```
// I2Cdev library collection - ADS1115 I2C device class header file
// Based on Texas Instruments ADS1113/4/5 datasheet, May 2009 (SBAS444B,
revised October 2009)
// Note that the ADS1115 uses 16-bit registers, not 8-bit registers.
// 8/2/2011 by Jeff Rowberg <jeff@rowberg.net>
// Updates should (hopefully) always be available at
https://github.com/jrowberg/i2cdevlib
//
// Changelog:
//     2011-08-02 - initial release
//     2011-10-29 - added getDifferentialx() methods, F. Farzanegan

/* ============================================
I2Cdev device library code is placed under the MIT license
Copyright (c) 2011 Jeff Rowberg

Permission is hereby granted, free of charge, to any person obtaining a
copy
of this software and associated documentation files (the "Software"), to
deal
in the Software without restriction, including without limitation the
rights
to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
copies of the Software, and to permit persons to whom the Software is
furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included
in
all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS
OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL
THE
AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING
FROM,
OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN
THE SOFTWARE.
===============================================
*/

#ifndef _ADS1115_H_
#define _ADS1115_H_

#include "I2Cdev.h"

#define ADS1115_ADDRESS_ADDR_GND    0x48 // address pin low (GND)
#define ADS1115_ADDRESS_ADDR_VDD    0x49 // address pin high (VCC)
#define ADS1115_ADDRESS_ADDR_SDA    0x4A // address pin tied to SDA pin
#define ADS1115_ADDRESS_ADDR_SCL    0x4B // address pin tied to SCL pin
#define ADS1115_DEFAULT_ADDRESS     ADS1115_ADDRESS_ADDR_GND
```

```
#define ADS1115_RA_CONVERSION       0x00
#define ADS1115_RA_CONFIG           0x01
#define ADS1115_RA_LO_THRESH        0x02
#define ADS1115_RA_HI_THRESH        0x03

#define ADS1115_CFG_OS_BIT          15
#define ADS1115_CFG_MUX_BIT         14
#define ADS1115_CFG_MUX_LENGTH      3
#define ADS1115_CFG_PGA_BIT         11
#define ADS1115_CFG_PGA_LENGTH      3
#define ADS1115_CFG_MODE_BIT        8
#define ADS1115_CFG_DR_BIT          7
#define ADS1115_CFG_DR_LENGTH       3
#define ADS1115_CFG_COMP_MODE_BIT   4
#define ADS1115_CFG_COMP_POL_BIT    3
#define ADS1115_CFG_COMP_LAT_BIT    2
#define ADS1115_CFG_COMP_QUE_BIT    1
#define ADS1115_CFG_COMP_QUE_LENGTH 2

#define ADS1115_OS_INACTIVE         0x00
#define ADS1115_OS_ACTIVE           0x01

#define ADS1115_MUX_P0_N1           0x00 // default
#define ADS1115_MUX_P0_N3           0x01
#define ADS1115_MUX_P1_N3           0x02
#define ADS1115_MUX_P2_N3           0x03
#define ADS1115_MUX_P0_NG           0x04
#define ADS1115_MUX_P1_NG           0x05
#define ADS1115_MUX_P2_NG           0x06
#define ADS1115_MUX_P3_NG           0x07

#define ADS1115_PGA_6P144           0x00
#define ADS1115_PGA_4P096           0x01
#define ADS1115_PGA_2P048           0x02 // default
#define ADS1115_PGA_1P024           0x03
#define ADS1115_PGA_0P512           0x04
#define ADS1115_PGA_0P256           0x05
#define ADS1115_PGA_0P256B          0x06
#define ADS1115_PGA_0P256C          0x07

#define ADS1115_MV_6P144            0.187500
#define ADS1115_MV_4P096            0.125000
#define ADS1115_MV_2P048            0.062500 // default
#define ADS1115_MV_1P024            0.031250
#define ADS1115_MV_0P512            0.015625
#define ADS1115_MV_0P256            0.007813
#define ADS1115_MV_0P256B           0.007813
#define ADS1115_MV_0P256C           0.007813

#define ADS1115_MODE_CONTINUOUS     0x00
#define ADS1115_MODE_SINGLESHOT     0x01 // default

#define ADS1115_RATE_8              0x00
#define ADS1115_RATE_16             0x01
```

```
#define ADS1115_RATE_32             0x02
#define ADS1115_RATE_64             0x03
#define ADS1115_RATE_128            0x04 // default
#define ADS1115_RATE_250            0x05
#define ADS1115_RATE_475            0x06
#define ADS1115_RATE_860            0x07

#define ADS1115_COMP_MODE_HYSTERESIS    0x00 // default
#define ADS1115_COMP_MODE_WINDOW        0x01

#define ADS1115_COMP_POL_ACTIVE_LOW     0x00 // default
#define ADS1115_COMP_POL_ACTIVE_HIGH    0x01

#define ADS1115_COMP_LAT_NON_LATCHING   0x00 // default
#define ADS1115_COMP_LAT_LATCHING       0x01

#define ADS1115_COMP_QUE_ASSERT1    0x00
#define ADS1115_COMP_QUE_ASSERT2    0x01
#define ADS1115_COMP_QUE_ASSERT4    0x02
#define ADS1115_COMP_QUE_DISABLE    0x03 // default


class ADS1115 {
    public:
        ADS1115();
        ADS1115(uint8_t address);

        void initialize();
        bool testConnection();

        // CONVERSION register
        int16_t getDifferential();
        int16_t getDifferential0();
        int16_t getDifferential1();
        int16_t getDifferential2();
        int16_t getDifferential3();
        int16_t getDiff0();
        int16_t getDiff1();
        int16_t getDiff2();
        int16_t getDiff3();
        float getMilliVolts();
        float getMvPerCount();

        // CONFIG register
        uint8_t getOpStatus();
        void setOpStatus(uint8_t mux);
        uint8_t getMultiplexer();
        void setMultiplexer(uint8_t mux);
        uint8_t getGain();
        void setGain(uint8_t gain);
        uint8_t getMode();
        void setMode(uint8_t mode);
        uint8_t getRate();
        void setRate(uint8_t rate);
```

```
        uint8_t getComparatorMode();
        void setComparatorMode(uint8_t mode);
        uint8_t getComparatorPolarity();
        void setComparatorPolarity(uint8_t polarity);
        bool getComparatorLatchEnabled();
        void setComparatorLatchEnabled(bool enabled);
        uint8_t getComparatorQueueMode();
        void setComparatorQueueMode(uint8_t mode);

        // *_THRESH registers
        int16_t getLowThreshold();
        void setLowThreshold(int16_t threshold);
        int16_t getHighThreshold();
        void setHighThreshold(int16_t threshold);

    private:
        uint8_t devAddr;
        uint16_t buffer[2];
        uint8_t devMode;
        uint8_t muxMode;
        uint8_t pgaMode;
};

#endif /* _ADS1115_H_ */
```