

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



LTTTS  
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

<b>LIST OF FIGURES .....</b>	<b>3</b>
<b>SHADOW PROJECT - [TEAM].....</b>	<b>5</b>
MODULES.....	5
<i>Topic and Subtopics .....</i>	<i>5</i>
OBJECTIVES & REQUIREMENTS.....	5
<b>REQUIREMENTS.....</b>	<b>5</b>
<b>DESIGN .....</b>	<b>6</b>
<b>TEST PLAN .....</b>	<b>7</b>
<b>IMPLEMENTATION SUMMARY .....</b>	<b>8</b>
CROSS COMPILE TOOL-CHAIN .....	8
BUILDING THE KERNEL.....	8
FLASHING THE IMAGE INTO THE SD CARD .....	9
PORTING CFG AND MAC DRIVERS.....	9
PORTING USB-WI-FI DRIVER .....	10
DOWNLOAD THE CORRESPONDING FIRMWARE I.E., ATH9K_HTC .....	12
DISCONNECTING THE WI-FI DONGLE AND OBSERVE THE LOGS: .....	14
CONNECTING TO A SECURED ACCESS POINT VIA WPA_SUPPLICANT IN LINUX UBUNTU DISTRIBUTION USING WI-FI DONGLE.....	14
<i>Connect to Wi-fi network using wpa_supplicant .....</i>	<i>15</i>
<i>Testing the connection by Pinging to 8.8.8.8.....</i>	<i>17</i>
<b>INDIVIDUAL CONTRIBUTION.....</b>	<b>17</b>
<b>SUMMARY.....</b>	<b>18</b>
<b>CHALLENGES FACED AND HOW WERE THEY OVERCOME.....</b>	<b>18</b>
<b>REFERENCES .....</b>	<b>18</b>

## LIST OF FIGURES

Figure 1 Block diagram of the system .....	6
Figure 2 Building the Kernel .....	8
Figure 3 Newly generated Image.....	8
Figure 4 Partitioning the SD Card and flashing the Image .....	9
Figure 5 Enabling cfg and mac drivers.....	9
Figure 6 Atheros related drivers .....	10
Figure 7 Enabling USB network support.....	10
Figure 8 Enabling USB wireless management system.....	11
Figure 9 Booting the board.....	12
Figure 10 Wi-Fi dongle connection logs .....	12
Figure 11 Atheros ath9k_htc firmware .....	12
Figure 12 Wi-Fi dongle connection logs .....	13
Figure 13 Network interface.....	13
Figure 14 Disconnecting the Wi-Fi dongle, logs.....	14
Figure 15 Starting the Network Manager.....	14

Figure 16 Stopping of Network Manager ..... 14

Figure 17 Wireless Network Interface..... 15

Figure 18 List of Access Points ..... 15

Figure 19 Creating wpa\_supplicant.conf ..... 16

Figure 20 Connecting to Access Point ..... 16

Figure 21 Connected to Access Point..... 16

Figure 22 Assigning IP address..... 17

Figure 23 Pinging 8.8.8.8 ..... 17

Figure 24 Individual Contribution ..... 17

**LIST OF TABLES**

Table 1 High Level Requirements ..... 5

Table 2 Low Level Requirements ..... 5

Table 3 Test plan ..... 7

## 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

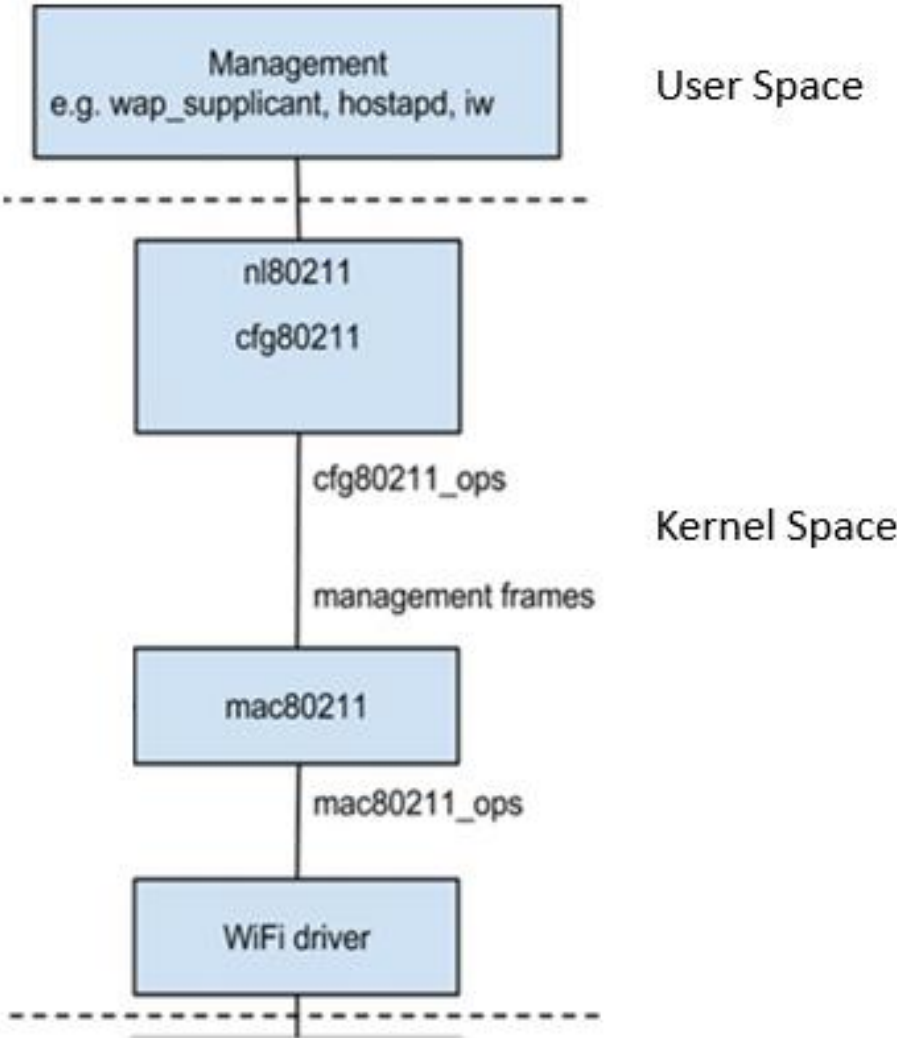


Figure 1 Block diagram of the system

## 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

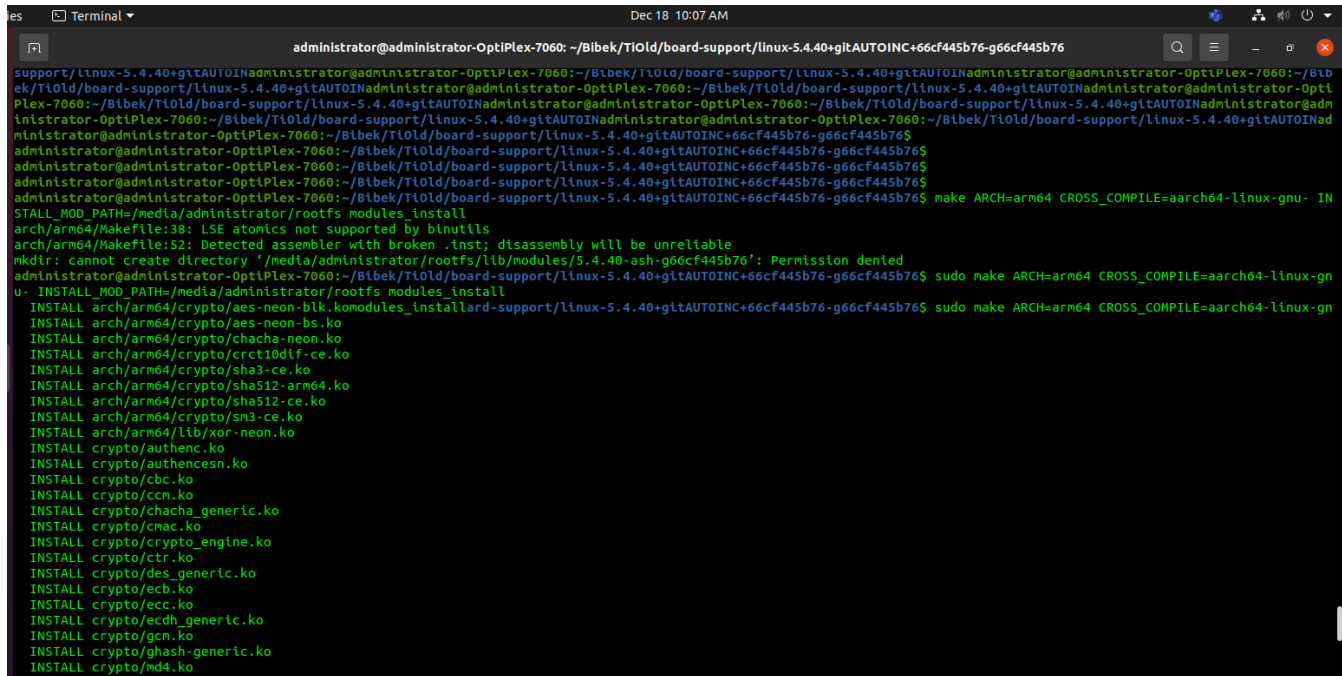


## Implementation Summary

### 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-gnueabi/bin
  - export PATH=\$GCC92PATH:\$PATH

### Building the kernel



```

administrator@administrator-OptiPlex-7060: ~/Bibek/Ti0ld/board-support/linux-5.4.40+gitAUTOINC+66cf445b76-g66cf445b76
support/linux-5.4.40+gitAUTOINC+66cf445b76-g66cf445b76$ make ARCH=arm64 CROSS_COMPILE=aarch64-linux-gnu- IN
arch/arm64/Makefile:38: LSE atomics not supported by binutils
arch/arm64/Makefile:52: Detected assembler with broken .inst; disassembly will be unreliable
mkdir: cannot create directory '/media/administrator/rootfs/lib/modules/5.4.40-ash-g66cf445b76': Permission denied
administrator@administrator-OptiPlex-7060: ~/Bibek/Ti0ld/board-support/linux-5.4.40+gitAUTOINC+66cf445b76-g66cf445b76$ sudo make ARCH=arm64 CROSS_COMPILE=aarch64-linux-gn
u- INSTALL_MOD_PATH=/media/administrator/rootfs modules_install
INSTALL arch/arm64/crypto/aes-neon-blk.komodules_installarch-support/linux-5.4.40+gitAUTOINC+66cf445b76-g66cf445b76$ sudo make ARCH=arm64 CROSS_COMPILE=aarch64-linux-gn
INSTALL arch/arm64/crypto/aes-neon-bs.ko
INSTALL arch/arm64/crypto/chacha-neon.ko
INSTALL arch/arm64/crypto/crc10dif-ce.ko
INSTALL arch/arm64/crypto/sha3-ce.ko
INSTALL arch/arm64/crypto/sha512-arm64.ko
INSTALL arch/arm64/crypto/sha512-ce.ko
INSTALL arch/arm64/crypto/sm3-ce.ko
INSTALL arch/arm64/lib/xor-neon.ko
INSTALL crypto/authenc.ko
INSTALL crypto/authencsn.ko
INSTALL crypto/cbc.ko
INSTALL crypto/ccn.ko
INSTALL crypto/chacha_generic.ko
INSTALL crypto/cmac.ko
INSTALL crypto/crypto_engine.ko
INSTALL crypto/ctr.ko
INSTALL crypto/des_generic.ko
INSTALL crypto/ecb.ko
INSTALL crypto/ecc.ko
INSTALL crypto/ecdh_generic.ko
INSTALL crypto/gcm.ko
INSTALL crypto/ghash-generic.ko
INSTALL crypto/md4.ko

```

Figure 2 Building the Kernel



```

DEPMOD 5.4.40-ash-g66cf445b76
administrator@administrator-OptiPlex-7060: ~/Bibek/Ti0ld/board-support/linux-5.4.40+gitAUTOINC+66cf445b76-g66cf445b76$ c
^C
administrator@administrator-OptiPlex-7060: ~/Bibek/Ti0ld/board-support/linux-5.4.40+gitAUTOINC+66cf445b76-g66cf445b76$
administrator@administrator-OptiPlex-7060: ~/Bibek/Ti0ld/board-support/linux-5.4.40+gitAUTOINC+66cf445b76-g66cf445b76$
administrator@administrator-OptiPlex-7060: ~/Bibek/Ti0ld/board-support/linux-5.4.40+gitAUTOINC+66cf445b76-g66cf445b76$
administrator@administrator-OptiPlex-7060: ~/Bibek/Ti0ld/board-support/linux-5.4.40+gitAUTOINC+66cf445b76-g66cf445b76$ cd arch/arm64/boot
administrator@administrator-OptiPlex-7060: ~/Bibek/Ti0ld/board-support/linux-5.4.40+gitAUTOINC+66cf445b76-g66cf445b76/arch/arm64/boot$ ls -l
total 22448
drwxr-xr-x 31 root root 4096 Dec 17 14:17 dts
-rw-r--r-- 1 root root 16654344 Dec 17 17:12 Image
-rw-r--r-- 1 root root 6885464 Dec 17 17:12 Image.gz
-rw-r--r-- 1 root root 1562 Aug 10 22:21 install.sh
-rw-r--r-- 1 root root 1200 Aug 10 22:21 Makefile
administrator@administrator-OptiPlex-7060: ~/Bibek/Ti0ld/board-support/linux-5.4.40+gitAUTOINC+66cf445b76-g66cf445b76/arch/arm64/boot$ sudo cp Image /media/administrator
administrator@administrator-OptiPlex-7060: ~/Bibek/Ti0ld/board-support/linux-5.4.40+gitAUTOINC+66cf445b76-g66cf445b76/arch/arm64/boot$

```

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
umounting 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
umounting /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]

```

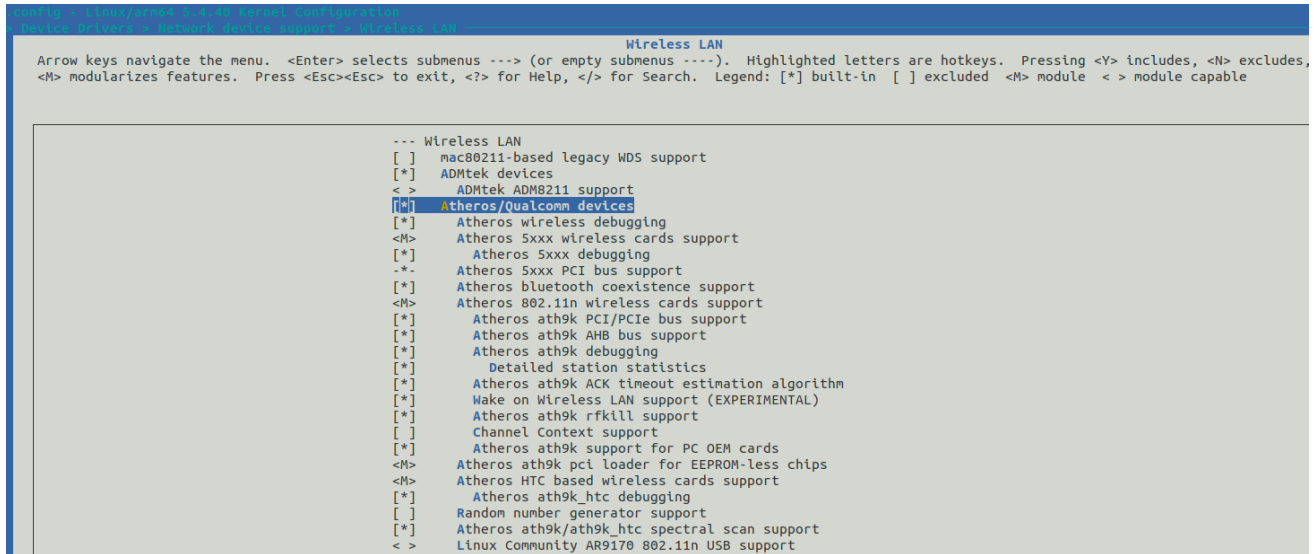
Wireless
Arrow keys navigate the menu. <Enter> selects submenus ---> (or empty submenu --->). Highlighted letters are hotkeys. Pressing <Y> includes, <N> excludes,
<M> modularizes features. Press <Esc><Esc> to exit, <?> for Help, </> for Search. Legend: [*] built-in [ ] excluded <M> module < > module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



```

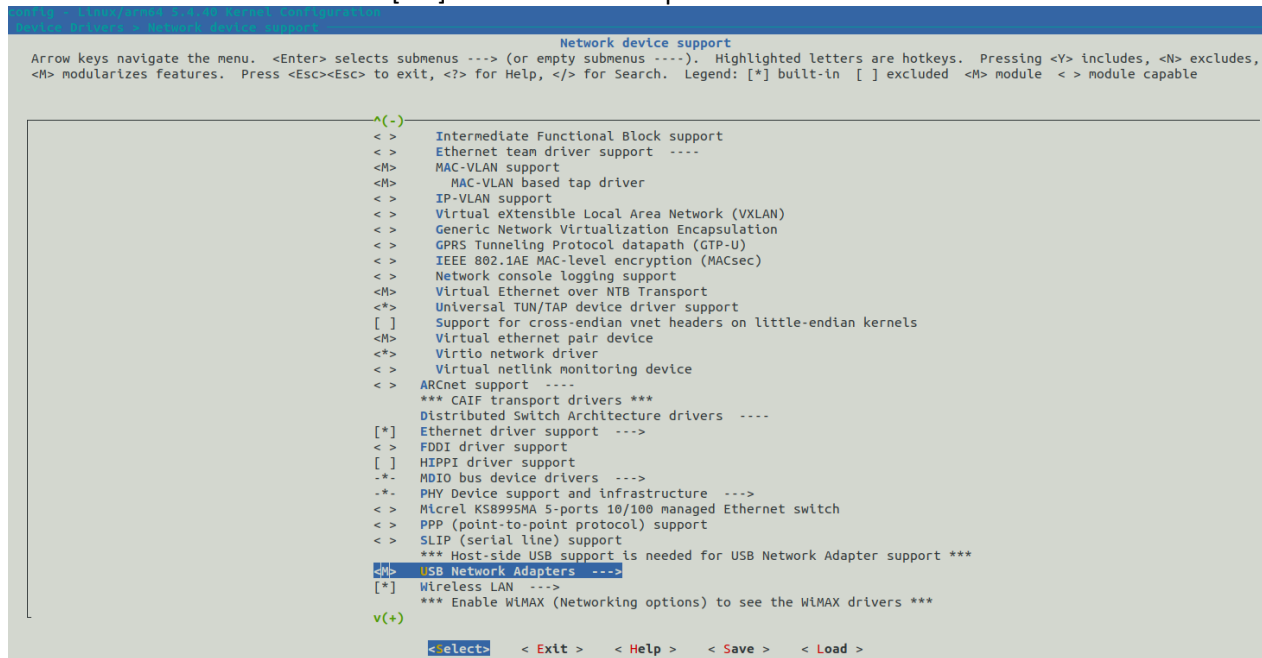
Wireless LAN
Arrow keys navigate the menu. <Enter> selects submenus --- (or empty submenu ---). Highlighted letters are hotkeys. Pressing <Y> includes, <N> excludes,
<M> modularizes features. Press <Esc><Esc> to exit, <?> for Help, </> for Search. Legend: [*] built-in [ ] excluded <M> module < > module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

--- Wireless LAN
[ ] mac80211-based legacy WDS support
[*] ADMtek devices
< > ADMtek ADM8211 support
[*] Atheros/Qualcomm devices
[*] Atheros wireless debugging
<M> Atheros 5xxx wireless cards support
[*] Atheros 5xxx debugging
[*] Atheros 5xxx PCI bus support
[*] Atheros bluetooth coexistence support
<M> Atheros 802.11n wireless cards support
[*] Atheros ath9k PCI/PCIe bus support
[*] Atheros ath9k AHB bus support
[*] Atheros ath9k debugging
[*] Detailed station statistics
[*] Atheros ath9k ACK timeout estimation algorithm
[*] Wake on Wireless LAN support (EXPERIMENTAL)
[*] Atheros ath9k rfkill support
[ ] Channel Context support
[*] Atheros ath9k support for PC OEM cards
<M> Atheros ath9k pci loader for EEPROM-less chips
<M> Atheros HTC based wireless cards support
[*] Atheros ath9k_htc debugging
[ ] Random number generator support
[*] Atheros ath9k/ath9k_htc spectral scan support
< > Linux Community AR9170 802.11n USB support
  
```

Figure 6 Atheros related drivers

## Porting USB-Wi-Fi driver

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



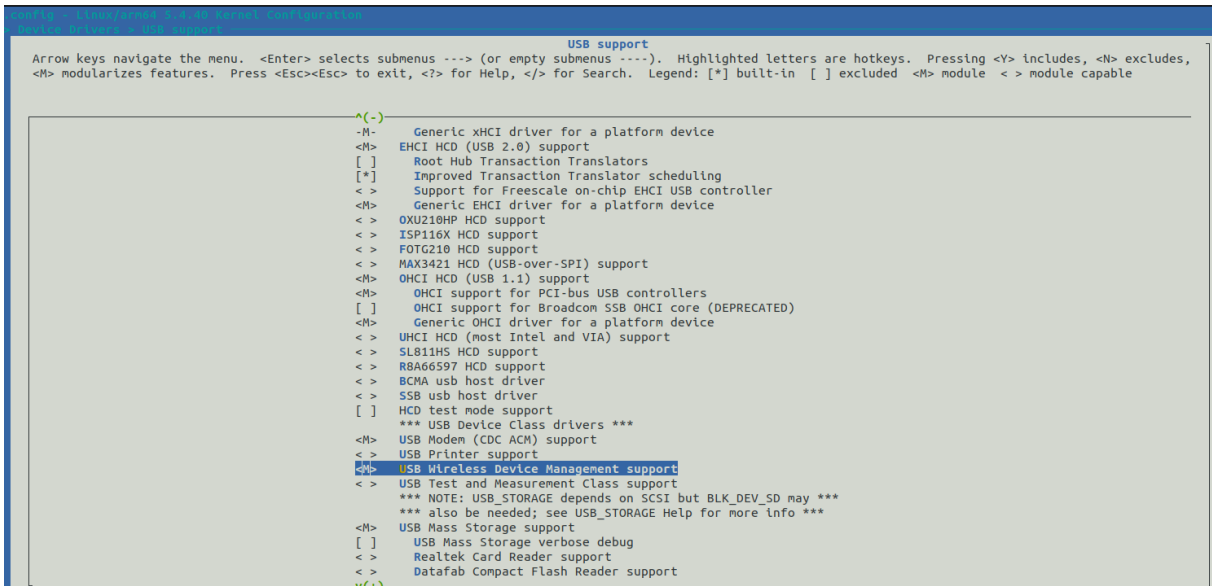
```

Network device support
Arrow keys navigate the menu. <Enter> selects submenus --- (or empty submenu ---). 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

^(-)
< > Intermediate Functional Block support
< > Ethernet team driver support ----
<M> MAC-VLAN support
<M> MAC-VLAN based tap driver
< > IP-VLAN support
< > Virtual eXtensible Local Area Network (VXLAN)
< > Generic Network Virtualization Encapsulation
< > GPRS Tunneling Protocol datapath (GTP-U)
< > IEEE 802.1AE MAC-level encryption (MACsec)
< > Network console logging support
<M> Virtual Ethernet over NTB Transport
<M> Universal TUN/TAP device driver support
[ ] Support for cross-endian vnet headers on little-endian kernels
<M> Virtual ethernet pair device
<M> Virtio network driver
< > Virtual netlink monitoring device
< > ARCnet support ----
*** CAIF transport drivers ***
Distributed Switch Architecture drivers ----
[*] Ethernet driver support ---
< > FDDI driver support
[ ] HIPPI driver support
-* MDIO bus device drivers ---
-* PHY Device support and infrastructure ---
< > Micrel KS8995MA 5-ports 10/100 managed Ethernet switch
< > PPP (point-to-point protocol) support
< > SLIP (serial line) support
*** Host-side USB support is needed for USB Network Adapter support ***
<M> USB Network Adapters ----
[*] Wireless LAN ----
*** Enable WiMAX (Networking options) to see the WiMAX drivers ***

v(+)
<select> < Exit > < Help > < Save > < Load >
  
```

Figure 7 Enabling USB network support



```

USB support
Arrow keys navigate the menu. <Enter> selects submenus --- (or empty submenu ---). 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

^(-)
-M- Generic xHCI driver for a platform device
<M> EHCI HCD (USB 2.0) support
[ ] Root Hub Transaction Translators
[*] Improved Transaction Translator scheduling
<> Support for Freescale on-chip EHCI USB controller
<M> Generic EHCI driver for a platform device
<> OXU210HP HCD support
<> ISP116X HCD support
<> FOTG210 HCD support
<> MAX3421 HCD (USB-over-SPI) support
<M> OHCI HCD (USB 1.1) support
<M> OHCI support for PCI-bus USB controllers
[ ] OHCI support for Broadcom SSB OHCI core (DEPRECATED)
<M> Generic OHCI driver for a platform device
<> UHCI HCD (most Intel and VIA) support
<> SL811HS HCD support
<> R8A66597 HCD support
<> BCMA usb host driver
<> SSB usb host driver
[ ] HCD test mode support
*** USB Device Class drivers ***
<M> USB Modem (CDC ACM) support
<> USB Printer support
<M> USB Wireless Device Management support
<M> USB Test and Measurement Class support
*** NOTE: USB_STORAGE depends on SCSI but BLK_DEV_SD may ***
*** also be needed; see USB_STORAGE Help for more info ***
<M> USB Mass Storage support
[ ] USB Mass Storage verbose debug
<> Realtek Card Reader support
<> Datafab Compact Flash Reader support
v(+)

```

**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

```

dosfstools
elfutils
gnash
gdb
gdbserver
glibc2
gstcamer1.0-libav
grip
less
libasound
libbsd
libdwarf
libelf1
libgdm-compat4
libgdm-dev
libgdm0
libgmp10
libidn2-0
libreadline-dev
libreadline8
libunistring2
nc-dev
nc
nettle
parted
python-anomaly-detection
tar
which

If you do not wish to distribute GPLv3 components please remove
the above packages prior to distribution. This can be done using
the opkg remove command. I.e.:
opkg remove <package>
where <package> is the name printed in the list above

NOTE: If the package is a dependency of another package you
will be notified of the dependent packages. You should
use the --force-remove-of-dependent-packages option to
also remove the dependent packages as well.
=====

am65xx-evm login: root
root@am65xx-evm:~#

```

Figure 9 Booting the board

Connect the Wi-Fi dongle and observe the logs

```

root@am65xx-evm:/lib/modules/5.4.40-ash-g66cf445b76/kernel/drivers/net/wireless/ath/ath9k# [ 765.287907] u
768.482521] usb 1-1: new high-speed USB device number 5 using xhci-hcd
768.654896] usb 1-1: New USB device found, idVendor=0cf3, idProduct=9271, bcdDevice= 1.08
768.663085] usb 1-1: Product: USB2.0 WLAN
768.670502] usb 1-1: Manufacturer: Atheros
768.674536] usb 1-1: SerialNumber: 12345
768.678665] usb 1-1: ath9k_htc: Firmware ath9k_htc/htc_9271-1.4.0.fw requested
768.685348] usb 1-1: Direct firmware load for ath9k_htc/htc_9271-1.4.0.fw failed with error -2
768.702113] usb 1-1: ath9k_htc: Firmware htc_9271.fw requested
768.708087] usb 1-1: Direct firmware load for htc_9271.fw failed with error -2
768.715370] usb 1-1: no suitable firmware found!
768.720024] usb 1-1: ath9k_htc: Failed to get firmware htc_9271.fw
768.726755] usb 1-1: ath9k_htc: USB layer deinitialized

```

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

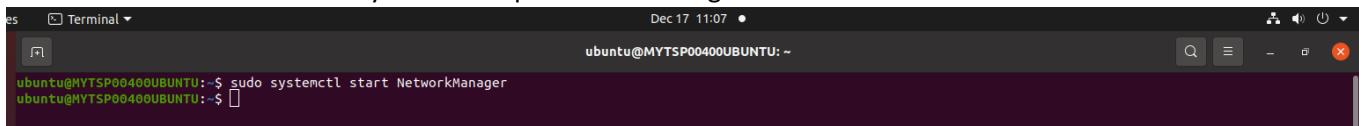
#### NOTE:

- 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

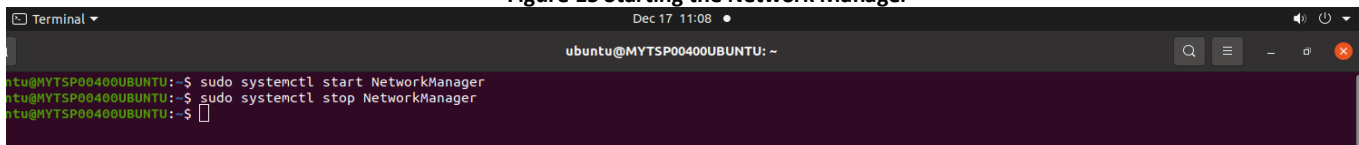


```

ubuntu@MYTSP00400UBUNTU: ~
$ sudo systemctl start NetworkManager

```

Figure 15 Starting the Network Manager



```

ubuntu@MYTSP00400UBUNTU: ~
$ sudo systemctl start NetworkManager
ubuntu@MYTSP00400UBUNTU: ~
$ sudo systemctl stop NetworkManager

```

Figure 16 Stopping 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
wlc04a0028f97e 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 wlc04a0028f97e up
- Then find your wireless network name by scanning nearby networks with the command below
  - sudo iwlist wlc04a0028f97e scan | grep ESSID

```
ubuntu@MYTSP00400UBUNTU:~$ sudo ifconfig wlc04a0028f97e up
ubuntu@MYTSP00400UBUNTU:~$ sudo iwlist wlc04a0028f97e 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=b39e004a45ebdcdb556d8e42fe024e5e33c73da44ed77b052559ab468fc89ad
}
```

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://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://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>  
<https://robbie-cao.github.io/2016/08/learning-wifi>