QNX - Over the Air (OTA) Demo

Anish Phadnis
Courtney Thirukumaran
Emmanuel Richmond
Huseyin Kabasakal
Manuel Osvaldo Lebron Flores

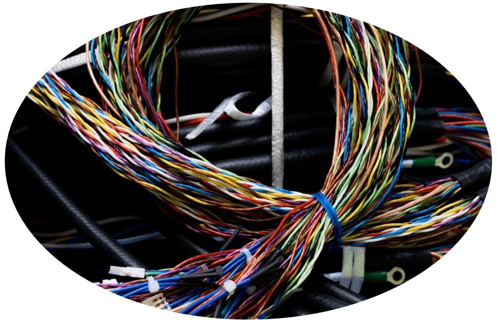
What problem are we solving?

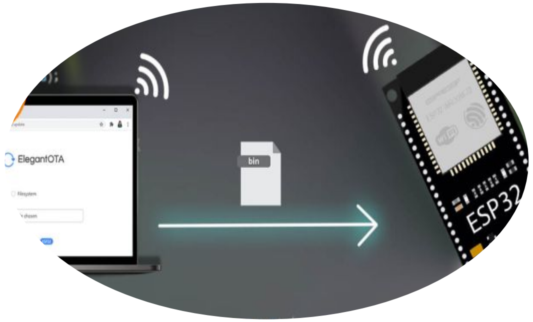
- Managing and updating software on devices can be challenging to do efficiently using traditional physical hardware-based solutions.
    - Especially when a device is located in a remote or hard-to-reach areas.

- Manual or physical hardware hookup systems can be **time-consuming** and **costly**.

- Over the air (OTA), will increase efficiency when attempting to improve existing systems.

- We will explore integration of the QNX operating system image loader and implementing an Over The Air (OTA) software solution on a Raspberry Pi device.

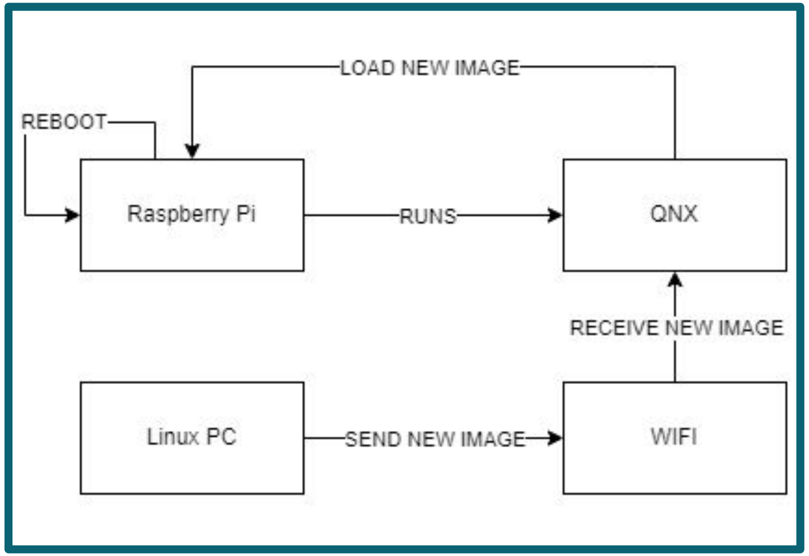




What is our Solution?

We have investigated and determined the easiest solution by completing the following steps to successfully send an update:

- Modify the original bin file which runs the original QNX version 7.1.
- This will involve having scripts mount an SD card and modify the files used to boot.
- The new files for the QNX 8.0 update will be placed here and then the full system will be rebooted.
- Entails scripts that work at a low level that modifies u-boot to load and install the new update directly.

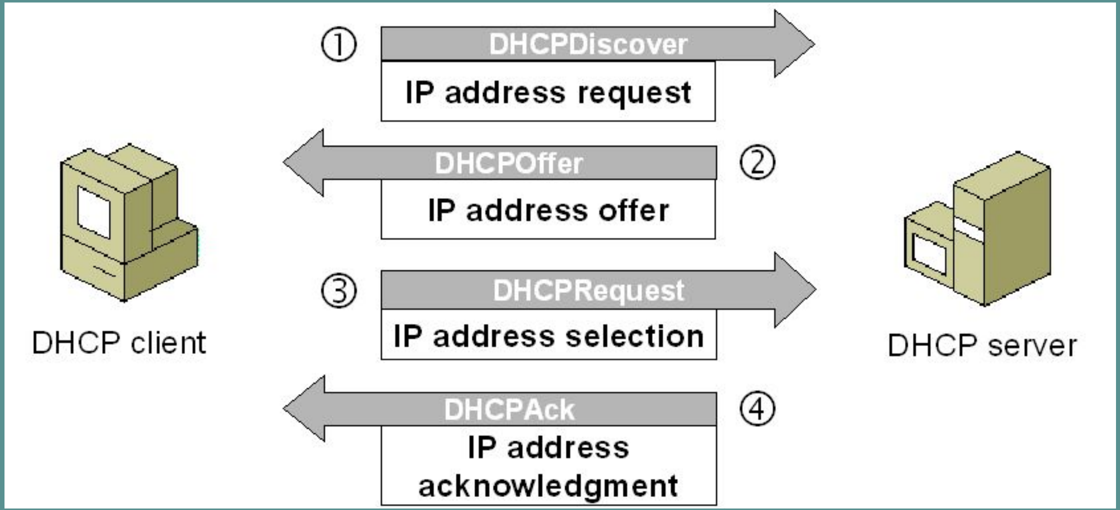


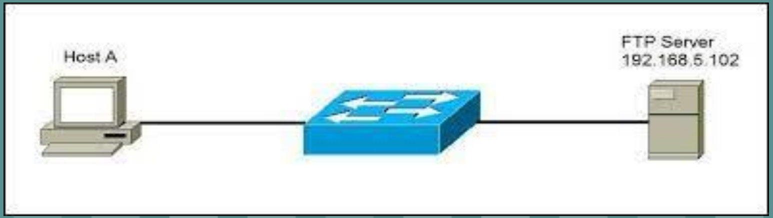
Live Demo

TFTP boot the Pi from QNX 7.1 to QNX 8.0 on Raspberry Pi!

- DHCP/TFTP server sends the new QNX image to the Pi
- U-boot pulls the new image and boots it
- Never saved the new image!!! (For now)

Custom DHCP/TFTP server





Mounting SD Card

- One barrier in the setup process..
- Detailing this process

Next Steps - Using Wifi

Instead of using a serial connection, we want to send the update using Wifi.

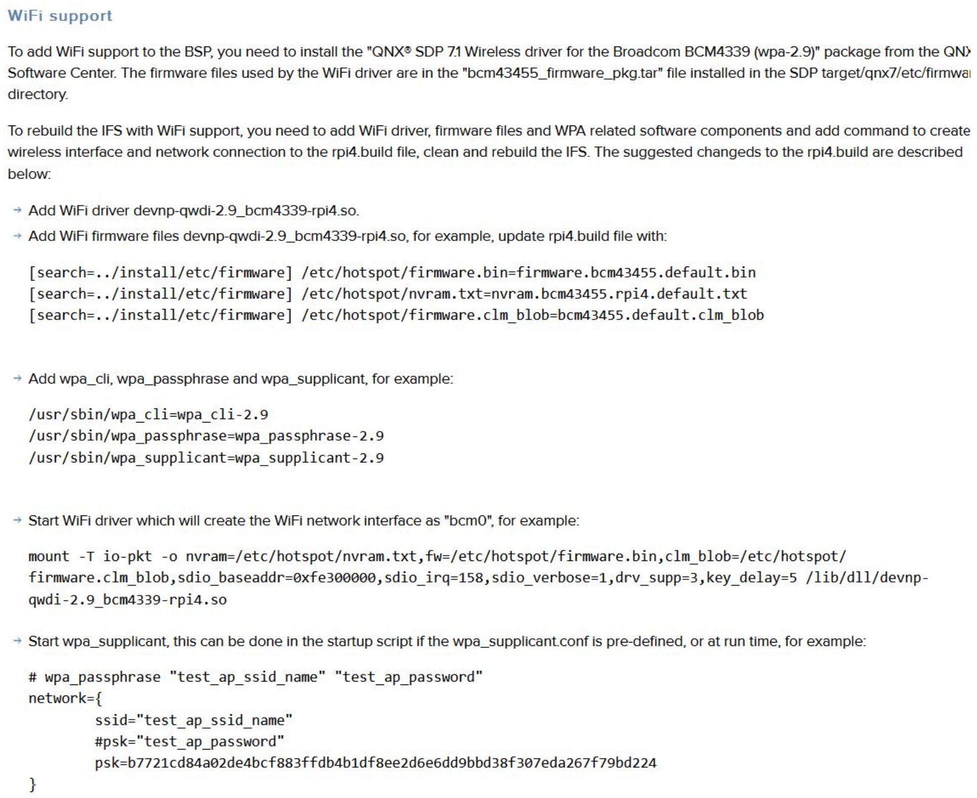
- ie. Over the Air

How are we proceeding?

- Currently have enabled wifi, working on setting up ssh!

```
void setup() {
 pinMode(13, OUTPUT);
}

void loop() {
 digitalWrite(13, HIGH);    delay(1000);
 digitalWrite(13, LOW);
}
```

Doesn’t work!



WiFi support

To add WiFi support to the BSP, you need to install the "QNX® SDP 7.1 Wireless driver for the Broadcom BCM4339 (wpa-2.9)" package from the QN Software Center. The firmware files used by the WiFi driver are in the "bcm43455_firmware_pkg.tar" file installed in the SDP target/qnx7/etc/firmwa directory.

To rebuild the IFS with WiFi support, you need to add WiFi driver, firmware files and WPA related software components and add command to create wireless interface and network connection to the rpi4.build file, clean and rebuild the IFS. The suggested changeds to the rpi4.build are described below:

- Add WiFi driver devnp-qwdi-2.9_bcm4339-rpi4.so.
- Add WiFi firmware files devnp-qwdi-2.9_bcm4339-rpi4.so, for example, update rpi4.build file with:

```
[search=../install/etc/firmware] /etc/hotspot/firmware.bin=firmware.bcm43455.default.bin
[search=../install/etc/firmware] /etc/hotspot/nvram.txt=nvram.bcm43455.rpi4.default.txt
[search=../install/etc/firmware] /etc/hotspot/firmware.clm_blob=bcm43455.default.clm_blob
```

- Add wpa_cli, wpa_passphrase and wpa_supplicant, for example:

```
/usr/sbin/wpa_cli=wpa_cli-2.9
/usr/sbin/wpa_passphrase=wpa_passphrase-2.9
/usr/sbin/wpa_supplicant=wpa_supplicant-2.9
```

- Start WiFi driver which will create the WiFi network interface as "bcm0", for example:

```
mount -T io-pkt -o nvram=/etc/hotspot/nvram.txt,fw=/etc/hotspot/firmware.bin,clm_blob=/etc/hotspot/
firmware.clm_blob,sdio_baseaddr=0xfe300000,sdio_irq=158,sdio_verbose=1,drv_supp=3,key_delay=5 /lib/dll/devnp-
qwdi-2.9_bcm4339-rpi4.so
```

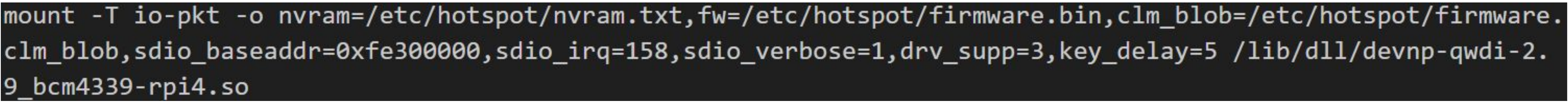
- Start wpa_supplicant, this can be done in the startup script if the wpa_supplicant.conf is pre-defined, or at run time, for example:

```
# wpa_passphrase "test_ap_ssid_name" "test_ap_password"
network={
        ssid="test_ap_ssid_name"
        #psk="test_ap_password"
        psk=b7721cd84a02de4bcf883ffdb4b1df8ee2d6e6dd9bbd38f307eda267f79bd224
}
```

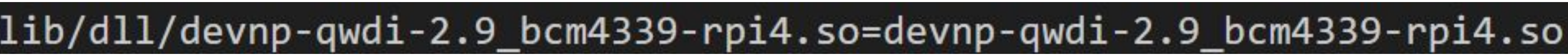
Compiling QNX Image

Wireless driver for the Broadcom BCM4339

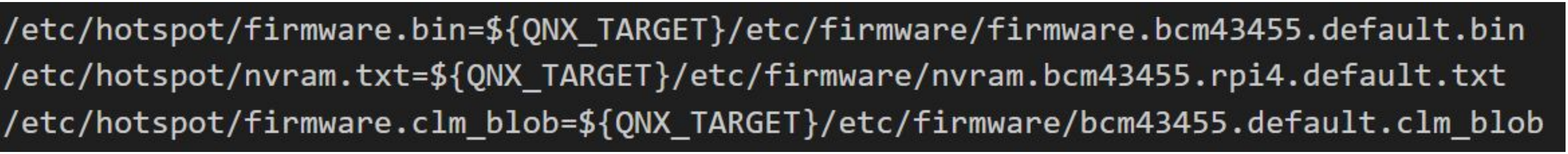
Wireless driver for BCM2711



```
mount -T io-pkt -o nvram=/etc/hotspot/nvram.txt,fw=/etc/hotspot/firmware.bin,clm_blob=/etc/hotspot/firmware.
clm_blob,sdio_baseaddr=0xfe300000,sdio_irq=158,sdio_verbose=1,drv_supp=3,key_delay=5 /lib/dll/devnp-qwdi-2.
9_bcm4339-rpi4.so
```



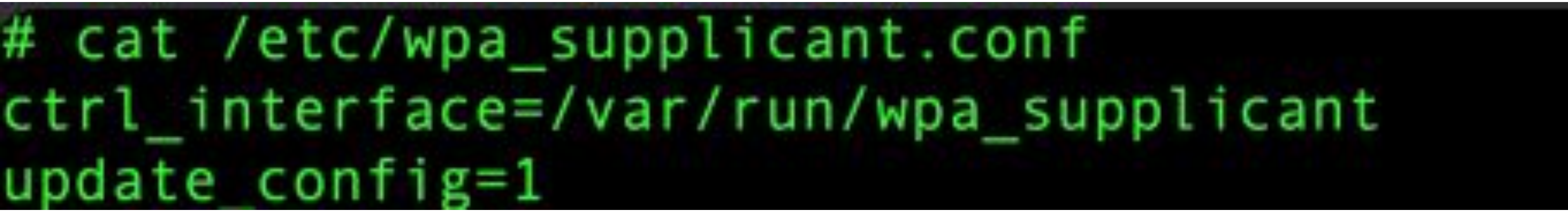
```
lib/dll/devnp-qwdi-2.9_bcm4339-rpi4.so=devnp-qwdi-2.9_bcm4339-rpi4.so
```



```
/etc/hotspot/firmware.bin=${QNX_TARGET}/etc/firmware/firmware.bcm43455.default.bin
/etc/hotspot/nvram.txt=${QNX_TARGET}/etc/firmware/nvram.bcm43455.rpi4.default.txt
/etc/hotspot/firmware.clm_blob=${QNX_TARGET}/etc/firmware/bcm43455.default.clm_blob
```

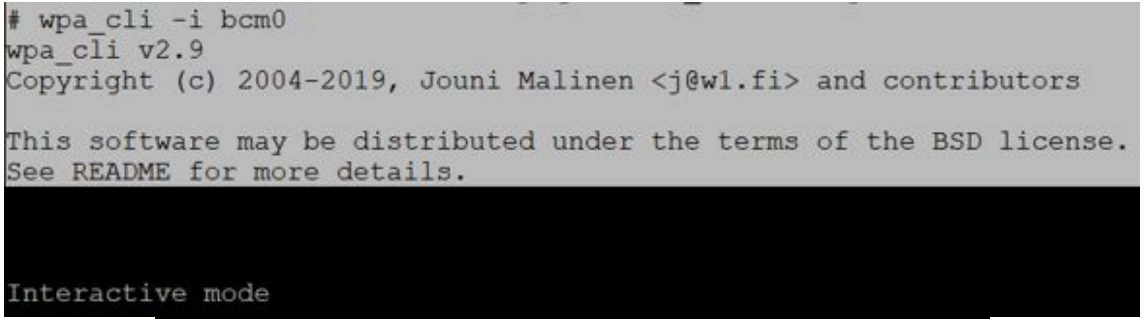
wpa_supplicant

wpa_supplicant -D qwdi -B -i bcm0 -c /etc/wpa_supplicant.conf

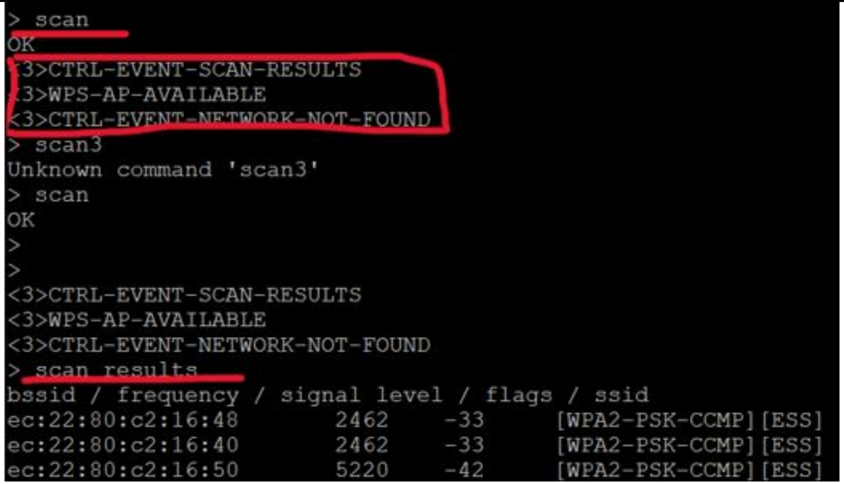


```
# cat /etc/wpa_supplicant.conf
ctrl_interface=/var/run/wpa_supplicant
update_config=1
```

wpa_cli

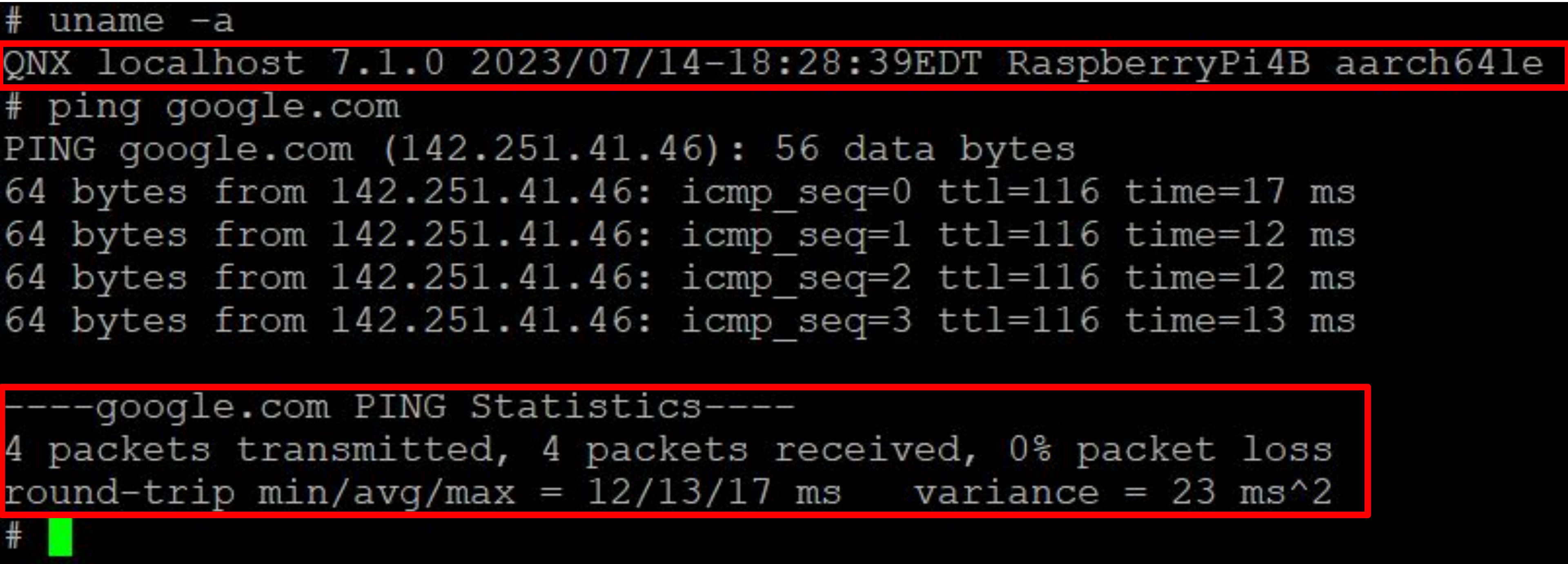


```
# wpa_cli -i bcm0
wpa_cli v2.9
Copyright (c) 2004-2019, Jouni Malinen <j@w1.fi> and contributors

This software may be distributed under the terms of the BSD license.
See README for more details.



Interactive mode
```



```
> scan
OK
<3>CTRL-EVENT-SCAN-RESULTS
<3>WPS-AP-AVAILABLE
<3>CTRL-EVENT-NETWORK-NOT-FOUND
> scan3
Unknown command 'scan3'
> scan
OK
>
>
<3>CTRL-EVENT-SCAN-RESULTS
<3>WPS-AP-AVAILABLE
<3>CTRL-EVENT-NETWORK-NOT-FOUND
> scan_results
bssid / frequency / signal level / flags / ssid
ec:22:80:c2:16:48       2462    -33     [WPA2-PSK-CCMP][ESS]
ec:22:80:c2:16:40       2462    -33     [WPA2-PSK-CCMP][ESS]
ec:22:80:c2:16:50       5220    -42     [WPA2-PSK-CCMP][ESS]
```

Working Wifi!



```
# uname -a
QNX localhost 7.1.0 2023/07/14-18:28:39EDT RaspberryPi4B aarch64le
# ping google.com
PING google.com (142.251.41.46): 56 data bytes
64 bytes from 142.251.41.46: icmp_seq=0 ttl=116 time=17 ms
64 bytes from 142.251.41.46: icmp_seq=1 ttl=116 time=12 ms
64 bytes from 142.251.41.46: icmp_seq=2 ttl=116 time=12 ms
64 bytes from 142.251.41.46: icmp_seq=3 ttl=116 time=13 ms

----google.com PING Statistics----
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max = 12/13/17 ms   variance = 23 ms^2
#
```

In the following weeks we hope to (and have now accomplished as of April 10)…

1. Enable ssh on QNX
2. Transfer image from PC to QNX wirelessly
3. Mount SD card to choose which image to run
4. Boot script on U-Boot to boot correct image
5. QNX program to initiate transfer and restart with new image

Future Work & Next Steps

We can take this project plan to new level by supporting other wifi protocols and cellular networks. Here are some possible ways in doing so (at a high level):

Supporting Other WiFi Protocols:

- Identify the specific WiFi protocols you want to support (e.g. 802.11ax).
- Update your scripts and configurations support these protocols by looking at documentation (eg. RFC's).
- Modify the WiFi driver to accommodate the new protocols.
- Test the updated scripts and configurations to ensure compatibility and functionality.

Cellular Networks:

- Research the cellular network standards and protocols used in your target environment (e.g., LTE, 5G).
- Obtain the hardware (e.g., cellular modem) and software (e.g., drivers, firmware) to enable cellular connectivity on your device.
- Integrate the cellular modem with your Raspberry Pi device and configure the network settings.
- Develop scripts to maintain cellular connectivity, such as connecting to the network, handling data transmission, and managing network resources.

Thank you!