

# **Jini Intelligent Computing Workbook of Lab. #1**

## Preamble

在 Lab. #1 的檔案系統中主要有下列專案目錄：

- hls\_Multiplication  
Vitis HLS 乘法器原始碼檔案
- vvd\_Multip2Num  
範例乘法器 Vivado Design Suite 參考檔案
  - design\_1.tcl  
範例乘法器 Block Design 完成 Generate Bitstream 後匯出之 TCL Script 檔
  - MakeBit.bat  
範例乘法器完成 Generate Bitstream 後，將.bit/.hwh 拷貝至專案根目錄之批次檔
- ipy\_Multip2Num  
範例乘法器系統程式 Python 原始碼檔及 Jupyter Notebook 原始碼編輯檔

## 1. Board Setup

### 1.1. Create SD Card with Boot Image

【施作環境為在使用者 PC/laptop/notebook (Windows Base) 。】

至以下網址下載 PYNQ-Z2 開機映像檔 pynq\_z2\_v2.6.0.zip，並解壓縮至 PC。

<http://www.pynq.io/board.html>

#### Downloadable PYNQ images

If you have a Zynq board, you need a PYNQ SD card image to get started. You can download a pre-compiled PYNQ image from the table below. If an image is not available for your board, you can build your own SD card image (see details below).

Board	SD card image	Documentation	Vendor webpage
PYNQ-Z2	<b>v2.6</b>	<a href="#">PYNQ setup guide</a>	<a href="#">TUL Pynq-Z2</a>

再將 pynq\_z2\_v2.6.0.img 製作到 SD card。SD card 最小容量需求 8GB 以上。

1. 將 SD card 插入電腦中
2. 將 SD card 進行格式化。格式化格式選擇 FAT。
3. 下載 Win32 Disk Imager 軟體工具。

4. 燒錄 Image 到 SD card 中。如下圖：映像檔選擇 pynq\_z2\_v2.6.0.img，裝置選擇 SD 卡槽。



【施作環境為在使用者 PC/laptop/notebook (Linux Base) 。】

在下列的示範是以 Linux PC 在 console 的 terminal 環境下實作。主要目的為清除原 SD card 內容，再將 pynq\_z2\_v2.6.0.img 製作至 SD card。

在 Linux console 環境下執行下列命令：

\$> lsblk

這個指令會列出系統的可用 Block Device

```
sda      8:0    0 931.5G  0 disk
└─sda1   8:1    0 931.5G  0 part /
sdb      8:16   1  1.9G  0 disk
└─sdb1   8:17   1  1.9G  0 part /media/xilinx/482E-22E4
```

此時 SD card 的 volume drive 顯示為/media/xilinx/482E-22E4。根據不同的 SD 卡或電腦系統，有可能出現 sda/sdb/sdc... etc.之類的 device name。

\$> umount /media/xilinx/482E-22E4

\$> sudo dd if=/dev/zero of=/dev/sdb bs=4M count=1 status=progress

詳細 dd 指令教學請自行 Google 查詢使用教學。

\$> sync

將 SD card 從 PC 卡槽退出再重新插入 SD card。

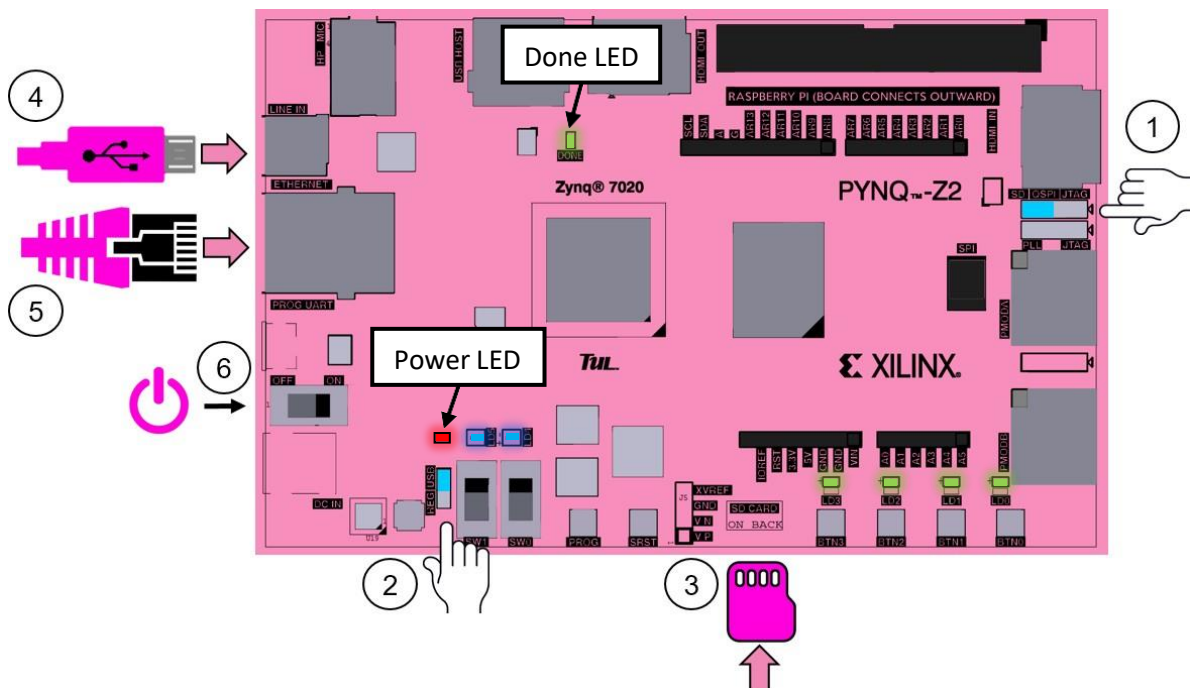
\$> lsblk

確定 SD card 裝置名稱為 sdb/sdc...。

再使用 dd 指令將路徑（~為使用者 home 目錄）及檔名 pynq\_z2\_v2.6.0.img 映像檔複製到 SD 卡上。

```
$> sudo dd if=~/.Downloads/pynq_z2_v2.6.0.img of=/dev/sdb bs=4M status=progress  
$> sync
```

## 1.2. Start up PYNQ-Z2



圖片來源及參考資料：

[https://pynq.readthedocs.io/en/latest/getting\\_started/pynq\\_z2\\_setup.html](https://pynq.readthedocs.io/en/latest/getting_started/pynq_z2_setup.html)

1. 將“Boot” jumper 設定在 SD 位置（如上圖 1 處）。
2. 如使用 micro USB 作為電源，將“Power” jumper 設定在 USB 位置（如上圖 2 處）。亦可使用外接 12V DC 電源（接頭為 2.1mm \* 5.5mm），此時將 Power jumper 設定在 REG 位置。
3. 將燒錄好 PYNQ-Z2 開機映像檔的 micro SD card 插入 PYNQ-Z2 主機背面的卡槽（如上圖 3 處）。
4. 將 USB cable 連接至使用者 PC 以及 PYNQ-Z2 主機上的 PROG-UART micro USB 插槽（如上圖 4 處）。

5. 連接乙太網路線至 PYNQ-Z2 上的 RJ-45 網路接口（如上圖 5 處）以及路由器（router），讓使用者 PC 與 PYNQ-Z2 屬於相同的 LAN（Local Area Network）區域網路內，即使用者 PC 與 PYNQ-Z2 隸屬於同一台 router。
6. 開啟 PYNQ-Z2 電源（上圖 6 處），打開後會亮起紅色 Power LED 代表電源開啟。數秒後綠色 Done LED 會亮起。再過幾秒，會看到兩個藍色 LED 和四個綠色 LED 閃爍數次，最後綠色 LED 會保持亮著，開機程序完成。

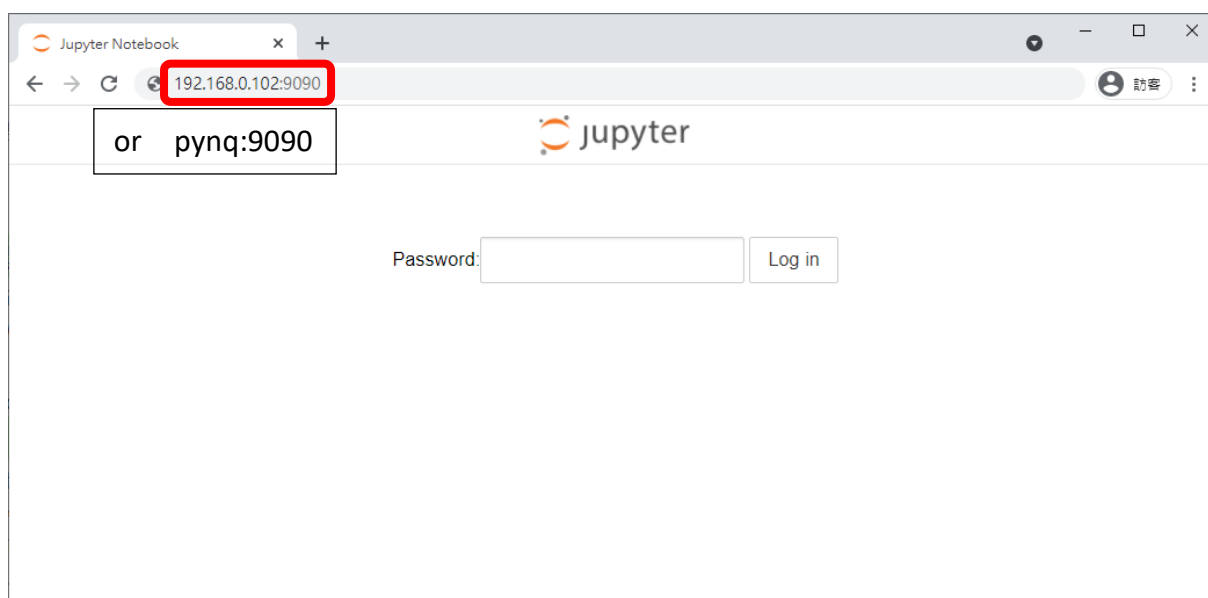
## 1.3. Connect with PC

### 1.3.1. Jupyter Notebook

【施作環境為在使用者 PC/laptop/notebook (Windows Base) 。】

Jupyter Notebook 是一個以瀏覽器介面來進行編輯程式碼的介面，同時可執行運行，運行的程式是在 PYNQ-Z2 的 Linux kernel 中運作。

啟動瀏覽器，以瀏覽器連線 PYNQ-Z2 主機 Jupyter Notebook Server 服務。連線 IP 為 PYNQ-Z2 主機被分配到的 IP address（取得方式見下節），或直接輸入 pynq；port 為 9090；登入密碼為 xilinx。



## 1.3.2. Terminal

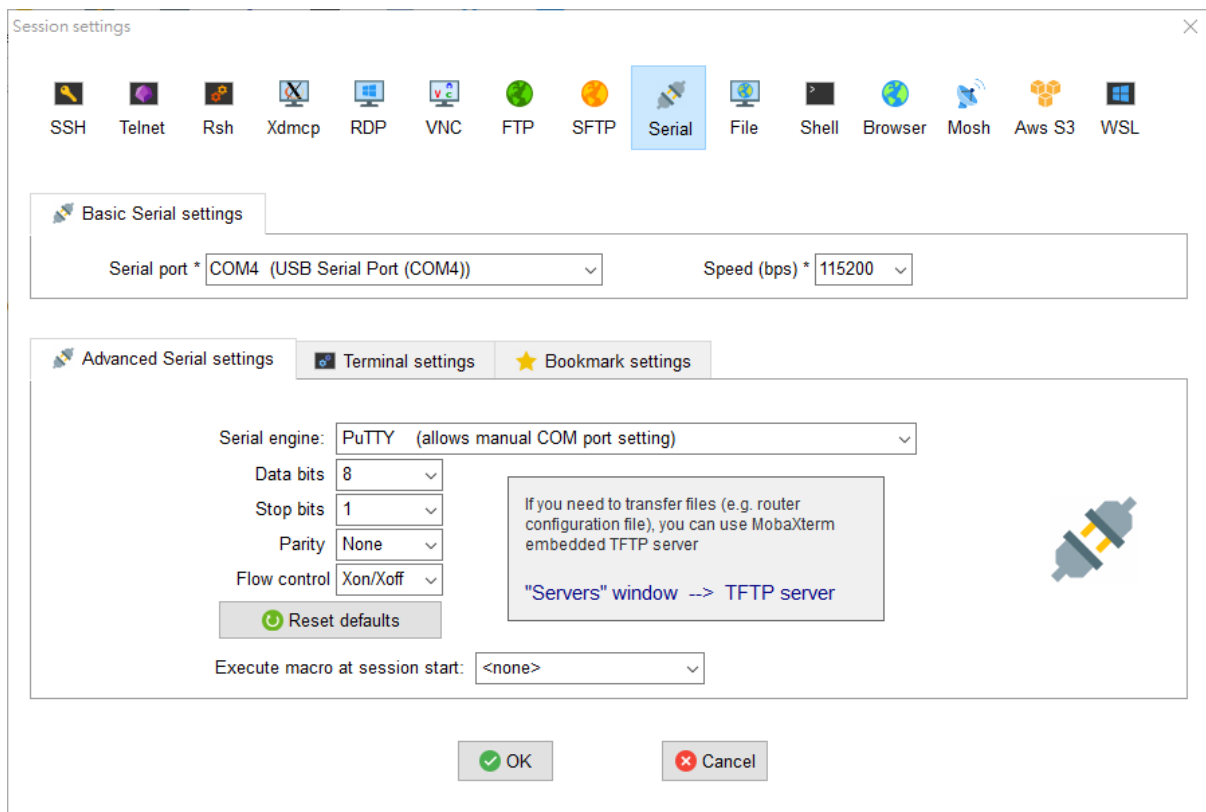
PYNQ-Z2 的 Terminal 有三種連線方式，可依需求選擇使用：

### 1.3.2.1. COM port

【施作環境為在 PYNQ-Z2 TTY console，由 PYNQ-Z2 Linux kernel 藉由 COM port 連線至使用者 PC/laptop/notebook (Windows Base) 。】

需要 TTY (Teletypewriter, Teleprinter) 軟體接收 console 訊息，TTY 是使用者與 PYNQ-Z2 溝通的媒介，透過 TTY 可以讓使用者在 PC 上藉由 HID key board 與 PYNQ-Z2 Linux kernel 做聯繫。

1. 開啟 MobaXterm，點選左上角 Session 後選取 Serial，Serial port 選取連線 FTDI USB Serial Converter 模組上的 COM port；Speed 設為 115200。



2. 在 terminal 中按一下 Enter 即可開始使用。或是重新開機並按 R 來 restart session 可看到以下開機訊息。由於設定上是自動登入所以無需輸入使用者名稱與密碼 (Username : xilinx ; Password : xilinx)

```

5 COM4 (USB Serial Port (COM4) x
Stopping Network Name Resolution...
[ OK ] Stopped Network Name Resolution.
Starting Network Name Resolution...
Starting resolvconf-pull-resolved.service...
[ OK ] Started Network Name Resolution.
[ OK ] Started resolvconf-pull-resolved.service.
[ OK ] Started Raise network interfaces.
[ OK ] Reached target Network.
Starting OpenBSD Secure Shell server...
Starting Permit User Sessions...
[ OK ] Started Unattended Upgrades Shutdown.
[ OK ] Reached target Network is Online.
[ OK ] Started ISC DHCP IPv6 server.
[ OK ] Started ISC DHCP IPv4 server.
Starting Samba NMB Daemon...
[ OK ] Started Permit User Sessions.
[ OK ] Started Serial Getty on ttyPS0.
[ OK ] Started Getty on tty1.
[ OK ] Reached target Login Prompts.

PYNQ Linux, based on Ubuntu 18.04 pynq ttyPS0

pynq login: xilinx (automatic login)

Last login: Sun Jun 27 18:28:58 UTC 2021 on ttyPS0
Welcome to PYNQ Linux, based on Ubuntu 18.04 (GNU/Linux 5.4.0-xilinx-v2020.1 armv7l)

* Super-optimized for small spaces - read how we shrank the memory
  footprint of MicroK8s to make it the smallest full K8s around.

https://ubuntu.com/blog/microk8s-memory-optimisation
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

xilinx@pynq:~$

```

輸入指令 `ifconfig`，可取得 PYNQ-Z2 在所屬 LAN 被分配到的 IP address。

```

xilinx@pynq:~$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.0.102 netmask 255.255.255.0 broadcast 192.168.0.255
    inet6 fe80::205:00ff:fe01:9f91 prefixlen 64 scopeid 0x20<link>
    ether          txqueuelen 1000 (Ethernet)
    RX packets 2613 bytes 494742 (494.7 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 2567 bytes 2745454 (2.7 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
    device interrupt 28 base 0xb000

eth0:1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.2.99 netmask 255.255.255.0 broadcast 192.168.2.255
    ether          txqueuelen 1000 (Ethernet)
    device interrupt 28 base 0xb000

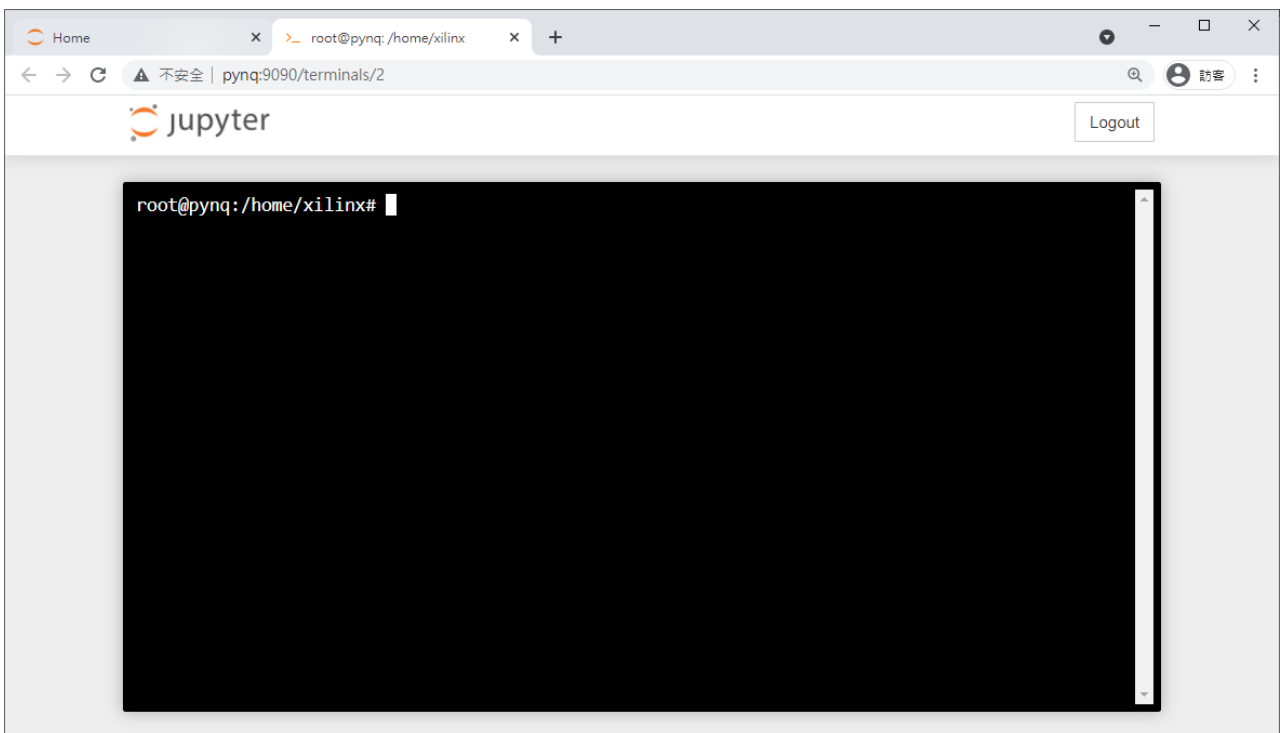
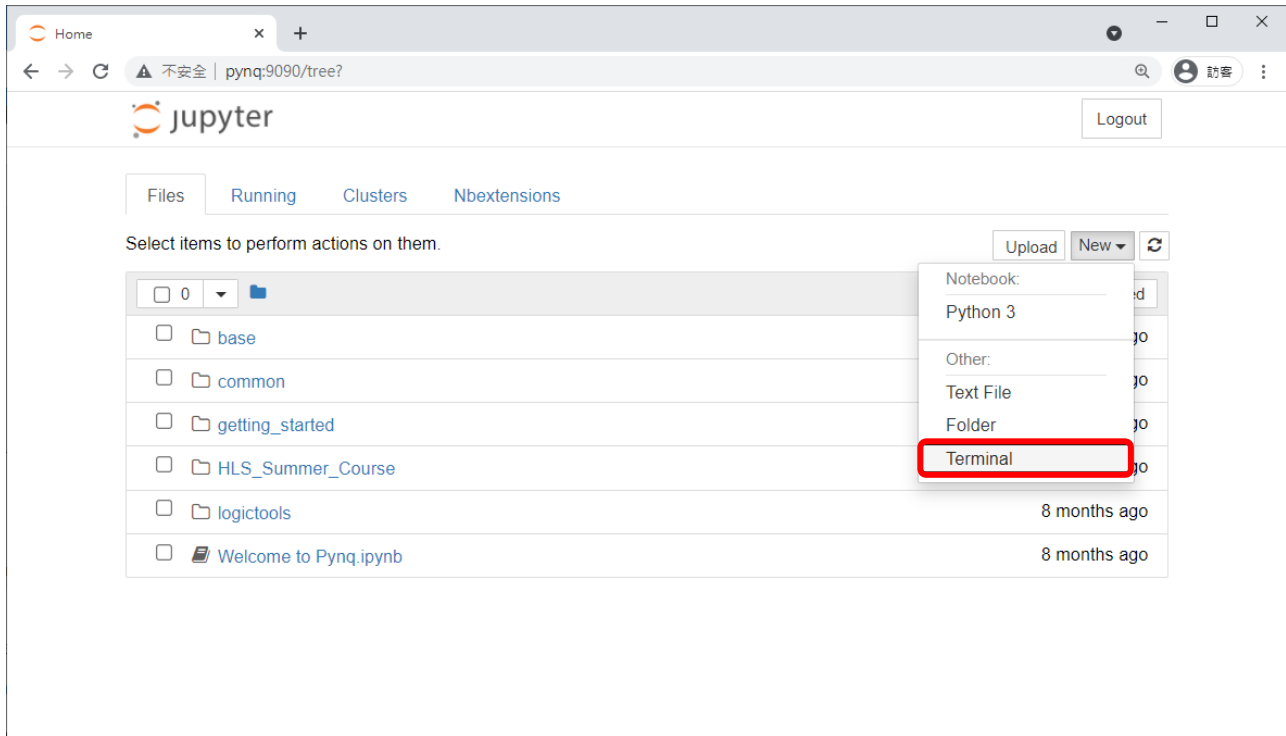
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 154 bytes 18564 (18.5 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 154 bytes 18564 (18.5 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

```

### 1.3.2.2. Jupyter Notebook

【施作環境為在使用者 PC/laptop/notebook (Windows Base) 。】

亦可從 Jupyter Notebook 開啟 terminal，登入後點選 New → Terminal



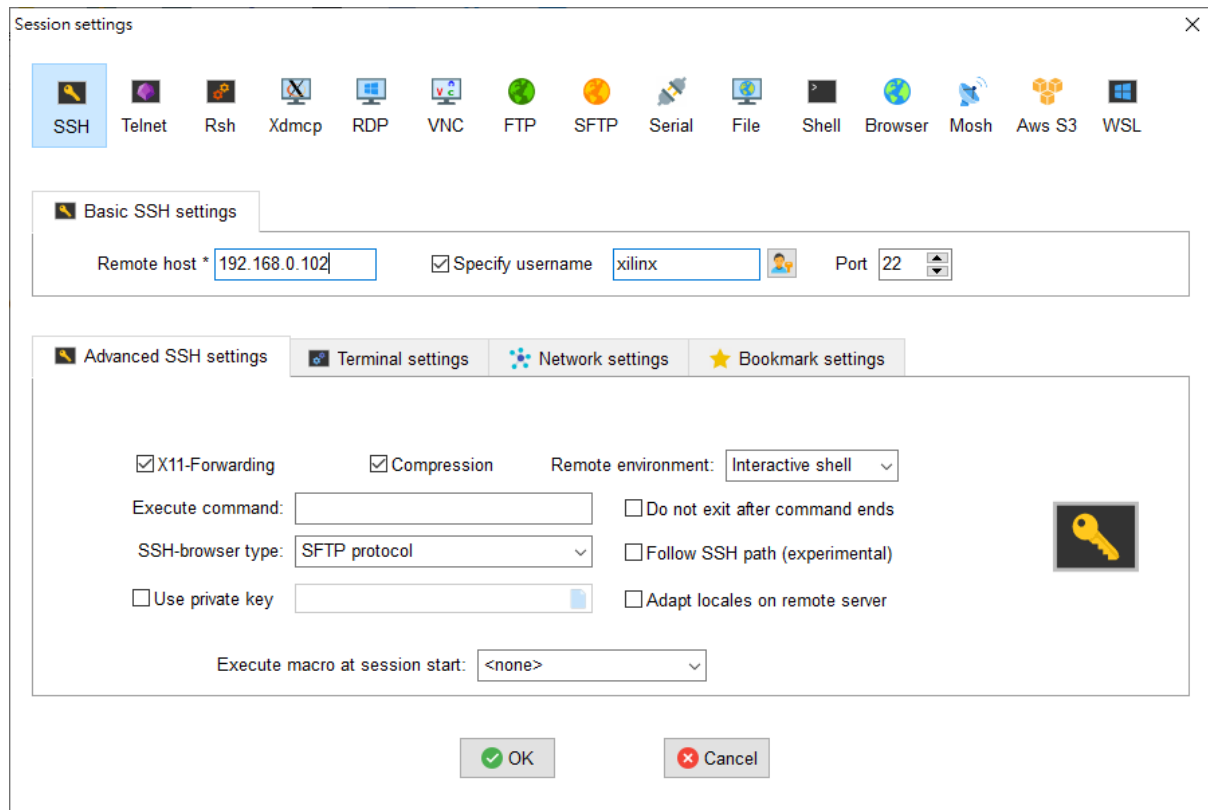


### 1.3.2.3. SSH Connection

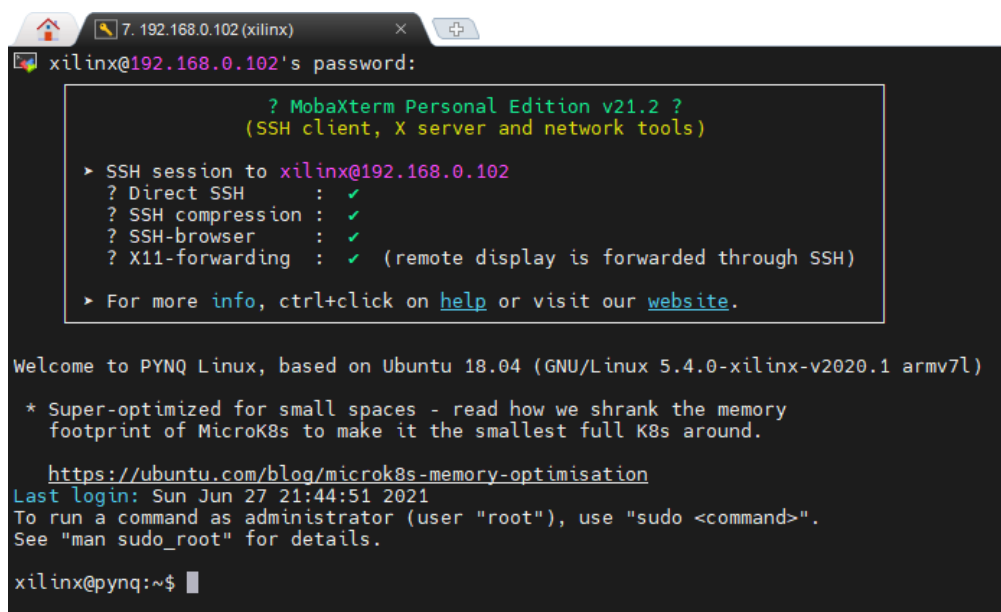
【施作環境為在使用者 PC/laptop/notebook (Windows Base) 。】

需要 SSH 軟體接收 console 訊息。

1. 開啟 MobaXterm，點選左上角 Session 後選取 SSH，Remote host 設為在所屬 LAN 被分配到的 IP address；username 為 xilinx；Port 為 22。



2. 輸入密碼（xilinx）後即可開始使用。



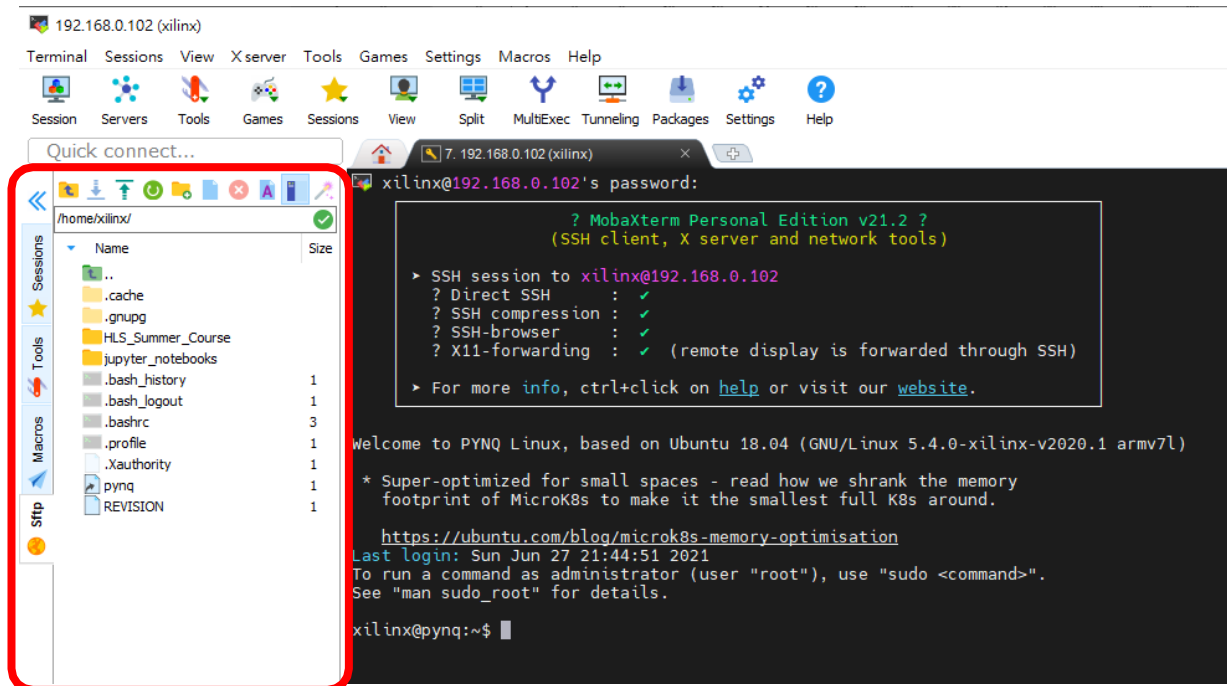
### 1.3.3. Data Transfer

PYNQ-Z2 有兩種傳輸檔案的方式，可依需求選擇使用：

#### 1.3.3.1. SFTP Connection

【施作環境為在使用者 PC/laptop/notebook (Windows Base) 。】

使用 MobaXterm 以 1.3.2.3 節 SSH 方式連接 Terminal，將自動啟用 SFTP 檔案傳輸，可在視窗左側 sftp 標籤頁內管理檔案。



#### 1.3.3.2. Samba

【施作環境為在使用者 PC/laptop/notebook (Windows Base) 。】

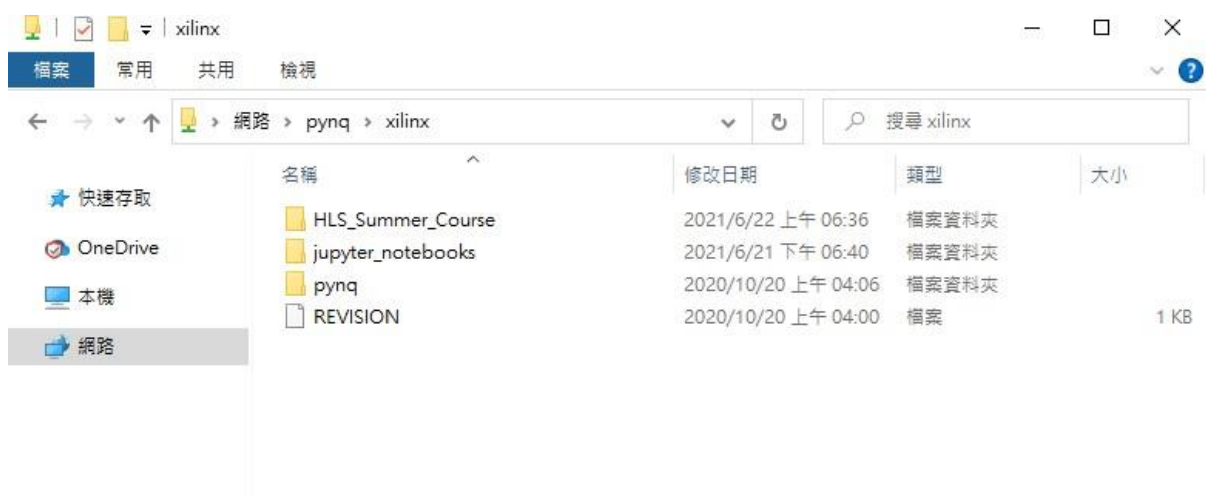
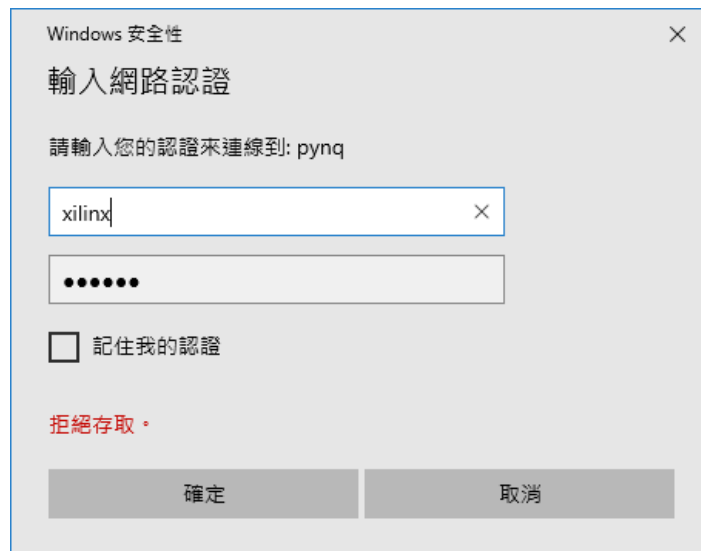
PYNQ-Z2 內建 Samba 檔案分享服務，可讓使用者像網路硬碟般存取 PYNQ home area 並傳輸檔案。

Note：此方法僅可存取\xilinx 資料夾，若需存取更上層資料夾仍須使用 SFTP。

1. 在 Windows 檔案總管點選網路，在上方路徑欄位輸入\\pynq\xilinx。



2. 輸入使用者名稱（xilinx）及密碼（xilinx），即可連線使用，進行檔案傳輸。



## 1.4. Network Setting

我們將 PYNQ-Z2 連上一個 LAN（Local Area Network）區域網路時，路由器（router）是在 WAN（Wide Area Network）下對外網路接口，另外路由器對內接口是對 LAN 區域網路分配並轉接區域網路內子節點的網路架構裝置，路由器由內建的 DHCP server 來分配區域網路內子節點的 IP address，所以只要在 PYNQ-Z2 的網卡設定好在子節點上接受 DHCP server 分配的 IP address 即可。

PYNQ-Z2 已有內建網路卡組態檔位於 `/etc/network/interfaces.d/eth0`，初始檔案內容如下：

---

```
auto eth0
iface eth0 inet dhcp

auto eth0:1
iface eth0:1 inet static
address 192.168.2.99
netmask 255.255.255.0
```

---

若遭遇因 MAC address 開機時隨機變換，導致 router 的 DHCP server 無法分配固定 IP，可藉由編輯網路卡組態來固定 MAC address，並重置 PYNQ-Z2 電源使組態生效。

```
$> sudo nano /etc/network/interfaces.d/eth0
```

---

```
auto eth0
iface eth0 inet dhcp
    hwaddress ether xx:xx:xx:xx:xx:xx    # if you want to force assign MAC address

auto eth0:1
iface eth0:1 inet static
address 192.168.2.99
netmask 255.255.255.0
```

---

## 2. Implement Flow

### 2.1. Vitis HLS/IP Design

【施作環境為在使用者 PC/laptop/notebook (Windows Base) 。】

啟動 Vitis HLS 開發套件。



使用 PYNQ-Z2 board 時，需要先安裝好 board file 資訊！

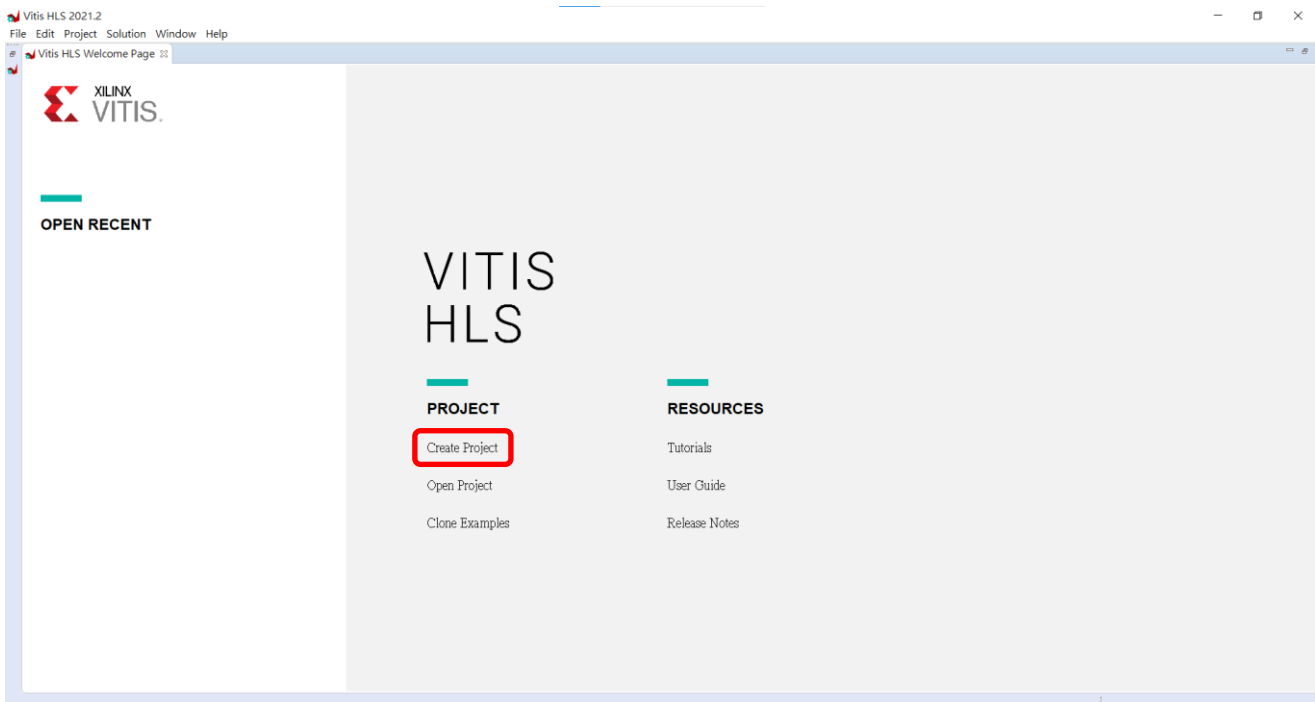
1. 下載 pynq-z2.zip 檔。連結網址：

<https://www.tulembedded.com/FPGA/ProductsPYNQ-Z2.html>

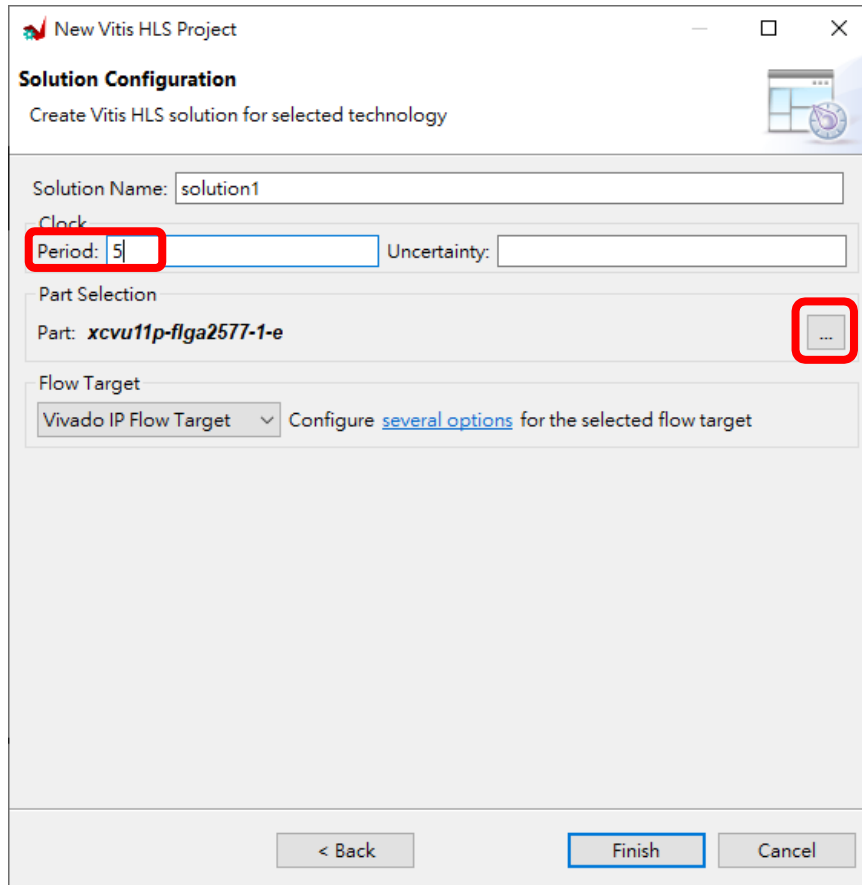
2. 解壓縮 pynq-z2.zip，並將目錄 pynq-z2 拷貝至  
C:\Xilinx\Vivado\2021.2\data\boards\board\_files 路徑下

#### 2.1.1. Create Source Project

開啟新的專案，設定好專案存放路徑：



若 PYNQ-Z2 的 FPGA 可運行於時脈 200MHz 以上，可以選擇週期為 5 ns，並選取 PYNQ-Z2 實作板 board file configuration。



**New Vitis HLS Project**

**Solution Configuration**  
Create Vitis HLS solution for selected technology

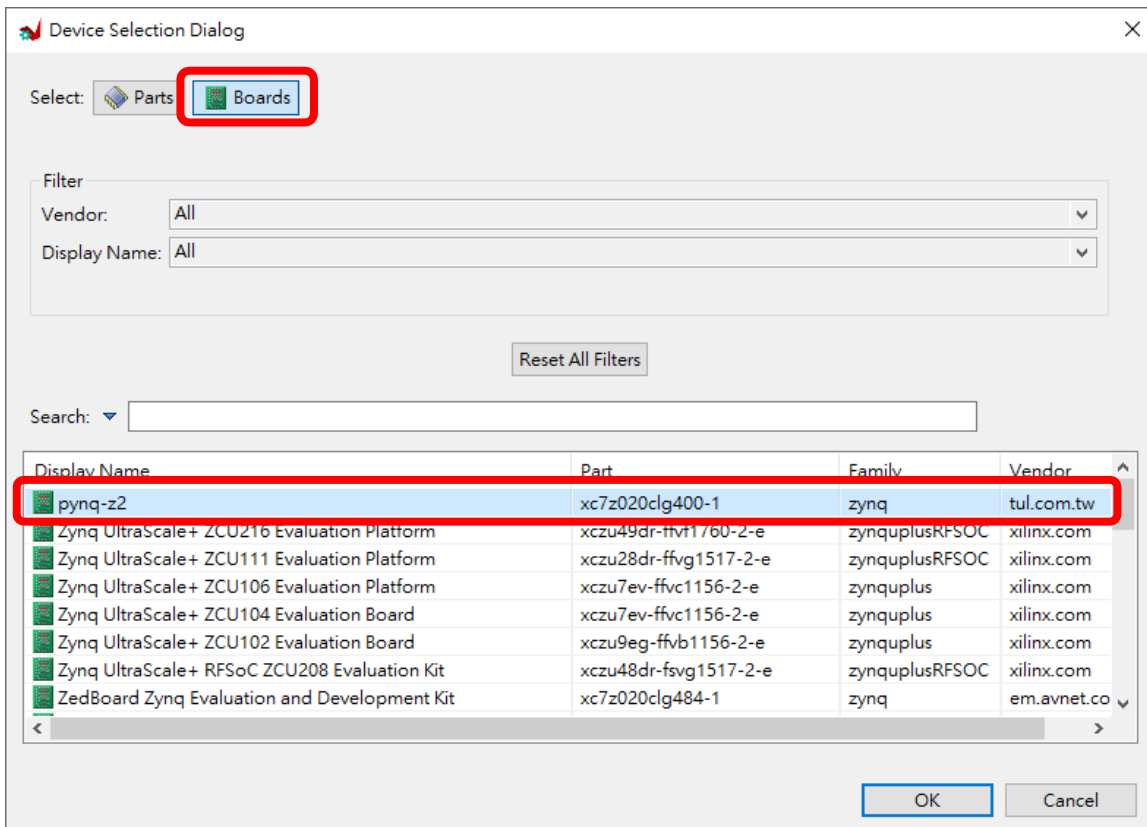
Solution Name:

Clock  
Period:  Uncertainty:

Part Selection  
Part: **xcvu11p-flga2577-1-e** ...

Flow Target  
Vivado IP Flow Target ▼ Configure [several options](#) for the selected flow target

< Back Finish Cancel



**Device Selection Dialog**

Select: Parts Boards

Filter  
Vendor: All ▼  
Display Name: All ▼

Reset All Filters

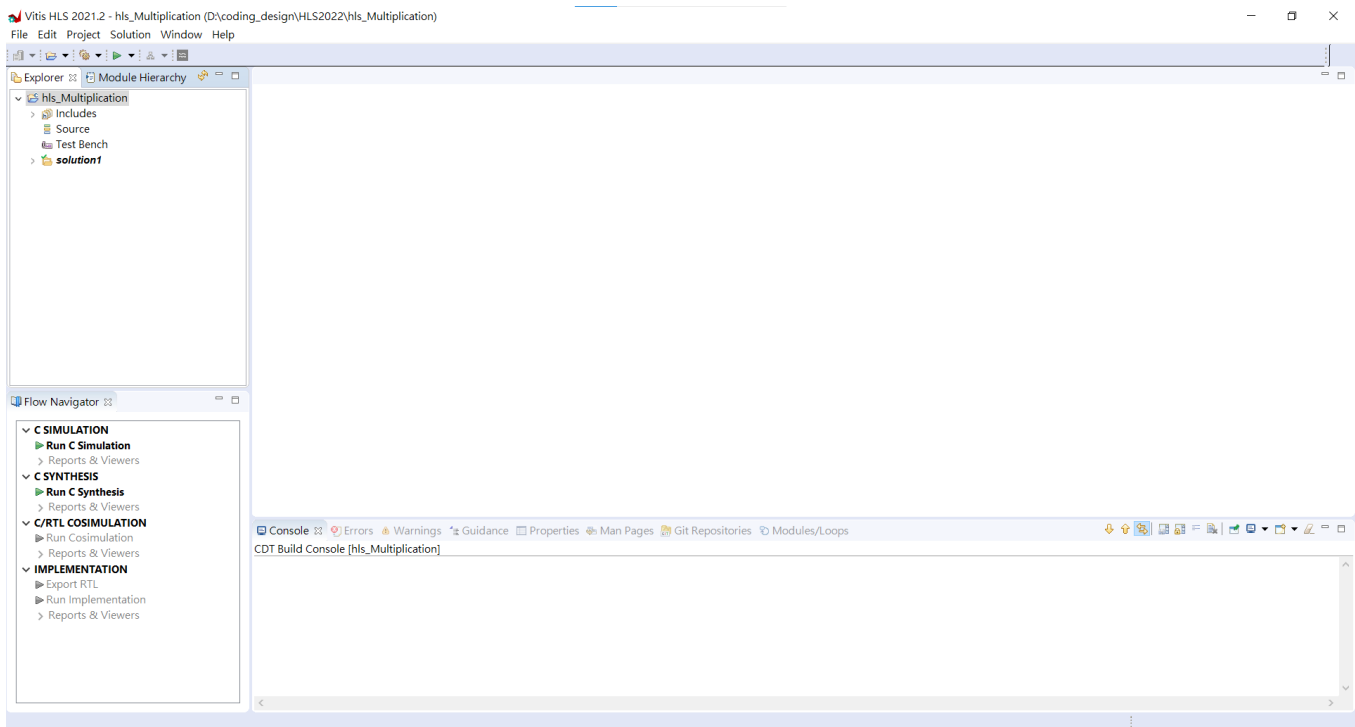
Search: ▼

Display Name	Part	Family	Vendor
pynq-z2	xc7z020clg400-1	zynq	tul.com.tw
Zynq UltraScale+ ZCU216 Evaluation Platform	xczu49dr-ftv11760-2-e	zynqplusRFSOC	xilinx.com
Zynq UltraScale+ ZCU111 Evaluation Platform	xczu28dr-ffvg1517-2-e	zynqplusRFSOC	xilinx.com
Zynq UltraScale+ ZCU106 Evaluation Platform	xczu7ev-ffvc1156-2-e	zynqplus	xilinx.com
Zynq UltraScale+ ZCU104 Evaluation Board	xczu7ev-ffvc1156-2-e	zynqplus	xilinx.com
Zynq UltraScale+ ZCU102 Evaluation Board	xczu9eg-ffvb1156-2-e	zynqplus	xilinx.com
Zynq UltraScale+ RFSOC ZCU208 Evaluation Kit	xczu48dr-fsvg1517-2-e	zynqplusRFSOC	xilinx.com
ZedBoard Zynq Evaluation and Development Kit	xc7z020clg484-1	zynq	em.avnet.co

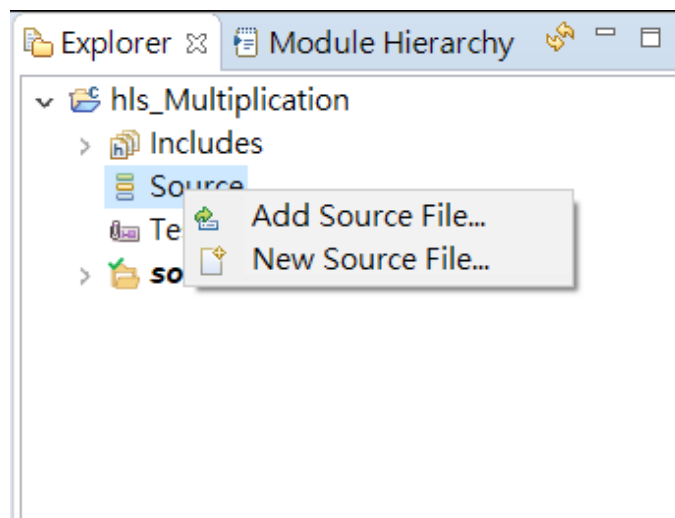
< >

OK Cancel

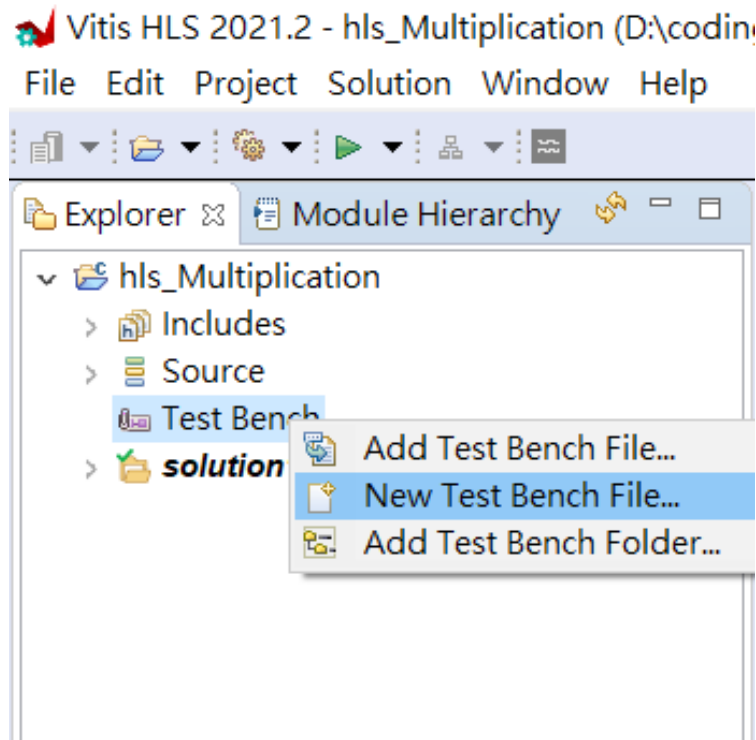
進入 Vitis HLS 專案 IDE 畫面後，加入 Source/Test Bench 的原始碼檔。



在專案 Source 按滑鼠右鍵加入新檔，在 Source 加入 Multiplication.cpp 及 Multiplication.h 檔。（#Include 部分注意大小寫問題。）

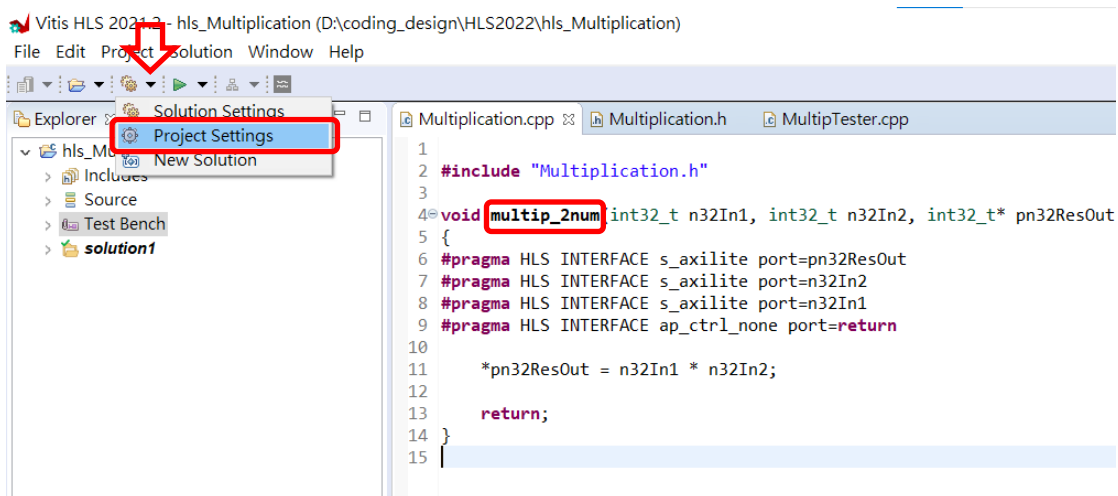


在專案 Test Bench 按滑鼠右鍵加入新檔，在 Test Bench 加入 MultipTester.cpp 檔。

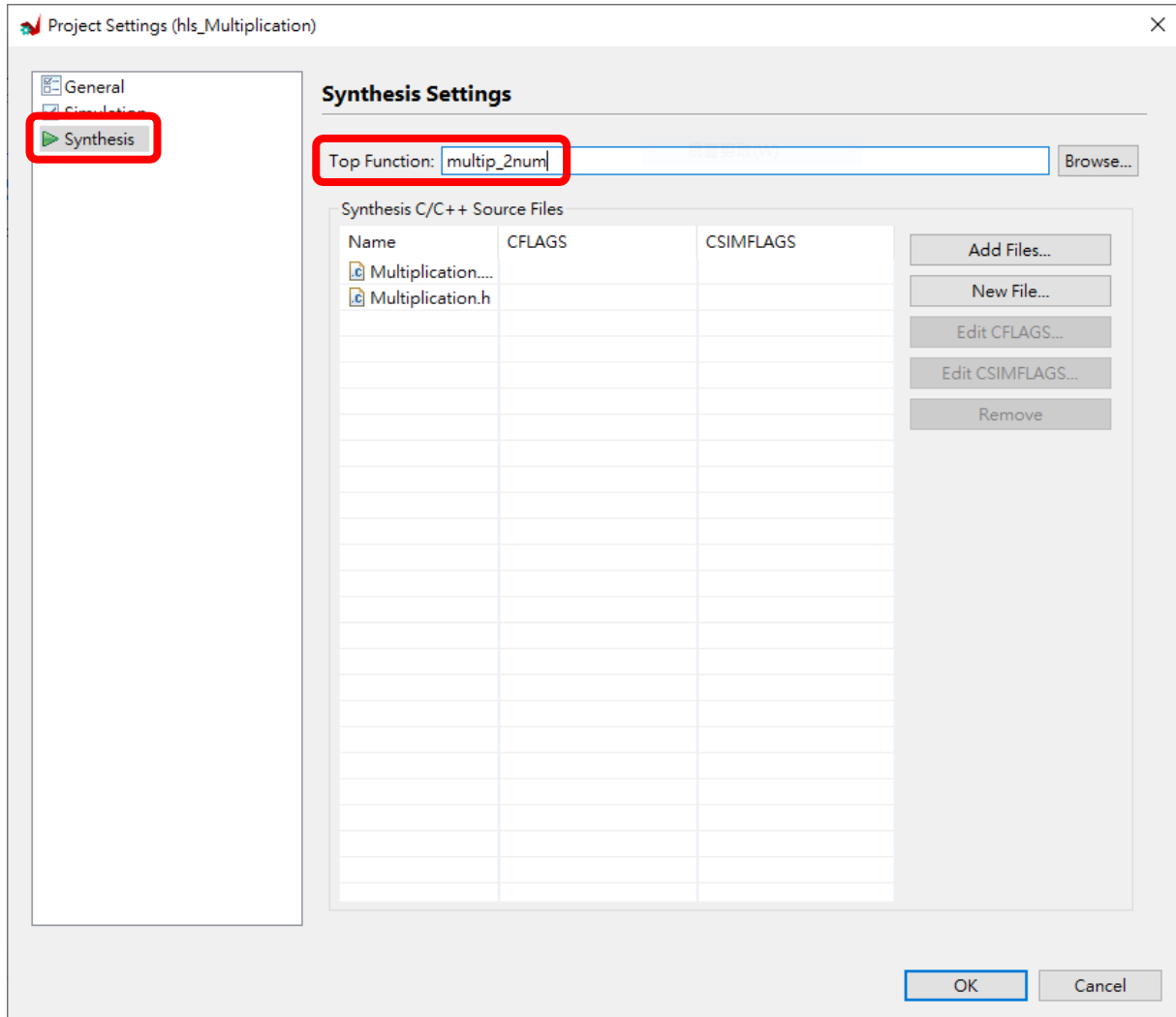


將#Lab. 1 提供的原始碼檔複製到 Multiplication.cpp / Multiplication.h / MultipTester.cpp。

專案必須指明 IP 匯出的 Top Function，在工具列設定圖加入 Top Function 名稱。



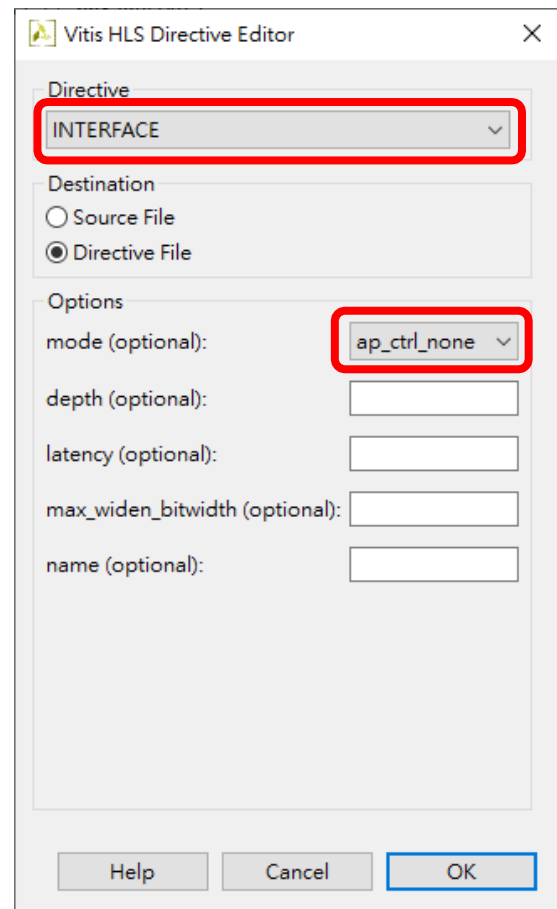
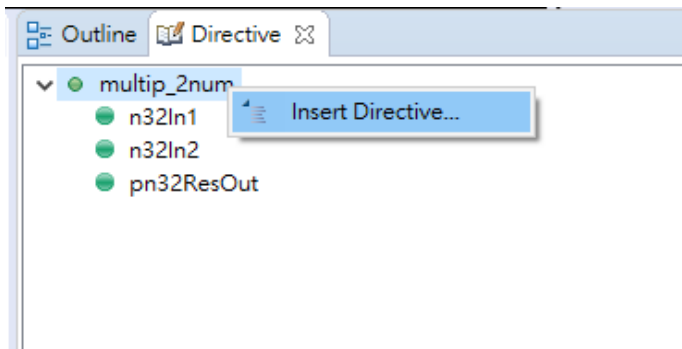
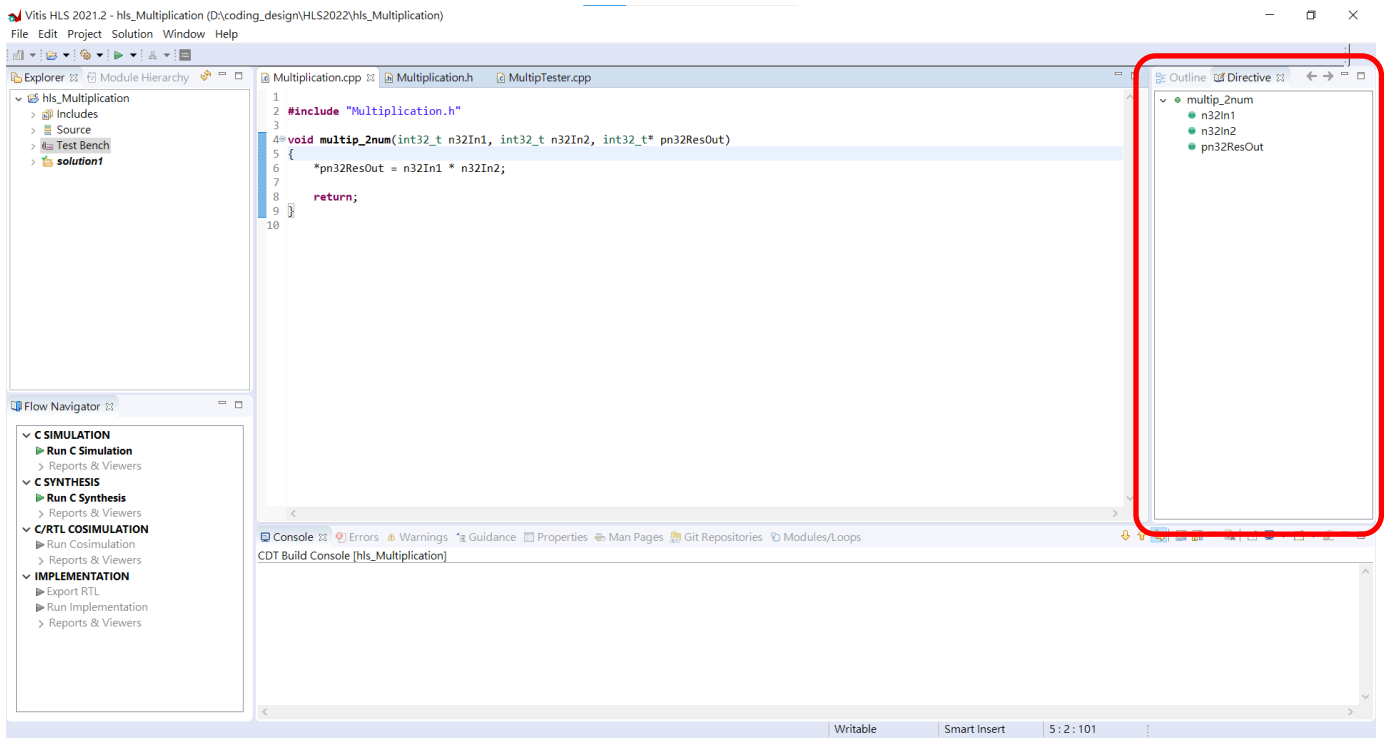




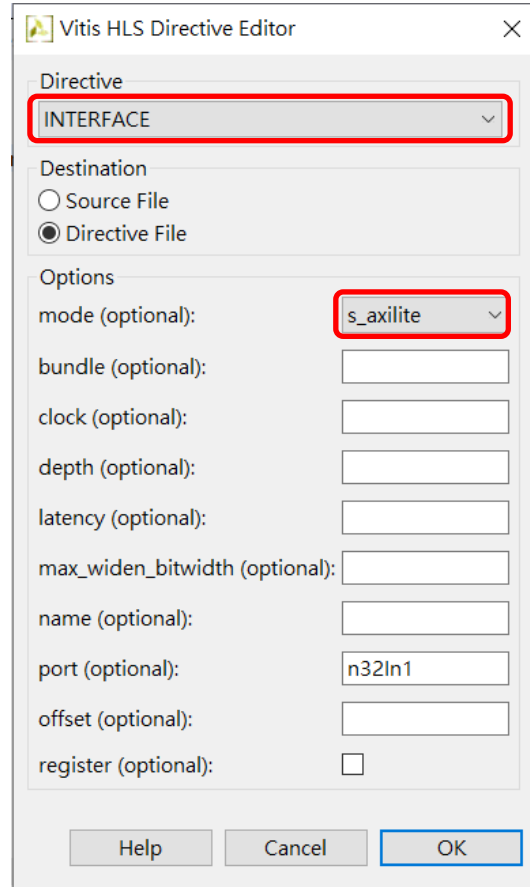
### 2.1.2. Directives Control

在開始 **Synthesis** 之前必須先加入 **directive** 描述。加入 **directive** 有兩種方式一是 **inline** 方式另一種是以 **directives.tcl** 來控制，目前以 **directives.tcl** 方式示範。本Lab. #1 原始碼已在 **source file** 中加入 **directive**。

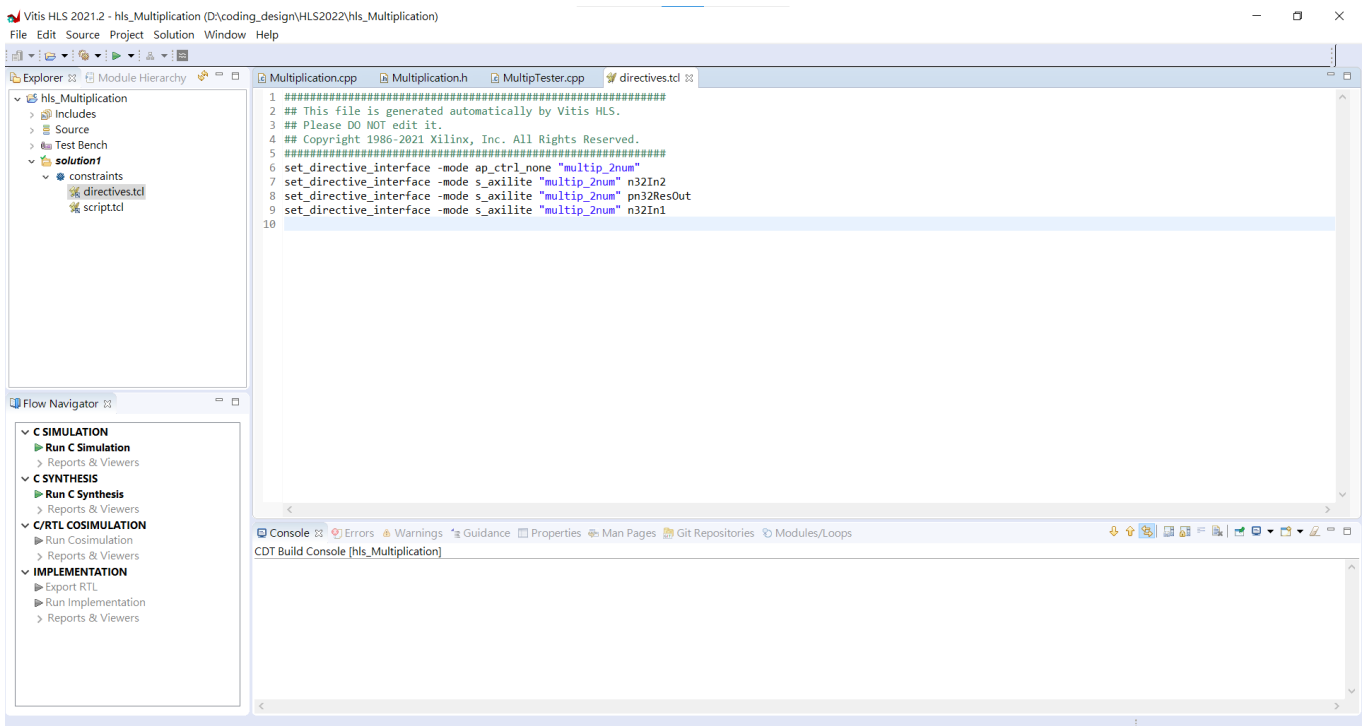
首先在 **directives** tab 頁面上編輯 **directives**。在 **top function** 按滑鼠右鍵插入 **interface** 的 **directive** 的組態：



相同地，為其他 top function 引入的引數插入 interface 的 directive 的組態：

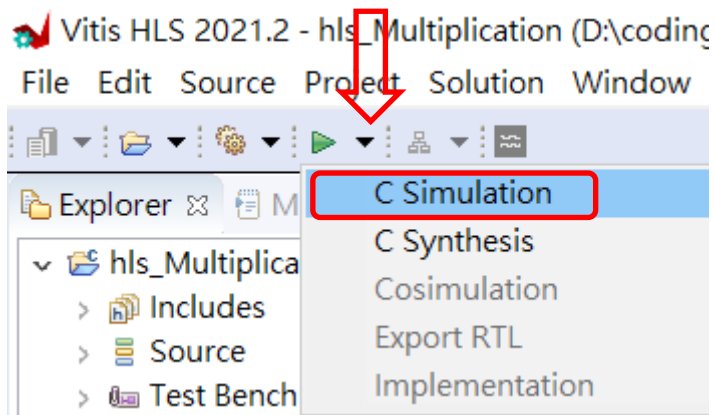
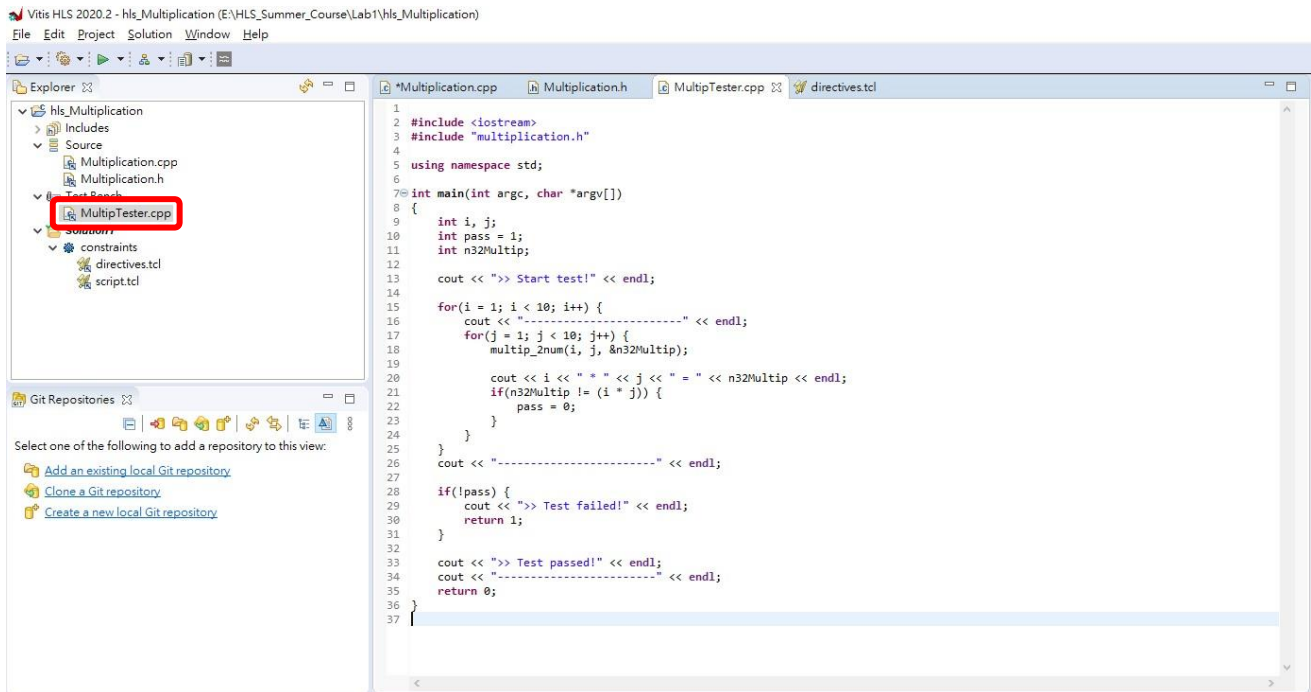


完成後可以檢視 directives.tcl 的輸出：



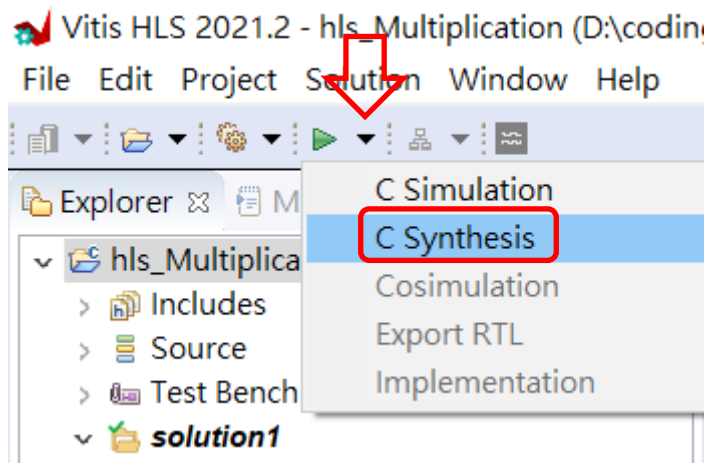
## 2.1.3. C Simulation

在 Lab. #1 裡提供 Test Bench 的 C Simulation 原始碼檔，已經有匯入專案。可執行 C Simulation 來驗證 IP 的執行結果。

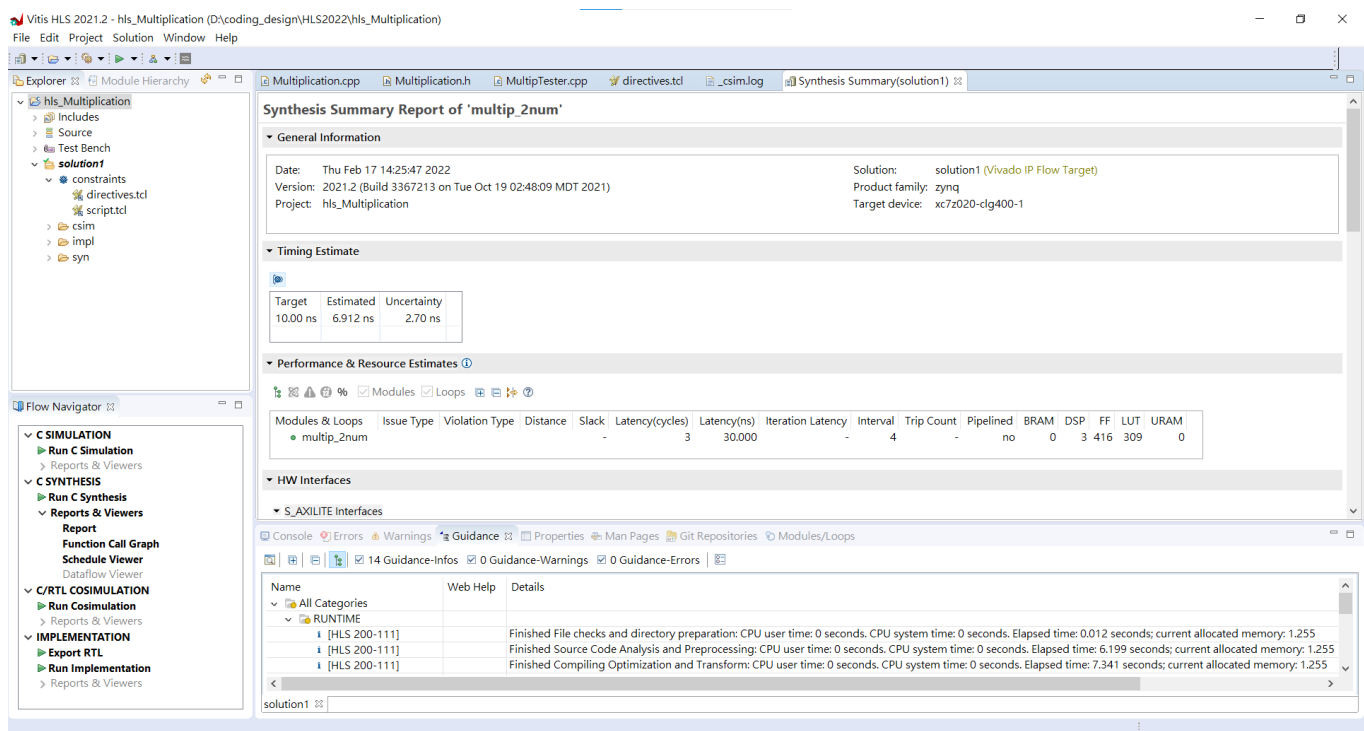


## 2.1.4. Synthesis

在 Vitis HLS 的功能列執行 C Synthesis：



完成 C Synthesis 會在主視窗回報 synthesis report：



The screenshot shows the Vitis HLS 2021.2 interface with the 'Synthesis Summary Report of multip\_2num' displayed. The report includes the following sections:

- General Information**
  - Date: Thu Feb 17 14:25:47 2022
  - Version: 2021.2 (Build 3367213 on Tue Oct 19 02:48:09 MDT 2021)
  - Project: hls\_Multiplication
  - Solution: solution1 (Vivado IP Flow Target)
  - Product family: zynq
  - Target device: xc7z020-clg400-1
- Timing Estimate**

Target	Estimated	Uncertainty
10.00 ns	6.912 ns	2.70 ns
- Performance & Resource Estimates**

Modules & Loops	Issue Type	Violation Type	Distance	Slack	Latency(cycles)	Latency(ns)	Iteration Latency	Interval	Trip Count	Pipelined	BRAM	DSP	FF	LUT	URAM	
multip_2num					3	30.000			4		no	0	3	416	309	0
- HW Interfaces**
  - S\_AXI\_LITE Interfaces
- Console**
  - 14 Guidance-Infos
  - 0 Guidance-Warnings
  - 0 Guidance-Errors
- Details**
  - [HLS 200-111] Finished File checks and directory preparation: CPU user time: 0 seconds. CPU system time: 0 seconds. Elapsed time: 0.012 seconds; current allocated memory: 1.255
  - [HLS 200-111] Finished Source Code Analysis and Preprocessing: CPU user time: 0 seconds. CPU system time: 0 seconds. Elapsed time: 6.199 seconds; current allocated memory: 1.255
  - [HLS 200-111] Finished Compiling Optimization and Transform: CPU user time: 0 seconds. CPU system time: 0 seconds. Elapsed time: 7.341 seconds; current allocated memory: 1.255

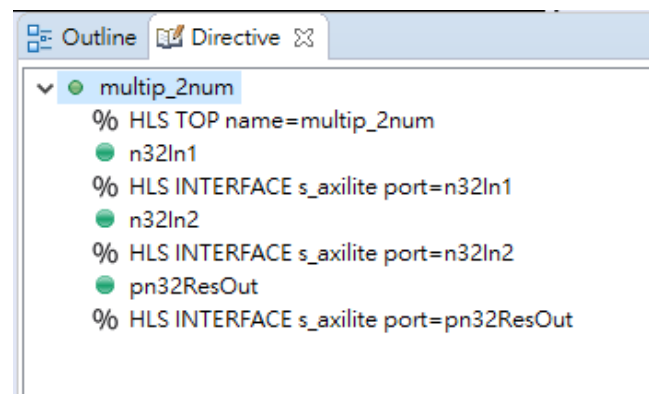
## 2.1.5. Cosimulation

當進行 Cosimulation 時 directive 的組態需改成如下圖，將 ap\_ctrl\_none 拿掉，不然會出現錯誤無法完成 Cosimulation。在執行 Cosimulation 必須重新 Synthesis。

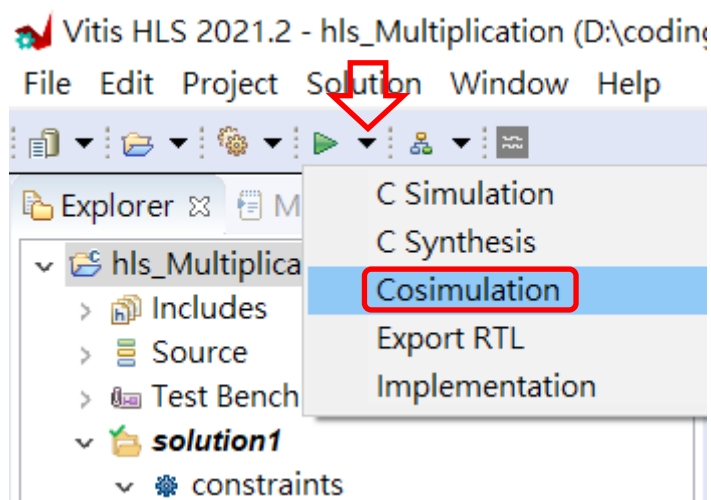
參考網址：

[https://www.xilinx.com/support/documentation/sw\\_manuals/xilinx2021\\_1/ug871-vivado-high-level-synthesis-tutorial.pdf](https://www.xilinx.com/support/documentation/sw_manuals/xilinx2021_1/ug871-vivado-high-level-synthesis-tutorial.pdf)

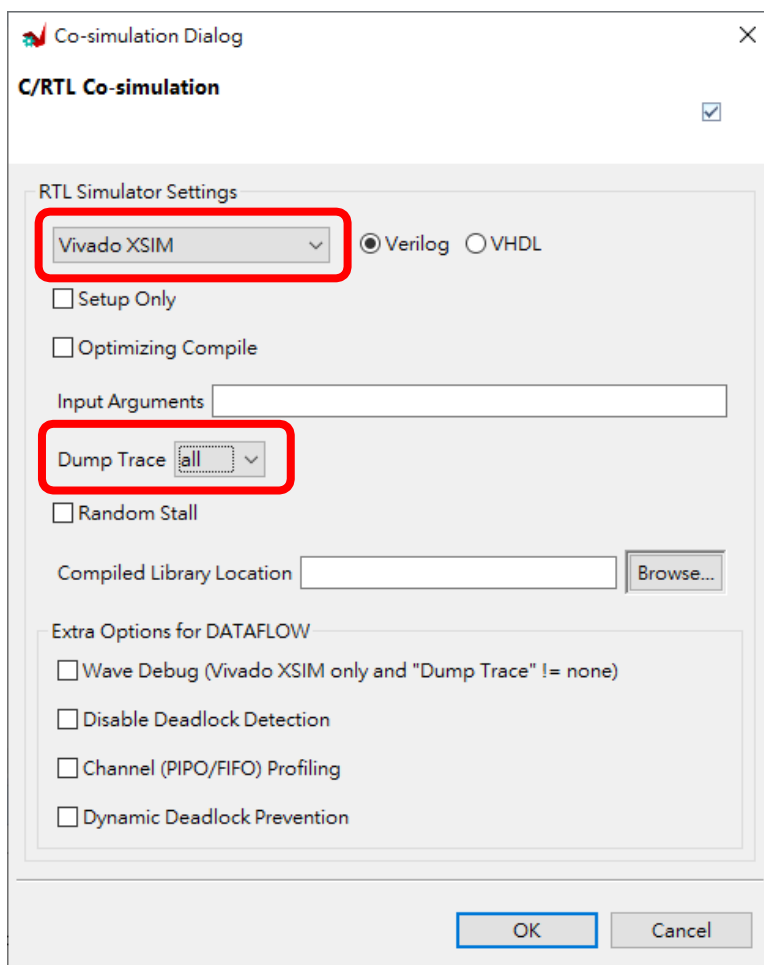
<https://forums.xilinx.com/t5/High-Level-Synthesis-HLS/Using-ap-memory-with-ap-control-none/td-p/681187>



按下 Cosimulation 鍵來驗證設計。



按下 Cosimulation 鍵後會彈出對話視窗



在輸出視窗檢視。要檢視波形可執行 Open Wave Veiwer。

Multiplication.cpp Multiplication.h MultipTester.cpp directives.tcl \_csim.log Synthesis Summary(solution1) Co-simulation Report(solution1)

### Cosimulation Report for 'multip\_2num'

▼ General Information

Date: Thu Feb 17 14:29:30 2022  
 Version: 2021.2 (Build 3367213 on Tue Oct 19 02:48:09 MDT 2021)  
 Project: hls\_Multiplication  
 Status: Pass

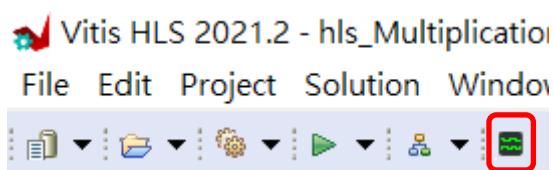
Solution: solution1 (Vivado IP Flow Target)  
 Product family: zynq  
 Target device: xc7z020-clg400-1

▼ Cosim Options

Tool: Vivado XSIM  
 Dump Trace: all  
 RTL: Verilog

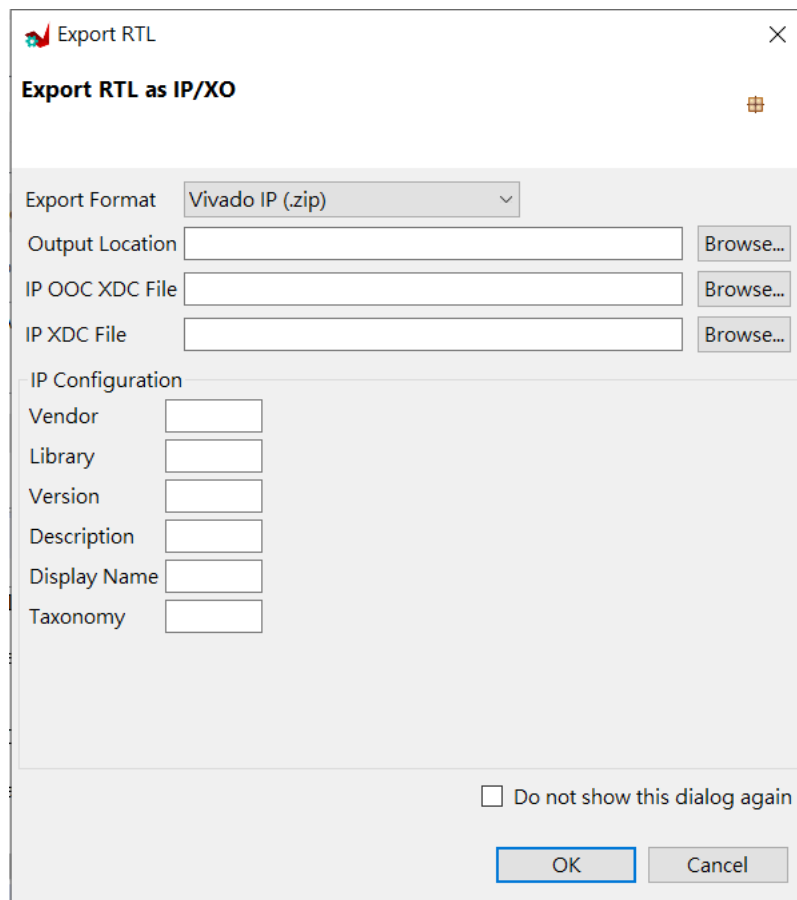
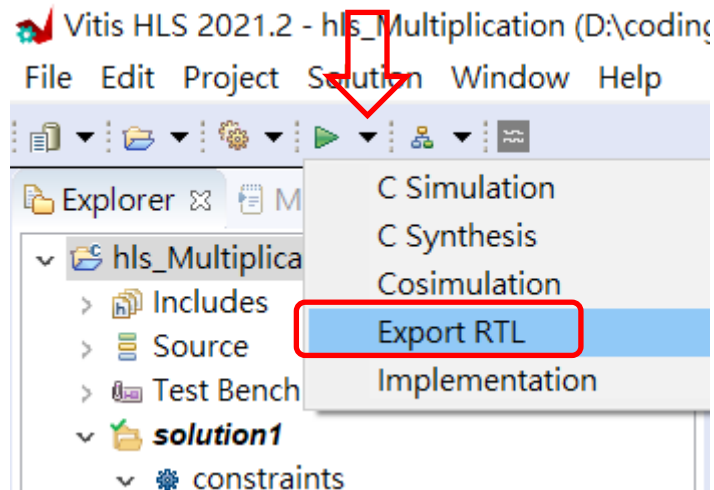
▼ Performance Estimates

Modules & Loops	Avg II	Max II	Min II	Avg Latency	Max Latency	Min Latency
• multip_2num	24	28	24	3	3	3



## 2.1.6. Export IP

完成 IP 的設計後，先從 Vitis HLS 匯出 IP，之後在 Vivado Design Suite 的 Vivado 還需要匯入由 Vitis HLS 匯出的 IP。現階段操作 Vitis HLS 匯出 RTL 功能，在匯出 RTL 前先還原原本的 directive，之後重新 Synthesis 才是做出的設計：





## 2.2. Vivado/Implement Flow

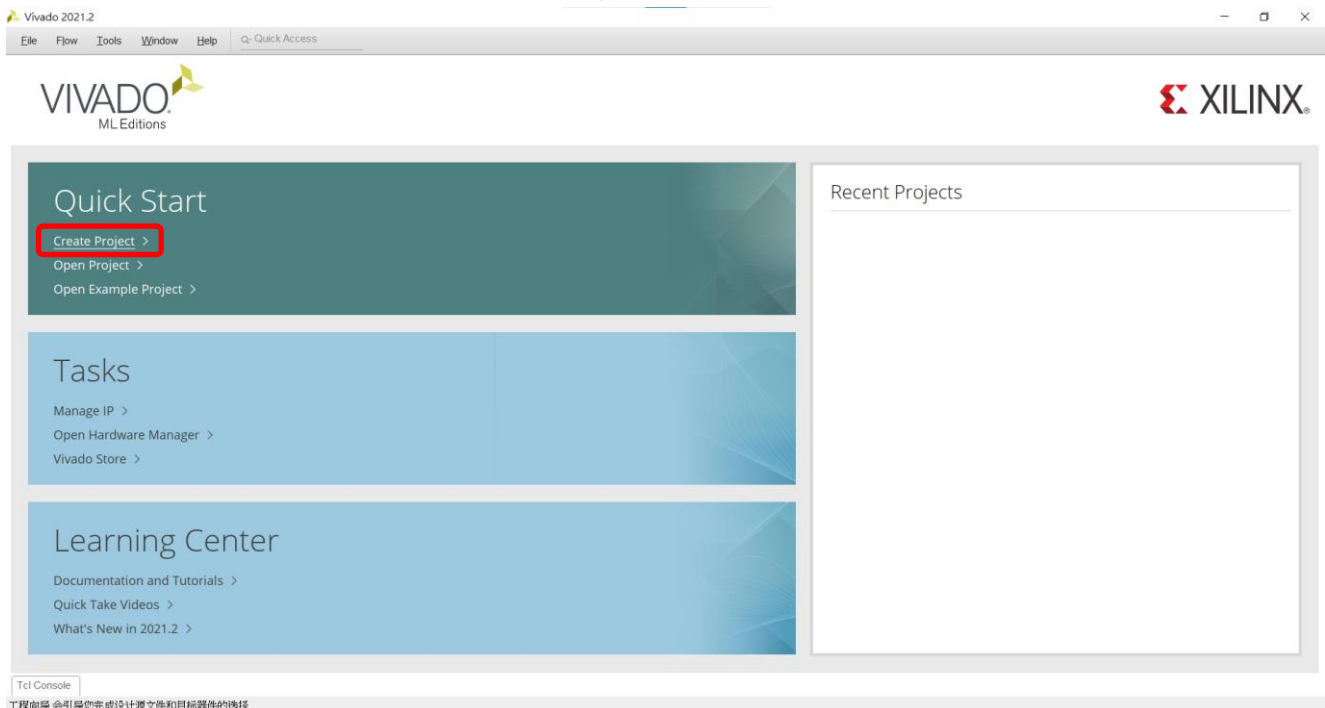
【施作環境為在使用者 PC/laptop/notebook (Windows Base) 。】

啟動 Vivado Design Suite 的 Vivado 開發套件。



### 2.2.1. Create Design Project

開啟新的專案，設定好專案存放路徑：



創建專案選取 PYNQ-Z2 並組建 PYNQ-Z2 組態。

### 2.2.2. Import IP

進入 Vivado 專案 IDE 畫面後，第一步驟是匯入由 Vitis HLS 所產生的 IP，在專案管理點擊 Settings 選項。



vvd\_Multip2Num - [D:/coding\_design/HLS2022/vvd\_Multip2Num/vvd\_Multip2Num.xpr] - Vivado 2021.2

File Edit Flow Tools Reports Window Layout View Help Q: Quick Access

Flow Navigator PROJECT MANAGER - vvd\_Multip2Num

**PROJECT MANAGER**

- Settings
- Add Sources
- Language Templates
- IP Catalog

**IP INTEGRATOR**

- Create Block Design
- Open Block Design
- Generate Block Design

**SIMULATION**

- Run Simulation

**RTL ANALYSIS**

- Open Elaborated Design

**SYNTHESIS**

- Run Synthesis
- Open Synthesized Design

**IMPLEMENTATION**

- Run Implementation
- Open Implemented Design

**Sources**

- Design Sources
- Constraints
- Simulation Sources
  - sim\_1
- Utility Sources

**Hierarchy** Libraries Compile Order

**Properties**

Select an object to see properties

**Project Summary**

**Overview** | Dashboard

**Settings** Edit

Project name: vvd\_Multip2Num  
Project location: D:/coding\_design/HLS2022/vvd\_Multip2Num  
Product family: Zynq-7000  
Project part: pynq-z2 (xc7z020c1g400-1)  
Top module name: Not defined  
Target language: Verilog  
Simulator language: Mixed

**Board Part**

Display name: pynq-z2  
Board part name: tul.com.tw.pynq-z2.part0.1.0

**Tcl Console** Messages Log Reports **Design Runs**

Name	Constraints	Status	WNS	TNS	WHS	THS	WBSS	TPWS	Total Power	Failed Routes	LUT	FF	BRAM	URAM	DSP	Start	Elapsed	Run Strategy
synth_1	constrs_1	Not started																Vivado Synthesis Defaults (Vivado Synthesis)
impl_1	constrs_1	Not started																Vivado Implementation Defaults (Vivado Imple)

Settings

Q:

**Project Settings**

- General
- Simulation
- Elaboration
- Synthesis
- Implementation
- Bitstream
- IP
  - Repository
  - Packager

**Tool Settings**

- Project
- IP Defaults
- Vivado Store
- Source File
- Display
- Help
- Text Editor
- 3rd Party Simulators
- Colors
- Selection Rules
- Shortcuts
- Strategies
- Window Behavior

**IP > Repository**

Add directories to the list of repositories. You may then add additional IP to a selected repository. If an IP is disabled then a tool-tip will alert you to the reason.

**IP Repositories**

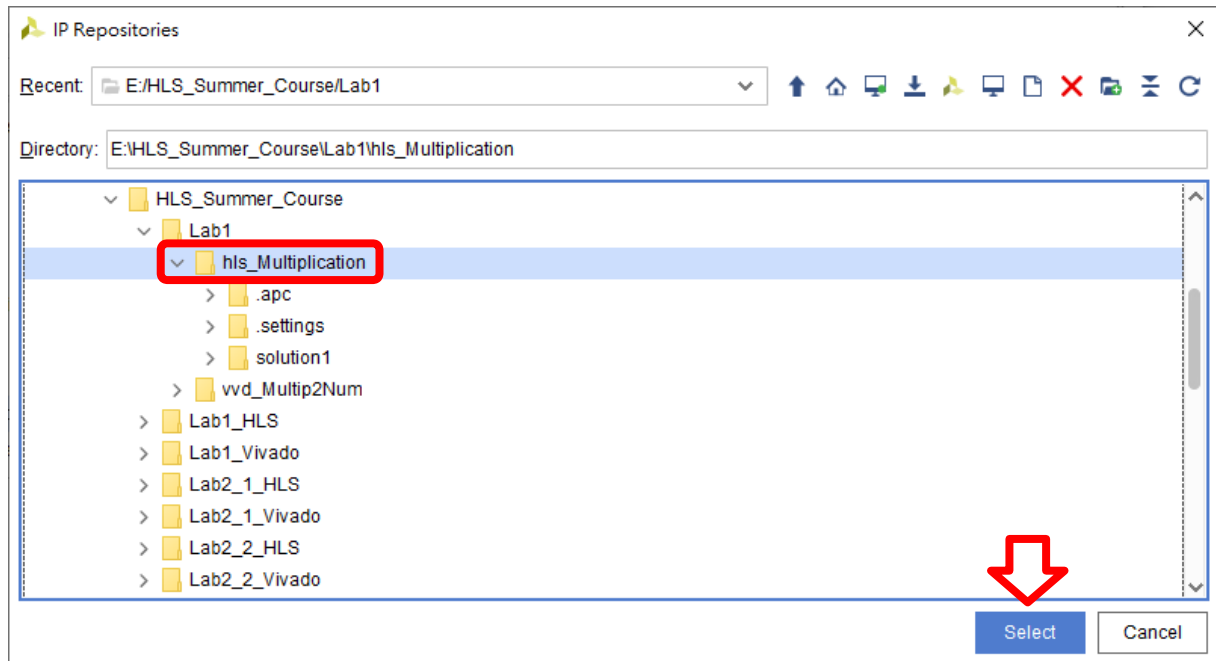
+ - ↑ ↓

No content

Refresh All

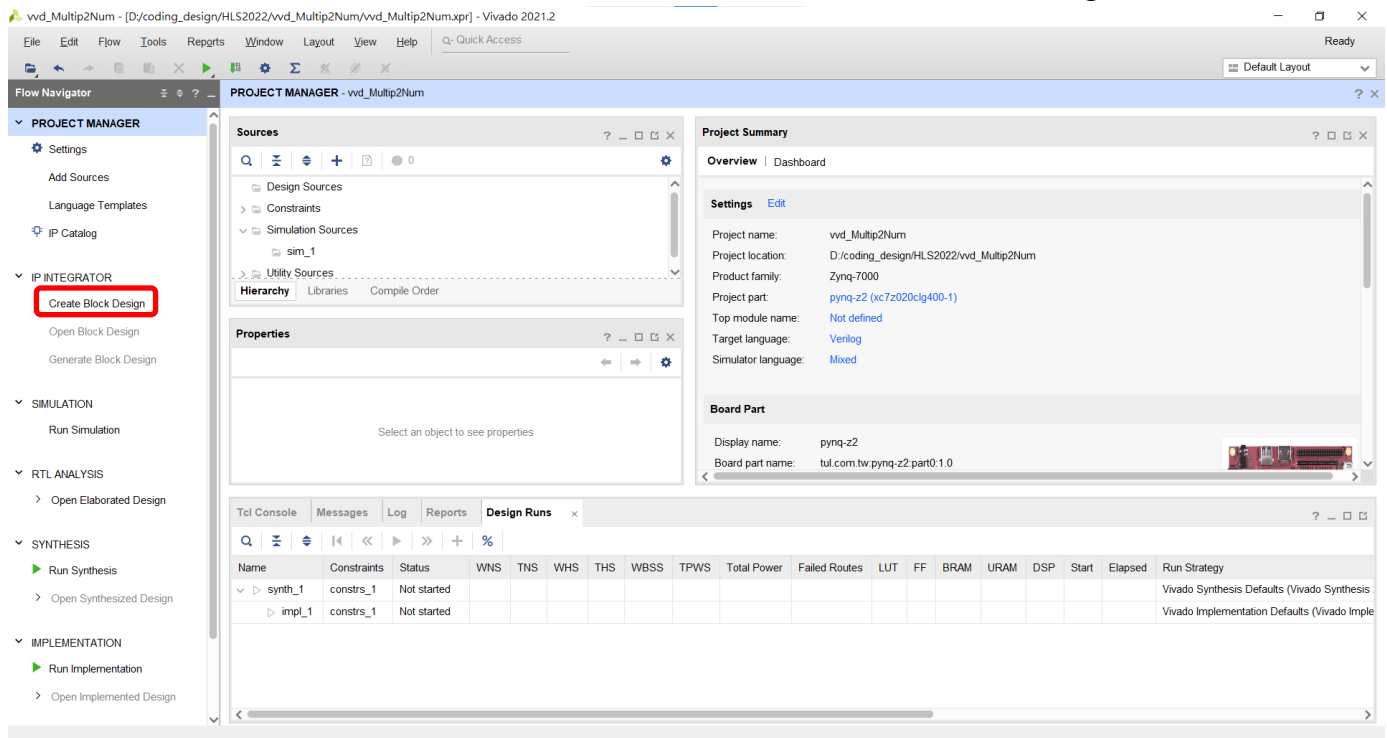
OK Cancel Apply Restore...

將 IP Repositories 指定到 Vitis HLS 專案目錄 hls\_Multiplication，下一步會匯入由 Vitis HLS 專案開發的 IP，且 IP 名稱會是專案裡的 Top Function 名稱：

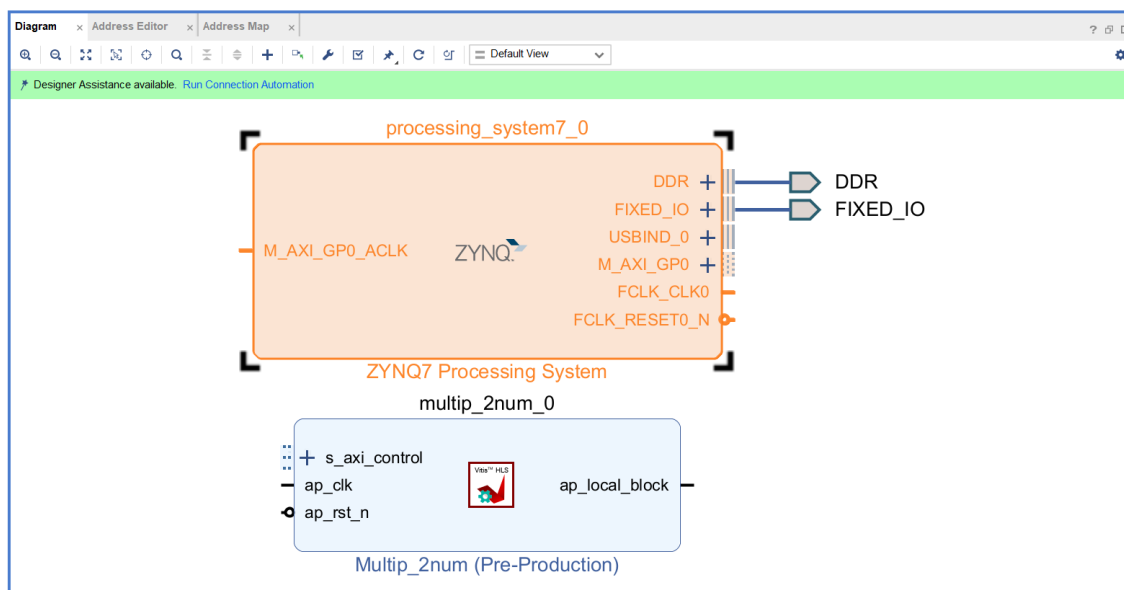


## 2.2.3. Block Design

回到 Vivado 專案 IDE 畫面，在專案管理點擊 Create Block Design 選項。



在 Diagram tab 視窗加入 components，並在 Run Block Automation 後用滑鼠左鍵雙擊 processing system block，將 PLL Fabric clock 設定為 200MHz。



Re-customize IP

### ZYNQ7 Processing System (5.5)

Documentation Presets IP Location Import XPS Settings

Page Navigator

- Zynq Block Design
- PS-PL Configuration
- Peripheral I/O Pins
- MIO Configuration
- Clock Configuration**
- DDR Configuration
- SMC Timing Calculation
- Interrupts

#### Clock Configuration

Basic Clocking Advanced Clocking

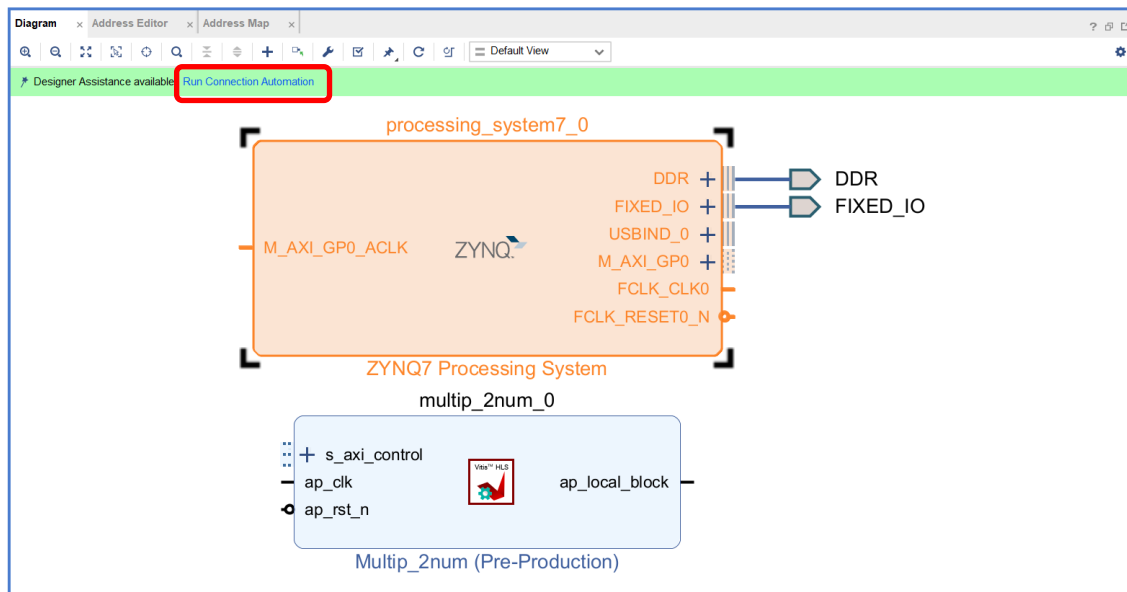
Input Frequency (MHz) 50 CPU Clock Ratio 6:2:1

Search: Q

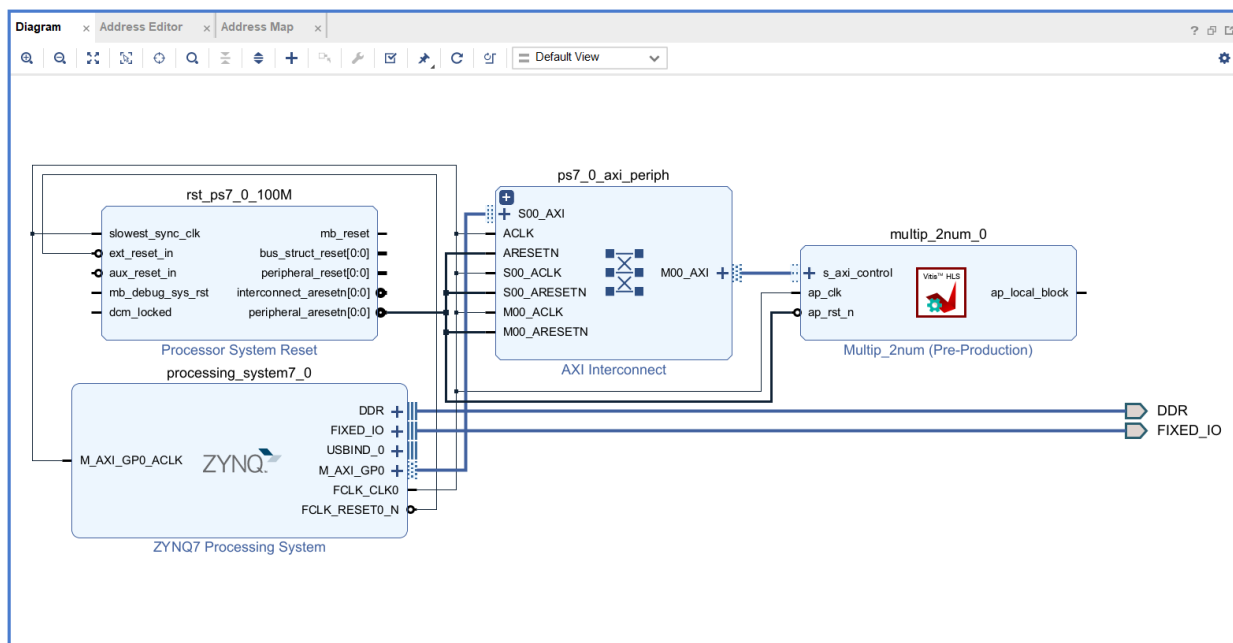
Component	Clock Source	Requested Frequ...	Actual Frequency...	Range(MHz)
Processor/Memory Clocks				
IO Peripheral Clocks				
PL Fabric Clocks				
<input checked="" type="checkbox"/> FCLK_CLK0	IO PLL	200	200.000000	0.100000 : 250.000000
<input type="checkbox"/> FCLK_CLK1	IO PLL	50	10.000000	0.100000 : 250.000000
<input type="checkbox"/> FCLK_CLK2	IO PLL	50	10.000000	0.100000 : 250.000000
<input type="checkbox"/> FCLK_CLK3	IO PLL	50	10.000000	0.100000 : 250.000000
System Debug Clocks				
Timers				

OK Cancel

因為 block design 只有 1 個 IP 的設計，連線完成可由系統自動完成，直接執行 Run Connection Automation 即可。

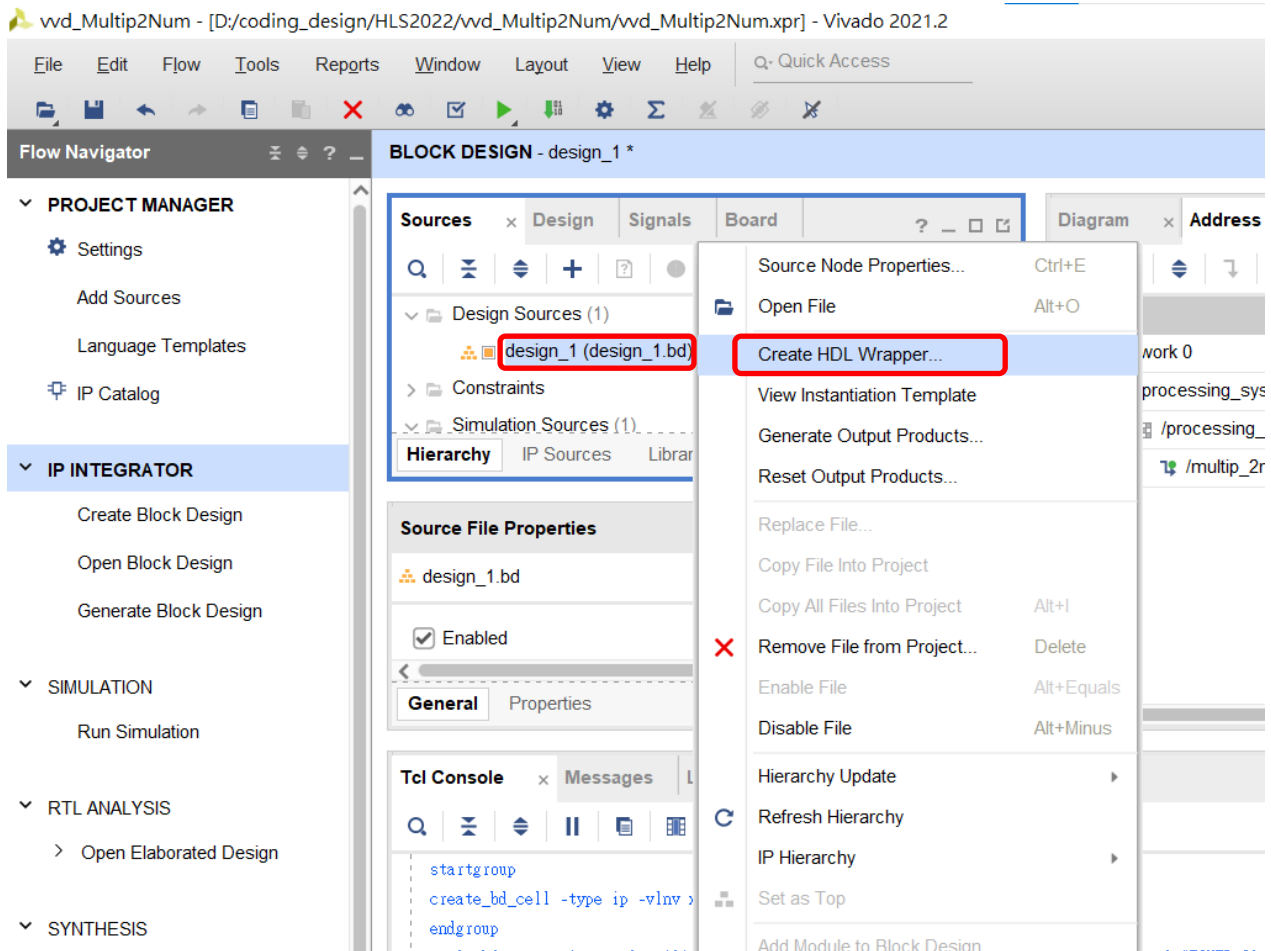


完成後整個完整的 diagram 圖，亦可切換到 Address Editor tab 檢視 memory map：



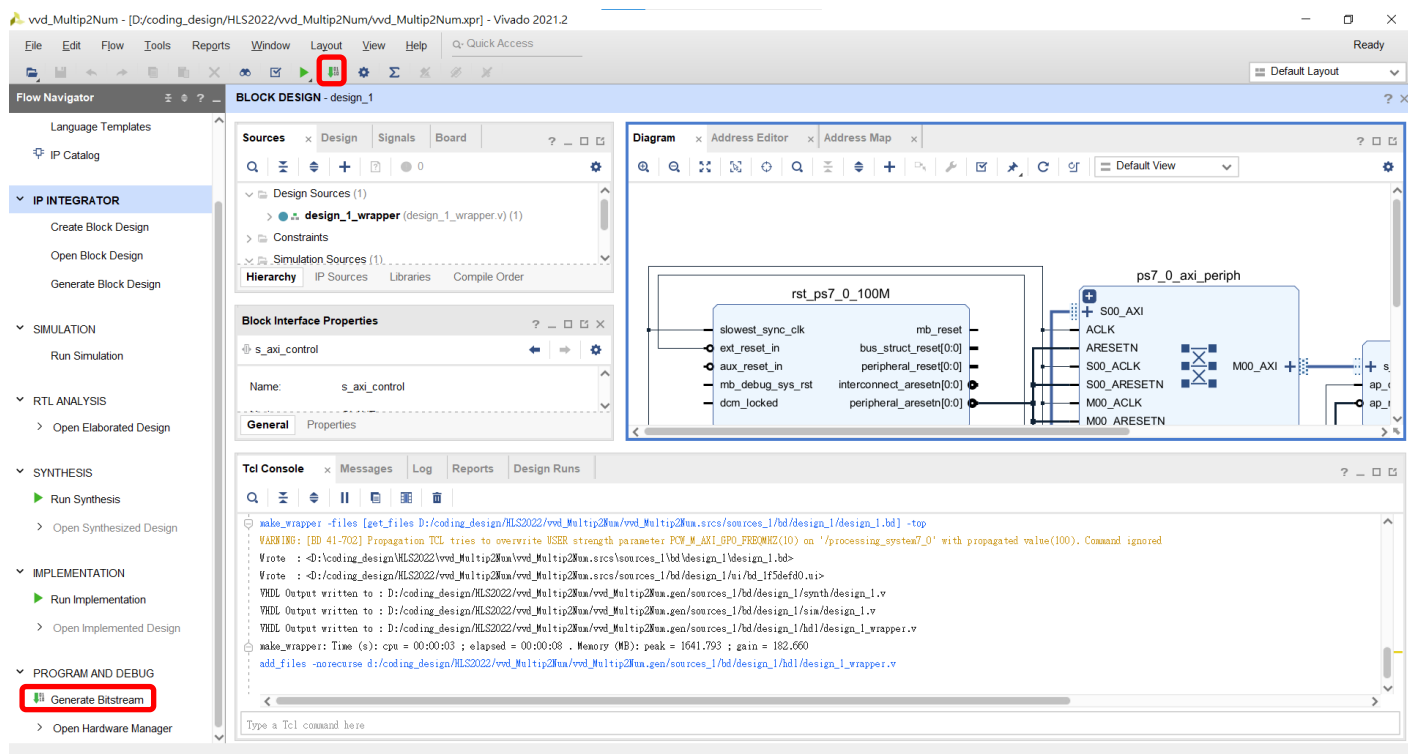
Name	Interface	Slave Segment	Master Base Address	Range	Master High Address
Network 0					
/processing_system7_0					
/processing_system7_0/Data (32 address bits : 0x40000000 [ 1G ])					
/multip_2num_0/s_axi_control	s_axi_control	Reg	0x4000_0000	64K	0x4000_FFFF

接下來進行 HDL Wrapper 動作，在 Design 子視窗 Source tab 頁面 Design Sources 的 design\_1.bd 點擊滑鼠右鍵進行 HDL Wrapper：



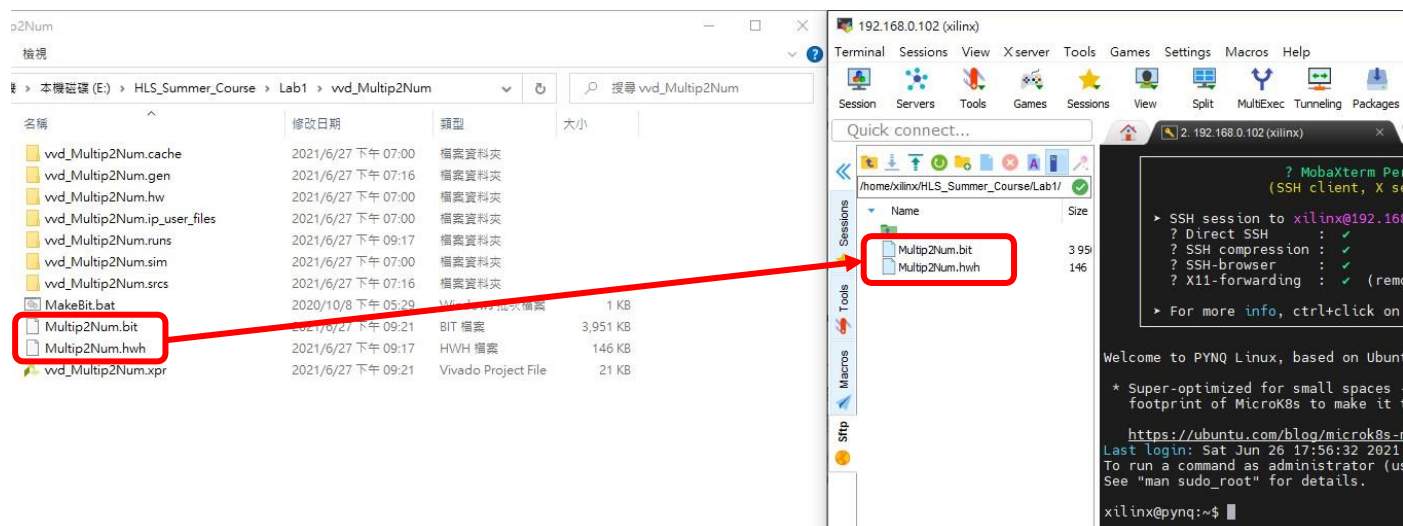
## 2.2.4. Synthesis/Placement/Routing/Generate Bit-stream

產生 FPGA 所需要的 bit-stream file，由於要產生 bit-stream file 需要經過 Synthesis/Placement/Routing 的步驟，但可以省去單步執行的流程以一鍵執行由 Vivado IDE 自動完成所有流程。在 Vivado 專案 IDE 畫面，在專案管理點擊 Generate Bitstream 選項或由工具列按下 Generate Bitstream 按鈕。



## 2.2.5. Bit-stream Transfer from Development Kit to Device

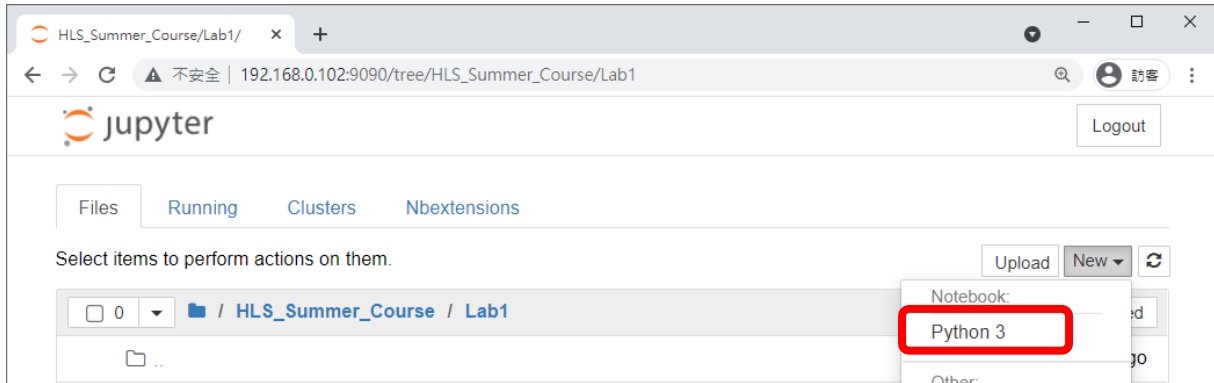
本實作已建立一個批次檔 MakeBit.bat 將 FPGA 運行時所需要 .bit/.hwh 拷貝到專案根目錄，此時只要將 .bit/.hwh 藉由 MobaXterm 或 Samba 傳送到 PYNQ-Z2 即可。提醒！開發者建立的專案名稱可能與 MakeBit.bat 內的專案名稱不相符，請自行修改 .bat 內專案路徑名稱。



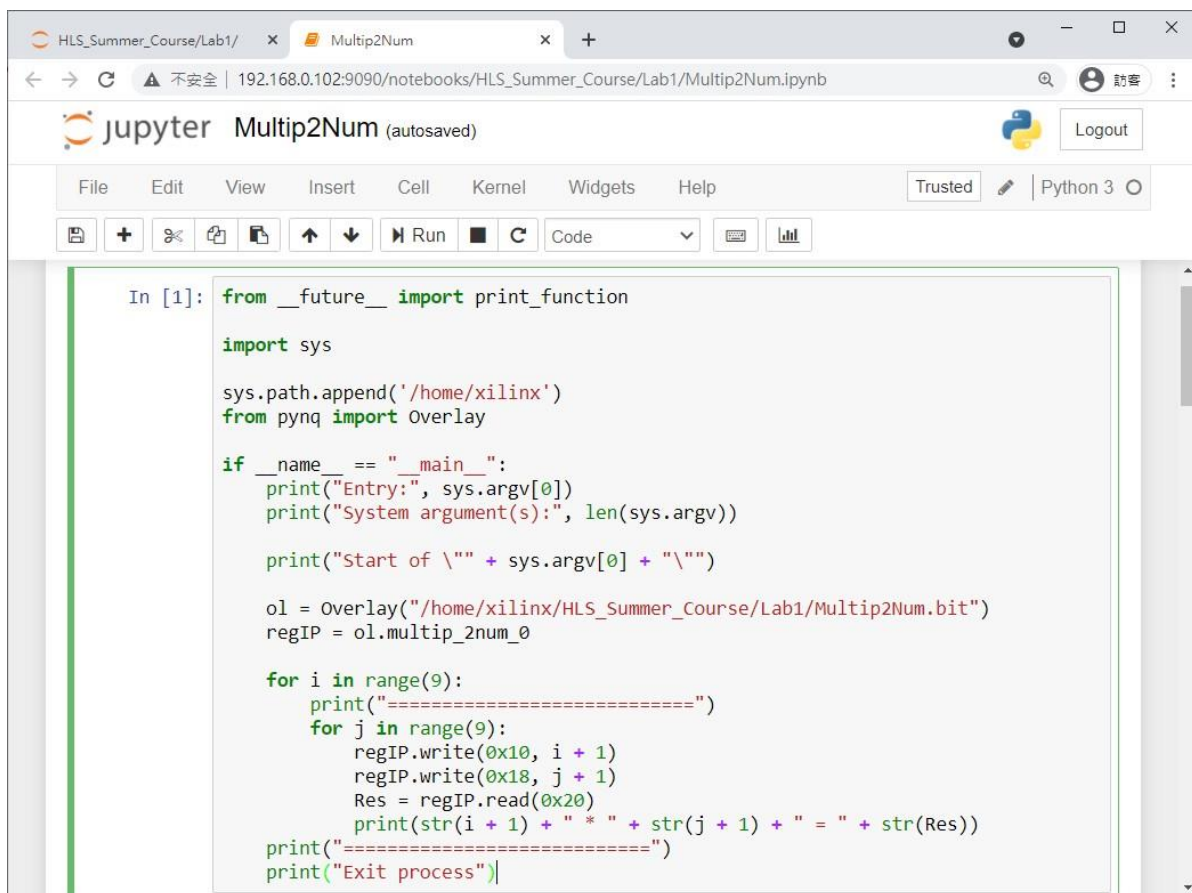
## 2.3. PYNQ/Host Program

### 2.3.1. Jupyter Notebook Browser Remote Editor

1. 啟動 Jupyter Notebook，並開啟一個新的 Python 3 檔案。



2. 將 Lab. #1 提供的 host program .py 內容拷貝到瀏覽器編輯視窗，並運行後檢視結果。





Note :

Kernel 的 register address offset 可在以下檔案中找到：

hls\_Multiplication\solution1\impl\misc\drivers\multip\_2num\_v1\_0\src\xmultip\_2num\_hw.h

## 2.3.2. Understanding PYNQ

PYNQ 是建立在 Xilinx platform 上的 API 模組，可運行 Python 程式碼的環境，PYNQ 提供 Python 語言模組可進行對 FPGA 組態建立及流程控制。參考連結：

<http://www.pynq.io/board.html>

PYNQ open source: <https://github.com/xilinx/pynq>