

Lab3: Push Buttons and LED Control



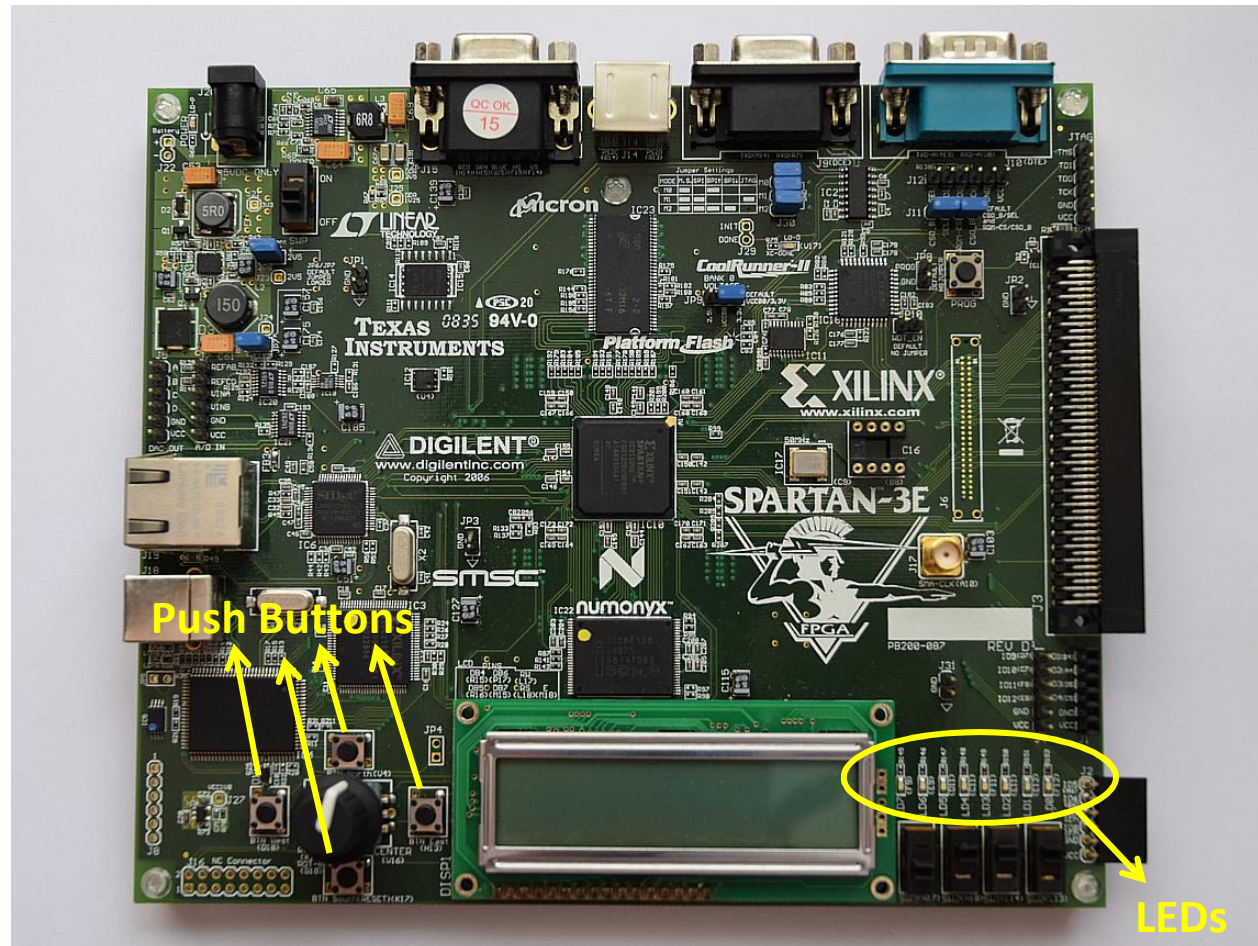
National Chiao Tung University
Chun-Jen Tsai
10/10/2015

Lab 3: Push Buttons and LED Control

- ❑ In this lab, you will use the Spartan-3E board to implement a simple I/O control circuit
 - There are 5 push-buttons and 8 LED lights on the board
 - You must design a synchronous circuit that reads the push-button inputs and use the inputs to increase/decrease the value of a 4-bit counter
 - You must display the counter value using the 8 LEDs in sign-extended 2's complement binary format at any time
- ❑ You will demo the design to your TA during the lab hours on 10/19

Buttons and LEDs on the Board

- ❑ The Spartan-3E Starter Kit



The System Behavior of Lab3

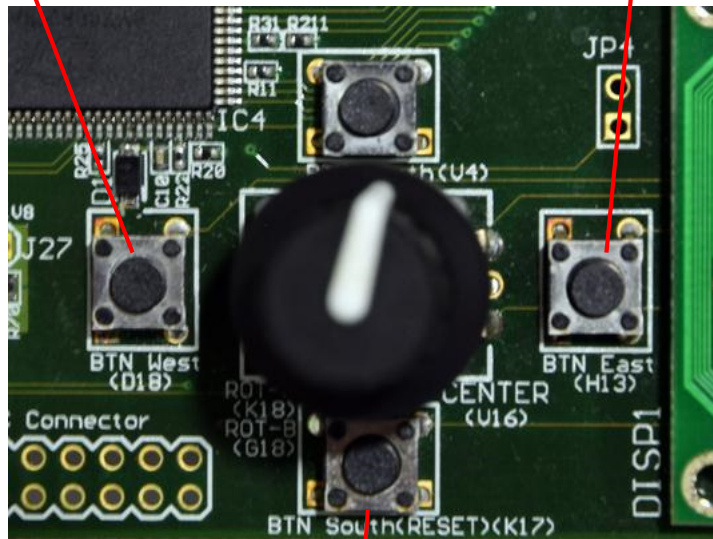
- ❑ Your circuit has a 4-bit counter register
 - The counter value is set to zero upon reset
 - The counter value is in 2's complement from $-8 \sim 7$
- ❑ Two push-buttons are used to increase/decrease the counter:
 - Push the west/east buttons increase/decrease the counter by 1
 - If the counter value becomes greater than 7, it is truncated to 7; if the value is smaller than -8 , it is set to -8
- ❑ The south button is used to reset the counter to zero
- ❑ The counter register value will be displayed in sign-extended binary format using the 8 LEDs at any time

Push buttons and LEDs Control

When a push button is pressed, the following actions is applied to the counter register:

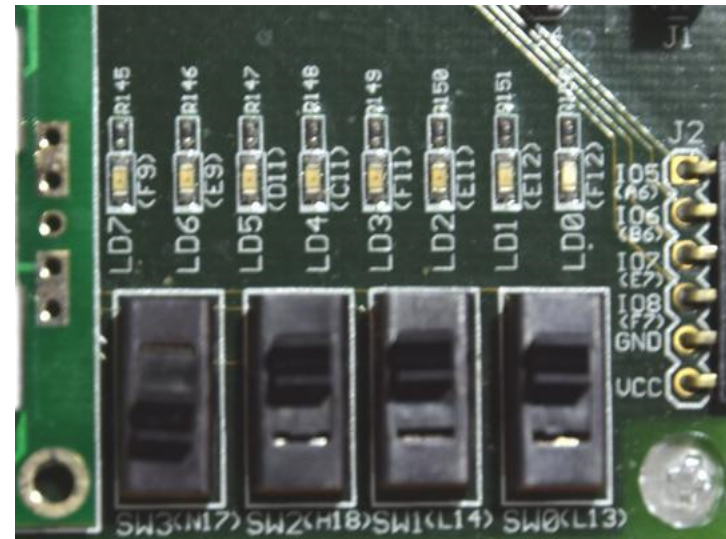
Counter + 1

Counter - 1



Counter reset

Use the LED lights to display the value of the counter register:
“on” means “1” and “off” means “0”.
The most significant bit is LD7.



User I/O Pins of an FPGA IC

- ❑ There are many “FPGA” pins that are used as user I/O pins: each pin connects to an I/O device such as the push-buttons or the LEDs:

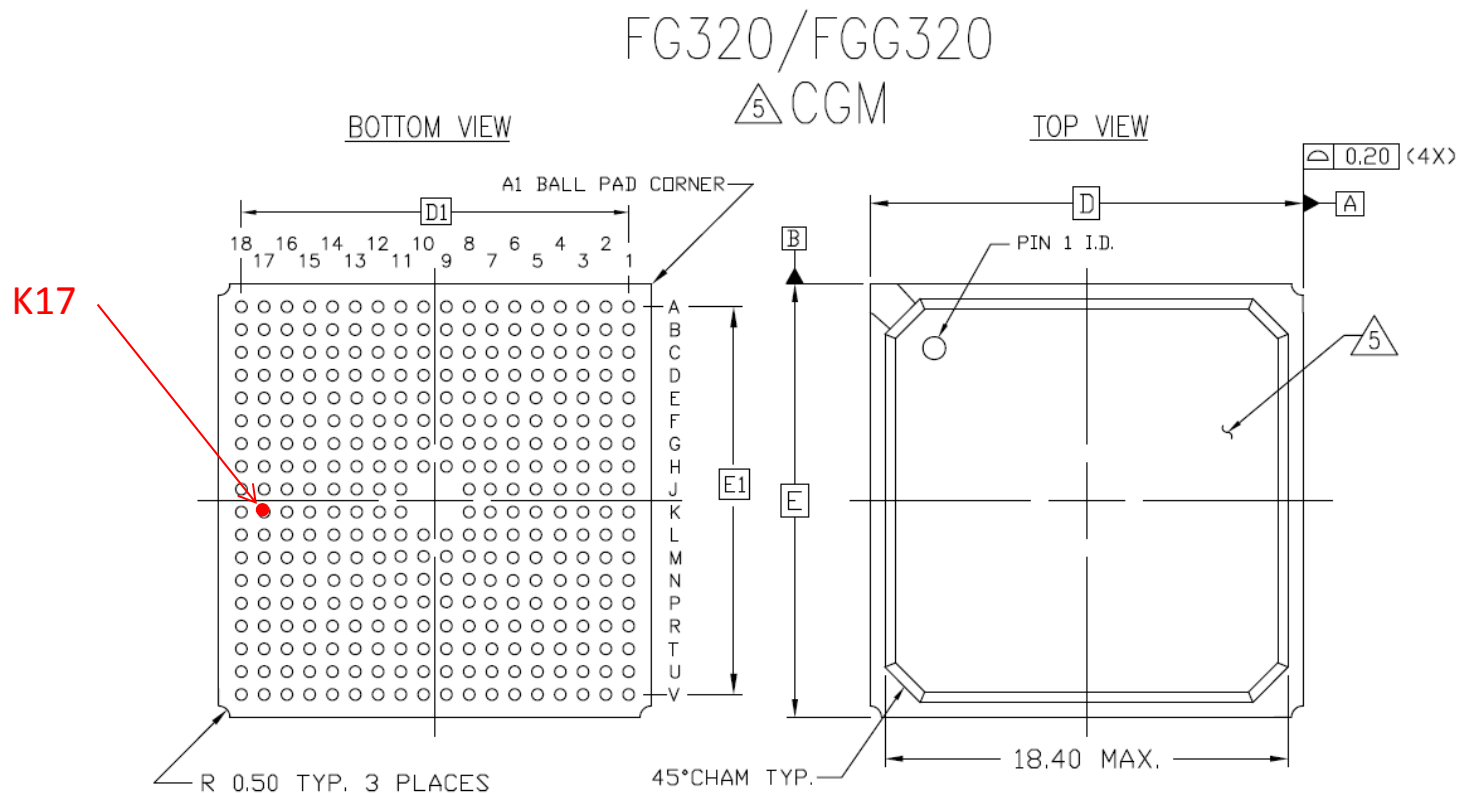


Bottom-view of an FPGA IC
(shows IC pins)

Top-view of an FPGA IC
(shows IC markings)

FPGA Pin Coordinates

- Each pin at the bottom of the FPGA has a coordinate. For example, “K17” is the coordinate of the red pin of our Spartan-3E IC:



Use the I/O Pin Signal in Verilog

- ❑ To read/write the I/O pins, we must map the pin coordinates to Verilog signals in our code
 - User constraints are used to do the job
- ❑ A user constraint is a text command that specifies the physical property in an HDL code. For example, for the four push-buttons, their mapping to Verilog signals can be as follows:

NET	"BTN_EAST"	LOC =	"H13"		IOSTANDARD =	LVTTTL		PULLDOWN;
NET	"BTN_NORTH"	LOC =	"V4"		IOSTANDARD =	LVTTTL		PULLDOWN;
NET	"BTN_SOUTH"	LOC =	"K17"		IOSTANDARD =	LVTTTL		PULLDOWN;
NET	"BTN_WEST"	LOC =	"D18"		IOSTANDARD =	LVTTTL		PULLDOWN;

Signal names to be used
in your Verilog code!

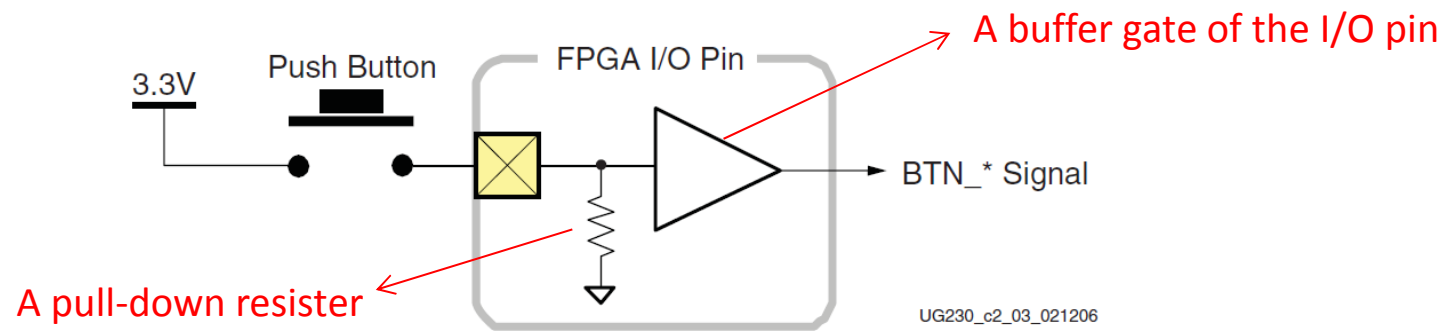
IC pin coordinates

Signal type

Signal property

How to Read the Input Push-Button

- ❑ The physical connection from an FPGA I/O pin to a push-button is as follows:

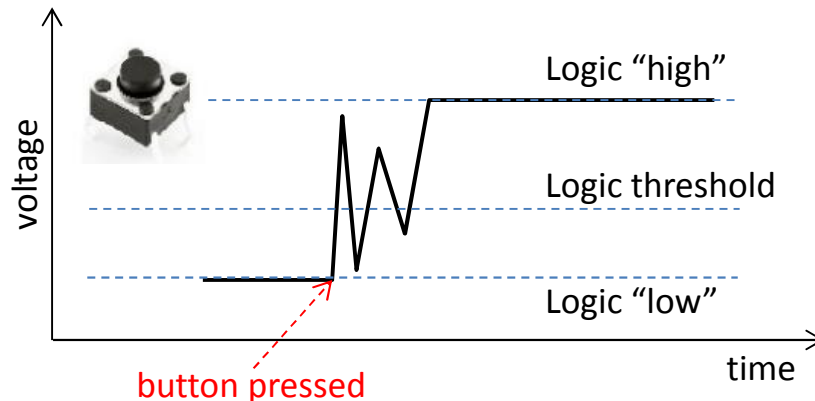


- ❑ Ideally, when a push-button is pushed (the circuit is closed), the FPGA pin that connects to the button becomes high voltage and the corresponding signal in Verilog reads “1”, otherwise it reads “0”.

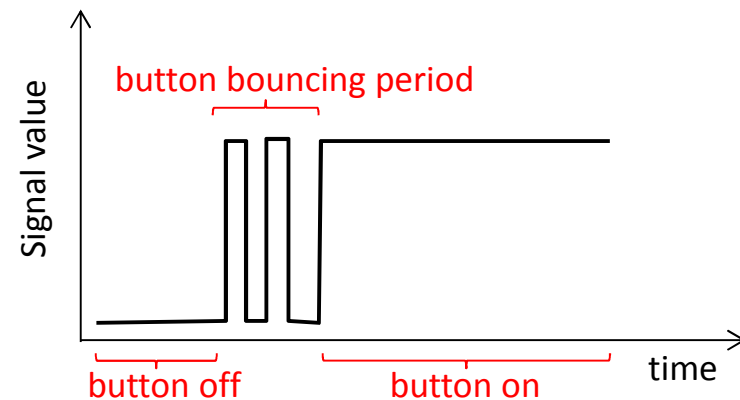
The Bouncing Problem

- ❑ In reality, however, the signal value oscillates between 0 and 1 several times before it stabilizes. This is called the bouncing behavior of a hardware button.

The physical voltage values:




The actual digital signal:



The De-bouncing Circuit

- ❑ To detect whether the button has been pressed, you cannot simply check the button signals:

```
always @(posedge clk, posedge reset)
  if (reset == 1)
    button_north_is_pressed = 0;
  else
    button_north_is_pressed = (BTN_NORTH) ? 1 : 0;
```

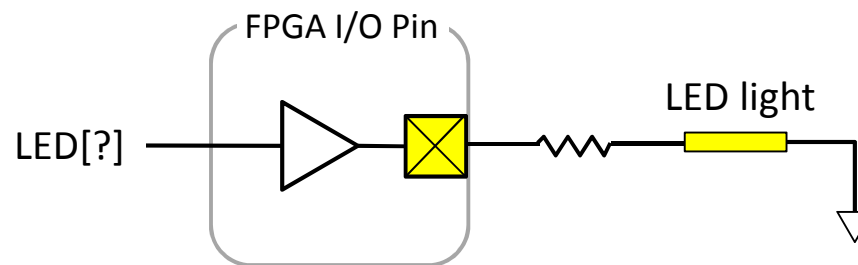


- This circuit will catch all the state changes during the bouncing period → a single button click will be treated as multiple clicks!
- ❑ You must find a way to average-out the noises of the push-button signal during the bouncing period
 - Hint: you can use a shift register to accumulate the input signal; or a timer to wait out the bouncing period

Turn On/Off the LEDs

- ❑ The LEDs can be turned on or off by writing 1 or 0 to the corresponding Verilog signals (`reg [7:0] LED`):

NET "LED[7]"	LOC = "F9"		IOSTANDARD = LVTTTL		DRIVE = 8;
NET "LED[6]"	LOC = "E9"		IOSTANDARD = LVTTTL		DRIVE = 8;
NET "LED[5]"	LOC = "D11"		IOSTANDARD = LVTTTL		DRIVE = 8;
NET "LED[4]"	LOC = "C11"		IOSTANDARD = LVTTTL		DRIVE = 8;
NET "LED[3]"	LOC = "F11"		IOSTANDARD = LVTTTL		DRIVE = 8;
NET "LED[2]"	LOC = "E11"		IOSTANDARD = LVTTTL		DRIVE = 8;
NET "LED[1]"	LOC = "E12"		IOSTANDARD = LVTTTL		DRIVE = 8;
NET "LED[0]"	LOC = "F12"		IOSTANDARD = LVTTTL		DRIVE = 8;



Clock and Reset Pins

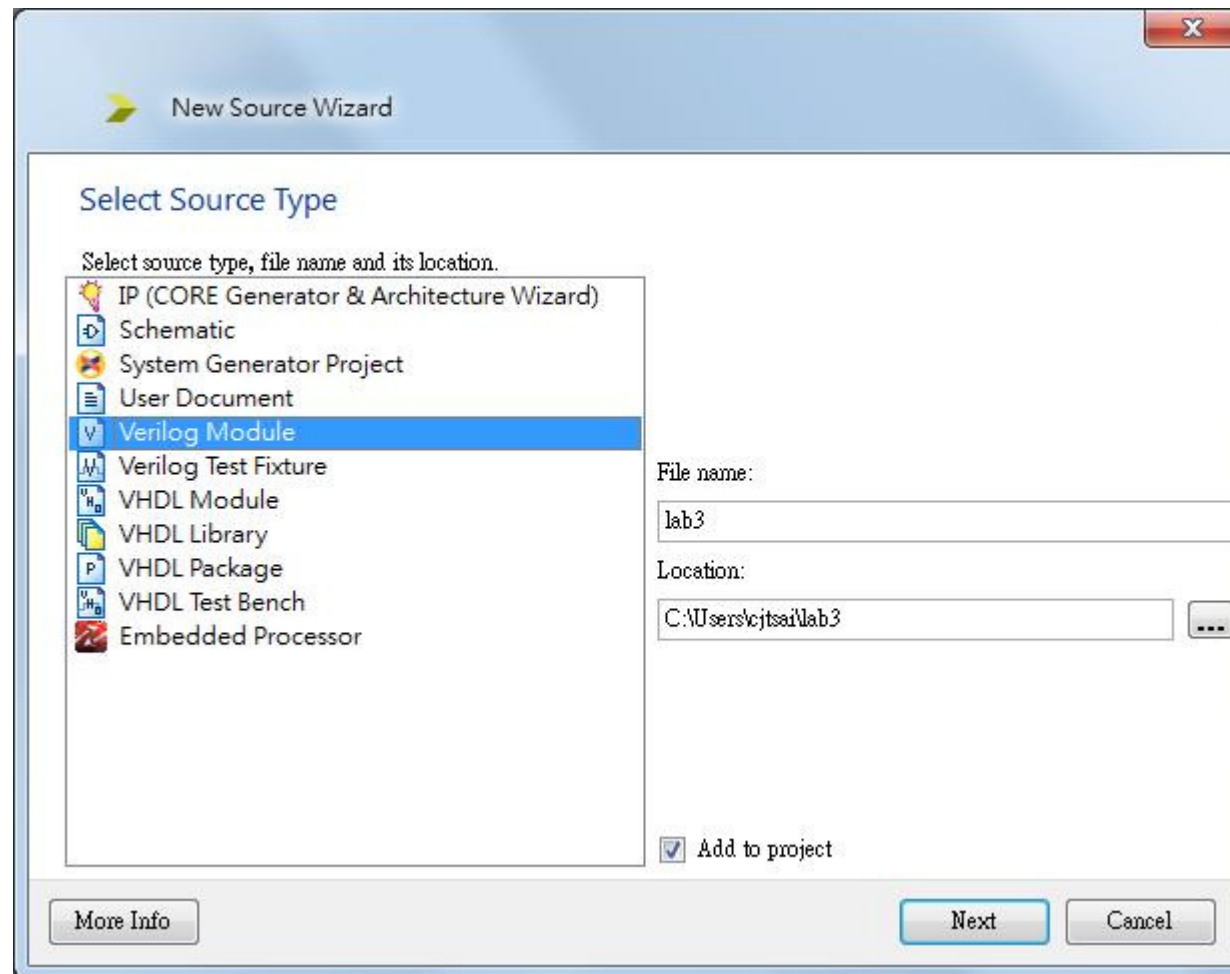
- ❑ For synchronous design, you need a clock signal for your circuit
 - The clock signal usually comes from an on-board oscillator
 - There is an FPGA pin that connects to the oscillator

```
NET "CLK" LOC = "C9" | IOSTANDARD = LVCMOS33;  
  
# Define clock period for 50 MHz oscillator  
NET "CLK" PERIOD = 20.0ns HIGH 40%;
```

- ❑ For the Spartan-3E Starter board, the south push-button (K17) is used as the reset pin
 - The reset pin is connected to an active de-bouncing IC so you do not have to de-bounce its input!

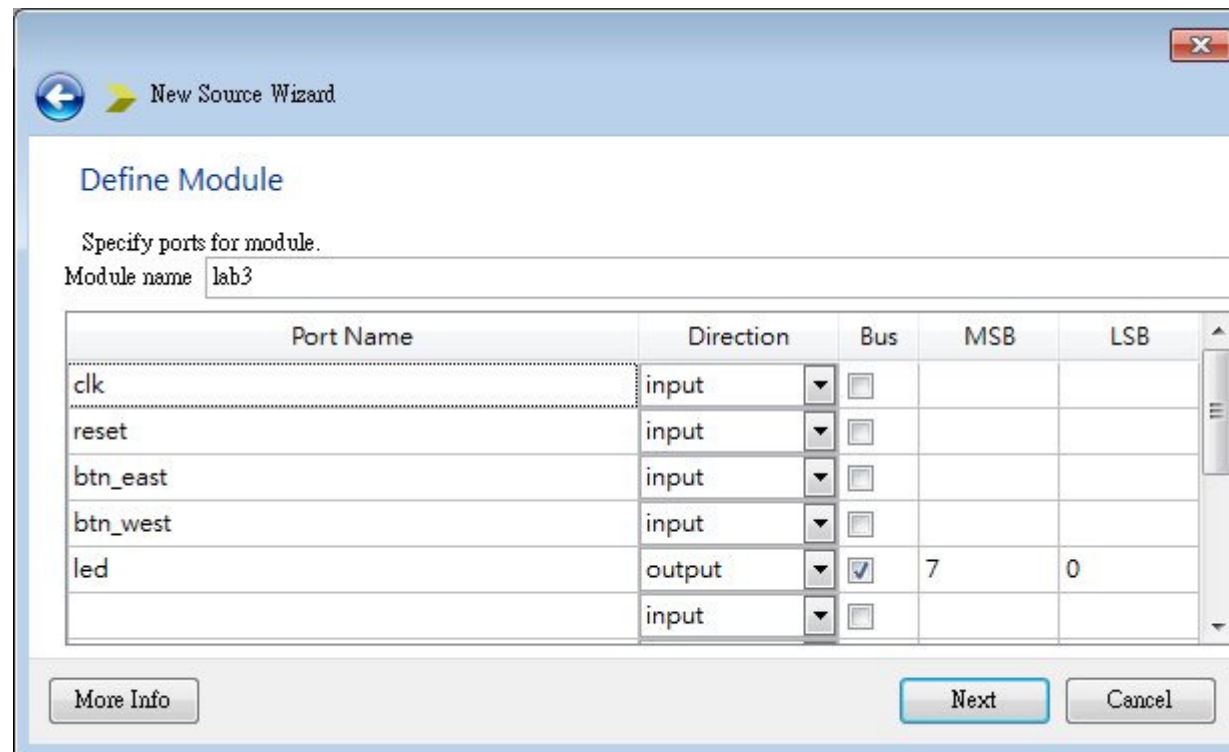
Create ISE Projects for the Board

- ❑ As before, we create a new ISE project:



Specify I/O Ports

- ❑ But this time, we specify the I/O ports of the module as follows:

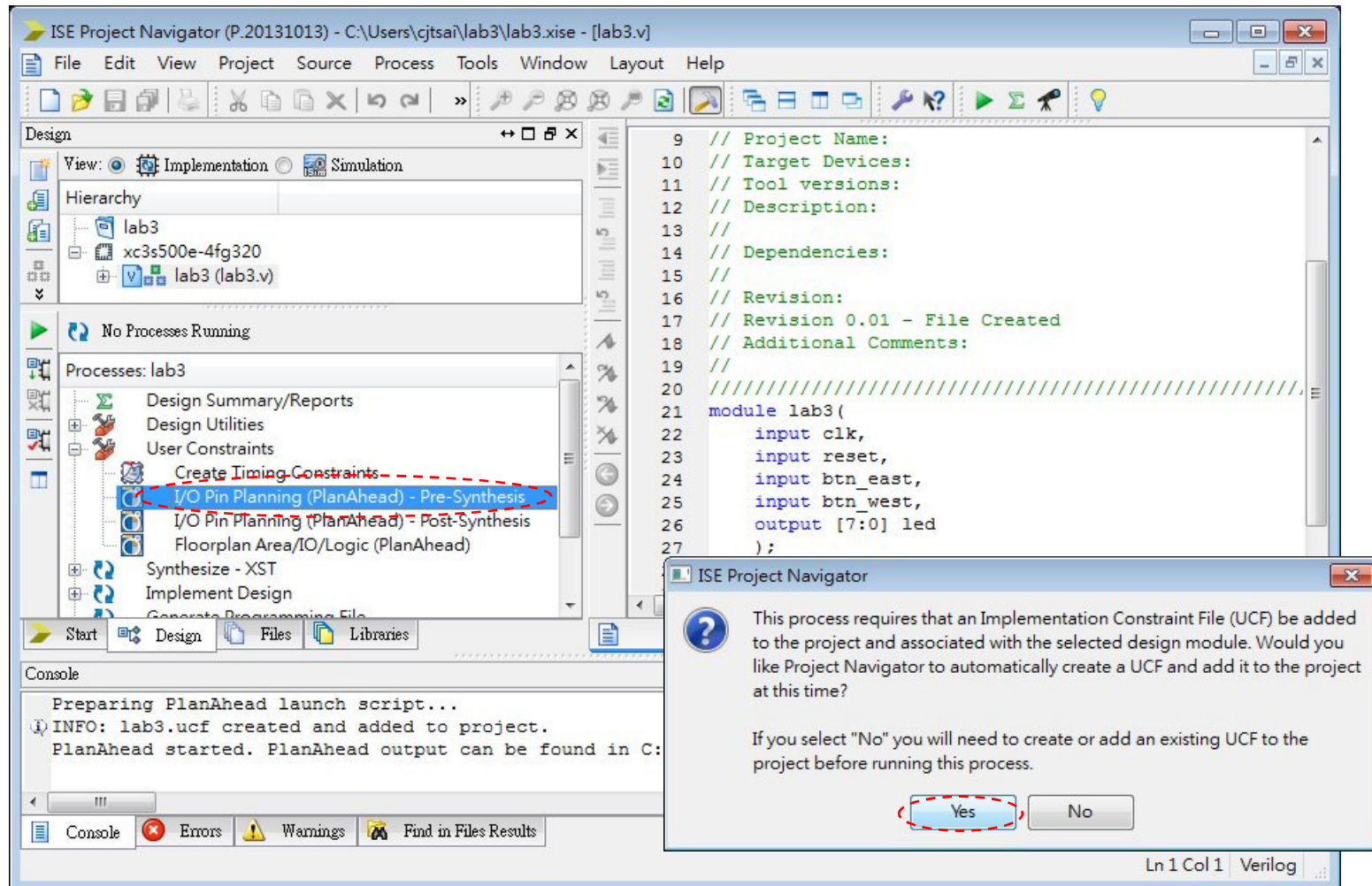


The image shows a screenshot of the 'New Source Wizard' dialog box, specifically the 'Define Module' step. The window title is 'New Source Wizard'. Below the title bar, there is a back arrow icon and the text 'New Source Wizard'. The main area is titled 'Define Module' and contains the instruction 'Specify ports for module.' Below this, there is a text field for 'Module name' with the value 'lab3' entered. A table is displayed with the following columns: 'Port Name', 'Direction', 'Bus', 'MSB', and 'LSB'. The table contains five rows of data. The first four rows have 'input' as the direction and no bus or bit range specified. The fifth row has 'output' as the direction, a checked 'Bus' checkbox, and '7' for MSB and '0' for LSB. At the bottom of the dialog, there are three buttons: 'More Info', 'Next', and 'Cancel'.

Port Name	Direction	Bus	MSB	LSB
clk	input	<input type="checkbox"/>		
reset	input	<input type="checkbox"/>		
btn_east	input	<input type="checkbox"/>		
btn_west	input	<input type="checkbox"/>		
led	output	<input checked="" type="checkbox"/>	7	0
	input	<input type="checkbox"/>		

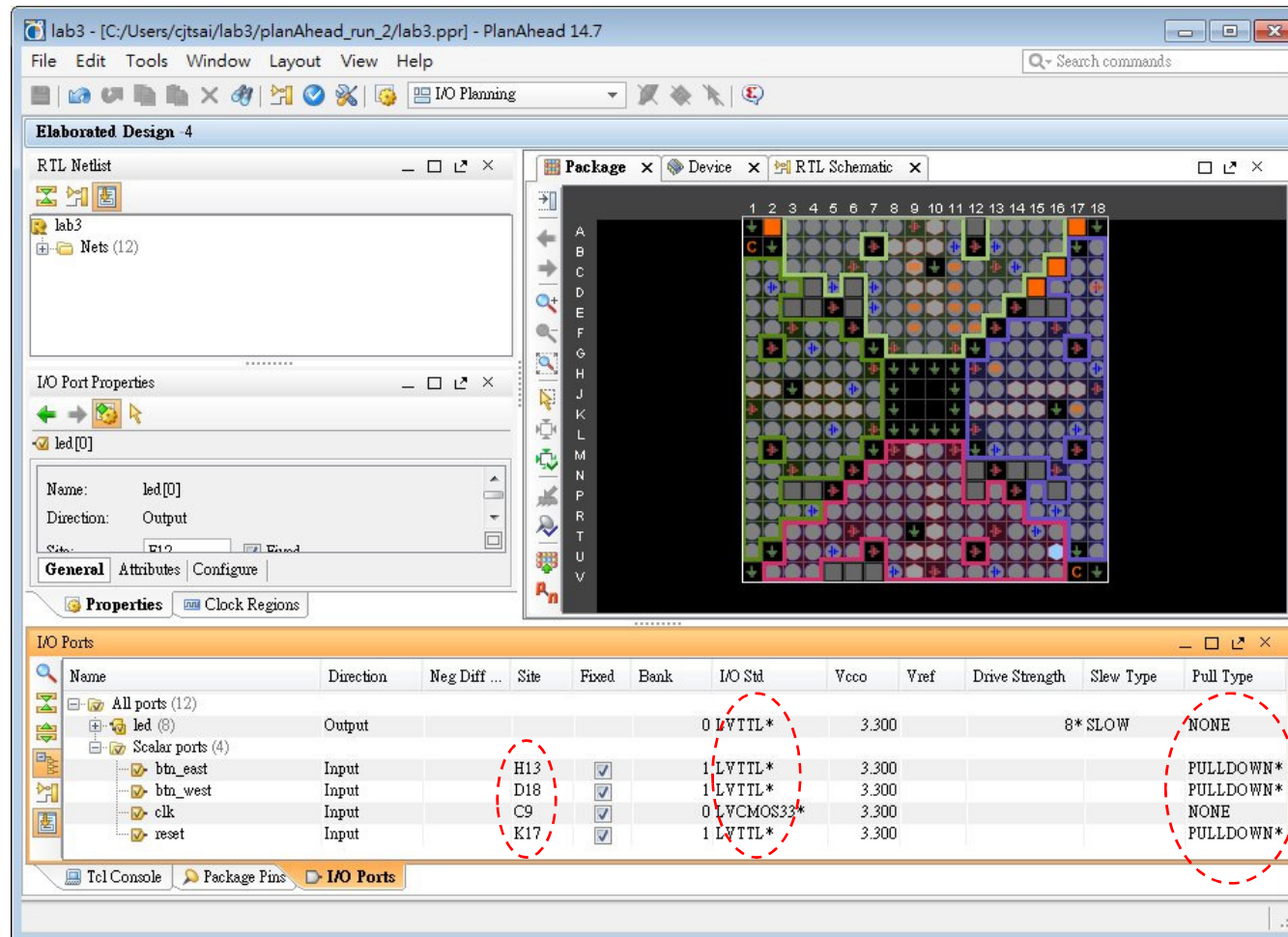
Create User Constraints

- ❑ Select “I/O Pin Planning” to create user constraints:



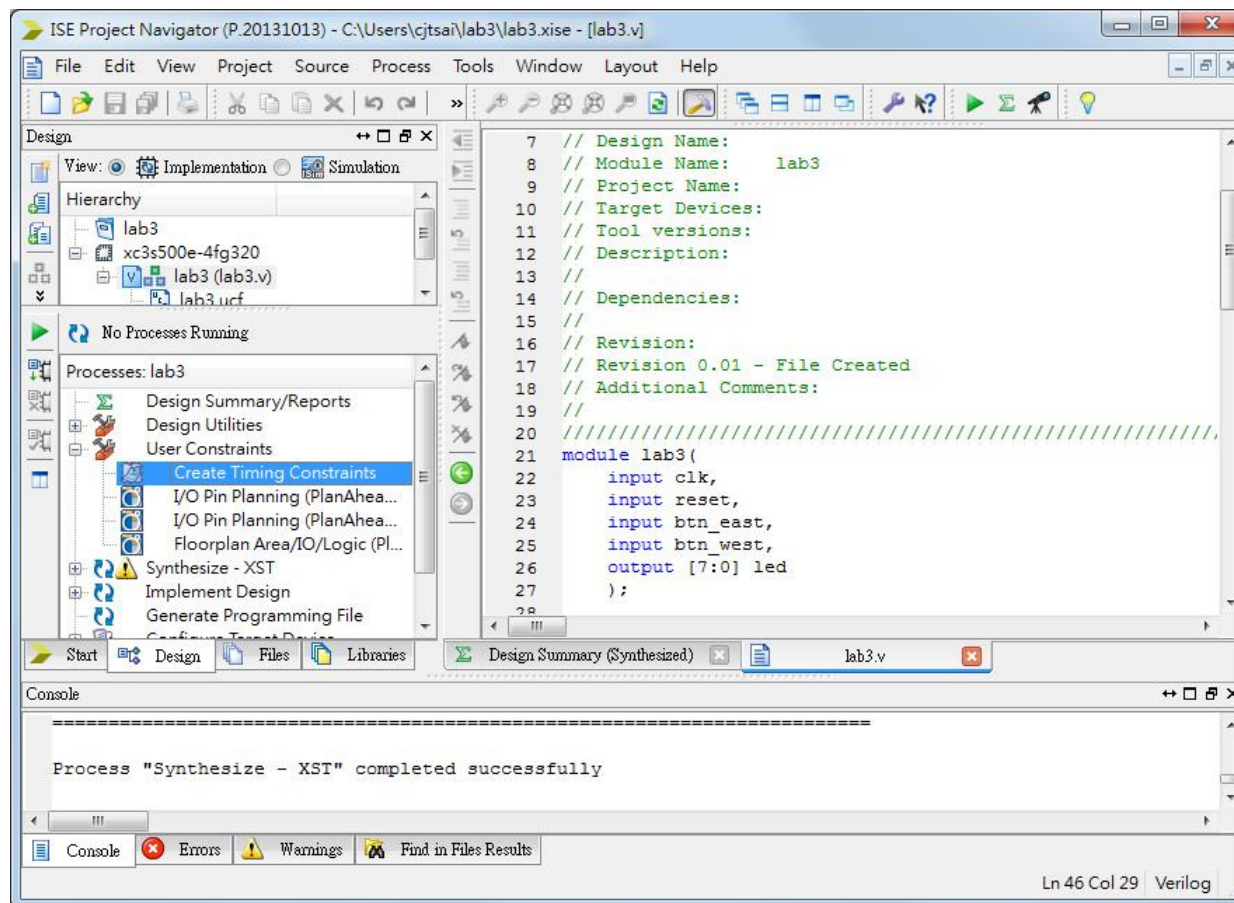
The User Constraint Editor

- ❑ Assign the Verilog I/O ports to pin coordinates:



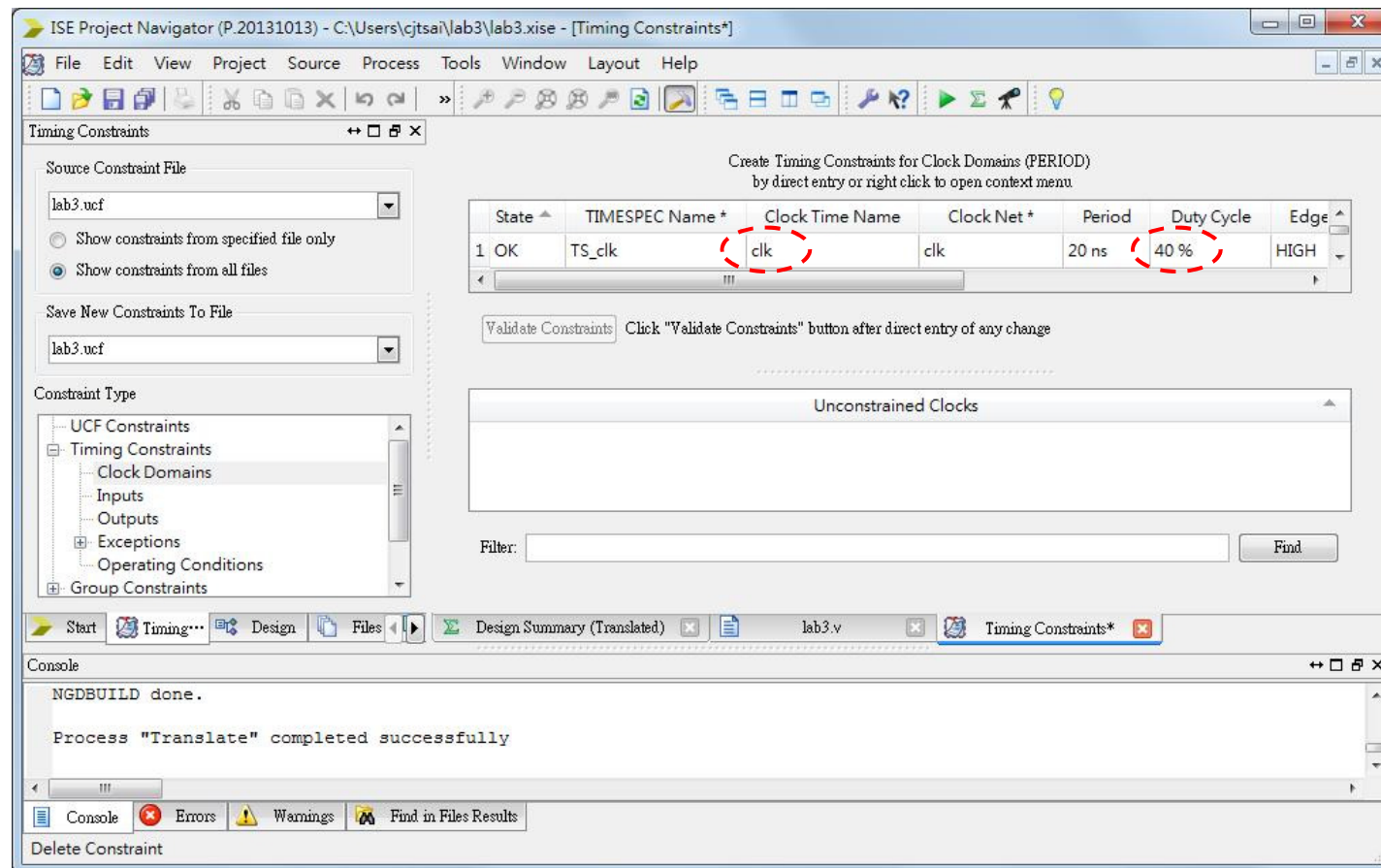
Adding Clock Signal Constraints (1/2)

- ❑ Now, the signal “clk” has a default timing constraint, we can modify it once the design is ready for synthesis:



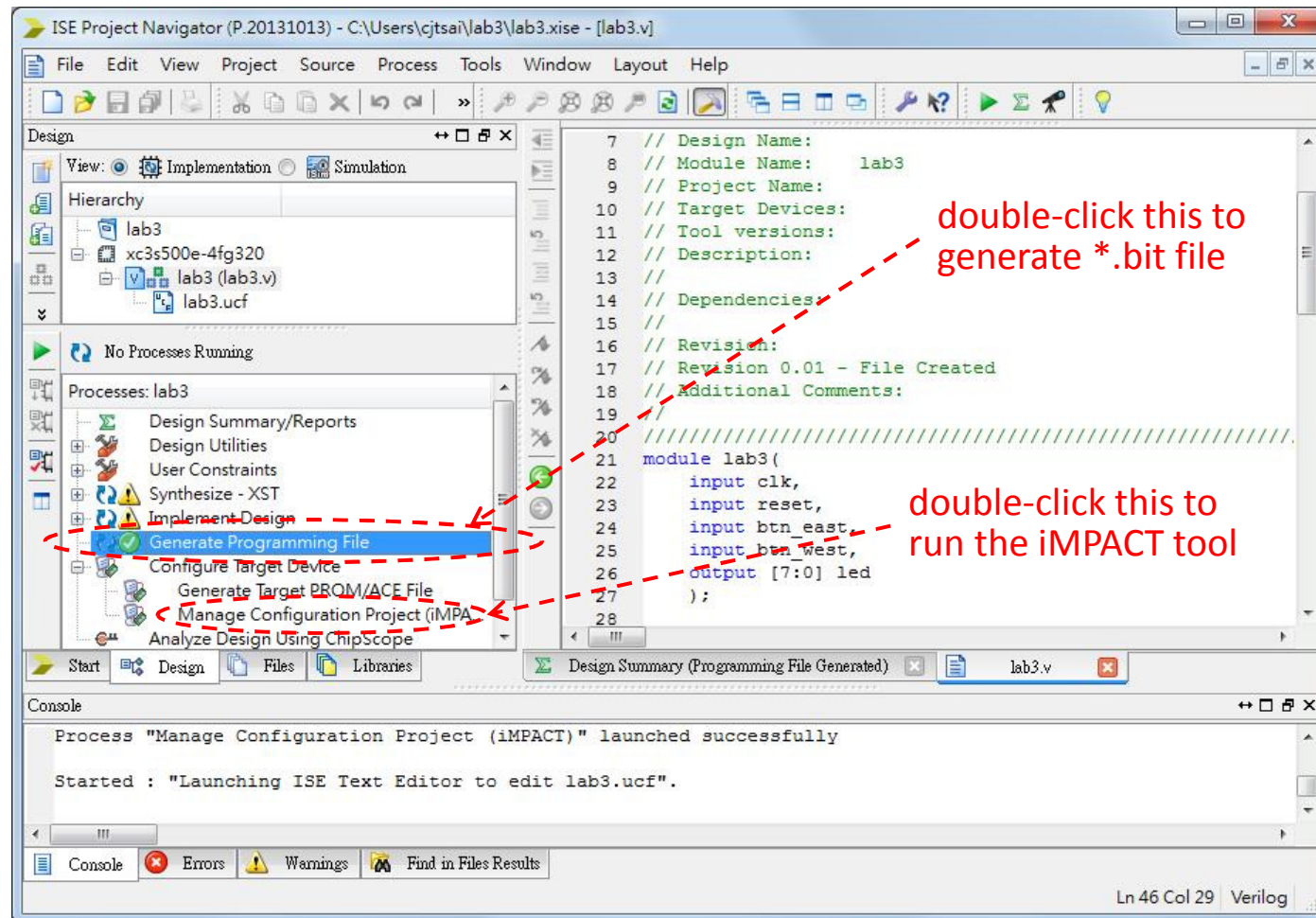
Adding Clock Signal Constraints (2/2)

- ❑ The default constraint should work fine, but we can change the duty cycle to better match the board



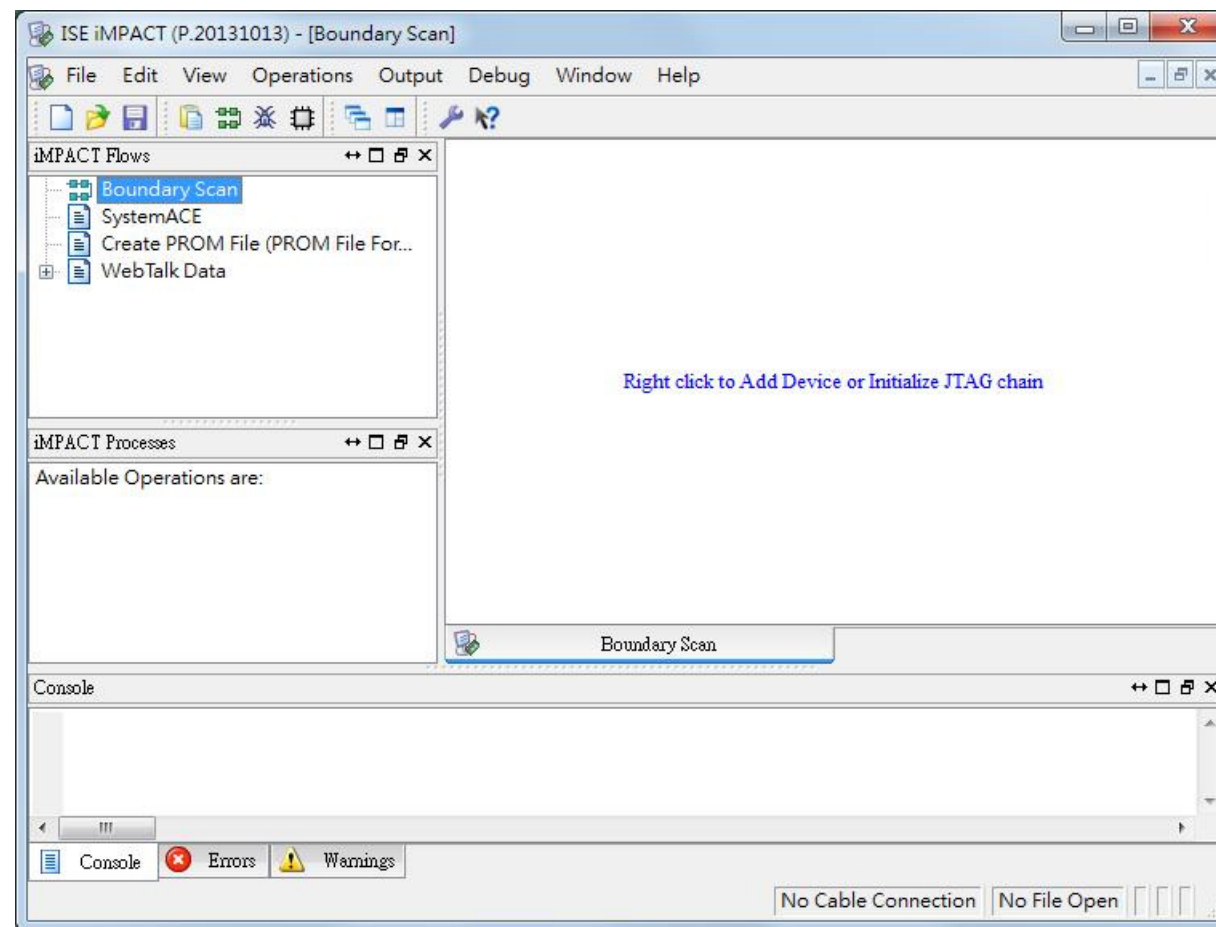
Generate the Programming File

- ❑ Now, create the programming file “lab3.bit” for FPGA:



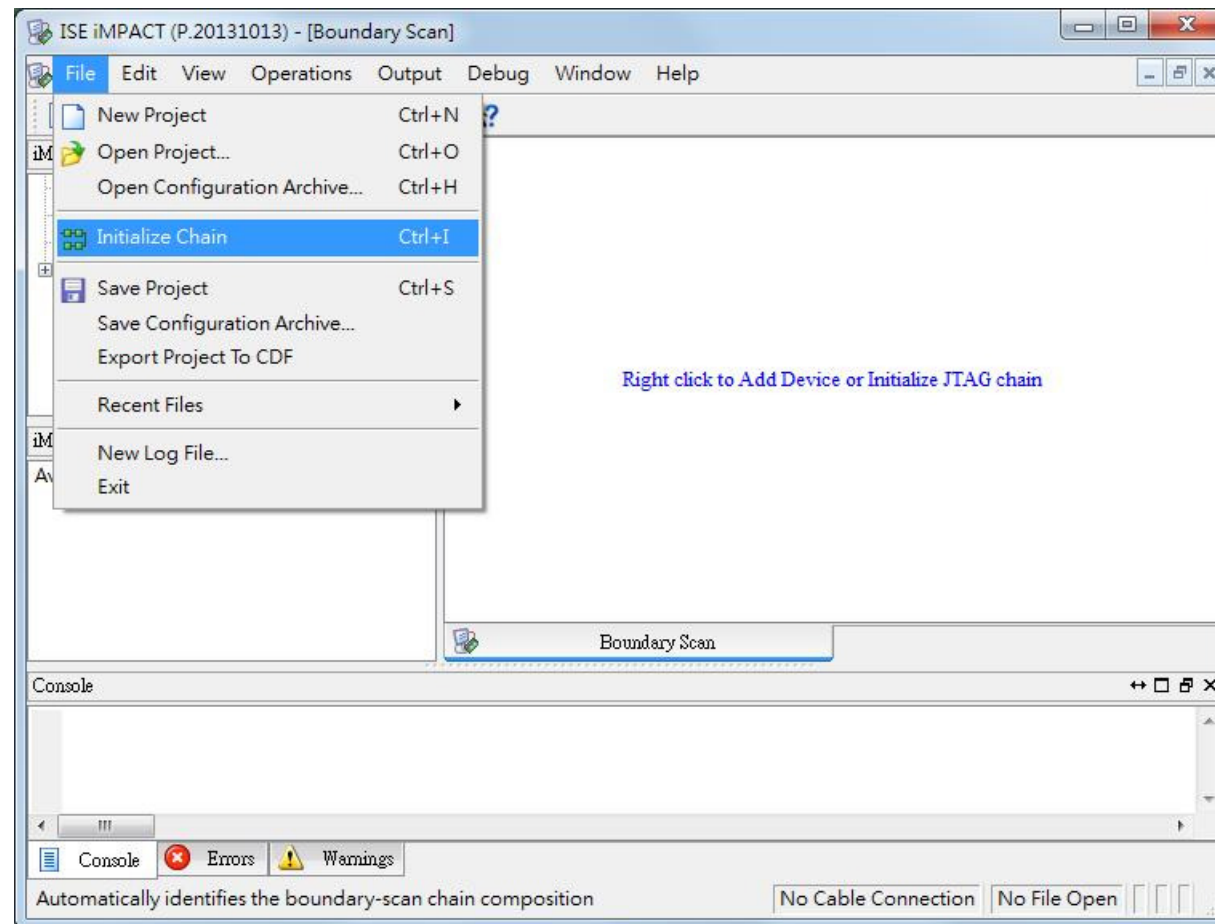
Downloading Your Circuit to the Board

- ❑ To download your programming file into the FPGA, you must run the “iMPACT” tool:



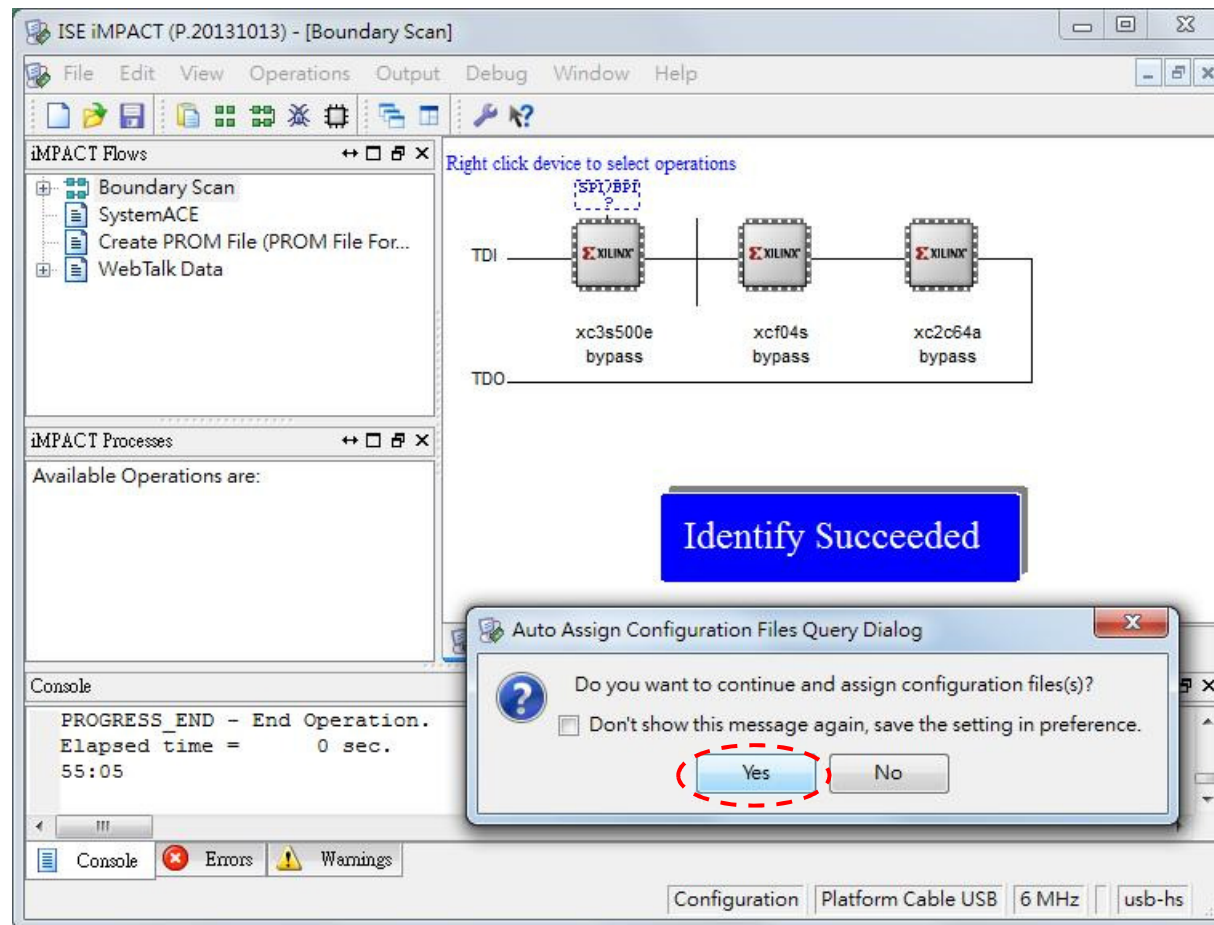
Detect the FPGA from iMPACT

- ❑ Power on the board and initialize the JTAG chain:



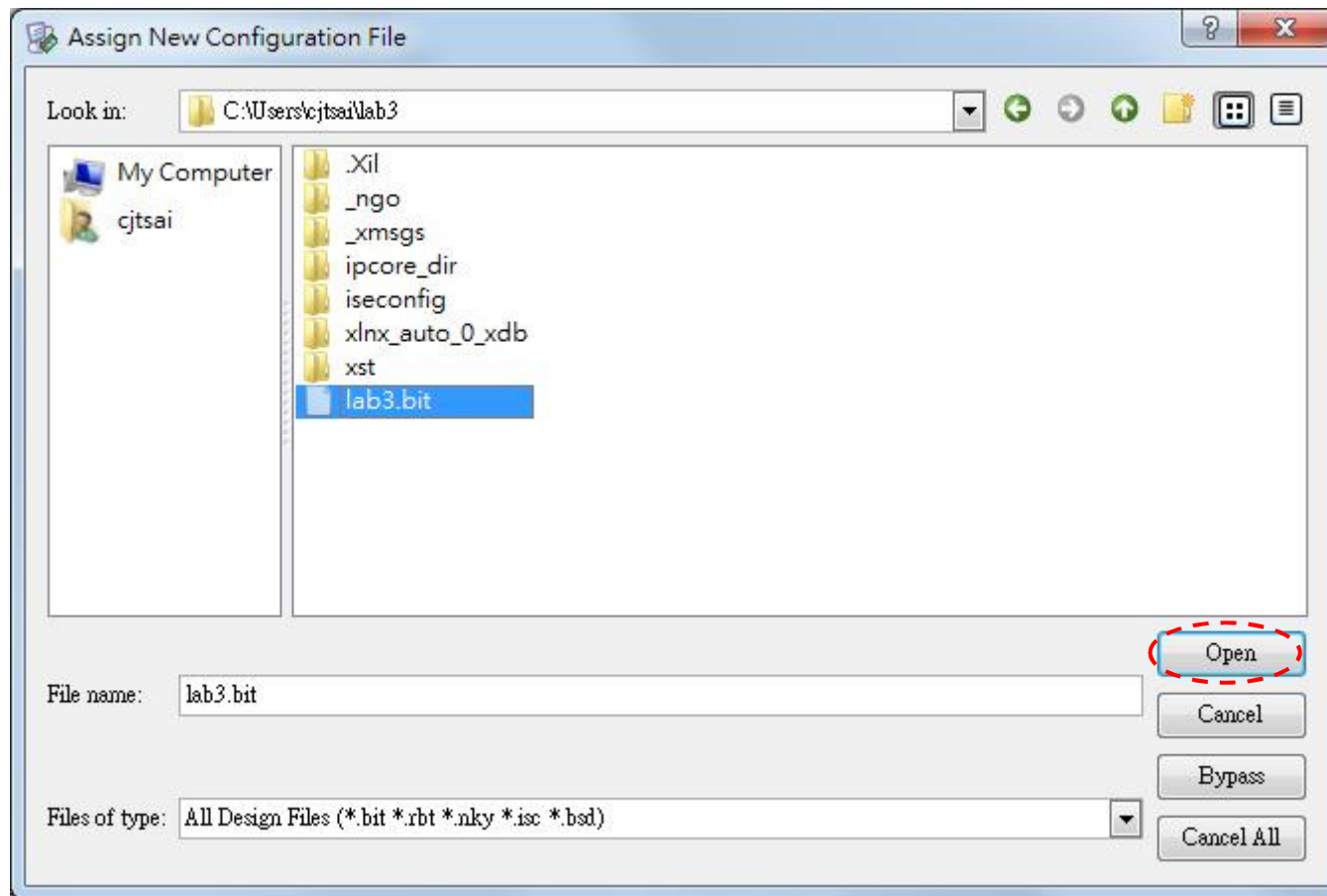
Assign the *.bit File to the FPGA

- ❑ JTAG shows three IC's, we only use the first one:



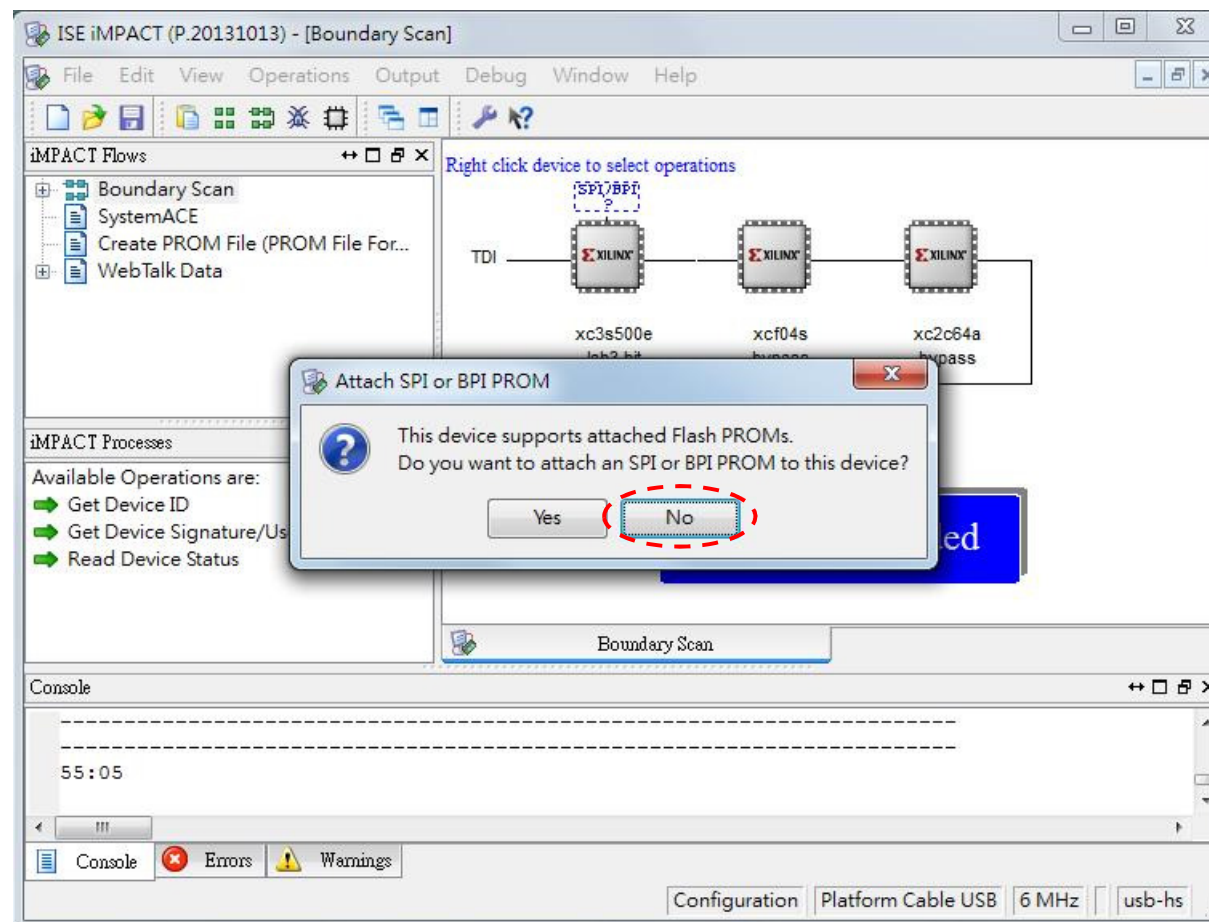
Browse to the *.bit File

- ❑ Browse to your directory and select the bit file for Lab3:



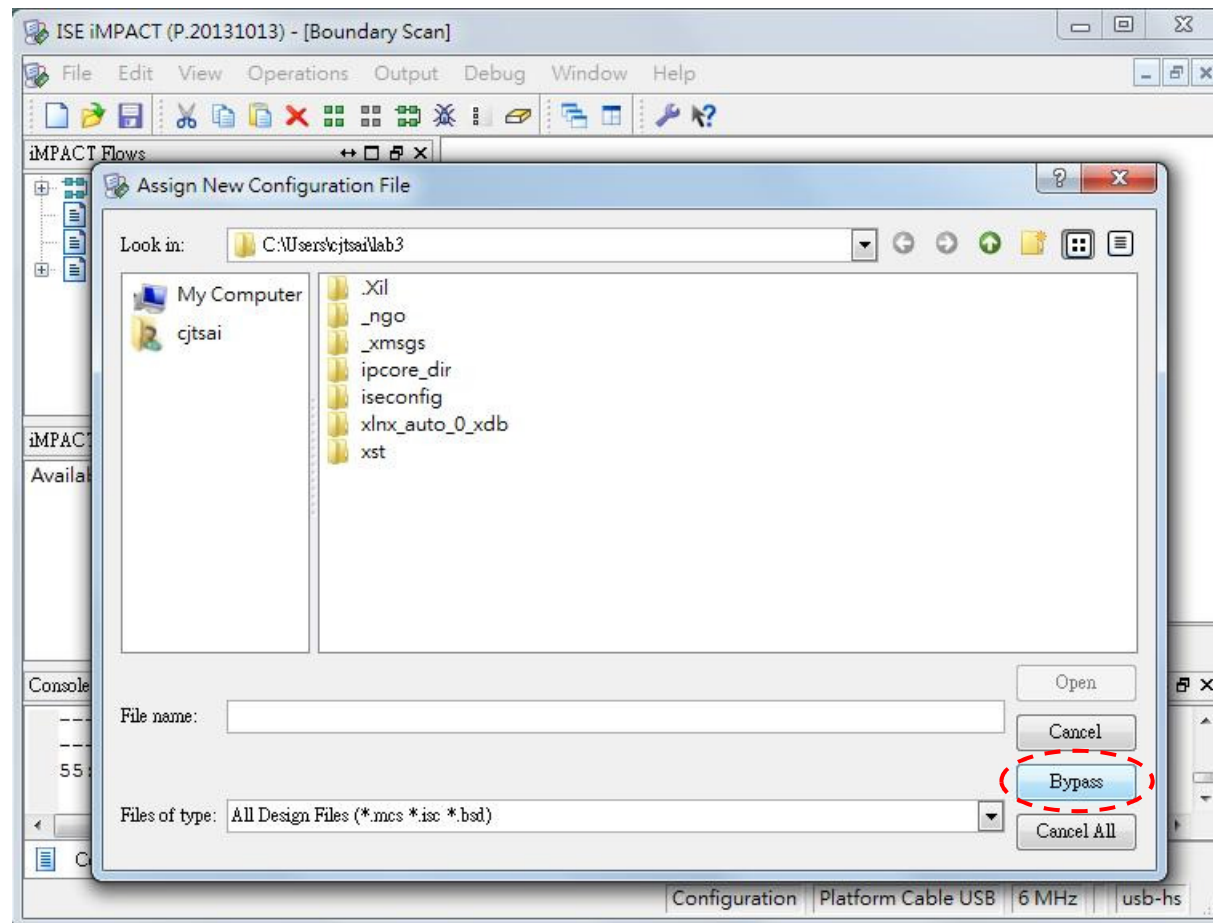
Skip Configuration from SPI ROM

- ❑ We do not use the SPI flash to configure the FPGA



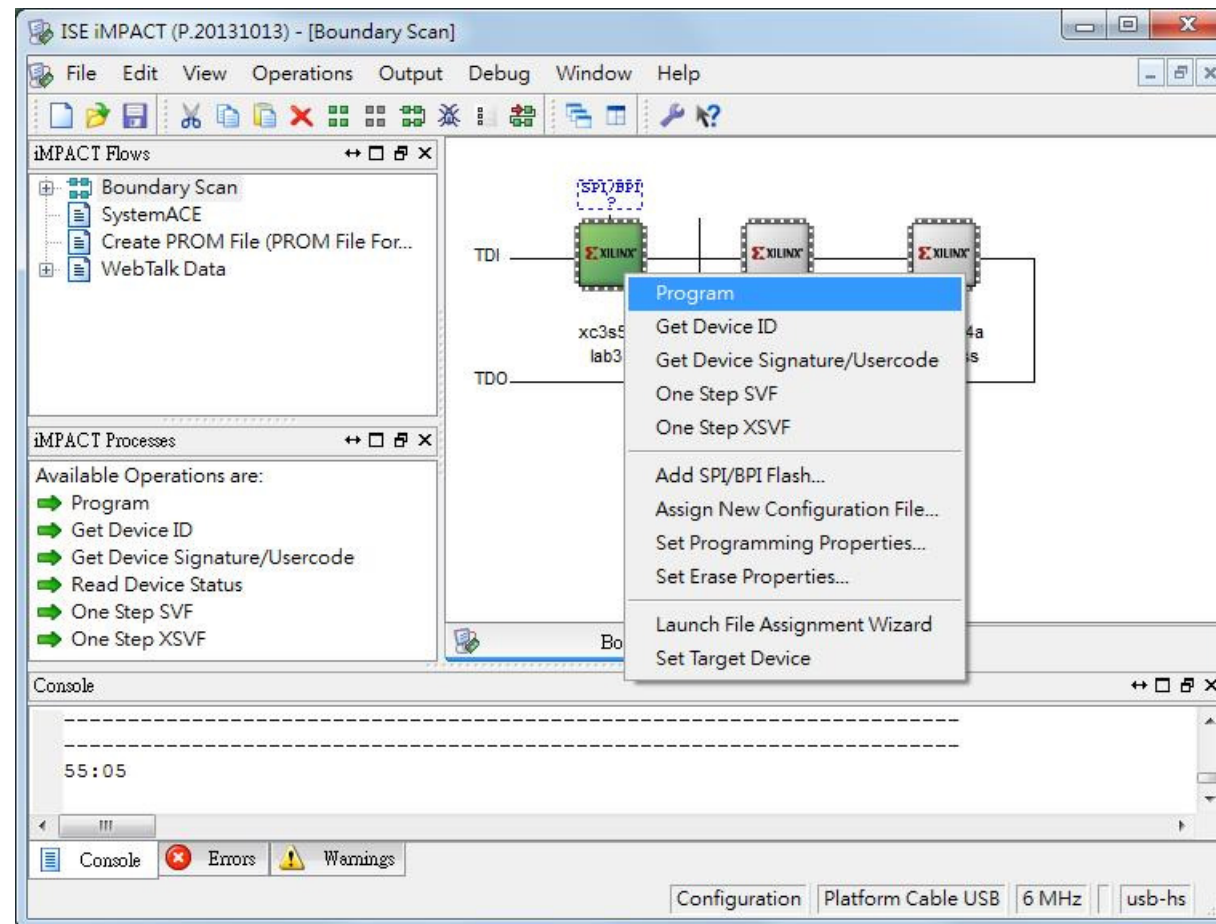
Bypass the Next Two IC's

- ❑ The next two IC's in the JTAG chain are special flash memory → they are obsolete, bypass next **two** boxes!



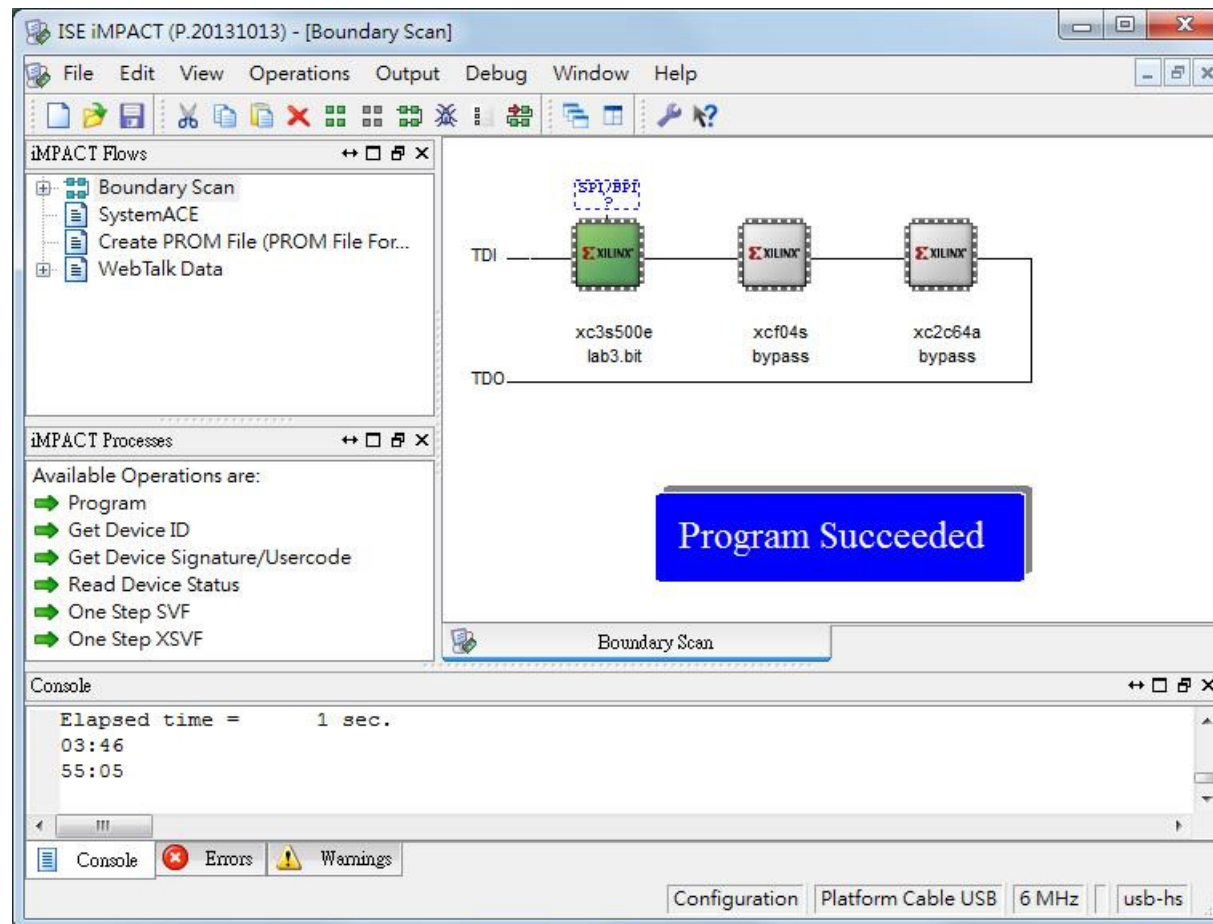
Program the FPGA

- ❑ Finally, we can right-click the green IC to program it!



Test Your Design

- ❑ You can now test your circuit by clicking the buttons:



Reference

- ❑ Xilinx, *Spartan-3E Starter Kit Board User Guide, UG230 (v1.0)*, March 9, 2006. Available from:

http://www.xilinx.com/support/documentation/boards_and_kits/ug230.pdf