

LoRa GPS HAT Single Channel LoRa & GPS module User Manual

Document Version: 1.0

Version	Description	Date
1.0	Release	2019-Mar-6

1. Introduction	3
1.1 What is LoRa GPS HAT	3
1.2 Specifications	4
1.3 Features	5
1.4 Applications	5
1.5 Pin Definition	6
1.6 Hardware Change log	7
1.7 LEDs	8
1.8 Dimension & Weight	8
2. Example 1: Set up as a Single Channel LoRaWAN gateway with Raspbian OS	9
2.1 Configure LoRa GPS HAT	9
2.1.1 Install packet forwarder software	9
2.1.2 Configure Frequency and server	11
2.2 Create a gateway in TTN Server	12
3. Example2: Set up as a Single Channel LoRaWAN gateway and connect to TTN with RPI and Windows 10 IOT core	14
4. Example3: Two RPI use LoRa to transmit to each other	15
5. FAQ	18
5.1 Why there is 433/868/915 version?	18
5.2 What is the frequency range of LoRa GPS HAT part?	18
6. Order Info	19
7. Packing Info	19
8. Support	19

1. Introduction

1.1 What is LoRa GPS HAT

LoRa GPS HAT is an expansion module for LoRaWAN and GPS for use with the **Raspberry Pi**. This product is intended for those interested in developing LoRaWAN solutions.

LoRa GPS HAT is **based on the SX1276/SX1278** transceiver. The add on **L80 GPS (Base on MTK MT3339)** is designed for applications that use a GPS connected via the serial ports to the Raspberry Pi such as timing applications or general applications that require GPS information.

The transceivers of the HAT feature the **LoRa™ long range modem** that provides ultra-long range spread spectrum communication and high interference immunity whilst minimizing current consumption. The LoRa/GPS HAT can achieve a sensitivity of over -148dBm using a low cost crystal and bill of materials. The high sensitivity combined with the integrated +20 dBm power amplifier yields industry leading link budget making it optimal for any application requiring range or robustness. LoRa™ also provides significant advantages in both blocking and selectivity over conventional modulation techniques, solving the traditional design compromise between range, interference immunity and energy consumption.

The L80 GPS module can calculate and predict orbits automatically using the ephemeris data (up to 3 days) stored in internal flash memory, so the HAT can fix position quickly even at indoor signal levels with low power consumption. With AlwaysLocate™ technology, the Lora/GPS HAT can adaptively adjust the on/off time to achieve balance between positioning accuracy and power consumption according to the environmental and motion conditions. The GPS also supports **automatic antenna switching function**. It can achieve the switching between **internal patch antenna and external active antenna**. Moreover, it keeps positioning during the switching process.

1.2 Specifications

LoRa Spec

- 168 dB maximum link budget.
- +20 dBm - 100 mW constant RF output vs.
- +14 dBm high efficiency PA.
- Programmable bit rate up to 300 kbps.
- High sensitivity: down to -148 dBm.
- Bullet-proof front end: IIP3 = -12.5 dBm.
- Excellent blocking immunity.
- Low RX current of 10.3 mA, 200 nA register retention.
- Fully integrated synthesizer with a resolution of 61 Hz.
- FSK, GFSK, MSK, GMSK, LoRaTM and OOK modulation.
- Built-in bit synchronizer for clock recovery.
- Preamble detection.
- 127 dB Dynamic Range RSSI.
- Automatic RF Sense and CAD with ultra-fast AFC.
- Packet engine up to 256 bytes with CRC.
- Built-in temperature sensor and low battery indicator.

GPS Spec

- Based on MT3339.
- Power Acquisition:25mA,Power Tracking:20mA.
- Compliant with GPS, SBAS.
- Programmable bit rate up to 300 kbps.
- Serial Interfaces UART: Adjustable 4800~115200 bps,Default: 9600bps.
- Update rate:1Hz (Default), up to10Hz.
- I/O Voltage:2.7V ~ 2.9V.
- Protocols:NMEA 0183,PMTK.
- Horizontal Position Accuracy:Autonomous <2.5 m CEP.
- TTFF@-130dBm with EASY™:Cold Start <15s,Warm Start <5s,Hot start <1s;TTFF@-130dBm.without EASY™:Cold Start <35s,Warm Start <30s,Hot Start <1s.
- Timing Accuracy:1PPS out 10ns, Reacquisition Time <1s.
- Velocity Accuracy Without aid <0.1m/s,Acceleration Accuracy Without aid 0.1m/s².
- Sensitivity Acquisition -148dBm, Tracking -165dBm, Reacquisition -160dBm.
- Environmental:Operating Temperature -40°C to 85°C,Storage Temperature -45°C to 125°C.
- Dynamic Performance Altitude Max.18000m, Maximum Velocity Max.515m/s, Maximum Acceleration 4G.
- L1 Band Receiver(1575.42MHz) Channel 22 (Tracking) /66 (Acquisition).

1.3 Features

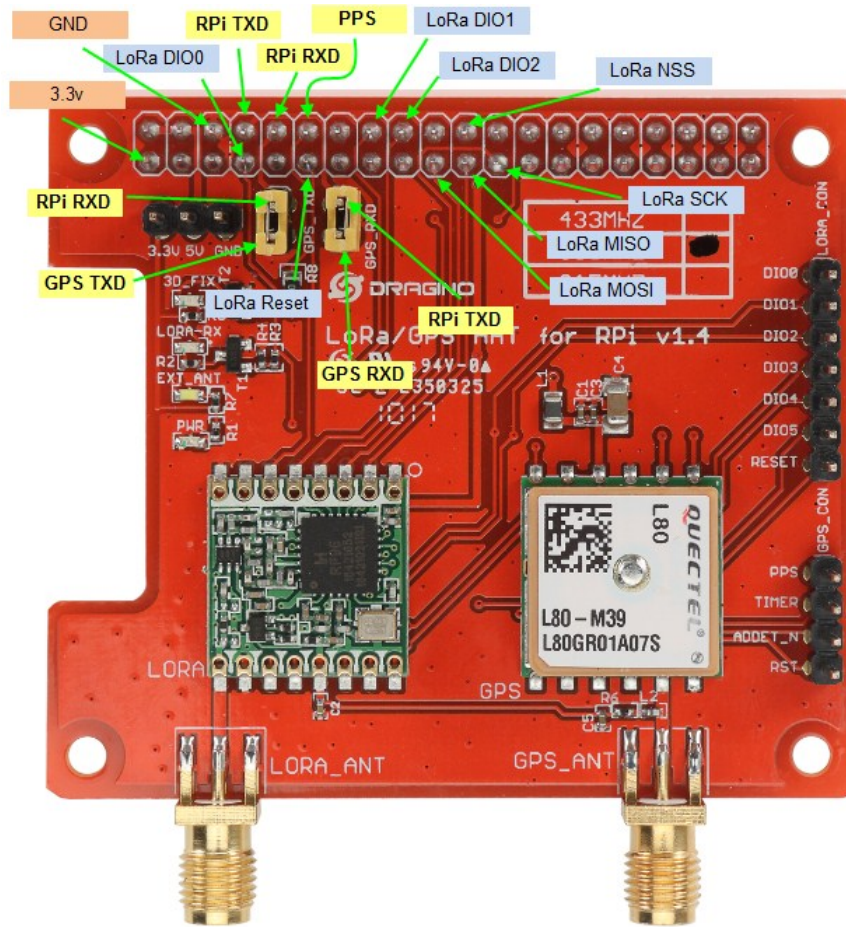
- ✓ Frequency Band: 868 MHZ/433 MHZ/915 MHZ(Pre-configure in factory)
- ✓ Low power consumption
- ✓ Compatible with Raspberry Pi 2 Model B/Raspberry Pi 3 model B/B+
- ✓ LoRa™ Modem
- ✓ FSK, GFSK, MSK, GMSK, LoRa™and OOK modulation
- ✓ Preamble detection
- ✓ Baud rate configurable
- ✓ Built-in temperature sensor and low battery indicator
- ✓ Excellent blocking immunity
- ✓ Automatic RF Sense and CAD with ultra-fast AFC
- ✓ Support DGPS, SBAS(WAAS/EGNOS/MSAS/GAGAN)
- ✓ GPS automatic switching between internal patch antenna and external active antenna
- ✓ PPS VS. NMEA can be used in time service
- ✓ Support SDK command
- ✓ Built-in LNA for better sensitivity
- ✓ EASY™, advanced AGPS technology without external memory
- ✓ AlwaysLocate™, an intelligent controller of periodic mode
- ✓ GPS FLP mode, about 50% power consumption of normal mode
- ✓ GPS support short circuit protection and antenna detection

1.4 Applications

- ✓ Smart Buildings & Home Automation
- ✓ Logistics and Supply Chain Management
- ✓ Smart Metering
- ✓ Smart Agriculture
- ✓ Smart Cities
- ✓ Smart Factory

1.5 Pin Definition

Pin Illustrator:



Pin Mapping

LoRa GPS HAT	RaspberryPi Wiring PI IO
3.3v	3.3v
5v	5v
GND	GND
DIO0	GPIO7
GPS_RX	GPIO15/TX
GPS_TX	GPIO16/RX
RESET	GPIO0
LoRa_NSS	GPIO6
LoRa_MISO	GPIO13/MISO
LoRa_MOSI	GPIO12/MOSI

SCK	GPIO14/SCLK
DIO1	GPIO4
DIO2	GPIO5
1PPS	GPIO1

BCM 编码方式	wPi 编码方式	功能名	物理接口				功能名	wPi 编码方式	BCM 编码方式	
BCM	wPi	Name	Mode	V	B Plus Physical	V	Mode	Name	wPi	BCM
		3.3v			1 2			5v		
2	8	SDA.1	ALTO	1	3 4			5V		
3	9	SCL.1	ALTO	1	5 6			0v		
4	7	GPIO. 7	IN	1	7 8	0	ALTO	TxD	15	14
		0v			9 10	1	ALTO	RxD	16	15
17	0	GPIO. 0	IN	0	11 12	0	IN	GPIO. 1	1	18
27	2	GPIO. 2	IN	0	13 14			0v		
22	3	GPIO. 3	IN	0	15 16	0	IN	GPIO. 4	4	23
		3.3v			17 18	1	OUT	GPIO. 5	5	24
10	12	MOSI	ALTO	0	19 20			0v		
9	13	MISO	ALTO	1	21 22	1	OUT	GPIO. 6	6	25
11	14	SCLK	ALTO	1	23 24	1	ALTO	CE0	10	8
		0v			25 26	1	ALTO	CE1	11	7
0	30	SDA.0	ALTO	1	27 28	1	ALTO	SCL.0	31	1
5	21	GPIO.21	IN	1	29 30			0v		
6	22	GPIO.22	IN	1	31 32	0	IN	GPIO.26	26	12
13	23	GPIO.23	IN	0	33 34			0v		
19	24	GPIO.24	IN	0	35 36	0	IN	GPIO.27	27	16
26	25	GPIO.25	IN	0	37 38	0	IN	GPIO.28	28	20
		0v			39 40	0	IN	GPIO.29	29	21

1.6 Hardware Change log

- LoRa/GPS_HAT v1.0: The first hardware release for the LoRa/GPS_HAT.
- LoRa/GPS_HAT v1.3:
 - ✓ Add a trace from LoRa DIO1 to RPi GPIO4(wiringPi definition).
 - ✓ Add a trace from Lora DIO2 to RPi GPIO5(wiringPi definition). They are required by LMIC library in RPi.
- LoRa/GPS HAT v1.4:
 - ✓ Change SMA connector to support active antenna
 - ✓ Add AADET_N LED to show if external antenna is active.
 - ✓ Connect GPS PPS pin to RPi BCM pin 18
 - ✓ Modify Silkscreen for GPS TXD/RXD

1.7 LEDs

- ✓ **PWR:** Power Indicate LED. Turns on once there is power.
- ✓ **LoRa-RX:** Indicate there is a wireless packet received in the LoRa module.
- ✓ **3D_FIX:** The led blink every 100ms after the GPS fixing position.
- ✓ **EXT_ANT:** Indicate there is an external GPS antenna connected.

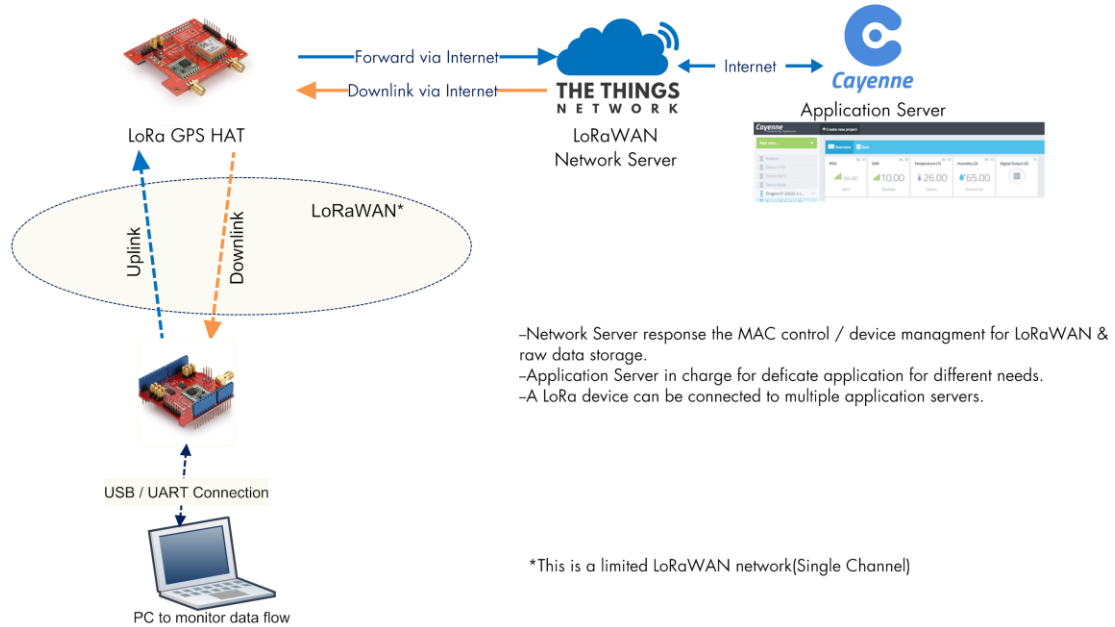
1.8 Dimension & Weight

- ✓ **Size:** 60mm*53mm*25mm
- ✓ **Net weight:** 30g.
- ✓ **Package Size:** 98mm x 81mm x 32mm

2. Example 1: Set up as a Single Channel LoRaWAN gateway with Raspbian OS

The network topology and dataflow for the example is as below:

Topology for Thethingsnetwork Connection:



2.1 Configure LoRa GPS HAT

2.1.1 Install packet forwarder software

a) Installation command git

```
sudo apt-get install git
```

b) SPI needs to be enabled on the Raspberry Pi

Run `sudo raspi-config` to open the config window

```

┌──────────┴──────────┐ Raspberry Pi software Configuration Tool (raspi-config)
1 Change User Password Change password for the current user
2 Network Options      Configure network settings
3 Boot Options         Configure options for start-up
4 Localisation options Set up language and regional settings to match your location
5 Interfacing Options Configure connections to peripherals
6 Overclock            Configure overclocking for your Pi
7 Advanced Options     Configure advanced settings
8 Update               Update this tool to the latest version
9 About raspi-config   Information about this configuration tool

<Select>                                     <Finish>

```

```

_____ | Raspberry Pi software Configuration Tool (raspi-config)
|
P1 Camera      Enable/Disable connection to the Raspberry Pi Camera
P2 SSH         Enable/Disable remote command line access to your Pi using SSH
P3 VNC         Enable/Disable graphical remote access to your Pi using RealVNC
P4 SPI         Enable/Disable automatic loading of SPI kernel module
P5 I2C         Enable/Disable automatic loading of I2C kernel module
P6 Serial      Enable/Disable shell and kernel messages on the serial connection
P7 1-wire      Enable/Disable one-wire interface
P8 Remote GPIO Enable/Disable remote access to GPIO pins

                                <Select>                                <Back>

```

c) Installation wiringpi (GPIO access library)

```
sudo apt-get install wiringpi
```

d) Install packet_forwarder

```
cd /home/pi
git clone https://github.com/dragino/dual_chan_pkt_fwd
make
```

After installation, run

```
sudo ./dual_chan_pkt_fwd
```

to start the packet forwarder and check the gateway ID.

```

pi@raspberrypi:~/dual_chan_pkt_fwd$ sudo ./dual_chan_pkt_fwd
server: .address = router.eu.staging.thethings.network; .port = 1700; .enable = 1
server: .address = router.eu.thethings.network; .port = 1700; .enable = 0
Gateway Configuration
  your name (a@b.c)
  Dual channel pkt forwarder
  Latitude=0.00000000
  Longitude=0.00000000
  Altitude=10
  Interface: eth0
Trying to detect module CE0 with NSS=6 DIO0=7 Reset=3 Led1=unused
SX1276 detected on CE0, starting.
Trying to detect module CE1 with NSS=6 DIO0=7 Reset=3 Led1=unused
SX1276 detected on CE1, starting.
Gateway ID: b8:27:eb:ff:78:3b:7f
Listening at SF7 on 868.100000 Mhz.
Listening at SF7 on 868.100000 Mhz.
-----
stat update: 2019-03-08 01:18:58 GMT no packet received yet
stat update: 2019-03-08 01:19:28 GMT no packet received yet
stat update: 2019-03-08 01:19:58 GMT no packet received yet
stat update: 2019-03-08 01:20:28 GMT no packet received yet

```

The gateway ID is **b827ebffff783b7f**

Use ctrl C to stop it. and install it as a system service.

```
sudo make install
```

To start service, as root or sudo (should already be started at boot if you done make install and rebooted of course), stop service or look service status

```
systemctl start dual_chan_pkt_fwd
systemctl stop dual_chan_pkt_fwd
systemctl status dual_chan_pkt_fwd
```

To see gateway log in real time

```
sudo journalctl -f -u dual_chan_pkt_fwd
```

2.1.2 Configure Frequency and server

For single channel gateway, we just need to set one frequency. The file is global_conf.json.

```
pi@raspberrypi:~/dual_chan_pkt_fwd$ cat global_conf.json
{
  "sx127x_conf":
  {
    "freq": 868100000,
    "freq_2": 868100000,
    "spread_factor": 7,
    "pin_nss": 6,
    "pin_dio0": 7,
    "pin_nss_2": 6,
    "pin_dio0_2": 7,
    "pin_rst": 3,
    "pin_led1": 4,
    "pin_NetworkLED": 22,
    "pin_InternetLED": 23,
    "pin_ActivityLED_0": 21,
    "pin_ActivityLED_1": 29
  },
  "gateway_conf":
  {
    "ref_latitude": 0.0,
    "ref_longitude": 0.0,
    "ref_altitude": 10,

    "name": "your name",
    "email": "a@b.c",
    "desc": "Dual channel pkt forwarder",

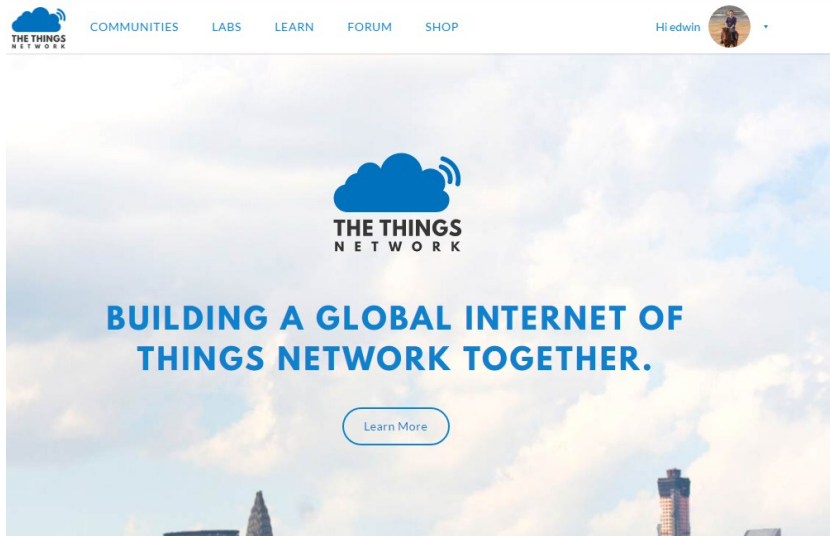
    "interface": "eth0",

    "servers":
    [
      {
        "address": "router.eu.staging.thethings.network",
        "port": 1700,
        "enabled": true
      },
      {
        "address": "router.eu.thethings.network",
        "port": 1700,
        "enabled": false
      }
    ]
  }
}
```

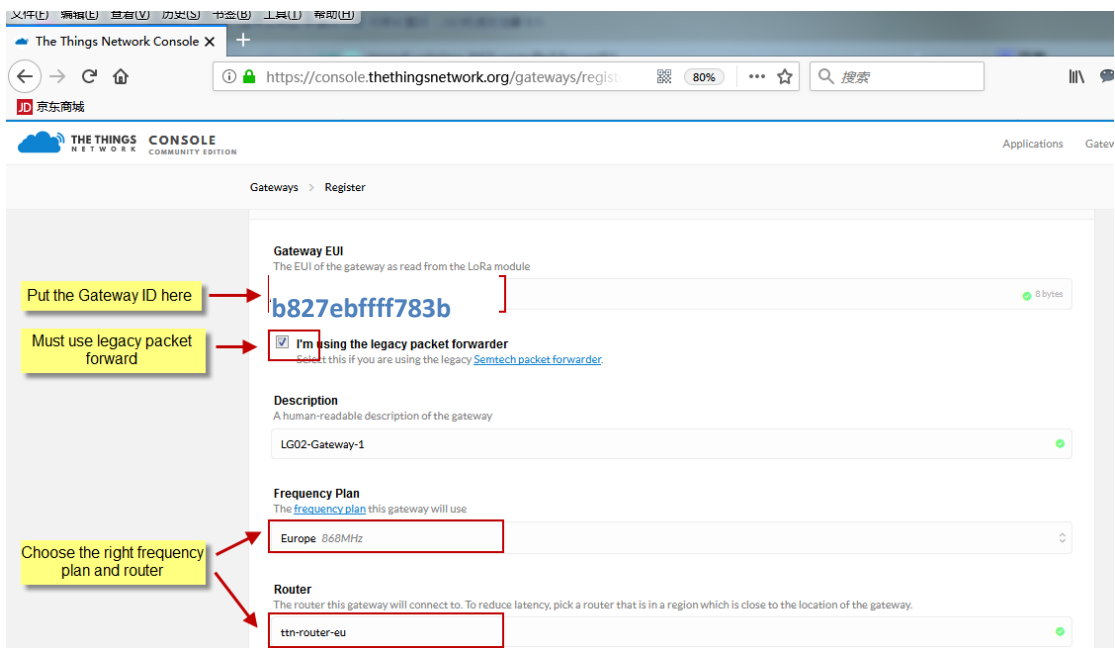
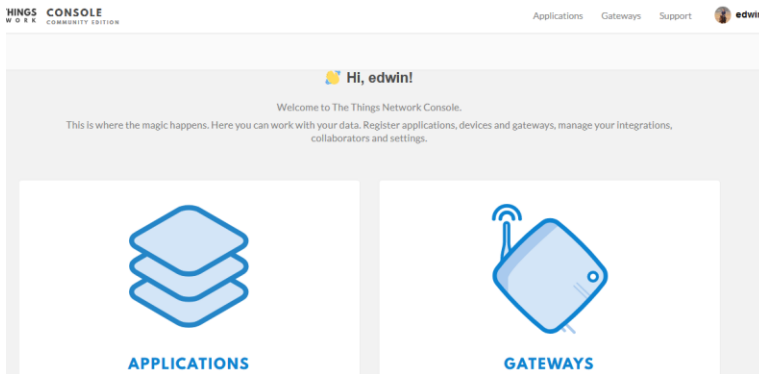
Stop/start the service after make the change.

2.2 Create a gateway in TTN Server

Step 1: Sign up a user account in TTN server



Step 2: Create a Gateway in TTN




After create the gateway, we can see the gateway info, as below, the **Status** shows “connected” because the LoRa GPS HAT is updating its status to server now.

GATEWAY OVERVIEW ⚙️ [setting](#)

Gateway ID eui-b827ebffff783b7f

Description RPi_model

Owner  **edwin** [Transfer ownership](#)

Status ● connected

Frequency Plan Europe 868MHz

Router ttn-router-eu

Gateway Key

Last Seen 3 seconds ago

3. Example2: Set up as a Single Channel LoRaWAN gateway and connect to TTN with RPI and Windows 10 IOT core

[This example](#) from [Mattias Larsson](#) describes how to Build your own LoRaWAN "The Things Network" packet-forwarding Gateway on Windows 10 IoT Core in native .NET code.

4. Example3: Two RPI use LoRa to transmit to each other

Download the lora transmit code :

```
pi@raspberrypi:~$ wget
```

```
https://codeload.github.com/dragino/rpi-lora-tranceiver/zip/master
```

Unzip and Install it:

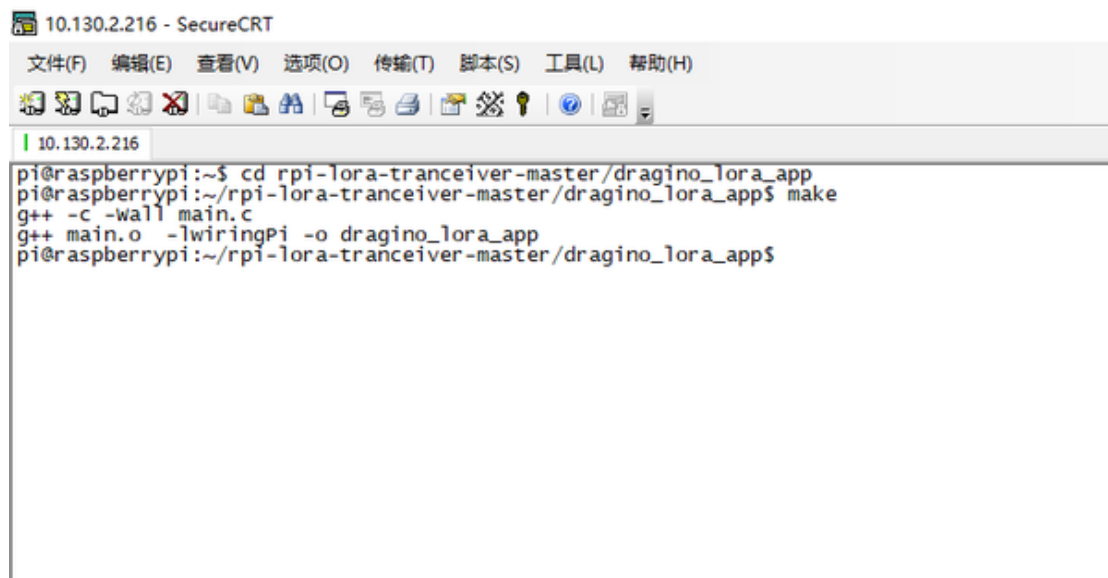


```

10.130.2.216 - SecureCRT
文件(F) 编辑(E) 查看(V) 选项(O) 传输(T) 脚本(S) 工具(L) 帮助(H)
10.130.2.216
pi@raspberrypi:~$ wget https://codeload.github.com/dragino/rpi-lora-tranceiver/zip/master
--2018-05-11 03:39:59-- https://codeload.github.com/dragino/rpi-lora-tranceiver/zip/master
Resolving codeload.github.com (codeload.github.com)... 13.229.189.0, 13.250.162.133, 54.251.140.56
Connecting to codeload.github.com (codeload.github.com)|13.229.189.0|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/zip]
Saving to: 欵樸aster欵?
master [ <=> ] 9.70K --.-KB/s in 0.002s
2018-05-11 03:40:00 (3.92 MB/s) - 欵樸aster欵?saved [9936]
pi@raspberrypi:~$ unzip master
unzip the zip.

```

Make



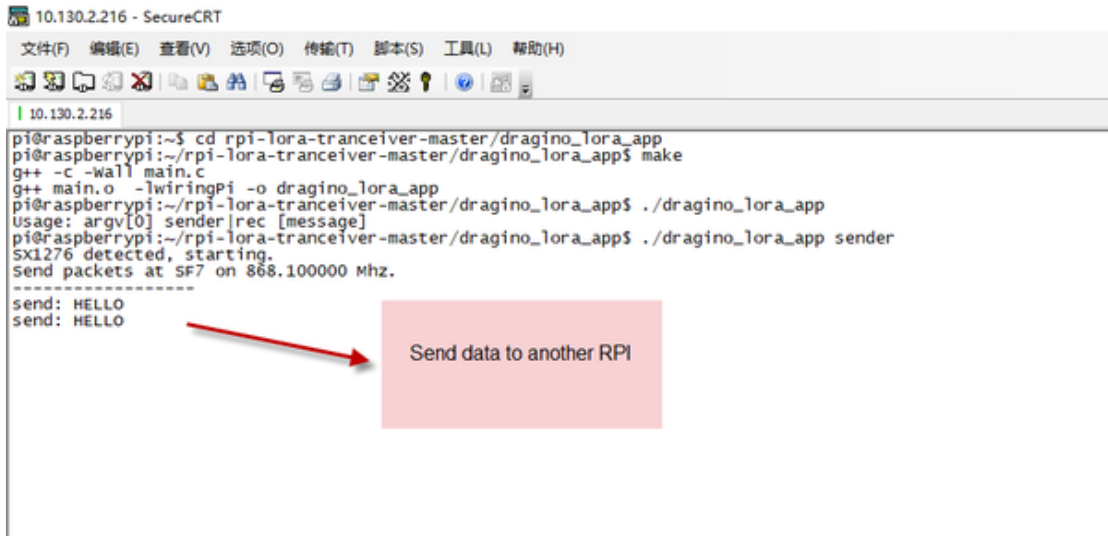
```

10.130.2.216 - SecureCRT
文件(F) 编辑(E) 查看(V) 选项(O) 传输(T) 脚本(S) 工具(L) 帮助(H)
10.130.2.216
pi@raspberrypi:~$ cd rpi-lora-tranceiver-master/dragino_lora_app
pi@raspberrypi:~/rpi-lora-tranceiver-master/dragino_lora_app$ make
g++ -c -Wall main.c
g++ main.o -lwiringPi -o dragino_lora_app
pi@raspberrypi:~/rpi-lora-tranceiver-master/dragino_lora_app$

```

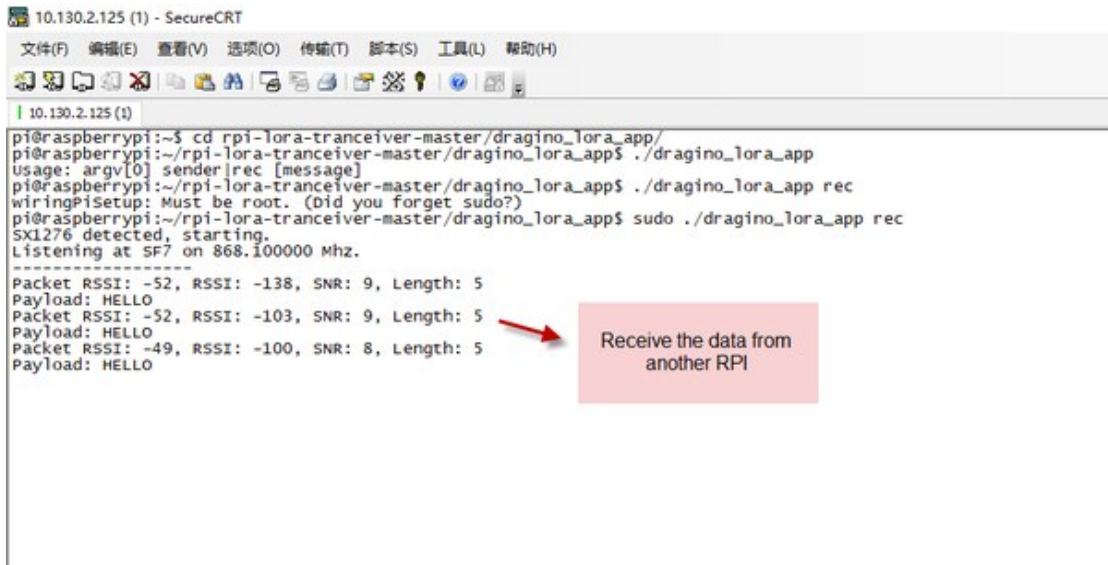
Send data, runs:

```
pi@raspberrypi:~/rpi-lora-tranceiver-master/dragino_lora_app$ ./dragino_lora_app sender
```



```
10.130.2.216 - SecureCRT
文件(F) 编辑(E) 查看(V) 选项(O) 传输(T) 脚本(S) 工具(L) 帮助(H)
10.130.2.216
pi@raspberrypi:~$ cd rpi-lora-tranceiver-master/dragino_lora_app
pi@raspberrypi:~/rpi-lora-tranceiver-master/dragino_lora_app$ make
g++ -c -Wall main.c
g++ main.o -lwiringPi -o dragino_lora_app
pi@raspberrypi:~/rpi-lora-tranceiver-master/dragino_lora_app$ ./dragino_lora_app
Usage: argv[0] sender|rec [message]
pi@raspberrypi:~/rpi-lora-tranceiver-master/dragino_lora_app$ ./dragino_lora_app sender
SX1276 detected, starting.
Send packets at SF7 on 868.100000 Mhz.
-----
send: HELLO
send: HELLO
```

Receive in another RPI:



```
10.130.2.125 (1) - SecureCRT
文件(F) 编辑(E) 查看(V) 选项(O) 传输(T) 脚本(S) 工具(L) 帮助(H)
10.130.2.125 (1)
pi@raspberrypi:~$ cd rpi-lora-tranceiver-master/dragino_lora_app/
pi@raspberrypi:~/rpi-lora-tranceiver-master/dragino_lora_app$ ./dragino_lora_app
Usage: argv[0] sender|rec [message]
pi@raspberrypi:~/rpi-lora-tranceiver-master/dragino_lora_app$ ./dragino_lora_app rec
wiringPiSetup: Must be root. (Did you forget sudo?)
pi@raspberrypi:~/rpi-lora-tranceiver-master/dragino_lora_app$ sudo ./dragino_lora_app rec
SX1276 detected, starting.
Listening at SF7 on 868.100000 Mhz.
-----
Packet RSSI: -52, RSSI: -138, SNR: 9, Length: 5
Payload: HELLO
Packet RSSI: -52, RSSI: -103, SNR: 9, Length: 5
Payload: HELLO
Packet RSSI: -49, RSSI: -100, SNR: 8, Length: 5
Payload: HELLO
```

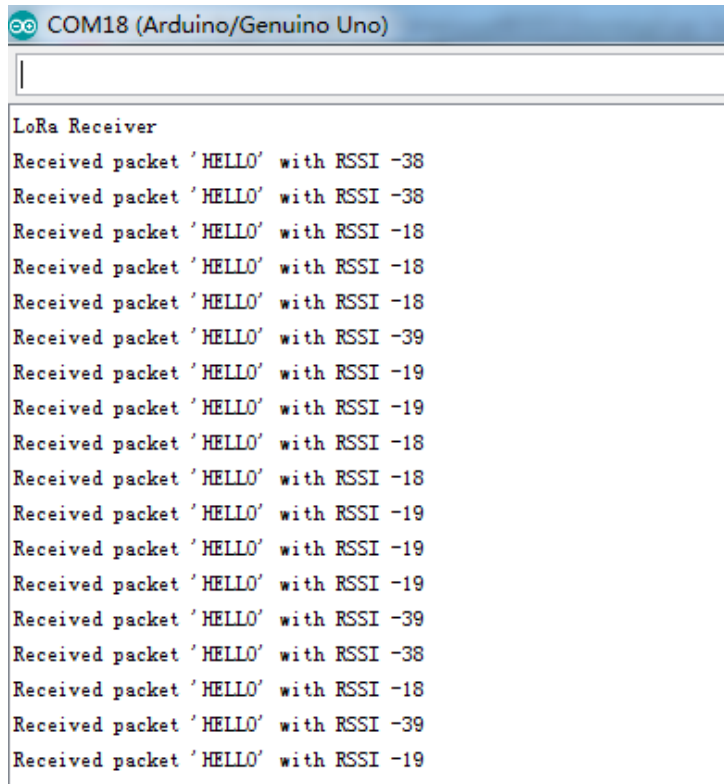
The data can also be received by Arduino: use the library from <https://github.com/sandeepmistry/arduino-LoRa> and below code:

```
#include <SPI.h>
#include <LoRa.h>
void setup() {
  Serial.begin(9600);
```



```
while (!Serial);  
Serial.println("LoRa Receiver");  
if (!LoRa.begin(868100000)) {  
  Serial.println("Starting LoRa failed!");  
  while (1);  
}  
LoRa.setSpreadingFactor(7);  
}
```

```
void loop() {  
  // try to parse packet  
  int packetSize = LoRa.parsePacket();  
  if (packetSize) {  
    // received a packet  
    Serial.print("Received packet ");  
    // read packet  
    while (LoRa.available()) {  
      Serial.print((char)LoRa.read());  
    }  
    // print RSSI of packet  
    Serial.print(" with RSSI ");  
    Serial.println(LoRa.packetRssi());  
  }  
}
```



```
COM18 (Arduino/Genuino Uno)  
|  
LoRa Receiver  
Received packet 'HELLO' with RSSI -38  
Received packet 'HELLO' with RSSI -38  
Received packet 'HELLO' with RSSI -18  
Received packet 'HELLO' with RSSI -18  
Received packet 'HELLO' with RSSI -18  
Received packet 'HELLO' with RSSI -39  
Received packet 'HELLO' with RSSI -19  
Received packet 'HELLO' with RSSI -19  
Received packet 'HELLO' with RSSI -18  
Received packet 'HELLO' with RSSI -18  
Received packet 'HELLO' with RSSI -19  
Received packet 'HELLO' with RSSI -19  
Received packet 'HELLO' with RSSI -19  
Received packet 'HELLO' with RSSI -39  
Received packet 'HELLO' with RSSI -38  
Received packet 'HELLO' with RSSI -18  
Received packet 'HELLO' with RSSI -39  
Received packet 'HELLO' with RSSI -19
```

5. FAQ

5.1 Why there is 433/868/915 version?

Different country has different rules for the ISM band for using the LoRa. Although the LoRa chip can support a wide range of Frequency, we provide different version for best tune in the LoRa part. That is why we provide different version of LoRa.

5.2 What is the frequency range of LoRa GPS HAT part?

Different LT version supports different frequency range, below is the table for the working frequency and recommend bands for each model :

Version	LoRa IC	Working Frequency	Best Tune Frequency	Recommend Bands
433	SX1278	Band2(LF): 410 ~525 Mhz	433Mhz	CN470/EU433
868	SX1276	Band1(HF):862~1020 Mhz	868Mhz	EU868
915	SX1276	Band1(HF):862 ~1020 Mhz	915Mhz	AS923/AU915/ KR920/US915
JP	SX1276	Band1(HF):862 ~1020 Mhz	915Mhz	AS923/AU915/ KR920/US915

6. Order Info

Part Number: **LoRa-GPS-HAT-XXX**

XXX: The best tuned frequency

- ✓ **433**: Best Tuned 433Mhz
- ✓ **868**: Best Tuned 868Mhz
- ✓ **915**: Best Tuned 915Mhz

7. Packing Info

Package Includes:

- ✓ 1 x LoRa/GPS HAT
- ✓ 4 x Brass cylinders
- ✓ 4 x Screws
- ✓ 4 x Nuts
- ✓ 1 x Glue Stick Antenna(868 MHZ, 433 MHZ or 915 MHZ depends on order)

Dimension and weight:

- ✓ Device Size: 10 x 8 x 3 mm/pcs
- ✓ Device Weight: G.W. 72g/pcs

8. Support

- Support is provided Monday to Friday, from 09:00 to 18:00 GMT+8. Due to different timezones we cannot offer live support. However, your questions will be answered as soon as possible in the before-mentioned schedule.
- Provide as much information as possible regarding your enquiry (product models, accurately describe your problem and steps to replicate it etc) and send a mail to

support@dragino.com