Infineon Arduino Library Documentation

Author: Infineon Technologies AG
Date: July 2, 2018

Contents

1 Arduino Library for Infineon's Magnetic 3D Sensor TLE493D-W2B6 **3**
- 1.1 Introduction . . . 3
- 1.2 Main Features . . . 3
  - 1.2.1 Interrupts . . . 3
  - 1.2.2 Collision Avoidance and Clock Stretching . . . 3
  - 1.2.3 Wake Up Mode . . . 4

2 Bug List **5**

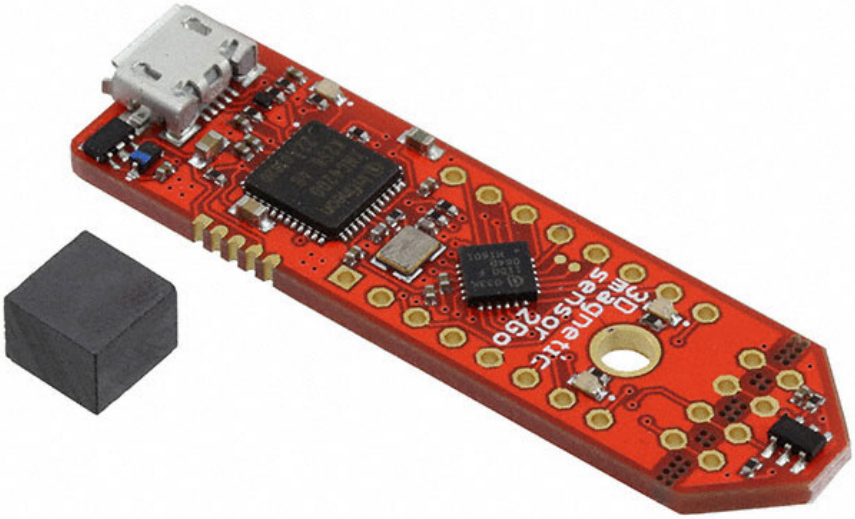
3 Data Structure Documentation **6**
- 3.1 tle493d_w2b6::BusInterface_t Struct Reference . . . 6
- 3.2 tle493d_w2b6::RegMask_t Struct Reference . . . 6
- 3.3 Tle493d_w2b6 Class Reference . . . 8
  - 3.3.1 Member Enumeration Documentation . . . 10
    - 3.3.1.1 AccessMode_e . . . 10
    - 3.3.1.2 TypeAddress_e . . . 11
  - 3.3.2 Constructor & Destructor Documentation . . . 11
    - 3.3.2.1 Tle493d_w2b6() . . . 11
    - 3.3.2.2 ~Tle493d_w2b6() . . . 11
  - 3.3.3 Member Function Documentation . . . 11
    - 3.3.3.1 begin() [1/2] . . . 11
    - 3.3.3.2 begin() [2/2] . . . 11
    - 3.3.3.3 calcParity() . . . 12
    - 3.3.3.4 concatResults() . . . 12
    - 3.3.3.5 disableCollisionAvoidance() . . . 12
    - 3.3.3.6 disableInterrupt() . . . 12
    - 3.3.3.7 disableTemp() . . . 13
    - 3.3.3.8 enableCollisionAvoidance() . . . 13
    - 3.3.3.9 enableInterrupt() . . . 13
    - 3.3.3.10 enableTemp() . . . 13
    - 3.3.3.11 getAzimuth() . . . 13
    - 3.3.3.12 getNorm() . . . 13
    - 3.3.3.13 getPolar() . . . 14
    - 3.3.3.14 getRegBits() . . . 14
    - 3.3.3.15 getTemp() . . . 14
    - 3.3.3.16 getX() . . . 14
    - 3.3.3.17 getY() . . . 15
    - 3.3.3.18 getZ() . . . 15
    - 3.3.3.19 resetSensor() . . . 15
    - 3.3.3.20 setAccessMode() . . . 15
    - 3.3.3.21 setRegBits() . . . 15
    - 3.3.3.22 setUpdateRate() . . . 16
    - 3.3.3.23 setWakeUpThreshold() . . . 16
    - 3.3.3.24 updateData() . . . 17
    - 3.3.3.25 wakeUpEnabled() . . . 17

Index **18**

1 Arduino Library for Infineon's Magnetic 3D Sensor TLE493D-W2B6

1.1 Introduction

The TLE493D-W2B6 sensor family comes with I2C interface and wake-up function. This sensor family TLE493D offers accurate three dimensional sensing with extremely low-power consumption.



1.2 Main Features

1.2.1 Interrupts

Interrupts can be sent from the sensor to the microcontroller to notify the completion of a measurement and its ADC conversion. Values read directly after interrupts are guaranteed to be consistent. Without interrupts values read might be stale.

1.2.2 Collision Avoidance and Clock Stretching

In case of a collision, the sensor interrupt disturbs the I2C clock, causing an additional SCL pulse which shifts the data read out by one bit. If collision avoidance is activated, the sensor monitors the start/stop conditions, and suppresses interrupts in between.

When interrupts are disabled, this feature becomes clock stretching, that is, the data read out only starts after the ADC conversion is finished. Thus it can be avoided that during an ADC conversion old or corrupted measurement results are read out, which may occur when the ADC is writing to a register while this is being read out by the microcontroller. When clock stretching is enabled, the sensor pulls the SCL line down during ongoing ADC conversions, reading of sensor registers or the transmission of valid ACKs.

Two register bits (CA and INT) work together for different configurations. APIs to modify these two bits are not offered since they have to be set according to different operating modes (master controlled mode, low power mode and fast mode) for the sensor to work.

1.2.3 Wake Up Mode

Wake up mode is intended to be used with low power mode or fast mode. This mode disables interrupts within a user-specified range, so that interrupts are generated only when relevant data are available.

2 Bug List

File Tle493d_w2b6.cpp

reset freezes the sensor, fast mode not working

User manual recommands INT=0 in fast mode however only disabling INT works

wake up mode not configured correctly (WA bit = 0)

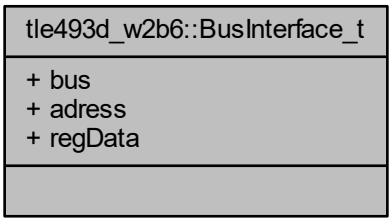
Global Tle493d_w2b6::resetSensor (void)

not working

3 Data Structure Documentation

3.1 tle493d_w2b6::BusInterface_t Struct Reference

Collaboration diagram for tle493d_w2b6::BusInterface_t:

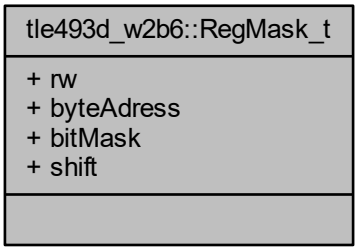


Data Fields

- TwoWire * **bus**
- uint8_t **adress**
- uint8_t **regData** [TLE493D_W2B6_NUM_REG]

3.2 tle493d_w2b6::RegMask_t Struct Reference

Collaboration diagram for tle493d_w2b6::RegMask_t:

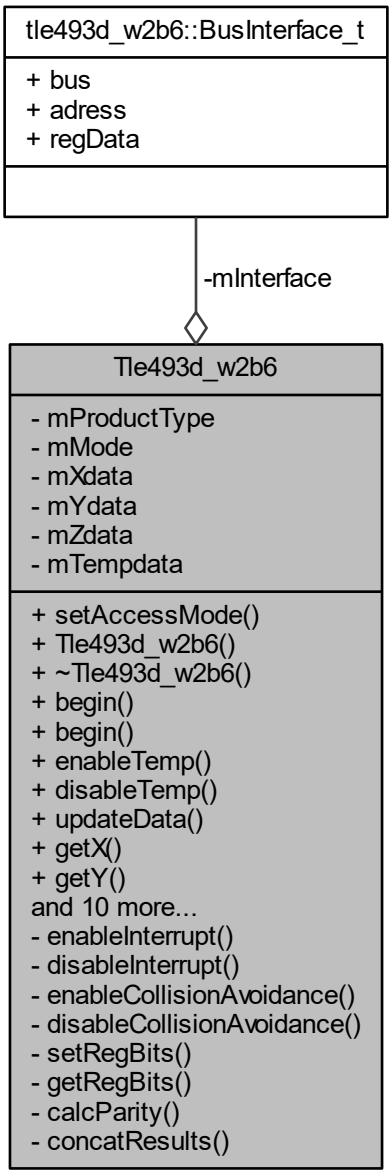


Data Fields

- uint8_t **rw**
- uint8_t **byteAdress**
- uint8_t **bitMask**
- uint8_t **shift**

3.3 Tle493d_w2b6 Class Reference

Collaboration diagram for Tle493d_w2b6:



Public Types

- enum TypeAddress_e { **TLE493D_W2B6_A0** = 0x35, **TLE493D_W2B6_A1** = 0x22, **TLE493D_W2B6_A2** = 0x78, **TLE493D_W2B6_A3** = 0x44 }
- enum AccessMode_e { **LOWPOWERMODE** = 0, **MASTERCONTROLLEDMODE** = 1, **FASTMODE** = 3 }

Public Member Functions

- bool setAccessMode (AccessMode_e mode)

  Sets the operating mode of the sensor.
- Tle493d_w2b6 (AccessMode_e mode=MASTERCONTROLLEDMODE, TypeAddress_e productType=TL↩E493D_W2B6_A0)

  Constructor of the sensor class.
- ~Tle493d_w2b6 (void)

  Destructor.
- void begin (void)

  Starts the sensor.
- void begin (TwoWire &bus, TypeAddress_e slaveAddress, bool reset, uint8_t oneByteRead)

  Starts the sensor.
- void enableTemp (void)

  Enables temperature measurement; by default already enabled.
- void disableTemp (void)

  Disables temperature measurement to reduce power consumption.
- Tle493d_w2b6_Error updateData (void)

  Reads measurement results from sensor.
- float getX (void)
- float getY (void)
- float getZ (void)
- float getNorm (void)
- float getAzimuth (void)
- float getPolar (void)
- float getTemp (void)
- void setWakeUpThreshold (float xh, float xl, float yh, float yl, float zh, float zl)

  The Wake Up threshold range disabling /INT pulses between upper threshold and lower threshold is limited to a window of the half output range. Here the adjustable range can be set with a ratio of size [-1,1] When all the measurement values Bx, By and Bz are within this range INT is disabled. If the arguments are out of range or any upper threshold is smaller than the lower one, the function returns without taking effect.
- void setUpdateRate (uint8_t updateRate)

  Sets the update rate in low power mode.
- void resetSensor (void)

  Resets the sensor.
- void **readDiagnosis** (uint8_t(&diag)[7])
- bool wakeUpEnabled (void)

  Checks if WA bit is set. When not interrupt configuration is as specified by the CA and INT bits.

Private Member Functions

- void enableInterrupt (void)

Enables interrupts.

- void disableInterrupt (void)

  Disables interrupts; Sensor read-out suppressed during ongoing ADC conversion.

- void enableCollisionAvoidance (void)

  Enables collision avoidance (clock stretching)

- void disableCollisionAvoidance (void)

  Disables collision avoidance; readouts may collide with ADC conversion.

- void setRegBits (uint8_t regMaskIndex, uint8_t data)

  Stores new values into the bus interface; for this function to take effect the function writeOut() should be called afterwards.

- uint8_t getRegBits (uint8_t regMaskIndex)

  Returns the value of a register field.

- void calcParity (uint8_t regMaskIndex)

  Sets FP (fuse parity) and CP (configuration parity)

- int16_t concatResults (uint8_t upperByte, uint8_t lowerByte, bool isB)

  Concatenates the upper bits and lower bits of magnetic or temperature measurements.

Private Attributes

- tle493d_w2b6::BusInterface_t **mInterface**
- const TypeAddress_e **mProductType**
- AccessMode_e **mMode**
- int16_t **mXdata**
- int16_t **mYdata**
- int16_t **mZdata**
- int16_t **mTempdata**

3.3.1 Member Enumeration Documentation

3.3.1.1 AccessMode_e

`enum Tle493d_w2b6::AccessMode_e`

Enumerates the three available modes; number 2 is reserved In low power mode cyclic measurements and ADC-conversions are carried out with a update rate; the wake-up function is already configured for this mode, so that the sensor can continue making magnetic field measurements. With this configuration the microcontroller will only consume power and access the sensor if relevant measurement data is available. In master controlled mode the sensor powered down if when it is not triggered. This library configures to ADC start before sending first MSB of data registers In fast mode the measurements and ADC-conversions are running continuously.

3.3.1.2 TypeAddress_e

```
enum Tle493d_w2b6::TypeAddress_e
```

Defines four types of the sensor family which are supported by this library and their corresponding addresses The addresses can be concatenated with 0 or 1 for reading or writing

3.3.2 Constructor & Destructor Documentation

3.3.2.1 Tle493d_w2b6()

```
Tle493d_w2b6::Tle493d_w2b6 (
            AccessMode_e mode = MASTERCONTROLLEDMODE,
            TypeAddress_e productType = TLE493D_W2B6_A0 )
```

Constructor of the sensor class.

Parameters

mode	Operating mode of the sensor; default is the master controlled mode
productType	The library supports product types from A0 to A3; default is type A0

3.3.2.2 ~Tle493d_w2b6()

```
Tle493d_w2b6::~Tle493d_w2b6 (
            void  )
```

Destructor.

3.3.3 Member Function Documentation

3.3.3.1 begin() [1/2]

```
void Tle493d_w2b6::begin (
            void  )
```

Starts the sensor.

3.3.3.2 begin() [2/2]

```
void Tle493d_w2b6::begin (
            TwoWire & bus,
```

```
            TypeAddress_e slaveAddress,
            bool reset,
            uint8_t oneByteRead )
```

Starts the sensor.

Parameters

bus	The I2C bus
slaveAddress	The 7-bit slave address as defined in Tle493d_Type
reset	If a reset should be initiated before starting the sensor
oneByteRead	If one-byte read protocol should be used. Otherwise the two-byte protocol is available

3.3.3.3 calcParity()

```
void Tle493d_w2b6::calcParity (
            uint8_t regMaskIndex )  [private]
```

Sets FP (fuse parity) and CP (configuration parity)

3.3.3.4 concatResults()

```
int16_t Tle493d_w2b6::concatResults (
            uint8_t upperByte,
            uint8_t lowerByte,
            bool isB )  [private]
```

Concatenates the upper bits and lower bits of magnetic or temperature measurements.

3.3.3.5 disableCollisionAvoidance()

```
void Tle493d_w2b6::disableCollisionAvoidance (
            void  )  [private]
```

Disables collision avoidance; readouts may collide with ADC conversion.

3.3.3.6 disableInterrupt()

```
void Tle493d_w2b6::disableInterrupt (
            void  )  [private]
```

Disables interrupts; Sensor read-out suppressed during ongoing ADC conversion.

3.3.3.7 disableTemp()

```
void Tle493d_w2b6::disableTemp (
            void  )
```

Disables temperature measurement to reduce power consumption.

3.3.3.8 enableCollisionAvoidance()

```
void Tle493d_w2b6::enableCollisionAvoidance (
            void  )  [private]
```

Enables collision avoidance (clock stretching)

3.3.3.9 enableInterrupt()

```
void Tle493d_w2b6::enableInterrupt (
            void  )  [private]
```

Enables interrupts.

3.3.3.10 enableTemp()

```
void Tle493d_w2b6::enableTemp (
            void  )
```

Enables temperature measurement; by default already enabled.

3.3.3.11 getAzimuth()

```
float Tle493d_w2b6::getAzimuth (
            void  )
```

Returns

the Azimuth angle arctan(y/x)

3.3.3.12 getNorm()

```
float Tle493d_w2b6::getNorm (
            void  )
```

Returns

norm of the magnetic field vector sqrt(x^2 + y^2 + z^2)

3.3.3.13 getPolar()

```
float Tle493d_w2b6::getPolar (
            void  )
```

Returns

the angle in polar coordinates arctan(z/(sqrt(x^2+y^2)))

3.3.3.14 getRegBits()

```
uint8_t Tle493d_w2b6::getRegBits (
            uint8_t regMaskIndex )  [private]
```

Returns the value of a register field.

Parameters

Register	mask index as defined in Registers_e

3.3.3.15 getTemp()

```
float Tle493d_w2b6::getTemp (
            void  )
```

Returns

the temperature value

3.3.3.16 getX()

```
float Tle493d_w2b6::getX (
            void  )
```

Returns

the Cartesian x-coordinate

3.3.3.17 getY()

```
float Tle493d_w2b6::getY (
            void  )
```

Returns

the Cartesian y-coordinate

3.3.3.18 getZ()

```
float Tle493d_w2b6::getZ (
            void  )
```

Returns

the Cartesian z-coordinate

3.3.3.19 resetSensor()

```
void Tle493d_w2b6::resetSensor (
            void  )
```

Resets the sensor.

Bug not working

3.3.3.20 setAccessMode()

```
bool Tle493d_w2b6::setAccessMode (
            AccessMode_e mode )
```

Sets the operating mode of the sensor.

Parameters

mode	MASTERCONTROLLEDMODE,LOWPOWERMODE or FASTMODE, default is MASTERCONTROLLEDMODE

3.3.3.21 setRegBits()

```
void Tle493d_w2b6::setRegBits (
            uint8_t regMaskIndex,
            uint8_t data )  [private]
```

Stores new values into the bus interface; for this function to take effect the function writeOut() should be called afterwards.

Parameters

Register	mask index as defined in Registers_e
Value	to be written into the register field specified by the register index

3.3.3.22 setUpdateRate()

```
void Tle493d_w2b6::setUpdateRate (
            uint8_t updateRate )
```

Sets the update rate in low power mode.

Parameters

updateRate	Update rate which is an unsigned integer from the 0 (the fastest) to 7 (the highest)

3.3.3.23 setWakeUpThreshold()

```
void Tle493d_w2b6::setWakeUpThreshold (
            float xh,
            float xl,
            float yh,
            float yl,
            float zh,
            float zl )
```

The Wake Up threshold range disabling /INT pulses between upper threshold and lower threshold is limited to a window of the half output range. Here the adjustable range can be set with a ratio of size [-1,1] When all the measurement values Bx, By and Bz are within this range INT is disabled. If the arguments are out of range or any upper threshold is smaller than the lower one, the function returns without taking effect.

Parameters

xh	Upper threshold in x direction
xl	Lower threshold in x direction
yh	Upper threshold in y direction
yl	Lower threshold in y direction
zh	Upper threshold in z direction
zl	Lower threshold in z direction

3.3.3.24 updateData()

```
Tle493d_w2b6_Error_t Tle493d_w2b6::updateData (
            void  )
```

Reads measurement results from sensor.

3.3.3.25 wakeUpEnabled()

```
bool Tle493d_w2b6::wakeUpEnabled (
            void  )
```

Checks if WA bit is set. When not interrupt configuration is as specified by the CA and INT bits.

Index

~Tle493d_w2b6
    Tle493d_w2b6, 11

AccessMode_e
    Tle493d_w2b6, 10

begin
    Tle493d_w2b6, 11

calcParity
    Tle493d_w2b6, 12
concatResults
    Tle493d_w2b6, 12

disableCollisionAvoidance
    Tle493d_w2b6, 12
disableInterrupt
    Tle493d_w2b6, 12
disableTemp
    Tle493d_w2b6, 12

enableCollisionAvoidance
    Tle493d_w2b6, 13
enableInterrupt
    Tle493d_w2b6, 13
enableTemp
    Tle493d_w2b6, 13

getAzimuth
    Tle493d_w2b6, 13
getNorm
    Tle493d_w2b6, 13
getPolar
    Tle493d_w2b6, 14
getRegBits
    Tle493d_w2b6, 14
getTemp
    Tle493d_w2b6, 14
getX
    Tle493d_w2b6, 14
getY
    Tle493d_w2b6, 14
getZ
    Tle493d_w2b6, 15

resetSensor
    Tle493d_w2b6, 15

setAccessMode
    Tle493d_w2b6, 15
setRegBits
    Tle493d_w2b6, 15
setUpdateRate
    Tle493d_w2b6, 16
setWakeUpThreshold
    Tle493d_w2b6, 16

Tle493d_w2b6, 8
    ~Tle493d_w2b6, 11
    AccessMode_e, 10
    begin, 11
    calcParity, 12
    concatResults, 12
    disableCollisionAvoidance, 12
    disableInterrupt, 12
    disableTemp, 12
    enableCollisionAvoidance, 13
    enableInterrupt, 13
    enableTemp, 13
    getAzimuth, 13
    getNorm, 13
    getPolar, 14
    getRegBits, 14
    getTemp, 14
    getX, 14
    getY, 14
    getZ, 15
    resetSensor, 15
    setAccessMode, 15
    setRegBits, 15
    setUpdateRate, 16
    setWakeUpThreshold, 16
    Tle493d_w2b6, 11
    TypeAddress_e, 10
    updateData, 16
    wakeUpEnabled, 17
tle493d_w2b6::BusInterface_t, 6
tle493d_w2b6::RegMask_t, 6
TypeAddress_e
    Tle493d_w2b6, 10

updateData
    Tle493d_w2b6, 16

wakeUpEnabled
    Tle493d_w2b6, 17

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.

IMPORTANT NOTICE

The information contained in this application note is given as a hint for the implementation of the product only and shall in no event be regarded as a description or warranty of a certain functionality, condition or quality of the product. Before implementation of the product, the recipient of this application note must verify any function and other technical information given herein in the real application. Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind (including without limitation warranties of non-infringement of intellectual property rights of any third party) with respect to any and all information given in this application note.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

WARNINGS

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury