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) STM32WL5х.

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 вебсайте Arm®)

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

2 Архитектура STM32WL5х

Двухъядерные микроконтроллеры STM32WL5х 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 (АР):

- АРО: порт доступа отладки СРU1 (АНВ-АР)
- АР1: порт доступа отладки СРU2 (АНВ-АР)

2.2 Загрузка СРU2

СРU2 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-М

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 (СТІ)
• Интерфейсный блок порта трассировки (TPIU)	
• Интерфейс перекрестного запуска (СТІ)	

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.



2.5 Глобальный контроллер безопасности (GTZC)

Устройства STM32WL5х 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 STM32WL5х 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 СМ4 (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.

Workspace	•	ņχ
Templates_CM4 - Templates_CM4		~
Files	Ф	•
🗆 🗂 Project		
├─⊞ 🌒 Templates_CM0PLUS - Templates	~	
	v	
🗕 🖃 📠 Application		
└─── ■ Output		
Overview Templates_CM0PLUS Templates_CM4		

Figure 3. EWARM project explorer view

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

а. Перейдите в Project > Options. В общих параметрах выберите устройство STM32WL55JC M4 (см. Рисунок ниже)

Figure 4. STM32WL55JC_M4 device selection

Options for node "Template	s_CM4"				×
Category: General Options Static Analysis Runtime Checking C/C++ Compiler Assembler Output Converter Custom Build Build Actions Linker Debugger Simulator CADI CMSIS DAP GDB Server I-jet J-Link/J-Trace TI Stellaris Nu-Link PE micro ST-LINK Third-Party Driver TI MSP-FET TI XDS	Library Options 2 Target Outp Processor variant O Core O CMSIS-Pack Endian mode O Little Big BE32 O BE32 O SP Extension Advanced SIM	MISRA- ut Library Co Cortex-M4 ST STM32WL55 None Roating point s FPU D registers D (NEON)	C:2004 onfiguration JC_M4 ettings None Mode Sec	MISRA-C:1998 Library Options 1	
			OK	Cancel	

×

Options for node "Templates_CM4"

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.

ategory:						Factory Settings		
General Options Static Analysis Suntime Checking								
C/C++ Compiler	#define	Diag	postics	Checksum	Encodings	Extra Options		
Assembler	Config	Library	Input	Optimizations	Advanced	Output List		
Output Converter Custom Build Build Actions	Linke	r configur verride de	ation file fault					
Linker	4	PROJ_D	IR\$/stm32	2wl55xx_flash_cm	4.icf			
Debugger Simulator CADI	Linker configuration file editor X							
GDB Server	Vector T	able Me	mory Reg	ions Stack/Hea	p Sizes			
J-Link/J-Trace TI Stellaris	ROM		Sta 0	art: x08000000	End: 0x0801F	FFF		
Nu-Link PE micro	RAM		0	×20000000	0x20003	3FFF		

Figure 5. CM4 project - Linker configuration



Category:	Factory Settin						
General Options Static Analysis Runtime Checking C/C++ Compiler Assembler Output Converter Custom Build	Setup Download Images Multicore Extra Options Plugins Driver I Run to ST-LINK main						
Build Actions Linker Debugger Simulator CADI CMSIS DAP GDB Server	Setup macros						
I-jet J-Link/J-Trace	Device description file						
TI Stellaris Nu-Link PE micro ST-LINK Third-Party Driver TI MSP-FET TI XDS	\$TOOLKIT_DIR\$\CONFIG\debugger\ST\STM32WL55JC_M4						

с. 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».

Category:						Factory S	Settings
General Options Static Analysis Runtime Checking							
C/C++ Compiler	Setup	Download	Images	Multicore	Extra Options	Plugins	
Assembler Output Converter Custom Build Build Actions Linker	⊠ Ve ⊡ Su ⊡ Su	rify download ppress down e flash loade	d Ioad r(s)				
Debugger		Override de	fault .boar	d file			
Simulator CADI CMSIS DAP		\$TOOLKIT	_DIR\$\co	onfig\flashlo	ader\ST\FlashS	TM3.	

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

Options for node "Templates_CM4" X		Figure 8. ST-LINK setup	
	Options for node "Templates	5_CM4"	×
Category: Factory Settings General Options Static Analysis Runtime Checking C/C++ Compiler Assembler Output Converter Output Converter Auto Custom Build Build Actions Linker Debugger Simulator Auto CADI Connect during reset OHSIS DAP Interface GDB Server I-jet J-Link/J>-Trace JTAG TI Stellaris Nu-Link PE micro SWD Stellaris SWD Nu-Link SWD PE micro SWD Static No Specify Its MSP-FET TI XDS	Category: General Options Static Analysis Runtime Checking C/C++ Compiler Assembler Output Converter Custom Build Build Actions Linker Debugger Simulator CADI CMSIS DAP GDB Server I-jet J-Link/J-Trace TI Stellaris Nu-Link PE micro ST-LINK Third-Party Driver TI MSP-FET TI XDS	Setup Communication Breakpoints Multicore Emulator	Factory Settings

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 СМ4).

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 × Options for node "Templates CM4" Category: Factory Settings General Options Static Analysis **Runtime Checking** Setup Communication Breakpoints Multicore C/C++ Compiler Assembler Output Converter Enable multicore debugging/shared mode Custom Build Build Actions Linker Debugger Simulator CADI CMSIS DAP GDB Server I-jet J-Link/J-Trace TI Stellaris Nu-Link PE micro ST J INI Third-Party Driver TI MSP-FET TI XDS ΟK Cancel

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 СМ0 + (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.

а. 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

с. 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).

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

Figure 10. ST32WL55JC_M0+ device selection

Category:						٦
General Options						
Static Analysis						
Runtime Checking	11-0-11-0	2	MICDU	0.0004	NICDA C 1000	
C/C++ Compiler	Library Option	\$ 2	MISKA	A-C:2004	MISHA-C: 1998	
Assembler	larget 0	output	Library (Configuration	Library Options 1	
Output Converter	Processor varia	ant				
Build Actions	OCore	Cort	ex-M0+	\sim		
Linker						
Debugger	Device	STS	STM32WL5	5JC_M0+	°.	
Simulator	0.000	Non	0			
CADI	O CMSIS-Pac	*	u			
CMSIS DAP	Endanmada		anting paint	actions		
GDB Server	Englan mode		loating point	settings		
I-jet	 Little 	F	PU	None	\sim	
J-Link/J-Trace	Big					
TI Stellaris	O BE32) registers	- V		
Nu-Link	BE8					
PE micro				T		
ST-LINK	DSP Extens	ion		TrustZo	ne	
Third-Party Driver				Mode Se	cure 🗸	
TI MSP-FET	Advanced S	SIMD (NE	ON)			

Figure 11. CM0+ project - Linker configuration

Jalegoly.						Factory 9	etting
General Options Static Analysis Runtime Checking							
C/C++ Compiler	#define	Diag	nostics	Checksum	Encodings	Extra (Options
Assembler	Config	Library	Input	Optimizations	Advanced	Output	List
Output Converter	Linker	configur	ation file				
Custom Build		uorrido do	f au de				
Build Actions		venide de			.		_
Linker	S	PROJ_D	IR\$/stm32	2wi55xx_flash_cm	0plus.icf		
Debugger			_				
Simulator	Linker	onfigura	ation file	editor			×
CADI		-					
CMSIS DAP	Vector	Table M	lemory Re	gions Stack/He	ap Sizes		
GDD Server							
1-Link/1-Trace			S	tart:	End:		_
TI Stellaris	ROM			0x08020000	0x080	3FFFF	
Nu-Link							_
PE micro	RAM			0x20004000	0x200	07FFF	
ST-LINK							
Third-Party Driver							
TT MOD FET						Sav	e
TIMSP-FET							

FIGURE 12. ST-LINK SETU	Figure 1	12. S	F-LINK	setur
-------------------------	----------	-------	---------------	-------

Options for node "Template	:_CM0PLUS"	×
Category: General Options Static Analysis Runtime Checking C/C++ Compiler Assembler Output Converter Custom Build Build Actions Linker Debugger Simulator CADI CMSIS DAP GDB Server I-jet J-Link/J-Trace TI Stellaris Nu-Link PE micro ST-LINK Third-Party Driver TI MSP-FET TI XDS	Setup Communication Breakpoints Multicore Emulator Auto Serial no: Always prompt for probe selection Reset System (default) Interface JTAG O JTAG O SWD Default O Specify	Factory Settings
	ОК	Cancel

f. Выберите порт доступа:

- Авто: порт доступа 1 для Cortex-M0 + используется автоматически.

- Вручную: можно выбрать порт доступа вручную (например, поставив 1 с СМ0 +).

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. 1/
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)
Ł
  /* 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 СМ4 (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.

File	Edit	View	Project	Flash	Debug	Peripherals	Tools	SVCS	Window	Help	
1	<u>2</u>		X 🗈	1	9 (*	← → 🏞	12.1	1 19 1		//= //# 🗹	\$
٢		🗄 🤪 ·	- 📖 🛛	Te	emplates_C	M4 🗸	*	a 5	۰ 🔶		
Projec	t			Te	emplates_C emplates_C	M4 MOPLUS) ma	in.c			
±.4	Proj	ect: Tem	plates				1	/* t	JSER CO	DE BEGIN	He

Figure 16. Project explorer view

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

а. 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 флэшпамяти по адресу 0x0800 0000 и загрузка из памяти SRAM1 по адресу 0x2000 0000.

с. 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

Options for Target 'Templates_CM4'	>
Device Target Output Listing User C/C++ Software Packs Vendor: STMicroelectronics Device: STM32WL55JCk:CM4 Toolset: ARM	(AC6) Asm Linker Debug Utilities V Software Pack Pack: Keil.STM32WLxx_DFP.1.0.6 URL: http://www.keil.com/pack
Search: STM32MP1 Series STM32WL Series STM32WL54 STM32WL55 STM32WL55 STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55JClx STM32WL55ZClx STM32WL55ZClx STM32WL55ZClx STM32WL55ZClx STM32W	The STM32WL5xx is a Dual core (ARMCortex-M4 + ARM Cortex- M0+) microcontroller where the ARM Cortex-M0+ can be configured as a secure co-processor. The dual core option enables Secure Firmware Install (SFI), secure Firmware Update(SFU), Secure Key Storage (SKS), and secure subGHz radio communication. The main features: • Multiprotocol LPWAN dual core 32-bit Arm Cortex-M4/M0+ MCUs LoRa, •(G)FSK, (G)MSK, BPSK, • up to 256KB Flash, • 64 KB SRAM The Cortex-M4 processor is an entry-level 32-bit ARM Cortex processor designed for a broad range of embedded applications. It
ОК	Cancel Defaults Help

Figure 18. Memory area selection

Microe	lectronics	STM32WL55J	IClx:CM4 Xtal (MHz):	2.0	Code C ARM (Generation Compiler:	Use defau	uit compiler ven	sion 6 💌
Operating System V STM32V	system: iewer File: VL5x_CM- Custom Fil	None 4.svd		v 	Vs Us	e MicroLIB		Big Endian	
	Doly Many	Areas			Deed	Wite Manu			
- Read/(default	off-chin	Start	Size	Startup	default	off-chip	Start	Size	Nolnit
default	off-chip ROM1:	Start	Size	Startup	default	off-chip RAM1:	Start	Size	Nolnit
default	off-chip ROM1: ROM2:	Start	Size	Startup C C	default	off-chip RAM1: RAM2:	Start	Size	Nolnit
default	ROM1: ROM2: ROM3:	Start	Size	Startup C C	default	off-chip RAM1: RAM2: RAM3:	Start	Size	Nolnit
Head/C default	off-chip ROM1: ROM2: ROM3: on-chip	Start	Size	Startup C C C	default	off-chip RAM1: RAM2: RAM3: on-chip IRAM1:	Start 0x20000000	Size	NoInit

Figure 19. Debug probe selection

Asm	Linker	Debug	Utilities		
] •	Use: ST	-Link De	bugger	•	Settings

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

Figure 2	0. Debu	g settin	igs tal	b
----------	---------	----------	---------	---

Cortex-M Target Driver Setup	×
Debug Trace Rash Download	
Debug Adapter SW Unit: ST-LINK/V3 ▼ I Shareable ST-Link localhost SWI 7184 7184 SWI Serial Number: 001700343038511234333935 (* 001700343038511234333935 (* C Version: HW: V3 FW: V3J7M2 C I Check version on start I C C C Clock Port: SWI SWI SWI Performance Profile: High Perf ▼ I	IDCODE Device Name Move DIO 0x6BA02477 ARM CoreSight SW-DP Up Automatic Detection ID CDDE: Down Annual Configuration Device Name: AP: 0 Add Delete Update IR len: AP: 0
Debug Connect & Reset Options Connect: under Reset ✓ Reset: I✓ Reset after Connect ✓ Stop after Reset	Cache Options Cache Code Cache Memory OK Cancel Apply

е. 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).

і. Выберите параметры загрузки.

Contex-M Target Driver Ceture

- Проверка загрузки кода останавливает ЦП после выполнения текущей инструкции.

- Download to Flash загружает код во всю область памяти.

ј. 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

Do not Erase Reset and Run Programming Algorithm	
Description Device Size Device Type Address Range	
STM32WLxx_CM4 Flash 256k On-chip Flash 0800000H - 0803FFFFH	
1	
Start: 0x08000000 Size: 0x00040000	
Start: 0x08000000 Size: 0x0000400000 Add Remove Image: Remove	
Start: 0x08000000 Size: 0x000400000 Add Remove Image: Remove	

4.1.2 Настройки проекта СМ0 + (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_CMOPLUS в другом экземпляре и убедитесь, что настройки совместимы с параметрами, указанными ниже.

а. Перейдите в Project -> Options for Target 'Templates_CMOPLUS'

б. Выберите правильное устройство, открыв окно конфигурации и выбрав

Project -> Options for Target -> Device, затем выберите устройство STM32WL55JCIx: СМ0Р из списка.

Figure 22. STM32WL55JCIx - CM0+ device selection

Options for Target 'Templates_CM0PLUS'		×
Device Target Output Listing User C/C++ Software Packs Vendor: STMicroelectronics Device: STM32WL55JClx:CM0P Toolset: ARM	 (AC6) Asm Linker Debug Utilities Software Pack Pack: Keil.STM32WLxx_DFP.1.0.6 URL: <u>http://www.keil.com/pack</u> 	
Search: STM32MP1 Series STM32WL Series STM32WL54 STM32WL55 STM32WL55 STM32WL55JClx STM32WL55JClx:CM0P STM32WL55JClx:CM0P STM32WL55JClx:CM0P STM32WL55JClx:CM0P STM32WL55JClx:CM0P	The STM32WL5xx is a Dual core (ARMCortex-M4 + ARM Cortex- M0+) microcontroller where the ARM Cortex-M0+ can be configured as a secure co-processor. The dual core option enables Secure Firmware Install (SFI), secure Firmware Update(SFU), Secure Key Storage (SKS), and secure subGHz radio communication. The main features: • Multiprotocol LPWAN dual core 32-bit Arm Cortex-M4/M0+ MCUs LoRa, • (G)FSK, (G)MSK, BPSK, • up to 256KB Flash, • 64 KB SRAM The Cortex-M0+ processor is an entry-level 32-bit ARM Cortex	
ок	Cancel Defaults Help	·

с. 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.

е. 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.

-SYSRESETREQвыполняетпрограммныйсброс, устанавливаябитSYSRESETREQ. Ядро Cortex-M0 + и периферийные устройства на кристалле сбрасываются (единственный режим сброса, поддерживаемый Cortex-M0 +).

і. 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

W Options for Target 'Templates_CM0PLUS'										
Device Target Output Listing User C/C++ (AC6) As	m Linker Debug Utilities									
STMicroelectronics STM32WL55JClx:CM0P Xtal (MHz): 48.0	Code Generation ARM Compiler: Use default compiler version 6									
Operating system: None										
System Viewer File: 🔽 Use MicroLIB 🔽 Big Endian										
STM32WL5x_CM0P.svd										
Use Custom File										
Read/Only Memory Areas	Read/Write Memory Areas									
default off-chip Start Size Startup	default off-chip Start Size NoInit									
□ ROM1: 0	□ RAM1: □ □									
□ ROM2: ○	□ RAM2: □									
□ ROM3: □ C	RAM3:									
on-chip	on-chip									
IROM1: 0x8020000 0x20000 €	IRAM1: 0x20004000 0x4000 □									
IROM2:	IRAM2: 0x2000C000 0x4000 □									
OK Can	cel Defaults Help									

Figure 24. Debugger settings

Debug Trace Flash Download				
Debug Adapter Unit: ST-LINK/V3 Shareable ST-Link localhost 7184 Serial Number:	SW Dev SWDIO	vice IDCODE 0x6BA02477	Device Name ARM CoreSight SW-DP	Move Up Down
001700343038511234333935 Version: HW: V3 FW: V3J7M3 ✓ Check version on start Target Com Port: SW ▼ Clock Req: 4 MHz Selected: 3.300 MHz Performance Profile: High Perf ▼	C Mai	omatic Detection nual Configuration	ID CODE: Device Name: pdate IR len:	AP: 1
Debug Connect & Reset Options Connect: Normal ▼ Reset: Stop after Reset	REQ -	Cache Optio	Download Optio Code Memory	ns Download) Flash

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. Cl	M0+ Flash	loader	settings
---------------	-----------	--------	----------

Cortex-M Target Driver Setup				×
Debug Trace Flash Download				
Download Function C Erase Full Chip Erase Sectors C Do not Erase	 ✓ Program ✓ Verify ☐ Reset and Run 	RAM for / Start:	Algorithm 0x20004000 Size: 0x0000100	00
Programming Algorithm				
Description	Device Size	Device Type	Address Range	
STMS2WDX_CM0+ Hash	2306	On-chip Hash	000200001 - 0003FFFF	
		Start:	0x08020000 Size: 0x000400	00
	Add	Remove		
			ОК	Cancel Apply

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														
	File	Edi	t ۱	/iew	I	Proj	ect	Fla	sh	D	ebug	Per	riphera	als
		2	P.	ø		¥				5	6	+	\Rightarrow	7
	٢			1	•				Te	mpl	ates_(.M4		\sim

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.



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

ptions for node "Template	s_CM0PLUS"
Category: General Options Static Analysis Runtime Checking C/C++ Compiler Assembler Output Converter Custom Build Build Actions Linker Debugger Simulator CADI CMSIS DAP GDB Server I-jet J-Link/J-Trace TI Stellaris Nu-Link PE micro	Factory Settings Setup Download Images Multicore Extra Options Plugins ✓ Use command line options. Command line options: (one per line) -macro_param EnableM0PlusCore=1
ST-LINK Third-Party Driver TI MSP-FET TI XDS	
	OK Cancel

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 - М0 + 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

File	Edit	View	Proj	ject	ST-Link	Tools	Window	Help	_						
1	0			Add	Files				-	< 0	2 >	\$	►E	< 🖸	> [
Works	space			Add	Group							v 1	чx	main.c	x
Tem	lates (MOPLUS		Imp	ort File Lis	t							~	main()	1
remp	nates_c	MOI LOS		Add	Project C	onnectio	n			1			-	81	3
File	s			Edit	Continue	ations					۰	•	^	84	4
	Proje	ct		Luit	coningui	auons								85	5
He	I	emplat		Rem	nove				1		~			8	6
	-8	Applic		_					-				-	81	7
	H	EW.		Crea	ate New Pr	oject			-				-	81	5
H	-9			Add	Existing F	Project								9(0
	17			Ont	ions			AIT+F7	1				-	93	i d
H				opt					-				-	92	2
li i		L _E		Vers	ion Contr	ol Syster	n		•					93	3
II F	-🕀 💼	Doc												94	4
	-0 💼	Drivers	-	мак	(e			F7						9:	6
li r	- 📮 🛋	Output		Con	npile			CTRL+F7						9	7
	\vdash	Ten		Reb	uild All									98	в
		Ten		Clea	an								- 11	99	9
무		emplate		Bate	ch build			E8			~			100	D
		Applic			ch bana			10	-					10:	1 - 1
		EVV.		C-ST	TAT Static A	nalysis		0	•				-	102	2
	L	-@ 🔳 C	B	Stor	o Build			CTRL+ATTN					~	104	4
Ove	rview	Templat		_					-					< 101	-
1			O	Dov	vnload an	d Debug	1	CTRL+D							_
Build			٠	Deb	ug witho	ut Down	loading								
	Mess	ages		Atta	ich to Run	ning Tar	get								
L '	Build	ing con	G	Mak	ke & Resta	rt Debug	gger	CTRL+R	s C	MOPL	US				
	Upda	ating bu	C	Rest	tart Debug	gger	CTI	RL+MAJ+R	-						
Cantonnatio			Dov	vnload					Dowr	nload	activ	e app	lication		
	Conti	guratio		SFR	Setup					Down	nload	file			
				CMS	SIS-Pack M	lanager			-	LIUSC	mem	.,,			

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. С2ВООТ 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 0х0000 8000 в PWR_CR4 (по адресу 0х5800 040С).

Figure 32. Enable CPU2 using MDK-ARM

Memory 1							
Address: 0x5800040C							
0x5800040C:	0008000	00000000	000001F7	00000000			
0x58000430:	00000000	00000000	00000000	00000000			
0x58000454:	00000000	00000000	00000000	00000000			

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.



При попытке проделать ту же манипуляцию перед настройкой С2ВООТ 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] х 0х0800), 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] х 0х0400) 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] х 0х0400),

AN5556

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] = 0х40 в FLASH_SFR для защиты второй половины области флэшпамяти.

• 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

Pro STN	132CubeProgrammer				– 🗆 ×
STM32 Cube	Programmer			🚳 f	🖪 🎽 🔆 🏹
	Option bytes				🔵 Connected
	Name	Value	Description	ST-UNK	* Disconnect
	ESE	\checkmark	Unchecked : Security disabled Checked : Security enabled		Contract
I.					ST-LINK configuration
	<			Senal number	001700343038511234333 * 5
OB	PCROP Protection			Port	SWD +
	Write Protection			Frequency (kHz)	12000 *
CPU	Security Configurati	on Option bytes	Description	Mode	Normal
=	Name	value	This bit can only be accessed by software when HDPADIS = 0	Access port	
swv	SFSA	0x40	When FSD=0: system and Flash secure SFSA[6:0] contain the start address of the first 2 kB page of the secure Flash area.	Reset mode	
	FSD		Unchecked : System and Flash secure. This bit can only be accessed when HDPADIS = 0 Checked : System and Flash non-secure. This bit can only be accessed when HDPADIS = 0	Shared	Software reset
_	DDS		Unchecked : CPU2 debug access enabled (when also enabled by C2SWDBGEN)	External loader	Calce
		0-76	HDPSA[6:0] contain the start address of the first 2 kB page of the User Flash hide protection area	Target voltage	
	HDPSA	0.71	This bit field can only be accessed by software when HDPADIS = 0 When FSD=0 and HDPAD = 0: User Flash hide protection area enabled.	Firmware version	
	HDPAD		User Flash hide protection area disabled This bit can only be accessed by software when HDPADIS = 0		
			Unchecked : User Flash hide protection area access enabled. Checked : User Flash hide protection area access disabled.		
	SUBGHSPISD		SPI3 security disable This bit can only be accessed by software when HDPADIS = 0 FSD=1: SPI3 security is disabled		
			Unchecked : FSD=0 and SUBGHSPISD=0: SPI3 security enabled Checked : FSD=0 and SUBGHSPISD=1: SPI3 security disabled		
	C2OPT	Image: A start and a start	Unchecked : SBRV will address SRAM1 or SRAM2, from start address 0x2000 0000 + SBRV. Checked : SBRV will address Flash memory, from start address 0x0800 0000 + SBRV.		
	NBRSD		Unchecked : SRAM1 is secure if FSD=0 and non-secure otherwise. This bit can only be accessed when F Checked : SRAM1 is non-secure if FSD=0 and secure otherwise. This bit can only be accessed when F		
	SNBRSA	0x10	SNBRSA[4:0] contain the start address of the first 1 kB page of the secure "non-backup" SRAM1 area.		
	BRSD		Unchecked : SRAM2 is secure if FSD=0 and non-secure otherwise. This bit can only be accessed when F Checked : SRAM2 is non-secure if FSD=0 and secure otherwise. This bit can only be accessed when F		
	SBRSA	0x10	SBRSA[4:0] contain the start address of the first 1 kB page of the secure backup SRAM2 area.		
			Apply Read		
	Log		Verbosity level 💿 1 💿 2 💿 3	Reard	Target information
	17:54:15 : ST-LI 17:54:15 : Board	NK FW : V3J7M3 : NUCLEO-WL55JC	1 🖈	Device	STM32WLxx
$\stackrel{\sim}{\sim}$	17:54:15 : Volta 17:54:15 : SWD f	ge : 3.28V req : 12000 KHz		Туре	MCU
${ }$	17:54:15 : Conne 17:54:15 : Reset 17:54:16 : Devic	mode : Software reset e ID : 0x497		Device ID Revision ID	0x497
0	17:54:16 : Revis	ion ID :	v	Flash size	256 KB
\odot				Cool -	Cortex-M0+/M4

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 СМ4 и убедитесь, что все параметры проекта настроены, как описано в Разделе 3.1.1.

3. Откройте проект СМ0 + 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

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 флэшпамяти выполняется с помощью интерфейса JTAG.

Figure 37. Select the secure flashloader

Open		×
\leftarrow \rightarrow \checkmark \uparrow \square \ll config $>$	flashloader > ST v 💆 Search ST	Q
Organize 👻 New folder		= • 🔳 🕐
 Documents Pictures Draft_nb_3 licence issue Pictures STM32G0_Documents OneDrive - STMicroelectronic This PC 	 Name FlashSTM32WB55xE.board FlashSTM32WB55xG.board FlashSTM32WB55xY.board FlashSTM32WLSEC.board FlashSTM32WLxxx8.board FlashSTM32WLxxx8_M4.board FlashSTM32WLxxx8_M4.board FlashSTM32WLxxx8_M4.board FlashSTM32WLxxx8_M4.board FlashSTM32WLxxx8_M4.board FlashSTM32WLxxx8_M4.board 	Date modified ^ 9/1/2020 1:30 P 9/1/2020 1:30 P 9/1/2020 1:30 P 10/22/2020 10:2 9/1/2020 1:30 P 5/7/2020 5:24 P 9/1/2020 1:30 P 5/7/2020 5:25 P 9/1/2020 1:30 P 10/22/2020 1:30 P
 3D Objects Desktop Documents File name: File 	FlashSTM32WLxxxC_M0+.board FlashSTM32WLxxxC_M4.board < < eshSTM32WL_SEC.board Open	5/7/2020 5:20 P 5/7/2020 5:21 P ard) ~
		.:

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 СМ4 и убедитесь, что все параметры проекта настроены, как описано в Разделе 4.1.1.

3. Откройте проект СМ0 + 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 флэшпамяти выполняется с помощью интерфейса JTAG.

Figure 38. Change the flashloader execution address

Cortex-M Target Driver Setup	×
Debug Trace Flash Download	
Download Function C Erase Full Chip Program Erase Sectors Verify Start: 0x20008000 Do not Erase Reset and Run Start: 0x20008000	
Programming Algorithm	
Description Device Size Device Type Address Range	
Start: 0x08020000 Size: 0x00040000	
Add Remove	
OK Cancel Apply	