



Adafruit PiTFT 3.5" Touch Screen for Raspberry Pi

Created by lady ada



Last updated on 2015-04-29 03:00:10 PM EDT

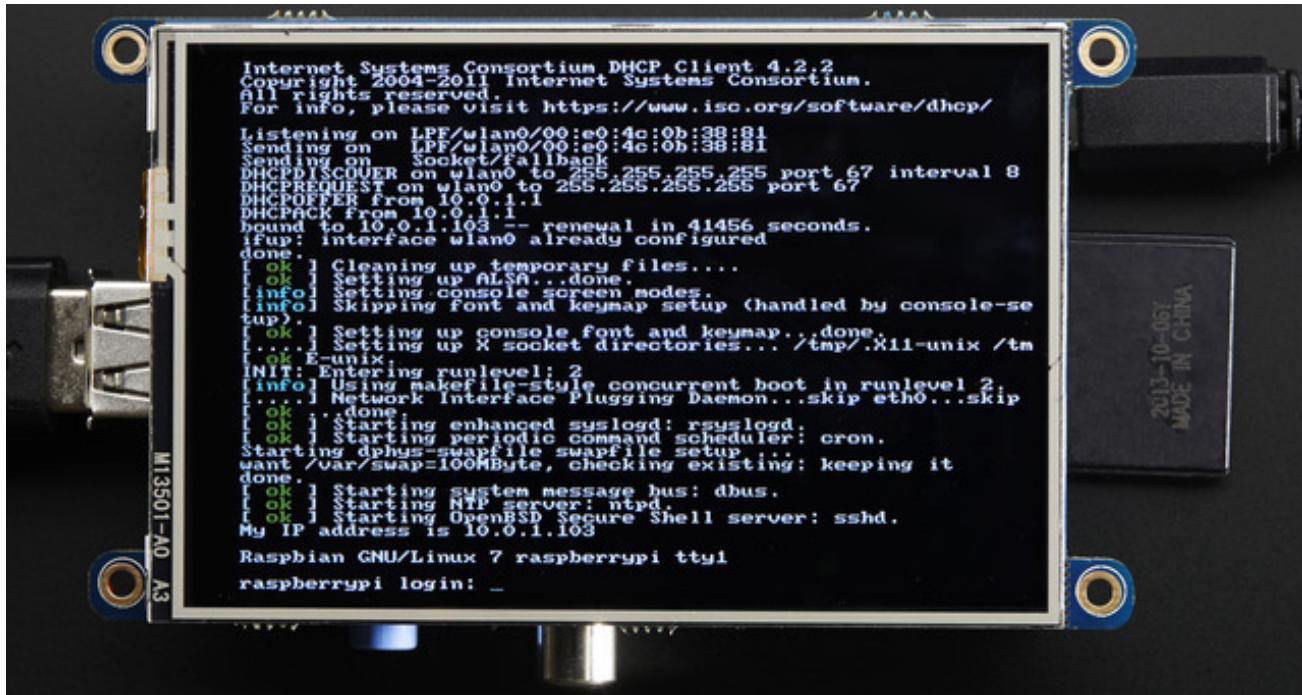
Guide Contents

Guide Contents	2
Overview	3
Easy Install	7
Ready to go image	7
DIY Installer script	7
Detailed Install	12
Before you start	13
Download & Install Kernel	13
Detailed Calibration	22
Setting up the Touchscreen	22
AutoMagic Calibration Script	25
Manual Calibration	27
X Calibration	29
Detailed Console Use	32
Turn off Console Blanking	35
Displaying Images	36
Backlight	38
On / Off Using STMPE GPIO	38
PWM Backlight Control with GPIO 18	39
More Tips	41
Making it easier to click icons in X	41
Gesture Input	43
FAQ	44
Downloads	47

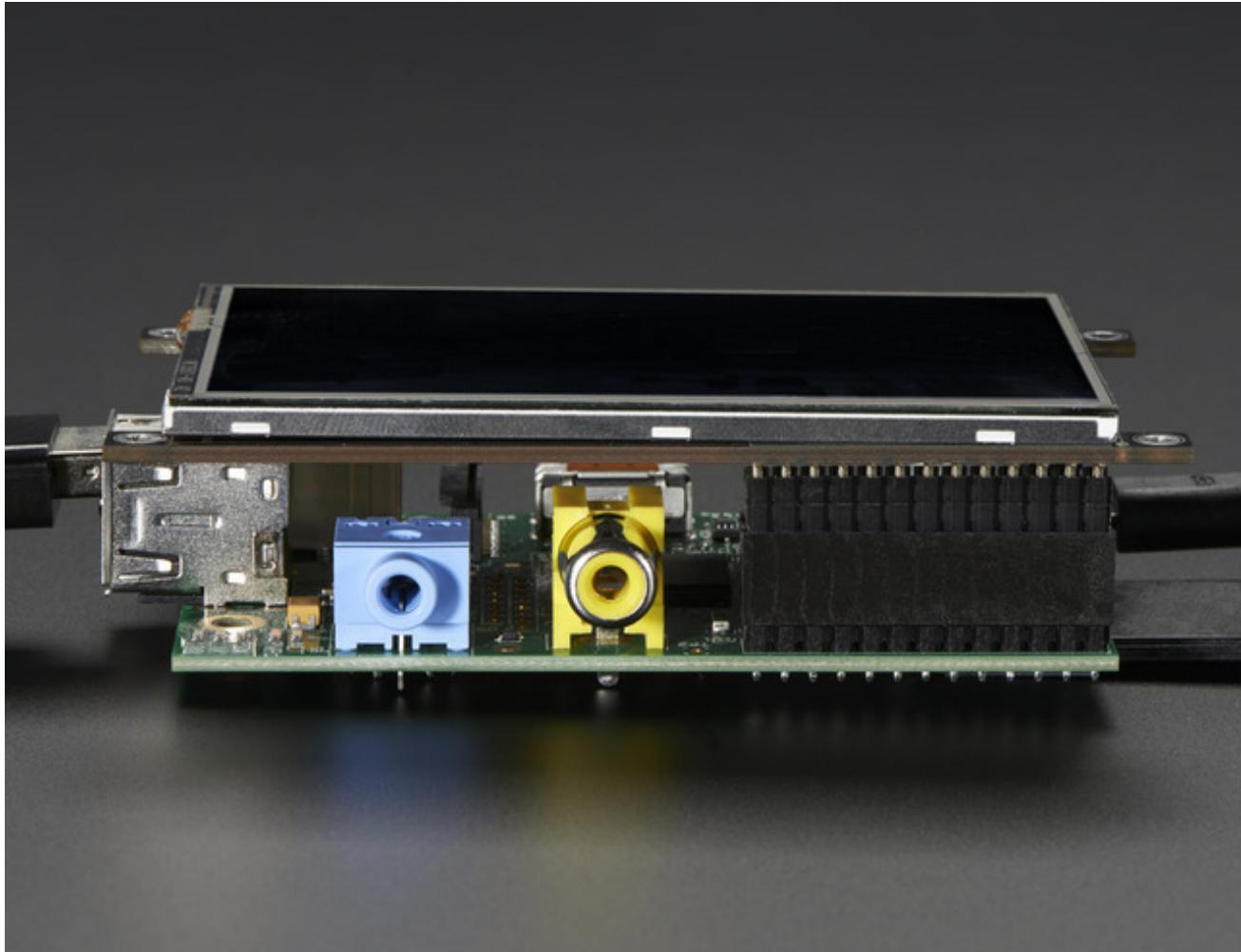
Overview



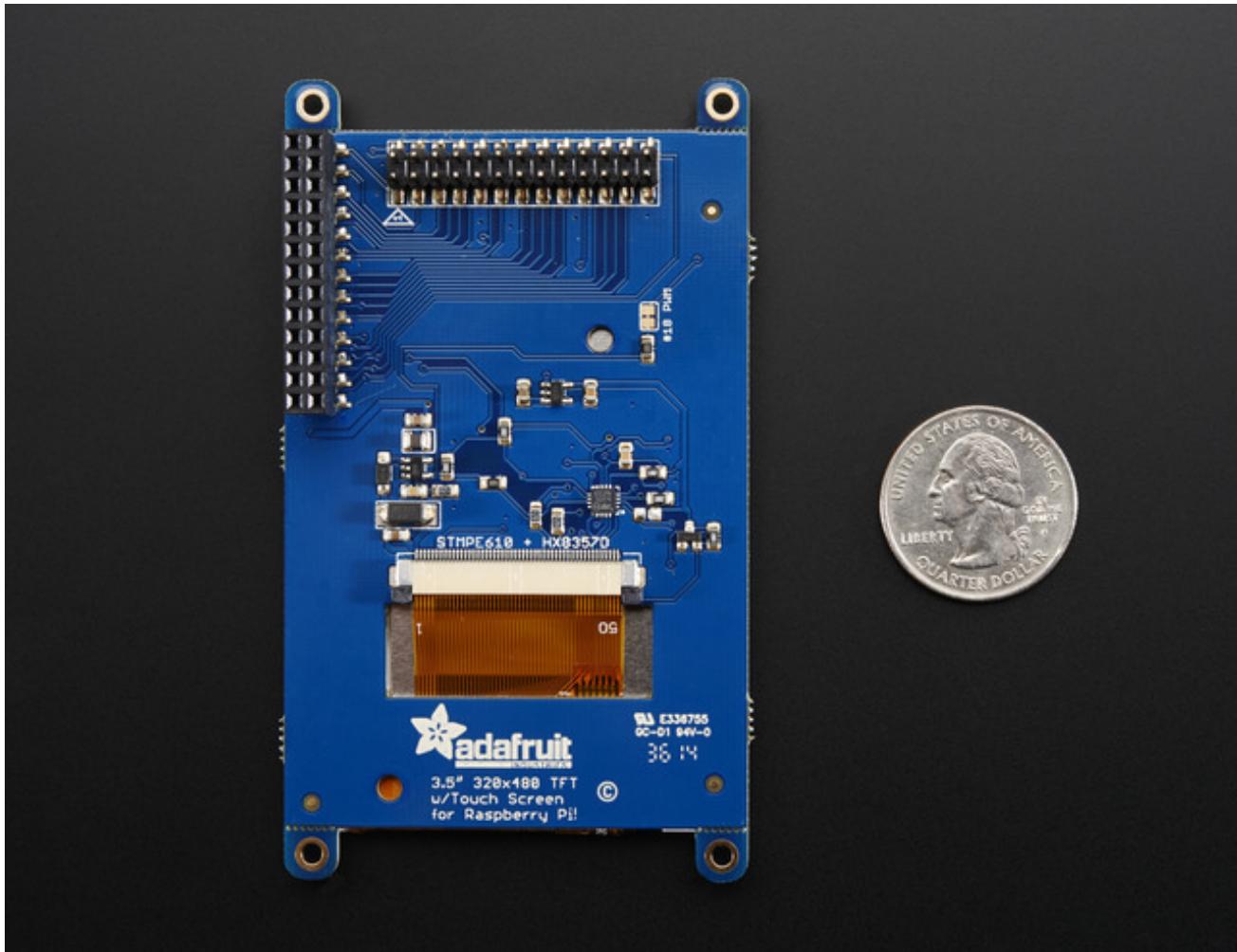
Is this not the cutest, little display for the Raspberry Pi? It features a **3.5" display with 480x320 16-bit color pixels** and a resistive touch overlay, but is only slightly larger than [our popular original](http://adafru.it/dDE) (<http://adafru.it/dDE>). The plate uses the high speed SPI interface 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!



It's designed to fit nicely onto the Pi Model A or B but also works perfectly fine with the Model B+ as long as you don't mind the PCB overhangs the USB ports by 5mm.



Uses the hardware SPI pins (SCK, MOSI, MISO, CE0, CE1) as well as GPIO #25 and #24. GPIO #18 can be used to PWM dim the backlight if you like. All other GPIO are unused. There's a 2x13 header on the bottom, you can connect a standard Pi GPIO cable to it to use any of the other pins ask you like



Best of all, it comes **fully assembled** and ready to plug into your Pi! You can use this as a display for running the X interface, or pygame. You can also have an HDMI display separately connected.

Easy Install

The PiTFT requires kernel support and a couple other things to make it a nice stand-alone display. We have a detailed step-by-step setup for hackers who want to tweak, customize or understand the PiTFT setup. If you just want to get going, check out the following for easy-install instructions!

Ready to go image

If you want to start with a fresh image, we have one for Raspbian - click here to download it and install into a new SD card. [Unzip and follow the classic SD card burning tutorials \(http://adafru.it/aMW\)](#)

Download PiTFT 3.5" Resistive Image
for Pi 1 and Pi 2 (March 12, 2015)

<http://adafru.it/eUE>

This image is customized for the RESISTIVE touch 3.5" TFT, also known as PID #2097! Not for PID #1601 or #1983

Older Images:

- [Raspbian 2014/09/09-based image \(http://adafru.it/e10\)](#)

DIY Installer script

If you don't want to download an image, you can run our installation package helper from inside your existing Raspbian install. It will download the kernel add-ons, and configure your Pi for PiTFT joy

[The helper is available for perusal here \(http://adafru.it/eIn\)](#) if you are interested in how it works

To download and run it, simply run the following commands:

```
curl -SLs https://apt.adafruit.com/add | sudo bash
sudo apt-get install adafruit-pitft-helper
```

The first command adds **apt.adafruit.com** to your repository list, so you can grab code directly from adafruit's servers

```
pi@raspberrypi: ~
pi@raspberrypi ~ $ curl -SLs https://apt.adafruit.com/add | sudo bash
.
.
.
-dMMs
+MMMMMo
.dMMMMMMN-
+NMMMMMMMd` 
`hMMMMMMMMMMo
-mMMMMMMMMMMMN.
dMMMMMMMMMMMMo
:hmmmmmmmmmmmdhs/. `'MMMMMMMMMMMMMMh
sMMMMMMMtMMMMMMMd+NMMMMMMMMMMMM+
/NMMMMMMMMMMMMMMMMMMMMMs+NMMMMm/+syys0/-` 
`hMMMMMMMMMMMMMMMMo hMMMMMMMMMMMMNNhs+:.
/NMMMMMMMMMMMNmmNMMMN. `mMMMMMMMMMMMMMMMMNNh-
.hMMMMMMMMMMh` `sNMMs-hMMMMMMMMMMMMMMMMMMMM+` 
/mMMMMMMMMNy+-./MMMMMy:....oMMMMMMMMMMMMNo.
:yMMMMMMMMMMMMMMMNy//oyMMMMMMMMMMMNy:
.yMMMMMMMs:oMNhNMMMMMMMMMMMMMNh/` 
.yMMMMMMMN/.dMy `sMMMMMMMMMNmy/` 
/NMMMMMMMM: `sMMMMM: sMMMMMs-...
-NMMMMMMMMMNMMMMMMMs ./MMMMMMh` 
mMMMMMMMtMMMMMMMMMMMMMMMMMMNs
sMMMMMMMMtMMMMNhMMMMMMMMMMMN
:MMMMMMMMMMMMMN+ hMMMMMMMMMMMN.
`dMMMMMMMMMMMNh/` `hMMMMMMMMMMMN.
```

The second does the actual download and installation, it'll take a while because there's a lot of software to replace for PiTFT support

```
pi@raspberrypi: ~ $ sudo apt-get install adafruit-pitft-helper
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  evtest fbi kpartx libraspberrypi-bin libraspberrypi-dev libraspberrypi-doc
  libraspberrypi0 libts-bin raspberrypi-bootloader wiringpi
Suggested packages:
  imagemagick
The following NEW packages will be installed:
  adafruit-pitft-helper evtest fbi kpartx libts-bin wiringpi
The following packages will be upgraded:
  libraspberrypi-bin libraspberrypi-dev libraspberrypi-doc libraspberrypi0
  raspberrypi-bootloader
5 upgraded, 6 newly installed, 0 to remove and 37 not upgraded.
Need to get 104 MB of archives.
After this operation, 97.6 MB of additional disk space will be used.
Do you want to continue [Y/n]? Y
Get:1 http://apt.adafruit.com/raspbian/ wheezy/main wiringpi armhf 2.24 [53.3 kB]
]
Get:2 http://apt.adafruit.com/raspbian/ wheezy/main libraspberrypi-bin armhf 1.2
0150309-1 [381 kB]
Get:3 http://apt.adafruit.com/raspbian/ wheezy/main libraspberrypi-dev armhf 1.2
0150309-1 [769 kB]
Get:4 http://mirrordirector.raspbian.org/raspbian/ wheezy/main fbi armhf 2.07-10
```

```
pi@raspberrypi: ~
hack/overlays/piscreen-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/pitft22-overlay.dtb to /usr/share/rpikernelh
ack/overlays/pitft22-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/pitft28c-overlay.dtb to /usr/share/rpikernel
hack/overlays/pitft28c-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/pitft28-resistive-overlay.dtb to /usr/share/
rpikernelhack/overlays/pitft28-resistive-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/pitft28r-overlay.dtb to /usr/share/rpikernel
hack/overlays/pitft28r-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/pitft35r-overlay.dtb to /usr/share/rpikernel
hack/overlays/pitft35r-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/pps-gpio-overlay.dtb to /usr/share/rpikernel
hack/overlays/pps-gpio-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/README to /usr/share/rpikernelhack/overlays/
README by rpikernelhack'
Adding 'diversion of /boot/overlays/rpi-display-overlay.dtb to /usr/share/rpiker
nelhack/overlays/rpi-display-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/spi-bcm2835-overlay.dtb to /usr/share/rpiker
nelhack/overlays/spi-bcm2835-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/w1-gpio-overlay.dtb to /usr/share/rpikernelh
ack/overlays/w1-gpio-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/w1-gpio-pullup-overlay.dtb to /usr/share/rpi
kernelhack/overlays/w1-gpio-pullup-overlay.dtb by rpikernelhack'
Unpacking replacement raspberrypi-bootloader ...
```

It's normal for the Pi to pause at this step for up to 20 minutes, there's a lot of kernel software to replace

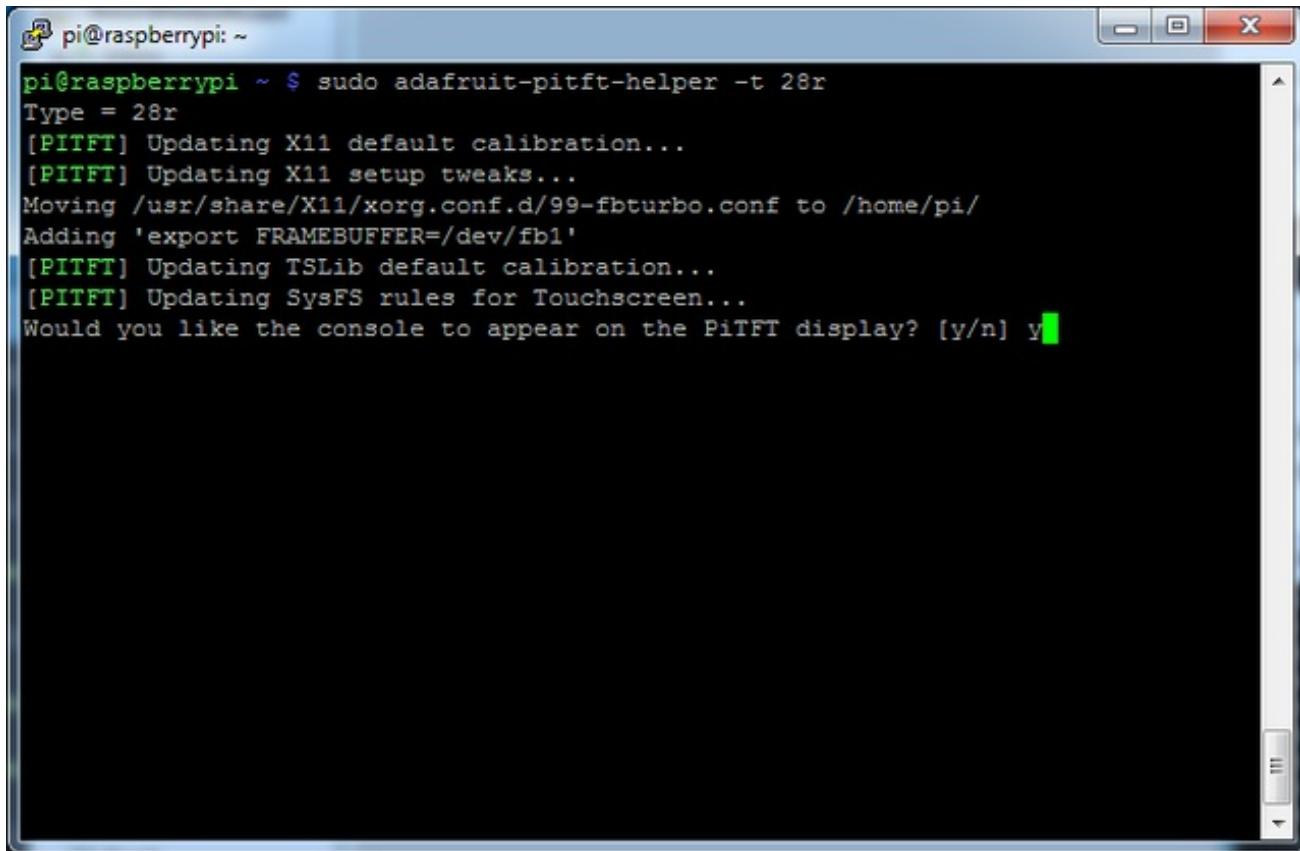
```
pi@raspberrypi: ~
Removing 'diversion of /boot/overlays/pitft28-resistive-overlay.dtb to /usr/share/rpikernelhack/overlays/pitft28-resistive-overlay.dtb by rpikernelhack'
Removing 'diversion of /boot/overlays/pitft28r-overlay.dtb to /usr/share/rpikernelhack/overlays/pitft28r-overlay.dtb by rpikernelhack'
Removing 'diversion of /boot/overlays/pitft35r-overlay.dtb to /usr/share/rpikernelhack/overlays/pitft35r-overlay.dtb by rpikernelhack'
Removing 'diversion of /boot/overlays/pps-gpio-overlay.dtb to /usr/share/rpikernelhack/overlays/pps-gpio-overlay.dtb by rpikernelhack'
Removing 'diversion of /boot/overlays/README to /usr/share/rpikernelhack/overlays/README by rpikernelhack'
Removing 'diversion of /boot/overlays/rpi-display-overlay.dtb to /usr/share/rpikernelhack/overlays/rpi-display-overlay.dtb by rpikernelhack'
Removing 'diversion of /boot/overlays/spi-bcm2835-overlay.dtb to /usr/share/rpikernelhack/overlays/spi-bcm2835-overlay.dtb by rpikernelhack'
Removing 'diversion of /boot/overlays/w1-gpio-overlay.dtb to /usr/share/rpikernelhack/overlays/w1-gpio-overlay.dtb by rpikernelhack'
Removing 'diversion of /boot/overlays/w1-gpio-pullup-overlay.dtb to /usr/share/rpikernelhack/overlays/w1-gpio-pullup-overlay.dtb by rpikernelhack'
Setting up libraspberrypi0 (1.20150309-1) ...
Setting up libraspberrypi-bin (1.20150309-1) ...
Setting up libraspberrypi-dev (1.20150309-1) ...
Setting up libraspberrypi-doc (1.20150309-1) ...
Setting up kpartx (0.4.9+git0.4dfdaf2b-7~deb7u2) ...
Setting up adafruit-pitft-helper (0.4.1-1) ...
pi@raspberrypi ~ $
```

OK now the kernel is installed, all you have to do is run the helper which will configure the kernel device tree overlays and add the few configurations to make the console show up, etc.

```
sudo adafruit-pitft-helper -t 35
```

This will install the "3.5 inch Resistive" type of PiTFT into the current install.

At the end you will be prompted on whether you want the text console to appear on the PiTFT. Answer Y or N depending on your personal desires!



```
pi@raspberrypi: ~
pi@raspberrypi ~ $ sudo adafruit-pitft-helper -t 28r
Type = 28r
[PITFT] Updating X11 default calibration...
[PITFT] Updating X11 setup tweaks...
Moving /usr/share/X11/xorg.conf.d/99-fbturbo.conf to /home/pi/
Adding 'export FRAMEBUFFER=/dev/fb1'
[PITFT] Updating TSlib default calibration...
[PITFT] Updating SysFS rules for Touchscreen...
Would you like the console to appear on the PiTFT display? [y/n] y
```

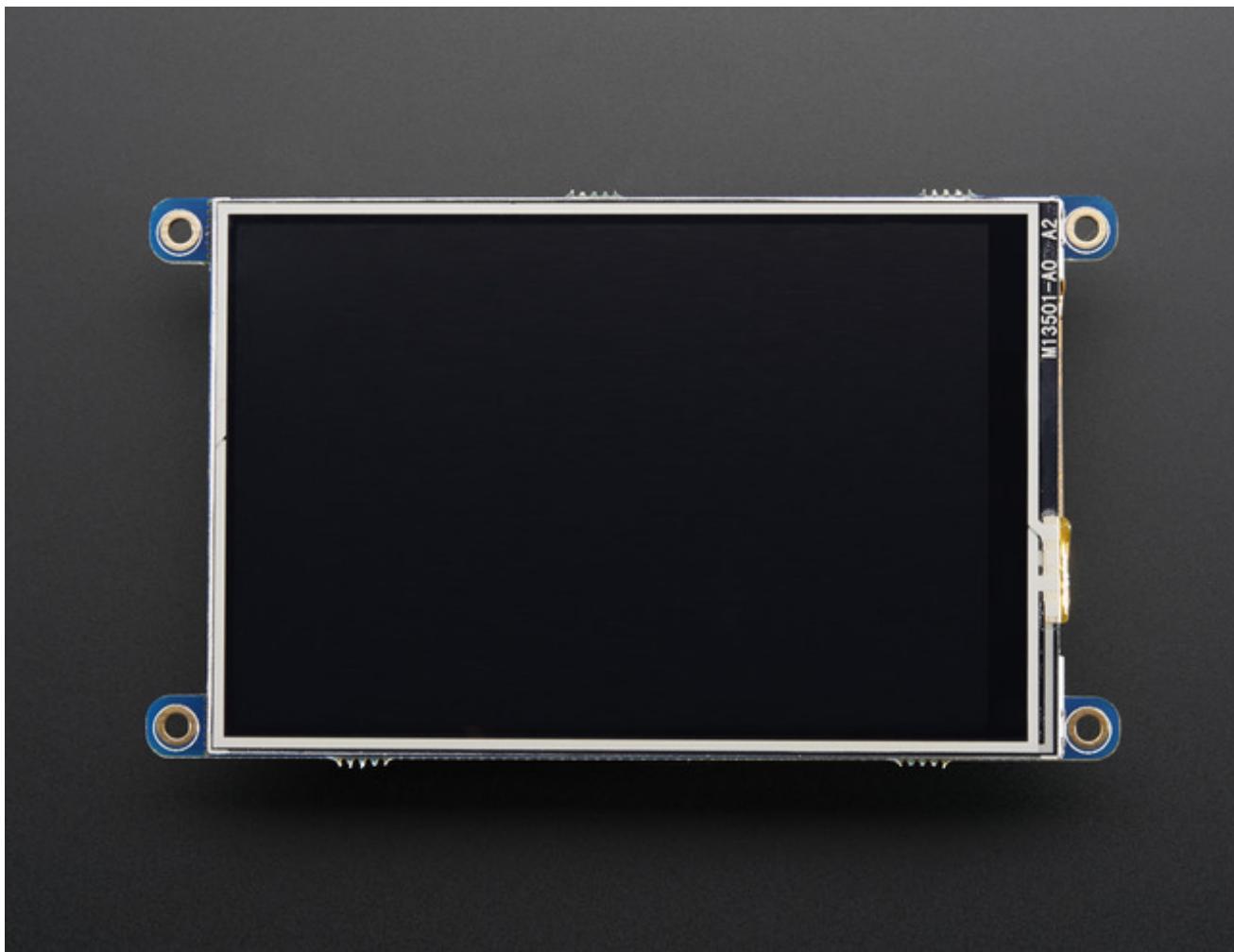
Thats it!

Run **sudo reboot** to try out your fancy new PiTFT :)

Detailed Install

If you've grabbed our Easy Install image, or use the 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 kernel install

In the next few steps we'll cover the **detailed** installation procedure. Chances are, you should grab the Easy Install image or script. If you have some interest in the details of how we install the PiTFT setup, read on!



In order to add support for the 3.5" TFT and touchscreen, we'll need to install a new Linux Kernel. Lucky for you, we created a kernel package that you can simply install over your current Raspbian (or Raspbian-derived) install instead of needing a whole new image. This makes it easier to keep your install up-to-date.

To use our kernel .deb files you must be using Raspbian or derivative. This wont work with Arch or other Linux flavors. As Raspbian is the official OS for the Pi, that's the only Linux we will support!

[Others can recompile their own kernel using our github commits](http://adafru.it/aPa) (<http://adafru.it/aPa>) but we have no tutorial or support or plans for such.

Before you start

You'll need a working install of Raspbian with network access. [If you need help getting that far, check out our collection of Pi tutorials](http://adafru.it/aWq) (<http://adafru.it/aWq>).

We'll be doing this from a console cable connection, but you can just as easily do it from the direct HDMI/TV console or by SSH'ing in. Whatever gets you to a shell will work!

Also, run **sudo apt-get update** !

To run these all the setup and config commands you'll need to be logged into a proper Terminal - use ssh, a console cable, or the main text console (on a TV). The WebIDE console may not work.

Download & Install Kernel

The only way we're distributing the PiTFT kernel packages right now is thru apt.adafruit.com so you'll still need to run:

```
curl -SLs https://apt.adafruit.com/add | sudo bash
```

To add apt.adafruit.com to your list of software sources

```
pi@raspberrypi: ~
pi@raspberrypi ~ $ curl -SLs https://apt.adafruit.com/add | sudo bash
          .
         -dMs
        +MMMMo
       .dMMMMMN-
      +NMMMMNMd` 
     `hMMMMMMNMMo
    -mMMMMMMNMN.
   dMMMMMMNMNMo
:hmmmmmmmmmmmdhs/. `MMMMMMMMNMNh
sMMMMMMMMMMMMMd+NMMMMMMMMNMNh+
/NMMMMMMMMMMMMMMMMMs+NMMMMNm/+syysc/-` 
`hMMMMMMMMMMMMMMMMMo hMMMMMMMMNMNh+::
/NMMMMMMMMMMNMnmNMMMN. `mMMMMMMMMNMNh-
.hMMMMNMNh` `sNMMs-hMMMMMMMMNMNh+::
/mMMMMMMNMNy+-./MMMMMy:...oMMMMNMNo.
:ymMMMMNMNy//oymMMMMNMNy:
.yMMMMNMNs :oMMNhNMMMNy/` 
.yMMMMNMN/ .dMy `sMMMMNMNy/` 
/NMMMMNMN: `sMMMM: sMMMMNs-..` 
-NMMMMNMNMMNs ./MMMMNMNh` 
mMMMMNMNMMNs ./MMMMNMNs
sMMMMNMNMMNs ./MMMMNMNhMMMMNMNMMNs
:MMMMNMNMMNs+ hMMMMNMNMMNs.
`dMMMMNMN Nh/` `hMMMMNMNMMNs.
```

Then install the kernel with

```
pi@raspberrypi: ~ $ sudo apt-get install adafruit-pitft-helper
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  evtest fbi kpartx libraspberrypi-bin libraspberrypi-dev libraspberrypi-doc
  libraspberrypi0 libts-bin raspberrypi-bootloader wiringpi
Suggested packages:
  imagemagick
The following NEW packages will be installed:
  adafruit-pitft-helper evtest fbi kpartx libts-bin wiringpi
The following packages will be upgraded:
  libraspberrypi-bin libraspberrypi-dev libraspberrypi-doc libraspberrypi0
  raspberrypi-bootloader
5 upgraded, 6 newly installed, 0 to remove and 37 not upgraded.
Need to get 104 MB of archives.
After this operation, 97.6 MB of additional disk space will be used.
Do you want to continue [Y/n]? Y
Get:1 http://apt.adafruit.com/raspbian/ wheezy/main wiringpi armhf 2.24 [53.3 kB]
]
Get:2 http://apt.adafruit.com/raspbian/ wheezy/main libraspberrypi-bin armhf 1.2
0150309-1 [381 kB]
Get:3 http://apt.adafruit.com/raspbian/ wheezy/main libraspberrypi-dev armhf 1.2
0150309-1 [769 kB]
Get:4 http://mirrordirector.raspbian.org/raspbian/ wheezy/main fbi armhf 2.07-10
```

```
pi@raspberrypi: ~
hack/overlays/piscreen-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/pitft22-overlay.dtb to /usr/share/rpikernelh
ack/overlays/pitft22-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/pitft28c-overlay.dtb to /usr/share/rpikernel
hack/overlays/pitft28c-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/pitft28-resistive-overlay.dtb to /usr/share/
rpikernelhack/overlays/pitft28-resistive-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/pitft28r-overlay.dtb to /usr/share/rpikernel
hack/overlays/pitft28r-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/pitft35r-overlay.dtb to /usr/share/rpikernel
hack/overlays/pitft35r-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/pps-gpio-overlay.dtb to /usr/share/rpikernel
hack/overlays/pps-gpio-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/README to /usr/share/rpikernelhack/overlays/
README by rpikernelhack'
Adding 'diversion of /boot/overlays/rpi-display-overlay.dtb to /usr/share/rpiker
nelhack/overlays/rpi-display-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/spi-bcm2835-overlay.dtb to /usr/share/rpiker
nelhack/overlays/spi-bcm2835-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/w1-gpio-overlay.dtb to /usr/share/rpikernelh
ack/overlays/w1-gpio-overlay.dtb by rpikernelhack'
Adding 'diversion of /boot/overlays/w1-gpio-pullup-overlay.dtb to /usr/share/rpi
kernelhack/overlays/w1-gpio-pullup-overlay.dtb by rpikernelhack'
Unpacking replacement raspberrypi-bootloader ...
```

```
pi@raspberrypi: ~
Removing 'diversion of /boot/overlays/pitft28-resistive-overlay.dtb to /usr/share/rpikernelhack/overlays/pitft28-resistive-overlay.dtb by rpikernelhack'
Removing 'diversion of /boot/overlays/pitft28r-overlay.dtb to /usr/share/rpikernelhack/overlays/pitft28r-overlay.dtb by rpikernelhack'
Removing 'diversion of /boot/overlays/pitft35r-overlay.dtb to /usr/share/rpikernelhack/overlays/pitft35r-overlay.dtb by rpikernelhack'
Removing 'diversion of /boot/overlays/pps-gpio-overlay.dtb to /usr/share/rpikernelhack/overlays/pps-gpio-overlay.dtb by rpikernelhack'
Removing 'diversion of /boot/overlays/README to /usr/share/rpikernelhack/overlays/README by rpikernelhack'
Removing 'diversion of /boot/overlays/rpi-display-overlay.dtb to /usr/share/rpikernelhack/overlays/rpi-display-overlay.dtb by rpikernelhack'
Removing 'diversion of /boot/overlays/spi-bcm2835-overlay.dtb to /usr/share/rpikernelhack/overlays/spi-bcm2835-overlay.dtb by rpikernelhack'
Removing 'diversion of /boot/overlays/w1-gpio-overlay.dtb to /usr/share/rpikernelhack/overlays/w1-gpio-overlay.dtb by rpikernelhack'
Removing 'diversion of /boot/overlays/w1-gpio-pullup-overlay.dtb to /usr/share/rpikernelhack/overlays/w1-gpio-pullup-overlay.dtb by rpikernelhack'
Setting up libraspberrypi0 (1.20150309-1) ...
Setting up libraspberrypi-bin (1.20150309-1) ...
Setting up libraspberrypi-dev (1.20150309-1) ...
Setting up libraspberrypi-doc (1.20150309-1) ...
Setting up kpartx (0.4.9+git0.4dfdaf2b-7~deb7u2) ...
Setting up adafruit-pitft-helper (0.4.1-1) ...
pi@raspberrypi ~ $
```

OK since you're not going to run the helper, lets add the device tree overlay manually. Edit /boot/config.txt with

sudo nano /boot/config.txt

and add the following lines at the end:

```
[pi1]
device_tree=bcm2708-rpi-b-plus.dtb
[pi2]
device_tree=bcm2709-rpi-2-b.dtb
[all]
dtparam=spi=on
dtparam=i2c1=on
dtparam=i2c_arm=on
dtoverlay=pitft35r,rotate=90,speed=42000000,fps=20
```

```
pi@raspberrypi: ~
GNU nano 2.2.6          File: /boot/config.txt

#config_hdmi_boost=4

# uncomment for composite PAL
#sdtv_mode=2

#uncomment to overclock the arm. 700 MHz is the default.
#arm_freq=800

[pi1]
device_tree=bcm2708-rpi-b-plus.dtb
[pi2]
device_tree=bcm2709-rpi-2-b.dtb
[all]
dtparam=spi=on
dtparam=i2c1=on
dtparam=i2c_arm=on
dtoverlay=pitft35r,rotate=90,speed=42000000,fps=20

^G Get Help  ^C WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is   ^V Next Page  ^U UnCut Text^T To Spell
```

The **rotate=** variable tells the driver to rotate the screen **0** **90** **180** or **270** degrees.

0 is portrait, with the bottom near the USB jacks

90 is landscape, with the bottom of the screen near the headphone jack

180 is portrait, with the top near the USB jacks

270 is landscape, with the top of the screen near the headphone jack.

You can change this file with **nano** and reboot to make the change stick.

The **speed=** variable tells the driver how fast to drive the display. 42MHz (**42000000**) is a good place to start but if your screen is acting funny, try taking it down to 16MHz (**16000000**) especially if you're doing something like using a GPIO extender to put the screen away from the Pi.

Save the file. Now we'll just reboot to let it all sink in.

sudo shutdown -h now (if you don't have the TFT installed, shutdown, place the TFT on the Pi and re-power)

or

sudo reboot (if you have the TFT plate installed already)

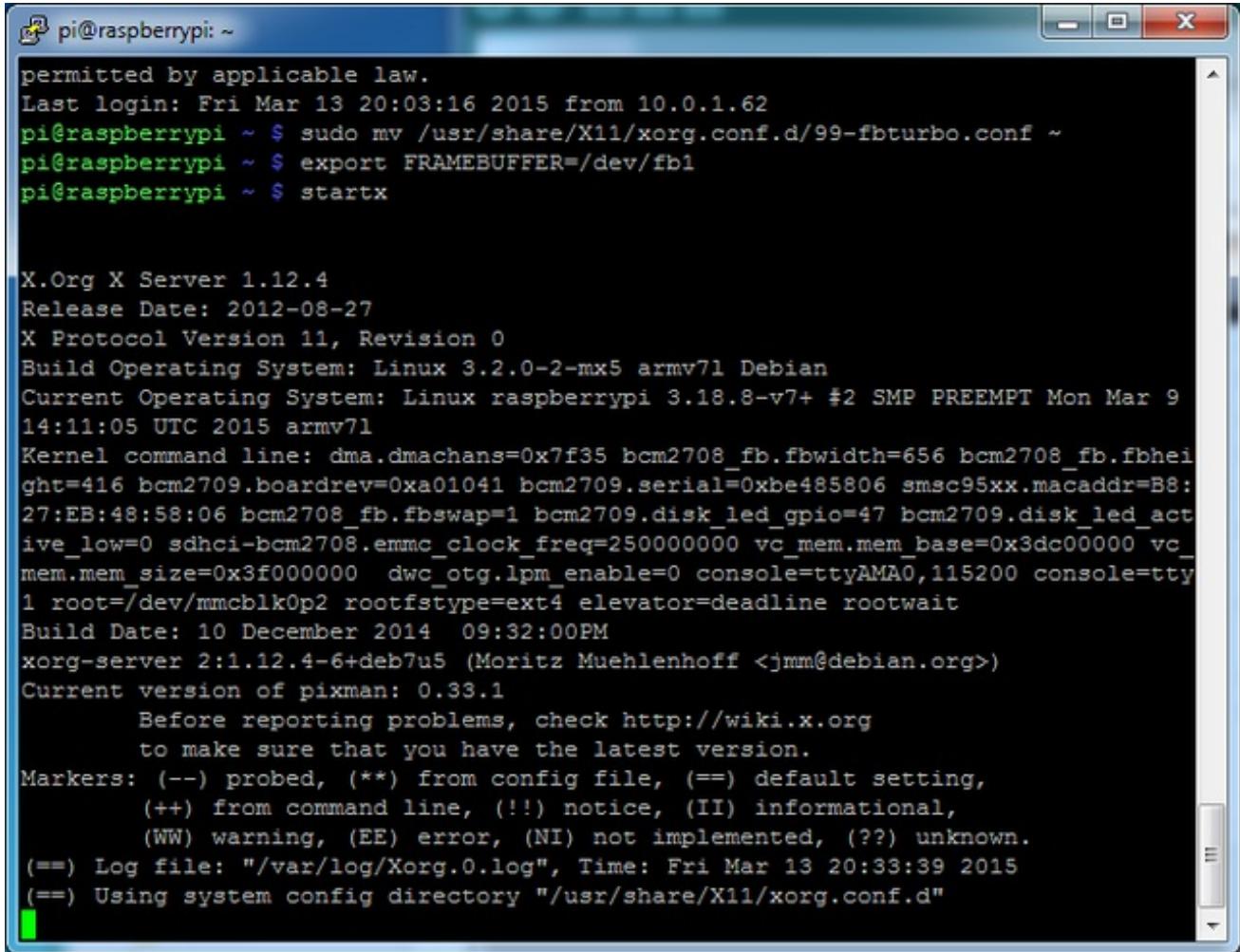
When the Pi restarts, the attached PiTFT should start out all white and then turn black. That means the kernel found the display and cleared the screen. If the screen did not turn black, that means that

likely there's something up with your connection or kernel install. Solder anything that needs resoldering!

Now that you're rebooted, log back in on the console/TV/SSH. There's nothing displayed on the screen yet, we'll do a test to make sure everything is perfect first!

Run the following commands to startx on the **/dev/fb1** framebuffer, a.k.a PiTFT screen:

```
sudo mv /usr/share/X11/xorg.conf.d/99-fbturbo.conf ~  
export FRAMEBUFFER=/dev/fb1  
startx
```



The screenshot shows a terminal window titled "pi@raspberrypi: ~". The window contains the following text:

```
permitted by applicable law.  
Last login: Fri Mar 13 20:03:16 2015 from 10.0.1.62  
pi@raspberrypi ~ $ sudo mv /usr/share/X11/xorg.conf.d/99-fbturbo.conf ~  
pi@raspberrypi ~ $ export FRAMEBUFFER=/dev/fb1  
pi@raspberrypi ~ $ startx  
  
X.Org X Server 1.12.4  
Release Date: 2012-08-27  
X Protocol Version 11, Revision 0  
Build Operating System: Linux 3.2.0-2-mx5 armv7l Debian  
Current Operating System: Linux raspberrypi 3.18.8-v7+ #2 SMP PREEMPT Mon Mar 9  
14:11:05 UTC 2015 armv7l  
Kernel command line: dma.dmachans=0x7f35 bcm2708_fb.fbwidth=656 bcm2708_fb.fbhei  
ght=416 bcm2709.boardrev=0xa01041 bcm2709.serial=0xbe485806 smsc95xx.macaddr=B8:  
27:EB:48:58:06 bcm2708_fb.fbswap=1 bcm2709.disk_led_gpio=47 bcm2709.disk_led_act  
ive_low=0 sdhci-bcm2708.emmc_clock_freq=250000000 vc_mem.mem_base=0x3dc00000 vc_  
mem.mem_size=0x3f000000 dwc_otg.lpm_enable=0 console=ttyAMA0,115200 console=tty  
1 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline rootwait  
Build Date: 10 December 2014 09:32:00PM  
xorg-server 2:1.12.4-6+deb7u5 (Moritz Muehlenhoff <jmm@debian.org>)  
Current version of pixman: 0.33.1  
Before reporting problems, check http://wiki.x.org  
to make sure that you have the latest version.  
Markers: (--) probed, (**) from config file, (==) default setting,  
        (++) from command line, (!!) notice, (II) informational,  
        (WW) warning, (EE) error, (NI) not implemented, (??) unknown.  
(==) Log file: "/var/log/Xorg.0.log", Time: Fri Mar 13 20:33:39 2015  
(==) Using system config directory "/usr/share/X11/xorg.conf.d"
```

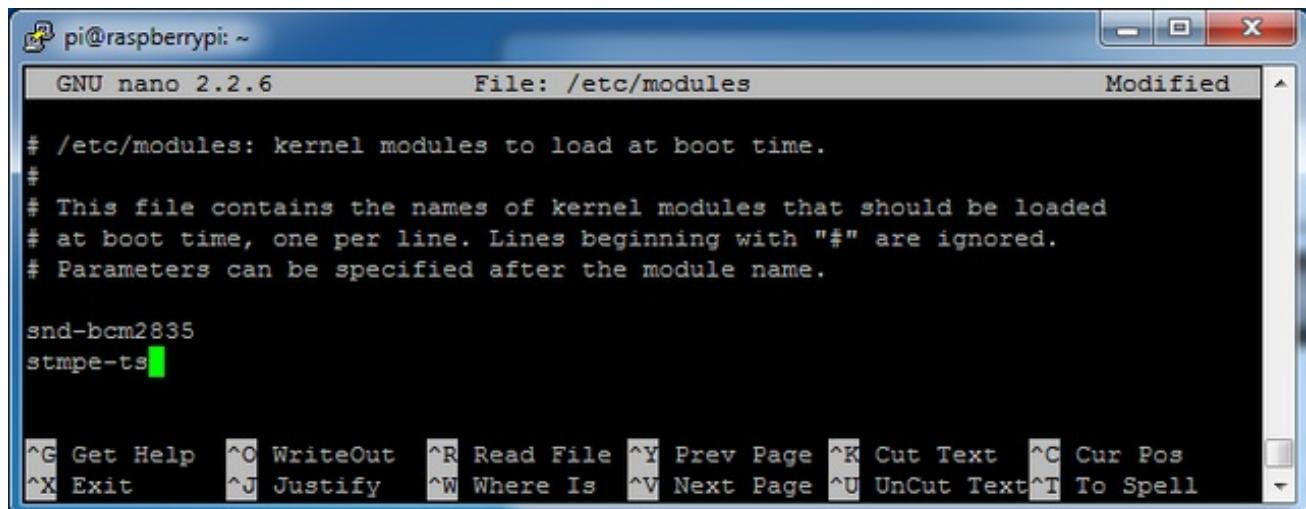
You should see the Pi desktop show up on the TFT! Congrats, you've completed the first test perfectly.

Hit Control-C in the console to quit the X server so we can continue configuration

Next up we'll add support for the touch screen automatically on boot. Edit the module list with

sudo nano /etc/modules

and add **stmpe-ts** on a line at the end

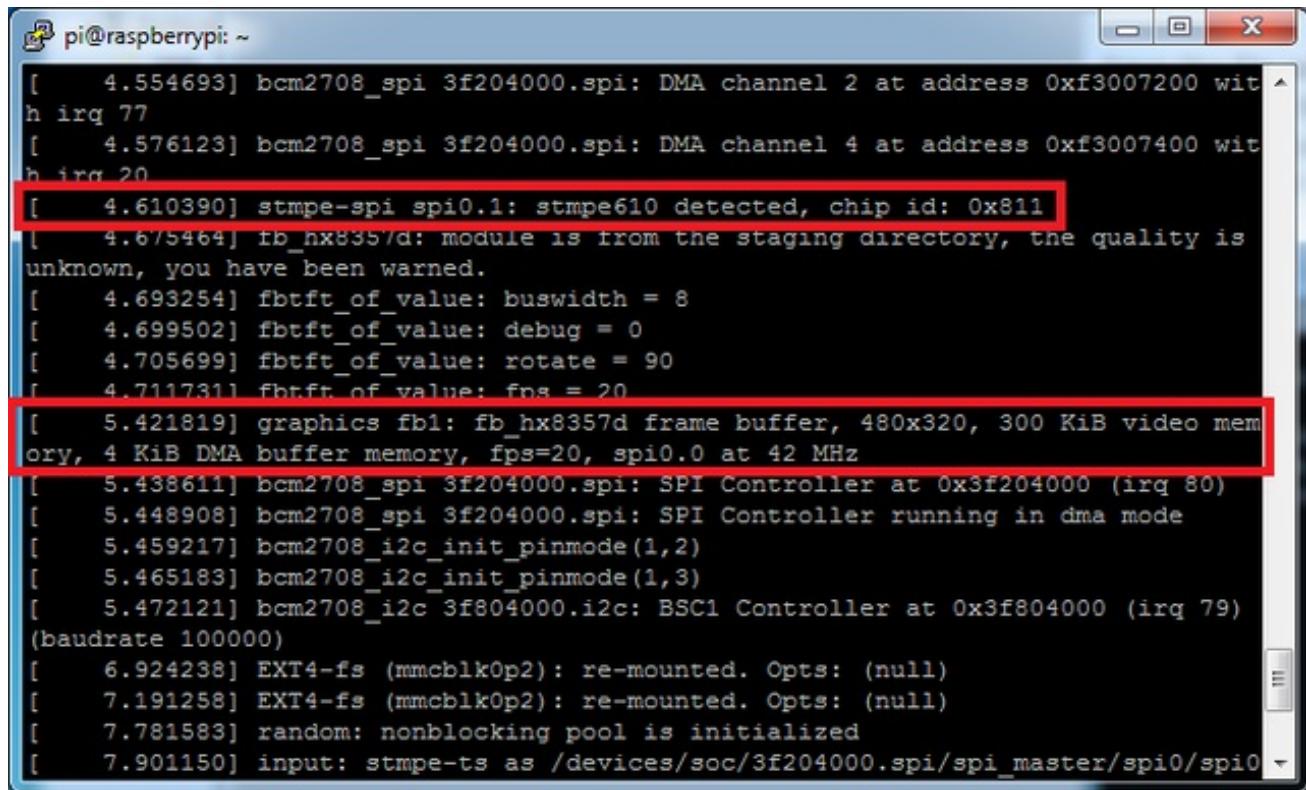


```
pi@raspberrypi: ~
GNU nano 2.2.6          File: /etc/modules          Modified
#
# /etc/modules: kernel modules to load at boot time.
#
# This file contains the names of kernel modules that should be loaded
# at boot time, one per line. Lines beginning with "#" are ignored.
# Parameters can be specified after the module name.

snd-bcm2835
stmpe-ts
```

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell

Save the file and reboot the Pi with **sudo reboot** and look at the console output (or run **dmesg** in the console window after logging in) you will see the modules install. Look in particular for the STMPE610 detection and the HX5387D screen frequency as highlighted here



```
[ 4.554693] bcm2708_spi 3f204000.spi: DMA channel 2 at address 0xf3007200 with irq 77
[ 4.576123] bcm2708_spi 3f204000.spi: DMA channel 4 at address 0xf3007400 with irq 20
[ 4.610390] stmpe-spi spi0.1: stmpe610 detected, chip id: 0x811
[ 4.675464] fb_hx8357d: module is from the staging directory, the quality is unknown, you have been warned.
[ 4.693254] fbftf_of_value: buswidth = 8
[ 4.699502] fbftf_of_value: debug = 0
[ 4.705699] fbftf_of_value: rotate = 90
[ 4.711731] fbftf_of_value: fps = 20
[ 5.421819] graphics fb1: fb_hx8357d frame buffer, 480x320, 300 KiB video memory, 4 KiB DMA buffer memory, fps=20, spi0.0 at 42 MHz
[ 5.438611] bcm2708_spi 3f204000.spi: SPI Controller at 0x3f204000 (irq 80)
[ 5.448908] bcm2708_spi 3f204000.spi: SPI Controller running in dma mode
[ 5.459217] bcm2708_i2c_init_pinmode(1,2)
[ 5.465183] bcm2708_i2c_init_pinmode(1,3)
[ 5.472121] bcm2708_i2c 3f804000.i2c: BSC1 Controller at 0x3f804000 (irq 79)
(baudrate 100000)
[ 6.924238] EXT4-fs (mmcblk0p2): re-mounted. Opts: (null)
[ 7.191258] EXT4-fs (mmcblk0p2): re-mounted. Opts: (null)
[ 7.781583] random: nonblocking pool is initialized
[ 7.901150] input: stmpe-ts as /devices/soc/3f204000.spi/spi_master/spi0/spi0
```

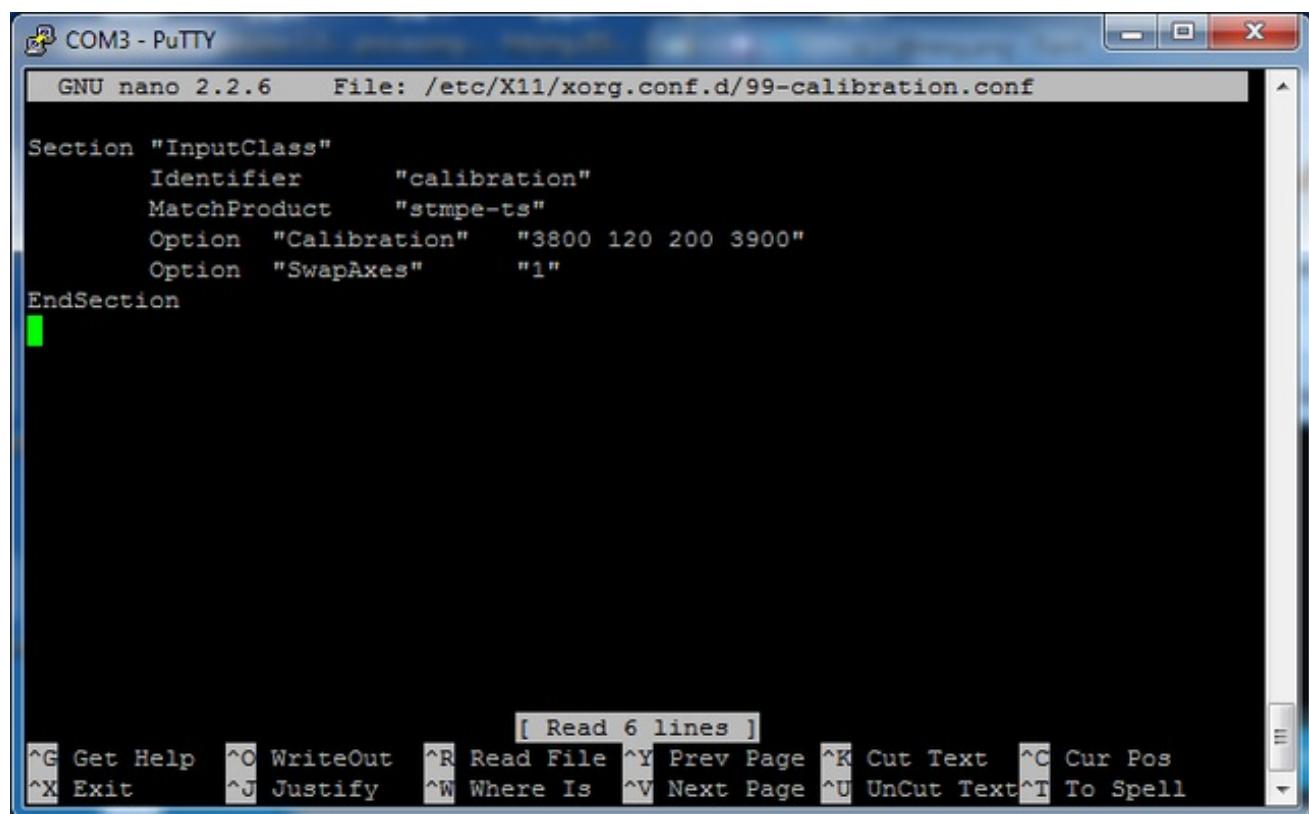
We can set up the touchscreen for **rotate=90** configuration by doing the following (for more delicate calibration or for other rotate=XX values, see the next section)

Create the directory and new calibration configuration file:

```
sudo mkdir /etc/X11/xorg.conf.d  
sudo nano /etc/X11/xorg.conf.d/99-calibration.conf
```

and enter in the following lines, then save.

```
Section "InputClass"  
    Identifier      "calibration"  
    MatchProduct    "stmpe-ts"  
    Option          "Calibration"  "3800 120 200 3900"  
    Option          "SwapAxes"     "1"  
EndSection
```



You can now try to run X again with

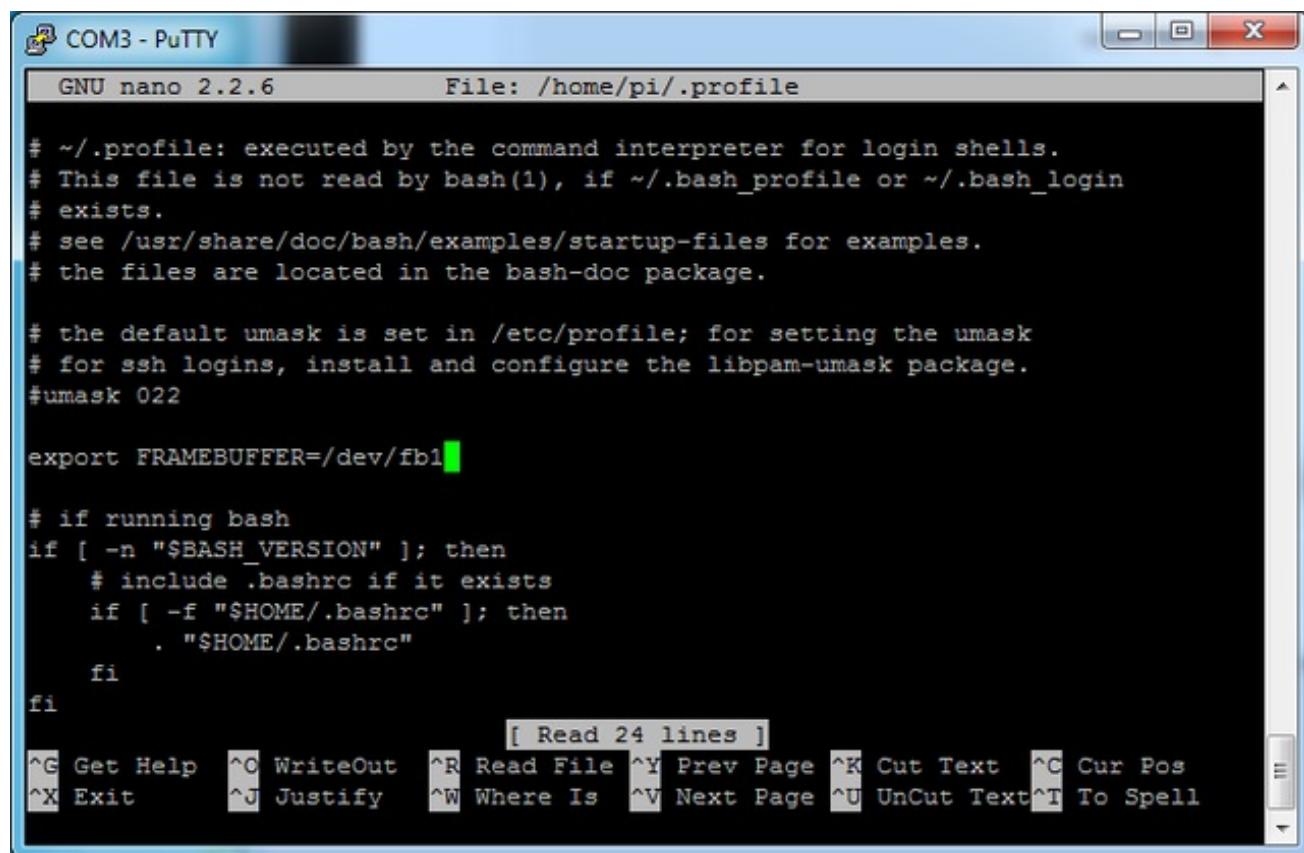
```
FRAMEBUFFER=/dev/fb1 startx
```

Type Control-C to quit X

If you don't ever want to have to type FRAMEBUFFER=/dev/fb1 before startx, you can make it a default state by editing your profile file: **sudo nano ~/.profile** and adding

```
export FRAMEBUFFER=/dev/fb1
```

near the top and saving the file. Then reboot to reload the profile file. It will now always assume you want to use /dev/fb1



```
GNU nano 2.2.6          File: /home/pi/.profile

# ~/.profile: executed by the command interpreter for login shells.
# This file is not read by bash(1), if ~/.bash_profile or ~/.bash_login
# exists.
# see /usr/share/doc/bash/examples/startup-files for examples.
# the files are located in the bash-doc package.

# the default umask is set in /etc/profile; for setting the umask
# for ssh logins, install and configure the libpam-umask package.
#umask 022

export FRAMEBUFFER=/dev/fb1

# if running bash
if [ -n "$BASH_VERSION" ]; then
    # include .bashrc if it exists
    if [ -f "$HOME/.bashrc" ]; then
        . "$HOME/.bashrc"
    fi
fi
[ Read 24 lines ]
^G Get Help  ^C WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is   ^V Next Page  ^U UnCut Text^T To Spell
```

Detailed Calibration

If you've grabbed our Easy Install image, or use the 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



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.

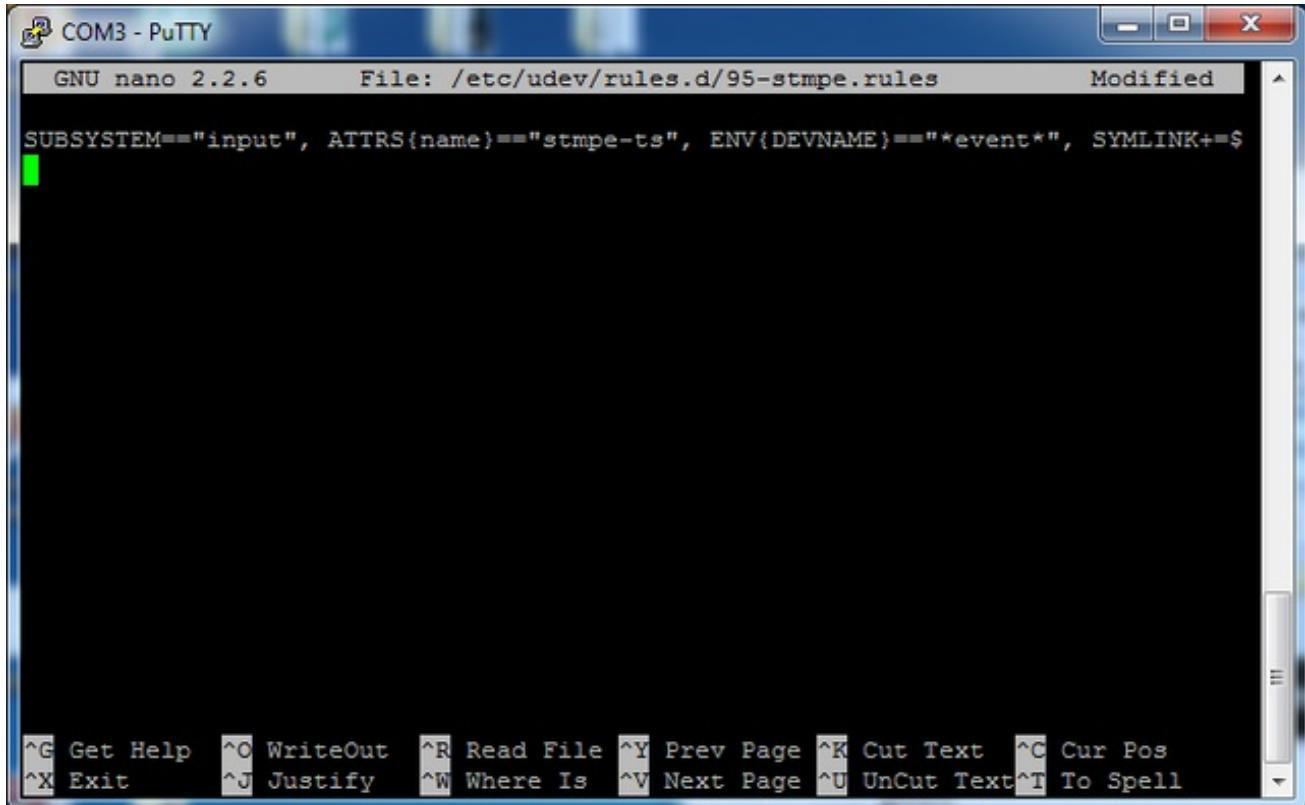
Before we start, we'll make a **udev** rule for the touchscreen. That's because the **eventX** name of the device will change a lot and its annoying to figure out what its called depending on whether you have a keyboard or other mouse installed.

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"
```



```
GNU nano 2.2.6      File: /etc/udev/rules.d/95-stmpe.rules      Modified  
SUBSYSTEM==\"input\", ATTRS{name}==\"stmpe-ts\", ENV{DEVNAME}==\"*event\", SYMLINK+=\"input/touchscreen\"
```

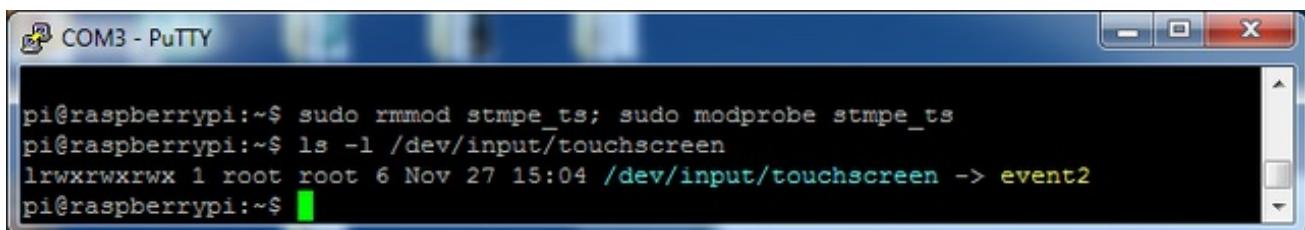
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell

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



```
pi@raspberrypi:~$ sudo rmmod stmpe_ts; sudo modprobe stmpe_ts  
pi@raspberrypi:~$ ls -l /dev/input/touchscreen  
lrwxrwxrwx 1 root root 6 Nov 27 15:04 /dev/input/touchscreen -> event2  
pi@raspberrypi:~$
```

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
```

```
pi@raspberrypi:~$ sudo apt-get install evtest tslib libts-bin
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'libts-0.0-0' instead of 'tslib'
libts-0.0-0 is already the newest version.
The following NEW packages will be installed:
  evtest libts-bin
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 0 B/55.0 kB of archives.
After this operation, 219 kB of additional disk space will be used.
Do you want to continue [Y/n]? Y
Selecting previously unselected package libts-bin.
(Reading database ... 62285 files and directories currently installed.)
Unpacking libts-bin (from .../libts-bin_1.0-11_armhf.deb) ...
Selecting previously unselected package evtest.
Unpacking evtest (from .../evtest_1%3a1.30-1_armhf.deb) ...
Processing triggers for man-db ...
Setting up libts-bin (1.0-11) ...
Setting up evtest (1:1.30-1) ...
pi@raspberrypi:~$
```

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.

```
COM3 - PuTTY
pi@raspberrypi:~$ sudo evtest /dev/input/touchscreen
Input driver version is 1.0.1
Input device ID: bus 0x18 vendor 0x0 product 0x0 version 0x0
Input device name: "stmpe-ts"
Supported events:
  Event type 0 (EV_SYN)
  Event type 1 (EV_KEY)
    Event code 330 (BTN_TOUCH)
  Event type 3 (EV_ABS)
    Event code 0 (ABS_X)
      Value        0
      Min         0
      Max       4095
    Event code 1 (ABS_Y)
      Value        0
      Min         0
      Max       4095
    Event code 24 (ABS_PRESSURE)
      Value        0
      Min         0
      Max       255
Properties:
Testing ... (interrupt to exit)
```

```
COM3 - PuTTY
Event: time 1385565357.639692, type 3 (EV_ABS), code 24 (ABS_PRESSURE), value 149
Event: time 1385565357.639699, ----- SYN_REPORT -----
Event: time 1385565357.645965, type 3 (EV_ABS), code 0 (ABS_X), value 1580
Event: time 1385565357.645973, type 3 (EV_ABS), code 1 (ABS_Y), value 1846
Event: time 1385565357.645980, ----- SYN_REPORT -----
Event: time 1385565357.652293, type 3 (EV_ABS), code 0 (ABS_X), value 1634
Event: time 1385565357.652301, type 3 (EV_ABS), code 1 (ABS_Y), value 1864
Event: time 1385565357.652305, type 3 (EV_ABS), code 24 (ABS_PRESSURE), value 143
Event: time 1385565357.652310, ----- SYN_REPORT -----
Event: time 1385565357.658614, type 3 (EV_ABS), code 0 (ABS_X), value 1658
Event: time 1385565357.658622, type 3 (EV_ABS), code 1 (ABS_Y), value 1877
Event: time 1385565357.658626, type 3 (EV_ABS), code 24 (ABS_PRESSURE), value 139
Event: time 1385565357.658631, ----- SYN_REPORT -----
Event: time 1385565357.664919, type 3 (EV_ABS), code 0 (ABS_X), value 1748
Event: time 1385565357.664928, type 3 (EV_ABS), code 1 (ABS_Y), value 1888
Event: time 1385565357.664935, ----- SYN_REPORT -----
Event: time 1385565357.671199, type 3 (EV_ABS), code 0 (ABS_X), value 1778
Event: time 1385565357.671207, type 3 (EV_ABS), code 1 (ABS_Y), value 1895
Event: time 1385565357.671211, type 3 (EV_ABS), code 24 (ABS_PRESSURE), value 134
Event: time 1385565357.671216, ----- SYN_REPORT -----
Event: time 1385565357.698600, type 3 (EV_ABS), code 24 (ABS_PRESSURE), value 0
Event: time 1385565357.698607, type 1 (EV_KEY), code 330 (BTN_TOUCH), value 0
Event: time 1385565357.698610, ----- SYN_REPORT -----
```

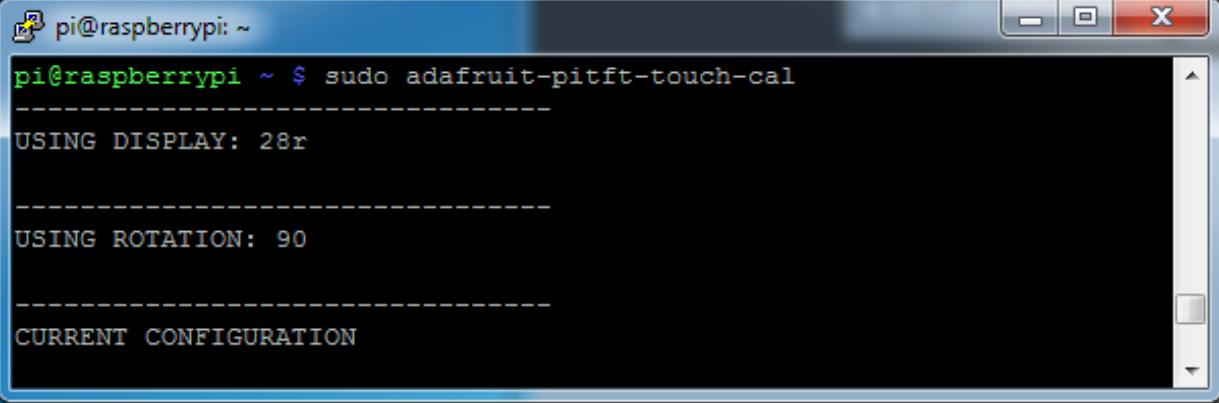
AutoMagic Calibration Script

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 run a small Python script which will automatically set a default touchscreen calibration based on the screen orientation.

This helper is automatically installed for you but if you'd like you can check it out here on github (<http://adafru.it/elu>)

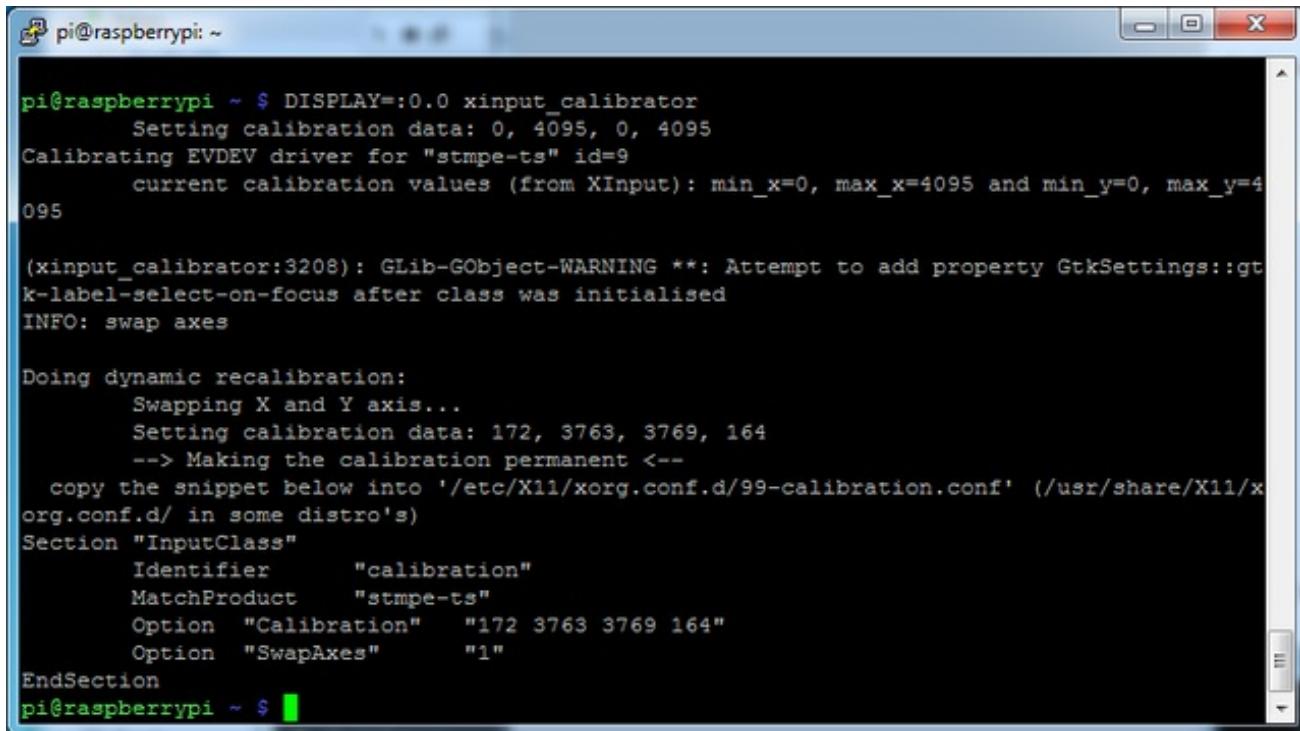
Run it at the command line with **sudo adafruit-pitft-touch-cal**

it will try to figure out what display you have installed and the rotation it's set up for



```
pi@raspberrypi: ~
pi@raspberrypi ~ $ sudo adafruit-pitft-touch-cal
-----
USING DISPLAY: 28r
-----
USING ROTATION: 90
-----
CURRENT CONFIGURATION
```

By default the script will attempt to read the screen orientation by examining the PiTFT module configuration with modprobe. If the script can read the orientation it will print out the current orientation, the current touchscreen calibration values, and the new touchscreen calibration values based on the current orientation. Before updating the calibration the script will ask you to confirm that you'd like to make the change. Press **y** and enter to confirm.



```
pi@raspberrypi ~ $ DISPLAY=:0.0 xinput_calibrator
      Setting calibration data: 0, 4095, 0, 4095
Calibrating EVDEV driver for "stmpe-ts" id=9
      current calibration values (from XInput): min_x=0, max_x=4095 and min_y=0, max_y=4095

(xinput_calibrator:3208): GLib-GObject-WARNING **: Attempt to add property GtkSettings::gtk-label-select-on-focus after class was initialised
INFO: swap axes

Doing dynamic recalibration:
      Swapping X and Y axis...
      Setting calibration data: 172, 3763, 3769, 164
      --> Making the calibration permanent <--
      copy the snippet below into '/etc/X11/xorg.conf.d/99-calibration.conf' (/usr/share/X11/xorg.conf.d/ in some distro's)
Section "InputClass"
      Identifier      "calibration"
      MatchProduct    "stmpe-ts"
      Option   "Calibration"    "172 3763 3769 164"
      Option   "SwapAxes"      "1"
EndSection
pi@raspberrypi ~ $
```

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 you rotate the display 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.

```
pi@raspberrypi:~$ sudo TSLIB_FBDEVICE=/dev/fb1 TSLIB_TSDEVICE=/dev/input/touchscreen ts_calibrate
xres = 320, yres = 240
Took 43 samples...
Top left : X = 989 Y = 3192
Took 56 samples...
Top right : X = 1049 Y = 674
Took 56 samples...
Bot right : X = 3191 Y = 695
Took 37 samples...
Bot left : X = 3167 Y = 3168
Took 41 samples...
Center : X = 2095 Y = 1913
330.127167 -0.000068 -0.088149
-18.096893 0.064811 0.001094
Calibration constants: 21635214 -4 -5776 -1185998 4247 71 65536
pi@raspberrypi:~$
```

A screenshot of a PuTTY terminal window titled "COM3 - PuTTY". The window shows the output of a command to calibrate a touchscreen. The output includes the resolution (xres = 320, yres = 240), the number of samples taken for each corner and center, and the resulting calibration constants.

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 **xinput_calibrator**

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

Normally you'd have to compile it but we have a ready to go package for you so run:

```
 wget http://adafruit-download.s3.amazonaws.com/xinput-calibrator_0.7.5-1_armhf.deb  
 sudo dpkg -i -B xinput-calibrator_0.7.5-1_armhf.deb
```

Before you start the xinput_calibrator you will need to delete the old calibration data so run

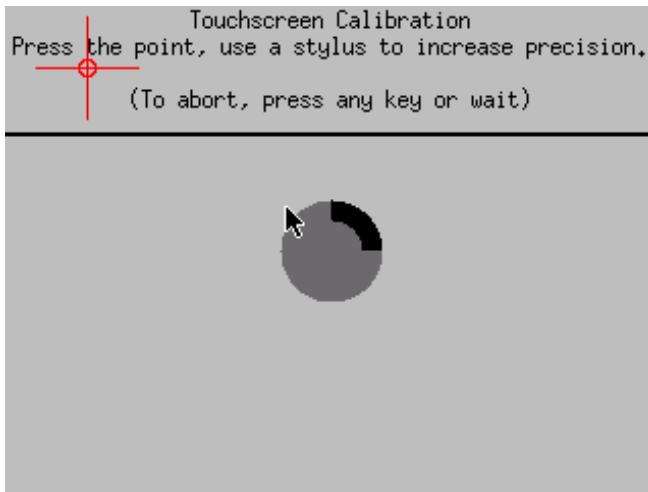
```
 sudo rm /etc/X11/xorg.conf.d/99-calibration.conf
```

Before running **startx** and the calibrator - otherwise it gets really confused!

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

```
 FRAMEBUFFER=/dev/fb1 startx &  
 DISPLAY=:0.0 xinput_calibrator
```

Follow the directions on screen



Once complete you'll get something like:

```
pi@raspberrypi:~$ DISPLAY=:0.0 xinput_calibrator
Calibrating EVDEV driver for "stmpe-ts" id=8
    current calibration values (from XInput): min_x=0, max_x=4095 and min_y=
0, max_y=4095

Doing dynamic recalibration:
    Swapping X and Y axis...
    Setting new calibration data: 119, 3736, 3850, 174

--> Making the calibration permanent <--
    copy the snippet below into '/etc/X11/xorg.conf.d/99-calibration.conf'
Section "InputClass"
    Identifier      "calibration"
    MatchProduct    "stmpe-ts"
    Option   "Calibration"    "119 3736 3850 174"
    Option   "SwapAxes"      "1"
EndSection
```

Run **sudo nano /etc/X11/xorg.conf.d/99-calibration.conf** and copy the

```
Section "InputClass"
    Identifier      "calibration"
    MatchProduct    "stmpe-ts"
    Option   "Calibration"    "119 3736 3850 174"
#    Option   "SwapAxes"      "1"
EndSection
```

or whatever you got, into there. You can quit X if you want by typing **fg** to bring that command into the foreground, and then Control-C to quit.

Depending on the 'rotation' of the screen, when you do this calibration, you may need to comment out the SwapAxes part with a # and/or swap the numbers around so looks like:

- Option "Calibration" "119 3736 3850 174"

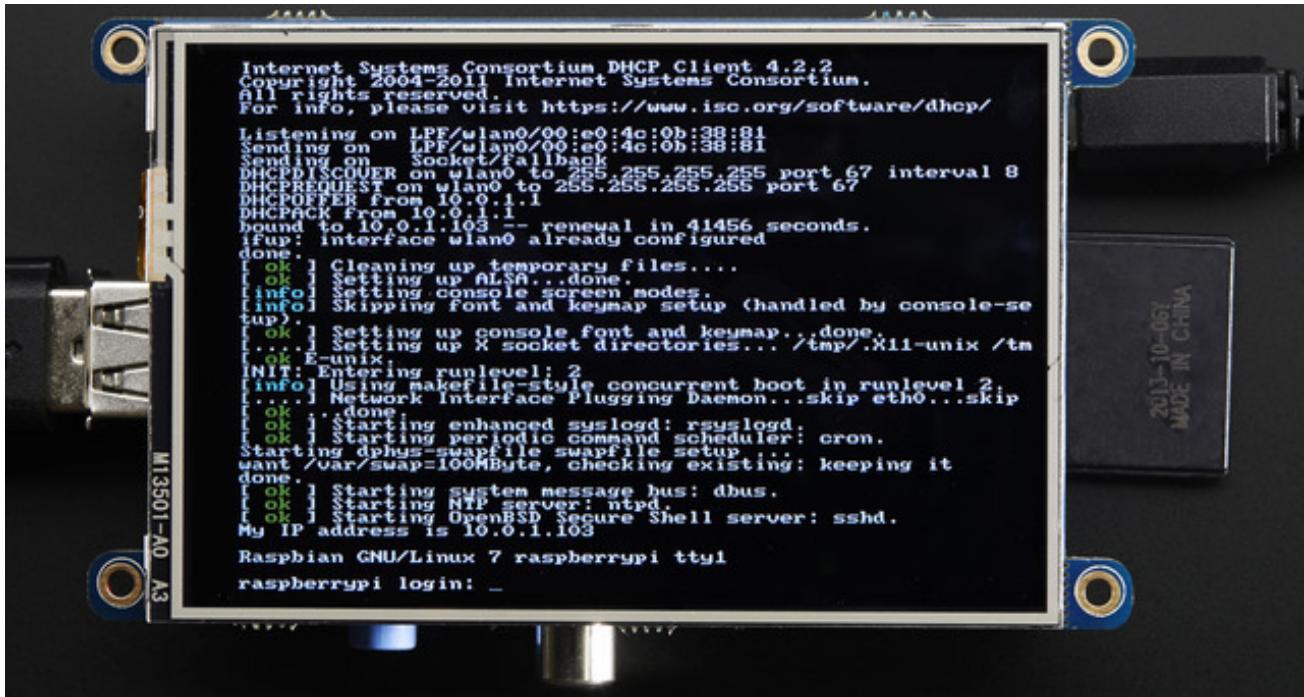
to

- Option "Calibration" "3736 119 174 3850"

Your touchscreen is now super calibrated, hurrah!

Detailed Console Use

If you've grabbed our Easy Install image, or use the 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 40 x 60 of text. For more details, check out <https://github.com/notro/fbtft/wiki/Boot-console> (<http://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**

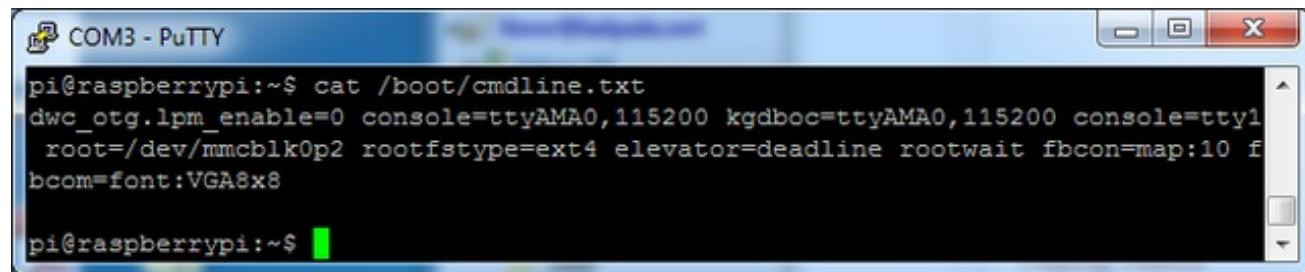
```
sudo nano /boot/cmdline.txt
```

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.

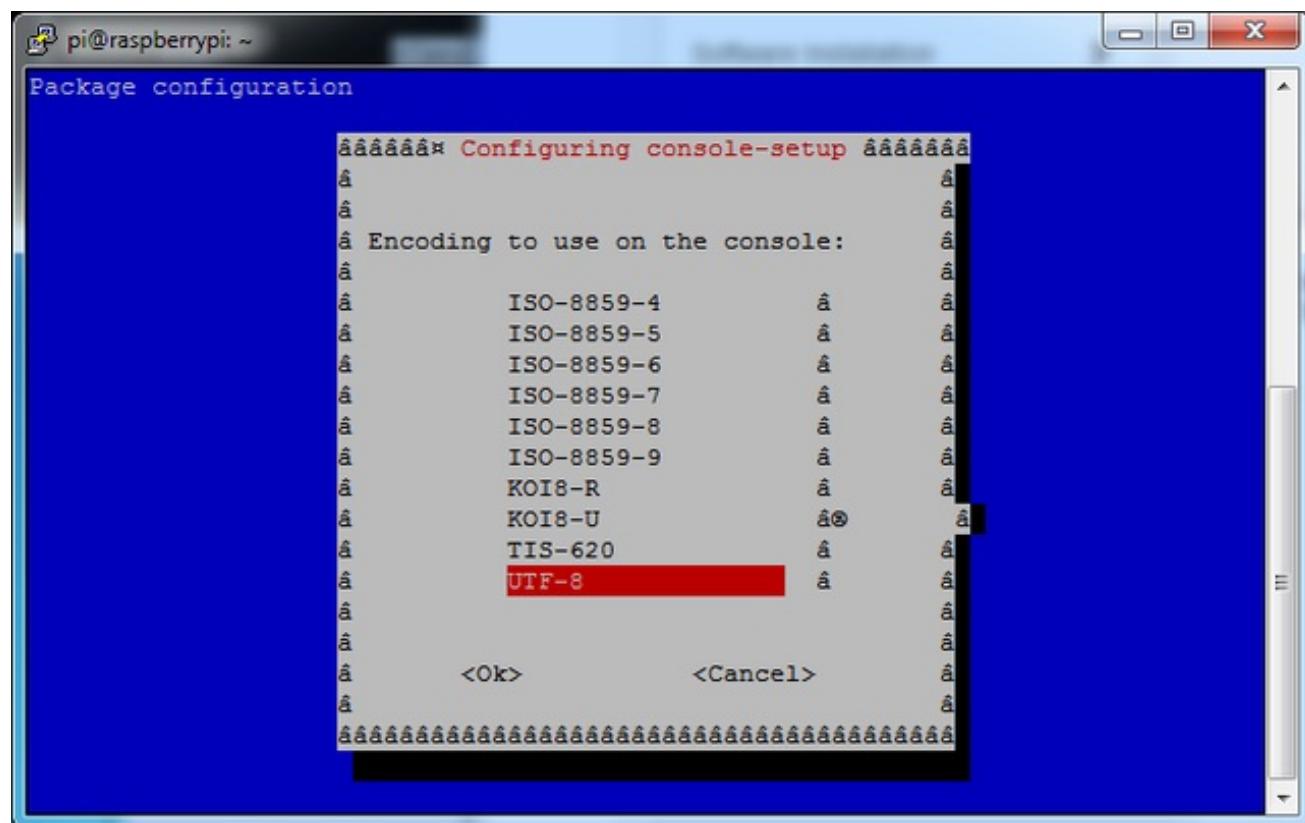
Note that the kernel has to load up the display driver module before it can display anything on it so you won't get the rainbow screen, a NooBs prompt, or a big chunk of the kernel details since the module is loaded fairly late in the boot process.

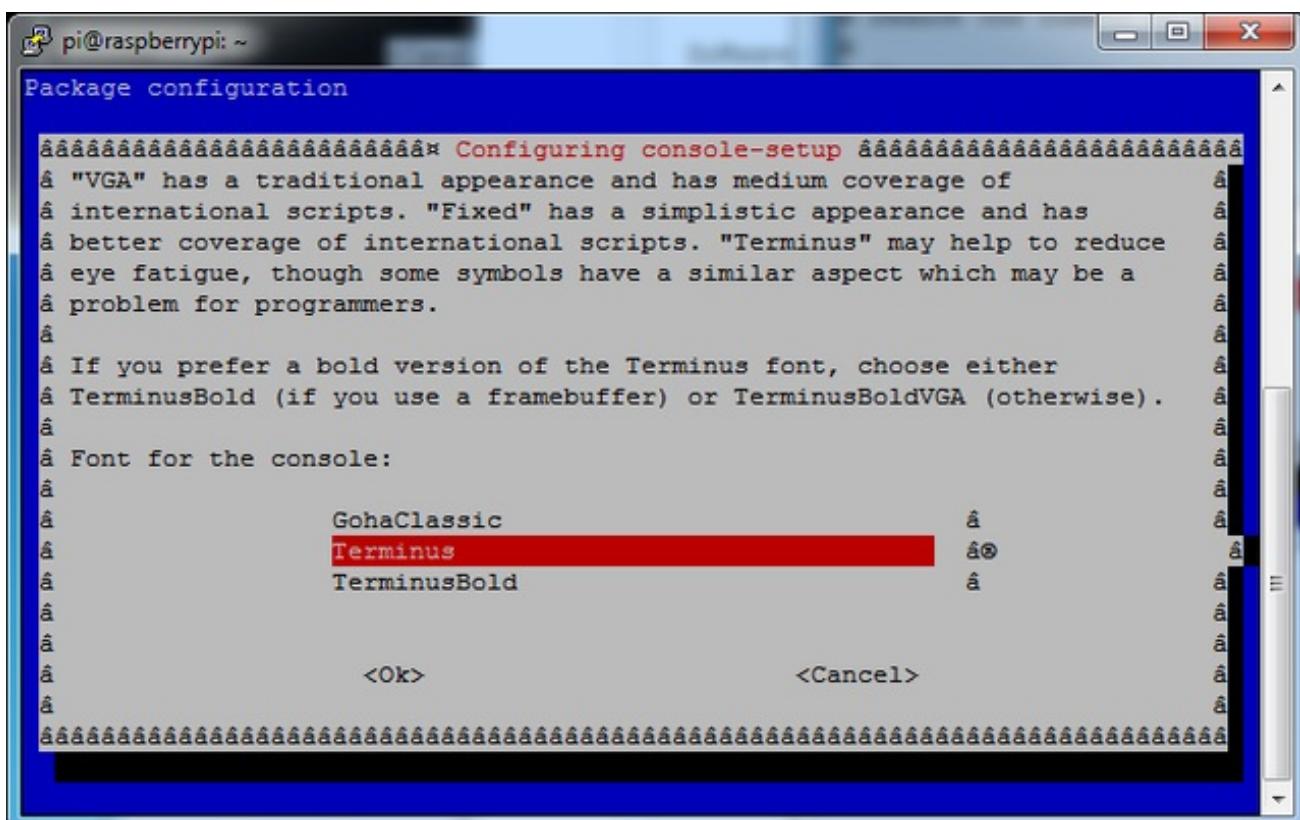
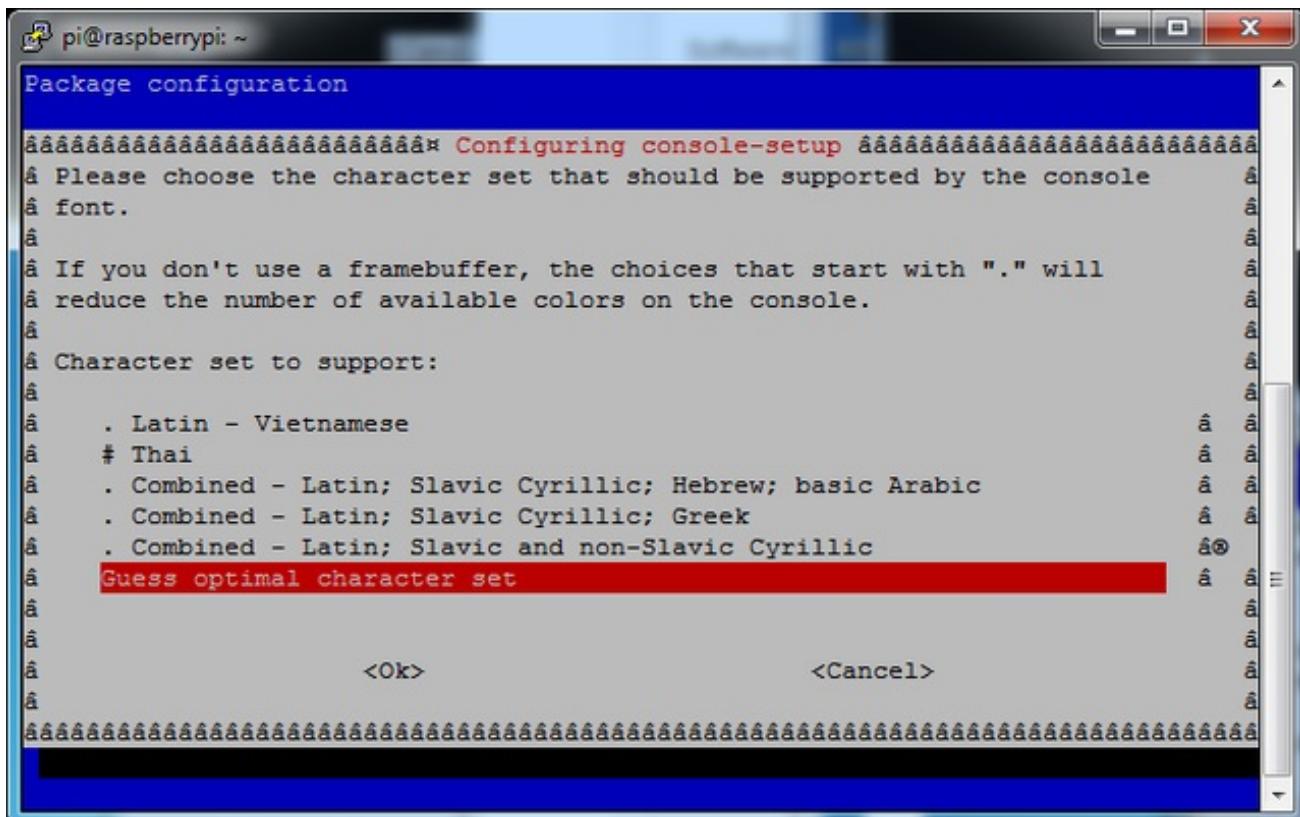


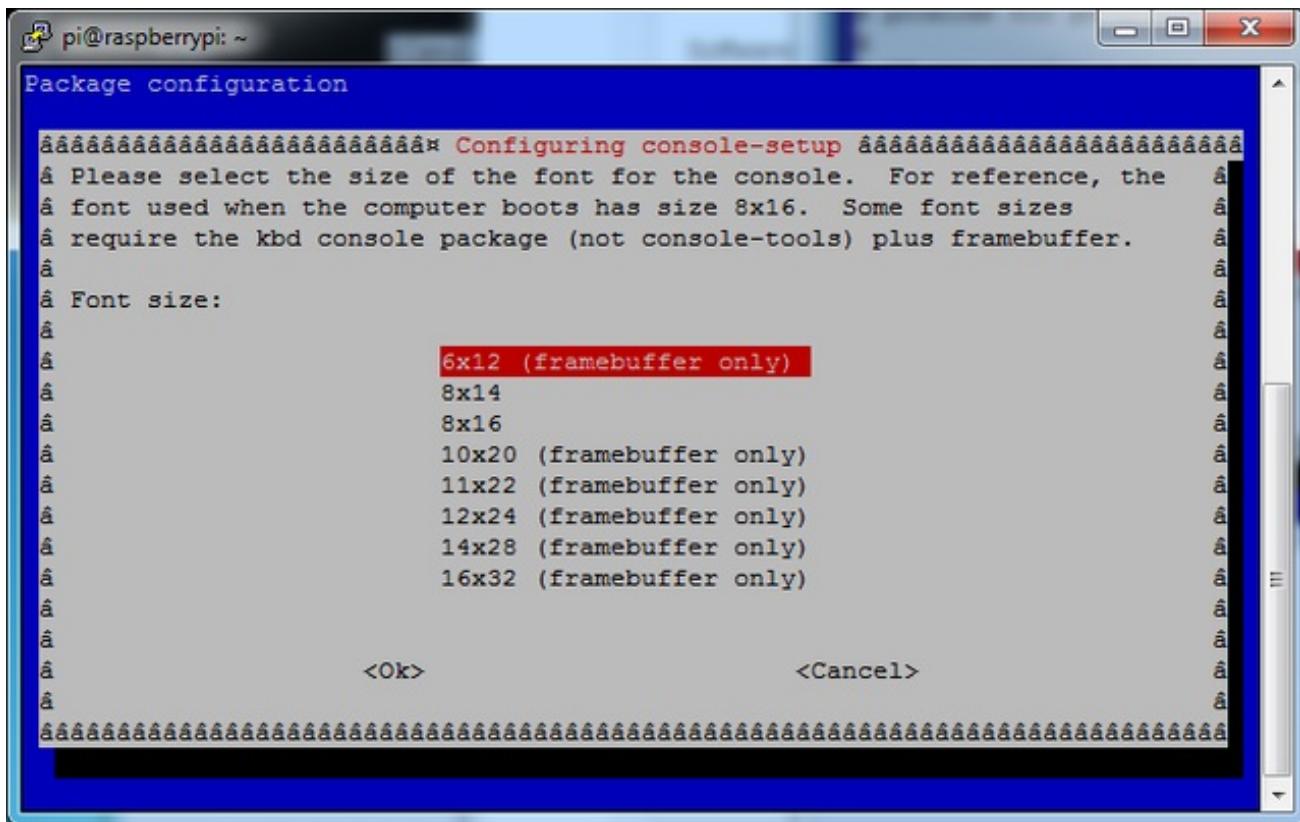
```
pi@raspberrypi:~$ cat /boot/cmdline.txt
dwc_otg.lpm_enable=0 console=ttyAMA0,115200 kgdboc=ttyAMA0,115200 console=tty1
root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline rootwait fbcon=map:10 f
bcom=font:VGA8x8

pi@raspberrypi:~$
```

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





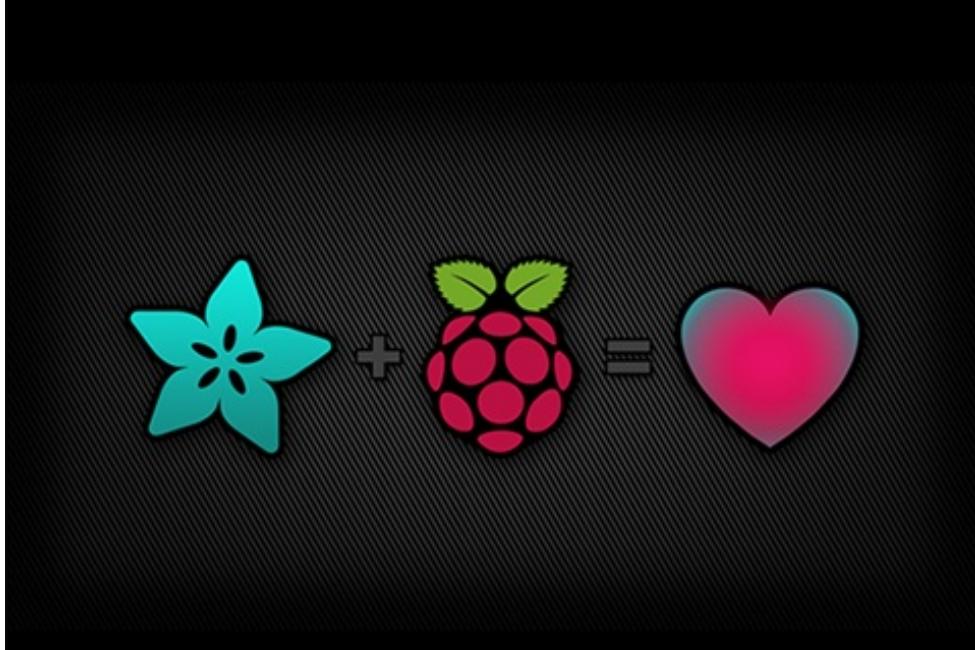


Turn off Console Blanking

You may notice the console goes black after 30 minutes, this is a sort of 'power saving' or 'screensaver' feature. You can disable this by editing `/etc/kbd/config` and setting the blank time to 0 (which turns it off)

```
BLANK_TIME=30
```

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

```
pi@raspberrypi:~$ sudo apt-get install fbi
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  imagemagick
The following NEW packages will be installed:
  fbi
0 upgraded, 1 newly installed, 0 to remove and 52 not upgraded.
Need to get 59.7 kB of archives.
After this operation, 157 kB of additional disk space will be used.
Get:1 http://mirrordirector.raspbian.org/raspbian/ wheezy/main fbi armhf 2.07-10 [59.7 kB]
Fetched 59.7 kB in 1s (40.0 kB/s)
Selecting previously unselected package fbi.
(Reading database ... 64758 files and directories currently installed.)
Unpacking fbi (from .../archives/fbi_2.07-10_armhf.deb) ...
Processing triggers for mime-support ...
Processing triggers for man-db ...
Setting up fbi (2.07-10) ...
pi@raspberrypi:~$
```

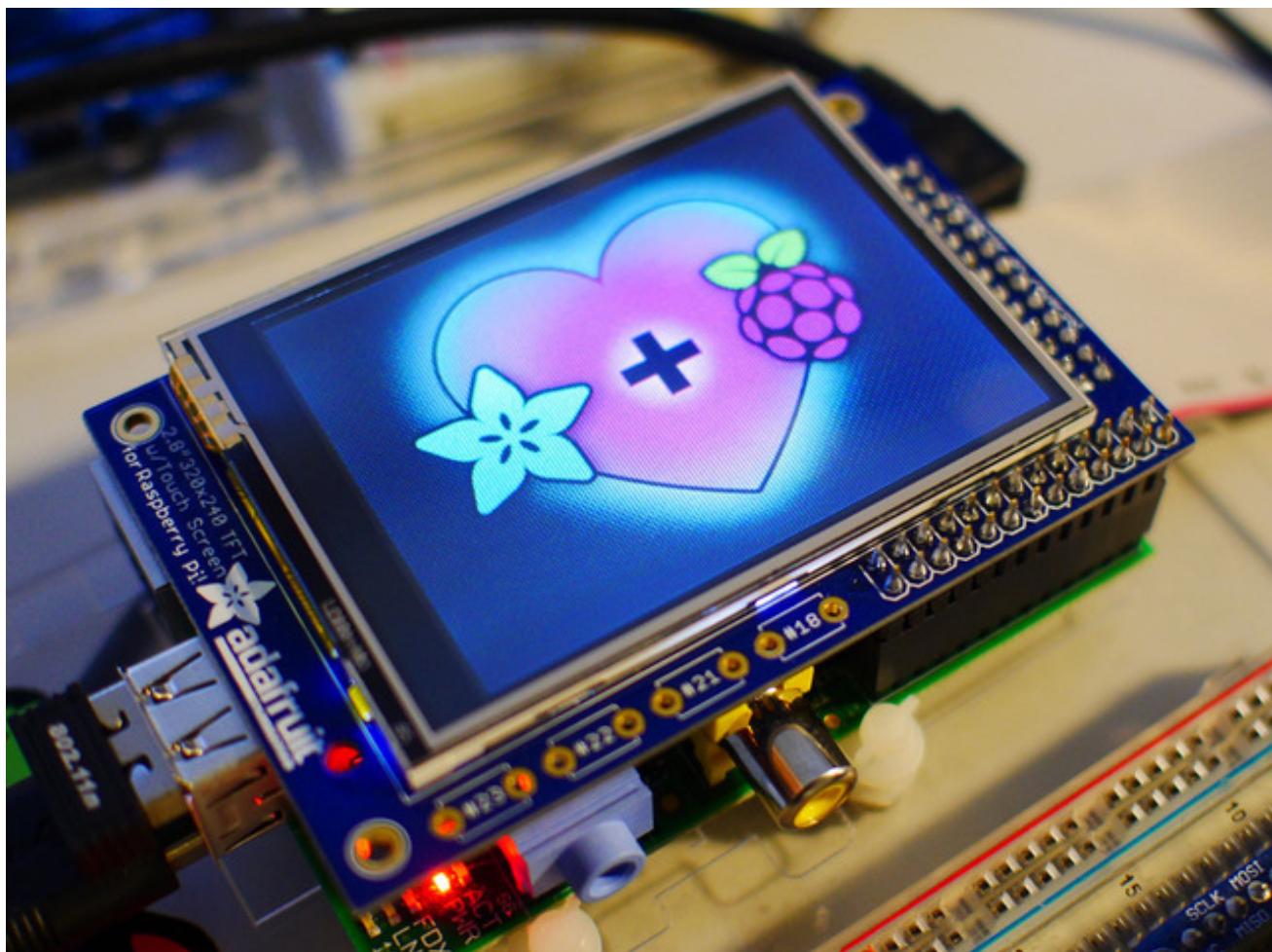
Grab our lovely wallpaper with

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

and view it with

```
sudo fbi -T 2 -d /dev/fb1 -noverbose -a adapiluv480x320.png
```

That's it!



Backlight

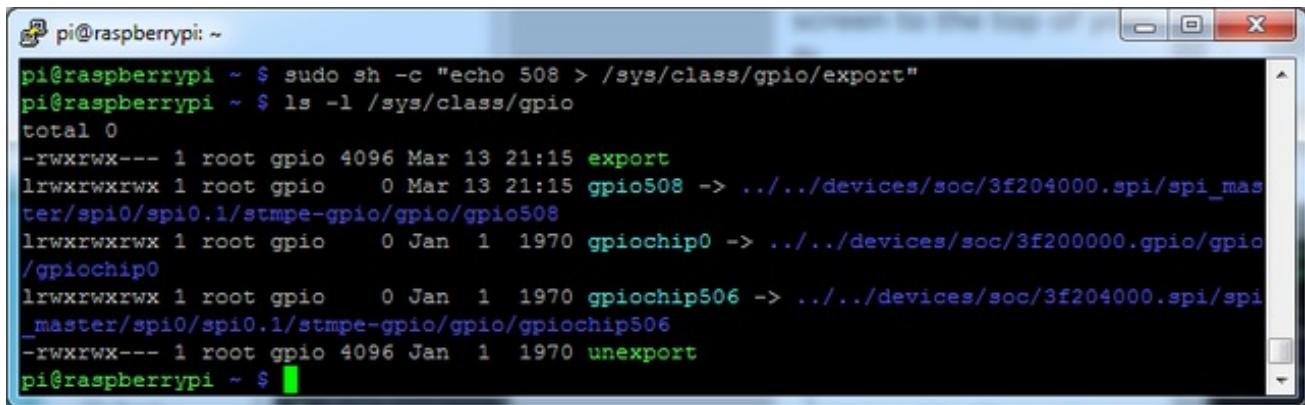
The backlight of the 3.5" display has 6 LEDs in a row, and we use a boost converter to get the 5V from the Pi up to the ~20V needed to light up all the LEDs. By default, the backlight's on...but you can control it in two ways.

On / Off Using STMPE GPIO

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

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
```



```
pi@raspberrypi: ~
pi@raspberrypi ~ $ sudo sh -c "echo 508 > /sys/class/gpio/export"
pi@raspberrypi ~ $ ls -l /sys/class/gpio
total 0
-rwxrwx--- 1 root gpio 4096 Mar 13 21:15 export
lrwxrwxrwx 1 root gpio      0 Mar 13 21:15 gpio508 -> ../../devices/soc/3f204000.spi/spi_master/spi0/spi0.1/stmpe-gpio/gpio/gpio508
lrwxrwxrwx 1 root gpio      0 Jan  1 1970 gpiochip0 -> ../../devices/soc/3f200000 gpio/gpio/gpiochip0
lrwxrwxrwx 1 root gpio      0 Jan  1 1970 gpiochip506 -> ../../devices/soc/3f204000.spi/spi_master/spi0/spi0.1/stmpe-gpio/gpio/gpiochip506
-rwxrwx--- 1 root gpio 4096 Jan  1 1970 unexport
pi@raspberrypi ~ $
```

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

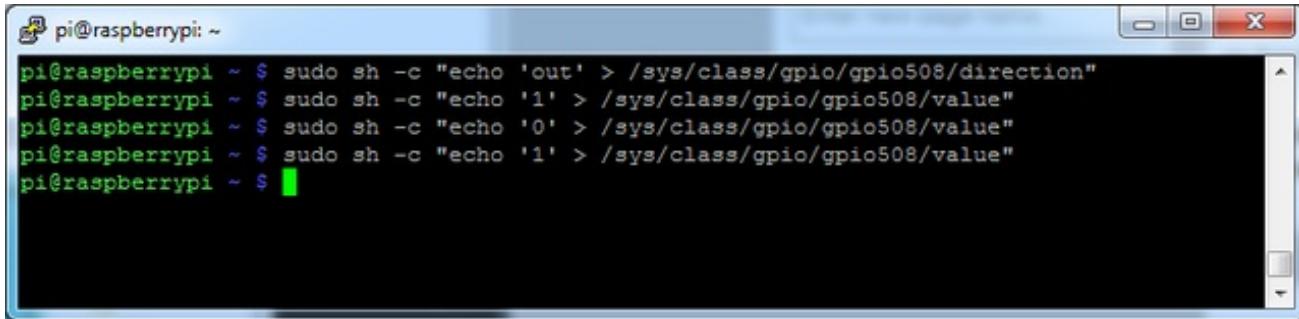
```
sudo sh -c "echo 'out' > /sys/class/gpio/gpio508/direction"
```

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 ~ $ 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 ~ $
```

PWM Backlight Control with GPIO 18

If you want more 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.

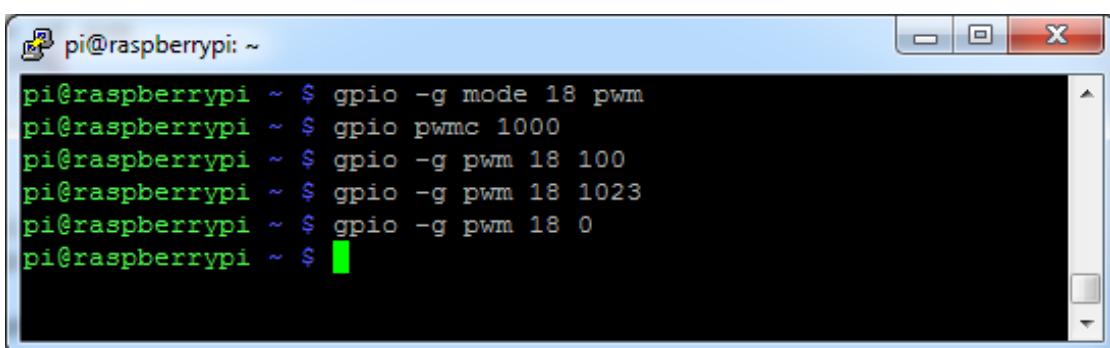
If you did the above commands, you'll need to turn off the STMPE GPIO which overrides the PWM output. You only have to run this if you set GPIO508 to an output in the previous option

```
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
```



```
pi@raspberrypi ~ $ gpio -g mode 18 pwm
pi@raspberrypi ~ $ gpio pwmc 1000
pi@raspberrypi ~ $ gpio -g pwm 18 100
pi@raspberrypi ~ $ gpio -g pwm 18 1023
pi@raspberrypi ~ $ gpio -g pwm 18 0
pi@raspberrypi ~ $
```

Try other numbers, from 0 (off) to 1023 (all the way on)!

More Tips

Making it easier to click icons in X

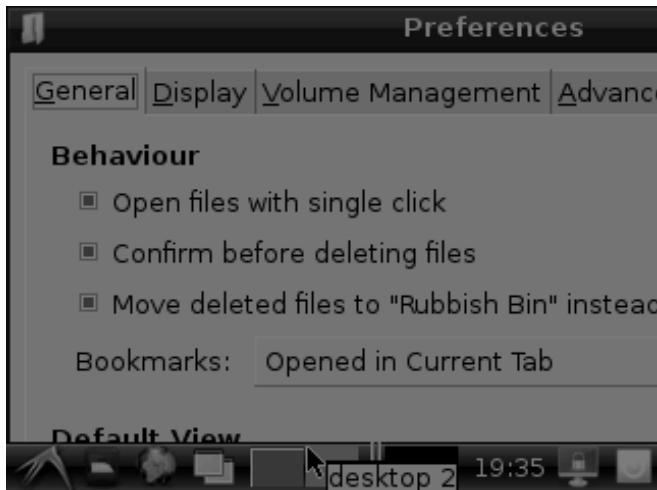
If you want to double-click on icons to launch something in X you may find it annoying to get it to work right. In LXDE you can simply set it up so that you only need to single click instead of double. From LXDE launch the file manager (sorry these pix are grayscale and from the 2.8" TFT, still figuring out how to screenshot the framebuffer!)



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

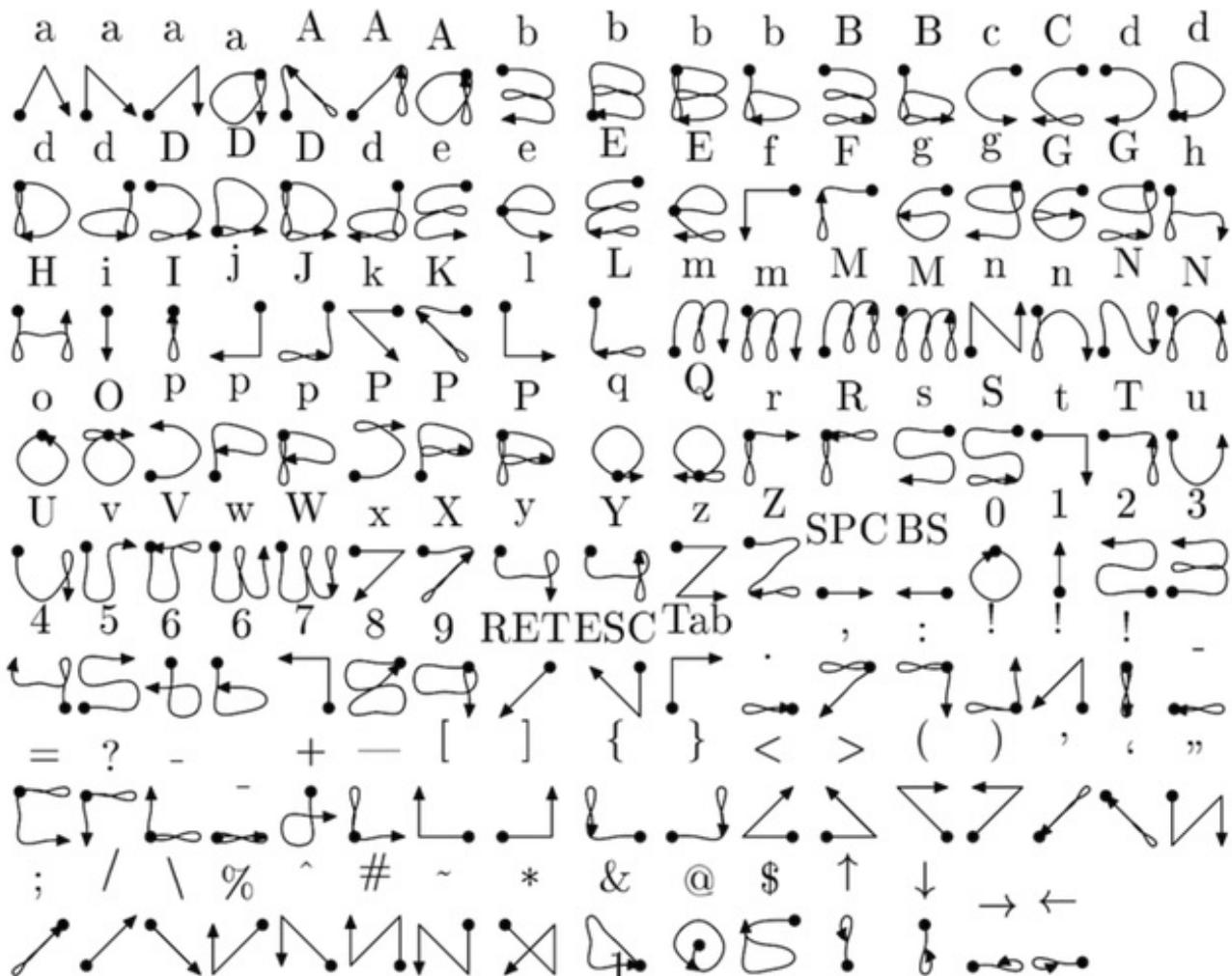


Then select **Open files with single click** and close the window (you'll need to drag it over to get to the X button)



Gesture Input

You can use the same tutorial we have for the 2.8" TFT if you'd like to use gesture input! (<http://adafru.it/dZf>)



FAQ

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

Check out the [2.8" resistive PiTFT FAQ](http://adafru.it/e1o) (<http://adafru.it/e1o>) for an answer to this common question.

How can I bring up X on the HDMI/TV monitor?

Use the **fb0** framebuffer when you want to display stuff on the HDMI/TV display, for example:

FRAMEBUFFER=/dev/fb0 startx

will use the HDMI/TV framebuffer for X windows instead of the PiTFT

That doesn't work! I can't get X on HDMI!

If both

FRAMEBUFFER=/dev/fb0 startx &

and

FRAMEBUFFER=/dev/fb1 startx &

wind up showing the GUI on your PiTFT, enter the following instruction from the command line:

```
sudo mv /usr/share/X11/xorg.conf.d/99-fbturbo.conf ~
```

I'm trying 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 ~

I want better performance and faster updates!

You can also change the SPI frequency (overclock the display) by editing
/etc/modprobe.d/adafruit.conf and changing the options line to:

```
options fbtft_device name=adafruitrt35 rotate=90 frequency=64000000 fps=30
```

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 and it will update a bit faster

You can tweak fps (frames per second) from 20 to 60 and frequency up to 42MHz 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. You can try 62MHz SPI but you may find the image corrupted so, again, tradeoffs! :)

How can I take screenshots of the display?

We took the screenshots for this tutorial with
[fbgra](http://adafruit.it/diV) (<http://adafruit.it/diV>) (<http://adafruit.it/diV>)

```
wget http://fbgrab.monells.se/fbgrab-1.2.tar.gz (http://adafruit.it/diW)
tar -zxvf fbgrab*.gz
cd fbgrab/
make
./fbgrab screenshot.png
```

COM3 - PuTTY

```
pi@raspberrypi:~$ wget http://fbgrab.monells.se/fbgrab-1.2.tar.gz
--2014-04-21 19:26:22-- http://fbgrab.monells.se/fbgrab-1.2.tar.gz
Resolving fbgrab.monells.se (fbgrab.monells.se)... 66.33.214.148
Connecting to fbgrab.monells.se (fbgrab.monells.se)|66.33.214.148|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 12836 (13K) [application/x-tar]
Saving to: `fbgrab-1.2.tar.gz'

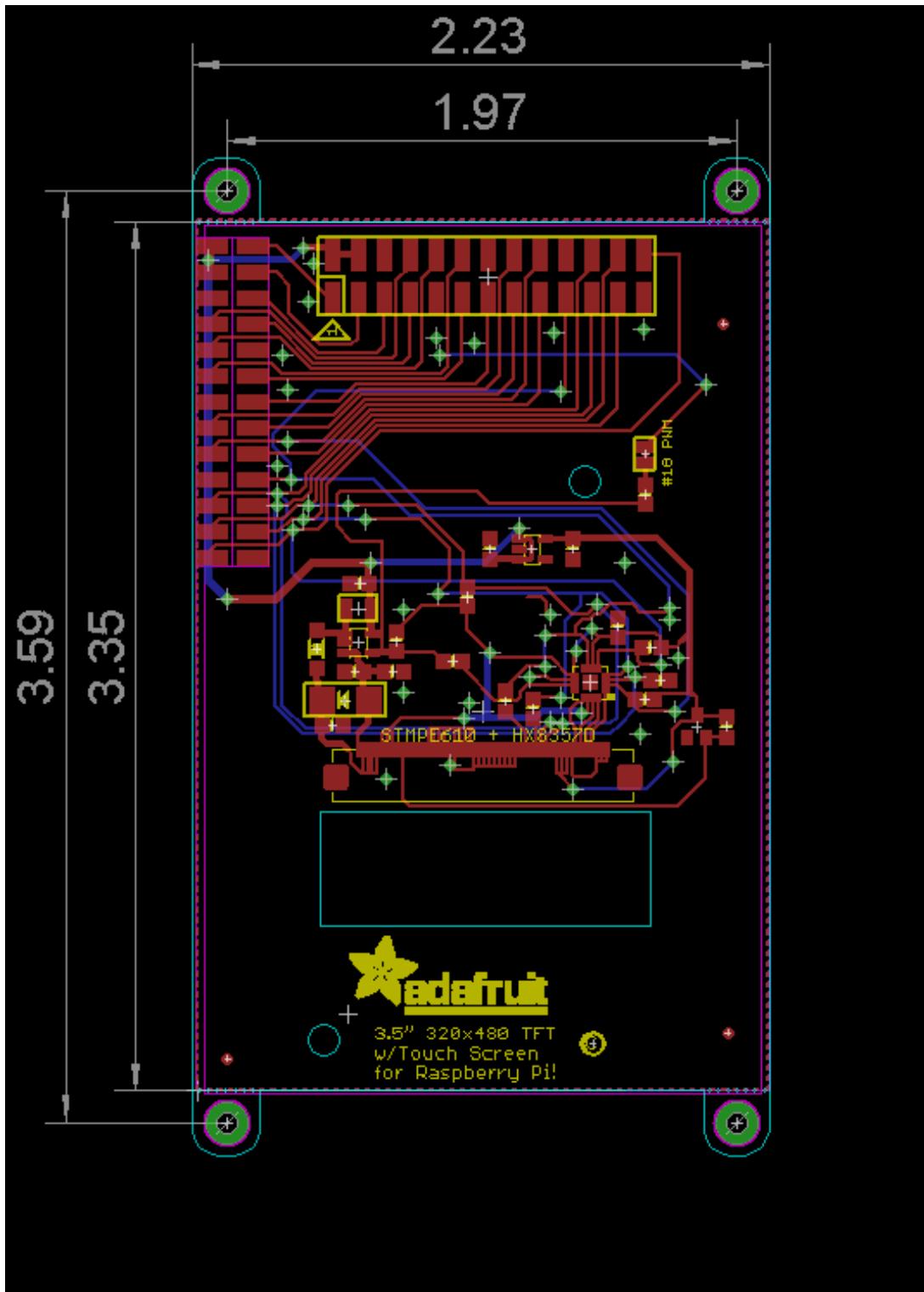
100%[=====] 12,836      --.-K/s   in 0.03s

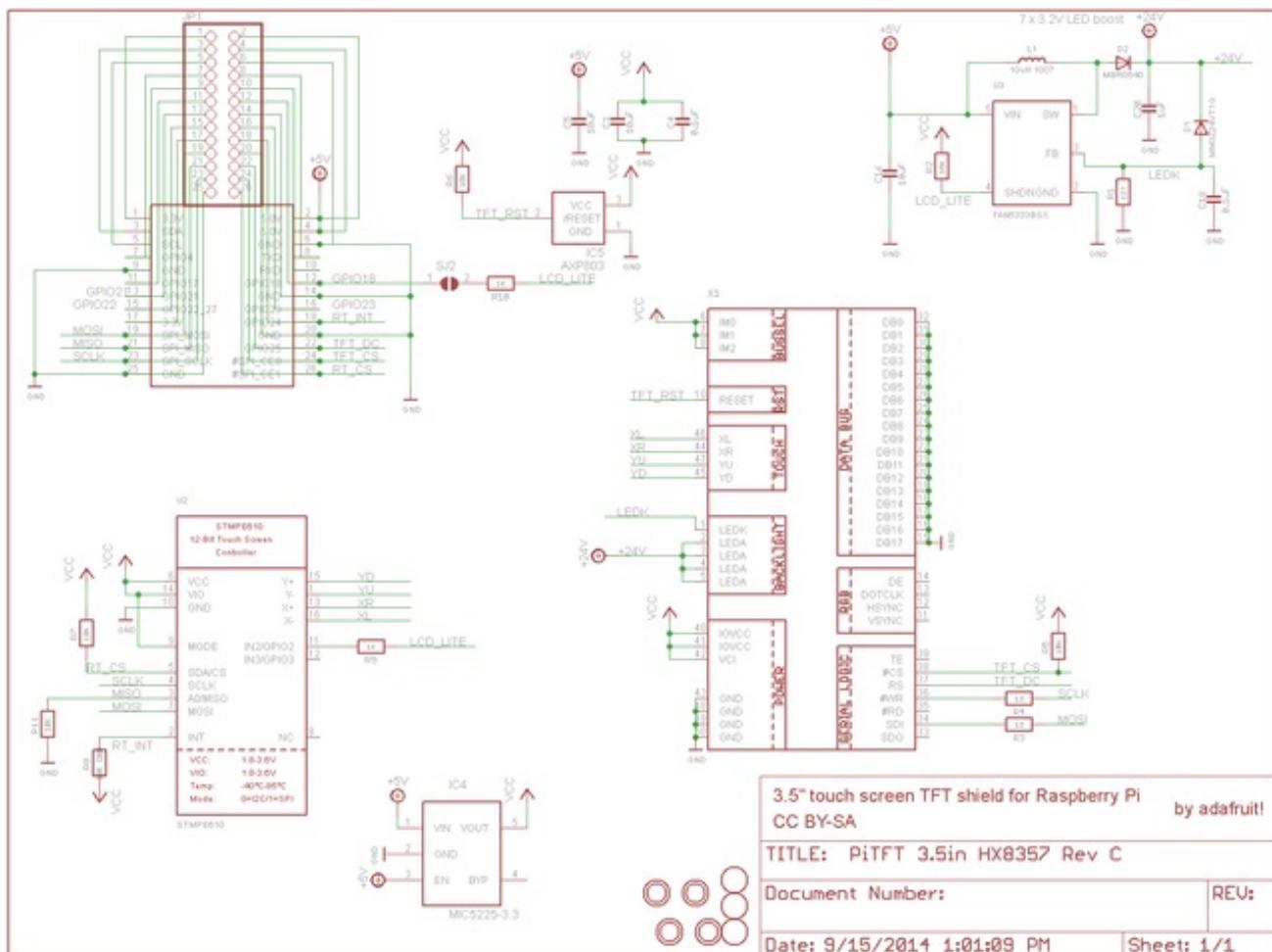
2014-04-21 19:26:22 (497 KB/s) - `fbgrab-1.2.tar.gz' saved [12836/12836]

pi@raspberrypi:~$ tar -zxvf fbgrab-1.2.tar.gz
fbgrab/
fbgrab/fbgrab.c
fbgrab/INSTALL
fbgrab/fbgrab.1.man
fbgrab/COPYING
fbgrab/Makefile
pi@raspberrypi:~$ cd fbgrab/
pi@raspberrypi:~/fbgrab$ make
cc -g -Wall  fbgrab.c -lpng -lz -o fbgrab
gzip --best --to-stdout fbgrab.1.man > fbgrab.1.gz
pi@raspberrypi:~/fbgrab$ ./fbgrab
Usage: ./fbgrab      [-hi] [-{C|c} vt] [-d dev] [-s n] [-z n]
                  [-f fromfile -w n -h n -b n] filename.png
pi@raspberrypi:~/fbgrab$ ./fbgrab filemanager.png
Resolution: 320x240 depth 16
Converting image from 16
Now writing PNG file (compression -1)
```

Downloads

- The latest kernel fork that adds all the TFT, touchscreen, and other addons is here on [github](http://adafru.it/aPa) (<http://adafru.it/aPa>)
- Datasheet for the controller chip (<http://adafru.it/dQQ>)
- Datasheet for the 'raw' 3.5" TFT display (<http://adafru.it/dR4>)





3.5" touch screen TFT shield for Raspberry Pi
CC BY-SA by adafruit!

TITLE: PiTFT 3.5in HX8357 Rev C

Document Number:

REV:

Date: 9/15/2014 1:01:09 PM

Sheet: 1/1