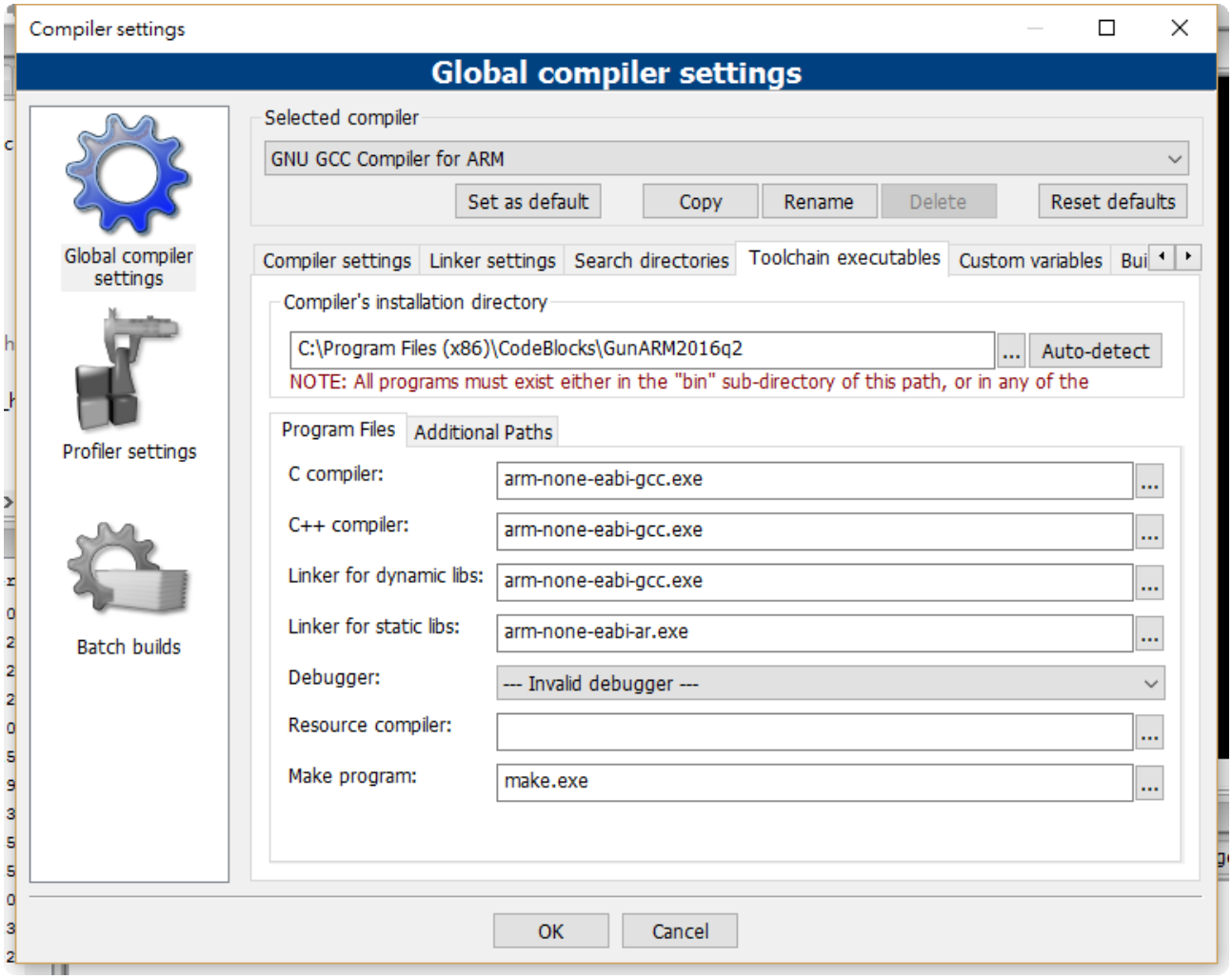
CodeBlcok 除錯 STM32

CodeBlocks 設定 ARM 環境

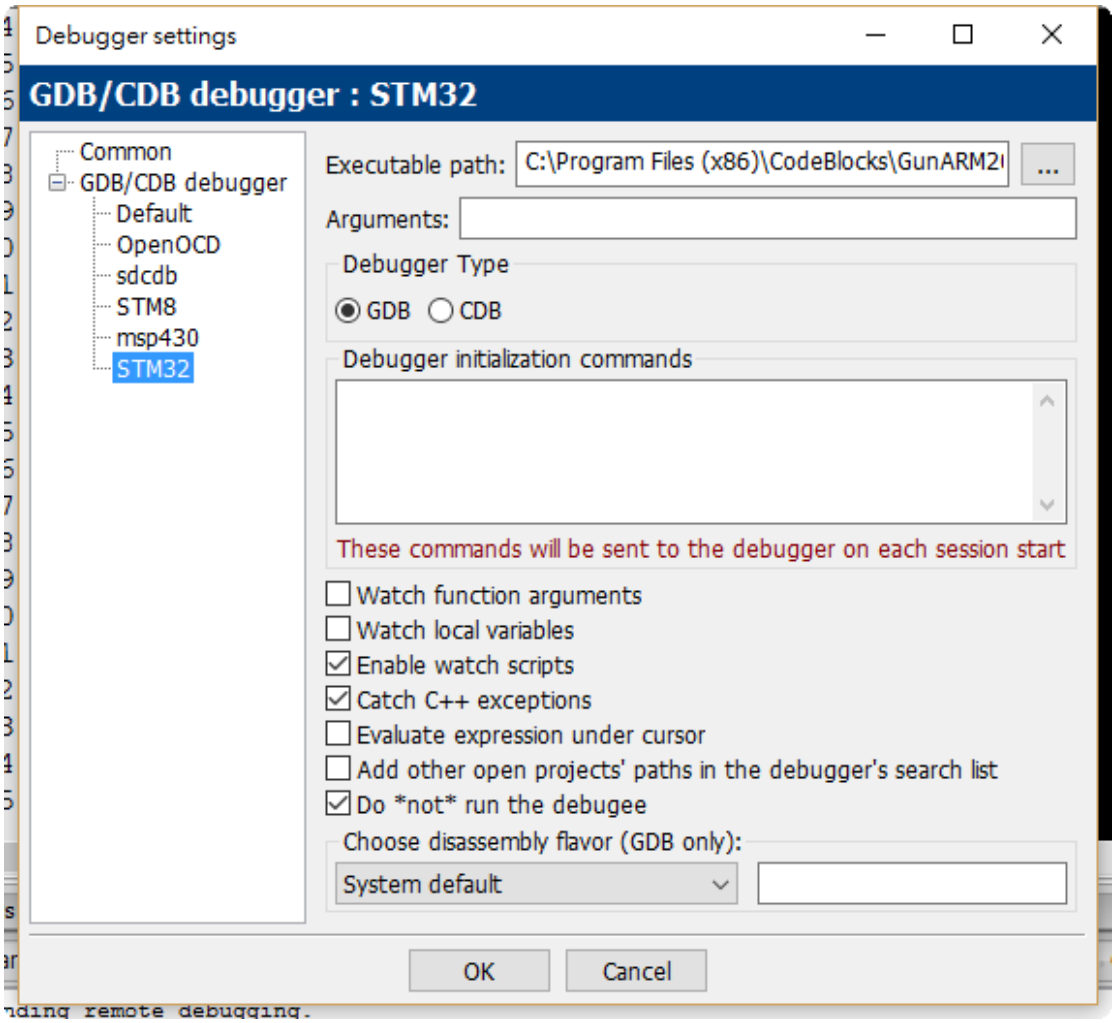
啟動 Code Blocks 做點設定。

➢ 從 Settings -> Compiler... 進入



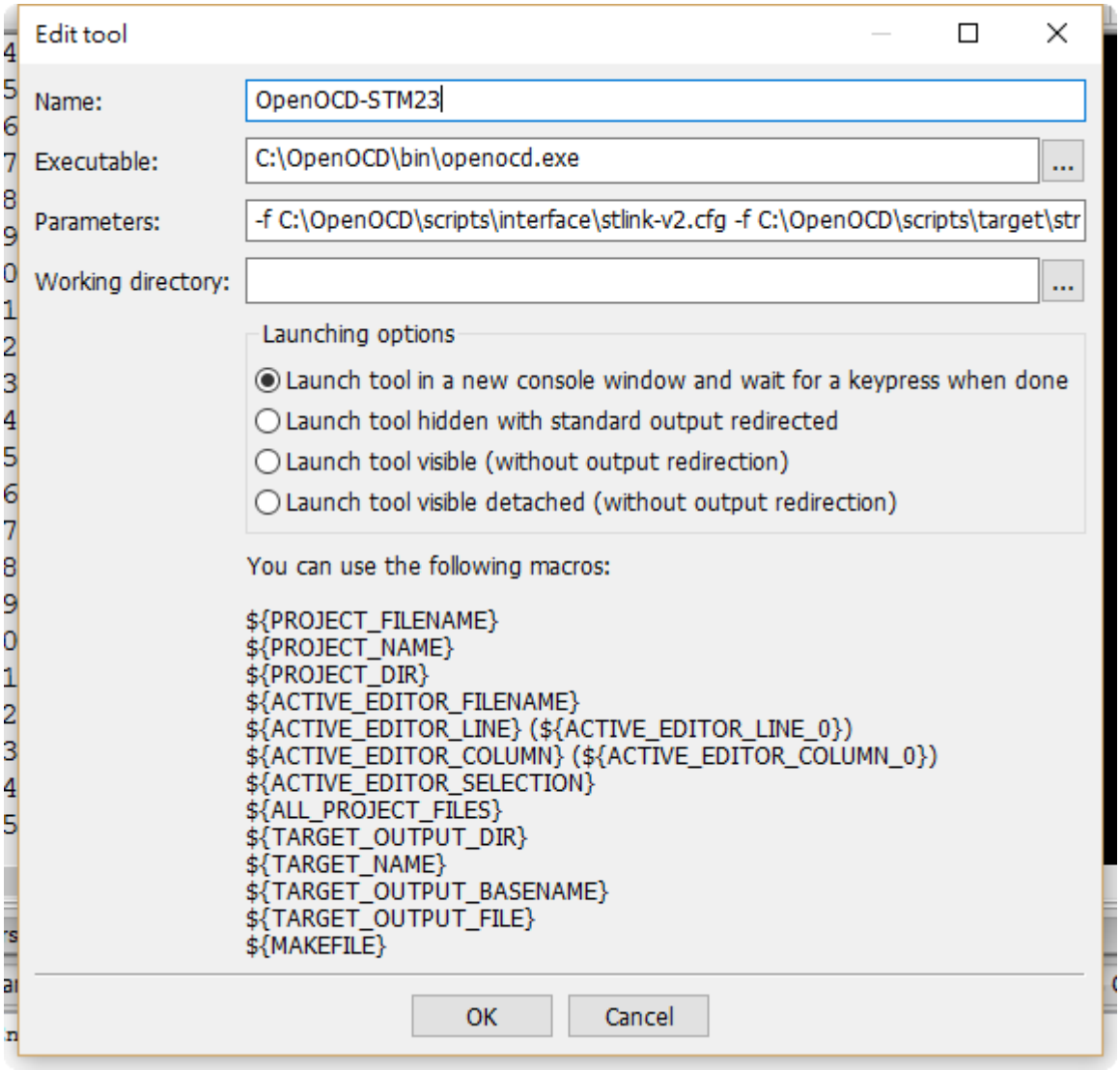
Selected compiler 選擇 GNU GCC compiler for ARM 的項目，根據你自己安裝 compiler 位置設定好。

➢ 然後進入 Settings -> Debugger 設定新的 debugger 專門針對 STM32 的



記得 `Do "not" run the debugee` 這個項目要打勾，避免在非預期狀態下被啟動。

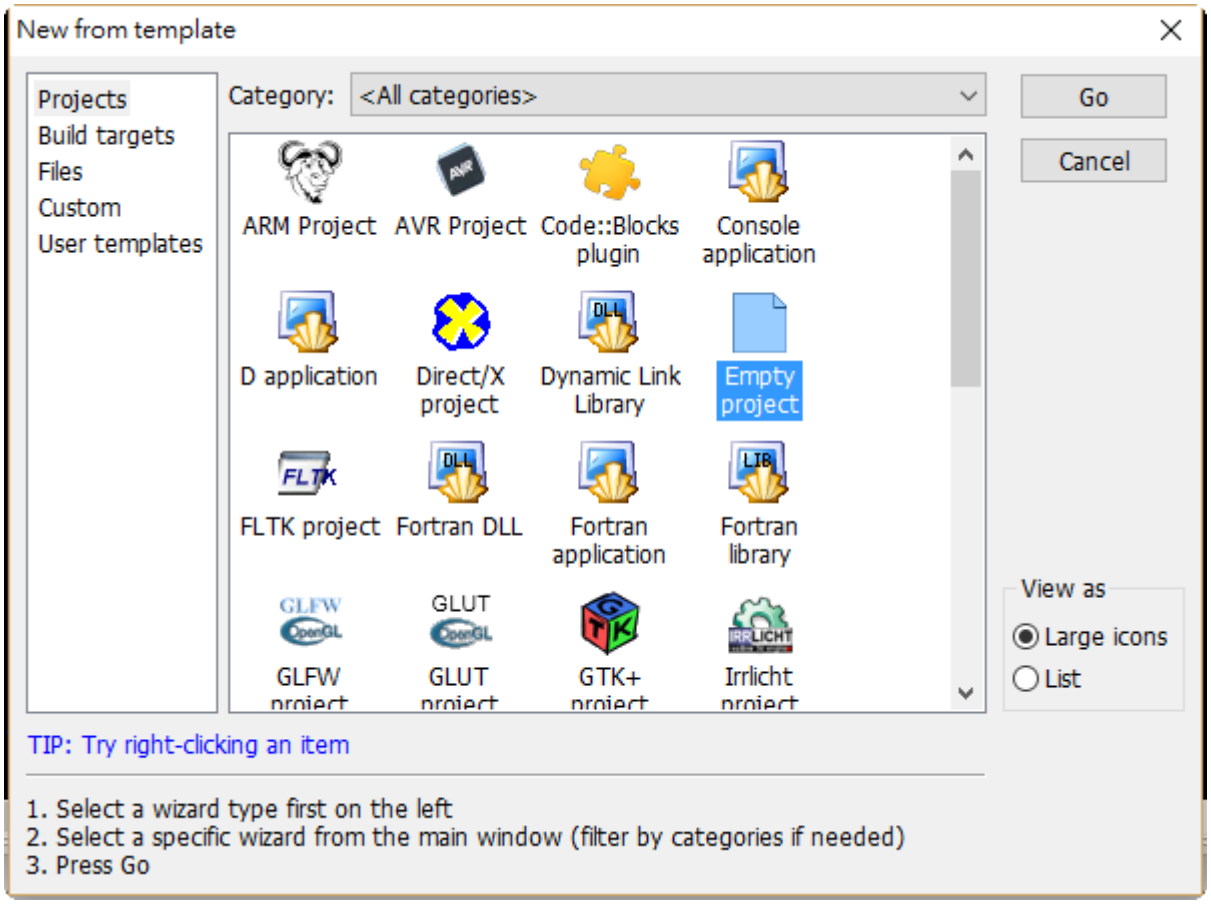
➢ 然後進入 Tools -> Configure tools ，設定一個快速工具好讓 OpenOCD 啟動。



這樣就可以在 GDB debugger 啟動前先快速的啟動 OpenOCD 以利連線。

創建 project

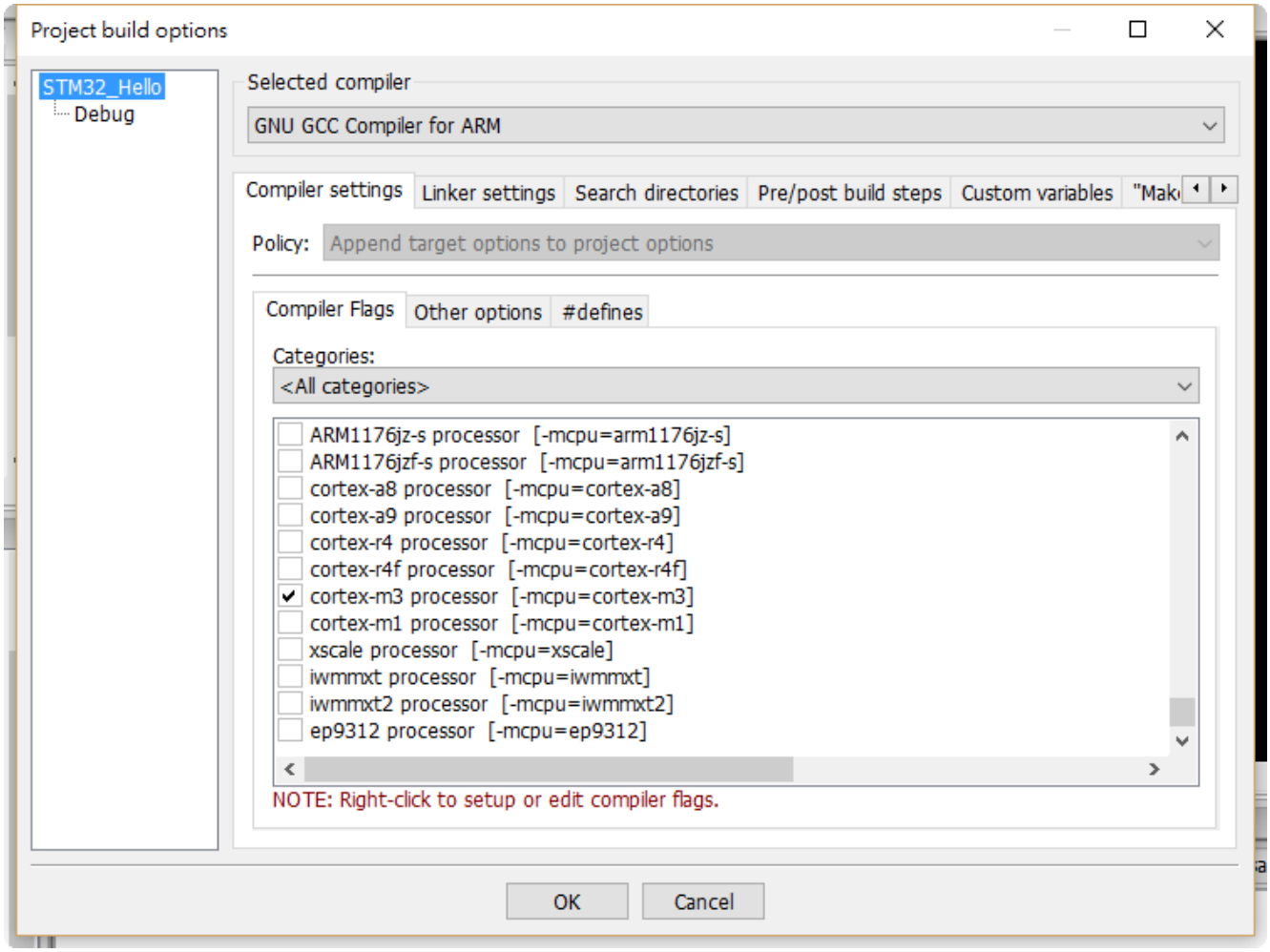
開啟一個新專案來做 STM32 的範例。



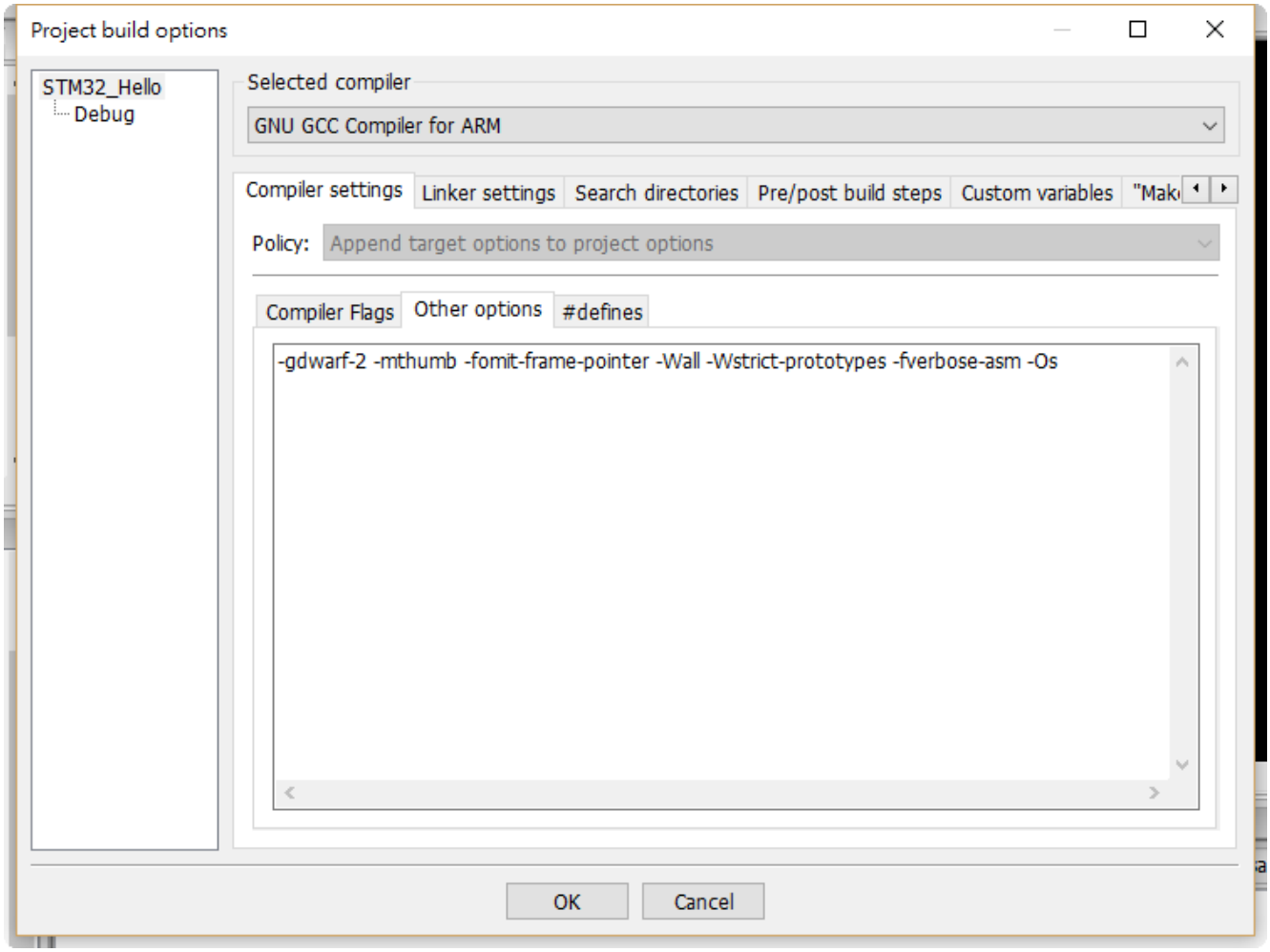
一樣使用空白的專案，但下一步之後的 Compiler 選項記得點選 ARM 專用的。
專案開好後，專案設定的一些東西要變更。

設定 build options

首先是 Projetc -> Build options



由於選定的晶片，STM32 屬於 Cortex-M3 系列的，所以要勾選 M3
然後在 Other options 底下要增加文字
“-gdwarf-2 -mthumb -fomit-frame-pointer -Wall -Wstrict-prototypes -fverbose-asm -Os”

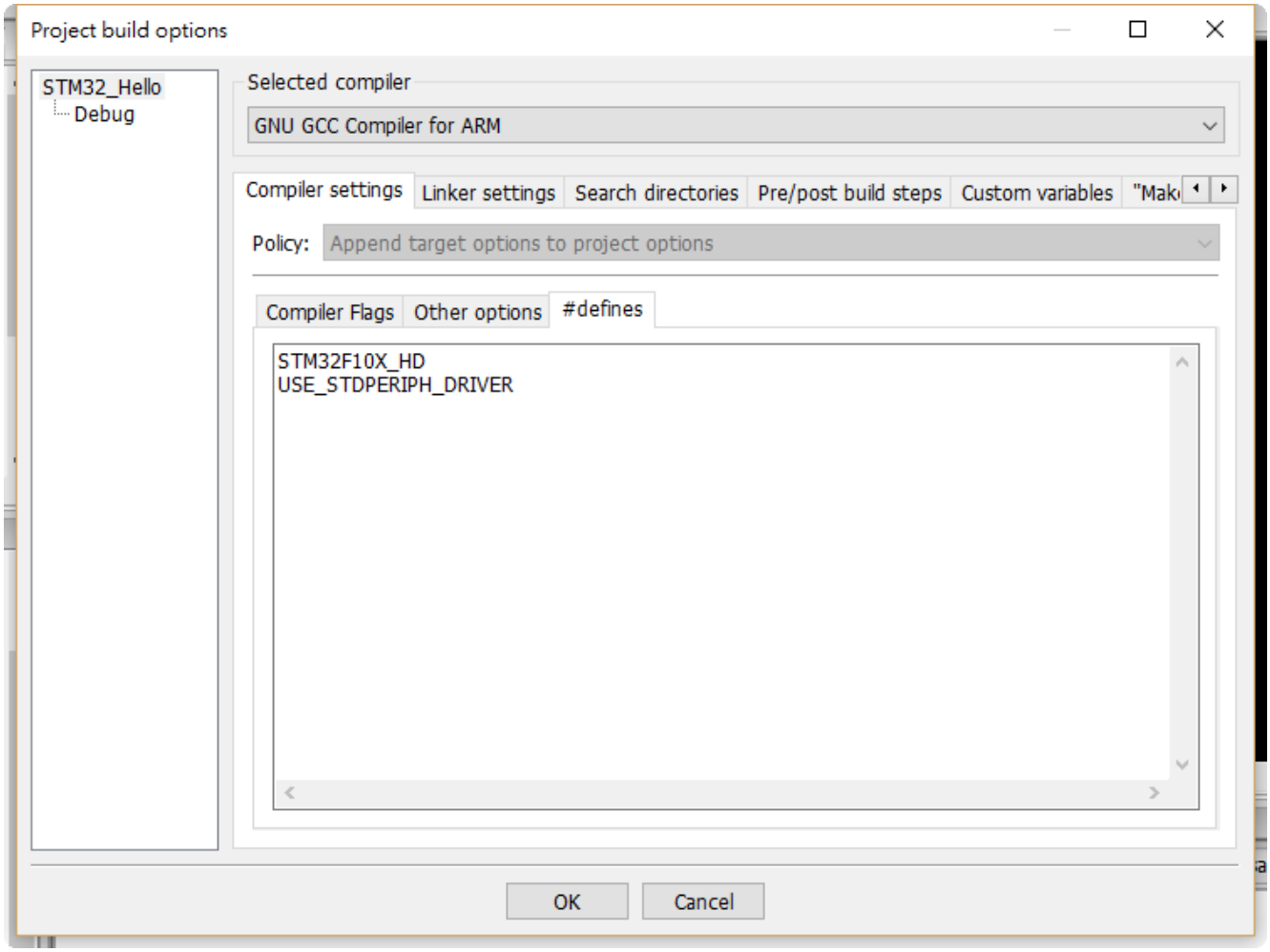


設定 compiler options

在 #defines tab 下要增加

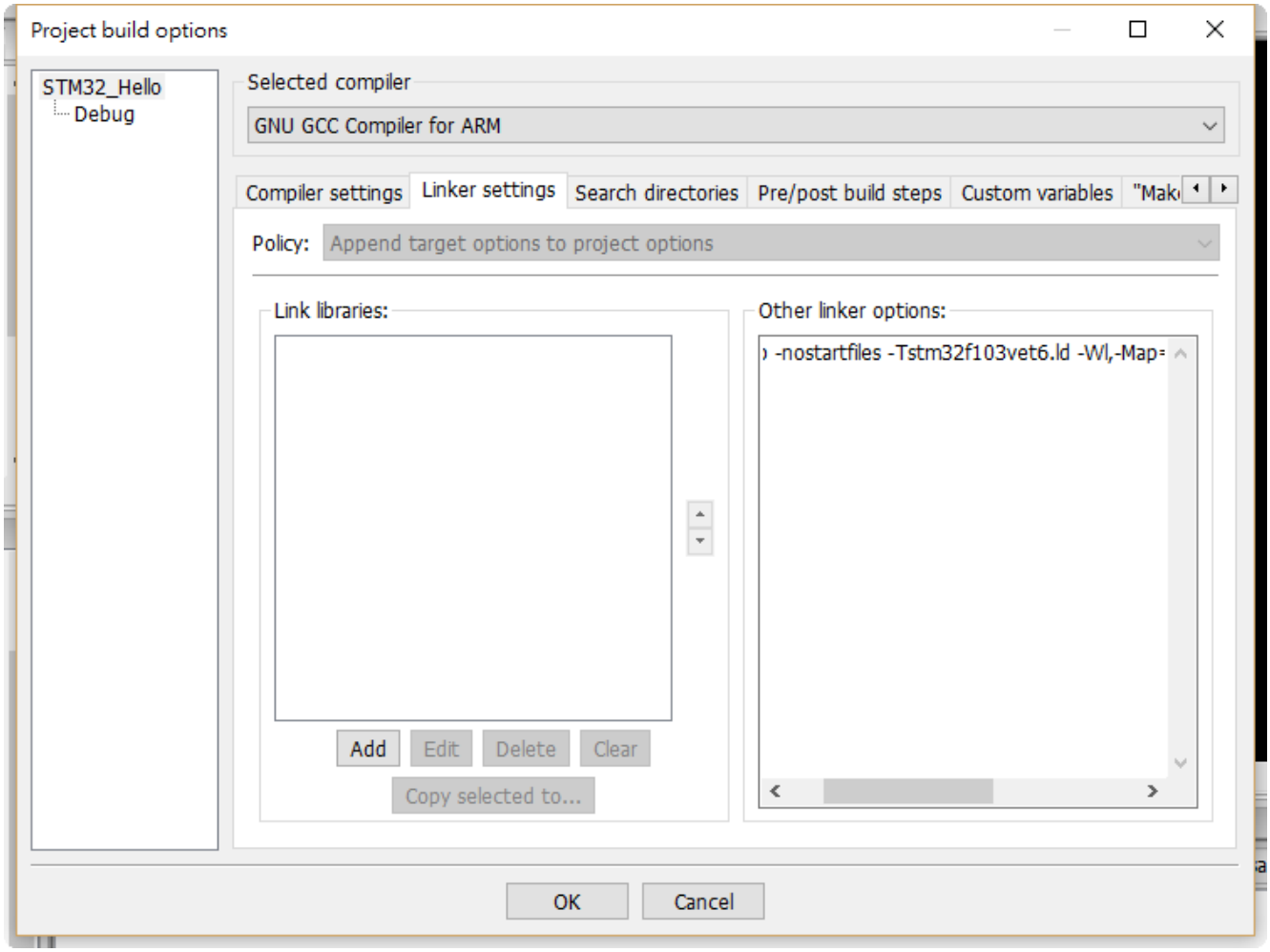
STM32F10X_HD

USE_STDPERIPH_DRIVER



設定 linker options

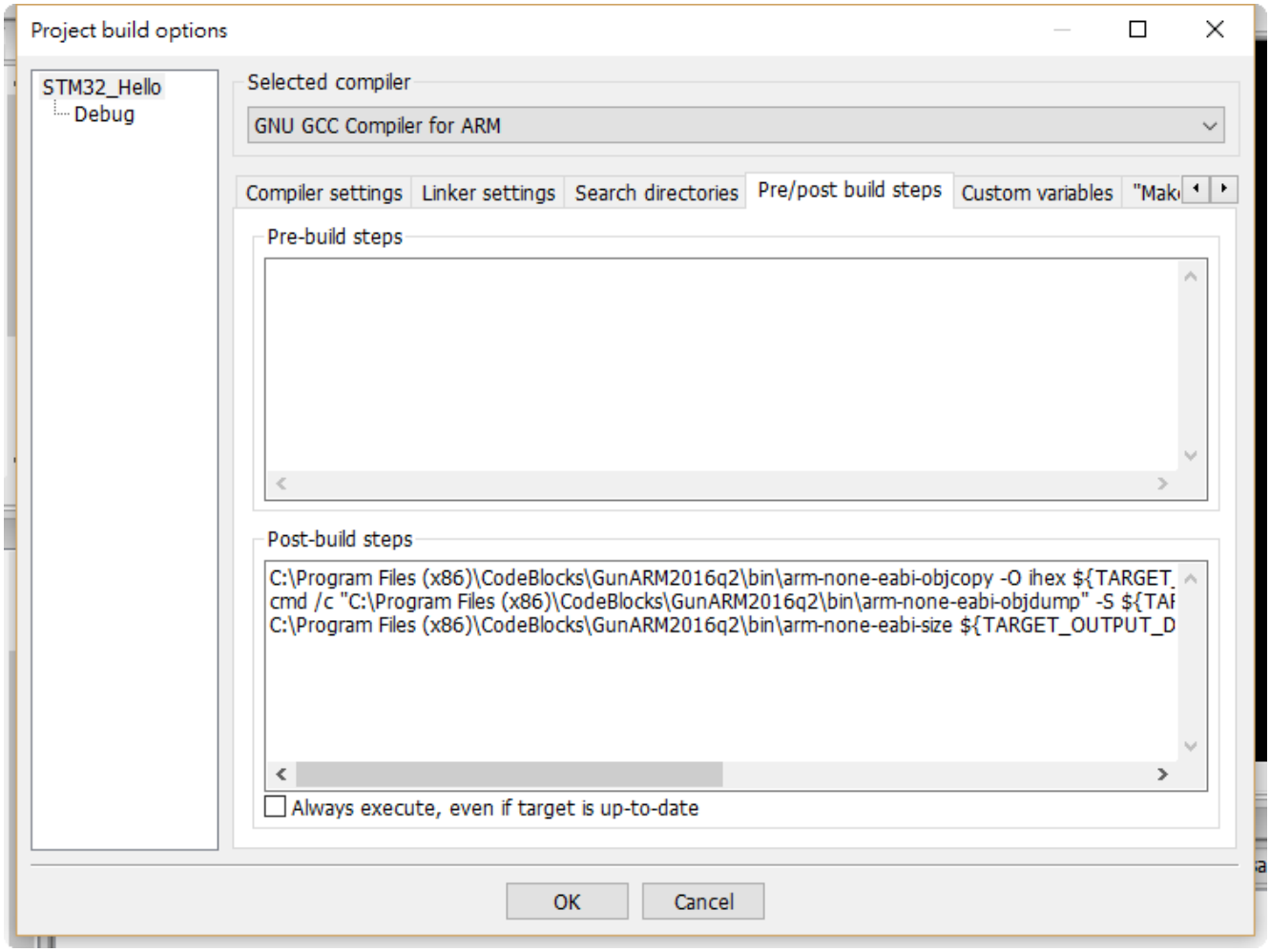
在 Linker settings 的頁籤下，other linker options 要新增
“-mthumb -nostartfiles -Tstm32f103vet6.ld
-Wl,-Map=main.map,--cref,--no-warn-mismatch”



設定 pre/post build step

最後是 Pre/post build steps 頁籤項目裡，Post-build steps 要增加

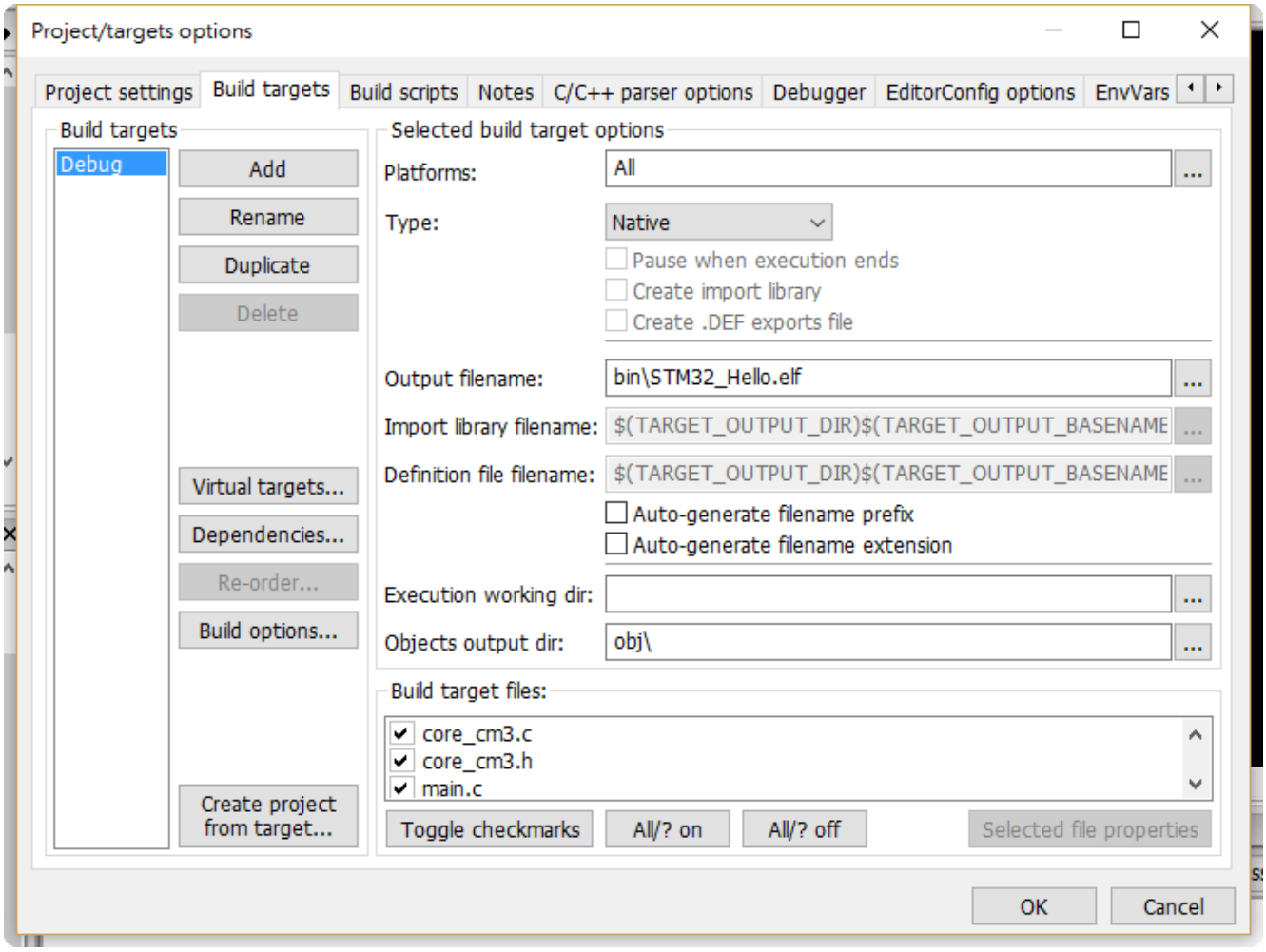
```
C:\GunARM2016q2\bin\arm-none-eabi-objcopy -O ihex
${TARGET_OUTPUT_DIR}${PROJECT_NAME}.elf  ${TARGET_OUTPUT_DIR}${PROJECT_NAME}.hex
C:\GunARM2016q2\bin\arm-none-eabi-objcopy -O binary -R .note -R .comment
${TARGET_OUTPUT_DIR}${PROJECT_NAME}.elf  ${TARGET_OUTPUT_DIR}${PROJECT_NAME}.bin
cmd /c "C:\GunARM2016q2\bin\arm-none-eabi-objdump" -S
${TARGET_OUTPUT_DIR}${PROJECT_NAME}.elf > ${TARGET_OUTPUT_DIR}${PROJECT_NAME}.asm
C:\GunARM2016q2\bin\arm-none-eabi-size ${TARGET_OUTPUT_DIR}${PROJECT_NAME}.elf
```



這段是為了在編譯完成後，自動產生可燒錄檔以及組合語言碼方便找錯誤。

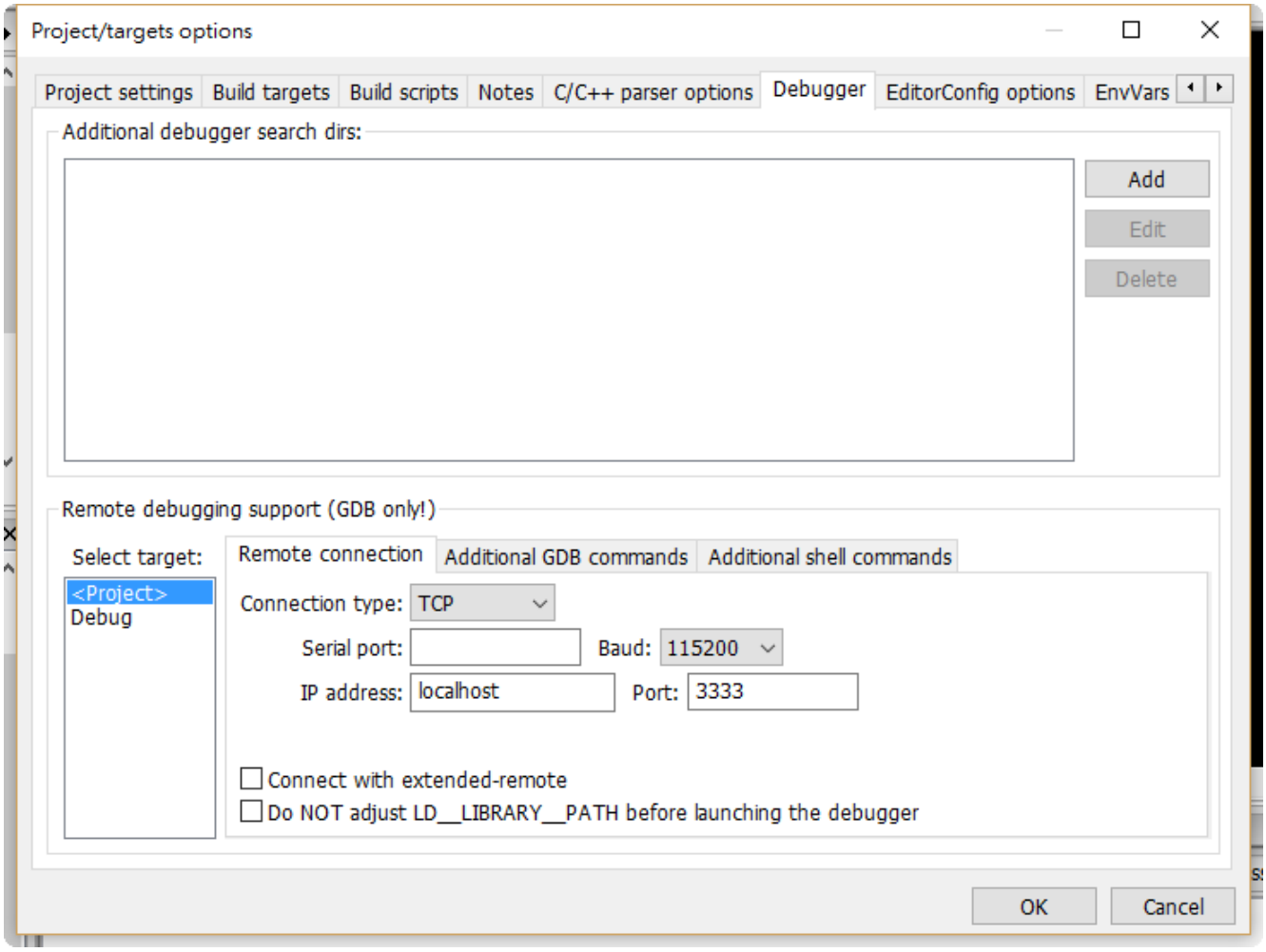
設定 Target

接著進入 Project-> Properties ，在 Build targets 頁籤裡做這樣的設定



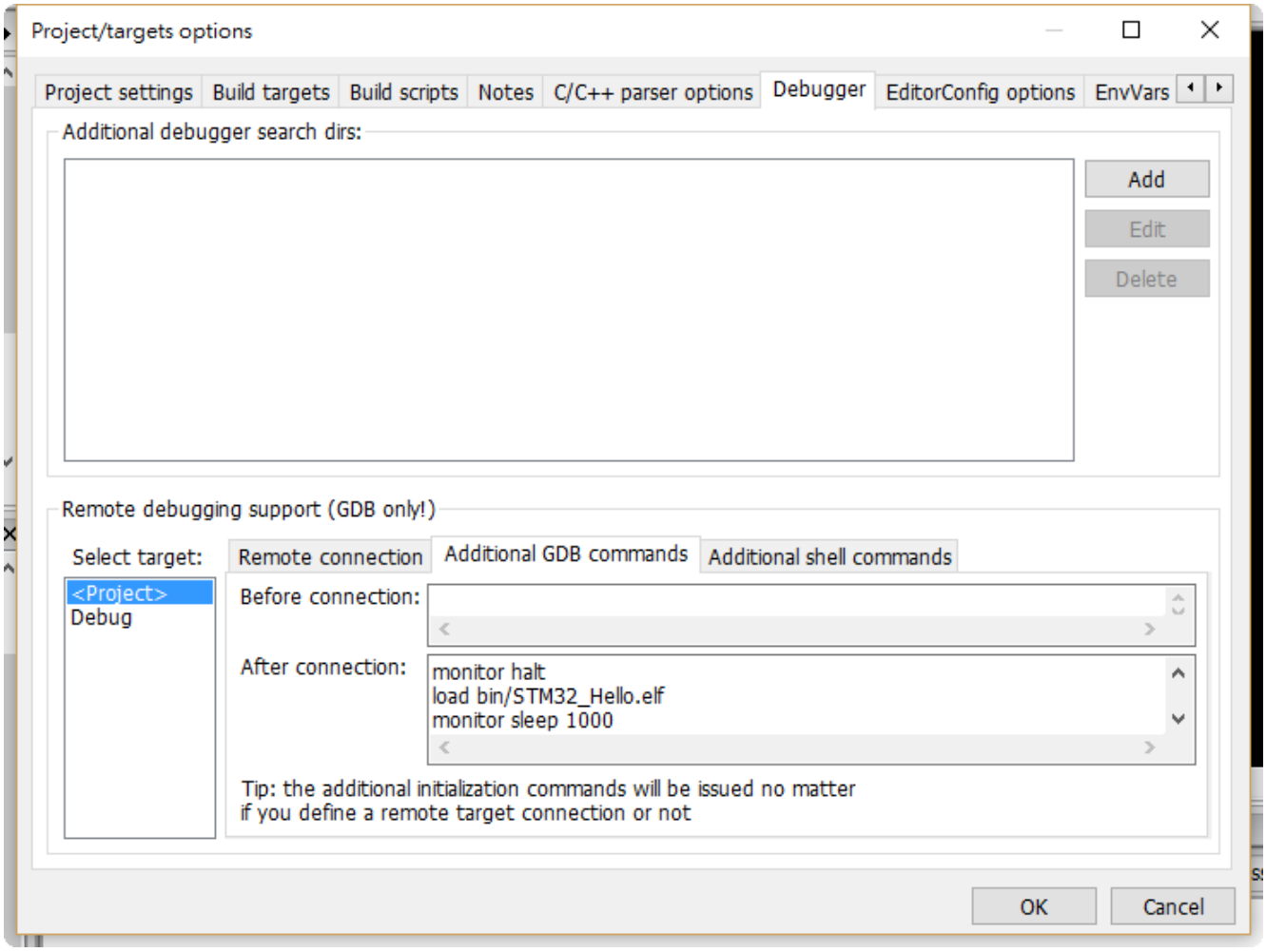
設定 Debugger

在 Debugger 頁籤加上遠端遙控的設定， **IP 位址是 localhost 而 port 是 3333**。



同樣在 Debugger 頁籤下， Additional GDB commands 新增連線後的指令設定

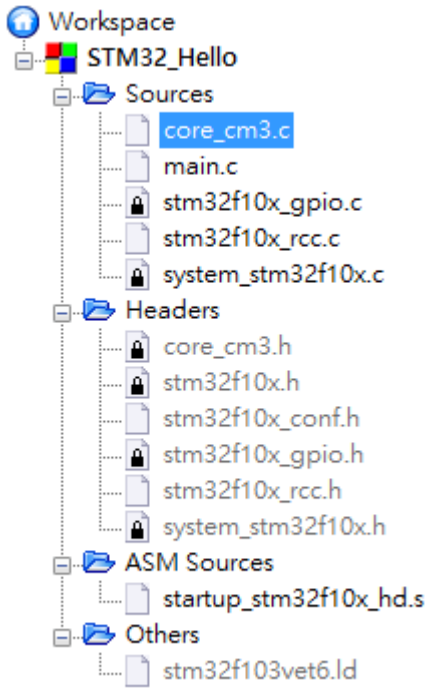
```
monitor halt
load bin/STM32_Hello.elf
monitor sleep 1000
monitor reset
```



這段指令的意思就是，啟動後先停止，然後載入執行檔，睡眠 1000mS 再 Reset。

Tips

載入的檔案有



core_cm3.c
core_cm3.h
system_stm32f10x.c
system_stm32f10x.h
stm3210x.h
stm32x.conf.h
startup_stm32f10x_hd.s
stm32f103vet6.ld
以上構成基本的開機啟動功能。

然後 main.c　放我們自己的程式
stm32f10x_gpio.c 和　stm32f10x_gpio.h 做 IO 操作
stm32f10x_rcc.c 和　stm32f10x_rcc.h 做振盪器控制操作

但是在 ARM compiler 裡，似乎有個辨識上的 bug　會使得 compiler 不會過關。
這兩個小小需要修改的點在 core_cm3.c 裡面。分別是

STREXB 和 STREXH

```
/**
 * @brief  STR Exclusive (16 bit)
 *
 * @param  value  value to store
 * @param  *addr  address pointer
 * @return        successful / failed
 *
 * Exclusive STR command for 16 bit values
 */
uint32_t __STREXH(uint16_t value, uint16_t *addr)
{
   uint32_t result=0;

   __ASM volatile ("strexh %0, %2, [%1]" : "=&r" (result) : "r" (addr), "r" (va
   return(result);
}

/**
 * @brief  STR Exclusive (32 bit)
 *
 * @param  value  value to store
 * @param  *addr  address pointer
```

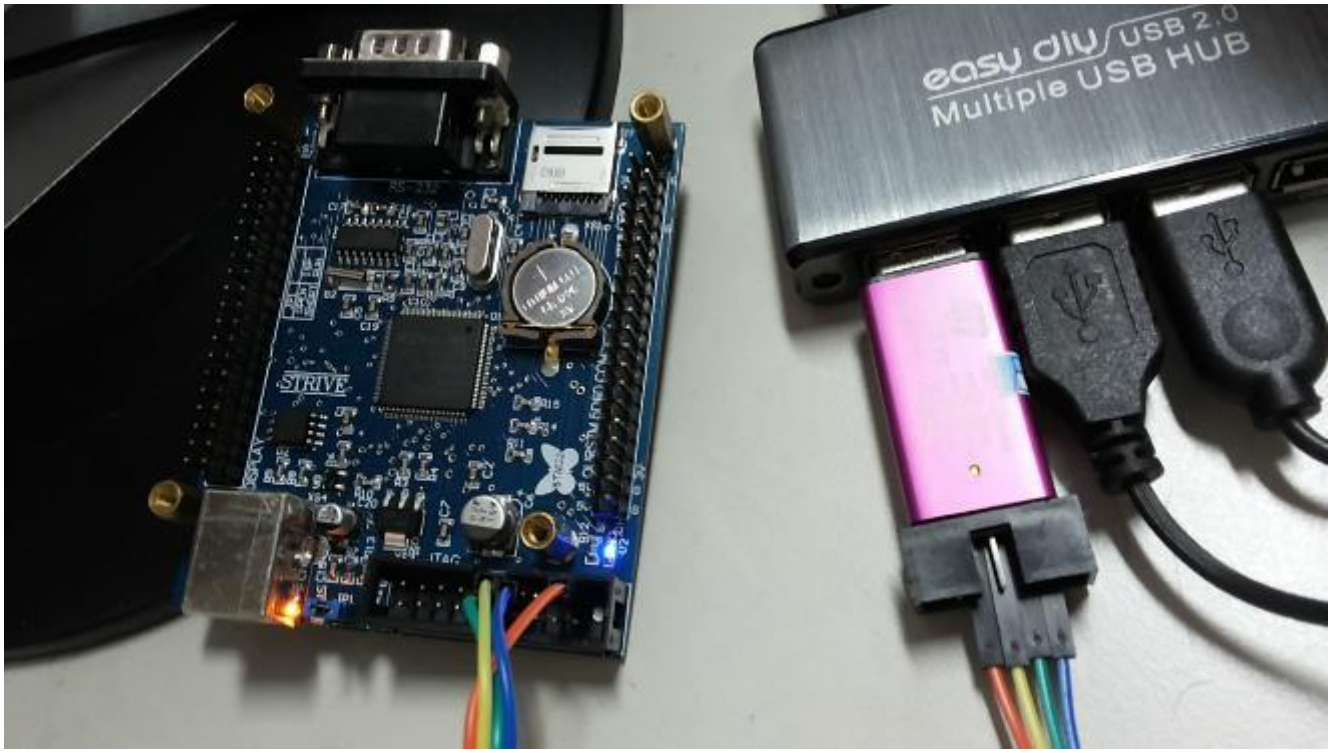
```
726  * @param  value  value to store
727  * @param  *addr  address pointer
728  * @return        successful / failed
729  *
730  * Exclusive STR command for 8 bit values
731  */
732 uint32_t __STREXB(uint8_t value, uint8_t *addr)
733 {
734    uint32_t result=0;
735
736    __ASM volatile ("strexb %0, %2, [%1]" : "=&r" (result) : "r" (addr), "r" (va
737    return(result);
738 }
739
740 /**
741  * @brief  STR Exclusive (16 bit)
742  *
743  * @param  value  value to store
744  * @param  *addr  address pointer
745  * @return        successful / failed
746  *
747  * Exclusive STR command for 16 bit values
748  */
```

在程式內 `"=r" (result)` 會導致編譯失敗，因此改為 `"=&r" (result)`
新增一個 & 符號解決。
該問題的修改法參考來自此
http://www.cesareriva.com/fix-registers-may-not-be-the-same-error/

以上的程式建置進專案後，build 完成，就可以啟動單步除錯器了。

實務操作

先接好硬體



透過 Tools 啟動 OpenOCD

```
C:\OpenOCD\bin\openocd.exe -f C:\OpenOCD\scripts\interface\stlink-v2.cfg -f C:\OpenOCD\scripts\target\stm32f1x_stlink.cfg
GNU ARM Eclipse 64-bits Open On-Chip Debugger 0.10.0-dev-00287-g85cec24-dirty (2016-01-10-10:13)
Licensed under GNU GPL v2
For bug reports, read
        http://openocd.org/doc/doxygen/bugs.html
WARNING: target/stm32f1x_stlink.cfg is deprecated, please switch to target/stm32f1x.cfg
Info : auto-selecting first available session transport "hla_swd". To override use 'transport select <transport>'.
Info : The selected transport took over low-level target control. The results might differ compared to plain JTAG/SWD
adapter speed: 1000 kHz
adapter_nsrst_delay: 100
none separate
Info : Unable to match requested speed 1000 kHz, using 950 kHz
Info : Unable to match requested speed 1000 kHz, using 950 kHz
Info : clock speed 950 kHz
Info : STLINK v2 JTAG v17 API v2 SWIM v4 VID 0x0483 PID 0x3748
Info : using stlink api v2
Info : Target voltage: 3.226333
Info : stm32f1x.cpu: hardware has 6 breakpoints, 4 watchpoints
```

當出現這些訊息時，代表 OpenOCD 已經成功啟動。等待 GDB 去做連線。
這時只要按下 紅色的三角鍵，Code Blocks 就會啟動 GDB 去做連線。按下後 log 視窗會出現

```
[debug]> monitor halt

In ?? () ()

[debug]> monitor halt
[debug]stm32f1x.cpu: target state: halted
[debug]target halted due to debug-request, current mode: Thread
[debug]xPSR: 0x21000000 pc: 0x080002ba msp: 0x2000fff8
[debug]>>>>>>cb_gdb:>>>>>>cb_gdb:
[debug]> load bin/STM32_Hello.elf
[debug]Loading section .isr_vector, size 0x1e4 lma 0x8000000
[debug]Loading section .text, size 0x840 lma 0x80001e4
[debug]Loading section .data, size 0x28 lma 0x8000a24
[debug]Start address 0x80001e4, load size 2636
[debug]Transfer rate: 9 KB/sec, 878 bytes/write.
[debug]>>>>>>cb_gdb:
[debug]> monitor sleep 1000
[debug]>>>>>>cb_gdb:
[debug]> monitor reset
[debug]>>>>>>cb_gdb:
[debug]> break "D:/MCUWorkSpace/STM32/STM32_Hello/main.c:36"
[debug]Breakpoint 2 at 0x8000a14: file main.c, line 36.
[debug]>>>>>>cb_gdb:
[debug]> break "D:/MCUWorkSpace/STM32/STM32_Hello/main.c:41"
[debug]No line 41 in file "D:/MCUWorkSpace/STM32/STM32_Hello/main.c".
[debug]Breakpoint 3 ("D:/MCUWorkSpace/STM32/STM32_Hello/main.c:41") pending.
[debug]>>>>>>cb_gdb:
[debug]> info registers
[debug]r0             0x0   0
[debug]r1             0x0   0
```

可以看到 load 指令後載程式到晶片的資訊。再按一次紅色的三角鍵就會開始運作，跑到斷點才會停止

```
24    GPIO_Init(GPIOB, &GPIO_InitStructure);
25
26    /* Set PB5  */
27    GPIOB->BSRR = 0x20;
28
29    delay();
30
31    while (1)
32    {
33      /* Set PB5  */
34      GPIOB->BSRR = 0x20;
35
36      delay();
37
38      /* Reset PB5 */
39      GPIOB->BRR  = 0x20;
40
41      delay();
42
43    }
44  }
```

當斷點停止的時候，可以在 OpenOCD 的視窗內看到被停止的位址

```
C:\OpenOCD\bin\openocd.exe -f C:\OpenOCD\scripts\interface\stlink-v2.cfg -f C:\OpenOCD\scripts\target\stm32f1x_stlink.cfg
stm32f1x.cpu: target state: halted
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x080001e4 msp: 0x20010000
stm32f1x.cpu: target state: halted
target halted due to breakpoint, current mode: Thread
xPSR: 0x61000000 pc: 0x2000003a msp: 0x20010000
stm32f1x.cpu: target state: halted
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x080001e4 msp: 0x20010000
Warn : WARNING! The target is already running. All changes GDB did to registers will be discarded! Waiting for target to
 halt.
Info : halted: PC: 0x080002b4
Info : halted: PC: 0x08000a14
Info : halted: PC: 0x080002b4
Info : halted: PC: 0x08000a14
Info : halted: PC: 0x080002b4
Info : halted: PC: 0x08000a14
Info : halted: PC: 0x080002b4
Info : halted: PC: 0x080002b4
Info : halted: PC: 0x080002b4
Info : halted: PC: 0x080002b4
Info : halted: PC: 0x080002b4
Info : halted: PC: 0x08000a1a
Info : halted: PC: 0x08000a0e
Info : halted: PC: 0x080002b4
Info : halted: PC: 0x08000a14
Info : halted: PC: 0x080002b4
Info : halted: PC: 0x08000a1a
Info : halted: PC: 0x08000a0e
Info : halted: PC: 0x080002b4
```

以上範例程式碼放在 GitHub

Code Blocks 的操作設定參考自 http://www.hackvandedam.nl/blog/?p=707

OpenOCD 的操作可參考自 詹姆士酷斯拉