

# WIFI RF Output Power Control\_ver01

---

SEONGJI

September 10, 2019

## Contents

Revision history.....	2
Overview .....	3
Control Rule .....	4
Example Source for Update .....	5

Revision history

Revision	Date	Description
ver01	2019.09.10	Initial release

## Overview

- If it have WIFI version 3 or higher, you can modify the Tx power table.
  - WIFI firmware : development/sigfox\_cfg2/tools/wifitools  
[https://github.com/seongji-SFM/SFM\\_15x](https://github.com/seongji-SFM/SFM_15x)
  - It can get the WIFI version by using cWifi\_get\_version\_info() function.
- The following AT commands are used.
  - AT+TXPWRITE="3B3B3B3B3838"

A total of six values can be controlled. : NV1,NV2,NV3, NV4,NV5,NV6

**It is recommended to correct only three values "NV1 ~ 3".**

AT+TXPWRITE	47	47	47	3B	38	38
Index	NV1	NV2	NV3	DO NOT CHANGE		
WiFi Modulation	11b (1 ~ 11Mbps) 11g (6 ~ 12Mbps) 11n (MCS0 ~ MCS1)	11g (18 ~ 24Mbps) 11n (MCS2 ~ MCS3)	11g (36Mbps) 11n (MCS4)	11g (48 ~ 54Mbps) 11n (MCS5 ~ MCS7)	WiFi Channel Number "1"	Channel High Group CE : 11 ~ 13 CH FCC/IC : 11CH TELEC : 11 ~ 14CH
Current WiFi RF power (CE)	0x3B	0x3B	0x3B	0x3B	0x38	0x38
Ex) If you want to raise 3dB (CE)	0x47	0x47	0x47	0x3B	0x38	0x38

\* Only NV1, NV2, and NV3 values should be changed. NV4, NV5, and NV6 should not be changed to fixed values. If you change this value, EVM performance of 11n, MCS7 decreases.

- See WIFI\_TX\_POWER\_TABLES\_UPDATE\_ENABLE in cfg\_board.c.
- Available WIFI Channels

CE (R1) : 1~13 (ETSI)

FCC (R2, R4) : 1~11 (USA Canada)

TELEC (R3) : 1~14 (Japan)

Channel	Center Frequency	USA Canada	ETSI	Japan
1	2412 MHz	X	X	X
2	2417 MHz	X	X	X
3	2422 MHz	X	X	X
4	2427 MHz	X	X	X
5	2432 MHz	X	X	X
6	2437 MHz	X	X	X
7	2442 MHz	X	X	X
8	2447 MHz	X	X	X
9	2452 MHz	X	X	X
10	2457 MHz	X	X	X
11	2462 MHz	X	X	X
12	2467 MHz		X	X
13	2472 MHz		X	X
14	2484 MHz			X

## Control Rule

11b		11g		11n		
Parameter	Data rate	Parameter	Data rate	Parameter	Data rate	
0x0	1 Mbps	0xb	6 Mbps	0x10	6.5 Mbps / MCS0	NV1
0x1	2 Mbps	0xf	9 Mbps	0x11	13 Mbps / MCS1	
0x2	5.5 Mbps	0xa	12 Mbps	0x12	19.5 Mbps / MCS2	NV2
0x3	11 Mbps	0xe	18 Mbps	0x13	26 Mbps / MCS3	
-	-	0x9	24 Mbps	0x14	39 Mbps / MCS4	NV3
-	-	0xd	36 Mbps	0x15	52 Mbps / MCS5	NV4
-	-	0x8	48 Mbps	0x16	58.5 Mbps / MCS6	
-	-	0xc	54 Mbps	0x17	65 Mbps / MCS7	

NV5 : WIFI Channel No 1

NV6 : WIFI Channel No 11~ (eg. ch 11, ch 12, ch 13, ch 14)

- 0x52(20.5) is the maximum value.  
Decreasing 1 will reduce 0.25DBm
- The following formula must be satisfied to ensure normal operation.  
 $NV1 \geq NV2 \geq NV3 \geq NV4 \geq NV5 \geq NV6$
- It is recommended to modify only NV1 ~ 3.

## Example Source for Update

```

const uint8_t wifi_tx_power_table_CE[6] = {0x45, 0x3F, 0x4D, 0x3b, 0x38, 0x38}; //default 0x3b, 0x3b, 0x3b, 0x3b, 0x38, 0x38
const uint8_t wifi_tx_power_table_FCC[6] = {0x44, 0x3E, 0x3C, 0x3a, 0x32, 0x2e }; //default 0x3a, 0x3a, 0x3a, 0x3a, 0x32, 0x2e
const uint8_t wifi_tx_power_table_TELECOM[6] = {0x46, 0x40, 0x3E, 0x3c, 0x3c, 0x3c }; //default 0x3c, 0x3c, 0x3c, 0x3c, 0x3c, 0x3c
int main(void)
{
    uint8_t wifiAppVer, check_tx_power_buf[6];
    uint16_t wifiInitDataVer;
    APP_TIMER_INIT(APP_TIMER_PRESCALER, APP_TIMER_OP_QUEUE_SIZE, false);
    ble_stack_init();
    //wifi init
    cWifi_resource_init();
    cWifi_prepare_start();

    if(cWifi_get_version_info(&wifiAppVer, &wifiInitDataVer) && (wifiAppVer >= 3) && cWifi_get_tx_power_tables(check_tx_power_buf))
    {
        const uint8_t *p_pwr_tables;
        bool check_tx_power = false;
        if(wifiInitDataVer == 0x0302) //ce
        {
            p_pwr_tables = wifi_tx_power_table_CE;
            check_tx_power = true;
        }
        else if((wifiInitDataVer == 0x0402) || (wifiInitDataVer == 0x0602)) //fcc (R2, R4)
        {
            p_pwr_tables = wifi_tx_power_table_FCC;
            check_tx_power = true;
        }
        else if(wifiInitDataVer == 0x0502) //telect
        {
            p_pwr_tables = wifi_tx_power_table_TELECOM;
            check_tx_power = true;
        }
        if(check_tx_power)
        {
            if(memcmp(check_tx_power_buf, p_pwr_tables, 6) != 0)
            {
                if(cWifi_bypass_req(NULL, NULL) == CWIFI_Result_OK)
                {
                    char sendAtCmd[32];
                    int sendATCmdSize;
                    int timeout;
                    timeout = 5000;
                    while(!cWifiState_is_bypass_mode())
                    {
                        if(--timeout==0)break; //wait bypassmode
                        nrf_delay_ms(1);
                    }
                    if(timeout > 0)
                    {
                        sendATCmdSize = sprintf((char *)sendAtCmd, "AT+TXPWRITE=W"%02X"%02X"%02X"%02X"%02XW"WrWn",
                                                p_pwr_tables[0], p_pwr_tables[1], p_pwr_tables[2], p_pwr_tables[3], p_pwr_tables[4], p_pwr_tables[5]);
                        cWifiState_bypass_write_request(sendAtCmd, sendATCmdSize);
                        nrf_delay_ms(500);
                        cWifi_abort_req(); //power off wifi module
                        nrf_delay_ms(2000);
                        NVIC_SystemReset();
                    }
                }
            }
        }
    }
}

```