**University of Texas at El Paso**
**Electrical and Computer Engineering Department**

EE 4178/5190 – Laboratory for Microprocessors II

# LAB 0

## LED Lightshow

### Objective:

- Understand how to use the gpio driver library from ESPRESSIF. The lab will consist of creating a sweeper and a led chaser. The sweeper will iterate over multiple LEDs by turning from the lowest to highest bit and then turn off the most significant bit to lowest bit. While led chaser as the name states will have a single led iterating from the lowest to most significant bit. For the sweeper and led chaser use up to 6 GPIOs.

### Bonus:

- For EE4178 is just a bonus and for EE5190 is mandatory
  - Do the led chaser
- Bonus for EE5190
  - Create a light show with different sequences and patterns as you like. Have fun!

### Pre-Lab:

- What is the function that resets the GPIO in into default?
- What function declare if the pin is and output or an input?
- What function turns ON or OFF the output in the selected pin?

*By: Erick Baca and Jesus Minjares*

## C helpful functions

For this lab, there are three main functions from ESPRESSIF that are important. First is to select which GPIO pin is going to be used by the following function. **gpio_pad_select(gpio_num_t gpio_num**). The first parameter gpio_num is the GPIO pin that is going to be used, for instance if we want to use the ***onboard led*** we should put a 2 on **gpio_num** such as **gpio_pad_select(2**).

- **esp_err_t gpio_pad_select_gpio(gpio_num_t gpio_num);**

Next, another important function is the selection of the direction GPIO pin. Therefore we must use **gpio_set_direction(gpio_num_t gpio_num, gpio_mode_t mode)** and for this lab we only care of ***GPIO_MODE_OUTPUT*** as we will be using outputs.

- **esp_err_t gpio_set_direction(gpio_num_t gpio_num, gpio_mode_t mode);**

Lastly, **gpio_set_level(gpio_num_t gpio_num, uint32_t level**) as the name states set the level of the GPIO pin that we pass through the function. Passing a **0** is set to low and **1** is set to high respectively.

- **esp_err_t gpio_set_level(gpio_num_t gpio_num, uint32_t level);** *// set LOGIC high*

The following function **vTaskDelay( const TickType_t xTicksToDelay**) is use to generate delay with ESP32 and it part of FreeRTOS, we will learn more of it in later labs. Therefore, for this lab just use it to generate delay in milliseconds

- **void vTaskDelay(const TickType_t xTicksToDelay);**