#ifndef FDB\_LIQUID\_CRYSTAL\_I2C\_H

#define FDB\_LIQUID\_CRYSTAL\_I2C\_H

#include <inttypes.h>

#include <Print.h>

// commands

#define LCD\_CLEARDISPLAY 0x01

#define LCD\_RETURNHOME 0x02

#define LCD\_ENTRYMODESET 0x04

#define LCD\_DISPLAYCONTROL 0x08

#define LCD\_CURSORSHIFT 0x10

#define LCD\_FUNCTIONSET 0x20

#define LCD\_SETCGRAMADDR 0x40

#define LCD\_SETDDRAMADDR 0x80

// flags for display entry mode

#define LCD\_ENTRYRIGHT 0x00

#define LCD\_ENTRYLEFT 0x02

#define LCD\_ENTRYSHIFTINCREMENT 0x01

#define LCD\_ENTRYSHIFTDECREMENT 0x00

// flags for display on/off control

#define LCD\_DISPLAYON 0x04

#define LCD\_DISPLAYOFF 0x00

#define LCD\_CURSORON 0x02

#define LCD\_CURSOROFF 0x00

#define LCD\_BLINKON 0x01

#define LCD\_BLINKOFF 0x00

// flags for display/cursor shift

#define LCD\_DISPLAYMOVE 0x08

#define LCD\_CURSORMOVE 0x00

#define LCD\_MOVERIGHT 0x04

#define LCD\_MOVELEFT 0x00

// flags for function set

#define LCD\_8BITMODE 0x10

#define LCD\_4BITMODE 0x00

#define LCD\_2LINE 0x08

#define LCD\_1LINE 0x00

#define LCD\_5x10DOTS 0x04

#define LCD\_5x8DOTS 0x00

// flags for backlight control

#define LCD\_BACKLIGHT 0x08

#define LCD\_NOBACKLIGHT 0x00

#define En B00000100 // Enable bit

#define Rw B00000010 // Read/Write bit

#define Rs B00000001 // Register select bit

/\*\*

\* This is the driver for the Liquid Crystal LCD displays that use the I2C bus.

\*

\* After creating an instance of this class, first call begin() before anything else.

\* The backlight is on by default, since that is the most likely operating mode in

\* most cases.

\*/

class LiquidCrystal\_I2C : public Print {

public:

/\*\*

\* Constructor

\*

\* @param lcd\_addr I2C slave address of the LCD display. Most likely printed on the

\* LCD circuit board, or look in the supplied LCD documentation.

\* @param lcd\_cols Number of columns your LCD display has.

\* @param lcd\_rows Number of rows your LCD display has.

\* @param charsize The size in dots that the display has, use LCD\_5x10DOTS or LCD\_5x8DOTS.

\*/

LiquidCrystal\_I2C(uint8\_t lcd\_addr, uint8\_t lcd\_cols, uint8\_t lcd\_rows, uint8\_t charsize = LCD\_5x8DOTS);

/\*\*

\* Set the LCD display in the correct begin state, must be called before anything else is done.

\*/

void begin();

/\*\*

\* Remove all the characters currently shown. Next print/write operation will start

\* from the first position on LCD display.

\*/

void clear();

/\*\*

\* Next print/write operation will will start from the first position on the LCD display.

\*/

void home();

/\*\*

\* Do not show any characters on the LCD display. Backlight state will remain unchanged.

\* Also all characters written on the display will return, when the display in enabled again.

\*/

void noDisplay();

/\*\*

\* Show the characters on the LCD display, this is the normal behaviour. This method should

\* only be used after noDisplay() has been used.

\*/

void display();

/\*\*

\* Do not blink the cursor indicator.

\*/

void noBlink();

/\*\*

\* Start blinking the cursor indicator.

\*/

void blink();

/\*\*

\* Do not show a cursor indicator.

\*/

void noCursor();

/\*\*

\* Show a cursor indicator, cursor can blink on not blink. Use the

\* methods blink() and noBlink() for changing cursor blink.

\*/

void cursor();

void scrollDisplayLeft();

void scrollDisplayRight();

void printLeft();

void printRight();

void leftToRight();

void rightToLeft();

void shiftIncrement();

void shiftDecrement();

void noBacklight();

void backlight();

void autoscroll();

void noAutoscroll();

void createChar(uint8\_t, uint8\_t[]);

void setCursor(uint8\_t, uint8\_t);

virtual size\_t write(uint8\_t);

void command(uint8\_t);

inline void blink\_on() { blink(); }

inline void blink\_off() { noBlink(); }

inline void cursor\_on() { cursor(); }

inline void cursor\_off() { noCursor(); }

// Compatibility API function aliases

void setBacklight(uint8\_t new\_val); // alias for backlight() and nobacklight()

void load\_custom\_character(uint8\_t char\_num, uint8\_t \*rows); // alias for createChar()

void printstr(const char[]);

private:

void send(uint8\_t, uint8\_t);

void write4bits(uint8\_t);

void expanderWrite(uint8\_t);

void pulseEnable(uint8\_t);

uint8\_t \_addr;

uint8\_t \_displayfunction;

uint8\_t \_displaycontrol;

uint8\_t \_displaymode;

uint8\_t \_cols;

uint8\_t \_rows;

uint8\_t \_charsize;

uint8\_t \_backlightval;

};

#endif // FDB\_LIQUID\_CRYSTAL\_I2C\_H