

智嵌物联 EG64X Linux 应用开发手册



深圳总部

地址：广东省深圳市宝安区新桥街道新桥社区

新和大道 6-18 号 1305

网址：www.zhiqwl.com

电话：0755-23203231



天猫店铺



淘宝店铺



京东店铺



微信公众号



公司官网

修订历史

版本	日期	原因
V1.0.0	2026.09.28	发布文档

目 录

1. 引言	1
2. 产品介绍	2
2.1 基本参数	2
2.2 型号选型表	3
2.3 系统版本查询	3
3. eMMC 资源分配	4
4. 功耗说明	5
5. 电源供电	6
5.1 电源端子	6
5.2 指示灯	6
6. 用户登录	7
6.1 默认网络配置	7
6.2 修改登录密码	7
6.3 网络状态查看	7
7. 系统服务说明	8
8. 硬件接口总览	9
9. 串口 RS485	10
9.1 接口说明	10
9.2 参数范围	10
9.3 测试说明	10
10. RS232	11
11. CAN 接口	12
12. DI/DO 数字量输入输出	14
13. LED 指示灯	15
14. 按键	16
15. USB 接口	17
16. 以太网	18
16.1 接口	18
16.2 查看网络状态	18
16.3 修改 LAN 口 IP (示例: 改为 192.168.10.1/24)	18
16.4 WAN 口配置	18
1.1 连通性测试	18
17. Wi-Fi	19
17.1 规格	19
17.2 扫描并连接 WiFi	19
17.3 查看 WiFi 状态	19
17.4 出厂预置连接	19
17.5 开启 AP 模式 (软 AP)	19
18. 4G 模块	21
18.1 规格	21
18.2 SIM 卡安装	21
18.3 拨号与状态查看	21
18.4 常见问题	21
19. GPS/GNSS	22

20. LoRa	23
20.1 接口	23
20.2 查看 SPI 设备	23
20.3 复位 LoRa 模块	23
21. RTC 与系统时间	24
21.1 查看 RTC	24
21.2 设置时间	24
21.3 开机自动同步	24
22. TF 卡	25
23. 看门狗	26
23.1 说明	26
23.2 查看看门狗状态	26
23.3 喂狗测试（手动触发脉冲）	26
23.4 应用层看门狗	26
24. 系统更新	27
25. GPIO 与资源清单	28
25.1 GPIO 编号说明	28
25.2 板载资源 GPIO 映射表	28
25.3 DI/DO 端子映射表（EG640-6X4Y）	28
25.4 DO 端子映射表（EG640-8Y）	29
25.5 gpio_control 工具	29
25.6 开发资源	29
25.7 常用调试工具清单	30
26. 设备管理 Web 界面	31
26.1 登录	31
26.2 功能模块	31
26.3 网页终端	31
27. Node-RED 可视化编程	32
27.1 运行时说明	32
27.2 示例流程	32
27.3 使用步骤	32
28. 系统配置	33
29. 系统服务与开机自启动	34
29.1 出厂预置 systemd 服务	34
29.2 特殊服务说明	34
29.3 自定义开机自启动	34
30. 常见问题 FAQ 与故障排查	36
30.1 网络	36
30.2 CAN	36
30.3 RS485	36
30.4 看门狗	36
30.5 WiFi	36
30.6 4G	36
30.7 TF 卡	36
30.8 日志与排障	36
销售网络	37

1. 引言

本手册面向使用 ZQWL-EG64X 系列工业边缘网关进行二次开发与现场部署的工程技术人员，介绍产品的硬件资源、系统结构、接口使用方法与常用测试命令。

ZQWL-EG64X 采用 Rockchip RK3506J 三核 Cortex-A7 处理器，运行 Ubuntu 22.04 LTS 系统（无桌面精简环境，rootfs 为 ext4），集成 4 路 RS485、2 路隔离 CAN FD、双百兆网口（10/100M）、USB 2.0 OTG、RTC、硬件看门狗等资源，并可选配 WiFi、4G、GPS/GNSS、LoRa、DI/DO、RS232 等功能，适用于工业数据采集、边缘计算、协议转换等场景。

约定

- 本文所有命令均以 `root` 用户执行，提示符为 `root@zqwl:~#`。
- 出厂默认 ssh 登录信息：端口 `22` 用户名 `root`，密码 `123456`（详见 第 6 章）。

2. 产品介绍

2.1 基本参数

项	规格
产品型号	ZQWL-EG64X 系列（EG640 / EG640-6X4Y / EG640-8Y / EG644 及 4GL / 4G / WF 等网络选配）
处理器	Rockchip RK3506J, 3 核 Cortex-A7, 1.2 GHz
操作系统	Ubuntu 22.04 LTS
内存	DDR3 512 MB
存储	eMMC 8 GB + MicroSD（TF）扩展
安装方式	水平桌面放置 / 挂壁式安装（选配） / 导轨式安装（选配）
工作温度	-40 °C ~ +85 °C
存储温度	-40 °C ~ +85 °C
工作湿度	5% ~ 95% RH（无凝结）
供电	DC 9 ~ 36 V（5.08 mm 凤凰端子）

说明：可用内存约 476 MB（512 MB 中部分被 GPU/保留占用）。

说明：eMMC 可用约 7.2 GB。

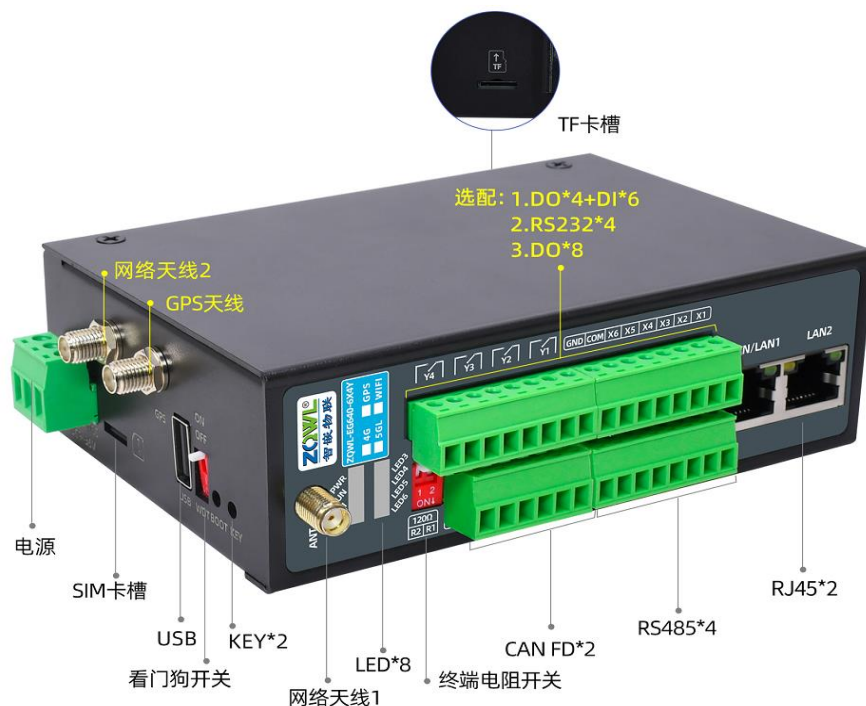


图 2-1 接口布局图

2.2 型号选型表

型号	网络类型	附加 IO
ZQWL-EG640	无	-
ZQWL-EG640-6X4Y	无	6 路 DI + 4 路 DO
ZQWL-EG640-8Y	无	8 路 DO
ZQWL-EG640-4GL	4G CAT1	-
ZQWL-EG640-6X4Y-4GL	4G CAT1	6 路 DI + 4 路 DO
ZQWL-EG640-8Y-4GL	4G CAT1	8 路 DO
ZQWL-EG640-WF	2.4G WiFi6	-
ZQWL-EG640-6X4Y-WF	2.4G WiFi6	6 路 DI + 4 路 DO
ZQWL-EG640-8Y-WF	2.4G WiFi6	8 路 DO
ZQWL-EG640-4G	4G CAT4	-
ZQWL-EG640-6X4Y-4G	4G CAT4	6 路 DI + 4 路 DO
ZQWL-EG640-8Y-4G	4G CAT4	8 路 DO
ZQWL-EG640-4GL-WF	4G CAT1 + 2.4G WiFi6	-
ZQWL-EG640-6X4Y-4GL-WF	4G CAT1 + 2.4G WiFi6	6 路 DI + 4 路 DO
ZQWL-EG640-8Y-4GL-WF	4G CAT1 + 2.4G WiFi6	8 路 DO

EG644 变体开放 RS232 模式（4 路 RS232 与 RS485 共用串口），详见 第 10 章。

2.3 系统版本查询

```
uname -a
cat /etc/os-release
cat /proc/cpuinfo | grep -i model
```

3. eMMC 资源分配

系统使用 eMMC（GPT 分区表）作为主存储介质，出厂分区布局如下（扇区大小 512 字节）：

分区名称	偏移 (sector)	大小 (sector)	说明
uboot	0x2000	0x3000	U-Boot 引导程序
misc	0x5000	0x800	misc 分区（引导参数等）
vnvm	0x5800	0x200	厂商 NVM（序列号、MAC 等）
recovery	0x5A00	0xA000	恢复系统
boot	0xFA00	0xA000	内核 / 设备树
rootfs	0x19A00	0x800000	根文件系统（ext4）
oem	0x819A00	0x8000	OEM 分区
userdata	0x821A00	grow（占满剩余）	用户数据分区

查看当前分区表：

```
lsblk
df -h
cat /proc/partitions
```

说明：eMMC 设备节点为 `/dev/mmcblk1`；userdata 分区可扩展至剩余全部空间，适合存放用户应用程序与数据。

userdata 分区使用：

- userdata 分区（`/dev/mmcblk1p8`）占满剩余空间，出厂挂载点为 `/userdata`（fstab 配置 `noauto`）；如未挂载可执行 `mount /userdata`，查看空间 `df -h /userdata`。
- 适合存放用户应用程序、数据采集文件与日志。

4. 功耗说明

状态	典型电流 (@12V DC)	说明
待机 (静态)	约 100 mA @12V	系统空闲, 无线模块关闭
正常工作	约 210 mA @12V	外设运行
4G/WiFi 射频工作	瞬间电流较大	上电及拨号瞬间, 需保证电源余量; WiFi 模组发射峰值电流 370.54 mA @ 3.3V VBAT

供电要求: 12 V 供电时电流尽量高 4G/WiFi 模组上电射频瞬间功率较大), 建议预留 20% 以上余量。

5. 电源供电

5.1 电源端子

- 端子类型：5.08 mm 凤凰端子，3 位（电源正 / 电源负 / 电源地）。
- 输入电压：DC 9 ~ 36 V。

端子	说明
V+（电源正）	DC 9 ~ 36 V 正极
V-（电源负）	电源负极（隔离参考地）
GND（电源地）	保护地 / 屏蔽地



图 5-1 电源端子接线示意

5.2 指示灯

指示灯	位置	说明
LED_PWR	前面板	电源灯（红灯常亮表示正常）
LED_RUN	前面板	系统运行指示灯（0.5 Hz 闪烁表示运行正常）
LED_1 ~ LED_6	前面板	用户自定义指示灯（见第 13 章）

注：设备 LED 资源为 1 个运行灯 (LED_RUN) + 6 个用户灯 (LED_1~6)，均非内核 leds 子系统管理；LED_PWR 为前面板物理电源指示灯。运行灯如果需要也可以自定义功能。

上电后 LED_RUN 以 0.5 Hz 频率闪烁，表示系统正常运行。若 LED_RUN 不亮或常亮，请检查电源及系统启动状态。

6. 用户登录

6.1 默认网络配置

出厂固件由 NetworkManager 统一管理网络：

接口	桥接	默认地址	说明
eth0	wan 桥	DHCP	WAN 口，连接上级网络
eth1	br-lan 桥	192.168.5.1/24	LAN 口， <code>ipv4.method shared</code> ，自带 DHCP 分配
eth2	cell 桥	DHCP	4G 拨号后获取地址
wlan0	-	DHCP	WiFi sta 模式（预置 SSIDZQWL-4GR6XX-XXXX， <code>route-metric 20</code> 见 16.2）

登录步骤：

1. 用网线将 PC 连接至设备 LAN 口（eth1）。
2. PC 自动获取 192.168.5.x 网段地址。
3. SSH 登录设备：
`ssh root@192.168.5.1`
4. 默认密码：`123456`（首次登录建议立即修改，见下文）。

注意：网络由 NetworkManager 托管，`can*`、`lo`、`dummy*`、`veth*`、`Featureid*` 接口不托管（unmanaged）。

6.2 修改登录密码

```
passwd
```

6.3 网络状态查看

```
nmcli device status
nmcli connection show
ip addr show br-lan
```

7. 系统服务说明

系统使用 systemd 作为初始化系统，出厂预置以下关键服务：

服务名	说明	主要行为
edge-base-wdt-kick.service	看门狗喂狗守护	GPIO20 (WDT) 每约 6 秒输出一次脉冲 (拉高 1 s、拉低 5 s)
edge-led-running.service	运行灯闪烁守护	LED_RUN 0.5 Hz 闪烁
edge-boot-hook.service	上电任务钩子	每次上电执行 /usr/local/bin/edge-boot-hook.sh

查看服务状态：

```
systemctl status edge-led-running.service #查看运行灯闪烁状态
systemctl list-units --type=service --state=running
systemctl restart edge-led-running.service # 重启运行灯闪烁程序
systemctl stop edge-led-running.service # 停止运行灯闪烁程序
```

上电钩子：edge-boot-hook.service 每次上电都会执行，用户可将需要随系统启动运行的脚本逻辑放入 /usr/local/bin/edge-boot-hook.sh 或改为调用自己的脚本。

8. 硬件接口总览

接口	数量	节点 / 说明
以太网	2 × 10/100M	eth0 (WAN)、eth1 (LAN 桥 br-lan)
RS485	4 路	uart1~4 → /dev/ttyS1~4, RS485 硬件自动换向
RS232	可选 4 路 (仅 EG644)	与 RS485 共用串口
CAN	2 路 (CAN FD, 带隔离)	can0 / can1
USB	1 × USB 2.0 OTG (Type-A)	默认 HOST, 支持固件升级, 5 V / 0.5 A
LED	6 个用户自定义 + 1 运行灯	LED_1~6、LED_RUN
按键	1 升级键 (BOOT) + 1 自定义键 (KEY)	-
RTC	1	(I2C0 @ 0x51), 无内置电池
硬件看门狗	1	外置硬件看门狗, GPIO20, 超时 27 s, 拨码使能
GPS/GNSS	选配	uart5 → /dev/ttyS5
LoRa	选配	SPI0 → /dev/spidev0.1, 复位 GPIO0_C3
DI/DO	按变体	EG640-6X4Y: 6 DI + 4 DO; EG640-8Y: 8 DO
Wi-Fi	选配	wlan0
4G	选配	eth2 (桥接 cell, CAT1 / CAT4)
Nano SIM	1	4FF, 热插拔 (USIM_DET 检测)
MicroSD (TF)	1	SPI1 → /dev/mmcblk0, 自动挂载

9. 串口 RS485

9.1 接口说明

设备提供 4 路 RS485 串口，采用硬件自动换向，无需软件控制收发方向。

串口	设备节点	端子
uart1	/dev/ttyS1	2 针凤凰端子
uart2	/dev/ttyS2	2 针凤凰端子
uart3	/dev/ttyS3	2 针凤凰端子
uart4	/dev/ttyS4	2 针凤凰端子

接线：每路 RS485 为 A (+) / B (-) 两线，与对端设备 A-A、B-B 连接，必要时共地。

9.2 参数范围

参数	范围
波特率	1200 ~ 1500000（固件保守配置 1200 ~ 961200）
校验	NONE / EVEN / ODD / MARK / SPACE
数据位	5 / 6 / 7 / 8（9 位不支持）
停止位	1 / 1.5 / 2
流控	无

9.3 测试说明

1. 查看串口设备

```
ls -l /dev/ttyS1 /dev/ttyS2 /dev/ttyS3 /dev/ttyS4
```

2. 配置波特率 (stty)

```
stty -F /dev/ttyS1 115200 cs8 -cstopb -parenb
```

```
stty -F /dev/ttyS1 -a # 查看当前参数
```

配置 8E1（8 数据位 + 偶校验 + 1 停止位）：

```
stty -F /dev/ttyS1 9600 cs8 -cstopb -parenb parodd
```

3. 发送与接收测试

提示：/dev/ttyS1 默认带有 icrnl/ixon/echo 终端行规则，直接 `cat /dev/ttyS1` 接收二进制数据可能被转换；建议先 `stty -F /dev/ttyS1 raw` 再收发。

终端 1: 接收

```
cat /dev/ttyS1 &
```

终端 2: 发送

```
echo -n "hello" > /dev/ttyS1
```

10. RS232

RS232 为选配功能，仅 EG644 变体开放，4 路 RS232 与 RS485 共用物理串口（uart1~4）。

项目	说明
适用型号	EG644
串口	uart1~4 → /dev/ttyS1~4（与 RS485 共用）
电平	RS232
波特率	1200 ~ 115200

11. CAN 接口

接口说明

设备提供 2 路带隔离的 CAN 接口（CAN FD）。

接口	节点	端子	隔离	终端电阻
CAN0	can0	2 针凤凰端子	是	2 位琴键式拨码开关（2P）控制，默认关闭
CAN1	can1	2 针凤凰端子	是	2 位琴键式拨码开关（2P）控制，默认关闭

参数与配置

支持帧格式：

- 标准帧（11 bit ID）+ 扩展帧（29 bit ID）
- 经典 CAN：数据 0~8 字节
- CAN FD：数据 0~64 字节（BRS 变速）

支持波特率：

模式	波特率
经典 CAN	10k / 20k / 50k / 100k / 125k / 250k / 500k / 800k / 1M
CAN FD 仲裁段	最高 1M
CAN FD 数据段	最高 8M（BRS）

重要：出厂默认 `fd on`。使用经典 CAN 时必须加 `fd off`，否则会报 `Invalid argument`。

配置并启用 CAN0（经典 CAN，500 kbit/s）：

```
ip link set can0 down
ip link set can0 up type can bitrate 500000 fd off
ip -details link show can0
```

配置 CAN FD（仲裁 500k / 数据 2M）：

```
ip link set can0 down
ip link set can0 up type can bitrate 500000 dbitrate 2000000 fd on
```

查看 CAN 状态：

```
ip -details -statistics link show can0
cat /proc/net/can/stats
```

收发测试（需安装 can-utils）：

```
# 终端 1：监听
candump can0 &

# 终端 2：发送标准帧数据（ID=0x123，数据 DE AD BE EF）
cansend can0 123#DEADBEEF

# 发送扩展帧（29 bit ID）
cansend can0 00000123#11.22.33.44.55.66.77.88

# CAN FD 长帧（64 字节，## 后第 1 个字符为 flags，flags=0 时写 `0`，即 `##0` + 64 字节数
```

据)

```
cansend can0
123##000102030405060708090A0B0C0D0E0F101112131415161718191A1B1C1D1E1F20212223242526
2728292A2B2C2D2E2F303132333435363738393A3B3C3D3E3F
```

断开 CAN:

```
ip link set can0 down
```

- 🔗 **终端电阻:** CAN0/CAN1 终端电阻由 2 位琴键式拨码开关 (2P) 控制, 默认关闭; 短接调试时必须保持全部关闭 (只开一端会单向通), 长线缆应用时在总线两端各开启 120 Ω 终端电阻。
- 🔗 **实测细则:**
 - FD 8M 验证方法: `cangen -f -b -L 64` (FD 帧 + BRS + 64 字节)。
 - 速率切换必须慢: `ip link set can0 down` 后 `sleep 2s`, 否则高速→低速切换会竞态假丢帧。
 - berr-counter 为接口级累积值、跨速率不清零, 不可作判据; 判定只看帧收发完整性。

12. DI/DO 数字量输入输出

DI/DO 为选配功能，按型号不同数量不同：

变体	DI	DO
EG640-6X4Y	6 路	4 路
EG640-8Y	0 路	8 路
EG640 / EG644	无	无

DO（继电器输出）：

- 触点容量：5 A @ 250 VAC / 5 A @ 30 VDC
- 上电默认断开

DI（数字量输入）：

- 支持脉冲输入检测，最高约 6 kHz（sysfs GPIO 边沿中断 + poll 方式）

使用 gpio_control 操作 DI/DO：

```
# 读取 DI_1 状态
gpio_control PIN_001 read

# 读取 DI_6 状态
gpio_control PIN_006 read

# 吸合 DO_1（写 1）
gpio_control PIN_007 write 1

# 断开 DO_1（写 0）
gpio_control PIN_007 write 0
```

- 说明：端子排丝印名称与 gpio_control 参数的对应关系见 第 25 章。DO 为继电器输出，操作时请注意负载容量。
- 说明：上述示例按 EG640-6X4Y（6DI+4DO）编写。EG640 基础款（无 DI/DO）下 PIN_001/PIN_006 等端子为 io 输入，gpio_control PIN_001 read 返回 0；具体 DI/DO 数量以第 25 章变体映射表为准。

13. LED 指示灯

设备提供 6 个用户自定义 LED (LED_1 ~ LED_6) 和 1 个运行灯 (LED_RUN)。所有 LED 均为低电平点亮 (active_low)。

名称	SOC 引脚	sysfs 节点	默认状态
LED_RUN	GPIO1_A4	/sys/class/leds/led_run	默认 none (由 edge-led-running 服务手动控制闪烁; 可用触发含 heartbeat/timer/none)
LED_1	GPIO2_A1	/sys/class/leds/led1	灭
LED_2	GPIO2_A2	/sys/class/leds/led2	灭
LED_3	GPIO2_A5	/sys/class/leds/led3	灭
LED_4	GPIO2_A4	/sys/class/leds/led4	灭
LED_5	GPIO2_A3	/sys/class/leds/led5	灭
LED_6	GPIO2_A0	/sys/class/leds/led6	灭

点亮 / 熄灭用户 LED (sysfs, 推荐):

LED 由内核 leds 子系统管理 (设备树已绑定), 请通过 `/sys/class/leds/` 操作:

```
echo 1 > /sys/class/leds/led1/brightness # LED_1 亮
echo 0 > /sys/class/leds/led1/brightness # LED_1 灭
```

注意: LED 引脚已被 leds 驱动占用, 不能通过 `gpio_control` 或 `sysfs export` 直接操作 (会报 "Failed to export GPIO")。

配置运行灯触发模式:

```
cat /sys/class/leds/led_run/trigger # 查看可用触发 (heartbeat/timer/none 等)
echo heartbeat > /sys/class/leds/led_run/trigger
echo none > /sys/class/leds/led_run/trigger # 恢复手动控制
```

14. 按键

设备提供 2 个按键：

按键	说明
BOOT（升级键）	按住 BOOT + 上电进入 LOADER 模式（用于 RKDevTool 升级）
KEY（自定义按键）	用户自定义功能键，高电平默认，按下为低电平，去抖 50 ms

读取 KEY 状态：

KEY 由内核 gpio-keys 驱动管理（设备树绑定为功能键，/dev/input/event*），应用层通过输入子系统读取：

```
cat /proc/bus/input/devices # 查看按键设备节点
```

监听按键事件（示例，使用 evtest 或直接读 event 节点）：

```
evtest /dev/input/event1 # 若已安装 evtest（gpio-keys 功能键在 event1；event0 为 adc-keys）
```

```
# 或直接读原始事件（按下/松开均产生事件）
```

```
cat /dev/input/event1 | hexdump
```

说明：KEY 引脚（GPIO1_A5）已被 gpio-keys 驱动占用，不能通过 gpio_control KEY read 或 sysfs export 直接读取（会报 "Failed to export GPIO"）。如需在应用层自定义按键功能，建议通过输入子系统事件或改造设备树按键绑定。

15. USB 接口

- 接口：1 × USB 2.0 OTG (Type-A)
- 默认模式：HOST (主机模式)
- 供电：5 V / 0.5 A
- 用途：外接 USB 设备 (U 盘、USB 转串口等)、固件升级

查看 USB 设备：

```
lsusb
cat /proc/bus/usb/devices # 内核 6.1 不提供该 procfs 接口，以 lsusb 为准
```

挂载 U 盘 (FAT32)：

```
lsblk
mkdir -p /mnt/usb
mount /dev/sda1 /mnt/usb
ls /mnt/usb
umount /mnt/usb
```

固件升级：USB OTG 口同时用于 RKDevTool 升级 (LOADER 模式)，详见 第 24 章。

注意：USB 口基于 USB 2.0 OTG0 (Type-A)，默认 HOST 模式；

16. 以太网

16.1 接口

接口	角色	默认配置
eth0	WAN	桥接 wan, DHCP 获取地址
eth1	LAN	桥接 br-lan, 192.168.5.1/24, shared (自带 DHCP 服务)

16.2 查看网络状态

```
nmcli device status
nmcli connection show
ip addr show wan
ip addr show br-lan
ip route
ethtool eth0 | grep -i speed # 查看网口协商速率 (10/100M)
```

提示：双网口均为 10/100M，使用 ethtool 可确认协商结果。

16.3 修改 LAN 口 IP (示例：改为 192.168.10.1/24)

```
nmcli connection modify br-lan ipv4.addresses 192.168.10.1/24 ipv4.method shared
nmcli connection up br-lan
```

注意：示例地址 192.168.10.1/24 仅为演示；实际使用时请按现场网段选取不与 WAN/LAN 现有网段冲突的地址。

可以使用 Web 界面实现网络网段的切换。

16.4 WAN 口配置

```
# 查看 WAN 地址
nmcli device show eth0

# WAN 使用 DHCP
nmcli connection modify wan ipv4.method auto

# WAN 使用静态 IP
nmcli connection modify wan ipv4.method manual ipv4.addresses 192.168.1.100/24
ipv4.gateway 192.168.1.1 ipv4.dns 114.114.114.114
nmcli connection up wan
```

1.1 连通性测试

```
ping -c 3 192.168.5.1 # 本机 LAN
ping -c 3 www.baidu.com # 外网 (需 WAN 已连通)
```

17. Wi-Fi

Wi-Fi 为选配功能（2.4 GHz 单频）。

17.1 规格

项	规格
无线标准	IEEE 802.11 b/g/n/ax
频段	2.4 GHz（2.400 ~ 2.4835 GHz）
模式	STA / AP
天线	SMA 母座外螺纹内孔
加密	WEP / WPA / WPA2
最大速率	286.8 Mbps（TX） / 229.4 Mbps（RX）

17.2 扫描并连接 WiFi

```
nmcli radio wifi on
nmcli device wifi list          # 扫描可用 Wi-Fi
nmcli device wifi connect "MySSID" password "MyPassword" # 连接指定 WiFi（替换为实际 SSID 与密码）
```

提示：也可通过设备管理 Web 界面的"网络"页面进行 WiFi 连接（见 第 26 章）。

17.3 查看 WiFi 状态

```
nmcli device show wlan0
iw dev wlan0 link
iw dev wlan0 scan | head -40
```

扫描结果解读（iw dev wlan0 scan 输出）：

字段	含义	参考
SSID	网络名称	-
freq / channel	信道频率 / 信道号	2.4G 信道 1~13
signal	信号强度（dBm）	-50 dBm 以上优、-60~-70 中、-80 以下弱
capability / RSN	加密能力	WPA2/WPA3 等

提示：nmcli device wifi list 输出更简洁（含 SSID / 信道 / 信号 / 加密列），现场优先使用。

17.4 出厂预置连接

出厂已预置连接 wan-wifi（SSID ZQWL-4GR6XX-XXXX，密码 12345678），上电后自动尝试连接：

```
nmcli connection up wan-wifi
```

17.5 开启 AP 模式（软 AP）

设备支持 AP 模式。注意：nmcli device wifi hotspot 方式会失败，请使用下述推

荐用法。

能力边界：AP 模式仅支持 2.4 GHz（信道 1~13），无 5 GHz；信道带宽固定 20 MHz（HT40 在该驱动 AP 模式下不稳）；加密仅 WPA2-PSK；单射频单天线，STA 与 AP 不能并发，切换为串行。

推荐用法一（独立热点 ap0 虚拟接口，NetworkManager 全程托管）：

```
iw phy phy0 interface add ap0 type __ap
ip link set ap0 up
nmcli con add type wifi ifname ap0 mode ap con-name ap0-test ssid "EG64X-AP"
nmcli con modify ap0-test wifi-sec.key-mgmt wpa-psk wifi-sec.psk 12345678
802-11-wireless.band bg 802-11-wireless.channel 6 ipv4.method shared
nmcli con up ap0-test
```

推荐用法二（单接口切换，产品推荐，wlan0 并入 br-lan 网桥）：

```
nmcli con add type wifi ifname wlan0 mode ap con-name wlan0-ap ssid "EG64X-AP"
nmcli con modify wlan0-ap wifi-sec.key-mgmt wpa-psk wifi-sec.psk 12345678
802-11-wireless.band bg 802-11-wireless.channel 6 master br-lan slave-type bridge
nmcli con up wlan0-ap # 切换到 AP
nmcli con up wan-wifi # 切回 STA
```

吞吐实测：AP 下行实测 TCP ~64~67 Mbps、UDP ~67~70 Mbps（20 MHz / 1×1 / WPA2，PHY 78~86 Mbps，TCP 协议开销约 20~25%；多流不能突破单流上限）。

提示：4G 模组电源由 NET_USB_PWR_EN 控制（见第 25 章 GPIO 表）；WiFi 模组共用该电源控制信号。

18. 4G 模块

4G 为选配功能，分 CAT1（型号后缀 4GL）与 CAT4（型号后缀 4G）两档。

18.1 规格

项	规格
制式	4G LTE（CAT1 / CAT4）
网络接口	eth2（桥接 cell）
SIM 卡	Nano SIM（4FF），热插拔（USIM_DET 检测）

18.2 SIM 卡安装

1. 断电后打开 SIM 卡槽。
2. 按丝印方向插入 Nano SIM 卡。
3. 上电，等待系统检测到 SIM（USIM_DET 引脚检测热插拔）。

18.3 拨号与状态查看

```
nmcli connection up cell
nmcli device show eth2
ip addr show eth2
ip route
```

18.4 常见问题

现象	排查方法
eth2/cell 无地址	检查 SIM 卡是否插好、是否欠费停机、天线是否接好
SIM 未识别	<code>gpio_control USIM_DET read</code> 查看检测电平（有卡为 1）
信号弱	确认 SMA 天线拧紧，天线位置避开金属遮挡

19. GPS/GNSS

GPS/GNSS 为选配功能（GPS 定位或 BDS 定位）。

接口

项	说明
串口	uart5 → /dev/ttyS5（8N1）
定位系统	GPS / BDS
天线	SMA 母座（GNSS）

读取定位数据

GNSS 模块通过串口输出 NMEA 0183 语句：

```
stty -F /dev/ttyS5 115200 cs8 -cstopb -parenb
cat /dev/ttyS5
$GNGGA,060306.000,3132.1234,N,12034.5678,E,1,08,1.0,10.0,M,0.0,M,,*46
$GNRMC,060306.000,A,3132.1234,N,12034.5678,E,0.5,180.2,230926,,D*7B
...
```

NMEA 主要字段说明（以 GNGGA / GNRMC 为例）：

语句	字段	含义
GNGGA	1	UTC 时间（时分秒，毫秒）
GNGGA	2/3	纬度（度分格式，如 3132.1234 = 31°32.1234'）/ N/S
GNGGA	4/5	经度（度分格式）/ E/W
GNGGA	6	定位状态：0=未定位、1=单点定位、2=差分定位
GNGGA	7	使用卫星数
GNGGA	8	HDOP 水平精度因子（越小越准）
GNRMC	2	定位状态：A=有效定位、V=无效
GNRMC	3/4	纬度 / N/S
GNRMC	5/6	经度 / E/W
GNRMC	7	地面速率（节）
GNRMC	9	UTC 日期（日月年）

说明：GNGGA / GNRMC 等语句可解析出经纬度、时间、卫星数等。GNSS 通过 uart5(/dev/ttyS5) 接入，串口参数 8N1；支持波特率 9600~921600；NMEA 默认波特率应为 115200 或 9600。

说明：设备出厂预装并运行 gpsd 服务（读取 /dev/ttyS5，监听 TCP 2947），应用可通过 gpsd 协议（如 gpstpipe/gpsmon）获取定位数据，也可直接读取串口 NMEA。

```
gpstpipe -r -n 3 # 通过 gpsd（TCP 2947）输出 NMEA 原始语句
gpsmon # 交互式监视窗口（需 gpsd-clients 包）
apt install gpsd-clients # 如未安装客户端工具
```

20. LoRa

LoRa 为选配功能，用于远距离低功耗无线通信。

20.1 接口

项	说明
SPI	SPI0 → /dev/spidev0.1 (CS1)
复位引脚	GPIO0_C3 (LORA_RST)

说明：SPI0 最高时钟 50 MHz，片选 CS1 对应 GPIO0_B7；LoRa 复位信号 GPIO0_C3 (LORA_RST)。

20.2 查看 SPI 设备

```
ls -l /dev/spidev0.1
cat /sys/bus/spi/devices/spi0.1/modalias
```

20.3 复位 LoRa 模块

```
gpio_control LORA_RST write 0
sleep 0.1
gpio_control LORA_RST write 1
```

21. RTC 与系统时间

设备内置 RTC 芯片（I2C0，地址 0x51），带纽扣电池，断电后时间保持。

21.1 查看 RTC

```
ls -l /dev/rtc0
hwclock -r          # 读取 RTC 时间
i2cdetect -y 0      # I2C0 总线设备扫描（应见 0x51）
```

说明：i2cdetect 中 0x51 显示为 UU，即表示 RTC 已被内核识别，属正常。

21.2 设置时间

```
# 设置系统时间
date -s "2026-09-23 12:00:00"

# 写入 RTC（系统时间 → RTC）
hwclock -w

# 从 RTC 读回系统（RTC → 系统时间）
hwclock -s
```

21.3 开机自动同步

系统启动时会从 RTC 读取时间。设备出厂预装 chrony 服务（chronyd）进行联网自动校时（NTP），默认作为 NTP 客户端/服务器运行；具体默认 NTP 服务器以出厂配置为准。

提示：RTC 由纽扣电池供电（默认不配电池），断电后时间可保持；若发现断电后时间不准确，可先通过 `hwclock -w` 重新写入。

22. TF 卡

设备支持 MicroSD(TF)卡扩展存储,TF 卡通过 SPI1 挂载(设备节点 /dev/mmcblk0),插入后由 udev 规则自动挂载。

自动挂载规则:

- 挂载点: /mnt/tfcard/<分区名>
- 规则文件: /etc/udev/rules.d/99-tf-automount.rules
- 插入分区自动挂载,拔出自动卸载

查看与使用:

```
lsblk
df -h
ls /mnt/tfcard/
ls /mnt/tfcard/mmcblk0p1    # 第一个分区的挂载点
```

手动挂载 / 卸载:

```
systemd-mount --no-block /dev/mmcblk0p1 /mnt/tfcard/mmcblk0p1
systemd-mount --umount /mnt/tfcard/mmcblk0p1
```

说明: TF 卡通过 SPI1 接口挂载(时钟 25 MHz),读写速率低于原生 MMC/SD 控制器,适合存放配置、日志、数据采集文件等,不建议用作系统盘。

23. 看门狗

23.1 说明

设备内置外置硬件看门狗，用于在系统或业务程序停摆时自动复位设备。

项	说明
芯片	外置硬件看门狗
控制引脚	GPIO20（GPIO0_C4）
超时时间	27 s（出厂固件配置）
使能方式	拨码开关（1 位琴键式）

喂狗机制：系统服务 `edge-base-wdt-kick.service` 每约 6 秒通过 GPIO20 输出一脉冲（拉高 1 s、拉低 5 s）。拨码开关使能看门狗后，若系统长时间（超过超时时间）未喂狗，设备将自动重启。

23.2 查看看门狗状态

```
systemctl status edge-base-wdt-kick.service
ls -l /dev/watchdog
dmesg | grep -i watchdog
```

说明：外置硬件看门狗为 GPIO 外部看门狗（GPIO20 脉冲控制），不走内核 watchdog 框架。

23.3 喂狗测试（手动触发脉冲）

注意：系统喂狗服务 `edge-base-wdt-kick.service` 运行时持续操作 GPIO20。手动测试前请先停止该服务，避免两者冲突：

```
systemctl stop edge-base-wdt-kick.service
echo 1 > /sys/class/gpio/gpio20/value
sleep 1
echo 0 > /sys/class/gpio/gpio20/value
systemctl start edge-base-wdt-kick.service # 测试完恢复
```

注意：请勿在未确认业务可恢复的情况下关闭喂狗服务。关闭 `edge-base-wdt-kick.service` 且看门狗拨码处于使能状态时，设备将在约 27 s 后复位。

23.4 应用层看门狗

开发者可在自己的业务程序中周期触发 GPIO20 脉冲，替代系统默认喂狗，实现“业务停摆即复位”：

```
# 示例：业务心跳喂狗
while true; do
    echo 1 > /sys/class/gpio/gpio20/value; sleep 1
    echo 0 > /sys/class/gpio/gpio20/value; sleep 5
done &
```

说明：具体喂狗时序需与外置硬件看门狗超时参数匹配，超时 27 s 内必须完成一次脉冲周期，否则复位。

24. 固件升级

升级方式

方式	说明
RKDevTool 烧录	LOADER 模式，通过 USB OTG 连接 PC 烧录整机固件

RKDevTool 烧录步骤

1. 安装 RKDevTool（Windows），并安装对应 USB 驱动。
2. 按住 BOOT 键不放，再给设备上电，进入 LOADER 模式（USB 设备枚举为 Loader 设备）。
3. 用 USB OTG 线连接设备与 PC。
4. 打开 RKDevTool，点击升级固件选项卡，点击固件选择对应.img 文件（包含 loader / parameter / uboot / boot / rootfs 等镜像）。
5. 点击"升级"开始烧录，等待进度完成，设备自动重启。

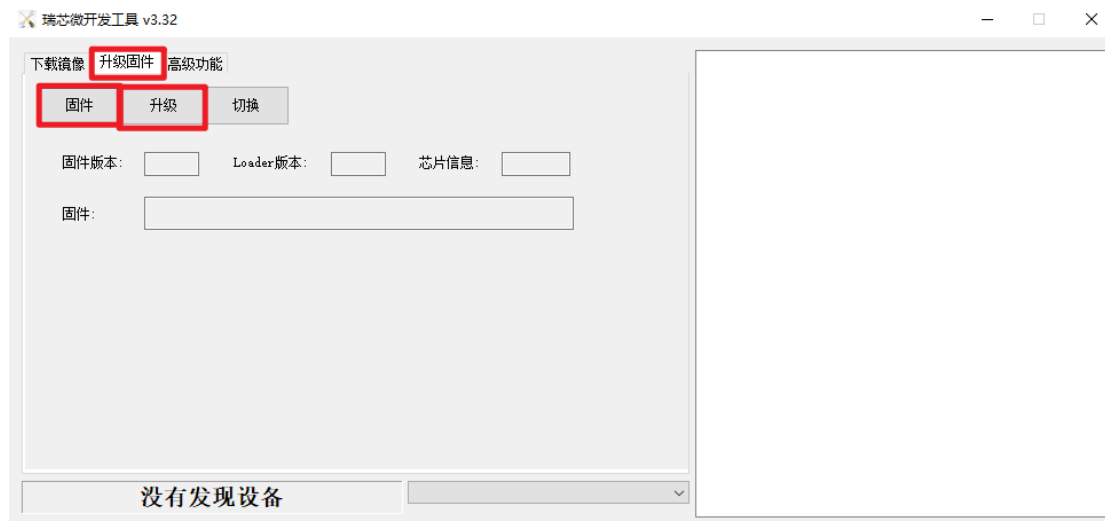


图 24-1

注意：烧录会覆盖 eMMC 全部内容，操作前请备份用户数据。烧录过程中请勿断电、拔线。

25. GPIO 与资源清单

25.1 GPIO 编号说明

RK3506 全局 GPIO 编号公式（与出厂 gpio_control 工具一致）：

GPIO 编号 = bank × 32 + group × 8 + pin

其中 group: A=0, B=1, C=2, D=3; pin 取值 0~7

示例: GPIO1_A4 = 1×32 + 0×8 + 4 = 36

25.2 板载资源 GPIO 映射表

名称	SOC 引脚	全局编号	方向	说明
LED_RUN	GPIO1_A4	36	输出	运行灯, active_low（出厂固件实际由服务控制, 见第 13 章）
KEY	GPIO1_A5	37	输入	自定义按键, 按下为低
USIM_DET	GPIO1_B0	40	输入	SIM 卡热插拔检测
NET_USB_PWR_EN	GPIO1_B1	41	输出	4G 模组电源使能, active_low
LORA_RST	GPIO0_C3	19	输出	LoRa 复位
WDT	GPIO0_C4	20	输出	看门狗喂狗（外置硬件看门狗）
LED_1	GPIO2_A1	65	输出	用户 LED, active_low
LED_2	GPIO2_A2	66	输出	用户 LED, active_low
LED_3	GPIO2_A5	69	输出	用户 LED, active_low
LED_4	GPIO2_A4	68	输出	用户 LED, active_low
LED_5	GPIO2_A3	67	输出	用户 LED, active_low
LED_6	GPIO2_A0	64	输出	用户 LED, active_low

25.3 DI/DO 端子映射表（EG640-6X4Y）

丝印	功能	SOC 引脚	全局编号
PIN_001	DI_1	GPIO1_B7	47
PIN_002	DI_2	GPIO1_C0	48
PIN_003	DI_3	GPIO1_C1	49
PIN_004	DI_4	GPIO1_C4	52
PIN_005	DI_5	GPIO1_C5	53
PIN_006	DI_6	GPIO1_C6	54
PIN_007	DO_1	GPIO1_C7	55
PIN_008	DO_2	GPIO1_D0	56
PIN_009	DO_3	GPIO1_D1	57
PIN_010	DO_4	GPIO0_A6	6

25.4 DO 端子映射表 (EG640-8Y)

丝印	功能	SOC 引脚	全局编号
PIN_001	DO_1	GPIO1_B7	47
PIN_002	DO_2	GPIO1_C0	48
PIN_003	DO_3	GPIO1_C1	49
PIN_004	DO_4	GPIO1_C4	52
PIN_005	DO_5	GPIO1_C5	53
PIN_006	DO_6	GPIO1_C6	54
PIN_007	DO_7	GPIO1_C7	55
PIN_008	DO_8	GPIO1_D0	56

25.5 gpio_control 工具

- 路径: `/usr/local/bin/gpio_control` (符号链接到 `/usr/local/command/gpio_frontend.sh`)
- 用法: `gpio_control <引脚名> read|write [值]`
- 支持的引脚名: PIN_001~PIN_010 (DI/DO/IO 端子, 中间号不映射)、USIM_DET、NET_USB_PWR_EN、USB_OTG_PWREN、LORA_RST、WDT, 以及 `P{bank}{group}{pin}` 格式 (如 P1B1)
- 注意: LED_1~LED_6、LED_RUN、KEY 已被内核驱动 (leds / gpio-keys) 占用, 不能通过本工具操作, 请分别使用 `/sys/class/leds/` 和 `/dev/input/` (见第 13、14 章)
- 注意: USB_OTG_PWREN (GPIO35) 已被 regulator 驱动 `vcc5v0-otg0` 占用, 不能通过本工具操作 (报 `Failed to export GPIO 35`); 控制它请用 `/sys/kernel/debug/regulator/vcc5v0_otg0/enable` (见第 15 章)

```
gpio_control PIN_001 read
gpio_control PIN_007 write 1
gpio_control P1B1 write 1
```

- 说明: P1B1 即 NET_USB_PWR_EN (GPIO41, 4G 模组电源使能, active_low)。写 1 会关闭 4G / Wi-Fi 模组电源 (eth2 网卡消失、4G 断开), 写 0 恢复上电; 操作时注意 4G / Wi-Fi 业务中断。
- 如果进行高频 IO 操作, 不要使用 gpio_control, gpio_control 脚本耗时较高

25.6 开发资源

资源	路径 / 说明
示例工程 (多文件)	<code>/usr/demo</code> (多文件示例工程: <code>src/main.c + math.c + utils.c</code> , Makefile 支持 <code>make / make clean / make run</code>)
示例工程 (hello)	<code>/usr/demo1</code> (最小 hello 工程, <code>bin/hello</code>)
交叉编译	ARM32 工具链 (arm-linux-gnueabi 系列), 推荐静态链接 (<code>-static</code>)

	后拷贝至设备运行；示例工程均以 gcc 编译（-Wall -Wextra）
网页终端	通过设备管理 Web 界面访问 （/api/system/remote/terminal/proxy/，由网关程序中转）
GPIO 工具	/usr/local/bin/gpio_control

板上编译示例（设备预装 gcc 11.4.0 与 make，可直接在板上编译示例工程，可执行文件输出在对应 bin 目录下）。

```
cd /usr/demo
make
./bin/demo
```

说明：/usr/demo 为多文件工程（src/main.c + math.c + utils.c，Makefile 支持 make / make clean / make run）；usr/demo1 为最小 hello 工程(bin/hello)。两个工程均为 ARM 32 位动态链接 ELF。

交叉编译：主机端使用 ARM32 交叉编译工具链（如 arm-linux-gnueabi-hf-gcc），静态链接后可拷贝至设备运行：

```
arm-linux-gnueabi-hf-gcc -static -o hello hello.c
scp hello root@192.168.5.1:/usr/local/bin/
/usr/local/bin/hello
```

25.7 常用调试工具清单

工具	用途
nmcli	网络管理（NetworkManager 命令行）
ip / ethtool	网口配置与查询
stty	串口参数配置
candump / cansend	CAN 收发测试（can-utils）
hwclock / timedatectl	RTC 与时间
lsblk / df / mount	存储管理
i2cdetect / i2cget	I2C 总线调试
dmesg / journalctl	系统日志
gpio_control	GPIO 读写
gpspipe / gpsmon	GPS 定位数据获取（gpsd 客户端，见第 19 章）
iperf3	网络吞吐测试（需自行安装，见第 17 章 WiFi 吞吐实测）

26. 设备管理 Web 界面

设备内置 Web 管理界面，出厂默认通过 HTTP 80 端口对外服务，实际端口以设备 Web 界面提示为准。

26.1 登录

1. 使用浏览器访问 `http://<设备 IP>/`（设备 IP 可通过 LAN 口默认地址 `192.168.5.1` 或 WAN 口动态地址访问）。
2. 输入默认账号：用户名 `admin`，密码 `admin`。
3. 首次登录后建议立即修改密码（见下一节 功能模块 之“用户管理”）。

提示：若无法访问请确认 Web 端口是否被其他程序占用。

26.2 功能模块

模块	页面 / 入口	主要功能
总览	/overview	设备性能（CPU / 内存 / Flash）、WAN / LAN / 4G / WiFi 状态、GPS 信息（页面路径以实际界面为准）
网络	网络管理	WAN / LAN / WiFi / 4G 管理、多 WAN 策略、路由规则、网络诊断（ping / traceroute / nslookup）
系统	系统设置	基本配置（主机名 / Web 端口 / 时区 / NTP）、日志管理、配置管理、定时重启、用户管理
远程管理	远程设置	SSH 开关 / 端口 / root 登录开关、网页终端启停
固件升级	升级页面	在线升级 / 本地 .bin 文件上传
可视化编程	Node-RED	启用 / 禁用 / 打开编辑器（详见第 27 章）

26.3 网页终端

Web 界面内置网页终端，入口为 `/api/system/remote/terminal/proxy/`，由固件预装的 `ttyd` 提供、网关程序负责中转。网页终端与 SSH 同样以 root 身份进入系统命令行，方便在无法使用 SSH 的场合进行远程调试。

```
# 确认 ttyd 进程运行情况（若存在）
ps aux | grep ttyd
```

说明：网页终端的启停可在“远程管理”页面控制。

27. Node-RED 可视化编程

设备管理 Web 界面提供"可视化编程"功能，用于启用 / 禁用 / 打开 Node-RED 编辑器，实现拖拽式流程编排（串口采集、协议转换、MQTT 上云等），无需编写代码。

27.1 运行时说明

- Node-RED 运行时由 **nodered.service** 系统服务承载（注意：服务名为 **nodered.service**，不是 **node-red.service**），设备已预装 Node-RED 并随系统自启（enabled + active）；node v20.19.4，Node-RED v4.1.8（**/usr/bin/node-red**）；Node-RED 监听 1880 端口。
- Node-RED 的启用、停用与状态查询由设备管理 Web 界面的接口管理（技术附录）：**/edge/nodered/status**（查询状态）、**/edge/nodered/control**（启停控制）。
- 同机预装 mosquitto MQTT broker（监听 127.0.0.1:1883），可用于 Node-RED 流程本地 MQTT 通信。

```
# 查看 Node-RED 服务状态
systemctl status nodered
```

27.2 示例流程

随手册提供的 **NodeRed-Example.json** 示例流程，可直接导入使用。流程实现"Modbus 采集 + MQTT 上云"：

环节	配置
数据源	modbus-client 节点，串口/dev/ttyS3，RTU 模式，9600 8N1
采集对象	Modbus 从站（寄存器点位）
寄存器映射	function 节点实现点位映射，采用分组读取策略： maxGap=10 、 maxRegisters=100
上云通道	MQTT 节点，broker 39.108.220.80:8883
可靠性	MQTT 上行 QoS 1 + function 节点离线缓存与自动重发（ maxQueueSize=1000 ）

说明：示例流程中的 MQTT broker 地址与端口为演示配置，实际使用请替换为自有服务器并配置好证书 / 账号。流程内置离线缓存与自动重发（队列上限 1000 条），网络恢复后自动补发。

27.3 使用步骤

1. 登录 Web 管理界面，进入"可视化编程"页面。
2. 点击"启用"，开启 Node-RED 运行时。
3. 点击"打开编辑器"，进入 Node-RED 画布。
4. 导入流程 JSON：菜单 → 导入 → 选择随手册提供的 **NodeRed-Example.json**。
5. 按现场需求修改节点配置（串口、Modbus 从站地址、MQTT broker 等）。
6. 点击右上角"部署"，流程即开始运行。

28. 系统配置

核心配置文件

硬件与功能配置集中存放于 `/etc/board_info/` 目录的出厂配置文件中，主要包括：

配置项	说明
model	型号（EG640-6X4Y / EG640-8Y / EG644 等）
firmware_ver	固件版本（当前 1.1）
machine_id	机型标识（当前 007）
GPIO 映射	DI/DO、LED、按键等引脚定义
串口 / CAN	uart1~5、can0/can1 参数（波特率等）
网络	网口、Wi-Fi、4G 相关配置
存储	eMMC / TF 卡相关配置
看门狗	超时参数（如看门狗超时 27 s）

查看当前配置（只读参考）

```
cat /etc/board_info/*.json
```

重要：普通用户无需直接编辑出厂配置文件。修改配置会导致部分功能异常。

29. 系统服务与开机自启动

29.1 出厂预置 systemd 服务

服务名	Type	Restart	ExecStart / 行为	说明
edge-base-wdt-kick.service	simple	always	内部喂狗守护脚本	GPIO20 (WDT) 每约 6 秒输出一次脉冲
edge-led-running.service	simple	always	内部运行灯守护脚本	LED_RUN 0.5 Hz 闪烁
edge-boot-hook.service	oneshot	on-failure	/usr/local/bin/edge-boot-hook.sh	每次上电执行用户钩子脚本（失败时自动重试）

29.2 特殊服务说明

- **edge-boot-hook.service**: 每次上电都会执行 `/usr/local/bin/edge-boot-hook.sh`, 用户可将随系统启动的脚本逻辑直接写入该脚本, 或在该脚本中调用自己的脚本。

```
cat /usr/local/bin/edge-boot-hook.sh # 查看钩子脚本内容
systemctl status edge-boot-hook.service
```

29.3 自定义开机自启动

方法一：编写自定义 systemd 服务（推荐）

在 `/etc/systemd/system/` 下新建服务文件，例如 `myapp.service`:

```
[Unit]
Description=My custom application
After=network.target

[Service]
Type=simple
ExecStart=/usr/local/bin/myapp
Restart=always
RestartSec=5

[Install]
WantedBy=multi-user.target
systemctl daemon-reload
systemctl enable --now myapp.service
```

方法二：利用 `edge-boot-hook.sh`

将自启动逻辑追加到 `/usr/local/bin/edge-boot-hook.sh` 末尾（或在该脚本中调用自己的脚本），即可实现每次上电自动执行：

```
echo "/usr/local/bin/myapp &" >> /usr/local/bin/edge-boot-hook.sh
```

- 注意：/usr/local/bin/edge-boot-hook.sh 为上电任务钩子脚本，内部承担 4G 模组电源上电脉冲控制（对电源使能引脚输出时序）。追加内容时请仅追加、不要覆盖或修改脚本原有内容；修改前建议先备份原脚本（cp /usr/local/bin/edge-boot-hook.sh /usr/local/bin/edge-boot-hook.sh.bak）。若脚本被错误覆盖或改写，可能导致 4G 模组无法正常上电。

方法三：使用 rc.local（传统方式）

设备已启用 rc-local.service（开机执行 /etc/rc.local），可在该脚本 exit 0 之前添加需要开机自动执行的命令：

```
#!/bin/bash
# 在 exit 0 之前添加你的命令，例如：
/usr/local/bin/myapp &
exit 0
```

- 编辑方式：vi /etc/rc.local（修改后确保脚本可执行：chmod +x /etc/rc.local）。
- 注意：长时间运行的命令请加 & 放入后台，避免阻塞开机流程；exit 0 必须保留在末尾。
- 查看执行状态：systemctl status rc-local.service。
- 该脚本在 network.target 之后执行（此时网络已就绪），适合需要网络环境的启动逻辑。

- 注意：若同时使用方法二（edge-boot-hook.sh），两者都会在开机时执行，请避免重复启动同一程序。
- 提示：方法一具备完整的依赖与重启管理，推荐用于正式业务程序；方法二适合轻量脚本；方法三为传统 rc.local 方式，适合简单命令的快速开机执行。

30. 常见问题 FAQ 与故障排查

30.1 网络

Q: LAN 口 IP 是多少? LAN 口(br-lan)默认地址为 192.168.5.1/24, PC 接入 LAN 口可自动获取 192.168.5.x 网段地址(见第 6 章)。

30.2 CAN

Q: 经典 CAN 配置报 Invalid argument? 出厂默认 fd on, 使用经典 CAN 时必须加 fd off, 否则会报 Invalid argument:

```
ip link set can0 down
ip link set can0 up type can bitrate 500000 fd off
```

详见第 11 章。

30.3 RS485

Q: 支持 9 位数据位吗? 不支持。9 位数据位为 16550A 驱动限制(见 9.2 参数范围)。

30.4 看门狗

Q: 设备无故自动重启? 请依次检查: ① 喂狗服务 edge-base-wdt-kick.service 是否在运行(systemctl status edge-base-wdt-kick.service); ② 看门狗拨码开关是否处于使能状态; ③ 是否关闭过喂狗服务且拨码使能(约 27s 后复位)。详见第 23 章。

30.5 WiFi

Q: AP 模式打不开? 请使用 17.5 节的推荐用法(NetworkManager 托管方式), 勿使用 nmcli device wifi hotspot 命令。Web 界面不支持配置 AP 模式, 另请注意能力边界: 仅 2.4 GHz、加密仅 WPA2-PSK、STA 与 AP 不能并发。

30.6 4G

Q: eth2 无地址? 请依次检查: SIM 卡是否插好、是否欠费停机、天线是否接好(见 18.4 常见问题)。

30.7 TF 卡

Q: 插入 TF 卡未自动挂载? 请依次检查: ① udev 自动挂载规则是否存在(/etc/udev/rules.d/99-tf-automount.rules); ② TF 卡分区格式是否为 FAT32/ext4 等支持格式; ③ 手动挂载测试(见第 22 章)。

30.8 日志与排障

Q: 如何查看系统日志?

```
dmesg # 内核日志
journalctl -b # 本次开机全部日志
```

Q: 如何导出日志? 通过 Web 界面"系统"页面"日志导出"功能导出; 命令行可用 journalctl -b > /mnt/tfcard/syslog.txt 将本次开

销售网络

智嵌物联，让连接更稳定！

企业愿景：成为国内物联网设备首选品牌！

企业使命：为客户利益而努力创新，为推动工业物联网发展而不懈奋斗！

产品理念：稳定！稳定！还是稳定！

服务理念：客户在哪里，我们就在哪里！



深圳总部

地址：广东省深圳市宝安区新桥街道新桥社区

新和大道 6-18 号 1305

网址：www.zhiqwl.com

电话：0755-23203231



天猫店铺



淘宝店铺



京东店铺



微信公众号



公司官网