**­Tracer System Research**

**Short description: -**

This IoT based Tracer System will control the Electric Vehicle, visualize the battery information and recognize the real time map location of a Vehicle.

**Objective: -**

* To design an embedded / IOT product for real time tracking

**Components: -**

Esp32

Stm8

UBLOX NEO-6M V2 GPS MODULE

LoRa Wan module

**Abstraction: -**

I'm going to design and develop my own products "Tracer device" using esp32 microcontroller and UBLOX NEO-6M V2 GPS MODULE for electrical vehicle, what should I used for communication between GPS point and location point? And can I find a map location from GPS measurements?

For communication between the GPS module (UBLOX NEO-6M V2) and our ESP32 microcontroller, we can use UART (Universal Asynchronous Receiver-Transmitter) communication protocol. The UBLOX NEO-6M V2 GPS module typically communicates over UART, so we'll connect the TX (transmit) pin of the GPS module to the RX (receive) pin of the ESP32, and vice versa. Then we can use UART communication functions in our ESP32 firmware to send commands to the GPS module and receive GPS data.

As for finding a map location from GPS measurements, yes, it is entirely possible. Once we have the latitude and longitude coordinates from our GPS module, we can use various mapping APIs or services like Google Maps API, OpenStreetMap, or Mapbox to convert those coordinates into a readable location on a map. These services typically offer APIs that allow we to send GPS coordinates and receive back information like street addresses, landmarks, or display the location on a map interface. We'll need an internet connection on our ESP32 to make requests to these APIs and receive the location data.

**this idea will need a WIFI internet connections, is any way to establish this working without internet?**

If we need to establish communication between our ESP32-based Tracer device and a user interface or server without relying on an internet connection, we can consider using alternative communication methods such as Bluetooth or LoRa (Long Range). Here's how we can approach each option:

**Bluetooth: -**

ESP32 modules often come with built-in Bluetooth capabilities. We can use Bluetooth Low Energy (BLE) to establish a communication link between our Tracer device and a smartphone or another device equipped with Bluetooth. This way, we can transmit GPS data or other information directly to nearby devices without needing an internet connection.

**LoRa: -**

LoRa (Long Range) is a wireless communication technology that enables long-range communication between devices without the need for a Wi-Fi or cellular network. We can use LoRa modules with the ESP32 to transmit GPS data over long distances to a LoRa gateway or receiver. This is especially useful for applications where devices are located in remote areas or where internet connectivity is not available.

By using Bluetooth or LoRa, we can create a local communication network for we Tracer device without relying on an internet connection. However, keep in mind that the range and data throughput of these technologies may vary depending on factors such as environment and interference.

**Networks connection options: -**

**Local Wi-Fi Network: -**

We can set up a local Wi-Fi network in the area where our Tracer device will be used. This could be achieved using a portable Wi-Fi hotspot or a dedicated local network setup. The ESP32 can then connect to this local Wi-Fi network to enable communication with a server or user interface without requiring an internet connection. We could host a local server or interface on a device within the network, such as a smartphone or a Raspberry Pi, to display the location on a map.

**Ad-Hoc Wi-Fi Network: -**

Another option is to create an ad-hoc (peer-to-peer) Wi-Fi network between the Tracer device and another device, such as a smartphone or a laptop. This allows direct communication between the two devices without the need for a centralized Wi-Fi access point. We can then develop a mobile application or software on the other device to receive GPS data from the Tracer device and display it on a map interface.

**Offline Map Data: -**

To view the location on a map without an internet connection, we can pre-load offline map data onto the device that will display the map. There are libraries and tools available that allow we to store map data locally on a device, such as Mapbox's offline maps feature. By pre-loading map data covering the area where the Tracer device will be used, we can display the location on a map interface even without internet connectivity.

Each of these options has its advantages and considerations depending on factors such as the intended use case, available resources, and desired level of functionality.

Note: -

**Local Wi-Fi Network and Ad-Hoc Wi-Fi Network will have the limited communication range, there are any range limit of Offline Map Data?**

The range limit of Offline Map Data depends on how much data we preload onto our device. Essentially, there's no inherent range limit since the map data is stored locally on the device. However, the extent of the map area we can cover with offline data will be limited by the storage capacity of the device and the size of the map data.

Here are some factors to consider regarding the range of Offline Map Data:

**Storage Capacity:** -

The amount of map data we can store on our device will be limited by its storage capacity. Higher-capacity devices can store more map data, allowing we to cover a larger area.

**Map Resolution and Detail: -**

Higher-resolution maps with more detail will require more storage space. If we need detailed maps for a large area, we may need to prioritize which areas to include in our offline map data based on their importance or relevance to our application.

**Map Data Compression: -**

Some mapping libraries and tools offer data compression techniques to reduce the size of offline map data while maintaining reasonable detail. Utilizing compression techniques can help we maximize the coverage area within the storage constraints of our device.

**Dynamic Updates: -**

Keep in mind that offline map data may become outdated over time. We may need to periodically update the offline maps to ensure they reflect any changes or updates to the geographic area.

In summary, while there's no inherent range limit to Offline Map Data, the practical range will be determined by factors such as device storage capacity, map resolution, and the level of detail required. By managing these factors effectively, we can optimize the coverage area of our offline maps to suit our application's needs.

**Mapbox: -**

Map box is a mapping platform that provides tools and APIs for developers to incorporate maps, geolocation, and navigation into their applications. It allows you to display maps, add custom markers, draw routes, and perform various other spatial operations. Mapbox typically relies on internet connectivity to fetch map tiles and other data, but it also supports offline map functionality where map data is pre-loaded onto the device.

**LoRa WAN: -**

LoRaWAN (Long Range Wide Area Network) is a wireless communication protocol designed for long-range, low-power communication between IoT devices and gateways. It enables devices to transmit small amounts of data over long distances, making it suitable for applications like remote monitoring, asset tracking, and smart city deployments. LoRaWAN operates in the unlicensed radio spectrum, providing coverage over several kilometers in urban environments and even farther in rural areas.

While Mapbox provides mapping functionality and visualization tools, LoRaWAN enables communication between IoT devices and back-end systems over long distances. In some applications, you might use both technologies together. For example:

You could use LoRaWAN to transmit GPS coordinates from IoT devices (such as your Tracer device) to a central server.

The central server could then use Mapbox APIs to visualize the GPS data on a map interface, allowing users to track the location of the devices in real-time.

So, rather than replacing LoRaWAN, Mapbox can complement it by providing a means to visualize and interact with the location data collected through LoRaWAN communication.

**Methodologies: -**

Choosing ESP-IDF platform-io in VSCODE for development and implementation.

**Monitoring unit: -**

A unit which is installed (may be movable) in a remote area to monitor with the real time location on map and control the vehicle from non-authentication path ways.

Wi-Fi is configured in security mode (WPA3, PMF) to fetch data from server and deliver to the web app or mobile app.

**Moving unit: -**

A unit which is installed in a Vehicle to receive sensor encrypted data and send to monitoring unit for further process

**Map JavaScript APIs overview: -**

[https://developers.google.com/maps/documentation/javascript/overview?hl=en&\_gl=1\*s94cyo\*\_ga\*MTE4NjA2ODIwLjE3MTM4NzEzNzY.\*\_ga\_NRWSTWS78N\*MTcxMzk1NjEzNi41LjEuMTcxMzk1NjY5Ni4wLjAuMA](https://developers.google.com/maps/documentation/javascript/overview?hl=en&_gl=1*s94cyo*_ga*MTE4NjA2ODIwLjE3MTM4NzEzNzY.*_ga_NRWSTWS78N*MTcxMzk1NjEzNi41LjEuMTcxMzk1NjY5Ni4wLjAuMA)

**Integrating method:**

<https://www.youtube.com/watch?v=Zxf1mnP5zcw>

**Implementation: -**