

T.O.M.A.S Team







 After successful code generation by STM32CubeMX this is the right time to import it into SW4STM32 toolchain for further processing





Handling the project in SW4STM32





Our goals for this session

□Handling the projects generated by STM32CubeMX in SW4STM32

- □Import project generated by STM32CubeMX
- Tune sources to run selected peripherals in desired algorithm
- Build project
- □Configure debug session
- □Run debug session
- Debug perspective
- □Watching the variables and registers content
- □Handling errors





Create a new workspace SW4STM32

6

• Start Workspace launcher if not done automatically by Eclipse.

File	Edit Source Refac	tor Navigate Search P	roject Run W	indow Help
	New Open File	Alt+Shift+N ►	▼ ☆ ▼ 0 ▼	• 9: • 9. • × @ 0
	Close Close All	Ctrl+W Ctrl+ <mark>Shift+W</mark>	9° · · · ·	
	Save Save As Save All Revert	Ctrl+S Ctrl+Shift+S		
	Move Rename Refresh Convert Line Delimite	F2 F5 rs To		
	Print	Ctrl+P		
	Switch Workspace Restart Import Export		C:\Users' C:\f7hon C:\Data\.sw C:\L4hon	v4stm32
	Properties	Alt+Enter	Other	
	Exit			





Create a new workspace SW4STM32

 Create new workspace in the desired location – but not in the same folder where the project which will be imported is located (it must be one level above the project)

Workspace Launcher	C/C++ - Eclipse
Select a workspace	File Edit Source Refactor Navigate Search Project Run Window Help
Eclipse stores your projects in a folder called a workspace. Choose a workspace folder to use for this session.	
Workspace: C:_Work_Seminar Browse	
Use this as the default and do not ask again	
 An empty workspace will be generated 	Problems ☆ Tasks □ Console Properties 0 items
life.augmented	0 items selected

Open STM32 Tools

Import the project into the workspace 1/3 SW4STM32

Import "L4_Blinky" project into empty workspace following below steps:

C/C++ - Eclipse Edit Source Refactor Navigate Search P File New Alt+Shift+N ▶ Open File... Close Ctrl+W Ctrl+Shift+W Close All Save Ctrl+S Save As... Ctrl+Shift+S Save All Revert Move... Rename. Refresh **IMPORT** project Convert Li into workspace Print... Switch Workspace Restart Import... Export...

Import	
Select Create new projects from an archive file or directory.	Ľ
Select an import source:	
type filter text	
 ▲ Seeneral ▲ Archive File ▲ Existing Projects into Workspace ▲ File System ➡ Preferences ▶ See C/C++ ▶ See CVS ▶ Se Git ▶ See Install 	
? < Back Next > Finish	Cancel

This is possible to import multiple projects into a single workspace

Open STM32 Tools

Import the project into the workspace 2/3 SW4STM32

In this example L4_Blinky project will be processed.

Import		Import
Import Projects		Import Projects Select a directory to search for existing Eclipse projects.
Select a directory to search for existing Eclipse projects.		● Select root directory: C:_Work_Seminar\L4_Blinky ■ Browse ○ Select archive file: ■ Browse
Select root directory:	Browse	Projects:
Select archive file:	B se	3 VL4_Blinky (C:_Work_Seminar\L4_Blinky) Select All Deselect All
Select project location (as configured in STM32CubeMX – Step2)		Options Search for nested projects Copy projects into workspace Hide projects that already exist in the workspace Working sets Add project to working sets Working sets:
Folder: L4_Blinky	▼	<pre></pre>
Iife.augmented	OK Cancel	



Once project is included into the workspace, its folder structure becomes visible in Project Explorer

Places dedicated for user code are marked by /* USER CODE ... BEGIN*/

```
and
```

```
/* USER CODE ... END*/
```

comment lines.

These places are protected from being removed during code re-generation by STM32CubeMX.

This is **possible** to define another user code places in **.c** source files but **not possible** in **.h** header files.









 To make green LED (connected to properly configured PA5 pin) we should continuously invoke GPIO toggle function with the proper delay to make blink visible





Modifying the code blinking green LED (PA5)

Tasks (within while(1) loop in main.c):

- 1. Add GPIO pin toggle function for PA5 pin. Which function we can use here?
- 2. Add 500ms delay between each change of the GPIO pin state. Which function we can use here?

Hints:

- All HAL function begins with **HAL_PPP_** prefix (PPP short name of the peripheral, i.e. GPIO)
- Please try to use Content Assistant (Ctrl+SPACE) in Eclipse





Modifying the code blinking green LED (PA5) - solution ¹³

Solutions (within while(1) loop in main.c):

1. Add GPIO pin toggle function for PA5 pin. Which function we can use here? HAL GPIO TogglePin();

Add 500ms delay between each change of the GPIO pin state. Which function we can use here? 2. HAL Delay();

```
/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
/* USER CODE END WHILE */
/* USER CODE BEGIN 3 */
  HAL GPIO TogglePin(GPIOA, GPIO PIN 5);
  HAL Delay(500);
/* USER CODE END 3 */
```









Building the project in SW4STM32

Project Explorer 🖾

- To build the project press Ctrl+B or click Make All
 icon
- In case of multiple compilation errors, re-run Indexing of the project
- After proper build there are information about code/data space usage in Console window displayed

13:37:56 Build Finished (took 25s.41ms)

🖹 Problems 🧔 Tasks 📮 Console 🛛 🔲 Properties CDT Build Console [L4 Blinkv] 'Generating binary and Printing size information:' arm-none-eabi-objcopy -O binary "L4 Blinky.elf" "L4 Blinky.bin" arm-none-eabi-size "L4 Blinky.elf" hex filename text data bss dec 1568 5848 16d8 L4 Blinky.elf 4256 24 . .

```
L4 Blinkv
             New
   പ്പി
            Go Into
  Þ 🕮
            Open in New Window
  s 🙉 1
            Copy
                                                             Ctrl+C
  a 🙉 🧐
             Paste
                                                             Ctrl+V
    Þ
            Delete
                                                            Delete
            Remove from Context
                                              Ctrl+Alt+Shift+Down
            Source
             Move...
                                                                F2
            Rename...
    1.
             Import...
            Export...
            Build Project
            Clean Project
            Refresh
                                                                F5
            Close Project
            Close Unrelated Projects
            Make Targets
            Index
                                                                          Rebuild
            Build Configurations
                                                                          Freshen All Files
            Show in Remote Systems view
                                                                          Update with Modified Files
            Profiling Tools
                                                                          Re-resolve Unresolved Includes
            Run As
                                                                          Search for Unresolved Includes
            Dobug As
```



Configure the debug session in SW4STM32 for single project in the workspace

Debu

Select

?

- Before running debug session this is • necessary to configure it for current project
- In case there is a single project in the • workspace this is enough to click the "bug" icon and:
 - 1. Select "Ac6 STM32 C/C++ Application" line and click 'OK' In case the project was generated on existing/defined board (like NUCLEO-L476RG in our example) debug will run automatically
 - 2. Otherwise (we will practice it in L4 DAC ADC example later) it is necessary to configure debug device (STLinkV2-1 in our case) and debug interface (SWD in our case) and click 'OK'
- Next step would be to run the debug session • (see the next slide)

Debug As			
Select a way to debug 'L4_BLinky':			
Section Ac6 STM32 C/C++ Application	🕓 No board found : Define new board or select an e 💻 🏹		
Local C/C++ Application	No board has been found for this project. (i) Before launch debug, a board has to be defined. Please create a new board or use an existing one.		
	Oefine new board Output Use existing board		
Description Debug with the Ac6 STM32 Debugging	Enter new board name : L4_DAC_ADC Select board chip		
	Select series : STM32L4		
	Select mcu : STM32L476RGTx -		
	Select board :		
OK Cancel	Debug device : 2 ST-LinkV2-1 Debug interface : SWD		
	Ok Cancel		

16





Configure the debug session in SW4STM32 for multiple projects in the workspace 1/2





Run the debug session in SW4STM32 for multiple projects in the workspace 2/2

- Connect Nucleo board with miniUSB cable (ST-Link)
- In case of the projects generated for ST board, there should be selected board configuration script which specifies debug device and its interface (you can check it in Debugger tab)
- Debug perspective will be run (please select Yes in the information window)
- This is enough just to click a "bug" icon to enter debug session next time.



Debug Configurations

	Name: L4_Blinky Debug
type filter text	📄 Mair 🏇 Debugger 🕞 Startup 🔲 Common 🦃 Source
 Ac6 STM32 Debugging L4_Blinky Debug C/C++ Application C/C++ Attach to Application C/C++ Postmortem Debugger C/C++ Remote Application C/C++ Unit GDB Hardware Debugging Launch Group 	GDB Setup GDB Command: \${openstm32_compiler_path}\arm-none-eabi-gdb Browse OpenOCD Setup OpenOCD Command: "\${openstm32_openocd_path}\openocd.exe" Browse OpenOCD Options : Port number: 3333 Script: Ouse default script Use local script Manual spec Configuration Script: nucleo_1476rg.cfg
Filter matched 9 of 16 items	Revert
(?)	
\odot	



Run the debug session in SW4STM32

for multiple projects in the workspace, but no board specification ¹⁹

- Connect Nucleo board with miniUSB cable (ST-Link)
- Under **Debugger** tab select debug device (ST LinkV2-1 for Nucleo ones) and debug interface (SWD)
- Click Apply and then Debug

- Debug perspective will be run (please select Yes in the information window)
- This is enough just to click a "bug" icon to enter debug session next time.

IV L4_Blinky Debug	
🗈 main 🕸 Debugger 🔪 🕨 Startup 🔲 Common 🦻 Source	
GDB Setup	
GDB Command:	
\${openstm32_compiler_path}\arm-none-eabi-gdb Browse Variables	
OpenOCD Setup	
OpenOCD Command:	
"\${openstm32_openocd_path}\openocd.exe" Browse Variables	Y
	Confirm Perspective Switch
OpenOCD Options :	This kind of launch is configured to open the Debug perspective when it suspends.
Port number: 3333	This Debug perspective is designed to support application debugging. It incorporates
Script:	views for displaying the debug stack, variables and breakpoint management.
Use default script Use local script Manual spec	Do you want to open this perspective now?
Configuratic 3 pt:	
Debug device: ST-LinkV2-1	Remember my decision
Debug interface: SWD	Yes No
500	



Debug session perspective

watching the variables 20

- This is possible to monitor CPU registers, peripherals registers and variables during debug session, but we need to pause the code execution (no live view is possible for the time being).
- To add variable to be monitored highlight it, press right mouse button and select "Add Watch Expression". It will appear in Expressions tab then.
- Values which has changed from previous project pause will be presented on yellow background

🗢 Debug - L4	4_Blinky/Drivers/STM32L4xx_HAL_Drive	er/Src/stm32l Watch	ned variables			
File Edit So	ource Refactor Navigate Search	Project Run				
Quick Access 🖻 🗟 C/C++ 🕸 Debug						
🎋 Debug 🛛	🧏 🕅 🖬 🕆 🗖 🗖	(x)= Variables 💁 Breakpo	oi 🙀 Expressio 🛛 🖁	🕮 Registers 🗰 I/O Regi	. 🛋 Modules 🗖 🗖	
🔺 🗳 L4_Blin	ky.elf [Ac6 STM32 Debugging]			1 🗤 🕸 🕞	🕂 🗶 💥 📑 🖻 🔻 👘	
⊿ 🔐 L4_B	Blinky.elf	Expression	Туре	Value		
4 🧬 T	hread #1 (Suspended : Signal : SIGIN	⇔ uwTick	volatile uint32_t	3234 📃	Values ch	hanged
	HAL_Delay() at stm32l4xx_hal.c:334	🐈 Add new expression			from pro	
=	main() at main.c:93 0x8000f8c					evious
📕 oper	nocd				application	n pause
📕 C:/A	.c6/SystemWorkbench/plugins/fr.ac6.					





Debug session perspective

watching the registers content

21

- This is possible to monitor CPU registers, peripherals registers and variables during debug session, but we need to pause the code execution (no live view is possible for the time being).
- To add peripheral register to watch click right mouse button and select "Activate". Peripheral icon and its registers names will be highlighted in green and will contain "caught" values on next debug pause.
- Values which has changed from previous project pause will be highlighted in red.

Debug - L	4_Blinky/Drivers/STM32L4xx_HAL_Driv Source Refactor Navigate Search	ver/Src/stm32l4xx_hal.c - E	ore regist	ers		Peripherals registers
	: ♥ ■ .N 33. 03 1→			Quick Access	💀 Debug	
🌾 Debug 🛙	3 🧏 🕷 it 🖓 🗖 🗖	(x)= Variables 💁 Breakpoi 🖧 Ex	pressio 🕮 Reg	isters 🗰 I/O Regi 😣 🛋 Module	es 🗖 🗖	
🔺 🗳 L4_Bli	nky.elf [Ac6 STM32 Debugging]					
A 🕃 14	Rlinky elf	 Double-click on register to feto 	ch value		\$	
0	Non-watched	Register	Hex value	Binary value	*	
	nerinheral	▷ ## ADC2			C	Value changed
		▷ 🛗 ADC3				value changed
J or		ADC123_Common				from previous
C:	Watched	GPIO				application pause
	nerinheral			10 10 10 11 11 11 11 11 11 11	01	application paudo
	peripricial		0x00000000	0000000000000000 0 0 0 0 0 0	0.0.0	
		▷ IIII OSPEEDR	0x0C0000F0	00 00 11 00 00 00 00 00 00 00 00	00 00	
		▷ 1919 PUPDR	0x64000000	01_10_01_00_00_00_00_00_00_00_	00_00	
		⊳ 1819 IDR	0x0000C028	000000000000001_1_0_0_0_0	_0_0_0 🔻	
 ■ III 	4	▲ III			•	





Handling the debug session SW4STM32



- 1. Skip all breakpoints
- 2. Run/resume
- 3. Suspend
- 4. Terminate debug session
- 5. Disconnect from the target
- 6. Step into
- 7. Step Over

life.augmented

8. Step Return



22

windows configuration in debug perspective



What have we learnt?

✓ Handling the projects generated by STM32CubeMX in SW4STM32

- ✓ Import project generated by STM32CubeMX
- ✓ Tune sources to run selected peripherals in desired algorithm
- ✓ Build project
- ✓ Configure debug session
- ✓ Run debug session
- ✓ Debug perspective
- ✓ Watching the variables and registers content
- ✓ Handling errors







www.st.com/mcu

