PY32F072 系列
32 位 ARM® Cortex®-M0+ 微控制器

HAL 库样例手册

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。

1.2 ADC_SingleConversion_TriggerSW_IT

此样例演示了 ADC 的中断功能，每隔 1s，软件触发 ADC 采样，在中断中通过串口打印通道 4 的 DR 值。

1.3 ADC_SingleConversion_TriggerTimer_DMA

此样例演示了 ADC 的多通道 DMA 传输的功能。

1.4 ADC_Vrefint

此样例演示了 ADC 模块的 VCC 采样功能，通过采样 VREFINT 的值，计算得出 VCC 的值，并通过串口打印出来。

2 CAN

2.1 CAN_BaseID_polling

此样例演示了采用 CAN2.0 协议标准帧轮询方式与 PCAN-View 的通信功能，MCU 首先自动向 PCAN-View 发送 8byte 数据 0x1~0x8，PCAN-View 接收到数据后，然后手动通过 PCAN-View 向 MCU 发送 ID 为 0x12F 的 8byte 数据，MCU 会自动将接收到数据通过串口打出。

2.2 CAN_ExtendedID_IT

此样例演示了采用 CAN2.0 协议扩展帧中断方式与 PCAN-View 的通信功能，MCU 首先自动向 PCAN-View 发送 8byte 数据 0x1~0x8，PCAN-View 接收到数据后，然后手动通过 PCAN-View 向 MCU 发送 ID 为 0x1234567F 的 8byte 数据，MCU 会自动将接收到数据通过串口打出。

2.3 CAN_ExtendedID_LBME_polling

此样例演示了采用 CAN2.0 协议、扩展帧、外部回环的轮询方式与 PCAN-View 的通信功能，MCU 首先自动发送 8byte 数据 0x1~0x8，MCU 接收到数据后，自动将接收到数据通过串口打出。

2.4 CAN_ExtendedID_LBMI_polling

此样例演示了采用 CAN2.0 协议、扩展帧、内部回环的轮询方式与 PCAN-View 的通信功能，MCU 首先自动发送 8byte 数据 0x1~0x8，MCU 接收到数据后，自动将接收到数据通过串口打出。

3 COMP

3.1 COMP_CompareGpioVsVrefint_IT

此样例演示了 COMP 比较器中断功能，PA00 作为比较器正端输入，VREFINT 作为比较器负端输入，当 PA0 的电压大于 Vref 电压时，LED 灯亮，小于 Vref 电压时,LED 灯灭。

3.2 COMP_CompareGpioVsVrefint_Polling

此样例演示了 COMP 比较器轮询功能，PA00 作为比较器正端输入，VREFINT 作为比较器负端输入，当 PA0 的电压大于 Vref 电压时，LED 灯亮，小于 Vref 电压时,LED 灯灭。

3.3 COMP_CompareGpioVsVrefint_WakeUpFromStop

此样例演示了 COMP 比较器唤醒功能，PA0 作为比较器正端输入，VREFINT 作为比较器负端输入，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。

4 CRC

4.1 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。

5 CTC

5.1 CTC_Autotrim

此样例演示了 CTC 使用 LSE 做参考时钟自动校准 HSI48M 时钟的功能。

6 DAC

6.1 DAC_SingleGeneration

此样例演示了 DAC 的软件触发功能,通道 PA4 能够输出 1/2 的供电电压值。

7 DIV

7.1 Division_signed

此样例演示了硬件除法器计算有符号除法。

7.2 Division_unsigned

此样例演示了硬件除法器计算无符号除法。

8 DMA

8.1 DMA_SramToSram

此样例演示了 DMA 从 SRAM 到 SRAM 传输数据的功能（SRAM 和外设之间传输的样例请参考相关外设样例工程)。

9 EXTI

9.1 EXTI_ToggleLed_IT

此样例演示了 GPIO 外部中断功能，按键 (PB0) 引脚上的每一个下降沿都会产生中断，中断函数中 LED 灯会翻转一次。

9.2 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。

10 FLASH

10.1 FLASH_BOR

此样例演示了修改 FLASH 的 BOR 功能。

10.2 FLASH_OptionByteWrite_RST

此样例演示了通过软件方式将 RESET 引脚改为普通 GPIO。

10.3 FLASH_PageEraseAndWrite

此样例演示了 flash page 擦除和 page 写功能。

10.4 FLASH_SectorEraseAndWrite

此样例演示了 flash sector 擦除和 page 写功能。

11 GPIO

11.1 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。

11.2 GPIO_Toggle

此样例演示了 GPIO 输出模式，配置 LED 引脚(PB02)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。

12 I2C

12.1 I2C_TwoBoard_CommunicationMaster_10BitAddr_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。

12.2 I2C_TwoBoard_CommunicationMaster_DMA

此样例演示了 I2C 通过 DMA 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。

12.3 I2C_TwoBoard_CommunicationMaster_DMA_MEM

此样例演示了主机 I2C 通过 DMA 方式进行通讯，从机使用 EEPROM 外设芯片 P24C32，按下 user 按键，主机先向从机写 15bytes 数据为 0x1~0xf，然后再从 EEPROM 中将写入的数据读出，读取成功后，主机板上的小灯处于“常亮”状态。

12.4 I2C_TwoBoard_CommunicationMaster_DualAddr_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。

12.5 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。

12.6 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。

12.7 I2C_TwoBoard_CommunicationSlave_10BitAddr_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。

12.8 I2C_TwoBoard_CommunicationSlave_DMA

此样例演示了 I2C 通过 DMA 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。

12.9 I2C_TwoBoard_CommunicationSlave_DualAddr_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。

12.10 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。

13 I2S

13.1 I2S_TwoBoard_CommunicationMaster_DMA

此样例是对I2S主机与I2S从机以DMA方式进行通信的演示，I2S主机先向I2S从机发送数据0x1~0x10，I2S 从机接收到数据后，再向 I2S 主机回发数据 0x0x1~0x10，当 I2S 主机、I2S 从机成功接收数据时，小灯处于常亮状态，否则小灯处于闪烁状态。

13.2 I2S_TwoBoard_CommunicationMaster_IT

此样例是对I2S主机与I2S从机以中断方式进行通信的演示，I2S主机先向I2S从机发送数据0x1~0x10，I2S 从机接收到数据后，再向 I2S 主机回发数据 0x0x1~0x10，当 I2S 主机、I2S 从机成功接收数据时，小灯处于常亮状态，否则小灯处于闪烁状态。

13.3 I2S_TwoBoard_CommunicationMaster_Polling

此样例是对 I2S 主机与 I2S 从机以 polling 方式进行通信的演示，I2S 主机先向 I2S 从机发送数据 0x1~0x10，I2S 从机接收到数据后，再向 I2S 主机回发数据 0x0x1~0x10，当 I2S 主机、I2S 从机成功接收数据时，小灯处于常亮状态，否则小灯处于闪烁状态。

13.4 I2S_TwoBoard_CommunicationSlave_DMA

此样例是对I2S主机与I2S从机以DMA方式进行通信的演示，I2S主机先向I2S从机发送数据0x1~0x10，I2S 从机接收到数据后，再向 I2S 主机回发数据 0x0x1~0x10，当 I2S 主机、I2S 从机成功接收数据时，小灯处于常亮状态，否则小灯处于闪烁状态。

13.5 I2S_TwoBoard_CommunicationSlave_IT

此样例是对I2S主机与I2S从机以中断方式进行通信的演示，I2S主机先向I2S从机发送数据0x1~0x10，I2S 从机接收到数据后，再向 I2S 主机回发数据 0x0x1~0x10，当 I2S 主机、I2S 从机成功接收数据时，小灯处于常亮状态，否则小灯处于闪烁状态。

13.6 I2S_TwoBoard_CommunicationSlave_Polling

此样例是对 I2S 主机与 I2S 从机以 polling 方式进行通信的演示，I2S 主机先向 I2S 从机发送数据 0x1~0x10，I2S 从机接收到数据后，再向 I2S 主机回发数据 0x0x1~0x10，当 I2S 主机、I2S 从机成功接收数据时，小灯处于常亮状态，否则小灯处于闪烁状态。

14 IWDG

14.1 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，程序能一直正常运行（LED 灯闪烁），如果喂狗时间超过 1s，程序会一直复位（LED 灯熄灭）。

15 LCD

15.1 LCD_Display

此样例是对单色无源液晶显示器(LCD)的演示，将偏置产生电路配置为内部电阻分压，使 LCD 全显,显示“88:88”字样。

16 LPTIM

16.1 LPTIM_Wakeup

此样例演示了 LPTIM 连续模式事件唤醒 STOP 模式。

17 OPA

17.1 OPA_VoltageFollow

此样例演示了 OPA 的电压跟随功能,PA7 为正端输入,PA5 为负端输入,PA6 为输出，PA6 会输出和 PA7 相同的电压值。

18 PWR

18.1 PVD

此样例演示了 PVD 电压检测功能，样例中配置 PB07 引脚的电压与 VREF(1.2v)进行比较，当 PB07 引脚的电压高于 VREF 时,LED 灯灭，当低于 VREF 时，LED 灯亮。

18.2 PWR_SLEEP_WFE

此样例演示了 sleep 模式下，通过 GPIO 事件唤醒功能。

18.3 PWR_SLEEP_WFI

此样例演示了 sleep 模式下，GPIO 外部中断唤醒功能。

18.4 PWR_STOP_WFE

此样例演示了 stop 模式下，通过 GPIO 事件唤醒功能。

18.5 PWR_STOP_WFI

此样例演示了 stop 模式下，通过 GPIO 中断唤醒功能。

19 RCC

19.1 RCC_HSEDiv

此样例配置系统时钟为 HSE，并通过 MCO（PA08）引脚输出

19.2 RCC_HSIOutput

此样例配置系统时钟为 HSI，并通过 MCO（PA08）引脚输出

19.3 RCC_LSEOutput

此样例配置系统时钟为 LSE，并通过 MCO（PA08）引脚输出，注意系统时钟切换为 LSE 之前，要求把 systick 中断关闭掉，因为 systick 中断默认是 1ms 一次中断，由于 LSE 时钟频率过低，systick 中断会导致程序无法正常运行。

19.4 RCC_LSIOutput

此样例配置系统时钟为 LSI，并通过 MCO（PA08）引脚输出，注意系统时钟切换为 LSI 之前，要求把 systick 中断关闭掉，因为 systick 中断默认是 1ms 一次中断，由于 LSI 时钟频率过低，systick 中断会导致程序无法正常运行。

19.5 RCC_PLLOutput

此样例配置系统时钟为 PLL，并通过 MCO（PA08）引脚输出，PLL 的输入时钟源选择 HSI。

19.6 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 LSI 切换到 HSE，并通过 MCO（PA08）引脚输出系统时钟。

20 RTC

20.1 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”，并且输出实时时间。

20.2 RTC_WakeUpAlarm

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

20.3 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。

21 SPI

21.1 SPI_TwoBoards_FullDuplexMaster_DMA

此样例是对串口外设接口（SPI）与外部设备以全双工串行方式进行通信的演示,此接口设置为主模式，为外部从设备提供通信时钟 SCK。主机通过 MOSI 引脚发送数据,从 MISO 引脚接收从机的数据，数据以主机提供的 SCK 沿同步被移位，完成全双工通信。

21.2 SPI_TwoBoards_FullDuplexMaster_IT

此样例是对串口外设接口（SPI）与外部设备以全双工串行方式进行通信的演示,此接口设置为主模式，为外部从设备提供通信时钟 SCK。主机通过 MOSI 引脚发送数据,从 MISO 引脚接收从机的数据，数据以主机提供的 SCK 沿同步被移位，完成全双工通信。

21.3 SPI_TwoBoards_FullDuplexMaster_Polling

此样例是对串口外设接口（SPI）与外部设备以全双工串行方式进行通信的演示,此接口设置为主模式，为外部从设备提供通信时钟 SCK。主机通过 MOSI 引脚发送数据,从 MISO 引脚接收从机的数据，数据以主机提供的 SCK 沿同步被移位，完成全双工通信。

21.4 SPI_TwoBoards_FullDuplexSlave_DMA

此样例是对串口外设接口（SPI）与外部设备以全双工串行方式进行通信的演示,此接口设置为主模式，为外部从设备提供通信时钟 SCK。主机通过 MOSI 引脚发送数据,从 MISO 引脚接收从机的数据，数据以主机提供的 SCK 沿同步被移位，完成全双工通信。

21.5 SPI_TwoBoards_FullDuplexSlave_IT

此样例是对串口外设接口（SPI）与外部设备以全双工串行方式进行通信的演示,此接口设置为主模式，为外部从设备提供通信时钟 SCK。主机通过 MOSI 引脚发送数据,从 MISO 引脚接收从机的数据，数据以主机提供的 SCK 沿同步被移位，完成全双工通信。

21.6 SPI_TwoBoards_FullDuplexSlave_Polling

此样例是对串口外设接口（SPI）与外部设备以全双工串行方式进行通信的演示,此接口设置为主模式，为外部从设备提供通信时钟 SCK。主机通过 MOSI 引脚发送数据,从 MISO 引脚接收从机的数据，数据以主机提供的 SCK 沿同步被移位，完成全双工通信。

22 TIM

22.1 TIM1_6Step

此样例是对高级定时器功能 “六步 PWM 的产生”的演示，通过 systick 中断作为 COM commutation 事件的触发源，实现（无刷电机的）换向下表是换向步骤，比如第一步中的 CH1 和 CH3N 为 1，即设置打开这两个通道的 PWM 输出。

22.2 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 中的第 56 行配置 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

22.3 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，此样例没有实现死区功能 CH1 -> PA8CH1N -> PA7CH2 -> PA9CH2N -> PB0CH3 -> PA10CH3N -> PB1

22.4 TIM1_ComplementarySignals_break

此样例实现了定时器的刹车功能，CH1 和 CH1N 互补 pwm 输出，接收到外部 IO 口的刹车信号（低电平）后，PWM 信号关闭，由于 BDTR.AOE 置位，所以刹车信号取消（高电平）后，继续 pwm 输出，此样例实现了死区功能。CH1 -> PA8CH1N -> PA7 刹车输入 -> PA6 通过调整 OCxE,CCxP,OISx,CCxNE,CCxNP,OISxN 的配置，可实现刹车功能的各种应用

22.5 TIM1_ComplementarySignals_break_it

此样例实现了定时器的刹车功能，CH1 和 CH1N 互补 pwm 输出，接收到外部 IO 口的刹车信号（低电平）后，PWM 信号关闭，由于 BDTR.AOE 置位，所以刹车信号取消（高电平）后，继续 pwm 输出，此样例实现了死区功能。本样例开启了刹车中断，并在刹车中断里翻转 LED 灯通过调整 OCxE,CCxP,OISx,CCxNE,CCxNP,OISxN 的配置，可实现刹车功能的各种应用

22.6 TIM1_ComplementarySignals_DeadTime

此样例实现了定时器的刹车功能，CH1 和 CH1N 互补 pwm 输出，接收到外部 IO 口的刹车信号（低电平）后，PWM 信号关闭，由于 BDTR.AOE 置位，所以刹车信号取消（高电平）后，继续 pwm 输出，此样例实现了死区功能。通过调整 OCxE,CCxP,OISx,CCxNE,CCxNP,OISxN 的配置，可实现刹车功能的各种应用

22.7 TIM1_DmaBurst_twice

此样例演示了在 TIM1 中使用 DMA 连续两次 burst 传输数据的功能，burst 每传输一次更新三个寄存器，PSC,ARR,RCR，在更新事件中断中，PA0 会进行翻转，通过逻辑分析仪监测，可看到 PA0 的翻转间隔会从第一次的 400ms，第二次 400ms，第三次 20ms,第四次及后续变为 200us，此时两次 burst 传输完成，并且 PCS,ARR,RCR 均更新完毕。

22.8 TIM1_EncoderTI2AndTI1

此样例实现了 TIM1 中的编码器计数功能，TI1(PA8)和 TI2(PA9)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。

22.9 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

22.10 TIM1_ExternalClockMode1_TI1F

此样例演示了 TIM1 的外部时钟模式 1 功能，选择 TI1FD(PA8)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

22.11 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。

22.12 TIM1_InputCapture_TI1FP1

此样例演示了在 TIM1(PA8)输入捕获功能，PA8 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

22.13 TIM1_InputCapture_XORCh1Ch2Ch3

此样例演示了在 TIM1 输入捕获功能，PA8 或 PA9 或 PA10 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

22.14 TIM1_OCToggle

此样例演示了 TIM1 比较模式下的 OC 翻转输出功能，使能 CH1(PA08),CH2(PA09),CH3(PA10),CH4(PA11)四个通道的输出功能，并且当计数器 TIMx_CNT 与 TIMx_CCRx 匹配时输出信号翻转，频率为 400KHz

22.15 TIM1_OCToggle_IT

此样例演示了 TIM1 比较模式下的中断功能，在中断中翻转 GPIO。

22.16 TIM1_OnePulseOutput

此样例演示了 TIM1 的单脉冲模式，CH2(PA09)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(PA08)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-TI1_CCR1)/CLK= (65535-16383) /32000000=1.536ms

22.17 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 8000000/(50+1)/800=196Hz

22.18 TIM1_SynchronizationEnable

定时器 1 的使能由定时器 3 控制，当定时器 3 计数时，LED 会常亮，当定时器 3 发生更新事件时，更新事件会触发定时器 1，定时器 1 开始计数后，LED 会以 5Hz 的频率进行翻转

22.19 TIM1_TIM3_Cascade

此样例实现了 TIM1 和 TIM3 级联成 32 位计数器，TIM3 做主机，TIM3 的计数溢出信号作为 TIM1 的输入时钟，通过配置 TIM1 和 TIM3 的重载寄存器值，(在 TIM1 中断回调函数中) 实现 LED 灯以 0.5Hz 频率闪烁。

22.20 TIM1_Update_DMA

此样例演示了在 TIM1 中使用 DMA 传输数据的功能，通过 DMA 从 SRAM 中搬运数据到 ARR 寄存器，实现 TIM1 周期变化，在 TIM1 第一次溢出后，PA0 会翻转，此时翻转间隔为 400ms，DMA 开始搬运数据到 TIM1_ARR，第二次 PA0 翻转间隔为 400ms，第三次翻转间隔为 100ms，第四次翻转间隔为 200ms，第四次翻转间隔为 300ms，此时 DMA 搬运结束，后续翻转间隔均为 300ms

22.21 TIM1_Update_IT

此样例演示了在 TIM1 中基本计数功能，并使能了更新中断，每次重装 ARR 值时会产生一次更新中断，并在中断中翻转 LED 灯，LED 灯会以 5Hz 的频率进行翻转。

23 USART

23.1 USART_HyperTerminal_AutoBaund_IT

此样例演示了 USART 的自动波特率检测功能，调试助手发送一个字符 0x7F，MCU 反馈字符串：Auto BaudRate Test。

23.2 USART_HyperTerminal_DMA

此样例演示了 USART 的 DMA 方式发送和接收数据，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 12 个数据，例如 0x1~0xC,则，MCU 会把接收到的数据再次发送。

23.3 USART_HyperTerminal_IT

此样例演示了 USART 的中断方式发送和接收数据，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，上位机通过 USART 会接收到 0x1-0xC,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。

23.4 USART_HyperTerminal_Polling

此样例演示了 USART 的 POLLING 方式发送和接收数据，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，通过 USART 会接收到 0x1-0xC,然后通过上位机下发 12 个数据，例如 0x1~0xC,MCU 会把接收到的数据再次发送。

24 WWDG

24.1 WWDG_IT

此样例演示了 WWDG 的提前唤醒中断功能，看门狗计数器向下计数到 0x40 时产生中断，中断中喂狗，可以确保看门狗不会复位。

24.2 WWDG_Window

此样例演示了 WWDG 的 窗口看门狗功能，配置 WWDG 的窗口上限（下限固定是 0x3F），程序中通过 delay 延时函数，确保程序是在 WWDG 计数窗口内进行喂狗动作，通过 LED 灯闪烁，可以判断窗口内喂狗并未产生复位。