/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 This is our GFX example for the Adafruit ILI9341 Breakout and Shield

 ----> http://www.adafruit.com/products/1651

 Check out the links above for our tutorials and wiring diagrams

 These displays use SPI to communicate, 4 or 5 pins are required to

 interface (RST is optional)

 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.

 MIT license, all text above must be included in any redistribution

 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Pat McMahon modified 19/9/2018

//A010

#include "SPI.h"

#include "Adafruit\_GFX.h"

#include "Adafruit\_ILI9341.h"

// For the Adafruit shield, these are the default.

#define TFT\_DC A4

#define TFT\_CS A5

// Use hardware SPI (on Uno, #13, #12, #11) and the above for CS/DC

Adafruit\_ILI9341 tft = Adafruit\_ILI9341(TFT\_CS, TFT\_DC);

// If using the breakout, change pins as desired

//Adafruit\_ILI9341 tft = Adafruit\_ILI9341(TFT\_CS, TFT\_DC, TFT\_MOSI, TFT\_CLK, TFT\_RST, TFT\_MISO);

void setup() {

 Serial.begin(9600);

 Serial.println("ILI9341 Test!");

 tft.begin();

 // read diagnostics (optional but can help debug problems)

 uint8\_t x = tft.readcommand8(ILI9341\_RDMODE);

 Serial.print("Display Power Mode: 0x"); Serial.println(x, HEX);

 x = tft.readcommand8(ILI9341\_RDMADCTL);

 Serial.print("MADCTL Mode: 0x"); Serial.println(x, HEX);

 x = tft.readcommand8(ILI9341\_RDPIXFMT);

 Serial.print("Pixel Format: 0x"); Serial.println(x, HEX);

 x = tft.readcommand8(ILI9341\_RDIMGFMT);

 Serial.print("Image Format: 0x"); Serial.println(x, HEX);

 x = tft.readcommand8(ILI9341\_RDSELFDIAG);

 Serial.print("Self Diagnostic: 0x"); Serial.println(x, HEX);

 Serial.println(F("Benchmark Time (microseconds)"));

 delay(10);

 Serial.print(F("Screen fill "));

 Serial.println(testFillScreen());

 delay(500);

 Serial.print(F("Text "));

 Serial.println(testText());

 delay(3000);

 Serial.print(F("Lines "));

 Serial.println(testLines(ILI9341\_CYAN));

 delay(500);

 Serial.print(F("Horiz/Vert Lines "));

 Serial.println(testFastLines(ILI9341\_RED, ILI9341\_BLUE));

 delay(500);

 Serial.print(F("Rectangles (outline) "));

 Serial.println(testRects(ILI9341\_GREEN));

 delay(500);

 Serial.print(F("Rectangles (filled) "));

 Serial.println(testFilledRects(ILI9341\_YELLOW, ILI9341\_MAGENTA));

 delay(500);

 Serial.print(F("Circles (filled) "));

 Serial.println(testFilledCircles(10, ILI9341\_MAGENTA));

 Serial.print(F("Circles (outline) "));

 Serial.println(testCircles(10, ILI9341\_WHITE));

 delay(500);

 Serial.print(F("Triangles (outline) "));

 Serial.println(testTriangles());

 delay(500);

 Serial.print(F("Triangles (filled) "));

 Serial.println(testFilledTriangles());

 delay(500);

 Serial.print(F("Rounded rects (outline) "));

 Serial.println(testRoundRects());

 delay(500);

 Serial.print(F("Rounded rects (filled) "));

 Serial.println(testFilledRoundRects());

 delay(500);

 Serial.println(F("Done!"));

}

void loop(void) {

 for(uint8\_t rotation=0; rotation<4; rotation++) {

 tft.setRotation(rotation);

 testText();

 delay(1000);

 }

}

unsigned long testFillScreen() {

 unsigned long start = micros();

 tft.fillScreen(ILI9341\_BLACK);

 yield();

 tft.fillScreen(ILI9341\_RED);

 yield();

 tft.fillScreen(ILI9341\_GREEN);

 yield();

 tft.fillScreen(ILI9341\_BLUE);

 yield();

 tft.fillScreen(ILI9341\_BLACK);

 yield();

 return micros() - start;

}

unsigned long testText() {

 tft.fillScreen(ILI9341\_BLACK);

 unsigned long start = micros();

 tft.setCursor(0, 0);

 tft.setTextColor(ILI9341\_WHITE); tft.setTextSize(1);

 tft.println("Hello World!");

 tft.setTextColor(ILI9341\_YELLOW); tft.setTextSize(2);

 tft.println(1234.56);

 tft.setTextColor(ILI9341\_RED); tft.setTextSize(3);

 tft.println(0xDEADBEEF, HEX);

 tft.println();

 tft.setTextColor(ILI9341\_GREEN);

 tft.setTextSize(5);

 tft.println("Groop");

 tft.setTextSize(2);

 tft.println("I implore thee,");

 tft.setTextSize(1);

 tft.println("my foonting turlingdromes.");

 tft.println("And hooptiously drangle me");

 tft.println("with crinkly bindlewurdles,");

 tft.println("Or I will rend thee");

 tft.println("in the gobberwarts");

 tft.println("with my blurglecruncheon,");

 tft.println("see if I don't!");

 return micros() - start;

}

unsigned long testLines(uint16\_t color) {

 unsigned long start, t;

 int x1, y1, x2, y2,

 w = tft.width(),

 h = tft.height();

 tft.fillScreen(ILI9341\_BLACK);

 yield();

 x1 = y1 = 0;

 y2 = h - 1;

 start = micros();

 for(x2=0; x2<w; x2+=6) tft.drawLine(x1, y1, x2, y2, color);

 x2 = w - 1;

 for(y2=0; y2<h; y2+=6) tft.drawLine(x1, y1, x2, y2, color);

 t = micros() - start; // fillScreen doesn't count against timing

 yield();

 tft.fillScreen(ILI9341\_BLACK);

 yield();

 x1 = w - 1;

 y1 = 0;

 y2 = h - 1;

 start = micros();

 for(x2=0; x2<w; x2+=6) tft.drawLine(x1, y1, x2, y2, color);

 x2 = 0;

 for(y2=0; y2<h; y2+=6) tft.drawLine(x1, y1, x2, y2, color);

 t += micros() - start;

 yield();

 tft.fillScreen(ILI9341\_BLACK);

 yield();

 x1 = 0;

 y1 = h - 1;

 y2 = 0;

 start = micros();

 for(x2=0; x2<w; x2+=6) tft.drawLine(x1, y1, x2, y2, color);

 x2 = w - 1;

 for(y2=0; y2<h; y2+=6) tft.drawLine(x1, y1, x2, y2, color);

 t += micros() - start;

 yield();

 tft.fillScreen(ILI9341\_BLACK);

 yield();

 x1 = w - 1;

 y1 = h - 1;

 y2 = 0;

 start = micros();

 for(x2=0; x2<w; x2+=6) tft.drawLine(x1, y1, x2, y2, color);

 x2 = 0;

 for(y2=0; y2<h; y2+=6) tft.drawLine(x1, y1, x2, y2, color);

 yield();

 return micros() - start;

}

unsigned long testFastLines(uint16\_t color1, uint16\_t color2) {

 unsigned long start;

 int x, y, w = tft.width(), h = tft.height();

 tft.fillScreen(ILI9341\_BLACK);

 start = micros();

 for(y=0; y<h; y+=5) tft.drawFastHLine(0, y, w, color1);

 for(x=0; x<w; x+=5) tft.drawFastVLine(x, 0, h, color2);

 return micros() - start;

}

unsigned long testRects(uint16\_t color) {

 unsigned long start;

 int n, i, i2,

 cx = tft.width() / 2,

 cy = tft.height() / 2;

 tft.fillScreen(ILI9341\_BLACK);

 n = min(tft.width(), tft.height());

 start = micros();

 for(i=2; i<n; i+=6) {

 i2 = i / 2;

 tft.drawRect(cx-i2, cy-i2, i, i, color);

 }

 return micros() - start;

}

unsigned long testFilledRects(uint16\_t color1, uint16\_t color2) {

 unsigned long start, t = 0;

 int n, i, i2,

 cx = tft.width() / 2 - 1,

 cy = tft.height() / 2 - 1;

 tft.fillScreen(ILI9341\_BLACK);

 n = min(tft.width(), tft.height());

 for(i=n; i>0; i-=6) {

 i2 = i / 2;

 start = micros();

 tft.fillRect(cx-i2, cy-i2, i, i, color1);

 t += micros() - start;

 // Outlines are not included in timing results

 tft.drawRect(cx-i2, cy-i2, i, i, color2);

 yield();

 }

 return t;

}

unsigned long testFilledCircles(uint8\_t radius, uint16\_t color) {

 unsigned long start;

 int x, y, w = tft.width(), h = tft.height(), r2 = radius \* 2;

 tft.fillScreen(ILI9341\_BLACK);

 start = micros();

 for(x=radius; x<w; x+=r2) {

 for(y=radius; y<h; y+=r2) {

 tft.fillCircle(x, y, radius, color);

 }

 }

 return micros() - start;

}

unsigned long testCircles(uint8\_t radius, uint16\_t color) {

 unsigned long start;

 int x, y, r2 = radius \* 2,

 w = tft.width() + radius,

 h = tft.height() + radius;

 // Screen is not cleared for this one -- this is

 // intentional and does not affect the reported time.

 start = micros();

 for(x=0; x<w; x+=r2) {

 for(y=0; y<h; y+=r2) {

 tft.drawCircle(x, y, radius, color);

 }

 }

 return micros() - start;

}

unsigned long testTriangles() {

 unsigned long start;

 int n, i, cx = tft.width() / 2 - 1,

 cy = tft.height() / 2 - 1;

 tft.fillScreen(ILI9341\_BLACK);

 n = min(cx, cy);

 start = micros();

 for(i=0; i<n; i+=5) {

 tft.drawTriangle(

 cx , cy - i, // peak

 cx - i, cy + i, // bottom left

 cx + i, cy + i, // bottom right

 tft.color565(i, i, i));

 }

 return micros() - start;

}

unsigned long testFilledTriangles() {

 unsigned long start, t = 0;

 int i, cx = tft.width() / 2 - 1,

 cy = tft.height() / 2 - 1;

 tft.fillScreen(ILI9341\_BLACK);

 start = micros();

 for(i=min(cx,cy); i>10; i-=5) {

 start = micros();

 tft.fillTriangle(cx, cy - i, cx - i, cy + i, cx + i, cy + i,

 tft.color565(0, i\*10, i\*10));

 t += micros() - start;

 tft.drawTriangle(cx, cy - i, cx - i, cy + i, cx + i, cy + i,

 tft.color565(i\*10, i\*10, 0));

 yield();

 }

 return t;

}

unsigned long testRoundRects() {

 unsigned long start;

 int w, i, i2,

 cx = tft.width() / 2 - 1,

 cy = tft.height() / 2 - 1;

 tft.fillScreen(ILI9341\_BLACK);

 w = min(tft.width(), tft.height());

 start = micros();

 for(i=0; i<w; i+=6) {

 i2 = i / 2;

 tft.drawRoundRect(cx-i2, cy-i2, i, i, i/8, tft.color565(i, 0, 0));

 }

 return micros() - start;

}

unsigned long testFilledRoundRects() {

 unsigned long start;

 int i, i2,

 cx = tft.width() / 2 - 1,

 cy = tft.height() / 2 - 1;

 tft.fillScreen(ILI9341\_BLACK);

 start = micros();

 for(i=min(tft.width(), tft.height()); i>20; i-=6) {

 i2 = i / 2;

 tft.fillRoundRect(cx-i2, cy-i2, i, i, i/8, tft.color565(0, i, 0));

 yield();

 }

 return micros() - start;

}