

# CODE FOR RECURSIVE MUTEX IN FreeRTOS

```
#include <Arduino.h>
#include "freertos/FreeRTOS.h"
#include "freertos/task.h"
#include "freertos/semphr.h"

// Define GPIO pins for LEDs
// Change to any free GPIO pin
#define LED1 18
#define LED2 19

SemaphoreHandle_t recursiveMutex; // Recursive Mutex handle
// Task for controlling LED1

void task1(void *parameter) {
    while (1) {
        // Acquire the recursive mutex
        if (xSemaphoreTakeRecursive(recursiveMutex, portMAX_DELAY)) {
            Serial.println("Task 1: LED1 ON");
            digitalWrite(LED1, HIGH); // Turn on LED1
            vTaskDelay(5000 / portTICK_PERIOD_MS); // Delay for 5 seconds
            digitalWrite(LED1, LOW); // Turn off LED1
            Serial.println("Task 1: LED1 OFF");
            // Release the recursive mutex
            xSemaphoreGiveRecursive(recursiveMutex);
        }
        vTaskDelay(100 / portTICK_PERIOD_MS); // Allow other tasks to run
    }
}

// Task for controlling LED2

void task2(void *parameter) {
    while (1) {
        // Acquire the recursive mutex
        if (xSemaphoreTakeRecursive(recursiveMutex, portMAX_DELAY)) {
            Serial.println("Task 2: LED2 ON");
            digitalWrite(LED2, HIGH); // Turn on LED2
            vTaskDelay(5000 / portTICK_PERIOD_MS); // Delay for 5 seconds
            digitalWrite(LED2, LOW); // Turn off LED2
            Serial.println("Task 2: LED2 OFF");
            // Release the recursive mutex
            xSemaphoreGiveRecursive(recursiveMutex);
        }
        vTaskDelay(100 / portTICK_PERIOD_MS); // Allow other tasks to run
    }
}

void setup() {
    // Initialize serial communication for debugging
    Serial.begin(115200);
    Serial.println("ESP32 FreeRTOS LED Blinking Starting...");

    // Set LED pins as outputs
    pinMode(LED1, OUTPUT);
    pinMode(LED2, OUTPUT);

    // Create a recursive mutex
    recursiveMutex = xSemaphoreCreateRecursiveMutex();
    if (recursiveMutex == NULL) {
        Serial.println("Failed to create recursive mutex!");
        while (1); // Stop execution if mutex creation fails
    }

    // Create two tasks for controlling the LEDs
    xTaskCreate(task1, "Task 1", 2048, NULL, 2, NULL);
    xTaskCreate(task2, "Task 2", 2048, NULL, 2, NULL);
}

void loop() {
    // Empty loop; tasks run independently
}
```

