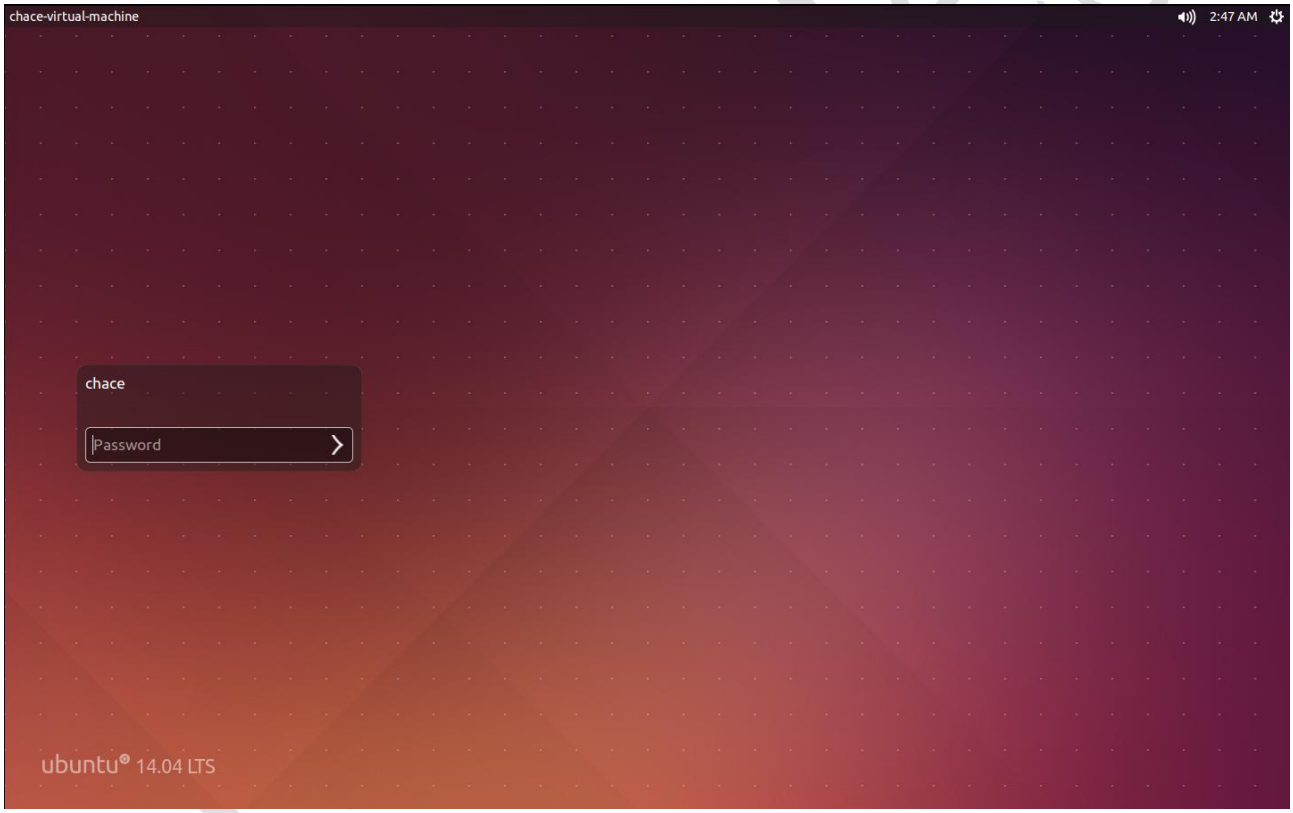
RAK831 Quick Start Guide V1.0

Content

1. Required materials (hardware, tools)

- RAK831 LoRa Gateway board x1
- FT2232HL Module x1
- Mini USB Data lines x1
- Ubuntu

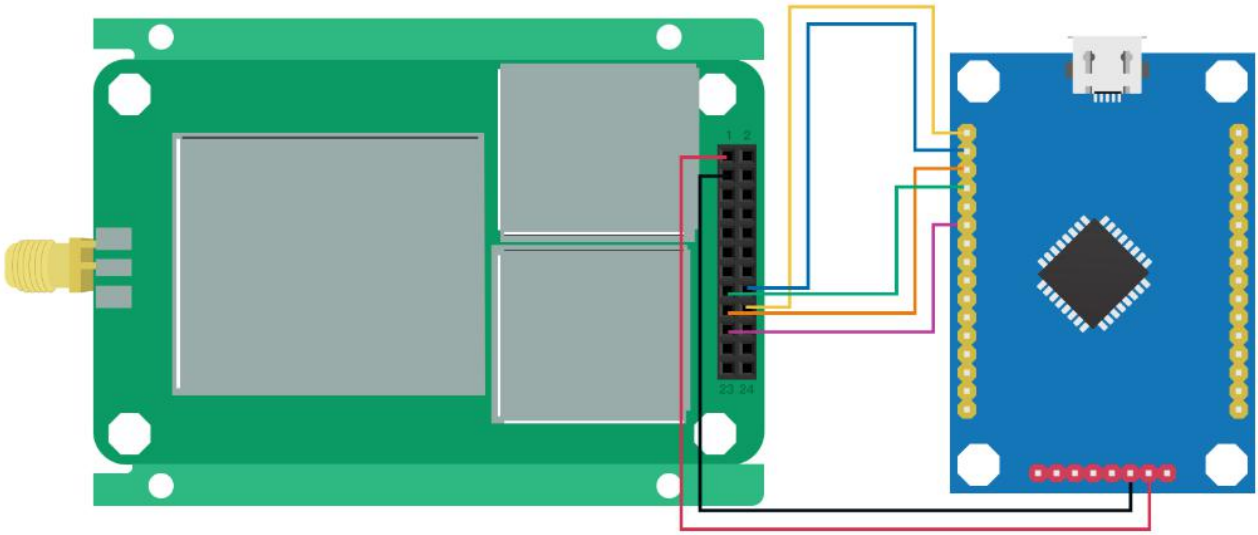
Note : The environment needs to use the entire Linux system to support, recommend the use of Ubuntu system, other Liunx system also can be achieved.



2. Hardware connection

As the RAK831 uses SPI interface communication, so the computer to communicate with it must use USB to SPI interface to achieve, so the use of FT2232HL module to achieve conversion, RAK831 and FT2232HL wiring as follows:

FT2232HL	RAK831	Pin meaning
VDD5V	Pin1	5V
GND	Pin3	GND
ADBUS0	Pin18	SCK
ADBUS1	Pin16	MOSI
ADBUS2	Pin17	MISO
ADBUS3	Pin15	CSN
ADBUS5	Pin19	RST



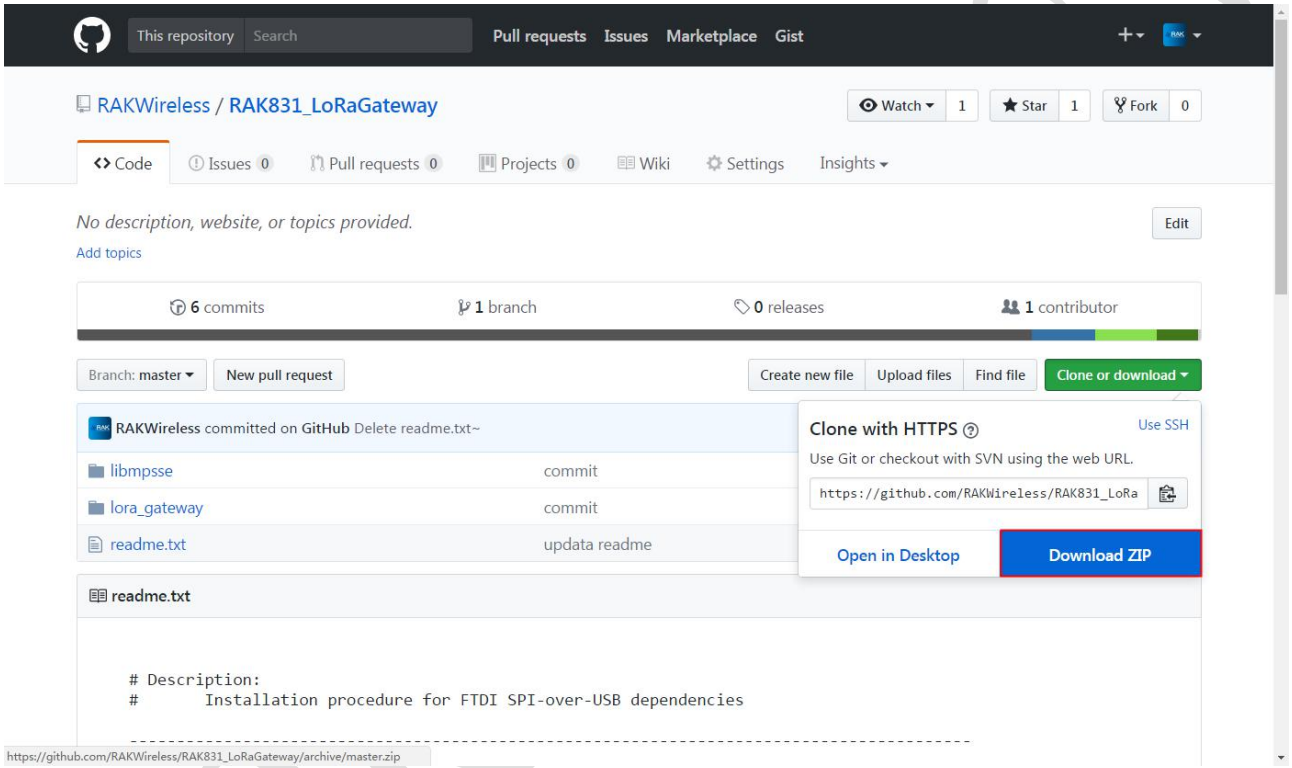
3. Install and compile

3.1 Download of the Open Source Driver

You can download the open source code on our official github:

https://github.com/RAKWireless/RAK831_LoRaGateway

If you are using a Windows system, you can click here to download.And then extract, copy the file to your own virtual machine Liunx system.We do not recommend this method to obtain, you directly in the Liunx system environment will be more simple to obtain.



If you are on a Linux system, you can use the following two methods to get the open source code.

Method 1. If you have already installed the github environment on Linux system, then you can get it directly using the github cloning feature.

git clone https://github.com/RAKWireless/RAK831_LoRaGateway.git

```
chace@chace-virtual-machine:~$ git clone https://github.com/RAKWireless/RAK831_LoRaGateway.git
Cloning into 'RAK831_LoRaGateway'...
remote: Counting objects: 133, done.
remote: Compressing objects: 100% (104/104), done.
remote: Total 133 (delta 22), reused 126 (delta 20), pack-reused 0
Receiving objects: 100% (133/133), 854.75 KiB | 147.00 KiB/s, done.
Resolving deltas: 100% (22/22), done.
Checking connectivity... done.
chace@chace-virtual-machine:~$
```

Method 2. If you do not have the github environment installed, it does not matter. You can use the following command to get the open source code.

wget https://github.com/RAKWireless/RAK831_LoRaGateway/archive/master.zip

```
chace@chace-virtual-machine:~$ wget https://github.com/RAKWireless/RAK831_LoRaGateway/archive/master.zip
--2017-07-16 20:05:41--  https://github.com/RAKWireless/RAK831_LoRaGateway/archive/master.zip
Resolving github.com (github.com)... 192.30.255.113, 192.30.255.112
Connecting to github.com (github.com)|192.30.255.113|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://codeload.github.com/RAKWireless/RAK831_LoRaGateway/zip/master [following]
--2017-07-16 20:05:42--  https://codeload.github.com/RAKWireless/RAK831_LoRaGateway/zip/master
Resolving codeload.github.com (codeload.github.com)... 192.30.255.120, 192.30.255.121
Connecting to codeload.github.com (codeload.github.com)|192.30.255.120|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/zip]
Saving to: ‘master.zip’

    [               <=>

2017-07-16 20:05:48 (222 KB/s) - ‘master.zip’ saved [908961]
```

Unzip the downloaded file：

unzip master.zip

```
chace@chace-virtual-machine:~$ unzip master.zip
Archive:  master.zip
bbc192dffcaaaa7577e43a2cf89304b88728d46b
   creating: RAK831_LoRaGateway-master/
   creating: RAK831_LoRaGateway-master/libmpsse/
  inflating: RAK831_LoRaGateway-master/libmpsse/README.md
   creating: RAK831_LoRaGateway-master/libmpsse/docs/
  inflating: RAK831_LoRaGateway-master/libmpsse/docs/AN_135_MPSSE_Basics.pdf
  inflating: RAK831_LoRaGateway-master/libmpsse/docs/COPYING
  inflating: RAK831_LoRaGateway-master/libmpsse/docs/INSTALL
  inflating: RAK831_LoRaGateway-master/libmpsse/docs/README
  inflating: RAK831_LoRaGateway-master/libmpsse/docs/README.BITBANG
  inflating: RAK831_LoRaGateway-master/libmpsse/docs/README.C
  inflating: RAK831_LoRaGateway-master/libmpsse/docs/README.GPIO
  inflating: RAK831_LoRaGateway-master/libmpsse/docs/README.I2C
  inflating: RAK831_LoRaGateway-master/libmpsse/docs/README.PYTHON
  inflating: RAK831_LoRaGateway-master/libmpsse/docs/README.SPI
   creating: RAK831_LoRaGateway-master/libmpsse/src/
  inflating: RAK831_LoRaGateway-master/libmpsse/src/Makefile
  inflating: RAK831_LoRaGateway-master/libmpsse/src/Makefile.in
  inflating: RAK831_LoRaGateway-master/libmpsse/src/config.h
  inflating: RAK831_LoRaGateway-master/libmpsse/src/config.log
  inflating: RAK831_LoRaGateway-master/libmpsse/src/config.status
  inflating: RAK831_LoRaGateway-master/libmpsse/src/configure
  inflating: RAK831_LoRaGateway-master/libmpsse/src/configure.ac
   creating: RAK831_LoRaGateway-master/libmpsse/src/examples/
  inflating: RAK831_LoRaGateway-master/libmpsse/src/examples/Makefile
  inflating: RAK831_LoRaGateway-master/libmpsse/src/examples/at93c46d.py
  inflating: RAK831_LoRaGateway-master/libmpsse/src/examples/bitbang.c
  inflating: RAK831_LoRaGateway-master/libmpsse/src/examples/bitbang.py
  inflating: RAK831_LoRaGateway-master/libmpsse/src/examples/ds1305.c
  inflating: RAK831_LoRaGateway-master/libmpsse/src/examples/ds1305.py
  inflating: RAK831_LoRaGateway-master/libmpsse/src/examples/gpio.c
```

3.2 Install libFTDI

Due to the need to use the FT2232HL module, it is necessary to install Liunx driver of FT2232HL module. The installation process is simple and can be done according to the following steps :

Step : install libftdi-dev

sudo apt-get install libftdi-dev

```
chace@chace-virtual-machine:~$ sudo apt-get install libftdi-dev
[sudo] password for chace:
Reading package lists... Done
Building dependency tree
Reading state information... Done
libftdi-dev is already the newest version.
0 upgraded, 0 newly installed, 0 to remove and 452 not upgraded.
chace@chace-virtual-machine:~$
```

3.3 Install the libmpsse

Step 1 : Go to download the open source code under the ../libmpsse/src directory.Execute the following command.

sudo ./configure --disable-python

```
chace@chace-virtual-machine:~/RAK831_LoRaGateway/libmpsse/src$ sudo ./configure --disable-python
[sudo] password for chace:
configure: WARNING: unrecognized options: --disable-python
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking how to run the C preprocessor... gcc -E
checking for grep that handles long lines and -e... /bin/grep
checking for egrep... /bin/grep -E
checking for ANSI C header files... yes
checking for sys/types.h... yes
checking for sys/stat.h... yes
checking for stdlib.h... yes
checking for string.h... yes
checking for memory.h... yes
checking for strings.h... yes
checking for inttypes.h... yes
checking for stdint.h... yes
checking for unistd.h... yes
checking for size_t... yes
checking for stdlib.h... (cached) yes
checking for GNU libc compatible malloc... yes
checking libftdi1/ftdi.h usability... no
checking libftdi1/ftdi.h presence... no
checking for libftdi1/ftdi.h... no
checking ftdi.h usability... yes
checking ftdi.h presence... yes
checking for ftdi.h... yes
checking for ftdi_init in -lftdi... yes
configure: creating ./config.status
config.status: creating Makefile
configure: WARNING: unrecognized options: --disable-python
chace@chace-virtual-machine:~/RAK831_LoRaGateway/libmpsse/src$
```

Step 2 : And then compile

make

```
chace@chace-virtual-machine:~/RAK831_LoRaGateway/libmpsse/src$ make
gcc -Wall -fPIC -fno-strict-aliasing -g -O2 -lftdi  -DLIBFTDI1=0 -c fast.c
gcc -Wall -fPIC -fno-strict-aliasing -g -O2 -shared -Wl,-soname,libmpsse.so *.o -o libmpsse.so -lftdi
ar rcs libmpsse.a *.o
chace@chace-virtual-machine:~/RAK831_LoRaGateway/libmpsse/src$
```

Step 3 : # Static and dynamic libraries compiled code is put into /usr/local/lib

Header file is put into /usr/local/include

sudo make install

```
chace@chace-virtual-machine:~/RAK831_LoRaGateway/libmpsse/src$ sudo make install
install -D -m644 libmpsse.so //usr/local/lib/libmpsse.so
install -D -m644 libmpsse.a  //usr/local/lib/libmpsse.a
install -D -m644 mpsse.h     //usr/local/include/mpsse.h
chace@chace-virtual-machine:~/RAK831_LoRaGateway/libmpsse/src$
```

Step 4 : On the Pcduino, you must regenerate the library cache (might some time).

sudo ldconfig

```
chace@chace-virtual-machine:~/RAK831_LoRaGateway/libmpsse/src$ sudo ldconfig
chace@chace-virtual-machine:~/RAK831_LoRaGateway/libmpsse/src$
```

3.4 Build the library

Unpack the LoRa Gateway project and go to lora_gateway directory. then build the library and examples.

make all

```
chace@chace-virtual-machine:~/RAK831_LoRaGateway/lora_gateway$ make all
make all -e -C libloragw
make[1]: Entering directory `/home/chace/RAK831_LoRaGateway/lora_gateway/libloragw'
*** Checking libloragw library configuration ***
Release version   : 1.5.0
SPI interface     : FTDI SPI-over-USB bridge using libmpsse/libftdi/libusb
Concentrator chip : Semtech SX1301 production chip
Radio chip(s)     : Auto check front end.
Frequency band    : Full range supported by the radio(s)
Board misc. param : China LoRa 433MHz gateway for RPi
Network type      : Private network
*** Configuration seems ok ***
gcc -c -O2 -Wall -Wextra -std=c99 -Iinc -I. src/loragw_hal.c -o obj/loragw_hal.o
gcc -c -O2 -Wall -Wextra -std=c99 -Iinc -I. src/loragw_gps.c -o obj/loragw_gps.o
gcc -c -O2 -Wall -Wextra -std=c99 -Iinc -I. src/loragw_reg.c -o obj/loragw_reg.o
gcc -c -O2 -Wall -Wextra -std=c99 -Iinc -I. src/loragw_spi.ftdi.c -o obj/loragw_spi.o
gcc -c -O2 -Wall -Wextra -std=c99 -Iinc -I. src/loragw_aux.c -o obj/loragw_aux.o
ar rcs libloragw.a obj/loragw_hal.o obj/loragw_gps.o obj/loragw_reg.o obj/loragw_spi.o obj/loragw_aux.o
gcc -O2 -Wall -Wextra -std=c99 -Iinc -I. -L. tst/test_loragw_spi.c -o test_loragw_spi -lloragw -lrt -lmpsse
gcc -O2 -Wall -Wextra -std=c99 -Iinc -I. -L. tst/test_loragw_reg.c -o test_loragw_reg -lloragw -lrt -lmpsse
gcc -O2 -Wall -Wextra -std=c99 -Iinc -I. -L. tst/test_loragw_hal.c -o test_loragw_hal -lloragw -lrt -lmpsse
gcc -O2 -Wall -Wextra -std=c99 -Iinc -I. -L. tst/test_loragw_tx.c -o test_loragw_tx -lloragw -lrt -lmpsse
gcc -O2 -Wall -Wextra -std=c99 -Iinc -I. -L. tst/test_loragw_rx.c -o test_loragw_rx -lloragw -lrt -lmpsse
gcc -O2 -Wall -Wextra -std=c99 -Iinc -I. -L. tst/test_loragw_gps.c -o test_loragw_gps -lloragw -lrt -lmpsse
gcc -O2 -Wall -Wextra -std=c99 -Iinc -I. -L. tst/test_loragw_full_duplex.c -o test_loragw_full_duplex -lloragw -lrt -lmpsse
make[1]: Leaving directory `/home/chace/RAK831_LoRaGateway/lora_gateway/libloragw'
make all -e -C util_band_survey
make[1]: Entering directory `/home/chace/RAK831_LoRaGateway/lora_gateway/util_band_survey'
gcc -c -O2 -Wall -Wextra -std=c99 -Iinc -I. -I../libloragw/inc src/util_band_survey.c -o obj/util_band_survey.o
gcc -L../libloragw obj/util_band_survey.o -o util_band_survey -lloragw -lrt -lmpsse
make[1]: Leaving directory `/home/chace/RAK831_LoRaGateway/lora_gateway/util_band_survey'
make all -e -C util_pkt_logger
make[1]: Entering directory `/home/chace/RAK831_LoRaGateway/lora_gateway/util_pkt_logger'
gcc -c -O2 -Wall -Wextra -std=c99 -Iinc -I. -I../libloragw/inc src/util_pkt_logger.c -o obj/util_pkt_logger.o
gcc -c -O2 -Wall -Wextra -std=c99 -Iinc -I. src/parson.c -o obj/parson.o
gcc -L../libloragw obj/util_pkt_logger.o obj/parson.o -o util_pkt_logger -lloragw -lrt -lmpsse
cp cfg/global_conf.empty.json global_conf.json
make[1]: Leaving directory `/home/chace/RAK831_LoRaGateway/lora_gateway/util_pkt_logger'
make all -e -C util_spi_stress
make[1]: Entering directory `/home/chace/RAK831_LoRaGateway/lora_gateway/util_spi_stress'
gcc -c -O2 -Wall -Wextra -std=c99 -Iinc -I. -I../libloragw/inc src/util_spi_stress.c -o obj/util_spi_stress.o
gcc -L../libloragw obj/util_spi_stress.o -o util_spi_stress -lloragw -lrt -lmpsse
make[1]: Leaving directory `/home/chace/RAK831_LoRaGateway/lora_gateway/util_spi_stress'
make all -e -C util_tx_test
```

4. LoRaP2P Test

After completing the above work, you can start the test. It is necessary to confirm whether the Liunx system recognizes the FT2232HL module and whether the wiring is connected. So first need SPI communication test.

4.1 SPI test

Go to the ../lora_gateway/libloragw directory. Execute the following command.

sudo ./test_loragw_reg

If the following information is returned. Indicates that SPI communication failed. Then you need to check whether the wiring is not connected, or FT2232 driver is not recognized.

```
chace@chace-virtual-machine:~/RAK831_LoRaGateway/lora_gateway/libloragw$ sudo ./test_loragw_reg
[sudo] password for chace:
Beginning of test for loragw_reg.c
ERROR: CONCENTRATOR UNCONNECTED
IMPLICIT_PAYLOAD_LENGHT = 0 (should be 197)
FRAME_SYNCH_PEAK2_POS = 0 (should be 11)
PREAMBLE_SYMB1_NB = 0 (should be 49253)
ADJUST_MODEM_START_OFFSET_SF12_RDX4 = 0 (should be 3173)
IF_FREQ_1 = 0 (should be -1947)
End of test for loragw_reg.c
chace@chace-virtual-machine:~/RAK831_LoRaGateway/lora_gateway/libloragw$
```

If this information is returned, it indicates that the SPI is on. At this point you can test the module other functions.

```
+++MATCH+++ reg number 318 read: 0 (0) default: 0 (0)
+++MATCH+++ reg number 319 read: 0 (0) default: 0 (0)
+++MATCH+++ reg number 320 read: 0 (0) default: 0 (0)
+++MATCH+++ reg number 321 read: 0 (0) default: 0 (0)
+++MATCH+++ reg number 322 read: 0 (0) default: 0 (0)
+++MATCH+++ reg number 323 read: 0 (0) default: 0 (0)
+++MATCH+++ reg number 324 read: 0 (0) default: 0 (0)
End of register verification
IMPLICIT_PAYLOAD_LENGHT = 197 (should be 197)
FRAME_SYNCH_PEAK2_POS = 11 (should be 11)
PREAMBLE_SYMB1_NB = 49253 (should be 49253)
ADJUST_MODEM_START_OFFSET_SF12_RDX4 = 3173 (should be 3173)
IF_FREQ_1 = -1947 (should be -1947)
End of test for loragw_reg.c
```

4.2 RX test

In the ../lora_gateway/libloragw directory, execute the following command to test the module receive function.

sudo ./test_loragw_rx 868.1 868.5

Note : the 868.1 is test the reception of Radio A 868.5 is test the reception of Radio A

```
chace@chace-virtual-machine:~/RAK831_LoRaGateway/lora_gateway/libloragw$ sudo ./test_loragw_rx 868.1 868.5
[sudo] password for chace:
Beginning of test for loragw_hal.c
*** Library version information ***
Version: 1.5.0; Options: ftdi sx1301 auto-check full ref_1301_433_v2 private;

F_RX0 = 868500000, F_RX1 = 868900000
8 freqeuncy channels are selected
channel: 0, freq: 868100000
channel: 1, freq: 868300000
channel: 2, freq: 868500000
channel: 3, freq: 868700000
channel: 4, freq: 868500000
channel: 5, freq: 868700000
channel: 6, freq: 868900000
channel: 7, freq: 869100000
*** Concentrator started ***
```

If the sending device sends data in the test band, the module will receive this information.

```
Rcv pkt #1 >>freq:868300000
 if_chain: 1 tstamp:0125939387 size:  3 LoRa SF7 CR1(4/5)
 RSSI: -76.0 SNR: +7.5 (min: +7.2, max:+13.5) payload:
 01 01 01 #
---
Rcv pkt #1 >>freq:868300000
 if_chain: 1 tstamp:0126973067 size:  3 LoRa SF7 CR1(4/5)
 RSSI: -74.0 SNR: +7.5 (min: +8.8, max:+12.2) payload:
 01 01 01 #
---
Rcv pkt #1 >>freq:868300000
 if_chain: 1 tstamp:0128006779 size:  3 LoRa SF7 CR1(4/5)
 RSSI: -76.0 SNR: +8.5 (min: +9.0, max:+15.5) payload:
 01 01 01 #
---
Rcv pkt #1 >>freq:868300000
 if_chain: 1 tstamp:0129040515 size:  3 LoRa SF7 CR1(4/5)
 RSSI: -75.0 SNR: +7.8 (min: +7.5, max:+14.0) payload:
 01 01 01 #
---
Rcv pkt #1 >>freq:868300000
 if_chain: 1 tstamp:0130074243 size:  3 LoRa SF7 CR1(4/5)
 RSSI: -74.0 SNR: +6.2 (min: +7.0, max:+10.8) payload:
 01 01 01 #
---
```

5. Contact information

Shanghai

FAE mailbox:steven.tang@rakwireless.com

Tel : +86 133 9124 2711

Address: Room B205, Green light kechuang garden, 2588 Lane, Hongmei South road, Minhang District, Shanghai

Shenzhen

FAE mailbox: ken.yu@rakwireless.com

Tel : 0755-86108311

Fax: 0755-86152201

Address: Room 1007, Hangsheng Technology Building, South Four Road, Science and Technology Park, Nanshan District, Shenzhen

6. Change Note

Version	Date	Change
V1.0	2017-07-17	Draft