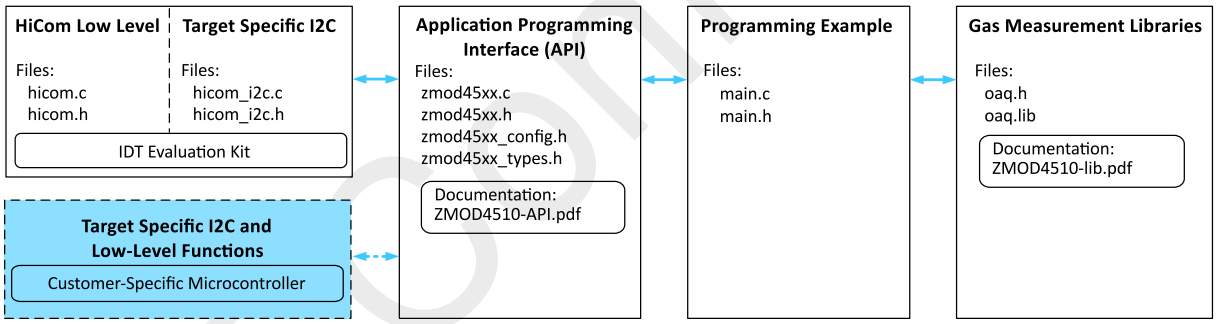
ZMOD4510 API Documentation

Contents

Chapter 1

ZMOD4510 Application Programming Interface Overview

This document refers to the IDT document *ZMOD4510 Programming Manual - Read Me*. The figure below shows an overview of the ZMOD4510 API, programming example and libraries. Custom microcontrollers can be used to establish I2C communication. Using the user's own microcontroller requires implementing the user's own target-specific I2C and low-level functions (highlighted in light blue). The following describes in detail the Application Programming Interface (API) of the ZMOD4510.



Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

Chapter 3

Data Structure Index

3.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 5

Module Documentation

5.1 Gas sensor IDs

Macros

- #define **ZMOD4510_PID** (0x6320)

5.1.1 Detailed Description

The gas sensor product IDs.

5.2 Error codes

Macros

- #define ERROR_INIT_OUT_OF_RANGE (1)
- #define ERROR_GAS_TIMEOUT (2)
- #define ERROR_I2C (3)
- #define ERROR_SENSOR_UNSUPPORTED (4)
- #define ERROR_CONFIG_MISSING (5)
- #define ERROR_SENSOR (6)
- #define ERROR_ACCESS_CONFLICT (7)
- #define ERROR_POR_EVENT (8)

5.2.1 Detailed Description

The gas sensor and API error codes.

5.2.2 Macro Definition Documentation

5.2.2.1 #define ERROR_ACCESS_CONFLICT (7)

AccessConflict.

5.2.2.2 #define ERROR_CONFIG_MISSING (5)

There is no pointer to a valid configuration.

5.2.2.3 #define ERROR_GAS_TIMEOUT (2)

The operation took too long.

5.2.2.4 #define ERROR_I2C (3)

Failure in i2c communication.

5.2.2.5 #define ERROR_INIT_OUT_OF_RANGE (1)

The initialize value is out of range.

5.2.2.6 #define ERROR_POR_EVENT (8)

POR_event.

5.2.2.7 #define ERROR_SENSOR (6)

Sensor malfunction.

5.2.2.8 #define ERROR_SENSOR_UNSUPPORTED (4)

Sensor is not supported with this firmware.

Chapter 6

Data Structure Documentation

6.1 zmod45xx_conf Struct Reference

Structure to hold the gas sensor module configuration.

```
#include <zmod45xx_types.h>
```

Data Fields

- uint8_t **start**
- zmod45xx_conf_str **h**
- zmod45xx_conf_str **d**
- zmod45xx_conf_str **m**
- zmod45xx_conf_str **s**
- zmod45xx_conf_str **r**

6.1.1 Detailed Description

Structure to hold the gas sensor module configuration.

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

- zmod45xx_types.h

6.2 zmod45xx_conf_str Struct Reference

A single data set for the configuration.

```
#include <zmod45xx_types.h>
```

Data Fields

- uint8_t **addr**
- uint8_t **len**
- const uint8_t * **data**

6.2.1 Detailed Description

A single data set for the configuration.

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

- zmod45xx_types.h

6.3 zmod45xx_dev_t Struct Reference

Device structure ZMOD45xx.

```
#include <zmod45xx_types.h>
```

Data Fields

- uint8_t i2c_addr
- uint8_t config [6]
- uint8_t general_purpose [9]
- uint16_t mox_er
- uint16_t mox_lr
- uint16_t pid
- zmod45xx_i2c_ptr_t read
- zmod45xx_i2c_ptr_t write
- zmod45xx_delay_ptr_p delay_ms
- const zmod45xx_conf * init_conf
- const zmod45xx_conf * meas_conf

6.3.1 Detailed Description

Device structure ZMOD45xx.

6.3.2 Field Documentation

6.3.2.1 uint8_t config[6]

configuration parameter set

6.3.2.2 zmod45xx_delay_ptr_p delay_ms

function pointer to delay function

6.3.2.3 uint8_t general_purpose[9]

general purpose data

6.3.2.4 uint8_t i2c_addr

i2c address of the sensor

6.3.2.5 const zmod45xx_conf* init_conf

pointer to the initialize configuration

6.3.2.6 const zmod45xx_conf* meas_conf

pointer to the measurement configuration

6.3.2.7 uint16_t mox_er

sensor specific parameter

6.3.2.8 uint16_t mox_lr

sensor specific parameter

6.3.2.9 uint16_t pid

product id of the sensor

6.3.2.10 zmod45xx_i2c_ptr_t read

function pointer to i2c read

6.3.2.11 zmod45xx_i2c_ptr_t write

function pointer to i2c write

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

- zmod45xx_types.h

Chapter 7

File Documentation

7.1 zmod45xx.c File Reference

ZMOD45xx functions.

```
#include "zmod45xx.h"
#include "zmod45xx_config.h"
```

Functions

- int8_t zmod45xx_read_sensor_info (zmod45xx_dev_t *dev)

  Read sensor parameter.
- int8_t **zmod45xx_calc_factor** (zmod45xx_dev_t *dev, float factor, uint8_t *data)
- int8_t zmod45xx_init_sensor (zmod45xx_dev_t *dev)

  Initialize the sensor after power on.
- int8_t zmod45xx_init_measurement (zmod45xx_dev_t *dev)

  Initialize the sensor for zmod4510 measurement.
- int8_t zmod45xx_start_measurement (zmod45xx_dev_t *dev)

  Start the measurement.
- int8_t zmod45xx_read_status (zmod45xx_dev_t *dev, uint8_t *status)

  Read the status of the device.
- int8_t zmod45xx_read_adc_results (zmod45xx_dev_t *dev, uint8_t *adc_result)

  Read adc values from the sensor.
- int8_t zmod45xx_calc_rmox (zmod45xx_dev_t *dev, uint8_t *adc_result, float *rmox)

  Calculate mox resistance.

7.1.1 Detailed Description

ZMOD45xx functions.

Version

2.0.0

Date

2019-10-02

Author

IDT

7.1.2 Function Documentation

7.1.2.1 int8_t zmod45xx_calc_rmox (zmod45xx_dev_t * *dev*, uint8_t * *adc_result*, float * *rmox*)

Calculate mox resistance.

Parameters

in	*dev*	pointer to the device
in,out	*adc_result*	pointer to the adc results
in,out	*rmox*	pointer to the rmox values

Returns

error code

Return values

0	success
!= 0	error

7.1.2.2 int8_t zmod45xx_init_measurement (zmod45xx_dev_t * *dev*)

Initialize the sensor for zmod4510 measurement.

Parameters

in	*dev*	pointer to the device

Returns

error code

Return values

0	success
!= 0	error

7.1.2.3 int8_t zmod45xx_init_sensor (zmod45xx_dev_t * *dev*)

Initialize the sensor after power on.

Parameters

in	*dev*	pointer to the device

Returns

error code

Return values

0	success
!= 0	error

7.1.2.4 int8_t zmod45xx_read_adc_results (zmod45xx_dev_t * *dev*, uint8_t * *adc_result*)

Read adc values from the sensor.

Parameters

in	*dev*	pointer to the device
in, out	*adc_result*	pointer to the adc results

Returns

error code

Return values

0	success
!= 0	error

7.1.2.5 int8_t zmod45xx_read_sensor_info (zmod45xx_dev_t * *dev*)

Read sensor parameter.

Parameters

in	*dev*	pointer to the device

Returns

error code

Return values

0	success
!= 0	error

Note

This function must be called once before running other sensor functions.

7.1.2.6 int8_t zmod45xx_read_status (zmod45xx_dev_t * *dev*, uint8_t * *status*)

Read the status of the device.

Parameters

in	*dev*	pointer to the device
in, out	*status*	pointer to the status variable

Returns

error code

Return values

0	success
!= 0	error

7.1.2.7 int8_t zmod45xx_start_measurement (zmod45xx_dev_t * *dev*)

Start the measurement.

Parameters

in	*dev*	pointer to the device

Returns

error code

Return values

0	success
!= 0	error

7.2 zmod45xx.h File Reference

ZMOD45xx functions.

```
#include "zmod45xx_types.h"
```

Macros

- #define **ZMOD4510_I2C_ADDRESS** (0x33)
- #define **ZMOD45XX_ADDR_PID** (0x00)
- #define **ZMOD45XX_ADDR_CONF** (0x20)
- #define **ZMOD45XX_ADDR_GENERAL_PURPOSE** (0x26)
- #define **ZMOD45XX_ADDR_CMD** (0x93)
- #define **ZMOD45XX_ADDR_STATUS** (0x94)
- #define **ZMOD45XX_ADDR_TRACKING** (0x3A)
- #define **ZMOD45XX_LEN_PID** (2)
- #define **ZMOD45XX_LEN_CONF** (6)
- #define **ZMOD45XX_LEN_TRACKING** (6)
- #define **ZMOD45XX_LEN_GENERAL_PURPOSE** (9)
- #define STATUS_SEQUENCER_RUNNING_MASK (0x80)
- #define STATUS_SLEEP_TIMER_ENABLED_MASK (0x40)
- #define STATUS_ALARM_MASK (0x20)
- #define STATUS_LAST_SEQ_STEP_MASK (0x1F)
- #define STATUS_POR_EVENT_MASK (0x80)
- #define STATUS_ACCESS_CONFLICT_MASK (0x40)

Functions

- int8_t zmod45xx_read_sensor_info (zmod45xx_dev_t *dev)

  Read sensor parameter.
- int8_t zmod45xx_init_sensor (zmod45xx_dev_t *dev)

  Initialize the sensor after power on.
- int8_t zmod45xx_init_measurement (zmod45xx_dev_t *dev)

  Initialize the sensor for zmod4510 measurement.
- int8_t zmod45xx_start_measurement (zmod45xx_dev_t *dev)

  Start the measurement.
- int8_t zmod45xx_read_status (zmod45xx_dev_t *dev, uint8_t *status)

  Read the status of the device.
- int8_t zmod45xx_read_adc_results (zmod45xx_dev_t *dev, uint8_t *adc_result)

  Read adc values from the sensor.
- int8_t zmod45xx_calc_rmox (zmod45xx_dev_t *dev, uint8_t *adc_result, float *rmox)

  Calculate mox resistance.

7.2.1 Detailed Description

ZMOD45xx functions.

Version

2.0.0

Date

2019-10-02

Author

IDT

7.2.2 Macro Definition Documentation

7.2.2.1 #define STATUS_ACCESS_CONFLICT_MASK (0x40)

AccessConflict

7.2.2.2 #define STATUS_ALARM_MASK (0x20)

Alarm

7.2.2.3 #define STATUS_LAST_SEQ_STEP_MASK (0x1F)

Last executed sequencer step

7.2.2.4 #define STATUS_POR_EVENT_MASK (0x80)

POR_event

7.2.2.5 #define STATUS_SEQUENCER_RUNNING_MASK (0x80)

Sequencer is running

7.2.2.6 #define STATUS_SLEEP_TIMER_ENABLED_MASK (0x40)

SleepTimer_enabled

7.2.3 Function Documentation

7.2.3.1 int8_t zmod45xx_calc_rmox (zmod45xx_dev_t * *dev*, uint8_t * *adc_result*, float * *rmox*)

Calculate mox resistance.

Parameters

in	*dev*	pointer to the device
in,out	*adc_result*	pointer to the adc results
in,out	*rmox*	pointer to the rmox values

Returns

error code

Return values

0	success
!= 0	error

7.2.3.2 int8_t zmod45xx_init_measurement (zmod45xx_dev_t * *dev*)

Initialize the sensor for zmod4510 measurement.

Parameters

in	*dev*	pointer to the device

Returns

error code

Return values

0	success
!= 0	error

7.2.3.3 int8_t zmod45xx_init_sensor (zmod45xx_dev_t * *dev*)

Initialize the sensor after power on.

Parameters

in	*dev*	pointer to the device

Returns

error code

Return values

0	success
!= 0	error

7.2.3.4 int8_t zmod45xx_read_adc_results (zmod45xx_dev_t * *dev*, uint8_t * *adc_result*)

Read adc values from the sensor.

Parameters

in	*dev*	pointer to the device
in, out	*adc_result*	pointer to the adc results

Returns

error code

Return values

0	success
!= 0	error

7.2.3.5 int8_t zmod45xx_read_sensor_info (zmod45xx_dev_t * *dev*)

Read sensor parameter.

Parameters

in	*dev*	pointer to the device

Returns

error code

Return values

0	success
!= 0	error

Note

This function must be called once before running other sensor functions.

7.2.3.6 int8_t zmod45xx_read_status (zmod45xx_dev_t * *dev*, uint8_t * *status*)

Read the status of the device.

Parameters

in	*dev*	pointer to the device
in, out	*status*	pointer to the status variable

Returns

error code

Return values

0	success
!= 0	error

7.2.3.7 int8_t zmod45xx_start_measurement (zmod45xx_dev_t * *dev*)

Start the measurement.

Parameters

in	*dev*	pointer to the device

Returns

error code

Return values

0	success
!= 0	error

7.3 zmod45xx_config.h File Reference

ZMOD45xx configuration.

```
#include <stdint.h>
#include "zmod45xx_types.h"
```

Variables

- const uint8_t **data_set_4510** [ ]
- const uint8_t **data_set_4510i** [ ]
- const zmod45xx_conf zmod4510
  ZMOD4510 configuration.
- const zmod45xx_conf zmod45xxi
  ZMOD45XX sensor initialization configuration.

7.3.1 Detailed Description

ZMOD45xx configuration.

Version

2.0.0

Date

2019-07-18

Author

IDT

7.3.2 Variable Documentation

7.3.2.1 const uint8_t data_set_4510[ ]

Initial value:

```
= { 0x20, 0x05, 0xA0, 0x18, 0xC0, 0x1C,
                                  0x03,
                                  0x00, 0x00, 0x00, 0x08, 0x00, 0x10, 0x00,
                                  0x01, 0x00, 0x09, 0x00, 0x11, 0x00, 0x02,
                                  0x00, 0x0A, 0x00, 0x12, 0x00, 0x03, 0x00,
                                  0x0B, 0x00, 0x13, 0x00, 0x04, 0x00, 0x0C,
                                  0x80, 0x14 }
```

7.3.2.2 const uint8_t data_set_4510i[ ]

Initial value:

```
= { 0x00, 0xA4, 0xC3, 0xE3,
                                    0x00, 0x00, 0x80, 0x40 }
```

7.3.2.3 const zmod45xx_conf zmod4510

Initial value:

```
= {
    .start = 0x80,
    .h = { .addr = 0x40, .len = 10 },
    .d = { .addr = 0x50, .len = 6, .data = &data_set_4510[0] },
    .m = { .addr = 0x60, .len = 1, .data = &data_set_4510[6] },
    .s = { .addr = 0x68, .len = 30, .data = &data_set_4510[7] },
    .r = { .addr = 0x97, .len = 30 }

}
```

ZMOD4510 configuration.

7.3.2.4 const zmod45xx_conf zmod45xxi

Initial value:

```
= {
    .start = 0x80,
    .h = { .addr = 0x40, .len = 2 },
    .d = { .addr = 0x50, .len = 2, .data = &data_set_4510i[0] },
    .m = { .addr = 0x60, .len = 2, .data = &data_set_4510i[2] },
    .s = { .addr = 0x68, .len = 4, .data = &data_set_4510i[4] },
    .r = { .addr = 0x97, .len = 4 }

}
```

ZMOD45XX sensor initialization configuration.

7.4 zmod45xx_types.h File Reference

ZMOD45xx types.

```
#include <stdint.h>
#include <stdio.h>
```

Data Structures

- struct zmod45xx_conf_str

  A single data set for the configuration.
- struct zmod45xx_conf

  Structure to hold the gas sensor module configuration.
- struct zmod45xx_dev_t

  Device structure ZMOD45xx.

Macros

- #define **ZMOD4510_PID** (0x6320)
- #define ZMOD45XX_OK (0)
- #define ERROR_INIT_OUT_OF_RANGE (1)
- #define ERROR_GAS_TIMEOUT (2)
- #define ERROR_I2C (3)
- #define ERROR_SENSOR_UNSUPPORTED (4)
- #define ERROR_CONFIG_MISSING (5)
- #define ERROR_SENSOR (6)
- #define ERROR_ACCESS_CONFLICT (7)
- #define ERROR_POR_EVENT (8)

Typedefs

- typedef int8_t(∗ zmod45xx_i2c_ptr_t) (uint8_t addr, uint8_t reg_addr, const uint8_t ∗data, uint8_t len)

  function pointer type for i2c access
- typedef void(∗ zmod45xx_delay_ptr_p) (uint32_t ms)

  function pointer to hardware dependent delay function

7.4.1 Detailed Description

ZMOD45xx types.

Version

2.0.0

Date

2019-07-18

Author

IDT

7.4.2 Macro Definition Documentation

7.4.2.1 #define ZMOD45XX_OK (0)

Return value if no fault has been found.

7.4.3 Typedef Documentation

7.4.3.1 typedef int8_t(* zmod45xx_i2c_ptr_t) (uint8_t addr, uint8_t reg_addr, const uint8_t *data, uint8_t len)

function pointer type for i2c access

Parameters

in	*addr*	7-bit I2C slave address of the ZMOD45xx
in	*reg_addr*	address of internal register to read/write
in,out	*data*	pointer to the read/write data value
in	*len*	number of bytes to read/write

Returns

error code

Return values

0	success
!= 0	error