SHARK TRACKING TAG PROTOTYPE WITH ESP32

The concept of this tracker revolves around an ESP32 microcontroller working together with sensors such as the MPU6050 accelerometer, gyroscope, and temperature sensor, and an SD module to store data while the shark is submerged. Once the shark emerges again, the collected data will be transmitted via the Swarm M138 satellite module, where the transmitted data will include date, time, gyroscope and accelerometer readings, and temperature.

C++ code for the tracker with ESP32:

```
#include <Wire.h>
#include <MPU6050.h>
#include <SPI.h>
#include <SD.h>
#include “time.h”

// ----------------- PIN CONFIGURATION -----------------
#define DAS_PIN 21
#define SCL_PIN 22
#define SD_CS 5

#define SWARM_RX 16
#define SWARM_TX 17
#define SWARM_BAUD 9600

// ----------------- VARIABLES -----------------
MPU6050 mpu;
Unsigned long lastSample = 0;
Const unsigned long SAMPLE_INTERVAL_MS = 1000; // 1 second between samples
Bool submerso = true;

// ----------------- HELPER FUNCTIONS -----------------
Bool initSD() {
    If (!SD.begin(SD_CS)) {
        Serial.println(“Erro ao iniciar SD card!”);
        Return false;
    }
    Serial.println(“SD card iniciado.”);
    If (!SD.exists(“/data_log.csv”)) {
        File f = SD.open(“/data_log.csv”, FILE_WRITE);
        If (f) {
            f.println(“data,hora,accX,accY,accZ,gyroX,gyroY,gyroZ,tempC”);
            f.close();
        } else {
            Serial.println(“Erro criando arquivo CSV.”);
            Return false;
        }
    }
```

```
    Return true;
}

Void logToSD(String line) {
    File f = SD.open("/data_log.csv", FILE_APPEND);
    If (f) {
        f.println(line);
        f.close();
        Serial.println("LOG OK: " + line);
    } else {
        Serial.println("Erro gravando no SD!");
    }
}

Bool isSatelliteAvailable() {
    // Simulated function: returns true if there is communication with the
Swarm M138
    // In practice, you can check for an "OK" response from the module
    Serial2.println("AT"); // AT command for testing
    Delay(500);
    If (Serial2.available()) {
        String resp = Serial2.readString();
        If (resp.indexOf("OK") >= 0) return true;
    }
    Return false;
}

Void sendDataSwarm() {
    Serial.println("Enviando dados via Swarm M138...");
    File f = SD.open("/data_log.csv");
    If (!f) {
        Serial.println("Arquivo não encontrado!");
        Return;
    }

    While (f.available()) {
        String line = f.readStringUntil('\n');
        Serial2.println(line); // send to Swarm module
        Delay(200); // wait for module to process each line
    }
    f.close();
    Serial.println("Dados enviados com sucesso.");

    SD.remove("/data_log.csv");
    File nf = SD.open("/data_log.csv", FILE_WRITE);
    If (nf) {
        Nf.println("data,hora,accX,accY,accZ,gyroX,gyroY,gyroZ,tempC");
        Nf.close();
    }
}
```

```
// ----------------- SETUP -----------------
Void setup() {
    Serial.begin(115200);
    Serial2.begin(SWARM_BAUD, SERIAL_8N1, SWARM_RX, SWARM_TX);
    Wire.begin(DAS_PIN, SCL_PIN);

    // Initialize MPU6050
    Mpu.initialize();
    If (!mpu.testConnection()) {
        Serial.println("Falha na conexao com MPU6050!");
        While (1) { delay(100); }
    }
    Serial.println("MPU6050 conectado.");

    // Initialize SD
    If (!initSD()) {
        Serial.println("Falha ao iniciar SD. Parando execução.");
        While(1) { delay(1000); }
    }

    // Initialize ESP32 time
    configTime(0, 0,"pool.ntp.org"); // UTC 0 timezone
}

// ----------------- LOOP -----------------
Void loop() {
    Unsigned long now = millis();
    If (now - lastSample < SAMPLE_INTERVAL_MS) return;
    lastSample = now;

    // -------- Read MPU6050 --------
    Int16_t ax, ay, az, gx, gy, gz;
    Mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
    Float accX = ax / 16384.0;
    Float accY = ay / 16384.0;
    Float accZ = az / 16384.0;
    Float gyroX = gx / 131.0;
    Float gyroY = gy / 131.0;
    Float gyroZ = gz / 131.0;
    Float tempC = mpu.getTemperature() / 340.0 + 36.53;

    // -------- Date and time --------
    Struct tm timeinfo;
    If (!getLocalTime(&timeinfo)) {
        Serial.println("Falha ao obter hora");
        Return;
    }
    Char dateBuffer[11];
    Char timeBuffer[9];
    Strftime(dateBuffer, 11,"%Y-%m-%d", &timeinfo);
    Strftime(timeBuffer, 9,"%H:%M:%S", &timeinfo);
```

```
    String dataLine = String(dateBuffer) + ”,” + String(timeBuffer) + ”,” +
    String(accX,4) + ”,” + String(accY,4) + ”,” + String(accZ,4) + ”,”+
    String(gyroX,3) + ”,” + String(gyroY,3) + ”,”+ String(gyroZ,3) +”,”+
    String(tempC,2);

    If (submerso) {
        logToSD(dataLine);
        // Check if Swarm is available (surface)
        If (isSatelliteAvailable()) {
            Submerso = false; // emerged
        }
    } else {
        sendDataSwarm();
        submerso = true; // reset for next collection
    }
}
```

One problem with this type of tracking is that the shark becomes “invisible” when submerged. However, this issue can be circumvented by storing the data and transmitting it as soon as the shark is close enough to the surface to establish satellite contact.

Connections:

MPU6050 (Accelerometer, Gyroscope, and Temperature) MPU6050 Module: - VCC 5V - GND GND (shared with ESP32) - DAS GPIO21 (DAS) - SCL GPIO22 (SCL) - AD0 GND or 3.3V - INT Optional, can be used for interrupts

SD Card Module - VCC 3.3V - GND GND (shared with ESP32) - CS GPIO5 Chip Select (can be changed in code) - MOSI GPIO23 - MISO GPIO19 - SCK GPIO18 Clock SPI

Swarm M138 Module - VCC 3.3V - GND GND (shared with ESP32) - TX RX of ESP32 (GPIO16 / Serial2 RX) - RX TX of ESP32 (GPIO17 / Serial2 TX) - RESET - PWR_EN Optional 3.3V or GPIO to turn module on/off