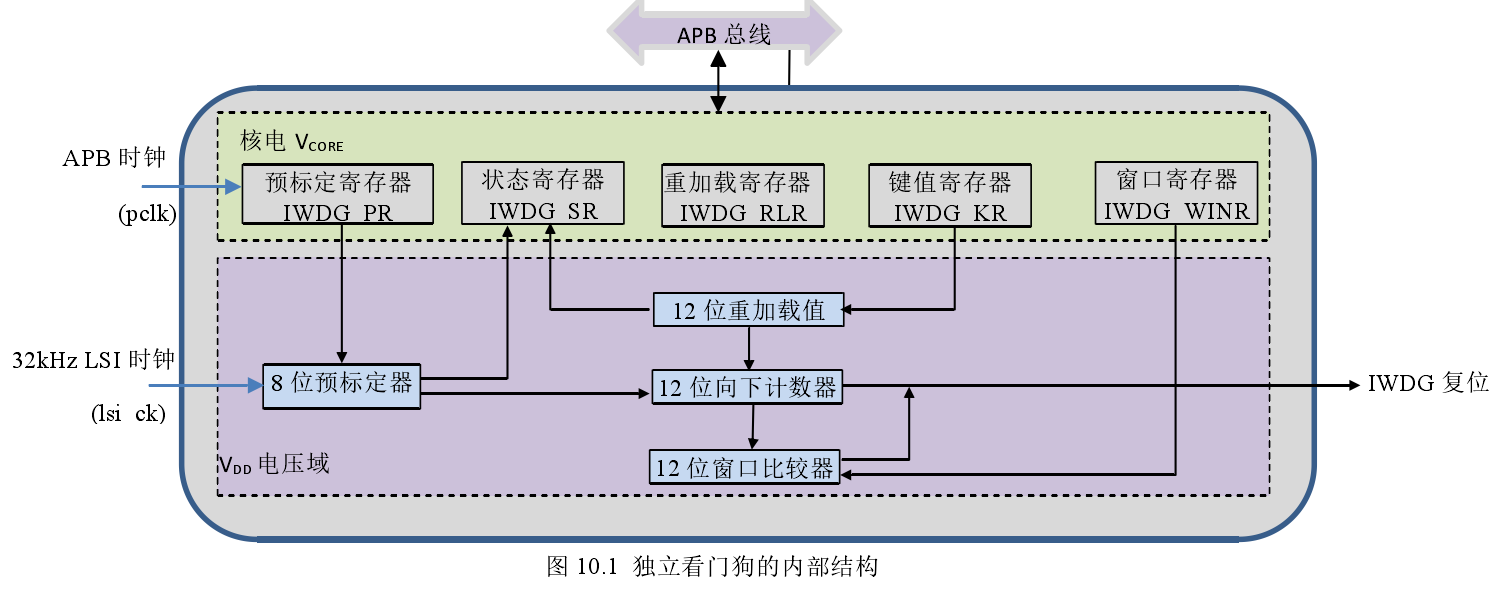
**独立看门狗的使用——底层（LL）驱动**

**第一步：**看门狗相关知识

看门狗：独立看门狗外设检测并解决由于软件失效引起的故障，并在计数器到达给定的超时值时触发系统复位。其结构如下图所示：

****

在IWDG中需要两个时钟资源，包括：

（1）APB总线中的时钟pclk，该时钟用于提供APB总线访问IWDG中寄存器的时钟信号。

（2）LSI时钟用于IWDG中功能部分。

在IWDG功能区中的8位预分频器用于对LSI振荡器时钟进行分频。12位递减（向下）计数器定义了超时值。当递减计数器中的值达到零时，将产生看门狗复位信号。

**第二步：**创建工程

①GPIO相关配置：

PA5“输出模式”、标号User Lable设置为“LED4”

PC13“下降沿触发中断模式” 、上拉、标号User Lable设置为“USER\_BUTTON”

②NVIC相关配置：

使能4-15号外部中断

③IWDG相关配置：

MODE设置为勾选“Actived”、IWDG window value设置为4095（保持默认）、IWDG down-counter reload value设置为“4078”

④RCC相关设置

Generate the peripherals clock configuration设置为“FALSE”

⑤在“Project Manager”选项卡下“Advanced Settings”页面，将默认选项“HAL”驱动改为“LL”驱动，在系统时钟配置“SystemClock\_Config”后的可视化选项“Visibility”打√

**第三步：**生成代码

在Project页面，添加工程名、选择目录、应用结构改为基础、工具改为MDK-ARM。

配置完成后点击“GENERATE CODE”生成代码。

**第四步：**编写代码

【注】在变量定义前加关键字“volatile”，它是一种类型修饰符，用它声明的类型变量表示可以被某些编译器未知的因素更改，比如：操作系统、硬件或者其它线程等。

①定义外部复位按键标志位（0即没有按下按键，当前无复位请求）

static volatile uint8\_t ubKeyPressed = 0;

②声明函数：检测独立看门狗复位函数与点亮LD4灯函数

void Check\_IWDG\_Reset(void);

void LED\_On(void);

③编写其子函数

void Check\_IWDG\_Reset(void)

{

if (LL\_RCC\_IsActiveFlag\_IWDGRST()) //如果RCC时钟计数到了预设的复位值

{

LL\_RCC\_ClearResetFlags(); //清除RCC复位标志

LED\_On(); //让LD4灯点亮

while(ubKeyPressed != 1) //外部复位按键标志位是0即没有复位请求则LD4常亮

{

}

ubKeyPressed = 0; //清空外部复位按键标志位

}

}

void LED\_On(void)

{

/\* Turn LED4 on \*/

LL\_GPIO\_SetOutputPin(LED4\_GPIO\_Port, LED4\_Pin);

}

④在主函数下声明并编写13号中断的回调函数

void UserButton\_Callback(void) //中断回调函数，生成外部中断复位请求

{

ubKeyPressed = 1;

}

⑤在stm32g0xx\_it.c文件下的中断函数EXTI4\_15\_IRQHandler下调用中断的回调函数

UserButton\_Callback();

⑥主函数编写

APB总线时钟使能、系统时钟初始化、GPIO初始化、看门狗初始化（保持默认）

在while(1)循环之前检测独立看门狗复位

Check\_IWDG\_Reset();

⑦在while(1)循环内添加如下代码

if (1 != ubKeyPressed) //没有外部复位请求则→

{

LL\_IWDG\_ReloadCounter(IWDG); //重新加载看门狗向下计数器（更新）

LL\_GPIO\_TogglePin(LED4\_GPIO\_Port, LED4\_Pin); //LD4状态反转——闪烁

LL\_mDelay(200); //延时

}

**第五步：**编译运行

编译下载之后应观察到如下现象：

①按下复位按键后LD4灯闪烁。

②第一次按下“USER”按键后LD4保持常亮。

③第二次按下“USER”按键后LD4恢复闪烁。

**第六步：**程序运行过程分析

①按下复位按键后：

先执行启动引导代码，然后跳转到main.c文件中的main主程序。进入main之后执行一些列初始化配置：APB总线上相关设备的时钟使能、系统时钟初始化、GPIO初始化、看门狗初始化。进而调用Check\_IWDG\_Reset检测独立看门狗复位，不满足锁死条件。之后进入while(1)循环，此时还是没有外部复位请求，正常喂狗，LD4闪烁。

②第一次按下“USER”按键后：

触发13号外部中断，执行其中断回调函数，将外部复位请求标志位置一。执行完中断程序后，跳转回主函数的while(1)循环之中。不满足ubKeyPressed！=1的条件，不会重新加载看门狗向下计数器，即不会喂狗。但是还在while(1)循环内，看门狗还在计数，当看门狗溢出后复位**【与按下复位按键不同】**，重新执行main主函数。但是现在ubKeyPressed被置为一，当执行到Check\_IWDG\_Reset检测独立看门狗复位时，将LD4灯点亮，并卡在while(ubKeyPressed != 1)循环中，故此时LD4常亮。

③第二次按下“USER”按键后：

触发13号外部中断，执行其中断回调函数，将外部复位请求标志位置一。执行完中断程序后，跳转回子函数Check\_IWDG\_Reset的while(ubKeyPressed != 1)循环之中，不满足循环条件，跳出此循环。之后在此子函数最后，将ubKeyPressed标志位清零。返回主函数，又进入while(1)循环，正常喂狗。

（如此反复，无穷无尽）……