

# Контроллеры ARMatura

© Dmitry Ponyatov <dponyatov@gmail.com>, SSAU ASCL

3 марта 2013 г.

## Оглавление

<b>Оглавление</b>	<b>1</b>
<b>I Введение</b>	<b>6</b>
<b>II Железо</b>	<b>8</b>
1 STM32VLDISCOVERY /STM32F100RBT6/	9
2 STM32F4DISCOVERY /SRM32F407VGT6/	11
3 ARMatura /STM32F407IGT/	13
4 PION /STM32F103C8/	14
<b>III Первые шаги</b>	<b>15</b>
5 Установка ПО	16
6 Первый проект: blink	17
6.1 Структура файлов . . . . .	22
6.2 blink.c . . . . .	23
7 Hell Of World	24
<b>IV Средства разработки</b>	<b>25</b>
8 IDE	26

8.1 Keil . . . . .	26
8.2 Настройки среды . . . . .	32
8.3 Настройки проекта . . . . .	33
8.4 Eclipse . . . . .	46
8.5 Code::Blocks . . . . .	47
8.6 gVim . . . . .	47
8.7 IAR . . . . .	47
<b>9 Компиляторы</b>	<b>48</b>
9.1 GCC . . . . .	48
9.2 KeilCC . . . . .	48
9.3 IAR . . . . .	48
<b>10 Адаптеры</b>	<b>49</b>
10.1 STlink . . . . .	49
10.2 JTAG . . . . .	56
<b>V Отладка</b>	<b>57</b>
<b>11 Отладка в Keil</b>	<b>58</b>
<b>12 STM32 SWD</b>	<b>59</b>
<b>13 GDB</b>	<b>60</b>
13.1 STlink gdbserver . . . . .	60
<b>14 OpenOCD</b>	<b>61</b>
<b>15 JTAG</b>	<b>62</b>
<b>VI Основы языка C<sup>+</sup></b>	<b>63</b>
<b>16 Синтаксис</b>	<b>64</b>
<b>17 Типы данных</b>	<b>65</b>
<b>18 Стандартная библиотека libc</b>	<b>66</b>
<b>VICMSIS</b>	<b>67</b>
<b>19 Startup: файлы стартового кода</b>	<b>69</b>
<b>20 Стандартная библиотека STM32</b>	<b>70</b>
20.1 stm32f4xx.h . . . . .	70
<b>21 USB client/host</b>	<b>71</b>

<i>Оглавление</i>	3
<b>VII Ядро Cortex-Mx</b>	<b>72</b>
<b>22 Режимы ARM и Thumb</b>	<b>73</b>
<b>23 DMA</b>	<b>74</b>
<b>24 DSP /Cortex-M3/</b>	<b>75</b>
<b>25 FPU /Cortex-M4F/</b>	<b>76</b>
<b>IX Интерфейсы</b>	<b>77</b>
<b>26 USB</b>	<b>78</b>
<b>27 UART</b>	<b>79</b>
<b>28 SPI</b>	<b>80</b>
<b>29 I2C</b>	<b>81</b>
<b>30 CAN</b>	<b>82</b>
<b>X Операционные системы OCPB</b>	<b>83</b>
<b>31 Keil RTX</b>	<b>84</b>
<b>32 FreeRTOS</b>	<b>85</b>
<b>33 eCos</b>	<b>86</b>
<b>34 Linux</b>	<b>87</b>
<b>XI Стек TCP/IP</b>	<b>88</b>
<b>35 Ethernet</b>	<b>89</b>
<b>36 PPP</b>	<b>90</b>
<b>XII Типовые применения</b>	<b>91</b>
<b>37 GPS</b>	<b>92</b>
37.1 Протокол NMEA 0183 . . . . .	92
37.2 \$GPRMC — рекомендуемый минимум навигационных данных . . . . .	93
37.3 Системы координат (датум) . . . . .	94
37.4 Методы преобразования систем координат . . . . .	95
37.5 Геоид и эллипсоиды . . . . .	96
37.6 Tistar15 . . . . .	97
37.7 WISMO228 . . . . .	97
<b>38 GSM</b>	<b>98</b>

38.1 WISMO228 . . . . .	98
<b>39 шина Dallas 1Wire</b>	<b>99</b>
39.1 RTC . . . . .	99
39.2 Датчики температуры DS18x20 . . . . .	99
<b>XIIИстраиваемый Linux</b>	<b>100</b>
<b>XIIIQA</b>	<b>101</b>
<b>XIVПриложения</b>	<b>103</b>
<b>40 Сводная таблица процессоров</b>	<b>104</b>

# Listings

6.1 blink.c ..... 23

# Часть I

## Введение

Эта книга – набор методичек по разработке ПО для встраиваемых систем, написанных для Института космического приборостроения СГАУ.

Для применения в реальных проектах научной аппаратуры была разработана линейка унифицированных модулей:

1. ARMatura — модуль на мощном микропроцессоре STM32F417IGT: 1M Flash, 192K SRAM, TQFP176, DSP, FPU,.. **40**

предназначен для использования в качестве центрального процессора цифровой системы: обработка данных, сложные алгоритмы управления, ЦОС, вычисления, реализация протоколов передачи данных по интерфейсам USB, Ethernet, RS232/UART, SPI, I2C, CAN,..

2. PION **4** — модуль на самом простом и дешевом STM32F100C4T6B: 128K Flash, 8K SRAM, UART, SPI **40**

периферийный модуль для стыковки с аналоговыми датчиками и исполнительными устройствами, предварительная ЦОС обработка, передача данных на ARMatura-модули для дальнейшей обработки данных.

также модуль применим в качестве самостоятельного простого интерфейса при замене на чип STM32F103 с портом USB или установки внешних интерфейсных микросхем FT232RL (USB Serial), CP1202, MC1551 (CAN).

3. BACKPLANE — коммутационная плата межмодульного интерфейса

4. POWER — модуль импульсного источника питания

5. STEPPER — модуль управления двухфазным шаговым двигателем

6. WISMO — несущая плата для GPS/GSM модуля WISMO 228

7. QVGA — несущая плата для TFT touch-панели

В качестве базового микроконтроллера были выбраны чипы семейства STM32Fxxx с ядрами Cortex-M3, Cortex-M4F (ARM) как самые дешевые, и имеющие хорошую поддержку в виде отладочных плат линейки Discovery.

В общем, линейка модулей ARMatura может рассматриваться в качестве замены устаревшей линейки периферийных контроллеров Arduino на базе МК AVR8.

Проект размещен в репозитории <https://github.com/ponyatov/ARMatura.git> и предоставляется на условиях OpenHardware licence (за исключением прошивок и схем по тематике ИКП СГАУ).

Контакты разработчиков:

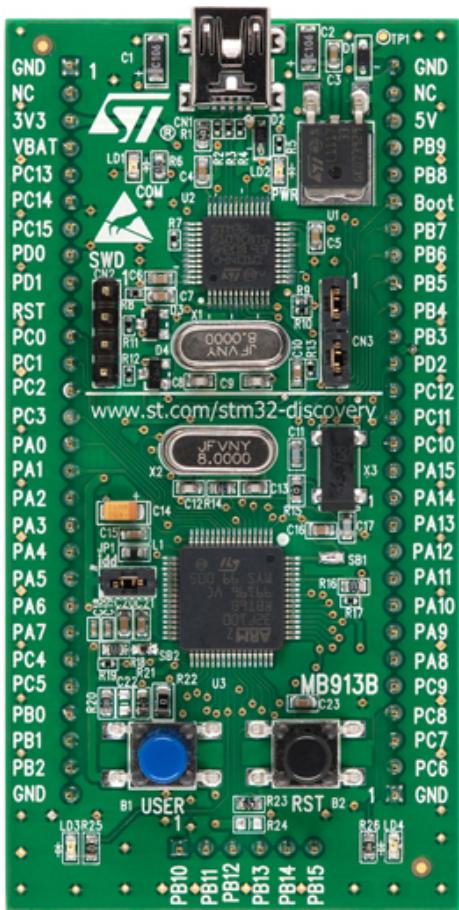
- ИКП СГАУ <[semkin@ssau.ru](mailto:semkin@ssau.ru)>
- Дмитрий Понятов <[dponyatov@gmail.com](mailto:dponyatov@gmail.com)>

## Часть II

### Железо

# Глава 1

## STM32VLDISCOVERY /STM32F100RBT6/



- Микроконтроллер STM32F100RBT6 **40**, 128 KB Flash, 8 KB RAM in 64-pin LQFP
- Встроенный ST-Link с возможностью использования в режиме внешнего программатора (только с SWD коннектором)
- Разработана для питания как от USB, так и от внешнего источника 3.3 или 5 вольт
- Может обеспечить питание 3 и 5 вольт для внешних устройств
- Два пользовательских светодиода (зеленый и синий)
- Пользовательская кнопка USER

- Кнопка сброса RESET
- Контактные гребенки для всех выводов QFP64 для быстрого подключения и монтажа прототипа

Цена в розницу: 750 р.

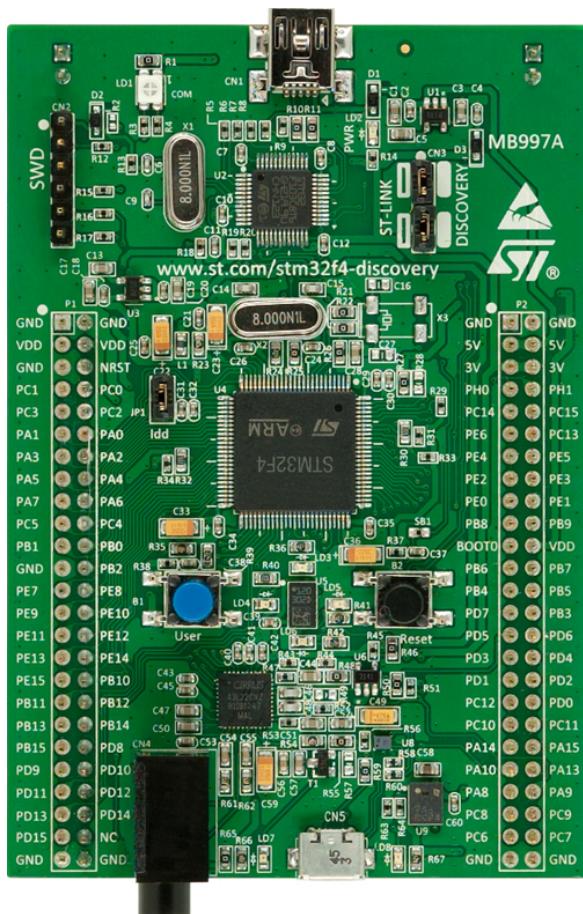
<http://www.voltmaster.ru/cgi-bin/qwery.pl?id=127000573571&group=7000046>

<http://we.easyelectronics.ru/STM32/stm32vldiscovery-the-grabli-ili-dlya-samyh-nach.html>

- Контроллер STM32F100RBT6 (128KB Flash, 8KB RAM, up to 24 MHz)
- Кварц 8MHz, установлен на цангах, так что можно ставить свой
- Кварц 32.768KHz **X3**
- Встроенный программатор-отладчик STlink<sup>10.1</sup>, только с выходом SWD, можно его использовать и для прошивки своих девайсов. Что характерно, отладчик выполнен на более серьёзном STM32F103CBT6<sup>40</sup>.
- Две кнопки (правда одна из них — RESET).
- Два светодиода **PC8**,**PC9**

## Глава 2

# STM32F4DISCOVERY /SRM32F407VGT6/



- микроконтроллер STM32F407VGT6 **40** на базе 32-битного ядра Cortex-M4F, 1 MB Flash, 192 KB RAM в корпусе LQFP100
- встроенный ST-LINK/V2 с возможностью использования в режиме внешнего программатора (только с SWD коннектором)
- Разработана для питания как от USB, так и от внешнего источника 5 вольт
- Может обеспечить питание 3 и 5 вольт для внешних устройств
- LIS302DL, ST MEMS датчик движения, 3-осевой цифровой гироскоп
- MP45DT02, ST MEMS аудиодатчик, всенаправленный цифровой микрофон

- CS43L22, аудиоЖАП со встроенным аудиоусилителем класса D
- Восемь LEDs: LD1 (red/green) for USB communicationLD2 (red) for 3.3 V power onFour user LEDs, LD3 (orange), LD4 (green), LD5 (red) and LD6 (blue)2 USB OTG LEDs LD7 (green) VBus and LD8 (red) over-current
- Пользовательская кнопка USER
- Кнопка сброса RESET
- USB OTG FS with micro-AB connector
- Контактные гребенки для всех выводов LQFP100 ля быстрого подключения и мон-тажа прототипа

Цена в розницу: 1140 р.

<http://www.voltmaster.ru/cgi-bin/qwery.pl?id=127000854271&group=7000046>

## Глава 3

### ARMatura /STM32F407IGT/

## Глава 4

### PION /STM32F103C8/

Модуль PION предназначен для мелких задач управления, первичной обработки данных, стыковки с устройствами измерения и исполнительными устройствами, т.е. для тех задач, для которых ранее использовались микроконтроллеры Atmel AVR8.

процессор	STM32F103C8	??
тактовая	72Mhz	
Flash	64K	
SRAM	20K	
шина	ARMAbus	
интерфейсы	UART	3
	USB	1
	CAN	1
	SPI	2
	АЦП	10x12b
	ЦАП	2x12b
буфер	Parallel Flash	64K

Выбор процессора определялся ценой, наличием CAN интерфейса (требование для шины ARMAbus), USB (возможность использования в виде изолированного контроллера для ПК), объемом SRAM для хранения рабочих данных, и количеством ног ввода/вывода.

# **Часть III**

## **Первые шаги**

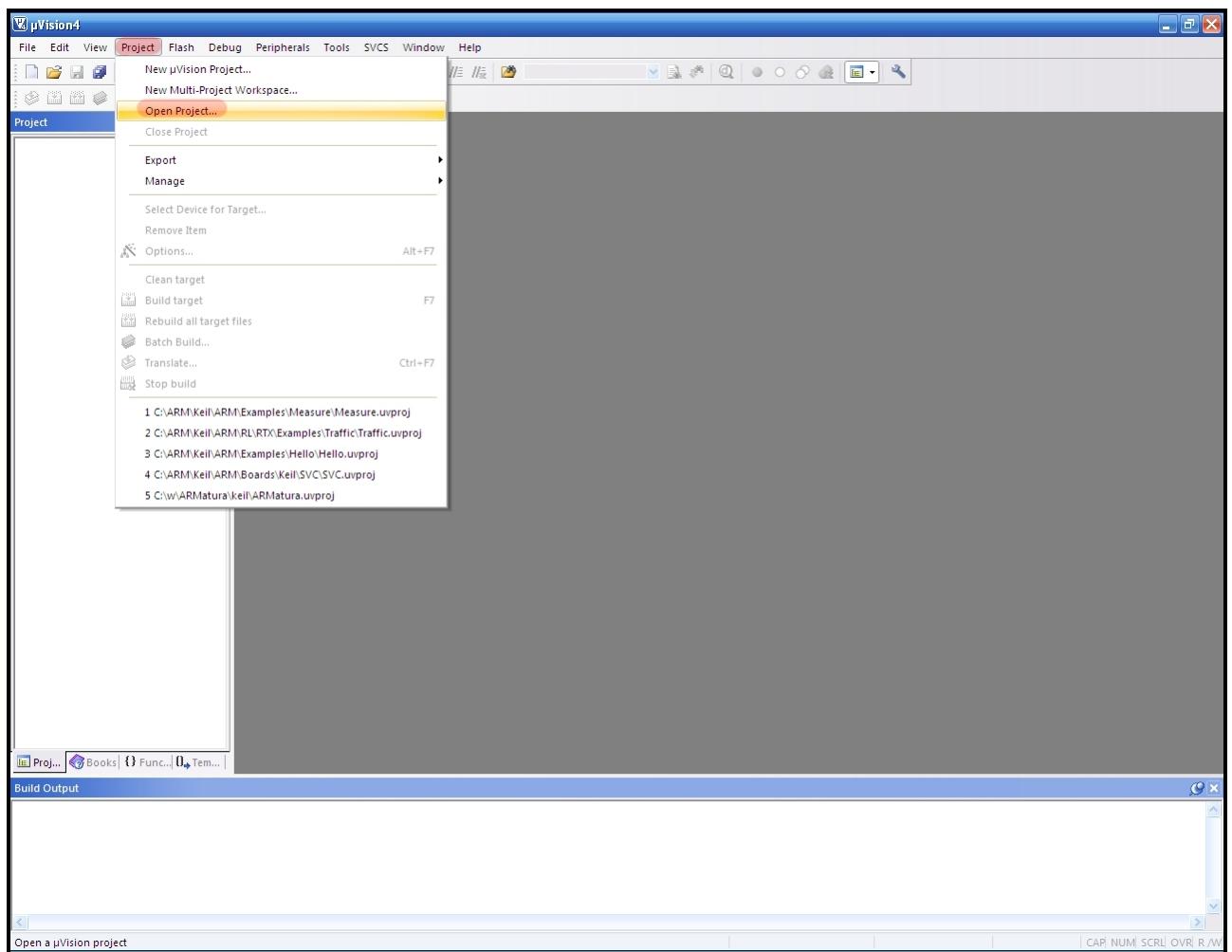
# Глава 5

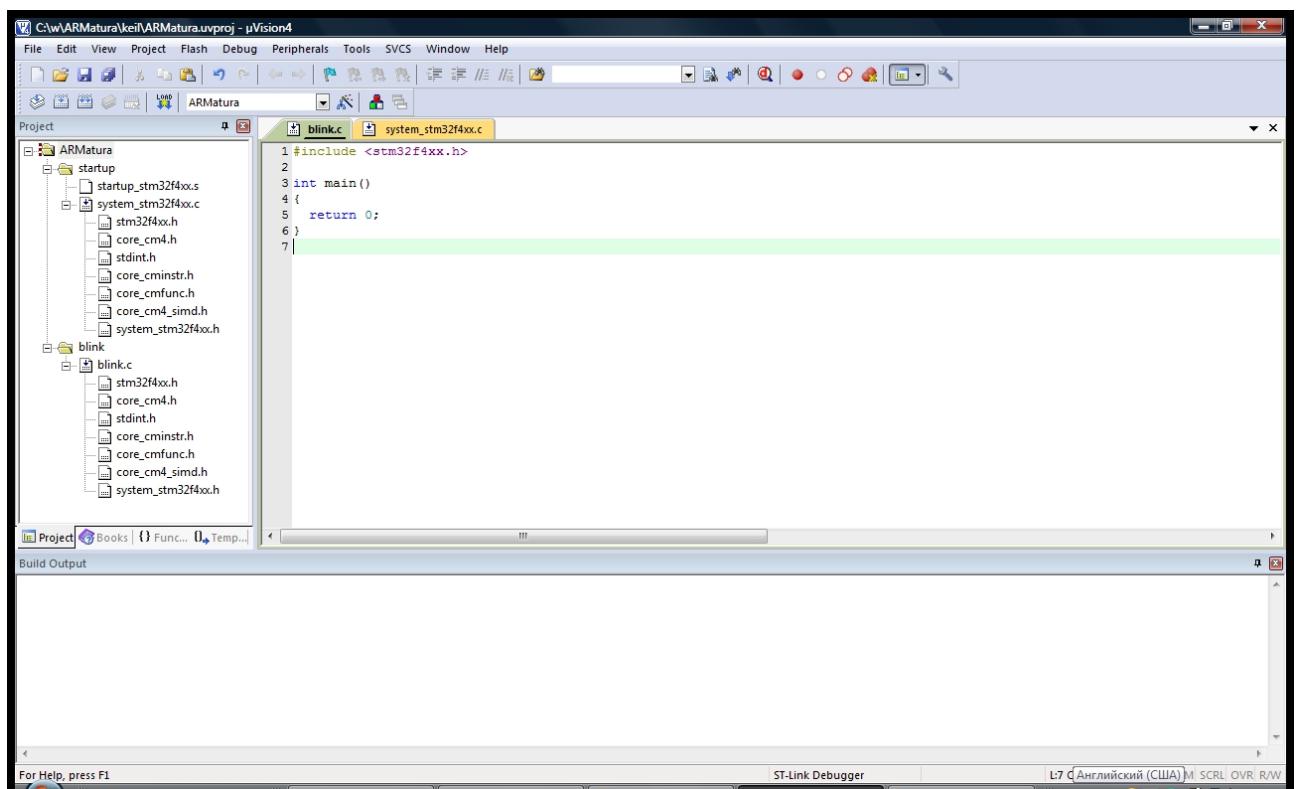
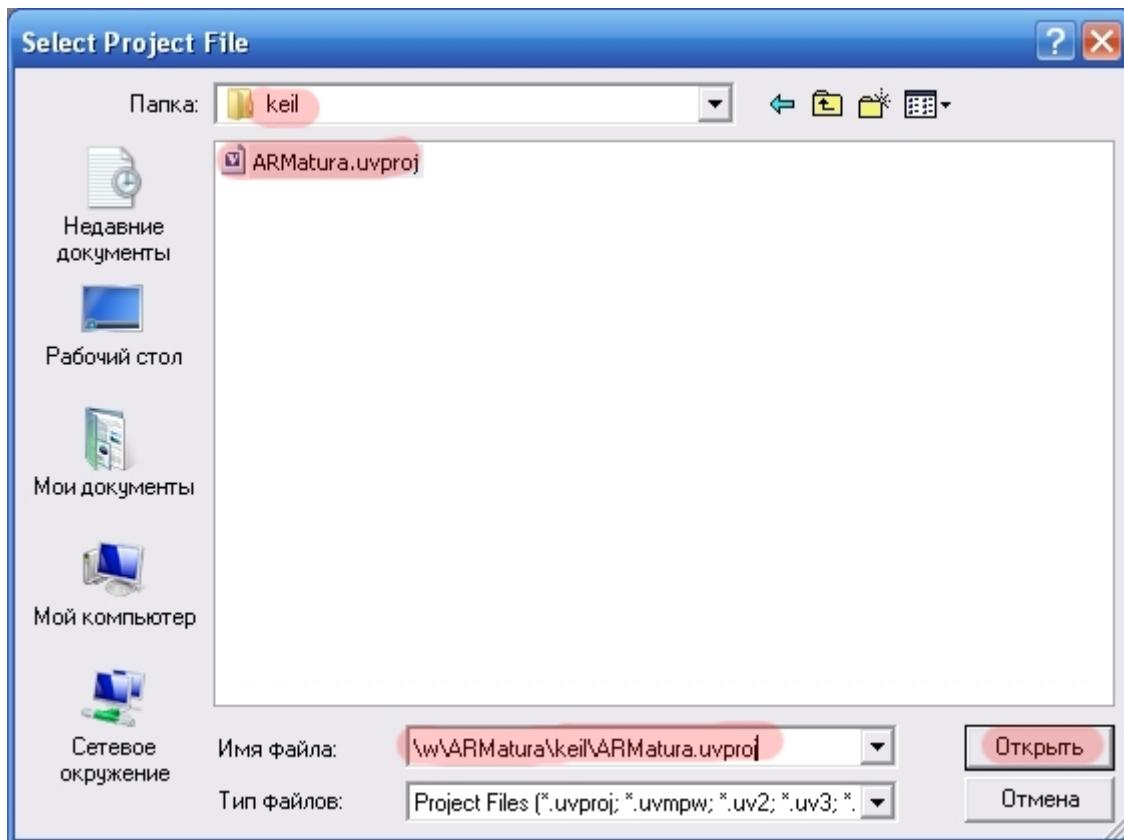
## Установка ПО

1. установка пакета ПО STlink [10.1](#)
2. установка Keil MDK-ARM [8.1](#)

# Глава 6

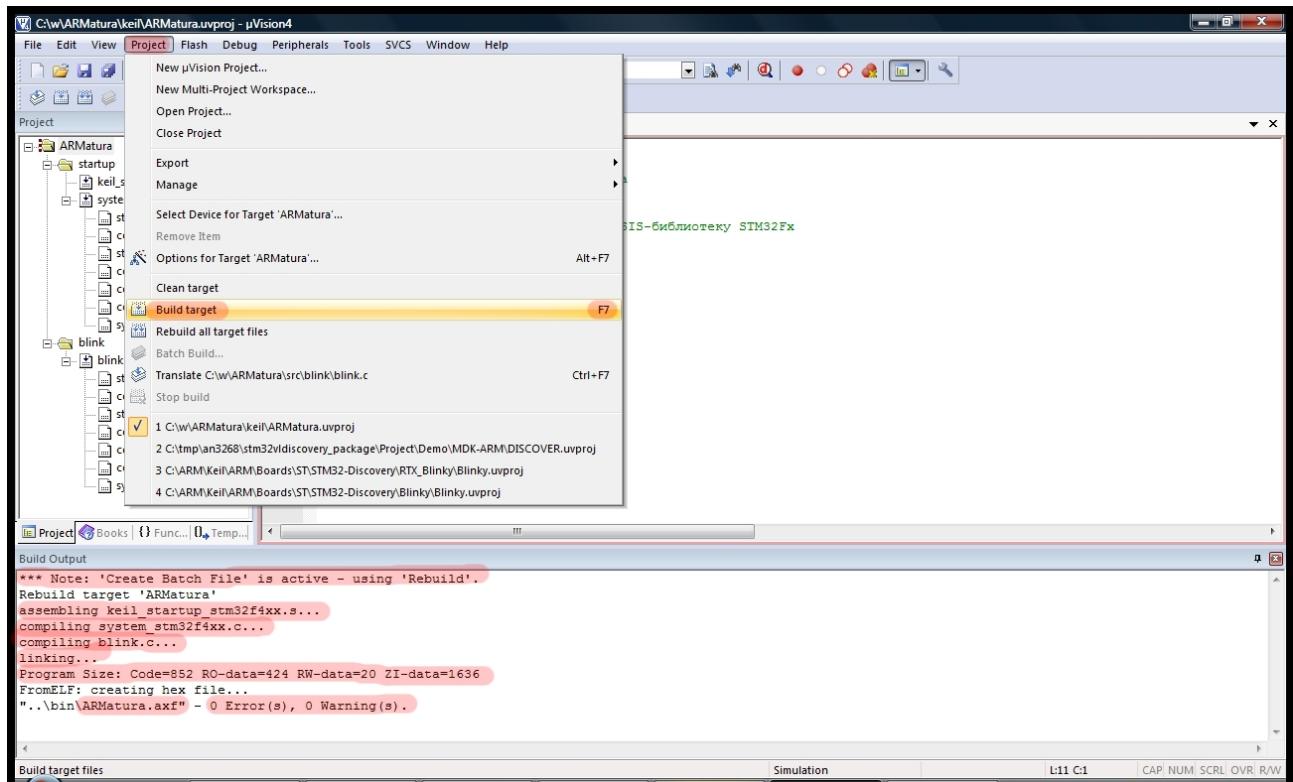
## Первый проект: blink



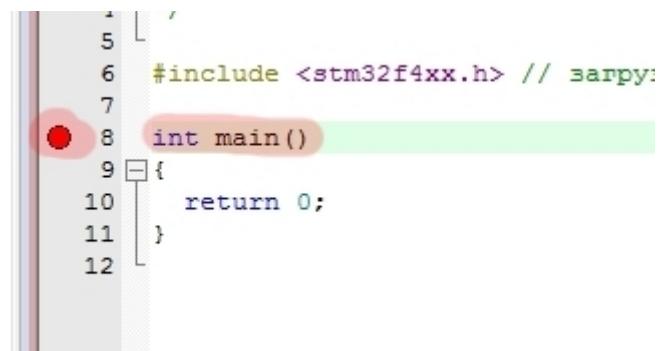


При необходимости можно изменить настройки проекта см. 8.3. Перенастройка может понадобиться если вы пытаетесь работать не с STM32F4DISCOVERY<sup>2</sup>, а с другой платой или самопальной железкой. Также перенастройка может понадобится если почему-то не срабатывает подключение к адаптеру STlink<sup>10.1</sup>.

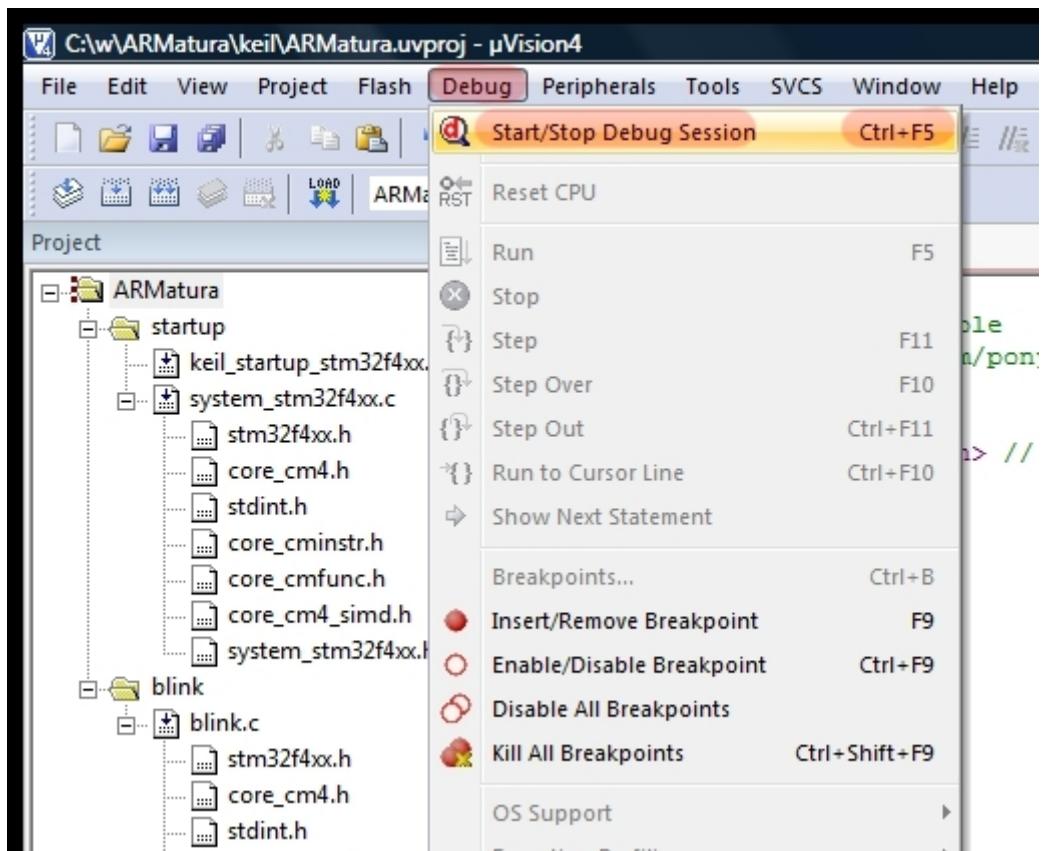
Собираем проект, нажав клавишу **F7**:



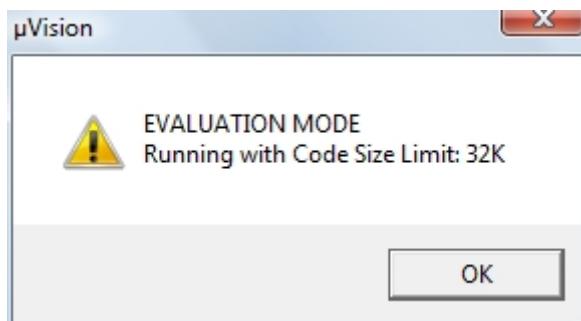
Проект успешно собрался, теперь можно проставить на `main()` точку останова (breakpoint) переместив курсор на заголовок функции и нажав кнопку **F9**. При работе программы останавливается на точках останова, позволяя посмотреть состояние регистров процессора, переменных и т.п.



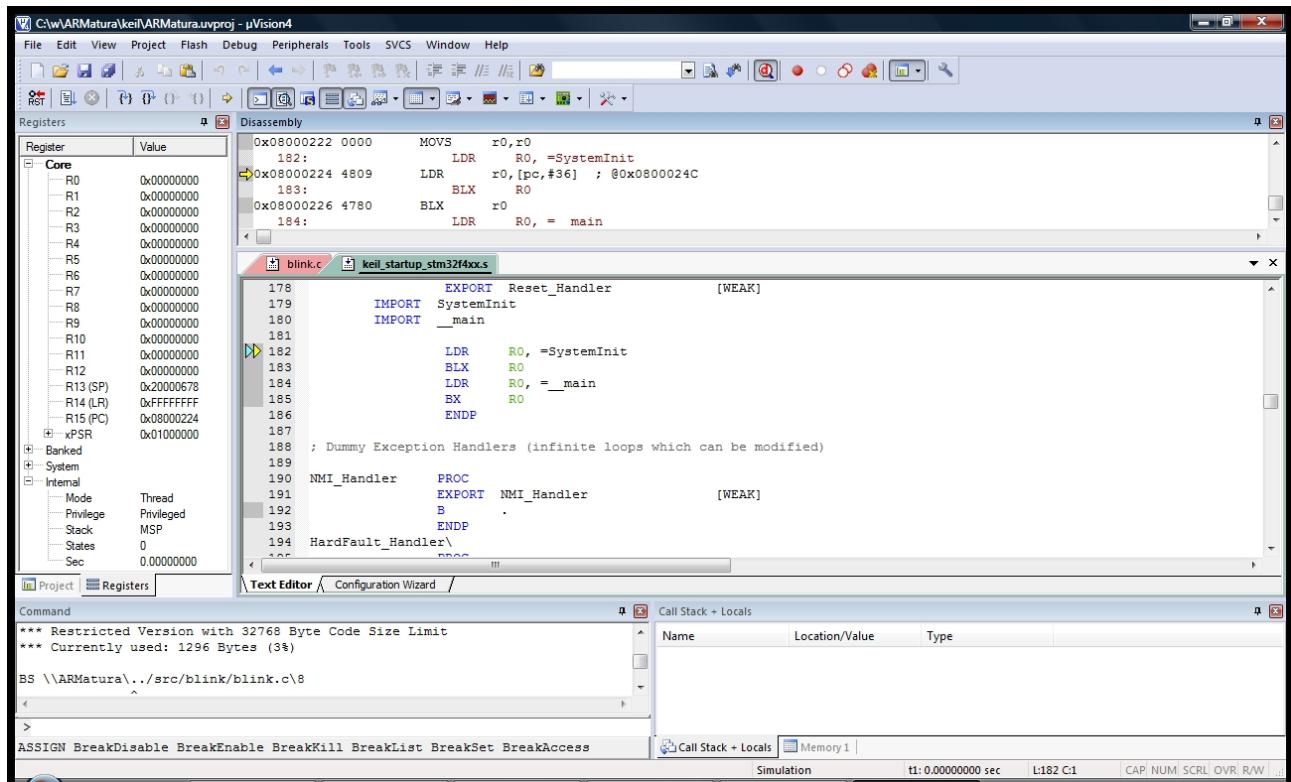
Затем запускаем отладку нажав **Ctrl + F5**



Вывалится диалог с сообщением об использовании оценочной версии Keil 8.1.



После дрыганья окнами откроется отладочный режим:

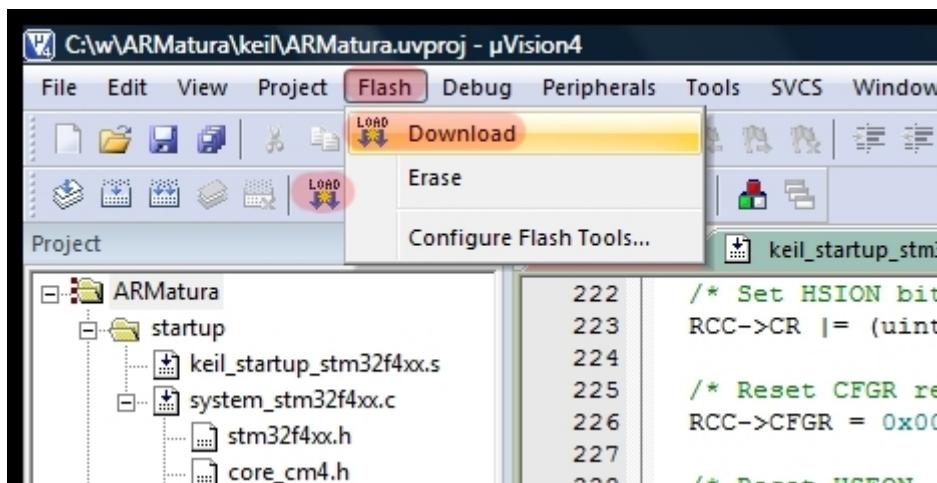


В окне редактора автоматически откроется код `keil_startup_stm32f4xx.s`

В окне Register показывается состояние регистров процессора. В верхней части расположено окно дизассемблированного кода, снизу слева командная консоль отладчика, снизу справа — просмотр стека и памяти.

Подробно работа с отладчиком описана в разделе 11, при желании вы можете пройтись в пошаговом режиме (нажимая кнопку F11) по коду инициализации, и глядя в даташит на ваш микроконтроллер, и попытаться разобраться что именно делает код инициализации 19.

Пока же разборки с работой в отладчике можно отложить, поэтому останавливаем отладчик нажатием на Ctrl-F5, подключаем отладочную плату, если она оказалась отключенной, и нажимаем Flash > Download или иконку в тулбаре:



Запускается прошивка флешпамяти на отладочной плате, на программаторе мигает светодиод передачи данных, и после успешной прошивки наша плата перестает подавать признаки жизни, так как мы забыли написать код, мигающий светодиодами, или выполняющий более полезные функции (см. скриншот исходного кода).

## 6.1 Структура файлов

```

ARMatura
└── startup ..... код инициализации ядра процессора
    └── keil_startup_stm32f4xx.s ..... ассемблерный код инициализации
        └── system_stm32f4xx.c ..... синый код инициализации STM32F4
            └── system_stm32f4xx.h ..... синый код инициализации STM32F4
                └── stm32f4xx.h ..... STM32F4 CMSISVII Cortex-M4 Device Peripheral
                    └── core_cm4.h ..... CMSISVII библиотека поддержки ядра Cortex-M4
                        └── stdint.h ..... стандартные целые типы C+
                        └── core_cmInstr.h ..... CMSISVII Core Instruction Interface
                        └── core_cmFunc.h ..... CMSISVII Core Register Access Functions
                        └── core_cm4_simd.h ..... CMSISVII Cortex-M4 расширение SIMD/DSP
    └── blink
        └── blink.c ..... простая программа мигания светодиодом или дрыгания ногой
            └── stm32f4xx.h

```

В отличие от кристаллов типа Atmel AVR, при программировании для архитектуры Cortex-Mx принято использовать готовую библиотеку CMSIS**VII**, для которой проводится активная стандартизация среди производителей процессоров.

Cortex > Microcontroller > Software > Interface > Standard

<http://www.arm.com/products/processors/cortex-m/cortex-microcontroller-software-interface-standard.php>

Кроме основной части CMSIS**VII**, общей для всех производителей МК Cortex-Mx, каждый подставщик предоставляет аппаратно-зависимую часть библиотеки:

- стартовый код инициализации ядра процессора
- библиотеку периферии например STM32F Standard Peripheral Library, включающую код обеспечивающий доступ к устройствам, встроенным в кристалл в т.ч.
- USB OTG Host & Device
- DSP /SIMD Extension (для Cortex-M4F) системо-зависимую часть расширения CMSIS**VII** DSP

Библиотеки доступны для загрузки тут:

**STSW-STM32065** STM32F4 DSP and standard peripherals library, including 82 examples for 26 different peripherals and template project for 5 different IDEs

[http://www.st.com/st-web-ui/static/active/en/st\\_prod\\_software\\_internet/resource/technical/software/firmware/stm32f4\\_dsp\\_stdperiph\\_lib.zip](http://www.st.com/st-web-ui/static/active/en/st_prod_software_internet/resource/technical/software/firmware/stm32f4_dsp_stdperiph_lib.zip)

## 6.2 blink.c

Сначала посмотрим код основной программы:

Listing 6.1: /src/blink/blink.c

```
/*
 * ARMatura book sample
 * https://github.com/ponyatov/ARMatura
 */

#include <stm32f4xx.h> // загрузить CMSIS-библиотеку STM32Fx

int main()
{
    return 0;
}
```

Как видим — в ней нет абсолютно ничего сложного: только подключение библиотеки нашего МК и единственная функция `int main(void)`, содержащая только возврат в стартовый код CMSIS**VII** значения по умолчанию — 0.

Файл `stm32f4xx.h` подробно рассмотрен в ?? — на текущем слое знаний не стоит в него углубляться: он слишком большой и содержит практически весь список функционала, обеспечивающий CMSIS**VII**.

То же самое можно сказать и о других файлах — все они относятся либо к библиотеке CMSIS**VII**, либо к файлам поддержки производителя чипов.

Подробности будут рассмотрены далее в разделе **VII**, а пока важнее получить практический результат.

## Глава 7

### Hell Of World

## **Часть IV**

### **Средства разработки**

# Глава 8

## IDE

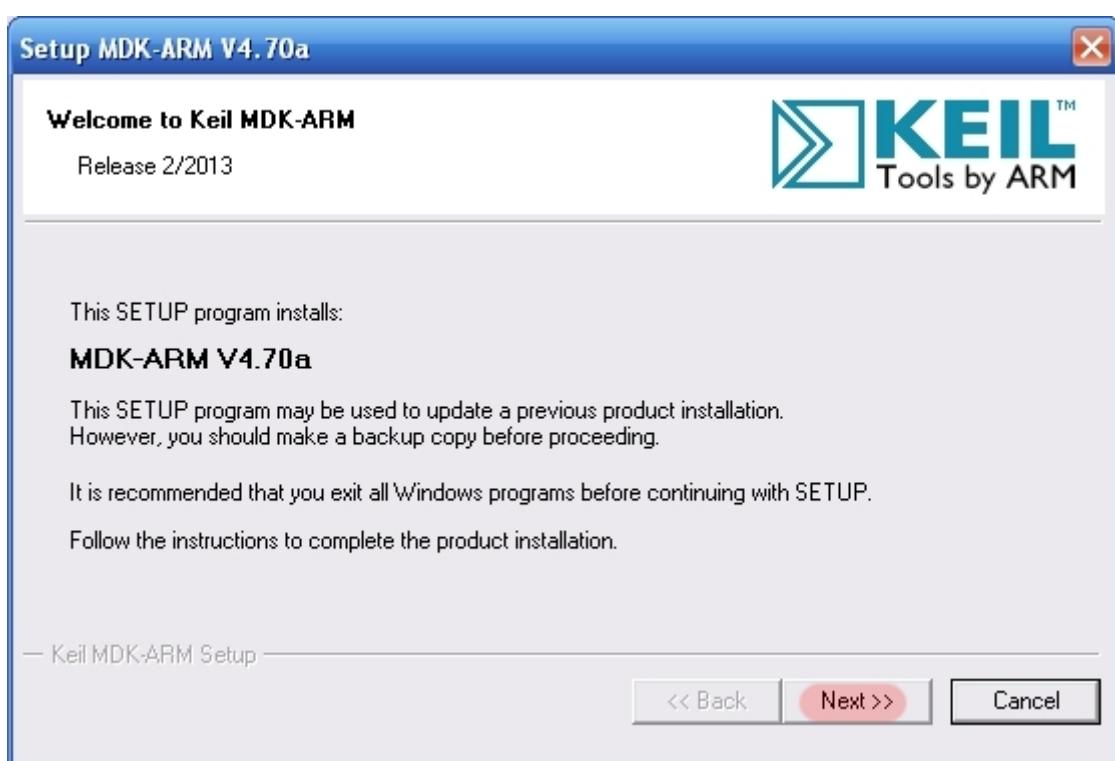
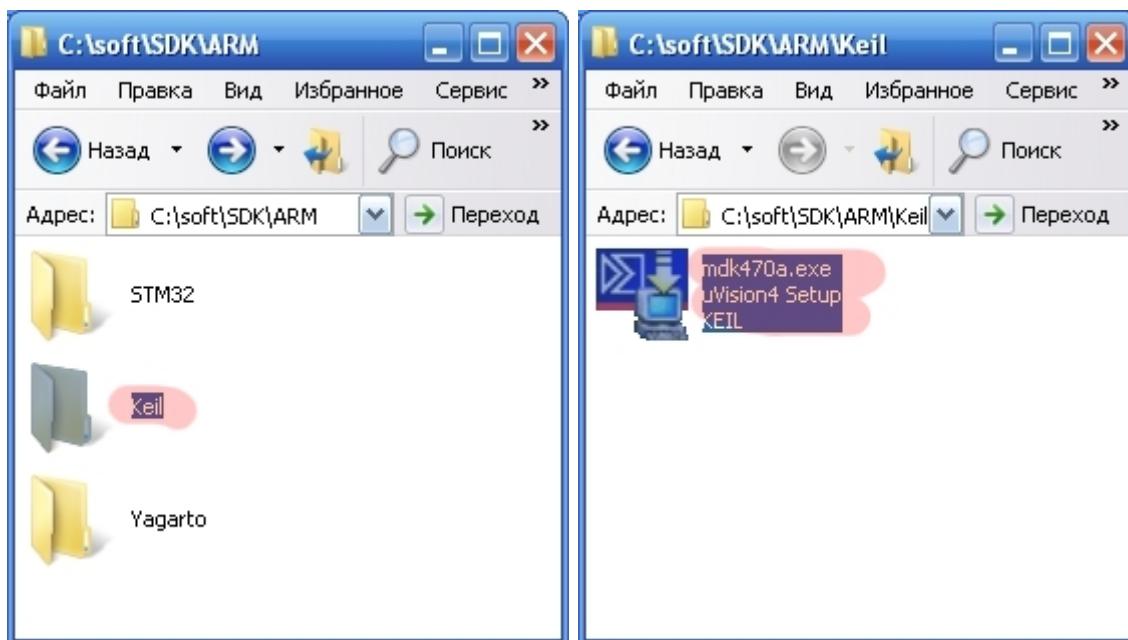
### 8.1 Keil

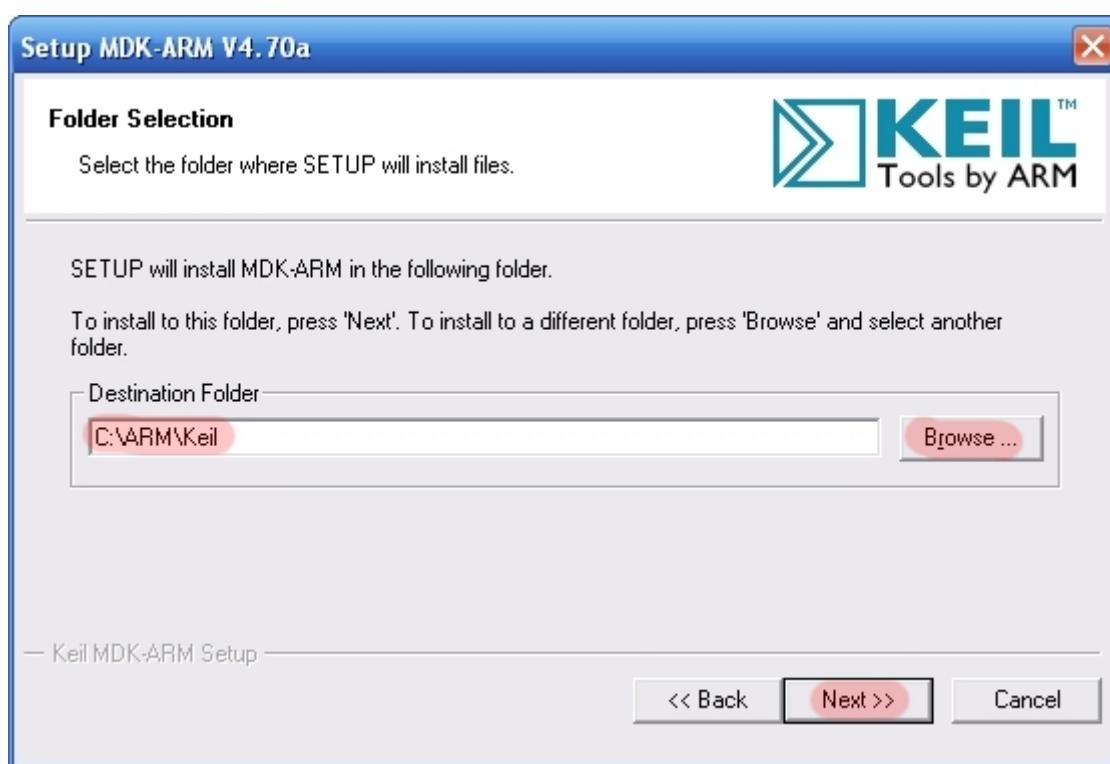
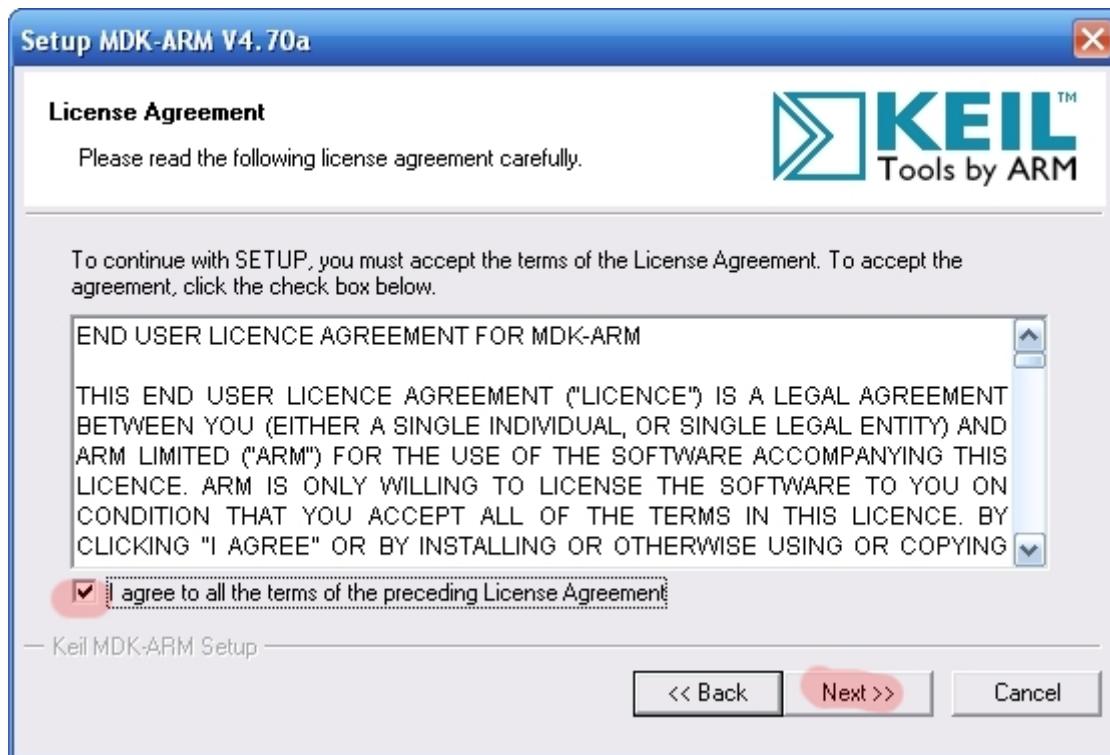


Для начала освоения программирования для ARM рекомендуем использовать бесплатный пакет от Keil: <http://www.keil.com/arm/mdk.asp> — ограничения бесплатной версии в 32К кода вполне достаточно для начального освоения программирования под процессоры семейства Cortex-Mx, а затем уже можно переползать на открытое ПО: GNU toolchain 9.1, Eclipse 8.4 и Linux XIII.

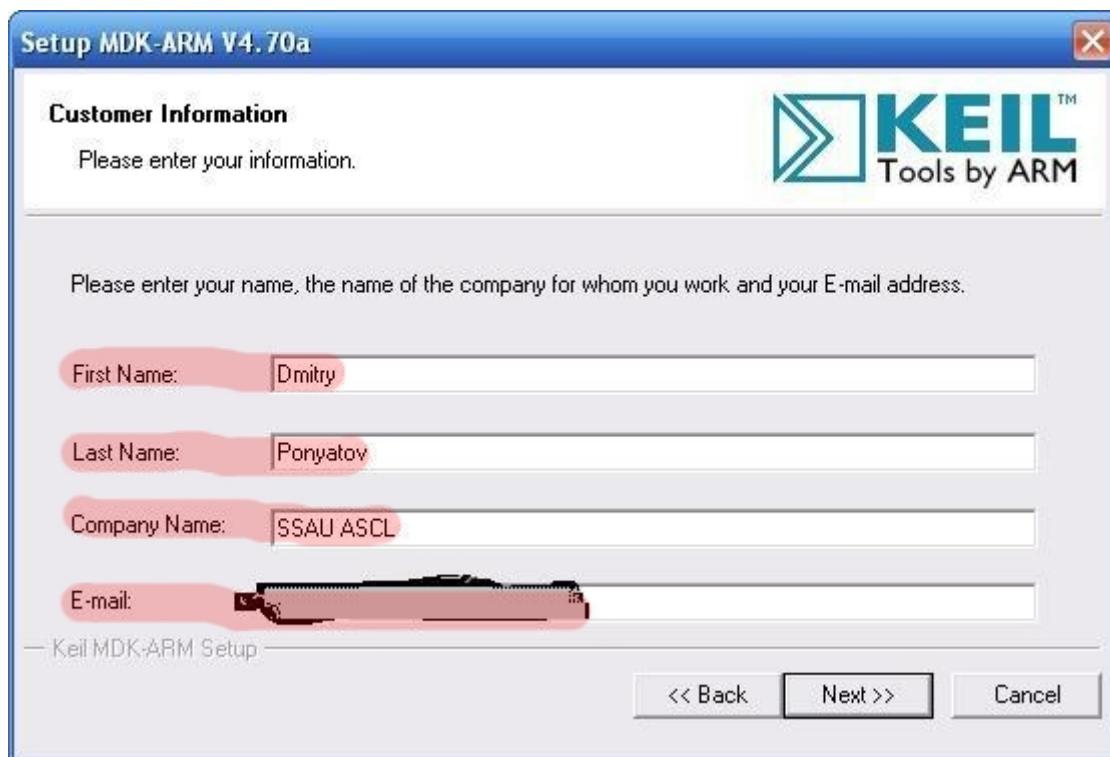
### Установка

Качаем пакет с официального сайта, заполнив анкету: <https://www.keil.com/demo/eval/arm.htm>.

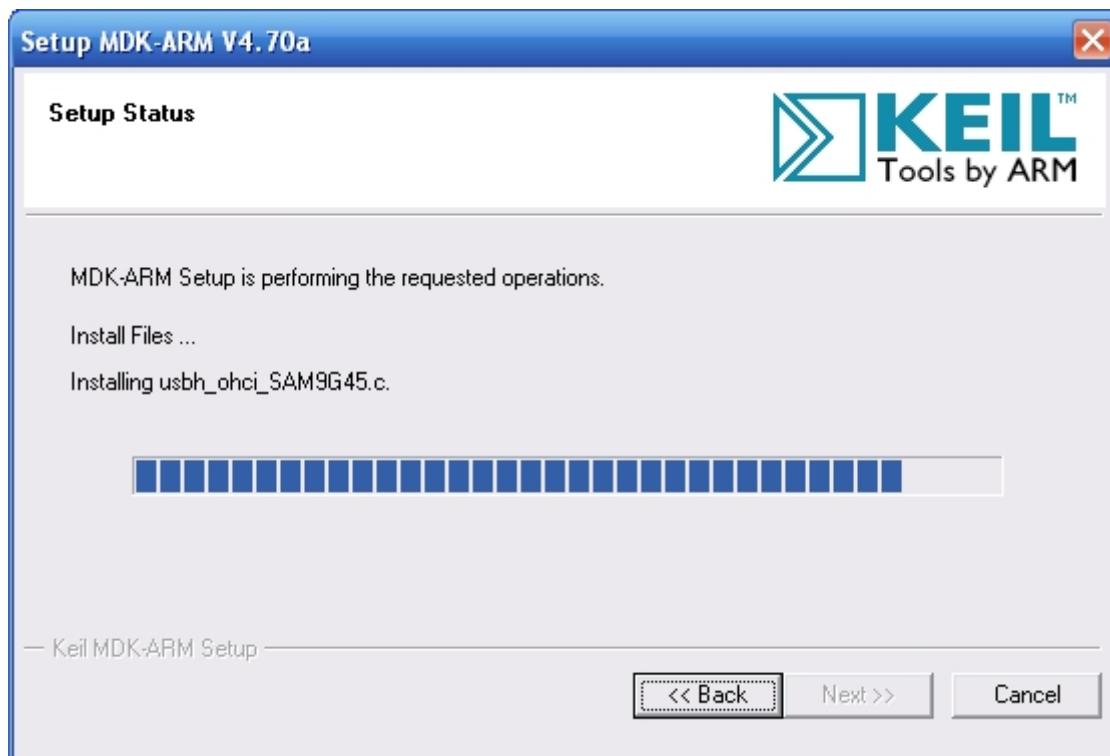


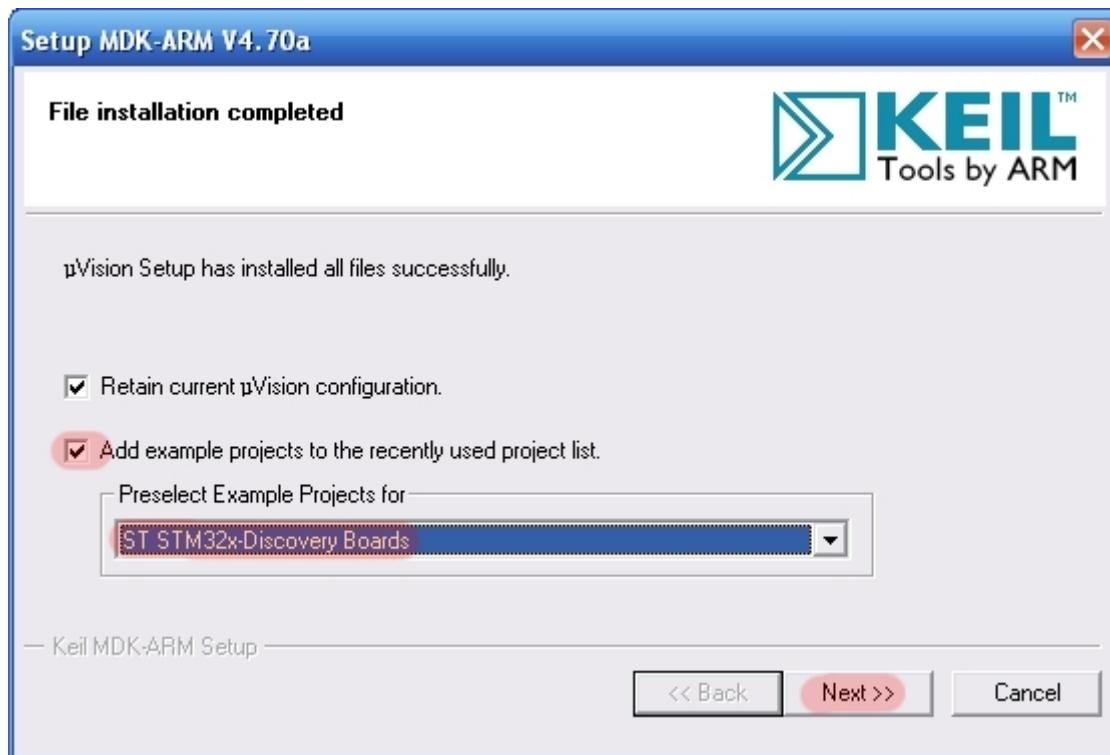


Путь установки пакета



Личные данные: имя, название компании или hobbit, адрес электронной почты.





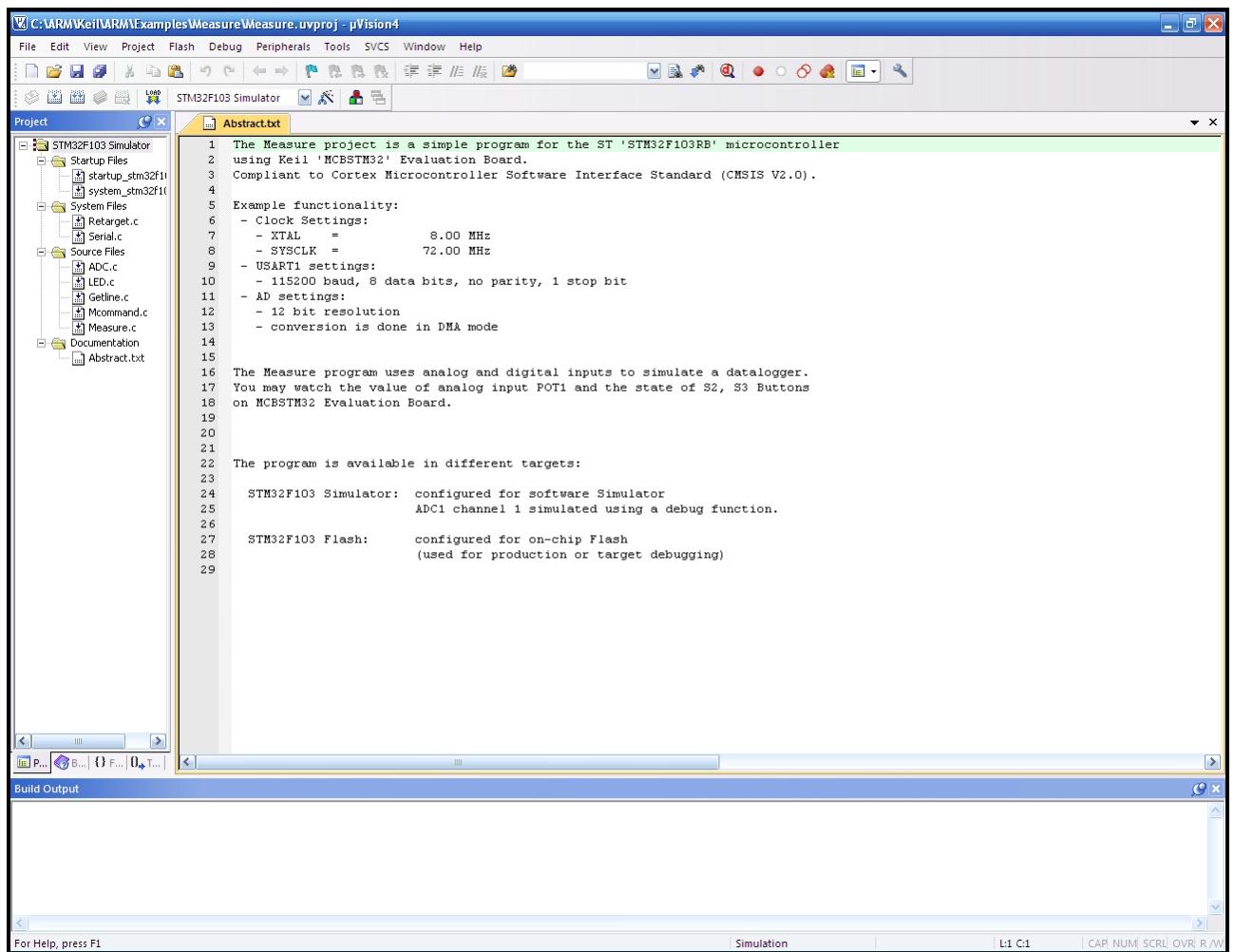
Укажите какие примеры кода добавить в список recently used project list: для работы с другими типами микропроцессоров выберете соответствующий раздел, или оставьте Simulation Hardware по умолчанию.



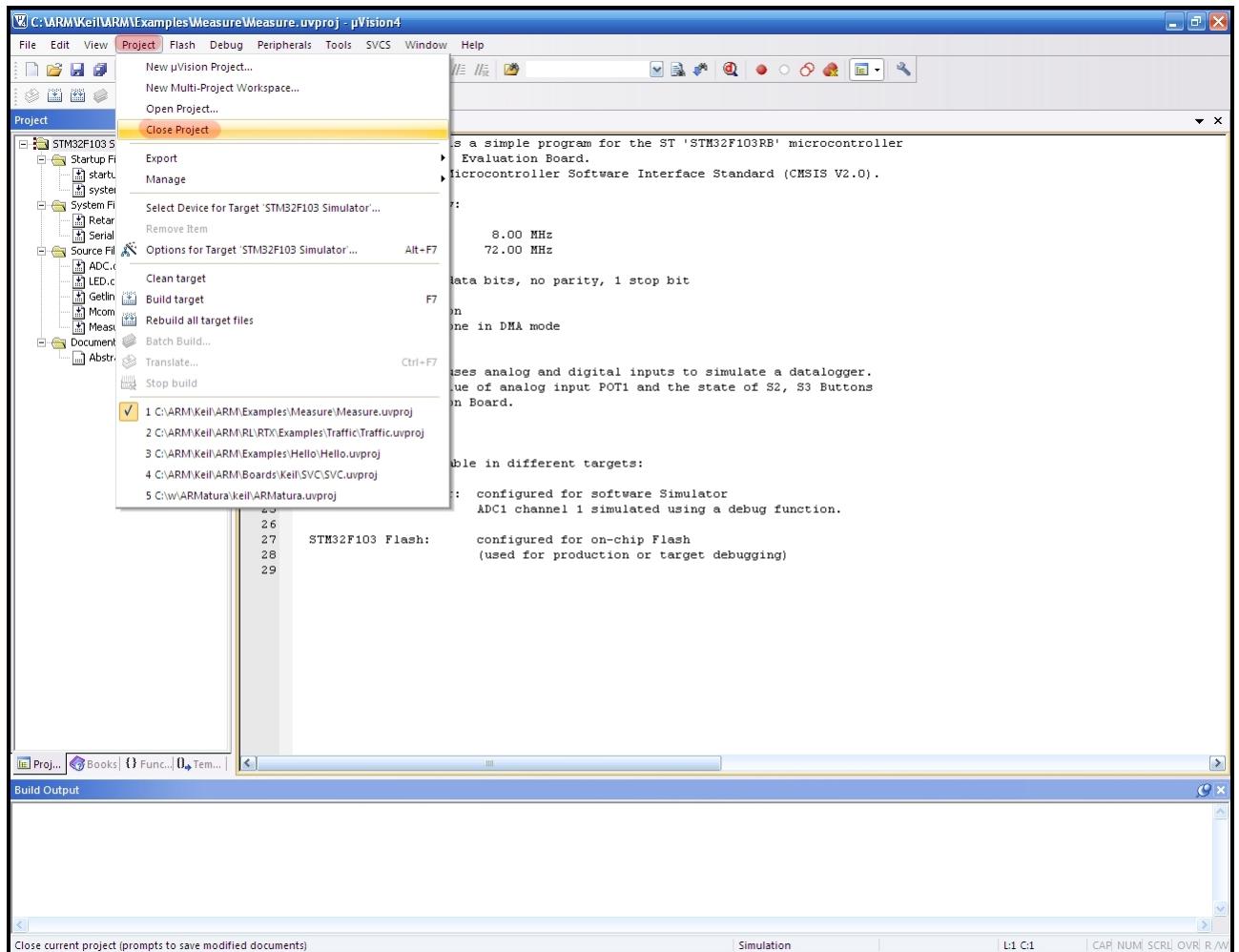
Снять установку драйвера программатора ULINK (если у вас его нет) и вывод текстового файла с последними изменениями Keil.



После запуска открывается проект по умолчанию, настроенный для программного симулятора STM32F103.



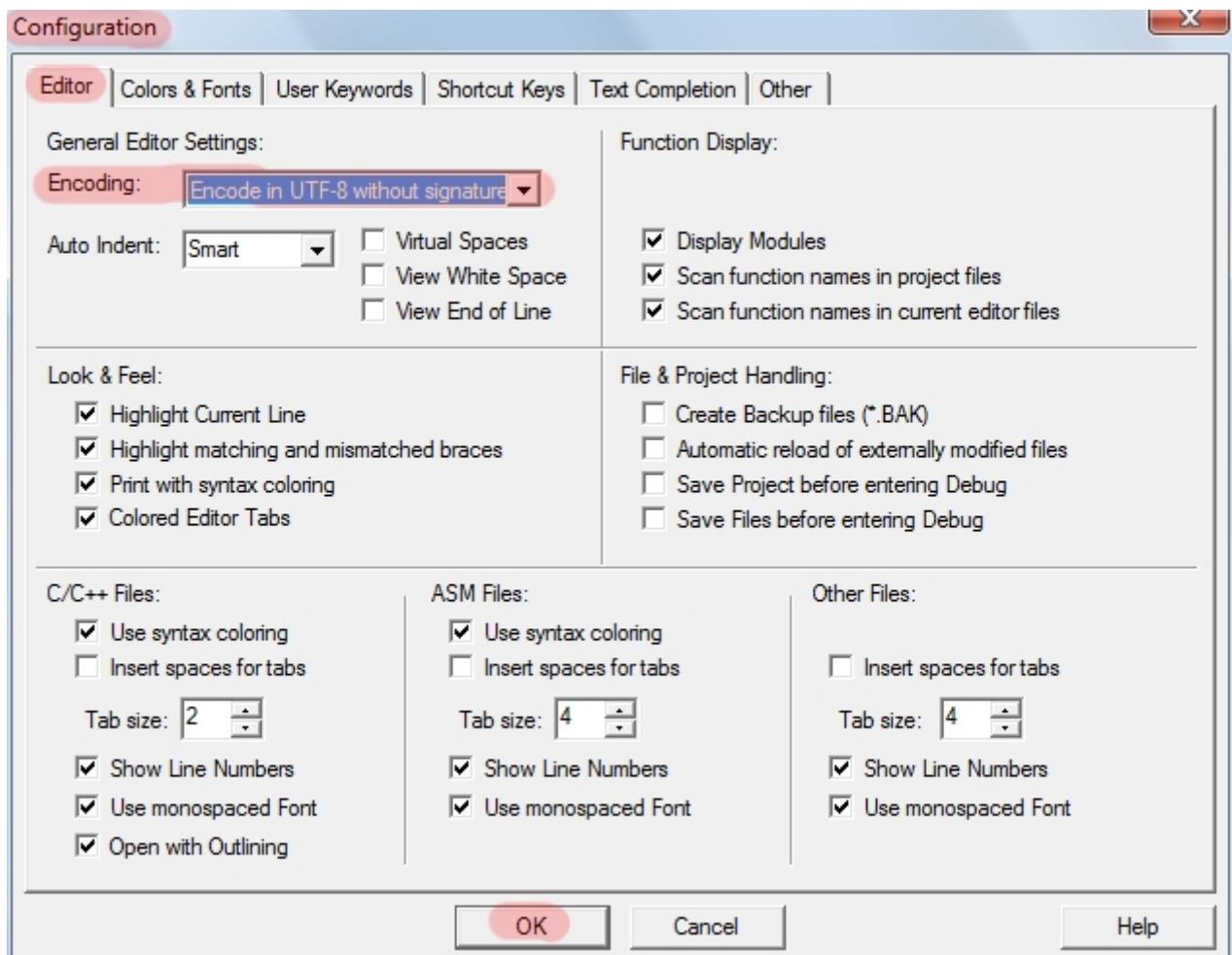
Нужно его закрыть,



и вернуться в раздел III и открыть первый проект 6.

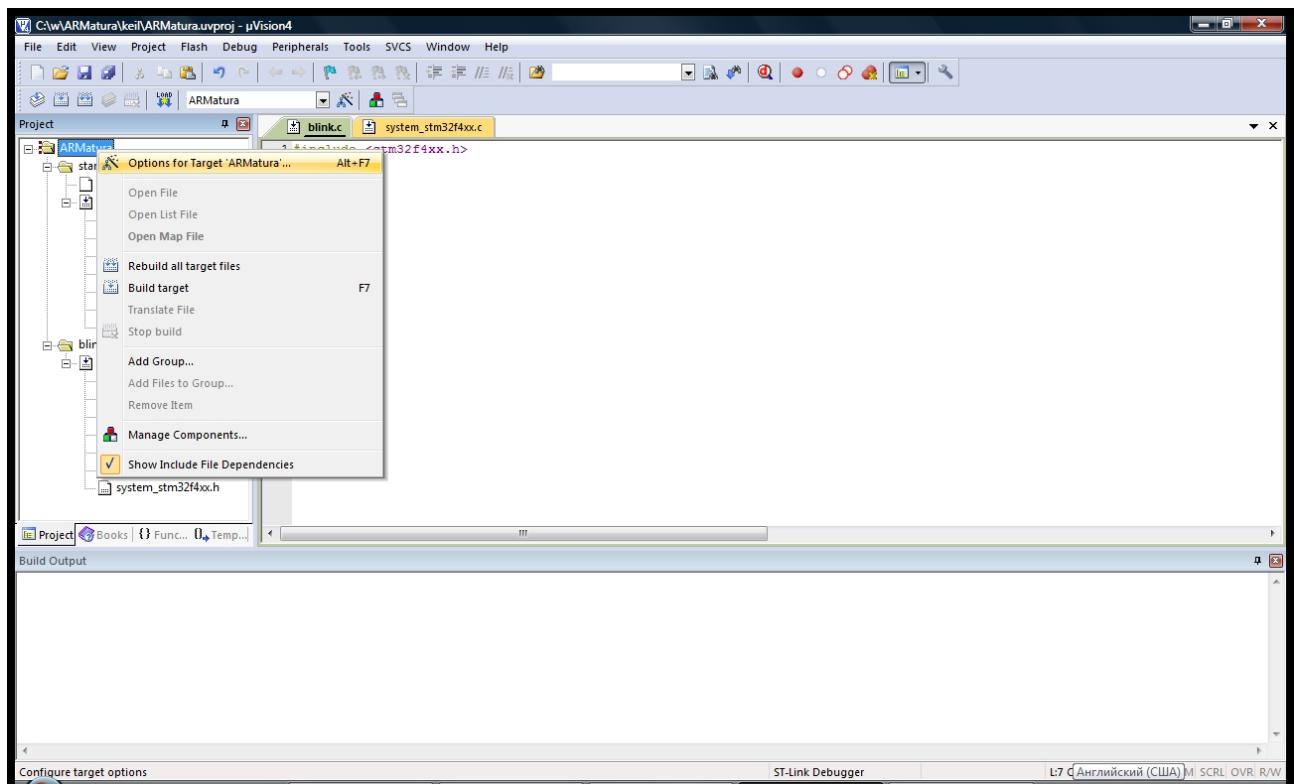
## 8.2 Настройки среды

По умолчанию Keil8.1 косячит с кириллицей, поэтому идем в меню **Edit > Configuration > Editor** и устанавливаем кодировку UTF8

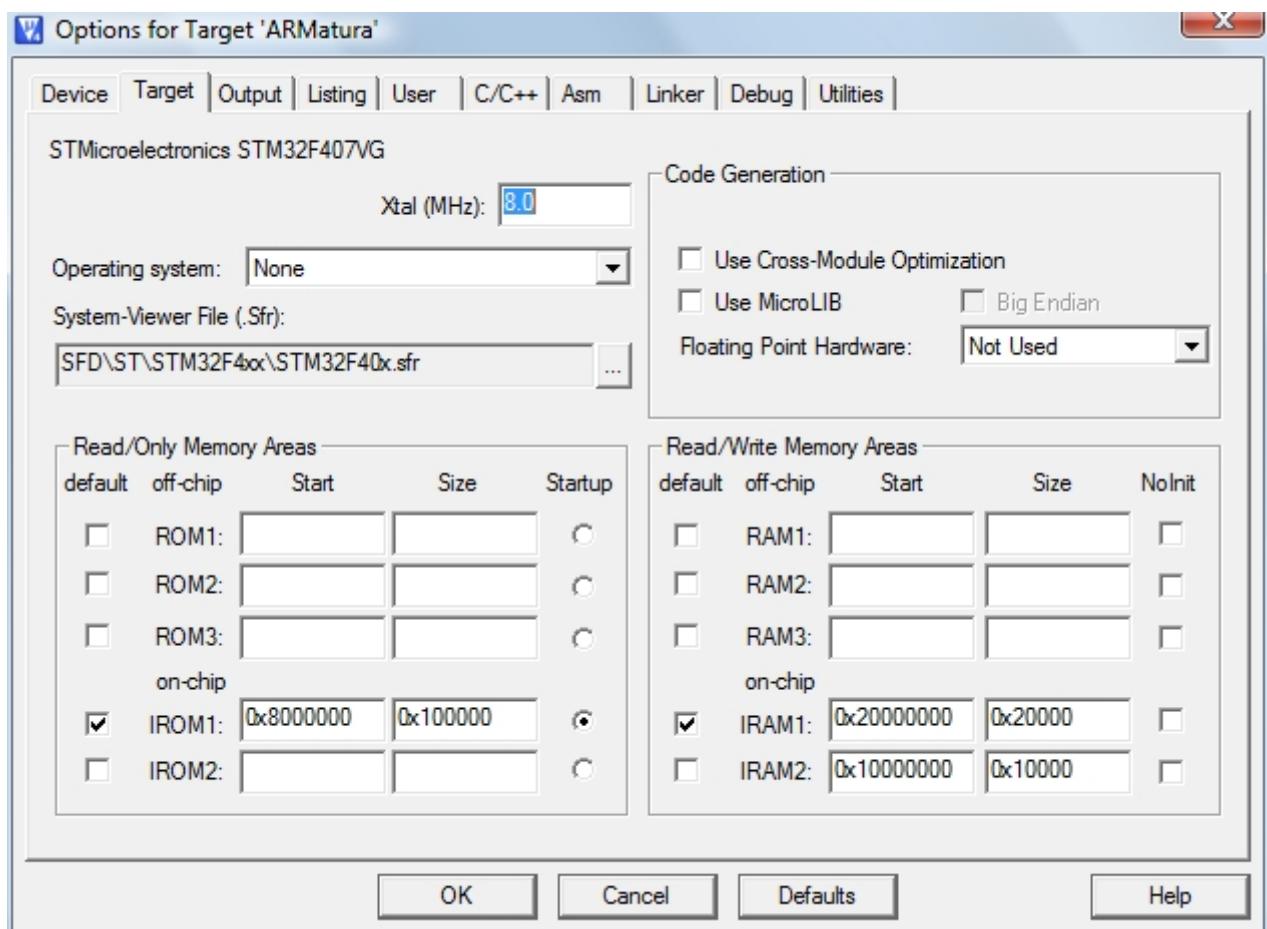


### 8.3 Настройки проекта

Настройки проекта вызываются из меню **Project** > **Options for Target 'ARMatura'...**, комбинацией клавиш **Alt** + **F7**, или выбором аналогичной опции из контекстного меню, вызываемого щелчком правой кнопкой мыши на корне дерева проекта **ARMatura** в левом окне **Project**.

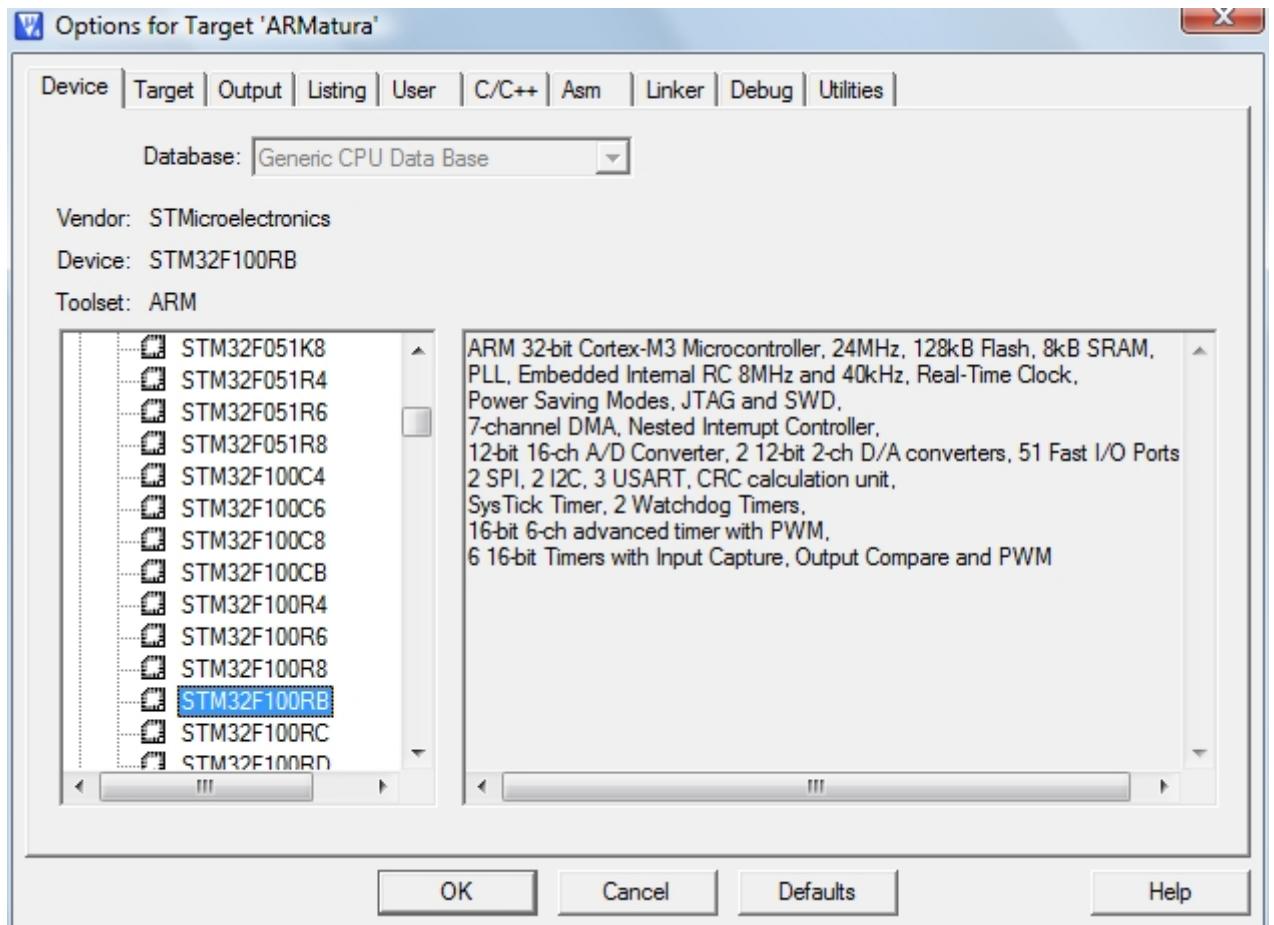


По умолчанию открывается вкладка **Target**:

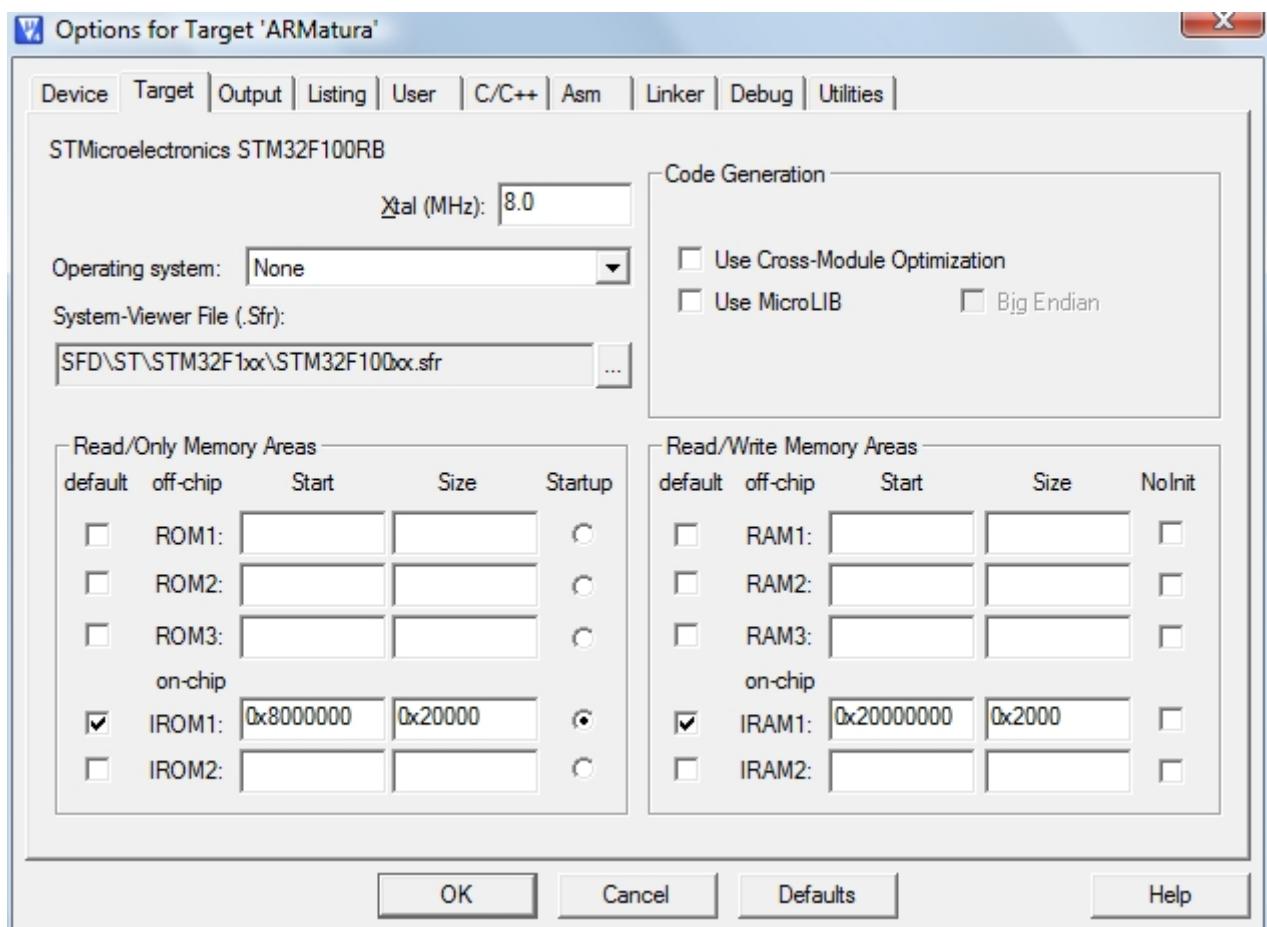


Xtal (MHz)	8.0	текущий процессор частота внешнего кварца на платах Discovery обычно стоит 8
Operating system:	None Use MicroLIB	или OCPB Keil RTX использовать libc от Keil
Floating Point Hardware:	Not Used	для Cortex-M4 доступен аппаратный FPU
	На этой вкладке также прописывается карта встроенной и внешней памяти Flash/SRAM.	

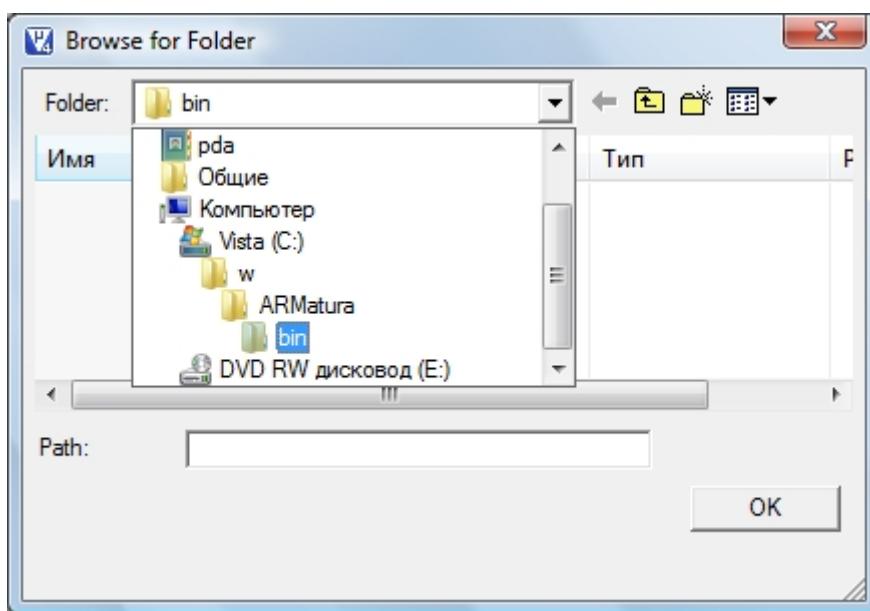
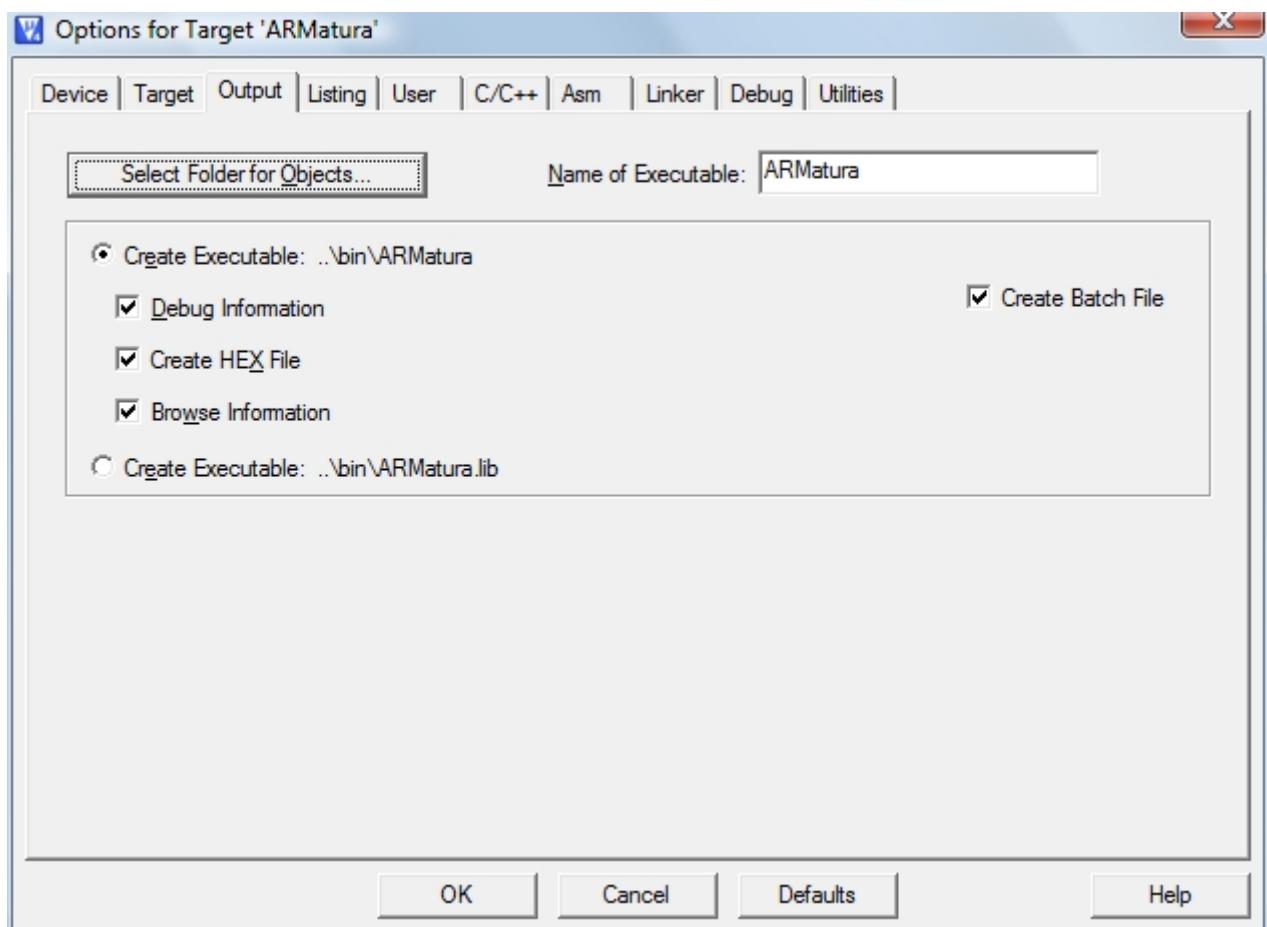
При необходимости собрать проект под другой процессор открываем вкладку **Device**, переконфигурируем проект под другую плату – STM32VLDISCOVERY 1:



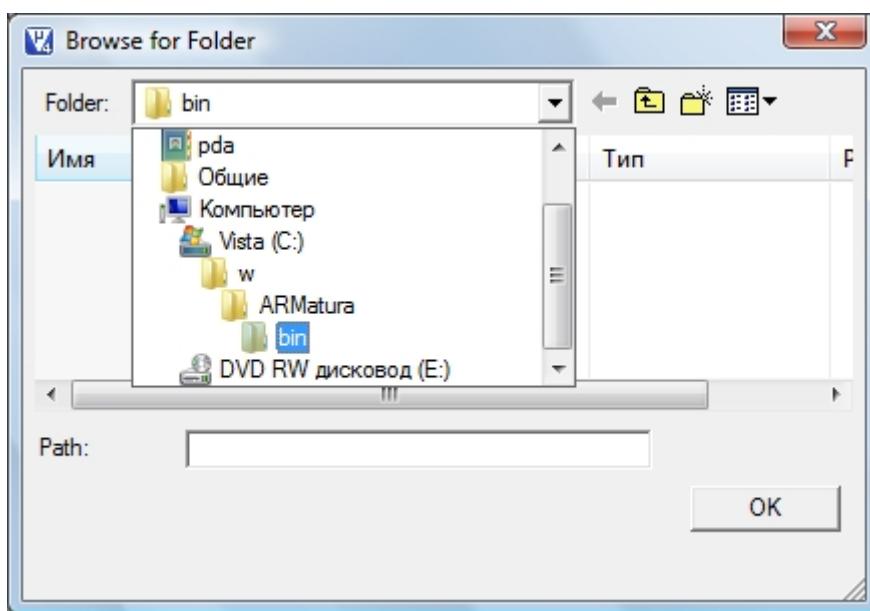
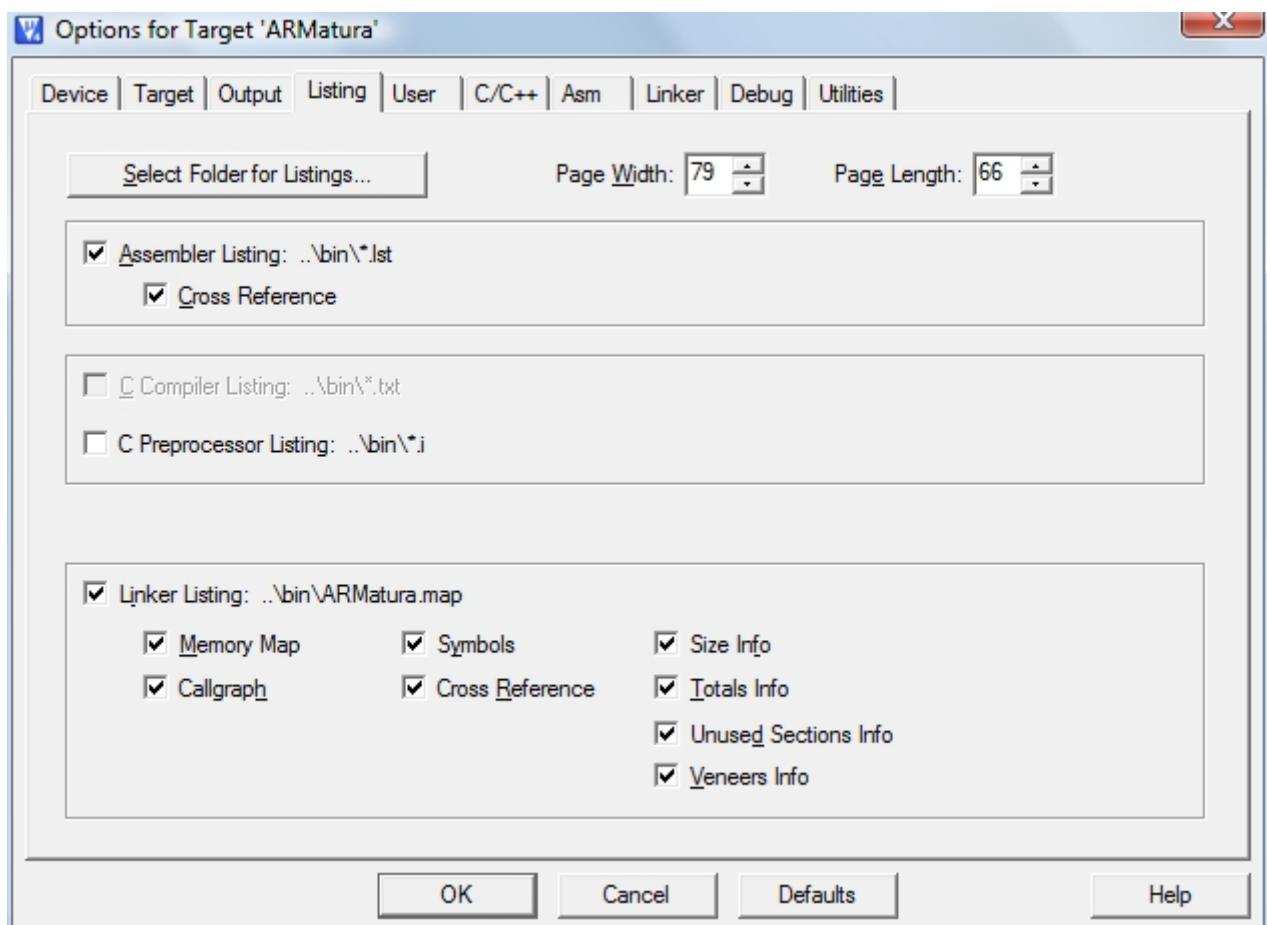
Обратите внимание что на вкладке **Target** изменился чип и настройки памяти:



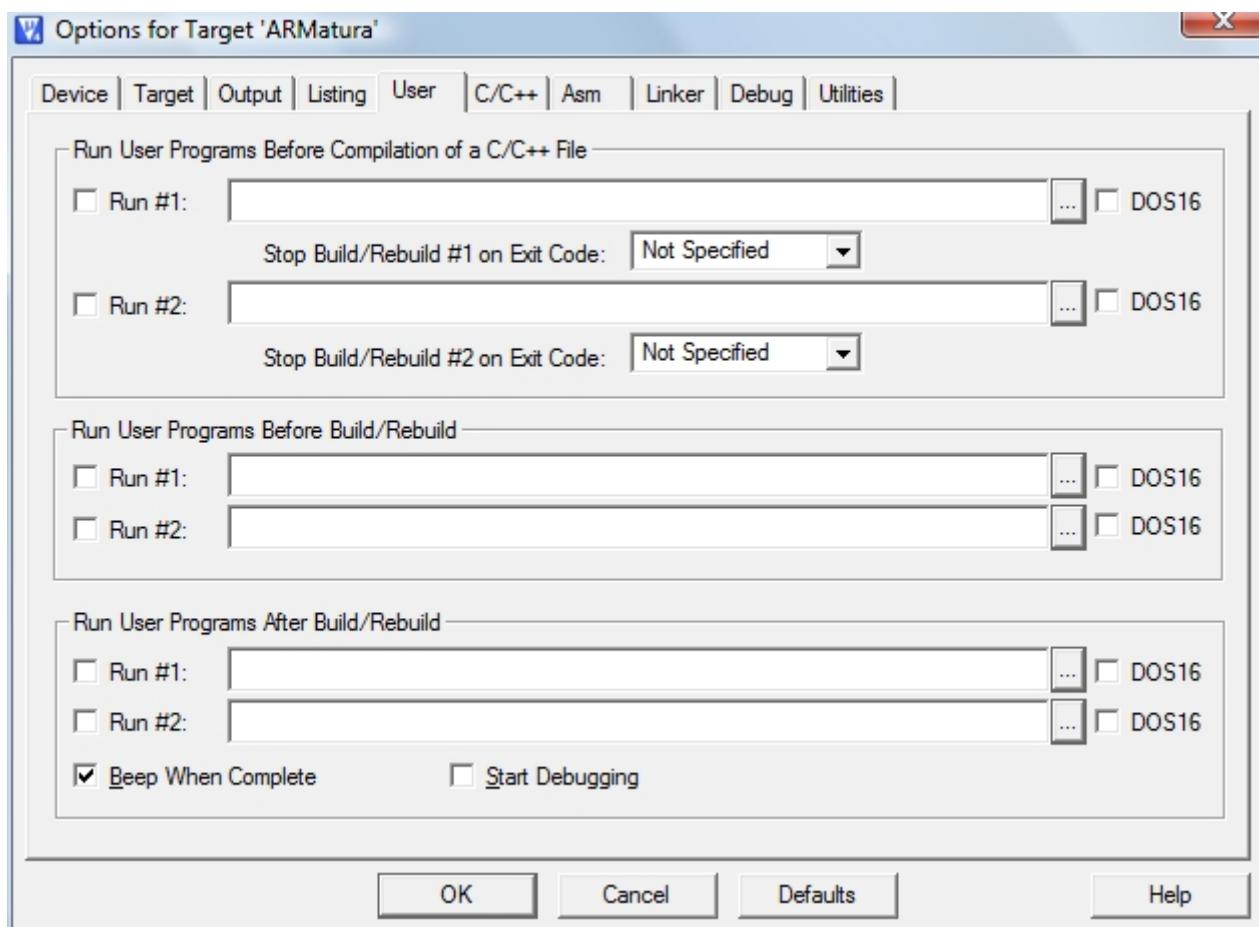
**Output** Настройки выходных файлов: каталог для бинарных файлов прошивки = firmware, при необходимости можно использовать .hex файл прошивки, имя файла прошивки



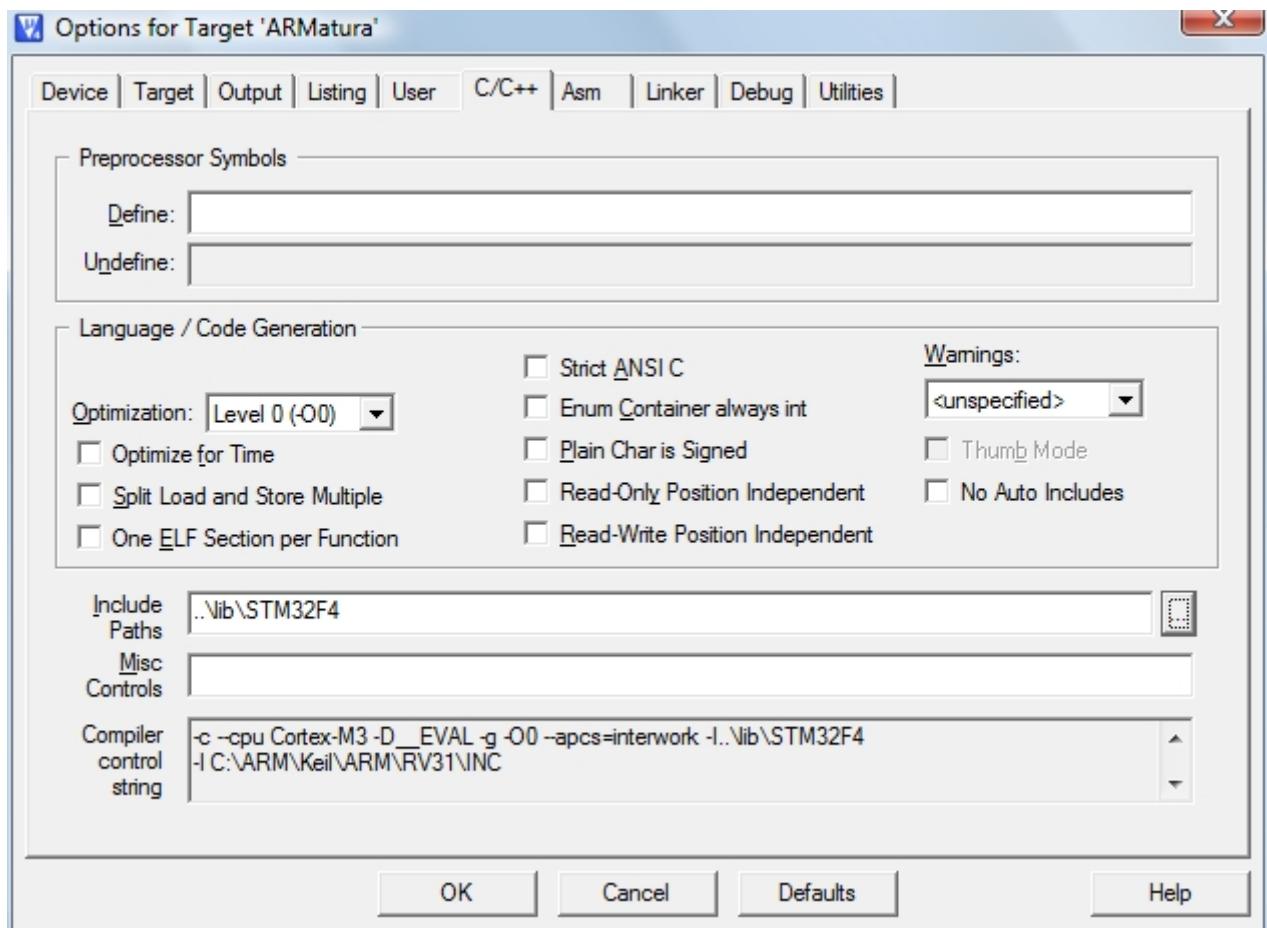
[Listing] Настройки генерации ассемблерных листингов в тот же каталог с прошивкой



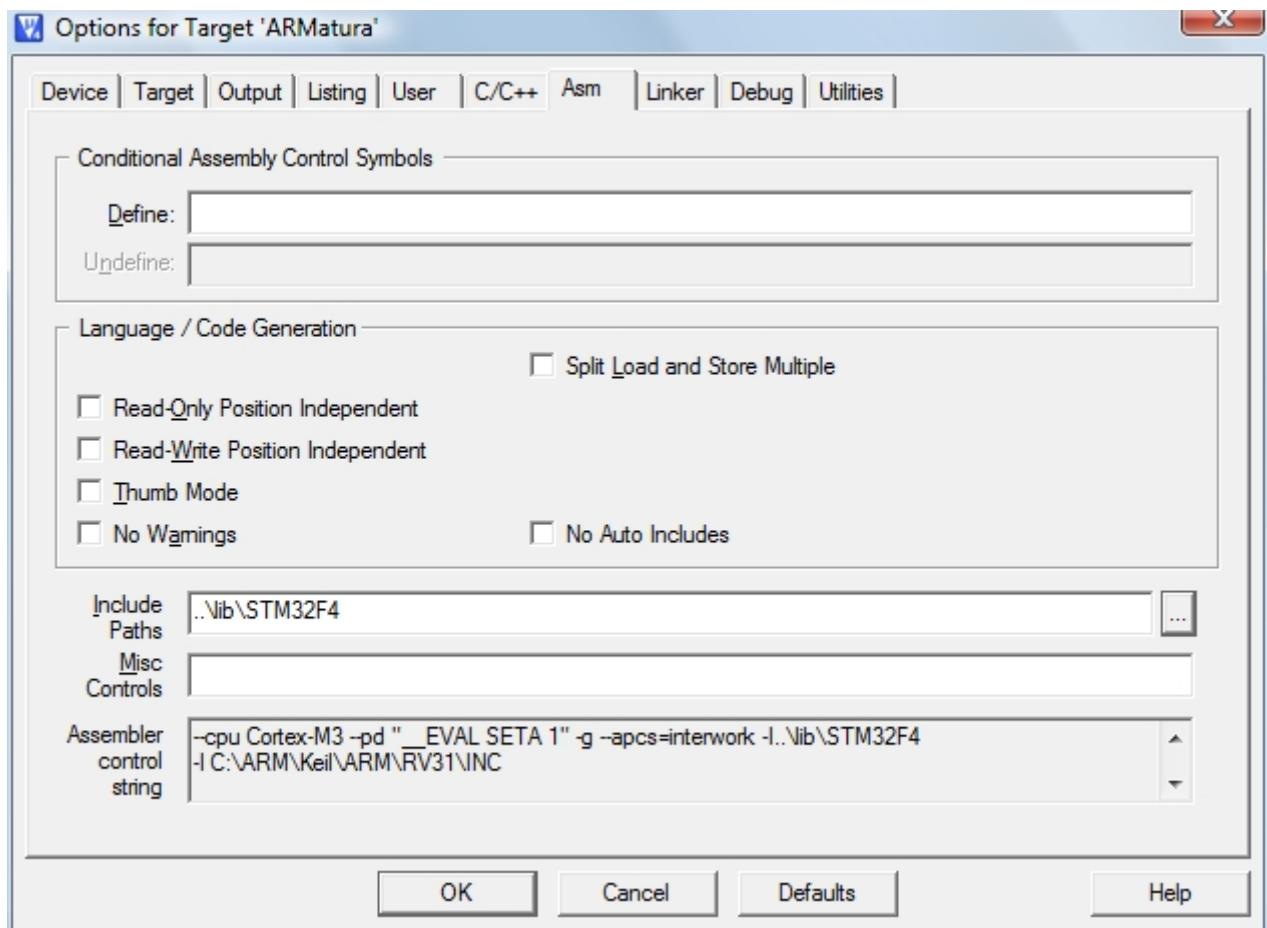
User Запуск пользовательских программ во время сборки проекта: пока не используем



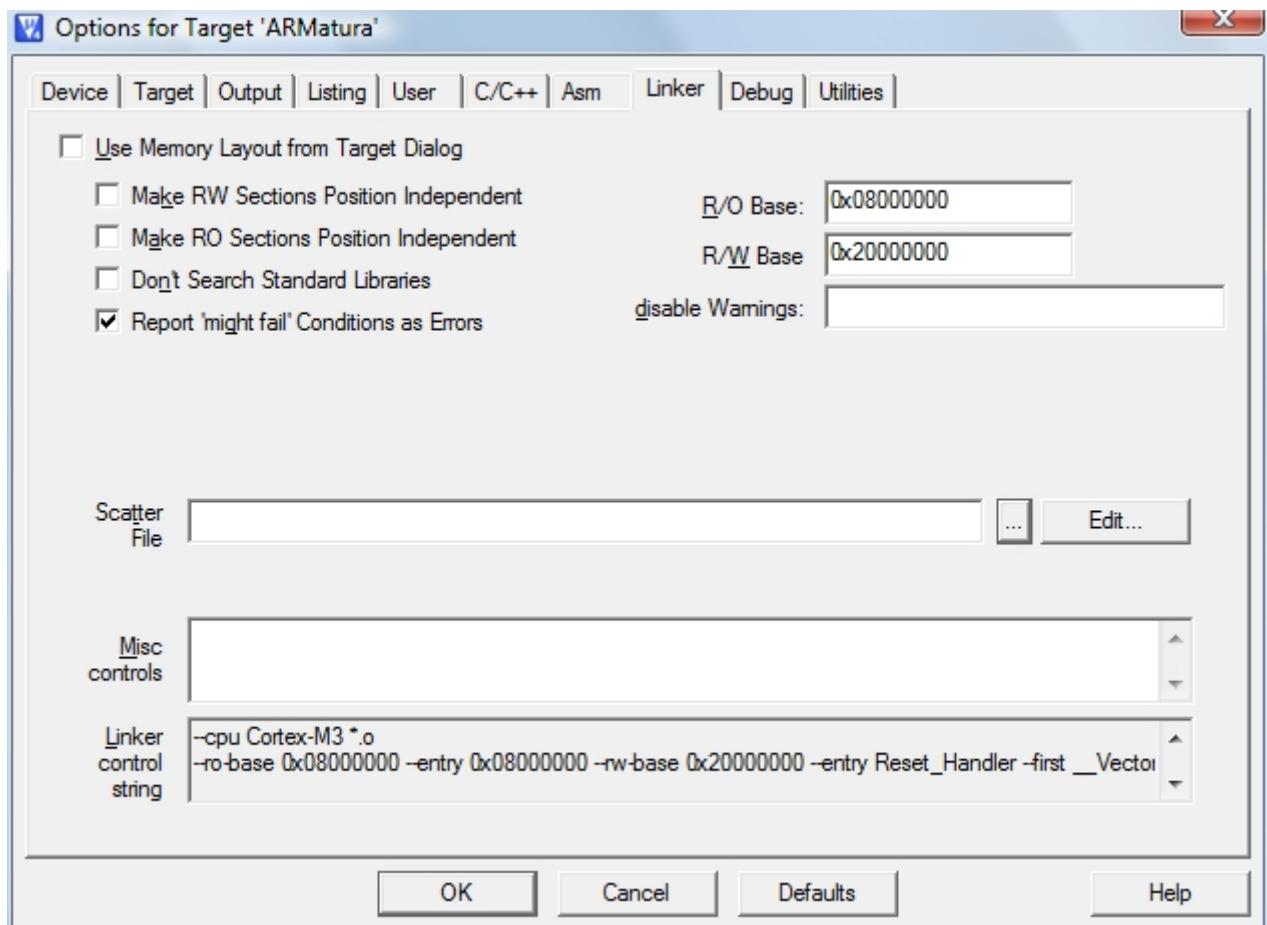
**C/C++** Настройки компилятора  $C^{++}$ : опции оптимизации, каталоги библиотек и включаемых .h файлов, в нижнем поле прописывается полная командная строка вызова компилятора, которая может вам в дальнейшем понадобится, если потребуется компилировать без использования Keil IDE.



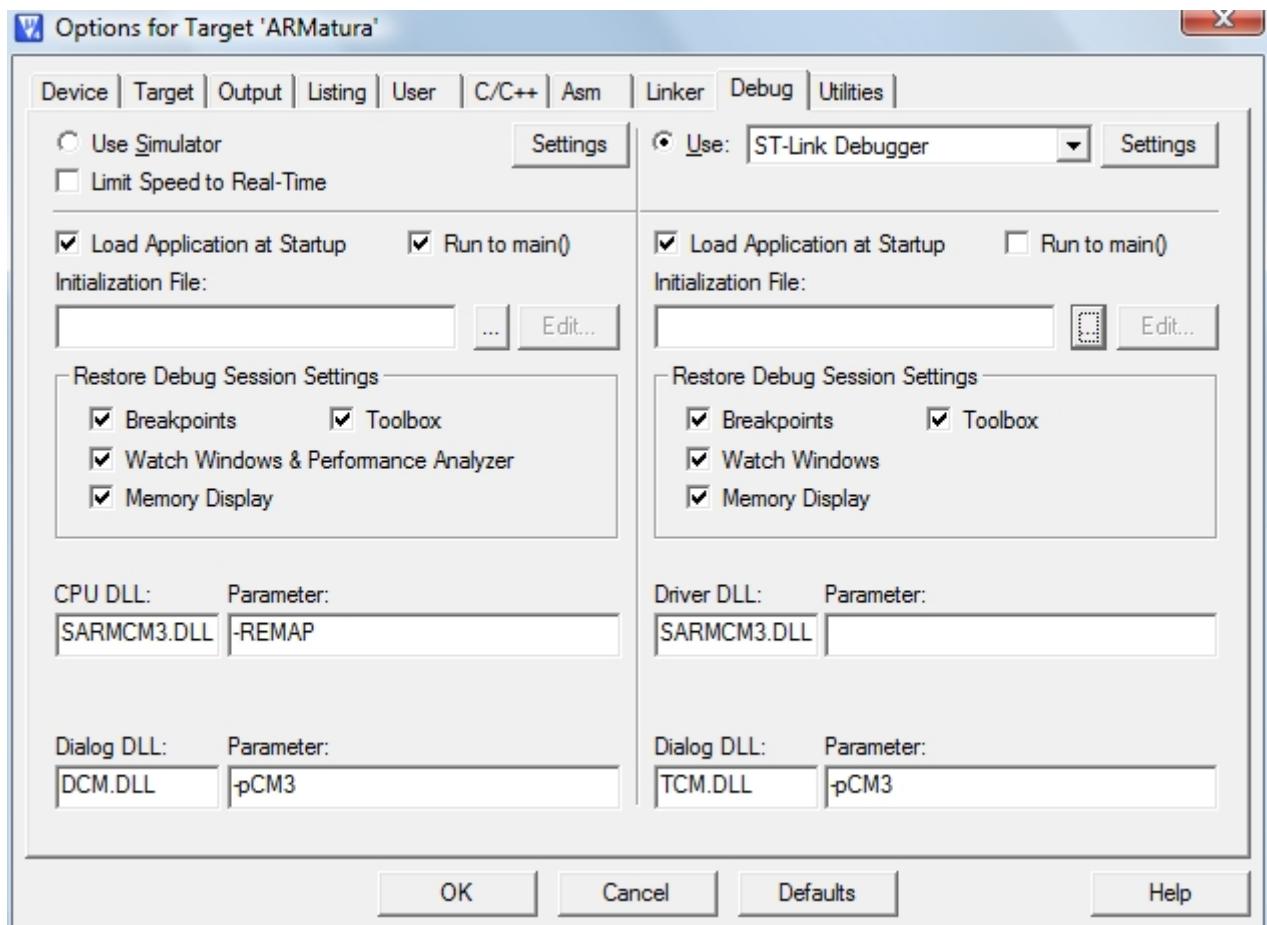
**Asm** Настройки ассемблера, приблизительно те же что и для  $C^{++}$



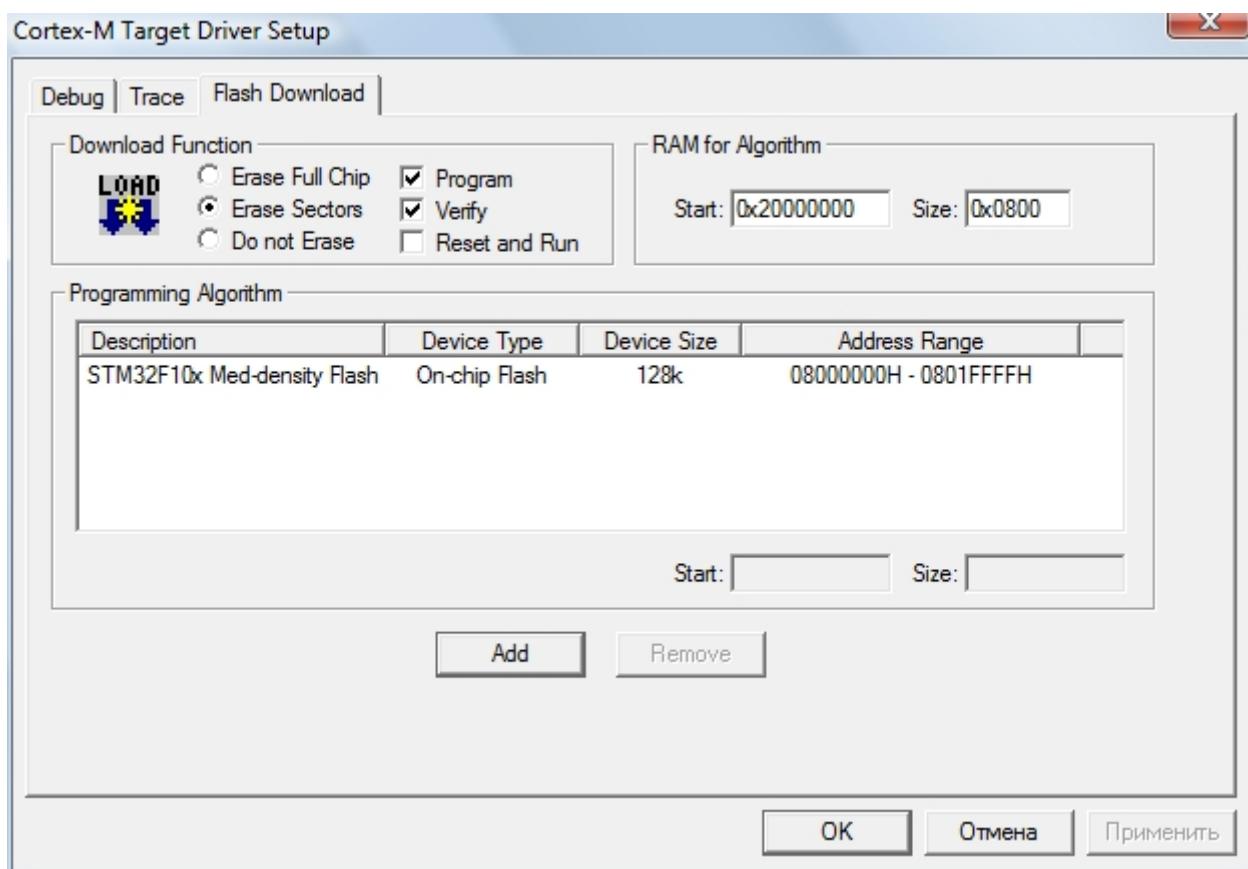
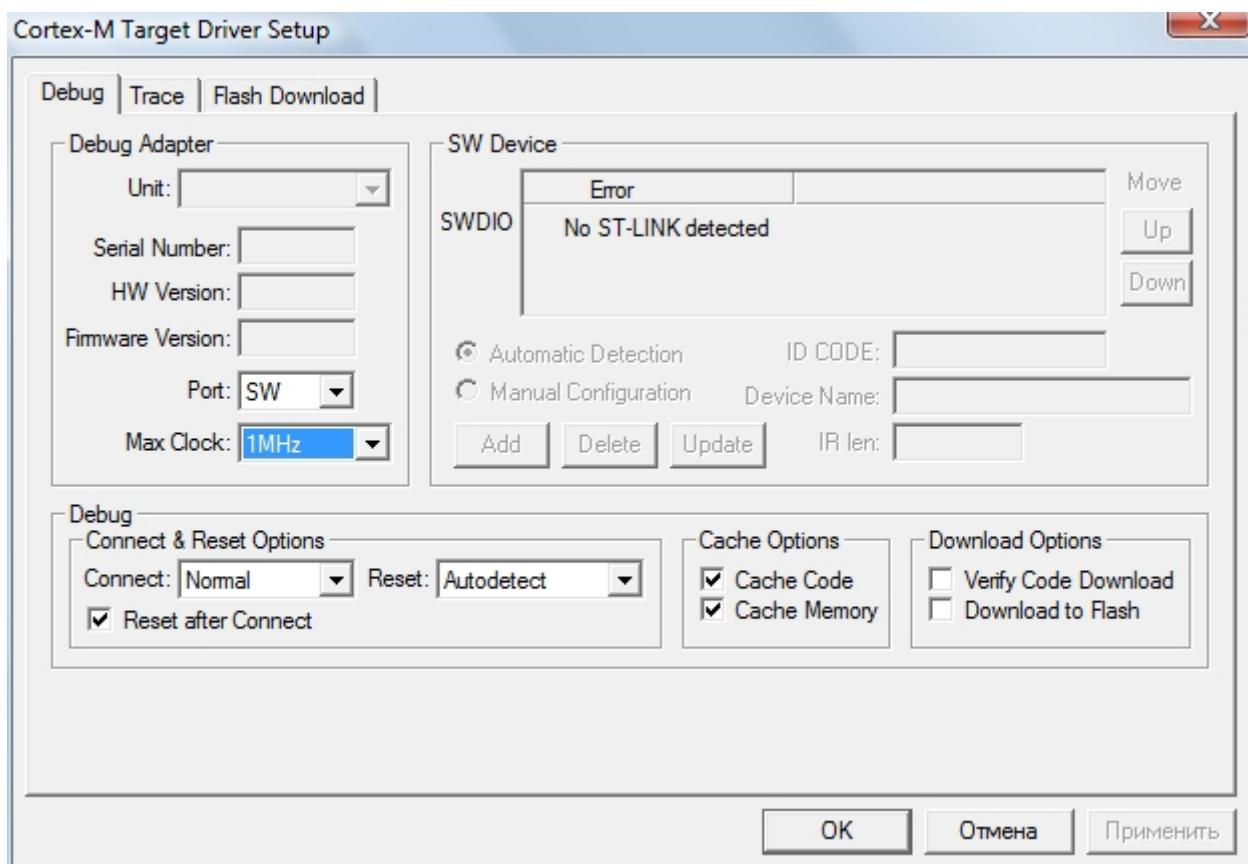
**Linker** Настройки линкера, который собирает объектные файлы .o, в которые компилируются каждый программный файл (модуль) по отдельности, в один готовый файл прошивки (настройки карты памяти контроллера, адреса точки входа программы и т.п.)



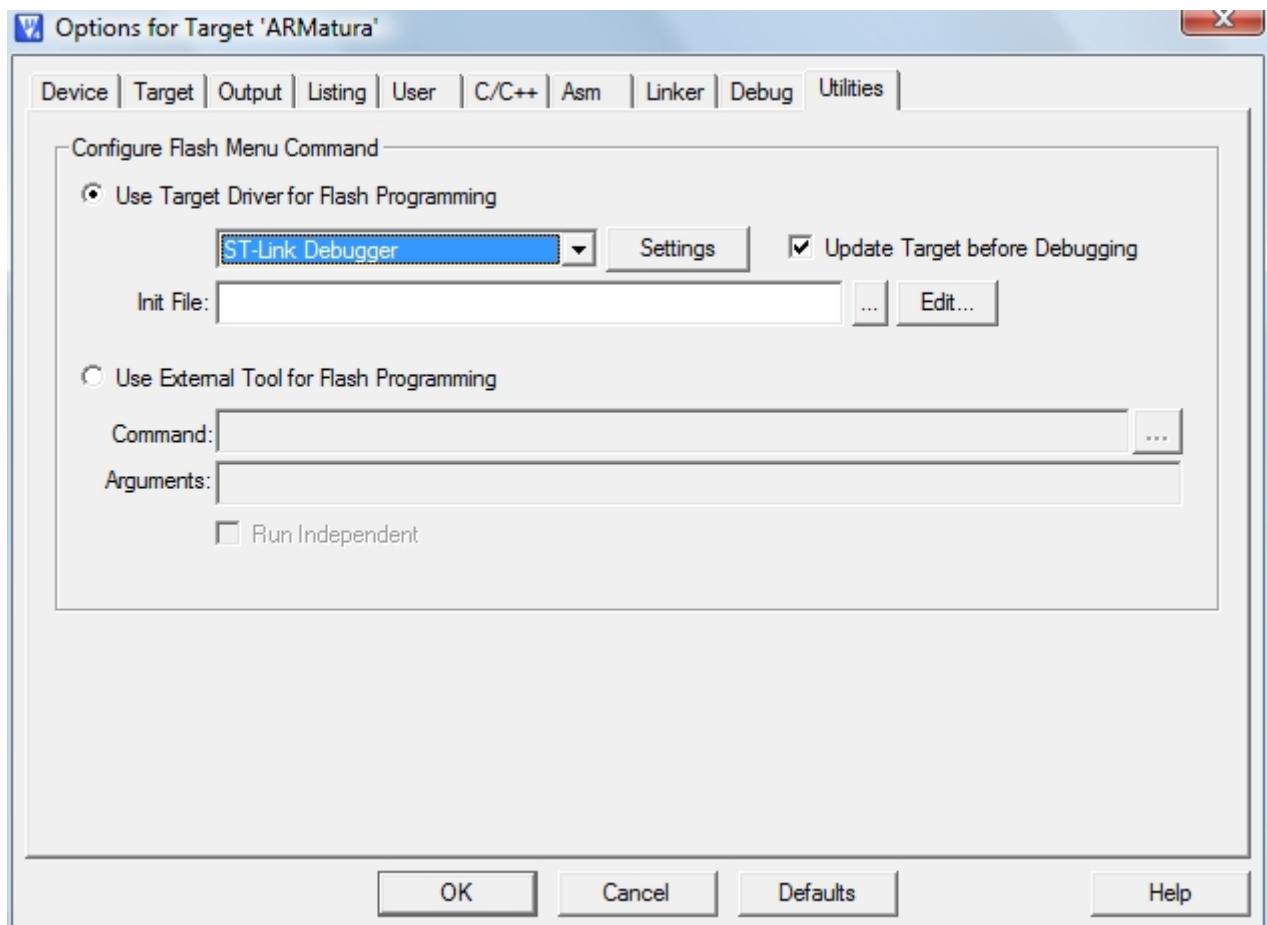
**Debug** Настройки отладки с помощью адаптера – STlink (внешний или встроенный в оценочную плату), JTAG.



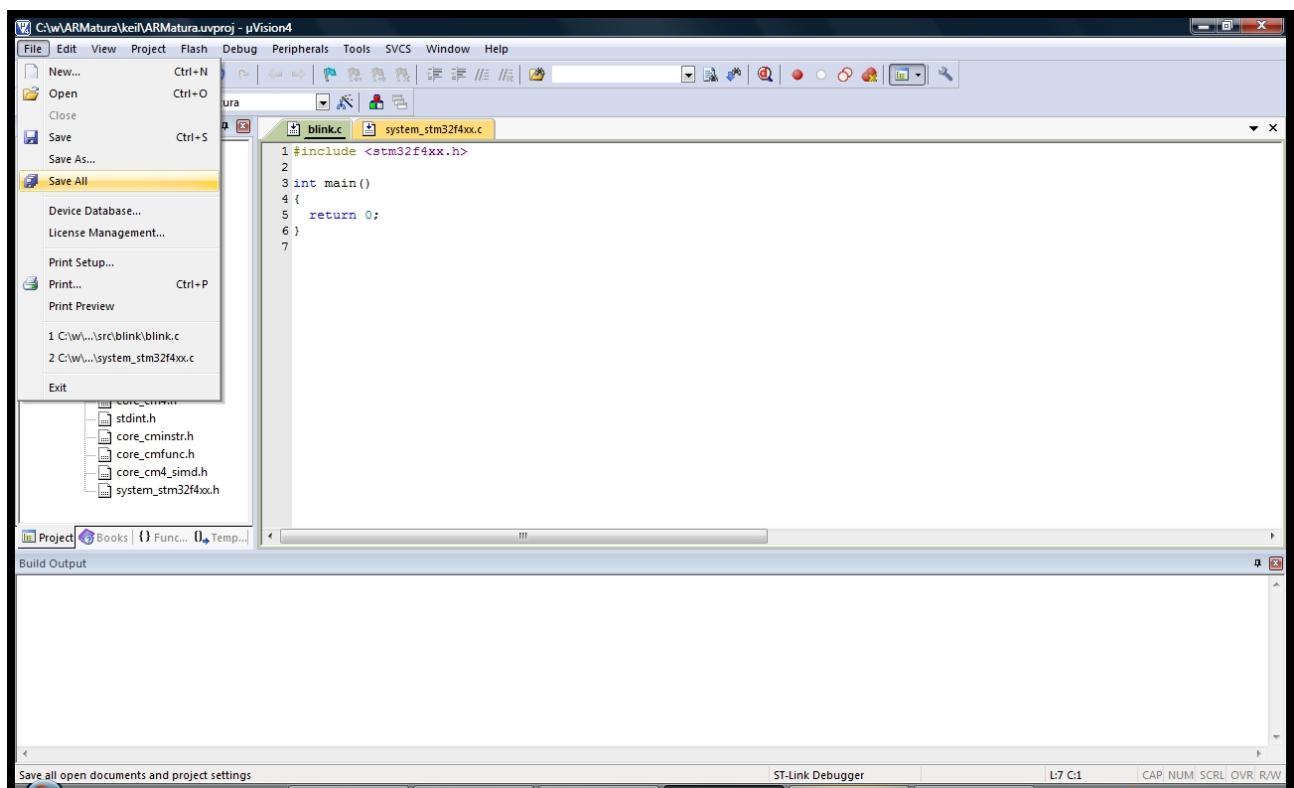
Настройки адаптера STlink – включаем режим SWD для варианта встроенного на оценочную плату, и обнаруживаем что не установили пакет поддержки STlink и драйвера.



**Utilities** Настраиваем в качестве программатора для прошивки тот же STlink



Сохраняем настроенный проект



## 8.4 Eclipse

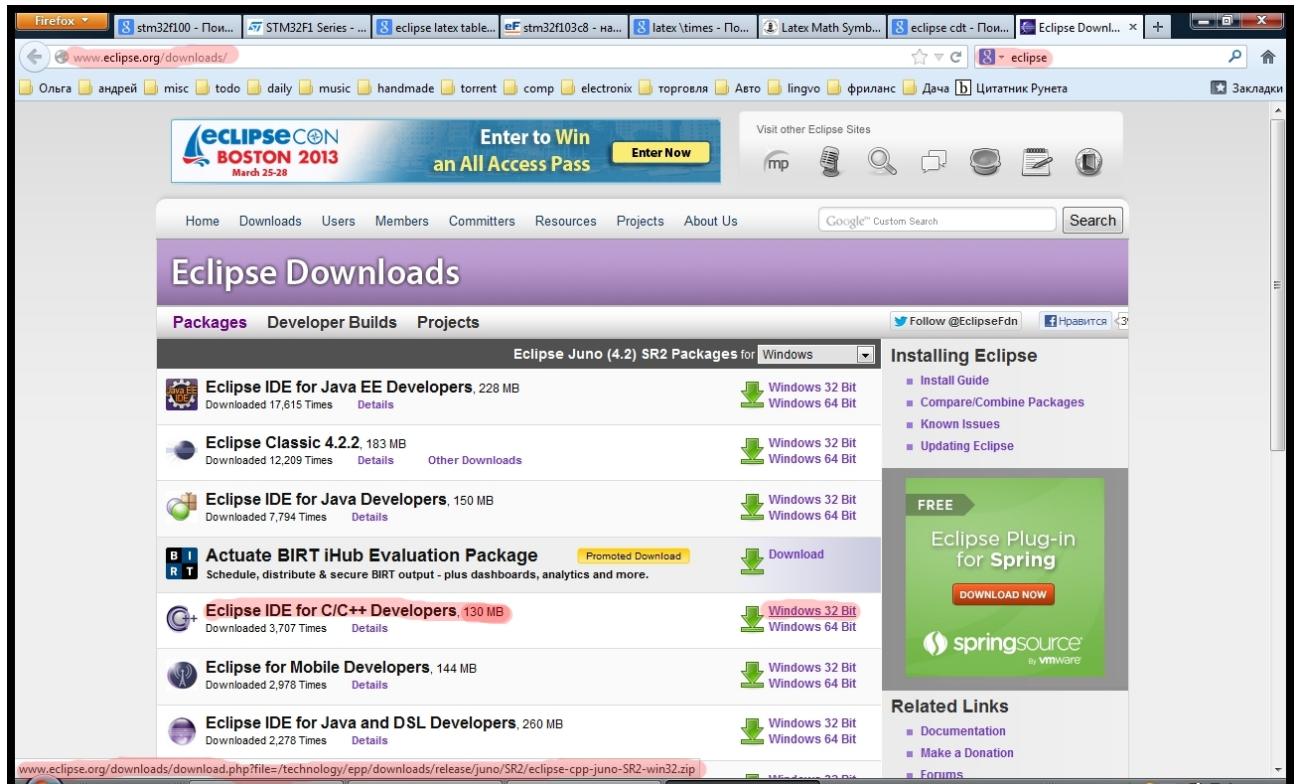
### Установка

Для работы нам потребуется сборка Eclipse 8.4 в варианте

```
http://www.eclipse.org/downloads/
http://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/
release/juno/SR2/eclipse-cpp-juno-SR2-win32.zip
```

Пакет поставляется в виде zip-архива, который достаточно распаковать в любое место, например в C:\Eclipse\.

После запуска нужно будет указать каталог для хранения проектов: C:\w\.

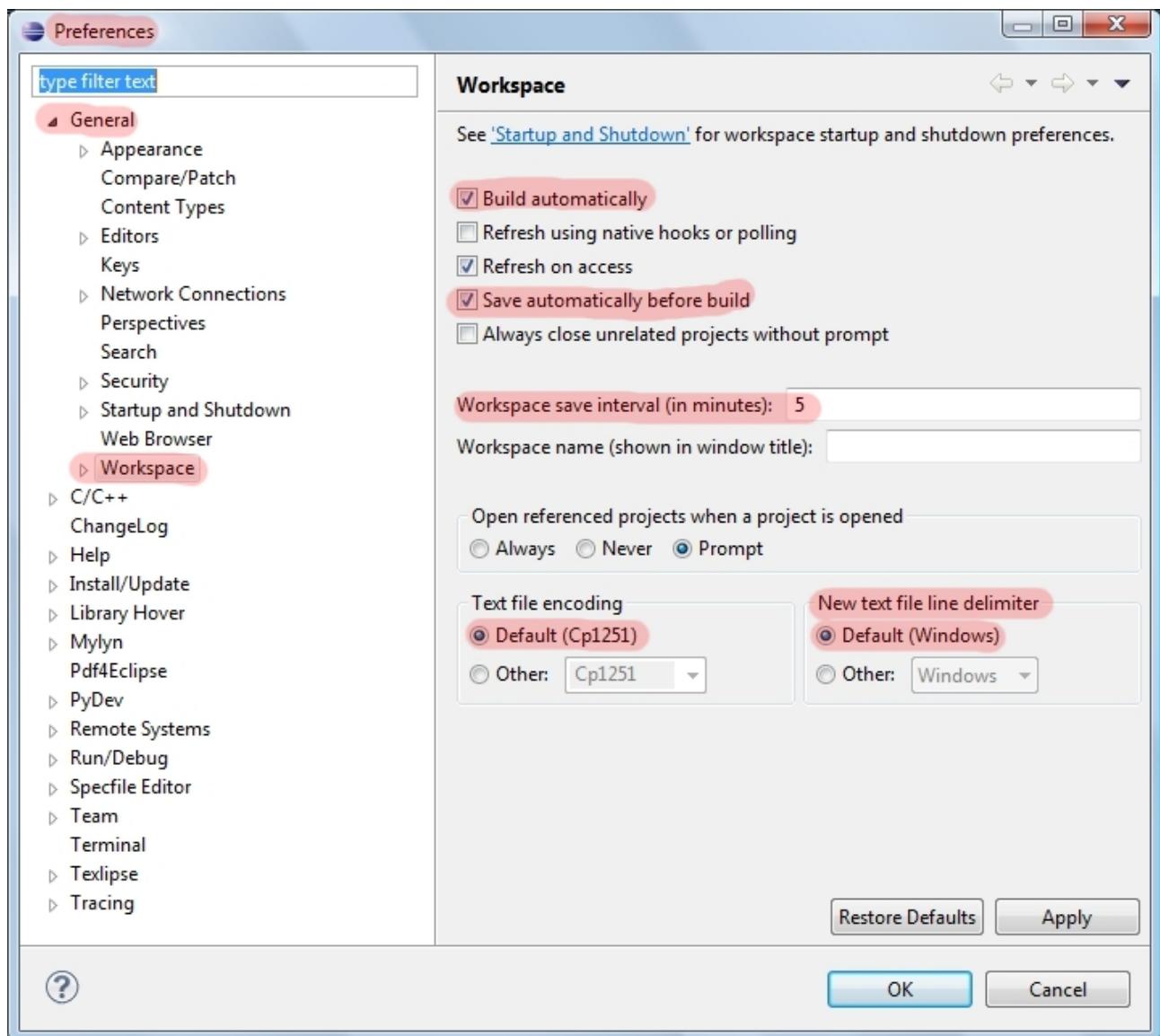


### Настройка глобальных свойств

Глобальные свойства настраиваются в меню **Windows > Preferences**.

Прежде всего настроим автосохранение при запуске Build **Ctrl + B**:

- запуск сборки при сохранении файла
- автоматическое сохранение всех измененных файлов перед сборкой
- автосохранение рабочих файлов
- кодировка файлов с кириллицей (может потребоваться корректировка при импорте файлов в кодировках KOI8, UTF8)
- конец строки в стиле dos/windows: <CR><LF> (в UNIX используется только <LF>)



## 8.5 Code::Blocks

## 8.6 gVim

## 8.7 IAR

# Глава 9

## Компиляторы

9.1 GCC

9.2 KeilCC

9.3 IAR

# Глава 10

## Адаптеры

Адаптер = программатор = внутрисхемный отладчик = ... — аппаратное устройство, подключаемое одним концом в компьютер (обычно по USB), а другим в отлаживаемое устройство с микроконтроллером.

По подключению к устройству разделяют JTAG-адAPTERЫ, адAPTERЫ, впариваемые производителями конкретных чипов (Atmel, STmicroelectronics), и коммерческих IDE (Keil Ulink), и разнообразный самопал.

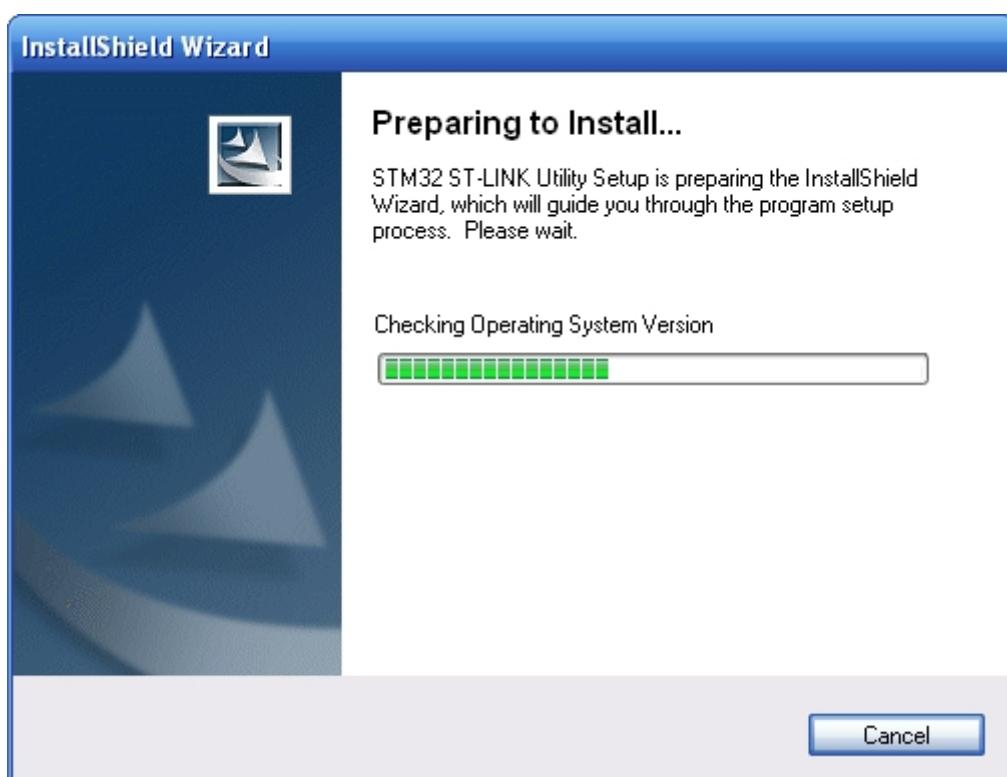
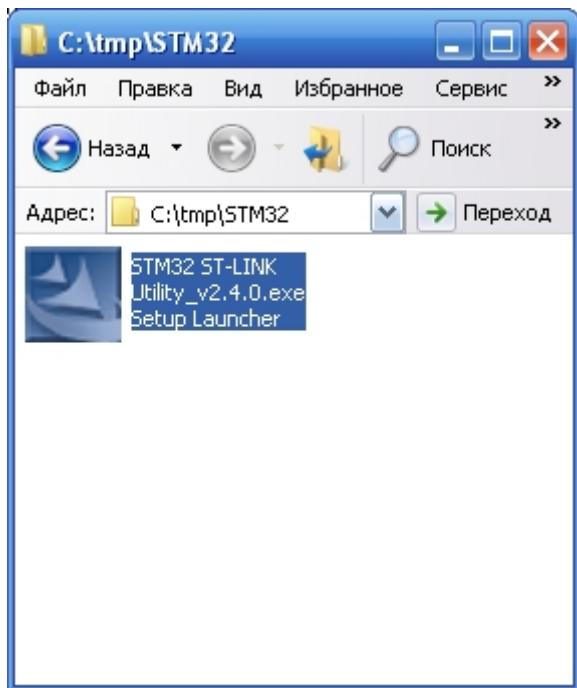
АдAPTER обеспечивает

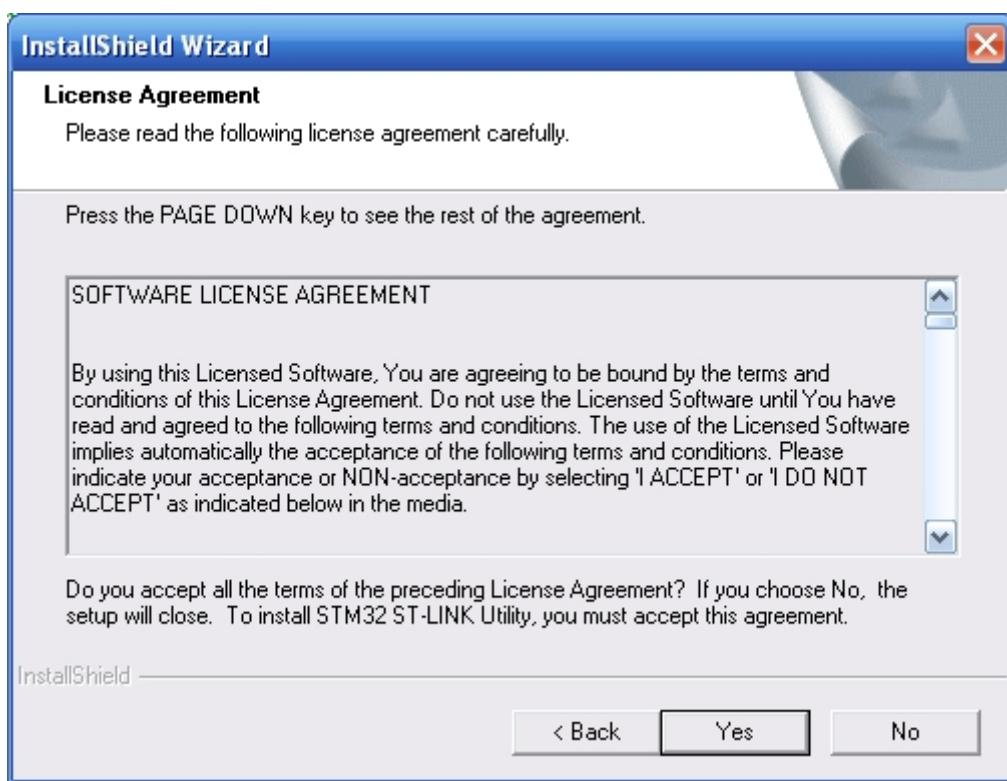
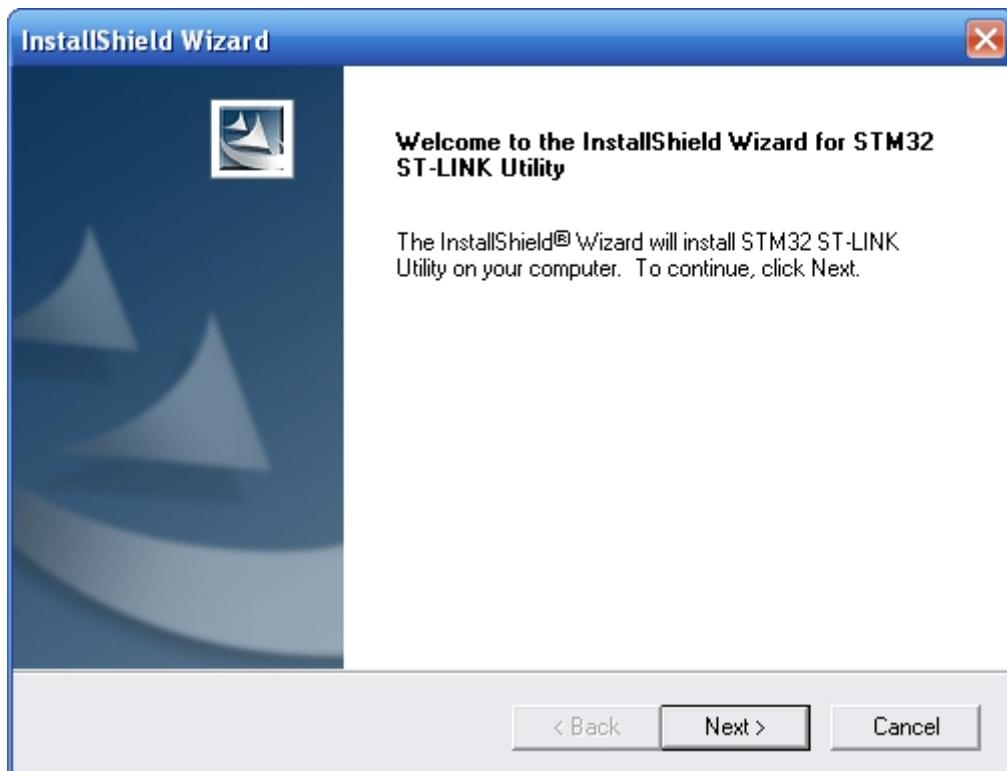
- загрузка прошивки во встроенную Flash (обеспечивается всеми адAPTERами)
- чтение/запись областей памяти
- интерактивную отладку в процессе работы программы ??

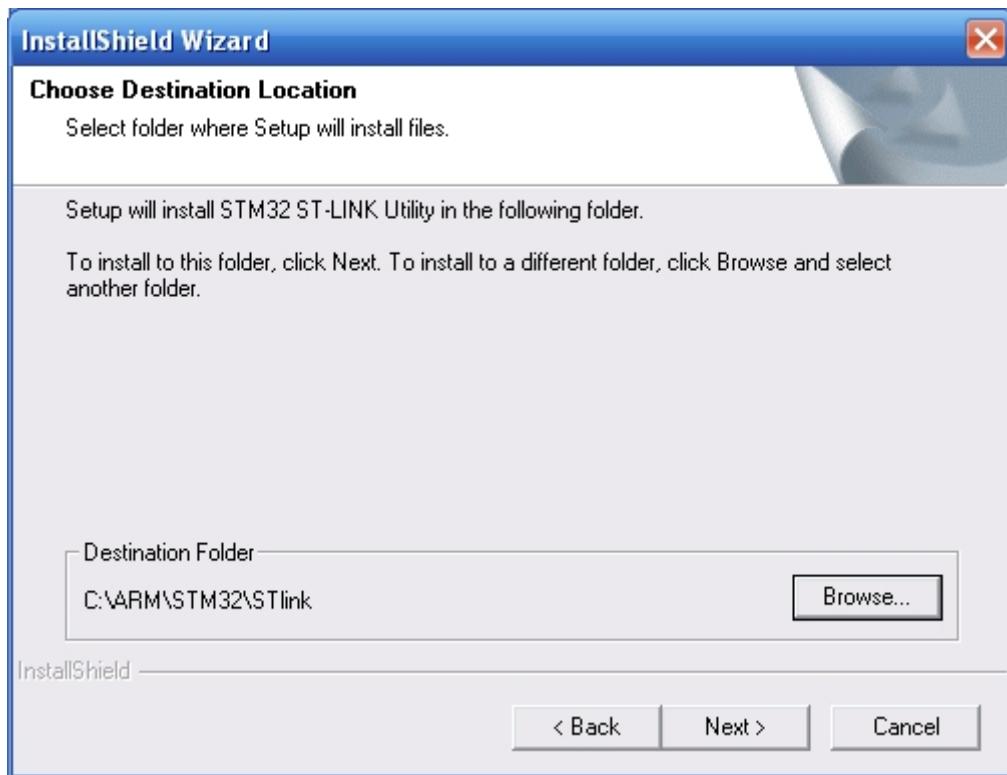
### 10.1 STlink

#### Установка ПО

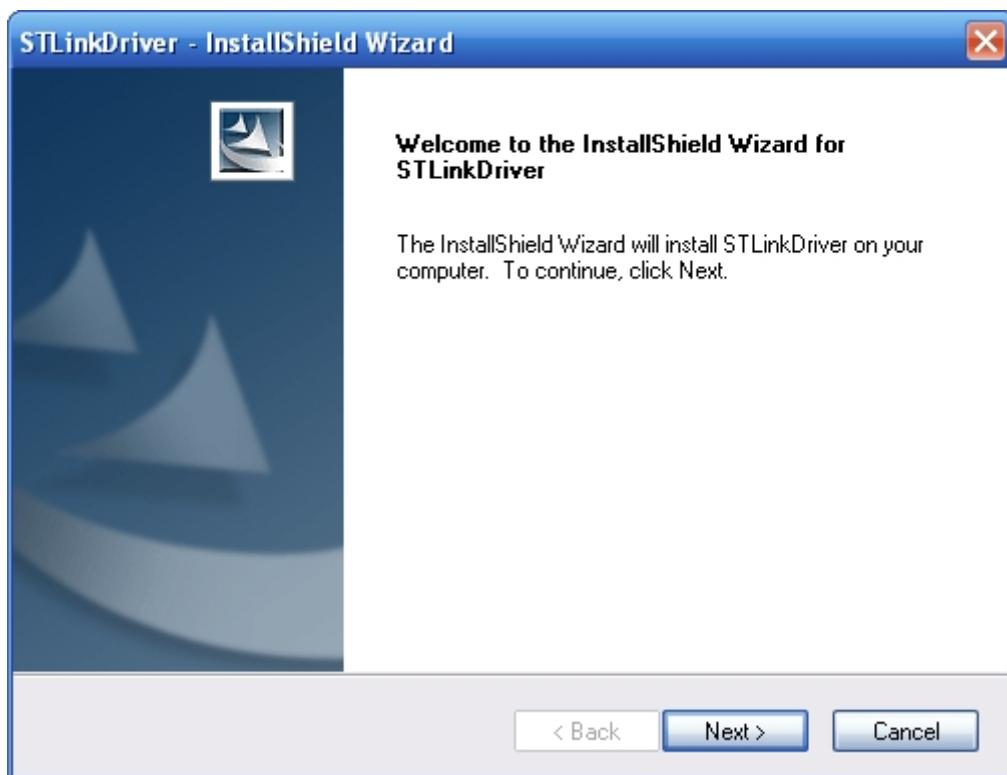
Для работы с чипами STM32 используя программаторы STlink, встроенные на демо-платы Discovery необходимо установить пакет STSW-LINK004 (STM32 ST-link Utility и драйвер), скачав его с <http://www.st.com/web/en/catalog/tools/PF258168>

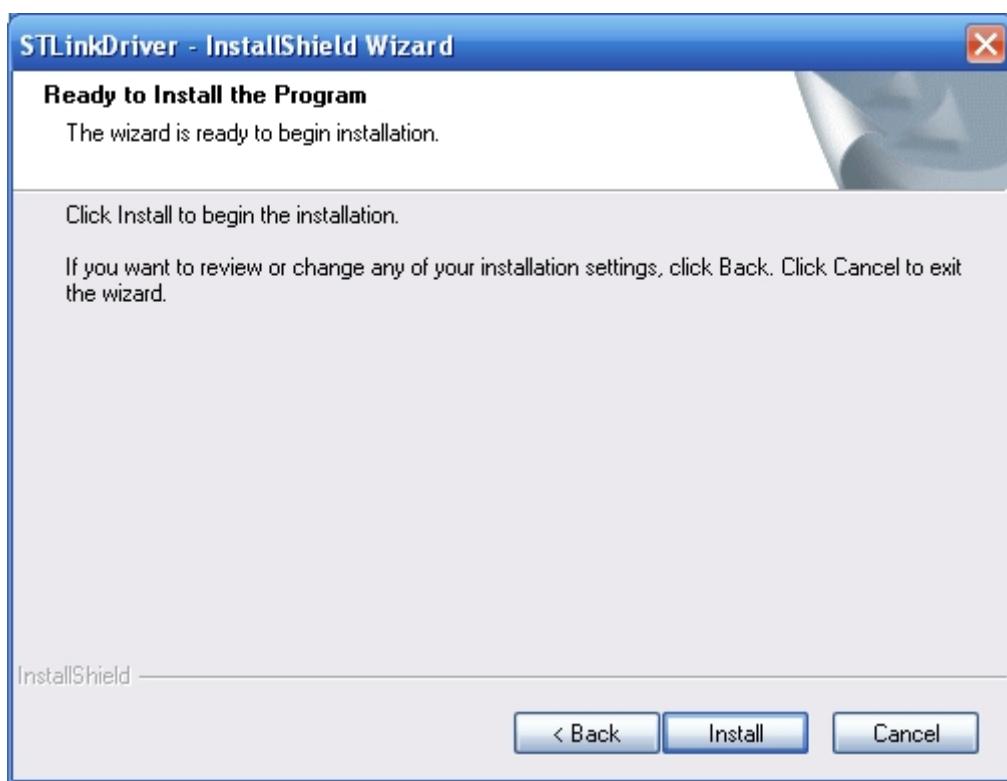
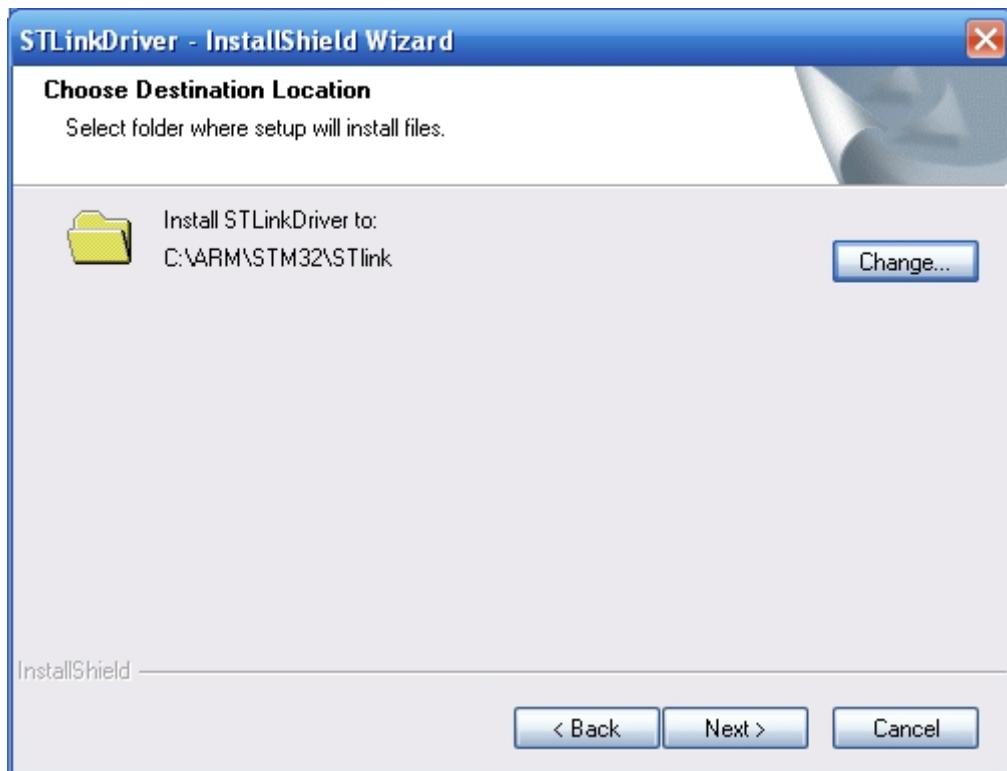


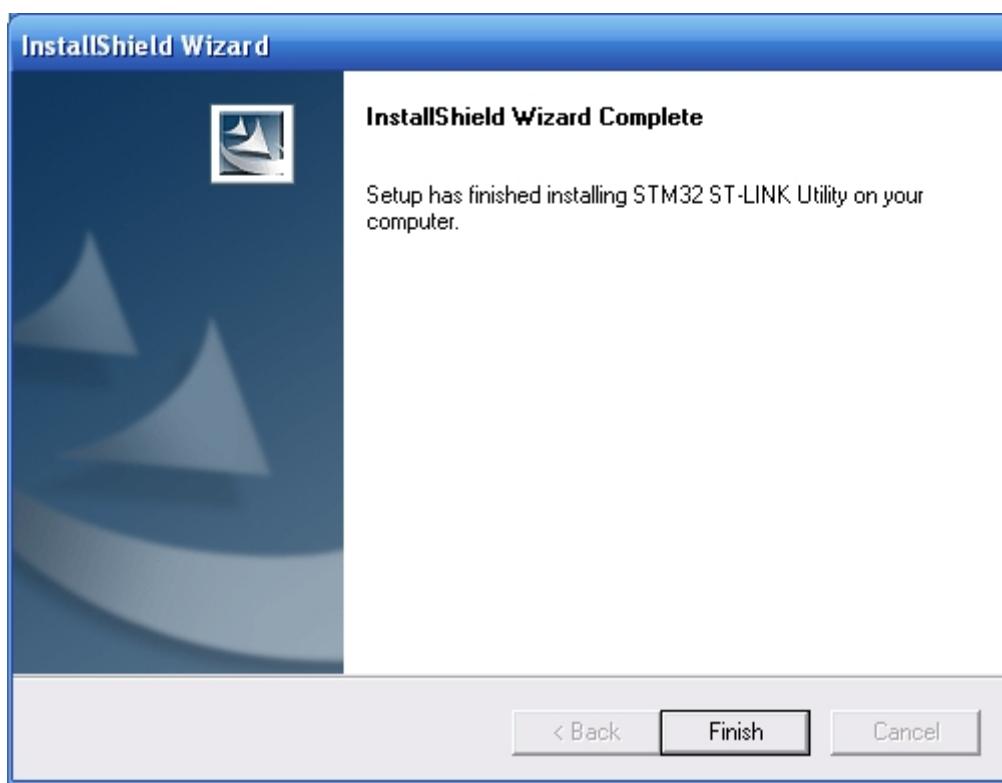
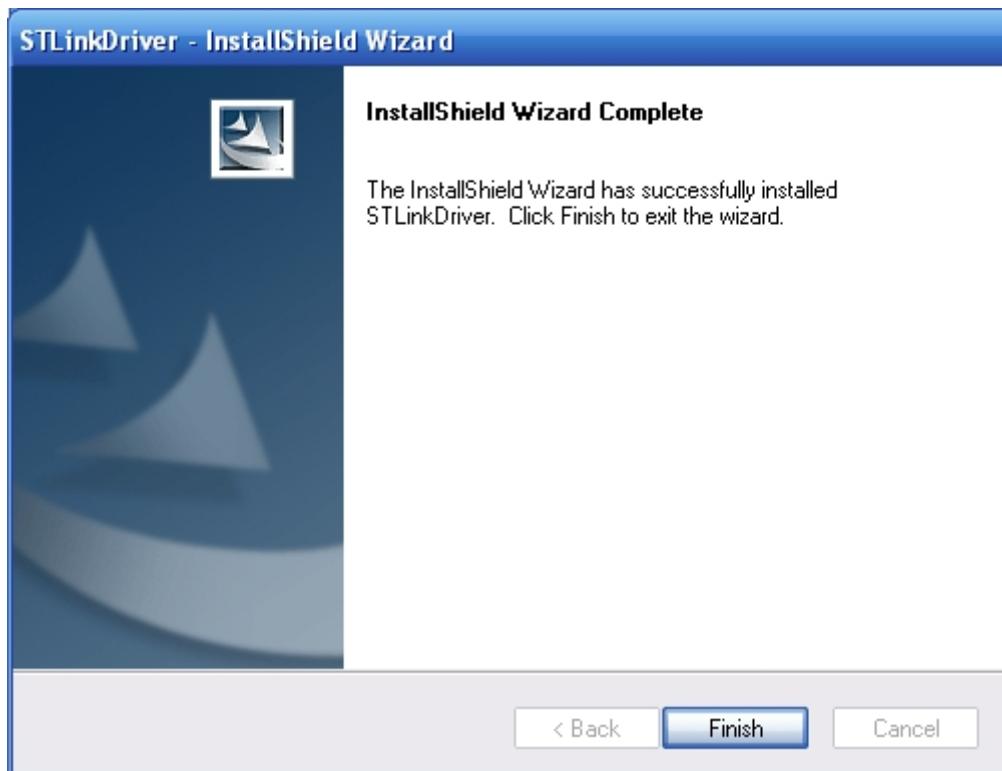




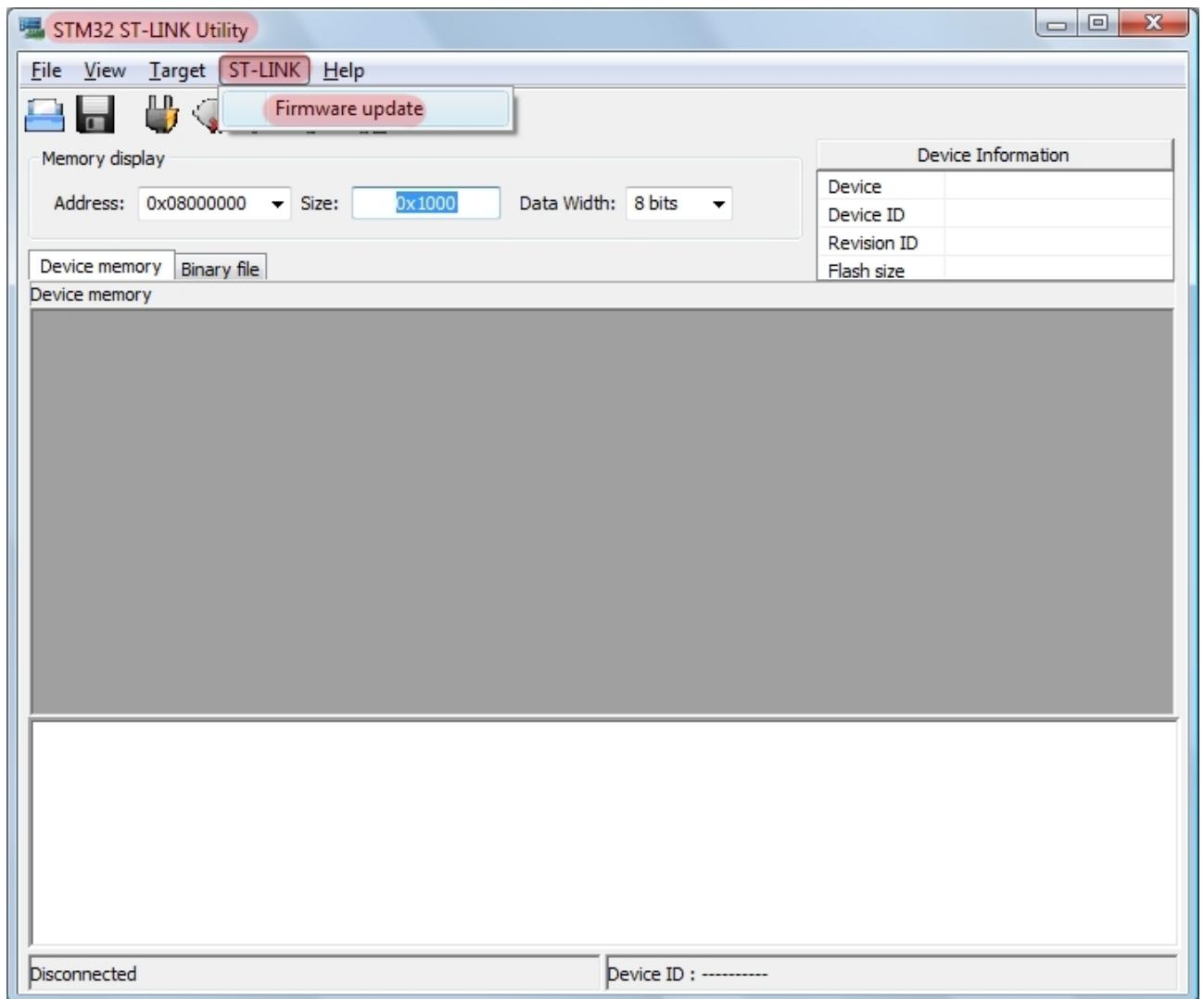
Пакет драйвера входит в пакет STSW-LINK004, и запускается автоматически



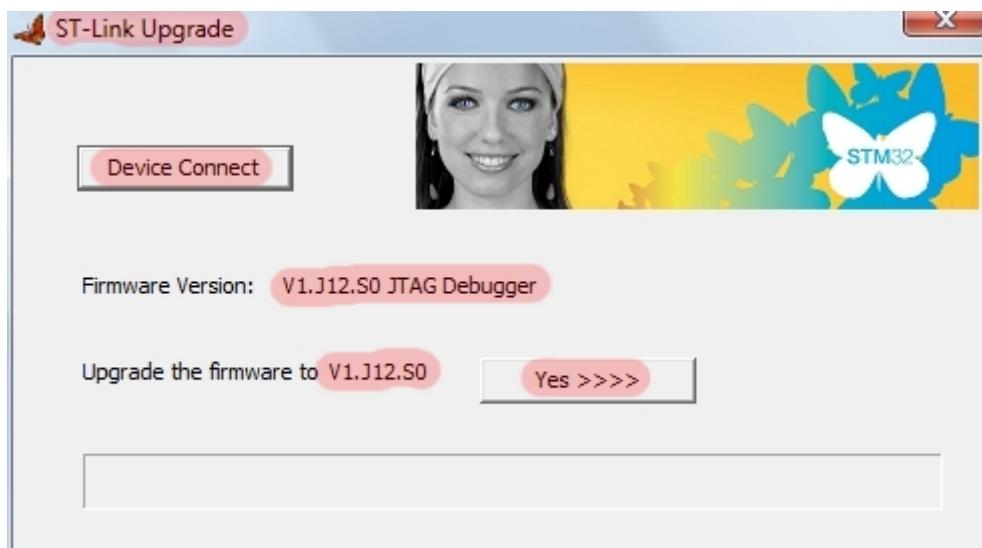




Подключаете отладочную плату, запускаете STlink10.1 Utility [Пуск > Программы > STMicroelectronics > STM32 ST-LINK Utility](#)



Запускаем **ST-LINK** > **Firmware update** > **Device Connect** и проверяем что прошивка STlink **10.1** не устарела (ПК должен быть подключен к Сети), при необходимости нажимаем **Yes** и обновляемся.



## STlink gdbserver

Необходим для отладки через GDB<sup>13</sup>.

### 10.2 JTAG

## Часть V

### Отладка

# Глава 11

## Отладка в Keil

# Глава 12

## STM32 SWD

# Глава 13

## GDB

### 13.1 STlink gdbserver

## Глава 14

### OpenOCD

# Глава 15

## JTAG

# **Часть VI**

## **Основы языка $C^{+^+}$**

# Глава 16

## Синтаксис

# Глава 17

## Типы данных

## Глава 18

### Стандартная библиотека libc

## **Часть VII**

### **CMSIS**

В этом разделе будут описаны только *некоторые* части библиотеки CMSIS**VII** для STM32F, которые используются в коде ARMatura. Объяснить это очень просто — попытавшись включить в книгу полный листинг `stm32f4xx.h` получил 205 страниц листинга 😊

# Глава 19

## Startup: файлы стартового кода

`keil_startup_stm32f4xx.s`

`system_stm32f4xx.h`

`system_stm32f4xx.c`

# Глава 20

## Стандартная библиотека STM32

Для работы необходимо скачать библиотеки поддержки от ST:

STSW-STM32078: STM32VLDISCOVERY1 firmware package (AN3268)

[http://www.st.com/st-web-ui/static/active/en/st\\_prod\\_software\\_internet/resource/technical/software/firmware/stsw-stm32078.zip](http://www.st.com/st-web-ui/static/active/en/st_prod_software_internet/resource/technical/software/firmware/stsw-stm32078.zip)

STSW-STM32068: STM32VLDISCOVERY1 firmware package (AN3268)

[http://www.st.com/st-web-ui/static/active/en/st\\_prod\\_software\\_internet/resource/technical/software/firmware/stsw-stm32068.zip](http://www.st.com/st-web-ui/static/active/en/st_prod_software_internet/resource/technical/software/firmware/stsw-stm32068.zip)

STSW-STM32065: STM32F4 DSP and standard peripherals library, including 82 examples

[http://www.st.com/st-web-ui/static/active/en/st\\_prod\\_software\\_internet/resource/technical/software/firmware/stm32f4\\_dsp\\_stdperiph\\_lib.zip](http://www.st.com/st-web-ui/static/active/en/st_prod_software_internet/resource/technical/software/firmware/stm32f4_dsp_stdperiph_lib.zip)

STSW-STM32078 STM32VLDISCOVERY1 firmware package (AN3268)

STSW-STM32068 STM32F4DISCOVERY2 board firmware package, including 22 examples

STSW-STM32065 STM32F4 DSP and standard peripherals library, including 82 examples

### 20.1 stm32f4xx.h

## Глава 21

### USB client/host

# **Часть VIII**

## **Ядро Cortex-Mx**

## Глава 22

### Режимы ARM и Thumb

# Глава 23

## DMA

## Глава 24

DSP /Cortex-M3/

## Глава 25

FPU /Cortex-M4F/

# **Часть IX**

## **Интерфейсы**

# Глава 26

## USB

# Глава 27

## UART

Глава 28

SPI

# Глава 29

I2C

# Глава 30

## CAN

# **Часть X**

## **Операционные системы ОСРВ**

# Глава 31

## Keil RTX

## Глава 32

### FreeRTOS

# Глава 33

eCos

# Глава 34

## Linux

подробно рассмотрен в отдельном разделе XIII

# **Часть XI**

## **Стек TCP/IP**

## Глава 35

### Ethernet

# Глава 36

PPP

## **Часть XII**

### **Типовые применения**

# Глава 37

## GPS

### 37.1 Протокол NMEA 0183

NMEA 0183— текстовый протокол связи морского и навигационного оборудования.

[http://www8.garmin.com/support/pdf/NMEA\\_0183.pdf](http://www8.garmin.com/support/pdf/NMEA_0183.pdf)  
[http://www8.garmin.com/support/pdf/NMEA\\_0183.pdf](http://www8.garmin.com/support/pdf/NMEA_0183.pdf)  
[http://ru.wikipedia.org/wiki/NMEA\\_0183](http://ru.wikipedia.org/wiki/NMEA_0183)  
<http://www.robosoft.info/ru/technologies/knowledgebase/nmea0183>

Датум *WGS<sup>84</sup>*

[http://ru.wikipedia.org/wiki/WGS\\_84](http://ru.wikipedia.org/wiki/WGS_84)

Большинство GPS приемников отдают данные по NMEA 0183, но вместо режима последовательного порта 4800 8N1 прописанного в протоколе, могут использовать другие режимы, характерные для портов RS232 (9600, 115200). Координаты передаются в датуме *WGS<sup>84</sup>*.

Формат сообщения NMEA<sup>1</sup>:

\\$[A-Z]5(,<data>)+\\*[0-9A-F]2<CR><LF>

- символ \$
- 5-буквенный идентификатор сообщения. Первые две буквы — идентификатор источника сообщения, следующие три буквы — идентификатор формата сообщения
  - GPGGA — данные о последнем определении местоположения
  - GPGLL — координаты, широта/долгота
  - GPGSA — DOP (GPS) и активные спутники
  - GPGSV — наблюдаемые спутники
  - GPWPL — параметры заданной точки
  - GPBOD — азимут одной точки относительно другой
  - GPRMB — рекомендуемый минимум навигационных данных для достижения заданной точки

---

<sup>1</sup>регулярные выражения [http://en.wikipedia.org/wiki/Regular\\_expression](http://en.wikipedia.org/wiki/Regular_expression)

- GPRMC — рекомендуемый минимум навигационных данных: информацию о времени, местоположении (*WGS<sup>84</sup>*), курсе и скорости, передаваемые навигационным GPS приёмником. Контрольная сумма обязательна для этого сообщения, интервалы передачи не должны превышать 2 секунды.
- GPRTE — маршруты
- HCHDG — данные от компаса
- блоки данных, разделённых запятыми. Пустые данные, не заданные в середине строки, оставляют запятыю. Пустые данные в конце строки могут отбрасываться вместе с запятой.
- символ \*
- двузначное hex число — контрольная XOR-сумма всех байт в строке между \$ и \*.
- конец строки <CR><LF> = 0x0D 0x0A = \r\n

## 37.2 \$GPRMC — рекомендуемый минимум навигационных данных

\$GPRMC, hhmmss.ss, [AV], GGMM.MM, [NS], gggmm.mm, [EW], v.v, b.b, ddmmyy, x.x, n, m\*hh<CR><LF>

- GP — приём сигналов GPS, GN — ГЛОНАСС
- RMC — Recommended Minimum sentence C
- hhmmss.ss — время фиксации местоположения по времени UTC
- A — данные достоверны, V — данные недостоверны
- GGMM.MM — широта, 2 цифры градусов, 2 цифры целых минут, точка и дробная часть минут;
- N/S — северная/южная широта
- gggmm.mm — долгота, 3 цифры градусов, 2 цифры целых минут, точка и дробная часть минут;
- E/W — восточная/западная долгота
- v.v — горизонтальная составляющая скорости в узлах
- b.b — путевой угол (направление скорости) в градусах: 0° север, 90° восток, 180° юг, 270° запад
- ddmmyy — дата
- x.x — магнитное склонение в градусах (часто отсутствует)
- n — направление магнитного склонения: для получения магнитного курса магнитное склонение необходимо вычесть (E) или прибавить (W) к истинному курсу
- m — индикатор режима: A — автономный, D — дифференциальный, E — аппроксимация, N — недостоверные данные (часто отсутствует, данное поле включая запятую отсутствует в старых версиях NMEA)

### 37.3 Системы координат (датум)

<http://ne-grusti.narod.ru/Glossary/datums.html>

Системы координат (datums) можно разделить на геоцентрические и топоцентрические.

В геоцентрической системе размеры эллипсоида, ориентация и положение его центра выбираются следующим образом:

- объем эллипсоида предполагается равным объему геоида;
- большая полуось эллипсоида лежит в плоскости экватора геоида;
- малая полуось направлена по оси вращения Земли;
- среднеквадратичное отклонение поверхности эллипсоида от поверхности геоида
- минимально по всей территории земного шара.

*WGS<sup>72</sup>* и сменившая ее *WGS<sup>84</sup>*, а также российская *SGS<sup>85</sup>* являются геоцентрическими системами координат на эллипсоидах *WGS72*, *GRS80* и *SGS85* соответственно. В системе GPS/NAVSTAR используется *WGS<sup>84</sup>*, а в системе GLONASS — *SGS<sup>85</sup>*.

Топоцентрическая (национальная) система координат появляется так: вы берете некоторый эллипсоид и располагаете его таким образом, чтобы для заданной территории среднеквадратичное отклонение поверхности эллипсоида от поверхности геоида было минимальным. При этом остальная часть мира вас не интересует: отклонения на другой стороне Земли может быть сколь угодно велико.

В России используются несколько геодезических систем координат: Пулково<sup>1942</sup> Пулково 1942 г., 1963 г. и 1991 г. Система координат 1963 г. используется военными и ее параметры преобразования засекречены. Обычно мы пользуемся картами, составленными в системе Пулково<sup>1942</sup>. Она базируется на эллипсоиде Красовского.

Параметры преобразования для Пулково<sup>1942</sup>:

#### Преобразование Bursa-Wolf (Position Vector Transformation)

<http://ne-grusti.narod.ru/Glossary/transformations.html#pvt>

направление	dX	dY	dZ	rX	rY	rZ	M	источник
<i>WGS84</i> → 1942	-27.0	+135.0	+84.5	0.0	0.0	0.554	-0.2263	Data+
1942 → <i>WGS84</i>	+25.0	-141.0	-78.5	0.0	0.35	0.736	0.0	Stefan A. Voser

#### Преобразование Молоденского

<http://ne-grusti.narod.ru/Glossary/transformations.html#molodensky>

направление	dX	dY	dZ	da	df	источник
1942 → <i>WGS84</i>	+28.0	-130.0	-95.0	-108.0	+0.00480795	Vladimir Zh. Boston-PC Forum
1942 → <i>WGS84</i>	+28.0	-130.0	-95.0	-108.0	+0.00480795	MADTRAN, Peter H. Dana
1942 → <i>WGS84</i>	+24.0	-123.0	-94.0	-108.0	+0.00480795	Stefan A. Voser

## 37.4 Методы преобразования систем координат

<http://ne-grusti.narod.ru/Glossary/transformations.html>

Часто требуется перейти от одной системы координат (datum) к другой. Например, вы хотите данные с GPS в системе координат  $WGS^{84}$  наложить на карту в системе координат Пулково<sup>1942</sup>.

Произодить преобразование проще (математически), если сначала перевести координаты из географических (широта и долгота) в прямоугольные (XYZ).

Чтобы получить точные прямоугольные координаты точки на поверхности Земли, необходимо знать не только широту и долготу, но и ее высоту над поверхностью эллипсоида. GPS показывает высоту над эллипсоидом  $WGS^{84}$ . Можно вычислить прямоугольные координаты только по широте и долготе, считая, что точка лежит на поверхности эллипса, но при этом точность понизится.

Высоту точки на другими эллипсоядами получить сложнее. Обычно мы знаем высоту в национальной системе высот. За нулевую высоту, как правило, принимается среднее значение уровня моря в определенной точке побережья по результатам многолетних наблюдений. Высоты в этой системе измеряются геодезическими методами относительно поверхности геоида. Поэтому, чтобы узнать высоту точки над эллипсоядом, нужно знать возвышение геоида над эллипсоядом в данном месте, которое трудно вычислить с хорошей точностью. Существуют различные математические модели геоида для разных территорий и для всего земного шара, которые постоянно уточняются.

На заре спутниковой геодезии, когда взаимосвязи между системами координат не были четко определены, а данные исследований были не очень точны, применяли простой сдвиг начала координат  $dX$ ,  $dY$ ,  $dZ$  для перехода от одной системы координат к другой. Это предполагало, что направления осей двух эллипсоядов параллельны (что во многих случаях не соответствует действительности). Для работ на небольшой территории погрешности, вносимые этим предположением, были меньше, чем точность самих данных. Однако, по мере накопления и уточнения данных и повышения точности измерений, стало очевидно, что преобразование по трем параметрам не подходит для больших территорий и глобального использования, если требуется максимальная точность и единый набор параметров преобразования.

Простейший метод — сдвиг центра координат прямоугольной системы, предполагая, что оси исходной и целевой систем координат параллельны.

$$\begin{pmatrix} X_B \\ Y_B \\ Z_B \end{pmatrix} = \begin{pmatrix} X_A \\ Y_A \\ Z_A \end{pmatrix} + \begin{pmatrix} dX \\ dY \\ dZ \end{pmatrix} \quad (37.1)$$

Молоденский разработал формулы для применения параметров сдвига к географическим координатам.

$$\Delta\varphi'' = \frac{206265}{\rho} (-dX \sin \varphi \cos \lambda - dY \sin \varphi \sin \lambda + dZ \cos \varphi + [a\Delta f + f\Delta a] \sin 2\varphi) \quad (37.2)$$

$$\Delta\lambda'' = \frac{206265}{\nabla \cos \varphi} (-dX \sin \lambda + dY \cos \lambda) \quad (37.3)$$

$$\Delta h = dX \cos \varphi \cos \lambda + dY \cos \varphi \sin \lambda + dZ \sin \varphi + (a\Delta f + f\Delta a) \sin^2 \varphi - \Delta a \quad (37.4)$$

здесь  $dX$ ,  $dY$ ,  $dZ$  — сдвиг по осям, м;  $a$  и  $f$  — большая полуось и сжатие исходного эллипса;  $da$  и  $df$  — разности между большой полуосью и сжатием исходного эллипса и целевого эллипса.

Повышенная точность достигается преобразованием Хелмерта с семью параметрами. Есть две его разновидности, различающиеся присвоением знака для параметров поворота.

### 1. Position Vector Transformation (Bursa-Wolf)

$$\begin{pmatrix} X_B \\ Y_B \\ Z_B \end{pmatrix} = M \cdot \begin{pmatrix} 1 & -Rz & +Ry \\ +Rz & 1 & -Rx \\ -Ry & +Rx & 1 \end{pmatrix} \begin{pmatrix} X_A \\ Y_A \\ Z_A \end{pmatrix} + \begin{pmatrix} dX \\ dY \\ dZ \end{pmatrix} \quad (37.5)$$

### 2. Coordinate Frame Transformation

$$\begin{pmatrix} X_B \\ Y_B \\ Z_B \end{pmatrix} = M \cdot \begin{pmatrix} 1 & +Rz & -Ry \\ -Rz & 1 & +Rx \\ +Ry & -Rx & 1 \end{pmatrix} \begin{pmatrix} X_A \\ Y_A \\ Z_A \end{pmatrix} + \begin{pmatrix} dX \\ dY \\ dZ \end{pmatrix} \quad (37.6)$$

Здесь  $M$  — это масштаб (scale), параметры берутся из описания системы координат.

## 37.5 Геоид и эллипсоиды

<http://ne-grusti.narod.ru/Glossary/ellipsoid-geoid.html>

**Геоид** — фигура сложной формы, образованная поверхностью уровня вод Мирового океана, продолженной под материками. Эта поверхность во всех точках перпендикулярна (нормальна) вектору силы тяжести. Отвес направлен перпендикулярно поверхности геоида, а не к центру Земли! Это связано с тем, что плотность Земли распределена неравномерно.

**Эллипсоид** — тело, полученное вращением эллипса вокруг его малой оси. Размеры подбирают так, чтобы среднеквадратичное отклонение от поверхности геоида было минимально либо по всей поверхности Земли, либо для заданной территории.

Параметры некоторых эллипсоидов

эллипсоид	использование	большая полуось a, м	малая полуось b, м	сжатие $f = (a - b)/a$
Красовского (1940)	Россия и др. Пулково <sup>1942</sup>	6378245	6356863	1/298.3
GRS80	международный <i>WGS<sup>84</sup></i>	6378137	6356752.31425	1/298.25722356
SGS85	ГЛОНАСС SGS85			
Бесселя	СССР до 1942 г.			

В таблицах эллипсоидов часто указывается не полярное сжатие  $f$ , а обратная величина  $1/f$ , например, для эллипсоида Красовского  $1/f = 298.3$ .

Отклонения эллипсоида Красовского от геоида на территории СНГ не превышают 150 м.

## 37.6 Tistar15



Цена: 250 руб. <http://www.voltmaster.ru/cgi-bin/qquery.pl?id=127000056703&group=700000>

## 37.7 WISMO228



Цена: 1070 руб. <http://www.voltmaster.ru/cgi-bin/qquery.pl?id=127000881980&group=700000>

# Глава 38

## GSM

### 38.1 WISMO228



Цена: 1070 руб. <http://www.voltmaster.ru/cgi-bin/qwery.pl?id=127000881980&group=700000>

# Глава 39

## шина Dallas 1Wire

39.1 RTC

39.2 Датчики температуры DS18x20

# **Часть XIII**

## **Встраиваемый Linux**

## **Часть XIV**

### **QA**

**Где перевод пунктов меню** Если вы собираетесь заниматься разработкой embedded ПО, знание английского на уровне пользования online словарями и Google Translate обязательно. Кому это не нравится, могут и дальше ждать перевод datasheetов и programmer's manualов на снимаемые с производства компоненты — перевод как раз появляются к этому времени.

# **Часть XV**

## **Приложения**

# Глава 40

## Сводная таблица процессоров

STM32F >> core >> crypto >> model >> pins >> flash >> package >> n

### core

- 1 Cortex-M3
- 3 Cortex-M4
- 4 Cortex-M4F с расширениями FPU (плавающая точка) и SIMD  
для приложений цифровой обработки сигналов

### crypto

- 0 Cortex-M3 только CRC
- 1 Crypto расширение с функционалом вычислений MD5, SHA,..
- 2 Cortex-M4 CRC
- 3 Crypto

### model

- 0
- 3 включена поддержка CAN, USB, SDIO
- 7

### package

- T LQFP

### pins

- C 48
- R 64
- V 100
- I 176

### flash

- 8 64K
- B 128K
- G 1M
- I 2M

	ядро Cortex-	MHz	Flash	SRAM	корпус LQFP	GPIO	ST family
STM32F100C4T6B	M3	24	16K	4K	48		LD
STM32F100RBT	M3	24	128K	8K	100		MD
STM32F103C8T	M3	72	64K	20K	48	37	MD
STM32F103CBT	M3	72	128K	20K	48	37	MD
STM32F103RBT	M3	72	128K	20K	64	51	MD
STM32F103VBT	M3	72	128K	20K	100	80	MD
STM32F407VGT	M4F	168	1M	192K	144		
STM32F407IGT	M4F	168	1M	192K	176		
STM32F427IIT	M4F	168	2M	256K	176		
	CAN	USB	UART	I2C	SPI	ADC	DAC
STM32F100C4T6B	○	○	2		1		
STM32F100RBT	○	○	1				
STM32F103C8T	●	●	3	2	2	2x12b (10ch)	
STM32F103C8T	●	●	3	2	2	2x12b (10ch)	
STM32F103RBT	●	●	3	2	2	2x12b (16ch)	
STM32F103VBT	●	●	3	2	2	2x12b (16ch)	
STM32F407VGT	2	2	6				
STM32F407IGT	2	2	6				
STM32F427IIT	2	2	8				
	таймеры IC/OC/PWM	PWM таймер	DMA	WDT	SysTick	RTC 24b	
STM32F100C4T6B							
STM32F100RBT							
STM32F103C8T	3x16b	1x16b	7ch	2	●	●	
STM32F103C8T	3x16b	1x16b	7ch	2	●	●	
STM32F103RBT	3x16b	1x16b	7ch	2	●	●	
STM32F103VBT	3x16b	1x16b	7ch	2	●	●	
STM32F407VGT					●	●	
STM32F407IGT					●	●	
STM32F427IIT					●	●	