/\*!

\* @file Adafruit\_GrayOLED.h

\*

\* This is part of for Adafruit's GFX library, supplying generic support

\* for grayscale OLED displays: http://www.adafruit.com/category/63\_98

\*

\* These displays use I2C or SPI to communicate. I2C requires 2 pins

\* (SCL+SDA) and optionally a RESET pin. SPI requires 4 pins (MOSI, SCK,

\* select, data/command) and optionally a reset pin. Hardware SPI or

\* 'bitbang' software SPI are both supported.

\*

\* Adafruit invests time and resources providing this open source code,

\* please support Adafruit and open-source hardware by purchasing

\* products from Adafruit!

\*

\* Written by Limor Fried/Ladyada for Adafruit Industries, with

\* contributions from the open source community.

\*

\* BSD license, all text above, and the splash screen header file,

\* must be included in any redistribution.

\*

\*/

#ifndef \_Adafruit\_GRAYOLED\_H\_

#define \_Adafruit\_GRAYOLED\_H\_

#if !defined(\_\_AVR\_ATtiny85\_\_) // Not for ATtiny, at all

#include <Adafruit\_GFX.h>

#include <Adafruit\_I2CDevice.h>

#include <Adafruit\_SPIDevice.h>

#include <SPI.h>

#include <Wire.h>

#define GRAYOLED\_SETCONTRAST 0x81 ///< Generic contrast for almost all OLEDs

#define GRAYOLED\_NORMALDISPLAY 0xA6 ///< Generic non-invert for almost all OLEDs

#define GRAYOLED\_INVERTDISPLAY 0xA7 ///< Generic invert for almost all OLEDs

#define MONOOLED\_BLACK 0 ///< Default black 'color' for monochrome OLEDS

#define MONOOLED\_WHITE 1 ///< Default white 'color' for monochrome OLEDS

#define MONOOLED\_INVERSE 2 ///< Default inversion command for monochrome OLEDS

/\*!

@brief Class that stores state and functions for interacting with

generic grayscale OLED displays.

\*/

class Adafruit\_GrayOLED : public Adafruit\_GFX {

public:

Adafruit\_GrayOLED(uint8\_t bpp, uint16\_t w, uint16\_t h, TwoWire \*twi = &Wire,

int8\_t rst\_pin = -1, uint32\_t preclk = 400000,

uint32\_t postclk = 100000);

Adafruit\_GrayOLED(uint8\_t bpp, uint16\_t w, uint16\_t h, int8\_t mosi\_pin,

int8\_t sclk\_pin, int8\_t dc\_pin, int8\_t rst\_pin,

int8\_t cs\_pin);

Adafruit\_GrayOLED(uint8\_t bpp, uint16\_t w, uint16\_t h, SPIClass \*spi,

int8\_t dc\_pin, int8\_t rst\_pin, int8\_t cs\_pin,

uint32\_t bitrate = 8000000UL);

~Adafruit\_GrayOLED(void);

/\*\*

@brief The function that sub-classes define that writes out the buffer to

the display over I2C or SPI

\*\*/

virtual void display(void) = 0;

void clearDisplay(void);

void invertDisplay(bool i);

void setContrast(uint8\_t contrastlevel);

void drawPixel(int16\_t x, int16\_t y, uint16\_t color);

bool getPixel(int16\_t x, int16\_t y);

uint8\_t \*getBuffer(void);

void oled\_command(uint8\_t c);

bool oled\_commandList(const uint8\_t \*c, uint8\_t n);

protected:

bool \_init(uint8\_t i2caddr = 0x3C, bool reset = true);

Adafruit\_SPIDevice \*spi\_dev = NULL; ///< The SPI interface BusIO device

Adafruit\_I2CDevice \*i2c\_dev = NULL; ///< The I2C interface BusIO device

int32\_t i2c\_preclk = 400000, ///< Configurable 'high speed' I2C rate

i2c\_postclk = 100000; ///< Configurable 'low speed' I2C rate

uint8\_t \*buffer = NULL; ///< Internal 1:1 framebuffer of display mem

int16\_t window\_x1, ///< Dirty tracking window minimum x

window\_y1, ///< Dirty tracking window minimum y

window\_x2, ///< Dirty tracking window maximum x

window\_y2; ///< Dirty tracking window maximum y

int dcPin, ///< The Arduino pin connected to D/C (for SPI)

csPin, ///< The Arduino pin connected to CS (for SPI)

rstPin; ///< The Arduino pin connected to reset (-1 if unused)

uint8\_t \_bpp = 1; ///< Bits per pixel color for this display

private:

TwoWire \*\_theWire = NULL; ///< The underlying hardware I2C

};

#endif // end \_\_AVR\_ATtiny85\_\_

#endif // \_Adafruit\_GrayOLED\_H\_