**Notice:** This document contains information on products in the design phase of development. The information here is subject to change without notice. Do not finalize a design with this information.

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL’S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

A “Mission Critical Application” is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL’S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS’ FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

Intel software products are copyrighted by and shall remain the property of Intel Corporation. Use, duplication, or disclosure is subject to restrictions stated in Intel’s Software License Agreement, or in the case of software delivered to the government, in accordance with the software license agreement as defined in FAR 52.227-7013.

All Code placed under the MIT License. Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS," WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

The main algorithm has been derived from https://github.com/bagilevi/android-pedometer, which has copyleft license (without any real license). Still this item needs to be checked with legal to make sure there are no compatibility issues. Some functions have been taken from MPU6050 specific libraries which shares the same licensing conditions with this code.

Some of the code is taken or derived from i2c-dev.h - i2c-bus driver, char device interface (Copyright ©1995-97 Simon G. Vogl Copyright © 1998-99 Frodo Looijaard frodol@dds.nl) and from i2c.h (Copyright © 2013 Parav Nagarsheth), which is under GNU General Public License as published by the Free Software Foundation.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

The code names presented in this document are only for use by Intel to identify products, technologies, or services in development that have not been made commercially available to the public, i.e., announced, launched, or shipped. They are not "commercial" names for products or services and are not intended to function as trademarks.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature may be obtained by calling 1-800-548-4725 or by visiting Intel’s website at http://www.intel.com/design/literature.htm.

Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. See http://www.intel.com/products/processor_number for details.

Intel and the Intel logo are trademarks of Intel Corporation in the US and other countries.

* Other brands and names may be claimed as the property of others.
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
<td>7</td>
</tr>
<tr>
<td>1.1</td>
<td>References</td>
<td>7</td>
</tr>
<tr>
<td>1.2</td>
<td>Acronyms and abbreviations</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>Software Architecture</td>
<td>9</td>
</tr>
<tr>
<td>2.1</td>
<td>Features</td>
<td>9</td>
</tr>
<tr>
<td>2.2</td>
<td>Services</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>First Steps</td>
<td>11</td>
</tr>
<tr>
<td>3.1</td>
<td>Different ways to connect</td>
<td>11</td>
</tr>
<tr>
<td>3.1.1</td>
<td>Serial connection</td>
<td>11</td>
</tr>
<tr>
<td>3.1.2</td>
<td>SSH connection</td>
<td>11</td>
</tr>
<tr>
<td>3.1.3</td>
<td>Manually powering Wi-Fi up or down</td>
<td>12</td>
</tr>
<tr>
<td>4</td>
<td>Setting up Wi-Fi connections</td>
<td>13</td>
</tr>
<tr>
<td>4.1</td>
<td>Configuring an Intel® Edison device as a Wi-Fi client</td>
<td>13</td>
</tr>
<tr>
<td>4.2</td>
<td>wpa_supplicant</td>
<td>14</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Scan available networks</td>
<td>15</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Connect to a protected AP</td>
<td>15</td>
</tr>
<tr>
<td>4.2.3</td>
<td>Connect to an open AP</td>
<td>16</td>
</tr>
<tr>
<td>4.2.4</td>
<td>Create/connect to an ad-hoc network</td>
<td>17</td>
</tr>
<tr>
<td>4.2.5</td>
<td>Display wpa supplicant traces</td>
<td>19</td>
</tr>
<tr>
<td>4.3</td>
<td>Connman</td>
<td>19</td>
</tr>
<tr>
<td>4.3.1</td>
<td>Starting connman</td>
<td>19</td>
</tr>
<tr>
<td>4.3.2</td>
<td>Scan available networks</td>
<td>20</td>
</tr>
<tr>
<td>4.3.3</td>
<td>Connect to an open AP</td>
<td>21</td>
</tr>
<tr>
<td>4.3.4</td>
<td>Connect to a protected AP</td>
<td>22</td>
</tr>
<tr>
<td>4.3.5</td>
<td>Display connman traces</td>
<td>22</td>
</tr>
<tr>
<td>5</td>
<td>Access Point Setup</td>
<td>23</td>
</tr>
<tr>
<td>5.1</td>
<td>Software-enable access point setup</td>
<td>23</td>
</tr>
<tr>
<td>5.2</td>
<td>Software-enable access point setup using hidden SSID</td>
<td>24</td>
</tr>
<tr>
<td>6</td>
<td>Wi-Fi Direct</td>
<td>27</td>
</tr>
<tr>
<td>6.1</td>
<td>Known limitations</td>
<td>27</td>
</tr>
<tr>
<td>6.2</td>
<td>Using Wi-Fi Direct</td>
<td>27</td>
</tr>
<tr>
<td>6.3</td>
<td>Basic peer-to-peer setup</td>
<td>28</td>
</tr>
<tr>
<td>6.4</td>
<td>P2P connection with PIN mode as autonomous GO</td>
<td>30</td>
</tr>
<tr>
<td>6.5</td>
<td>P2P connection with PIN mode with group negotiation</td>
<td>33</td>
</tr>
<tr>
<td>6.6</td>
<td>P2P connection in PBC mode with first device as GO (with group negotiation)</td>
<td>34</td>
</tr>
<tr>
<td>6.7</td>
<td>Terminate and reinvoke a connection</td>
<td>35</td>
</tr>
<tr>
<td>6.8</td>
<td>P2P connection with WPS-PBC mode as autonomous GO</td>
<td>37</td>
</tr>
<tr>
<td>6.9</td>
<td>P2P connection using PBC method</td>
<td>38</td>
</tr>
<tr>
<td>6.9.1</td>
<td>Disconnecting and reconnecting</td>
<td>41</td>
</tr>
<tr>
<td>6.10</td>
<td>Wi-Fi Direct troubleshooting</td>
<td>42</td>
</tr>
<tr>
<td>6.10.1</td>
<td>Cycling power</td>
<td>42</td>
</tr>
<tr>
<td>6.10.2</td>
<td>Wi-Fi multirole</td>
<td>43</td>
</tr>
<tr>
<td>7</td>
<td>Additional Configurations</td>
<td>44</td>
</tr>
<tr>
<td>7.1</td>
<td>Disable power management</td>
<td>44</td>
</tr>
<tr>
<td>7.2</td>
<td>DHCP client vs. static IP address</td>
<td>44</td>
</tr>
<tr>
<td>7.3</td>
<td>Autoreconnect after a reboot</td>
<td>45</td>
</tr>
</tbody>
</table>
8 Wi-Fi Software Components

8.1 Source folders
8.2 Binary files
8.3 Recipe folders
8.4 Build commands
8.5 Files copied onto root of the Intel® Edison file system
8.6 Driver modules

9 Sample Scripts

9.1 Script to connect to an open AP
9.2 Script to connect to a secure AP (with WPA2-PSK)
9.3 Script to disable power management
9.4 Script to create/connect to an ad-hoc network

Figures

Figure 1 Software architecture
Figure 2 AP and STA modes
Figure 3 ifconfig after Wi-Fi device has powered up
Figure 4 configure_edison --wifi
Figure 5 Device information webpage
Figure 6 wpa_supplicant status
Figure 7 Scan result
Figure 8 Connecting to a protected AP
Figure 9 Connecting to a protected AP
Figure 10 Connection to open AP with wpa_cli command
Figure 11 wpa_cli status
Figure 12 Creating an ad-hoc network
Figure 13 Scanning for an ad-hoc network
Figure 14 Connecting to an ad-hoc network
Figure 15 Configuring a static IP address with ifconfig on the first device
Figure 16 Configuring a static IP address with ifconfig on the second device
Figure 17 Starting connmanctl
Figure 18 Quitting connmanctl
Figure 19 Network scan output
Figure 20 hostapd.conf file
Figure 21 Ignore broadcast SSID
Figure 22 Configure as AP with hostapd
Figure 23 IP address of the wlan0 interface
Figure 24 wpa_supplicant status
Figure 25 Running the secure AP script
Figure 26 Connection status
Figure 27 Ping
Figure 28 Establishing a P2P connection between two Intel® Edison devices
Figure 29 Verifying on the first Intel® Edison device
Figure 30 Verifying on the second Intel® Edison device
Figure 31 Verifying on the first Intel® Edison device
Figure 32 Verifying on the second Intel® Edison device
Figure 33 P2P connection between an Intel® Edison device and an Android* device using PBC method
Figure 34 Enabling Wi-Fi Direct on an Android* device
Figure 35 Peer devices on an Android*
Figure 36 Android* invitation to connect
Figure 37  Intel® Edison connected ................................................................................................................................. 40
Figure 38  Query status .......................................................................................................................................................... 41
Figure 39  Android* remembered groups .......................................................................................................................... 41
Figure 40  Android* reconnected .......................................................................................................................................... 42
Figure 41  Available wireless interfaces .......................................................................................................................... 43
Figure 42  Power Management status .................................................................................................................................. 44

Tables

Table 1  Terminology .............................................................................................................................................................. 8
Table 2  Pros and cons of connman and wpa_cli .................................................................................................................... 9
Table 3  Basic P2P commands ................................................................................................................................................. 27
## Revision History

<table>
<thead>
<tr>
<th>Revision</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>Initial release.</td>
<td>February 4, 2015</td>
</tr>
</tbody>
</table>

§
1 Introduction

This document is for software developers providing Wi-Fi guidelines on the Intel® Edison development platform. The reader should have a basic knowledge of the Linux operating system and Wi-Fi connectivity.

1.1 References

<table>
<thead>
<tr>
<th>Reference</th>
<th>Name</th>
<th>Number/location</th>
</tr>
</thead>
<tbody>
<tr>
<td>331191</td>
<td>Intel® Edison Kit for Arduino® Hardware Guide</td>
<td></td>
</tr>
<tr>
<td>329686</td>
<td>Intel® Galileo and Intel® Edison Release Notes</td>
<td></td>
</tr>
<tr>
<td>[GSG]</td>
<td>Intel® Edison Getting Started Guide</td>
<td></td>
</tr>
<tr>
<td>331438</td>
<td>Intel® Edison Wi-Fi Guide</td>
<td>(This document)</td>
</tr>
<tr>
<td>331704</td>
<td>Intel® Edison Bluetooth® Guide</td>
<td></td>
</tr>
<tr>
<td>Yocto Project</td>
<td></td>
<td><a href="http://www.yoctoproject.org/docs/1.5.1/dev-manual/dev-manual.html">http://www.yoctoproject.org/docs/1.5.1/dev-manual/dev-manual.html</a></td>
</tr>
<tr>
<td>hostapd</td>
<td><a href="http://w1.fi/hostapd">http://w1.fi/hostapd</a></td>
<td></td>
</tr>
<tr>
<td>Linux WPA/WPA2/IEEE 802.1X Supplicant</td>
<td><a href="http://w1.fi/wpa_supplicant">http://w1.fi/wpa_supplicant</a></td>
<td></td>
</tr>
<tr>
<td>wpa_cli</td>
<td><a href="http://linux.die.net/man/8/wpa_cli">http://linux.die.net/man/8/wpa_cli</a></td>
<td></td>
</tr>
<tr>
<td>iwconfig</td>
<td><a href="http://www.linuxcommand.org/man_pages/iwconfig8.html">http://www.linuxcommand.org/man_pages/iwconfig8.html</a></td>
<td></td>
</tr>
</tbody>
</table>
### 1.2 Acronyms and abbreviations

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES</td>
<td>Advanced Encryption Standard</td>
</tr>
<tr>
<td>AP</td>
<td>Access point</td>
</tr>
<tr>
<td>DHCP</td>
<td>Dynamic Host Configuration Protocol</td>
</tr>
<tr>
<td>DUT</td>
<td>Device under test</td>
</tr>
<tr>
<td>GO</td>
<td>Group owner</td>
</tr>
<tr>
<td>IBSS</td>
<td>Independent basic service set</td>
</tr>
<tr>
<td>MAC</td>
<td>Media Access Control</td>
</tr>
<tr>
<td>P2P</td>
<td>Peer to peer</td>
</tr>
<tr>
<td>PBC</td>
<td>Push button control</td>
</tr>
<tr>
<td>PIN</td>
<td>Personal identification number</td>
</tr>
<tr>
<td>PSK</td>
<td>Preshared key</td>
</tr>
<tr>
<td>SoftAP</td>
<td>Software-enabled access point</td>
</tr>
<tr>
<td>ssh</td>
<td>Secure shell</td>
</tr>
<tr>
<td>SSID</td>
<td>Service set identifier</td>
</tr>
<tr>
<td>STA</td>
<td>Station</td>
</tr>
<tr>
<td>UART</td>
<td>Universal asynchronous receiver/transmitter</td>
</tr>
<tr>
<td>WEP</td>
<td>Wired equivalent privacy</td>
</tr>
<tr>
<td>WPA</td>
<td>Wi-Fi protected access</td>
</tr>
<tr>
<td>WPA_CLI</td>
<td>WPA command line client</td>
</tr>
<tr>
<td>WPS</td>
<td>Wi-Fi protected setup</td>
</tr>
</tbody>
</table>
2 Software Architecture

The Intel® Edison development board includes the BCM43340—a combination Bluetooth/Wi-Fi chip. This connectivity module enables system-level connectivity features. This chapter includes an overview of the overall Wi-Fi stack and highlights the components that provide connectivity services to applications.

2.1 Features

Wi-Fi feature list
- 802.11 a/b/g/n support.
- WEP encryption.
- WPA encryption.
- WPA2 encryption.
- STA mode.
- AP mode.
- Wakeup on Wi-Fi.
- Multirole
- Wi-Fi Direct/P2P

Connection Manager

There are two user interfaces available to control network services:
- connman.
- wpa_cli.

Firmware and drivers
- Broadcom provides the open source Linux kernel drivers. (The firmware and drivers are included in our tar download.)

2.2 Services

Refer to Table 2 to determine whether wpa_cli or connman is the correct tool for your application. Both of these are capable of managing Wi-Fi connections, but each has advantages and disadvantages. Your choice may depend on your Wi-Fi and WLAN expertise and on how you intend to integrate WLAN into your middleware.

<table>
<thead>
<tr>
<th>Table 2 Pros and cons of connman and wpa_cli</th>
</tr>
</thead>
<tbody>
<tr>
<td>connman</td>
</tr>
<tr>
<td><strong>Pros</strong></td>
</tr>
<tr>
<td>Supported on Intel® Galileo platform.</td>
</tr>
<tr>
<td>Easy to use.</td>
</tr>
<tr>
<td>Exports simple DBUS API for middleware.</td>
</tr>
<tr>
<td>Simple CLI commands. (You do not need to be a Wi-Fi expert.)</td>
</tr>
<tr>
<td><strong>Cons</strong></td>
</tr>
<tr>
<td>P2P connection not yet supported.</td>
</tr>
<tr>
<td>Not all EAP mode supported.</td>
</tr>
<tr>
<td>Concurrent P2P/STA configuration not supported.</td>
</tr>
</tbody>
</table>

Once you have decided on the tool, download it:
- wpa_cli command line client: [http://linux.die.net/man/8/wpa_cli](http://linux.die.net/man/8/wpa_cli)
- connman command line: [https://wiki.archlinux.org/index.php/Connman](https://wiki.archlinux.org/index.php/Connman)
Figure 1 shows the basic software architecture.

Note the following:

- **hostapd** is a user space daemon for access point and authentication server, which is used to create a virtual Wi-Fi access point.
- **wpa_supplicant** is a user space daemon that runs in the background and acts as the backend component controlling the wireless connection.
- Broadcom does not support STA/AP concurrent mode. So **hostapd** and **wpa_supplicant** cannot run simultaneously.
- In STA mode, you can use either **connman** or **wpa_cli** to manage the STA connection.
- In P2P mode, only **wpa_cli** is supported to manage the P2P connection.
3 First Steps

Before you attempt to connect your Intel® Edison device to a Wi-Fi network, make sure you complete these steps.

1. If you are using your Intel® Edison compute module for the first time, follow the OS-specific guidelines in the How-tos section of the Intel® Edison webpage: http://www.intel.com/support/maker/edison.htm#how.

2. Make sure that your device has the latest image. Visit the Support and downloads page to download the latest firmware: http://www.intel.com/support/edison/sb/CS-035180.htm.

3.1 Different ways to connect

After you complete the preparatory steps as described above, you can configure your Intel® Edison devices in either of the following operating modes:

- **Access point (AP) mode.** In this mode, an Intel® Edison device can act as a master to enable multiple Wi-Fi adapters (CLIs) to connect to it.

- **Station (STA) mode.** This is the default mode for any wireless driver. In STA mode, an Intel® Edison device can act as a client to connect with a Wi-Fi access point.

![Figure 2 AP and STA modes](image)

3.1.1 Serial connection

To connect an Intel® Edison device via serial console (UART), from a terminal enter the following:

```
$ sudo screen /dev/ttyUSB0 115200
```

You may need to enter your password. At the blank screen, press Enter twice, then log on as root. If you do not have “screen” installed, install it now by opening a new terminal window and entering `sudo apt-get install screen`.

3.1.2 SSH connection

To connect an Intel® Edison device via SSH (Ethernet over USB), from a terminal enter the following:

```
$ ssh root@192.168.2.15
```

Note: 192.168.2.15 is the Intel® Edison board’s static IP address.
3.1.3 Manually powering Wi-Fi up or down

If the `ifconfig` command lists the wlan0 interface in its results, Wi-Fi is powered up; if it does not list wlan0, Wi-Fi is powered down. You can power up or down the chipset with these commands:

```
$ ifconfig wlan0 up
$ ifconfig wlan0 down
```

From a terminal, use the `ifconfig` command to check if the network interface is configured (Figure 3).

**Figure 3** `ifconfig` after Wi-Fi device has powered up

```
root@edison:~# ifconfig
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:1120 errors:0 dropped:0 overruns:0 frame:0
    TX packets:1120 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:0
    RX bytes:84480 (82.5 Kb)  TX bytes:84480 (82.5 Kb)

usb0      Link encap:Ethernet  HWaddr 02:00:86:73:b5:2e
    inet addr:192.168.2.15  Bcast:192.168.2.255  Mask:255.255.255.0
    inet6 addr: fe80::86ff:fe73:b52e/64 Scope:Link
    UP BROADCAST MULTICAST  MTU:1500  Metric:1
    RX packets:43 errors:0 dropped:0 overruns:0 frame:0
    TX packets:18 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:1000
    RX bytes:7231 (7.0 Kb)  TX bytes:4284 (4.1 Kb)

wlan0     Link encap:Ethernet  HWaddr fc:2e:de:34:5e:31
    inet addr: fe80::fecd:deff:fe34:5e31/64 Scope:Link
    UP BROADCAST MULTICAST DYNAMIC  MTU:1500  Metric:1
    RX packets:0 errors:0 dropped:0 overruns:0 frame:0
    TX packets:14 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:1000
    RX bytes:0 (0.0 B)  TX bytes:2721 (2.6 Kib)
root@edison:~#
```

**Note:** The `rfkill unblock/block wlan` command is generally used to power up/down the WLAN.
4 Setting up Wi-Fi connections

This chapter shows how to set up Wi-Fi connections with standard command-line tools like connman and wpa_cli, which allow you to configure and connect to a Wi-Fi network.

By default, the Wi-Fi driver is loaded in the down state at boot time, so Wi-Fi is completely powered down. You will have to start wpa_supplicant or hostapd at boot time, depending on whichever one you decide to use. Intel provides the configure_edison --wifi script, which will start either one for you.

- systemctl: This is used to introspect and control the state of the systemd system and service manager. (Refer to the system man page for details.)
- wpa_cli: This text-based frontend program for interacting with wpa_supplicant is used to query current status, change configuration, trigger events, and request interactive user input. (Refer to the system man page for details.)

4.1 Configuring an Intel® Edison device as a Wi-Fi client

After successfully booting your Intel® Edison device, enter the following:

```
$ configure_edison --wifi
```

This command will scan for available networks and ask you which network you want to connect to. Enter the Wi-Fi AP number for the appropriate network (Figure 4). If your network does require a password or other information, enter the appropriate credentials. If the connection fails, you may have typed in your credentials incorrectly. Try again by typing configure_edison --wifi and repeat the procedure above.

Figure 4 configure_edison --wifi

![Configure Edision --wifi](image)
Once you connect to the Wi-Fi network, if your computer is connected to the same network, you can open up a browser and type in the IP address or name of your Intel® Edison device. For example, http://192.168.1.67. Visiting this address should open up a device information webpage (Figure 5).

**Figure 5 Device information webpage**

![Device Information](image)

- Hostname: myedison
- IP Address: 10.0.0.16

### 4.2 wpa_supplicant

The `wpa_supplicant` daemon is a minimal-code open-source implementation of IEEE 802.11i supplicant for Linux®. It runs in the background as the backend component controlling a wireless connection, implements key negotiation with a WPA authenticator, and controls roaming and IEEE 802.11 authentication/association of the WLAN driver.

By default, `wpa_supplicant` does not start at boot time. To have it start at boot time automatically, based on the last connection saved with `wpa_cli`, enter the following:

```bash
$ systemctl enable wpa_supplicant
$ systemctl disable wpa_supplicant (to remove the startup at boot time)
```

Use the following commands to start/stop `wpa_supplicant` manually:

```bash
$ systemctl start wpa_supplicant
$ systemctl stop wpa_supplicant
```

Once `wpa_supplicant` has started, you can use `wpa_cli` to query current status, scan the network, select security mode, and connect to AP.

To verify that `wpa_supplicant` has started correctly, enter the following (see Figure 6):

```bash
$ systemctl status wpa_supplicant
```

**Figure 6 wpa_supplicant status**

![Status](image)

Because `connman` could automatically reconnect with saved connection parameters, when using `wpa_cli` to connect Wi-Fi, disable `connman` to avoid any connection conflicts. To stop `connman`, enter the following:

```bash
$ systemctl stop connman
```
4.2.1 Scan available networks

To display the list of configured networks, enter the following:

```
$ wpa_cli –i wlan0 scan
$ wpa_cli –i wlan0 scan_results
```

Figure 7 Scan result

<table>
<thead>
<tr>
<th>BSSID / Frequency / Signal Level / Flags / SSSID</th>
</tr>
</thead>
<tbody>
<tr>
<td>18:33:9d:8c:ae:ef 5220 -60 [WPA2-EAP-CCMP][ESS] RSNofficeWLAN</td>
</tr>
<tr>
<td>10:6f:3f:87:2c:be 2462 -56 [WPA-EAP-CCMP][ESS] qoesetup</td>
</tr>
<tr>
<td>c4:01:7c:6e:5f:3f 5590 -70 [WPA2-PSK-CCMP][ESS]</td>
</tr>
<tr>
<td>18:33:9d:a1:0a:b5 2412 -60 [WPA2-EAP-CCMP][ESS] EmployeeHotspot</td>
</tr>
<tr>
<td>72:ee:2e:36:4e:60 2437 -65 [WPA2-PSK-CCMP][ESS] FREEBOX_CELED</td>
</tr>
<tr>
<td>18:33:9d:a1:0a:ba 5260 -68 [WPA2-EAP-CCMP][ESS] EmployeeHotspot</td>
</tr>
<tr>
<td>18:33:9d:a1:0a:be 2437 -67 [WPA2-PSK-CCMP][ESS] FreeWiFi_secure</td>
</tr>
<tr>
<td>18:33:9d:a1:0a:be 5260 -69 [WPA-EAP-CCMP][ESS] TSONofficeWLAN</td>
</tr>
<tr>
<td>18:33:9d:a1:0a:bf 5200 -71 [WPA2-EAP-CCMP][ESS] RSNofficeWLAN</td>
</tr>
<tr>
<td>18:33:9d:a1:0b:ce 5180 -71 [WPA-EAP-TKIP][WPA2-EAP-CCMP][ESS] TSONofficeWLAN</td>
</tr>
<tr>
<td>18:33:9d:a1:0b:ca 5180 -71 [WPA2-EAP-CCMP][ESS] EmployeeHotspot</td>
</tr>
<tr>
<td>18:33:9d:a1:0b:cf 5180 -71 [WPA2-EAP-CCMP][ESS] RSNofficeWLAN</td>
</tr>
<tr>
<td>18:33:9d:a1:0a:fa 5200 -75 [WPA2-EAP-CCMP][ESS] EmployeeHotspot</td>
</tr>
<tr>
<td>18:33:9d:a1:0a:fe 5200 -72 [WPA-EAP-TKIP][WPA2-EAP-CCMP][ESS] TSONofficeWLAN</td>
</tr>
<tr>
<td>18:33:9d:a1:0a:ff 5200 -78 [WPA2-EAP-CCMP][ESS] RSNofficeWLAN</td>
</tr>
<tr>
<td>7c:95:f3:00:6d:2b 5180 -81 [WPA2-PSK-CCMP][ESS] ROAMING_HS26</td>
</tr>
<tr>
<td>7c:95:f3:00:6d:2c 5180 -82 [WPA2-PSK-CCMP][ESS] ROAMING_PSX</td>
</tr>
<tr>
<td>7c:95:f3:00:6d:27 5180 -83 [WPA2-PSK-CCMP][ESS] ROAMING_5G</td>
</tr>
<tr>
<td>10:6f:3f:06:6f:8c 2412 -54 [WPS][ESS] Buffalo soldier, dreadlock rasta</td>
</tr>
<tr>
<td>72:ee:2e:36:4e:d1 2437 -66 [ESS] FreeWiFi</td>
</tr>
<tr>
<td>10:6f:3f:87:2c:8f 5240 -67 [WPA2-PSK-CCMP][ESS] qoesetup</td>
</tr>
<tr>
<td>c8:eb:cb:3e:5d:3f 2462 -65 [WPS][ESS] Linksys-2.4GHz</td>
</tr>
</tbody>
</table>
| root@edison:~

4.2.2 Connect to a protected AP

You will find a script connecting secure access point in section 9.2. You must run the script from the target (the Intel® Edison shell).

```
$ sh wpacli_connect_wpa2.sh <SSID> <Password>
```

To push files into the target, enter the following from your host terminal:

```
$ scp wpacli_connect_wpa2.sh root@192.168.2.15
```

In this example, the network SSID name is AndroidHotspot and the password for the network is 0123456789 (Figure 8).
Verify that you are correctly connected with the `status` command. The line `wpa_state = COMPLETED` in Figure 9 shows you are connected to the AP.

### 4.2.3 Connect to an open AP

Section 9.1 details a WPA2-PSK script for connecting via secure access. You must run the script from the target.

```
$ sh wpacli_connect_open.sh <SSID>
```

Verify that you are correctly connected with the following command:

```
$ wpa_cli status
```
Setting up Wi-Fi connections

The line \texttt{wpa\_state = COMPLETED} in Figure 11 shows you are connected to the AP.

![Figure 11](image1.png)

\textbf{Figure 11} \hspace{1cm} \texttt{wpa\_cli status}

```
root@edison:# wpa\_cli status
Selected interface 'wlan0'
BSSID=0:10:00:10:00:00
SSID=Linksys-2.4GHz
Mode=station
Pairwise_cipher=None
Group_cipher=None
Key_mgmt=None
wpa\_state=COMPLETED
IP_address=192.168.1.132
P2P_device_address=fe:00:00:10:00:00
Root Ed son: #
```

### 4.2.4 Create/connect to an ad-hoc network

You can use an ad-hoc network to create a wireless network or join an existing \textbf{independent basic service set} (IBSS) network without involving a master AP. To do so, do the following:

1. Run the \texttt{wpacli\_ibss\_open.sh} script listed in section 9.4 from the shell on the first Intel® Edison device to create an ad-hoc network with "Edison" as its SSID name (Figure 12).

![Figure 12](image2.png)

\textbf{Figure 12} \hspace{1cm} Creating an ad-hoc network

```
root@edison:# sh wpacli_ibss_open.sh Edison
OK
```

2. On the second Intel® Edison device, scan and list the ad-hoc interfaces (Figure 13).

![Figure 13](image3.png)

\textbf{Figure 13} \hspace{1cm} Scanning for an ad-hoc network

```
root@edison:# wpa\_cli scan
Selected interface 'wlan0'
OK
root@edison:# wpa\_cli scan\_results | grep IBSS
aa:bb:ab:a7:73 2412 .-52 [IBSS] Edison
root@edison: #
```

3. From the second Intel® Edison device, connect to the newly created ad-hoc network by running the script \texttt{wpacli\_ibss\_open.sh} in section 9.4. (See Figure 14.)
4. When you create an IBSS network or join an existing one, the DHCP client and DHCP server are not started, so you must manually assign the IP address of the wlan0 interface. You can statically configure the IP address on both of the Intel® Edison devices using the `ifconfig wlan0 <ip_address>` command (Figure 17).

5. Once the connection is established and an IP address is assigned to the peered devices, verify the connection using the `ping` command on both devices.
Setting up Wi-Fi connections

4.2.5 Display wpa_supplicant traces

Because `wpa_supplicant` starts with a `systemd` service, `wpa_supplicant` traces display in the journal. You can retrieve these traces with the following command:

```
$ journalctl -f -u wpa_supplicant
```

By default, the level of `wpa_supplicant` of traces is set to INFO. You can check this with the following command:

```
$ wpa_cli log_level
```

You can increase the verbosity of `wpa_supplicant` with the following command:

```
$ wpa_cli log_level DEBUG
```

If you start `wpa_cli` without parameters, `wpa_cli` displays the events sent by `wpa_supplicant`. These events can help you troubleshoot problems.

**Note:** Because `connman` uses `wpa_supplicant`, the information above also applies when using `connman`.

4.3 Connman

`Connman` is a connection manager daemon that lets you manage network connections over Wi-Fi and designed for use with embedded devices. We recommend using `connman` release 1.25 or above. To check the release version, enter the following:

```
$ connmanctl -v
```

4.3.1 Starting connman

Because `wpa_supplicant` could automatically reconnect with `wpa_cli`-saved connection parameters, when using `connman` to connect Wi-Fi, before starting `connman`, use the following commands to remove all networks that would have been configured or saved with `wpa_cli`:

```
$ wpa_cli -i wlan0 remove_network all
$ wpa_cli save
```

Use the following commands to start/stop `connman`:

```
$ systemctl start connman
$ systemctl stop connman
```

When the `connman` service starts, `wpa_supplicant` automatically starts. By default, `connman` does not start at boot time. To start it at boot time automatically, based on the last connection, enter the following:

```
$ systemctl enable connman
$ systemctl disable connman (to remove the startup at boot time)
```

`Connman` has a standard command line client `connmanctl`. It can run in two modes:

- **In command mode, commands are entered as arguments to the `connmanctl` command, just like `systemctl`.
- Interactive mode is started by typing `connmanctl` without arguments. The prompt will change to `connmanctl>` to indicate it is waiting for user commands.
Setting up Wi-Fi connections

4.3.2 Scan available networks

Enter following connman commands:

$ connmanctl enable wifi  
$ connmanctl scan wifi

If you encounter an error when you run the scan wifi command, enter the following commands as a workaround, then later scan the Wi-Fi networks:

$ rfkill unblock wlan  
$ systemctl start wpa_supplicant  
$ systemctl start connman

To list the available networks found after a scan, enter the following:

$ connmanctl services

Figure 19 shows typical output from a network scan.
### Connect to an open AP

To connect to an open network, use the `connmanctl connect` command and enter the full "wifi_" name. To use an example from Figure 19:

```
$ connmanctl connect wifi_000000000000_716f657365747570_managed_psk
```

Verify that you are connected to the network by entering the following:

```
$ connmanctl state
```
4.3.4 Connect to a protected AP

For protected access points, you will need to provide some information to the connman daemon—at the very least a password or a passphrase. The commands in this section show how to run connmanctl in interactive mode, which is required for running the agent command. To start interactive mode, enter the following:

$ connmanctl

To scan for available Wi-Fi networks, enter the following:

connmanctl> scan wifi

To list services, enter the following:

connmanctl> services

To register the agent to handle user requests, enter the following:

connmanctl> agent on

To connect to one of the protected services, enter the following:

connmanctl> connect <service>

...where <service> is not the name of SSID but the full “wifi_” name, as shown in Figure 19. You might have to enter a passphrase if the selected Wi-Fi network requires authentication. For a WPS connection, enter an empty string for the passphrase so the system can then ask you to enter the WPS PIN or WPS PBC.

4.3.5 Display connman traces

Since connman starts with a systemd service, connman traces will display in the journal. You can retrieve these traces with the following command:

$ journalctl -f -u connman
5 Access Point Setup

5.1 Software-enable access point setup

Configure your device for Access Point mode by following the steps outlined here:

- https://communities.intel.com/docs/DOC-23137

SoftAP is supported through hostapd with the following configuration files for AP mode:

- /etc/hostapd/hostapd.conf. This is the main configuration file, which you will need to edit to set up a SoftAP. The SSID is in the form EDISON-XX-XX (where the Xs are hexadecimal digits), and the passphrase is the serial number of the Intel® Edison compute module, both of which appear on the box label your Intel® Edison compute module shipped in. Verify the SSID and passphrase by comparing them to the values for ssid and wpa_passphrase in the hostapd.conf file.

- /etc/hostapd/udhcpd-for-hostapd.conf. This is the configuration file for the DHCP server.

We recommend using the default hostapd.conf file, which comes with the Intel® Edison image package and is well documented with comments. You can edit and modify this file to suit your requirements. For example, instead of using the default hostapd.conf, you could use a minimal configuration setting with just the following:

```
interface=wlan0
driver=nl80211
ssid=test
channel=1
```

Start hostapd. Use the following to start/stop hostapd:

```
$ systemctl start hostapd
$ systemctl stop hostapd
```

To download the hostapd configuration file, visit this website: http://w1.fi/cgit/hostap/plain/hostapd/hostapd.conf.

**Note:** Because the Broadcom module does not support STA/AP concurrently, hostapd and wpa_supplicant cannot run simultaneously.

Use the following commands to start/stop wpa_supplicant manually:

```
$ systemctl start wpa_supplicant
$ systemctl stop wpa_supplicant
```
5.2 Software-enable access point setup using hidden SSID

We can configure an Intel® Edison device in AP mode with hidden SSID by setting the field `ignore_broadcast_ssid` to 1 (by default, the value is 0) in file `/etc/hostapd/hostapd.conf`. This example uses two Intel® Edison devices: the first acting as an AP with hidden SSID, and the second acting as an STA.

On the first Intel® Edison device:

1. Obtain the first Intel® Edison compute module's SSID and passphrase. Open the `/etc/hostapd/hostapd.conf` file and note the `ssid` and `wpa_passphrase` fields. In this example, the SSID is `EDISON-78-EC`, and the passphrase is `FZED433D0012G501` (Figure 20).

2. Change the value of `ignore_broadcast_ssid` field to 1. Edit the `/etc/hostapd/hostapd.conf` file, set the field to 1, and save the file.

3. To configure the Intel® Edison device as an access point, start `hostapd` (Figure 22).

4. Once `hostapd` starts successfully, you can see the IP address assigned to the wlan0 interface.
Access Point Setup

At this point, the first Intel® Edison device is acting as an AP; however, its SSID is hidden.

On the second Intel® Edison device:

1. From the second Intel® Edison device, connect as an STA using wpa_supplicant. Check the status of wpa_supplicant and start it if it is not started already (Figure 24).

2. To display the list of configured networks, enter the following:

   ```
   $ wpa_cli -i wlan0 scan
   $ wpa_cli -i wlan0 scan_results
   ```

   Because the SSID of the first Intel® Edison device is hidden, it will not show in the discovered lists.

3. Section 9.2 features a script that lets you connect to a secure access point. Run the script from the target (the Intel® Edison shell).

   ```
   $ wpacli_connect_wpa2.sh <ssid> <password>
   ```

   ...where ssid and passphrase are the network ID and password of the first Intel® Edison device.
4. Check the status to see the connection was successful (Figure 26).

5. The line `wpa_state = COMPLETED` indicates that the connection is complete. Ping the first Intel® Edison device (in AP mode) to check the connection.
6 Wi-Fi Direct

Wi-Fi Direct, also known as Wi-Fi Peer to Peer (P2P), allows Wi-Fi capable devices to connect directly to each other without a Wi-Fi hotspot. Only one of the participating devices must comply with Wi-Fi Direct to establish a peer-to-peer connection. In theory, devices having Wi-Fi Direct establish their own peer-to-peer networks with the WPA2-PSK security features. Wi-Fi Direct is much faster than Bluetooth® and allows devices to communicate or transfer data to each other across greater distances.

To set up Wi-Fi Direct on the Intel® Edison platform, we manage P2P connections with `wpa_supplicant`, which runs as a daemon and manages wireless communication.

Table 3 lists some basic P2P commands.

<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>p2p_find [timeout] [type=*]</td>
<td>Find/search P2P devices for up to [timeout] seconds.</td>
</tr>
<tr>
<td>p2p_stop_find</td>
<td>Stop P2P devices search.</td>
</tr>
<tr>
<td>p2p_connect &lt;addr&gt; &lt;&quot;pbc&quot;</td>
<td>PIN&gt; [ht40]</td>
</tr>
<tr>
<td>p2p_listen [timeout]</td>
<td>Listen for P2P devices for up to [timeout] seconds.</td>
</tr>
<tr>
<td>p2p_group_remove &lt;ifname&gt;</td>
<td>Remove P2P group interface. (Terminate group if GO.)</td>
</tr>
<tr>
<td>p2p_group_add [ht40]</td>
<td>Add a new P2P group. (Local end as GO.)</td>
</tr>
<tr>
<td>p2p_prov_disc &lt;addr&gt; &lt;method&gt;</td>
<td>Request provisioning discovery.</td>
</tr>
<tr>
<td>p2p_invite &lt;cmd&gt; [peer=addr]</td>
<td>Invite peer.</td>
</tr>
<tr>
<td>p2p_peers [discovered]</td>
<td>List known (optionally, only fully discovered) P2P peers.</td>
</tr>
<tr>
<td>p2p_peer &lt;address&gt;</td>
<td>Show information about known P2P peer.</td>
</tr>
<tr>
<td>p2p_set &lt;field&gt; &lt;value&gt;</td>
<td>Set a P2P parameter.</td>
</tr>
<tr>
<td>p2p_flush</td>
<td>Flush P2P state.</td>
</tr>
<tr>
<td>p2p_cancel</td>
<td>Cancel P2P group formation.</td>
</tr>
<tr>
<td>p2p_remove_client &lt;address</td>
<td>iface=address&gt;</td>
</tr>
</tbody>
</table>

6.1 Known limitations

Because of regulatory requirements, the Intel® Edison platform currently only supports P2P at 2.4 GHz.

6.2 Using Wi-Fi Direct

We can use Wi-Fi Direct to connect between two Intel® Edison devices (or between an Intel® Edison device and an Android® device) as peer devices. We can form P2P connections using these modes:

- PBC (push button control)
- PIN (pin mode)
6.3 Basic peer-to-peer setup

To set up a basic peer-to-peer connection, do the following:

1. Check the status of wpa_supplicant on both Intel® Edison devices. By default, wpa_supplicant does not start at boot time.

   ```
   root@edison:~# systemctl status wpa_supplicant
   ● wpa_supplicant.service - WPA supplicant service
     Loaded: loaded (/lib/systemd/system/wpa_supplicant.service; disabled)
     Active: inactive (dead)
   root@edison:~#
   ```

   If the state is inactive, start wpa_supplicant on both of the Intel® Edison devices:

   ```
   root@edison:~# systemctl start wpa_supplicant
   root@edison:~# systemctl status wpa_supplicant
   ● wpa_supplicant.service - WPA supplicant service
     Loaded: loaded (/lib/systemd/system/wpa_supplicant.service; disabled)
     Active: active (running) since Fri 2015-01-02 19:08:27 UTC, 2s ago
       Process: 278 ExecStartPost=/bin/systemctl start wpa_supplicant_p2p_event (code=exited, status=0/SUCCESS)
       Process: 275 ExecStartPost=/bin/systemctl start wpa_supplicant_wlan0_event (code=exited, status=0/SUCCESS)
     MAIN PID: 274 (wpa_supplicant)
       CGroup: /system.slice/wpa_supplicant.service
       /usr/sbin/wpa_supplicant -u -c/etc/wpa_supplicant/wpa...
   Jan 02 19:08:27 edison systemd[1]: Started WPA supplicant service.
   root@edison:~#
   ```

2. If the state is inactive, start wpa_supplicant on both of the Intel® Edison devices:

   ```
   root@edison:~# systemctl start wpa_supplicant
   root@edison:~# systemctl status wpa_supplicant
   ● wpa_supplicant.service - WPA supplicant service
     Loaded: loaded (/lib/systemd/system/wpa_supplicant.service; disabled)
     Active: active (running) since Fri 2015-01-02 19:08:27 UTC, 2s ago
       Process: 278 ExecStartPost=/bin/systemctl start wpa_supplicant_p2p_event (code=exited, status=0/SUCCESS)
       Process: 275 ExecStartPost=/bin/systemctl start wpa_supplicant_wlan0_event (code=exited, status=0/SUCCESS)
     MAIN PID: 274 (wpa_supplicant)
       CGroup: /system.slice/wpa_supplicant.service
       /usr/sbin/wpa_supplicant -u -c/etc/wpa_supplicant/wpa...
   Jan 02 19:08:27 edison systemd[1]: Started WPA supplicant service.
   ```

3. Check Wi-Fi Direct activation on both of the Intel® Edison devices: Activation of the P2P on the Intel® Edison device can be confirmed with the activation of the interface “p2p-dev-wlan0”.

   Use the following command to list all wireless interfaces:

   ```
   root@edison:~# wpa_cli -iwlan0 interface
   Available interfaces:
p2p-dev-wlan0
   wlan0
   root@edison:~#
   ```

4. Start discovery of the P2P device on both of the Intel® Edison devices:

   ```
   root@edison:~# wpa_cli -ip2p-dev-wlan0 p2p_find
   OK
   root@edison:~#
   ```

5. Use the p2p_peers command to list and obtain the MAC address of the discovered peer devices on both of the Intel® Edison devices.

   First Intel® Edison device:

   ```
   root@edison:~# wpa_cli -ip2p-dev-wlan0 p2p_peers
   7a:4b:87:a3:45:0d
   root@edison:~#
   ```
Second Intel® Edison device:

```
root@edison:~# wpa_cli -ip2p-dev-wlan0 p2p_peers
fe:c2:de:3a:78:ec
root@edison:~#
```

**Note:** Because the `p2p_find` command can take a few seconds before it finds a peer device, the `p2p_peers` command may return an empty value. If it does, wait a few seconds and enter the command again.

6. Optionally, you could list the detailed information of the peers, using their discovered MAC addresses:

```
root@edison:~# wpa_cli -ip2p-dev-wlan0 p2p_peer 7a:4b:87:a3:45:0d
7a:4b:87:a3:45:0d
  pri_dev_type=0-00000000-0
device_name=Edison
  manufacturer=Intel
model_number=Edison
serial_number=
config_methods=0x188
dev_capab=0x25
group_capab=0x0
level=0
persistent=0
age=336
listen_freq=2437
wps_method=not-ready
interface_addr=7a:4b:87:a3:c5:0d
member_in_go_dev=00:00:00:00:00
member_in_go_iface =00:00:00:00:00:00
go_neg_req_sent=0
go_state=local
dialog_token=1
intended_addr=7a:4b:87:a3:c5:0d
country=XX
oper_freq=2412
req_config_methods=0x0
flags=[REPORTED][PREFER_PERSISTENT_GROUP]
status=0
wait_count=9
invitation_reqs=0
oper_ssid=DIRECT-3Y-Edison
root@edison:~#
```
6.4 P2P connection with PIN mode as autonomous GO

In this example, we will configure the first Intel® Edison device as a Group Owner and form a P2P group with the discovered peer using PIN mode. To do so, perform the procedure in section 6.3 to set up a basic peer-to-peer connection, then do the following:

1. Manually configure the first Intel® Edison device as a group owner (GO) with the following command. This command will create a new “autonomous GO” interface with the format p2p-wlan0-x (where x is an integer).

   ```
   root@edison:~# wpa_cli -ip2p-dev-wlan0 p2p_group_add
   OK
   ```

2. You can list the new interface, along with existing interfaces, with the following command:

   ```
   root@edison:~# wpa_cli -ip2p-dev-wlan0 interface
   Available interfaces:
   p2p-wlan0-0
   p2p-dev-wlan0
   wlan0
   root@edison:~#
   ```

3. Enter the `wpa_cli -ip2p-wlan0-x wps_pin any` command, where x is an integer. This will return a PIN, an eight-digit number that you will later use to enter in the connection command for the second Intel® Edison device (the client). The `any` parameter allows any station to use the PIN. The PIN returned in the example below is 25889055.

   ```
   root@edison:~# wpa_cli -ip2p-wlan0-0 wps_pin any
   25889055
   root@edison:~#  
   ```

4. On the second Intel® Edison device, enter the following command to connect with the first Intel® Edison device with the PIN from above. The `join` parameter indicates that this is a command to join an existing group as a client.

   ```
   root@edison:~# wpa_cli -ip2p-dev-wlan0 p2p_connect fe:c2:de:3a:78:ec
   25889055 join
   OK
   root@edison:~#
   ```

*Note:* When a P2P connection is established, the GO has to enable a DHCP server, in order to assign IP addresses to CLIs. It does so with a service named `wpa_cli_event`, which is called by `wpa_supplicant` when the wireless connection is activated.
5. Query the IP address of the p2p-wlan0-x interface with `ifconfig p2p-wlan0-x`:

First Intel® Edison device:

```
root@edison:~# ifconfig p2p-wlan0-0
p2p-wlan0-0 Link encap:Ethernet HWaddr fe:c2:de:3a:f8:ec
    inet addr:192.168.42.1  Bcast:192.168.42.255  Mask:255.255.255.0
    inet6 addr: fe80::fcc2:deff:fe3a:f8ec/64 Scope:Link
    UP BROADCAST RUNNING MULTICAST  MTU:1500 Metric:1
    RX packets:33 errors:0 dropped:0 overruns:0 frame:0
    TX packets:33 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:1000
    RX bytes:4376 (4.2 KiB)  TX bbytes:5061 (4.9 KiB)
root@edison:~#
```

Second Intel® Edison device:

```
root@edison:~# ifconfig p2p-wlan0-0
p2p-wlan0-0 Link encap:Ethernet HWaddr 7a:4b:87:a3:c5:0d
    inet addr:192.168.42.20  Bcast:0.0.0.0  Mask:255.255.255.0
    UP BROADCAST RUNNING MULTICAST  MTU:1500 Metric:1
    RX packets:17 errors:0 dropped:0 overruns:0 frame:0
    TX packets:32 errors:0 dropped:0 overruns:0 carrier:0
    collisions:0 txqueuelen:1000
    RX bytes:2549 (2.4 KiB)  TX bbytes:4896 (4.7 KiB)
root@edison:~#
```

6. Optionally, you can query the “status” as well.

First Intel® Edison device:

```
root@edison:~# wpa_cli -ip2p-wlan0-0 status
    bssid=fe:c2:de:3a:f8:ec
    ssid=DIRECT-xf-Edison
    id=0
    mode=P2P GO
    pairwise_cipher=CCMP
    group_cipher=CCMP
    key_mgmt=WPA2-PSK
    wpa_state=COMPLETED
    ip_address=192.168.42.1
    p2p_device_address=fe:c2:de:3a:78:ec
    address=fe:c2:de:3a:f8:ec
root@edison:~#
```
Second Intel® Edison device:

```
root@edison:~# wpa_cli -ip2p-wlan0-0 status
bssid=fe:c2:de:3a:f8:ec
ssid=DIRECT-TS-Edison
id=0
mode=station
  pairwise_cipher=CCMP
  group_cipher=CCMP
  key_mgmt=WPA2-PSK
  wpa_state=COMPLETED
  ip_address=192.168.42.21
  p2p_device_address=7a:4b:87:a3:45:0d
  address=7a:4b:87:a3:45:0d
root@edison:~#
```

7. Verify the connection between the devices by cross-pinging them. From the command prompt of the first Intel® Edison device, ping the IP address of the second Intel® Edison device, and vice versa.

First Intel® Edison device:

```
root@edison:~# ping -c 5 192.168.42.20
PING 192.168.42.20 (192.168.42.20): 56 data bytes
64 bytes from 192.168.42.20: seq=0 ttl=64 time=96.037 ms
64 bytes from 192.168.42.20: seq=1 ttl=64 time=116.956 ms
64 bytes from 192.168.42.20: seq=2 ttl=64 time=144.259 ms
64 bytes from 192.168.42.20: seq=3 ttl=64 time=163.489 ms
64 bytes from 192.168.42.20: seq=4 ttl=64 time=194.749 ms
--- 192.168.42.20 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 96.037/143.098/194.749 ms
root@edison:~#
```

Second Intel® Edison device:

```
root@edison:~# ping -c 5 192.168.42.1
PING 192.168.42.1 (192.168.42.1): 56 data bytes
64 bytes from 192.168.42.1: seq=0 ttl=64 time=31.709 ms
64 bytes from 192.168.42.1: seq=1 ttl=64 time=30.777 ms
64 bytes from 192.168.42.1: seq=2 ttl=64 time=29.662 ms
64 bytes from 192.168.42.1: seq=3 ttl=64 time=39.124 ms
64 bytes from 192.168.42.1: seq=4 ttl=64 time=49.662 ms
--- 192.168.42.20 ping statistics ---
5 packets transmitted, 5 packets received, 0% packet loss
round-trip min/avg/max = 29.662/36.186/49.662 ms
root@edison:~#
```
6.5 P2P connection with PIN mode with group negotiation

Perform the procedure in section 6.3 to set up a basic peer-to-peer connection, then do the following:

1. Start the process of group negotiation without actually starting GO. The peer is expected to initiate a GO negotiation. On the first Intel® Edison device, enter the following `pin` command, which will return the PIN code that the peer device will use to connect. The PIN returned in the example below is 787062935.

   ```shell
   root@edison:~# wpa_cli -ip2p-dev-wlan0 p2p_connect 7a:4b:87:a3:45:0d
   pin auth
   78706293
   root@edison:~#  
   ```

2. On the second Intel® Edison device, enter the following command with the returned PIN to connect with the peer device:

   ```shell
   root@edison:~# wpa_cli -ip2p-dev-wlan0 p2p_connect fe:c2:de:3a:78:ec
   78706293
   OK
   root@edison:~#  
   ```

3. Once the connection is established, list the new interface, check the status of the interface, and cross-ping the devices (Figure 29 and Figure 30).

   **Figure 29** Verifying on the first Intel® Edison device

   ```shell
   root@edison:~# wpa_cli -ip2p-dev-wlan0 interface
   Available interfaces:
   p2p-wlan0-0
   p2p-dev-wlan0
   wlan0
   root@edison:~# wpa_cli -ip2p-dev-wlan0 status
   bssid=7a:4b:87:a3:45:0d
   asid=DIRECT-0C-Edison
   id=0
   mode=station
   pairwise_cipher=CCMP
   group_cipher=CCMP
   key_mgmt=WPA2-PSK
   wpa_state=COMPLETED
   ip address=192.168.42.20
   p2p_device_address=fe:c2:de:3a:78:ec
   address=fe:c2:de:3a:f0:ec
   root@edison:~# ping -c 3 192.168.42.1
   PING 192.168.42.1 (192.168.42.1): 56 data bytes
   64 bytes from 192.168.42.1: seq=0 ttl=64 time=30.692 ms
   64 bytes from 192.168.42.1: seq=1 ttl=64 time=29.766 ms
   64 bytes from 192.168.42.1: seq=2 ttl=64 time=37.712 ms
   --- 192.168.42.1 ping statistics ---
   3 packets transmitted, 3 packets received, 0% packet loss
   round-trip min/avg/max = 37.742/38.066/38.692 ms
   root@edison:~#  
   ```
6.6 P2P connection in PBC mode with first device as GO (with group negotiation)

Perform the procedure in section 6.3 to set up a basic peer-to-peer connection so that the peer devices discover each other, then do the following:

1. Start a P2P connection between the peer devices by passing the pbc parameter with the `p2p_connect` command. Use the optional `persistent` parameter if you want a persistent group to be formed.

   On the first Intel® Edison device, use a `go_intent` value of 15, which will make the DUT a GO:

   ```
   root@edison:~# wpa_cli -ip2p-dev-wlan0 p2p_connect 7a:4b:87:a3:45:0d
   pbc persistent go_intent=15
   OK
   root@edison:~#
   ```

   On the second Intel® Edison device, use a `go_intent` value of 0, which will make the DUT a client:

   ```
   root@edison:~# wpa_cli -ip2p-dev-wlan0 p2p_connect fe:c2:de:3a:78:ec
   pbc persistent go_intent=0
   OK
   root@edison:~#
   ```

2. Once the connection is established, list the new interface, check the status of the interface, and cross-ping the devices:

   ```
   wpa_cli -ip2p-dev-wlan0 interface
   wpa_cli -ip2p-wlan0-x status
   ping <p2p_ip_address>
   ```
6.7 Terminate and reinvoke a connection

You can terminate a P2P connection with the `p2p_group_remove` command, which removes the P2P group interface and disconnects the connection. (Use the network interface name of the group as an argument for this command.)

**Note:** As in earlier examples, the first Intel® Edison device is configured as GO (group owner), and the second Intel® Edison device is configured as CLI (client). You can reinvoke a previous connection, but only if you used the optional `persistent` connection parameter when you originally made the connection.

To terminate or reinvoke a connection, do the following:

1. On the first Intel® Edison device, enter the following command to break the connection and remove the group:

   ```
   root@edison:~# wpa_cli -ip2p-dev-wlan0 p2p_group_remove p2p-wlan0-0
   OK
   root@edison:~#
   ```

2. To reconnect, use either the `p2p_find` command or the `p2p_listen` command on the client (the second Intel® Edison device in this case):
   ```
   • wpa_cli -ip2p-dev-wlan0 p2p_find
   • wpa_cli -ip2p-dev-wlan0 p2p_listen
   ```

3. On the first Intel® Edison device, reinvoke the connection using `p2p_invite` by passing the MAC address of the discovered peer device.

   ```
   root@edison:~# wpa_cli -ip2p-dev-wlan0 p2p_invite persistent=0 peer=7a:4b:87:a3:45:0d
   OK
   root@edison:~#
   ```

4. Once the connection is established, list the new interface, check the status of the interface, and cross-ping the devices:
**Figure 31** Verifying on the first Intel® Edison device

```
root@edison:~$ wpa_cli -ip2p-dev-wlan0 interface
Available interfaces:
  p2p-wlan0-1
  p2p-dev-wlan0
  wlan0
root@edison:~$ wpa_cli -ip2p-dev-wlan0-1 status
isid=fe:c2:de:3a:78:ec
ssid=DIRECT-EX-Edison
id=0
mode=p2p_co
pairwise_cipher=CCMP
group_cipher=CCMP
key_mgmt=WPA2-PSK
wpa_state=COMPLETED
ip_address=192.168.42.1
p2p_device_address=fe:c2:de:3a:78:ec
addresses=fe:c2:de:3a:78:ec
root@edison:~$ ping -c 3 192.168.42.20
PING 192.168.42.20 (192.168.42.20): 56 data bytes
64 bytes from 192.168.42.20: seq=0 ttl=64 time=117.206 ms
64 bytes from 192.168.42.20: seq=1 ttl=64 time=134.315 ms
64 bytes from 192.168.42.20: seq=2 ttl=64 time=153.035 ms
--- 192.168.42.20 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 117.206/134.315/153.035 ms
root@edison:~$
``` 

**Figure 32** Verifying on the second Intel® Edison device

```
root@edison:~$ wpa_cli -ip2p-dev-wlan0 interface
Available interfaces:
  p2p-wlan0-1
  p2p-dev-wlan0
  wlan0
root@edison:~$ wpa_cli -ip2p-dev-wlan0-1 status
isid=fe:c2:de:3a:78:ec
ssid=DIRECT-FX-Edison
id=0
mode=station
pairwise_cipher=CCMP
group_cipher=CCMP
key_mgmt=WPA2-PSK
wpa_state=COMPLETED
ip_address=192.168.42.20
p2p_device_address=7a:4b:87:a3:45:0d
address=7a:4b:87:a3:45:0d
root@edison:~$ ping -c 3 192.168.42.1
PING 192.168.42.1 (192.168.42.1): 56 data bytes
64 bytes from 192.168.42.1: seq=0 ttl=64 time=50.768 ms
64 bytes from 192.168.42.1: seq=1 ttl=64 time=38.878 ms
64 bytes from 192.168.42.1: seq=2 ttl=64 time=37.948 ms
--- 192.168.42.1 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 37.948/42.495/50.768 ms
root@edison:~$
```
6.8 P2P connection with WPS-PBC mode as autonomous GO

In this example, we will configure the first Intel® Edison device as an autonomous group owner. To do so, perform the procedure in section 6.3 to set up a basic peer-to-peer connection so that the peer devices discover each other, then do the following:

1. Connect and form a P2P group with the discovered P2P peer using PBC mode.
2. Manually configure the first Intel® Edison device as a group owner (GO) with the following command, which will create a new "autonomous GO" interface with the format p2p-wlan0-x (where x is an integer):
   ```
   root@edison:~# wpa_cli -ip2p-dev-wlan0 p2p_group_add
   OK
   ```
3. You can display the newly created interface name with the wpa_cli -ip2p-dev-wlan0 interface command:
   ```
   root@edison:~# wpa_cli -ip2p-dev-wlan0 interface
   Available interfaces:
   p2p-wlan0-0
   p2p-dev-wlan0-0
   wlan0
   root@edison:~#
   ```
4. When the first Intel® Edison device has been configured as a GO, enter the following command, which starts the WPS PBC method and allows a single WPS enrollee to connect to the GO:
   ```
   root@edison:~# wpa_cli -ip2p-wlan0-0 wps_pbc
   OK
   ```
5. On the second Intel® Edison device, enter the following command to make a P2P connection. The join parameter indicates that this is a command to join an existing group as a client.
   ```
   root@edison:~# wpa_cli -ip2p-dev-wlan0 p2p_connect fe:c2:de:3a:78:ec
   join
   OK
   ```
6. Once the connection is established, list the new interface, check the status of the interface, and cross-ping the devices:
   ```
   wpa_cli -ip2p-dev-wlan0 interface
   wpa_cli -ip2p-wlan0-x status
   ping <p2p_ip_address>
   ```
6.9 P2P connection using PBC method

You can create a P2P connection using the PBC method if you want to connect several devices to the network and enable data encryption by pushing a button, either a physical button or a virtual one.

Figure 33 P2P connection between an Intel® Edison device and an Android* device using PBC method

1. Enable Wi-Fi Direct on an Android* device that supports the Wi-Fi Direct feature (Figure 34).

Figure 34 Enabling Wi-Fi Direct on an Android* device

2. Perform the procedure in section 6.3 to set up a basic peer-to-peer connection. On the Android* device, you will see “Edison” in Peer Devices (Figure 35).

Figure 35 Peer devices on an Android*
3. Optionally, you can list the detailed information of the discovered peers with this command:

```
root@edison:~# wpa_cli -ip2p-dev-wlan0 p2p_peer 02:1e:67:80:a8:3b
02:1e:67:80:a8:3b
pri_dev_type=10-0050F204-5
device_name=Android_de6e
manufacturer=Intel
model_name=merrifield
model_number=R4
serial_number=11111
config_methods=0x188
dev_capab=0x24
group_capab=0x0
level=-47
persistent=8
age=0
listen_freq=2437
wps_method=not-ready
interface_addr=00:00:00:00:00:00
member_in_go_dev=00:00:00:00:00:00
member_in_go_iface =00:00:00:00:00:00
go_neg_req_sent=no
go_state=unknown
dialog_token=0
intended_addr=00:00:00:00:00:00
country=0
oper_freq=0
req_config_methods=0x0
flags=[REPORTED]
status=0
wait_count=0
invitation_reqs=0
```

4. From the Intel® Edison device, connect with the peered Android* device:

```
wpa_cli -ip2p-dev-wlan0 p2p_connect <discovered_mac_address> pbc
persistent go_intent=<0...15>
```

When using autonegotiation, an Intel® Edison device can be forced to GO for a P2P connection by assigning a value of 15 to the connect parameter `go_intent`. (For `go_intent`, a value of 15 means the device under test (DUT) will become GO; a value of 0 means the DUT will become client.)

```
root@edison:~# wpa_cli -ip2p-dev-wlan0 p2p_connect 02:1e:67:80:a8:3b
pbc persistent go_intent=15
OK
root@edison:~#
```

On the Android* device, you will get a notification to connect to the Intel® Edison device (Figure 36).
5. After you accept the connection on the Android* device, you should see the Intel® Edison device as Connected (Figure 37).

6. Once the connection is established, list the new interface, check the status of the interface, and cross-ping the devices:

   ```
   wpa_cli -ip2p-dev-wlan0 interface
   wpa_cli -ip2p-wlan0-x status
   ping <p2p_ip_address>
   ```

7. Optionally, you can query the status with the `wpa_cli -ip2p-wlan0-x status` command (Figure 38).
6.9.1 Disconnecting and reconnecting

Disconnecting and reconnecting is quick and easy because once-connected Wi-Fi devices remember former P2P connections. To disconnect, do the following:

1. Use the `p2p_group_remove` command to terminate the connection between the Intel® Edison device and the Android* device.

   ```bash
   root@edison:~# wpa_cli -ip2p-dev-wlan0 p2p_group_remove p2p-wlan0-0
   ```

   When you enter this command, the connection breaks and the Intel® Edison device will appear under *Remembered groups* in your Android* device (Figure 39).

2. Reinvoke the connection by passing the peer device’s MAC address with the `p2p_invite` command.

   ```bash
   root@edison:~# wpa_cli -ip2p-dev-wlan0 p2p_invite persistent=0 peer=02:1e:67:80:a8:3b
   OK
   root@edison:~#
   ```

   Once the connection is reestablished, you can see the new interface on the Intel® Edison device using:

   ```bash
   root@edison:~# wpa_cli -ip2p-dev-wlan0 interface
   ```
Wi-Fi Direct

A “locked wireless” connection icon appears on the Android* device (Figure 40).

**Figure 40** Android* reconnected

6.10 Wi-Fi Direct troubleshooting

When two P2P devices find each other, they alternately listen and scan on channels 1, 6, and 11. If you have trouble finding a peer device or you want to find it faster, you can tell each device independently to listen on a specific channel with the following command:

```
wpa_cli -ip2p-dev-wlan0 p2p_set listen_channel x
```

...where x is the channel (1, 6, or 11). We recommend using channel 1.

Once the devices find each other and the connection is triggered, the peer devices negotiate a channel on which the P2P group starts. You can force the P2P group channel by adding `freq=<frequency>` to the `p2p_connect` command, where `<frequency>` is an integer in megahertz. For example:

```
go_intent=0 freq=2412
```

6.10.1 Cycling power

Because Wi-Fi-Direct is a complex technology, we recommend cycling power on all devices and retrying the Wi-Fi-Direct connection whenever you experience interoperability problems. You can power Intel® Edison devices off/on with these commands:

```
systemctl stop wpa_supplicant
modprobe -r bcm4334x
modprobe bcm4334x
systemctl start wpa_supplicant
```

You can enter all of these commands in interactive mode, by starting `wpa_cli`:

```
wpa_cli -i p2p-dev-wlan0
```
6.10.2 Wi-Fi multirole

The multirole feature is the capability to have simultaneously a station (STA) connection (to an AP) and a P2P connection (GO role). The BCM-43340 Wi-Fi driver supports this multirole feature.

To enable multirole from a test setup perspective, do the following:

1. Establish a P2P connection by following any of the methods described in this chapter.
2. Make an STA connection by following any of the methods described in this chapter.
   
   **Note:** Due to current software limitations, if you first make an STA connection and then try to establish a P2P connection, the P2P connection may fail.
3. When you have a Wi-Fi Direct connection and STA connection to an access point at the same time, you can list all wireless interfaces using the `wpa_cli -ip2p-dev-wlan0 interface` command (Figure 41).

4. Once the connection is established, list the new interface, check the status of the interface, and cross-ping the devices:

   ```
   wpa_cli -ip2p-dev-wlan0 interface
   wpa_cli -ip2p-wlan0-x status
   ping <p2p_ip_address>
   ```
7 Additional Configurations

This chapter explains a few miscellaneous configurations.

7.1 Disable power management

Disabling power management might be useful for some debug/performance reasons. You can turn power management off with the following command:

```
$ iwconfig wlan0 power off
```

To disable power management at boot time so that it's always disabled, create a file in the `/lib/systemd/system` folder named `wifi-power-management-off.service` with the content as shown in section 9.3.

To disable power management, use the following command:

```
$ systemctl start wifi-power-management-off
```

To verify that power management is disabled, use the following command:

```
$ iwconfig wlan0
```

You should see **Power Management:off**.

![Power Management status](image)

**Note:** You can only disable power management after the Intel® Edison device connects to AP; disabling power management before the Intel® Edison device connects to AP will have no effect.

7.2 DHCP client vs. static IP address

By default, the DHCP client dynamically assigns an IP address to the `wlan0` interface as soon as the Wi-Fi connection is made:

- When `connman` makes the Wi-Fi connection, it uses the built-in DHCP client.
- When `wpa_cli` makes the Wi-Fi connection, the `/etc/wpa_supplicant/wpa_cli-actions.sh` script calls and uses the DHCP client `udhcpc`.

When assigning a static IP address, disable the DHCP client to avoid conflicts:

- When `connman` makes the Wi-Fi connection, execute the following command before making the connection:
  
  ```
  connmanctl>config <service> --ipv4 off
  ```

- When `wpa_cli` makes the Wi-Fi connection, comment out the following line in the `/etc/wpa_supplicant/wpa_cli-actions.sh` script:
  
  ```
  udhcpc -I $IFNAME -p /var/run/udhcpc-$IFNAME.pid -S
  ```
Additional Configurations

When the DHCP client is disabled and the Wi-Fi connection is complete, you can assign the IP address with the following command:

```
ifconfig wlan0 <ip_address>
```

If you want the static address assigned after an autoconnection following a reboot or Wi-Fi reconnection, do the following:

- When `connman` makes the Wi-Fi connection, execute the following command before making the connection:
  
  `connmanctl>config <service> --ipv4 manual <ip_address> <netmask> <gateway>`

- When `wpa_cli` makes the Wi-Fi connection, add the following line in the `/etc/wpa_supplicant/wpa_cli-actions.sh` script:
  
  `ifconfig wlan0 <ip_address> (where the call to udhcpc has been made)`

### 7.3 Autoreconnect after a reboot

With `connman`, once the connection is made, the connection parameters are automatically saved in the specific `connman` folder. In this case, nothing is saved in `/etc/wpa_supplicant/wpa_supplicant.conf`.

With `wpa_cli`, by default, the connection parameters are not saved. To save them, you must enter the command:

```
$ wpa_cli -i wlan0 save
```

This adds a new network, along with its connection parameters, in `/etc/wpa_supplicant/wpa_supplicant.conf`. At the next reboot, `wpa_supplicant` reads the conf file and tries to connect to the "network" configured.

### 7.4 Ping latency

When pinging, mainly from AP device to STA device, you can observe a latency. This is related to the overall power management scheme to reduce the power consumption. Hence, when there is minimal data traffic, after some inactivity timeouts, the following events occur:

- The WLAN chipset automatically switches from active mode to passive mode. (This is related to the Wi-Fi power management mentioned above.)
- The SDIO host controller automatically powers down.
- The Intel® Edison CPU is automatically clock-gated.

So when a packet is received, there are some delays to restore the components in an active state for the reception of the packet. This delay is then a relevant part of the ping delay.

**Note:** Because the interval between two packets during normal data traffic is shorter than the smaller inactivity timeout, the hardware above the components remains in an “active” state, and there is no significant delay per packet.

§
8 Wi-Fi Software Components

8.1 Source folders

These are the Wi-Fi source folders:

- **broadcom_cws/wlan/driver_bcm43x**: Source files for the Broadcom Linux Driver.
- **wpa_supplicant / wpa_cli**: Source files for wpa_supplicant / wpa_cli 2.1
- **linux-kernel/net/wireless**: Source files for the full Mac model defined in mac80211

8.2 Binary files

These are the Wi-Fi configuration/binary files:

- **broadcom_cws/wlan/firmware**: Firmware and NVRAM files of Broadcom chipset 43340.

8.3 Recipe folders

These are the Wi-Fi recipe folders:

- **device-software/meta-edison-distro/recipes-connectivity/wpa_supplicant/wpa_supplicant**: Recipe folder of wpa_supplicant.
- **device-software/meta-edison/recipes-kernel/bcm43340**: Recipe folder for the Broadcom driver and FW/NVRAM.

8.4 Build commands

The `bitbake edison-image` command builds the overall image; you can also rebuild solitary recipes with the following:

```
bitbake <recipe> -c clean -f -v
bitbake <recipe> -c compile -f -v
```

Recipes include:

- **bcm43340-mod**: Broadcom driver module bcm4334x.ko
- **wpa-supplicant**: wpa_supplicant / wpa_cli

8.5 Files copied onto root of the Intel® Edison file system

The following binary/configuration files are copied onto root of the Intel® Edison file system:

- **/etc/firmware/fw_bcmdhd.bin**: Chipset firmware.
- **/etc/firmware/bcmdhd.cal**: Chipset NVRAM file for the Murata 43340 module on the Intel® Edison board.
- **/etc/modprobe.d/bcm4334x.conf**: Kernel module configuration file used when inserting the kernel driver with the command modprobe.
- **/lib/modules/3.10.17-poky-edison+/extra/bcm4334x.ko**: Driver module.
- **/etc/wpa_supplicant/wpa_supplicant.conf**: Configuration file for wpa_supplicant.
8.6 Driver modules

To list driver modules, enter following:

```
$ lsmod
```

To load the Broadcom driver module, enter the following:

```
$ modprobe bcm4334x
```

To remove the Broadcom driver module, enter the following:

```
$ modprobe -r bcm4334x
```

This powers down the chipset, if it is not already powered down.
The scripts in this chapter work on the Intel® Edison board from the /tty/USBx console or from an SSH connection. To execute these generic scripts through SSH, you must store them in the same folder on your computer that has the edison_env file.

9.1 Script to connect to an open AP

The wpacli_connect_open.sh script initially configures the wlan0 interface into a disconnected state and removes all networks from the list of configured networks. It also adds a network with the wlan0 interface and configures and sets network variables such SSID (as provided from the command line input) and some other attributes, such as authentication algorithm, key management protocol, and security protocol. It then selects and enables the network with Index 0.

The wpacli_connect_open.sh script prompts for one input: the SSID of the target Wi-Fi network.

Run the script from the shell command prompt on the Intel® Edison device:

```
if [ $# != 1 ] ; then
    echo "$0 <SSID>"
    exit
fi

wpa_cli -iwlan0 disconnect
wpa_cli -iwlan0 remove_network all
wpa_cli -iwlan0 add_network
wpa_cli -iwlan0 set_network 0 mode 0
wpa_cli -iwlan0 set_network 0 ssid "$1"
wpa_cli -iwlan0 set_network 0 auth_alg OPEN
wpa_cli -iwlan0 set_network 0 key_mgmt NONE
wpa_cli -iwlan0 set_network 0 scan_ssid 1
wpa_cli -iwlan0 select_network 0
wpa_cli -iwlan0 enable_network 0
wpa_cli -iwlan0 reassociate
wpa_cli -iwlan0 status
```
9.2 Script to connect to a secure AP (with WPA2-PSK)

The `wpacli_connect_wpa2.sh` script initially configures the wlan0 interface into a disconnected state and removes all networks from the list of configured networks. It also adds a network with the wlan0 interface, and configures and sets the network variables such as SSID, WPA passphrase (as provided from the command line), and some other attributes, such as authentication algorithm, key management protocol, and security protocol. It then selects and enables the network with Index 0.

The `wpacli_connect_wpa2.sh` script prompts for two inputs: the **SSID** and **password** of the target Wi-Fi network.

Run the script from the shell command prompt on the Intel® Edison device:

```bash
if [ $# != 2 ] ; then
    echo "$0 <SSID> <passphrase>"
    exit
fi

wpa_cli -iwlan0 disconnect
wpa_cli -iwlan0 remove_network all
wpa_cli -iwlan0 add_network
wpa_cli -iwlan0 set_network 0 mode 0
wpa_cli -iwlan0 set_network 0 ssid "$1"
wpa_cli -iwlan0 set_network 0 auth_alg OPEN
wpa_cli -iwlan0 set_network 0 key_mgmt WPA-PSK
wpa_cli -iwlan0 set_network 0 proto RSN
wpa_cli -iwlan0 set_network 0 psk "$2"
wpa_cli -iwlan0 set_network 0 scan_ssid 1
wpa_cli -iwlan0 select_network 0
wpa_cli -iwlan0 enable_network 0
wpa_cli -iwlan0 reassociate
wpa_cli -iwlan0 status
```

9.3 Script to disable power management

To disable power management at boot time so that it’s always disabled, you need to create a file with the name `wifi-power-management-off.service` and save it in the `/lib/systemd/system` folder.

The `wifi-power-management-off.service` file should look like this:

```ini
[Unit]
Description=Disable power management for wlan0
Requires=sys-subsystem-net-devices-wlan0.device
After=sys-subsystem-net-devices-wlan0.device

[Service]
Type=oneshot
ExecStart=/sbin/iwconfig wlan0 power off
```

The **systemd** service manager daemon will invoke this file on bootup, and the service will disable power management for the wlan0 interface.
9.4 Script to create/connect to an ad-hoc network

The wpacl_iibss_open.sh script lets you join an IBSS network or create one on channel 1, without security:

```bash
if [ $# != 1 ]; then
echo "$0 <SSID>"
exit
fi

$rmt wpa_cli -iwlan0 disconnect
$rmt wpa_cli -iwlan0 remove_network all
$rmt wpa_cli -iwlan0 add_network
$rmt wpa_cli -iwlan0 set_network 0 frequency 2412
$rmt wpa_cli -iwlan0 set_network 0 mode 1
if [ ! -n "$rmt" ]; then
$rmt wpa_cli -iwlan0 set_network 0 ssid "$1"
else
$rmt wpa_cli -iwlan0 set_network 0 ssid '"'$1'"'
fi
$rmt wpa_cli -iwlan0 set_network 0 auth_alg OPEN
$rmt wpa_cli -iwlan0 set_network 0 key_mgmt NONE
$rmt wpa_cli -iwlan0 set_network 0 scan_ssid 1
$rmt wpa_cli -iwlan0 select_network 0
$rmt wpa_cli -iwlan0 enable_network 0
$rmt wpa_cli -iwlan0 status
```