SPC) Protocol, Hall Switch Mode (HSM) and Incremental Interface (IIF). These interfaces can be used in parallel with SSC or alone. Pre-configured sensor derivates with different interface settings are available. Online diagnostic functions are provided to ensure reliable operation.

- **Tle5012b** ()
- ~Tle5012b ()

  constructor sets the SPI setup
- errorTypes begin ()

  destructor stops the Sensor
- errorTypes **begin** (uint8_t cs, uint8_t en)
- errorTypes **begin** (uint8_t miso, uint8_t mosi, uint8_t sck, uint8_t cs, uint8_t en)
- void **end** ()
- errorTypes readBlockCRC ()
- errorTypes readFromSensor (uint16_t command, uint16_t &data, updTypes upd=UPD_low, safetyTypes safe=SAFE_high)
- errorTypes readMoreRegisters (uint16_t command, uint16_t data[], updTypes upd=UPD_low, safetyTypes safe=SAFE_high)
- errorTypes readStatus (uint16_t &data, updTypes upd=UPD_low, safetyTypes safe=SAFE_high)
- errorTypes readActivationStatus (uint16_t &data, updTypes upd=UPD_low, safetyTypes safe=SAFE_high)
- errorTypes **readSIL** (uint16_t &data)
- errorTypes **readIntMode1** (uint16_t &data)
- errorTypes **readIntMode2** (uint16_t &data)
- errorTypes **readIntMode3** (uint16_t &data)
- errorTypes **readIntMode4** (uint16_t &data)
- errorTypes **readSynch** (uint16_t &data)

- errorTypes **readIFAB** (uint16_t &data)
- errorTypes **readOffsetX** (uint16_t &data)
- errorTypes **readOffsetY** (uint16_t &data)
- errorTypes **readTempDMag** (uint16_t &data)
- errorTypes **readTempIIFCnt** (uint16_t &data)
- errorTypes **readTempCoeff** (uint16_t &data)
- errorTypes **readTempRaw** (uint16_t &data)
- errorTypes **readTempT25** (uint16_t &data)
- errorTypes readRawX (int16_t &data)
- errorTypes readRawY (int16_t &data)
- errorTypes getAngleRange (double &angleRange)
- errorTypes getAngleValue (double &angleValue)
- errorTypes getAngleValue (double &angleValue, int16_t &rawAnglevalue, updTypes upd=UPD_low, safetyTypes safe=SAFE_high)
- errorTypes getNumRevolutions (int16_t &numRev, updTypes upd=UPD_low, safetyTypes safe=SAFE_high)
- errorTypes getTemperature (double &temp)
- errorTypes getTemperature (double &temp, int16_t &rawTemp, updTypes upd=UPD_low, safetyTypes safe=S↩ AFE_high)
- errorTypes getAngleSpeed (double &angleSpeed)
- errorTypes getAngleSpeed (double &angleSpeed, int16_t &rawSpeed, updTypes upd=UPD_low, safetyTypes safe=SAFE_high)
- errorTypes writeToSensor (uint16_t command, uint16_t dataToWrite, bool changeCRC)
- errorTypes writeTempCoeffUpdate (uint16_t dataToWrite)
- errorTypes **writeTempCoeff** (uint16_t dataToWrite)
- errorTypes **writeActivationStatus** (uint16_t dataToWrite)
- errorTypes **writeIntMode1** (uint16_t dataToWrite)
- errorTypes **writeSIL** (uint16_t dataToWrite)
- errorTypes writeIntMode2 (uint16_t dataToWrite)
- errorTypes **writeIntMode3** (uint16_t dataToWrite)
- errorTypes **writeOffsetX** (uint16_t dataToWrite)
- errorTypes **writeOffsetY** (uint16_t dataToWrite)
- errorTypes **writeSynch** (uint16_t dataToWrite)
- errorTypes **writeIFAB** (uint16_t dataToWrite)
- errorTypes **writeIntMode4** (uint16_t dataToWrite)

Tle5012b_SPI.h - Library for Arduino for the TLE5012B angle sensor.

GMR-based angle sensor for angular position sensing in automotive applications

Author

Infineon Technologies AG

Copyright

Version

1.0.1

This library includes the 3-wire SPI connection for the TLE5012B -E1000/E5000/E9000 sensor2GO kits

- void triggerUpdate ()

  Switches the sensor off

4.22.1 Member Function Documentation

4.22.1.1 begin()

```
errorTypes Tle5012b::begin ( )
```

All these functions cover the SPI interface and should be implemented into XMC SPI wrapper. This functions use XMC structures and functions!

4.22.1.2 triggerUpdate()

```
void Tle5012b::triggerUpdate ( )
```

Triggers an update in the register buffer. This function should be triggered once before UPD registers where read as it generates a snapshot of the UPD register values at trigger point
Referenced by readFromSensor(), and writeTempCoeffUpdate().

4.22.1.3 readBlockCRC()

```
errorTypes Tle5012b::readBlockCRC ( )
```

Reads the block of _registers from addresses 08 - 0F in order to figure out the CRC.

Returns

CRC error type

4.22.1.4 readFromSensor()

```
errorTypes Tle5012b::readFromSensor (
            uint16_t command,
            uint16_t & data,
            updTypes upd = UPD_low,
            safetyTypes safe = SAFE_high )
```

General read function for reading _registers from the Tle5012b.
structure of command word, the numbers represent the bit position of the 2 byte command 15 - 0 write, 1 read 14:11 - 0000 for default operational access for addresses between 0x00 - 0x04, 1010 for configuration access for addresses between 0x05 - 0x11 10 - 0 access to current value, 1 access to value in update buffer 9:4 - access to 6 bit register address 3:0 - 4 bit number of data words.

Parameters

in	*command*	the command for reading

Parameters

out	*data*	where the data received from the _registers will be stored
in	*upd*	read from update (UPD_high) register or directly (default, UPD_low)
in	*safe*	generate safety word (default, SAFE_high) or no (SAFE_low)

Returns

CRC error type

References triggerUpdate().
Referenced by getAngleValue(), getNumRevolutions(), getTemperature(), readActivationStatus(), readRawX(), readRawY(), and readStatus().
Here is the call graph for this function:

4.22.1.5 readMoreRegisters()

```
errorTypes Tle5012b::readMoreRegisters (
            uint16_t command,
            uint16_t data[],
            updTypes upd = UPD_low,
            safetyTypes safe = SAFE_high )
```

Can be used to read 1 or more consecutive _registers, and the values used to read 1 or more than 1 consecutive _registers

Parameters

in	*command*	the command for reading
out	*data*	where the data received from the _registers will be stored
in	*upd*	read from update (UPD_high) register or directly (default, UPD_low)
in	*safe*	generate safety word (default, SAFE_high) or no (SAFE_low)

Returns

CRC error type

Referenced by getAngleSpeed(), and readSensorType().

4.22.1.6 readStatus()

```
errorTypes Tle5012b::readStatus (
            uint16_t & data,
            updTypes upd = UPD_low,
            safetyTypes safe = SAFE_high )
```

This functions reads the main status word for the sensor, mainly for checking with the additional safety word

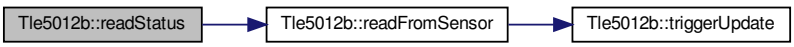
Parameters

out	*data*	pointer with the received data word
in	*upd*	read from update (UPD_high) register or directly (default, UPD_low)
in	*safe*	generate safety word (default, SAFE_high) or no (SAFE_low)

Returns

CRC error type

References readFromSensor().
Referenced by writeTempCoeffUpdate().
Here is the call graph for this function:



4.22.1.7 readActivationStatus()

```
errorTypes Tle5012b::readActivationStatus (
            uint16_t & data,
            updTypes upd = UPD_low,
            safetyTypes safe = SAFE_high )
```

This functions reads activation status word for the sensor, which held on/off information for all optional checks and additional functions

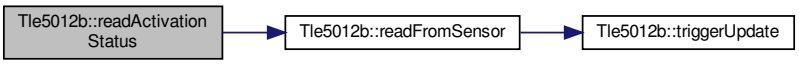
Parameters

out	*data*	pointer with the received data word
in	*upd*	read from update (UPD_high) register or directly (default, UPD_low)
in	*safe*	generate safety word (default, SAFE_high) or no (SAFE_low)

Returns

CRC error type

References readFromSensor().
Here is the call graph for this function:



4.22.1.8 readActiveStatus()

```
errorTypes Tle5012b::readActiveStatus (
            uint16_t & data )
```

The next functions are used primarily for storing the parameters and control of how the sensor works. The values stored in them are used to calculate the CRC, and their values are stored in the private component of the class, _registers.

Parameters

out	*data*	where the data received from the _registers will be stored

Returns

CRC error type

4.22.1.9 readRawX()

```
errorTypes Tle5012b::readRawX (
            int16_t & data )
```

The rawX value is signed 16 bit value

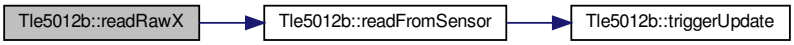
Parameters

data	pointer to 16bit word

Returns

CRC error type

References readFromSensor().
Here is the call graph for this function:



4.22.1.10 readRawY()

```
errorTypes Tle5012b::readRawY (
            int16_t & data )
```

The rawY value is signed 16 bit value

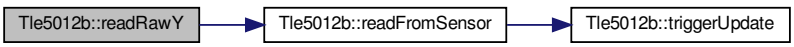
Parameters

data	pointer to 16bit word

Returns

CRC error type

References readFromSensor().
Here is the call graph for this function:



4.22.1.11 getAngleRange()

```
errorTypes Tle5012b::getAngleRange (
            double & angleRange )
```

returns the Angle Range Angle Range is stored in bytes 14 - 4 of MOD_2.

Parameters

angleRange	pointer to 16bit double value

Returns

CRC error type

4.22.1.12 getAngleValue() [1/2]

```
errorTypes Tle5012b::getAngleValue (
            double & angleValue )
```

Returns the angleValue calculated on the base of a 15 bit signed integer. However, the register returns 16 bits, so we need to do some bit arithmetic.

Parameters

in,out	*angleValue*	pointer to 16bit double angle value

Returns

CRC error type

4.22.1.13 getAngleValue() [2/2]

```
errorTypes Tle5012b::getAngleValue (
            double & angleValue,
            int16_t & rawAnglevalue,
            updTypes upd = UPD_low,
            safetyTypes safe = SAFE_high )
```

Same function as before but also returns a pointer to the raw data

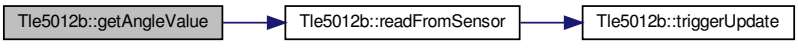
Parameters

in,out	*angleValue*	pointer to 16bit double angle value
in,out	*rawAnglevalue*	point to an int16_t raw data value
in	*upd*	read from update (UPD_high) register or directly (default, UPD_low)
in	*safe*	generate safety word (default, SAFE_high) or no (SAFE_low)

Returns

CRC error type

References readFromSensor().
Here is the call graph for this function:



4.22.1.14 getNumRevolutions()

```
errorTypes Tle5012b::getNumRevolutions (
            int16_t & numRev,
            updTypes upd = UPD_low,
            safetyTypes safe = SAFE_high )
```

Returns the number of revolutions done from the angle value which is a 9 bit signed integer. However, the register returns 16 bits, so we need to do some bit arithmetic. Therefore the resulting revolution can b only between -256 $<$ numRev $<$ 256 and it will switch from positive to negative and vice versa values at the borders.

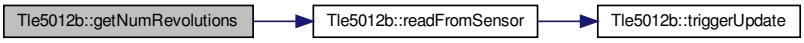
Parameters

in,out	*numRev*	pointer to 16bit word for the number of revolutions
in	*upd*	read from update (UPD_high) register or directly (default, UPD_low)
in	*safe*	generate safety word (default, SAFE_high) or no (SAFE_low)

Returns

CRC error type

References readFromSensor().
Here is the call graph for this function:



4.22.1.15 getTemperature() [1/2]

```
errorTypes Tle5012b::getTemperature (
            double & temp )
```

Return the temperature. The temperature value is a 9 bit signed integer. However, the register returns 16 bits, so we need to do some bit arithmetic.

Parameters

in,out	*temp*	pointer to 16bit double value of the temperature

Returns

CRC error type

4.22.1.16 getTemperature() [2/2]

```
errorTypes Tle5012b::getTemperature (
            double & temp,
```

```
            int16_t & rawTemp,
            updTypes upd = UPD_low,
            safetyTypes safe = SAFE_high )
```

Same as above but also returns a pointer to the raw data

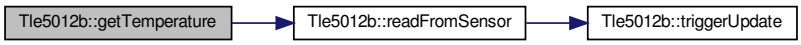
Parameters

in,out	*temp*	pointer to 16bit double value of the temperature
in,out	*rawTemp*	pointer to int16_t raw value data
in	*upd*	read from update (UPD_high) register or directly (default, UPD_low)
in	*safe*	generate safety word (default, SAFE_high) or no (SAFE_low)

Returns

CRC error type

References readFromSensor().
Here is the call graph for this function:



4.22.1.17 getAngleSpeed() [1/2]

```
errorTypes Tle5012b::getAngleSpeed (
            double & angleSpeed )
```

Returns the calculated angle speed. The angle speed is a 15 bit signed integer, however, the register returns 16 bits, so we need to do some bit arithmetic.

Parameters

in,out	*angleSpeed*	pointer to 16bit double value

Returns

CRC error type

4.22.1.18 getAngleSpeed() [2/2]

```
errorTypes Tle5012b::getAngleSpeed (
            double & angleSpeed,
            int16_t & rawSpeed,
            updTypes upd = UPD_low,
            safetyTypes safe = SAFE_high )
```

Same as above but also returns a pointer to the raw data

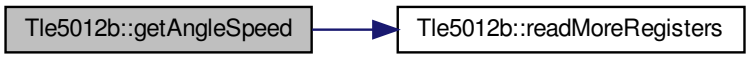
Parameters

in,out	*angleSpeed*	angleSpeed pointer to 16bit double value
in,out	*rawSpeed*	pointer to int16_t raw value data
in	*upd*	read from update (UPD_high) register or directly (default, UPD_low)
in	*safe*	generate safety word (default, SAFE_high) or no (SAFE_low)

Returns

CRC error type

References readMoreRegisters().
Here is the call graph for this function:



4.22.1.19 writeToSensor()

```
errorTypes Tle5012b::writeToSensor (
            uint16_t command,
            uint16_t dataToWrite,
            bool changeCRC )
```

General write function for writing _registers from the Tle5012b.

Parameters

in	*command*	the command to execute the write
in	*dataToWrite*	the new data that will be written to the register
in	*changeCRC*	the registerIndex helps figure out in which register the value changed, so that we don't need to read all the register again to calculate the CRC

Returns

CRC error type

Referenced by writeIntMode2().

4.22.1.20 writeTempCoeffUpdate()

```
errorTypes Tle5012b::writeTempCoeffUpdate (
```

```
            uint16_t dataToWrite )
```

This function is used in order to update the CRC in the register 0F(second byte)

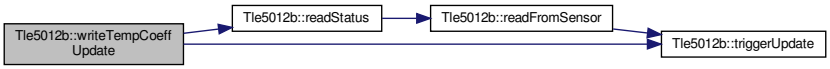
Parameters

in	*dataToWrite*	the new data that will be written to the register

Returns

CRC error type

References readStatus(), and triggerUpdate().
Here is the call graph for this function:



4.22.1.21 writeIntMode2()

```
errorTypes Tle5012b::writeIntMode2 (
            uint16_t dataToWrite )
```

The Interface Mode 2 register stores the following values

- angle range from bit 14 - 4, where 0x200 is 90° (-45° to 45°) and 0x80 is 360°(-180° to 180°). The calculation is based on the formula (360 ∗ (2^7 / 2^9))
- angle direction in bit 3, 0 = counterclockwise rotation of magnet and 1 = clockwise rotation of magnet
- prediction in bit 2, where 0 = prediction disabled and 1 = prediction enabled
- Autocalibration mode in bits 1 - 0, where 00 = no autocalibration mode, 01 = autocalibartion mode 1, 10 = autocalibration mode 2, 11 = autocalibration mode 3

Be careful when changing the values of this register. If the angle range is changed to 0x80 and the angle value exceeds the valid range of -45 to 45, you will get a DSPU overflow error, and the safety word will show a system error. Furthermore, autocalibration only works with the angle range of 0x80, so if you change the angle range in autocalibration mode, then an error will occur.
References writeToSensor().
Here is the call graph for this function:

4.22.1.22 readSensorType()

```
errorTypes Tle5012b::readSensorType (
            uint16_t reg[] )
```

Parameters

in,out	*reg*	

Returns

CRC error type

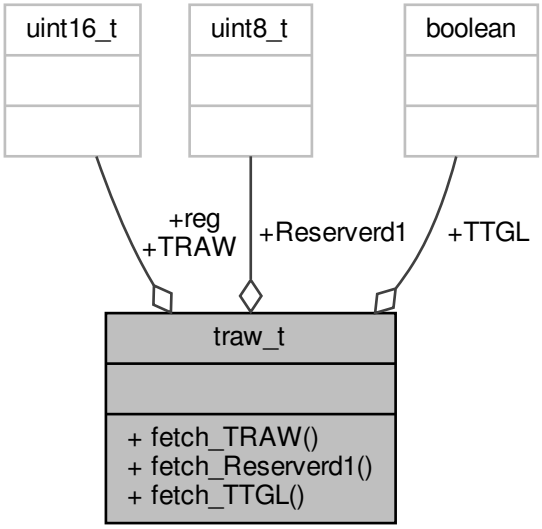
References adc_t::ADCX, adc_t::ADCY, readMoreRegisters(), stat_t::reg, acstat_t::reg, aval_t::reg, aspd_t::reg, arev_t::reg, fsync_t::reg, mod1_t::reg, sil_t::reg, mod2_t::reg, mod3_t::reg, offx_t::reg, offy_t::reg, synch_t::reg, ifab_t::reg, mod4_t::reg, tcoy_t::reg, dmag_t::reg, traw_t::reg, iifcnt_t::reg, and t250_t::reg.
Here is the call graph for this function:

The documentation for this class was generated from the following files:

- src/Tle5012b.h
- src/Tle5012b.cpp
- src/util/Tle5012b_conf.cpp
- src/util/Tle5012b_SPI.cpp

4.23 traw_t Struct Reference

Collaboration diagram for traw_t:



Public Member Functions

- boolean fetch_TRAW (uint16_t reg)

  the register value
- uint8_t **fetch_Reserverd1** (uint16_t reg)
- uint16_t **fetch_TTGL** (uint16_t reg)

Data Fields

- boolean TTGL

  T_RAW temperature raw data offset 0x15
- uint8_t Reserverd1

  bits 15:15 Temperature Sensor Raw-Value Toggle toggles after every new temperature value
- uint16_t TRAW

  bits 14:10
- uint16_t reg

  bits 9:0 Temperature Sensor Raw-Value at ADC without offset

The documentation for this struct was generated from the following file:

- src/util/Tle5012b_conf.h

information given in this document with respect to such application.

Trademarks of Infineon Technologies AG

µHVIC™, µIPM™, µPFC™, AU-ConvertIR™, AURIX™, C166™, CanPAK™, CIPOS™, CIPURSE™, CoolDP™, CoolGaN™, COOLiR™, CoolMOS™, CoolSET™, CoolSiC™, DAVE™, DI-POL™, DirectFET™, DrBlade™, EasyPIM™, EconoBRIDGE™, EconoDUAL™, EconoPACK™, EconoPIM™, EiceDRIVER™, eupec™, FCOS™, GaNpowIR™, HEXFET™, HITFET™, HybridPACK™, iMOTION™, IRAM™, ISOFACE™, IsoPACK™, LEDrivIR™, LITIX™, MIPAQ™, ModSTACK™, my-d™, NovalithIC™, OPTIGA™, OptiMOS™, ORIGA™, PowIRaudio™, PowIRStage™, PrimePACK™, PrimeSTACK™, PROFET™, PRO-SIL™, RASIC™, REAL3™, SmartLEWIS™, SOLID FLASH™, SPOC™, StrongIRFET™, SupIRBuck™, TEMPFET™, TRENCHSTOP™, TriCore™, UHVIC™, XHP™, XMC™.

Trademarks Update 2015-12-22

Other Trademarks

All referenced product or service names and trademarks are the property of their respective owners.