A Brief Introduction to USB Driver Power Saving

DISCLAIMER

Realtek provides this document "as is", without warranty of any kind, neither expressed nor implied, including, but not limited to, the particular purpose. Realtek may make improvements and/or changes in this document or in the product described in this document at any time. This document could include technical inaccuracies or typographical errors.

TRADEMARKS

Realtek is a trademark of Realtek Semiconductor Corporation. Other names mentioned in this document are trademarks/registered trademarks of their respective owners.

CONFIDENTIALITY

This document is confidential and should not be provided to a third-party without the permission of Realtek Semiconductor Corporation.

COPYRIGHT

Realtek Semiconductor Corp.
No. 2, Innovation Road II, Hsinchu Science Park, Hsinchu 300, Taiwan
Tel.: +886-3-578-0211. Fax: +886-3-577-6047
www.realtek.com.tw

Subject	Power saving configuration of USB wireless LAN adapter in Linux platform

Content

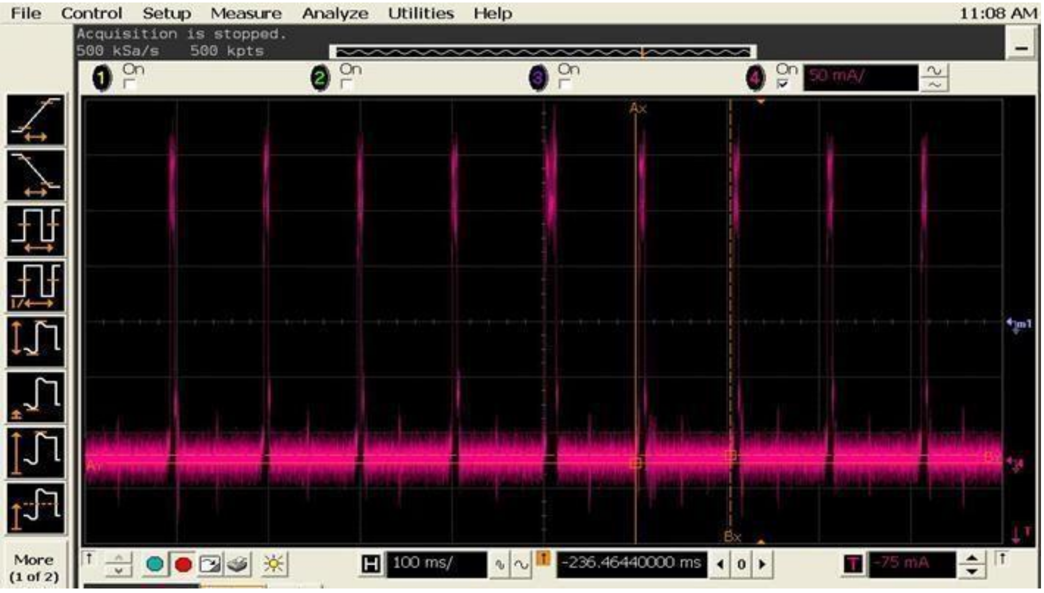
1. Revision History

Revision	Date	Editor	Note
0.1	2016/7/28	MH/Emerson	First create
0.2	2016/8/19	Emerson	Add Max-PS with 32K mode
0.3	2016/9/29	Emerson	Revise auto-suspend section

2. Power Saving mode

2.1. Minimum Power Saving mode

In this PS mode, Realtek WLan device wakes up every beacon interval. Figure 1 shows the Realtek WLan device power consumption under Minimum Power Saving mode.



Acquisition is stopped.
500 kSa/s 500 kpts

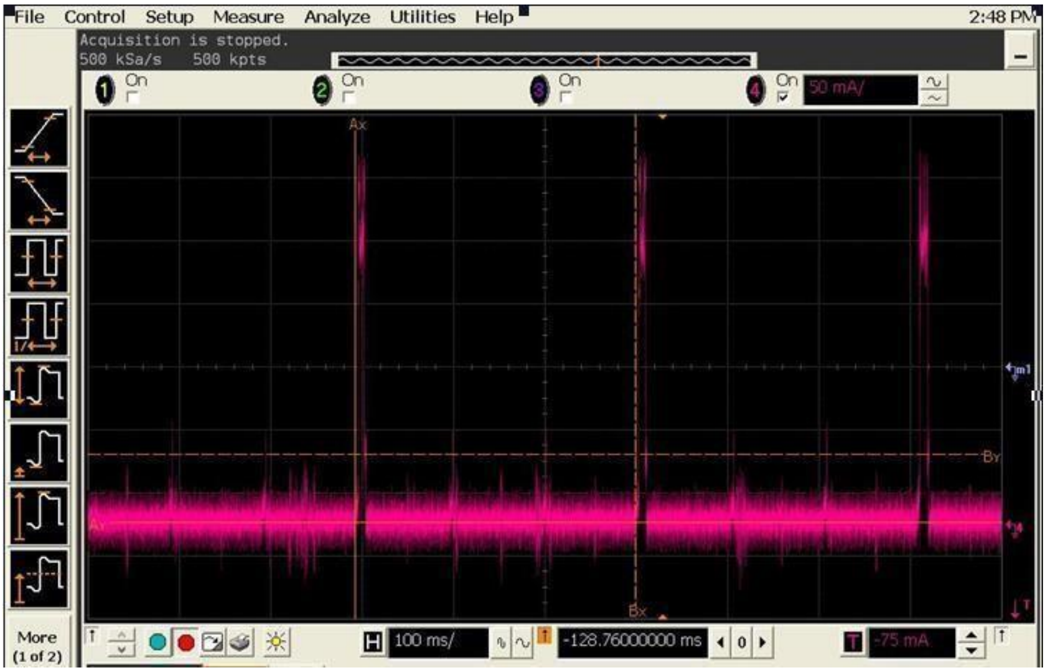
100 ms/
-236.46440000 ms
-75 mA

Figure 1. Minimum Power Saving mode (Associated idle)

2.2. Maximum Power Saving mode

When driver be set to maximum power saving mode, which means Realtek WLan device wakes up to receive beacon when DTIM count of beacon is zero.

If DTIM period of beacon is **3** Realtek WLan device would wake up every **3** bacon interval. Figure 2 illustrates Realtek WLan device power consumption under Maximum Power Saving mode:



Acquisition is stopped.
500 kSa/s 500 kpts

50 mA/

H 100 ms/ -128.76000000 ms 0

T -75 mA

Figure 2. Maximum Power Saving mode (Associated idle)

3. Driver Configuration

The following sections will describe the way to let this driver to enable the power saving functionality

```
==============================================================
        A-1.) Power Saving Mode – by modify Makefile
==============================================================
```

You can easy set the power saving to ***Min-PS*** (rtw_power_mgnt=1) by modify the **CONFIG_POWER_SAVING** to **y**, at Makefile file.

Ex:

CONFIG_POWER_SAVING = y

```
  ============================================================
        A-2.) Power Saving Mode – by change load-time parameters
==============================================================
(1) In order to enter PS Mode, you need to add the parameter of
    "rtw_power_mgnt=1" when executing "insmod 8XXXXu.ko" :
      $>insmod 8XXXXu.ko rtw_power_mgnt=1  or
      $>insmod 8XXXXu.ko rtw_power_mgnt=2

(2)Parameter Notes:
            rtw_power_mgnt=1 ;//enable PS, MIN_PS Mode
            rtw_power_mgnt=2 ;//enable PS, MAX_PS Mode

==============================================================
        A-3.) Power Saving Mode – by modify autoconf.h
```

===

You can set the power saving to ***Max-PS with 32K*** by remove the comment around **CONFIG_LPS_LCLK** definition, at include/autoconf.h file.

Ex:

```
#if defined(CONFIG_LPS)
	#define CONFIG_LPS_LCLK
#endif
```

==

B-1.) USB Autosuspend (Selective Suspend) - by modify Makefile

===

You can turn on USB autosuspend function by modify the **CONFIG_USB_AUTOSUSPEND** to **y**, at Makefile file.

Ex:

```
CONFIG_USB_AUTOSUSPEND  =  y
```

If you want to turn on USB autosuspend function under association status, **CONFIG_WOWLAN_AUTOSUSPEND, CONFIG_WOWLAN**, and **CONFIG_DEFAULT_PATTERNS_EN** also need to modify to **y** at Makefile**.**.

Ex:

```
CONFIG_USB_AUTOSUSPEND  =  y
CONFIG_WOWLAN  =  y
CONFIG_DEFAULT_PATTERNS_EN  =  y
CONFIG_WOWLAN_AUTOSUSPEND  =  y
```

==

B-2.) USB Autosuspend - by change load-time parameters

===

(1) In PS mode, you can turn on or turn off the USB autosuspend functionality by adding the following parameter.

```
 $>insmod 8XXXXu.ko rtw_power_mgnt=1 rtw_enusbss = 1
or
 $>insmod 8XXXXu.ko rtw_power_mgnt=1 rtw_enusbss = 0
or
  $>insmod 8XXXXu.ko rtw_power_mgnt=2 rtw_enusbss = 1
         or
 $>insmod 8XXXXu.ko rtw_power_mgnt=2 rtw_enusbss = 0
```

(2)Parameter Notes:

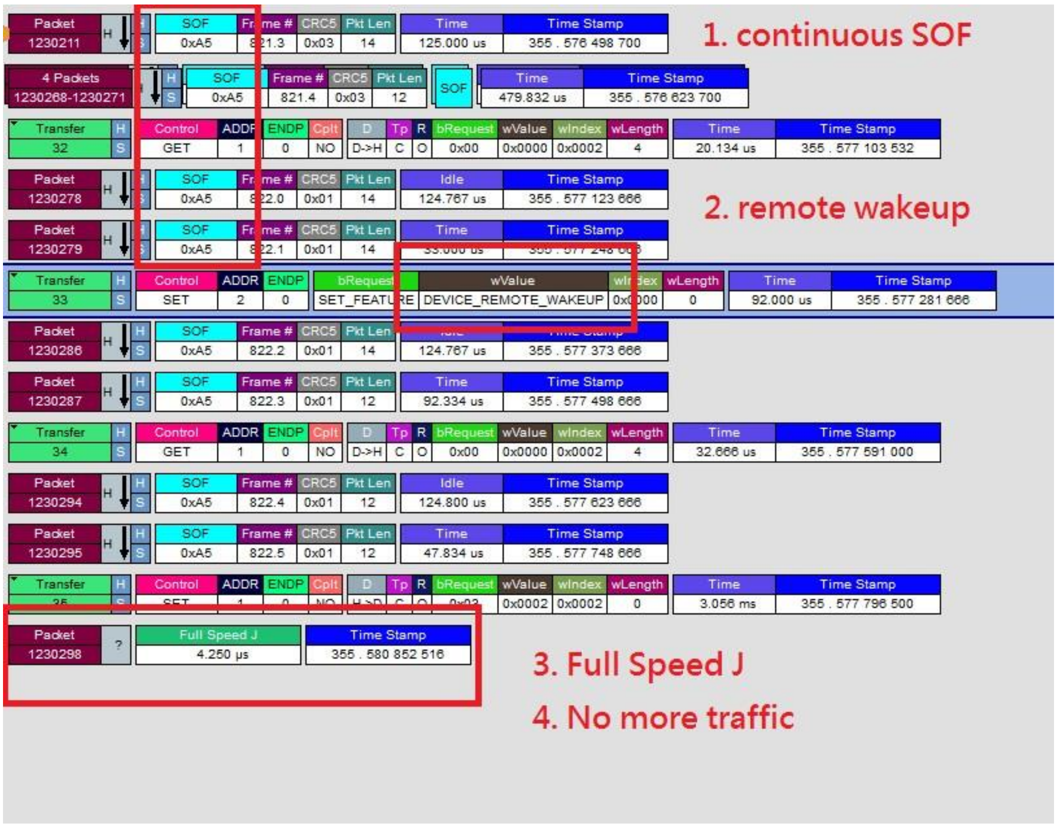
```
		rtw_enusbss = 1; // enable USB autosuspend
		rtw_enusbss = 0; // default, disable USB autosuspend
```

==

B-3.) USB Autosuspend – More information

===

(1)	CAT-C Log for Selective Suspend Check (rtw_enusbss = 1):



```
Packet 1230211     H | SOF 0xA5 | Frame # 821.3 | CRC5 0x03 | Pkt Len 14 | Time 125.000 us | Time Stamp 355 . 576 498 700
4 Packets 1230268-1230271  H S | SOF 0xA5 | Frame # 821.4 | CRC5 0x03 | Pkt Len 12 | SOF | Time 479.832 us | Time Stamp 355 . 576 623 700
Transfer 32  H S | Control GET | ADDR 1 | ENDP 0 | Cplt NO | D D->H | Tp C | R O | bRequest 0x00 | wValue 0x0000 | wIndex 0x0002 | wLength 4 | Time 20.134 us | Time Stamp 355 . 577 103 532
Packet 1230278     H | SOF 0xA5 | Frame # 822.0 | CRC5 0x01 | Pkt Len 14 | Idle 124.767 us | Time Stamp 355 . 577 123 666
Packet 1230279     H | SOF 0xA5 | Frame # 822.1 | CRC5 0x01 | Pkt Len 14 | Time 33.000 us | Time Stamp 355 . 577 248 666
Transfer 33  H S | Control SET | ADDR 2 | ENDP 0 | bRequest SET_FEATURE | wValue DEVICE_REMOTE_WAKEUP | wIndex 0x0000 | wLength 0 | Time 92.000 us | Time Stamp 355 . 577 281 666
Packet 1230286     H S | SOF 0xA5 | Frame # 822.2 | CRC5 0x01 | Pkt Len 14 | Idle 124.767 us | Time Stamp 355 . 577 373 666
Packet 1230287     H S | SOF 0xA5 | Frame # 822.3 | CRC5 0x01 | Pkt Len 12 | Time 92.334 us | Time Stamp 355 . 577 498 666
Transfer 34  H S | Control GET | ADDR 1 | ENDP 0 | Cplt NO | D D->H | Tp C | R O | bRequest 0x00 | wValue 0x0000 | wIndex 0x0002 | wLength 4 | Time 32.666 us | Time Stamp 355 . 577 591 000
Packet 1230294     H S | SOF 0xA5 | Frame # 822.4 | CRC5 0x01 | Pkt Len 12 | Idle 124.800 us | Time Stamp 355 . 577 623 666
Packet 1230295     H S | SOF 0xA5 | Frame # 822.5 | CRC5 0x01 | Pkt Len 12 | Time 47.834 us | Time Stamp 355 . 577 748 666
Transfer 35  H S | Control SET | ADDR 1 | ENDP 0 | Cplt NO | D H->D | Tp C | R O | bRequest 0x03 | wValue 0x0002 | wIndex 0x0002 | wLength 0 | Time 3.056 ms | Time Stamp 355 . 577 796 500
Packet 1230298     ? | Full Speed J 4.250 μs | Time Stamp 355 . 580 852 516
```

1. continuous SOF

2. remote wakeup

3. Full Speed J

4. No more traffic

(2) Power Consumption:
When USB selective suspend is enabled, the unassociated idle mode power consumption should be less than 2.5mA.

(3) Block USB autosuspend mode in diver run time
You can block/unblock USB autosuspend by echo the proc entry of autosuspnd to acquire a period that the USB bus will never enter USB autosuspend mode.

EX:

```
echo 1 > /proc/net/rtl8xxxyy/wlanX/auto_suspend_info  // block autosuspend
echo 0 > /proc/net/rtl8xxxyy/wlanX/auto_suspend_info  // unblock autosuspend
```

(4) Show USB autosuspend information
You can get currently information of USB autosuspend function by cat the proc entry of autosuspnd.

EX:

```
cat /proc/net/rtl8xxxyy/wlanX/auto_suspend_info
```

==

C-1.) RF on/off detection – by modify Makefile

==

You can turn on this function by modify the **CONFIG_HW_PWRP_DETECTION** to **y**, at Makefile file.
Ex:

CONFIG_HW_PWRP_DETECTION = y

==

C-2.) RF on/off detection – by change load-time parameters

==

(1) If you want turn on this function in load time, you need to add the parameter of " hwpwrp_detect =1" when executing "insmod 8XXXXu.ko" :
$>insmod 8XXXXu.ko rtw_hwpwrp_detect =1

(2)Parameter Notes: rtw_hwpwrp_detect =1 ;//enable hw power pin detection rtw_hwpwrp_detect =0 ;//disable hw power pin detection

==

D.) Two Levels of Unassociated Idle

==

In unassociated idle state, we can choose 2 levels of power saving as the default setting: IPS_NORMAL and IPS_LEVEL_2. IPS_NORMAL has lower power consumption, but needs to take more time than IPS_LEVEL_2 to wake up, while the power consumption of IPS_LEVEL_2 is higher than IPS_NORMAL about 10 mA but is faster to be woken up. IPS_LEVEL_2 is suitable for users who care the transition speed more than power consumption especially when using wifi on low speed IO interface such as USB1.1.

By default, the unassociated idle is IPS_NORMAL, You can change the default setting to IPS_LEVEL_2 with two ways:

1. Compilation time:
Enabling flag “CONFIG_IPS_LEVEL2” in Include/autoconf.h

```
#ifdef CONFIG_IPS
#define CONFIG_IPS_LEVEL2
#endif
```

2. Driver module insertion time: insmod Insert module with “rtw_ips_mode=1”
$>insmod 8XXXXu.ko rtw_power_mgnt=1 rtw_ips_mode=1