A high-reliability DIY TTN LoRa gateway using RAK2247 & Raspberry Pi3B+ with PoE and UPS

Steve at MarvellConsultants dot com *21-12-2019*

Description

This LoRa gateway integration was designed for low cost and high reliability on a Powered-Over-Ethernet Raspberry Pi 3B+ with uninterruptable power supply and hardware watchdog. Furthermore it can be remotely located and managed whilst being robust against SD card corrupting power failures events. This is a very Do_It_Yourself, hands-on project but the end result will be as good, if not better, than any of the expensive commercial LoRa gateway products currently available.

Components:

- RAK2247 concentrator module (SPI version)
- Beyond logic Pi HAT board for the above
- Raspberry Pi 3B+ (with PoE)
- PoEUPSPi Uninterruptable PoE power supply & watch-dog board
- 60cm outdoor antenna with low loss down-lead
- Low-cost custom enclosure
- Software

RAK2247 concentrator module

This is the latest addition to RAK Wireless's range of standard Semtech chip-set based LoRa concentrators, its a very neat product with an essential built-in heatsink on its PCI Express mini card format board. I got mine from:

https://store.rakwireless.com

Be sure to select the correct operating frequency for your region and also specify the SPI version. It costs only $95 but took a while to arrive from China. It's well documented at:

https://downloads.rakwireless.com/en/LoRa/RAK2247-Mini-PCIe

There's also a very helpful and well run forum for your hardware & software Q&As

https://forum.rakwireless.com

Beyond Logic Pi HAT

The Beyond Logic web site:

https://www.beyondlogic.org/rak833-lorawan-concentrator

offers a PCB design for a Raspberry Pi HAT that supports a RAK-compatible PCI Express Mini Card form-factor interface, it incorporates a 3.3V switch-mode power supply and a GPS receiver. It was originally designed around RAK's earlier 833 module but is 100% compatible with the newer RAK2247. It's a superb

little board, well documented, easy to source components, straightforward to build and works perfectly. Beyond Logic recommend using HK-based https://JLCPCB.com to manufacture the PCB, a recommendation that I can fully endorse. They are amazingly cheap but the quality and service is excellent, if you're happy to spend a little extra on P&P you can get your boards delivered surprisingly quickly. Just a couple of additional build notes: take resistor values from the schematic in preference to the BOM, omit resistors R1 & R5, link R6. Note that it requires an active GPS antenna, any of the ones on sale at Amazon/Ebay etc should be fine. For GPS you'll probably need a UFL to SMA pigtail (which might come bundled with the GPS antenna) plus a UFL pigtail for the LoRa antenna - I'd recommend N-type, see below. Be sure not to power up your concentrator module without an antenna attached.

PoEUPSPi supercap-based UPS PoE power supply & watch-dog board

This is a somewhat more demanding build, all the resources you should need are at:

https://www.elektormagazine.com/labs/poe-ups-pi

I designed this board especially for the LoRa gateway project but it could well find many uses elsewhere. This is the component that makes the gateway robust, reliable and dependable. With this board included there is no prospect of a power-outage trashing the Pi's SD card - an issue I've been plagued with over the years when trying to embed Raspberry Pis. Basically, in the event of power failure the board provides a 30 second hold-up time : 15 seconds of run-through during which if power returns the system just carries on un-scathed. Then 15 seconds in which to cleanly shut down the Pi before the inevitable power-out. Once power returns the Pi's supply is held off while the supercaps fully re-charge (about 20s) at which point the Pi is cleanly powered-up. The watchdog function provides an additional line of defence, automatically rebooting the Pi should it for some reason crash or stop working for more than 60 seconds. Note that PoEUPSPi is compatible only with model 3B+ Raspberry Pis - they're the only ones with a PoE header. It might work, with minor modifications, with Pi4s but this is untested.

While this part of the system is optional I would definitely recommend the extra work and expense. Also PoE is the ideal way to power any such system.

60cm outdoor antenna

The antenna is a vitally important part of any RF installation. A large antenna can provide up to 6dB of valuable extra gain and should definitely be mounted high up and outside for best coverage. Don't skimp on the antenna or the down-lead. Most outdoor antennas come with an N-type connector and ready-made N-type to N-type low-loss down-leads are widely available, so I'd recommend an N-type connector for the system, I got the requisite UFL-to-N-type bulkhead pigtail from:

https://www.amazon.co.uk/Ochoos-Bulkhead-Pigtail-Wholesale-Connector/dp/B07Q64MY42

It's now out of stock, FYI it took some time to arrive from China but was a good quality product.

Low-cost custom enclosure

I opted for an indoor enclosure that therefore doesn't need IP65 waterproofing and which can benefit from convection cooling in a benign environment, an outdoor enclosure option is quite possible but comes with extra challenges - the system dissipates about 4W so thermal management needs to be considered. As PCBs from JLCPCB.com are so cheap I opted to construct the skeleton of my enclosure from a set of 3 custom PCBs, a carrier board and two identical end pieces. The RPi bolts directly to the carrier, the PoEUPSPi sits on top of the RPi with the RAK concentrator board atop that. The end pieces bolt to either end of the carrier and a variety of holes provide mounting points for connectors, ventilation and room for an optional 30mm fan - I found in practice the fan was unnecessary. This little assembly then slides into a short section of 60mm square-section plastic drain-pipe , available from all good DIY stores, and screws into place with two M4 bolts. The gerber file-sets for carrier and carrier-end PCBs are available to download from my web site, where you'll also find some photos:

http://www.marvellconsultants.com/LoRa

Mechanical bits and pieces come from Toby electronics, the four right-angle M4 screw-mounts are part No PCB-2-M4, while the flat M4 screw-mounts are part No PCB-9-M4

https://www.toby.co.uk

Software

From RAK's downloads site (above) I simply followed their getting started instructions, downloading the RPi-firmware disk image and writing it to an 8GB SD card (using Win32diskimager) from which the RPi duly booted. If you're already up and running with a PoEUPSPi board then you'll know you need a service script running to (a) keep the watch-dog alive and (b) monitor for shutdown requests. While setting up the RAK software & without a service script in place, be sure to disable the PoEUPSPi watchdog function by linking pins 5 & 3 on J4. Continue with RAK's getting started guide to configure the gateway, open an account with The Things Network, and get your gateway registered and linked up to TTN. I found it all fairly straightforward. Ideally you'll have a LoRa device handy and a TTN application set up for it so that it can connect and transmit data through your new gateway and you can see it arriving on your TTN console. One little wrinkle I found, that had me scratching my head for while, was that the RAK board did not seem to be getting a suitable reset pulse from the driver software, I added a few lines to PoEUPSPi.sh to explicitly generate a RAK reset pulse on GPIO25 at boot-time which sorted out the problem. PoEUPSPi.sh should now look something like this:

```
#! /bin/bash
# Support script for PoEUPSPi shut-down management
# and optionally watch-dog pulse generation.
# Also with RAK reset pulse generation on GPIO25
# This script must run with root privileges and start
# automatically at boot time.
#  Eg use sudo crontab -e to add
#   @reboot /home/pi/PoEUPSPi.sh
#  to root's crontab.
# Remember to make the script executable.
# If you need to generate watch-dog kicks on GPIO8 for
# PoEUPSPi hardware here then set WDOGRQ=1, if not WDOGRQ=0
```

```
# If you want to use PoEUPSPi h/w without the watchdog function
# then it should be disabled by wiring J4pin5 to J4pin3
WDOGRQ=0
# Optionally turn HDMI off to save power
#/opt/vc/bin/tvservice -o
# Setup GPIO8 as an output if requested
if [ $WDOGRQ -gt 0 ] ; then
  echo 8   > /sys/class/gpio/export
  echo out > /sys/class/gpio/gpio8/direction
fi
# Setup GPIO25 as an output and send an active high pulse
#  to reset any attached RAK concentrator module
echo 25  > /sys/class/gpio/export
echo out > /sys/class/gpio/gpio25/direction
echo 1   > /sys/class/gpio/gpio25/value
echo 0   > /sys/class/gpio/gpio25/value
# Setup for h/w shut-down request detection on GPIO4
echo 4  > /sys/class/gpio/export
echo in > /sys/class/gpio/gpio4/direction
# Allow boot to complete
sleep 10
# Loop for shutdown requests from GPIO4
# and optionally watch-dog pulse generation on GPIO8.
# Initialise loop variables...
R0=2; R1=2; R2=2;
X=0;
TOGGLE=0
# loop forever...
while [ $X -eq 0 ]; do
  # loop slowly
  sleep 0.5
  # If requested, toggle GPIO8 to kick PoEUPSPi's watchdog
  if [ $WDOGRQ -gt 0 ] ; then
    echo $TOGGLE > /sys/class/gpio/gpio8/value
    if [ $TOGGLE -gt 0 ] ; then TOGGLE=0 ; else TOGGLE=1; fi
  fi
  # Maintain 3 consecutive reads of our GPIO bit,
  # look for high-low-low sequence
  R0=$R1; R1=$R2
  R2=$(cat /sys/class/gpio/gpio4/value)
  [ $R0$R1$R2 -eq 100 ] && X=1
done
sudo shutdown now
```

Note that watchdog pulse generation is turned off in the above script as watchdog pulses now come direct from the RAK driver software (CE0 on SPI0). You can now remove the link on PoEUPSPi J4 to enable its watchdog function. I assume by now you're running headless with SSH, if not Google it.

Some essential tweeks

You should now have a working TTN gateway, but to help with long-term robustness we should minimise the number of writes made to the SD card. The chief culprit in this area is logging - the RAK gateway software in particular is VERY verbose with its logs, at this stage in my set-up it was generating 6MB of log data per hour to both daemon.log and syslog. Most of this being errors and warnings relating to unconfigured extraneous software packages not required with a TTN-based gateway. These can easily be stopped, once and for all with these command lines:

```
sudo systemctl disable loraserver
sudo systemctl disable lora-app-server
sudo systemctl disable lora-gateway-bridge
sudo systemctl disable postgresql
sudo systemctl disable mosquitto
sudo systemctl disable redis
```

Next are a load of error lines to do with the unused WiFi AP that the software tries to set up, this can be disabled using: `sudo gateway-config` -> (5) Configure WiFi -> (2) Enable client mode/Disable AP

If you're not using GPS, but haven't configured for no GPS this will also generate copious errors.

OK, so hopefully you've been able to fix all those error messages in daemon.log but that still leaves roughly 3500 lines/hour of 'normal' log activity which I've found no way of further reducing. The next step is to re-mount all of Rasbian's highly volatile directories to tmpfs ramdisk. This is easily done with an edit to /etc/fstab - simply add the following three lines:

```
tmpfs /tmp tmpfs defaults,noatime,nosuid,size=100m 0 0
tmpfs /var/tmp tmpfs defaults,noatime,nosuid,size=30m 0 0
tmpfs /var/log tmpfs defaults,noatime,nosuid,mode=0755,size=200m 0 0
```

This all massively improves long-term SD card write wear but we now still have the problem of large log files building up in our small ramdisk mounts. I added a few extra lines to PoEUPSPi.sh's while-done loop to truncate syslog to a manageable size and delete daemon.log once a day:

```
date | grep 04:00:00 > /dev/null
if [ $? -eq 0 ] ; then
  sudo bash -c 'echo "$(tail -1000 /var/log/syslog)" > /var/log/syslog'
  sudo rm /var/log/daemon.log
fi
```

Finally, to save a few milliwatts of power and extend slightly the UPS hold-up time it's possible to turn off some of the RPi's unused peripherals, HDMI is already dealt with in PoEUPSPi.sh, but we can do more: WiFi and bluetooth can be disabled with a one-off edit to /etc/modprobe.d/raspi-blacklist.conf, add the lines:

```
blacklist brcmfmac
blacklist brcmutil
```

Likewise disable sound by editing /etc/modprobe.d/alsa-blacklist.conf to add:

```
blacklist snd_bcm2835
```

Summary

I hope this might inspire you to build your own TTN gateway and expand this fantastic free resource to ever greater coverage. So maybe next time my dog goes missing his LoRa tracker will send GPS data back via your gateway! Good luck and enjoy.