Определение текущей тактовой частоты при отладке

<http://easyelectronics.ru/opredelenie-tekushhej-taktovoj-chastoty-pri-otladke.html>

Тактовая частота это важнейший параметр, точное знание которого необходимо при расчете многих значений. Начиная от банальных выдержек, до генерации UART передачи. 

И если даже на AVR, где тактовые режимы можно по пальцам одной руки пересчитать, и где все изначально предсказуемо работает по дефолту, прозрачно и явно, у многих новичков были проблемы с пониманием того, в каком режиме работает генератор.

То в STM32, где описание системы RCC занимает 35 страниц убористого текста, схема не влезает на экран, а куча библиотек, вроде CMSIS и SPL или визардов, наподобие того что есть в CoIDE, генерируют стартовый код, определить текущую тактовую частоту превращается в непростую задачу. Ее и будем решать.

Как же понять, что у нас получилось на SYSCLOCK после всех этих HSI, HSE, делителей, мультиплексоров и PLL умножителей? 

**Master Clock Output**  
Для STM32 у нас есть отличная штука, как MCO вывод. На который можно пнуть тактовую частоту и поглядеть осциллографом или частотомером. Правда если вы работаете на максимальной частоте в 72Мгц, то на MCO в лучшем случае вы можете выдать частоту поделеную всего лишь на два. А это 36Мгц, далеко не каждый осцил или частотомер в мультиметре такое схавает. 

Но метод хорош, позволяет произвести непосредственный замер.

Вот тут, в нижней части схемы стоит мультиплексор MCO. На который можно вывести

* PLLCLK/2 — на крайняк, когда точно знаем, что SYSCLOCK переведен на PLL можно смотреть ее
* SYSCLOCK — собственно тактовая частота
* HSI — внутренний RC генератор
* HSE — внешний кварцевый генератор

Таким образом, через MCO можно поглядеть также на то, насколько плавает частота у HSI или поглядеть запустился ли HSE. А то и затактовать внешнюю периферию. В общем, вариантов применения много.

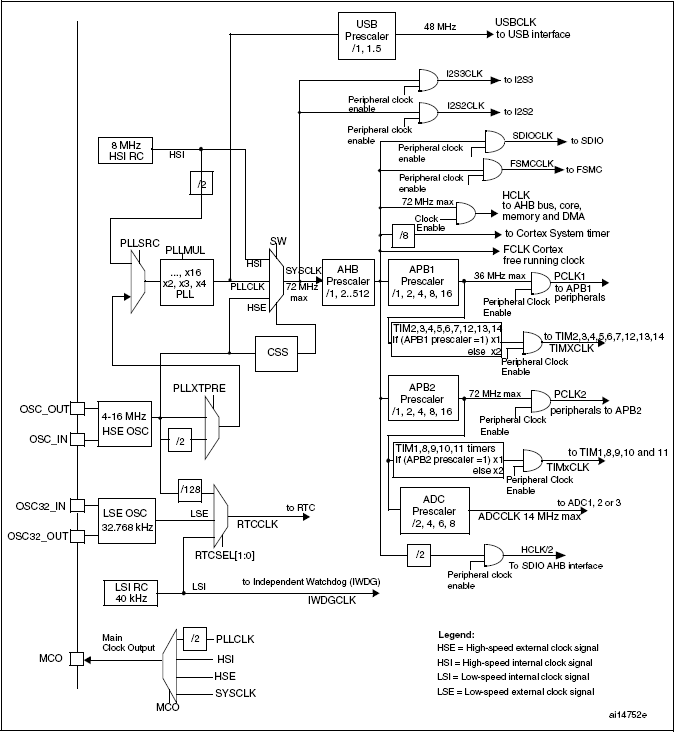


Рис. 1 – Блок-схема системы тактирования STM32F103 (дать для STM32F401/407)

Итак, надо всего лишь правильно сконфигурировать вывод МСО. На STM32F103C8T6, который используется в модулях ARM STM для [Pinboard](http://shop.easyelectronics.ru/index.php?categoryID=102) это вывод А08  
(дать для STM32F401/407)

Если через SPL, то это выглядит вообще просто:

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOA, ENABLE); *// Тактуем порт*

GPIO\_InitStructure.GPIO\_Pin = GPIO\_Pin\_8;

GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_50MHz;

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_AF\_PP;

GPIO\_Init(GPIOA, &GPIO\_InitStructure); *// Конфигурируем вывод MCO*

RCC\_MCOConfig(RCC\_MCO\_PLLCLK\_Div2); *// Подаем на MCO сигнал с PLL/2*

Вручную чуть сложней, надо будет полазать в даташите и в определениях CMSIS:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | RCC->APB2ENR |= RCC\_APB2ENR\_IOPAEN; *// Подаем тактирование на порт*    GPIOA->CRH &= ~GPIO\_CRH\_CNF8; *// Сбрасываем биты CNF для бита 8. Режим 00 - Push-Pull*  GPIOA->CRH |= GPIO\_CRH\_CNF8\_1; *// Ставим режим для 8 го бита режим CNF = 10 (альтернативная функция, Push-Pull)*    GPIOA->CRH &=~GPIO\_CRH\_MODE8; *// Сбрасываем биты MODE для бита 8*  GPIOA->CRH |= GPIO\_CRH\_MODE8\_1 | GPIO\_CRH\_MODE8\_0; *// Выставляем бит MODE для пятого пина. Режим MODE11 = Max Speed 50MHz*    RCC->CFGR &=~(RCC\_CFGR\_MCO); *// Обнуляем MCO*  RCC->CFGR |=RCC\_CFGR\_MCO\_PLL; *// Выставлем для MCO сигнал с PLL/2*  *//RCC->CFGR |=RCC\_CFGR\_MCO\_SYSCLK; // Выставляем для МСО сигнал с SYSCLK* |

RCC->APB2ENR |= RCC\_APB2ENR\_IOPAEN; *// Подаем тактирование на порт*

GPIOA->CRH &= ~GPIO\_CRH\_CNF8; *// Сбрасываем биты CNF для бита 8. Режим 00 - Push-Pull*

GPIOA->CRH |= GPIO\_CRH\_CNF8\_1; *// Ставим режим для 8 го бита режим   
 // CNF = 10 (альтернативная функция, Push-Pull)*

GPIOA->CRH &=~GPIO\_CRH\_MODE8; *// Сбрасываем биты MODE для бита 8*

GPIOA->CRH |= GPIO\_CRH\_MODE8\_1 | GPIO\_CRH\_MODE8\_0; *// Выставляем бит MODE для пятого пина. Режим MODE11 = Max Speed 50MHz*

RCC->CFGR &=~(RCC\_CFGR\_MCO); *// Обнуляем MCO*

RCC->CFGR |=RCC\_CFGR\_MCO\_PLL; *// Выставлем для MCO сигнал с PLL/2*

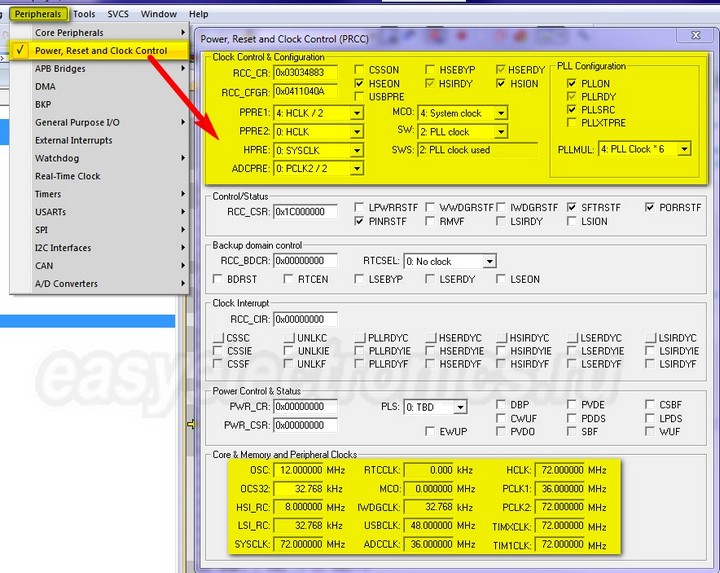
*//RCC->CFGR |=RCC\_CFGR\_MCO\_SYSCLK; // Выставляем для МСО сигнал с SYSCLK*

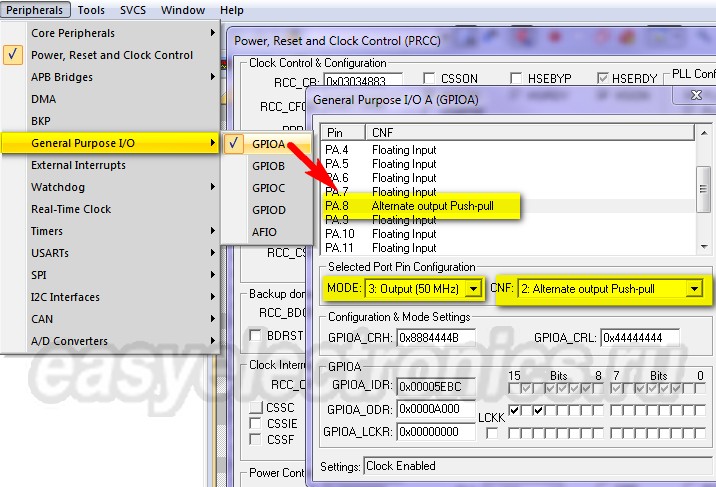
(дать для STM32F401/407)

Но результат тот же — тактовый сигнал на входе MCO который можно измерить осциллографом и понять на какой частоте завелся контроллер.

**Отладочные средства Keil**  
При работе в Keil, если запустить под отладчиком. например CoLinkEX (или просто CoLink, что установлен на [Pinboard II](http://shop.easyelectronics.ru/index.php?categoryID=102)), то можно открыть окно с настройками RCC и там посмотреть как происходит инициализация генератора. Какие режимы выставляются и так далее.

И, что самое приятное, там можно в режиме реального времени что угодно поменять на лету. Например, выдав сигнал на MCO (предварительно сконфигурировав вывод. Причем сделать это можно там же, в Keil, вызывав окно настроек GPIO и расставив галочки)





**Эталонный код**  
На худой конец можно сунуть в память простейший эталонный код. Который выдает заведомо известную выдержку на заведомо известной частоте. И по расхождении частоты судить о реальной частоте контроллера.

Вот этот код, например, выдает по 1килогерцу на 1 мегагерц тактовой. Замеряя частотомером сигнал на выходе B05 можно узнать величину тактовой частоты контроллера в текущий момент:

RCC->APB2ENR |= RCC\_APB2ENR\_IOPBEN;

*// Конфигурируем CRL регистры.*

GPIOB->CRL &= ~GPIO\_CRL\_CNF5; *// Сбрасываем биты CNF для бита 5. Режим 00 - Push-Pull*

GPIOB->CRL |= GPIO\_CRL\_MODE5\_0; *// Выставляем бит MODE0 для пятого пина. Режим MODE01 = Max Speed 10MHz*

while(1)

{

GPIOB->BSRR = GPIO\_BSRR\_BR5; *// Сбросили бит.*

Delay(55560);

GPIOB->BSRR = GPIO\_BSRR\_BS5; *// Установили бит.*

Delay(55560);

}

}

*/\* Тупая задержка \*/*

void Delay( uint32\_t Val)

{

for( ; Val != 0; Val--)

{

\_\_nop();

}

}

В примере используется порт B05 т.к. мне удобно его бросить на светодиод на [Pinboard](http://shop.easyelectronics.ru/index.php?categoryID=102). А так можно его перенастроить на любой другой. Разумеется изменив инициализацию.

Вот как это выглядит:

<https://youtu.be/vViNCbBg4AU>

FAQ

В RM0090 для камня STM32F4Dsicovery пишут:  
«MCO2: Microcontroller clock output 2. Set and cleared by software. Clock source selection may generate glitches on MCO2. It is highly recommended to configure these bits only after reset before enabling the external oscillators and the PLLs.»  
Т.е. конфигурировать вывод надо в области инициализации тактирования. Не возникают те самые » glitches» при твоей конфигурации ?