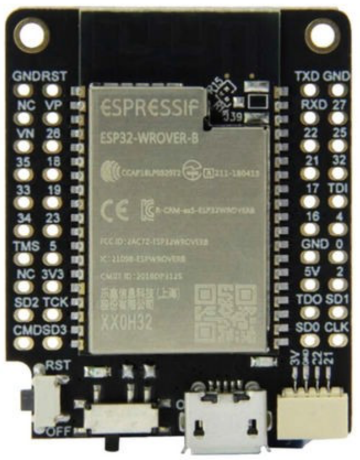
# Course on Internet of Things

## Exercises Session 2: LEDs and NeoPixel

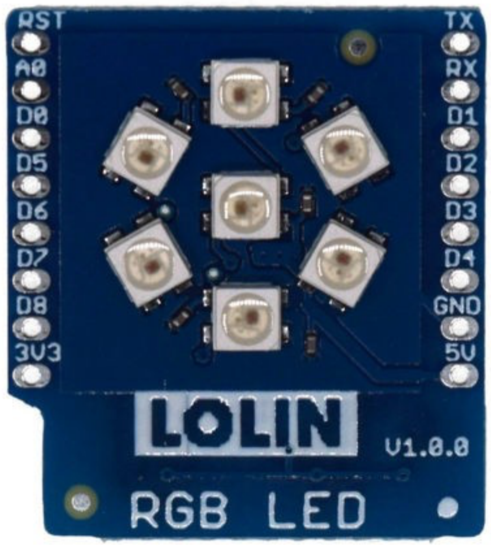
## Introduction:

The ESP32 has 2 LEDs on board. The first one indicates power while the second one is user programmable. The user programmable built-in LED is connected to GPIO pin 19.



You can see the LEDs in the far left, bottom corner. The power LED is marked "ON" while the user programmable one is marked "L".

The NeoPixel is based on the WS2812 addressable RGB LED which is often used in LED chains. In our case we have 6 LEDs arranged in a ring with a 7th one placed in the center of the ring.



Again the LEDs are accessible, this time through a dedicated serial protocol, on a single GPIO line. For the board we are using, the GPIO line can be selected with solder jumpers. If you look very closely you can see that the LED ring uses D0, corresponding to GPIO 26 on the ESP32.

## Further introductory remarks:

In comparison with the first exercise session these exercises are rather simple. This is mainly due to the fact that all difficulties in accessing the devices are hidden from you in the drivers which are part of MicroPython. If you want to know what is going on behind the scene please have a look at https://www.parallax.com/sites/default/files/downloads/28085-WS2812B-RGB-LED-Datasheet.pdf

## Exercise 1: Switching the user LED with REPL

It is actually very easy to access the LED because all you need is already available in MicroPython. Just look it up at
https://docs.micropython.org/en/latest/esp32/quickref.html#pins-and-gpio and try the example using GPIO pin 2. Don't write a script just yet but switch the LED on and off with REPL.

## Exercise 2: The Embedded System's Hello World Program: The blinking LED

Write a script that makes the LED blink at 1 Hz (500ms on, 500 ms off).

Improve the program by capturing Ctrl-C, which stops the endless loop, switching off the LED before exiting the program. This can easily be done in a *try..except* clause capturing the *KeyboardInterrupt* exception.

## Exercise 3: SOS

Probably the most well known Morse sequence is SOS (Save Our Souls) which consists of 3 long sounds (Morse "S") followed by 3 short ones (Morse "O") again followed by 3 long ones. Make the LED blink the SOS sequence. Pause for 1 s between 2 SOS sequences.

## Exercise 4: Change the LED light intensity

The light intensity on the LED can be changed if we do not supply a fixed signal level to it but a frequency. The duty cycle of the signal determines the light intensity. This is called **P**ulse **W**idth **M**odulation or PWM for short. Write a program that increases the intensity in a linear fashion and then decreases it again linearly.

## Exercise 5: The WS2812 addressable LED

The WS2812 LED is often used in LED chains. It is an RGB LED using a single data line which can be cascaded. It has a *data **in*** pin and a *data **out*** pin where the *data out* pin is connected to the *data in pin* of the following LED. It uses a serial protocol that must be precisely timed. For more information please consult the data sheet. Fortunately a driver for the WS2812 running on the ESP32 is already integrate into the !MicroPython binary.

**Warning:** The WS2812 produces very bright light! Do not look into it directly or reduce the light intensity by software.

The RGB LED chain has 7 LEDs installed on its PCB. Write a program that allows you to find out which LED corresponds to which address. To do so, please switch on a single LED e.g. its red component, and print out the corresponding address. Wait for 5s before you switch on the next LED. Only a single LED should light for each of the seven addresses.

## Exercise 6: Change the Colors

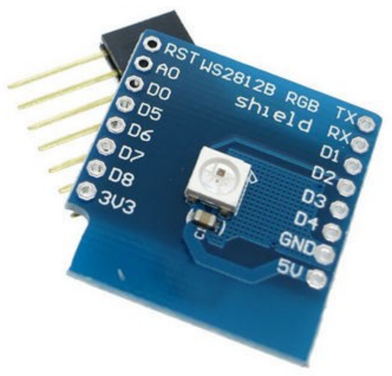
Write a program that consecutively switches on the LEDs in different colors. First switch on the red component of the LED with address 0 then add the one with address 1 etc. Once all LEDs are red, switch them off again, change the color to green and repeat the cycle. Finally do the same for the blue component. Wait for 1 s before switching the LEDs.

## Exercise 7: Change to CCW

In exercise 6 the LEDs will turn on clockwise (cw). Modify the program such that the LEDs turn on counter clock wise (ccw).

## Replacement exercises for Exercise 5-7

The following exercises replace the ones we want to do during the future IoT course where we should have a ring of 7 WE2812 LEDs. In our kit we currently have a shield with only 1 WS2812 and the following exercises restrict themselves to this more basic hardware.



The WS2812 single LED shield works with GPIO 21

## Exercise 5:

The WS2812 has 3 LEDs of red,green and blue color internally. Write a script that switches all LEDs off.

## Exercise 6:

Write a script switching the individual LEDs on one by one:

- red

- green
- blue

and then all of the possible combinations (3 bits <> 8 combinations). Switch them on at only ½ light intensity.

At the end of the program switch the LEDs off again.

## Exercise 7:

Cycle through all possible color combinations showing the full functionality of the WS2812