Lab 1 for CSE121, Spring'23

Due Date: 04/12/23

This lab is worth 20 Points. Project check-off takes place during the TA section.

The overall objective of this lab is to setup the Pi4 (lab1.1), and program a couple of simple ESP32 programs using the Pi4 board (lab1.2 and lab1.3). This requires having a wifi setup (eduroam works) in the Pi4.

It is VERY important to submit the report.pdf. If this file is missing, you lose ½ of the points.

Lab1.1: setup Pi4 (10 points)

Lab1.1 requires you to have a working Pi4 with eduroam. The installation should be ubuntu server or archlinux (not the Pi4 default)

1-Use the Pi Imager to flash the microSD

https://www.raspberrypi.com/software/

2-Insert the microSD in the CanaKit USB/uSD

3-Insert USB in the computer & start the raspberrypi imager

Select the UBUNTU 64bit server image (not default one)

Flash the microSD

4-Insert in PI4 the uSD
(connect cables like keyboard, monitor)

login: ubuntu password: ubuntu

NOTE: if you have a serial port to connect to the PI4 UART, this is also a valid setup for this lab. Either keyboard+monitor or UART to Pi4.

5-Setup wifi

This is a bit trickier because the default ubuntu server does not install the required commands to run eduroam. So you must either connect the ethernet port OR use a non-eduroam to connect with wifi (option 2).

# Option 2: (not school eduroam, but home wireless)

https://linuxconfig.org/ubuntu-20-04-connect-to-wifi-from-command-line

6-upgrade ubuntu and install required packages

```
  sudo apt update
  sudo apt upgrade
```

```
# To install a LXDE window manager (not needed until logic analyzer is used)
  sudo apt install lxde xinit firefox
```

(The LXDE will also simplify your wifi setup tests if you are not so used to command line)

7-Connect/setup to eduroam

To connect t eduroam, you must install the nmcli (there may be other options which are OK if you get it working). Notice nmcli is not installed by default. You may need a wired or open wifi (step 5).

```
  sudo apt install network-manager

  nmcli con add type wifi con-name "eduroam" ifname wlan0 ssid "eduroam" wifi-sec.key-mgmt
wpa-eap 802-1x.identity "XXX@ucsc.edu" 802-1x.password "XXX" 802-1x.system-ca-certs yes
802-1x.eap "peap" 802-1x.phase2-auth mschapv2
  nmcli connection up eduroam --ask
```

Lab1.2: Run hello world in ESP32 (5 points)

1-upgrade ubuntu and install required packages

```
  sudo apt update
  sudo apt upgrade
  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
```

2-Get esp32 software toolchain

https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/linux-macos-setup.html

```
  mkdir -p ~/esp
  cd ~/esp
  git clone --recursive https://github.com/espressif/esp-idf.git
  cd ~/esp/esp-idf
  ./install.sh esp32c3
```

WARNING: This is a TOP OF THE TREE checkout. This means that the day that you clone the repo, it may (or not) have a BUG and it does not work. E.g: I cloned 2 months ago, and the export.sh script did not work with bash in OSX, but it was fine with Linux. If you have issues, you can pull a previous version, but you should document in the report.pdf what happened and why. If you create a pull request to ESP32 main repo with the patch, and the pull request gets accepted, you get extra credit (½ lab).

3-Get simple hello_world running

```
  # Setup (once)
  cd ~/esp/esp-idf
  . export.sh
  cp -a examples/get-started/hello_world ~/esp/
```

  # Patch the hello world (print your name, not mine)

  Edit the "hello_world_main.c" so that after printing the "Minimum free heap..." it prints your name. E.g:

```
        print("Jose Renau\n"); // Change Jose Renau for your name
```

```
  # Build
  cd ~/esp/hello_world

  idf.py set-target esp32c3
  cd build
  ninja
```

4-Deploy hello_word

  Connect the ESP32 board to the raspberry PI4 USB
  (Notice ESP32 has a USB-C, must connect to PI4 USB-2 "blue is OK")

```
  idf.py flash
  idf.py monitor
```

  (to stop monitor ctrl+])

Lab 1.3: Flast LED on ESP32 (5 Points)

The ESP32C3 board that has a LED connected to GPIO2. Write a simple C program that flashes the LED on/off once per second.

The app_main should call something like this:
```
 xTaskCreate(blink_task, "blink_task", 2048, NULL, 5, NULL);
```

This means that it needs to use FreeRTOS.

What/How to submit

Create a zip file of your project source code (DO NOT INCLUDE the BUILD DIRECTORY) and upload to Canvas. Lab1.2 and lab1.3 should have different directories (lab1_2 and lab1_3). For example, these are the lab files and directories for my lab1.3.

```
report.pdf
lab1_3/
lab1_3/sdkconfig
lab1_3/sdkconfig.ci
lab1_3/CMakeLists.txt
lab1_3//README.md
lab1_3/main
lab1_3/main/CMakeLists.txt
lab1_3/main/main.c
lab1_2/…. More files here
```

The README.md documents any issue (like not working) you may have. If everything works, just write “everything works” inside the README.md.

There should also be a SINGLE PDF file named report.pdf that includes:

- Any question/answer to GPT or equivalent LLM that you used
- Any google search/code/repo that you used
    - Cut and paste the search query (text or screenshot is fine)
        - E.g: if you use some reddit, cut&paste (or screenshot) where the code comes from. Failing to do so, and then finding a match would be considered academic dishonesty.
    - If you use a specific repo as a result from search, cut and paste the URL
    - You can **ONLY used open repositories that have APACHE or BSD-like license.** Not OK to use GPL or repositories that do not have any explicit license.
        - Using a code repo without keeping a license or using something like GPL is a ZERO for the whole lab.
    - You can NOT use any repo from other UCSC students.

        - It will be considered academic integrity (cheating)
    - **<u>You can NOT share</u>** your query searches. Part of the class is to learn how to look/find
        - If you share, it is also considered cheating