Compass 2 click

From MikroElektonika Documentation
Revision as of 16:23, 24 June 2016 by Srdjan.misic (talk | contribs)

(diff) ← Older revision | Latest revision (diff) | Newer revision → (diff)

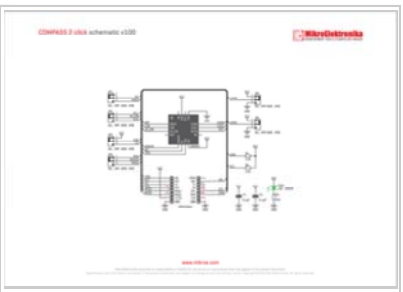
Compass 2 click carries an AK8963 3-axis electronic compass. The high sensitivity sensor is based on the Hall effect. The built-in ADC converter can be set up at either 14 or 16 bit resolution, for each of the 3 axes. The sensitivity is 0.6µT/LSB typ. at 14-bit, and 0.15µT/LSB at 16-bit. Compass 2 click communicates with the target MCU through either through the I2C or SPI interface, with an added INT pin. Onboard jumpers enable you to switch between two interfaces. The board is designed to use a 3.3 power supply only.

Compass 2 click

Compass 2 click

IC/Module	AK8963 (http://www.akm.com/akm/en/product/datasheet1/?partno=AK8963)
Interface	SPI, I2C, INT
Power supply	3.3V
Website	www.mikroe.com/click/compass-2 (http://www.mikroe.com/click/compass-2)

Features and usage notes



Schematic also available in PDF (http://cdn-docs.mikroe.com/images/1/1a/Compass_2_click_sch

The AK8964 has several operating modes which can be configured by setting a specific register (CNTL1) to certain values. The following is a list of available operating modes with partial descriptions (to give you an overview). The configuration details are available on page 13 of the official data sheet, while the complete descriptions

(1) Power-down mode

Power to almost all internal circuits is turned off. All registers are accessible in power-down mode. However, fuse ROM data cannot be read correctly. Data stored in read/write registers are remained. They can be reset by soft reset.

(2) Single measurement mode

When single measurement mode (MODE[3:0]="0001") is set, sensor is measured, and after sensor measurement and signal processing is finished, measurement data is stored to measurement data registers (HXL to HZH), then AK8963 transits to power-down mode automatically.

(3) Continuous measurement mode 1 and 2

When continuous measurement mode 1 (MODE[3:0]="0010") or 2 (MODE[3:0]="0110") is set, sensor is measured periodically at 8Hz or 100Hz respectively. When sensor measurement and signal processing is finished, measurement data is stored to measurement data registers (HXL ~ HZH) and all circuits except for the minimum circuit required for counting cycle lentgh are turned off (PD).

(4) External trigger measurement mode

When external trigger measurement mode (MODE[3:0]="0100") is set, AK8963 waits for trigger input. When a pulse is input from TRG pin, sensor measurement is started on the rising edge of TRG pin. When sensor measurement and signal processing is finished, measurement data is stored to measurement data registers (HXL to HZH) and all circuits except for the minimum circuit required for trigger input waiting are turned off (PD state).

(5) Fuse ROM access mode

Fuse ROM access mode is used to read Fuse ROM data. Sensitivity adjustments for each axis is stored in fuse ROM.

Compass 2 click has both SPI and I2C interfaces. The active interface is configured with onboard jumpers. If you use I2C, an additional jumper will allow you to set the I2C address.

Programming

This code snippet initiates Compass 2 with I2C communication, and reads out the heading value, along with a direction, (N, NE, E, etc) from the module to a UART terminal every 100 ms.

```
#include <stdint.h>
#include "compass2_hw.h"

sbit COMPASS2_CS at GPIOD_ODR.B13;

void system_setup( bus_mode_t mode, uint8_t addr );

float mRes;                 // scale resolutions per LSB for the sensors
uint8_t asax, asay, asaz;
float adjusted_ASAX, adjusted_ASAY, adjusted_ASAZ;
```

```
float heading, adjusted_MX, adjusted_MY, adjusted_MZ, magbias[3];
int16_t mx, my, mz;
char text[20] = { 0 };

void main()
{
  // Local Declarations
        uint8_t address = 0x0F;
        bus_mode_t my_mode = I2C;
        float heading = 0;
        char uart_text[5] = { 0 };

        system_setup( my_mode, address );

        while(1)
        {
            compass2_get_all_values( &mx, &my, &mz );
            heading = compass2_get_compass_heading( mx, my, mz );

            if( heading < 0 )
                heading += 360;

            UART1_Write_Text( "Heading: " );
            FloatToStr( heading, text );
            UART1_Write_Text( text );
            UART1_Write_Text( " Direction: " );

            if( heading >= 330 || heading <= 30 )
            {
                uart_text[0] = 'N';
                uart_text[1] = '\n';
            }
            else if( heading >= 300 && heading <= 330 )
            {
                uart_text[0] = 'N';
                uart_text[1] = 'W';
                uart_text[2] = '\n';
            }
            else if( heading >= 240 && heading <= 300)
            {
                uart_text[0] = 'W';
                uart_text[1] = '\n';
            }
            else if( heading >= 210 && heading <= 240 )
            {
                uart_text[0] = 'S';
                uart_text[1] = 'W';
                uart_text[2] = '\n';
            }
            else if( heading <= 210 && heading >= 150 )
            {
                uart_text[0] = 'S';
                uart_text[1] = '\n';
            }
            else if( heading <= 150 && heading >= 120 )
            {
                uart_text[0] = 'S';
                uart_text[1] = 'E';
                uart_text[2] = '\n';
            }
            else if( heading <= 120 && heading >= 60 )
            {
                uart_text[0] = 'E';
                uart_text[1] = '\n';
            }
            else if( heading <= 60 && heading >= 30 )
            {
                uart_text[0] = 'N';
                uart_text[1] = 'E';
                uart_text[2] = '\n';
            }

            UART1_Write_Text( uart_text );
            UART1_Write_Text( "\r\n" );

            Delay_ms(100);
        }
}

void system_setup( bus_mode_t mode, uint8_t addr )
{
        // GPIOs
        GPIO_Digital_Output( &GPIOB_BASE, _GPIO_PINMASK_13 );

        // UART
        UART1_Init( 9600 );
        UART1_Write_Text( "UART Initialized\r\n" );

        // I2C
        I2C1_Init_Advanced( 100000, &_GPIO_MODULE_I2C1_PB67 );
        UART1_Write_Text( "I2C Initialized\r\n" );

        // Compass 2
        UART1_Write_Text( "Getting Device ID..." );
        compass2_hw_init( addr, mode );
        UART1_Write_Text( "Compass Initialized\r\n" );

        // Compass 2 setup
        mRes = compass2_set_scale_factor( RES_16 );
        magbias[0] = +470;
        magbias[1] = +120;
        magbias[2] = +125;

        compass2_get_self_test( &mx, &my, &mz );
        UART1_Write_Text( "x y z Values: " );
        LongWordToStr( mx, text );
        UART1_Write_Text( text );
        UART1_Write_Text( "\t" );
        LongWordToStr( my, text );
        UART1_Write_Text( text );
        UART1_Write_Text( "\t" );
        LongWordToStr( mz, text );
        UART1_Write_Text( text );
        UART1_Write_Text( "\r\n" );

```

```
127         compass2_get_adjustment( &asax, &asay, &asaz );
128         adjusted_ASAX = ( (float)asax - 128 ) / 256 + 1;
129         adjusted_ASAY = ( (float)asay - 128 ) / 256 + 1;
130         adjusted_ASAZ = ( (float)asaz - 128 ) / 256 + 1;
131
132         compass2_set_mode( MODE_CONT_1 );
133         compass2_set_scale_factor( RES_16 );
134
135         UART1_Write_Text( "Compass2 Setup Completed..\r\n" );
136 }
```

Code examples that demonstrate the usage of Compass 2 click with MikroElektronika hardware, written for mikroC for ARM, AVR, dsPIC, FT90x, PIC and PIC32 are available on Libstock (http://libstock.mikroe.com/projects/view/1833/compass-2-click).

Resources

- Compass 2 click example on Libstock (http://libstock.mikroe.com/projects/view/1833/compass-2-click)

- Vendor’s data sheet (https://www.akm.com/akm/en/file/datasheet/AK8963C.pdf)

- mikroBUS™ standard specifications (http://www.mikroe.com/downloads/get/1737/mikrobus_specification.pdf)