LoRa GSM Gateway

Max-Felix Müller

2020
September
-
October

Contents

List of Figures

1 Introduction

For testing the range of different antennas and to get sensor data into the cloud, a LoRa Gateway in a remote location can proof very useful. By placing it on a larger hill, the range will significantly improofe over having it at home. Line of sight is very important for low power signals and radio signals overall.

1.1 Gateway

A gateway is a component either in hardware or software, that is used to create a connection between two systems. A gateway will modify the data before passing it forward. Depending on the data this modification can happen on all layers of the OSI-model.

1.2 GSM And GPRS

Originally from france, the Groupe Spécial Mobile is now called Global System for Mobile Communications. It was introduced in 1990 and is the first standard of the second generation 2G. In 2020, we are currently being introduced to the fifth generation 5G of mobile communications.

It uses a combination of frequency multiplexing between the transmitter and receiver to support full duplex operation, and time multiplexing for the data. That means that each device has a time slot assigned in which it can communicate. GSM 900 uses frequencies between 890 MHz and 915 MHz with 124 channels for the uplink and another 124 channels in the downlink between 935 MHz and 960 MHz.

For the modulation a kind of frequency shift keying called GMSK Gaussian Minimum Shift Keying is used. The gaussian curve flattens the edges of digital signals and thus reduces higher harmonics and the necessary bandwidth for the data transfer.

The GPRS General Packet Radio Service is the name of the service for data transmission in GSM networks. The data is split into packets at the sender and then recombined at the receiver. The advantage of the packet oriented service is that there is no continuous connection between the devices necessary and therefor other devices can use the same band in time division multiplexing.

1.3 LoRa

LoRa is an abbreviation for Low Power, Long Range. That also defines the use cases. LoRa is intended to be used with battery powered IoT devices that have to last for a long time and still trasmit data from a remote location. The low power and long range come with the downside of reduced bandwith but for sending some sensor data every once in a while this is perfectly fine.

LoRa works in the different ISM bands at about 433 MHz, 868 MHz and 915 MHz. This gateway is desiged for use only with 868 MHz devices.

LoRa uses CSS Chirp Spread Spektrum modulation. The symbols are transmitted in chirp impulses that change frequeency continuously over time. With the increased sensitivity of LoRa receivers even signals under the noise can be detected and decoded, allowing for very weak signals from far away.

2 Hardware Overview

The gateway is build up of three parts. The LoRa Board which receives data over LoRa. It was the cheapest one I could find on Amazon so that's the one I choose.

The data received by the LoRa board is captured with a Teensy 4.0. The Teensy then performs some checks to confirm that the captured data is compatible to the protocol I defined. Only when the data fits it is accepted and passed along.

The GSM Shield, originally intended for use with an Arduino, then connects via the GSM network to my server. The MQTT broker running on the server will receive the messages and push them to all the clients subscribed to that topic.

2.1 LoRa Board

The cheapest LoRa board with a SMA connector from Amazon was used. The SMA connector is necessary to be able to connect different antennas to compare their performance. The EBYTE E20 868T20D was selected. It is advertised with TCXO which should be a temperature controlled oscillator granting good frequency stability even with change of the outside temperature. That is useful when the gateway is deployed outside and left there during winter and summer.

The cheap price comes with the cost of poor documentation and very little software support. Some person released open source code on GitHub which serves as a base for this work.

Figure 1: The LoRa Board from Amazon. The price is an incredible 3,50€

2.2 GSM Shield

The GSM Shiled for the Arduino was build in two versions. This is the first one. It was discontinued rather quickly. One possible reason might become obvious in a later charpter.

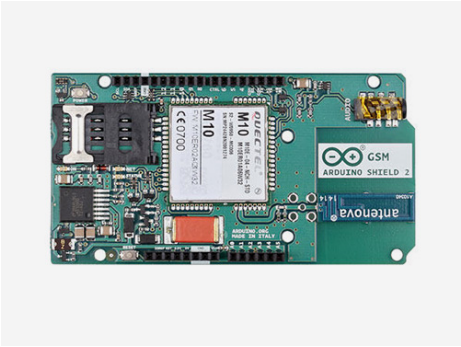

Figure 2: The GSM Shield for sending SMS and connecting to the internet with an Arduino

2.3 Teensy

Figure 3: The Teensy 4.0 is a microcontroller that can be used with the Arduino bootloader. It has much more storage and a faster clock compared to the Arduino

For me personally the Teensy is the next generation of Arduinos. The Teensy has an incredible amount of storage onboard and a very high clock speed. That enables its use for nearly every hobby project ever.

3 Software

The main part of this project is the software running on the gateway. It has to continuously check for new messages via LoRa. At the same time, to save energy, the GSM connection has to be turned off for as long as possible. Therefor the software has to provide storage for the messages. While a GSM connection is established, new messages have to be handled at the same time.

3.1 GSM Shiled Library

Since the GSM Shield was made by Arduino, there is a libraray for use with the shield. However, when using it with the Arduino, the network or the module randomly disconnect or do not work at all.

When switched to the Teensy the output of the code in the serial terminal was very different, indicating that the Arduino just was not fast enough to be used with the shield (for internet access). To send some SMS it was probably fast enough, but that might very well be the reason why the shield was discontinued.

Instead of relying on the library a different library was created using AT commands to communicate with the shield.

3.2 AT Commands

AT commands are now the standard for configuration of modems. The letters AT stand for attention. By sending AT before every command and measuring their length, the connection speed (baud rate) can be measured and set automatically.

Using the extended command set (AT+) different commands can be sent to initialize a internet connection and send data via that connection. There is a datasheet listing all the possible commands with parameters and information on the returned messages for the chip that is on the GSM shield.

The steps necessary for the connection in the right order will be displayed in the state machine section.

3.3 LoRa Protocol

There has to be some handling of the LoRa messages in such a way, that only allowed messages will get passed on to the server. The protocol has 4 parts: A header, the data bytes, one byte checksum and a signal for the end.

The header is fixed to 0x00 and the end is 0xFF. To calculate the checksum, all bytes of the message are added, allowing for overflows. The result of that calculation is then appended to the message before the end byte.

To check compliance of any received message with the protocol, a receive state machine is implemented. The Teensy will continuously read bytes until a 0x00 is received. Then the bytes are stored in a temporary buffer until a 0xFF is received. Next the checksum is checked by performing the addition of all data bits and comparing the result to the last byte before the 0xFF. If the checksum is correct, the message and its length will be stored in a buffer, and, if not already the case, the transmission is initiated.

There is the potential problem, that 0x00 or 0xFF is part of the data within the message.

If there is a 0x00 it does not matter. In case the Teensy missed the initial beginning of the message, the checksum is unlikely to fit. Otherwise, if the message was captured correctly from the beginning, following 0x00 bytes will be ignored until the first message ends.

The real problem occurs with 0xFF. This will cause the Teensy to end the message. Then the checksum will probably fail, thus no message is forwarded at all.

The sender must be aware of this situation and possibly avoid 0xFF, be it by reducing it to a 0xFE and accepting the changed data point. One could compensate with an additional bit indicating that such a change took place.

3.4 MQTT and MQTT Packets

MQTT is a protocol build on top of TCP/IP. It can be used to send messages from one device to multiple other devices. To do so, a device sends its message as a publish packet to the so called broker. Each device that wants to listen to the messages can subscribe to a channel (called topic) on the broker. The broker will relay the received publish packets to all subscibers.

Before publishing or subscribing to a topic, the devices must connect to the broker. The connection can be open or locked behind a login with username and password to make the connection more secure.

The MQTT protocol uses different headers for the packets and length bytes to indicate the number of data bytes following. A helper libraray was created to add this information to messages.

3.5 State Machine

The transmission via the GSM network happens inside a state machine. Reasoning behind this is the fact that a delay is necessary between the different steps of the connection. Also the LoRa receiver has to run in the background and still capture new messages. Thus it is not possible to use delays. The time lost not receiving LoRa packets would be too long for the internal serial buffer of the Teensy to store the data for that time.

3.5.1 Idle State

The idle state is only reached for a brief moment to initiate the reset. Otherwise, while in idle, the transmission bit will not be set and thus the state machine is not even evaluated most of the time.

3.5.2 Reset State

The reset state will complete the reset initiated when exiting the idle state. When resetting the GSM module also the buffer gets cleared to void weird overflows.

3.5.3 Pin State

The pin state is used to enter the pin of the sim card. To use the iternet access, the pin has to be entered. If only surronding networks shall be discovered, this would no be necessary.

3.5.4 APN State

The APN Access Point Name is the name of a gateway to which the mobile device shall connect. It also sets the network that the GSM module will connect to.

3.5.5 GPRS State

Before connecting to a server the service for packet transfer has to be started. In this state, the GPRS is initiated.

3.5.6 Server State

Now it is possible to log onto the server. Since the MQTT protocol is used, the port has to be set correctly. I use the port similar to a pin code, since I have changed it from the standard port.

3.5.7 Connect State

Being logged in at the right port, now the connect packet can be sent to the MQTT broker. There currently is no check if the connection was accepted, so I just hope. To connect, the connect packet is sent which includes an ID, username and password.

3.5.8 Publish State

In the publish state the messages in the buffer are published one by one. Between the messages there is a small pause that is covered by the state machine, meaning new messages will still be received during this time, as long as the buffer is not full. Messages are transmitted until the transmit pointer is at the same position as the buffer end pointer.

3.5.9 Disconnect State

When all messages are sent, the module disconnects from the server.

3.5.10 Pause State

To give the module a break, there is a pause of a few seconds where no new transmission will be initiated. If there are a lot of messages coming in very quickly, it is possible that the message buffer fills up in this time and some messages might not be handled. However, it is not expected that such an amount of traffic is to be handled very soon. If it is, there is a lot more that will have to change.

4 Radio Frequency Spectrum

For those interested in the RF part of this project (as I am), I decided to have a look at the tranmitted signals using a SDR.

I will have to have a deeper look into both signals, but here is a brief overview.

4.1 The LoRa Signal

As stated before, the data is transmitted by sweeping over a frequency band. The change in frequency can for example be seen in the time domain.

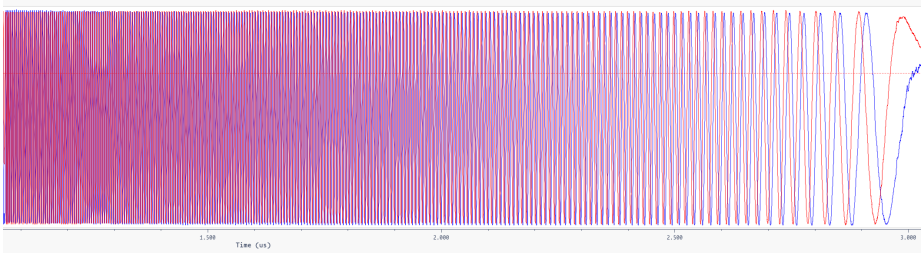

Figure 4: A LoRa message in the time domain. The frequency is changed over the duration with the lowest frequency being on the right of the image

Looking at a waterfall display shows the chirp signal even better. The waterfall block in gnuradio has to be speed up to show the signal cleanly. There are sweeps to synchronize the sender and receiver.

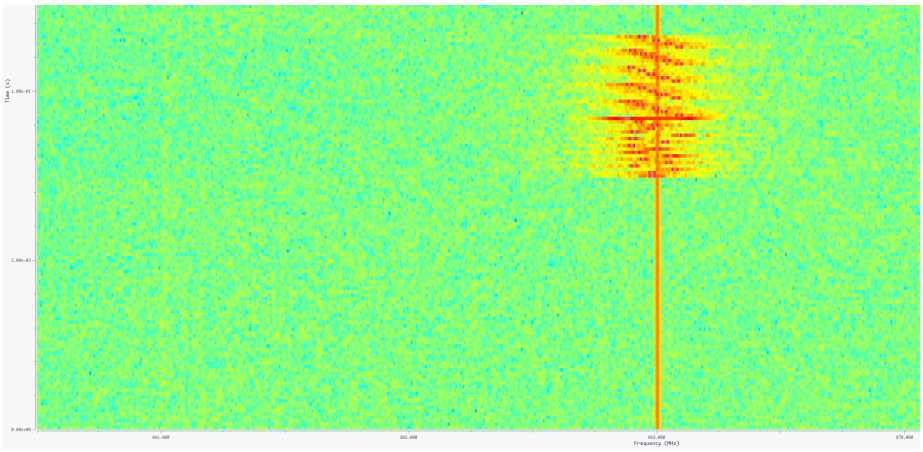

Figure 5: The waterfall display shows the frequency change over time. The sweeps at the top are the preamble of the signal. With the following sweeps in the other direction (to lower frequency) the devices are synchronized

4.2 The GSM Signal

Again, as described earlier, the GSM signal is in frequency shift keying. The signal here is just at one frequency. By converting the Q and I samples to a magnitude, the smooth edges of the gaussian filter become obvious.

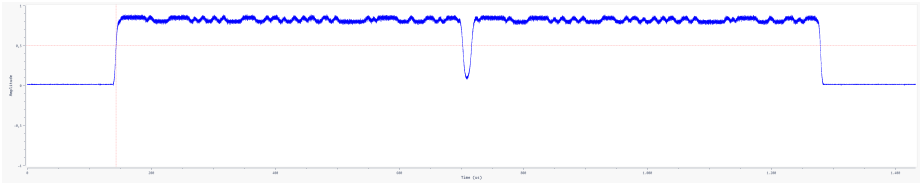

Figure 6: The GSM signal as magnitude in the time domain.

5 Deploying The Gateway

The gateway will be deployed on a hill on my property. It will be powered using a car battery that is charged during the day using a solar cell and a mppt charger. Using a voltage divider, the battery voltage can be monitored and a message can be sent for example by SMS to inicate a low voltage.

6 Links

GitHub:

EBYTE LoRa module code, forked from KrisKasprzak
$https://github.com/MxFxM/EBYTE$

Code for GSM module with Teensy
$https://github.com/MxFxM/Teensy_GSM_Shield$

Code to create MQTT packets
$https://github.com/MxFxM/Teensy_Mqtt_Helper$