

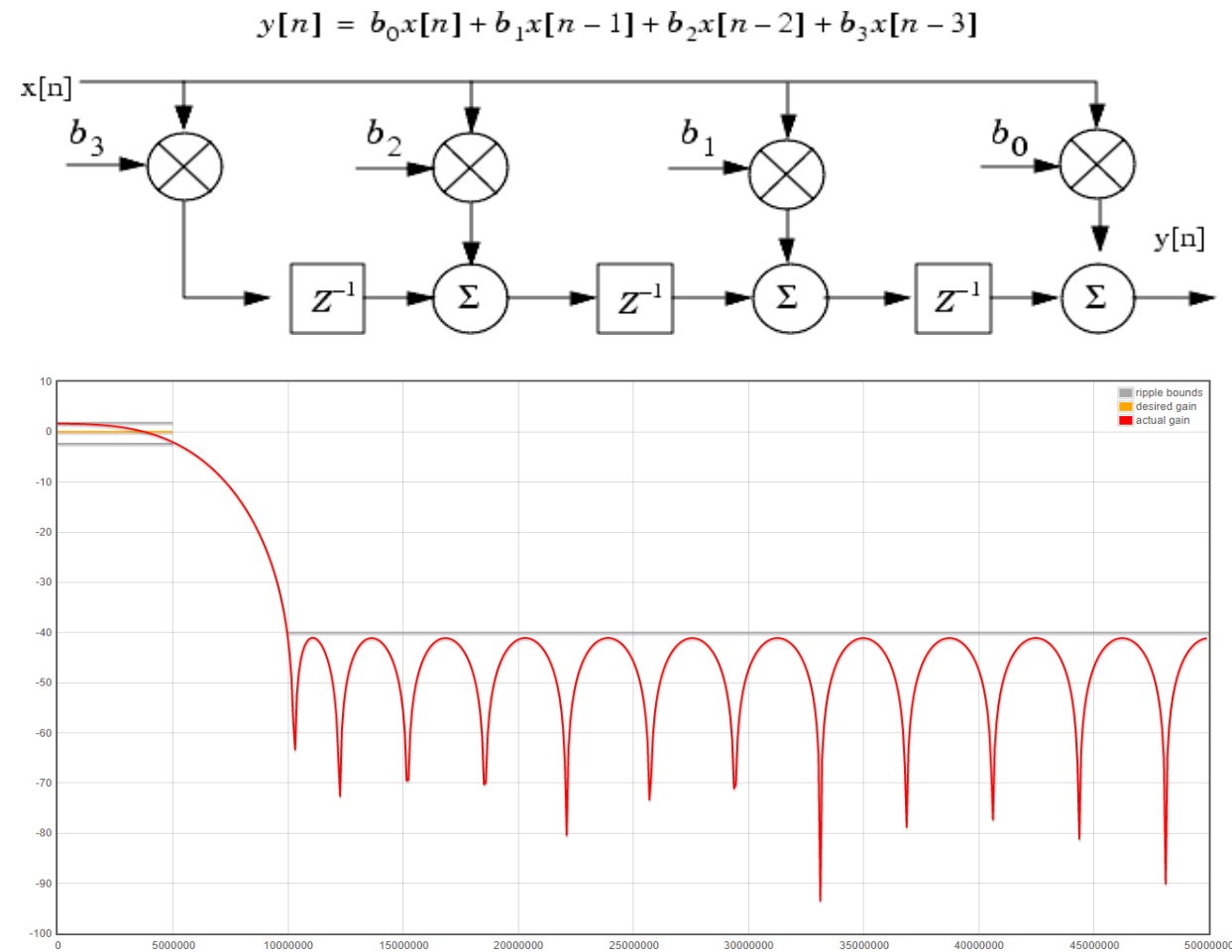
# Lab09 FPGA

## Low-pass FIR filter

---

# Low-pass FIR filter

- FIR filter Block Diagram
- 數位低通濾波器
- 相較於類比低通濾波器來說製作難易度低、成本低。
- 硬體上只需要暫存器、乘法器和加法器即可實現濾波效果。



# FPGA

- 利用軟體運算FIR filter

```
In [3]: from scipy.signal import lfilter
```

```
coeffs = [-255,-260,-312,-288,-144,153,616,1233,1963,2739,3474,4081,4481,4620,4481,4081,3474,2739,1963,1233,616,153,-144,-288,-31
```

```
import time
```

```
start_time = time.time()
```

```
sw_fir_output = lfilter(coeffs,70e3,samples)
```

```
stop_time = time.time()
```

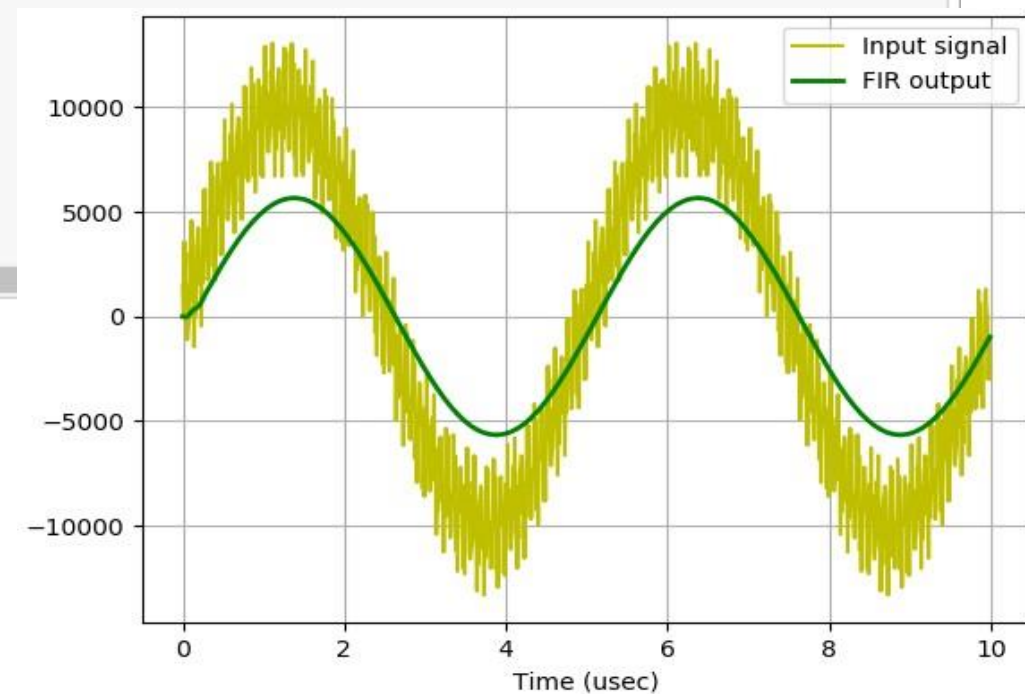
```
sw_exec_time = stop_time - start_time
```

```
print('Software FIR execution time: ',sw_exec_time)
```

```
# Plot the result to notebook
```

```
plot_to_notebook(t,samples,1000,out_signal=sw_fir_output)
```

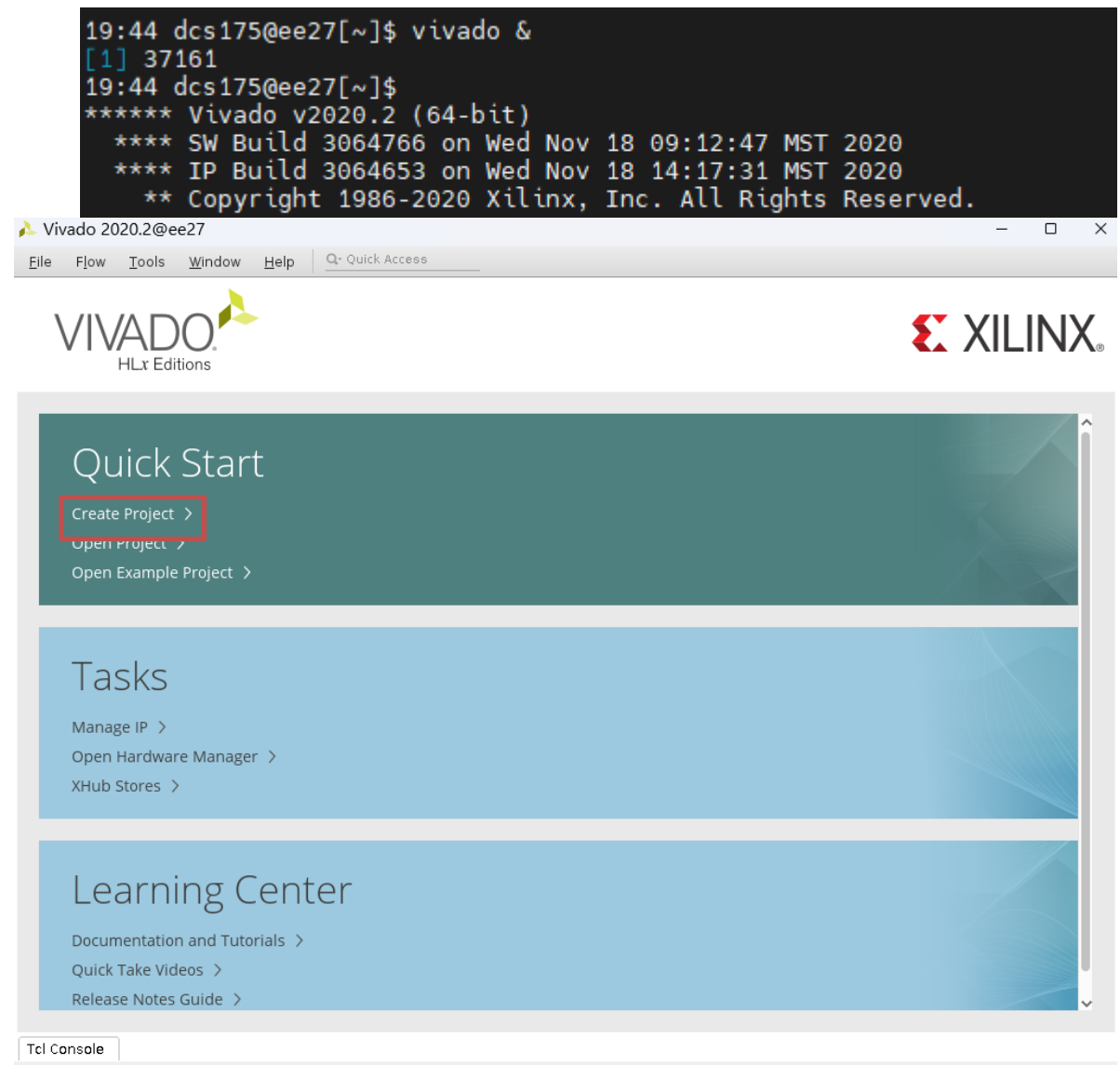
```
Software FIR execution time: 0.08370518684387207
```



# Block Design Flow

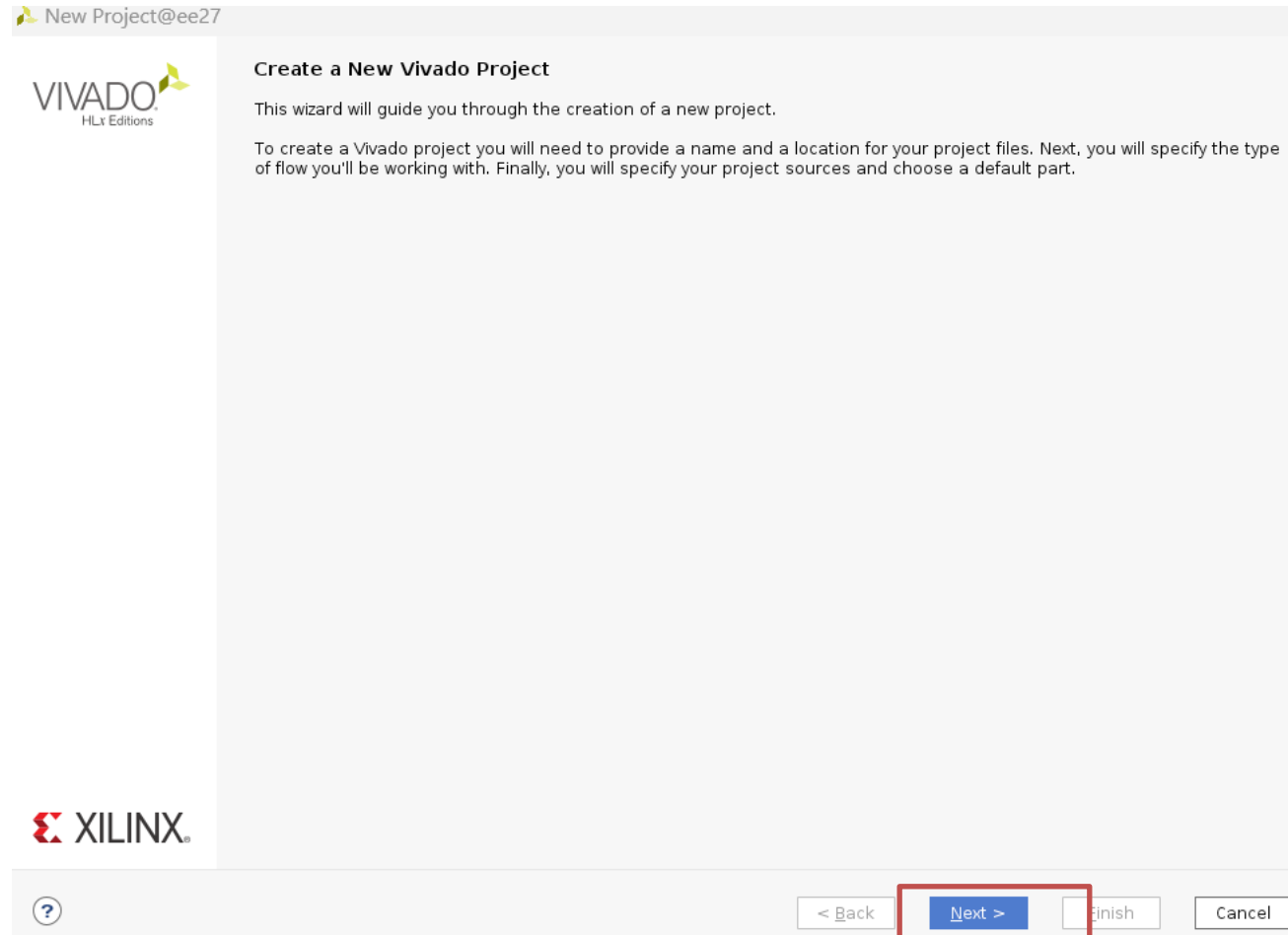
# 開啟vivado

- 在terminal輸入指令 vivado &
- 點選create project



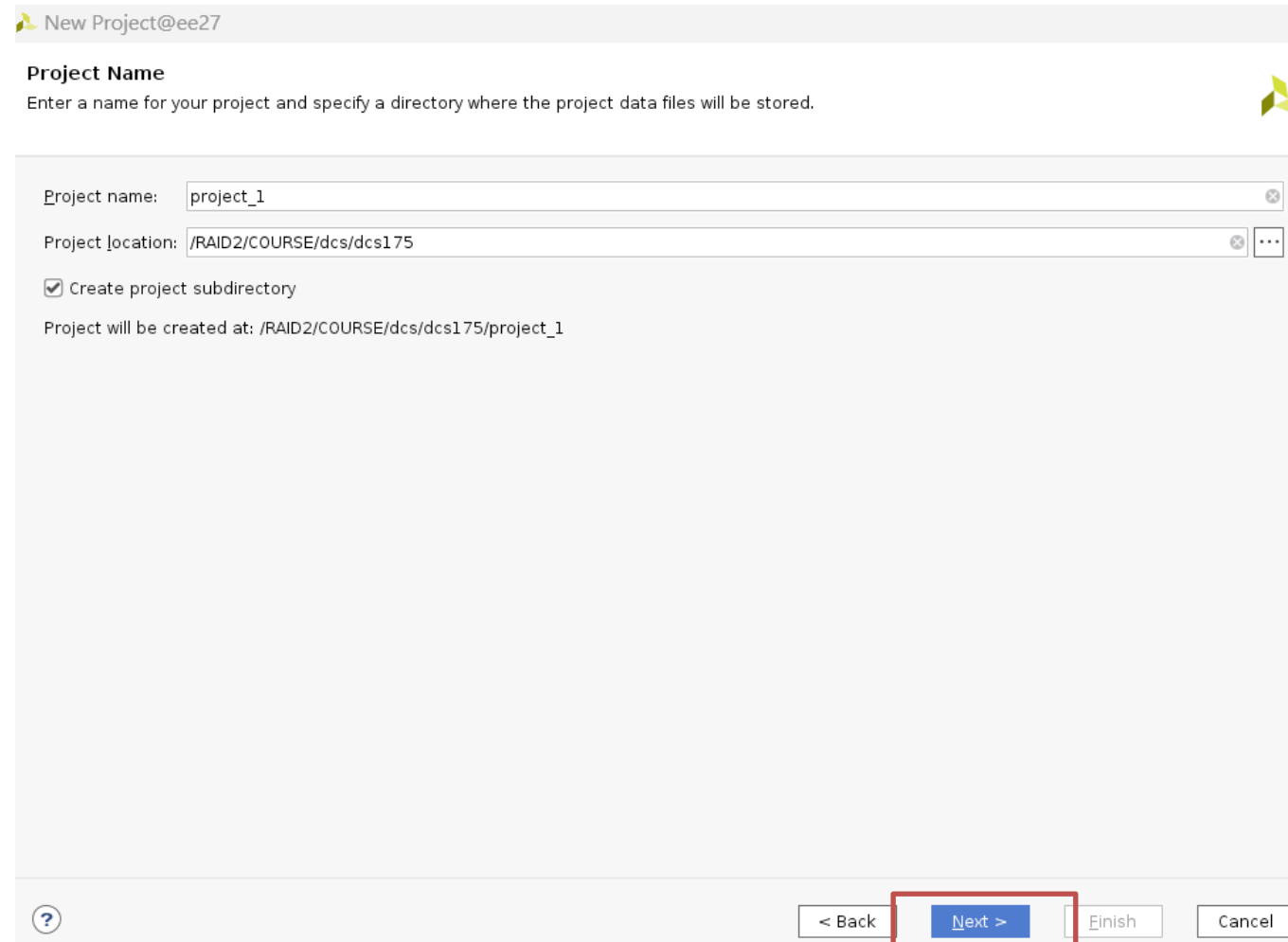
# 開啟vivado

- 點選next



# 開啟vivado

- 點選next



New Project@ee27

**Project Name**  
Enter a name for your project and specify a directory where the project data files will be stored.

Project name:

Project location:

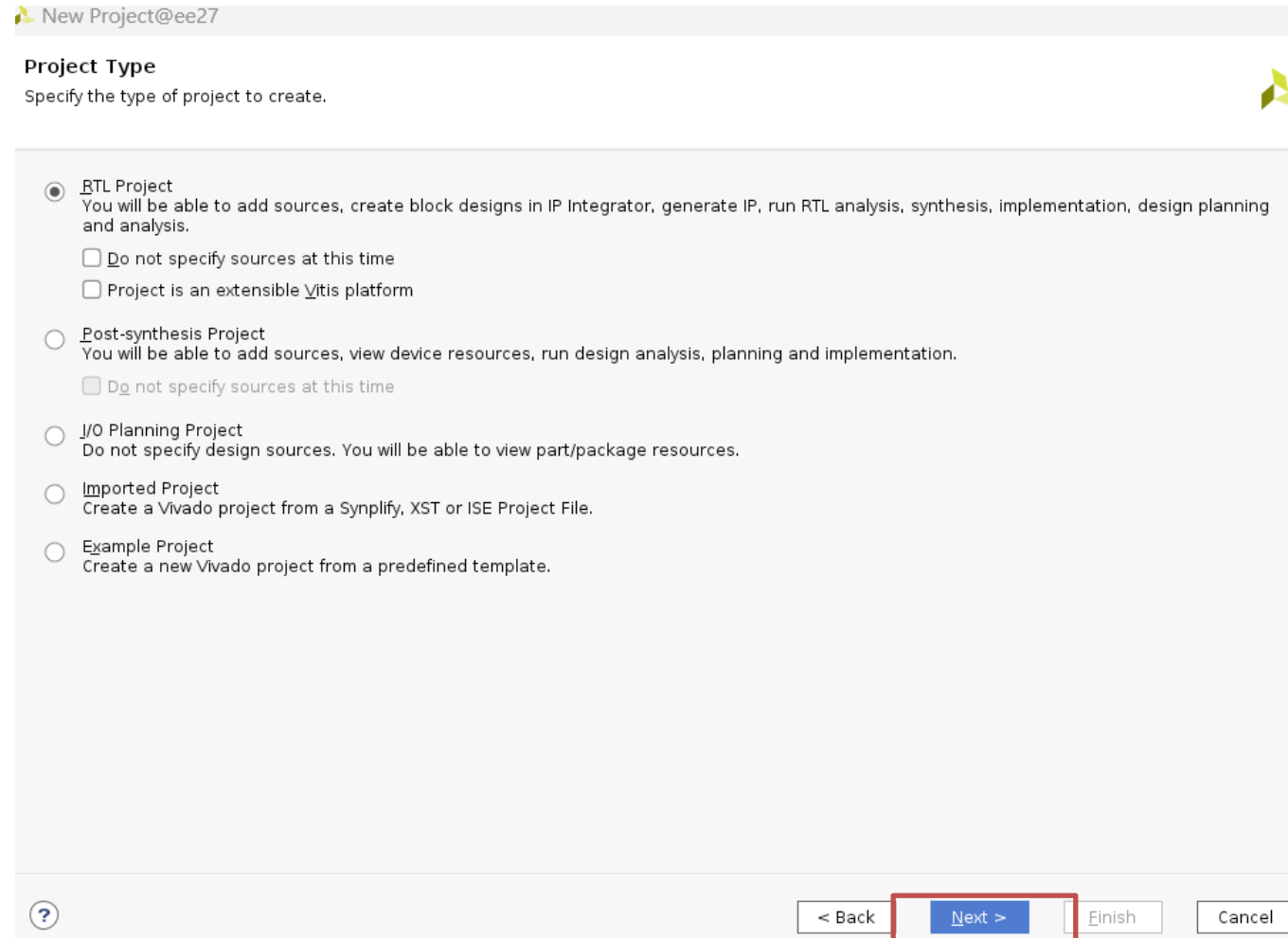
☒ Create project subdirectory

Project will be created at: /RAID2/COURSE/dcs/dcs175/project\_1

[?](#)  [< Back](#) [Next >](#) [Finish](#) [Cancel](#)

# 開啟vivado

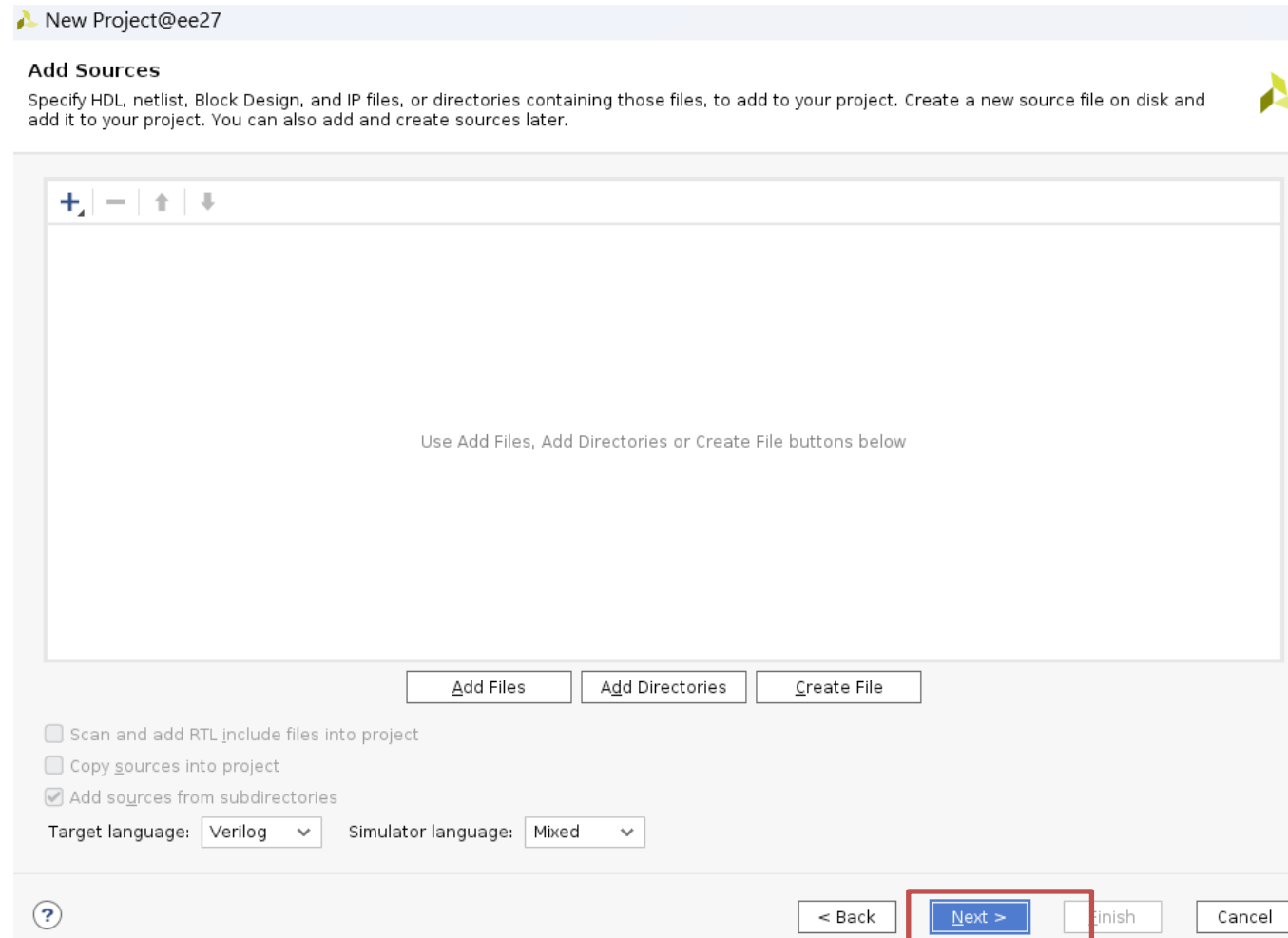
- 點選next





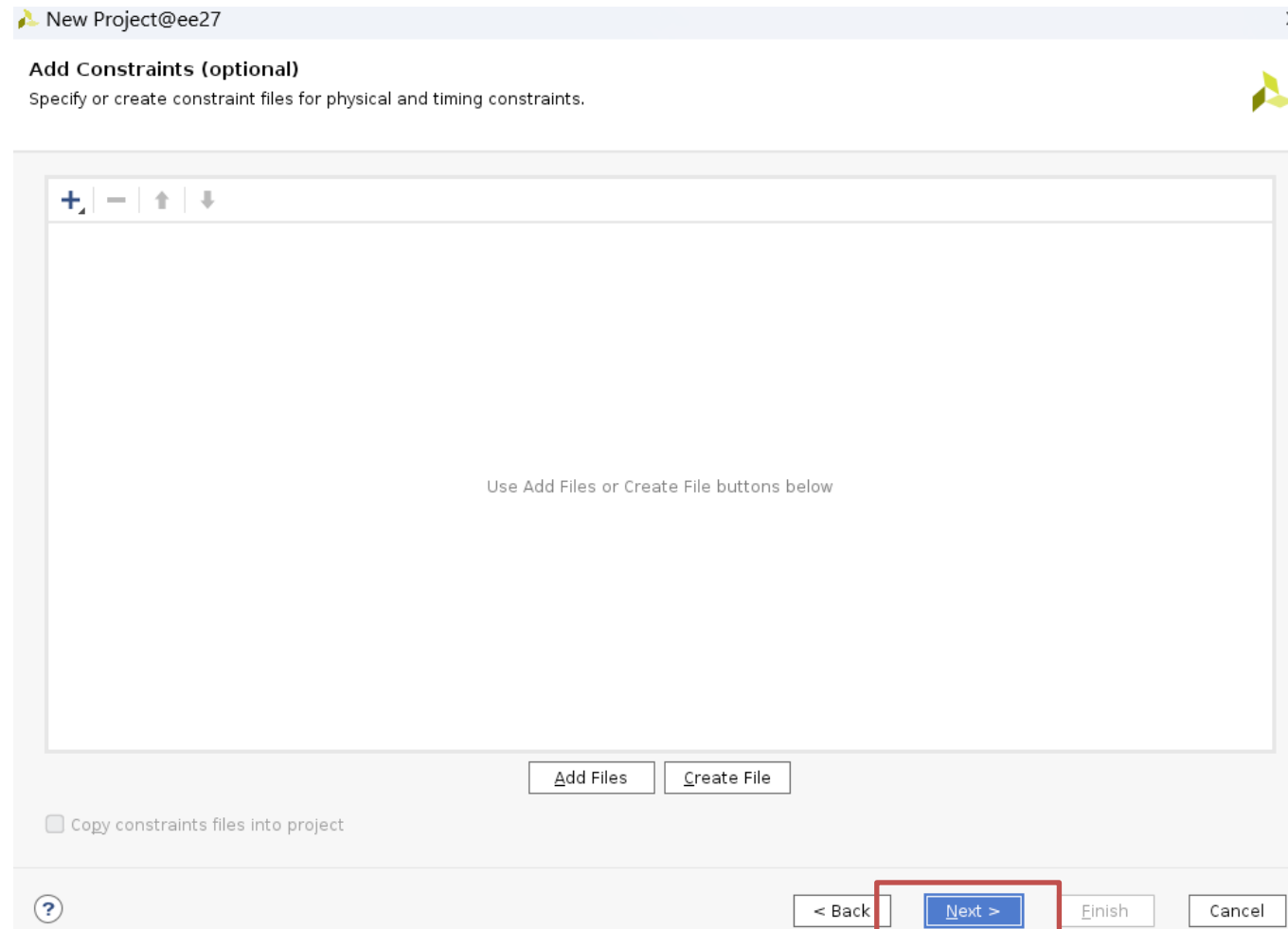
# 開啟vivado

- 點選next



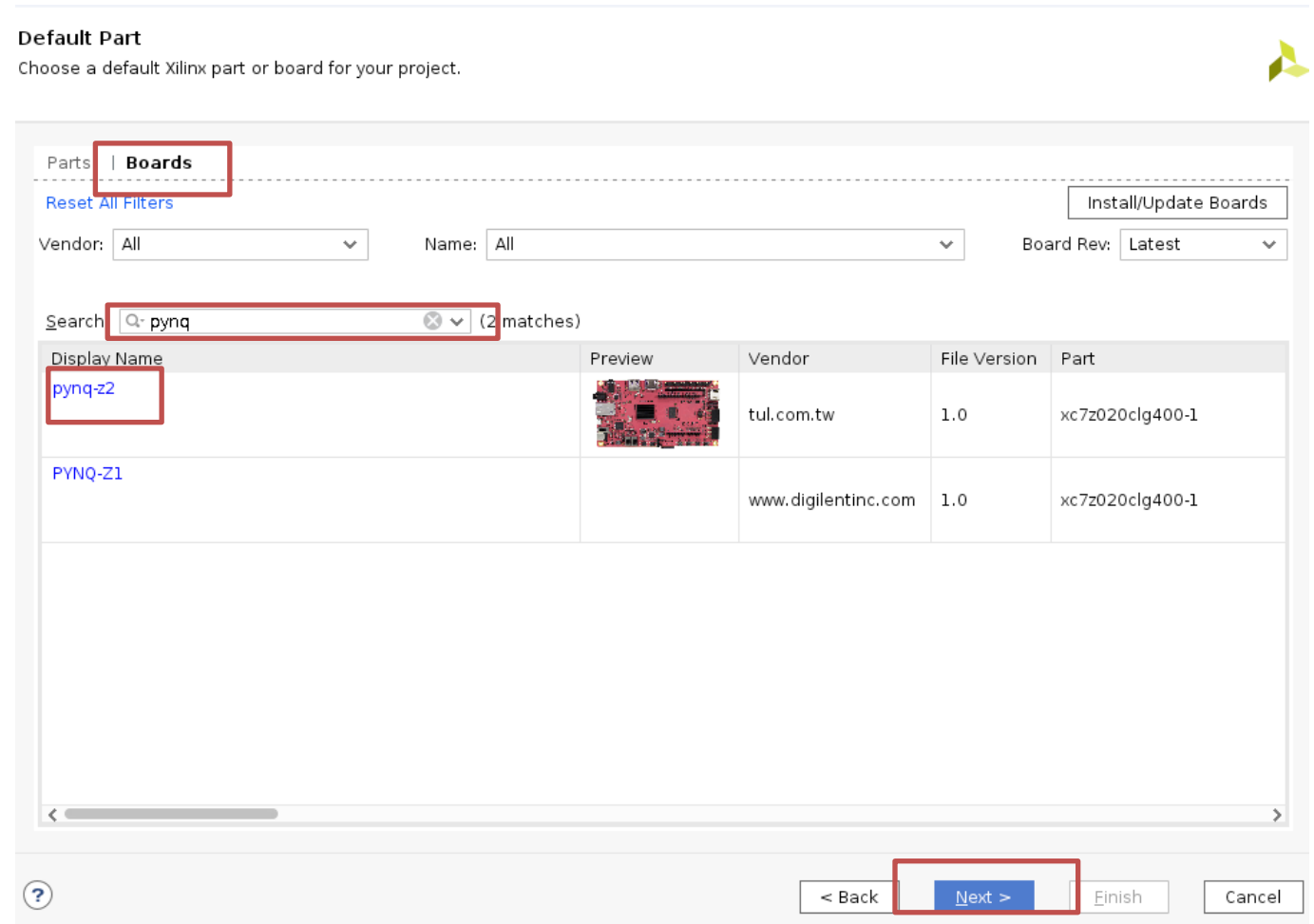
# 開啟vivado

- 點選next



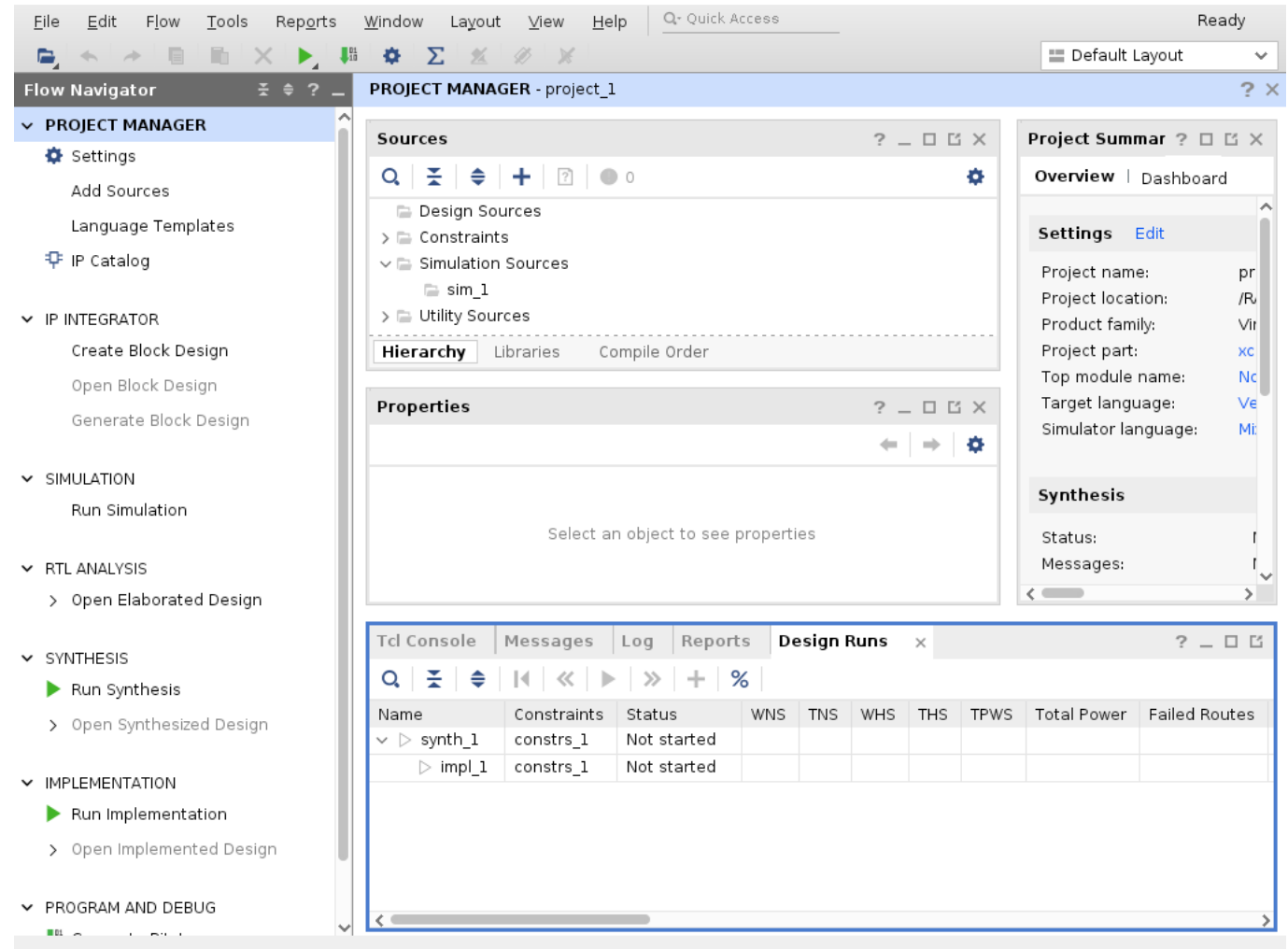
# 開啟vivado

- 點選boards
- 在search 搜尋pynq
- 點選pynq-z2
- 點選next

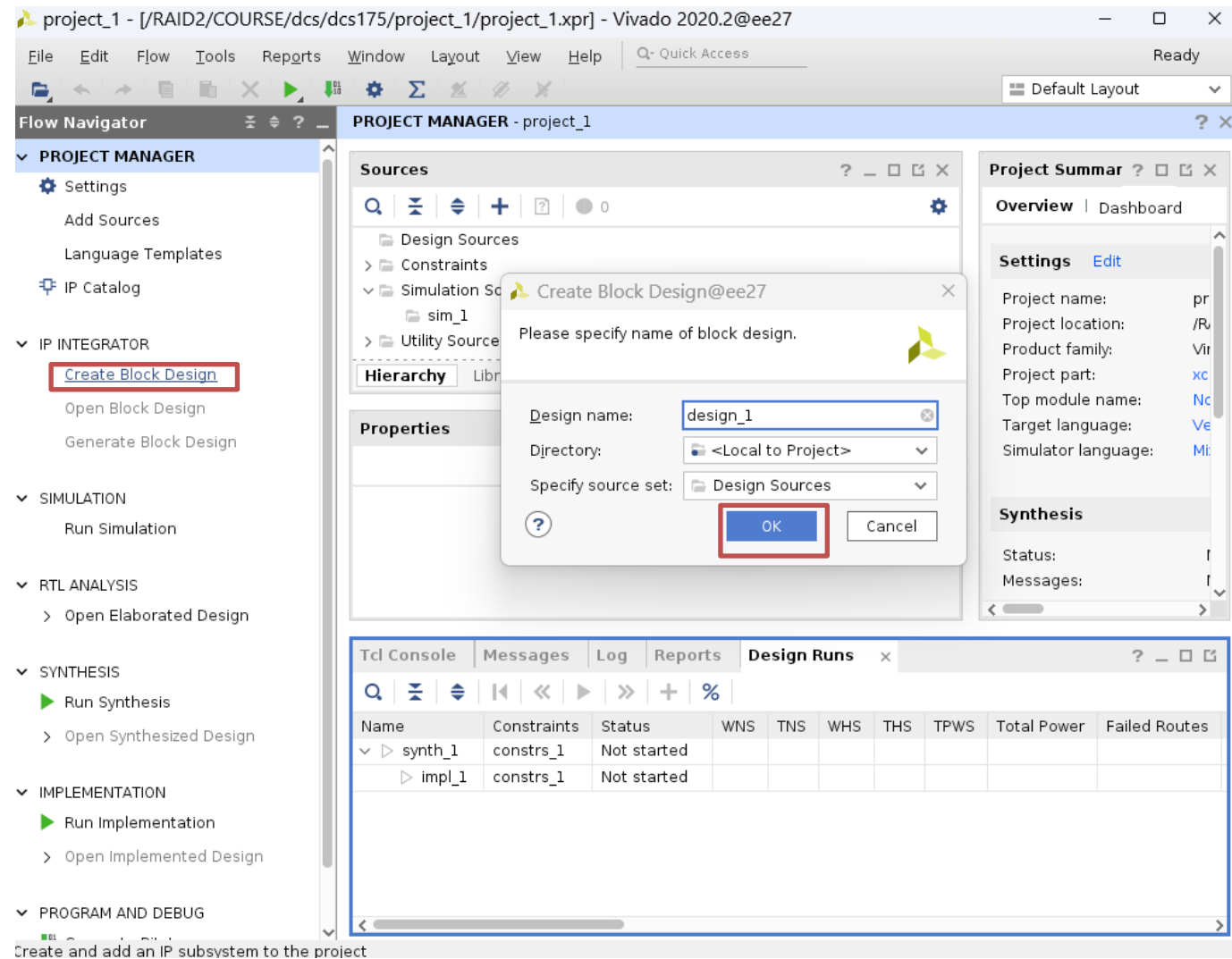


# 開啟vivado

- 點選finish後進入vivado

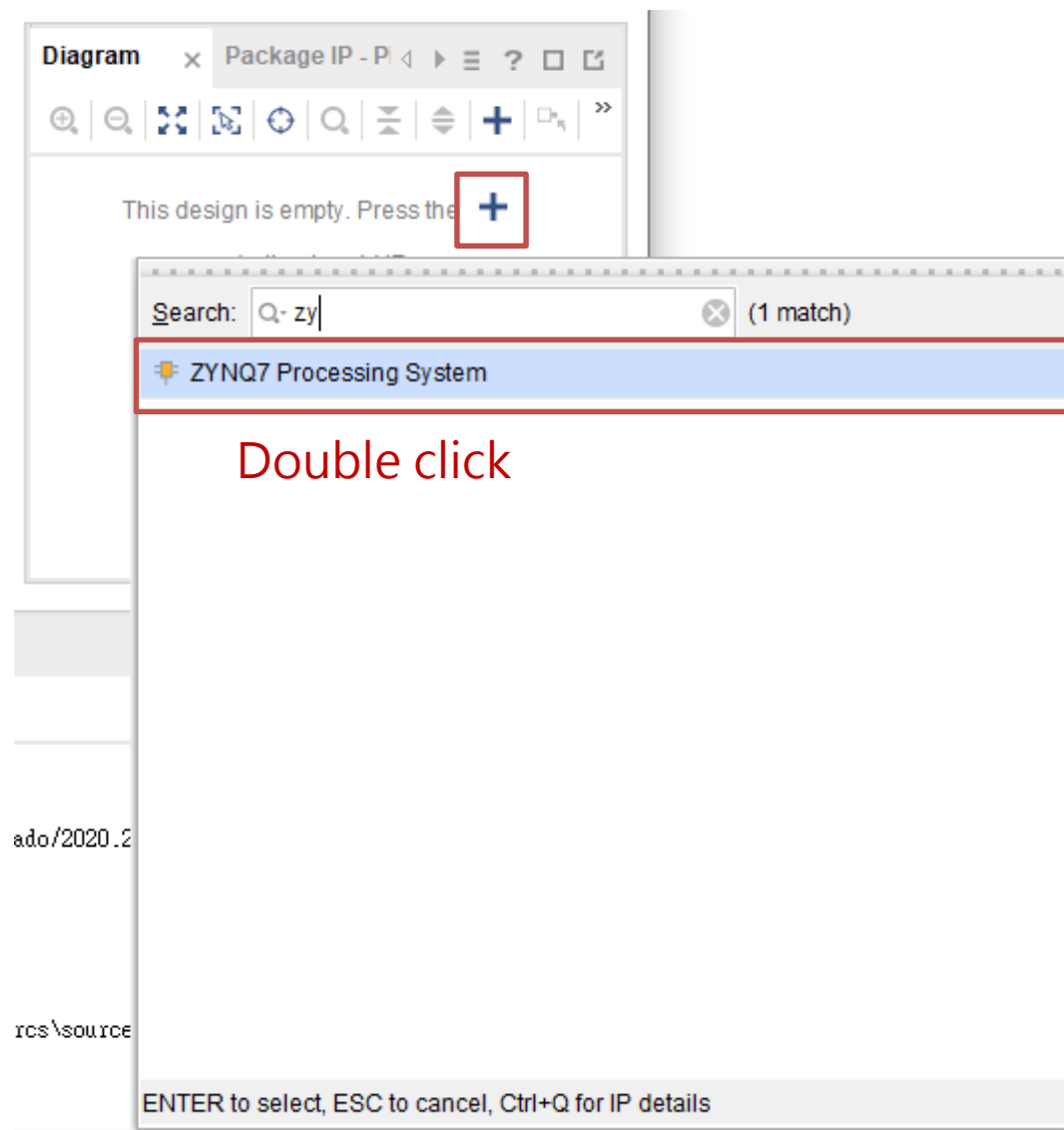


# 創建overlay



# 加入ZYNQ CPU

- CPU是拿來做資料間傳輸的控制，
- 拿來運行電腦下過來的指令。



# 加入ZYNQ CPU

The image shows the Xilinx Vivado IDE interface. On the left, the 'Diagram' tab is active, displaying a block diagram of the 'processing\_system7\_0' block. The block is labeled 'ZYNQ7 Processing System' and has several ports: 'M\_AXI\_GP0\_ACLK' on the left, 'M\_AXI\_GP0' on the right, 'FCLK\_CLK0' on the right, and 'FCLK\_RESET0\_N' on the right. The 'Run Block Automation' dialog is open on the right side of the screen. The dialog has a title bar 'Run Block Automation' and a close button. Below the title bar, there is a description: 'Automatically make connections in your design by checking the boxes of the blocks to connect. Select a block on the left to display its configuration options on the right.' The left pane shows a list of blocks with 'processing\_system7\_0' selected. The right pane shows the configuration options for 'processing\_system7\_0'. The 'Description' section states: 'This option sets the board preset on the Processing System. All current properties will be overwritten by the board preset. This action cannot be undone. Zynq7 block automation applies current board preset and generates external connections for FIXED\_IO, Trigger and DDR interfaces. NOTE: Apply Board Preset will discard existing IP configuration - please uncheck this box, if you wish to retain previous configuration. Instance: /processing\_system7\_0'. The 'Options' section has 'Make Interface External: FIXED\_IO, DDR' and two dropdown menus for 'Cross Trigger In' and 'Cross Trigger Out', both set to 'Disable'. At the bottom right, there are 'OK' and 'Cancel' buttons, with the 'OK' button highlighted by a red box.

Diagram x Package IP - Pitch x Address Editor

Designer Assistance available. **Run Block Automation**

processing\_system7\_0

M\_AXI\_GP0\_ACLK ZYNQ7 Processing System

DDR +  
FIXED\_IO +  
M\_AXI\_GP0 +  
FCLK\_CLK0  
FCLK\_RESET0\_N

Run Block Automation

Automatically make connections in your design by checking the boxes of the blocks to connect. Select a block on the left to display its configuration options on the right.

Q Z

✓ All Automation (1 out of 1 selected)  
✓ processing\_system7\_0

**Description**

This option sets the board preset on the Processing System. All current properties will be overwritten by the board preset. This action cannot be undone. Zynq7 block automation applies current board preset and generates external connections for FIXED\_IO, Trigger and DDR interfaces.

NOTE: Apply Board Preset will discard existing IP configuration - please uncheck this box, if you wish to retain previous configuration.

Instance: /processing\_system7\_0

**Options**

Make Interface External: FIXED\_IO, DDR

Cross Trigger In: Disable

Cross Trigger Out: Disable

OK Cancel

# ZYNQ7 Processing System setup

- 左鍵點兩下ZYNQ7的Block

The screenshot displays the ZYNQ7 Processing System setup interface. The top left shows a block diagram of the ZYNQ7 Processing System with various pins labeled: M\_AXI\_GP0\_ACLK, ZYNQ7, DDR, FIXED\_IO, M\_AXI\_GP0, FCLK\_CLK0, and FCLK\_RESET0\_N. The top right shows the PS-PL Configuration window with a table of settings. The bottom left shows a Critical Messages dialog box with four warning messages. The bottom right shows the OK and Cancel buttons.

**PS-PL Configuration**

Name	Select	Description
> General		
> AXI Non Secure Enablement	0	Enable AXI Non Secure Transaction
> GP Slave AXI Interface		
> HP Slave AXI Interface		
> S AXI HP0 interface	<input checked="" type="checkbox"/>	Enables AXI high performance slave interface 0
> S AXI HP1 interface	<input type="checkbox"/>	Enables AXI high performance slave interface 1
> S AXI HP2 interface	<input type="checkbox"/>	Enables AXI high performance slave interface 2
> S AXI HP3 interface	<input type="checkbox"/>	Enables AXI high performance slave interface 3
> ACP Slave AXI Interface		
> DMA Controller		
> PS-PL Cross Trigger interface	<input type="checkbox"/>	Enables PL cross trigger signals to PS and vice-versa

開啟一個High performance interface，加速stream傳輸

**Critical Messages@ee27**

There were four critical warning messages while Customize IP.

**Messages**

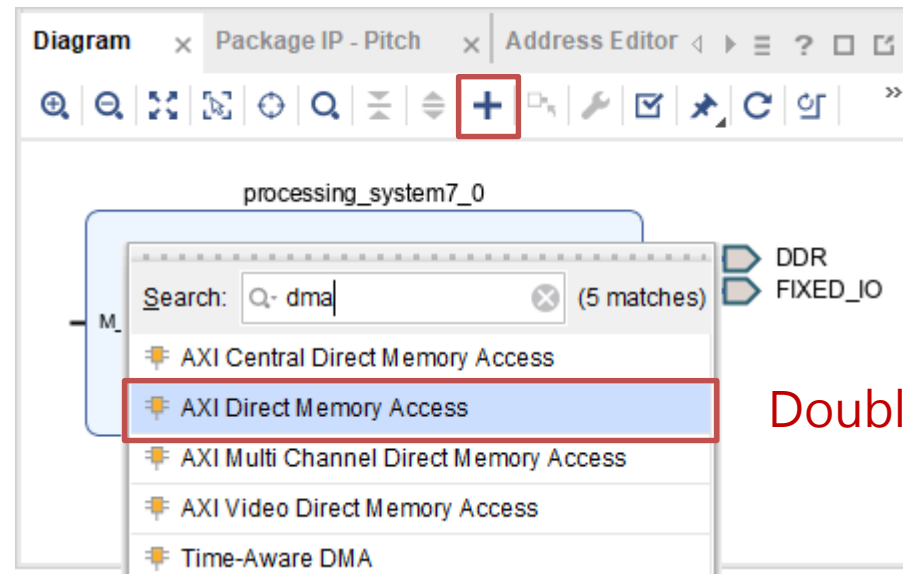
- negative DQS skew values.
- [PSU-2] Parameter : PCW\_UIPARAM\_DDR\_DQS\_TO\_CLK\_DELAY\_1 has negative value -0.006 . PS DDR interfaces might fail when entering negative DQS skew values.
- [PSU-3] Parameter : PCW\_UIPARAM\_DDR\_DQS\_TO\_CLK\_DELAY\_2 has negative value -0.009 . PS DDR interfaces might fail when entering negative DQS skew values.
- [PSU-4] Parameter : PCW\_UIPARAM\_DDR\_DQS\_TO\_CLK\_DELAY\_3 has negative value -0.033 . PS DDR interfaces might fail when entering negative DQS skew values.

OK Open Messages View OK Cancel



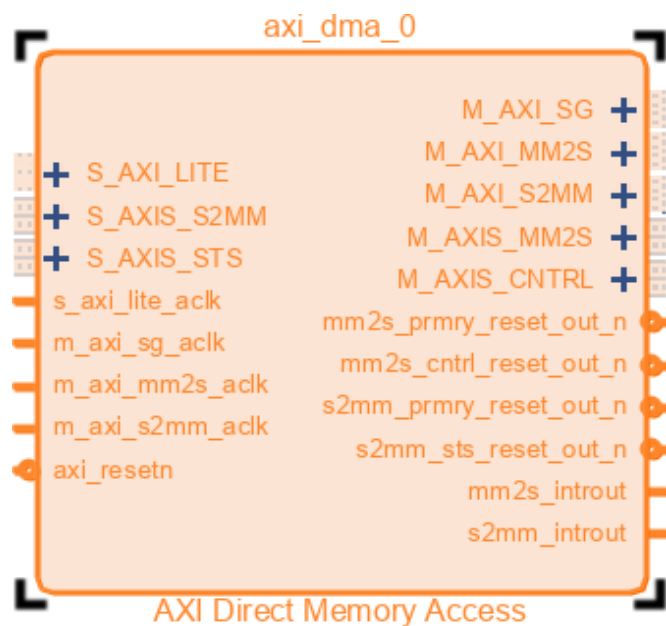
# 加入DMA

- DMA是用來與PYNQ板上的記憶體做溝通橋梁，
- 分配記憶體空間、存取記憶體中的資料，
- 取資料給你的Design、你的Design再寫資料回記憶體。



# AXI DMA setup

- 左鍵點兩下DMA的Block



Component Name: filter/fir\_dma

☐ Enable Asynchronous Clocks (Auto)

☐ Enable Scatter Gather Engine

☐ Enable Micro DMA

☐ Enable Multi Channel Support

☐ Enable Control / Status Stream

Width of Buffer Length Register (8-26): 26 bits

Address Width (32-64): 64 bits

☒ Enable Read Channel

Number of Channels: 1

Memory Map Data Width: 32

Stream Data Width: 32

Max Burst Size: 16

☐ Allow Unaligned Transfers

☒ Enable Write Channel

Number of Channels: 1

Memory Map Data Width: 32

Stream Data Width (Auto): 32

Max Burst Size: 16

☐ Allow Unaligned Transfers

☐ Use Rxlenght In Status Stream

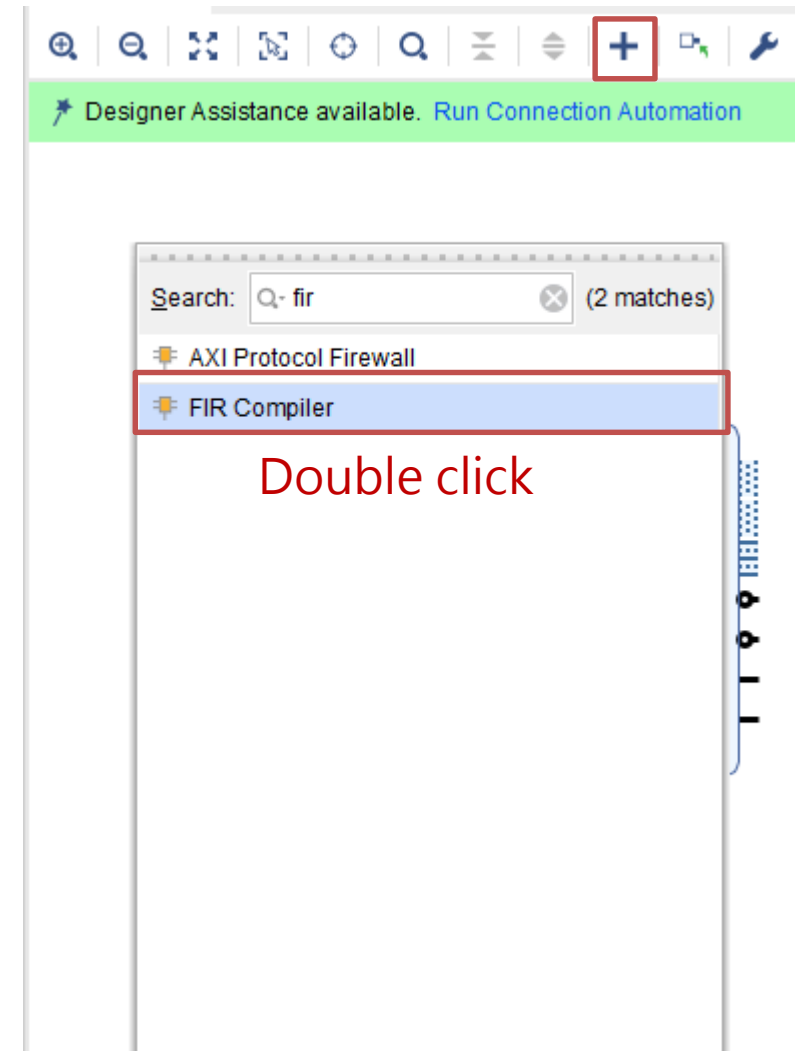
☐ Enable Single AXI4 Data Interface

開到最大

這邊是依照SPEC做更改，  
這次Lab依照下面參數設定

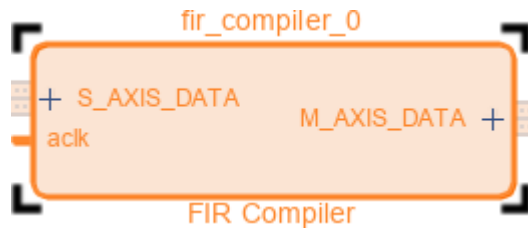
# 加入Project的IP

- 將寫好的.sv接上AXI4介面後包成IP，在這裡呼叫。
- 這次Lab使用內建FIR即可。



# FIR setup

- 點兩下FIR的block



Filter Options | Channel Specification | Implementation | Detailed Implementation | Interface | Summary

**Filter Coefficients**

Select Source: Vector

Coefficient Vector: -255, -260, -312, -288, -144, 153, 616, 1233, 1963, 2739, 3474, 4081, 4481, 4620, 4481, 4081, 3474, 2739, 1963, 1233, 616, 153, -144, -288, -312, -260, -255

Coefficient File: no\_coe\_file\_loaded

Number of Coefficient Sets: 1 [1 - 1024]

Number of Coefficients (per set): 21

☐ Use Reloadable Coefficients

The IP name you have specified is long. The Windows operating system has a path length limitations. It is recommended you use shorter names to reduce the likelihood of issues.

- 本次要做Sampling Rate 100MHz、Pass Band 0~5MHz、Stop Band 10M~50MHz的FIR LP Filter
- Coefficient Vector改成下面的數列:  
-255, -260, -312, -288, -144, 153, 616, 1233, 1963, 2739, 3474, 4081, 4481, 4620, 4481, 4081, 3474, 2739, 1963, 1233, 616, 153, -144, -288, -312, -260, -255

# FIR setup

Filter Options **Channel Specification** Implementation Detailed Implementation

Hardware Oversampling Specification

Select Format Frequency Specification ▾

Sample Period (Clock Cycles) 1 [1.0 - 1.0E7]

Input Sampling Frequency (MHz) 100 [1.0E-6 - 189952.0]

Clock Frequency (MHz) 100 [0.390625 - 742.0]

Filter Options Channel Specification **Implementation** Detailed Implementation Interface

Data Path Options

☐ AUTO ☐ MANUAL Input Data Type Signed ▾

☐ MANUAL Input Data Width 32 [2 - 47]

☐ AUTO Input Data Fractional Bits 0 [0 - 32]

Output Rounding Mode Non Symmetric Rounding Up ▾

Output Width 32 [1 - 47]

Output Fractional Bits : 0

Filter Options Channel Specification Implementation Detailed Implementation **Interface** Summary

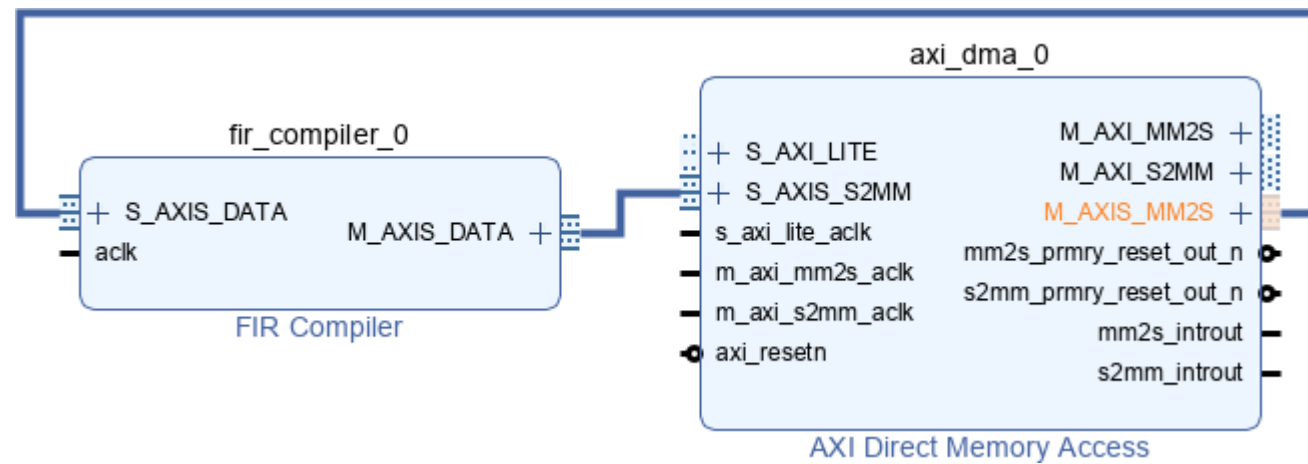
Data Channel Options

TLAST Packet Framing ▾

☒ Output TREADY ☒ Input FIFO

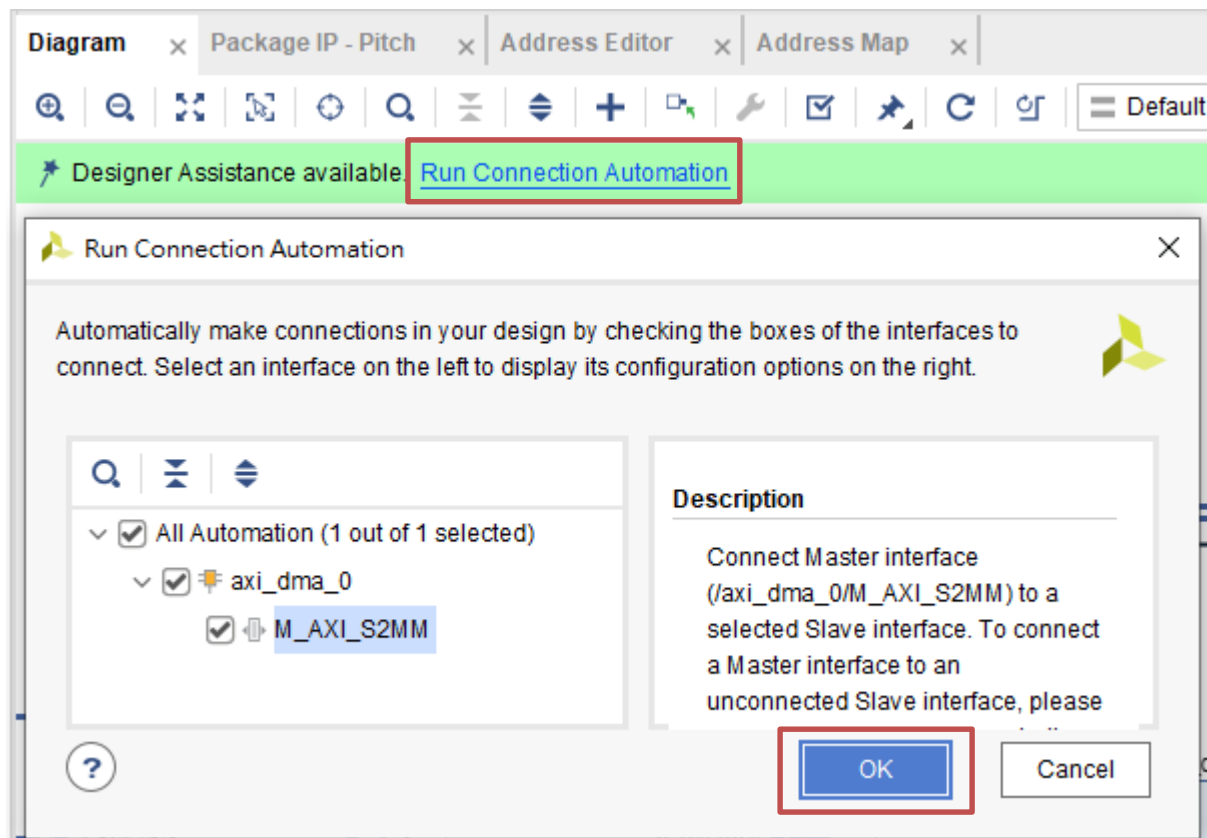
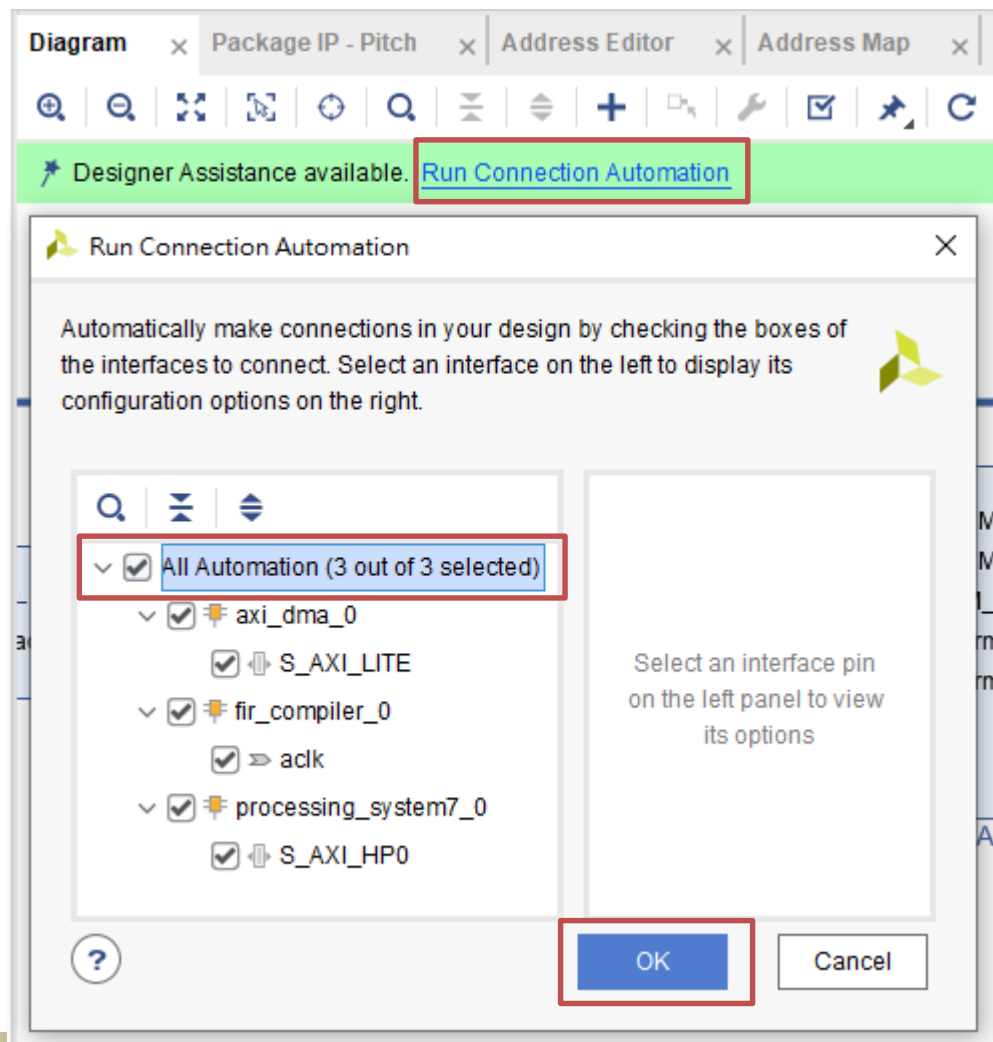
click ok

- 將FIR與DMA相連接，對著Port長按拖曳到另一個Port即可連接。

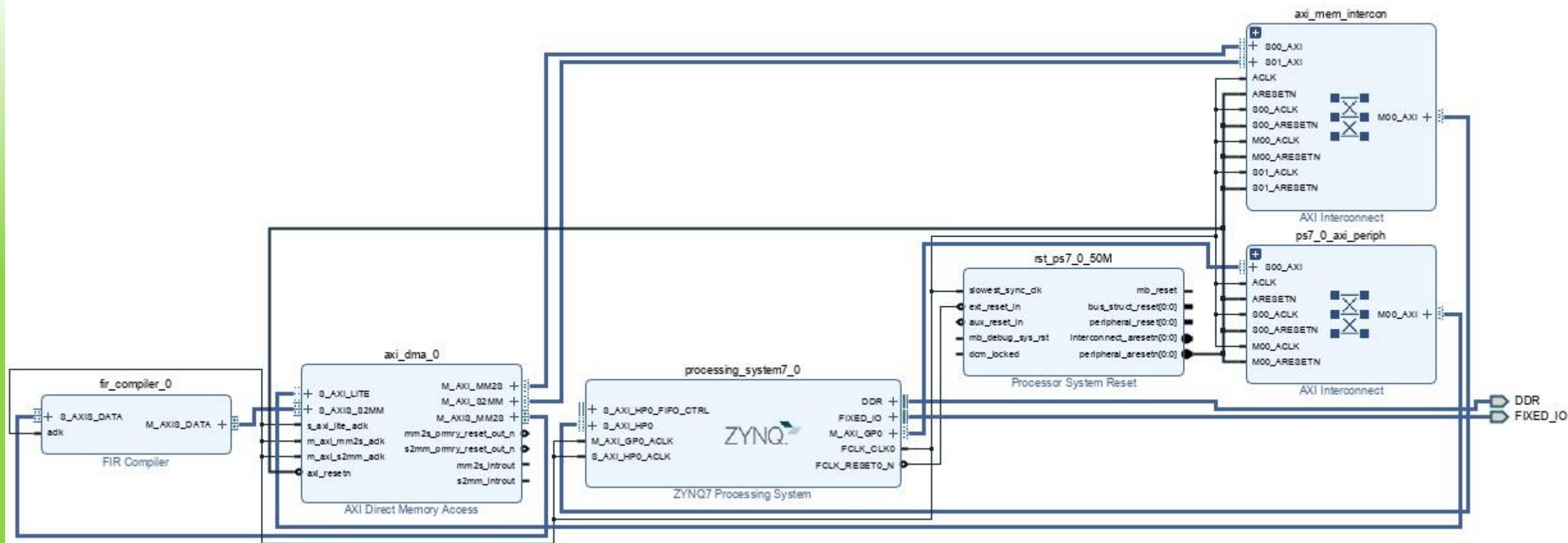


# Run Connection Automation

- 跑兩次Run Connection Automation



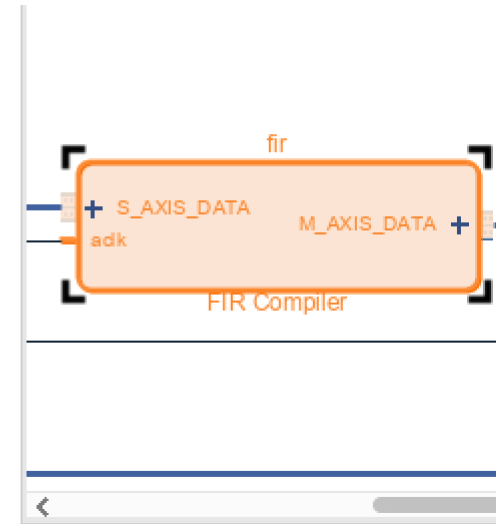
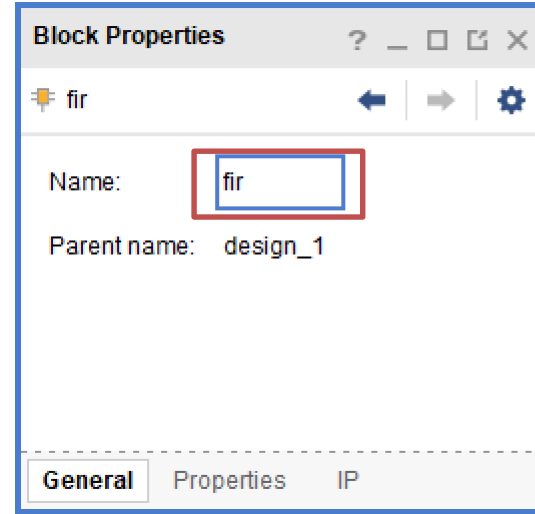
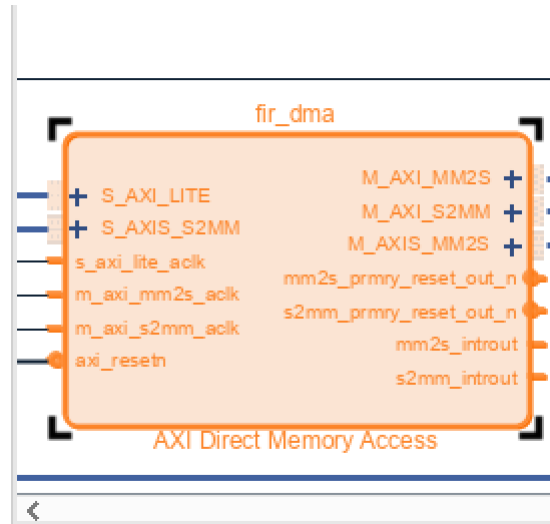
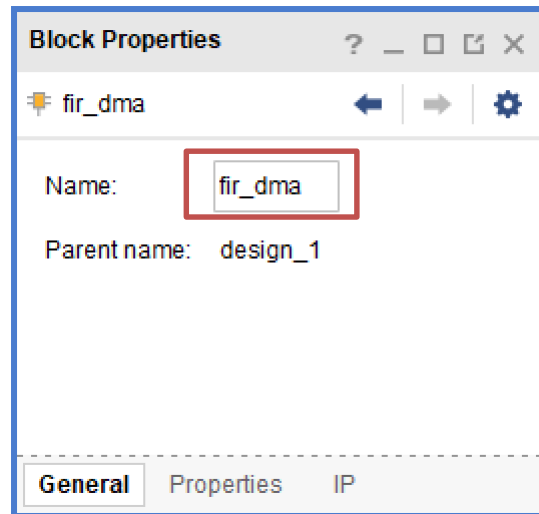
- 至此為止，Block Diagram會大致長這樣



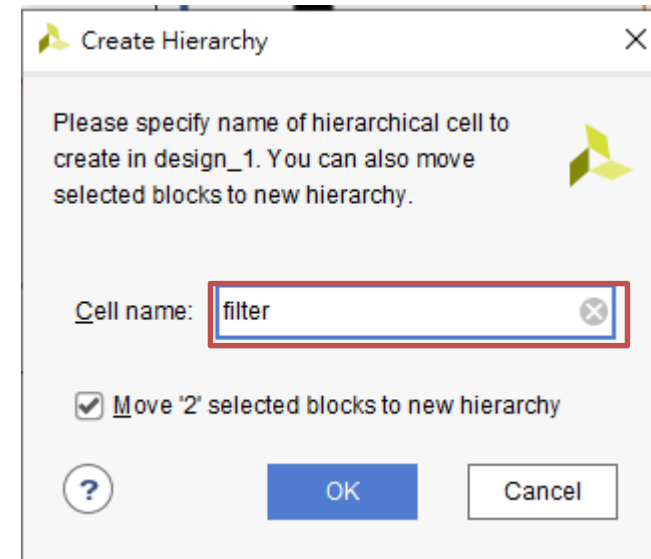
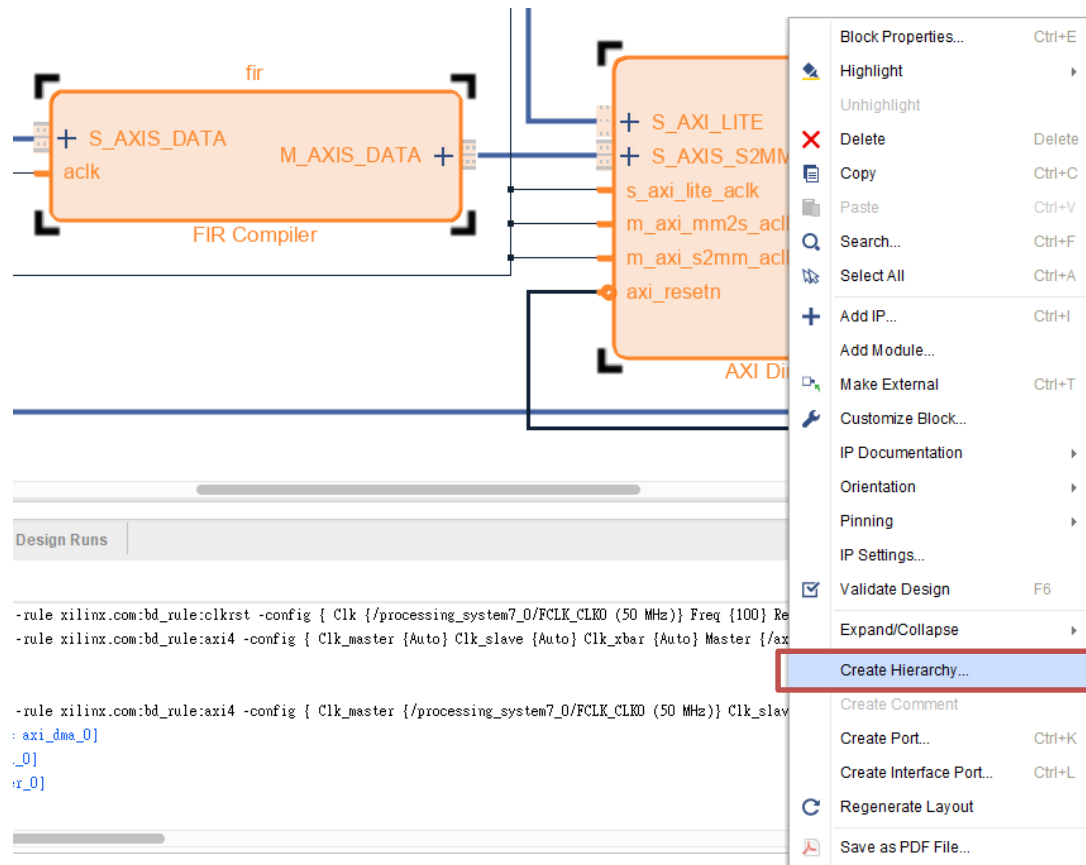


# Rename

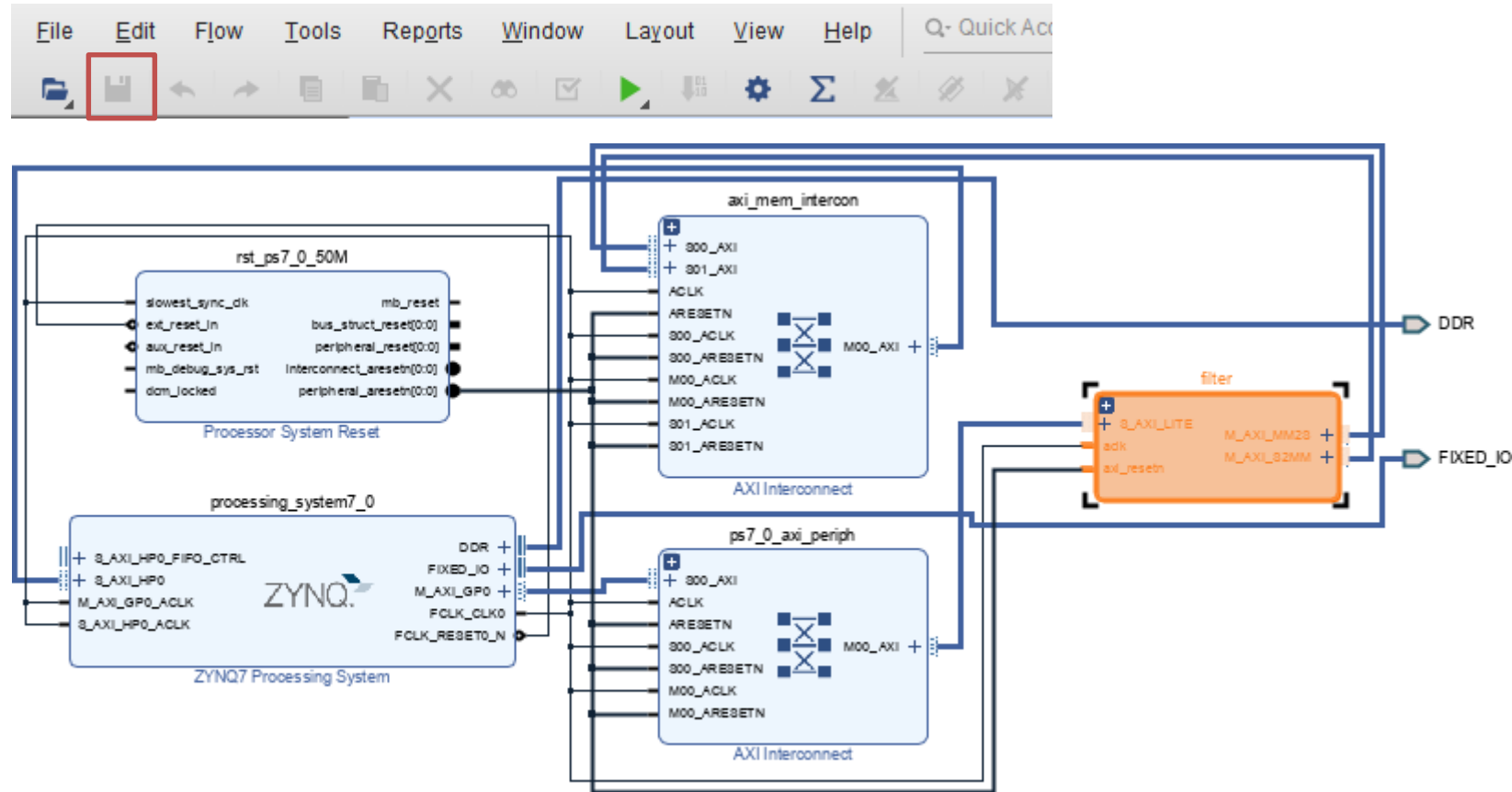
- 將fir\_compiler\_0和axi\_dma\_0的名稱改成fir和fir\_dma，
- 以便在Jupyter notebook呼叫module。



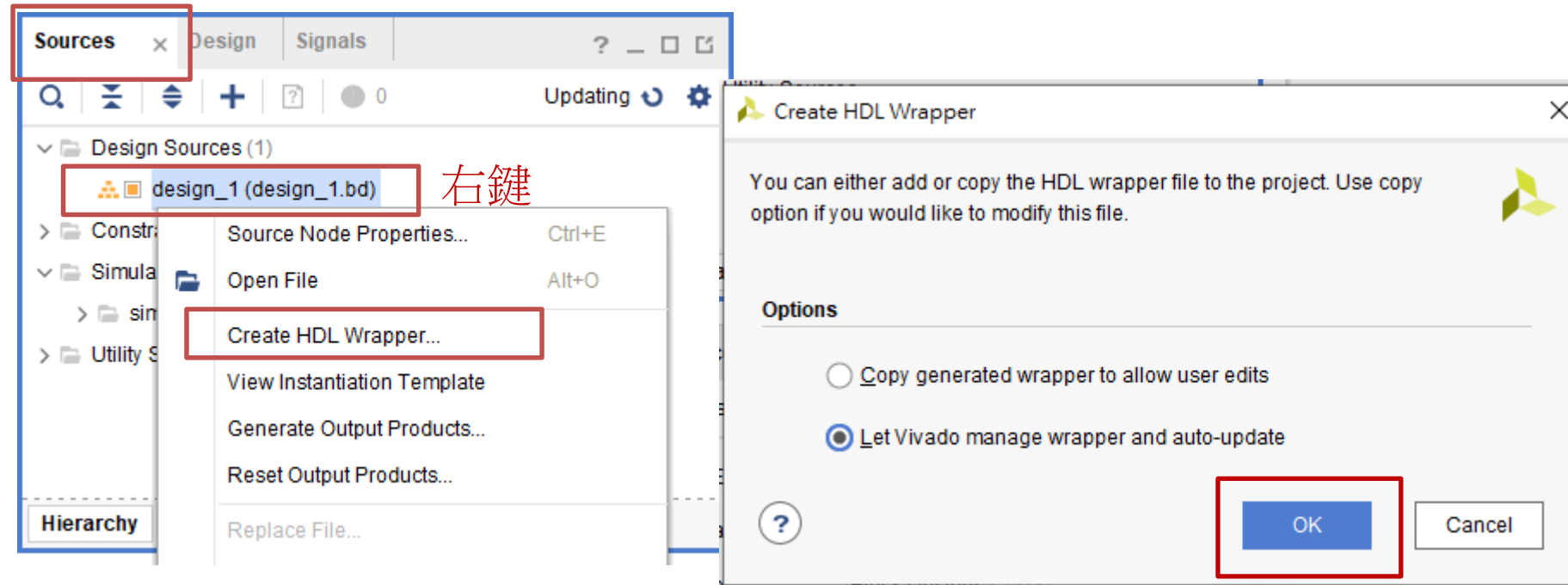
- Ctrl+左鍵，同時選取fir和fir\_dma→右鍵→ Create Hierarchy
- 將fir和fir\_fma包在一起，取名叫filter。



- 至此為止，Block Diagram會大致長這樣，可以先存檔。

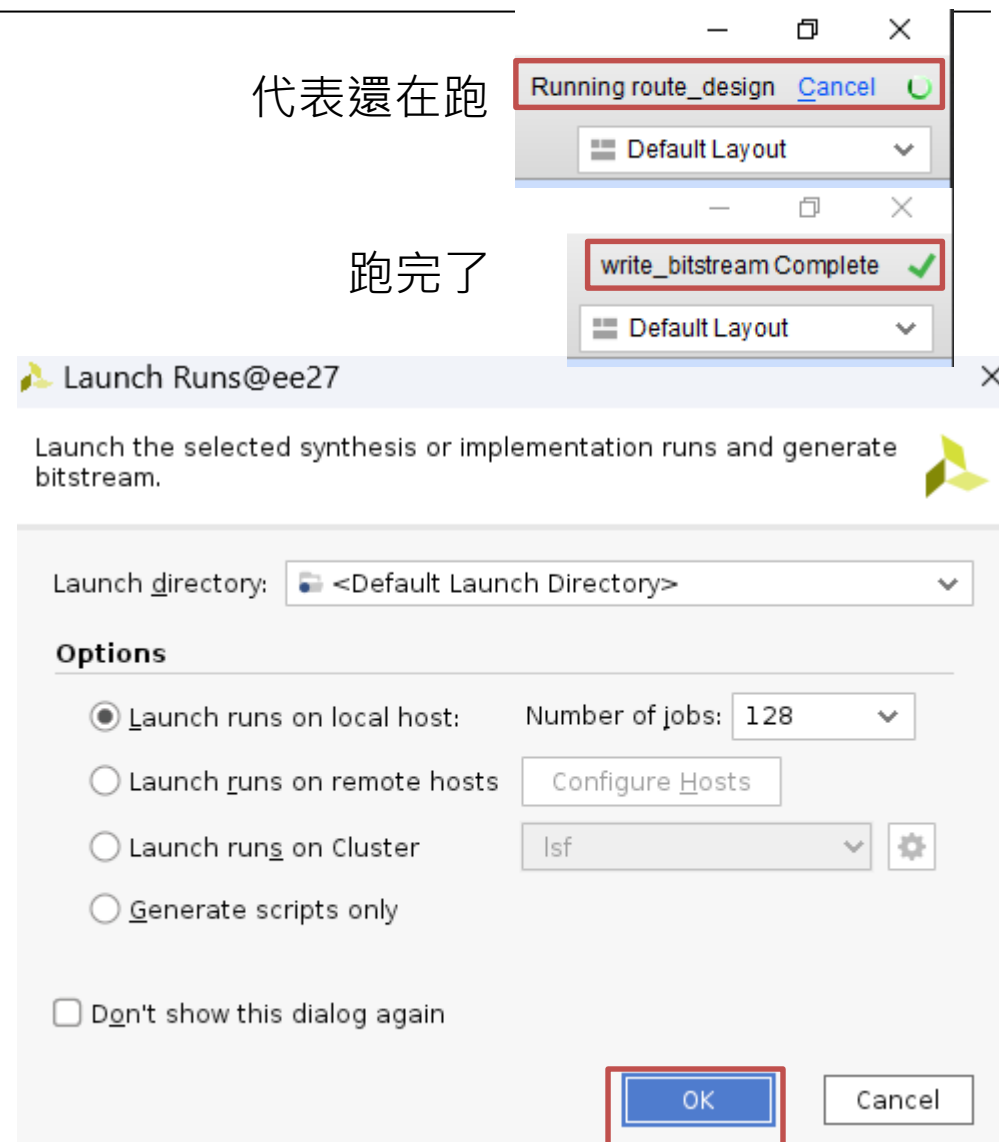
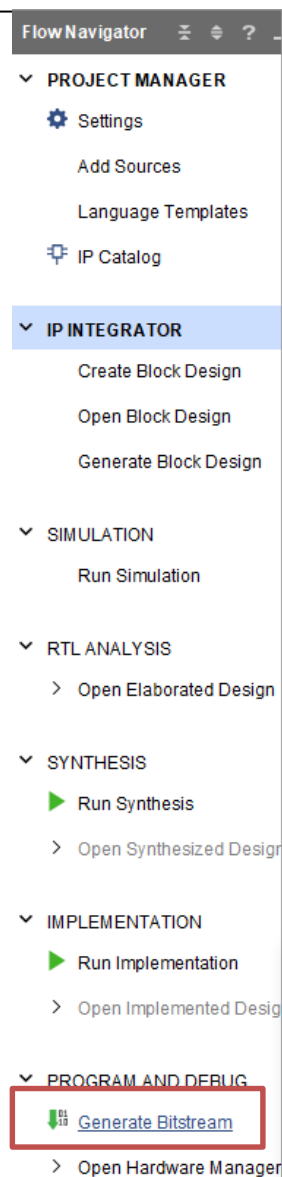
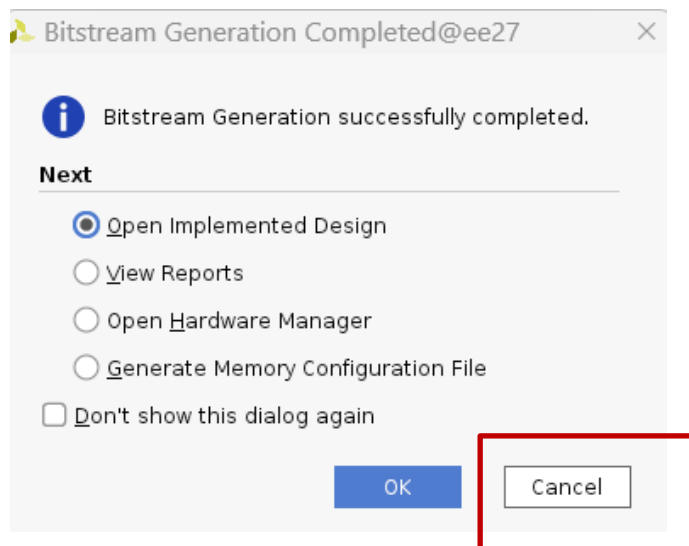


# Generate HDL wrapper



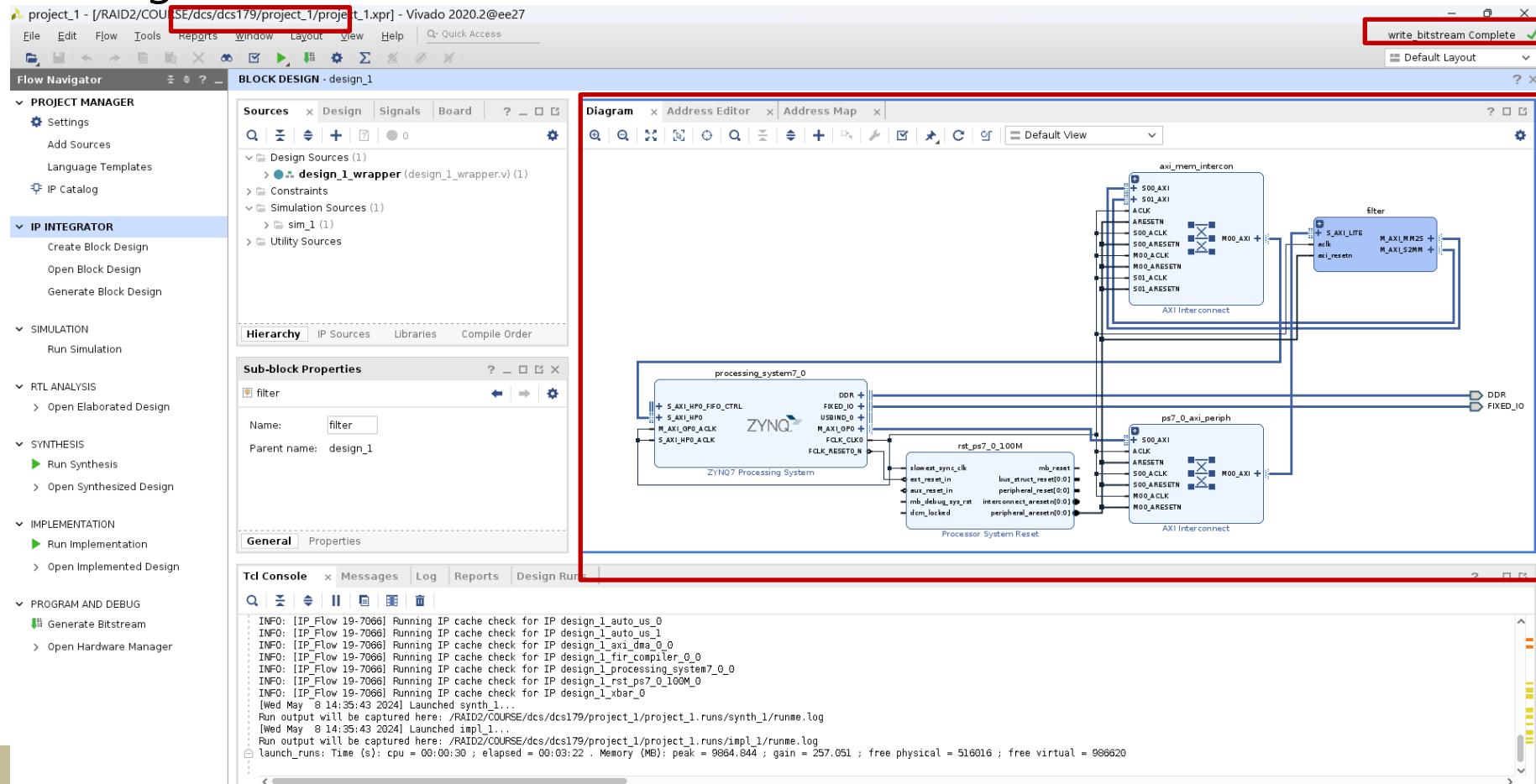
# Generate Bitstream

- 這個會跑一段時間，  
右上角沒有在跑東西之後  
再往下一步。



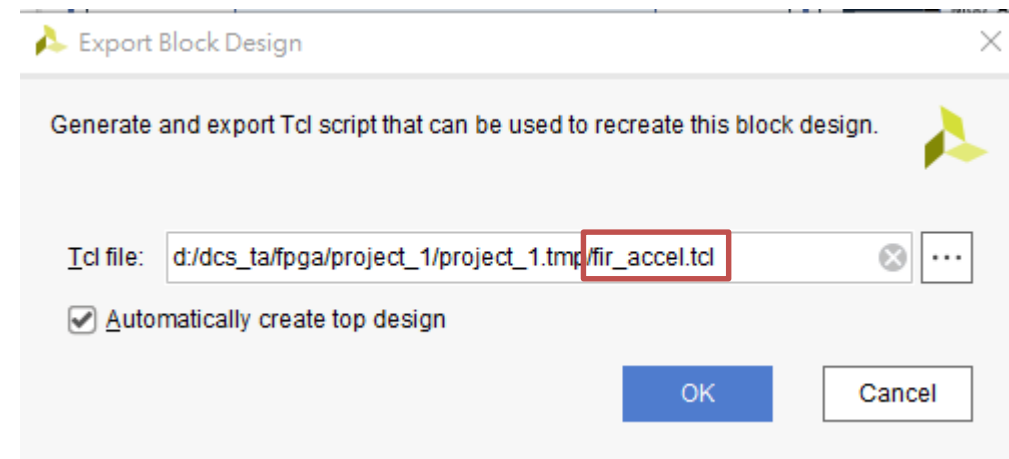
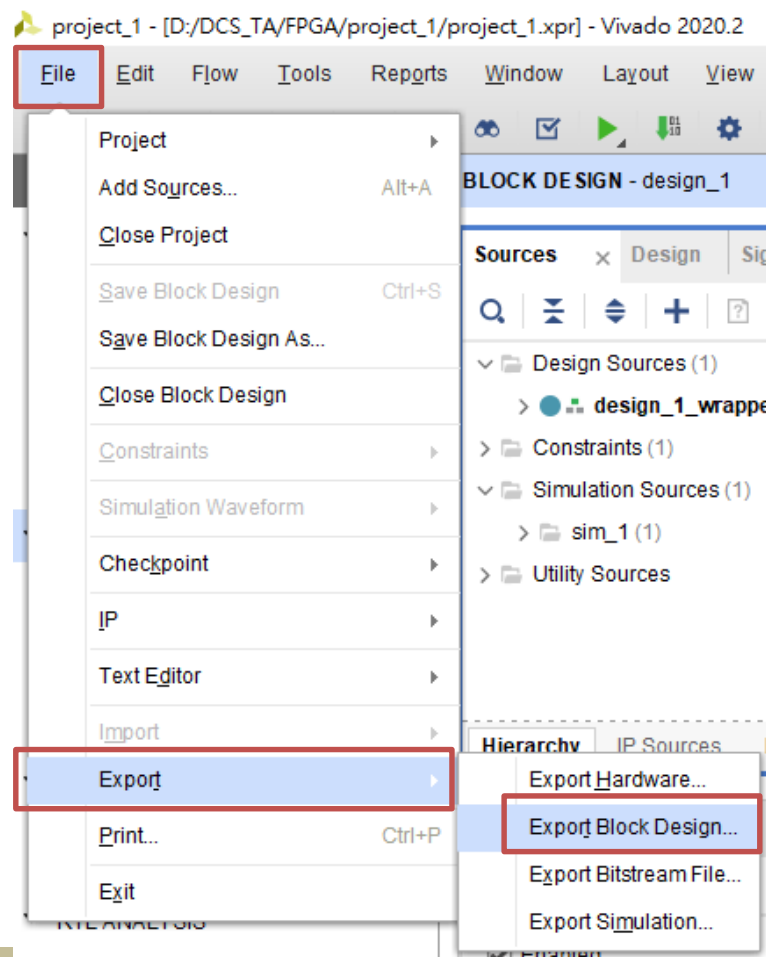
# 截圖vivado畫面 (demo依據之一)

- 截圖須包含
  1. 工作站帳號
  2. Write\_bitstream complete 勾勾
  3. 完整Diagram

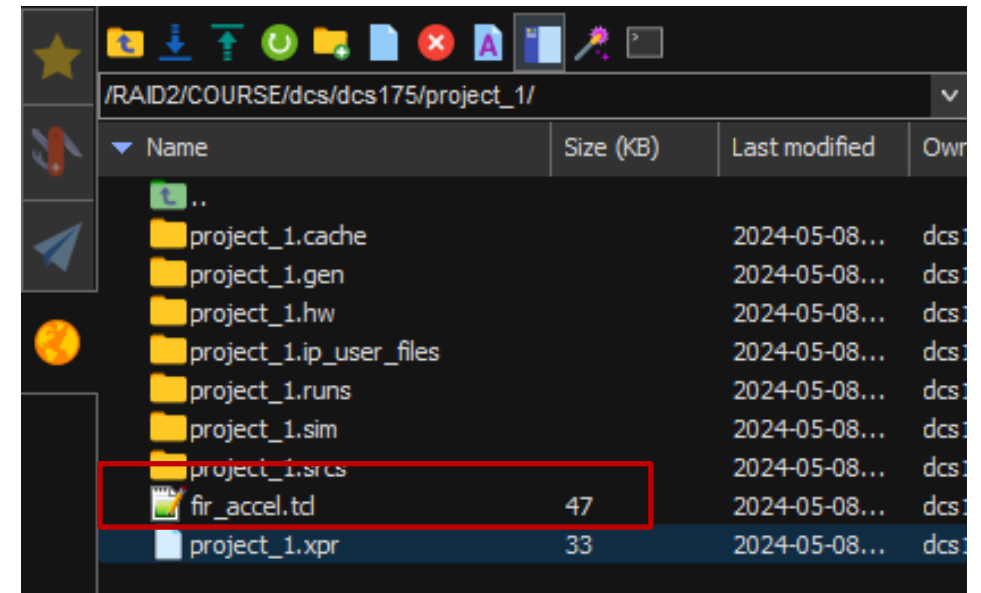
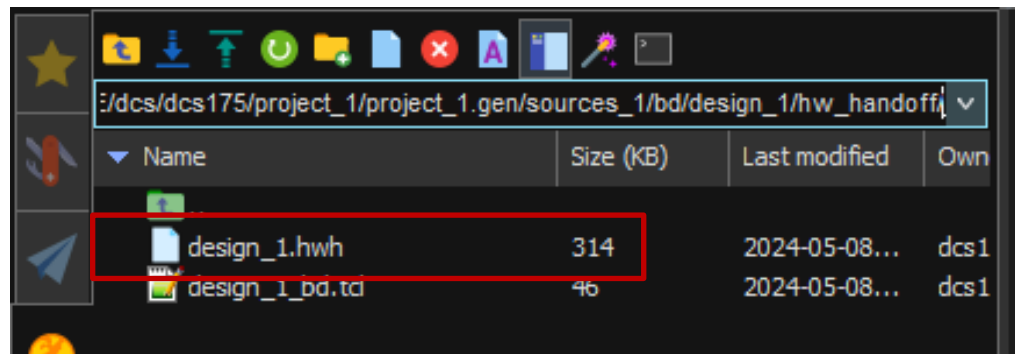
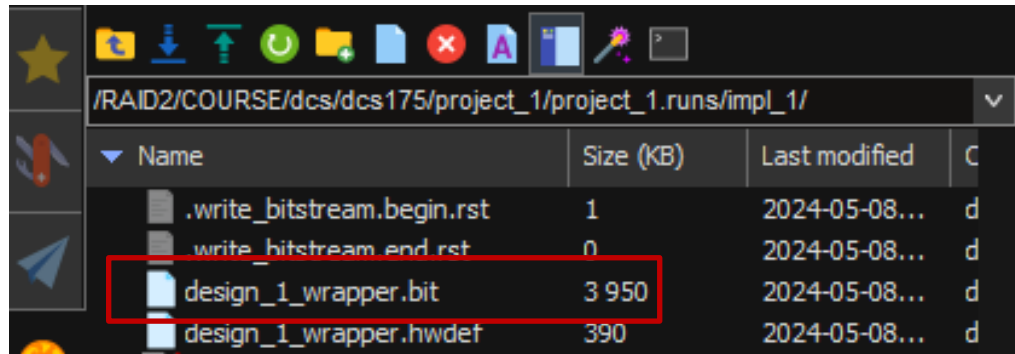


# Generate tcl file

- 用Jupyter叫module overlay的時候，
- PYNQ需要讀tcl檔取得Hardware Design。



- 根據下面的路徑找到design\_1\_wrapper.bit、design\_1.hwh 和 fir\_accel.tcl，下載這三個檔案到自己的電腦，並將design\_1\_wrapper.bit、design\_1.hwh 重新命名成fir\_accel.bit、fir\_accel.hwh





# Connect to jupyter notebook

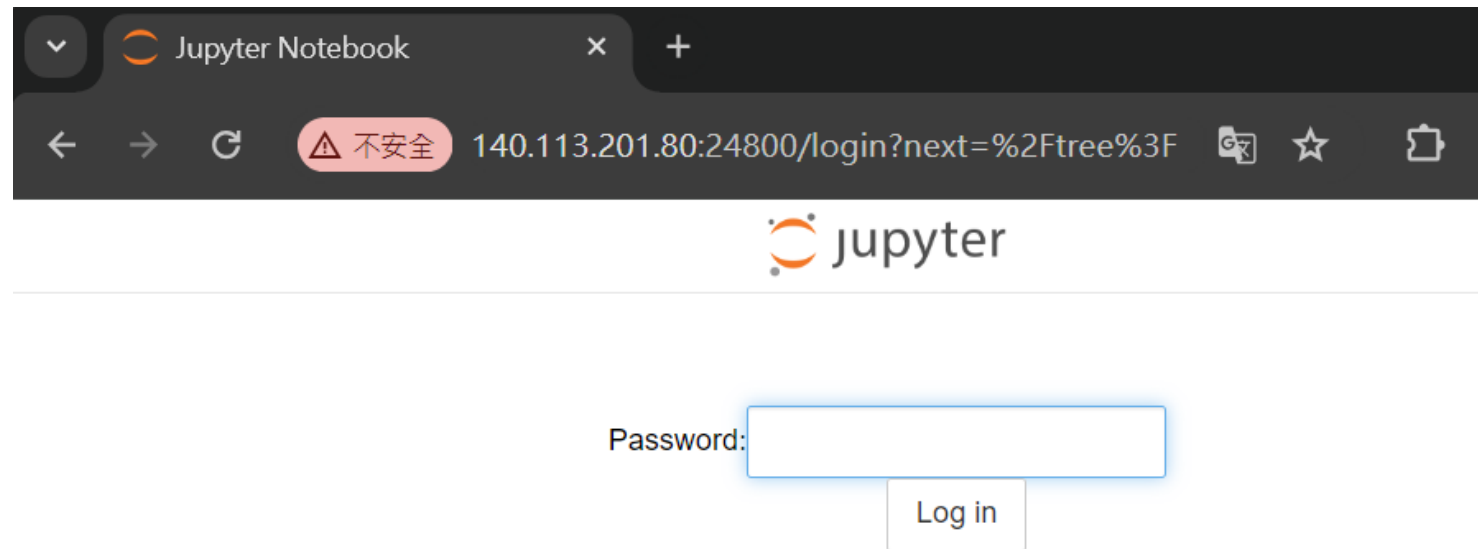
# 在表單上登記FPGA使用時間

---

- 因FPGA數量只有48個，請同學輪流使用
- 請同學在連接上jupyter notebook前先確認板子是沒人使用的，避免跑到一半的結果被其他人洗掉
- 請同學先到下面網址登記使用時間，一個時段為2小時，如果在登記時段結束時還沒完成才可再登記下個時段
- <https://docs.google.com/spreadsheets/d/1GCMX8VIO5hEjZ77t3UhFzwIPOfHLOxZT1oMgRcgrvYo/edit?usp=sharing>
- 另外若板子有連線問題請在表單的註記欄打X，並先用其他板子

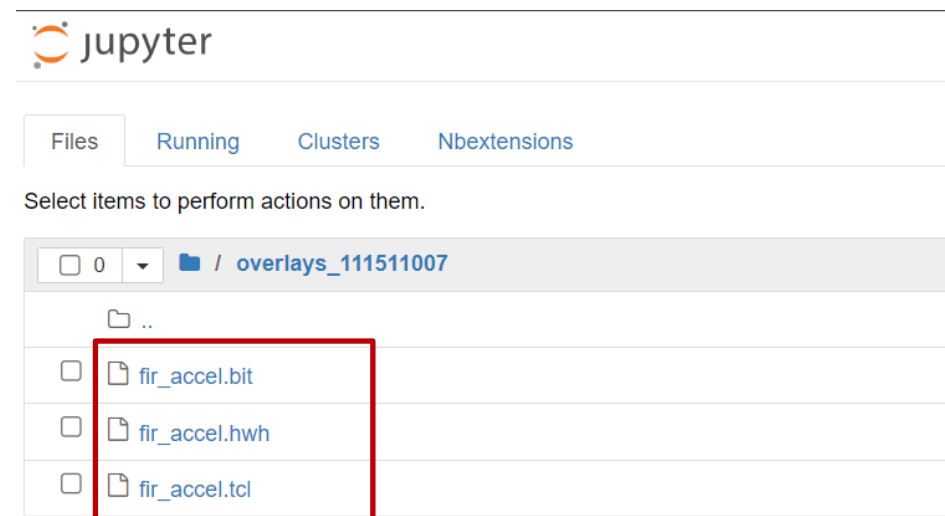
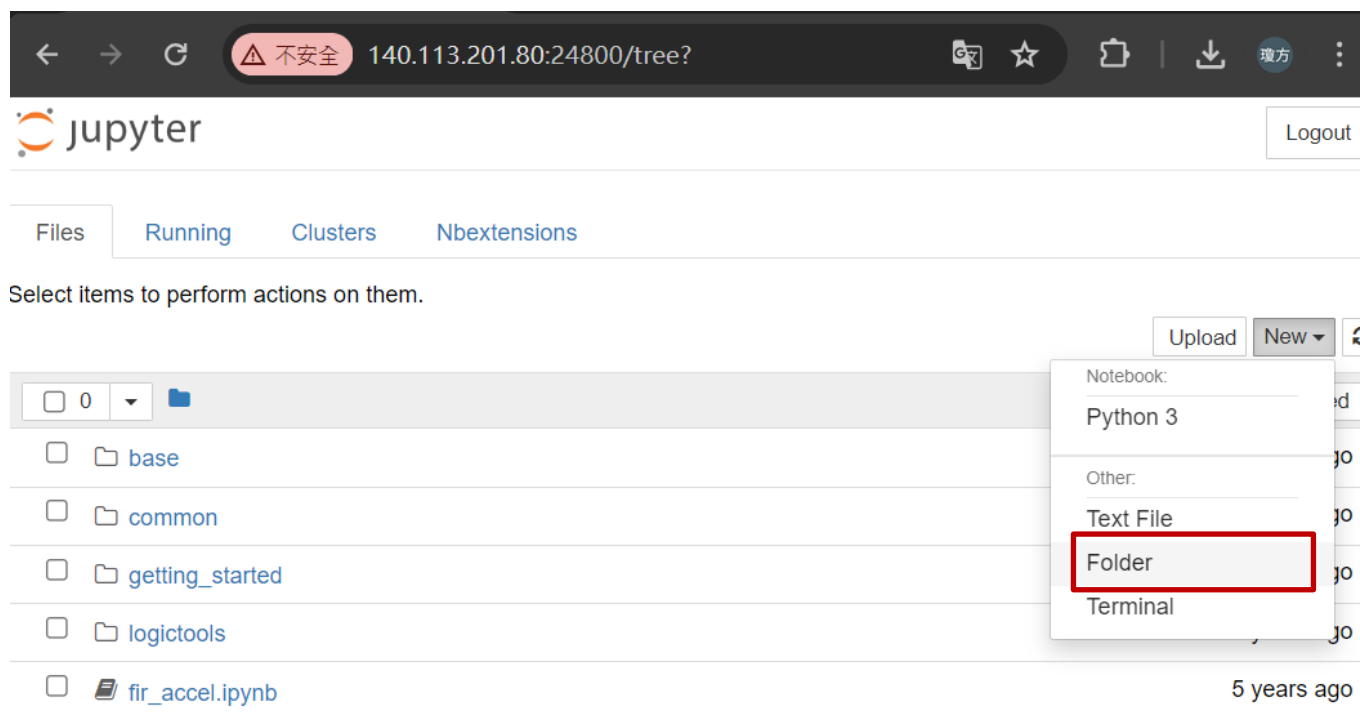
# 開啟Jupyter Notebook

- 打開瀏覽器 → 輸入 140.113.201.80:板子編號
- 密碼: xilinx



# 上傳檔案到jupyter notebook

- 創建資料夾，命名為overlays\_學號
- 將剛剛下載的三個檔案上傳至 overlays\_學號 資料夾



# fir\_accel.ipynb

```
from pynq import Overlay
import pynq.lib.dma

# Load the overlay
overlay = Overlay('/home/xilinx/pynq/overlays/fir_accel/fir_accel.bit') //讀入Overlay

# Load the FIR DMA
dma = overlay.filter.fir_dma
```

```
from pynq import Xlnk
import numpy as np

# Allocate buffers for the input and output signals
xlnk = Xlnk() //xlnk申請記憶體空間，指標到虛擬記憶體
in_buffer = xlnk.cma_array(shape=(n,), dtype=np.int32)
out_buffer = xlnk.cma_array(shape=(n,), dtype=np.int32) //分配記憶體空間給in_buffer和out_buffer

# Copy the samples to the in_buffer
np.copyto(in_buffer,samples) //將input資料寫入in_buffer

# Trigger the DMA transfer and wait for the result
import time
start_time = time.time()
dma.sendchannel.transfer(in_buffer) //開始進行DMA傳輸
dma.recvchannel.transfer(out_buffer)
dma.sendchannel.wait() //等待DMA傳輸結束
dma.recvchannel.wait()

stop_time = time.time()
hw_exec_time = stop_time-start_time
print('Hardware FIR execution time: ',hw_exec_time)
print('Hardware acceleration factor: ',sw_exec_time / hw_exec_time)

# Plot to the notebook
plot_to_notebook(t,samples,1000,out_signal=out_buffer)

# Free the buffers
in_buffer.close() //釋放記憶體空間
out_buffer.close()
```

# fir\_accel.ipynb

## Hardware FIR implementation

In the following code blocks, we test out the hardware FIR implementation and measure it's performance.

```
from pynq import Overlay
import pynq.lib.dma

# Load the overlay
overlay = Overlay('/home/xilinx/jupyter_notebooks/overlays_111511007/fir_accel.bit')

# Load the FIR DMA
dma = overlay.filter.fir_dma
```

← 將路徑修改成你的資料夾名稱

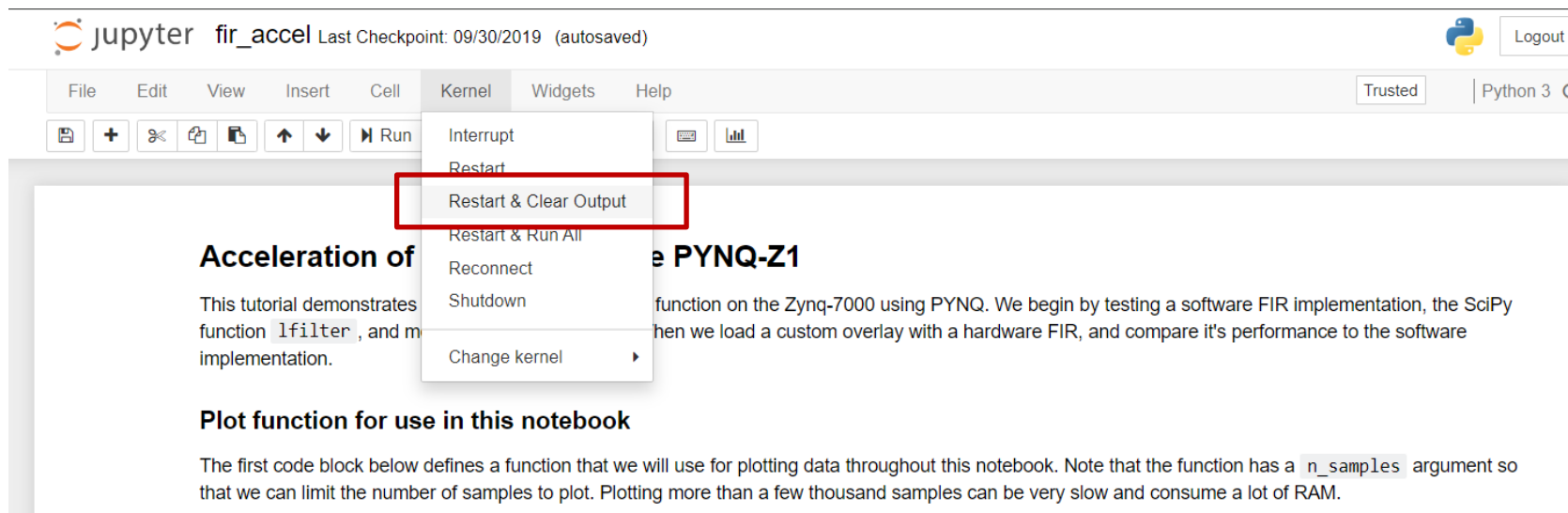
# fir\_accel.ipynb

---

- 依序跑過所有的code並截下Software FIR filter using SciPy和Hardware FIR implementation的執行結果(兩張截圖)
- 兩個都須包含執行的execution time和input/output的圖形

# 注意事項

- 在連線到jupyter notebook前請先登記時段，且不要在非登記時段使用
- 產生檔案後記得要上傳到jupyter notebook
- 跑完結果後請將ipynb結果清除，並刪除你上傳的overlays\_學號 資料夾





# Demo

- Demo需要上傳兩個檔案到E3：**vivado產生的bit檔**，以及包含**vivado畫面截圖**、**jupyter notebook兩張截圖**的report，命名規則如下
  - dcsxxx.bit
  - dcsxxx\_report.pdf
- Report可以只貼三張截圖就好，但請註明哪一張圖是對應哪個結果，沒註記會斟酌扣分
- Vivado 畫面截圖要求請參考p30，jupyter notebook截圖要求請參考p39，未達要求demo不通過
- 檔名錯誤扣5分，缺交任一檔案則不會通過demo
- **1de: 5/10 23:59**