Setting up SX-SDMAC on an i.MX 6 SoloX Platform

Part Number 140-20135-100
Revision C

Copyright & Trademark

© 2017 Silex Technology America, Inc. All rights reserved. November, 2017

Silex Technology America SPECIFICALLY DISCLAIMS THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS OF THIS PRODUCT FOR A PARTICULAR PURPOSE. Silex shall not be liable for any errors contained in this manual or for any damages resulting from loss of use, data, profits, or any incidental or consequential damages arising from the use of SILEX products or services. The information contained in this documentation is subject to change without notice.

Information and descriptions contained herein are the property of Silex. Such information and descriptions may not be copied, disseminated, or distributed without the express written consent of Silex. This publication is subject to change without notice.

The software embedded in this evaluation image includes the Linux operating system. Linux and certain other software programs used in the SX-SDMAC evaluation image are licensed under GNU GPL compatible Free Software Licenses. In compliance with these licenses, you can obtain the relevant source code at no charge by contacting Silex at support@silexamerica.com.

Silex Technology America, Inc.
www.silexamerica.com

Table of Contents

Figures

Tables

1 Document Conventions

The following section describes the conventions used within the document to identify and highlight the different applications of displayed text and provide valuable contextual information related to a section or paragraph.

Note that any identified context provided in proximity to any text overrides the conventions listed below.

1.1 Text Conventions

Bold	Bold type within paragraph text indicates commands, file names, directory names, paths, output, or returned values.
<> or Italics	Within commands, italics indicate a variable that the user must specify.
`Courier`	The Courier font indicates output or display. Example: `Error:Unable to allocate memory for transfer!`
[ ]	Within commands, items enclosed in square brackets are optional parameters or values that the user can choose to specify or omit.
{ }	Within commands, items enclosed in braces are options from which the user must choose.
|	Within commands, the vertical bar separates options.
…	An ellipsis indicates a repetition of the preceding parameter.
>	The right angle bracket separates successive menu selections. Example: Applications > Accessories > Terminal

1.2 Notes

A note contains information that requires special attention or provides specific information relating to the section or paragraph displayed above it. The following highlighted icon will be used. The area next to the icon will identify the specific information and make any references necessary.

This is a note section that contains supplemental information required to fully understand the contents of the inline text.
Additionally, it will grow with the size of the contents.

1.3 Caution

A caution contains information that, if not followed, may cause permanent damage to the product, render the product inoperable or cause injury to the user. The following highlighted icon will be used. The area next to the icon will identify the specific information and make any references necessary.

All cautions MUST be read and understood.

This is a caution section that contains supplemental information required to fully understand the contents of the inline text.

Additionally, it will grow with the size of the contents.

2 Product Overview

This manual covers how to start the evaluation process of the SX-SDMAC, a dual-band 802.11a/b/g/n/ac plus Bluetooth4.1 SDIO combo module based on the QCA9377-3, with the NXP Sabre Smart Devices (Sabre-SD) evaluation kit. Our SX-SDCAC which is in a SD card form factor (using the SX-SDMAC) is built to work with the NXP i.MX 6SoloX Sabre SD board. We have created an evaluation image using the Silex reference driver for the i.MX platform. The evaluation image includes the Silex reference radio driver, supplicant and tools to test the following:

- Basic Wireless functionality
- Advanced Wi-Fi Security
- Data Throughput

Please contact your Silex sales representative to learn more about Silex Technology America, Inc. Embedded Wireless LAN solutions, including hardware options, production radio drivers with 802.1X authentication along with development and support services.

3 Contents of SX-SDCAC Sample Pack

The following items should be present in the SX-SDCAC Sample Pack:

1. SX-SDCAC radio card
2. One Unictron Antennas Part number AA258

If any of the above items are missing please contact Silex Technology America technical support immediately:

Tel (US): 866.765.8761
Tel (International): +1.801.748.1199
email: support@silexamerica.com

4 Preparing SD Card Boot Image

This section covers how to write an SD Card with the Silex Wireless LAN evaluation image for the i.MX6SoloX Sabre evaluation kit. The image is intended to provide a platform with which to evaluate our wireless modules in conjunction with the i.MX6SoloX processor.

The image and SD card will **only** work with the appropriate Silex wireless card and the i.mx6 Sabre-SD kit. The Sabre Smart Devices EVK part number referenced in this Users Guide is **MCIMX6SX-SDB.**

4.1 Downloading the EVK Image

This application note requires that a valid bootable OS image has been obtained from Silex Technology. It is available from the Silex website.

To obtain the image on the web site use the following instructions:

1) Go to https://www.silextechnology.com/silex-driver-evaluation-license-agreement
2) Under the "**What you will need?"** section, click on **"Evaluation Software"**.
3) You will be asked to **Accept** an evaluation agreement.
4) Complete the information form, click **Submit**.
5) A confirmation email will be sent to the email provided. Upon verification you will be provided a link to download the evaluation image.
6) Decompress the EVK image file to a location where it can be accessed for the SD card writing process.

4.2 SD Card Requirements

You will need a card supporting the following specification:

Capacity: **8GB minimum**
Type: **SDHC**
Class: **4** (4MB/s minimum data transfer speed)

Figure 1 - Example SD Card

4.3 Creating Bootable SD Card – Linux Host Platform

To create a bootable SD card we will use the **dd** command, from the Linux command console.

The format of the **dd** command is as follows:

```
dd if=[input file/device] of=[output file/device] [options]
```

To write the bootable card we will use the Silex EVK card image as the input file and the SD card device as the output file. We will use a specific option to help the writing of the SD card.

Firstly, you must have completed section 4.1 and downloaded the EVK SD card image to a known location. You will need to know the path and full name of the image (.img) file. Secondly you will need to establish the device name for the SD card you are going to write.

To do this you will need to be root:

```
sudo su
```

Next we need to find the name of the device in the /dev directory. To do this you should insert your blank SD card into the SD slot on your machine then enter the following:

```
dmesg | tail
```

This will pipe the device messages into the tail command. You should see one of the lines, probably the last one in the list, it will look something like:

```
[1345842.064126] sd 3:0:0:0: [sdb] Attached  SCSI removable disk
```

You will use the sdb or similar device name assigned to the SD card. To write the image to the card issue the following command:

```
dd if=~/<image file name> of=/dev/sdb bs=512
```

This command writes the Silex image file, stored in the home directory to the SD card, using a block size of 512 bytes. The process make take a while when complete the Linux command prompt will be displayed. You can then remove the card and follow the instructions in Section 5.

Note: **sdb** is an example of the SD card device name assigned by Linux. Different Linux machine may have different device name assignment. Specifing an incorrect SD card device name may results in data loss on unintended data storage devices.

4.4 Creating Bootable SD Card – Windows Host Platform

To create a bootable SD card using a windows machine you will need a compatible SD slot. An external USB SD card writer/Reader is recommended.

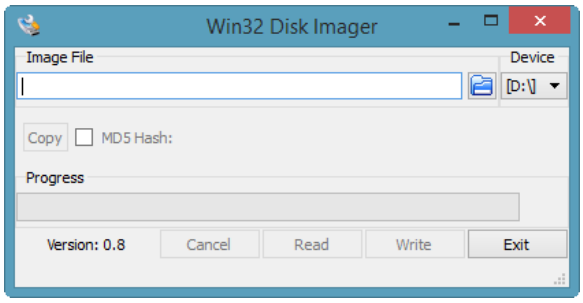
The following section is an example of how an SD card can be written using an application running on the Microsoft Windows OS. Silex has no affiliation with the developer of the mentioned application nor is responsible for issues using the application or bugs found. Use at your own risk.

An example of an application that can write to an SD card using the windows OS is Win32 Disk Imager. The application can be found for free on the sourceforge.net website. The following section will step through how to write an SD card using Win32 Disk Imager and the Silex EVK demo image.

To complete this section you must have first downloaded the EVK image (section 4.1) and installed the Win32 Disk Imager application.

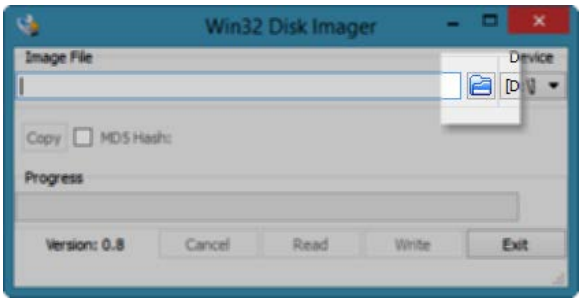
Firstly, run the Win32 Disk Imager application (as an administrator). You will see the following screen:

Figure 2 - Win32 Disk Image Initial Screen



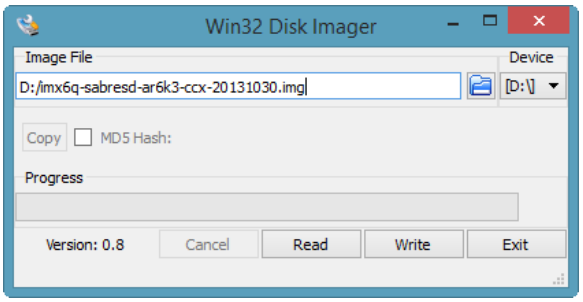
Next select the disk image file from the location it was stored. This is done by pressing the folder icon next to the **Image File** field:

Figure 3 - Win32 Disk Image Find Image File



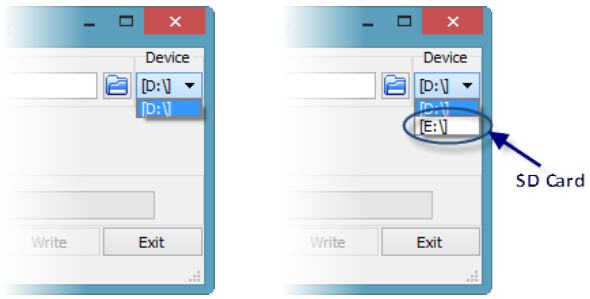
When the image file is selected it should look like the following:

Figure 4 - Win32 Disk Image Select Image File



Next you must select the SD card to write to. This is done using the **Device** selection. It is important to select the correct device as the writing process will destroy all data on the device selected. The best way to find out which device holds the SD card is to press the down arrow next to the device letter, note the displayed devices. Insert your SD card that you want to write to and then repeat the step above, the new device letter in the list indicates the assigned device for the SD card:

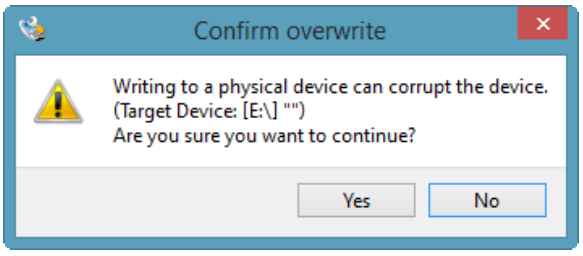
Figure 5 - Win32 Disk Device Selection



Once the image and SD card are selected you are ready to write the image to the card.

Press the **Write** button. You will see the following warning:

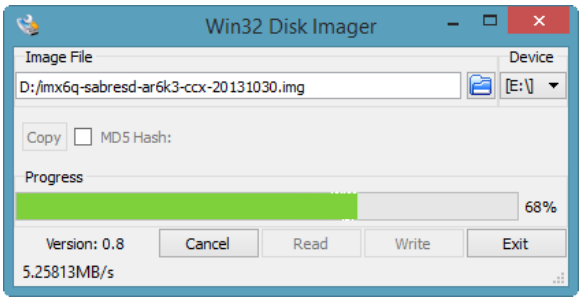
Figure 6 - Win32 Disk Image Confirm Writing



Press **Yes**.

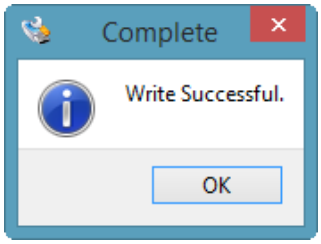
The Windows will display the progress bar and indicate the percentage complete:

Figure 7 - Win32 Disk Image Writing Image



When complete, the following message will be displayed:

Figure 8 - Win32 Disk Image Writing Complete



Press **OK**.

The SD card is now ready for use with the Silex EVK.

5 Installing the SX-SDCAC

Before you can install the Silex SX-SDCAC you must have purchased the NXP SABRE Board for Smart Devices based on the i.MX 6SoloX processor. It can be purchased directly from NXP or one of their authorized distributors.

The Sabre Smart Devices (Sabre-SD) EVK part number referenced in this Users Guide is **MCIMX6SX-SDB.**

The following steps assume:

1) You have already successfully verified the operation of the Sabre-SD EVK using the default Freescale SD card with the OS image that is shipped with it.

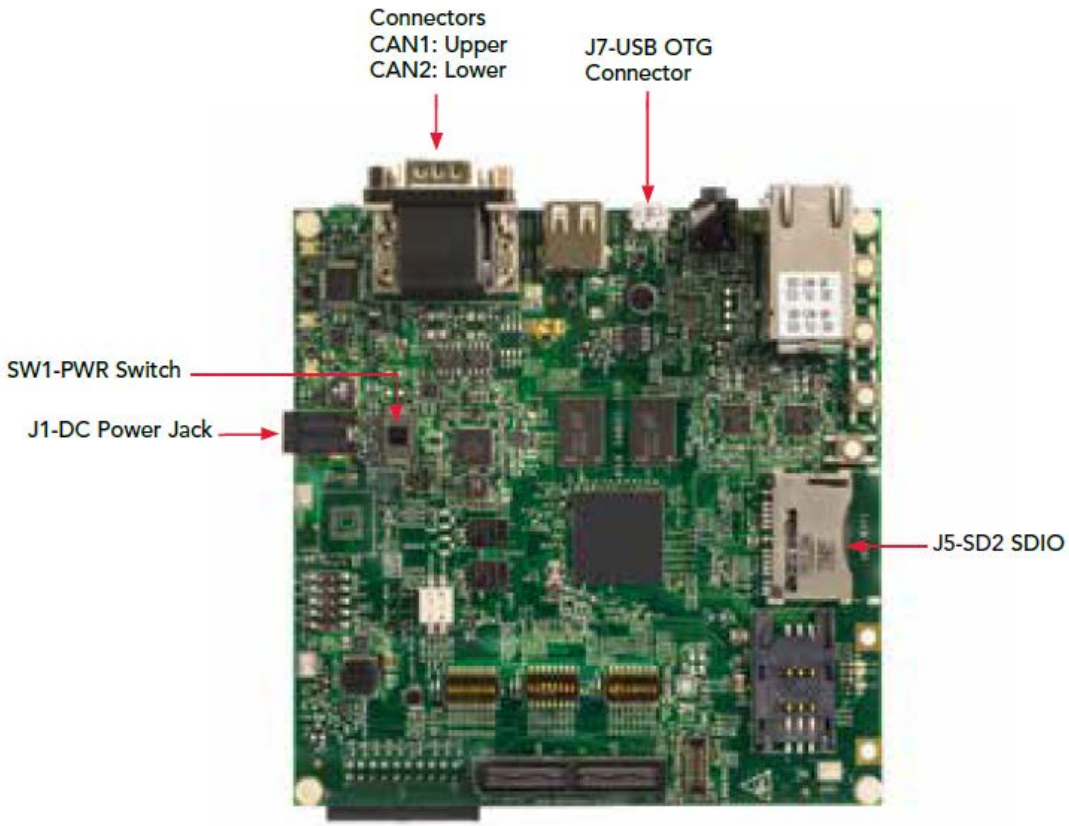
2) You already have a SD card with 8GB memory (or more) and a micro USB cable.

The SX-SDCAC has the onboard antenna enabled by default for evaluation. Please contact Silex technology if you wish to evaluate using the u.fl antenna option included in the box.

To install the SX-SDCAC radio module on the i.MX 6SoloX platform, perform the following steps and refer to Figure 9 below.

1) Turn off the Sabre-SD EVK card using SW1.
2) Disconnect the power supply from the Sabre-SD card.
3) Go to SD4 (J4) and remove the SD Card containing the Freescale supplied OS image.
4) Go to the SX-SDCAC product support page and download the evaluation image as described in Section 4.
5) Load the evaluation image on a blank SD card per the instructions in Section 4.
6) Insert the SD card into SD4 card slot. This is the same location you just removed the Freescale SD card from (See Figure 9 for the location).
7) Insert the SX-SDCAC card into SD3, SD card slot. This is located at the bottom side of the EVK card. (See Figure 9 for the location).
8) The SX-SDCAC driver is pre-configured to Tx and Rx on the u.fl connector (ANT2). A configuration parameter in the qcom_cfg.ini file can change the antenna path to the onboard chip antenna (ANT1) for evaluation. Connect an appropriate dual-band antenna to the u.fl connector.
9) Connect the micro-B end of the Freescale supplied cable into the debug port (J16) on the Sabre-SD card.
10) Connect the Bluetooth connector of the SX-SDCAC to the J19 connecotr of the i.MX6SX SABRE SDB board. (See Figure 10)
11) Plug the other end into the PC to be used as the terminal.

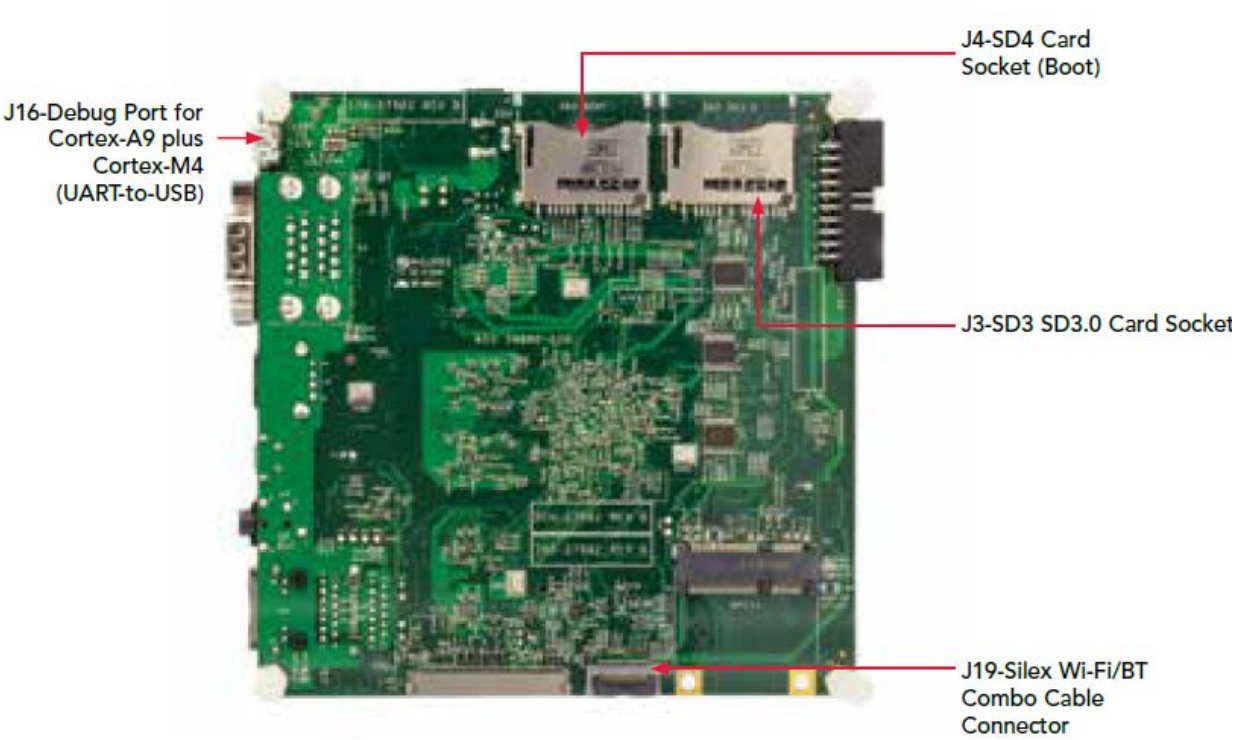
12) Insert the power supply.



13) Power up the Sabre-SD card using SW1.

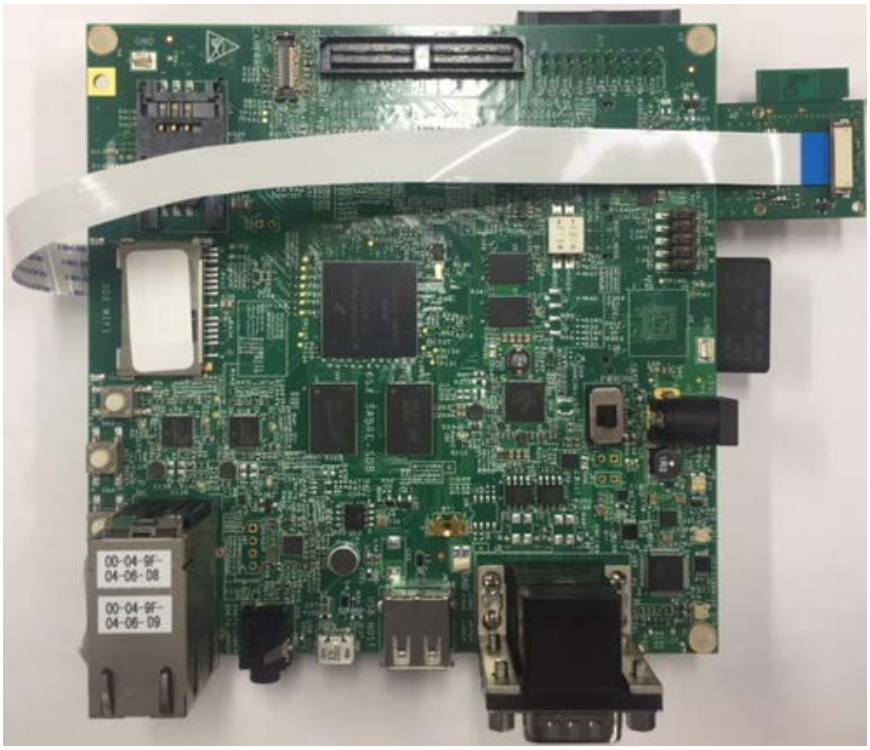
Figure 9 - i.MX 6SoloX SABRE-SDB Layout

Top side



Bottom Side

Figure 10 - Bluetooth Cable Connection



Once the card has been powered up, the boot sequence will start. Connecting to the terminal and logging in is described in Section 8. The serial-to-USB console cable will allow the PC or laptop to show the boot sequence in the console port output. During the boot sequence the LED indicators on the Sabre-SD may change state. This is expected.

When the boot sequence is complete, you should see the following output in the console:

```
Freescale i.MX Release Distro 4.1.15-2.0.1 imx6sxsabresd ttymxc0ttymxc0
imx6sxsabresd login:
```

6 SX-SDCAC EVK System Configuration

The EVK image is configured based around the following system:

- Yocto Build System 2.1.1 (krogoth)
- Freescale-Yocto distribution – 4.1.15-2.0.1
- Silex Reference Driver (version 4.5.20.020 sx 1.0.0.a08)

6.1 Supported Functionality

This image supports the following functionality:

Table 1 - Supported EVK Functionality

Function	Tool	User Guide Documentation Provided?
Wireless LAN Management	iw	✓
Throughput test	iperf3	✓
Soft AP	hostapd	✓
Client Authentication	wpa_supplicant	✓
Configuration utility for network interfaces	ifconfig	✓
DHCP server	udhcpd	✓
DHCP Client	udhcpc	✓
TFTP Client	tftp	✓
Telnet Client	telnet	✓
SSH Client	dropBear	✓

6.2 Login information

- Login: `root`
- No password is set

7 Shutting down or Restarting the i.MX 6SoloX Sabre-SD

The i.MX6SoloX Sabre-SD EVK board should be shut down from the console interface before power is removed.

DO NOT REMOVE POWER WITHOUT SHUTTING DOWN THE OS!

Removing power from the EVK without properly shutting down the operating system could result in the OS being unable to correctly boot due to corrupt system files.

To shut down using the CLI, type the following command:

```
shutdown -h now
```

Wait until the console output indicates that it has shut down and the LED's near the power button on the Sabre-SD EVK have turned off.

Additionally, it may be required to restart the unit from time to time. The following command can be used to restart the Sabre-SD:

```
shutdown –r now
```

8 Configuring WLAN using Command Line Interface

Our evaluation image supports command line control through the Linux console. Access to the Linux console can be found through the USB-Serial interface provided on the Sabre-SD card. Additionally, it is necessary to have a Telnet/terminal emulator application. With the Command Line interface it is possible to configure, enquire and interrogate the network interfaces on the Sabre-SD EVK directly.

The following section will cover how to establish the Linux console connection and provide common console commands and actions for the WLAN device, all supported by the SX-SDCAC from Silex.

8.1 Loading USB-Serial Driver

The PC connected to the Sabre debug port must load a driver to interact with the interface and create a virtual COM port on the debug PC. This should happen automatically when the Sabre-SD board is first powered up.

When the install is complete a COM port number will be assigned to the Sabre-SD card. You will need to know this when using the terminal emulator application.

If needed, the Serial-to USB drivers can be found at
www.ftdichip.com/FTDrivers.htm

8.2 Opening a Linux Console Session

Open the terminal emulator of choice and enter the following:

Type:	**Serial interface**
COM Port:	**<assigned when the virtual COM port driver was installed>**
Configuration:	**115.2K BAUD, 8 data bits, 1 stop bit, no parity**

When the session is opened the Ubuntu Linux prompt will be displayed.

There are a number of available terminal emulators. The following list is not comprehensive but provides guidance on the type of application required a list of available emulators can be found here.

The following two are widely used and are known to function correctly with the Sabre-SD debug port:

PuTTY	www.putty.org
TeraTerm	ttssh2.sourceforge.jp/index.html.en

8.3 Temporary Wireless Network Setup

There is more than one way to configure a network connection from the command line. The method described below uses the commands ifconfig, iw, and udhcpc. The applications are built into the SX-SDCAC evaluation image.

PLEASE NOTE THAT USING THESE COMMAND LINE UTILITIES DOES NOT CREATE A PERMANENT CONFIGURATION. ANY CONFIGURATION INFORMATION WILL BE LOST AFTER RESTARTING THE OPERATING SYSTEM.

A method for creating a permanent configuration is described below in Section 9.

8.4 Logging into the Linux Console

At startup, the system will ask you to login. The login is **root** and there is no set password. Log in with the following command:

```
imx6sxsabresd login: root
```

8.5 Loading the WLAN Driver

As delivered, the wireless driver is not loaded upon startup. To load the driver, the first step is to execute the driver installation script file

```
/home/root/install-wlan-module.sh
```

During the execution of the script file, a prompt will appear and ask whether to modify a couple of parameters in the driver configuration file qcom_cfg.ini. Press the Enter key to use the pre-set values for roaming, or enter “n” and press the Enter key to use the original default settings.

The script file execution copies radio firmware and configuraton files to the **/lib/firmware** and **/lib/firmware/qca** directories respectively.

load_drv.sh, **sxsdpac.sh** and **check_boardata.sh** script files will appear in the **/home/root** directory after completion of the script file.

At the end of the execution, a prompt will appear and ask whether to initiate a reboot to make file changes effective. Type “y” and press the Enter key to initiate reboot.

After the i.MX6SX board has completed the reoobt, execute the driver start-up script file

```
/home/root/load_drv.sh
```

During the execution of the script file, a prompt will appear and ask whether to load BlueZ. Press the Enter key to initialize the Bluetooth I/F of the SX-SDCAC module and to start the BlueZ daemons. Otherwise, Type “n” and press the Entery key to skip the BlueZ start-up process.

The load_drv.sh script file loads compat.ko, cfg80211.ko, and wlan.ko kernel modules. In addition, it sets the regulatory domain to US.

Note: Unless there is a need to reinstall radio firmware and configuration files, **install-wlan-module.sh** only needs to be executed once. Use the **load_drv.sh** script file to load the WLAN driver and to start the Bluetooth I/F.

For customers who plan to install SX-SDPAC wireless module, instead of the SX-SDCAC (SX-SDMAC) wireless module, execute the **sxsdpac.sh** script file to update the board data file for SX-SDPAC.

/home/root/sxsdpac.sh

After completion of the board data file update, the script file will prompt the user to initiate a reboot. Enter y and press the Enter key to initiate reboot.

Unless a need arises to change the wireless module from SX-SDPAC to SX-SDCAC, this script file only needs to be executed once.

To revert back to the SDMAC board data file, delete

/lib/firmware/sdpac_board_data_installed.txt

and execute

/home/root/install-wlan-module.sh

After reboot, run the following command to load the WLAN driver.

/home/root/load_drv.sh

To verify which board data file is installed, run the following script file.

/home/root/check_boarddata.sh

8.6 Making the WLAN Interface Available

By default, the WLAN interface will be enabled after loading the WLAN driver. If the WLAN interface remains disabled, issue the following command:

ifconfig wlan0 up

8.7 List Available Network Interfaces

The first thing you should confirm is that the WLAN interface driver has loaded and the interface is available. To see the available network interfaces enter:

ifconfig –a

Sample output:

```
eth0      Link encap:Ethernet  HWaddr 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)

sit0      Link encap:IPv6-in-IPv4
          NOARP  MTU:1480  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:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

wlan0     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)
```

The listed **`wlan0`** interface indicates the Silex SX-SDCAC WLAN device driver has been loaded and the network interface is available for configuration.

8.8 Check Current Connection Status

To check the current connection status of the WLAN interface, enter the following command:

```
iw wlan0 link
```

Sample Output:

```
Connected to 08:86:3b:28:90:1c (on wlan0)
        SSID: Silex_Test_G
        freq: 2462
        RX: 4329333 bytes (19234 packets)
        TX: 463343 bytes (7356 packets)
        signal: -22 dBm
        tx bitrate: 40.5 MBit/s MCS 2 40MHz

        bss flags:
        dtim period:    1
        beacon int:     100
```


If the interface is not associated to a network, the interface will indicate that no connection has been established.

8.9 Scanning for WLAN Networks

Once the WLAN network interface is available looking for available networks can provide feedback on the RF environment and provide confirmation your target network can be seen by the SX-SDCAC. The

scan has many options that allow control of the scanning function in order to limit the frequencies scanned. There are options for active or passive scanning and scanning for specific SSID's.

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

```
iw wlan0 scan
```

Sample output:

```
BSS 08:86:3b:28:90:1c (on wlan0) -- associated
        TSF: 826785183 usec (0d, 00:13:46)
        freq: 2462
        beacon interval: 100
        capability: ESS ShortSlotTime (0x0401)
        signal: -16.00 dBm
        last seen: 210 ms ago
        Information elements from Probe Response frame:
        SSID: Silex_Test_G
        Supported rates: 1.0* 2.0* 5.5* 11.0* 9.0 18.0 36.0 54.0
        DS Parameter set: channel 11
        ERP: Barker_Preamble_Mode
        Extended supported rates: 6.0 12.0 24.0 48.0
        HT capabilities:
                Capabilities: 0x6c
                        HT20
                        SM Power Save disabled
                        RX HT20 SGI
                        RX HT40 SGI
                        No RX STBC
                        Max AMSDU length: 3839 bytes
                        No DSSS/CCK HT40
                Maximum RX AMPDU length 65535 bytes (exponent: 0x003)
                Minimum RX AMPDU time spacing: 4 usec (0x05)
                HT RX MCS rate indexes supported: 0-15
                HT TX MCS rate indexes are undefined
        HT operation:
                 * primary channel: 11
                 * secondary channel offset: no secondary
                 * STA channel width: 20 MHz
                 * RIFS: 0
                 * HT protection: nonmember
                 * non-GF present: 1
                 * OBSS non-GF present: 0
                 * dual beacon: 0
                 * dual CTS protection: 0
                 * STBC beacon: 0
                 * L-SIG TXOP Prot: 0
                 * PCO active: 0
                 * PCO phase: 0
        Secondary Channel Offset: no secondary (0)
        WMM:     * Parameter version 1
                 * BE: CW 15-1023, AIFSN 3
                 * BK: CW 15-1023, AIFSN 7
                 * VI: CW 7-15, AIFSN 2, TXOP 3008 usec
                 * VO: CW 3-7, AIFSN 2, TXOP 1504 usec
        BSS Load:
                 * station count: 1
                 * channel utilisation: 145/255
                 * available admission capacity: 31250 [*32us]
        Extended capabilities: HT Information Exchange Supported
        Country: TW     Environment: bogus
                Channels [1 - 11] @ 16 dBm
        WPS:     * Version: 1.0
                 * Wi-Fi Protected Setup State: 2 (Configured)
                 * Response Type: 3 (AP)
                 * UUID: bc329e00-1dd8-11b2-8601-08863b28901c
                 * Manufacturer: Belkin International
                 * Model: Belkin N750DB Wireless Router
                 * Model Number: F9K1103 v1
                 * Serial Number: 121119GG106784
                 * Primary Device Type: 6-0050f204-1
                 * Device name: Belkin N750DB Wireless Router
```

```
* Config methods: Label, PBC
* RF Bands: 0x1
```

8.10 Connecting to an Access Point (No security)

The easiest way to get connected and learn how to use the command line interface is to initially connect to an open Access Point (AP). This is an AP with no security enabled. Later in the section we will cover connecting to AP's with network security enabled.

To connect to an AP enter the following command. You will need to know the SSID or network name you wish to connect to:

```
iw dev wlan0 connect <Network SSID>
```

After issuing the above command, you can confirm the Sabre-SD has associated to the AP by repeating the command in Section 8.8.

The iw connect command will fail if an instance of the wpa_supplicant for the WLAN interface is running.

Use the following command to terminate all instances of currently running wpa_supplicant.

killall wpa_supplicant

Note: Configuring DHCP

Setting up the WLAN interface to obtain an IP address dynamically requires the following command:

```
udhcpc –i wlan0
```

To determine if the WLAN interface has successfully obtained an IP address, refer to section 0. If an IP address has been leased to the interface it will be displayed in the returned results for the WLAN interface.

8.11 Configuring a Static IP Address

If the target network does not support DHCP or you are using an AdHoc network it may be necessary to assign a static IP configuration to the WLAN interface. As a minimum, an IP address and subnet mask must be configured, additional settings like a gateway address and DNS server addresses are optional.

To configure a static IP address use the following command:

```
ifconfig wlan0 <IP Address> netmask <network mask>
e.g. ifconfig wlan0 192.168.10.120 netmask 255.255.255.0
```

To configure a gateway IP address use the following command:

```
ip route add default via <gateway IP>
e.g. ip route add default via 192.168.10.1
```

A gateway IP address is required if access to devices on a different subnet is desired. e.g. access to the internet.

To configure a DNS server IP address, use the following procedure:

1) Edit **`resolv.conf`** using the following command:

```
vi /etc/resolv.conf
```

2) Add your name servers by entering the following information to the **`resolv.conf`** file:

```
nameserver {DNS_Server_1_IP_Address}
nameserver {DNS_Server_2_IP_Address}
```

3) Save your changes.

```
:wq
```

One or more DNS server addresses are required if URL's are to be used to access network resources.

9 Persistent WLAN Configuration

To create a persistent configuration (one which survives power cycles) you must use the **wpa_supplicant.conf** file to store WLAN network profiles. Section 12.5 lists a sample wpa_supplicant file as well as explanations for all the various settings.

Below are some of the guidelines to consider when using wpa_supplicant files.

- Multiple supplicant configuration files may be created and stored on the device. The different files can be invoked when the WPA supplicant process is started. This will be covered in section 9.2.
- To use the WPA supplicant you must first establish a network configuration in an appropriately named file. Within the file each network configuration must be defined in a network block. A single configuration file may contain several network blocks, each containing a different network and network security setting. There is no naming convention but references to the security or networks stored in the file will help manage the supplicant configuration files.
- The files are text based and must be stored in the file system. There is no set location for the storage of the configuration files. The full path and file name will be needed when the WPA supplicant is started.

To get started with establishing a persistent WLAN configuration copy one of the example files below and save it to the EVK. You can pick any location you wish but the example below is using the **/etc/wpa_supplicant** subdirectory.

9.1 Stopping the wpa_supplicant process

Before changing the WLAN configuration using WPA supplicant it is necessary to halt any existing wpa supplicant processes. This section assumes the network manager has been stopped (section 9).

To stop any existing wpa supplicant processes follow these steps:

1) Run the following command:

```
ps –A | grep wpa
```

2) The command will output the process information to allow the **wpa_supplicant** process to be stopped.

   Example output:

```
# ps -A |grep wpa
6747 ?        00:00:00 wpa_supplicant
```

3) The process ID is located to the left of the information that is displayed. In the above example, the process ID is **6747**.
4) To stop the **wpa_supplicant** process in the above example type the following command and hit enter (replace the example process ID with your EVK's process ID when running the command):

```
kill 6747
```

An alternate and simpler way to kill the wpa_supplicant process is to issue the following command:

```
killall wpa_supplicant
```

Once the kill command is issued the WPA supplicant process will terminate and an updated process

can be started. Repeating the command in step 1 will return no found processes if the process is killed.

9.2 Configuring WPA / WPA2-PSK using WPA Supplicant

If the target network is using security, it will most likely be a pre-shared key type using WPA or WPA2 encryption. The following covers how to configure the Sabre-SD EVK and SX-SDCAC to use WPA-PSK or WPA2-PSK.

Configuration of the EVK requires editing the WPA supplicant configuration file. Prior to editing the file the network SSID, encryption type and pre-shared key must be known.

To configure the EVK for WPA or WPA2 security, follow these steps:

1) Change the current directory to the one that contains the **`wpa_supplicant.conf`** file:

```
cd /etc
```


If no WPA supplicant configuration file exists you must create one. To do this follow step 1 and enter the name of the configuration file you want to use in step 2.

a) Edit the **`wpa_supplicant.conf`** file:

```
vi wpa_supplicant.conf
```


If you are not familiar of how to use vi editor in Linux, please refer to the following tutorial to use the vi editor to make ans save changes to the file.
http://ryanstutorials.net/linuxtutorial/vi.php

The WPA supplicant configuration file does not have to be named wpa_supplicant.conf.

Providing a file name that suggests the contents of the file can be advantageous when maintenance and use of the files is required e.g. a configuration file holding the network blocks for the manufacturing plant could be named wpa_supplicant_manuf.conf.

See section **Error! Reference source not found.** for a description of the WPA configuration file structure/format.

2) To configure the EVK for use with a network using WPA-PSK:
   a) Find the first line containing the following. This is the first network block and will be used as the network configuration example:

```
network={
```

   b) Edit the lines in the block under the network line, shown in RED, with the details of your specific network configuration:

```
ssid="replace with your network name"
scan_ssid=1
proto=WPA  # for WPA-PSK
key_mgmt=WPA-PSK
pairwise=TKIP
group=TKIP
psk="replace with your pre-shared key"
```

3) To configure the EVK for use with a network using WPA2-PSK:
   a) Find the first line containing the following. This is the first network block and will be used as the network configuration example:

```
network={
```

   b) Edit the lines in the block under the network line, shown in RED, with the details of your specific network configuration:

```
ssid="replace with your network name"
scan_ssid=1
proto=RSN  # for WPA2-PSK
key_mgmt=WPA-PSK
pairwise=CCMP
group=CCMP
psk="replace with your passphrase"
```

4) Each network block must be terminated with a **}**.
5) Save the edited file to the chosen location **(/etc/wpa_supplicant/** in the example above).
6) We can now run the WPA supplicant with the modified configuration file. Enter the following command:

```
wpa_supplicant –B –Dnl80211
–c/etc/wpa_supplicant.conf –i wlan0
```

7) If the WPA supplicant is successfully initialized the system will report back that the initialization was successful. Any other message will identify the errors in the configuration file. If initialization is unsuccessful read the debug message response and adjust the WPA supplicant configuration file accordingly.

8) Successful initialization of the WPA supplicant does not guarantee connection to the target network. For a successful connection, both the configuration must be correct and the network must be in range. To confirm association to the target network issue the following command:

```
iw wlan0 link
```

9) If the response is **Not connected.** Check the contents in the WPA configuration file, edit accordingly, save it and issue the following command:

```
killall –HUP wpa_supplicant
```

10) This will apply the changes made to the WPA supplicant configuration file in step 9. The edited file must be the one initially called during step 5. Repeat steps 7 and 8 until you successfully associate and authenticate to the network.

9.3 Configuring Adhoc Mode

The SX-SDCAC supports peer-to-peer WLAN connectivity. This can be established using the AdHoc network configuration. To configure the WLAN interface for Adhoc, follow the steps in section 9.2,

replacing the block contents in step 2b with the one listed below:

```
ssid=”ADHOC_SDCAC”
mode=1
frequency=2412
key_mgmt=NONE
```

Enabling wpa_supplicant with the command in Section 9.2 Step 6) above will automatically bring up the wlan0 interface in Adhoc mode. It will then try to join an existing Adhoc network. If none is found, it will then create a new network for others to join. The interface will continue to scan for IBSS STAs until one is found to connect with.

Since AdHoc networks do not typically support DHCP servers it is necessary to assign a static IP address to the Sabre-SD WLAN interface, when using an AdHoc network. See section 8.12 to understand how to configure the interface to use a static IP address.

9.4 Configuring EAP-TLS

Configuring the WPA supplicant to use EAP-TLS is the same as WPA-PSK with the exception of the WPA supplicant network block contents. To configure the WLAN interface for EAP-TLS follow the steps in section 9.2, replacing the block contents in step 2b with the one listed below:

```
ssid="replace with your network name"
scan_ssid=1
key_mgmt=WPA-EAP
pairwise=CCMP TKIP
group=CCMP TKIP
eap=TLS
identity="user@example.com"
ca_cert="/etc/cert/ca.pem"
client_cert="/etc/cert/user.pem"
private_key="/etc/cert/user.prv"
private_key_passwd="replace with your private key password"
```

EAP-TLS requires the use of three certificates, the Certificate Authority (CA) certificate, the user certificate and the Private Key certificate. Prior to authentication you will need to source the certificates form the network administrator and load them on the Sabre-SD EVK. The certificates are required to complete authentication on the network.

Loading certificates will be covered in section 10 of the manual.

9.5 Configuring EAP-TTLS

Configuring the WPA supplicant to use EAP-TTLS is the same as WPA-PSK with the exception of the WPA supplicant network block contents. To configure the WLAN interface for EAP-TTLS follow the steps in section 9.2, replacing the block contents in step 2c with the one listed below:

```
ssid="replace with your network name"
scan_ssid=1
key_mgmt=WPA-EAP
eap=TTLS
identity=user@example.com
anonymous_identity="anonymous@example.com"
```

```
ca_cert="/etc/cert/ca.pem"
password="replace with your password"
client_cert="/etc/cert/user.pem"
phase2="auth=MD5"
```

EAP-TTLS requires the use of a Certificate Authority (CA) certificate. Prior to authentication you will need to source the certificate form the network administrator and load it on the Sabre-SD EVK. The certificate is required to complete authentication on the network.

Loading certificates will be covered in section 10 of the manual.

9.6 Configuring PEAP / MsCHAPv2 (PEAPv0)

Configuring the WPA supplicant to use PEAP is the same as WPA-PSK with the exception of the WPA supplicant network block contents. To configure the WLAN interface for EAP-PEAP follow the steps in section 9.2, replacing the block contents in step 2b with the one listed below:

```
ssid="replace with your network name"
scan_ssid=1
key_mgmt=WPA-EAP
eap=PEAP
identity="user@example.com"
password="replace with your password"
ca_cert="/etc/cert/ca.pem"
phase1="peaplabel=0"
phase2="auth=MSCHAPV2"
```

PEAP requires the use of a Certificate Authority (CA) certificate. Prior to authentication you will need to source the certificate form the network administrator and load it on the Sabre-SD EVK. The certificate is required to complete authentication on the network.

Many PEAP implementations ignore the need for the CA Cert. If the certificate is not provided by the network administrator comment out the **`ca_cert`** line in the configuration file.

Loading certificates will be covered in section 10 of the manual.

9.7 Configuring EAP-FAST

Configuring the WPA supplicant to use EAP-FAST is the same as WPA-PSK with the exception of the WPA supplicant network block contents. To configure the WLAN interface for EAP-FAST follow the steps in section 9.2, replacing the block contents in step 2b with the one listed below:

```
ssid="replace with your network name"
scan_ssid=1
key_mgmt=WPA-EAP
eap=FAST
anonymous_identity="optional user identity"
identity="replace with your user identity"
password="replace with your user password"
phase1="fast_provisioning=1"
pac_file="/tmp/wpa_supplicant.eap-fast-pac"
```

10 Loading Certificates

The following section covers how to load certificates on to the file system for use with the EAP-TLS/TTLS/PEAP security modes. Refer to section 12.2.2, to learn how to upload files in linux using TFTP.

Before loading any certificates you will need to obtain them from the network administrator.

The WPA supplicant supports X.509 certificates in .PEM and .DER formats for the CA, user and private key certificates. If the certificates are received in the .PFX or .P12 formats they will need to be converted to the suitable format before being used, see steps below on how to do this.

To convert .PFX or .P12 certificates in to a compatible format for the WPA supplicant follow these steps:

1) Copy certificates to the /etc/cert/ subdirectory.
2) Move to the /etc/cert/ subdirectory:

```
cd /etc/cert
```

3) To extract the CA certificate from the PFX file and convert to .PEM format, issue the following command:

```
openssl pkcs12 -in example.pfx -out ca.pem -cacerts –nokeys
```

4) To extract the user certificate and private keys and convert to .PEM format, issue the following command:

```
openssl pkcs12 -in example.pfx -out user.pem –clcerts
```

The above steps will create a `user.pem` and `ca.pem` file and place them in the /etc/cert/ subdirectory. These files will be the ones referenced in the WPA supplicant network block configuration.

11 Configuring the EVK for AP Mode

11.1 System Configuration

The SabreSD EVK image provided by Silex supports SoftAP mode. This mode allows the SX-SDCAC to act as an access point and manage connections for up to 10 simultaneous clients.

Three programs are installed by default to support this feature:

1) hostapd - Program that controls and configures AP functionality
2) udhcpd – Program that controls the DHCP Server functionality and automatically assigns IP address to connected clients.
3) brctl – Bridge control UI that creates the bridge required to route packets from the wireless interface back to the Ethernet interface

The hostadp and udhcpd programs require configuration files to run, while the bridge is configured in the CLI directly and must be redone on every reboot. The following sections describe how to create and configure these files and programs.

11.2 Stopping the hostapd process

Before setting up the hostapd configuration files, it is required to stop the hostapd process if it is currently running.

To stop any existing hostapd processes, follow these steps:

1) Run the following command:

```
ps –A | grep hostapd
```

2) The command will output the process information to allow the **wpa_supplicant** process to be stopped.

   Example output:

```
# ps -A |grep hostapd
6748 ?        00:00:00 hostapd
```

3) The process ID is located to the left of the information that is displayed. In the above example, the process ID is **6748**.
4) To stop the hostapd process in the above example type the following command and hit enter (replace the example process ID with your EVK's process ID when running the command):

```
kill 6748
```

An alternate and simpler way to kill the hostapd process is to issue the following command:

```
killall hostapd
```

Once the kill command is issued the hostapd process will terminate and an updated process can be started. Repeating the command in step 1 will return no found processes if the process is killed.

11.3 Setup the Wi-Fi for AP Mode

Using the AP mode requires a hostapd configuration file to be prepared. The EVK ships with the full hostapd configuration example, found in **/etc/hostapd.conf**. In this section, we will create two custom hostapd.con files, one for 2.4GHz and one for 5Ghz.

11.3.1 Creating a 2.4GHz Hostapd File

1) Change the current directory to the one that you with to contain the custom hostapd configuration files:

```
cd /etc
```

2) Create and open **hostapd_custom_g.conf** file for editing with the following command:

```
vi hostapd_custom_g.conf
```

3) Modify the **hostapd_custom_g.conf** file to the following. This example file with the default values set to open security for easy testing. The # indicates a comment, and can be removed to enable testing with WPA2-PSK security. Once the modifications have been made save and exit out of the vi editor by typing :wq.

```
ssid=EVK-AP-Test-G
interface=wlan0
driver=nl80211
ctrl_interface=/var/run/hostapd
country_code=US
hw_mode=g
channel=6
ieee80211n=1
#wpa=2
#wpa_passphrase=secret_password
#wpa_key_mgmt=WPA-PSK
#rsn_pairwise=CCMP
```

4) Run AP mode with the command

```
hostapd -B /etc/hostapd_custom_g.conf
```

11.3.2 Creating a 5GHz Hostapd file

1) Change the current directory to the one that you wish to contain the custom hostapd configuration files:

```
cd /etc
```

2) Create and open **hostapd_custom_na.conf** file for editing with the following command:

```
vi hostapd_custom_na.conf
```

3) Modify the **hostapd_custom_na.conf** file to the following. This example file has default values set to open security for making testing easier. The # indicates a comment, and can be removed to enable testing with WPA2-PSK security. Once the modifications have been made save and exit out of the vi editor by typing :wq.

```
ssid=EVK-AP-Test-NA
```

```
interface=wlan0
ctrl_interface=/var/run/hostapd
driver=nl80211
country_code=US
hw_mode=a
channel=36
ht_capab=[HT40+]
ieee80211n=1
#wpa_passphrase=secret_password
#wpa_key_mgmt=WPA-PSK
#rsn_pairwise=CCMP
```

4) Run AP mode with the command

```
hostapd -B /etc/hostapd_custom_na.conf
```


It is recommended that you run hostapd without **-B** (background) option first to ensure that the AP is up and running and that you can connect the intended Wireless client. After this is verified, stop the hostapd and then re-start with "-B" option.

11.4 Prepare the DHCP Server

A configuration file is also required for the DHCP server program to run. For this guide, we will create the **udhcpd.conf** file in the **/etc** folder following the steps below.

If the EVK will eventually bridge to a network that is currently running a DHCP server, this section can be skipped. Enabling two DHCP servers on a network is highly unstable.

1) Change the current directory to the one that you with to contain the **udhcpd.conf** file:

```
cd /etc
```

2) Create and edit the **udhcpd.conf** file. This does not have to be the file name you use, but is chosen for this example.

```
vi udhcpd.conf
```

3) The IP address of the AP and the DHCP lease range must be defined. In this case, "192.168.1.1" is used for the IP address and "192.168.1.100 - 192.168.1.199" is used for the DHCP lease range. Edit the **udhcpd.conf** file to the following:

```
start 192.168.1.100
end   192.168.1.199
interface wlan0
opt router 192.168.1.1
```

Once the modifications have been made save and exit out of the vi editor by typing :wq.

11.5 Testing

To verify AP functionality, use the following command to see the AP status:

```
hostapd_cli get_config
```

Sample output:

```
# hostapd_cli get_config

Selected interface 'wlan0'
bssid=00:80:92:4d:e7:48
ssid=sasaki-ap-test
wps_state=disabled
```

You can also use the following command to check the status of the connected stations:

```
hostapd_cli all_sta
```

Sample output:

```
# hostapd_cli all_sta
Selected interface 'wlan0'
00:24:d6:90:12:be
```

12 Configure the EVK for Bridge Mode

The Silex SabreSD EVK image also comes preinstalled with bridge utility installed to support bridging of two network interfaces. The Linux kernel is also preconfigured to support the bridging function.

12.1 Setup

This section lists the general steps required to setup a wireless bridge.

- Terminate any instance of wpa_supplicant that is currently controlling the WLAN interface. For simplicity use the following command to terminate all instances of wpa_supplicant.

```
killall wpa_supplicant
```

- Launch hostapd using a hostapd.conf with a bridge interface name defined. (e.g. br0)

```
ssid=EVK-AP-Test-G
interface=wlan0
bridge=0
driver=nl80211
ctrl_interface=/var/run/hostapd
country_code=US
hw_mode=g
channel=6
ieee80211n=1
#wpa=2
#wpa_passphrase=secret_password
#wpa_key_mgmt=WPA-PSK
#rsn_pairwise=CCMP
```

- Issue an brctl command to bridge the WLAN interface and the Ethernet interface.

```
brctl addif br0 eth0
```

- Bring up the **br0** interface and assign an IP address.

```
ipconfig br0 192.168.1.1 up
```

- Launch udhcpd server using a udhcpd.conf with the bridge interface name (same name defined in the hostapd.conf)

```
start 192.168.1.100
end   192.168.1.199
interface br0
opt router 192.168.1.1
```

12.2 Testing

To check the bridge function executing the following commands. The test setup assumes the WLAN driver has not been loaded.

```
/home/root/load_drv.sh
```

```
killall wpa_supplicant
```

hostapd -B hostapd.conf

brctl addif br0 eth0

ifconfig br0 192.168.1.1 up

udhcpd udhcpd.conf

Connect PC #1 to EVK's Eth0 port via an Ethernet cable.

Connect PC #2 to EVK via the WLAN interface. (SSID = EVK-AP-Test-G)

Identify PC #1 and PC #2's assigned IP addresses. (e.g 192.168.1.100)

At PC #1 initate a ping command to ping PC #2. (e.g ping 192.168.1.101)

13 Bluetooth

See BlueZ Application Note, P/N 142-20135-210 for instructions on setting up the Bluetooth I/F of the SX-SDCAC module. Section 8 of the document provides basic intructions for setting up a Bluetooth Connection with a PC. Section 9 of the document provides instructions for testing some Bluetooth profiles.

14 Appendix

14.1 Throughput testing with iperf3

The Silex SabreSD EVK image comes preinstalled with iperf v3.1, a program used for evaluating throughput capabilities between the two clients on a network. Using this program will give you a baseline for understanding the throughput capabilities of your network.

This section only discusses basic testing and configuration. For full configuration information, please refer to the iperf documentation at https://iperf.fr/.

14.1.1 Setup

The iperf program must be installed on both the EVK and another client PC to test end to end throughput in the network. The steps below show how to download iperf for your test PC.

1) Download the iperf v3.1 to your client PC from https://iperf.fr/iperf-download.php. Since version 3 is not compatible with version 2, it is important to download this exact version.
2) Extract the file to a location of your choosing. For this example, we will use

   ```
   C:\iperf-3.1.3-win64
   ```

3) Open the Windows Command Prompt by right-clicking on the **Start Menu** and selecting **Command Prompt (Admin)**.
4) Navigate to the iperf directory with the command

   ```
   cd C:\iperf-3.1.3-win64
   ```

5) From here, interact with iperf using the text interface commands.
   a) To run iperf as a server allowing the EVK to connect, type:

      ```
      iperf3 –s
      ```

   b) To run iperf as a client connecting to a server running on EVK, type:

      ```
      iperf3 -c <EVK IP ADDRESS>
      ```

6) Connect the PC to the same network as the Sabre-SD EVK and note the IP address

Since iperf is already installed on the Sabre-EVK image, no external iperf configuration is required. It is required that the wireless interface be enabled and connected to the same network as the test PC.

Both the Client PC and Sabre-SD EVK must be connected to the same network to run iperf as documented in this example.

14.1.2 Testing

To run a throughput test using the EVK to generate the connection to the PC, run the following setup:

On the PC:

iperf3 –s

Sample output:

```
C:\iperf-3.1.3-win64>iperf -s
-----------------------------------------------------------
Server listening on 5201
-----------------------------------------------------------
Accepted connection from 192.168.1.1, port 45470
[  5] local 192.168.1.101 port 5201 connected to 192.168.1.1 port 45471
[ ID] Interval           Transfer     Bandwidth
[  5]   0.00-1.00   sec  4.10 MBytes  34.4 Mbits/sec
[  5]   1.00-2.00   sec  4.55 MBytes  38.1 Mbits/sec
[  5]   2.00-3.00   sec  4.30 MBytes  36.1 Mbits/sec
[  5]   3.00-4.00   sec  4.52 MBytes  37.9 Mbits/sec
[  5]   4.00-5.00   sec  4.50 MBytes  37.7 Mbits/sec
[  5]   5.00-6.00   sec  4.85 MBytes  40.7 Mbits/sec
[  5]   6.00-7.01   sec  4.86 MBytes  40.5 Mbits/sec
[  5]   7.01-8.00   sec  4.56 MBytes  38.5 Mbits/sec
[  5]   8.00-9.00   sec  4.96 MBytes  41.7 Mbits/sec
[  5]   9.00-10.00  sec  4.59 MBytes  38.5 Mbits/sec
[  5]  10.00-10.03  sec   109 KBytes  31.6 Mbits/sec
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth
[  5]   0.00-10.03  sec  0.00 Bytes  0.00 bits/sec                  sender
[  5]   0.00-10.03  sec  45.9 MBytes  38.4 Mbits/sec                  receiver
-----------------------------------------------------------
Server listening on 5201
-----------------------------------------------------------
```

On the Sabre-SD EVK:

iperf3 –c <PC IP Address>

Sample output:

```
root@imx6sxsabresd:~# iperf -c 192.168.0.101
Connecting to host 192.168.1.101, port 5201
[  4] local 192.168.1.1 port 45471 connected to 192.168.1.101 port 5201
[ ID] Interval           Transfer     Bandwidth       Retr  Cwnd
[  4]   0.00-1.00   sec  4.33 MBytes  36.3 Mbits/sec    0   66.5 KBytes
[  4]   1.00-2.00   sec  4.52 MBytes  37.9 Mbits/sec    0   66.5 KBytes
[  4]   2.00-3.00   sec  4.31 MBytes  36.1 Mbits/sec    0   66.5 KBytes
[  4]   3.00-4.00   sec  4.49 MBytes  37.6 Mbits/sec    0   66.5 KBytes
[  4]   4.00-5.00   sec  4.56 MBytes  38.2 Mbits/sec    0   66.5 KBytes
[  4]   5.00-6.00   sec  4.82 MBytes  40.4 Mbits/sec    0   66.5 KBytes
[  4]   6.00-7.00   sec  4.87 MBytes  40.9 Mbits/sec    0   66.5 KBytes
[  4]   7.00-8.00   sec  4.61 MBytes  38.7 Mbits/sec    0   66.5 KBytes
[  4]   8.00-9.00   sec  4.95 MBytes  41.5 Mbits/sec    0   66.5 KBytes
[  4]   9.00-10.00  sec  4.53 MBytes  38.0 Mbits/sec    0   66.5 KBytes
- - - - - - - - - - - - - - - - - - - - - - - - -
[ ID] Interval           Transfer     Bandwidth       Retr
[  4]   0.00-10.00  sec  46.0 MBytes  38.6 Mbits/sec    0             sender
[  4]   0.00-10.00  sec  45.9 MBytes  38.5 Mbits/sec                  receiver
```

The above test configuration generates data from the PC and sends it to the Sabre-SD EVK. To reverse the configuration, simply reverse the commands on the PC and Sabre-SD.

The throughput may be different depending on the direction that the data is transferred. This is expected behavior and due to processor architecture differences between the i.MX6 (RISC) and Intel (CISC) chips and how efficient each is at processing data.

14.2 TFTP Client

The Silex SabreSD EVK image also comes preinstalled with a tftp client, which allows it to get a file from or put a file onto a remote host using the Trivial File Transfer Protocol (TFTP).

This section only discusses basic testing and configuration. For more details on TFTP please refer to https://linux.die.net/man/1/tftp.

14.2.1 Setup

The setup assumes you have a TFTP server installed on the network that you need to upload or pull a file from. To test this you will need the below information:

1) The IP address of the TFTP server
2) The file name that the TFTP server is expecting to pull or push.

14.2.2 Testing

To upload a file to the TFTP server follow the below steps.

```
tftp <Server IP Address> -c put <myfile> <theirfile>
```

Where "myfile" is the name of the file you wish to upload and "theirfile" is the name that the file should have on the TFTP server.

You might also want to use the "-v" command parameter so that if something goes wrong you can see what it was:

```
tftp -v 192.168.1.1 -c put myfile theirfile
```

If the server is running on another port, say 8069, then the command syntax would be

```
tftp -v 192.168.1.1 8069 -c put myfile theirfile
```

If the local file already has the correct name, then the command is simply

```
tftp -v 192.168.1.1 -c put myfile
```

To download files from the server, “get” command is used instead of “put”.

14.3 Telnet

Telnet is a user command and an underlying TCP/IP protocol for remotely accessing and managing a device. Through Telnet, an administrator or another user can access someone else's computer remotely.

This section only discusses basic testing and configuration. For more details on Telnet, please refer to http://www.Telnet.org/htm/faq.htm.

14.3.1 Setup

The setup assumes you have a server installed on the network. You will need the below information

1) The IP address of the Telnet server
2) Login and Password of the host if applicable.

14.3.2 Testing

A Telnet client does not listen on any ports, but rather attempts to connect to TCP. If it connects correctly, the user may be prompted to log in and then has an interactive shell session. Because Telnet communicates by default using plain, unencrypted text (including your password!), this is NOT recommended.

A quick test is to use Telnet to test the http server. The Telnet client program can be used to connect to other TCP ports than the default port 23.

```
telnet <hostname> <TCP port>
```

Because http servers usually listen on port 80, you can use the Telnet program to connect to port 80, and see what the http server is sending to browsers when they connect:

```
telnet google.com 80
```

That will connect, but you must then type the following text, which is just a simple http request (case is important):

```
GET / HTTP/1.1
host: test.com
```

You'll then see all the raw html that google serves to browsers. You can use this same concept to test your own http servers.

14.4 SSH

The Silex SabreSD EVK image also comes preinstalled with an ssh client (dropbear). It is a Secure Shell-

compatible client designed to be small enough to be used in small memory environments, while still being functional and secure enough for general use.

This section only discusses basic testing and configuration. For more details on dropbear ssh, please refer to https://linux.die.net/man/1/dbclient. http://linuxcommand.org/man_pages/ssh1.html.

14.4.1 Setup

It's recommended to have a virtual server with a newly built template so that if you accidentally delete something not meant to be deleted, you could rebuild the server and start everything from scratch.

14.4.2 Testing

SSH stands for Secure Shell. It's a protocol used to securely connect to a remote server/system.

The basic command used to connect to a server is:

```
ssh user@serverip
```

where,

serverip: *Server's IP Address*

user: *username*

Once you enter this command, you will be prompted for a password. If you are connecting for the first time, you will also be prompted with a warning message that the server you are connecting to is not recognized, simply type in "yes" on the command line.

That's it, you are connected and can manage your files via Terminal. If you want to exit the remote server and get back to your local machine, simply type in "exit" in the command line and hit enter.

14.5 wpa_supplicant

The WPA supplicant used in the SX-SDCAC is a cross platform WPA supplicant implementation of 802.11i, with support for a number of operating systems, including Linux. Not only is it a full featured WPA2 supplicant, it also supports WPA and older wireless LAN security protocols.

wpa_supplicant can authenticate with any of the following EAP (Extensible Authentication Protocol) methods:

- EAP-TLS
- EAP-PEAP (both PEAPv0 and PEAPv1)
- EAP-TTLS
- EAP-SIM
- EAP-AKA
- EAP-PSK (experimental)
- EAP-FAST

- EAP-PAX
- EAP-SAKE
- EAP-GPSK
- LEAP (note: requires special functions in the driver)

The above list of EAP types is included for reference only and does not represent the supported and tested capabilities of the WPA supplicant included in the SX-SDCAC evaluation image.

When initiated the wpa_supplicant requires a configuration file to identify the available network configurations that should be used to authenticate. The configuration file contains a number of parameters, a control interface, scanning selection and one or more network blocks. An example of the configuration file contents is shown below:

```
ctrl_interface=/var/run/wpa_supplicant
ap_scan=1
network={
        ssid="Network_SSID1"
        scan_ssid=1
        proto=RSN  # for WPA2-PSK
        key_mgmt=WPA-PSK
        pairwise=CCMP TKIP
        group=CCMP TKIP
        psk="password1"
}
network={
        ssid="Network_SSID2"
        scan_ssid=1
        proto=WPA  # for WPA-PSK
        key_mgmt=WPA-PSK
        pairwise=CCMP TKIP
        group=CCMP TKIP
        psk="password2"
}
```

Full descriptions of the wpa_supplicant and the wpa_supplicant.conf file can be found online. The following table outlines the commonly used parameters in the network blocks

Table 2 - WPA Supplicant Configuration File Parameters

Parameter	Options	Description
ssid	<network name>	Network name [Mandatory]
ap_scan	1	wpa_supplicant initiates scanning and AP selection; Must be set to this value.
scan_ssid	Control device probe format	
	0	Do not scan this SSID with specific Probe Request frames (default)
	1	Scan with SSID-specific Probe Request frames (this can be used to find APs that do not accept broadcast SSID or use multiple SSIDs; this will add latency to scanning, so enable this only when needed)
proto	List of acceptable protocols	
	WPA	WPA/IEEE 802.11i/D3.0
	RSN	WPA2/IEEE 802.11i (WPA2 is an acceptable alias for RSN)
key_mgmt	List of acceptable authenticated key management protocols	
	WPA-PSK	WPA pre-shared key

Parameter	Options	Description
	WPA-EAP	WPA using EAP authentication
	IEE8021X	IEEE 802.1X using EAP authentication and (optionally) dynamically generated WEP keys.
	NONE	WPA is not used. Plain text or static WEP can be used.
pairwise	List of accepted pairwise (unicast) ciphers for WPA	
	CCMP	AES in Counter mode with CBC-MAC [RFC 3610, IEEE 802.11i/D7.0]
	TKIP	Temporal Key Integrity Protocol [IEEE 802.11i/D7.0]
	NONE	Use only Group Keys (deprecated, should not be included if APs support pairwise keys)
group	List of accepted group (broadcast/multicast) ciphers for WPA	
	CCMP	AES in Counter mode with CBC-MAC [RFC 3610, IEEE 802.11i/D7.0]
	TKIP	Temporal Key Integrity Protocol [IEEE 802.11i/D7.0]
	WEP104	WEP (Wired Equivalent Privacy) with 104-bit key
	WEP40	WEP (Wired Equivalent Privacy) with 40-bit key
psk	<network passphrase>	The 256-bit pre-shared key used in WPA-PSK mode can be entered either as 64 hex-digits, i.e., 32 bytes or as an ASCII passphrase (in which case, the real PSK will be generated using the passphrase and SSID). ASCII passphrase must be between 8 and 63 characters (inclusive). This field is not needed, if WPA-EAP is used.
eap	List of accepted EAP methods	
	TLS	EAP-TLS. Requires client and server (CA) certificates.
	PEAP	EAP-PEAP with tunneled EAP authentication. Requires server (CA) certificate.
	TTLS	EAP-TTLS with tunneled EAP or PAP/CHAP/MSCHAP/MSCHAPV2 authentication. Requires client and server (CA) certificates.
	FAST	EAP-FAST.
identity	<identity>	Identity string for EAP authentication
anonymous_identity	<anon_identity>	Identity string for EAP authentication to be used as the unencrypted identity with EAP types that support different tunneled identity e.g. EAP-TTLS.
password	<password>	A password string for EAP authentication.
ca_cert	<cert path and file name>	File path to CA certificate file (PEM/DER). This file can have one or more trusted CA certificates. If ca_cert and ca_path are not included, server certificate will not be verified. This is insecure and a trusted CA certificate should always be configured when using EAP-TLS/TTLS/PEAP. Full path should be used since working directory may change when wpa_supplicant is run in the background.
client_cert	<cert path and file name>	File path to client certificate file (PEM/DER). Full path should be used since working directory may change when wpa_supplicant is run in the background.
private_key	<cert path and file name>	File path to client private key file (PEM/DER/PFX). When PKCS#12/PFX file (.p12/.pfx) is used, client_cert should be commented out. Both the private key and certificate will be read from the PKCS#12 file in this case. Full path should be used since working directory may change when wpa_supplicant is run in the background.
private_key_passwd	<password>	Password for Private key file.
phase1	<phase1 parameter>	outer authentication parameters.
	peaplabel=0	EAP-PEAP. Required for PEAPv0.
	fast_provisioning=1	EAP-FAST. Allows unauthenticated provisioning.
phase2	<phase 2 parameter>	Inner authentication parameters
	auth=MSCHAPV2	EAP-PEAP (required)
	autheap=MSCHAPV2	EAP-TTLS (required)
	autheap=MD5	
pac_file	<file path and name>	File path for the PAC entries. wpa_supplicant will need to be able to create this file and write updates to it when PAC is being provisioned or refreshed. Full path to the file should be used since working directory may change when wpa_supplicant is run in the background.

The above table is NOT comprehensive and is not meant as a complete guide to the wpa_supplicant parameter list and options. Please refer to a full description for completeness:

wpa_supplicant full description...

14.6 Bluetooth Support

The SX-SDCAC evaluation image provides the ability to examine the Wi-Fi capability of the SX-SDMAC module, but does not provide any support to evaluate its Bluetooth capability.
Although the SX-SDCAC has a custom connector which provides access to the Bluetooth functionality of the module, this interface is at a logical level and is not suitable for quick connection to a host device for evaluation. If you would like to evaluate the Bluetooth capability of the SX-SDMAC, please contact your Silex sales representative at 866-765-8761to obtain the necessary hardware and documentation.

Revision History

Rev No.	Date	By	Comments
B	8/28/2017	Babar Hashim	Initial Release

Silex Technology America, Inc.
www.silexamerica.com