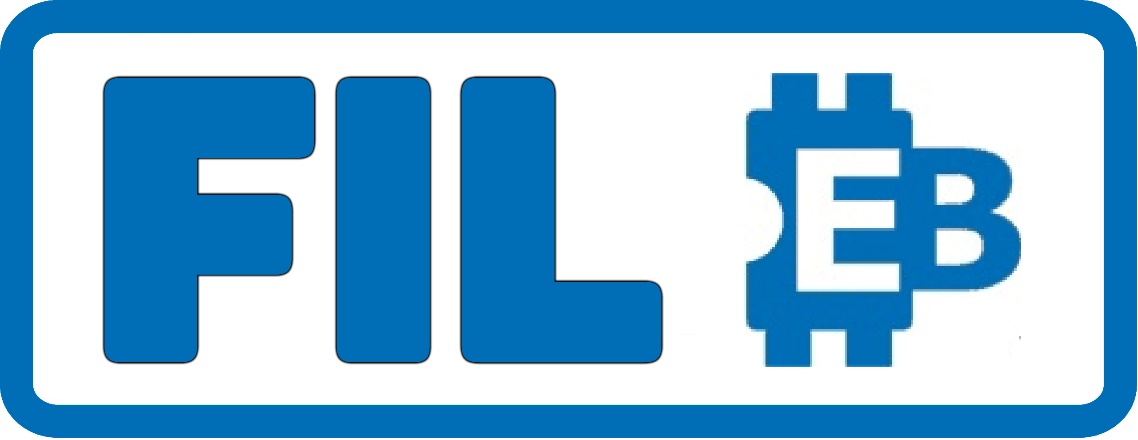
**STM32 Fast Initialization Library (FIL).**

Библиотека быстрой инициализации для микроконтроллеров STM32 на базе программной IDE EmBitz.

**Авторы: Назаров А.А., Гаранин Е.О.**

****

***[STM32F407](https://www.st.com/resource/en/reference_manual/dm00031020-stm32f405-415-stm32f407-417-stm32f427-437-and-stm32f429-439-advanced-arm-based-32-bit-mcus-stmicroelectronics.pdf)***

***[STM32F401](https://www.st.com/resource/en/reference_manual/dm00096844-stm32f401xb-c-and-stm32f401xd-e-advanced-arm-based-32-bit-mcus-stmicroelectronics.pdf)***

***[STM32F411](https://www.st.com/resource/en/reference_manual/dm00119316-stm32f411xc-e-advanced-arm-based-32-bit-mcus-stmicroelectronics.pdf)***

**Содержание**

|  |  |  |
| --- | --- | --- |
|  | [Введение](#Введение" \o "Введение) | 4 |
| **1.** | [Структура библиотеки FIL](#Раздел1" \o "Раздел 1. Структура Библиотеки FIL) | 6 |
|  | **1.1** [Архитектурное строение](#Раздел11) | 6 |
|  | **1.2** [Рекомендации по установке и список поддерживаемых устройств](#Раздел12) | 7 |
|  | **1.2.1** [Версия для начинающих](#Раздел121) | 7 |
|  | **1.2.2** [Версия для опытных пользователей](#Раздел122) | 6 |
|  | **1.3** [История разработки](#Раздел13) | 6 |
| **2.** | [Конфигурация библиотеки](#Раздел2" \o "Раздел 2. Конфигурация Библиотеки) | 6 |
|  | **2.1** [Файл-карта портов](#Раздел21" \o "Раздел 2.1 Файл-карта портов) | 6 |
|  | **2.2** Минимальная версия конфигурации | 6 |
|  | **2.3** Создание своей конфигурации | 6 |
| **3.** | Список API функций и макросов библиотеки FIL | 6 |
|  | **3.1** Регламентация названий | 6 |
|  | **3.2** Опциональные правила использования | 6 |
|  | **3.3** Перечень макросов конфигураций | 6 |
|  | **3.4** Перечень API функций | 6 |
|  | **3.4.1** Функции RCC (Reset Clock Configuration) | 6 |
|  | **3.4.2** Функции GPIO (General Purpose Input/Output) | 6 |
|  | **3.4.3** Функции TIM (Timer Interface Mode) | 6 |
|  | **3.4.4** Функции ADC (Analog-Digital Converter) | 6 |
|  | **3.4.5** Функции USART (Universal Asynchronous Receiver-Transmitter) | 6 |
|  | **3.4.6** Функции I2C (Inter Integrated Circuit) | 6 |
|  | **3.4.7** Функции DMA (Data Memory Access) | 6 |
|  | **3.4.8** Функции Регулятора | 6 |
|  | **3.4.9** Функции EXTI (External Interrupts) | 6 |
|  | **3.4.10** Функции SPI (Serial Peripheral Interface) | 6 |
|  | **3.4.11** Функции DAC (Digital-Analog Converter) | 6 |
|  | **3.4.12** Функции CAN (Controller Area Network) | 6 |
| **4.** | Использование операционной системы реального времени для микроконтроллеров | 6 |
| **5.** | Помощь в поддержке продукта | 6 |
|  | Листинг А | 6 |
|  | Листинг Б | 6 |
|  | Листинг В | 6 |
|  | Список литературы | 6 |

**[Введение](#АВведение)**

Популярность микроконтроллеров Atmega (Arduino series) в любительских проектах, а также некоторых профессиональных, обосновывается простотой использования и уменьшении времени разработки полномасштабного и функционального алгоритма управления. Однако, для дальнейшего углубленного изучения принципа функционирования микроконтроллерных систем среда не предоставляет набор инструментария и функций. Пользователь погружаясь в дальнейшее изучение сталкивается с трудностями восприятия нового стиля написания программ через регистровые переменные, что увеличивает время обучения и проектирования алгоритмов.

Одним из популярных аналогов являются микроконтроллеры компании ST Microelectronics – STM8, STM32. Данные линейки контроллеров имеют обширную периферию, а доступ к изменению регистровых параметров позволяет разработчику настроить под себя систему управления, обмена данными и иными функциями. Общение через регистры низкоуровневым стилем программирования для начинающего компенсируется познанием базовых принципов и нюансов при инициализации и отладке программного кода. Данный подход не эффективен применительно к начинающим программировать данные микроконтроллеры. Поэтому сотрудниками Ресурсного центра робототехники (РЦР) Донского государственного технического университета (ДГТУ) была выдвинута идея создания полномасштабной библиотеки обучения и плавного перехода с высокого на низкий уровень проектирования алгоритмов.

Альфа версия библиотеки Fast Initialization Library (FIL) вышла в октябре 2021 года, имея в своем функционале инициализацию малого участка периферии. Поддержка продукта осуществляется и по сей день. За развитием проекта можно следить через платформу Github[1].

В данном документе приведены основные сведения для взаимодействия с библиотекой быстрой инициализации микроконтроллеров STM32. В основных разделах содержится информация по управляющим командам, а также примеры их использования. Приведена структура и принцип работы обучающей платформы, на основе которой демонстрируется функционал библиотеки. Документ дополняется по мере увеличения функционала библиотеки.

Все файлы и программный код библиотеки одобрены для коммерческого использования, любой желающий может использовать её в своих проектах. Приветствуется распространение и отзыв обратной связи о функциональности.

Для этого просьба обращаться по адресу - г.Ростов-на-Дону, пл.Гагарина 1, ДГТУ, 2 корпус, 2-205.

Также возможен вариант отправки письма на почту –

[sasha.sanya.nazarov@yandex.ru](mailto:sasha.sanya.nazarov@yandex.ru)

garanin1392@gmail.com

1. **[Структура библиотеки FIL](#АРаздел1)**

**1.1** **[Архитектурное строение](#АРаздел11)**

Библиотека имеет структуру, изображенную на рисунке 1.1. В ней есть участок пользовательского кода, написанный или планируемый программистом, файл-карта портов текущего микроконтроллера, конфигурационный файл с определениями режимов работы каждого периферийного участка, интегрированный в IDE EmBitz файл с определениями библиотеки CMSIS, позволяющий обращаться к адресам ключевых участков через словесные интерпретации и файлы библиотеки FIL, представленные двумя фрагментами: линкер файлом и определениями основных функций (FIL API).

Библиотека использует определения конфигурационного файла и карты портов, чтобы предоставить к использованию необходимые функции, по умолчанию доступ к библиотеке через пользовательское пространство недоступен. После инициализации периферии линкер связывает необходимые файлы, в которых хранятся API функции. Определения регистровой библиотеки CMSIS необходимы для работы библиотеки, обычно они встроены в большинство IDE и имеют большой список поддерживаемых контроллеров ST.

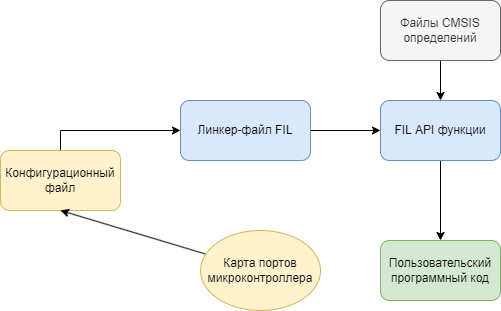
****

Рисунок 1.1 – архитектурное строение и взаимодействие с библиотекой FIL.

**1.2** **[Рекомендации по установке и список поддерживаемых устройств](#АРаздел12)**

**1.2.1** **[Версия для начинающих](#АРаздел121)**

Установка начинается с создание нового проекта, согласно рисунку 1.1 и выбора архитектуры и названия проекта (рисунок 1.2).

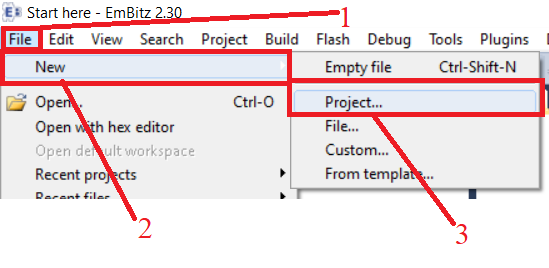


Рисунок 1.1 – Создание проекта.

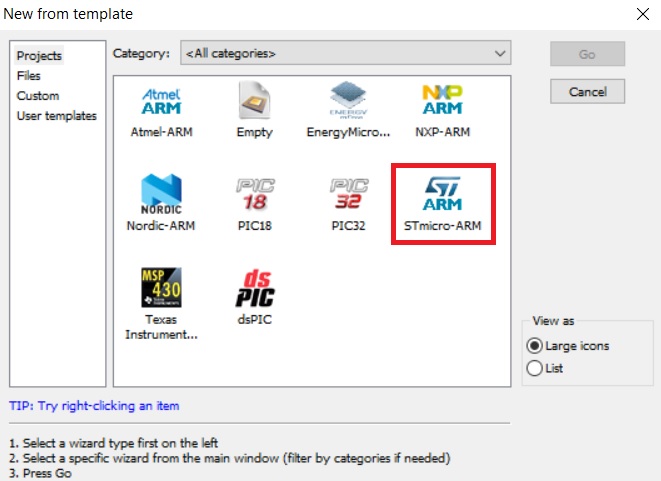
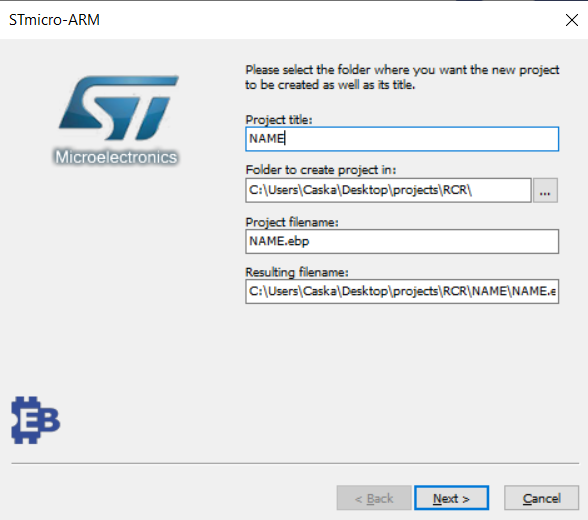
 

Рисунок 1.2 – Выбор линейки контроллера и названия проекта.

Далее потребуется выбрать микроконтроллер, с которым предстоит работать ([смотри список поддерживаемых устройств](#Поддержка)). Инструкция приведена на рисунках 1.3. На остальных окнах, не приведенных в текущей документации просто нажимать далее.

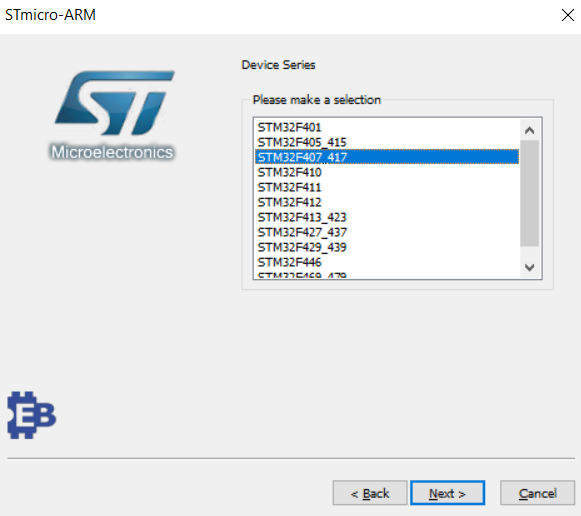
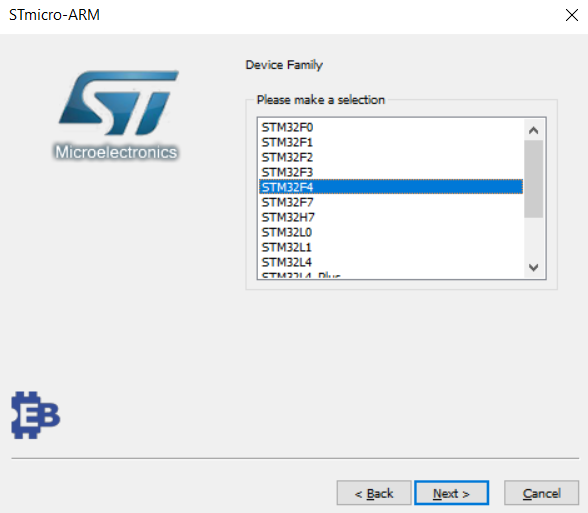


Рисунок 1.3 – выбор линейки микроконтроллера.

Далее, отключаем встроенные библиотеки SPL и HAL (рисунок 1.4). Потребуется только CMSIS определения, которые в любом случае будут добавлены в новых проект.

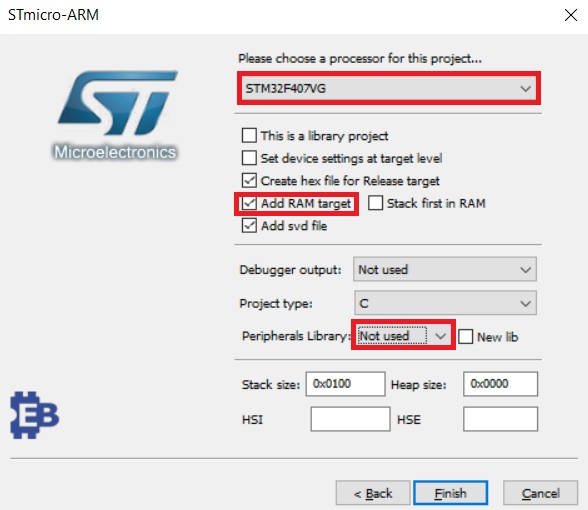


Рисунок 1.4 – Выбор модели, отключение встроенных библиотек.

Скопируйте файлы библиотеки FIL и любые другие файлы, необходимые для работы (рисунок 1.5). Итоговый вид проекта из проводника должен быть как на рисунке 1.6.

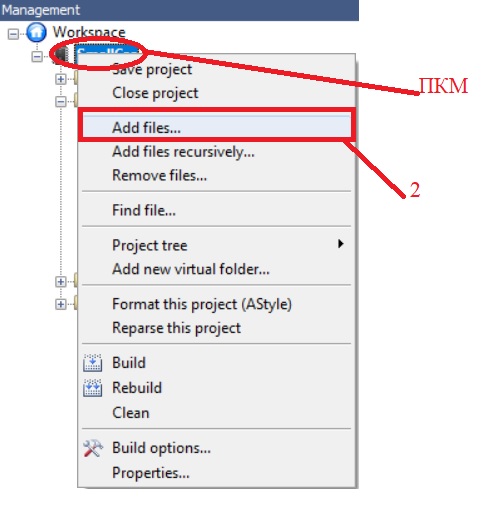


Рисунок 1.5 – Добавление новых файлов.

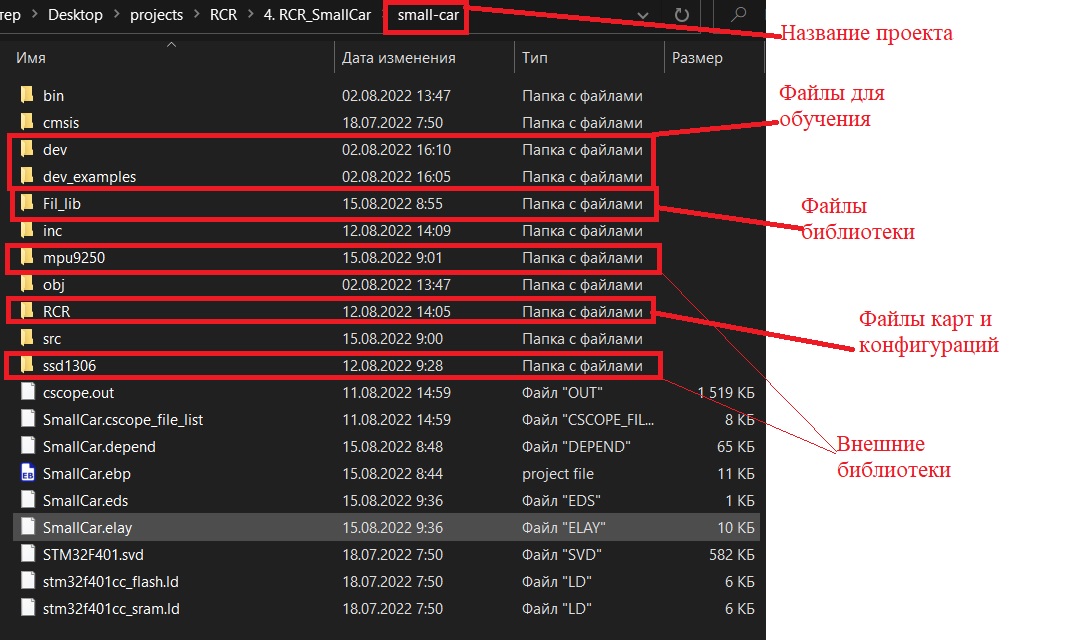


Рисунок 1.6 – добавление в пространство проекта файлов (проводник Windows).

Необходимо перейти в файл main.h и добавить подключение конфигурационного файла через *#include* как на рисунке 1.7.

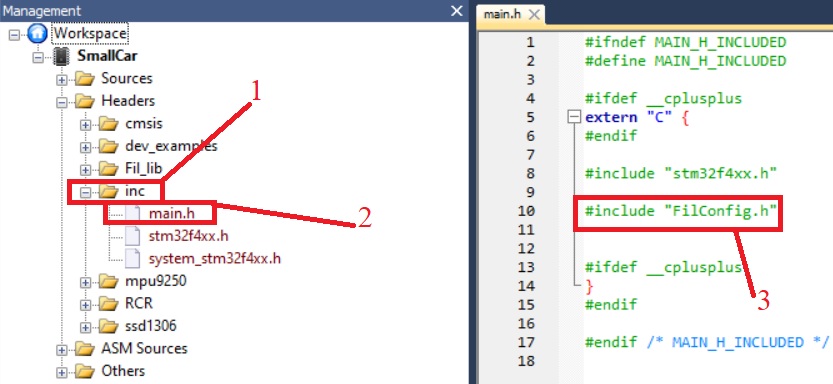


Рисунок 1.7 – Добавление библиотеки в проект.

**1.2.2** **Версия для опытных пользователей**

На данный момент версия библиотеки имеет несколько особенностей, которые необходимо учитывать перед непосредственной работой: Карта портов микроконтроллера, с примером которой можно ознакомиться в Листинге С, содержит в себе базовые определения из библиотеки FIL, таким образом получается замкнутый цикл, где конфигурация для библиотеки использует малую долю себя, чтобы расширить пространство предоставляемых функций. Данная опция облегчает создание проекта пользователем и ускорение времени разработки программного кода, однако потребовалось оставить участок GPIO открытым для доступа.

Несмотря на наличие большого количества предоставляемых функций, библиотека может содержать недоработки и ошибки, связанные с недостаточной проработанностью и защищенностью участков кода. Если таковые нашлись в явном виде, и они мешают разработке, просьба сообщить по контактам, приведенным во введении.

Для установки потребуется проделать следующие шаги:

1) Скопировать файлы библиотеки через GitHub или любым другим способом;

2) Создать проект в IDE EmBitz, настроив его под текущую линейку используемого контроллера (без подключения встроенных библиотек SPL, HAL);

3) Подключить линкер файл библиотеки, сделать в файле main.h;



4) Использовать готовые конфигурации или создать свою. Конфигурация представляет собой файл с настройками и картой портов. Для простоты хранения и подключения они хранятся в файле BoardSelection.h, структура которого изображена на рисунке 1.9.



Рисунок 1.9 Файл с проектами.

Для создания своей конфигурации необходимо установить аргумент \_\_configUSEBoards равным 4 и добавить в пустое пространство (*Development Board №4*) свои файлы. О написании своих конфигурационных файлов будет рассказано в следующих разделах.

**1.3** **История разработки**

13.10.2021 – начата работа над библиотекой.

23.10.2021 – разработаны функции инициализации для таймеров. Конфигурация портов ввода-вывода была использована заблаговременно, использована версия, разработанная в Ресурсном Центре Робототехники предположительно в 2013-2015 годах.

25.10.2021 – появление версии 1.0. Включала в себя инициализацию через ручное включение секторов кода, было разделение на уровни Default, Advanced и Developer.

31.01.2022 – добавление функций для работы с интерфейсом USART/UART. Некоторые изменения в таймерах и портах ввода-вывода.

02.02.2022 – добавлен файл-линкер FilConfig.h и добавлены исправления, не позволяющие конкретным API быть использованным без включения через линкер.

04.03.2022 – добавлены файлы настройки АЦП и контроллера прямого доступа к памяти(DMA). Исправление в файлах TIM.h, FilConfig.h.

13.05.2022 – изменена конфигурация линкера: теперь не нужно вручную изменять его настройки, для этого потребуется конфигурационный файл.

25.07.2022 – приезд после полевого тестирования библиотеки, добавлена поддержка и возможность подключить операционную систему микроконтроллеров FreeRTOS.

28.07.2022 – исправление работы функций, добавлена поддержка микроконтроллеров STM32F401CC и STM32F411VE.

11.08.2022 – добавление инициализации интерфейса I2C. Разработана функция сканирования шины и поиска устройств.

12.08.2022 – обновление документации по проекту.

15.08.2022 – начата работа по интеграции микроконтроллера STM32WL5. Тестовые участки кода для данного микроконтроллера добавлены на GitHub, создана ветка тестирования.

16.08.2022 – расширена конфигурация I2C. Добавлены функции записи и считывания данных с устройств.

**Список поддерживаемых контроллеров:**

- ARM Cortex-M4 (STM32F407VG, STM32F411VE, STM32F401CC);

*Примечание 1.1: Список будет дополняться по мере продвижения разработки. Проект библиотеки протестирован в среде EmBitz v2.30. В будущих версиях планируется протестировать библиотеку в среде Keil uVision. Альтернативный путь – попытка запуска и интеграции с более старыми версиями EmBitz (1.11 – 2.10).*

1. **[Конфигурация библиотеки](#АРаздел2)**

Конфигурационный файл – набор параметров, который используется для текущих настроек системы. В данном случае, предоставляет пользователю возможность инициализировать ключевые участки периферии микроконтроллера на максимально верхнем уровне абстракции машинно-ориентированных команд. Как было оговорено ранее, библиотека FIL использует саму себя, чтобы помочь программисту задать необходимую конфигурацию. По мере роста требований библиотека будет расширять набор предоставляемых инструментов и функций.

Обычно, конфигурационный сектор представлен двумя файлами. Первый – файл установления начальных настроек, его можно различить по приставкам Setup, Conf. Представляет собой набор параметров (рисунок 2.1), которые можно настраивать и, потенциально, использовать в рабочем коде для выполнения разных сценариев старта микроконтроллера. Второй файл был назван картой портов не случайно, в нем содержатся определения портов микроконтроллера и, соответственно, режимы работы. О нем речь пойдет далее. В готовых примерах содержатся, также, конфигурации, содержащие карту и конфигурацию в одном файле (например, MaxKit.h).

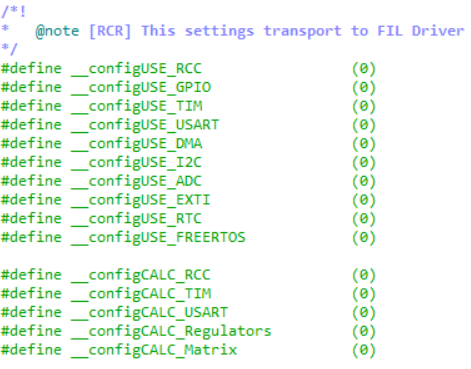


Рисунок 2.1 – Пример записи периферийных настроек в конфигурационном файле.

* 1. **[Файл-карта портов](#АРаздел21" \o "К содержанию)**

Изображенный на рисунке 2.2 пример написания карты представляет собой определения идентификаторов каждой подключаемой ножки микроконтроллера. Для конфигурации ID используется базовая функция библиотеки – *pin\_id(GROUP, PIN).* Исходя из перевода на русский язык, аргументы значат следующее: группа(group) используется для обозначения принадлежности к группе портов (у микроконтроллеров STM предполагается деление на A, B, C, D, E, F, G и т.д). Также встречается и вывод параметров различных счетчиков, например, как в случае с ENCODER1\_CNT или TIM\_CCR1.

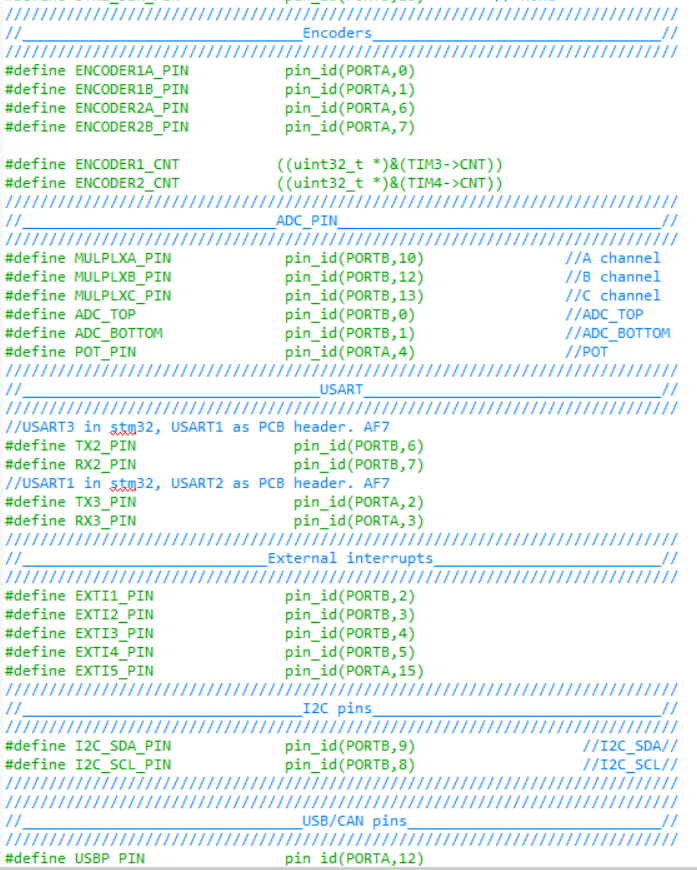


Рисунок 2.2 – пример написания карты портов микроконтроллера.

Чтобы добавить или изменить порт, необходимо определить требуемый. Например, потребуется порт А5 (группа А, номер 5). Тогда, в карте необходимо в области функции ID заменить аргументы на (PORTA,5). Названия после директивы #define является пользовательским, в стандартном наборе они имеют просто название PAx, а продвинутые пользователи используют другие названия для лучшего восприятия как на рисунке выше. Описание функции можно подробно изучить в списке макросов.

**2.2** [**Минимальная версия конфигурации**](#АРаздел22)

Минимальная версия конфигурации (MinimalKit) предназначена для обучения, настройки с целью отладки новых измерительных устройств, тестирования новых пользовательских решений. Минимальную комплектацию можно использовать в качестве безопасного режима, поскольку малое количество имеющихся в ней параметров удобно с точки зрения восприятия картины конфигурации в целом. Конфигурация точно также, как и в любых других, содержит чрезвычайно важные для настройки параметры и опциональные.