Wlan-Station ZigBee Switch Application

User guide
Version 0.1

September 2015

Redpine Signals, Inc.

2107 N. First Street, #680

San Jose, CA 95131.

Tel: (408) 748-3385

Application Overview:

The coex application demonstrates how information can be exchanged seamlessly using two wireless protocols (WLAN and ZigBee) running in the same device.

Setup required:

1. Linux PC.
2. ZigBee Light Coordinator.
3. WLAN Access Point and a Linux PC with openssl support

Description:

The coex application has WLAN and ZigBee tasks.

ZigBee end device connects to the coordinator. The Client connects to the AP.

The socket opened at AP sends message to client which is in listen mode, Wlan task accepts and sends to ZigBee task, which in turn sends command to ZigBee Light.

On reception of data confirmation, the ZigBee task sends "RECVD CNFRM" string to the AP via client task.

Thus messages can be seamlessly transferred between Linux PC and Coordinator.

Details of the Application:

WiSeConnect/WiSeConnect Plus WLAN acts as a Station and connects to an Access Point

WiSeConnect/WiSeConnect Plus ZigBee acts as a HA switch device.

Initially, ZigBee end-device is connects to the coordinator and then command exchanges occur as follows.

- The WLAN task (running in WiSeConnect/WiSeConnect Plus device) mainly includes following steps.
    1. Connects to a Access Point
    2. Exchanges data over SSL socket with the peer(Linux PC)
- The ZigBee task (running in WiSeConnect/WiSeConnect Plus device) mainly includes following steps.
    1. Receives the command from wlan task.
    2. Sends the command to the HA light.

3. On receiving confirmation sends back confirm string to the AP.

WLAN and ZigBee tasks forever run in the application to serve the asynchronous events

Configuring the WLAN task:

Edit the **`rsi_wlan_app.c`** file in the following path to establish connection with the Access-point.

`sapis/examples/wlan_zigbee/wlan_zigbee_switch/`

1. From given configuration,

   `SSID` refers to the Access point to which user wants to connect.

   `SECURITY_TYPE` is the security type of the Access point.

   `PSK` refers to the secret key if the Access point is configured in WPA/WPA2 security modes.

```
      #define SSID                 "<ap_name>"
      #define SECURITY_TYPE         <security-type>
      #define PSK                     ""
```

2. Load the SSL CA- certificate using rsi_wlan_set_certificate API after wireless initialization.

> **Note**: rsi_wlan_set_certificate expects the certificate in the form of linear array. Python script is provided in the release package named "`certificate_script.py`" in the following path "`sapis/examples/wlan/certificates`" to convert the pem certificate into linear array
>
> Example: If the certificate is ca-cert.pem, give the command as
>
> ```
> python certificate_script.py ca-cert.pem
> ```
>
> The script will generate cacert.pem, in which one linear array named *cacert* contains the certificate

3. Enable/Disable DHCP mode

   1 – Enables DHCP mode (gets the IP from DHCP server)

   0 – Disables DHCP mode

```
      #define DHCP_MODE 1
```

4. If DHCP mode is disabled, then change the following macros to configure static IP address

IP address to be configured to the device should be in long format and in little endian byte order.

Example: To configure "192.168.10.101" as IP address, update the macro **DEVICE_IP** as **0x650AA8C0**.

```
#define  DEVICE_IP          0x650AA8C0
```

IP address of the gateway should also be in long format and in little endian byte order

Example: To configure "192.168.10.1" as Gateway, update the macro **GATEWAY** as **0x010AA8C0**

```
#define  GATEWAY            0x010AA8C0
```

IP address of the network mask should also be in long format and in little endian byte order

Example: To configure "255.255.255.0" as network mask, update the macro **NETMASK** as **0x00FFFFFF**

```
#define   NETMASK           0x00FFFFFF
```

5. To establish TCP connection and transfer data to the remote socket configure the below macros. If SSL is enabled, open the socket with protocol type as 1.

   Internal socket port number.

```
#define DEVICE_PORT 5001
```

   Port number of the remote server

```
#define SERVER_PORT 5001
```

   IP address of the remote server

```
#define SERVER_IP_ADDRESS 0x650AA8C0
```

   To enable SSL on socket creation (on device)

```
#define SSL 1
```

   Number of packets to send

```
#define NUMBER_OF_PACKETS 1000
```

   Application memory length which is required by the driver

```
#define  GLOBAL_BUFF_LEN  8000
```

6. Include rsi_wlan_app.c , rsi_zigbee_app.c and main.c files in the project, build and launch the application

7. Open SSL server socket on remote machine

   For example, to open SSL socket with port number 5001 on remote side, use the command as given below

```
openssl s_server  -accept<SERVER_PORT> -cert
<server_certificate_file_path> -key
<server_key_file_path> -tls<tls_version>
```

Example: `openssl s_server –accept 5001 –cert server-cert.pem -key server-key.pem –tls1_2`

Note: All the certificates are given in the release package path: `sapis/examples/wlan/certificates`

Then type commands

TOGGLE - To send TOGGLE command to the HA Light.

ON - To send On command to the HA Light.

OFF - To send OFF command to the HA Light.

On successfully reception of commands the HA Light replies back with "RECVD CNFRM" string.

Configuring the ZigBee Application:

Edit the **`rsi_zb_app.c`** file in the following path of the Application

`sapis/examples/wlan_zigbee/wlan_zigbee_switch/`

- Edit **`rsi_zb_app.c`** and update the following parameters of the device to establish connection with the Coordinator.

  From given configuration, **`g_CHANNEL_MASK_c`** refers to the channel in which the coordinator is operating.

  This macro defines a 32-bit mask starting from the 11th bit from

  the LSB to the 26th bit, consisting of 16 channels represented by

  each bit.

  For example : channel mask for 26th channel

```
#define g_MASK_FOR_26_CHANNEL_c  0x04000000
#define g_CHANNEL_MASK_c   g_MASK_FOR_26_CHANNEL_c
```

Run the application, the end-device connects to the coordinator and sends command to the coordinator as received from the wlan remote client.