**Setting Up LTE Data using the Sixfab HAT**

By this point the reader should have a fully set up tailscale network, capable of providing the drone with a static IP for BVLOS use. The following documents the process of enabling the Sixfab LTE modem HAT on the Raspberry Pi 5. Some prerequisites at this point: The builder has inserted and activated their Sixfab Sim card via the Sixfab website. Properly checked the connection of the Sixfab antennae and board seating. We will be using the QMI mode of the LTE hat. QMI or Qualcomm MSM Interface is a binary protocol that provides better throughput and more efficient communication with the modem than the alternative method, Ethernet Control Model (ECM). It is lower level-giving faster speeds and more stability, which is important for drone streaming or remote control. Sixfabs PPP/QMI Cellular loT application officially supports QMI as well. QMI is also highly more resilient to modem reconnects and doesn't require a full network stack reset. This is important if the drone experiences momentary signal drops or transitions between LTE bands.

To install the proper divers follow the procedure:

| sudo apt install -y libqmi-utils udhcpc |
| --- |

* libqmi-utils : Tools to communicate over QMI
* Udhcpc : Lightweight client to grab an IP from the modem

Disable ModemManager if you have not already

| sudo systemctl stop ModemManagersudo systemctl disable ModemManager |
| --- |

Check to ensure ModemManager is down via

| systemctl status ModemManager |
| --- |

Identify the modem interface : *(A good response is something like /dev/cdc-wdm0)*

| ls /dev/cdc-wdm\* |
| --- |

Check the network interface : (*A good response is something like wwan0) LTE!*

| ls /sys/class/net/ |
| --- |

Set the device to come online:

| sudo qmicli -d /dev/cdc-wdm0 --device-open-proxy --dms-set-operating-mode=online |
| --- |

***Now we begin the process of manually configuring and starting the network***

| sudo nano /etc/qmi-network.conf |
| --- |

Write:

| APN=super DEVICE=/dev/cdc-wdm0 |
| --- |

* Originally we tested using the default “Internet” instead of “super” for QMI application we must use “super”

| sudo qmicli -d /dev/cdc-wdm0 --dms-set-operating-mode=reset |
| --- |

A quick reboot of the modem only, then we bring it back online with the following

sudo qmicli -d /dev/cdc-wdm0 --device-open-proxy --dms-set-operating-mode=online

**Finally we start up the network**

| sudo qmi-network /dev/cdc-wdm0 start |
| --- |

You should see Network Started Successfully.

Reboot the RPI

| sudo reboot |
| --- |

The following section outlines the automation of the LTE network. We will proceed by setting up the Pi such that it prefers wifi on boot, but if it is unable to connect, it falls back to cellular via QMI. We will create a systemd service that waits ~10 seconds after boot and checks if wifi is connected, if not the system starts the QMI connection

The script:

| sudo nano /usr/local/bin/qmi-fallback.sh |
| --- |

Write:

| #!/bin/bash# Wait for system to settle and boot up (still before MAVlink)sleep 10# Check if wlan0 is connectedif iw dev wlan0 link | grep -q 'Connected'; then echo "Wi-Fi is connected. Skipping QMI." exit 0fiecho "Wi-Fi not connected. Starting QMI..."# Set modem online modeqmicli -d /dev/cdc-wdm0 --device-open-proxy --dms-set-operating-mode=online# Start networkqmi-network /dev/cdc-wdm0 start# Request IPudhcpc -q -f -i wwan0 |
| --- |

Make the script excusable with chmod:

| sudo chmod +x /usr/local/bin/qmi-fallback.sh |
| --- |

 Create a systemd service to perform the script on boot

| sudo nano /etc/systemd/system/qmi-fallback.service |
| --- |

Write to the file;

| [Unit]Description=QMI Cellular FallbackAfter=network.target[Service]Type=simpleExecStart=/usr/local/bin/qmi-fallback.shRestart=on-failure[Install]WantedBy=multi-user.target |
| --- |

Enable the service daemon:

| sudo systemctl daemon-reexecsudo systemctl daemon-reloadsudo systemctl enable qmi-fallback.service |
| --- |

*Reboot the pi - after reboot if you are on a wifi network, the pi should autoconnect to that network. If you boot the pi elsewhere, the LTE will kick in. You can verify the script ran via:*

| systemctl status qmi-fallback.service |
| --- |

*State DOWN is what you're looking for when on wifi, and UP for off wifi.*

*To test the program live, down the wifi and allow the service to kick in.*

| sudo ifconfig wlan0 down |
| --- |

Run the following or reboot and let the service handle it

| sudo /usr/local/bin/qmi-fallback.sh |
| --- |

Test using the GUI browser with *google.com or fast.com*