# Guide Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td>3</td>
</tr>
<tr>
<td>Pinouts</td>
<td>5</td>
</tr>
<tr>
<td>Power Pins</td>
<td>5</td>
</tr>
<tr>
<td>SPI Pins</td>
<td>5</td>
</tr>
<tr>
<td>Touch Screen control pins</td>
<td>6</td>
</tr>
<tr>
<td>SD Card control pins</td>
<td>7</td>
</tr>
<tr>
<td>TFT Graphics Test</td>
<td>8</td>
</tr>
<tr>
<td>Install Libraries</td>
<td>8</td>
</tr>
<tr>
<td>Install Adafruit HX8357D TFT Library</td>
<td>8</td>
</tr>
<tr>
<td>Basic Graphics Test</td>
<td>9</td>
</tr>
<tr>
<td>Adafruit GFX Library</td>
<td>12</td>
</tr>
<tr>
<td>Resistive Touch Screen</td>
<td>14</td>
</tr>
<tr>
<td>Touchscreen Paint Demo</td>
<td>14</td>
</tr>
<tr>
<td>Resistive Touch Screen</td>
<td>16</td>
</tr>
<tr>
<td>Touchscreen Paint Demo</td>
<td>16</td>
</tr>
<tr>
<td>Drawing Bitmaps</td>
<td>19</td>
</tr>
<tr>
<td>Downloads</td>
<td>21</td>
</tr>
<tr>
<td>Datasheets &amp; More</td>
<td>21</td>
</tr>
<tr>
<td>Troubleshooting</td>
<td>23</td>
</tr>
</tbody>
</table>

Display does not work on initial power but does work after a reset.
Overview

A Feather board without ambition is a Feather board without FeatherWings! Spice up your Feather project with a beautiful 3.5" touchscreen display shield with built in microSD card socket. This TFT display is 3.5" diagonal with a bright 6 white-LED backlight. You get a massive 480x320 pixels with individual 16-bit color pixel control. It has way more resolution than a black and white 128x64 display, and twice as much as our 2.4" TFT FeatherWing. As a bonus, this display comes with a resistive touchscreen attached to it already, so you can detect finger presses anywhere on the screen.

This FeatherWing uses a SPI display, touchscreen and SD card socket. It works with any and all Feathers but given the large display it works best with our faster boards like the ESP8266, ESP32, M4, M0, nRF52, WICED, and Teensy. We also include an SPI resistive touchscreen controller so you only need one additional pin to add a high quality touchscreen controller. One more pin is used for an optional SD card that can be used for storing images for display.
This Wing comes fully assembled with dual sockets for your Feather to plug into. You get two sockets per pin so you can plug in wires if you want to connect to Feather pins. Alternatively, each pin has a large square pad on the PCB for direct soldering.
Pinouts

Unlike most FeatherWings, the TFT FeatherWing is fully assembled and has a dual socket set for your Feather to plug into.

This makes it really easy to use, but a little different to change if you don't want the default setup

The FeatherWing has an ENABLE switch which will disable both Feather and Wing - so make sure it is turned ON while you’re using it

Power Pins

The 3.5" TFT FeatherWing has a really beefy backlight, and we don't want to overwhem the 3.3V regulator on the Feather so we use two Schottkey diodes to take the higher of VBAT and VUSB to feed into the backlight boost converter.

The GND and 3.3V pins are used for powering the rest of the device such as the TFT and touchscreen controller.

You can turn off the 3.3V power supply with the EN pin or the switch attached to that pin. The enable/disable switch on the bottom of the Wing will also deactivate the backlight.

Note that on the Teensy 3x Feather Adapter, this pin doesn't do anything and on the FONA feather, this will not disable the VBAT power supply which is used to power the cellular module.

SPI Pins
The TFT display, SD card and touch screen use the SPI interface to communicate. That means **MISO**, **MOSI** and **SCK** are used whenever either are accessed.

*(Ooops the highlighted pins are off by one, they should be shifted to the right one pin!)*

In addition, for the TFT display there is are **D/C** (Data/Command) and **CS** (Chip Select) pins. These are used to select the display and tell it what kind of data is being sent. These pins can theoretically be changed by cutting the jumper trace and soldering a small wire from the farther-from-the-socket-header pad to the pin you'd like to use.

On the **ESP8266**, TFT-CS is pin #0, TFT-DC is pin #15

On the **ESP32**, TFT-CS is pin #15, TFT-DC is pin #33

On the **Atmega32u4, ATmega328p, M4 or M0 Feather**, TFT-CS is pin #9, TFT-DC is pin #10

On the **Teensy Feather**, TFT-CS is pin #4, TFT-DC is pin #10

On the **WICED Feather**, TFT-CS is PA15 and TFT-DC is PB4

On the **nRF52 Feather**, TFT-CS is #31 and TFT-DC is #11

There is also **LITE** pin which is not connected to any pads but you can use to control the backlight. Pull low to turn off the backlight. You can connect it to a PWM output pin.

> Note: Pin 9 is used for communication with the SIM800 chip on the Feather Fona. You will have to remap pin 9 to an unused pin when using with a Feather Fona.

**Touch Screen control pins**
The touch screen also has a Chip Select line, labeled RT. This pin can theoretically be changed by cutting the jumper trance and soldering a small wire from the right-hand pad to the pin you’d like to use.

On the ESP8266, RT is pin #16
On the ESP32, RT is pin #32
On the Atmega32u4, ATmega328p, M4 or M0 Feather, RT is pin #6
On the Teensy Feather, RT is pin #3
On the WICED Feather, RT is PC7
On the nRF52 Feather, RT is #30

There is also an IRQ pin which is not connected to any pads but you can use to detect when touch events have occurred.

SD Card control pins

The SD Card also has a Chip Select line, labeled SD. This pin can theoretically be changed by cutting the jumper trance and soldering a small wire from the right-hand pad to the pin you’d like to use.

On the ESP8266, SD is pin #2
On the ESP32 SD is pin #14
On the Atmega32u4, ATmega328p, M4 or M0 Feather, SD is pin #5
On the Teensy Feather, SD is pin #8
On the WICED Feather, SD is PC5
On the nRF52 Feather, SD is pin #27
The TFT FeatherWing is basically a combination of our 3.5" TFT Breakout (https://adafruit.it/dRb) with the STMPE610 resistive touch-screen breakout attached (http://adafruit.it/1571).

Before you continue, make sure the FeatherWing's ENABLE switch is turned ON! If it's OFF, the Feather won't work and you will be very confused :)

Install Libraries

You'll need a few libraries to use this FeatherWing!

From within the Arduino IDE, open up the Library Manager...

Install Adafruit HX8357D TFT Library

We have example code ready to go for use with these TFTs.
Two libraries need to be downloaded and installed: first is the Adafruit HX8357 library (https://adafru.it/dQW) (this contains the low-level code specific to this device), and second is the Adafruit GFX Library (https://adafru.it/aJa) (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.

Search for HX8357 and install the Adafruit HX8357 library that pops up!

![Library Manager: HX8357 library](image)

Next up, search for Adafruit GFX and locate the core library. A lot of libraries may pop up because we reference it in the description so just make sure you see Adafruit GFX Library in bold at the top.

Install it!

![Library Manager: Adafruit GFX library](image)

Repeat the search and install steps for the Adafruit_ImageReader and Adafruit_ZeroDMA libraries. That’s four libraries in total.

For more details about this process, we have a tutorial introducing Arduino library concepts and installation (https://adafru.it/aYM).

Basic Graphics Test

After restarting the Arduino software, you should see a new example folder called Adafruit_HX8357 and inside, an example called graphicstest_fetherwing.
Upload that sketch to your Feather. You should see a collection of graphical tests draw out on the TFT.

If you're having difficulties, check the serial console. The first thing the sketch does is read the driver configuration from the TFT, you should see the same numbers as below. That will help you determine if the TFT is found, if not, check your Feather soldering!
Once you've got the demo working, check out the detailed documentation over at http://learn.adafruit.com/adafruit-gfx-graphics-library (https://adafru.it/aPx) for more information on how to use the GFX library to draw whatever you like!
Adafruit GFX Library

The Adafruit_GFX library for Arduino provides a common syntax and set of graphics functions for all of our TFT, LCD and OLED displays. This allows Arduino sketches to easily be adapted between display types with minimal fuss...and any new features, performance improvements and bug fixes will immediately apply across our complete offering of color displays.

The GFX library is what lets you draw points, lines, rectangles, round-rects, triangles, text, etc.
Check out our detailed tutorial here: [http://learn.adafruit.com/adafruit-gfx-graphics-library](http://learn.adafruit.com/adafruit-gfx-graphics-library)

It covers the latest and greatest of the GFX library. The GFX library is used in both 8-bit and SPI modes so the underlying commands (drawLine() for example) are identical!
Resistive Touch Screen

The LCD has a 4-wire resistive touch screen glued onto it. You can use this for detecting finger-presses, stylus', etc. Normally, you'll need 4 pins to talk to the touch panel but we decided to go all snazzy and put a dedicated touch screen driver onto the shield. The driver shares the SPI pins with the TFT and SD card, so only one extra pin is needed. This allows you to query the controller when you're ready to read touchscreen data, and saves 3 pins.

To control the touchscreen you'll need one more library (https://adafru.it/d4f) - the STMPE610 controller library which does all the low level chatting with the STMPE610 driver chip. Click below to download and then install it as before.

Touchscreen Paint Demo

Now that you've got the basic TFT graphics demo working, let's add in the touchscreen. Run and upload the `touchpaint_featherwing` demo

Upload to your Feather and have fun!

The touch screen is made of a thin glass sheet, and it's very fragile - a small crack or break will make the entire touch screen unusable. Don't drop or roughly handle the TFT and be especially careful of the corners and edges. When pressing on the touchscreen, sometimes people can use the tip of their fingers, or a fingernail. If you don't find the touchscreen responds well to your fingers, you can use a rounded stylus which will certainly work. Do not press harder and harder until the screen cracks!

Getting data from the touchscreen is fairly straightforward. Start by creating the touchscreen object with

```
Adafruit_STMPE610 ts = Adafruit_STMPE610(STMPE_CS);
```

We're using hardware SPI so the clock, mosi and miso pins are not defined here. Then you can start the touchscreen with

```
ts.begin()
```

Check to make sure this returns a True value, which means the driver was found. If it wasn't, make sure you have the Feather soldered right and the correct CS pin!

Now you can call

```
if (! ts.bufferEmpty())
```

to check if there's any data in the buffer. The touchscreen driver will store touchpoints at all times. When you're ready to get the data, just check if there's any data in the buffer. If there is, you can call

```
TS_Point p = ts.getPoint();
```
To get the oldest point from the buffer, TS_Point has .x, .y and .z data points. The x and y points range from 0 to 4095. The STMPE610 does not store any calibration data in it and it doesn't know about rotation. So if you want to rotate the screen you'll need to manually rotate the x/y points! The z point is 'pressure' and ranges from 0 to 255, we don't use it here but you can experiment with it on your own, the harder you press, the lower the number.

Since data from the STMPE610 comes in 0-4095 but our screen is 320 pixels by 240 pixels, we can use map to convert 0-4095 to 0-320 or 0-240. Something like

```cpp
p.x = map(p.x, 0, 4095, 0, tft.width());
p.y = map(p.y, 0, 4095, 0, tft.height());
```

However, the touchscreen is a bit bigger than the screen, so we actually need to ignore presses beyond the touchscreen itself. We found that these numbers reflected the true range that overlaps the screen

```cpp
#define TS_MINX 150
#define TS_MINY 130
#define TS_MAXX 3800
#define TS_MAXY 4000
```

So we use

```cpp
p.x = map(p.x, TS_MINX, TS_MAXX, 0, tft.width());
p.y = map(p.y, TS_MINY, TS_MAXY, 0, tft.height());
```

instead.

One last point (pun intended!) since the touchscreen driver stores points in a buffer, you may want to ask the driver "is the touchscreen being pressed RIGHT NOW?" You can do that with

```cpp
if (ts.touched())
```
Resistive Touch Screen

The LCD has a 4-wire resistive touch screen glued onto it. You can use this for detecting finger-presses, stylus', etc. Normally, you’ll need 4 pins to talk to the touch panel but we decided to go all snazzy and put a dedicated touch screen driver onto the shield. The driver shares the SPI pins with the TFT and SD card, so only one extra pin is needed. This allows you to query the controller when you're ready to read touchscreen data, and saves 3 pins.

To control the touchscreen you'll need one more library (https://adafruit.it/d4f) - the STMPE610 controller library which does all the low level chatting with the STMPE610 driver chip. Use the library manager to install the Adafruit STMPE610 library.

Touchscreen Paint Demo

Now that you've got the basic TFT graphics demo working, let's add in the touchscreen. Run and upload the touchpaint_featherwing demo

- If you have the 2.4" TFT Featherwing, run the Adafruit ILI9341->touchpaint_featherwing demo
- If you have the 3.5" TFT Featherwing, run the Adafruit HX8357->touchpaint_featherwing demo

Upload to your Feather and have fun!
Getting data from the touchscreen is fairly straightforward. Start by creating the touchscreen object with

```c
Adafruit_STMPE610 ts = Adafruit_STMPE610(STMPE_CS);
```

We're using hardware SPI so the clock, mosi and miso pins are not defined here. Then you can start the touchscreen with

```c
ts.begin()
```

Check to make sure this returns a True value, which means the driver was found. If it wasn't, make sure you have the Feather soldered right and the correct CS pin!

Now you can call

```c
if (!ts.bufferEmpty())
```

to check if there's any data in the buffer. The touchscreen driver will store touchpoints at all times. When you're ready to get the data, just check if there's any data in the buffer. If there is, you can call

```c
TS_Point p = ts.getPoint();
```
To get the oldest point from the buffer. TS_Point has \( x, y \) and \( z \) data points. The \( x \) and \( y \) points range from 0 to 4095. The STMPE610 does not store any calibration data in it and it doesn't know about rotation. **So if you want to rotate the screen you'll need to manually rotate the \( x/y \) points!** The \( z \) point is 'pressure' and ranges from 0 to 255, we don't use it here but you can experiment with it on your own, the harder you press, the lower the number.

Since data from the STMPE610 comes in 0-4095 but our screen is 320 pixels by 240 pixels, we can use `map` to convert 0-4095 to 0-320 or 0-240. Something like

```c
p.x = map(p.x, 0, 4095, 0, tft.width());
p.y = map(p.y, 0, 4095, 0, tft.height());
```

However, the touchscreen is a bit bigger than the screen, so we actually need to ignore presses beyond the touchscreen itself. We found that these numbers reflected the true range that overlaps the screen.

```c
#define TS_MINX 150
#define TS_MINY 130
#define TS_MAXX 3800
#define TS_MAXY 4000
```

So we use

```c
p.x = map(p.x, TS_MINX, TS_MAXX, 0, tft.width());
p.y = map(p.y, TS_MINY, TS_MAXY, 0, tft.height());
```

instead.

One last point (pun intended!) since the touchscreen driver stores points in a buffer, you may want to ask the driver "is the touchscreen being pressed RIGHT NOW?" You can do that with

```c
if (ts.touched())
```
Drawing Bitmaps

There is a built-in microSD card slot on the FeatherWing, and we can use that to load bitmap images! You will need a microSD card formatted **FAT16 or FAT32** (they almost always are by default), and an SD card reader on whatever computer you’re currently reading this with.

It's really easy to draw bitmaps. Let's start by downloading this image of Adabot and friends:

![Adabot Image](image)

Download these two smaller images as well:

![Parrot Image](image)

![Wales Image](image)

The files should be renamed (if needed) to “adabot.bmp”, “parrot.bmp” and “wales.bmp”, respectively, and copied to the base directory of the microSD card (not inside a folder).

(If it's easier, you can also find these images in the “images” folder within the Adafruit_ImageReader library folder.)

Insert the microSD card into the socket in the shield. Now select the sketch file→examples→Adafruit_ImageViewer→FeatherWingHX8357 and upload this example to your Feather +
Wing. You will see the your electronic friends appear! (Plus parrots...and if you’re using one of the more powerful Feather boards, a whole lot of dragons.)

The Adafruit_ImageReader library, which is being used here to display .BMP images, is fully explained in its own page of the Adafruit_GFX guide (https://adafruit.it/DpM).
Downloads
Datasheets & More

- STMPE610 Touch Controller Datasheet (https://adafru.it/d4k)
- Datasheet for the HX8357D chipset controller (https://adafru.it/dQQ)
- Datasheet for the 3.5" TFT display (raw) (https://adafru.it/dR4)
- EagleCAD PCB files on GitHub (https://adafru.it/Btm)
- Fritzing object in Adafruit Fritzing library (https://adafru.it/aP3)
Troubleshooting

Display does not work on initial power but does work after a reset.
The display driver circuit needs a small amount of time to be ready after initial power. If your code tries to write to
the display too soon, it may not be ready. It will work on reset since that typically does not cycle power. If you are
having this issue, try adding a small amount of delay before trying to write to the display.

In Arduino, use `delay()` to add a few milliseconds before calling `tft.begin()`. Adjust the amount of delay as needed to
see how little you can get away with for your specific setup.