ECE 40862: Software for Embedded Systems

Fall 2023

Lab 3 – Wi-Fi Networking, Touch Pad Inputs, and Sleep/Wake-Up

---

To be done individually; Due by 11:59pm, Thursday, October 26, 2023.

1. Overview

This assignment deals with connecting your ESP32 V2 board to the Internet over WiFi and exploring sleep modes and wake up sources in the ESP32. You will also be using the **RTC** module and the on-board **touch sensor** in this lab. The hardware and software implementation details for this assignment are described in the following sections.

2. Programming Exercise

2.1. Hardware Interfacing

You will use the following hardware components:

- The on-board red LED and on-board Neo Pixel
    - The red LED should indicate if the board is *awake* (ON) or in *sleep* mode (OFF).
    - The Neo Pixel should be controlled by touch-based input (see Table 1).
- One external **push button** as a GPIO input to use as a wake-up source.
- One external **Male-Male jumper wire** connected to a *Touch-enabled* pin. Insert the wire into the header on the board. This page gives information about the Touch-enabled pins: http://docs.micropython.org/en/latest/esp32/quickref.html#capacitive-touch. You can select any of the 10 capacitive-touch enabled pins on the board with the following caveat. The pins indicated on the webpage above are for the ESP32 microcontroller itself. On your board however, only 6 out of the 10 touch-enabled pins are available for use. You can figure out the 6 pins by trial and error (HINT: you will get a 'ValueError' if you try to assign an unavailable pin as a TouchPad input).

2.2. Software Implementation (main.py)

Your program should implement the following functionality.

2.2.1.Connect to the Internet over WiFi

- The program should use the *network* module in MicroPython to connect your ESP32 to a WiFi network using the '*SSID'* and '*Password*' for that network. You can create a WiFi hotspot on your mobile/laptop and connect your ESP32 to the hotspot. Refer to the example given in the MicroPython documentation for the network module **Error! Reference source not found.**

  e.g., if your mobile hotspot has an *SSID*: 'Lenovo-SSID' and *Password*: '12345678', then use this *SSID* and *password* for connecting the ESP32 to the Internet.

  NOTE: You can also connect to any other WiFi network which works directly with an SSID and password. However, be aware that **enterprise networks such as PAL3.0** which require SSID, username, password, certificates, etc., **are unfortunately not supported** in MicroPython.

- Each time the board successfully connects to Wi-Fi, the program should print the SSID it has connected to and the interface's IP address.

```
Connected to Lenovo-SSID
IP Address: 192.168.0.107
```

2.2.2. Display Current Date and Time using Network Time Protocol (NTP)

- Get the current time from the Internet using NTP. The program should fetch the *current date and time* from the NTP server '**pool.ntp.org**' and use it to set the **RTC** (**real time clock**). HINT: check module *ntptime.* Then, *manually* adjust the RTC to convert UTC (Coordinated Universal Time) to the current local time zone in West Lafayette.

- Similar to lab 2, initialize a **hardware timer** and display the current *date* & *time* every **15** seconds. Do not use *time.sleep()*. Instead, use the RTC and Timer interrupt/callback like you did in Lab 2. Format the date and time as shown in the example below:

```
Date: 09/29/2021
Time: 10:00:00 HRS
```

2.2.3. NeoPixel Control by Touch Input

- Initialize the Touchpad-enabled pin connected to the jumper wire and calibrate it by observing how the touchpad pin values change when you physically touch the jumper wire.

- Initialize a second **hardware timer** and read the *touch pin* values every **50 milliseconds** using a Timer interrupt/callback and implement the following pattern. Use calibrated values to detect whether the wire is touched or not.

  - Touch Pin not touched: Neo Pixel should be OFF
  - Touch Pin touched: Neo Pixel should be ON and lit GREEN

2.2.4. Red LED, Deep Sleep, and Different Wake Up Sources

- The red LED should be ON whenever the ESP32 is awake and OFF when it is in sleep mode.

- Use a third hardware timer to put the ESP32 into **deep sleep** every **30 seconds** for a duration of **1 minute**. Print out a message on the terminal before going to sleep like:

```
I am going to sleep for 1 minute.
```

AWAKE duration: **30 seconds** (this will be the duration of the third hardware timer)

SLEEP duration: **1 minute** (do not need a timer for this, check this link for details on deep sleep: https://docs.micropython.org/en/latest/esp32/quickref.html#deep-sleep-mode)

- You will be using two different sources of waking up from deep sleep. Your program should check for both sources and the board should be able to wake up from either source.

2.2.4.1. Wake up Sources

- **Timer Wake Up:** Wake up the ESP32 after the predefined sleep duration (1 minute) and print that it's a timer wake-up.
- **External Wake Up Mode 0:** Configure the switch as an external wake-up source. Pressing the switch within the 1-minute sleep duration should wake up the board and print out it's an EXT0 wake-up. If one-minute passes and no touch is detected, then your board should wake up due to the Timer Wake Up above.

2.3. Overall Program Flow and Sample Output

The board is powered on for the first time. It connects to the Wi-Fi network, fetches the current time from NTP server, displays them after 15s, and turns on the Neo Pixel to green whenever the touch pin wire is touched. It again displays the date and time at 30s and then goes to deep sleep. The program now checks for the push button wake-up signal. If you press the switch or if 1 minute expires, the board wakes up and starts the overall process again. Table 1 shows the LED status functionality. A sample output is also provided here for your understanding. For this sample, we assume that the program started at 10:00:00 hrs. EST.

Table 1. Sleep and Wake Up Program Flow

Mode	Touch Pin	Switch	Red Led	Neo Pixel
Sleep	Not Pressed: No Effect	Not Pressed: No Effect	Off	Off
Sleep	Pressed: No Effect	Pressed: Wake-Up	Off	Off
Awake	Not Pressed: Neo Pixel off	Not Pressed: No Effect	On	Off
Awake	Pressed: Neo Pixel green	Pressed: No Effect	On	Green

Sample Output

```
I (200) wifi: wifi driver task: 3ffe2c34, prio:23, stack:3584, core=0
.........
.........
.........
I (22233) network: CONNECTED
I (22883) event: sta ip: 192.168.0.107, mask: 255.255.255.0, gw: 192.168.0.1
I (22883) network: GOT_IP
Connected to Lenovo-SSID
IP Address: 192.168.0.107

Date: 09/11/2019
```

Wi-Fi connection messages posted by ESP

```
Time: 10:00:15 HRS

Date: 09/11/2019
Time: 10:00:30 HRS

I am going to sleep for 1 minute.

ets Jun  8 2016 00:22:57

rst:0x5 (DEEPSLEEP_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 0, SPIWP:0xee……
……
……
I (0) cpu_start: Starting scheduler on APP CPU.

Woke up due to EXT0 wakeup.

I (200) wifi: wifi driver task: 3ffe2c34, prio:23, stack:3584, core=0

///continues
```

Default Messages by ESP on Wakeup and Reset

3. Submission

Follow the same instructions as lab 2.

4. Grading

We will use manual code inspection for evaluation. If your code is unusually complex or substantially different from expected solutions, you may be asked to attend an office hour and explain your code.

REFERENCES

[1] Getting started with MicroPython on the ESP32 https://docs.micropython.org/en/latest/esp32/tutorial/intro.html

[2] ESP32 PICO MINI 02 Datasheet https://www.espressif.com/sites/default/files/documentation/esp32-pico-mini-02_datasheet_en.pdf

[3] ESP32 Technical Reference Manual https://www.espressif.com/sites/default/files/documentation/esp32_technical_reference_manual_en.pdf

[4] Adafruit HUZZAH32 - ESP32 V2 Feather Online Manual https://learn.adafruit.com/adafruit-esp32-feather-v2

[5] Adafruit ESP32 Feather Pinouts: https://learn.adafruit.com/adafruit-esp32-feather-v2/pinouts

[6] MicroPython GitHub https://github.com/micropython/micropython

[7] REPL http://docs.micropython.org/en/latest/esp8266/tutorial/repl.html

[8] Thonny IDE https://thonny.org

[9] Rshell GitHub https://github.com/dhylands/rshell

[10] Sleep Modes API in C https://docs.espressif.com/projects/esp-idf/en/latest/api-reference/system/sleep_modes.html

[11] Touch Sensor API in C https://docs.espressif.com/projects/esp-idf/en/latest/api-reference/peripherals/touch_pad.html

[12] ESP32 specific functionalities in MicroPython http://docs.micropython.org/en/latest/library/esp32.html