FAQ0073

常见问题解答

RTC_毫秒计时

Questions： 如何在 AT32F403A, AT32F407, AT32F403, AT32F413, AT32F415, AT32F421 获得毫秒计时？

Answer：

Example Code 请见雅特力官网示例代码《SC0007_AT32F4xx_RTC_毫秒计时》。

因 AT32F403A, AT32F407, AT32F403, AT32F413 的 RTC 硬件结构相同，请参考项目工程 AT32F403A_407_Firmware_Library。用 AT-START-F403 V1.2 下载验证时，请将 R7 焊上 0 欧电阻。

因 AT32F415, AT32F421 的 ERTC 硬件结构相同，请参考项目工程 AT32F415_Firmware_Library。

AT32F403A RTC 设置：

```
typedef struct
{
  __IO uint8_t  hour;        /*!< specifies the hours of calendar.        */
  __IO uint8_t  min;         /*!< specifies the minutes of calendar.      */
  __IO uint8_t  sec;         /*!< specifies the second of calendar.       */
  __IO uint16_t msec;        /*!< specifies the millisecond of calendar. */
} calendar_type;
calendar_type time_struct;
/* config calendar */
time_struct.hour   = 12;
time_struct.min    = 0;
time_struct.sec    = 0;
time_struct.msec   = 100;
rtc_init(&time_struct);

uint8_t rtc_init(calendar_type *calendar)
{
  /* enable pwc and bpr clocks */
  crm_periph_clock_enable(CRM_PWC_PERIPH_CLOCK, TRUE);
  crm_periph_clock_enable(CRM_BPR_PERIPH_CLOCK, TRUE);

  /* enable the battery-powered domain write operations */
  pwc_battery_powered_domain_access(TRUE);

  /* reset battery-powered domain register */
  bpr_reset();

  /* enable the lext osc */
  crm_clock_source_enable(CRM_CLOCK_SOURCE_LEXT, TRUE);
  /* wait lext is ready */
  while(crm_flag_get(CRM_LEXT_STABLE_FLAG) == RESET);
```

```
  /* select the rtc clock source */
  crm_rtc_clock_select(CRM_RTC_CLOCK_LEXT);

  /* enable rtc clock */
  crm_rtc_clock_enable(TRUE);

  /* wait for rtc registers update */
  rtc_wait_update_finish();

  /* wait for the register write to complete */
  rtc_wait_config_finish();

  /* set rtc divider: set rtc period to 1sec */
  rtc_divider_set(32);

  /* wait for the register write to complete */
  rtc_wait_config_finish();

  /* set time */
  rtc_time_set(calendar);

  /* writes data to bpr register */
  bpr_data_write(BPR_DATA1, 0x1234);

  return 1;
}
```

AT32F415 ERTC 设置：

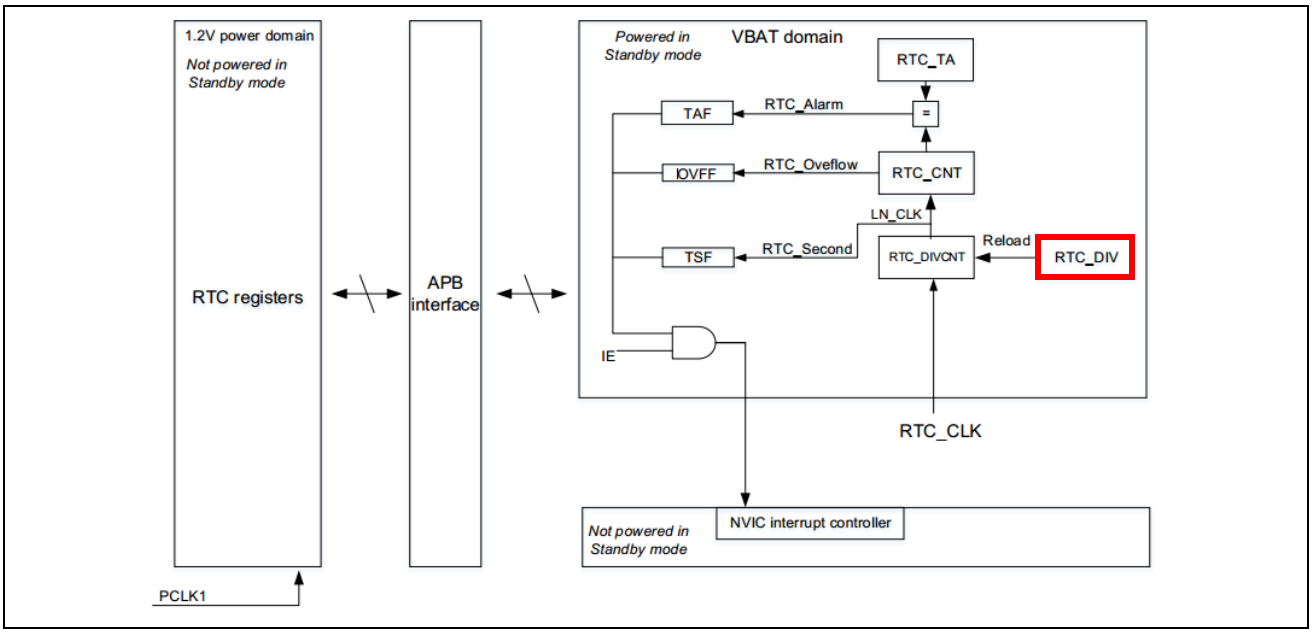
```
void ertc_config(void)
{
  /* enable the pwc clock interface */
  crm_periph_clock_enable(CRM_PWC_PERIPH_CLOCK, TRUE);

  /* allow access to ertc */
  pwc_battery_powered_domain_access(TRUE);

  /* reset ertc domain */
  crm_battery_powered_domain_reset(TRUE);
  crm_battery_powered_domain_reset(FALSE);

#if defined (ERTC_CLOCK_SOURCE_LICK)
  /* enable the lick osc */
  crm_clock_source_enable(CRM_CLOCK_SOURCE_LICK, TRUE);

  /* wait till lick is ready */
  while(crm_flag_get(CRM_LICK_STABLE_FLAG) == RESET)
  {
  }
```

```
  /* select the ertc clock source */
  crm_ertc_clock_select(CRM_ERTC_CLOCK_LICK);

  /* ertc second(1hz) = ertc_clk(lick) / (ertc_clk_div_a + 1) * (ertc_clk_div_b + 1) */
  ertc_clk_div_b = 999;
  ertc_clk_div_a = 32;
#elif defined (ERTC_CLOCK_SOURCE_LEXT)
  /* enable the lext osc */
  crm_clock_source_enable(CRM_CLOCK_SOURCE_LEXT, TRUE);

  /* wait till lext is ready */
  while(crm_flag_get(CRM_LEXT_STABLE_FLAG) == RESET)
  {
  }

  /* select the ertc clock source */
  crm_ertc_clock_select(CRM_ERTC_CLOCK_LEXT);

  /* ertc second(1hz) = ertc_clk / (ertc_clk_div_a + 1) * (ertc_clk_div_b + 1) */
  ertc_clk_div_b = 999;
  ertc_clk_div_a = 32;
#endif

  /* enable the ertc clock */
  crm_ertc_clock_enable(TRUE);

  /* deinitializes the ertc registers */
  ertc_reset();

  /* wait for ertc registers update */
  ertc_wait_update();

  /* configure the ertc divider */
  ertc_divider_set(ertc_clk_div_a, ertc_clk_div_b);

  /* configure the ertc hour mode */
  ertc_hour_mode_set(ERTC_HOUR_MODE_24);

  /* set date: 2021-05-01 */
  ertc_date_set(21, 5, 1, 5);

  /* set time: 12:00:00 */
  ertc_time_set(12, 0, 0, ERTC_AM);

  /* indicator for the ertc configuration */
  ertc_bpr_data_write(ERTC_DT1, 0x1234);
}
```

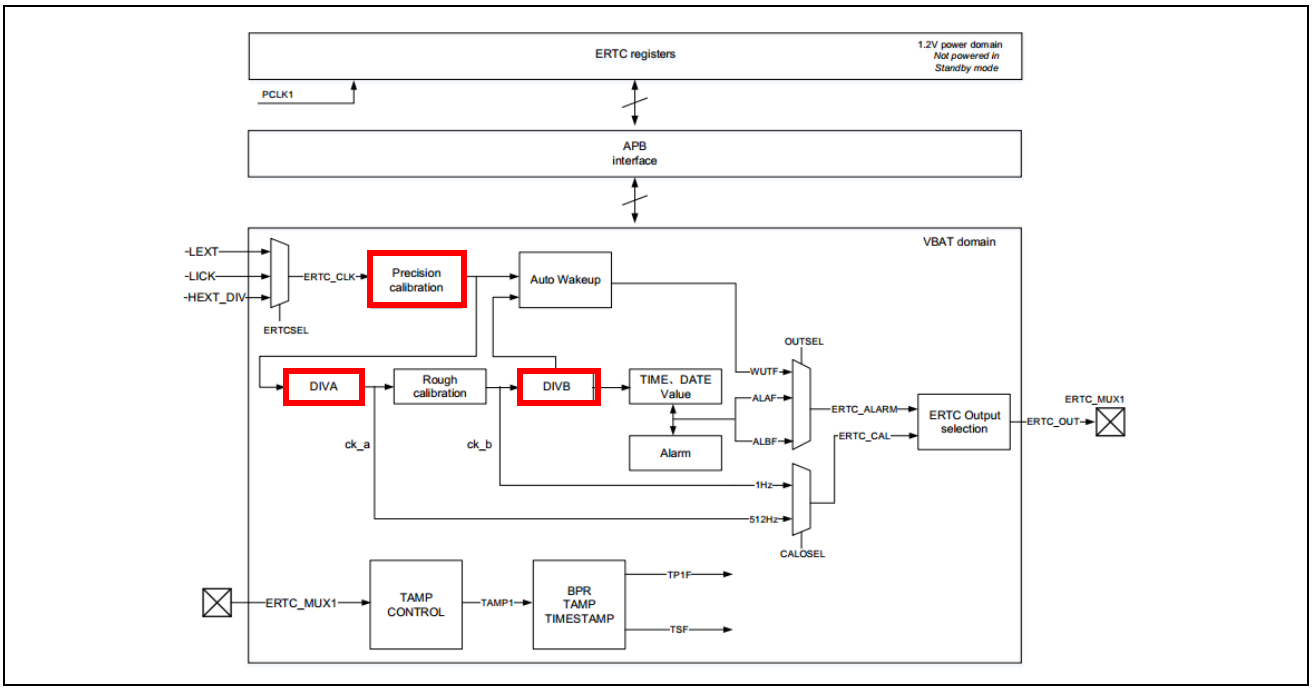
需注意以下事项：

1. 修改 RTC 设置时需修改写入备份域寄存器的值(如下代码)，或者是断主电源的同时拨掉纽扣电池；
bpr_data_write(BPR_DATA1, 0x1234);
2. 程序中 RTC 分频系数寄存器（RTC_DIVH/RTC_DIVL）设置时要实际计算值减 1，因为寄存器会自动加 1，时钟频率为 = f_{RTCCLK}/（DIV[19：0]+1）；ERTC 预分频器寄存器(ERTC_DIV) 设置时 DIVA/DIVB 要实际计算值减 1, 因为寄存器会自动加 1，日历时钟=ERTC_CLK/((DIVA+1)x(DIVB+1))。
3. AT32F403A 与 AT32F415 获取毫秒的差别在于 AT32F403A 是通过分频，获得 1ms 时基，如下 AT32F403A RTC 框图（最小可获得时基 61us）； AT32F415 是在 1s 时基，通过亚秒寄存器（ERTC_SBS）获得同步分频值，如下 AT32F415 ERTC 框图，计算得到毫秒值 SBS = (DIVB - SBS) / (DIVB + 1)。

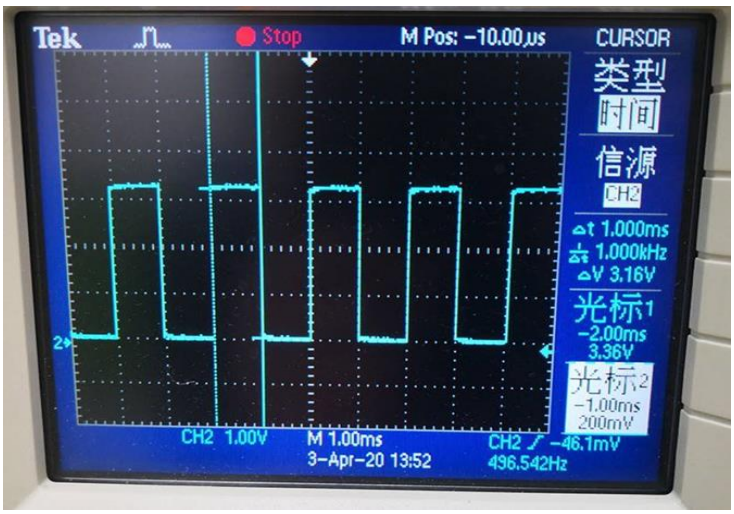
AT32F403A RTC 框图

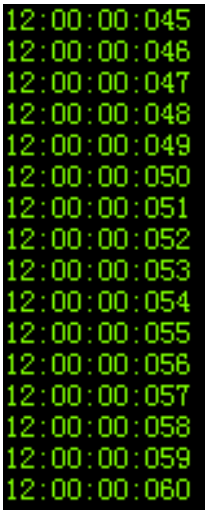


AT32F415 ERTC 框图



4. 实际示波器测试和打印结果





类型： MCU 应用

适用型号： AT32F403A, AT32F407, AT32F403, AT32F413, AT32F415, AT32F421

主功能： RTC, ERTC

次功能： 无

文档版本历史

日期	版本	变更
2022.2.18	2.0.0	最初版本

重要通知 - 请仔细阅读

买方自行负责对本文所述雅特力产品和服务的选择和使用，雅特力概不承担与选择或使用本文所述雅特力产品和服务相关的任何责任。

无论之前是否有过任何形式的表示，本文档不以任何方式对任何知识产权进行任何明示或默示的授权或许可。如果本文档任何部分涉及任何第三方产品或服务，不应被视为雅特力授权使用此类第三方产品或服务，或许可其中的任何知识产权，或者被视为涉及以任何方式使用任何此类第三方产品或服务或其中任何知识产权的保证。

除非在雅特力的销售条款中另有说明，否则，雅特力对雅特力产品的使用和/或销售不做任何明示或默示的保证，包括但不限于有关适销性、适合特定用途(及其依据任何司法管辖区的法律的对应情况)，或侵犯任何专利、版权或其他知识产权的默示保证。

雅特力产品并非设计或专门用于下列用途的产品：(A) 对安全性有特别要求的应用，如：生命支持、主动植入设备或对产品功能安全有要求的系统；(B) 航空应用；(C) 汽车应用或汽车环境；(D) 航天应用或航天环境，且/或(E) 武器。因雅特力产品不是为前述应用设计的，而采购商擅自将其用于前述应用，即使采购商向雅特力发出了书面通知，风险由购买者单独承担，并且独力负责在此类相关使用中满足所有法律和法规要求。

经销的雅特力产品如有不同于本文档中提出的声明和/或技术特点的规定，将立即导致雅特力针对本文所述雅特力产品或服务授予的任何保证失效，并且不应以任何形式造成或扩大雅特力的任何责任。