ОТЛАДОЧНАЯ ПЛАТА RADIANT MIK32

Руководство по быстрому старту

17 ЯНВАРЯ 2025 Г.
RADIANT.SU
Г МОСКВА

Оглавление

Введение

Отладочная плата **RADIANT MIK32** предназначена для ознакомления с микроконтроллером К1948ВК018 АДКБ.431290.418ТУ (MIK32 Амур), производства АО «Микрон». Данный продукт нацелен на помощь в изучении архитектуры микроконтроллера и не предназначена для встраивания в конечные устройства. На плате установлен JTAG отладчик и виртуальный последовательный порт, соединенный с выводами UART1. Данный продукт совместим с платами расширения стандарта ARDUINO-UNO. Для удобства работы с микроконтроллером все порты ввода-вывода продублированы на штыревые разъемы по краям платы.

Таблица 1 Основные сведения

Характеристика	Значение
Микроконтроллер	К1948ВК018
Объем флэш памяти для приложения	8Мб, QSPI
Отладочный порт	JTAG
Интерфейс подключения платы	USB
Возможность программирования внешних схем	Да
Возможностью отделения от целевой платы	Да
Встроенный виртуальный COM порт для отладки	Да
Встроенный регулятор 3.3В	MIK1117
Порт 485	MIK3485
Разъем расширения ARDUINO-UNO	Да
Портов ввода-вывода на штыревых разъемах, шт	40 (все)
Кнопки управления	2
Светодиоды для отладки	2
Кварцевый резонатор высокочастотный	32МГц
Кварцевый резонатор низкочастотный	32768Гц
Габариты	68мм*93мм

Системные требования

Для работы с микроконтроллерной платой потребуется следующее техническое оснащение:

1. Компьютер на базе Windows 10 или новее.
    a. Права администратора, чтобы иметь возможность устанавливать новое ПО
    b. Оперативная память – не менее 8Гб
    c. Свободное место на диске – 50Гб
    d. Свободный порт USB
2. Кабель с разъемом USB-type C

Краткое описание платы

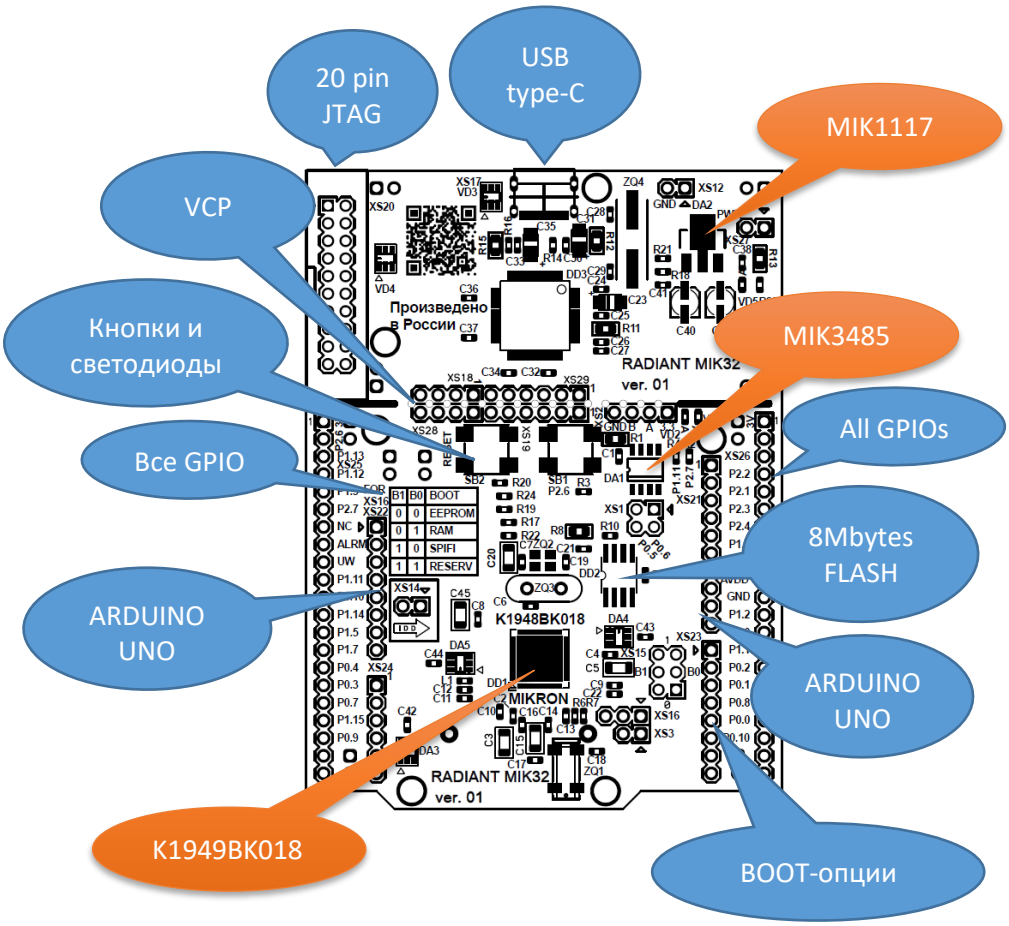


Рис. 1 Кратное представление отладочной платы и ее блоков

Для основной работы с платой достаточно простого подключения кабеля USB.

Питание резервного домена от батарейки (для питания часов реального времени RTC в режиме отключения основного питания) для плат версий 0.3 и ранее не предусмотрено.

Vprg – вход для подачи повышенного напряжения питания для программирования OTP памяти 256 бит. Если эта память не используется, то оставить этот вход не подключенным. Для чтения этой памяти следует подавать 3.3В.

Джампер на разъеме XS14 “IDD” можно удалить и, подключив в разрыв амперметр, измерить потребление тока микроконтроллером (+ внешняя память).

Установка необходимого ПО

Скачайте установочный файл https://disk.yandex.ru/d/WVTSVBOD6Jj6nw и установите среду разработки в папку C:\MikronIDE или в удобное место. Рекомендуется в корневую директорию C:\.

Установка драйвера WinUSB

Подключите плату к компьютеру кабелем USB. Дождаться пока автоматическая установка драйверов завершится.

Сначала рекомендуется удалить все драйвера

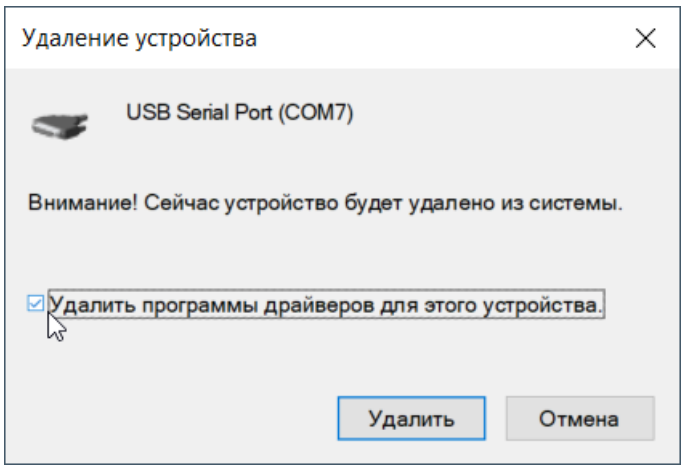


Рис. 2 Удаление автоматически установленных драйверов

Таким образом, чтобы в установленных устройствах в диспетчере устройств появилось два неопознанных компонента:

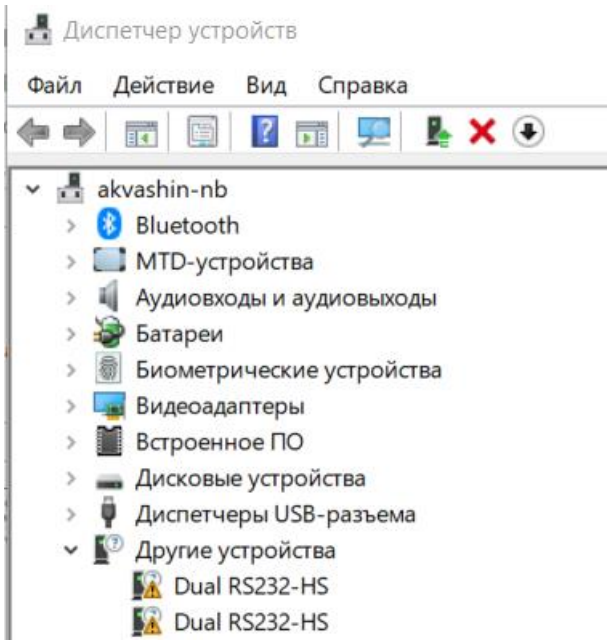


Рис. 3 Устаройста отладочной платы в Диспетчере устройств

Активируйте программу Zadig из “MikronIDE\tools\zadig-2.4.exe”. Ссылка на последнюю версию: https://zadig.akeo.ie.

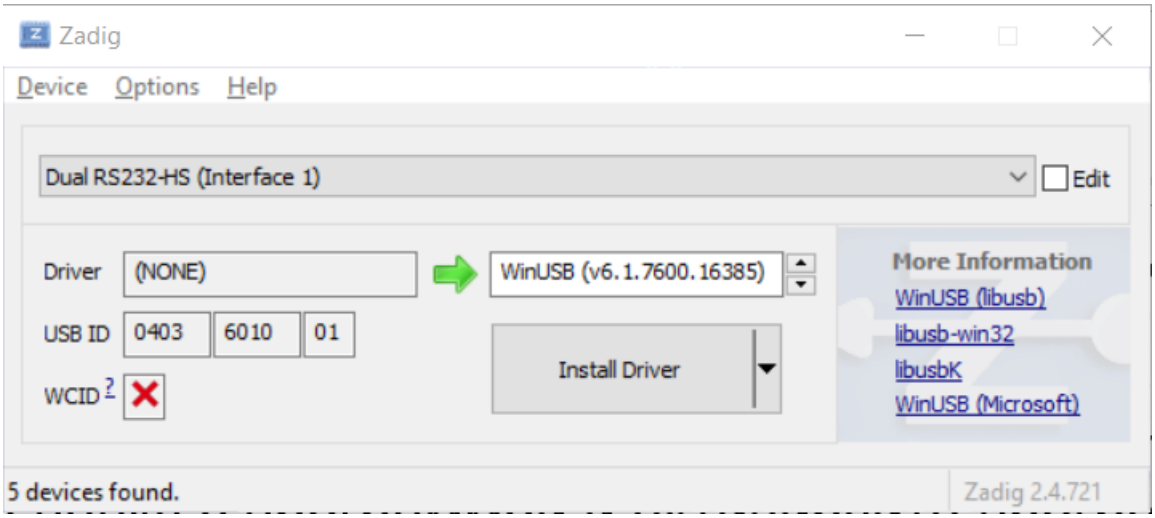


Рис. 4 Установка драйвера для JTAG

Подключите плату. В меню Options нажмите на **List All Devices**. В выпадающем списке выберите устройство с названием:

Dual RS232-HS (Interface 1)

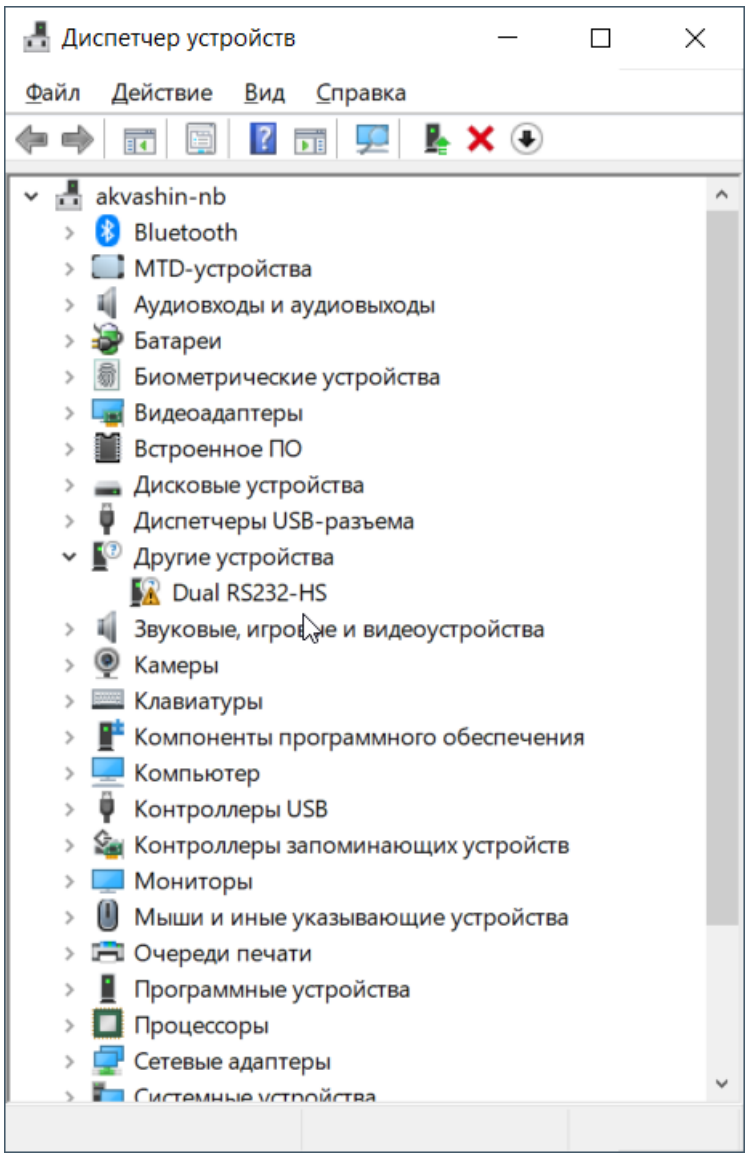
Затем проверьте что выбран один из совместимых драйверов, например, WinUSB или libusbK или libusb-win32 в белом поле ввода (выбирать кнопками со стрелками) и нажмите кнопку **Replace Driver** или **Install Driver**.

Для виртуальных машин Windows рекомендуется libusbK.

После установки **отключите** устройство и **подключите заново для начала работы**. Обновите список. На сером поле должен быть обозначен новый драйвер.

Установка драйвера VCP FTDI

Установка виртуального COM-порта. На канал Dual RS232-HS (Interface 0) – нужно установить вручную драйвер


Диспетчер устройств
Файл
Действие
Вид
Справка
akvashin-nb
Bluetooth
MTD-устройства
Аудиовходы и аудиовыходы
Батареи
Биометрические устройства
Видеоадаптеры
Встроенное ПО
Дисковые устройства
Диспетчеры USB-разъема
Другие устройства
Dual RS232-HS
Звуковые, игровые и видеоустройства
Камеры
Клавиатуры
Компоненты программного обеспечения
Компьютер
Контроллеры USB
Контроллеры запоминающих устройств
Мониторы
Мыши и иные указывающие устройства
Очереди печати
Программные устройства
Процессоры
Сетевые адаптеры

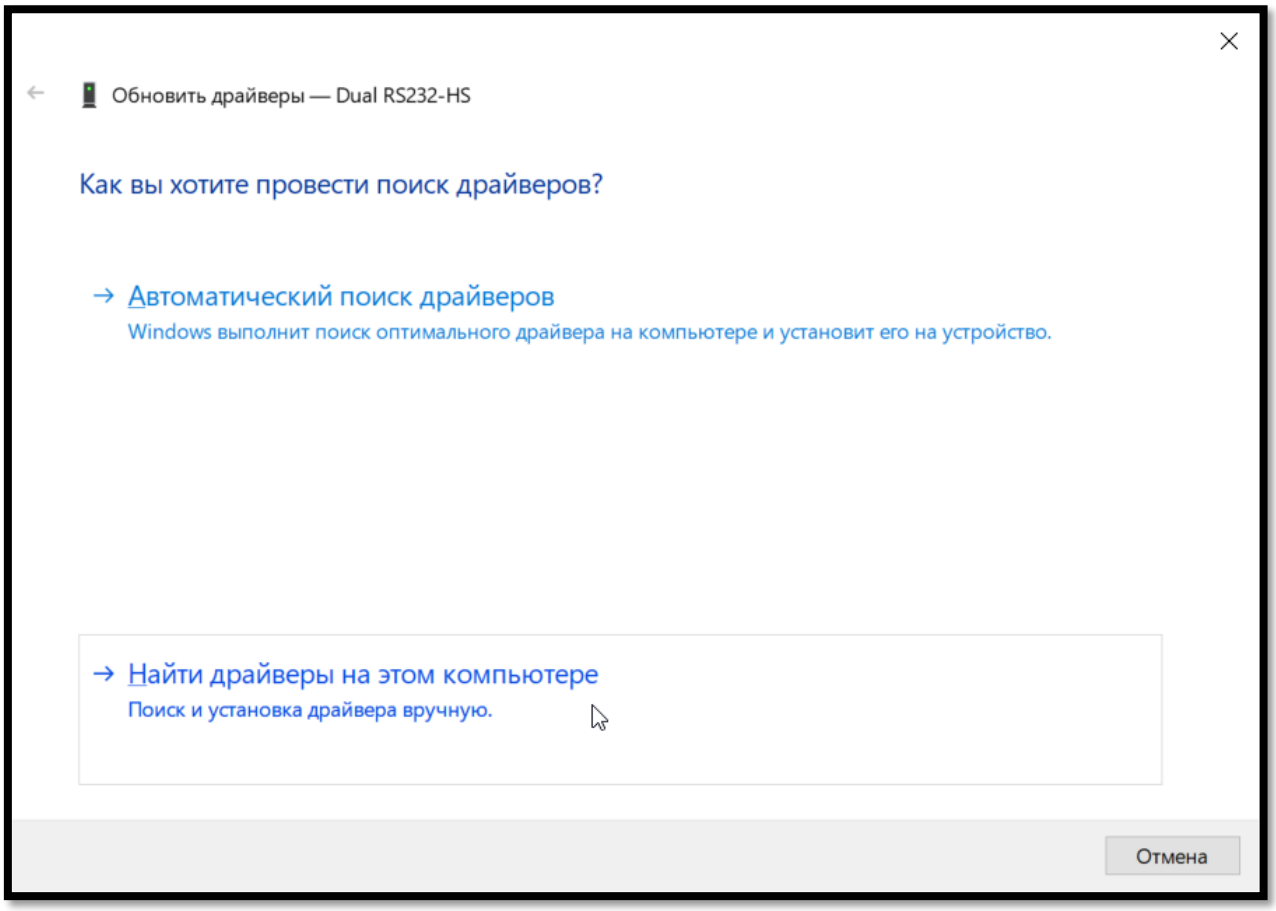


Рис. 5 Установка драйвера для VCP

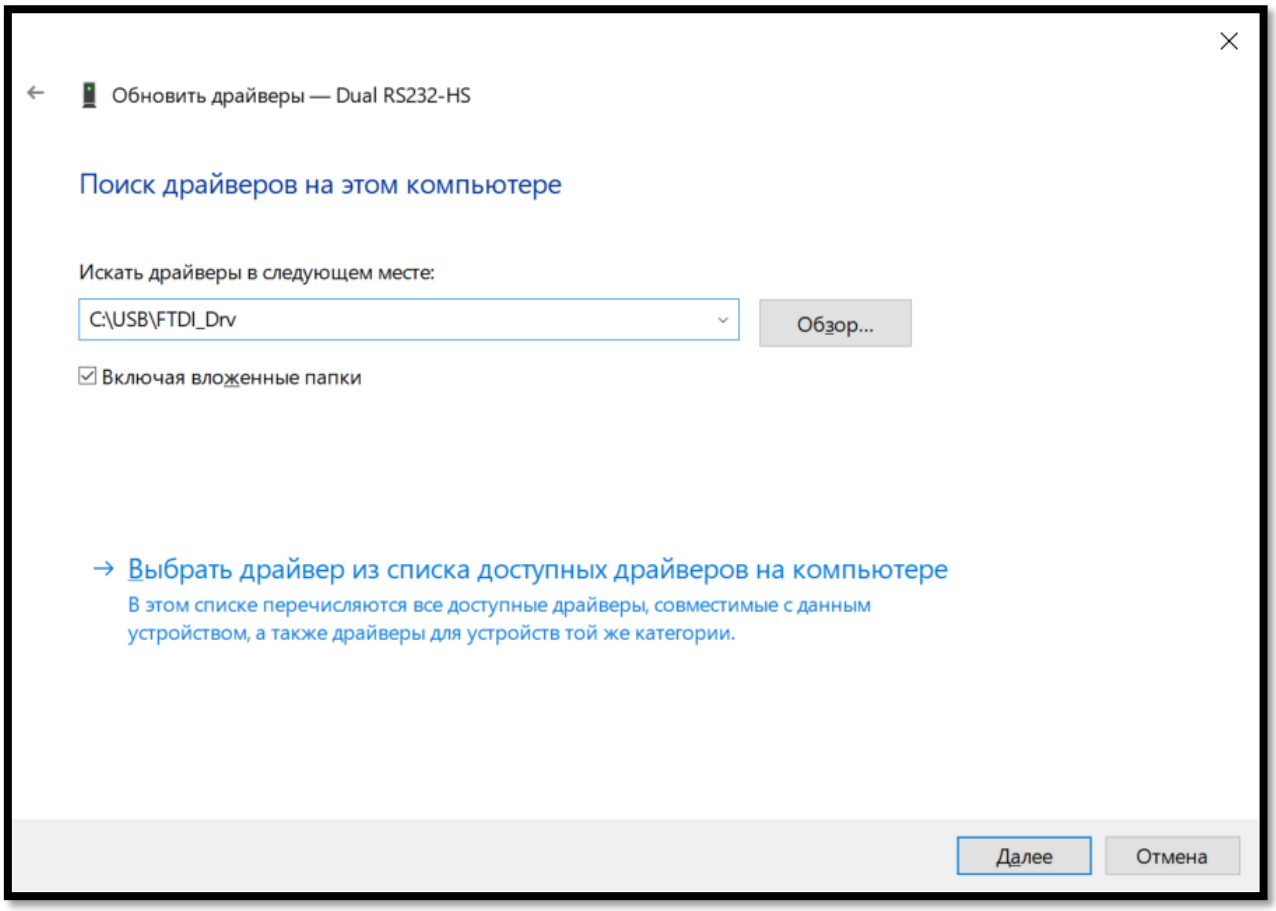


Рис. 6 Установка драйвера для VCP

Драйвер находится в папке tools/FTDI_Drv. Новую версию можно скачать с сайта https://ftdichip.com/

После этого возникнет еще одно неопознанное устройство:

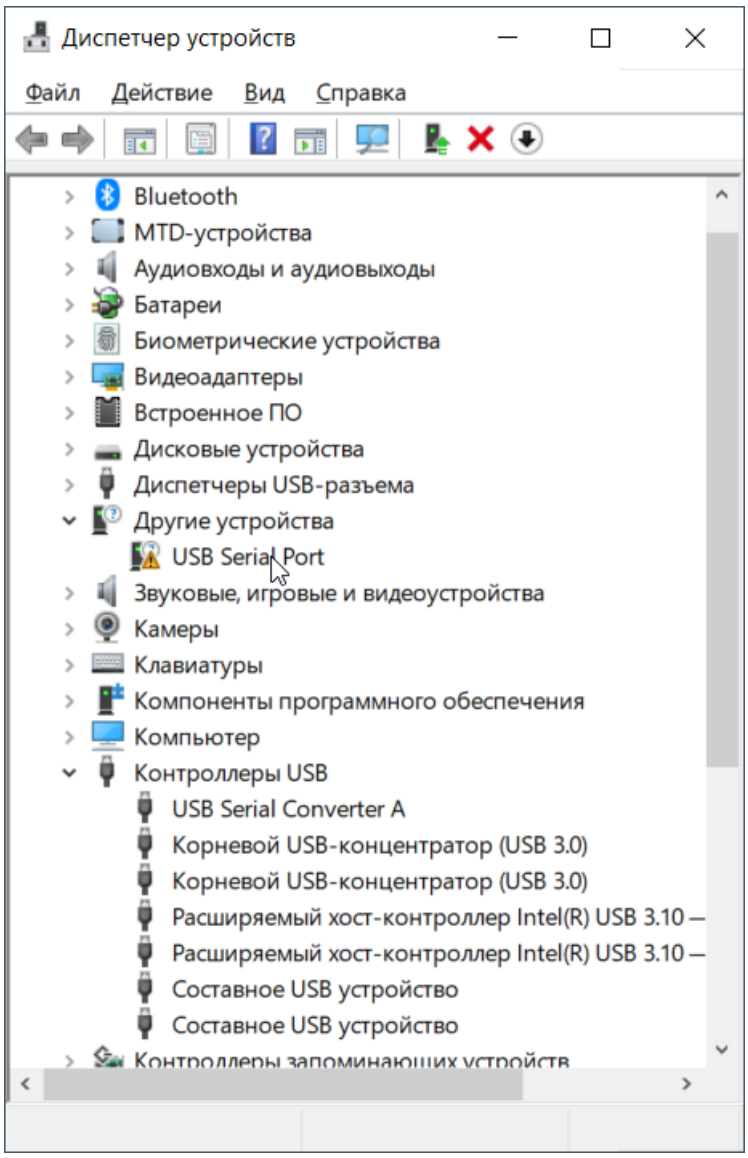
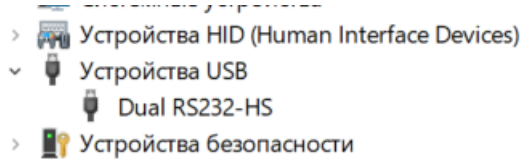


Рис. 7 После первого прохода установки

Нужно повторить процедуру установки FTDI_Drv на это устройство.

В итоге должно образоваться два устройства в диспетчере устройств:



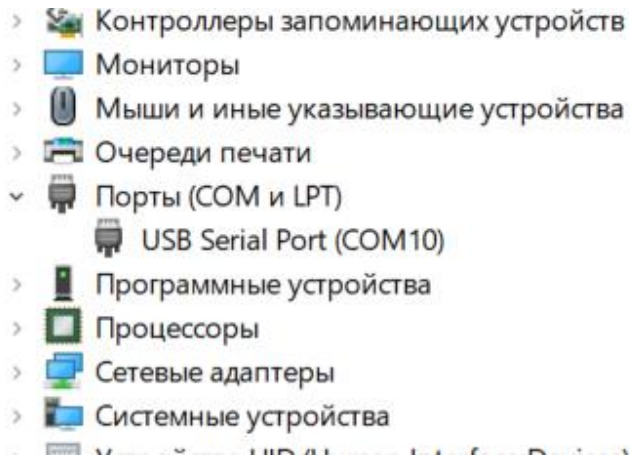


Рис. 8 Диспетчер устройств после корректной установки всех драйверов

Варианты сборки проекта

Есть три основных варианта работы кода (приложения пользователя):

1. Из оперативной памяти RAM
2. Из встроенной EEPROM
3. Из внешней QSPI FLASH памяти

Режим запуска выбирается с помощью джамперов на отладочной плате B0 и B1 (коммутирующие выводы микроконтроллера BOOT0-BOOT1 на 0V/3.3V) на разъеме CON2.

Запуск из ОЗУ удобен для проверки работы небольших блоков кода в рамках отладки отдельных небольших частей приложения или отладки функционирования периферийных устройств.

Работа основного приложения из EEPROM допускается. Данный подход ограниченно полезен в силу небольшого объёма EEPROM - 8кБ.

Работа «боевого» приложения во внешней памяти предполагается в качестве основного режима работы. Автономный запуск из внешней QSPI памяти (выводами контроллера BOOT0-BOOT1) возможен только при наличии резисторов подтяжки к линиям памяти D2 и D3. Штатная рекомендованная последовательность запуска приложения в QSPI памяти:

- запуск из встроенного EEPROM, конфигурация микроконтроллера и его интерфейсов
- переход (прыжок) в QSPI/SPIFI,

В этом случае подтяжки не обязательны.

Запуск из внешней памяти лишь установкой BOOT0-BOOT1 рекомендуется использовать только для тестовых задач, так как включается этот интерфейс в базовой производительности:

- Без активации кэш памяти
- QSPI в MISO-MOSI режиме

В результате получается невысокая скорость работы всей системы.

Стратегия прошивки памяти программ

Из-за различий видов памяти, из которых может работать приложение, запуск дебаггера разделен на две части:

- Загрузка кода в память с помощью скрипта mik32_uploader
- Собственно вход в отладчик

Так же присутствует групповой запуск – вход в дебагер с предварительной загрузкой прошивки.

Первый запуск среды программирования

Для запуска среды нажмите «Пуск» и найдите MikronIDE и выполните его

По умолчанию рабочее пространство проекта настроено на путь \MikronIDE\workspace. При первом запуске менять этот путь не нужно.

Разберем состав типового проекта (его файловую систему) на примере проекта “Bootloader”:

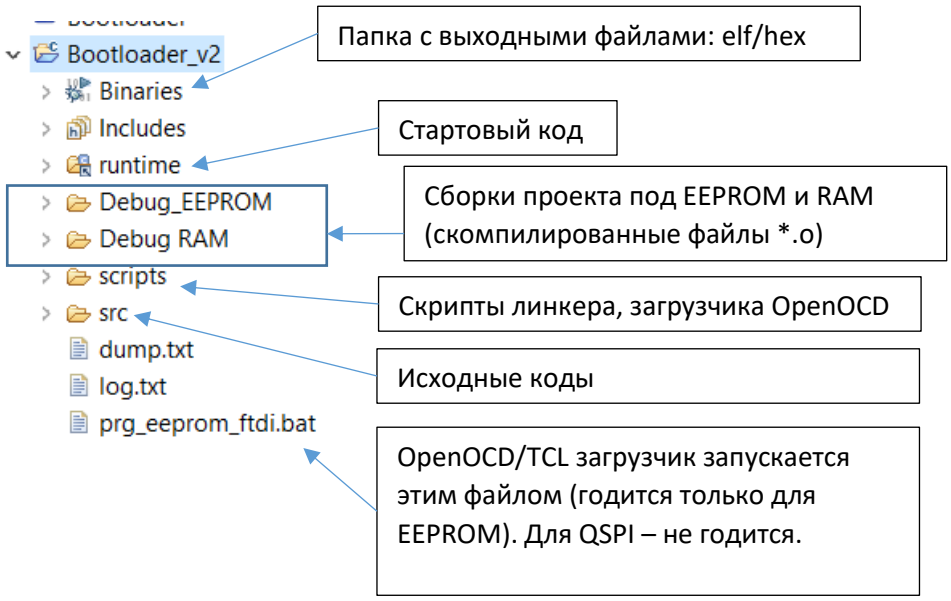


Рис. 9 Базовый состав типового проекта

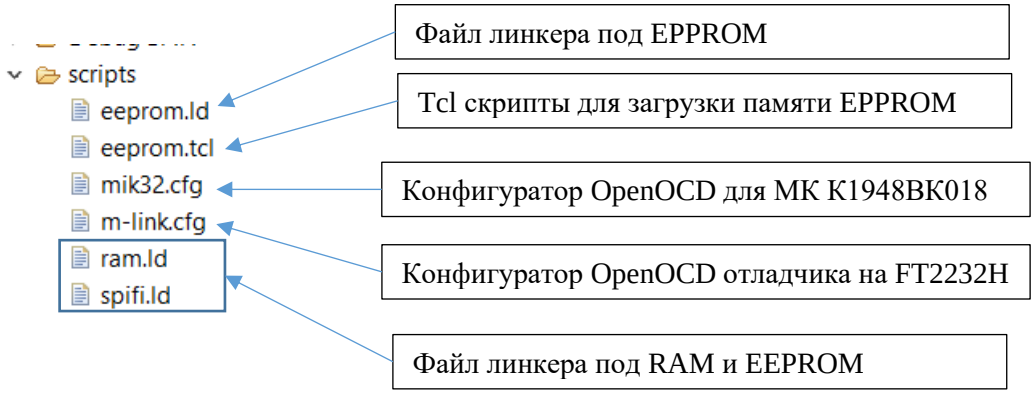


Рис. 10 Описание папки scripts

Запуск примера

Выберите интересующий пример из окна проектов и двойным щелчком мыши откройте его

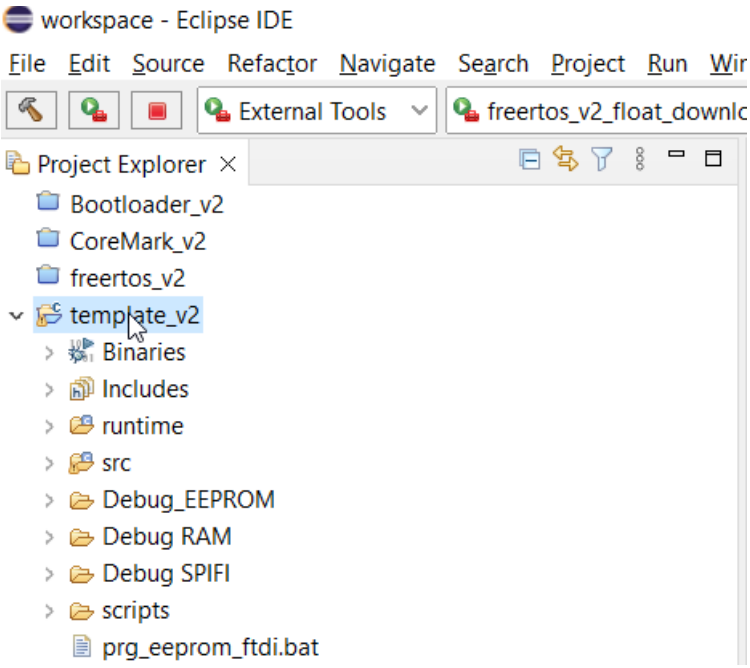


Рис. 11 Браузер проекта

Откройте файл main.c.

Далее следует выбрать конфигурацию сборки, варианты в примерах:

- Debug RAM
- Debug EEPROM
- Debug SPIFI (QSPI)

Конфигураций сборки может быть много, и они могут варьироваться помимо типа памяти, в которой будет работать код, степенью оптимизации и прочими опциями. Эти конфигурации можно создавать самостоятельно.

Для этого правой кнопкой мыши по проекту выбирается нужная конфигурация:

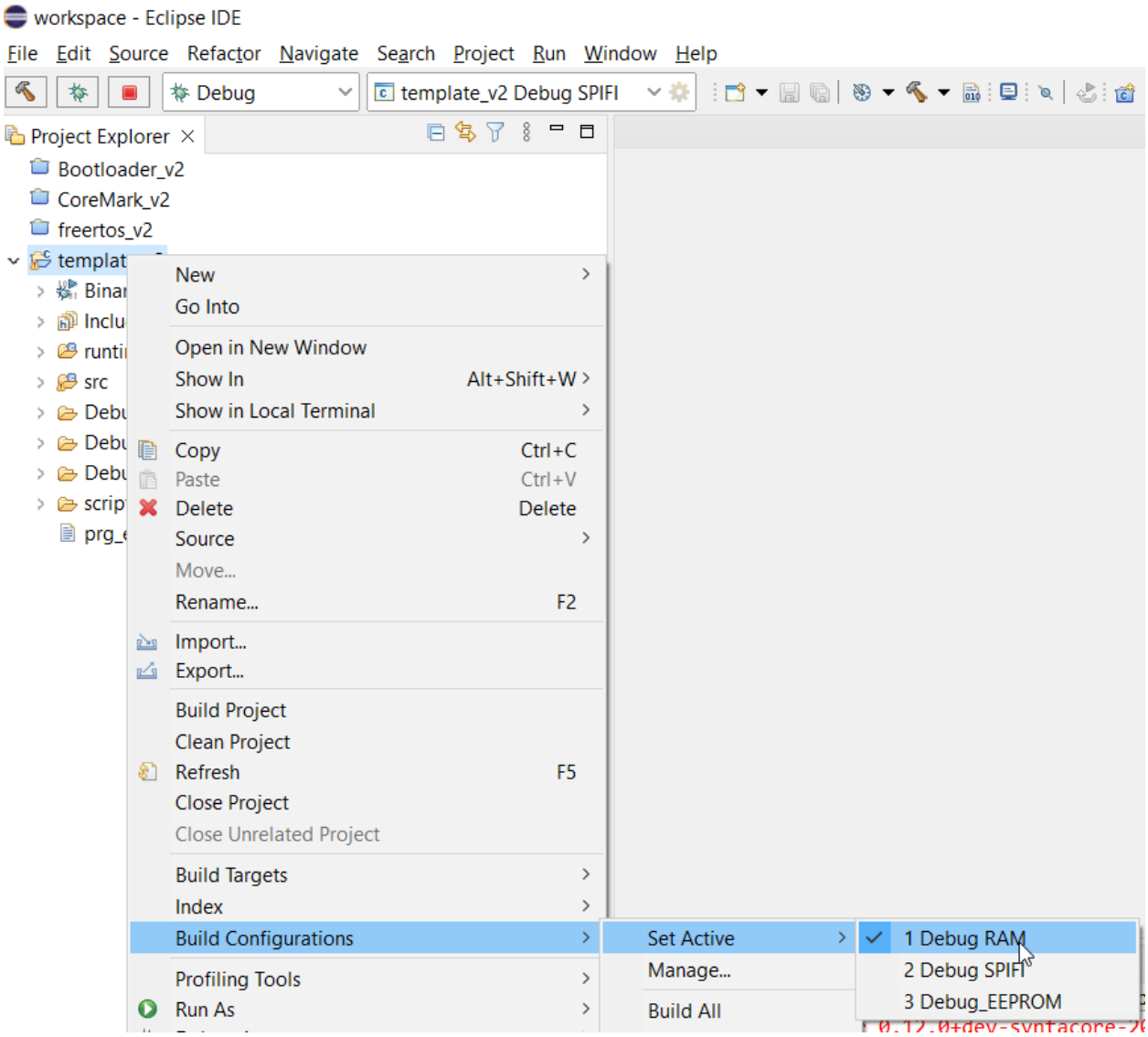


Рис. 12 Выбор конфигурации сборки RAM

После – можно собирать проект:

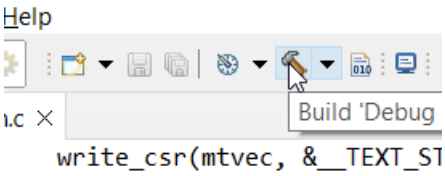


Рис. 13 Запуск загрузки и отладки

Или Ctrl+B.

Если сборка прошла без ошибок, о чем появится сообщение в консоли, то можно загружать прошивку в память и отлаживать код.

Прошивка и отладка

Выберите конфигурацию, например, «Debug RAM» (как на скриншоте выше). Перемычками установите запуск из RAM.

Убедитесь, что курсор установлен на любой строке файла main.c. Если курсор будет установлен у другом окне, то отладка не запустится.

Запустите дебагер через панель инструментов:

a.

Рис. 14 Запуск загрузки через панель инструментов – выпадающее меню

b.

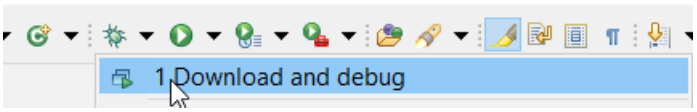


Рис. 15 Запуск загрузки через панель инструментов

После чего запустится отладчик и можно отлаживать код, ставить точки останова, контролировать состояние переменных, памяти и т д.

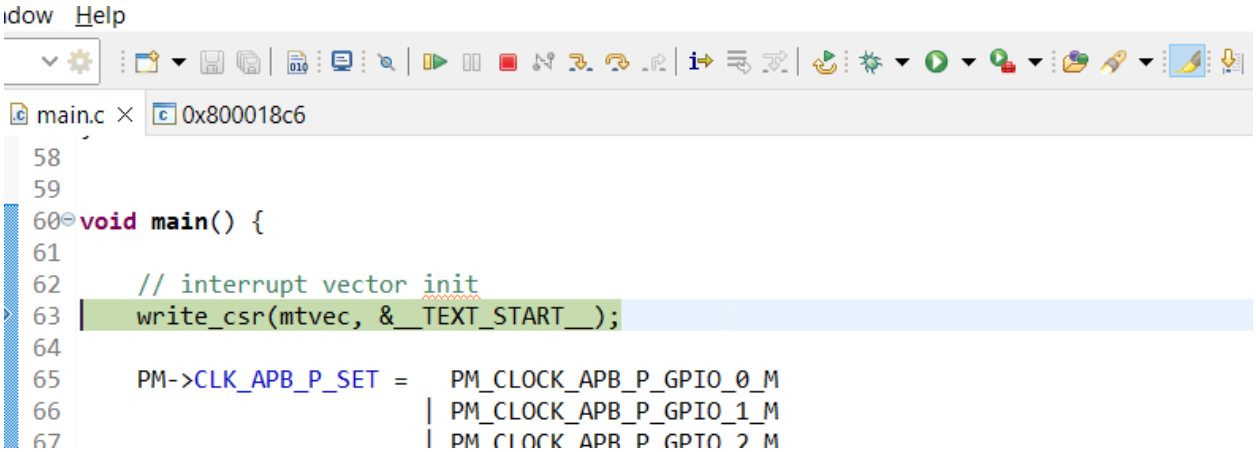


Рис. 16 Отладчик

Принцип запуска кода из SPIFI с максимальной производительностью

Принцип запуска кода из SPIFI следующий:

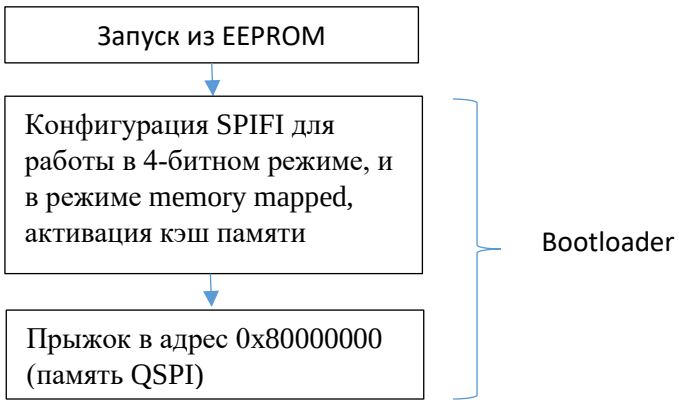


Рис. 17 Стратегия работы через загрузчик

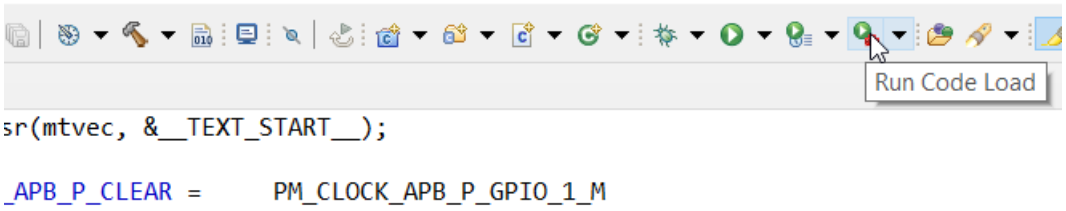
В примерах есть проект Bootloader (стартовый загрузчик), который выполняет эти базовые функции. Загрузчик предварительно должен быть прошит в EEPROM микроконтроллера. После этого можно прошивать и отлаживать код в SPIFI на максимальной скорости. При этом сигналы B0-B1 должны быть всегда на выборе EEPROM.

Bootloader можно доработать под выбранную микросхему памяти. А также добавить в него функционал обновления прошивки в QSPI памяти.

Прошивка кода без запуска отладчика

Прошивка кода осуществляется через конфигурацию «внешний инструмент»: «External tool». Прошивальщик в SPIFI/EEPROM/RAM уже преднастроен в рабочем пространстве.

Для прошивки откройте интересующий проект. Выберите файл из этого проекта, например, main.c, установите курсор в этот файл. И в панели инструментов, нажав на иконку, запустите прошивку:



В результате код прошьется в память, на которую настроена конфигурация сборки, но отладчик не запустится.

Проекты-шаблоны

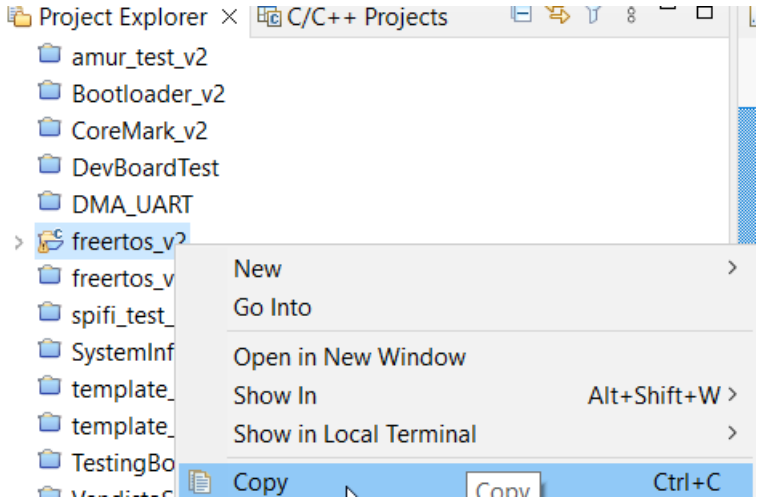
Шаблоны из комплекта поставки

В рабочую среду включены несколько примеров-шаблонов:

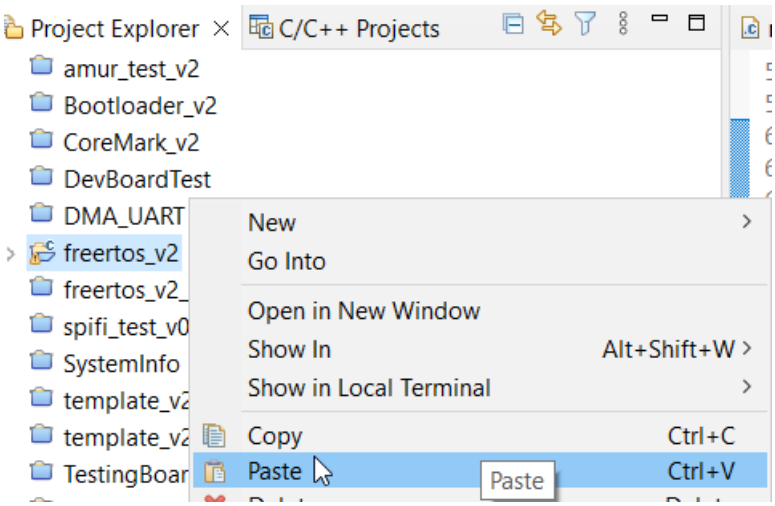
1. **Bootloader**. Работающий из встроенного EEPPROM – его же можно назвать «загрузчиком». Так же в отладочных целях можно запустить проект из оперативной памяти. Код выполняет функции инициализации внешней памяти QSPI, включает кэш, и переводит исполнение кода на неё. Так же во внутреннюю EEPROM память можно поместить приложение для обновления внешней памяти.
2. **Template**. Код, преимущественно рассчитанный на работу из внешней QSPI памяти – он же - шаблон для «боевого приложения». Так же в конфигурацию этого проекта включены конфигурации на сборку в RAM и EEPROM. Для каких-то приложений может быть достаточно и этой небольшой памяти 8кБ, тогда внешняя память будет не нужна.
3. **CoreMark**. Проект для тестирования производительности микроконтроллера. Присутствуют конфигурации для QSPI, EEPROM и RAM.
4. **Template_FreeRTOS**. Присутствуют конфигурации для QSPI, RAM. Для EEPROM – объем выходного кода слишком большой, поэтому не рекомендуется.
5. **SystemInfo**. Данный пример при запуске на отладочной плате с подключенным часовым кварцевым резонатором позволяет определить калибровочный коэффициент для регистра калибровки HSI (в составе настроечного регистра WU->CLOCKS_SYS). А так же данный пример вычитывает ID микроконтроллера и дату его производства. Вся информация выводится в UART_1 на скорости 9600N1.
6. **DMA_UART** – Пример с использованием DMA.

Как создать свой проект на базе шаблонов.

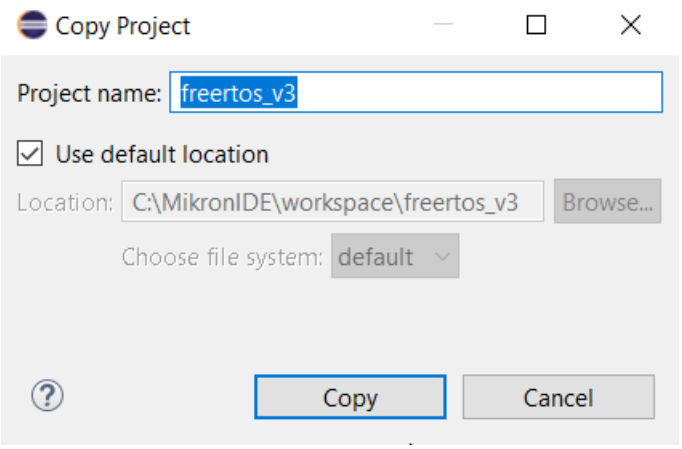
Выбирите интеерсущий шаблон-проект, откройте его и правой кнопкой мыши по проекту выберите «Copy»:



Далее по пустому полю нажмите правой кнопкой мыши и выберите «Paste»:



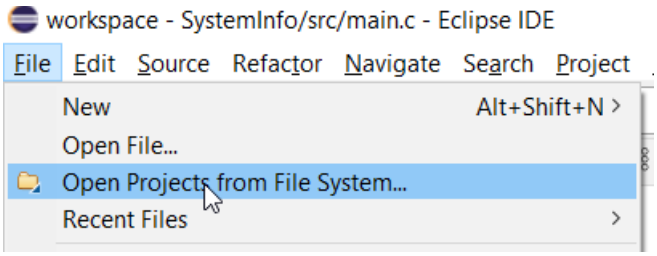
И появится окно с выбором имени для нового проекта. После выбора имени – нажмите кнопку «Copy»



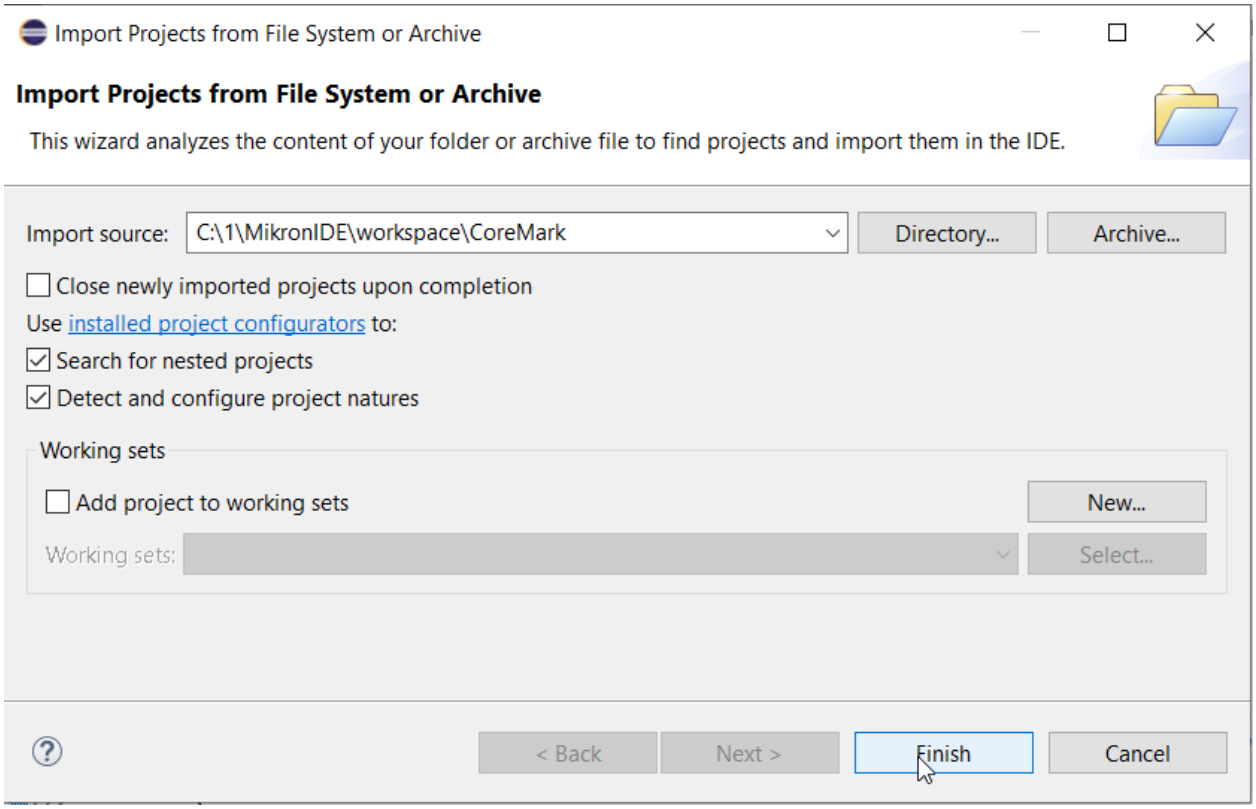
Проект можно переименовать в будущем.

Открытие проектов из комплекта примеров MikronIDE

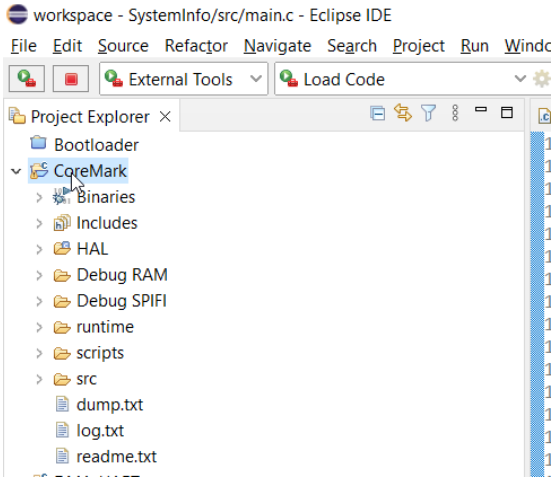
В папке workspace содержатся проекты-примеры, которые по-умолчанию не добавлены в рабочее пространство. Для их добавления нужно «Открыть проект», сначала выбирается в меню функция «Open Project…»:



Далее выбирается папка, где находится интересующий проект, после чего следует нажать на кнопку Finish.



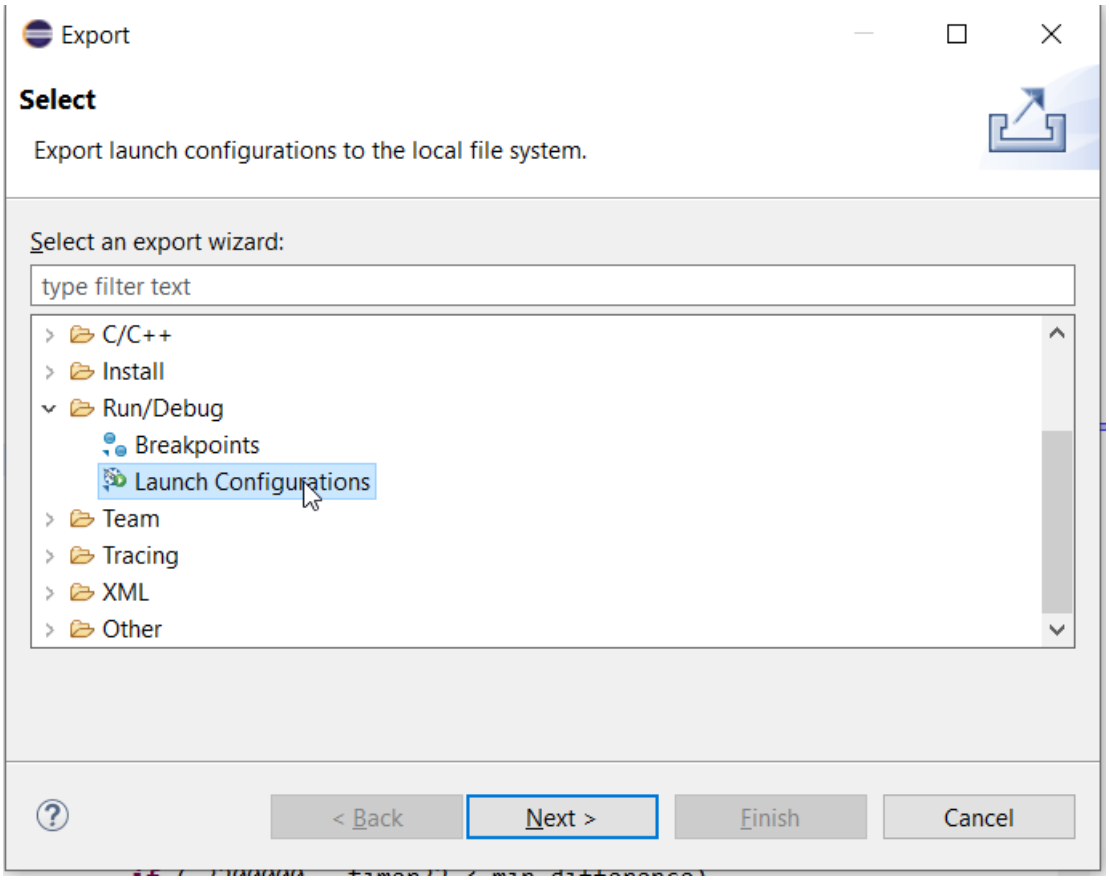
После этого проект будет добавлен в среду разработки и с ним можно работать:



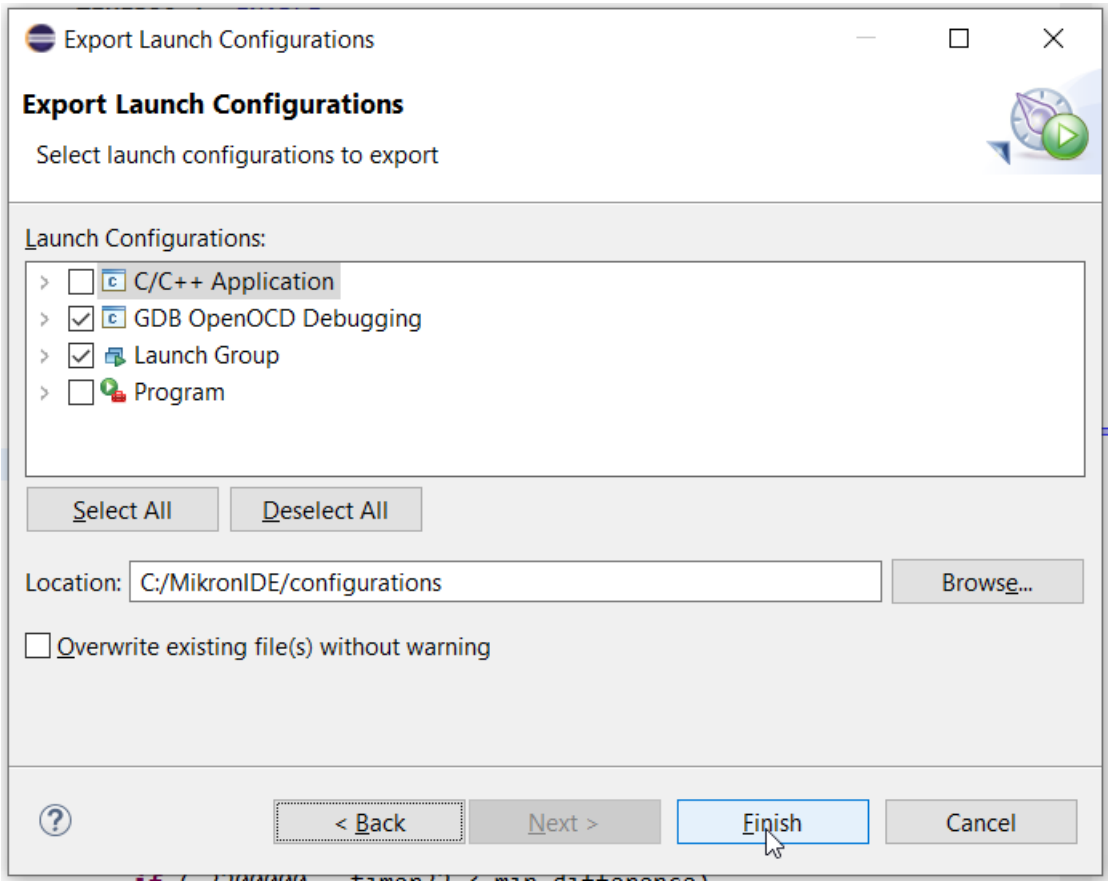
Создание нового workspace

При создании нового workspace, основная проблема – в новом workspace отсутствуют конфигурации загрузки прошивки и запуска дебагера

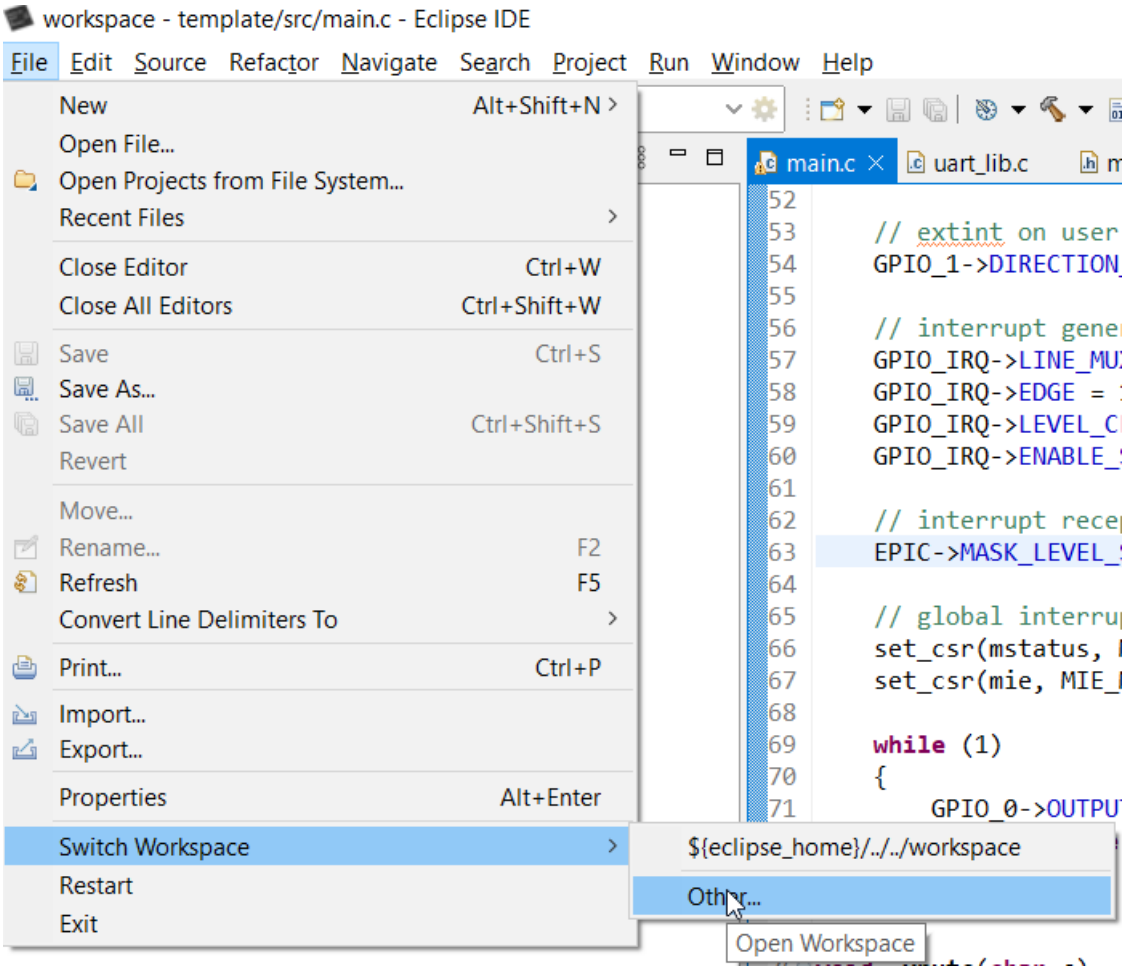
Поэтому из шаблонного workspace сначала следует экспортировать все рабочие «конфигурации»:



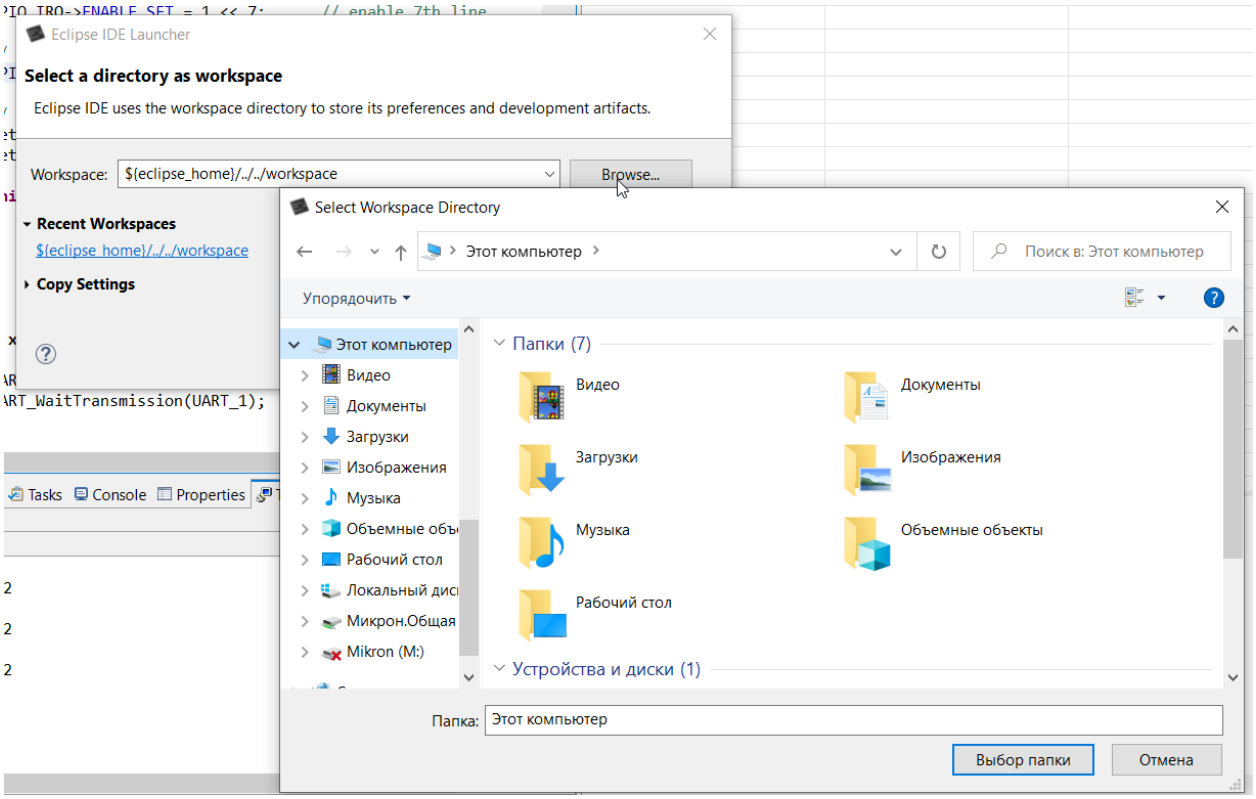
Сохраняем на диске и запоминаем путь.



После, создаем папку для нового workcpase

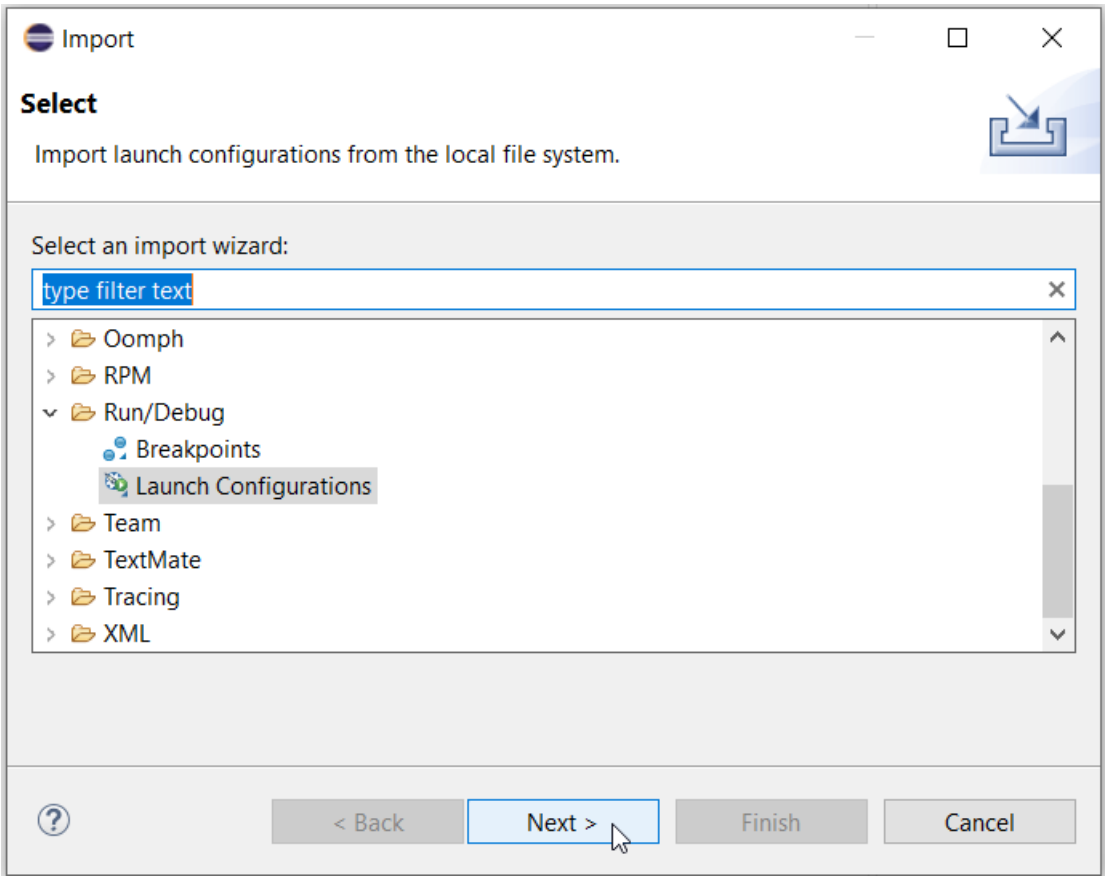


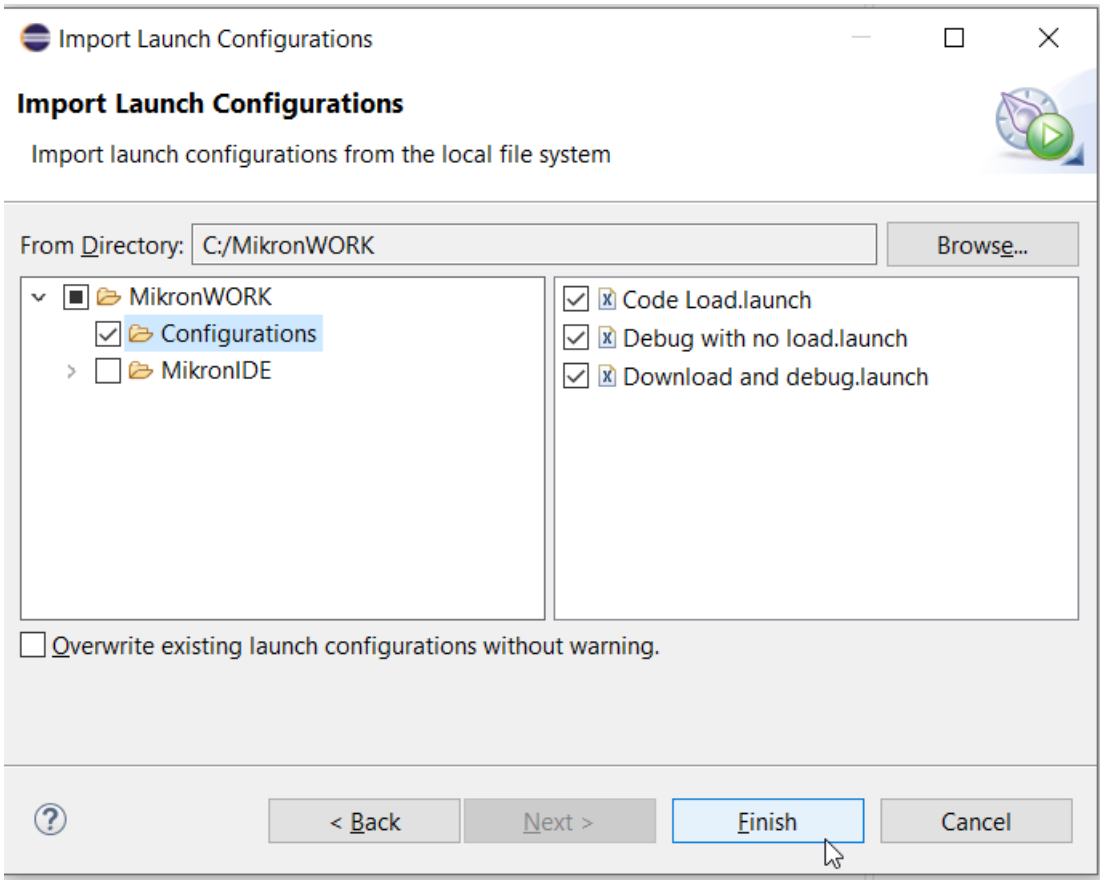
Выбираем папку для нового workspace:



Далее открываем ново-созданный «пустой» workspace и импортируем сохраненные настройки

Через File→Import…





Конфигурации теперь на своих местах, Все готово к работе!