E106 Series LoRa Wireless Module

Instructions for Using E106 Series Modules
with the Official SDK on Raspberry Pi



Chengdu Ebyte Electronic Technology Co.,Ltd.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Contents

Introduction: 3
Step 1: Setting up the Raspberry Pi environment.....3
Step 2: Download and compile the source code.....3
Step 3: Connect the Raspberry Pi and the E106 6
Step 4:Testing E106 module.....6
Revision History 12
About Us 12

Copyright ©2012–2026, Chengdu Ebyte Electronic Technology Co.,Ltd.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Introduction:

This module is a hardware module that requires an external device to drive it. This document uses the Raspberry Pi used in the official Semtech examples to drive this module . This gateway module uses the Semtech SX1302 as the main chip. Since the official documentation for this chip does not provide a register manual, driving this module and any secondary development must refer to the official examples. The SX1303 module is identical to the SX1302 except for the addition of fine-grained timestamp functionality.

Step 1: Setting up the Raspberry Pi environment

First, the Raspberry Pi system must be installed and accessible via SSH. If the system is not installed, you can download the image from the following link.

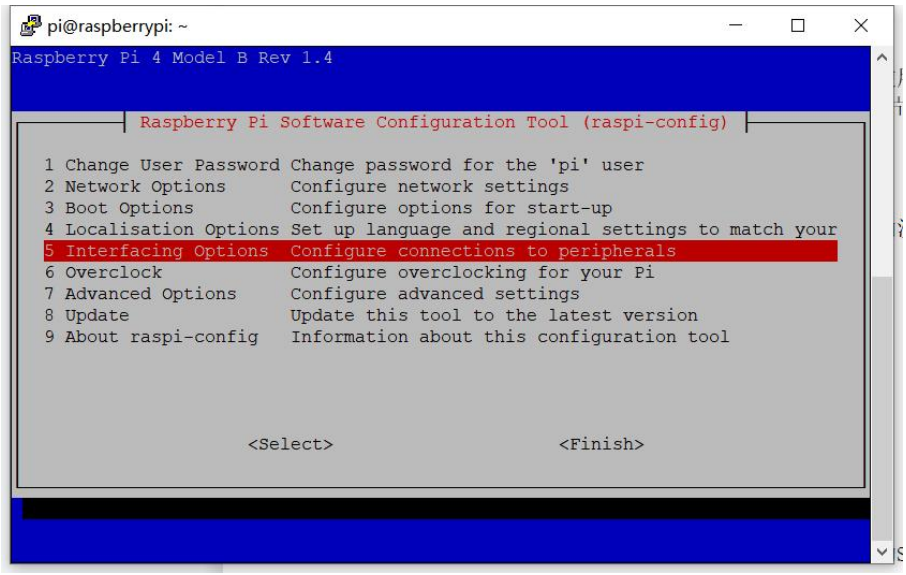
· Raspbin Lite (Simplified Version)
· raspbin full version

First, log in to the Raspberry Pi system via SSH.
Update Linux ~ Enter the following commands to update
- $ sudo apt-get update
- $ sudo apt-get upgrade
- $ sudo apt-get dist-upgrade
- $ sudo rpi-update

Install Git service
- $ sudo apt install git

• Enable Linux SPI/I2C/UART (Enter the command to access the options and select Enable SPI/I2C/UART)

- $ sudo raspi-config:



Enter the reboot command to restart.

- $ sudo reboot

Step 2: Download and compile the source code

• Obtain the latest Semtech software package from LoRa® GitHub (internet connection required)

- $ git clone https://github.com/Lora-net/sx1302_hal.git
- $ cd ~/sx1302_hal/

Copyright ©2012–2026, Chengdu Ebyte Electronic Technology Co.,Ltd.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Since we are not using an I2C temperature sensor or AD converter , this part needs to be disabled; otherwise, errors will occur during use. (Alternatively, you can directly use the modified SDK from our official website without modification.)

```
sudo vim libloragw/src/loragw_ha l .c
```

```
Line 205 :
#if 0
/* I2C temperature sensor handles */
static int ts_fd = -1 ;
static uint8_t ts_addr = 0xFF;
/* I2C AD5338 handles */
static int ad_fd = -1 ;
#endif

109, line 4 :
#if 0
if (CONTEXT_COM_TYPE == LGW_COM_SPI) {
/* Find the temperature sensor on the known supported ports */
for (i = 0 ; i < (int)(sizeof I2C_PORT_TEMP_SENSOR); i ++ ) {
ts_addr = I2C_PORT_TEMP_SENSOR[i];
err = i2c_linuxdev_open(I2C_DEVICE, ts_addr, &ts_fd);
if (err ! = LGW_I2C_SUCCESS) {
printf( "ERROR: failed to open I2C for temperature sensor on
port 0x%02X\n" , ts_addr);
return LGW_HAL_ERROR;
}
err = stts751_configure(ts_fd, ts_addr);
if (err ! = LGW_I2C_SUCCESS) {
printf( "INFO: no temperature sensor found on port 0x%02X\n" ,
ts_addr);
i2c_linuxdev_close(ts_fd);
ts_fd = -1 ;
} else {
printf( "INFO: found temperature sensor on port 0x%02X\n" ,
ts_addr);
break
}
}
if (i == sizeof I2C_PORT_TEMP_SENSOR) {
printf( "ERROR: no temperature sensor found.\n" );
return LGW_HAL_ERROR;
}
/* Configure ADC AD338R for full duplex (CN490 reference design) */
if (CONTEXT_BOARD.full_duplex == true ) {
err = i2c_linuxdev_open(I2C_DEVICE, I2C_PORT_DAC_AD5338R,
&ad_fd);
if (err ! = LGW_I2C_SUCCESS) {
printf( "ERROR: failed to open I2C for ad5338r\n" );
return LGW_HAL_ERROR;
}
err = ad5338r_configure(ad_fd, I2C_PORT_DAC_AD5338R);
if (err ! = LGW_I2C_SUCCESS) {
printf( "ERROR: failed to configure ad5338r\n" );
i2c_linuxdev_close(ad_fd);
ad_fd = -1 ;
return LGW_HAL_ERROR;
}/* Turn off the PA: set DAC output to 0V */
uint8_t volt_val[AD5338R_CMD_SIZE] = { 0x39,
(uint8_t)VOLTAGE2HEX_H(0), (uint8_t)VOLTAGE2HEX_L(0) };
err = ad5338r_write(ad_fd, I2C_PORT_DAC_AD5338R, volt_val);
if (err ! = LGW_I2C_SUCCESS) {
printf( "ERROR: AD5338R: failed to set DAC output to 0V\n" );
return LGW_HAL_ERROR;
}
printf( "INFO: AD5338R: Set DAC output to 0x%02X 0x%02X\n" ,
```

Copyright ©2012–2026, Chengdu Ebyte Electronic Technology Co.,Ltd.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

```
(uint8_t)VOLTAGE2HEX_H(0), (uint8_t)VOLTAGE2HEX_L(0));
}
}
#endif
```

Line 1225 :

```
#if 0
if (CONTEXT_COM_TYPE == LGW_COM_SPI) {
DEBUG_MSG( "INFO: Closing I2C for temperature sensor\n" );
x = i2c_linuxdev_close(ts_fd);
if (x ! = 0 ) {
printf( "ERROR: failed to close I2C temperature sensor device
(err=%i)\n" , x);
err = LGW_HAL_ERROR;
}
if (CONTEXT_BOARD.full_duplex == true ) {
DEBUG_MSG( "INFO: Closing I2C for AD5338R\n" );
x = i2c_linuxdev_close(ad_fd);
if (x ! = 0 ) {
printf( "ERROR: failed to close I2C AD5338R device
(err=%i)\n" , x);
err = LGW_HAL_ERROR;
}
}
}
#endif
```

Line 1289 :

```
#if 0
res = lgw_get_temperature(&current_temperature);
if (res ! = LGW_I2C_SUCCESS) {
printf( "ERROR: failed to get current temperature\n" );
return LGW_HAL_ERROR;
}
#endif
```

Line 1416:

```
#if 0
/* Set PA gain with AD5338R when using full duplex CN490 ref design */
if (CONTEXT_BOARD.full_duplex == true) {
uint8_t volt_val[AD5338R_CMD_SIZE] = {0x39, VOLTAGE2HEX_H(2.51), VOLTAGE2HEX_L(2.51)};
err = ad5338r_write(ad_fd, I2C_PORT_DAC_AD5338R, volt_val);
if (err != LGW_I2C_SUCCESS) {
printf(" ERROR: failed to set voltage by ad5338r\n ");
return LGW_HAL_ERROR;
}
printf( "INFO: AD5338R: Set DAC output to 0x%02X 0x%02X\n" ,
(uint8_t)VOLTAGE2HEX_H(2.51),(uint8_t)VOLTAGE2HEX_L(2.51));
}
#endif
```

Line 1592 : In the lgw_get_temperature function
int err = LGW_HAL_ERROR; changed to int err = LGW_HAL_SUCCESS;
1600 : In the lgw_get_temperature function
//err = stts751_get_temperature(ts_fd, ts_addr, temperature);

· Compile source code

In the sx1302_hal directory
- $ make clean all
` $ ssh -keygen -t rsa`, are executed to avoid entering the user password during the installation process.
- $ ssh -copy-id -i ~/.ssh/id_rsa.pub pi@localhost You should install the following after entering your Raspberry Pi password:
There is a program
- $ make install
- `$ make install_conf` copies executable files such as global_conf.json to the bin folder.

Copyright ©2012–2026, Chengdu Ebyte Electronic Technology Co.,Ltd.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

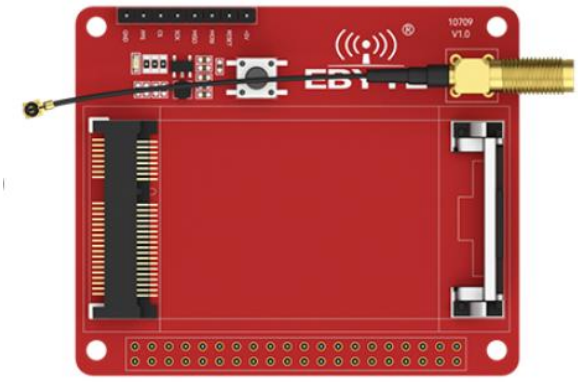
Step 3: Connect the Raspberry Pi and the E106

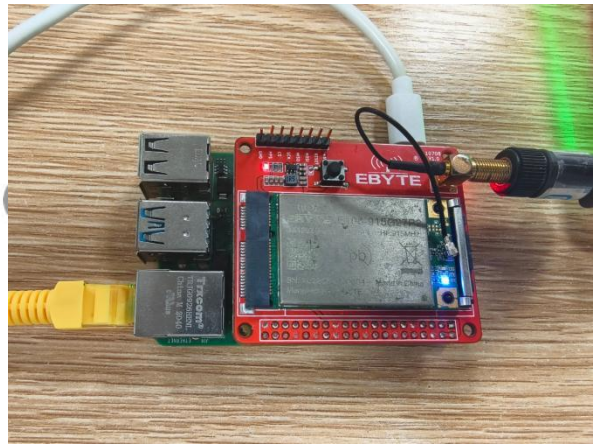
This document describes how to connect an E106 module to a digital strawberry Pi using our E15-LW-T1 mini-PCIe to SPI adapter board. The main connections are the four SPI pins, the sx1302_reset pin, VCC, and GND.

The location of the sx1302_reset pin on the Raspberry Pi varies depending on the adapter board used. Modify the reset_lgw.sh file in the sx1302_hal/bin folder according to your specific situation.

In this script, we will only use the SX1302_RESET_PIN pin; the others can be commented out.

By default, SX1302_RESET_PIN=23 corresponds to pin 23 in Raspberry Pi kernels below Linux 6.0; however, newer systems flashed directly to the Raspberry Pi Imager are based on Linux 6.0 and above, and the pin name used directly has changed, requiring the addition of the base 512. If a new system requires modifying SX1302_RESET_PIN=535, the Raspberry Pi kernel version can be checked using the `uname -a` command.





Step 4: Testing E106 module

1、Register test

After logging in, navigate to the directory containing the test commands: cd lorawan/sx1302_hal/bin

Then view the files in the current folder: ls

```
pi@raspberrypi:~ $ cd lorawan/sx1302_hal/bin
pi@raspberrypi:~/lorawan/sx1302_hal/bin $ ls
boot                                    spectral_scan
chip_id                                 test_loragw_cal_sx125x
global_conf.json                        test_loragw_capture_ram
global_conf.json.sx1250.AS923.USB       test_loragw_com
global_conf.json.sx1250.CN490           test_loragw_com_sx1250
global_conf.json.sx1250.CN490.USB       test_loragw_com_sx1261
global_conf.json.sx1250.EU868           test_loragw_counter
global_conf.json.sx1250.EU868.USB       test_loragw_gps
global_conf.json.sx1250.US915           test_loragw_hal_rx
global_conf.json.sx1250.US915.USB       test_loragw_hal_tx
global_conf.json.sx1255.CN490.full-duplex  test_loragw_i2c
global_conf.json.sx1257.EU868           test_loragw_reg
lora_pkt_fwd                            test_loragw_sx1261_rssi
net_downlink                            test_loragw_toa
reset_lgw.sh
pi@raspberrypi:~/lorawan/sx1302_hal/bin $
```

Enter test command

- $ ./test_loragw_reg

Copyright ©2012–2026, Chengdu Ebyte Electronic Technology Co.,Ltd.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

```
pi@raspberrypi:~/lorawan/sx1302_hal/bin $ ./test_loragw_reg
CoreCell reset through GPIO7...
SX1261 reset through GPIO7...
CoreCell power enable through GPIO18...
CoreCell ADC reset through GPIO13...
Opening SPI communication interface
Note: chip version is 0x10 (v1.0)
## TEST#1: read all registers and check default value for non-read-only register
s
-------------------
 TEST#1 PASSED
-------------------

## TEST#2: read/write test on all non-read-only, non-pulse, non-w0clr, non-w1clr
 registers
-------------------
 TEST#2 PASSED


Closing SPI communication interface
CoreCell reset through GPIO7...
SX1261 reset through GPIO7...
CoreCell power enable through GPIO18...
CoreCell ADC reset through GPIO13...
pi@raspberrypi:~/lorawan/sx1302_hal/bin $
```

If both TEST1 and TEST2 are PASSED , it means the register test is passed . If they fail, please check if the reset pin is correct and if the SPI communication is normal. If the register test is passed, it means the module is basically fine.

2. Send test

First, use the command: ./test_loragw_hal_tx -h View the help information and set different parameters according to your needs.

```
pi@raspberrypi:~/lorawan/sx1302_hal/bin $ ./test_loragw_hal_tx -h
Available options:
 -h print this help
 -u         Set COM type as USB (default is SPI)
 -d <path>  COM path to be used to connect the concentrator
            => default path: /dev/spidev0.0
 -k <uint>  Concentrator clock source (Radio A or Radio B) [0..1]
 -c <uint>  RF chain to be used for TX (Radio A or Radio B) [0..1]
 -r <uint>  Radio type (1255, 1257, 1250)
 -f <float> Radio TX frequency in MHz
 -m <str>   modulation type ['CW', 'LORA', 'FSK']
 -o <int>   CW frequency offset from Radio TX frequency in kHz [-65..65]
 -s <uint>  LoRa datarate 0:random, [5..12]
 -b <uint>  LoRa bandwidth in khz 0:random, [125, 250, 500]
 -l <uint>  FSK/LoRa preamble length, [6..65535]
 -n <uint>  Number of packets to be sent
 -z <uint>  size of packets to be sent 0:random, [9..255]
 -t <uint>  TX mode timestamped with delay in ms. If delay is 0, TX mode GPS trigger
 -p <int>   RF power in dBm
 -i         Send LoRa packet using inverted modulation polarity
 -j         Set radio in single input mode (SX1250 only)
 --fdev <uint>  FSK frequency deviation in kHz [1:250]
 --br   <float> FSK bitrate in kbps [0.5:250]
 --pa   <uint> PA gain SX125x:[0..3], SX1250:[0,1]
 --dig  <uint> sx1302 digital gain for sx125x [0..3]
 --dac  <uint> sx125x DAC gain [0..3]
 --mix  <uint> sx125x MIX gain [5..15]
 --pwid <uint> sx1250 power index [0..22]
 --nhdr        Send LoRa packet with implicit header
 --loop        Number of loops for HAL start/stop (HAL unitary test)
 --fdd         Enable Full-Duplex mode (CN490 reference design)
pi@raspberrypi:~/lorawan/sx1302_hal/bin $
```

Copyright ©2012–2026, Chengdu Ebyte Electronic Technology Co.,Ltd.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Several important parameter configurations

Order	parameter	effect
-c	0 or 1	Configure RF chip A(0) or RF Or RF chip B (1) is for transmitting
-r	1255,1257,1250	For the RF chip selection, our module uses the 1250.
-f	Frequency (MHz)	Set the sending frequency
-m	'CW', 'LORA', 'FSK'	Work mode
-o	[-65..65]khz	Transmission frequency offset
- s	[5…12]	Spreading factor
-b	[125,250,500]	bandwidth
-l	[6…65535]	Fsk/lora preamble length
-n	any positive number	Number of packets sent
-z	[9…255]	How many bytes are sent per packet?
- - pa	[0,1]	RF chip P A
- pwid	[0…22]	Transmit power

For example:

./test_loragw_hal_tx -r 1250 -f 915.2 -m LORA -b 125 -s 12 -z 20 --pa 1 --pwid 20

```
pi@raspberrypi:~/lorawan/sx1302_hal/bin $ ./test_loragw_hal_tx -r 1250 -f 915.2 -m LORA -b 125 -s 12 -z 20 --p
a 1 --pwid 2
Sending 1 LoRa packets on 915200000 Hz (BW 125 kHz, SF 12, CR 1, 20 bytes payload, 8 symbols preamble, explici
t header, non-inverted polarity) at 0 dBm
CoreCell reset through GPIO7...
SX1261 reset through GPIO7...
CoreCell power enable through GPIO18...
CoreCell ADC reset through GPIO13...
Opening SPI communication interface
Note: chip version is 0x10 (v1.0)
INFO: using legacy timestamp
ARB: dual demodulation disabled for all SF
TX done

Nb packets sent: 1 (1)
Closing SPI communication interface
CoreCell reset through GPIO7...
SX1261 reset through GPIO7...
CoreCell power enable through GPIO18...
CoreCell ADC reset through GPIO13...
========== Test End ==========
pi@raspberrypi:~/lorawan/sx1302_hal/bin $
```

2、Configured for use as a gateway

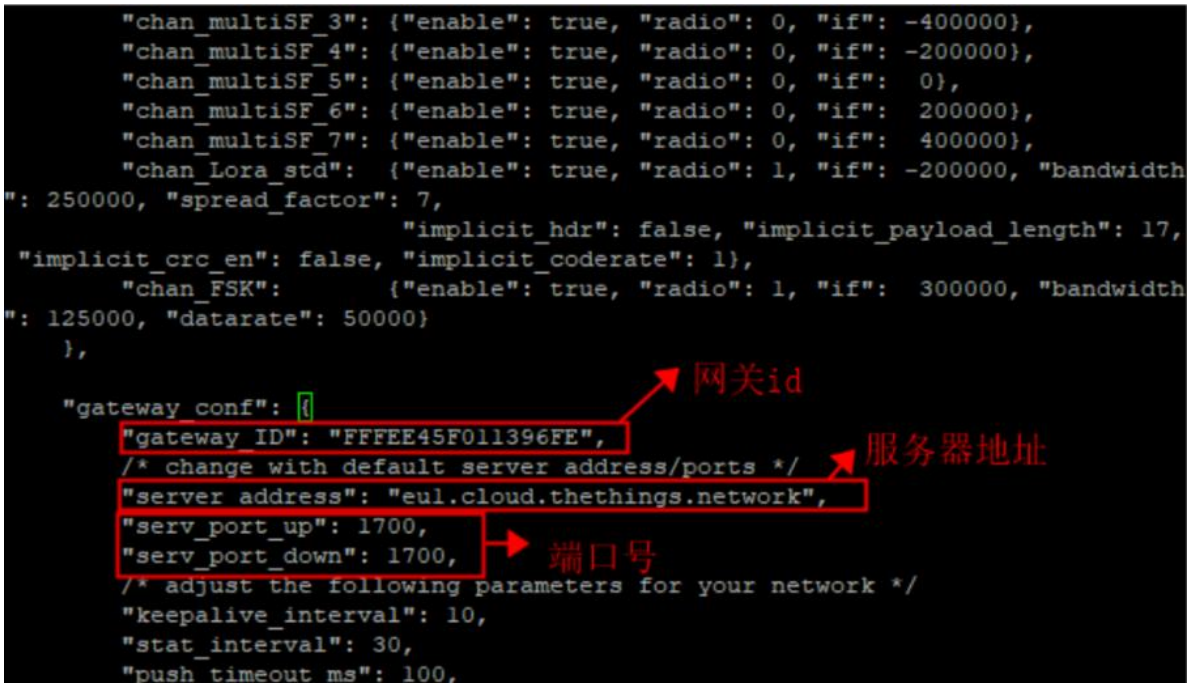
Different regional standards require different JSON files. For example, in this case, we are using the EU868 frequency band and the gateway module uses SPI connection, so we are using the global_conf.json.sx1250.EU868 file.

Edit this file to set the gateway ID and specify the server address.

- $ sudo vim global_conf.json.sx1250.EU868

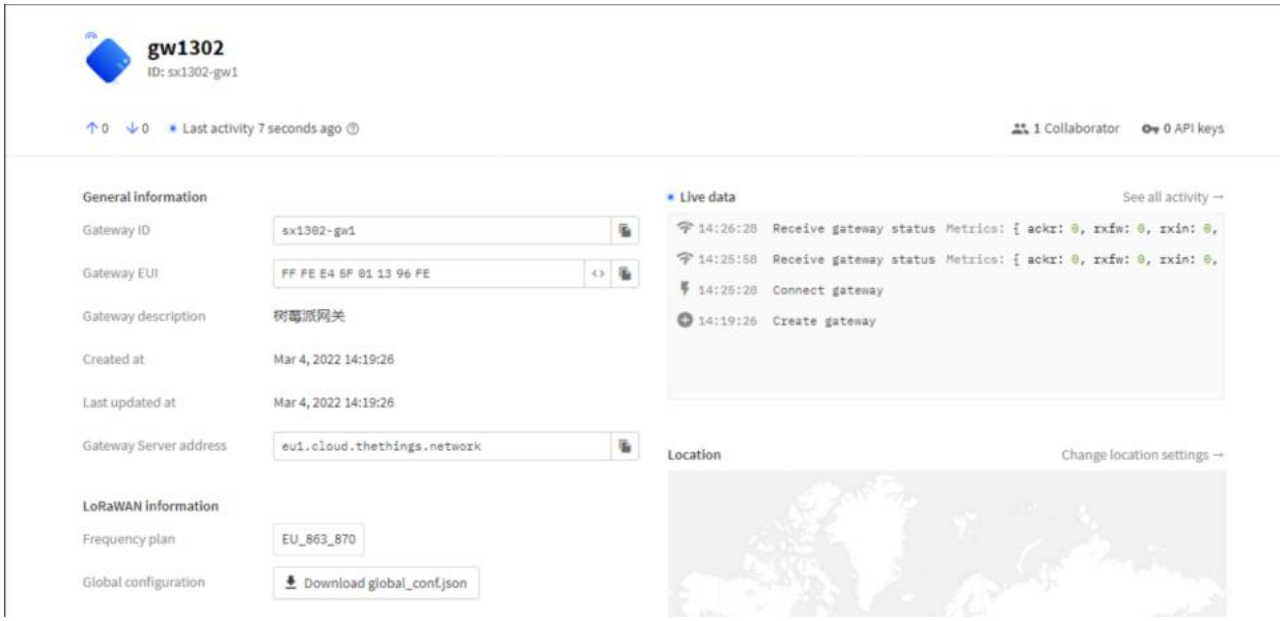
Copyright ©2012–2026, Chengdu Ebyte Electronic Technology Co.,Ltd.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000



After setting it up, enter the following in the sx1302_hal/bin directory:
- $ ./lora_pkt_fwd -c global_conf.json.sx1250.EU868

Here we are using a TTN server and have registered the gateway on it beforehand. When we check the gateway status, we find that it is already online.



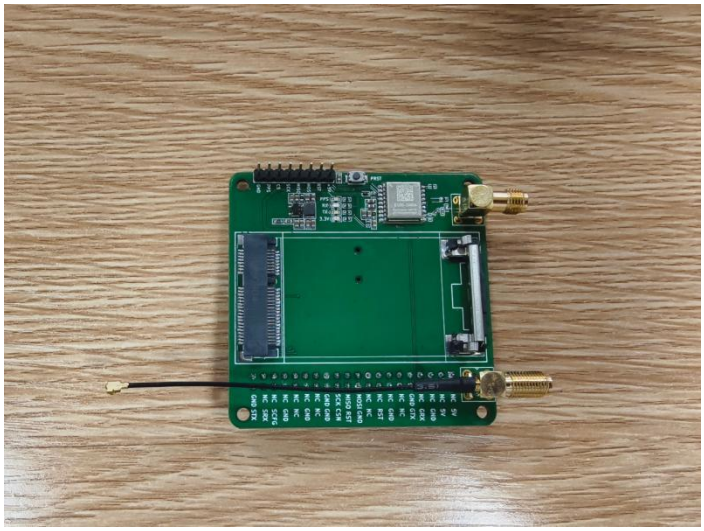
Step 5: SX1303 Fine Timestamping Extension Feature

To use the fine-grained timestamps or time synchronization of the E106- xxx G27P2(SX1303) module, you need to use our E15-LW-T2 adapter board.

To use this function, only the E15-LW-T1 is needed. The E15-LW-T2 adds a GPS module compared to the E15-LW-T1. The E106- xxx G27P2 is an SX1302 and does not have a fine-grained timestamp function.

Copyright ©2012–2026, Chengdu Ebyte Electronic Technology Co.,Ltd.

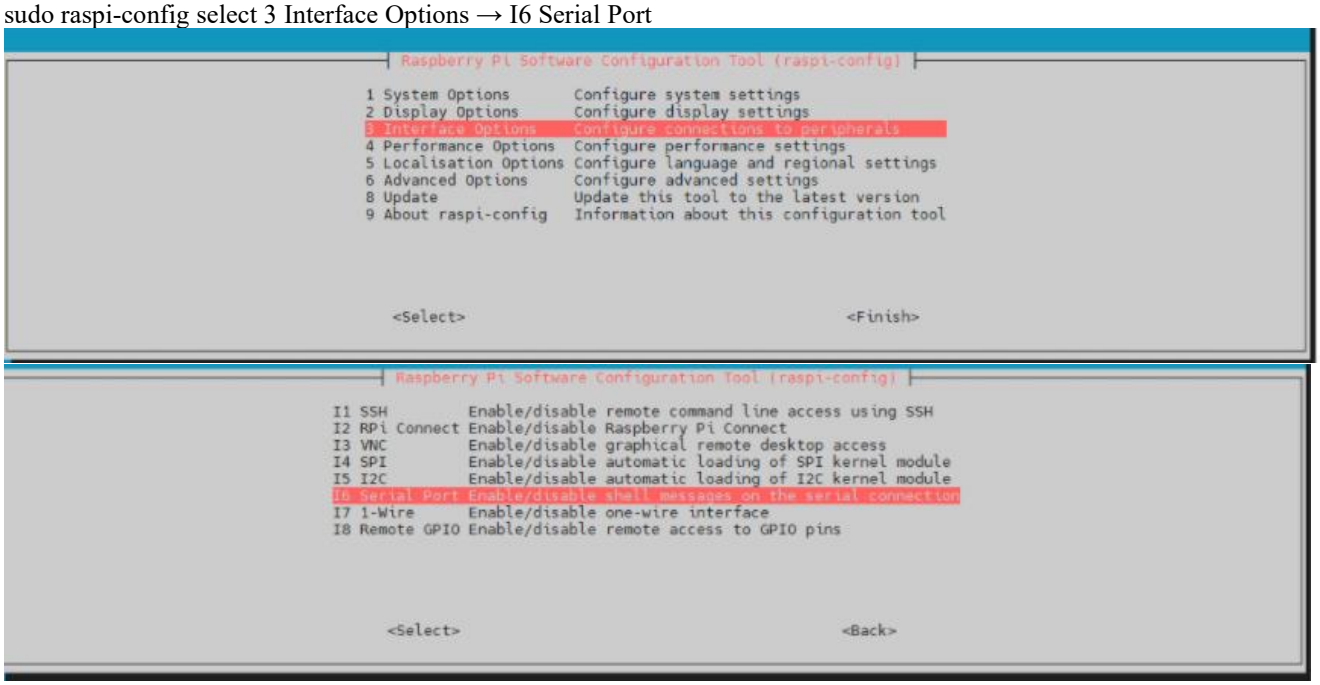
Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000



Note: If you need to obtain GPS time or use a fine timestamp, the GPS antenna needs to be placed in an open outdoor area, and the PPS light needs to flash at 1-second intervals to work properly.

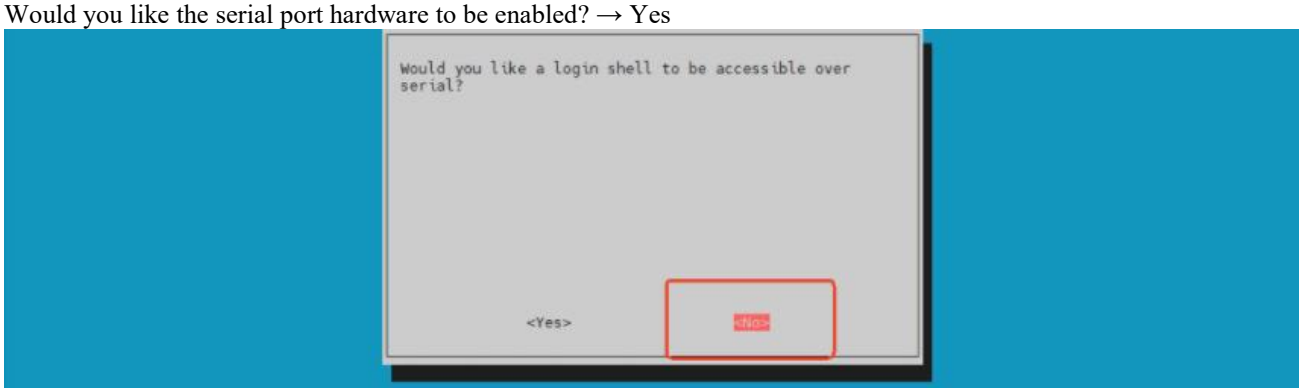
1、Enable serial port on Raspberry Pi

sudo raspi-config select 3 Interface Options → I6 Serial Port



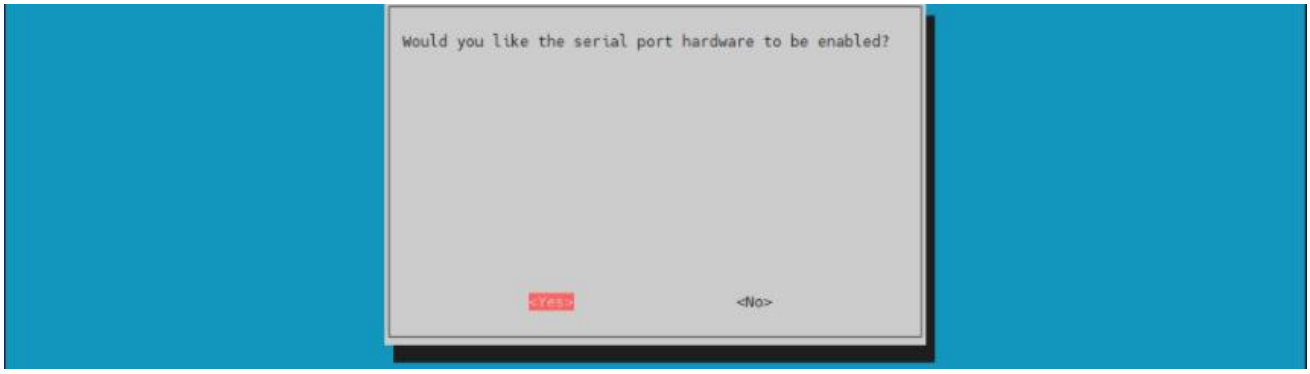
Would you like a login shell that is accessible over serial? → No (NO is mandatory! Disable serial login)

Would you like the serial port hardware to be enabled? → Yes



Copyright ©2012–2026, Chengdu Ebyte Electronic Technology Co.,Ltd.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000



Save and restart

2、Modify the serial port baud rate in the SDK

The GPS module on the E15-LW-T2 uses a baud rate of 38400, while the default baud rate for the GPS serial port in the SDK is 9600. Therefore, [the following is required]:

Modify line 58 in the libloragw/src/loragw_gps.c file..

#define DEFAULT_BAUDRATE B9600 → #define DEFAULT_BAUDRATE B38400

3、Modify the ubx parsing section

The ubx GPS on the E15-LW-T2 is a newer version, which adds one byte to the data frame. Therefore, the ubx data parsing section needs to be modified.

Revise

Line 538 in the loragw_gps.c file

if (nb_fields != 13) {

/* Change to */

if ((nb_fields != 13) && (nb_fields != 14)) {

4、Compilation

- $ make
- $ make install

5、Test fine timestamp

Fine-grained timestamps are primarily reflected in the fact that when receiving LoRa data reports, the data packet will include the `ftime` field. This requires modifying the specific `global_conf.json` file used, changing the `enable` field in `fine_timestamp` from `false` to `true`.

```
{
    "SX130x_conf": {
        "com_type": "SPI",
        "com_path": "/dev/spidev0.0",
        "lorawan_public": true,
        "clksrc": 0,
        "antenna_gain": 0, /* antenna gain, in dBi */
        "full_duplex": false,
        "fine_timestamp": {
            "enable": true,
            "mode": "all_sf" /* high_capacity or all_sf */
        },
```

sudo ./lora_pkt_fwd -c global_conf.json.sx1250.xxx (corresponding frequency) or sudo ./lora_pkt_fwd (in this case, global_conf.json is used) this configuration file.

At this point, when receiving uplink data from the LoRa node, you can see that the data packets contain two additional fields: time and ftime.

```
JSON up: {"rxpk":[{"jver":1,"tmst":24516430,"time":"2026-08-
28T08:12:31.927823Z","tmms":1471939969928,"ftime":707308859,"chan":6,"rfch":1,"freq":471.
500000,"mid":1,"stat":1,"modu":"LORA","datr":"SF12BW125","codr":"4/5","rssis":-119,"lsnr"
:-7.2,"foff":332,"rssi":-112,"size":23,"data":"AAAAAAAAAAA8qk4BRXhgAAg0ErOiW8="}]}
```

Copyright ©2012–2026, Chengdu Ebyte Electronic Technology Co.,Ltd.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000

Revision History

Version	Revision Date	Revision Notes	maintainer
1.0	2023-1-4	Initial version	Luo
1.1	2026-9-3	Revised the reset pin description and added usage instructions for the SX1303 series fine timestamp feature.	Luo

About us

Technical support: support@cdebyte.com
Documents and RF Setting download link: https://www.cdebyte.com
Thank you for using Ebyte products! Please contact us with any questions or suggestions: info@cdebyte.com
---
Web: https://www.cdebyte.com
Address: B5 Mould Industrial Park, 199# Xiqu Ave, High tech Zone, Chengdu, Sichuan, China

Chengdu Ebyte Electronic Technology Co.,Ltd.

Copyright ©2012–2026, Chengdu Ebyte Electronic Technology Co.,Ltd.

Distribuída no Brasil pela LRI | vendas@lri.com.br | 51 2129-3000