PY32F002A 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,反推 VCC 电压。

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_MultichannelSwitch

此样例演示了 ADC 的多通道切换。

This sample demonstrates the multichannel switching of ADC.

1.3 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.

1.4 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 PA2/PA3 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.

1.5 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，用户点击按钮,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_Computing_Results

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 EXTI

4.1 EXTI_Event_Init

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.

4.2 EXTI_IT_Init

此样例演示了 GPIO 外部中断功能，PB2 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 PB2, an interrupt is triggered, and the interrupt handler toggles the state of the LED.

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

5.2 FLASH_PageEraseAndWrite

此样例演示了 flash page 擦除和 page 写功能。

5.3 FLASH_SectorEraseAndWrite

此样例演示了 flash sector 擦除和 page 写功能。

6 GPIO

6.1 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.

6.2 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.

7 I2C

7.1 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.

7.2 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.

7.3 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.

7.4 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.

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).

9 LPTIM

9.1 LPTIM_Wakeup

此样例演示了 LPTIM 中断唤醒 stop 模式，500ms 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 500ms.

10 PWR

10.1 PWR_SLEEP_WFI

此样例演示了通过 WFI(wait for interrupt)指令进入 sleep 模式，使用 GPIO 中断唤醒

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

10.2 PWR_STOP_WFI

此样例演示了通过 WFI(wait for interrupt)指令进入 stop 模式，使用 GPIO 中断唤醒

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

11 RCC

11.1 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.

11.2 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.

11.3 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.

12 SPI

12.1 SPI_TwoBoards_FullDuplexMaster_IT_Init

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。

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.

12.2 SPI_TwoBoards_FullDuplexSlave_IT_Init

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。

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.

13 TIMER1

13.1 TIM1_InputCapture_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.

13.2 TIM1_PWM3CH_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.

13.3 TIM1_TimeBase_Init

此样例演示了 TIM1 的重装载功能和自动重载预装载功能，在初始化阶段配置重载值为 1000，在 ARR 中断回调函数中重新设置重载值为 500。每次进入中断时翻转 LED。可以通过注释 main.c 中 LL_TIM_EnableARRPreload(TIM1); 代码关闭自动重载预装载功能。若关闭自动重载预装载功能，新的重载值将在第一次进入中断后立即生效，则 LED 引脚前 1 次翻转为 1000ms 后续保持 500ms。若开启自动重载预装载功能，新的重载值将在下一次进入中断后生效，则 LED 引脚前 2 次翻转为 1000ms 后续保持 500ms。

This sample demonstrates the reload function and the auto-reload preload function of TIM1. configure the reload value to 1000 during the initialisation phase. The reload value is reset to 500 in the ARR interrupt callback function. flipping the LED each time an interrupt is entered. You can turn off the automatic reload preload by commenting the LL_TIM_EnableARRPreload(TIM1); code in main.c. function in main.c. If you disable the automatic reload preload function, the new reload value will take effect immediately after the first interrupt entry, then the first flip-flop of the LED pin will be 1000ms and the subsequent flip-flops will be 500ms. The first flip of the LED pin will be 1000ms and the subsequent hold will be 500ms. If the auto reload preload function is enabled, the new reload value will take effect after the next interrupt entry, then the first 2 times of LED pin flip-flop will be 1000ms and then hold for 500ms. 1000ms and then hold for 500ms.

14 USART

14.1 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".

14.2 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.

14.3 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.

14.4 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.

15 UTILS

15.1 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 24MHz。

This sample demonstrates how to configure SYSCLK (system clock), HCLK (AHB clock),and PCLK (APB clock), and outputs the system clock of 24MHz through MCO.

15.2 UTILS_ReadDeviceInfo

本样例主要读取 DBGMCU->IDCODE 寄存器和 UID 的值。UID Word0 表示 LotNumber, Word1 表示 WaferNumber, Word2 表示 X 和 Y 的坐标。

This sample mainly reads the values of DBGMCU->IDCODE register and UID. UID Word0 means LotNumber, Word1 means WaferNumber, Word2 means X and Y coordinates.