PY32L020 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_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.

1.3 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.

1.4 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.

1.5 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，PA04 作为比较器负端输入，1/2VCCA 作为正端输入，通过调整 PA04 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 PA04 as the negative input of the comparator and 1/2VCCA as the positive input. By adjusting the input voltage on PA04, 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，PA04 作为比较器负端输入，1/2VCCA 作为正端输入，通过调整 PA04 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 PA04 as the negative input of the comparator and 1/2VCCA as the positive input. By adjusting the input voltage on PA04, 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_WakeUpFromSleep

此样例演示了 COMP 比较器唤醒功能，PA04 作为比较器负端输入，1/2VCC 作为比较器正端输入，上完电 LED 灯会常亮，用户点击按钮，LED 灯灭，进入 sleep 模式，通过调整 PA04 上的输入电压，产生中断唤醒 sleep 模式。

This example demonstrates the wake-up function of the COMP comparator, with PA04 as the negative input and 1/2VCC as the positive input. After power on, the LED light will remain on. When the user clicks the button, the LED light will go out and enter sleep mode. By adjusting the input voltage on PA04, an interrupt wake-up sleep mode is generated.

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 IO2(1/2VCCA)作为输入，PB0 作为比较器负端输入，当 PB0 的电压值大于 1.65V 时,LED 灯灭，小于 1.65V 时,LED 灯亮。

This example demonstrates the window function of the COMP comparator. The Plus end of comparator 1 uses the IO2 (1/2VCCA) of comparator 2 as the input, and PB0 as the negative end input. When the voltage value of PB0 is greater than 1.65V, the LED light turns off, and when it is less than 1.65V, the LED light turns on.

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，PA0 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 example demonstrates the GPIO external interrupt function, where each falling edge on the PA0 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(PA1)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 (PA1) 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 example demonstrates I2C communication through interrupt mode. The master first sends 15byte data to the slave, and then receives the 15byte data sent by the slave. After the master and slave successfully receive the data, the small lights on the master and slave boards are in a "constant on" state.

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 example demonstrates I2C communication through polling. The host first sends 15byte data to the slave, and then receives the 15byte data sent by the slave. After the host and slave receive the data successfully, the small lights on the host and slave boards are in a "constantly on" state.

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 example demonstrates I2C communication through interrupt mode. The master first sends 15byte data to the slave, and then receives the 15byte data sent by the slave. After the master and slave successfully receive the data, the small lights on the master and slave boards are in a "constant on" state.

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 LPTIM

9.1 LPTIM_Wakeup

此样例演示了 LPTIM 连续模式事件唤醒 STOP 模式。

This example demonstrates the LPTIM continuous mode event wake-up STOP mode.

10 PWR

10.1 PWR_DEEPSTOP_WFE

此样例演示了在 deep stop 模式下，使用 GPIO 事件唤醒。

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

10.2 PWR_DEEPSTOP_WFI

此样例演示了在 deep stop 模式下，使用 GPIO 中断唤醒。

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

10.3 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.

10.4 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.

10.5 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.

10.6 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.

11 RCC

11.1 RCC_HSEBypassOutput

此样例演示了时钟输出功能，可输出 HSE 波形。

This example demonstrates the clock output function, which can output HSE waveforms.

11.2 RCC_HSIOutput

此样例配置系统时钟为 HSI，并通过 MCO（PA07）引脚输出。

This example configures the system clock as HSI and outputs it through the MCO (PA07) pin.

11.3 RCC_LSEOutput

此样例配置系统时钟为 LSE，并通过 MCO（PA07）引脚输出。

This example configures the system clock as LSE and outputs it through the MCO (PA07) pin.

11.4 RCC_LSIOutput

此样例配置系统时钟为 LSI，并通过 MCO（PA07）引脚输出。

This example configures the system clock as LSI and outputs it through the MCO (PA07) pin.

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,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 example is a demonstration of using interrupts to communicate with the serial peripheral interface (SPI) and external devices in a full duplex serial mode. This interface is set as the main mode and provides communication clock SCK for external slave devices. The master sends data through the MOSI pin and receives data from the slave through the MISO pin. The data is synchronously shifted along the SCK provided by the master, completing 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,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 example is a demonstration of communication between the serial peripheral interface (SPI) and external devices in full duplex serial mode through polling. This interface is set as the main mode and provides communication clock SCK for external slave devices. The host sends data through the MOSI pin and receives data from the slave through the MISO pin. The data is synchronously shifted along the SCK provided by the host, completing 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,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 example is a demonstration of using interrupts to communicate with the serial peripheral interface (SPI) and external devices in a full duplex serial mode. This interface is set as the main mode and provides communication clock SCK for external slave devices. The master sends data through the MOSI pin and receives data from the slave through the MISO pin. The data is synchronously shifted along the SCK provided by the master, completing 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,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 example is a demonstration of communication between the serial peripheral interface (SPI) and external devices in full duplex serial mode through polling. This interface is set as the main mode and provides communication clock SCK for external slave devices. The host sends data through the MOSI pin and receives data from the slave through the MISO pin. The data is synchronously shifted along the SCK provided by the host, completing full duplex communication.

13 TIM

13.1 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.

13.2 TIM1_InputCapture

此样例演示了 TIM1 的输入捕获功能，配置 PA0 作为输入捕获引脚，PA0 每检测到一个下降沿触发捕获中断,在捕获中断回调函数中翻转 LED 灯。

This example demonstrates the input capture function of TIM1, where PA0 is configured as the input capture pin. Every time PA0 detects a falling edge, it triggers a capture interrupt and flips the LED light in the capture interrupt callback function.

13.3 TIM1_InputCapture_XORCh1Ch2Ch3

此样例演示了 TIM1 的三通道异或输入捕获功能。配置 PA0、PA3、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 PA0, PA3, 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.

13.4 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 PA5，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 PA5, turn on capture/compare channel 1 (CH1), and set it to compare output flipping mode.

13.5 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.

13.6 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 USART

14.1 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。

This example demonstrates USART's automatic Baud detection function. The debugging assistant sends a character 0x7F and MCU feedback string: Auto BaudRate Test.

14.2 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。

This example demonstrates the interrupt method of USART to send and receive data. USART is configured as 115200, with data bit 8, stop bit 1, and check bit None. After downloading and running the program, the upper computer will receive 0x1-0xC through USART, 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.

14.3 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。

This example demonstrates the POLLING method of USART to send and receive data. The USART configuration is 115200, with data bit 8, stop bit 1, and check bit None. After downloading and running the program, it will receive 0x1-0xC through USART, and then send 12 data through the upper computer, such as 0x1-0xC. The MCU will send the received data again.