Руководство по настройке и использованию открытого ПО совместно с микроконтроллерами ОАО “НИИЭТ” на базе ядра ARM Cortex-M4

Содержание

О руководстве

Целью данного руководства является демонстрация одного из возможных путей организации полноценного рабочего окружения на основе открытого ПО для взаимодействия с микроконтроллерами на базе ядра ARM Cortex-M4 производства ОАО «НИИЭТ».

Основными инструментами, необходимыми для разработки являются: редактор исходного кода, компилятор, программно-аппаратный инструментарий для программирования и отладки. В данном случае для реализации среды разработки предлагается использовать:

- *Qt Creator 3.4.2* – популярная и довольно мощная среда разработки на C/C++ с открытым исходным кодом, на основе которой и будет построено рабочее окружение;
- *GCC ARM Embedded 4.9* – открытый компилятор для ARM;
- *OpenOCD* – программа, необходимая для программирования и отладки, в которую была добавлена поддержка микроконтроллеров НИИЭТ на базе ядра ARM Cortex-M4.

В качестве аппаратного отладчика в примерах использовался *St-Link*, как один из наиболее дешевых и распространённых, но *OpenOCD* также позволяет использовать и другие отладочные устройства (полная информация в официальной документации на утилиту). Вопрос установки отладчиков в данном руководстве не поднимается – подразумевается, что драйвера на отладчик установлены и все необходимые настройки сделаны.

Рекомендации по настройке окружения ориентированы на микроконтроллер К1921ВК01Т и 32-битную версию Windows 7, но практически все рекомендации, за исключением этапа установки необходимых утилит, справедливы и для настройки под Linux или Mac OS, так как все использованное ПО – кроссплатформенное.

1 Установка ПО

1.1 Windows 7 x86

1.1.1 Qt Creator

Установите Qt Creator 3.4.2. Инсталлятор можно скачать с официального сайта.

1.1.2 Python

Qt требует наличия Python 2.7 для работы режима отладки. Инсталлятор можно скачать с официального сайта. Желательно добавить директорию с исполняемыми файлами в переменную среды PATH. Это можно сделать либо вручную, либо при установке включить опцию, отмеченную на рисунке 1.1.

Рисунок 1.1

1.1.3 OpenOCD

Для установки достаточно скопировать папку openocd, находящуюся в директории руководства, по любому удобному пути. Также можно добавить директорию с исполняемыми файлами в переменную среды PATH.

1.1.4 GCC ARM

Набор из компилятора и сопутствующих утилит можно скачать с официального сайта. Архив распаковываем по любому простому пути, например в корень диска C (*C:\gcc-arm-4.9*). Теперь необходимо добавить путь до папки с исполняемыми файлами компилятора (например, *C:\gcc-arm-4.9\bin*) в переменную среды PATH. Сделать это можно, зайдя в *Панель управления* → *Система* → *Дополнительные параметры системы* → *Дополнительно* → *Переменные среды*.

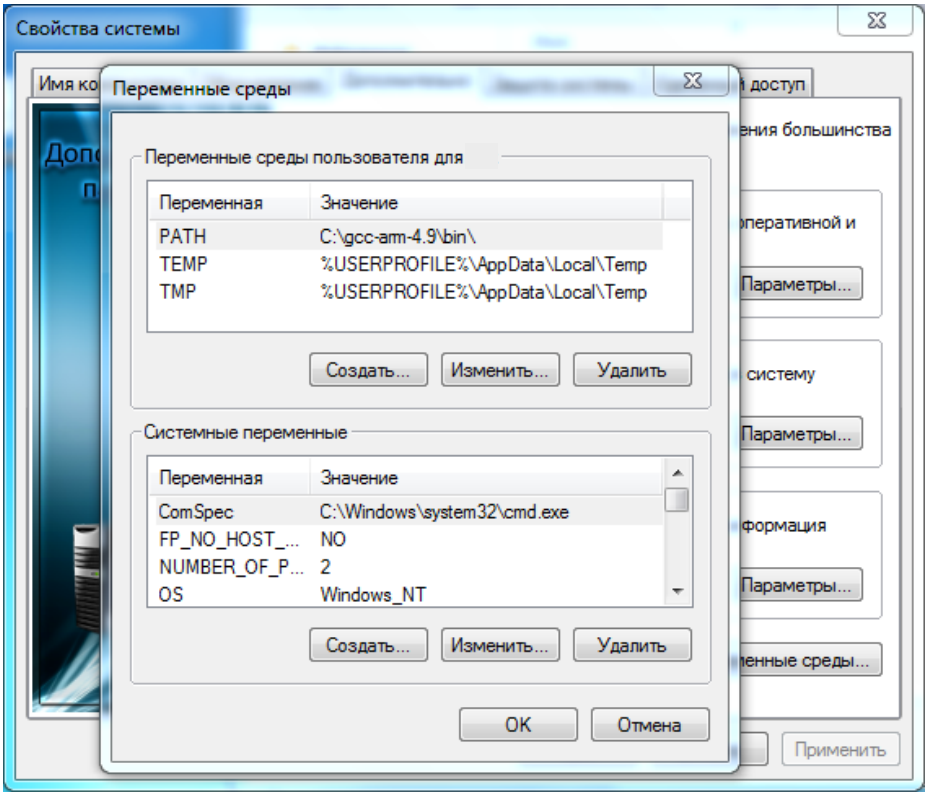

Рисунок 1.2

1.2 Linux Mint 13 (Ubuntu 12.04) x86

1.2.1 Qt Creator

Установите Qt Creator 3.4.2. Инсталлятор можно скачать с официального сайта.

1.2.2 OpenOCD

Сперва необходимо установить все зависимости. Сделать это можно, выполнив в терминале:

```
sudo apt-get install libtool autoconf texinfo libusb-dev
libusb-1.0-0-dev
```

Затем нужно получить исходный код OpenOCD. Для этого переходим в директорию, куда планируем его сохранить, и клонируем репозиторий:

```
git clone git://git.code.sf.net/p/openocd/code openocd
```

Перед компиляцией также можно добавить дополнительные скрипты с примерами подключения к нашим контроллерам. Для этого нужно скопировать папку *openocd-0.9.0/kits* из директории руководства в папку *tcl* клонированного openocd.

Теперь переходим в папку *openocd*, компилируем исходный код и устанавливаем программу:

```
cd openocd
./bootstrap
./configure --enable-maintainer-mode
make
sudo make install
```

1.2.3 GCC ARM

Архив с компилятором и сопутствующими утилитами можно скачать с официального сайта. Архив распаковываем, например в */opt*. Теперь необходимо добавить путь до папки *bin* с исполняемыми файлами компилятора в переменную среды PATH. Например, сделать это можно, добавив в конец файла ~/.profile строчку вида:

```
PATH=$PATH:<путь_до_компилятора>/bin
```

2 Настройка Qt Creator

2.1 Плагины

Qt Creator имеет гибкую систему плагинов и некоторые из них понадобятся в дальнейшей работе.

Запускаем Qt Creator и сразу заходим в *Справка → О модулях*. Необходимо убедиться в том, что плагины *QbsProjectManager* и *BareMetal* включены.

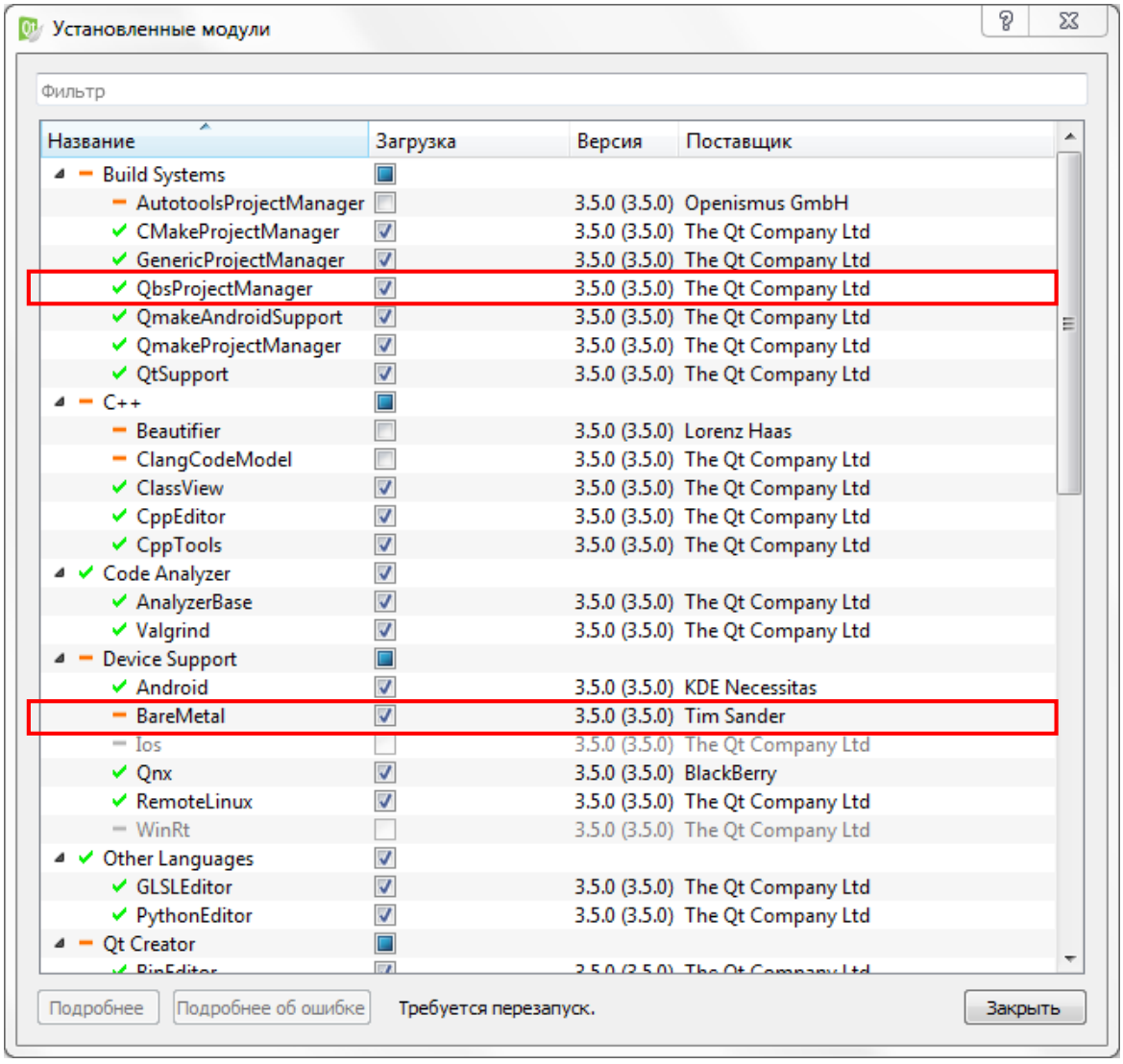

Рисунок 2.1

Если плагины были неактивны, то включаем их и перезапускаем программу.

2.2 Настройка BareMetal

Этот плагин позволяет осуществлять соединение сервера отладки, создаваемого подключенной к микроконтроллеру утилитой *OpenOCD*, со средой разработки.

Переходим в *Инструменты* → *Параметры* → *BareMetal*. Нажимаем *Добавить* → *OpenOCD*. И устанавливаем параметры сервера, согласно рисунку 2.2.

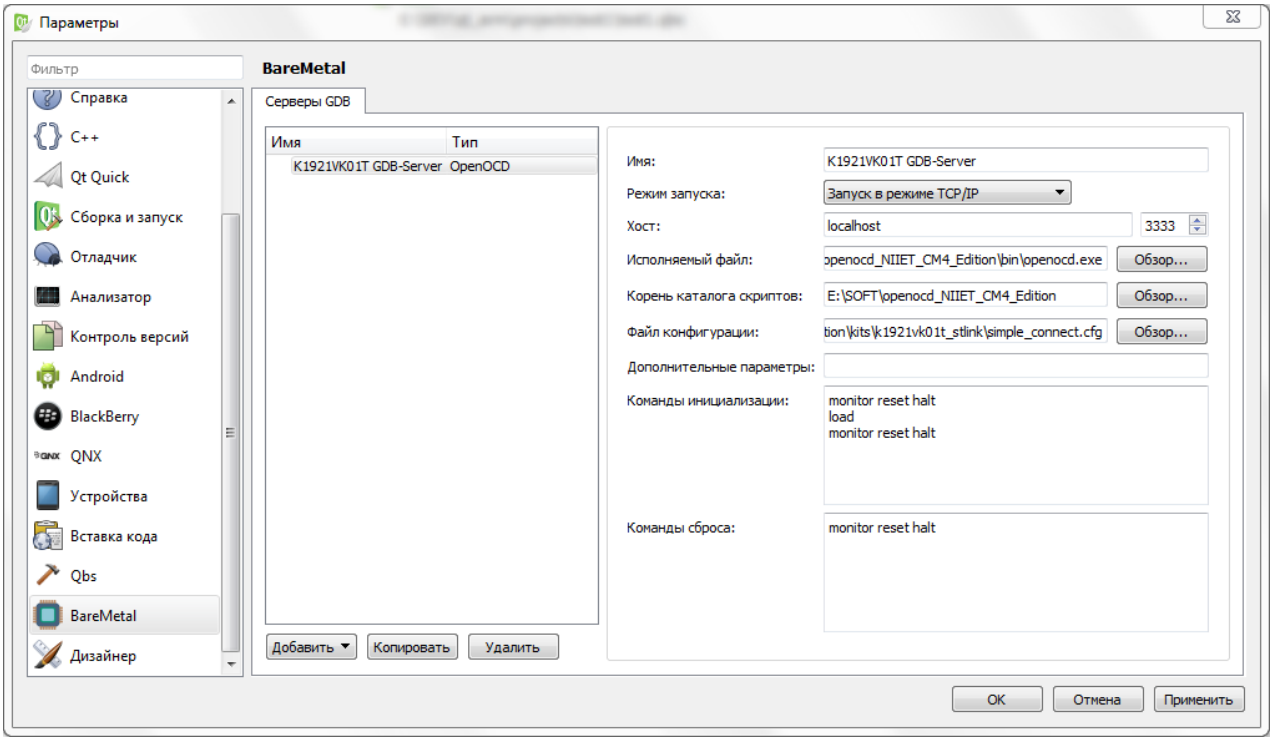

Рисунок 2.2

О некоторых из параметров стоит рассказать подробнее:

- *Исполняемый файл* – путь до файла *openocd.exe*. Расположен в *<папка_openocd>/bin/*. В Linux просто указать *openocd*.
- *Корень каталога скриптов* – в Linux указать путь *<папка_openocd>/tcl*. В Windows указать просто корневую папку *openocd*.
- *Файл конфигурации* – путь до файла *simple_connect.cfg*. Расположен в *<папка_openocd>/tcl/kits/k1921vk01t_stlink/*.

2.3 Создание целевого устройства

В окне *Параметры* Qt Creator переходим на вкладку *Устройства.*

Эта вкладка позволяет создавать профили целевых устройств, т.е. тех устройств, под которые будет вестись компиляция и отладка.

Нажимаем *Добавить* → *Голое устройство* → *Запустить мастера*. Вводим название устройства и выбираем созданный ранее GDB сервер.

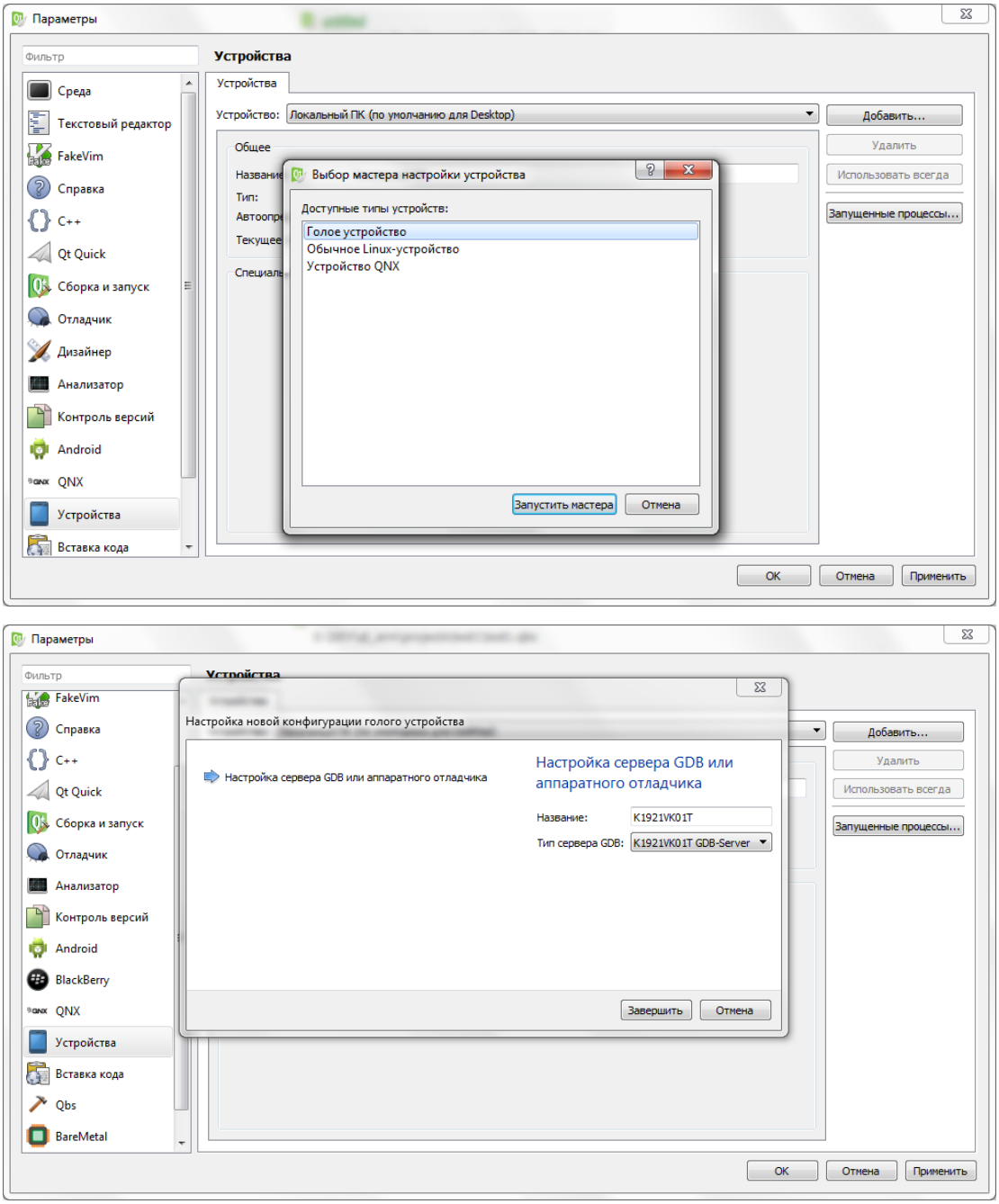

Рисунок 2.3

2.4 Настройка комплекта

Комплектом в Qt Creator называется набор из компилятора, отладчика и целевого устройства.

Настроим компилятор. Для этого в окне *Параметры* Qt Creator переходим на вкладку *Сборка и запуск* → *Компиляторы*. Нажимаем *Добавить* → *GCC* и вводим параметры компилятора аналогично представленным на рисунке 2.4.

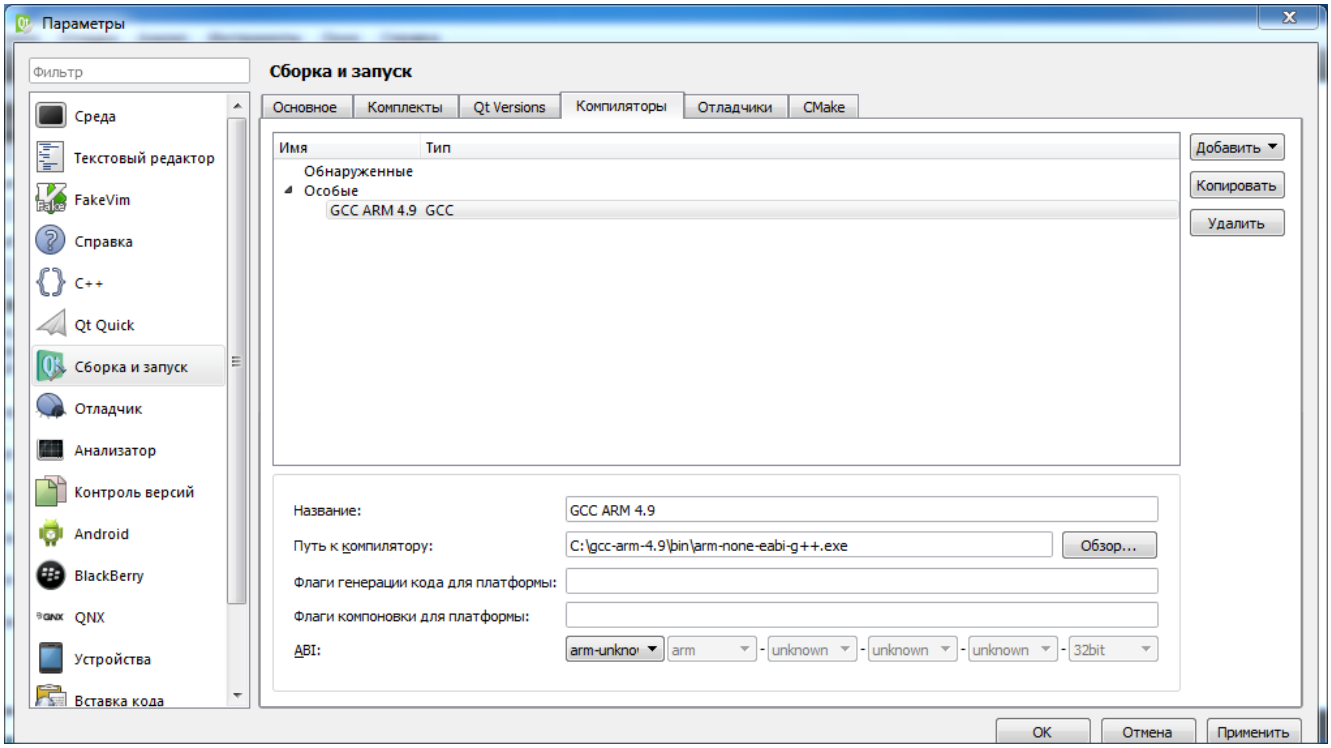

Рисунок 2.4

Далее перейдем к настройке отладчика. Для этого переходим на вкладку *Отладчики*. Нажимаем *Добавить* и вводим соответствующие параметры (рисунок 2.5).

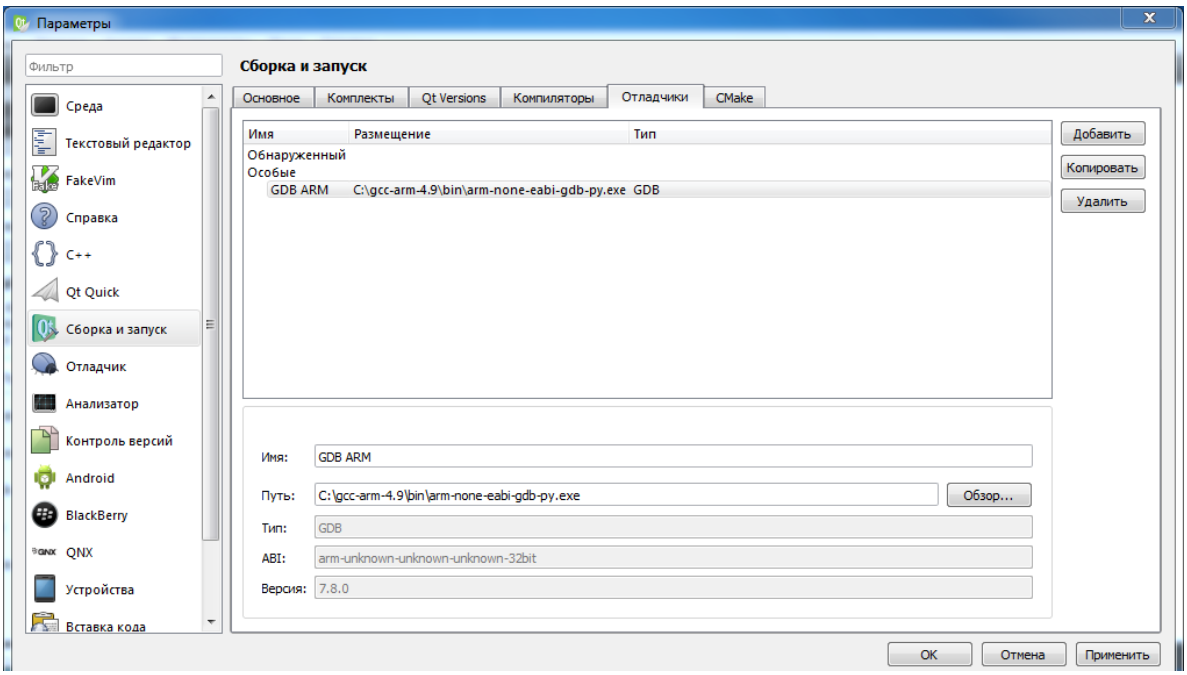

Рисунок 2.5

После того как отладчик с компилятором настроены можно собирать комплект. Обратите внимание на обведенную область – с помощью данной кнопки можно выбрать значок комплекта. Логотип НИИЭТ можно найти в корне директории ркуоводства. Обязательно установите его, это чрезвычайно важно!

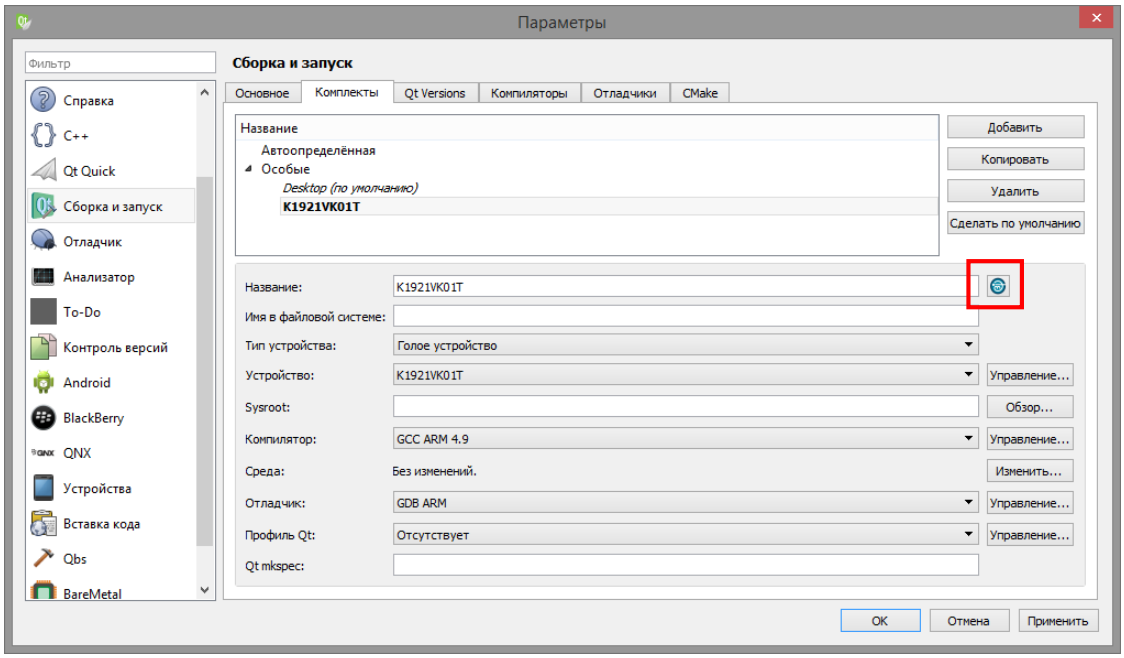

Рисунок 2.6

После настройки комплекта можно сделать еще несколько необязательных настроек, которые сделают дальнейшую работу несколько более комфортной.

Перейдем на вкладку *Основное*. Переопределим каталог проектов на другую, более удобную директорию, а также изменим каталог сборки по умолчанию (по умолчанию сборка ведется в директории, находящейся уровнем выше директории проекта). Имя каталога для сборки можно изменить также по желанию.

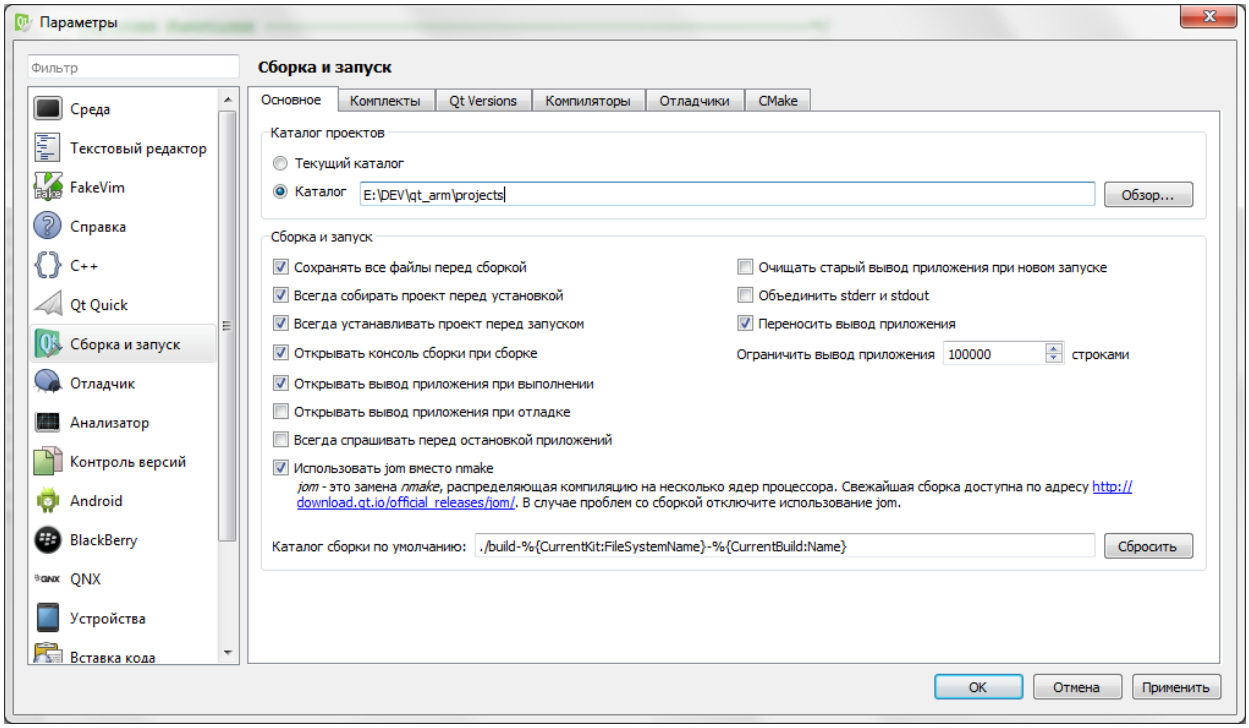

Рисунок 2.7

3 Создание проектов

3.1 Шаблоны создания проектов

Для ведения проектов Qt Designer позволяет использовать различные системы сборки: cmake, qmake, QBS. А само создание проекта в самом общем случае выглядит так: создается папка, в которой по соответствующим путям располагают исходные коды, затем формируется управляющий файл или файлы для системы сборки, в котором указываются все компилируемые файлы, ключи компилятора и компоновщика, макроопределения, библиотеки.

Для того чтобы упростить этот процесс, Qt позволяет конструировать шаблоны создания проектов. Пример шаблона проекта под систему сборки QBS для микроконтроллера К1921ВК01Т можно найти в папке руководства *wizards\K1921VK01T_with_qbs.*

Установить шаблон можно путем копирования папки *K1921VK01T_with_qbs* в директорию *<папка Qt_Creator>\ share\qtcreator\templates\wizards*.

3.2 Тестовый проект

Создадим тестовый проект, используя установленный шаблон. Переходим в *Файл → Создать файл или проект.* Выбираем соответствующий шаблон, указываем путь размещения проекта и, по желанию, добавляем его под управление системы контроля версий (если таковые установлены в системе).

После создания в окне редактора должен открыться файл *main.c*, в котором реализована простая программа, переключающая пин H[7] по прерыванию системного таймера SysTick.

Сама структура проекта определяется файлом *<имя_проекта>.qbs*, который можно легко отредактировать и оптимизировать под определенные нужды. Справочная документация о системе сборки QBS может быть найдена на официальном сайте.

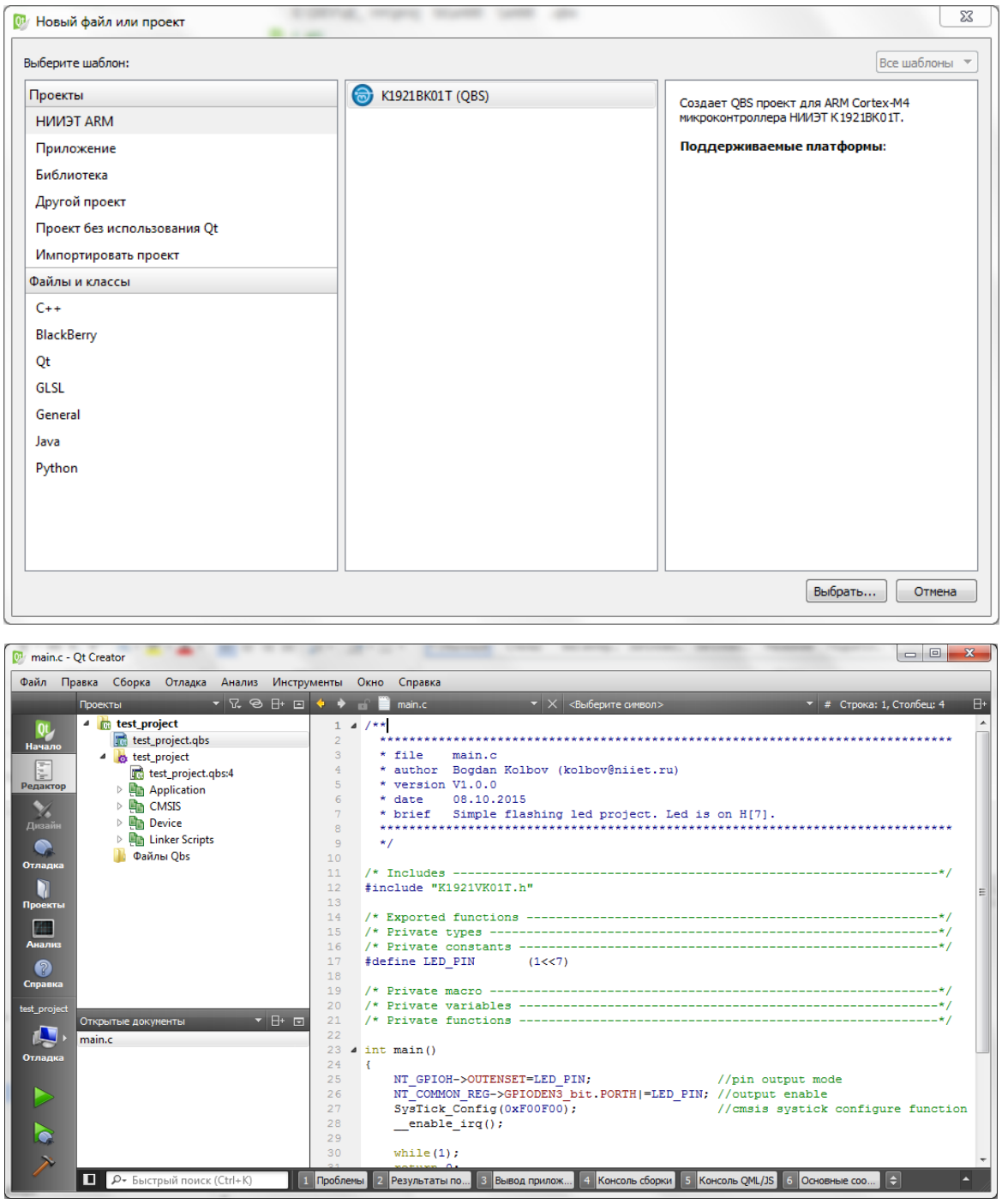

Рисунок 3.1

В данном проекте, помимо файла с основным кодом и заголовочного файла с описанием регистров периферии, используется стандартная библиотека ядра CMSIS, а также ассемблерный файл, исполняемый при старте микроконтроллера, и скрипт компоновщика, в котором определено адресное пространство данного микроконтроллера. Примеры составления последних двух файлов могут быть найдены в каталоге *<папка_gcc_arm>\share\gcc-arm-none-eabi\samples*.

3.3 Сборка и отладка

Перед тем как собрать тестовый проект, необходимо произвести его настройку. Для этого из режима *Редактор* переходим в режим *Проекты*, кликнув по значку на панели выбора режимов в левой части экрана.

Далее необходимо добавить созданный ранее комплект *K1921VK01T* в качестве основного. Нажимаем *Добавить* → *K1921VK01T*, а комплект *Desktop* удаляем.

Таким образом, окно проектов должно выглядеть также как на рисунке 3.2.

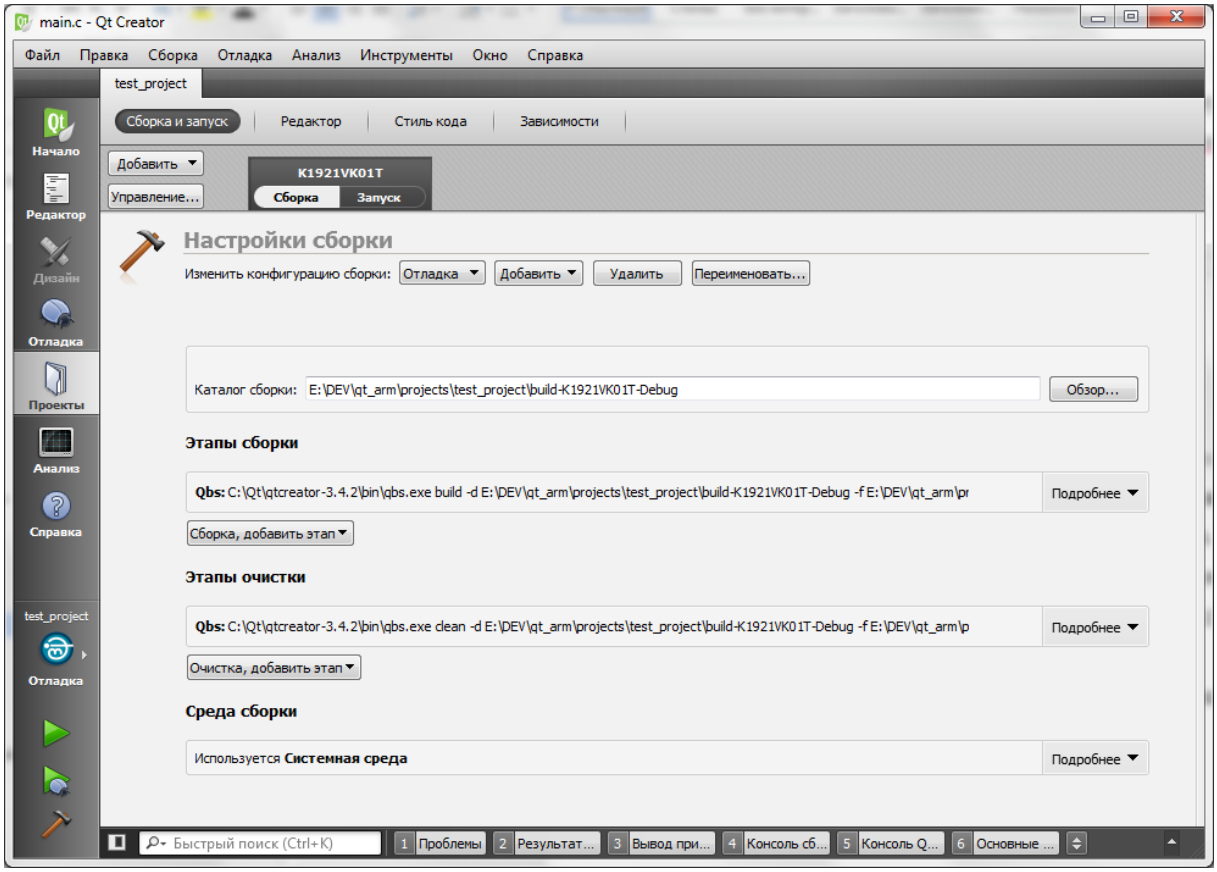

Рисунок 3.2

Теперь возможно осуществление сборки проекта. Для этого нажимаем *Ctrl+B* или используем кнопку в левой нижней части экрана. Проект должен сразу же без каких-либо ошибок или предупреждений собраться.

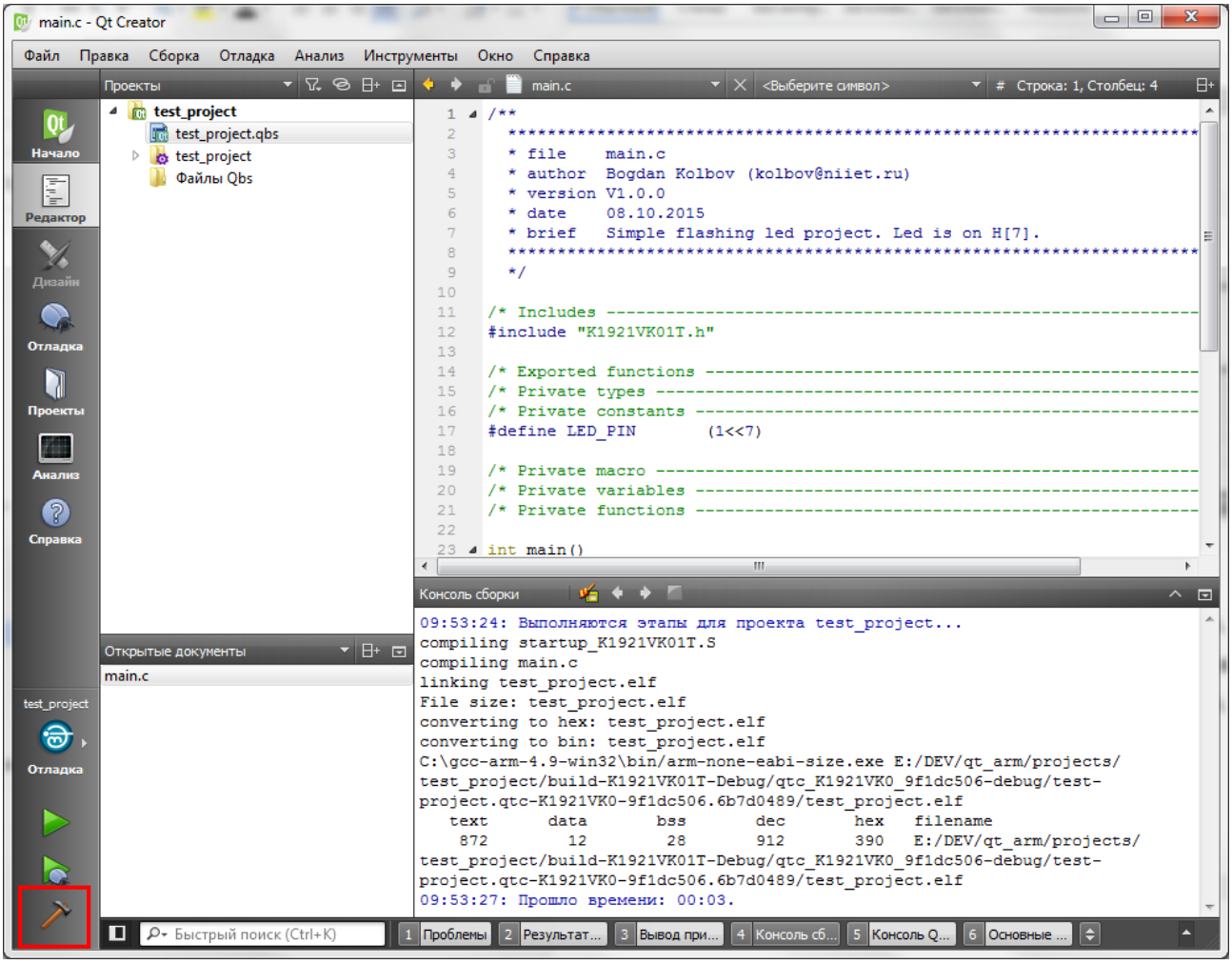

Рисунок 3.3

Перед тем как начать отладку убедитесь, что драйвера на аппаратный отладчик установлены, и контроллер к нему подключен корректно. Один из способов это сделать – запустить из папки OpenOCD скрипт *k1921vk01t_stlink_simple_connect.bat*. Или в Linux выполнить:

```
openocd -s <путь_до_openocd>/tcl/ -f
kits/k1921vk01t_stlink/simple_connect.cfg
```

В открывшейся консоли должна быть показана информация о том, что утилита обнаружила отладчик и смогла определить подключенный к нему контроллер. Затем консольное окно можно закрыть.

Для того чтобы начать отладку в Qt Creator достаточно нажать *F5* или воспользоваться иконкой в левой нижней части экрана. После успешного

подключения отладчика, программа будет залита во флэш-память устройства и запущена.

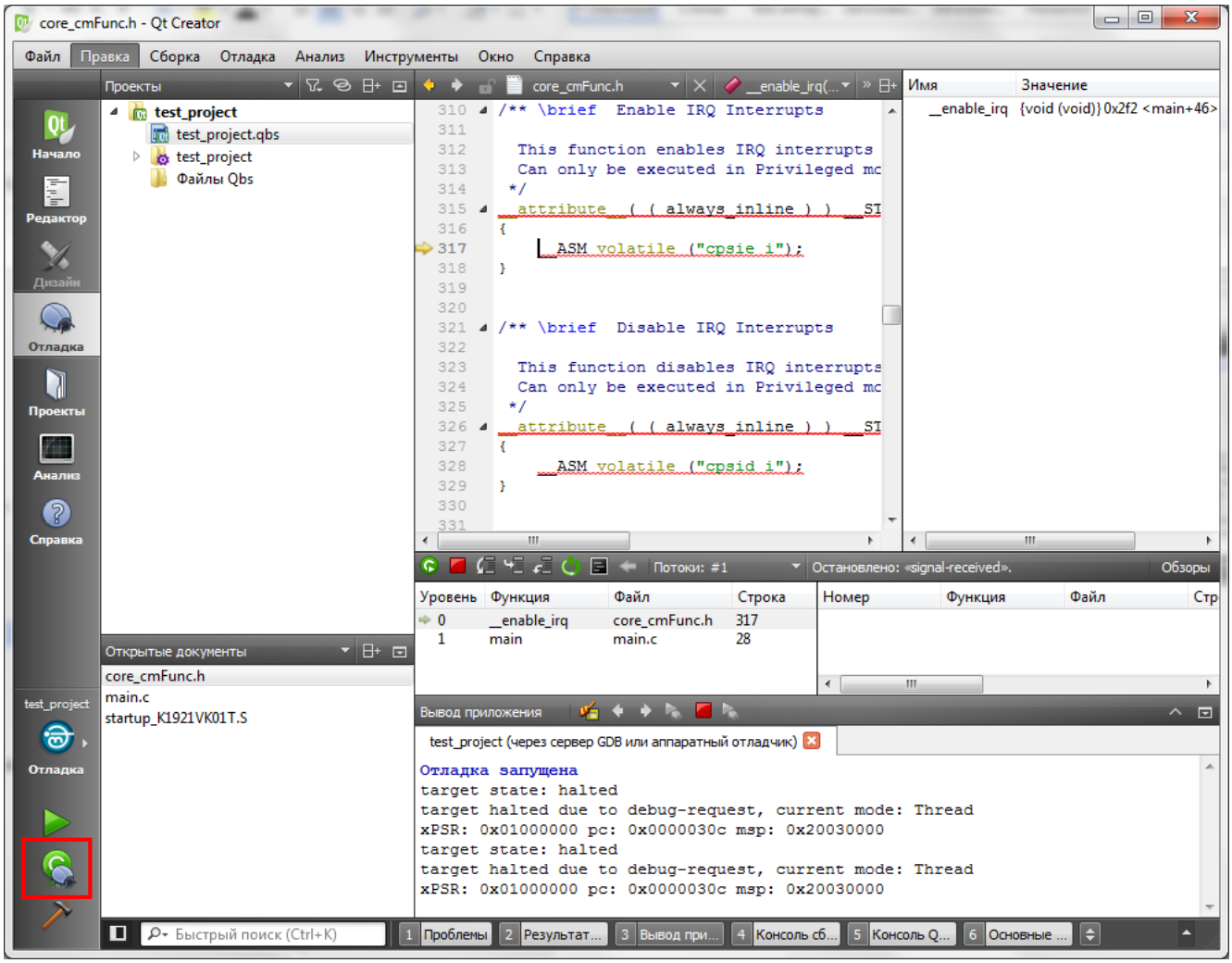

Рисунок 3.4

4 Краткая справка по OpenOCD

4.1 Общие сведения

Название OpenOCD произошло от сокращения Open On-Chip Debugger. OpenOCD предоставляет инструментарий отладки, внутрисхемного программирования, внутрисхемного тестирования для встраиваемых систем (микроконтроллеров, FPGA и т. п.). OpenOCD предоставляет доступ к адаптеру для отладки - аппаратному модулю, который помогает получить требуемые сигналы для отладки целевого устройства (обычно с одной стороны адаптер подключается к ПК через USB, а с другой стороны имеет интерфейс JTAG, через который подключено отлаживаемое устройство).

В следующих подразделах будет показано, как можно использовать OpenOCD, чтобы программировать, стирать и совершать некоторые специальные действия по отношению к микроконтроллеру К1921ВК01Т.

Представленная ниже информация является лишь свободной трактовкой некоторых функций утилиты с точки зрения соединения с упомянутым ранее микроконтроллером. Очевидно, что за дополнениями и пояснениями следует обращаться в первую очередь к официальной документации.

4.2 Программирование

В каталоге с установленным OpenOCD можно найти скрипт *k1921vk01t_stlink_program.bat* с помощью которого можно осуществлять программирование К1921ВК01Т. Содержимое скрипта:

```
"bin\openocd" -f kits/k1921vk01t_stlink/simple_connect.cfg -c "program filename.elf verify reset exit"
```

Как видно из исходного кода, запускается исполняемый файл *openocd*, которому в качестве параметров передаются: скрипт соединения с микроконтроллером К1921ВК01Т через StLink, и команда запрограммировать

флэш из файла с именем *filename*. Кроме *.elf* возможно использование *.hex, .bin* файлов. Подкоманды, идущие за именем файла являются необязательными, и, как не сложно догадаться, включают верификацию прошивки после записи, последующий сброс контроллера и выход из программы.

4.3 Стирание

Для того чтобы осуществлять стирание микроконтроллера можно использовать скрипт *k1921vk01t_stlink_erase.bat.* Его содержимое:

```
"bin\openocd" -f kits/k1921vk01t_stlink/simple_connect.cfg -c "flash erase_sector 0 0 last"
```

Принцип его действия аналогичен скрипту для программирования, с одним лишь отличием что передается команда стирания сектора со следующими параметрами: использовать 0-ой банк памяти (основная флэш), и стирать с 0-ой по последнюю страницу памяти включительно.

4.4 Дополнительные функции драйвера

Для работы OpenOCD с контроллером К1921ВК01Т был написан драйвер *niietcm4,* который помимо стандартного интерфейса для записи, чтения и стирания внутренней флэш, реализует специальные команды, характерные для этого контроллера. Список этих команд представлен ниже. Использовать их можно двумя путями: передавать после ключа *–c* при вызове openocd (подразделы 4.2 и 4.3) или добавлять их в конец вызываемого *.cfg* скрипта.

Во всех специальных командах фигурирует параметр *bank_id*, он характеризует номер используемого банка памяти. В данном случае, под банком памяти понимается вся внутренняя флэш-память, а следовательно, банк существует лишь один и его номер равняется нулю.

Все цифровые значения вводятся в десятичной системе счисления, если не указано иного.

4.4.1 Чтение байта пользовательской флэш

```
niietcm4 uflash_read_byte bank_id ('main'|'info') address
```

Команда выполняет чтение байта по адресу равному *address* из основной (*main*) или информационной (*info*) области пользовательской флэш.

4.4.2 Запись байта пользовательской флэш

```
niietcm4 uflash_write_byte bank_id ('main'|'info') address
value
```

Команда выполняет запись байта равному *value* по адресу *address* в основной (*main*) или информационной (*info*) области пользовательской флэш. Для того чтобы оставлять остальные данные неизменными, запись производится по алгоритму *дамп страницы* → *модификация дампа* → *стирание страницы* → *запись дампа*. Из-за применения подобного подхода команда записи может выполнятся значительное время.

4.4.3 Полное стирание пользовательской флэш

```
niietcm4 uflash_full_erase bank_id
```

Команда выполняет полное стирание пользовательской флэш (основная и информационная области).

4.4.4 Постраничное стирание пользовательской флэш

```
niietcm4 uflash_erase bank_id ('main'|'info') first_sector_num
last_sector_num
```

Команда выполняет постраничное стирание основного (*main*) или информационного (*info*) блока пользовательской флэш начиная со страницы с номером *first_sector_num* по страницу с номером *last_sector_num* включительно.

4.4.5 Проверка защиты пользовательской флэш

```
niietcm4 uflash_protect_check bank_id ('main'|'info')
```

Команда выполняет проверку защиты всех секторов основной (main) или информационной (info) области пользовательской флэш и выводит результаты на экран.

4.4.6 Защита пользовательской флэш

```
niietcm4 uflash_protect bank_id ('main'|'info')
first_sector_num last_sector_num ('on'|'off')
```

Команда выполняет включение (*on*) или снятие (*off*) защиты секторов основной (*main*) или информационной (*info*) области пользовательской флэш. Модификация производится с секторами с номерами начиная от *first_sector_num* по *last_sector_num* включительно.

4.4.7 Проброс информационной области основной флэш

```
niietcm4 bflash_info_remap bank_id ('on'|'off')
```

Команда выполняет включение (*on*) или отключение (*off*) подмены первой страницы основной флэш её скрытой информационной областью.

4.4.8 Конфигурация режима загрузки с внешней памяти

```
niietcm4 extmem_cfg bank_id
('gpioa'|'gpiob'|'gpioc'|'gpiod'|'gpioe'|'gpiof'|'gpiog'|'gpioh
') pin_num ('func1'|'func3')
```

Команда выполняет конфигурацию интерфейса внешней памяти для режима загрузки из внешней памяти. Выбирается порт и номер управляющего пина (“низкий” уровень на нем – загрузка из внутренней флэш, “высокий” – из внешней памяти), а также альтернативная функция пинов самого интерфейса.

4.4.9 Включение возможности загрузки из внешней памяти

```
niietcm4 extmem_boot bank_id ('on'|'off')
```

Команда выполняет включение (*on*) или отключение (*off*) возможности загрузки из внешней памяти.

4.4.10 Сервисный сброс

```
niietcm4 service_mode_erase bank_id
```

Команда выполняет полное стирание основной и пользовательской флэш (информационный области в том числе). Команду можно выполнить только в сервисном режиме. Для того чтобы зайти в сервисный режим, необходимо удерживать высокий уровень на пине H[2] во время сброса.

Контакты

Все вопросы, замечания и предложения, касаемо данного руководства в частности, а также использования микроконтроллеров ОАО “НИИЭТ” Cortex-M4 с открытым ПО в целом, просьба направлять на электронный адрес kolbov@niiet.ru.