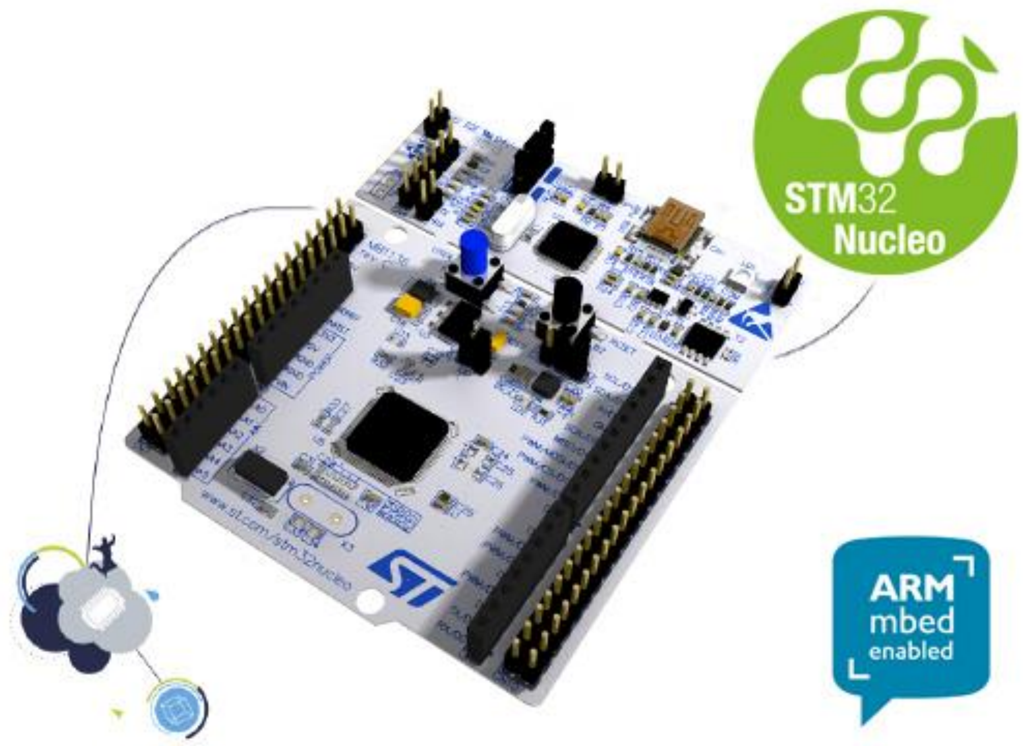
MCSL2017

Lab1

TA:柴俊瑜

STM32 Nucleo Board

- An ARM Cortex-M4 development board
- Build in a ST-LINK as debugger
- Arduino pin compatible
- One user button
- One LED



Hardware Block

Figure 3. Top layout

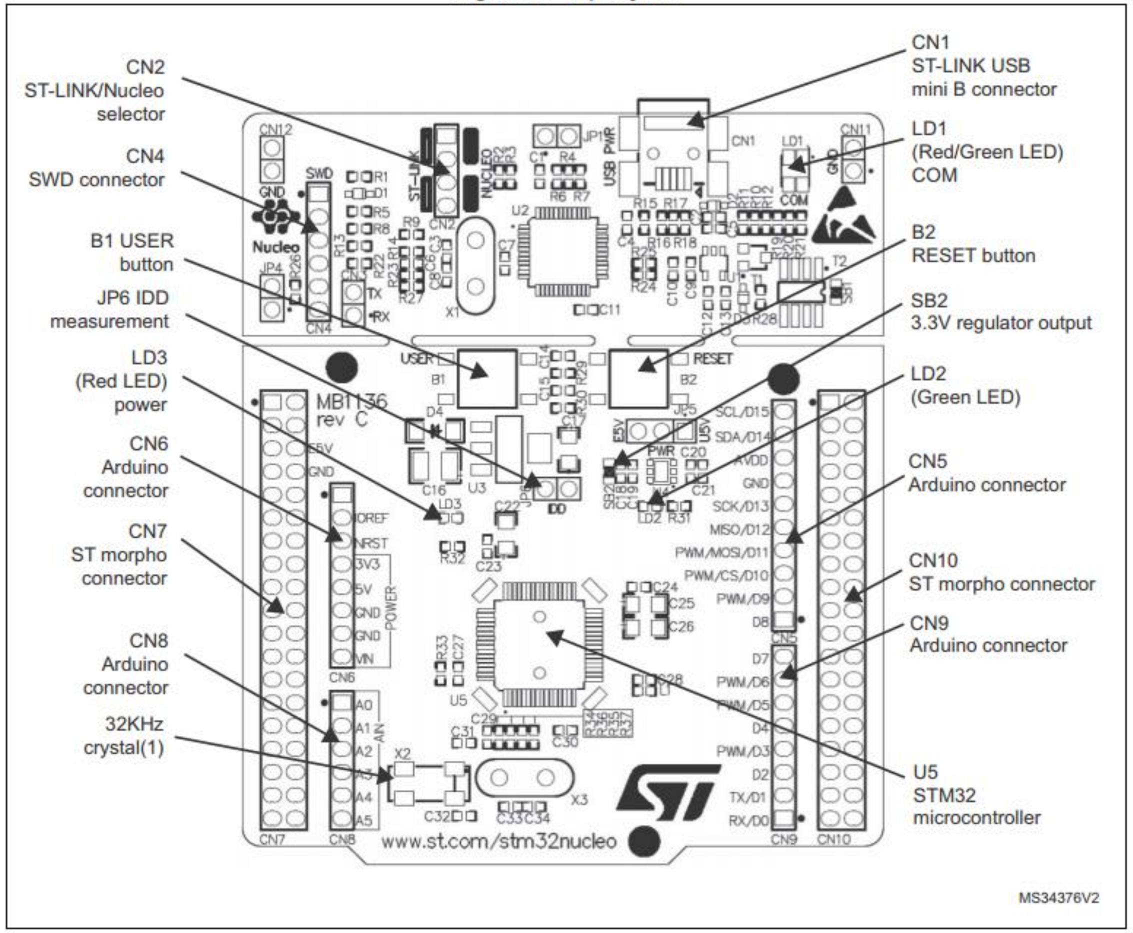
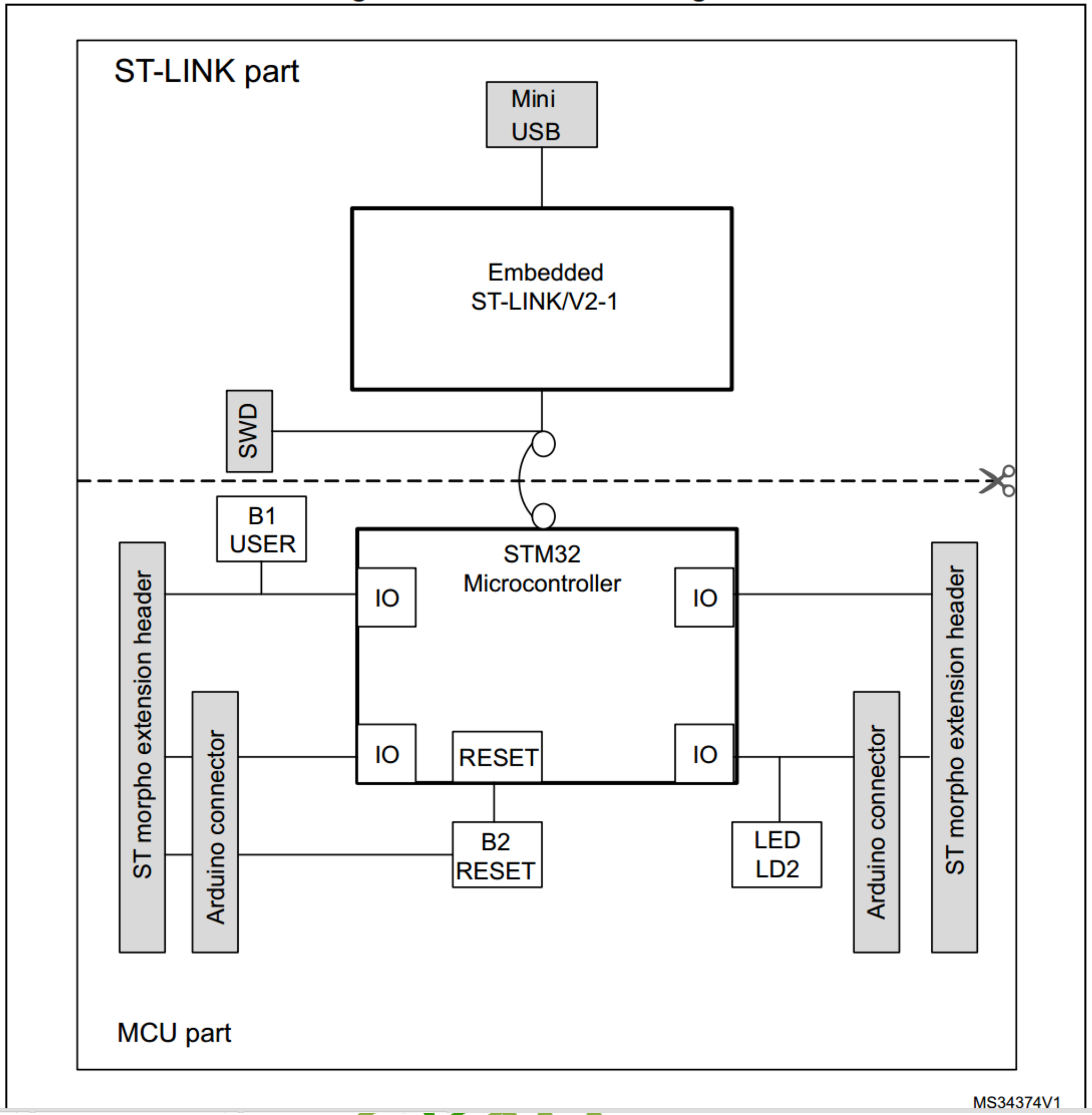
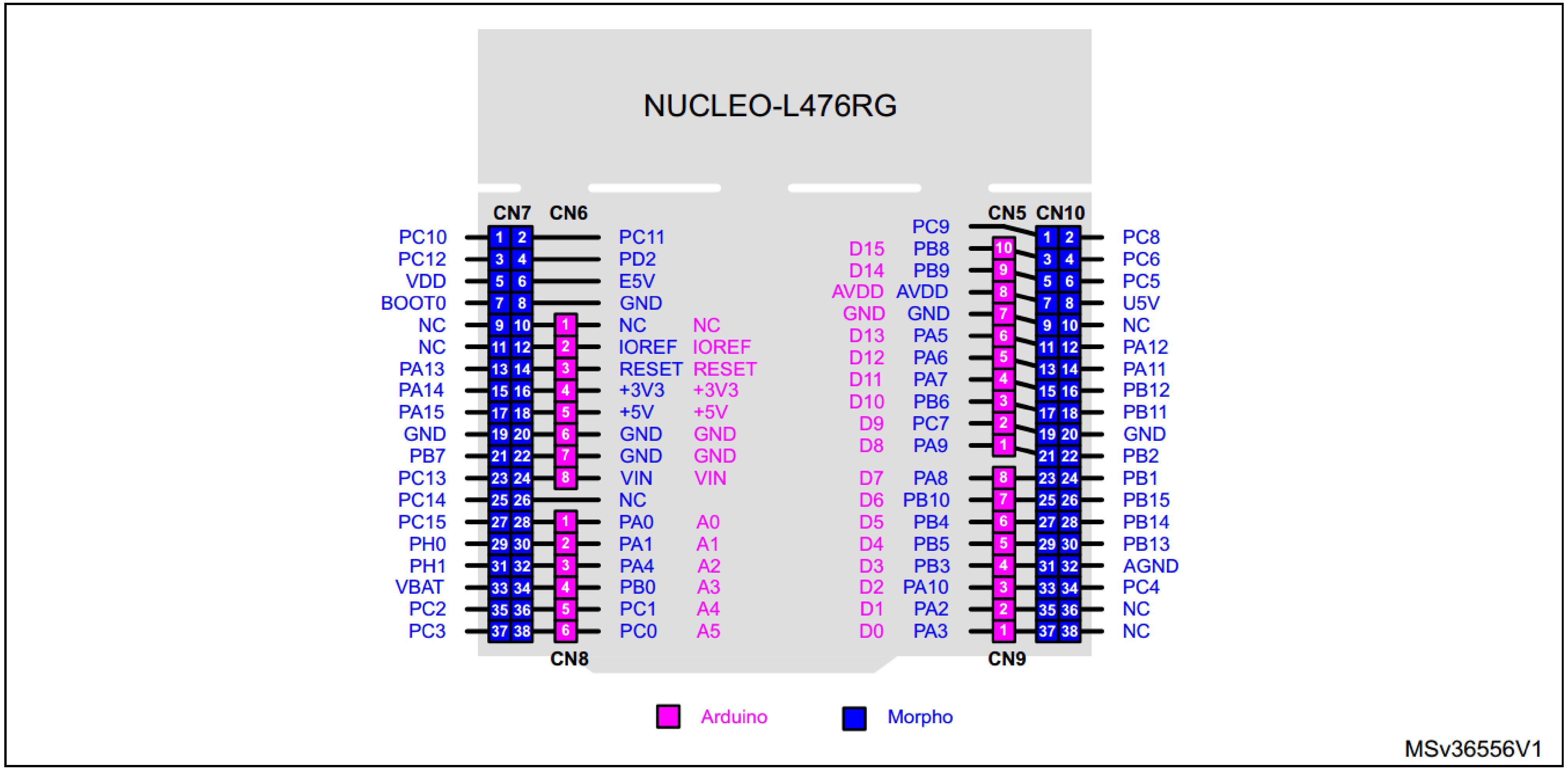


Figure 2. Hardware block diagram



Pin Map

Figure 22. NUCLEO-L476RG



Development Environment

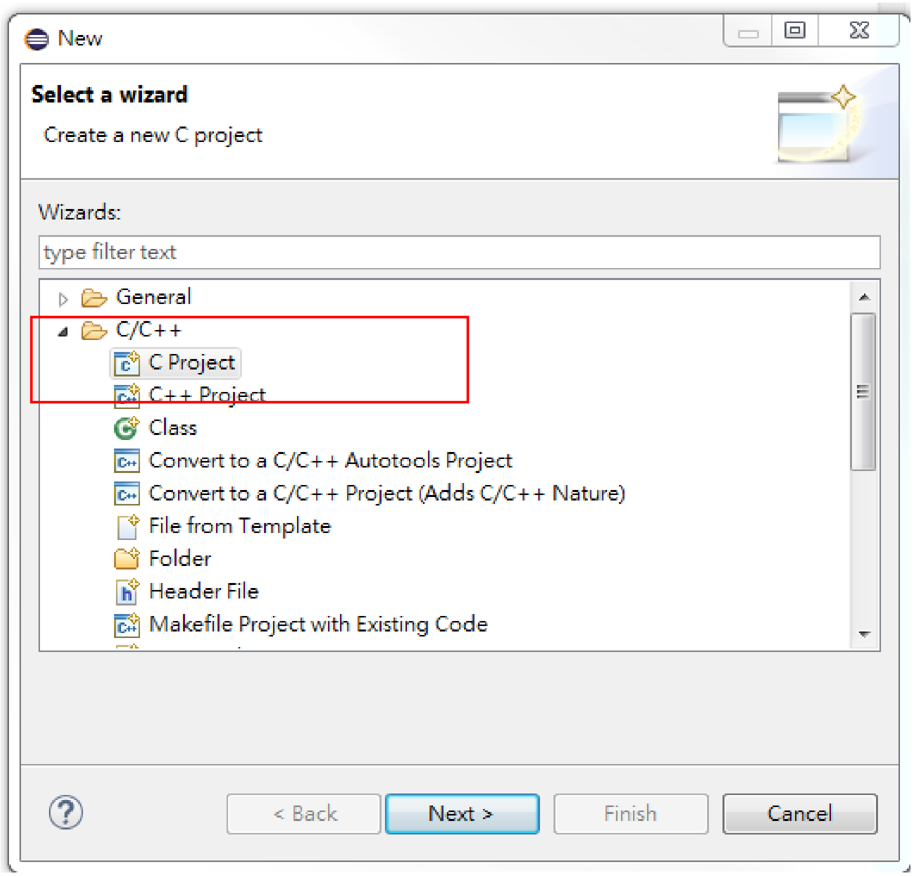
- We use SW4STM32 which is a eclipse based STM32 IDE tool
  - STM32 Devices database and libraries
  - Source code editor
  - Linker script generator
  - Building tools (GCC-based cross compiler, assembler, linker)
  - Debugging tools (OpenOCD, GDB)
  - Flash programing tools
  - http://www.openstm32.org/HomePage

SW4STM32

- Check wiki from http://www.openstm32.org/
- Download Page
- Windows 7
  - http://www.ac6-tools.com/downloads/SW4STM32/install_sw4stm32_win_64bits-v2.2.exe
- Linux
  - http://www.ac6-tools.com/downloads/SW4STM32/install_sw4stm32_linux_64bits-v2.2.run
  - Dependence
    - JRE7
    - sudo apt-get install libc6:i386 lib32ncurses5

Create Project

- Create a ‘lab1’ project


New

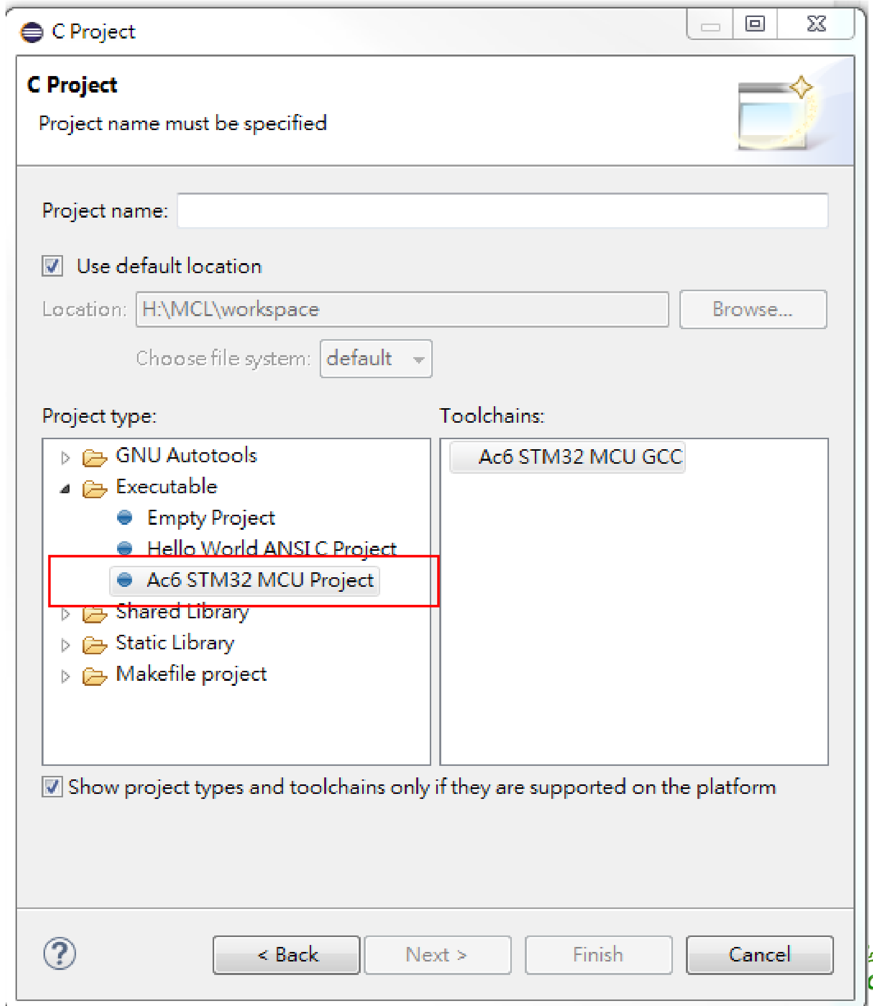
Select a wizard

Create a new C project

Wizards:

type filter text

- General
- C/C++
  - C Project
  - C++ Project
  - Class
  - Convert to a C/C++ Autotools Project
  - Convert to a C/C++ Project (Adds C/C++ Nature)
  - File from Template
  - Folder
  - Header File
  - Makefile Project with Existing Code


C Project

C Project

Project name must be specified

Project name:

☑ Use default location

Location: H:\MCL\workspace

Choose file system: default

Project type:

- GNU Autotools
- Executable
  - Empty Project
  - Hello World ANSI C Project
  - Ac6 STM32 MCU Project
- Shared Library
- Static Library
- Makefile project

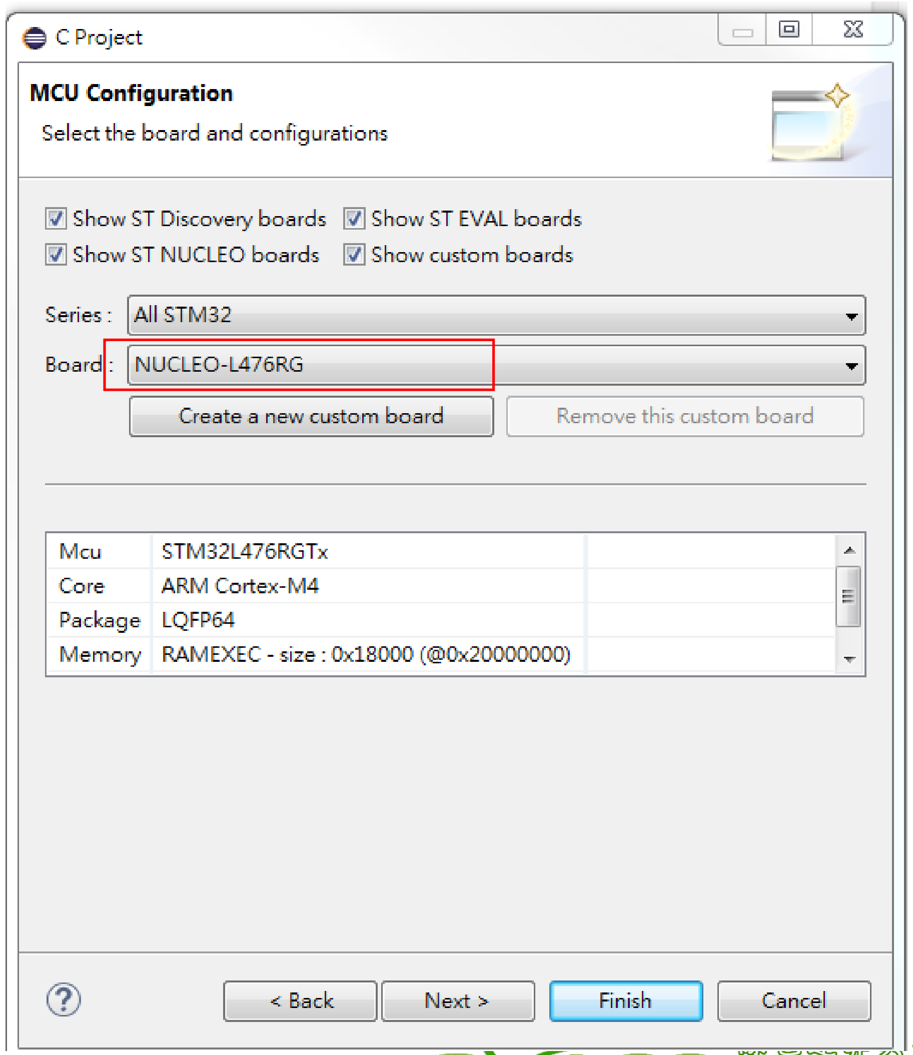
Toolchains:

Ac6 STM32 MCU GCC

☑ Show project types and toolchains only if they are supported on the platform

MCU Configuration

- Select NUCLEO-L476RG board

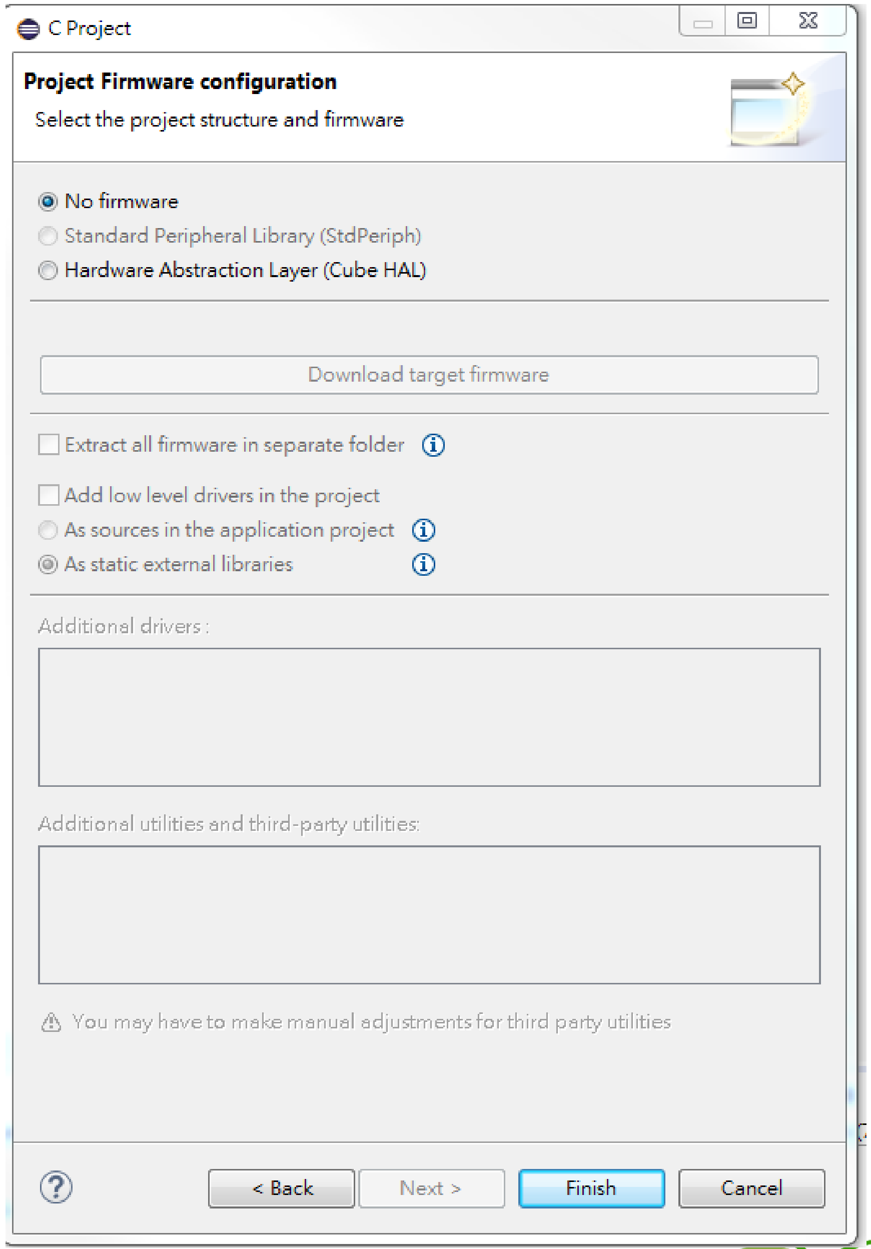


☑ Show ST Discovery boards ☑ Show ST EVAL boards
☑ Show ST NUCLEO boards ☑ Show custom boards

Series : All STM32
Board : NUCLEO-L476RG

Mcu	STM32L476RGTx
Core	ARM Cortex-M4
Package	LQFP64
Memory	RAMEXEC - size : 0x18000 (@0x20000000)

- Choose ‘No firmware’
- Then press ‘Finish’



Project Firmware configuration

Select the project structure and firmware

[✓ No firmware]
Standard Peripheral Library (StdPeriph)
Hardware Abstraction Layer (Cube HAL)

☐ Extract all firmware in separate folder
☐ Add low level drivers in the project
As sources in the application project
[✓ As static external libraries]

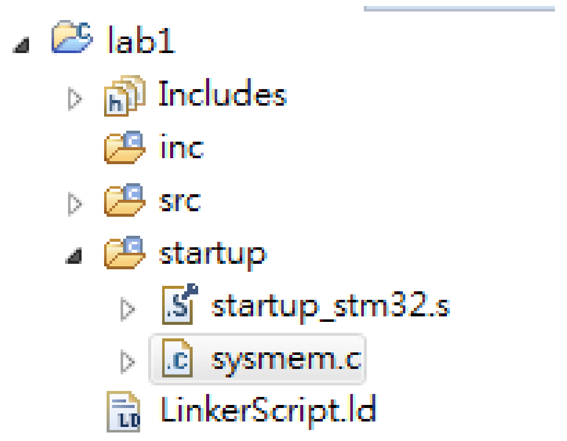
Additional drivers :

Additional utilities and third-party utilities:

You may have to make manual adjustments for third party utilities

Project Files

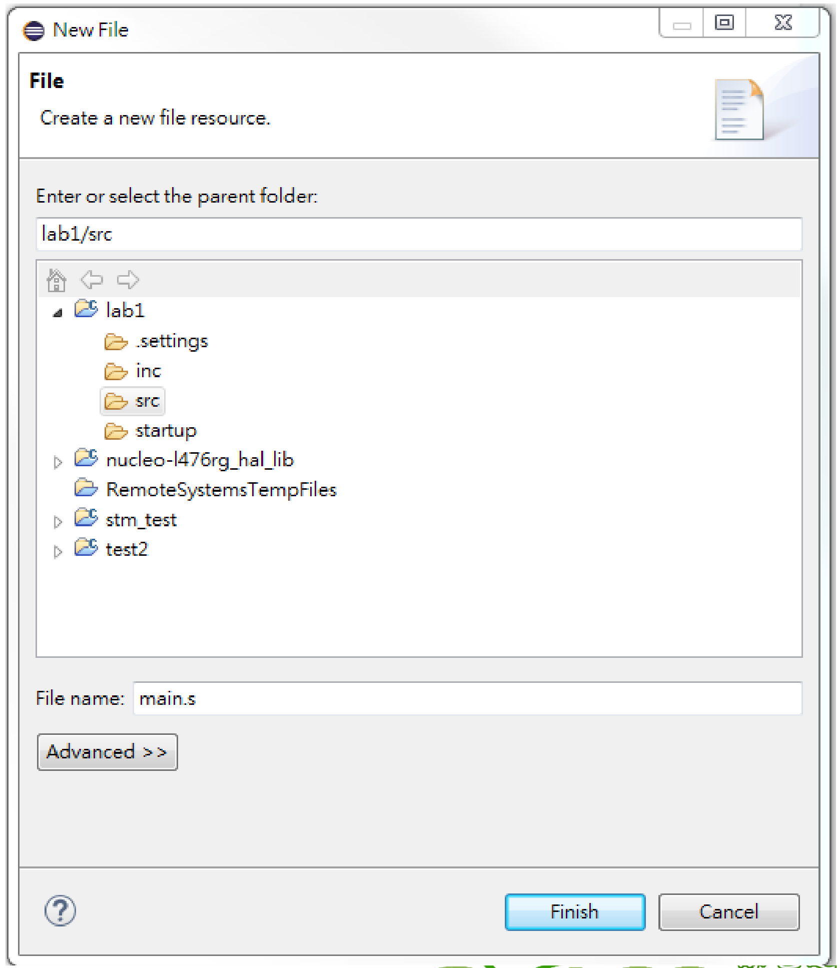
- Then you can see the project files in the 'Project Explorer' list
- It contain the board startup code '**startup_stm32.s**' and linker script '**LinderScript.ld**'



```
lab1
  Includes
  inc
  src
  startup
    startup_stm32.s
    sysmem.c
  LinkerScript.ld
```

Create File

- Right click the lab1/src folder and create a file call **‘main.s’**



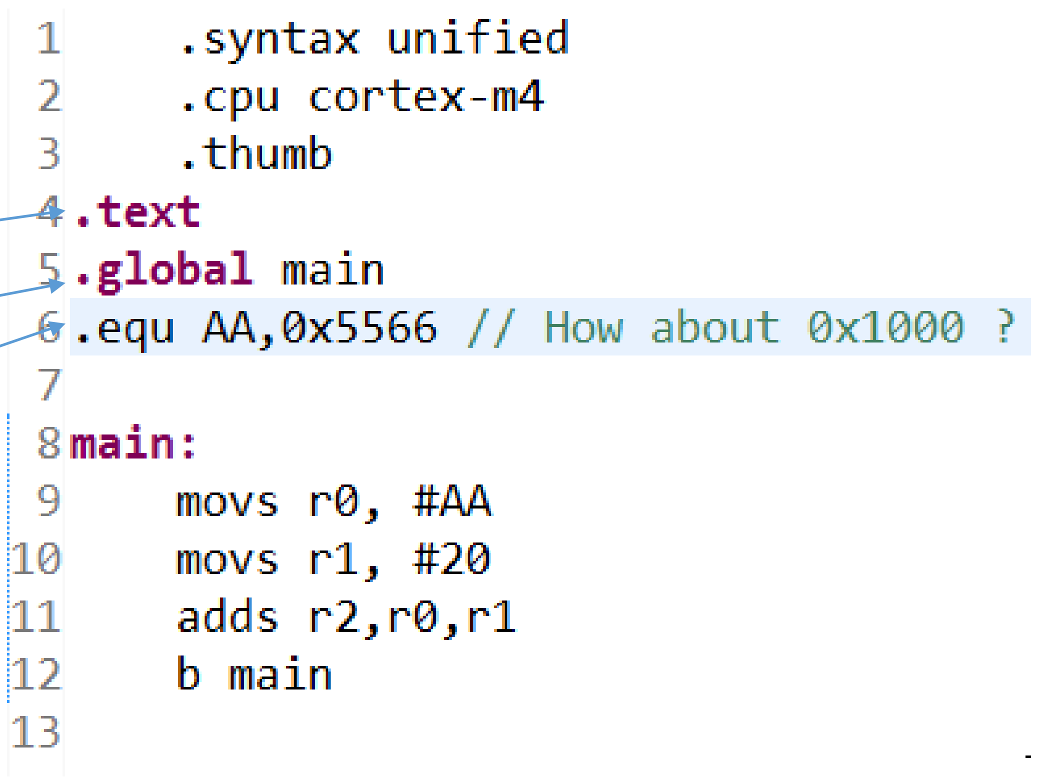
Write Your First Code

Use UAL syntax

Text section start point

Define global symbol

Define a constant symbol ‘AA’

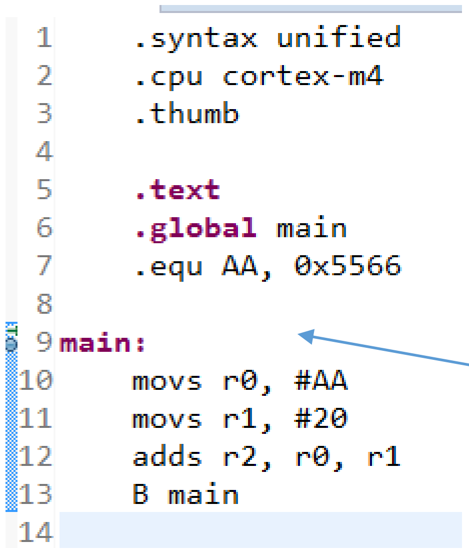


```
     .syntax unified
     .cpu cortex-m4
     .thumb
.text
.global main
.equ AA,0x5566 // How about 0x1000 ?

main:
     movs r0, #AA
     movs r1, #20
     adds r2,r0,r1
     b main
```

main.s

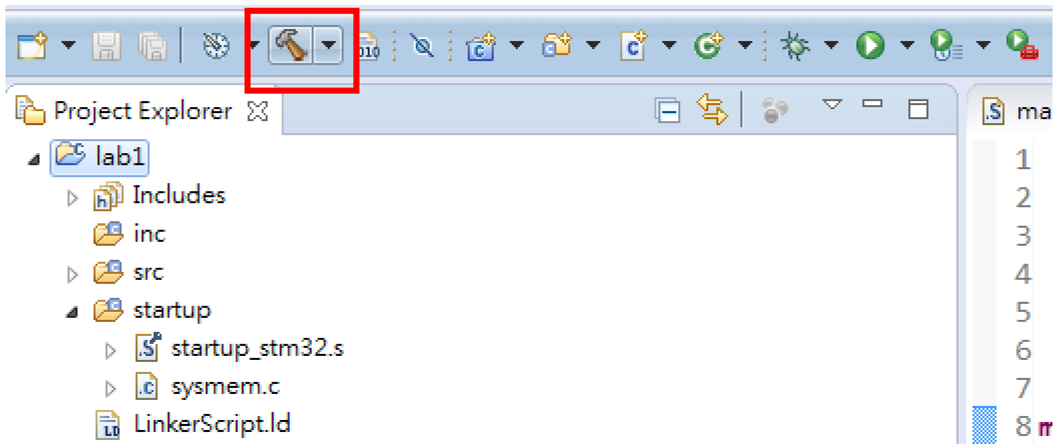
Build Code

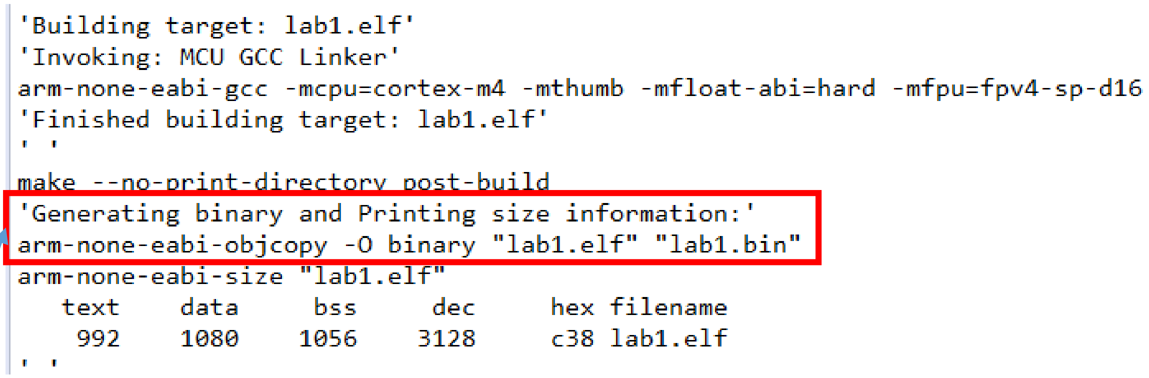
- Write your first code
- Project->Build all



```
    .syntax unified
    .cpu cortex-m4
    .thumb

    .text
    .global main
    .equ AA, 0x5566

main:
    movs r0, #AA
    movs r1, #20
    adds r2, r0, r1
    B main

```

Main entry point.





```
'Building target: lab1.elf'
'Invoking: MCU GCC Linker'
arm-none-eabi-gcc -mcpu=cortex-m4 -mthumb -mfloat-abi=hard -mfpu=fpv4-sp-d16
'Finished building target: lab1.elf'
' '
make --no-print-directory post-build
'Generating binary and Printing size information:'
arm-none-eabi-objcopy -O binary "lab1.elf" "lab1.bin"
arm-none-eabi-size "lab1.elf"
   text    data     bss     dec     hex filename
    992    1080    1056    3128     c38 lab1.elf
' '
```

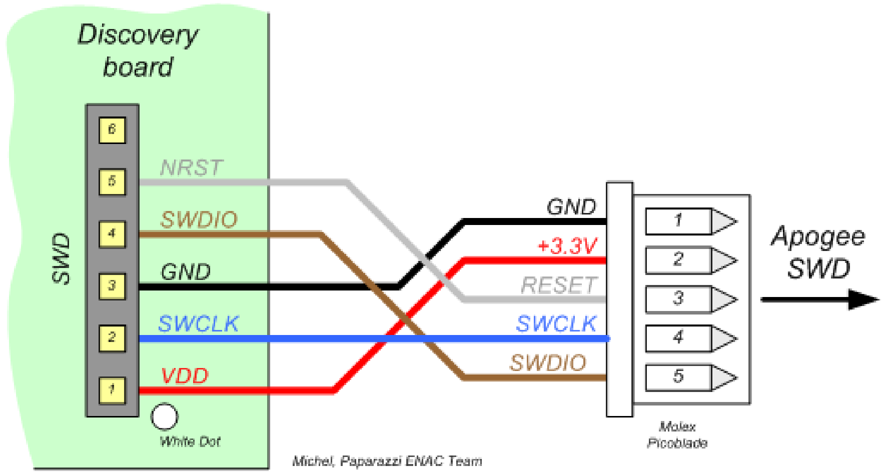
Create the target image file

Build result

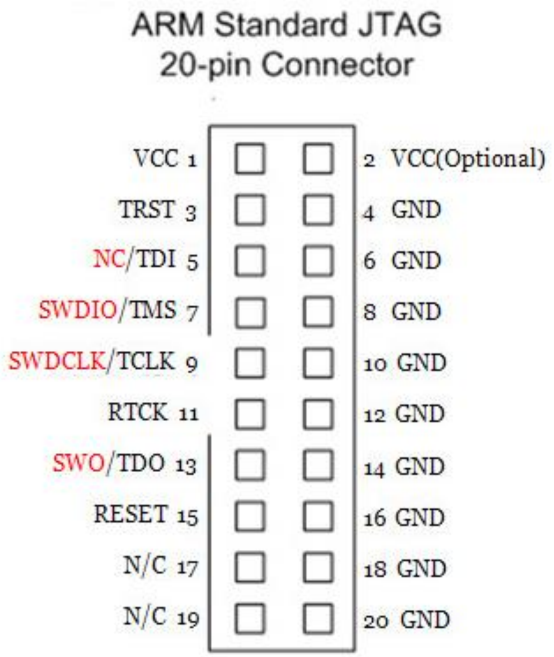
綠色運算與嵌入式系統實驗室
GReen computing And embedded SyStem lab.

Debug Interface

- JTAG(Joint Test Action Group)
  - A standard ASICs hardware debug interface
- SWD(Serial Wire Debug)
  - Only use 5 wires from part of JTAG interface



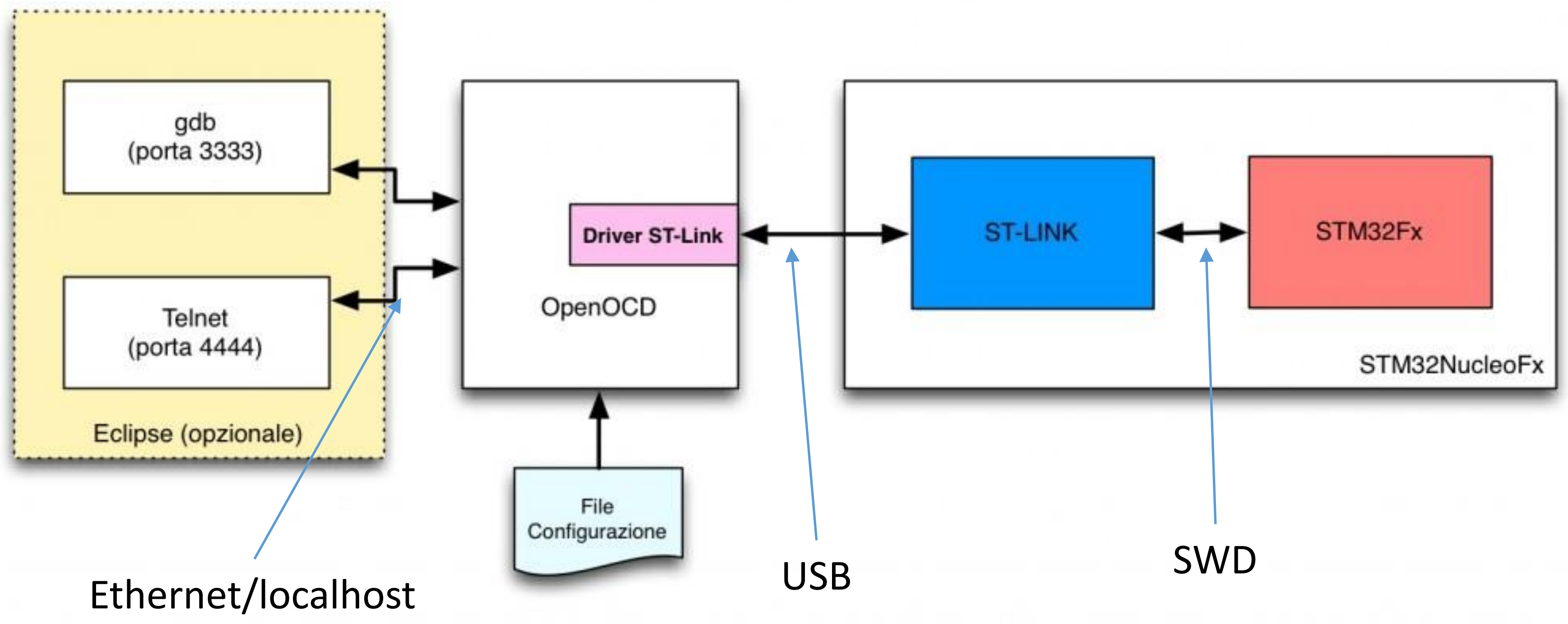
ARM Standard JTAG Connector(20-pins)



VCC	1	2	VCC(Optional)
TRST	3	4	GND
NC/TDI	5	6	GND
SWDIO/TMS	7	8	GND
SWDCLK/TCLK	9	10	GND
RTCK	11	12	GND
SWO/TDO	13	14	GND
RESET	15	16	GND
N/C	17	18	GND
N/C	19	20	GND

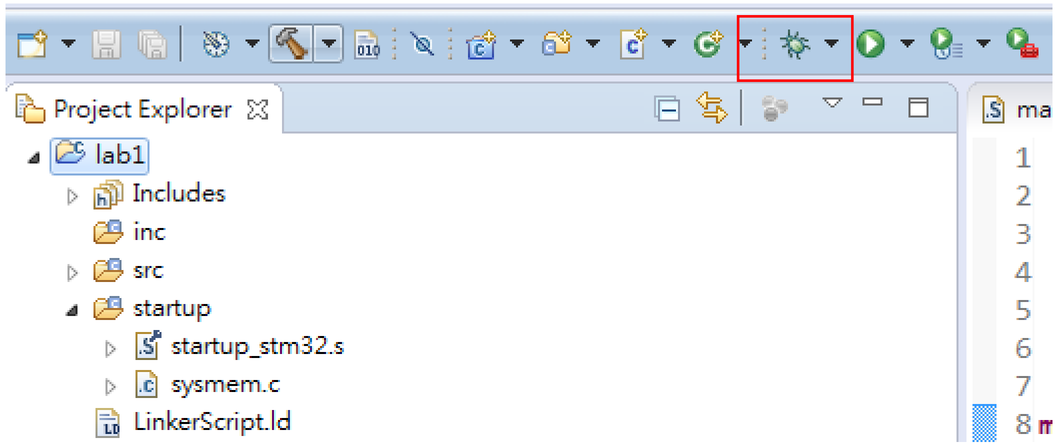
Debug

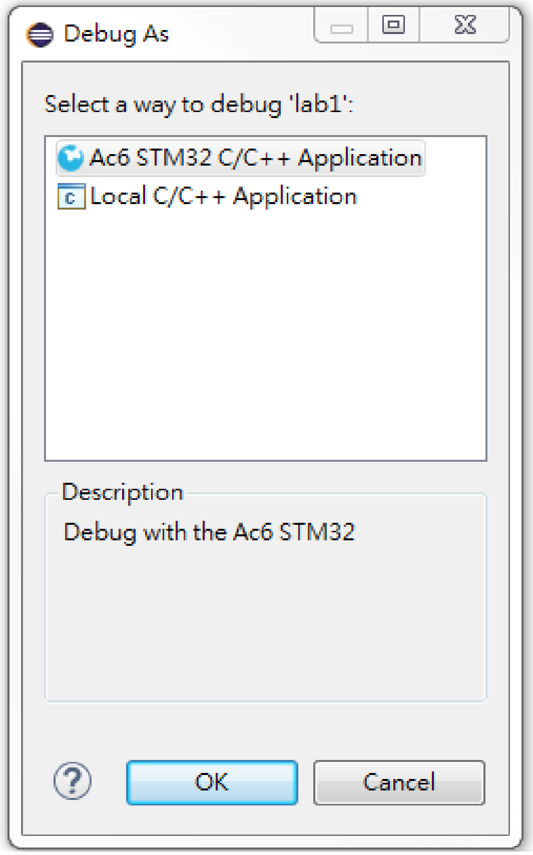
- ST-Link: A STM32 hardware flasher and debugger
- OpenOCD: An open source GDB server



Create a debug configure

- Run->Debug
- Debug as 'AC6 STM32 C/C++ Application'




Debug As

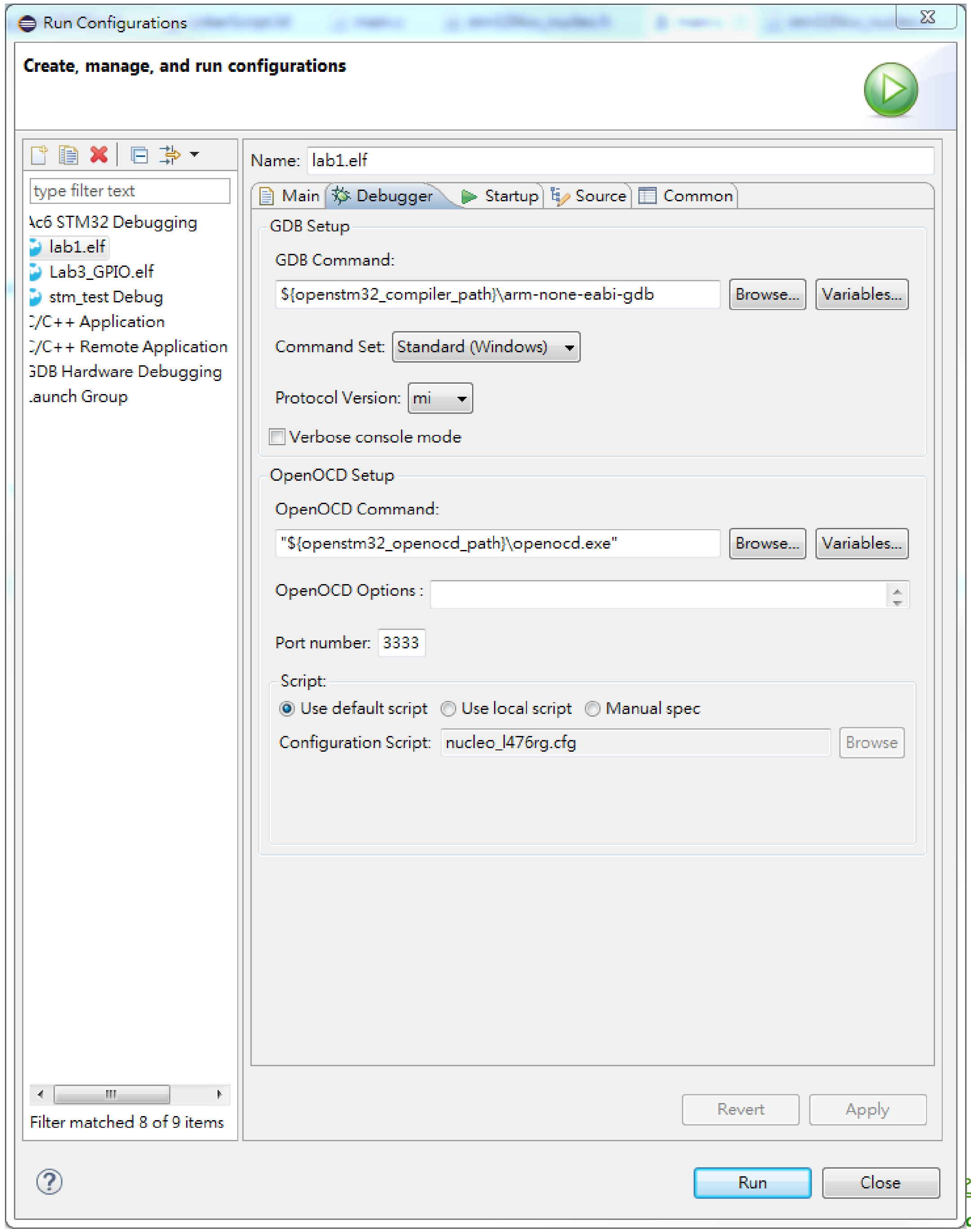
Select a way to debug 'lab1':

Ac6 STM32 C/C++ Application
Local C/C++ Application

Description
Debug with the Ac6 STM32

- Check your debugger configuration
- Run -> Debug Configuration

Note: Make sure your port 3333 no bind any network service!


Run Configurations

Create, manage, and run configurations

type filter text

- Ac6 STM32 Debugging
  - lab1.elf
  - Lab3_GPIO.elf
  - stm_test Debug
- C/C++ Application
- C/C++ Remote Application
- GDB Hardware Debugging
- Launch Group

Filter matched 8 of 9 items

Name: lab1.elf

Main | Debugger | Startup | Source | Common

GDB Setup

GDB Command: ${openstm32_compiler_path}\arm-none-eabi-gdb

Command Set: Standard (Windows)

Protocol Version: mi

Verbose console mode

OpenOCD Setup

OpenOCD Command: "${openstm32_openocd_path}\openocd.exe"

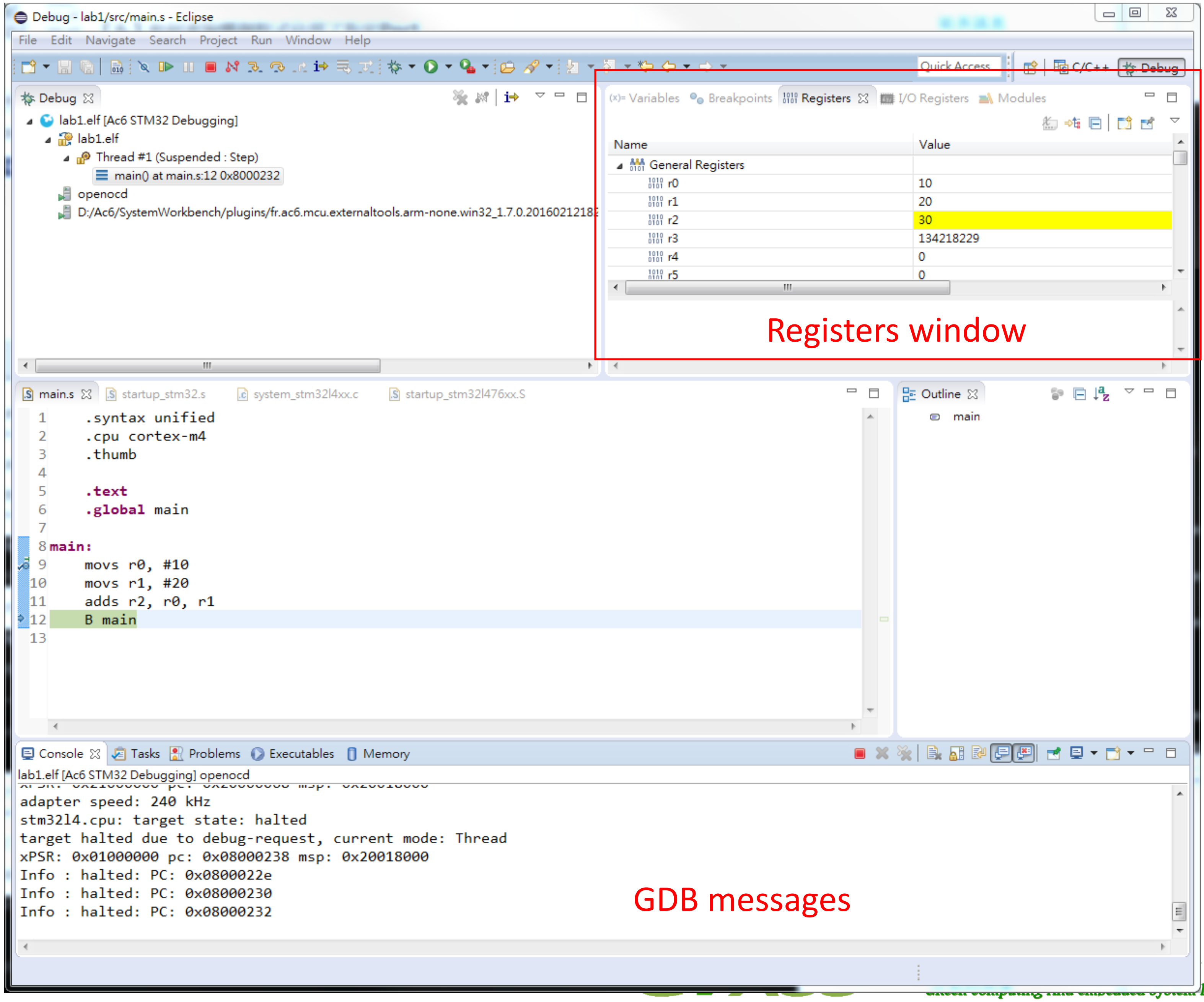
OpenOCD Options :

Port number: 3333

Script: [✓ Use default script] Use local script Manual spec

Configuration Script: nucleo_l476rg.cfg

- By default the GDB will set the first breakpoint at 'main'
- Press 'Step into' button or 'F5' will debug your code step by step.


Debug - lab1/src/main.s - Eclipse
File Edit Navigate Search Project Run Window Help
Debug
lab1.elf [Ac6 STM32 Debugging]
lab1.elf
Thread #1 (Suspended : Step)
main() at main.s:12 0x8000232
openocd
D:/Ac6/SystemWorkbench/plugins/fr.ac6.mcu.externaltools.arm-none.win32_1.7.0.2016021218
Variables Breakpoints Registers I/O Registers Modules
Registers window
main.s startup_stm32.s system_stm32l4xx.c startup_stm32l476xx.S
Outline
main
Console Tasks Problems Executables Memory
lab1.elf [Ac6 STM32 Debugging] openocd
GDB messages

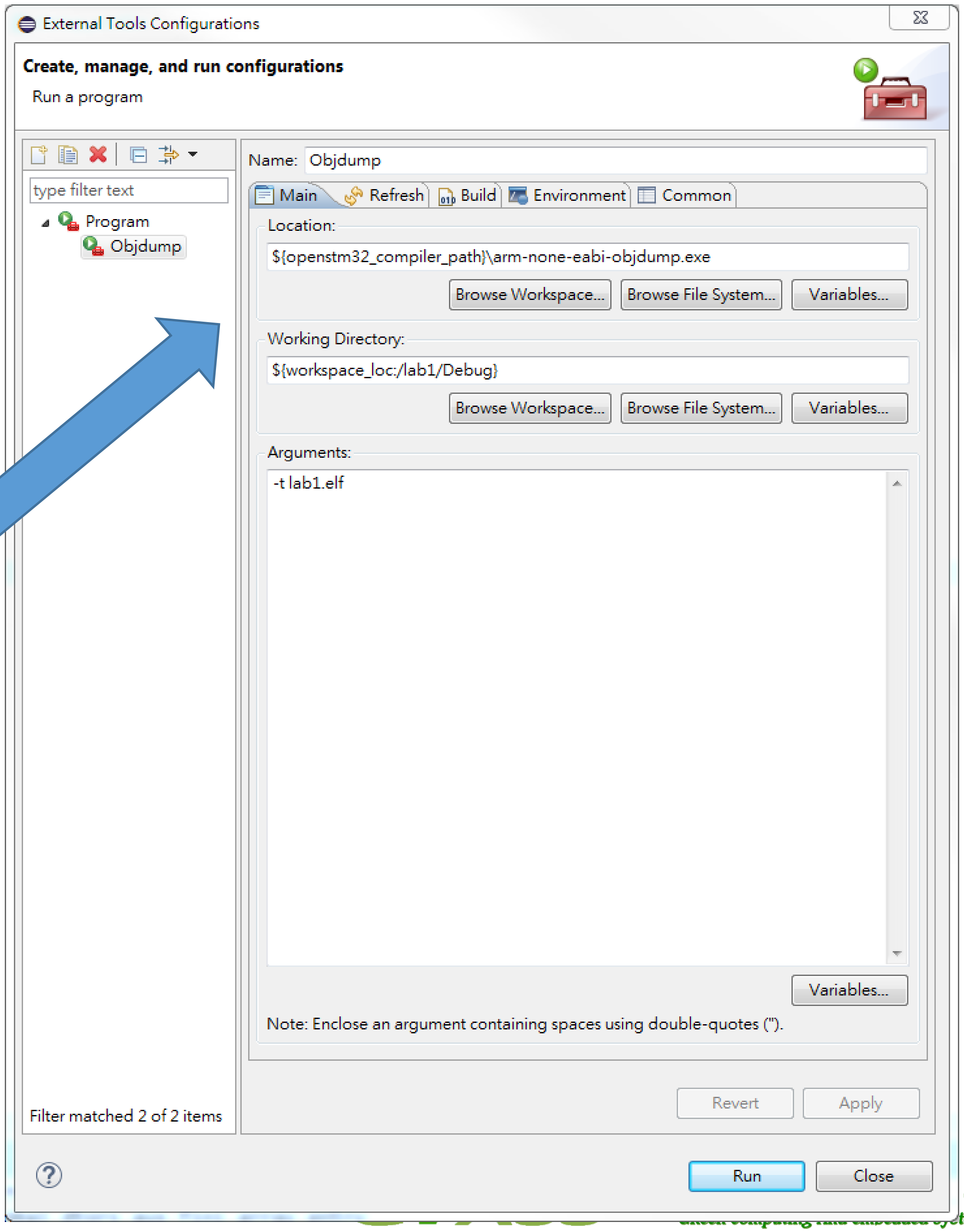
Name	Value
General Registers	
r0	10
r1	20
r2	30
r3	134218229
r4	0
r5	0

```
    .syntax unified
    .cpu cortex-m4
    .thumb

    .text
    .global main

main:
    movs r0, #10
    movs r1, #20
    adds r2, r0, r1
    B main
```

```
adapter speed: 240 kHz
stm32l4.cpu: target state: halted
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x08000238 msp: 0x20018000
Info : halted: PC: 0x0800022e
Info : halted: PC: 0x08000230
Info : halted: PC: 0x08000232
```

Object Dump

- This tool can help you show the program's *symbol table*
- Run->External Tool-> External Tool Configurations
- Set a new program Objdump with the same settings
- Objdump usage guide
  - https://sourceware.org/binutils/docs/binutils/objdump.html



External Tools Configurations

Create, manage, and run configurations

Run a program

Name: Objdump

Location:
${openstm32_compiler_path}\arm-none-eabi-objdump.exe

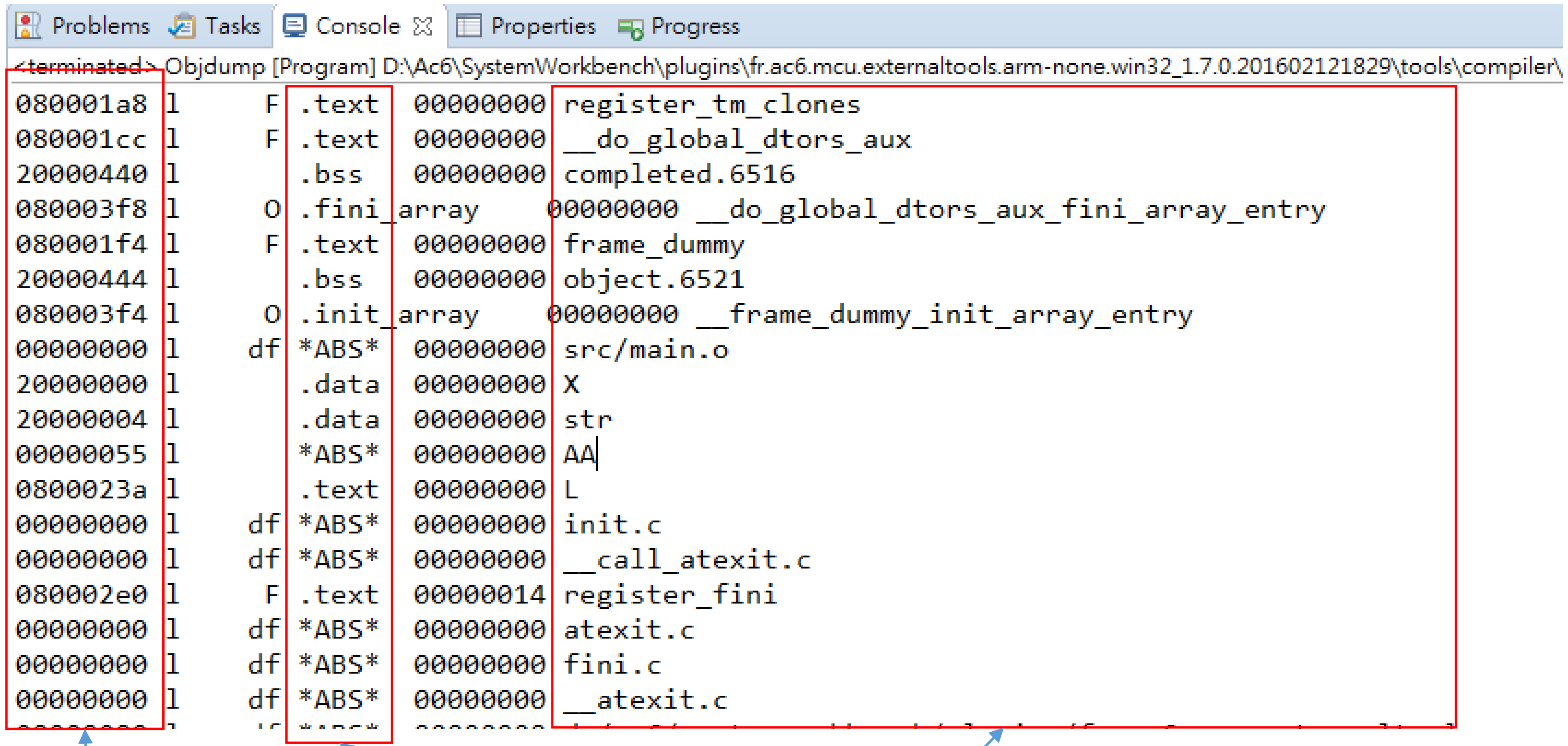
Working Directory:
${workspace_loc:/lab1/Debug}

Arguments:
-t lab1.elf

Note: Enclose an argument containing spaces using double-quotes (").

Filter matched 2 of 2 items

Symbol Table



```
<terminated> Objdump [Program] D:\Ac6\SystemWorkbench\plugins\fr.ac6.mcu.externaltools.arm-none.win32_1.7.0.201602121829\tools\compiler\
080001a8 l     F .text  00000000 register_tm_clones
080001cc l     F .text  00000000 __do_global_dtors_aux
20000440 l       .bss   00000000 completed.6516
080003f8 l     O .fini_array     00000000 __do_global_dtors_aux_fini_array_entry
080001f4 l     F .text  00000000 frame_dummy
20000444 l       .bss   00000000 object.6521
080003f4 l     O .init_array     00000000 __frame_dummy_init_array_entry
00000000 l    df *ABS*  00000000 src/main.o
20000000 l       .data  00000000 X
20000004 l       .data  00000000 str
00000055 l       *ABS*  00000000 AA
0800023a l       .text  00000000 L
00000000 l    df *ABS*  00000000 init.c
00000000 l    df *ABS*  00000000 __call_atexit.c
080002e0 l     F .text  00000014 register_fini
00000000 l    df *ABS*  00000000 atexit.c
00000000 l    df *ABS*  00000000 fini.c
00000000 l    df *ABS*  00000000 __atexit.c
```

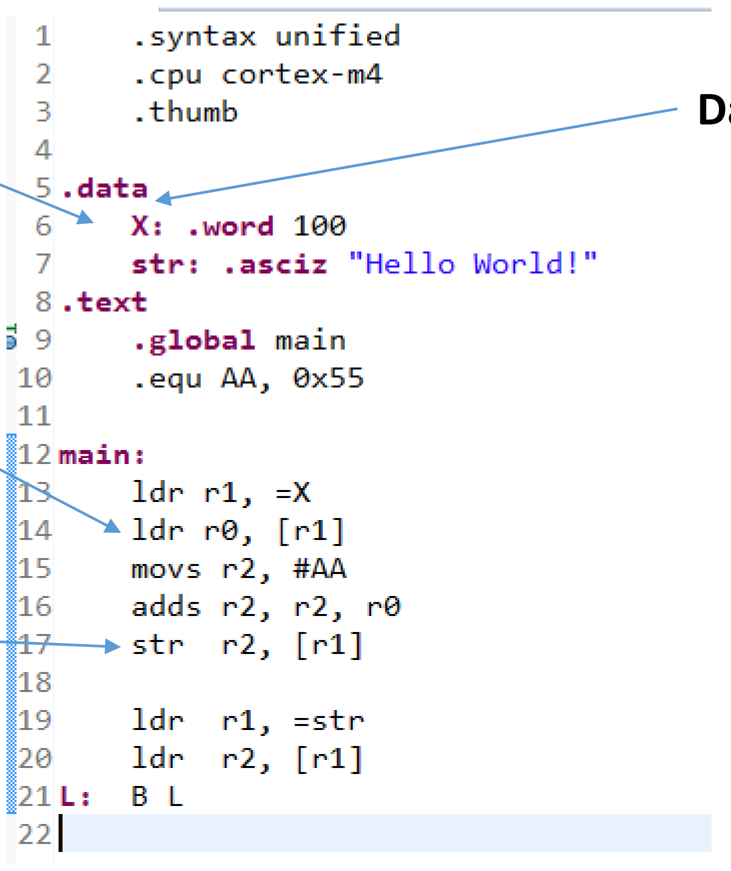
Symbol address

Section locate

Symbol name

Memory Access

- Define data variable
- Direct access
- Indirect read access



```
	.syntax unified
	.cpu cortex-m4
	.thumb

.data
	X: .word 100
	str: .asciz "Hello World!"
.text
	.global main
	.equ AA, 0x55

main:
	ldr r1, =X
	ldr r0, [r1]
	movs r2, #AA
	adds r2, r2, r0
	str  r2, [r1]

	ldr  r1, =str
	ldr  r2, [r1]
L:	B L

```

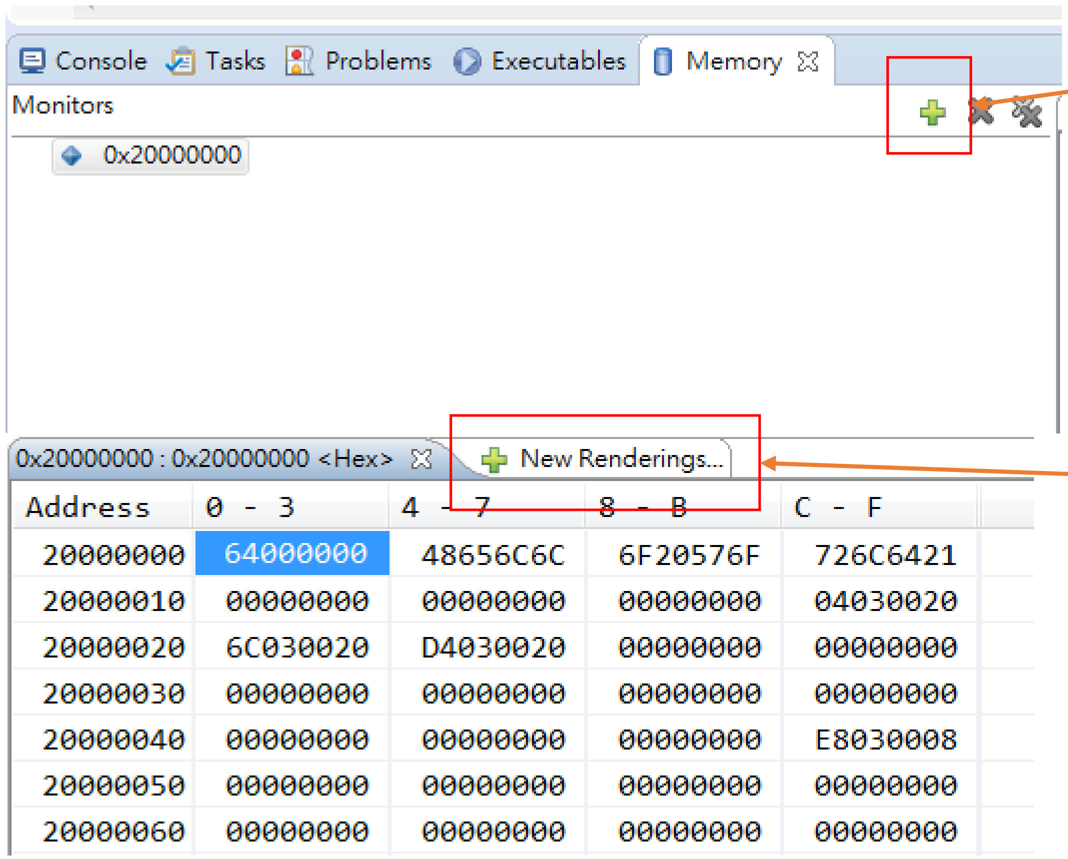
Data section start point

Write the data register into memory

綠色運算與嵌入式系統實驗室
GReen computing And embedded SyStem lab.

Memory Monitors

- That can help you watch the memory content



```
Address    0 - 3     4 - 7     8 - B     C - F
20000000   64000000  48656C6C  6F20576F  726C6421
20000010   00000000  00000000  00000000  04030020
20000020   6C030020  D4030020  00000000  00000000
20000030   00000000  00000000  00000000  00000000
20000040   00000000  00000000  00000000  E8030008
20000050   00000000  00000000  00000000  00000000
20000060   00000000  00000000  00000000  00000000
```

Press it to add a memory monitor

Press “New Renderings” can change the display format

綠色運算與嵌入式系統實驗室 GReen computing And embedded SyStem lab.

Reference

- Getting started with STM32 Nucleo board software development tools
  - http://www.st.com/content/ccc/resource/technical/document/user_manual/1b/03/1b/b4/88/20/4e/cd/DM00105928.pdf/files/DM00105928.pdf/jcr:content/translations/en.DM00105928.pdf
- STM32 Nucleo-64 boards user manual
  - http://www.st.com/content/ccc/resource/technical/document/user_manual/98/2e/fa/4b/e0/82/43/b7/DM00105823.pdf/files/DM00105823.pdf/jcr:content/translations/en.DM00105823.pdf

Linker Script

- https://www.math.utah.edu/docs/info/ld_toc.html#SEC4