

Lab1 Readme.pdf

Part 1.1 Pi4 Setup:

Everything for steps one through six went smoothly. I was able to download the pi imager and properly install the ubuntu server image onto the microSD card. There was a little confusion on how to download things and use the pi imager application to get it to work as intended but I just had to play around with it for a few minutes and figure it out by trial and error. For example, originally for the desktop option in pi imager, I had originally thought I had to download the image from the link in lab1.md, but after playing with pi imager for a few minutes I realized that I could click “other general purpose OS” and then “Ubuntu” and could select the server image from the pi imager to download straight to the microSD.

The Pi4 booted up properly and I was prompted with a login. I had changed the username and password when I installed Ubuntu to the microSD so I used those credentials. After logging in I tried to run `sudo apt update` and `sudo apt upgrade` but kept getting a failure to fetch error. After trying the same thing over and over again a few times I realized that I probably couldn't check for updates without an internet connection so I turned it off, plugged in ethernet, and turned it back on and that fixed that problem.

After fixing that, I moved onto step 10, installing xfce4. The installation using the command given in the lab1.md worked without problems and I moved on. In moving onto the other steps I realized my gui was not actually opened even though I had installed it. So I was going to turn it on and off again but then I saw a post on the Piazza about “sudo apt reboot.” I noticed that this command was added at the end of posts where course staff were giving command line prompts to fix other errors. In looking it up I realized it did just what I wanted, turning the Pi on and off again so that the things I installed would be implemented. After using “sudo apt reboot”, I was brought into the gui. I was prompted for a username and password, the same as when I booted into the command line but when I entered the same credentials from before, I was faced with a “failed to start session.” I looked up this error hoping it was a common, fixable one and found this link: <https://askubuntu.com/questions/1440313/ubuntu-server-22-10-failed-to-start-session>. Most importantly, I found how to get out of the gui and back to the command line: “CTRL+ALT+F1”. I tried rebooting a couple of times from the command line to see if it was a one time error or if it would work itself out and it never did. I couldn't seem to find a fix that I felt comfortable trying and I saw this line in the lab1.md: “You can complete the rest of this lab, using the command line, or X-Windows” so I just decided to work in the command line for the rest of this lab.

Finally for the Pi setup, I moved onto setting up an eduroam connection. I started by trying to follow the command given in the lab1.md file but kept getting errors that said something to the effect of “missing proper packages” or “packages not installed correctly”. With the advice of a TA, I reflashed the microSD with the Ubuntu server to get a clean download and then used the sd card in the lab with eduroam.zip. A TA helped with some instructions on commands to run to get the .zip working properly and after all that everything worked properly. I used “ping www.google.com” to demonstrate the connection was working and got the check off.

Part 1.2 Run Hello World in ESP32:

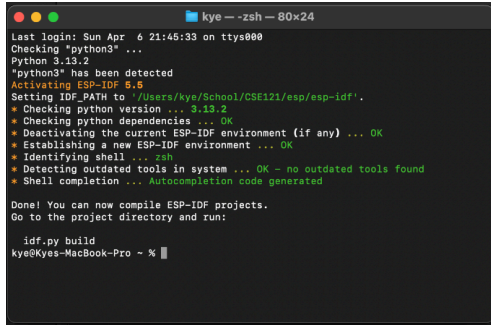
I started this part of the lab working on the Pi4 and intended to do so on the Pi4 but I had a lot of trouble trying to clone the esp-idf repository on it. The problems started with step one of this section, running this

```
command: sudo apt-get install fish neovim g++ git wget \
        flex bison gperf python3 python3-venv cmake \
        ninja-build ccache libffi-dev libssl-dev \
        dfu-util libusb-1.0-0
```

I kept getting an error that went something like “does not recognize the command fish”, “does not recognize the command flex”, “does not recognize the command ninja-build”, “does not recognize the command pthyon” and “does not recognize the command dfu-util”. I took this as those commands were not already installed on my pi so I ran: “sudo apt install XXX” for all of the above but in doing so I got messages saying I had them already installed. So I decided to try running the commands one at a time like: sudo apt-get install fish neovim g++ git wget, sudo apt-get install flex bison gperf python3 python3-venv cmake, sudo apt-get install ninja-build ccache libffi-dev libssl-dev, sudo apt-get install dfu-util libusb-1.0-0 and that worked. I also realized the python error message was because I spelled it wrong. I then tried to run this command: git clone --recursive <https://github.com/espressif/esp-idf.git>. This was the biggest headache of this lab for me. I kept getting “Permission denied (public key).” I looked up the error as it was displayed (and typed previously) and found this link from git: <https://docs.github.com/en/authentication/troubleshooting-ssh/error-permission-denied-publickey?platform=linux>. I saw that it mentioned using a ssh key and I had previous experience with setting those up from previous classes. After setting it up using: ssh-keygen -t ed25519 -C "myemailhere@gmail.com" (<https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent>) I tried again and couldn't get it to work. I kept getting “Permission denied (public key).” After trying to troubleshoot with the git documentation above, I eventually caved and decided to do the rest of the lab on my personal macbook.

The clone of the repo worked as expected on my personal computer. I used the espressif documentation for setting up the environment on a mac seen here: <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/linux-macos-setup.html>. After I got the repo cloned, I moved on to the next step: ./install.sh esp32c3. I kept getting this error: env: bash\r: No such file or directory. After looking up the exact error, I found the reason and a fix. All of the line endings are formatted for a Windows machine, not Mac. So I had to convert the files to work on Mac. I first had to install doc2unix using: “brew install doc2unix”, then use this command to change all of the non-Mac stuff to Mac stuff: find . “-type f -exec dos2unix {} \;” which fixed the problem (<https://stackoverflow.com/questions/29045140/env-bash-r-no-such-file-or-directory>).

I added this line: source /Users/kye/School/CSE121/esp/esp-idf/export.sh to my PATH file (~/.zshrc) so that everytime I open a shell on my local mac terminal or xcode terminal, export.sh runs:

A terminal window titled 'kye --zsh -- 80x24' showing the process of activating ESP-IDF 5.5. The output includes: 'Last login: Sun Apr 6 21:45:33 on ttys000', 'Checking "python3" ...', 'python 3.13.2', '"python3" has been detected', 'Activating ESP-IDF 5.5', 'Setting IDF_PATH to "/Users/kye/School/CSE121/esp/esp-idf"', 'Checking python version ... 3.13.2', 'Checking python dependencies ... OK', 'Deactivating the current ESP-IDF environment (if any) ... OK', 'Establishing a new ESP-IDF environment ... OK', 'Identifying shell ... zsh', 'Detecting outdated tools in system ... OK - no outdated tools found', 'Shell completion ... Autocompletion code generated', 'Done! You can now compile ESP-IDF projects. Go to the project directory and run: idf.py build', and the prompt 'kye@Kyes-MacBook-Pro ~ %'.

(I saw Professor Sifferman do this in lecture and it was included in the lecture notes posted to canvas).

While implementing the commands given in the lab1.md file I only ran into one issue which was this error:

```
kye@Kyes-MacBook-Pro hello_world % idf.py set-target esp32c3
Adding "set-target"'s dependency "fullclean" to list of commands with default set of options.
Executing action: fullclean
Directory '/Users/kye/School/CSE121/esp/hello_world/build' doesn't seem to be a CMake build
directory. Refusing to automatically delete files in this directory. Delete the directory manually to
'clean' it.
```

After reading the error, I noticed the problem was the build folder I generated from a previous flash isn't being deleted for the next build so to remedy this I used the command: `rm -rf build` to remove the old directory to allow for the set-target to work for the current build. Other than that, everything worked.

Part 1.3 Flash LED on ESP32:

For this part of the lab, I followed what Professor Sifferman went over in lecture: copy the blink directory from the cloned espressif idf repo and edit the `sdkconfig.defaults.esp32c3` file to have the proper sdk constraints at build time. I added these lines to the previously mentioned file as done in class:

```
CONFIG_BLINK_LED_STRIP=n
CONFIG_BLINK_GPIO=7
```

When I ran this, the build worked as planned and the LED at GPIO7 began blinking. The only error I ran into for this part was when I changed the name of the file `"blink_example_main.c"` to `"main.c"`. I made this change because at the end of the lab1.md, it asks for main.c. This gave me the error:

```
CMake Generate step failed. Build files cannot be regenerated correctly.
cmake failed with exit code 1, output of the command is in the
/Users/kye/School/CSE121/lab1/lab1_3/build/log/idf_py_stderr_output_3492 and
/Users/kye/School/CSE121/lab1/lab1_3/build/log/idf_py_stdout_output_3492
```

To fix this I clicked (CMD+click) on the error output and found this message:

Cannot find source file:

`/Users/kye/School/CSE121/lab1/lab1_3/main/blink_example_main.c`

This prompted me to look through my files and find where `blink_example_main.c` was being called. I found it in `CMakeList.txt`:

```
idf_component_register(SRCS "blink_example_main.c"  
                        INCLUDE_DIRS ".")
```

In seeing this I changed the `SRCS` to `main.c` and that fixed the issue.