AN1277: Using RS9116N with Raspberry Pi

Version 1.1

10/21/2020

Table of Contents

1 About this Document

This document helps users to bring up RS9116N EVK with Raspberry Pi CM3 module.

2 Introduction

RS9116N Open Source Driver (OSD) is a SoftMAC driver which interacts with the Linux wireless MAC layer, i.e., MAC80211. The driver is a group of simple and efficient kernel modules which currently supports RS9116N chipsets and it can be ported to any embedded platform in-addition to X-86 platform.

We will walk you through the detailed steps with which user can bring up RS9116N device on Raspberry Pi CM3 module to enable WLAN support.

3 Prerequisites

3.1 Software

WLAN Driver – RS9116N Open Source Driver package can be downloaded from:

https://www.silabs.com/wireless/wi-fi/rs9116-wi-fi-transceiver-modules

https://www.silabs.com/support/resources

3.2 Hardware

- RS9116N n-Link Module
- Raspberry Pi CM3 Board

4 Terminology

In this section common acronyms and abbreviations used in this document are listed.

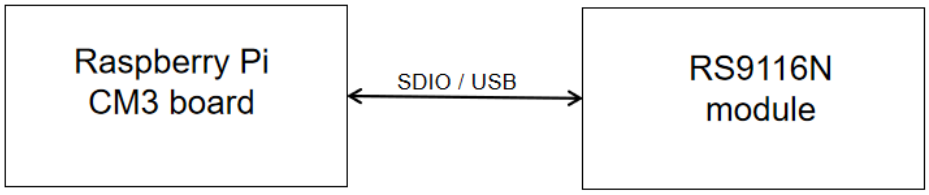
- NL 80211 - nl80211 is the new 802.11 netlink interface public header.
- AP- Access point
- RPi - Raspberry Pi
- STA - Wi-Fi client
- EVB- Evaluation Board
- EVK – Evaluation Kit

5 Bring up of RS9116N on Raspberry Pi CM3

5.1 Description

WLAN module enables user for Wi-Fi capability on a target platform. We will be enabling WLAN support on Raspberry Pi CM3 platform here using RS9116N EVK with SDIO interface.

5.2 Block Diagram



5.3 Use Case

Enabling WLAN support on target platform, will be used as AP / STA Mode.

5.4 Benefits

Wi-Fi capability on target platform, which can be used to use for network connectivity.

6 Execution Steps

6.1 Configuration Params for Raspberry Pi CM3

6.1.1 Connecting RS9116 EVK with SDIO Interface and Configuring SDIO Pins:

RPi has built-in WLAN module on RPI3. Inbuilt WLAN module is configured with GPIO pins 22-27(SD1).

1. For integrating Silicon Labs SDIO interface, we need to disable built in WLAN chip and map RS9116N SDIO with the GPIO pins 34-39(SD1) by adding below configuration to the /boot/config.txt file on RPI3 platform:

```
dtoverlay=sdio, poll_once=off, gpios_34_39, bus_width=4, sdio_overclock=12
```

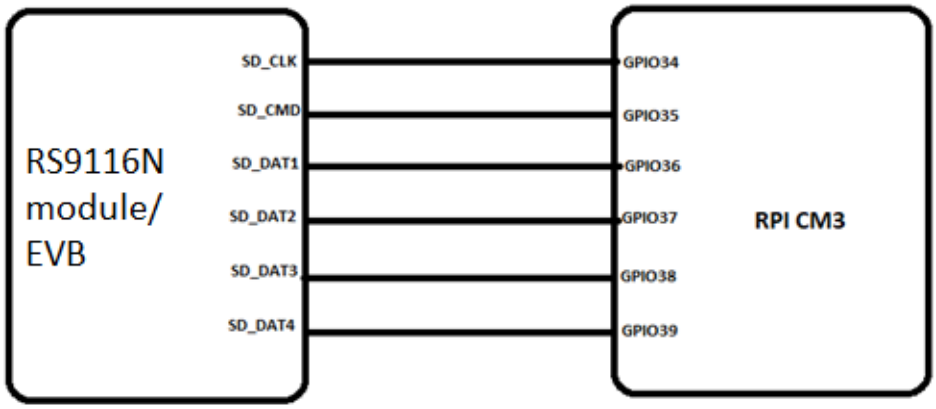
2. Map the SDIO pins from Silicon Labs EVK to the RPI3 GPIOs properly as mentioned in the below table.

GPIO PIN	Default pull	ALT3
34	High	SD1_CLK
35	High	SD1_CMD
36	High	SD1_DAT0
37	Low	SD1_DAT1
38	Low	SD1_DAT2
39	Low	SD1_DAT3

3. After configuring the **/boot/config.txt** file reboot the RPI3.
4. Verify the SDIO detection with the below command from command terminal
5. It will display **0x41b** as output to make sure Silicon Labs SDIO is detected properly.

```
# cat /sys/bus/sdio/devices/mmcX\:fffd\:/vendor
```

Following is SD Line Mapping between RS9116N EVK and RPi CM3.



6.2 Configuration Params for Driver Package

a. Download the image for Raspberry Pi Module from the below link or use the user kernel
https://downloads.raspberrypi.org/raspbian_full_latest
and flash the image into SD card as per manufacturer instructions**.**

b. Insert the SD card in the SD card slot of the Raspberry pi Platform and power up the platform.

c. Install the following Packages on Raspberry pi CM3 board.

```
# apt-get install libnl-3-dev
# apt-get install libnl-genl-3-dev
# apt-get install raspberrypi-kernel-headers
# apt-get install -y build -essential bc bison flex libssl-dev
```

These packages will install the required kernel headers and build essentials for the RPI Board.

d. Download the RS9116N OSD driver package from the below link:
https://www.silabs.com/support/resources.p-wireless_wi-fi_rs9116-wi-fi-transceiver-modules
- RS9116N n-link Open Source Driver for Linux

e. Extract the package using the following command:

```
# unzip RS9116.NX0.NL.GNU.LNX.OSD.<version>.zip
```

6.2.1 Compiling Driver

Get into rsi directory of the driver package and do compilation as given below.

1. Configure build flags in driver source.

```
# cd rsi
```

2. Open Makefile and configure build flags. Below are the build flags to be set based on the usage of driver. Selecting the required options shall reduce the binary size which is important for kernel modules particularly on embedded platforms.

   a. **KERNELDIR**: Provide the kernel source path here. For example, on X-86 below path is used.

```
KERNELRELEASE=$(Shell uname -r)
KERNELDIR=/lib/modules/$(KERNELRELEASE)/build
```

   b. **CONFIG_RSI_COEX_MODE**: Enable this flag when Wi-Fi and BT coexistence mode is used.

   c. **CONFIG_RSI_DEBUGFS**: Debugfs is used by driver to take dynamic configuration from user. Supported debugfs based configurations are listed in the corresponding feature sections in RS9116N Open Source Driver Technical Reference Manual.

   d. **CONFIG_RSI_BT_ALONE:** Enable this flag when only BT EDR/ BT LE only mode is used.

3. Build the driver using make command.

```
# make
```

After completion of compilation, the driver generates the following modules in the rsi folder according to the configuration. They are outlined below:

- rsi_91x.ko

- rsi_usb.ko
- rsi_sdio.ko

6.3 Installation of Driver

In order to install the driver, use the following commands:

1.) Before installing the driver, install the dependencies using below commands.

```
# modprobe mac80211
# modprobe cfg80211
# modprobe bluetooth
```

Insert rsi_91x.ko with the required module params (configuration) as shown below:

```
# insmod rsi_91x.ko dev_oper_mode=<mode> rsi_zone_enabled=<val> . . .
```

Select dev_oper_mode = 1 for STA/AP modes. For all other supported modes please refer RS9116N OSD Technical Reference Manual.

2.) For USB interface, enter the below command:

```
# insmod rsi_usb.ko
```

3.) For SDIO interface, enter the below command:

```
# insmod rsi_sdio.ko sdio_clock=<clk_val>
```

Here the clk_val is 1 to 50 in Mhz.

You can install either USB or SDIO or both depending upon the selection of the interface while compiling the driver.

After successful installation, a new wireless interface shall be created or WLAN and/or BT/BLE as per the dev_oper_mode selection.

If **WLAN** is selected, interface details can be verified using below commands.

Name of the Wi-Fi interface created after successful installation of the driver can be seen using 'ifconfig' command.

```
# ifconfig -a
```

You should expect an output like the sample shown below with all other available interfaces included.

```
wlan0  flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet6 fe80::8da:1aff:fe1e:d1c8  prefixlen 64 scopeid 0x20<link>
        ether 88:da:1a:1e:d1:c8  txqueuelen 1000  (Ethernet)
        RX packets: 3 bytes 372 (372.0 B)
        RX errors 0 dropped 0 overruns 0 frame 0
        TX packets: 6 bytes 696 (696.0 B)
        TX errors 0 dropped 0 overruns 0 collisions:0
```

6.3.1 Testing Wi-Fi STA Mode

This section provides steps to configure Wi-Fi station mode using wpa_supplicant.

1. Before running supplicant, stop the existing network manager and unblock WLAN from rfkill. Below commands are used to stop the network-manager on different Linux distribution.

    a. For ubuntu, we need to use below command.

```
# service network-manager stop
```

    b. For fedora, we need to use below command.

```
# service NetworkManager stop
```

    c. To stop rfkill blocking WLAN, we need to use below command.

```
# rfkill unblock wlan (or) #rfkill unblock all
```

For station mode connectivity, ensure that the dev_oper_mode is set equal to 1.

- dev_oper_mode=1

2. Create a **sta_settings.conf** file with below information. Also please fill the information like ssid, psk etc corresponding to the AP you intend to connect in this file. Sample sta_settings.conf file is available within scripts directory of release package with basic configurations required. User may use this file and edit the information as explained below. For the details of all configurations available please refer open source supplicant wpa_supplicant.conf file.

```
ctrl_interface=/var/run/wpa_supplicant
update_config=1
```

Also add network block to the sta_settings.conf file as per the AP security. Example network block for different security modes are listed below.

i. For Open (non-Secure) mode:

```
network={
        ssid="<SSID of Access Point>"
        key_mgmt=NONE
        priority=3
}
```

ii. For WPA2-PSK (CCMP) mode:

```
network={
        ssid="<SSID of Access Point>"
        key_mgmt=WPA-PSK
        psk=<passphrase specified in the Access Point>
        proto=WPA2
        pairwise=CCMP
        group=CCMP
}
```

The pass phrase can be input either in ASCII or Hexadecimal formats:

ASCII Format: psk="very secret passphrase"
Hexadecimal Format: psk= 06b4be19da289f475aa46a33cb793029d4ab3db7a23ee92382eb0106c7

iii. For WPA3 security mode

To connect in WPA3, we need to compile the latest supplicant with above flags enabled in wpa_supplicant .config file
Note: WPA3 Enterprise security mode is not supported in this release.

```
CONFIG_SAE=y
CONFIG_IEEE80211W=y
```

```
pmf=2
network={
        ssid="<SSID of Access Point>"
        key_mgmt=SAE
        psk=<passphrase specified in the Access Point>
        ieee80211w=2
 }
```

iv. For WPA2-EAP TLS (Enterprise mode) mode:

```
network={
      ssid="<SSID of Access Point>"
      key_mgmt=WPA-EAP
      eap=TLS
      anonymous_identity="tlsuser"
      identity="test"
      password=<passphrase specified in the Access Point>
      ca_cert="/etc/certs/wifiuser.pem"
      client_cert="/etc/certs/wifiuser.pem"
      private_key_passwd=<private key password>
      private_key="/etc/certs/wifiuser.key"
      pairwise=CCMP TKIP
      group=CCMP TKIP
      proto=WPA2 WPA
      priority=20
}
```

In EAP-TLS user must copy client certificates in a path and the path needs to be configured in network block as given above.

v. For WPA2-EAP PEAP (Enterprise mode) mode:

```
network={
        ssid="<SSID of Access Point>"
        key_mgmt=WPA-EAP
        eap=PEAP
        anonymous_identity="peapuser"
        identity="test"
        password=<passphrase specified in the Access Point>
        pairwise=CCMP TKIP
        group=CCMP TKIP
        proto=WPA2 WPA
        priority=20
}
```

vi. For WPA2-EAP TTLS (Enterprise mode) mode:

```
network={
        ssid="<SSID of Access Point>"
        key_mgmt=WPA-EAP
        eap=TTLS
        anonymous_identity="ttlsuser"
        identity="test"
        password=<passphrase specified in the Access Point>
        pairwise=CCMP TKIP
        group=CCMP TKIP
        proto=WPA2 WPA
        priority=20
}
```

To connect to an Access Point whose SSID is not broadcast (Hidden), add the following line to the network block.

```
scan_ssid=1
```

For example**:**

```
network={
         ssid="<SSID of Access Point>"
         scan_ssid=1
         key_mgmt=NONE
 }
```

3. Start the supplicant using below command:

```
# wpa_supplicant -i <interface_name> -D nl80211 -c sta_settings.conf -dttt > supp.log &
```

- "–i" option specifies the Wi-Fi interface name
- <interface name> - This name as listed in iw dev output.
- "-D" specifies the driver interface to be used. In open source driver it is nl80211.
- "-c" specifies the supplicant configuration file.
- "-d" specifies the log level of supplicant. You can append more d's to it for more detailed logs.

Note: For configuring supplicant conf for 4.19.97 kernel please refer section "WPA_Supplicant configuration for 4.19.97 kernel on RPI "

4. To check the scan results please use below command.

```
# wpa_cli -i <interface_name> scan_results
```

For example, above command will give scan results output as follows.

```
bssid               / frequency   / signal level    / flags                  / ssid
50:d4:f7:1e:5a:40   2457            -21             [WPA2-PSK-CCMP][ESS]     TP_LINK
04:79:70:72:03:e7   2412            -31             [ESS]                    honor_9i
```

5. To check whether the connection is successful or not use below command

```
# iwconfig <interface_name>
```

For example, if connection is successful, we will see below output.

```
wlan0     IEEE 802.11bgn  ESSID:"Range"  Nickname:""
          Mode:Managed  Frequency:2.412 GHz  Access Point: 38:A4:ED:DE:BB:06
          Bit Rate:39 Mb/s   Tx-Power=16 dBm
          Retry short limit:7   RTS thr:2353 B   Fragment thr:2352 B
          Encryption key:off
          Power Management:off
          Link Quality=80/80  Signal level=-28 dBm  Noise level:0 dBm
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

If the connection is successful, then the connected Access point SSID along with the MAC address is displayed as shown above. If it is not connected to an Access point, a message **"Not Associated"** is displayed as shown below.

```
wlan0     IEEE 802.11  ESSID:off/any
          Mode:Managed  Access Point: Not-Associated   Tx-Power=0 dBm
          Retry short limit:7   RTS thr:off   Fragment thr:off
          Encryption key:off
          Power Management:off
```

6. IP for the device can be set in two ways either get IP dynamically from AP or set static IP. To obtain dynamic IP from AP, use below commands.

```
# dhclient < interface_name > -r
# dhclient < interface_name > -v
```

To set static IP to STA use below command.

```
# ifconfig <interface_name> <IP_address>
```

7. To check whether IP is assigned or not use below command.

```
# ifconfig <interface_name>
```

Output:

```
wlan0:  flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.114  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::224:d7ff:fe56:54dc  prefixlen 64  scopeid 0x20<link>
        ether 00:24:d7:56:54:dc  txqueuelen 1000  (Ethernet)
        RX packets 31160  bytes 31082515 (29.6 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 23356  bytes 3367496 (3.2 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Now user can perform data transfer tests like ping, iperf … etc.

6.3.2 WPA_Supplicant Configuration for 4.19.97 Kernel on RPI

By default, wpa_supplicant service will be running in background on RPI CM3 platform for default built in wlan chip which is having an interface wlan0.

1. Connect RS9116N EVB to RPI CM3 through USB.

2. Rename the wpa_supplicant.conf file at path /etc/wpa_supplicant/ to wpa_supplicant-wlan0.conf

```
#mv /etc/wpa_supplicant/wpa_supplicant.conf /etc/wpa_supplicant/wpa_supplicant-wlan0.conf
```

3. Create one more supplicant config file for wifi interface (i.e. Silicon Labs interface).

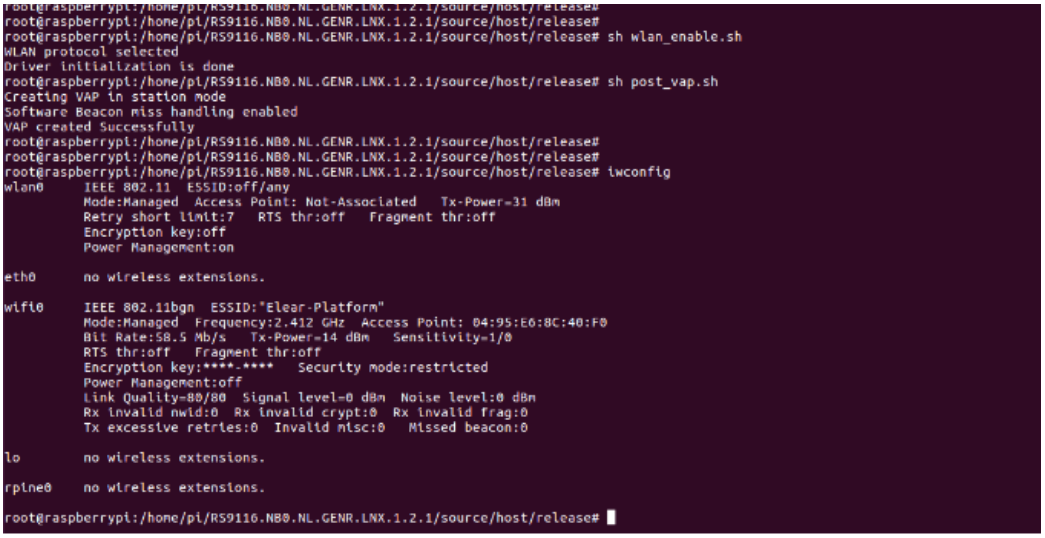
```
#vi wpa_supplicant-<interface_name>.conf
```

4. Copy the content of config file (Example: sta_settings.conf ….) you require into wpa_supplicant-<interface_name>.conf and place it at path /etc/wpa_supplicant/folder.

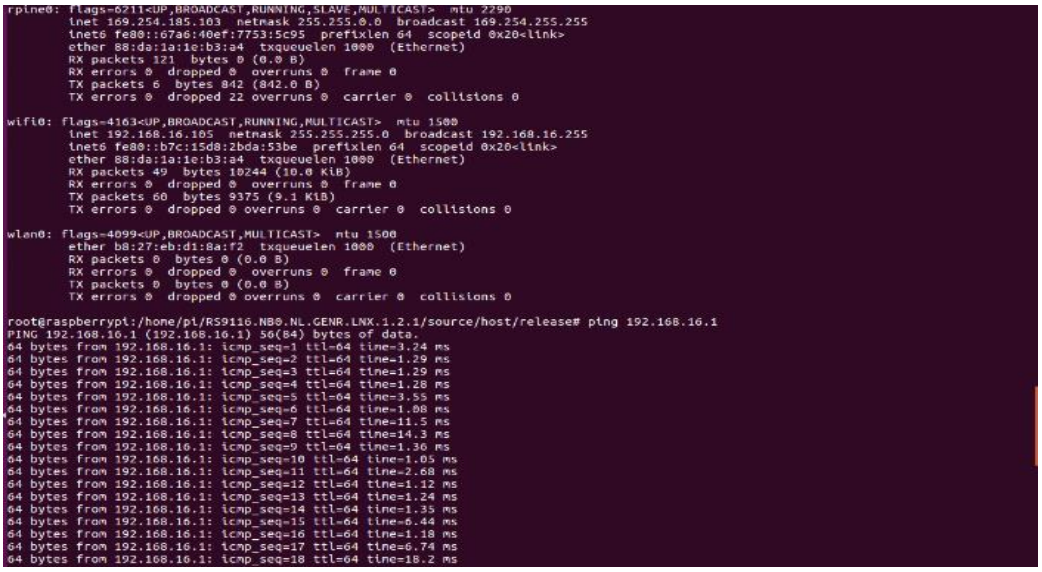
Edit the wpa_supplicant-<silabs-interface>.conf at path /etc/wpa_supplicant/ for any config changes.

7 Expected Results

7.1 Station Successfully Connected to Access Point



7.2 Pinging to IP address of AP is Successful



8 Summary/Conclusion

This document provides detailed instructions for compilation, installation, testing, and troubleshooting of RS9116N module in station mode using Raspberry Pi CM3 board.

9 References and Related Documentation

- Refer to our RS9116N Open Source Driver Technical Reference Manual, Data Sheet and Module Integration Guide, which are available under TECH-DOCS section of https://www.silabs.com/wireless/wi-fi/rs9116-wi-fi-transceiver-modules.in our documentation portal: https://www.silabs.com/support/resources

- Refer to below link for Raspberry Pi CM3:
  https://www.raspberrypi.org/documentation/hardware/computemodule/datasheets/rpi_DATA_CM_1p0.pdf

10 Troubleshooting

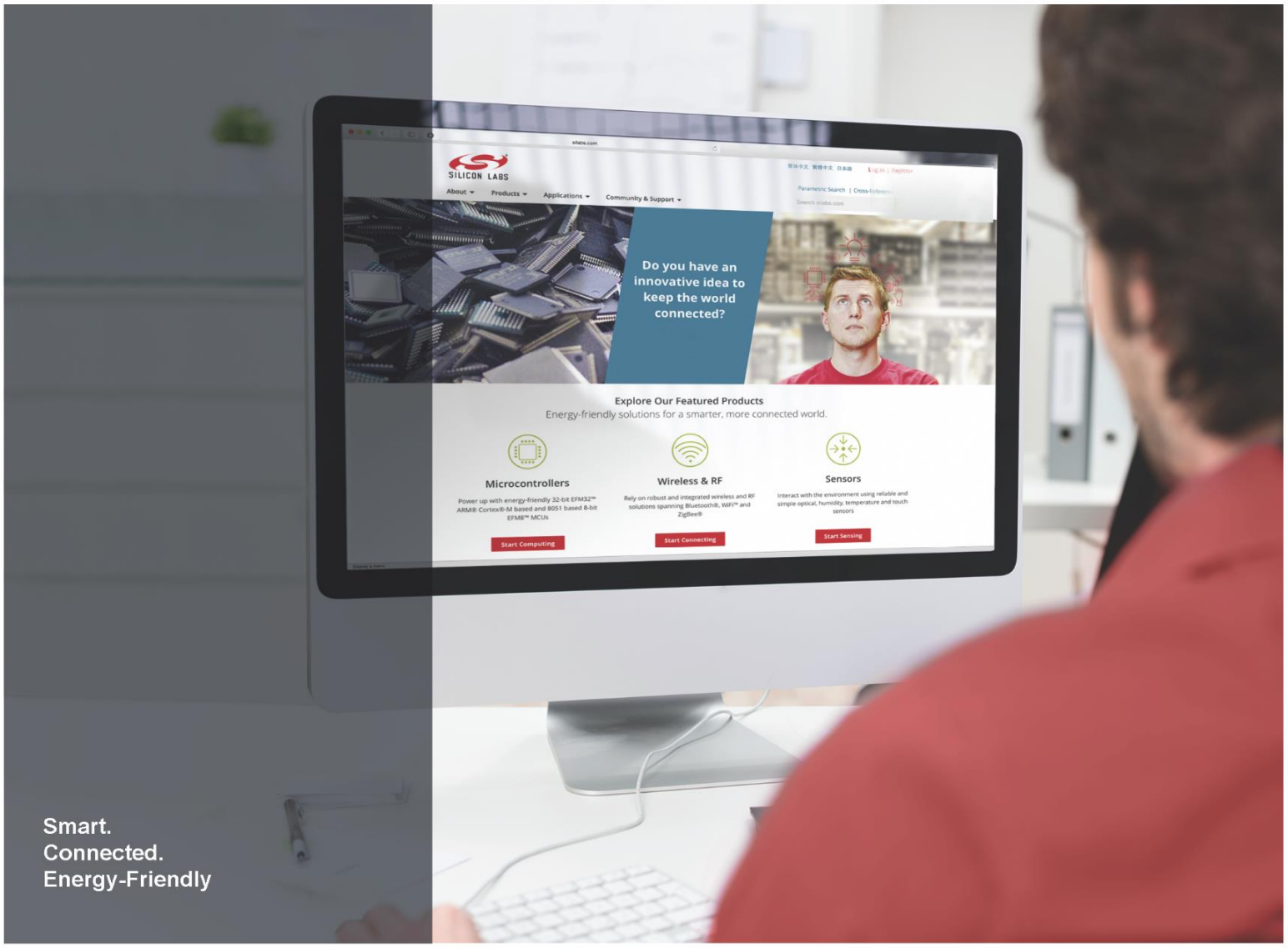
a. Make sure dev_oper_mode configured as per the mode selected. (For STA dev_oper_mode=1, For AP dev_oper_mode =1 etc)

b. If you observe unknown symbols in dmesg logs, run the below commands and reload the driver

- modprobe cfg80211
- modprobe mac80211
- modprobe bluetooth

11 Revision History

Revision No	Version No	Date	Changes
1	1.0	October, 2020	Preliminary version Updated hyperlinks. OSD specific changes.
2	1.1	October 16, 2020	Updated OSD specific changes.



Products
www.silabs.com/products

Quality
www.silabs.com/quality

Support and Community
community.silabs.com

SILICON LABS

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

http://www.silabs.com