IOT- FIRMWARE DEVELOPMENT

R MADESHWAR
(CEG-Anna university)

2.
Write an RTOS-based code using an esp32 microcontroller that has the following tasks
a. Read MPU6050 sensor data at 200 readings per second intervals and GPS sensor data at 1-second intervals from ESP32
b. Read temperature sensor data at every 5-second interval from the Arduino microcontroller.
c. Transfer the collected sensor data from Arduino to Esp32 via UART protocol
d. Publish the combined sensor data in JSON format above to the AWS MQTT or any other MQTT endpoint every 5 seconds.

```
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BME280.h>
#include <MPU6050.h>
#include <TinyGPS++.h>
#include <WiFi.h>
#include <PubSubClient.h>

// Constants for WiFi and MQTT
const char* ssid = "your-SSID";
const char* password = "your-PASSWORD";
const char* mqtt_server = "your-MQTT-server";

// MQTT topics
const char* mqtt_topic = "sensor_data";

// Objects for sensors
Adafruit_BME280 bme;
MPU6050 mpu;
TinyGPSPlus gps;

// Mutex for data sharing between tasks
SemaphoreHandle_t xMutex;

// Task handles
TaskHandle_t taskReadMPU6050, taskReadTemperature, taskTransferData,
taskPublishData;

void readMPU6050(void* parameter) {
  while (1) {
    // Read MPU6050 sensor data
    // Modify this part according to your actual MPU6050 library

    xSemaphoreTake(xMutex, portMAX_DELAY);
    // Process and store MPU6050 data
    xSemaphoreGive(xMutex);

    vTaskDelay(pdMS_TO_TICKS(5));  // Interval: 5 milliseconds
  }
}

void readTemperature(void* parameter) {
  while (1) {
    // Read temperature sensor data from Arduino
    // Modify this part according to your actual temperature sensor
library

    xSemaphoreTake(xMutex, portMAX_DELAY);
    // Process and store temperature data
    xSemaphoreGive(xMutex);

    vTaskDelay(pdMS_TO_TICKS(5000));  // Interval: 5 seconds
  }
}

void transferData(void* parameter) {
  while (1) {
    xSemaphoreTake(xMutex, portMAX_DELAY);
    // Transfer data from Arduino to ESP32 via UART

    // Transfer data from MPU6050 and temperature sensor to ESP32
    xSemaphoreGive(xMutex);

    vTaskDelay(pdMS_TO_TICKS(5));  // Interval: 5 milliseconds
  }
}

void publishData(void* parameter) {
  while (1) {
    xSemaphoreTake(xMutex, portMAX_DELAY);
    // Combine sensor data in JSON format
    // Example: String jsonData = createJsonData();
    // Publish data to MQTT
    // Example: mqttClient.publish(mqtt_topic, jsonData.c_str());

    xSemaphoreGive(xMutex);

    vTaskDelay(pdMS_TO_TICKS(5000));  // Interval: 5 seconds
  }
}

void setup() {
  Serial.begin(115200);

  // Initialize sensors

  // Create a mutex to protect shared data
  xMutex = xSemaphoreCreateMutex();

  // Connect to WiFi
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println("Connecting to WiFi...");
  }
  Serial.println("Connected to WiFi");

  // Connect to MQTT broker

  // Create tasks
  xTaskCreatePinnedToCore(readMPU6050, "TaskReadMPU6050", 10000, NULL,
1, &taskReadMPU6050, 0);
  xTaskCreatePinnedToCore(readTemperature, "TaskReadTemperature",
10000, NULL, 1, &taskReadTemperature, 0);
  xTaskCreatePinnedToCore(transferData, "TaskTransferData", 10000,
NULL, 1, &taskTransferData, 0);
  xTaskCreatePinnedToCore(publishData, "TaskPublishData", 10000, NULL,
1, &taskPublishData, 0);
}

void loop() {
  // Empty loop
}
```