

AN5564

Application note

Getting started with projects based on dual-core STM32WL microcontrollers in STM32CubeIDE

Introduction

This application note describes how to get started with projects based on STM32WL Series dual-core microcontrollers in the STMicroelectronics STM32CubeIDE integrated development environment.





1 General information

STM32CubeIDE supports STM32 32-bit products based on the Arm[®] Cortex[®] processor.

Note: Arm is a registered trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere.

arm

1.1 Prerequisites

The following tools are prerequisites for understanding the tutorial in this document and developing an application based on the STM32WL Series:

- STM32CubeIDE 1.5.0 or newer
- STM32Cube_FW_WL_V1.0.0 or newer
- STM32CubeMX 6.1.0 or newer

Users are advised to keep updated with the documentation evolution of the STM32WL Series at www.st.com/en/microcontrollers-microprocessors/stm32wl-series.html.

1.2 The use cases in this document

In the STM32CubeIDE context, users have different ways to explore and get started with the development of projects based on the STM32WL Series. From the list below, select the description that best fits the use case considered and refer to the corresponding section in this application note:

- Create an STM32CubeMX project using the STM32CubeMX tool integrated inside STM32CubeIDE, or the stand-alone STM32CubeMX tool
- Import an STM32CubeIDE project from the STM32CubeWL MCU Package to learn by using an example project

1.3 Specific features of dual-core microcontrollers in the STM32WL Series

- Advanced security use cases (such as security area, isolated and protected code and secrets on locked Cortex[®]-M0+ side, and others)
- Applicative flexibility with both cores opened for developers
- Real time capabilities for standard sub-GHz stacks or proprietary protocols when required, with a dedicated core

1.3.1 Dual-core STM32WL project structure

When a dual-core STM32WL project is created, its structure is automatically made hierarchical. The project structure for single-core projects is flat. When the user creates or imports a dual-core STM32WL project, it consists of one root project together with sub-projects, referred to as MCU projects, for each core.

The MCU projects are real CDT[™] projects that can contain both build and debug configurations, as opposed to the root project, which is a simple container allowing common code sharing between the cores. The root project can contain neither build nor debug configurations.

If the project is not shown in a hierarchical structure, this can be changed as shown in Figure 1.

IDE workspace - STM32CubeIDE					
File Edit Source Refactor Navigate	Se	arch Project Run Window	/ Н	elp	
📑 • 🗟 🕼 🗞 • 🗞 • 🔯 • 6	*	₢ - ₢ - ₩ - ० - 0	-	3	x @ 🕭 🔗 🕇 🖻 🛛
Project Explorer 🛛 📃 🛱	000				
		Projects Presentation	>		Flat
		Top Level Elements	>	•	Hierarchical
		Select Working Set Deselect Working Set Edit Active Working Set			
	රා	1 Window Working Set			
	7	Filters and Customization			
	€\$	Link with Editor			

Figure 1. Setting the project hierarchical view

2 Create and import projects

This chapter describes how to create or import projects for dual-core microcontrollers in the STM32WL Series.

2.1 Create a new STM32 project

To start a new project, go to [File]>[New]>[STM32 Project] as shown in Figure 2.

Figure	2. New	STM32	pro	ject

IDE	workspace -	STM3	2CubelDE	E					
File	Edit Sou	ce F	efactor	Navigate	Search	Project	Ru	n Window Help	
	New				Alt+Sh	ift+N >	C#	Makefile Project with Existing Code	
	Open File						C	C/C++ Project	
	Open Proje	ts fro	m File Sy	stem			IDE	STM32 Project	
	Recent Files					>	МХ	STM32 Project from an Existing STM32CubeMX Configuration File (.ioc)	
	Close Edito				C	trl+W	Ċ	Project	
	Close All Ec	itors			Ctrl+Sh	ift+W	62	Source Folder	
	Save				(Ctrl+S	ĊŶ	Folder	
	Save As						C	Source File	
	Save All				Ctrl+Sł	nift+S	h	Header File	
	Revert						Ċ	File from Template	
	Move						C	Class	
	Rename					F2		Other	Ctrl+N



Select the desired MCU or board. In the example shown in Figure 3, the selected board is the NUCLEO-WL55JC2. Click on [Next >].

STM32 Project								_	• ×
Target Selection									IDE
Select STM32 targ	et or STM32Cube example								
MCU/MPU S	elector Board Selector Ex	ample Selector	r Cross Selector						
Board Filter	s								
			Features	Large Picture	Docs & Resource	es	Datasheet	Buy	
Comme Part Nu	mber NUCLEO-WL55JC2	~	STM32WL Series						
Vendor		>	NUCLEO-	WL55JC2	STMicroelectronics Examples - Low Fre	NUCLEO-WL55 quency bands	JC2 Board Sup	port and	
Туре		>	Not available		Part Number : NA		Unit Price (US\$): NA	N Contraction of the second seco	
MCU/MF	PU Series	>	Please contact your sale interested by this product	s representative if	Commercial Part Number : N	UCLEO-WL55JC2	Mounted Device : ST	M32WL55JCIx	
Other		>			#				-
Periphera	al	B	HB1389B		Intercial Part. Type	Marketing Sta	us X Unit Price (tures STMicroelectron Morpho connect (2 × 38) STMicroelectron Arduino connect 10 + (2 × 8) + 6 Buttons: B1, B2 B3 User button E US) K Mounted D STM32WL5SIC	iics or iics toor 3 8 &
?						< <u>B</u> ack	<u>N</u> ext >	Einish	Cancel

Figure 3. Board selector

After the target selection comes the project setup step shown in Figure 4. The *Targeted Project Type* setting determines whether the project gets generated by STM32CubeMX or not. An *Empty* project is a skeleton of a project that needs building upon while *STM32Cube* indicates an STM32CubeMX-managed project.

- Empty projects contain the bare-minimum code to build an debug an empty main ().
- STM32Cube projects are managed by STM32CubeMX. Drivers and middleware are generated in the project based on the configurations done in the .ioc file editor in STM32CubeIDE (the .ioc file editor is the integrated version of STM32CubeMX).

IDE STM32 Proje	ct —		×
Setup STM32 pr	oject		DE
Project			
Project Name:	NUCLEO_WL55JC2		
🗹 Use default	location		
Location:	C:/Users/ · · · /STM32CubelDE/workspace_1.5.0.20w40	Brow	vse
Options Targeted La	inguage + +		
Targeted D	evice Usage Multi Cpus Configuration		
Targeted Bi	nary Type		
Executat	ole OStatic Library		
Targeted Pr STM32C	oject Type ube O Empty		
0	c Pack Next > Einich	Con-	al
\mathcal{O}	< Back Next > Finish	Canc	el

Figure	4	Projet setu	n
Iguie	- T. S.	i i ojet setu	μ

Note: Select the [**Enable Multi Cpus Configuration**] option to allow the creation of multicore project with both the Cortex[®]-M0+ and Cortex[®]-M4. Unselect this option to create a project with the Cortex[®]-M4 only.

(7/

2.2 Import a project from the STM32CubeWL MCU Package

To import the STM32Cube firmware project into STM32CubeIDE, go to [File]>[New]>[STM32 Project] as shown in Figure 5 and select the desired example.

Division Selector Examples and Selector Examples	nple Selector Cros	ss Selector						
Board	>				Features			
MCU / MPU	~							
Mana	. *							
Name								
Series	Example	les List: 230	tems					📤 Export
Check/Upshock All		*	Name 🗢	Board	× Board Type	× Series	X Project Type	× Driver
Check/Oncheck All		\$	ADC AnalogWatc	NUCLEO-WL55JC1	Nucleo-64	STM32WL	Example	HAL
STM32F4		\$	ADC_AnalogWatc	NUCLEO-WL55JC1	Nucleo-64	STM32WL	Example	LL
STM32F7		\$	ADC_Continuous	NUCLEO-WL55JC1	Nucleo-64	STM32WL	Example	LL
STM32G0		\$	ADC_Continuous	NUCLEO-WL55JC1	Nucleo-64	STM32WL	Example	LL
STM32G4		\$ A	ADC_MultiChann	NUCLEO-WL55JC1	Nucleo-64	STM32WL	Example	HAL
		\$	ADC_Oversampling	NUCLEO-WL55JC1	Nucleo-64	STM32WL	Example	HAL
		\$	ADC_Oversamplin	NUCLEO-WL55JC1	Nucleo-64	STM32WL	Example	LL
STM32L5		\$	ADC_SingleConve	NUCLEO-WL55JC1	Nucleo-64	STM32WL	Example	LL
STM32WB		\$	ADC_SingleConve	NUCLEO-WL55JC1	Nucleo-64	STM32WL	Example	LL
STM32WL		☆ /	ADC_SingleConve	NUCLEO-WL55JC1	Nucleo-64	STM32WL	Example	HAL
		\$	ADC_SingleConve	NUCLEO-WL55JC1	Nucleo-64	STM32WL	Example	MIX
		\$ ·	ADC_SingleConve	NUCLEO-WL55JC1	Nucleo-64	STM32WL	Example	LL
roject		☆ /	ADC_SingleConve	NUCLEO-WL55JC1	Nucleo-64	STM32WL	Example	HAL
		\$	ADC_SingleConve	NUCLEO-WL55JC1	Nucleo-64	STM32WL	Example	LL
Device Configuration Tool capable		\$ I	ADC_Temperatur	NUCLEO-WL55JC1	Nucleo-64	STM32WL	Example	LL
		~	AT Clave	1000000000000000	Nucleo 64	CT112011/1	Application	U AI

Figure 5. Project example selector

Note: Users can also import projects using the import mechanism by going to [File]>[Project...]>[Import...] and selecting [Existing Projects into Workspace].

3 Debugging

This chapter highlights some of the points to bear in mind while debugging a device in the STM32WL Series. In the next two sections, this application note covers the configurations needed to start debug sessions with ST-LINK GDB server and OpenOCD.

Note: By default, the Cortex[®]-M0+ is not available until the C2BOOT bit is set in power control register 4 (PWR_CR4). It is the user's responsibility to enable the C2BOOT bit through the application code running on the Cortex[®]-M4. If there is no security enabled on the Cortex[®]-M0+, STMicroelectronics recommends to use access port 0 (AP0) to program both CPUs.



3.1 Setting up with ST-LINK GDB server

To create a debug configuration using ST-LINK GDB server, perform the following steps:

- 1. Select the Cortex[®]-M4 project in the *Project Explorer* view
- 2. Right-click [Debug As...], select [Debug Configuration...], and then double-click on [STM32 Cortex-M C/C++ Application]

Figure 6. ST-LINK GDB server debug configuration (1 of 6)

Project Explorer	ß	E 🕏 7 🕴 🗆 🗖	.c m	ain.c 🛛		
 ✓ INTERPORT ✓ INTERPORT ✓ INTERPORT ✓ INTER ✓ INTER 	55JC	2 5JC2_CM0plus (in CM0PLUS)	1 2 3 4 5	/* US ⊝ /** *** * @ * @	ER CC	DDE BEGIN Header */ : main.c f : Main program body
> 🎇 Binar > 🔊 Inclu > 🕰 Comi		New Go Into			>	<pre>ion inter>©: Copyright (c) 2020 STMicroelectro</pre>
> 📇 Core > 📇 Drive > 👝 Debu		Open in New Window Show In		Alt+Shift	+W >	<pre>shts reserved. oftware component is licensed by ST under BSD</pre>
STM3		Copy Paste		Ctr Ctr	I+C +V	cense"; You may not use this file except in o . You may obtain a copy of the License at: opensource.org/licenses/BSD-3
> 🎇 Binar > 🔊 Inclue > 🖓 Com	K	Delete Source Move		De	lete >)E END Header */
✓ [™] Core > [™] In		Rename			F2	s nain.h"
∽ (<u>⇒</u> Sri È > [c] ⊨	y Y	Import Export				includes E BEGIN Includes */
> @ > @ > @ > @ Str > @ Drive > @ Debu	2	Build Project Clean Project Refresh Close Project Close Unrelated Projects			F5	<pre>>E END Includes */ typedef >E BEGIN PTD */ >E END PTD */</pre>
 NUCI STM3 mx.scrat NUCLEC 		Build Configurations Build Targets Index			> > >	ks
		Show in Remote Systems view Run As Debug As	1		>	1 STM22 Cortex-M C/C++ Application 77
<	κ.,	Profile As Team Compare With			> > >	Debug Configurations



Debugger tab - Cortex[®]-M4

To use the ST-LINK GDB server (refer to Figure 7), make sure that:

- The type of reset behavior is selected as Connect under reset
- The Halt all cores option is enabled
- The Shared ST-LINK is enabled

Figure 7. ST-LINK GDB	server debug	configuration	(2 c	of 6)
-----------------------	--------------	---------------	------	------	---

pe filter text © C/C++ Application © C/C++ Attach to Application © C/C++ Postmortem Debugger © C/C++ Remote Application © GDB Hardware Debugging ■ Launch Group ▶ Launch Group (Deprecated) © STM32 Cortex-M C/C++ Application © NUCLEO_WL55JC2_CM4 Debug	Main Startup Source Common GDB Connection Settings Autostart local GDB server Host name or IP address localhost Connect to remote GDB server Port number 61234 Debug probe ST-LINK (ST-LINK GDB server) ~ GDB Server Command Line Options Interface SWD JTAG ST-LINK S/N Scan Frequency (kHz): Auto Access port: 0 - Cortex-M4 Reset behaviour Type: Connect under reset ~ MHalt all cores Serial Wire Viewer (SWV) Enable Clock Settings Core Clock: 16.0 MHz SWO Clock: 2000 ~ kHz
 C //C++ Application C //C++ Attach to Application C //C++ Postmortem Debugger C //C++ Remote Application GDB Hardware Debugging Launch Group Launch Group (Deprecated) STM32 Cortex-M C/C++ Application TM2 Cortex-M C/C++ Application TM2 NUCLEO_WL55JC2_CM4 Debug 	GDB Connection Settings • Autostart local GDB server Host name or IP address localhost • Connect to remote GDB server Port number 61234 GDB Server Command Line Options Interface • SWD ST-LINK S/N Frequency (kHz): Auto Access port: • Connect under reset Serial Wire Viewer (SWV) Enable Clock Settings Core Clock: Swo Clock:
[€] C/C++ Attach to Application [€] C/C++ Remote Application [€] C/C++ Remote Application [€] GDB Hardware Debugging [] Launch Group [] Launch Group (Deprecated) [] STM32 Cortex-M C/C++ Application [] Inf [] NUCLEO_WL55JC2_CM4 Debug	
C C/C++ Remote Application C C/C++ Remote Application C GDB Hardware Debugging Launch Group D Launch Group (Deprecated) STM32 Cortex-M C/C++ Application INE NUCLEO_WL55JC2_CM4 Debug	○ Connect to remote GDB server Port number €1234 Debug probe ST-LINK (ST-LINK GDB server) ~ GDB Server Command Line Options Interface ● SWD O JTAG ST-LINK S/N ~ Frequency (kHz): Auto Access port: 0 - Cortex-M4 Reset behaviour Type: Type: Connect under reset Serial Wire Viewer (SWV) □ Enable Clock Settings Core Clock: 16.0 SWO Clock: 2000
☑ GDB Hardware Debugging □ ☑ Launch Group □ ☑ Launch Group (Deprecated) □ ☑ STM32 Cortex-M C/C++ Application □ □□ NUCLEO_WL55JC2_CM4 Debug □	Debug probe ST-LINK (ST-LINK GDB server) V GDB Server Command Line Options Interface SWD JTAG ST-LINK S/N Scan Frequency (kHz): Auto Access port: 0 - Cortex-M4 Reset behaviour Type: Connect under reset V Halt all cores Serial Wire Viewer (SWV) Enable Clock Settings Core Clock: 16.0 MHz SWO Clock: 2000 V kHz
Launch Group (Deprecated) STM32 Cortex-M C/C++ Application IDE NUCLEO_WL55JC2_CM4 Debug	GDB Server Command Line Options Interface SWD JTAG St-LINK S/N Scan Frequency (kHz): Auto Access port: 0 - Cortex-M4 Reset behaviour Type: Connect under reset Serial Wire Viewer (SWV) Enable Clock Settings Core Clock: 16.0 MHz SWO Clock: 2000 kHz
STM32 Cortex-M C/C++ Application	Interface SWD JTAG SSWD JTAG ST-LINK S/N Frequency (kHz): Auto Access port: 0 - Cortex-M4 Reset behaviour Type: Connect under reset Halt all cores Serial Wire Viewer (SWV) Enable Clock Settings Core Clock: 16.0 MHz SWO Clock: 2000 kHz
	SWD JTAG SWD SWD Scan Frequency (kHz): Auto Access port: 0 - Cortex-M4 Reset behaviour Type: Connect under reset Halt all cores Serial Wire Viewer (SWV) Enable Clock Settings Core Clock: 16.0 MHz SWO Clock: 2000 kHz
	St-LINK S/N Scan Frequency (kHz): Auto Access port: 0 - Cortex-M4 Reset behaviour Image: Connect under reset Type: Connect under reset Serial Wire Viewer (SWV) Image: Clock Settings Core Clock: 16.0 MHz SWO Clock: SWO Clock: 2000 KHz
	Frequency (kHz): Auto Access port: 0 - Cortex-M4 Reset behaviour Type: Connect under reset Serial Wire Viewer (SWV) Enable Clock Settings Core Clock: 16.0 MHz SWO Clock: 2000 KHz
	Access port: 0 - Cortex-M4 Reset behaviour Type: Connect under reset Serial Wire Viewer (SWV) Enable Clock Settings Core Clock: 16.0 MHz SWO Clock: 2000 KHz
	Reset behaviour Type: Connect under reset Serial Wire Viewer (SWV) Enable Clock Settings Core Clock: 16.0 MHz SWO Clock: 2000 kHz
	Type: Connect under reset Halt all cores Serial Wire Viewer (SWV) Enable Clock Settings Core Clock: 16.0 MHz SWO Clock: 2000 kHz
	Serial Wire Viewer (SWV) Enable Clock Settings Core Clock: 16.0 MHz SWO Clock: 2000 KHz
	Serial Wire Viewer (SWV) Enable Clock Settings Core Clock: 16.0 MHz SWO Clock: 2000 KHz
	Clock Settings Core Clock: 16.0 MHz SWO Clock: 2000 V kHz
	Core Clock: 16.0 MHz SWO Clock: 2000 v kHz
	SWO Clock: 2000 V kHz
	Port number: 01230
	Device settings
	Complexity in low power modes:
	Suspend watchdog counters while nated: No configuration
	Allow other cores to halt this core Signal halt events to other cores
	Misc
	☐ Enable live expressions
	Log to file: C:\Users\vionf\STM32CubelDE\workspace_1.5.0.20w40\NUCLI Browse
	External Loader:
	Shared ST-LINK
	Max halt timeout(s): 2



Startup tab - Cortex[®]-M4

The Cortex[®]-M4 debug configuration is responsible for loading both the Cortex[®]-M4 and Cortex[®]-M0+ images. Go to the *Startup* tab to set this up as shown in Figure 8.

Debug Configurations					
Create, manage, and run configurations					1
					No.
Image: specific constraints Image: specif	Name: NUCLEO-WL55JC2_CM4 Debug				
 C/C++ Postmortem Debugger C/C++ Remote Application GDB Hardware Debugging Launch Group 	Load Image and Symbols				~
Launch Group (Deprecated) IDE STM32 Cortex-M C/C++ Application	File	Build	Download	Load symbols	Add
IDE NUCLEO-WL55JC2_CM0plus Debug	Debug/NUCLEO-WL55JC2_CM0plus.elf [NUCLEO-WL55JC2_CM0plus] Debug\NUCLEO-WL55JC2_CM4.elf [NUCLEO-WL55JC2_CM4]	V true See Main tab	🖌 true 🖌 true	v true v true	Edit
					Remove Move up
	<			>	Move down
	Runtime Options Start Address Default start address Set program counter (hex): Specify vector table (hex): Set breakpoint at: main Exception on divide by zero Exception on unaligned access Halt on exception Resume Run Commands				
Filter matched 10 of 10 items				Re <u>v</u> ert	Apply
?				<u>D</u> ebug	Close

Figure 8. ST-LINK GDB server debug configuration (3 of 6)

To also download the Cortex[®]-M4 image, click on [Add...], point to the right project and build the configuration. The result is shown in Figure 9.

DE Add/Edit item	— 🗆 X							
Project:	NUCLEO-WL55JC2_CM0plus ~	·						
Build configuration:	Debug ~	•						
Program path:	Debug/NUCLEO-WL55JC2_CM0plus.elf]						
	File system							
Perform build								
🗹 Download								
Use download offset (hex)								
Load symbols		-						
Use symbol address	; (hex)							
-	OK Cancel							

Figure 9. ST-LINK GDB server debug configuration (4 of 6)

The order in the load list is very important. The debugger sets the start of execution using the entry point of the last loaded image in this list. In practice, this means that the program counter for the Cortex[®]-M4 is set to the location of the Reset Handler of the Cortex[®]-M4 binary. This is indicated by the green arrow.

Note: It is not necessary to load symbols for the Cortex[®]-M0+ in the debug configuration of the Cortex[®]-M4 because they are loaded in the debug configuration of the Cortex[®]-M0+.

These steps conclude the debug configuration for the Cortex[®]-M4. The next steps present the creation of a debug configuration for the Cortex[®]-M0+ based on the Cortex[®]-M0+ project.



Debugger tab - Cortex[®]-M0+

Contrary to the Cortex[®]-M4, as shown in Figure 10:

- Make sure that the *Port number* exceeds the value of the previous debug configuration by at least 3 (61238 in this example)
- Select 1 Cortex-M0+ for [Access port]
- Select *None* for [**Reset behaviour**]
- Select Shared ST-LINK

Figure 10. ST-LINK GDB server debug configuration (5 of 6)

Debug Configurations	- D X
Create, manage, and run configurations	the second se
📑 🖻 🐢 🗎 🗶 📄 🍸 🗸	Name: NUCLEO_WL55JC2_CM0plus Debug
type filter text	📄 Main 🏇 Debugger 🌘 Startup 🦆 Source 🔲 Common
C/C++ Application	GDB Connection Settings
C/C++ Attach to Application	Autostart local GDB server Host name or IP address localhost
C/C++ Remote Application	Connect to remote GDB server Port number 61238
C GDB Hardware Debugging	Debug probe ST-LINK (ST-LINK GDB server) 🗸
Launch Group (Deprecated)	GDB Server Command Line Options
✓ IDE STM32 Cortex-M C/C++ Application IDE NUCLEO WL55JC2 CM0plus Debug	Interface
IDE NUCLEO_WL55JC2_CM4 Debug	● SWD () JTAG
	Scan
	Frequency (kHz): Auto
	Access port: 1 - Cortex-M0plus v
	Reset behaviour
	Type: None V
	Serial Wire Viewer (SWV)
	Enable
	Clock Settings
	Core Clock: 10.0 MHz
	SWO Clock: 2000 V KHZ
	Port number: 61235
	Wait for sync packet
	Device settings
	Debug in low power modes: No configuration
	Suspend watchdog counters while halted: No configuration
	Cross Ingger Interface (CII) Allow other cores to halt this core Signal halt events to other cores
	Misc
	☑ Verify flash download
	C Enable live expressions
	Log to file: C:\Users\vionf\STM32CubelDE\workspace_1.5.0.20w40\NUCLEO_WL55 Browse
	External Loader:
	Shared ST-LINK
	Max halt timeout(s): 2
Filter matched 10 of 10 items	Revert Apply
?	Debug Close



Startup tab - Cortex[®]-M0+

(7/

Go to the *Startup* tab and select [**Edit...**]>[**Disable Download**]. This is required since the download is already performed by the Cortex[®]-M4 configuration.

_								
Debug Configurations					- 0 X			
Create, manage, and run configurations					The second			
					200			
🗋 🖻 🔯 🗎 🗶 🖻 🏹 🗸	Name: NUCLEO-WL55JC2_CM0plus Debug							
type filter text	Main 🕸 Debugger 🕟 Startup 🦆 Source 🔳 Common							
C/C++ Application	Initialization Commands							
C/C++ Attach to Application					^			
C C/C++ Remote Application								
C GDB Hardware Debugging								
Launch Group (Deprecated)	Load Image and Symbols							
✓ IDE STM32 Cortex-M C/C++ Application IDE NUICEE0WI 55IC2 CM0plus Debug	File	Build	Download	Load symbols	Add			
IDE NUCLEO-WL55JC2_CM4 Debug	Debug\NUCLEO-WL35JC2_CMUplus.eff [NUCLEO-WL35JC2.	See Main tab	😻 false	V true	Edit			
					Remove			
					Move up			
					Move down			
	Runtime Options							
	Start Address							
	Set program counter (bev):							
	Specify vector table (hex):							
	Set breakpoint at: main							
	Exception on divide by zero Exception on upaligned access							
	Resume							
	Run Commands							
					~			
					Y			
٢								
Filter matched 10 of 10 items				Re <u>v</u> ert	Apply			
				<u>D</u> ebug	Close			

Figure 11. ST-LINK GDB server debug configuration (6 of 6)

The configuration is complete.

Note:

- It is possible to program the $Cortex^{\ensuremath{\mathbb{R}}}$ -M0+ through access port 1:
- Delete the binary of the Cortex[®]-M0+ from the list in the Startup tab of the Cortex[®]-M4 debug configuration
 - Enable the download of the binary on the Startup tab of the Cortex[®]-M0+ debug configuration

Flash programming via access port 1 in "hot-plug" (Cortex[®]-M0+) fails if the existing application code on the $Cortex^{®}$ -M0+ enables interrupts.

- 1. Launch the Cortex[®]-M4 configuration to download both the Cortex[®]-M0+ and Cortex[®]-M4 images.
- Resume the Cortex[®]-M4 core until the C2BOOT bit is set in power control register 4 (PWR_CR4) to enable the Cortex[®]-M0+.
- 3. Launch the Cortex[®]-M0+ configuration using the arrow next to the debug icon. The Cortex[®]-M0+ is in the running mode and the user can halt it after the debugger is started.

Figure 12. ST-LINK GDB server debug configuration launch



Note: After creating the debug configurations for both cores, they are not shown in the scroll-down menu if they have never been launched before. This is because the arrow provides access to the history of latest launches, with a grayed "no history" message if there are none. First-time debug launch must be done through the "Debug Configurations..." wizard.

3.1.2 Cross-trigger Interface

The cross-trigger interface is used to send halt signals from one core to the other. To enable the Cortex[®]-M0+ to halt the Cortex[®]-M4, apply the following configuration:

- In the Cortex[®]-M0+ debug configuration: select Signal halt events to other cores
- In the Cortex[®]-M4 debug configuration: select Allow other cores to halt this core

Figure 13. ST-LINK GDB server debug cross-trigger interface

Cross Trigger Interface (CTI)

Allow other cores to halt this core Signal halt events to other cores

Note: Checking both checkboxes in both debug configurations enables both cores to halt each other.

3.2 Setting up with OpenOCD

To create a debug configuration using OpenOCD, perform the following steps:

- 1. Select the Cortex[®]-M4 project in the *Project Explorer* view
- Right-click [Debug As...], select [Debug Configuration...], and then double-click on [STM32 Cortex-M C/C++ Application]

Figure 14. OpenOCD debug configuration (1 of 3)

陷 Project Explorer	X	🖻 😫 🍸 🕴 🗖	c ma	in.c 🛙		
 ✓ INVELEO_WL55JC2 > ➢ Common > ➢ Drivers ✓ INVELEO_WL55JC2_CM0plus (in CM0PLUS) 		1 26 3 4 5	/* US > /** * @ * @	ER CO ***** file brief	DDE BEGIN Header */ : main.c : Main program body	
> 🐝 Binar > 🔊 Inclu > 🚜 Comi		New Go Into		>		tion
> 🔑 Core > 😕 Drive > 冯 Debu		Open in New Window Show In	Alt+Shift+W >		•w >	<pre>inter>© Copyright (c) 2020 STMicroelectr shts reserved. offware component is licensed by ST under RSD</pre>
	D D	Copy Paste	Ctrl+C Ctrl+V Ctrl+V			cense"; You may not use this file except in o . You may obtain a copy of the License at: opensource.org/licenses/BSD-:
> 🐝 Binar > 🔊 Inclu > 🖓 Com	×	Delete Source Move		Del	ete >)E END Header */
✓ [™] Core > [™] In ✓ [™] Sr		Rename Import			F2	ain.h"
> @ > @		Export Build Project				DE END Includes */
> ₪ > ₪ > ₪ > ⇒ Sti > ∰ Drive > @ Debu		Clean Project Refresh Close Project Close Unrelated Projects			F5	typedef DE BEGIN PTD */ DE END PTD */
NUCL STM3 mx.scrat NUCLEC		Build Configurations Build Targets Index			> > >	ks
	D	Show in Remote Systems view Run As	1		>	^
3	*	Debug As Profile As			>	DE 1 STM32 Cortex-M C/C++ Application 7 Debug Configurations
<		Team Compare With			>	

Debugger tab - Cortex®-M4

Select *ST-LINK (OpenOCD)* as the [**Debug probe**]. Select *Autostart local GDB server* for the configuration that launches first, which is the Cortex[®]-M4 in the example in Figure 15.

Set all the default options and verify that:

- Connect under reset is selected as [Reset Mode].
- [Shared ST-LINK] is selected; this option is mandatory to run the multicore target.

Figure 15. OpenOCD debug configuration (2 of 3)

Debug Configurations	- D X
Create, manage, and run configurations	
📑 🖻 💫 🗎 🗙 🗉 🍸 🗸	Name: NUCLEO-WL55JC2 CM4 Debug
Filter matched 10 of 10 items	Name: NUCLEO-WL53/C2_CM4 Debug Main % Debugger Startup GDB Connection Setup Main (% Debugger) Startup Main % Autostati local GDB server Host name or IP address localhost O Connection Setup 3333
	Serial Wire Viewer (SWV) Enable Core Clock: 16.0 MHz SWO Clock: 2000 KHz Port number: 3344 Wait for sync packet Cross Trigger Interface (CTI) Allow other cores to halt this core Signal halt events to other cores ST-LINK Client Setup Shared ST-LINK Boards detected by ST-Link Server: 1 Name Blink Mame Blink Refresh Refresh Refresh Refresh Refresh
?	Debug Close

Create the debug configuration for the other core, which is the Cortex[®]-M0+ in the example in Figure 16:

- Select ST-LINK (OpenOCD) as the [Debug probe]
- Select Autostart local GDB server as default
- Make sure that the *Port number* exceeds the value of the previous debug configuration by at least 2 (3335 in this example)
- Open [Generator Options] and select None as [Reset Mode]

Figure 16. OpenOCD debug configuration (3 of 3)

Debug Configurations		_	o x
Create, manage, and run configurations			Ť.
📑 🖻 🗫 📄 🗙 📄 🍸 🗝	Name: NUCLEO-WL55JC2_CM0plus Debug		
Image: Content of the image: Conten	Name: NUCLEO-WL55JC2_CMOplus Debug Main Statup Source Common GDB Connection Settings Autostart local GDB server Host name or IP address Oconnect to remote GDB server Port number 3335 Debug probe ST-LINK (OpenOCD) GDB Server Command Line Options OpenOCD Setup OpenOCD Command: "S(stm32cubeide_openocd_path)\openocd_exe" OpenOCD Options : Configuration Script Browse OpenOCD Options : Environment Hide ge Script File: \${ProjDirPath}\NUCLEO-WL55JC2_CM0plus Debug.cfg Browse Connection Setup Mode Setup Reset Mode: None Interface: Swd Word Setup	enerator options a Reload	
	Frequency: 8 MHz Image: Step watchdog counters when halt Serial Wire Viewer (SWV) Enable Clock Settings Core Clock: 16.0 WHz Port number: 3344 Wait for sync packet Cross Trigger Interface (CTI) Allow other cores to halt this core ST-LINK Client Setup Shared ST-LINK Boards detected by ST-Link Server: 1 Name Ø 066CFF504955707267243858		×
Filter matched 10 of 10 items		e <u>v</u> ent A	pply
()		<u>D</u> ebug	Close

The configuration of the *Startup* tab is the same as with the ST-LINK GDB server probe for both debug configurations (refer to *Startup* tab - Cortex[®]-M4 and *Debugger* tab - Cortex[®]-M0+ in Section 3.1 Setting up with ST-LINK GDB server).

To launch the debug and enable the cross-trigger interface, refer to Section 3.1.1 Launching the configurations and Section 3.1.2 Cross-trigger Interface.

4 Limitations

57/

STM32CubeIDE is subject to the limitations listed below:

- The Flash memory of the Cortex[®]-M0+ through access port 1 (AP1) is not working using STM32CubeIDE with the OpenOCD probe. There is also a limitation using the ST-LINK GDB server if the existing application code on the Cortex[®]-M0+ enables interrupts. To program through AP1, it is recommended to use the external tool STM32CubeProgrammer (STM32CubeProg) version 2.6.0 or above. After programming, refer to Section 3 to create the debug configurations without programming the Cortex[®]-M0+.
- When the security is enabled on the Cortex[®]-M0+, it is also recommended to use the external tool STM32CubeProgrammer to program the Cortex[®]-M0+ using OpenOCD.

	Behaviour on	STM32CubeIDE probe		
Initial conditions	Cortex [®] -M0+ using AP1	ST-LINK GDB server	OpenOCD	
The whole Flash memory is empty	Download only	Use STM32CubeProgrammer standalone ⁽¹⁾	Use STM32CubeProgrammer standalone ⁽¹⁾	
 Application code already loaded for the Cortex[®]-M4 The Flash memory is empty for the Cortex[®]-M0+ 	Download only	Use theCortex [®] -M0+ debug configuration to program the Flash memory	Use STM32CubeProgrammer standalone ⁽¹⁾	
Application code already loaded for both Cortex®-Mx	Debug only	Refer to Section 3	Refer to Section 3	
The Flash memory security sector is enabled and the system is in secure mode for the Cortex [®] -M0+	Download only	Use theCortex [®] -M0+ debug configuration to program the Flash memory	Use STM32CubeProgrammer standalone ⁽¹⁾	
The Flash memory security sector is enabled and the system is in secure mode for the $\mbox{Cortex}^{\ensuremath{\mathbb{R}}}\mbox{-}\mbox{M0+}$	Debug only	Refer to Section 3	Refer to Section 3	

Table 1. Programming/debugging status using access port 1 (AP1)

1. STM32CubeProgrammer (STM32CubeProg) is available from www.st.com

Revision history

Date	Version	Changes
17-Nov-2020	1	Initial release.
14-Dec-2020	2	Added and updated the Shared ST-LINK configuration in Debugger tab - Cortex [®] -M4 and Debugger tab - Cortex [®] -M0+.

Table 2. Document revision history



Contents

Gene	ral info	prmation	2
1.1	Prerequ	uisites	2
1.2	The us	e cases in this document	2
1.3	Specific	c features of dual-core microcontrollers in the STM32WL Series	2
	1.3.1	Dual-core STM32WL project structure	2
Creat	te and i	mport projects	4
2.1	Create	a new STM32 project	4
2.2	Import	a project from the STM32CubeWL MCU Package	7
Debu	gging		8
3.1	Setting	up with ST-LINK GDB server	9
	3.1.1	Launching the configurations	15
	3.1.2	Cross-trigger Interface	15
3.2	Setting	up with OpenOCD	16
Limit	ations		20
ision ł	nistory		21
tents			22
of tab	les		23
of figu	ures		24
	Gene 1.1 1.2 1.3 Creat 2.1 2.2 Debu 3.1 3.2 Limit ision H tents of tab of figu	General info 1.1 Prereq 1.2 The us 1.3 Specifie 1.3.1 Create and i 2.1 Create 2.2 Import Debugging 3.1 Setting 3.1.1 3.1.2 3.2 Setting Limitations ision history tents of tables of figures	General information 1.1 Prerequisites 1.2 The use cases in this document 1.3 Specific features of dual-core microcontrollers in the STM32WL Series 1.3.1 Dual-core STM32WL project structure Create and import projects

List of tables

Table 1.	Programming/debugging status using access port 1 (AP1)	20
Table 2.	Document revision history	21

List of figures

Figure 1.	Setting the project hierarchical view.	. 3
Figure 2.	New STM32 project.	. 4
Figure 3.	Board selector	. 5
Figure 4.	Projet setup	. 6
Figure 5.	Project example selector	. 7
Figure 6.	ST-LINK GDB server debug configuration (1 of 6)	. 9
Figure 7.	ST-LINK GDB server debug configuration (2 of 6)	10
Figure 8.	ST-LINK GDB server debug configuration (3 of 6)	11
Figure 9.	ST-LINK GDB server debug configuration (4 of 6)	12
Figure 10.	ST-LINK GDB server debug configuration (5 of 6)	13
Figure 11.	ST-LINK GDB server debug configuration (6 of 6)	14
Figure 12.	ST-LINK GDB server debug configuration launch	15
Figure 13.	ST-LINK GDB server debug cross-trigger interface	15
Figure 14.	OpenOCD debug configuration (1 of 3)	16
Figure 15.	OpenOCD debug configuration (2 of 3)	17
Figure 16.	OpenOCD debug configuration (3 of 3)	18

IMPORTANT NOTICE - PLEASE READ CAREFULLY

STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST's terms and conditions of sale in place at the time of order acknowledgement.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of Purchasers' products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

ST and the ST logo are trademarks of ST. For additional information about ST trademarks, please refer to www.st.com/trademarks. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2020 STMicroelectronics - All rights reserved