

# THỰC HÀNH SOC

## Bài 3 – GIAO TIẾP I/O TỰ THIẾT KẾ

### 1. MỤC ĐÍCH

Thông qua bài thực hành này, sinh viên sẽ hiểu rõ:

- o Cách xây dựng và kết nối các giao tiếp IO tự thiết kế vào hệ thống SoPC.
- o Cách sử dụng công cụ NIOS II để điều khiển các giao tiếp IO trên.

### 2. NỘI DUNG

#### 2.1. Hệ thống phần cứng

##### 2.1.1. Tạo project Quartus

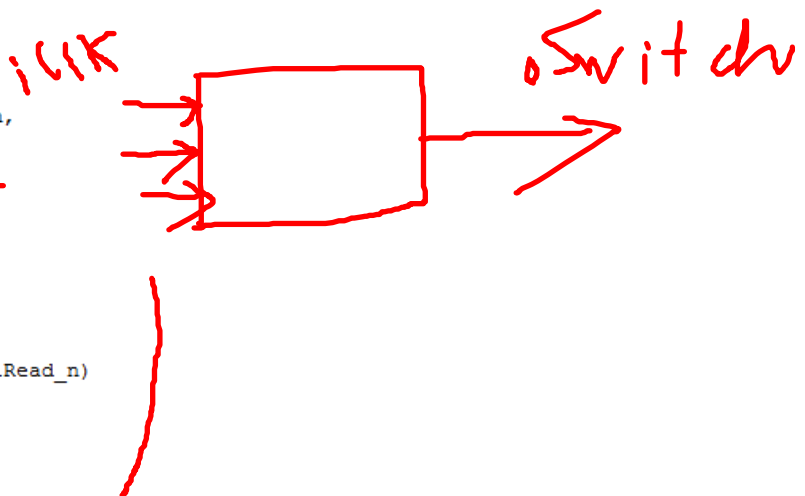
Tạo project Quartus tên là “lab3”. Lưu ý đường dẫn thư mục project không được có khoảng trắng.

Nếu sử dụng board DE2-115, chọn Family là Cyclone IV E, device là EP4CE115F29C8.

Nếu sử dụng board DE2, chọn Family là Cyclone II, device là EP2C35F672C6.

Tạo file “switch.v” có nội dung như bên dưới và thêm vào project.

```
1 module switch(  
2  
3     // Avalon Slave interface  
4     input    iClk,  
5     input    iReset_n,  
6     input    iChip_select_n,  
7     input    iRead_n,  
8     output reg [31:0] oSwitch,  
9  
10    // Export signal  
11    input [31:0] iSwitch);  
12  
13 always @ (posedge iClk) begin  
14     if (~iReset_n)  
15         oSwitch = 32'h0;  
16     else if (~iChip_select_n && ~iRead_n)  
17         oSwitch <= iSwitch;  
18 end  
19  
20 endmodule
```



Tạo file “led.v” có nội dung như bên dưới và thêm vào project.

```

1 module led(
2
3     // Avalon Slave interface
4     input    iClk,
5     input    iReset_n,
6     input    iChip_select_n,
7     input    iWrite_n,
8     input    [31:0] iLed,
9
10    // Export signal
11    output reg [31:0] oLed);
12
13 always @ (posedge iClk) begin
14     if (~iReset_n)
15         oLed <= 32'h0;
16     else if (~iChip_select_n && ~iWrite_n)
17         oLed <= iLed;
18 end
19
20 endmodule

```

## 2.1.2. Xây dựng hệ thống Qsys

Xây dựng hệ thống phần cứng như hình bên dưới.

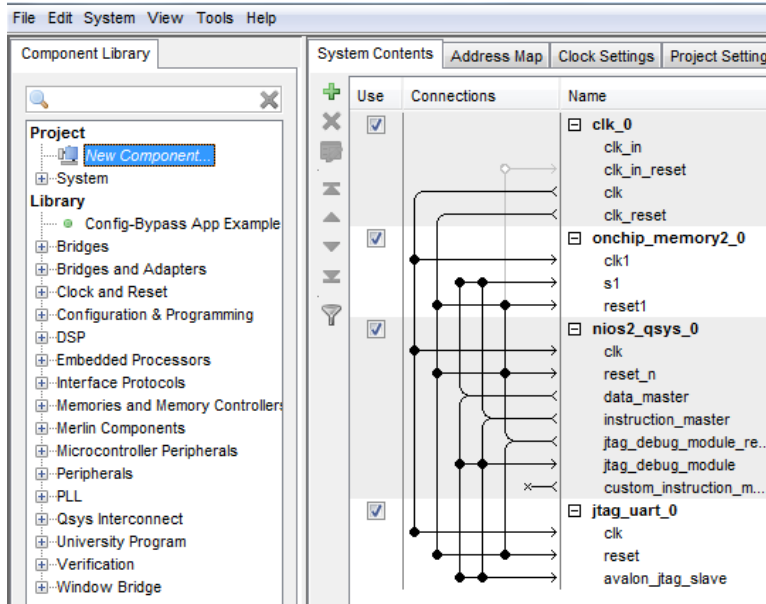
The screenshot shows the Qsys software interface with the following components listed in the table:

Use	Connections	Name	Description	Export	Clock	Base	End	IRQ	Opcc
<input checked="" type="checkbox"/>		clk_0	Clock Source	clk	clk_0				
<input checked="" type="checkbox"/>		clk_in_reset	Clock Input	reset					
<input checked="" type="checkbox"/>		clk	Clock Output	Double-c	clk_0				
<input checked="" type="checkbox"/>		clk_reset	Reset Output	Double-c					
<input checked="" type="checkbox"/>		onchip_memory2_0	On-Chip Memory (RAM or ROM)						
<input checked="" type="checkbox"/>		clk1	Clock Input	Double-c	clk_0				
<input checked="" type="checkbox"/>		s1	Avalon Memory Mapped Slave	Double-c	[clk1]	0x0000_0000	0x0000_ffff		
<input checked="" type="checkbox"/>		reset1	Reset Input	Double-c	[clk1]				
<input checked="" type="checkbox"/>		nios2_qsys_0	Nios II Processor						
<input checked="" type="checkbox"/>		clk	Clock Input	Double-c	clk_0				
<input checked="" type="checkbox"/>		reset_n	Reset Input	Double-c	[clk]				
<input checked="" type="checkbox"/>		data_master	Avalon Memory Mapped Master	Double-c	[clk]			IRQ 0	IRQ 31
<input checked="" type="checkbox"/>		instruction_master	Avalon Memory Mapped Master	Double-c	[clk]				
<input checked="" type="checkbox"/>		jtag_debug_module_reset	Reset Output	Double-c	[clk]				
<input checked="" type="checkbox"/>		jtag_debug_module	Avalon Memory Mapped Slave	Double-c	[clk]	0x0001_0800	0x0001_0fff		
<input checked="" type="checkbox"/>		custom_instruction_master	Custom Instruction Master	Double-c					
<input checked="" type="checkbox"/>		jtag_uart_0	JTAG UART						
<input checked="" type="checkbox"/>		clk	Clock Input	Double-c	clk_0				
<input checked="" type="checkbox"/>		reset	Reset Input	Double-c	[clk]				
<input checked="" type="checkbox"/>		avalon_jtag_slave	Avalon Memory Mapped Slave	Double-c	[clk]	0x0001_1008	0x0001_100f		

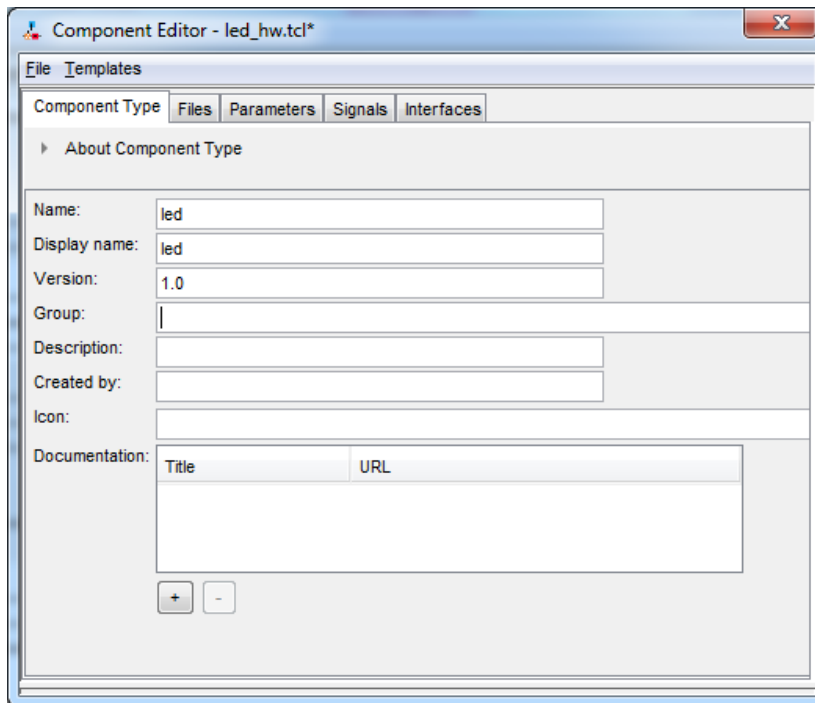
The Messages pane at the bottom shows "0 Errors, 0 Warnings".

Tiếp theo, chúng ta sẽ thêm module “led.v” đã thiết kế ở mục 2.1.1 vào thư viện của Qsys.

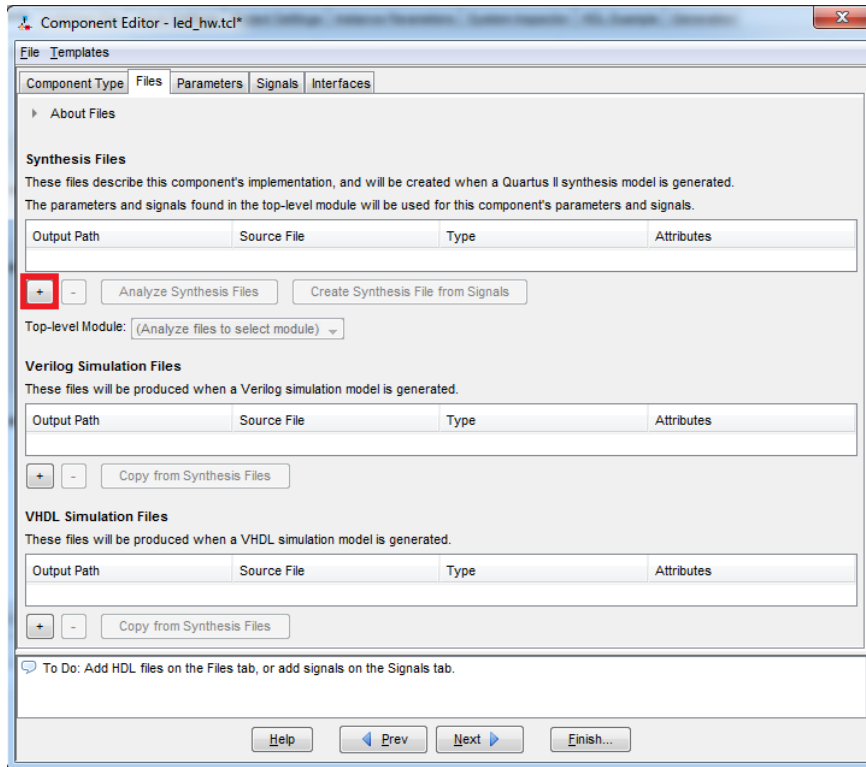
Double click vào “New component” ở cửa sổ thư viện.



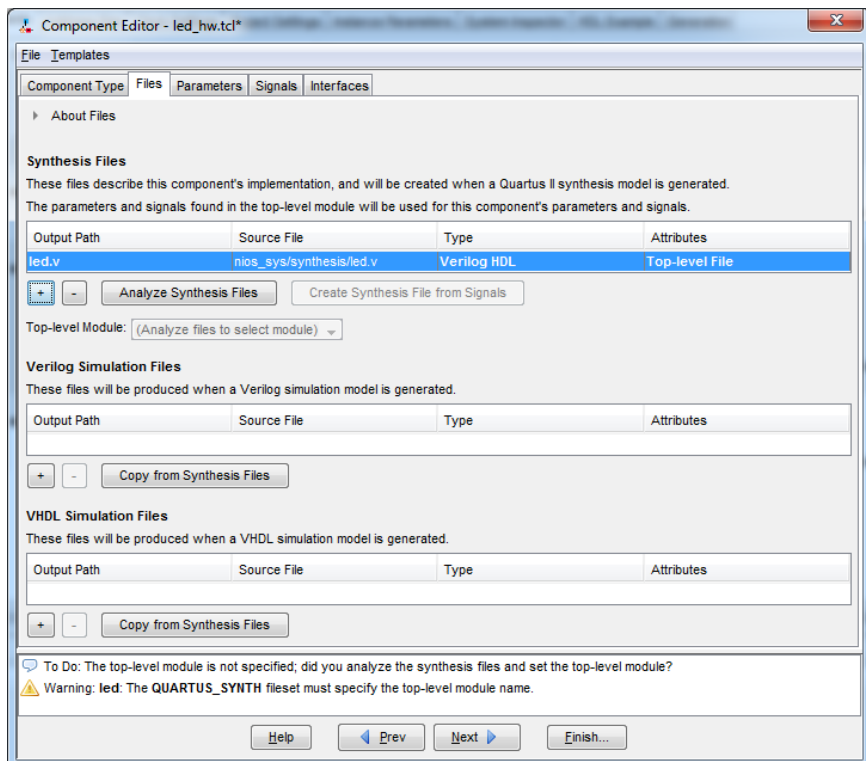
Đặt tên cho module ở tab “Component Type” như bên dưới.



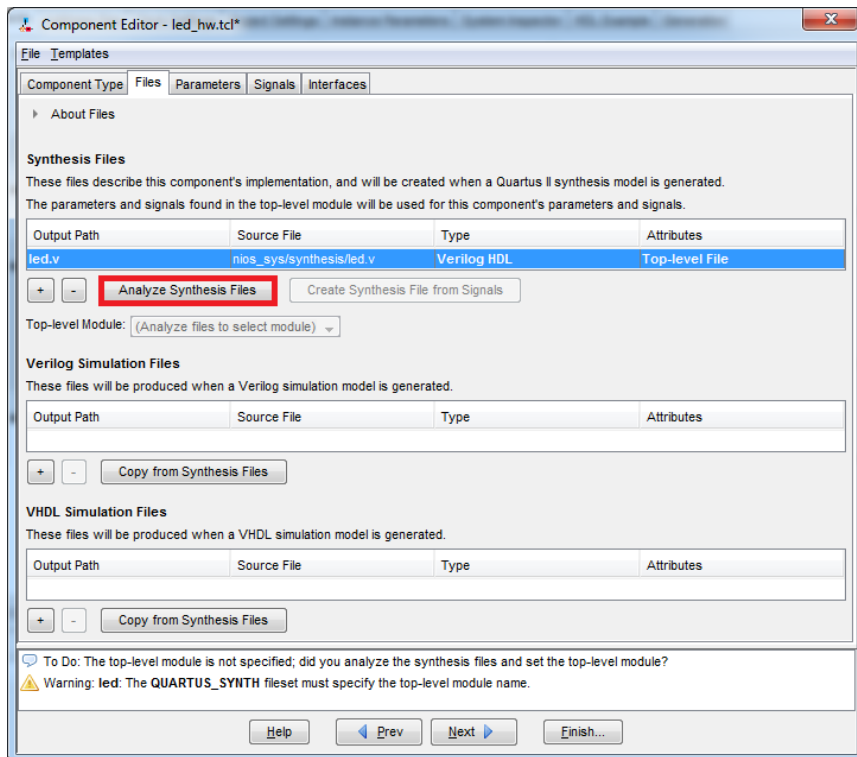
Chuyển sang tab “Files”, click vào dấu cộng trong mục “Synthesis Files”.



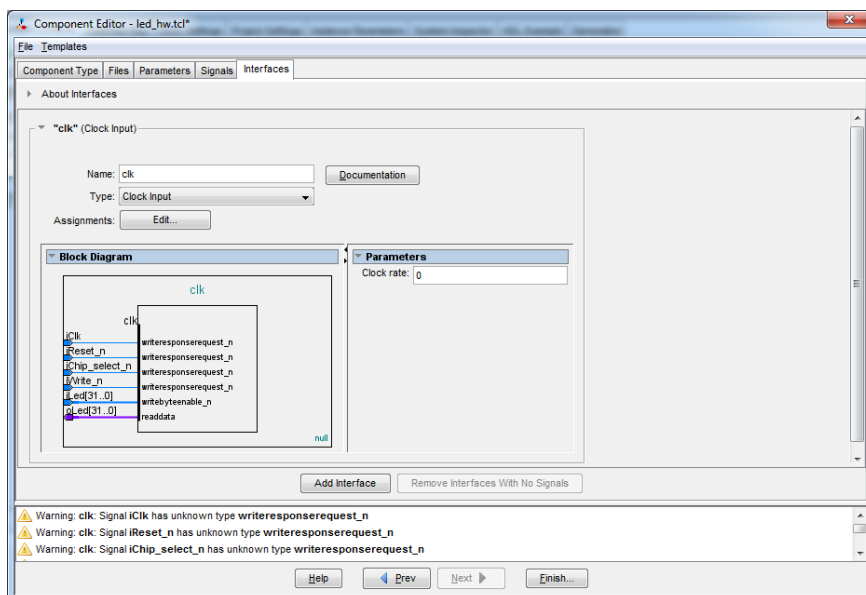
Dẫn đến file “led.v” đã thiết kế ở mục 2.1.1.



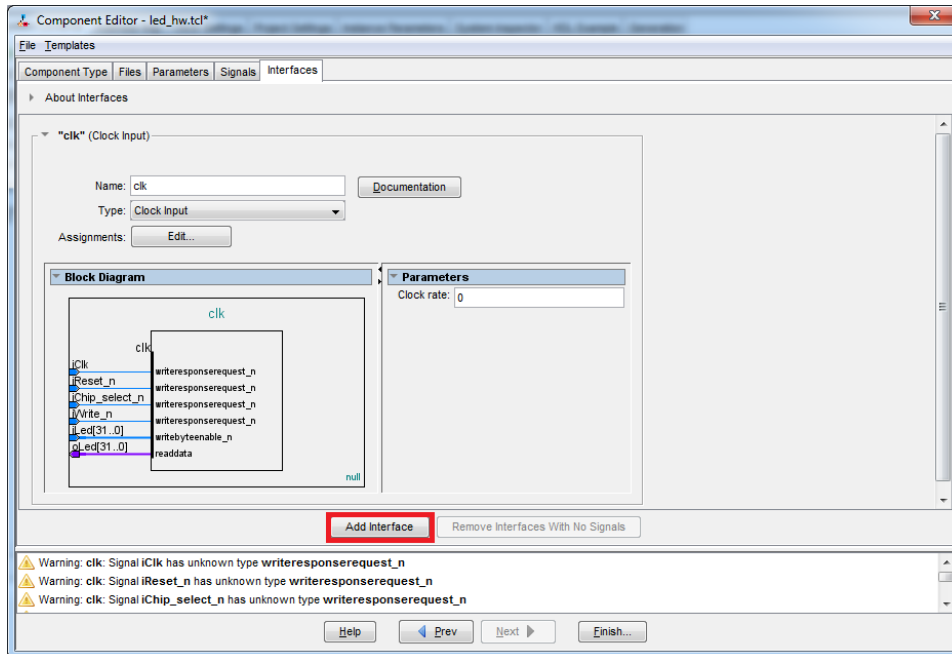
Click vào “Analyze Synthesis Files”.



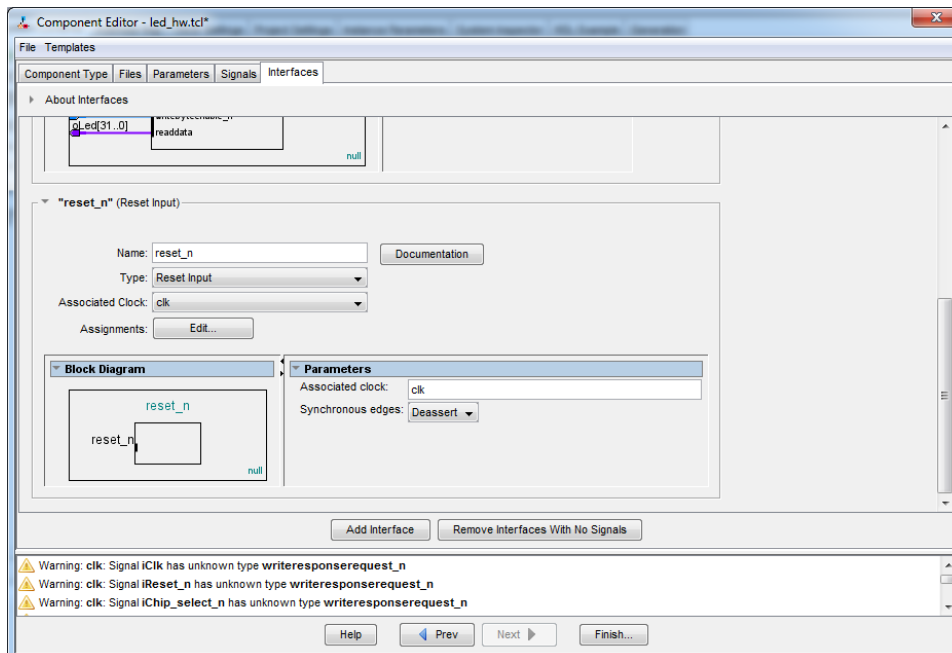
Chuyển qua tab “Interfaces”, thiết lập như bên dưới.



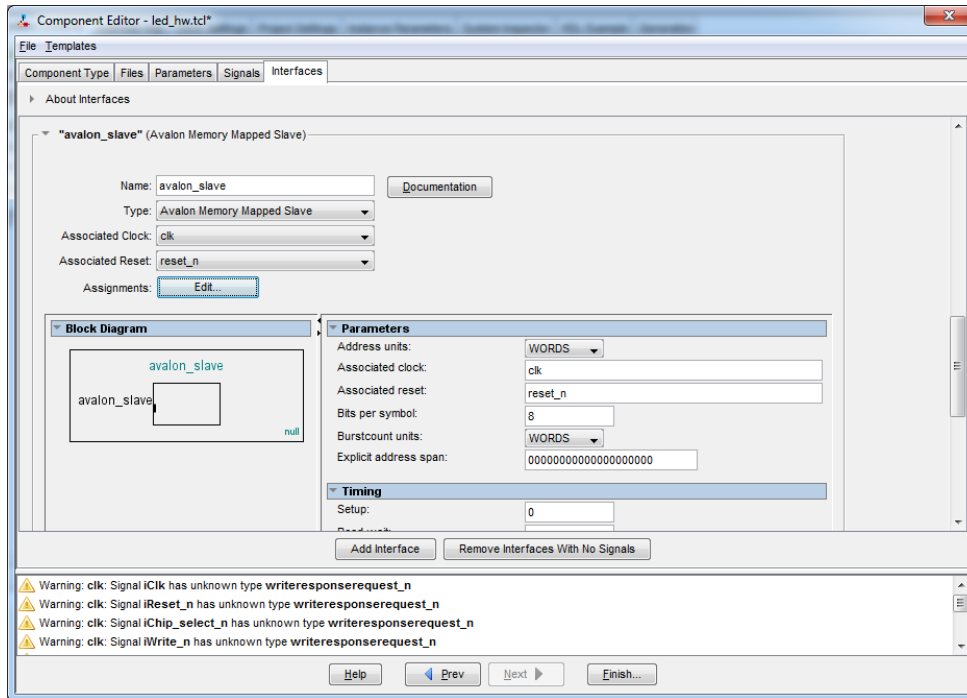
Click vào “Add Interface”.



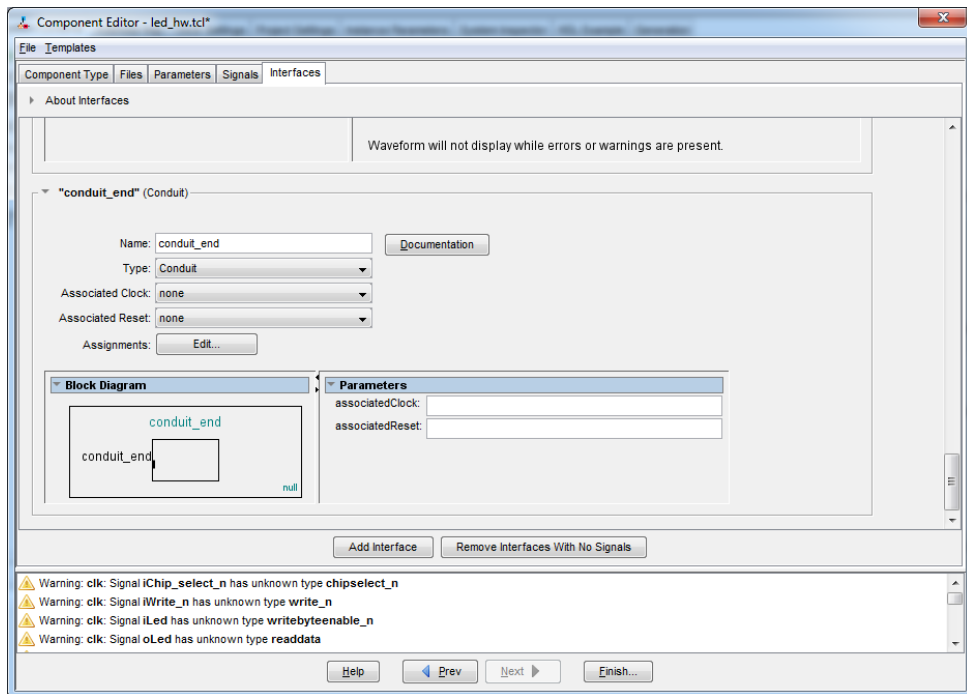
Thiết lập Interface mới như bên dưới.



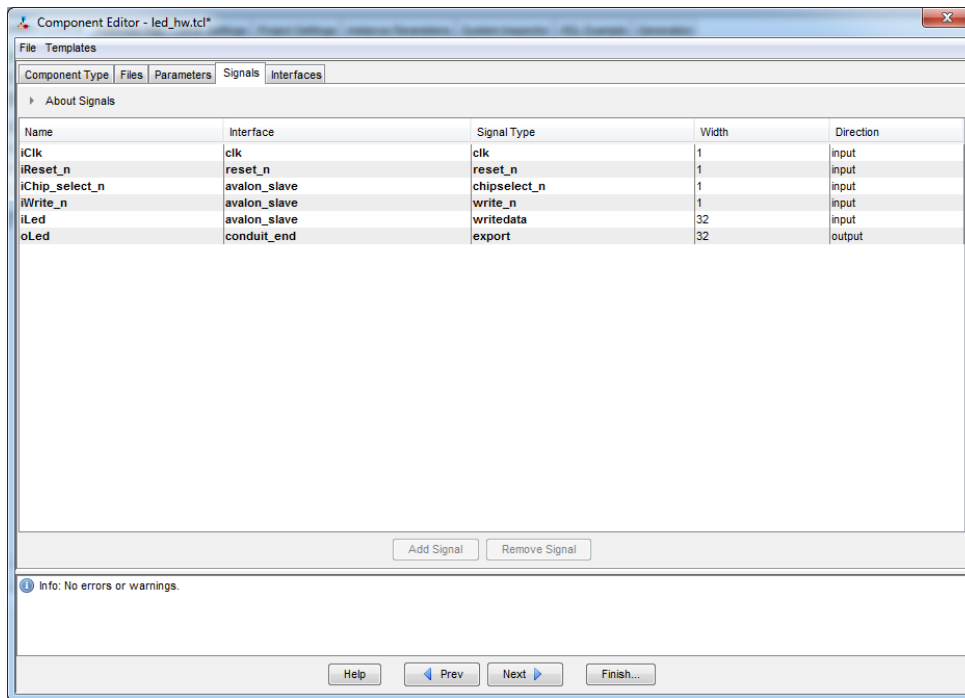
Click vào “Add Interface”, và thiết lập interface mới như bên dưới.



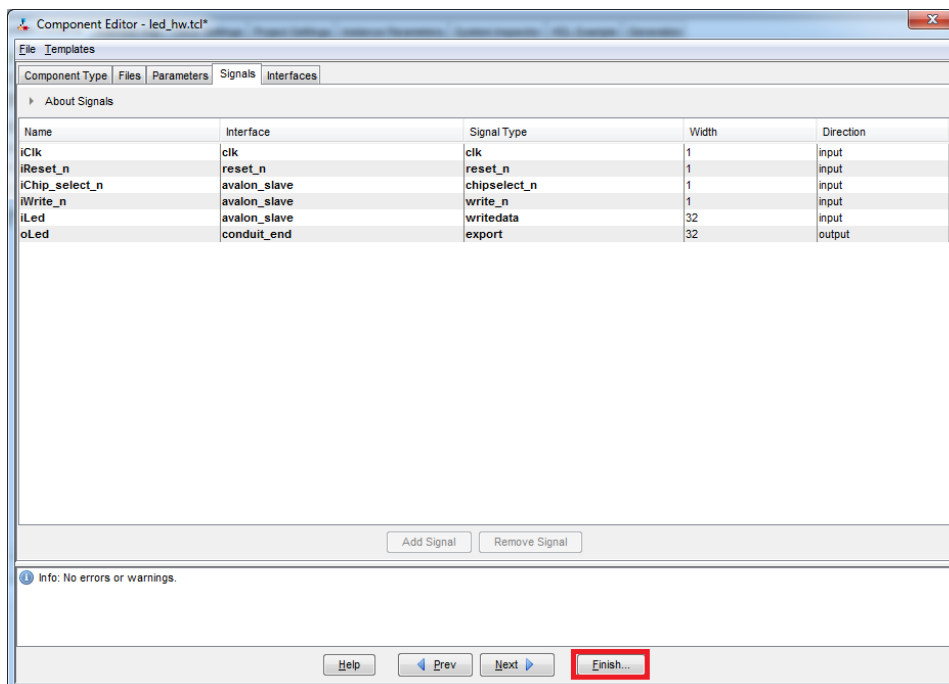
Click vào “Add Interface”, và thiết lập interface mới như bên dưới.



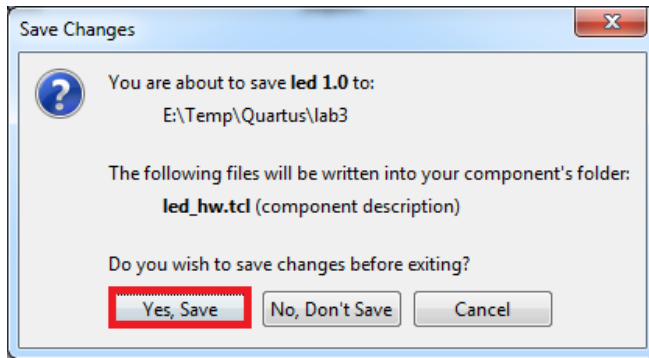
Chuyển qua tab “Signals”, Click vào từng tín hiệu ở cột “Interface” và “Signal Type” để đổi loại tín hiệu như bên dưới.



Click “Finish”.

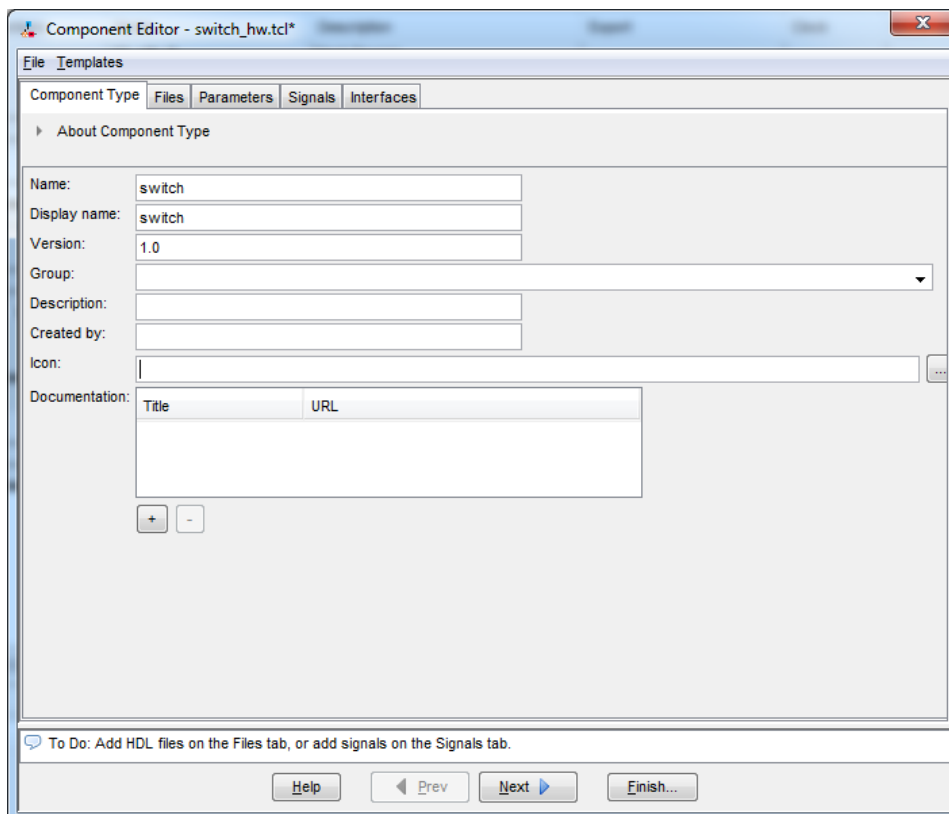




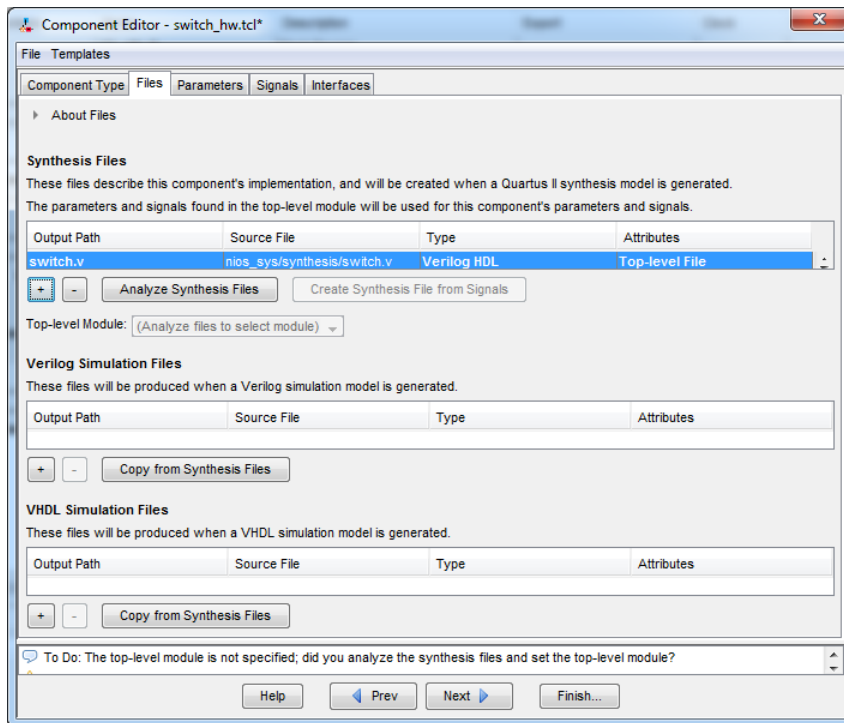


Tiếp theo, chúng ta sẽ thêm module “switch.v” đã thiết kế ở mục 2.1.1 vào thư viện của Qsys.

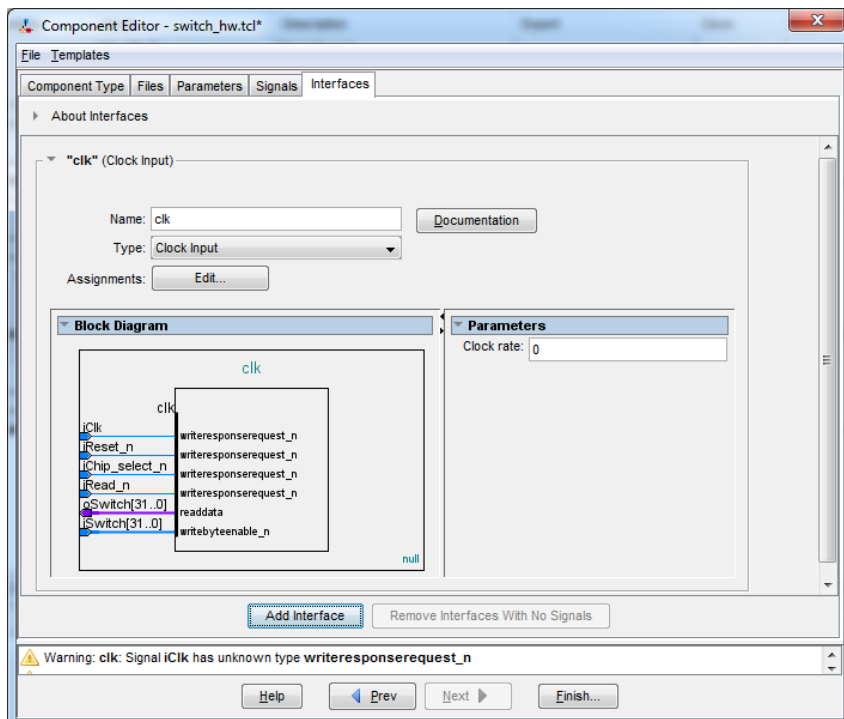
Double click vào “New component” ở cửa sổ thư viện. Đặt tên cho module ở tab “Component Type” như bên dưới.



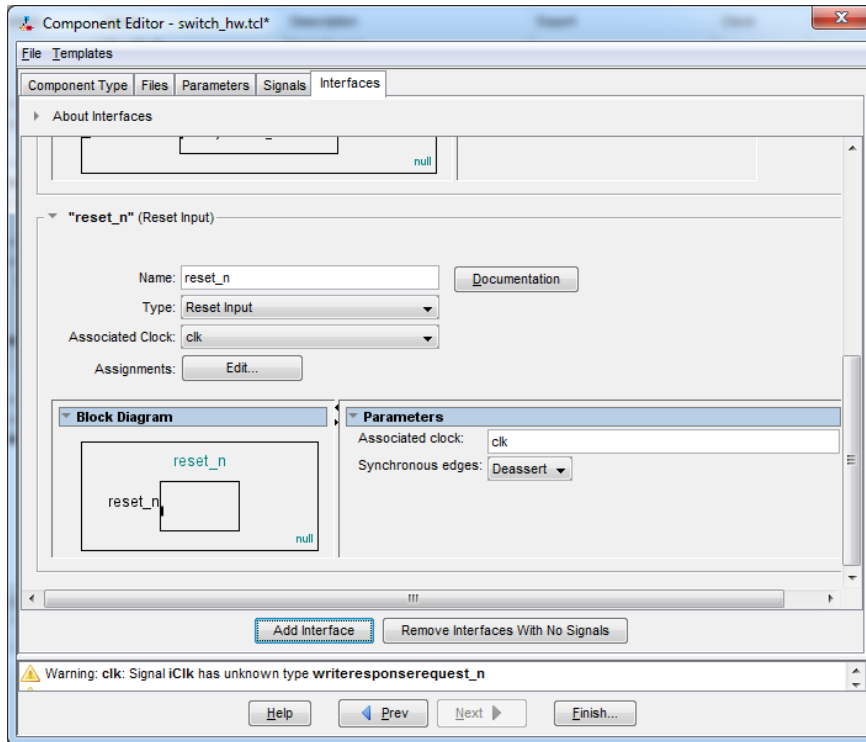
Chuyển sang tab “Files”, click vào dấu cộng trong mục “Synthesis Files”. Dẫn đến file “switch.v” đã thiết kế ở mục 2.1.1.



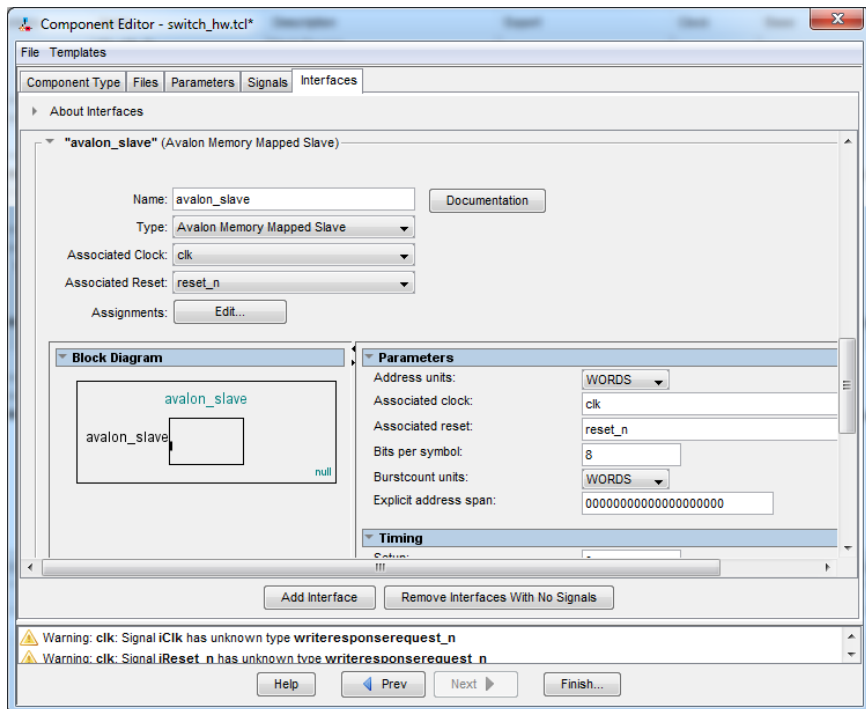
Click vào “Analyze Synthesis Files”. Chuyển qua tab “Interfaces”, thiết lập như bên dưới.



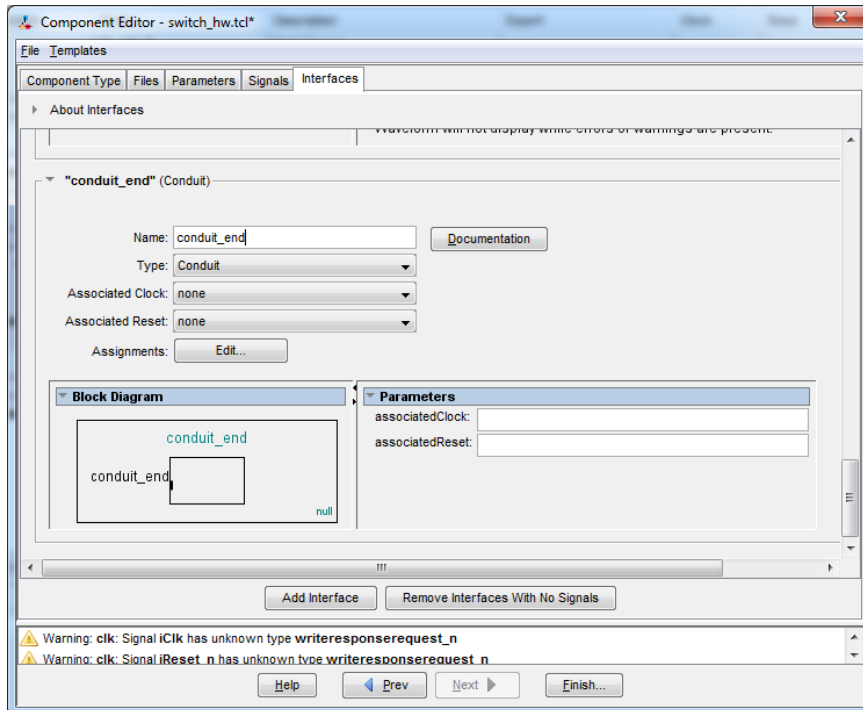
Click vào “Add Interface”, và thiết lập interface mới như bên dưới.



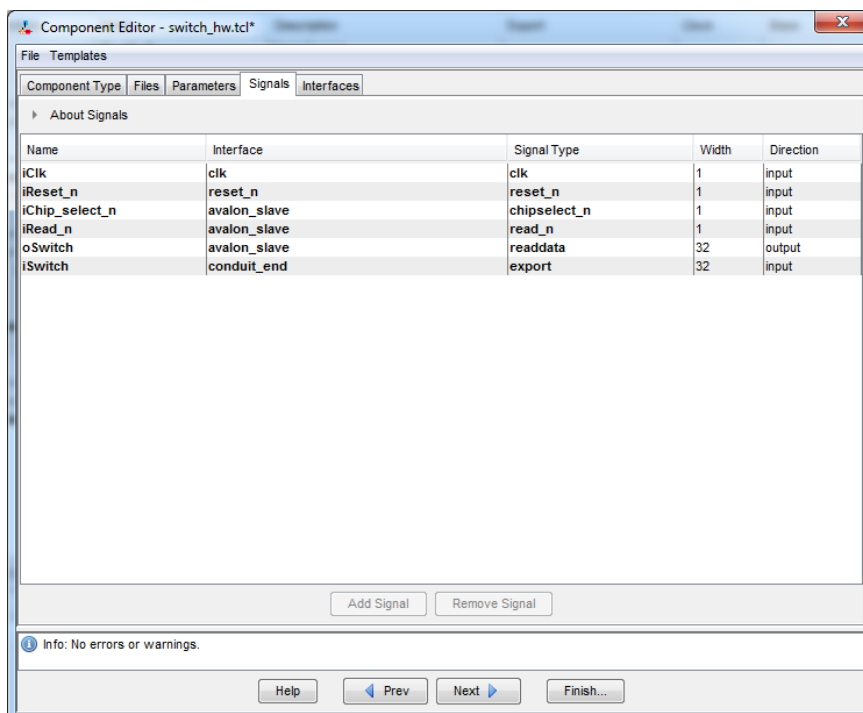
Click vào “Add Interface”, và thiết lập interface mới như bên dưới.



Click vào “Add Interface”, và thiết lập interface mới như bên dưới.

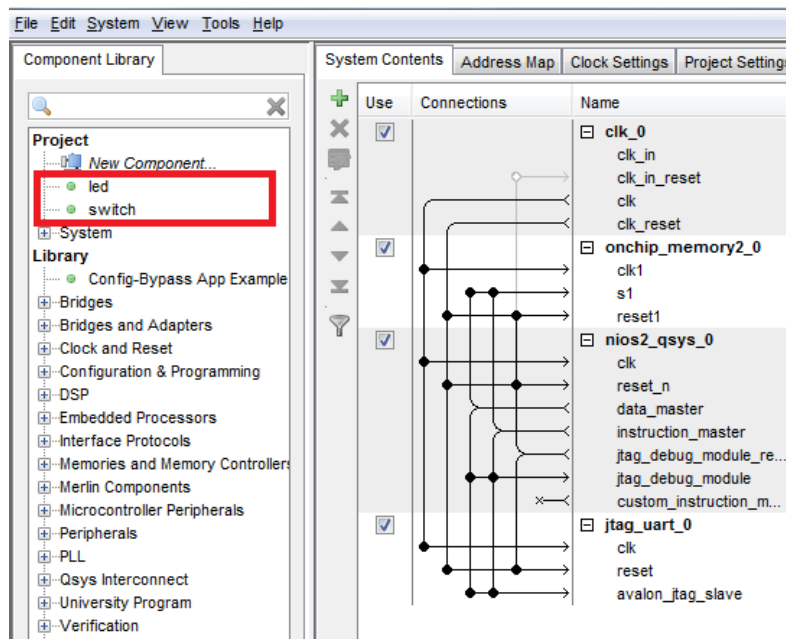


Chuyển qua tab “Signals”, Click vào từng tín hiệu ở cột “Interface” và “Signal Type” để đổi loại tín hiệu như bên dưới.

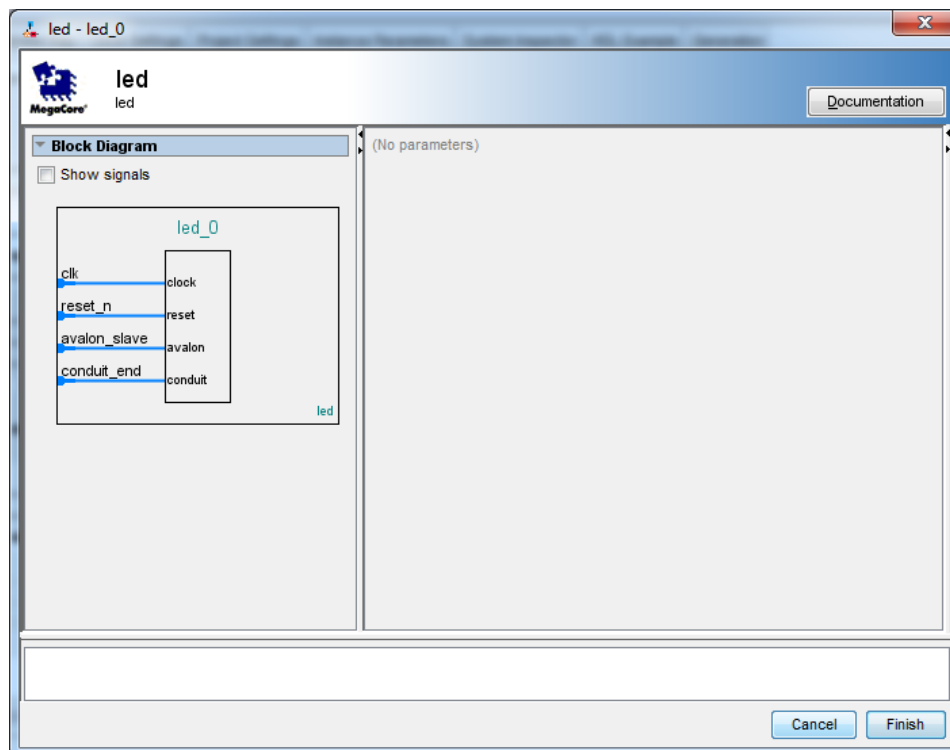


Click “Finish”, chọn “Yes, Save”.

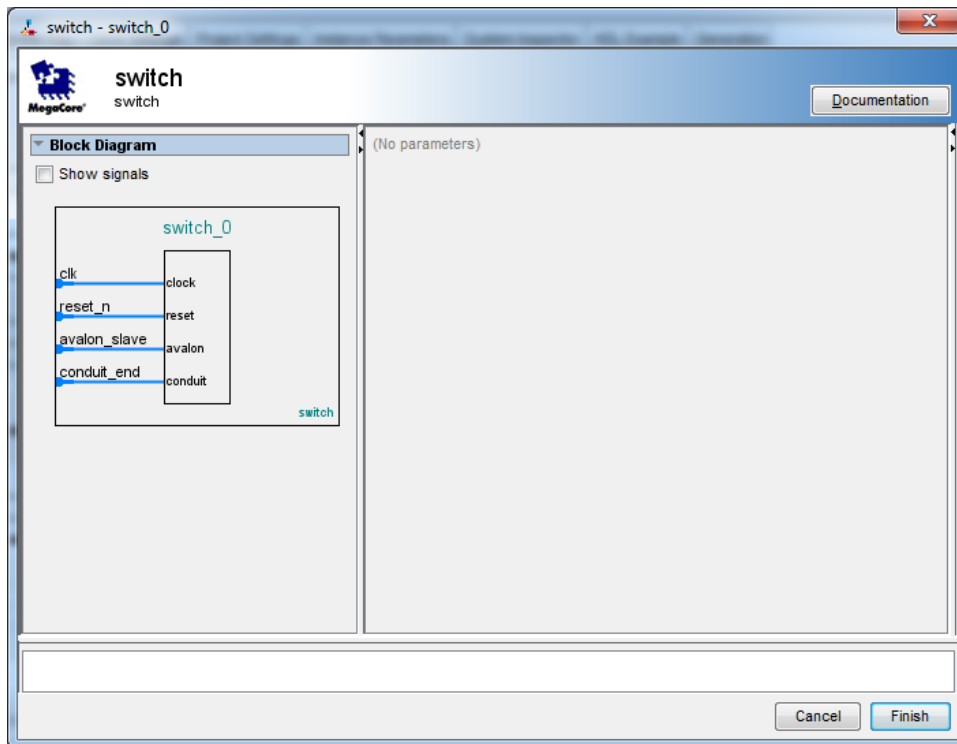
Lúc này, hai module “led.v” và “switch.v” đã được thêm vào thư viện Qsys.



Trong cửa sổ thư viện, chọn module “led” và click “Add” để thêm module này vào hệ thống Qsys. Click “Finish”.



Trong cửa sổ thư viện, chọn module “switch” và click “Add” để thêm module này vào hệ thống Qsys. Click “Finish”.



Kết nối các tín hiệu của hai module mới thêm vào hệ thống.

Connections	Name	Description	Export	Clock	Base	End	IRQ
	<b>clk_0</b>	Clock Source					
	clk_in	Clock Input	clk				
	clk_in_reset	Reset Input	reset				
	clk	Clock Output	Double-click to export	clk_0			
	clk_reset	Reset Output	Double-click to export				
	<b>onchip_memory2_0</b>	On-Chip Memory (RAM or ROM)					
	clk1	Clock Input	Double-click to export	clk_0			
	s1	Avalon Memory Mapped Slave	Double-click to export	[clk1]	0x0000_8000	0x0000_ffff	
	reset1	Reset Input	Double-click to export				
	<b>nios2_qsys_0</b>	Nios II Processor					
	clk	Clock Input	Double-click to export	clk_0			
	reset_n	Reset Input	Double-click to export	[clk]			
	data_master	Avalon Memory Mapped Master	Double-click to export	[clk]		IRQ 0	IRQ 31
	instruction_master	Avalon Memory Mapped Master	Double-click to export	[clk]			
	jtag_debug_module_re...	Reset Output	Double-click to export	[clk]			
	jtag_debug_module	Avalon Memory Mapped Slave	Double-click to export	[clk]	0x0001_0800	0x0001_0fff	
	custom_instruction_m...	Custom Instruction Master	Double-click to export				
	<b>jtag_uart_0</b>	JTAG UART					
	clk	Clock Input	Double-click to export	clk_0			
	reset	Reset Input	Double-click to export	[clk]			
	avalon_slave	Avalon Memory Mapped Slave	Double-click to export	[clk]	0x0001_1008	0x0001_100f	
	<b>led_0</b>	LED					
	clk	Clock Input	Double-click to export	clk_0			
	reset_n	Reset Input	Double-click to export	[clk]			
	avalon_slave	Avalon Memory Mapped Slave	Double-click to export	[clk]	0x0001_101c	0x0001_101f	
	conduit_end	Conduit	led conduit_end				
	<b>switch_0</b>	switch					
	clk	Clock Input	Double-click to export	clk_0			
	reset_n	Reset Input	Double-click to export	[clk]			
	avalon_slave	Avalon Memory Mapped Slave	Double-click to export	[clk]	0x0001_1018	0x0001_101b	
	conduit_end	Conduit	switch conduit_end				

Gán địa chỉ cho các module (System > Assign Base Addresses).

Hệ thống phần cứng đã hoàn thành, không còn thông báo lỗi. Save lại hệ thống với tên “nios\_sys”.

Chuyển sang tab “Generation”, click “Generate”

### 2.1.3. Tích hợp hệ thống Qsys vào project Quartus

Thực hiện tương tự như bài thực hành trước.

Tạo file top module, đặt tên là “lab3.v” với nội dung như sau.

```
1  module lab3(CLOCK_50,KEY,LEDR,SW);
2
3  input  CLOCK_50;
4  input  [0:0] KEY;
5  input  [15:0] SW;
6  output [15:0] LEDR;
7
8  nios_sys u0 (
9      .clk_clk          (CLOCK_50),
10     .reset_reset_n    (KEY[0]),
11     .switch_0_conduit_end_export ({16'b0,SW}),
12     .led_0_conduit_end_export  (LEDR));
13
14  endmodule
```

Build project Quartus và download hệ thống phần cứng xuống board

## 2.2. Lập trình phần mềm

Tạo và đặt tên project là “lab3”.

Thêm file “lab3.c” vào project “lab3”. File “lab3.c” có nội dung như bên dưới. Build project và download phần mềm xuống board. Đổi trạng thái của Switch trên board và quan sát LED.

```
main.c X
#include "system.h"
#include <io.h>

int main(){
    short tmp;

    while(1)
    {
        tmp = IORD(SWITCH_BASE,0);
        IOWR(LED_BASE,0,tmp);
    }
}
```

31

Đổi nội dung của file “labe.c” thành bên dưới. Build lại project và download phần mềm xuống board. Đổi trạng thái của Switch trên board và quan sát LED.

```
main.c ✕  
#include "system.h"  
  
int main(){  
    volatile short * LED_ptr = (short *) LED_BASE;  
    volatile short * SW_ptr = (short *) SWITCH_BASE;  
  
    short tmp;  
  
    while(1)  
    {  
        tmp = *SW_ptr;  
        *LED_ptr = tmp;  
    }  
}
```

b2



**BÁO CÁO THỰC HÀNH**  
**BÀI 3: GIAO TIẾP IO TỰ THIẾT KẾ**

Sinh viên:

.....

Lớp: ..... Nhóm: .....

Bài ~~2~~ Sử dụng các module IO tự thiết kế, xây dựng hệ thống hiển thị dữ liệu 4 bits từ Switches ra Led 7 đoạn HEX0 (thực hiện giải mã Led 7 đoạn bằng code C trên phần mềm).