W800_SDK DEMO 运行指南

V1.3

北京联盛德微电子有限责任公司（winner micro)

地址：北京市海淀区阜成路 67 号银都大厦 18 层

电话：+86-10-62161900

公司网址：www.winnermicro.com

文档修改记录

版本	修订时间	修订记录	作者	审核
V0.1	2019/9/25	[C]创建文档	Zhangwl	
V0.2	2020/7/2	更新 I2C 和 I2S 的 Demo 复用脚及说明	Cuiyc	
V0.3	2020/7/8	统一字体	Cuiyc	
V1.0	2020/8/4	增加 ADC、DSP 和 BLE 示例	Cuiyc	
V1.1	2020/10/29	更新 BLE 示例	Pengxg	
V1.2	2021/4/16	更新 httpget 和 httpfwup 示例参数	Cuiyc	
V1.3	2021/11/4	增加 TOUCHSENSOR 示例，增加 DEMO_HTTP 参数，更新 DEMO_RSA 测试结果	Chenzx	

目录

文档修改记录 2

目录 3

1 **引言** 7

1.1 编写目的 7

1.2 预期读者 7

1.3 术语定义 7

2 DEMO 概要 7

3 配网联网类 DEMO 功能描述 7

3.1 **DEMO_CONNECT_NET 操作步骤** 7

3.1.1 t-connect 加网 8

3.1.2 t-oneshot(oneshot 配网) 9

3.1.3 t-oneshot(airkiss 配网) 10

3.1.4 t-webcfg(网页配网) 10

3.2 **DEMO_APSTA 操作步骤** 11

3.3 **DEMO_SOFT_AP** 操作步骤 13

3.4 **DEMO_WPS** 操作步骤 14

3.4.1 t-wps-start-pbc 15

3.4.2 t-wps-start-pin 15

3.5 **DEMO_SCAN** 操作步骤 16

4 硬件驱动类 DEMO 功能描述 17

4.1 **DEMO_UARTx** 操作步骤 17

4.2 **DEMO_GPIO 操作步骤** 18
4.2.1 t-gpio 18
4.2.2 t-gpioirq 19
4.3 **DEMO_FLASH 操作步骤** 20
4.4 **DEMO_ENCRYPT 操作步骤** 21
4.5 **DEMO_RSA 操作步骤** 23
4.6 **DEMO_RTC 操作步骤** 24
4.7 **DEMO_TIMER 操作步骤** 25
4.8 **DEMO_PWM 操作步骤** 26
4.9 **DEMO_PMU 操作步骤** 27
4.9.1 t-pmuT0 28
4.9.2 t-pmuT1 28
4.10 **DEMO_I2C 操作步骤** 29
4.11 **DEMO_I2S 操作步骤** 31
4.12 **DEMO_MASTER_SPI 操作步骤** 33
4.13 **DEMO_ADC 操作步骤** 34
4.14 **DEMO_SLAVE_SPI 操作步骤** 35
4.15 **DEMO_SDIO_HOST 操作步骤** 37
4.16 **DEMO_TOUCHSENSOR 操作步骤** 38
5 应用类 DEMO 功能描述 39
5.1 **DEMO_STD_SOCKET_CLIENT 操作步骤** 39
5.2 **DEMO_STD_SOCKET_SERVER 操作步骤** 41
5.3 **DEMO_SOCKET_CLIENT_SERVER 操作步骤** 43

5.3.1 t-client 43
5.3.2 t-server 44
5.4 **DEMO_UDP 操作步骤** 46
5.4.1 UDP 广播 46
5.4.2 UDP 单播 47
5.4.3 UDP 组播 48
5.5 **DEMO_NTP 操作步骤** 50
5.5.1 t-ntp 50
5.5.2 t-setntps 51
5.5.3 t-queryntps 52
5.6 **DEMO_HTTP 操作步骤** 53
5.6.1 t-httpget 55
5.6.2 t-httpput 57
5.6.3 t-httppost 59
5.6.4 t-httpfwup 61
5.7 **DEMO_SSL_SERVER 操作步骤** 62
5.8 **DEMO_WEBSOCKETS 操作步骤** 64
5.8.1 websocket 不加密方式的数据通信 64
5.8.2 websocket 加密方式的数据通信 66
5.9 **DEMO_HTTPS 操作步骤** 67
5.10 **DEMO_MQTT 操作步骤** 68
5.11 **DEMO_DSP 操作步骤** 72
5.12 **DEMO_BT 操作步骤** 73

5.12.1 Ble server 示例 73
5.12.2 Ble client 示例 77
5.12.3 Ble 广播示例 78
5.12.4 Ble 扫描示例 79
5.13 **DEMO_FATFS** 操作步骤 80
5.14 **DEMO_MBEDTLS** 操作步骤 81

1 引言

1.1 编写目的

为基于 W80X 芯片 SDK 进行二次开发的软件开发工程师提供相关功能的代码示例。

1.2 预期读者

FAE，客户方软件开发工程师。

1.3 术语定义

2 DEMO 概要

该文档中用到的所有 DEMO 相关的宏定义都在 wm_demo.h 中。运行 DEMO 时必须打开该 DEMO 对应的宏定义，建议关闭不相关宏定义。DEMO 演示需要在控制台下进行，打开 DEMO_CONSOLE 编译选项，即打开了控制台。

DEMO_CONSOLE 同时还控制了 AT 指令的启用，如果使能此宏，则 AT 指令失效；关闭此宏，AT 指令生效。

以下三节将分别以配网联网类示例，硬件驱动类示例以及应用类示例来分别介绍其测试使用方法。

3 配网联网类 DEMO 功能描述

3.1 **DEMO_CONNECT_NET 操作步骤**

注：此 DEMO 下有五个示例。

3.1.1 t-connect 加网

功能描述	本例实现了使 WiFi 设备连接指定名称和密码的路由器的功能
命令格式	t-connect(“ssid_name”，“password”)

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_wifi_disconnect();
	tls_wifi_softap_destroy();
	tls_wifi_set_oneshot_flag(0);
	tls_mem_alloc();
	tls_netif_add_status_event();
	tls_wifi_connect();

涉及到的常用功能块	将设备的工作模式设置成 sta 模式：
	`tls_param_get(TLS_PARAM_ID_WPROTOCOL, (void *) &wireless_protocol, TRUE);`
	`if (TLS_PARAM_IEEE80211_INFRA != wireless_protocol)`
	`{`
	`  tls_wifi_softap_destroy();`
	`  wireless_protocol = TLS_PARAM_IEEE80211_INFRA;`
	`  tls_param_set(TLS_PARAM_ID_WPROTOCOL, (void *) &wireless_protocol, FALSE);`
	`}`

示例测试步骤	1. 打开宏定义 DEMO_CONNECT_NET; 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令; 3. 通过 uart0 发送 t-connect("TEST_N40_6","1234567890")来让模块加入名称为 TEST_N40_6，密码为 1234567890 的无线网络(根据现有网络来修改名称，这里的这个只是示例)。 注：**所有命令需要带回车换行,命令中使用英文符号**; 4. 加网成功后 uart0 会打印模块 ip。

3.1.2 t-oneshot(oneshot 配网)

功能描述	本例实现了使 WiFi 设备进行一键配网的功能，其中一键配网包括了官方的 oneshot 配网和 airkiss 配网
命令格式	t-oneshot

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_netif_add_status_event();
	tls_wifi_set_oneshot_config_mode();
	tls_wifi_set_oneshot_flag();

涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_CONNECT_NET，（TLS_CONFIG_UDP_ONE_SHOT 和 TLS_CONFIG_UDP_LSD_SPECIAL 默认是打开的）； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-oneshot； 4. 手机加入目标网络，安装 OneShotActivity（SDK ver2.0.0），在 app 界面输入正确 ssid 和 password，点 Start Configuration； 5. 模块加网成功后 uart0 会打印 ip。
App 下载地址	http://www.winnermicro.com/html/1/156/158/497.html，在页面下找到“软件材料”标签里的 oneshotconfig2.0.zip

3.1.3 t-oneshot(airkiss 配网)

功能描述	本例实现了使 WiFi 设备进行一键配网的功能，其中一键配网包括了官方的 oneshot 配网和 airkiss 配网
命令格式	t-oneshot

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_netif_add_status_event();
	tls_wifi_set_oneshot_config_mode();
	tls_wifi_set_oneshot_flag();

涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_CONNECT_NET，TLS_CONFIG_AIRKISS_MODE_ONESHOT; 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令; 3. 通过 uart0 发送 t-oneshot; 4. 手机加入目标网络（需要外网），打开微信，关注公众号【联盛德微电子】，进入公众号后点击产品应用下的 AirKiss 配网，进入配置设备上网页面，设置正确 Wi-Fi 密码，点击连接按钮; 5. 模块加网成功后 uart0 会打印 ip。

3.1.4 t-webcfg(网页配网)

功能描述	本例实现了通过内置网页来对设备进行网络配置的功能

命令格式	t-webcfg

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_netif_add_status_event();
	tls_wifi_set_oneshot_config_mode();
	tls_wifi_set_oneshot_flag();

涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_CONNECT_NET，（TLS_CONFIG_WEB_SERVER_MODE 默认是打开的）； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-webcfg； 4. 手机或者有无线网卡的电脑加入”softap_XXXX”（其中 XXXX 是模块 mac 地址的后 4 位），用浏览器访问 192.168.1.1，在页面 List 中选择目标网络(如果找不到目标网络，尝试刷新页面或者手动输入 ssid)，然后在 pwd 输入框中输入正确密码，点击 save 按钮； 5. 模块加网成功后 uart0 会打印模块 ip，同网络设备可以 ping 通模块 ip。

3.2 DEMO_APSTA 操作步骤

功能描述	本例实现了让设备建立一个 apsta 共存状态的功能，同时作为 sta 时去连

	接指定的路由器，而作为 ap 时也允许其它 sta 设备通过指定的密码来连接。同时建立起了 udp 的数据转发功能，具体功能在测试步骤中的详细描述；
命令格式	t-apsta("ssid_name","password","softapssid","87654321")，其中的 4 个参数分别是待连接的路由器的名称和密码及作为 ap 时的名称和密码。

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_netif_add_status_event();
	tls_wifi_set_oneshot_config_mode();
	tls_wifi_set_oneshot_flag();

涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_APSTA； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-apsta("TEST_N40_6","1234567890","softapssid","87654321"); 4. uart0 会打印 softap 的 ip 与模块 sta 的 ip； 5. 在与模块同网络的 PC1 上打开调试助手 UDP 的 65530 端口，设置十六进制显示； 6. 使用其它 PC2 加入 softap，uart0 会打印设备上线； 7. 设置 PC2 打开调试助手监听 UDP 的 65530 端口，设置十六进制显示；

	8. 通过 uart0 发送 t-asskt； 9. 此时 PC1 上的调试助手会收到 sta 重复发的 mac 地址； 10. 大约 1 分钟之后 PC2 上的调试助手会收到 softap 重复发的 mac 地址； 11. 手机加入 softap 后，uart0 会打印设备上线，手机可以 ping 通路由器下的设备。

3.3 **DEMO_SOFT_AP** 操作步骤

功能描述	本例实现了使设备工作在 softAP 模式的功能
命令格式	t-softap("softap1s","1234567890",6,4,1)；其中的 5 个参数分别表示 ap 的名称，密码，所用信道，加密方式和密码的格式； 加密方式：/*0:open，1:wep64，2:wep128,3:TKIP WPA ,4: CCMP WPA, 5:TKIP WPA2 ,6: CCMP WPA2*/ 密码格式：/*key's format:0-HEX, 1-ASCII*/

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_mem_alloc();
	tls_mem_free();
	tls_wifi_set_oneshot_flag();
	tls_wifi_disconnect();
	tls_wifi_softap_create();
	tls_os_timer_create();
	tls_os_timer_start();
	tls_os_timer_delete();

涉及到的常用功能块	将设备的工作模式设置成 ap 模式：
	`tls_param_get(TLS_PARAM_ID_WPROTOCOL, (void *) &wireless_protocol, TRUE);`
	`if (TLS_PARAM_IEEE80211_SOFTAP != wireless_protocol)`
	`{`
	`wireless_protocol = TLS_PARAM_IEEE80211_SOFTAP;`
	`tls_param_set(TLS_PARAM_ID_WPROTOCOL, (void *) &wireless_protocol, FALSE);`
	`}`

示例测试步骤	1. 打开宏定义 DEMO_SOFT_AP； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-softap("softap1s","1234567890",6,4,1)可以使设备建立起名为“softap1s”，密码为“1234567890”的热点； 4. 手机可以扫描到"softap1s"网络，加入 softap 后，uart0 会打印手机 mac。

3.4 **DEMO_WPS** 操作步骤

注：此 DEMO 下有两个示例，需要路由器支持 wps，需要单独索取支持 WPS 的库。

3.4.1 t-wps-start-pbc

功能描述	本例实现了通过内置网页来对设备进行网络配置的功能
命令格式	t-wps-start-pbc

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_netif_add_status_event();
	tls_wifi_set_oneshot_config_mode();
	tls_wifi_set_oneshot_flag();

涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_WPS; 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令; 3. 通过 uart0 发送 t-wps-start-pbc，并在路由器上按 wps 按钮，稍候 uart0 打印 [CMD]t-wps-start-pbcStart WPS pbc mode ... WiFi JOIN SUCCESS NET UP OK,Local IP:192.168.1.101

3.4.2 t-wps-start-pin

功能描述	本例实现了通过 wps pin 的方式来对设备进行网络配置的功能
命令格式	t-wps-start-pin
涉及到的常用 api(其中 api 的具	tls_netif_add_status_event();

体释义请参考相关头文件注释)	tls_wifi_set_oneshot_flag(); tls_wps_start_pin();
涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_WPS； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-wps-get-pin，uart0 打印 pin 码并自动给模块设置； 4. 在路由器中输入 pin 码，启动连接； 5. 通过 uart0 发送 t-wps-start-pin，稍候 uart0 打印 [CMD]t-wps-start-pinStart WPS pin mode ... WiFi JOIN SUCCESS NET UP OK,Local IP:192.168.1.101

3.5 **DEMO_SCAN** 操作步骤

功能描述	本例实现了使用设备来扫描周围无线网络的功能
命令格式	t-scan
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_wifi_scan_result_cb_register; tls_wifi_scan;
涉及到的常用功能块	无
示例测试步骤	1，打开宏定义 DEMO_SCAN；

	2，编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3，通过 uart0 发送 t-scan； 4，设备收到 uart0 的命令后会去扫描周围网络，扫描完成后会将其打印到 uart0。

4 硬件驱动类 DEMO 功能描述

4.1 **DEMO_UARTx** 操作步骤

功能描述	本例实现串口 1 echo 数据的功能； 备注：如果需要测试其它串口，则需要将函数 demo_uart_task()中的宏定义“TLS_UART_1”修改成相应的串口号，同时将复用功能口也修改成相应的复用接口。
命令格式	t-uart=(baudrate,parity,stopbit)，其中的参数如其名称所示； Parity: 0，无校验；1，奇校验；2，偶校验； Stopbit: 0，一个停止位；1，两个停止位；

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_os_queue_create();
	tls_os_task_create();
	tls_os_queue_send();
	tls_os_queue_receive();
	wm_uart1_rx_config();
	wm_uart1_tx_config();

	tls_uart_rx_callback_register();
	tls_uart_read();
	tls_uart_write();

涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_ UARTx； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-uart=(9600,0,0)修改 uart1 的参数； 4. 串口工具设置波特率 9600、校验位 NONE、数据位 8、停止位 1，打开 uart1 发数据，模块会把收到的数据从 uart1 打印出来(PB06_TX PB07_RX)。

4.2 **DEMO_GPIO** 操作步骤

注：此 DEMO 下有两个示例。

4.2.1 t-gpio

功能描述	本例实现了使用 PB6，用于演示 GPIO 的输入输出及上拉浮空功能
命令格式	t-gpio
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_gpio_cfg(); tls_gpio_read();

	tls_gpio_write();
涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_ GPIO； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-gpio，uart0 会打印测试结果 gpioB[6] default value==[0] gpioB[6] floating high value==[1] gpioB[6] floating low value==[0] gpioB[6] pullhigh high value==[1] gpioB[6] pullhigh low value==[0]

4.2.2 t-gpioirq

功能描述	本例实现了使用 PA1 作为输入脚来产生中断的功能；
命令格式	t-gpioirq

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_gpio_cfg();
	tls_gpio_isr_register();
	tls_gpio_irq_enable();
	tls_get_gpio_irq_status();
	tls_clr_gpio_irq_status();

涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_ GPIO；

	2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-gpioirq，把 PA1 拉低，uart0 打印 int flag =1 after int io =0 4. 把 PA1 拉高，uart0 打印 int flag =1 after int io =1

4.3 **DEMO_FLASH** 操作步骤

功能描述	本例实现了内部 flash 的读写功能. 写之前用户无需调用擦除函数，因其写函数内部已经集成擦除功能。
命令格式	t-flash
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_fls_write(); tls_fls_read();
涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_ FLASH； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-flash，uart0 会打印 success。

4.4 DEMO_ENCRYPT 操作步骤

功能描述	本例介绍了 sdk 内部自带的加密哈希等相关的函数使用方法；
命令格式	t-crypt

涉及到的加密算法	RNG_hard_demo();
	rc4_hard_demo();
	aes_hard_demo();
	des_hard_demo();
	des3_hard_demo();
	crc_hard_demo();
	md5_hard_demo();
	sha1_hard_demo();

涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_ ENCRYPT； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-crypt，uart0 会打印 [CMD]t-cryptRNG out: 1 0 0 0 2A 0 0 0 5E 50 RNG out: C2 1F 1 8D 34 5E F8 23 47 40 E3 85 B 7F 4 34 D0 78 E1 8F

	rc4 test success aes ecb test success aes cbc test success aes ctr test success des ecb test success des cbc test success 3des ecb test success 3des cbc test success CRYPTO_CRC_TYPE_8 normal value:0x000000B1 CRYPTO_CRC_TYPE_8 INPUT_REFLECT value:0x0000008B CRYPTO_CRC_TYPE_8 OUTPUT_REFLECT value:0x0000008D CRYPTO_CRC_TYPE_8 INPUT_REFLECT | OUTPUT_REFLECT value:0x000000D1 CRYPTO_CRC_TYPE_16_MODBUS normal value:0x00004755 CRYPTO_CRC_TYPE_16_MODBUS INPUT_REFLECT value:0x000090B1 CRYPTO_CRC_TYPE_16_MODBUS OUTPUT_REFLECT value:0x0000AAE2 CRYPTO_CRC_TYPE_16_MODBUS INPUT_REFLECT |

	OUTPUT_REFLECT value:0x00008D09 CRYPTO_CRC_TYPE_16_CCITT normal value:0x0000B888 CRYPTO_CRC_TYPE_16_CCITT INPUT_REFLECT value:0x00005B58 CRYPTO_CRC_TYPE_16_CCITT OUTPUT_REFLECT value:0x0000111D CRYPTO_CRC_TYPE_16_CCITT INPUT_REFLECT | OUTPUT_REFLECT value:0x00001ADA CRYPTO_CRC_TYPE_32 normal value:0x3F96E516 CRYPTO_CRC_TYPE_32 INPUT_REFLECT value:0x1DD50C89 CRYPTO_CRC_TYPE_32 OUTPUT_REFLECT value:0x68A769FC CRYPTO_CRC_TYPE_32 INPUT_REFLECT | OUTPUT_REFLECT value:0x9130ABB8 md5 test success sha1 test success

4.5 **DEMO_RSA** 操作步骤

功能描述	本例实现了不同长度的 rsa 算法的使用步骤；
命令格式	t-rsa

涉及到的 rsa 计算的长度	rsa128_demo();
	rsa256_demo();
	rsa512_demo();
	rsa1024_demo();
	rsa2048_demo();

涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_ RSA; 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令; 3. 通过 uart0 发送 t-rsa，uart0 会打印 [CMD]t-rsa RSA key validation: passed PKCS#1 encryption : passed PKCS#1 decryption : passed PKCS#1 data sign : passed PKCS#1 sig. verify: passed

4.6 **DEMO_RTC** 操作步骤

功能描述	本例实现了芯片内置的 RTC 的使用步骤;
命令格式	t-rtc
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_set_rtc(); tls_rtc_isr_register();

	tls_rtc_timer_start();
	tls_get_rtc();
	tls_os_time_delay();

涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_ RTC; 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令; 3. 通过 uart0 发送 t-rtc 开启 rtc clock，20 秒时 uart0 会打印 rtc clock 表示进入 rtc 中断。

4.7 **DEMO_TIMER** 操作步骤

功能描述	本例实现了芯片内置的硬件定时器的使用方法; 备注：，芯片共内置有 5 个定时器，相关的 api “tls_timer_create”会返回当前未使用到的定时器句柄号；定时器的时间单位可设置成微秒或者毫秒两种。
命令格式	t-timer
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_timer_create(); tls_timer_start();
涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_ TIMER; 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令;

	3. 通过 uart0 发送 t-timer 开启 timer，uart0 每 2 秒打印 timer irq 表示进入 timer 中断。

4.8 **DEMO_PWM** 操作步骤

功能描述	本例实现了芯片内置的 PWM 外设的使用方法；
命令格式	t-pwm=(1,250,99,4,0) 第一个参数为通道序号，包含两组复用，序号 0-4 分别对应 demo 中的 PB00、PB01、PB02、PB03、PA07 共五路，5-9 对应 PB19、PB20、PA00、PA01、PA04；第二个参数是期望输出的 pwm 频率；第三个参数是占空比，比如此处是 99 则表示实际占空比为 99/255；第四个参数表示当前模式，其中 4 表示独立模式，即只此路 pwm 输出波形；第 5 个参数表示输出的波形周期数，其中 0 表示持续输出波形。具体定义可参考函数 pwm_demo()的上方注释。

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	wm_pwm0_config();
	wm_pwm1_config();
	wm_pwm2_config();
	wm_pwm3_config();
	wm_pwm4_config();
	tls_pwm_stop();
	tls_pwm_init();
	tls_pwm_start();

	tls_pwm_out_init();
	tls_pwm_isr_register();
	tls_pwm_cap_init();
	tls_dma_start();
	tls_dma_irq_register();

涉及到的常用功能块	pwm_demo_allsyc_mode();
	pwm_demo_multiplex_config();
	pwm_demo_2syc_mode();
	pwm_demo_mc_mode();
	pwm_demo_break_mode();
	pwm_isr_callback();
	pwm_capture_mode_int();
	pwm_capture_mode_dma();

示例测试步骤	1. 打开宏定义 DEMO_ PWM； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-pwm=(1,250,99,4,0),示波器量 PB01 可以测到 250Hz，占空比约为 39%(99/255)的波形。

4.9 **DEMO_PMU** 操作步骤

注：此 DEMO 下有两个示例。

4.9.1 t-pmuT0

功能描述	本例实现了控制设备进入 standby 的低功耗模式并定时将其唤醒的功能；
命令格式	t-pmuT0

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_pmu_timer0_isr_register();
	tls_pmu_timer0_start();
	tls_pmu_standby_start();

涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_ PMU； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-pmuT0 模块启动 timer0 进入 standby，10 秒左右 uart0 打印模块复位，表示 timer0 中断唤醒。

4.9.2 t-pmuT1

功能描述	本例实现了
命令格式	t-pmuT1

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_pmu_timer1_isr_register();
	tls_pmu_timer1_start();
	tls_pmu_standby_start();

涉及到的常用功能块	无

示例测试步骤	1. 打开宏定义 DEMO_ PMU； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-pmuT1 模块启动 timer1 进入 standby，5 秒左右 uart0 打印模块复位，表示 timer1 中断唤醒。

4.10**DEMO_I2C** 操作步骤

注：此 DEMO 需要 AT24CXX 芯片

功能描述	本例实现了使用芯片内置的 i2c 模块来向 at24cxx 设备来进行写读数据的过程； 备注：上图所示的测试板上默认接口上拉电阻，如果用户使用其它 i2c 设备测试不成功进，需要检查下连接电路的两条线上是否有上拉或者下拉。此处是不可以有下拉电阻的。
命令格式	t-i2c
涉及到的常用 api(其中 api 的具	wm_i2c_scl_config();

体释义请参考相关头文件注释)	wm_i2c_sda_config();
	tls_i2c_init();
	tls_i2c_write_byte();
	tls_i2c_wait_ack();
	tls_i2c_read_byte();

涉及到的常用功能块	AT24CXX_ReadOneByte(); AT24CXX_ReadLenByte(); AT24CXX_WriteOneByte(); AT24CXX_Write();
示例测试步骤	1. 打开宏定义 DEMO_ I2C; 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令; 3. 模块 PIN 连接 AT24CXX 芯片: 4. PA01 接 SCL，PA04 接 SDA，GND 接 GND，VCC 接 3.3v 5. 通过 uart0 发送 t-i2c，uart0 返回 [CMD]t-i2c AT24CXX check success read data is:AT24CXX I2C TEST OK

4.11 DEMO_I2S 操作步骤

功能描述	此 DEMO 用于演示设备进行 i2s 格式的数据通信。需要另一个相应的主设备或者从设备来配合发送或者接收数据。 备注：接线方式 ck-ck ws-ws, di-do, do-di
命令格式	* @param[in] format * - \ref 0: i2s * - \ref 1: msb * - \ref 2: pcma * - \ref 3: pcmb * * @param[in] tx_rx * - \ref 1: transmit * - \ref 2: receive * * @param[in] freq * sample rate * * @param[in] datawidth * - \ref 8: 8 bit * - \ref 16: 16 bit * - \ref 24: 24 bit * - \ref 32: 32 bit * * @param[in] stereo * - \ref 0: stereo * - \ref 1: mono * * @param[in] mode * - \ref 0: interrupt * - \ref 1: dma * * @retval * * @note * t-i2s=(0,1,44100,16,0,0) -- M_I2S send(ISR mode) * t-i2s=(0,1,44100,16,0,1) -- M_I2S send(DMA mode) * t-i2s=(0,2,44100,16,0,0) -- S_I2S recv(ISR mode) * t-i2s=(0,2,44100,16,0,1) -- S_I2S recv(DMA mode) */

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	wm_i2s_port_init();
	wm_i2s_tx_int();
	wm_i2s_rx_int();
	wm_i2s_tx_rx_int();
	wm_i2s_tx_dma();
	wm_i2s_rx_dma();

	wm_i2s_tx_rx_dma();
涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_ I2S； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 设备相应的 pin 接测试设备对应的 pin： 设备侧的引脚定义为：ck--PB08, ws--PB09, di--PB10, do--PB11，四条信号线接好后还需要将两个通信设备共地。 4. 通过两个设备的 uart0 发送 t-i2sioinit 让设备初始化 io； 5. 通过 uart0 发送 t-i2s=(0,2,44100,16,0,1)将使用 DMA 方式来接收数据，此时的设备将处于 slave 状态； 6. 通过 uart0 发送 t-i2s=(0,1,44100,16,0,1)将使用 DMA 方式来发送数据，此时的设备将处于 master 状态（slave 端会打印全双工和半双工接收数据的对比结果）； 7. 复位设备，重新初始化 io； 8. 通过 uart0 发送 t-i2s=(0,4,44100,16,0,1)将使用 DMA 方式来接收数据，此时的设备将处于 slave 状态；

	9. 通过 uart0 发送 t-i2s=(0,3,44100,16,0,1)将使用 DMA 方式来发送数据，此时的设备将处于 master 状态（两端会打印全双工和半双工接收数据的对比结果）。

4.12 **DEMO_MASTER_SPI** 操作步骤

功能描述	本例实现了芯片侧作为 master 通过 spi 接口与 slave 侧的设备进行数据收发的过程； 备注：测试此示例时，如果有需要可以在四条信号线上串口几十欧姆的电阻来保证通信正常。 此 DEMO 需要下载对端代码；
命令格式	t-mspi-s t-mspi-r

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_spi_trans_type();
	tls_spi_setup();
	tls_spi_write();
	tls_spi_read();

涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_ MASTER_SPI； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 用 keil 打开

	STM32_SOC_TEST_SLAVE_SPI\Project\STM32F10x_StdPeriph_Template\MDK-ARM\Project 编译后通过 jlink 给 stm32 升级； 注：STM32 开发板型号：**STM32_Mini_V2.0**   4. 模块 PIN 连接对端 stm32(PA9tx，PA10rx 作为打印口)： PB4 接 PB12(cs)，PB2 接 PB13(ck)，PB3 接 PB14(so)，PB5 接 PB15(si)，GND 接 GND； 5. 通过 uart0 发送 t-mspi-s(1000000,0)发送 1500 数据，stm32 的 uart0 打印 down data len: 1500； 6. 通过 uart0 发送 t-mspi-r，模块 uart0 打印 [CMD]t-mspi-rSPI Master receive 1500 byte, modeA, little endian rcv data len: 1500。

4.13**DEMO_ADC** 操作步骤

功能描述	本例实现了 ADC 针对芯片温度采集和外部输入电压检测的功

	能
命令格式	t-adcvolt(x)，x 取值 0 表示通道 0，1 表示通道 1，8 表示差分； t-adctemp
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	adc_temp wm_adc_config adc_get_inputVolt
涉及到的常用功能块	无
示例测试步骤	1）针对芯片温度测试，串口 0 直接输入 t-adctemp 命令执行即可返回当前的芯片温度：tem:xxx 2）针对输入电压，串口 0 输入命令： 单端测试：t-adcvolt(0)或者 t-adcvolt(1) 差分测试：t-adcvolt(8) 执行完成后，返回： chan:x, xxxx(mV) or x.xxx(V)

4.14DEMO_SLAVE_SPI 操作步骤

功能描述	本例实现了设备作为 slave 时通过 HSPI 接口与主设备进行数据通信的过程； 注：此 DEMO 使用 W800_ARDUINO_V1.0 开发板,并且需要下载对端代码，STM32 开发板型号：STM32_Mini_V2.0。下图为主机开发板；

	 
命令格式	t-sspi(0)

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	wm_hspi_gpio_config();
	tls_slave_spi_init();
	tls_set_high_speed_interface_type();
	tls_set_hspi_user_mode();
	tls_hspi_rx_data_callback_register();
	tls_hspi_rx_cmd_callback_register();

涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_ SLAVE_SPI 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 用 keil 打开 stm32_ucos_ri\uCOSDemo 编译后通过 jlink 给 stm32 升级； 4. 模块 PIN 连接对端 stm32(PA9tx，PA10rx 作为打印口)： PB09 接 PA4(cs)，PB06 接 PA5(ck)，PB11 接 PA6(mi)，PB10 接 PA7(mo)，PB07 接 PA2(cts)，

	GND 接 GND 5. 通过 uart0 发送 t-sspi(0) 6. 复位 stm32，模块 uart0 打印： HspiRxCmdCb rx[5] :5a 00 05 01 60 RX ok 100 RX ok 200 RX ok 300 7. Stm32 打印： ###kevin debug ... tx start cmd kevin debug TX_BUFF_AVAIL = 3, cmdlen=8 RX ok 100 RX ok 200 RX ok 300

4.15DEMO_SDIO_HOST 操作步骤

功能描述	本例实现了通过芯片内置的 sdio 接口来对 sd 卡进行读写操作的过程；
命令格式	t-sdh
涉及到的常用 api(其中 api 的具体释义请参考相关头文	wm_sd_card_set_bus_width(); wm_sd_card_set_blocklen();

件注释)	wm_sd_card_block_write();
	wm_sd_card_block_read();
	wm_sd_card_blocks_write();
	wm_sd_card_blocks_read();

涉及到的常用功能块	无
示例测试步骤	1，打开宏定义 DEMO_SDIO_HOST； 2，编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3，在开发板上接好 sd 卡，本示例使用的 IO 口为 PB06-PB11； 4，通过 uart0 发送 t-sdh； 5，设备收到串口 0 的命令后分别使用中断方式和 dma 的方式来向 sd 卡的指定 block 写入并读出数据；若写入的和读出的数据均相同，则会打印测试成功相关的消息；若有不同则会打印失败相关的消息。

4.16DEMO_TOUCHSENSOR 操作步骤

功能描述	本例实现了芯片的 touchsensor 功能；
命令格式	t-touch(touchnum); touchnum 输入参数为十进制，每 1bit 对应 1 个 touchsensor，如 3 对应打开 touch2 和 touch1，7 对应打开 touch3、touch2 和 touch1。注意：touchnum 为 0 时，会打开全部 15 个 touchsensor；当仅打开 1 个 touch

	时，touch1 也需要打开，如 5 对应打开 touch3 和 touch1。
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	无
涉及到的常用功能块	无
示例测试步骤	1，打开宏定义 DEMO_TOUCHSENSOR； 2，编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3，把开发板的 touchio 接到触摸板； 4，通过 uart0 给开发板发送 t-touch(9)； 5,点击触摸板 touch4 位置，uart0 会打印触发的 key4。

5 应用类 DEMO 功能描述

5.1 DEMO_STD_SOCKET_CLIENT 操作步骤

注：通过 uart0 发送 demohelp 模块 uart0 会返回控制台信息。

功能描述	本例实现了使用标准的 socket 函数来创建 tcp 客户端来与同局域网内 PC 上的服务器端进行数据通信的过程；设备端作为客户端，会将从服务端收到的数据的长度打印出来，并将数据通过串口发送出去；
命令格式	t-sockc(port, ip) t-skcsnd(len, uart_trans)
涉及到的常用 api(其中 api	socket();

的具体释义请参考相关头文件注释)	connect();
	closesocket();
	recv();
	tls_uart_write();

涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_STD_SOCKET_CLIENT 和 DEMO_CONNECT_NET; 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令; 3. 通过 uart0 发送 t-connect("TEST_N40_6","1234567890") 让模块加网; 4. 在与模块同网络的 PC（ip 为 192.168.1.100）上打开调试助手 tcp server 端口号 1000; 5. 通过 uart0 发送 t-sockc(1000,192.168.1.100)让模块创建 tcp client 连接对端 server，连接成功后 uart0 会打印 socket num; 6. Server 发数据，模块收到数据后 uart0 会打印收到的数据长度，每次累加; 7. 通过 uart0 发送 t-skcsnd(0,1)设置使用 uart1 透传; 8. 串口工具设置波特率 115200、校验位 NONE、数据

	位 8、停止位 1 打开 uart1，通过 uart1 与 server 双向透传；

5.2 DEMO_STD_SOCKET_SERVER 操作步骤

功能描述	本例实现了使用标准的 socket 函数来创建 tcp 服务端来与同局域网内 PC 上的客户端端进行数据通信的过程； 设备端建立 tcp server 成功后，可以在 PC 打开工具建立 client 端与来与其建立连接；建立连接成功后，通过工具由 PC 向设备发送数据，设备收到后会打印收到的数据的累加长度值；也可以通过串口向设备发送透传数据，使数据被传输到 PC 的 client 处；
命令格式	t-socks(port) t-skssnd(sock,len,uart_number)

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	socket();
	connect();
	closesocket();
	recv();
	bind();
	listen();
	accept();
	send();
	tls_uart_write();

涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_STD_SOCKET_ SERVER 和 DEMO_CONNECT_NET； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-connect("HUAWEI-6SEWE5","123456789") 或 t-oneshot 让模块加网； 4. 通过 uart0 发送 t-socks(2000)让模块创建 tcp server，uart0 会打印监听的端口； 5. 在与模块同网络的 PC 上打开调试助手，创建 tcp client（设置模块的 ip 和端口号）连接模块 server，连接成功后 uart0 会打印 client 信息（模块 server 最多连接 7 个 client）； 6. client 发数据，模块收到数据后 uart0 会打印收到对应连接的数据长度，每次累加； 7. 通过 uart0 发送 t-skssnd(1,16,0)使用 1 号连接发送长度 16 的固定数据，client 能收到数据； 8. 通过 uart0 发送 t-skssnd(1,0,1)设置 1 号连接在 uart1 透传； 9. 串口工具设置波特率 115200、校验位 NONE、数据位 8、停止位 1 打开 uart1，通过 uart1 与 client

	双向透传。

5.3 DEMO_SOCKET_CLIENT_SERVER 操作步骤

本测试宏开关下共有两个示例，分别是设备作为 tcp client 与设备作为 tcp server。

5.3.1 t-client

功能描述	本例实现了使设备去连接指定名称和密码的路由器，并建立 tcp 客户端，再去连接指定地址和端口的 tcp 服务端并进行数据通信的过程；
命令格式	t-client("ssid","password",port,“ip”)

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	socket();
	connect();
	closesocket();
	recv();
	tls_wifi_connect();

涉及到的常用功能块	`static int c_connect_wifi(char *ssid, char *pwd)`
	`{`
	`  if (!ssid)`
	`  {`
	`  return WM_FAILED;`
	`  }`
	`  printf("\nssid:%s\n", ssid);`
	`  printf("password=%s\n",pwd);`
	`  tls_netif_add_status_event(c_con_net_status_changed_event);`
	`  tls_wifi_connect((u8 *)ssid, strlen(ssid), (u8 *)pwd, strlen(pwd));`
	`  return 0;`
	`}`

示例测试步骤	1，打开宏定义 DEMO_SOCKET_CLIENT_SERVER、DEMO_CONNECT_NET； 2，编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令；

	3，在 PC 上建立一个 tcp server，设置监听端口为 8080。 4，通过 uart0 发送 t-client("TEST_N40_6","1234567890", 8080，“192.168.1.100”);其中的四个参数分别为待连接路由器的名称，密码，待连接服务器的端口号及 ip 地址。 5，设备收到串口 0 的命令后会去连接路由器，连接路由成功后会去连接服务器；连接服务器成功后会向其发送一条消息；用户可以在服务器侧看到此消息，此时可通过服务器返回一条消息给设备，设备收到消息后会有相应打印； 6，设备会一直处于发送接收再发送再接收的过程，直到连接断开。

5.3.2 t-server

功能描述	本例实现了使设备去连接指定名称和密码的路由器，并建立 tcp 服务端，再去接收客户端的连接并与其进行数据通信的过程；
命令格式	t-server("ssid","password", port)

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	socket();
	connect();
	closesocket();
	recv();
	bind();
	listen();
	accept();

	send();
	
	tls_wifi_connect();

涉及到的常用功能块	`static int s_connect_wifi(char *ssid, char *pwd)`
	`{`
	`  if (!ssid)`
	`  {`
	`  return WM_FAILED;`
	`  }`
	`  printf("\nssid:%s\n", ssid);`
	`  printf("password=%s\n",pwd);`
	`  tls_netif_add_status_event(s_con_net_status_changed_event);`
	
	`  tls_wifi_connect((u8 *)ssid, strlen(ssid), (u8 *)pwd, strlen(pwd));`
	`  return 0;`
	`}`

示例测试步骤	1，打开宏定义 DEMO_SOCKET_CLIENT_SERVER、DEMO_CONNECT_NET; 2，编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3，通过 uart0 发送 t-server("TEST_N40_6","1234567890", 8080,)；其中的三个参数分别为待连接路由器的名称，密码，服务器的端口号。 4，设备收到串口 0 的命令后会去连接路由器，连接路由成功后会打印 ip 地址并去建立 tcp 服务器，并监听自己的 8080 端口； 5，在处于相同局域网的 PC 上使用 tcp 工具建立一个 tcp 客户端去连接此服务器的 ip 和端口；建立成功后，可以通过工具向其发送数据； 6，服务器收到数据后，会向客户端侧发送”message from server”的字符串。 7，设备将一直处于接收，发送，再接收再发送的过程，直到连接断开。

5.4 DEMO_UDP 操作步骤

注：此 DEMO 下有三个示例，需要使用抓包网卡

5.4.1 UDP 广播

功能描述	本例实现了通过 udp 方式来向外广播数据的过程；
命令格式	t-udp(mode,port,ip) t-sndudp(len)
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_netif_get_ethif() socket() bind() closesocket() setsockopt() recvfrom() sendto();
涉及到的常用功能块	`ethif = tls_netif_get_ethif();` `printf("local ip : %d.%d.%d.%d\n", ip4_addr1(ip_2_ip4(&ethif->ip_addr)), ip4_addr2 ip4_addr3(ip_2_ip4(&ethif->ip_addr)), ip4_addr4(ip_2_ip4(&ethif->ip_addr)));`
示例测试步骤	1. 打开宏定义 DEMO_UDP 和 DEMO_CONNECT_NET； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-connect("TEST_N40_6","1234567890")或 t-oneshot 让模块加网； 4. 通过 uart0 发送 t-udp(0,1000,0)uart0 打印

	udp demo,cast:0, port:1000 localip : 192.168.1.104 local port :3000 5. 在与模块同网络的 PC 上打开调试助手 udp 端口 1000； 6. 通过 uart0 发送 t-sndudp(10)，抓包网卡可以抓到模块到路由器的 Destination 为 Ethernet Broadcast 的包，同时调试助手收到了 10 个数据； 7. 调试助手发数据，模块收到数据后 uart0 会打印地址和数据长度。

5.4.2 UDP 单播

功能描述	本例实现了通过 udp 方式来向指定设备单播数据的过程；
命令格式	t-udp(mode,port,ip) t-sndudp(len)
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_netif_get_ethif() socket() bind() closesocket() setsockopt() recvfrom() sendto();
涉及到的常用功能块	`ethif = tls_netif_get_ethif();` `printf("local ip : %d.%d.%d.%d\n", ip4_addr1(ip_2_ip4(&ethif->ip_addr)), ip4_addr2 ip4_addr3(ip_2_ip4(&ethif->ip_addr)), ip4_addr4(ip_2_ip4(&ethif->ip_addr)));`

示例测试步骤	1. 打开宏定义 DEMO_UDP 和 DEMO_CONNECT_NET； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-connect("TEST_N40_6","1234567890")或 t-oneshot 让模块加网； 4. 通过 uart0 发送 t-udp(1,1001,192.168.1.100)uart0 会打印 udp demo,cast:1, port:1001 localip : 192.168.1.104 local port :3000 5. 在与模块同网络的 PC（ip 为 192.168.1.100）上打开调试助手连接 udp 端口 1001； 6. 通过 uart0 发送 t-sndudp(10)抓包网卡可以抓到模块到路由器的 Destination 为 PC 网卡的包，同时调试助手收到了 10 个数据； 7. 调试助手发数据，模块收到数据后 uart0 会打印地址和数据长度。

5.4.3 UDP 组播

功能描述	本例实现了通过 udp 方式来向外组播数据的过程；
命令格式	t-udp(mode,port,ip) t-sndudp(len)
涉及到的常用	tls_netif_get_ethif()

api(其中 api 的具体释义请参考相关头文件注释)	socket() bind() closesocket() setsockopt() recvfrom() sendto();
涉及到的常用功能块	`ethif = tls_netif_get_ethif();` `printf("local ip : %d.%d.%d.%d\n", ip4_addr1(ip_2_ip4(&ethif->ip_addr)), ip4_addr2 ip4_addr3(ip_2_ip4(&ethif->ip_addr)), ip4_addr4(ip_2_ip4(&ethif->ip_addr)));`
示例测试步骤	1. 打开宏定义 DEMO_UDP 和 DEMO_CONNECT_NET; 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令; 3. 通过 uart0 发送 t-connect("TEST_N40_6","1234567890")或 t-oneshot 让模块加网; 4. 通过 uart0 发送 t-udp(2,5100,224.1.2.1)uart0 会打印; udp demo,cast:2, port:5100 localip : 192.168.1.104 local port :3000 setmuticast 5. 在与模块同网络的 PC 上打开组播工具，在接收测试中添加地址(组播地址为 224.1.2.1，端口为 5100)，选择地址，点击接收按钮; 6. 通过 uart0 发送 t-sndudp(1024)，组播工具中显示未丢包;

	7. 在 PC 打开调试助手，设置目标组播地址 224.1.2.1 目标端口 3000，发送数据，模块收到数据后 uart0 打印地址和数据长度。

5.5 **DEMO_NTP** 操作步骤

注：此 DEMO 下有三个示例。

5.5.1 t-ntp

功能描述	本例实现了使用 ntp 方式来获取当前时间的过程；
命令格式	t-ntp

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_ntp_client();
	localtime();
	tls_set_rtc();

涉及到的常用功能块	`static int isNetworkOk(void)`
	`{`
	`  struct tls_ethif *etherIf = tls_netif_get_ethif();`
	`  return etherIf->status;`
	`}`
	
	`static void setAutoConnectMode(void)`
	`{`
	`  u8 auto_reconnect = 0xff;`
	`  tls_wifi_auto_connect_flag(WIFI_AUTO_CNT_FLAG_GET, &auto_reconnect);`
	`  if(auto_reconnect != WIFI_AUTO_CNT_ON)`
	`  {`
	`  auto_reconnect = WIFI_AUTO_CNT_ON;`
	`  tls_wifi_auto_connect_flag(WIFI_AUTO_CNT_FLAG_SET, &auto_reconnect);`
	`  }`
	`}`

示例测试步骤	1. 打开宏定义 DEMO_ NTP 和 DEMO_CONNECT_NET； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送

	t-connect("TEST_N40_6","1234567890")或 t-oneshot 让模块加网（有外网）； 4. 通过 uart0 发送 t-ntp，uart0 会打印当前时间。

5.5.2 t-setntps

功能描述	本例实现了通过命令来修改默认的 ntp 服务器的过程；
命令格式	t-setntps("ntp_server_name1","ntp_server_name2","ntp_server_name3")
涉及到的常用 api（其中 api 的具体释义请参考相关头文件注释)	tls_ntp_set_server()
涉及到的常用功能块	无

示例测试步骤	1. 打开宏定义 DEMO_ NTP 和 DEMO_CONNECT_NET； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-setntps("120.25.108.11", "ntp.sjtu.edu.cn", "us.pool.ntp.org")手动设置 ntp 服务器； 4. 复位模块后，通过 uart0 发送 t-queryntps 返回 [CMD]t-queryntps"120.25.108.11","ntp.sjtu.edu.cn","us.pool.ntp.org" 5. 通过 uart0 发送 t-connect("TEST_N40_6","1234567890")或 t-oneshot 让模块加网（有外网）； 6. 通过 uart0 发送 t-ntp，uart0 会打印当前时间。

5.5.3 t-queryntps

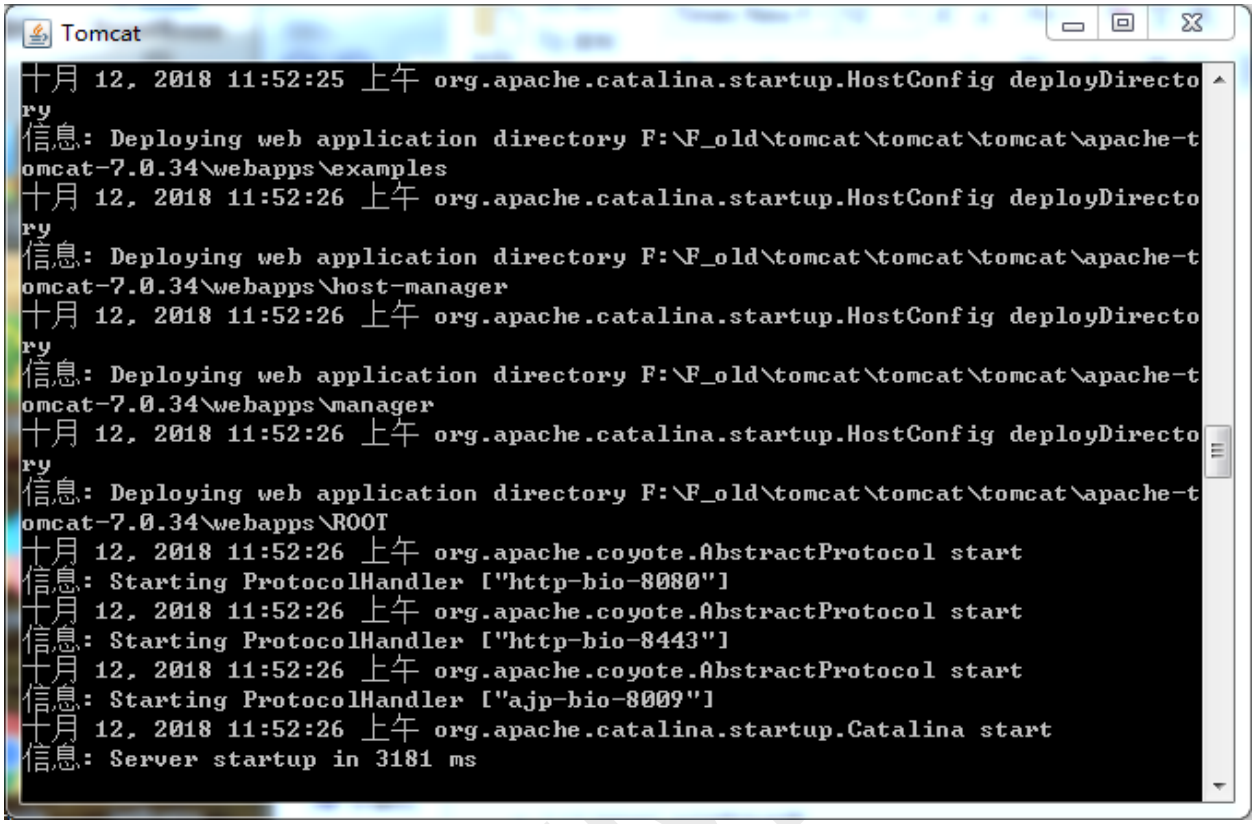
功能描述	本例实现了通过命令来查询当前所使用的 ntp 服务器名称的过程；
命令格式	t-queryntps
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_ntp_query_sntpcfg()
涉及到的常用功能块	无
示例测试步骤	1，打开宏定义 DEMO_ NTP 和 DEMO_CONNECT_NET； 2，编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令；

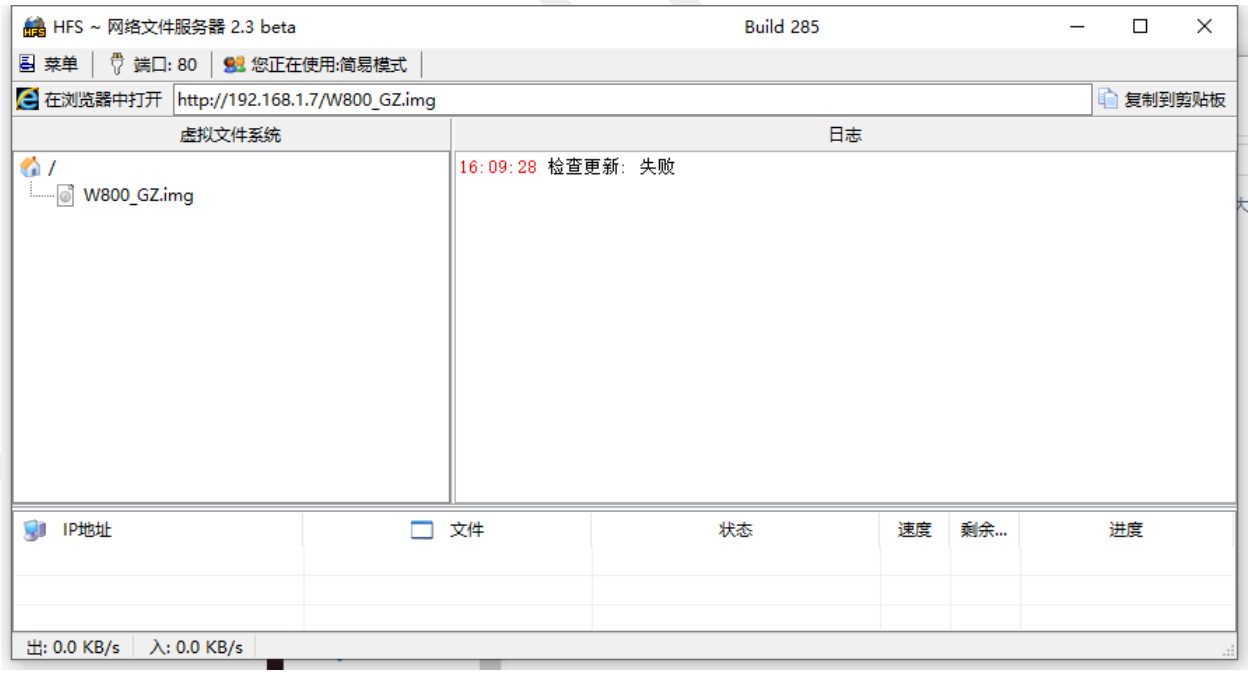
	3，通过 uart0 发送 t-connect("TEST_N40_6","1234567890")或 t-oneshot 让模块加网（有外网）； 4，通过 uart0 发送 t-queryntps，uart0 会打印当前所用到的 ntp 服务器的地址。

5.6 DEMO_HTTP 操作步骤

注：此 DEMO 下有四个示例，需要下载 tomcat 服务器（需要放置所需脚本文件）和 hfs 服务器。相关配件的下载地址在官网 http://www.winnermicro.com/html/1/156/158/497.html 的软件资料标签页下的”配套 wmsdk demo 使用的工具代码:”处。

下图分别为 tomcat 服务器启动后的页面以及 http 服务器添加固件就绪后的页面：





其中 hfs 服务器及 tomcat 服务器可以从网上下载，hfs 下载后直接可用，tomcat(已测试过 7.0.34 及 8.5.23 版本)服务器下载下来后需要在里面修改添加一些脚本文件。具体为将 tomcat 根目录下的 webapps 文件夹下的 TestWeb 文件夹替换为官方提供的 TestWeb 文

件夹(已在里面添加了测试 httpget httpput httppost 所需要的相应脚本文件)。

5.6.1 t-httpget

功能描述	本例实现了 http 格式数据通信中的 get 数据的过程;
命令格式	t-httpget=(http://xxx.xxx.xxx.xxx:8080/filepath/)
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	HTTPClientOpenRequest() HTTPClientSetVerb() HTTPClientSendRequest() HTTPClientRecvResponse() HTTPClientReadData() HTTPClientCloseRequest()

涉及到的常用功能块	`int http_get_demo(char *buf)`
	`{`
	`  HTTPParameters httpParams;`
	`  memset(&httpParams, 0, sizeof(HTTPParameters));`
	`  httpParams.Uri = (char *)tls_mem_alloc(128);`
	`  if(httpParams.Uri == NULL)`
	`  {`
	`  printf("malloc error.\n");`
	`  return WM_FAILED;`
	`  }`
	`  memset(httpParams.Uri, 0, 128);`
	`  sprintf(httpParams.Uri, "http://%d.%d.%d.%d:8080/TestWeb/",`
	`  httpParams.Verbose = TRUE;`
	`  printf("Location: %s\n", httpParams.Uri);`
	`  http_get(httpParams);`
	`  tls_mem_free(httpParams.Uri);`
	`  return WM_SUCCESS;`
	`}`

示例测试步骤	1. 打开宏定义 DEMO_HTTP 和 DEMO_CONNECT_NET; 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令; 3. 通过 uart0 发送 t-connect("TEST_N40_6","1234567890")或

t-oneshot 让模块加网；

4. 在与模块同网络的 PC（ip 为 192.168.1.100）上打开 tomcat 服务器并放置文件；
5. 通过 uart0 发送

t-httpget=(http://192.168.1.100:8080/TestWeb/)，uart0 返回

[CMD]t-httpgetLocation: http://192.168.1.100:8080/TestWeb/

HTTP Client v1.0

Start to receive data from remote server...

<html>

<body>

<h2>Hello World!</h2>

<form method="POST" action="/TestWeb/login.do">

userd: <input id="user" type="text" name="user"/>

<input type="submit" value="Submit" />

</form>

</body>

</html>

HTTP Client terminated 1000 (got 213 b)

5.6.2 t-httpput

功能描述	本例实现了 http 格式数据通信中的 put 数据的过程；
命令格式	t-httpput=(“put_data”)
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	HTTPClientOpenRequest() HTTPClientSetVerb() HTTPClientSendRequest() HTTPClientRecvResponse() HTTPClientReadData() HTTPClientCloseRequest()

涉及到的常用功能块	`int http_put_demo(char *putData)`
	`{`
	`  HTTPParameters httpParams;`
	`  memset(&httpParams, 0, sizeof(HTTPParameters));`
	`  httpParams.Uri = (char *)tls_mem_alloc(128);`
	`  if(httpParams.Uri == NULL)`
	`  {`
	`  printf("malloc error.\n");`
	`  return WM_FAILED;`
	`  }`
	`  memset(httpParams.Uri, 0, 128);`
	`  sprintf(httpParams.Uri, "http://%d.%d.%d.%d:8080/T`
	`  printf("Location: %s\n", httpParams.Uri);`
	`  httpParams.Verbose = TRUE;`
	`  http_put(httpParams, putData);`
	`  tls_mem_free(httpParams.Uri);`
	`  return WM_SUCCESS;`
	`}`

示例测试步骤	1. 打开宏定义 DEMO_HTTP 和 DEMO_CONNECT_NET； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-connect("TEST_N40_6","1234567890")或 t-oneshot 让模块加网；

	4. 在与模块同网络的 PC（ip 为 192.168.1.100）上打开 tomcat 服务器并放置文件； 5. 通过 uart0 发送 t-httpput=(user=winnermicroput)，uart0 返回 Location: http://192.168.1.100:8080/TestWeb/login_put.do HTTP Client v1.0 Start to receive data from remote server... <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd"> <html> <head> <meta http-equiv="Content-Type" content="text/html; charset=GBK"> <title>Insert title here</title> </head> <body> :winnermicroput </body> </html>

	HTTP Client terminated 1000 (got 277 b)

5.6.3 t-httppost

功能描述	本例实现了 http 格式数据通信中的 post 数据的过程；
命令格式	t-httppost=(“post_data”)
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	HTTPClientOpenRequest() HTTPClientSetVerb() HTTPClientSendRequest() HTTPClientRecvResponse() HTTPClientReadData() HTTPClientCloseRequest()

涉及到的常用功能块	`int http_post_demo(char *postData)`
	`{`
	`  HTTPParameters httpParams;`
	`  memset(&httpParams, 0, sizeof(HTTPParameters));`
	`  httpParams.Uri = (char *)tls_mem_alloc(128);`
	`  if(httpParams.Uri == NULL)`
	`  {`
	`  printf("malloc error.\n");`
	`  return WM_FAILED;`
	`  }`
	`  memset(httpParams.Uri, 0, 128);`
	`  sprintf(httpParams.Uri, "http://%d.%d.%d.%d:8080/T`
	`  printf("Location: %s\n", httpParams.Uri);`
	`  httpParams.Verbose = TRUE;`
	`  http_post(httpParams, postData);`
	`  tls_mem_free(httpParams.Uri);`
	`  return WM_SUCCESS;`
	`}`

示例测试步骤	1. 打开宏定义 DEMO_HTTP 和 DEMO_CONNECT_NET； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送

t-connect("TEST_N40_6","1234567890")或 t-oneshot 让模块加网；

4. 在与模块同网络的 PC（ip 为 192.168.1.100）上打开 tomcat 服务器并放置文件；
5. 通过 uart0 发送 t-httppost=(user=winnermicropost)，uart0 返回

Location: http://192.168.1.100:8080/TestWeb/login.do

HTTP Client v1.0

Start to receive data from remote server...

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=GBK">

<title>Insert title here</title>

</head>

<body>

:winnermicropost

	</body> \</html> HTTP Client terminated 1000 (got 278 b)

5.6.4 t-httpfwup

功能描述	本例实现了设备通过 ota 的方式来完成固件升级功能。
命令格式	t-httpfwup=(http://192.168.1.100:80/w800_ota.img) 上述命令中 ip 地址为 ota 服务器的 Ip 地址，冒号后为相应的端口号；
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	t_http_fwup()
涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_HTTP 和 DEMO_CONNECT_NET； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-connect("TEST_N40_6","1234567890")或 t-oneshot 让模块加网； 4. 在与模块同网络的 PC（ip 为 192.168.1.100）上打开 hfs 服务器，

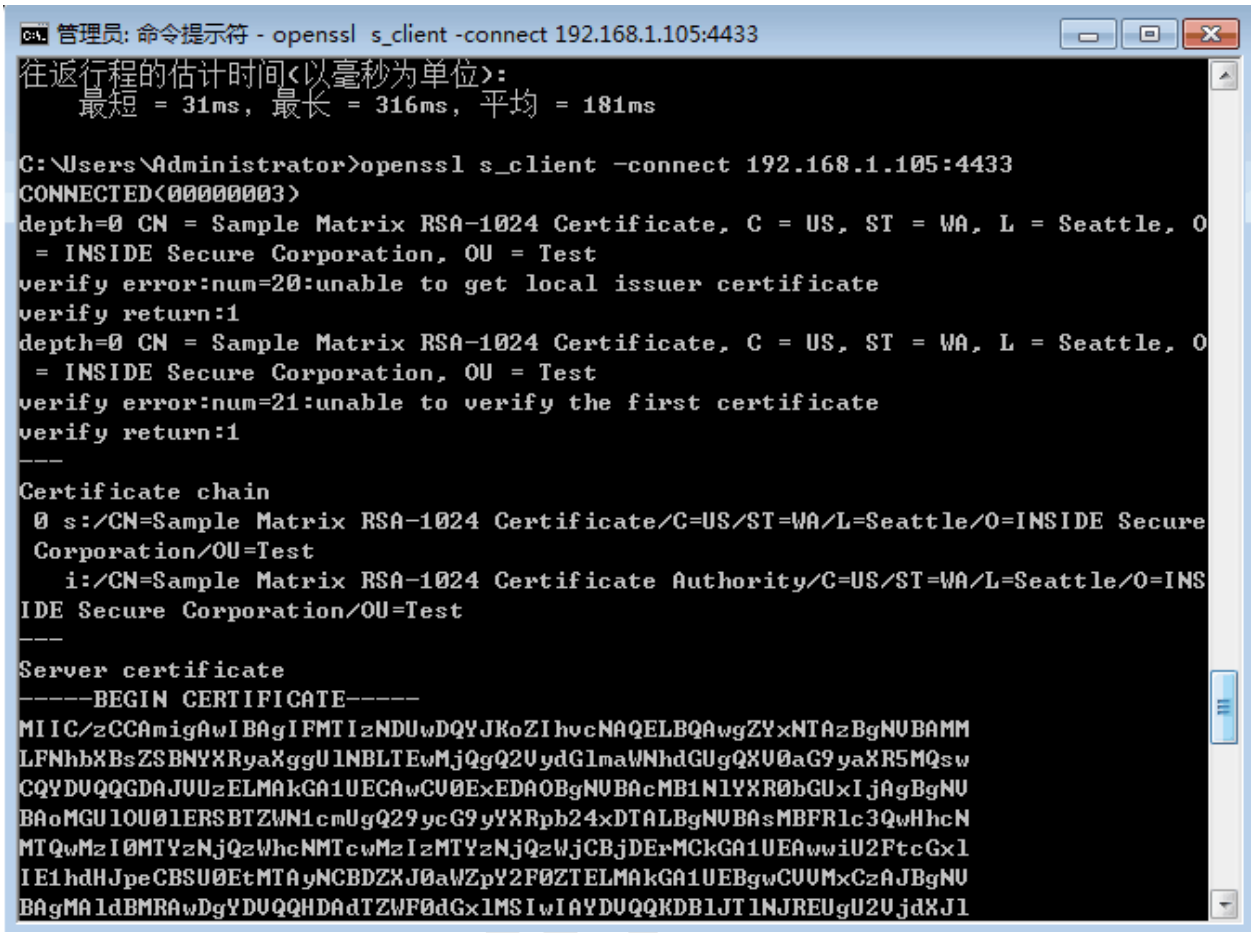
	端口 8080，并放置名称为 WM_W800_SEC.img 的固件； 5. 通过 uart0 发送 t-httpfwup=(http://192.168.1.100:80/w800_ota.img)， uart0 打印升级进度，模块升级成功后复位。升级压缩的 img。

5.7 **DEMO_SSL_SERVER** 操作步骤

功能描述	本例实现了 ssl server；允许其它客户端与设备侧建立 tls 连接； 注：需要打开 TLS_CONFIG_SERVER_SIDE_SSL，演示其他 DEMO 时需要关闭此宏开关。测试需要下载 openssl 或其他可以连接 ssl server 的工具
命令格式	t-ssl-server
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_ssl_server_init() tls_ssl_server_load_keys() tls_ssl_server_handshake() tls_ssl_server_recv() tls_ssl_server_send() tls_ssl_server_close()
涉及到的常用功能块	无
示例测试步骤	具体 demo 测试步骤如下： 1. 打开宏定义 DEMO_ SSL_SERVER 和 DEMO_CONNECT_NET； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应

	命令； 3. 通过 uart0 发送 t-connect("TEST_N40_6","1234567890")或 t-oneshot 让模块加网（ip 为 192.168.1.104）； 4. 通过 uart0 发送 t-ssl-server，uart0 返回 [CMD]t-ssl-server ssl server task Listening on port 4433 5. 在与模块同网络的 PC 上打开 openssl，执行命令 s_client -connect 192.168.1.104:4433，其中的 ip 地址及端口号为设备的 ip 地址及开放的相应端口号。 6. 此时模块的 uart0 打印 accept fd 1 tls_mem_alloc cp 2001ef88 tls_ssl_server_handshake rc 0 cp->time.tv_sec 0

下图为使用 openssl(需要用户自己安装)工具连接 ssl server 成功后的命令行页面信息。



5.8 **DEMO_WEBSOCKETS** 操作步骤

注：此 DEMO 下有两个示例，需要下载 WEBSOCKET_SERVER 测试服务器。

5.8.1 websocket 不加密方式的数据通信

功能描述	本例实现了使用 websocket 的方式与 websocket 服务器建立不加密连接并收发数据的过程；
命令格式	t-websockets
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	lws_create_context() lws_client_connect_via_info() lws_callback_on_writable()

	lws_service() lws_context_destroy() lws_write()

涉及到的常用功能块	`static void setAutoConnectMode(void)`
	`{`
	`  u8 auto_reconnect = 0xff;`
	
	`  tls_wifi_auto_connect_flag(WIFI_AUTO_CNT_FLAG_GET, &auto_reconnect);`
	`  if(auto_reconnect != WIFI_AUTO_CNT_ON)`
	`  {`
	`  auto_reconnect = WIFI_AUTO_CNT_ON;`
	`  tls_wifi_auto_connect_flag(WIFI_AUTO_CNT_FLAG_SET, &auto_reconnect);`
	`  }`
	`}`
	
	`static int isNetworkOk(void)`
	`{`
	`  struct tls_ethif* etherIf= tls_netif_get_ethif();`
	
	`  return etherIf->status;`
	`}`

示例测试步骤	1. 打开宏定义 DEMO_WEBSOCKETS 和 DEMO_CONNECT_NET，关闭 LWS_USE_SSL； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-connect("TEST_N40_6","1234567890")或 t-oneshot 让模块加网； 4. 如果使用 WEBSOCKET_SERVER 测试服务器，在与模块同网络的 PC（ip 为 192.168.1.100）上命令行运行 websocketd --port=8080 echo_client.bat； 5. 通过 uart0 发送 t-websockets，uart0 返回 [CMD]t-websocketsCLIENT_ESTABLISHED send {"msg_type":"keepalive"} 2 recv:websocket server send

	recv:{"msg_type":"keepalive"} 2

5.8.2 websocket 加密方式的数据通信

功能描述	本例实现了使用 websocket 的方式与 websocket 服务器建立加密连接并收发数据的过程；
命令格式	t-websockets
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	lws_create_context() lws_client_connect_via_info() lws_callback_on_writable() lws_service() lws_context_destroy() lws_write()

涉及到的常用功能块	`static void setAutoConnectMode(void)`
	`{`
	`  u8 auto_reconnect = 0xff;`
	`  tls_wifi_auto_connect_flag(WIFI_AUTO_CNT_FLAG_GET, &auto_reconnect);`
	`  if(auto_reconnect != WIFI_AUTO_CNT_ON)`
	`  {`
	`  auto_reconnect = WIFI_AUTO_CNT_ON;`
	`  tls_wifi_auto_connect_flag(WIFI_AUTO_CNT_FLAG_SET, &auto_reconnect);`
	`  }`
	`}`
	`static int isNetworkOk(void)`
	`{`
	`  struct tls_ethif* etherIf= tls_netif_get_ethif();`
	`  return etherIf->status;`
	`}`

示例测试步骤	1. 打开宏定义 DEMO_ WEBSOCKETS、DEMO_CONNECT_NET、LWS_USE_SSL，如果使用 WEBSOCKET_SERVER 测试服务器，请按 wm_websockets_demo.c 中 Notice 步骤修改代码（正

	规服务器测试时无需关注 Notice 中的步骤 3）； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-connect("TEST_N40_6","1234567890")或 t-oneshot 让模块加网； 4. 如果使用 WEBSOCKET_SERVER 测试服务器，在与模块同网络的 PC（ip 为 192.168.1.100）上命令行运行 websocketd --port=8080 --ssl --sslcert="certificate.pem" --sslkey="key.pem" echo_client.bat； 5. 通过 uart0 发送 t-websockets，uart0 返回 [CMD]t-websocketsCLIENT_ESTABLISHED send {"msg_type":"keepalive"} 1 recv:websocket server send recv:{"msg_type":"keepalive"} 1

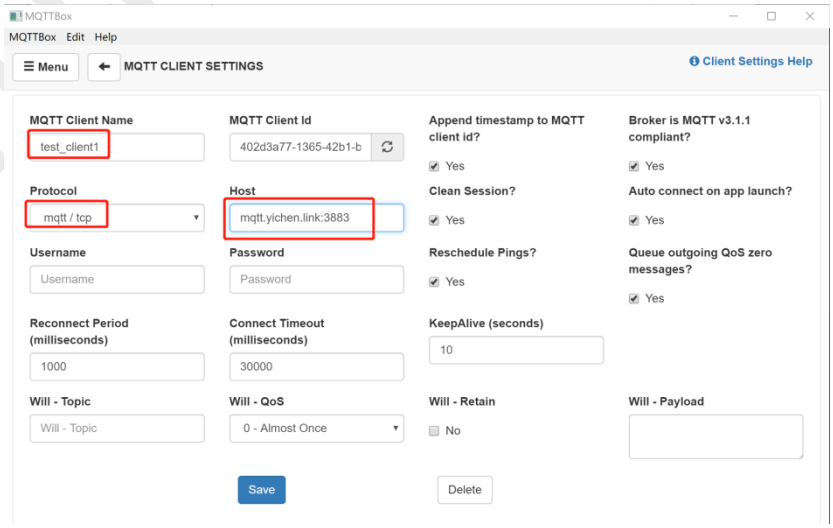
5.9 **DEMO_HTTPS** 操作步骤

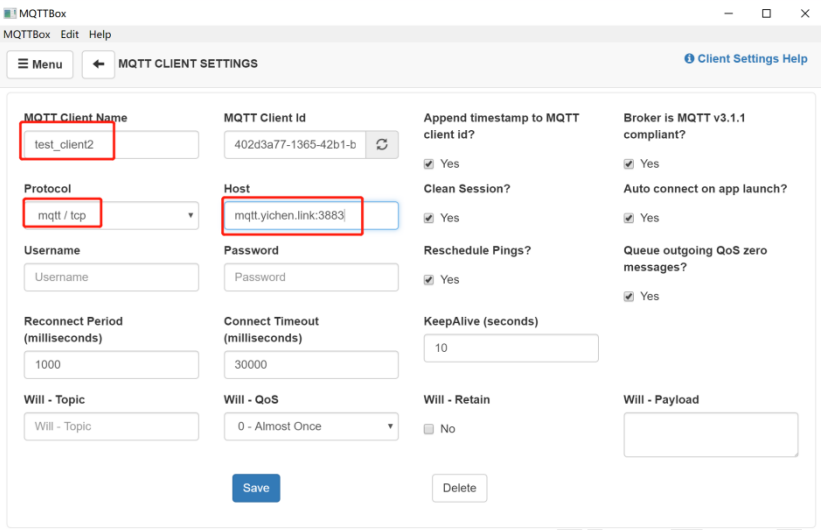
功能描述	本例实现了通过 https 的方式来获取网页数据的过程；
命令格式	t-https
涉及到的常用 api(其	Gethostbyname()

中 api 的具体释义请参考相关头文件注释)	HTTPWrapperSSLConnect() HTTPWrapperSSLSend() HTTPWrapperSSLRecv() HTTPWrapperSSLClose()
涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_HTTPS、DEMO_CONNECT_NET、TLS_CONFIG_HTTP_CLIENT 和 TLS_CONFIG_HTTP_CLIENT_SECURE; 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令; 3. 通过 uart0 发送 t-connect("TEST_N40_6","1234567890")或 t-oneshot 让模块加网（有外网）; 4. 通过 uart0 发送 t-https，uart0 会打印出 https://www.tencent.com/legal/html/zh-cn/index.html 的内容（注意 demo 中有打印信息）。

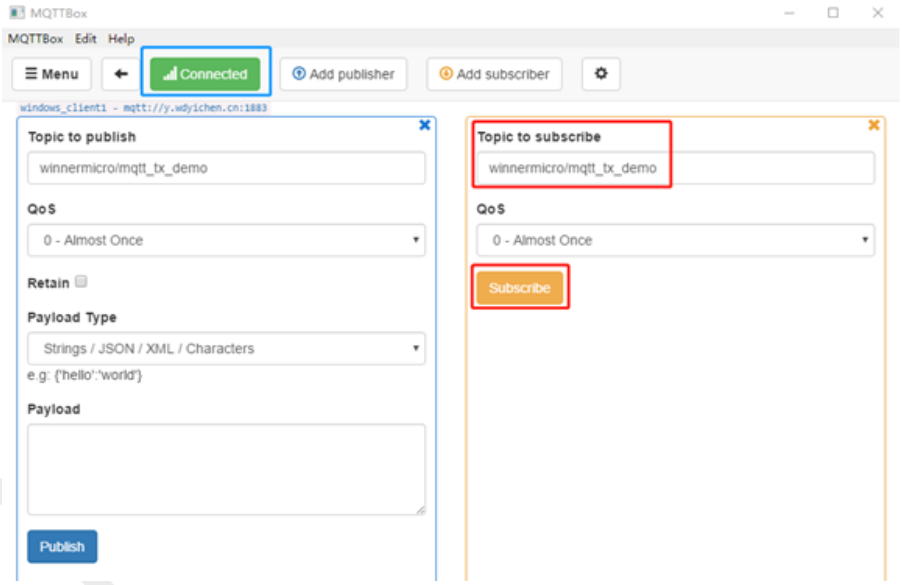
5.10**DEMO_MQTT** 操作步骤

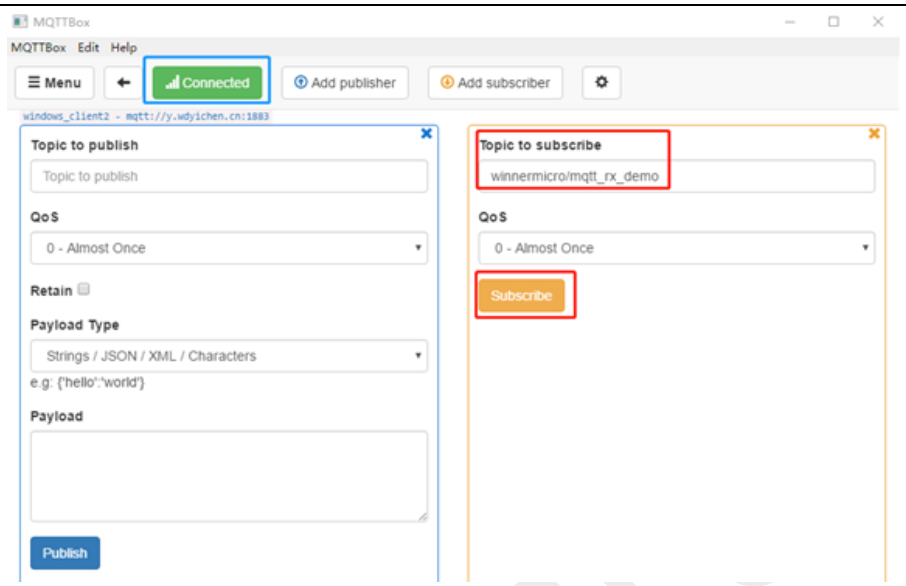
功能描述	本例实现了用例 mqtt 的方式与服务器建立连接并进行通信的过程;
命令格式	t-mqtt
涉及到的常用 api(其中 api	mqtt_init() mqtt_connect()

的具体释义请参考相关头文件注释)	MQTTParseMessageType() mqtt_subscribe() mqtt_publish() mqtt_parse_msg_id() mqtt_ping()
涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_MQTT 和 DEMO_CONNECT_NET； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-connect("TEST_N40_6","1234567890")或 t-oneshot 让模块加网（有外网）； 4. 通过 uart0 发送 t-mqtt，uart0 会打印出和“mqtt.yichen.link:3883”建立 mqtt 连接。 5. 下载 MQTTBox 软件，打开两个 MQTTBox 窗口，分别如下设置： 

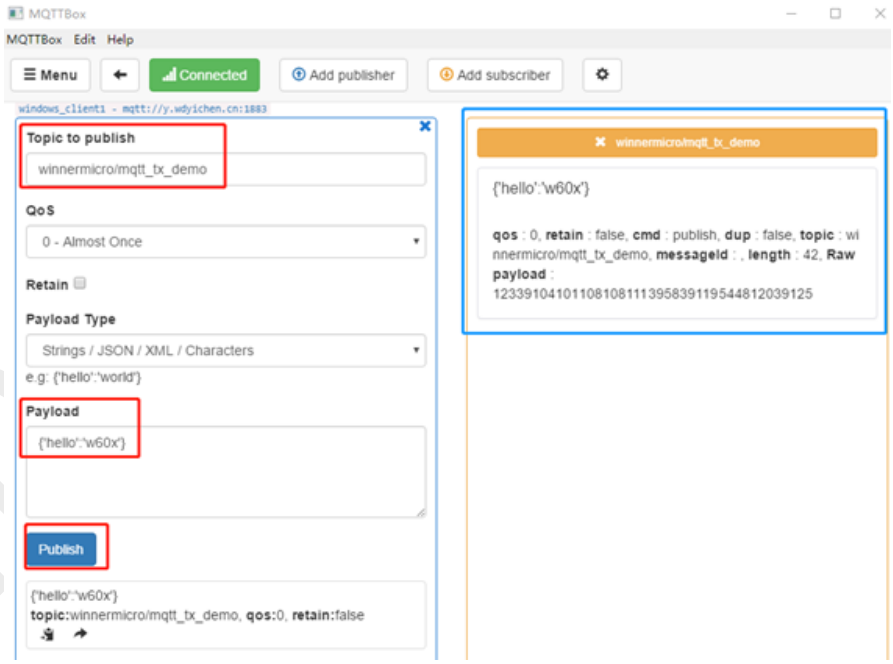


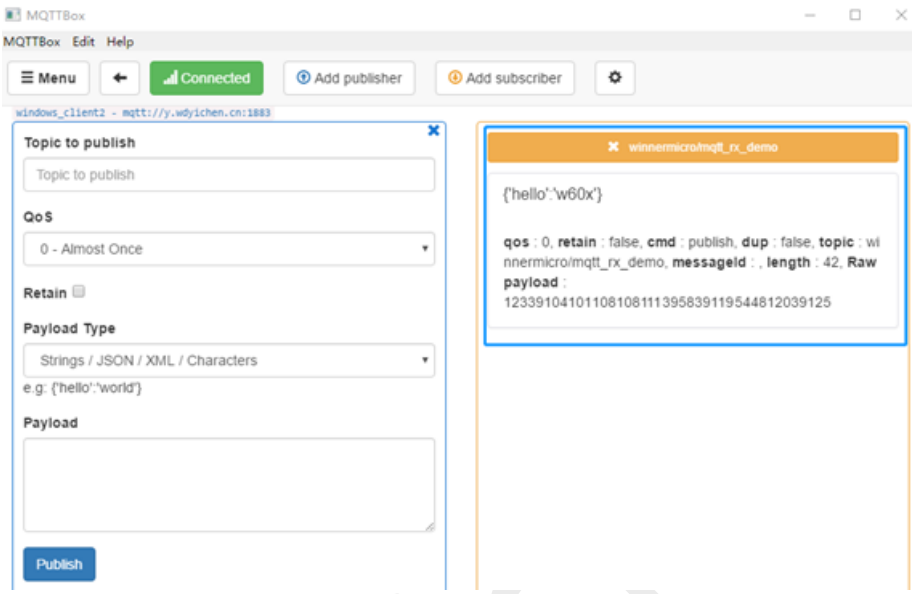
6. 点击 save 按钮即可和服务器建立连接，分别设置客户端订阅如下：





7. 在 windows_client1 客户端推送一条消息“{'hello':'w60x'}”，uart0 会打印该消息，windows_client2 客户端也会收到该消息。



5.11 **DEMO_DSP** 操作步骤

功能描述	本例实现了 DSP 的处理示例
命令格式	t-dsp(x),x 取值为 0,1,2,3,4
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	csky_fir_init_q15 csky_fir_q15 csky_mat_init_q31 csky_mat_mult_q31 csky_rfft_q15 csky_sin_q31 csky_var_q15
涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_DSP；

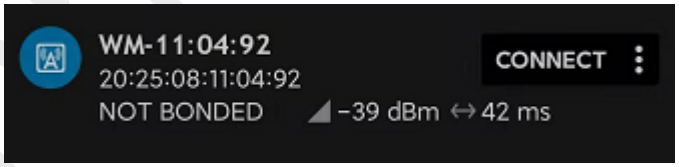
	2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-dsp(0)，uart0 打印：dsp fir run success! 4. 通过 uart0 发送 t-dsp(1)，uart0 打印：dsp matrix cal run success! 5. 通过 uart0 发送 t-dsp(2)，uart0 打印：dsp rfft run success! 6. 通过 uart0 发送 t-dsp(3)，uart0 打印：dsp sin run success! 7. 通过 uart0 发送 t-dsp(4)，uart0 打印：dsp variance run success!

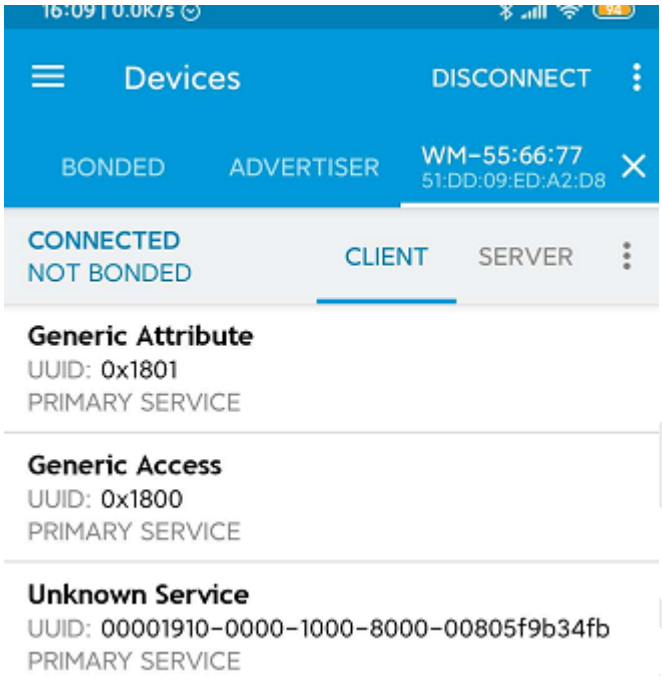
5.12 **DEMO_BT** 操作步骤

注：此 DEMO 下有四个示例。

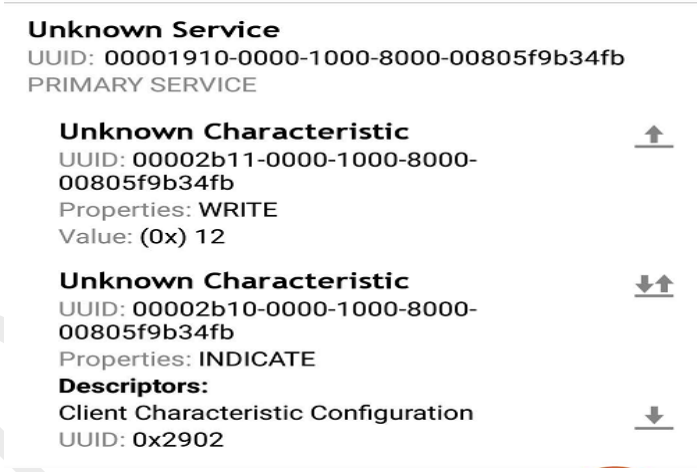
5.12.1 Ble server 示例

功能描述	本例实现了 W800 作 Ble server 的处理示例，此 DEMO 需要手机安装 nRF Connect（从应用商店下载即可）
命令格式	t-bt-on t-bt-off t-ble-server-on t-ble-server-off
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_open_peripheral_clock tls_bt_enable tls_bt_disable tls_close_peripheral_clock

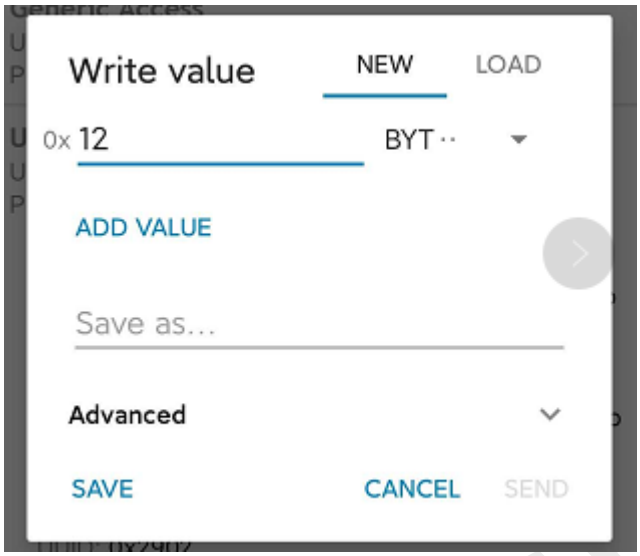
涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_BT（确认使用 SDK 发布时默认 ble 的 lib，确认 wm_config.h 中打开宏定义 TLS_CONFIG_BLE、关闭宏定义 TLS_CONFIG_BR_EDR）； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-bt-on，uart0 打印 init base application 相关信息； 4. 通过 uart0 发送 t-ble-server-on，成功后 uart0 打印 [WM_I] <0:00:07.188> ### wm_ble_server_api_demo_init success 5. 手机打开蓝牙，使用 nRF connect 扫描到设备（名称默认为 WM-XX:XX:XX，即模块 btmac 后六位）；  6. App 连接设备（注意：如果 app 主动断开连接，此 DEMO 需要设置 t-ble-server-off 和 t-ble-server-on 重新开启才能正常连接）；



7. App 点击查看 service；



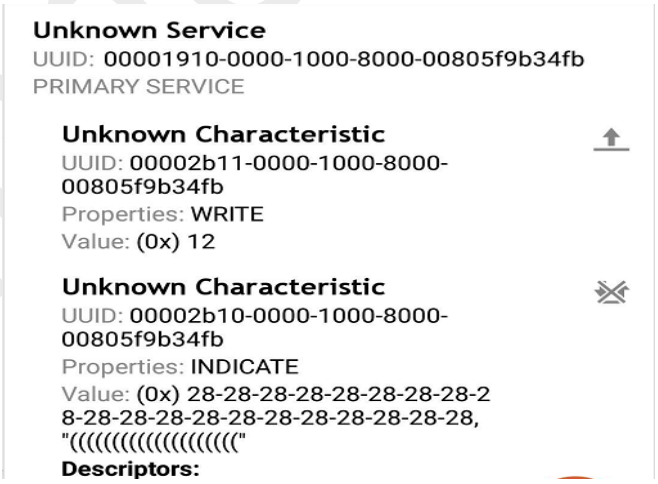
8. App 点击向上箭头，写特征值；



点击 SEND 后，uart0 打印 app 发的数据：###write cb12；

9. App 点击向下箭头，读取描述符，app 显示设备发的“Hello”；

10. App 点击上下箭头，使能 Indication；



11. App 再次点击上下箭头，关闭 Indications；

12. 通过 uart0 发送 t-ble-server-off，关闭 demo server 功能；

13. 通过 uart0 发送 t-bt-off，uart0 打印 bt system cleanup host 相关信息。

5.12.2 Ble client 示例

功能描述	本例实现了 W800 作 Ble client 的处理示例，此 DEMO 需要使用两个开发板，开发板 A 做 Ble server，开发板 B 做 Ble client。
命令格式	t-bt-on t-bt-off t-ble-server-on t-ble-server-off t-ble-client-on t-ble-client-off
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_open_peripheral_clock tls_bt_enable tls_bt_disable tls_close_peripheral_clock
涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_BT（确认使用 SDK 发布时默认 ble 的 lib，确认 wm_config.h 中打开宏定义 TLS_CONFIG_BLE、关闭宏定义 TLS_CONFIG_BR_EDR）； 2. 编译，两块开发板板升级固件，升级成功后，在 uart0 打印的控制台信息中能看到对应命令；

	3. 开发板 A 通过 uart0 发送 t-bt-on，uart0 打印 init base application 相关信息； 再通过 uart0 发送 t-ble-server-on； 4. 开发板 B 通过 uart0 发送 t-bt-on，uart0 打印 init base application 相关信息； 再通过 uart0 发送 t-ble-client-on； 5. 此时 B 会扫描，连接，并使能 A 的 Indication 功能。A 会不停的向 B 通过 Indication 发送数据。B 间隔一段时间在 uart0 打印统计结果。

5.12.3 Ble 广播示例

功能描述	本例实现了 W800 作 Ble server 的处理示例，此 DEMO 需要手机安装 nRF Connect（从应用商店下载即可）
命令格式	t-bt-on t-bt-off t-ble-adv=(type) type 定义为：1 可连接广播；2 不可连接广播；0 停止广播
涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	tls_open_peripheral_clock tls_bt_enable tls_bt_disable tls_close_peripheral_clock

涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_BT（确认使用 SDK 发布时默认 ble 的 lib，确认 wm_config.h 中打开宏定义 TLS_CONFIG_BLE、关闭宏定义 TLS_CONFIG_BR_EDR）； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-bt-on，uart0 打印 init base application 相关信息； 4. 通过 uart0 发送 t-ble-adv=(1)，手机可以扫描到蓝牙设备，并且可以连接成功； 5. 通过 uart0 发送 t-ble-adv=(2)，手机可以扫描到蓝牙设备，并且不能连接； 6. 通过 uart0 发送 t-ble-adv=(0)，手机扫不到蓝牙设备。

5.12.4 Ble 扫描示例

功能描述	本例实现了 W800 作 Ble server 的处理示例，此 DEMO 需要手机安装 nRF Connect（从应用商店下载即可）
命令格式	t-bt-on t-bt-off t-ble-scan=(type) type 定义为：1 开始扫描；0 关闭扫描
涉及到的常用 api(其中 api	tls_open_peripheral_clock tls_bt_enable

的具体释义请参考相关头文件注释)	tls_bt_disable tls_close_peripheral_clock
涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_BT（确认使用 SDK 发布时默认 ble 的 lib，确认 wm_config.h 中打开宏定义 TLS_CONFIG_BLE、关闭宏定义 TLS_CONFIG_BR_EDR）； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-bt-on，uart0 打印 init base application 相关信息 4. 通过 uart0 发送 t-ble-scan=(1)，uart0 打印扫描结果 5. 通过 uart0 发送 t-ble-scan=(0)，uart0 停止打印

5.13**DEMO_FATFS** 操作步骤

功能描述	本示例演示了如何使用设备来在 sd 卡上使用文件系统。 备注：若 sd 卡容易过大，可能会出现尝试多次才能格式化成功的现象。这不影响正常的读写，可以根据实际需要来调整使用多大的空间来建立文件系统，可通过修改函数 disk_ioctl() 中的 SDCardInfo.CardCapacity 的值来设置。
命令格式	t-fatfs
涉及到的常用 api(其中 api	wm_sdio_host_config()

的具体释义请参考相关头文件注释)	f_mkfs() f_mount() f_open() f_write() f_read() f_close()
涉及到的常用功能块	无
示例测试步骤	1，打开宏定义 DEMO_FATFS； 2，编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3，在开发板上接好 sd 卡，本示例使用的 IO 口为 PB06-PB11； 4，通过 uart0 发送 t-fatfs； 5，设备收到 uart0 的命令后会先格式化 sd 卡； 格式化成功后去挂载文件系统； 挂载成功后，建立一个新文件并向其中写入数据； 写入成功后，在 uart0 打印写入的数据，再从文件中读取数据； 读取成功后，在 uart0 打印读取的数据。

5.14**DEMO_MBEDTLS** 操作步骤

功能描述	本例实现了通过 https 的方式来获取网页数据的过程；
命令格式	t-mbedtls

涉及到的常用 api(其中 api 的具体释义请参考相关头文件注释)	
涉及到的常用功能块	无
示例测试步骤	1. 打开宏定义 DEMO_CONNECT_NET 和 DEMO_MBEDTLS； 2. 编译，升级成功后，在 uart0 打印的控制台信息中能看到对应命令； 3. 通过 uart0 发送 t-connect("TEST_N40_6","1234567890")或 t-oneshot 让模块加网（有外网）； 4. 通过 uart0 发送 t-mbedtls，uart0 会打印出 https://www.tencent.com/legal/html/zh-cn/index.html 的内容（注意 demo 中有打印信息）。