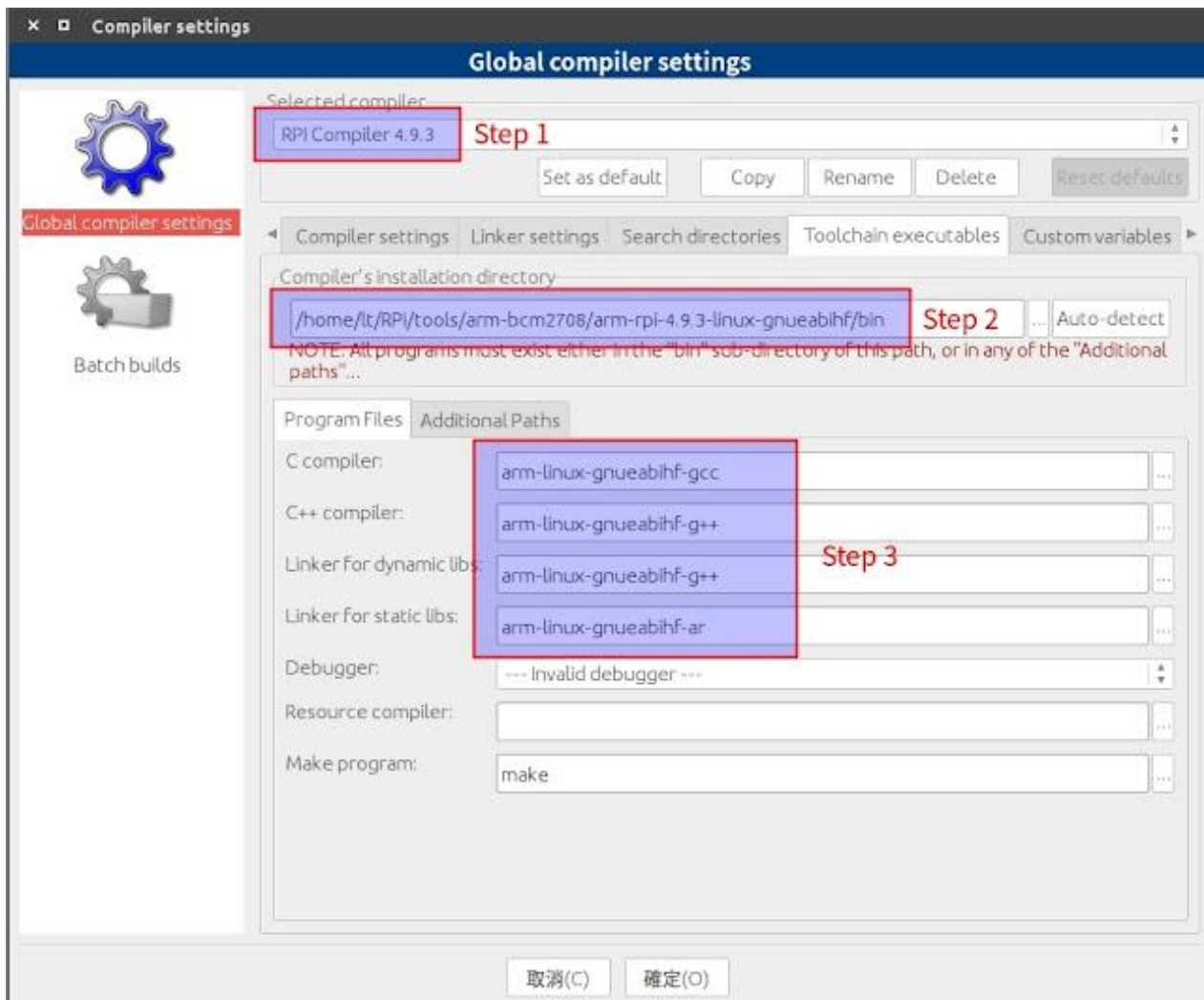


[如何設定 Code::Blocks 遠端除錯 Raspberry Pi](#)

這篇文章的 Host 是 Ubuntu 16.04 LTS，Target 是 Raspberry Pi 2 B-Type。主要說明如何在 Code::Blocks 16.01 上設定 Corross Compiler 及 Remote Debug 的相關選項。當設定成功後可以在 Host 上開發 Target 的 Applications，充份利用 Host 的運算能力不再受限於開發板的速度。

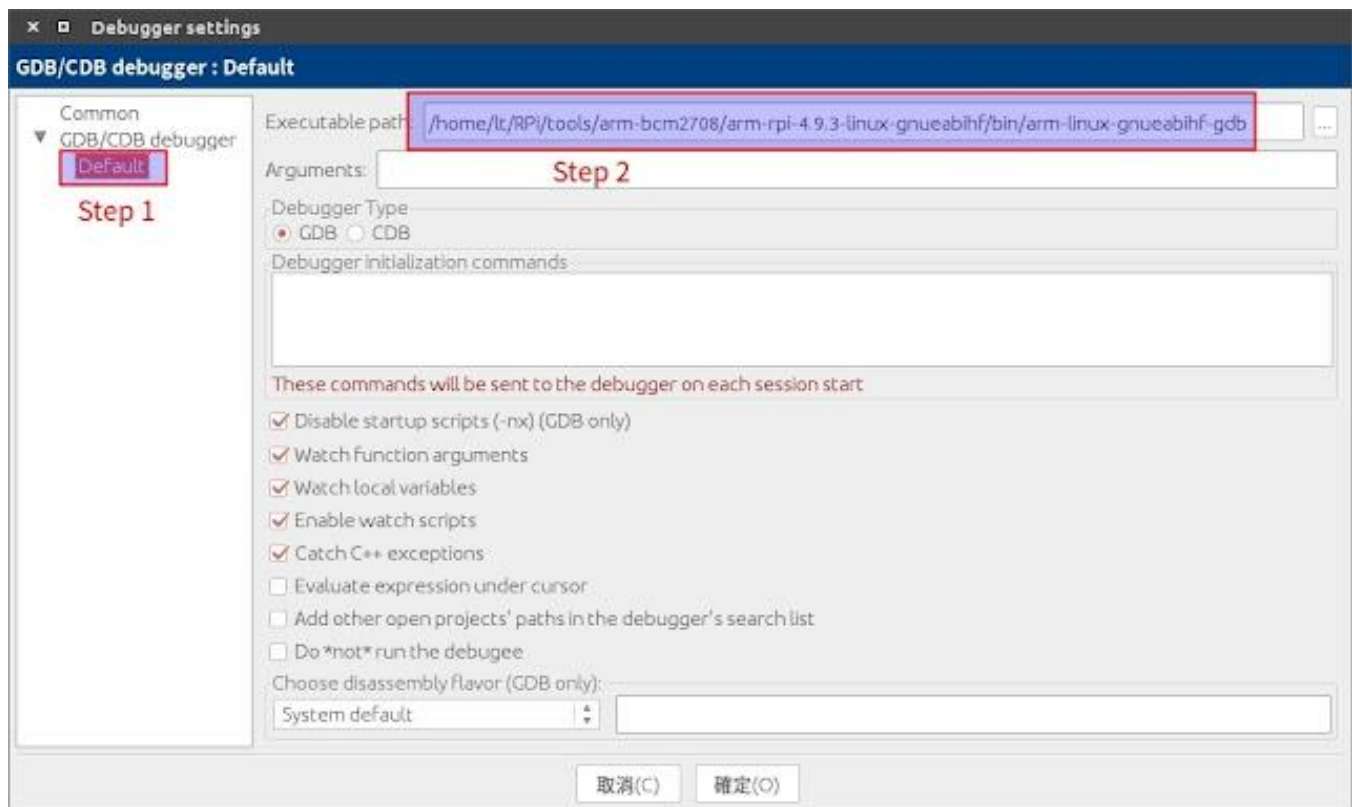
1. 在 Host 下載 RPi 官方提供的 [GitHub's Tools](#) 套件。
2. 在 C::B 的 Settings\Compiler\開啟如下視窗，並依照相關 Step 操作設定。
 - Step1：設定一個新的 Compiler 選項。
 - Step2：從項 1 下載回來的 Tools 裡選定一個路徑，本例是使用 4.9.3 這個路徑下的工具組。
 - Step3：選好各個相關 Program 的名稱。



3. 在 C::B 的 Settings\Debugger\開啟如下視窗，並依照相關 Step 做操作設定。

Step1：選擇 GDB/CDB Default 項目，出現如下畫面。

Step2：選擇要執行的 gdb 程式完整路徑與名稱(預設完整路徑名稱是/usr/bin/gdb)。

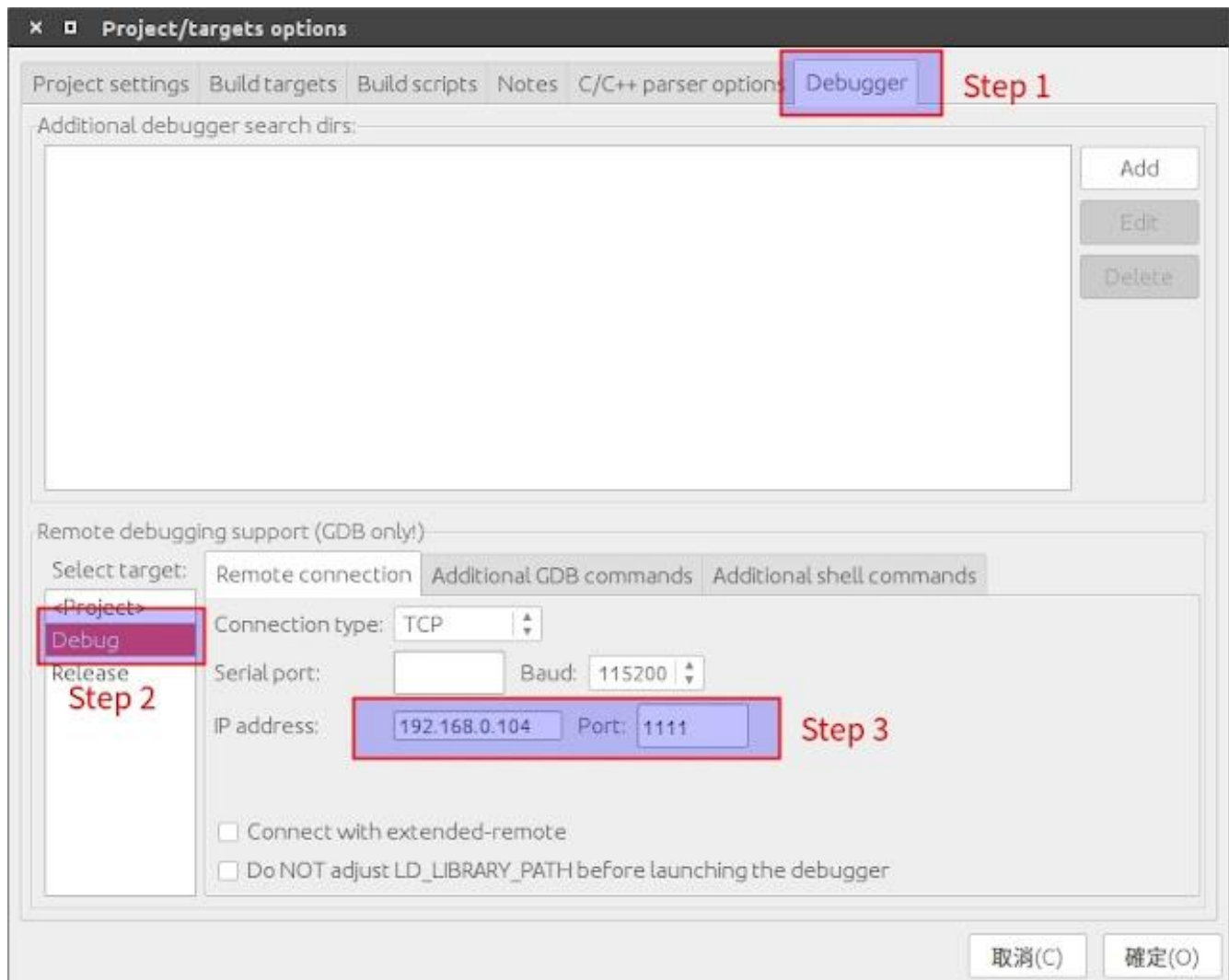


4. 產生新的 C++ Project 並寫好程式碼後，在 Code::Blocks 的 Project\Properties\ 開啟如下視窗，並依照相關 Step 做操作設定。

Step 1：切到 Debugger 頁面。

Step2：選擇 Debug 項目。

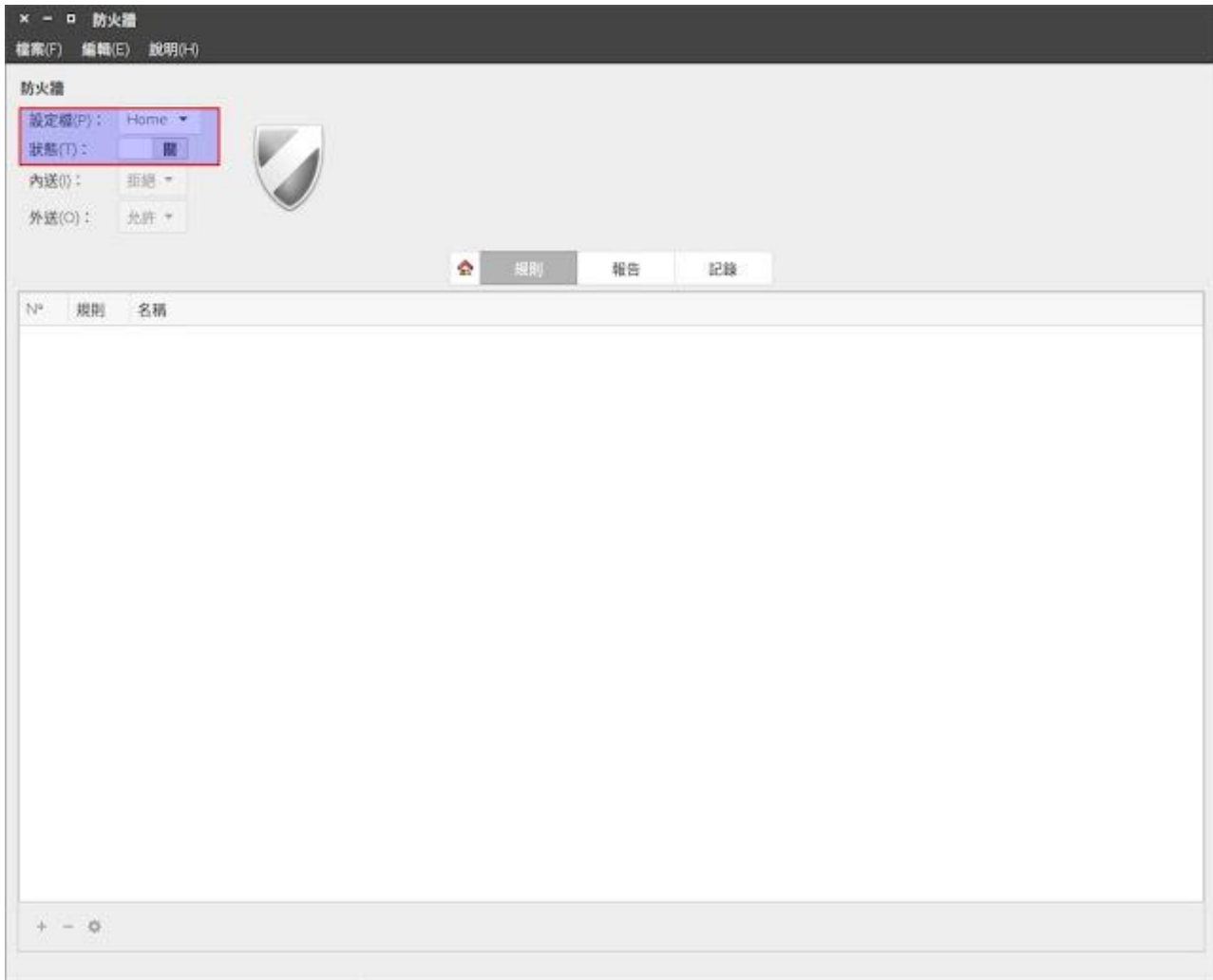
Step3：設定 Target 的 IP 與工作 Port，未來 Host Port 要和此 Port 相匹配才能除錯。



5. 基本上；上述 4 項動作完成也沒有發生錯誤的話，應該能在 Host 上 Compiler 出 Target 可執行的程式。

6. Host 和 Target 的 NFS 分享與掛載可以參考此篇說明。當掛載完成後，Target 可以直接在 Console 執行位於 Host 上的程式。

7. 若 Host 有設防火牆，可利用其 GUI 程式將之關閉，以方便兩機間除錯。



8. 將 Host 上；項 1 剛下載回來的 Tools 相關路徑下的 gdbserver 程式，拷貝到 Target 的 /usr/bin/gdbserver 下。如不知位置可以用

```
$ sudo find / -name 'gdbserver'
```

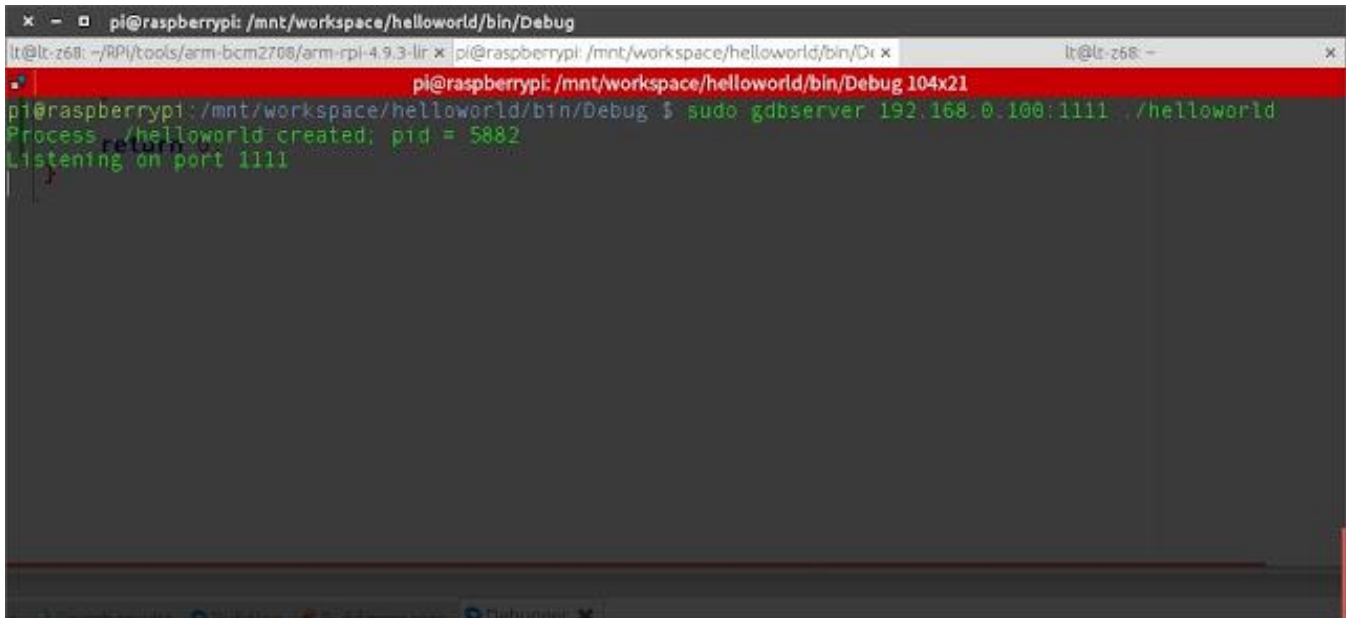
搜尋尋找(我沒有 copy 過去也能正常除錯)。

9. 先在 Target 上執行如下命令。

```
$ sudo gdbserver IP:Port ./程式名稱
```

接著就等 Host 的 C::B 連線進行除錯動作。

用 cat /proc/5882/maps 可看到 AP 使用記憶體狀況。



```
pi@raspberrypi: /mnt/workspace/helloworld/bin/Debug
pi@raspberrypi: /mnt/workspace/helloworld/bin/Debug 104x21
pi@raspberrypi: /mnt/workspace/helloworld/bin/Debug $ sudo gdbserver 192.168.0.100:1111 ./helloworld
Process ./helloworld created: pid = 5882
Listening on port 1111
```

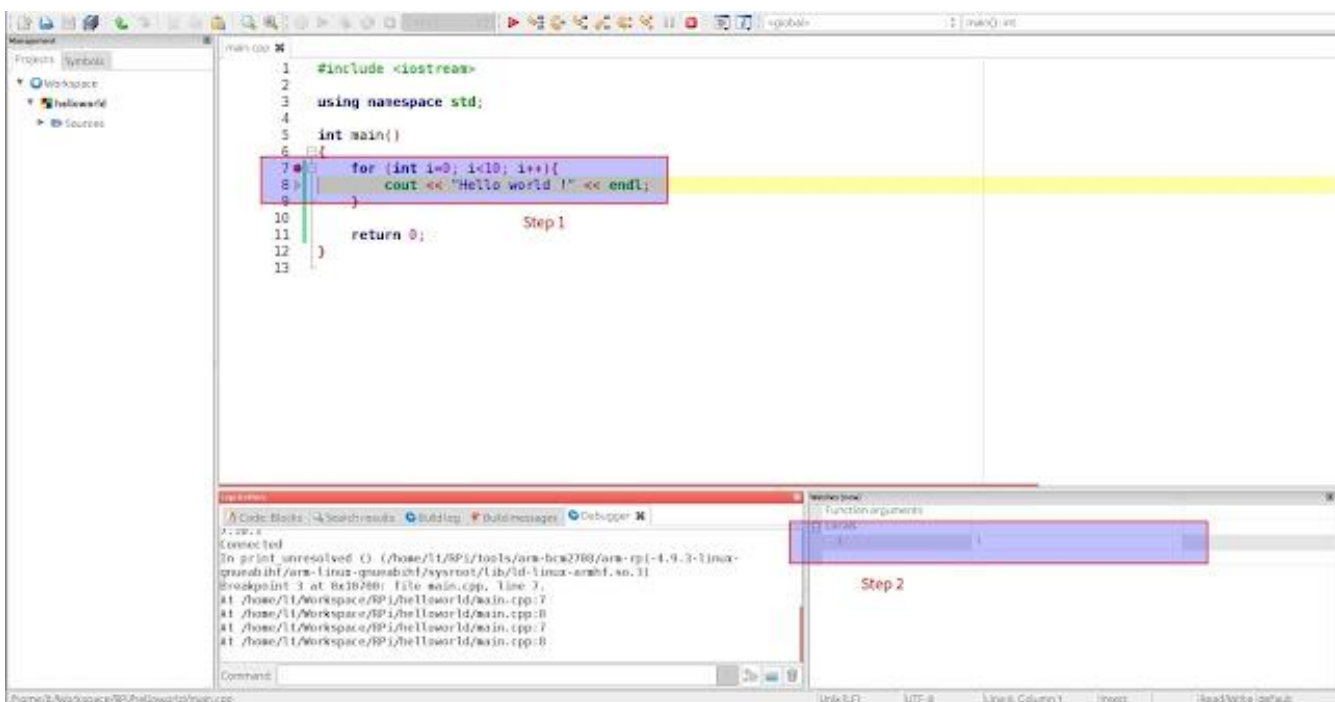
10. 除錯

Step 1：在 C::B 按 F5 設置斷點。

Step 2：在 Watches 視窗設定要看的變數，此例為 i。按 F8 開始除錯，程式會停在斷點上。

Step 3：按 F7 即可單步追蹤，Shift+F7 可 Step Into。

Step 4：Shift+F8 停止除錯。



目前這樣的設定可以除錯一般的 Application，至於 Kernel 或 Library 或 OpenCL 的除錯還沒試，不曉得可不可行？到時再補充相關資訊。