adafruit learning system

Adafruit PiTFT - 2.8" Touchscreen Display for Raspberry Pi

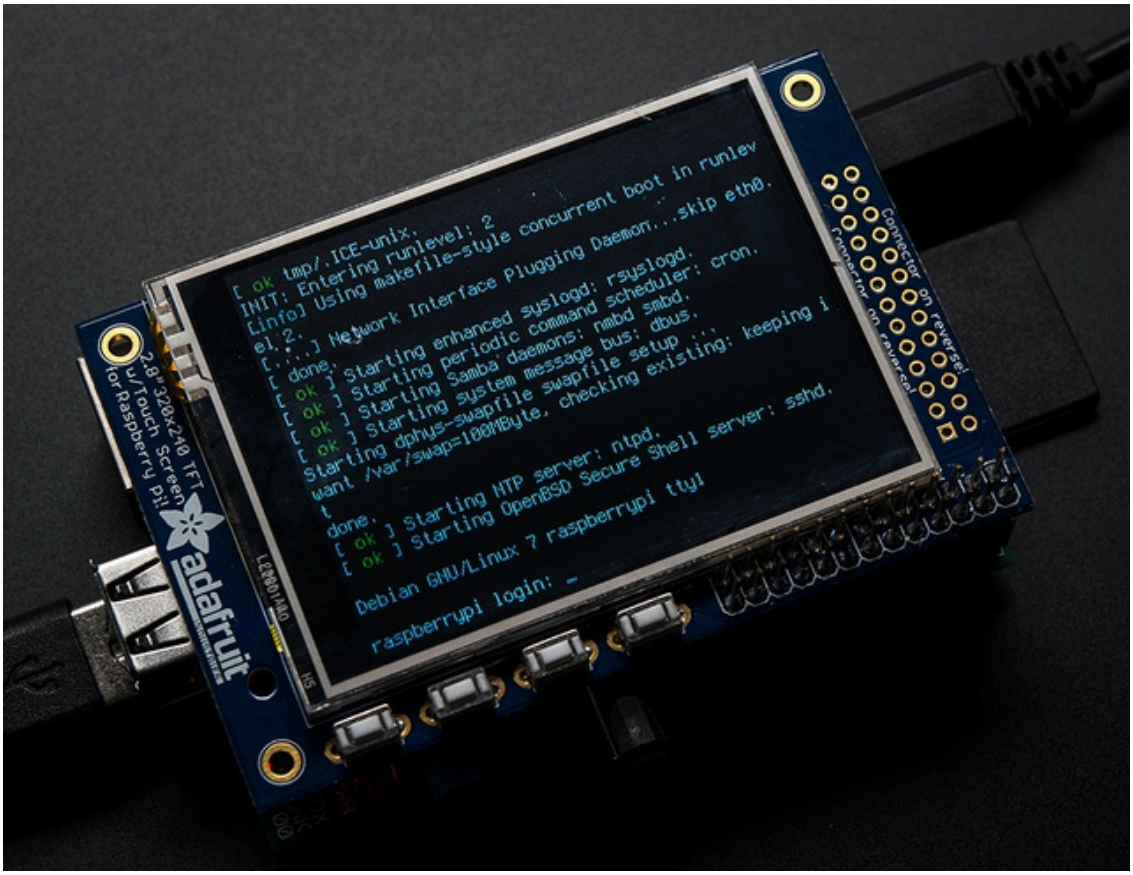
Created by lady ada

Last updated on 2014-04-24 02:00:36 PM EDT

Guide Contents

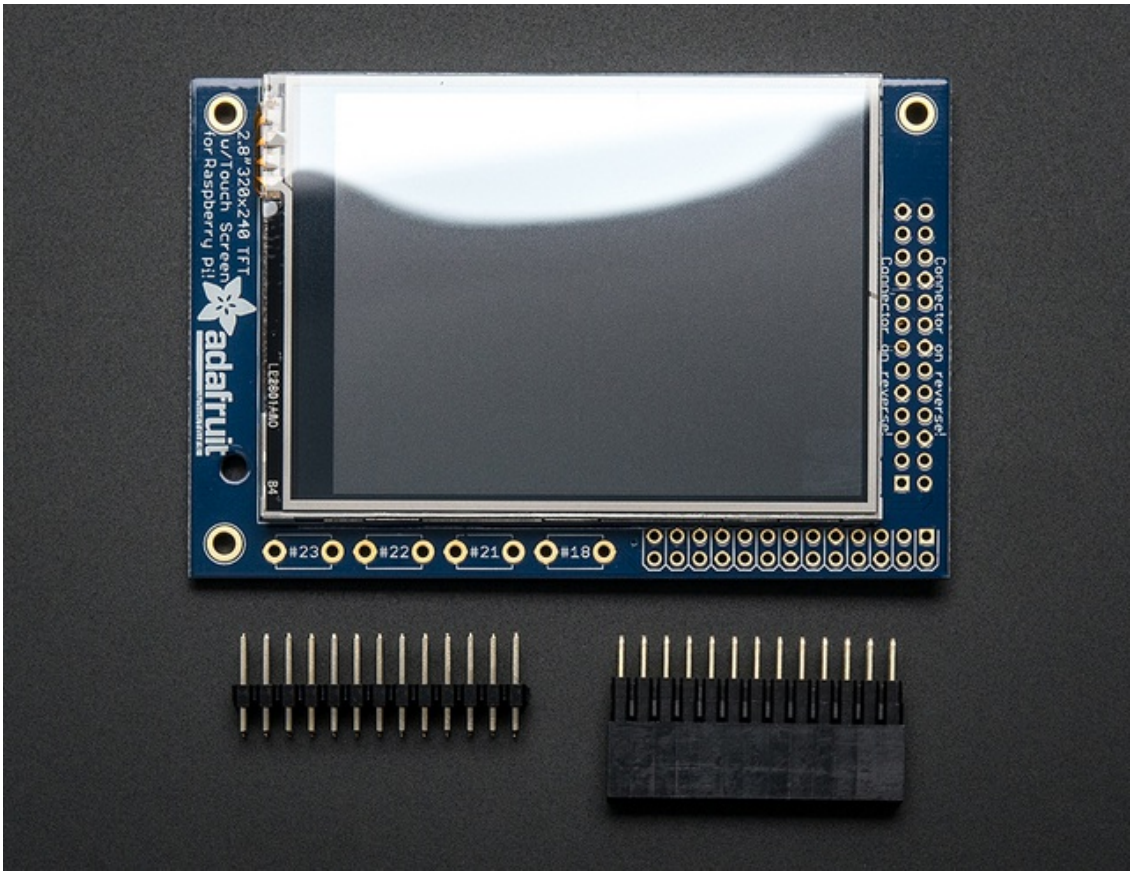
Overview

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!

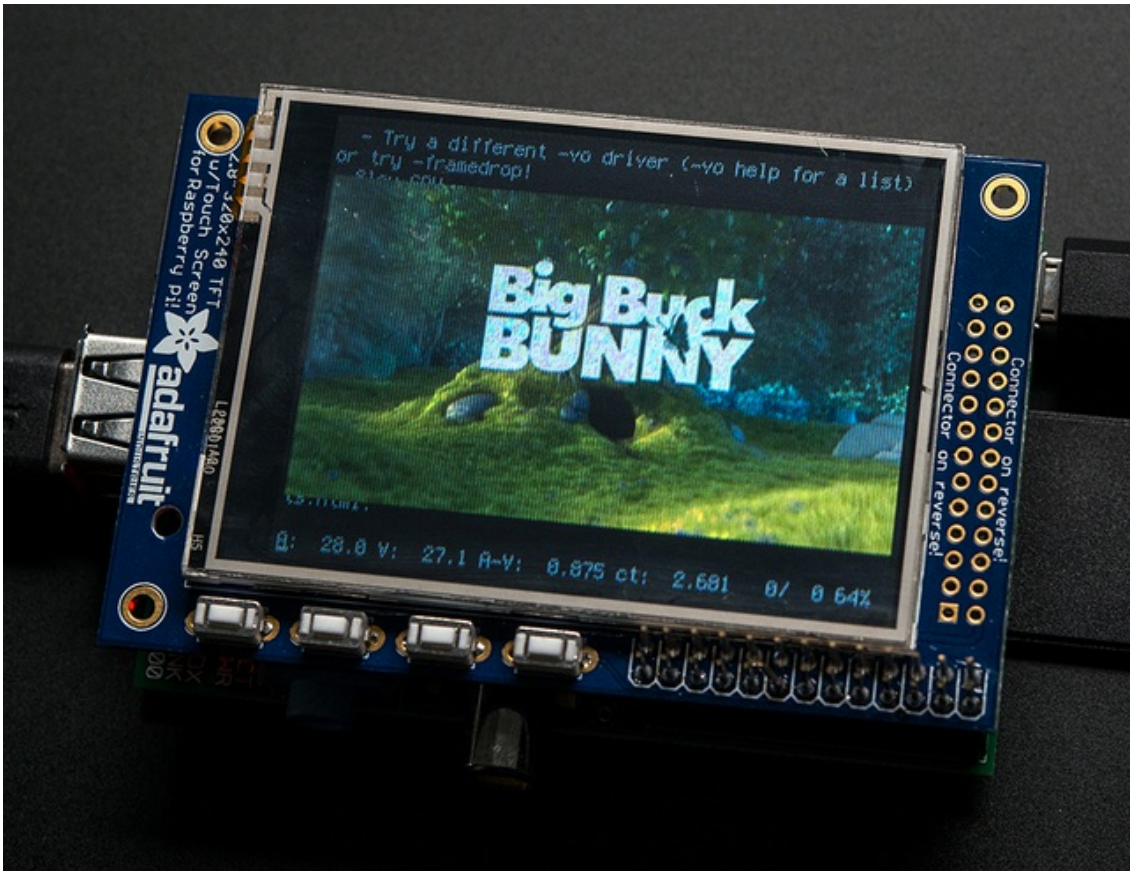


This design uses the hardware SPI pins (SCK, MOSI, MISO, CE0, CE1) as well as GPIO #25 and #24. All other GPIO are unused. Since we had a tiny bit of space, there's 4 spots for optional slim tactile switches wired to four GPIOs, that you can use if you want to make a basic user interface. For example,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

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

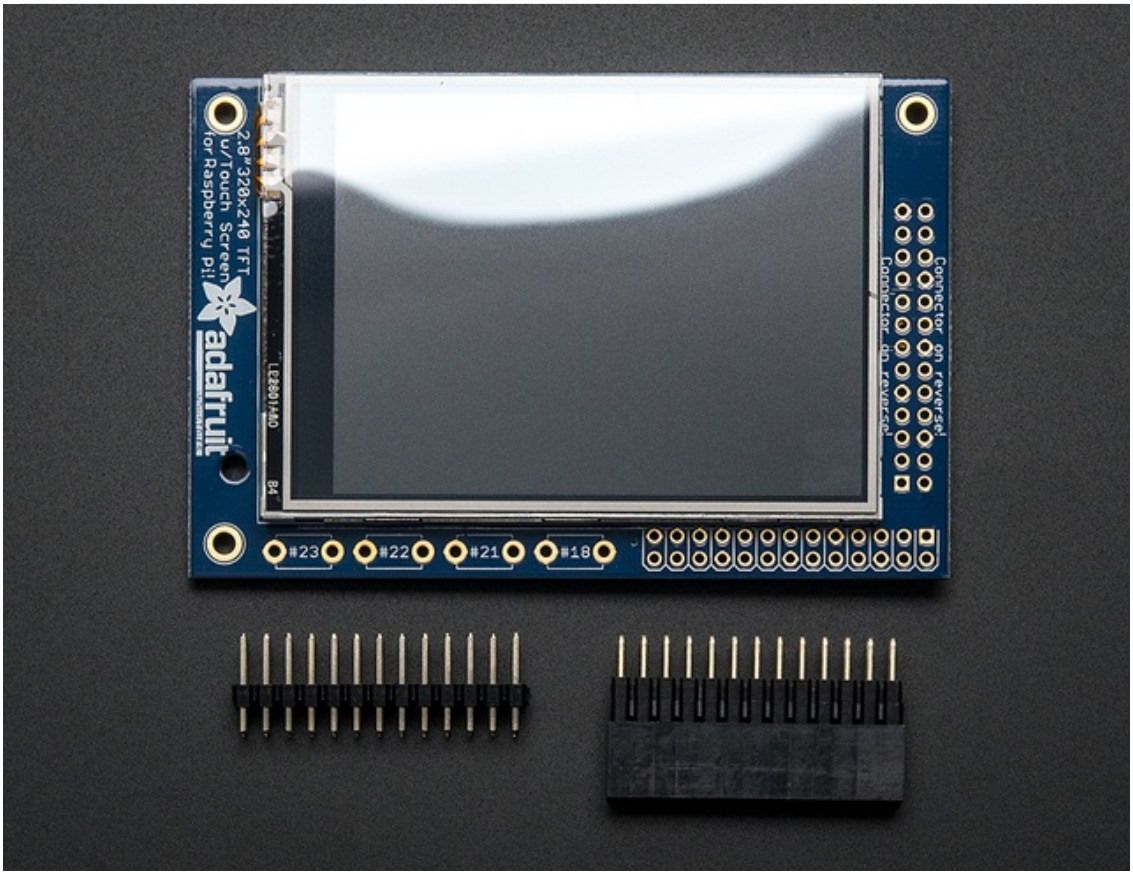


To make it super easy for use: we've created a custom kernel package based of off Notro's awesome framebuffer work, so you can install it over your existing Raspbian (or derivative) images in just a few commands.

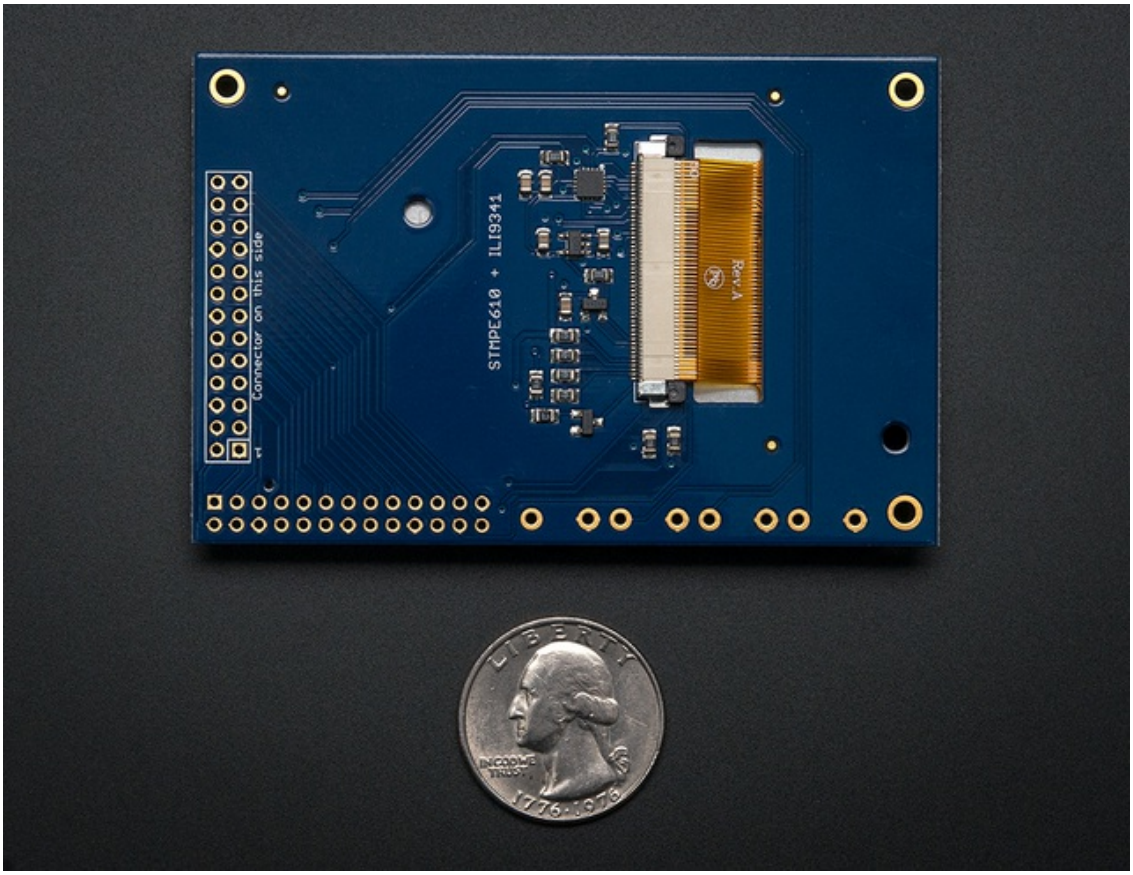


This tutorial series shows you how to install the software, as well as calibrate the touchscreen, splay videos, display images such as from your PiCam and more!

Assembly

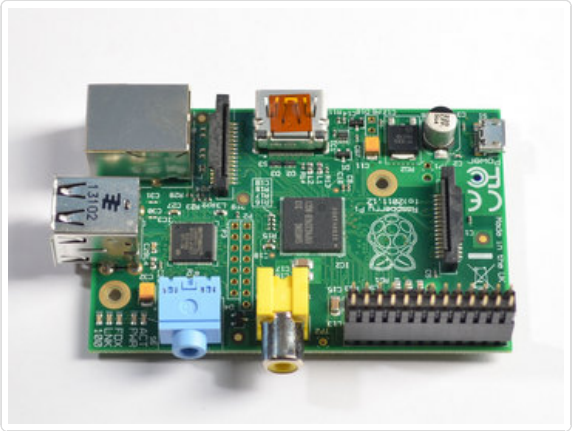


Before you start check that you have the parts you need: an assembled PiTFT plate with the 2.8" screen, extra tall female header and the 2x13 male header. Note that it is normal for the screen to be 'loose' - this is so its easier for you to solder the connector on!



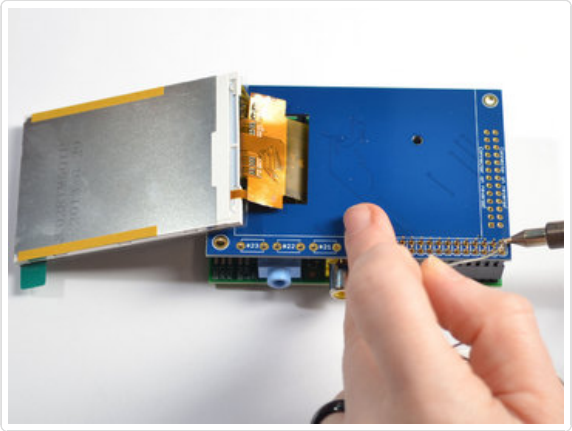
Check also on the back that the TFT is attached and that the flex connector is seated into the onboard FPC socket.

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

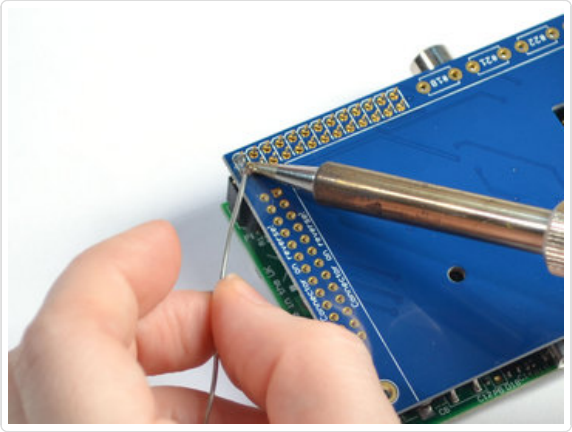


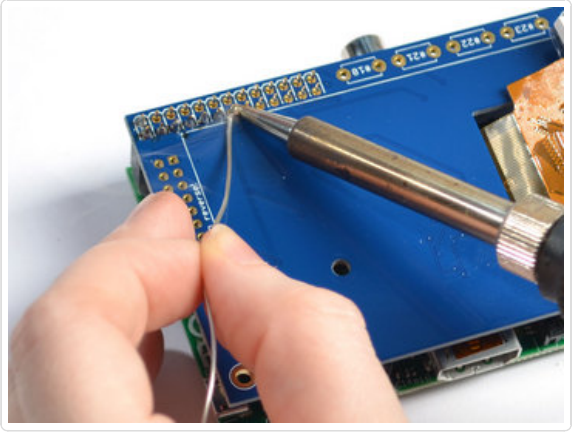
Plug the extra tall female header into the GPIO port on the Pi as shown. Make sure its seated nice and flat

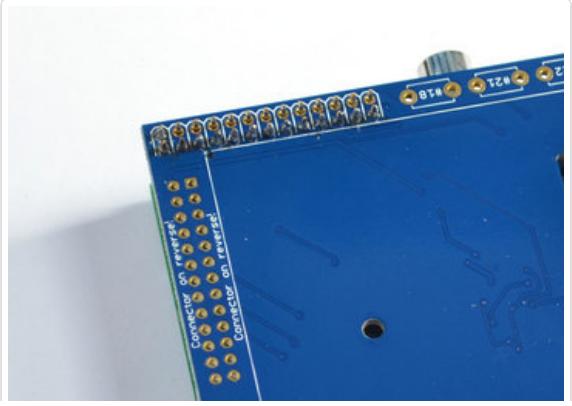
Place the PiTFT shield on top so all the pins stick through the connector on the side. Gently flip the TFT so its off to the side and wont be in your way while you solder



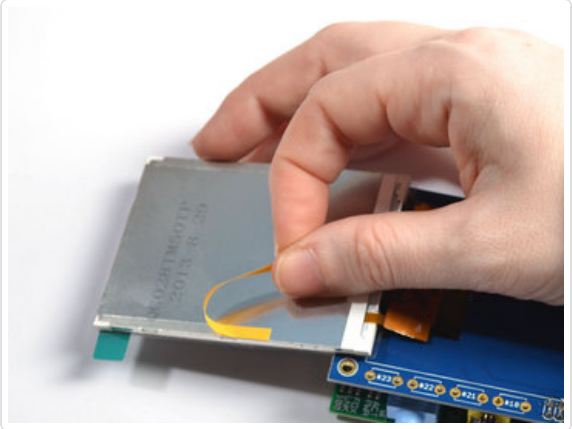
Heat up your soldering iron, and grab some solder. Start by tack-soldering one of the corners while pressing on the plate to make it sit flat. Once you have one or two pins done you can continue to solder each of the pins.



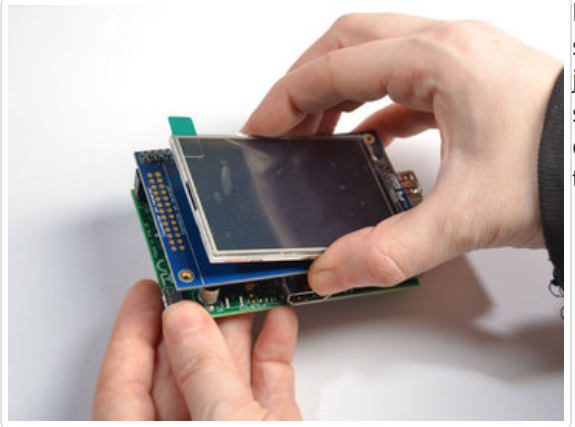




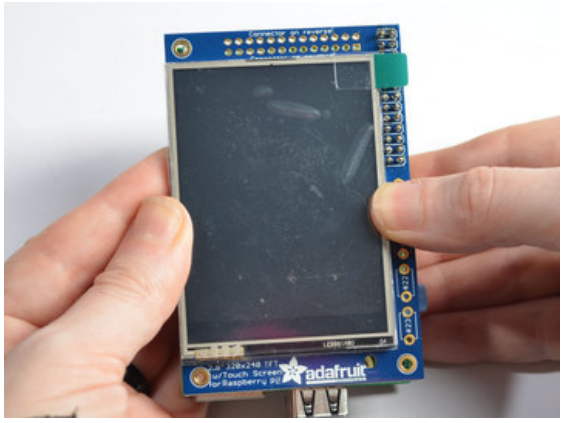
Before attaching the display, check that all the pins are soldered nicely and there's no bridging, cold solder, shorts, or unsoldered pins.



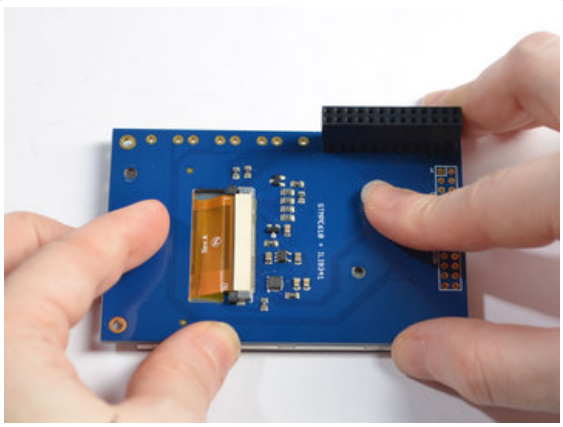
Now we can attach the screen. Remove the two thin tape cover strips.



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.

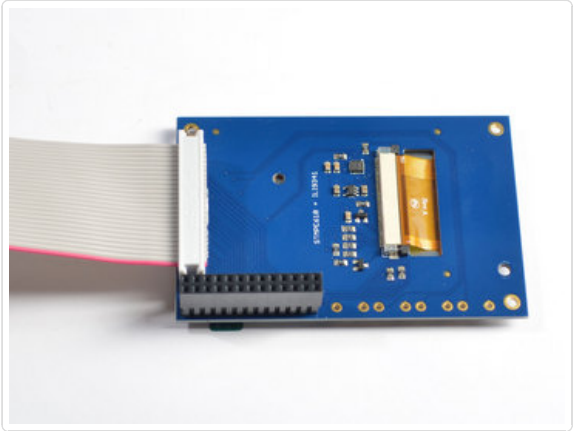


Once you have the screen so it is definitely not touching the header, you can gently press on the sides to secure the tape.



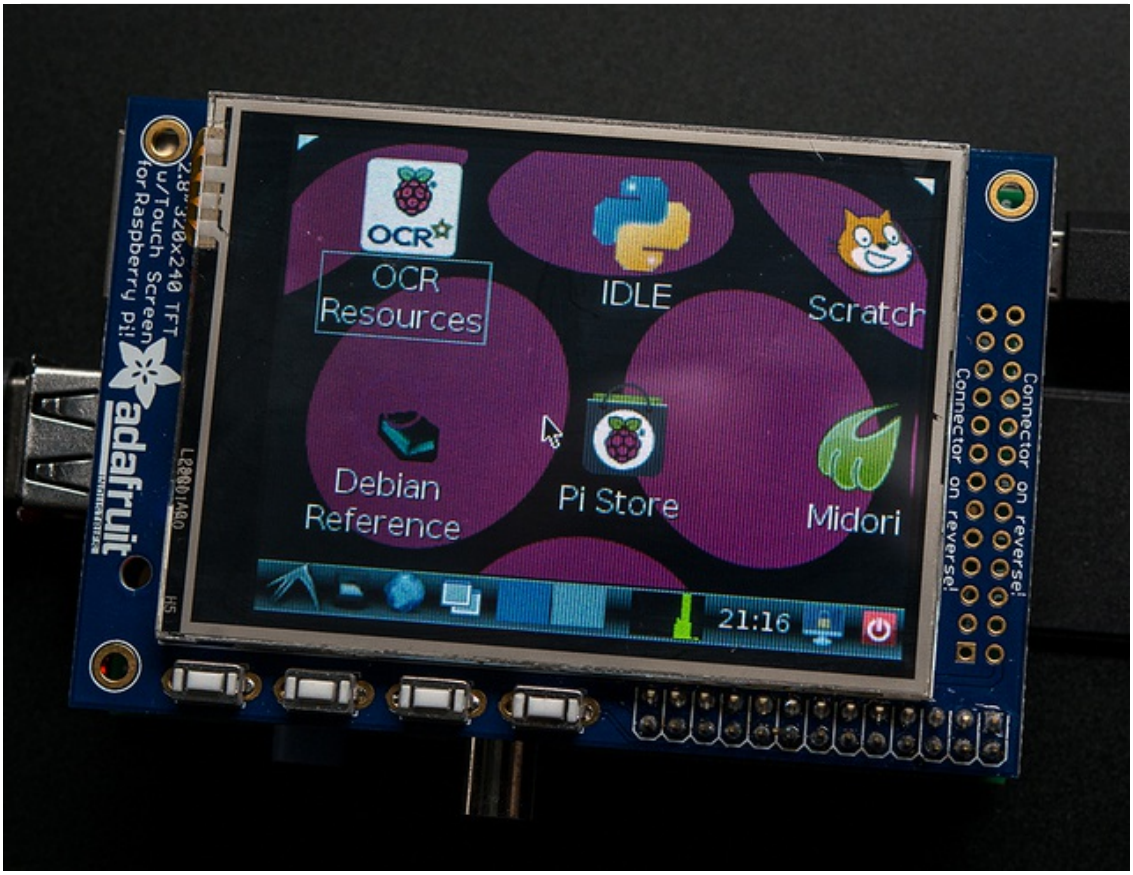
If the protective plastic cover is still on the screen you can press it against a clean table from above. That way you will really securely attach it!

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



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!

Software Installation



In order to add support for the 2.8" 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 patchfile (http://adafru.it/cY2), 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).

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** !

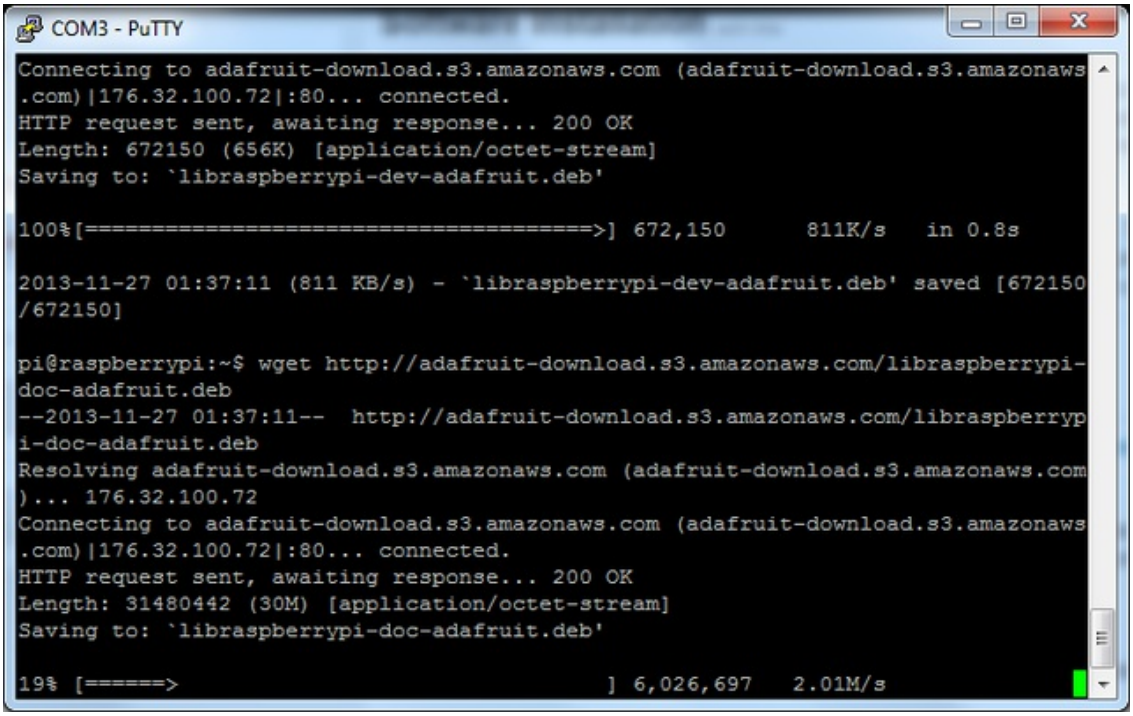
> To run these all the setup and config commands you'll need to be logged into a proper Terminal - use ssh, a console cable, or the main text console (on a TV). The WebIDE console may not work.

Download & Install

Download the multiple files you'll need by running the following commands

```
cd ~
wget http://adafruit-download.s3.amazonaws.com/libraspberrypi-bin-adafruit.deb
wget http://adafruit-download.s3.amazonaws.com/libraspberrypi-dev-adafruit.deb
wget http://adafruit-download.s3.amazonaws.com/libraspberrypi-doc-adafruit.deb
wget http://adafruit-download.s3.amazonaws.com/libraspberrypi0-adafruit.deb
wget http://adafruit-download.s3.amazonaws.com/raspberrypi-bootloader-adafruit-112613.deb
```

Advanced users! Want to beta test our new DMA-enabled kernel? Its even faster! Instead of the last wget item - grab the April 2014 kernel deb file with "wget http://adafruit-download.s3.amazonaws.com/raspberrypi-bootloader-adafruit-20140421-3.deb" You can always install this over the 11-26-13 version or go back and forth



Next install the new kernel with

> **sudo dpkg -i -B *.deb**

This will take a few minutes so go make a sandwich or coffee

```
COM3 - PuTTY
pi@raspberrypi:~$ sudo dpkg -i -B *.deb
Selecting previously unselected package libraspberrypi0-adafruit.
dpkg: considering removing libraspberrypi0 in favour of libraspberrypi0-adafruit
 ...
dpkg: yes, will remove libraspberrypi0 in favour of libraspberrypi0-adafruit
(Reading database ... 62260 files and directories currently installed.)
Unpacking libraspberrypi0-adafruit (from libraspberrypi0-adafruit.deb) ...
De-configuring libraspberrypi-doc, to allow removal of libraspberrypi0 ...
De-configuring libraspberrypi-dev, to allow removal of libraspberrypi0 ...
De-configuring libraspberrypi-bin, to allow removal of libraspberrypi0 ...
Selecting previously unselected package libraspberrypi-bin-adafruit.
dpkg: considering removing libraspberrypi-bin in favour of libraspberrypi-bin-ad
afruit ...
dpkg: libraspberrypi-bin is not properly installed; ignoring any dependencies on
 it
dpkg: yes, will remove libraspberrypi-bin in favour of libraspberrypi-bin-adafru
it
Unpacking libraspberrypi-bin-adafruit (from libraspberrypi-bin-adafruit.deb) ...
Selecting previously unselected package libraspberrypi-dev-adafruit.
dpkg: considering removing libraspberrypi-dev in favour of libraspberrypi-dev-ad
afruit ...
dpkg: libraspberrypi-dev is not properly installed; ignoring any dependencies on
 it
```

```
COM3 - PuTTY
in by rpikernelhack'
Removing 'diversion of /boot/fixup.dat to /usr/share/rpikernelhack/fixup.dat by
rpikernelhack'
Removing 'diversion of /boot/fixup_cd.dat to /usr/share/rpikernelhack/fixup_cd.d
at by rpikernelhack'
Removing 'diversion of /boot/fixup_x.dat to /usr/share/rpikernelhack/fixup_x.dat
 by rpikernelhack'
Removing 'diversion of /boot/kernel.img to /usr/share/rpikernelhack/kernel.img b
y rpikernelhack'
Removing 'diversion of /boot/kernel_cutdown.img to /usr/share/rpikernelhack/kern
el_cutdown.img by rpikernelhack'
Removing 'diversion of /boot/kernel_emergency.img to /usr/share/rpikernelhack/ke
rnel_emergency.img by rpikernelhack'
Removing 'diversion of /boot/start.elf to /usr/share/rpikernelhack/start.elf by
rpikernelhack'
Removing 'diversion of /boot/start_cd.elf to /usr/share/rpikernelhack/start_cd.e
lf by rpikernelhack'
Removing 'diversion of /boot/start_x.elf to /usr/share/rpikernelhack/start_x.elf
 by rpikernelhack'
Setting up libraspberrypi0-adafruit (1.20130902-1) ...
Setting up libraspberrypi-bin-adafruit (1.20130902-1) ...
Setting up libraspberrypi-dev-adafruit (1.20130902-1) ...
Setting up libraspberrypi-doc-adafruit (1.20130902-1) ...
pi@raspberrypi:~$
```

If you have a version of Raspbian more recent than Sept. 2013, you'll need to turn off the accelerated X framebuffer here, run "sudo mv /usr/share/X11/xorg.conf.d/99-fbturbo.conf ~" to remove the accelerated X buffer and save it in your home directory

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)

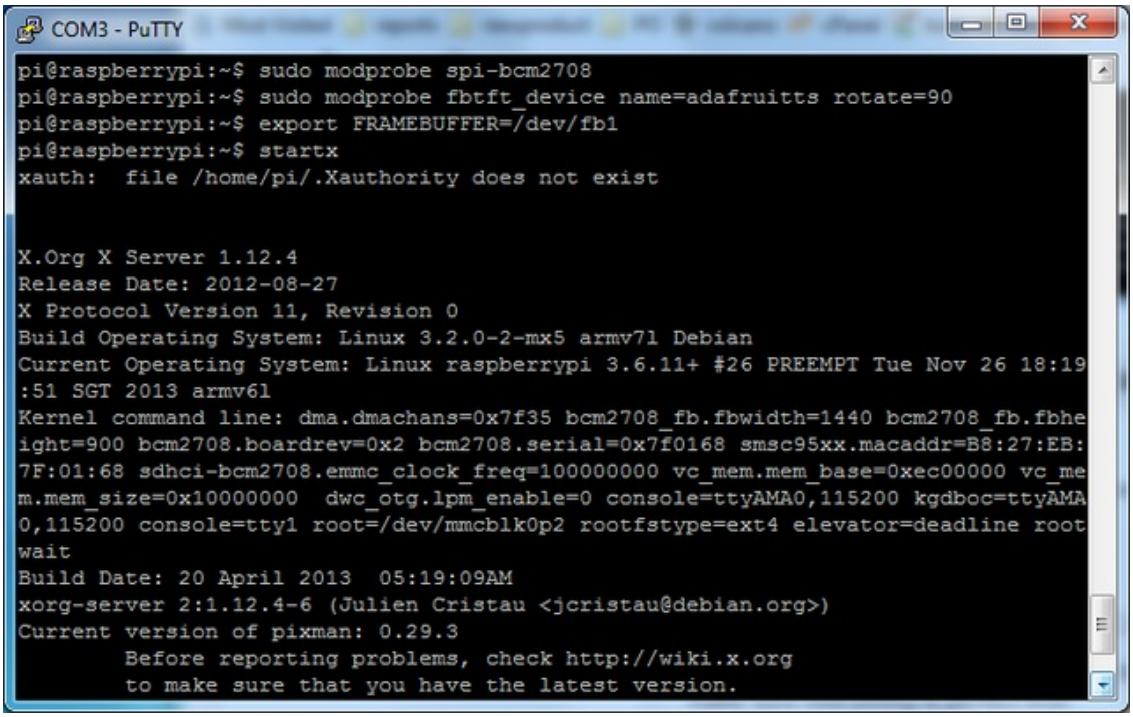
or

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

Now that you're rebooted, log back in on the console/TV/SSH. There's no screen yet, we'll do a test to make sure everything is perfect first!

Run the following commands to install the screen driver. Note that the screen will go from white to black indicating the commands succeeded.

> **sudo modprobe spi-bcm2708**
> **sudo modprobe fbtft_device name=adafruitts rotate=90**
> **export FRAMEBUFFER=/dev/fb1**
> **startx**



You should see the Pi desktop show up on the TFT! Congrats, you've completed the first test perfectly. Now we'll make the TFT kernel modules automatically install on boot.

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

Module Auto-loading

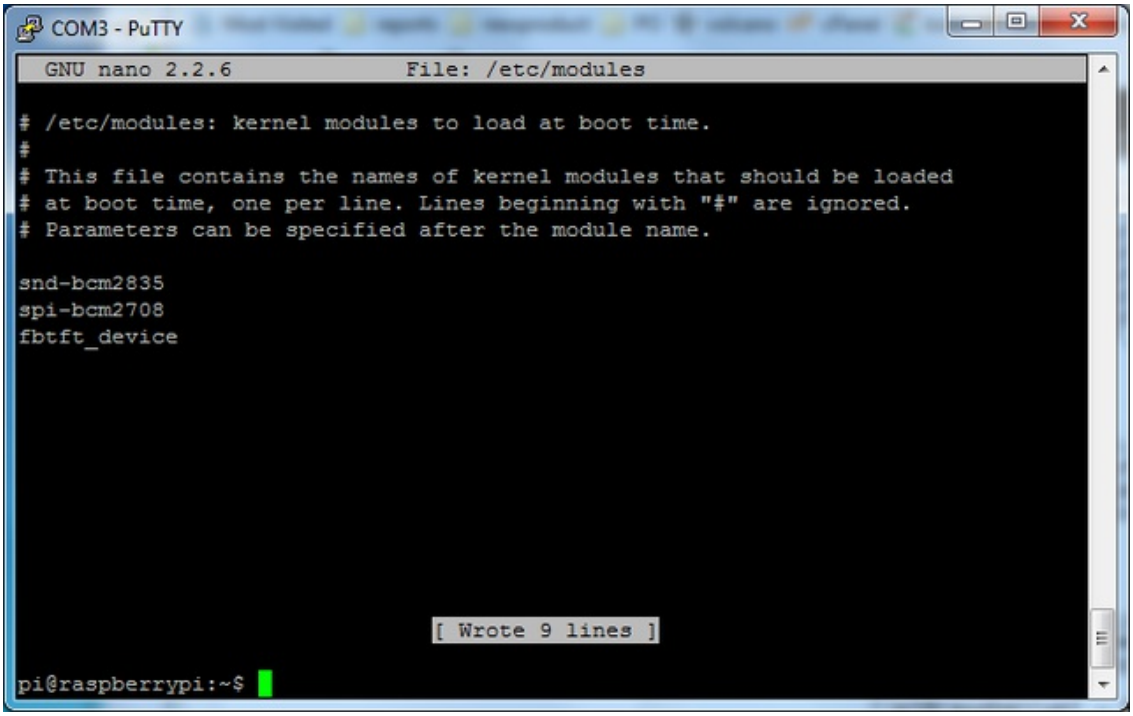
We'll now make the modules auto-load. Lets edit the /etc/modules list with

> **sudo nano /etc/modules**

add two lines

```
spi-bcm2708
fbtft_device
```

Then **Control-X Y [return]** to save



You're not done yet! Now we have to add the configuration details in a new modprobe file:

```
sudo nano /etc/modprobe.d/adafruit.conf
```

add the following line

```
options fbtft_device name=adafruitts rotate=90
frequency=32000000
```

and save like you did before.

The **rotate=** variable tells the driver to rotate the screen **0 90 180** or **270** degrees.
0 is portrait, with the bottom near the "Adafruit Logo"
90 is landscape, with the bottom of the screen near the buttons.
180 is portrait, with the top near the "Adafruit Logo"
270 is landscape, with the top of the screen near the buttons.
You can change this file with **nano** and reboot to make the change stick.

The frequency= variable tells the driver how to fast to drive the display. 32MHz (**32000000**) is a pretty nice 20 FPS rate but if your screen is acting funny, try taking it down to 16MHz (**16000000**)

```
COM3 - PuTTY
  GNU nano 2.2.6        File: /etc/modprobe.d/adafruit.conf

options fbtft_device name=adafruitts rotate=90 frequency=32000000

                                [ Wrote 2 lines ]

pi@raspberrypi:~$
```

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 ILI9340 screen frequency as highlighted here

```
COM3 - PuTTY
th irq 16
[   12.377691] bcm2708_spi bcm2708_spi.0: DMA channel 4 at address 0xf2007400 wi
th irq 20
[   12.401498] spi_master spi0: will run message pump with realtime priority
[   12.429977] bcm2708_spi bcm2708_spi.0: SPI Controller at 0x20204000 (irq 80)
[   12.450193] bcm2708_spi bcm2708_spi.0: SPI Controller running in polling mode
[   12.618455] fbtft_device:  SPI devices registered:
[   12.637464] fbtft_device:      spidev spi0.0 500kHz 8 bits mode=0x00
[   12.652365] fbtft_device:      spidev spi0.1 500kHz 8 bits mode=0x00
[   12.663233] fbtft_device:  'fb' Platform devices registered:
[   12.673221] fbtft_device:      bcm2708_fb id=-1 pdata? no
[   12.683062] fbtft_device: Deleting spi0.1 (spi0.1)
[   12.695163] fbtft_device: Looking at item 0
[   12.705831] fbtft_device: Setting pin 24 to 2
[   12.800476] stmpe-spi spi0.1: stmpe610 detected, chip id: 0x811
[   12.818094] fbtft_device: Deleting spi0.0 (spi0.0)
[   12.840693] fbtft_device:  GPIOS used by 'adafruitts':
[   12.860192] fbtft_device:    'dc' = GPIO25
[   12.868574] fbtft_device:  SPI devices registered:
[   12.897923] fbtft_device:      stmpe610 spi0.1 500kHz 8 bits mode=0x00
[   12.915964] fbtft_device:      fb_ili9340 spi0.0 32000kHz 8 bits mode=0x00
[   12.959334] input: stmpe-ts as /devices/virtual/input/input3
[   13.488121] graphics fb1: fb_ili9340 frame buffer, 320x240, 150 KiB video mem
ory, 16 KiB buffer memory, fps=20, spi0.0 at 32 MHz
```

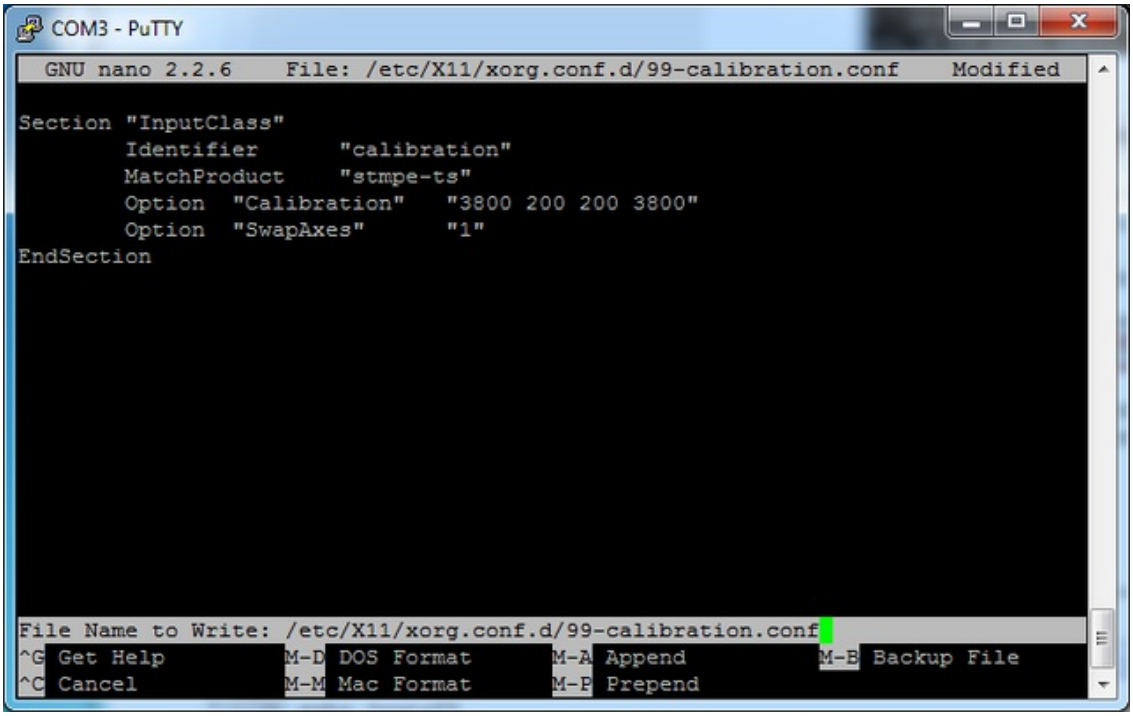
We can set up the touchscreen for **rotate=90** configuration by doing the following (for more delicate calibration or for other rotate=XX values, see the next section)
Create the directory and new calibration configuration file:

> **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 200 200 3800"
        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

```
COM3 - PuTTY
  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  ^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
```

Touchscreen Install & Calibrate

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"

```
COM3 - PuTTY
  GNU nano 2.2.6       File: /etc/udev/rules.d/95-stmpe.rules          Modified

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

^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

```
COM3 - PuTTY
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:~$
```

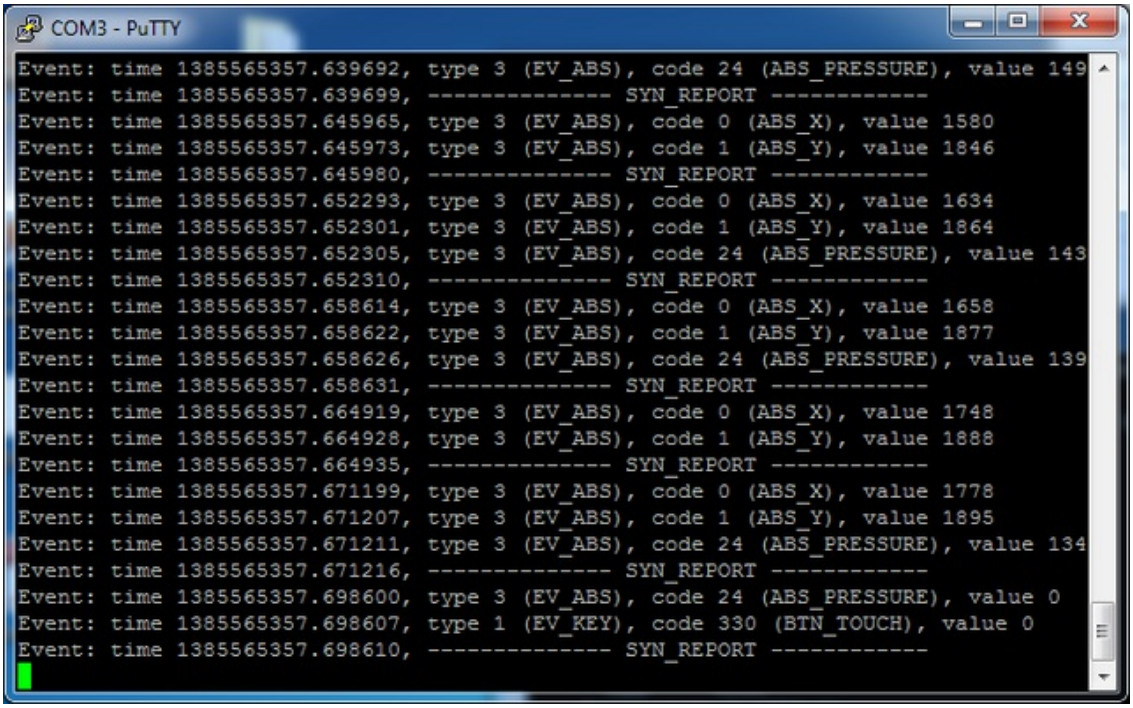
There are some tools we can use to calibrate & debug the touchscreen. Install the "event test" and "touchscreen library" packages with

> **sudo apt-get install evtest tslib libts-bin**

```
COM3 - PuTTY
pi@raspberrypi:~$
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.

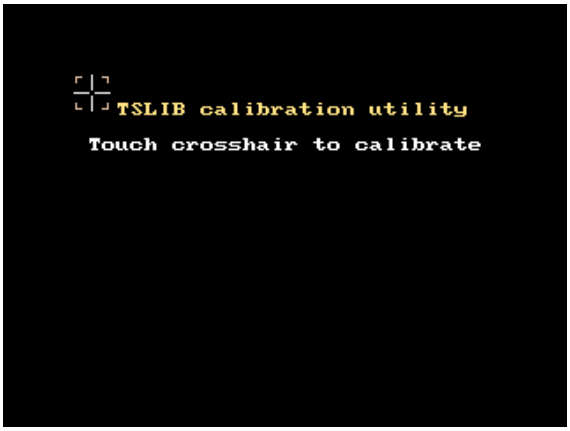
```
COM3 - PuTTY
pi@raspberrypi:~$ sudo evtest /dev/input/touchscreen
Input driver version is 1.0.1
Input device ID: bus 0x18 vendor 0x0 product 0x0 version 0x0
Input device name: "stmpe-ts"
Supported events:
  Event type 0 (EV_SYN)
  Event type 1 (EV_KEY)
    Event code 330 (BTN_TOUCH)
  Event type 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)
```

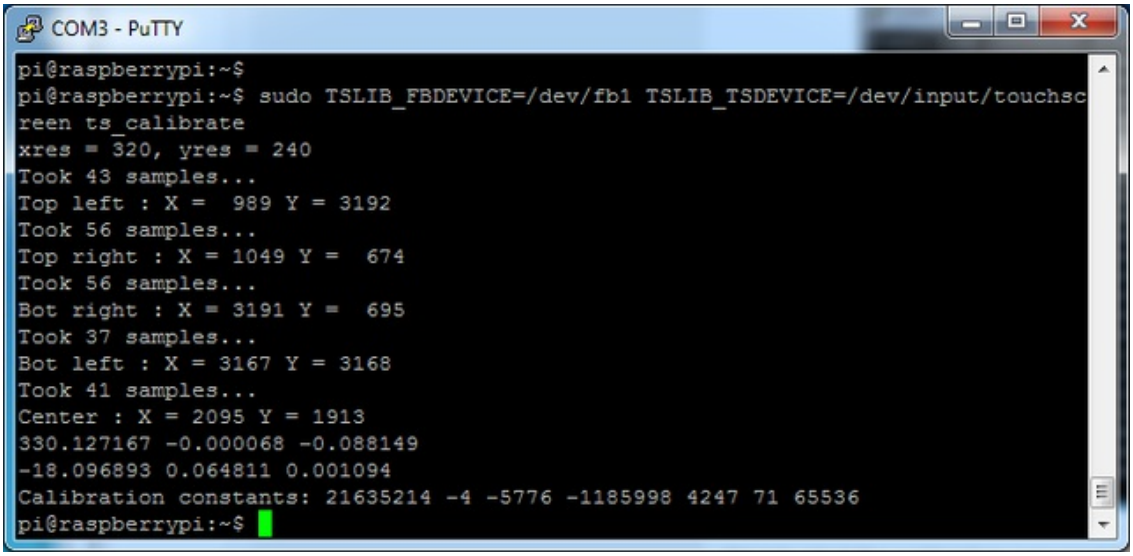


OK lets start the calibration process! You will want to calibrate the screen once but shouldn't have to do it more than that. We'll begin by calibrating on the command line by running

```
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!





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 isnt 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**

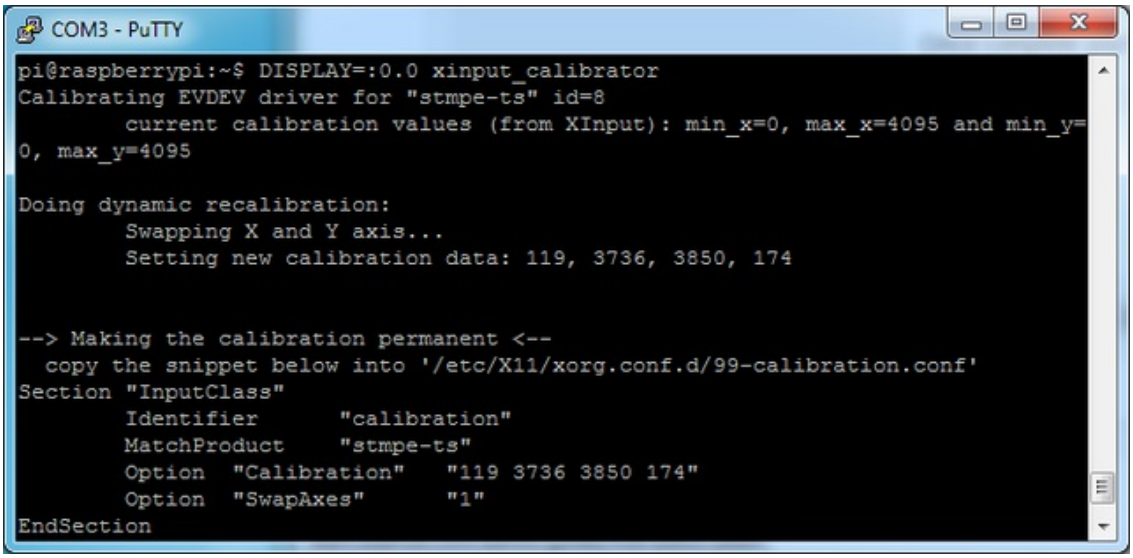
Before running **startx** and the calibrator - otherwise it gets really confused!

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 OR you can do what we do which is run startx in a SSH/Terminal shell and then run the xinput_calibrator in the X window, 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:



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.

For some reason 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!

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** (http://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!

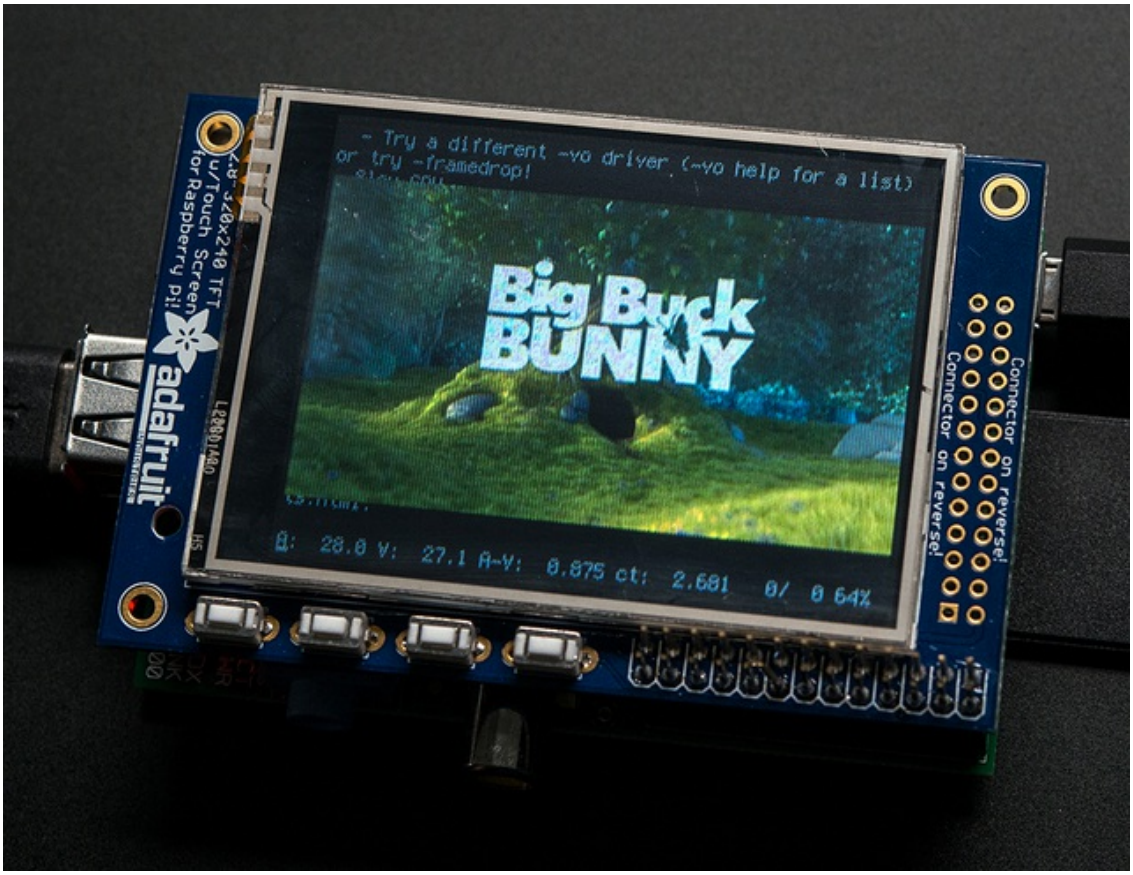
If you don't have **mplayer** yet, run **sudo apt-get install mplayer** to install it. It may take a few minutes to complete

```
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

mplayer -vo fbdev2:/dev/fb1 -x 240 -y 320 -framedrop bigbuckbunny320p.mp4

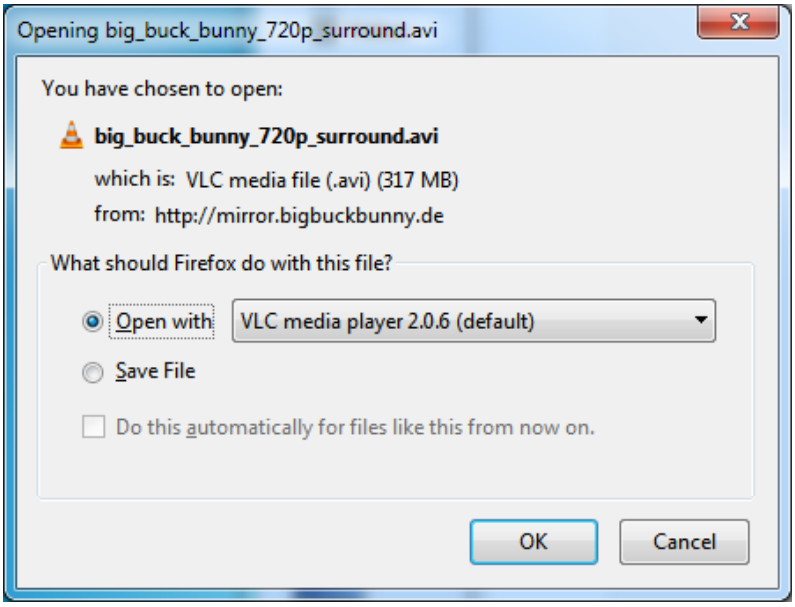
if your video is not sized for 320 wide, you may need to add a -zoom after -framedrop so that it will resize - note that this is quite taxing for the Pi, so it may result in a choppy or mis-synced video!



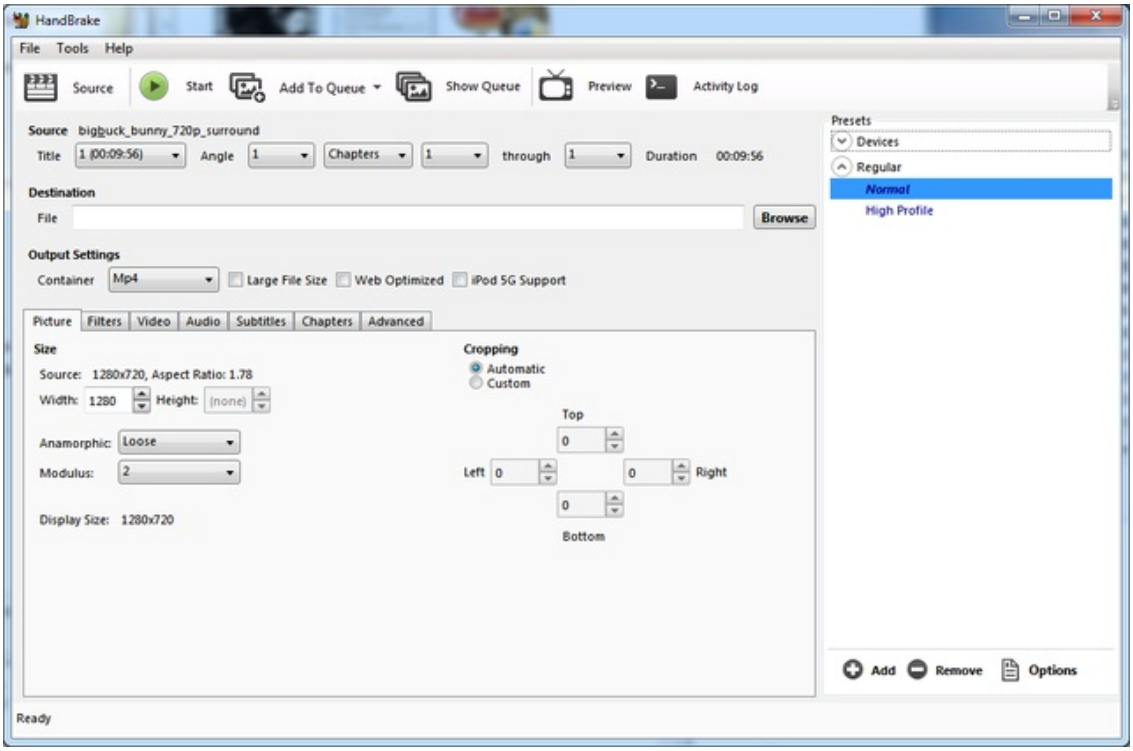
Converting/Resizing Videos

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 (http://adafru.it/cXS) video, which is creative commons and also very funny!

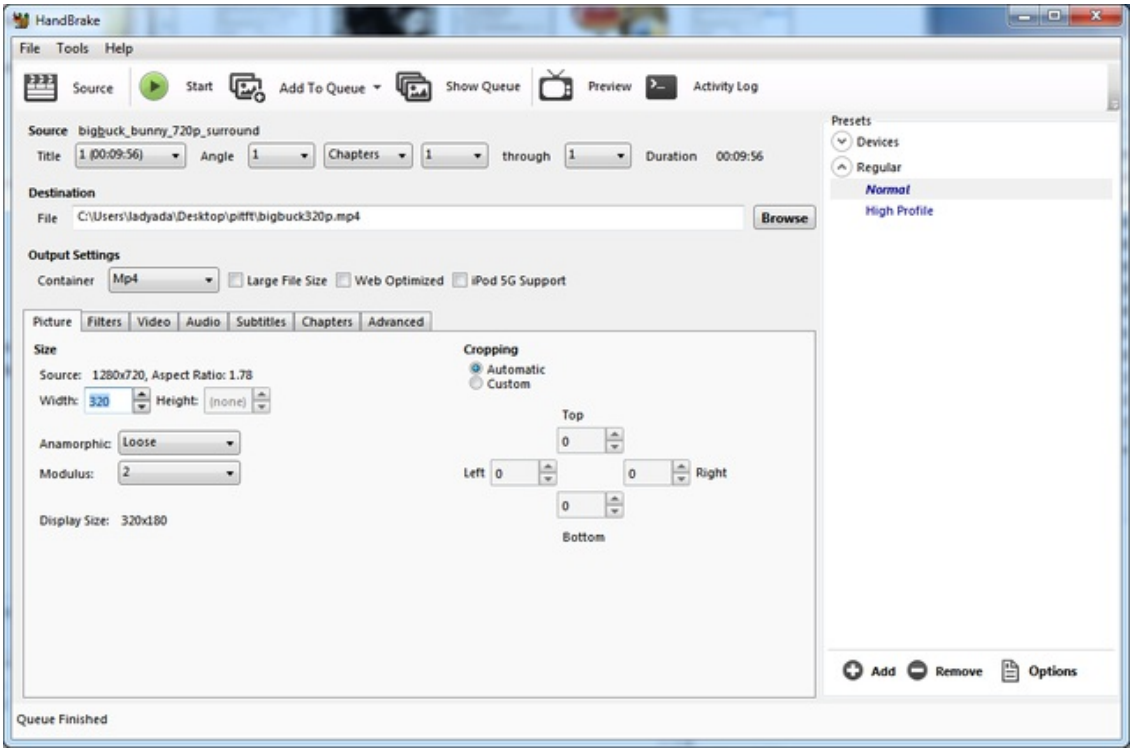
You can download it from the link above, we'll be using the 720p AVI version.



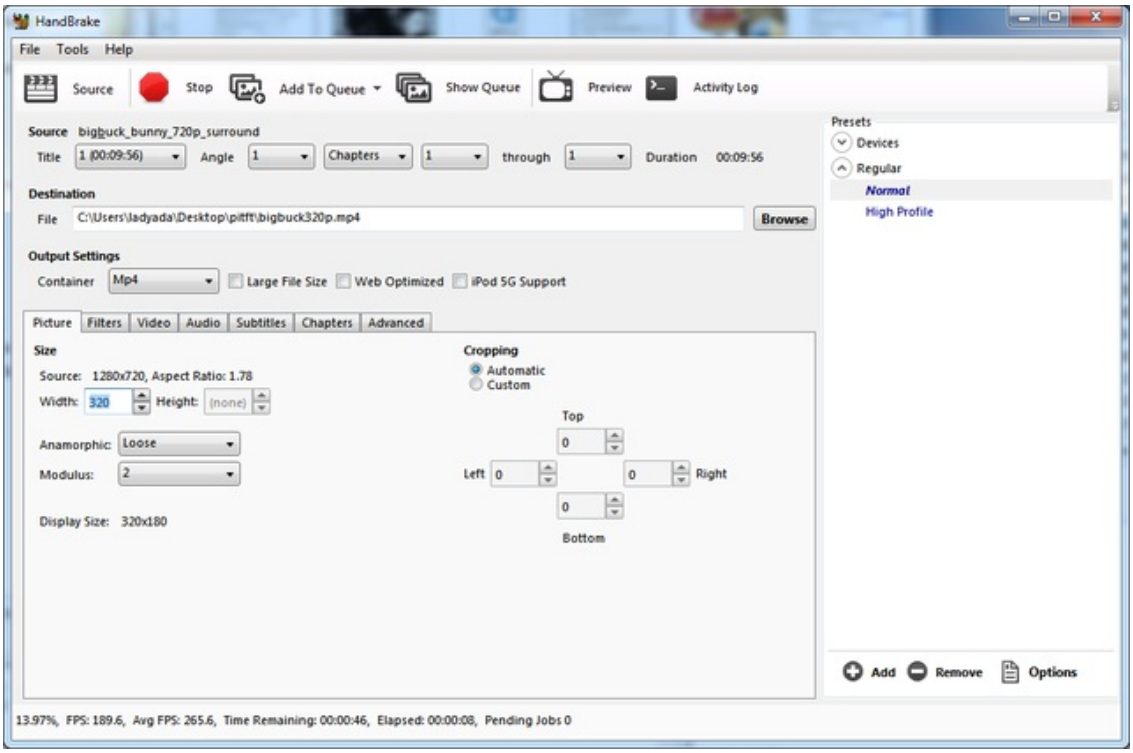
To do the conversion itself, we suggest HandBrake (http://adafru.it/cXT) which works great and is open source so it runs on all operating systems! Download and install from the link. Then run the installed application and open up the AVI file from before. The app will pre-fill a bunch of information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

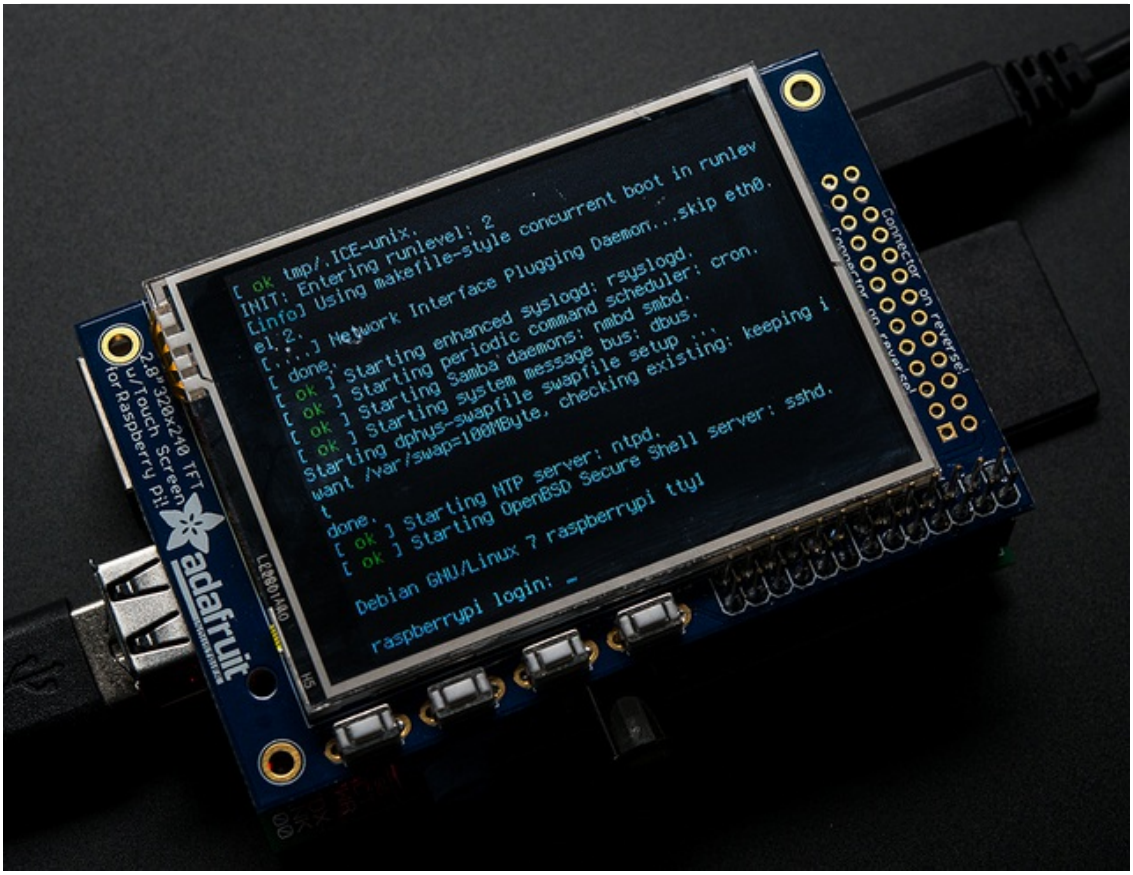


Click **START** to begin the conversion, it will take a minute or two.



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

Using 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 (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

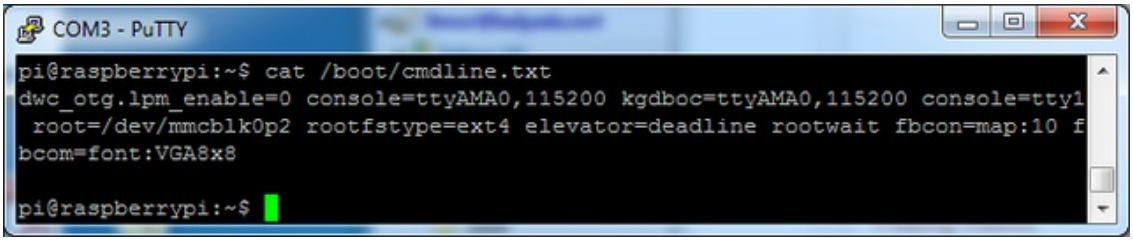
> **sudo nano /boot/cmdline.txt**

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

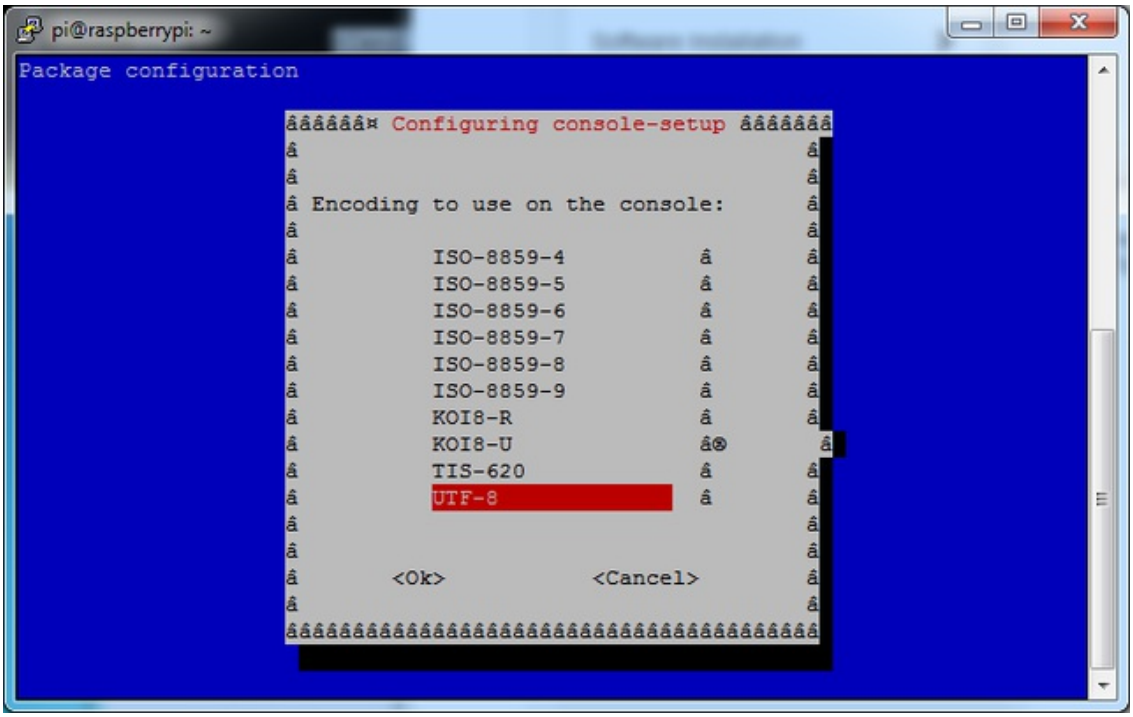
At the end of the line, find the text that says **rootwait** and right after that, enter in: **fbcon=map:10 fbcon=font:VGA8x8** then save the file.

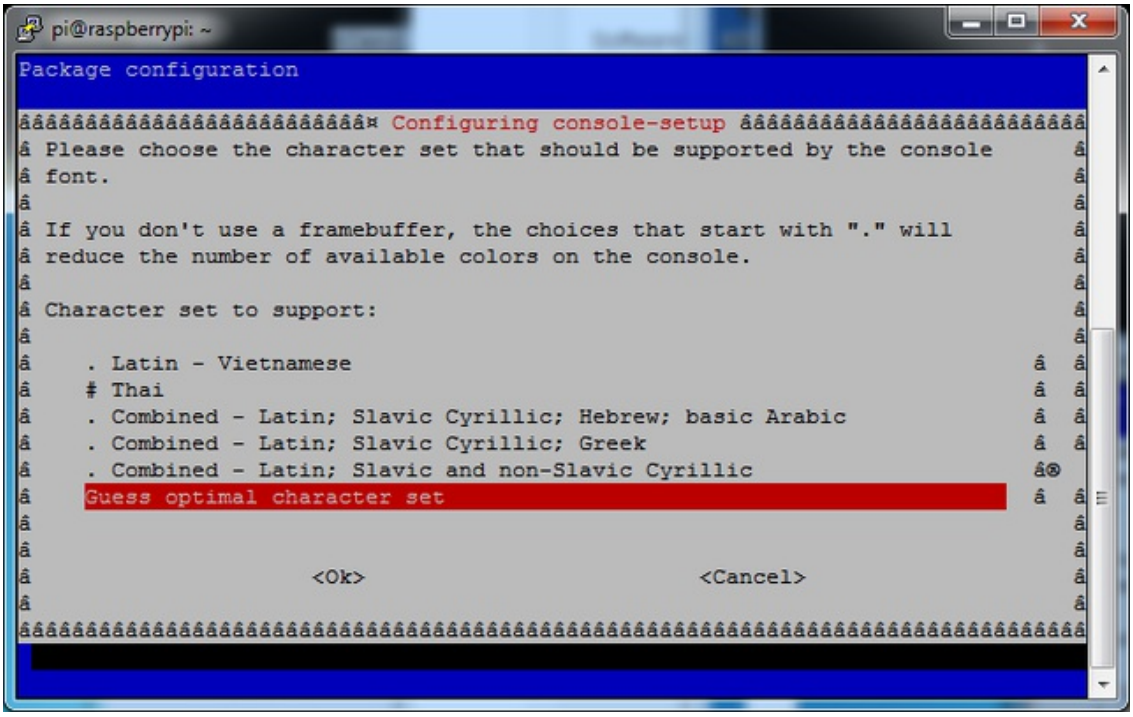
On the next boot, it will bring up the console.

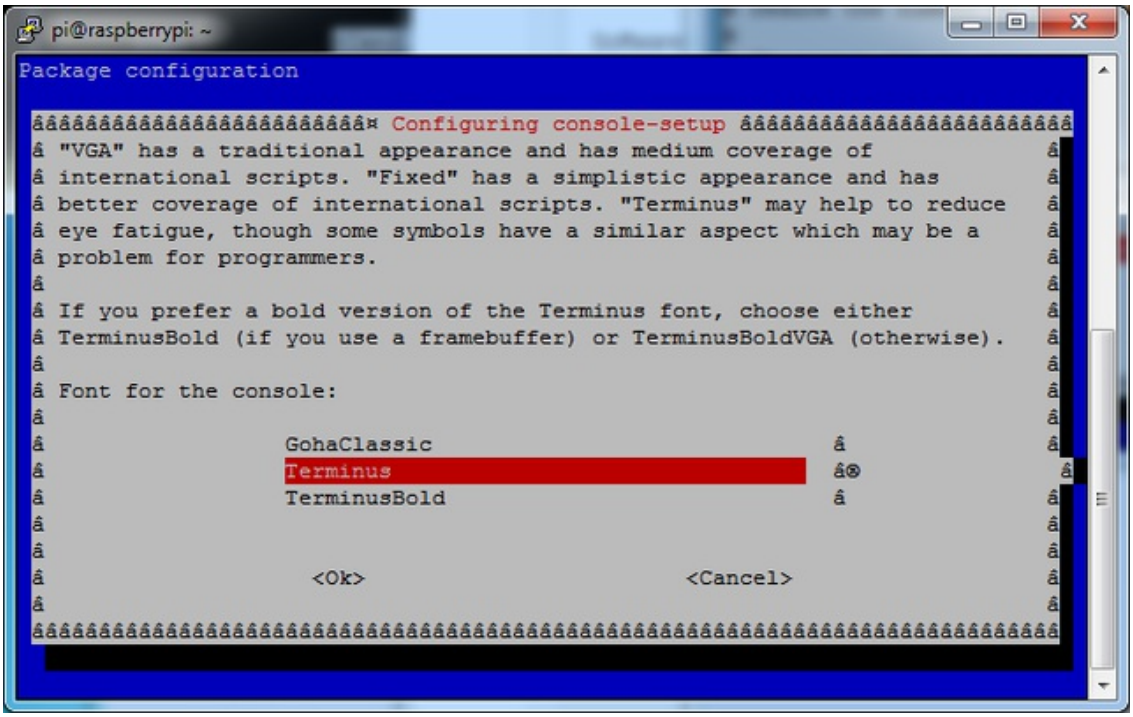
Note that the kernel has to load up the display driver module before it can display anything on it so you won't get the rainbow screen, a NooBs prompt, or a big chunk of the kernel details since the module is loaded fairly late in the boot process.

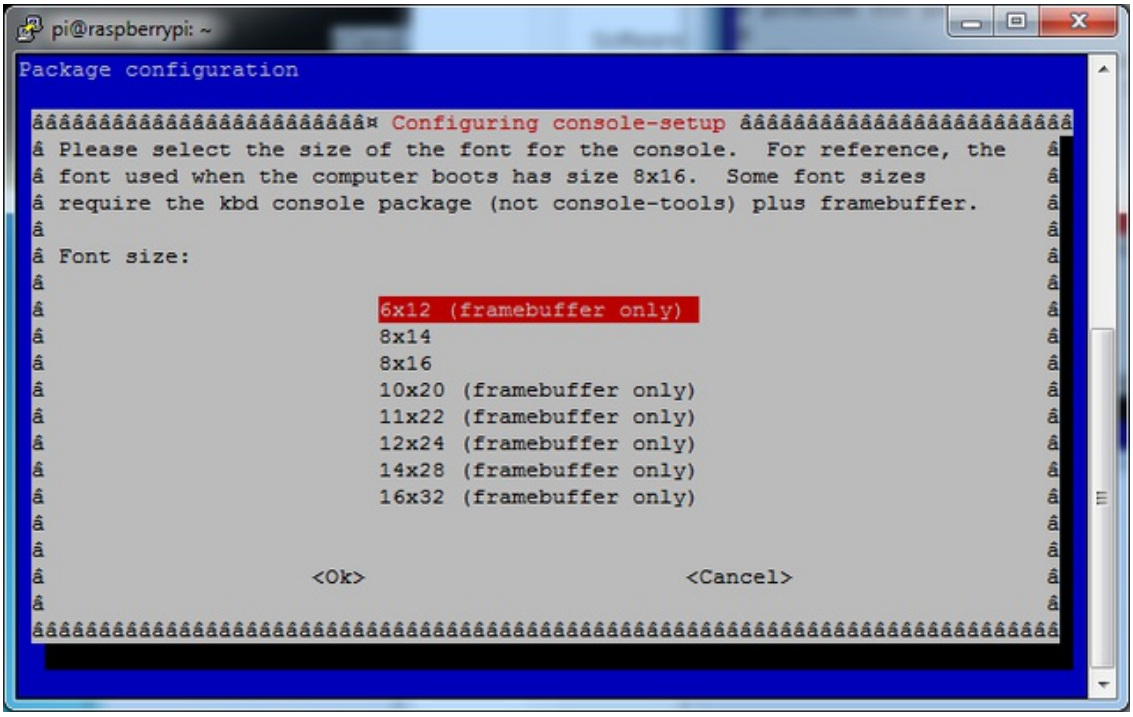


I think the VGA8x8 font is a bit chunky, you probably want 12x6 which is what is shown in the photo above. To change the font, run **sudo dpkg-reconfigure console-setup** and go thru to select Terminus 6x12




pi@raspberrypi: ~
Package configuration
Configuring console-setup
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>

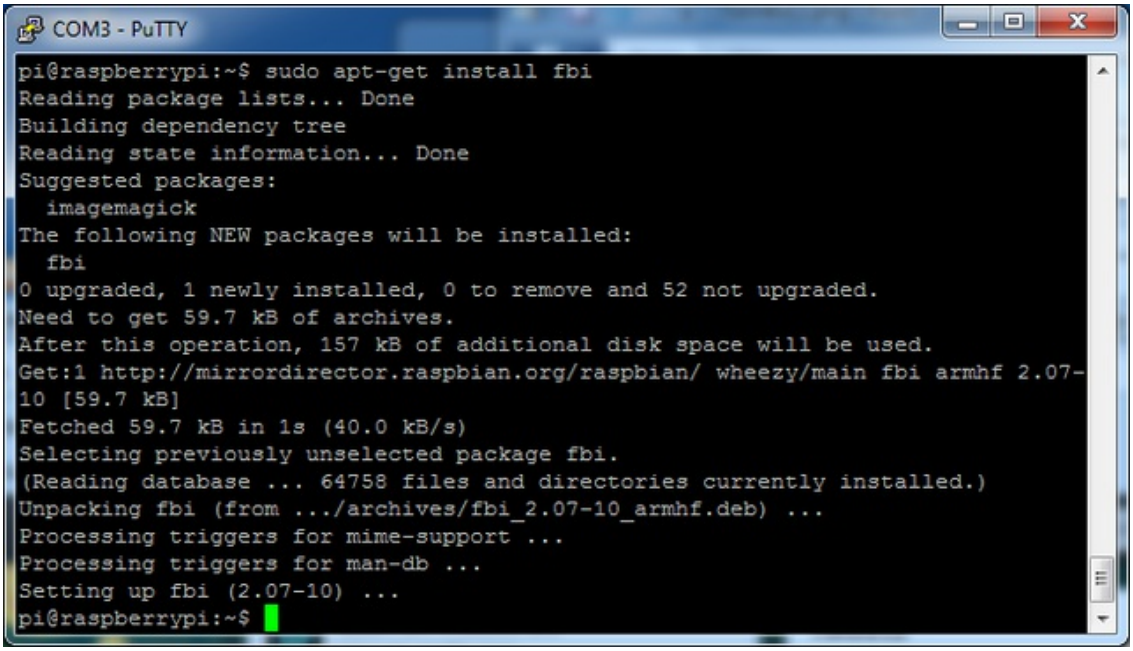

pi@raspberrypi: ~
Package configuration
Configuring console-setup
"VGA" has a traditional appearance and has medium coverage of international scripts. "Fixed" has a simplistic appearance and has better coverage of international scripts. "Terminus" may help to reduce eye fatigue, though some symbols have a similar aspect which may be a problem for programmers.
If you prefer a bold version of the Terminus font, choose either TerminusBold (if you use a framebuffer) or TerminusBoldVGA (otherwise).
Font for the console:
GohaClassic
Terminus
TerminusBold
<Ok>
<Cancel>

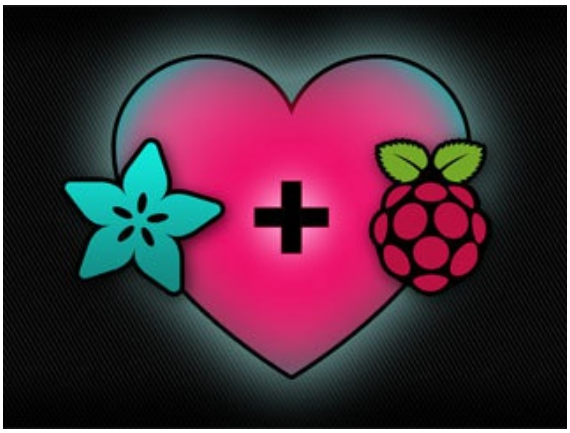

pi@raspberrypi: ~
Package configuration
Configuring console-setup
Please select the size of the font for the console. For reference, the font used when the computer boots has size 8x16. Some font sizes require the kbd console package (not console-tools) plus framebuffer.
Font size:
6x12 (framebuffer only)
8x14
8x16
10x20 (framebuffer only)
11x22 (framebuffer only)
12x24 (framebuffer only)
14x28 (framebuffer only)
16x32 (framebuffer only)
<Ok>
<Cancel>

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





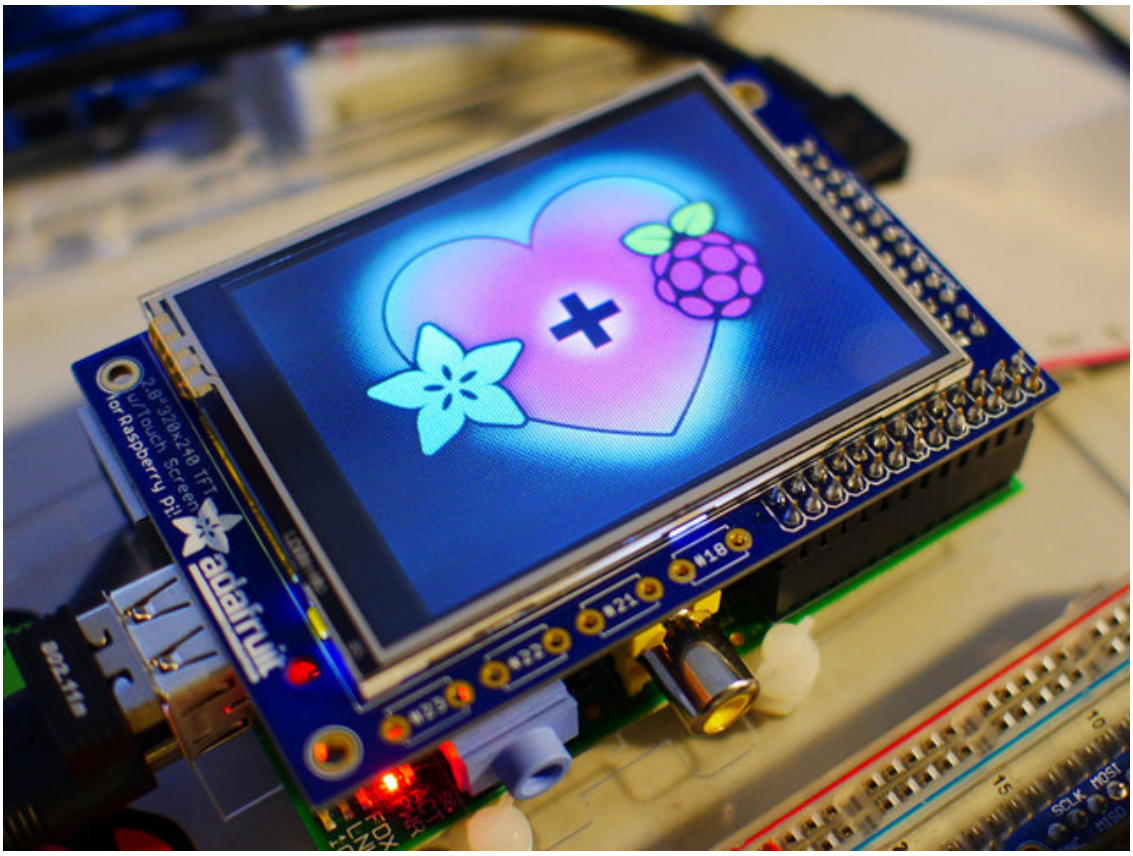
Grab our lovely wallpaper with

> **wget http://adafruit-download.s3.amazonaws.com/adapiluv320x240.jpg** (http://adafru.it/cXU)

and view it with

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

That's it!


2.8" 320x240 TFT
w/Touch Screen
for Raspberry Pi!
adafruit
GND
N23
N22
N21
N18
SCLK
MOSI

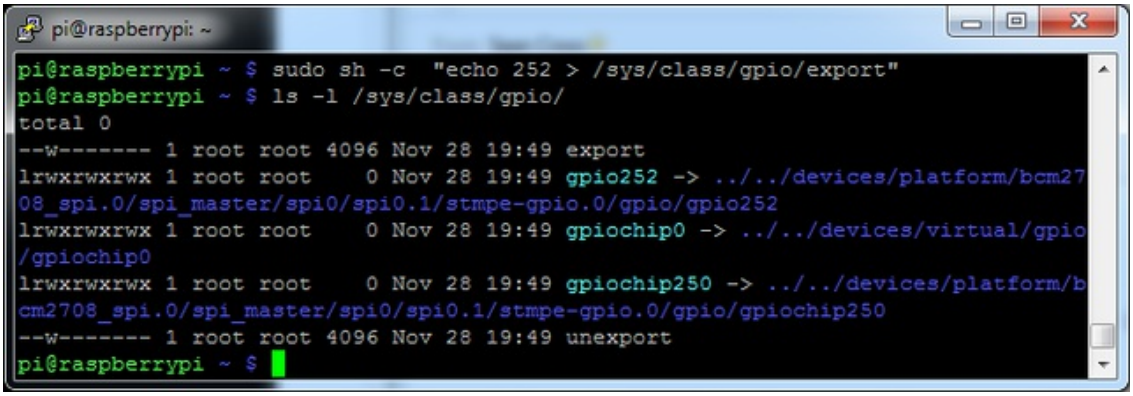
Extras!

Controlling the backlight

There's a 4-LED backlight on the TFT and it draws ~75mA at all times. 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 the transistor that controls the backlight. You can use the command line to control the backlight.

Start by getting access to the GPIO by making a device link

```
sudo sh -c "echo 252 > /sys/class/gpio/export"
ls -l /sys/class/gpio
```



Once you verify that you see GPIO #252, then you can set it to an output, this will turn off the display since it will output 0 by default

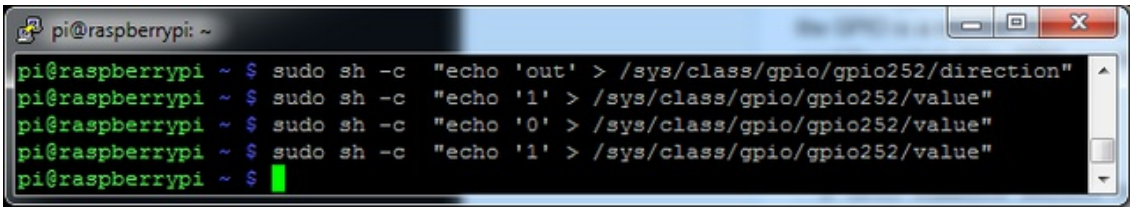
```
sudo sh -c "echo 'out' > /sys/class/gpio/gpio252/direction"
```

Then turn the display back on with

```
sudo sh -c "echo '1' > /sys/class/gpio/gpio252/value"
```

or back off

```
sudo sh -c "echo '0' > /sys/class/gpio/gpio252/value"
```



Tactile switch as power button

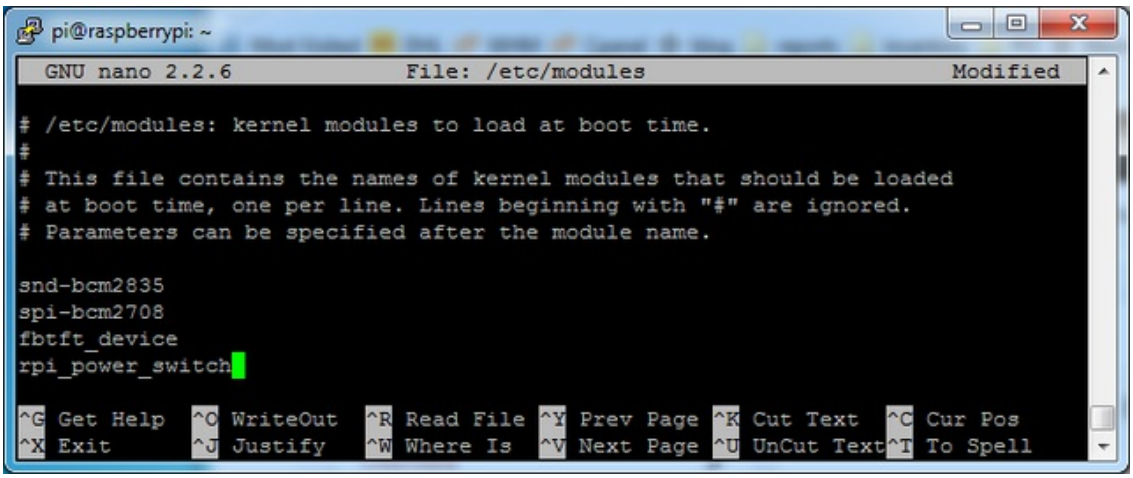
Its a good idea to safely turn off your Pi with a good **sudo shutdown -h now** but that often means pulling out a keyboard or connecting to the console. With our kernel we added

a cool module that will let you turn any GPIO into a power button. Since there's a couple of tactile switches right there on the front, lets turn one into a power button. Press once to properly turn off the pi, press again to start it up. Isn't that nice?

We'll be using GPIO #23, the left-most button. You can use any of them or other GPIO but #23's our favorite number anyways.

You will have to grab a pack of slim tactile switches (http://adafru.it/1489) or otherwise solder in a button

Add **rpi_power_switch** to **/etc/modules** and save



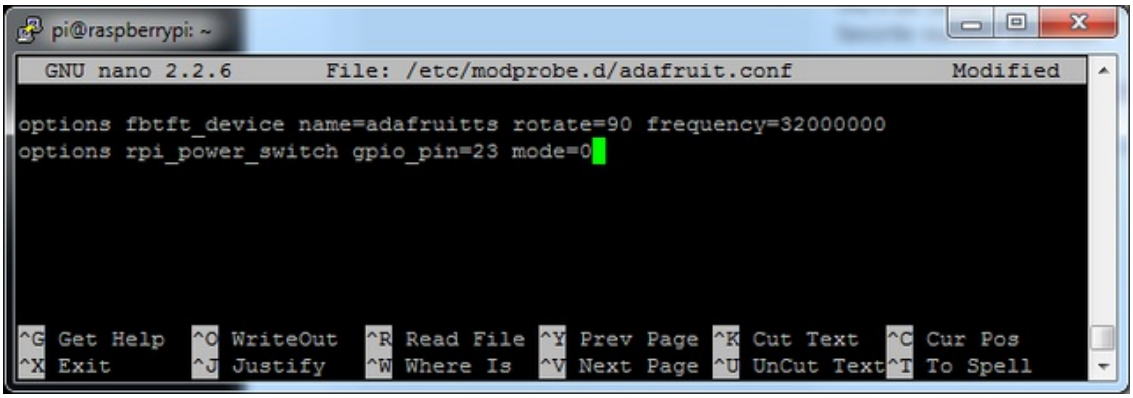
Now create a new conf file or edit our existing one with

```
sudo nano /etc/modprobe.d/adafruit.conf
```

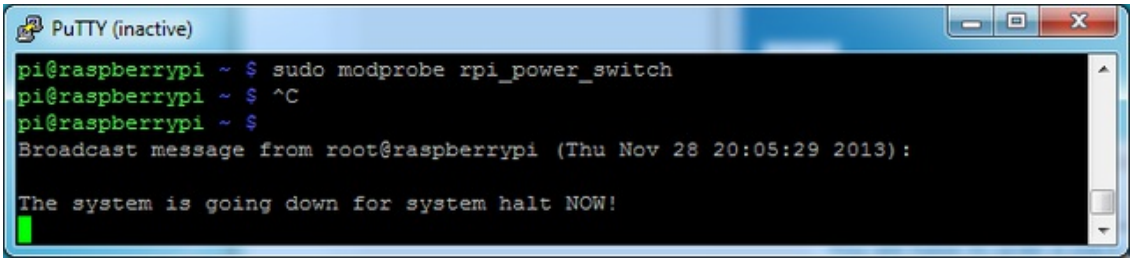
and enter in the line

```
options rpi_power_switch gpio_pin=23 mode=0
```

Of course, change the **gpio_pin** setting to some other # if you wish. **mode=0** means its a pushbutton *not* a switch. If you happen to install an on/off switch, use **mode=1**



To make it active immediately run **sudo modprobe rpi_power_switch**

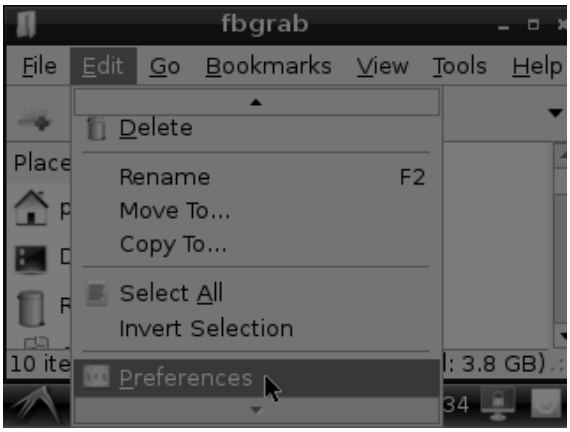


Making it easier to click icons in X

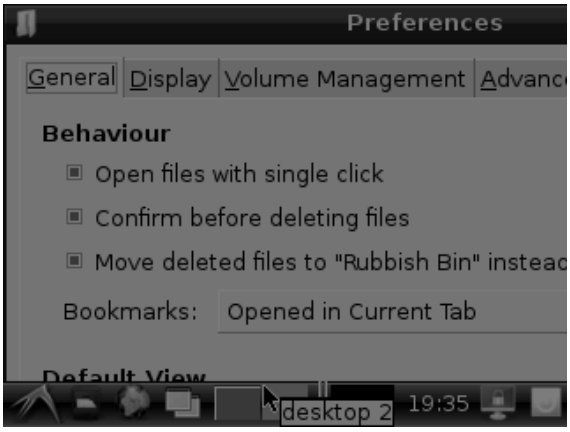
If you want to double-click on icons to launch something in X you may find it annoying to get it to work right. In LXDE you can simply set it up so that you only need to single click instead of double.

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


Preferences
General
Display
Volume Management
Behaviour
Open files with single click
Confirm before deleting files
Move deleted files to "Rubbish Bin" instead
Bookmarks:
Opened in Current Tab
desktop 2
19:35

FAQ

How can I bring up X on the HDMI/TV monitor?
Use the **fb0** frambuffer 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

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!
Advanced users! Want to beta test our new DMA-enabled kernel? Its even faster! Instead of the last wget item - grab the Feb 2014 kernel deb file with

> **wget http://adafruit-download.s3.amazonaws.com/raspberrypi-bootloader-adafruit-20140227-1.deb** (http://adafru.it/day)

and run

> **sudo dpkg -i -B *.deb**

to install. You can always install this over the 11-26-13 version or go back and forth

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=adafruitts rotate=90 frequency=64000000 fps=60**

You can tweak fps (frames per second) from 20 to 60 and frequency up to 80000000 for tradeoffs in performance and speed. Reboot after each edit to make sure the settings are loaded properly

How can I take screenshots of the little screen?
We took the screenshots for this tutorial with
(http://adafru.it/diV)**fbgra (http://adafru.it/diV)b (http://adafru.it/diV)**

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

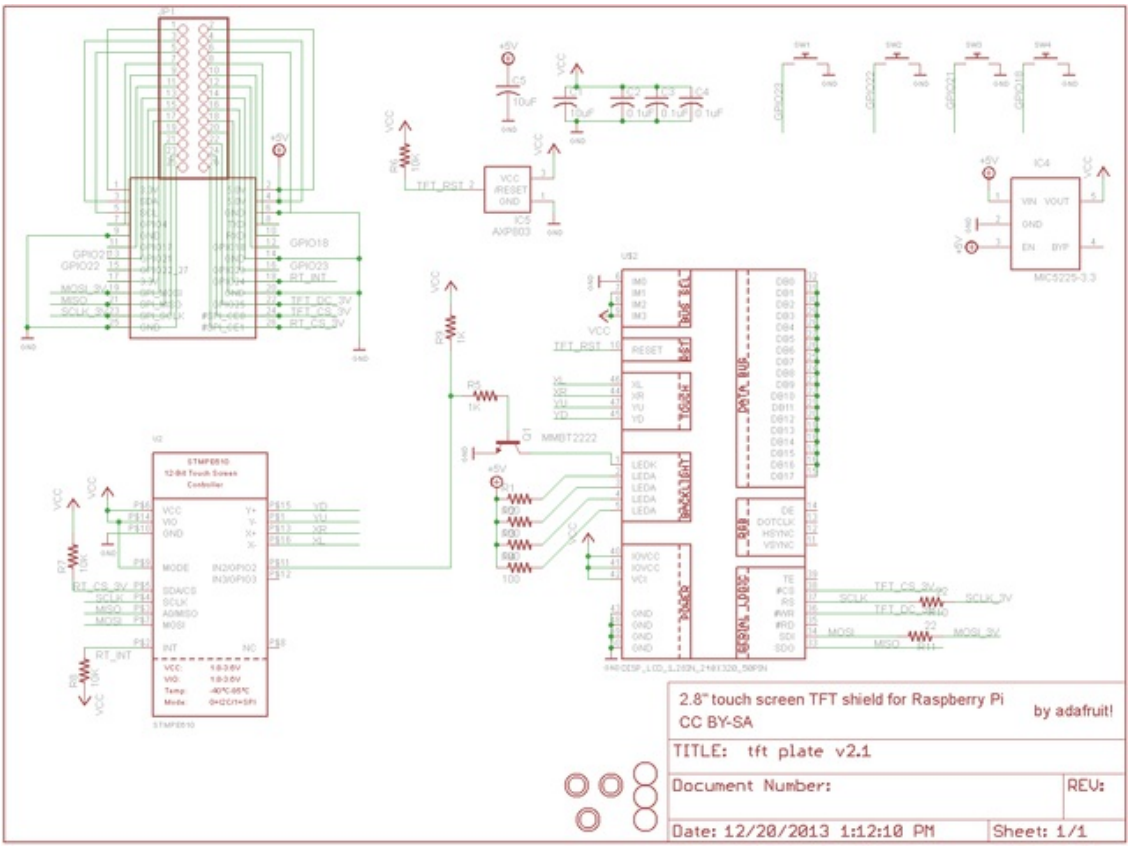
COM3 - PuTTY

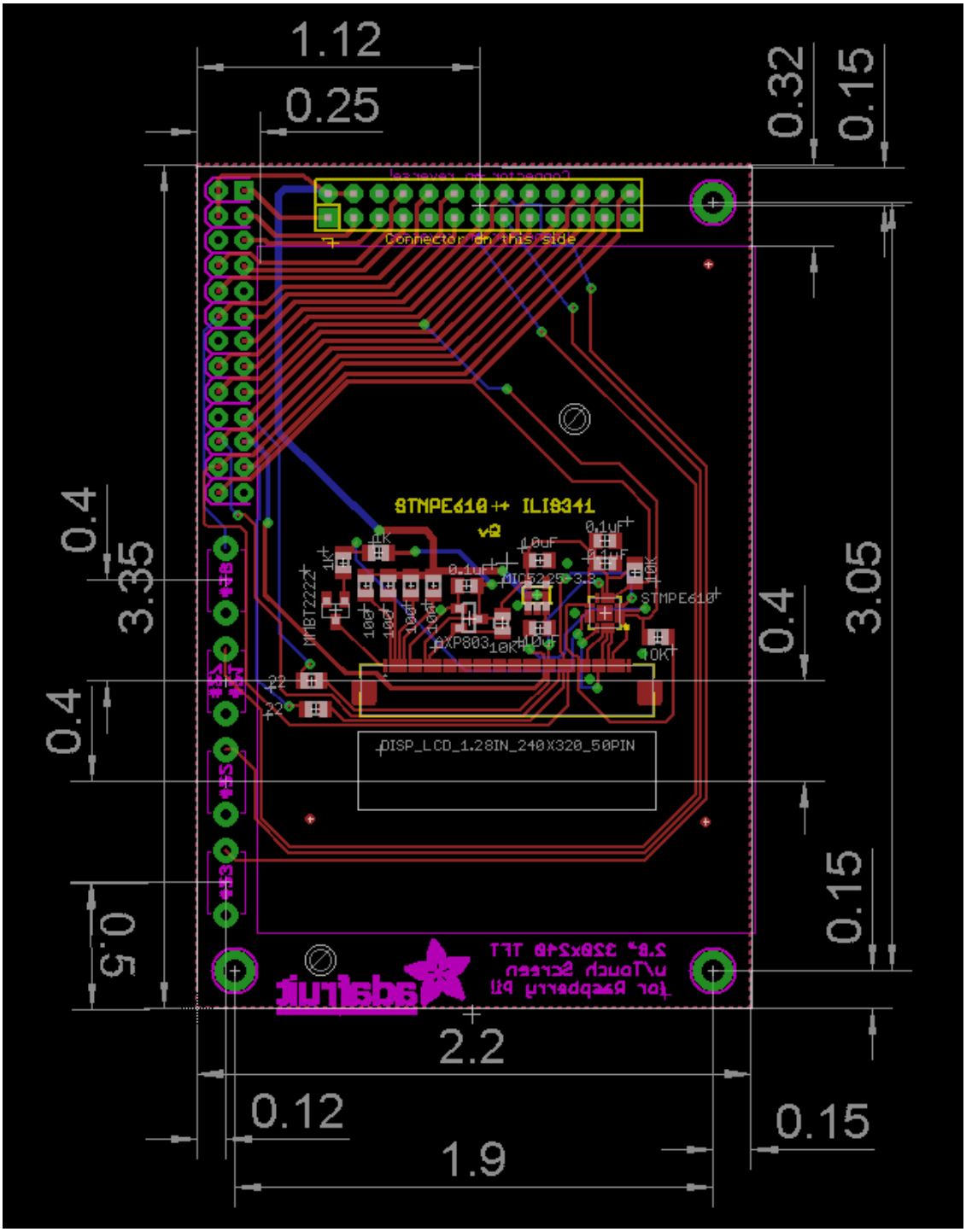
```
pi@raspberrypi:~$ wget http://fbgrab.monells.se/fbgrab-1.2.tar.gz
--2014-04-21 19:26:22--  http://fbgrab.monells.se/fbgrab-1.2.tar.gz
Resolving fbgrab.monells.se (fbgrab.monells.se)... 66.33.214.148
Connecting to fbgrab.monells.se (fbgrab.monells.se)|66.33.214.148|:80... connect
ed.
HTTP request 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/dcA)
- Datasheet for the 'raw' 2.8" TFT display (http://adafru.it/d4m)

Schematic & Layout




1.12
0.25
0.32
0.15
Connector on this side
STMPE610 + ILI9341
v2
0.1uF
10uF
1K
0.1uF
PIC5225-3.3
10K
STMPE610
MMBT2222
10uF
10uF
10uF
10uF
AXP803
10K
10uF
10K
22
22
DISP_LCD_1.28IN_240X320_50PIN
0.4
3.35
0.4
0.5
0.4
3.05
0.15
2.2
0.12
1.9
0.15