VITA – Vehicle Information and Telemetry Awesomeness

The Feasibility and Design of an Affordable IoT Vehicular Module

Lan Tran, Jakov Sola, Mak Midzic, Ivana Tihi-Babic

Problem

The world is moving fast towards the age of the Internet of Things, and rightly so. But some devices such as cars are not something that people buy too often, especially not in low income regions. A lack of funds is however not a reason for people to lag behind in the ongoing IoT revolution. Old cars should also be able to participate in the modern Smart City infrastructure that is being built up.

Apart from the fact that they would miss out on receiving live traffic information, they will also not be able to participate in peer-to-peer communication that autonomous cars of the future will rely on. For example a self-driving vehicle might find it useful to know the exact position and velocity of vehicles, and instead of using sensors, it is much easier to get the information directly from the other car.

Solution

We propose a compact and cheap product that would have the ability to communicate with:

- The infrastructure
- Other cars participating in traffic
- The owner's smartphone

Infrastructure communication (V2I)

The device would send the following data to the infrastructure:

- Car identifier (unique but anonymous code)
- Temperature and humidity (can be used for weather forecasting)
- Location and speed (can be used to detect traffic flow and available parking)
- Distance to car in front (can be used for detecting traffic jams)

It would receive the following:

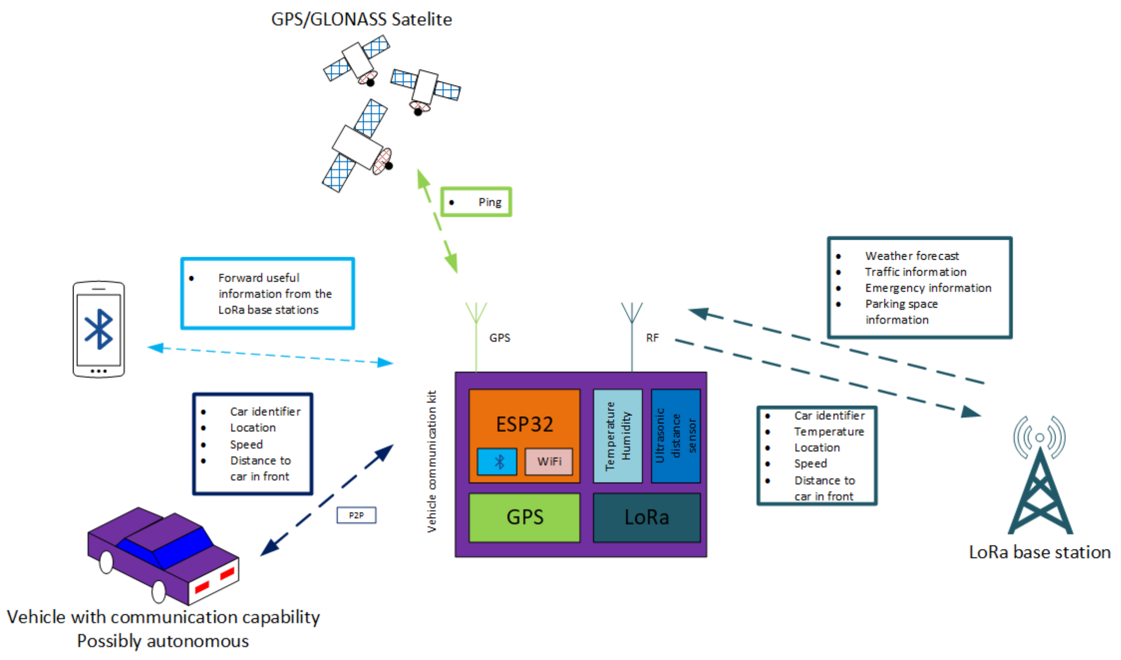
- Weather forecast
- Traffic information
- Emergency information
- Parking space information

Vehicle to vehicle communication (V2V)

The device will just broadcast the following information about itself:

- Car identifier
- Location
- Speed
- Distance to car in front

Hardware guidelines



Communication

To implement the V2I requirement, we have decided to use a **LoRa** transceiver. More and more cities are building LoRaWAN infrastructure and it seems that this is the direction in which the IoT will move. A drawback of LoRa is that it has a relatively low speed of data transfer. However, it uses very little power and can transmit over very long distances which makes it perfect for this application.

For this we have decided to use a simple **WiFi** module to broadcast the necessary information. WiFi is relatively short range but a very fast protocol which is perfect for broadcasting live information to surrounding vehicles.

Finally, to communicate with the owner’s smartphone, a **Bluetooth** module will be used.

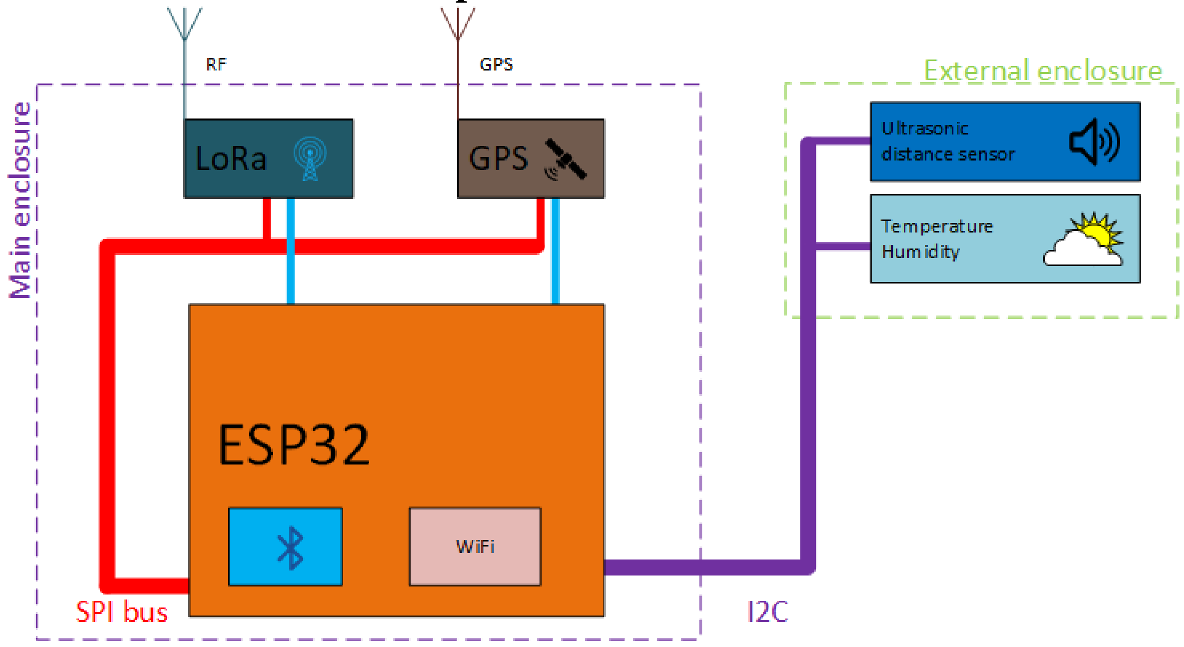
Sensors

The VITA device will have to be able to effectively measure its exact location and its speed. However, as the device is not intended to be integrated with the car itself, it cannot use the built in speedometer. This is why we have chosen to use **GNSS** to measure both the location and speed (change in location over time).

The **temperature and humidity** can be measured by a single IC that are also widely available.

Finally, the distance to the car in front can be measured using a single **ultrasonic sensor** that is mounded at the front of the car. Since the purpose of this sensor is to detect and relay information about traffic jams (when cars are very close to each other), the range of the sensor can be only a few metres and the precision does not need to be too high.

Hardware selection and operation



Bill of materials (main components)

Part	Price [HUF]	Description	Digi-Key
MAX-M8C-0	3230.17	The MAX-M8 series of GNSS modules offers a high precision positioning using GPS, GLONAS, QZSS and SBAS. It is an economical but quality option for out VITA product. It offers communication to our MCU via UART, **SPI** and I2C. Positioning accuracy is 2m which is very good for the price.	Link
RFM95W-868S2	4519.72	A cheap and compact LoRa transciever modem. It operates at a 868MHz frequency which is the EU standard for LoRa. It uses minimal power (only 0.2uA in sleep mode and up to 20 mA while receiving or transmiting)	Link
SHTC3-TR-10KS	366.33	Temperatrue and humidity sensor IC with I2C digital interface	Link
MCP3021A5T-E/OT	378.73	10-bit I2C ADC	Link
Analogue distance measurement		A short range (up to 4m) ultrasonic distance sensor. It would convey the distance from an object in front by setting its output voltage to be a percentage of the input voltage corresponding to the distance/maximum range	
ESP32-WROOM-32D	939	Low cost MCU model with built in WiFi and BT functionality	Link
NE555DRG4	33.41	Precision timing circuit capable of astable operation to generate a square wave of specified frequency.	Link

The components within the main enclosure would communicate using **SPI**. The reason is that GPS and LoRa modules that communicate in that way are very cheap and easy to come by. This is an important factor as one of the main appeals of our device would be its low price. One of the drawbacks of SPI is that it uses much more connections than I2C or CAN, but this is not a problem in this case as all of the components are very close to each other and would probably find themselves on the same PCB.

The sensors that must be placed on the exterior of the car must have a connection to the main module. SPI, although a good protocol, would require a total of 5 wires (SCK, MOSI, MISO, 2XCS) just to communicate with two devices. Ideally a CAN bus would be used for this as it uses only two wires.

Furthermore, they are a differential pair which means they are very robust to changes in the EMF (that is why they are used in the automotive industry). However, it is much easier to find cheap components using I2C (which also uses two wires) and that is why we decided that the main module and the exterior module should be connected via an **I2C** bus.

Schematics

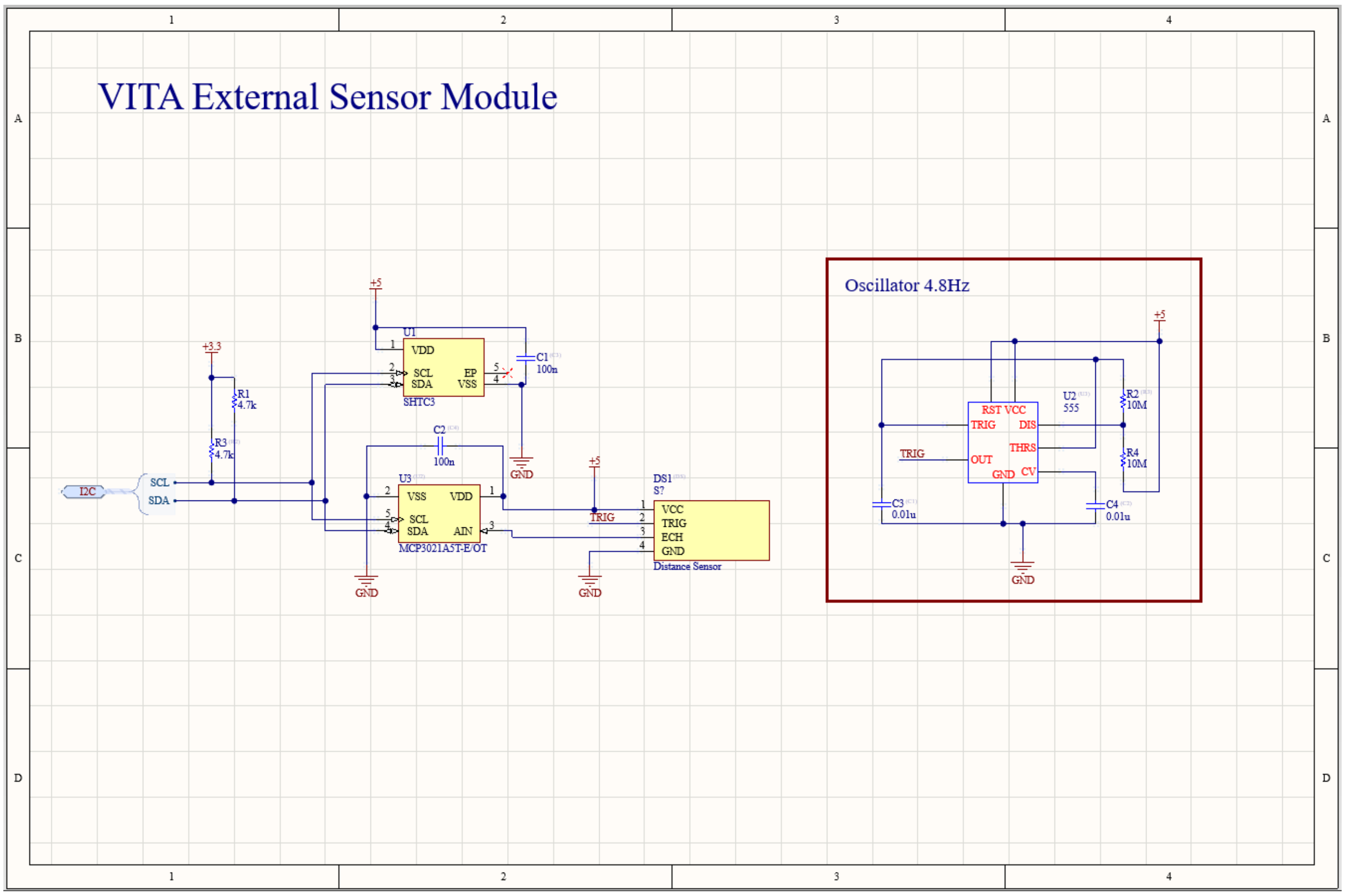
Schematics are only approximate as there is not enough time for a complete design with all possible considerations. **PDF of schematics can be found in our GitHub repo.**

External sensor module

The external sensor module connects to the main one by an I2C bus that is pulled up by 4.7k resistors to 3.3V this is done on the main module. All of the components on the external module use 5V power so there is no need to have a 3.3V line connecting the two. The only wires are 5V, GND, SCL and SDA.

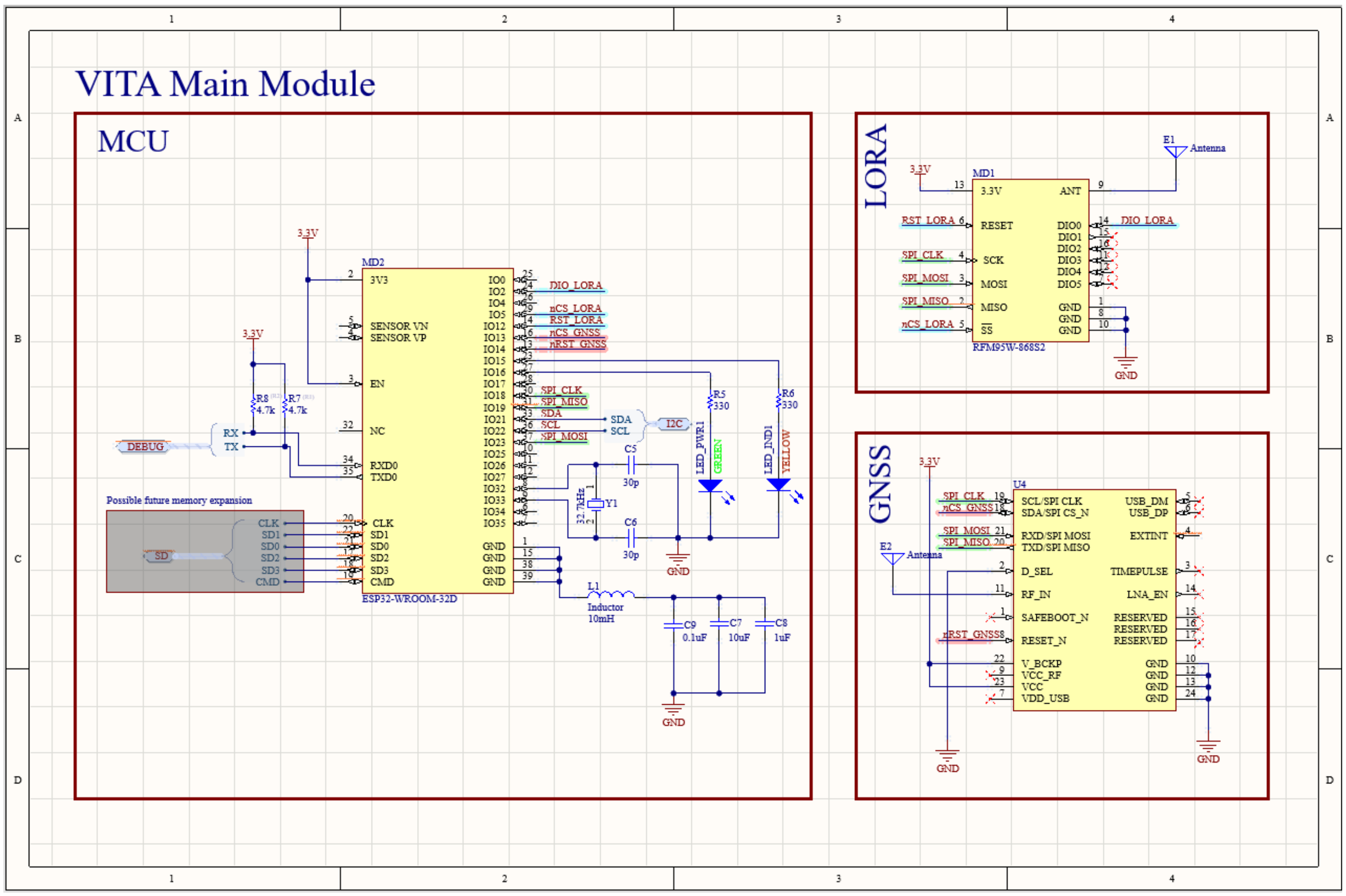
The ultrasonic sensor needs a trigger signal that would make it emit a sound for whose reflection it will listen. For this purpose there is an NE555 oscillating circuit that generates a 4.8Hz square wave that will generate this signal. The resistor and capacitor values were calculated using the following equation:

$$f = \frac{1.44}{(R_4 + 2R_3)C_1}$$



Main Module

The main module has contains the ESP32 MCU, LoRa and GNSS modules. All of them are located on the same SPI bus where the MCU is the master and the other two modules are slaves.



Communication protocol

For the best application, the infrastructure in the concept should be the existing the mobile communication infrastructure of the city. By doing this, we could make use of the central computing and central database of the whole city!

Message protocol:

Every message: data message or set-up message (signalling) uses SSL protocol which is the TCP protocol enhanced with the security feature. Why TCP even though it has overhead for setting up? Because the information contained in the data message such as car speed, car position, etc. are very important for the safety of vehicles. Consequently, the integrity of the messages cannot be compromised.

Protocol for actions

Some general rules:

Wifi/Bluetooth module is in SLEEP state when it is not called by the microcontroller. It wakes up to be ON when get called.

While the GPS or Lora module is in OFF (when they are not used) or ON state (when they are used).

There are two important modes of working for the device:

1. Inside the city
2. Outside the city

Outside-or-inside protocol

The ESP microcontroller always turns on the Lora module to send the request to connect to any base stations (the infrastructure in the concept) periodically, at the same time it also sets up a timer for 4 minutes for example. If the ESP microcontroller receives any response from any Base station before the timer is up, it proves the vehicle carrying VITA is inside the city. Otherwise, the controller turns on the GPS module to make sure that its position is inside the city. And so, the vehicle can avoid making false assumption that it was outside the city due to the bad quality of the transmission channel with the base stations. Then the ESP operates with the mode inside the city.

If the GPS tells it that the position is outside the city, it switch to the mode for outside the city.

Protocol inside the city:
ESP turns on the Lora module in order to exchange information with the base station. The exchanged information could include the traffic in the neighborhood, the empty parking slots, etc. The device user can have the choice to interrupt the connection with the base station and start to talk directly with their neighbours using Wifi/Bluetooth. When the option is chosen by the user, ESP turns off Lora and wakes up Wifi/Bluetooth modules. When the ESP detects the Wifi/Bluetooth is in IDLE state, it turns on the Lora again.

Because when Lora is ON, it always periodically sends request message to connect to any Base station to check where it is (inside or outside the city), if the ESP knows that it is outside. It switches to the outside mode.

Protocol outside the city:
ESP turns off Lora module, turns on GPS. The GPS module will periodically update the position for the user.

When outside the city, the mobile infrastructure does not exist, however, its aim is to provide the traffic information of its surrounding to the user in order to avoid unwanted accidents. In order to achieve this safety, the ESP turns off the Loral module, wakes up its Wifi/Bluetooth module in order to detect any incoming connection from other vehicles. If any vehicle is in the communication range, they start to exchange messages like the vehicle speed, their locations to each other. This information can help user to decide which actions are the safest for them.

With the periodical message from the GPS, the user can know if their vehicle comes back to the city or not. If they are back to the city, ESP switches to the inside-city mode.

The benefits of the protocols for actions:

1. Only necessary modules are ON when needed. It mitigate the computing power of the ESP, as a result, the microcontroller can focus its full power for maintaining the required actions.

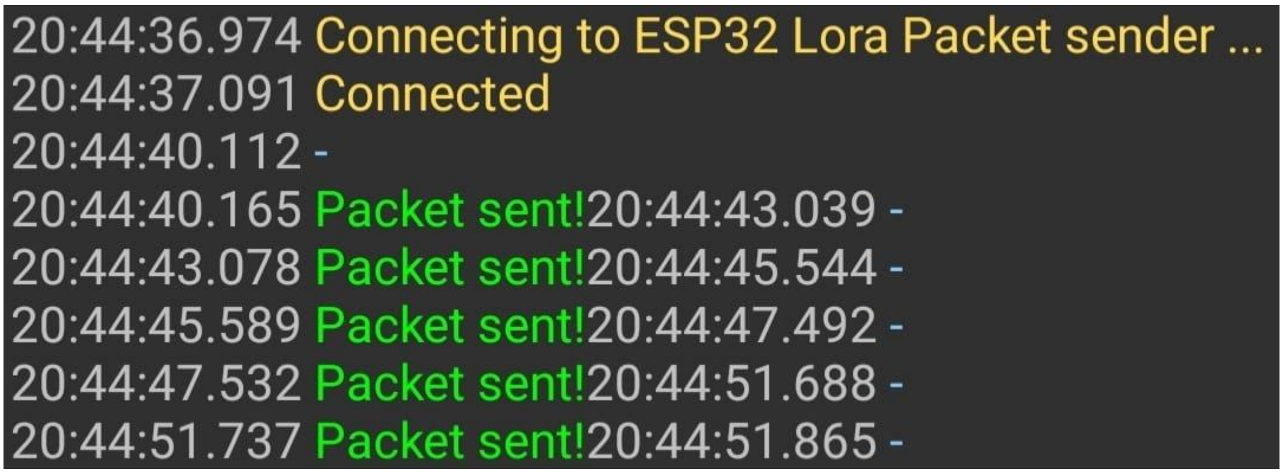
2. The consumption power of the system might be less as the modules are called only when they play significant roles in guarantee to provide necessary information for the user so that they can be safe while travelling.

Prototype

To make a proof of concept we gave our best to make a prototype of a device similar to the one we have described in our project. We didn’t have all of the components and as extensive as this would take months to make perfect, but we did manage to do something.

We successfully made a LoRa “base station” that was able to transmit and receive messages from our prototype device. We were also able to forward received LoRa messages to our phone via Bluetooth and to initialize a LoRa transmission with a Bluetooth message from our phone. We also successfully connected to our Wi-Fi network just to test how it works. Since this competition was focusing on hardware we mostly used some template firmware code and edited it just for testing.

You can also see some picture of our device with the enclosure we made as well as some screenshots from testing.



```
20:44:36.974 Connecting to ESP32 Lora Packet sender ...
20:44:37.091 Connected
20:44:40.112 -
20:44:40.165 Packet sent!20:44:43.039 -
20:44:43.078 Packet sent!20:44:45.544 -
20:44:45.589 Packet sent!20:44:47.492 -
20:44:47.532 Packet sent!20:44:51.688 -
20:44:51.737 Packet sent!20:44:51.865 -
```