Быстрый старт с микроконтроллером К1921ВГ015

(версия от 16.07.2024г.)

Содержание

1 Описание микроконтроллера К1921ВГ015

Микросхема К1921ВГ015 представляет собой СБИС 32-разрядного микроконтроллера на базе ядра RISC-V, предназначенного для промышленных и потребительских приложений, включая системы дистанционного мониторинга, контрольно-измерительные приборы, системы автоматизации производственных процессов, автомобильную электронику, а также устройства с батарейным питанием.

1.1 Состав микроконтроллера К1921ВГ015

В состав микроконтроллера входят функциональные элементы:

- 32-разрядное ядро архитектуры RISC-V с поддержкой системы команд RV32IMFCN_ZBA_ZBB_ZBC_ZBS, набора команд умножения, арифметических и логических команд, встроенным модулем обработки команд с плавающей запятой с одинарной точностью FPU, кэшем команд и поддержкой отладочного интерфейса JTAG;
- блок управления сбросом и синхронизацией RCU, имеющий в своем составе RC генератор (1 МГц), синтезатор частоты SYSPLL и блок управления системными тактовыми сигналами SCM;
- системный блок управления энергопотреблением PMUSYS;
- блок управления энергопотреблением в составе с RTC модулем (PMURTC);
- блок коммутации AXI AHB;
- Flash-память объемом 1 Мбайт;
- SRAM0 (ОЗУ0) объемом 256 Кбайт;
- SRAM1 (ОЗУ1), подключенное к домену батарейного питания, объемом 64 Кбайт;
- 24-канальный контроллер прямого доступа к памяти DMA;
- блок часов реального времени RTC со входами контроля целостности;
- датчик температуры TSENSOR;
- сторожевой таймер WDT;
- независимый сторожевой таймер IWDT;
- один 8-канальный 12-разрядный быстродействующий АЦП с режимами цифрового компаратора для каждого из каналов (ADCSAR);
- один 8-канальный 16-разрядный сигма-дельта АЦП (ADCSD);
- три 16-разрядных порта ввода-вывода A, B, C;
- восемь аналоговых входов, подключенных к каналам АЦП (ADCSD и ADCSAR);
- один 32-разрядный таймер TMR32;
- три 16-разрядных таймеров TMR0 – TMR2;
- пять приемопередатчиков UART0 – UART4;
- блок криптографии CRYPTO;
- два блока вычисления CRC (CRC0, CRC1);
- генератор случайных чисел (TRNG);
- HASH процессор;
- контроллеры интерфейсов:
- CAN 2.0b;
- USB 2.0 FullSpeed (Device);
- один контроллер I2C;
- один контроллер QSPI;
- два контроллера SPI (SPI0 – SPI1).

2 Подготовка к работе

Для запуска Bare Metal кода на RISC-V понадобится:
• RISC-V toolchain: xpack-riscv-none-elf-gcc;
• Система сборки: make, cmake, platformio, Eclipse и тп.;
• startup файл;
• Скрипт компоновщика (.ld).

2.1 RISC-V toolchain

Флаг –march

Флаг **–march** определяет набор расширений, поддерживаемый текущим процессором.

Существующие расширения:

• **RV32I**: Стандартный набор целочисленных инструкций. Содержит набор из 32 регистров 32-бит.

• **RV32E**: ISA для встраиваемых систем. Совпадает с RV32I, но содержит только 16 регистров.

• **RV64I**: 64-бит версия RV32I.

• **M** - целочисленное умножение/деление.

• **A** - атомарные операции с памятью.

• **F/D** - вычисления с плавающей точкой одинарной/двойной точности.

• **C** - сжатый формат команд 16-бит.

• **Zicsr** – Инструкции доступа к Control and Status Register (CSR).

• **zb*** - битовые операции.

Для К1921ВГ015 флаг **–march** выглядит следующим образом:

–march=rv32imfc_zba_zbb_zbc_zbs_zicsr

Флаг –mabi

Флаг **–mabi** определяет используемый ABI:

• **ilp32** - int, long, и указатели имеют длину 32-бит. long long длину 64-бит, char длину 8-бит, и short длину 16-бит.

• **Lp64** - long и указатели имеют длину 64-бит, но int длину 32-бит. Остальные типы такие же как в ilp32.

• **””** (пустая) – Целочисленные аргументы функций передаются в регистрах, с плавающей точкой через стек.

• **f**: 32-бит и меньше аргументы с плавающей точкой передаются через регистры FPU. Данная ABI требует поддержку F расширения.

• **d**: 64-бит и меньше аргументы с плавающей точкой передаются через регистры FPU. Данная ABI требует поддержку D расширения

Для К1921ВГ015 флаг **–mabi** выглядит следующим образом:

–mabi=ilp32f

На рисунке ниже приведены установки флагов –**march** и –**mabi** в среде Syntacore IDE.

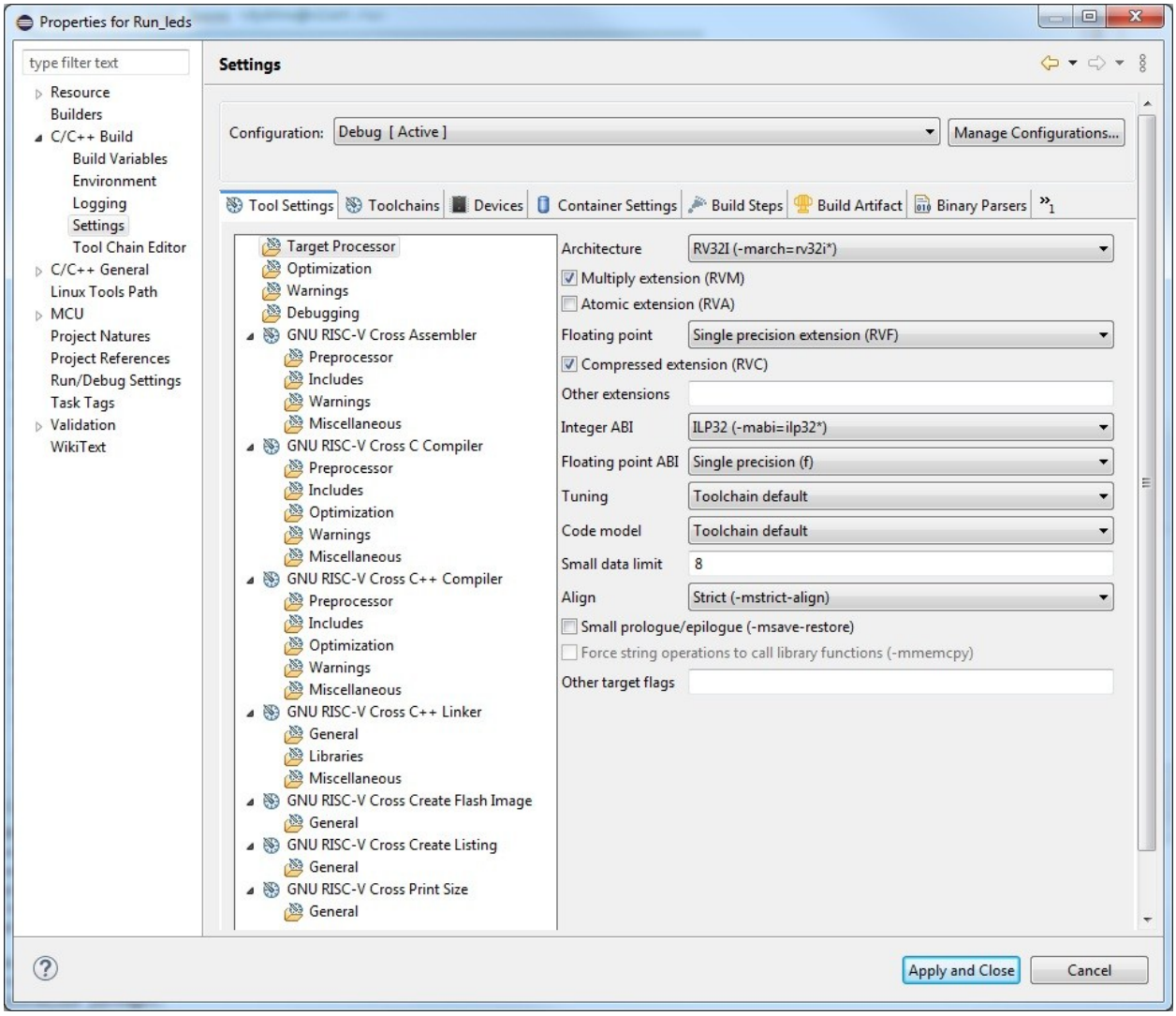

Рисунок 2.1 – Установки флагов –march и –mabi

2.2 Аппаратные отладчики JTAG

Для обеспечения взаимодействие между интегрированной средой разработки Eclipse (с GCC компилятором для RISC-V и системой отладки OpenOCD), установленной на персональном компьютере, и отладочными ресурсами, встроенными в микроконтроллер К1921ВГ015 можно использовать адаптер Olimex ARM-USB-OCD-H, J-Link или отладчик, построенный на ИС FT2232.

Также можно использовать любой отладчик поддерживающий интерфейс JTAG и имеющий драйвер для OpenOCD.

3 Настройка среды Syntacore IDE под Widnows

Для разработки проектов, а также запуска примеров в среде Syntacore IDE потребуются:

– Набор инструментов для разработки Syntacore Development Toolkit (можно скачать по ссылке: https://syntacore.com/page/products/sw-tools). Имеются версии под Windows и Linux.

– Инструмент отладки OpenOCD с драйвером Flash K1921VG015 (можно скачать по ссылке: https://github.com/DCVostok/openocd-k1921vk/releases/tag/v0.12.0-k1921vk). Имеются версии под Windows и Linux.

– Набор программных средств разработки для микроконтроллеров RISC-V НИИЭТ NIIET_RISKV SDK (расположены по ссылке: https://gitflic.ru/project/niiet/niiet_riscv_sdk).

Скачать можно с помощью git-bush используя ссылку https://gitflic.ru/project/niiet/k1921vkx_sdk.git. Необходимая ветка – master.

3.1 Подготовка среды Syntacore IDE для работы с МК К1921ВГ015

Шаг 1 – Установка Syntacore Development Toolkit

Переходим по ссылке https://syntacore.com/page/products/sw-tools, скачиваем архив **sc-dt-2023.11-win.zip** и распаковываем в любом удобном для работы месте.

Шаг 2 – Скачивание NIIET_RISKV SDK

Скачивание осуществляется с помощью git-bush. Консольной командой **cd** выбираем директорию, куда будет скачан пакет SDK.

Далее для скачивания в консоли git-bush вводим команду: **git clone -b master https://gitflic.ru/project/niiet/niiet_riscv_sdk.git**

Набор программных средств разработки для микроконтроллеров RISC-V НИИЭТ NIIET_RISKV SDK будет скачан в выбранную ранее директорию.

Структура каталогов NIIET_RISKV SDK:

– **platform**: Общие библиотеки.
  - Device: Заголовочные файлы микроконтроллера, файлы startup и скрипты линкера.

– **projects**: Примеры проектов.
  - NIIET-DEV-K1921VG015: Проекты для отладочной платы NIIET-DEV-K1921VG015.

– **tools**: Вспомогательный инструментарий.
  - openocd : Файлы для осуществления отладки мк.
  - svd : SVD файлы микроконтроллеров.
  - sc-dt_Patch_Niiet_Win32.zip : Архив для поддержки микроконтроллера К1921ВГ015 в Syntacore Development Toolkit.

Шаг 3 – Интеграция поддержки МК К1921ВГ015 средой Syntacore IDE

Нам понадобится архив **sc-dt_Patch_Niiet_Win32.zip** из папки **tools**. Его необходимо распаковать в каталог **sc-dt** ранее скачанной среды Syntacore IDE (с подтверждением замены файлов). Также после распаковки необходимо проверить наличие файла **k1921vg015.cfg** в каталоге: \sc-dt\tools\share\openocd\scripts\target.

Шаг 4 – Запуск и настройка среды Syntacore IDE

Запуск среды осуществляется с помощью скрипта **start-scr-ide.cmd** находящегося в каталоге **sc-dt**. При первом запуске появится окно выбора рабочей области workspace.

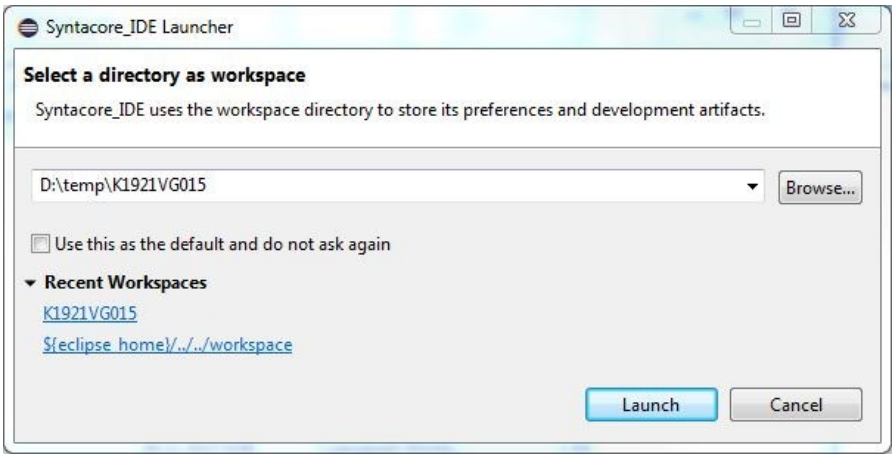

Рисунок 3.1 – Настройка workspace

В окне настройки рабочей области (рисунок 3.1) нужно выбрать и указать путь к рабочему пространству, где будут находиться проекты по работе с микроконтроллером К1921ВГ015. Можно расположить в любом удобном месте.

Далее, чтобы добавить готовые проекты из NIIET_RISKV SDK нужно переместить их из каталога niiet_riscv_sdk\projects\NIIET-DEV-K1921VG015 в созданный каталог workspace и импортировать их в рабочее пространство. Для импорта проектов заходим в раздел File/Import и выбираем Projects from Folder of Archive.

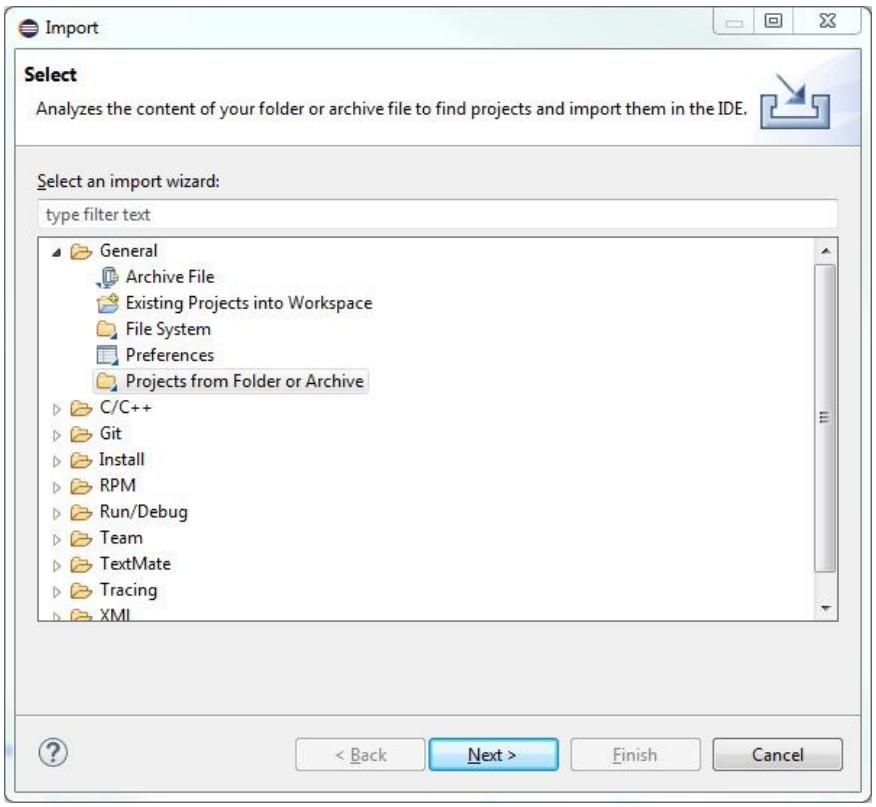

Рисунок 3.2 – Импорт проектов в рабочую область

Далее выбираем необходимые для импорта проекты (см. рисунок 3.3).

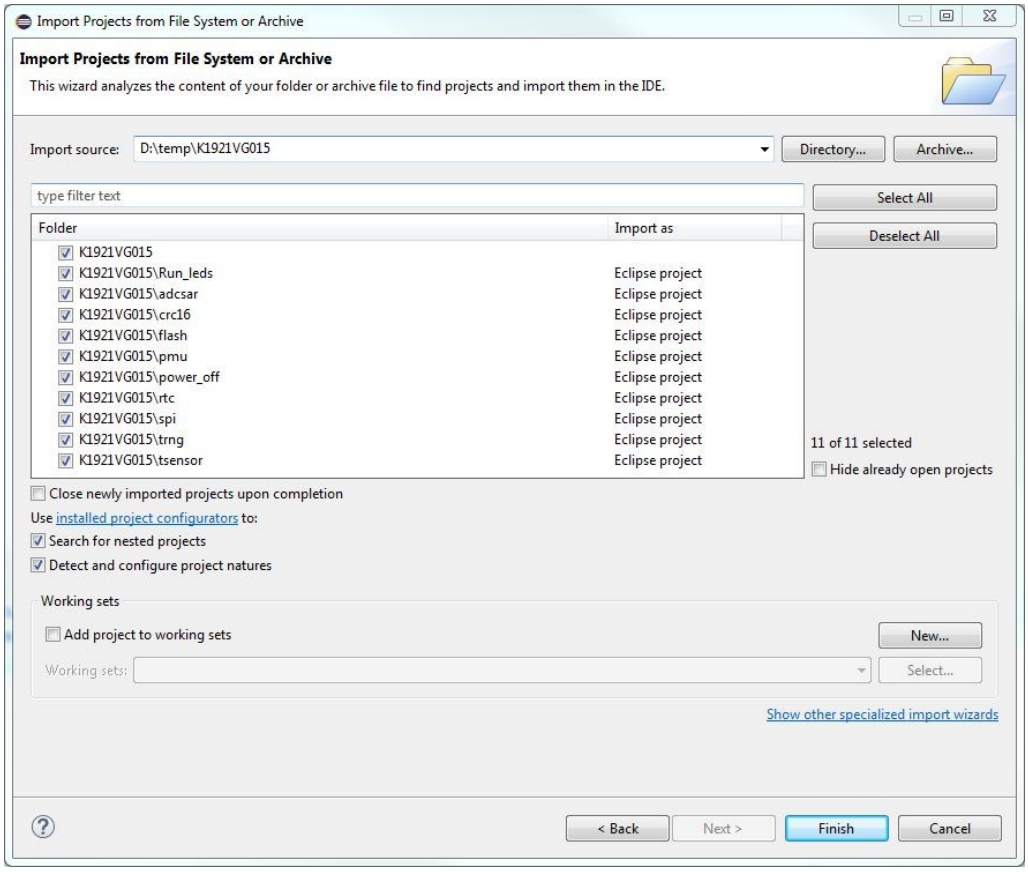

Рисунок 3.3 – Выбор проектов для импорта в рабочую область

После импортирования выбранные проекты появятся в окне Project Explorer (см. рисунок 3.4).

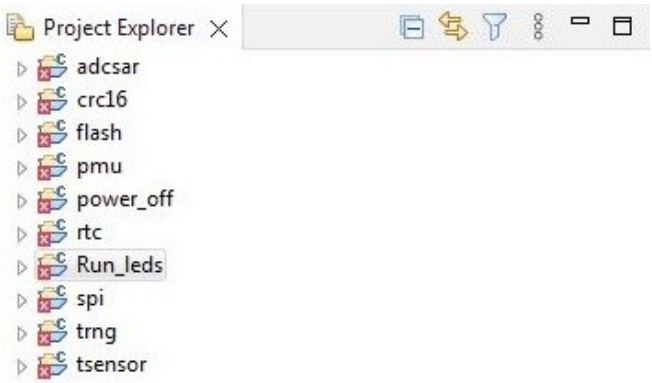

Рисунок 3.4 – Окно с импортированными проектами

Шаг 5 – Сборка проекта для К1921ВГ015

Для того, чтобы собрать проект с примером необходимо предварительно зайти в настройки проекта (нажатием правой кнопки мыши по названию проекта в окне Project Explorer и выбором во всплывающем окне кнопки Properties). Далее нужно выбрать раздел “C/C++ Build”, вкладку “Settings / Tool Settings” и затем ничего не изменяя нажать внизу окна кнопку “Apply and Close”. Это необходимо для обновления и применения настроек проекта (см. рисунок 3.5).

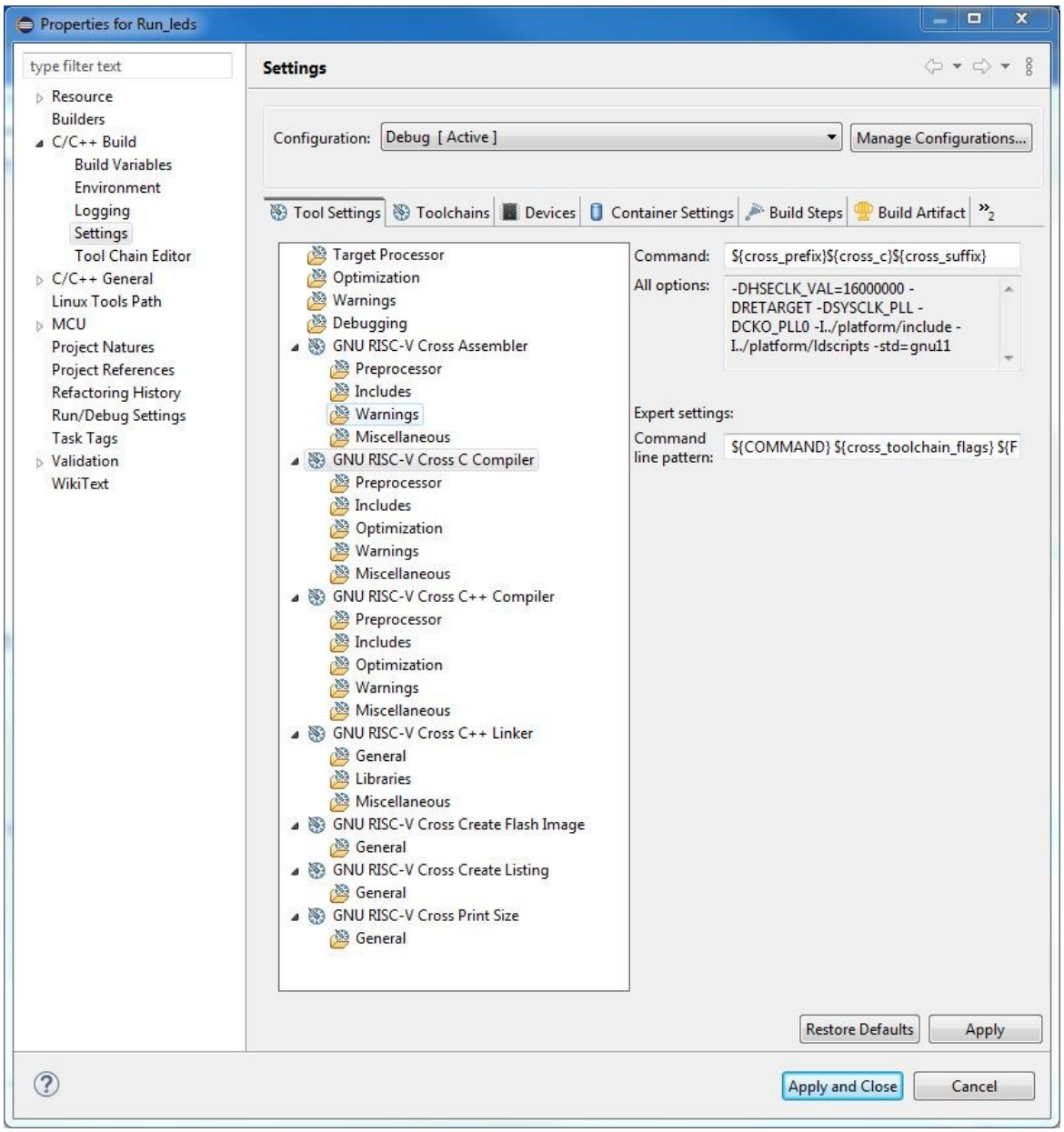

Рисунок 3.5 – Настройки проекта в Syntacore IDE

После обновления настроек проекта можно собрать проект (нажатием правой кнопки мыши по названию проекта в окне Project Explorer и выбором во всплывающем окне кнопки Build Project). О результате сборки можно узнать из окна консоли, пример указан на рисунке 3.6.

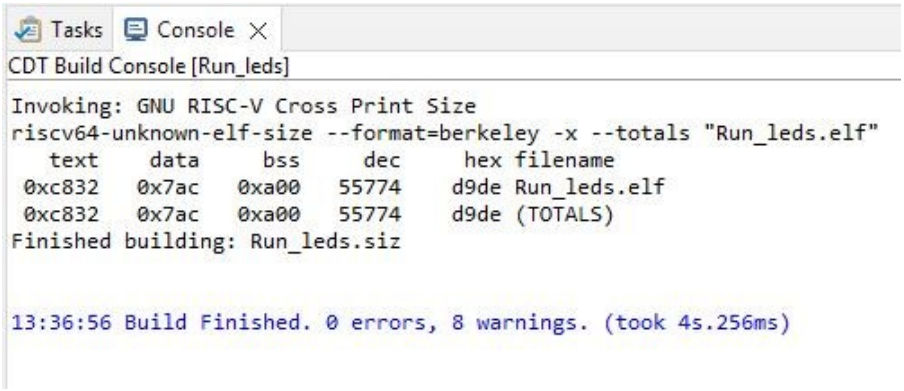

Рисунок 3.6 – Результаты сборки проекта

Шаг 6 – Установка драйверов JTAG-эмулятора

Чтобы установить драйверы для JTAG-эмулятора (таких как «J-Link» или отладчик, построенный на ИС FT2232), совместимых с используемым в Syntacore IDE отладчиком «OpenOCD» необходима программа «Zadig». Скачать ее можно по ссылке: https://zadig.akeo.ie.

Подключите JTAG-адаптер к USB-порту вашего ПК. Дождитесь, пока закончится процедура автоматической установки драйверов средствами Windows. Независимо от успешности её результата по окончании запустите программу «Zadig» (см. рисунок 3.7).

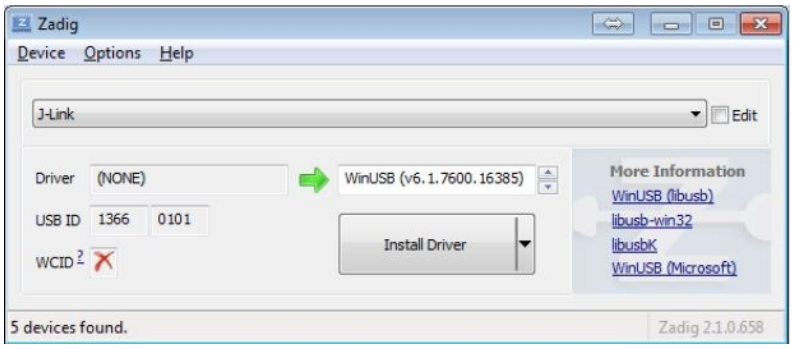

Рисунок 3.7 – Главное окно программы Zadig

В меню «Options» поставьте галочку «List all devices». В выпадающем списке выберите используемое устройство (в нашем примере «J-Link») или неизвестное устройство (убедитесь в таком случае, что это именно jtag вынимая/вставляя USB кабель). В строке справа от стрелки выберите драйвер «WinUSB (v.xxxx)». Нажмите на кнопку «Install Driver» (если кнопка называется по-другому, значит у вас установлены другие драйвера и нужно нажать «Replace Driver»). Драйвера установлены. Однако по опыту эксплуатации рекомендуется вынуть/вставить JTAG из USB компьютера – без этого драйвер может не заработать. В некоторых случаях требуется перезагрузка. В диспетчере устройств на вкладке с устройством JTAG по кнопке «сведения» должно открываться окно с похожим содержимым, указанном на рисунке 3.8.

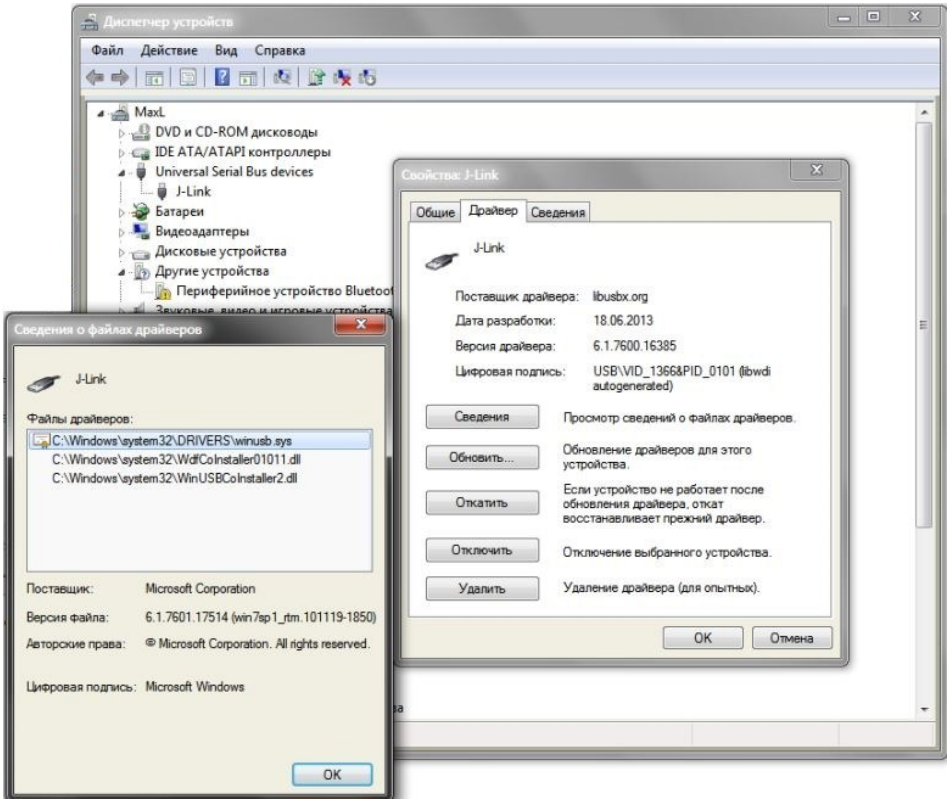

Рисунок 3.8 – Сведения о файлах драйверов

Из рисунка видно, что использован драйвер «WinUSB». Если драйвер не установился или не работает, можно попробовать нажать на кнопку «Install Driver» несколько раз (если «Zadig» сообщает, что установка не удалась), вынуть/вставить JTAG из USB, перезагрузить компьютер, отключить антивирус, запустить программу с правами администратора, включить службу "Центр обновления Windows" (если она отключена), проверить состояние устройства в диспетчере устройств Windows. Чтобы вернуться к стандартным драйверам JTAG для работы с другой средой, в диспетчере устройств Windows найдите свое устройство (например J-Link), правой кнопкой → Обновить драйверы... → Выполнить поиск драйверов на этом компьютере → Выбрать драйвер из списка уже установленных" и выбрать, например, "J-link driver".

Примечание: при замене драйверов на Windows 10 с помощью программы «Zadig» следует соблюдать осторожность, поскольку можно заменить общий (generic) драйвер, который Windows может автоматически использовать при подключении нового устройства. Generic-драйвер может одновременно использоваться несколькими USB-устройствами. После такой процедуры остальные устройства перестанут нормально функционировать.

Шаг 7 – Настройка и запуск отладочной сессии для К1921ВГ015

Для настройки отладочной сессии для работы с микроконтроллером К1921ВГ015 сначала нужно включить режим перспективы “Debug” кнопкой “Open Perspective” в правом верхнем углу рабочего окна Syntacore IDE (см. рисунок 3.9).

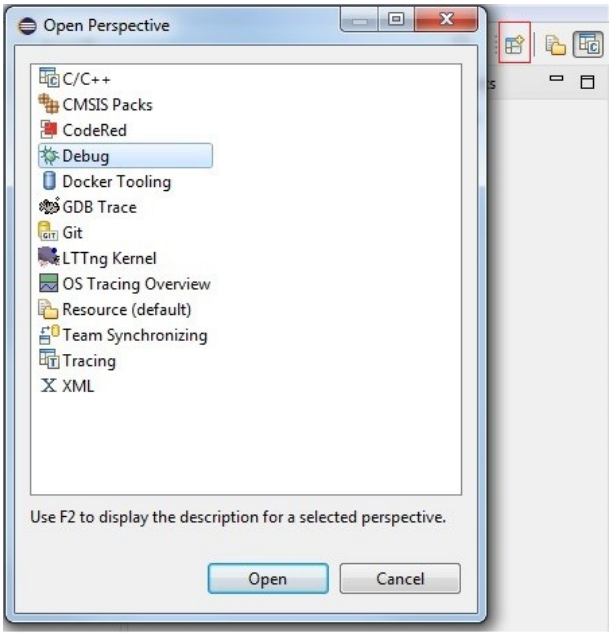

Рисунок 3.9 – Включение режима Debug

Затем нужно зайти в настройки отладки выбранного проекта (см. рисунок 3.10).

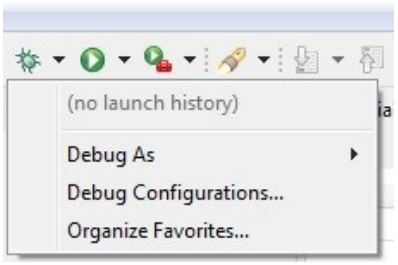

Рисунок 3.10 – Настройка конфигурации отладки

При открытии окна Debug Configurations двойным щелчком мыши по вкладке “GDB OpenOCD Debugging” создаем отладочную конфигурацию выбранного проекта (см. рисунок 3.11).

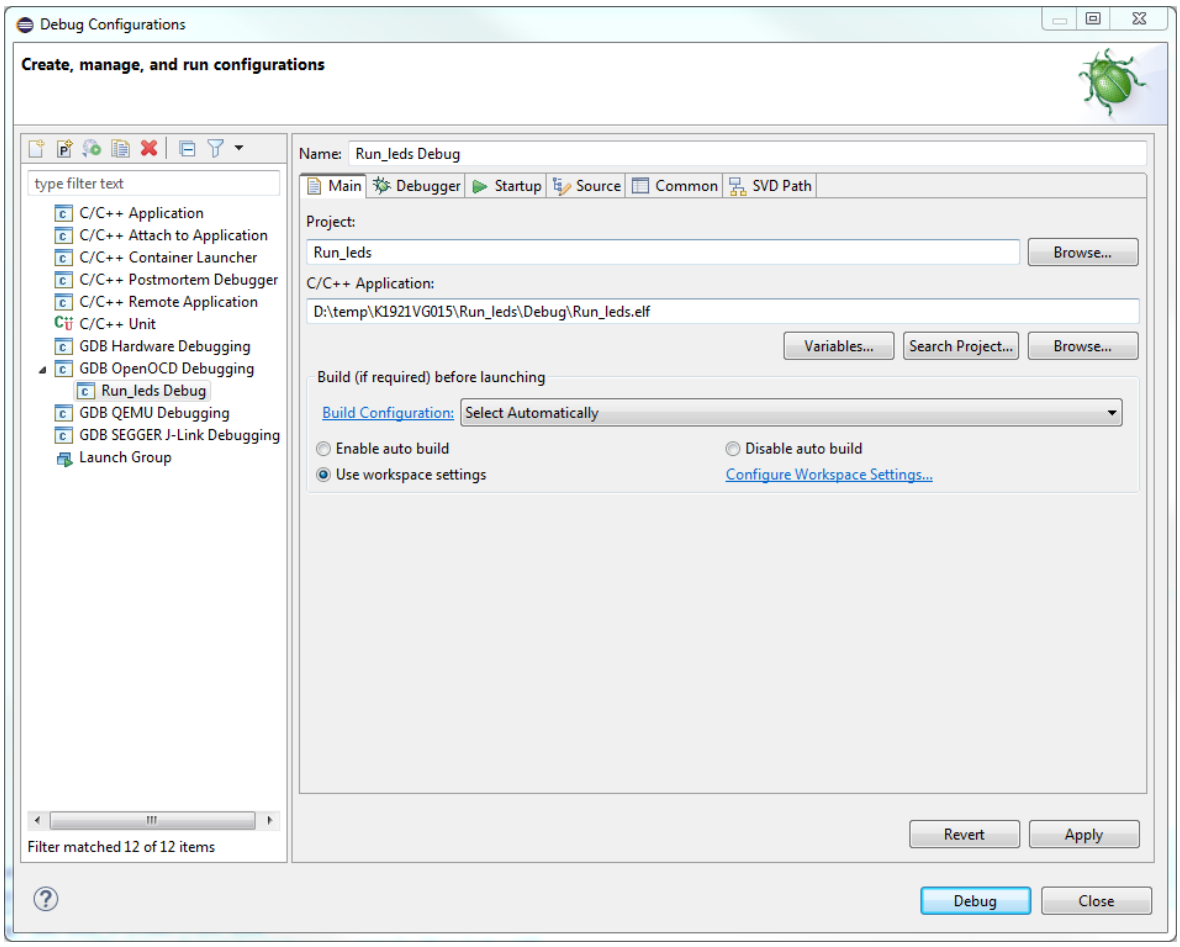

Рисунок 3.11 – Создание отладочной конфигурации проекта

В строке “C/C++ Application” нужно указать путь к **.elf** файлу, созданному при сборке проекта. Набор опций конфигурации отладчика находится на вкладке “Debugger” в разделе “Config options” (см. рисунок 3.12).

Состав набора опций конфигурации:

-s ${eclipse_home}/../tools/share/openocd/scripts
-s ${eclipse_home}/../tools/share/openocd/scripts/interface/ftdi
-s ${eclipse_home}/../tools/share/openocd/scripts/interface
-s ${eclipse_home}/../tools/share/openocd/scripts/target
-f jlink.cfg
-f k1921vg015.cfg
-c "init;halt"

Если используется отладчик, отличный от jlink, тогда в строке -f jlink.cfg вместо jlink.cfg необходимо указать конфигурационный файл используемого отладчика. Путь к местонахождению конфигурационных файлов: sc-dt/tools/share/openocd/scripts/interface.

При использовании макетно-отладочной платы VG015 DK разработки ДЦЭ Восток, с установленной ИС FTDI на плате, необходимо использовать конфигурационный файл vg015_dev_onboard_ftdi.cfg. В таком случае заменяем строку «-f jlink.cfg» на «-f vg015_dev_onboard_ftdi.cfg».

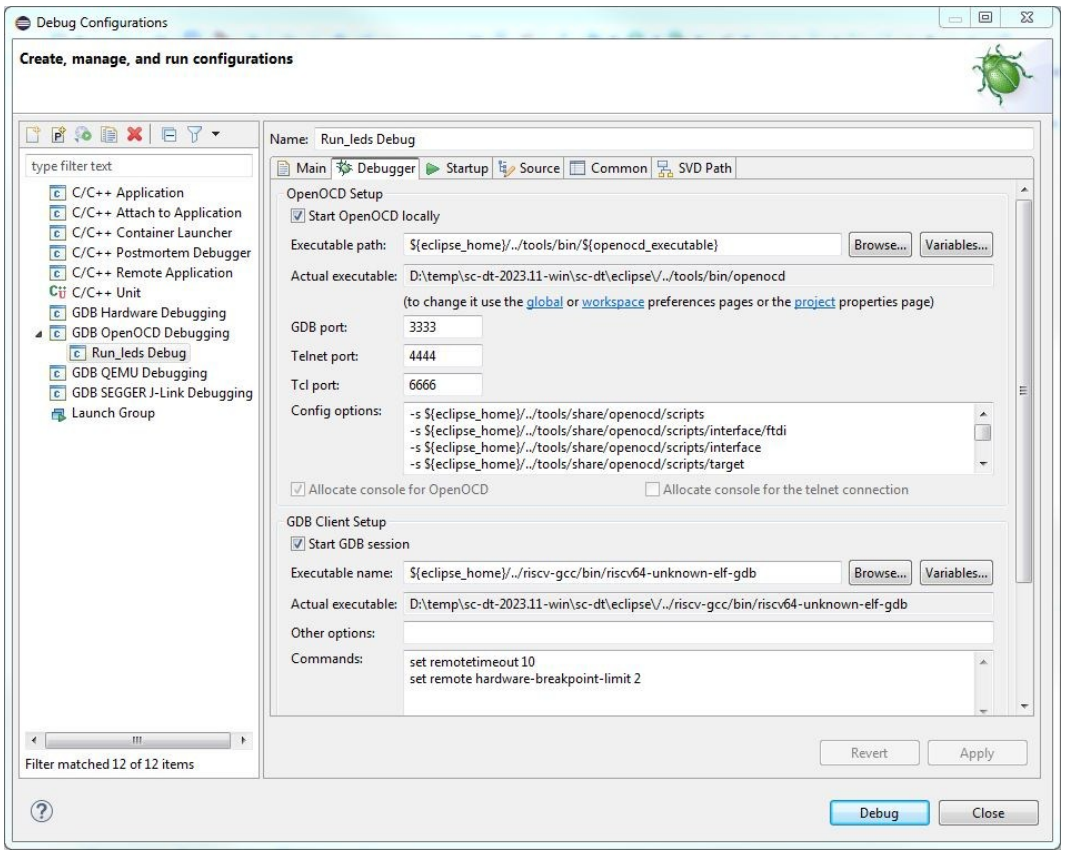

Рисунок 3.12 – Набор конфигурации отладчика

Теперь при нажатии кнопки Debug собранный проект будет записан в микроконтроллер и запущена сессия отладки. В дальнейшем отладку можно будет запускать быстрее, используя историю запусков (см. рисунок 3.13).

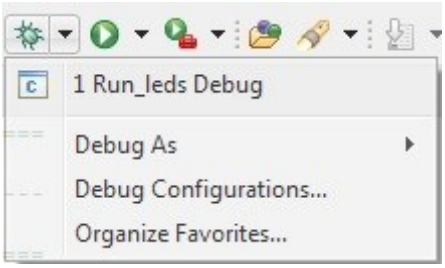

Рисунок 3.13 – Запуск ранее сконфигурированной отладочной сессии

Шаг 8 – Создание нового проекта для работы с К1921ВГ015

При создании нового проекта для работы с микроконтроллером К1921ВГ015 наиболее простым способом будет копирование и корректировка одного из примеров.

Для этого нужно взять один из примеров проектов, очистить его (нажатием правой кнопки мыши по названию проекта в окне Project Explorer и выбором во всплывающем окне кнопки Clean Project) и затем скопировать его (нажатием во всплывающем окне кнопки Copy, а затем Paste), и наконец выбрать название для нового проекта. Пример создания копии проекта изображен на рисунке 3.14.

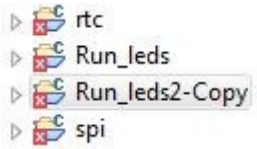

Рисунок 3.14 – Создание копии проекта

В новом созданном проекте большинство настроек будут заимствованы из скопированного (родительского) проекта. Необходимо будет только перед запуском отладки создать отладочную конфигурацию для нового проекта, в строке “C/C++ Application” которой указать путь к **.elf**-файлу, созданному при сборке нового проекта (см. рисунок 3.11 из раздела «Шаг 7…»). Настройки из подраздела “Config options” добавлять не потребуется, поскольку они будут заимствованы из скопированного проекта. Чтобы во время отладки получить доступ к регистрам периферийных блоков микроконтроллера необходимо при конфигурации отладки указать путь к **.svd**-файлу (см. рисунок 3.15). Сам **.svd**-файл можно взять в каталоге: niiet_riscv_sdk\tools\svd.

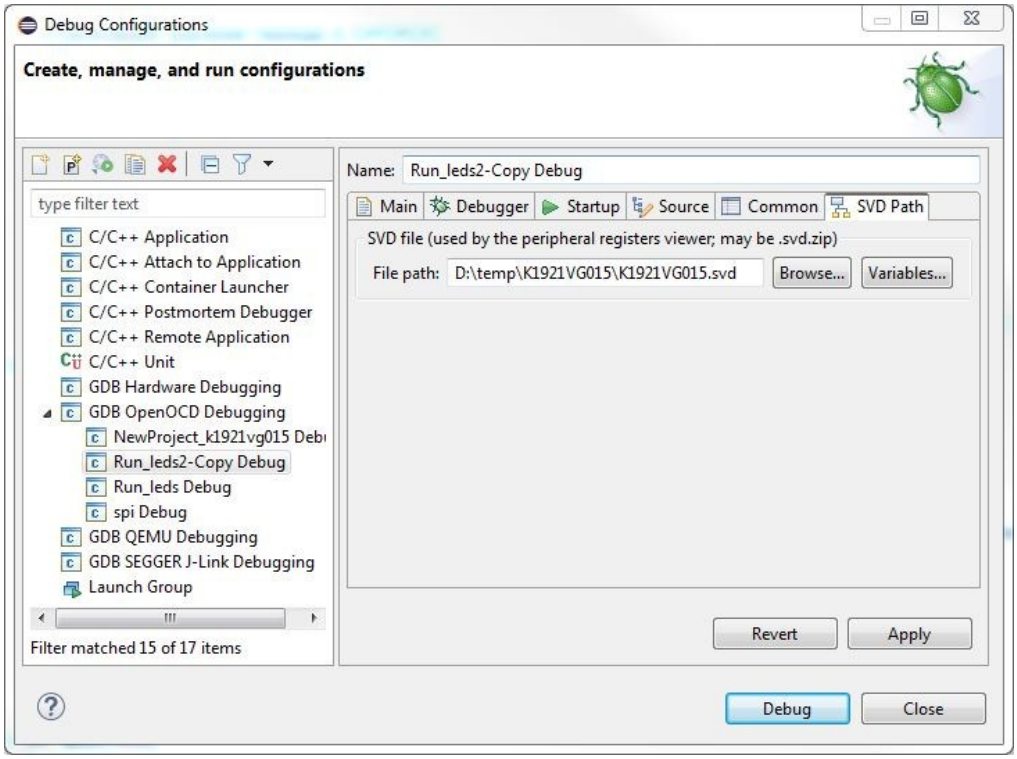

Рисунок 3.15 – Подключение SVD-файла

Если требуется создать новый проект с нуля потребуется:

1. Создать новый проект, используя набор инструментов RISC-V Cross GCC (см. рисунок 3.16).

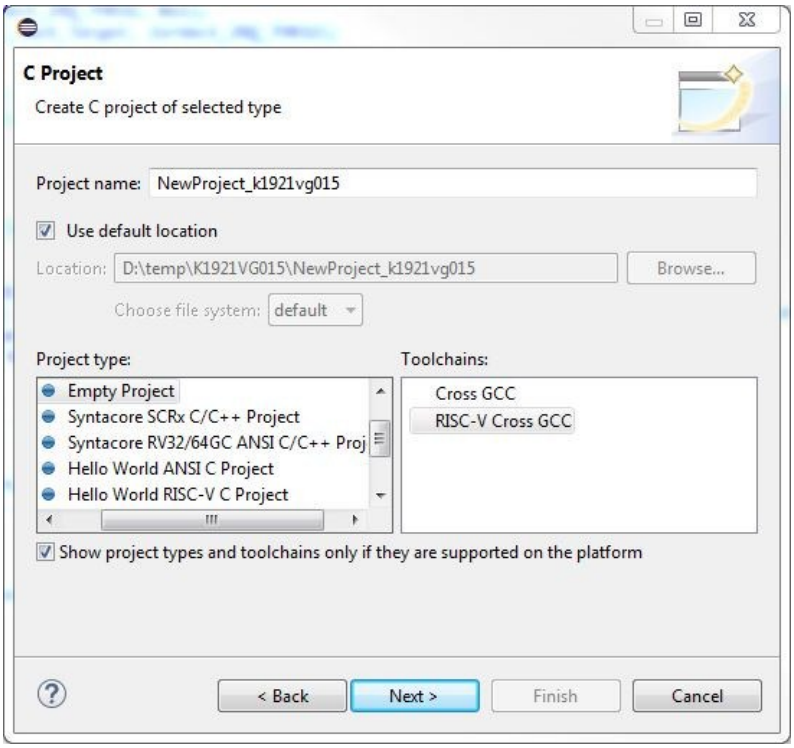

Рисунок 3.16 – Выбор Toolchains для нового проекта

2. Создать каталог под названием «platform» в каталоге с названием нового проекта в рабочей среде workspace (см. рисунок 3.17). Скопировать внутрь каталога «platform» каталоги «Include», «ldscripts», и «Source» (находятся в niiet_riscv_sdk\platform\Device\K1921VG015).

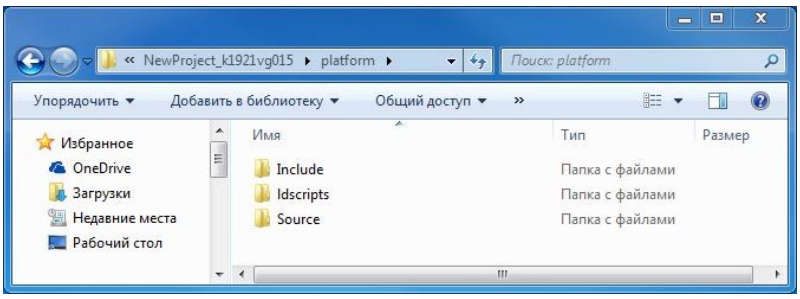

Рисунок 3.17 – Содержание каталога platform

3. Установить флаги –march и –mabi в настройках созданного проекта (см. рисунок 3.18).

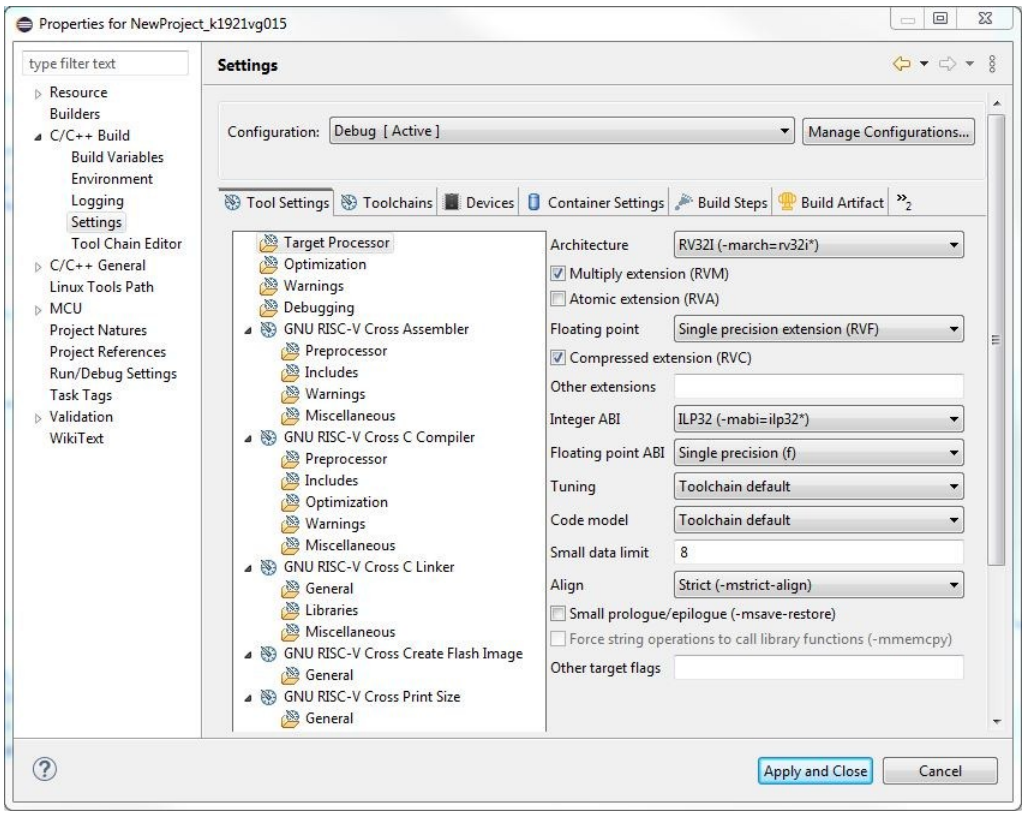

Рисунок 3.18 – Установка флагов–march и –mabi

4. Прописать пути к заголовочным файлам и скриптам в разделе GNU RISC-V Cross Assembler (см. рисунок 3.19) и в разделе GNU RISC-V Cross C Compiler (см. рисунок 3.20).

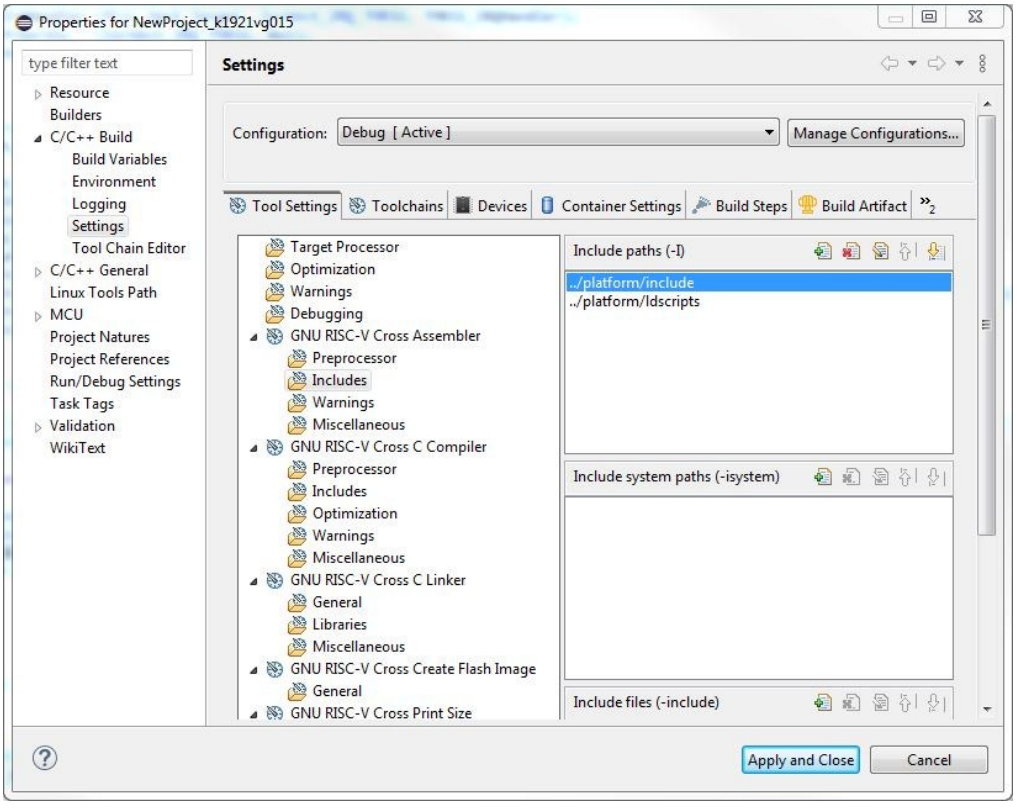

Рисунок 3.19 – Указание путей для транслятора ассемблера

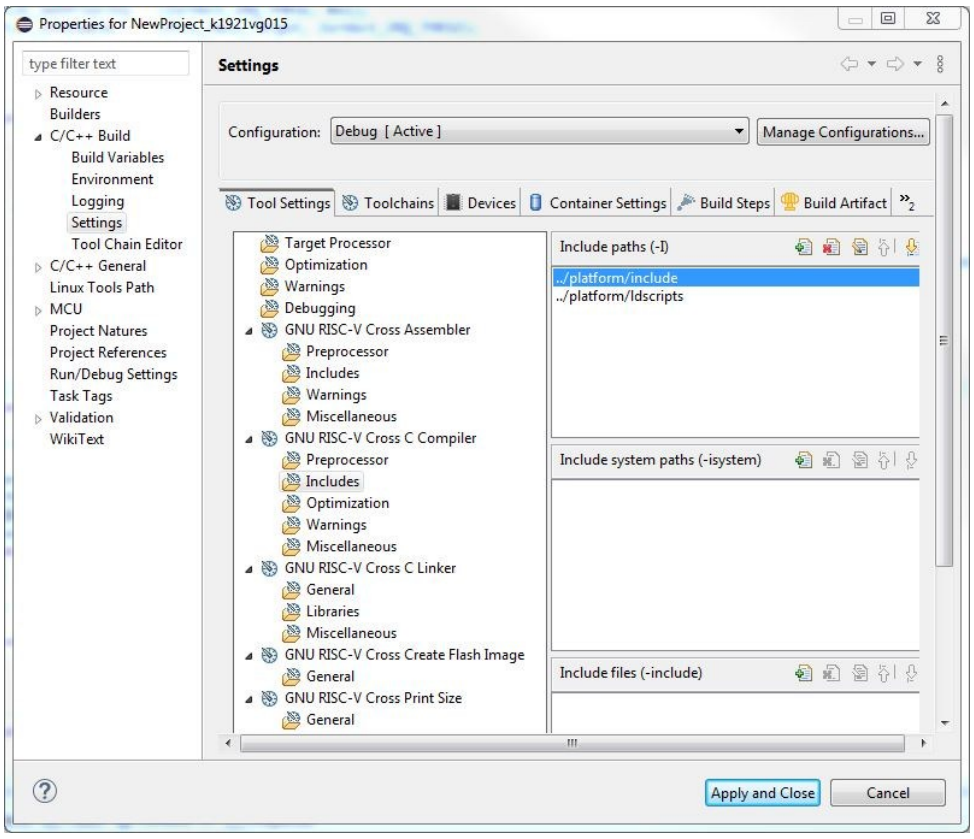

Рисунок 3.20 – Указание путей для компилятора Си

5. Указать директивы препроцессора (см. рисунок 3.21).

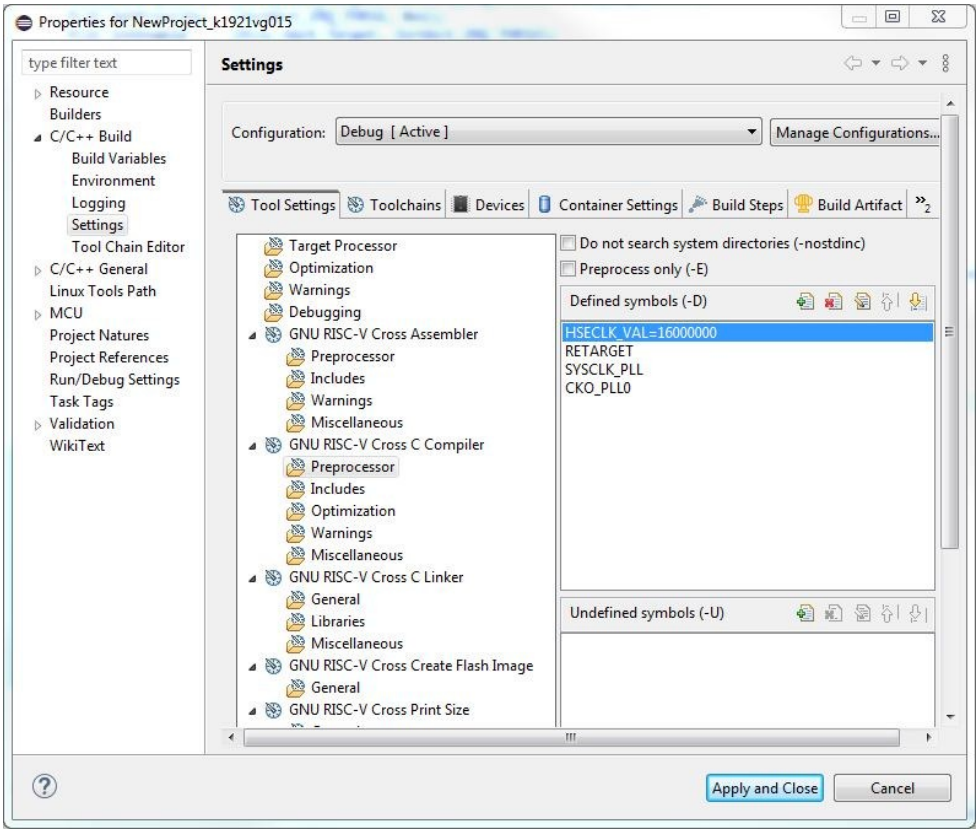

Рисунок 3.21 – Указание директив препроцессора

6. Указать путь к скрипту линкера, а также выбрать настройки линкера (см. рисунок 3.22).

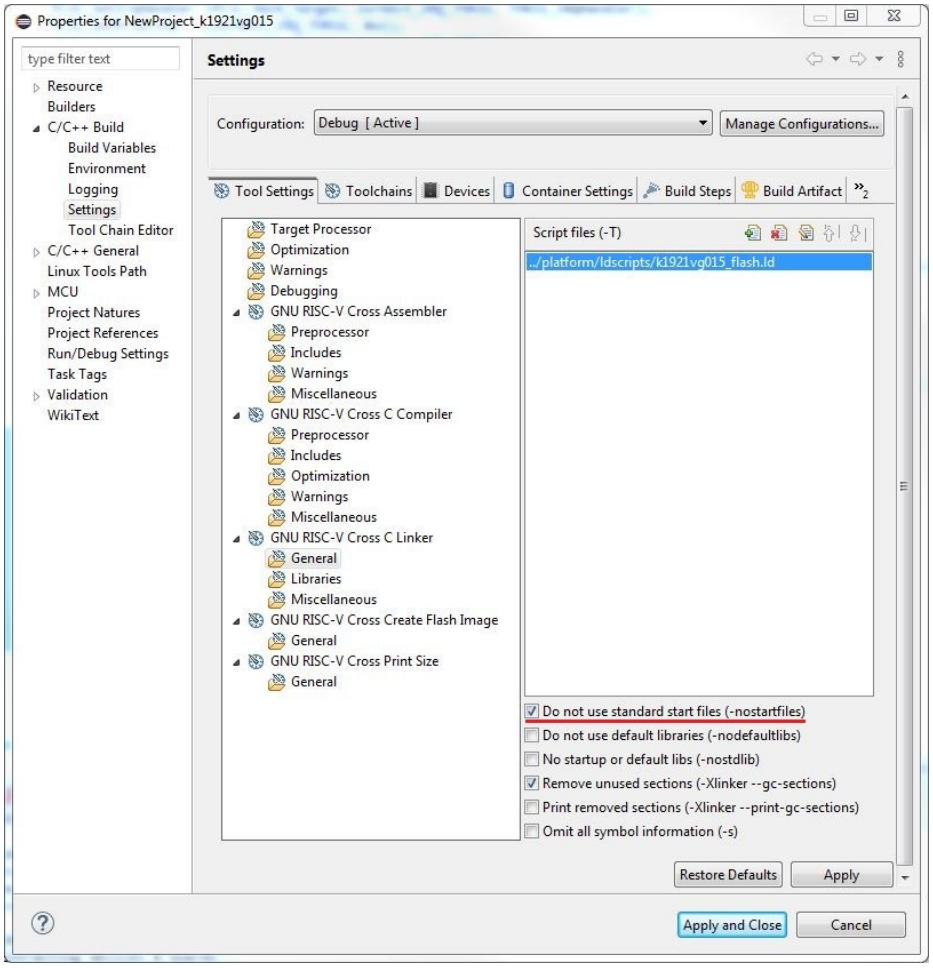

Рисунок 3.22 – Указание пути к скрипту линкера

7. Указать библиотеки линкера и путь к ним (см. рисунок 3.23).

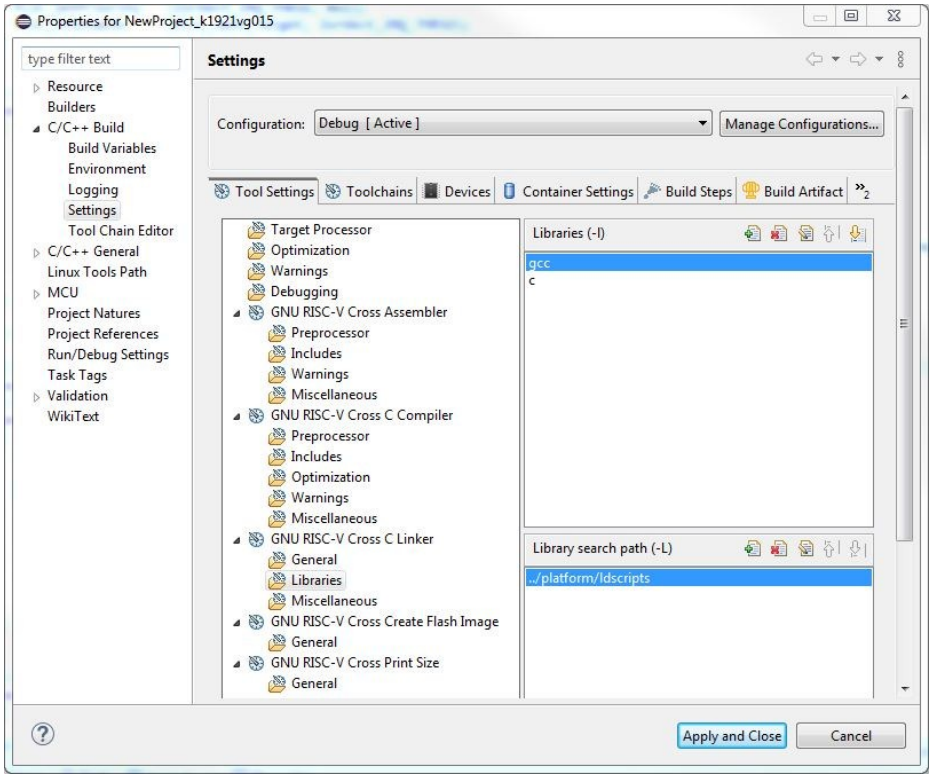

Рисунок 3.23 – Указание библиотек линкера и пути к ним

8. При необходимости формирования файла прошивки нужно разрешить его формирование в подразделе Toolchains (см. рисунок 3.24), а затем сконфигурировать настройки выходного файла прошивки (см. рисунок 3.25).

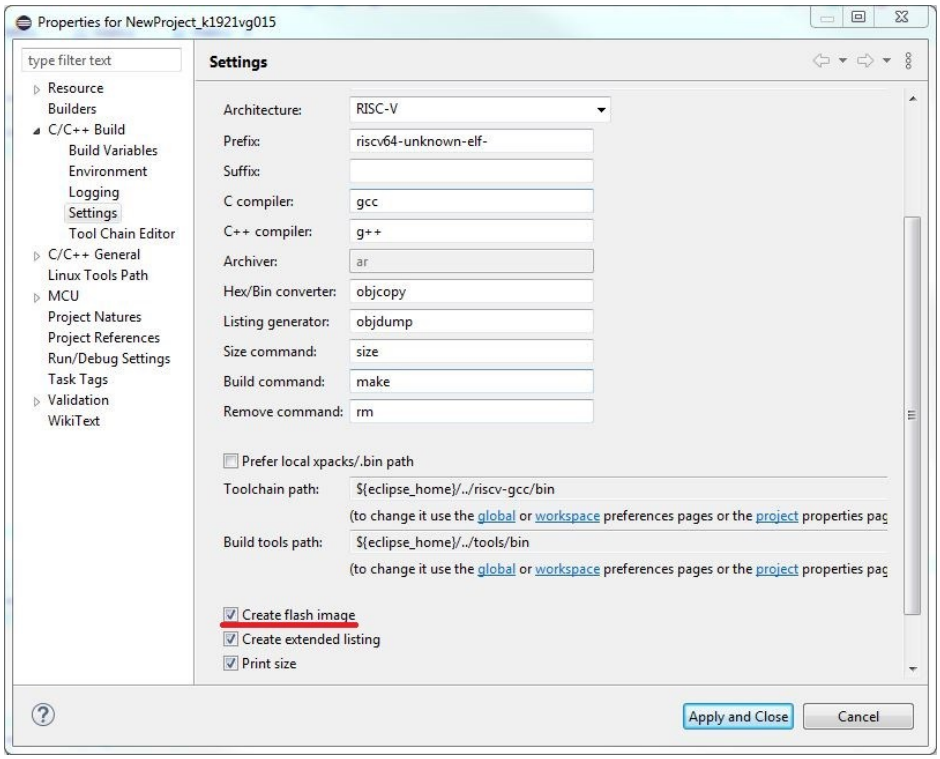

Рисунок 3.24 – Разрешение формирования выходного файла прошивки

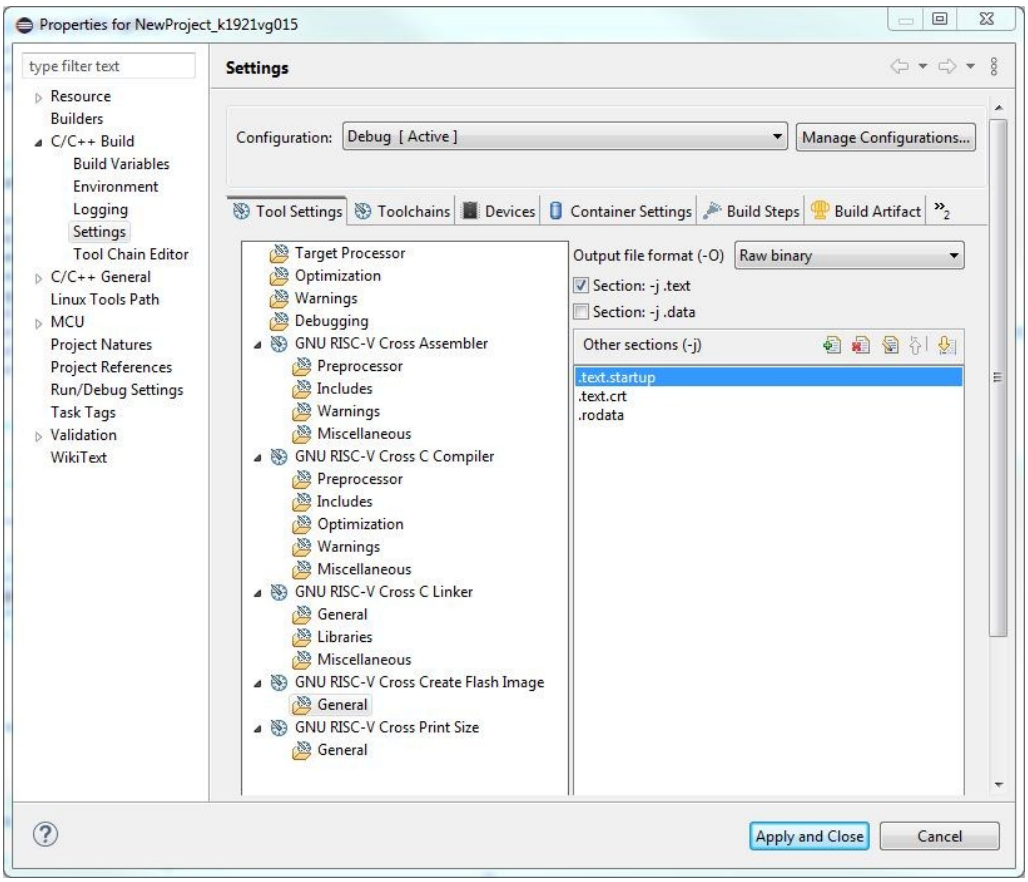

Рисунок 3.25 – Настройки формирования выходного файла прошивки

Теперь вновь созданный проект готов к сборке (см. подраздел «Шаг 5 – Сборка проекта для К1921ВГ015») и запуску отладки (см. подраздел «Шаг 7 – Настройка и запуск отладочной сессии для К1921ВГ015»).