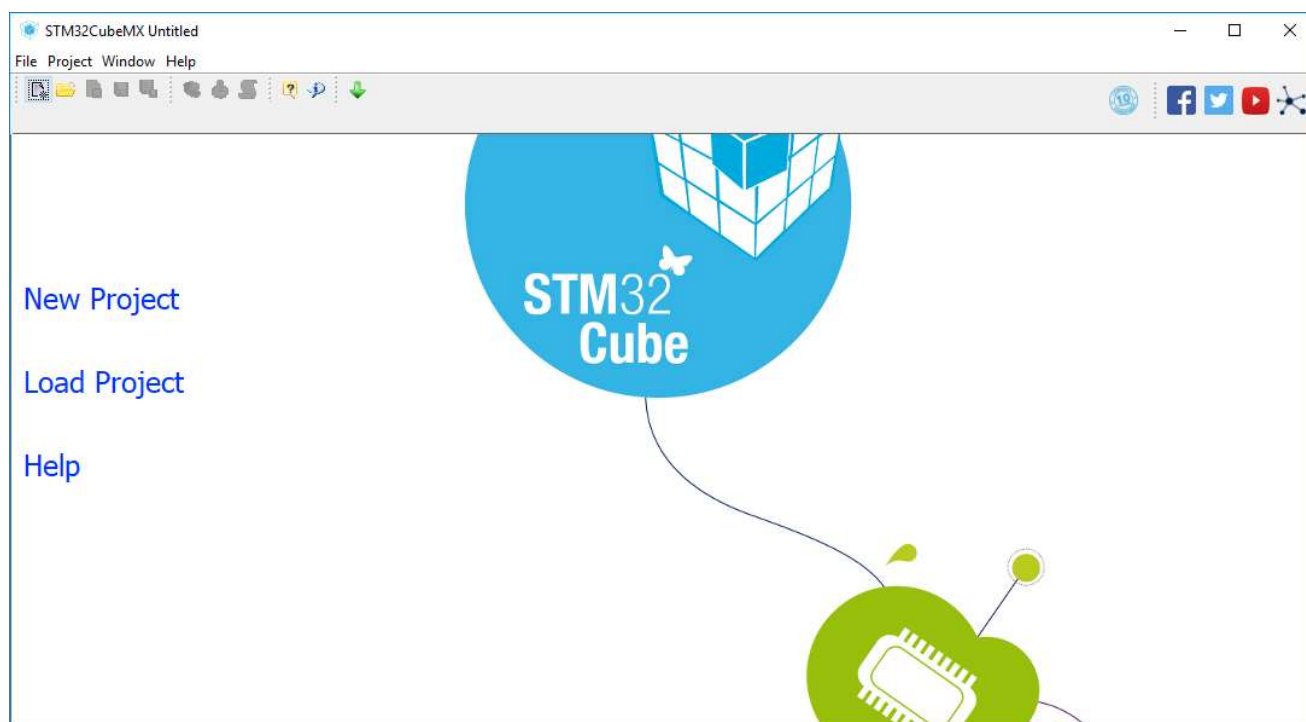


Генератор кода STM32CubeMX

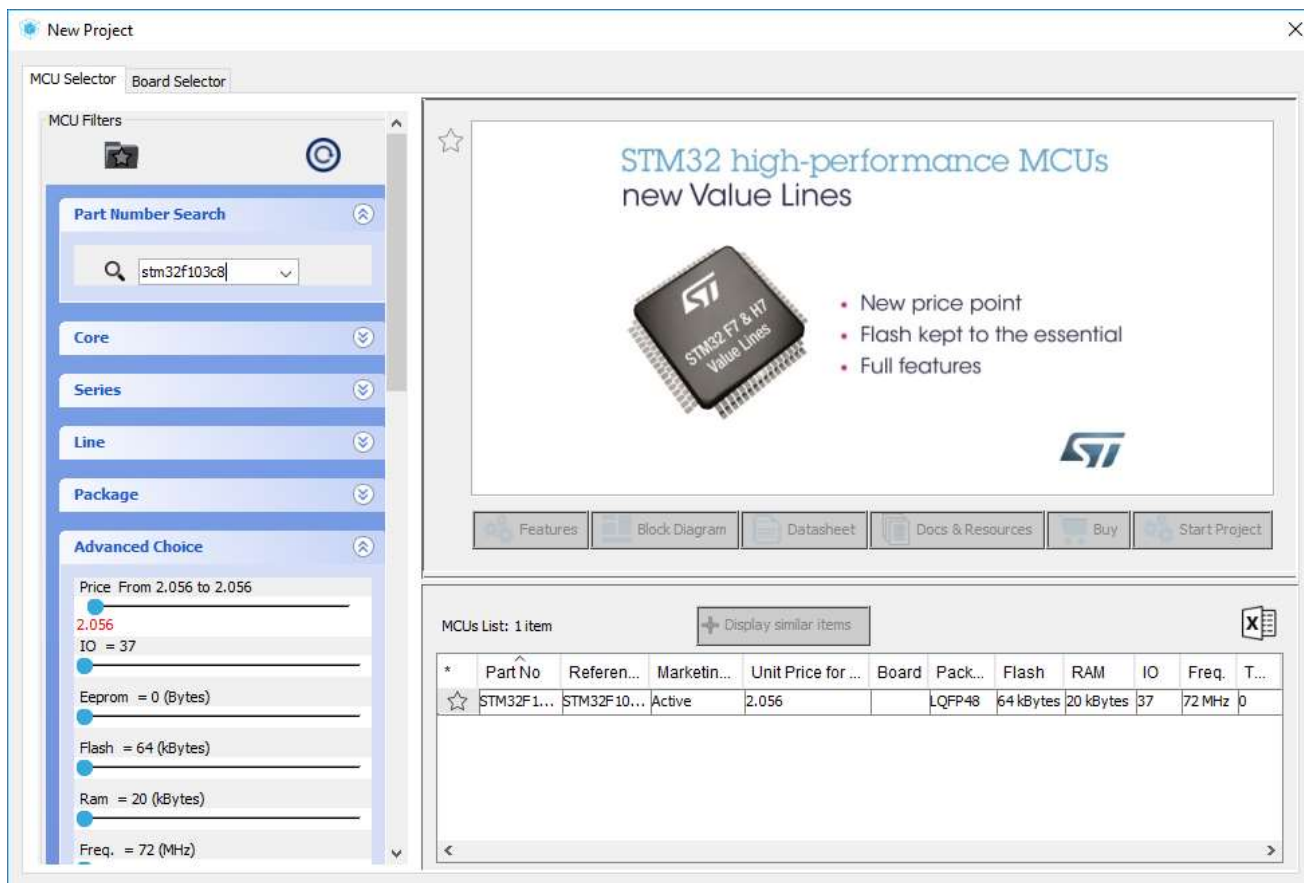
По всей видимости после невероятного успеха Arduino в компании ST задумались об ответе; нужно было упростить процесс разработки. Так появилась переосмысленная библиотека, HAL, и генератор кода [STM32CubeMX](#), который позволяет в графическом интерфейсе настроить всю необходимую в проекте периферию.

Пользоваться генератором кода, когда вы только учитесь — откровенно плохая идея. Вы получите результат быстро, но не осознаете всех этапов, а значит будете весьма посредственным специалистом. Мы не будем использовать вывод STM32CubeMX для настройки периферии, но воспользуемся им для создания проекта для IAR или Atollic TrueStudio (отдельно для HAL и LL, зависит от того, какую библиотеку вы решили использовать).

Скачайте и запустите STM32CubeMX.



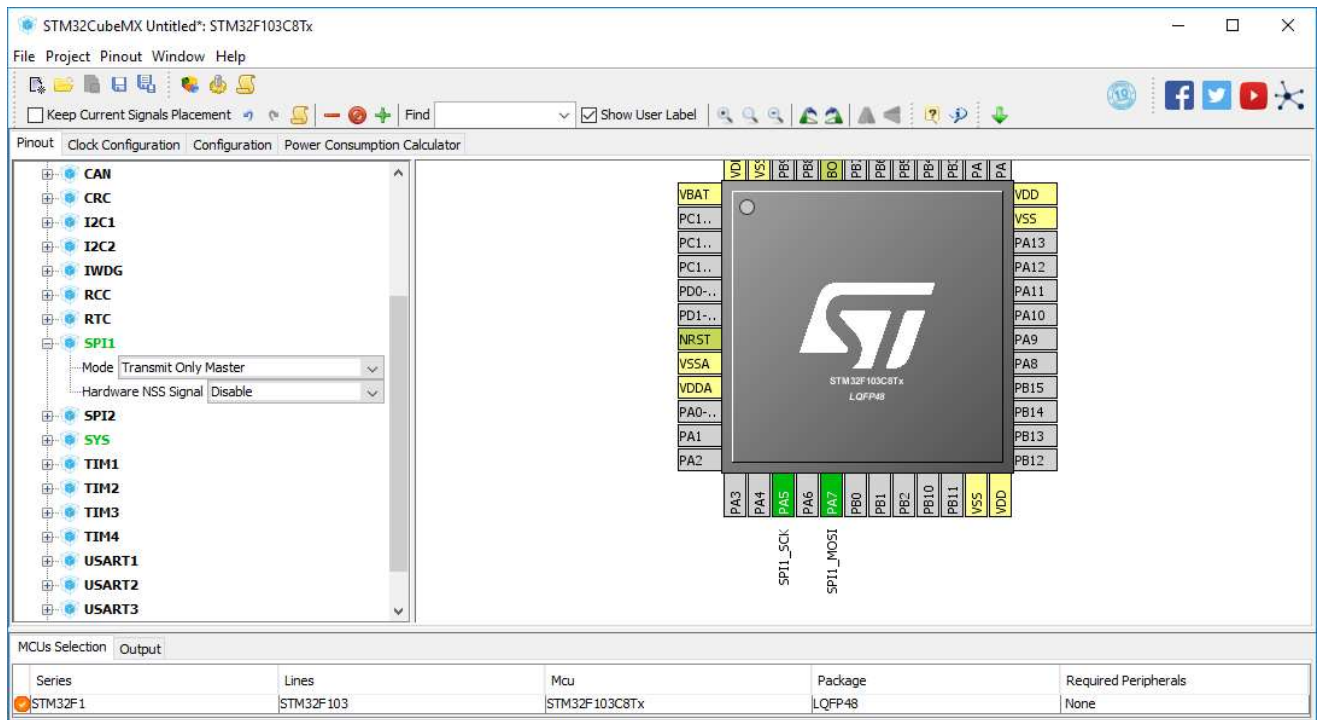
Создайте новый проект и выберите целевой микроконтроллер (через поиск), в нашем случае это `stm32f103c8`.



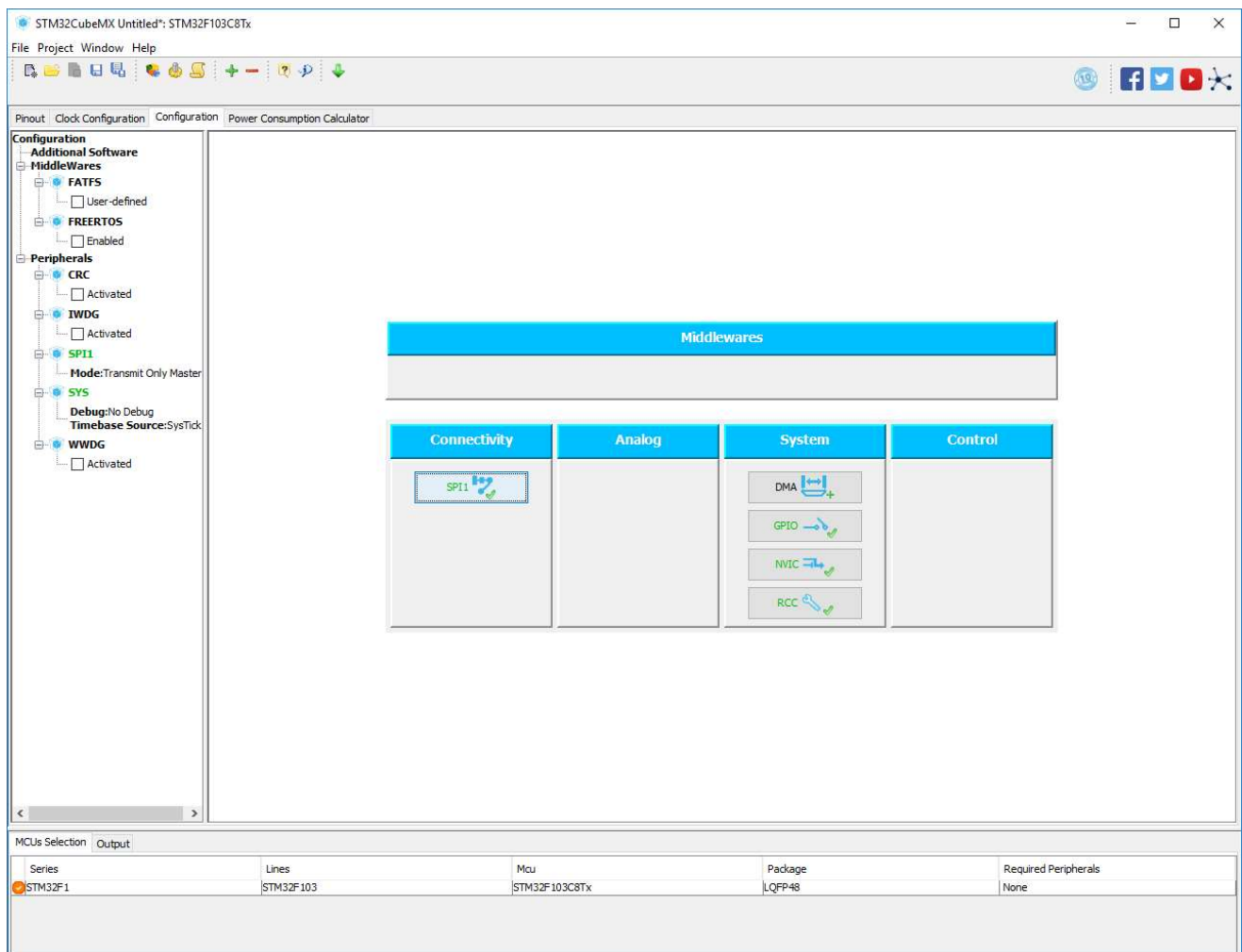
Если вы задумали какое-то своё устройство, то STM32CubeMX — это прекрасный инструмент для подбора подходящего МК: можно задать частоту, выбрать необходимую периферию и отсортировать результат поиска по цене.

Существует отдельная утилита (в том числе на Android и iOS) — MCU Finder.

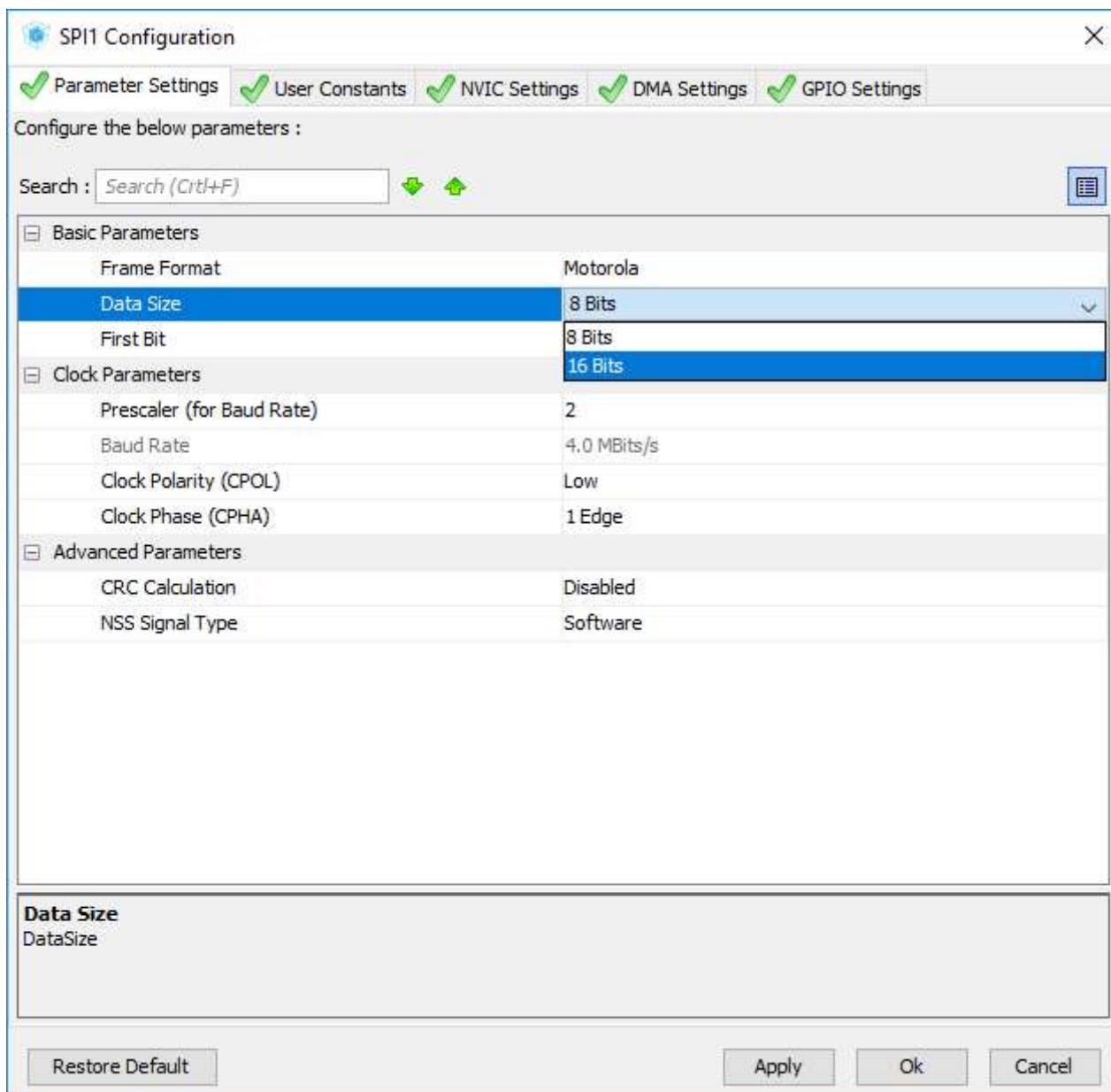
Выбрав микроконтроллер откроется следующее окно, где вам будет предложено расписать функции ножек. Для того что бы генератор автоматически добавил все необходимые файлы, вам необходимо задействовать необходимую вам периферию.



Перейдя во вкладку **Configuration** вы можете настроить каждый блок в отдельности, а так же подключить к нему блок прямого доступа к памяти.



Например, по-умолчанию размер данных SPI установлен 8 бит, но для микросхемы max7219 (или какой-нибудь другой в вашем проекте) нужно 16. Интерфейс позволяет быстро подправить значения на нужные.



SPI1 Configuration

Parameter Settings User Constants NVIC Settings DMA Settings GPIO Settings

Configure the below parameters :

Search :

Basic Parameters

Frame Format	Motorola
Data Size	16 Bits
First Bit	8 Bits

Clock Parameters

Prescaler (for Baud Rate)	2
Baud Rate	4.0 MBits/s
Clock Polarity (CPOL)	Low
Clock Phase (CPHA)	1 Edge

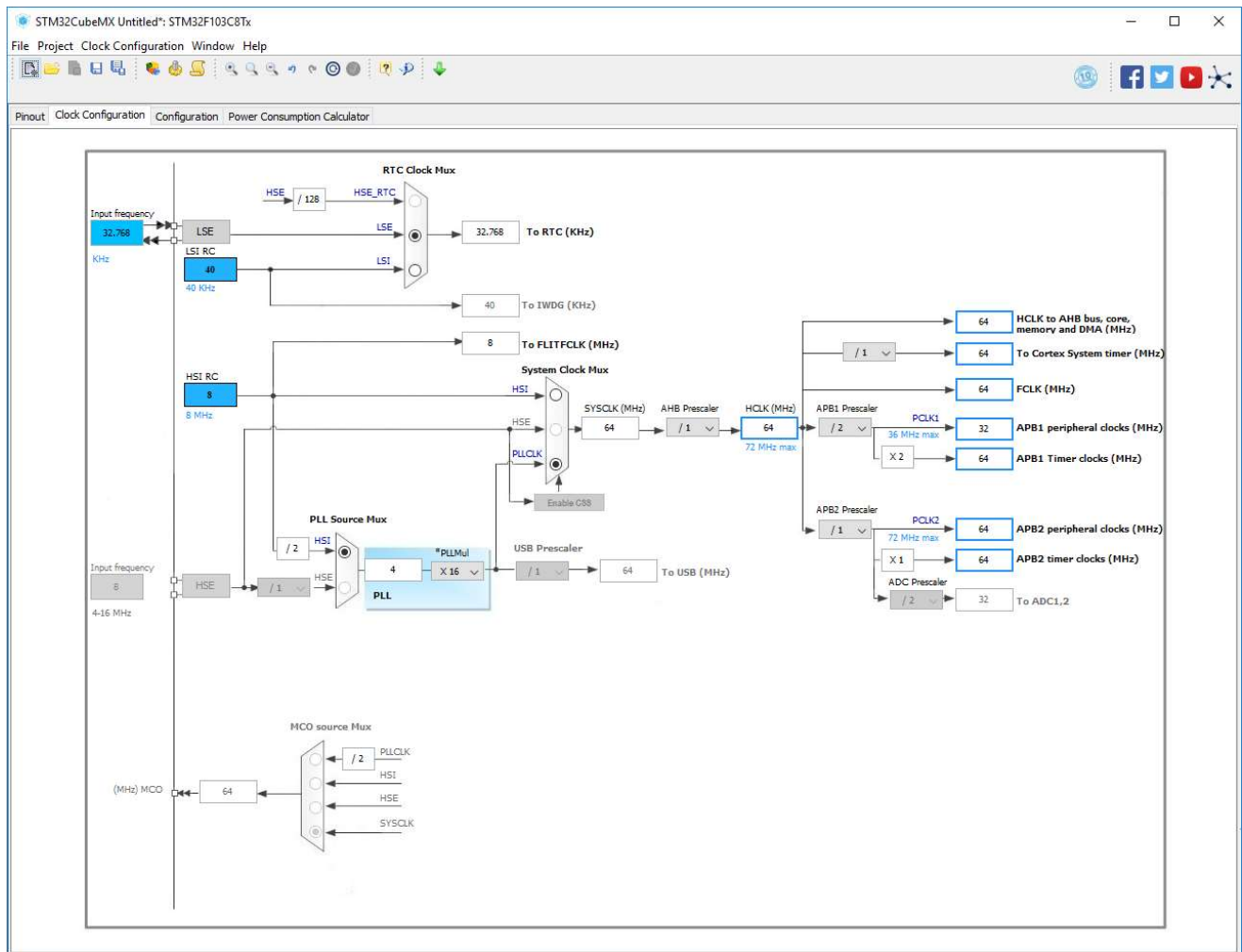
Advanced Parameters

CRC Calculation	Disabled
NSS Signal Type	Software

Data Size
DataSize

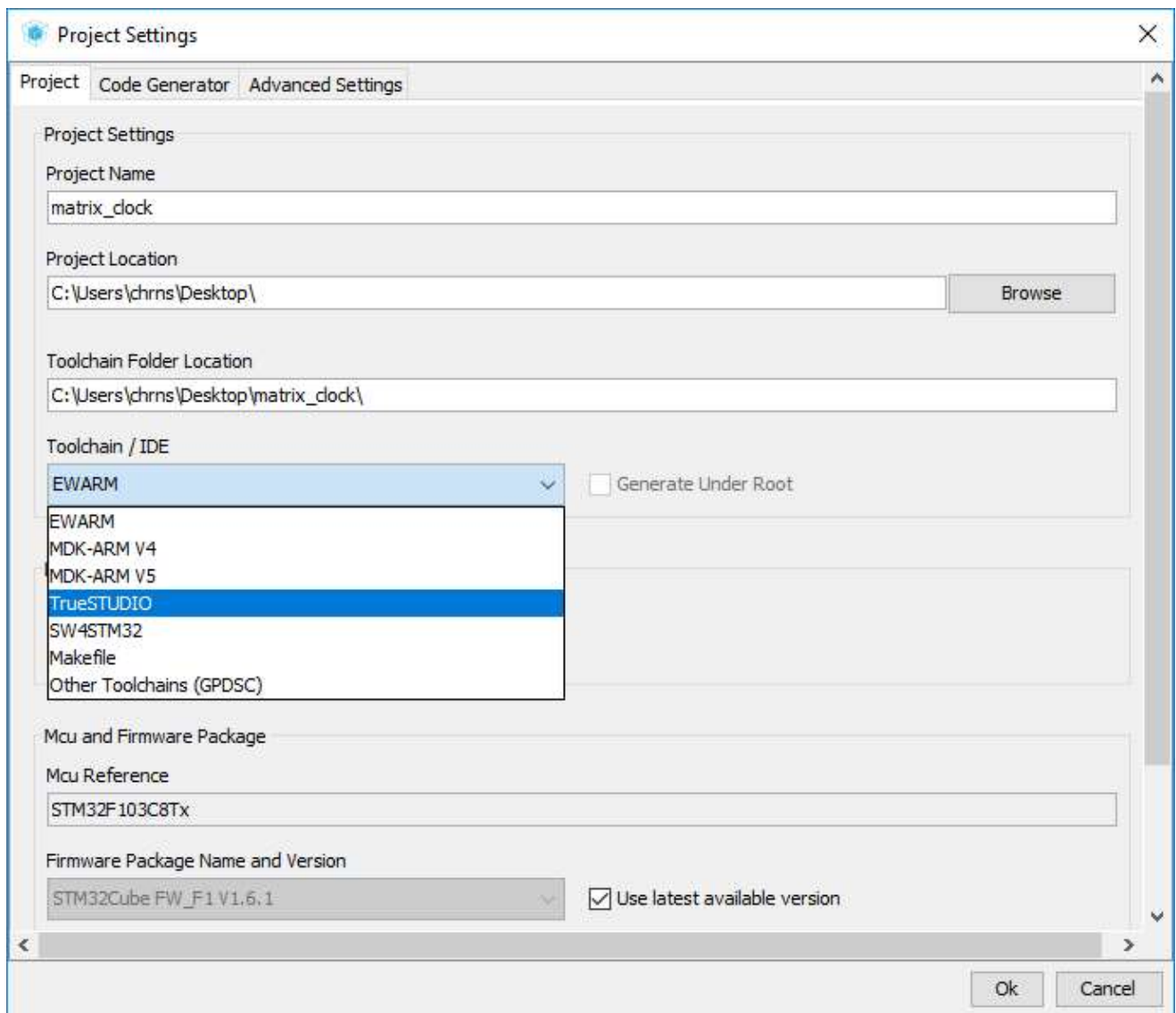
Restore Default Apply Ok Cancel

Ещё одной важным этапом настройки является система тактирования. Ниже приведена её диаграмма. Мы будем использовать LSE (внешний кварцевый резонатор) для часов реального времени, а сам МК будет тактироваться от внутренней RC-цепочки, сигнал от которой пропущен через умножитель частоты PLL (в итоге тактовая частота — 64МГц). Подробнее систему тактирования мы рассмотрим позже.



Так же STM32CubeMX позволяет настроить режимы сна (для экономии энергии), но мы данную функциональность трогать не будем, так как она нам не очень здесь интересна — устройство питается от сети, вопрос автономности не стоит.

Последнее, нужно сгенерировать проект. Делается это нажатием на шестерёнку на панели, либо через **Project ⇒ Generate Code**.



В сгенерированном проекте во всех пользовательских файлах будут подобные секции:

```
/* USER CODE BEGIN 3 */

/* USER CODE END 3 */
```

Предполагается, что весь пользовательский код, должен быть написан между ними. Это нужно для того, чтобы у вас осталась возможность безболезненно переконфигурировать периферию в STM32CubeMX и сгенерировать код заново, при этом пользовательский код останется нетронутым.

Так как мы не планируем использовать генератор в дальнейшем, пожалуйста удалите содержание файла `main.c`, он должен принять следующий вид:

```
#include "hal.h"

int main() {
    while(1) {
        // code
    }
}
```

Либо замените файл, взяв `main.c` из репозитория на [github](https://github.com).

[Назад](#) | [Оглавление](#) | [Дальше](#)