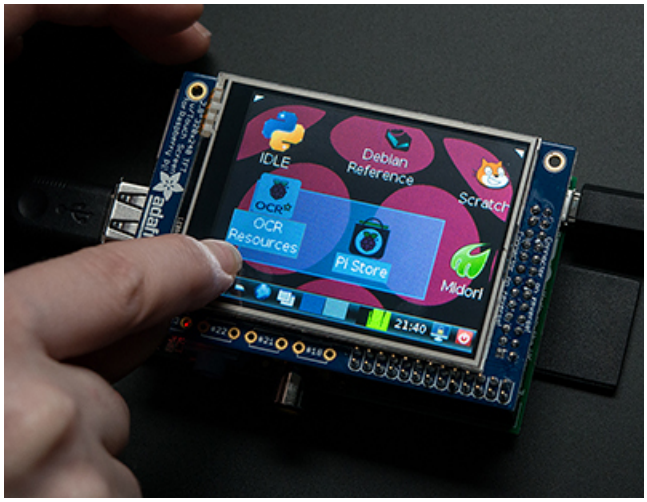
adafruit learning system

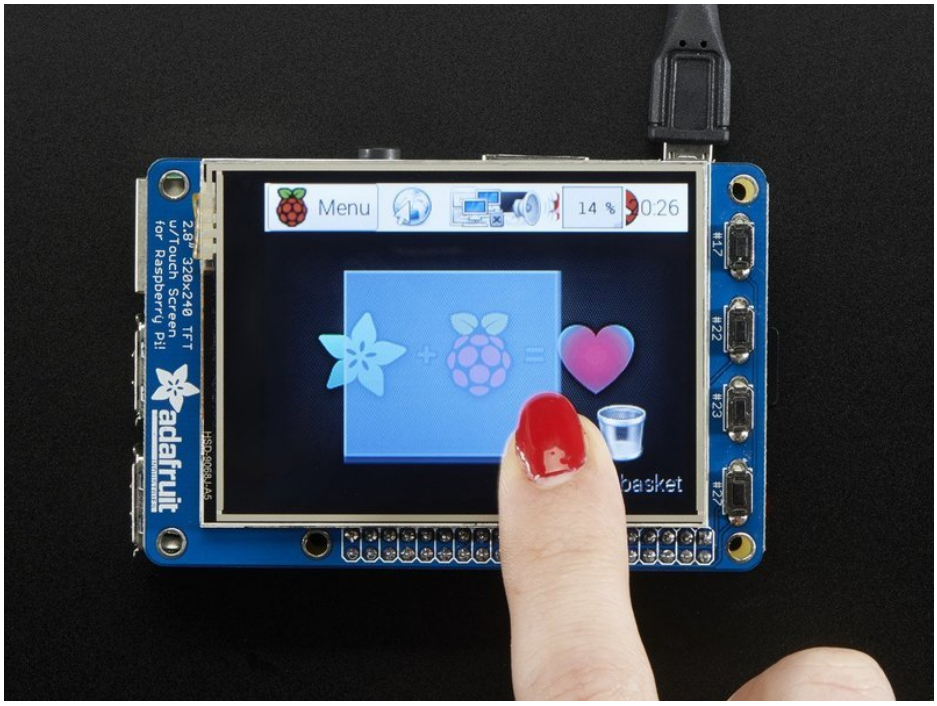
Adafruit PiTFT - 2.8" Touchscreen Display for Raspberry Pi

Created by lady ada



Last updated on 2020-03-09 11:33:40 PM UTC

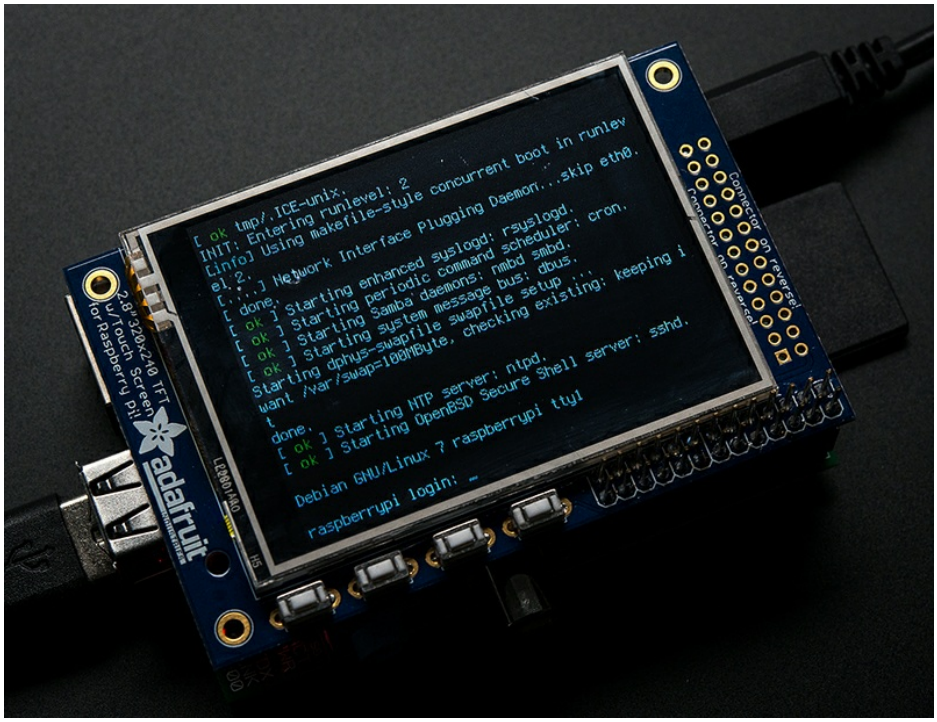
Overview



Is this not the cutest little display for the Raspberry Pi? It features a 2.8" display with 320x240 16-bit color pixels and a resistive touch overlay. The plate uses the high speed SPI interface on the Pi and can use the mini display as a console, X window port, displaying images or video etc. Best of all it plugs right in on top!

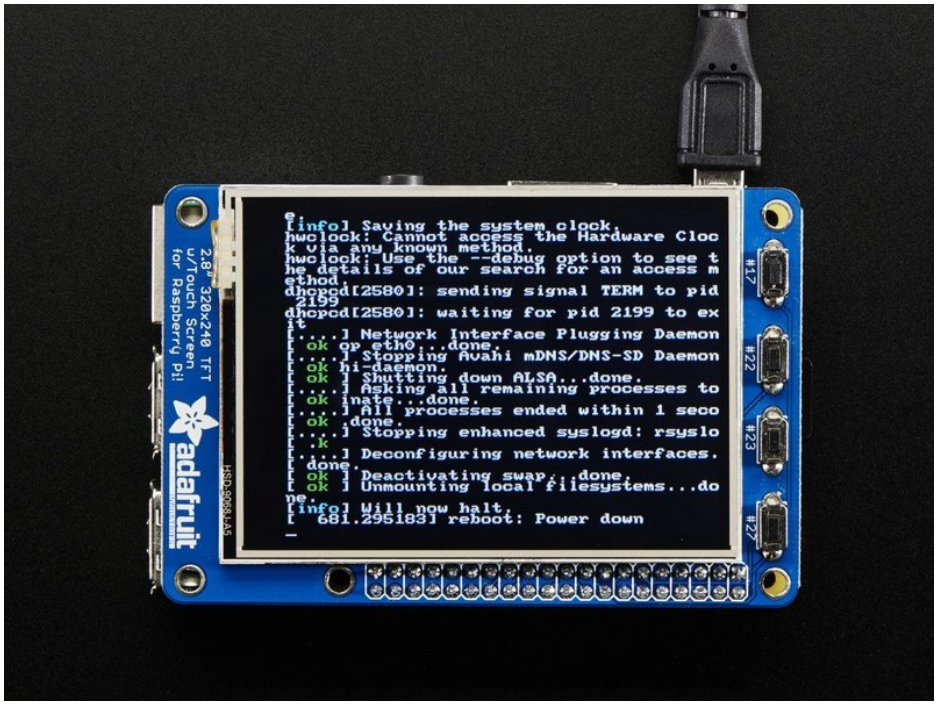
Original PiTFT

The original version PID 1601 is designed to fit nicely onto the Pi Model A or B but also works perfectly fine with the Pi Zero, Pi 2, Pi 3 or Pi 1 Model A+ or B+ as long as you don't mind the PCB overhangs the USB ports by 5mm



PiTFT Plus

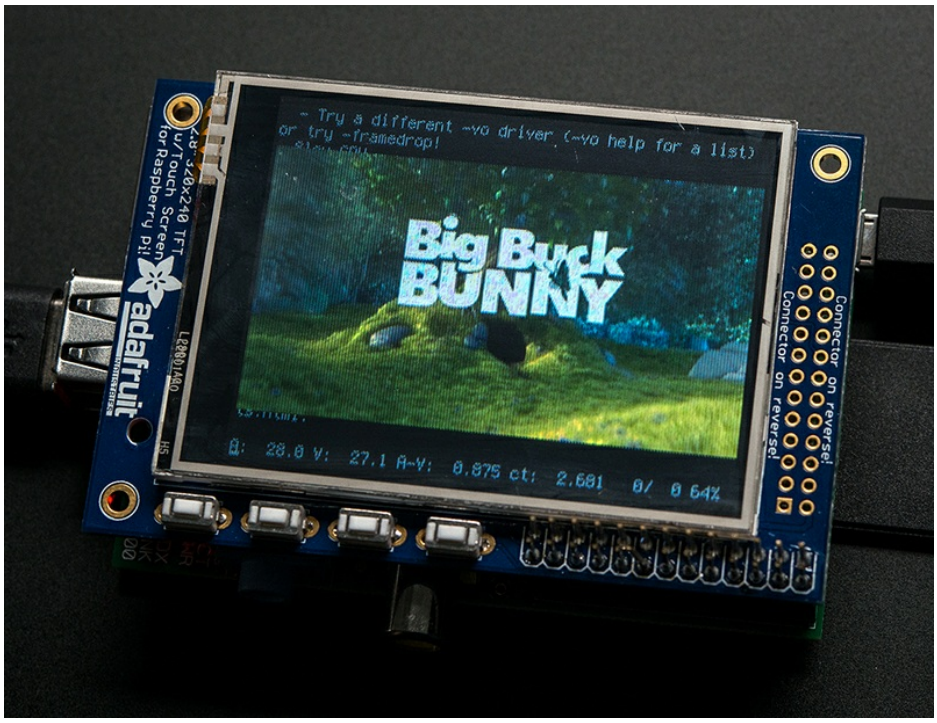
The newer PiTFTs are updated to fit perfectly onto the Pi Zero, Pi 3, Pi 2 or Model A+, B+! (Any Pi with a 2x20 connector) **Not for use with an old Pi 1 with 2x13 connector**



This design uses the hardware SPI pins (SCK, MOSI, MISO, CE0, CE1) as well as GPIO #25 and #24. All other GPIO are unused. Since we had a tiny bit of space, there's 4 spots for optional slim tactile switches wired to four GPIOs, that you can use if you want to make a basic user interface. For example, you can use one as a power on/off button.

We bring out GPIO #23, #22, #21, and #18 to the four switch locations!

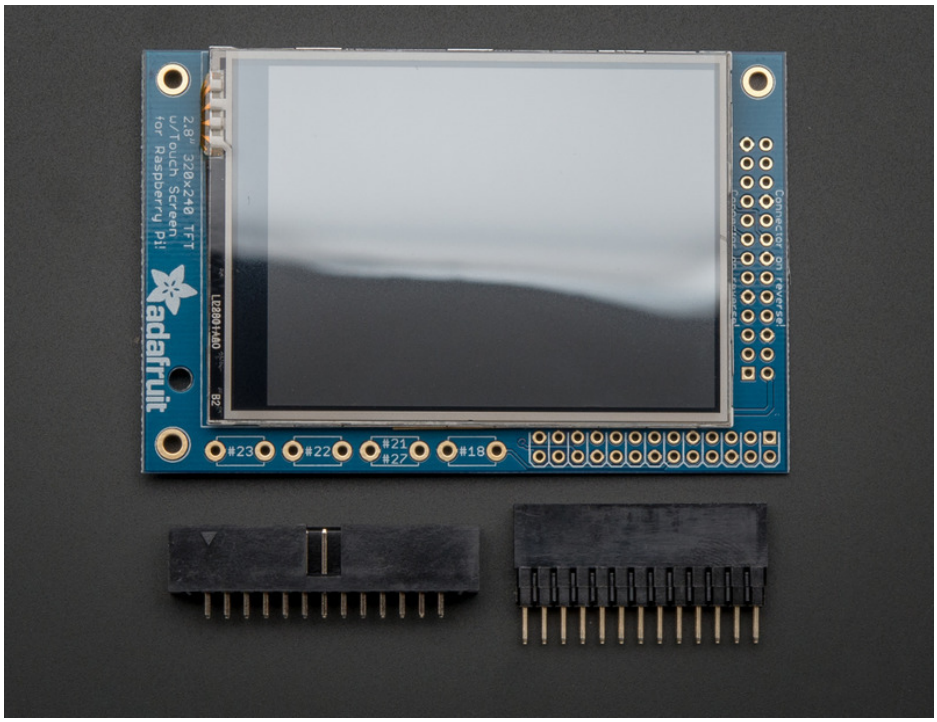
To make it super easy for use: we've created a custom kernel package based of off Notro's awesome framebuffer work, so you can install it over your existing Raspbian (or derivative) images in just a few commands.



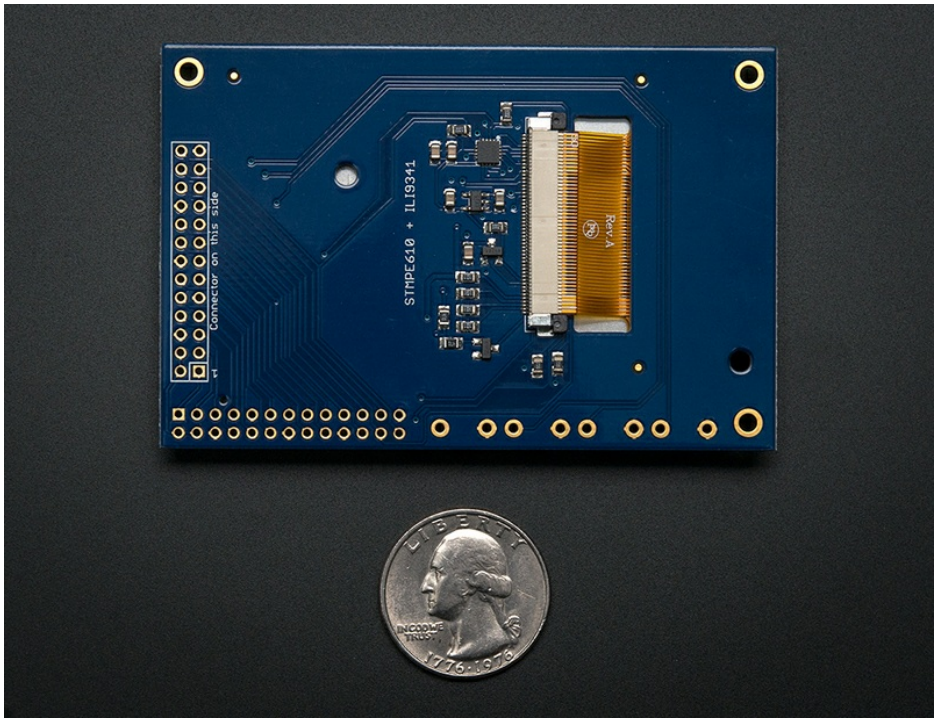
This tutorial series shows you how to install the software, as well as calibrate the touchscreen, splay videos, display images such as from your PiCam and more!

Assembly

This tutorial page is for PiTFT that came as a kit. If your PiTFT is already assembled, skip this step!



Before you start check that you have the parts you need: an assembled PiTFT plate with the 2.8" screen, extra tall female header and the 2x13 IDC socket. Note that it is normal for the screen to be 'loose' - this is so its easier for you to solder the connector on!

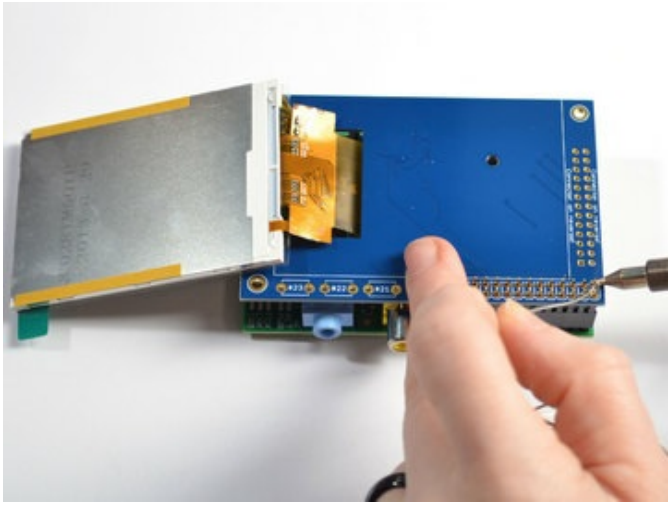


Check also on the back that the TFT is attached and that the flex connector is seated into the onboard FPC socket.

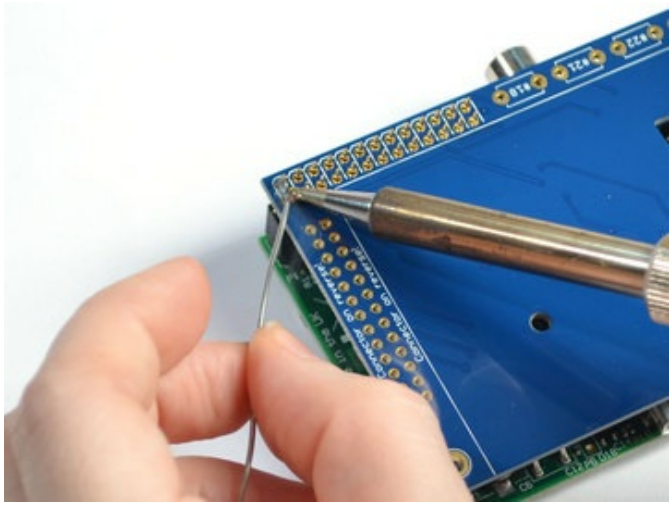
The easiest way to attach the header is if you have a Raspberry Pi as a 'stand' - make sure its powered off & unplugged!

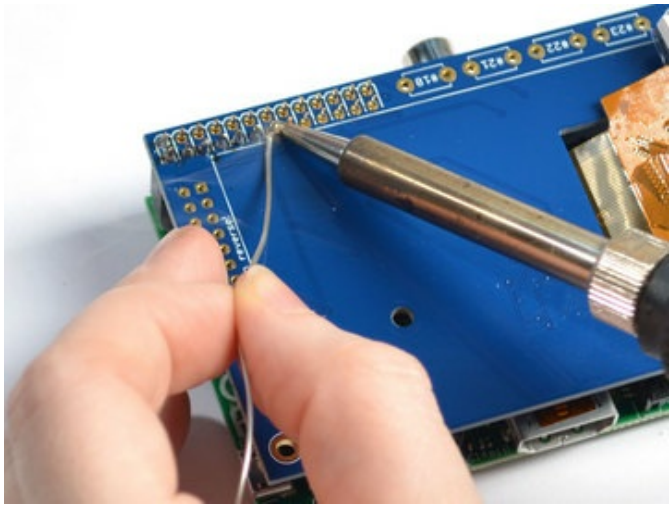
Plug the extra tall female header into the GPIO port on the Pi as shown. Make sure its seated nice and flat

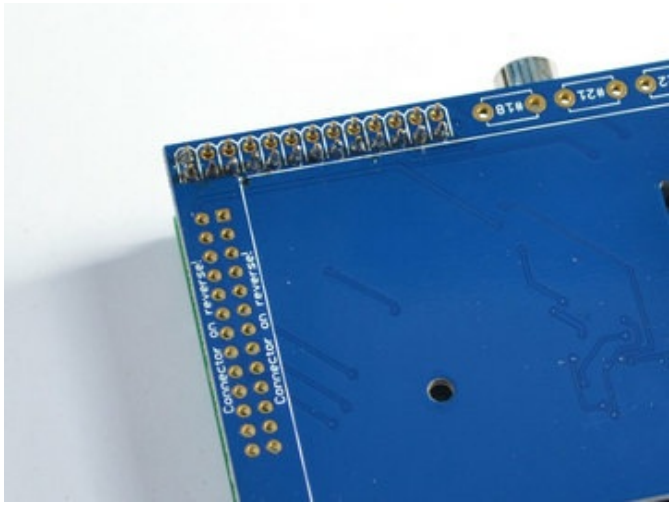
Place the PiTFT shield on top so all the pins stick through the connector on the side. Gently flip the TFT so its off to the side and wont be in your way while you solder



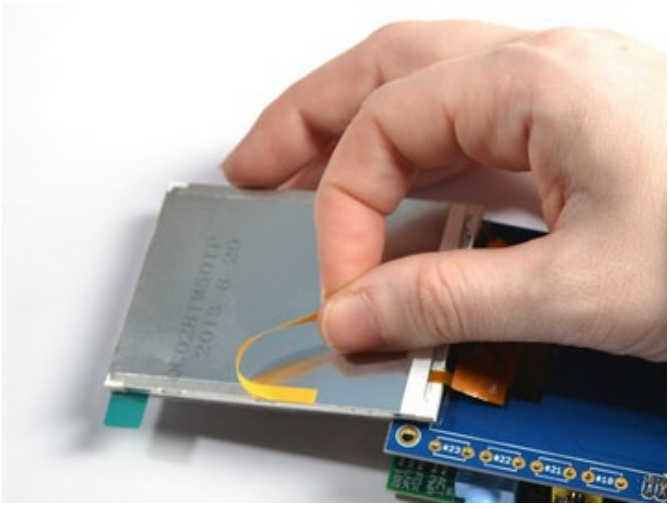
Heat up your soldering iron, and grab some solder. Start by tack-soldering one of the corners while pressing on the plate to make it sit flat. Once you have one or two pins done you can continue to solder each of the pins.





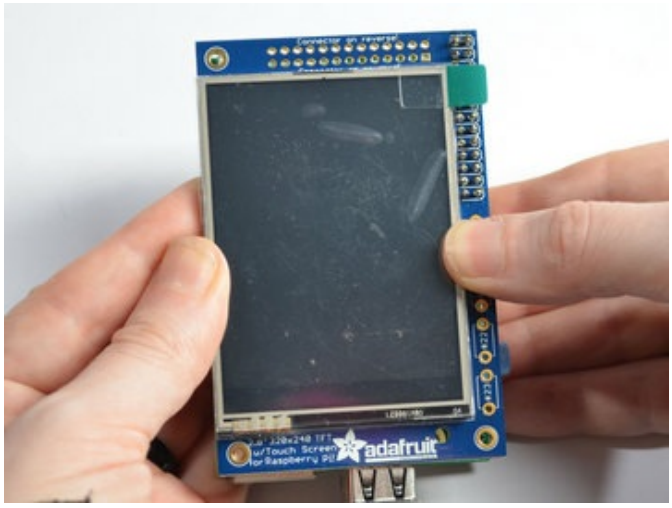

Connector on reverse!
Connector on reverse!

Before attaching the display, check that all the pins are soldered nicely and there's no bridging, cold solder, shorts, or unsoldered pins.

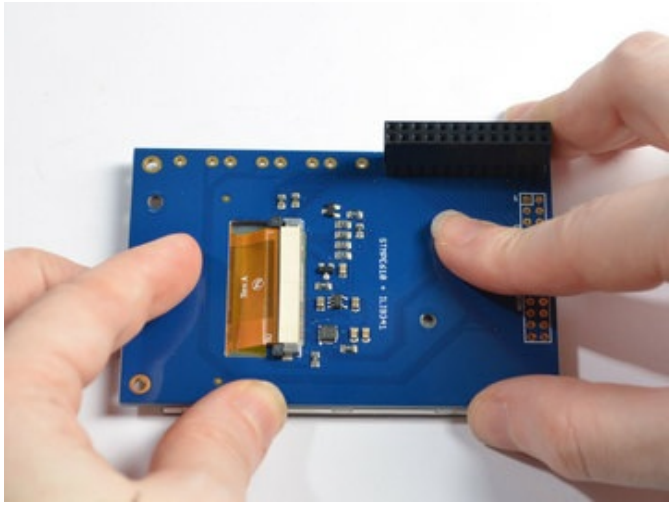


Now we can attach the screen. Remove the two thin tape cover strips.

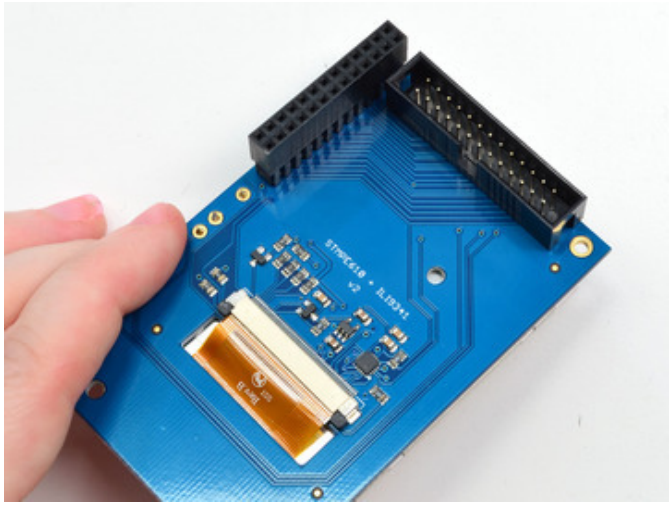
Line up the screen on the white outline, make sure there's some space from the header you just soldered in and the metal sides of the screen. As long as you don't really press down on the screen you can reposition it once or twice.



Once you have the screen so it is definitely not touching the header, you can gently press on the sides to secure the tape.

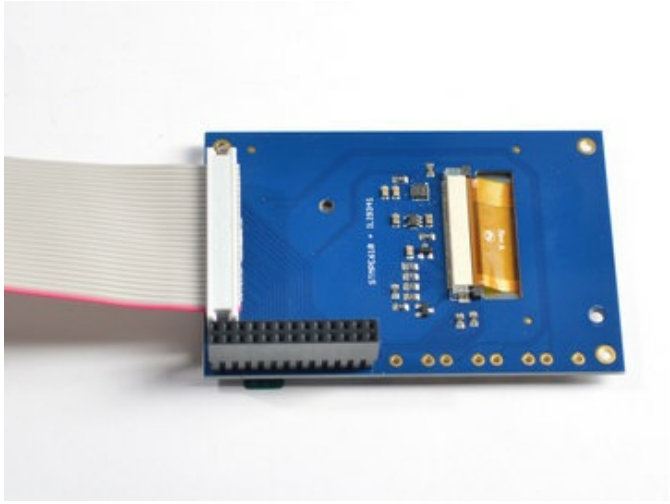


If the protective plastic cover is still on the screen you can press it against a clean table from above. That way you will really securely attach it!



If you want to attach an Adafruit Cobbler or similar, you can solder in the optional 2x13 IDC on the **bottom** of the screen as shown here. This will keep the top side clean and flat. Solder in all 26 pins

The picture shows a 2x13 male header. We've since updated this product to include an IDC socket so it's easier to add a cobbler. Both will work, though!



You can attach a 26-pin IDC cable just make sure the pin 1 indicator is on the right as indicated in this photo - there's also a #1 marking on the PCB!

The PiTFT requires some device tree support and a couple other things to make it a nice stand-alone display. If you just want to get going, check out the following for easy-install instructions!

The same installer is used for all PiTFTs, you will pick and configure the setup during installation!

Install Raspbian on an SD Card

You'll need to start with Raspbian or Raspbian Lite.

The last known for-sure tested-and-working version is March 13, 2018 (https://downloads.raspberrypi.org/raspbian/images/raspbian-2018-03-14/ (https://adafru.it/F2K)) from https://downloads.raspberrypi.org/raspbian/images/ (https://adafru.it/BFU)

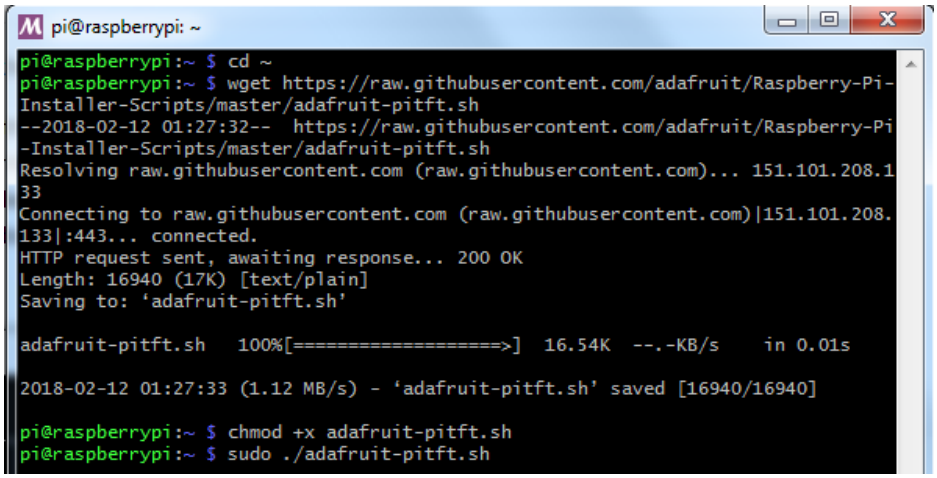
Raspbian does often 'break' stuff when new versions come out so to be safe, if you are having problems try this version!

Installer script

This script will do all the work for you, and install both device tree overlay support as well as configure rotation and any HDMI mirroring. PiTFT no longer needs any custom kernels or modules, so you can continue to update/upgrade your Pi and it will work with the most recent releases.

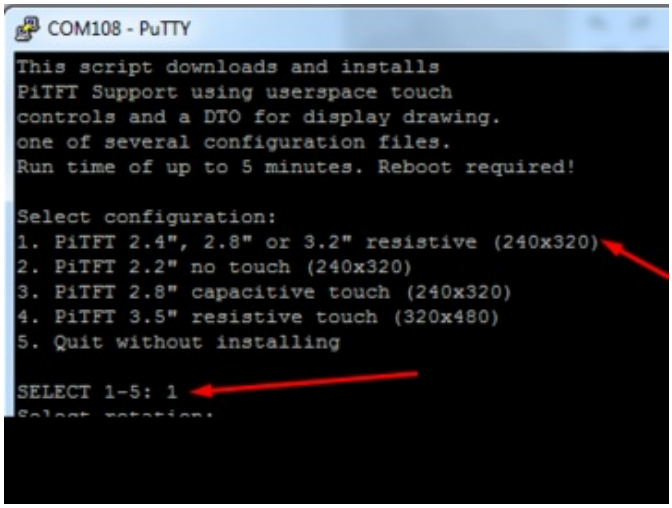
Here's the commands to run. Make sure your Pi has network access, it needs to download the software!

```
cd ~
wget https://raw.githubusercontent.com/adafruit/Raspberry-Pi-Installer-Scripts/master/adafruit-pitft.sh
chmod +x adafruit-pitft.sh
sudo ./adafruit-pitft.sh
```

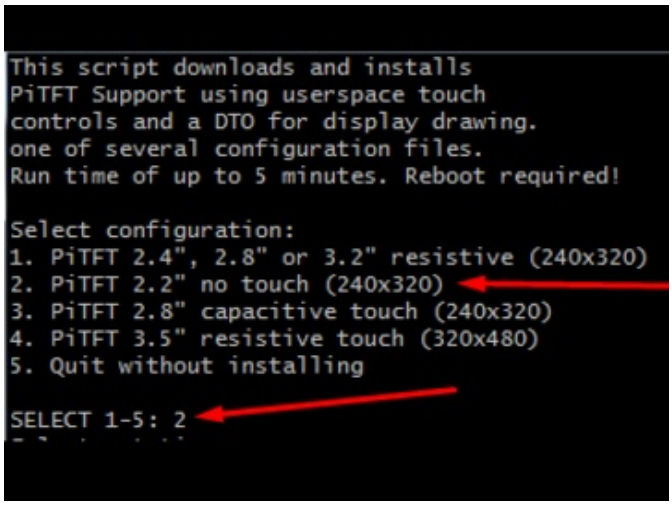


PiTFT Selection

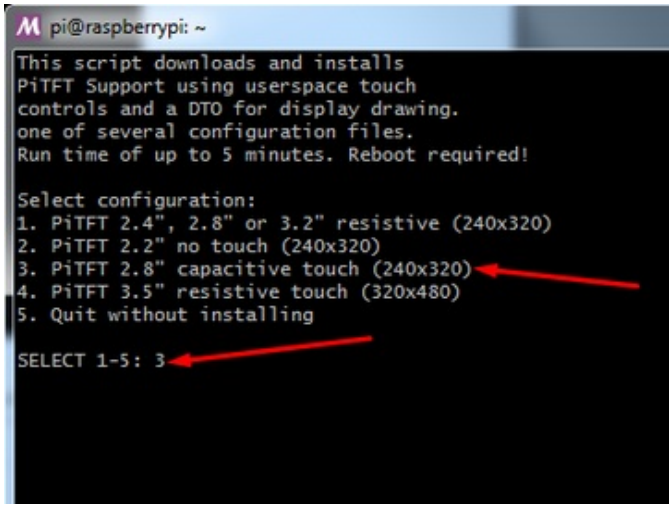
Once you run it you will be presented with menus for configuration.



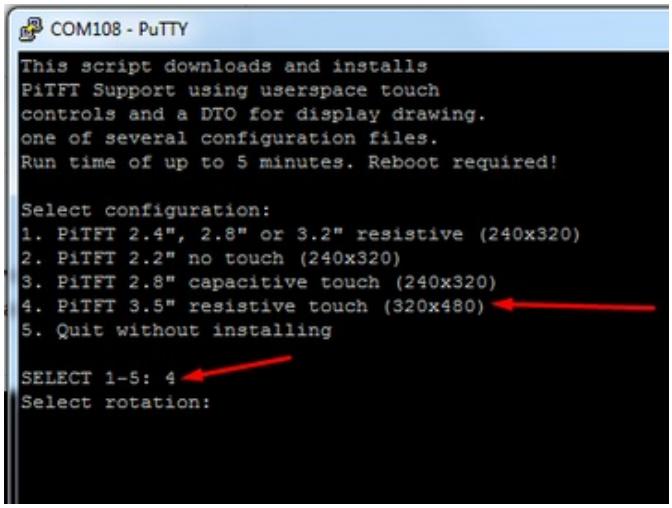
For the 2.4", 2.8" and 3.2" PiTFT with resistive touchscreen overlay select **#1**



For the 2.2" PiTFT select **#2**



For the 2.8" Capacitive PiTFT select **#3**



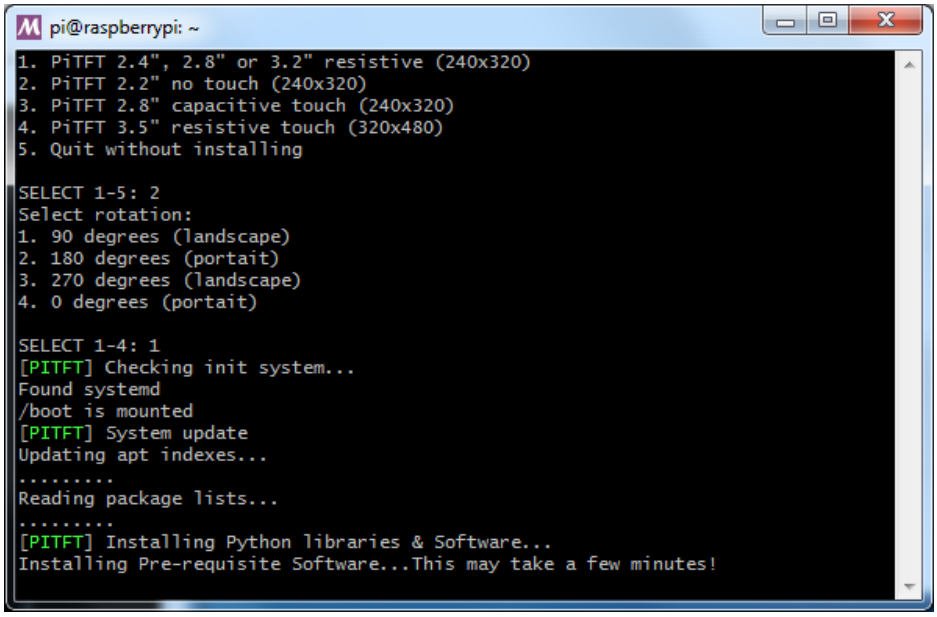
For the 3.5" PiTFT select **#4**

Rotation

Next you will be asked for the rotation you want, don't worry if you're not 100% sure which you want, you can always change this later by re-running the script

```
SELECT 1-5: 2
Select rotation:
1. 90 degrees (landscape)
2. 180 degrees (portait)
3. 270 degrees (landscape)
4. 0 degrees (portait)

SELECT 1-4: 1
```

It will take a few minutes to install the software and download all the things...



Configuring what shows where

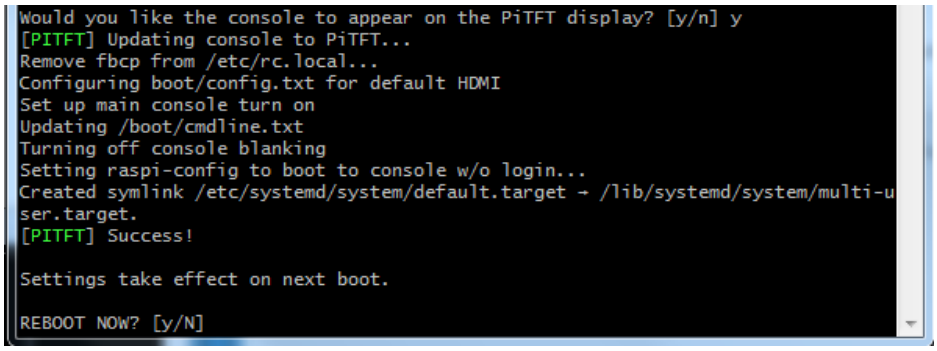
You have a few different ways to set up the PiTFT, we ask **2** questions to figure out what you want

PiTFT as Text Console (best for Raspbian 'Lite')

This is the simplest to set-up type of use. Its great if you have a simple text based or pygame/SDL based interface. If you want the PiTFT to act as a text console you can expect:

- HDMI will be 'deactivated' - nothing appears on the HDMI output but a black screen
- The login prompt appears on the Pi
- The Pi is all text, not a GUI (no PIXEL desktop)
- Keyboard and mouse are used only by the PiTFT interface
- Framebuffer-capable software (such as **fbi** for displaying images, **mplayer** for videos, or pygame software, etc) appear on the PiTFT
- OpenGL accelerated software *will not appear on the PiTFT* (it is unaccelerated framebuffer only)
- But, non-OpenGL-accelerated graphics software is a bit faster than using HDMI mirroring (not tons faster but you're not running **fbcp** which will always make it faster)

If you want that say **Yes** to the question `Would you like the console to appear on the PiTFT display`



Then simply reboot. Once rebooted you will not see anything on HDMI, but the console will appear on the PiTFT. That's it!

PiTFT as HDMI Mirror (Best for Raspbian Full/PIXEL)

This option is the easiest to understand: whatever appears on the HDMI display will be 'mirrored' to the PiTFT. Note that HDMI is much higher resolution so it's not like it turns the PiTFT into a 1080p display. This is great for when you want to run OpenGL-optimized software, PIXEL desktop software, or really anything. The down-side is its a little slower than drawing directly to the framebuffer. You may not notice it but it's worth us mentioning!

- HDMI will be 'activated' but at a lower resolution - you can change this later but it looks best at 320x240 (PiTFT 2.2", 2.4", 2.8" and 3.2") or 480x320 (PiTFT 3.5")
- The login prompt or GUI appears on both HDMI and PiTFT at the same time
- Keyboard and mouse are shared, since the display is mirrored
- All graphics appear on both HDMI and PiTFT, thanks to **fbcp**

If you want that say Yes to the question `Would you like the HDMI display to mirror to the PiTFT display?`

PiTFT as Raw Framebuffer Device

For advanced users who are comfortable using framebuffer devices, it is possible to have the PiTFT and HDMI graphics be *both* active and display different data.

- HDMI will be active and act like a normal Pi
- The login prompt or GUI (PIXEL) appears on the HDMI
- PiTFT appears black, nothing appears on it
- Keyboard and mouse are used by the HDMI interface but can, in theory, be captured and used to change graphics on PiTFT through programming
- Framebuffer-capable software (such as **fbi** for displaying images, **mplayer** for videos, or pygame software, etc) *can* appear on the PiTFT if you set it up to display to **/dev/fb1**
- OpenGL accelerated software *will never appear on the PiTFT* (it is unaccelerated framebuffer only)

If you want that, say **No** to both of the configuration questions!

You can always change your mind after setting up one of the configurations, depending on your needs! Just re-run the script

Unsupported Full Images

Historically, we provided full 'images' of Raspbian. This worked OK until Raspbian started doing releases every few

months. These are no longer supported, and won't even boot on Pi 3B+, so we recommend the script above.

There's the larger 'classic Jessie' image that will boot into X by default, and requires a 8G image, it has a lot more software installed. There's also the smaller 'Jessie Lite' that will boot into the command line, and can be burned onto a 2G card! Click below to download and install into a new SD card. Unzip and follow the classic SD card burning tutorials (https://adafru.it/aMW)

PiTFT 2.2" Images

- Raspbian Jessie 2016/10/23-based image (https://adafru.it/sbg)
- Raspbian Jessie Lite 2016/10/23-based image (https://adafru.it/sbh)
- Raspbian Jessie 2016/03/25-based image (https://adafru.it/mAe)
- Raspbian Jessie Lite 2016/03/25-based image (https://adafru.it/mAf)
- Raspbian Jessie 2015/09/24-based image (https://adafru.it/iDC)
- Raspbian Wheezy 2015/09/09-based image (https://adafru.it/idt)

PiTFT 2.4"/2.8"/3.2" Resistive Images

- Raspbian Jessie 2016/9/23-based image (https://adafru.it/s7f)
- Raspbian Jessie Lite 2016/9/23-based image (https://adafru.it/s7A)
- Raspbian Jessie 2016/03/25-based image (https://adafru.it/mA9)
- Raspbian Jessie Lite 2016/03/25-based image (https://adafru.it/mAa)
- Raspbian Jessie 2015/09/24-based image (https://adafru.it/iDA)
- Raspbian Wheezy 2015/09/09-based image (https://adafru.it/idJ)
- Raspbian 2014/06/20-based image (https://adafru.it/dSM)
- Raspbian 2014/09/09-based image (https://adafru.it/e12)

PiTFT 2.8" Capacitive

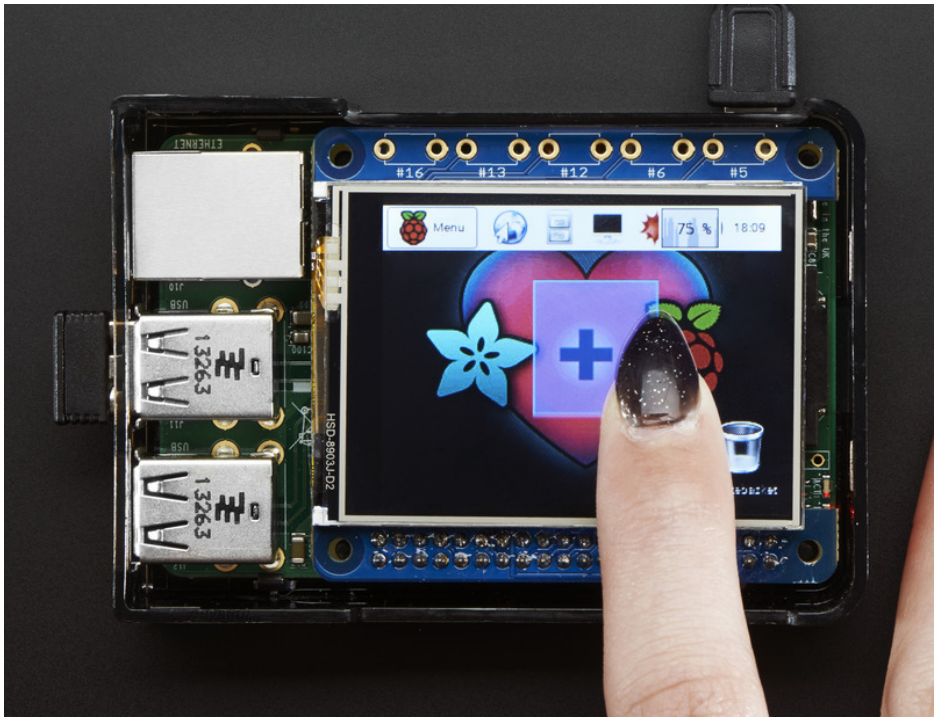
- Raspbian Jessie 2016-09-23-based image (https://adafru.it/saM)
- Raspbian Jessie Lite 2016-09-23-based image (https://adafru.it/saN)
- Raspbian Jessie 2016-03-25-based image (https://adafru.it/mAc)
- Raspbian Jessie Lite 2016-03-25-based image (https://adafru.it/mAd)
- Raspbian Jessie 2015/09/24-based image (https://adafru.it/iDy)
- Raspbian Wheezy 2015/09/24-based image (https://adafru.it/idz)
- Raspbian 2014/09/18-based image (https://adafru.it/e11)
- Raspbian 2014/06/20-based image (https://adafru.it/dSO)
- Raspbian image from 2015/03/03 (https://adafru.it/eUI)

PiTFT 3.5" Images

- Raspbian Jessie 2016/9/23-based image (https://adafru.it/siF)
- Raspbian Jessie Lite 2016/9/23-based image (https://adafru.it/sja)
- Raspbian Jessie 2016/03/25-based image (https://adafru.it/mAb)
- Raspbian Jessie 2016/03/25-based image (https://adafru.it/mAG)
- Raspbian Jessie 2015/09/24-based image (https://adafru.it/iDD)
- Raspbian Wheezy 2015/09/24-based image (https://adafru.it/idy)
- Raspbian 2014/09/09-based image (https://adafru.it/e10)
- Raspbian 2015/03/12 image (https://adafru.it/eUE)

If you've grabbed our Easy Install image, or used the installer script, this step is not required, it's already done! This is just for advanced users who are curious on how to configure and customize the touchscreen

This procedure is identical for the 2.4", 2.8", 3.2" and 3.5" Resistive PiTFTs. Not for use with the Capacitive PiTFT!



Setting up the Touchscreen

Now that the screen is working nicely, we'll take care of the touchscreen. There's just a bit of calibration to do, but it isn't hard at all.

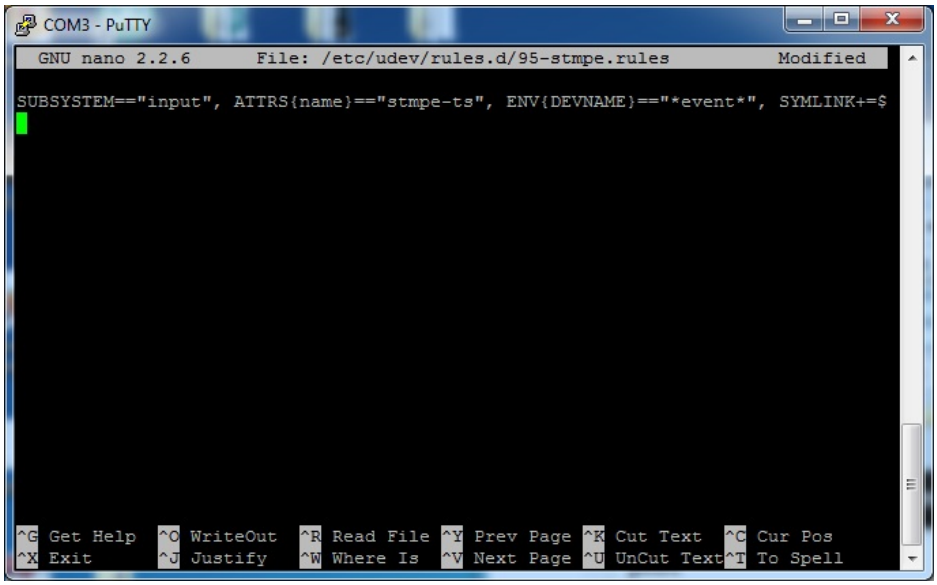
Before we start, we'll make a **udev** rule for the touchscreen. That's because the **eventX** name of the device will change a lot and its annoying to figure out what its called depending on whether you have a keyboard or other mouse installed.

Run

```
sudo nano /etc/udev/rules.d/95-stmpe.rules
```

to create a new **udev** file and copy & paste the following line in:

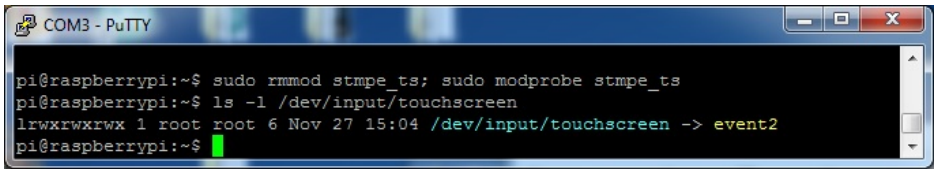
```
SUBSYSTEM=="input", ATTRS{name}=="stmpe-ts", ENV{DEVNAME}=="*event*", SYMLINK+="input/touchscreen"
```



Remove and re-install the touchscreen with

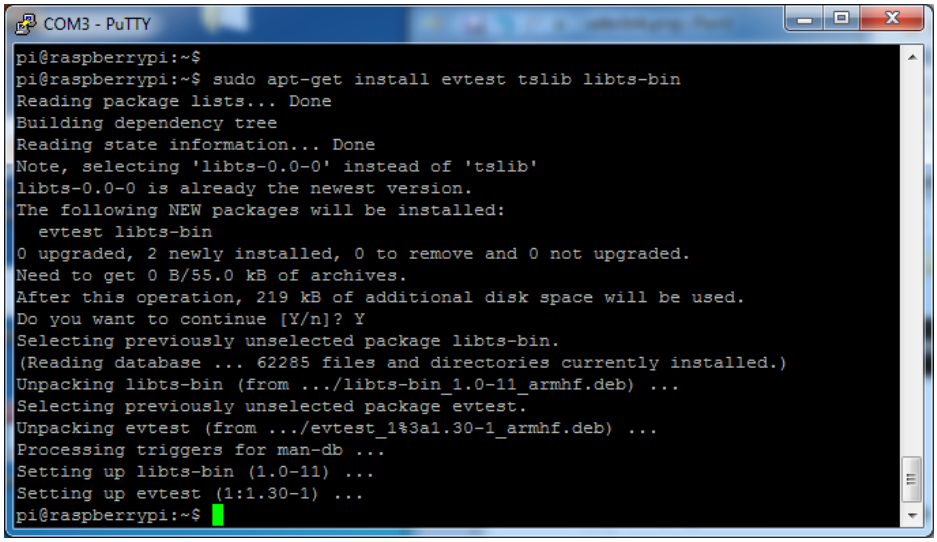
```
sudo rmmod stmpe_ts; sudo modprobe stmpe_ts
```

Then type **ls -l /dev/input/touchscreen**
It should point to **eventX** where X is some number, that number will be different on different setups since other keyboards/mice/USB devices will take up an event slot



There are some tools we can use to calibrate & debug the touchscreen. Install the "event test" and "touchscreen library" packages with

```
sudo apt-get install evtest tslib libts-bin
```

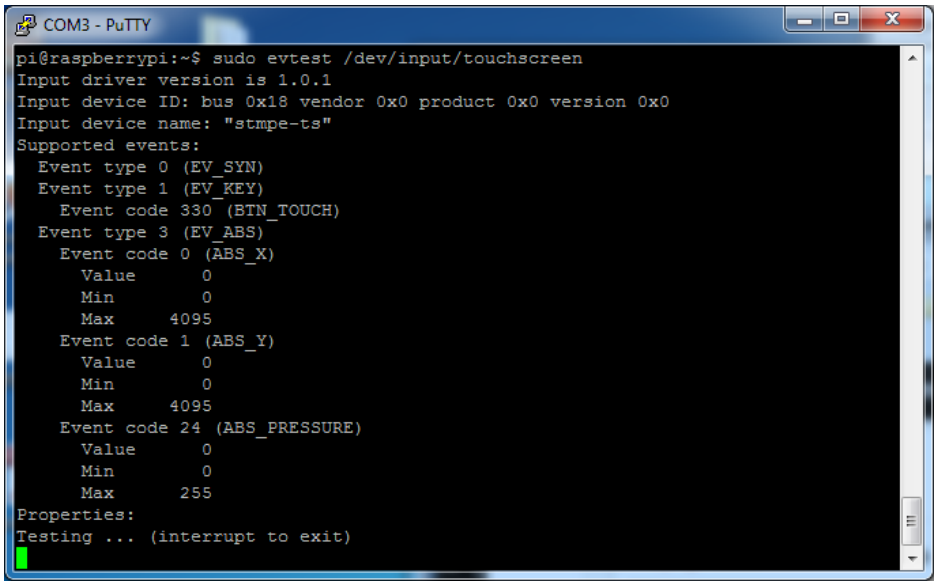


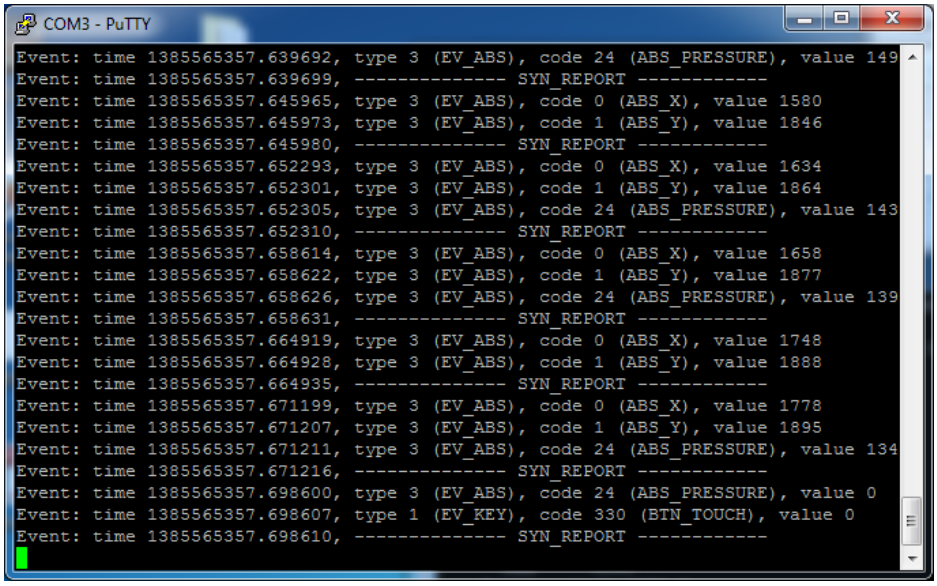
Running evtest

Now you can use some tools such as

```
sudo evtest /dev/input/touchscreen
```

which will let you see touchscreen events in real time, press on the touchscreen to see the reports.





AutoMagic Calibration Script

If you rotate the display you need to recalibrate the touchscreen to work with the new screen orientation. You can manually run the calibration processes in the next section, or you can re-run the installer script and select a new rotation:

Try using this default calibration script to easily calibrate your touchscreen display. Note that the calibration values might not be exactly right for your display, but they should be close enough for most needs. If you need the most accurate touchscreen calibration, follow the steps in the next section to manually calibrate the touchscreen.

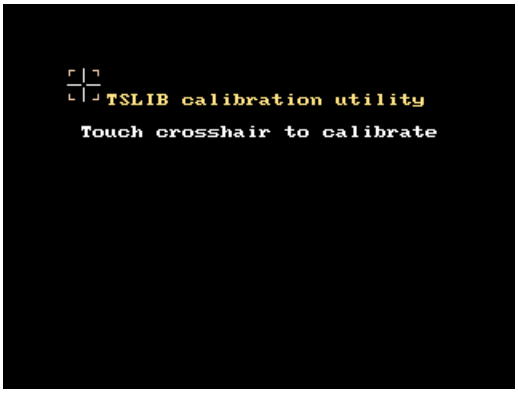
Manual Calibration

If the "automagic" calibration technique isn't working for you, or you have some other setup where you need to carefully calibrate you can do it 'manually'

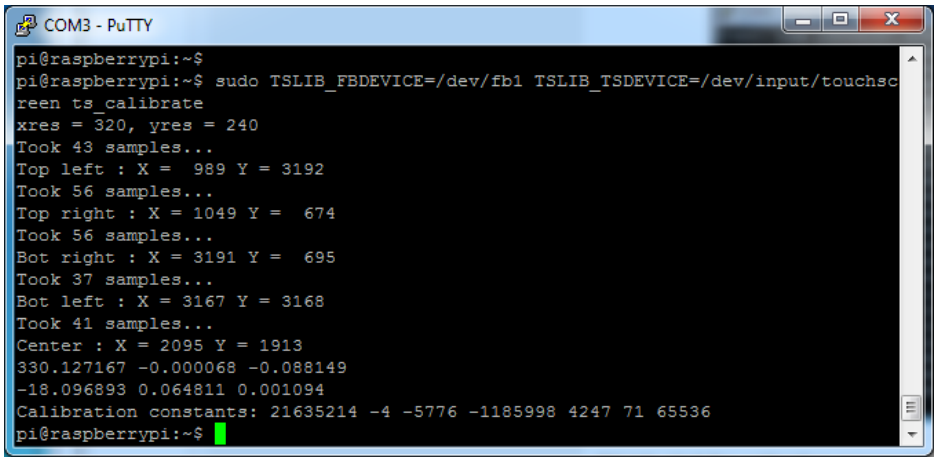
You will want to calibrate the screen once but shouldn't have to do it more than that. We'll begin by calibrating on the command line by running

```
sudo TSLIB_FBDEVICE=/dev/fb1 TSLIB_TSDEVICE=/dev/input/touchscreen ts_calibrate
```

follow the directions on the screen, touching each point. Using a stylus is suggested so you get a precise touch. Don't use something metal, plastic only!



You should see five crosshair targets. If you see less than that, the touchscreen probably generated multiple signals for a single touch, and you should try calibrating again.



Next you can run

```
sudo TSLIB_FBDEVICE=/dev/fb1 TSLIB_TSDEVICE=/dev/input/touchscreen ts_test
```

which will let you draw-test the touch screen. Go back and re-calibrate if you feel the screen isn't precise enough!

X Calibration

You can also calibrate the X input system but you have to use a different program called **xtcal** (xinput_calibrator no longer works)

You can do this if the calibration on the screen isn't to your liking or any time you change the **rotate=XX** module settings for the screen. Since the screen and touch driver are completely separated, the touchscreen doesn't auto-rotate

Download and compile it with the following:

```
sudo apt-get install libxaw7-dev libxxf86vm-dev libxaw7-dev libxft-dev
git clone https://github.com/KurtJacobson/xtcal
cd xtcal
make
```

You must be running PIXEL (the GUI) while calibrating.

Before you start the calibrator you will need to 'reset' the old calibration data so run

```
DISPLAY=:0.0 xinput set-prop "stmpe-ts" 'Coordinate Transformation Matrix' 1 0 0 0 1 0 0 0 1
```

Now you'll have to run the calibrator while also running X. You can do this by opening up the terminal program and running the the **xtcal** command (which is challenging to do on such a small screen) OR you can do what we do which is create an SSH/Terminal shell and then run the calibrator from the same shell, which requires the following command:

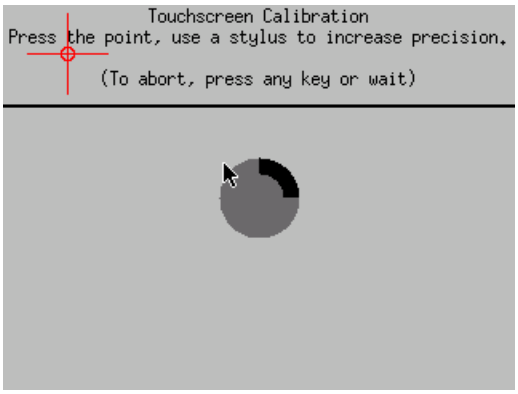
```
DISPLAY=:0.0 xtcal/xtcal -geometry 640x480
```

Note that the `geometry` may vary!

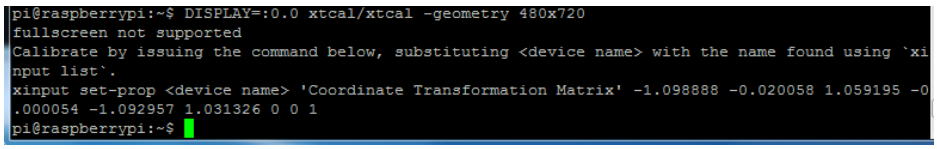
If you are using a 2.4"/2.8"/3.2" 320x240 display with landscape orientation, use 640x480. If you're in portrait, use 480x640.

If you are using a 3.5" display with landscape, use 720x480, portrait is 480x720

Follow the directions on screen



Once complete you'll get something like:



Run **sudo nano /usr/share/X11/xorg.conf.d/20-calibration.conf** and copy the 9 numbers into the TransformationMatrix option so it looks like:

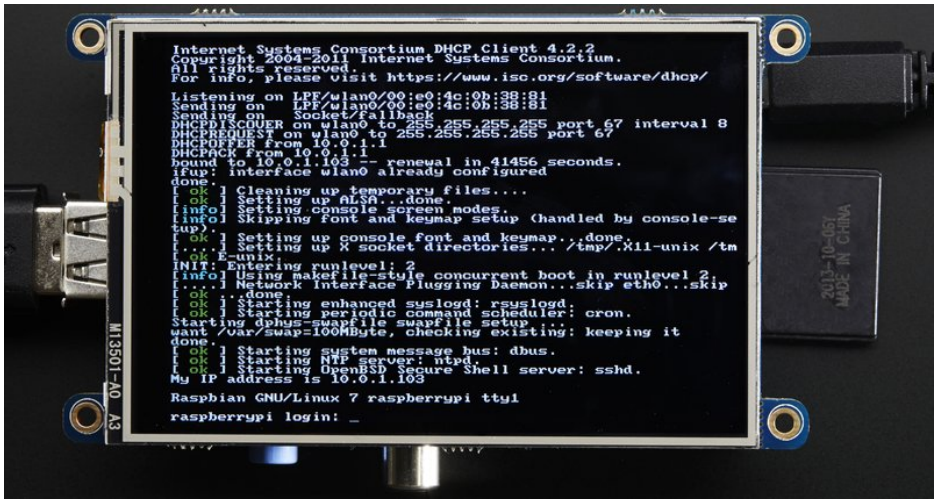
```
Section "InputClass"
        Identifier "STMPE Touchscreen Calibration"
        MatchProduct "stmpe"
        MatchDevicePath "/dev/input/event*"
        Driver "libinput"
        Option "TransformationMatrix" "-0.000087 1.094214 -0.028826 -1.091711 -0.004364 1.057821 0 0 1"
EndSection
```

or whatever you got, into there.

You will want to reboot your Pi to verify you're done

Your touchscreen is now super calibrated, hurrah!

If you've used our installer script, this step is not required, it's already done! This is just for advanced users who are curious on how to configure and customize the console



One fun thing you can do with the display is have it as your main console instead of the HDMI/TV output. Even though it is small, with a good font you can get 20 x 40 of text. For more details, check out https://github.com/notro/fbtft/wiki/Boot-console (https://adafru.it/cXQ)

First up, we'll update the boot configuration file to use the TFT framebuffer **/dev/fb1** instead of the HDMI/TV framebuffer /dev/fb0

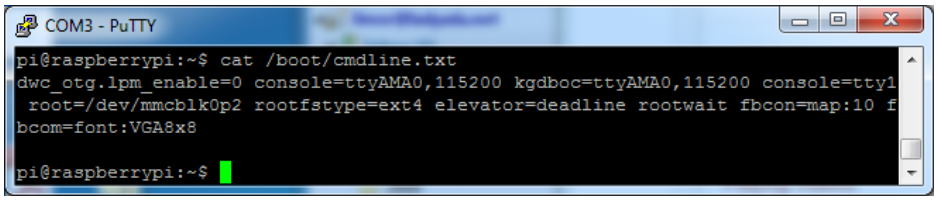
```
sudo nano /boot/cmdline.txt
```

you can also edit it by putting the SD card into a computer and opening the same file.

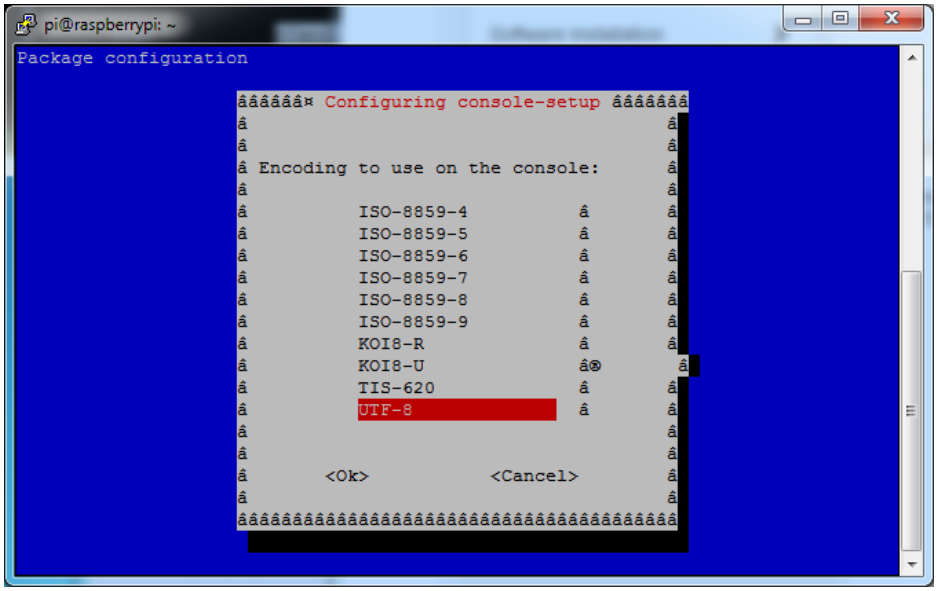
At the end of the line, find the text that says `rootwait` and right after that, enter in: `fbcon=map:10 fbcon=font:VGA8x8` then save the file.

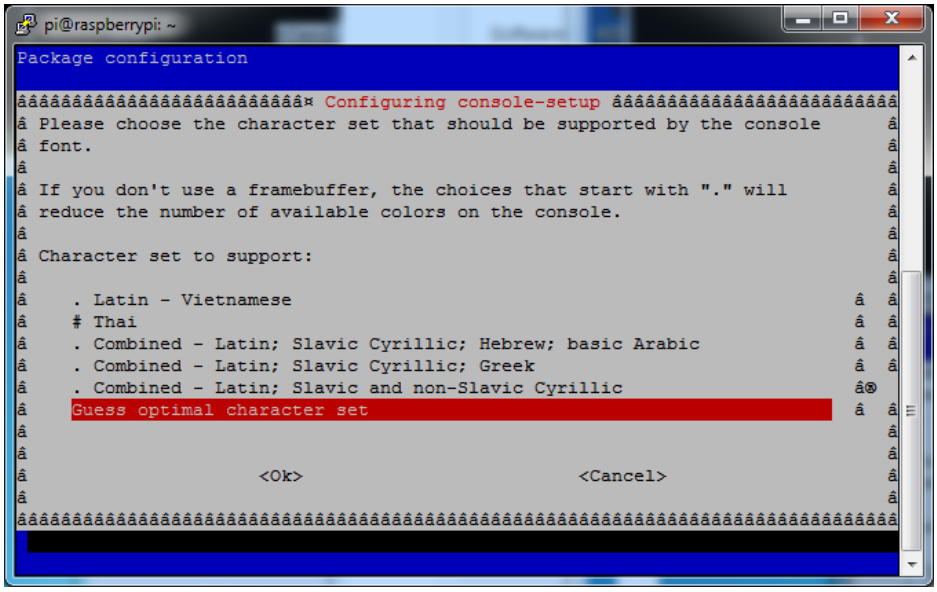
On the next boot, it will bring up the console.

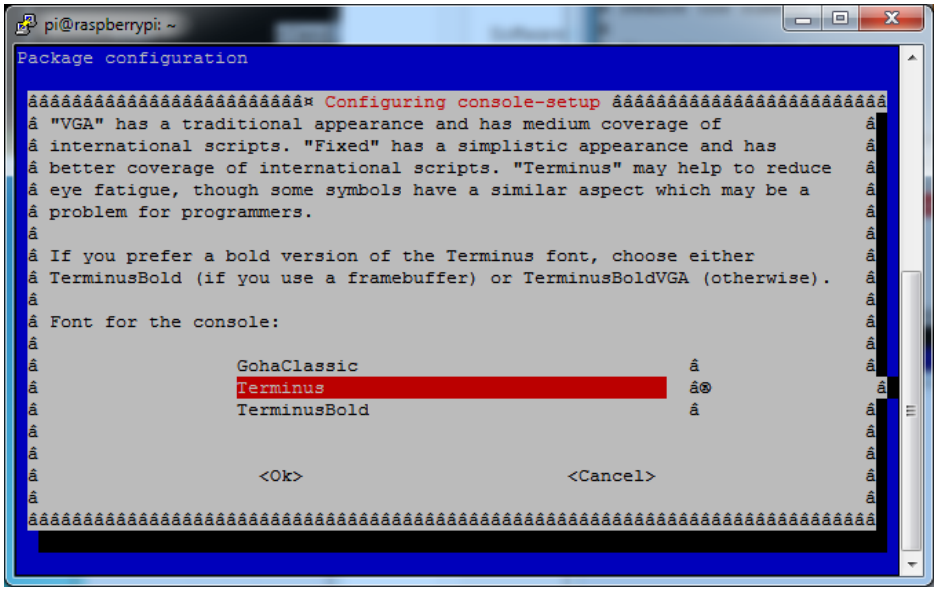
Note that the kernel has to load up the display driver module before it can display anything on it so you won't get the rainbow screen, a NooBs prompt, or a big chunk of the kernel details since the module is loaded fairly late in the boot process.

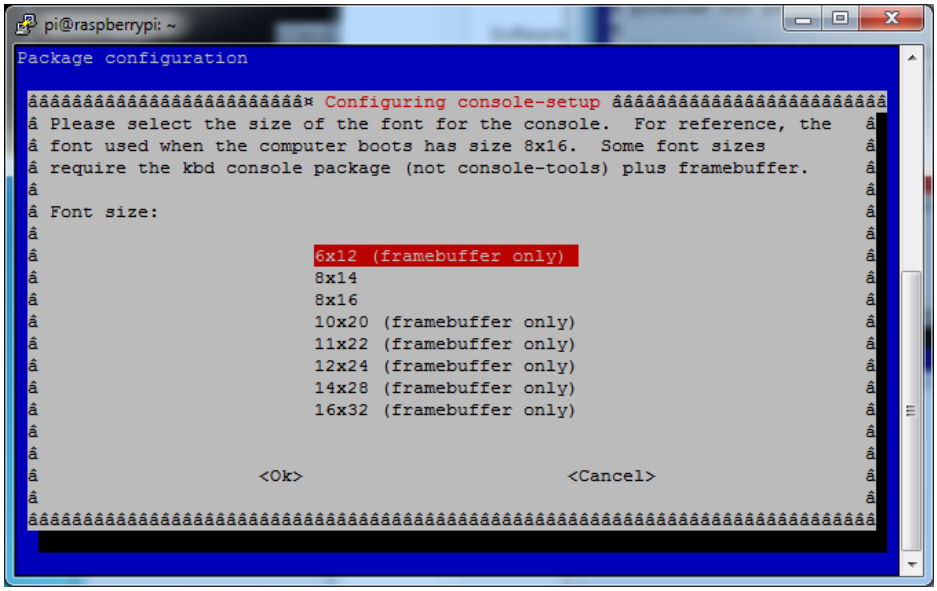


I think the VGA8x8 font is a bit chunky, you probably want 12x6 which is what is shown in the photo above. To change the font, run **sudo dpkg-reconfigure console-setup** and go thru to select Terminus 6x12


pi@raspberrypi: ~
Package configuration
Configuring console-setup
Encoding to use on the console:
ISO-8859-4
ISO-8859-5
ISO-8859-6
ISO-8859-7
ISO-8859-8
ISO-8859-9
KOI8-R
KOI8-U
TIS-620
UTF-8
<Ok>
<Cancel>


pi@raspberrypi: ~
Package configuration
Configuring console-setup
Please choose the character set that should be supported by the console font.
If you don't use a framebuffer, the choices that start with "." will reduce the number of available colors on the console.
Character set to support:
. Latin - Vietnamese
Thai
. Combined - Latin; Slavic Cyrillic; Hebrew; basic Arabic
. Combined - Latin; Slavic Cyrillic; Greek
. Combined - Latin; Slavic and non-Slavic Cyrillic
Guess optimal character set
<Ok>
<Cancel>





Turn off Console Blanking

You may notice the console goes black after 30 minutes, this is a sort of 'power saving' or 'screensaver' feature.

Raspbian Jessie

Add the following line to /etc/rc.local

```
sudo sh -c "TERM=linux setterm -blank 0 >/dev/tty0"
```

on the line before the final `exit 0`

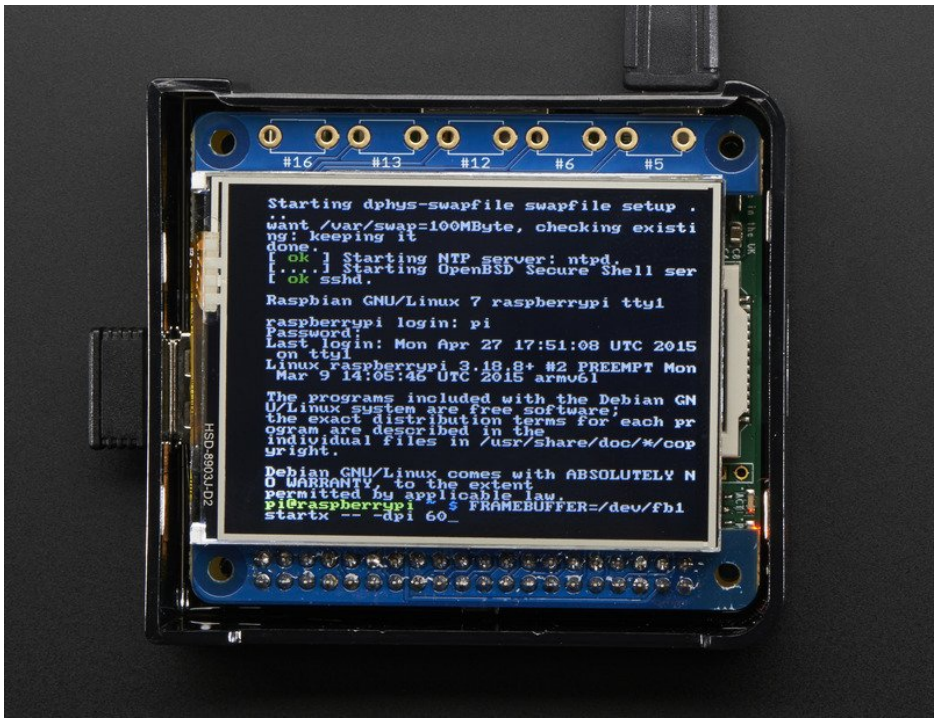
Raspbian Wheezy

You can disable this by editing **/etc/kbd/config** and looking for

`BLANK_TIME=30`

and setting the blank time to 0 (which turns it off)

```
BLANK_TIME=0
```



Userspace Tools

Sometimes the PiTFT device tree and related kernel package don't work across different OS releases, so we've experimented with an alternate approach that **doesn't rely on a custom kernel** — it instead works in "**user space**." So far it's worked well regardless of the OS version being used!

There are tradeoffs. The code is still in a rough state with many features yet to be implemented, and also the performance is slightly less than the kernel approach. It's typically adequate though, even for game emulation (RetroPie, etc.), so give it a try if you've had trouble with the "classic" approach.

This currently requires a bit of Linux-y knowledge, editing files and such...

Download, Test and Install

PiTFT displays use **SPI** to communicate, so make sure that's enabled using the **raspi-config** utility:

```
sudo raspi-config
```

Menu options move around from time to time...at the time of this writing, SPI is under "Interfacing Options."

Then retrieve the software using *wget...*

```
wget https://github.com/adafruit/Adafruit_Userspace_PiTFT/archive/master.zip
unzip master.zip
```

And then a quick test...

```
cd Adafruit_Userspace_PiTFT-master
sudo ./tftcp
```

The PiTFT should mirror the contents of the Raspberry Pi's HDMI output at this point. Text and everything will be microscopic, but we're just checking that the program runs. If not, confirm that the file **/dev/spidev0.0** exists — this should happen when SPI is enabled. Double-check raspi-config and it never hurts to reboot.

Does it run? Good. Press control+c to kill the program, and we'll set it up to run automatically on boot.

First, copy the **tftcp** executable to **/usr/local/bin**:

```
sudo cp tftcp /usr/local/bin
```

Then edit the file **/etc/rc.local** as **root** (you can substitute your editor of preference for nano):

```
sudo nano /etc/rc.local
```

Just above the final "exit 0" line, insert the following line:

```
/usr/local/bin/tftcp &
```

The screen looks best if the HDMI resolution exactly matches the PiTFT resolution, so the final step is to configure the system for 320x240 video:

```
sudo nano /boot/config.txt
```

Append the following lines to the bottom of the file:

```
disable_overscan=1
hdmi_force_hotplug=1
hdmi_group=2
hdmi_mode=87
hdmi_cvt=320 240 60 1 0 0 0
```

OPTIONAL: you can also use “640 480” in place of “320 240” above. This is exactly twice the PiTFT native resolution, and the tftcp utility will perform a smooth 2:1 filtering of the image. Any larger though and the image isn’t as sharp (and text becomes tiny, like when we first tested it).

Now **reboot** and the PiTFT should activate toward the end of the boot process.

Resistive Touchscreen Support

This is even more experimental than the **tftcp** utility...it only works with the resistive screen, and there’s no calibration support yet, but if you’d like to try it out...

First there’s some prerequisite software to install:

```
sudo apt-get update
sudo apt-get install python-pip python-smbus python-dev
sudo pip install evdev
```

“cd” to the same directory where the software was downloaded earlier, and try it out...

```
cd Adafruit_Userspace_PiTFT-master
sudo python touchmouse.py
```

Whether you’re in X11 or in text console mode (e.g. Raspbian Lite), the cursor should move in response to touch, which is emulating a mouse.

If that seems OK, press control+c to stop it and we’ll use the same steps to make it auto-run on boot:

```
sudo cp touchmouse.py /usr/local/bin
sudo nano /etc/rc.local
```

Insert this line just above the “exit 0” at the end of the file:

```
/usr/bin/python /usr/local/bin/touchmouse.py &
```

reboot and both PiTFT and touch should be active now.

HELP! (FAQ)

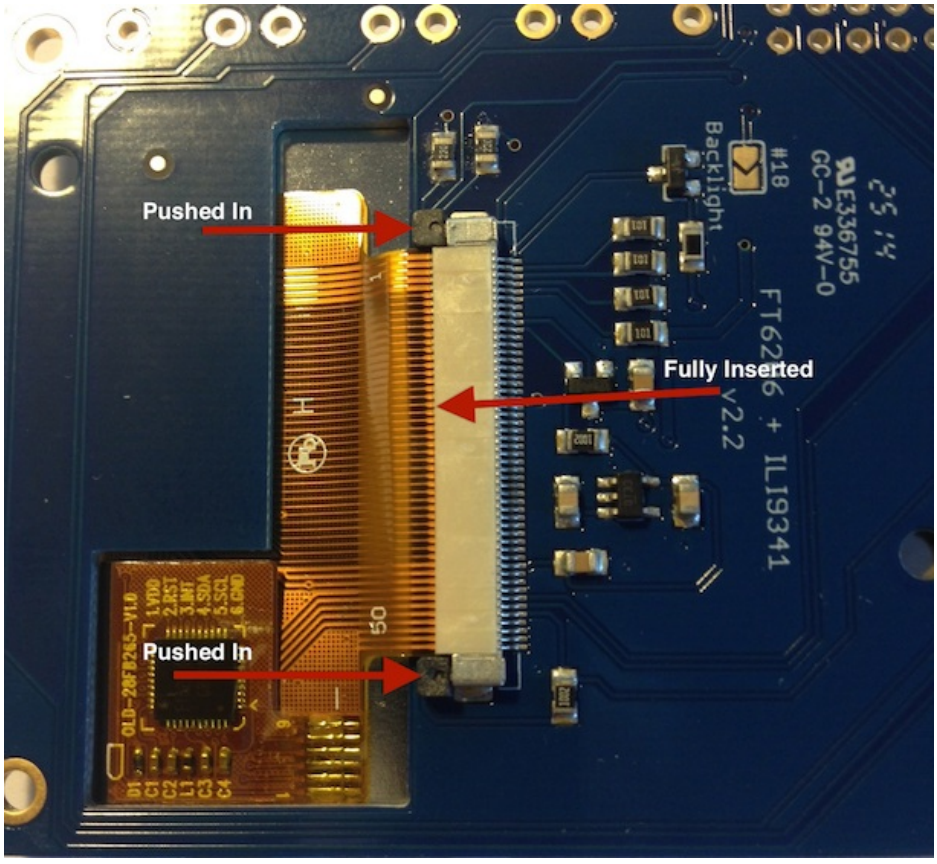
My PiTFT used to work, now it doesn't!

If you messed with `/boot/config.txt` or `/etc/rc.local` you may have removed or disabled some of the elements required for the PiTFT to work. Try re-running the Easy Installer script!

I'm booting my Pi with the PiTFT and the HDMI output 'locks up' during boot!

It looks like the Pi is 'halting' or 'locking' up during boot but what is really happening is the console is switching from the HDMI output to the PiTFT console output.

Check your PiTFT connections, particularly make sure you seated the PiTFT on the Pi properly, nothing is in the way, and the TFT flex connector is seated properly.



My PiTFT works for a bit and then I get a black screen with a short line of white pixels in one corner

Sounds like you tried to configure your Pi to 'boot straight to X', that is, start up the graphics interface on boot. This doesn't work by default because the Pi operating system is not expecting a PiTFT so it boots to the HDMI output. See below for how to set up your Pi to boot to X on the PiTFT

To 'fix' this, you can either connect an HDMI monitor, then in a terminal window run **sudo raspi-config** and configure the Pi to boot to the command line not X! If you do not have an HDMI monitor, you can also try a console cable

I'm tring to run startx and I get FATAL: Module g2d_23 not found.

don't forget you have to remove the turbo file!
sudo mv /usr/share/X11/xorg.conf.d/99-fbturbo.conf ~

How come OMX-Player and Minecraft and other programs don't appear on the PiTFT display?

Some programs are graphics-optimized, particularly the video playback tools and some other programs like Minecraft. They write 'directly' to the HDMI output, and cannot write to the PiTFT so there is no way to directly make them work. However, you *can* have the output go to HDMI and then mirror the HDMI onto the PiTFT with **fbcp**. Using the Easy Installer, select **Mirror HDMI**

Why doesn't the tactile button on GPIO #21 work?

On some older PiTFTs we had one of the buttons labeled #21 - that's the original RasPi name for that pin. If you're using a V2 (chance is, you are!) that is now called #27.
All the PiTFT's we ship now have the button labeled #21 and #27

I want better performance and faster updates!

You can change the SPI frequency (overclock the display) by editing **/boot/config.txt** and changing the **dtoverlay** options line to:

```
dtoverlay=pitft28r,rotate=90,speed=62000000,fps=25
```

Or whatever you like for speed, rotation, and frames-per-second. BUT, here's the thing, the Pi only supports a *fixed number* of SPI frequencies. So tweaking the number a little won't do anything. The kernel will round the number to

the closest value. You will always get frequencies that are 250MHz divided by an even number. Here's the only SPI frequencies this kernel supports

- 15,625,000 (a.k.a 16000000 = 16 MHz)
- 17,857,142 (a.k.a. 18000000 = 18 MHz)
- 20,833,333 (a.k.a 21000000 = 21 MHz)
- 25,000,000 (= 25 MHz)
- 31,250,000 (a.k.a 32000000 = 32MHz)
- 41,666,666 (a.k.a 42000000 = 42MHz)
- 62,500,000 (a.k.a 62000000 = 62MHz)

So if you put in 48000000 for the speed, you won't actually get 48MHz, you'll actually only get about 42MHz because it gets rounded down. We tested this display nicely with 32MHz and we suggest that. But you can put in 42MHz or even try 62MHz and it will update faster

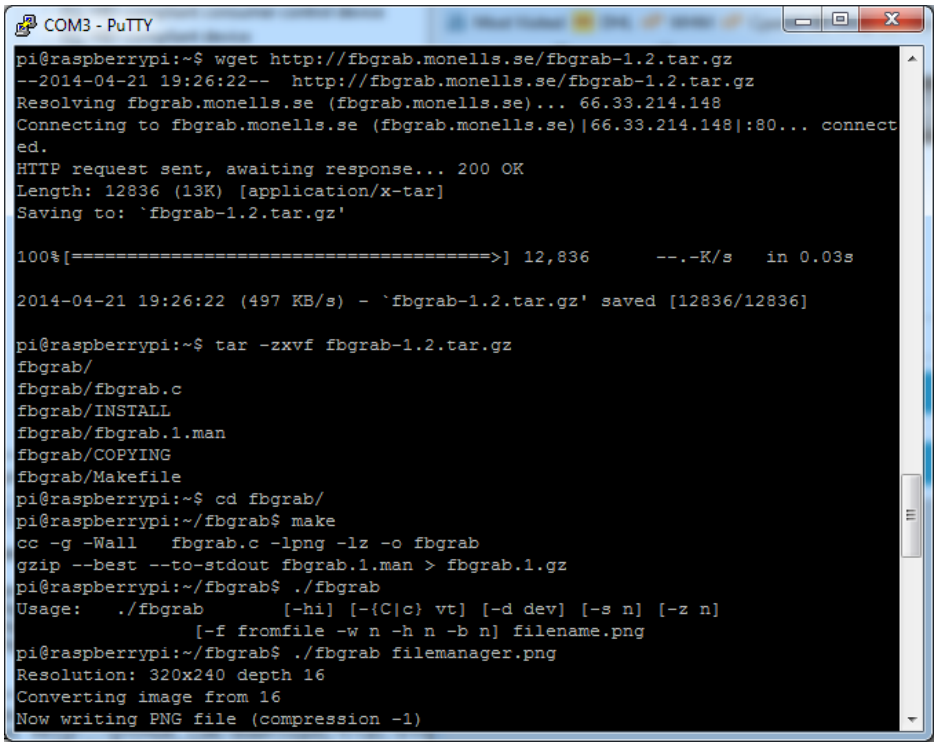
You can tweak fps (frames per second) from 20 to 60 and frequency up to 62MHz for tradeoffs in performance and speed. Reboot after each edit to make sure the settings are loaded properly. There's a trade off that if you ask for higher FPS you're going to load the kernel more because it's trying to keep the display updated.

How can I take screenshots of the little screen?

We took the screenshots for this tutorial with **fbgrab**

```
wget http://fbgrab.monells.se/fbgrab-1.2.tar.gz
tar -zxvf fbgrab*gz
cd fbgrab/
make
```

```
./fbgrab screenshot.png
```

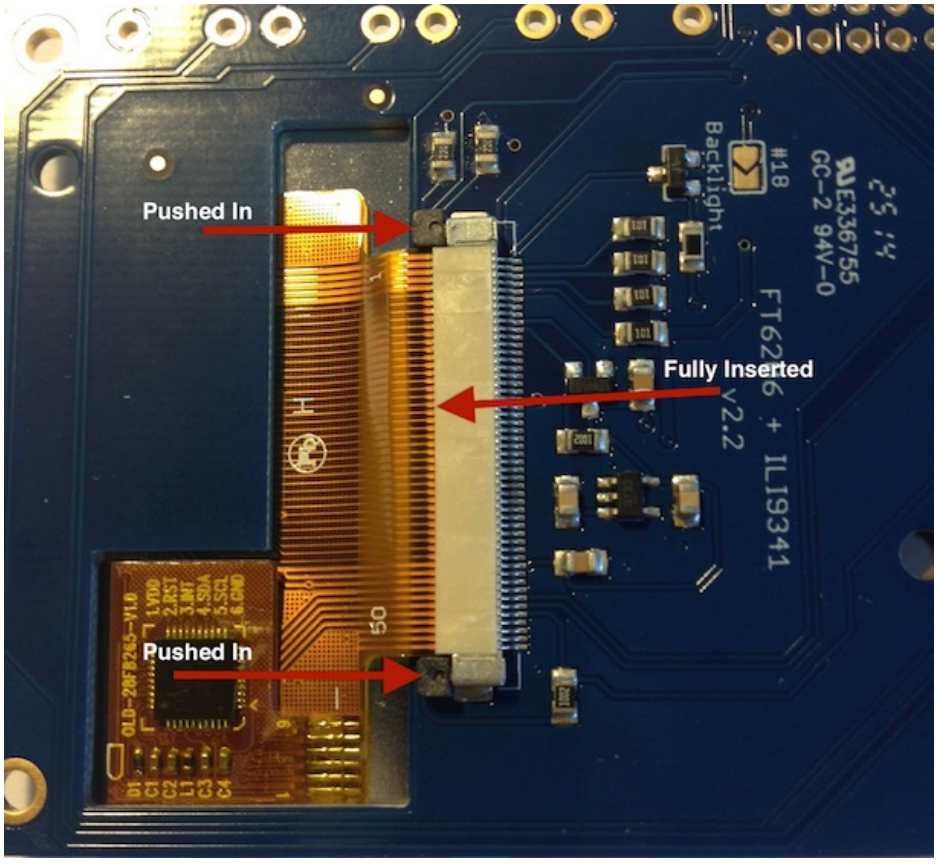


How do I automatically boot to X windows on the PiTFT?

Make sure your Pi boots to the graphical PIXEL desktop on the HDMI output monitor, then using the Easy Installer, select **Mirror HDMI**

My screen isn't working/works erratically/looks funny

Check to make syre that the flat flex cable is fully seated in the connetor and the 'ears' are pushed in to secure it. See the picture for what it should look like:


Pushed In
Fully Inserted
Pushed In
Backlight
#18
E336755
GC-2 94V-0
FT6206 + ILI9341
v2.2
50

On my first run of startx I get a window saying "GDBus Error.org.Freedesktop Policy Kit1 Error: Failed Cannot determine user of subject"

This happens on the Raspberry Pi the first time you run startx, no matter what display. You can just re-start X and it wont appear again.

Can I get a right-click from the touch-screen?

Yes! Please see this post:

https://forums.adafruit.com/viewtopic.php?f=47&t=77528&p=393280#p393322

I'm having difficulties with the STMPE resistive touch screen controller

Here's a hack for the device tree overlay that can force different SPI modes, sometimes that helps!

My PiTFT's rotation/calibration isn't working in X11

X11 (the graphical system) has changed how it gets touchscreen input, so if you rotate the display and the calibration isn't being picked up:

Check /usr/share/X11/xorg.conf.d for a file called 10-evdev.conf

If you don't see that file

1. You need to `sudo apt-get install xserver-xorg-input-evdev`, and then...
2. If you do have a 40-libinput.conf in that same directory, you must remove it even if/once evdev is installed, since it will override the 10-evdev.conf otherwise.

Thanks to cerebrate in the forums for the hint!

Playing Videos

How To Play Videos

You can play many types of videos on the screen, using mplayer you don't even need to run X and you can script the movies to play using Python. We'll show you how to just play one video for now.
To demo, we'll use an mp4 of Big Buck Bunny for 320 pixel wide screens. Below we show you how to create/resize videos, but to make it easy, just download our version with:

```
wget http://adafruit-download.s3.amazonaws.com/bigbuckbunny320p.mp4 (https://adafru.it/cXR)
```

The video is 30MB which is a lot if you haven't expanded your SD card yet. Before you do this, run sudo raspi-config to expand the SD card so you don't run out of space!

If you don't have **mplayer** yet, run

```
sudo apt-get update
```

```
sudo apt-get install mplayer
```

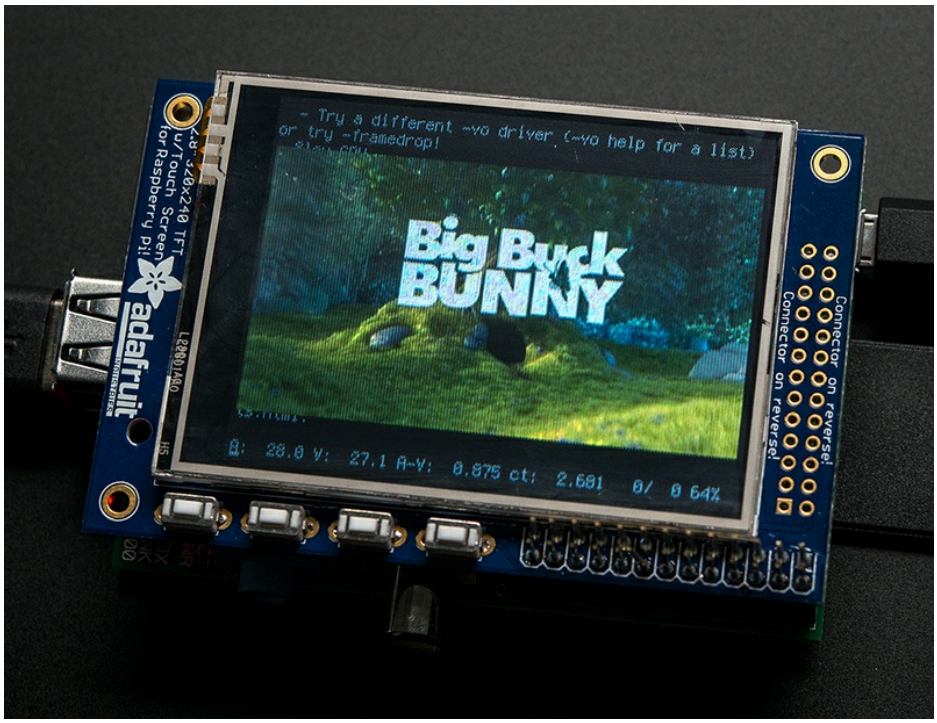
to install it. It may take a few minutes to complete

```
pi@raspberrypi: ~
pi@raspberrypi ~ $ sudo apt-get install mplayer
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  esound-common libaa1 libaudiofile1 libavcodec53 libavformat53 libavutil51
  libcdparanoia0 libdca0 libdirac-encoder0 libdvdnav4 libdvdread4 libenca0
  libesd0 libfaad2 libfribidi0 libgpm2 libgsm1 libjack-jackd2-0 liblircclient0
  liblzo2-2 libmp3lame0 libmpeg2-4 libopenal-data libopenal1 libpostproc52
  libschroedinger-1.0-0 libspeex1 libswscale2 libtheora0 libva1 libvpx1
  libx264-123 libxvidcore4 libxvmc1
Suggested packages:
  libdvdcss2 pulseaudio-esound-compat gpm jackd2 lirc libportaudio2
  libroar-compat2 speex mplayer-doc netselect fping
The following NEW packages will be installed:
  esound-common libaa1 libaudiofile1 libavcodec53 libavformat53 libavutil51
  libcdparanoia0 libdca0 libdirac-encoder0 libdvdnav4 libdvdread4 libenca0
  libesd0 libfaad2 libfribidi0 libgpm2 libgsm1 libjack-jackd2-0 liblircclient0
  liblzo2-2 libmp3lame0 libmpeg2-4 libopenal-data libopenal1 libpostproc52
  libschroedinger-1.0-0 libspeex1 libswscale2 libtheora0 libva1 libvpx1
  libx264-123 libxvidcore4 libxvmc1 mplayer
0 upgraded, 35 newly installed, 0 to remove and 52 not upgraded.
Need to get 9,296 kB of archives.
After this operation, 20.6 MB of additional disk space will be used.
Do you want to continue [Y/n]?
```

OK now you just have to run:

```
sudo SDL_VIDEODRIVER=fbcon SDL_FBDEV=/dev/fb1 mplayer -vo sdl -framedrop bigbuckbunny320p.mp4
```

If your video is not sized for 320 wide, you may need to add a `-zoom` after `-framedrop` so that it will resize - note that this is quite taxing for the Pi, so it may result in a choppy or mis-synced video!

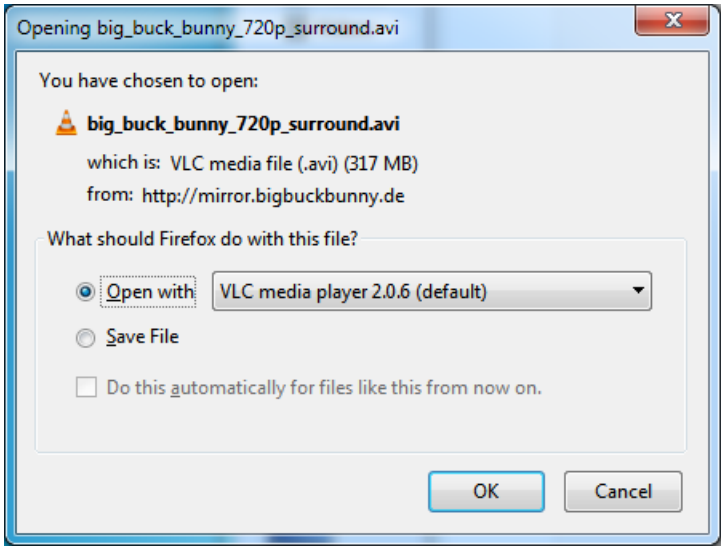


Converting/Resizing Videos

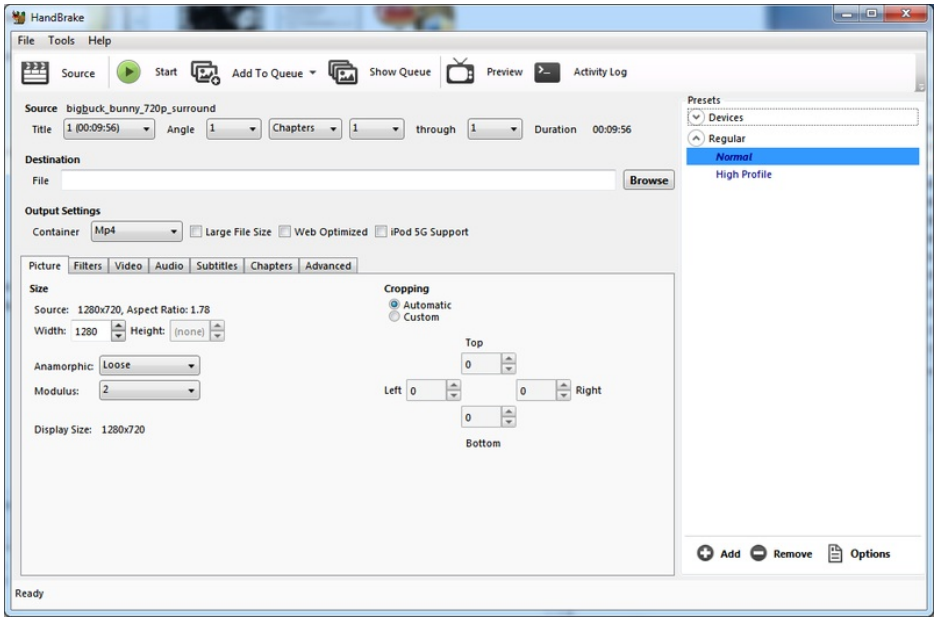
It's possible to play full length videos on the TFT plate, but since the screen is small and the Pi cant use hardware accelleration to play the videos its best to scale them down to 320x240 pixels. This will be easier for the Pi to play and also save you tons of storage space. For this demo, we'll be using the famous Big Buck Bunny (https://adafru.it/cXS)

video, which is creative commons and also very funny!

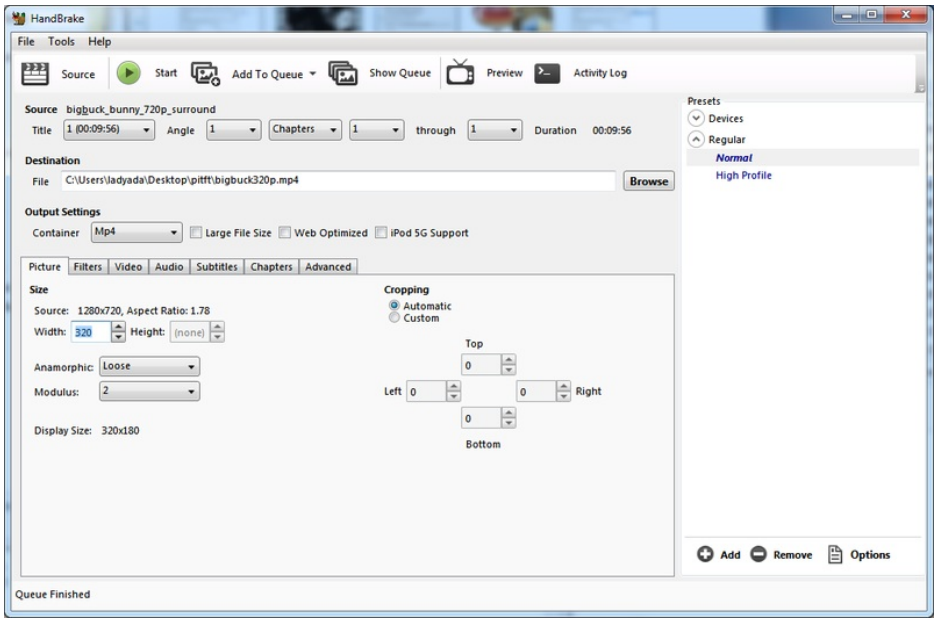
You can download it from the link above, we'll be using the 720p AVI version.



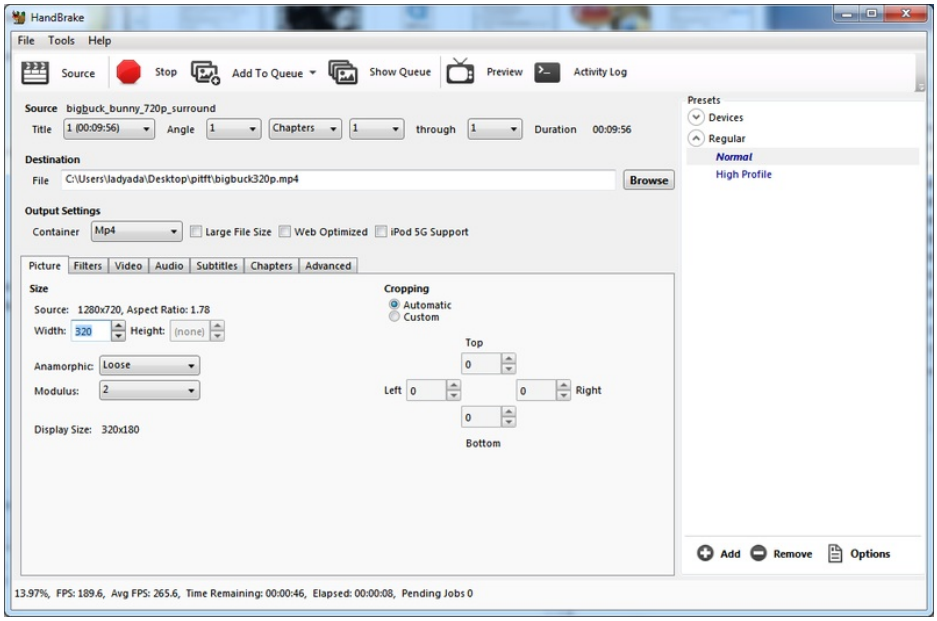
To do the conversion itself, we suggest HandBrake (https://adafru.it/cXT) which works great and is open source so it runs on all operating systems! Download and install from the link. Then run the installed application and open up the AVI file from before. The app will pre-fill a bunch of information about it.



Under **Destination** click **Browse...** to select a new MP4 file to save. Then under **Picture** change the **Width** to 320 (the height will be auto-calculated)



Click **START** to begin the conversion, it will take a minute or two.

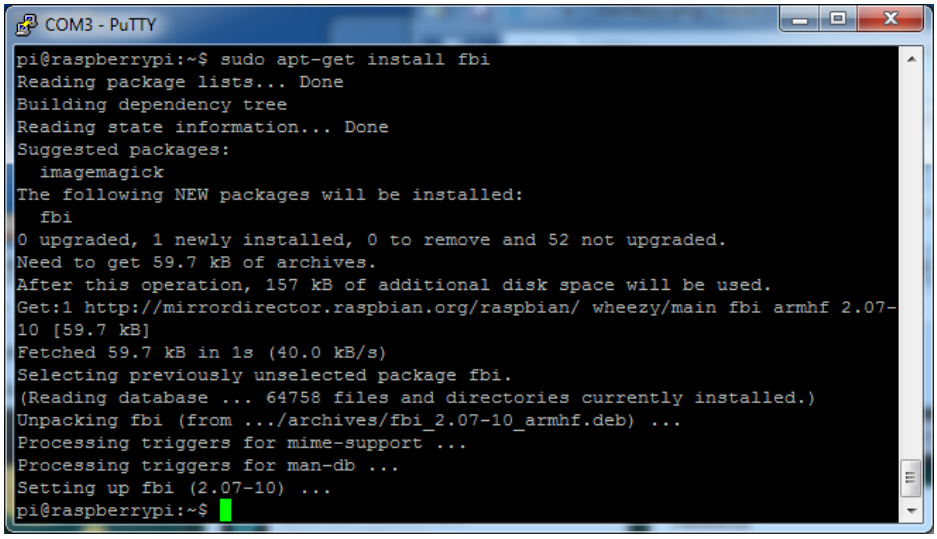


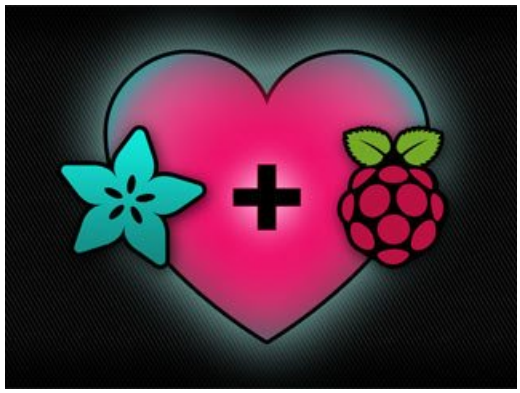
That's it! You now have a smaller file. Don't forget to play it on your computer to make sure it plays right before copying it to your Pi

Displaying Images

You can display every day images such as GIFs, JPGs, BMPs, etc on the screen. To do this we'll install **fbi** which is the **frame buffer image** viewer (not to be confused with the FBI agency!)

sudo apt-get install fbi will install it





Grab our lovely wallpapers with

```
wget http://adafruit-download.s3.amazonaws.com/adapiluv320x240.jpg
wget http://adafruit-download.s3.amazonaws.com/adapiluv480x320.png (https://adafru.it/cXU)
```

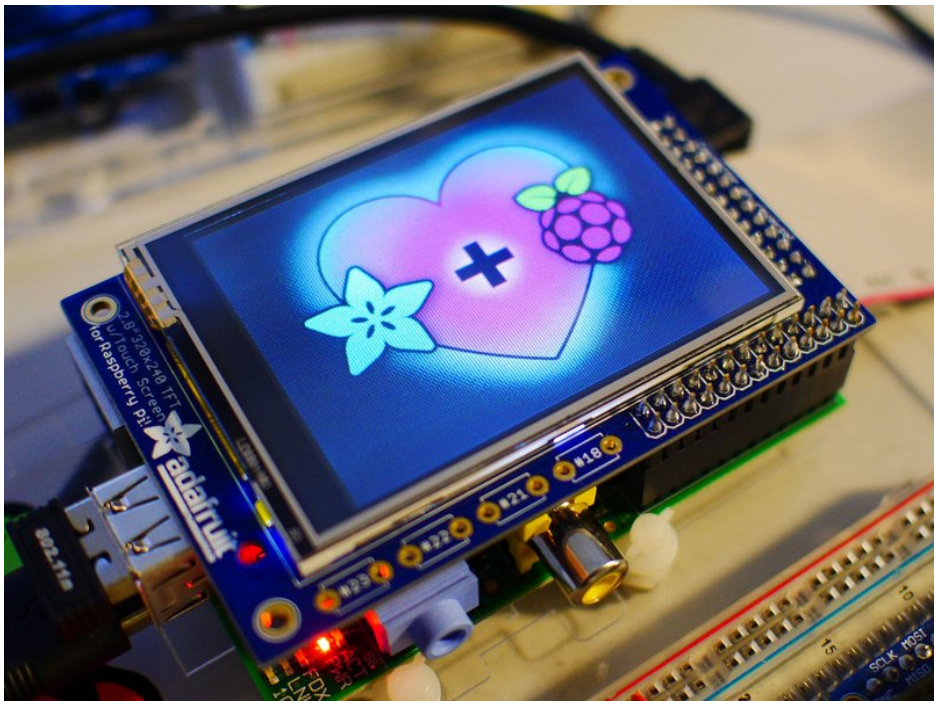
For 320x240 PiTFTs (2.2", 2.4", 2.8" or 3.2") view it with

```
sudo fbi -T 2 -d /dev/fb1 -noverbose -a adapiluv320x240.jpg
```

or for 3.5" PiTFTs:

```
sudo fbi -T 2 -d /dev/fb1 -noverbose -a adapiluv480x320 (https://adafru.it/cXU).png
```

That's it!


adafruit
2.8"320x240 TFT
w/Touch Screen
for Raspberry Pi!
#18
#21
#22
#25
SCLK
MOSI

Using FBCP

The Ideal: Adafruit's PiTFT displays are razor sharp. Whereas small composite screens on the Raspberry Pi usually require some video scaling (resulting in blurriness), PiTFT uses the GPIO header, digitally controlled pixel-by-pixel for a rock steady image. Though not a *lot* of pixels, it works great for retro gaming (and the display neatly stacks above the board, no side protuberances for video cables).

The Downside: this GPIO link entirely bypasses the Pi's video hardware, including the graphics accelerator. Many games and emulators *depend* on the GPU for performance gains. So the PiTFT has traditionally been limited to just a subset of specially-compiled emulators that can work and run well enough without the GPU.

The Solution: our latest PiTFT drivers, along with a tool called *fbcp* (framebuffer copy), careful system configuration, and (optionally) the more potent Raspberry Pi 2 board open the doors to many more gaming options. Existing emulator packages (such as RetroPie, with *dozens* of high-performance emulators and ports) — previously off-limits to the PiTFT — can run quite effectively now!

https://adafru.it/fbe

https://adafru.it/fbe

Backlight Control

The backlight of the 2.8" PiTFT has 4 LEDs in series and it draws ~75mA at all times, controlled by a transistor. The PiTFT 3.5" display has 6 LEDs in a row, and we use a boost converter to get the 5V from the Pi up to the ~20V needed to light up all the LEDs.

There might be times you'd like to save some power and turn off the backlight. The screen and touchplate will still work, you just can't see anything. We designed the board with the STMPE610 touchscreen controller which has 2 extra GPIO and tied one of them to control the backlight. You can use the command line to control the backlight.

By default, the backlight's on...but you can control it in two ways!

PWM Backlight Control with GPIO 18

If you want precise control, you can use the PWM output on GPIO 18. There's python code for controlling the PWM but you can also just use the kernel module and shell commands.

You'll need to make sure the STMPE control is not 'active' as the STMPE GPIO overrides the PWM output.

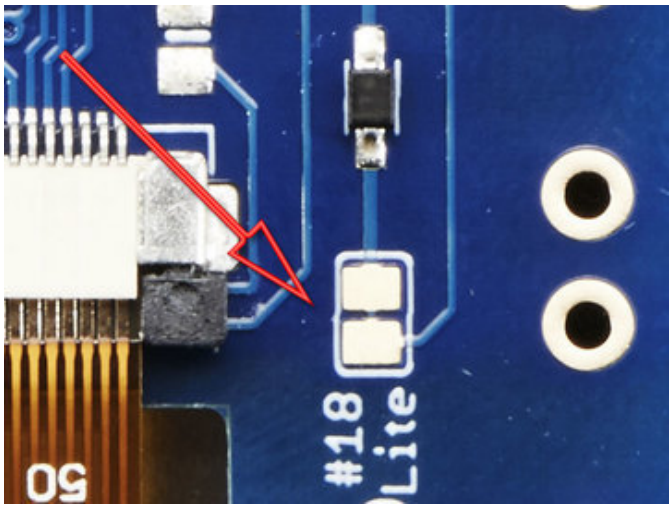
```
sudo sh -c 'echo "0" > /sys/class/backlight/soc\:backlight/brightness'
```

(Or if you are running an old kernel before the backlight object, try **sudo sh -c "echo 'in' > /sys/class/gpio/gpio508/direction"**)

OK now you can set the GPIO #18 pin to PWM mode using WiringPi's **gpio** command

With these basic shell commands, you can set the GPIO #18 pin to PWM mode with 1000 Hz frequency, set the output to 100 (out of 1023, so dim!), set the output to 1023 (out of 1023, nearly all the way on) and 0 (off)

```
gpio -g mode 18 pwm
gpio pwmc 1000
gpio -g pwm 18 100
gpio -g pwm 18 1023
gpio -g pwm 18 0
```



If you'd like to not have #18 control the backlight, simply cut the solder jumper, the tiny trace between the two large gold pads marked **Lite #18**

On / Off Using STMPE GPIO

Another option is to just turn it on and off using the extra GPIO created by the touchscreen driver

Thanks to the raspberry Pi overlay system, this GPIO is already set up for you in a file called **/sys/class/backlight/soc:backlight/brightness**

To turn the backlight off run

```
sudo sh -c 'echo "0" > /sys/class/backlight/soc\:backlight/brightness'
```

To turn it back on, run

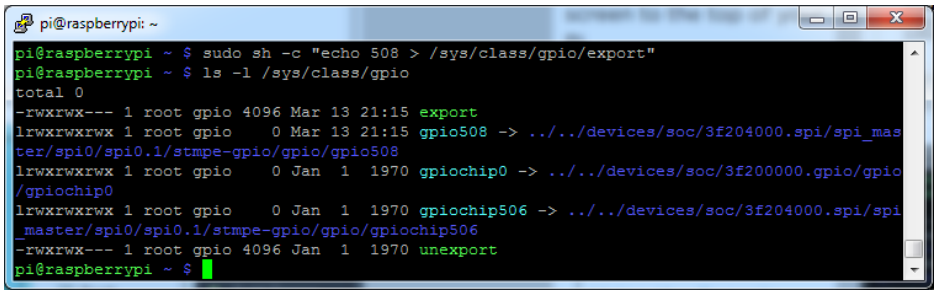
```
sudo sh -c 'echo "1" > /sys/class/backlight/soc\:backlight/brightness'
```

For older versions of PiTFT Kernel

On older versions of the PiTFT kernel/overlay, the GPIO was not tied to the backlight device. Start by getting access to the GPIO by making a device link

```
sudo sh -c "echo 508 > /sys/class/gpio/export"
ls -l /sys/class/gpio
```

For some *really* old versions, the GPIO pin was #252 not #508 so substitute that if you're running something from 2014 or earlier



Once you verify that you see GPIO #508, then you can set it to an output, this will turn off the display since it will output 0 by default

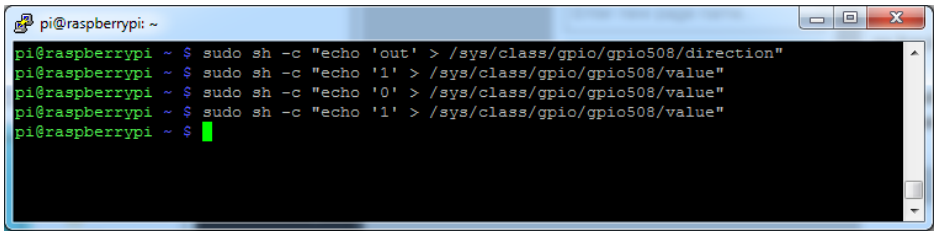
```
sudo sh -c "echo 'out' > /sys/class/gpio/gpio508/direction"
```

Then turn the display back on with

```
sudo sh -c "echo '1' > /sys/class/gpio/gpio508/value"
```

or back off

```
sudo sh -c "echo '0' > /sys/class/gpio/gpio508/value"
```


pi@raspberrypi: ~
pi@raspberrypi ~ $ sudo sh -c "echo 'out' > /sys/class/gpio/gpio508/direction"
pi@raspberrypi ~ $ sudo sh -c "echo '1' > /sys/class/gpio/gpio508/value"
pi@raspberrypi ~ $ sudo sh -c "echo '0' > /sys/class/gpio/gpio508/value"
pi@raspberrypi ~ $ sudo sh -c "echo '1' > /sys/class/gpio/gpio508/value"
pi@raspberrypi ~ $

PiTFT PyGame Tips

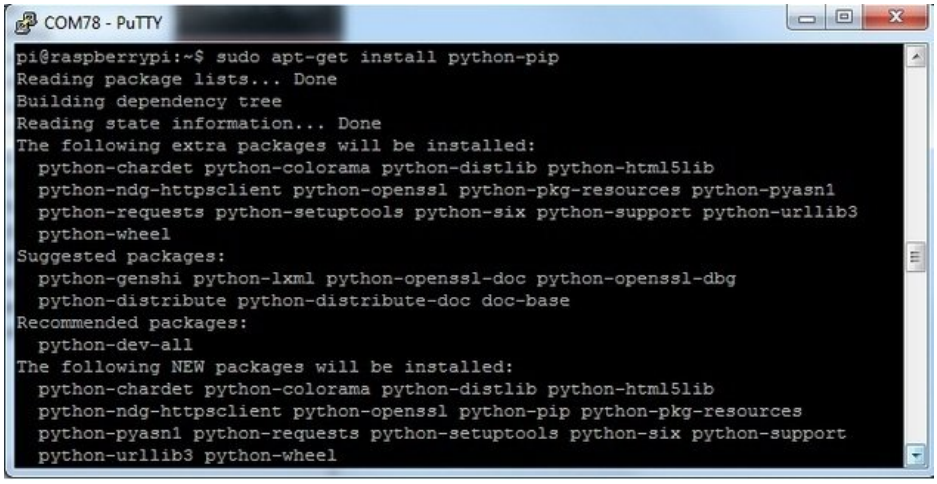
Since the PiTFT screen is fairly small, you may need to write custom UI programs. Pygame is the easiest way by far to do this.

Jeremy Blythe has an excellent tutorial here on getting started. (https://adafru.it/saw)

However, *before* you follow that link you'll want to set up pygame for the best compatibility:

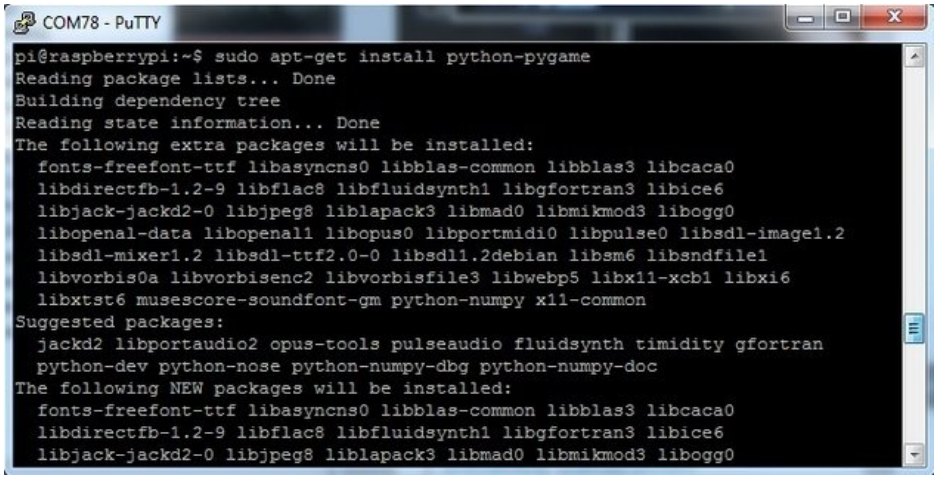
Install pip & pygame

Install Pip: **sudo apt-get install python-pip**



Install Pygame: **sudo apt-get install python-pygame**

(this will take a while)



Ensure you are running SDL 1.2

SDL 2.x and SDL 1.2.15-10 have some serious incompatibilities with touchscreen. You can force SDL 1.2 by running a script. (Thanks to heine in the forums! (https://adafru.it/sax))

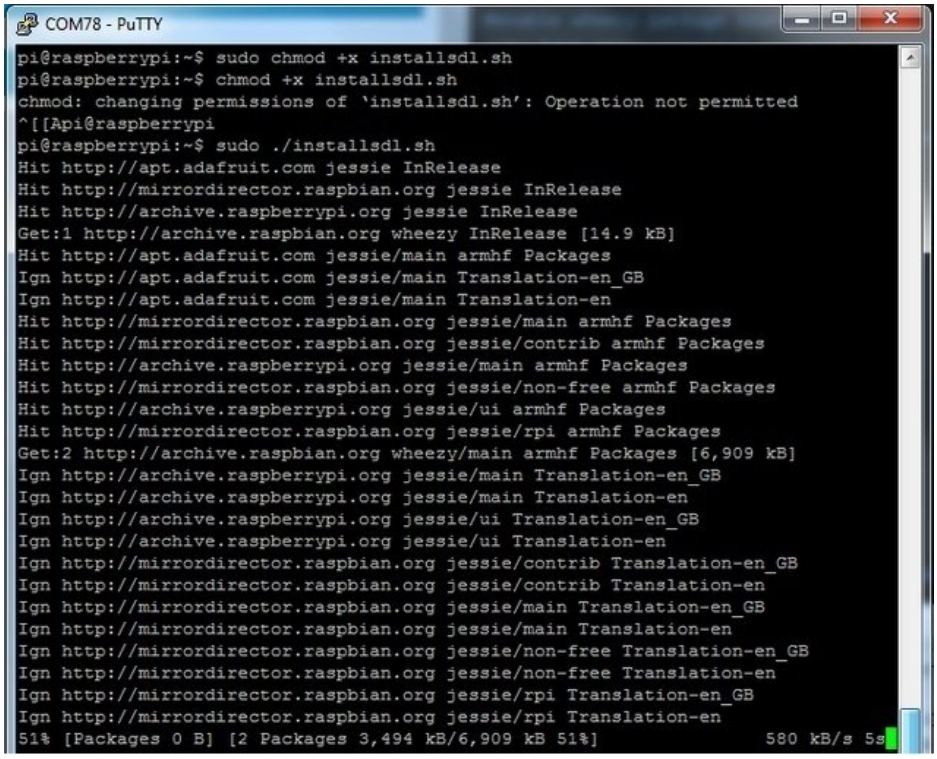
Edit a new file with **sudo nano installsdl.sh**
and paste in the following text:

```
#!/bin/bash

# enable wheezy package sources
echo "deb http://archive.raspbian.org/raspbian wheezy main
" > /etc/apt/sources.list.d/wheezy.list

# set stable as default package source (currently stretch)
echo "APT::Default-release \"stable\";
" > /etc/apt/apt.conf.d/10defaultRelease

# set the priority for libsdl from wheezy higher then the stretch package
echo "Package: libsdl1.2debian
Pin: release n=stretch
Pin-Priority: -10
Package: libsdl1.2debian
Pin: release n=wheezy
Pin-Priority: 900
" > /etc/apt/preferences.d/libsdl

# install
apt-get update
apt-get -y --allow-downgrades install libsdl1.2debian/wheezy
```

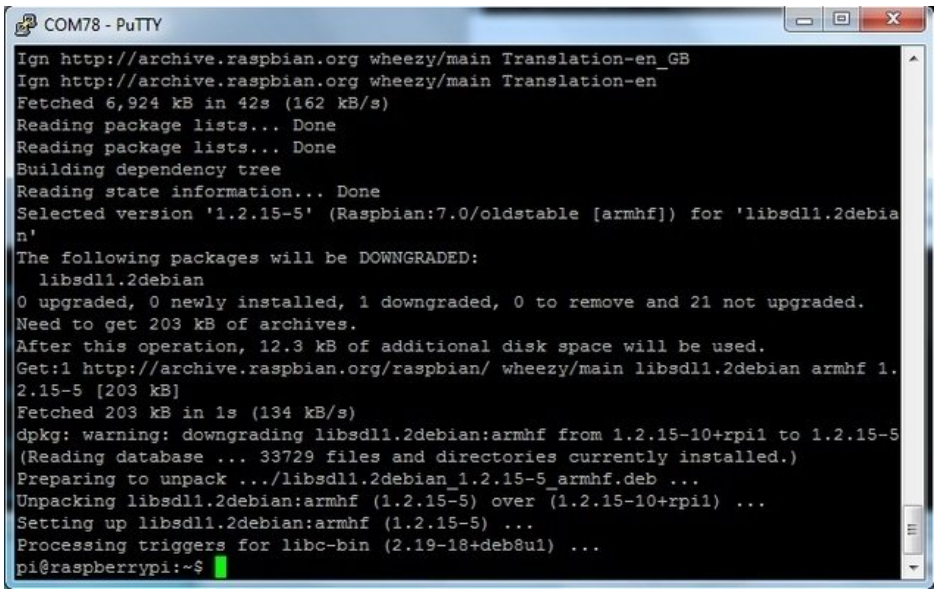
run

sudo chmod +x installsdl.sh

sudo ./installsdl.sh



it will force install SDL 1.2



OK **now** you can continue with pygame

Using the Capacitive touch screen in PyGame

The 2.8" Capacitive touch screen driver may not work by default in pygame, but this handy script shows how you can capture the device messages in python to create a UI

- https://github.com/nift4/pigame (https://adafru.it/CYv)
  For examples:

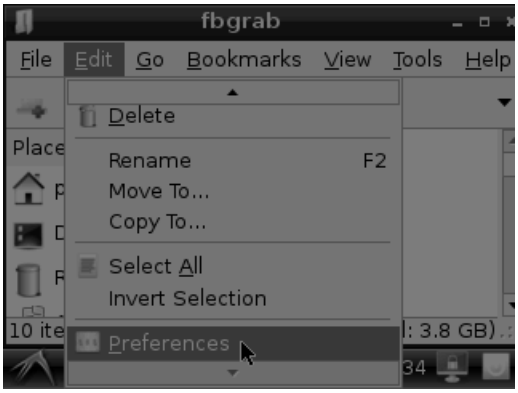
https://github.com/nift4/Raspberry-Pi-Testing (https://adafru.it/CYy)

Extras!

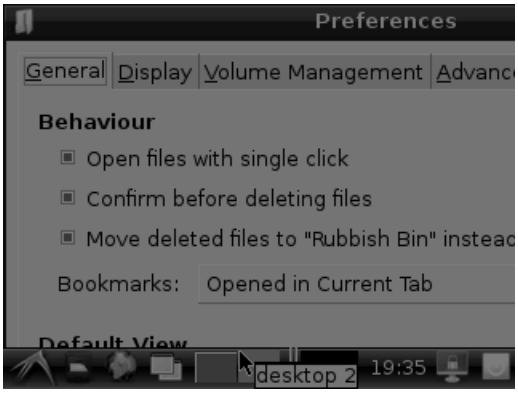
Making it easier to click icons in X

If you want to double-click on icons to launch something in X you may find it annoying to get it to work right. In LXDE you can simply set it up so that you only need to single click instead of double.
From LXDE launch the file manager (sorry these pix are grayscale, still figuring out how to screenshot the framebuffer!)

Then under the **Edit** menu, select **Preferences**



Then select **Open files with single click** and close the window (you'll need to drag it over to get to the X button



Right-click on a touchscreen

Obviously if you have a touchscreen, it cannot tell what finger you are pressing with. This means that all 'clicks' are left

clicks. But if you want a right-click, you *can* do it.

Just add the following lines into your InputClass of **/etc/X11/xorg.conf.d/99-calibration.conf** after the calibration section

```
 Option "EmulateThirdButton" "1"
Option "EmulateThirdButtonTimeout" "750"
Option "EmulateThirdButtonMoveThreshold" "30"
```

So for example your file will look like:

```
Section "InputClass"
  Identifier      "calibration"
  MatchProduct    "stmpe-ts"
  Option  "Calibration"   "3800 120 200 3900"
  Option  "SwapAxes"      "1"
  Option "EmulateThirdButton" "1"
  Option "EmulateThirdButtonTimeout" "750"
  Option "EmulateThirdButtonMoveThreshold" "30"
EndSection
```

This makes a right mouse click emulated when holding down the stylus for 750 ms.

(Thx adamaddin! (https://adafru.it/fH3))

Gesture Input

With the PiTFT touchscreen and xstroke (https://adafru.it/dD0) you can enter text in applications by drawing simple character gestures on the screen! Check out the video below for a short demonstration and overview of gesture input with xstroke:

Installation

Unfortunately xstroke hasn't been actively maintained for a few years so there isn't a binary package you can directly install. However compiling the tool is straightforward and easy with the steps below. Credit for these installation steps goes to mwilliams03 at ozzmaker.com (https://adafru.it/dD1).

First install a few dependencies by opening a command window on the Pi and executing:

```
sudo apt-get -y install build-essential libxft-dev libxpm-dev libxtst-dev
```

Now download, compile, and install xstroke by executing:

```
cd ~
wget http://mirror.egtvedt.no/avr32linux.org/twiki/pub/Main/XStroke/xstroke-0.6.tar.gz
tar xfv xstroke-0.6.tar.gz
cd xstroke-0.6
./configure
sed -i '/^X_LIBS = / s/$/ -lXrender -lX11 -lXext -ldl/' Makefile
make
sudo make install
```

If the commands above execute successfully xstroke should be installed. If you see an error message, carefully check the dependencies above were installed and try again.

Once xstroke is installed you will want to add a couple menu shortcuts to start and stop xstroke. Execute the following commands to install these shortcuts:

```
wget https://github.com/adafruit/PiTFT_Extras/raw/master/xstroke.desktop
wget https://github.com/adafruit/PiTFT_Extras/raw/master/xstrokekill.desktop
sudo cp xstroke*.desktop /usr/share/applications/
```

Usage

To use xstroke I highly recommend using a plastic stylus instead of your finger. Also calibrate the touchscreen for X-Windows (https://adafru.it/dD2) so you have the best control over the cursor possible.

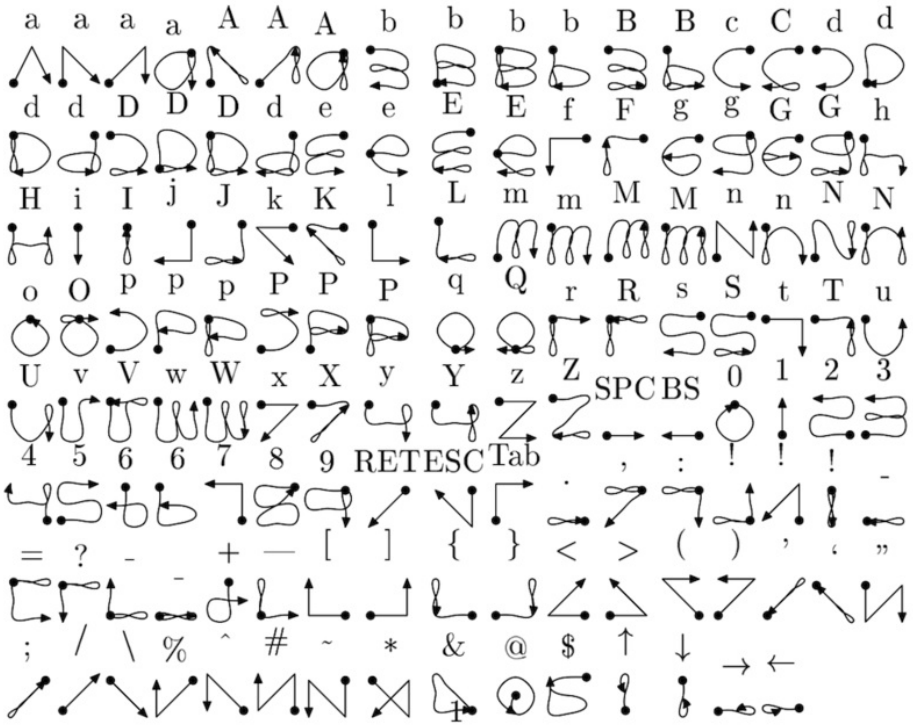
Don't use a ballpoint pen or sharp metal stylus as it could scratch or damage the touchscreen!

Start X-Windows on the PiTFT and open the LXDE menu by clicking the icon in the lower left corner. Scroll up to the **Accessories** menu at the top and notice the new **XStroke** and **XStroke Kill** commands.

Click the **XStroke** menu option to start xstroke. You should see a small pencil icon appear on the bottom right side of the screen. The pencil icon means xstroke is running, however by default it's not yet looking for gesture input.

Open an application that takes text input, such as LXTerminal. To enable gesture input click the xstroke pencil icon. You should see the pencil turn green and the text 'abc' written over top of the icon. You might need to click the icon a few times to get the click to register in the right spot.

When xstroke is looking for gesture input you can drag the mouse cursor in a gesture anywhere on the screen to send specific key strokes. Here's a picture of the possible gestures you can send:



(credit to Carl Worth for the image above)

To draw a gesture from the above image, press anywhere on the screen, start from the circle in the gesture, and follow the gesture pattern towards the arrow. As you draw a gesture you should see a blue line displayed that shows what you've drawn. Lift up the stylus when you get to the end of the gesture at the arrow. If xstroke recognizes the gesture it will send the appropriate key press to the active window. Try drawing a few characters from the image above to get the hang of writing gestures.

A few very useful gestures are backspace (which deletes a character), return/enter, and space. To draw a backspace gesture just draw a line going from the right side of the screen to the left side. The gesture for return/enter is a diagonal line from the top right to bottom left. Finally a space is a straight line from the left to the right.

Note that when xstroke is looking for gestures you might not be able to click or control the cursor as you normally would expect. To stop xstroke's gesture recognition carefully press the xstroke pencil icon again until the 'abc' text disappears. I've found this process can be a little finicky as the icon is very small and any movement will be interpreted as a gesture. Use a light touch and try a few times to click the icon.

If you get stuck completely and can't disable xstroke by clicking the icon, connect to the Raspberry Pi in a terminal/SSH connection and run 'killall xstroke' (without quotes) to force xstroke to quit. The normal way to stop xstroke is to

navigate to the **Accessories** -> **XStroke Kill** command, but you might not be able to do that if xstroke is listening for gesture input.

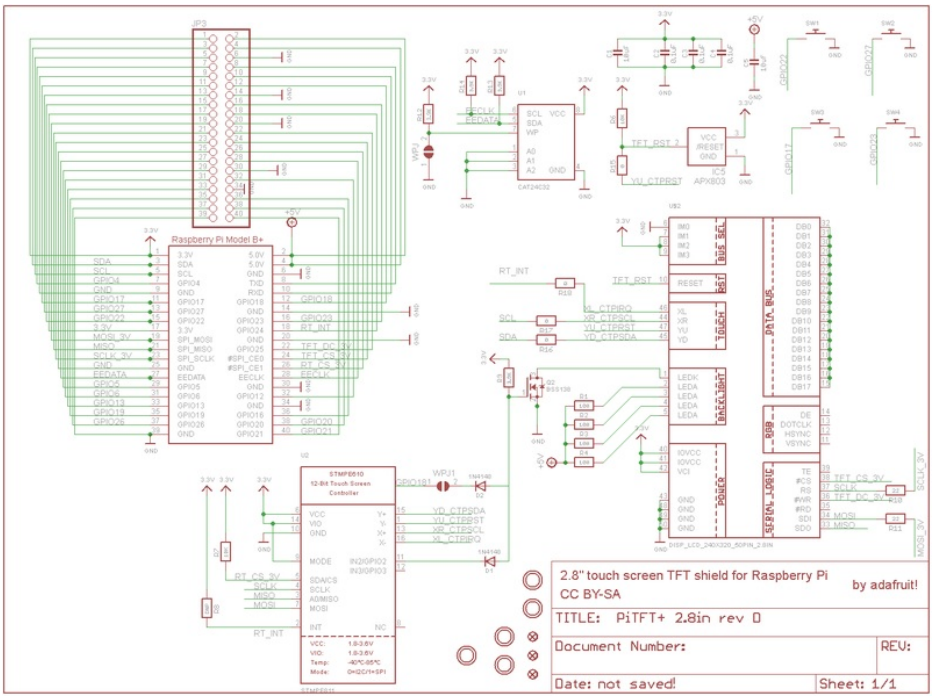
Have fun using xstroke to control your Pi by writing gestures on the PiTFT screen!

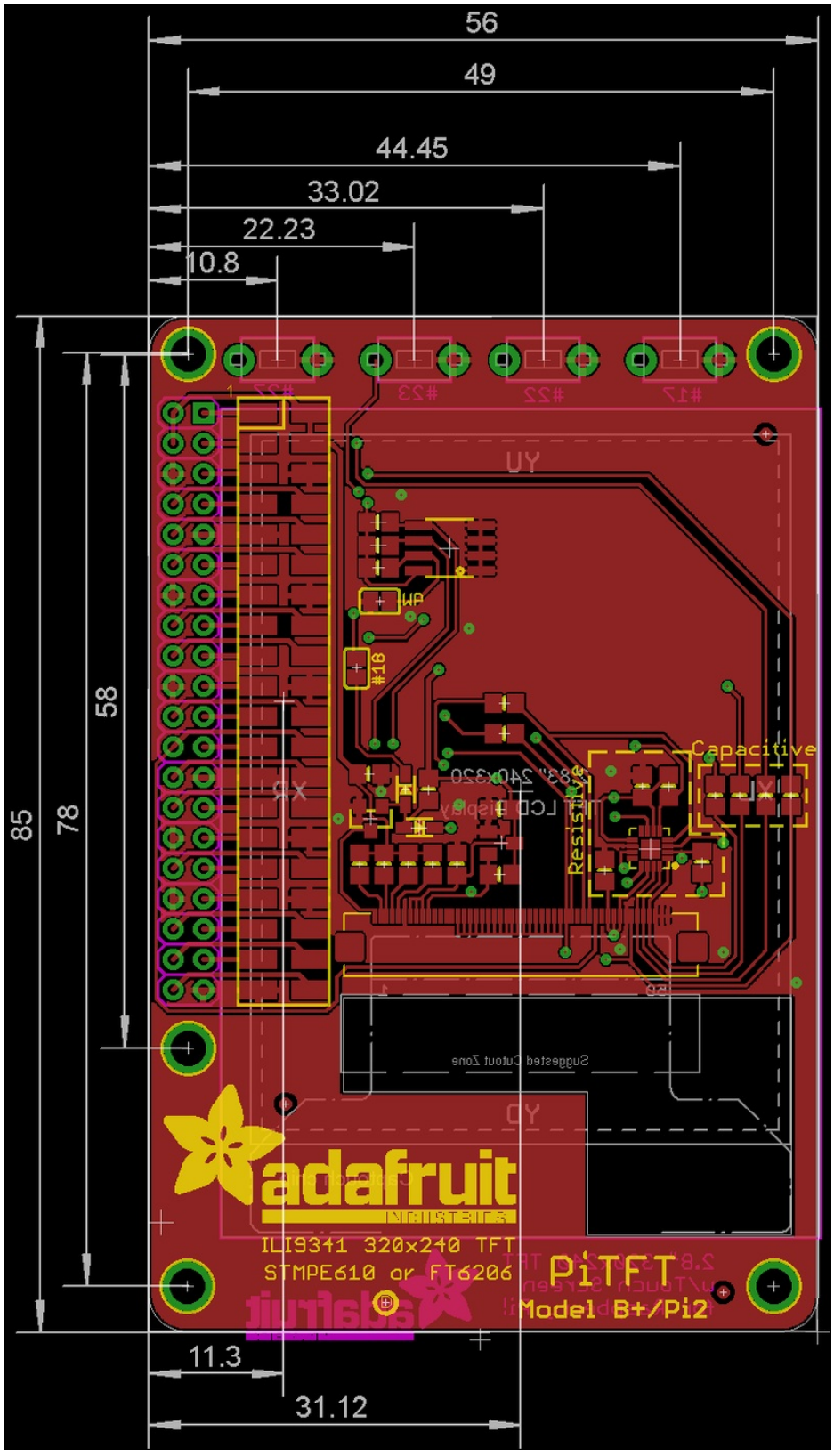
Downloads

- The latest kernel fork that adds all the TFT, touchscreen, and other addons is here on github (https://adafru.it/dcA)
- Datasheet for the 'raw' 2.8" TFT display (https://adafru.it/sEt)
- Original 2.8" PiTFT EagleCAD PCB Files on GitHub (https://adafru.it/oYB)
- PiTFT Plus 2.8" EagleCAD PCB Files on GitHub (https://adafru.it/oYC)
- PiTFT Plus 3.2" EagleCAD PCB Files on GitHub (https://adafru.it/rDv)
- Fritzing Files in the Adafruit Fritzing Library (https://adafru.it/aP3)

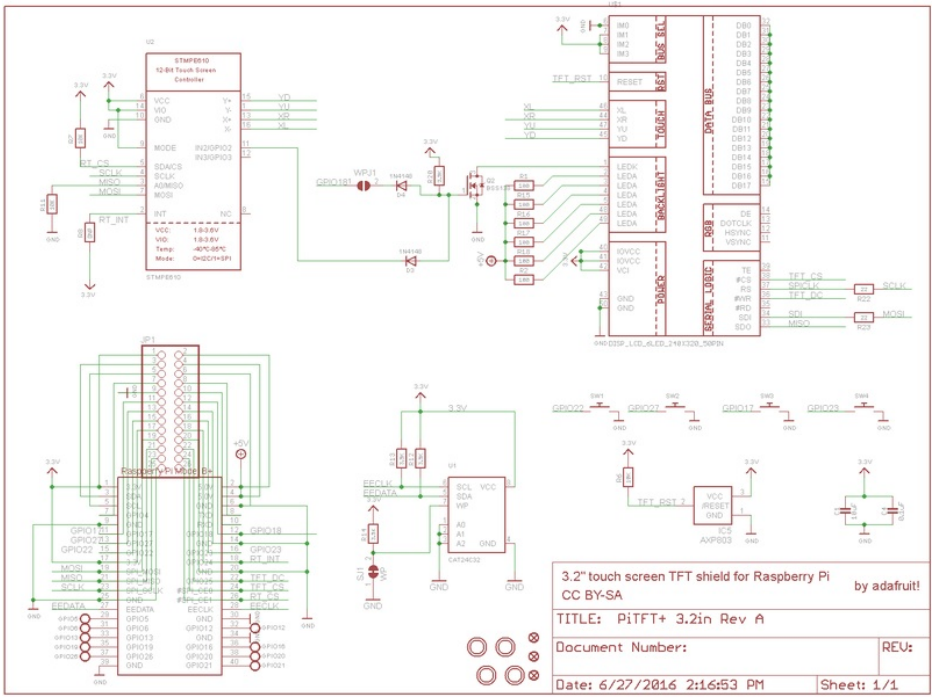
2.8" PiTFT Plus Schematic & Layout

For the Pi B+ & Pi 2 version (2x20 header)



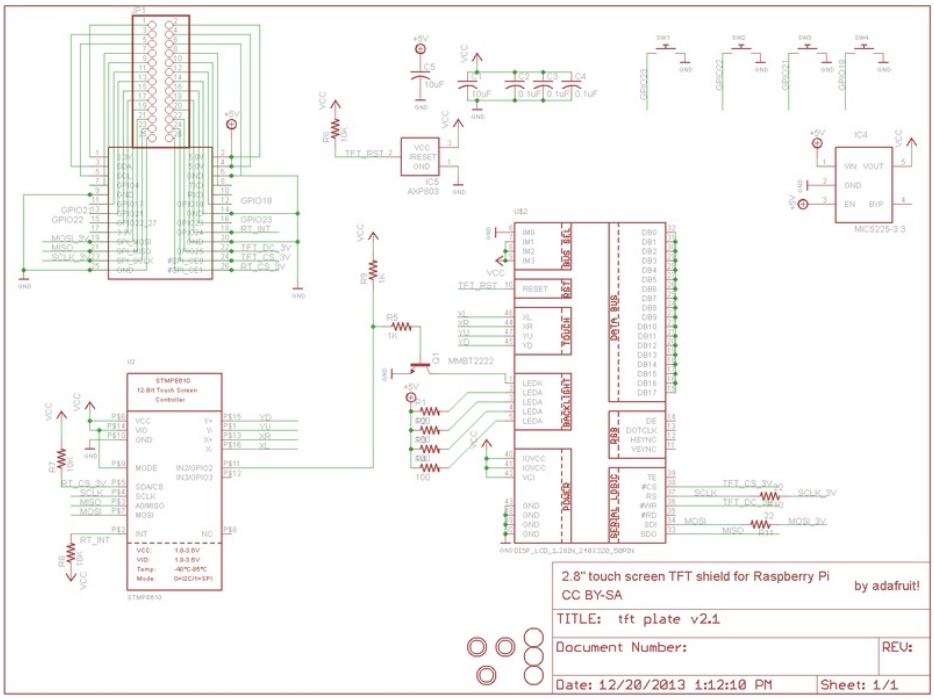


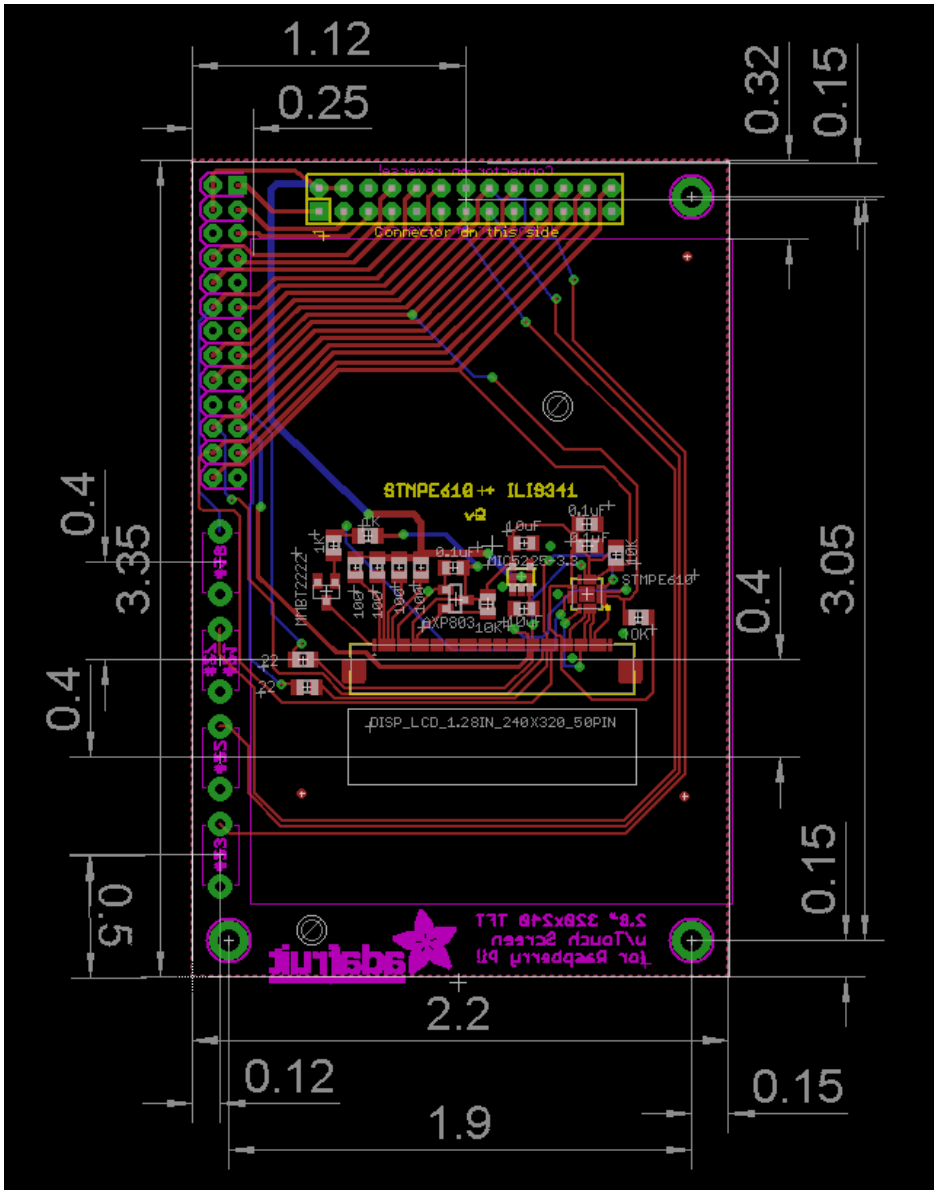
PiTFT 3.2" Plus Schematic



Original PiTFT 2.8" Schematic & Layout

For the Original Pi 1 version (2x13 header)




1.12
0.25
0.32
0.15
Connector on this side
STMPE610 + ILI9341
v2
0.4
3.35
0.4
3.05
0.4
0.15
0.5
2.2
0.12
1.9
0.15
DISP_LCD_1.28IN_240X320_50PIN
adafruit

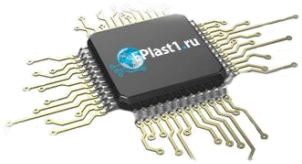
Компания «ЭлектроПласт» предлагает заключение долгосрочных отношений при поставках импортных электронных компонентов на взаимовыгодных условиях!

Наши преимущества:

- Оперативные поставки широкого спектра электронных компонентов отечественного и импортного производства напрямую от производителей и с крупнейших мировых складов;
- Поставка более 17-ти миллионов наименований электронных компонентов;
- Поставка сложных, дефицитных, либо снятых с производства позиций;
- Оперативные сроки поставки под заказ (от 5 рабочих дней);
- Экспресс доставка в любую точку России;
- Техническая поддержка проекта, помощь в подборе аналогов, поставка прототипов;
- Система менеджмента качества сертифицирована по Международному стандарту ISO 9001;
- Лицензия ФСБ на осуществление работ с использованием сведений, составляющих государственную тайну;
- Поставка специализированных компонентов (Xilinx, Altera, Analog Devices, Intersil, Interpoint, Microsemi, Aeroflex, Peregrine, Syfer, Eurofarad, Texas Instrument, Miteq, Cobham, E2V, MA-COM, Hittite, Mini-Circuits,General Dynamics и др.);

Помимо этого, одним из направлений компании «ЭлектроПласт» является направление «Источники питания». Мы предлагаем Вам помощь Конструкторского отдела:

- Подбор оптимального решения, техническое обоснование при выборе компонента;
- Подбор аналогов;
- Консультации по применению компонента;
- Поставка образцов и прототипов;
- Техническая поддержка проекта;
- Защита от снятия компонента с производства.



Как с нами связаться

Телефон: 8 (812) 309 58 32 (многоканальный)
Факс: 8 (812) 320-02-42
Электронная почта: org@eplast1.ru
Адрес: 198099, г. Санкт-Петербург, ул. Калинина, дом 2, корпус 4, литера А.