AN1038
应用笔记

PY32F030_003_002A 的时钟安全系统使用注意事项

前言

PY32F030_003_002A 微控制器带有时钟安全系统。时钟安全系统可以被软件激活。在这种情况下，HSE 的启动延迟后，时钟检测功能被打开。当这个 HSE 被关闭后，时钟检测功能被关闭。如果在 HSE 上发现时钟 failure，HSE 会被自动关闭，时钟 failure 事件被送给 TIM1 (高级 Timer)和 TIM16/TIM17 (通用 timer)的刹车输入端，并产生中断通知软件该 failure（Clock Security System Interrupt CSSI），进而允许 MCU 进行拯救操作。CSSI 被链接到 Cortex-M0+的 NMI（Non-maskable interrupt）exception 向量。

本应用笔记将帮助用户了解 PY32F030_003_002A 的时钟安全系统使用注意事项并快速着手开发。

表 1. 适用产品

类型	产品系列
微型控制器系列	PY32F030、PY32F003、PY32F002A

目录

1 时钟安全系统

1.1 注意事项

- 时钟安全系统在触发后，需要再次使能；
- 一旦 CSS 被使能，并且让一个 HSE 时钟 failure，就会产生 CSS 中断，并自动产生一个 NMI。该 NMI 将不断执行，直到 CSS 中断挂起位被清除。因此，在 NMI 的处理程序中必须通过设置时钟中断寄存器（RCC_CICR）里的 CSSC 为来清除 CSS 中断；
- 如果 HSE 被直接或者间接的用作系统时钟，时钟 failure 将导致系统自动切换到 HSI,同时关闭 HSE；
- 使用定时器模块是为了检测在规定时间内是否还有干扰，如果有则重新计数，如果没有就将系统时钟切回 HSE.

1.2 操作流程

- 配置系统时钟为 HSE；
- 初始化定时器模块；
- 开启时钟安全系统。
- 在 NMI 中断中清除计数，并且清除时钟安全系统的中断标志

1.3 代码示例

```
  static void APP_SystemClockConfig(void)
  {
    RCC_OscInitTypeDef RCC_OscInitStruct = {0};
    RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};

    /*配置时钟源 HSE/HSI/LSE/LSI*/
    RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSE |
RCC_OSCILLATORTYPE_HSI | RCC_OSCILLATORTYPE_LSI | RCC_OSCILLATORTYPE_LSE;
    RCC_OscInitStruct.HSIState = RCC_HSI_ON;
/* 开启 HSI */
    RCC_OscInitStruct.HSICalibrationValue = RCC_HSICALIBRATION_8MHz;
/* 配置 HSI 输出时钟为 4MHz */
    RCC_OscInitStruct.HSIDiv = RCC_HSI_DIV1;
/* HSI 不分频 */
    RCC_OscInitStruct.HSEState = RCC_HSE_ON;
/* 关闭 HSE */
    RCC_OscInitStruct.HSEFreq   = RCC_HSE_8_16MHz;
    RCC_OscInitStruct.LSIState = RCC_LSI_OFF;
/* 关闭 LSI */
```

```
    RCC_OscInitStruct.LSEState = RCC_LSE_OFF;
/* 关闭 LSE */
    RCC_OscInitStruct.PLL.PLLState = RCC_PLL_OFF;
/* 关闭 PLL */

    if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
/* RCC 振荡器初始化 */
    {
      APP_ErrorHandler();
    }

    /* 初始化 CPU,AHB,APB 总线时钟 */
    RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK | RCC_CLOCKTYPE_SYSCLK |
RCC_CLOCKTYPE_PCLK1; /* RCC 系统时钟类型 */
    RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_HSE;
/* SYSCLK 的源选择为 HSI */
    RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
/* APH 时钟不分频 */
    RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV1;
/* APB 时钟不分频 */

    if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_1) != HAL_OK)
/* 初始化 RCC 系统时钟 */
    {
      APP_ErrorHandler();
    }
  }

  void Timer_init(void)
  {
    TimHandle.Instance = TIM1;                                    /* 选择 TIM1 */
    TimHandle.Init.Period            = 8000 - 1;                  /* 自动重装载值 */
    TimHandle.Init.Prescaler         = 1000 - 1;                  /* 预分频为 1000-1 */
    TimHandle.Init.ClockDivision     = TIM_CLOCKDIVISION_DIV1;    /* 时钟不分频 */
    TimHandle.Init.CounterMode       = TIM_COUNTERMODE_UP;        /* 向上计数 */
    TimHandle.Init.RepetitionCounter = 1 - 1;                     /* 不重复计数 */
```

```
    TimHandle.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;    /* 自动
重装载寄存器没有缓冲 */
    /* TIM1 初始化 */
    if (HAL_TIM_Base_Init(&TimHandle) != HAL_OK)
    {
      APP_ErrorHandler();
    }

    /* TIM1 使能启动，并使能中断 */
    if (HAL_TIM_Base_Start_IT(&TimHandle) != HAL_OK)
    {
      APP_ErrorHandler();
    }

}
/* 开启时钟安全系统 */
   SET_BIT(RCC->CR,RCC_CR_CSSON);
void NMI_Handler(void)
{
     /* 清除计数器 */
       CSS_CNT = 0;
     /*清时钟安全系统中断标志*/
       SET_BIT(RCC->CICR,RCC_CICR_CSSC);
}
```

2 版本历史

版本	日期	更新记录
V0.1	2023.09.04	初版

Puya Semiconductor Co., Ltd.