

ОТЛАДОЧНАЯ ПЛАТА DIP  
Руководство по эксплуатации

Редакция 1.4

Применимо к изделиям:

DIP-МК32-V4

## Оглавление

Введение.....	3
Основные сведения.....	4
Поддерживаемые программаторы.....	5
Начало работы с MIK32 IDE v1.2.2.....	6
Создание проекта.....	7
Использование примеров.....	8
Сборка прошивки.....	8
Загрузка прошивки.....	8
Запуск отладки.....	9
Расширенная конфигурация MIK32 IDE v1.2.2.....	10
Состав сборки.....	10
Добавление нового отладчика.....	10
Создание конфигураций сборки.....	11
Обновление компонентов сборки.....	12
Установка PlatformIO IDE в VS Code.....	14
Установка Visual Studio Code.....	14
Установка плагина PlatformIO.....	14
Подготовка компонентов для разработки на MIK32.....	15
Работа в среде PlatformIO.....	17
Создание проекта.....	17
Примеры программ для MIK32.....	19
Настройки platformio.ini.....	20
Загрузка программы на плату.....	21
Схемотехнические элементы отладочной платы.....	22
Режим загрузки MIK32.....	22
Кнопки управления.....	23
Светодиодные индикаторы.....	24
Питание и выводы.....	25
ПРИЛОЖЕНИЕ 1. Назначение выводов.....	27
ПРИЛОЖЕНИЕ 2. Габаритные размеры.....	28
ПРИЛОЖЕНИЕ 3. Пример схемы включения.....	29

## Введение

Отладочная плата МК32-DIP - сменный микроконтроллерный модуль для использования в прототипах устройств на базе МК32 Амур.

Плата содержит минимальную обвязку для запуска микроконтроллера и подходит для проектов различной сложности. Демонстрационные устройства и учебные комплекты АО Микрон используют в своем составе этот модуль.

Основное предназначение устройства – отработка схемотехники изделий при помощи макетных плат, изучения характеристик микроконтроллера, обучения основам программирования и электроники, быстрый выпуск прототипов устройств.

## Основные сведения

Плата содержит:

- Микроконтроллер МК32 «Амур» (К1948ВК018);
- Порты ввода-вывода микроконтроллера МК32;
- Кварцевый резонатор высокочастотный 32 МГц;
- Кварцевый резонатор низкочастотный 32768 Гц;
- Разъем JTAG 10-ти выводной для подключения внешнего отладчика;
- Разъем для выбора режима загрузки МК32;
- Тестовые точки, в т.ч. для подачи напряжения программирования однократно программируемой памяти VPROG;
- Два пользовательских светодиода, а также светодиод-индикатор подачи питания;
- Кнопку сброса и пользовательскую кнопку.

## Поддерживаемые программаторы

Для программирования платы рекомендуется использовать программаторы на основе микросхемы FT2232. Возможно использовать большинство программаторов, совместимых с OpenOCD и поддерживающих JTAG.

Список рекомендуемых программаторов:

1. Olimex ARM-USB-OCD-H;
2. Olimex ARM-USB-OCD;
3. ELJTAG.

## Начало работы с MIK32 IDE v1.2.2

Ссылку на скачивание программы можно найти на сайте <https://docs.mikron.ru/> в выпадающем списке **Среды разработки** в верхней части страницы.

Программа поставляется в виде ZIP-архива. Для начала работы требуется распаковать архив в любую папку. Среда разработки запускается файлом **start-ide.cmd**. После запуска появится меню выбора рабочих областей, в котором следует нажать кнопку **Launch**.

При первом запуске потребуются открыть проект-шаблон: перейти в **File->Open Projects From File System...** Откроется окно **Import Projects from File System or Archive**, в котором потребуется выбрать путь к проекту (Рисунок 1). Нажмите кнопку **Directory...** и в папке **{путь к сборке}\mik32-ide-v1-2-2\templates-project** выберете один из шаблонных проектов.

Проект **template-c-project** содержит в себе папку **framework**, содержащую компоненты для сборки. Библиотеки, стартовый файл, скрипт линковки хранятся внутри проекта.

Проект **template-c-project-shared** содержит ссылки на директорию **framework-shared** в корне сборки, библиотеки и другие компоненты хранятся вне проекта и разделены между подобными проектами.

После указания пути, в центральном списке будет отмечен выбранный проект. Для завершения открытия проекта следует нажать кнопку **Finish**.

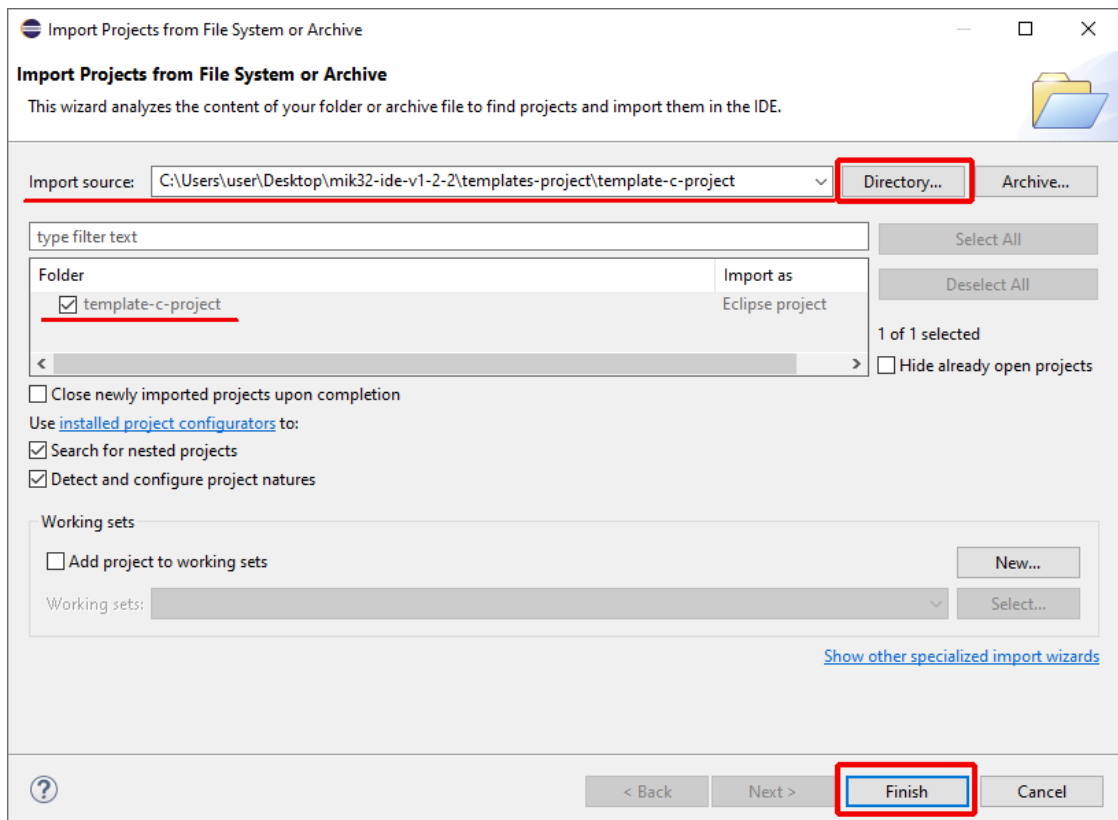


Рисунок 1. Окно открытия проекта

## Создание проекта

Создание нового проекта выполняется копированием проекта **template-c-project** и изменением названия проекта. Для этого в панели **Project Explorer** следует выбрать проект **template-c-project**, скопировать и вставить его горячими клавишами **Ctrl+C** или **Ctrl+V** или через контекстное меню пунктами **Copy** и **Paste**. При выполнении вставки откроется окно **Copy Project**, в котором следует задать название нового проекта (Рисунок 2).

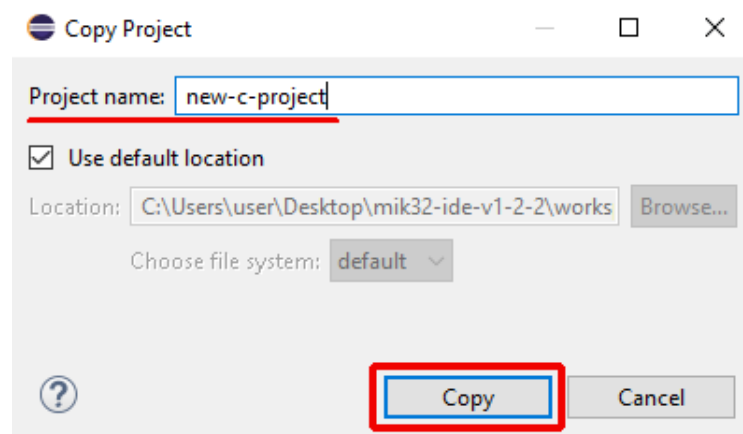



Рисунок 2. Окно копирования проекта


## Использование примеров

Чтобы начать работу с примерами, следует скопировать файл исходного кода примера из папки **src** соответствующего примера в папку **src** созданного проекта. Примеры находятся в папке **examples** рабочей области **workspace**.

## Сборка прошивки

При нажатии на кнопку **Build**  будет собран проект, папка или файл которого были выделены. По умолчанию выбрана конфигурация **Debug EEPROM**, в которой настроено сохранение программы во внутренней **EEPROM** памяти. В сборке доступны конфигурации **Debug RAM**, с сохранением программы в ОЗУ, и **Debug Flash**, с сохранением программы во внешнюю флеш память.

## Загрузка прошивки

Прошивка в микроконтроллер загружается с помощью механизма запуска внешних инструментов. Перед загрузкой, необходимо выбрать папку или файл соответствующего проекта. Для этого необходимо в верхней панели нажать на треугольник рядом с кнопкой **External Tools**  для открытия выпадающего меню внешних инструментов (Рисунок 3). Нажатие на пункт соответствующий нужному программатору запустит процесс загрузки прошивки.

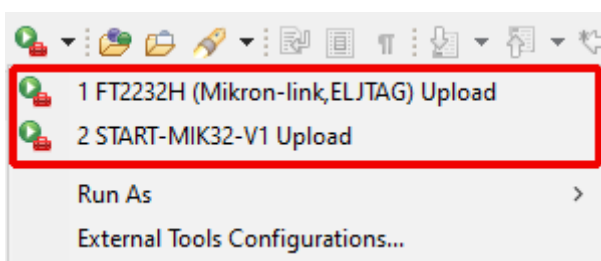



Рисунок 3. Выпадающее меню внешних инструментов

На данный момент есть готовые настройки для программаторов на основе микросхемы FT2232H с JTAG на канале 1 и программатора на плате START-МІК32-V1.

## Запуск отладки

Запуск отладки выполняется аналогично загрузке прошивки, запуском подготовленной отладочной конфигурации. Перед запуском отладки, необходимо выбрать папку или файл соответствующего проекта. Для этого необходимо в верхней панели нажать на треугольник рядом с кнопкой **Debug**

**Configurations...**  для открытия выпадающего меню конфигураций отладки. Нажатие на пункт соответствующий нужному программатору запустит процесс загрузки прошивки. Конфигурации разделены на отладочные и группы запуска, обозначаемые **Debug** и **Debug Upload** соответственно. Конфигурация, обозначенная **Debug** выполняет запуск отладки выбранной прошивки, без выполнения загрузки. Группы запуска **Debug Upload** сначала запускают загрузку, затем запускают отладку.


# Расширенная конфигурация MIK32 IDE v1.2.2

## Состав сборки

В корневом каталоге сборки можно найти следующие директории:

- build-tools – используется программа make для сборки;
- curl – используется для скачивания svd описания из ветки main репозитория;
- eclipse – среда разработки eclipse ide;
- examples – набор примеров для копирования исходников в проекты;
- framework-shared – компоненты для сборки, подключаемые к проектам;
- git – клиент для обновления компонентов для сборки и примеров из репозитория;
- openocd – для подключения к контроллеру для загрузки прошивки и отладки;
- riscv-gcc – набор инструментов для компиляции;
- templates-debug-upload-configurations – конфигурации загрузки и отладки для импорта в новые рабочие области;
- templates-project – шаблоны проектов с настройками сборки;
- uploader – программа для загрузки прошивок в память микроконтроллера;
- workspace – рабочая область по умолчанию.

## Добавление нового отладчика

Для добавления нового отладчика перейдите в раздел **Debug Configurations**, выберите одну из существующих конфигураций в разделе **GDB OpenOCD Debugging** в списке слева, например, **FT2232H (Mikron-link,ELJTAG) Debug**. Затем, в верхней части списка над строкой поиска, нажмите кнопку дублирования . Выберите созданную конфигурацию. На вкладке **Debugger** в поле **Config options** измените путь в первом аргументе **-f** на путь, соответствующий скрипту необходимого

отладчика (Рисунок 4). Скрипты для многих отладчиков уже есть в составе OpenOCD. Например, для выбора отладчика J-Link, пропишите следующий путь: `"${eclipse_home}/../openocd/openocd/scripts/interface/jlink.cfg"`.

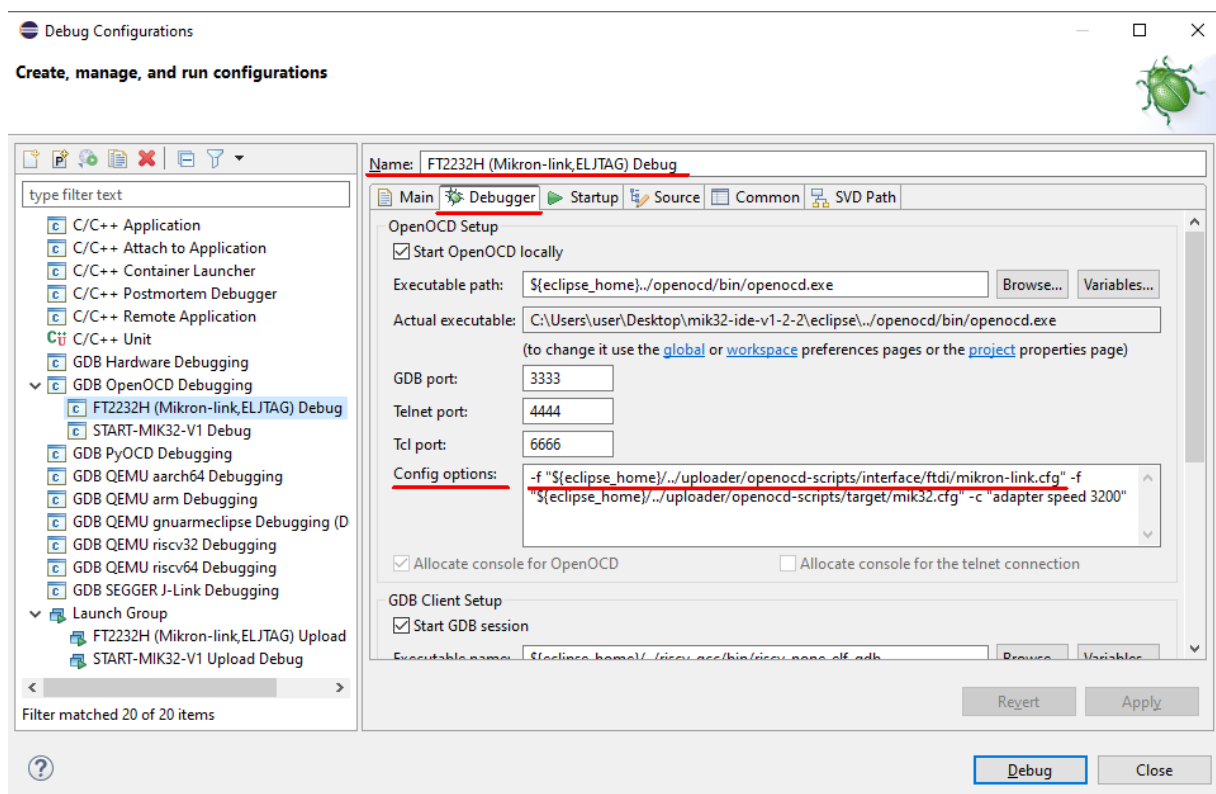


Рисунок 4. Изменение отладчика в конфигурации отладки

Помимо настроек отладки, аналогичным образом нужно дублировать конфигурацию внешнего инструмента для загрузки прошивки (см. раздел Загрузка прошивки) в окне настроек внешних инструментов **External Tools**, и заменить скрипт для отладчика на вкладке **Main** в поле **Arguments**. Затем, в меню отладочных конфигураций **Debug Configurations** следует продублировать группу запуска **Upload Debug** и заменить конфигурацию загрузки и конфигурацию отладки. Это можно сделать в меню **Edit Launch Configuration**, которое открывается по кнопке **Edit**.

## Создание конфигураций сборки

Для создание дополнительной конфигурации сборки выберите любой папку или файл, относящийся к проекту, и выберите в строке меню среды разработки **Project**→**Build Configurations**→**Manage...** Откроется окно управления конфигурациями сборки, в котором можно выбрать активную

конфигурацию, удалить и создать новую. Для создания новой конфигурации нажмите кнопку **New...** В поле **Name** введите название новой конфигурации, а в группе **Copy settings from** выберите пункт **Existing configuration** и конфигурацию, настройки которой будут скопированы в созданную конфигурацию.

Чтобы изменить настройки конфигурации сборки выберите любой папку или файл, относящийся к проекту, и выберите в строке меню среды разработки **Project→Properties**. В дереве слева выберите пункт **C/C++ Build→Settings**. В правой части окна откроются настройки конфигурации сборки. В верхней части окна выбирается настраиваемая конфигурация.

Чтобы изменить скрипт линковки, например, для изменения памяти, в которой будет размещаться прошивка, на вкладке **Tool Settings** настроек конфигурации, в дереве выберите **GNU RISC-V Cross Linker→General**. В поле **Script files** можно изменить скрипт линковки. В текущем **framework** имеются варианты **eeeprom.ld**, **spifi.ld**, **ram.ld**.

## Обновление компонентов сборки

Компоненты для сборки программ, находятся в папке **framework-shared** в корневой директории сборки. Для их обновления из GitFlic репозитория **MikronMIK32/framework-mik32v2-sdk** следует запустить скрипт **update-framework.bat**.

Компоненты для сборки программ в папках проектов находятся в директории **framework**. Для их обновления из GitFlic репозитория **MikronMIK32/framework-mik32v2-sdk** следует запустить скрипт **update-framework.bat** в каталоге проекта.

Примеры находятся в папке **examples** в корневой директории сборки. Для их обновления из GitFlic репозитория **MikronMIK32/framework-mik32v2-sdk** следует запустить скрипт **update-framework.bat**.

Программа для прошивки памяти контроллера находится в папке **uploader** в корневой директории сборки. Для его обновления следует перейти по ссылке <https://gitflic.ru/project/mikron-mik32/mik32-uploader/release/latest> и скачать архив **mik32-uploader-vX.Y.Z-win32-x64.zip**. Содержимое папки **uploader** нужно удалить, затем содержимое директории **mik32\_upload** скачанного архива перенести в каталог **uploader**.

Файл описания регистров `mik32v2.svd` находится в рабочей области `workspace`. Для его обновления следует запустить скрипт `update-mik32v2-svd.bat`.

# Установка PlatformIO IDE в VS Code

## Установка Visual Studio Code

Скачайте и установите среду по ссылке:

<https://code.visualstudio.com/download>.

## Установка плагина PlatformIO

Для разработки ПО под микроконтроллеры требуется установить плагин PlatformIO.

Выберите пункт Extensions или нажмите сочетание клавиш Ctrl+Shift+X. Найдите его в списке плагинов и нажмите Install (Рисунок 1).

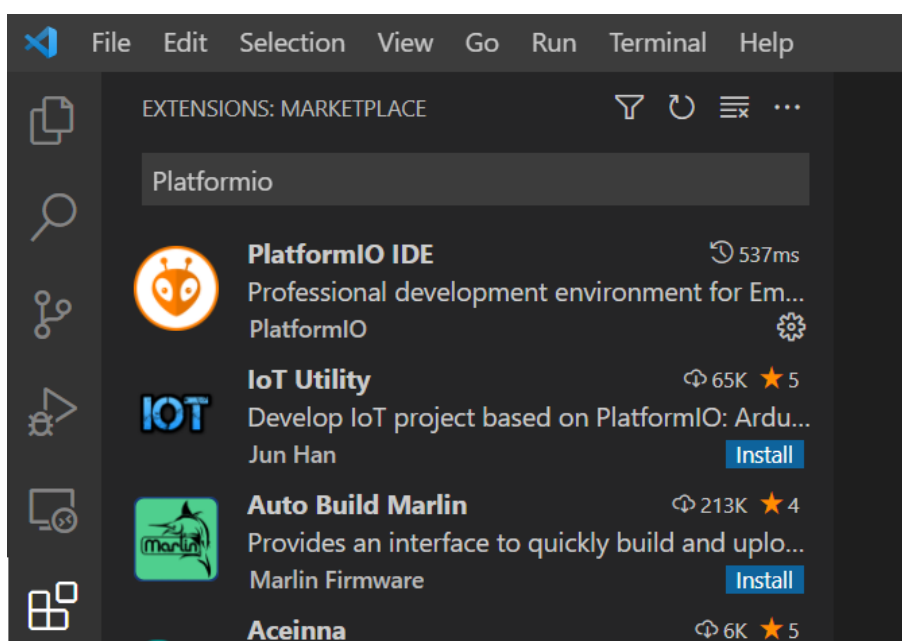


Рисунок 1 – Установка плагина PlatformIO

Дождитесь окончания установки (статус отображается на панели снизу) и перезапустите программу.

Теперь при запуске через некоторое время в панели инструментов (слева) появится логотип Platformio (Рисунок 2).

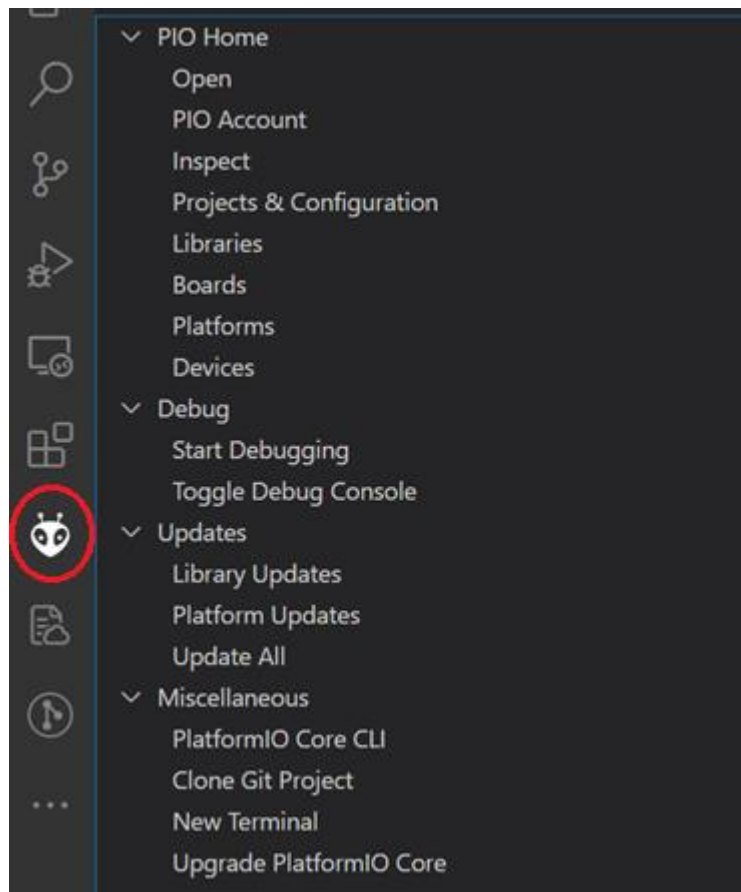


Рисунок 2 – Логотип PlatformIO в боковом меню

## Подготовка компонентов для разработки на MIK32

После установки расширения потребуется распаковка архива по пути %USERNAME%\platformio (для Windows) или ~/.platformio (для Linux). В Windows при установке VS Code на уровне системы (System Installer) директорию можно найти по пути C:\.platformio. В Linux каталоги, название которых начинается с точки, считаются скрытыми, поэтому потребуется включить отображение скрытых файлов.

Необходимо скачать архив из последнего релиза по адресу <https://gitflic.ru/project/mikron-mik32/mik32-platformio/release/latest>. Последние релизы находятся в верхней части экрана, в его карточке необходимо найти группу Assets и в списке выбрать элемент с названием mik32-platformio-{версия}.zip. Нажатие по этому элементу запустит скачивание архива. Скачивать файлы, обозначенные как Source Code не следует, поскольку в них будут отсутствовать подмодули пакетов. Затем нужно распаковать содержимое архива по вышеуказанному пути.

После распаковки всех архивов должна получиться следующая структура .platformio:

1. packages
  - 1.1.framework-mik32v2-sdk
    - 1.1.1. hal
    - 1.1.2. shared
  - 1.2.tool-mik32-uploader
2. platforms
  - 2.1.mik32

Перезапустите **VS Code**. Теперь у вас подготовлена среда для программирования MIK32.

# Работа в среде PlatformIO

## Создание проекта

Дождитесь появления значка PlatformIO в левой части окна и нажмите на него. Откроется панель, в которой следует нажать на Open. Откроется вкладка PIO Home, нажмите на кнопку New Project для создания нового проекта (Рисунок 3).

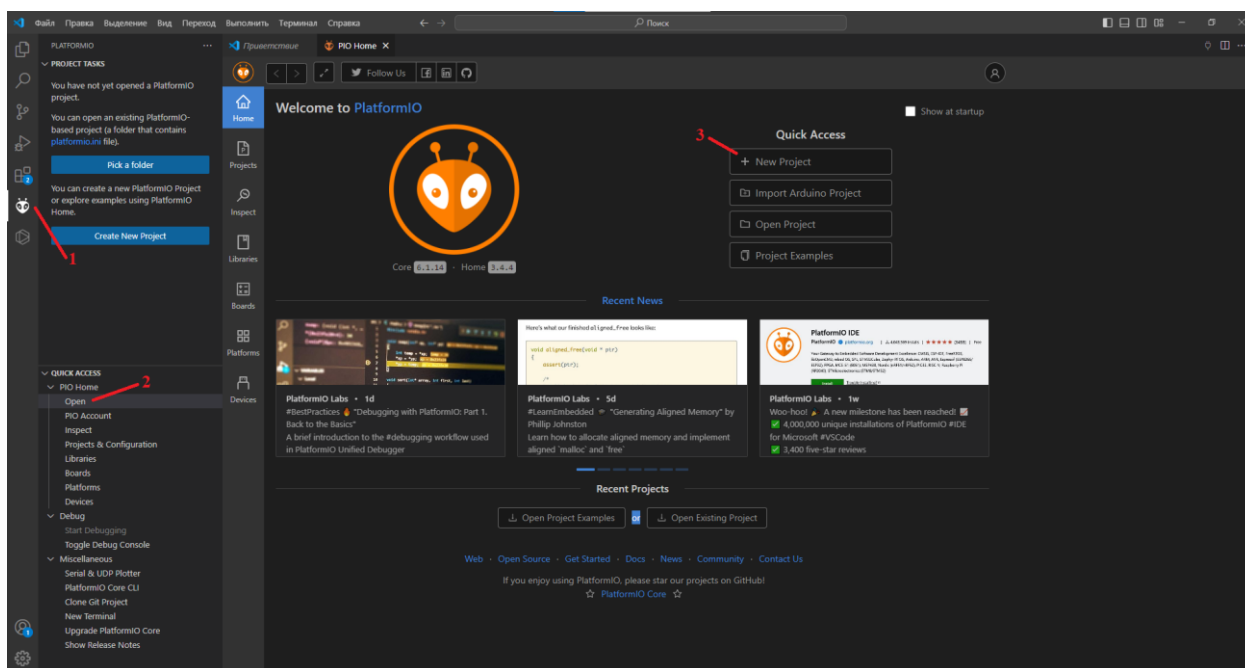


Рисунок 3 – Открытие меню создания проекта

В диалоговом окне Project Wizard в поле Name введите имя проекта, в выпадающем списке Board введите mik32 и выберите START-MIK32-V1 (Mikron), нажмите Finish (Рисунок 4).

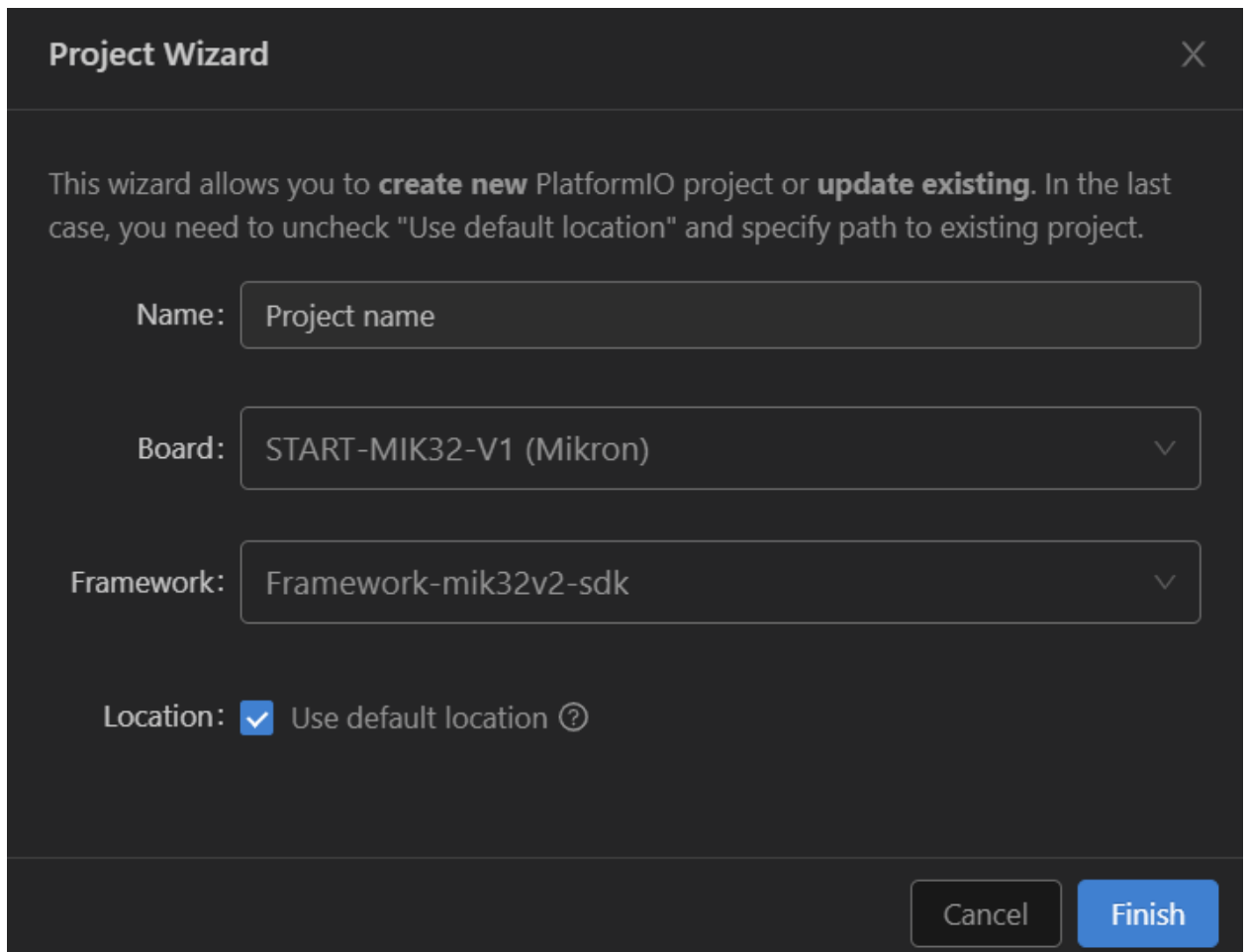


Рисунок 4 – Окно создания проекта

При первом запуске Platformio скачает набор для разработки под RISC-V. Для этого требуется подключение к интернету, будет скачано около 1 ГБ. В это время platformio скачивает такие пакеты как toolchain-riscv, tool-openocd и tool-scons в папку C:\Users\ИМЯ-ПОЛЬЗОВАТЕЛЯ\.platformio\packages. Дождитесь завершения создания проекта.

Для создания файла будущей программы в левой части панели проводника нажмите правой кнопкой мыши на папку src, выберите «Создать файл», введите название файла, например, main.c, и нажмите Enter (Рисунок 5).

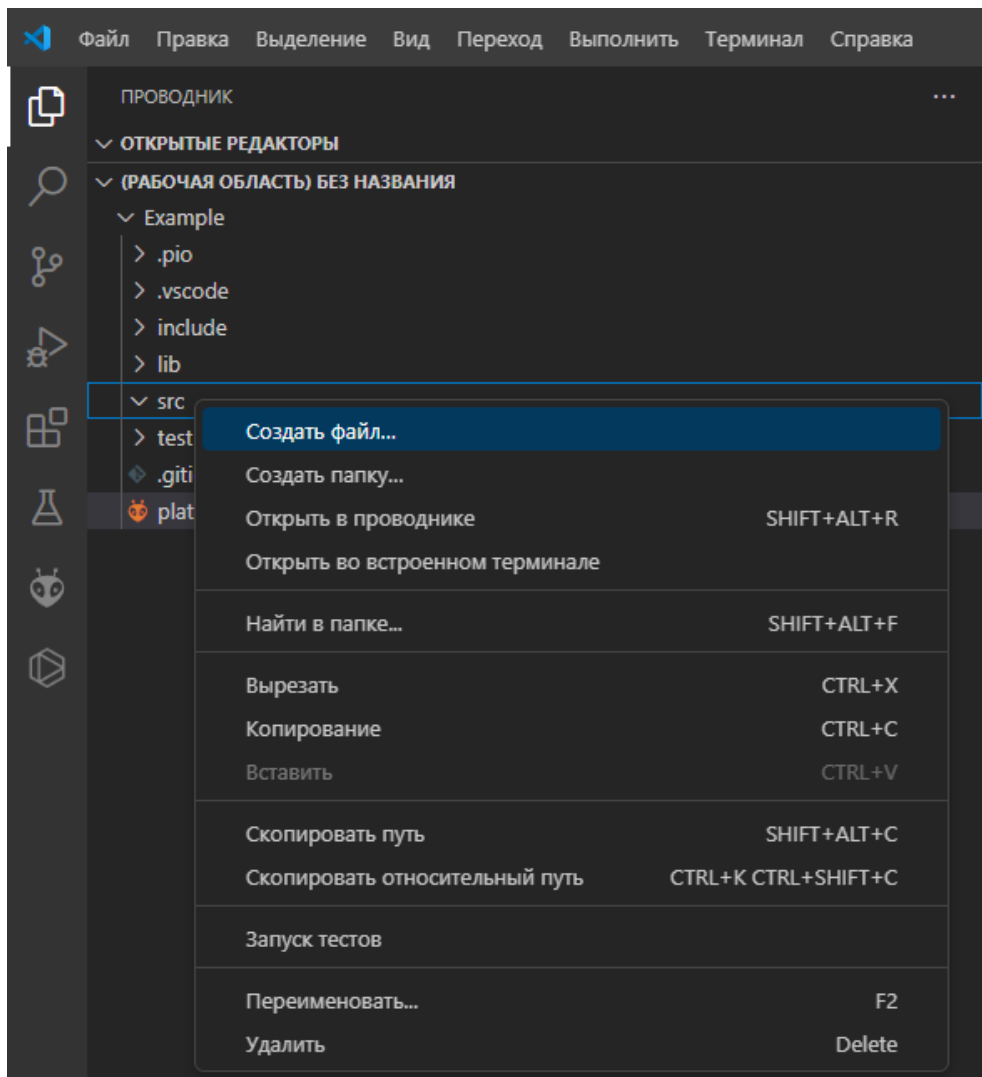


Рисунок 5 – Создание файла в проекте PlatformIO

## Примеры программ для МК32

Ознакомьтесь с примерами и параметрами конфигурационного файла platformio.ini, который находится в каждом проекте platformio.

Ссылка на примеры: <https://gitflic.ru/project/mikron-mik32/mik32-examples>.

В файле platformio.ini содержатся параметры проекта. Пример настроек сборки программы для внешней flash памяти представлен ниже.

По-умолчанию выбирается настройка программатора mikron-link, которая совместима с программатором ELJTAG

```
; Пример комментария  
[env:mik32v2]  
platform = MIK32  
board = mik32v2  
framework = framework-mik32v2-sdk  
board_build.ldscript = eeprom
```

## Настройки platformio.ini

В корне проекта расположен файл platformio.ini который содержит настройки проекта. Основные настройки приведены ниже:

**upload\_protocol = mikron-link**

**board\_build.ldscript** - выбор ld скрипта;

Значение по умолчанию: eeprom;

Возможные значения:

- eeprom;
- spifi;
- ram.

Пример использования: board\_build.ldscript = spifi

**upload\_speed** - частота тактового сигнала интерфейса jtag, кГц.

Значение по умолчанию: 500;

Пример использования: upload\_speed = 3200

**board\_build.f\_cpu** - переопределение частоты основного генератора тактовой частоты, Гц в формате длинного числа Си.

Значение по умолчанию: 32000000L;

Пример использования: board\_build.f\_cpu = 4000000L

**board\_upload.maximum\_size** - переопределение максимального размера программы, байты.

Размер программы вычисляется Platformio по суммарному размеру секций .text и .data. maximum\_size устанавливается по объему памяти, в которой будет храниться программа.

Значение по умолчанию: 8192;

Пример использования: board\_upload.maximum\_size = 4194304.

Больше настроек можно найти в официальной документации проекта: <https://docs.platformio.org/en/latest/projectconf/sections/env/index.html#options>

## Загрузка программы на плату

Перед загрузкой программы убедитесь, что проект выбран в качестве активного. Для этого в platformio toolbar, расположенном внизу, выберите "switch platformio project environment". В открывшемся выпадающем списке выберите нужный проект. Затем нажмите upload (Рисунок 6).

При загрузке в ОЗУ в случае успешной загрузке в конце будет надпись вида «downloaded xxx bytes in...».

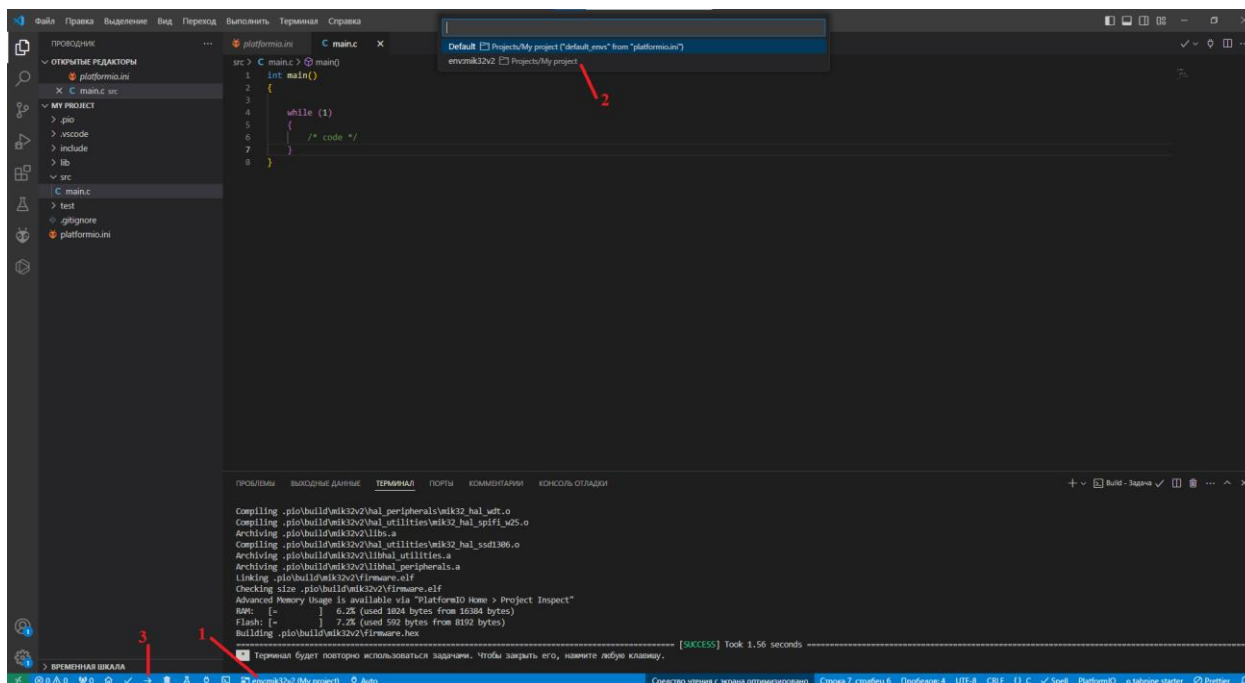


Рисунок 6 – Выбор проекта и загрузка программы в микроконтроллер

Больше информации о MIK32 на сайте <https://wiki.mik32.ru>

# Схемотехнические элементы отладочной платы

## Режим загрузки МК32

На плате выводы BOOT (B0 и B1) определяют режим загрузки МК32 при включении питания или при внешнем сбросе (RESET) (Рисунок 7).

Состояние B0	Состояние B1	Режим загрузки
0	0	EEPROM
0	1	SPIFI
1	0	RAM
1	1	Зарезервировано

где 0 – низкий логический уровень, 1 – высокий логический уровень.

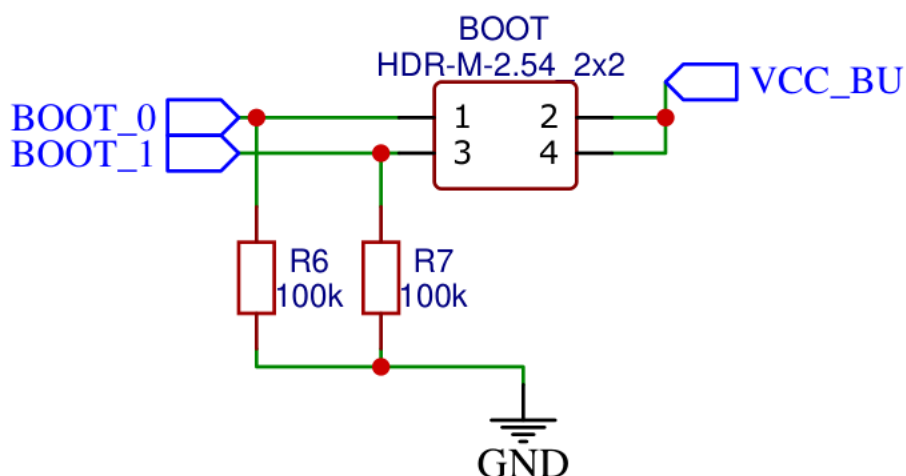


Рисунок 7 – Схема выбора режима загрузки

По умолчанию выводы B0 и B1 подтянуты к земле, что соответствует режиму загрузки из EEPROM. Для изменения режима загрузки следует установить перемычку между одним из выводов BOOT и выходом внутренней линии питания батарейного домена 3.3В - VCC\_BU (Рисунок 8).

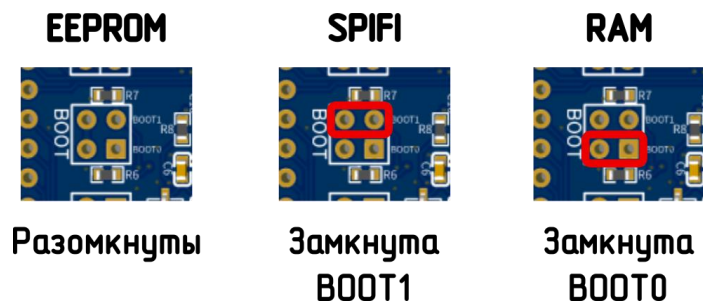


Рисунок 8 – Выбор режима загрузки перемычкой

## Кнопки управления

На плате расположены три кнопки:

- Пользовательская кнопка USER\_V с подтяжкой к питанию, которая подключена к выводу MIK32 PORT0\_8 и замыкает его на землю;
- Кнопка RESET для внешнего сброса MIK32.

Схема подключения кнопок показана на Рисунок 9

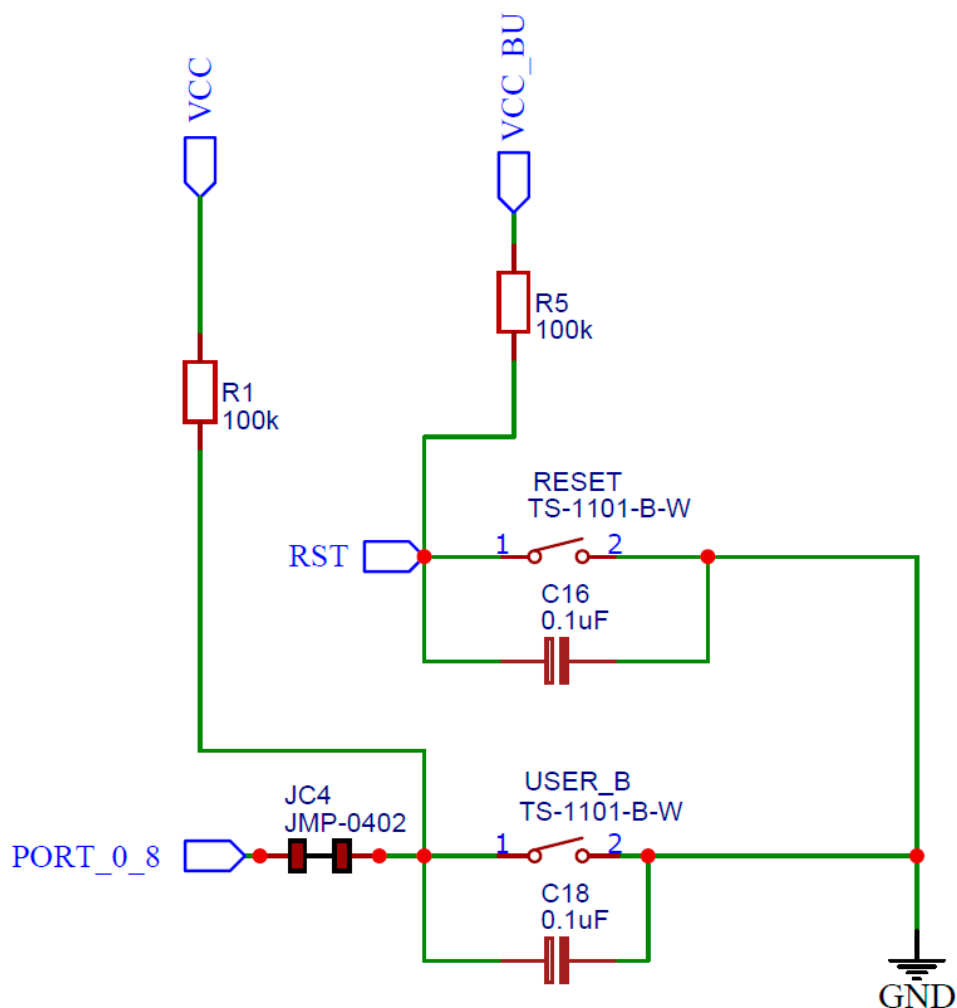


Рисунок 9 – Схема подключения кнопок программирования

## Светодиодные индикаторы

Два пользовательских светодиода VD3 и VD4 подключены к выводам МК32 PORT0\_3 и PORT1\_3 через перемычку и загораются при подаче на них логической «1» в режиме GPIO. Выводы PORT0\_3 и PORT1\_3 можно использовать в качестве вывода ШИМ сигнала, который формируется с помощью TIMER32\_1 и TIMER32\_2 соответственно.

Светодиод POWER (обозначение POWER1) сигнализирует о наличии питания платы.

Схема подключения светодиодных индикаторов показана на Рисунок 10.

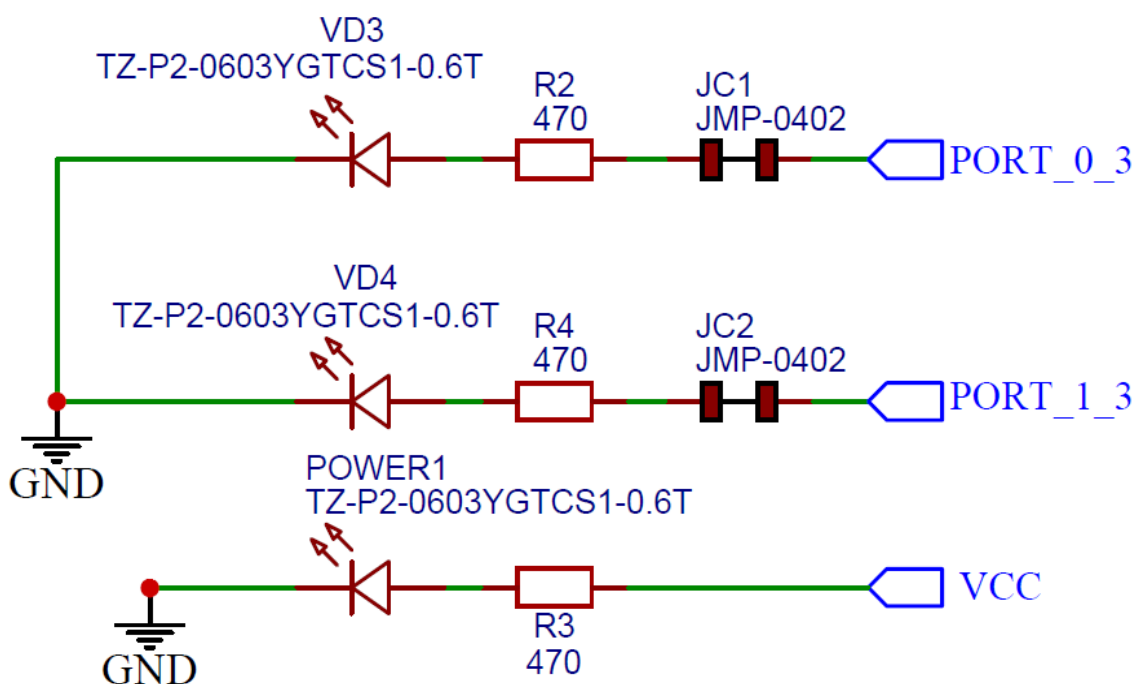


Рисунок 10 – Схема подключения светодиодных индикаторов

## Питание и выводы

Напряжение питания МІК32 на выводе VCC составляет 3,3 В, допустимый диапазон (2,97...3,63) В. Вывод AVCC МІК32 - напряжение питания аналоговых блоков по умолчанию соединен через перемычку с выводом VCC. Возможно подать на вывод AVCC напряжение отличное от VCC. Для этого следует разрезать перемычку JC3 (обратная сторона платы) и подать напряжение на вывод платы AVCC величиной 3,3 В, допустимый диапазон (3,15...3,45) В (Рисунок 11).

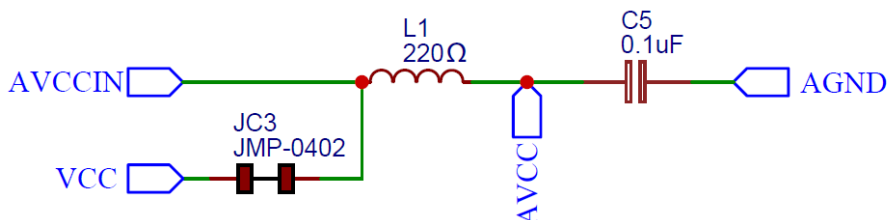


Рисунок 11 – Схема подачи питания аналоговых блоков

Для чтения данных из однократно-программируемой памяти (ОТР) МІК32 вывод VPROG на плате соединен с VCC через диодную сборку. Если

требуется записать данные в память, на контактную группу J2 (вывод VPROG) следует подать напряжение программирования OTP 8.0В (Рисунок 12).

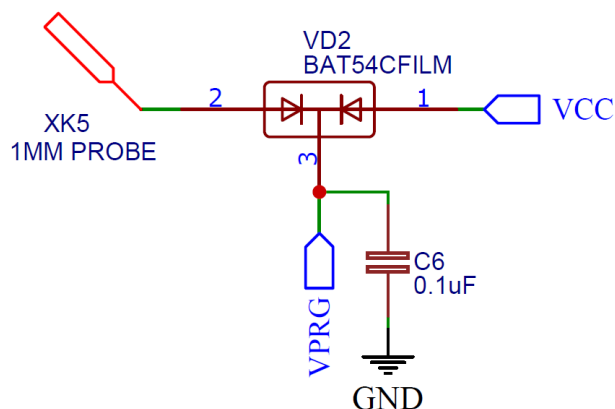


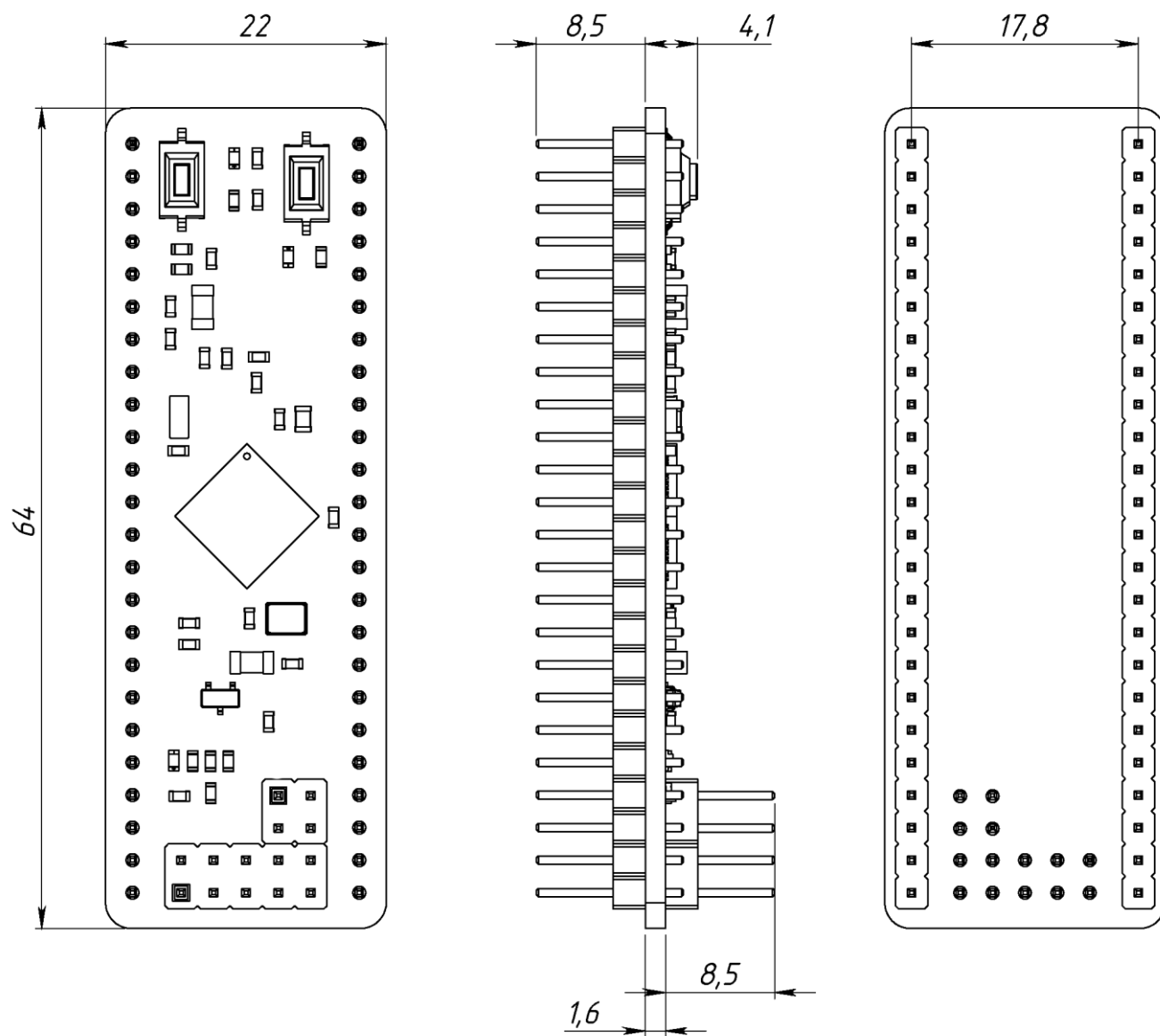
Рисунок 12 – Схема подачи питания однократно программируемой памяти

Запасное питание батарейного домена на выводе VBAT предназначено для питания батарейного домена МК32 величиной 3,3 В, допустимый диапазон (2,5...3,63) В.

Кроме этого, на плате выведена линия RTC\_ALARM (вывод R\_AL), которая сигнализирует о срабатывании будильника RTC, а также линия External Wakeup (вывод EW), которая подтянута к земле для выхода из режимов низкого энергопотребления.



## ПРИЛОЖЕНИЕ 2. Габаритные размеры



# ПРИЛОЖЕНИЕ 3. Пример схемы включения

