EVIA

Host based Wi-Fi (802.11a/b/g/n/ac) Modules

EVIA
EVIA M.2

USB DVK User Guide

Version 1.0

For additional Information, please contact info@ivativ.com

Table of Contents

Table of Figures

List of Tables

1 Overview

EVIA is a high performance multi-protocol certified wireless module supporting dual-band single stream Wi-Fi 802.11 a/b/g/n/ac. This small form-factor module provides superior throughputs and long range by integrating high performance baseband, MAC, fine-tuned RF front end.

EVIA is based on Qualcomm QCA9377 SoC and connects to host processor with multiple host interfaces including USB. It supports Wi-Fi over USB 2.0 interface.

This document is applicable for both EVIA and EVIA M.2 modules.

This document covers the following topics

- Describes the USB/USB M.2 DVK features and corresponding jumper / switch settings to test the functionality.
- Instructions to build the Linux driver in x86 and embedded platforms
- It also contains the instructions to test the basic Wi-Fi functionality

2 Requirements

2.1 Hardware

- Any Linux PC or Laptop with the USB interface connector
  (Or)
  Ixora / Apalis iMX6 carrier board from Toradex
- EVIA USB/USB M.2 EVK board

2.2 Software

- Kernel Version 4.4.15, 4.9.11 and 4.11Build for USB QCA proprietary driver
- For the detailed steps to build the driver for x86 host or the embedded target, please refer to the section "Linux driver build preparation"

3 Hardware Setup

3.1 EVIA USB EVK Image

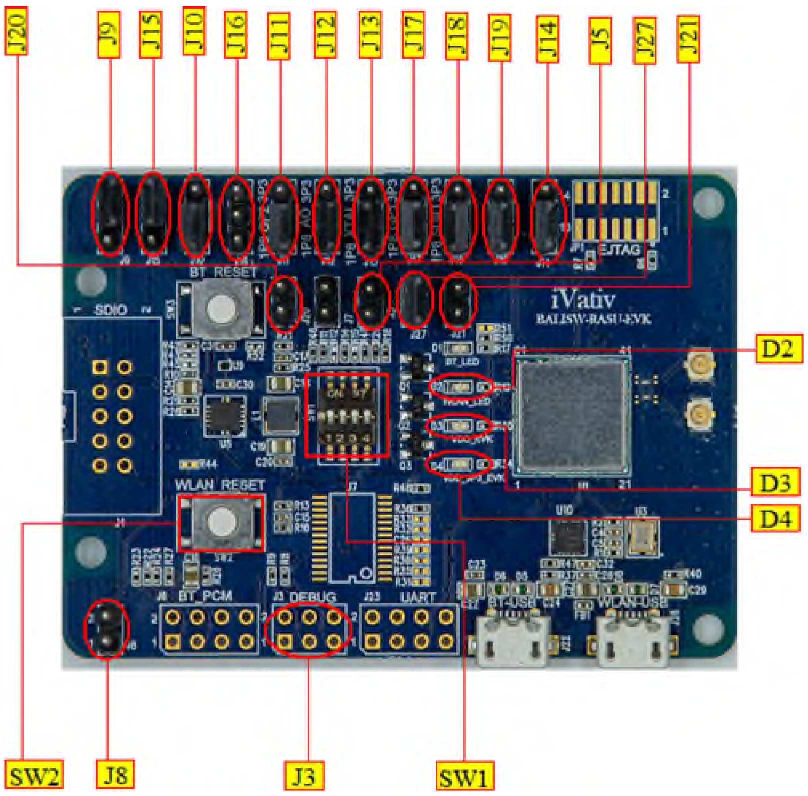


Figure 1: EVIA USB EVK

3.2 EVIA Jumper setting description

Jumper	Pin Number	Description	Usage
J3		Reserved	
J8		Reserved	
J9	J9.1	VDDIO_SDCON	Connect J9.1 and J9.2 for SDIO supply Connect J9.2 and J9.3 for USB supply
	J9.2	SD_VIN	
	J9.3	USB1.1_PWR	
J10	J10.1	USB_VIN	Connect J10.1 and J10.2 for Internal supply Connect J10.2 and J10.3 for External supply
	J10.2	PWR_IN	
	J10.3	EXT_PWR	

J15	J15.1	SD_VIN	Connect J15.1 and J15.2 for SD power supply Connect J15.2 and J15.3 for USB power supply
	J15.2	USB_VIN	
	J15.3	USB2_PWR	
J16	J16.1 & 16.2	EXT_PWR & Ground	Connect J16.1 and J16.2 for External power supply
J11	J11.1	VDD_3P3_MOD	Connect J11.1 and JP11.2 for 3.3V Operation Connect JP11.2 and JP11.3 for 1.8V Operation
	J11.2	VDDIO_GPIO2	
	J11.3	VDD_1P8_EVK	
J12	J12.1	VDD_3P3_MOD	Connect J12.1 and J12.2 for 3.3V Operation Connect J12.2 and J12.3 for 1.8V Operation
	J12.2	VDDIO_AO	
	J12.3	VDD_1P8_EVK	
J17	J17.1	VDD_3P3_MOD	Connect J17.1 and J17.2 for 3.3V Operation Connect J17.2 and J17.3 for 1.8V Operation
	J17.2	VDDIO_GPIO1	
	J17.3	VDD_1P8_EVK	
J13	J13.1	VDD_3P3_MOD	Connect J13.1 and J13.2 for 3.3V Operation Connect J13.2 and J13.3 for 1.8V Operation
	J13.2	VDDIO_XTAL	
	J13.3	VDD_1P8_EVK	
J18	J18.1	VDD_3P3_MOD	Connect J18.1 and J18.2 for 3.3V Operation Connect J18.2 and J18.3 for 1.8V Operation
	J18.2	VDDIO_SDIO	
	J18.3	VDD_1P8_EVK	
J19	J19.1	VDD_3P3_MOD	Connect J19.1 and J19.2 for 3.3V Operation
	J19.2	VDDIO_USB2	
	J19.3	Ground	
J14	J14.1	VDD_3P3_EVK	Connect J14.1 and J14.2 for 3.3V Operation Connect J14.2 and J14.3 for 1.8V Operation
	J14.2	VDD_EVK	
	J14.3	VDD_1P8_EVK	
J5	J5.1 & J5.2	WLAN_RESET	Connect these 2 pins for continuous WLAN Reset

J20	J20.1 & J20.2	Reserved	Reserved
J21	J21.1 & J21.2	Reserved	Reserved
J27	J27.1 & J27.2	For Current measurement	Connect the equipment between these 2 pins (Please refer Issues/Errata section)
SW1	Boot Strap SW	DIP-8 Switch	Default for normal operation
SW2		WLAN reset push button	Press the button to enable active low WLAN_RESET
D2		WLAN LED	WLAN Indication LED for WLAN Activity (Please refer Issues/Errata section)
D3		Power On LED 1.8V	
D4		Power On LED 3.3V	

Table 1: EVIA USB Jumper setting description

3.3 EVIA USB M.2 EVK Image

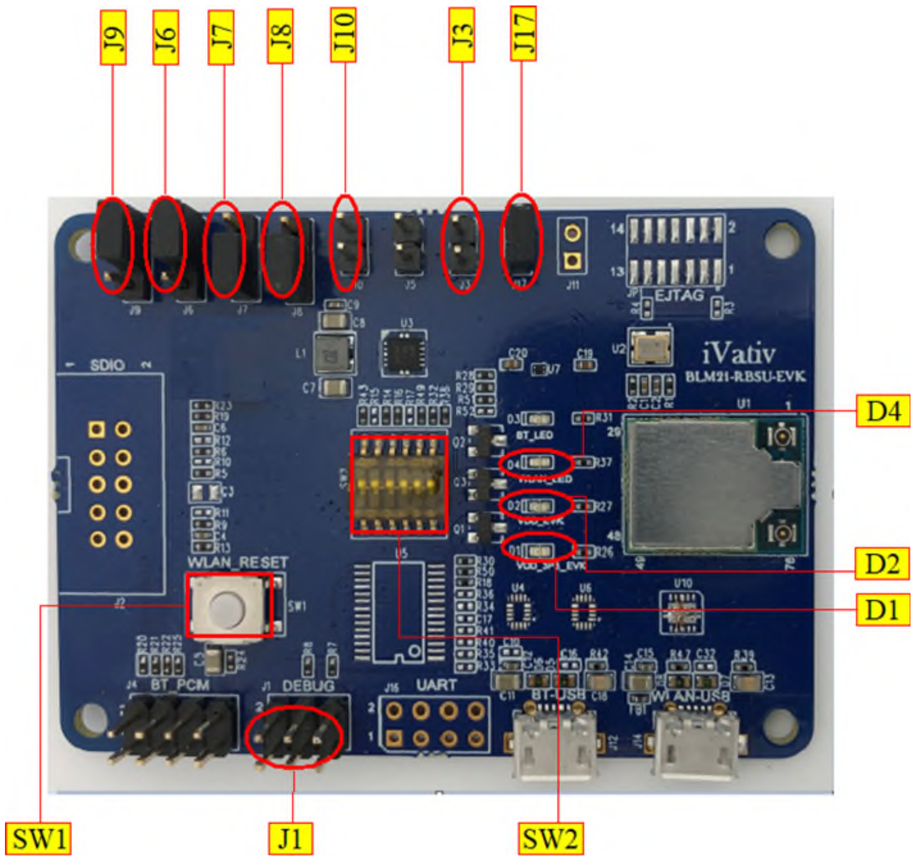


Figure 2: EVIA USBM.2 EVK

3.4 EVIA M.2 Jumper setting description

Jumper	Pin Number	Description	Usage
J1		Reserved	
J6	J6.1	VDDIO_SDCON	Connect J6.1 and J6.2 for SDIO supply Connect J6.2 and J6.3 for USB supply
	J6.2	SD_VIN	
	J6.3	USB1.1_PWR	
J7	J7.1	USB_VIN	Connect J7.1 and J7.2 for Internal supply Connect J7.2 and J7.3 for External supply
	J7.2	PWR_IN	
	J7.3	EXT_PWR	
J9	J9.1	SD_VIN	Connect J9.1 and J9.2 for SD power supply Connect J9.2 and J9.3 for USB power supply
	J9.2	USB_VIN	
	J9.3	USB2_PWR	
J8	J8.1	VDD_3P3_EVK	Connect J14.1 and J14.2 for 3.3V Operation Connect J14.2 and J14.3 for 1.8V Operation
	J8.2	VDD_EVK	
	J8.3	VDD_1P8_EVK	
J17	J17.1 & J17.2	Current measurement header	Connect equipment between these pins to measure current (Please refer Issues/Errata section)
J10	J10.1 & J10.2	EXT_PWR & Ground	Connect J10.1 & J10.2 for External power supply
J11		Reserved	
J3	J3.1 & J3.2	WLAN_RESET	Connect these 2 pins for continuous WLAN Reset
SW1		WLAN Reset Push Button	Press the button to enable active low WLAN_RESET
SW2		Boot Strap Switch	Default for normal operation
D1, D2		3.3V and 1.8V power On LEDs	Power ON Indication LED’s
D4		WLAN LED	WLAN Indication LED for WLAN Activity. Please refer Issues/Errata section

Table 2: EVIA USBM.2 Jumper setting description

Once the jumper settings are done according to the above table, please insert the USB EVK to the compatible USB slot on PC / Laptop / embedded Linux target board

4 Software

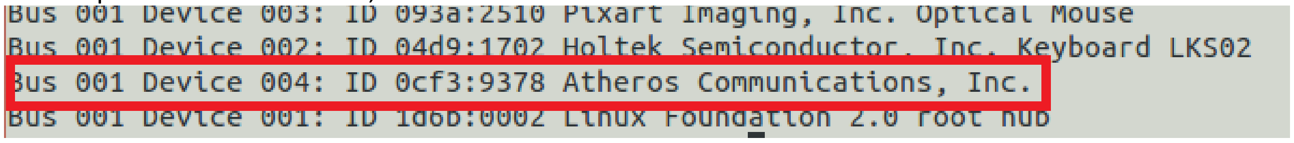
4.1 Card detection

If the driver build is ready for USB, you may start testing the features. Otherwise, please refer to the section "Linux driver build preparation" to prepare the required Linux driver build for proprietary USB driver.

- Connect the WLAN-USB to the PC to enable Wi-Fi
- Check the card detection status as below

    $ lsusb (to list out all the connected USB devices)

The output list is shown below, look for **Atheros communications**



```
Bus 001 Device 003: ID 093a:2510 Pixart Imaging, Inc. Optical Mouse
Bus 001 Device 002: ID 04d9:1702 Holtek Semiconductor, Inc. Keyboard LKS02
Bus 001 Device 004: ID 0cf3:9378 Atheros Communications, Inc.
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

Figure 3: EVIA USB Interface Detection

4.2 Test WLAN

This section describes the following procedure:

- Load the drivers and bring up the board
- Connecting to AP in open mode
- Connecting to AP in secure mode
- Throughputs testing

4.2.1 Loading the driver for x86 platforms

1. Turn off your internal Wi-Fi. You may use "Airplane" mode button to do this
2. Before loading drivers we need to remove the existing cfg80211 kernel module using the following command

    $ sudo rmmod cfg80211

3. There will be some dependent kernel modules for cfg80211.ko.You need to remove them as well. Some of the most commonly seen dependent modules are listed below

    $ sudo rmmod iwldvm

    $ sudo rmmod mac80211

    $ sudo rmmod iwlWi-Fi

    $ sudo rmmod cfg80211

4. Then insert the required kernel modules

    $ sudo insmod /lib/modules/<kernel version>/kernel/net/wireless/cfg80211.ko

Example: *$ sudo insmod /lib/modules/4.9.11+/kernel/net/wireless/cfg80211.ko*

$ sudo insmod <path to>/driver_files/kernel-module-wlan_usb/usb_wlan.ko

4.2.2 Loading the driver for embedded boards

Generally, the USB driver loads automatically upon detecting the USB interface on the board. Otherwise, please follow step #2 to step #4 of section 4.2.1 to load the required kernel modules. Please note that the path to the usb_wlan.ko / wlan.ko will be different for embedded boards. Usually you can find the driver module in the path "/lib/modules/extra/wlan.ko".

4.2.3 List all available Network Interfaces

- Check if the interface is available. To see the available network interfaces, give

  $ ifconfig -a

Sample output:

```
eth0    Link encap:EthernetHWaddr 00:04:9F:02:B4:10
        inet addr:192.168.0.20 Bcast:192.168.0.255 Mask:255.255.255.0
        inet6 addr: 2605:6000:ee8a:9000:204:9fff:fe02:b410/64 Scope:Global
        inet6 addr: fe80::204:9fff:fe02:b410/64 Scope:Link
        UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
        RX packets:1247 errors:0 dropped:67 overruns:0 frame:0
        TX packets:118 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:87449 (85.3 KiB) TX bytes:13933 (13.6 KiB)

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:10 errors:0 dropped:0 overruns:0 frame:0
        TX packets:10 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:0
```

```
          RX bytes:700 (700.0 B) TX bytes:700 (700.0 B)


wlx50DE19896801       Link encap:EthernetHWaddr 84:25:3F:0A:9B:39
                      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)
```

The "**wlx50DE19896801**" network interface listed above shows the successful detection of the USB interface and the driver. Please follow below sections of the document to bring up the interface and use it for functional testing.

4.2.4 WLAN interface bring up

- Enable WLAN interface by issuing the following command

  $ ifconfig <wlan_interafce> up

  Example: *$ ifconfig* wlx50DE19896801 *up*

- Host's *rfkill* software may block the wireless devices by default, need to unblock them by running the following command

  $ rfkill unblock all

- Check the "wlan_interface" is up

  $ ifconfig

  The sample output shows the interface is up, please observe the highlighted text

```
wlx50DE19896801       Link encap: Ethernet HWaddr 84:25:3F:0A:9B:39
                      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)
```

4.2.5 Scanning for WLAN Networks

- To scan for available networks on all channels, issue the following command:

  $ sudo iw dev <wlan_interface> *scan*

4.2.6 Connect to AP in open mode

- Configure the AP in open mode, make sure the Access point you want to connect is within the range. You may check the signal strength in the scan results to know if the AP is reachable
- Need to kill already running supplicant application before connecting to a new network, follow below command to kill the previous wpa_supplicant

  $ sudo killall wpa_supplicant (keep issuing this command until ***no process found*** *as an output)*

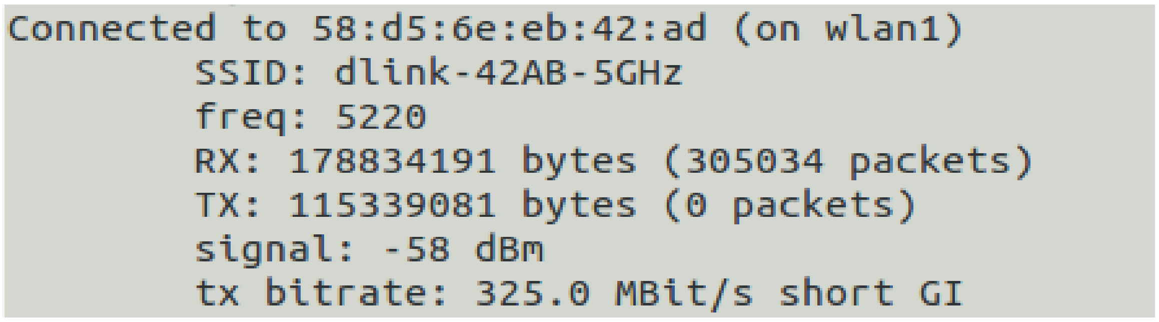
- Connect to the AP by mentioning its SSID in the below command

  $ sudo iw dev <wlan_interface> connect <SSID>

  Example: $ sudo iw dev *wlx50DE19896801* connect dlink-42AB-5Ghz

- It may take some time (few seconds) to find and associate to the given AP. Check the connection status by using below command.

  $ iw dev <wlan_interface> link



```
Connected to 58:d5:6e:eb:42:ad (on wlan1)
        SSID: dlink-42AB-5GHz
        freq: 5220
        RX: 178834191 bytes (305034 packets)
        TX: 115339081 bytes (0 packets)
        signal: -58 dBm
        tx bitrate: 325.0 MBit/s short GI
```

Figure 4: Station connected to AP

NOTE: If it is not connected then follow above three steps once again.

4.2.7 Configure DHCP

- To obtain an IP address dynamically, issue the following commands:

```
$ sudo dhclient wlan0 -r (kill all the previous process dhcp clients if any)
$ sudo dhclient wlan0 -v (enable verbose log messages)
```

(Or)

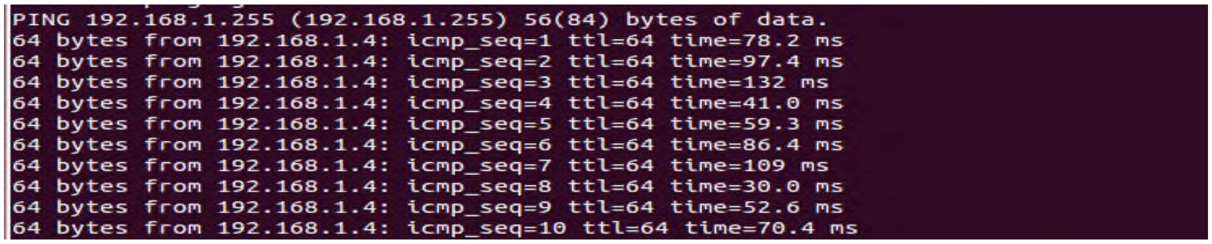
```
$ udhcpc -i wlan0
```

4.2.8 Ping with AP

- To check the ping, issue the below command

  $ ping < IP_addr_of_AP >

- Check below image for sample ping result



```
PING 192.168.1.255 (192.168.1.255) 56(84) bytes of data.
64 bytes from 192.168.1.4: icmp_seq=1 ttl=64 time=78.2 ms
64 bytes from 192.168.1.4: icmp_seq=2 ttl=64 time=97.4 ms
64 bytes from 192.168.1.4: icmp_seq=3 ttl=64 time=132 ms
64 bytes from 192.168.1.4: icmp_seq=4 ttl=64 time=41.0 ms
64 bytes from 192.168.1.4: icmp_seq=5 ttl=64 time=59.3 ms
64 bytes from 192.168.1.4: icmp_seq=6 ttl=64 time=86.4 ms
64 bytes from 192.168.1.4: icmp_seq=7 ttl=64 time=109 ms
64 bytes from 192.168.1.4: icmp_seq=8 ttl=64 time=30.0 ms
64 bytes from 192.168.1.4: icmp_seq=9 ttl=64 time=52.6 ms
64 bytes from 192.168.1.4: icmp_seq=10 ttl=64 time=70.4 ms
```

Figure 5: Ping Results

4.2.9 Disconnect from AP

- To disconnect from AP, use the below command

  $ iw dev <wlan_interface> disconnect

4.2.10 Connect to AP with security: WPA/WPA2-PSK

Configure the AP in secure mode.

Need to edit the wpa supplicant configuration file to configure the SSID, PSK and encryption type.

To configure the WPA or WPA2 security, follow the below steps:

1. Change the current directory to the one that contains the ***wpa_supplicant.conf*** file by issuing the following command

   $ cd /etc

2. If no wpa_supplicant configuration file exists you need to create one. The created file should be in **/etc** path only by issuing the following command

   $ sudo vi wpa_supplicant.conf *(To Create and open* the ***wpa_supplicant.conf*** file*)*

3. Edit and save the configuration file ***wpa_supplicant.conf*** *file* with the following content,

For WPA-PSK

```
ctrl_interface=/var/run/wpa_supplicant
update_config=1
network={
ssid="replace with your network name"
proto=WPA #for WPA-PSK
key_mgmt=WPA-PSK
auth_alg=OPEN
pairwise=CCMP
group=CCMP
psk="replace with your pre-shared key"
}
```

For WPA2-PSK

```
ctrl_interface=/var/run/wpa_supplicant
update_config=1
network={
ssid="replace with your network name"
proto=RSN #for WPA2-PSK
key_mgmt=WPA-PSK
auth_alg=OPEN
pairwise=CCMP
group=CCMP
psk="replace with your pre-shared key"
}
```

4. issue the following command from console to connect with AP.

   ```
   $ sudo wpa_supplicant –D nl80211 -i wlanX–c wpa_supplicant.conf -B
   ```

5. The above command may take some time (usually few seconds) to discover the configured network and connect. For a successful connection, the access point has to be within the range of the STA. To confirm the connection with the configured network, issue the following command:

   ```
   $ iw dev wlan0 link
   ```

6. If the response is not connected, check the contents in the WPA configuration file, edit accordingly, save it.
7. Before proceeding to run the supplicant again, we need to kill already running supplicant application by issuing the following command

   ```
   $ sudo killall wpa_supplicant
   ```

   NOTE: Keep issuing this command until you get output as **no process found**.
8. Repeat step 4 and 5 to retry connecting to the Access Point
9. Once the connection is successful, get the IP address by following below procedure

```
$ sudo dhclient wlan0 -r          (kill all the previous process dhcp clients if any)

$ sudo dhclient wlan0 -v          (enable verbose log messages)
```

(Or)

```
$ udhcpc -i wlan0
```

4.2.11 Throughput test

- To perform throughputs test we need the following setup
  - 1 x Linux laptop acting as server with iperf installed on it
  - 1 x Linux laptop acting as client with iperf installed on it
  - 1 x Access point
  - 1 x Ethernet Cable
- Connect the access point to the laptop acting as server with the Ethernet cable
- Open command terminal and issue the following command to run the server

  Server side: *iperf3 -s -i 1*

- Connect the BALI EVK to the laptop acting as client and connect to the access point by following the procedure mentioned in the link Connect to WPA-PSK Mode AP
  Note: Make sure that for throughput test both server and client are connected to the same Access point
- After connecting to the access point, perform throughput tests by issuing the following commands.

  Client side: *iperf3 -c <server_IP> -I 1 -t 30* *(For TX)*

  iperf3 -c <server_IP> -I 1 -t 30 –R *(For Rx)*

5 Linux driver build preparation

5.1 X86 platform

5.1.1 Prerequisites

NOTE: Make sure the laptop/PC is connected to the uninterrupted power supply.

1. Before updating to a new kernel, check the current OS version

   $ uname –a

2. Make sure you have either Ubuntu 14.04 with 32 bit (Or) Ubuntu 16.04 with 64bit OS installed on laptop. If not, please install either one of them and boot into that
3. Please update the system using the following command

   $ sudo apt-get update

4. Install following software tools

   $ sudo apt-get install vim

   $ sudo apt-get install git

   $ sudo apt-get install ncurses-dev

   $ sudo apt-get install libssl-dev

   $ sudo apt-get install libnl-3-dev

   $ sudo apt-get install libnl-genl-3-dev

   $ sudo apt-get install libelf-dev

   $ sudo apt-get install bison

   $ sudo apt-get install flex

   $ sudo apt-get install automake libtool dpkg-dev libasound2-dev

NOTE: This process will take significant time (few minutes)

5.1.2 Proprietary driver build procedure

5.1.2.1 Release package contents

Linux_proprietary_driver_release.tgz comprises the below files and folders

1. BALI-EVIA_Linux_Driver_Userguide.pdf: This document
2. drivers folder: Contains driver source code
   a. wlan_usb folder

i. kernel-module-wlan_usb.tgz: USB driver source code
ii. usb_wlan_firmware/
iii. usb_ini_files/

3. 0001-kernel-4.9.11-patch-for-QCACLD-9377-driver.patch: Patch file for 4.9.11
4. kernel_configurations folder
    a. build_32/ : Contains defconfig file for 32bit OS
    b. build_64/ : Contains defconfig file for 64bit OS

5.1.2.2 Set up Linux build environment with kernel 4.9.11

5.1.2.2.1 Get Linux Source code

NOTE: If you are already having the v4.9.11, then you can ignore this section, but it is better to remove your old kernel and try with below procedure.

- Download Linux kernel source - version 4.9.11 as below

  NOTE: This step will take some time (~30 minutes) depending on the internet speed

  ```
  $ git clone git://git.kernel.org/pub/scm/linux/kernel/git/stable/linux-stable.git

  $ cd linux-stable

  $ git checkout v4.9.11

  $ git log                /* The First commit should be same as below*/

      Commit:   eee1550b3e89217321b63efba64f03b2546180d6

      Author:   Greg Kroah-Hartman <gregkh@linuxfoundation.org>

      Date:    Mon Jul 11 09:31:24 2016 -0700

      HEAD is now at eee1550... Linux 4.9.11
  ```

- Press ‘q’ to exit from “git log”

5.1.2.2.2 Apply Patches

- Copy the patch file from the release directory to the linux-stable directory

  ```
  $ cp <Path to>Linux_proprietary_driver_release/ 0001-kernel-4.9.11-patch-for-QCACLD-9377-driver.patch linux-stable/
  ```

- Apply the patch

  ```
  $ patch -p1 < 0001-kernel-4.9.11-patch-for-QCACLD-9377-driver.patch
  ```

Now the patches are applied and the kernel source is ready to build

5.1.2.3 Build the kernel

Before building the kernel, need to select the appropriate defines in the kernel configuration

1. Configure the kernel by using below command

$ make menuconfig

2. Select below options in the menu

   Save -> Exit

3. Copy the appropriate kernel configuration file to the linux-stable directory from the provided package

   For 32bit OS

   $ sudo cp <Path to >Linux_proprietary_driver_release/kernel_configurations/build_32/defconfig .config

   For 64bit OS

   $ sudo cp<Path to >Linux_proprietary_driver_release/kernel_configurations/build_64/defconfig .config

4. Configure kernel, load and save the configuration. Follow the below instruction. Run this command.

   $ make menuconfig

5. Select the below options in the menu

   Load ->Save -> Exit

NOTE: If any changes are done in .config file manually, repeat the step 4 to apply the changes.

Then build the kernel by issuing the following commands.

NOTE: This may take significant amount of time (1hr or more) depending on the system processing power

$ make

Use "make –j8" to speed up this process

$ make modules

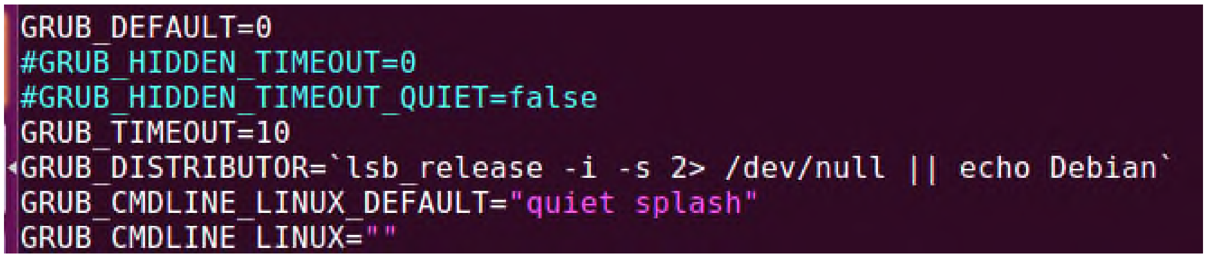
$ sudo make modules_install (requires superuser privileges)

$ sudo make headers_install (requires superuser privileges)

$ sudo make install (requires superuser privileges)

- Once the Kernel is built successfully, follow the instructions mentioned below to setup the boot scripts
- Update the grub for selecting the kernel version in grub menu on startup

- Do the changes in grub file (which is available at /etc/default/grub) as below to increase the GRUB_TIMEOUT. It's value 10, means it waits for 10 seconds for the user selection before booting to the default kernel



```
GRUB_DEFAULT=0
#GRUB_HIDDEN_TIMEOUT=0
#GRUB_HIDDEN_TIMEOUT_QUIET=false
GRUB_TIMEOUT=10
GRUB_DISTRIBUTOR=`lsb_release -i -s 2> /dev/null || echo Debian`
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash"
GRUB_CMDLINE_LINUX=""
```

Figure 6: GRUB File

Now, save the file and update the grub by using below command

$ sudo update-grub (requires super user privileges)

$ sudo reboot (requires super user privileges)

- Select 4.9.11+ kernel in Grub menu upon system boot up

The kernel build process is successful now and ready to build either SDIO / USB driver

5.1.3 Build driver

Extract the release package - Linux_proprietary_driver_release.tgz

$ tar -xvzf Linux_proprietary_driver_release.tgz

5.1.3.1 USB Driver Build

1. Extract the kernel-module-wlan_usb.tgz file

   $ cd drivers/wlan_usb

   $ tar -xvzf kernel-module-wlan_usb.tgz

   $ cd kernel-module-wlan_usb/

2. Compile the driver source. Issue the following command. This may take a few minutes (~10 minutes)

   $ make

3. After successful compilation, usb_wlan.ko is generated. This is the kernel driver module
4. Please be in the release package directory (Linux_proprietary_driver_release/)
5. Copy the firmware from release package to /lib/firmware/ directory

   $ sudo cp usb_wlan_firmware/ /lib/firmware/*

6. Need to create the wlan/usb_wlan directory in the root file system. This is required for ini file

$ sudo mkdir -p /lib/firmware/wlan/usb_wlan

7. Copy the ini file from release package to /lib/firmware/wlan/usb_wlan/directory

   $ sudo cp usb_ini_files/ /lib/firmware/wlan/usb_wlan/*

8. Now, the USB driver is built successfully.

5.2 Toradex Apalis board

5.2.1 Prerequisites

- Toradex Release package
  - IV_Toradex_Relv1_USB.tar.gz
- It is recommended to use a Linux PC / Laptop with >=8GB RAM and high processing power for yocto builds.
- Either Ubuntu 14.04 32-bit OS or Ubuntu 16.04 64-bit OS.
- Install other tools/libraries required for build

  $ sudo apt-get install gawk wget git-core diffstat unzip texinfo gcc-multilib build-essential chrpath socat cpio python python3 python3-pip python3-pexpect xz-utils debianutils iputils-ping

  $ sudo apt-get install chrpath curl dosfstools gawk g++ gcc libcrypto++9 libcrypto++-dev liblzo2-dev lzop libsdl1.2-dev libstdc++-4.8-dev libusb-1.0-0 libusb-1.0-0-dev texinfo uuid-dev zlib1g (for 32 bit)

NOTE: Below procedure is tested on Ubuntu 14.04 with 32-bit OS

NOTE: Make sure the laptop/PC is connected to the uninterrupted power supply as this process will take significant amount of time (few hours)

5.2.2 Build procedure

NOTE: Run below commands as a normal user ($) but not as root user (#)

- Untar the **IV_Toradex_Relv1_USB.tar.gz** file by issuing below command

  $ tar -xvzf IV_Toradex_Relv1_USB.tar.gz

- Follow below procedure to complete the build

  $ cd IV_Toradex_Relv1_USB.tar.gz/

  $ cp kernel-module-wlan_usb.tgz ~/

  $ cp oe-core ~/ -r

  $ cd ~/oe-core

  $. export

Note: After this command, the directory changes from ~/oe-core to ~/oe-core/build automatically.

$ cp <path to>/IV_Toradex_Relv1_USB /conf/local.conf ~/oe-core/build/conf/local.conf

$ cp <path_to>/IV_Toradex_Relv1_USB/conf/bblayers.conf ~/oe-core/build/conf/bblayers.conf

$ bitbake -k angstrom-lxde-image

(Or)

- Extract the **IV_Toradex_Relv1_USB.tar.gz** file by using below command

  $ tar -xvzf IV_Toradex_Relv1_USB.tar.gz

- Go to the extracted folder by using below command

  $ cd IV_Toradex_Relv1_USB

- Run the below script to initialize the setup

  $ sh yocto_build.sh

- Create the build environment using below command

  $. export

  Note: After this command, the directory changes from ~/oe-core to ~/oe-core/build automatically.

- To build the Linux Image use below command

  $ bitbake -k angstrom-lxde-image

NOTE: Build takes approximately 4-5 hours to complete.

NOTE: For complete yocto build procedure for Toradex, please refer Toradex Embedded platform

5.2.2.1 Verify Output

- Find out the output image (.tar file) after the build is completed successfully, in below path
  - ~/oe-core/build/deploy/images/${MACHINE}/
  - For example Apalis-iMX6_LXDE-Image-Tezi_2.8b6-20190624.tar

5.2.2.2 Rebuild Image

- If you need to remove the previous image for rebuilding the new Image. Please follow below steps
- Create build environment

  $ cd ~/oe-core

  $. export

- Remove the previous image

  $ bitbake –c cleanall angstrom-lxde-image

  $ bitbake –c cleansstate virtual/kernel

- *Build the image*

  $ bitbake –k angstrom-lxde-image

5.2.3 Prepare the bootable SD card with the image

5.2.3.1 Requirements

5.2.3.1.1 Hardware requirements

- Toradex Apalis board
- Power adaptor - 12V 30W power supply
- USB Type-B to Type-A cable
- 8GB USB Stick, it is needed for eMMC boot
- 8GB SD card, it is needed for SD card boot
- Monitor
- VGA cable

5.2.3.1.2 Software requirements

- Linux image for Toradex Apalis, which is obtained in Verify Output section
- A Laptop / Desktop with Putty software

5.2.3.2 Procedure

Toradex has two options for boot up

- eMMC
- SD card

5.2.3.3 Flashing Linux image to eMMC using Toradex Easy Installer

1. Download the Toradex Easy installer in the host PC / Laptop
   a. Get the software from the below link
      https://docs.toradex.com/104375-apalis-imx6-toradexeasyinstaller.zip
   b. Unzip the zip file
2. Extract the Linux image (.tar file)
3. Format the USB memory stick
4. Move the extracted folder contents (not folder) into USB memory stick (i.e., go to that folder and then move the all files to memory stick)
5. Make sure the memory stick contain only the extracted folder contents only
6. Now you can safely eject the USB memory stick
7. Connect the 12V 30W power supply and plug the USB Type-B to Type-A cable to **X50 USB** port, plug the display and keyboard

8. Enter into recovery mode. (**NOTE:** This process will erase the previous image on the eMMC)
    a. Make sure the board is turned off
    b. Shorten the pads on the picture (e.g. with tweezers) and power on the board using on/off switch while the pads being shorted so that it will enter into the recovery mode

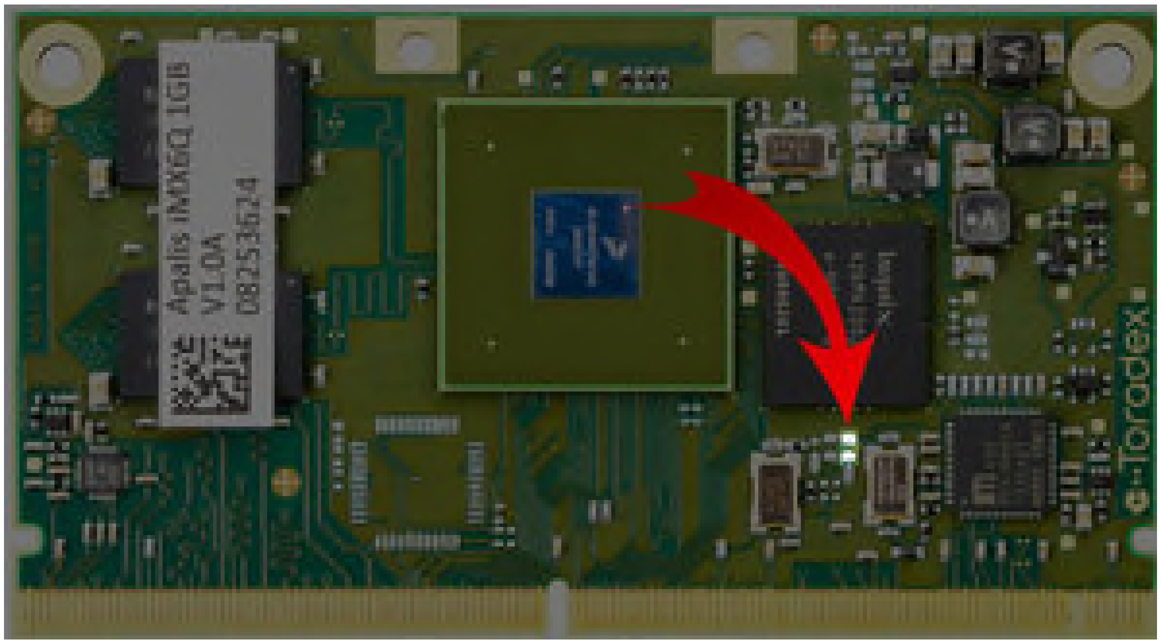


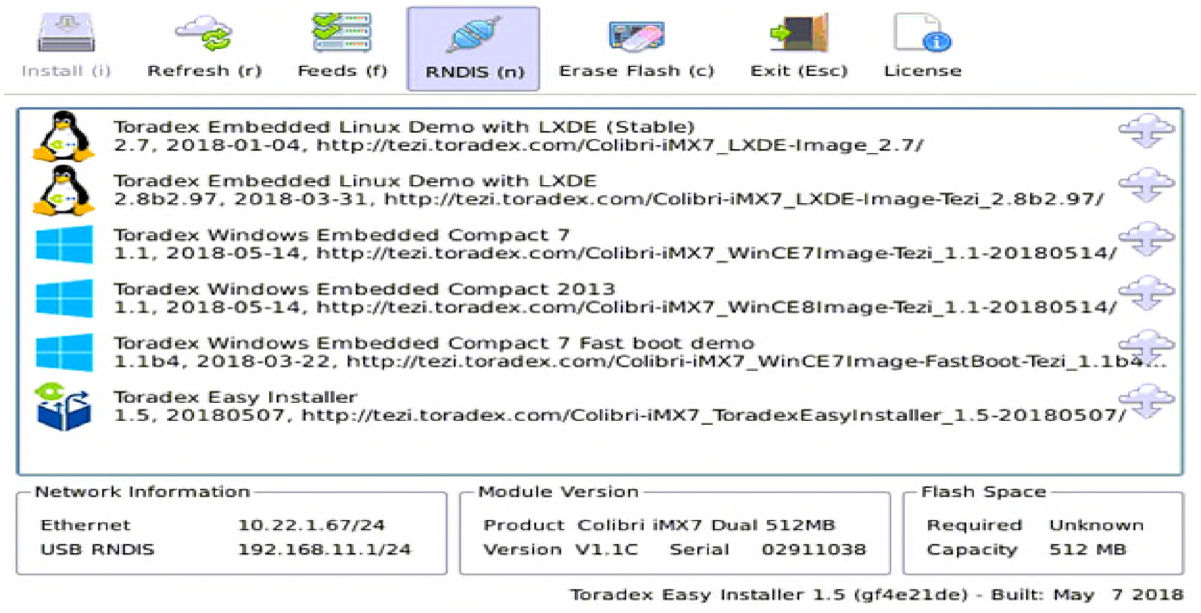
Figure 7: Recovery mode enables pads

9. Use one of the following scripts on the host machine to download and run the Toradex Easy Installer through USB:
    i. Windows: **recovery-windows.bat**
    ii. Linux: **recovery-linux.sh**

Note: This does not write the Toradex Easy Installer to flash, it only loads it into the module's memory. You will have to redo these steps if you power off your module

10. You should see the Toradex Easy Installer displayed on the monitor attached to the VGA port of the Apalis Evaluation Board
11. Now, insert the USB memory stick which is having the required Linux image

12. Once you plug USB memory stick into the target the image should appear in the list of available images with an icon indicating that it is an image found on a local storage device. Please check the screenshot below



Toradex Embedded Linux Demo with LXDE (Stable)
2.7, 2018-01-04, http://tezi.toradex.com/Colibri-iMX7_LXDE-Image_2.7/
Toradex Embedded Linux Demo with LXDE
2.8b2.97, 2018-03-31, http://tezi.toradex.com/Colibri-iMX7_LXDE-Image-Tezi_2.8b2.97/
Toradex Windows Embedded Compact 7
1.1, 2018-05-14, http://tezi.toradex.com/Colibri-iMX7_WinCE7Image-Tezi_1.1-20180514/
Toradex Windows Embedded Compact 2013
1.1, 2018-05-14, http://tezi.toradex.com/Colibri-iMX7_WinCE8Image-Tezi_1.1-20180514/
Toradex Windows Embedded Compact 7 Fast boot demo
1.1b4, 2018-03-22, http://tezi.toradex.com/Colibri-iMX7_WinCE7Image-FastBoot-Tezi_1.1b4...
Toradex Easy Installer
1.5, 20180507, http://tezi.toradex.com/Colibri-iMX7_ToradexEasyInstaller_1.5-20180507/

Network Information
Ethernet 10.22.1.67/24
USB RNDIS 192.168.11.1/24

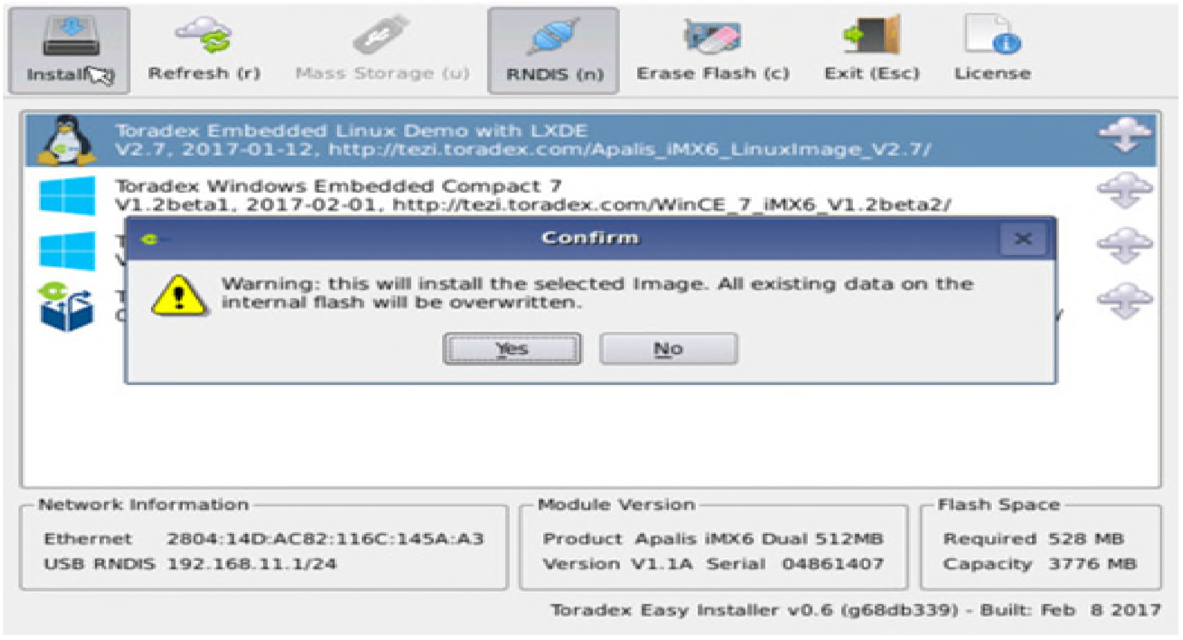
Module Version
Product Colibri iMX7 Dual 512MB
Version V1.1C Serial 02911038

Flash Space
Required Unknown
Capacity 512 MB

Toradex Easy Installer 1.5 (gf4e21de) - Built: May 7 2018

Figure 8: Toradex Easy Installer

13. Use the connected keyboard’s arrow key buttons to navigate and select the Toradex Embedded Linux Demo with LXDE and press "i" to install and enter to confirm



Toradex Embedded Linux Demo with LXDE
V2.7, 2017-01-12, http://tezi.toradex.com/Apalis_iMX6_LinuxImage_V2.7/
Toradex Windows Embedded Compact 7
V1.2beta1, 2017-02-01, http://tezi.toradex.com/WinCE_7_iMX6_V1.2beta2/

Warning: this will install the selected Image. All existing data on the internal flash will be overwritten.

Network Information
Ethernet 2804:14D:AC82:116C:145A:A3
USB RNDIS 192.168.11.1/24

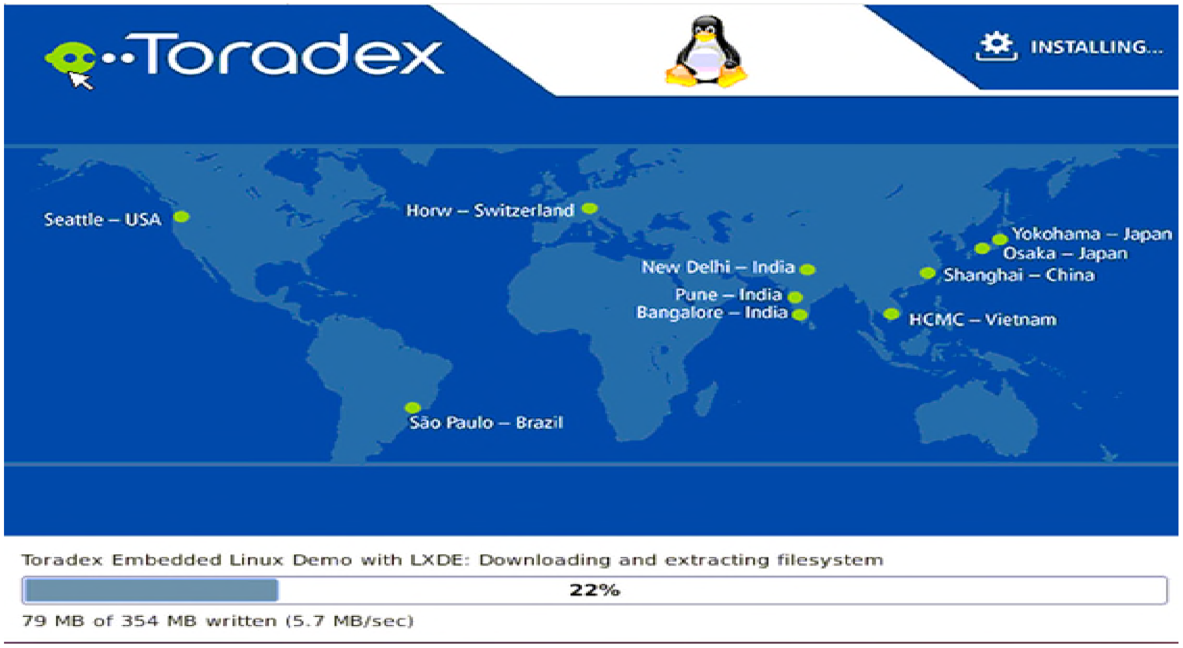
Module Version
Product Apalis iMX6 Dual 512MB
Version V1.1A Serial 04861407

Flash Space
Required 528 MB
Capacity 3776 MB

Toradex Easy Installer v0.6 (g68db339) - Built: Feb 8 2017

Figure 9: Confirm the Embedded Linux installation

14. Wait for the image to be downloaded and installed



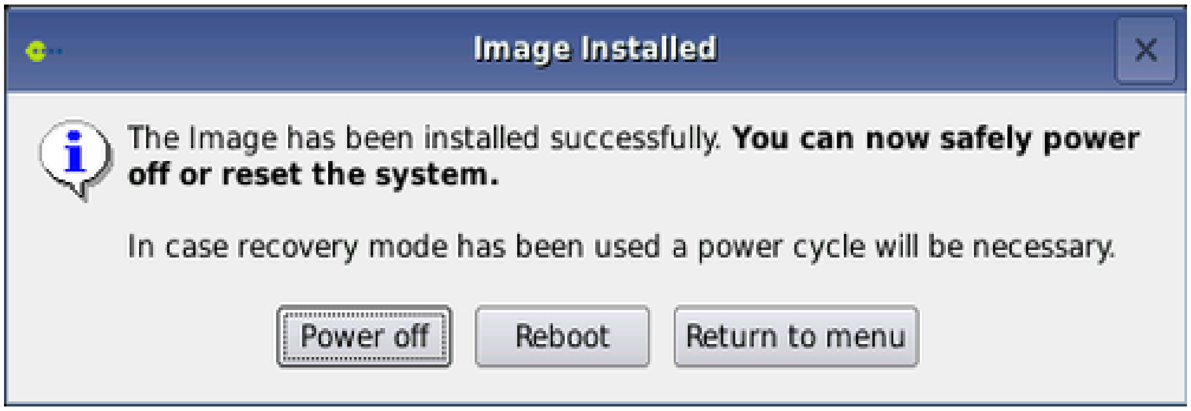
Toradex Embedded Linux Demo with LXDE: Downloading and extracting filesystem

22%

79 MB of 354 MB written (5.7 MB/sec)

Figure 10: Toradex image being downloaded and installed

15. Once the **Image Installed** message pops-up, use the on/off button or select the **Power off** option to shut down the system.



The Image has been installed successfully. **You can now safely power off or reset the system.**

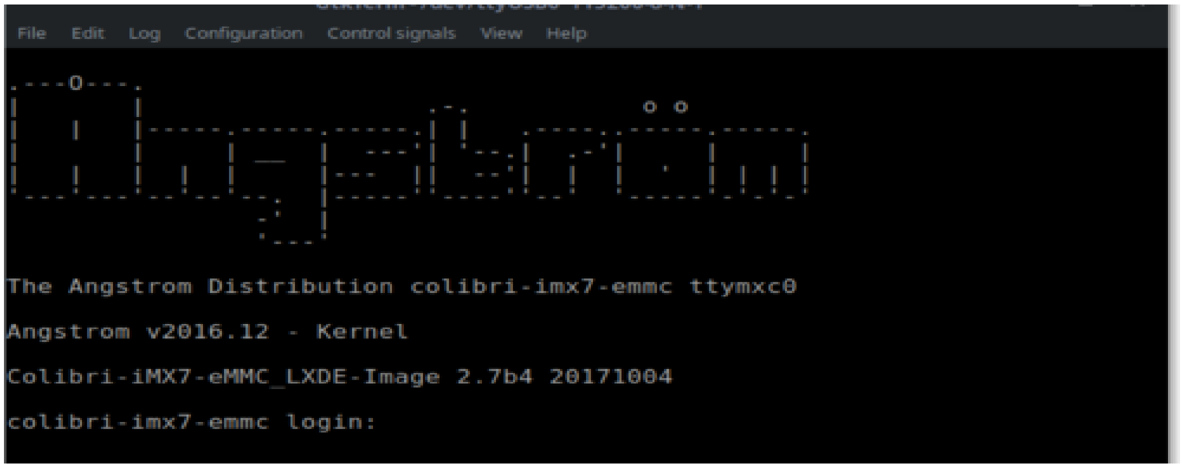
In case recovery mode has been used a power cycle will be necessary.

Figure 11: Installation finished

16. Unplug the USB Type-B to Type-A cable from **X50** and plug into **X29** (Serial port for Putty terminal)
17. You may remove the keyboard, monitor and/or mouse connected to the Toradex Apalis board
18. Check the dmesg for the serial port detection in the terminal for Linux Host or Device manager for Windows. In Linux usually it is detected as “/dev/ttyUSBx” where x is 0/1/2 etc. In Windows, it is detected as COMx, where x is decimal number
19. Open the Putty and configure the serial settings as below:

   a. Select the interface name as detected in the above step. For example "/dev/ttyUSB0" for Linux and COM6 for Windows
   b. Baud rate - 115200
   c. Then open the serial console

20. Now, power on the Toradex Apalis board
21. You should see the Linux boot up messages and finally a login prompt as shown in the picture below



```
The Angstrom Distribution colibri-imx7-emmc ttymxc0

Angstrom v2016.12 - Kernel

Colibri-iMX7-eMMC_LXDE-Image 2.7b4 20171004

colibri-imx7-emmc login:
```

Figure 12: Host terminal

22. Login as "root" user. No password required
23. Now, it's ready for testing Wi-Fi functionality

5.2.3.4 Burn the Image to SD card

This method is used if you don't want to erase the eMMC image and can simultaneously work by switching between images from two boot disks - SD card or eMMC

5.2.3.4.1 Requirements

In order to boot from SD card you need one SD card

5.2.3.4.2 Procedure

1. Copy the Linux image to one folder
2. Extract the Linux image by using below command

   $ tar –xvzf <Linux_image>

   For example $ tar –xvzf Apalis-iMX6_LXDE-Image-Tezi_2.8b6-20190624.tar

3. You can see bootfs and rootfs files. Ensure the path of these files is proper in the script (**Boot_from_SD_Toradex.sh)**
4. Now go to the extracted Toradex Release package folder

   $ cd <path to>/IV_Toradex_Relv1_USB

5. Run the below script to flash the image into SD card

$ sh Boot_from_SD_Toradex.sh

6. Remove the SD card from Linux machine and insert into 8 bit mmc card slot of Apalis carrier board.
7. Power on the carrier board and hit any key immediately after clicking power ON button to stay in U-Boot prompt mode.

You can now boot with **run sdboot** command.

NOTE: If you want to prepare step by step go to Boot from SD card

5.3 Toradex Ixora board

5.3.1 Prerequisites

This procedure is same as in Toradex Apalis board

5.3.2 Build procedure

This procedure is same as in Toradex Apalis board

5.3.3 Flash the image to eMMC

5.3.3.1 Requirements

5.3.3.1.1 Hardware requirements

- Toradex Ixora board
- Power adaptor - 12V 2.5A power supply
- Micro USB cable
- Small jumper male to male wire
- 8GB USB Stick, it is needed for eMMC boot
- Monitor
- HDMI cable
- Keyboard/ Mouse
- USB-RS232 converter cable

5.3.3.1.2 Software requirements

- Linux image for Toradex Ixora, which is obtained in Verify Output section
- A Laptop / Desktop with Putty software

5.3.3.2 Flashing Linux image to eMMC using Toradex Easy Installer

1. Download the Toradex Easy Installer in the host PC / Laptop
    a. Get the software from the below link
    https://docs.toradex.com/104375-apalis-imx6-toradexeasyinstaller.zip
    b. Unzip the zip file
2. Extract the Linux image (.tar file)

3. Format the USB memory stick
4. Move the extracted folder contents (not folder) into USB memory stick (i.e., go to that folder and then move the all files to memory stick)
5. Make sure the memory stick contain only the extracted folder contents only
6. Now you can safely eject the USB memory stick which is having the required Linux image
7. Insert remove USB stick from PC and plug it into X7 on Ixora board
8. Remove jumper JP2 on Ixora board
9. Connect the power supply into X2 port
10. Plug the Micro USB cable to Ixora OTG X9 port
11. Connect HDMI display cable between Ixora and Monitor
12. Connect the USB keyboard or Mouse to X8 port
13. Connect small jumper male-male cable to JP4 pins 1-2 for the system to enter recovery mode

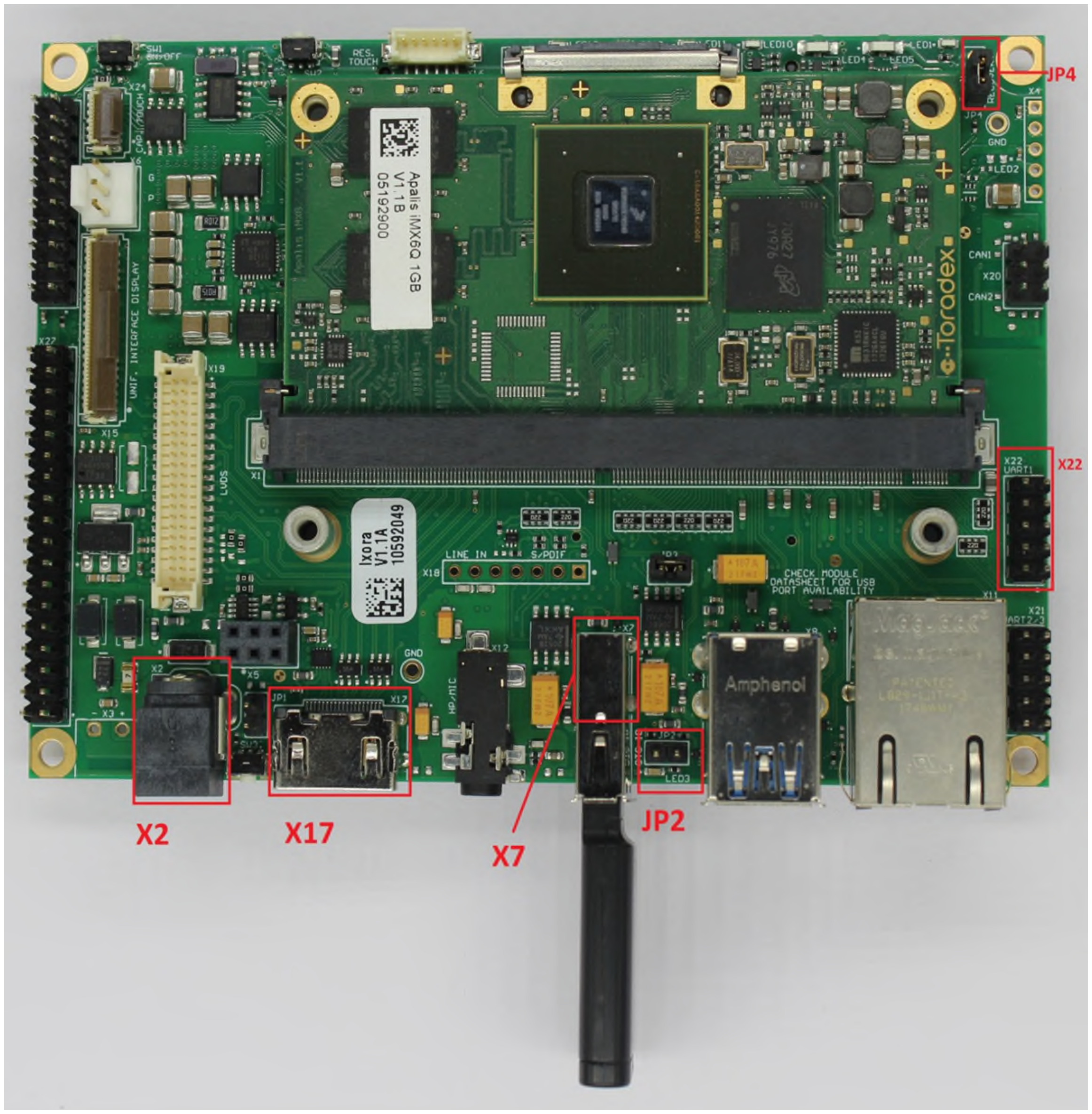
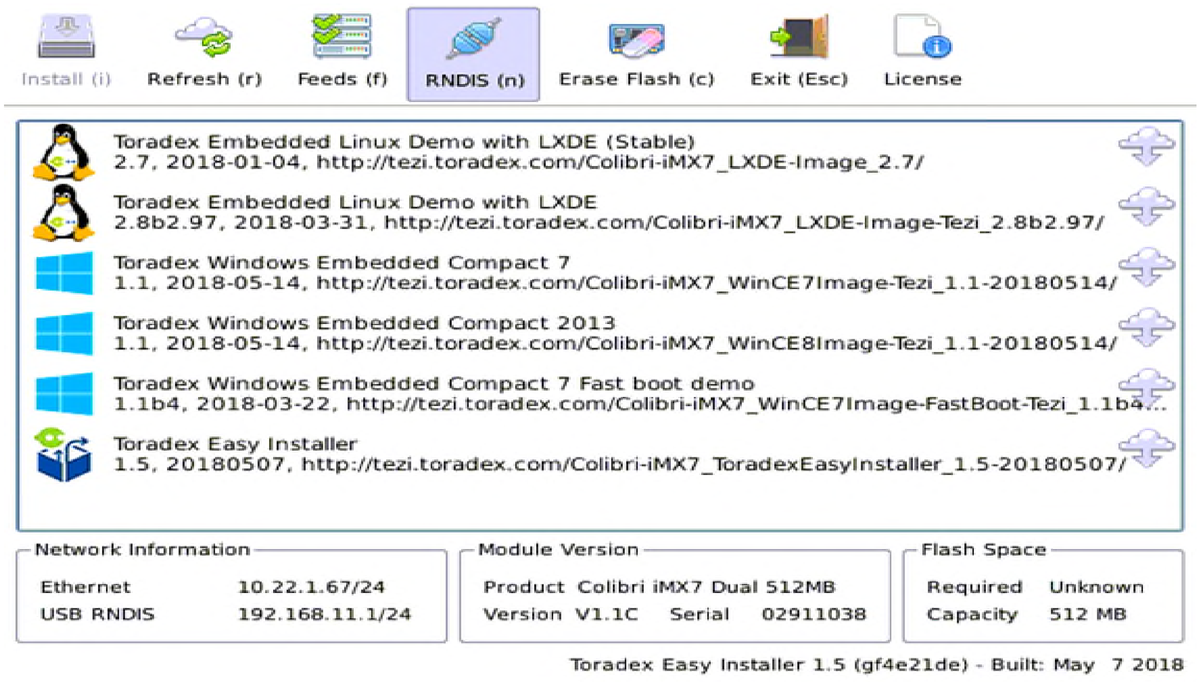


Figure 13: Recovery mode setup on Ixora

14. Power on the Ixora board by pressing SW1
15. On Linux/ Windows host, use one of the following scripts to download and run the Toradex Easy Installer through USB:
    i. Windows: **recovery-windows.bat**
    ii. Linux: **recovery-linux.sh**

Note: This does not write the Toradex Easy Installer to the flash, it only loads it into the module's RAM memory. You will have to redo these steps if you power off your module

16. You should see the Toradex Easy Installer displayed on the monitor attached to the HDMI port of the Ixora Evaluation Board
17. The image should appear in the list of available images with an icon indicating that it is an image found on a local storage device. Please check the screenshot below



Toradex Embedded Linux Demo with LXDE (Stable)
2.7, 2018-01-04, http://tezi.toradex.com/Colibri-iMX7_LXDE-Image_2.7/
Toradex Embedded Linux Demo with LXDE
2.8b2.97, 2018-03-31, http://tezi.toradex.com/Colibri-iMX7_LXDE-Image-Tezi_2.8b2.97/
Toradex Windows Embedded Compact 7
1.1, 2018-05-14, http://tezi.toradex.com/Colibri-iMX7_WinCE7Image-Tezi_1.1-20180514/
Toradex Windows Embedded Compact 2013
1.1, 2018-05-14, http://tezi.toradex.com/Colibri-iMX7_WinCE8Image-Tezi_1.1-20180514/
Toradex Windows Embedded Compact 7 Fast boot demo
1.1b4, 2018-03-22, http://tezi.toradex.com/Colibri-iMX7_WinCE7Image-FastBoot-Tezi_1.1b4...
Toradex Easy Installer
1.5, 20180507, http://tezi.toradex.com/Colibri-iMX7_ToradexEasyInstaller_1.5-20180507/

Network Information
Ethernet 10.22.1.67/24
USB RNDIS 192.168.11.1/24

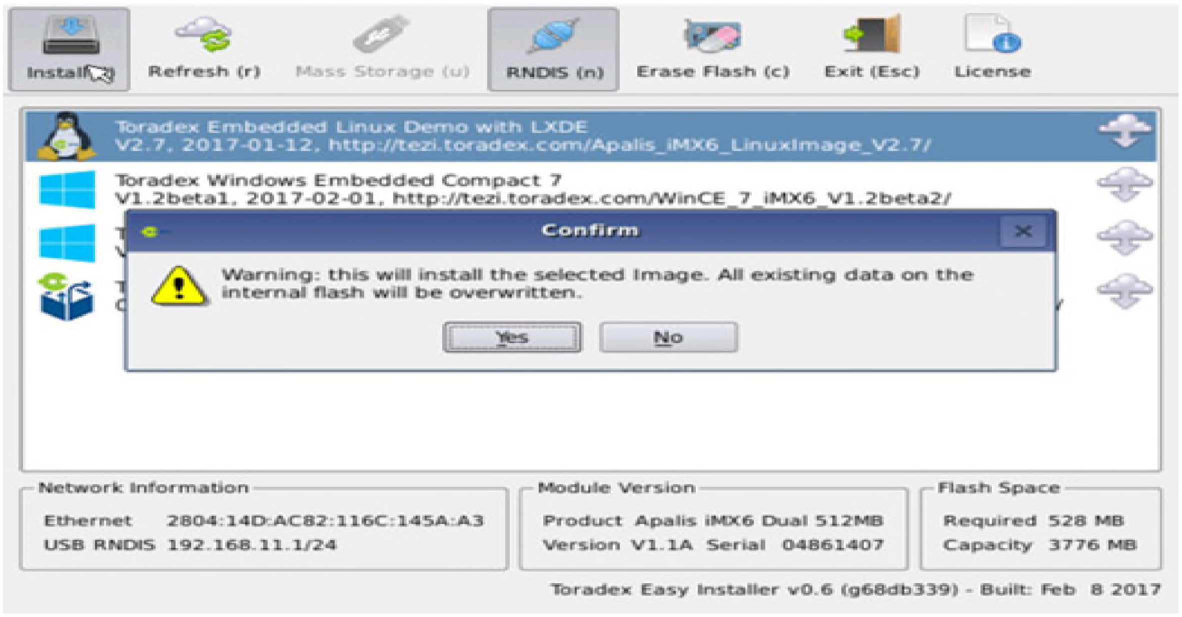
Module Version
Product Colibri iMX7 Dual 512MB
Version V1.1C Serial 02911038

Flash Space
Required Unknown
Capacity 512 MB

Toradex Easy Installer 1.5 (gf4e21de) - Built: May 7 2018

Figure 14: Toradex Easy Installer

18. Use the connected keyboard's arrow key buttons to navigate and select the Toradex Embedded Linux Demo with LXDE and press "i" to install and enter to confirm



Toradex Embedded Linux Demo with LXDE
V2.7, 2017-01-12, http://tezi.toradex.com/Apalis_iMX6_LinuxImage_V2.7/

Toradex Windows Embedded Compact 7
V1.2beta1, 2017-02-01, http://tezi.toradex.com/WinCE_7_iMX6_V1.2beta2/

Warning: this will install the selected image. All existing data on the internal flash will be overwritten.

Network Information
Ethernet 2804:14D:AC82:116C:145A:A3
USB RNDIS 192.168.11.1/24

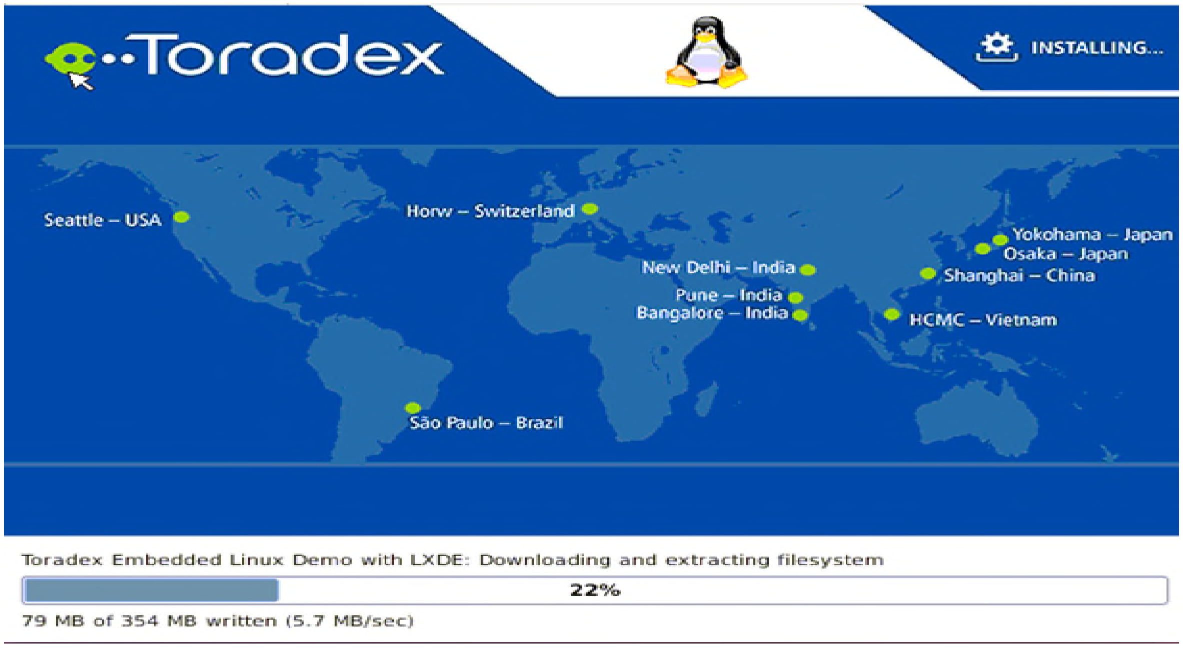
Module Version
Product Apalis iMX6 Dual 512MB
Version V1.1A Serial 04861407

Flash Space
Required 528 MB
Capacity 3776 MB

Toradex Easy Installer v0.6 (g68db339) - Built: Feb 8 2017

Figure 15: Confirm the Embedded Linux installation

19. Wait for the image to be downloaded and installed



Toradex Embedded Linux Demo with LXDE: Downloading and extracting filesystem
22%
79 MB of 354 MB written (5.7 MB/sec)

Figure 16: Toradex image being downloaded and installed

20. Once the **Image is Installed,** message pops-up, use the on/off button or select the **Power off** option to shut down the system.

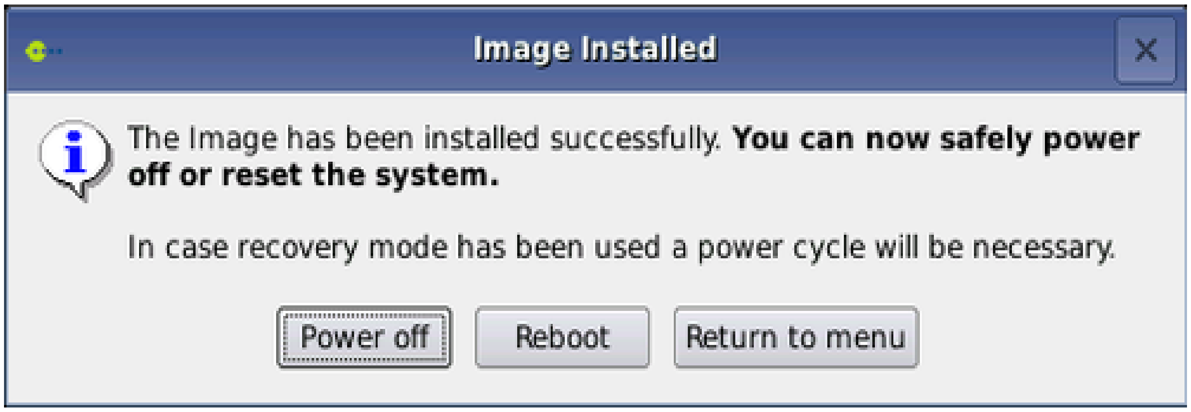


Image Installed

The Image has been installed successfully. **You can now safely power off or reset the system.**

In case recovery mode has been used a power cycle will be necessary.

Figure 17: Installation finished

21. Remove HDMI, USB keyboard/ Mouse, JP4 cable, USB memory stick
22. Keep JP2 jumper
23. Connect the USB-RS232 converter cable to X22 port on Ixora
24. Check the dmesg for the serial port detection in the terminal for Linux Host or Device manager for Windows. In Linux usually it is detected as "/dev/ttyUSBx" where x is 0/1/2 etc. In Windows, it is detected as COMx, where x is decimal number
25. Open the Putty and configure the serial settings as below:
    a. Select the interface name as detected in the above step. For example "/dev/ttyUSB0" for Linux and COM6 for Windows
    b. Baud rate - 115200
    c. Then open the serial console
26. Now, power on the Toradex Ixora board
27. You should see the Linux boot up messages and finally a login prompt
28. Login as "root" user. No password required
29. Now, it is ready for testing Wi-Fi functionality

6 Toradex Embedded Platform

6.1 Prerequisites

- It is recommended to use a Linux PC / Laptop with >=8GB RAM and high processing power for yocto builds.
- Make sure you have either Ubuntu 14.04 32-bit OS (or) 16.04 Ubuntu 64-bit OS installed on laptop/desktop. If not, please install and boot into one of them.
- Below procedure is tested on Ubuntu 14.04 32-bit OS
- Toradex_Driver_Relv0_USB.tar.gz

NOTE: Make sure the laptop is connected to the power socket as this process will take significant amount of time (few hours).

6.2 Procedure

6.2.1 Installation

NOTE: Run below commands as a user ($) but not as root (#)

Follow these steps for Yocto Embedded Linux build for the first time

- Install the repo bootstrap binary:

```
$ mkdir ~/bin
$ export PATH=~/bin:$PATH      (Set PATH environment variable to search in it)
$ curl http://commondatastorage.googleapis.com/git-repo-downloads/repo> ~/bin/repo
$ chmod a+x ~/bin/repo  # Set repo permissions
```

- *Repo* uses *Git*. Make sure you have it installed and user name and e-mail configured. Below is an example

```
$ sudo apt install git
$ git config --global user.name "John Doe"
$ git config --global user.email “johndoe@example.com”
```

- The stable branch version is 2.8 that is based on rocko.

```
$ mkdir oe-core
$ cd oe-core
$ repo init -u http://git.toradex.com/toradex-bsp-platform.git -b LinuxImageV2.8
$ repo sync
```

- Need to install some packages in the Linux PC for the build to proceed

  $ sudo apt-get install gawk wget git-core diffstat unzip texinfo gcc-multilib build-essential chrpath socat cpio python python3 python3-pip python3-pexpect xz-utils debianutils iputils-ping

  $ sudo apt-get install chrpath curl dosfstools gawk g++ gcc libcrypto++9 libcrypto++-dev liblzo2-dev lzop libsdl1.2-dev libstdc++-4.8-dev libusb-1.0-0 libusb-1.0-0-dev texinfo uuid-dev zlib1g (for 32 bit)

- Setup the build environment

  $. export

- Open *~/oe-core/build/conf/local.conf* file and make these changes:
  - In “Machine Selection” section, uncomment **MACHINE ?= ”apalis-imx6”** and comment other MACHINE selection lines
  - Need to add **ACCEPT_FSL_EULA = "1"** i.e., we are accepting the license of Freescale End User License Agreement.

NOTE: To find out more details follow https://developer.toradex.com/knowledge-base/board-support-package/openembedded-(core)

6.2.2 USB Driver Build

- Extract the Toradex_Driver_Rel_v0.tar.gz file by using below command

  $ tar –xvzf Toradex_Driver_Relv0_USB.tar.gz

- Copy the Driver file and layers to home directory by using below commands

  $ cd Toradex_Driver_Relv0_USB

  $ cp kernel-module-wlan_usb.tgz ~/

- Copy layers **meta-qcacld(patch and firmware)** and **meta-wlan(driver)** to Toradex layers folder "oe-core/layers"

  $ cp <path_to>/Toradex_Driver_Relv0_USB/ meta-qcacld ~/oe-core/layers

  $ cp <path_to>/Toradex_Driver_Relv0_USB/meta-wlan ~/oe-core/layers

- Add the layers to the Yocto build using below command, make sure you are in build folder.

  $ bitbake-layers add-layer ../layers/meta-qcacld

  $ bitbake-layers add-layer ../layers/meta-wlan

- Add the below defines in ~/oe-core/layers/meta-toradex-nxp/recipes-kernel/linux/linux-toradex-<kernel_version>/apalis-imx6/defconfig

  CONFIG_WIRELESS=y

  CONFIG_WEXT_CORE=y

CONFIG_CFG80211=m

CONFIG_NL80211_TESTMODE=y

CONFIG_CFG80211_WEXT=y

CONFIG_CFG80211_INTERNAL_REGDB=y

CONFIG_MAC80211=y

CONFIG_MMC_DEBUG=y

- Please find the **kernel-module-wlan_0.1.bb** in *~/oe-core/layers/meta-wlan/recipes-kernel/kernel-modules*
- Need to mention the directory of driver file(kernel-module-wlan_usb.tgz) at SRC_URI in **kernel-module-wlan_0.1.bb** file
- In **kernel-module-wlan_0.1.bb** file, verify the **COMPATIBLE_MACHINE** is **${MACHINE}** or not.
- So finally the **kernel-module-wlan_0.1.bb** file should be like

```
SUMMARY = "QCACLD external Linux kernel module"

LICENSE = "GPLv2"

LIC_FILES_CHKSUM = "file://COPYING.MIT;md5=3da9cfbcb788c80a0384361b4de20420"

inherit module

#kernel-module-wlan path

SRC_URI = "file://{TOPDIR}/../../kernel-module-wlan_usb.tgz"

#CFLAGS += "-Wno-error -Wno-format-truncation -Wno-maybe-uninitialized -Wno-implicit-fallthrough"

S = "${WORKDIR}"

# The inherit of module.bbclass will automatically name module packages with

# "kernel-module-" prefix as required by the oe-core build environment.

COMPATIBLE_MACHINE = "(apalis-imx6)"
```

- Edit the build/conf/local.conf file and add the below(same as below) statements to include the required packages to the final kernel image
  - **IMAGE_INSTALL_append = " iw openssl openssh wireless-tools \
    wpa-supplicant hostapd dhcp-client dhcp-server iperf3 \
    kernel-module-wlan "**
  - **IMAGE_INSTALL += "linux-firmware-example"**
  - **IMAGE_INSTALL += "linux-firmware-ini"**
- Remove the back ports by following below steps

- In meta-toradex-demos/recipes-images/images/tdx-extra.inc file
  Remove back ports at IMAGE_INSTALL_remove_apalis-tk1-mainline = " \
  backports "

  Example:

  ```
  IMAGE_INSTALL_remove_apalis-tk1-mainline = " \
  "
  ```

- in local.conf add:
  BBMASK = "meta-toradex-bsp-common/recipes-kernel/backports/"
  IMAGE_INSTALL_remove = "backports"

- *Set up the bitbake environment*

  $. export

- Bitbake is a tool to build the Linux image. It will schedule the build related tasks in an order

  $ bitbake -k angstrom-lxde-image

- Build takes 4-5 hours to complete.

6.3 Boot from SD card

This method is used if you don't want to erase the eMMC image and can work by switching between images from two boot disks - SD card or eMMC

6.3.1 Requirements

- 1 x SD card

6.3.2 Procedure

1. Insert the SD card into a Linux machine.
2. If any partitions exist, delete that partitions and format the SD Card
3. Open Terminal
4. Type dmesg command to know the device name corresponding to your SD card. It may show up with the name **sdb** (or) **sdc** or similar to that. Please note this as we have to use this name in the following commands. In the below commands, **sdb** is taken as an example to describe the commands. However, please use the appropriate name as per detection **dmesg** log.
5. Type the following commands to make the partitions in SD Card. The order of partitions (as in the following commands) is important since U-Boot will look for the kernel in first partition, for the rootfs in the second partition.

   $ sudo parted -s /dev/sdb mklabelgptmkpart primary fat32 1MiB 100MiB mkpart ext4 100MiB 2000MiB name 1 boot name 2 rootfs

   $ sync

```
$ sudo mkfs.fat -n boot /dev/sdb1 && sudo mkfs.ext4 -L rootfs /dev/sdb2

$ sync

$ sudo lsblk /dev/sdb -o NAME,FSTYPE,LABEL                    (to verify)

$ sync
```

6. You have to create mount points to mount the SD card partitions. For that create two new directories at /mnt.
7. Issue the following two commands

```
$ mkdir /mnt/boot

$ mkdir /mnt/rootfs
```

8. Mount the partitions to be able to access them

```
$ sudo mount -L boot /mnt/boot

$ sudo mount -L rootfs /mnt/rootfs
```
Here -L boot and -L rootfs are the labels created in step 6.

9. You need to copy the kernel image and the device trees into the first partition, the rootfs into the second partition.
10. Extract the image, find bootfs, rootfs tar files in the extracted image directory
11. Go to the path where the extracted image folders is available

```
$ sudo tar xf Apalis-iMX6_LXDE-Image.bootfs.tar.xz --no-same-owner -C /mnt/boot

$ sync

$ sudo tar xf Apalis-iMX6_LXDE-Image.rootfs.tar.xz -C /mnt/rootfs

$ sync
```

12. Finally unmount the partitions with

```
$ sudo umount /mnt/boot

$ sync

$ sudo umount /mnt/rootfs

$ sync
```

13. Remove the SD card from Linux machine and insert into 8 bit mmc card slot of carrier board.
14. Power on the carrier board and hit any key immediately after clicking power ON button to stay in U-Boot prompt mode.

You can now boot with **run sdboot** command.

15. You should see the Linux boot up messages and finally a login prompt as shown in the Figure 14.
16. Login as “root” user. No password required. Now, it is ready for testing Wi-Fi functionality

7 Evaluation on Freescale i.MX6 SoloLite EVK

7.1.1 Requirements

- EVIA USB /
    - o / USB M.2 Module Evaluation Kit
- i.MX 6SoloLite Evaluation Kit. The kit contents are as follows:
    - o Board: MCIMX6SLEVK
- Micro USB USB M.2 Evaluation Kit contents are as follows:
    - o EVIA USB cable
    - o Power supply: 100/240 V input, 5 V, 2.4 A output W/AC adaptor
    - o SD card: Programmed Linux
- Linux PC with Serial-to-USB FTDI drivers installed – this will be used to communicate with the i.MX6 platform.
- Make sure uninterrupted internet connection exist until image is built
- Software Requirements
    - o BALI USB iMX Sololite Release Package: IV_sololite_Relv1_USB.tgz
    - o This document assumes that Ubuntu installed high end PC is used.
    - o The recommended Ubuntu version is 16.04 64 bit
    - o Minimum hard disk space required is about 50 GB and 4GB of RAM.

7.1.2 Hardware Setup

1. Insert bootable SD card into J21 SD2 4-Bit BOOT slot
2. Connect the i.MX6 board to the Linux PC using the USB-to-micro USB cable – the cable has to be connected to port J26 (micro USB) of the board
3. Connect power supply adapter to jack J6
4. Connect the BALI USB Evaluation Kit (EVK) to the i.MX6 board
5. Toggle SW14 switch to ON position

7.1.3 Evaluation with pre-loaded Linux image in SD Memory

iVativ provides SD Memory cards with Linux image and BALI Wi-Fi EVB drivers. Users can evaluate the Wi-Fi module without building the kernel. Follow the below steps to setup the hardware to evaluate the module.

1. Connect micro USB cable between iMX6 USB-UART connector and PC USB.
2. Open minicom/ Hyperterminal/ Putty.
3. Configure port name as ttyUSB0 in Linux OR COMx in Windows, baud rate 115200, Flow control as “No”.
4. Power ON imx6slevk and observe boot messages on the Terminal.
5. Type root after boot up to Log in as root user

6. Please follow software section

7.1.4 Yocto Project Linux Kernel Building for i.MX6 Sololite EVK

7.1.4.1 Host Packages

A Yocto Project build requires that some packages be installed for the build that are documented under the Yocto Project. You can go to Yocto Project Quick Start and check for the packages that must be installed for your build machine.

1. Essential Yocto Project host packages are:

   $ sudo apt-get install gawk wget git-core diffstat unzip texinfo gcc-multilib build-essential chrpath socat libsdl1.2-dev cpio python python3 python3-pip python3-pexpect xz-utils debianutils iputils-ping curl ncurses-dev

7.1.4.2 Image building

1. Extract the IV_sololite_Relv1_ USB.tgz file to IV_sololite_Relv1_ USB directory
2. Change to IV_sololite_Relv1_ USB directory and give below command
   $ source setup-environment build
   The path automatically changes to build directory.
2. Try to free up the RAM available
3. To compile the yocto file system, execute the below command

   $ bitbake core-image-base

4. After a build is complete, the created image resides in <build-x11>/tmp/deploy/images/imx6slevk/. An SD card image provides the full system to boot with U-Boot and kernel. Check the device ID(X) of SD card under /dev like /dev/sda, /dev/sdb. To flash an SD card image, run the following.
   $ gunzip -c <image-name>.wic.gz | sudo dd of=/dev/sdX bs=1M && sync

   Ex: gunzip -c core-image-base-imx6slevk-20181112080812.rootfs.wic.gz | sudo dd of=/dev/sdb bs=1M && sync

5. To re-build the yocto image, execute the below commands.
   $ bitbake -c cleanall core-image-base

   $ bitbake core-image-base

6. Follow from section "*Hardware setup*"

8 Issues/Errata

1. Current consumption measurements are not completed, numbers will be published soon.
2. WLAN_LED is reserved for future use