

# LoRa TTN gateway with Orange Pi Zero and RAK831.

## 1 Introduction.

This document describes how to build a LoRaWan gateway. For this gateway a RAK831 gateway module and an Orange Pi Zero (the so-called “Backhaul”) is used. Info about the RAK31 can be found at <http://www.rakwireless.com>. The RAK831 is compatible with the iC880A-SPI.

The decription is based on Armbian 5.38 for the Orange Pi Zero, also called “ARmbia Stretch” .

The bash-commands in this document start with “#” or a “\$” to show if a command must be given as user “root” or as a common user. Do not include this character in the commands.

## 2 Orange Pi Zero configuration.

For this project an Orange Pi Zero is used.

Other Orange Pi models are possible, but it is vital that an (extra) SPI connection is available on the connector. The Orange Pi PC Plus lacks this feature.

The used Linux distribution is Armbian 5.38 (see <http://www.armbian.com/orange-pi-zero>). For installation of Armbian an SD card (16 GB) must be prepared. I used “Etcher” for this purpose. With the SD card the Orange PI Zero can be booted up. Connect the OPi through an Ethernet cable with your local network or through a USB-TTL converter (115200 baud) connected to the OPi. The IP address of the OPi should be visible in the admin pages of your router.

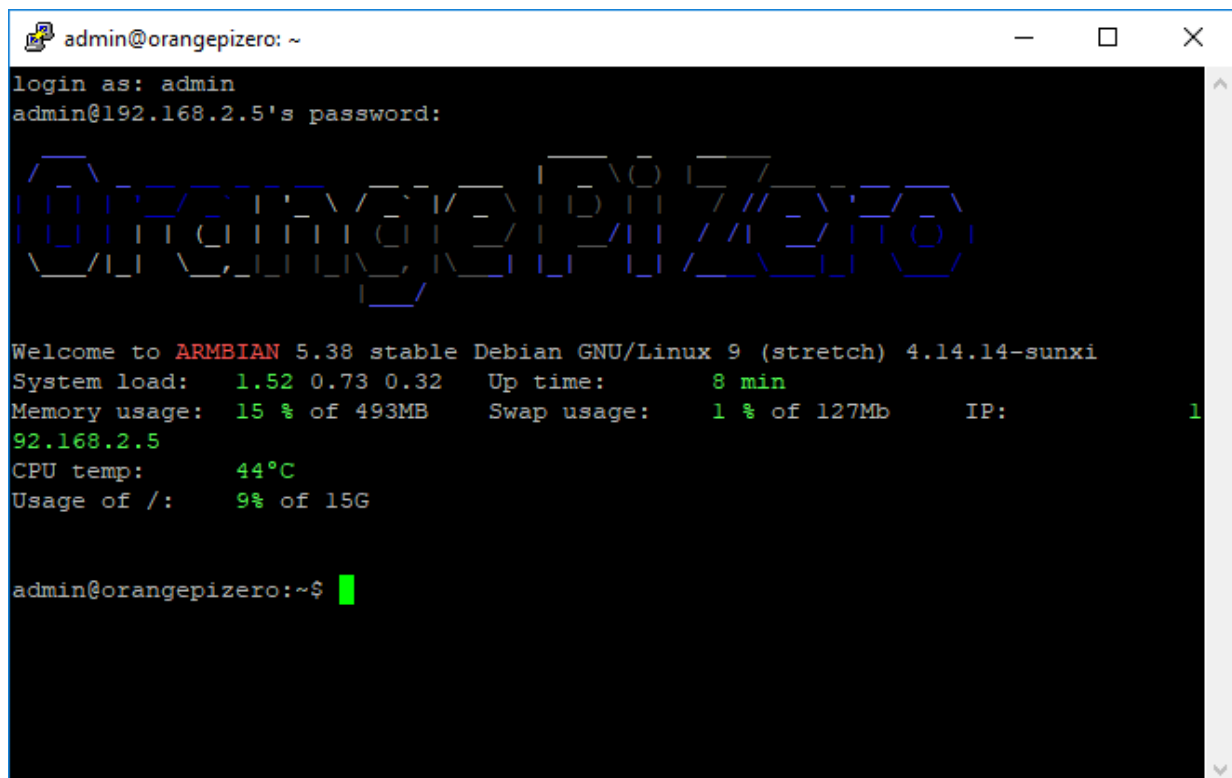
Now the Orange Pi can be connected through SSH (I used puTTY for this).

At first you may login as user “root” with password “1234”. The password must be changed immediately. You have to enter the old password first (“1234”) and then the new one.

After that you will be asked to add a new (not root) user. Add for example a user “admin” and enter a password and other data for his user.

Update the software with “apt-get update” and “apt-get upgrade”. This will take some time.

If you log in later as “admin”, you will see a screen like this:



```
admin@orangezipero: ~
login as: admin
admin@192.168.2.5's password:
Welcome to ARMBIAN 5.38 stable Debian GNU/Linux 9 (stretch) 4.14.14-sunxi
System load:  1.52 0.73 0.32   Up time:    8 min
Memory usage: 15 % of 493MB   Swap usage: 1 % of 127Mb   IP: 192.168.2.5
CPU temp:    44°C
Usage of /:   9% of 15G

admin@orangezipero:~$
```

## 2.1 Enable WiFi.

For your convenience enable WiFi with the command “nmtui”. Choose “Activate a connection” from the menu. Here you can also set the hostname, for instance “GwLoRa”, but that can also wait until you run the gateway install script.

## 2.2 Timezone.

To show the right clock time, you have to set the time zone. Check the setting by using the “date” command. For the Netherlands (for example) you have to give the following commands as “root”:

```
# rm /etc/localtime
# ln -s /usr/share/zoneinfo/Europe/Amsterdam /etc/localtime
```

## 2.3 Enable SPI.

SPI is probably not enabled in this distribution. Check if /dev/spidev1.0 exists. Spidev0.0 is used for the on-board flash.

If there is no /dev/spidev1.0, you have to add an overlay for it. The document /boot/dtb-4.14.14-sunxi/overlay/README.sun8i-h3-overlays gives some information. The right overlay “spi-spidev” should be added to the “overlays”-line in /boot/armbianEnv.txt. Change this line from:

```
overlays=usbhost2 usbhost3
```

to:

```
overlays=usbhost2 usbhost3 spi-spidev
param_spidev_spi_bus=1
```

And reboot the OPI.

Now /dev/spidev1.0 should be present.

You may check the SPI by connecting a LED (in series with a 1 kΩ resistor) at pin 23 (SCLK) or pin 19 (MOSI). Use the command:

```
# cat /dev/random >/dev/spidev1.0
```

The LED will light and show some flickering. Abort with Ctrl-C.

## 2.4 Enable GPIO.

GPIO is enabled in this distribution. Again, this could tested with be a LED (in series with a 1 kΩ resistor) connected to a free GPIO pin.

It should be possible to control for instance GPIO10 (pin 26) with the commands:

```
# echo 10 > /sys/class/gpio/export
# echo out > /sys/class/gpio/gpio10/direction
# echo 1 > /sys/class/gpio/gpio10/value
# echo 0 > /sys/class/gpio/gpio10/value
```

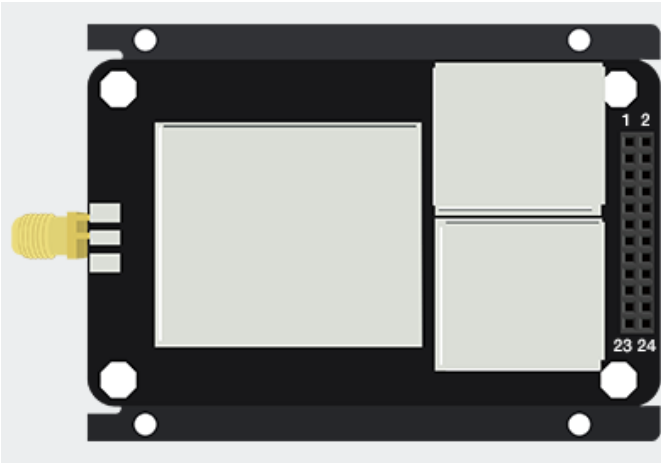
## 2.5 Install tcpdump.

Tcpdump can be a handy tool for debugging the messages to and from TTN. Install it with:

```
# apt-get install tcpdump
```

### 3 RAK831 connections.

The lay-out of the RAK831 connector looks like this:

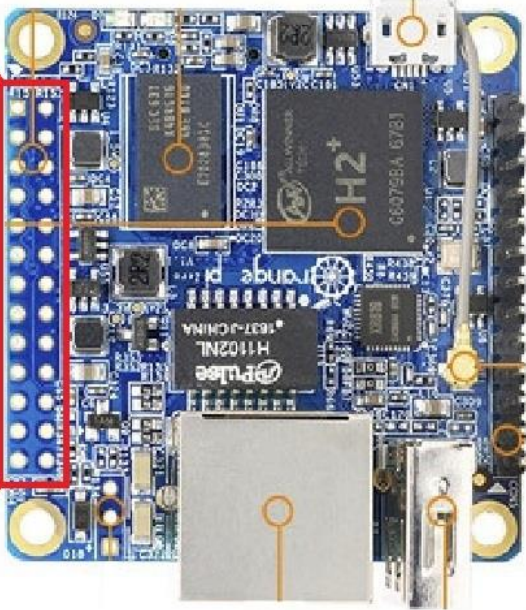


+5V	1	+5V	2
GND	3	LNA_EN_A	4
GND	5	GND	6
RADIO_EN_A	7	PA_G8	8
RADIO_EN_B	9	PA_G16	10
PA_EN_A	11	GND	12
RADIO_RST	13	GND	14
CSN	15	MOSI	16
MISO	17	SCK	18
RESET	19	GPIO0	20
GPIO1	21	GPIO2	22
GPIO3	23	GPIO4	24

#### 3.1 Orange Pi Zero I/O pins.

The lay-out of the Orange Pi connector looks like this:

Alternate Function		Pin No.		Alternate Function
	3.3V PWR	1	2	5V PWR
I2C1 SCA	GPIO 12	3	4	5V PWR
I2C1 SDL	GPIO 11	5	6	GND
	GPIO 6	7	8	GPIO198
	GND	9	10	GPIO199
UART2 RX	GPIO 1	11	12	GPIO 7
UART2 TX	GPIO 0	13	14	GND
	GPIO 3	15	16	GPIO 19
	3.3V PWR	17	18	GPIO 18
SPI1 MOSI	GPIO15	19	20	GND
SPI1 MISO	GPIO 16	21	22	GPIO 2
SPI1 SCLK	GPIO 14	23	24	GPIO 13
	GND	25	26	GPIO 10

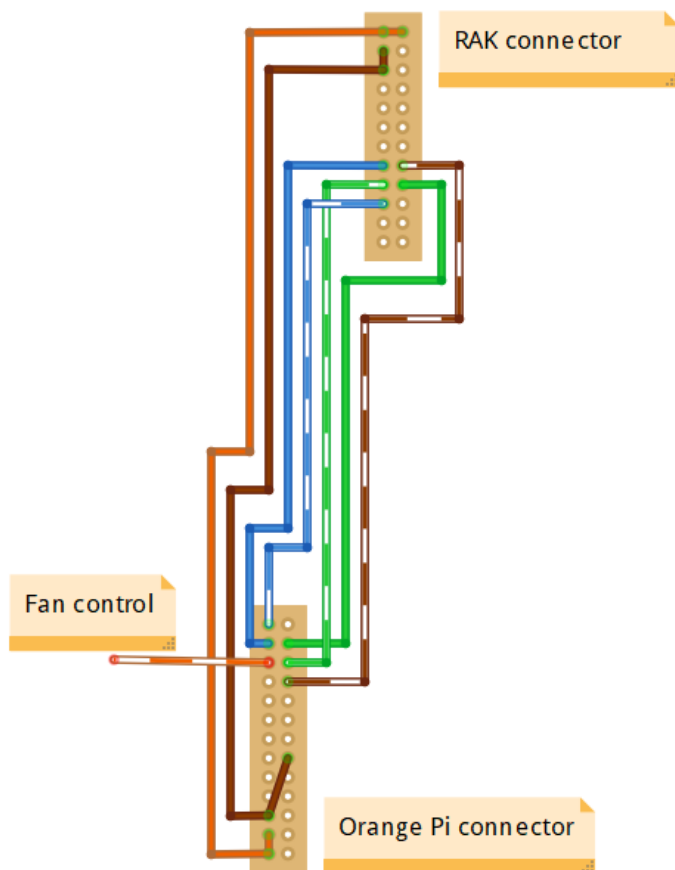


### 3.2 Connecting the RAK and the Pi.

The power supply (5 Volt and ground) is connected to pins 2 and 6 of the Orange Pi respectively. The micro USB connector will not be used. For the connections between the RAK831 and Orange Pi, a cable is prepared with the following wiring:

RAK831 pin	Description	Orange Pi pin	Description	Color
1,2	+5 Volt	2, 4		Orange
3,5	Ground	6,9		Brown
15	CSN (chip select)	24	GPIO13 - SPI1 CS0	Blue
18	SCP (SPI clock)	23	GPIO14 – SPI1 SCLK	Green
16	MOSI	19	GPIO15 – SPI1 MOSI	Brown-white
17	MISO	21	GPIO16 – SPI1 MISO	Green-white
19	RST (Reset)	26	GPIO10	Blue-white
		22	GPIO2 - Fan control	Orange-white

The cable looks like this:



The 24 pin RAK connector is “male”, the 26 pin Pi connector is “female”.

When the cable is connected and the power supply is switched on, a red LED will light near pin 3 of the RAK831 header. For a picture of the cable, see the pictures at the end of this document.

## 4 Install the gateway software.

For this the installer from ttn-zh can be used. This installer is meant for a iC880A, but it will also work for a RAK831. During the installation the latitude and longitude of the position of the gateway must be entered. These numbers can be found at <https://www.latlong.net>. For example: N = 51.65571 and E = 5.037537. These numbers are stored in /opt/ttn-gateway/bin/local\_conf.json after installation.

Install the software as follows:

```
# git clone https://github.com/ttn-zh/ic880a-gateway.git /root/gateway
```

The install script is now in /root/gateway. Go to this directory:

```
# cd /root/gateway
```

Start the script with:

```
# ./install.sh spi
```

Choose “N” as the answer to the question “Do you want to use remote settings file?”.

During the script you have to fill in some more data, like a hostname, a descriptive name and your e-mail address.

The scripts gives some warnings during the build process and will reboot the OPi.

At the end of the install script the next info will be showed (example):

```
Gateway EUI is: 0212FAFFFE3A7BE9
```

```
The hostname is: GwLoRa
```

```
Open TTN console and register your gateway using your EUI:
```

```
https://console.thethingsnetwork.org/gateways
```

```
Installation completed.
```

The gateway EUI is needed later on to register the gateway to TTN. The Orange Pi will reboot automatically. However it will not work in the current configuration. Read on...

The file /opt/ttn-gateway/bin/local\_conf.json contains now for instance:

```
{
  "gateway_conf": {
    "gateway_ID": "0212FAFFFE3A7BE9",
    "servers": [ { "server_address": "router.eu.thethings.network",
    "serv_port_up": 1700, "serv_port_down": 1700, "serv_enabled": true } ],
    "ref_latitude": 51.7557100,
    "ref_longitude": 5.1375370,
    "ref_altitude": 6,
    "contact_email": "info@example.com",
    "description": "ttn-RAK831"
  }
}
```

### 4.1 Reset signal for the RAK831.

After installation and reboot, the software will be started automatically through /lib/systemd/system/ttn-gateway.service.

The command:

```
# systemctl | grep ttn
```

Or

```
# service ttn-gateway status
```

Will show the running service.

Stop the service with:

```
# service ttn-gateway stop
```

The gateway needs to be reset before it is started. The reset line should stay low under normal circumstances. During reset the line must be pulsed with a short positive signal. For this you have to edit the script at /opt/ttn-gateway/bin/start.sh. At the beginning of this script you will find some lines to generate a reset on pin 25. Pin 25 does not exist for the Orange Pi. Change the pin number in line 4 from “25” to “10”.

## 4.2 Correction SPI device and platform name.

The communication with the RAK831 uses “native” SPI. However, in the standard setting, the wrong SPI device is used, “spidev0.0” instead of “spidev1.0”. That is why the service “ttn-gateway” will not start properly. In the logging at /var/log/syslog you will find the famous messages:

```
Feb 26 11:28:27 GwLoRa ttn-gateway[6294]: INFO: [main] Starting the concentrator
Feb 26 11:28:27 GwLoRa ttn-gateway[6294]: ERROR: [main] failed -1 to start the concentrator
```

Edit the file “/opt/ttn-gateway/lora\_gateway/libloragw/inc/imst\_rpi.h” and replace “spidev0.0” door “spidev1.0”. Change also the definition of “DISPLAY\_PLATFORM” to “RAK831 + OPi”.

Then recompile the software. Use the next commands:

```
# cd /opt/ttn-gateway/lora_gateway
# make clean ; make
# cd ../packet_forwarder
# make clean ; make
```

Restart the service with:

```
# service ttn-gateway restart
```

After some seconds the “RX” LED will light at the RAK831.

## 5 Switch service on/off.

You may stop the gateway service with:

```
# service ttn-gateway stop
```

Switch on again with:

```
# service ttn-gateway start
```

## 6 If it doesn't work...

If the gateway does not start, you will find in the logging: “ERROR: [main] failed to start the concentrator”. In that case the SPI bug is probably not functioning. First thing to do is: check the wiring.

/opt/ttn-gateway/lora\_gateway/util\_spi\_stress/util\_spi\_stress is a utility to check the SPI bus.

When you start this, a message “ERROR: lgw\_connect() did not return SUCCESS” may show. That points in the direction of an SPI bus problem.

lgw\_connect.c can be found at /opt/ttn-gateway/lora\_gateway/libloragw/src/loragw\_reg.c.

The definition of the SPI device is in /opt/ttn-gateway/lora\_gateway/libloragw/inc/imst\_rpi.h. Change the definition of “SPI\_DEV\_PATH” to “/dev/spidev1.0” and recompile everything.

You may set all debug-options in ./lora\_gateway/libloragw/library.cfg and recompile to more trouble shooting.

## 7 Temperature control.

The temperature in degrees Celcius can be made visible by:

```
$ cat /etc/armbianmonitor/datasources/soctemp
```

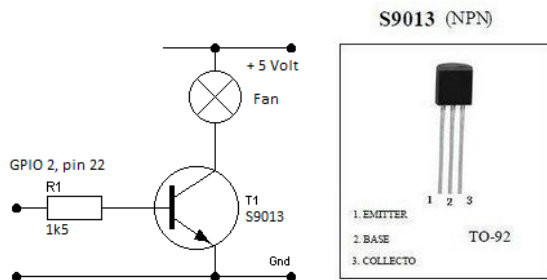
It looks like this temperature is multiplied by 1000.

More info is available with:

```
# armbianmonitor -m
```

This command is not very reliable. You may try the command with “-M”. After several tries it will work suddenly.

Then you see the CPU frequency, the system load and the temperature. We can control the temperature by switching a small fan on and off. We use GPIO 2 for this function. The diagram is as follows:



In parallel of the transistor (collector and emitter) I added a 39 Ohm resistor in order to switch between low and high revolutions.

A script to control the fan looks like this:

```
#!/bin/bash
#
# Power-up the fan if temperature is too high.
# 27-02-2018, Ed Smalenburg      First set-up
# 01-06-2018, Ed Smalenburg      Correction for temperature in "millidegrees".
#

# Define GPIO pin for fancontrol
GPIO=2
TEMP=45000

# Check if gpio is already exported
if [ ! -d /sys/class/gpio/gpio${GPIO} ]
then
    echo ${GPIO} > /sys/class/gpio/export
    sleep 1 ;# Short delay while GPIO permissions are set up
fi

# Set pin to OUTPUT
echo out > /sys/class/gpio/gpio${GPIO}/direction

while true; do
    if [[ $(cat /sys/class/thermal/thermal_zone0/temp) -gt ${TEMP} ]]; then
        echo 1 > /sys/class/gpio/gpio${GPIO}/value
    else
        echo 0 > /sys/class/gpio/gpio${GPIO}/value
    fi
    sleep 10
done
```

You may put this script at /usr/local/sbin/fancontrol.sh (chmod 700) and run it permanently by adding a configuration file at “/lib/systemd/system/fancontrol.service” with the following contents:

```
[Unit]
Description=Activate fan on high temperature

[Service]
WorkingDirectory=/usr/local/sbin
ExecStart=/usr/local/sbin/fancontrol.sh
SyslogIdentifier=fancontrol
Restart=on-failure
RestartSec=5

[Install]
WantedBy=multi-user.target
```

Then execute the following command:

```
# systemctl enable fancontrol
# systemctl start fancontrol
```

Now, the temperature will be held close to 45 degrees:

Time	CPU	load	%cpu	%sys	%usr	%nice	%io	%irq	CPU	C.St.
19:36:46:	1008MHz	0.85	11%	2%	2%	0%	5%	0%	45°C	0/7
19:36:51:	240MHz	0.78	1%	1%	0%	0%	0%	0%	46°C	0/7
19:36:57:	240MHz	0.72	2%	1%	0%	0%	0%	0%	46°C	0/7
19:37:02:	240MHz	0.66	2%	1%	0%	0%	0%	0%	45°C	0/7
19:37:07:	240MHz	0.61	2%	1%	0%	0%	0%	0%	45°C	0/7
19:37:13:	240MHz	0.56	1%	1%	0%	0%	0%	0%	45°C	0/7
19:37:18:	240MHz	0.51	2%	1%	0%	0%	0%	0%	44°C	0/7
19:37:23:	240MHz	0.43	2%	1%	0%	0%	0%	0%	41°C	0/7
19:37:28:	240MHz	0.40	2%	1%	0%	0%	0%	0%	40°C	0/7
19:37:34:	240MHz	0.45	1%	1%	0%	0%	0%	0%	40°C	0/7
19:37:39:	240MHz	0.41	2%	1%	0%	0%	0%	0%	40°C	0/7
19:37:44:	240MHz	0.38	2%	1%	0%	0%	0%	0%	41°C	0/7

## 8 Error logging.

In /var/log/syslog you can see the logging of the gateway. Relevant line contain the text “ttn-gateway”.

## 9 TCP logging.

The command:

```
# tcpdump -XUq port 1700
```

Can be used to inspect the traffic between the gateway and TTN.

If no node is active, you will see a status message to TTN every 30 seconds as well as a keep-alive message every 10 seconds.

When an (OTAA) node is switched on, it will do a JOIN REQUEST. That will activate a message of about 243 bytes. An ACK of 4 bytes will be returned. After 4 seconds TTN will return a JOIN ACCEPT of about 209 bytes.

## 10 Issues.

### 10.1 SSH login fails.

Sometimes SSH login via WiFi fails. When you send a ping to the IP-address of the gateway, it will take some time before the gateway reacts. After that, the login will succeed.

Now I turned off the WiFi power management:

Edit /etc/rc.local and insert before the last line:

```
iwconfig wlan0 power off
```

Reboot the Orange Pi.



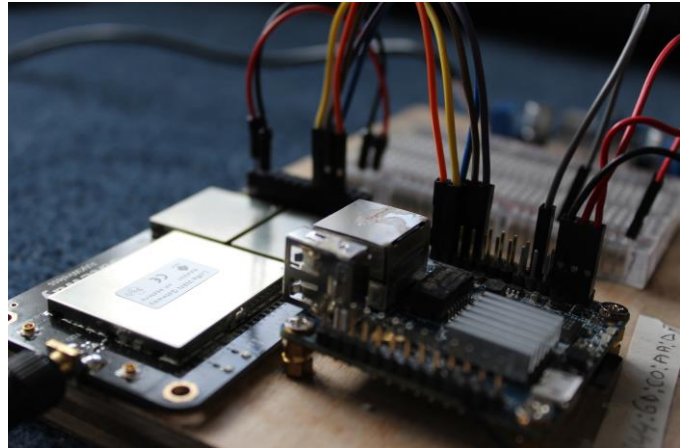
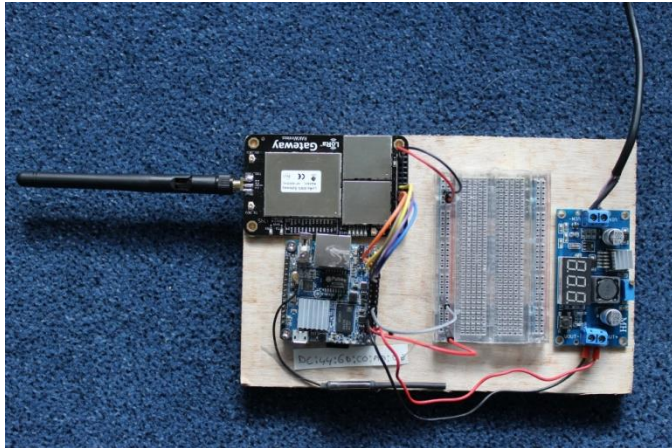
Let's see if this helps.

## **10.2 Armbianmonitor.**

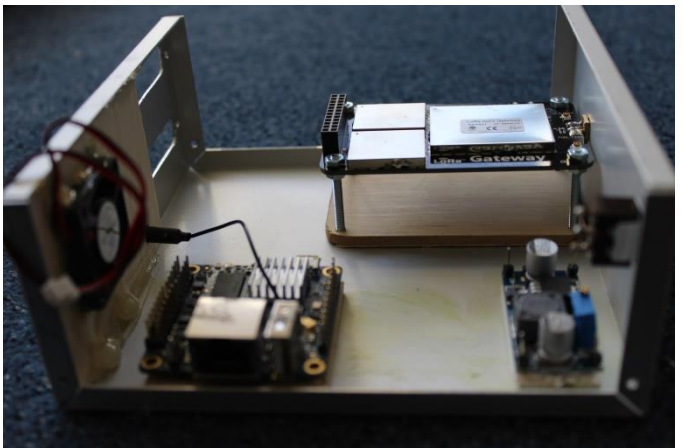
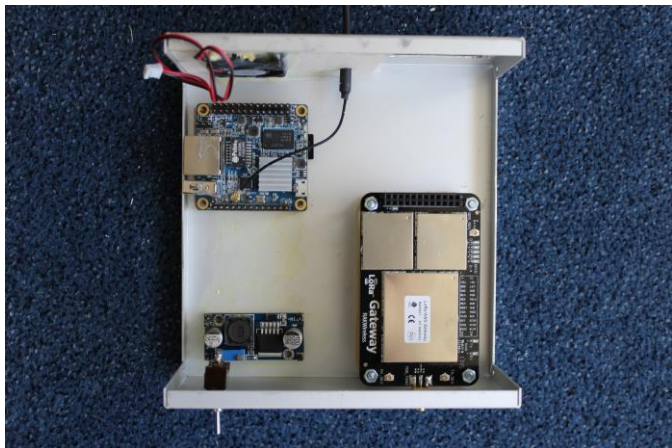
The command “armbianmonitor” fails now and then. I did not find a solution yet. The monitor is not used during normal operations.

## 11 Pictures.

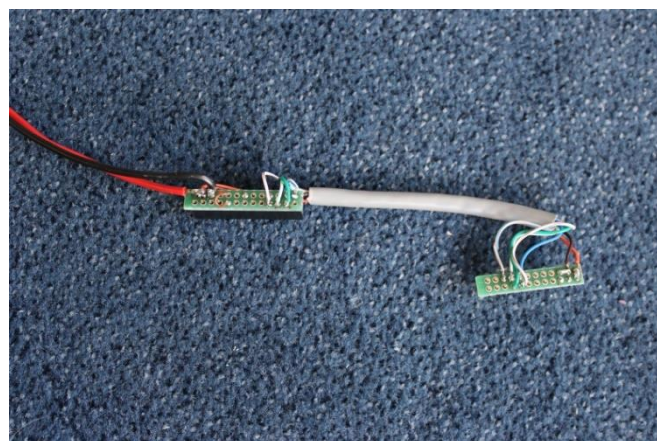
Test setting:



In a box:

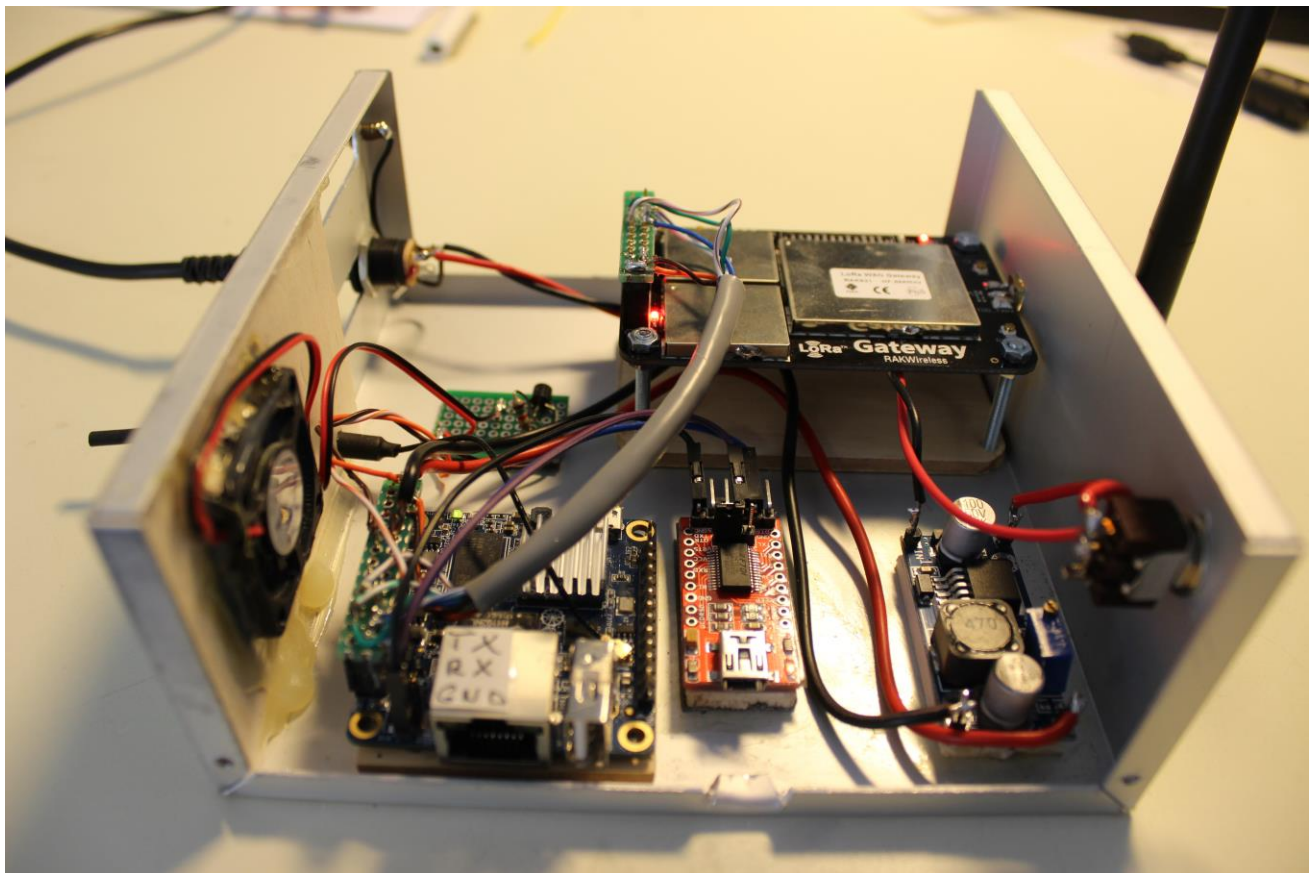
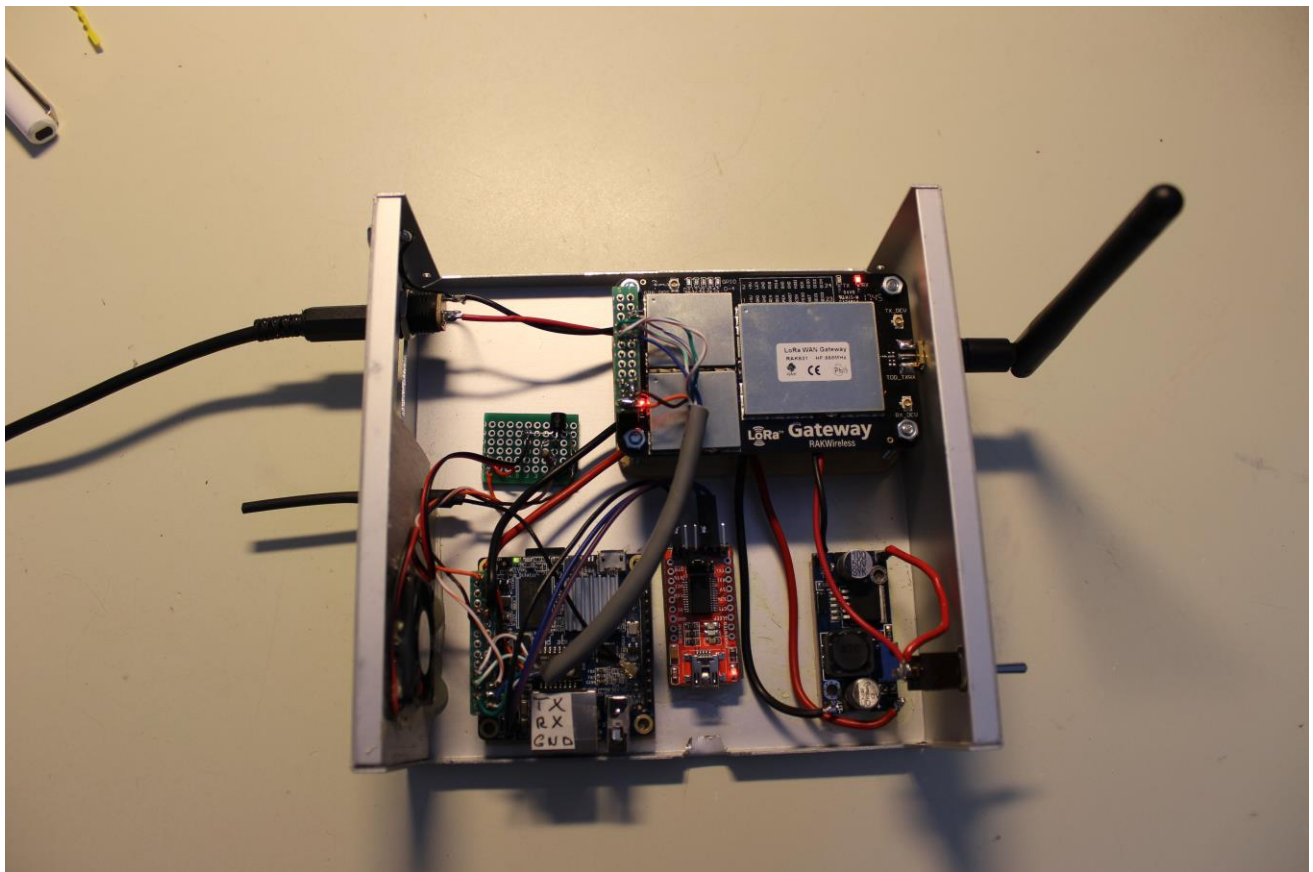


Cable between RAK831 and Orange Pi Zero:



For the connectors I used 2 rows of headers mounted on a small piece of perfboard.

Fully wired:



The red module in the middle is used temporarily to make a connection to the serial port over USB.



## 11.1 Normal boot output.

```
U-Boot SPL 2017.11-armbian (Jan 25 2018 - 08:04:30)
DRAM: 512 MiB
Trying to boot from MMC1
```

```
U-Boot 2017.11-armbian (Jan 25 2018 - 08:04:30 +0100) Allwinner Technology
```

```
CPU:   Allwinner H3 (SUN8I 1680)
Model: Xunlong Orange Pi Zero
DRAM:  512 MiB
MMC:   SUNXI SD/MMC: 0
*** Warning - bad CRC, using default environment

In:    serial
Out:   serial
Err:   serial
Net:   phy interface0
eth0: ethernet@1c30000
starting USB...
USB0:  USB EHCI 1.00
USB1:  USB OHCI 1.0
scanning bus 0 for devices... 1 USB Device(s) found
       scanning usb for storage devices... 0 Storage Device(s) found
Autoboot in 1 seconds, press <Space> to stop
switch to partitions #0, OK
mmc0 is current device
Scanning mmc 0:1...
Found U-Boot script /boot/boot.scr
3708 bytes read in 203 ms (17.6 KiB/s)
## Executing script at 43100000
U-boot loaded from SD
Boot script loaded from mmc
262 bytes read in 164 ms (1000 Bytes/s)
5097894 bytes read in 605 ms (8 MiB/s)
6972808 bytes read in 788 ms (8.4 MiB/s)
Found mainline kernel configuration
31935 bytes read in 1249 ms (24.4 KiB/s)
504 bytes read in 773 ms (0 Bytes/s)
Applying kernel provided DT overlay sun8i-h3-usbhost2.dtbo
504 bytes read in 866 ms (0 Bytes/s)
Applying kernel provided DT overlay sun8i-h3-usbhost3.dtbo
780 bytes read in 853 ms (0 Bytes/s)
Applying kernel provided DT overlay sun8i-h3-spi-spidev.dtbo
4179 bytes read in 879 ms (3.9 KiB/s)
Applying kernel provided DT fixup script (sun8i-h3-fixup.scr)
## Executing script at 44000000
## Loading init Ramdisk from Legacy Image at 43300000 ...
   Image Name:   uInitrd
   Image Type:   ARM Linux RAMDisk Image (gzip compressed)
   Data Size:    5097830 Bytes = 4.9 MiB
   Load Address: 00000000
   Entry Point:  00000000
   Verifying Checksum ... OK
## Flattened Device Tree blob at 43000000
   Booting using the fdt blob at 0x43000000
   Loading Ramdisk to 49b23000, end 49fff966 ... OK
   reserving fdt memory region: addr=43000000 size=6e000
   Loading Device Tree to 49ab2000, end 49b22fff ... OK
```

```
Starting kernel ...
```

```
Loading, please wait...
starting version 232
Begin: Loading essential drivers ... done.
Begin: Running /scripts/init-premount ... done.
Begin: Mounting root file system ... Begin: Running /scripts/local-top ... done.
Begin: Running /scripts/local-premount ... Scanning for Btrfs filesystems
done.
Begin: Will now check root file system ... fsck from util-linux 2.29.2
[/sbin/fsck.ext4 (1) -- /dev/mmcblk0p1] fsck.ext4 -a -C0 /dev/mmcblk0p1
/dev/mmcblk0p1: clean, 37915/911904 files, 363752/3846192 blocks
done.
done.
Begin: Running /scripts/local-bottom ... done.
```

Begin: Running /scripts/init-bottom ... done.

Welcome to Debian GNU/Linux 9 (stretch)!

```
[ OK ] Reached target Remote File Systems.
[ OK ] Listening on /dev/initctl Compatibility Named Pipe.
[ OK ] Listening on Journal Audit Socket.
[ OK ] Created slice User and Session Slice.
[ OK ] Listening on udev Kernel Socket.
[ OK ] Listening on Journal Socket (/dev/log).
[ OK ] Started Dispatch Password Requests to Console Directory Watch.
[ OK ] Set up automount Arbitrary Executable File System Automount Point.
[ OK ] Created slice System Slice.
      Mounting POSIX Message Queue File System...
[ OK ] Created slice system-getty.slice.
      Mounting Debug File System...
[ OK ] Reached target Slices.
[ OK ] Listening on Syslog Socket.
[ OK ] Listening on Journal Socket.
      Starting Set the console keyboard layout...
      Starting Nameserver information manager...
[ OK ] Started Forward Password Requests to Wall Directory Watch.
[ OK ] Reached target Encrypted Volumes.
[ OK ] Reached target Paths.
[ OK ] Created slice system-serial\x2dgetty.slice.
[ OK ] Listening on fsck to fsckd communication Socket.
      Starting Remount Root and Kernel File Systems...
[ OK ] Listening on udev Control Socket.
      Starting Restore / save the current clock...
      Starting Load Kernel Modules...
      Starting Create list of required static device nodes for the current kernel...
[ OK ] Mounted POSIX Message Queue File System.
[ OK ] Mounted Debug File System.
[ OK ] Started Set the console keyboard layout.
[ OK ] Started Remount Root and Kernel File Systems.
[ OK ] Started Restore / save the current clock.
[ OK ] Started Load Kernel Modules.
[ OK ] Started Create list of required static device nodes for the current kernel.
[ OK ] Started Nameserver information manager.
      Starting Create Static Device Nodes in /dev...
      Starting Apply Kernel Variables...
      Mounting Configuration File System...
      Starting Load/Save Random Seed...
      Starting udev Coldplug all Devices...
      Activating swap /var/swap...
[ OK ] Mounted Configuration File System.
[ OK ] Started Create Static Device Nodes in /dev.
[ OK ] Started Apply Kernel Variables.
[ OK ] Started Load/Save Random Seed.
[ OK ] Activated swap /var/swap.
[ OK ] Reached target Swap.
      Starting udev Kernel Device Manager...
[ OK ] Reached target Local File Systems (Pre).
      Mounting /tmp...
[ OK ] Mounted /tmp.
[ OK ] Started udev Kernel Device Manager.
[ OK ] Found device /dev/ttyGS0.
[ OK ] Reached target Local File Systems.
      Starting Set console font and keymap...
      Starting Armbian enhanced Log2Ram...
      Starting Raise network interfaces...
[ OK ] Started udev Coldplug all Devices.
[ OK ] Started Set console font and keymap.
[ OK ] Found device /dev/ttyS0.
[ OK ] Started Armbian enhanced Log2Ram.
[ OK ] Started Raise network interfaces.
[ OK ] Listening on Load/Save RF Kill Switch Status /dev/rfkill Watch.
      Starting Journal Service...
[ OK ] Started ifup for eth0.
      Starting Load/Save RF Kill Switch Status...
[ OK ] Started Load/Save RF Kill Switch Status.
[ OK ] Started Journal Service.
      Starting Flush Journal to Persistent Storage...
[ OK ] Started Flush Journal to Persistent Storage.
      Starting Create Volatile Files and Directories...
[ OK ] Started Create Volatile Files and Directories.
```

```

[ OK ] Reached target System Time Synchronized.
        Starting Update UTMP about System Boot/Shutdown...
[ OK ] Started Entropy daemon using the HAVEGE algorithm.
[ OK ] Started Update UTMP about System Boot/Shutdown.
[ OK ] Reached target System Initialization.
[ OK ] Started Daily apt download activities.
[ OK ] Started Daily apt upgrade and clean activities.
[ OK ] Listening on D-Bus System Message Bus Socket.
[ OK ] Reached target Sockets.
[ OK ] Started Daily Cleanup of Temporary Directories.
[ OK ] Reached target Timers.
[ OK ] Reached target Basic System.
        Starting LSB: Armbian gathering hardware information...
[ OK ] Started Regular background program processing daemon.
        Starting LSB: Load kernel modules needed to enable cpufreq scaling...
[ OK ] Started D-Bus System Message Bus.
        Starting LSB: Start/stop sysstat's sadc...
        Starting Network Manager...
[ OK ] Started The Things Network Gateway.
[ OK ] Started Activate fan on high temperature.
        Starting Login Service...
        Starting System Logging Service...
[ OK ] Started LSB: Start/stop sysstat's sadc.
[ OK ] Started System Logging Service.
[ OK ] Started Login Service.
[ OK ] Started LSB: Load kernel modules needed to enable cpufreq scaling.
        Starting LSB: set CPUFreq kernel parameters...
[ OK ] Started LSB: set CPUFreq kernel parameters.
        Starting LSB: Set sysfs variables from /etc/sysfs.conf...
[ OK ] Started LSB: Set sysfs variables from /etc/sysfs.conf.
[ OK ] Started Network Manager.
        Starting Network Manager Script Dispatcher Service...
        Starting Network Manager Wait Online...
[ OK ] Reached target Network.
        Starting Permit User Sessions...
        Starting OpenBSD Secure Shell server...
[ OK ] Started Unattended Upgrades Shutdown.
[ OK ] Started Permit User Sessions.
[ OK ] Started Network Manager Script Dispatcher Service.
[ OK ] Started OpenBSD Secure Shell server.
        Starting Hostname Service...
[ OK ] Started Hostname Service.
        Starting Authorization Manager...
        Starting WPA supplicant...
[ OK ] Started Authorization Manager.
[ OK ] Started WPA supplicant.
[ OK ] Started LSB: Armbian gathering hardware information.
[ OK ] Started Network Manager Wait Online.
[ OK ] Reached target Network is Online.
        Starting LSB: Start NTP daemon...
        Starting LSB: disk temperature monitoring daemon...
        Starting /etc/rc.local Compatibility...
        Starting LSB: Advanced IEEE 802.11 management daemon...
[ OK ] Started /etc/rc.local Compatibility.
[ OK ] Started LSB: Advanced IEEE 802.11 management daemon.
[ OK ] Started Serial Getty on ttyGS0.
[ OK ] Started Serial Getty on ttyS0.
[ OK ] Started Getty on tty1.
[ OK ] Reached target Login Prompts.
[ OK ] Started LSB: disk temperature monitoring daemon.
[ OK ] Started LSB: Start NTP daemon.
[ OK ] Reached target Multi-User System.
[ OK ] Reached target Graphical Interface.
        Starting Update UTMP about System Runlevel Changes...
[ OK ] Started Update UTMP about System Runlevel Changes.

```

Debian GNU/Linux 9 GwLoRa ttyS0