PY32M030 series
32-bit ARM® Cortex®-M0+ microcontroller

LL Library Sample Manual

1 ADC

1.1 ADC_ContinuousConversion_TriggerSW_Vrefint_Init

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。

This sample demonstrates the VCC sampling function of the ADC module. By samplingthe value of VREFINT, the value of VCC is calculated and printed via the serial port.

1.2 ADC_MultiChannelSingleConversion_TriggerSW_DMA

此样例演示了 ADC 采集通过 DMA 传输并从串口打印四个通道的电压值，PA4/PA5/PA6/PA7 为模拟输入，每隔 1s 会从串口 PA2/PA3 打印当前的电压值。

This sample demonstrates ADC data acquisition using DMA and printing the voltage values of four channels via the serial port. PA4/PA5/PA6/PA7 are analog inputs, andthe current voltage values will be printed every 1s via the serial port PA2/PA3.

1.3 ADC_MultichannelSwitch

此样例演示了 ADC 的多通道切换。

This sample demonstrates the multichannel switching of ADC.

1.4 ADC_SingleConversion_AWD

此样例演示了 ADC 的模拟看门狗功能，PA4 为模拟输入，当 PA4 的电压值不在设定的上下限中，会进入看门狗中断。

This sample demonstrates the analog watchdog function of the ADC. PA4 is an analog input,and when the voltage value of PA4 is not within the set upper and lower limits, it will enter the watchdog interrupt.

1.5 ADC_SingleConversion_TriggerTimer_DMA

此样例演示了 ADC 采集通过 TIM 触发采集并通过 DMA 传输的模式，PA4 为模拟输入，每隔 1s 会从串口打印当前的电压值。

This sample demonstrates ADC data acquisition using TIM-triggered sampling and DMA transfer mode. PA4 is an analog input, and the current voltage value will be printed via the serial port every 1s.

1.6 ADC_SingleConversion_TriggerTimer_IT

此样例演示了 ADC 采集通过 TIM 触发的方式打印通道 4 的电压值，PA4 为模拟输入，每隔 1s 会从串口打印当前的电压值。

This sample demonstrates ADC data acquisition by triggering with TIM and printing the voltage value of channel 4. PA4 is an analog input, and the current voltage value willbe printed via the serial port every 1s.

1.7 ADC_TempSensor_Init

此样例演示了 ADC 模块的温度传感器功能，程序下载后，串口每隔 200ms 打印一次当前检测的温度值和对应的采样值。

This sample demonstrates the temperature sensor function of the ADC module. After downloading the program, the current temperature value and corresponding sampled value will be printed via the serial port every 200ms.

2 COMP

2.1 COMP_CompareGpioVsVrefint_HYST_Init

此样例演示了 COMP 比较器迟滞功能，PA01 作为比较器正端输入，VREFINT 作为比较器负端输入，PA00 作为比较器的输出端口，通过调整 PA01 上的输入电压，观测 PA00 引脚上的电平变化。

This sample demonstrates the hysteresis function of the COMP comparator. PA01 is used as the positive input of the comparator, VREFINT is used as the negative input, and PA00 is the output port of the comparator. By adjusting the input voltage on PA01, the level change on the PA00 pin can be observed.

2.2 COMP_CompareGpioVsVrefint_IT_Init

此样例演示了 COMP 比较器中断功能，PA01 作为比较器正端输入，VREFINT 作为比较器负端输入，运行程序，PA01 输入 1.3V 电压，LED 灯亮。

This sample demonstrates the interrupt function of the COMP comparator. PA01 is used as the positive input of the comparator, VREFINT is used as the negative input. When the program is running, if a voltage of 1.3V is applied to PA01, the LED will turn on.

2.3 COMP_CompareGpioVsVrefint_Polling_Init

此样例演示了 COMP 比较器轮询功能，PA01 作为比较器正端输入，VREFINT 作为比较器负端输入，通过调整 PA01 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 sample demonstrates the polling function of the COMP comparator. PA01 is used as thepositive input of the comparator, VREFINT is used as the negative input. By adjusting the input voltage on PA01, the LED will turn on when the comparator output state is high, and the LED will turn off when the comparator output state is low.

2.4 COMP_CompareGpioVsVrefint_WakeUpFromStop

此样例演示了 COMP 比较器唤醒功能，PA01 作为比较器正端输入，VREFINT 作为比较器负端输入，上完电 LED 灯会常亮，用户点击按键，LED 灯灭，进入 stop 模式，通过调整 PA01 上的输入电压，产生中断唤醒 stop 模式。

This sample demonstrates the wake-up function of the COMP comparator. PA01 is used as the positive input of the comparator, VREFINT is used as the negative input. After power on, the LED will be constantly on. When the user clicks the button, the LED will turn off and enter stop mode. By adjusting the input voltage on PA01, an interrupt is generated to wake up the stop mode.

2.5 COMP_CompareGpioVsVrefint_Window

此样例演示了 COMP 比较器的 window 功能，比较器 2 的 Plus 端用比较器 1 的 IO3(PA1)作为输入，VREFINT 作为比较器负端输入，当 PA1 的电压值大于 1.3V 时,LED 灯亮，小于 1.1V 时,LED 灯灭。

This sample demonstrates the window function of the COMP comparator. The positive input of comparator 2 is connected to the IO3 (PA1) of comparator 1, and VREFINT is used as the negative input of the comparator. When the voltage on PA1 is greater than 1.3V, the LED turns on. When the voltage is less than 1.1V, the LED turns off.

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 sample demonstrates the CRC checksum function. It performs a checksum on the data in an array and compares the calculated checksum with the expected checksum. If they are equal, the LED turns on; otherwise, the LED turns off.

4 DMA

4.1 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)。

This sample demonstrates the functionality of DMA data transfer between SRAM locations.Please refer to the relevant peripheral sample projects for examples of data transferbetween SRAM and peripherals.

5 EXTI

5.1 EXTI_ToggleLed_IT_Init

此样例演示了 GPIO 外部中断功能，PA12 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。

This sample demonstrates the functionality of GPIO external interrupts. Whenever a falling edge is detected on pin PA12, an interrupt is triggered, and the interrupt handler toggles the state of the LED.

5.2 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 sample demonstrates the functionality of waking up the MCU using the PA6 pin. Once the program is downloaded and running, the LED will remain lit. Pressing the user button will turn off the LED and put the MCU into STOP mode. Pulling the PA6 pin low will wake up the MCU, and the LED will start blinking.

6 FLASH

6.1 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 sample demonstrates the functionality of changing the RESET pin to a general-purpose GPIO pin using software.

6.2 FLASH_PageEraseAndWrite

此样例演示了 flash page 擦除和 page 写功能。

This sample demonstrates the functionality of erasing a flash page and programming data to a page.

6.3 FLASH_SectorEraseAndWrite

此样例演示了 flash sector 擦除和 page 写功能。

This sample demonstrates the functionality of erasing a flash sector and writing data to a page.

7 GPIO

7.1 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 sample demonstrates the FAST IO output functionality of GPIO, which can achieve a single-cycle toggling speed.

7.2 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 100ms 翻转一次 LED 引脚电平，运行程序，可以看到 LED 灯闪烁。

This sample demonstrates GPIO output mode by configuring the LED pin as a digital output. The LED pin's level is toggled every 100ms, causing the LED to blink. Run the program to observe the LED blinking.

7.3 GPIO_Toggle_Init

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 100ms 翻转一次 LED 引脚电平，运行程序，可以看到 LED 灯闪烁。

This sample demonstrates GPIO output mode by configuring the LED pin as a digital output. The LED pin level is toggled every 100ms. When running the program, the LED will blink.

8 I2C

8.1 I2C_TwoBoards_MasterTx_SlaveRx_Polling

此样例演示了主机 I2C、从机 I2C 通过轮询方式进行通讯，当按下从机单板的用户按键，再按下主机单板的用户按键后，主机 I2C 向从机 I2C 发送"LED ON"数据。当主机 I2C 成功发送数据，从机 I2C 成功接收数据时，主机单板和从机单板 LED 灯分别亮起。

This sample demonstrates I2C communication between a master and a slave using polling. When the user button on the slave board is pressed, followed by pressing the user button on the master board, the master I2C sends the "LED ON" data to the slave I2C. When the master successfully sends the data and the slave successfully receives the data, the LEDs on the master and slave boards will light up.

8.2 I2C_TwoBoard_CommunicationMaster_DMA_Init

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。

This sample demonstrates I2C communication using DMA. The master sends 15 bytes of data to the slave, 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 the master and slave boards will remain constantly lit.

8.3 I2C_TwoBoard_CommunicationMaster_DMA_MEM_Init

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。

This sample demonstrates I2C communication using DMA on the master. The slave device uses the EEPROM peripheral chip P24C32. When the user button is pressed on the master, the master writes 15 bytes of data (0x1 to 0xf) to the slave. Then, the master reads the written data from the EEPROM. After successful data retrieval, the LED on the master board will remain constantly lit.

8.4 I2C_TwoBoard_CommunicationMaster_IT_Init

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 I2C communication using interrupts. The master sends 15 bytes of data to the slave, 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 the master and slave boards will remain constantly lit.

8.5 I2C_TwoBoard_CommunicationMaster_Polling_Init

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 I2C communication using polling. The master sends 15 bytes ofdata to the slave, 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 the master and slave boards will remain constantly lit.

8.6 I2C_TwoBoard_CommunicationSlave_DMA_Init

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。

This sample demonstrates I2C communication using DMA. The master sends 15 bytes of data to the slave, 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 the master and slave boards will remain constantly lit.

8.7 I2C_TwoBoard_CommunicationSlave_IT_Init

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 I2C communication using interrupts. The master sends 15 bytes of data to the slave, 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 the master and slave boards will remain constantly lit.

9 IWDG

9.1 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 灯不亮）。

This sample demonstrates the IWDG watchdog function. It configures the watchdog reload counter value to reset after counting for 1 second. By adjusting the time for feeding the watchdog (code in the main function's while loop), you can observe that if the feeding time is less than 1 second, the program will continue to run normally (LED blinking), but if the feeding time exceeds 1 second, the program will keep resetting (LED off).

10 LED

10.1 LED

此样例演示了 LED 的控制数码管显示的功能，样例中同时控制 4 个数码管，4 个数码管逐步显示“8888”。

This sample demonstrates the LED control digital tube display function, sample at the same time control 4 digital tubes, 4 digital tubes gradually displayed "8888".

10.2 LED_IT

此样例演示了 LED 的控制数码管显示的功能，样例中同时控制 4 个数码管，4 个数码管的显示内容可以在中断中实时修改。

This sample demonstrates the control of LEDs to display on a 4-digit display. The example simultaneously controls 4 digits, and the display content of the 4-digit display can be modified in real-time in an interrupt.

10.3 LED_IT_Init

此样例演示了 LED 的控制数码管显示的功能，样例中同时控制 4 个数码管，4 个数码管的显示内容可以在中断中实时修改。

This sample demonstrates the control of LEDs to display on a 4-digit display. The example simultaneously controls 4 digits, and the display content of the 4-digit display can be modified in real-time in an interrupt.

11 LPTIM

11.1 LPTIM_ARR_Init

此样例演示了 LPTIM 单次计数功能，配置 LPTIM 计数周期为 0.5s，在 ARR 中断回调函数中翻转 LED 灯，并再次启动单次计数模式。

This sample demonstrates the single-count function of LPTIM. It configures the LPTIM count period as 0.5s. In the ARR interrupt callback function, it toggles the LED and restarts the single-count mode.

11.2 LPTIM_WakeUp

此样例演示了 LPTIM 中断唤醒 stop 模式，每次唤醒后再次进入 stop 模式，每 200ms 唤醒一次。

This sample demonstrates the LPTIM interrupt wake-up from stop mode. It enters stop mode after each wake-up and wakes up every 200ms.

12 PWR

12.1 PWR_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。

This sample demonstrates the PVD (Power Voltage Detector) voltage detection function. It configures PB07 pin to compare its voltage with VREF (1.2V). When the voltage of PB07 exceeds VREF, the LED turns off. When it is lower than VREF, the LED lights up.

12.2 PWR_SLEEP_WFE

此样例演示了在 sleep 模式下，使用 GPIO 事件唤醒。

This sample demonstrates using GPIO event to wake up the MCU from sleep mode.

12.3 PWR_SLEEP_WFI

此样例演示了在 sleep 模式下，使用 GPIO 中断唤醒。

This sample demonstrates using GPIO interrupt to wake up the MCU from sleep mode.

12.4 PWR_STOP_WFE

此样例演示了在 stop 模式下，使用 GPIO 事件唤醒。

This sample demonstrates using GPIO event to wake up the MCU from stop mode.

12.5 PWR_STOP_WFI

此样例演示了在 stop 模式下，使用 GPIO 中断唤醒。

This sample demonstrates using GPIO interrupt to wake up the MCU from stop mode.

13 RCC

13.1 RCC_HSE_OUTPUT

此样例演示了时钟输出功能，可输出 HSE 波形。

This sample demonstrates the clock output function, which can output the HSE waveform.

13.2 RCC_HSI_OUTPUT

此样例演示了时钟输出功能，可输出 HSI 波形。

This sample demonstrates the clock output function, which can output the HSI waveform.

13.3 RCC_LSE_OUTPUT

此样例演示了将系统时钟设置为 LSE，并通过 MCO 引脚输出系统时钟。

This example demonstrates setting the system clock to LSE and outputting the system clock through the MCO pin.

13.4 RCC_LSI_OUTPUT

此样例演示了将系统时钟设置为 LSI，并通过 MCO 引脚输出系统时钟。

This example demonstrates setting the system clock to LSI and outputting the system clock through the MCO pin.

13.5 RCC_PLL_OUTPUT

此样例演示了时钟输出功能，可输出以 HSI 为源的 PLL 波形。

This sample demonstrates the clock output function, which can output PLL waveform with HSI as the source.

13.6 RCC_Sysclock_Switch

此样例演示了时钟切换，由 HSI 切换至 HSE（24MHz）。

This sample demonstrates clock switching from HSI to HSE (24MHz).

14 RTC

14.1 RTC_Alarm_Init

此样例演示 RTC 的闹钟中断功能，在数组 aShowTime 中显示当前时间，在数组 aShowDate 中显示当前日期，当达到闹钟值时，LED 灯会亮起。

This sample demonstrates the alarm interrupt function of the RTC. It displays the current time in the array aShowTime and the current date in the array aShowDate. When the alarm value is reached, the LED will light up.

14.2 RTC_WakeUpAlarm_Init

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

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

14.3 RTC_WakeUpSecond_Init

此样例演示通过 RTC 秒中断从 STOP 模式下唤醒，唤醒后，小灯处于闪烁状态；否则处于熄灭状态。

This sample demonstrates waking up the MCU from STOP mode using RTC second interrupt. After waking up, the LED will be in a blinking state. Otherwise, it will be turned off.

15 SPI

15.1 SPI_TwoBoards_FullDuplexMaster_DMA_Init

此样例是利用 DMA 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。

This sample demonstrates communication with an external device in full-duplex serial mode using DMA for the SPI peripheral interface. The interface is set to master mode, providing the communication clock SCK to the external slave device. The master sends data through the MOSI pin and receives data from the slave through the MISO pin. The data is shifted along the provided SCK edge, enabling full-duplex communication.

15.2 SPI_TwoBoards_FullDuplexMaster_IT_Init

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,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。

This sample demonstrates communication with an external device in full-duplex serial mode using interrupts for the SPI peripheral interface. The interface is set to master mode, providing the communication clock SCK to the external slave device. The master sends data through the MOSI pin and receives data from the slave through the MISO pin. The data is shifted along the provided SCK edge, enabling full-duplex communication.

15.3 SPI_TwoBoards_FullDuplexSlave_DMA_Init

此样例是利用 DMA 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,此接口设置为从模式。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。

This sample demonstrates the communication between the serial peripheral interface (SPI) and an external device in full-duplex mode using DMA. The SPI interface is configured as the slave. The host sends data through the MOSI pin, and receives data from the slave through the MISO pin. The data is shifted synchronously with the SCK provided by the host, achieving full-duplex communication.

15.4 SPI_TwoBoards_FullDuplexSlave_IT_Init

此样例是利用 DMA 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,此接口设置为从模式。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。

This sample demonstrates the communication between the serial peripheral interface (SPI) and an

external device in full-duplex mode using DMA. The SPI interface is configured as the slave. The host sends data through the MOSI pin and receives data from the slave through the MISO pin. The data is shifted synchronously with the SCK provided by the host, achieving full-duplex communication.

16 TIM

16.1 TIM1_6Step_Init

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 sample demonstrates the generation of "six-step PWM signals" using TIM1. The commutation of the brushless motor is triggered every 1ms in the SysTick interrupt.

16.2 TIM1_ComplementarySignals_Break_DeadTime_Init

此样例演示了使用 TIM1 输出三路频率为 10Hz 占空比分别为 25%、50%、75%的 PWM 波形以及他们的互补信号，并实现了 250ns 的死区和刹车功能。

This sample demonstrates the generation of three PWM waveforms with frequencies of 10Hz and duty cycles of 25%, 50%, and 75% using TIM1. It also generates their complementary signals and implements a 250ns dead-time and brake function.

16.3 TIM1_ComplementarySignals_Init

此样例演示了使用 TIM1 输出三路频率为 10Hz 占空比分别为 25%、50%、75%的 PWM 波形以及他们的互补信号。

This sample demonstrates the generation of three PWM waveforms with frequencies of 10Hz and duty cycles of 25%, 50%, and 75% using TIM1. It also generates their complementary signals.

16.4 TIM1_DmaBurst_Init

此样例演示了 TIM1 的 DMA Burst 传输，配置 TIM1 为 PWM 模式，更新中断触发 DMA 传输请求。每次产生更新中断时将 TIM1DataBuff[]中的值按顺序写入 RCR 和 CCR1 寄存器，改变 PWM 脉冲的占空比和该占空比的脉冲数量。

This sample demonstrates the DMA Burst transfer of TIM1. It configures TIM1 in PWM mode and triggers DMA transfer requests on update interrupt. Each time an update interrupt occurs, the values in TIM1DataBuff[] are sequentially written to RCR and CCR1 registers, changing the duty cycle and the number of pulses for the PWM waveform.

16.5 TIM1_EncoderTI2AndTI1_Init

此样例演示了 TIM1 的编码器接口模式。TIM1 配置为编码器接口模式 3，PA8 和 PA9 配置为通道 1 和通道 2 当 PA8 输入信号的上升沿在前，PA9 输入信号上升沿在后时 TIM1 向上计数，反之向下计数。开

启通道 1 和通道 2 的捕获中断，在中断中打印当前 CNT 值。

This sample demonstrates the encoder interface mode of TIM1. TIM1 is configured in encoder interface mode 3, with PA8 and PA9 configured as channel 1 and channel 2, respectively. When the rising edge of the input signal on PA8 occurs before the rising edge of the input signal on PA9, TIM1 counts up; otherwise, it counts down. The capture interrupts for channel 1 and channel 2 are enabled, and the current CNT value is printed in the interrupt.

16.6 TIM1_ExternalClockMode1_Init

此样例演示了 TIM1 的外部时钟模式 1 的功能，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 functionality of TIM1 in external clock mode 1. It selects ETR (PA12) pin as the external clock input source and enables the update interrupt. In the interrupt, the LED light is toggled.

16.7 TIM1_ExternalClockMode1_TI1F_Init

此样例演示了 TIM1 的外部时钟模式 1 的功能，选择 TI1F_ED(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。

This sample demonstrates the functionality of TIM1 in external clock mode 1. It selects TI1F_ED (PA8) pin as the external clock input source and enables the update interrupt. In the interrupt, the LED light is toggled.

16.8 TIM1_ExternalClockMode2_Init

此样例演示了 TIM1 的外部时钟模式 2 的功能，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 functionality of TIM1 in external clock mode 2. It selects the ETR (PA12) pin as the external clock input source and enables the update interrupt. In the interrupt, the LED light is toggled.

16.9 TIM1_InputCapture_TI1FP1_Init

此样例演示了 TIM1 的输入捕获功能，配置 PA3 作为输入捕获引脚，PA3 每检测到一个上升沿触发捕获中断在捕获中断回调函数中翻转 LED 灯。

This sample demonstrates the input capture functionality of TIM1 by configuring PA3 as the input capture pin. Whenever an rising edge is detected on PA3, it triggers the capture interrupt and toggles the LED in the interrupt callback function.

16.10 TIM1_InputCapture_XORCh1Ch2Ch3

此样例演示了 TIM1 的三通道异或输入捕获功能。配置 PA8、PA9、PA10 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 sample demonstrates the XOR input capture functionality of TIM1 using three channels: PA8, PA9, and PA10 as the input pins for channel 1, channel 2, and channel 3, respectively. Whenever there is a change in the level of any of the input pins, it triggers the capture interrupt and toggles the LED in the interrupt handler.

16.11 TIM1_OCToggle

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 PA8，开启捕获/比较通道 1（CH1）并设置为比较输出翻转模式，同时开启捕获/比较中断，每次计数值与比较值匹配时翻转输出电平，在捕获/比较中断处理中翻转 LED 灯。

This sample demonstrates the output compare mode of TIM1. The output of capture/compare channel 1 (CH1) is mapped to pin PA8. Capture/compare channel 1 (CH1) is enabled and set to compare output toggle mode. Capture/compare interrupt is also enabled. Whenever the counter value matches the compare value, the output level will toggle. In the interrupt handler of capture/compare, the LED will also toggle.

16.12 TIM1_OnePulseOutput

此样例演示了 TIM1 的单脉冲模式。配置 TIM1 为从模式触发模式，触发源为 TI2FP2，通道 1 为 PWM2 模式，映射到 PA8，通道 2 为输入模式，映射到 PA9。当 PA9 上检测到一个上升沿时，PA8 延迟 20ms 后产生一个宽度为 80ms 的脉冲。

This sample demonstrates the single pulse mode of TIM1. TIM1 is configured in slave mode trigger mode with TI2FP2 as the trigger source. Channel 1 is configured as PWM mode 2 and mapped to pin PA8, while channel 2 is configured as input mode and mapped to pin PA9. When an rising edge is detected on PA9, a 20ms delay is applied, and then PA8 will output a pulse with a width of 80ms.

16.13 TIM1_PWM_Init

此样例演示了使用 TIM1 PWM2 模式输出三路频率为 10Hz 占空比分别为 25%、50%、75%的 PWM 波形。

This sample demonstrates the use of TIM1 PWM2 mode to output three PWM waveforms with frequencies of 10Hz and duty cycles of 25%, 50%, and 75% respectively.

16.14 TIM1_SynchronizationEnable

此样例演示了 TIM 的同步触发模式。TIM3 设置为主模式触发，TIM1 设置从模式触发。程序复位后 TIM3

开始计数，触发更新事件后向 TIM1 发送同步信号，TIM1 开始计数，并在更新中断中翻转 LED。

This sample demonstrates the synchronous trigger mode of TIM. TIM3 is set as the master trigger mode, and TIM1 is set as the slave trigger mode. After the program is reset, TIM3 starts counting. After the update event is triggered, it sends a synchronization signal to TIM1. TIM1 starts counting and toggles the LED in the update interrupt.

16.15 TIM1_TIM3_Cascade

此样例演示了 TIM1 和 TIM3 级联成 32 位计数器，TIM3 做主机，TIM3 的溢出信号作为 TIM1 的输入时钟。TIM3 每 1ms 计数一次，计数 1000 次后产生溢出，TIM1 计数一次。

This sample demonstrates the cascading of TIM1 and TIM3 as a 32-bit counter, with TIM3 as the master and the overflow signal of TIM3 as the input clock of TIM1. TIM3 counts every 1ms, and after counting 1000 times, it overflows and TIM1 counts once.

16.16 TIM1_TimeBase_Init

此样例演示了 TIM1 的重装载功能，配置计数周期为 1s，在 ARR 中断回调函数中翻转 LED。

This sample demonstrates the reload function of TIM1. It configures the counting period as 1 second and toggles the LED in the ARR interrupt callback function.

16.17 TIM1_Update_DMA_Init

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 更新周期变化,TIM1 第一次溢出后 LED 会翻转,此次翻转时间间隔为 1000ms,DMA 将数据搬运到 TIM1_ARR,第二次 LED 翻转间隔为 900ms,以此类推,最后 LED 翻转间隔为 100msDMA 搬运结束,LED 保持 100ms 的翻转间隔闪烁。

This sample demonstrates the use of DMA to transfer data in TIM1, copying data from SRAM to the ARR register to achieve varying update periods for TIM1. After the first overflow of TIM1, the LED will toggle, with a time interval of 1000ms. After the data is transferred to TIM1_ARR using DMA, the LED toggling interval gradually decreases: 900ms, 800ms, 700ms, 600ms, 500ms, 400ms, 300ms, 200ms, 100ms. Finally, the LED will blink with a constant toggling interval of 100ms.

17 USART

17.1 USART_HyperTerminal_AutoBaund_IT_Init

此样例演示了 USART 的自动波特率检测功能,上位机发送 1 字节的波特率检测字符 0x55,如果 MCU 检测成功，则返回字符：Auto BaudRate Test。

This sample demonstrates the automatic baud rate detection feature of USART. The PC sends a 1-byte baud rate detection character 0x55, and if the MCU detects it successfully, it returns the string "Auto BaudRate Test".

17.2 USART_HyperTerminal_DMA_Init

此样例演示了 USART 的 DMA 方式发送和接收数据，USART 配置为 9600，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 USART to send an amount of data in DMA mode. USART configuration is 9600 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.

17.3 USART_HyperTerminal_IndefiniteLengthData_IT_Init

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.

17.4 USART_HyperTerminal_IT_Init

此样例演示了 USART 的中断方式发送和接收数据，USART 配置为 9600，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 USART to send an amount of data in interrupt mode. USART configuration is 9600 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.

17.5 USART_HyperTerminal_Polling_Init

此样例演示了 USART 的轮询方式发送和接收数据，USART 配置为 9600，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 USART to send an amount of data in polling mode. USART configuration is 9600 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.

18 UTILS

18.1 UTILS_ConfigureSystemClock

本样例主要演示如何配置 SYSCLK(系统时钟), HCLK(AHB 时钟), PCLK(APB 时钟)。通过 MCO 输出系统时钟 48MHz。

This sample demonstrates how to configure SYSCLK (system clock), HCLK (AHB clock),and PCLK (APB clock), and outputs the system clock of 48MHz through MCO.

18.2 UTILS_ReadDeviceInfo

本样例主要读取 DBGMCU->IDCODE 寄存器和 UID 的值,并通过串口打印出来。

This sample mainly reads the values of the DBGMCU->IDCODE register and UID, and prints them out via the serial port.

19 WWDG

19.1 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。

This sample demonstrates the early wake-up interrupt function of the WWDG. When the watchdog counter counts down to 0x40, an interrupt is generated. In the interrupt handler, the watchdog is fed to ensure that the watchdog does not reset the system.

19.2 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。

This sample demonstrates the window watchdog (WWDG) function. The upper limit of the WWDG window is configured (the lower limit is fixed at 0x3F). In the program, a delay function is used to ensure that the watchdog feeding action occurs within the WWDG counting window. The LED blinking indicates whether the watchdog is fed within the window, thereby verifying that no reset occurs.