1.2

#include <Arduino.h>

#define LED\_BUILTIN 2 // Pin donde está conectado el LED

#define DELAY\_TIME 1000 // Tiempo de espera (500 milisegundos)

void setup() {

// Inicializar el pin del LED como salida

pinMode(LED\_BUILTIN, OUTPUT);

// Iniciar la comunicación serie a 115200 baudios

Serial.begin(115200);

}

void loop() {

// Encender el LED

digitalWrite(LED\_BUILTIN, HIGH);

// Enviar mensaje "ON" por el puerto serie

Serial.println("1");

// Esperar 500 milisegundos

delay(DELAY\_TIME);

// Apagar el LED

digitalWrite(LED\_BUILTIN, LOW);

// Enviar mensaje "OFF" por el puerto serie

Serial.println("O");

// Esperar otros 500 milisegundos

delay(DELAY\_TIME);

}

1.3

#include <Arduino.h>

// Definir el pin al que se conecta el LED

#define LED\_PIN 2 // Pin 2 en ESP32-S3

void setup() {

// Configurar el pin como salida

pinMode(LED\_PIN, OUTPUT);

// Iniciar la comunicación serie

Serial.begin(115200);

}

void loop() {

// Encender el LED utilizando el acceso directo a los registros del ESP32

GPIO.out\_w1ts = (1 << LED\_PIN); // Establecer el bit correspondiente al pin 2 (encender LED)

// Enviar "ON" por el puerto serie

Serial.println("ON");

// Esperar 500 milisegundos sin usar delay()

unsigned long startMillis = millis();

while (millis() - startMillis < 500) {

// Espera activa sin bloqueo

}

// Apagar el LED utilizando el acceso directo a los registros del ESP32

GPIO.out\_w1tc = (1 << LED\_PIN); // Borrar el bit correspondiente al pin 2 (apagar LED)

// Enviar "OFF" por el puerto serie

Serial.println("OFF");

// Esperar 500 milisegundos sin usar delay()

startMillis = millis();

while (millis() - startMillis < 500) {

// Espera activa sin bloqueo

}

}

1.4.1

#include <Arduino.h>

#define LED\_PIN 2 // Usar el pin 2 (puedes cambiarlo a cualquier pin de salida)

void setup() {

// Inicializar el pin como salida

pinMode(LED\_PIN, OUTPUT);

// Iniciar la comunicación serie

Serial.begin(115200);

}

void loop() {

// Encender el LED

digitalWrite(LED\_PIN, HIGH);

// Enviar "ON" por el puerto serie

Serial.println("ON");

// Espera activa sin delay (utilizamos millis() para el control de tiempo)

while(millis() % 2 == 0); // Espera activa (sin delay)

// Apagar el LED

digitalWrite(LED\_PIN, LOW);

// Enviar "OFF" por el puerto serie

Serial.println("OFF");

// Espera activa sin delay

while(millis() % 2 == 1); // Espera activa (sin delay)

}

1.4.2

#include <Arduino.h>

#define LED\_PIN 2 // Usar el pin 2 (puedes cambiarlo a cualquier pin de salida)

void setup() {

// Configurar el pin como salida

pinMode(LED\_PIN, OUTPUT);

// Iniciar la comunicación serie

Serial.begin(115200);

}

void loop() {

// Encender el LED (accediendo directamente a los registros del ESP32)

GPIO.out\_w1ts = (1 << LED\_PIN); // Establecer el bit correspondiente al pin

// Enviar "ON" por el puerto serie

Serial.println("ON");

// Espera activa sin delay (utilizamos millis() para el control de tiempo)

while(millis() % 2 == 0); // Espera ativa (sin delay)

// Apagar el LED (accediendo directamente a los registros del ESP32)

GPIO.out\_w1tc = (1 << LED\_PIN); // Borrar el bit correspondiente al pin

// Enviar "OFF" por el puerto serie

Serial.println("OFF");

// Espera activa sin delay

while(millis() % 2 == 1); // Espera activa (sin delay)

}

4.3

#include <Arduino.h>

#define LED\_PIN 2 // Usar el pin 2 (puedes cambiarlo a otro pin disponible)

void setup() {

// Configurar el pin como salida (sin usar pinMode, pero lo dejamos para claridad)

pinMode(LED\_PIN, OUTPUT);

}

void loop() {

// Encender el LED (accediendo directamente a los registros del ESP32)

GPIO.out\_w1ts = (1 << LED\_PIN); // Establecer el bit correspondiente al pin

// Espera activa sin delay

while(millis() % 2 == 0); // Espera activa (sin delay)

// Apagar el LED (alternando el bit correspondiente al pin)

GPIO.out\_w1tc = (1 << LED\_PIN); // Borrar el bit correspondiente al pin

// Espera activa sin delay

while(millis() % 2 == 1); // Espera activa (sin delay)

}

4.4

#include <Arduino.h>

#define LED\_PIN 2 // Usar el pin 2 (puedes cambiarlo a otro pin disponible)

void setup() {

// Configurar el pin como salida

pinMode(LED\_PIN, OUTPUT);

}

void loop() {

// Encender el LED (accediendo directamente a los registros del ESP32)

GPIO.out\_w1ts = (1 << LED\_PIN); // Establecer el bit correspondiente al pin

// Espera activa sin delay

while(millis() % 2 == 0); // Espera activa (sin delay)

// Apagar el LED (alternando el bit correspondiente al pin)

GPIO.out\_w1tc = (1 << LED\_PIN); // Borrar el bit correspondiente al pin

// Espera activa sin delay

while(millis() % 2 == 1); // Espera activa (sin delay)

}

6

**Tiempo libre aproximado = 500 ms (tiempo de espera activa) / 500 ms (tiempo total de un ciclo) ≈ 100% de tiempo libre**.

El procesador está **casi completamente libre durante los 500 ms de espera activa** en cada ciclo. La cantidad de tiempo "ocupado" es mínima comparado con el tiempo de espera. Esto se debe a que el procesador está ejecutando solo el bucle de espera, sin hacer cálculos o tareas adicionales.

5

Diagrama de tiempos visual (esto se representará en el siguiente paso).

|----|----------------------|--------|----------------------|--------|-------------------|

Encender Enviar "ON" Espera Apagar LED Enviar "OFF" Espera

LED mensaje activa (rápido) mensaje activa

"ON" (rápido) (500ms)

Diagrama de flujo

Inicio

│

├── Configurar pin y comunicación serie

│

├── Encender LED

│

├── Enviar "ON"

│

├── Espera 500ms (sin delay)

│

├── Apagar LED

│

├── Enviar "OFF"

│

├── Espera 500ms (sin delay)

│

└── Repetir ciclo

adicional neopixel adafruit

#include <Arduino.h>

#include <Adafruit\_NeoPixel.h>

#define PIN 48 // Pin donde está conectado el LED NeoPixel

#define NUMPIXELS 1 // Solo un LED

Adafruit\_NeoPixel pixels(NUMPIXELS, PIN, NEO\_GRB + NEO\_KHZ800);

#define DELAYVAL 500 // Tiempo de espera (en milisegundos) entre cambios de color

void setup() {

pixels.begin(); // Inicializar el objeto NeoPixel (REQUIERIDO)

}

void loop() {

pixels.clear(); // Apagar todos los píxeles

// Establecer el color del primer (y único) LED a verde (puedes cambiar el valor RGB)

pixels.setPixelColor(0, pixels.Color(0, 150, 0)); // Verde brillante

pixels.show(); // Enviar los colores actualizados al hardware

delay(DELAYVAL); // Pausa antes de volver a cambiar el color

pixels.clear(); // Apagar todos los píxeles

// Establecer el color del primer LED a rojo

pixels.setPixelColor(0, pixels.Color(150, 0, 0)); // Rojo brillante

pixels.show(); // Enviar los colores actualizados al hardware

delay(DELAYVAL); // Pausa antes de volver a cambiar el color

pixels.clear(); // Apagar todos los píxeles

// Establecer el color del primer LED a azul

pixels.setPixelColor(0, pixels.Color(0, 0, 150)); // Azul brillante

pixels.show(); // Enviar los colores actualizados al hardware

delay(DELAYVAL); // Pausa antes de volver a cambiar el color

}