Начало работы с микроконтроллерами STM32WL5x с использованием IAR Embedded Workbench® и MDK-ARM

Вступление

В этом документе приведены рекомендации по использованию программных инструментальных средств IAR Embedded Workbench® for Arm® (EWARM) и MDK-ARM от Keil® с двухъядерными микроконтроллерами (MCU) STM32WL5x.

1 Предпосылки

Перед началом этого руководства необходимо установить следующие инструменты:

- EWARM IDE доступна для загрузки с официального веб-сайта IAR System®.
- MDK-ARM IDE доступна для загрузки с официального веб-сайта Arm® Keil®.
- STM32CubeProgrammer (STM32CubeProg) доступен для загрузки с официального веб-сайта ST
- Пакет MCU STM32CubeWL доступен для загрузки с официального веб-сайта ST
- Сервер ST-LINK доступен для загрузки с официального веб-сайта ST.

Перечисленные ниже документы также могут помочь:

- Справочное руководство STM32WL5x (RM0453)
- STM32WL55xx STM32WL54xx техническое описание (DS13293)
- Технические справочные руководства Cortex®-M4 и M0 + (доступны на веб-сайте Arm®)

Примечание. Arm является зарегистрированным товарным знаком Arm Limited (или ее дочерних компаний) в США и / или в других странах.

2 Архитектура STM32WL5x

Двухъядерные микроконтроллеры STM32WL5x используют архитектуру гетерогенного ядра, которая состоит из ядра Cortex-M4 (с именем CPU1) и ядра Cortex-M0 + (с именем CPU2).

Эти два ядра всегда загружаются отдельно. Двухъядерная отладка позволяет одновременно отлаживать оба ядра с помощью одного аппаратного отладочного зонда.

2.1 Порт доступа

Устройства содержат два порта доступа (AP):

- AP0: порт доступа отладки CPU1 (AHB-AP)
- AP1: порт доступа отладки CPU2 (AHB-AP)

2.2 Загрузка CPU2

CPU2 загружается после того, как CPU1 установил бит C2BOOT в регистре управления мощностью 4 (PWR_CR4). Значение C2BOOT сохраняется в режиме ожидания, и CPU2 загружается соответствующим образом при выходе из режима ожидания. Когда защита включена с помощью байта опции, CPU2 также загружается при обнаружении незаконного доступа. CPU2 может загружаться из системной памяти (RSS / SFI) или из любого места пользовательской флэш-памяти (0x0800 0000), SRAM1 (0x2000 0000) или SRAM2 (0x2000 8000).

2.3 Поддержка отладки

На рисунке ниже показано логическое разделение инфраструктуры отладки с двумя интерфейсами: последовательный провод и порт отладки JTAG.

Рисунок 1. Блок-схема инфраструктуры поддержки отладки.

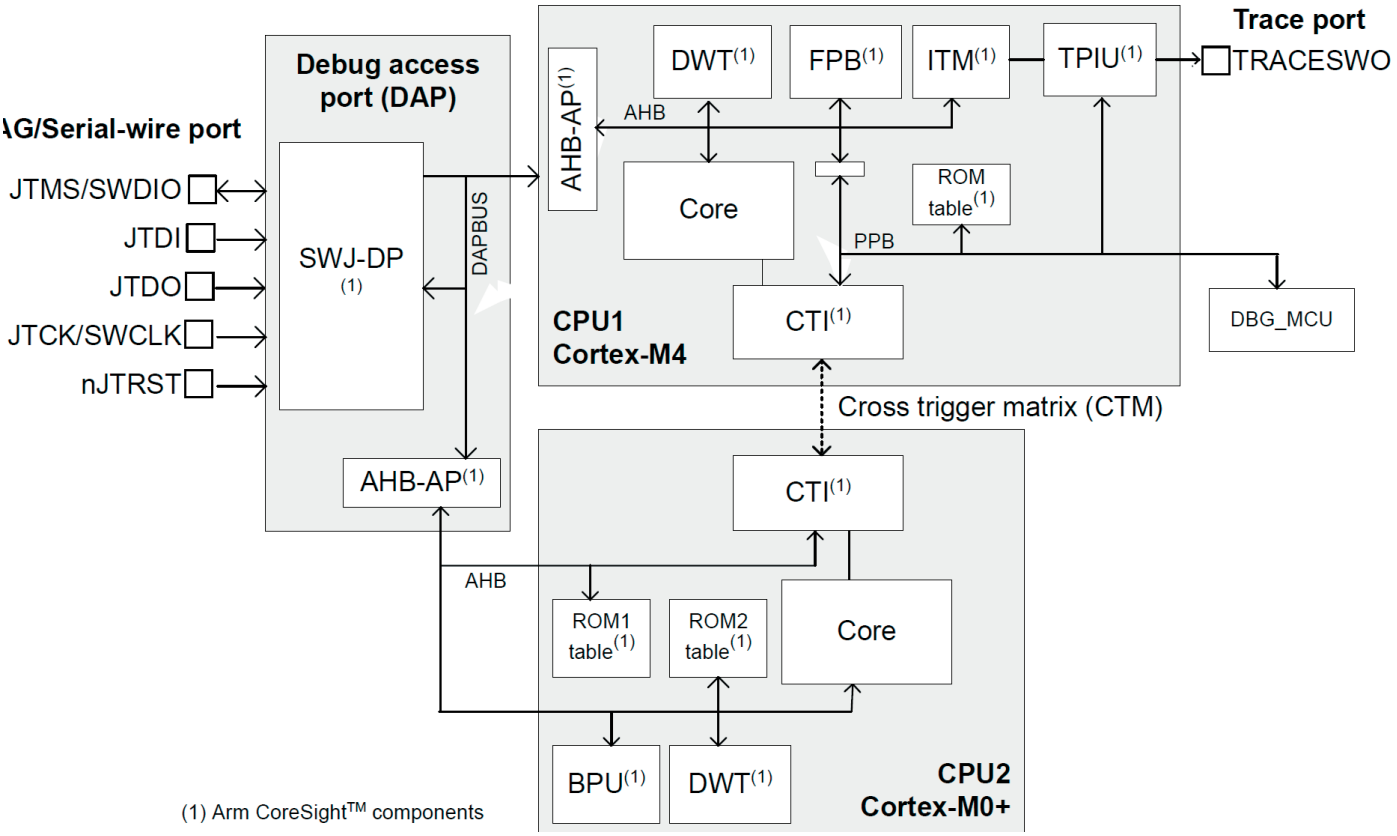


Таблица 1. Функции отладки ядер Cortex-M

CPU1 (Cortex-M4)	CPU2 (Cortex-M0 +)
• Таблица ROM	• Таблица ROM
• Пространство управления системой (SCS)	• Пространство управления системой (SCS)
• Блок точки останова (FPB)	• Блок точки останова (FPB)
• Точка наблюдения и отслеживание данных (DWT)	• Точка наблюдения и отслеживание данных (DWT)
• Макроячейка инструментальной трассировки (ITM)	• Интерфейс перекрестного запуска (CTI)
• Интерфейсный блок порта трассировки (TPIU)	
• Интерфейс перекрестного запуска (CTI)	

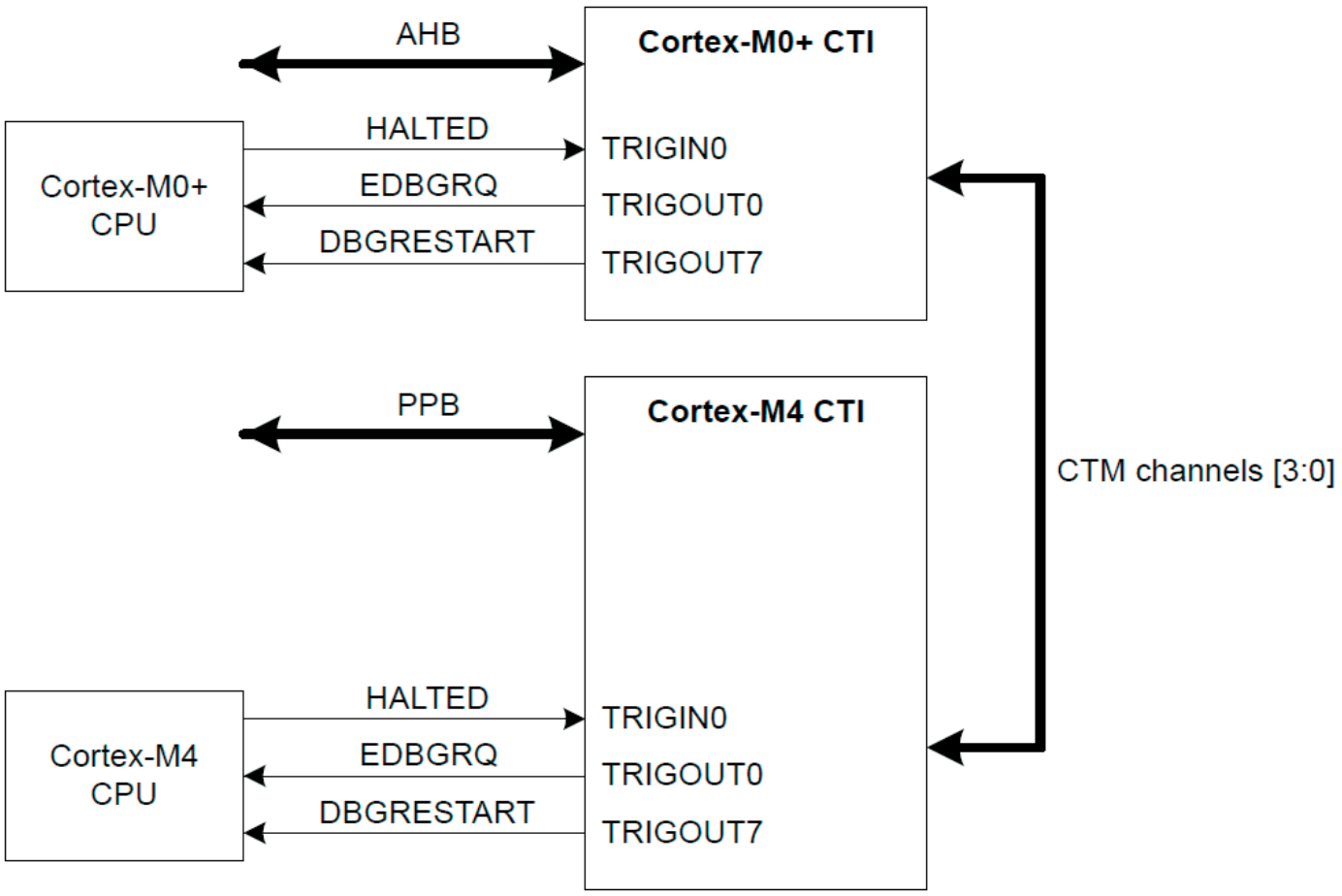
Функции отладки на уровне устройства управляются регистрами DBGMCU, доступными только для CPU1.

2.4 Интерфейс кросс-триггера (CTI)

В STM32WL5x есть два компонента CTI: один предназначен для CPU1, а другой - для CPU2. Эти CTI связаны друг с другом через матрицу перекрестного запуска (CTM), и они позволяют событиям из различных источников запускать отладочную активность.

Например, точка останова, достигнутая в одном из ядер процессора, может остановить другой процессор.

Figure 2. Embedded cross trigger



2.5 Глобальный контроллер безопасности (GTZC)

Устройства STM32WL5x поддерживают безопасность с изоляцией между:

• безопасный мир, управляемый CPU2, где обычно запускаются приложения, чувствительные к безопасности, и расположены критические ресурсы

• небезопасный или общедоступный мир, обрабатываемый ЦП1, в котором используются незащищенные транзакции

2.6 Безопасная конфигурация (биты опций ESE / FSD)

Вся или часть флэш-памяти и памяти SRAM1, SRAM2 может быть сделана защищенной, доступной исключительно для чтения и записи защищенным ведущим устройством: каналы CPU2 и DMA (которые были настроены как защищенные). Защищенный CPU2 может работать только из защищенных областей, тогда как CPU1 может работать только из незащищенных областей.

Примечание. Защита CPU2 включается, когда часть или вся флэш-память защищена (FSD = 0) или когда загрузка байта опции не выполняется. В этом случае установлен бит ESE в FLASH_OPTR.

Измените режим безопасности CPU2

Безопасность CPU2 легко включить, загрузив в пользовательскую опцию FSD значение 0 (безопасность применяется на любом уровне RDP). Защищенный начальный адрес CPU2 может быть изменен защищенным CPU2 путем загрузки новой пользовательской опции SFSA. Чтобы полностью отключить защиту, CPU2 устанавливает бит FSD в 1.

Примечание. Перед снятием защиты с части или всей памяти рекомендуется стереть страницу с той части флэш-памяти, которая становится небезопасной. И незащищенный CPU1, и защищенный CPU2 могут снять защиту, сбросив бит ESE в 0 в FLASH_OPTR и регрессив уровень RDP с уровня 1 на уровень 0. В этом случае регистры резервного копирования основной флэш-памяти (RTC_BKPxR), SRAM1, SRAM2 и PKA SRAM стираются. Безопасная область флэш-памяти Безопасная область флэш-памяти имеет сектор размером 2 Кбайта и определяется пользовательской опцией безопасного начального адреса флэш-памяти, управляемой из SFSA [6: 0] в FLASH_SFR. В том же регистре бит FSD определяет, включена ли защита флэш-памяти. При включении активируется вся система безопасности системы.

Безопасные области SRAM

Области SRAM1 и SRAM2 защищены только тогда, когда включена защита флэш-памяти (ESE = 1 и FSD = 0).

Защищенные области SRAM1 и SRAM2 имеют гранулярность 1 Кбайт. Резервная SRAM2 может быть сконфигурирована как защищенная с помощью бита BRSD и ее начального адреса из SBRSA [4: 0] в FLASG_SRRVR. Не резервная SRAM может быть настроена как защищенная с помощью бита NBRSD и ее начального адреса из SNBRSA [4: 0] в FLASG_SRRVR.

Отладочный доступ

Выбор доступа для отладки CPU2 не зависит от безопасности:

• Когда система небезопасна (ESE = 0), CPU1 и CPU2 могут включать и отключать отладочный доступ CPU2 через бит DDS в FLASH_SFR. В этом случае бит C2SWDBGEN недоступен для записи, и его значение по умолчанию - отладка включена. Отладка CPU2 DDS включается / выключается после перезапуска OBL.

• Когда система защищена (ESE = 1), доступ к отладке CPU2 может быть разрешен только защищенным CPU2 через биты DDS и C2SWDBGEN. Отладка CPU2 может быть отключена непосредственно CPU2 через бит DDS и косвенно обоими процессорами при регрессии ESE. Отладка CPU2 включается / выключается после перезапуска OBL.

3 Использование EWARM

EWARM (IAR Embedded Workbench for Arm) по умолчанию устанавливается в папку C: \ Program Files (x86) \ IAR Systems.

Двухъядерные устройства STM32WL5x официально поддерживаются (без патчей ST) на EWARM, начиная с версии 8.50.9.

В этом разделе используется EWARM v8.50.6 с внутренним патчем для STM32WL5x (EWARMv8_STM32WLxx_V4.7) и шаблон проекта из STM32Cube_FW_WL_V1.0.0.

3.1 Двухъядерные шаги отладки (EWARM)

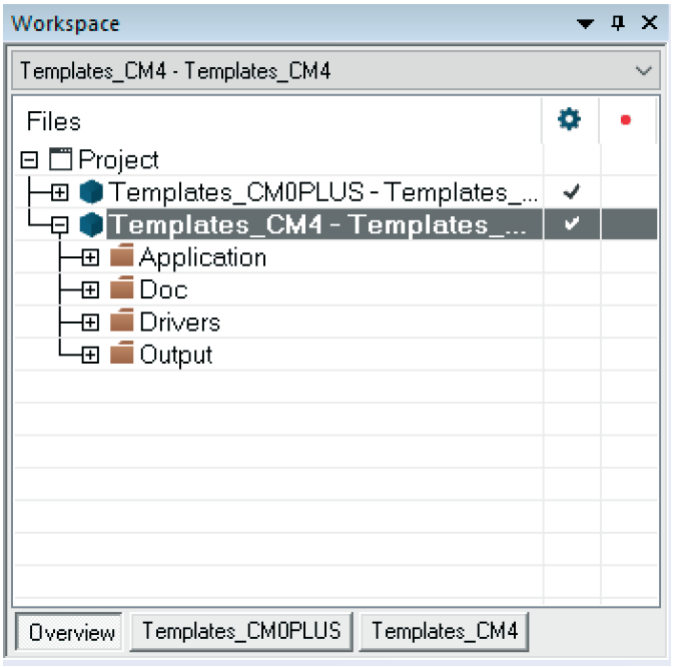
3.1.1 Настройки проекта CM4 (EWARM)

В этом разделе описаны настройки проекта CM4 и используется шаблон проекта из STM32Cube_FW_WL_V1.0.0 с именем Templates_CM4.

1. Откройте проект Template DualCore в
\ STM32Cube_FW_WL_V1.0.0 \ Проекты \ NUCLEO-WL55JC \ Templates \ DualCore \ EWARM.

Этот проект используется для одновременной работы с обоими ядрами. Этот проект теперь отображается в представлении Project Explorer, как показано на рисунке ниже.

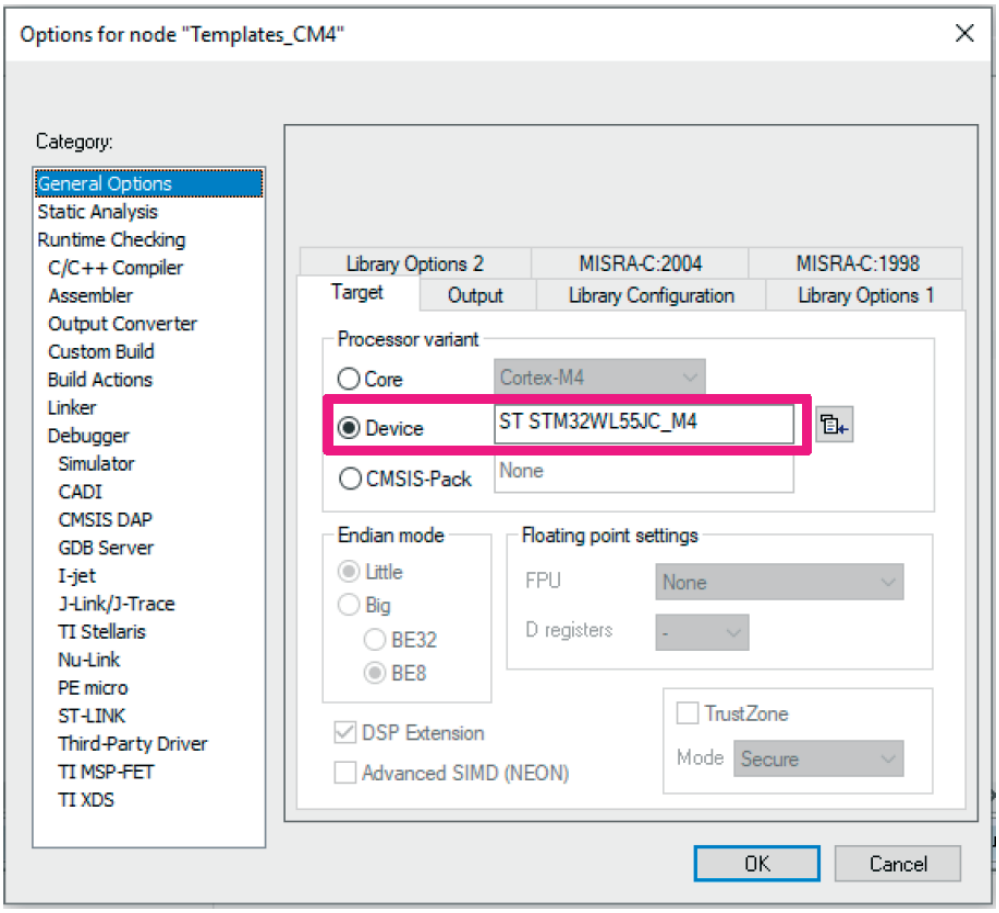
Figure 3. EWARM project explorer view



2. Установите проект Templates_CM4 как активный и убедитесь, что настройки совместимы с опциями проекта ниже.

a. Перейдите в Project > Options. В общих параметрах выберите устройство STM32WL55JC_M4 (см. Рисунок ниже)

Figure 4. STM32WL55JC_M4 device selection



b. Перейдите на вкладку Linker -> Config Linker configuration file editor section. Щелкните Edit, чтобы отобразить редактор файла конфигурации компоновщика. Убедитесь, что приложение связано с правильным адресом: загрузка из основной флэш-памяти по адресу 0x0800 0000 и загрузка из памяти SRAM по адресу 0x2000 0000.

Figure 5. CM4 project - Linker configuration

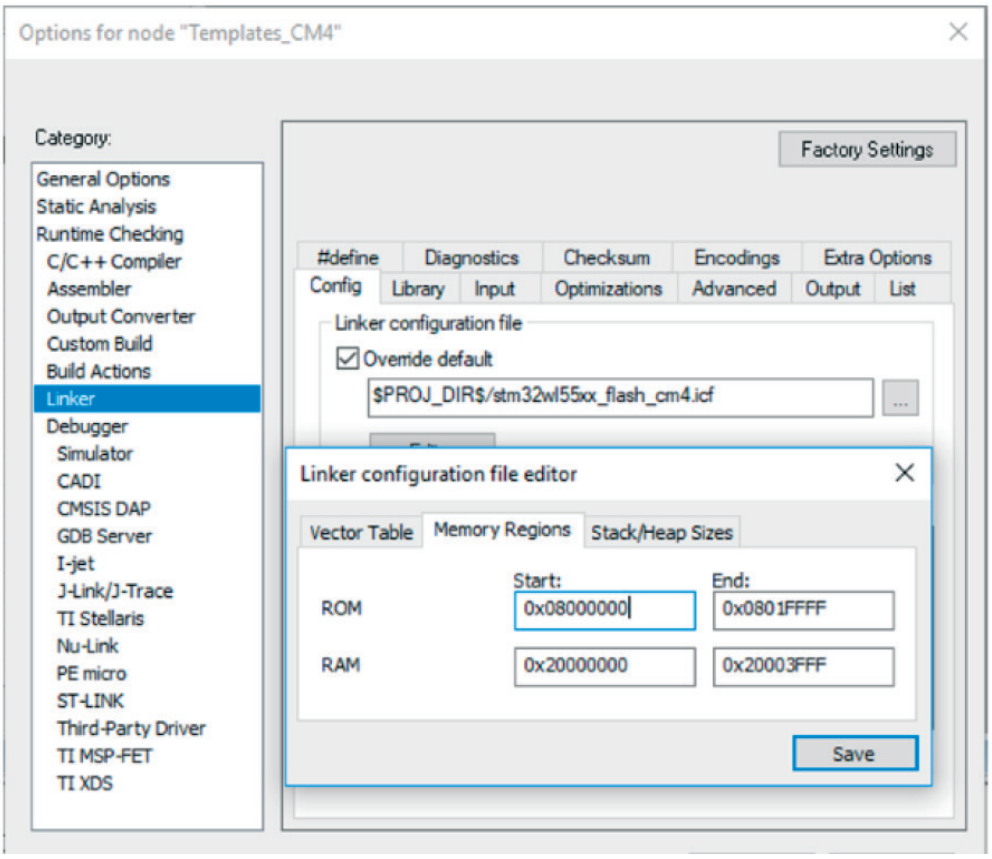
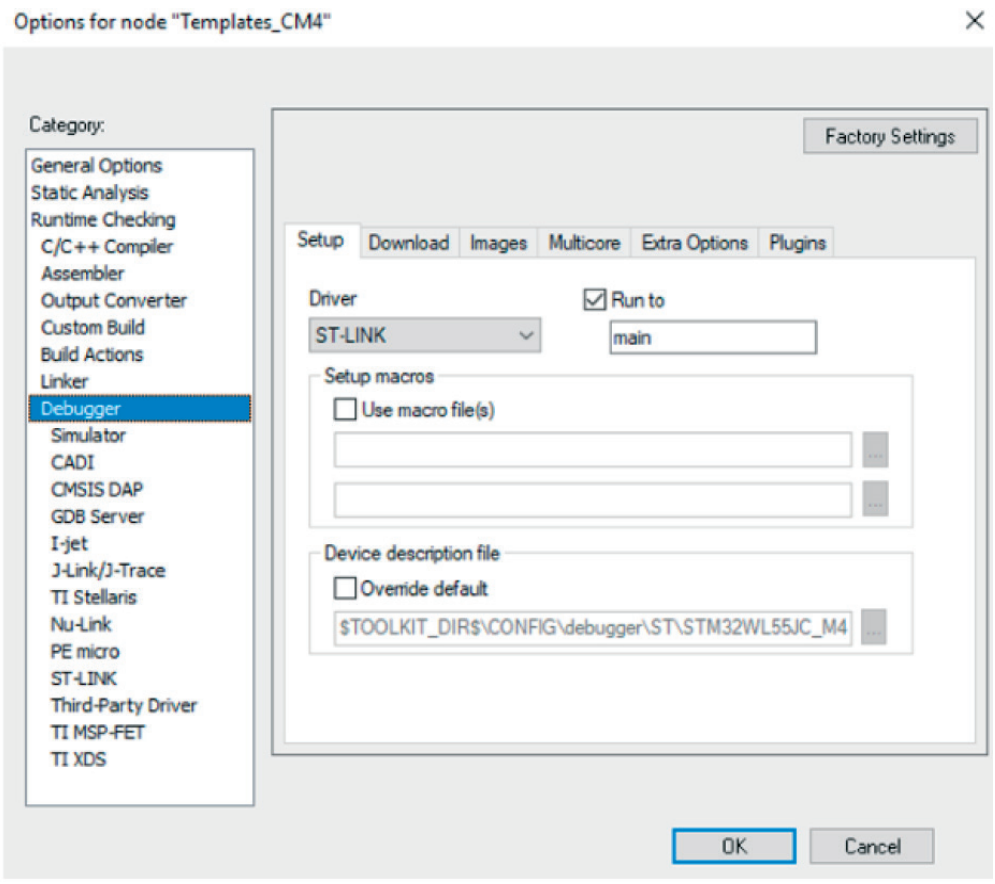
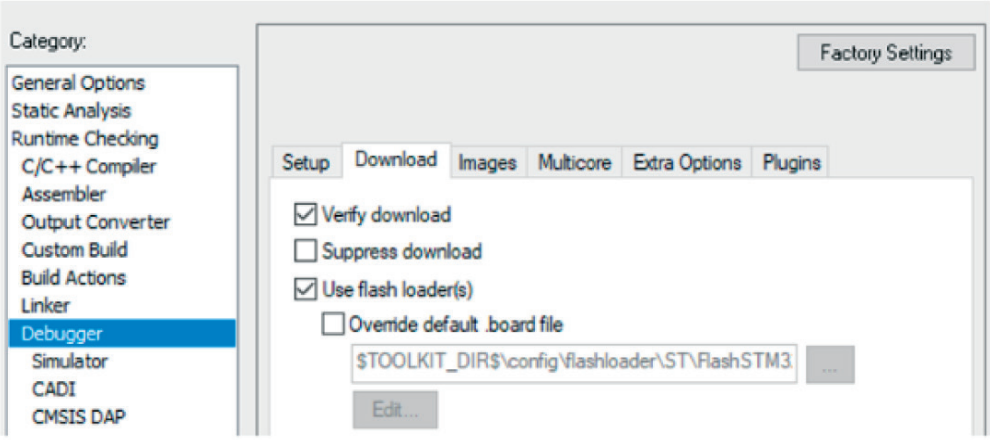


Figure 6. Debugger setup



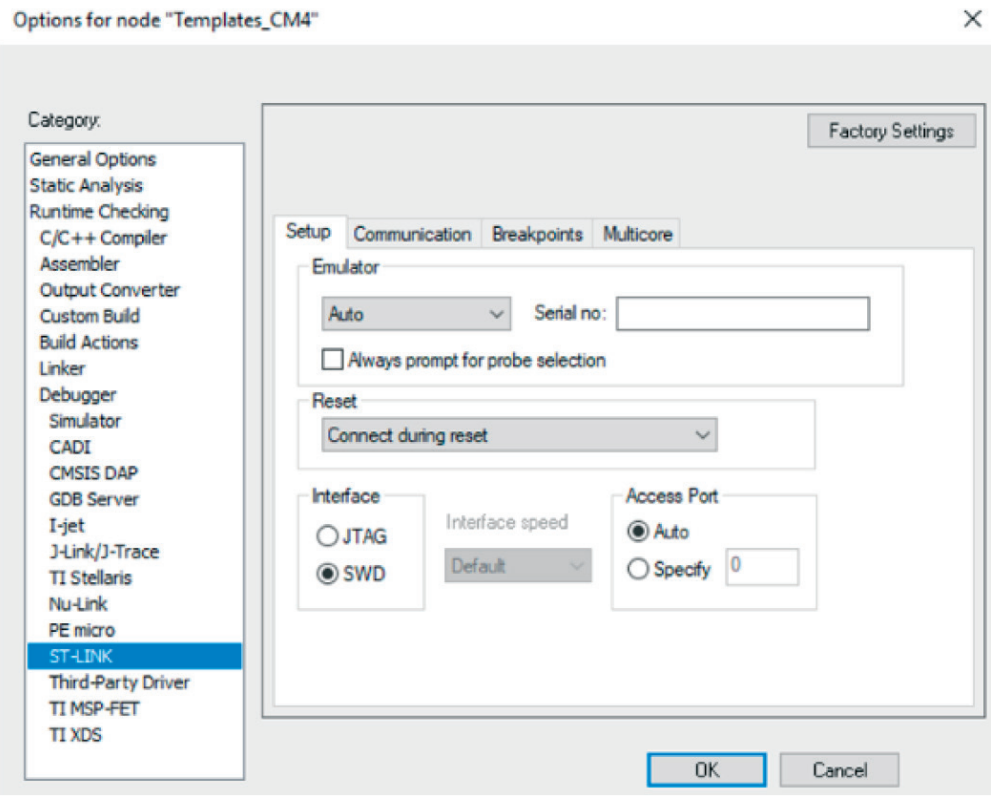
c. На вкладке Debugger выберите ST-LINK в качестве отладчика в поле «Драйвер» (плата NUCLEO - WL55JC содержит встроенный отладчик STLINK-V3).

d. Перейдите на вкладку Debugger -> Download и установите флажок «Использовать загрузчик флэш-памяти».



e. Перейдите в Project -> Options -> ST-LINK -> Setup.

Figure 8. ST-LINK setup



f. Выберите Access Port (порт доступа):

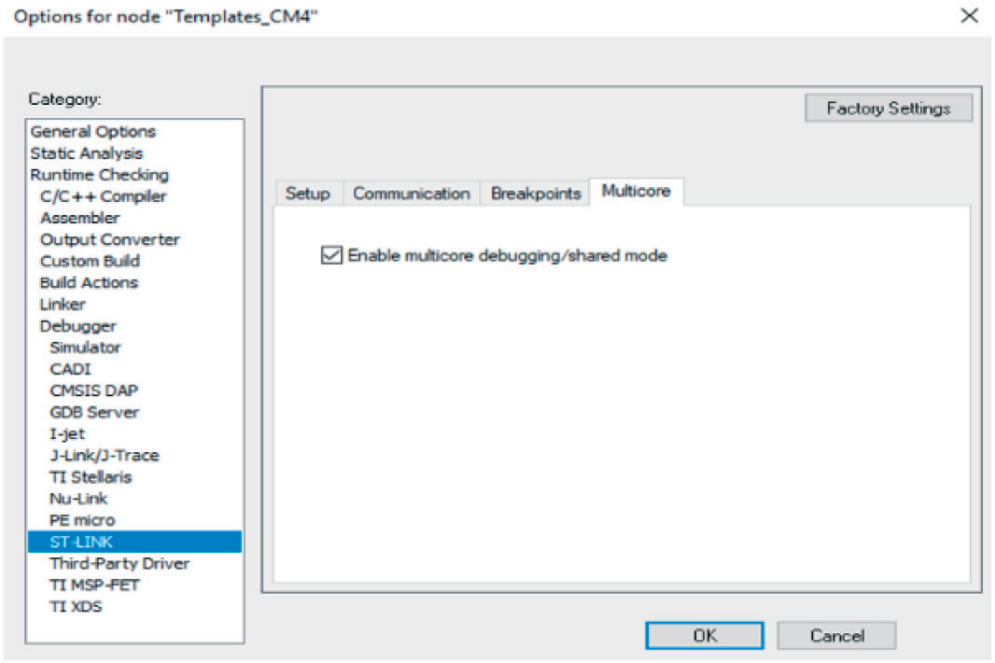
- Авто: порт доступа 0 для Cortex-M4 используется автоматически.
- Вручную: порт доступа можно выбрать вручную (например, поставив 0 с помощью CM4).

g. Выберите SWD в интерфейсе связи, чтобы использовать канал связи последовательного вывода (SWO) (меньше контактов, чем JTAG).

h. Выберите Reset type (тип сброса): Connect during reset (при подключении во время сброса) ST-LINK подключается к цели, сохраняя активным сброс. Сброс выполняется на низком уровне и остается на низком уровне при подключении к цели.

i. Перейдите на вкладку Multicore, чтобы включить использование общего режима для одновременной отладки обоих ядер.

Figure 9. Shared mode activation



Примечание. Чтобы настроить многоядерную отладку с помощью зонда отладки ST-LINK, установите последнюю версию сервера ST-LINK, доступную на www.st.com.

3.1.2 Настройки проекта CM0 + (EWARM)

В этом разделе описаны настройки проекта CM0 + и используется шаблон проекта из STM32Cube_FW_WL_V1.0.0 с именем Templates_CM0PLUS.

Важно:

CPU2 (Cortex-M0 +) загружается после того, как CPU1 (Cortex-M4) установил бит C2BOOT в регистре управления мощностью 4 (PWR_CR4). Это позволяет CPU1 инициализировать систему после сброса или выхода из режима пониженного энергопотребления системы перед загрузкой CPU2.

1. Откройте в другом экземпляре проект Templates_CM0PLUS в \ STM32Cube_FW_WL_V1.0.0 \ Projects \ NUCLEO-WL55JC \ Templates \ DualCore \ EWARM, и убедитесь, что настройки совместимы с параметрами, указанными ниже.

a. Перейдите в Project -> Options. В общих параметрах выберите устройство STM32WL55JC_M0 + (см. Рисунок ниже).

b. Перейдите на вкладку Linker tab -> Config Linker configuration file editor section. Щелкните Изменить, чтобы отобразить редактор файла конфигурации компоновщика. Убедитесь, что приложение было связано с правильным адресом: загрузка из основной флэш-памяти по адресу 0x0802 0000 и загрузка из памяти SRAM по адресу 0x2000 4000

c. На вкладке Debugger tab выберите ST-LINK в качестве отладчика в поле «Драйвер».

d. Перейдите Debugger -> Download tab and tick the Use flash loader(s) check box (вкладка «Скачать» и установите флажок Использовать флеш-загрузчик (и).

e. Перейдите в Project -> Options -> ST-LINK -> Setup.

Figure 10. ST32WL55JC_M0+ device selection

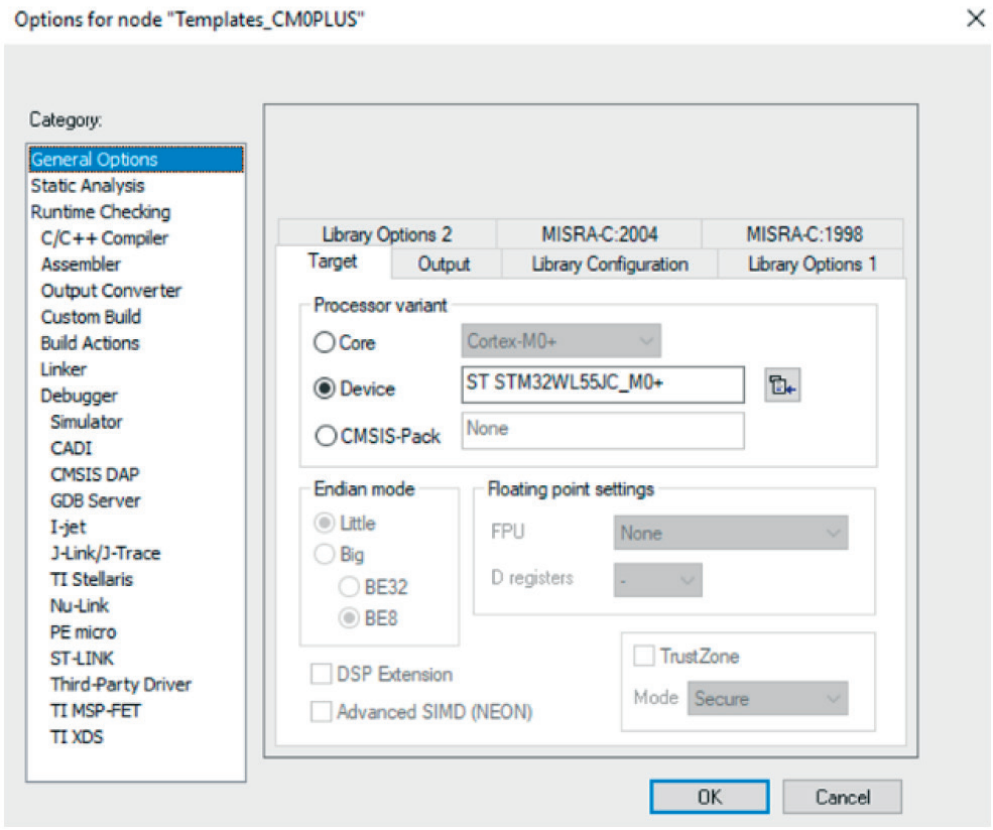


Figure 11. CM0+ project - Linker configuration

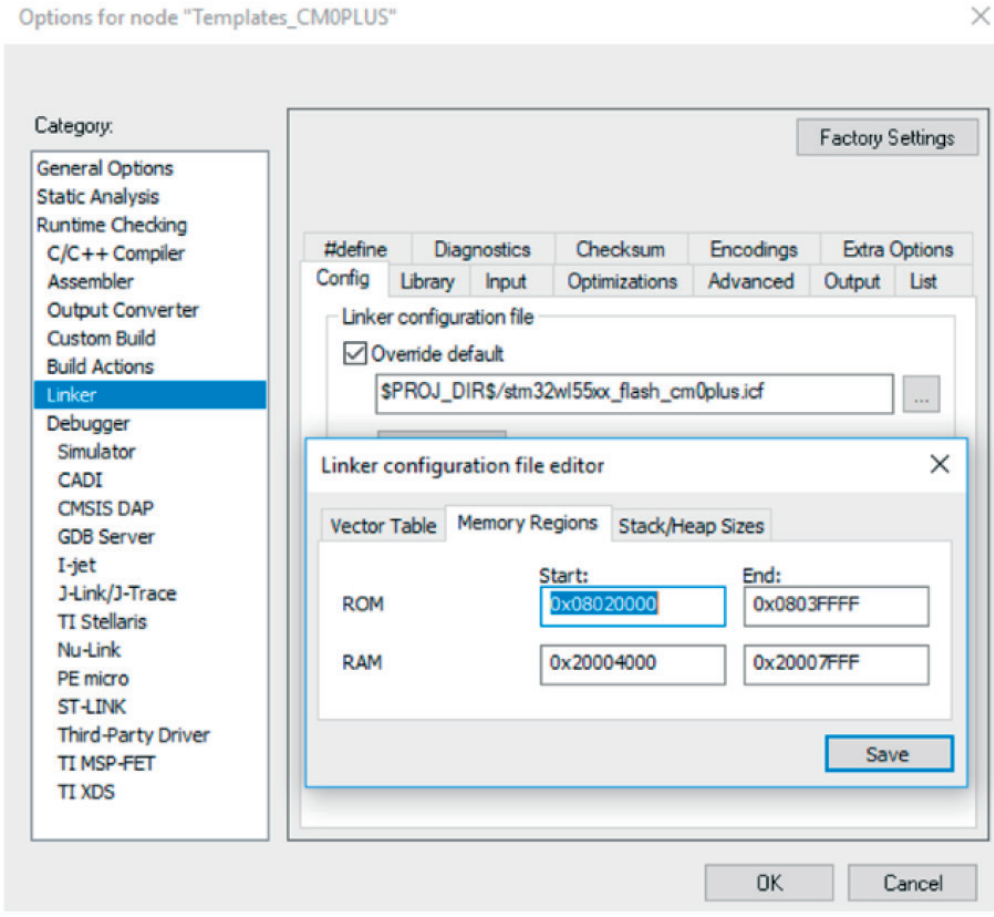
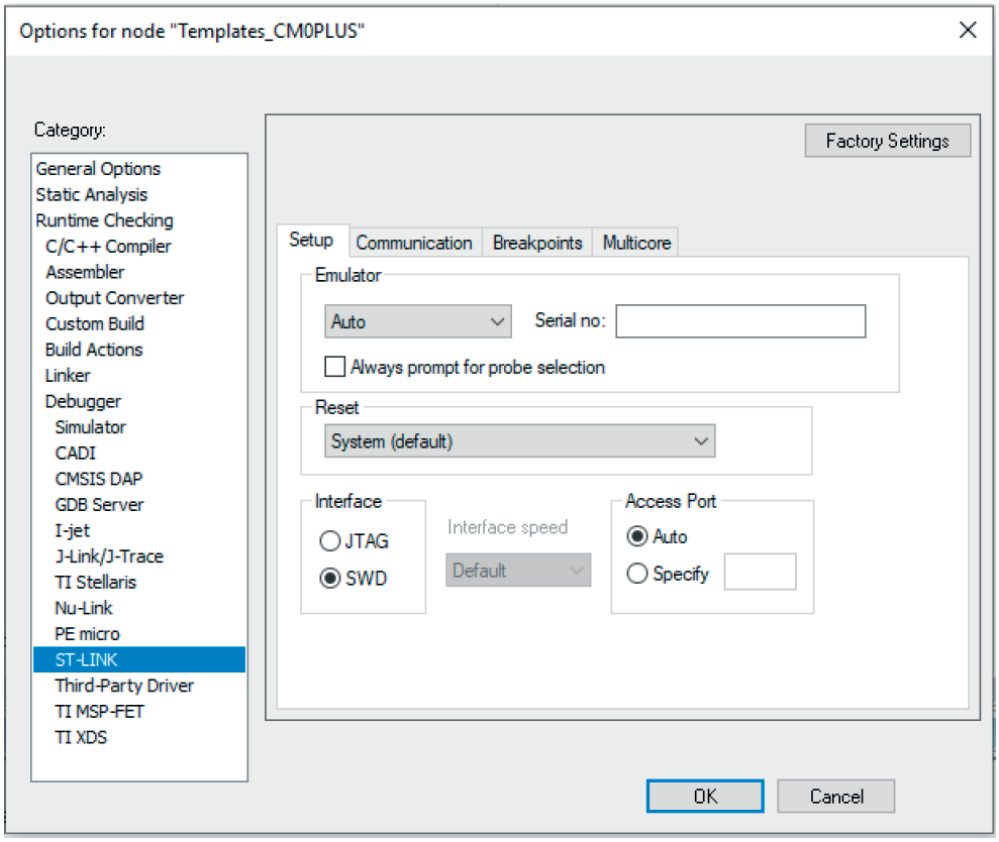


Figure 12. ST-LINK setup



f. Выберите порт доступа:

- Авто: порт доступа 1 для Cortex-M0 + используется автоматически.
- Вручную: можно выбрать порт доступа вручную (например, поставив 1 с CM0 +).

g. Выберите SWD в интерфейсе связи, чтобы использовать канал связи SWO.

h. Измените Reset type тип сброса software to system reset (resets the core and peripherals) с программного на сброс системы (сбрасывает ядро и периферийные устройства).

Примечание. В этом примере по умолчанию используется программный сброс, поскольку он был разработан со старой версией IDE, которая содержит некоторые ограничения со сбросом системы.

i. Перейдите на вкладку Multicore, чтобы включить общий режим, установив флажок Включить многоядерную отладку Enable multicore debugging / shared mode.

3.1.3 Загрузка и отладка обоих проектов (EWARM)

Перед загрузкой проекта подключитесь к плате NUCLEO-WL55JC:

- Подключите инструмент программирования и отладки STLINK-V3E к плате NUCLEO-WL55JC.
- Подключите кабель USB к разъему CN1 USB STLINK на плате.
- LED6 горит красным при подключении ST-LINK.

1. Загрузите проект Templates_CM4 и запустите сеанс отладки, нажав кнопку загрузки и отладки.

Figure 14. Download and Debug button

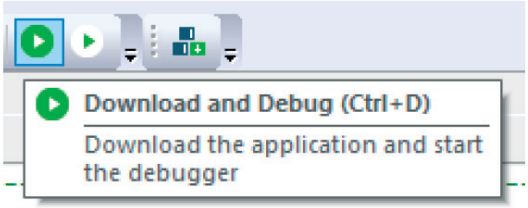


Figure 13. STM32WL55JC NUCLEO board in connected status

Примечание: эта цифра не является контрактной.

2. Выполняйте код до тех пор, пока не появится инструкция, устанавливающая C2BOOT (см. Рисунок ниже). Этот проект загружает Cortex-M0 +, устанавливая бит C2BOOT в PWR_CR4.

Figure 15. Release the CPU2 from holding

```
/* Reset of all peripherals, Initializes the Flash interface and the Systick. */
HAL_Init();

/* USER CODE BEGIN Init */

/* USER CODE END Init */

/* Configure the system clock */
SystemClock_Config();
/* Boot CPU2 */
HAL_PWREx_ReleaseCore(PWR_CORE_CPU2);

/* USER CODE BEGIN SysInit */

/* USER CODE END SysInit */

/* Initialize all configured peripherals */
/* USER CODE BEGIN 2 */

/* USER CODE END 2 */

/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
  /* USER CODE BEGIN 3 */

  /* USER CODE END 3 */
}
  /* USER CODE END WHILE */
```

3.2 Известные ограничения (EWARM)

• При использовании опции сброса программного обеспечения в первый раз для загрузки с Cortex-M0 + (флэш-память пуста) загрузка выполняется успешно, но на протяжении всего сеанса отладки отображается сообщение HardFault.

• Параметр сброса системы не работает в сеансе отладки: при применении сброса системы из сеанса отладки генерируется ошибка HardFault.

4 Использование MDK-ARM

MDK-ARM (от Keil) по умолчанию устанавливается в каталог C: \ Keil: программа установки создает ярлык µVision® 5 в меню «Пуск».

В этом разделе используется MDK-ARM v5.32 с внутренним пакетом для STM32WL5x (Keil.STM32WLxx_DFP.1.0.8) и шаблон проекта из STM32Cube_FW_WL_V1.0.0.

Примечание: все необходимые пакеты доступны для загрузки с официального вебсайта Arm Keil.

4.1 Шаги отладки двухядерных микроконтроллеров (MDK-ARM)

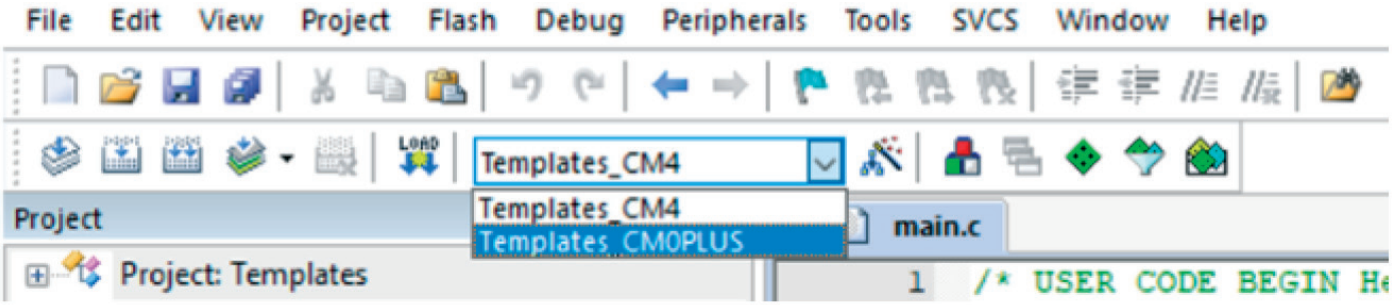
4.1.1 Настройки проекта CM4 (MDK-ARM)

В этом разделе описаны настройки проекта CM4 и используется шаблон проекта из STM32Cube_FW_WL_V1.0.0 с именем Templates_CM4.

1. Откройте проект Template DualCore в
\ STM32Cube_FW_WL_V1.0.0 \ Проекты \ NUCLEO-WL55JC \ Templates \ DualCore \ MDK-ARM.

Этот проект используется для одновременной работы с обоими ядрами. Этот проект теперь отображается в представлении Project Explorer, как показано на рисунке ниже.

Figure 16. Project explorer view



2. Установите проект Templates_CM4 как активный и убедитесь, что настройки совместимы с параметрами, указанными ниже.

a. Выберите правильное устройство, открыв окно конфигурации и выбрав: Project -> Options for Target -> Device устройство STM32WL55JCIx: CM4 из списка.

b. В разделе Project -> Options for Target -> Target -> Read / Only Memory Areas убедитесь, что выбрана правильная область памяти: загрузка из основной флэш-памяти по адресу 0x0800 0000 и загрузка из памяти SRAM1 по адресу 0x2000 0000.

c. Выберите ST-LINK Debugger в качестве отладчика из Project -> Options for Target -> Debug.

Плата NUCLEO-WL55JC содержит встроенный отладчик ST-LINK V3.

Figure 17. STM32WL55JCIx - CM4 device selection

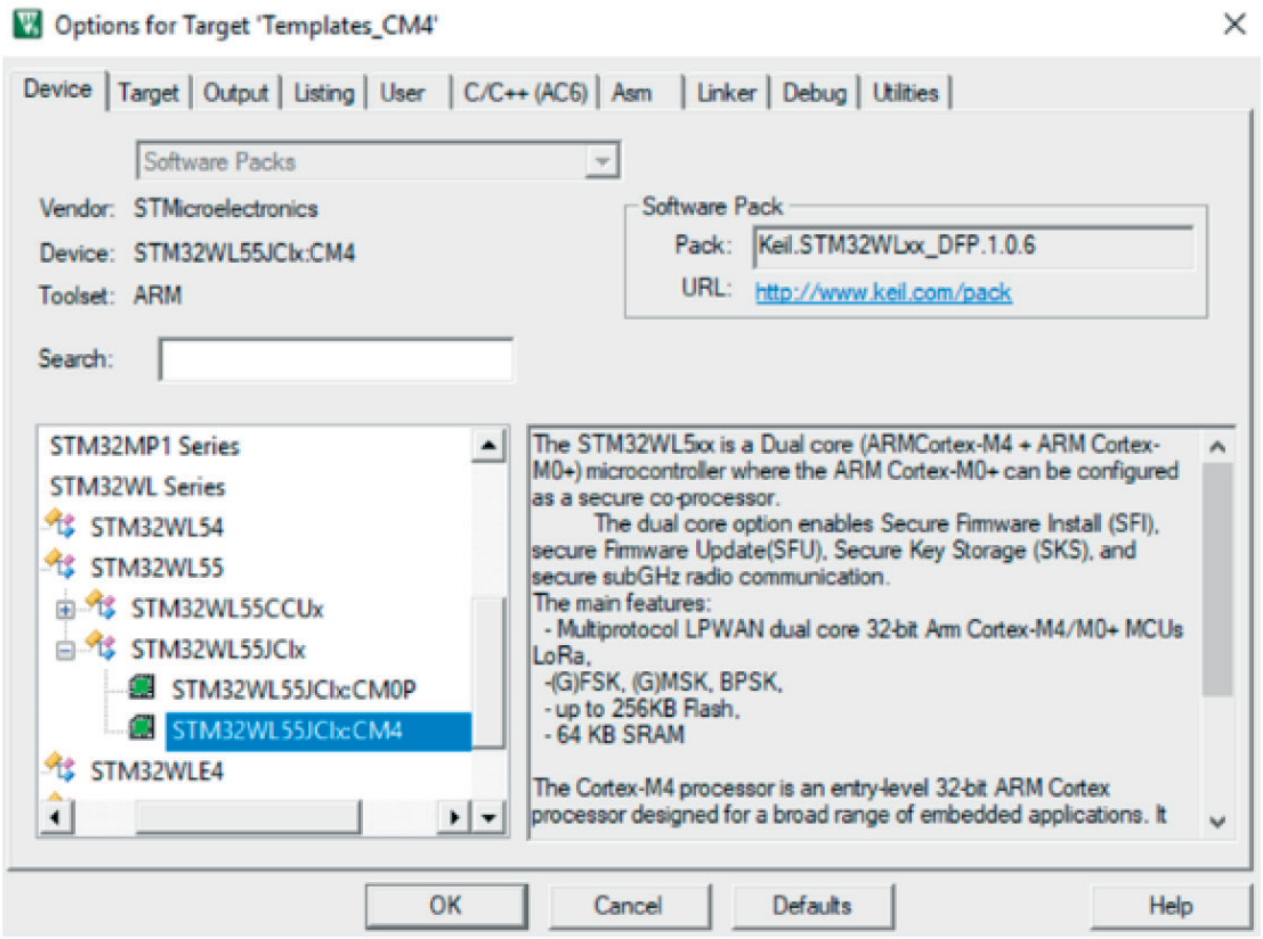


Figure 18. Memory area selection

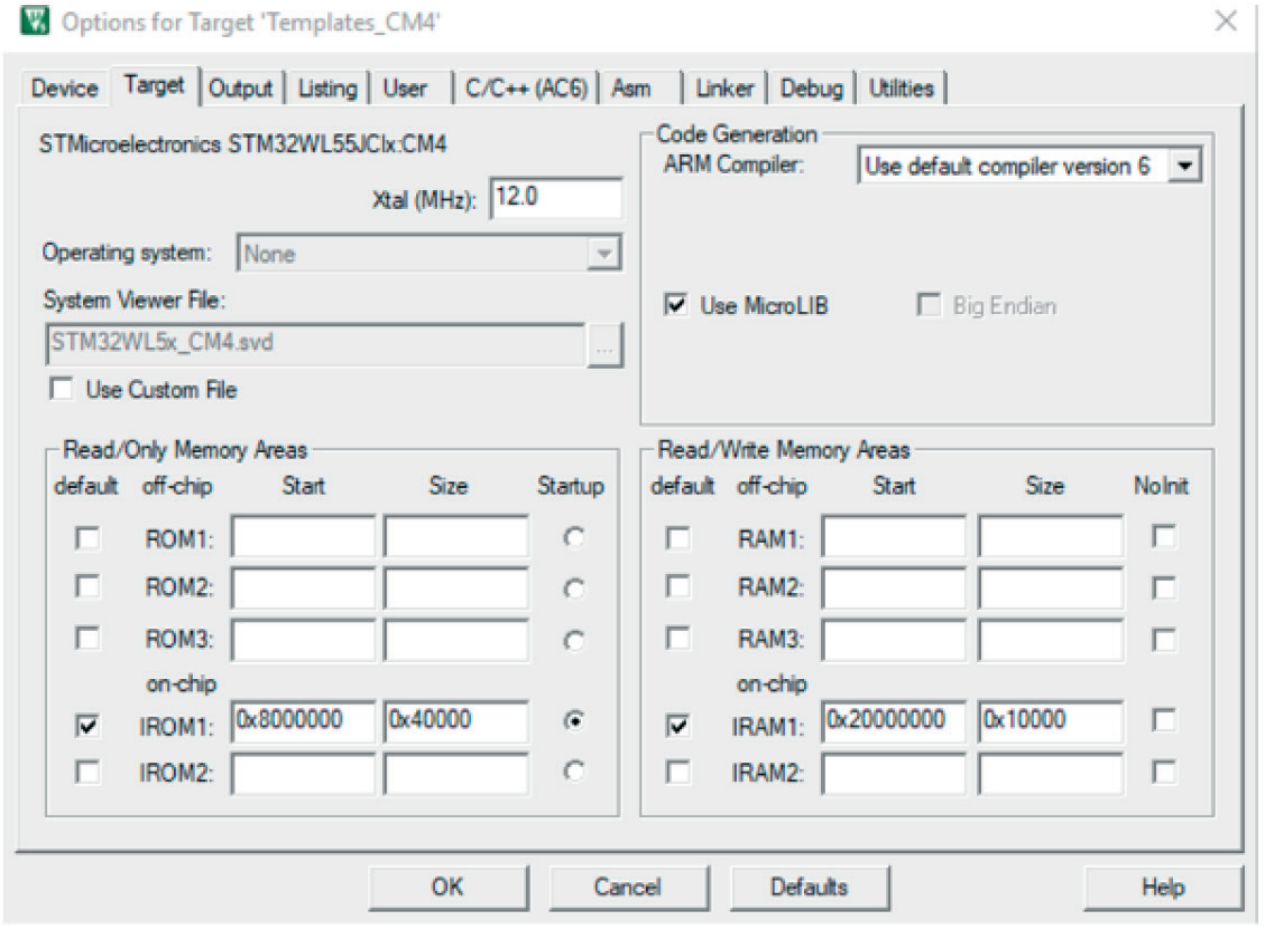
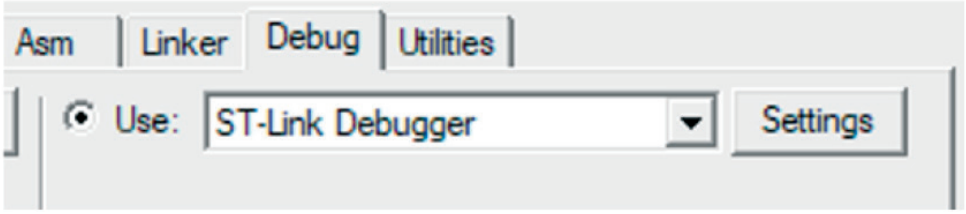
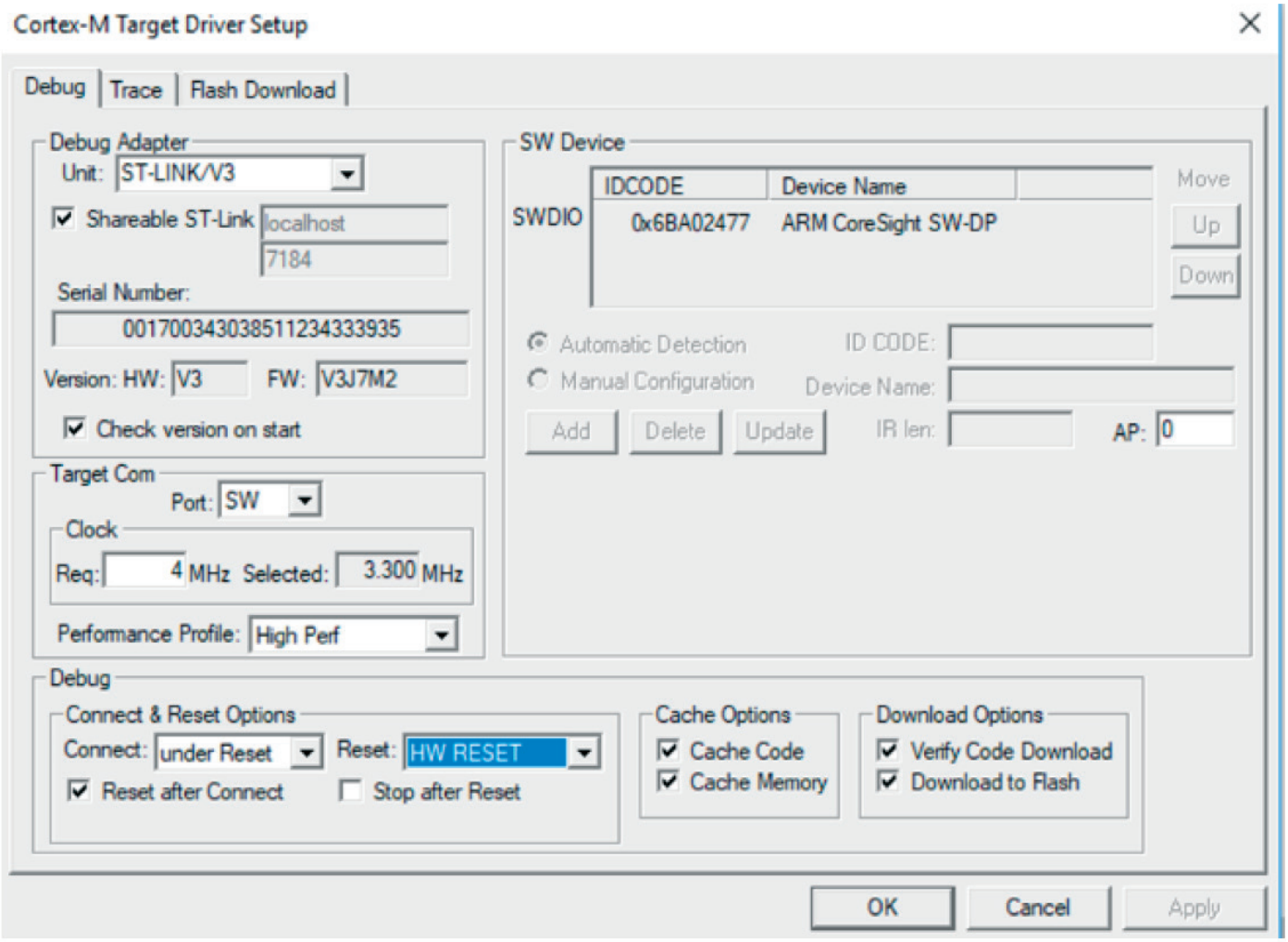


Figure 19. Debug probe selection



d. Перейдите в Debugger -> Settings и убедитесь, что отладчик подключен, как показано на рисунке ниже.

Figure 20. Debug settings tab



e. Включите двухъядерную отладку, установив флажок Shareable ST-Link, чтобы иметь возможность отлаживать оба ядра одновременно.

Примечание. Чтобы настроить многоядерную отладку с помощью зонда отладки ST-LINK, установите последнюю версию сервера ST-LINK.

f. Выберите порт доступа (порт 0 для Cortex-M4).

g. Выберите интерфейс связи как SWD, чтобы использовать канал связи последовательного вывода (SWO) (меньше контактов, чем JTAG).

h. Выберите параметры подключения и сброса.

- Connect under reset сохраняет активным сигнал аппаратного сброса (HW RESET) при подключении к устройству.

- HW RESET выполняет аппаратный сброс путем подачи сигнала аппаратного сброса (HW RESET).

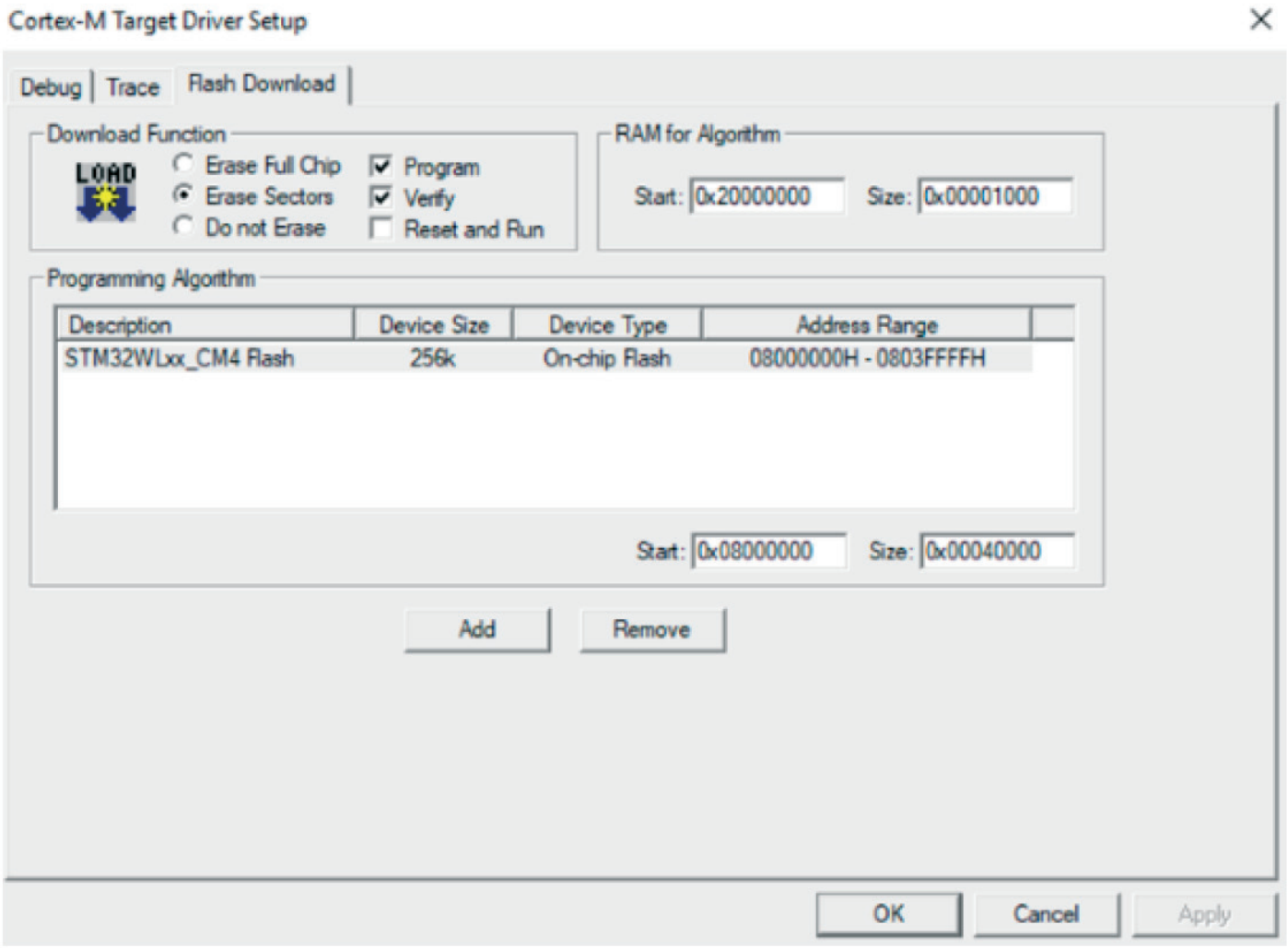
i. Выберите параметры загрузки.

- Проверка загрузки кода останавливает ЦП после выполнения текущей инструкции.
- Download to Flash загружает код во всю область памяти.

j. Из окна загрузки Flash, показанного на рисунке ниже:

- Функция загрузки используется для настройки операций Flash.
- RAM for Algorithm определяет адресное пространство, в котором загружаются и выполняются алгоритмы программирования. Обычно адресное пространство находится во встроенной оперативной памяти.
- Программный алгоритм содержит определения для программирования флэш-памяти.

Figure 21. CM4 Flash loader settings



4.1.2 Настройки проекта CM0 + (MDK-ARM)

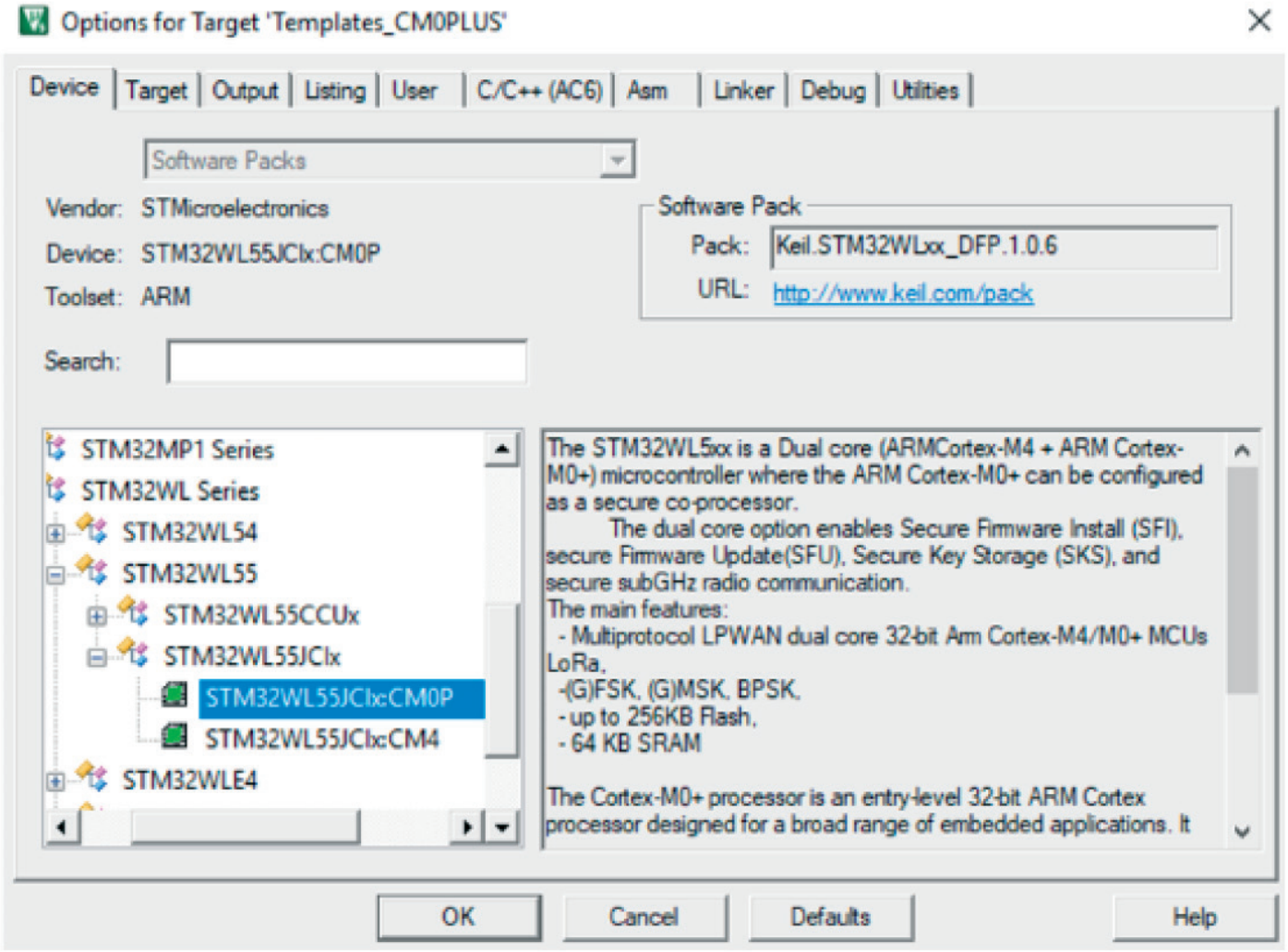
В этом разделе описаны настройки проекта CM0 + и используется шаблон проекта из STM32Cube_FW_WL_V1.0.0 с именем Templates_CM0PLUS.

1. Откройте проект Templates_CM0PLUS в другом экземпляре и убедитесь, что настройки совместимы с параметрами, указанными ниже.

а. Перейдите в Project -> Options for Target ‘Templates_CM0PLUS’

б. Выберите правильное устройство, открыв окно конфигурации и выбрав Project -> Options for Target -> Device, затем выберите устройство STM32WL55JCIx: CM0P из списка.

Figure 22. STM32WL55JCIx - CM0+ device selection



c. В разделе Project -> Options for Target -> Target -> Read / Only Memory Areas убедитесь, что выбрана правильная область памяти: загрузка из основной флэш-памяти по адресу 0x0802 0000 и загрузка из памяти SRAM1 по адресу 0x2000 4000

d. Перейдите в Debugger -> Settings и убедитесь, что отладчик подключен, как показано на рисунке ниже.

e. Включите двухъядерную отладку, установив флажок Shareable ST-Link, чтобы иметь возможность отлаживать оба ядра одновременно.

f. Выберите порт доступа (порт 1 для Cortex-M0 +).

g. Выберите интерфейс связи как SWD, чтобы использовать канал связи последовательного вывода (SWO) (меньше контактов, чем JTAG).

h. Выберите параметры подключения и сброса.

- Connect Normal останавливает CPU на выполняемой в данный момент инструкции после подключения.

- SYSRESETREQ выполняет программный сброс, устанавливая бит SYSRESETREQ. Ядро Cortex-M0 + и периферийные устройства на кристалле сбрасываются (единственный режим сброса, поддерживаемый Cortex-M0 +).

i. Выберите параметры загрузки.

- Проверка загрузки кода останавливает ЦП после выполнения текущей инструкции.

- Download to Flash загружает код во всю область памяти.

Figure 23. Memory area selection

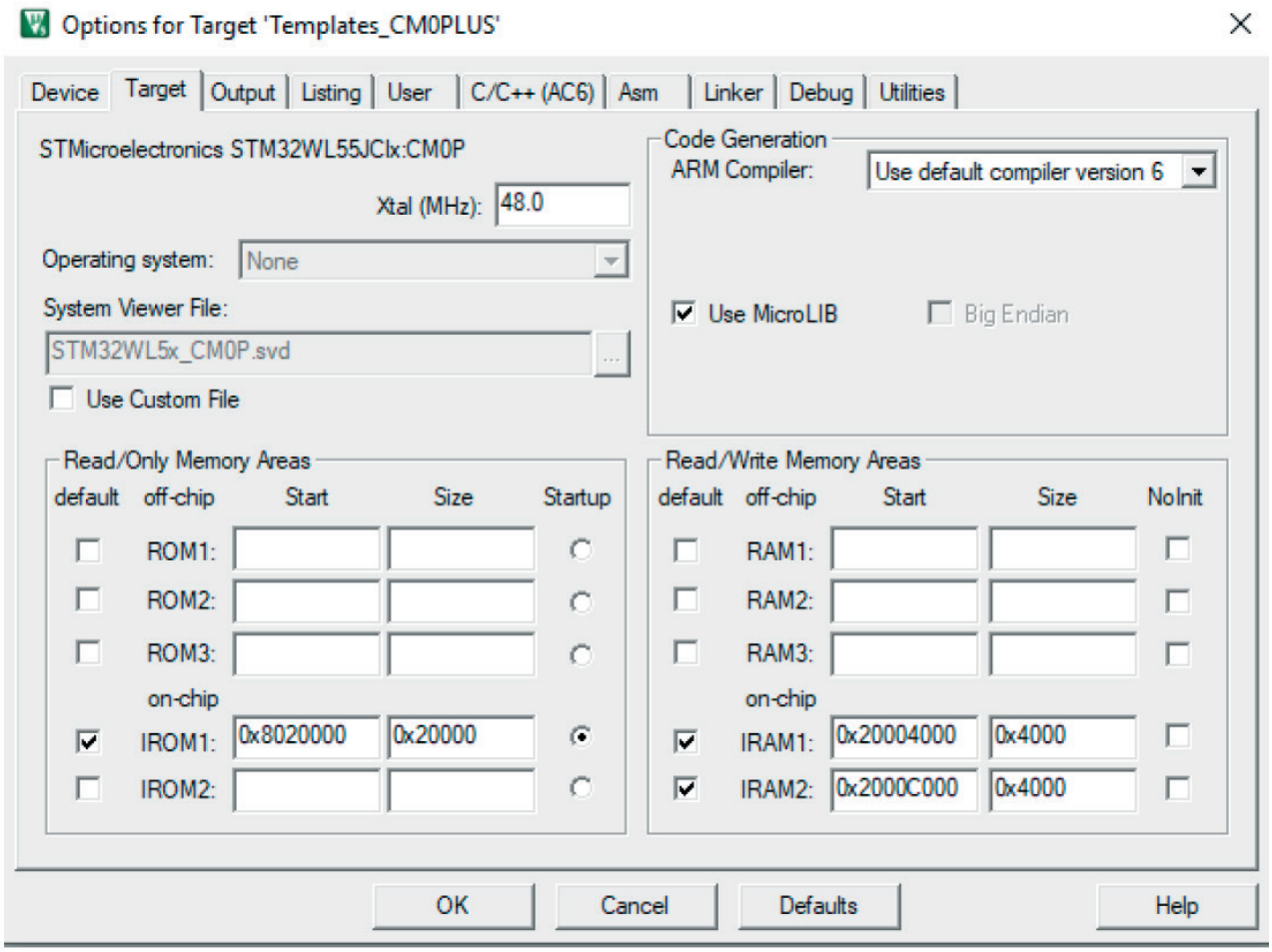
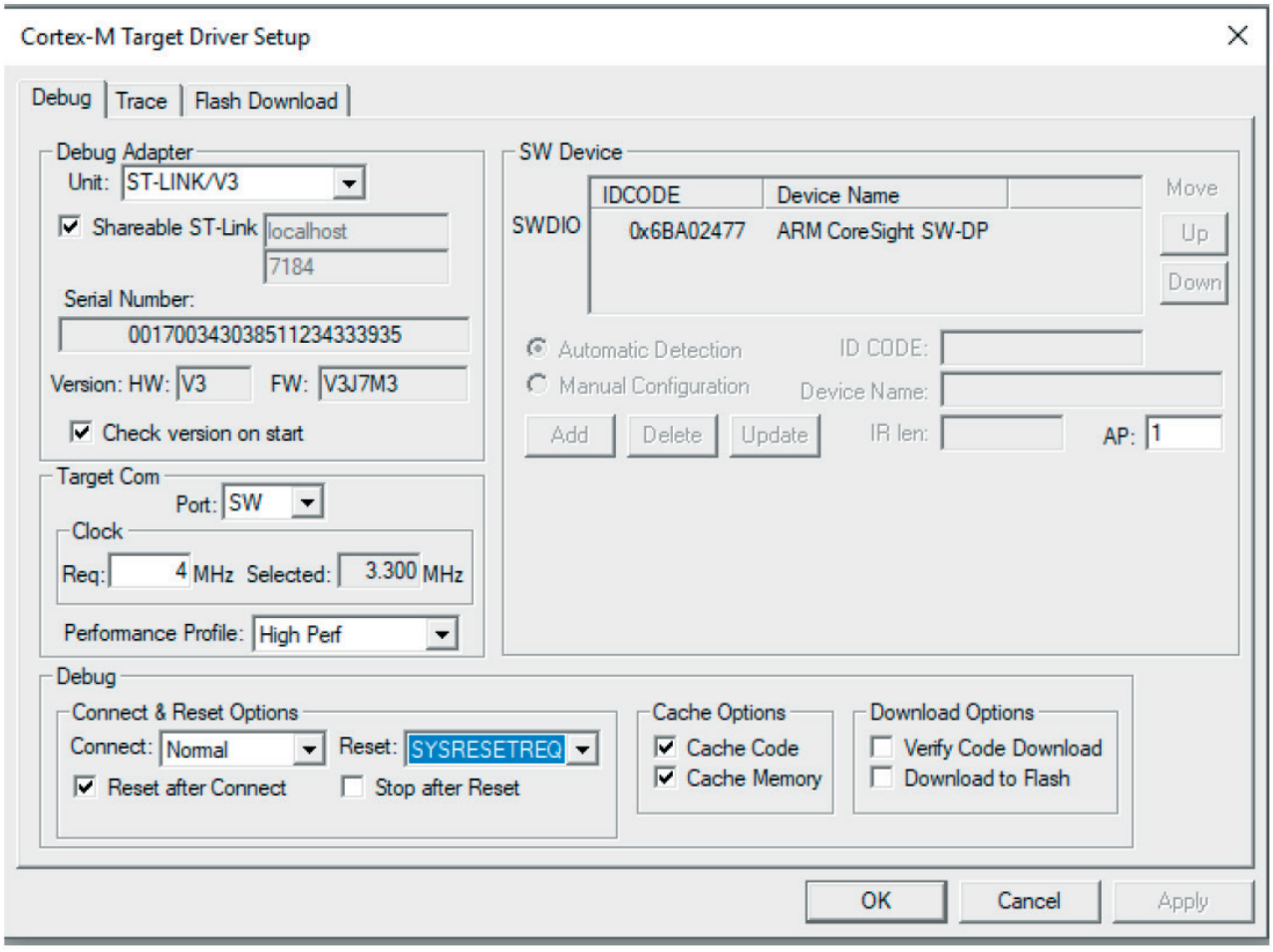


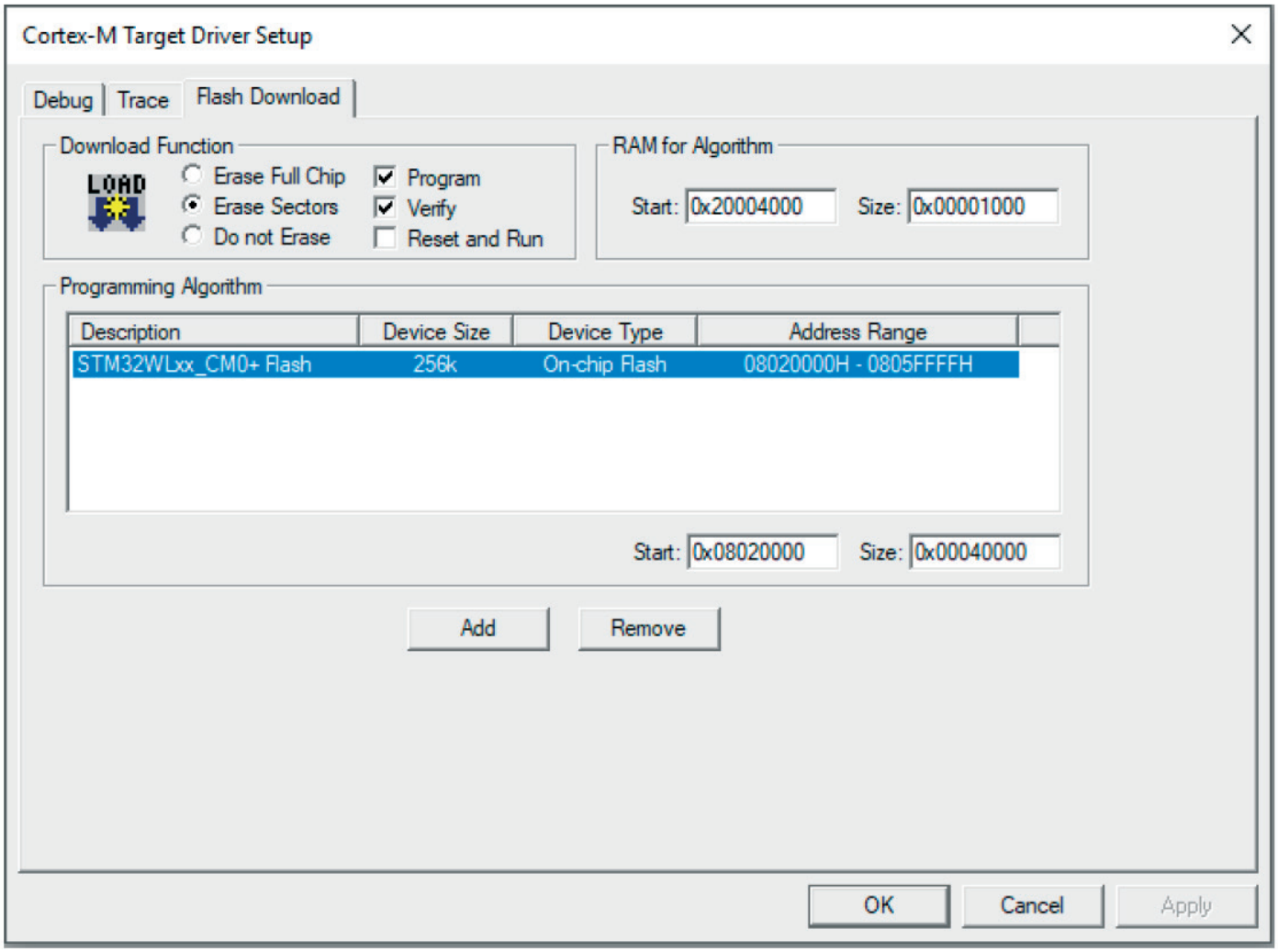
Figure 24. Debugger settings



j. Из окна загрузки Flash, показанного на рисунке ниже:

- Функция загрузки используется для настройки операций Flash.
- RAM for Algorithm определяет адресное пространство, в котором загружаются и выполняются алгоритмы программирования. Обычно адресное пространство находится во встроенной оперативной памяти.
- Алгоритм программы содержит определения для программирования флэш-памяти.

Figure 25. CM0+ Flash loader settings



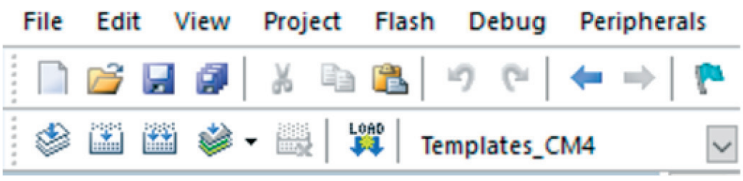
4.1.3 Загрузка и отладка обоих проектов (MDK-ARM)

Перед загрузкой проекта подключитесь к плате NUCLEO-WL55JC (см. Рисунок 13)

- Подключите инструмент программирования и отладки STLINK-V3E к плате NUCLEO-WL55JC.
- Подключите кабель USB к разъему CN1 USB STLINK на плате.
- LED6 горит красным при подключении ST-LINK

1. Скомпилируйте и загрузите проект templates_CM4, используя кнопку Build and Load (Создать и загрузить).

Figure 26. Build and Load button



2. Запустите сеанс отладки, затем выполняйте приложение до тех пор, пока не появится инструкция, устанавливающая C2BOOT (см. Рисунок ниже). Этот проект загружает Cortex-M0 +, устанавливая бит C2BOOT в PWR_CR4.

Figure 27. **Start debug session**

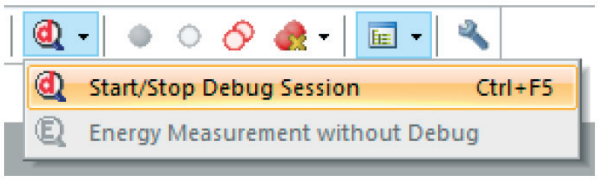


Figure 28. **Main.c sample code**

```
/* Reset of all peripherals, Initializes the Flash interface and the Systick. */
HAL_Init();

/* USER CODE BEGIN Init */

/* USER CODE END Init */

/* Configure the system clock */
SystemClock_Config();
/* Boot CPU2 */
HAL_PWREx_ReleaseCore(PWR_CORE_CPU2);

/* USER CODE BEGIN SysInit */

/* USER CODE END SysInit */

/* Initialize all configured peripherals */
/* USER CODE BEGIN 2 */

/* USER CODE END 2 */

/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
  /* USER CODE BEGIN 3 */

  /* USER CODE END 3 */
}
  /* USER CODE END WHILE */
```

Release the CPU2 from holding

3. Соберите и загрузите проект Templates_CM0PLUS.

4. Запустите сеанс отладки.

Примечание:

• Подключение к двум ядрам одновременно в режиме совместного использования возможно только в том случае, если опция ”Stop after Reset” (Остановить после сброса) отключена, чтобы гарантировать, что C2BOOT не сбрасывается. Cortex-M0 загружается только после того, как Cortex-M4 установил C2BOOT в PWR_CR4.

• Возможно подключение к каждому ядру отдельно (без режима совместного использования), но бит C2BOOT должен быть установлен перед подключением к Cortex-M0 +

5 Отладка и программирование сценариев использования с EWARM и MDK-ARM

В этом разделе подробно описывается конкретный вариант использования, описывая поддержку устройств STM32WL5x в EWARM и MDK - ARM.

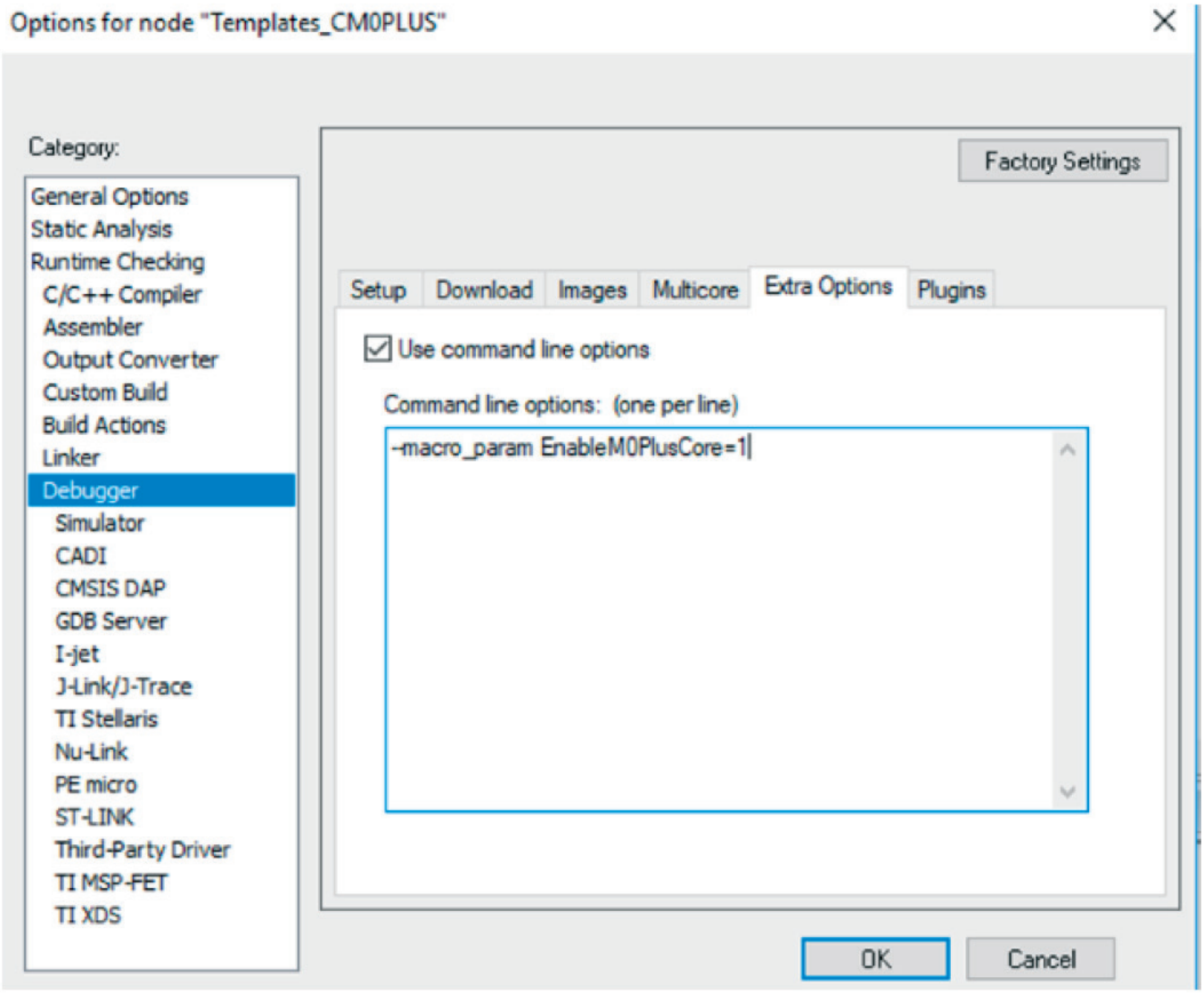
5.1 Как подключить и загрузить приложение Cortex-M0 +, когда вся флэш-память пуста (EWARM)

Важно: перед подключением к ядру Cortex-M0 + убедитесь, что бит C2BOOT в PWR_CR4 включен.

С помощью следующих шагов можно загрузить приложение Cortex-M0 +, пока вся флэш-память пуста.

1. C2BOOT должен быть автоматически установлен перед подключением AP1 для шага загрузки путем добавления дополнительной опции --macro_param EnableM0PlusCore = 1 из опции Project -> Debugger -> Extra options.

Figure 29. Enable Cortex-M0+ core

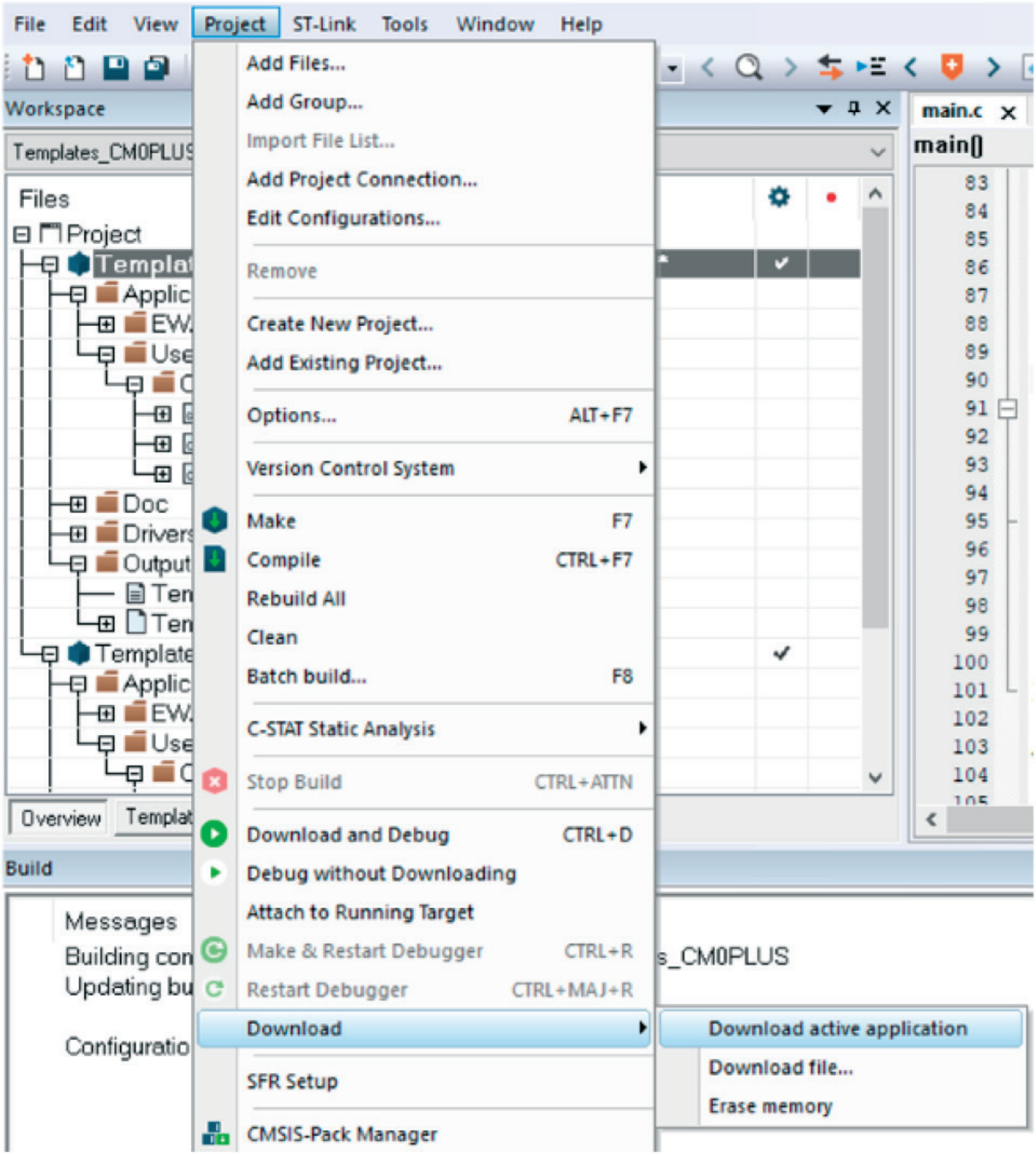


2. Перейдите в Project -> Option -> ST-LINK -> Setup и выберите режим сброса в качестве сброса системы.

3. Соберите приложение и выберите Project -> Download -> Download active application (Загрузить активное приложение), чтобы запрограммировать приложение Cortex - M0 + во флэш-памяти по адресу 0x0802 0000.

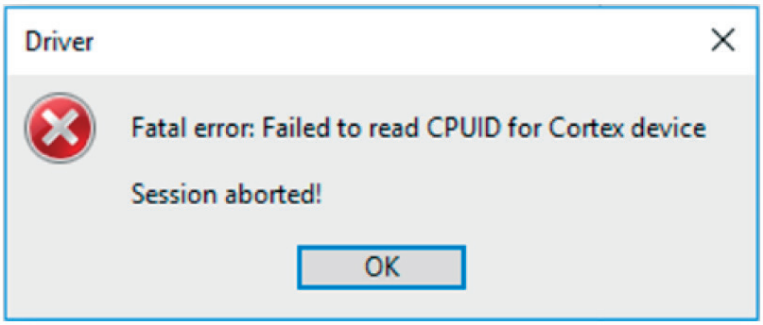
4. Загрузите активное приложение в целевой объект без запуска полного сеанса отладки, используя Project -> Download -> Download active application.

Figure 30. Download active application



Следующее сообщение об ошибке указывает, что C2BOOT не установлен. При необходимости среда IDE пытается подключиться с помощью нескольких попыток, пока не истечет время ожидания в 1 с (для ожидания возможного включения C2BOOT из приложения), иначе среда IDE возвращает ошибку, указывающую, что точка доступа AP1 недоступна.

Figure 31. **Error - AP1 not accessible**



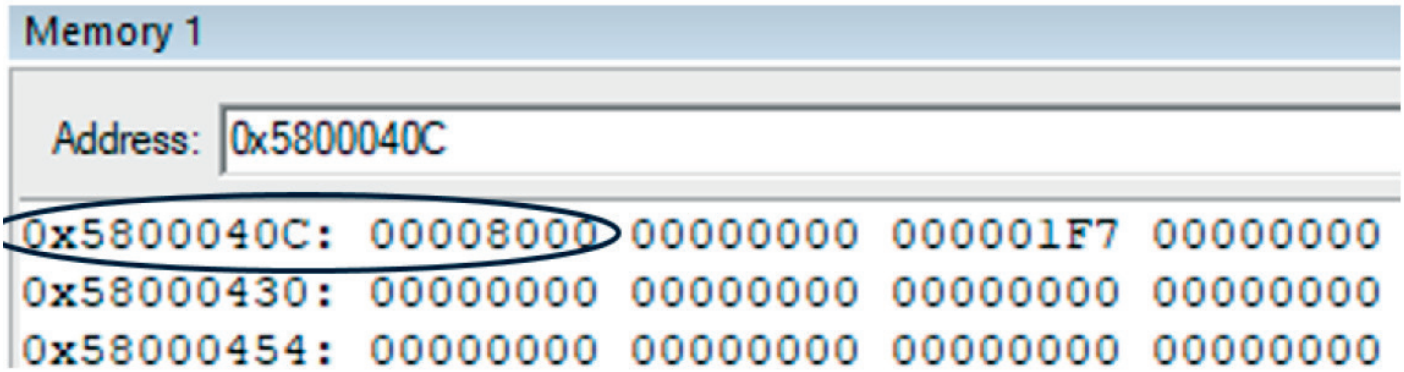
5.2 Как подключить и загрузить приложение Cortex-M0 +, когда вся флэш-память пуста (MDK-ARM)

С помощью следующих шагов можно загрузить приложение Cortex-M0 +, пока вся флэш-память пуста.

1. C2BOOT должен быть автоматически установлен перед подключением AP1 для этапа загрузки.

- Подключитесь к AP0 с помощью MDK-ARM (см. Раздел 4.1.1).
- Откройте окно памяти, выбрав View > Memory.
- Записать 0x0000 8000 в PWR_CR4 (по адресу 0x5800 040C).

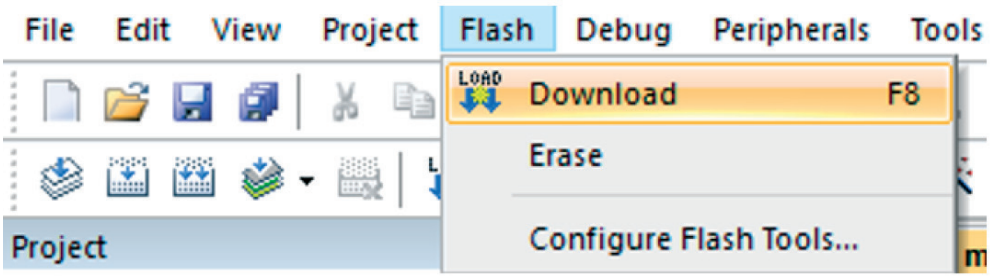
Figure 32. **Enable CPU2 using MDK-ARM**



2. Перейдите на вкладку Debug и убедитесь, что отладчик подключен, как показано на рисунке 24.

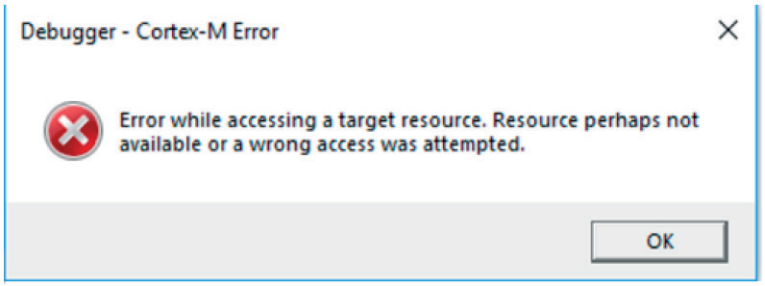
3. Соберите приложение и перейдите в Flash -> Download, чтобы запрограммировать активное приложение на целевое устройство без запуска полного сеанса отладки.

Figure 33. **Download option**



При попытке проделать ту же манипуляцию перед настройкой C2BOOT появляется сообщение об ошибке, указывающее, что AP1 недоступен.

Figure 34. AP1 not accessible



6 Безопасное программирование

В этом разделе описываются шаги, необходимые для включения безопасности, чтобы защитить области памяти (флэш-память, SRAM1 и SRAM2) от доступа со стороны любого неавторизованного мастера шины.

Области SRAM1 и SRAM2 защищены только тогда, когда включена защита флэш-памяти (ESE = 1 и FSD = 0).

Безопасная флэш-память

Безопасная область флэш-памяти:

• начинается с базового адреса флэш-памяти + (включенный SFSA [6: 0] x 0x0800), где SFSA [6: 0] является безопасным стартовым адресом флэш-памяти и содержит начальный адрес первой 2-килобайтной страницы защищенной области флэш-памяти. .

• заканчивается на последнем адресе флэш-памяти

Примечание. Когда включена защита CPU2, минимальный размер защищенной области CPU2 составляет один сектор (2 Кбайта). Например, с защищенной областью CPU2 от 0x0802 7000 (включительно) до 0x0803 FFFF (включительно), FLASH_SFR должен быть запрограммирован с SFSA = 0x4E.

Флаг ESE в FLASH_OPTR указывает, включена ли защита CPU2. Любой доступ CPU1 к области безопасности CPU2 вызывает ошибку флага RDERR или WRPERR.

Безопасная не резервная SRAM1

Безопасная не резервная область SRAM1:

• начинается с небезопасного базового адреса SRAM1 + (SNBRSA [4: 0] x 0x0400) включен, где SNBRSA [4: 0] - это безопасный начальный адрес небезопасной SRAM1 и содержит начальный адрес первой страницы размером 1 Кбайт безопасная не резервная область SRAM1.

• заканчивается на не резервном последнем адресе SRAM1

Например, с защищенной областью SRAM1 CPU2 от 0x2000 6C00 (включительно) до 0x2000 7FFF (включительно), FLASH_SRRVR должен быть запрограммирован с SNBRSA = 0x1B. Любой доступ на чтение CPU1 возвращает нулевые данные. Доступ для записи в защищенную область SRAM1 CPU2 отбрасывается и генерирует событие незаконного доступа.

Примечание. Если для NBRSD установлено значение 1, SRAM1 не является защищенным.

Безопасная SRAM2

Безопасная область SRAM2:

• начинается с базового адреса SRAM2 + (включенный SBSRA [4: 0] x 0x0400),

где SBSRA [4: 0] - это безопасный начальный адрес SRAM2 и содержит начальный адрес первой 1-килобайтной страницы защищенной области SRAM2.

• заканчивается на последнем адресе SRAM2. Например, для защищенной области SRAM2 CPU2 от 0x2000 A800 (в комплекте) до 0x2000 FFFF (в комплекте), FLASH_SRRVR должен быть запрограммирован с SBSRA = 0x0A. Любой доступ на чтение CPU1 возвращает нулевые данные. Доступ на запись в защищенную область SRAM2 CPU2 отбрасывается и генерирует событие незаконного доступа.

Примечание. Если для BRSD установлено значение 1, SRAM2 не является защищенным.

Option-byte setup (Установка байт опций)

Перед защитой системы и памяти, как подробно описано в следующих разделах, байты опций должны быть настроены следующим образом с помощью инструмента STM32CubeProgrammer:

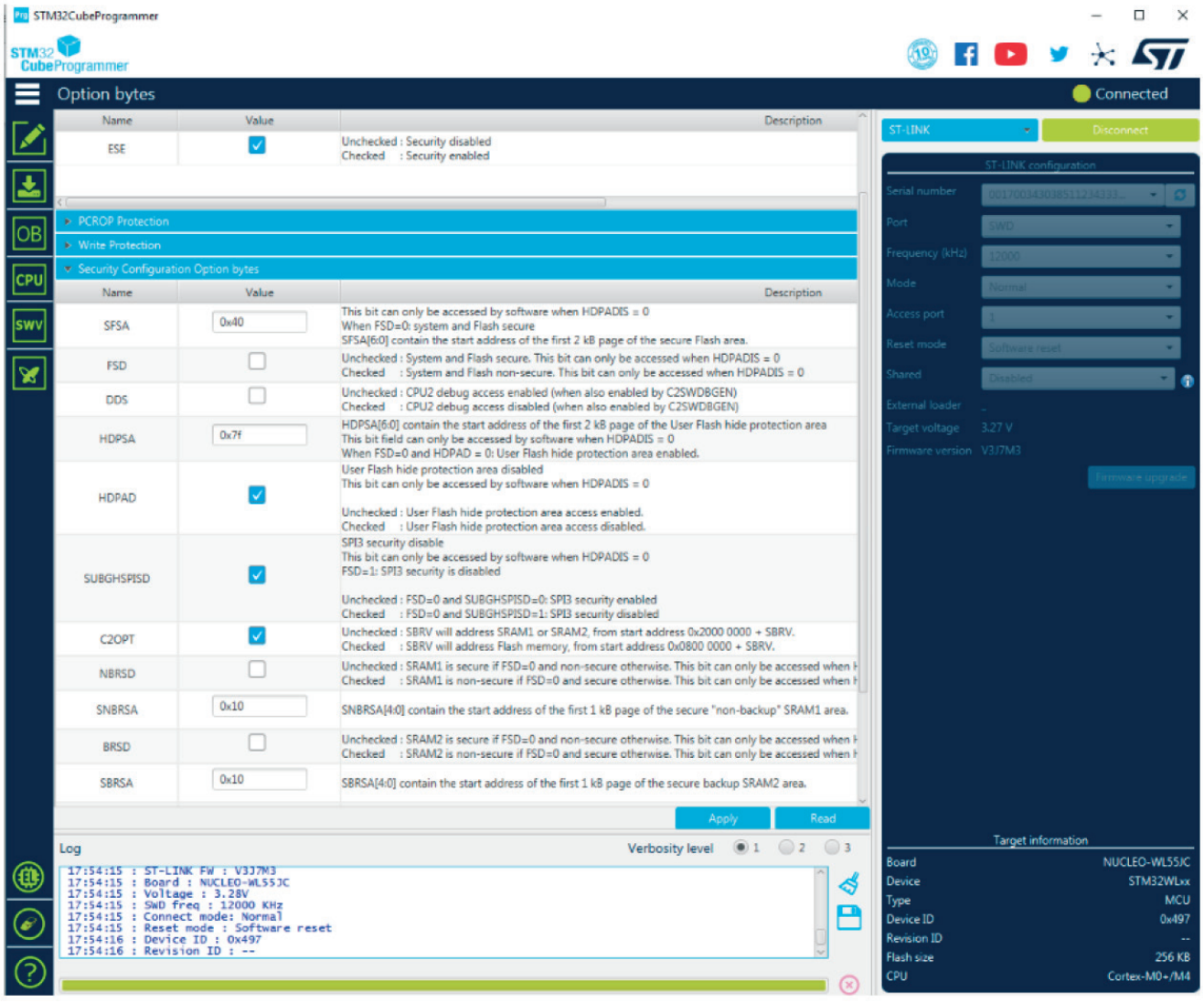
• FSD = 0 для включения общей безопасности системы

• SFSA [6: 0] = 0x40 в FLASH_SFR для защиты второй половины области флэш-памяти.

• SNBRSA [4: 0] = 0x10 и NBRSD = 0 в FLASH_SRRVR для защиты второй половины нерезервной SRAM1

• SBRSA [4: 0] = 0x10 и BRSD = 0 в FLASH_SRRVR для защиты второй половины резервной SRAM2

Figure 35. Option-byte configuration



6.1 Безопасное программирование с помощью EWARM

Следующие шаги необходимы для выполнения безопасного программирования с использованием платы NUCLEO_WL55JC и EWARM:

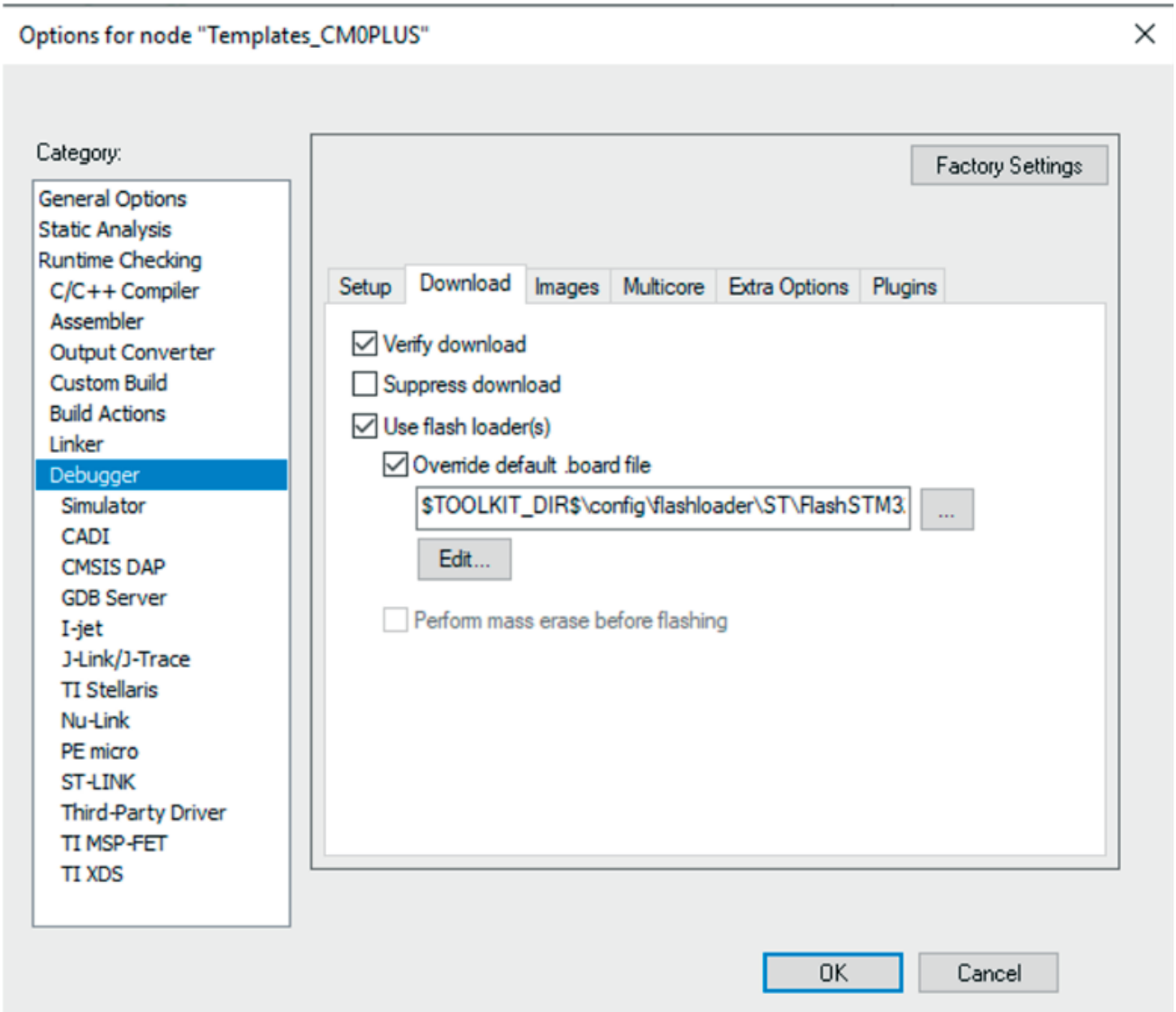
1. Убедитесь, что все байты опций сконфигурированы, как во введении в Раздел 6 «Безопасное программирование».

2. Откройте проект CM4 и убедитесь, что все параметры проекта настроены, как описано в Разделе 3.1.1.

3. Откройте проект CM0 + и убедитесь, что все параметры проекта настроены, как описано в Разделе 3.1.2.

4. Перейдите в Project options -> Debugger -> Download tab from the CM0+ project и замените флэш-загрузчик, выбранный FlashSTM32WL_SEC.board, который доступен в разделе
C: \ Program Files (x86) \ IAR Systems \ Embedded Workbench 8.4_3 \ arm \ config \ flashload er \ ST.

Figure 36. Change the default flashloader

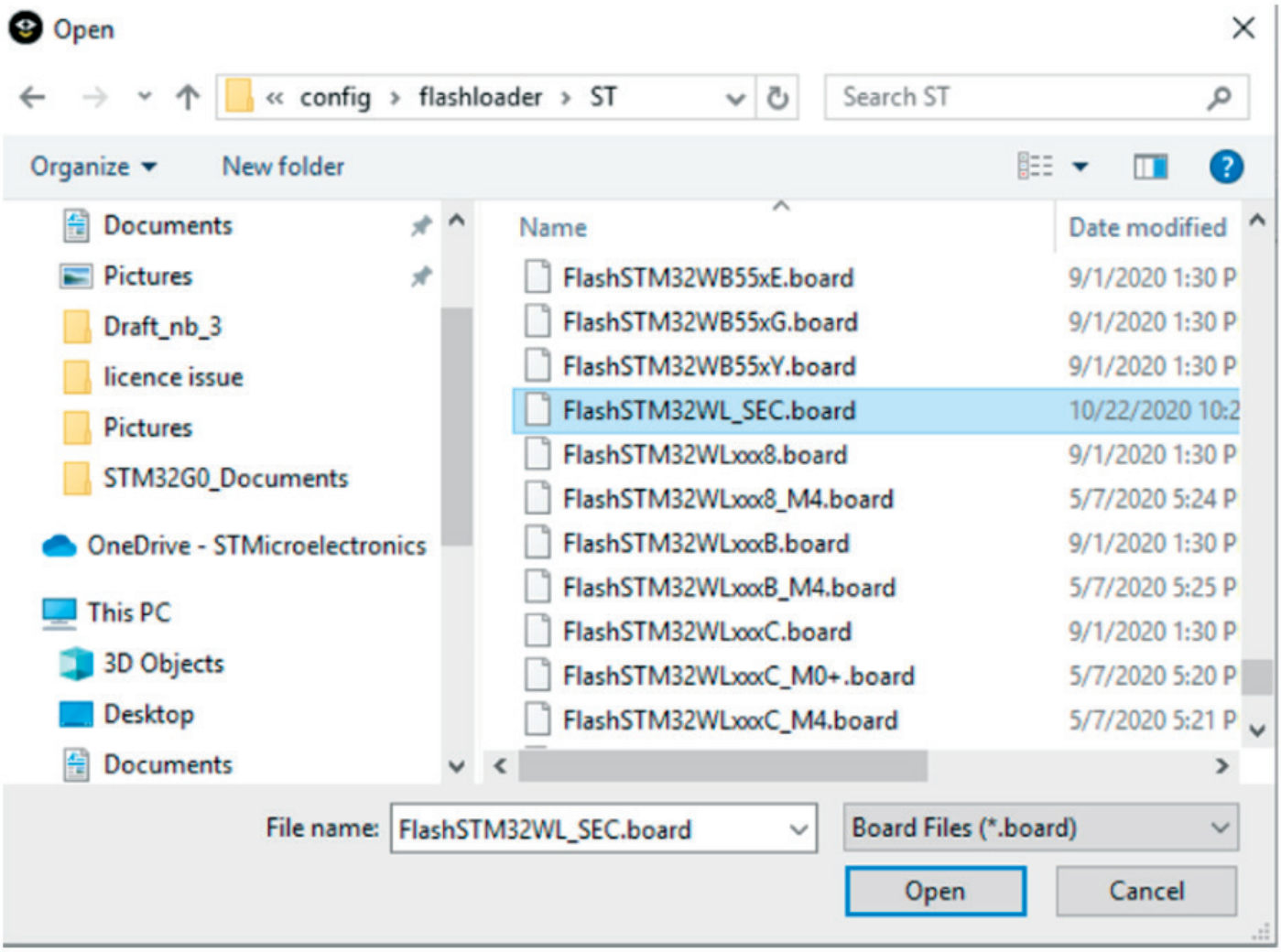


5. Соберите и загрузите приложение Cortex-M4.

6. Настройте C2BOOT на выполнение загрузки CPU2 (Cortex-M0 +) в PWR_CR4.

7. Соберите и загрузите приложение Cortex-M0 +: программирование флэш-памяти выполняется с помощью интерфейса JTAG.

Figure 37. Select the secure flashloader



6.2 Безопасное программирование с использованием MDK-ARM

Следующие шаги необходимы для выполнения безопасного программирования с использованием платы NUCLEO_WL55JC и MDK-ARM:

1. Убедитесь, что все байты опций настроены, как во введении в Раздел 6.
2. Откройте проект CM4 и убедитесь, что все параметры проекта настроены, как описано в Разделе 4.1.1.
3. Откройте проект CM0 + и убедитесь, что все параметры проекта настроены, как описано в Разделе 4.1.2.
4. Перейдите в Project options -> Debugger > Flash Download tab и обновите адрес выполнения флэш-загрузчика в ОЗУ для алгоритма, указав безопасный адрес SRAM.
5. Соберите и загрузите приложение Cortex-M4.
6. Настройте C2BOOT на выполнение загрузки CPU2 (Cortex-M0 +) в PWR_CR4.
7. Соберите и загрузите приложение Cortex-M0 +: программирование флэш-памяти выполняется с помощью интерфейса JTAG.

Figure 38. Change the flashloader execution address