Инструкция по программированию Курского конструктора регулярной лиги для Воздушно-инженерной школы «CanSat в России»

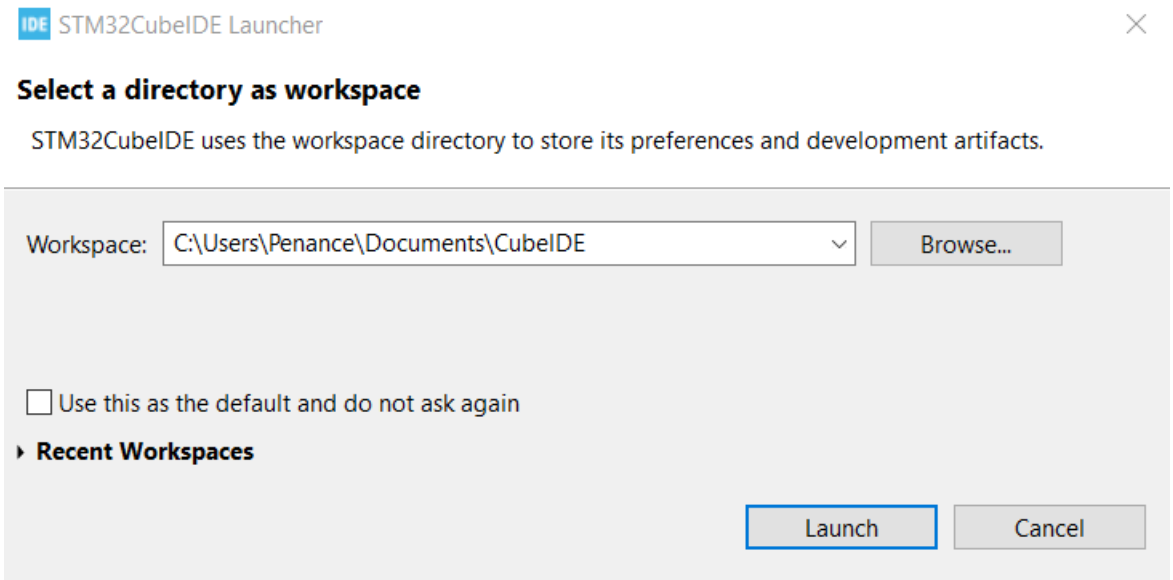
Необходимое ПО

Рекомендуется производить установку программ с параметрами по умолчанию.
Примечание: при установке в пути не должна присутствовать кириллица и сторонние символы (Аналогично и при создании проектов).

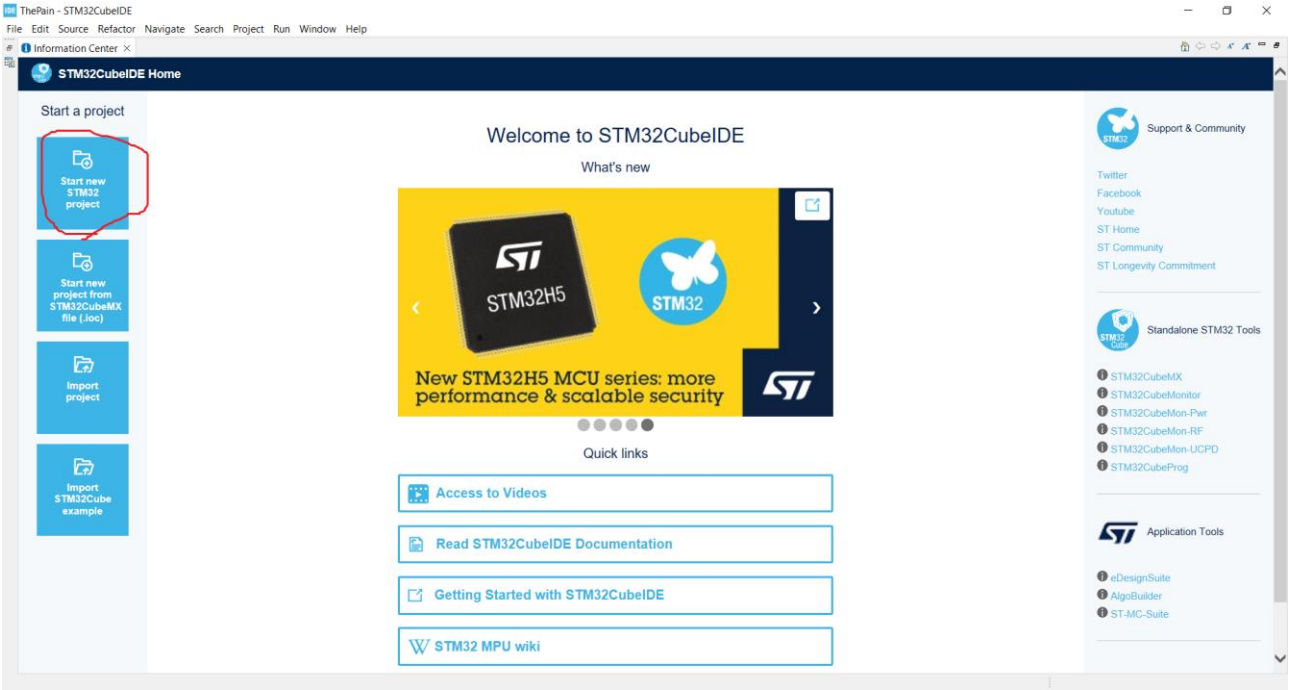
STM32CubeIDE – официальная среда разработки для STM32 (**ссылка**).
STM32CubeProgremmer – в случае, если решите прошивать микроконтроллер через USB загрузчик (**ссылка**).

Создание проекта

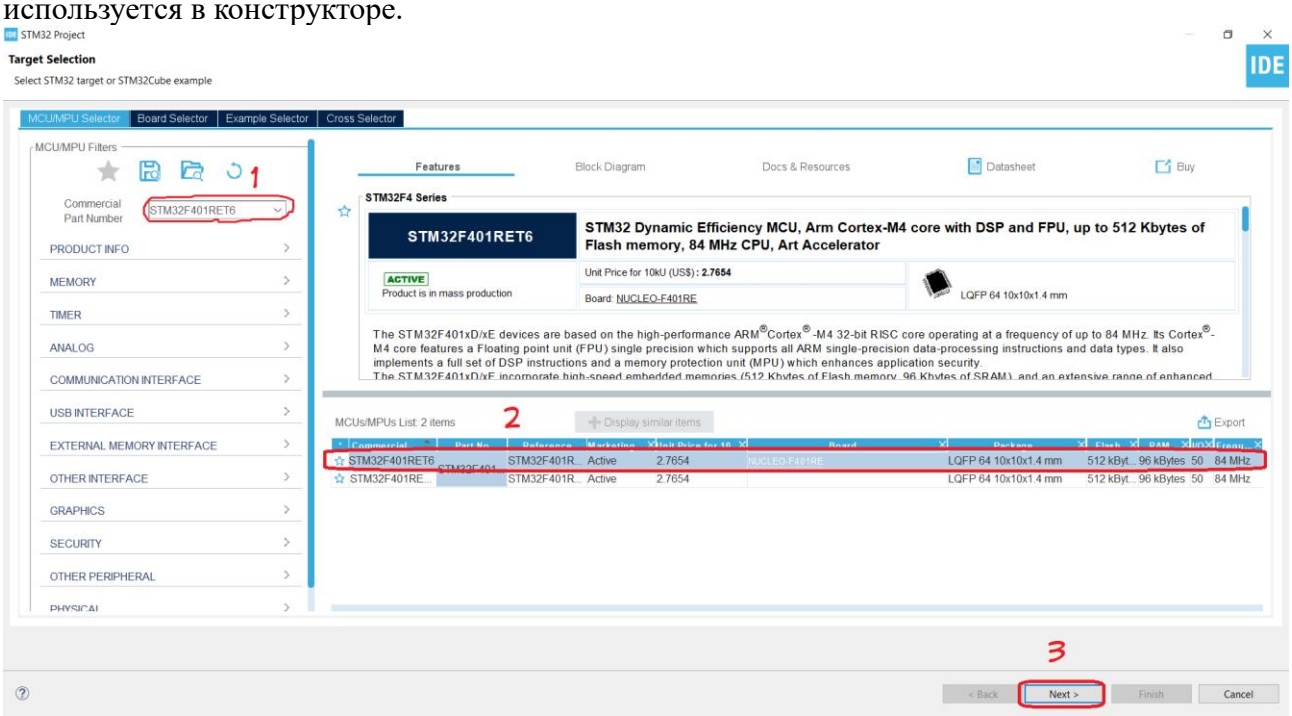
1. Запускаем CubeIDE.
2. При первом запуске программа предложит выбрать рабочую директорию (Папку, где будут храниться проекты). Необходимо её указать, любую удобную на выбор. *Примечание: в пути не должна присутствовать кириллица и сторонние символы.* Установив галочку в поле “Use this as the default and do not ask again” указанная папка установится по умолчанию и данное сообщение больше не будет показываться.



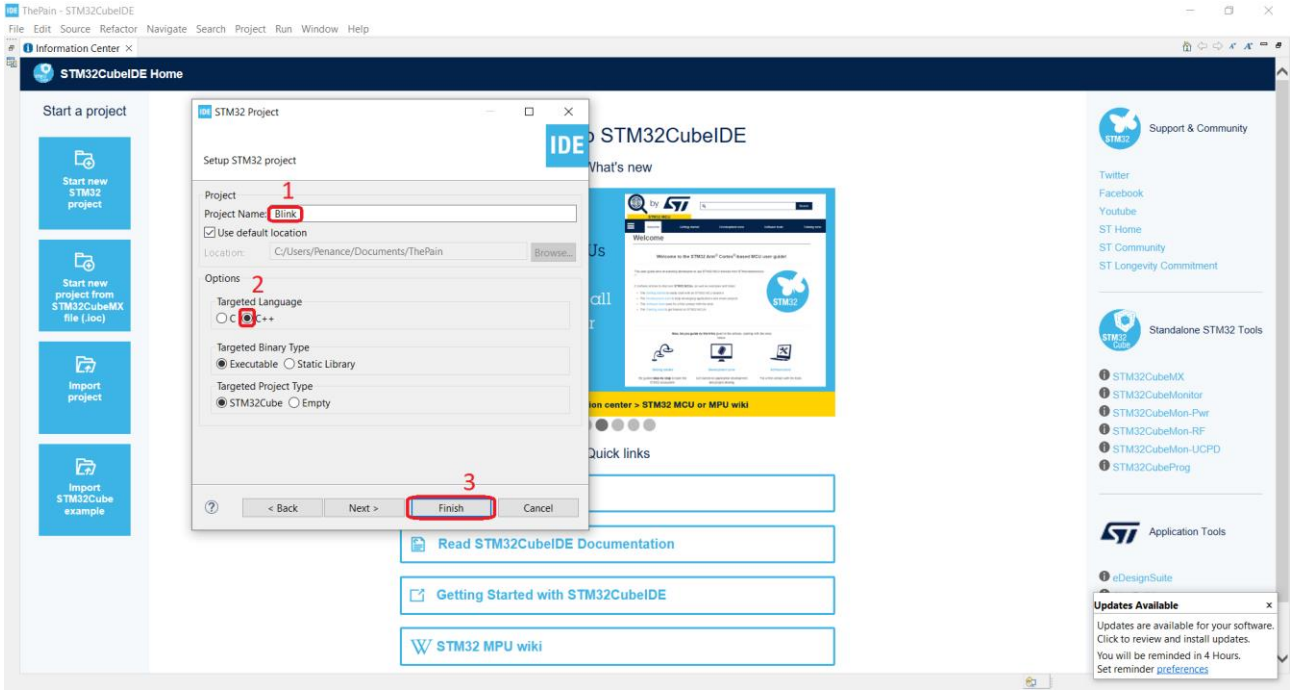
3. Далее CubeIDE покажет главное окно для создания проектов. Создадим новый проект, нажав на ссылку “Start new STM32 project”, либо через File -> New -> STM32 Project.



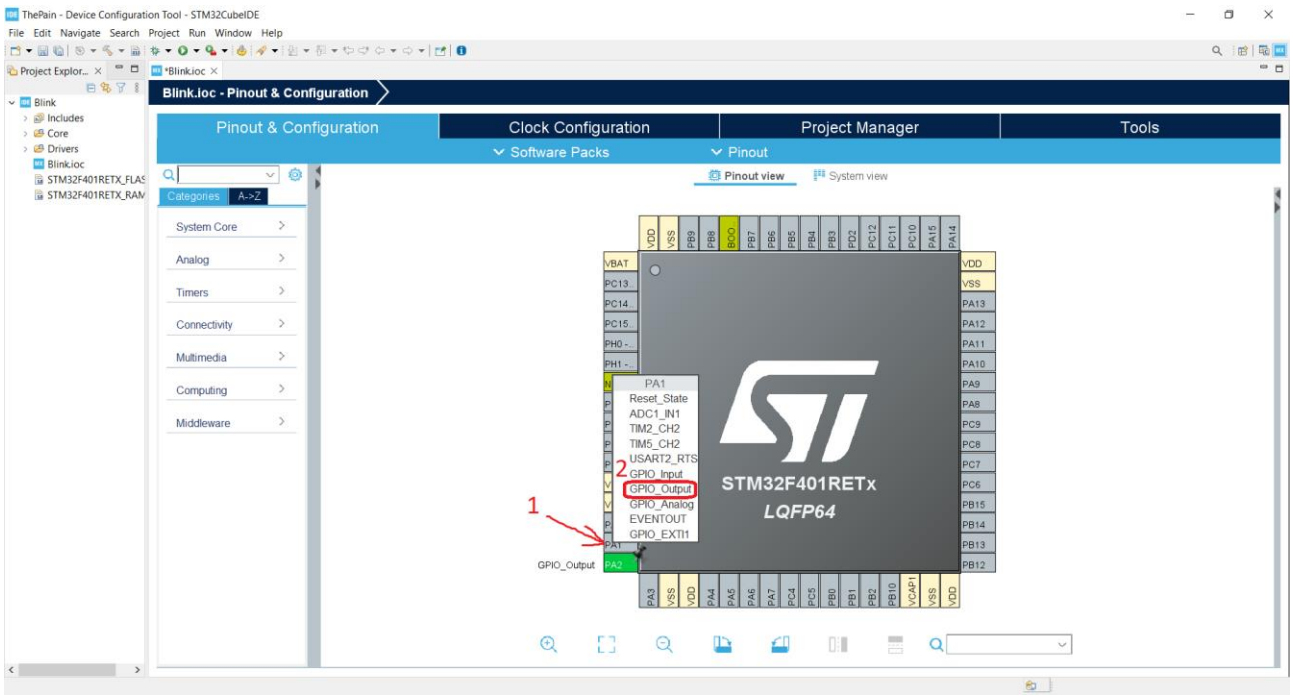
4. Открывается окно выбора микроконтроллера. Выбираем STM32F401RET6, который используется в конструкторе.



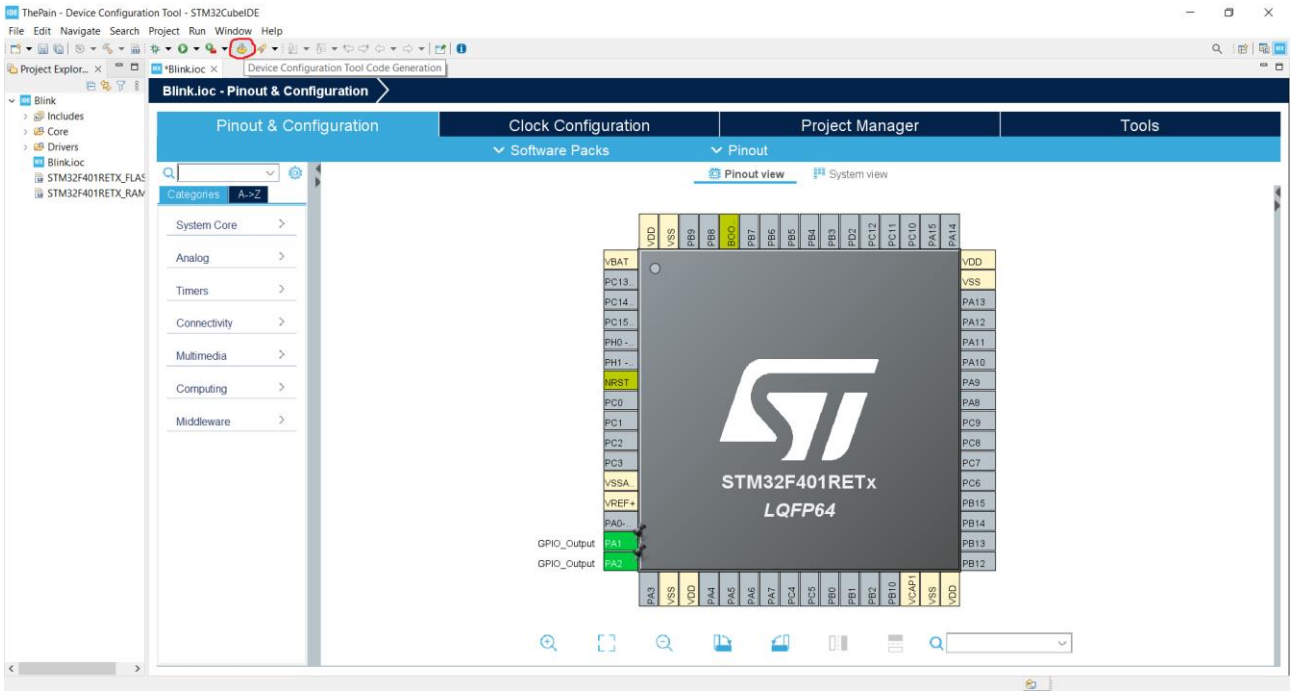
5. Необходимо указать название проекта и язык программирования, который вы будете использовать.



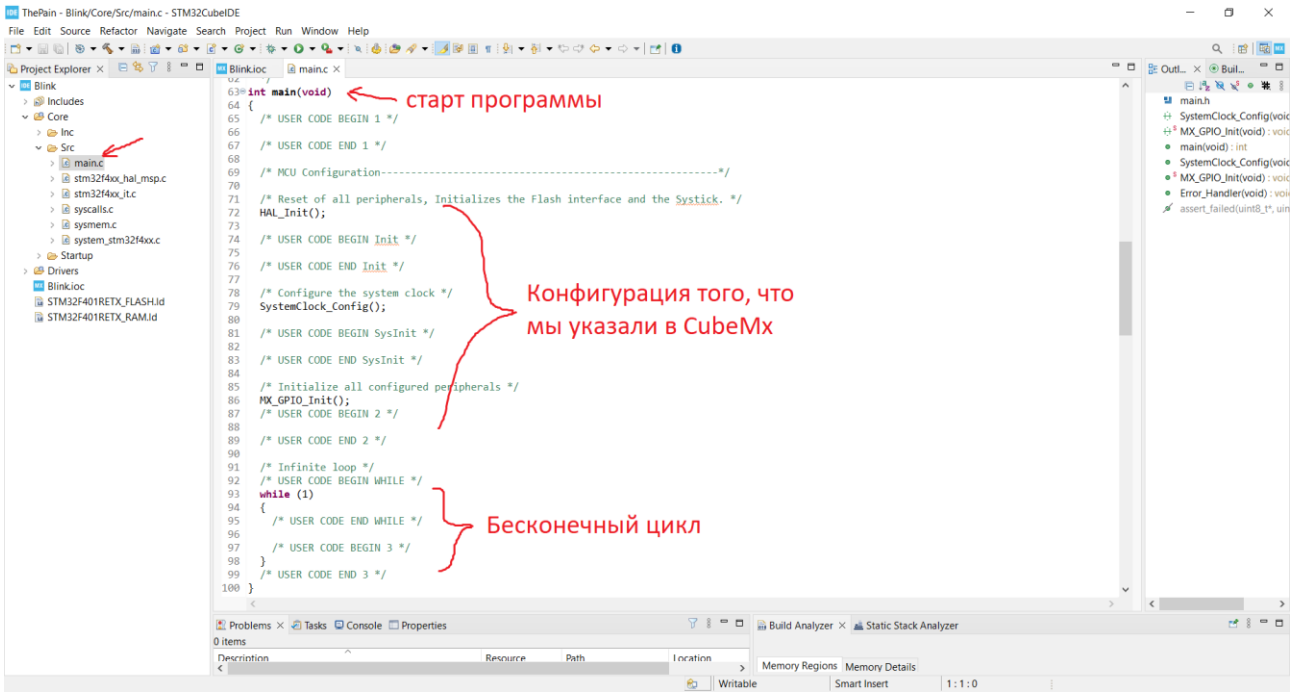
6. Открывается окно CubeMX – графический конфигуратор всей периферии STM32. Здесь можно легко настраивать все доступные интерфейсы, систему тактирования МК, подключать некоторые встроенные библиотеки и многое другое. Давайте попробуем помигать светодиодами LED3 и LED4 (Они подключены к ногам PA1 и PA2 соответственно, что можно определить по принципиальной электрической схеме). Настроим их на выход (Аналогично команде pinMode(OUTPUT, n) для Arduino). Для этого нажмём левой кнопкой мыши на PA2 и выберем из выпадающего списка “GPIO_Output”. Аналогичные операции производим для PA1.



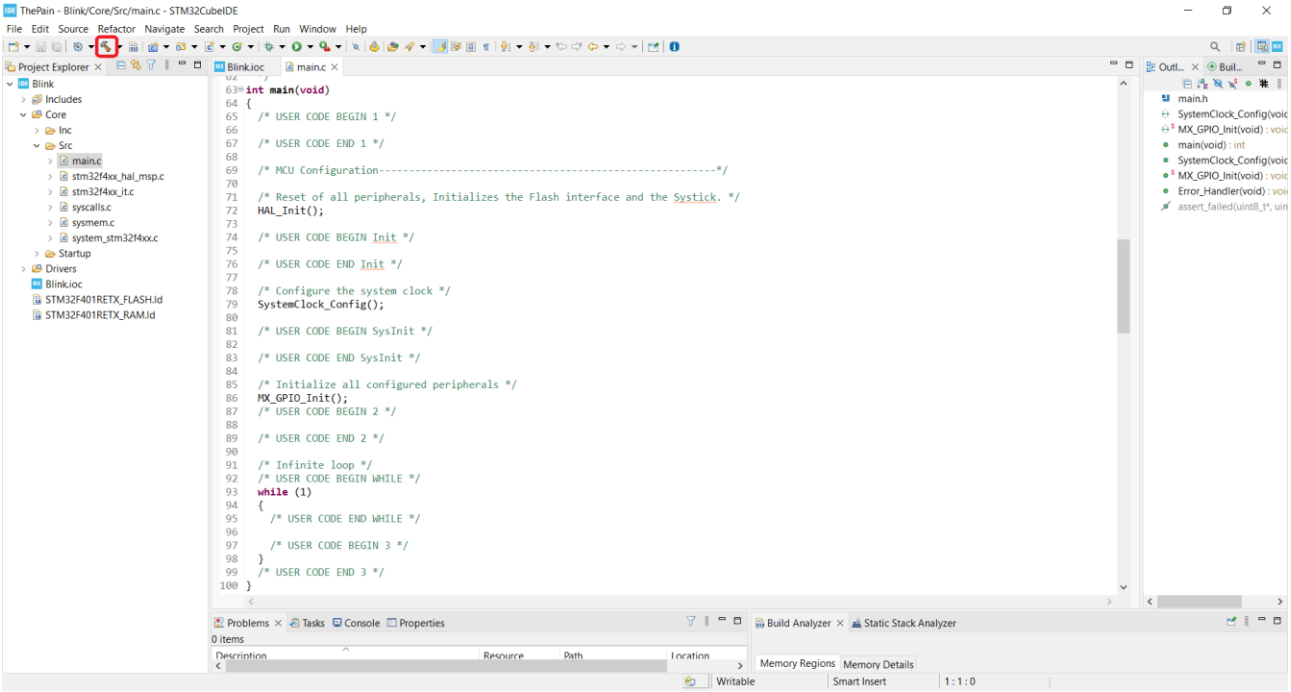
7. Сгенерируем проект, нажав на шестерёнку.



8. Откроется файл main.c – это основной файл вашей программы. Можно обнаружить множество других файлов, но они сейчас не нужны. Файл main.c лежит в Project→Core→Src. *Примечание для тех, кто раньше работал только с Arduino: здесь нет функций void setup() и void loop(). Старт программы происходит с функции int main() (Точка входа в программу), а вместо void loop() используется бесконечный цикл while(1).*



9. Чтобы скомпилировать проект нужно нажать на молоточек. В случае наличия ошибок или предупреждений, компилятор выведет их список.



10. Напишем простой код мигания светодиода:

```
HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, GPIO_PIN_SET);
HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, GPIO_PIN_RESET);
HAL_Delay(200);
HAL_GPIO_WritePin(GPIOA, GPIO_PIN_1, GPIO_PIN_RESET);
HAL_GPIO_WritePin(GPIOA, GPIO_PIN_2, GPIO_PIN_SET);
HAL_Delay(200);
```

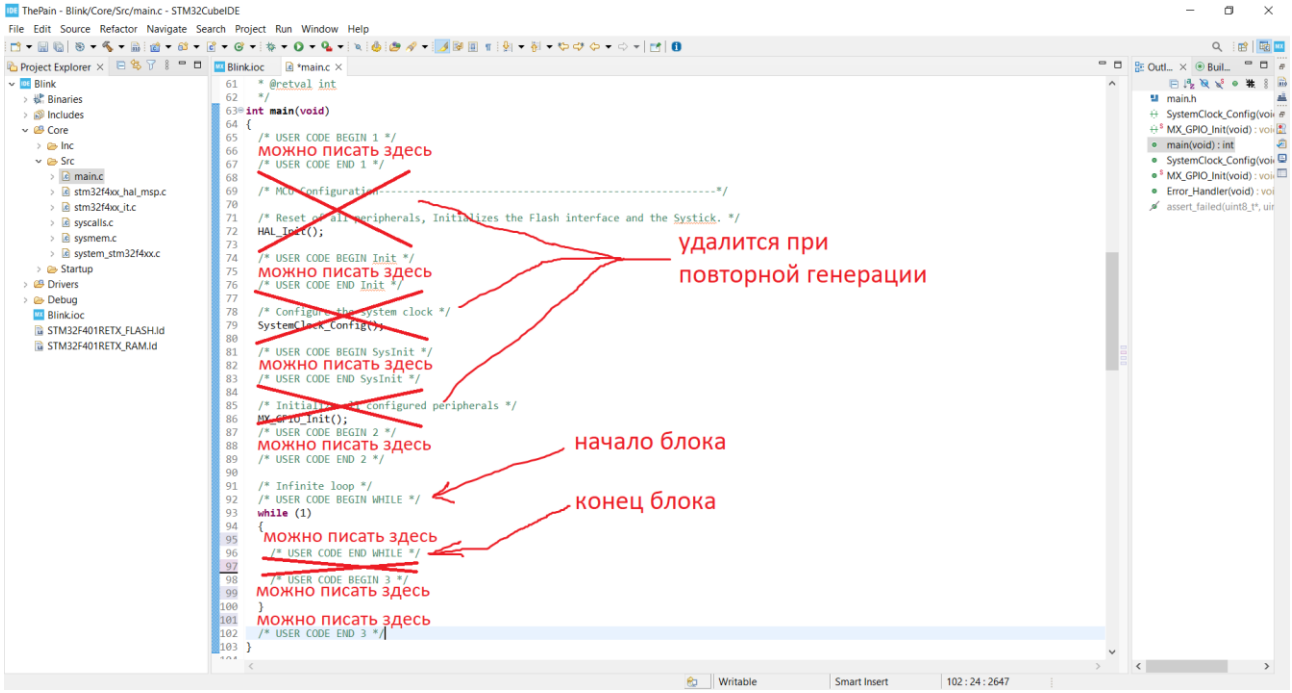
`HAL_GPIO_WritePin()` – это функция установки состояния вывода:

- Первый аргумент `GPIOA` – группа выводов
- Второй аргумент `GPIO_PIN_1` – номер вывода
- Третий аргумент *`GPIO_PIN_SET`* — состояние уровня (Низкий или высокий)

Подробнее можно прочитать здесь (**ссылка**).

`HAL_Delay(200)` – задержка выполнения программы, где аргумент – время в миллисекундах.

Примечание: если снова сгенерировать код в CubeMX, то код, который был написан вне специальных блоков, может пропасть.

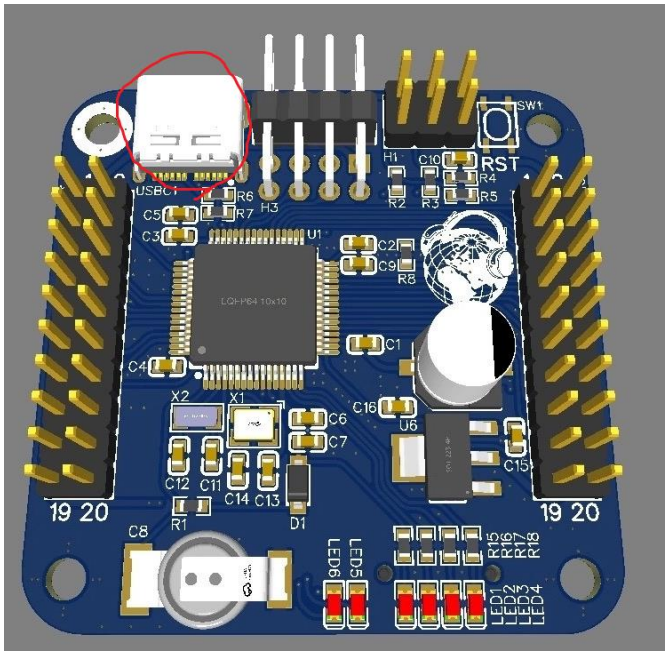


Прошивка микроконтроллера

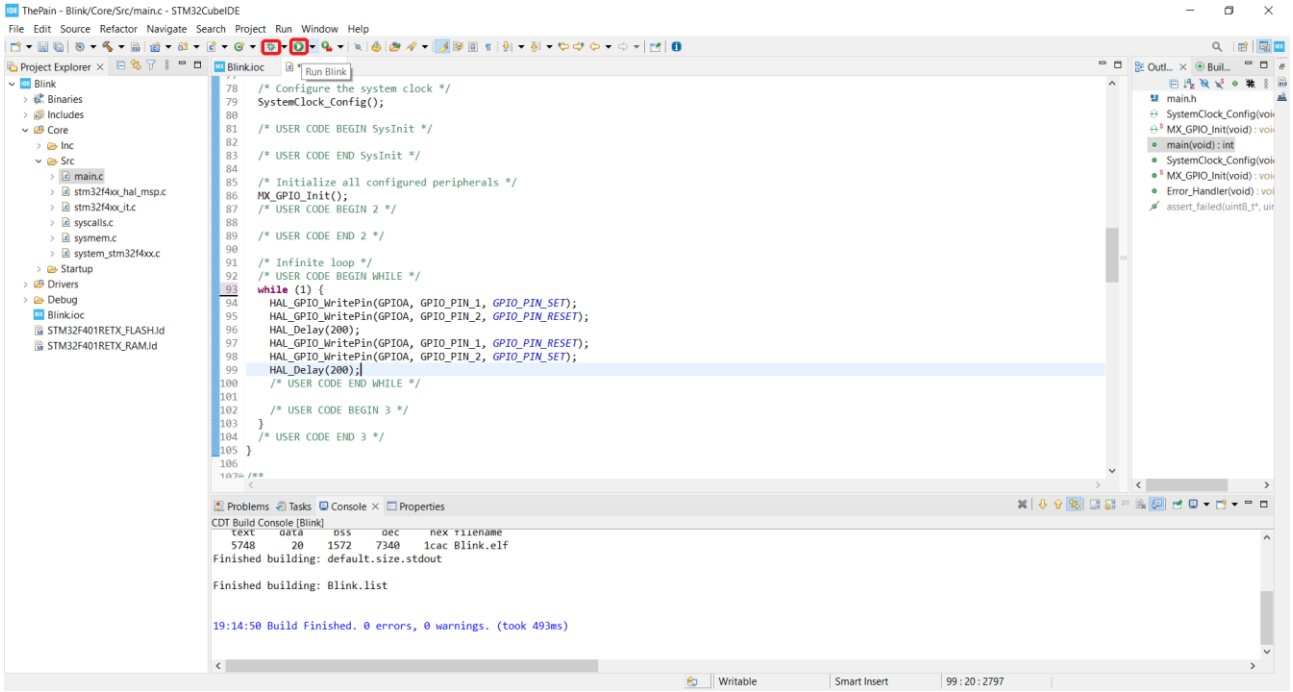
Есть множество способов загрузить программу в STM32, заложенных в микроконтроллер производителем. Здесь же мы приведём два основных способа сделать это.

1. Способ первый, ST-Link

1.1. Просто подключите компьютер к основной плате конструктора через кабель USB-C (Платы MCU и IMU&PWR должны быть соединены вместе (Перемычки DM, DP и BOOT должны быть разомкнуты)

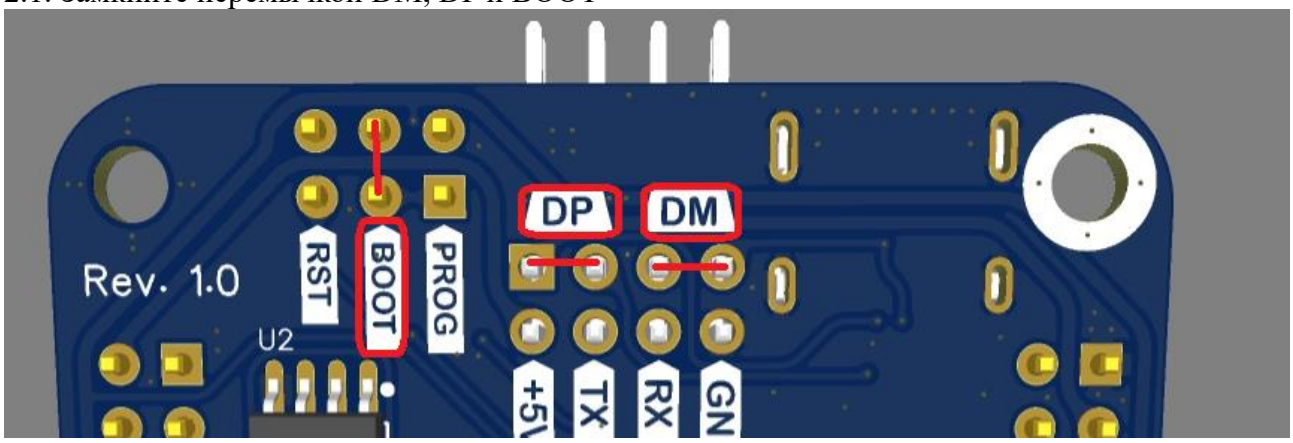


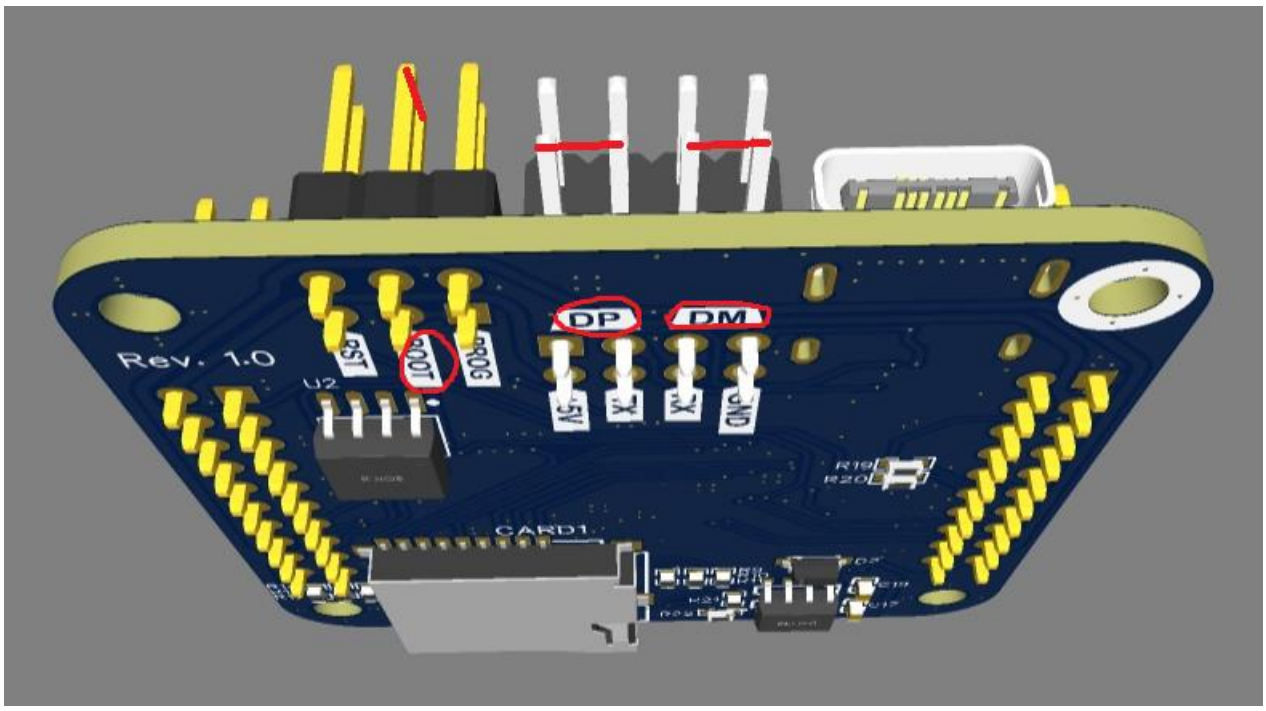
1.2. Открыв желаемый проект, нажмите на кнопку 'run', чтобы прошить микроконтроллер, или 'debug' (Жучок), чтобы прошить и войти в режим отладки. Если откроется окно настроек - оставьте всё по умолчанию и нажмите OK. Возможно, потребуется установка драйвера для ST-Link (ссылка).



2. Способ второй, USB загрузчик

2.1. Замкните перемычкой DM, DP и BOOT

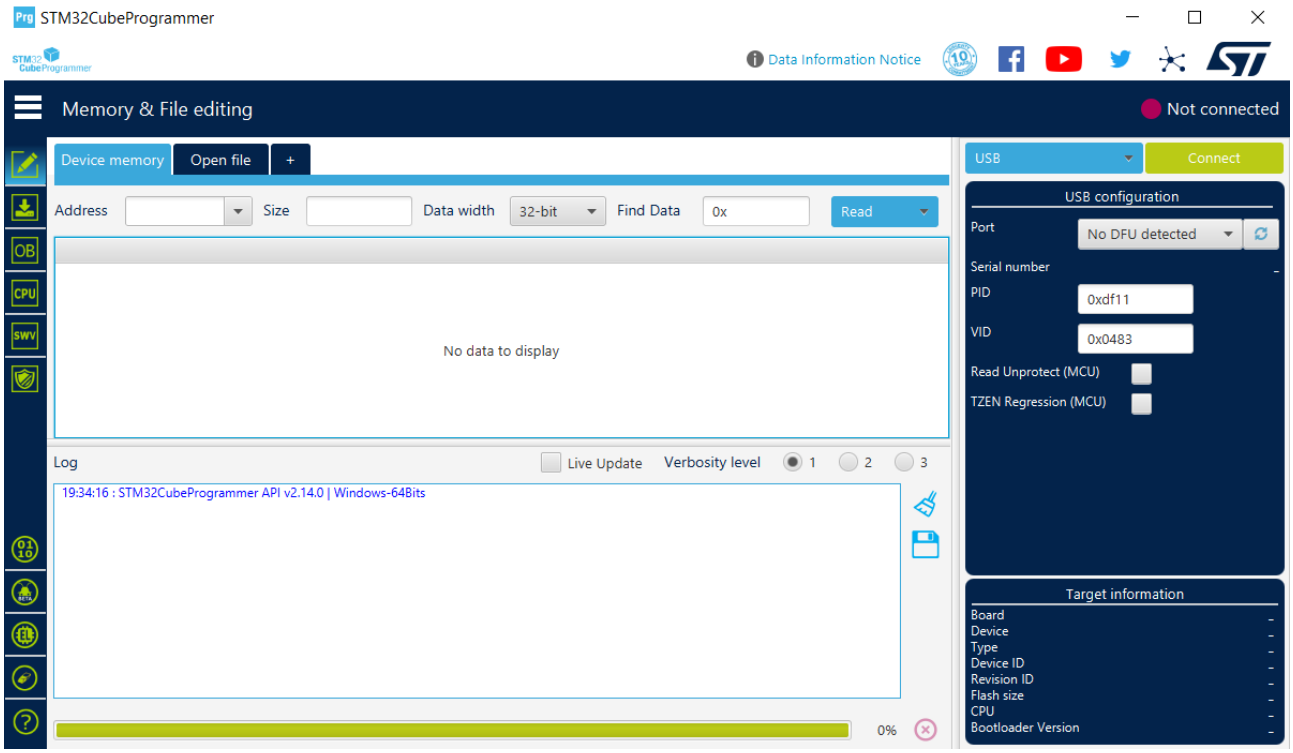




2.2. Подключите компьютер к основной плате конструктора через кабель USB-C (Платы MCU и IMU&PWR должны быть **разъединены**, а DM, DP и BOOT замкнуты (Замкните их перемычкой или проводом мама-мама)).

2.3. Скомпилируете проект (Было описано выше).

2.4. Откройте CubeProgrammer.



2.5. Выберите из списка справа USB, затем обновите список доступных устройств и выберите нужное. Затем подключитесь к нему, нажав “Connect“.

2.6. После успешного подключения к STM32, перейдите на вкладку загрузки кода, затем выберите файл с прошивкой (После компиляции, он доступен в папке с проектом→Debug→<название_проекта>.elf), после чего нажмите на “Start Programming“, чтобы запрограммировать микроконтроллер. Если поставить флаг “Run after programming“, то код начнёт работать сразу после загрузки.
Примечание: разомкните BOOT после прошивки, иначе микроконтроллер будет уходить в режим загрузчика, и код работать не будет.

Полезные материалы или «Что изучать дальше?»

Первым делом лучше всего подтянуть базовые знания C/C++. В интернете есть очень много всего: от книг и серии онлайн статей, до видеоуроков. Найдите то, что вам понравится. Если вам, как и нам, нравятся серии статей, то можем порекомендовать **metanit.com**.

Далее советую прочитать вот это:
Описание основных функций для работы с выводами GPIO на русском — возможно не всё сразу будет понятно, так как есть много терминологии, например «открытый сток» или «высокоимпедансное состояние», но со временем во всём можно разобраться.
Система тактирования STM32 — тут важно отметить, что здесь описывается система тактирования другой модели STM32, но различия будут небольшие. Возможно следует сначала прочесть про само тактирование. Вот небольшая статья про это (**ссылка**)
Таймеры в STM32 — очень качественная статья про таймеры, если вы не работали раньше с прерываниями, можем предложить эту статью про прерывания на Arduino (**ссылка**).
АЦП В STM32 — если вы собираетесь работать с аналоговым сигналом, то это статья для вас. Опять же сложно всё понять с первого раза, но, возможно, стоит прочитать хотя бы начало и запустить АЦП в базовом режиме

Затем нужно научится работать с такими интерфейсами, как: I2C, SPI, UART.
Мы также привели несколько статей здесь, но они скорее для того, чтобы показать это выглядит, поэтому вам придётся самостоятельно искать информацию про них, чтобы лучше разобратся.
UART.
Описание SPI, **Маленький пример.**
Описание I2C, **Описание работы с I2C на STM32.**

Далее необходимо научиться работать с той периферией, что установлена на платах и с библиотеками (Понять, как их можно установить и как переделать библиотеку с Arduino на STM32). Вот пример библиотеки для LIS3MD (**Библиотека**) — но как её установить?
Надо просто разложить файлы lis3mdl.c, lis3mdl.h по папкам Core→Inc и Core→Src, а затем открыть lis3mdl.h и посмотреть объявление функций и на комментарии в коде, что оставил автор. Если вы уже достаточно хорошо владеете C/C++, то это не составит для вас труда.

Точного алгоритма по переписыванию библиотеки с Arduino для STM32 нет. Необходимо понять, что делает чужой код для Arduino, и переписать код для STM32.

Заметьте, кстати, что некоторые выводы, выведенные на гребёнки, идут не от STM32, а от расширителя портов, которым управляет STM32 по I2C. Это что-то вроде дополнительных

выводов для микроконтроллера. Они могут работать, как на вход, так и на выход. Также к ним подключены и выводы с прерываниями от некоторых устройств.

О том, как работать с передатчиком, подробно описано в отдельным документе.

Помните самое главное, чтобы добиться успехов в любом деле нужно — работать, изучать и практиковаться и ещё раз работать, изучать и практиковаться.
Желаем всем успехов в ВИШ и в освоении нашего конструктора!

Наши контакты

Пожалуйста, не стесняйтесь писать нам по любым вопросам, связанным с конструктором, например если столкнётесь с какими-либо проблемами в ходе работы с ним или хотите оставить отзыв или предложение по его улучшению.

Левент Денис Вейсиевич (VK, Telegram) – инженер-схемотехник.
Кондрашков Хамза Амирович (VK, Telegram) – программист, тестировщик.
КБ «Инженерно-космическая школа» (VK).