W800_SDK DEMO Operation Guide

V1.3

Beijing Lianshengde Microelectronics Co., Ltd. (winner micro)

Address: 18th Floor, Yindu Building, No. 67, Fucheng Road, Haidian District, Beijing

Tel: +86-10-62161900

Company website: www.winnermicro.com

Document modification record

Version revision time	revision history	Author review	
V0.1	2019/9/25 [C] Create document	Zhangwl	
V0.2	2020/7/2 Update I2C and I2S Demo multiplexing pins and illustrate	cuiyc	
V0.3	2020/7/8 Unified font	cuiyc	
V1.0	2020/8/4 Added ADC, DSP and BLE examples Cuiyc		
V1.1	2020/10/29 Updated BLE example	Pengxg	
V1.2	2021/4/16 Update httpget and httpfwup example parameters number	cuiyc	
V1.3	2021/11/4 Add TOUCHSENSOR example, add DEMO_HTTP parameter , renew DEMO_RSA test result	Chenzx	

Table of contents

1 **Introduction**

1.1 Purpose of writing

Provide code examples of related functions for software development engineers who are doing secondary development based on W80X chip SDK.

1.2 Intended audience

FAE, client-side software development engineer.

1.3 Definition of terms

2 DEMO overview

All DEMO-related macro definitions used in this document are in wm_demo.h. Required when running DEMO

Open the macro definition corresponding to this DEMO, and it is recommended to close irrelevant macro definitions. The DEMO demonstration needs to be performed under the console,

Turning on the DEMO_CONSOLE compile option opens the console.

DEMO_CONSOLE also controls the enabling of the AT command. If this macro is enabled, the AT command will be invalid;

When this macro is closed, the AT command takes effect.

The following three sections will introduce the test with examples of distribution network networking, hardware driver and application respectively.

Instructions.

3 Function description of distribution network DEMO

3.1 **DEMO_CONNECT_NET operation steps**

Note: There are five examples under this DEMO.

3.1.1 t-connect network

Function description	This example implements the function of making the WiFi device connect to the router with the specified name and password
command format	t-connect("ssid_name", "password")

Commonly used APIs involved (which
Please refer to the specific definition of api in
Refer to the relevant header file comments)
	tls_wifi_disconnect();
	tls_wifi_softap_destroy();
	tls_wifi_set_oneshot_flag(0);
	tls_mem_alloc();
	tls_netif_add_status_event();
	tls_wifi_connect();

The common function blocks involved	set the working mode of the device to sta mode:
	`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);`
	`}`

Example test steps	1. Open the macro definition DEMO_CONNECT_NET; 2. After compiling and upgrading successfully, you can see the correct information in the console information printed by uart0. on command; 3. Pass Pass uart0 send deliver t-connect("TEST_N40_6","1234567890") to let the module join A wireless network with name TEST_N40_6 and password 1234567890 (root Modify the name according to the existing network, this is just an example). **Note: All commands require carriage return and line feed, and English symbols are used in the commands;** 4. After the network is successfully added, uart0 will print the module ip.

3.1.2 t-oneshot (oneshot distribution network)

Function description	This example implements the function of enabling WiFi devices to perform one-key network configuration, in which the one-click network configuration includes Fang's oneshot distribution network and airkiss distribution network
command format	t-oneshot

Commonly used api (where api's tool For body definition, please refer to related header file comments)	tls_netif_add_status_event();
	tls_wifi_set_oneshot_config_mode();
	tls_wifi_set_oneshot_flag();

Commonly used functions involved piece	none
Example test steps	1. Open the macro definition DEMO_CONNECT_NET, (TLS_CONFIG_UDP_ONE_SHOT and TLS_CONFIG_UDP_LSD_SPECIAL is on by default); 2. After compiling and upgrading successfully, you can see the corresponding information in the console information printed by uart0 Order; 3. Send t-oneshot through uart0; 4. Add the mobile phone to the target network, install OneShotActivity (SDK ver2.0.0), Enter the correct ssid and password in the app interface, Click Start Configuration; 5. After the module is successfully networked, uart0 will print the ip.
App download address	http://www.winnermicro.com/html/1/156/158/497.html, Find oneshotconfig2.0.zip in the "Software Materials" tab under the page

3.1.3 t-oneshot (airkiss distribution network)

Function description	This example implements the function of enabling WiFi devices to perform one-key network configuration. The distribution network includes the official oneshot distribution network and airkiss distribution network
command format	t-oneshot

Commonly used apis involved (among which api's tools
For body definition, please refer to the relevant header file comments)
	tls_netif_add_status_event();
	tls_wifi_set_oneshot_config_mode();
	tls_wifi_set_oneshot_flag();

Commonly used function blocks involved	none
Example test steps	1. Open the macro definition DEMO_CONNECT_NET, TLS_CONFIG_AIRKISS_MODE_ONESHOT; 2. After compiling and upgrading successfully, the console information printed in uart0 You can see the corresponding command in the; 3. Send t-oneshot through uart0; 4. The mobile phone joins the target network (external network is required), open WeChat, close Note the public account [Lianshengde Microelectronics], click after entering the public account AirKiss distribution network under the product application, enter the configuration device to access the Internet page, set the correct Wi-Fi password, and click the connect button; 5. After the module is successfully networked, uart0 will print the ip.

3.1.4 t-webcfg (web configuration network)

Function description	This example implements the function of network configuration of the device through the built-in web page

command format	t-webcfg

Commonly used apis involved (among which api's tools
For body definition, please refer to the relevant header file comments)
	tls_netif_add_status_event();
	tls_wifi_set_oneshot_config_mode();
	tls_wifi_set_oneshot_flag();

Commonly used function blocks involved	none
Example test steps	1. Open the macro definition DEMO_CONNECT_NET, (TLS_CONFIG_WEB_SERVER_MODE defaults to opened); 2. After compiling and upgrading successfully, the console information printed in uart0 You can see the corresponding command in the; 3. Send t-webcfg through uart0; 4. Mobile phone or computer with wireless network card Enter "softap_XXXX" (where XXXX is the module mac The last 4 digits of the address), use a browser to access 192.168.1.1, Select the target network in the page List (if the target network is not found network, try refreshing the page or entering the ssid manually), then Enter the correct password in the pwd input box and click the save button; 5. After the module is successfully added to the network, uart0 will print the module ip, which is the same as the network The device can ping the module ip.

3.2 **DEMO_APSTA operation steps**

Function description This example implements the function of letting the device establish an apsta coexistence state, and at the same time as a sta to disconnect	

	Connect to the specified router, and when used as an ap, it also allows other sta devices to connect through the specified password. At the same time, the data forwarding function of udp is established, and the specific function is described in detail in the test steps;
Command format t-apsta("ssid_name","password","softapssid","87654321"), where 4	The parameters are the name and password of the router to be connected and the name and password when it is used as an ap.

Commonly used api (where api please explain Reference related headers file notes)	tls_netif_add_status_event();
	tls_wifi_set_oneshot_config_mode();
	tls_wifi_set_oneshot_flag();

Commonly used function block	none
Example test steps	1. Open the macro definition DEMO_APSTA; 2. After compiling and upgrading successfully, you can see the corresponding command in the console information printed by uart0; 3. Pass Pass uart0 send deliver t-apsta("TEST_N40_6","1234567890","softapssid","87654321 "); 4. uart0 will print the ip of softap and the ip of module sta; 5. Open the 65530 port of the UDP of the debugging assistant on PC1 on the same network as the module, and set hex display; 6. Use other PC2 to join softap, uart0 will print the device online; 7. Set PC2 to open the debugging assistant to monitor the UDP port 65530, set the hexadecimal display Show;

	8. Send t-asskt through uart0; 9. At this time, the debugging assistant on PC1 will receive the mac address repeatedly sent by sta; 10. After about 1 minute, the debugging assistant on PC2 will receive the mac repeatedly sent by softap address; 11. After the mobile phone is added to the softap, uart0 will print the device online, and the mobile phone can ping the channel devices under the router.

3.3 **DEMO_SOFT_AP** operation steps

Function description	This example implements the function of making the device work in softAP mode
command format	t-softap("softap1s","1234567890",6,4,1); 5 parameters are divided into Do not indicate the ap name, password, channel used, encryption method and password format; Encryption method: /*0:open, 1:wep64, 2:wep128,3:TKIP WPA,4: CCMP WPA, 5:TKIP WPA2 ,6:CCMP WPA2*/ Password format: /*key's format:0-HEX, 1-ASCII*/

Commonly used APIs involved (among them
For the specific definition of api, please refer to
Relevant header file comments)
	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();

The common function blocks involved set the working mode of the device to ap mode:	`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);`
	`}`

Example test steps	1. Open the macro definition DEMO_SOFT_AP; 2. After compiling and upgrading successfully, you can see in the console information printed by uart0 corresponding command; 3. Pass Pass uart0 send deliver t-softap("softap1s","1234567890",6,4,1) can make the device Create a hotspot named "softap1s" with password "1234567890" point; 4. The mobile phone can scan to the "softap1s" network, after adding softap, uart0 It will print the mobile phone mac.

3.4 **DEMO_WPS** operation steps

Note: There are two examples under this DEMO, the router needs to support wps, and the library that supports WPS needs to be obtained separately.

3.4.1 t-wps-start-pbc

Function description	This example implements the function of network configuration of the device through the built-in web page
command format	t-wps-start-pbc

Commonly used APIs involved (where API For the specific definition, please refer to the relevant header file notes)	tls_netif_add_status_event();
	tls_wifi_set_oneshot_config_mode();
	tls_wifi_set_oneshot_flag();

Commonly used function blocks involved	none
Example test steps	1. Open the macro definition DEMO_WPS; 2. After compiling and upgrading successfully, the console information printed by uart0 can be displayed. see the corresponding command; 3. Send t-wps-start-pbc through uart0, and press wps button, wait for uart0 to print [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

Function description	This example implements the network configuration of the device through wps pin function
command format	t-wps-start-pin
Commonly used apis involved (the api has tls_netif_add_status_event();	

For body definition, please refer to the relevant header file comments)	tls_wifi_set_oneshot_flag(); tls_wps_start_pin();
Commonly used function blocks involved	none
Example test steps	1. Open the macro definition DEMO_WPS; 2. After compiling and upgrading successfully, the console information printed in uart0 You can see the corresponding command in the; 3. Send t-wps-get-pin through uart0, uart0 prints pin code and automatically set to the module; 4. Enter the pin code in the router to start the connection; 5. Send t-wps-start-pin through uart0, wait for uart0 Print [CMD]t-wps-start-pinStart WPS pin mode... WiFi JOIN SUCCESS NET UP OK, Local IP: 192.168.1.101

3.5 **DEMO_SCAN** operation steps

Function description	This example implements the function of using the device to scan for surrounding wireless networks
command format	t-scan
Commonly used apis involved (among which api's tools For body definition, please refer to the relevant header file comments)	tls_wifi_scan_result_cb_register; tls_wifi_scan;
Commonly used function blocks involved	none
Example test steps	1. Open the macro definition DEMO_SCAN;

	2. After compiling and upgrading successfully, the console information printed in uart0 You can see the corresponding command in the; 3. Send t-scan through uart0; 4. After the device receives the command from uart0, it will scan the surrounding network. It will be printed to uart0 when the trace is complete.

4 Hardware driver class DEMO function description

4.1 **DEMO_UARTx** operation steps

Function description	This example realizes the function of serial port 1 echo data; Note: If you need to test other serial ports , then the function needs to be The macro definition "TLS_UART_1" in demo_uart_task() is modified to The corresponding serial port number, and the multiplexing function port is also modified to the corresponding multiplexing interface.
command format	t-uart=(baudrate,parity,stopbit), where the parameters are as their names said; Parity: 0, no parity; 1, odd parity; 2, even parity; Stopbit: 0, one stop bit; 1, two stop bits;

Commonly used apis involved (among which api's tools
For body definition, please refer to the relevant header file comments)
	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();

Commonly used function blocks involved	none
Example test steps	1. Open the macro definition DEMO_UARTx; 2. After compiling and upgrading successfully, the console information printed in uart0 You can see the corresponding command in the; 3. Send t-uart=(9600,0,0) through uart0 to modify uart1 parameter; 4. The serial port tool sets the baud rate to 9600, check digit NONE, number According to bit 8, stop bit 1, open uart1 to send data, the module will Print the received data from uart1 (PB06_TX PB07_RX).

4.2 **DEMO_GPIO** operation steps

Note: There are two examples under this DEMO.

4.2.1 t-GPIO

Function description	This example implements the use of PB6 to demonstrate the input and output of GPIO and the Pull the floating function
command format	t-gpio
Commonly used apis involved (among which api's tools For body definition, please refer to the relevant header file comments)	tls_gpio_cfg(); tls_gpio_read();

	tls_gpio_write();
Commonly used function blocks involved	none
Example test steps	1. Open the macro definition DEMO_ GPIO; 2. After compiling and upgrading successfully, the console information printed in uart0 You can see the corresponding command in the; 3. Send t-gpio through uart0, uart0 will print the test result 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

Function description	This example implements the function of using PA1 as an input pin to generate an interrupt;
command format	t-gpioirq

Commonly used apis involved (among which api's tools
For body definition, please refer to the relevant header file comments)
	tls_gpio_cfg();
	tls_gpio_isr_register();
	tls_gpio_irq_enable();
	tls_get_gpio_irq_status();
	tls_clr_gpio_irq_status();

Commonly used function blocks involved	none
Example test steps	1. Open the macro definition DEMO_ GPIO;

	2. After compiling and upgrading successfully, the console information printed in uart0 You can see the corresponding command in the; 3. Send t-gpioirq through uart0, pull down PA1, uart0 Print int flag =1 after int io =0 4. Pull PA1 high, uart0 prints int flag =1 after int io =1

4.3 **DEMO_FLASH** operation steps

Function description	This example implements the read and write functions of the internal flash. The user does not need to call the erase function before writing, because the write function has internal With integrated erase function.
command format	t-flash
Commonly used apis involved (among which api's tools For body definition, please refer to the relevant header file comments)	tls_fls_write(); tls_fls_read();
Commonly used function blocks involved	none
Example test steps	1. Open the macro definition DEMO_ FLASH; 2. After compiling and upgrading successfully, the console information printed in uart0 You can see the corresponding command in the; 3. Send t-flash through uart0, uart0 will print success.

4.4 **DEMO_ENCRYPT** operation steps

Function description	This example introduces how to use the encryption hash and other related functions that come with the sdk;
command format	t-crypt

Encryption Algorithms Involved	RNG_hard_demo();
	rc4_hard_demo();
	aes_hard_demo();
	des_hard_demo();
	des3_hard_demo();
	crc_hard_demo();
	md5_hard_demo();
	sha1_hard_demo();

Commonly used function blocks involved	
Example test steps	1. Open the macro definition DEMO_ ENCRYPT; 2. After compiling and upgrading successfully, you can see the console information printed by uart0 to the corresponding command; 3. Send t-crypt through uart0, uart0 will print [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** operation steps

Function description	This example implements the steps of using the rsa algorithm of different lengths;
command format	t-rsa

the length of the rsa computation involved	rsa128_demo();
	rsa256_demo();
	rsa512_demo();
	rsa1024_demo();
	rsa2048_demo();

Commonly used function blocks involved	none
Example test steps	1. Open the macro definition DEMO_ RSA; 2. After compiling and upgrading successfully, the console information printed in uart0 You can see the corresponding command in the; 3. Send t-rsa through uart0, uart0 will print [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** operation steps

Function description	This example implements the steps of using the built-in RTC in the chip;
command format	t-rtc
Commonly used apis involved (among which api's tools For body definition, please refer to the relevant header file comments)	tls_set_rtc(); tls_rtc_isr_register();

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

Commonly used function blocks involved	none
Example test steps	1. Open the macro definition DEMO_RTC; 2. After compiling and upgrading successfully, the console information printed in uart0 You can see the corresponding command in the; 3. Send t-rtc through uart0 to turn on rtc clock, 20 seconds uart0 will print rtc clock to indicate entering rtc interrupt.

4.7 **DEMO_TIMER** operation steps

Function description	This example implements the use of the built-in hardware timer of the chip; Remarks: The chip has a total of 5 built-in timers, the related api "tls_timer_create" will return an unused timer handle number; the time unit of the timer can be set to two microseconds or milliseconds.
command format	t-timer
Commonly used apis involved (among which api's tools For body definition, please refer to the relevant header file comments)	tls_timer_create(); tls_timer_start();
Commonly used function blocks involved	none
Example test steps	1. Open the macro definition DEMO_ TIMER; 2. After compiling and upgrading successfully, the console information printed in uart0 You can see the corresponding command in the;

	3. Send t-timer through uart0 to turn on the timer, uart0 every 2 Seconds print timer irq to indicate entering the timer interrupt.

4.8 **DEMO_PWM** operation steps

Function description	This example implements the use of the built-in PWM peripheral in the chip;
command format	t-pwm=(1,250,99,4,0) The first parameter is the channel number, including Two groups of multiplexing, serial numbers 0-4 correspond to PB00, PB01, PB02, PB03, PA07 have five channels, 5-9 correspond to PB19, PB20, PA00, PA01, PA04; the second parameter is the expected output pwm Frequency; the third parameter is the duty cycle, for example, 99 here means the actual The duty cycle is 99/255; the fourth parameter represents the current mode, where 4 Indicates independent mode, that is, only this pwm output waveform; the fifth parameter Indicates the number of waveform cycles output, where 0 means continuous output waveform. Tool The body definition can refer to the above comment of the function pwm_demo().

Commonly used apis involved (among which api's tools
For body definition, please refer to the relevant header file comments)
	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();

Commonly used function blocks involved	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();

Example test steps	1. Open the macro definition DEMO_PWM; 2. After compiling and upgrading successfully, the console information printed in uart0 You can see the corresponding command in the; 3. Send t-pwm=(1,250,99,4,0) through uart0, oscilloscope The amount of PB01 can be measured to 250Hz, and the duty cycle is about 39% (99/255) of the waveform.

4.9 **DEMO_PMU** operation steps

Note: There are two examples under this DEMO.

4.9.1 t-pmuT0

Function description	This example implements the control device to enter the standby low-power mode and timing function to wake it up;
command format	t-pmuT0

Commonly used apis involved (among which api's tools For body definition, please refer to the relevant header file comments)	tls_pmu_timer0_isr_register();
	tls_pmu_timer0_start();
	tls_pmu_standby_start();

Commonly used function blocks involved	none
Example test steps	1. Open the macro definition DEMO_PMU; 2. After compiling and upgrading successfully, the console information printed in uart0 You can see the corresponding command in the; 3. Send t-pmuT0 module to start timer0 through uart0 Enter standby, the uart0 printing module resets in about 10 seconds, Indicates timer0 interrupt wake-up.

4.9.2 t-pmuT1

Function description	This example achieves
command format	t-pmuT1

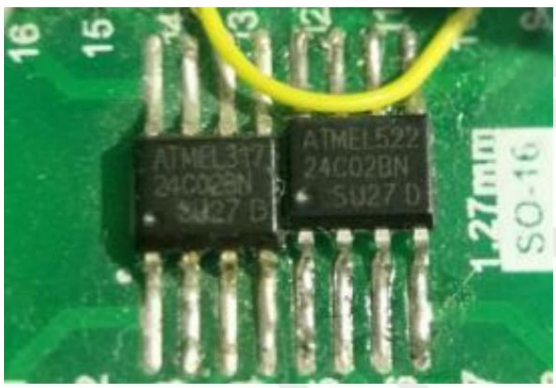
Commonly used apis involved (among which api's tools For body definition, please refer to the relevant header file comments)	tls_pmu_timer1_isr_register();
	tls_pmu_timer1_start();
	tls_pmu_standby_start();

Commonly used function blocks involved	none

Example test steps	1. Open the macro definition DEMO_PMU; 2. After compiling and upgrading successfully, the console information printed in uart0 You can see the corresponding command in the; 3. Send t-pmuT1 module to start timer1 through uart0 Enter standby, the uart0 print module resets in about 5 seconds, the table It shows that timer1 is interrupted to wake up.

4.10 DEMO_I2C operation steps

Note: This DEMO requires AT24CXX chip



Function description	This example implements the use of the built-in i2c module to the at24cxx device to perform the process of writing and reading data; Note: The default interface pull-up resistor on the test board shown in the figure above, if you use If the user uses other i2c devices to test unsuccessfully, they need to check the connection Whether there is a pull-up or pull-down on the two lines of the road. Is it not possible to have the following pull-up resistors.
command format	t-i2c
The commonly used apis involved (the api has wm_i2c_scl_config();	

For body definition, please refer to the relevant header file comments)	wm_i2c_sda_config();
	tls_i2c_init();
	tls_i2c_write_byte();
	tls_i2c_wait_ack();
	tls_i2c_read_byte();

Commonly used function blocks involved	AT24CXX_ReadOneByte(); AT24CXX_ReadLenByte(); AT24CXX_WriteOneByte(); AT24CXX_Write();
Example test steps	1. Open the macro definition DEMO_I2C; 2. After compiling and upgrading successfully, the console information printed in uart0 You can see the corresponding command in the; 3. The module PIN is connected to the AT24CXX chip: 4. PA01 is connected to SCL, PA04 is connected to SDA, GND is connected to GND, VCC to 3.3v 5. Send t-i2c through uart0, uart0 returns [CMD]t-i2c AT24CXX check success read data is: AT24CXX I2C TEST OK

4.11 DEMO_I2S operation steps

Function description	This DEMO is used to demonstrate the device for data communication in i2s format. Another corresponding master device or slave device is required to cooperate with sending or receiving receive data. Remarks: Wiring method ck-ck ws-ws, di-do, do-di
command format	* @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) */

Commonly used apis involved (among which api's tools
For body definition, please refer to the relevant header file comments)
	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();
Commonly used function blocks involved	none
Example test steps	1. Open the macro definition DEMO_I2S; 2. After compiling and upgrading successfully, the console information printed in uart0 You can see the corresponding command in the; 3. Connect the corresponding pin of the device to the corresponding pin of the test device: The pins on the device side are defined as: ck--PB08, ws--PB09, di--PB10, do--PB11, after the four signal lines are connected The two communication devices should be grounded together. 4. Send t-i2sioinit through uart0 of the two devices to make the device initialize initialize io; 5. Sending t-i2s=(0,2,44100,16,0,1) through uart0 will make Using DMA to receive data, the device will be in the slave state; 6. Sending t-i2s=(0,1,44100,16,0,1) through uart0 will make Using DMA to send data, the device at this time will be in the master status (slave side will print full duplex and half duplex the comparison result of the received data); 7. Reset the device and reinitialize io; 8. Sending t-i2s=(0,4,44100,16,0,1) through uart0 will make Using DMA to receive data, the device will be in the slave state;

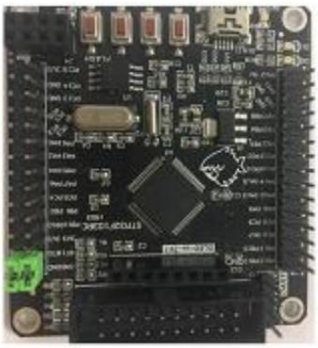
	9. Sending t-i2s=(0,3,44100,16,0,1) through uart0 will make Using DMA to send data, the device at this time will be in the master status (full duplex and half duplex reception will be printed at both ends data comparison).

4.12 **DEMO_MASTER_SPI** operation steps

Function description	In this example, the chip side is used as the master to communicate with the slave side through the spi interface. The process of sending and receiving data; Note: When testing this example, if necessary, you can connect the serial port on the four signal lines Ten ohm resistor to ensure normal communication. This DEMO needs to download the peer code;
command format	t-mspi-s t-mspi-r

Commonly used APIs involved (where API
For the specific definition, please refer to the relevant header
file notes)
	tls_spi_trans_type();
	tls_spi_setup();
	tls_spi_write();
	tls_spi_read();

Commonly used function blocks involved	none
Example test steps	1. Open the macro definition DEMO_ MASTER_SPI; 2. After compiling and upgrading successfully, the console information printed by uart0 can be displayed. see the corresponding command; 3. Open with keil

	STM32_SOC_TEST_SLAVE_SPI\Project\ STM32F10x_StdPeriph_Template\MDK-ARM\Project After compiling, upgrade stm32 through jlink; Note: STM32 **development board model: STM32_Mini_V2.0**   4. The module PIN is connected to the peer stm32 (PA9tx, PA10rx as a print mouth): PB4 is connected to PB12(cs), PB2 is connected to PB13(ck), PB3 is connected to PB14(so), PB5 to PB15(si), GND to GND; 5. Send 1500 numbers by sending t-mspi-s(1000000,0) through uart0 According to, uart0 of stm32 prints down data len: 1500; 6. Send t-mspi-r through uart0, the module uart0 prints [CMD]t-mspi-rSPI Master receive 1500 bytes, modeA, little endian rcv data len: 1500.

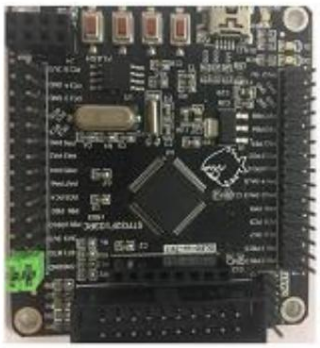
4.13 DEMO_ADC operation steps

Function description	This example implements the functions of the ADC for chip temperature acquisition and external input voltage detection.

	can
command format	t-adcvolt(x), x value 0 means channel 0, 1 means channel 1, 8 means difference Minute; t-adctemp
Commonly used APIs involved (where API For the specific definition, please refer to the relevant header file notes)	adc_temp wm_adc_config adc_get_inputVolt
Commonly used function blocks involved	none
Example test steps	1) For the chip temperature test, serial port 0 directly enters the t-adctemp command to execute line to return the current chip temperature: tem:xxx 2) For input voltage, serial port 0 input command: Single-ended test: t-adcvolt(0) or t-adcvolt(1) Differential test: t-adcvolt(8) After the execution is complete, return: chan:x, xxxx(mV) or x.xxx(V)

4.14 DEMO_SLAVE_SPI operation steps

Function description	This example realizes that when the device acts as a slave, the data is performed with the master device through the HSPI interface. the process of communication; Note: This DEMO uses the W800_ARDUINO_V1.0 development board, and requires Download the peer code, STM32 development board model: STM32_Mini_V2.0. Down The picture shows the host development board;

	 
command format	t-sspi(0)

Commonly used APIs involved (where API
For the specific definition, please refer to the relevant header
file notes)
	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();

Commonly used function blocks involved	none
Example test steps	1. Open the macro definition DEMO_ SLAVE_SPI 2. After compiling and upgrading successfully, the console information printed by uart0 can be displayed. see the corresponding command; 3. Open stm32_ucos_ri\uCOSDemo with keil and compile and pass jlink upgrades stm32; 4. The module PIN is connected to the peer stm32 (PA9tx, PA10rx as a print mouth): PB09 is connected to PA4(cs), PB06 is connected to PA5(ck), PB11 is connected to PA6(mi), PB10 is connected to PA7(mo), PB07 is connected to PA2(cts),

	GND to GND 5. Send t-sspi(0) through uart0 6. Reset stm32, module uart0 prints: HspiRxCmdCb rx[5] : 5a 00 05 01 60 RX ok 100 RX ok 200 RX ok 300 7. Stm32 prints: ###kevin debug... tx start cmd kevin debug TX_BUFF_AVAIL = 3, cmdlen=8 RX ok 100 RX ok 200 RX ok 300

4.15 DEMO_SDIO_HOST operation steps

Function description	In this example, the sd card is read and written through the built-in sdio interface of the chip. the process of making;
command format	t-sdh
Commonly used APIs involved (where API For the specific definition, please refer to the relevant header	wm_sd_card_set_bus_width(); wm_sd_card_set_blocklen();

file notes)	wm_sd_card_block_write();
	wm_sd_card_block_read();
	wm_sd_card_blocks_write();
	wm_sd_card_blocks_read();

Commonly used function blocks involved	none
Example test steps	1. Open the macro definition DEMO_SDIO_HOST; 2. After the compilation and upgrade are successful, the console information printed by uart0 can be displayed. see the corresponding command; 3. Connect the sd card on the development board. The IO port used in this example is PB06-PB11; 4. Send t-sdh through uart0; 5. After the device receives the command from serial port 0, it uses interrupt mode and dma respectively. way to write and read data to the specified block of the sd card; If the data and the read data are the same, a message related to the success of the test will be printed; If there is a difference, a failure-related message will be printed.

4.16 DEMO_TOUCHSENSOR operation steps

Function description	This example implements the touchsensor function of the chip;
command format	t-touch(touchnum); touchnum input parameter is decimal, every 1bit Corresponding to 1 touchsensor, such as 3 corresponding to open touch2 and touch1, 7 Corresponds to open touch3, touch2 and touch1. Note: touchnum When it is 0, all 15 touchsensors will be turned on; when only 1 touch is turned on

	, touch1 also needs to be opened, for example, 5 corresponds to open touch3 and touch1.
Commonly used APIs involved (where API For the specific definition, please refer to the relevant header file notes)	none
Commonly used function blocks involved	none
Example test steps	1. Open the macro definition DEMO_TOUCHSENSOR; 2. After the compilation and upgrade are successful, the console information printed by uart0 can be displayed. see the corresponding command; 3. Connect the touchio of the development board to the touchpad; 4. Send t-touch(9) to the development board through uart0; 5. Click the touch4 position of the touchpad, uart0 will print the triggered key4.

5 Function description of application class DEMO

5.1 **DEMO_STD_SOCKET_CLIENT operation steps**

Note: Sending demohelp module uart0 through uart0 will return console information.

Function description	This example implements the use of standard socket functions to create a tcp client to communicate with The process of data communication with the server side on the PC in the local area network; the device side As a client, it will print out the length of the data received from the server, and Send data through the serial port;
command format	t-sockc(port, ip) t-skcsnd(len, uart_trans)
Commonly used APIs involved (including api socket();	

For the specific definition, please refer to the relevant header
 	
file notes)
	connect();
	
	closesocket();
	
	recv();
	
	tls_uart_write();

Commonly used function blocks involved	none
Example test steps	1. Open the macro definitions DEMO_STD_SOCKET_CLIENT and DEMO_CONNECT_NET; 2. After compiling and upgrading successfully, in the console information printed by uart0 Can see the corresponding command; 3. Pass Pass uart0 send deliver t-connect("TEST_N40_6","1234567890") let the module add net; 4. On the PC on the same network as the module (ip is 192.168.1.100) Open debugging assistant tcp server port number 1000; 5. Send t-sockc(1000,192.168.1.100) through uart0 The module creates a tcp client to connect to the peer server, after the connection is successful uart0 will print socket num; 6. The server sends data, after the module receives the data, uart0 will print the received data The length of data is accumulated each time; 7. Send t-skcsnd(0,1) through uart0 to set the transparent using uart1 pass; 8. Serial port tool set baud rate 115200, parity bit NONE, data

	Bit 8, stop bit 1 Open uart1, through uart1 and server Two-way transparent transmission;

5.2 **DEMO_STD_SOCKET_SERVER operation steps**

Function description	This example implements the use of standard socket functions to create a tcp server to communicate with The process of data communication with the client on the PC in the local area network; After the tcp server is successfully established on the device side, you can open the tool on the PC to establish The client side establishes a connection with it; after the connection is established successfully, the The PC sends data to the device, and the device will print the accumulation of the received data after receiving it length value; you can also send transparent data to the device through the serial port, so that the data is transmitted Input to the client of the PC;
command format	t-socks(port) t-skssnd(sock,len,uart_number)

Commonly used APIs involved (where API
For the specific definition, please refer to the relevant header
file notes)
	socket();
	connect();
	closesocket();
	recv();
	bind();
	listen();
	accept();
	send();
	tls_uart_write();

Commonly used function blocks involved	none
Example test steps	1. Open the macro definitions DEMO_STD_SOCKET_ SERVER and DEMO_CONNECT_NET; 2. After compiling and upgrading successfully, in the console information printed by uart0 Can see the corresponding command; 3. Pass Pass uart0 send deliver t-connect("HUAWEI-6SEWE5","123456789") or t-oneshot allows modules to be screened; 4. Send t-socks(2000) through uart0 to let the module create tcp server, uart0 will print the listening port; 5. Open the debugging assistant on the PC on the same network as the module and create tcp The client (set the ip and port number of the module) connects to the module server, After the connection is successful, uart0 will print the client information (module server Connect up to 7 clients); 6. The client sends data. After the module receives the data, uart0 will print the received data. Corresponding data length of the connection, accumulated each time; 7. Send t-skssnd(1,16,0) through uart0 Send fixed data of length 16, the client can receive the data; 8. Send t-skssnd(1,0,1) through uart0 to set No. 1 connection on uart1 transparent transmission; 9. Serial port tool set baud rate 115200, parity bit NONE, data Bit 8, stop bit 1 Open uart1, communicate with client through uart1

	Two-way transparent transmission.

5.3 **DEMO_SOCKET_CLIENT_SERVER** operation steps

There are two examples under this test macro switch, which are the device as the tcp client and the device as the tcp server.

5.3.1 t-client

Function description	This example enables the device to connect to the router with the specified name and password, and establishes tcp The client, then connect to the tcp server with the specified address and port and perform data communication the process of;
command format	t-client("ssid","password",port,"ip")

Commonly used APIs involved (among them
For the specific definition of api, please refer to
Relevant header file comments)
	socket();
	connect();
	closesocket();
	recv();
	tls_wifi_connect();

Commonly used function blocks involved	`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;`
	`}`

Example test steps	1. Open the macro definition DEMO_SOCKET_CLIENT_SERVER, DEMO_CONNECT_NET; 2. After compiling and upgrading successfully, you can see in the console information printed by uart0 corresponding command;

	3. Create a tcp server on the PC and set the listening port to 8080. 4. Send t-client through uart0("TEST_N40_6","1234567890", 8080, "192.168.1.100"); the four parameters are the Router name, password, port number and ip address of the server to be connected. 5. After the device receives the command from serial port 0, it will connect to the router, and the connection route is successful After that, it will connect to the server; after the connection to the server is successful, a message will be sent to it; The user can see this message on the server side, and can return a message through the server at this time. The message is sent to the device, and the device will print accordingly after receiving the message; 6. The device will always be in the process of sending and receiving, sending and receiving again, until the connection is made. disconnect.

5.3.2 t-server

Function description	This example implements the device to connect to the router with the specified name and password, and establishes a tcp server, The process of receiving the connection from the client and communicating with it;
command format	t-server("ssid","password",port)

Commonly used
api (where api's
For specific definitions, please refer to
Relevant header file comments)
	socket();
	connect();
	closesocket();
	recv();
	bind();
	listen();
	accept();

	send();
	
	tls_wifi_connect();

Commonly used functions involved
 	
energy block
	`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;`
	`}`

Example test steps	1. Open the macro definition DEMO_SOCKET_CLIENT_SERVER, DEMO_CONNECT_NET; 2. After compiling and upgrading successfully, you can see the corresponding command in the console information printed by uart0 make; 3. Send t-server through uart0("TEST_N40_6","1234567890", 8080,); the three parameters are the name, password, service of the router to be connected port number of the device. 4. After the device receives the command from serial port 0, it will connect to the router. After the connection is successful, it will call Print the ip address and set up a tcp server, and listen to its own port 8080; 5. Use the tcp tool to create a tcp client on the PC in the same local area network to Connect the ip and port of this server; after the establishment is successful, you can send data to it through the tool according to; 6. After the server receives the data, it will send "message from server" string. 7. The device will always be in the process of receiving, sending, receiving and sending again until the connection is disconnected open.

5.4 **DEMO_UDP operation steps**

Note: There are three examples under this DEMO, which need to use a packet capture network card

5.4.1 UDP broadcast

Function description	This example implements the process of broadcasting data through udp;
command format	t-udp(mode,port,ip) t-sndudp(len)
Commonly used api (where api's For specific definitions, please refer to Relevant header file comments)	tls_netif_get_ethif() socket() bind() closesocket() setsockopt() recvfrom() sendto();
Commonly used functions involved energy block	`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)));`
Example test steps	1. Open the macro definitions DEMO_UDP and DEMO_CONNECT_NET; 2. After compiling and upgrading successfully, you can see the corresponding command in the console information printed by uart0 make; 3. Send t-connect("TEST_N40_6","1234567890") through uart0 or t-oneshot allows modules to be screened; 4. Send t-udp(0,1000,0)uart0 to print through uart0

	udp demo,cast:0,port:1000 localip : 192.168.1.104 local port :3000 5. Open the debugging assistant udp port 1000 on the PC on the same network as the module; 6. Send t-sndudp(10) through uart0, the packet capture network card can capture the module to the route The destination of the device is the Ethernet Broadcast package, and the debugging assistant 10 data received; 7. The debugging assistant sends data. After the module receives the data, uart0 will print the address and data length. Spend.

5.4.2 UDP Unicast

Function description	This example implements the process of unicasting data to a specified device through udp;
command format	t-udp(mode,port,ip) t-sndudp(len)
Commonly used api (where api's For specific definitions, please refer to Relevant header file comments)	tls_netif_get_ethif() socket() bind() closesocket() setsockopt() recvfrom() sendto();
Commonly used functions involved energy block	`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)));`

Example test steps	1. Open the macro definitions DEMO_UDP and DEMO_CONNECT_NET; 2. After compiling and upgrading successfully, you can see the corresponding command in the console information printed by uart0 make; 3. Send t-connect("TEST_N40_6","1234567890") through uart0 or t-oneshot allows modules to be screened; 4. Send t-udp(1,1001,192.168.1.100) through uart0 and uart0 will print udp demo,cast:1,port:1001 localip : 192.168.1.104 local port :3000 5. Open the debugging assistant on the PC on the same network as the module (ip is 192.168.1.100) connect udp port 1001; 6. Send t-sndudp(10) to capture packets through uart0, the network card can capture the module to the router The destination is the package of the PC network card, and the debugging assistant has received 10 numbers according to; 7. The debugging assistant sends data. After the module receives the data, uart0 will print the address and data length. Spend.

5.4.3 UDP Multicast

Function description	This example implements the process of multicasting data to the outside through udp;
command format	t-udp(mode,port,ip) t-sndudp(len)
Commonly used tls_netif_get_ethif() involved	

api (where api's For specific definitions, please refer to Relevant header file comments)	socket() bind() closesocket() setsockopt() recvfrom() sendto();
Commonly used functions involved energy block	`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)));`
Example test steps	1. Open the macro definitions DEMO_UDP and DEMO_CONNECT_NET; 2. After compiling and upgrading successfully, you can see the corresponding command in the console information printed by uart0 make; 3. Send t-connect("TEST_N40_6","1234567890") through uart0 or t-oneshot allows modules to be screened; 4. Send t-udp(2,5100,224.1.2.1) through uart0 and uart0 will print; udp demo,cast:2,port:5100 localip : 192.168.1.104 local port :3000 setmuticast 5. Open the multicast tool on the PC on the same network as the module, and add the address in the receiving test (The multicast address is 224.1.2.1, the port is 5100), select the address, and click the Receive button. button; 6. Send t-sndudp(1024) through uart0, the multicast tool shows no packet loss;

	7. Open the debugging assistant on the PC and set the target multicast address 224.1.2.1 target port 3000, send data, after the module receives the data, uart0 prints the address and data length.

5.5 **DEMO_NTP** operation steps

Note: There are three examples under this DEMO.

5.5.1 t-ntp

Function description	This example implements the process of using ntp to obtain the current time;
command format	t-ntp

Commonly used APIs involved (where API
For the specific definition, please refer to the relevant header
file notes)
	tls_ntp_client();
	localtime();
	tls_set_rtc();

Commonly used function blocks involved	`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);`
	`  }`
	`}`

Example test steps	1. Open the macro definitions DEMO_NTP and DEMO_CONNECT_NET; 2. After compiling and upgrading successfully, the console information printed by uart0 can be displayed. see the corresponding command; 3. Send via uart0

	t-connect("TEST_N40_6","1234567890") or t-oneshot allows the module to be networked (with an external network); 4. Send t-ntp through uart0, uart0 will print the current time.

5.5.2 t-setntps

function description stated	This example implements the process of modifying the default ntp server through commands;
command case Mode	t-setntps("ntp_server_name1","ntp_server_name2","ntp_server_name3")
related to commonly used api (its middle api Specific Interpretation please reference phase critical essay file notes)	tls_ntp_set_server()
related to commonly used function block	none

Example test test steps	1. Open the macro definitions DEMO_NTP and DEMO_CONNECT_NET; 2. After compiling and upgrading successfully, you can see the corresponding command in the console information printed by uart0; 3. Send t-setntps through uart0("120.25.108.11", "ntp.sjtu.edu.cn", "us.pool.ntp.org") manually set the ntp server; 4. After resetting the module, send t-queryntps through uart0 to return [CMD]t-queryntps"120.25.108.11","ntp.sjtu.edu.cn","us.pool.ntp. org" 5. Send t-connect("TEST_N40_6","1234567890") through uart0 or t-oneshot allows the module to be networked (with an external network); 6. Send t-ntp through uart0, uart0 will print the current time.

5.5.3 t-queryntps

Function description	This example implements the command to query the name of the currently used ntp server Procedure;
command format	t-queryntps
Commonly used APIs involved (among them For the specific definition of api, please refer to the key file comment)	tls_ntp_query_sntpcfg()
Commonly used function blocks involved	
Example test steps	1. Open the macro definitions DEMO_NTP and DEMO_CONNECT_NET; 2. After compiling and upgrading successfully, you can see the console information printed by uart0 to the corresponding command;

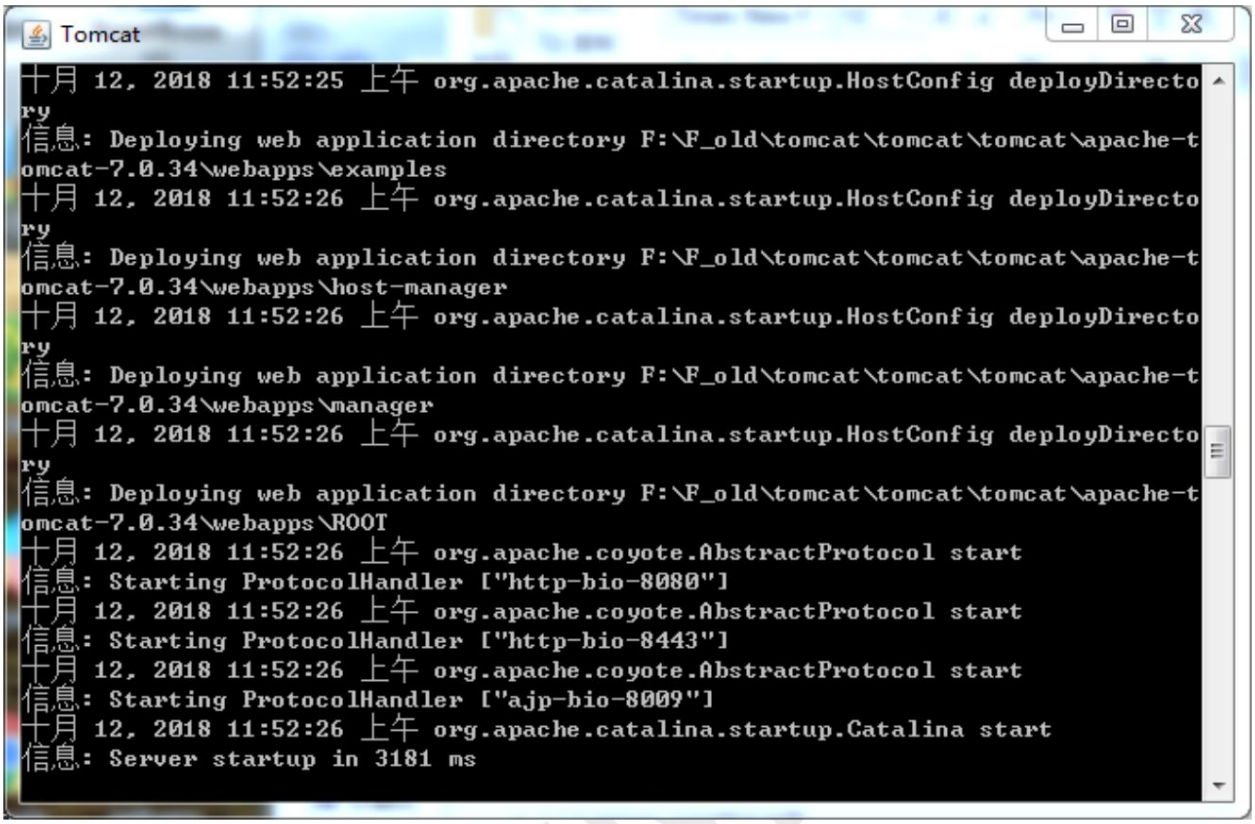
	3. Send via uart0 t-connect("TEST_N40_6","1234567890") or t-oneshot let Module network (with external network); 4. Send t-queryntps through uart0, uart0 will print the currently used The address of the ntp server.

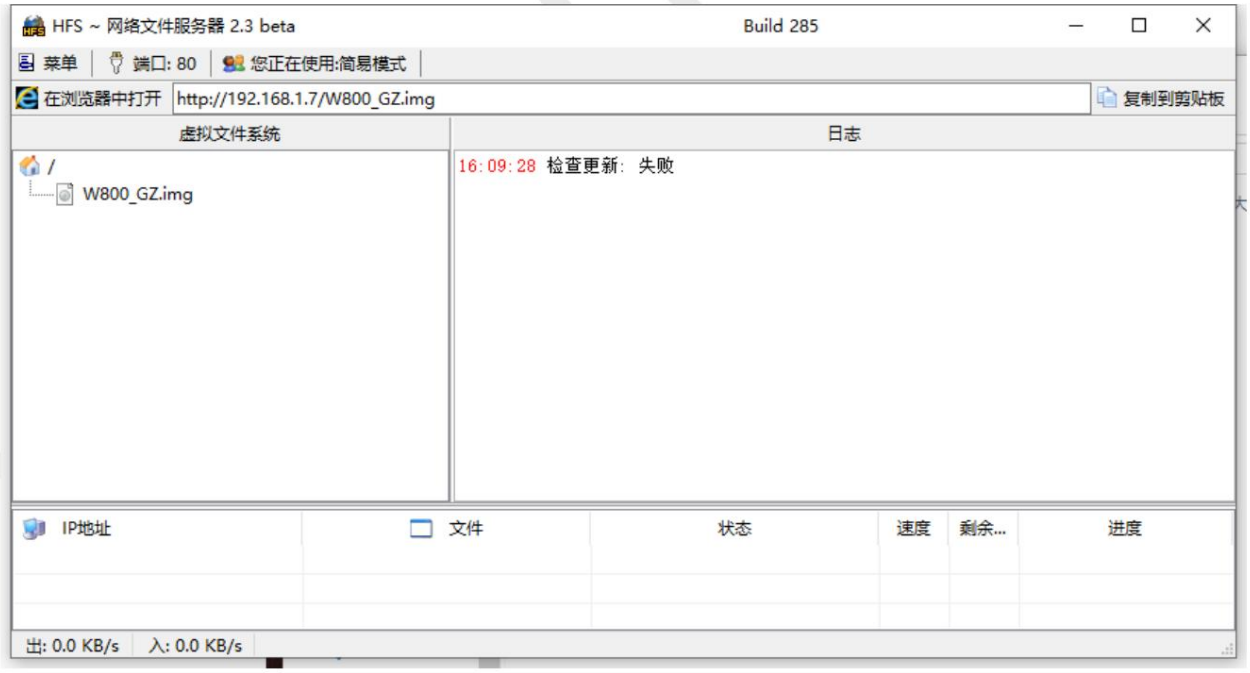
5.6 **DEMO_HTTP** operation steps

Note: There are four examples under this DEMO, you need to download the tomcat server (you need to place the required script files) and hfs server . The download address of related accessories is on the official website http://www.winnermicro.com/html/1/156/158/497.html under the Software Information tab"

The tool code used by the wmsdk demo: ".

The following figures are the page after the tomcat server is started and the page after the http server is ready to add firmware:





Among them, hfs server and tomcat server can be downloaded from the Internet, hfs can be used directly after downloading, tomcat (tested

After the 7.0.34 and 8.5.23 versions) are downloaded from the server, you need to modify and add some script files in it. Specifically, the

Replace the TestWeb folder under the webapps folder in the tomcat root directory with the official TestWeb file.

folder (in which the corresponding script files needed to test httpget httpput httppost have been added).

5.6.1 t-httpget

Function description	This example implements the process of get data in http format data communication;
command format	t-httpget=(http://xxx.xxx.xxx.xxx:8080/filepath/)
Commonly used api (where api's For specific definitions, please refer to Relevant header file comments)	HTTPClientOpenRequest() HTTPClientSetVerb() HTTPClientSendRequest() HTTPClientRecvResponse() HTTPClientReadData() HTTPClientCloseRequest()

Commonly used functions involved
energy block
	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;
	}

Example test steps	1. Open the macro definitions DEMO_HTTP and DEMO_CONNECT_NET; 2. After compiling and upgrading successfully, you can see the corresponding command in the console information printed by uart0 make; 3. Send t-connect("TEST_N40_6","1234567890") through uart0 or

t-oneshot allows modules to be screened;

4. Open tomcat on the PC on the same network as the module (ip is 192.168.1.100)

server and place files;

5. Send via uart0

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

return

[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

<table>
<tr><td>Function description</td><td>This example implements the process of putting data in http format data communication;</td></tr>
<tr><td>command format</td><td>t-httpput=("put_data")</td></tr>
<tr><td>Commonly used APIs involved (among them
For the specific definition of api, please refer to the
key file comment)</td><td>HTTPClientOpenRequest()
HTTPClientSetVerb()
HTTPClientSendRequest()
HTTPClientRecvResponse()
HTTPClientReadData()
HTTPClientCloseRequest()</td></tr>
<tr><td>Commonly used function blocks involved</td><td><pre>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;
}</pre></td></tr>
<tr><td>Example test steps</td><td>1. Open the macro definitions DEMO_HTTP and DEMO_CONNECT_NET;
2. After compiling and upgrading successfully, you can see the console information printed by uart0
to the corresponding command;
3. Send via uart0
t-connect("TEST_N40_6","1234567890") or
t-oneshot allows modules to be screened;</td></tr>
</table>

	4. Open the PC on the same network as the module (ip is 192.168.1.100) tomcat server and place files; 5. Send t-httpput=(user=winnermicroput) through uart0, uart0 returns 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

Function description	This example implements the process of post data in http format data communication;
command format	t-httppost=("post_data")
Commonly used APIs involved (which Please refer to the specific definition of api in Refer to the relevant header file comments)	HTTPClientOpenRequest() HTTPClientSetVerb() HTTPClientSendRequest() HTTPClientRecvResponse() HTTPClientReadData() HTTPClientCloseRequest()

Commonly used function blocks involved	`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;`
	`}`

Example test steps	1. Open the macro definitions DEMO_HTTP and DEMO_CONNECT_NET; 2. After compiling and upgrading successfully, you can see in the console information printed by uart0 corresponding command; 3. Send via uart0

t-connect("TEST_N40_6","1234567890") or t-oneshot

Let the module add the net;

4. Open the PC on the same network as the module (ip is 192.168.1.100)

tomcat server and place files;

5. Send t-httppost=(user=winnermicropost) through uart0,

uart0 returns

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

Function description	This example implements the firmware upgrade function of the device through ota.
Command format	t-httpfwup=(http://192.168.1.100:80/w800_ota.img) The IP address in the above command is the IP address of the ota server, followed by the corresponding port number;
Commonly used api (where api's For specific definitions, please refer to Exam related header files note)	t_http_fwup()
Commonly used function block	none
Example test steps	1. Open the macro definitions DEMO_HTTP and DEMO_CONNECT_NET; 2. After compiling and upgrading successfully, you can see the corresponding command in the console information printed by uart0 make; 3. Send t-connect("TEST_N40_6","1234567890") through uart0 or t-oneshot allows modules to be screened; 4. Open the hfs server on the PC on the same network as the module (ip is 192.168.1.100),

	port 8080 and place the firmware named WM_W800_SEC.img; 5. Send via uart0 t-httpfwup=(http://192.168.1.100:80/w800_ota.img), uart0 Prints the upgrade progress, and resets after the module is upgraded successfully. Upgrade compressed img.

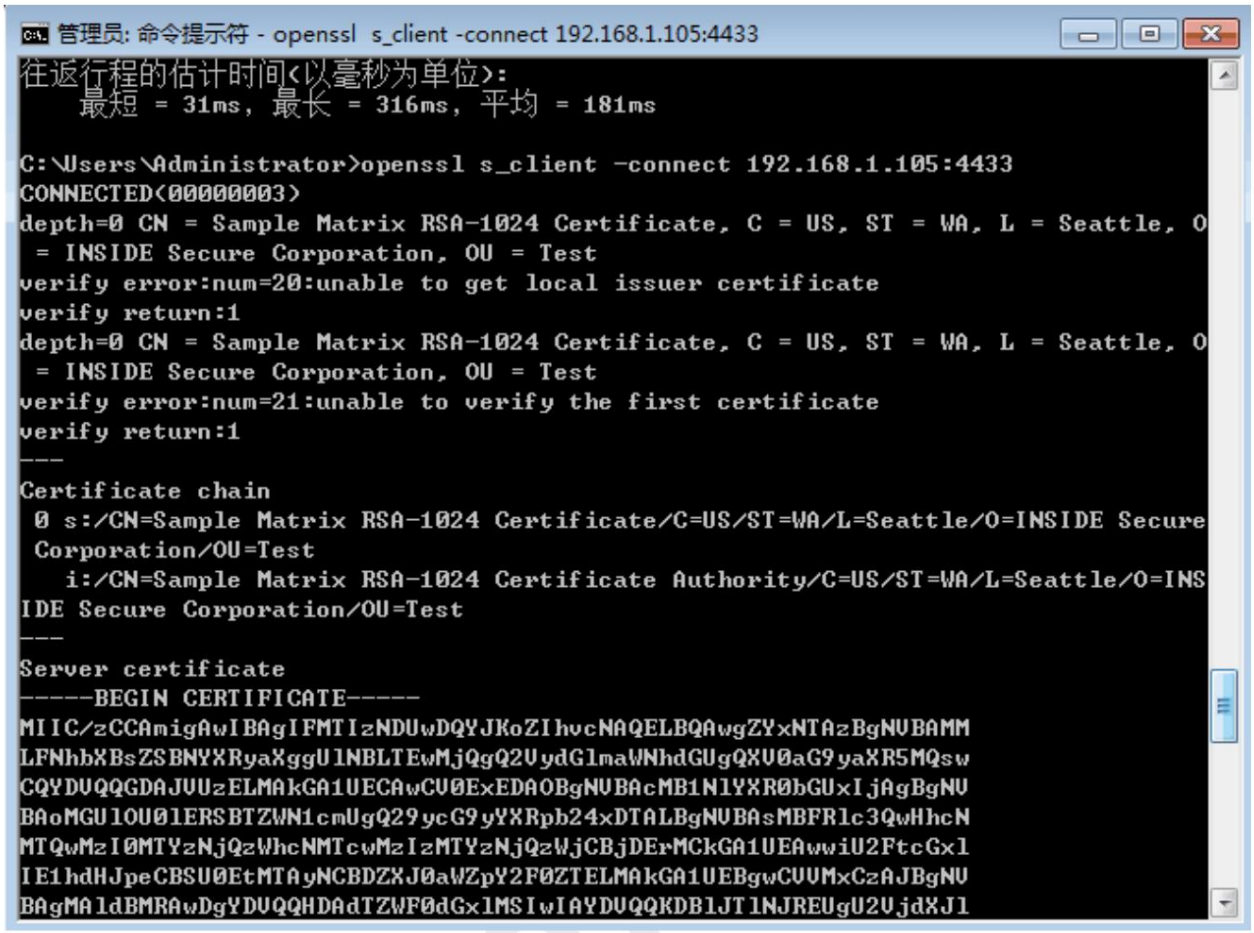
5.7 **DEMO_SSL_SERVER** operation steps

Function description	This example implements ssl server; allows other clients to establish tls connection with the device side; Note: TLS_CONFIG_SERVER_SIDE_SSL needs to be turned on, when demonstrating other DEMOs This macro switch needs to be turned off. The test needs to download openssl or other can connect to ssl server Tool of
command format	t-ssl-server
Commonly used api (where api's tool For body definition, please refer to key file comment)	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()
Commonly used functions involved energy block	none
Example test steps	The specific demo test steps are as follows: 1. Open the macro definitions DEMO_SSL_SERVER and DEMO_CONNECT_NET; 2. After compiling and upgrading successfully, you can see the corresponding information in the console information printed by uart0

	Order; 3. Send t-connect("TEST_N40_6","1234567890") through uart0 or t-oneshot allows the module to be networked (ip is 192.168.1.104); 4. Send t-ssl-server through uart0, uart0 returns [CMD]t-ssl-server ssl server task Listening on port 4433 5. Open openssl on the PC on the same network as the module, and execute the command s_client –connect 192.168.1.104:4433, where the ip address and port number are set to The IP address of the device and the corresponding open port number. 6. At this time, the uart0 of the module is printed accept fd 1 tls_mem_alloc cp 2001ef88 tls_ssl_server_handshake rc 0 cp->time.tv_sec 0

The following figure shows the command line page message after successfully connecting to the ssl server using the openssl tool (which needs to be installed by the user)

interest.



5.8 **DEMO_WEBSOCKETS** operation steps

Note: There are two examples under this DEMO, you need to download the WEBSOCKET_SERVER test server.

5.8.1 Websocket data communication without encryption

Function description	This example implements the use of websocket to establish a connection with the websocket server. The process of encrypting the connection and sending and receiving data;
command format	t-websockets
Commonly used APIs involved (among them For the specific definition of api, please refer to Relevant header file comments)	lws_create_context() lws_client_connect_via_info() lws_callback_on_writable()

<table>
<tr><td></td><td>lws_service()

lws_context_destroy()

lws_write()</td></tr>
<tr><td>Commonly used function blocks involved</td><td>

```
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;
}
```

</td></tr>
<tr><td>Example test steps</td><td>1. Open the macro definitions DEMO_WEBSOCKETS and

DEMO_CONNECT_NET, turn off LWS_USE_SSL;

2. After compiling and upgrading successfully, you can see the console information printed by uart0

to the corresponding command;

3. Send via uart0

t-connect("TEST_N40_6","1234567890") or

t-oneshot allows modules to be screened;

4. If using the WEBSOCKET_SERVER test server,

Run the command line on a PC on the same network (ip is 192.168.1.100)

websocketd --port=8080 echo_client.bat;

5. Send t-websockets through uart0, uart0 returns

[CMD]t-websocketsCLIENT_ESTABLISHED

send {"msg_type":"keepalive"} 2

recv:websocket server send</td></tr>
</table>

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

5.8.2 Data communication by websocket encryption

Function description	This example implements the use of websocket to establish a connection with the websocket server. The process of densely connecting and sending and receiving data;
command format	t-websockets
Commonly used APIs involved (among them For the specific definition of api, please refer to Relevant header file comments)	lws_create_context() lws_client_connect_via_info() lws_callback_on_writable() lws_service() lws_context_destroy() lws_write()

Commonly used function blocks involved	`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;`
	`}`

Example test steps	1. Open the macro definition DEMO_WEBSOCKETS, DEMO_CONNECT_NET, LWS_USE_SSL, if used WEBSOCKET_SERVER test server, press Notice step modification code in wm_websockets_demo.c (positive

	There is no need to pay attention to step 3 in the Notice when testing the regulatory server; 2. After compiling and upgrading successfully, you can see the console information printed by uart0 to the corresponding command; 3. Send via uart0 t-connect("TEST_N40_6","1234567890") or t-oneshot allows modules to be screened; 4. If using the WEBSOCKET_SERVER test server, Run the command line on a PC on the same network (ip is 192.168.1.100) websocketd --port=8080 --ssl --sslcert="certificate.pem" --sslkey="key.pem" echo_client.bat; 5. Send t-websockets through uart0, uart0 returns [CMD]t-websocketsCLIENT_ESTABLISHED send {"msg_type":"keepalive"} 1 recv:websocket server send recv:{"msg_type":"keepalive"} 1

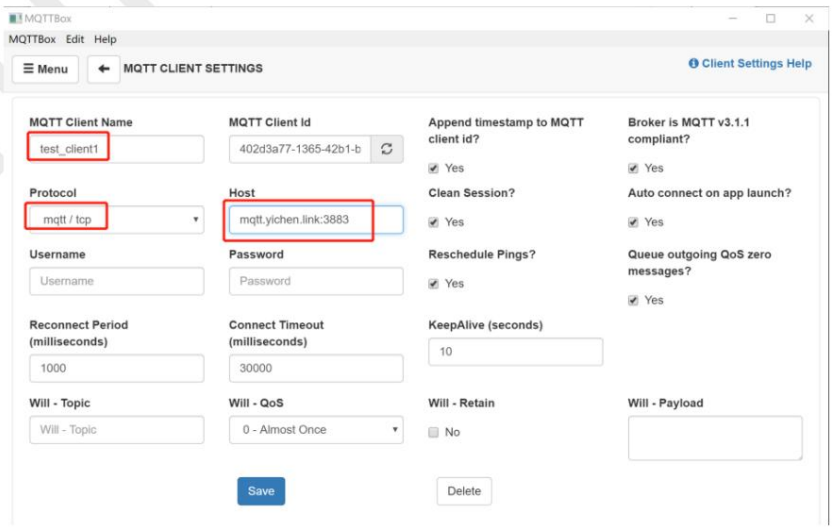
5.9 **DEMO_HTTPS** operation steps

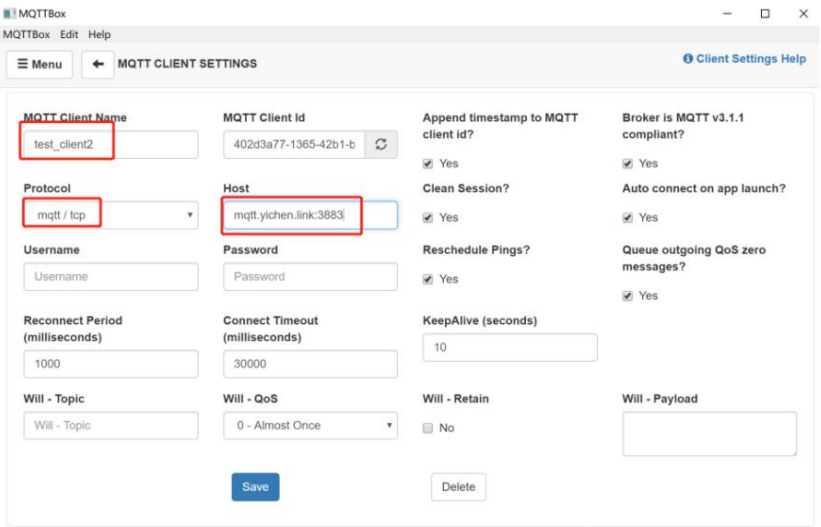
Function description	This example implements the process of obtaining web page data through https;
command format	t-https
Commonly used APIs involved (its Gethostbyname()	

Please refer to the specific definition of api in Refer to the relevant header file comments)	HTTPWrapperSSLConnect() HTTPWrapperSSLSend() HTTPWrapperSSLRecv() HTTPWrapperSSLClose()
Commonly used function blocks involved	
Example test steps	1. Open the macro definitions DEMO_HTTPS, DEMO_CONNECT_NET, TLS_CONFIG_HTTP_CLIENT and TLS_CONFIG_HTTP_CLIENT_SECURE; 2. After compiling and upgrading successfully, you can see in the console information printed by uart0 corresponding command; 3. Send via uart0 t-connect("TEST_N40_6","1234567890") or t-oneshot Let the module add the network (with external network); 4. Send t-https through uart0, uart0 will print out https://www.tencent.com/legal/html/en-us/index. The content of html (note that there is printing information in the demo).

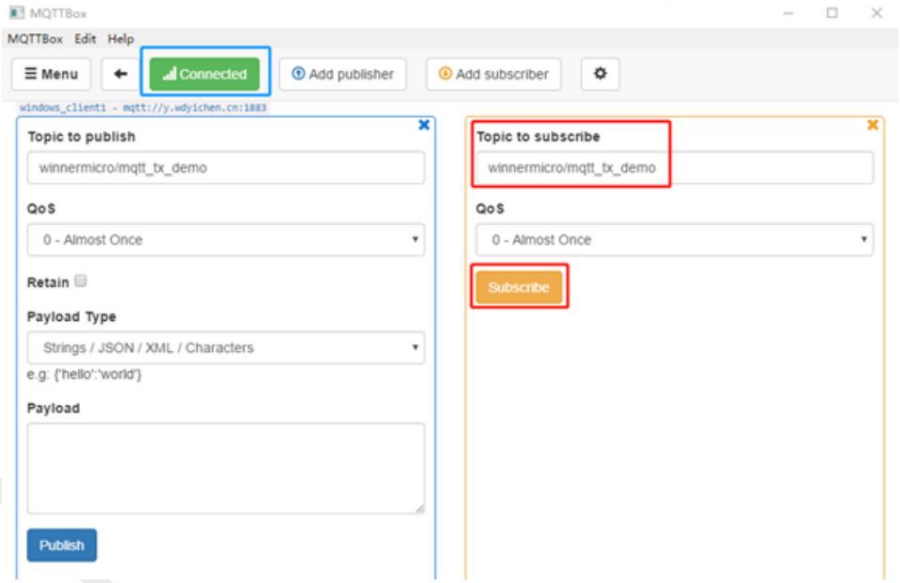
5.10 DEMO_MQTT operation steps

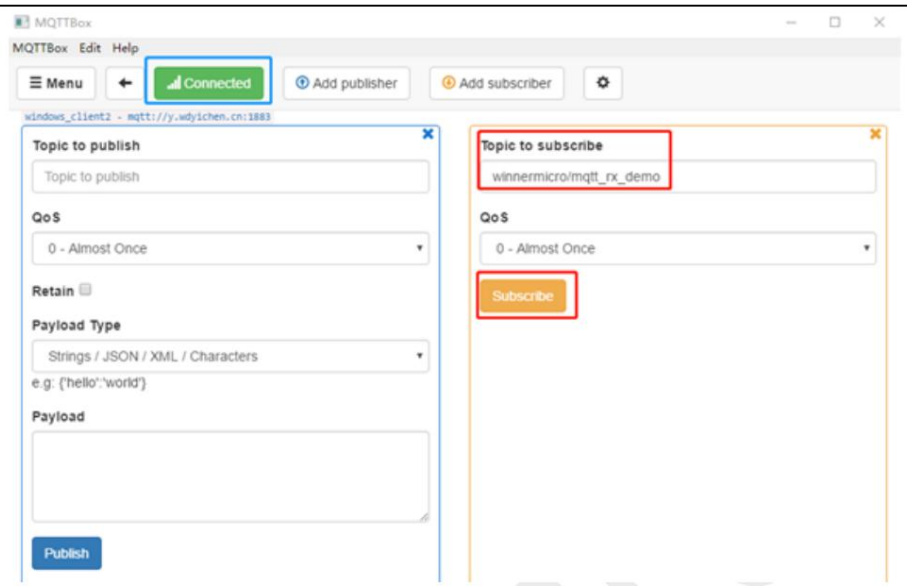
Function description This example implements the process of establishing a connection with the server and communicating with the server in the way of the use case mqtt;	
Command format t-mqtt	
Commonly used api (where api	mqtt_init() mqtt_connect()

please explain Reference related headers file notes)	MQTTParseMessageType() mqtt_subscribe() mqtt_publish() mqtt_parse_msg_id() mqtt_ping()
Commonly used function block	none
Example test steps	1. Open the macro definitions DEMO_MQTT and DEMO_CONNECT_NET; 2. After compiling and upgrading successfully, you can see the corresponding command in the console information printed by uart0; 3. Send t-connect("TEST_N40_6","1234567890") through uart0 or t-oneshot allows the module to be networked (with an external network); 4. Send t-mqtt through uart0, uart0 will print out and "mqtt.yichen.link:3883" establishes the mqtt connection. 5. Download the MQTTBox software, open two MQTTBox windows, and set them as follows: 



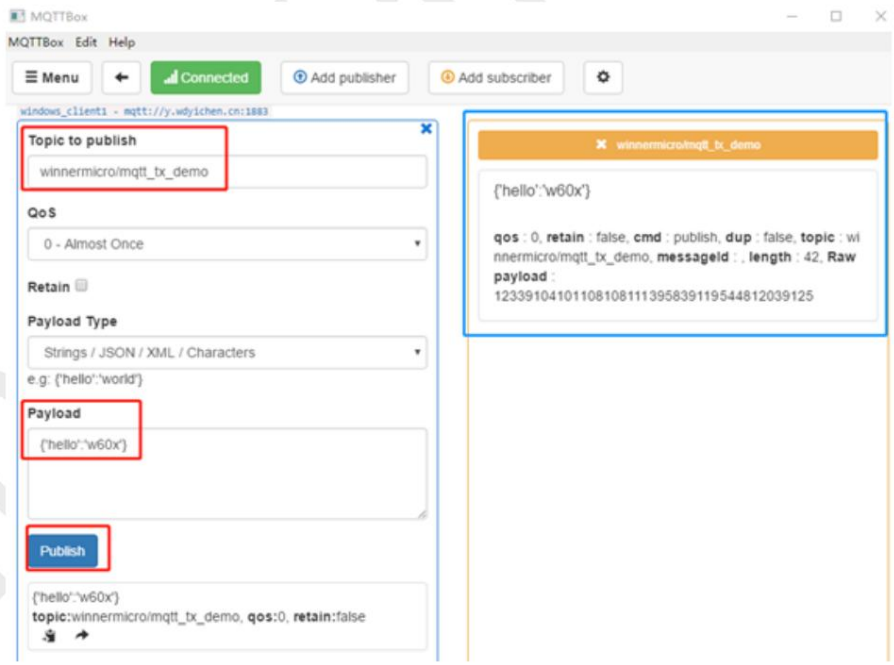
6. Click the save button to establish a connection with the server, and set the client subscription as follows:

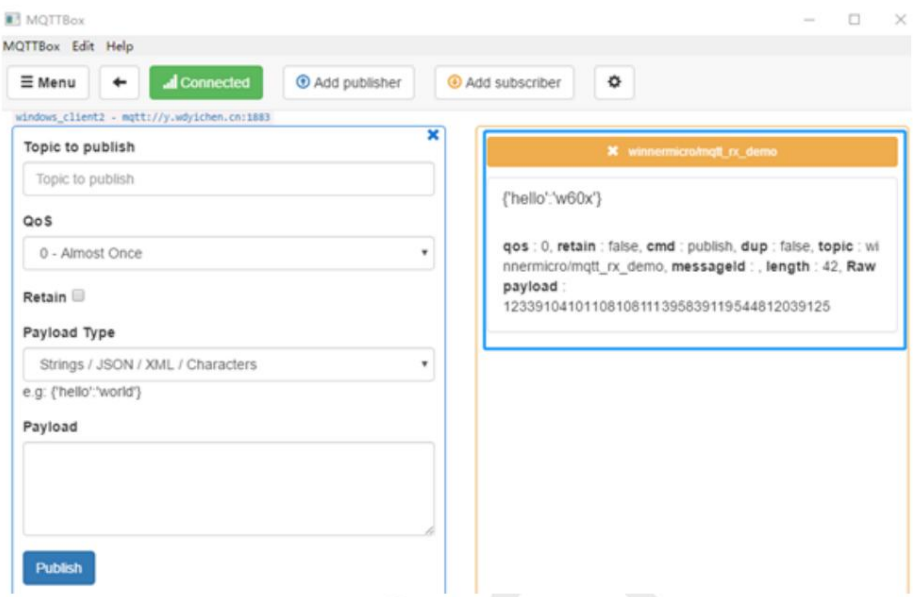




7. Push a message "{'hello':'w60x'}" on the windows_client1 client,

uart0 prints the message, and the windows_client2 client also receives it.



5.11 DEMO_DSP operation steps

Functional Description This example implements the processing example of the DSP	
Command format t-dsp(x), the value of x is 0,1,2,3,4	
Commonly used api (where api please explain Reference related headers file notes)	csky_fir_init_q15 csky_fir_q15 csky_mat_init_q31 csky_mat_mult_q31 csky_rfft_q15 csky_sin_q31 csky_var_q15
Commonly used function block	none
Example test steps	1. Open the macro definition DEMO_DSP;

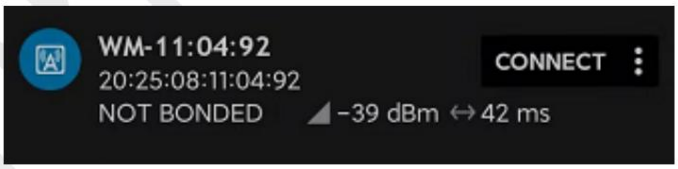
	2. After compiling and upgrading successfully, you can see the corresponding command in the console information printed by uart0; 3. Send t-dsp(0) through uart0, uart0 prints: dsp fir run success! 4. Send t-dsp(1) through uart0, uart0 prints: dsp matrix cal run success! 5. Send t-dsp(2) through uart0, uart0 prints: dsp rfft run success! 6. Send t-dsp(3) through uart0, uart0 prints: dsp sin run success! 7. Send t-dsp(4) through uart0, uart0 prints: dsp variance run success!

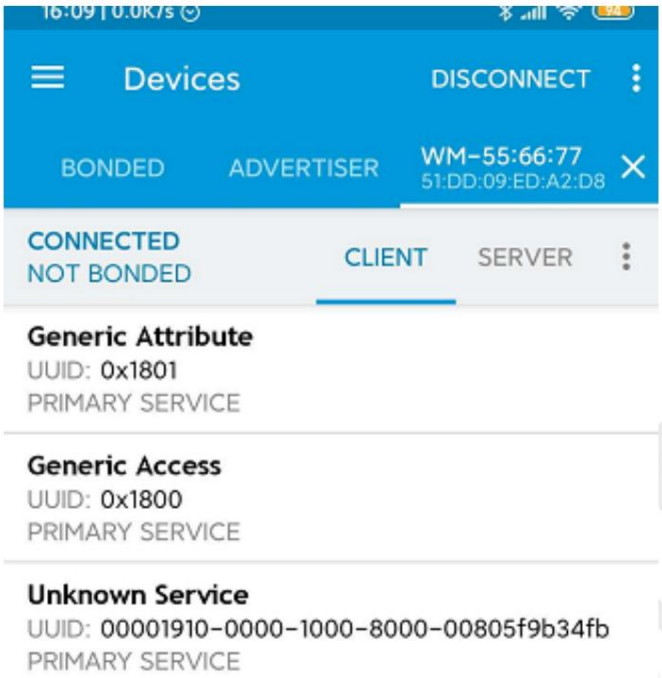
5.12 DEMO_BT operation steps

Note: There are four examples under this DEMO.

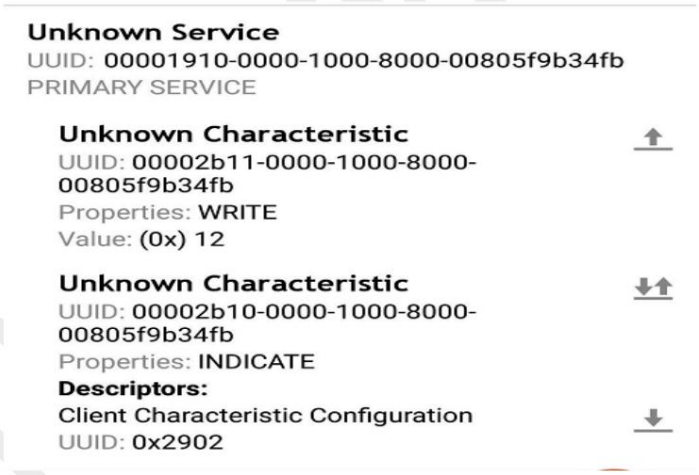
5.12.1 Ble server example

Function description	This example implements the processing example of W800 as Ble server, this DEMO requires mobile phone to install nRF Connect (download it from the app store)
Command format	t-bt-on t-bt-off t-ble-server-on t-ble-server-off
Commonly used api (where api please explain Reference related headers file notes)	tls_open_peripheral_clock tls_bt_enable tls_bt_disable tls_close_peripheral_clock

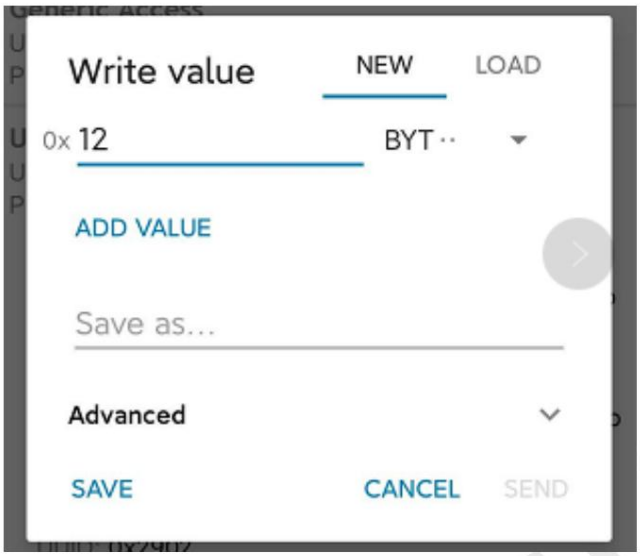
Commonly used function block	none
Example test steps	1. Open the macro definition DEMO_BT (make sure to use the lib of the default ble when the SDK is released, make sure Identify the macro definition TLS_CONFIG_BLE and close the macro definition in wm_config.h TLS_CONFIG_BR_EDR); 2. After compiling and upgrading successfully, you can see the corresponding command in the console information printed by uart0; 3. Send t-bt-on through uart0, uart0 prints init base application related information; 4. Send t-ble-server-on through uart0, uart0 prints after success [WM_I] <0:00:07.188> ### wm_ble_server_api_demo_init success 5. Turn on Bluetooth on the phone and scan to the device using nRF connect (the default name is WM-XX:XX:XX, the last six digits of the module btmac);  6. App connects to the device (Note: If the app disconnects actively, this DEMO needs to Set t-ble-server-off and t-ble-server-on to restart to connect normally);



7. App click to view service;



8. App click the up arrow to write the characteristic value;



After clicking SEND, uart0 prints the data sent by the app: ###write cb12;

9. The app clicks the down arrow to read the descriptor, and the app displays the "Hello" sent by the device;

10. The App clicks the up and down arrows to enable the Indication;

11. App click the up and down arrows again to close the Indications;

12. Send t-ble-server-off through uart0 to turn off the demo server function;

13. Send t-bt-off through uart0, uart0 prints bt system cleanup host

Related Information.

5.12.2 Ble client example

Function description	This example implements the processing example of W800 as Ble client, this DEMO needs to use two developers board, development board A is the Ble server, and development board B is the Ble client.
Command format	t-bt-on t-bt-off t-ble-server-on t-ble-server-off t-ble-client-on t-ble-client-off
Commonly used api (where api please explain Reference related headers file notes)	tls_open_peripheral_clock tls_bt_enable tls_bt_disable tls_close_peripheral_clock
Commonly used function block	none
Example test steps	1. Open the macro definition DEMO_BT (make sure to use the lib of the default ble when the SDK is released, make sure Identify the macro definition TLS_CONFIG_BLE and close the macro definition in wm_config.h TLS_CONFIG_BR_EDR); 2. Compile, upgrade the firmware of the two development boards, after the upgrade is successful, the control of printing in uart0 The corresponding command can be seen in the station information;

	3. Development board A sends t-bt-on through uart0, and uart0 prints init base application related information; Then send t-ble-server-on through uart0; 4. Development board B sends t-bt-on through uart0, and uart0 prints init base application related information; Then send t-ble-client-on through uart0; 5. At this time, B will scan, connect, and enable A's Indication function. A will keep Send data to B through Indication. B Print statistics on uart0 at intervals result.

5.12.3 Ble Broadcast Example

Function description	This example implements the processing example of W800 as Ble server, this DEMO requires mobile phone to install nRF Connect (download it from the app store)
Command format	t-bt-on t-bt-off t-ble-adv=(type) type is defined as: 1 can connect to broadcast; 2 can not connect to broadcast; 0 stop broadcast
Commonly used api (where api please explain Reference related headers file notes)	tls_open_peripheral_clock tls_bt_enable tls_bt_disable tls_close_peripheral_clock

Commonly used function block	none
Example test steps	1. Open the macro definition DEMO_BT (make sure to use the lib of the default ble when the SDK is released, make sure Identify the macro definition TLS_CONFIG_BLE and close the macro definition in wm_config.h TLS_CONFIG_BR_EDR); 2. After compiling and upgrading successfully, you can see the corresponding command in the console information printed by uart0; 3. Send t-bt-on through uart0, and uart0 prints the init base application related information; 4. Send t-ble-adv=(1) through uart0, the mobile phone can scan the bluetooth device, and can connect successfully; 5. Send t-ble-adv=(2) through uart0, the mobile phone can scan the bluetooth device, and can not connect; 6. Send t-ble-adv=(0) through uart0, the mobile phone cannot scan the bluetooth device.

5.12.4 Ble scan example

Function description	This example implements the processing example of W800 as Ble server, this DEMO requires mobile phone to install nRF Connect (download it from the app store)
Command format	t-bt-on t-bt-off t-ble-scan=(type) type is defined as: 1 to start scanning; 0 to close scanning
Commonly used api (where api	tls_open_peripheral_clock tls_bt_enable

please explain Reference related headers file notes)	tls_bt_disable tls_close_peripheral_clock
Commonly used function block	none
Example test steps	1. Open the macro definition DEMO_BT (make sure to use the lib of the default ble when the SDK is released, make sure Identify the macro definition TLS_CONFIG_BLE and close the macro definition in wm_config.h TLS_CONFIG_BR_EDR); 2. After compiling and upgrading successfully, you can see the corresponding command in the console information printed by uart0; 3. Send t-bt-on through uart0, and uart0 prints the init base application related information 4. Send t-ble-scan=(1) through uart0, uart0 prints the scan result 5. Send t-ble-scan=(0) through uart0, uart0 stops printing

5.13 DEMO_FATFS operation steps

Function description	This example demonstrates how to use the device to use the file system on the sd card. Note: If the sd card is easy to be too large, there may be several attempts to format it into phenomenon of work. This does not affect normal reading and writing, and can be adjusted according to actual needs. How much space to use to create a file system, you can modify the function disk_ioctl() Set the value of SDCardInfo.CardCapacity in.
command format	t-fatfs
Commonly used apis involved (the api wm_sdio_host_config()	

For the specific definition, please refer to the relevant header file notes)	f_mkfs() f_mount() f_open() f_write() f_read() f_close()
Commonly used function blocks involved	none
Example test steps	1. Open the macro definition DEMO_FATFS; 2. After the compilation and upgrade are successful, the console information printed by uart0 can be displayed. see the corresponding command; 3. Connect the sd card on the development board. The IO port used in this example is PB06-PB11; 4. Send t-fatfs through uart0; 5. After the device receives the uart0 command, it will format the SD card first; After the format is successful, mount the file system; After the mount is successful, create a new file and write data to it; After the writing is successful, print the written data in uart0, and then read it from the file data; After successful reading, print the read data in uart0.

5.14 DEMO_MBEDTLS operation steps

Function description	This example implements the process of obtaining web page data through https;
command format	t-mbedtls

Commonly used APIs involved (which Please refer to the specific definition of api in Refer to the relevant header file comments)	
Commonly used function blocks involved	
Example test steps	1. Open the macro definitions DEMO_CONNECT_NET and DEMO_MBEDTLS; 2. After compiling and upgrading successfully, you can see in the console information printed by uart0 corresponding command; 3. Send via uart0 t-connect("TEST_N40_6","1234567890") or t-oneshot Let the module add the network (with external network); 4. Send t-mbedtls through uart0, uart0 will print out https://www.tencent.com/legal/html/en-us/index. html content (note that there is printing information in the demo).