Работа с SysTick без прерывания

Мы сделали функцию задержки с помощью прерываний, однако такая реализация нам не подходит, так как нам потребуется функция `delay()` для работы с датчиком DS18B20, где необходимо создавать задержки менее одной миллисекунды. Конечно, можно сделать так, чтобы прерывания происходили чаще, но такой режим вряд ли можно назвать оптимальным. По этой причине, придётся переписать данную функцию.

В отличие от прерывания, событие — это... событие (аппаратное или программное), на которое может реагировать как ядро, так и периферийные блоки микроконтроллера. Ещё проще говоря, когда происходит какое-то внутреннее изменение в некотором блоке, он выставляет `1` (может и наоборот, `0`), в одном из битов своего регистра. Такой бит ещё называют флагом (англ. flag). Он может служить триггером для другой внутренней цепочки, например для контроллера DMA, который начнёт копировать данные из заданного участка памяти, а может ничего не делать. Так или иначе, значение флага можно считать.

Если внимательно изучить документацию, то можно найти такой флаг — `COUNTFLAG`, на его изменение мы и будем реагировать в функции задержки [1].

> Сбрасывать данный необходимо вручную, иначе воспользоваться SysTick хотя бы еще один раз не получится.

Перепишем функцию задержки, согласно следующему описанию:

1. отключить прерывание от SysTick;
2. установить источник тактирования;
3. задать значение перезагрузки;
4. сбросить текущее значение;
5. включить таймер;
6. ожидать появления `1` в бите, отвечающем за `COUNTFLAG`;
7. сбросить `COUNTFLAG`;
8. отключить SysTick.

Функция приведена ниже (разместим её в `utils.c` и добавим прототип функции в `utils.h`):

```
#define SYSTICK_MAX_VALUE 16777215
// ...
void delay(const uint32_t time) {
    if (time > SYSTICK_MAX_VALUE || time == 0)
        return;
    SysTick->CTRL &= ~SysTick_CTRL_TICKINT_Msk;
    SysTick->CTRL |= SysTick_CTRL_CLKSOURCE_Msk;
    SysTick->LOAD = (SystemCoreClock / 1000000 * time);
    SysTick->VAL = 0;
    SysTick->CTRL |= SysTick_CTRL_ENABLE_Msk;
    while (!(SysTick->CTRL & SysTick_CTRL_COUNTFLAG_Msk));
    SysTick->CTRL &= ~SysTick_CTRL_COUNTFLAG_Msk;
    SysTick->CTRL &= ~SysTick_CTRL_ENABLE_Msk;
```

```
}
```

В макросе `SYSTICK_MAX_VALUE` хранится максимальное значение таймера. Попробуйте, основываясь на данном коде, самостоятельно доработать функцию, добавив в аргументы свой тип `enum` для разрядности задержки (миллисекунды, микросекунды). Полную версию кода можно найти на github: **CMSIS**.

> При частоте частоте 64 МГц на SysTick не получится реализовать задержку в 1 секунду. Попробуйте самостоятельно посчитать максимальное время паузы (обратите внимание на разрядности регистров).

Назад | **Оглавление** | **Дальше**

---

1. Такое постоянное *опрашивание* регистра на английском называется polling. ↩