STM32Cube_FW_F4 中 RTC_Calendar 例程的 BUG

前言

实时时钟（RTC）是一个独立的 BCD 定时器/计数器，用来提供准确的日历和时间信息。准确性是其重要的指标。

问题

某客户在其产品的设计中，使用了 STM32F429IIT6。客户在使用过程发现一个问题，虽然已经有使用电池对 VBAT 进行供电，但是在经常频繁的 VDD 上下电之后，发现时钟会比准确的时间慢几秒钟。

调研

1. 了解问题

向客户了解其使用的固件库，得知他的程序是参考 STM32Cube_FW_F4_V1.3.0\Projects\STM324x9I_EVAL\Examples\RTC 中的 RTC_Calendar 例程。于是找来 STM32439I-EVAL2 来进行验证，测试发现，STM32Cube_FW_F4_V1.3.0\Projects\STM324x9I_EVAL\Examples\RTC 中的 RTC_Calendar 例程确实存在频繁上下电会导致时间变慢的情况；而对标准外设库 STM32F4xx_DSP_StdPeriph_Lib_V1.4.0\Project\STM32F4xx_StdPeriph_Examples\RTC 中的 RTC_Calendar 例程进行测试，则不存在此问题。所以，怀疑 STM32Cube_FW_F4_V1.3.0\Projects\STM324x9I_EVAL\Examples\RTC 中的 RTC_Calendar 例程存在 Bug。

2. 问题分析

仔细阅读 STM32Cube_FW_F4_V1.3.0\Projects\STM324x9I_EVAL\Examples\RTC 中的 RTC_Calendar 例程，分析一下 main.c 主程序，“if(HAL_RTCEx_BKUPRead(&RtcHandle, RTC_BKP_DR0) != 0x32F2)”是用来判断 RTC 是否是已经被配置过的，所以怀疑的重点可放在这之前的“if(HAL_RTC_Init(&RtcHandle) != HAL_OK)”中的 HAL_RTC_Init()函数。

进入位于 stm32f4xx_hal_rtc.c 中的 HAL_RTC_Init()函数，再进入其调用的位于 stm32f4xx_hal_msp.c 中的 HAL_RTC_MspInit()函数，在这个函数中，可以看到以下代码：

```
  /*##-1- Configue LSE as RTC clock soucre ###################################*/
  RCC_OscInitStruct.OscillatorType =  RCC_OSCILLATORTYPE_LSI | RCC_OSCILLATORTYPE_LSE;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_NONE;
  RCC_OscInitStruct.LSEState = RCC_LSE_ON;
  RCC_OscInitStruct.LSIState = RCC_LSI_OFF;
  if(HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }
```

```
  PeriphClkInitStruct.PeriphClockSelection = RCC_PERIPHCLK_RTC;
  PeriphClkInitStruct.RTCClockSelection = RCC_RTCCLKSOURCE_LSE;
  if(HAL_RCCEx_PeriphCLKConfig(&PeriphClkInitStruct) != HAL_OK)
  {
    Error_Handler();
  }
```

这段代码中由于选中了 LSE，在其所调用的 HAL_RCC_OscConfig()中对 LSE 进行了重新配置。在 stm32f4xx_hal_rcc.c 中找到 HAL_RCC_OscConfig()函数，发现其对 LSE 重新配置的时候，对 LSE 进行关闭，然后再配置。

到此，回来再看 main.c，由于“if(HAL_RTC_Init(&RtcHandle) != HAL_OK)”位于“if(HAL_RTCEx_BKUPRead(&RtcHandle, RTC_BKP_DR0) != 0x32F2)”之前，从程序流程来看，每次 VDD 上电，都会进行一次 HAL_RTC_Init()，也就是说，每次上电都会有一个关闭 LSE 再打开的动作，这个动作多了，时间变慢的现象就变得很明显了。

3. 问题解决

问题原因很明显了，那么解决办法也很简单，只需要将 HAL_RTC_Init()这个初始化函数挪到判断 RTC 是否是已经被配置过的 if…else…语句里边就行了。如果是 RTC 已经被配置过的，就不需要再重新初始化一次了。如下：

```
  /*##-1- Configure the RTC peripheral #######################################*/
  RtcHandle.Instance = RTC;

  /*##-2- Check if Data stored in BackUp register0: No Need to reconfigure RTC#*/
  /* Read the BackUp Register 0 Data */
  if(HAL_RTCEx_BKUPRead(&RtcHandle, RTC_BKP_DR0) != 0x32F2)
  {
    /* Configure RTC prescaler and RTC data registers */
    /* RTC configured as follow:
        - Hour Format    = Format 24
        - Asynch Prediv  = Value according to source clock
        - Synch Prediv   = Value according to source clock
        - OutPut         = Output Disable
        - OutPutPolarity = High Polarity
        - OutPutType     = Open Drain */
    RtcHandle.Init.HourFormat = RTC_HOURFORMAT_24;
    RtcHandle.Init.AsynchPrediv = RTC_ASYNCH_PREDIV;
    RtcHandle.Init.SynchPrediv = RTC_SYNCH_PREDIV;
    RtcHandle.Init.OutPut = RTC_OUTPUT_DISABLE;
    RtcHandle.Init.OutPutPolarity = RTC_OUTPUT_POLARITY_HIGH;
    RtcHandle.Init.OutPutType = RTC_OUTPUT_TYPE_OPENDRAIN;

    if(HAL_RTC_Init(&RtcHandle) != HAL_OK)
    {
      /* Initialization Error */
      Error_Handler();
    }

    /* Configure RTC Calendar */
```

```
    RTC_CalendarConfig();
  }
  else
  {
    /* Check if the Power On Reset flag is set */
    if(__HAL_RCC_GET_FLAG(RCC_FLAG_PORRST) != RESET)
    {
      /* Power on reset occured: Turn LED2 on */
      BSP_LED_On(LED2);
    }
    /* Check if Pin Reset flag is set */
    if(__HAL_RCC_GET_FLAG(RCC_FLAG_PINRST) != RESET)
    {
      /* External reset occured: Turn LED4 on */
      BSP_LED_On(LED4);
    }

    /* Enable the PWR clock */
    __PWR_CLK_ENABLE();

    /* Allow access to RTC */
    HAL_PWR_EnableBkUpAccess();

    /* Wait for RTC APB registers synchronisation */
    if(HAL_RTC_WaitForSynchro(&RtcHandle) != HAL_OK)
    {
      /* synchronisation Error */
      Error_Handler();
    }

    /* Clear the RTC Alarm Flag */
    __HAL_RTC_ALARM_CLEAR_FLAG(&RtcHandle,RTC_FLAG_ALRAF);

    /* Clear the EXTI Line 17 Pending bit (Connected internally to RTC Alarm) */
    __HAL_RTC_EXTI_CLEAR_FLAG(RTC_EXTI_LINE_ALARM_EVENT);

    /* Clear Reset Flag */
    __HAL_RCC_CLEAR_RESET_FLAGS();
  }
```

结论

由于 STM32Cube_FW_F4_V1.3.0\Projects\STM324x9I_EVAL\Examples\RTC 中的 RTC_Calendar 例程没有注意到 HAL_RTC_Init()函数里边会有关闭 LSE 的动作，而每次上电都会运行这个函数，每次上电都会导致时间变慢，上电的次数多了，变慢就很明显了。所以，例程上是有 Bug 的，需要进行修复。

处理

需要将 HAL_RTC_Init()这个初始化函数的位置做个修改。如果 RTC 未被配置过，则进行配置；如果是已经被配置过的，就不需要再重新初始化一次了。标准外设库 STM32F4xx_DSP_StdPeriph_Lib_V1.4.0\Project\STM32F4xx_StdPeriph_Examples\RTC 中的 RTC_Calendar 例程是没有问题的，参考此例程，修改得一基于 STM32Cube_FW_F4 的 RTC_Calendar 例程，见附件。

建议

上电时对 LSE 进行重新初始化可能会导致 RTC 计时不准确，所以在实际应用过程中应该对此注意一下。