Hello YouTubers, here is the guy with the Swiss accent. If you plan to start with LoRa and do not want to spend lots of money you can start with an Arduino Uno and a Dragino shield for around 20 dollars and connect it to the nearest gateway. In episode #117 I built such node. But, if no gateway is around your location? Then, you can build your own full blown public gateway as I did in episode #115. This costs you around 300$. Which is too much for most of us. Today, we will build a gateway for much less using an old raspberry and a LoRa hat. The hat is around 30$ and the whole gateway is quite simple to build. And I will show you its limitations, and what you have to do to avoid problems. In parallel, we will learn a little more about the LoRa protocol.

So, let’s get started with assembling the hat and the Raspberry Pi. This is straight forward. I use a small Brass stand to make it more stable. This hat also has a GPS module, but today, we do not use it.

After assembling the hat and the Pi, we have to create an SD card with the needed software. We start with an SD card with an installed Raspbian jessie release. You find a link in the description on how to proceed.

Now, we can start to setup the rest. You also find a link with the tutorial on how to do that and I will follow this document for the next steps. The tutorial does not use a Pi hat, it just uses the bare RFM95W module. This is also a possibility, if you want to save another 20 dollars, which is the difference between the Pi hat and the bare module.

According to the instructions, you first have to clone the repository of the gateway software. Then, we have to enable SPI, if not already done. Next, we have to install wiringpi. This software enables the possibility to control the various GPIO pins of the Pi.

In the next step, we have to adjust some constants in the main.cpp file. Please check first the pin assignment. Then, you change the server address, the frequency your gateway runs on, and the latitude and longitude of your gateway. I use here 868.1 MHz, but you can use also .3 or .5, which are the three mandatory channels for TTN

Then, you make the executable file and start the gateway software. Be aware, that your gateway only works as long as this program executes. If you press ctrl C, the program stops and your Pi is no more a gateway.

Right at the start of the gateway software, you get the unique number of your device. This will be needed to connect to the TTN network later. So, write it down and remove all colons.

Now, you go to the page of TTN and create an account (if not already done). Then, you enter the management console and register a new gateway as “packet forwarder”. Key-in your gateway number from before and enter the frequency plan. If you did not enter the position of your gateway in the main.cpp file, you have now a chance to enter its position here.

After registering the gateway, you should see status “connected” and a small number of seconds in the field “last seen”. That’s all. Your gateway is now ready and can receive and transfer messages!

Now, apart from the price, what are the differences of the expensive gateway on my roof compared to this one in my lab. There are three main differences:

1. My small gateway here, as its name says, has only one channel
2. It can only deal with one spreading factor at a time
3. Currently, it has no downlink possibility. However, this is not a hardware limitation and maybe, once somebody will enhance the software to include this feature

So, let’s play around with it to discover, if these differences really matter. To do that, I have here my LoRa node from episode #117. It sends every minute a message in the air. Let’s see now on TTN, what happens.

The first message sent by the node arrives and we see, that it was received by both gateways. Great! Our small gateway works! The signal on the roof was, of course, much weaker. But still acceptable. Spreading factor was SF7.

The next message, however, did only arrive at the gateway on the roof. Why is that? I did not change anything and the signal should still be very strong.

So, This cheap gateway does not to be very reliable. Let’s give it a second chance and wait for the next message.

And also here, it is not received by my new gateway. So, I give it a last chance. Maybe it knew, that it was its last change. It transferred this message again. So, this small device is not very reliable. Should I throw it into the trash bin? Was this just a joke?

If we have a close look at the transmission frequencies, we see, that the node changed it with every message. This is a good behavior. If channel one would be disturbed, we would get at least the next message on channel two. But we said before, our single channel gateway was compiled for channel 0. So, it did not hear the signal on channel 1 or 2. So, we have at least an explanation of its behavior. But how can we deal with that? Let’s assume, our node would always send its messages on channel 0, then, our new gateway would receive all messages and we would have resolved this issue. But what kind of other problems do we cause by that?

For a productive environment with hundreds of nodes, the concept of frequency hopping is useful. But if you just want to test the technology or use it around your house, then you only have a handful of sensors, and this frequency hopping is not necessary. So, we just have to find a possibility to switch it off in our node software. This is easily done by this command. We just disable all channels we do not want. For Europe, these are channels 1 to 8. Now, all messages are sent on channel 0, which is 868.1 MHz. So, we have now a gateway for our own use. And even our neighbor can use it, if he constrains his node to just our channel, too. No problem.

Just to show you, that the node can only receive one spreading factor at a time, I change my node to SF12. An, the message is not received, even on channel 0. So, we also have to make sure, that the nodes and the gateway uses the same Spreading factor.

So, summarized

* We built a simple a less expensive TTN gateway with a raspberry Pi and a LoRa hat
* The software is easy to be installed and adapted
* If we make sure, that our node only uses one channel, this setup is very usable for our own purpose
* The capacity of such a node is good for at least 20 concurrent sensors or more, depending on the sending intervals.

And the best: Because we are connected to the ttn network, we can profit from other things. For example, we easily can connect our devices to our node-red installation. A TTN node already exists and we can connect our device straight to node-red. At the end, we see here as an example the frequency of the transmission and the RSSI values of both gateways. With this, you can now attach your sensors (or buttons) via LoRa and TTN to your home-automation system.

I hope, this video was useful or at least interesting for you. If true, then like. Bye