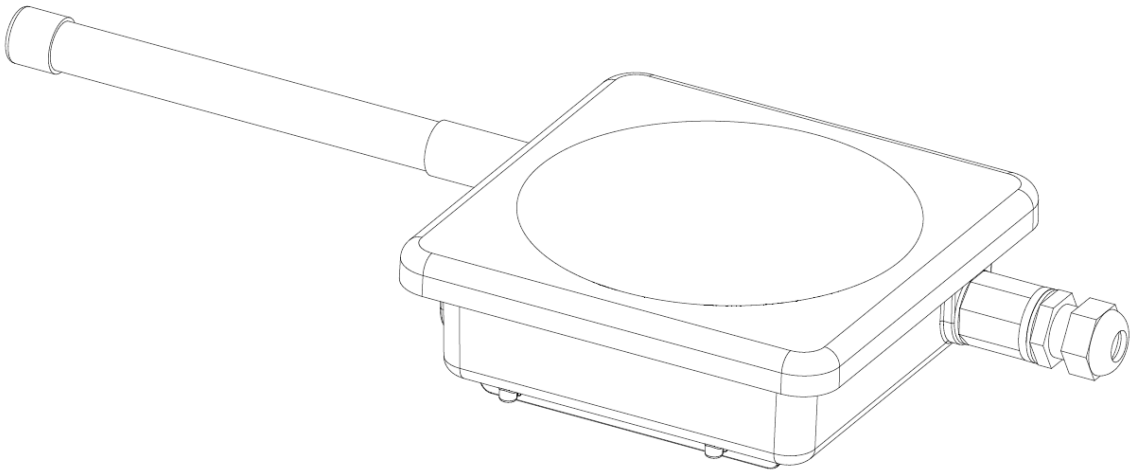


ED-GWL2110

一款基于 Raspberry Pi CM4 设计的室外网关

上海晶珩电子科技有限公司
2024年7月

版权声明

免责声明

上海晶珩电子科技有限公司不保证本手册中的信息是最新的、正确的、完整的或高质量的。上海晶珩电子科技有限公司也不对这些信息的进一步使用作出保证。如果由于使用或不使用本手册中的信息，或由于使用错误或不完整的信息而造成的物质或非物质相关损失，只要没有证明是上海晶珩电子科技有限公司的故意或过失，就可以免除对上海晶珩电子科技有限公司的责任索赔。上海晶珩电子科技有限公司明确保留对本手册的内容或部分内容进行修改或补充的权利，无需特别通知。

目 录

1 产品概述 5
1.1 目标应用 5
1.2 规格参数 5
1.3 系统框图 6
1.4 主板接口 6
1.5 包装清单 7
1.6 订购编码 8
2 快速启动 9
2.1 设备清单 9
2.2 硬件连接 9
2.3 首次启动 9
2.3.1 Raspberry Pi OS (Lite) 9
2.3.2 使能 SSH 功能 10
2.3.3 查找设备 IP 11
3 软件操作指引 11
3.1 按键 11
3.2 LED 指示 11
3.3 以太网配置 12
3.4 LTE 4G（选配） 12
3.5 Wi-Fi 13
3.6 蓝牙 13
3.6.1 基本配置命令 14
3.6.2 配置示例 14
3.7 SD 卡扩展存储 15
3.7.1 挂载 15
3.7.2 卸载 16
3.7.3 命令行中设置自动挂载 16
3.8 RTC 16
3.9 Watch Dog 17
3.10 GNSS 18
3.10.1 引脚配置 18
3.10.2 修改 config.txt 使能串口 18
3.10.3 查看 GNSS 信息 18
3.10.4 使用 u-center 工具查看定位信息 19
3.11 LoRaWAN 21
3.11.1 安装 LoRa 服务和 ChirpStack 客户端 21
3.11.2 配置 LoRa 服务 22
3.11.3 安装 ChirpStack 服务端 23
3.11.4 添加 LoRa 网关和终端 25
3.12 加密芯片 28
4 操作系统安装 28
4.1 镜像下载 29

4.2 系统烧录 29
5 FAQ 29
5.1 默认用户名密码 29
6 关于我们 29
6.1 关于 EDATEC 29
6.2 联系方式 30

1 产品概述

ED-GWL2110 是一款基于 Raspberry Pi CM4 的户外型网关，整机采用全铝合金外盒密封，拥有良好的防雨、防潮、防虫、防雷电性能，支持多种不同频段的 LoRa 模块(需搭配不同频段的外置天线)。支持选配 4G 模块，保证在室外设备可以正常上传下载数据；设备板载 GNSS 模块，可方便能实现定位需求；提供了看门狗模块，可以有效的防止设备出现卡死的情况，大大增加了设备运行的稳定性。ED-GWL2110 还内置加密芯片，挂载在 I2C 总线上，保证设备信息安全；内置 RTC 模块，保证设备可靠性。

1.1 目标应用

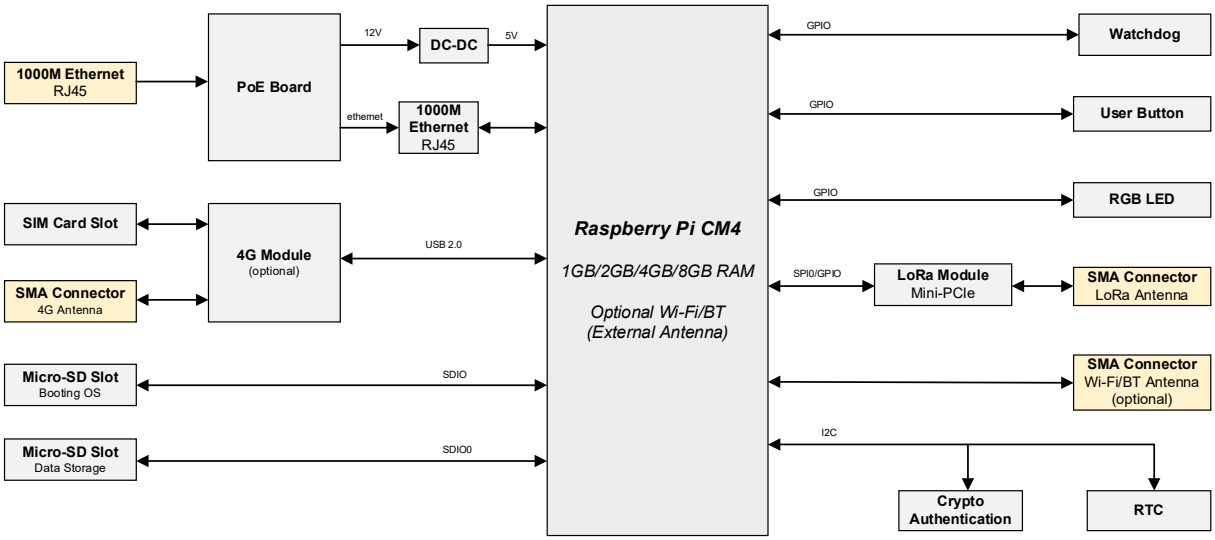
- LoRa 智能网关
- 智慧城市
- 智慧交通
- 智慧农业

1.2 规格参数

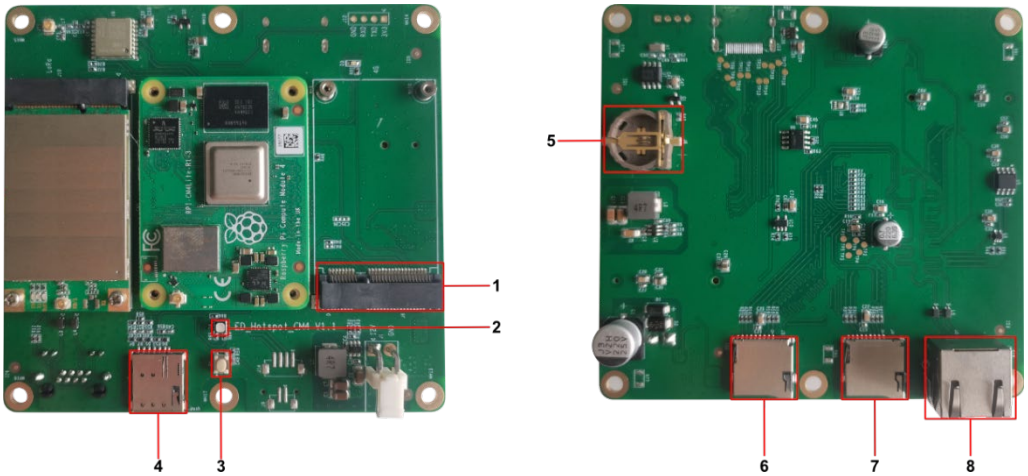
功能	参数
CPU	Broadcom BCM2711 4核ARM Cortex-A72(ARM v8) 64位1.5GHz SoC
内存	1GB/2GB/4GB/8GB可选
SD 卡	32GB/64GB 可选，从 SD 卡启动系统
以太网	1x 千兆以太网口
Wi-Fi/BT（选配）	● 2.4GHz & 5GHz 双频 Wi-Fi, 兼容 IEEE 802.11 b/g/n/ac ● 蓝牙 5.0，支持 BLE
4G（选配）	支持多种 4G LTE 模块
LoRa	兼容 LoRa WAN 协议，支持 3 种频段 ● 868MHz(EU868) ● 915MHz(US915) ● 470MHz(CN470)
GNSS	内置 GNSS 功能，支持多卫星系统： ● GPS L1 C/A：1575.42 ±1.023 MHz ● BeiDou B1I：1561.098 ±2.046 MHz ● GLONASS L1：1597.78~1605.66 MHz
内部 IO	● 1x Serial(TTL)，可用于系统默认控制台 ● 1x 用户自定义按键 ● 1x RGB 三色 LED ● 1x RTC 电池底座，可安装 CR1220 纽扣电池 ● 1x Nano SIM 卡槽 ● 2x Micro-SD 卡槽
扩展功能	● 支持看门狗功能，防止系统卡死 ● 内置加密芯片
电源输入	PoE 供电，支持 802.3af 标准

功能	参数
尺寸	194.2mm(长) x 194.2mm(宽) x 65.3 mm (高)
外壳	铸铝防水外壳，IP65 防水等级
工作温度	-25 ~ 60°C

1.3 系统框图



1.4 主板接口

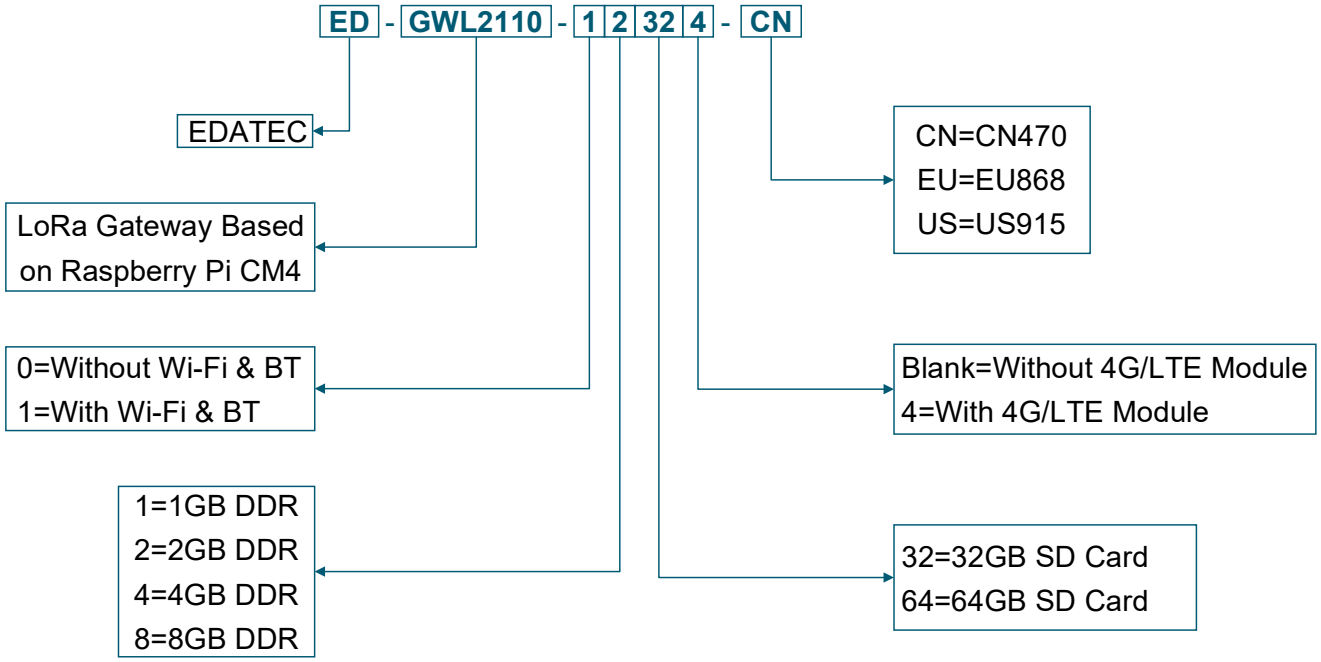


编号	功能描述
1	4G mini-PCIe 接口
2	RGB LED LoRa mini-PCIe 接口
3	用户按键
4	Nano SIM 卡槽
5	RTC 电池底座
6	Micro-SD 卡槽（存储用户数据）
7	Micro-SD 卡槽（系统启动）
8	千兆网口

1.5 包装清单

- 1x ED-GWL2110 主机
- 1x LoRa 天线
- [选配 Wi-Fi/BT 版本]1x 2.4GHz/5GHz Wi-Fi/BT 天线
- [选配 4G 版本]1x 4G/LTE 天线

1.6 订购编码



Example

P/N: **ED-GWL2110-12324-CN**

Configuration: An outdoor light gateway based on Raspberry Pi CM4, with Wi-Fi & Bluetooth, 2GB DDR, 32GB SD card, 4G and CN470 LoRa frequency.

2 快速启动

2.1 设备清单

- 1x ED-GWL2110 主机
- 1x Wi-Fi / BT 外置天线（选配）
- 1x LoRa 外置天线
- 1x 4G 天线（选配）
- 1x 网线

2.2 硬件连接

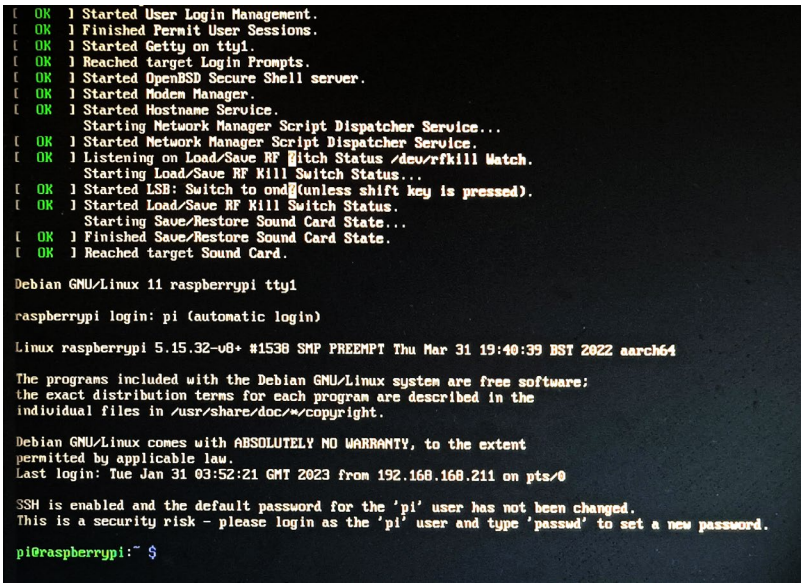
1.安装 Wi-Fi/LoRa/4G 外部天线。
2.插入网线到以太网网口，网线连接可上网的带 PoE 功能路由器/交换机等网络设备。

2.3 首次启动

ED-GWL2110 无电源开关，接入 PoE 供电后，系统将会开始启动。

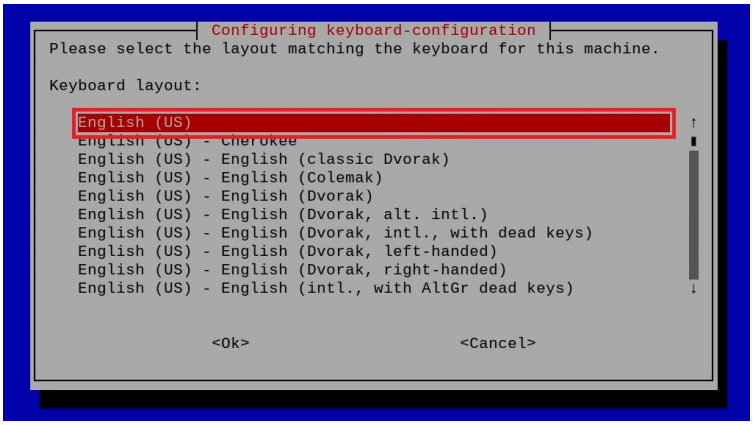
2.3.1 Raspberry Pi OS (Lite)

如果您使用我们提供的系统镜像，系统启动后会使用用户名 pi 自动登入，默认密码为 raspberry。

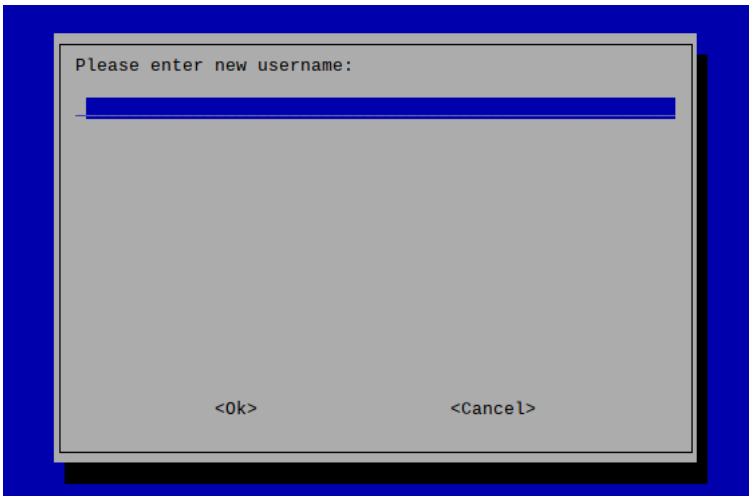


如果您使用官方系统镜像，并且烧录前没有配置镜像，首次启动时，会出现配置窗口，需要依次配置键盘布局，设置用户名及对应密码。

1. 设置配置键盘布局



2. 创建新用户名



然后按提示设置用户对应的密码，并再次输入密码进行确认。至此您就可以使用刚才设置的用户名及密码进行登入了。

2.3.2 使能 SSH 功能

出厂默认的镜像已使能 SSH 功能，如果使用官方镜像则需要参考如下步骤使能 SSH。

2.3.2.1 通过 raspi-config 命令使能 SSH

1. 执行sudo raspi-config命令。
2. 选择 3 Interface Options。
3. 选择 I2 SSH。
4. Would you like the SSH server to be enabled? 选 Yes
5. 选择右下角 Finish。

2.3.2.2 添加空文件使能 SSH

在 boot 分区中创建一个名为 ssh 的空文件，设备上电后将会自动使能 SSH 功能。

2.3.3 查找设备 IP

- 设备启动后，如果已连接显示屏，则可以使用ifconfig 命令查看当前设备IP。
- 设备启动后，如果没有连接显示屏，则可以通过路由器查看设备被分配的IP。
- 设备启动后，如果没有连接显示屏，则可以下载Nmap工具扫描当前网络下的IP。

  nmap 支持 Linux、macOS、Windows 等多个平台。如果希望使用 nmap 扫描 192.168.3.0~255 的网段，则可以使用以下命令：

```
nmap -sn 192.168.3.0/24
```

  等待一段时间后即会输出结果，类似与下方输出：

```
Starting Nmap 7.92 ( https://nmap.org ) at 2022-12-30 21:19 中国标准时间
Nmap scan report for 192.168.3.1 (192.168.3.1)
Host is up (0.0010s latency).
MAC Address: XX:XX:XX:XX:XX:XX (Phicomm (Shanghai))
Nmap scan report for DESKTOP-FGEOUUK.lan (192.168.3.33)
Host is up (0.0029s latency).
MAC Address: XX:XX:XX:XX:XX:XX (Dell)
Nmap scan report for 192.168.3.66 (192.168.3.66)
Host is up.
Nmap done: 256 IP addresses (3 hosts up) scanned in 11.36 seconds
```

3 软件操作指引

3.1 按键

ED-GWL2110 包含 1 个用户自定义按键，在设备内部，连接到 CPU 的 GPIO23 管脚，默认状态下为高电平，当按键按下时，该管脚为低电平。

可使用 raspi-gpio 命令进行测试。

◆ 按键未按下时查询 GPIO23 管脚

```
raspi-gpio get 23
GPIO 23: level=1 fsel=0 func=INPUT
```

level 为 1 表示 GPIO23 管脚为高电平。

◆ 按键按下时，查询 GPIO23 管脚

```
raspi-gpio get 23
GPIO 23: level=0 fsel=0 func=INPUT
```

level 为 0 表示 GPIO23 管脚为低电平。

3.2 LED 指示

ED-GWL2110 包含一个 RGB 三色 LED 指示灯，相连的 GPIO 管脚对应如下：

RGB LED 管脚	对应 GPIO
蓝色	GPIO16
绿色	GPIO20
红色	GPIO21

GPIO 输出为低时，对应 LED 有效。
我们使用 raspi-gpio 命令进行操作，配置参数为 op 表示设置为输出，dl 设置管脚为低电平，dh 设置管脚为高电平。

LED 显示为蓝色

```
sudo raspi-gpio set 16 op dl
sudo raspi-gpio set 20 op dh
sudo raspi-gpio set 21 op dh
```

LED 显示为绿色

```
sudo raspi-gpio set 16 op dh
sudo raspi-gpio set 20 op dl
sudo raspi-gpio set 21 op dh
```

LED 显示为红色

```
sudo raspi-gpio set 16 op dh
sudo raspi-gpio set 20 op dh
sudo raspi-gpio set 21 op dl
```

LED 显示为黄色

```
sudo raspi-gpio set 16 op dh
sudo raspi-gpio set 20 op dl
sudo raspi-gpio set 21 op dl
```

3.3 以太网配置

ED-GWL2110 包含一路自适应 10/100/1000M 以太网接口。

Raspberry Pi 官方系统默认使用的 dhcpcd 作为网络管理工具。

设置静态 IP 通过修改/etc/dhcpcd.conf 来设置，示例设置 eth0，用户可以根据自己的不同需要设置 wlan0 等网络接口。

```
interface eth0
static ip_address=192.168.0.10/24
static routers=192.168.0.1
static domain_name_servers=192.168.0.1 8.8.8.8 fd51:42f8:caae:d92e::1
```

3.4 LTE 4G（选配）

在使用 4G 之前首先需要添加我司 APT 库，我司给出的镜像均已添加此库，用户无需再手动添加。

```
sudo apt update
sudo apt install ed-ec20-qmi
```

默认不启动自动拨号，如果用户希望开机自动拨号则需要使能 lte-reconnect.service 服务

```
sudo systemctl enable lte-reconnect.service
sudo systemctl start lte-reconnect.service
```

拨号成功后可以使用 ifconfig 命令看到出现 wwan0 网口。

如果需要额外设置 APN，则需要修改/usr/share/ed-ec20-qmi/lte-reconnect.sh 中的拨号命令

```
$BSP_HOME_PATH/quectel-CM -4 -f $LOGFILE &
```

其中 quectel-CM 拨号配置信息如下：

```
$BSP_HOME_PATH/quectel-CM -4 -f $LOGFILE -s <APN> &
```

设置完成后重启 lte-reconnect.service

```
sudo systemctl restart lte-reconnect.service
```

复位 4G 模组

```
raspi-gpio set 10 pd
raspi-gpio set 10 op dl
sleep 0.5
raspi-gpio set 10 dh
sleep 0.5
raspi-gpio set 10 dl
```

3.5 Wi-Fi

ED-GWL2110 支持 2.4GHz&5GHz IEEE 802.11 b/g/n 双频 Wi-Fi。

Raspberry Pi 官方系统默认使用的 dhcpcd 作为网络管理工具。

1. 执行sudo raspi-config。
2. 选择 1 System Options。
3. 选择 S1 Wireless LAN。
4. 在 Select the country in which the Pi is to be used 窗口中选择您的国家，然后选择 OK，此提示仅在第一次设置 WIFI 时出现。
5. Please enter SSID，输入 WIFI SSID 名称。
6. Please enter passphrase. Leave it empty if none，输入密码，然后重启设备即可。

3.6 蓝牙

ED-GWL2110 支持蓝牙 5.0，同时支持蓝牙低功耗（BLE），蓝牙功能默认是开启的。

可使用 bluetoothctl 扫描，配对，连接蓝牙设备，请参考 ArchLinux-Wiki-Bluetooth 指引配置和使用蓝牙。

3.6.1 基本配置命令

命令	功能说明
bluetoothctl scan on	开启蓝牙扫描
bluetoothctl scan off	关闭蓝牙扫描
bluetoothctl discoverable on	开启蓝牙发现（可以让对方发现）
bluetoothctl discoverable off	关闭蓝牙发现
bluetoothctl trust *device_MAC*	信任设备
bluetoothctl connect *device_MAC*	连接设备
bluetoothctl disconnect *device_MAC*	与设备断开连接

3.6.2 配置示例

通过配置示例来介绍配置蓝牙的具体操作。

前提条件：

- 已开启蓝牙扫描。
- 已开启蓝牙发现。

操作步骤：

1. 进入 bluetootch 视图；

   sudo bluetoothctl

2. 使能 bluetooth；

   power on

3. 扫描蓝牙设备；

   scan on

   回显信息：
   Discovery started
   [CHG] Controller B8:27:EB:85:04:8B Discovering: yes
   [NEW] Device 4A:39:CF:30:B3:11 4A-39-CF-30-B3-11

4. 查找开启的蓝牙设备名称；

   devices

   回显信息：
   Device 6A:7F:60:69:8B:79 6A-7F-60-69-8B-79
   Device 67:64:5A:A3:2C:A2 67-64-5A-A3-2C-A2
   Device 56:6A:59:B0:1C:D1 Lefun
   Device 34:12:F9:91:FF:68 test

5. 配对目标设备；

   pair *34:12:F9:91:FF:68*

   34:12:F9:91:FF:68 为目标设备的 *device_MAC*

   回显信息：

   Attempting to pair with 34:12:F9:91:FF:68
   [CHG] Device 34:12:F9:91:FF:68 ServicesResolved: yes
   [CHG] Device 34:12:F9:91:FF:68 Paired: yes
   Pairing successful

6. 添加为信任设备

   trust *34:12:F9:91:FF:68*

   34:12:F9:91:FF:68 为目标设备的 *device_MAC*

   回显信息：

   [CHG] Device 34:12:F9:91:FF:68 Trusted: yes
   Changing 34:12:F9:91:FF:68 trust succeeded

3.7 SD 卡扩展存储

3.7.1 挂载

提示：

桌面版系统支持自动挂载存储设备，仅 Lite 版本系统需要手动挂载。

您可以将存储设备安装在特定的文件夹位置。通常在/mnt 文件夹中进行，例如/mnt/mydisk。请注意，文件夹必须是空的。

1. 将 SD 卡插入设备上的 SD 卡槽。
2. 执行以下命令列出所有磁盘分区:

   ```
   sudo lsblk -o UUID,NAME,FSTYPE,SIZE,MOUNTPOINT,LABEL,MODEL
   ```

   Raspberry Pi 使用挂载点/和/boot。您的存储设备将显示在此列表中，以及任何其他连接的存储设备。
3. 使用“大小”、“标签”和“型号”列来标识指向您的存储设备的磁盘分区的名称。例如 sda1。
4. 运行以下命令获取磁盘分区的位置:

   ```
   sudo blkid
   ```

   比如显示: /dev/sda1
5. 创建一个目标文件夹作为存储设备的装载点。本例中使用的挂载点名称是 mydisk。您可以指定自己选择的名称:

   ```
   sudo mkdir /mnt/mydisk
   ```
6. 在您创建的装载点装载存储设备:

   ```
   sudo mount /dev/sda1 /mnt/mydisk
   ```

7. 通过列出以下内容来验证存储设备是否已成功装载:

```
ls /mnt/mydisk
```

3.7.2 卸载

当设备关闭时，系统会负责卸载存储设备，以便安全地将其拔出。如果您想要手动卸载设备，可以使用以下命令:

```
sudo umount /mnt/mydisk
```

如果您收到“目标繁忙”的错误，这意味着存储设备未卸载。如果没有显示错误，您现在可以安全地拔出设备。

3.7.3 命令行中设置自动挂载

可以通过修改 fstab 设置自动挂载。

1. 首先需要获取磁盘 UUID

```
sudo blkid
```

2. 找到挂载设备的 UUID，例如 5C24-1453
3. 打开 fstab 文件

```
sudo nano /etc/fstab
```

4. 添加以下内容到 fstab 文件中

```
UUID=5C24-1453 /mnt/mydisk fstype defaults,auto,users,rw,nofail 0 0
```

将 fstype 替换为您的文件系统的类型，您可以在上面的“挂载存储设备”的步骤 2 中找到，例如:ntfs。

5. 如果文件系统类型是 FAT 或 NTFS，则在 nofail 之后立即添加 umask = 000 这将允许所有用户对存储设备上的每个文件进行完全读/写访问。

关于更多 fstab 命令的信息可以使用 man fstab 来查看。

3.8 RTC

默认出货系统镜像，会集成我们编写的 RTC 自动同步服务，客人无需设置，即可自动同步时钟，可无感使用 RTC。大概的原理是：

1. 系统开机时，服务自动从 RTC 读出保存的时间，并同步到系统时间。
2. 若有连接互联网，系统会自动从 NTP 服务器同步时间, 使用互联网时间更新本地系统时间。
3. 系统关机时，服务自动把系统时间写入 RTC, 更新 RTC 的时间。
4. 因为有安装纽扣电池，尽管设备下电，但是 RTC 仍在工作计时。

这样，可以保证我们的时间是准确可靠的。

WARN：若是第一次开机，因为 RTC 中无有效时间，可能会同步失败，直接重启即可。重启的时候，会把系统时间写入 RTC，后续正常使用。

若您不想用此服务, 可手动关闭：

```
sudo systemctl disable rtc
sudo reboot
```

重新使能此服务：

```
sudo systemctl enable rtc
sudo reboot
```

手动读取 RTC 的时间：

```
sudo hwclock -r
2022-11-09 07:07:30.478488+00:00
```

手动同步 RTC 时间到系统：

```
sudo hwclock -s
```

把系统时间写入 RTC：

```
sudo hwclock -w
```

问题排查

请首先看是否有 rtc 设备(/dev/rtc0)加载:

```
ls /dev/rtc0
```

如果没有，可能是您使用了官方标准系统，但是没有安装我们的 BSP 包，请参考章节: "通过 apt-get 安装 BSP 包"安装 BSP，另外, 您同样需要安装 ed-rtc 包使能 RTC 自动同步功能.

其他可能的检查点:

- CR1220 纽扣电池有没有安装
- NTP 网络时间协议, 需要连接互联网才可自动同步时间, 另外, 需要开放端口(UDP, 123), 否则同步失败

3.9 Watch Dog

ED-GWL2110 配有看门狗模块，可以防止系统卡死。

看门狗逻辑表

GPIOx	pin	H/L	H/L	H/L
GPIO17	OE	H	H	L
GPIO16	A	H	L	X
output	Y	H	L	Z

使用看门狗模块需要安装 ed-gwl2100-wdt.dtbo，文件链接：todo
用户下载下来后需要放在设备上，可使用 scp 命令将文件拷贝至设备目录：

```
scp /path/ed-gwl2100-wdt.dtbo pi@ip-address:/home/pi
sudo cp /home/pi/ed-gwl2100-wdt.dtbo /boot/overlays
sudo chmod +x /boot/overlays/ed-gwl2100-wdt.dtbo
```

并在/boot/config.txt 末尾添加以下内容：

```
sudo nano /boot/config.txt
```

```
dtoverlay=ed-gwl2100-wdt
```

3.10 GNSS

ED-GWL2110 网关集成 L76K GPS 模块，其与 CPU 的 UART0 串口相连。模块通过 NMEA 0183 通用协议输出语句上报 GNSS 信息。

3.10.1 引脚配置

L76K GPS 模块的 WakeUp 信号与 GPIO4 相连，拉低该管脚模块将进入待机模式，拉高或悬空该管脚模块将返回连续模式。Reset 信号与 GPIO5 相连，拉低该管脚并持续 100ms 将复位模块。SET 信号与 GPIO6 相连，用于配置卫星组合，当该管脚为悬空或高电平时，卫星组合为 GPS 和北斗，当管脚为低电平时，卫星组合为 GPS 和 GLONASS。

#	Signal	CM4 Pinout
1	GPS_WakeUp	GPIO4
2	GPS_Reset	GPIO5
3	GPS_Set	GPIO6

3.10.2 修改 config.txt 使能串口

```
sudo nano /boot/config.txt
```

在最后面添加

```
[all]
enable_uart=1
```

3.10.3 查看 GNSS 信息

```
sudo cat /dev/ttyS0
```

显示 GPS 数据如下：

```
$BDGSV,3,1,11,04,29,117,20,10,,,19,16,75,160,,24,51,328,,0*4C
$BDGSV,3,2,11,25,,,27,26,,,21,34,12,198,,35,45,063,,0*76
$BDGSV,3,3,11,39,62,159,17,41,,,25,59,44,137,,0*7A
$GNRMC,053557.000,A,3027.47401,N,11424.34027,E,1.17,186.64,070223,,,A,V*05
$GNVTG,186.64,T,,M,1.17,N,2.17,K,A*2D
$GNZDA,053557.000,07,02,2023,00,00*4F
$GPTXT,01,01,01,ANTENNA OPEN*25
$GNGGA,053558.000,3027.47438,N,11424.34119,E,1,07,1.5,75.0,M,-14.1,M,,*52
$GNGLL,3027.47438,N,11424.34119,E,053558.000,A,A*4F
$GNGSA,A,3,07,08,16,31,195,,,,,,,,2.1,1.5,1.5,1*05
$GNGSA,A,3,04,39,,,,,,,,,,,,2.1,1.5,1.5,4*39
$GPGSV,3,1,12,04,54,241,16,07,19,314,15,08,63,208,15,09,38,291,,0*67
$GPGSV,3,2,12,16,51,029,17,18,07,046,,21,08,175,,26,24,063,,0*6A
```

```
$GPGSV,3,3,12,27,77,065,,31,09,122,22,194,61,058,,195,46,125,21,0*66
```

NMEA 0183 通用语句说明如下：
$BDGSV 可视北斗卫星信息
$GNRMC 推荐的 GNSS 数据
$GNVTG 相对地面航向和速度信息
$GNZDA 时间和日期，UTC 格式
$GPTXT 文本传送
$GNGGA 多星联合定位数据
$GNGLL 地理位置纬度和经度
$GNGSA 表示 GNSS 精度因子和有效卫星
$GPGSV 可视的 GNSS 卫星

3.10.4 使用 u-center 工具查看定位信息

3.10.4.1 安装串口转网络工具 ser2net

```
sudo apt-get update
sudo apt-get install ser2net
```

启用 ser2net 服务
ser2net 配置文件为/etc/ser2net.yaml，默认已配置/dev/ttyS0，波特率为 9600，无校验，对应的 TCP 端口为 2000。

```
connection: &con0096
   accepter: tcp,2000
   enable: on
   options:
     banner: *banner
     kickolduser: true
     telnet-brk-on-sync: true
   connector: serialdev,
            /dev/ttyS0,
            9600n81,local
```

3.10.4.2 检查 ser2net 端口转发服务

使用如下指令查询 ser2net 是否已启动 2000 端口转发

```
sudo netstat -ltnp | grep 2000
```

如果已启动端口转发，将显示如下信息

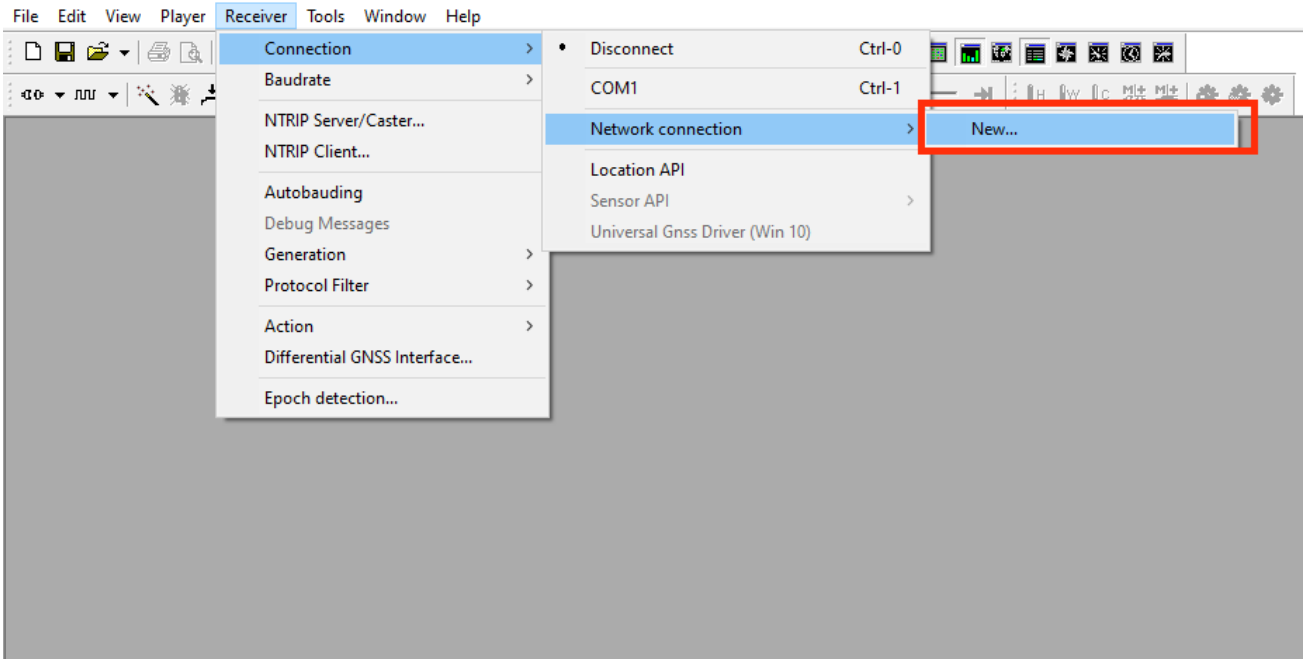
```
tcp6       0      0 :::2000                 :::*                    LISTEN      720/ser2net
```

如果无显示，则重启 ser2net 服务

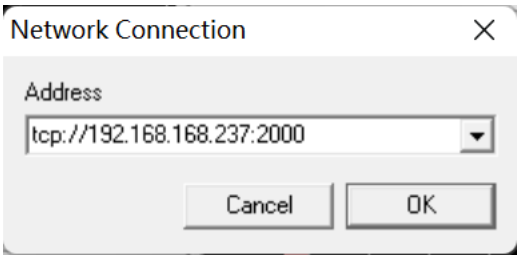
```
sudo systemctl restart ser2net
```

下载 u-center 工具并安装，如果提示缺少 MSVCR120.dll 文件，请安装 vcredist_x86.exe。

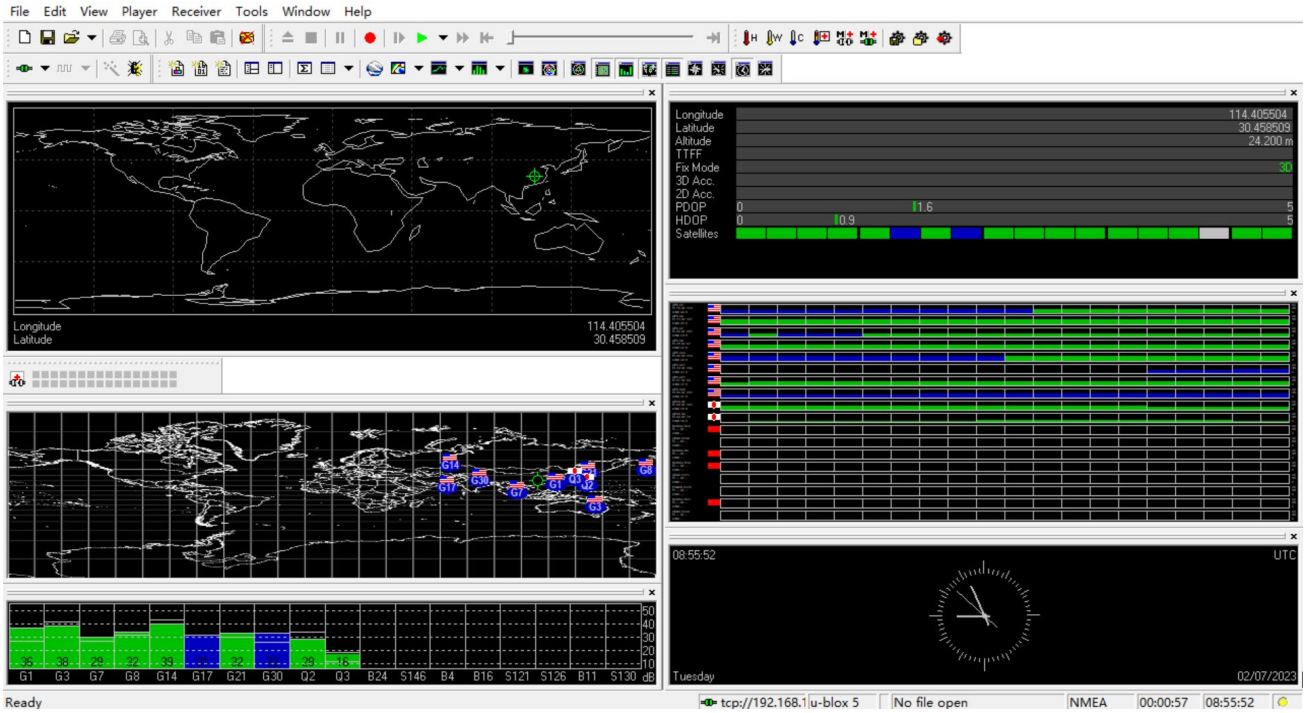
打开 u-center，选择 Receiver->Port->Network connection->New…



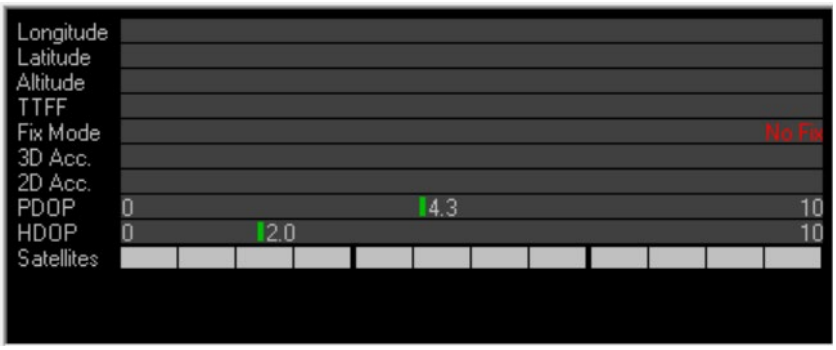
输入您的设备 IP 和端口号 2000。



配置完成后会立刻看到 GPS 定位信息



如果 Fix Mode 显示为 No Fix，表示未能成功定位，一般是由于天线在室内的原因造成，请将模块或天线置于户外进行试验。



NOTE:模块首次定位，在户外没有大型建筑遮挡的情况下，大概需要 30 秒才能定位成功，如果天气条件恶劣，可能需要更长的定位时间或无法定位。

3.11 LoRaWAN

ED-GWL2110 支持 LoRaWAN 开源服务平台 ChirpStack，请参考如下步骤进行安装和配置。

3.11.1 安装 LoRa 服务和 ChirpStack 客户端

我们通过 APT 的方式进行安装。

- 添加 edatec APT 仓库

```
$ curl -sS https://apt.edatec.cn/pubkey.gpg | sudo apt-key add -
$ echo "deb https://apt.edatec.cn/raspbian stable main" | sudo tee /etc/apt/sources.list.d/edatec.list
$ sudo apt update
$ sudo apt install -y ed-gwl-pktfwd
```

- 安装 ChirpStack

```
$ sudo apt install -y apt-transport-https dirmngr
$ sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 1CE2AFD36DBCCA00
$ echo "deb https://artifacts.chirpstack.io/packages/4.x/deb stable main" | sudo tee
/etc/apt/sources.list.d/chirpstack.list
$ sudo apt update

$ sudo apt install -y chirpstack-gateway-bridge
```

- 修改 config.txt

```
[all]
dtparam=i2c_arm=on
dtparam=spi=on

gpio=16=op,dl
gpio=20=op,dl
gpio=21=op,dl
```

修改/etc/modules，在最后面添加 i2c-dev

```
i2c-dev
```

ED-GWL2110 使用 i2c-1 和 spidev0.0。

3.11.2 配置 LoRa 服务

3.11.2.1 pktfwd 配置

```
# update region
$ cat /etc/ed_gwl/region
EU868   # EU868 / US915
```

pktfwd 使用 1700 作为 UDP 端口

```
$ sudo systemctl restart ed-pktfwd.service
```

3.11.2.2 chirpstack-gateway-bridge 的配置

您可以使用 nano 进行配置文件 chirpstack-gateway-bridge.toml 的编辑

```
$ sudo nano /etc/chirpstack-gateway-bridge/chirpstack-gateway-bridge.toml
```

```
# This configuration provides a Semtech UDP packet-forwarder backend and
 # integrates with a MQTT broker. Many options and defaults have been omitted
 # for simplicity.
 #
 # See https://www.chirpstack.io/gateway-bridge/install/config/ for a full
 # configuration example and documentation.


 # Gateway backend configuration.
 [backend]
 # Backend type.
 type="semtech_udp"

   # Semtech UDP packet-forwarder backend.
   [backend.semtech_udp]

   # ip:port to bind the UDP listener to
   #
   # Example: 0.0.0.0:1700 to listen on port 1700 for all network interfaces.
   # This is the listener to which the packet-forwarder forwards its data
   # so make sure the 'serv_port_up' and 'serv_port_down' from your
   # packet-forwarder matches this port.
   udp_bind = "0.0.0.0:1700"

 # Integration configuration.
 [integration]
 # Payload marshaler.
 #
 # This defines how the MQTT payloads are encoded. Valid options are:
 # * protobuf:  Protobuf encoding
 # * json:      JSON encoding (easier for debugging, but less compact than 'protobuf')
 marshaler="protobuf"
```

```
# MQTT integration configuration.
[integration.mqtt]
# Event topic template.
event_topic_template="eu868/gateway/{{ .GatewayID }}/event/{{ .EventType }}"

# Command topic template.
command_topic_template="eu868/gateway/{{ .GatewayID }}/command/#"

# MQTT authentication.
[integration.mqtt.auth]
# Type defines the MQTT authentication type to use.
#
# Set this to the name of one of the sections below.
type="generic"

  # Generic MQTT authentication.
  [integration.mqtt.auth.generic]
  # MQTT server (e.g. scheme://host:port where scheme is tcp, ssl or ws)
  server="tcp://127.0.0.1:1883"

  # Connect with the given username (optional)
  username=""

  # Connect with the given password (optional)
  password=""
```

- 'event_topic_template / command_topic_template' 需要修改带有网关区域的前缀。

  Example:

  ```
  event_topic_template="eu868/gateway/{{ .GatewayID }}/event/{{ .EventType }}"
  ```

  如果您使用 US915 或 CN470 模块请将前缀 eu868 修改为 us915_0 / cn470_10

  ```
  event_topic_template="us915_0/gateway/{{ .GatewayID }}/event/{{ .EventType }}"
  ```

- integration.mqtt 服务器地址需要是您的 chirpstack 服务器

  ```
  $ sudo systemctl restart chirpstack-gateway-bridge.service
  ```

修改 chirpstack-gateway-bridge.toml 配置后，重启 chirpstack-gateway-bridge 服务生效。

3.11.2.3 Reboot

```
$ sudo reboot
```

3.11.3 安装 ChirpStack 服务端

配置云端服务器，配置之前首先需要在服务器安装 docker。

安装 docker：https://docs.docker.com/get-docker/

安装 docker-compose

```
sudo apt install docker-compose
```

3.11.3.1 配置 chirpstack-docker

我们采用 docker 容器的方式部署 ChirpStack 服务端。

```
$ git clone https://github.com/chirpstack/chirpstack-docker.git
```

需要对 chirpstack-docker 的 docker-compose.yml 进行配置。

```
$ cd chirpstack-docker
$ nano docker-compose.yml
# Remove the chirpstack-gateway-bridge, because we run the bridge on gateway.
```

删除红色字体部分。

```
$ nano docker-compose.yml

version: "3"

services:
  chirpstack:
    image: chirpstack/chirpstack:4
    command: -c /etc/chirpstack
    restart: unless-stopped
    volumes:
      - ./configuration/chirpstack:/etc/chirpstack
      - ./lorawan-devices:/opt/lorawan-devices
    depends_on:
      - postgres
      - mosquitto
      - redis
    environment:
      - MQTT_BROKER_HOST=mosquitto
      - REDIS_HOST=redis
      - POSTGRESQL_HOST=postgres
    ports:
      - 8080:8080

  chirpstack-gateway-bridge-eu868:
    image: chirpstack/chirpstack-gateway-bridge:4
    restart: unless-stopped
    ports:
      - 1700:1700/udp
    volumes:
      - ./configuration/chirpstack-gateway-bridge:/etc/chirpstack-gateway-bridge
    depends_on:
      - mosquitto

  chirpstack-rest-api:
    image: chirpstack/chirpstack-rest-api:4
    restart: unless-stopped
    command: --server chirpstack:8080 --bind 0.0.0.0:8090 --insecure
    ports:
      - 8090:8090
    depends_on:
      - chirpstack

  postgres:
    image: postgres:14-alpine
```

```
    restart: unless-stopped
    volumes:
      - ./configuration/postgresql/initdb:/docker-entrypoint-initdb.d
      - postgresqldata:/var/lib/postgresql/data
    environment:
      - POSTGRES_PASSWORD=root

  redis:
    image: redis:7-alpine
    restart: unless-stopped
    volumes:
      - redisdata:/data

  mosquitto:
    image: eclipse-mosquitto:2
    restart: unless-stopped
    ports:
      - 1883:1883
    volumes:
      - ./configuration/mosquitto/mosquitto.conf:/mosquitto/config/mosquitto.conf

volumes:
  postgresqldata:
  redisdata:
```

启动 chirpstack 服务端

```
$ docker-compose up -d
```

3.11.3.2 登录 chirpstack 服务管理界面

在 PC 端浏览器输入服务器的 IP 地址和端口号 8080，在网络正常的情况下会出现登录界面
默认管理员用户名和密码如下：

```
user: admin
psw : admin
```

3.11.4 添加 LoRa 网关和终端

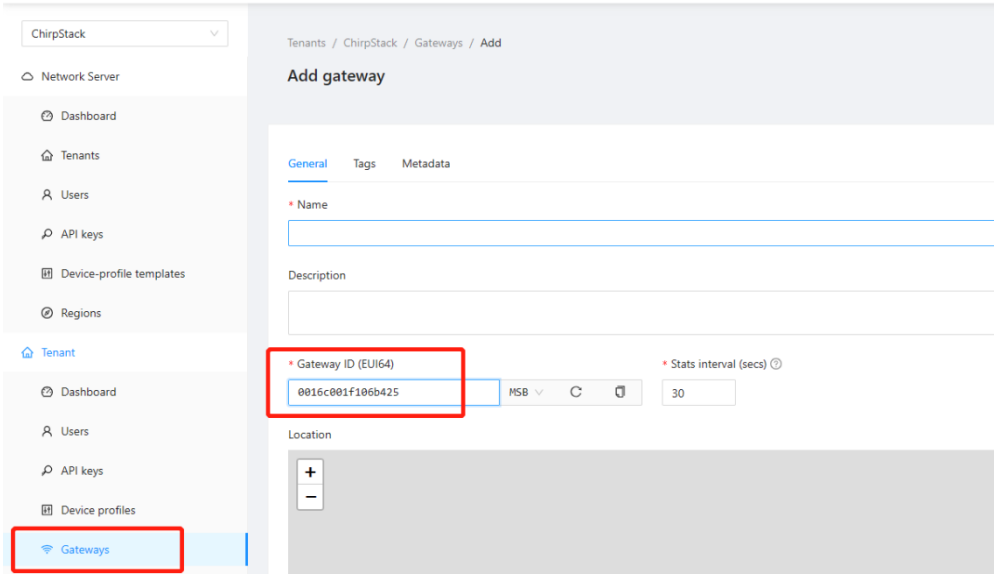
3.11.4.1 获取 LoRa 网关 ID

执行如下指令即可获得 LoRa 网关的 ID，在向 chirpstack 服务端添加 LoRa 网关时，需要填加对应的网关 ID。

```
$ /opt/ed-gwl-pktfwd/ed-gateway_id
```

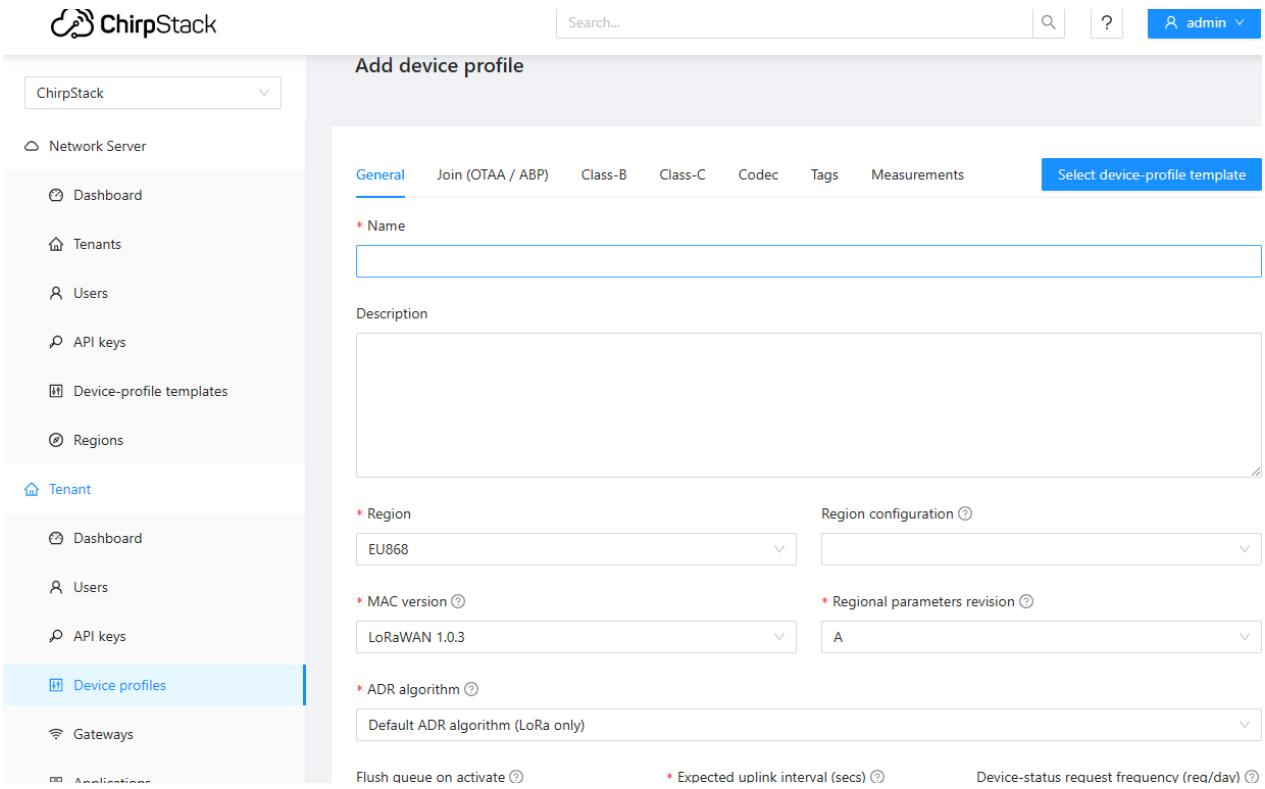
3.11.4.2 添加 LoRa 网关

在 PC 端浏览器打开 chirpstack 管理界面，点击 Gateway -> Add gateway，填入设备对应的 Gateway ID，并设置 Name，然后点击 Submit，如果网络连接正确，稍等片刻即可看到添加的网关变为 Online 状态。



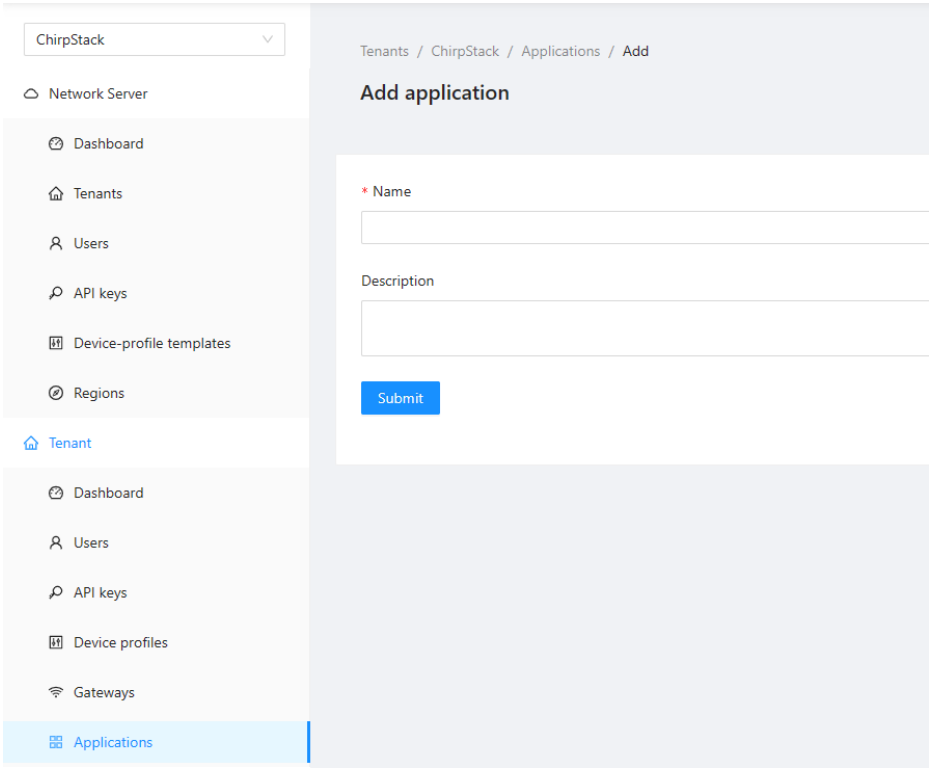
3.11.4.3 Add Device Profile

点击 Device Profile -> Add device profile 可以进一步完善设备信息。



3.11.4.4 Add Application

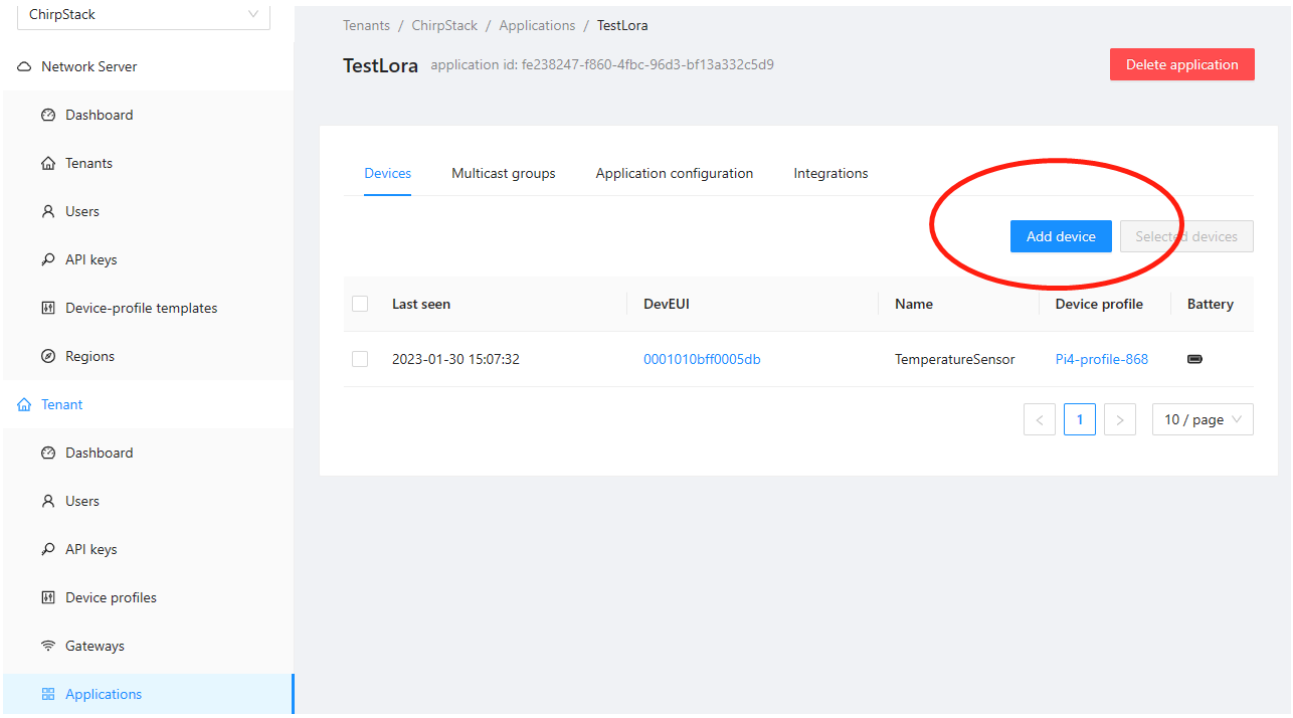
点击 Applications -> Add application

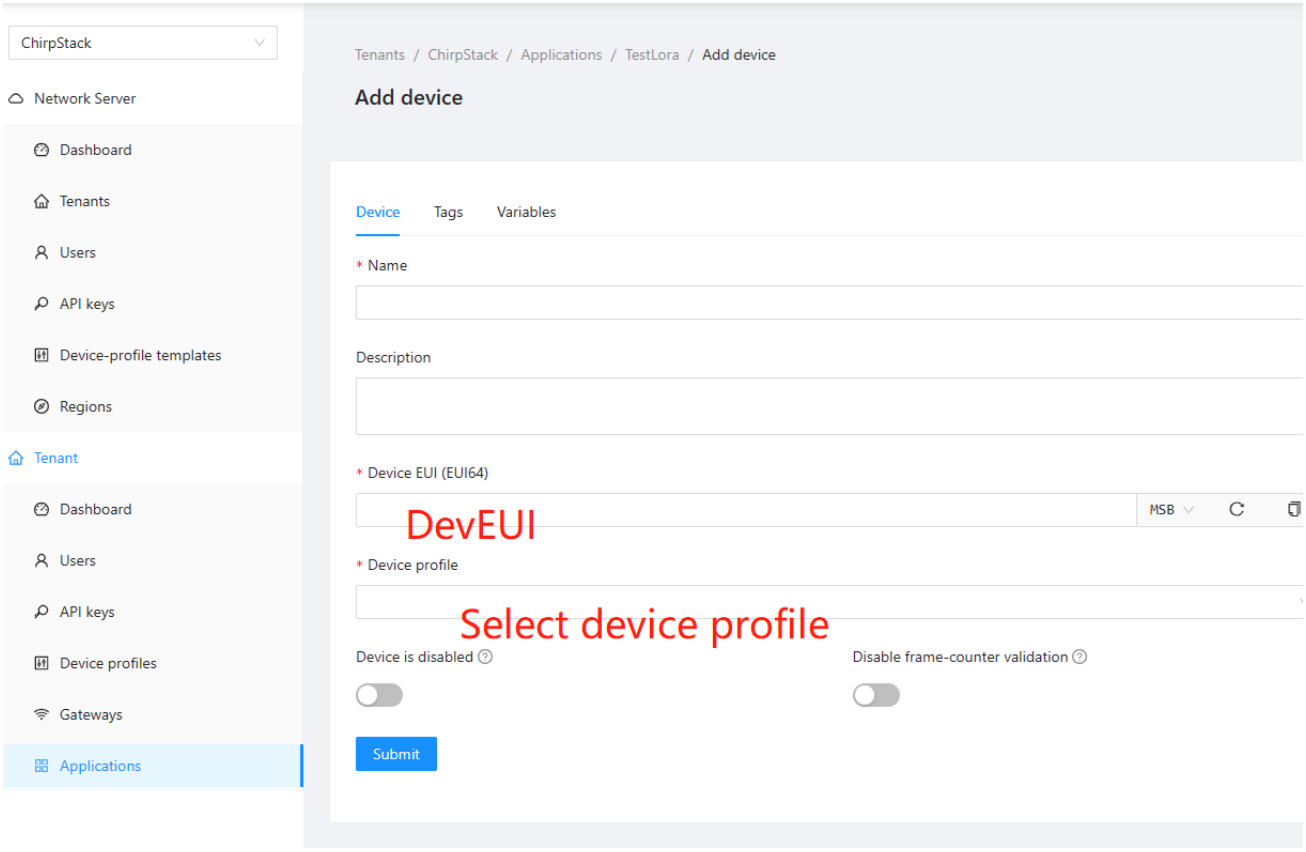


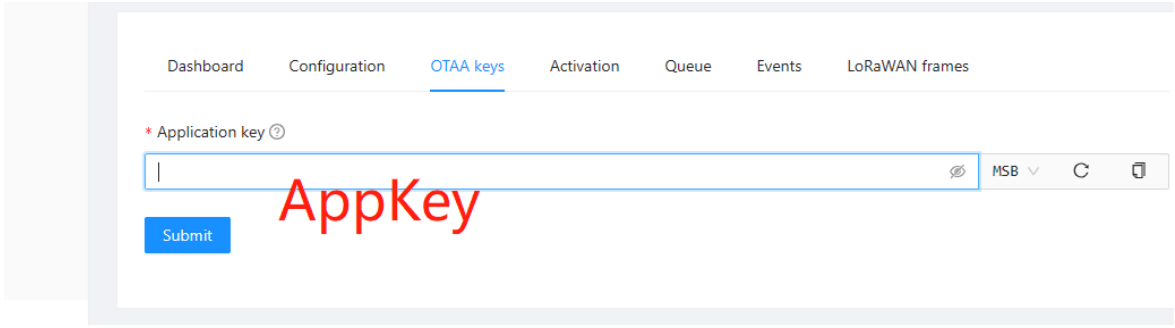
3.11.4.5 Add Device

应该知道 LoRa 终端产品的 DevEUI 和 AppKey，它们由 LoRa 终端设备制作商提供。

点击 Application -> your application -> Add device 进行 LoRa 终端设备的添加。







等待几分钟即可看到设备变成 online 状态。

3.12 加密芯片

ED-GWL2110 板载 ATECC608 加密芯片，它连接到 i2c-1 总线，器件默认地址为 0x60。

atecc: https://github.com/wirenboard/atecc-util

```
atecc -b 1 -c 'serial'
```

4 操作系统安装

4.1 镜像下载

我们提供了出厂镜像，如果系统恢复出厂设置，请点击以下链接下载出厂镜像。

Raspberry Pi OS Lite, 32-bit

- Release date: 2023-02-10
- System: 32-bit
- Kernel version: 5.15
- Debian version: 11 (bullseye)
- Downloads: ED-GWL2110

4.2 系统烧录

ED-GWL2110 默认使用 Lite 版，通过 SD 卡启动系统，。

- 下载安装 Raspberry Pi Imager 或 balenaEtcher 镜像烧写工具。
- 将 micro SD 卡插入读卡器，然后将读卡器插入电脑 USB 口。
- 打开镜像烧写工具，选择您要烧录的镜像，路径选择为识别出的大容量存储设备的路径。
- 点击烧录，等待烧录和校验完成，弹出读卡器设备。
- 打开 ED-GWL2110 设备上盖，将烧录好镜像的 micro SD 卡插入对应卡槽。
- 设备重新上电即可。

5 FAQ

5.1 默认用户名密码

用户名：pi
密码：　raspberry

6 关于我们

6.1 关于 EDATEC

EDATEC 位于上海，是 Raspberry Pi 的全球设计合作伙伴之一。我们的愿景是提供基于 Raspberry Pi 技术平台的物联网、工业控制、自动化、绿色能源和人工智能的硬件解决方案。

我们提供标准的硬件解决方案，定制设计和制造服务，以加快电子产品的开发和上市时间。

6.2 联系方式

邮箱 – sales@edatec.cn / support@edatec.cn
手机 – +86-18621560183
网站 – https://www.edatec.cn
地址 – 上海市嘉定区嘉罗公路 1661 号 29 栋