All examples must run with Bootloader.
You can find Bootloader bin in \Middleware\Bootloader\

MCU peripheral examples	Example purpose	Example running flow
Comparator	Compare input voltage by Comparator, and trigger Comparator interrupt.	This example set Comparator channel, voltage, and polarity. When the comparison condition is matched, trigger the interrupt.
Crypto	Demonstrate the use of crypto API for 1. AES Encryption/AES Decryption 2. ECC point multiply for SECP192R1 3. ECC point multiply for SECP256R1 4. ECC point multiply for SECT163R2 5. ECC point multiply for Curve 25519	This example call crypto API to do related secure operations. When the operation is completed, compare the result with expected data. If the result matched expected data, these functions will output success message. User should see all success message without error. Example of Curve25519 (function curve_c25519_test2, it is RFC7748 test vector) will take more than 2 minutes, if you don't
Crypto_CCM	Use crypto API for AES CCM, including CCM encryption and decryption verification.	This example calculate some test vector examples of RFC3610 to demonstrate API usage. This program compares the result data with expected data, and output message to show correct or not.
Crypto_HMAC	Use crypto API for SHA256 HMAC.	This example calculate some test vectors example of RFC4231 to demonstrate API usage. This program compares the result data with expected data, and output message to show correct or not.
DMA_Linklist	Use DMA API for scattered data move.	This example move scatter data of source memory to continuous destination memory. This link list only for data movement between memory to memory. Program will check the execution result. If error, you will not see the ending message "DM for memory link list test success".
DMA_Memory	Use DMA API for memory copy. Use both interrupt mode and polling mode.	This example move data from source memory to destination memory. The memory copy operation is memory to memory. It can not be used for hardware peripheral. Program will check the memory execution result. If error occur, you will not see the ending message "DMA for memory to memory success".
Flash	Use some basic flash API to access data in flash. Including check flash size, get flash unique ID, erase sector (4KB), write page and read page back for verification.	This example read some information from flash, including unique id and flash device info, then erase 4K sector in (0xEF000), and write special data to page (0xEF000~0xEF0FF, 256 bytes per page), then read back the page and verify. You should see the message "Page verify success".
FreeRTOS	FreeRTOS porting and basic operation.	This example create FreeRTOS task, queue, SW timer, and start the task scheduler. The program also implement idle task sleep and SW timer wakeup mechanism.
GPIO	Use GPIO API to output signal or configure input pin as interrupt source. This example also illustrates de-bounce gpio function for input pins.	1. After reboot, all pins are set to output mode. You can see the signal all output LOW. 2. Press key in PC console, GPIO0 will output HIGH, and when pressing any key again, GPIO0 output LOW. 3. Next same operation will let GPIO1 output HIGH and then LOW. 4. The GPIO will output in sequence order (0->1->2...->31). Please notice: GPIO16 and GPIO17 is used for UART, not GPIO mode. 5. After the last GPIO31 action completed, program will show "GPIO_test_output ok". 6. Set all pins (except UART pins and GPIO31) to input mode, and enable de-bounce function. Connect the GPIO[N] to ground, you should see GPIO31 output HIGH/LOW toggles. The toggle is implemented in GPIO ISR, which means GPIO[N] input ISR is

I2C	Use I2C API to illustrate I2C read/write operations.	This example need an "ATEML" I2C EEPROM (AT24CXXX) slave to test the program. At first, I2C master try to send command to a non-exist slave (I2C address 0x47), so the API will return ERR_NOACK. Then I2C master try to send read/write commands to ATEML EEPROM (I2C address 0x50), the write/read operations should be all successful. User should see "I2C read write OK" message
MP_Sector	MP sector stores user information and calibration data. This example define MP sector default value and generate bin file.	According to MP sector structure, this example modify MP sector constant table default value. Download the MP sector bin to last 64KB of Flash. We can also use ISP tool or MP Tool to download MP sector bin.
PWM	Demonstrate PWM API usage. PWM can wire with LED, motor, etc.	This example configure PWM0~PWM4 sequence controller, clock rate, counter trigger, counter mode, play mode. Generate various kinds of PWM signal output.
QSPI	Use QSPI API to illustrate QSPI operation	This example need an external GD QSPI flash for demonstration. The demonstration flow is: 1. Erase the first 4KB sector. 2. Program test pattern to different flash page by using different QSPI mode (1-bit or 4-bit mode), by polling mode or DMA mode. 3. Read page data back by using [1-bit, 2-bits, 4-bits mode], [polling mode or DMA mode]. 4. If everything is correct, user will see "QSPI verify ok" message. (Notice: there is no external QSPI flash in RT58x-EVK. So if no QSPI flash connect with EVK QSPI, this program will hang in and wait for finish polling state.)
RTC	Use RTC API to illustrate RTC operation	This example generate interrupt when RTC alarm_mode matched.
RTC_DeepSleep	Using RTC to wakeup CPU in deep sleep	Demonstration step: 1. For first time reboot, this example set RTC time 21-07-22 11:40:00. Then CPU enter deep sleep mode automatically. 2. The program wakeup every minute by RTC event interrupt. 3. By connecting GPIO20 to ground. User can also wakeup the CPU. In this example, RTC counter keeps counting during deep sleep period. After CPU wakeup, it will print current RTC time. After setting RTC alarm mode (because Cortex-m3 reset), it will go to
RTC_Sleep_Wakeup	Using RTC to wakeup CPU in sleep	For first time reboot, this example set RTC time 21-07-22 11:40:00. The program shows message in console. User can press "0", "1", "2" to enter different level sleep mode. CPU will wakeup at every minute (mm:00). After CPU wakeup, it outputs message of RTC wakeup time, then enter sleep mode
Sleep	Demonstrate how to enter low power mode and wakeup by GPIO.	Enter low power mode (level 0~3) by pushing EVK key 0~3 (GPIO0~GPIO3). Push key4 (GPIO4) to wakeup. GPIO20 wires to LED, which shows execution status.
SPI_DMA_Loopback	Use SPI DMA API to illustrate SPI DMA mode	1. Connect GPIO6 to GPIO28, GPIO7 to GPIO29, GPIO8 to GPIO30, GPIO9 to GPIO31. 2. SPI0(use GPIO6~9 pins) is master and SPI1(use GPIO28~31 pins) is slave. They use SPI DMA function to transfer data. 3. In this example, master send data to slave, slave also send data to master. If send/receive data is all correct, user will see message "TEST SPI loopback OK!"
SPI_Master_PIO	Use SPI API to illustrate SPI Master PIO mode (non-DMA mode)	Use SPI0 to implement SPI non-DMA data transfer. SPI0 also supports multiple chip select, if you want to enable this option, please add #define SUPPORT_QSPI0_MULTI_CS 1 in project_config.h

SWI	Demonstrate SW interrupt operation. SW can actively trigger interrupt	This example enable SW interrupt and register SW interrupt callback function. When program actively trigger SW interrupt, SWI interrupts the running program and jump to corresponding ISR and callback function.
UART	Use UART API to illustrate UART transfer function.	Connect GPIO28(UART TX) to GPIO29(UART RX). After reboot, the program generate some test pattern and send from UART TX to UART RX. Program compares the data. If data is correct, user will see "TEST OK!"
UART_Break_Wakeup	Use UART API to illustrate UART break, which can wakeup CPU in sleep mode.	User can press "0", "1", "2" in console to enter different sleep mode. After that, user can use some terminal tool, like tera term, to send "UART break". When CPU receive the signal "UART break", CPU wakeup from sleep mode. CPU will enter sleep mode again after user press another key in console.
UART1_FlowCtrl	Use UART API to illustrate UART1 flow control.	In this example, CPU use GPIO20 as RTS, and GPIO21 as CTS. Please connect UART1 TX and UART1 RX pin in this example. If CTS is in HOGH (this is default mode), UART1 TX stops to transfer. If CTS is LOW, UART1 TX sends data. You can also call function uart_set_modem_status(1, state) to set RTS state. When state=1, it will allow other side to send data to RT58x. When state=0, the other side should stop to send data to RT58x. To support this option, please add #define SUPOORT_UART1_FLOWCNTL 1