**Embedded Systems**

*Engr. Jobert DC. Cadiz*

***Prelim Period***

**ACActivity#4: PWM**

***Members:* AGUILAR, BERMAS, CLAMOR, SALAZAR, SANTUYO**

***Group Number: 1 Section: COM801***

***Grade***

**I - Objectives:**

1. Define and attach the PWM GPIO pin.

2. Configure the PWM channel (frequency & resolution)

3. Gradually increase the PWM's duty cycle to max value, and gradually decrease it to the minimum value, and repeat!

4. Compare PWM output of 4bit and 8bit resolution ESP32 thru oscilloscope.

**II – Materials:**

1. Any ESP32 Dev Board

2. Breadboard

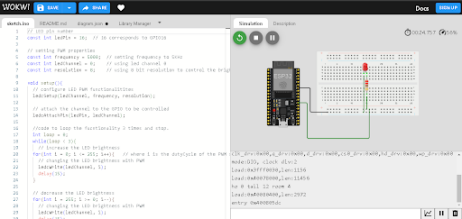
3. Resistors

4. Jumper Wires

5. LEDs

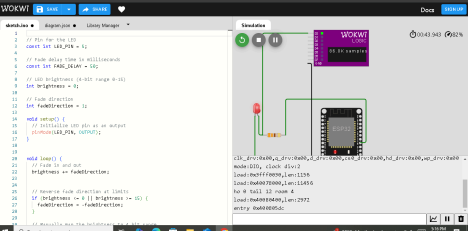
**III – Wokwi Circuit Screenshot**

**8-bit resolution**

****

There are only 10 types of people in the world: those who understand binary and those who don't. Page **1** of **4**

**4-bit resolution**

****

**IV – ESP32 Program Codes**

**Note:** Code should have proper comments: It should include group number, members and explanation of the program code.

**[8-bit resolution]**

**/\*GROUP 1**

**AGUILAR**

**BERMAS**

**CLAMOR**

**SALAZAR**

**SANTUYO**

**\*/**

**// LED pin number**

**const int ledPin = 16; // 16 corresponds to GPIO16**

**// setting PWM properties**

**const int frequency = 5000; // setting frequency to 5KHz**

**const int ledChannel = 0; // using led channel 0**

**const int resolution = 8; // using 8 bit resolution to control the brightness from 0 to 255.**

**void setup(){**

**// configure LED PWM functionalitites**

**ledcSetup(ledChannel, frequency, resolution);**

**// attach the channel to the GPIO to be controlled**

**ledcAttachPin(ledPin, ledChannel);**

Page **2** of **4**

**“STUDY AND PRAY. ALWAYS PUT GOD FIRST”…**

** //code to loop the fucntionality 3 times and stop.**

**int loop = 0;**

**while(loop < 3){**

**// increase the LED brightness**

**for(int i = 0; i <= 255; i++){ // where i is the dutyCycle of the PWM signal.**

**// changing the LED brightness with PWM**

**ledcWrite(ledChannel, i);**

**delay(15);**

**}**

**// decrease the LED brightness**

**for(int i = 255; i >= 0; i--){**

**// changing the LED brightness with PWM**

**ledcWrite(ledChannel, i);**

**delay(15);**

**}**

**loop++;**

**}**

**}**

**void loop(){**

**// To loop continuously place the code here.**

**}**

**[4-bit resolution]**

**/\* GROUP 1**

**AGUILAR**

**BERMAS**

**CLAMOR**

**SALAZAR**

**SANTUYO**

**\*/**

**// Pin for the LED**

**const int LED\_PIN = 5;**

**// Fade delay time in milliseconds**

**const int FADE\_DELAY = 50;**

**// LED brightness (4-bit range 0-15)**

**int brightness = 0;**

**// Fade direction**

**int fadeDirection = 1;**

**void setup() {**

There are only 10 types of people in the world: those who understand binary and those who don't. Page **3** of **4**

**// Initialize LED pin as an output**

**pinMode(LED\_PIN, OUTPUT);**

**}**

**void loop() {**

**// Fade in and out**

**brightness += fadeDirection;**

**// Reverse fade direction at limits**

**if (brightness <= 0 || brightness >= 15) {**

**fadeDirection = -fadeDirection;**

**}**

**// Manually map the brightness to 4 bit range**

**int pwmValue = map(brightness, 0, 15, 0, 15);**

**// Output PWM signal to LED (scaled for 4-bit resolution)**

**analogWrite(LED\_PIN, pwmValue);**

**// Delay for fade effect**

**delay(FADE\_DELAY);**

**}**

**V - Conclusion**

The two code examples demonstrate how to control the brightness of an LED using Pulse Width Modulation (PWM), but they do so with different resolutions. The first code uses an 8-bit resolution, which allows for a much finer control of the LED's brightness, with values ranging from 0 to 255. This means that the LED can be dimmed or brightened in smaller increments, giving a smoother transition. The code gradually increases the brightness from 0 to 255 and then decreases it back to 0, repeating this process three times. This results in a noticeable fade effect, where the LED smoothly transitions between its darkest and brightest states.

On the other hand, the second code uses a 4-bit resolution, which only allows for 16 levels of brightness, ranging from 0 to 15. This creates a less smooth fade effect compared to the 8-bit version, as the LED brightness can only change in larger steps. The code makes the LED gradually brighten and dim in a continuous loop, reversing the fade direction once it reaches the limits (either 0 or 15). While this provides a simpler effect, it is less precise than the 8-bit resolution and gives a more noticeable "stepping" transition between brightness levels.

Page **4** of **4**

**“STUDY AND PRAY. ALWAYS PUT GOD FIRST”…**