



# **PY32T020 series**

## **32-bit ARM® Cortex®-M0+ microcontroller**

### **HAL Library Sample Manual**

# 1 ADC

## 1.1 ADC\_AnalogWatchdog

此样例演示了 ADC 的模拟看门狗功能，当开启看门狗的通道的电压值不在设定的上下限中，会进入看门狗中断。

This example demonstrates the analog watchdog function of ADC. When the voltage value of the channel that opens the watchdog is not within the set upper or lower limits, Will enter watchdog interrupt.

## 1.2 ADC\_MultichannelSwitch

此样例演示了 ADC 的多通道切换转换。

This example demonstrates the channel switching for an ADC.

## 1.3 ADC\_SingleConversion\_TriggerSW\_Polling

此样例演示了 ADC 的软件触发和轮询功能。

This example demonstrates the software triggering and polling functions of ADC.

## 1.4 ADC\_SingleConversion\_TriggerTimer\_IT

此样例演示了 ADC 的 TIM 触发和中断的功能。

This example demonstrates the TIM trigger and interrupt functions of ADC.

## 1.5 ADC\_TempSensor

此样例演示了 ADC 的 Tempsensor 的采样功能。

This sample demonstrates the sampling function of ADC's Tempsensor.

## 1.6 ADC\_VrefbufAndVrefint

此样例演示了 ADC 的 VREFINT 采样功能和 VREFBUF 的功能，通过 VREFINT 推算出 VREFBUF 的电压。

This example demonstrates the VREFINT sampling function and VREFBUF function of ADC, and calculates the voltage of VREFBUF through VREFINT.



## 2 COMP

### 2.1 COMP\_CompareGpioVs1\_2VCC\_IT

此样例演示了 COMP 比较器中断功能，PA02 作为比较器负端输入，1/2VCCA 作为正端输入，通过调整 PA02 上的输入电压，当检测到比较器输出状态为高时，LED 灯亮，比较器输出状态为低时，LED 灯灭。

This example demonstrates the COMP comparator interrupt function, with PA02 as the negative input of the comparator and 1/2VCCA as the positive input. By adjusting the input voltage on PA02, when the comparator output state is detected to be high, the LED light will turn on, and when the comparator output state is low, the LED light will turn off.

### 2.2 COMP\_CompareGpioVs1\_2VCC\_Polling

此样例演示了 COMP 比较器轮询功能，PA02 作为比较器负端输入，1/2VCCA 作为正端输入，通过调整 PA02 上的输入电压，当检测到比较器输出状态为高时，LED 灯亮，比较器输出状态为低时，LED 灯灭。

This example demonstrates the COMP comparator polling function, with PA02 as the negative input of the comparator and 1/2VCCA as the positive input. By adjusting the input voltage on PA02, the LED lights up when the comparator output state is detected to be high, and turns off when the comparator output state is low.

### 2.3 COMP\_CompareGpioVs1\_2VCC\_WakeUpFromStop

此样例演示了 COMP 比较器唤醒功能，PA0 作为比较器负端输入，32/64VCC 作为比较器正端输入，进入 stop 模式后，通过调整 PA0 上的输入电压，产生中断唤醒 stop 模式。

This example demonstrates the COMP comparator wake-up function, with PA0 as the negative input and 32/64VCC as the positive input of the comparator. After entering stop mode, the interrupt wake-up stop mode is generated by adjusting the input voltage on PA0.

### 2.4 COMP\_CompareGpioVs1\_2VCC\_Window

此样例演示了 COMP 比较器的 window 功能，比较器 1 的 Plus 端用比较器 2 的 IO4(1/2VCCA)作为输入，PB1 作为比较器 1 负端输入，当 PB1 的电压值大于 1.65V 时，LED 灯灭，小于 1.65V 时，LED 灯亮。PA2 作为比较器 2 负端输入，当 PA2 的电压值大于 1.65V 时，PA4 拉低，小于 1.65V 时，PA4 拉高

This example demonstrates the window function of the COMP comparator. The Plus end of comparator 1 uses the IO4 (1/2VCCA) of comparator 2 as the input, and PB1 as the negative end input. When the voltage value of PB1 is greater than 1.65V, the LED light turns off, and when it is less than 1.65V, the LED light turns on. PA2 is input as the negative end of comparator 2, when the voltage value of PA2 is greater than 1.65V, the PA4 pull down, and when it is less than 1.65V, the PA4 pull up



## 3 CRC

### 3.1 CRC\_CalculateCheckValue

此样例演示了 CRC 校验功能，通过对一个数组里的数据进行校验，得到的校验值与理论校验值进行比较，相等则 LED 灯亮，否则 LED 灯熄灭。

This example demonstrates the CRC verification function. By verifying the data in an array, the obtained verification value is compared with the theoretical verification value. If it is equal, the LED light will be on, otherwise the LED light will be off.

## 4 EXTI

### 4.1 EXTI\_ToggleLed\_IT

此样例演示了 GPIO 外部中断功能，PA15 引脚上的每一个上升沿都会产生中断，中断函数中 LED 灯会翻转一次。

This example demonstrates the GPIO external interrupt function, where each rising edge on the PA15 pin generates an interrupt, and the LED light in the interrupt function flips once.

### 4.2 EXTI\_WakeUp\_Event

此样例演示了通过 PA6 引脚唤醒 MCU 的功能。下载程序并运行后，LED 灯处于常亮状态；按下用户按键后，LED 灯处于常暗状态，且 MCU 进入 STOP 模式；拉低 PA6 引脚后，MCU 唤醒，LED 灯处于闪烁状态。

This example demonstrates the function of waking up an MCU through the PA6 pin. After downloading the program and running it, the LED light is constantly on; After pressing the user button, the LED light is in a constant dark state and the MCU enters STOP mode; After pulling down the PA6 pin, the MCU wakes up and the LED light is in a flashing state.

## **5 FLASH**

### **5.1 FLASH\_OptionByteWrite\_RST**

此样例演示了通过软件方式将 RESET 引脚改为普通 GPIO。

This example demonstrates changing the RESET pin to regular GPIO through software.

### **5.2 FLASH\_PageEraseAndWrite**

此样例演示了 flash page 擦除和 page 写功能。

This example demonstrates the flash page erase and page write functions.

### **5.3 FLASH\_SectorEraseAndWrite**

此样例演示了 flash sector 擦除和 page 写功能。

This example demonstrates the flash sector erase and page write functions.



## 6 GPIO

### 6.1 GPIO\_FastIO

本样例主要展示 GPIO 的 FAST IO 输出功能,FAST IO 速度可以达到单周期翻转速度。

This example mainly demonstrates the FAST IO output function of GPIO, which can achieve a single cycle flip speed.

### 6.2 GPIO\_Toggle

此样例演示了 GPIO 输出模式，配置 LED 引脚为数字输出模式，并且每隔 250ms 翻转一次 LED 引脚电平，运行程序，可以看到 LED 灯以 2Hz 的频率闪烁。

This example demonstrates the GPIO output mode, configuring the LED pin to be in digital output mode, and flipping the LED pin level every 250ms. Running the program, you can see that the LED light flashes at a frequency of 2Hz.

## 7 I2C

### 7.1 I2C\_TwoBoard\_CommunicationMaster\_IT

此样例演示了 I2C 通过中断方式进行通讯，主机先向从机发送 15byte 数据，然后再接收从机发送的 15byte 数据;主机、从机接收数据成功后，主机和从机板上的小灯处于“常亮”状态。

This sample demonstrates communication between I2C devices using interrupts. The master device sends 15 bytes of data to the slave device and then receives 15 bytes of data from the slave. After successful data transmission and reception between the master and slave, the LEDs on both boards remain constantly lit.

### 7.2 I2C\_TwoBoard\_CommunicationMaster\_Polling

此样例演示了 I2C 通过轮询方式进行通讯，主机先向从机发送 15byte 数据，然后再接收从机发送的 15byte 数据;主机、从机接收数据成功后，主机和从机板上的小灯处于“常亮”状态。

This sample demonstrates communication between I2C devices using polling. The master device sends 15 bytes of data to the slave device and then receives 15 bytes of data from the slave. After successful data transmission and reception between the master and slave, the LEDs on both boards remain constantly lit.

### 7.3 I2C\_TwoBoard\_CommunicationSlave\_IT

此样例演示了 I2C 通过中断方式进行通讯，主机先向从机发送 15byte 数据，然后再接收从机发送的 15byte 数据;主机、从机接收数据成功后，主机和从机板上的小灯处于“常亮”状态。

This sample demonstrates communication between I2C devices using interrupts. The master device sends 15 bytes of data to the slave device and then receives 15 bytes of data from the slave. After successful data transmission and reception between the master and slave, the LEDs on both boards remain constantly lit.

### 7.4 I2C\_TwoBoard\_MasterTxIndefiniteLengthData\_IT

此样例演示了通过中断方式，主机发送不定长数据，从机接收不定长数据。主机向从机发送 10 字节的数据 (0~9)，然后从机接收数据 (0~9) 并通过串口打印；主机向从机发送 100 字节数据 (1~100)，然后从机接收数据 (1~100) 并通过串口打印；主机向从机发送 10 字节的数据 (0~9)，然后从机接收数据 (0~9) 并通过串口打印。

This example demonstrates how the host sends variable length data and the slave receives variable length data through interrupt mode. The host sends 10 bytes of data (0-9) to the slave, and then the slave receives the data (0-9) and prints it through the serial port; The host sends 100 bytes of data (1-100) to the slave, and then the slave receives the data (1-100) and prints it through the serial port; The host sends 10 bytes of data (0-9) to the slave, and then the slave receives the data (0-9) and prints it through the serial port.

## 7.5 I2C\_TwoBoard\_SlaveRxIndefiniteLengthData\_IT

此样例演示了通过中断方式，主机发送不定长数据，从机接收不定长数据。主机向从机发送 10 字节的数据（0~9），然后从机接收数据（0~9）并通过串口打印；主机向从机发送 100 字节数据（1~100），然后从机接收数据（1~100）并通过串口打印；主机向从机发送 10 字节的数据（0~9），然后从机接收数据（0~9）并通过串口打印。

This example demonstrates how the host sends variable length data and the slave receives variable length data through interrupt mode. The host sends 10 bytes of data (0-9) to the slave, and then the slave receives the data (0-9) and prints it through the serial port; The host sends 100 bytes of data (1-100) to the slave, and then the slave receives the data (1-100) and prints it through the serial port; The host sends 10 bytes of data (0-9) to the slave, and then the slave receives the data (0-9) and prints it through the serial port.

## 8 IWDG

### 8.1 IWDG\_Reset

此样例演示了 IWDG 看门狗功能，配置看门狗重载计数值，计数 1s 后复位，然后通过调整每次喂狗的时间（main 函数 while 循环中代码），可以观察到，如果每次喂狗时间小于 1s 钟，程序能一直正常运行（LED 灯闪烁），如果喂狗时间超过 1s 钟，程序会一直复位（LED 灯熄灭）。

This example demonstrates the IWDG watchdog function, configuring the watchdog overload count value, resetting after counting for 1 second, and then adjusting each time The feeding time of the dog (code in the main function while loop) can be observed that if the feeding time is less than 1 second each time, the program Can continue to operate normally (LED flashing), if the dog feeding time exceeds 1 second, the program will continue to reset (LED light off).

## **9 PWR**

### **9.1 PWR\_DEEPSTOP\_WFE**

此样例演示了在 Deepstop 模式下，使用 GPIO 事件唤醒。

This example demonstrates using GPIO event wake-up in Deepstop mode.

### **9.2 PWR\_DEEPSTOP\_WFI**

此样例演示了在 Deepstop 模式下，使用 GPIO 中断唤醒。

This example demonstrates using GPIO interrupt wake-up in Deepstop mode.

### **9.3 PWR\_HIBERNATE\_WFE**

此样例演示了在 Hibernate 模式下，使用 GPIO 事件唤醒。

This example demonstrates using GPIO event wake-up in Hibernate mode.

### **9.4 PWR\_HIBERNATE\_WFI**

此样例演示了在 Hibernate 模式下，使用 GPIO 中断唤醒。

This example demonstrates using GPIO interrupt wake-up in Hibernate mode.

### **9.5 PWR\_SLEEP\_WFE**

此样例演示了在 sleep 模式下，使用 GPIO 事件唤醒。

This example demonstrates using GPIO event wake-up in sleep mode.

### **9.6 PWR\_SLEEP\_WFI**

此样例演示了在 sleep 模式下，使用 GPIO 中断唤醒。

This example demonstrates using GPIO interrupt wake-up in sleep mode.

### **9.7 PWR\_STOP\_WFE**

此样例演示了在 stop 模式下，使用 GPIO 事件唤醒。

This example demonstrates using GPIO event wake-up in stop mode.

## 9.8 PWR\_STOP\_WFI

此样例演示了在 stop 模式下，使用 GPIO 中断唤醒。

This example demonstrates using GPIO interrupt wake-up in stop mode.

## 10 RCC

### 10.1 RCC\_HSEOutput

此样例配置系统时钟为 HSE，并通过 MCO（PA08）引脚输出。

This sample configures the system clock as HSE and outputs it through the MCO (PA08) pin.

### 10.2 RCC\_HSIOutput

此样例配置系统时钟为 HSI，并通过 MCO（PA08）引脚输出。

This sample configures the system clock as HSI and outputs it through the MCO (PA08) pin.

### 10.3 RCC\_LSEOutput

此样例使能 LSE，并通过 MCO（PA08）引脚输出。

This sample enables the LSE and is output via the MCO (PA08) pin.

### 10.4 RCC\_LSIOutput

此样例使能 LSI，并通过 MCO（PA08）引脚输出。

This sample enables the LSI and is output via the MCO (PA08) pin.

### 10.5 RCC\_SysclockSwitch

此样例演示系统时钟切换功能。样例中配置系统时钟从 HSI 切换到 HSE，并通过 MCO（PA08）引脚输出系统时钟。

This sample demonstrates the system clock switching functionality. The sample configures the system clock to switch from HSI to HSE and outputs the system clock through the MCO (PA08) pin.

## 11 RTC

### 11.1 RTC\_AlarmSecond\_IT

此样例演示 RTC 的秒中断和闹钟中断功能。每次秒中断，在中断函数中会打印字符“RTC\_IT\_SEC”并输出实时时间。

This sample demonstrates the RTC's second interrupt and alarm interrupt functionality. Each time the second interrupt occurs, the interrupt function prints the string "RTC\_IT\_SEC" and outputs the current RTC count time.

### 11.2 RTC\_WakeUpAlarm

此样例演示通过 RTC 闹钟中断每隔 1 秒将 MCU 从 STOP 模式下唤醒，并且每次唤醒会翻转 LED，LED 翻转间隔为 1 秒。

This sample demonstrates waking up the MCU from STOP mode every 1 second using RTC alarm interrupt. Each time the MCU wakes up, the LED will toggle its state. The LED toggling interval is 1 second.

### 11.3 RTC\_WakeUpSecond

此样例演示了通过 RTC 的秒中断唤醒 MCU 的功能。下载程序并运行后，LED 灯处于常亮状态；按下用户按键后，LED 灯处于常暗状态，且 MCU 进入 STOP 模式；RTC 秒中断唤醒 MCU 后，LED 灯处于闪烁状态。

This sample demonstrates waking up the MCU using RTC second interrupt. After downloading and running the program, the LED is continuously on. Pressing the user button turns off the LED and puts the MCU into STOP mode. When the RTC second interrupt wakes up the MCU, the LED starts blinking.



## 12 SPI

### 12.1 SPI\_TwoBoards\_FullDuplexMaster\_IT

此样例是利用中断对串口外设接口（SPI）与外部设备以全双工串行方式进行通信的演示，主设备提供通信时钟 SCK，通过 MOSI/MISO 引脚发送/接收数据。从设备通过 MOSI/MISO 引脚接收/发送数据。数据以主机提供的 SCK 沿同步被移位，完成全双工通信。

This sample is a demonstration of using interrupts to communicate with a serial peripheral interface (SPI) and an external device in full-duplex serial mode. The master device provides the communication clock SCK and sends/receives data through the MOSI/MISO pin. The slave device receives/transmits data through the MOSI/MISO pins. The data is shifted synchronously along the SCK provided by the master to complete full-duplex communication.

### 12.2 SPI\_TwoBoards\_FullDuplexMaster\_Polling

此样例是通过轮询方式对串口外设接口（SPI）与外部设备以全双工串行方式进行通信的演示。主设备提供通信时钟 SCK，通过 MOSI/MISO 引脚发送/接收数据。从设备通过 MOSI/MISO 引脚接收/发送数据。数据以主机提供的 SCK 沿同步被移位，完成全双工通信。

This sample is a demonstration of the Serial Peripheral Interface (SPI) communicating with an external device in full-duplex serial mode by polling. The master device provides the communication clock SCK and sends/receives data via the MOSI/MISO pin. The slave device receives/transmits data through the MOSI/MISO pins. The data is shifted synchronously along the SCK provided by the master to complete full-duplex communication.

### 12.3 SPI\_TwoBoards\_FullDuplexSlave\_IT

此样例是利用中断对串口外设接口（SPI）与外部设备以全双工串行方式进行通信的演示，主设备提供通信时钟 SCK，通过 MOSI/MISO 引脚发送/接收数据。从设备通过 MOSI/MISO 引脚接收/发送数据。数据以主机提供的 SCK 沿同步被移位，完成全双工通信。

This sample is a demonstration of using interrupts to communicate with a serial peripheral interface (SPI) and an external device in full-duplex serial mode. The master device provides the communication clock SCK and sends/receives data through the MOSI/MISO pin. The slave device receives/transmits data through the MOSI/MISO pins. The data is shifted synchronously along the SCK provided by the master to complete full-duplex communication.

### 12.4 SPI\_TwoBoards\_FullDuplexSlave\_Polling

此样例是通过轮询方式对串口外设接口（SPI）与外部设备以全双工串行方式进行通信的演示。主设备提供通信时钟 SCK，通过 MOSI/MISO 引脚发送/接收数据。从设备通过 MOSI/MISO 引脚接收/发送数据。数据以主机提供的 SCK 沿同步被移位，完成全双工通信。

This sample is a demonstration of the Serial Peripheral Interface (SPI) communicating with an external device in full-duplex serial mode by polling. The master device provides the communication clock SCK and sends/receives data via the MOSI/MISO pin. The slave device receives/transmits data through the MOSI/MISO pins. The data is shifted synchronously along the SCK provided by the master to complete full-duplex communication.

## 13 TIM

### 13.1 TIM14\_LSI Calibrate

此样例配置 LSI 时钟从 MCO (PA8) 输出并将 TIM14 的通道 1 连接到 MCO。将系统时钟配置为 24MHz, TIM14 时钟为 240KHz, 设置重载值为 10001, 使能 TIM14 的输入捕获功能, 调整 LSI trimming 值, LED 从闪烁变成常亮表示校准完成。

This example configures the LSI clock to output from MCO (PA8) and connects channel 1 of TIM14 to MCO. Set the system clock to 24MHz, set the clock of TIM14 to 240KHz, set the overload value to 10001, enable the input capture function of TIM14, adjust the value of LSI trimming. When the LED changes from blinking to steady on, the calibration is complete

### 13.2 TIM1\_6Step

此样例演示了使用 TIM1 产生“六步 PWM 信号”, 每间隔 1ms 在 SysTick 中断中触发换向, 实现无刷电机的换向。

This example demonstrates the use of TIM1 to generate a "six step PWM signal", which triggers commutation in the SysTick interrupt every 1ms to achieve commutation of a brushless motor.

### 13.3 TIM1\_AutoReloadPreload

此样例实现了定时器的基本计数功能, 以及演示了 ARR 自动重载功能, 样例在定时器重载中断中翻转 LED 灯 修改 main.c 中的配置 `TimHandle.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_ENABLE`;使能自动重载功能, 新的 ARR 值在第四次进中断时生效, 配置 `TimHandle.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE`;禁止自动重载功能, 新的 ARR 值在第三次进中断时生效,生效后, LED 灯以 2.5HZ 的频率翻转

This sample demonstrates base count function of the timer, and show ARR register autoreload function. Example toggle LED in timer update interrupt. Modify in main.c. Set `TimHandle.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_ENABLE` to enable autoreload, and new ARR value will takes effect on the fourth interrupt generate. Set `TimHandle.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE` to disable autoreload, and new ARR value will takes effect on the third interrupt generate. After taking effect, the LED lights blinked at a frequency of 2.5HZ.

### 13.4 TIM1\_ComplementarySignals

此样例实现了定时器的互补输出功能, 三组互补共六路 pwm 输出。

This sample demonstrates complementary output function of the timer, Three sets of complementary outputs total six pwm outputs.

### 13.5 TIM1\_EncoderTI2AndTI1

此样例实现了 TIM1 中的编码器计数功能，TI1(PA3)和 TI2(PA5)作为编码器输入引脚，通过 CNT 寄存器可观察到计数器变化，通过 uwDirection 变量可观察到计数器的计数方向，通过打印数据也可观察计数方向和 CNT 寄存器计数值，打印数据 Direction = 0 为向上计数，Direction = 1 为向下计数。

This sample demonstrates encoder count function of the TIM1, TI1(PA3) and TI2(PA5) configured as encoder input pins. The change of the counter can be observed through the CNT register, and the counting direction of the counter can be observed through the uwDirection variable. The counting Direction and CNT register can also be observed by printing data. The printed data Direction = 0 indicates CounterMode:Up, and direction = 1 indicates CounterMode:down.

### 13.6 TIM1\_ExternalClockMode1

此样例演示了 TIM1 的外部时钟模式 1 功能，选择 ETR(PA12)引脚作为外部时钟输入源，并使能更新中断，在中断中翻转 LED 灯

This sample demonstrates external clock mode 1 function of the TIM1. Select the ETR(PA12) pin as the external clock input source and enable the update interrupt to flip the LED light in the interrupt.

### 13.7 TIM1\_ExternalClockMode1\_TI1F

此样例演示了 TIM1 的外部时钟模式 1 功能，选择 TI1FP(PA3)引脚作为外部时钟输入源，并使能更新中断，在中断中翻转 LED 灯

This sample demonstrates the external clock mode 1 function of TIM1, selects the TI1FP(PA3) pin as the external clock input source, and enables the update interrupt and toggle the LED light in the interrupt

### 13.8 TIM1\_ExternalClockMode2

此样例演示了 TIM1 的外部时钟模式 2 功能，选择 ETR(PA12)引脚作为外部时钟输入源，并使能更新中断，在中断中翻转 LED 灯。

This sample demonstrates the external clock mode 2 function of TIM1, selects the ETR(PA12) pin as the external clock input source, and enables the update interrupt and toggle the LED light in the interrupt

### 13.9 TIM1\_InputCapture\_TI1FP1

此样例演示了在 TIM1(PA3)输入捕获功能，PA3 输入时钟信号，TIM1 捕获成功后，会进入捕获中断，每进一次中断，翻转一次 LED

This sample demonstrates the input capture function of TIM1(PA3), PA3 input clock signal, when TIM1 capture success, will enter the capture interrupt, and toggle the LED in the interrupt

### 13.10 TIM1\_InputCapture\_XORCh1Ch2Ch3

此样例演示了 TIM1 的三通道异或输入捕获功能。配置 PA3、PA5、PA4 为通道 1、通道 2、通道 3 的输入引脚。每当有一个引脚电平变化时会触发捕获中断，并在中断处理中翻转 LED。

This example demonstrates the three channel XOR input capture function of TIM1. Configure PA3, PA5, and PA4 as input pins for channels 1, 2, and 3. Whenever a pin level changes, a capture interrupt is triggered and the LED is flipped during interrupt processing.

### 13.11 TIM1\_OC\_Toggle

此样例演示了 TIM1 的输出比较模式。将捕获/比较通道 1 (CH1) 的输出映射到 PA3，开启捕获/比较通道 1 (CH1) 并设置为比较输出翻转模式。

This example demonstrates the output comparison mode of TIM1. Map the output of capture/compare channel 1 (CH1) to PA3, turn on capture/compare channel 1 (CH1), and set it to compare output flipping mode.

### 13.12 TIM1\_OnePulseOutput

此样例演示了 TIM1 的单脉冲模式，CH2(PA05)引脚上的上升沿，触发计数器开始计数，当计数值与 CCR1 匹配时，CH1(PA03)输出高电平，直到计数器溢出，CH1 再次输出低电平，计数器溢出后，定时器停止工作，本例程脉冲宽度计算  $(TIM1\_ARR - TIM1\_CCR1) / CLK = (65535 - 16383) / 24000000 = 2.048ms$

This sample demonstrates the one pulse mode of TIM1. The rising edge on the CH2(PA05) pin triggers the counter to start counting. When the count value matches CCR1, CH1(PA03) outputs a high level. When the counter overflows, CH1 outputs the low level again. After the counter overflows, the timer stops working. This example pulse width calculation  $(TIM1\_ARR - TIM1\_CCR1) / CLK = (65,535 - 16383) / 24,000,000 = 2.048ms$

### 13.13 TIM1\_PWM

本例程输出 4 路 PWM，通道 1 的占空比为 20%，通道 2 为 40%，通道 3 为 60%，通道 4 为 80%。本例程周期为  $24000000 / 2000 / 1200 = 10Hz$ 。

This routine outputs 4 PWM channels, with a duty cycle of 20% for channel 1, 40% for channel 2, 60% for channel 3, and 80% for channel 4. The cycle of this routine is  $24000000 / 2000 / 1200 = 10Hz$ .

### 13.14 TIM1\_Update\_IT

此样例演示了 TIM1 的更新中断功能，在更新中断中翻转 LED。

This example demonstrates the update interrupt function of TIM1, flipping the LED during the update interrupt.

## 14 UART

### 14.1 UART\_HyperTerminal\_IndefiniteLengthData\_IT

此样例演示了 USART 的中断方式发送和接收不定长数据，USART 配置为 115200，数据位 8，停止位 1，校验位 None，下载并运行程序后，然后通过上位机下发任意长度个数据（不超过 128bytes），例如 0x1~0xC，则 MCU 会把接收到的数据再次发送到上位机。

This example demonstrates the interrupt method of USART to send and receive variable length data. USART is configured as 115200, with data bit 8, stop bit 1, and check bit None. After downloading and running the program, the MCU will send any length of data (not exceeding 128bytes) through the upper computer, such as 0x1~0xC. The MCU will send the received data to the upper computer again.

### 14.2 UART\_HyperTerminal\_IT

此样例演示了 UART 的中断方式发送和接收数据，UART 配置为 115200，数据位 8，停止位 1，校验位 None，下载并运行程序后，打印提示信息，然后通过上位机下发 12 个数据，例如 0x1~0xC，则 MCU 会把接收到的数据再次发送到上位机，然后打印结束信息。

This example demonstrates how to use UART to send an amount of data in interrupt mode. UART configuration is 115200 baud rate, data bit 8, stop bit 1, check bit None. After download and run the program, Print the prompt message, and then send 12 data through the upper computer, such as 0x1~0xC, the MCU will send the received data to the upper computer again, Then print the end message

### 14.3 UART\_HyperTerminal\_Polling

此样例演示了 UART 的轮询方式发送和接收数据，UART 配置为 115200，数据位 8，停止位 1，校验位 None，下载并运行程序后，打印提示信息，然后通过上位机下发 12 个数据，例如 0x1~0xC，则 MCU 会把接收到的数据再次发送到上位机，然后打印结束信息。

This example demonstrates how to use UART to send an amount of data in polling mode. UART configuration is 115200 baud rate, data bit 8, stop bit 1, check bit None. After download and run the program, Print the prompt message, and then send 12 data through the upper computer, such as 0x1~0xC, the MCU will send the received data to the upper computer again, Then print the end message