Lab1 Week1 & Week2 Report

Names: Zixiao Wang, Yue Zhang
NetID: zw579, yz2455
Date: September 28, 2019

Introduction:

We did lab1 through 3 steps:

1. Installed Linux kernell on SD Card. Assembled the Raspberry Pi, SD Card,TFT screen and power supply, then configured Linux kernel on Raspberry Pi.
2. Installed all softwares that support piTFT, configured drivers and setup environment variances for piTFT. Then tested play video and audio playback on both piTFT and monitor.
3. Used different methods to control video playing, including controlling mplayer with a fifo file by sending command via FIFO; controlling mplayer using python to send command via FIFO; testing button interrupt; creating python script that handled interrupt from button(s) to send command to FIFO to control mplayer; creating bash script to automatically start python process for video controlling in background and mplayer process for playing video on foreground.

In a nutshell, we can control video on both TFT and monitor and audio playbacking with 4 buttons using bash script based on the Linux kernel installed and configured beforehand. And all progress is backed up on another SD Card.

Design and Testing:

Below are the steps we took and issues we came into:

1.Format SD Card, rename it and make sure it has correct format type FAT. Then download the correct Linux kernel, a “Buster” Released

by July 12, 2019. Finally load the kernel image onto the SD Card using balenaEtcher.

In this step, everything went well on Mac but failed on Windows 10. Reason is still not clear.

2.Assemble parts of Rasp Pi kit including Raspberry Pi 3 Model B, Power adapter (mini USB), 16GB SD Card, 2.8 inch TFT screen, USB keyboard and mouse, DVI connected monitor display, DVI to HDMI adapter, 40 pin breakout cable and Ethernet cable. After correctly assembling, power up R-Pi.

In this step, we saw R-Pi booting once powered-up.

3.Shutdown and reboot R-Pi using Linux Commands: sudo shutdown -h now / sudo reboot.

In this step, R-Pi was shutdown/rebooted as expected.

4.Use raspi-config tool to configure Raspbian kernel following below steps:

a).In the 'boot options' select 'Desktop/CLI' then 'B1 Console';

b).In 'Change User Password', Change the pi user password;

c).In 'Network Options', select 'Hostname' and set it to yz2455-zw579;

d).In 'Locale' Options, De-select 'en_GB.UTF-8 UTF-8'. Set the Locale 'en-us.UTF8 UTF8'. When prompted, select 'None' for the Default Locale using space bar to (de)select.

e).Set the local time to America, set the location to 'New York', and reboot.

f).Restart and change keyboard layout to 'Generic 101 key PC' keyboard;

g).In Interfacing Options: enable ssh; In the 'Advanced' settings, set A1, expand the file system and A4 Audio: Force 3.5 mm jack, then update this tool to the latest version. Reboot R-Pi.

h).Use ifconfig -a to check entry for eth0 and save Ethernet address.

i).Check for Raspbian kernel upgrades using 'sudo apt-get update' and 'sudo apt-get upgrade'. Then reboot and run 'uname -a' to check kernel version.

j).Check hostname, startx command, network connections, time/date and htop. Then try log in Pi remotely on our laptop. Finally back up the SD Card.

k).Install pip, evtest and libts-bin application to support TFT. Then add piTFT info to config.txt and run dmesg to check for piTFT.

l).Add udev rule to rules.d and reboot. Then restart touchscreen module and test its functionality with touching it and evtest to verify touchscreen operation. Set initial piTFT calibration.

m).Start console window on piTFT by modifying cmdline.txt about fbcon and font. Add entry in re.local to turn off piTFT blanking then reboot the system. Linux console start on piTFT and show up on monitor with startx command.

n).Load sample video online, install mplayer and play video on piTFT, then boost the audio volumn. Finally run startx to display video on monitor.Back up SD Card.

In this step, we came to an issue with select/deselect Locale options because space bar operation is neglected. We have to start over when console wouldn't show up at the end of the lab session. After correcting configuration, everything went as expected: Host name, time/date and Linux version showed up correctly, ssh remote connection worked well, video could play on piTFT or monitor with command startx. Noted that we need to change fb0/1 when launching video on monitor/piTFT.

5.Explore mplayer for playback control commands and control mplayer by echoing commands into FIFO and make sure mplayer respond to these commands.

In this step, we created FIFO using mkfifo. We opened one console for mplayer and another console for sending command via FIFO, both of which are on monitor. Linked mplayer input file to the FIFO we created, then mplayer was able to respond to the commands we sent.

6.Use python to control mplayer via FIFO.

In this step, we run the python script on foreground that took our input as commands to send via FIFO to mplayer. Therefore, mplayer was able to respond to the commands we typed in.

7. Get input from one/four button(s) connected to GPIO using RPi.GPIO module in python. Initialize button(s) correctly according to schematic and monitor button(s) for press.

In this step, text “Button NN has been pressed” showed up on PiTFT when we push the button(s). GPIO number showed up at the place where “NN” is. When button(s) on the edges are pushed, python process quit as expected.

8. Combine previous two steps to create a python routine that setup 4 buttons correctly and send according commands to FIFO when a button is pushed. Commands are Pause, Forward 10s, Rewind 10s and Quit in this case.

In this case, we created video_control.py routine that configure GPIO first and echo commands to FIFO then. After that, we started

video_control.py then an instance of mplayer and tested for button operation. All buttons worked as they are programmed: one for pause/continue, one for forward, one for rewind and one for quit(both mplayer and python routine)

9. Combine all previous steps in one bash script called start_video that launch python routine firstly on background and then mplayer on foreground. When star_video is executed, we should be able to control mplayer with 4 buttons. Then back up SD Card!

In this step, we created a bash script named start_video and started these processes in order and it worked as expected with proper control and nice quit . Then we tried to reverse the order, video controlling still worked well but it would not go back to console when 'quit' button is pushed, which means python process didn't stop and we have to use CTRL + C to kill it.

Conclusions:

What worked and what didn't work during the lab?

At last, all things work well, however, we made some mistakes during the lab and then corrected them.
Things that worked smoothly:

1. Set up and our SD cards with the correct version of the Raspbian kernel.
2. Assembled parts of the ECE5725 kit.
3. Boot Raspbian kernel to Raspbian desktop and configured it.
4. Upgrade the kernel which took a few minutes.
5. Install vim editor and use vim to edit system file later.
6. Switch display platform between piTFT and desktop.

7. backup the SD card and upload the image to another SD card.
8. Installed software to support the piTFT.
9. Add piTFT info to config.txt.
10. Run dmesg to check whether touch screen are installed correctly.
11. Add a udev rule which assign the touch screen an eventX number.
12. Unload the driver and reload it for system to pick up new information.
13. Used evtest to verify touchscreen operation.
14. Set initial piTFT calibration.
15. Start Linux console window on piTFT, and use “startx” could change to desktop.
16. Install mplayer and video, display video with mplayer on piTFT and desktop, boost audio volume.
17. Explore commands for mplayer.
18. Control mplayer with a FIFO by first creating a video_fifo file(fifo type).
19. Use python to control mplayer with a fifo by writing a python routine(fifo_test.py).
20. Get input from one button and four buttons connected to GPIO.
21. Get button information and use python to control mplayer through fifo.
22. Use bash script which combined python file and mplayer command to control mplayer.

Things didn’t work at first but modified at last:

1. Linux console didn’t start on the piTFT at the first time, we once thought it might be caused by incorrectly modifying files, however, after we checked all the file content and configuration, we realized it was caused by localization option which has to be

selected by spacebar. After we selected locale correctly, console display on piTFT.

2. When we control mplayer through fifo using python script, video couldn't display on piTFT at first, after we changed SDL_FBDEV=/dev/fb0 to SDL_FBDEV=/dev/fb1, things worked well.

Were there any issues you experienced during the lab?

Yes, we encountered the linux console didn't start on the piTFT at week 1. But we solved it by modifying configuration in localization options, we didn't notice we have to use spacebar to select at first, after we select it correctly, problem solved.
Besides, we came through video couldn't display on piTFT at week2, then we realized we just forgot to change SDL_FBDEV=/dev/fb0 to SDL_FBDEV=/dev/fb1, after we changed it, things worked well.

Are there any improvements you would suggest?

1. Emphasize some uncommon operations, such as "use spacebar and tab to navigate" , because other configurations are selected by "enter".
2. Add some instructions on how to connect wifi on Raspberry Pii.

Were there any items that could be clearer in the lab description?

1. Add more details on each commands in files to be modified. For example, we modified /boot/cmdline.txt to include "fbcon=map:10 fbcon=font:VGA8*8", if add function of fbcon may be better.

2. Add some comments on configuration, such as the function of “boot options-B1 Console”.

Code:

We have upload our code to server in the directory of /home/Lab1/zw579_yz2455_Lab1.