PY32F030 系列
32 位 ARM® Cortex®-M0+ 微控制器

LL 库样例手册

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。

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。

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 的电压值大于 1/2 供电电压时，LED 灯会闪烁

1.4 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 PA2/PA3 打印当前的电压值。

1.5 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。

1.6 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。

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。

2.2 COMP_CompareGpioVsVrefint_IT_Init

此样例演示了 COMP 比较器功能，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。

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。

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。

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。

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。

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），数据传输完成 LED 灯闪烁，传输失败则 LED 灯熄灭

5 EXTI

5.1 EXTI_Event

此样例演示了通过 PA12 引脚唤醒 MCU 的功能，下载程序后，MCU 会进入 STOP 模式，按下用户键 MCU 会退出 STOP 模式，然后 LED 会以 500ms 的间隔进行翻转

5.2 EXTI_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

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

6.2 FLASH_PageEraseAndWrite

此样例演示了 flash page 擦除和 page 写功能，通过 keil debug 仿真界面，可观察 flash 存储器中是否擦除成功和 page 写成功。

6.3 FLASH_SectorEraseAndWrite

此样例演示了 flash sector 擦除和 page 写功能，通过 keil debug 仿真界面，可观察 flash 存储器中是否擦除成功和 page 写成功。

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；

7.2 GPIO_Toggle

此样例演示了 GPIO 输出模式，配置 LED 引脚(PA11)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

7.3 GPIO_Toggle_Init

此样例演示了 GPIO 输出模式，配置 LED 引脚(PA11)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

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 SW1 按键，再下主机单板的 SW1 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 LED2 灯分别亮起。

8.2 I2C_TwoBoard_CommunicationMaster_DMA_Init

此样例演示了主机 I2C 通过 DMA 方式进行通讯，从机使用 PY32F030x，按下 user 按键，主机先向从机发送 15 byte 数据，然后再接收从机发送的数据，数据接收后，会保存在 aRxBuffer 数组中，接收数据为 0x1~0xf。

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 15 bytes 数据为 0x1~0xf，然后再从 EEPROM 中读取数据，数据读取后，会保存在 aRxBuffer 数组中，接收数据为 0x1~0xf。

8.4 I2C_TwoBoard_CommunicationMaster_IT_Init

此样例演示了主机 I2C 通过中断方式进行通讯，从机使用 PY32F030x，按下 user 按键，主机先向从机发送 15 byte 数据，然后再接收从机发送的数据，数据接收后，会保存在 aRxBuffer 数组中，接收数据为 0x1~0xf。

8.5 I2C_TwoBoard_CommunicationMaster_Polling_Init

此样例演示了主机 I2C 通过 POLLING 方式进行通讯，从机使用 PY32F030x，按下 user 按键，主机先向从机发送 15 byte 数据，然后再接收从机发送的数据，数据接收后，会保存在 aRxBuffer 数组中，接收数据为 0x1~0xf。

8.6 I2C_TwoBoard_CommunicationSlave_DMA_Init

此样例演示了主机 I2C 通过 DMA 方式进行通讯，从机使用 PY32F030x，按下 user 按键，主机先向从机发送 15 byte 数据，然后再接收从机发送的数据，数据接收后，会保存在 aRxBuffer 数组中，接收数据为 0x1~0xf。

8.7 I2C_TwoBoard_CommunicationSlave_IT_Init

此样例演示了主机 I2C 通过中断方式进行通讯，从机使用 PY32F030x，按下 user 按键，主机先向从机发送 15 byte 数据，然后再接收从机发送的数据，数据接收后，会保存在 aRxBuffer 数组中，接收数据为 0x1~0xf。

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 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）。

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 个数码管同时显示“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。

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。

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。

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。

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，

12.2 PWR_SLEEP_WFE

此样例演示了通过执行 WFE(wait for event)指令进入 sleep 模式，使用 GPIO 事件唤醒

12.3 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

12.4 PWR_STOP_WFE

此样例演示了通过执行 WFE(wait for event)指令进入 stop 模式，使用 GPIO 事件唤醒

12.5 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

13 RCC

13.1 RCC_HSE_DIV

此样例演示了时钟输出功能，可输出 HSE 波形

13.2 RCC_HSI_OUTPUT

此样例演示了时钟输出功能，可输出 HSI 波形

13.3 RCC_LSE_OUTPUT

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。

13.4 RCC_LSI_OUTPUT

此样例演示了时钟输出功能，可输出 LSI 波形

13.5 RCC_PLL_OUTPUT

此样例演示了时钟输出功能，可输出以 HSE 为源的 PLL 波形

13.6 RCC_Sysclock_Switch

此样例演示了时钟切换，由 LSI 切换至 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。

14.2 RTC_WeakUpAlarm_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。

14.3 RTC_WeakUpSecond_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。

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。

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。

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。

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。

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。

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 200ns 的死区和刹车功能。

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。

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。

16.5 TIM1_EncoderTI2&TI1_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。

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。

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。

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。

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。

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。

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。

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。

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。

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。

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。

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。

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。

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。

17.2 USART_HyperTerminal_DMA_Init

此样例演示了通过 DMA 收发数据的功能，复位 MCU 并重新运行，PC 端收到字符串“UART Test”；PC 端发送 12 个字符，MCU 会反馈同样的 12 个字符给 PC 端。

17.3 USART_HyperTerminal_IT_Init

此样例演示了通过 USART 中断收发数据的功能，复位 MCU 并重新运行，PC 端收到字符串：UART Test；PC 端发送 12 个字符，MCU 会反馈同样的 12 个字符给 PC 端。

17.4 USART_HyperTerminal_Polling_Init

此样例演示了通过 USART 轮询收发数据的功能，MCU 复位后会向 PC 端发送"UART Test",PC 端发送 12 个字符，MCU 会反馈同样的 12 个字符给 PC 端。

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 48Hz。

18.2 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。

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。

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。