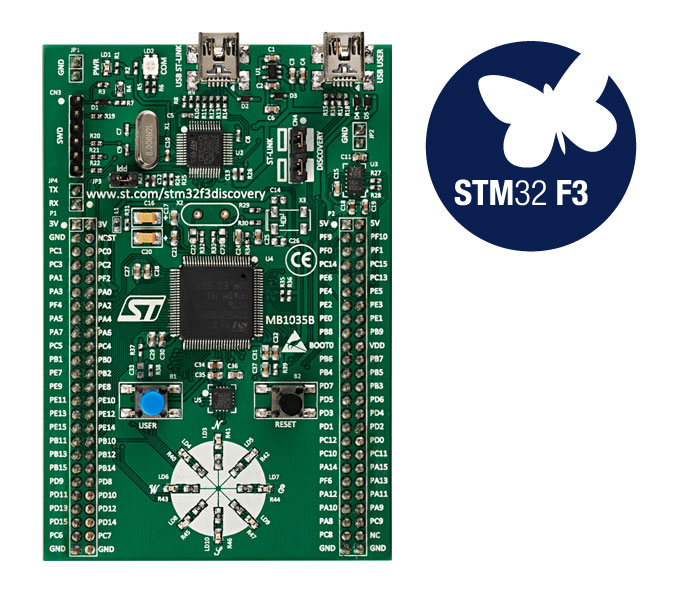
Основы работы с RTC в STM32F3



# Основные сведенье о часах реального времени

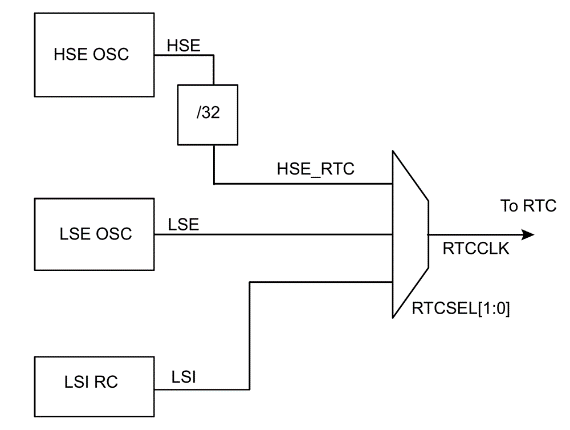
Часы реального времени (real time clock, RTC) предназначены для отсчета текущего времени. Как и другие таймеры, они имеют счетчик, но в отличии от обычных таймеров:

1. обладают более высокой точностью;
2. могут питаться от резервного источника (например, CR2032), в то время как основное питание выключено;
3. имеют крайне малое потребление энергии (около 300 нА);

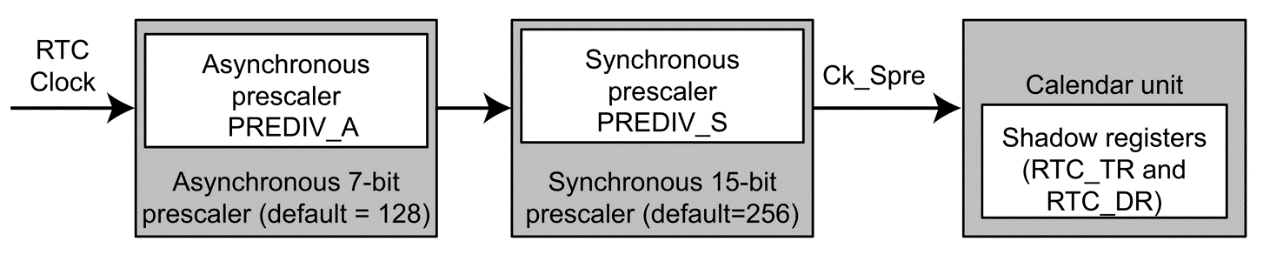
В STM32 они имеют следующие особенности:

1. аппаратно-реализованный календарь с возможностью автоматического перехода на зимнее и летнее время;
2. встроенные будильники;
3. backup регистры для хранения пользовательской информации.

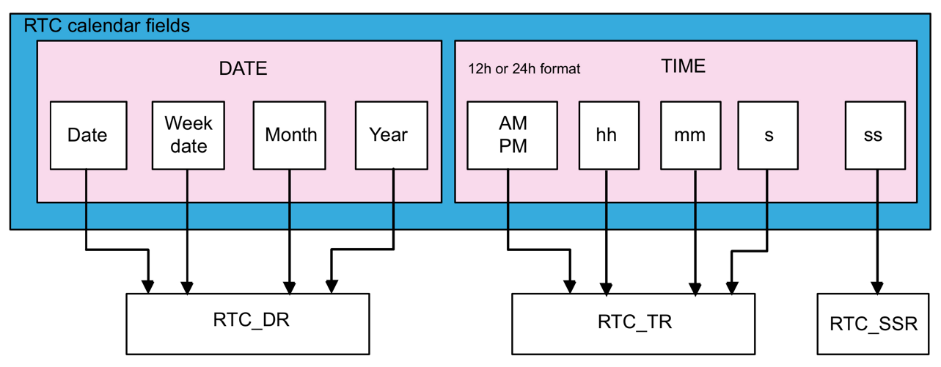
Общая схема работы RTC следующая. Часы подключены к одному из 3-х источников тактового сигнала:



Частота выбранного тактового сигнала уменьшается до 1 Гц с помощью двух определителей:



Полученный сигнал подается на календарь:



который сам заботится об инкременте секунд, минут, часов, дней, месяцев и года.

### Предделители RTC

Тактовая частот, подаваемая на календарь рассчитывается по формуле:

,

где PREDDIV\_A – значение асинхронного предделителя (от 0 до 127),

PREDDIV\_S – значение синхронного предделителя (от 0 до 32767).

Обычно значение PREDDIV\_A выбирают максимальным (127), а значение PREDDIV\_S, чтобы получить  Гц.

### Календарь

Особенностью часов в STM32 является то, что регистры календаря хранят минуты, часы и т.д. в BCD (двоично-десятичный код) формате.

### Wakup таймер

Для организации периодических прерываний с большим периодом, возможно использовать 16-ти битный wakeup таймер, который может работать от , и как обычные таймеры генерировать прерывания с заданным периодом.

### Особенности регистров RTC

В силу отдельного питания RTC, его регистры не сбрасываются при перезагрузке платы. Кроме того, по умолчанию, все регистры RTC защищены от записи, поэтому для снятия защиты нужно выполнить определенные действия, о которых будет сказано ниже.

# Базовая инициализация RTC

Рассмотрим, как выполнить базовую инициализацию RTC, чтобы инициализировать календарь.

1. так как у нас нет внешнего источника тактового сигнала, придется использовать внутренний (LSI), который необходимо каждый раз включать после перезапуска микроконтроллера. Для этого:
   1. включите LSI (регистр RCC->CSR, бит LSION);
   2. дождитесь включения LSI (регистр RCC->CSR, бит LSIRDY).
2. включите интерфейс часов (регистр RCC->APB1ENR, бит PWREN). Это не включит часы сами по себе, но необходимо для работы с ними;
3. разрешите запись в регистры связные с RTC (регистр PWR->CR, бит DBP). Это разрешит писать в регистры RTC и регистр RCC->BDCR. Однако часть регистров имеет дополнительную защиту, которую нужно будет снять отдельно. После этого подождите установку бита DBP, т.к. на это может понадобится немного времени.
4. Проверьте, были ли инициализированы часы до этого (регистр RTC->ISR, бит INITS). Так как часы сохраняют значение регистров, несмотря на перезагрузку микроконтроллера, то можно не инициализировать их каждый раз заново, оставив предыдущие настройки и время. Если часы не инициализированы, то необходимо сделать следующее:
   1. выберете в качестве источника тактового сигнала часов LSI (регистр RCC->BDCR, бит RTCSEL);
   2. подключите выбранный источник к часам (регистр RCC->BDCR, бит RTCEN);
   3. разблокируйте часы, записав в регистр RTC->WPR последовательность 0xCA, 0x53:

RTC->WPR = 0xCA;

RTC->WPR = 0x53;

* 1. включите режим инициализации часов (регистр RTC->ISR, бит INIT);
  2. дождитесь, когда часы перейдут в режим инициализации (регистр RTC->ISR, бит INITF);
  3. установить значения асинхронного и синхронного предделителей (регистр RTC->PRER). Данный регистр содержит значения для обоих предделителей;
  4. установите текущую дату (регистр RTC->DR);
  5. установите текущее время (регистр RTC->TR);
  6. выключите режим инициализации (регистр RTC->ISR, бит INIT). После этого часы начнут идти;
  7. обратно включите защиту от записи. Для этого запишите 0xFF в регистр RTC->WPR:

RTC->WPR = 0xFF;

На этом базовая инициализация часов закончена.

## Получение текущего времени

Для получения текущего времени необходимо:

1. считать значение времени из RTC->TR регистра
2. считать дату из RTC->DR регистра
3. получить отдельные части (год, месяц, день, час, минута, секунда) из считанных значений

Примечание: порядок считывания TR, а потом DR важен из-за внутренней работы RTC по синхронизации своих регистров с регистрами микроконтроллера.

## Настройка wakeup таймера

Как и обычные таймеры, RTC можно использовать для организации периодических прерываний. Но с RTC таймера имеются следующие отличия по сравнению с обычными таймерами:

1. низкое энергопотребление
2. устойчивость к перезагрузкам микроконтроллера. То есть, при выключении микроконтроллера, регистры таймера RTC не сбрасываются, и он продолжает работать (если имеется отдельное питание и осциллятор).

Для работы с wakeup таймером нужно сделать 3 пункта:

1. реализовать обработчик прерывания. Для этого нужно реализовать функцию:

void RTC\_WKUP\_IRQHandler(void)

{

…

}

где выполнить следующие действия:

* 1. оповестить RTC, что прерывание «принято» (очистите бит WUTF, регистра RTC->ISR)
  2. оповестите микроконтроллер, что wakeup прерывание «принято»:

EXTI->PR = 0x00100000;

* 1. выполните логику своей программы. Например, считайте текущее время, и выведите его по UART.

Отметим, что если не очистить биты прерывания (пункты 1 и 2), то после выхода из функции RTC\_WKUP\_IRQHandler, она будет немедленно вызвана снова, что приведет к своеобразному зависанию программы.

1. Настроить период и включить прерывания на стороне RTC:
   1. разблокируйте часы, записав в регистр RTC->WPR последовательность 0xCA, 0x53:

RTC->WPR = 0xCA;

RTC->WPR = 0x53;

* 1. выключите wakeup таймер (регистр RTC->CR, бит WUTE)
  2. дождитесь, когда wakeup таймер выключиться (регистр RTC->ISR, бит WUTWF);
  3. выберите источник тактового сигнала для таймера (регистр RTC->CR, биты WUCKSEL). Можно выбрать ck\_spre (код 10x), чтобы счетчик таймер инкрементировался с частотой 1 Гц.
  4. установите период таймера (регистр RTC->WUTR)
  5. сбросьте флаг прерываний wakeup таймера (регистр RTC->ISR, бит WUTF)
  6. включите wakeup прерывания (регистр RTC->CR, бит WUTIE)
  7. включите wakeup таймер (регистр RTC->CR, бит WUTE)
  8. обратно включите защиту от записи. Для этого запишите 0xFF в регистр RTC->WPR:

RTC->WPR = 0xFF;

Отметим, что, как и в случае инициализации RTC, данные пункты не обязательно выполнять каждый раз после перезагрузки микроконтроллера.

1. Включить прерывания на стороне микроконтроллера. Так как прерывания это довольно большая, тема, то сразу приведем код для RTC wakeup таймера:

// 1. Enable RTC wakeup interrupt in the NVIC (reg 0, bit 3)

NVIC->ISER[0] |= 0x00000008;

// 2. Allow interrupt in the EXTI (bit 20)

const uint32\_t MASK\_BIT\_20 = 1 << 20;

EXTI->IMR |= MASK\_BIT\_20;

// 3. Set rising edges detection

EXTI->RTSR |= MASK\_BIT\_20

На этом настройка wakeup таймера закончена.