**#ifndef LiquidCrystal\_I2C\_h**

**#define LiquidCrystal\_I2C\_h**

**#include <inttypes.h>**

**#include <Print.h>**

**#include "I2CIO.h"**

**#include "LCD.h"**

**class LiquidCrystal\_I2C : public LCD**

**{**

**public:**

**/\*!**

**@method**

**@abstract Class constructor.**

**@discussion Initializes class variables and defines the I2C address of the**

**LCD. The constructor does not initialize the LCD.**

**\*/**

**LiquidCrystal\_I2C (uint8\_t lcd\_Addr);**

**// Constructor with backlight control**

**LiquidCrystal\_I2C (uint8\_t lcd\_Addr, uint8\_t backlighPin, t\_backlighPol pol);**

**LiquidCrystal\_I2C( uint8\_t lcd\_Addr, uint8\_t En, uint8\_t Rw, uint8\_t Rs);**

**// Constructor with backlight control**

**LiquidCrystal\_I2C(uint8\_t lcd\_Addr, uint8\_t En, uint8\_t Rw, uint8\_t Rs,**

**uint8\_t backlighPin, t\_backlighPol pol);**

**/\*!**

**@method**

**@abstract Class constructor.**

**@discussion Initializes class variables and defines the I2C address of the**

**LCD. The constructor does not initialize the LCD.**

**@param lcd\_Addr[in] I2C address of the IO expansion module. For I2CLCDextraIO,**

**the address can be configured using the on board jumpers.**

**@param En[in] LCD En (Enable) pin connected to the IO extender module**

**@param Rw[in] LCD Rw (Read/write) pin connected to the IO extender module**

**@param Rs[in] LCD Rs (Reset) pin connected to the IO extender module**

**@param d4[in] LCD data 0 pin map on IO extender module**

**@param d5[in] LCD data 1 pin map on IO extender module**

**@param d6[in] LCD data 2 pin map on IO extender module**

**@param d7[in] LCD data 3 pin map on IO extender module**

**\*/**

**LiquidCrystal\_I2C(uint8\_t lcd\_Addr, uint8\_t En, uint8\_t Rw, uint8\_t Rs,**

**uint8\_t d4, uint8\_t d5, uint8\_t d6, uint8\_t d7 );**

**// Constructor with backlight control**

**LiquidCrystal\_I2C(uint8\_t lcd\_Addr, uint8\_t En, uint8\_t Rw, uint8\_t Rs,**

**uint8\_t d4, uint8\_t d5, uint8\_t d6, uint8\_t d7,**

**uint8\_t backlighPin, t\_backlighPol pol);**

**/\*!**

**@function**

**@abstract LCD initialization and associated HW.**

**@discussion Initializes the LCD to a given size (col, row). This methods**

**initializes the LCD, therefore, it MUST be called prior to using any other**

**method from this class or parent class.**

**The begin method can be overloaded if necessary to initialize any HW that**

**is implemented by a library and can't be done during construction, here**

**we use the Wire class.**

**@param cols[in] the number of columns that the display has**

**@param rows[in] the number of rows that the display has**

**@param charsize[in] size of the characters of the LCD: LCD\_5x8DOTS or**

**LCD\_5x10DOTS.**

**\*/**

**virtual void begin(uint8\_t cols, uint8\_t rows, uint8\_t charsize = LCD\_5x8DOTS);**

**/\*!**

**@function**

**@abstract Send a particular value to the LCD.**

**@discussion Sends a particular value to the LCD for writing to the LCD or**

**as an LCD command.**

**Users should never call this method.**

**@param value[in] Value to send to the LCD.**

**@param mode[in] DATA - write to the LCD CGRAM, COMMAND - write a**

**command to the LCD.**

**\*/**

**virtual void send(uint8\_t value, uint8\_t mode);**

**/\*!**

**@function**

**@abstract Sets the pin to control the backlight.**

**@discussion Sets the pin in the device to control the backlight. This device**

**doesn't support dimming backlight capability.**

**@param 0: backlight off, 1..255: backlight on.**

**\*/**

**void setBacklightPin ( uint8\_t value, t\_backlighPol pol );**

**/\*!**

**@function**

**@abstract Switch-on/off the LCD backlight.**

**@discussion Switch-on/off the LCD backlight.**

**The setBacklightPin has to be called before setting the backlight for**

**this method to work. @see setBacklightPin.**

**@param value: backlight mode (HIGH|LOW)**

**\*/**

**void setBacklight ( uint8\_t value );**

**private:**

**/\*!**

**@method**

**@abstract Initializes the LCD class**

**@discussion Initializes the LCD class and IO expansion module.**

**\*/**

**int init();**

**/\*!**

**@function**

**@abstract Initialises class private variables**

**@discussion This is the class single point for initialising private variables.**

**@param lcd\_Addr[in] I2C address of the IO expansion module. For I2CLCDextraIO,**

**the address can be configured using the on board jumpers.**

**@param En[in] LCD En (Enable) pin connected to the IO extender module**

**@param Rw[in] LCD Rw (Read/write) pin connected to the IO extender module**

**@param Rs[in] LCD Rs (Reset) pin connected to the IO extender module**

**@param d4[in] LCD data 0 pin map on IO extender module**

**@param d5[in] LCD data 1 pin map on IO extender module**

**@param d6[in] LCD data 2 pin map on IO extender module**

**@param d7[in] LCD data 3 pin map on IO extender module**

**\*/**

**void config (uint8\_t lcd\_Addr, uint8\_t En, uint8\_t Rw, uint8\_t Rs,**

**uint8\_t d4, uint8\_t d5, uint8\_t d6, uint8\_t d7 );**

**/\*!**

**@method**

**@abstract Writes an 4 bit value to the LCD.**

**@discussion Writes 4 bits (the least significant) to the LCD control data lines.**

**@param value[in] Value to write to the LCD**

**@param more[in] Value to distinguish between command and data.**

**COMMAND == command, DATA == data.**

**\*/**

**void write4bits(uint8\_t value, uint8\_t mode);**

**/\*!**

**@method**

**@abstract Pulse the LCD enable line (En).**

**@discussion Sends a pulse of 1 uS to the Enable pin to execute an command**

**or write operation.**

**\*/**

**void pulseEnable(uint8\_t);**

**uint8\_t \_Addr; // I2C Address of the IO expander**

**uint8\_t \_backlightPinMask; // Backlight IO pin mask**

**uint8\_t \_backlightStsMask; // Backlight status mask**

**I2CIO \_i2cio; // I2CIO PCF8574\* expansion module driver I2CLCDextraIO**

**uint8\_t \_En; // LCD expander word for enable pin**

**uint8\_t \_Rw; // LCD expander word for R/W pin**

**uint8\_t \_Rs; // LCD expander word for Register Select pin**

**uint8\_t \_data\_pins[4]; // LCD data lines**

**};**

**#endif**