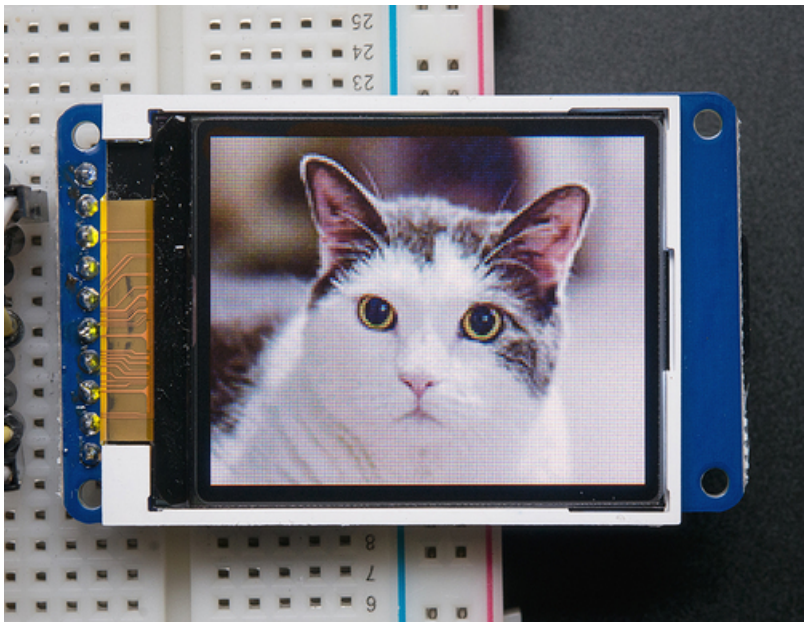
adafruit learning system

1.8" TFT Display Breakout and Shield

Created by lady ada



Last updated on 2017-11-17 05:51:22 PM UTC

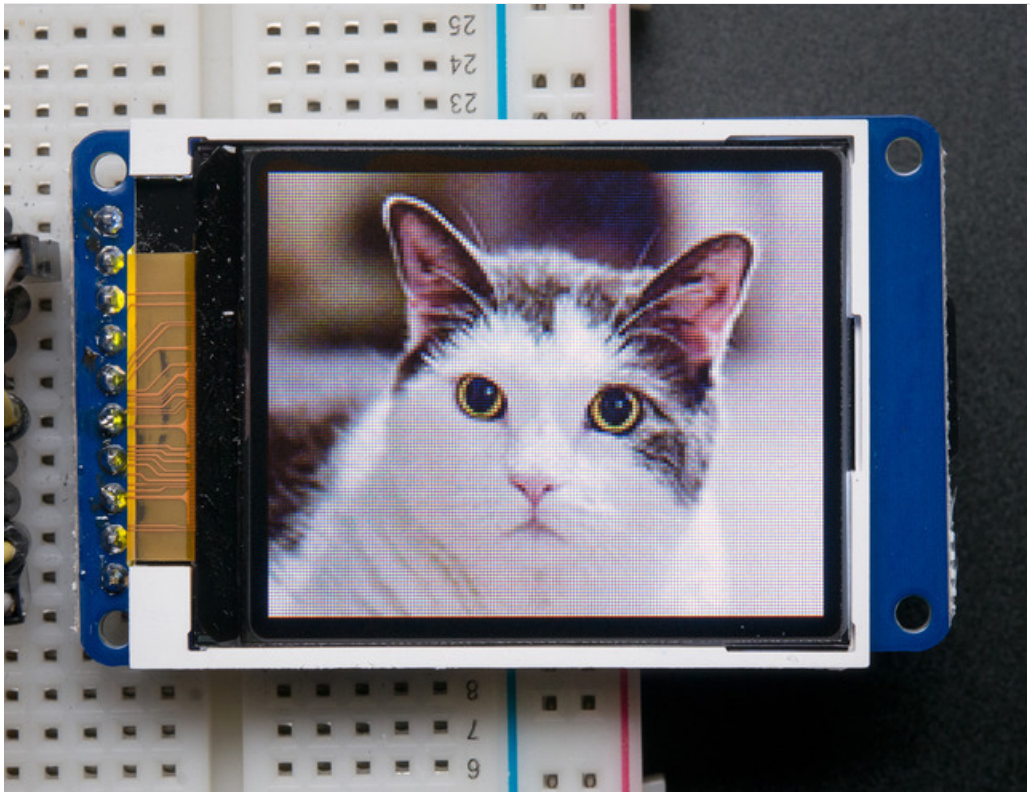
Guide Contents

Guide Contents 2
Overview 3
Breakout Pinouts 5
Breakout Assembly 6
- Prepare the header strip: 6
- Add the breakout board: 7
- And Solder! 7

Breakout Wiring & Test 9
Install Adafruit ST7735 TFT Library 9
Changing Pins 11
Assembling the Shield 13
- Cut the Header Sections 13
- Insert the Headers into an Arduino 14
- Add the Shield 15
- And Solder! 16

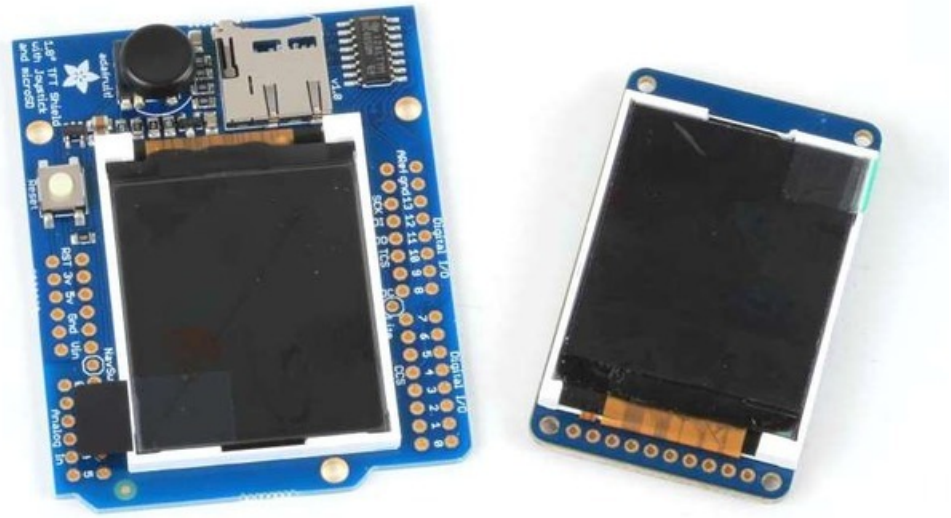
Testing the Shield 17
Reading the Joystick 19
Graphics Library 21
Displaying Bitmaps 22
Breakout Wiring 23
Example Sketch 25
Downloads 26
Files & Datasheets 26
Breakout Schematic 26
Breakout Fabrication print 26
Shield Schematic 27
Shield Fabrication Print 27

Overview



This tutorial is for our 1.8" diagonal TFT display. It comes packaged as a breakout or as an Arduino shield. Both styles have a microSD interface for storing files and images. These are both great ways to add a small, colorful and bright display to any project. Since the display uses 4-wire SPI to communicate and has its own pixel-addressable frame buffer, it requires little memory and only a few pins. This makes it ideal for use with small microcontrollers.

The shield version plugs directly into an Arduino with no wiring required. The breakout version can be used with every kind of microcontroller.

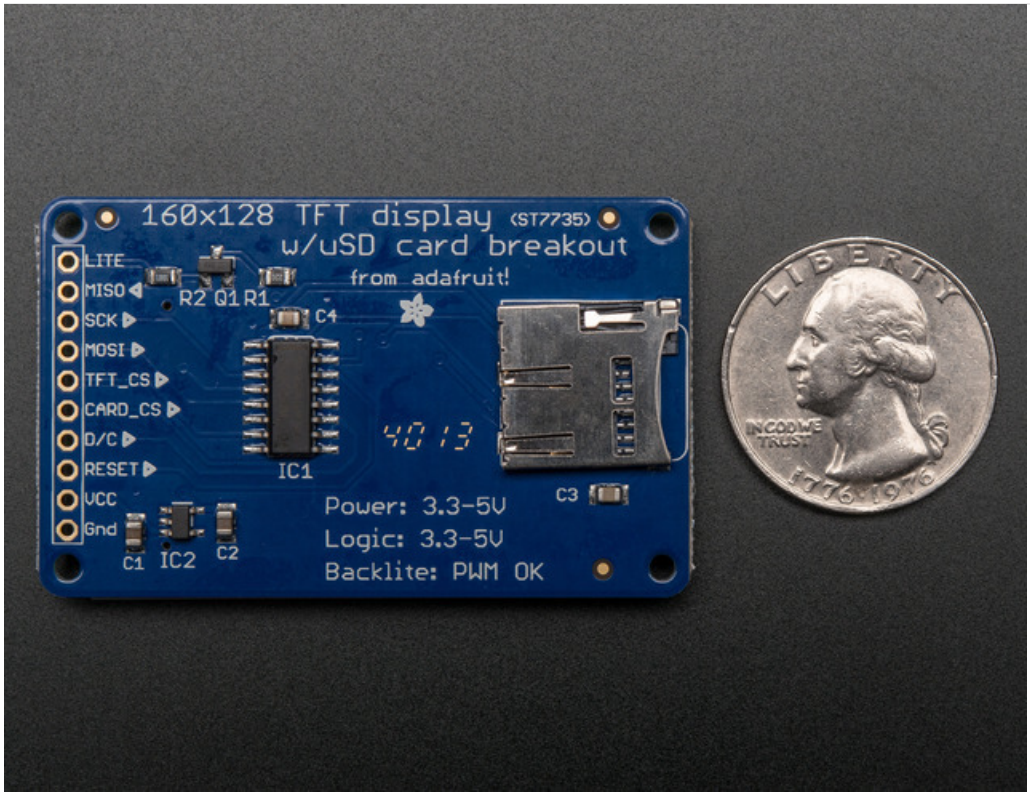


The 1.8" display has 128x160 color pixels. Unlike the low cost "Nokia 6110" and similar LCD displays, which are CSTN type and thus have poor color and slow refresh, this display is a true TFT! The TFT driver (ST7735R) can display full 18-bit color (262,144 shades!). And the LCD will always come with the same driver chip so there's no worries that your code will not work from one to the other.

Both boards have the TFT soldered on (it uses a delicate flex-circuit connector) as well as a ultra-low-dropout 3.3V regulator and a 3/5V level shifter so you can use it with 3.3V **or** 5V power and logic. These also include a microSD card holder so you can easily load full color bitmaps from a FAT16/FAT32 formatted microSD card. And on the Shield version, we've added a nifty 5-way joystick navigation switch!

You can pick up one of these displays in the Adafruit shop!
1.8" 18-bit color TFT breakout
1.8" 18-bit Color TFT Shield

Breakout Pinouts

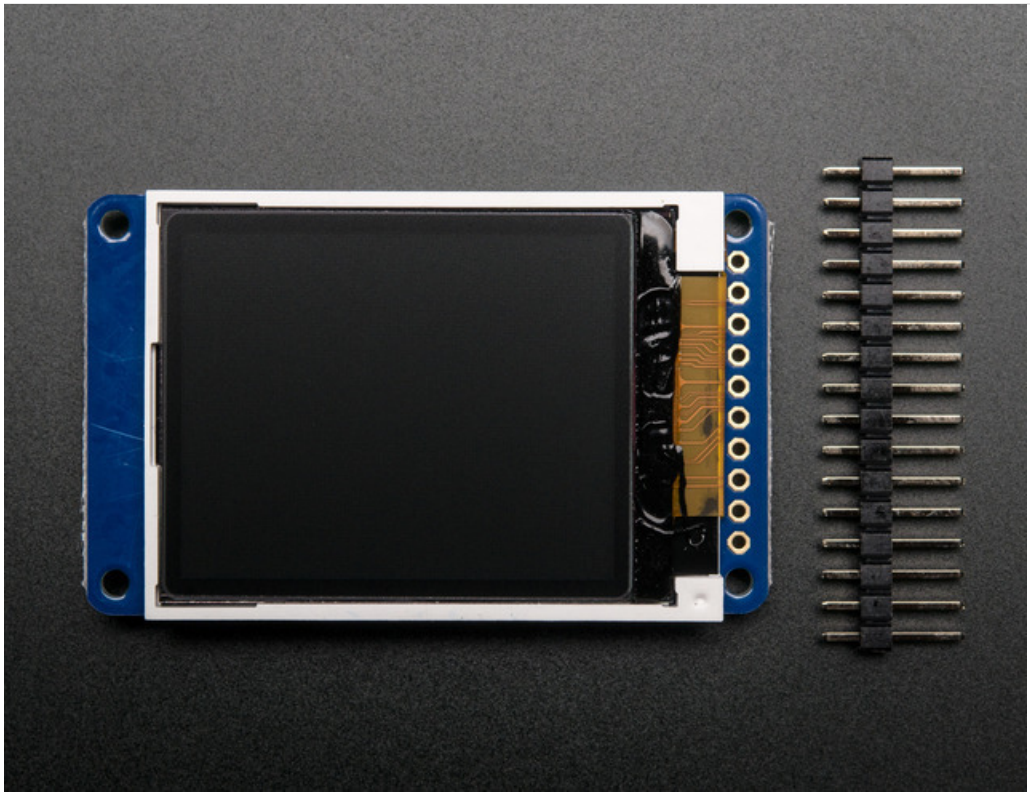


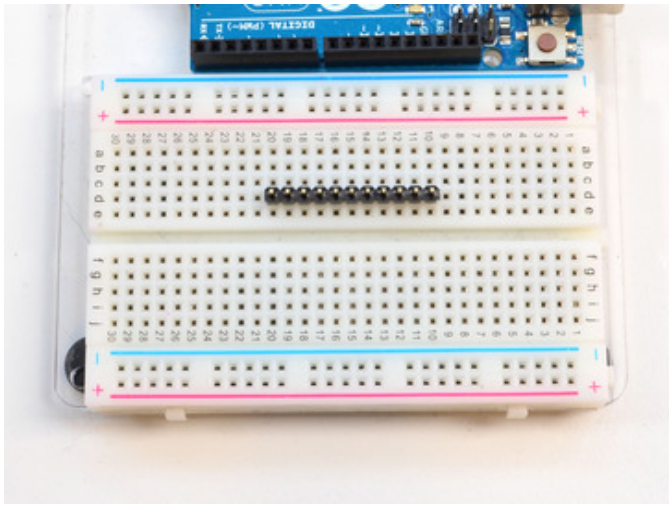
This color display uses SPI to receive image data. That means you need at least 4 pins - clock, data in, tft cs and d/c. If you'd like to have SD card usage too, add another 2 pins - data out and card cs. However, there's a couple other pins you may want to use, lets go thru them all!

- **Lite** - this is the PWM input for the backlight control. Connect to 3-5VDC to turn on the backlight. Connect to ground to turn it off. Or, you can PWM at any frequency.
- **MISO** - this is the SPI Master In Slave Out pin, its used for the SD card. It isn't used for the TFT display which is write-only
- **SCLK** - this is the SPI clock input pin
- **MOSI** - this is the SPI Master Out Slave In pin, it is used to send data from the microcontroller to the SD card and/or TFT
- **TFT_CS** - this is the TFT SPI chip select pin
- **Card CS -** this is the SD card chip select, used if you want to read from the SD card.
- **D/C** - this is the TFT SPI data or command selector pin
- **RST** - this is the TFT reset pin. Connect to ground to reset the TFT! Its best to have this pin controlled by the library so the display is reset cleanly, but you can also connect it to the Arduino Reset pin, which works for most cases.
- **Vcc** - this is the power pin, connect to 3-5VDC - it has reverse polarity protection but try to wire it right!
- **GND** - this is the power and signal ground pin

For the level shifter we use the CD74HC4050 which has a typical propagation delay of ~10ns

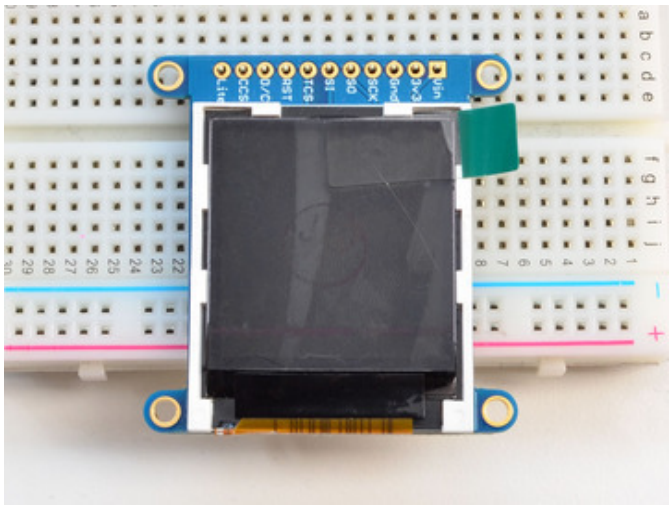
Breakout Assembly





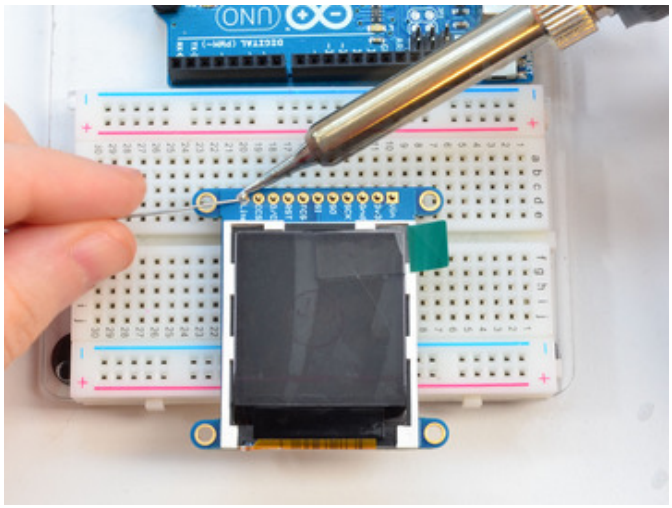
Prepare the header strip:

Cut the strip to length if necessary. It will be easier to solder if you insert it into a breadboard - **long pins down**



Add the breakout board:

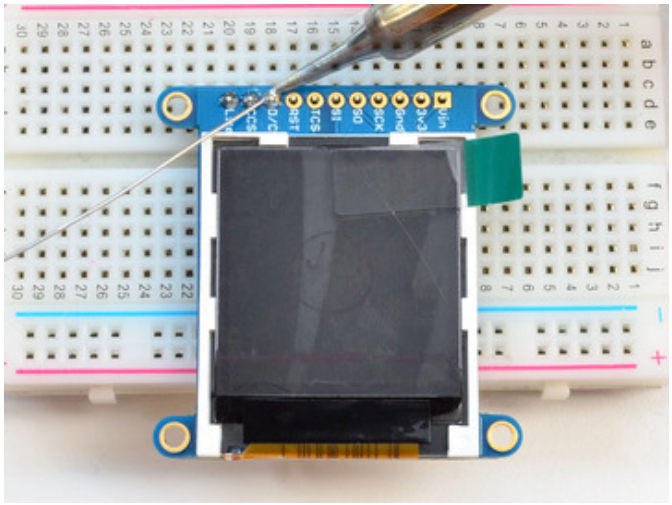
Place the breakout board over the pins so that the short pins poke through the breakout pads

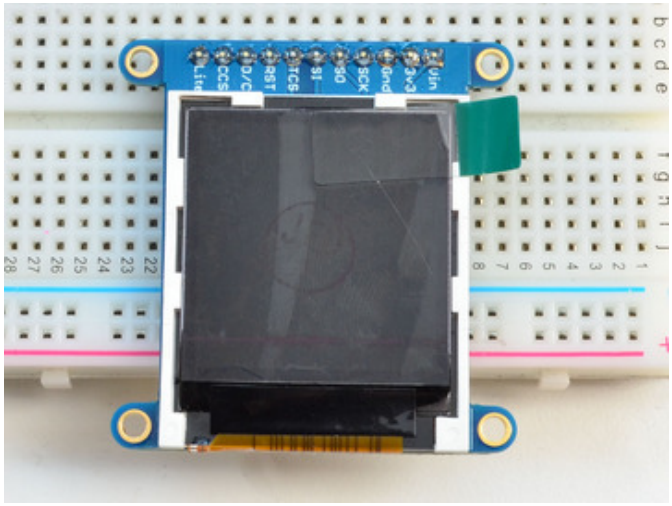


And Solder!

Be sure to solder all pins for reliable electrical contact.

(For tips on soldering, be sure to check out our Guide to Excellent Soldering (https://adafru.it/aTk).

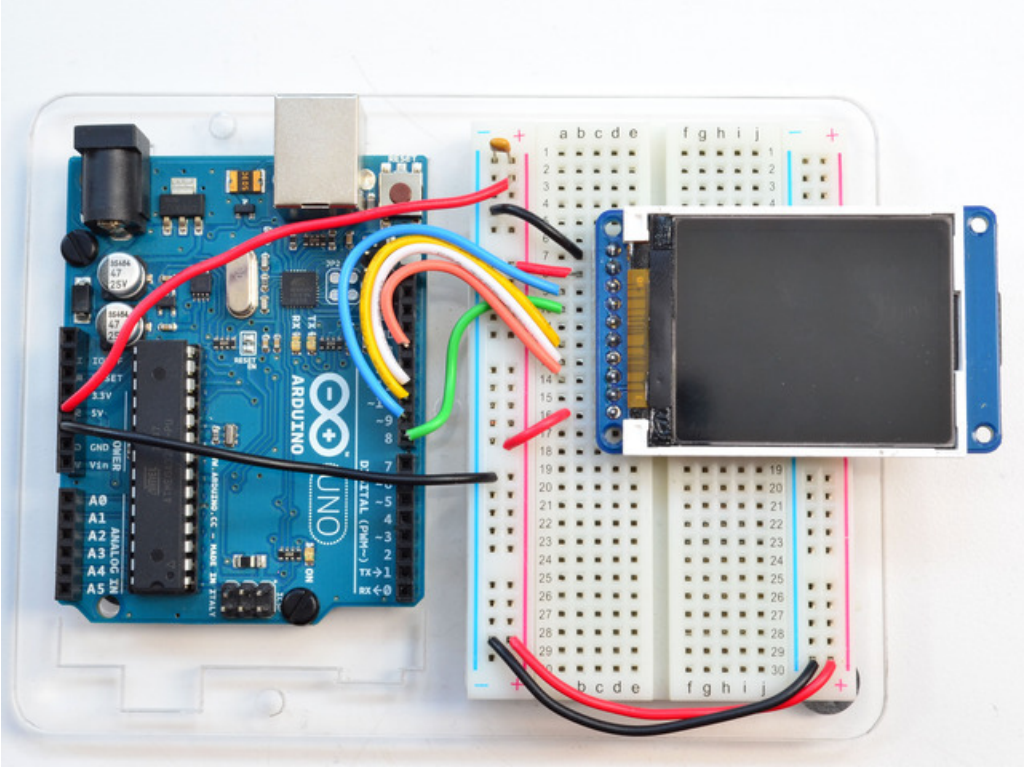




You're done! Check your solder joints visually and continue onto the next steps

Breakout Wiring & Test

There are two ways to wire up these displays - one is a more flexible method (you can use any pins on the Arduino) and the other is much faster (4-8x faster, but you are required to use the hardware SPI pins) We will begin by showing how to use the faster method, you can always change the pins later for flexible 'software SPI'



Wiring up the display in SPI mode is pretty easy as there's not that many pins! We'll be using hardware SPI, but you can also use software SPI (any pins) later. Start by connecting the power pins

- **3-5V Vin** connects to the Arduino **5V** pin - red wires
- **GND** connects to Arduino ground - black wires
- **CLK** connects to SPI clock. On Arduino Uno/Duemilanove/328-based, thats **Digital 13**. On Mega's, its **Digital 52** and on Leonardo/Due its **ICSP-3** (See SPI Connections for more details) - this is the orange wire
- **MOSI** connects to SPI MOSI. On Arduino Uno/Duemilanove/328-based, thats **Digital 11**. On Mega's, its **Digital 51** and on Leonardo/Due its **ICSP-4** (See SPI Connections for more details) - this is the white wire
- **CS** connects to our SPI Chip Select pin. We'll be using **Digital 10** but you can later change this to any pin - this is the yellow wire
- **RST** connects to our TFT reset pin. We'll be using **Digital 9** but you can later change this pin too - this is the blue wire
- **D/C** connects to our SPI data/command select pin. We'll be using **Digital 8** but you can later change this pin too - this is the green wire

Install Adafruit ST7735 TFT Library

We have example code ready to go for use with these TFTs. It's written for Arduino, which should be portable to any microcontroller by adapting the C++ source.

Two libraries need to be downloaded and installed: first is the Adafruit_ST7735 library (this contains the low-level code specific to this device), and second is the Adafruit GFX Library (which handles graphics operations common to many

displays we carry). If you have **Adafruit_GFX** already, make sure its the most recent version since we've made updates for better performance

Download Adafruit ST7735 Library

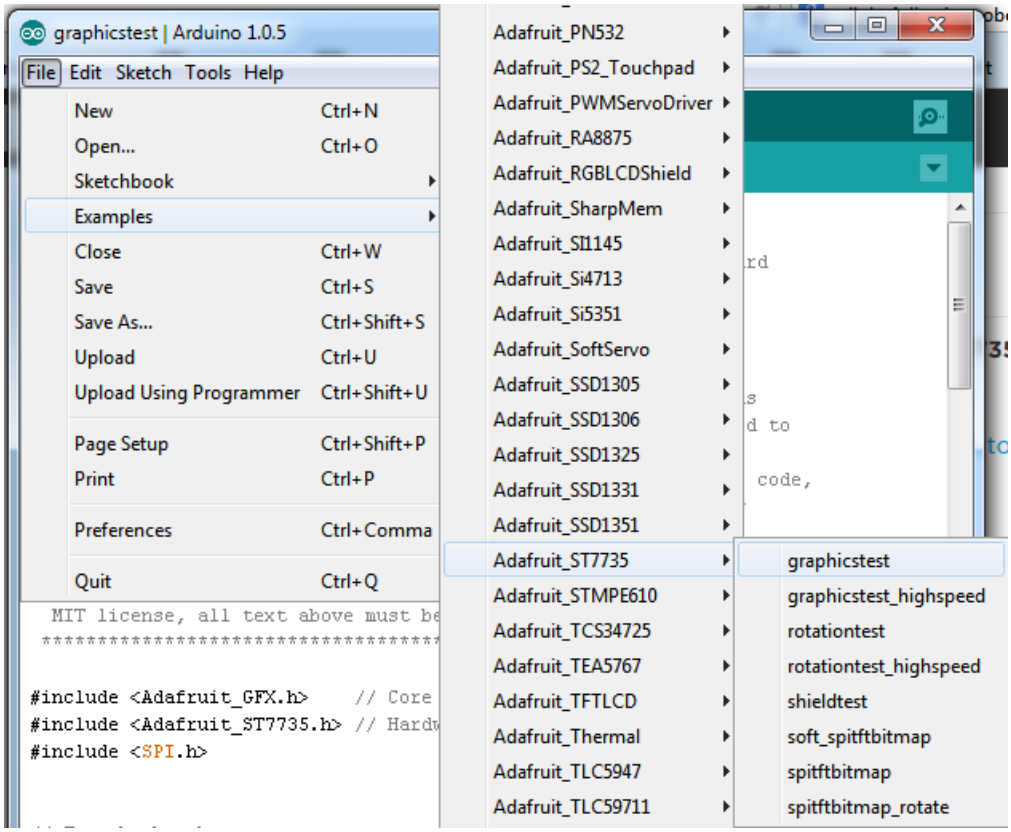
https://adafru.it/dXk

Download Adafruit GFX Library

https://adafru.it/cBB

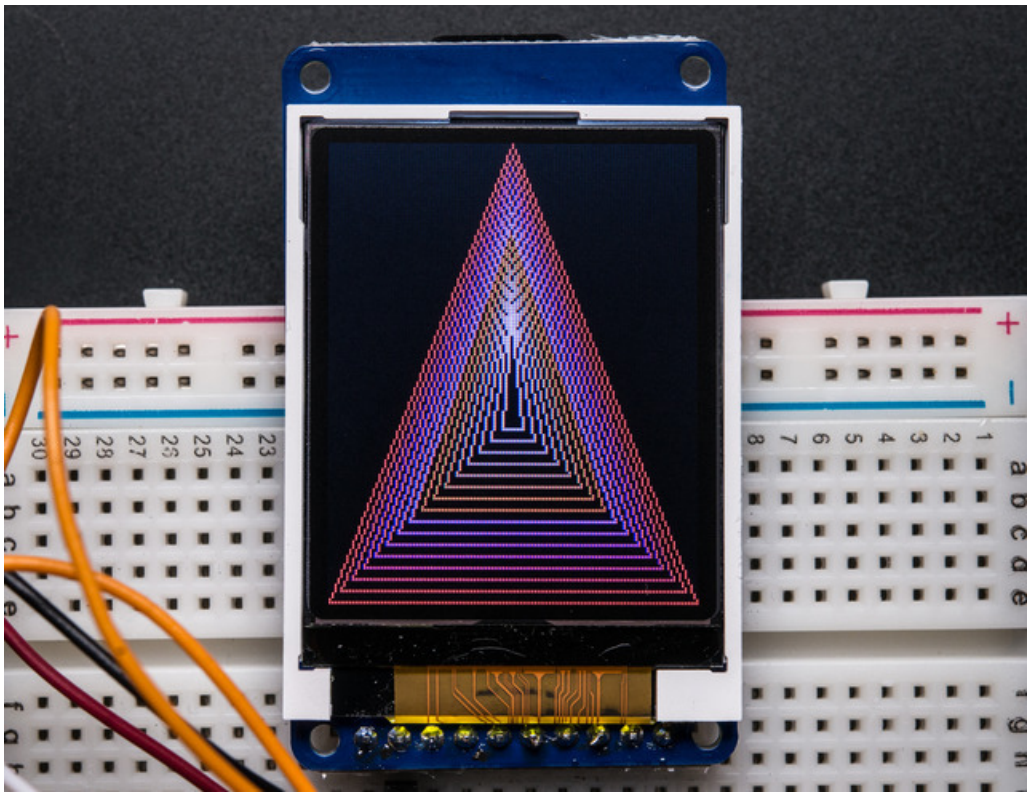
Download both ZIP files, uncompress and rename the folders to **Adafruit_ST7735** (contains **Adafruit_ST7735.cpp** and **.h**) and **Adafruit_GFX** (contains **Adafruit_GFX.cpp** and **.h**) respectively. Then place them inside your Arduino **libraries** folder and restart the Arduino IDE. If this is all unfamiliar, we have a tutorial introducing Arduino library concepts and installation.

Restart the IDE!



After restarting the Arduino software, you should see a new **example** folder called **Adafruit_ST7735** and inside, an example called **graphicstest**.

Now upload the sketch to your Arduino. You may need to press the Reset button to reset the arduino and TFT. You should see a collection of graphical tests draw out on the TFT.



Once uploaded, the Arduino should perform all the test display procedures! If you're not seeing anything - first check if you have the backlight on, if the backlight is not lit something is wrong with the power/backlight wiring. If the backlight is lit but you see nothing on the display make sure you're using our suggested wiring.

Changing Pins

Now that you have it working, there's a few things you can do to change around the pins.

If you're using Hardware SPI, the CLOCK and MOSI pins are 'fixed' and cant be changed. But you can change to software SPI, which is a bit slower, and that lets you pick any pins you like. Find these lines:

```
// Option 1 (recommended): must use the hardware SPI pins
// (for UNO thats sclk = 13 and sid = 11) and pin 10 must be
// an output. This is much faster - also required if you want
// to use the microSD card (see the image drawing example)
Adafruit_ST7735 tft = Adafruit_ST7735(TFT_CS,  TFT_DC, TFT_RST);

// Option 2: use any pins but a little slower!
#define TFT_SCLK 13   // set these to be whatever pins you like!
#define TFT_MOSI 11   // set these to be whatever pins you like!
//Adafruit_ST7735 tft = Adafruit_ST7735(TFT_CS, TFT_DC, TFT_MOSI, TFT_SCLK, TFT_RST);
```

Comment out option 1, and uncomment option 2. Then you can change the **TFT_** pins to whatever pins you'd like!

You can also save a pin by setting

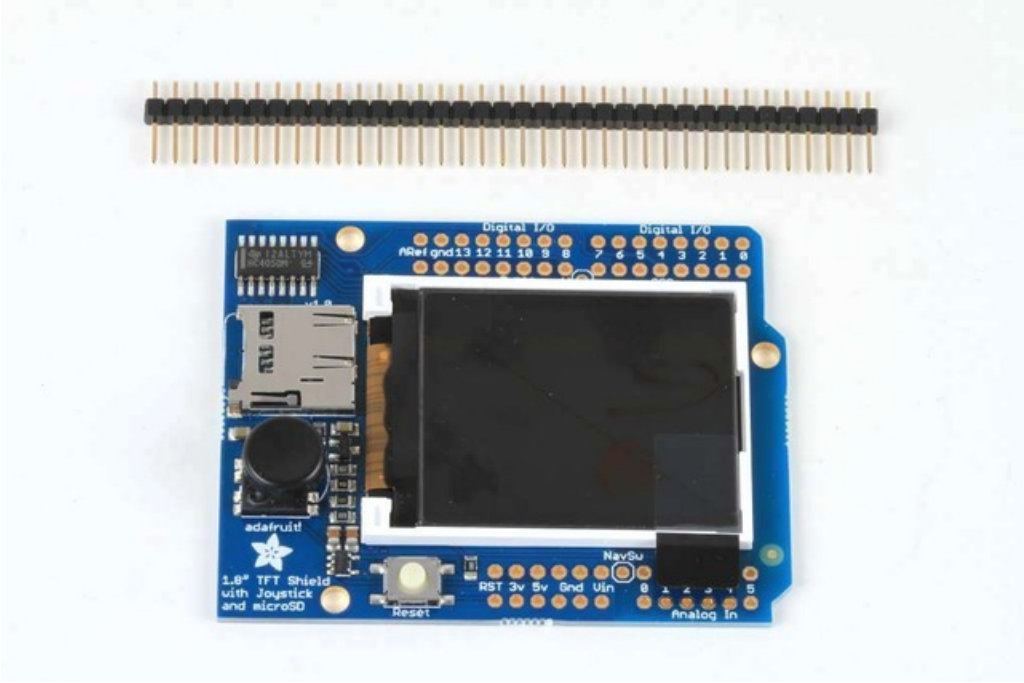
`#define TFT_RST  9`

to

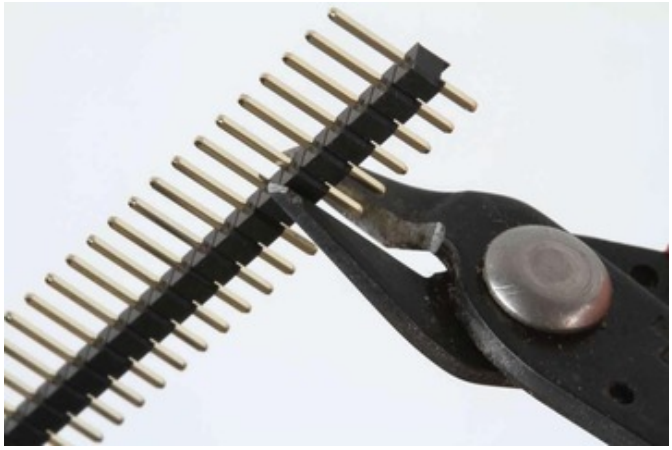
```
#define TFT_RST   0
```

and connecting the RST line to the Arduino Reset pin. That way the Arduino will auto-reset the TFT as well.

Assembling the Shield



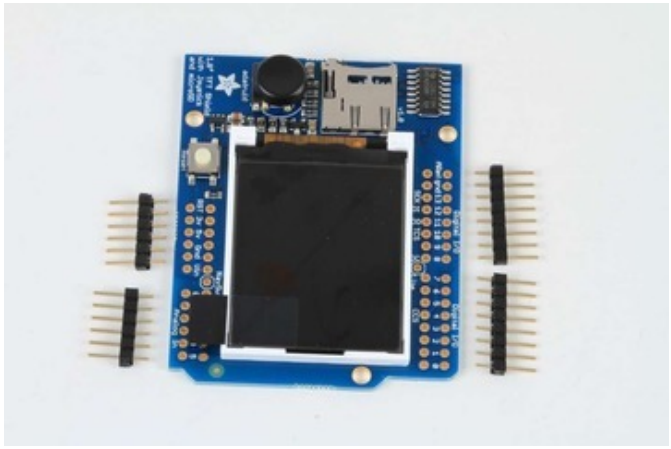
The shield comes with all surface mount parts pre-soldered. All that remains is to install the headers!

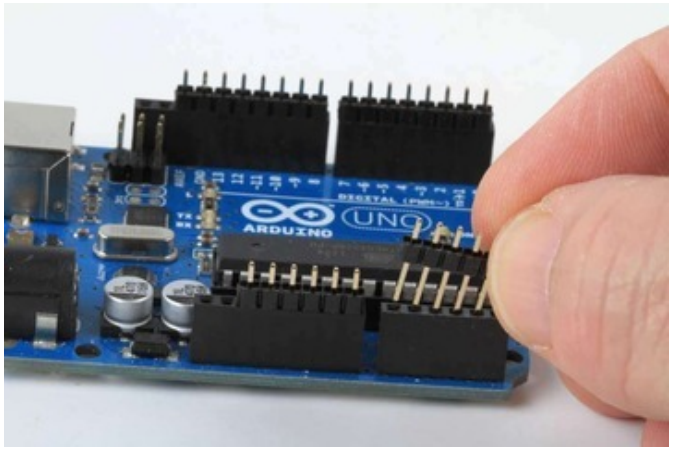


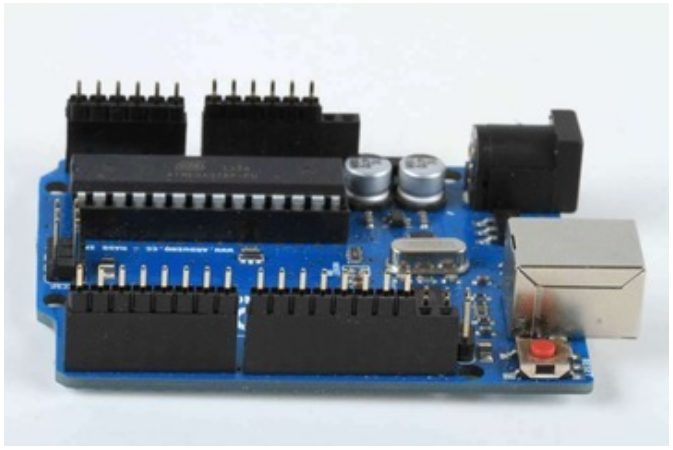
Cut the Header Sections

Cut the breakaway header strip into sections to fit the holes on the edge of the shield. You will need 2 sections of 6-pins and 2 sections of 8 pins.

You can use wire-cutters as shown or pliers to snap them apart between pins.



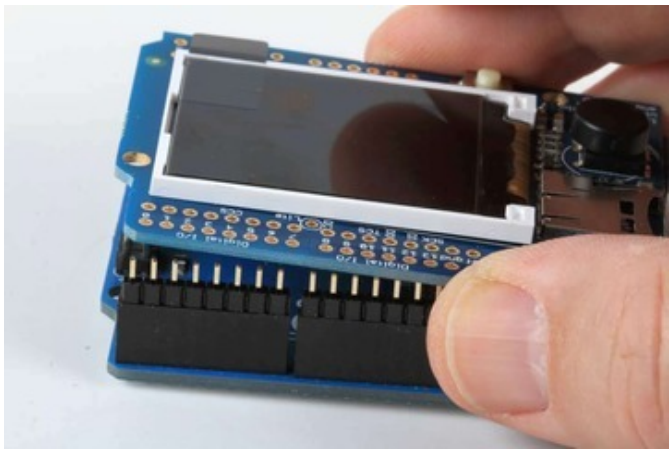


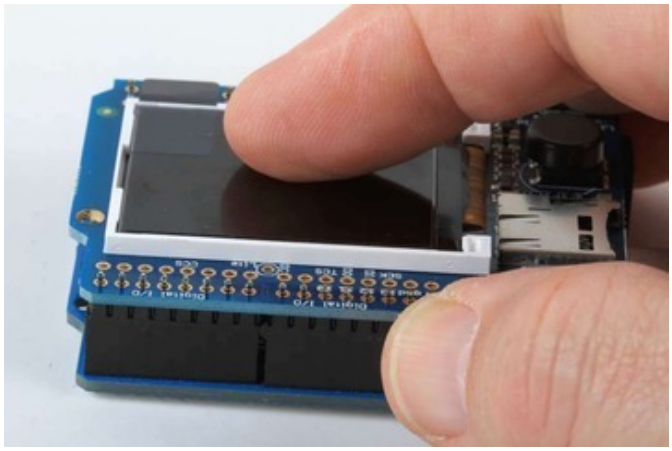


Insert the Headers into an Arduino

To align the header strips for soldering, insert them (long pins down) into the headers of an Arduino.

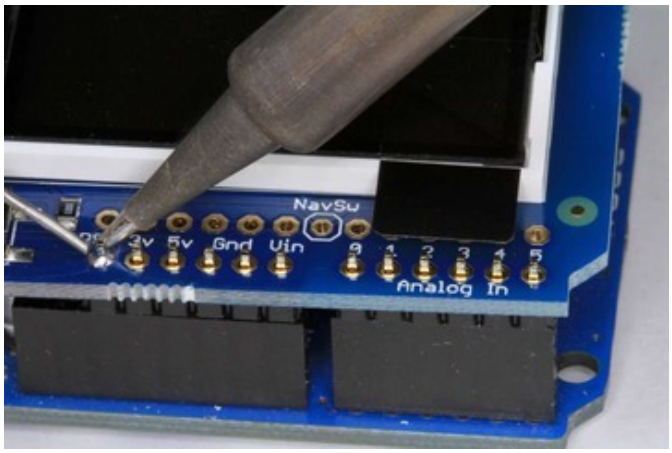
Note that for R3 and later Arduinos, there will be an extra 2 unused pins on the end closest the USB and DC power jacks.

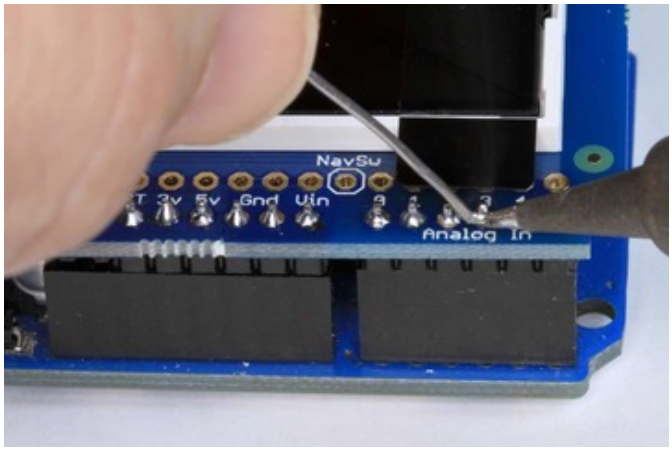




Add the Shield

Place the shield over the header strips so that the short pins stick up through the holes.



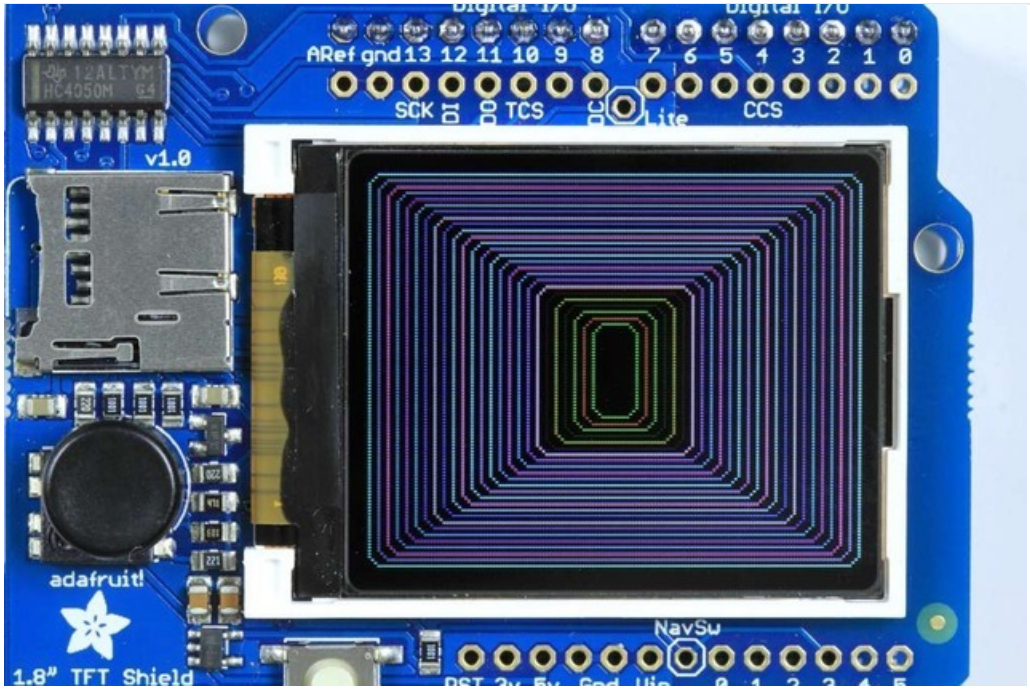


And Solder!

Solder each pin to assure good electrical contact.

For tips on soldering see the Adafruit Guide to Excellent Soldering (https://adafru.it/aTk).

Testing the Shield



You can test your assembled shield using the example code from the library.

Download our Arduino library (see bottom of page) from github by clicking on **Download** in the top right corner. Uncompress the folder and rename it **Adafruit_ST7735** - inside the folder you should see the **Adafruit_ST7735.cpp** and**Adafruit_ST7735.h** files. Install the **Adafruit_ST7735** library foler by placing it in your ***arduinosketchfolder*/libraries** folder. You may have to create the **libraries** subfolder if this is your first library. You can read more about installing libraries in our tutorial.

Restart the Arduino IDE. You should now be able to select **File** > **Examples** > **Adafruit_ST7735** > **graphicstest** sketch. Upload the sketch to your Arduino wired as above.

The shield uses the "Classic Arduino" SPI wiring and will perform best with Atmega 328-based Arduinos such as the Uno or Duemilanove. But it will also work with the Leonardo or Mega.

If you are using an Arduino UNO, Duemilanove or compatible with the ATmega328 chipset, you don't have to do anything!

If you're using a Mega, Leonardo, Due or other non-ATmega328 chipset, you'll have to make a modification

To use with the shield, modify the example code pin definitions as follows.

Find these lines:

```
// Option 1 (recommended): must use the hardware SPI pins
// (for UNO thats sclk = 13 and sid = 11) and pin 10 must be
// an output. This is much faster - also required if you want
// to use the microSD card (see the image drawing example)
Adafruit_ST7735 tft = Adafruit_ST7735(TFT_CS,  TFT_DC, TFT_RST);

// Option 2: use any pins but a little slower!
#define TFT_SCLK 13   // set these to be whatever pins you like!
#define TFT_MOSI 11   // set these to be whatever pins you like!
//Adafruit_ST7735 tft = Adafruit_ST7735(TFT_CS, TFT_DC, TFT_MOSI, TFT_SCLK, TFT_RST);
```

The Example code has 2 options for defining the display object. **Uno, Duemilanove** and other Atmega 328-based processors can use the "Option 1" version of the constructor for best performance:

```
Adafruit_ST7735 tft = Adafruit_ST7735(TFT_CS,  TFT_DC, TFT_RST);
```

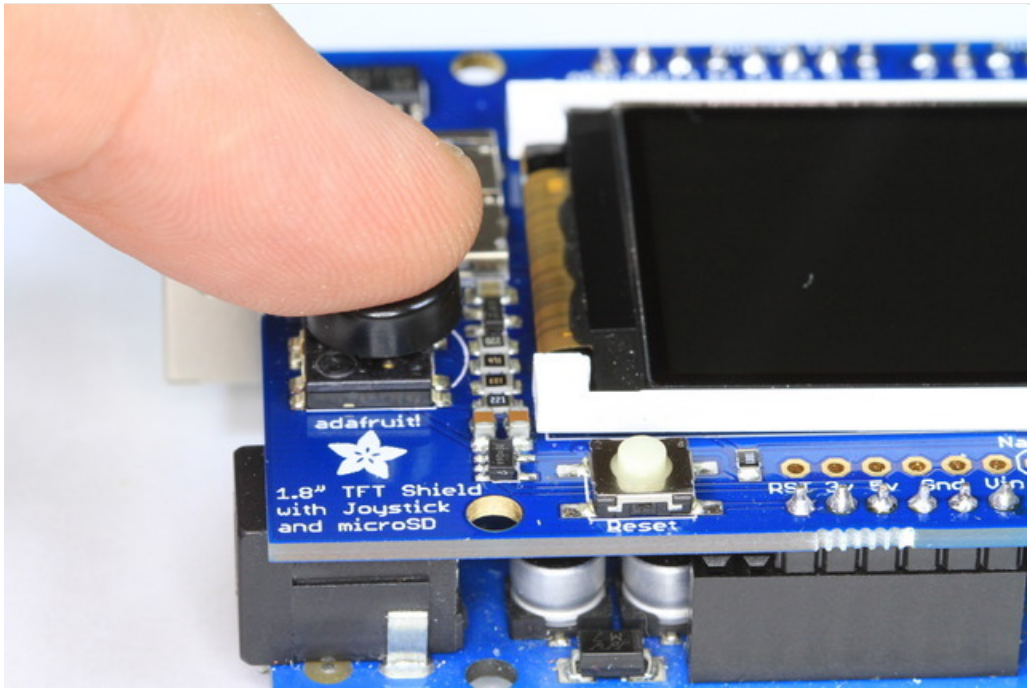
Mega and **Leonardo** users should use the "Option 2" version of the constructor for compatibility:

```
Adafruit_ST7735 tft = Adafruit_ST7735(TFT_CS, TFT_DC, TFT_MOSI, TFT_SCLK, TFT_RST);
```

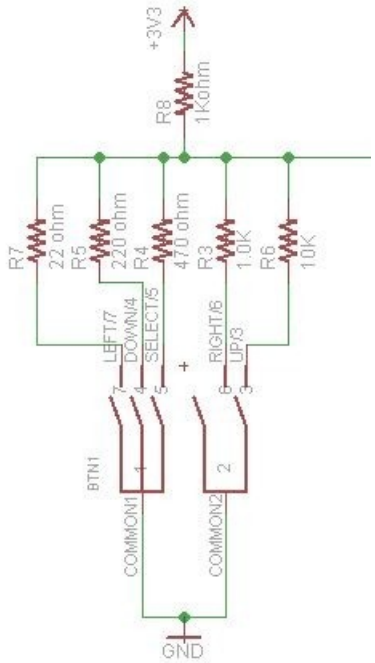
Be sure to select only one option and comment out the other with a pair of //'s.

Now upload the sketch to see the graphical display!

Reading the Joystick



The 5-way joystick on the shield is great for implementing menu navigation or even for use as a tiny game controller. To minimize the number of pins required, the joystick uses a different resistor on each leg of the control to create a variable voltage divider that can be monitored with a single analog pin. Each movement of the joystick control connects a different resistor and results in a different voltage reading.



In the code example below, the CheckJoystick() function reads the analog pin and compares the result with 5 different ranges to determine which (if any) direction the stick has been moved. If you upload this to your Arduino and open the Serial Monitor, you will see the current joystick state printed to the screen.

You can use this code as the input method for your menu system or game:

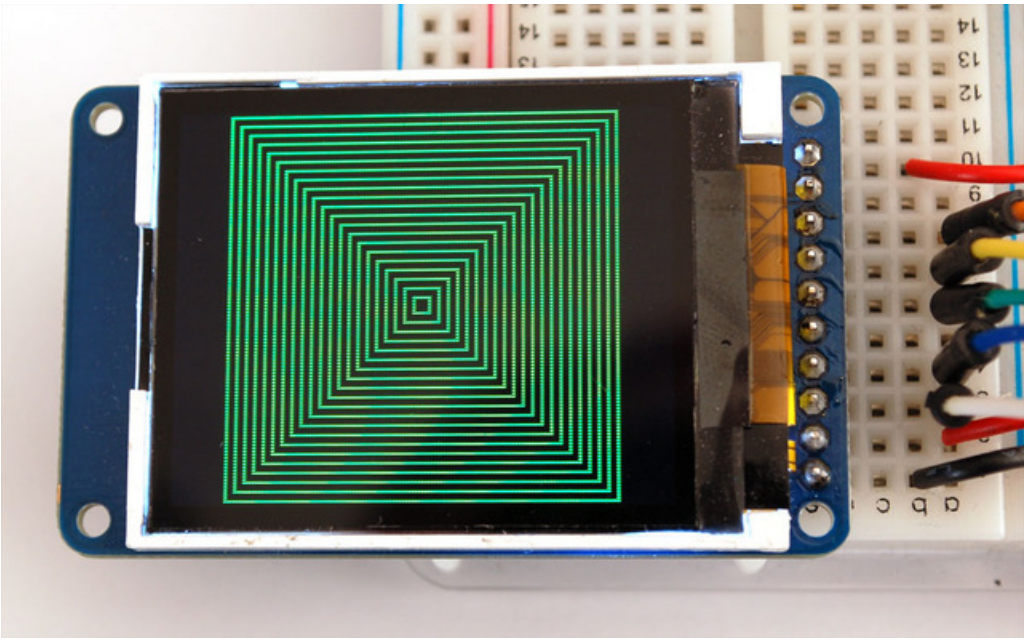
```
void setup()
{
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

#define Neutral 0
#define Press 1
#define Up 2
#define Down 3
#define Right 4
#define Left 5

// Check the joystick position
int CheckJoystick()
{
  int joystickState = analogRead(3);

  if (joystickState < 50) return Left;
  if (joystickState < 150) return Down;
  if (joystickState < 250) return Press;
  if (joystickState < 500) return Right;
  if (joystickState < 650) return Up;
  return Neutral;
}

void loop()
{
  int joy = CheckJoystick();
  switch (joy)
  {
    case Left:
      Serial.println("Left");
      break;
    case Right:
      Serial.println("Right");
      break;
    case Up:
      Serial.println("Up");
      break;
    case Down:
      Serial.println("Down");
      break;
    case Press:
      Serial.println("Press");
      break;
  }
}
```

Graphics Library

We've written a full graphics library specifically for this display which will get you up and running quickly. The code is written in C/C++ for Arduino but is easy to port to any microcontroller by rewritting the low level pin access functions.

The TFT LCD library is based off of the Adafruit GFX graphics core library. GFX has many ready to go functions that should help you start out with your project. Its not exhaustive and we'll try to update it if we find a really useful function. Right now it supports pixels, lines, rectangles, circles, round-rects, triangles and printing text as well as rotation.

Two libraries need to be downloaded and installed: first is the ST7735 library (this contains the low-level code specific to this device), and second is the Adafruit GFX Library (which handles graphics operations common to many displays we carry). Download both ZIP files, uncompress and rename the folders to 'Adafruit_ST7735' and 'Adafruit_GFX' respectively, place them inside your Arduino libraries folder and restart the Arduino IDE. If this is all unfamiliar, we have a tutorial introducing Arduino library concepts and installation.



Check out the GFX tutorial for detailed information about what is supported and how to use it!

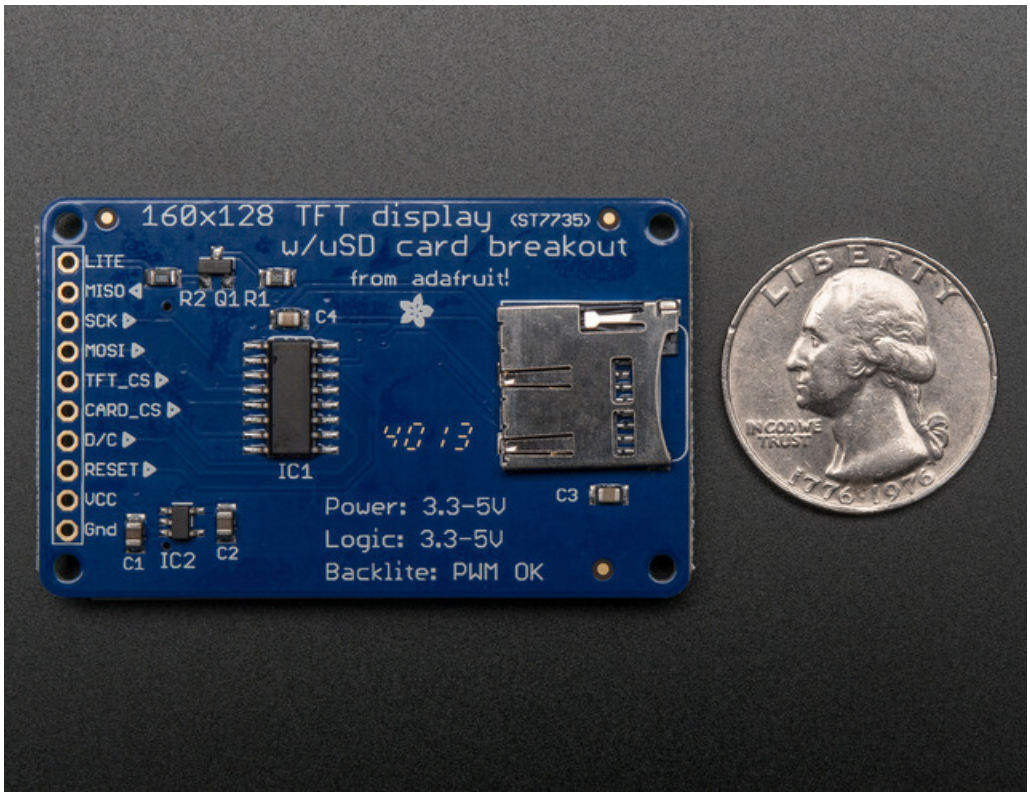
Displaying Bitmaps

In this example, we'll show how to display a 128x160 pixel full color bitmap from a microSD card.

We have an example sketch in the library showing how to display full color bitmap images stored on an SD card. You'll need a microSD card such as this one. You'll also need to download our SD library modified to allow faster reads (these changes will hopefully be added to arduino v23) but for now you can download the new library here. Download the library by clicking the **Downloads** button and uncompressing the folder. Replace the files in your**ArduinoIDE/libraries/SD** folder (make a backup of course) and restart the IDE.

You'll also need an image. We suggest starting with this bitmap of a parrot. If you want to later use your own image, use an image editing tool and crop your image to no larger than 160 pixels high and 128 pixels wide. Save it as a 24-bit color **BMP** file - it must be 24-bit color format to work, even if it was originally a 16-bit color image - becaue of the way BMPs are stored and displayed!

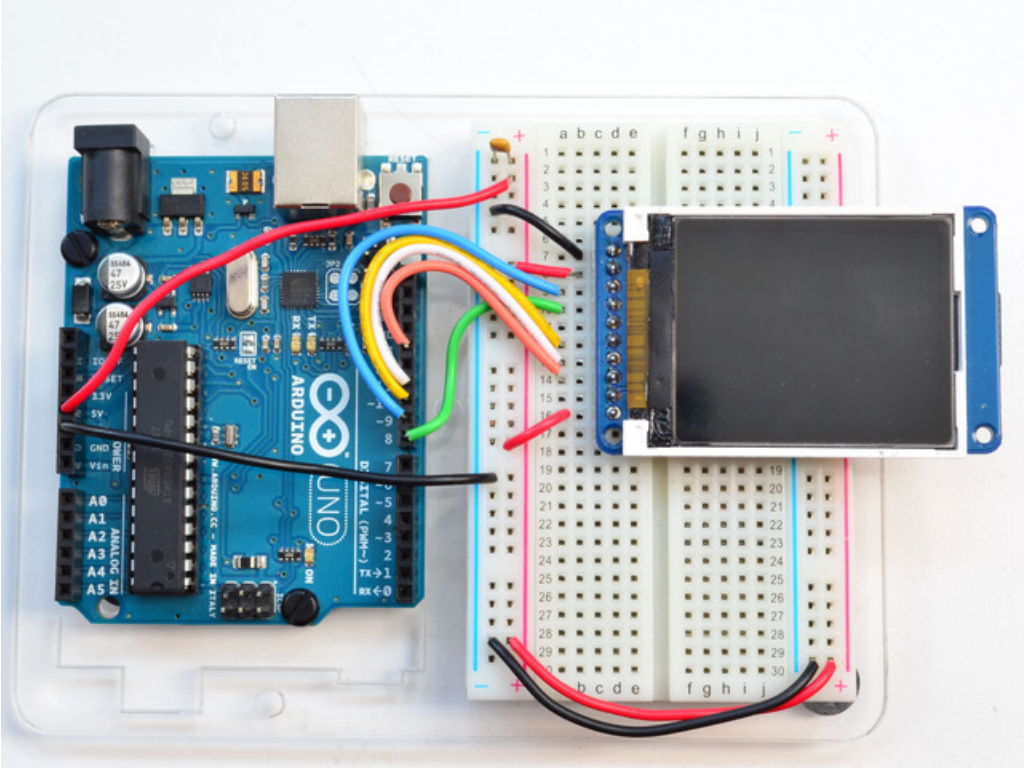
Copy the **parrot.bmp** to the microSD card and insert it into the micro SD card holder on your shield or breakout board.



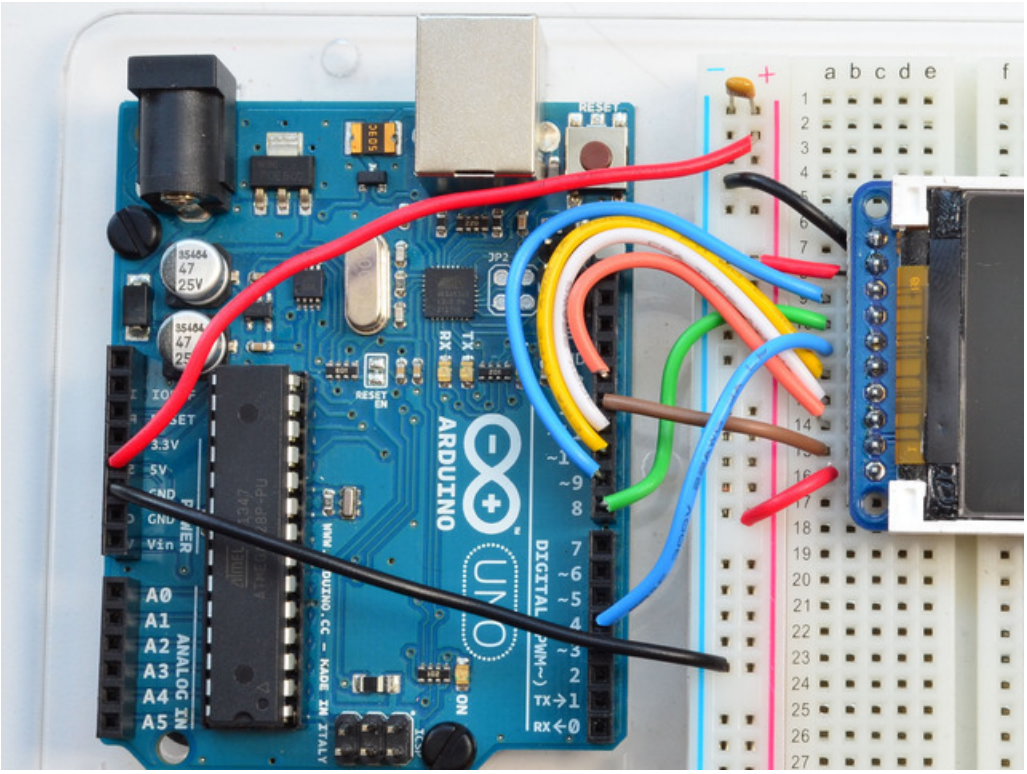
Breakout Wiring

Shield users can skip directly to the "**Example Sketch**" section.

Wire up the TFT as described on the wiring & test page. Test that your wiring is correct by uploading the graphics test sketch with the high speed SPI lines.



Once you are sure that the TFT is wired correctly, add the two wires for talking to the SD card. Connect **CARD_CS** (the unconnected pin in the middle) to digital pin **4** (you can change this later to any pin you want). Connect **MISO** (second from the right) to the Arduino's hardware SPI **MISO** pin. For Classic arduinos, this is pin **12**. For Mega's this is pin **50**. You can't change the **MISO** pin, its fixed in the chip hardware.



Example Sketch

Load the **spitftbitmap** example sketch into the Arduino IDE.

Breakout and **Shield on an Arduino Uno** users can use the code as-is.

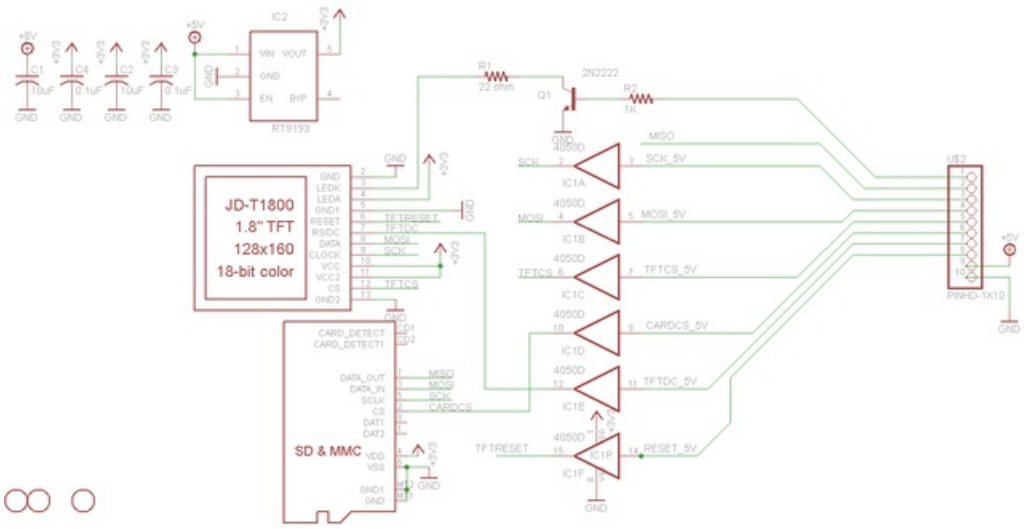
Now upload the **spitftbitmap** example sketch to the Arduino. It should display the parrot image. If you have any problems, check the serial console for any messages such as not being able to initialize the microSD card or not finding the image.

Downloads

Files & Datasheets

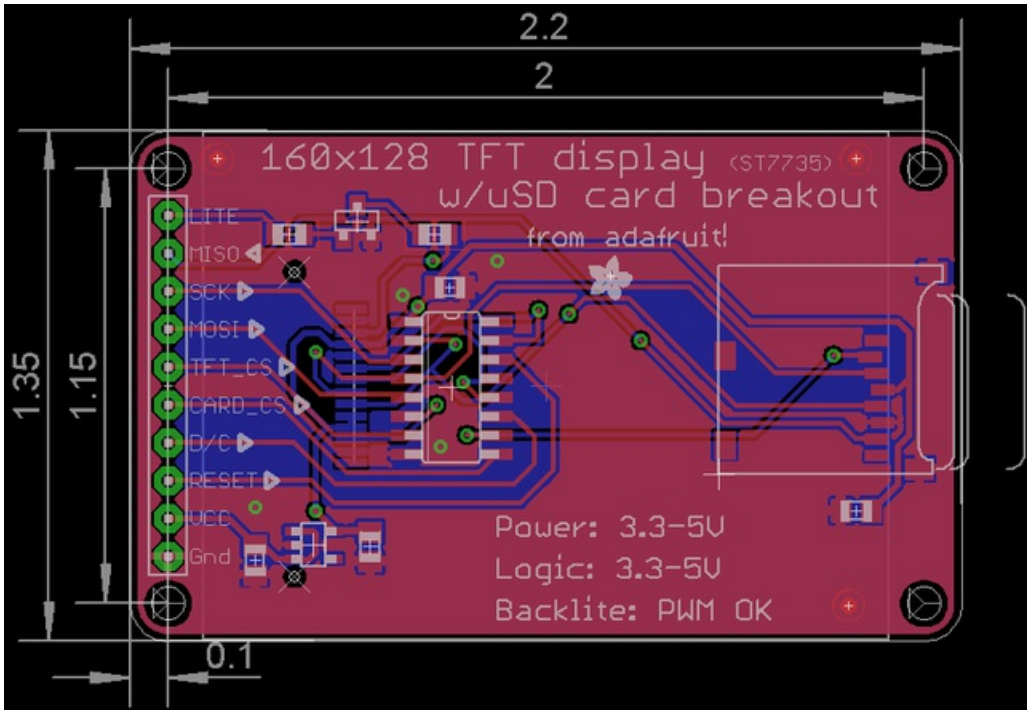
- Adafruit GFX library
- Adafruit ST7735 library (See our detailed tutorial for installation assistance)
- Fritzing object in the Adafruit library
- Datasheet for the display
- Datasheet for the display driver chip.
- EagleCAD PCB files for TFT shield
- EagleCAD PCB files for TFT breakout

Breakout Schematic

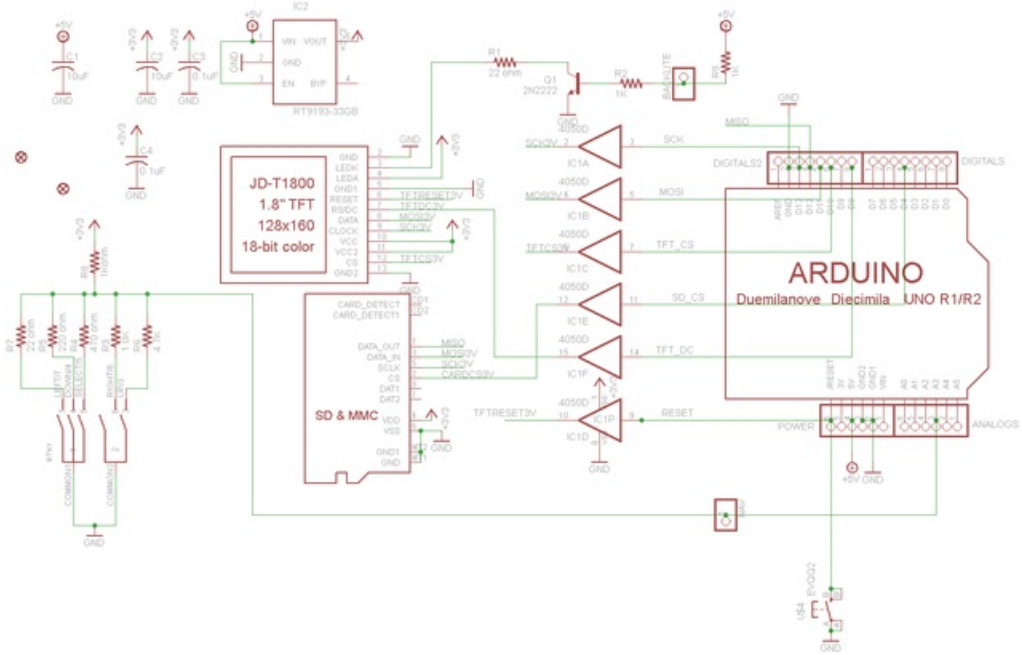


For the level shifter we use the CD74HC4050 which has a typical propagation delay of ~10ns

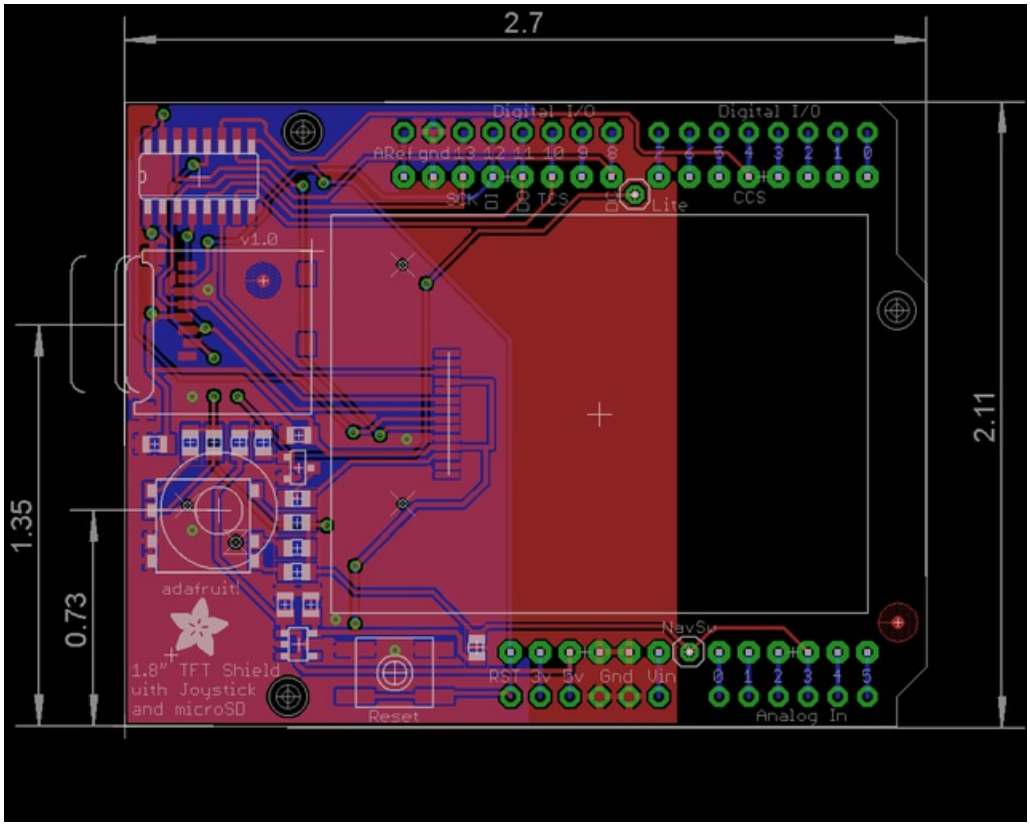
Breakout Fabrication print



Shield Schematic



Shield Fabrication Print


2.7
Digital I/O
Digital I/O
ARef gnd 13 12 11 10 9 8
7 6 5 4 3 2 1 0
SCK
TCS
Lite
CCS
v1.0
2.11
1.35
0.73
adafruit
1.8" TFT Shield
with Joystick
and microSD
Reset
NavSw
RST 3v 5v Gnd Vin
0 1 2 3 4 5
Analog In