4G??????

移远通信 LTE Standard EM05系列是一款专为 IoT/M2M应用而设计的 LTE Cat 4模块。采用节省空间的 M.2（NGFF）封装类型，超薄、紧凑的设计使其更易于嵌入到小尺寸产品中。

EM05系列支持最大下行速率 150 Mbps，最大上行速率 50 Mbps,包含三个型号: EM05-CN、EM05-E和 EM05-G。支持 LTE-FDD、LTE-TDD、DC-HSDPA、HSPA+、HSDPA、HSUPA、WCDMA和 CDMA 等多种网络制式。

要在 Jetson系列上使用 EM-05模块需要重新编译系统内核并刷入新编译的系统。

?????????

1.1 ????????

1.2 ?????????

1.3 ????????????

1.4 ?????????

????????????

2.1 ????????

- 安装编译依赖工具

```
sudo apt install build-essential bc git bison flex libssl-dev zip libncurses-dev make git
```

- 创建工具链放置目录

```
mkdir $HOME/l4t-gcc-toolchain
cd $HOME/l4t-gcc-toolchain
```

- 下载并解压交叉编译工具链

```
wget https://developer.nvidia.com/downloads/embedded/l4t/r36_release_v3.0/toolchain/aarch64--glibc--stable-2022.08-1.tar.bz2
tar xf aarch64--glibc--stable-2022.08-1.tar.bz2
```

- 进入到缓存好的刷机固件目录

```
cd
$HOME/nvidia/nvidia_sdk/JetPack_6.2.1_Linux_JETSON_ORIN_NANO_TARGETS/Linux_for_Tegra/source  #
□□□□□□□□□□□
```

- □□ release tag □□□□□ Jetson Linux Archive | NVIDIA Developer

```
 ./source_sync.sh -k -t <release-tag> #<release-tag>□□□□□□□□□□□□□□ jetson_36.4.3
```

2.2 ??????

> □□□□□□□□ □□ Linux&Andriod□□□□□□

Jetson Orin Nano□□□□□□□ arch/arm64/configs/defconfig

- □□ USB□□□□□□□□□□□

```
CONFIG_USB_SERIAL=y
CONFIG_USB_SERIAL_WWAN=y
CONFIG_USB_SERIAL_OPTION=y
CONFIG_USB_NET_DRIVERS=y
CONFIG_USB_USBNET=y
CONFIG_USB_NET_QMI_WWAN=y
CONFIG_USB_WDM=y
```

- □□ drivers/usb/serial/option.c□□ USB□□□□□□□ VID□ PID

```
static const struct usb_device_id option_ids[] = {
#if 1 // 2025-04-24 Added by Quectel
    { USB_DEVICE(0x2C7C, 0x0125) },
#endif
    ... ...
}
```

- □□ USBNet□□□□□□□ drivers/usb/serial/option.c

```
static int option_probe(struct usb_serial *serial,
            const struct usb_device_id *id)
{
    ... ...
```

```
#if 1  // 2025-04-24 Added by Quectel
if (serial->dev->descriptor.idVendor == cpu_to_le16(0x2C7C)) {
        __u16 idProduct = le16_to_cpu(serial->dev->descriptor.idProduct);
        struct usb_interface_descriptor *intf = &serial->interface->cur_altsetting->desc;

        if (intf->bInterfaceClass != 0xFF || intf->bInterfaceSubClass == 0x42) {
                //ECM, RNDIS, NCM, MBIM, ACM, UAC, ADB
                return -ENODEV;
        }

        if ((idProduct&0xF000) == 0x0000) {
                //MDM interface 4 is QMI
                if (intf->bInterfaceNumber == 4 && intf->bNumEndpoints == 3
                        && intf->bInterfaceSubClass == 0xFF && intf->bInterfaceProtocol ==
0xFF)
                        return -ENODEV;
        }
}
#endif

    /* Store the device flags so we can use them during attach. */
    usb_set_serial_data(serial, (void *)device_flags);

    return 0;
}
```

- □□□□□□□□□□□□ drivers/usb/serial/usb_wwan.c

```
static struct urb *usb_wwan_setup_urb(struct usb_serial_port *port,
                      int endpoint,
                      int dir, void *ctx, char *buf, int len,
                      void (*callback) (struct urb *))
{
 ... ...
    usb_fill_bulk_urb(urb, serial->dev,
              usb_sndbulkpipe(serial->dev, endpoint) | dir,
              buf, len, callback, ctx);

#if 1   //2025-04-24 Added by Quectel for zero packet
    if (dir == USB_DIR_OUT) {
```

```
        struct usb_device_descriptor *desc = &serial->dev->descriptor;

        if (desc->idVendor == cpu_to_le16(0x2C7C))
        urb->transfer_flags |= URB_ZERO_PACKET;
    }
#endif
    return urb;
}
```

- □□ Reset-resume□□□□□□□□□□□□ drivers/usb/serial/option.c

```
static struct usb_serial_driver option_1port_device = {
    ... ...
#ifdef CONFIG_PM
    .suspend           = usb_wwan_suspend,
    .resume            = usb_wwan_resume,
#if 1  //2025-04-24 Added by Quectel
    .reset_resume   = usb_wwan_resume,
#endif
#endif
};
```

- □□ QMI_WWAN□□
  □□□□□ QMI_WWAN□□□□□ qmi_wwan_q.c□□□□□□ drivers/net/usb/ □□□□□□□□ drivers/net/usb/Makefile□□□□□□ qmi_wwan_q.c

```
#Makefile
# must insert qmi_wwan_q.o before qmi_wwan.o
obj-${CONFIG_USB_NET_QMI_WWAN} += qmi_wwan_q.o
obj-${CONFIG_USB_NET_QMI_WWAN} += qmi_wwan.o
```

2.3 ????

> □□□□□□□□□□□□□□□□

- □□□□□□

```
cd ../../
mkdir kernel_out
```

- □□□□□□

```
export CROSS_COMPILE=$HOME/l4t-gcc-toolchain/aarch64--glibc--stable-2022.08-1/bin/aarch64-buildroot-linux-gnu-        #□□□□□□□□□
make -C kernel                      #□□ Jetson Linux □□□□
sudo -E make install -C kernel     #□□□□□□□□□□□□
```

- □□□□□□□□□□□□□□

```
cp kernel/kernel-jammy-src/arch/arm64/boot/Image ../Linux_for_Tegra/kernel/Image
```

- □□ NVIDIA□□□□□□□□□□

```
export CROSS_COMPILE=$HOME/l4t-gcc-toolchain/aarch64--glibc--stable-2022.08-1/bin/aarch64-buildroot-linux-gnu-
export KERNEL_HEADERS=$PWD/kernel/kernel-jammy-src
make modules
```

- □□□□□□□□

```
export INSTALL_MOD_PATH=$HOME/nvidia/nvidia_sdk/JetPack_6.2.1_Linux_JETSON_ORIN_NANO_TARGETS/Linux_for_Tegra/rootfs/ #□□□□□□□□□□□
sudo -E make modules_install
```

- □□□□ DTB

```
export CROSS_COMPILE=$HOME/l4t-gcc-toolchain/aarch64--glibc--stable-2022.08-1/bin/aarch64-buildroot-linux-gnu-
export KERNEL_HEADERS=$PWD/kernel/kernel-jammy-src
make dtbs
```

- □□□□ dtb□□□□□□□□□

```
cp kernel-devicetree/generic-dts/dtbs/* ../kernel/dtb/
```

2.4 ????

- □□□□□□□□□

```
cd $HOME/nvidia/nvidia_sdk/JetPack_6.2.1_Linux_JETSON_ORIN_NANO_TARGETS/Linux_for_Tegra/source  #
```

□□□□□□□□□□□

□□□□□□□□

- super□□

```
sudo ./tools/kernel_flash/l4t_initrd_flash.sh --external-device nvme0n1p1 -c
tools/kernel_flash/flash_l4t_t234_nvme.xml -p "-c bootloader/generic/cfg/flash_t234_qspi.xml"
--showlogs --network usb0 jetson-orin-nano-devkit-super internal
```

- □□□□

```
sudo ./flash.sh jetson-orin-nano-devkit-nvme internal
```

2.5 ????

- □ Quectel_Qconnectmanager□□□□□
- □□□□□

```
unizp Quectel_QConnectManager_Linux_V1.6.5.zip
cd Quectel_QConnectManager_Linux_V1.6.5
make
```

□□□□□□ 4G□□□□

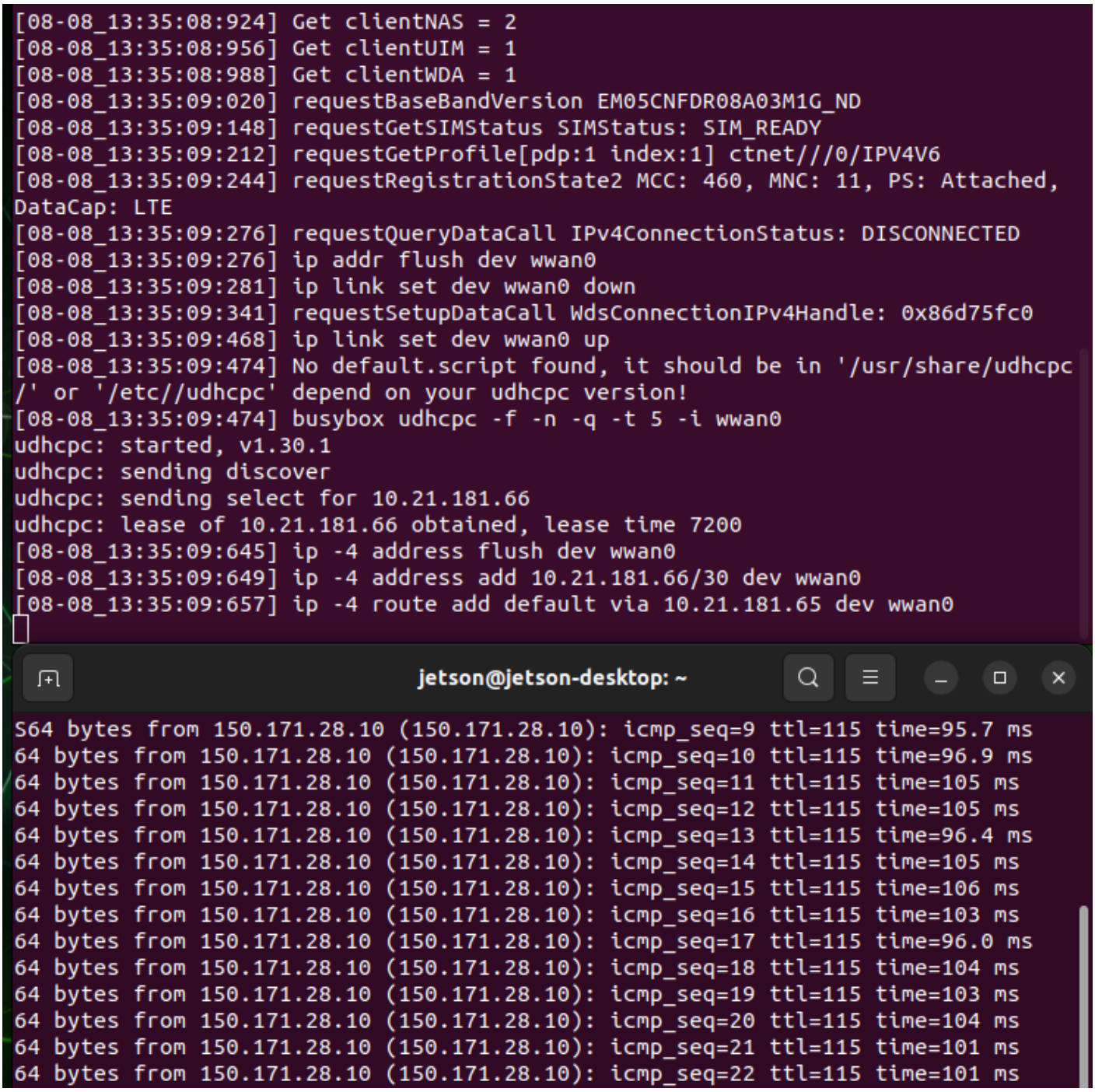
```
jetson@jetson-desktop:~/Downloads/Quectel_QConnectManager_Linux_V1.6.5$ sudo ./quectel-CM
[08-08_11:39:04:415] QConnectManager_Linux_V1.6.5
[08-08_11:39:04:416] Find /sys/bus/usb/devices/1-2.2 idVendor=0x2c7c idProduct=0x125,
bus=0x001, dev=0x007
[08-08_11:39:04:416] Auto find qmichannel = /dev/cdc-wdm0
[08-08_11:39:04:417] Auto find usbnet_adapter = wwan0
[08-08_11:39:04:417] netcard driver = qmi_wwan_q, driver version = V1.2.6
[08-08_11:39:04:417] Modem works in QMI mode
[08-08_11:39:04:447] cdc_wdm_fd = 7
[08-08_11:39:04:523] Get clientWDS = 5
[08-08_11:39:04:557] Get clientDMS = 1
[08-08_11:39:04:589] Get clientNAS = 2
[08-08_11:39:04:620] Get clientUIM = 1
[08-08_11:39:04:653] Get clientWDA = 1
[08-08_11:39:04:684] requestBaseBandVersion EM05CNFDR08A03M1G_ND
[08-08_11:39:04:812] requestGetSIMStatus SIMStatus: SIM_READY
[08-08_11:39:04:876] requestGetProfile[pdp:1 index:1] ctnet///0/IPV4V6
```

```
[08-08_11:39:04:908] requestRegistrationState2 MCC: 460, MNC: 11, PS: Attached, DataCap: LTE
[08-08_11:39:04:940] requestQueryDataCall IPv4ConnectionStatus: DISCONNECTED
[08-08_11:39:04:941] ip addr flush dev wwan0
[08-08_11:39:04:947] ip link set dev wwan0 down
[08-08_11:39:05:003] requestSetupDataCall WdsConnectionIPv4Handle: 0x8723e530
[08-08_11:39:05:132] ip link set dev wwan0 up
[08-08_11:39:05:141] No default.script found, it should be in '/usr/share/udhcpc/' or
'/etc//udhcpc' depend on your udhcpc version!
[08-08_11:39:05:142] busybox udhcpc -f -n -q -t 5 -i wwan0
udhcpc: started, v1.30.1
udhcpc: sending discover
udhcpc: sending select for 10.21.181.66
udhcpc: lease of 10.21.181.66 obtained, lease time 7200
[08-08_11:39:05:282] ip -4 address flush dev wwan0
[08-08_11:39:05:286] ip -4 address add 10.21.181.66/30 dev wwan0
[08-08_11:39:05:292] ip -4 route add default via 10.21.181.65 dev wwan0
```

□□□□□

```
jetson@jetson-desktop:~$  ifconfig wwan0
wwan0: flags=193<UP,RUNNING,NOARP>  mtu 1500
        inet 10.21.181.66  netmask 255.255.255.252
        inet6 fe80::5804:41ff:feda:ce83  prefixlen 64  scopeid 0x20<link>
        ether 5a:04:41:da:ce:83  txqueuelen 1000  (Ethernet)
        RX packets 9  bytes 2304 (2.3 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 22  bytes 1854 (1.8 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

□□□□



□□□ #30
□ edit □□□ 5 □□ 2025 06:44:19
□ edit □□□ 28 □□ 2025 09:10:22