NEWRACOM
A New semiconductor Company for New Era

NRC7292 Evaluation Kit User Guide

(Deep Sleep Mode)

Ultra-low power & Long-range Wi-Fi

Ver 1.1
May 30, 2020

NEWRACOM, Inc.

Contents

1 Introduction 5
2 Deep Sleep Operation 5
2.1 Overview 5
2.2 State operation and state transitions 6
2.3 Host power control 7
3 Deep Sleep Function Test 7
3.1 Basic setup 7
3.2 Test procedure 8
3.3 Performance Measurement 12
4 Revision history 15

List of Tables

Table 2.1 Component power on/off state 5
Table 2.2 Power control GPIO value 7
Table 3.1 Analysis of operation time 14

List of Figures

Figure 2.1 State transition diagram 6
Figure 3.1 Test system configuration........ 7
Figure 3.2 Pin connection 8
Figure 3.3 Results of running STA with $\boldsymbol{\mu}$-code enable 9
Figure 3.4 Enter the “test ucode” in the cli_app 10
Figure 3.5 Running state indicator sequence in console log 10
Figure 3.6 Wake up indicator in console log........ 10
Figure 3.7 AP send ping to the STA........ 11
Figure 3.8 GPIO pin connection 11
Figure 3.9 GPIO control from another host 12
Figure 3.10 Test sequence 12
Figure 3.11 Test interval 13
Figure 3.12 Measurement for reassociation using sniffer logs 13
Figure 3.13 Timeline for test sequence 14

1 Introduction

This document describes the optional NRC7292 STA deep sleep feature in compliance with 802.11ah requirements and specifications. The deep sleep mode drastically reduces the total power usage at the expense of increased latency. For scenarios preferring extended battery life over latency-free real-time communication, by integrating the deep sleep functionality into the existing 802.11ah network infrastructure substantially improves the overall operational performance.

2 Deep Sleep Operation

2.1 Overview

With the deep sleep feature enabled, the STA module can find itself in one of the following three states: **active mode, sleep mode, and μ-code mode** at any given point except during the transitional period between states. The table below summarizes which components are powered on or off as a function of the state:

Table 2.1 Component power on/off state

Component Type	Active Mode	Deep Sleep Mode	μ-code Mode
Raspberry Pi Host	ON	OFF	OFF
CPU	ON	OFF	ON
Peripherals	ON	OFF	ON
Modem	ON	OFF	ON
RAM	ON	OFF	ON
Retention-RAM	ON	ON	ON
32MHz Clock Oscillator	ON	OFF	ON
32.768KHz Clock Oscillator	ON	ON	ON

The arrows in the state transition diagram below characterize all possible state transitions:

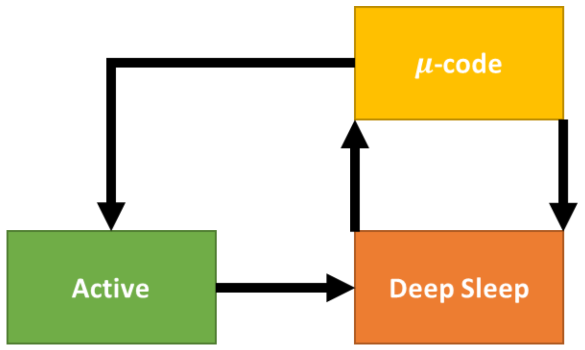


Figure 2.1 State transition diagram

2.2 State operation and state transitions

Under the active mode, the STA module can freely exchange packets with the AP. The host can optionally trigger the module to enter the deep sleep state if some pre-defined condition is satisfied. The triggering is achieved by executing './cli_app test ucode' from the script directory 'nrc_pkg/script' on the host. For example, if there is no packet exchange between the AP and the module for a fixed time interval, the host may put the module into the deep sleep mode. When the module enters deep sleep mode, many of the components such as: Raspberry Pi host, CPU, peripherals, modem, RAM switches from the regular 32MHz clock oscillator to the 32.768KHz clock oscillator, except for the retention-RAM power off component.

In deep sleep mode, the STA module is unable to exchange packets with the AP. As soon as the module enters the deep sleep mode, the module starts a timer which triggers an RTC wake-up interrupt signal upon the time expiring. The default timer duration is 1 second. The timer duration can be configured by the user according to preference. The interrupt signal initiates the transition from the deep sleep mode to the μ-code mode. The transition involves powering on all components except for Raspberry Pi host and switching back to the regular 32MHz clock oscillator.

Under the μ-code mode, the STA module still cannot freely exchange packets with the AP. By examining the traffic indication map (TIM), the module determines the existence of any packets addressed to the module in the next received long beacon. The module stays in the μ-code mode until the reception of the next long beacon. For maximal efficiency, only one channel is scanned during μ-code mode operation. If the TIM fields in the long beacon indicates the absence of a packet addressed to the module, the module re-enters the deep sleep mode. However, if the TIM fields indicate the existence of a packet addressed to the module, then the module initiates the transition to active mode. Transition from the μ-code mode to the active mode involves powering on the Raspberry Pi host. Upon boot up, the host uploads the firmware binary onto the RAM to finalize the transition.

2.3 Host power control

Table 2.2 Power control GPIO value

Mode	Active	Deep Sleep	μ-code
GPIO Value	HIGH	LOW	LOW

The external Raspberry Pi host power reset is directly controlled from the STA module by utilizing the dedicated module GPIO14 pin. The NRM7292 GPIO14 pin is connected to the reset pin of the Raspberry pi3 marked 'RUN' pulled high via a 10K resistor. It is closed when the value of the GPIO14 pin is high and it is opened when the pin value is low. Therefore, out of the three possible states that the module can be in, the GPIO value is high only in active mode. This mechanism will be illustrated further in the next chapter.

3 Deep Sleep Function Test

3.1 Basic setup

The device under test (DUT) consists of a Raspberry Pi 3B+ host and an NRC7292 module, both modules connected by one power source. It needs to connect the reset pin marked "RUN" of the Raspberry Pi3 B+ and the GPIO 14 pin of the NRC7292 module. When the NRC7292 module wakes up from sleep mode, it controls the power reset of the Raspberry Pi 3B+ via GPIO14.

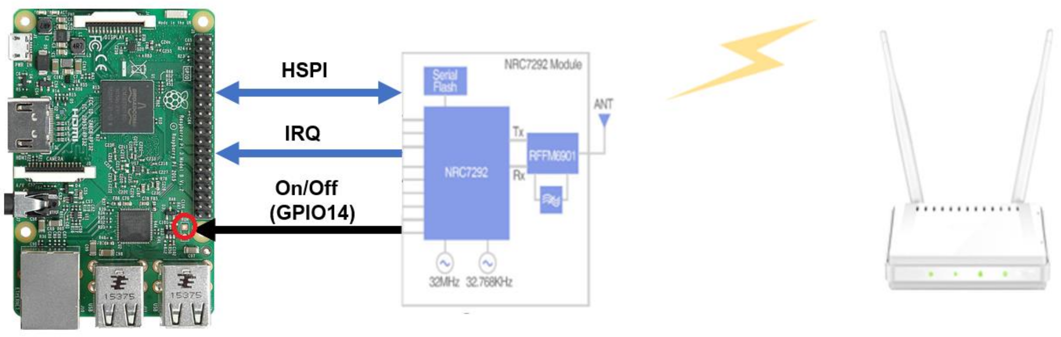


Figure 3.1 Test system configuration

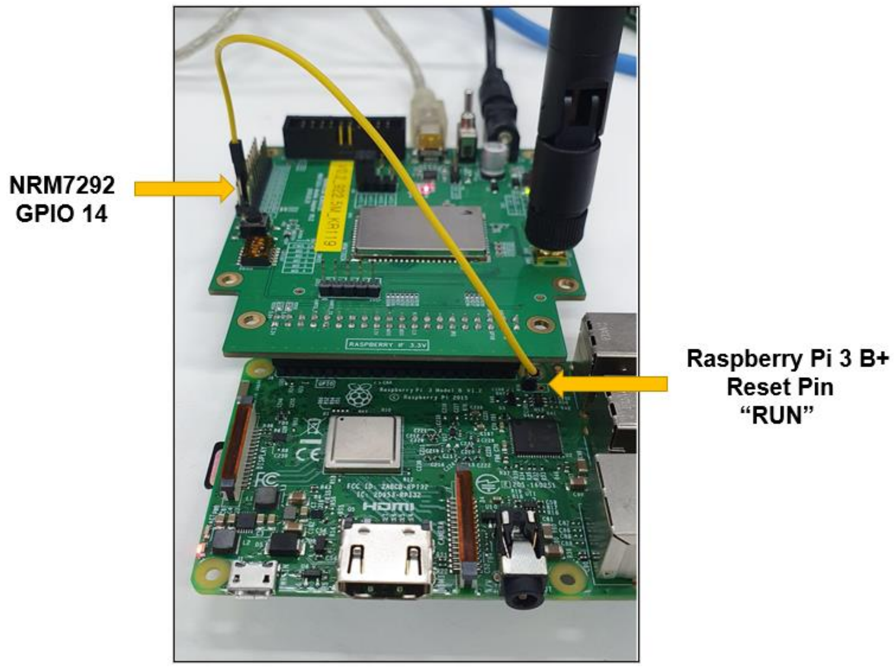


Figure 3.2 Pin connection

3.2 Test procedure

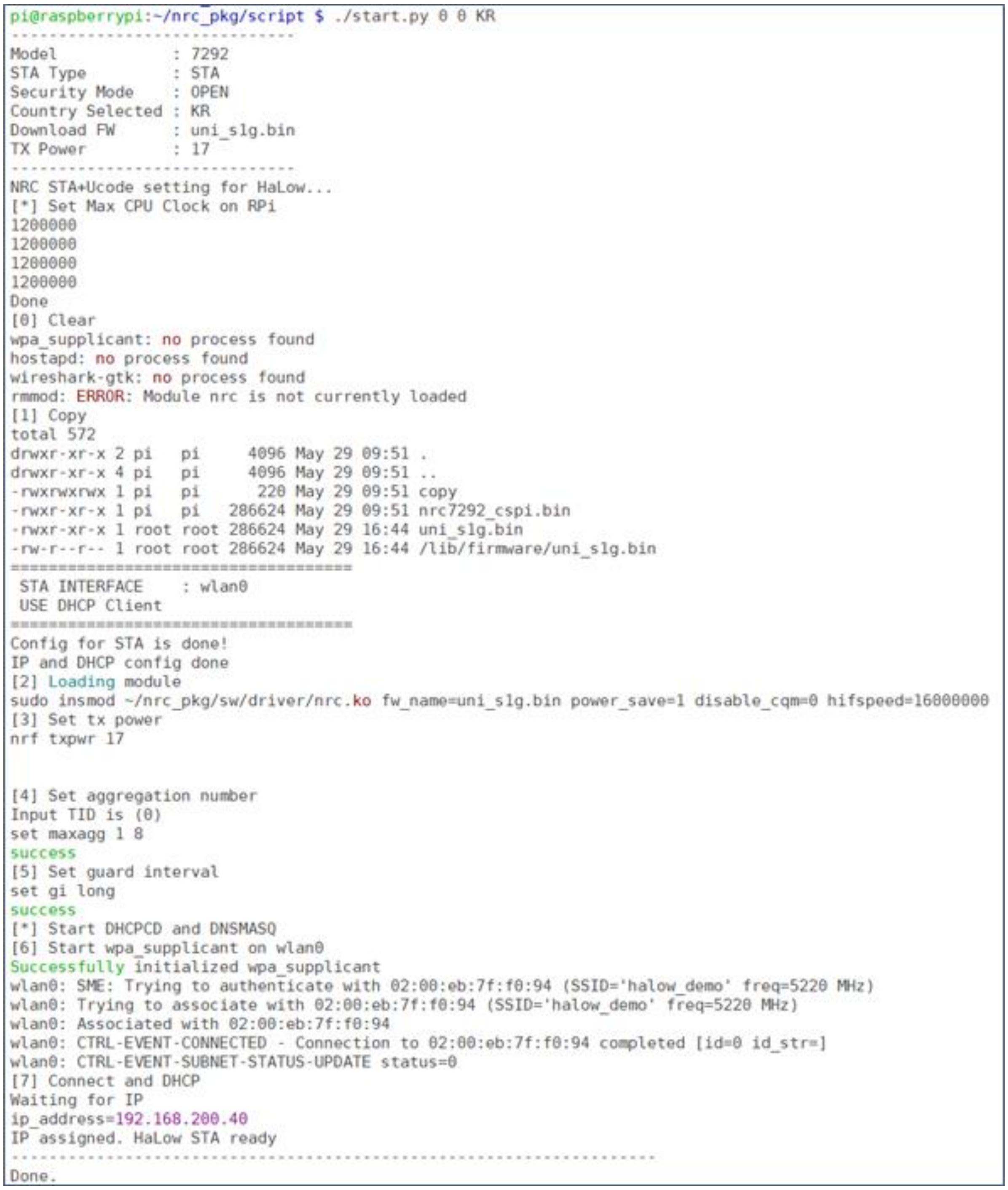
3.2.1 Running the module as a STA with μ-code enabled

1) **Running the start script**

The script 'start.py' in the package directory '*nrc_pkg/script*' is used to start the module operation either as an AP or a STA. The script takes three arguments '*./start.py <sta_type> <security_mode> <country>*' To run the module as an STA, the value 0 must be provided to the ***sta_type*** (0: STA, 1: AP, 2: Sniffer, 3: Relay) argument. The ***security_mode*** (0: Open, 1: WPA2-PSK, 2: WPA3-OWE, 3: WPA3-SAE), ***country*** (US, JP, TW, KR, EU, CN).

2) Checking the start script log

The start script sequentially runs various procedures including inserting the module, copying the firmware binary, configuring the module, and starting the *wpa supplicant* process. If all procedures run successfully, the '*ifconfig*' log shown at the end of the script will display an IP dynamically assigned by the AP to the wlan0 interface.



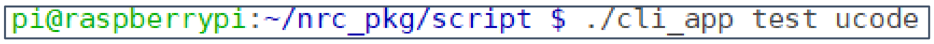
```
pi@raspberrypi:~/nrc_pkg/script $ ./start.py 0 0 KR
------------------------------
Model            : 7292
STA Type         : STA
Security Mode    : OPEN
Country Selected : KR
Download FW      : uni_s1g.bin
TX Power         : 17
------------------------------
NRC STA+Ucode setting for HaLow...
[*] Set Max CPU Clock on RPi
1200000
1200000
1200000
1200000
Done
[0] Clear
wpa_supplicant: no process found
hostapd: no process found
wireshark-gtk: no process found
rmmod: ERROR: Module nrc is not currently loaded
[1] Copy
total 572
drwxr-xr-x 2 pi   pi     4096 May 29 09:51 .
drwxr-xr-x 4 pi   pi     4096 May 29 09:51 ..
-rwxrwxrwx 1 pi   pi      220 May 29 09:51 copy
-rwxr-xr-x 1 pi   pi   286624 May 29 09:51 nrc7292_cspi.bin
-rwxr-xr-x 1 root root 286624 May 29 16:44 uni_s1g.bin
-rw-r--r-- 1 root root 286624 May 29 16:44 /lib/firmware/uni_s1g.bin
====================================
 STA INTERFACE    : wlan0
 USE DHCP Client
====================================
Config for STA is done!
IP and DHCP config done
[2] Loading module
sudo insmod ~/nrc_pkg/sw/driver/nrc.ko fw_name=uni_s1g.bin power_save=1 disable_cqm=0 hifspeed=16000000
[3] Set tx power
nrf txpwr 17


[4] Set aggregation number
Input TID is (0)
set maxagg 1 8
success
[5] Set guard interval
set gi long
success
[*] Start DHCPCD and DNSMASQ
[6] Start wpa_supplicant on wlan0
Successfully initialized wpa_supplicant
wlan0: SME: Trying to authenticate with 02:00:eb:7f:f0:94 (SSID='halow_demo' freq=5220 MHz)
wlan0: Trying to associate with 02:00:eb:7f:f0:94 (SSID='halow_demo' freq=5220 MHz)
wlan0: Associated with 02:00:eb:7f:f0:94
wlan0: CTRL-EVENT-CONNECTED - Connection to 02:00:eb:7f:f0:94 completed [id=0 id_str=]
wlan0: CTRL-EVENT-SUBNET-STATUS-UPDATE status=0
[7] Connect and DHCP
Waiting for IP
ip_address=192.168.200.40
IP assigned. HaLow STA ready
--------------------------------------------------------------------
Done.
```

Figure 3.3 Results of running STA with μ-code enable

3.2.2 μ-code mode console log

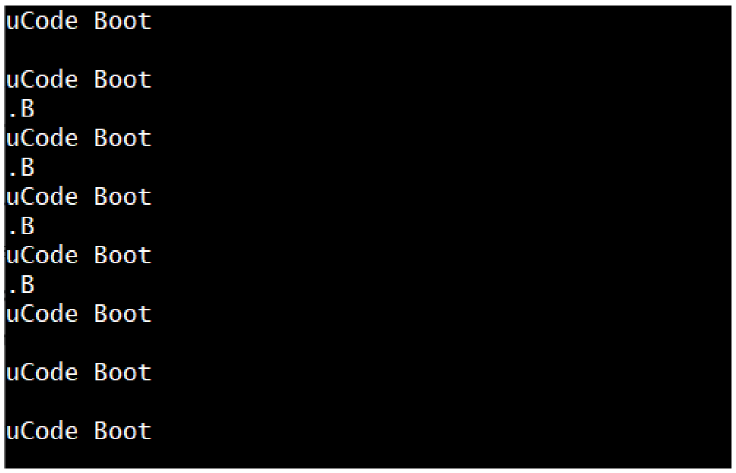
For entering the μ-code mode, "./cli_app test ucode" is run in host. cli_app is under 'nrc_pkg/script/'.



```
pi@raspberrypi:~/nrc_pkg/script $ ./cli_app test ucode
```

Figure 3.4 Enter the "test ucode" in the cli_app

As soon as entering the μ-code mode, the UART console will start displaying a running state indicator character sequence. Each character in the running sequence can be either "uCode boot", "B" or ".". The character "uCode boot" indicates entering the μ-code mode, "B" indicates the reception of a short or long beacon and "." indicates the reception of any other packets. As explained in the previous chapter, the μ-code mode waits until the reception of the next long beacon. After examining the TIM fields in the long beacon, the module state conditionally transitions to either active mode or deep sleep mode. The reception of short beacons does not trigger state transitions and is indicated in the running character sequence. The character "WakeUp" indicates waking up from deep sleep mode.



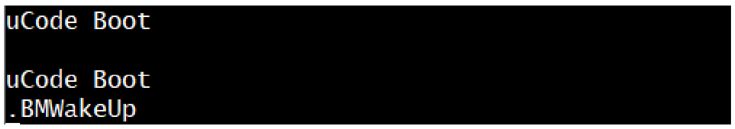
```
uCode Boot

uCode Boot
.B
uCode Boot
.B
uCode Boot
.B
uCode Boot
.B
uCode Boot

uCode Boot

uCode Boot
```

Figure 3.5 Running state indicator sequence in console log



```
uCode Boot

uCode Boot
.BMWakeUp
```

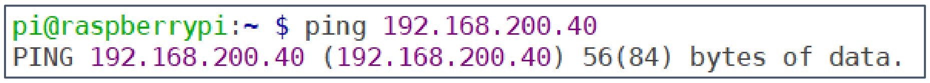
Figure 3.6 Wake up indicator in console log

3.2.3 Wake up sources

NRC7292 supports 3 kinds of wakeup source in host mode. There are RTC, GPIO, HSPI.

3.2.3.1 Ping from AP

When the user set the RTC for wakeup source, the target could wakeup from deep sleep mode by TIM. The AP send a ping to the STA and wakes the host. When the NRC7292 receives TIM and decodes it and receives a ping packet from the AP, the NRC7292 powers on the Raspberry Pi host through the reset pin marked "RUN".



```
pi@raspberrypi:~ $ ping 192.168.200.40
PING 192.168.200.40 (192.168.200.40) 56(84) bytes of data.
```

Figure 3.7 AP send ping to the STA

3.2.3.2 GPIO Wakeup pin

When the user set the GPIO for wakeup source, the GPIO interrupt serves as a wake-up source for waking up the CPU from deep sleep mode. In the NRC7292 module, the GPIO 15 pin is basically used as a wake-up source, and when the GPIO 15 pin is triggered as Active High, it wakes up from deep sleep mode. Prepare another Raspberry Pi to control GPIO to interrupt the GPIO module of the NRC7292 module. Connect the GPIO21 pin of the Raspberry Pi3 to GPIO Control and the GPIO 15 pin of the NRC7292 module. And connect the GND pin between the two boards to match the ground voltage.

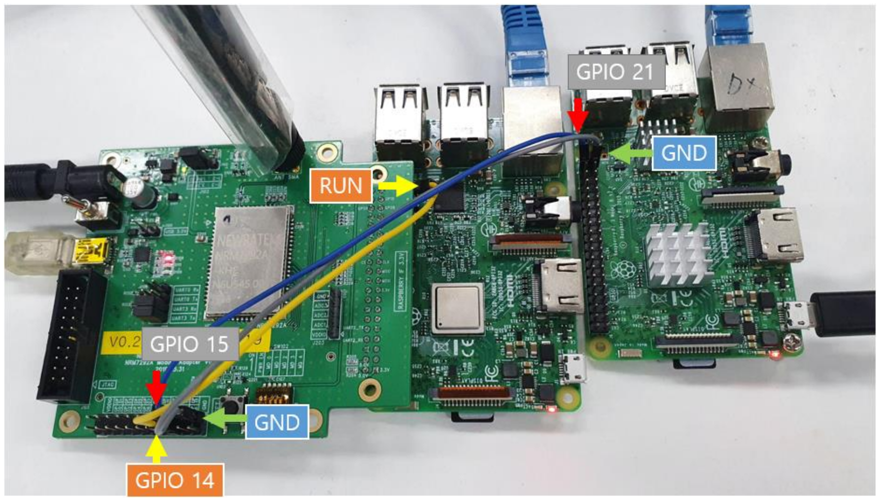
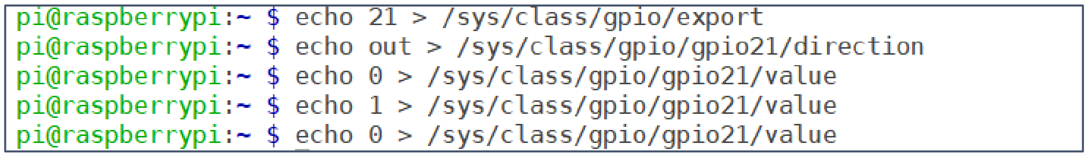


Figure 3.8 GPIO pin connection



```
pi@raspberrypi:~ $ echo 21 > /sys/class/gpio/export
pi@raspberrypi:~ $ echo out > /sys/class/gpio/gpio21/direction
pi@raspberrypi:~ $ echo 0 > /sys/class/gpio/gpio21/value
pi@raspberrypi:~ $ echo 1 > /sys/class/gpio/gpio21/value
pi@raspberrypi:~ $ echo 0 > /sys/class/gpio/gpio21/value
```

Figure 3.9 GPIO control from another host

3.2.3.3 Data from Host

When the user set the HSPI for wakeup source, the HSPI interrupt serves as a wake-up source for waking up the CPU from deep sleep mode.

3.2.4 Starting host booting

Host boot up is initiated. Firmware is downloaded from host driver. STA is associated with the AP through 1 channel scanning.

3.3 Performance Measurement

3.3.1 Test configuration

Test is performed in channel (915MHz center frequency and 2MHz bandwidth) with WPA2 security mode.

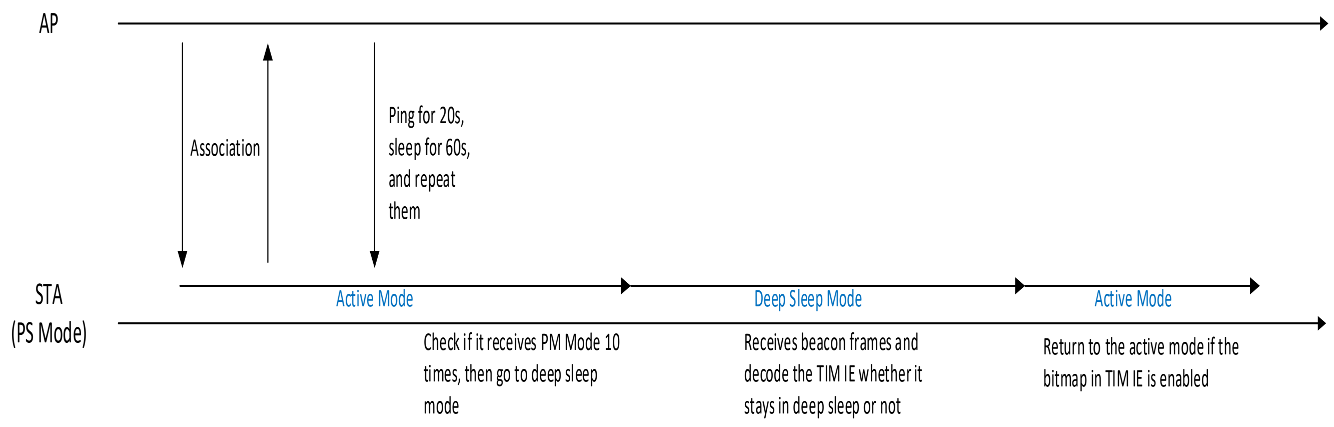

AP
Association
Ping for 20s, sleep for 60s, and repeat them
STA (PS Mode)
Active Mode
Deep Sleep Mode
Active Mode
Check if it receives PM Mode 10 times, then go to deep sleep mode
Receives beacon frames and decode the TIM IE whether it stays in deep sleep or not
Return to the active mode if the bitmap in TIM IE is enabled

Figure 3.10 Test sequence

3.3.2 Test measurement method

Test result is measured by oscilloscope and sniffer. To measure the interval by oscilloscope, we set the GPIO pin to 1, 0 and 1 in sequence. In doing so, the interval we intend to measure will equal to the interval while the pin is set to 0.

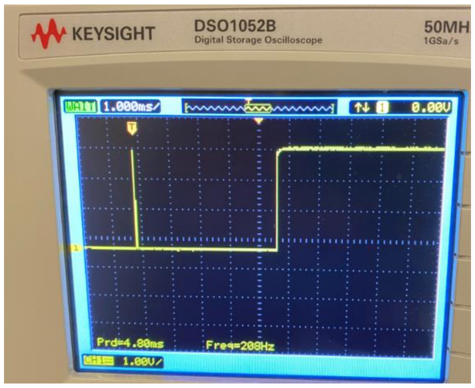
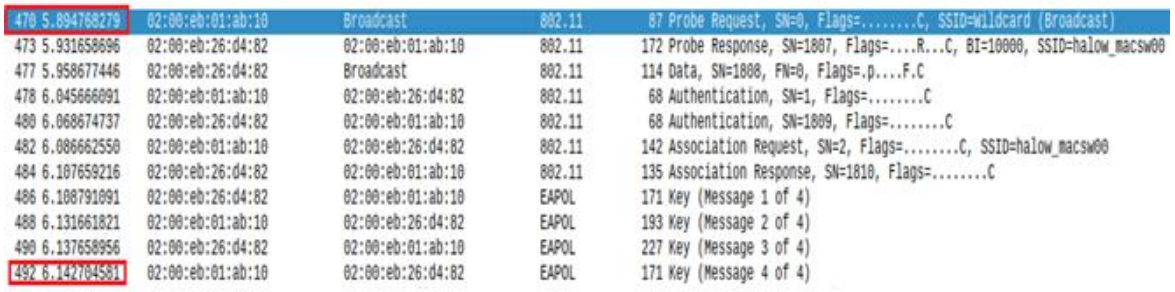


Figure 3.11 Test interval

To measure the reassociation, we use a Sniffer to measure the intervals between the probe request and the last packet in a 4-way handshake.



```
470 5.894768279  02:00:eb:01:ab:10  Broadcast          802.11   87 Probe Request, SN=0, Flags=........C, SSID=Wildcard (Broadcast)
473 5.931658696  02:00:eb:26:d4:82  02:00:eb:01:ab:10  802.11  172 Probe Response, SN=1807, Flags=....R...C, BI=10000, SSID=halow_macsw00
477 5.958677446  02:00:eb:26:d4:82  Broadcast          802.11  114 Data, SN=1808, FN=0, Flags=.p....F.C
478 6.045666091  02:00:eb:01:ab:10  02:00:eb:26:d4:82  802.11   68 Authentication, SN=1, Flags=........C
480 6.068674737  02:00:eb:26:d4:82  02:00:eb:01:ab:10  802.11   68 Authentication, SN=1809, Flags=........C
482 6.086662550  02:00:eb:01:ab:10  02:00:eb:26:d4:82  802.11  142 Association Request, SN=2, Flags=........C, SSID=halow_macsw00
484 6.107659216  02:00:eb:26:d4:82  02:00:eb:01:ab:10  802.11  135 Association Response, SN=1810, Flags=........C
486 6.108791091  02:00:eb:26:d4:82  02:00:eb:01:ab:10  EAPOL   171 Key (Message 1 of 4)
488 6.131661821  02:00:eb:01:ab:10  02:00:eb:26:d4:82  EAPOL   193 Key (Message 2 of 4)
490 6.137658956  02:00:eb:26:d4:82  02:00:eb:01:ab:10  EAPOL   227 Key (Message 3 of 4)
492 6.142704581  02:00:eb:01:ab:10  02:00:eb:26:d4:82  EAPOL   171 Key (Message 4 of 4)
```

Figure 3.12 Measurement for reassociation using sniffer logs

3.3.3 Timeline during test sequence

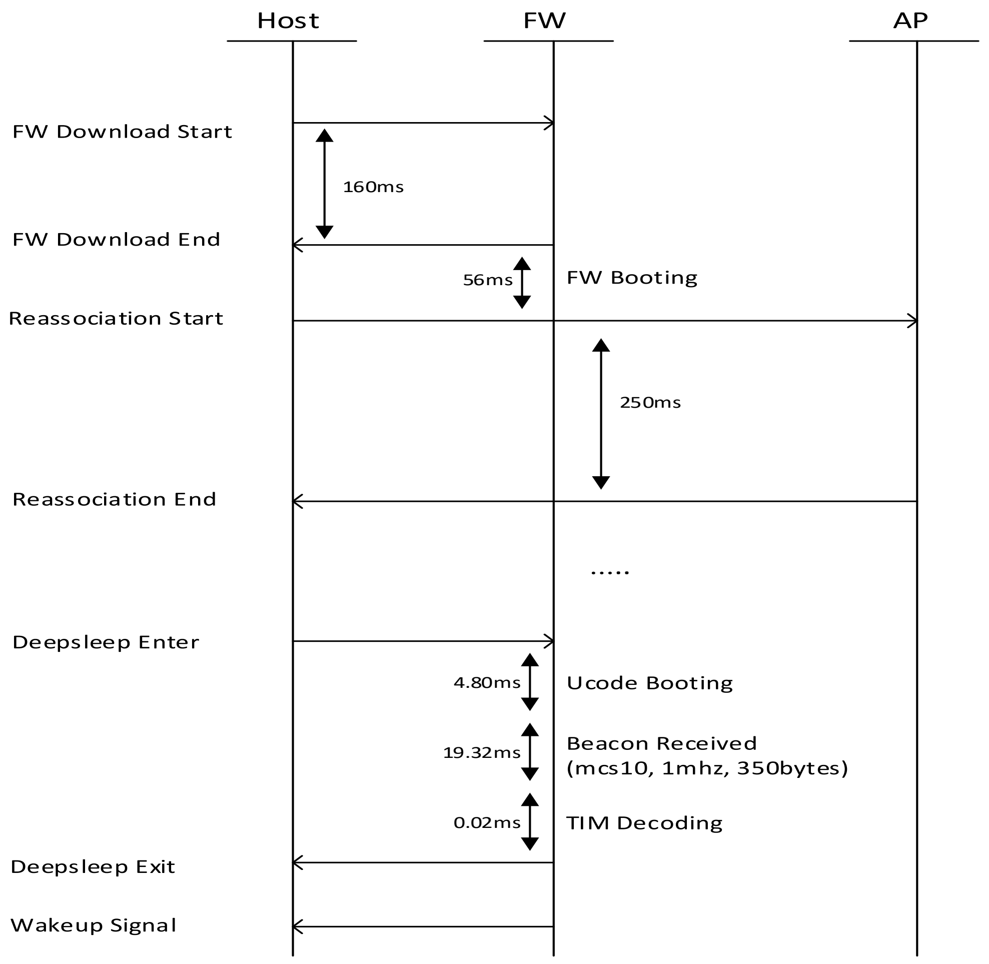


Figure 3.13 Timeline for test sequence

3.3.4 Test results

Table 3.1 Analysis of operation time

Name	Time (ms)	Not
Firmware Download	160	
Firmware Booting	56	without log message
Association	250	Between AP and STA
u-code booting	4.8	
beacon decoding	19.32	MCS10, 1M BW, 350 Bytes
Total	490.12	

If log message is enabled, the firmware booting duration will take approximately 472 ms instead of 56 ms. The amount of time to boot up Linux and load wpa supplicant is excluded from the measurement.

4 Revision history

Revision No	Date	Comments
Ver 1.0	2/10/2020	Initial version for customer release created
Ver 1.1	5/29/2020	Add Wake-up source