Демонстрационная плата с микроконтроллером К1948ВК018 АМУР

Руководство по быстрому старту

Оглавление

Введение

Отладочная плата предназначена для ознакомления с микроконтроллером К1948ВК018. 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.

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

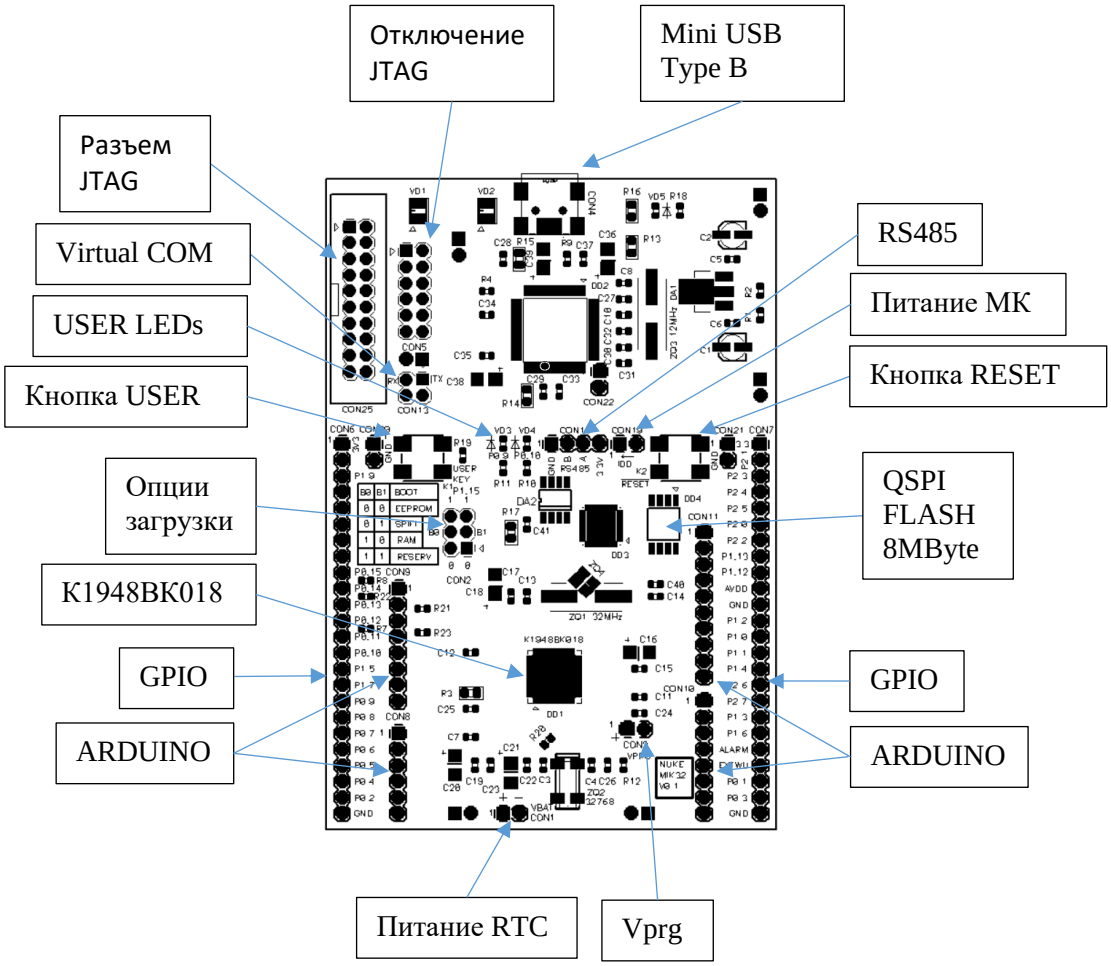
Встроенный регулятор 3.3В	MIK1117
Порт 485	MIK3485
Разъем расширения ARDUINO-UNO	Да
Портов ввода-вывода на штыревых разъемах, шт	50 (все)
Кнопки управления	2
Светодиоды для отладки	2
Кварцевый резонатор высокочастотный	32МГц
Кварцевый резонатор низкочастотный	32768Гц
Габариты	70мм*100мм

Системные требования

Для работы с микроконтроллерной платой потребуется следующее техническое оснащение:

1. Компьютер на базе Windows 10 или новее, но не ниже Windows 7.
    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-mini

Краткое описание платы



Для основной работы с платой достаточно простого подключения кабеля USB.

Питание RTC необходимо только для сохранения хода часового кварца в момент, когда основное питание отключено.

Vprg – вход для подачи повышенного напряжения питания для программирования OTP памяти 256 бит. Если эта память не используется, то оставить этот вход не подключенным.

Джампер “Питание МК” можно удалить и, подключив в разрыв амперметр, и измерить потребление тока микроконтроллером.

Ошибки первой версии платы V0.1

1. Подписи к разъему CON2 – B1 и B0 перепутаны местами, B1 – соответствует выводы BOOT0, а B0 соответствует BOOT1.
2. Отсутствуют подтяжки питания к сигналам QSPI к линиям D2 и D3. Опционально они нужны.
3. Перепутаны линии питания резервного домена и линия подключения батарейки. Решение: замкнуть выводы VBAT и VCC_BU.
4. К светодиодам VD3 и VD4 нарисовано изображение диода в неправильную сторону

Ошибки второй версии платы V0.2

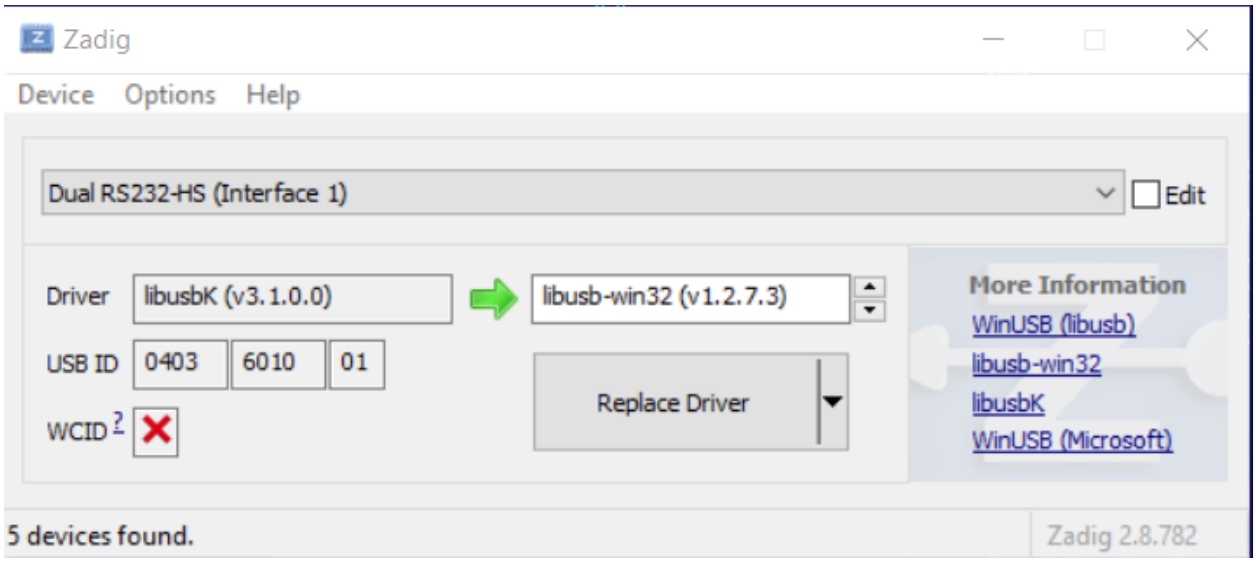
1. Перепутаны линии питания резервного домена и линия подключения батарейки. Решение: замкнуть выводы VBAT и VCC_BU.

Установка необходимого ПО

Скачайте архив https://disk.yandex.ru/d/WVTSVBOD6Jj6nw разархивируйте папку MikronIDE в удобное место, рекомендуется в корневую директорию C:\.

Установка драйвера WinUSB

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



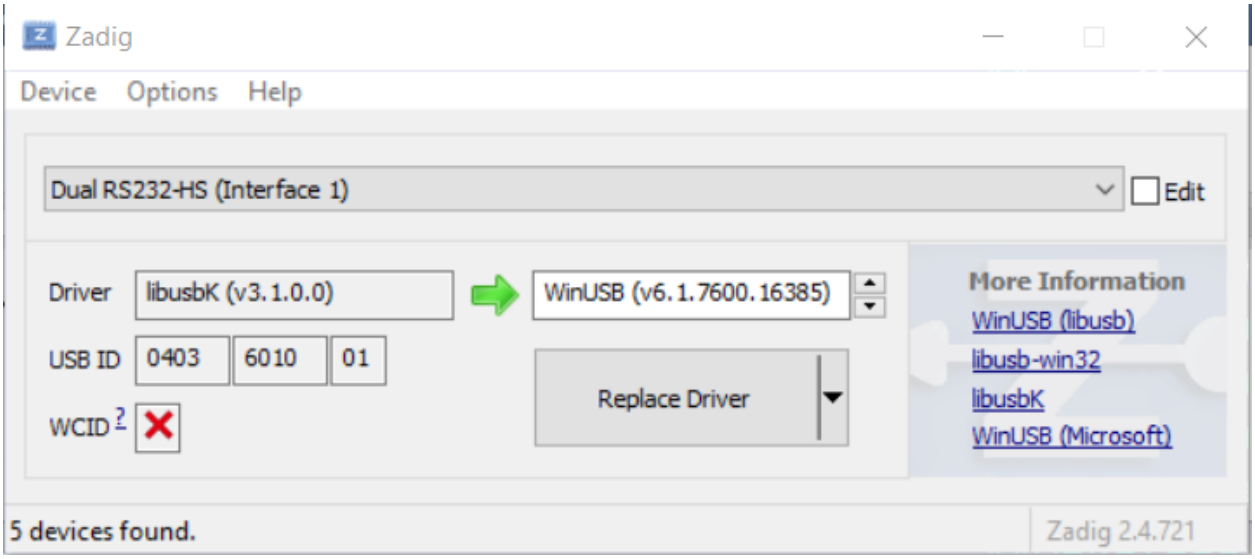
Подключите плату. В меню Options нажмите на **List All Devices**. В выпадающем списке выберите устройство с названием:

Dual RS232-HS (Interface 1)

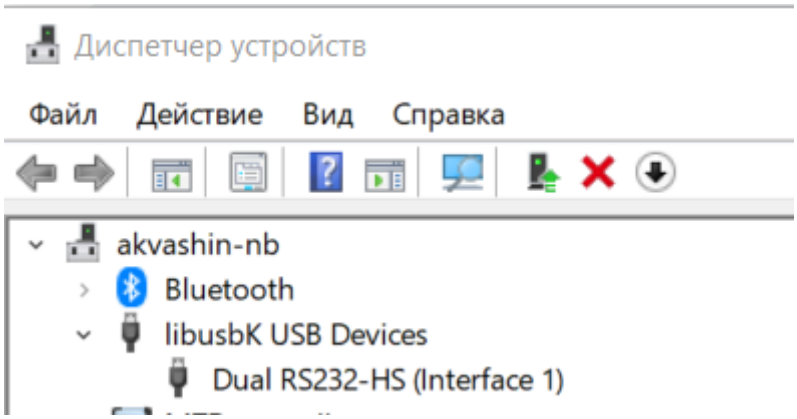
Затем проверьте что выбран один из совместимых драйверов, например, libusb-win32 в белом поле ввода (выбирать кнопками со стрелками) и нажмите кнопку **Replace Driver** или **Install Driver**.

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

Dual RS232-HS (Interface 0) – под умолчанию установится драйвер FTDIBUS и его менять не нужно. Он был и останется виртуальным ком-портом, который будет нужен для строковой отладки приложений.



Если после этого в диспетчере устройств он не появится в таком виде,



то варианты действий:

- Отключить плату и подключить заново
- Перезагрузить компьютер
- Установить другой драйвер из доступных в Zadig, например, WinUSB (v6.1.7600.16385)

Знакомство с проектом-шаблоном

Общие принципы

Есть три основных варианта работы кода:

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

Режим запуска выбирается с помощью джамперов B0 и B1 (пины BOOT0-1)на разъеме CON2.

Запуск из ОЗУ удобен для отладки небольших блоков кода в рамках отладки работы отдельных небольших блоков приложения.

Работа основного приложения из EEPROM допускается, но 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 (пинами BOOT0-1) возможен только при наличии резисторов подтяжки к линиям памяти D2 и D3. Штатная рекомендованная последовательность запуска приложения в QSPI памяти – запуск из встроенного EEPROM и переход оттуда в QSPI, в этом случае подтяжки не обязательны.

Запуск из внешней памяти «джамперами» не рекомендуется, так как включается этот интерфейс в базовой производительности:

- Без активации кэш памяти
- В базовом 1-битном режиме

Этот режим не позволит получить высокую производительность, поэтому рекомендуется только для случаев тестирования.

Проекты-шаблоны

Есть четыре основных примера-шаблона:

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. FreeRTOS. Присутствуют конфигурации для QSPI, RAM. Для EEPROM – объем выходного кода слишком большой, поэтому не рекомендуется.

Стратегия прошивки памяти программ

Из-за различий видов памяти, из которых может работать приложение загрузка может быть так же осуществлена несколькими разными способами. Все способы загрузки базируются на возможностях OpenOCD.

В OpenOCD новую прошивку можно отправлять разными способами:

1. Напрямую отправляя данные в OpenOCD
   Этот способ годится для записи прошивки во встроенное ОЗУ. Этот режим заложен в базовом функционале загрузки Eclipse. Более сложные сценарии загрузки (EEPROM и особенно QSPI) сложно реализуемы.
2. С помощью скрипта OpenOCD (tcl формат)
   Этот вариант загрузки поддерживает запись любой памяти, но в реализации довольно сложен, так как язык tcl загрузчика слишком прост для сложных последовательностей выполнения. Для программирования встроенной EEPROM этот режим годится, но не более. В частности для этого разработан скрипт eeprom.tcl (в каждом шаблоне проекта в папке scripts).
3. С помощью внешнего приложения, которое отправляет данные в открытый порт OpenOCD.

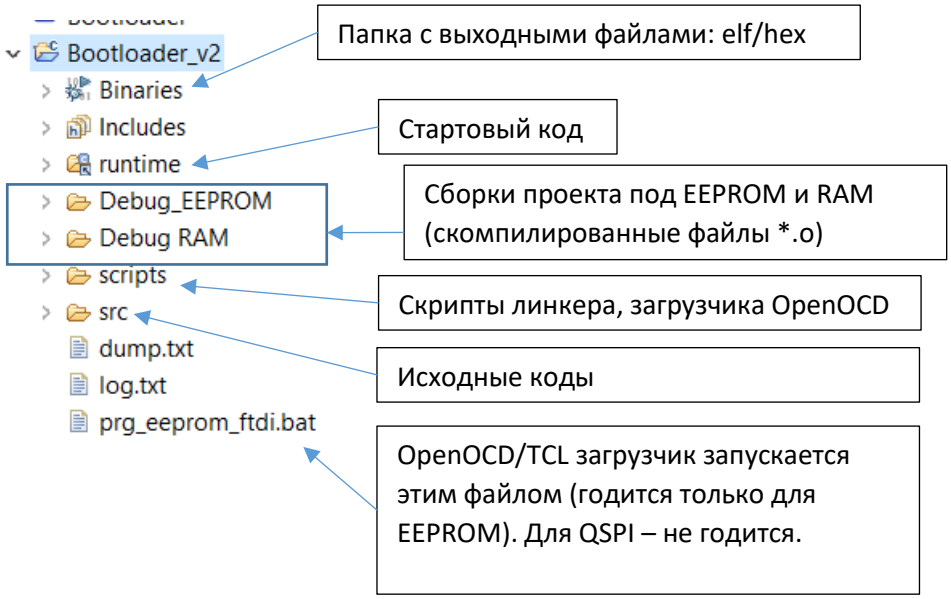
   Потенциально в этом подходе можно осуществлять манипуляцию микроконтроллерной периферией почти без ограничений, свойственных для первых двух способов. Для такого программирования создан загрузчик \MikronIDE\tools\mik32-uploader-master\mik32_upload.exe. Этот загрузчик может программировать внешнюю память QSPI, а так же можно прошивать внутреннюю EEPROM и деже RAM (загрузчик автоматически выбирает алгоритм прошивки ориентируюясь на адреса, в которые линкован код).

Первый запуск среды программирования

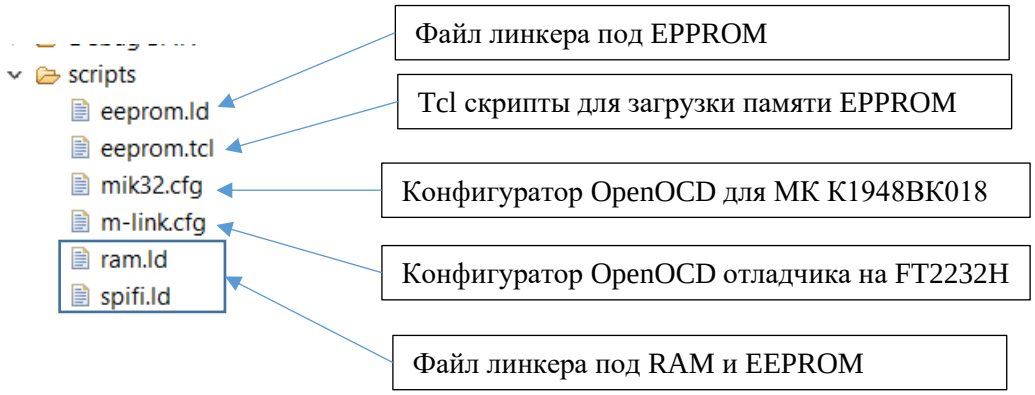
Для запуска среду зайдите в папку MikronIDE и выполните файл start-Mikron-ide.cmd.

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

Разберем состав типового проекта (его файловую систему) на примере загрузчика:

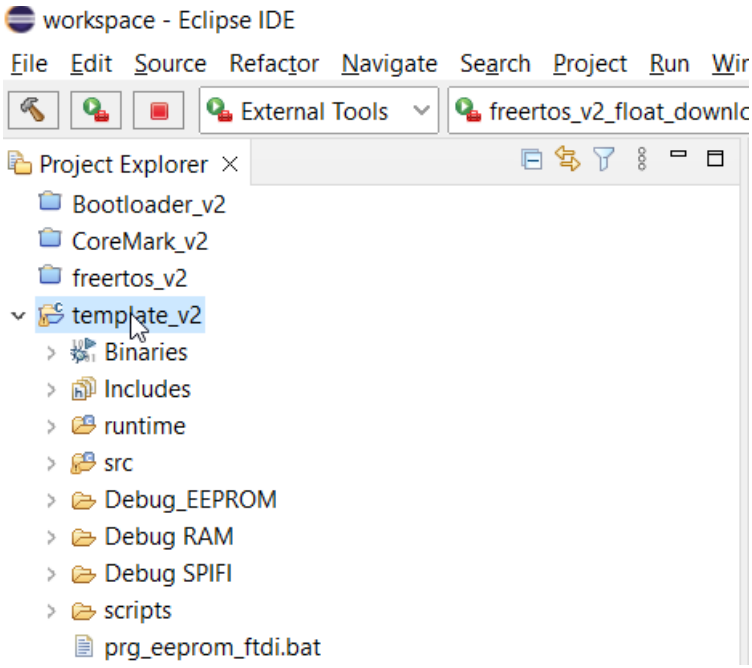


Описание папки scripts:



Запуск примера

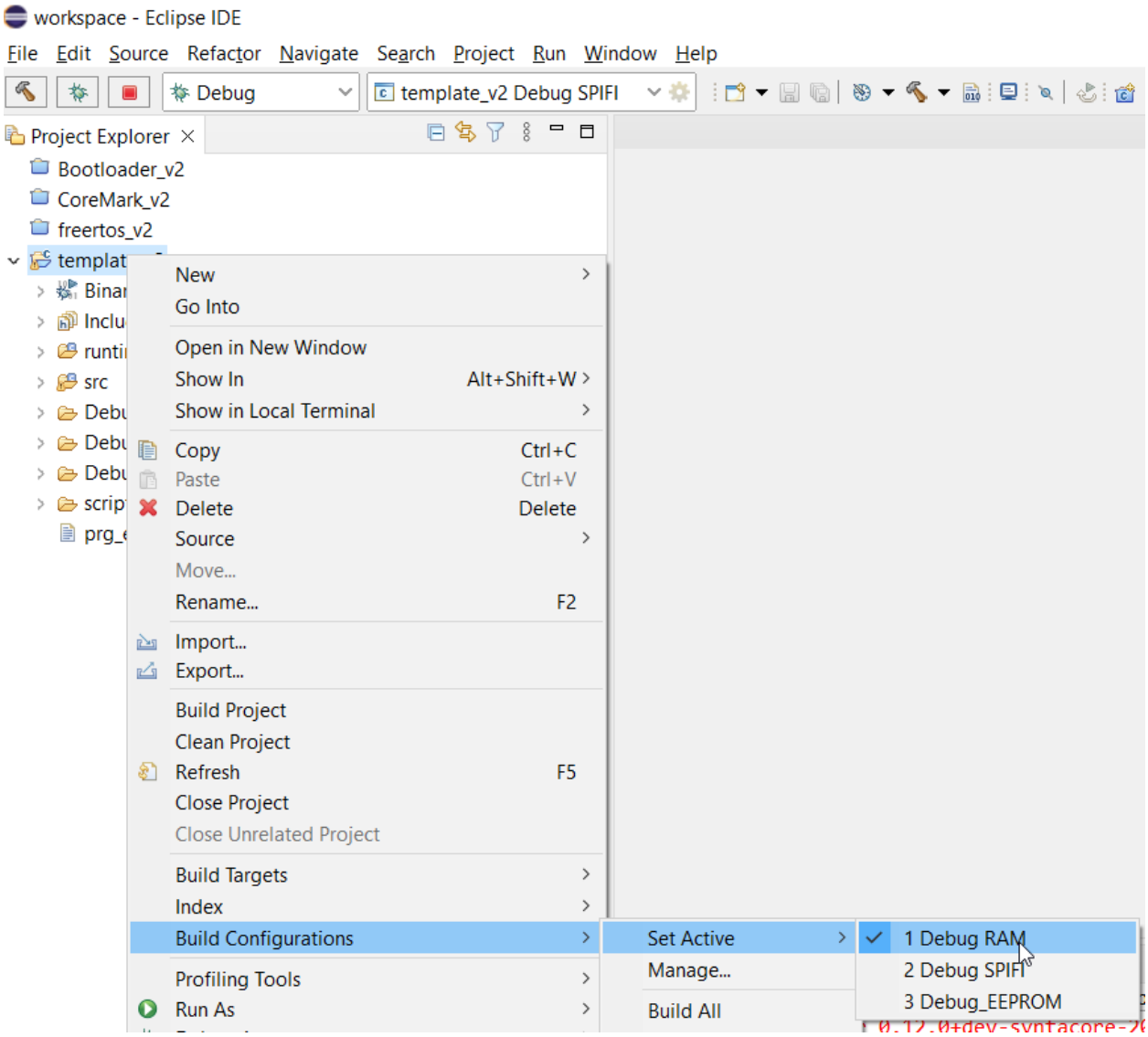
Выберите интересующий пример из окна проектов и двойным щелчком мыши откройте его



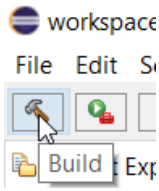
Далее следует выбрать конфигурацию, обычно это

- RAM
- EEPROM
- SPIFI (QSPI)

Для этого правой кнопкой мыши по проекту выбирается нудная конфигурация:



После – можно собирать проект:

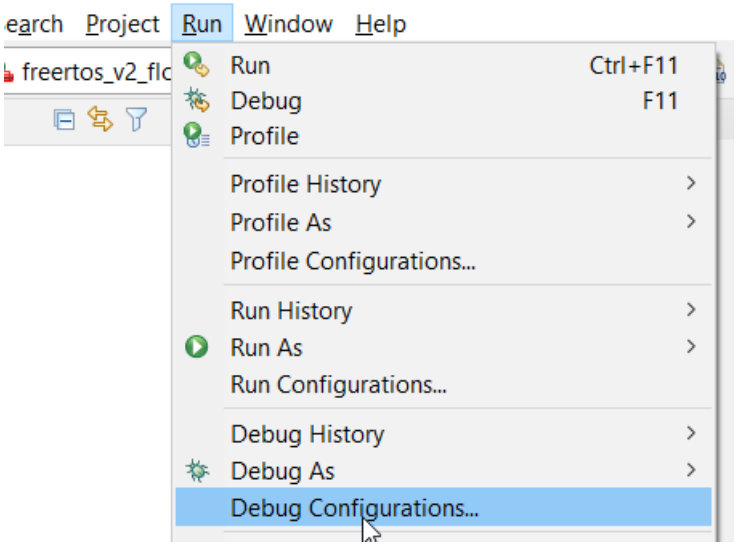


Если сборка прошла без ошибок, то можно загружать прошивку в память и отлаживать код.

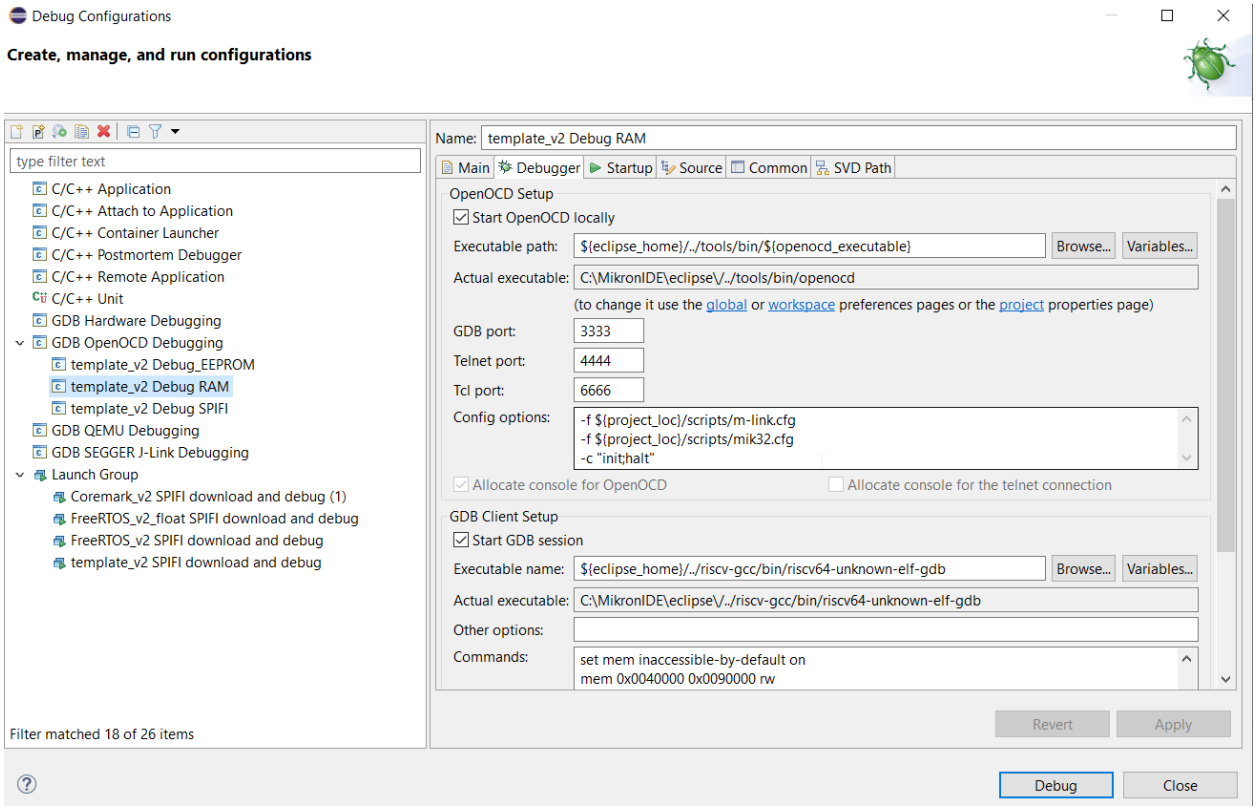
Загрузка и отладка в ОЗУ

Выберите конфигурацию «Debug RAM».

Откройте менеджер конфигураций отладки:

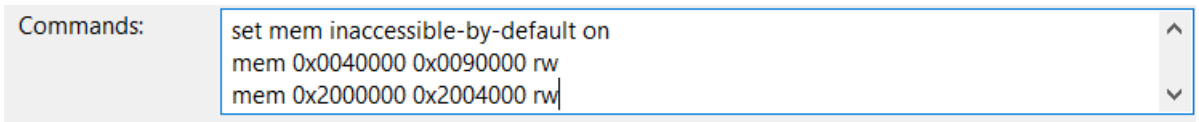


Откроется окно:



Здесь можно нажать на «Debug» и код «прошьется» в ОЗУ и включится дебаггер.

Подробнее про доступ к памяти для OpenOCD:



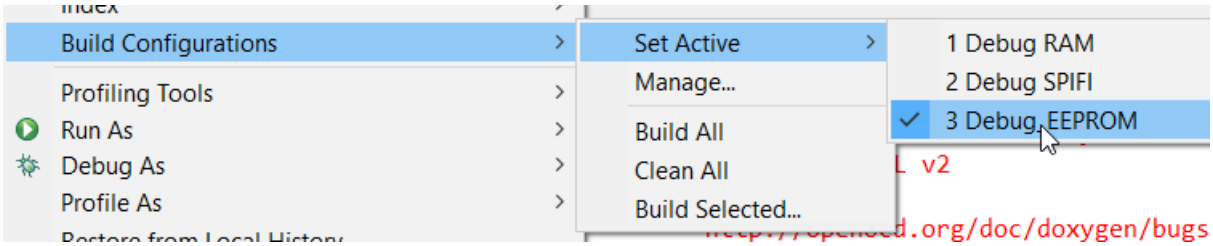
Здесь должен быть такой текст:

```
set mem inaccessible-by-default on
mem 0x0040000 0x0090000 rw
mem 0x2000000 0x2004000 rw
mem 0x1000000 0x1002000 ro
```

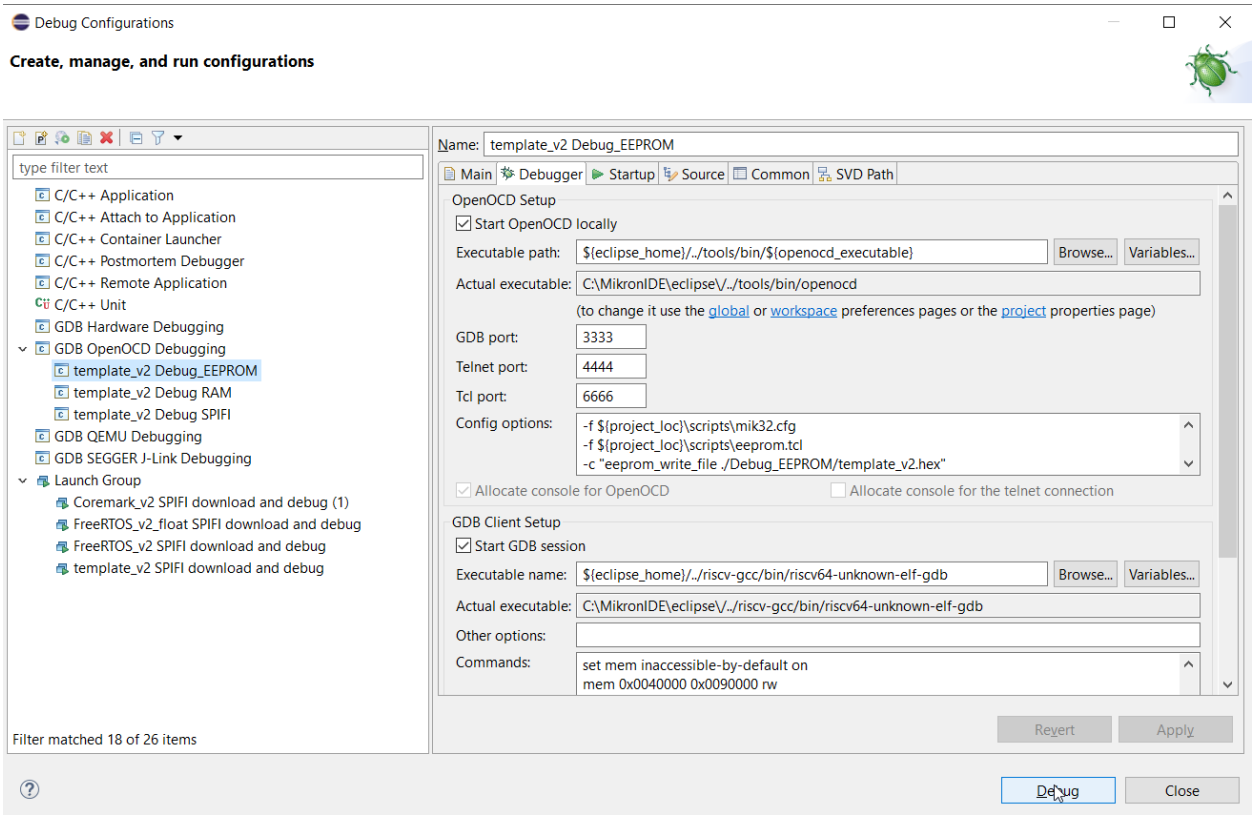
```
mem 0x80000000 0xffffffff ro
set arch riscv:rv32
set remotetimeout 10
set remote hardware-breakpoint-limit 2
```

Загрузка и отладка в EEPROM

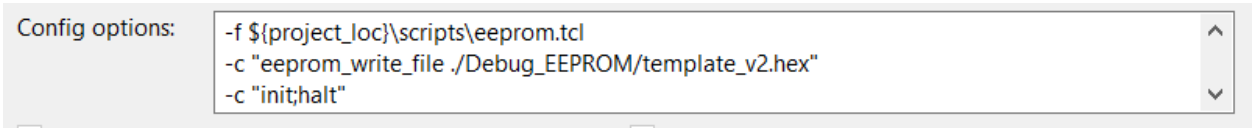
Выберите конфигурацию «Debug_EEPROM».



Так же открывается менеджер конфигураций отладки:

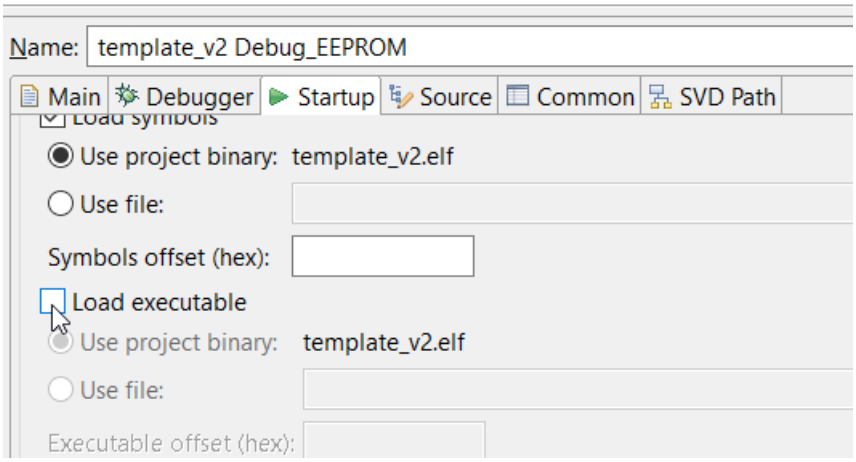


В примере применена загрузка памяти через tcl скрипт. Здесь в отличии от конфигурации RAM присутствует активация загрузки:



Можно нажать на кнопку «Debug» и код загрузится в EEPROM и включится режим отладки.

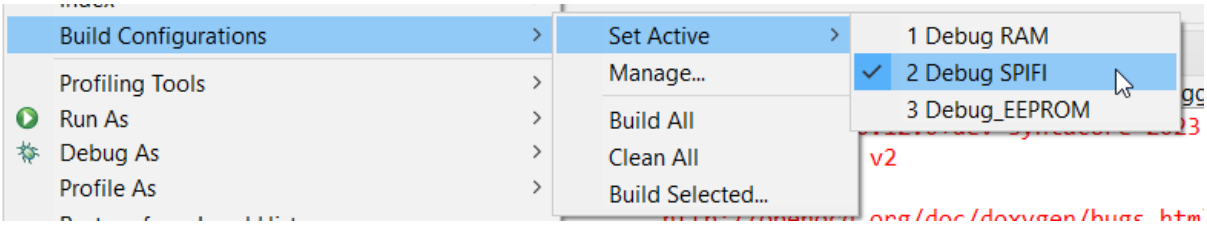
И в опциях запуска отключена загрузка прошивки штатным методом:



Внимание: для отладки в памяти EEPROM может использоваться только 2 точки останова.

Загрузка и отладка в QSPI

Выберите конфигурацию «Debug SPIFI».

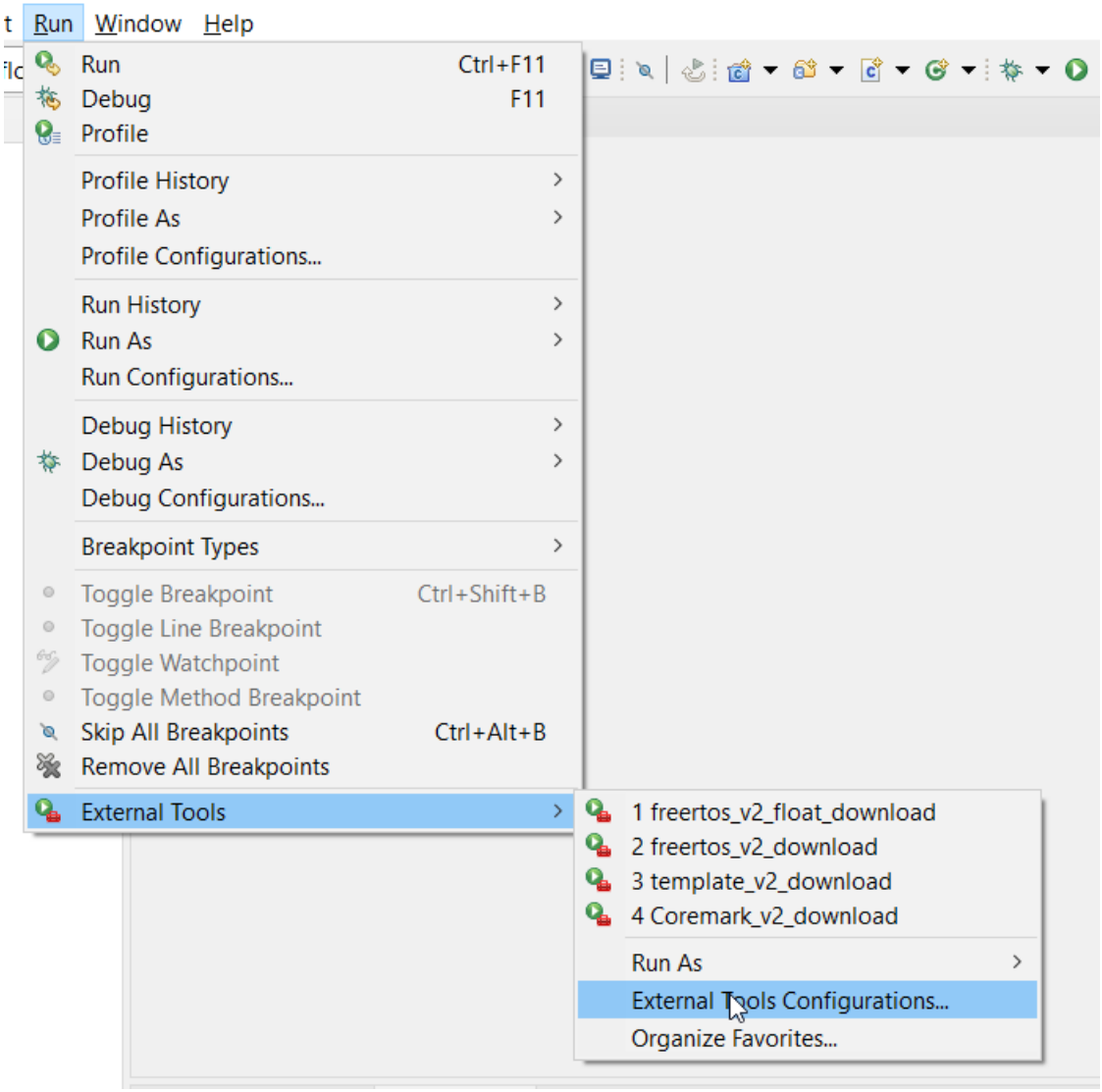


В случае если выбрана память QSPI (SPIFI интерфейс), то предварительно настраивается «прошивальщик» (в примере уже всё настроено)

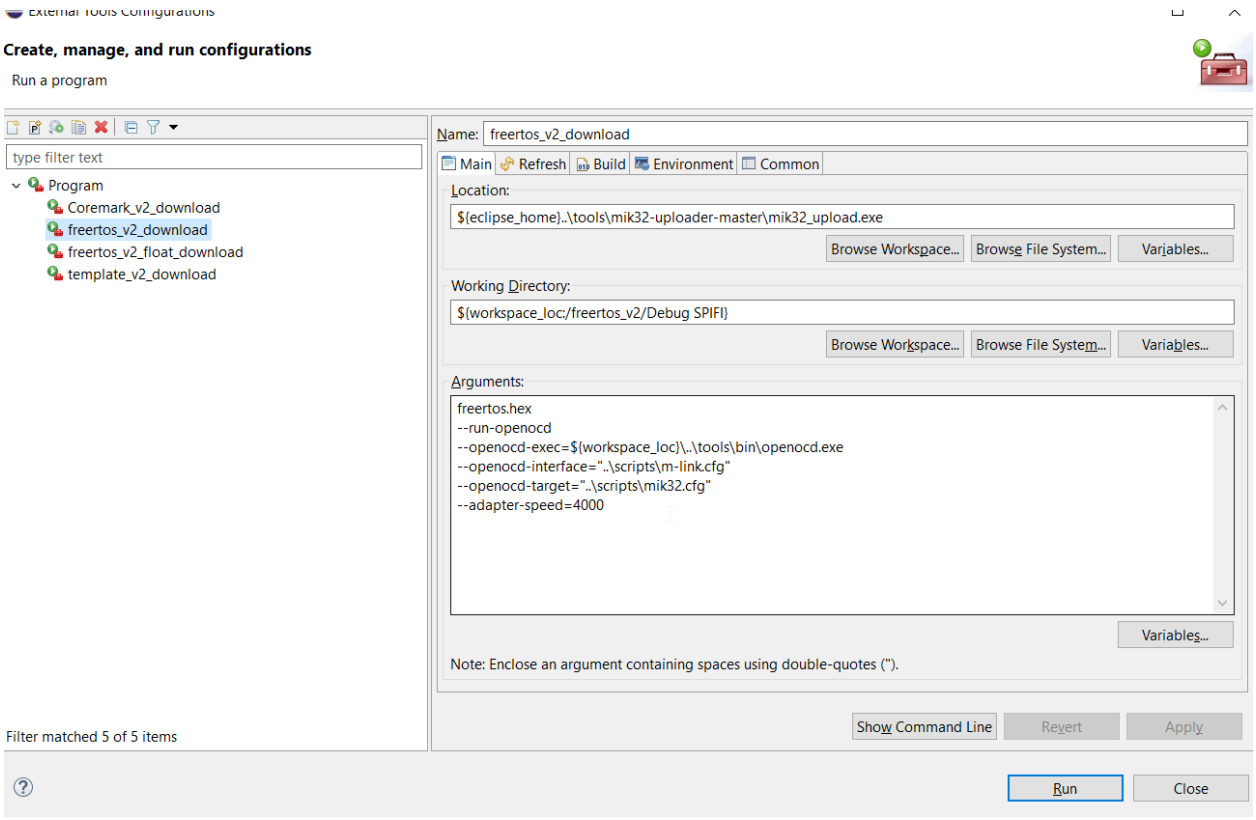
Поэтом запуск приложения разделяется на два+1 этапа:

1. Предварительно должен быть прошит в EEPROM стартовый загрузчик QSPI: проект Bootloader.
2. Загрузка кода через внешний инструмент загрузки
3. Запуск отладчика

Прошивка кода осуществляется через конфигурацию внешнего инструмента «External tool»:

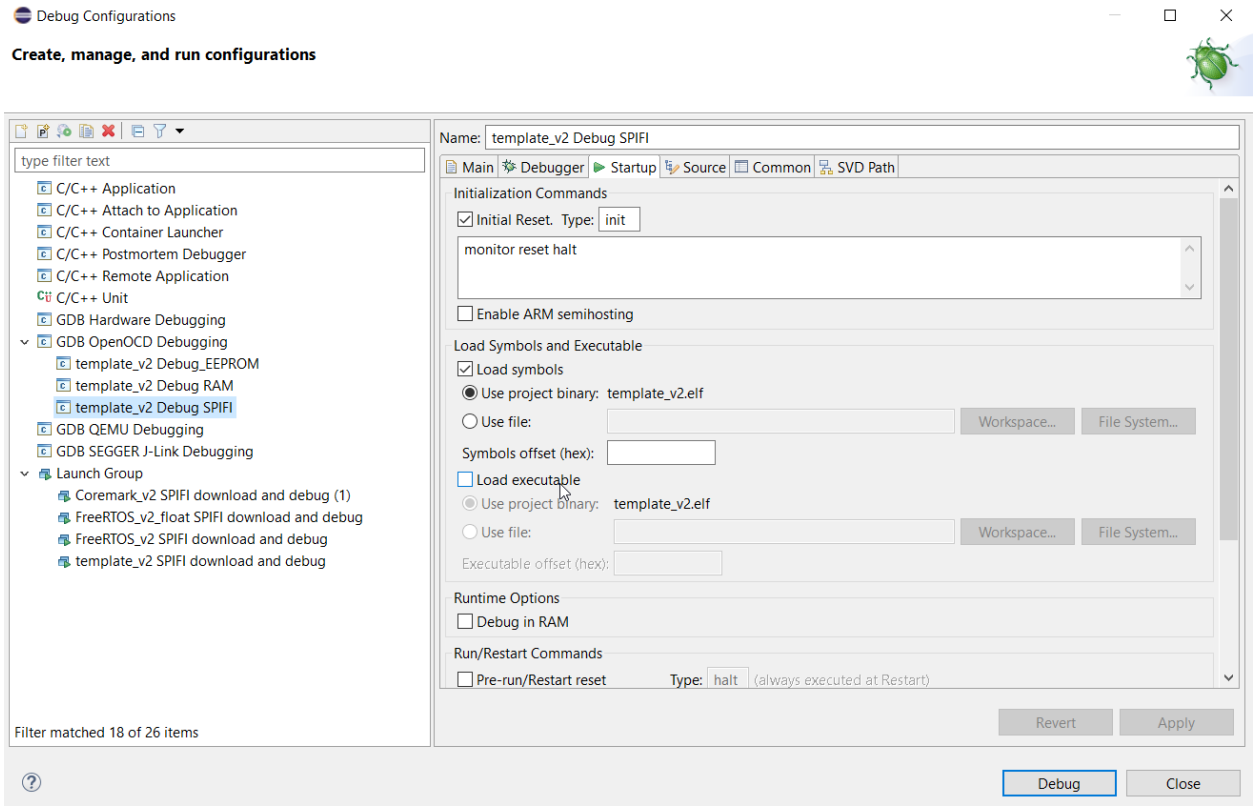


Загрузка будет выглядеть следующим образом:



Можно нажать на «Run» и код прошьется в QSPI память.

Для запуска отладки нужно перейти в менеджер отладки:



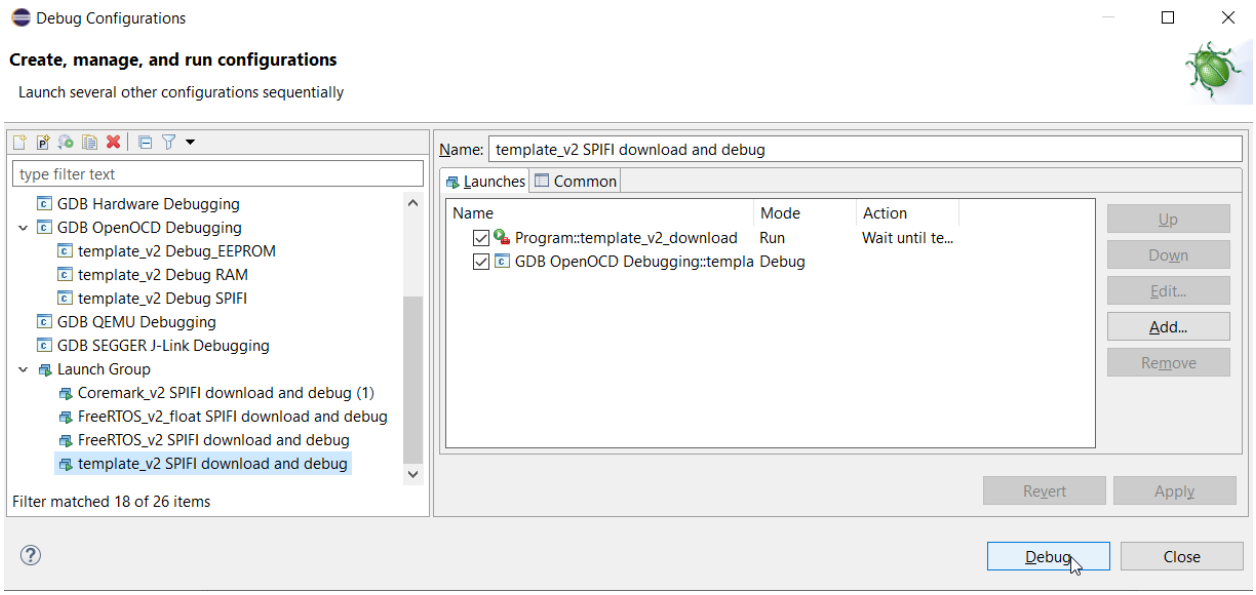
Данная конфигурация идентична как для RAM, кроме загрузки исполняемого файла, которая не нужна, так как код уже будет УЖЕ находиться в QSPI памяти.

Далее можно нажать на Debug и начнется отладка.

Внимание: для отладки во внешней памяти может использоваться только 2 точки останова.

Запуск загрузки и отладки одной кнопкой для QSPI

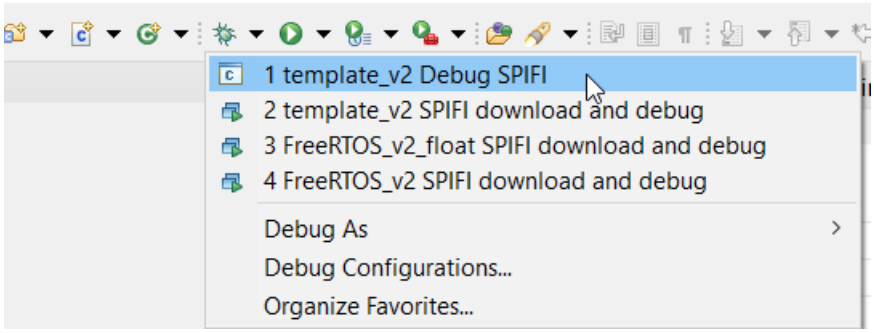
Более привычно запускать одновременную прошивку памяти и отладку, а вместе с ними и автоматическую сборку проекта нажатием одной кнопки в меню или иконки на панели инструментов. И настроить это можно. Для этого снова зайдите в Конфигурационный диалог и создавайте Launch Group таким образом (в примере уже собрана такая конфигурация):



Обратите внимание, как и где выбраны свойства «Mode» и «Action».

При нажатии «Debug» произойдет последовательная сборка, прошивка и запуск дебаггера.

В дальнейшем выбирайте в меню или в иконке Debug нужный Launch Group и выполнится сборка, загрузка и вход в отладку.



Конец.