

Documentation for sensorModelo06 Project

Overview

The sensorModelo06 project is an Arduino-based system designed for environmental monitoring and data collection. It integrates multiple sensors, GPS, GSM/LTE communication, and AES encryption to collect, process, and securely transmit data. The system is designed to operate efficiently with low power consumption, utilizing deep sleep modes to conserve energy.

Key Features

1. **Sensor Integration:**
 - Collects data from various environmental sensors (e.g., temperature, humidity, light, soil properties).
 2. **GPS Module:**
 - Retrieves and stores latitude and longitude for geolocation.
 3. **GSM/LTE Communication:**
 - Sends collected data to a remote server using TCP.
 4. **AES Encryption:**
 - Encrypts data before transmission for secure communication.
 5. **Low Power Mode:**
 - Utilizes deep sleep to conserve energy during idle periods.
-

File Structure

1. [sensorModelo06.ino](#):
 - Main entry point for the Arduino program.
 - Initializes components and manages the main loop.
2. [gpssen.cpp / gpssen.h](#):
 - Handles GPS data collection, parsing, and storage.
3. [sensores.cpp / sensores.h](#):
 - Manages sensor initialization, data collection, and debugging.
4. [gsmlte.cpp / gsmlte.h](#):

- Manages GSM/LTE communication, including TCP connections and modem control.
 - 5. [aesenc.cpp](#) / [aesenc.h](#):
 - Provides AES encryption and decryption for secure data transmission.
 - 6. [timeData.cpp](#) / [timeData.h](#):
 - Handles time-related operations, including RTC synchronization and epoch time management.
 - 7. [sleepSensor.cpp](#) / [sleepSensor.h](#):
 - Manages deep sleep mode and wake-up handling.
 - 8. [type_def.h](#):
 - Defines the [sensordata_type](#) structure for storing sensor data.
-

Global Constants

- [SEND_DATA_INTERVAL](#):
 - Interval (in milliseconds) for sending data.
 - [MAX_RETRIES](#):
 - Maximum number of retries for sending data before entering sleep mode.
-

Global Variables

- [sensordata](#):
 - Stores sensor data in a structured format.
 - [uuid](#):
 - Unique identifier for the device.
 - [epoch](#):
 - Epoch time as a string.
 - [latitud](#) / [longitud](#):
 - Latitude and longitude as strings.
 - [rssí](#):
 - Signal strength (e.g., "-70dBm").
-

Main Components

1. Sensors

- **Initialization:**
 - Sensors are initialized in [setupSensors\(\)](#) with retries for robustness.
- **Data Collection:**
 - Sensor data is read and stored in the [sensordata](#) structure using [readSensorData\(\)](#).
- **Debugging:**
 - Sensor data can be printed to the serial monitor using [debugData\(\)](#).

2. GPS

- **Initialization:**
 - GPS module is initialized in [setupGps\(\)](#).
- **Data Handling:**
 - GPS data is read, validated, and stored using [readGps\(\)](#) and [parseGNRMC\(\)](#).

3. GSM/LTE Communication

- **Initialization:**
 - GSM/LTE module is initialized in [inicioGsmLte\(\)](#).
- **Data Transmission:**
 - Data is sent to a remote server using [tcpOpen\(\)](#), [tcpSendData\(\)](#), and [tcpClose\(\)](#).

4. AES Encryption

- **Encryption:**
 - Data is encrypted using [encrypt\(\)](#) before transmission.
- **Decryption:**
 - Encrypted data can be decrypted using [decrypt\(\)](#).

5. Time Management

- **RTC Synchronization:**
 - Internal RTC is synchronized with an external RTC using [setFechaRtc\(\)](#).
- **Epoch Time:**
 - Epoch time is updated and stored using [epoch\(\)](#).

6. Sleep Mode

- **Deep Sleep:**
 - The system enters deep sleep mode using `sleepIOT()` to conserve energy.
 - **Wake-Up Handling:**
 - Wake-up reasons are printed and handled using `printWakeupReason()`.
-

Workflow

Setup Phase

1. Initialize serial communication.
2. Print wake-up reason.
3. Initialize time, sensors, GPS, and GSM/LTE modules.

Main Loop

1. Check if the data send interval has elapsed.
 2. Read sensor data and debug it.
 3. Encrypt the sensor data.
 4. Send the encrypted data to the server.
 5. Enter sleep mode after successful data transmission or retries.
-

Key Functions

`agro.ino`

- `setup():`
 - Initializes all components.
- `loop():`
 - Manages the main workflow, including data collection, encryption, and transmission.
- `check_interval():`
 - Checks if the data send interval has elapsed.
- `convertStructToString():`
 - Converts the `sensordata` structure to a concatenated string.
- `envioData():`

- Sends data over GSM/LTE with retries.

[gpsen.cpp](#)

- [setupGps\(\)](#):
 - Initializes the GPS module.
- [runGpsSen\(\)](#):
 - FreeRTOS task for continuously reading GPS data.
- [readGps\(\)](#):
 - Reads GPS data from the serial port.
- [parseGNRMC\(\)](#):
 - Parses GPS data to extract latitude and longitude.

[sensores.cpp](#)

- [setupSensores\(\)](#):
 - Initializes sensors with retries.
- [readSensorData\(\)](#):
 - Reads data from all connected sensors.
- [debugData\(\)](#):
 - Prints sensor data for debugging.

[gsmlte.cpp](#)

- [inicioGsmLte\(\)](#):
 - Initializes the GSM/LTE module.
- [startLTE\(\)](#):
 - Starts the LTE connection.
- [tcpSendData\(\)](#):
 - Sends data over TCP with retries.

[aesenc.cpp](#)

- [encrypt\(\)](#):
 - Encrypts plaintext using AES and encodes it in Base64.
- [decrypt\(\)](#):
 - Decrypts Base64-encoded AES-encrypted data.

timeData.cpp

- [setupTimeData\(\)](#):
 - Initializes time-related data.
- [setFechaRtc\(\)](#):
 - Synchronizes the internal RTC with the external RTC.
- [epoch\(\)](#):
 - Updates the epoch time.

sleepSensor.cpp

- [sleepIOT\(\)](#):
 - Puts the system into deep sleep mode.
- [printWakeupReason\(\)](#):
 - Prints the reason for waking up from sleep.

Example Output

- Serial Monitor

```
ESP-ROM:esp32s3-20210327
Build:Mar 27 2021
rst:0xc (RTC_SW_CPU_RST),boot:0xb (SPI_FAST_FLASH_BOOT)
Saved PC:0x4037a22a
SPIWP:0xee
mode:DIO, clock div:1
load:0x3fce2820,len:0x118c
load:0x403c8700,len:0x4
load:0x403c8704,len:0xc20
load:0x403cb700,len:0x30e0
entry 0x403c88b8
Wakeup was not causeESP-ROM:esp32s3-20210327
Build:Mar 27 2021
rst:0x1 (POWERON),boot:0xb (SPI_FAST_FLASH_BOOT)
SPIWP:0xee
mode:DIO, clock div:1
load:0x3fce2820,len:0x118c
load:0x403c8700,len:0x4
load:0x403c8704,len:0xc20
load:0x403cb700,len:0x30e0
entry 0x403c88b8
Wakeup was not caused by deep sleep: 0
1747403304
Task GPS running on core 0
20.6447449
0.0000000
----- INIT LTE -----
Modem: R1951.07
Connected to network.
----Debug---
uuid: 89883030000096467599
epoc: 1747403304
temperatura_ambiente: 31.74
humedad_relativa : 22.61
luz: 12.50
radiación: 0.00
presión_barometrica: 0.00
co2: 0.00
humedad_suelo: 3.82
temperatura_suelo: 27.89
conductividad_suelo: 0.01
ph_suelo: 0.00
nitrógeno: 0.00
fósforo: 10.10
potasio: 0.00
viento: 0.00
latitud: 20.6447449
logitud: 0.0000000
battery voltage: 3.55
rssi: 10
-----
String constructed to send:
$,$988303000096467599,1747403304,31.74,22.61,12.50,0.00,0.00,0.00,3.82,27.89,0.01,0.00,0.00,10.10,0.00,0.00,20.6447449,0.0000000,3.55,10,#
Data sent successfully on attempt 1

Setup ESP32 to sleep for every 600 seconds
Going to sleep now
```

Key Notes

- **Error Handling:**
 - Retries are implemented for sensor initialization and data transmission.
 - The system enters sleep mode after maximum retries.
- **Low Power Design:**
 - Deep sleep mode is used to conserve energy during idle periods.
- **Modular Design:**
 - Functions are modularized for better readability and maintainability.

Server side data store

```
{ "tr_dispositivos_modelo_id": "422", "uuid": "89883030000096467599", "epoch": "1747402095", "humedad_relativa": "21.25", "temperatura_ambiente": "31.57", "temperatura_suelo": "27.87", "humedad_suelo": "27.89", "ce_suelo": "0.01", "nitrogeno": "0", "fosforo": "10.1", "potasio": "0", "presion_barometrica": "0", "luz": "12.5", "co2": "0", "viento": "0", "ph_suelo": "0.00", "radiacion": "0", "bat": "3.55", "lat": "20.644745", "lng": "0.000000", "rss": "10", "time_stamp": "2025-05-16 19:51:23"}, { "tr_dispositivos_modelo_id": "423", "uuid": "89883030000096467599", "epoch": "1747402700", "humedad_relativa": "22.19", "temperatura_ambiente": "31.62", "temperatura_suelo": "27.87", "humedad_suelo": "27.89", "ce_suelo": "0.01", "nitrogeno": "0", "fosforo": "10.1", "potasio": "0", "presion_barometrica": "0", "luz": "12.5", "co2": "0", "viento": "0", "ph_suelo": "0.00", "radiacion": "0", "bat": "3.55", "lat": "20.644745", "lng": "0.000000", "rss": "10", "time_stamp": "2025-05-16 19:51:23"}, { "tr_dispositivos_modelo_id": "424", "uuid": "89883030000096467599", "epoch": "1747403304", "humedad_relativa": "22.61", "temperatura_ambiente": "31.74", "temperatura_suelo": "27.89", "humedad_suelo": "27.89", "ce_suelo": "0.01", "nitrogeno": "0", "fosforo": "10.1", "potasio": "0", "presion_barometrica": "0", "luz": "12.5", "co2": "0", "viento": "0", "ph_suelo": "0.00", "radiacion": "0", "bat": "3.55", "lat": "20.644745", "lng": "0.000000", "rss": "10", "time_stamp": "2025-05-16 19:51:23"}, { "tr_dispositivos_modelo_id": "425", "uuid": "89883030000096467599", "epoch": "1747403908", "humedad_relativa": "22.9", "temperatura_ambiente": "31.85", "temperatura_suelo": "27.94", "humedad_suelo": "27.96", "ce_suelo": "0.01", "nitrogeno": "0", "fosforo": "10.1", "potasio": "0", "presion_barometrica": "0", "luz": "12.5", "co2": "0", "viento": "0", "ph_suelo": "0.00", "radiacion": "0", "bat": "3.55", "lat": "20.644745", "lng": "0.000000", "rss": "10", "time_stamp": "2025-05-16 19:51:23"}, { "tr_dispositivos_modelo_id": "426", "uuid": "89883030000096467599", "epoch": "1747404513", "humedad_relativa": "23.01", "temperatura_ambiente": "31.95", "temperatura_suelo": "27.96", "humedad_suelo": "27.96", "ce_suelo": "0.01", "nitrogeno": "0", "fosforo": "10.1", "potasio": "0", "presion_barometrica": "0", "luz": "12.5", "co2": "0", "viento": "0", "ph_suelo": "0.00", "radiacion": "0", "bat": "3.55", "lat": "20.644745", "lng": "0.000000", "rss": "10", "time_stamp": "2025-05-16 19:51:23"}]
```

Comparative power measures, optimized firmware vs. original firmware

Memory

```
ago.ino aesenc.cpp aesench gpssen.cpp gpssen.h gsmlite.cpp gsmlite.h sensors.cpp SoftwareSerial.h AHT20.h sensors.h sleep
35
36 String uuid = ""; // UUID (36 characters + null terminator)
37 String epoch = ""; // Epoch time as a string
38 String latitud = ""; // Latitude as a string
39 String longitud = ""; // Longitude as a string
40 String rssi = ""; // Signal strength (e.g., "-70dBm")
41
42 // Setup function, runs once at startup
43 void setup() {
44   Serial.begin(115200); // Initialize serial communication at 115200 baud
45   printWakeUpReason(); // Print the reason for waking up from sleep
46   setTimeData(); // Initialize time-related data
47   setupSensors(); // Initialize sensors
48   setupGps(); // Initialize GPS module
49   inicioGsmte(); // Initialize GSM/LTE communication
50   // TODO: Add wait for GPS setup
51 }
52
53 // Main loop, runs repeatedly
54 void loop() {
55   static bool send_data_interval_flag = true; // Flag to check if data should be sent
56   String data_encrypted = ""; // Variable to store encrypted data
57
58   if (send_data_interval_flag) { // If it's time to send data
59     readSensorData(&sensordata); // Read sensor data into the 'sensordata' structure
60     debugData(&sensordata); // Print sensor data for debugging
61     data_encrypted = encrypt(convertStructToString(sensordata)); // Encrypt the sensor data
62     //Serial.println("Encrypted data:");
63     //Print a label for the encrypted data
64     //Serial.println(data_encrypted); // Print the encrypted data
65   }
66 }
67
68 Output Serial Monitor
Sketch uses 459074 bytes (35%) of program storage space. Maximum is 1310720 bytes.
Global variables use 24168 bytes (7%) of dynamic memory, leaving 303520 bytes for local variables. Maximum is 327680 bytes.
espTool.py v4.8.1
Serial port COM3
Connecting...
Chip is ESP32-S3 (QFN56) (revision v0.2)
Features: WiFi, BLE
Crystal is 40MHz
MAC: e4:b3:23:e9:db:18
```

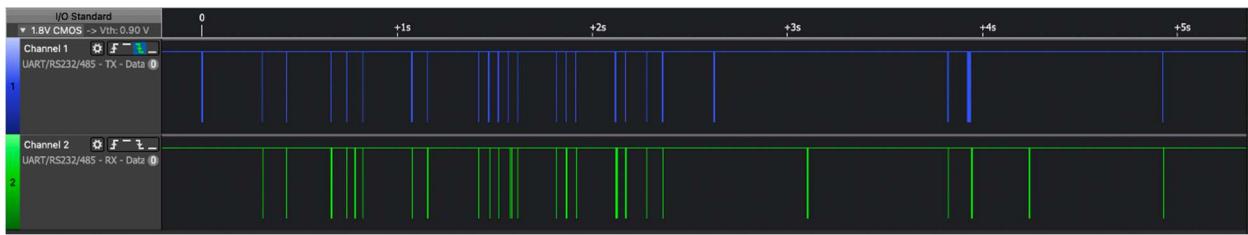
Optimized firmware.

```
sensorModel06.ino aesenc.cpp aesench dataSave.cpp dataSave.h debugStr.cpp debugStr.h funaux.cpp funaux.h gpssen.cpp gpssen.h gsm
114     |     modemPowerOff();
115     |     sleepIOT();
116     |
117     |     delay(1000);
118 }
119 modem.sendAT("+CPSI?");
120 if (modem.waitForResponse(1000L, res) == 1) {
121 // res.replace(GSM_NL "OK" GSM_NL, "");
122 Serial.printin(res);
123 }
124
125
126 res = "";
127
128 //modem.sendAT("+CGDCONT=1,\"IP\",\"eapn1.net\"");
129 modem.sendAT("+CGDCONT=1,\"IP\",\"em\"");
130 if (modem.waitForResponse(1000L, res) == 1) {
131 }
132
133 modem.sendAT("+CINACT=0,1");
134 if (modem.waitForResponse(1000L, res) == 1) {
135 }
136
137 for (int i = 0; i <= 5; i++) {
138     iccidSIM0 = modem.getSimCID();
139     imeiGSM0 = modem.getIMEI();
140 }

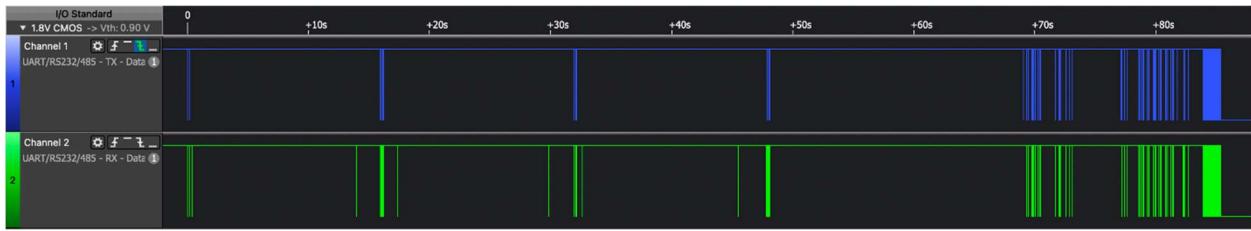
Output
Sketch uses 471106 bytes (35%) of program storage space. Maximum is 1310720 bytes.
Global variables use 24432 bytes (7%) of dynamic memory, leaving 303248 bytes for local variables. Maximum is 327680 bytes.
esptool.py v4.8.1
Serial port COM3
Connecting...
Chip is ESP32-S3 (QFN56) (revision v0.2)
Features: WiFi, BLE
Crystal is 40MHz
MAC: e4:b3:23:e0:db:18
Uploading stub...
Running stub...
Stub running...
Changing baud rate to 921600
```

Original firmware.

Modem time used from boot to end send message.

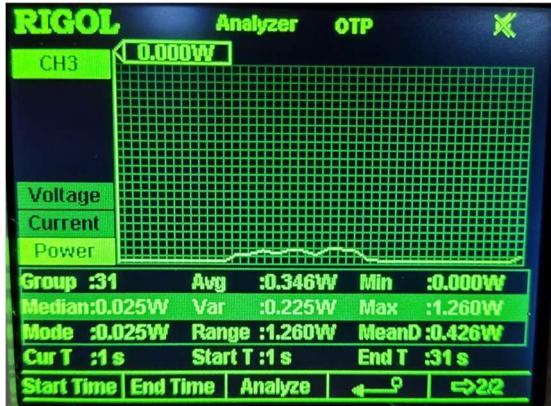


Optimized firmware.



Original firmware.

Power battery drain



Optimized firmware.



Original firmware.