Lab3
RISC on FPGA

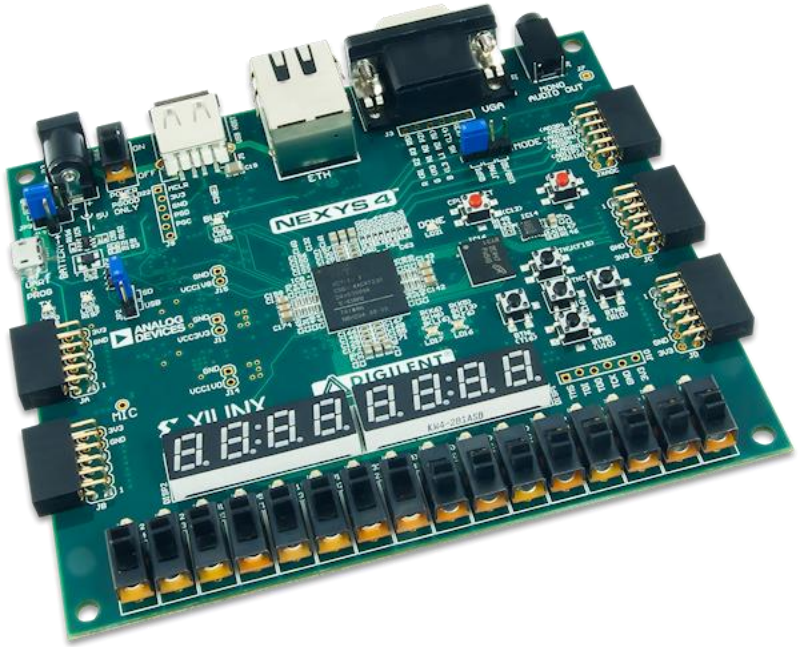
楊昕

實驗目的

- Check your design in lab2 is synthesizable
  - Gate-level simulation
- Port your design FPGA
  - Nexys 4 board DDR

實驗器材

- Nexys™4 Artix-7 FPGA Board



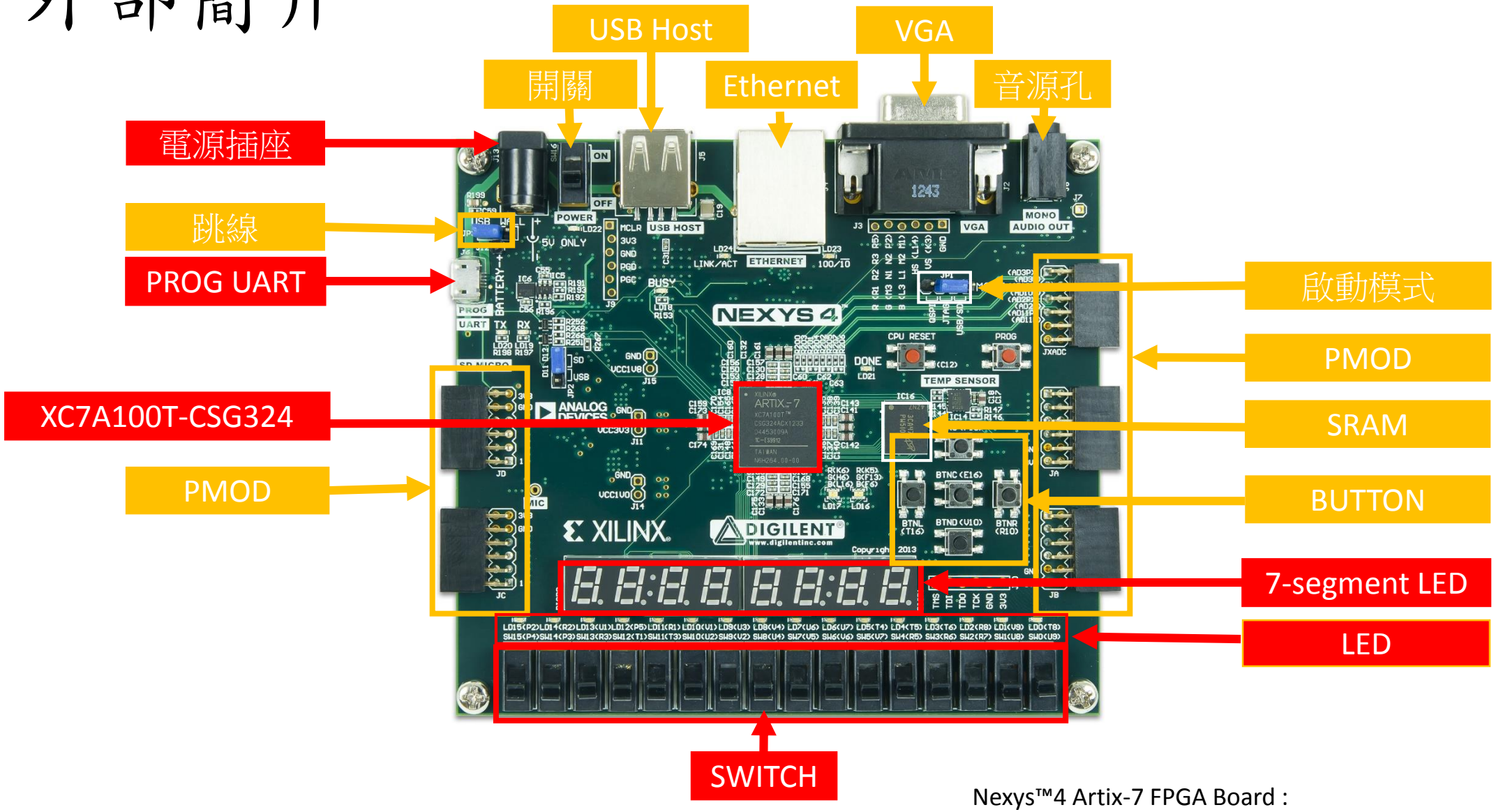
FPGA

- Field-programmable gate array，為可重複程式設計的晶片。

- 目前以硬體描述語言（Verilog或VHDL）描述的邏輯電路，可以利用logic synthesis和placement 、 routing 工具軟體，載至FPGA上進行測試。

FPGA 基本概念 **:**
http://www.ni.com/white-paper/6983/zht/

外部簡介



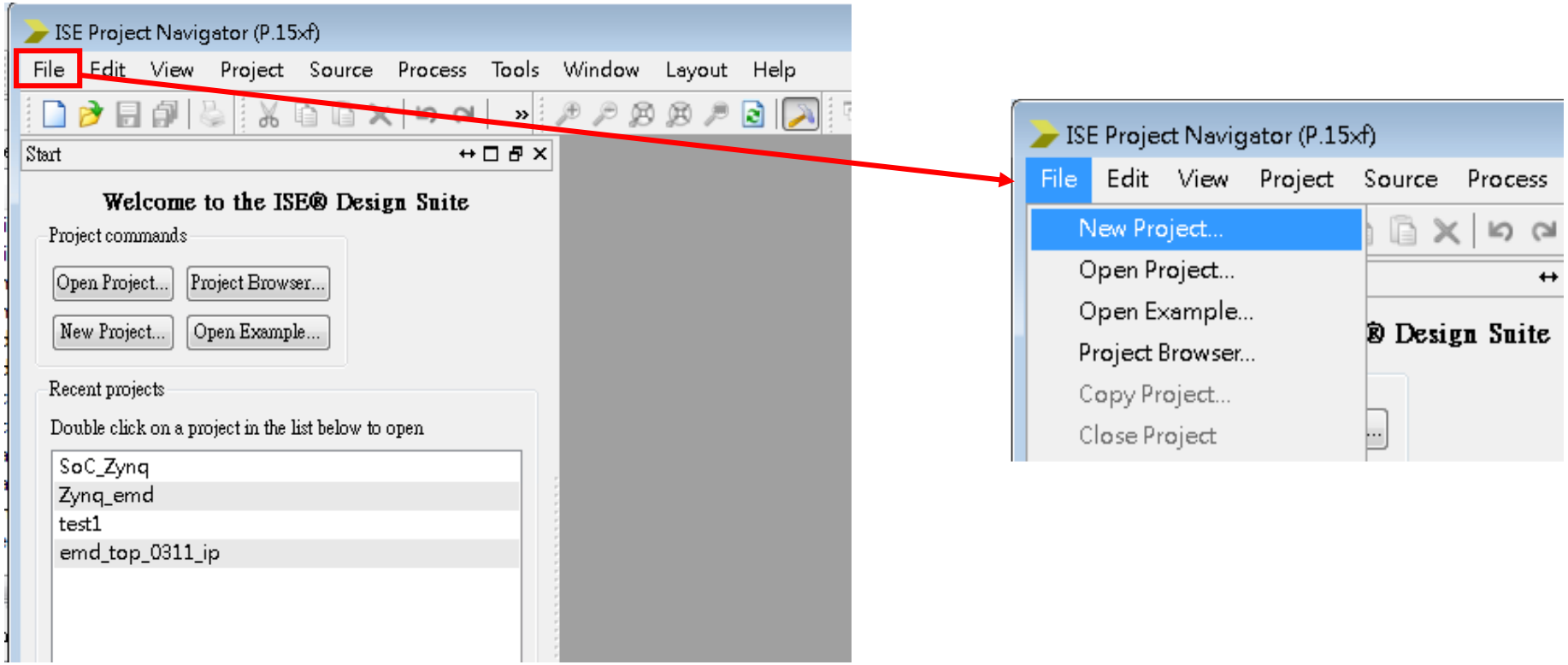
Nexys™4 Artix-7 FPGA Board :
http://www.xilinx.com/products/boards-and-kits/1-3yznp5.html

Integrated Synthesis Environment

- Xilinx ISE is a software tool produced by Xilinx for synthesis and analysis of HDL designs, enabling the developer to synthesize ("compile") designs, perform timing analysis, examine RTL diagrams, simulate a design's reaction to different stimuli, and configure the target device with the programmer.

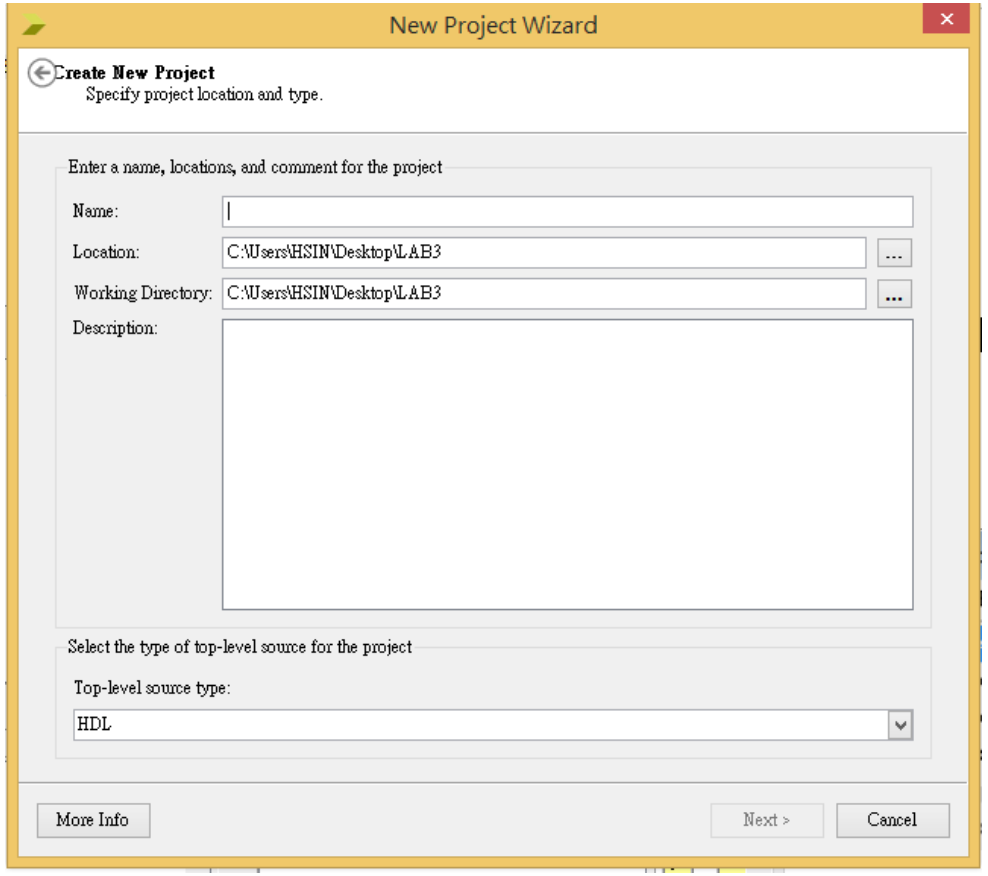
Create an ISE Project

- Xilinx Design Tools -> ISE Design Suite -> ISE Design Tools -> Project Navigator



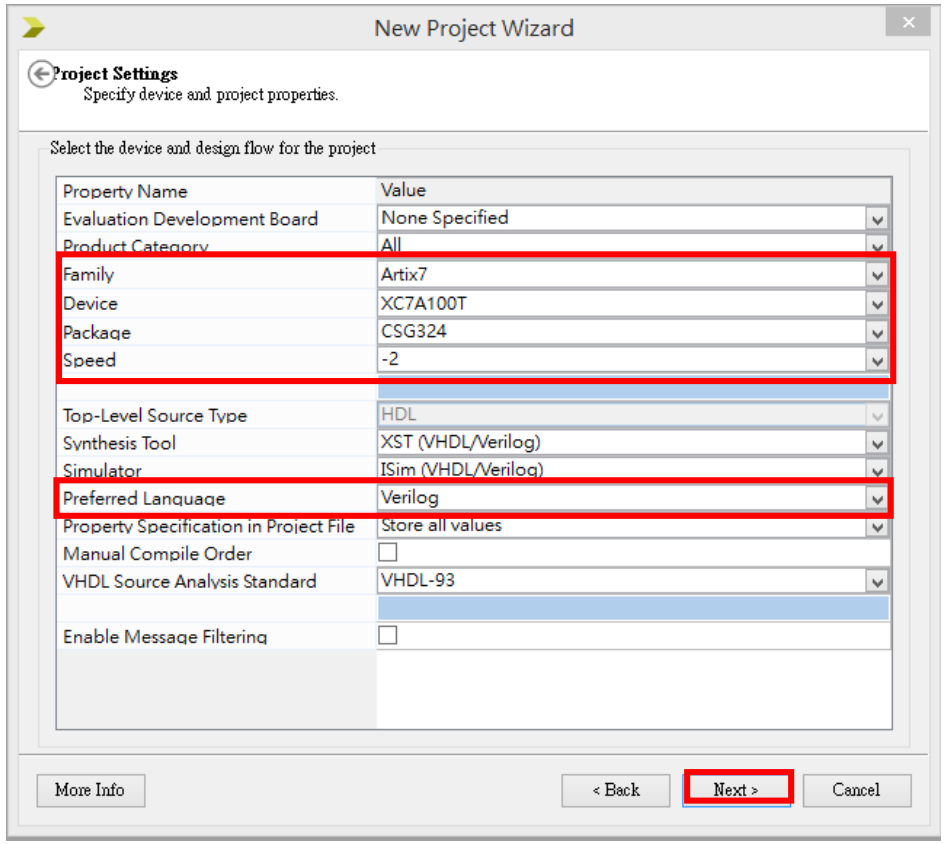
Create New Project

- Specify project location type



Set Device Environment

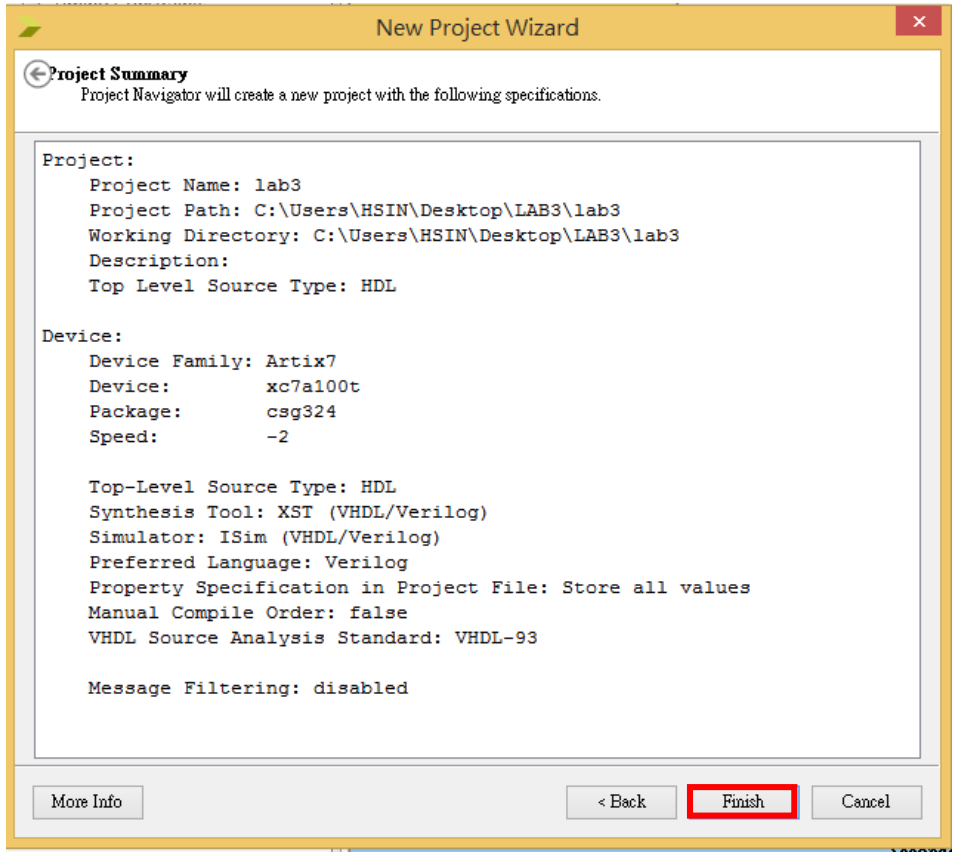
- Select your FPGA device correctly



FPGA Device :
Nexys™4 Artix-7 FPGA Board
XC7A100T-CSG324

Project Summary

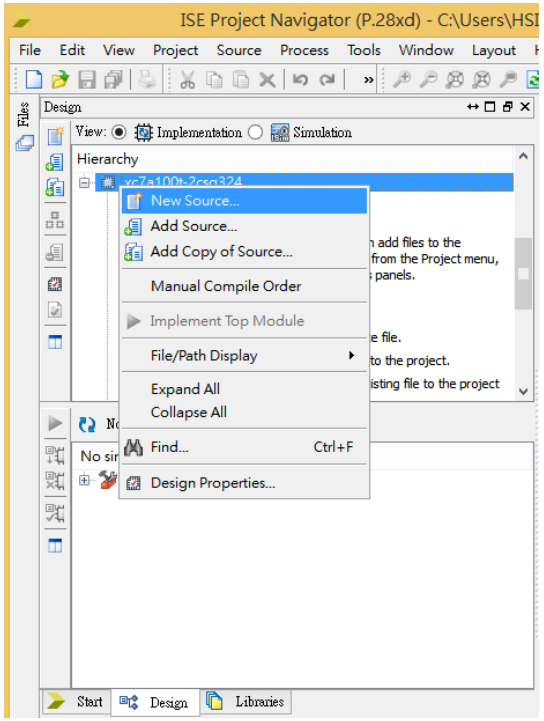
- Your can see all setting in this page, then press “Finish”.

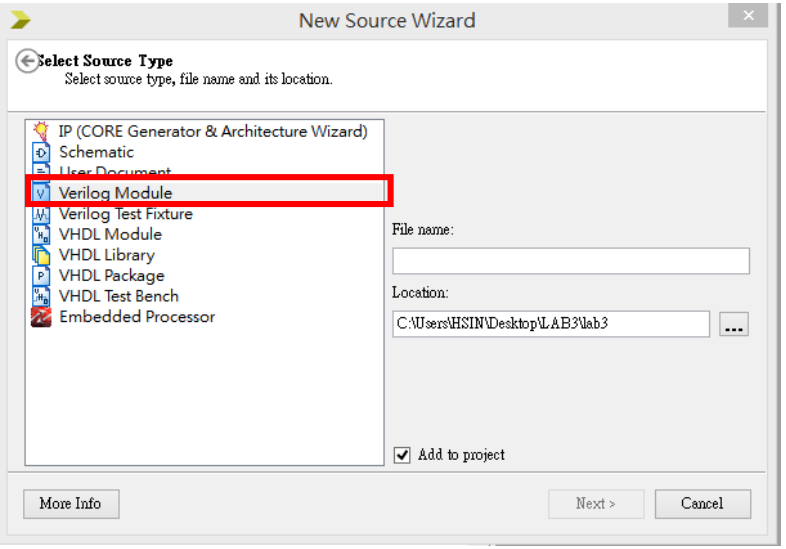


Design Entry(1/3)

You can choose “New Source” to create a space module or choose “Add Source” to add an exist module .

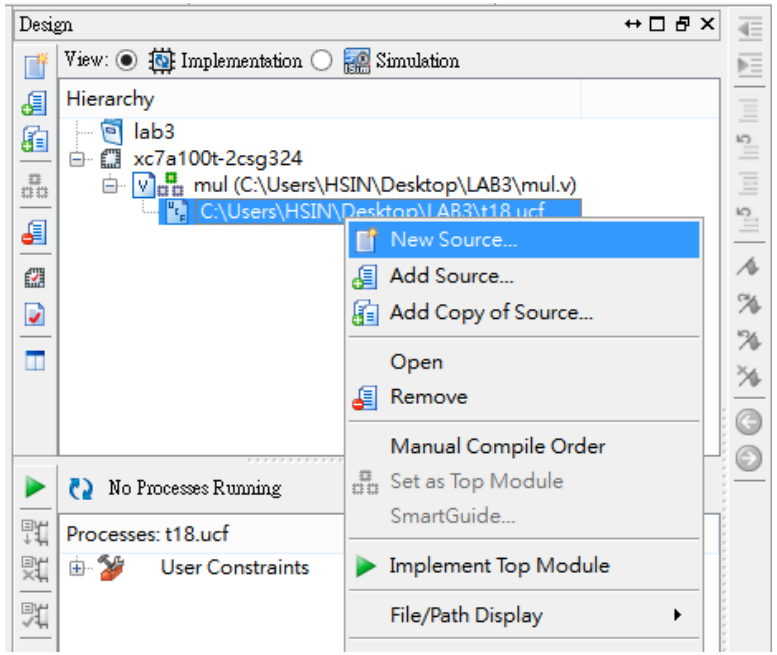
- Verilog code (.v) : 描述邏輯電路行為

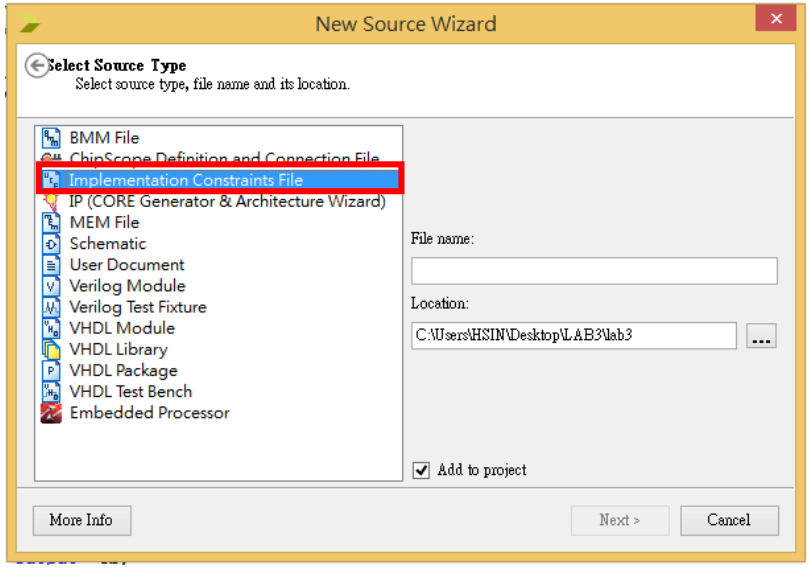




Design Entry(2/3)

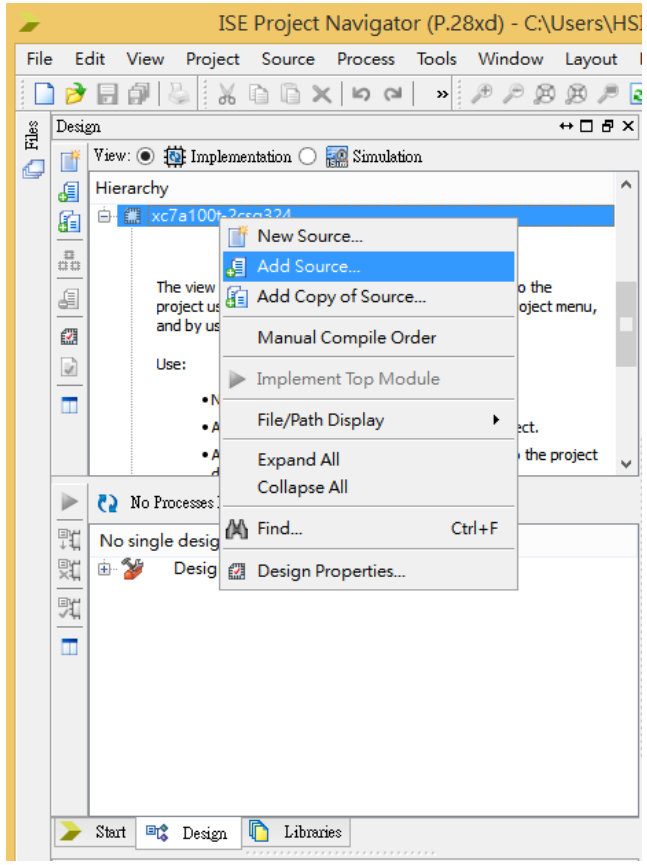
- User Constraints File (.ucf)：配置端口名稱和腳位編號

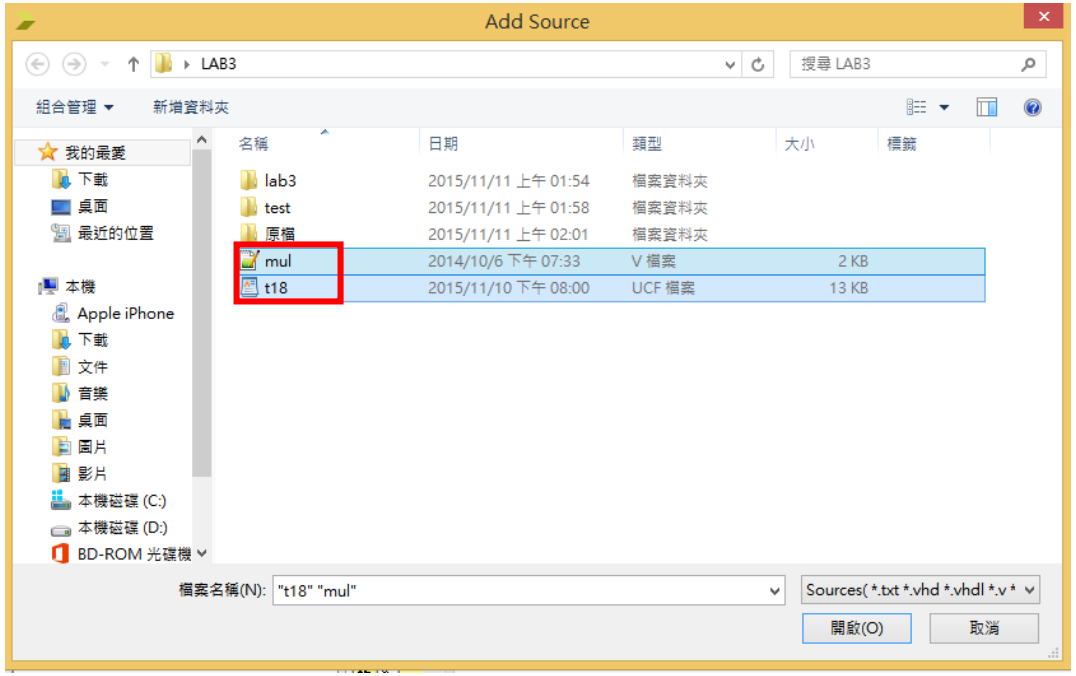




Design Entry(3/3)

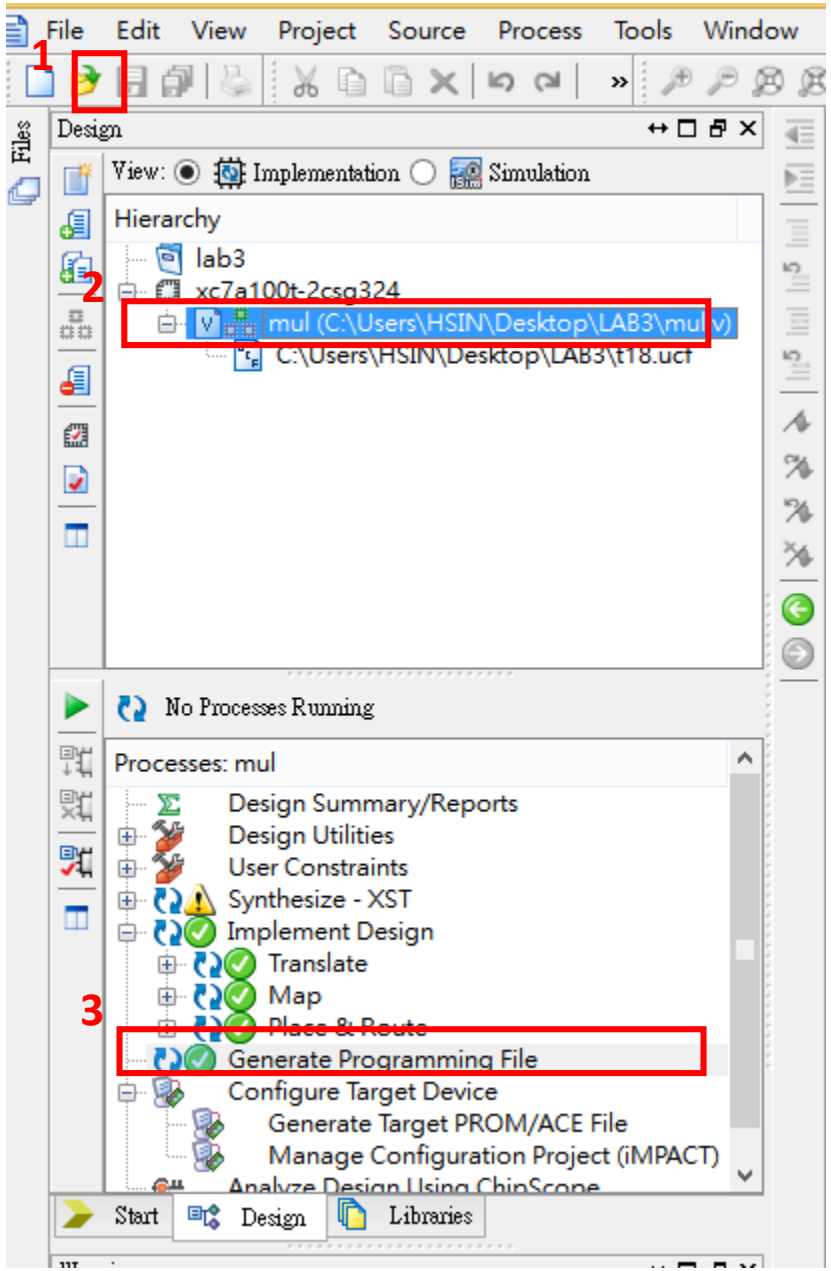
- 加入Lab3資料中的 .V 和.ucf 檔





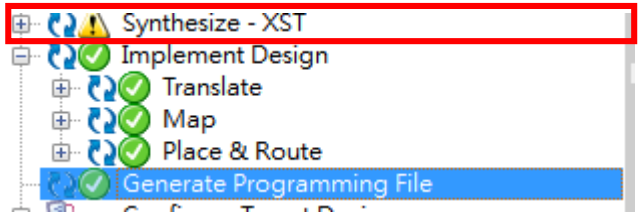
Generate Program File(1/3)

- step :
  1. Save all file
  2. Click top module
  3. Press “Generate Program File ”



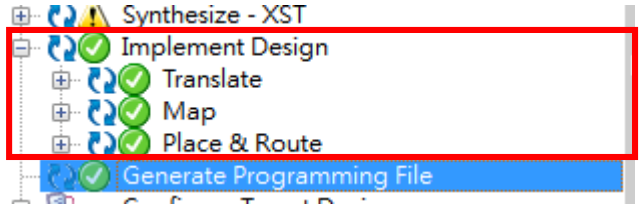
Generate Program File(2/3)

- Synthesize : 所設計數位電路經過布林函數化簡、優化後，轉換到的邏輯閘級別的電路連線網表的過程。



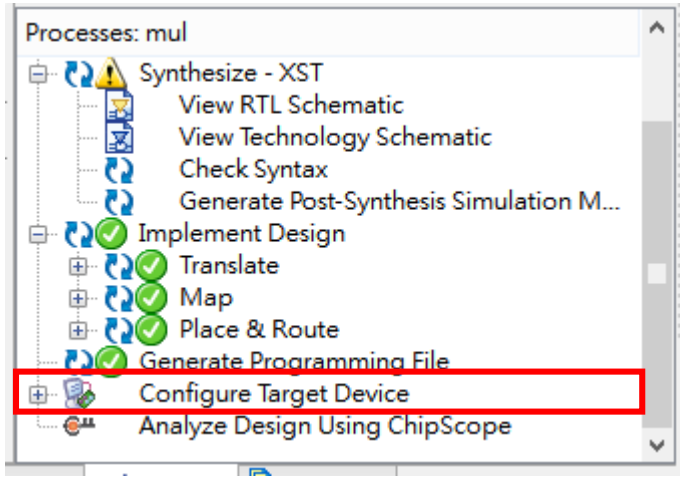
Generate Program File(3/3)

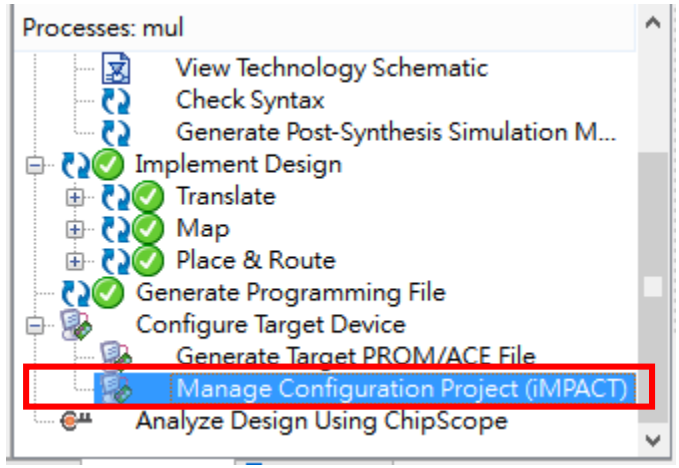
- Implement Design:
  - Translate : 將Synthesize輸出後的網表(Netlist Constraints File,ncf)整合，輸出到 Xilinx自帶的通用數據庫(Native Generic Database,NGD)
  - Map : 將design映射到device器件上(Flip-flop 、LUT)。
  - Place & Route : 根據.ucf和.pcf，進行實際的布局和布線。
- Generate Program File : 產生.bit



Download(1/4)

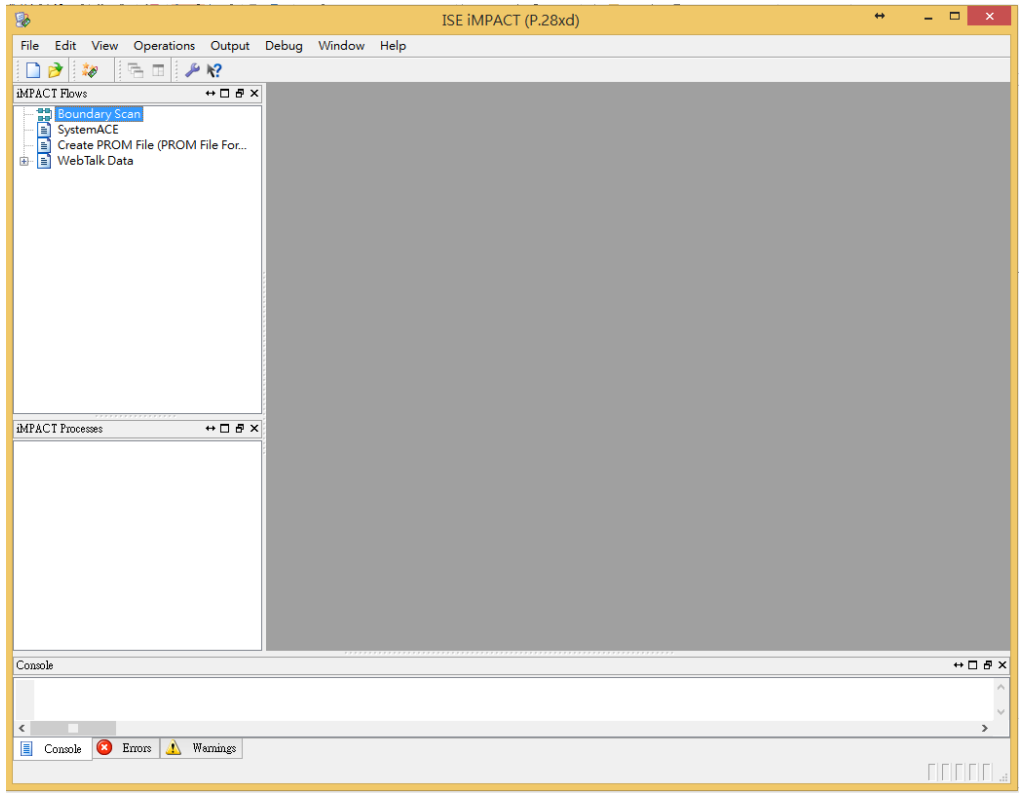
1. 點選Configure Target Device
2. 選擇第二個impact選項

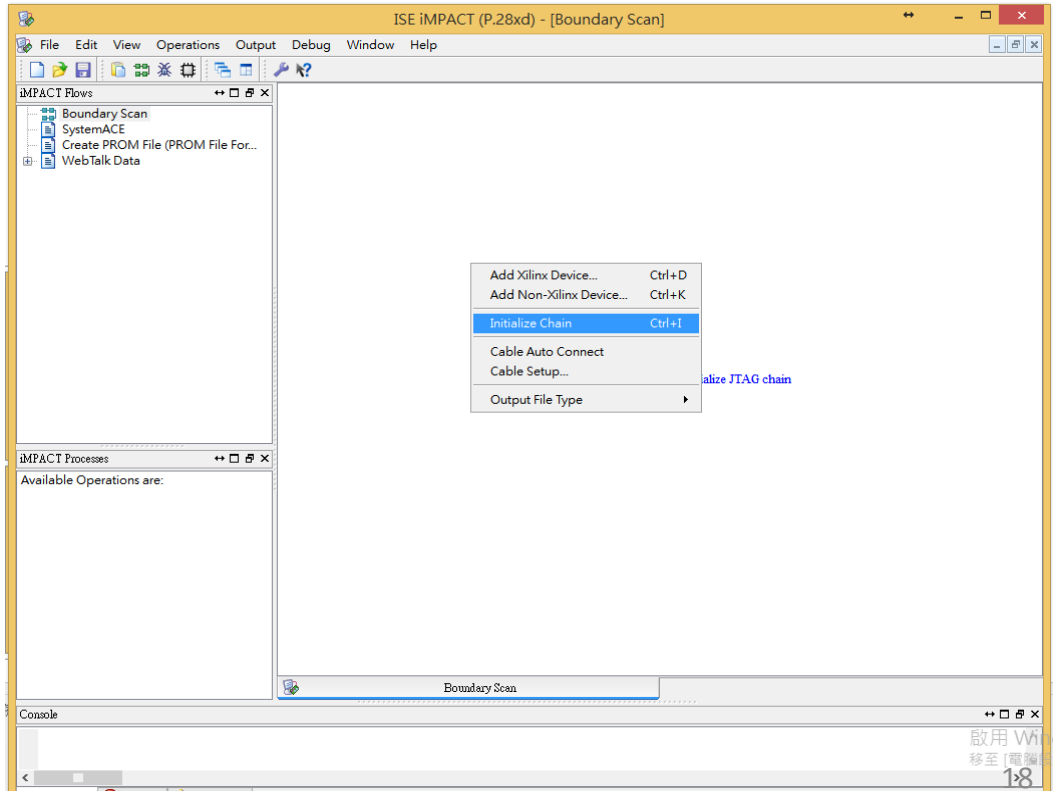




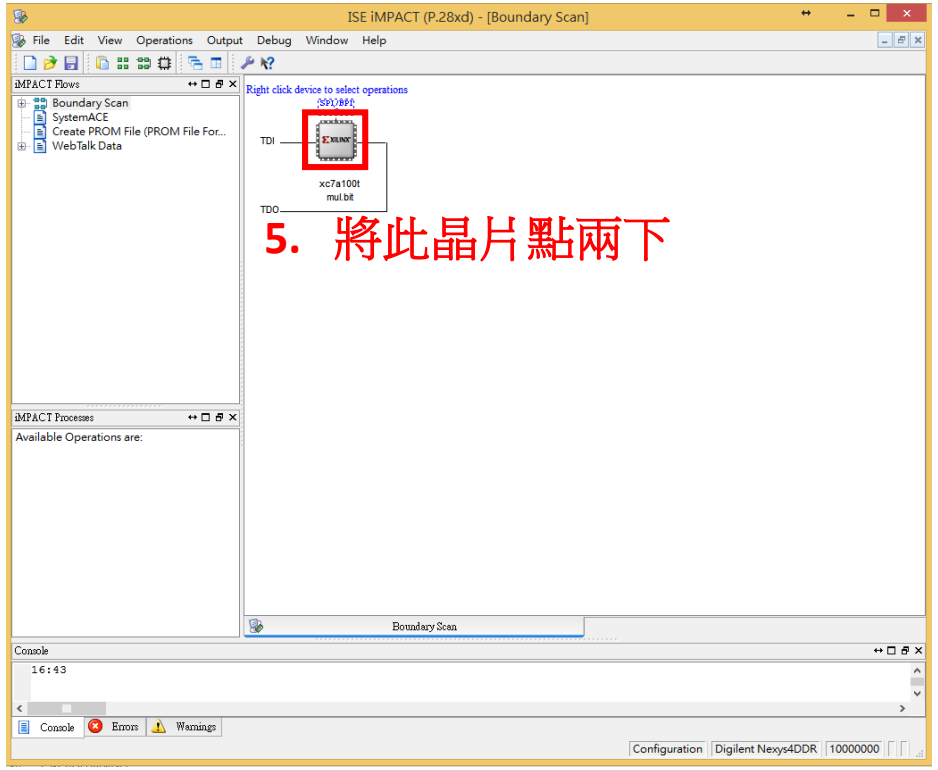
Download(2/4)

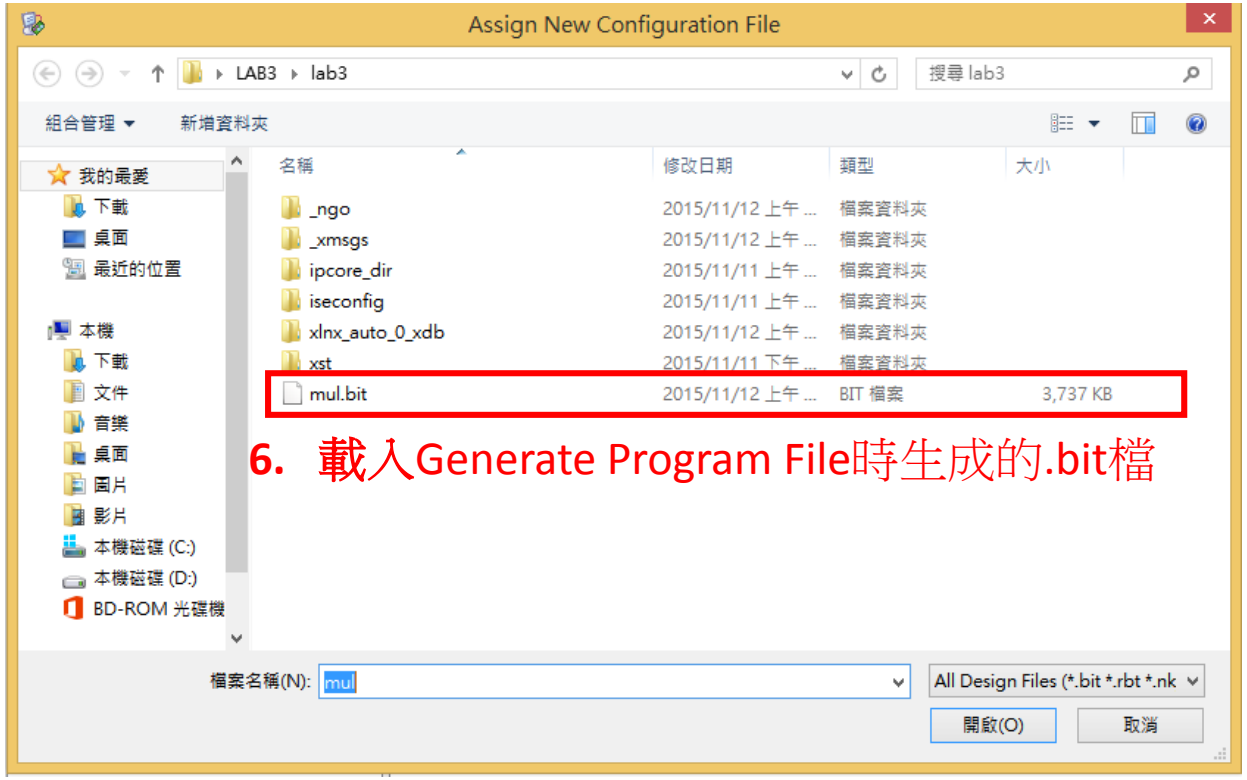
3. 點選 “Boundary Scan”
4. 在右側區塊空白處點擊右鍵，選擇 “Initialize Chain”



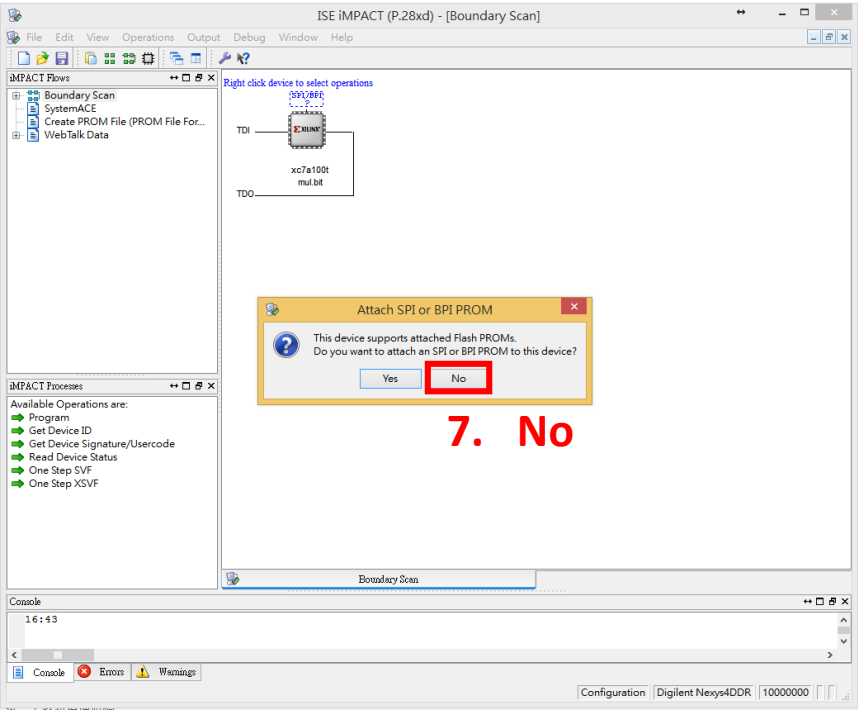


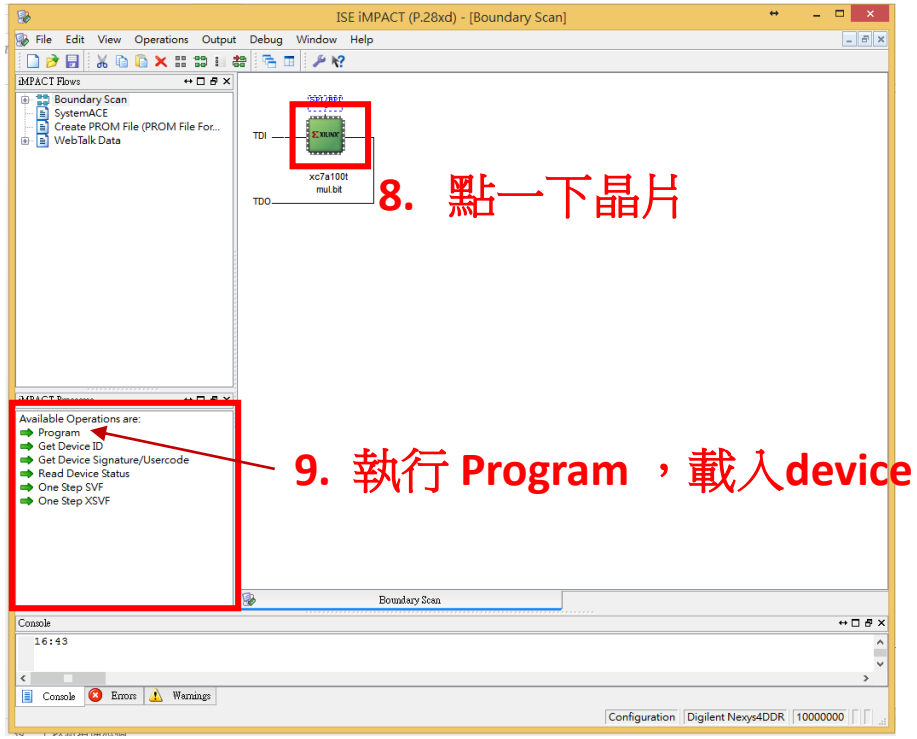
Download(3/4)


ISE iMPACT (P.28xd) - [Boundary Scan]
File Edit View Operations Output Debug Window Help
iMPACT Flows
Boundary Scan
SystemACE
Create PROM File (PROM File For...
WebTalk Data
Right click device to select operations
TDI
xc7a100t
mul.bit
TDO
5. 將此晶片點兩下
iMPACT Processes
Available Operations are:
Boundary Scan
Console
16:43
Console Errors Warnings
Configuration Digilent Nexys4DDR 10000000


Assign New Configuration File
LAB3 ▸ lab3
搜尋 lab3
組合管理
新增資料夾
我的最愛
下載
桌面
最近的位置
本機
下載
文件
音樂
桌面
圖片
影片
本機磁碟 (C:)
本機磁碟 (D:)
BD-ROM 光碟機
名稱
修改日期
類型
大小
_ngo 2015/11/12 上午 ... 檔案資料夾
_xmsgs 2015/11/12 上午 ... 檔案資料夾
ipcore_dir 2015/11/11 上午 ... 檔案資料夾
iseconfig 2015/11/11 上午 ... 檔案資料夾
xlnx_auto_0_xdb 2015/11/12 上午 ... 檔案資料夾
xst 2015/11/11 下午 ... 檔案資料夾
mul.bit 2015/11/12 上午 ... BIT 檔案 3,737 KB
6. 載入Generate Program File時生成的.bit檔
檔案名稱(N): mul
All Design Files (*.bit *.rbt *.nk
開啟(O)
取消

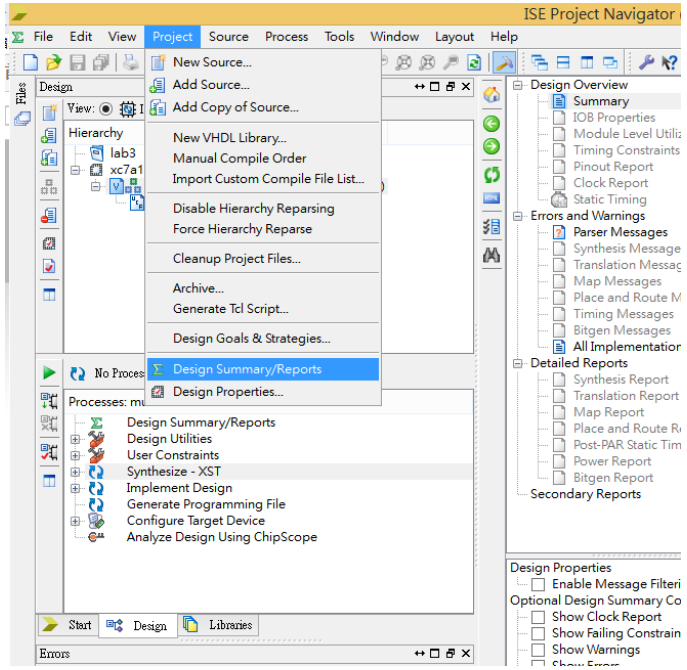
Download(4/4)


ISE iMPACT (P.28xd) - [Boundary Scan]
File Edit View Operations Output Debug Window Help
iMPACT Flows
Boundary Scan
SystemACE
Create PROM File (PROM File For...
WebTalk Data
Right click device to select operations
TDI
TDO
xc7a100t
mul.bit
Attach SPI or BPI PROM
This device supports attached Flash PROMs.
Do you want to attach an SPI or BPI PROM to this device?
Yes
No
7. No
iMPACT Processes
Available Operations are:
Program
Get Device ID
Get Device Signature/Usercode
Read Device Status
One Step SVF
One Step XSVF
Boundary Scan
Console
16:43
Console
Errors
Warnings
Configuration
Digilent Nexys4DDR
10000000


ISE iMPACT (P.28xd) - [Boundary Scan]
File Edit View Operations Output Debug Window Help
iMPACT Flows
Boundary Scan
SystemACE
Create PROM File (PROM File For...
WebTalk Data
TDI
TDO
xc7a100t
mul.bit
8. 點一下晶片
Available Operations are:
Program
Get Device ID
Get Device Signature/Usercode
Read Device Status
One Step SVF
One Step XSVF
9. 執行 Program ，載入device
Boundary Scan
Console
16:43
Console
Errors
Warnings
Configuration
Digilent Nexys4DDR
10000000

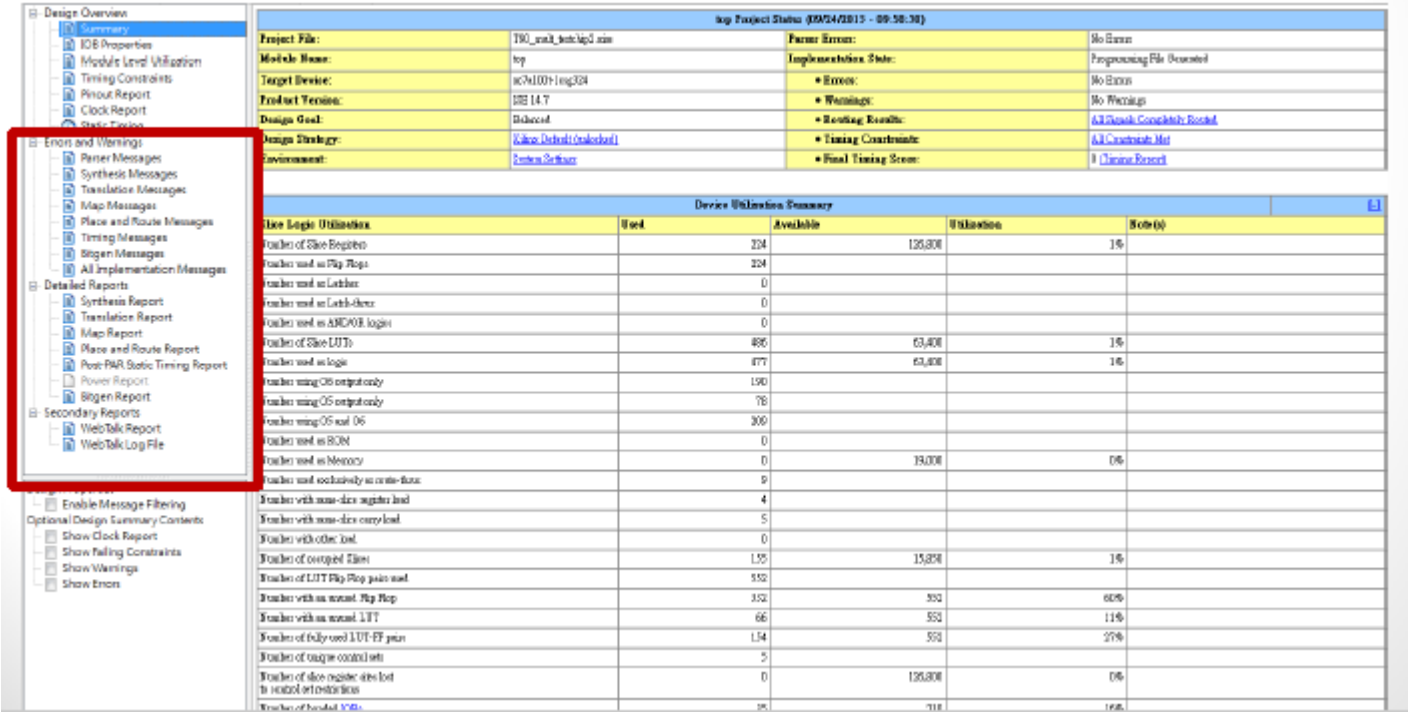
Error Messages & Detail Report(1/2)

- 當有錯誤發生時，可以點選Project -> Design Summary/Report ，會告知什麼環節出問題，方便 debug



Error Messages & Detail Report(2/2)

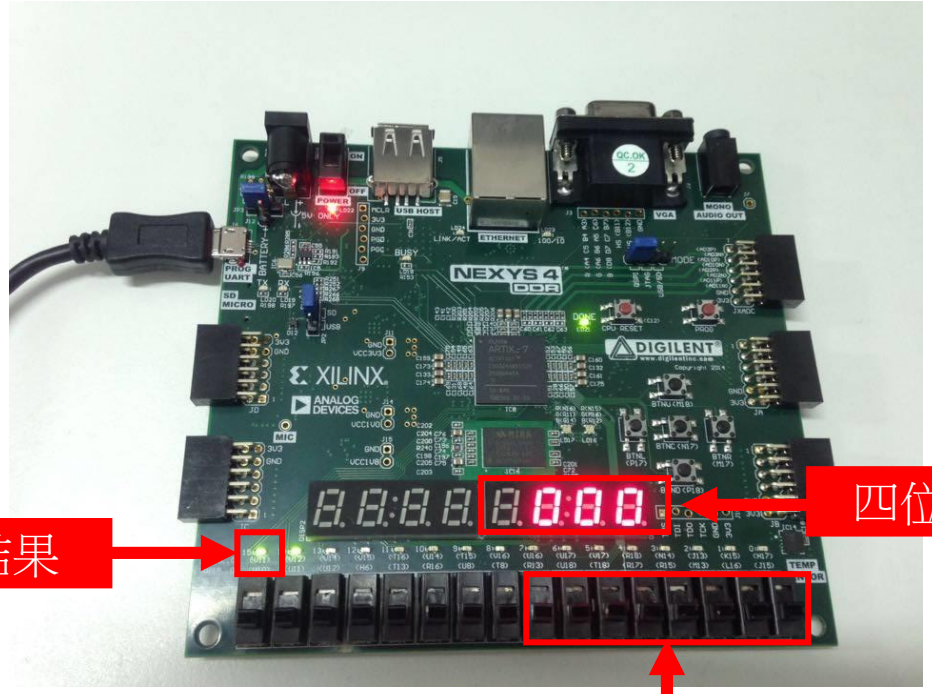
- Detail Report 會顯示整個設計的 critical path delay，並告知設計如何被合成



Lab3

- 將LAB 2.1的Serial Multiplier 實現在FPGA板上
  - LED_15 : 是否完成計算 (1: 顯示計算　0: 正在計算)
  - LED _0~7 : 二進制顯示Serial Multiplier 運算過程
  - DISP 1 : 四位七段顯示顯示結果
  - Swich_0~3 : 被乘數
  - Swich_4~7 : 乘數

Lab3

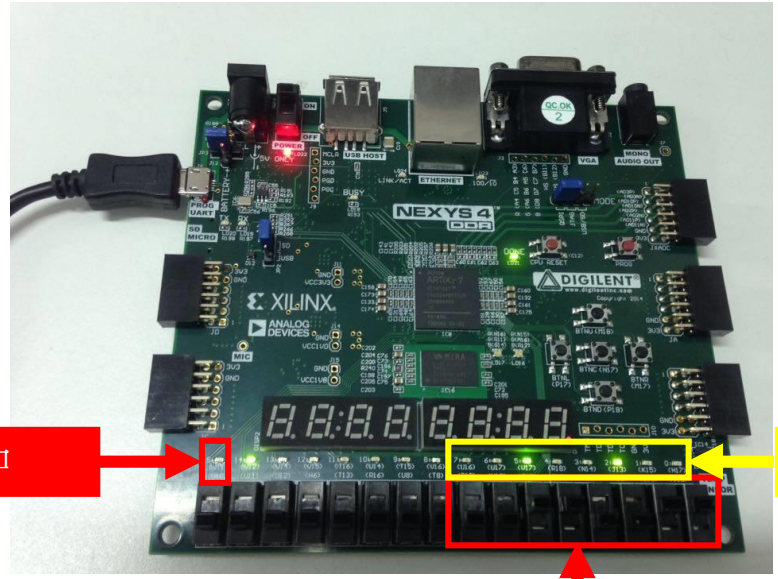


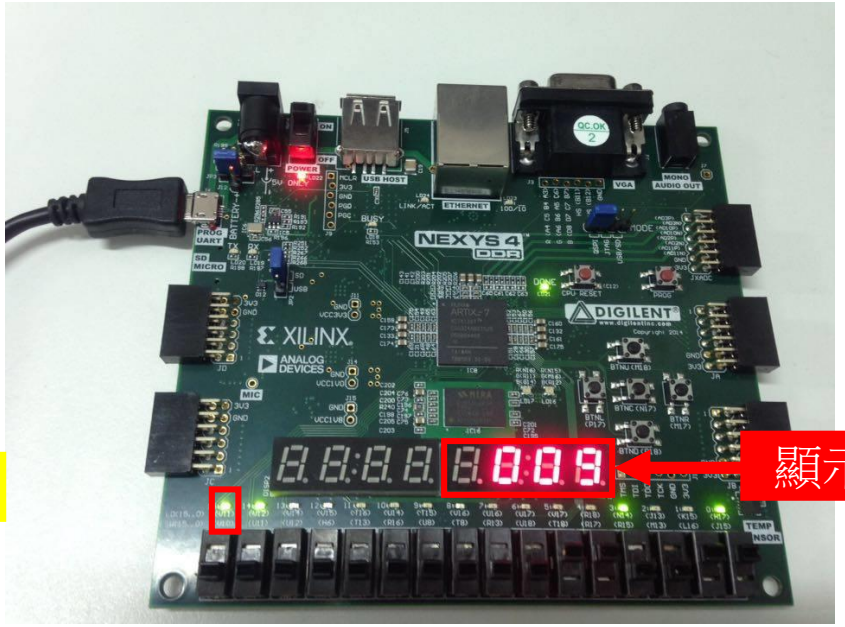
四位七段顯示器: 顯示結果

LED_15: 運算結束，顯示結果

sw_0~7 ?: 0*0=0

Lab3





User Constraints File (UCF)

- User Constraints File (UCF)
  - 使用者可進行編輯端口名稱和腳位編號
  - NET “變數名稱” LOC = “內部PIN腳號碼”

```
## Switches

NET "a1"          LOC=J15 | IOSTANDARD=LVCMOS33; #IO_L24N_T3_RS0_15
NET "a2"          LOC=L16 | IOSTANDARD=LVCMOS33; #IO_L3N_T0_DQS_EMCCLK_14
NET "a3"          LOC=M13 | IOSTANDARD=LVCMOS33; #IO_L6N_T0_D08_VREF_14
NET "a4"          LOC=R15 | IOSTANDARD=LVCMOS33; #IO_L13N_T2_MRCC_14
NET "b1"          LOC=R17 | IOSTANDARD=LVCMOS33; #IO_L12N_T1_MRCC_14
NET "b2"          LOC=T18 | IOSTANDARD=LVCMOS33; #IO_L7N_T1_D10_14
NET "b3"          LOC=U18 | IOSTANDARD=LVCMOS33; #IO_L17N_T2_A13_D29_14
NET "b4"          LOC=R13 | IOSTANDARD=LVCMOS33; #IO_L5N_T0_D07_14

## LEDs
NET "o1"         LOC=H17 | IOSTANDARD=LVCMOS33; #IO_L18P_T2_A24_15
NET "o2"         LOC=K15 | IOSTANDARD=LVCMOS33; #IO_L24P_T3_RS1_15
NET "o3"         LOC=J13 | IOSTANDARD=LVCMOS33; #IO_L17N_T2_A25_15
NET "o4"         LOC=N14 | IOSTANDARD=LVCMOS33; #IO_L8P_T1_D11_14
NET "o5"         LOC=R18 | IOSTANDARD=LVCMOS33; #IO_L7P_T1_D09_14
NET "o6"         LOC=V17 | IOSTANDARD=LVCMOS33; #IO_L18N_T2_A11_D27_14
NET "o7"         LOC=U17 | IOSTANDARD=LVCMOS33; #IO_L17P_T2_A14_D30_14
NET "o8"         LOC=U16 | IOSTANDARD=LVCMOS33; #IO_L18P_T2_A12_D28_14
```

```
## 7 segment display
NET "ca"             LOC=T10 | IOSTANDARD=LVCMOS33; #IO_L24N_T3_A00_D16_14
NET "cb"             LOC=R10 | IOSTANDARD=LVCMOS33; #IO_25_14
NET "cc"             LOC=K16 | IOSTANDARD=LVCMOS33; #IO_25_15
NET "cd"             LOC=K13 | IOSTANDARD=LVCMOS33; #IO_L17P_T2_A26_15
NET "ce"             LOC=P15 | IOSTANDARD=LVCMOS33; #IO_L13P_T2_MRCC_14
NET "cf"             LOC=T11 | IOSTANDARD=LVCMOS33; #IO_L19P_T3_A10_D26_14
NET "cg"             LOC=L18 | IOSTANDARD=LVCMOS33; #IO_L4P_T0_D04_14
#NET "dp"             LOC=H15 | IOSTANDARD=LVCMOS33; #IO_L19N_T3_A21_VREF_15

NET "an0"          LOC=J17 | IOSTANDARD=LVCMOS33; #IO_L23P_T3_FOE_B_15
NET "an1"          LOC=J18 | IOSTANDARD=LVCMOS33; #IO_L23N_T3_FWE_B_15
NET "an2"          LOC=T9 | IOSTANDARD=LVCMOS33; #IO_L24P_T3_A01_D17_14
NET "an3"          LOC=J14 | IOSTANDARD=LVCMOS33; #IO_L19P_T3_A22_15
NET "an4"          LOC=P14 | IOSTANDARD=LVCMOS33; #IO_L8N_T1_D12_14
NET "an5"          LOC=T14 | IOSTANDARD=LVCMOS33; #IO_L14P_T2_SRCC_14
NET "an6"          LOC=K2 | IOSTANDARD=LVCMOS33; #IO_L23P_T3_35
NET "an7"          LOC=U13 | IOSTANDARD=LVCMOS33; #IO_L23N_T3_A02_D18_14
```

User Constraints File (UCF)

- .v檔內的module i/o 變數與.ucf內腳位變數相對應

```
module mul(
        input clk100mhz,
        input rst,
        input a1,
        input a2,
        input a3,
        input a4,
        input b1,
        input b2,
        input b3,
        input b4,

        output o1,
        output o2,
        output o3,
        output o4,
        output o5,
        output o6,
        output o7,
        output o8,
```

```
## Switches

NET "a1"          LOC=J15 | IOSTANDARD=LVCMOS33; #IO_L24N_T3_RS0_15
NET "a2"          LOC=L16 | IOSTANDARD=LVCMOS33; #IO_L3N_T0_DQS_EMCCLK_14
NET "a3"          LOC=M13 | IOSTANDARD=LVCMOS33; #IO_L6N_T0_D08_VREF_14
NET "a4"          LOC=R15 | IOSTANDARD=LVCMOS33; #IO_L13N_T2_MRCC_14
NET "b1"          LOC=R17 | IOSTANDARD=LVCMOS33; #IO_L12N_T1_MRCC_14
NET "b2"          LOC=T18 | IOSTANDARD=LVCMOS33; #IO_L7N_T1_D10_14
NET "b3"          LOC=U18 | IOSTANDARD=LVCMOS33; #IO_L17N_T2_A13_D29_14
NET "b4"          LOC=R13 | IOSTANDARD=LVCMOS33; #IO_L5N_T0_D07_14

## LEDs
NET "o1"         LOC=H17 | IOSTANDARD=LVCMOS33; #IO_L18P_T2_A24_15
NET "o2"         LOC=K15 | IOSTANDARD=LVCMOS33; #IO_L24P_T3_RS1_15
NET "o3"         LOC=J13 | IOSTANDARD=LVCMOS33; #IO_L17N_T2_A25_15
NET "o4"         LOC=N14 | IOSTANDARD=LVCMOS33; #IO_L8P_T1_D11_14
NET "o5"         LOC=R18 | IOSTANDARD=LVCMOS33; #IO_L7P_T1_D09_14
NET "o6"         LOC=V17 | IOSTANDARD=LVCMOS33; #IO_L18N_T2_A11_D27_14
NET "o7"         LOC=U17 | IOSTANDARD=LVCMOS33; #IO_L17P_T2_A14_D30_14
NET "o8"         LOC=U16 | IOSTANDARD=LVCMOS33; #IO_L18P_T2_A12_D28_14
```

Clock / 除頻

- 使用 100mhz

```
## Clock signal
NET "clk100mhz"   LOC = "E3"  | IOSTANDARD = "LVCMOS33";
#Bank = 35, Pin name = #IO_L12P_T1_MRCC_35,Sch name = clk100mhz
NET "clk100mhz" TNM_NET = sys_clk_pin;
#TIMESPEC TS_sys_clk_pin = PERIOD sys_clk_pin 100 MHz HIGH 50%;
```

- 若運算過程不經除頻，一秒運算100萬次，肉眼會無法辨識。

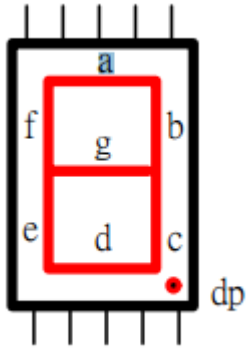
```
always@(posedge clk100mhz)
begin
   if(rst)
      count_div <= 32'd0;
   else if (count_div == 32'd50000000)
      count_div <= 32'd0;
   else
      count_div <= count_div + 1;
end

always@(posedge clk100mhz)
begin
      if(rst)begin
         count <= 5'd0;
      end else if(count_div == 32'd50000000) begin
         if(count < 5'd16)
            count <= count + 5'd1;
         else
            count <= 5'd0;
      end else begin
         count <= count;
      end

end
```

七段顯示器

- 為共陰七段顯示器，給低電位會亮。

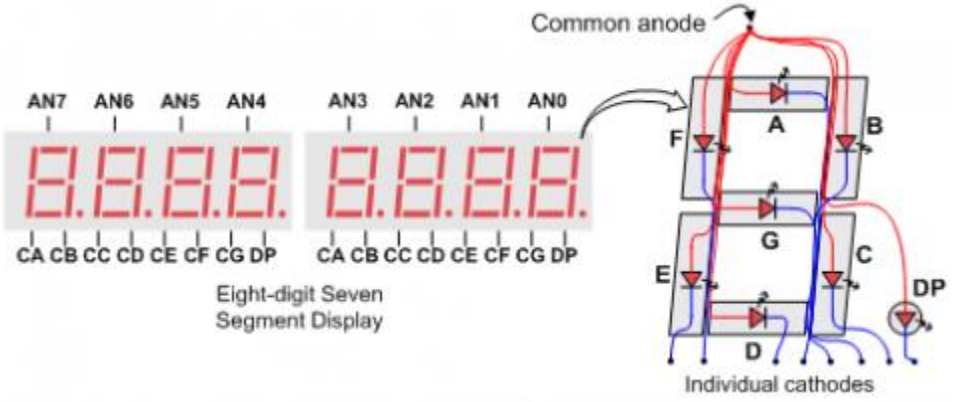
```
initial
begin
   seg[0] = 7'b1000000;//h40;//8'hc0;
   seg[1] = 7'b1111001;//h79;//8'hf9;
   seg[2] = 7'b0100100;//h24;//8'ha4;
   seg[3] = 7'b0110000;//h30;//8'hb0;
   seg[4] = 7'b0011001;//h19;//8'h99;
   seg[5] = 7'b0010010;//h12;//8'h92;
   seg[6] = 7'b0000010;//h02;//8'h82;
   seg[7] = 7'b1111000;//h78;//8'hf8;
   seg[8] = 7'b0000000;//h00;//8'h80;
   seg[9] = 7'b0010000;//h10;//8'h90;
end
```

無小數點



四位七段顯示器

- 原理：人類因「視覺暫留」在看過東西之後，約有 40mSec 的影像殘存，因此，只要影像消失不超過 40mSec，則眼睛就無法察覺。所以會感覺看 到一個持續存在的影像。
- 但若每個位數之間的間隔過短，人眼辨識速度過慢，會看到重疊的數字。(AN3~AN0之間位移過快)
- 使用除頻，讓數字連串並辨識清晰。



圖片來源：https://reference.digilentinc.com/nexys:nexys4:refmanual

HW3 & 作業上傳格式

- 將RISC載入到Nexys 4 ，實作GCD(a,b)
  - 實作JAL、JR指令，並利用這兩道指令實作遞迴完成GCD (利用範例實作，增加JAL、JR)
  - Input : sw_0~7 為a、 sw_8~15 為b
  - Output : 使用十進制利用四位七段顯示器顯示答案
  - 將input存入DM(data memory)中
  - 將input和結果 轉成十進位顯示
- 繳交：
  - 整個專案
  - Due:11/25(三) 11:59 p.m.
  - 請於12/1(二)、12/3(四)開放時段來Demo。
  - 下載e-course 11/25當天繳交作業檔案Demo。

	12/1	12/3
10-12	◎	◎
13-17		
20-22	◎	◎

評分標準

- Lab 3 共佔總成績的10%
- 課堂實作2分
- 回家作業8分
- 繳交期限：11/25 23:59
- 評分標準：
  1. 功能正常：
     - A->B-> answer : 5 分
  2. 計算Slice Logic Utilization :
     - 前5名得3分；6-10名得2分；11-20名得1分

Device Utilization Summary				
Slice Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Registers	227	126,800	1%	
Number used as Flip Flops	227			
Number used as Latches	0			
Number used as Latch-thrus	0			
Number used as AND/OR logics	0			
Number of Slice LUTs	288	63,400	1%	
Number used as logic	263	63,400	1%	
Number using O6 output only	176			
Number using O5 output only	28			
Number using O5 and O6	59			
Number used as ROM	0			
Number used as Memory	20	19,000	1%	
Number used as Dual Port RAM	0			
Number used as Single Port RAM	0			
Number used as Shift Register	20			
Number using O6 output only	3			
Number using O5 output only	0			
Number using O5 and O6	17			
Number used exclusively as route-thrus	5			
Number with same-slice register load	3			
Number with same-slice carry load	2			

JAL & JR

JAL -- ***Jump and link***

Description:	Jumps to the calculated address and stores the return address in $31
Operation:	$31 = PC + 8 (or nPC + 4); PC = nPC; nPC = (PC & 0xf0000000) | (target << 2);
Syntax:	jal target
Encoding:	`0000 11ii iiii iiii iiii iiii iiii iiii`

JR -- ***Jump register***

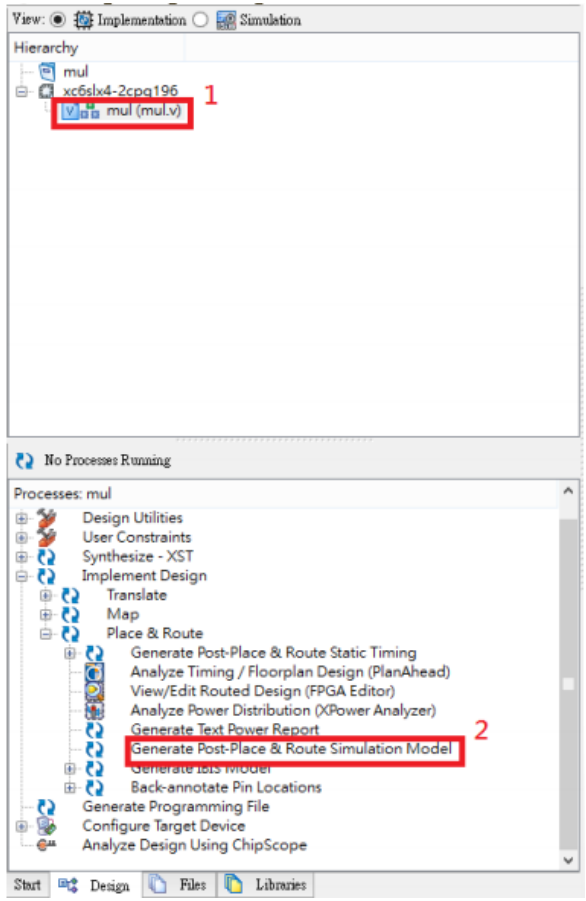
Description:	Jump to the address contained in register $s
Operation:	PC = nPC; nPC = $s;
Syntax:	jr $s
Encoding:	`0000 00ss sss0 0000 0000 0000 0000 1000`

Gate level simulation(補充)

- 在將程式放上 FPGA 前，可利 用 gate level simulation 模擬最後結果

- Synthesize step

1. Click .v file
2. Press to synthesize



Gate level simulation(補充)

- Select the file
  - Your project destination -> netgen -> par
  - Select mul_timesim.sdf and mul_timesim.v

- Select the library
  - C:\-> Xilinx -> 14.1 -> ISE_DS -> ISE -> verilog -> src
  - Select simprims and XilinxCoreLib

Gate level simulation(補充)

- 步驟一
  - Compile command
    - ncverilog +libext+.v +access+r -y /tmp/simprims -y /tmp/XilinxCoreLib mul_timesim.v tb.v

```
birdymans@raichu[6:01pm]~/aagain>ls
in  mul_timesim.sdf  mul_timesim.v  out  simprims  tb.v  XilinxCoreLib
birdymans@raichu[6:01pm]~/aagain>ncverilog +libext+.v +access+r -y simprims -y XilinxCoreLib mul_timesim.v tb.v
```

Gate level simulation(補充)

- Compiler done

```
        Annotation completed successfully...
        Building instance overlay tables: ..................... Done
        Generating native compiled code:
                simprims.X_DSP48A1:v <0x3bf2ac38>
                        streams: 106, words: 59433
                simprims.X_OBUF:v <0x1c64fbf8>
                        streams:   1, words:   109
                simprims.X_OBUF:v <0x342194a4>
                        streams:   1, words:   109
                simprims.X_ONE:v <0x404b8400>
                        streams:   0, words:     0
                simprims.X_ZERO:v <0x1b2cf928>
                        streams:   0, words:     0
                worklib.glbl:v <0x202a995e>
                        streams:  11, words:  2627
                worklib.mul:v <0x45accb6f>
                        streams:   1, words:   228
                worklib.tb:v <0x5b101c8b>
                        streams:   3, words:  7730
        Loading native compiled code:     ..................... Done
        Building instance specific data structures.
        Design hierarchy summary:
                                  Instances  Unique
                Modules:                215      10
                Primitives:             170       5
                Timing outputs:         297     159
                Registers:               66      66
                Scalar wires:           352       -
                Expanded wires:         182       6
                Vectored wires:           1       -
                Always blocks:           34      35
                Initial blocks:          21      21
                Cont. assignments:       35      10
                Pseudo assignments:       5       5
                Timing checks:          795       -
                Interconnect:            68      32
                Delayed tcheck signals:  194     119
                Simulation timescale:    1ps
        Writing initial simulation snapshot: worklib.glbl:v
Loading snapshot worklib.glbl:v ..................... Done
*Verdi3* Loading libsscore_ius111.so
*Verdi3* : Enable Parallel Dumping.
ncsim> source /usr/cad/cadence/INCISIV/cur/tools/inca/files/ncsimrc
ncsim> run
Simulation complete via $finish(1) at time 1980039800 PS + 0
./tb.v:89               $finish;
ncsim> exit
```

Gate level simulation(補充)

- Simulation

```
                                  Instances  Unique
                Modules:                175      14
                UDPs:                    24       2
                Primitives:             558       5
                Timing outputs:         135      12
                Registers:              284      48
                Scalar wires:           490       -
                Always blocks:           81       7
                Initial blocks:          37      15
                Cont. assignments:      178      23
                Pseudo assignments:       8       8
                Timing checks:          440     126
                Interconnect:           257      82
                Delayed tcheck signals: 128      64
                Simulation timescale:   1ps
        Writing initial simulation snapshot: worklib.glbl:v
Loading snapshot worklib.glbl:v ..................... Done
ncsim> source /cad/cadence/INCISIV/cur/tools/inca/files/ncsimrc
ncsim> run
  1 *  8 =    8,your ans=   8
  3 *  6 =   18,your ans=  18
  5 *  4 =   20,your ans=  20
  7 *  2 =   14,your ans=  14
  9 *  0 =    0,your ans=   0
Right=          5 ,Wrong=          0
Simulation complete via $finish(1) at time 1000230 NS + 0
./tb.v:44     #30 $finish;
ncsim> exit
yang@Euler-ES:~/Verilog/CO/LAB3/simulation$
```