GENESIS - Learning Outcome & Shadow Project Summary Report: Wi-Fi Driver Integration

LTTS
GLOBAL
ENGINEERING
ACADEMY

L&T Technology Services

Details

Ver. Rel. No.	Release Date	Prepared. By	Reviewed By	To be Approved	Remarks/Revision Details
1.0	13/11/2020	Ashwin M (99002601), Bibek (99002473), Santhosh (99002552)	-	-	Shadow Project
1.1	23/12/2020	Ashwin M (99002601), Bibek (99002473), Santhosh (99002552)	-	-	Shadow Project

CONTENTS

LIST OF FIGURES

LIST OF TABLES

Shadow project - [Team]

TOPIC: Enablement of Wi-Fi on Linux/android target and debug the packet flow

Modules

- Applied SDLC
- Linux OS
- Embedded Linux

Topic and Subtopics

1. Applied SDLC
   - Requirements
   - Design
   - Test Plan

2. Linux OS
   - File System
   - Makefile

3. Embedded Linux
   - Building Kernel
   - Image file generation
   - Making SD Card bootable

Objectives & Requirements

2.1 Objective

- Understanding Wi-Fi concepts
- Enablement of Wi-Fi on Linux
- Debug the packet flow

Requirements

Table 1 High Level Requirements

ID	Description
HL_01	TI PROCESSOR-SDK-LINUX-AM335X 06.03.00.106
HL_02	TI AM335x EVM board
HL_03	WireShark Packet capturing tool

Table 2 Low Level Requirements

ID	Description
HL_01_L1	Configuring the Kernel for TI WLAN Drivers
HL_01_L2	Generation of Image file for board with Wi-Fi enabled
HL_01_L3	create-sdcard.sh script file for partitioning SD Card
HL_02_L1	Bringing up WLAN on the board
HL_03_L1	Packet capture using Wireshark

Design

Block Diagram

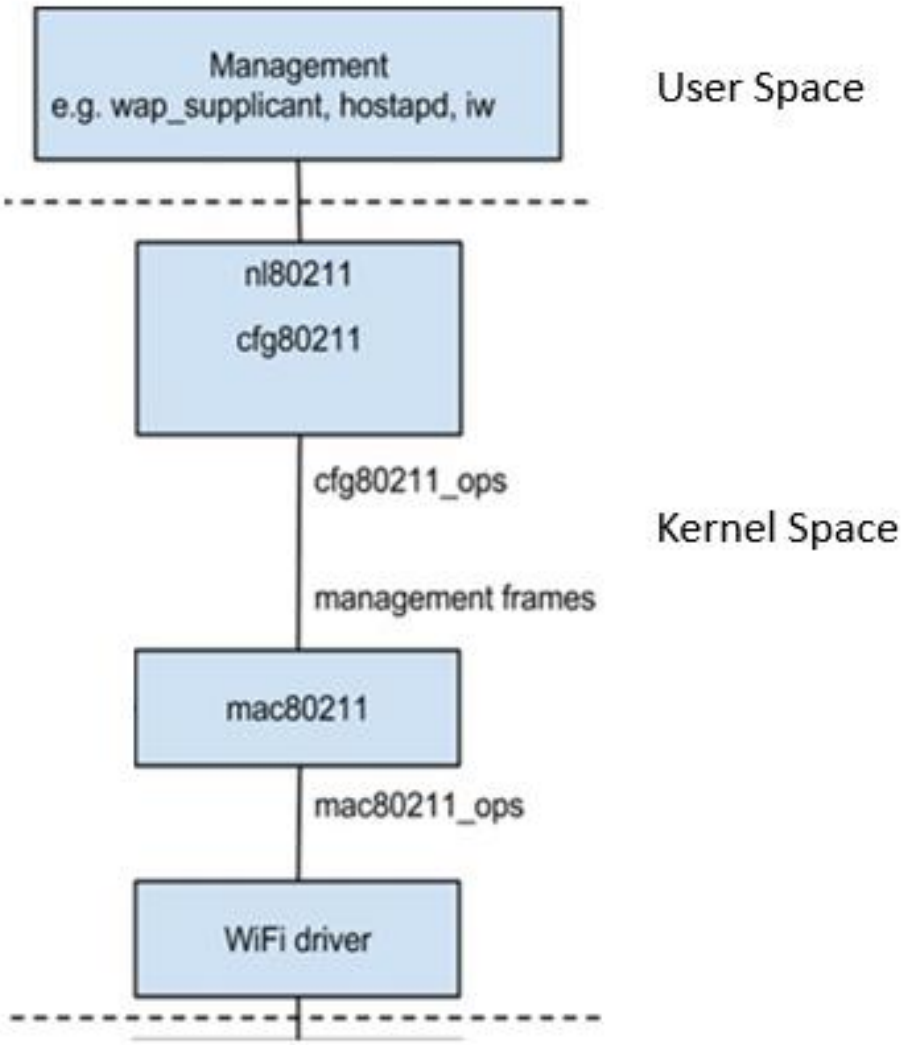

Figure 1 Block diagram of the system

Test Plan

Table 3 Test plan

ID	DESCRIPTION	PRE-CONDITION	EXPECTED INPUT	EXPECTED OUTPUT	ACTUAL OUTPUT
HL_02_L1_T1	Check for the type of network interface		Type command: ifconfig	Should display networks available	wlxc04a0028f97e
HL_02_L1_T2	Bringing up the WLAN	Wlan network	Type command: <network name> up. Example: ifconfig wlxc04a0028f97e up	Wlcore firmware should be booted	Wlcore firmware should be booted
HL_02_L1_T3	Scanning for Aps	Wlan enabled	Type command: iwlist wlxc04a0028f97e scan	Display the available wi-fi Aps within the range	Display the available wi-fi Aps within the range
HL-02_L1_T4	Connecting with the AP	Passcode for the AP	Type command: wps_passphrase <SSID-name> <Password>	Should connect to the SSID and then authentication and association process starts	Connecting to Realme2 SSID
HL_02_L1_T5	Checking whether PC is connected to AP	-	Type command: iwconfig	Information about the AP	Information about the AP
HL_02_L1_T6	Requesting IP address of the AP	-	Type command: dhclient wlxc04a0028f97e	Ip address of the AP	Ip address of the AP
HL_02_L1_T7	Checking the Wi-Fi connectivity by pinging google server	-	Type command: ping 8.8.8.8	Shows the connectivity with google server	Shows the connectivity with google server
HL_03_L1_T1	Packet capture using Wireshark	Wireshark tool	Connect to Wi-Fi	Capturing all frames	Capturing all frames

Implementation Summary

Cross compile tool-chain

- In /ti-processor-sdk-linux-am65xx-evm-07_00_01_06/board-support/linux-5.4.40+gitAUTOINC+66cf445b76-g66cf445b76 execute the following commands
  - GCC92PATH=$HOME/gcc-arm-9.2-2019.12-x86_64-aarch64-none-linux-gnu/bin:$HOME/gcc-arm-9.2-2019.12-x86_64-arm-none-linux-gnueabihf/bin
  - export PATH=$GCC92PATH:$PATH

Building the kernel

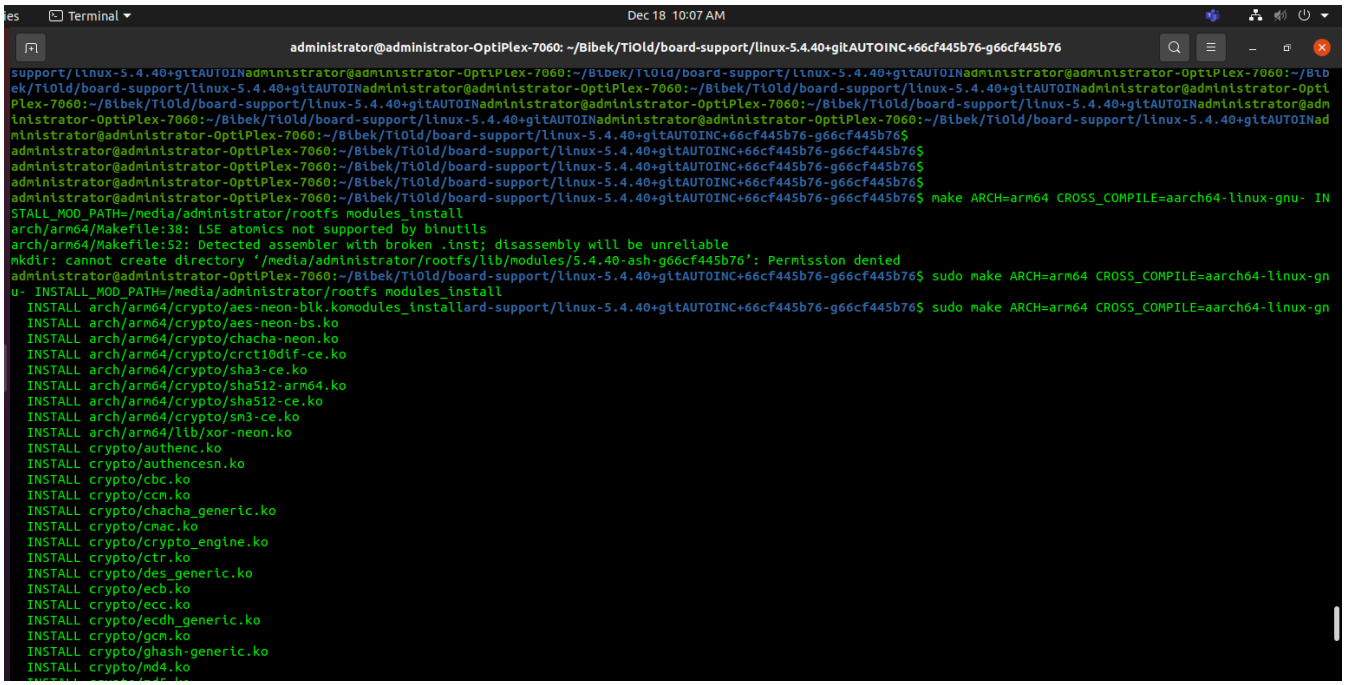

Figure 2 Building the Kernel

Figure 3 Newly generated Image

Flashing the image into the SD Card

- sudo ./bin/mksdboot.sh --device /dev/sdb --sdk .

```
15224831):
Created a new partition 1 of type 'Linux' and of size 62 MiB.

Command (m for help): Partition type
   p   primary (1 primary, 0 extended, 3 free)
   e   extended (container for logical partitions)
Select (default p): Partition number (2-4, default 2): First sector (129024-15224831, default 129024): Last sector, +/-sectors or +/-size{K,M,G,T
fault 15224831):
Created a new partition 2 of type 'Linux' and of size 7.2 GiB.
Partition #2 contains a ext4 signature.

Command (m for help): Partition number (1,2, default 2): Hex code (type L to list all codes):
Changed type of partition 'Linux' to 'W95 FAT32 (LBA)'.

Command (m for help): Partition number (1,2, default 2):
The bootable flag on partition 1 is enabled now.

Command (m for help): The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.

ls: cannot access '/dev/sdb2': No such file or directory
unmounting device '/dev/sdb'
Formatting /dev/sdb ...
mkfs.fat 4.1 (2017-01-24)
mkfs.fat: warning - lowercase labels might not work properly with DOS or Windows
mke2fs 1.45.5 (07-Jan-2020)
/dev/sdb2 contains a ext4 file system labelled 'rootfs'
        last mounted on /tmp/sdk/117093/rootfs on Thu Dec 17 14:33:41 2020
Proceed anyway? (y,N) y
Creating filesystem with 1886976 4k blocks and 472352 inodes
Filesystem UUID: 0c1c96c0-1c01-44c7-a978-f9f061a5b1b3
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done

Partitioning and formatting completed!
Copying filesystem on /dev/sdb1, /dev/sdb2
unmounting /dev/sdb1, /dev/sdb2
completed!
```

Figure 4 Partitioning the SD Card and flashing the Image

Porting CFG and MAC drivers

- sudo make ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu- menuconfig
- Enable the following options in the kernel configuration as well as specific device drivers
- [*] Network Support -----> [CONFIG_NET]
  - [*] Wireless -----> [CONFIG_WIRELESS]
    - <M> cfg80211 – wireless configuration API -----> [CONFIG_CFG80211]
      - [*] cfg80211 wireless extensions compatibility ------> [CONFIG_CFG80211_WEXT]
    - <M> Generic IEEE 802.11 Networking stack (mac80211) -----> [CONFIG_MAC80211]

```
nfig - Linux/arm64 5.4.40 Kernel Configuration
Networking support > Wireless
                                    Wireless
Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty submenus ----).  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
<M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable

                                    --- Wireless
                                    <M>   cfg80211 - wireless configuration API
                                    [*]     nl80211 testmode command
                                    [ ]     enable developer warnings
                                    [ ]     cfg80211 certification onus
                                    [*]     enable powersave by default
                                    [ ]     cfg80211 DebugFS entries
                                    [*]     support CRDA
                                    [*]     cfg80211 wireless extensions compatibility
                                    <M>   Generic IEEE 802.11 Networking Stack (mac80211)
                                    [*]   Minstrel
                                          Default rate control algorithm (Minstrel)  --->
                                    [*]   Enable mac80211 mesh networking support
                                    -*-   Enable LED triggers
                                    -*-   Export mac80211 internals in DebugFS
                                    [ ]   Trace all mac80211 debug messages
                                    [ ]   Select mac80211 debugging features  ----
```

Figure 5 Enabling cfg and mac drivers

- Device Drivers

- [*] Network device support ------> [CONFIG_NETDEVICES]
  - [*] Wireless LAN ------> [CONFIG_WLAN]
- Enable the Atheros/Qualcomm devices since the Wi-Fi used in this project is ATHEROS manufactured
  - [*] Atheros/Qualcomm devices
  - Enable all Atheros related drivers

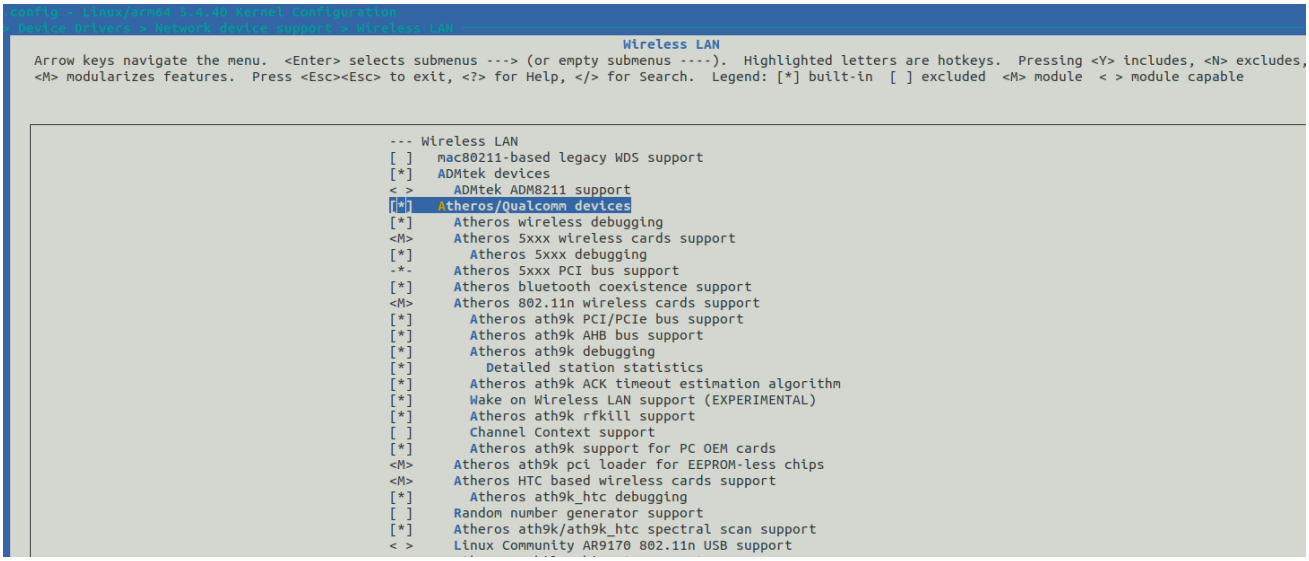

Figure 6 Atheros related drivers

Porting USB-Wi-Fi driver

- Device Drivers
  - [*] Network device support
    - [M] USB Network Adapters

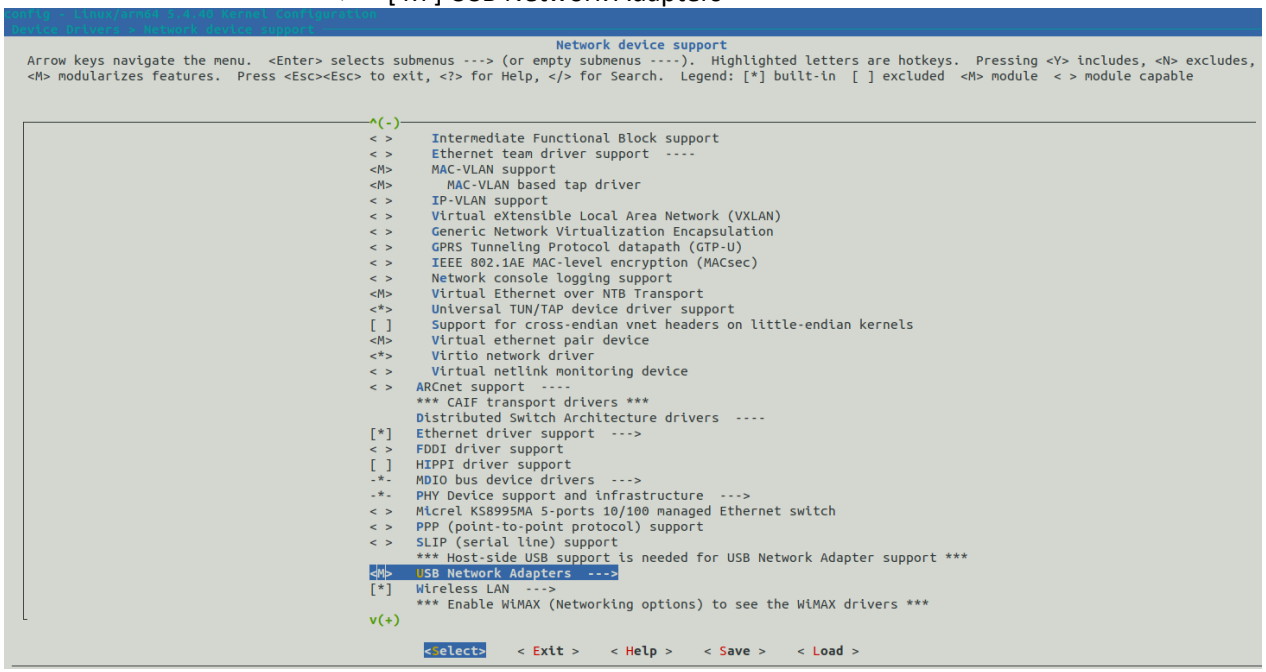

Figure 7 Enabling USB network support

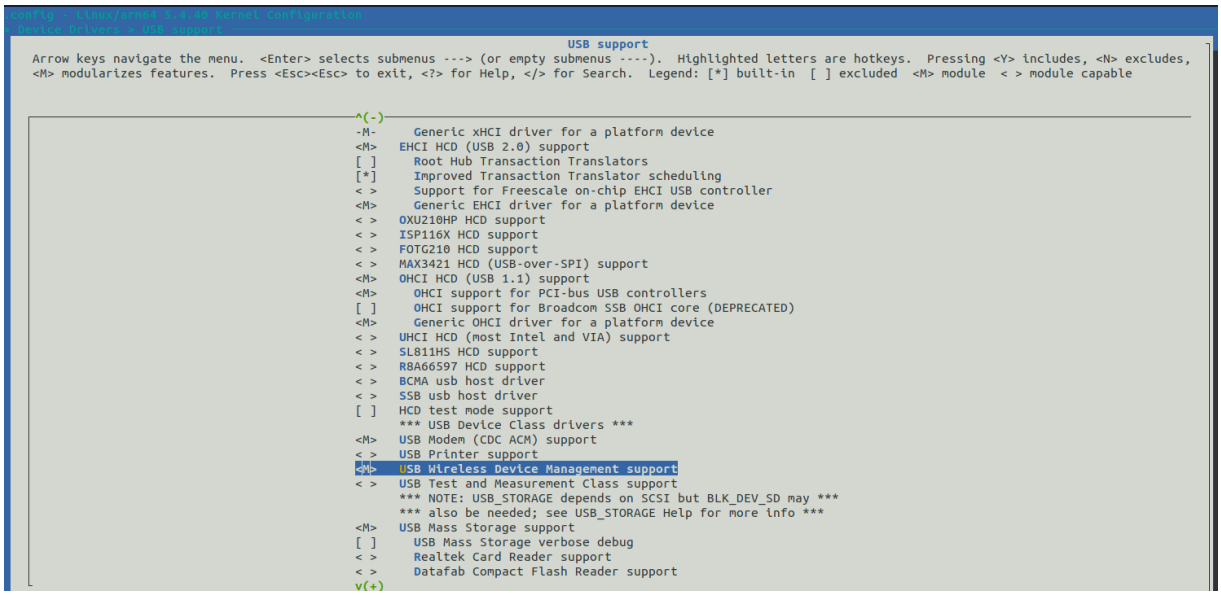

Figure 8 Enabling USB wireless management system

Re-build the Kernel and execute the following commands to copy the files to the SD Card

- make ARCH=arm64 CROSS_COMPILE=aarch64-none-linux-gnu- -j6
- make ARCH=arm64 CROSS_COMPILE=aarch64-none-linux-gnu- -j6 modules
- make ARCH=arm64 CROSS_COMPILE=aarch64-none-linux-gnu - INSTALL_MOD_PATH=/media/administrator/rootfs modules_install

Copy the following file and replace in the destination directory mentioned

- Copy the Image from /ti-processor-sdk-linux-am65xx-evm-07_00_01_06/board-support/linux-5.4.40+gitAUTOINC+66cf445b76-g66cf445b76/arch/arm64/boot
    - sudo cp Image /media/administrator/rootfs/boot
- Create a gzip file of vmlinux (vmlinux.gz) by executing the following command
    - sudo gzip vmlinux (Install gzip – sudo apt install gzip)
- Copy vmlinux.gz from -/ti-processor-sdk-linux-am65xx-evm-07_00_01_06/board-support/linux-5.4.40+gitAUTOINC+66cf445b76-g66cf445b76/
    - sudo cp vmlinux.gz /media/administrator/rootfs/boot
- Copy dtb files: -ti-processor-sdk-linux-am65xx-evm-07_00_01_06/board-support/linux-5.4.40+gitAUTOINC+66cf445b76-g66cf445b76/arch/arm64/boot/dts/ti
    - sudo cp *.dtb* /media/administrator/rootfs/boot/

Note: -use sync command after copying the files to rootfs

- Boot the TI AM654 Evaluation module board

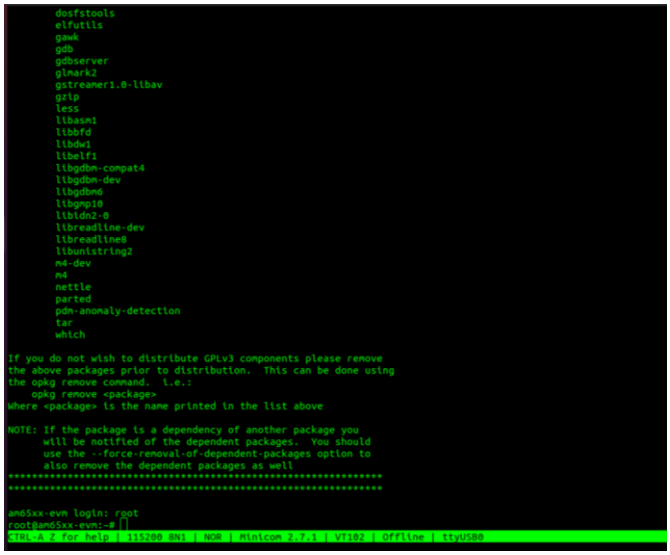

Figure 9 Booting the board

Connect the Wi-Fi dongle and observe the logs

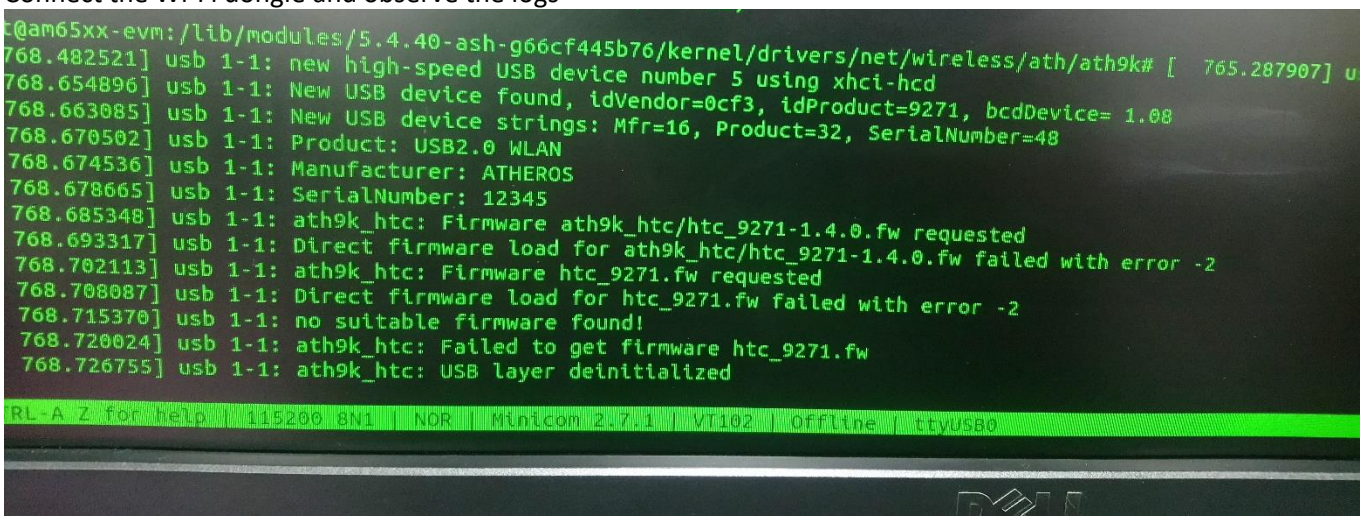

Figure 10 Wi-Fi dongle connection logs

Since there is no firmware that supports the dongle the board is not finding the firmware that matches with the connected dongle.

Download the corresponding firmware i.e., ath9k_htc

Figure 11 Atheros ath9k_htc firmware

Then copy these files to SD Card rootfs partition by executing the following command

- sudo cp -r ath9k_htc /media/administrator/rootfs1/lib/firmware

Re boot the board, connect the Wi-Fi dongle and observe the logs

```
root@am65xx-evm:~# [  146.353799] usb 1-1: new high-speed USB device number 3 using xhci-hcd
[  146.526161] usb 1-1: New USB device found, idVendor=0cf3, idProduct=9271, bcdDevice= 1.08
[  146.534348] usb 1-1: New USB device strings: Mfr=16, Product=32, SerialNumber=48
[  146.541749] usb 1-1: Product: USB2.0 WLAN
[  146.545775] usb 1-1: Manufacturer: ATHEROS
[  146.549886] usb 1-1: SerialNumber: 12345
[  146.556464] usb 1-1: ath9k_htc: Firmware ath9k_htc/htc_9271-1.4.0.fw requested
[  146.843792] usb 1-1: ath9k_htc: Transferred FW: ath9k_htc/htc_9271-1.4.0.fw, size: 51008
[  147.094330] ath9k_htc 1-1:1.0: ath9k_htc: HTC initialized with 33 credits
[  147.326966] ath9k_htc 1-1:1.0: ath9k_htc: FW Version: 1.4
[  147.332406] ath9k_htc 1-1:1.0: FW RMW support: On
[  147.337135] ath: EEPROM regdomain: 0x809c
[  147.341160] ath: EEPROM indicates we should expect a country code
[  147.347260] ath: doing EEPROM country->regdmn map search
[  147.352581] ath: country maps to regdmn code: 0x52
[  147.357379] ath: Country alpha2 being used: CN
[  147.361829] ath: Regpair used: 0x52
[  147.383196] ieee80211 phy1: Atheros AR9271 Rev:1

root@am65xx-evm:~#
```

Figure 12 Wi-Fi dongle connection logs

Note: To see the devices connected type "dmesg" or "lsusb" for USB based devices

Execute the following command to view the wireless network interface

- ifconfig

```
root@am65xx-evm:~# ifconfig
eth0      Link encap:Ethernet  HWaddr F4:84:4C:ED:38:32
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:82 errors:0 dropped:0 overruns:0 frame:0
          TX packets:82 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:6220 (6.0 KiB)  TX bytes:6220 (6.0 KiB)

wlan0     Link encap:Ethernet  HWaddr C0:4A:00:28:F9:7E
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

root@am65xx-evm:~#
CTRL-A Z for help | 115200 8N1 | NOR | Minicom 2.7.1 | VT102 | Off
```

Figure 13 Network interface

Disconnecting the Wi-Fi dongle and observe the logs:

```
root@am65xx-evm:~# [  155.976608] usb 1-1: USB disconnect, device number 3
[  155.982447] xhci-hcd xhci-hcd.2.auto: WARN Cannot submit Set TR Deq Ptr
[  155.989057] xhci-hcd xhci-hcd.2.auto: A Set TR Deq Ptr command is pending.
[  156.116051] ath: phy1: timeout (100000 us) on reg 0x7044: 0x00f6ea11 & 0x0000000f != 0x00000002
[  156.226578] ath: phy1: timeout (100000 us) on reg 0x806c: 0x40f6ea11 & 0x01f00000 != 0x00000000
[  156.235271] ath: phy1: RX failed to go idle in 10 ms RXSM=0x90060000
[  156.251749] ath: phy1: Failed to wakeup in 500us
[  156.358900] ath: phy1: timeout (100000 us) on reg 0x7000: 0x10f6ea11 & 0x00000003 != 0x00000000
[  156.469505] ath: phy1: timeout (100000 us) on reg 0x7000: 0x10f6ea11 & 0x00000003 != 0x00000000
[  156.579989] ath: phy1: timeout (100000 us) on reg 0x806c: 0x30f6ea11 & 0x01f00000 != 0x00000000
[  156.588681] ath: phy1: RX failed to go idle in 10 ms RXSM=0x90060000
[  156.678227] usb 1-1: ath9k_htc: USB layer deinitialized
```

Figure 14 Disconnecting the Wi-Fi dongle, logs

NOTE:

- **To connect to secured access point (AP) we need WPA (Wi-Fi Protected Access) daemon i.e., wpa_supplicant in our case.**
- **To show and manipulate the network configurations, we need wireless tools**

In TI AM65x EVM board, we are not able to install these packages and executed the rest of the project in the PC.

Connecting to a secured Access point via wpa_supplicant in Linux Ubuntu distribution using Wi-Fi dongle

- Enable wireless interface
  - make sure your wireless card is enabled. You can use rfkill
    - sudo apt install rfkill
  - To check the status of wireless card, run
    - rfkill list
  - To unblock it, use the following command
    - rfkill unblock wifi
  - If you are using the desktop version of Ubuntu, then you also need to stop Network Manager with the following command, otherwise it will cause connection problem when using wpa_supplicant.
    - sudo systemctl stop NetworkManager

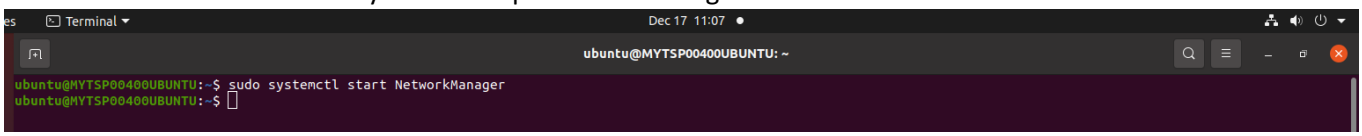

Figure 15 Starting the Network Manager

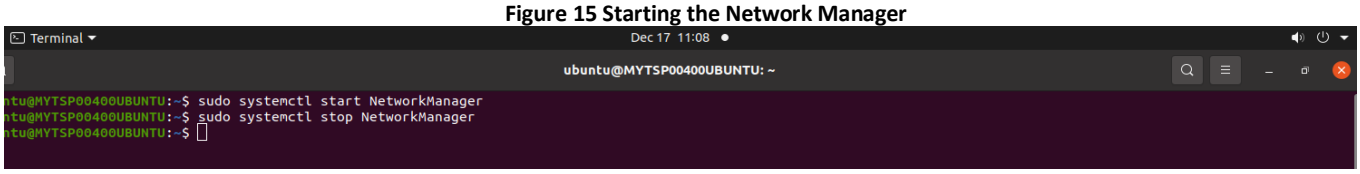

Figure 16 Stoping of Network Manager

    - sudo systemctl start NetworkManager ---> To start again

- sudo systemctl disable NetworkManager
- sudo systemctl enable NetworkManager ---> To enable again

- Find your wireless interface name and wireless network name
  - Run iwconfig to find the name of your wireless interface.
    - iwconfig

```
ubuntu@MYTSP00400UBUNTU:~$ iwconfig
wlxc04a0028f97e  IEEE 802.11  ESSID:off/any
          Mode:Managed  Access Point: Not-Associated   Tx-Power=off
          Retry short limit:7   RTS thr:off   Fragment thr:off
          Power Management:off

enp0s31f6  no wireless extensions.

lo        no wireless extensions.

ubuntu@MYTSP00400UBUNTU:~$
```

Figure 17 Wireless Network Interface

  - If your wireless interface isn't shown, perhaps you need to bring it up with the following command.
    - sudo ifconfig wlxc04a0028f97e up

  - Then find your wireless network name by scanning nearby networks with the command below
    - sudo iwlist wlxc04a0028f97e scan | grep ESSID

```
ubuntu@MYTSP00400UBUNTU:~$ sudo ifconfig wlxc04a0028f97e up
ubuntu@MYTSP00400UBUNTU:~$ sudo iwlist wlxc04a0028f97e scan | grep ESSID
                    ESSID:"TSGUEST"
                    ESSID:"LTTS"
                    ESSID:"LTTS-D"
                    ESSID:"Realme2"
                    ESSID:"GUEST-OS"
                    ESSID:"LTTS-CTEA"
                    ESSID:"LTTS"
                    ESSID:"LTTS-CTEA"
                    ESSID:"virus attack"
                    ESSID:"LTTS-CTEA"
                    ESSID:"LTTS-CTEA"
                    ESSID:"GUEST-OS"
                    ESSID:"GUEST-OS"
                    ESSID:"LTTS-D"
                    ESSID:"TSGUEST"
                    ESSID:"LTTS"
```

Figure 18 List of Access Points

Connect to Wi-fi network using wpa_supplicant

  - Now install wpa_supplicant on Ubuntu 20.04.
    - sudo apt install wpasupplicant
  - create a file named wpa_supplicant.conf using the wpa_passphrase utility

- wpa_supplicant.conf is the configuration file describing all networks that the user wants the computer to connect to
  - wpa_passphrase Realme2 12345678 | sudo tee /etc/wpa_supplicant.conf

```
ubuntu@MYTSP00400UBUNTU:~$ wpa_passphrase Realme2 12345678 | sudo tee /etc/wpa_supplicant.conf
network={
        ssid="Realme2"
        #psk="12345678"
        psk=b39e004a45ebdcdbe556d8e42fe024e5e33c73da44ed77b052559ab468fc89ad
}
ubuntu@MYTSP00400UBUNTU:~$
```

Figure 19 Creating wpa_supplicant.conf

- Connect wireless Dongle to wireless access point
  - sudo wpa_supplicant -c /etc/wpa_supplicant.conf -i wlxc04a0028f97e

```
ubuntu@MYTSP00400UBUNTU:~$ sudo wpa_supplicant -c /etc/wpa_supplicant.conf -i wlxc04a0028f97e
Successfully initialized wpa_supplicant
wlxc04a0028f97e: SME: Trying to authenticate with b8:c7:4a:bb:ef:7f (SSID='Realme2' freq=2412 MHz)
wlxc04a0028f97e: Trying to associate with b8:c7:4a:bb:ef:7f (SSID='Realme2' freq=2412 MHz)
wlxc04a0028f97e: Associated with b8:c7:4a:bb:ef:7f
wlxc04a0028f97e: CTRL-EVENT-SUBNET-STATUS-UPDATE status=0
wlxc04a0028f97e: WPA: Key negotiation completed with b8:c7:4a:bb:ef:7f [PTK=CCMP GTK=CCMP]
wlxc04a0028f97e: CTRL-EVENT-CONNECTED - Connection to b8:c7:4a:bb:ef:7f completed [id=0 id_str=]
```

Figure 20 Connecting to Access Point

- By default, wpa_supplicant runs in the foreground. If the connection is completed, then open up another terminal window and run
  - iwconfig

```
ubuntu@MYTSP00400UBUNTU:~$ iwconfig
wlxc04a0028f97e  IEEE 802.11  ESSID:"Realme2"
          Mode:Managed  Frequency:2.412 GHz  Access Point: B8:C7:4A:BB:EF:7F
          Bit Rate=57.8 Mb/s   Tx-Power=20 dBm
          Retry short limit:7   RTS thr:off   Fragment thr:off
          Power Management:off
          Link Quality=66/70  Signal level=-44 dBm
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:2   Missed beacon:0

enp0s31f6  no wireless extensions.

lo         no wireless extensions.
```

Figure 21 Connected to Access Point

  - You can see that the wireless interface is now associated with an access point.
- You can press CTRL+C to stop the current wpa_supplicant process and run it in the background by adding -B option.
  - sudo wpa_supplicant -B -c /etc/wpa_supplicant.conf -i wlxc04a0028f97e
- Although we're authenticated and connected to wireless network, but we don't have an IP address yet. To obtain a private IP address from DHCP server, use the following command:
  - sudo dhclient wlxc04a0028f97e

```
ubuntu@MYTSP00400UBUNTU:~$ sudo dhclient wlxc04a0028f97e
[sudo] password for ubuntu:
cmp: EOF on /tmp/tmp.rp2MMZdnRn which is empty
ubuntu@MYTSP00400UBUNTU:~$ ip addr show wlxc04a0028f97e
6: wlxc04a0028f97e: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether c0:4a:00:28:f9:7e brd ff:ff:ff:ff:ff:ff
    inet 192.168.43.253/24 brd 192.168.43.255 scope global dynamic wlxc04a0028f97e
       valid_lft 3575sec preferred_lft 3575sec
    inet6 2401:4900:26ba:840d:d1cd:a8af:853f:b766/64 scope global temporary dynamic
       valid_lft 3209sec preferred_lft 3209sec
    inet6 2401:4900:26ba:840d:c24a:ff:fe28:f97e/64 scope global dynamic mngtmpaddr
       valid_lft 3209sec preferred_lft 3209sec
    inet6 fe80::c24a:ff:fe28:f97e/64 scope link
       valid_lft forever preferred_lft forever
```

Figure 22 Assigning IP address

- Now your wireless interface has a private IP address, which can be shown with
  - ifconfig wlxc04a0028f97e

Testing the connection by Pinging to 8.8.8.8

```
ubuntu@MYTSP00400UBUNTU:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=117 time=14.0 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=117 time=13.2 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=117 time=14.1 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=117 time=13.4 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=117 time=13.7 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=117 time=14.1 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=117 time=13.5 ms
64 bytes from 8.8.8.8: icmp_seq=8 ttl=117 time=13.3 ms
64 bytes from 8.8.8.8: icmp_seq=9 ttl=117 time=14.3 ms
64 bytes from 8.8.8.8: icmp_seq=10 ttl=117 time=17.0 ms
64 bytes from 8.8.8.8: icmp_seq=11 ttl=117 time=14.7 ms
64 bytes from 8.8.8.8: icmp_seq=12 ttl=117 time=13.3 ms
64 bytes from 8.8.8.8: icmp_seq=13 ttl=117 time=13.6 ms
64 bytes from 8.8.8.8: icmp_seq=14 ttl=117 time=13.4 ms
```

Figure 23 Pinging 8.8.8.8

Individual contribution

Tasks	Bibek	Santhosh	Ashwin
Understanding the Problem statements and the basics of wi-fi concepts			
Installation of required software for assigned Project			
Configuring the Kernel for TI WLAN Drivers			
Generation of Image file for Board with Wi-Fi enabled			
Mksdboot.sh script file for partitioning SD card			
Porting of Drivers			
Bringing up WLAN on the Board			
Bringing up WLAN in host PC			
Packet Capturing using Wireshark			
Documentation			

Figure 24 Individual Contribution

Summary

- Enabling Wi-Fi in Linux and debugging the packet flow.
- Understanding the packet flow in different variations of 802.11
- Getting a better understanding and hands-on experience on packet capturing tools like Wireshark.
- Understanding the association and authentication process of the 802.11 protocol.

Challenges faced and how were they overcome

- Failure of building the kernel and SDK, since no bison and flex packages.
- Failing to boot the board initially, since the system framework file (sysfw) version was not compatible with board and not properly named.
- During initial build of the bootable SD-card, the automated shell script was blocking sometimes. For which manually the steps were followed from the partition stage to extract the files respectively to boot & rootfs partition.
- On the hardware side, the evm is a legacy board which uses SD card, while our bootable drive is a microSD one. Hence due to microSD – SD card adapter issue, board was not booting up which got resolved by changing the adapter.
- Board was not connecting to the ethernet for downloading the required packages due to protected network, for which we need to manually copy the required files from host PC to evm board.
- Resources for arm 335xx evm and for the driver integration.

REFERENCES

https://www.cablefree.net/wireless-technology/history-of-wifi-technology/
https://worldwide.espacenet.com/patent/search/family/003776560/publication/EP0599632A2?q=pn%3DEP0599632
https://www.netspotapp.com/explaining-wifi-standards.html
https://en.wikipedia.org/wiki/IEEE_802.11#History
https://purple.ai/blogs/history-wifi/
https://www.intuitibits.com/2017/08/11/understanding-scan-modes-wifiexplorerpro/
https://howiwifi.com/2020/07/13/802-11-frame-types-and-formats/
https://iwd.wiki.kernel.org/
https://www.linux.com/training-tutorials/linux-wireless-networking-short-walk/
https://wireless.wiki.kernel.org/en/developers/Documentation/mac80211
https://wireless.wiki.kernel.org/en/developers/documentation/glossary
https://wireless.wiki.kernel.org/en/users/documentation
http://www.haifux.org/lectures/206/wirelessLec.pdf
https://elixir.bootlin.com/linux/latest/source/include/net/mac80211.h
https://www.linuxbabe.com/command-line/ubuntu-server-16-04-wifi-wpa-supplicant
https://askubuntu.com/questions/660994/disable-enable-wifi-driver
https://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Tools.html
https://elixir.bootlin.com/linux/latest/source/include/uapi/linux/nl80211.h
https://www.infradead.org/~tgr/libnl/
https://stackoverflow.com/questions/21456235/how-nl80211-library-cfg80211-work
http://robbie-cao.github.io/2016/08/learning-wifi