ECE 5725: Lab 1

Outline

In this first lab, we will go through the following steps:

- Part 1:
  - Assemble the lab kit for ECE5725
  - Using the NOOBS install on the SD card, install and configure the Raspbian Linux kernel
- Part 2:
  - Install and configure drivers and software for the piTFT display
  - Install and configure drivers and software for video and audio playback
- Part 3:
  - Develop Python scripts to control playback on the screen using physical buttons installed on the piTFT board

Lab safety

We have all been in lab before and handled electronic components. Please apply all the cautions that you have learned including the items below:
Integrated electronics are REALLY susceptible to static discharge.

- Avoid walking around while holding bare components
- Confine assembly to the grounded mats on the lab bench
- Make sure you ground yourself by staying in contact with the mat.

Personal safety

- Safety goggles are REQUIRED for soldering

Experimental Safety

- If something on your board is
- Really hot
- Smoking
- Just doesn't look right
- Immediately unplug power
- Train yourself so that this is your first reaction – know where the power switch/cutoff is for your experiment.

Parts of the ECE5725 kit

Raspberry Pi 3 Model B
5V, 2.5A power adapter (mini USB)
8 GByte SD card (mini card in SD carrier)
2.8 inch TFT screen
USB Keyboard and Mouse
DVI connected display
DVI to HDMI adapter
40 pin breakout cable
Ethernet cable

Notes before you start:

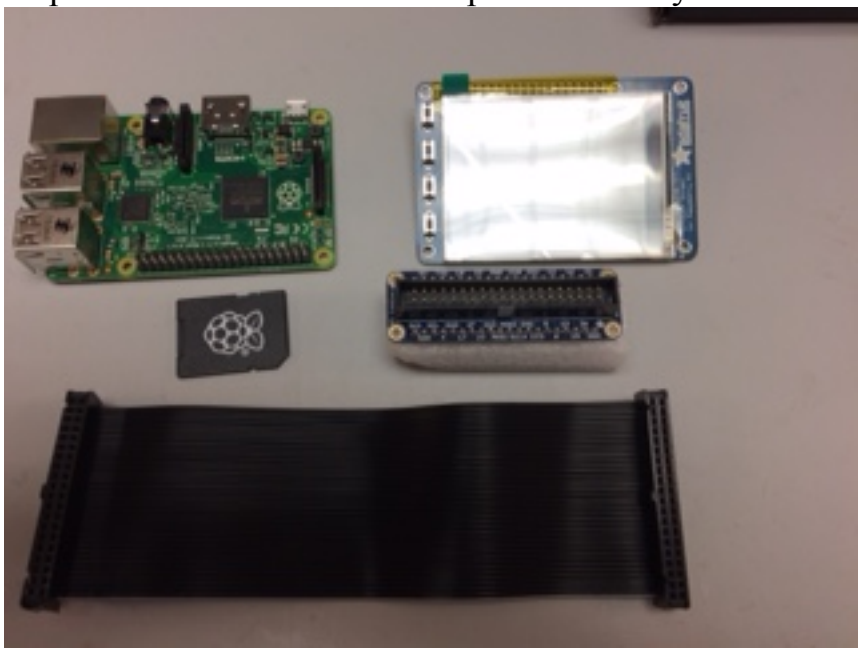
- The R-Pi is powered up at the END of this assembly AFTER you show your assembly to the TA.
- The 40-pin GPIO connectors are NOT keyed. This means they will fit in a number of different orientations. Check that your assembly matches the photos AND check your assembly with the TA before applying power.

Initial assembly and start up:
In the following order, assemble all the components (please don't forget to work on the grounded mat). Here is a photo of the initial system elements:

Assemble your system using the following steps:

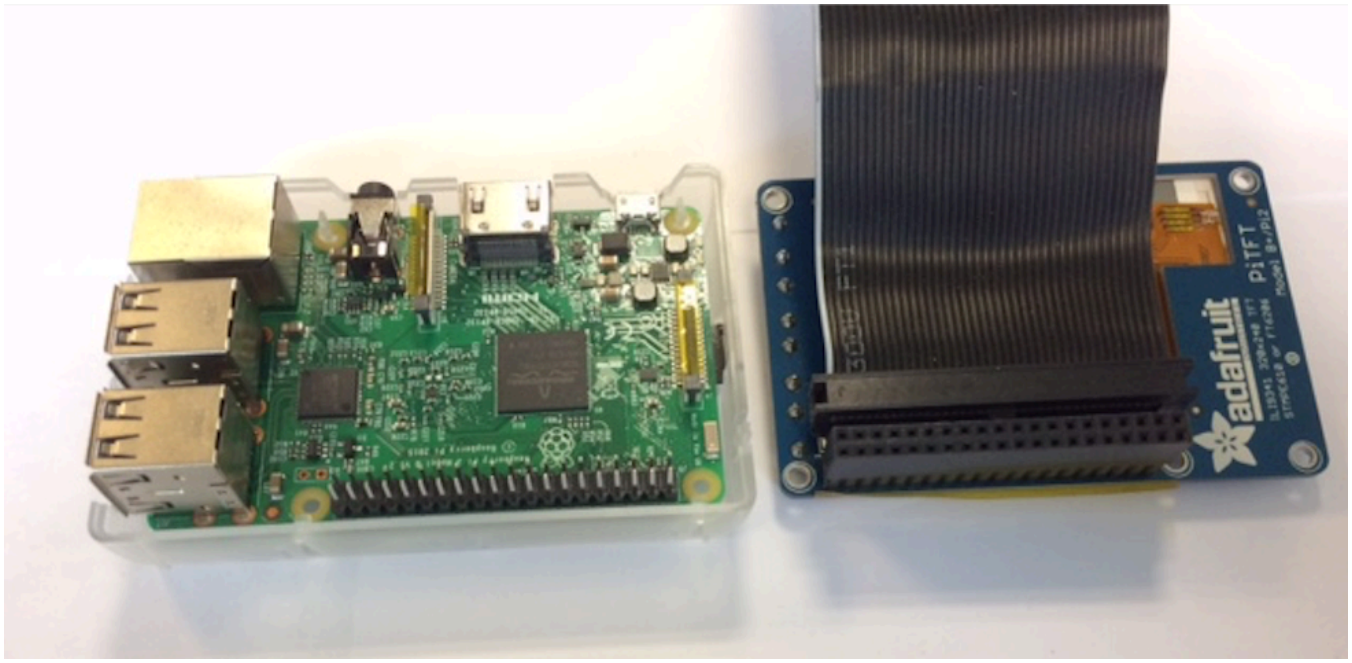
- Unpack the R-Pi
- Install the R-Pi in the bottom of the R-Pi case. To install the RPi in the case, slide one edge under the plastic tabs of the case, then snap the 2 clips through the RPi mounting holes. Note that the top of the case will not be used. The photos below show the RPi and case:

- Unpack the TFT
- Unpack the breakout cable. A photo of the system elements:



- Install the breakout cable on the underside of the TFT (the reverse of the display side). Shown below is the underside of the piTFT showing the connectors. The breakout cable is shown on the left side of the piTFT:

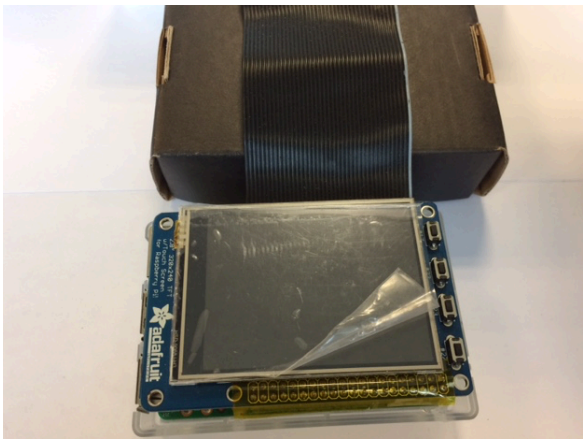
- Plug the TFT into the RPi 40 pin GPIO connector.
  Here is a photo of the piTFT, with the cable correctly plugged, shown next to the RPi.

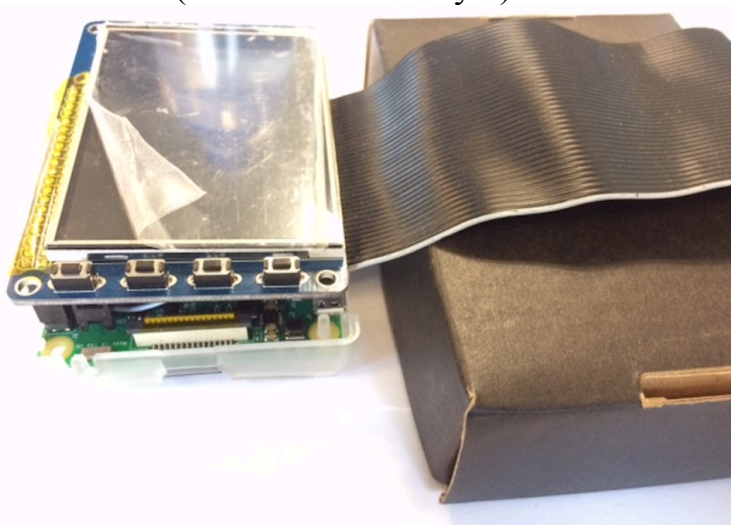


Flip the screen onto the RPi and carefully plug the screen into the 40 pin connector on the RPi. Your completed R-Pi, PiTFT screen, and breakout cable should look like the following photos.

Note that the white stripe on the cable is positioned next to the small ‘1’ on the pin side of the piTFT. The white stripe on the cable should be under the buttons on the side of the TFT screen.

Note that the breakout cable header is not installed (do not install this yet):





- Extract the miniSD Card from the carrier and insert in the bottom slot of the R-Pi. Note the correct position of the mini-SD.
- Plug the display into the HDMI port using the DVI-HDMI adapter and DVI cable.
- Plug into the USB ports:
  - Mouse
  - Keyboard
- Plug in the Ethernet cable

Before you power-up your system:

- Show the TA your setup and get a power adapter
- Plug in the mini-USB connector from the power adapter
- Plug adapter into 110V outlet

You must cycle thru display settings using the on-display buttons to make sure ‘DVI’ is selected for display.

You should see the R-Pi booting

Note: to power off the R-Pi

- Issue the command ‘sudo shutdown –h now’ which instructs Linux to enter a ‘shutdown’ state.

- Once on-screen messages are completed and all LEDs on the R-Pi stop flashing, unplug power. Note that the flashing green led on the RPi indicates that the RPi is accessing the SD card
- It is best to unplug the 110V power adapter rather than the mini-USB adapter (on the R-Pi board).

Initial Raspbian Kernel install

As part of Homework 1, you should have setup your SD card with the correct version of the Raspbian kernel; We will be using the latest Jessie release, 4.9.35, as experiments with the most recent 'Stretch' version of the Raspbial kernel does not yet have support for the PiTFT screen.

The system should boot to the Raspbian desktop. We will use the raspi-config tool which will start a configuration utility to assist in setting up the Raspbian kernel.

There are some raspi-config details in the following link:

https://www.raspberrypi.org/documentation/configuration/raspi-config.md

general configuration link:
https://www.raspberrypi.org/documentation/configuration/

Raspi-config helps with some standard Linux install tasks that you can also run via the Linux command line, or by modifying Linux config files.

An initial install of the Raspbian Jessie kernel booted directly to the Raspbian Desktop (kernel version 4.9.35-v7+). If this happens, try the following:

- Find the terminal icon and open a terminal window
- Issue the command ' uname –a' to see version of installed OS
- issue the command 'sudo raspi-config'
- In the 'boot options' select 'Desktop/CLI' then 'Console'

This change will boot directly to the console window on the next RPi reboot. Since you are already in raspi-config, continue with the notes below for more configuration.

Working your way through the config screens, please make sure to apply the following settings:

- Change the pi user password to something you will remember!
- Hostname: set a unique hostname to 'netid_netid' where 'netid_netid' are the Cornell netids of the two team members in your group.
- Boot Options: Set the system to boot to a command line (may have already done this, above)
- For Internationalization options:
    - Change locale:
        - Use the Spacebar and tab to navigate
        - De-select 'en_GB.UTF-8 UTF-8'. Set the Locale 'en-us.UTF8 UTF8'. When prompted, select 'None' for the Default Locale
    - Change Timezone
        - Set the local time to America
        - Set the location to 'New York'
    - Change Keyboard Layout
        - 'Generic 101 key PC' keyboard
        - select 'Other', then 'English US'
        - 'configuring keyboard-configuration' take 'The default for the keyboard layout'
        - 'no compose key'
        - Select "Ctrl-Alt-backspace to exit startx"
    - Change WIFI country
        - Select 'US United States'

Why these settings? These are the best settings for the time, keyboard and mouse you will be using

- Interfacing Options: enable ssh
- In the 'Advanced' settings
    - A1, expand the file system
    - A4 Audio: Force 3.5 mm jack
- Update: update this tool to the latest version

Once you hit 'Finish', raspi-config will save your changes and ask if you want to reboot now. Answer 'yes'.

Note that you can always run raspi-config to adjust configuration settings at a later time.

If all is well, you will see the R-Pi boot to the Raspbian kernel.

Result of uname –a after this step:

```
pi@jfs9_RPi3:~ $ hostname
jfs9_RPi3
pi@jfs9_RPi3:~ $ uname -a
Linux jfs9_RPi3 4.9.35-v7+ #1014 SMP Fri Jun 30 14:47:43 BST 2017
armv7l GNU/Linux
```

Result of cat /etc/os-release:

```
pi@jfs9_RPi3:~ $ cat /etc/os-release
PRETTY_NAME="Raspbian GNU/Linux 8 (jessie)"
NAME="Raspbian GNU/Linux"
VERSION_ID="8"
VERSION="8 (jessie)"
ID=raspbian
ID_LIKE=debian
HOME_URL="http://www.raspbian.org/"
SUPPORT_URL="http://www.raspbian.org/RaspbianForums"
BUG_REPORT_URL=http://www.raspbian.org/RaspbianBugs
```

STOP: Check with the TA that the kernel has been successfully installed and is booting

Updates

Once the system boots to the command line, login as the pi user

Run 'ifconfig –a' and check the entry for eth0 to determine the Ethernet address; save the Ethernet address so you can log in remotely using putty or a terminal window (for example, iTerm on Mac)

There have been some examples of really slow updates to RPi because of a single application. To avoid this in lab, run the command:
'sudo apt-get remove wolfram-engine'. This may always be installed later, if needed.

Check for Raspbian kernel upgrades:

- ‘sudo apt-get update’
- ‘sudo apt-get upgrade’

If you watch the screen during these operations, you will notice the packages that are loaded and installed to update the kernel.

The upgrade could take 3-5 minutes. Once upgrade is completed:

Reboot to start the upgraded kernel

Once updates are done, run:

‘sudo reboot’ # to reboot the system

You can also shut down the system cleanly using:
‘sudo shutdown –r now’
To restart the system after a shutdown, cycle power on the RPi.

Note: ‘sudo shutdown –h now’ should always be run before you unplug the R-Pi setup. The shutdown command allows Linux to gracefully shutdown all processes before the system is powered down.

have a look at the ‘shutdown’ command to explore the –r and –h flags

Run ‘uname –a’ to show the updated kernel version. Here is an example:

```
pi@jfs9_RPi3:~ $ hostname
jfs9_RPi3
pi@jfs9_RPi3:~ $ uname -a
Linux jfs9_RPi3 4.9.35-v7+ #1014 SMP Fri Jun 30 14:47:43 BST 2017
armv7l GNU/Linux
```

You can also make sure you have loaded Jessie, not wheezy, by looking at the /etc/os-release file. Here is an example:

```
pi@jfs9_RPi3:~ $ cat /etc/os-release
PRETTY_NAME="Raspbian GNU/Linux 8 (jessie)"
NAME="Raspbian GNU/Linux"
VERSION_ID="8"
VERSION="8 (jessie)"
ID=raspbian
```

```
ID_LIKE=debian
HOME_URL="http://www.raspbian.org/"
SUPPORT_URL="http://www.raspbian.org/RaspbianForums"
BUG_REPORT_URL="http://www.raspbian.org/RaspbianBugs"
```

Note - NEVER run:
'Sudo rpi-update' (This loads the latest, un-released, un-tested, development kernel)
… just a warning, because you might see this as you wander the R-Pi pages

Load an update for the vim editor:
'sudo apt-get install vim'

Load the htop application
'sudo apt-get install htop'

Some simple checks:

- 'hostname'
- run 'startx' and make sure the desktop starts
- try 'ctrl-alt-backspace' to exit startx
- 'ifconfig –a' to check network connections
- 'cat /proc/cpuinfo' to check the number of cores
- 'time'
- 'date'
- 'top'

On the R-pi, run 'ifconfig –a'. Note the IP address for the 'eth0' Ethernet adapter. Using this ip address, you can log in remotely using your laptop. An example of the ssh command: 'ssh pi@nnn.mmm.aa.bb Where 'pi' is the user name and 'nnn.mmm.aa.bb' is the ip address for the RPi.

From ifconfig –a, record the MAC addresses of the eth0 (wired Ethernet) and wlan0 (WiFi adapter). Using this link:

https://dnsdb.cit.cornell.edu/dnsdb-cgi/mycomputers.cgi

Add in both devices by selecting 'Add Device with no Browser' at the bottom of the page. This will add both network adapters to the list of devices you are using at Cornell, allowing access to features used on the Cornell networks.

Using putty (on windows) or terminal (on mac), login to the R-Pi to make sure you can! From your laptop, you will use secure shell. An example (on a Mac) is:

ssh pi@123.456.78.90

Where pi is the username and 123.456.78.90 is the IP of your Ethernet (etho) connection.

Note: The raspberry Pi does indeed have a built-in WIFI adapter. In general wired Ethernet will be much faster and is preferred for downloading big kernels, patches, or other large hunks of software.

A Word of caution: You can load a lot of programs on RPi. Sometimes, the installs can drop into many different directories and create large files. In later labs, we might need additional free space to proceed and it may be difficult to recall what was installed and how to get rid of it. You are indeed encouraged to explore. Here are some tips for keeping your SD card trim:

- You may want to consider a second SD card for loading experimental applications
- Keep a list of all installed apps (and how to completely remove them)
- Use list of best practices for good SD card behavior
- Keep backups as you go…restore one of the cleaner backups during a later lab if you are running out of space.

At this point, the RPi contains a clean install of the latest Raspbian kernel.

Backup the SD card!

You will thank us later……

TFT Screen Install

From startx, and a terminal window, go through detailed instructions in:
Detailed Installation:
https://learn.adafruit.com/adafruit-pitft-28-inch-resistive-touchscreen-display-raspberry-pi/software-installation

To install the screen, we will be using the instructions from the above link; the instructions are quite detailed and include a lot of good discussion. As we were testing the instructions, we have discovered some issues. Please go through the indicated pages using the guidelines below. The guidelines will indicate fixes to the instructions.

To setup the piTFT, you will go through 4 pages from the Adafruit Detailed Installation instructions; from the left side index, the pages are:

Detailed Installation
Resistive Touchscreen Manual Install & Calibrate
Console Configuration
Playing Videos

Beginning with the Detailed Installation page:

Detailed Installation:
https://learn.adafruit.com/adafruit-pitft-28-inch-resistive-touchscreen-display-raspberry-pi/software-installation

First step is a sudo apt-get update

Next step will load the kernel (the 'curl' followed by the bootloader install). Notes on these instructions:

- The 20 minute installation time is more like 1 to 3 minutes.
- The sample screens show a wheezy installation rather than a jessie installation.

Next, update config.txt with the following entries:

```
[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=pitft28-resistive,rotate=90,speed=32000000,fps=20
```

Note the change to the 'dtoverlay' setting on the final line.

After this change, follow the directions to:

- Reboot the system
- Check startx on the piTFT
- Modify /etc/modules to include touch support at boot
- Setup initial calibration

NOTE: Once you run FRAMEBUFFER-/dev/fb1 startx from the guide, check that the touch screen responds to taps….

- SKIP the setup of FRAMEBUFFER in the .profile file

General notes:

- The piTFT kernel maps on top of Raspbian. Once you start down this path, do not run 'sudo apt-get upgrade' because this will overwrite the piTFT kernel you are working on.
- The piTFT kernel is loaded from adafruit.com
- Note that the instructions use the 'nano' editor. Vi or vim are alternative editors that may also be used.
- The /boot/config.txt file will be altered for piTFT…alteration to steps used by the bootloader.
- The screen will be rotated and redirected as Framebuffer 1.
- Modules will be set to load at R-Pi startup
- Rough calibration will be set (in 99-calibration.conf)

Resistive Touchscreen Manual Install & Calibrate
https://learn.adafruit.com/adafruit-pitft-28-inch-resistive-touchscreen-display-raspberry-pi/touchscreen-install-and-calibrate

These instructions are correct for piTFT setup. In short:

- Setup a shorthand rule for the touch screen (udev rule)
- Test this with rmmod and modprobe: removes and reinstalls touchscreen driver
- Install and run evtest to check touch on piTFT (note that evtest may already be installed)
- Calibrate the touch screen
    - Important for later in the lab and use of touchscreen 'buttons'
    - Try the 'AutoMagic' python calibration. Note that AutoMagic modifies the default calibration parameters, set earlier, for your specific screen and RPi.

On at least one example test setup, the AutoMagic code was not installed. Try these steps if the script is not found:

```
wget https://raw.githubusercontent.com/adafruit/PiTFT_Extras/master/pitft_touch_cal.py
```

This command grabs the script from Github

```
sudo python pitft_touch_cal.py
```

This command runs the utility. You should see a report about updated calibration for your particular setup. Accept the results and calibration is complete.

Note that if AutoCalibration runs, there is no need to run manual calibration.

Console Configuration :
https://learn.adafruit.com/adafruit-pitft-28-inch-resistive-touchscreen-display-raspberry-pi/using-the-console
These instructions are correct for piTFT installation. In short:

- Edit cmdline.txt and add the 'fbcon' entries to the end of the file
- Make the changes to adjust the piTFT console font
- Turn off console blanking

Playing Videos :
https://learn.adafruit.com/adafruit-pitft-28-inch-resistive-touchscreen-display-raspberry-pi/playing-videos

Instructions on this page are correct. In short:

- Load the sample video
- Setup the piTFT for video playback

Some additional notes for this step:

Audio Playback

Audio was setup in the earlier raspi-config step to play from the headphone jack. Please make sure that audio playback operates correctly by either using powered speakers (see the TA for a set) or through headphones. Some useful utilities:

Run 'mplayer –ao help' to list all possible audio drivers

Check the page:

https://learn.adafruit.com/playing-sounds-and-using-buttons-with-raspberry-pi/install-audio?view=all

to learn more about alsa-utils including installation.

Add the flags ‘-ao alsa’ to the mplayer call to use alsa utilities

To boost the audio volume run:
‘amixer sset PCM,0 100%’

At this point, the TFT has been successfully configured and you should be at the end of your journey through the Adafruit piTFT links.

Test from several consoles

Test by starting mplayer directly on piTFT
Run ‘startx’ to run desktop on external display. Start mplayer using a console window
Remote login using your laptop. Start the mplayer video remotely

Are there any problems with any of these experiments (for example, does the video always run on the piTFT? Is the audio synchronized correctly?) Let the video run and check as it progresses. Document your findings and discuss any issues with a TA.

Stop and Backup your SD card

Once this part of your lab has been completed, back up your SD card using instructions posted on blackboard.

Once you have created the latest backup, restore the backup to the second SD card. Once restored, switch cards in the RPi and boot the second card. Confirm that this card functions identically to the initial card.

Once your SD card is backed up, please check with the TA:

- Demonstrate the piTFT screen by playing the video and audio
    - Launch mplayer from the console window on piTFT
    - Launch mplayer from a console window in startx
- Show the TA your backup files
- Switch SD cards and launch mplayer from the switched card using the piTFT console window (demonstrate two running SD cards)

Week 2

Controlling video playback with external devices:

Once the initial system is up and running, and video/audio playback are working, we want to explore playback control using external methods. You will be creating a number of scripts for different operations. Save all scripts for later demonstration

Step 1: explore mplayer

The first step is to explore mplayer options for playback control. Type 'mplayer' with no options to see a list of video controls. There are controls for pause, skip ahead and back, and start and stop. Explore these options and decide which of these you might want to map into a control 'panel'.

Step 2: control mplayer with a FIFO

Create a fifo (called 'video_fifo'). Run mplayer and pass appropriate commands ('pause' for example) to the running instance of mplayer using echo commands from a command line screen. Make sure mplayer responds correctly to commands.

Note that one way to proceed would include:

- Boot the RPi to the console on the TFT
- Run startx so that the desktop starts on the monitor
- Open two console windows
- In one console window, run mplayer with appropriate arguments to use the fifo
- In the second window, issue commands to the fifo to control mplayer

For reference on how to use mplayer with a FIFO, look at this link :
http://userbound.com/blog/Mplayers-FIFO/
which describes running mplayer using a FIFO. Please Note that this is a great example page, however, keep in mind that the user is describing his own system. So, commands such as: *mkfifo /home/mil/fifos/mplayer* must be modified to work for your setup.

There is an additional link on the page copied here:
http://www.mplayerhq.hu/DOCS/tech/slave.txt
describing mplayer commands recognized when using the FIFO

Step 3: use Python to control mplayer with a FIFO

Write a python routine (called fifo_test.py) to experiment with sending some sample commands (pause is a simple command to begin with). The python routine should:

- Run in the foreground, waiting for some input from the keyboard
- Recognize a valid command (pause, for example) from the user
- Send the valid command(s) to the FIFO setup and used by mplayer
- Control mplayer as expected given the command descriptions
- Recognize a command which quits the script and returns to the command line

Note that this script should pass commands to a process instance of mplayer that is already running on the piTFT. Clearly, the mplayer process will have to monitor the FIFO for input from the script.

Step 4: Get input from a button connected to GPIO

There are 4 buttons on the piTFT. Using the schematic for the piTFT (see the 'Reference' section on the blackboard page), determine which GPIO pins these buttons are connected to and how they are connected.

With this information, write a python script (called 'one_button.py') that will:

- Use the RPi.GPIO module in python
- Set GPIO numbering to Broadcom
- Initialize one of the buttons correctly
- Monitor the button for a press and display 'Button NN has been pressed' where NN is the pin number of the button

Step 5: Get input from four buttons connected to GPIO

Once the one_button.py routine is working, expand the routine to include all 4 buttons. This routine (called 'four_buttons.py') should

- Be based on one_button.py
- Extend the function to include checks for all 4 buttons being pressed by printing the message 'Button NN has been pressed' where NN is the pin number of the appropriate button.
- For one of the buttons on the 'edge' of the screen, print out 'Button NN has been pressed' and also quit the python program.

Step 6: Control mplayer through a FIFO using a python script

- Create a python routine (named 'video_control.py') which will:
- Setup the 4 buttons correctly for detection when pressed
- Connect the following mplayer actions to the buttons:
    - Pause
    - Fast forward 10 seconds
    - Rewind 10 seconds
    - Quit mplayer
- Start an instance of mplayer and test for correct button operation

This program may be demonstrated on the piTFT (consider running video_control.py in the background then starting mplayer) or by using ctrl-alt-F2 to open a second console on the monitor, stating video_control, then mplayer.

Step 7: bash script

Create a bash script (called 'start_video') to launch mplayer and 'video_control.py'. You should be able to launch this from the command line on the piTFT where the video will begin and you will be able to control the video with the piTFT buttons. 'start_video' should:

- Start 'video_control.py
- Note: Consider paths for scripts, and for the FIFO
- Note: Consider foreground or background operation for 'video_control.py'
- Mplayer should be started next to execute on piTFT

At this point, Save an image of your SD card for Lab 1, Part B

Before completing the lab, demonstrate the following programs for the TA:

- Fifo_test.py
- One_button.py
- Four_buttons.py
- Video_control.py
- Start_video

Take the TA through the operation of each piece of software and explain the function of each program indicating any features of the code.