

THỰC HÀNH SOC

Bài 2 – GIAO TIẾP I/O

1. MỤC ĐÍCH

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

- o Cấu trúc bộ PIO (Parallel Input/Output) của Altera (các thanh ghi, chức năng...).
- o Cách xây dựng hệ thống phần cứng bằng Qsys để thực hiện chức năng I/O đơn giản.
- o Cách xây dựng chương trình C trên công cụ Nios II – EDS để điều khiển I/O đơn giản.

2. NỘI DUNG

2.1. Lý thuyết module PIO

2.1.1. Giới thiệu

Module PIO của Altera có chức năng thực hiện các giao tiếp IO, thu nhận dữ liệu (từ switches, nút nhấn,...) hoặc xuất dữ liệu ra ngoài (leds đỏ, leds xanh, ...).

Module PIO có thể được cấu hình để hoạt động ở một trong bốn chế độ sau:

- Input: Thu nhận dữ liệu từ các tín hiệu bên ngoài
- Output: Xuất dữ liệu ra các tín hiệu bên ngoài
- Bidir: Vừa thu nhận dữ liệu, vừa xuất dữ liệu (không đồng thời)
- InOut: Vừa thu nhận dữ liệu, vừa xuất dữ liệu (đồng thời)

CPU sẽ điều khiển module PIO bằng cách đọc hoặc ghi lên các thanh ghi của nó.

2.1.2. Thanh ghi của PIO

Module PIO bao gồm 4 thanh ghi như bên dưới.

Offset	Register Name	R/W	(n-1)	...	2	1	0
0	data	read access	R	Data value currently on PIO inputs			
		write access	W	New value to drive on PIO outputs			
1	direction (1)	R/W		Individual direction control for each I/O port. A value of 0 sets the direction to input; 1 sets the direction to output.			
2	interruptmask (1), (2)	R/W		IRQ enable/disable for each input port. Setting a bit to 1 enables interrupts for the corresponding port.			
3	edgecapture (1), (2)	R/W		Edge detection for each input port.			

Chức năng của mỗi thanh ghi:

- Data Register: thanh ghi chứa dữ liệu thu vào (chế độ Input) hoặc xuất ra (chế độ Output).
- Direction Register: thanh ghi hướng của dữ liệu (chế độ Bidir).
- Interrupt-mask Register: thanh ghi cho phép ngắt (chế độ Input).
- Edge-capture Register: thanh ghi báo hiệu có cạnh lên/cạnh xuống của tín hiệu ngõ vào (chế độ Input).

Mỗi thanh ghi của PIO có thể được truy xuất như một vị trí nhớ trong bộ nhớ. Địa chỉ để truy xuất các thanh ghi sẽ là tổng của Base Address và Offset.

- Base Address: địa chỉ của khối PIO.
- Offset: Vị trí tương đối của các thanh ghi so với thanh ghi đầu tiên (Data Register).

2.2. Hệ thống phần cứng

2.2.1. Tạo project Quartus

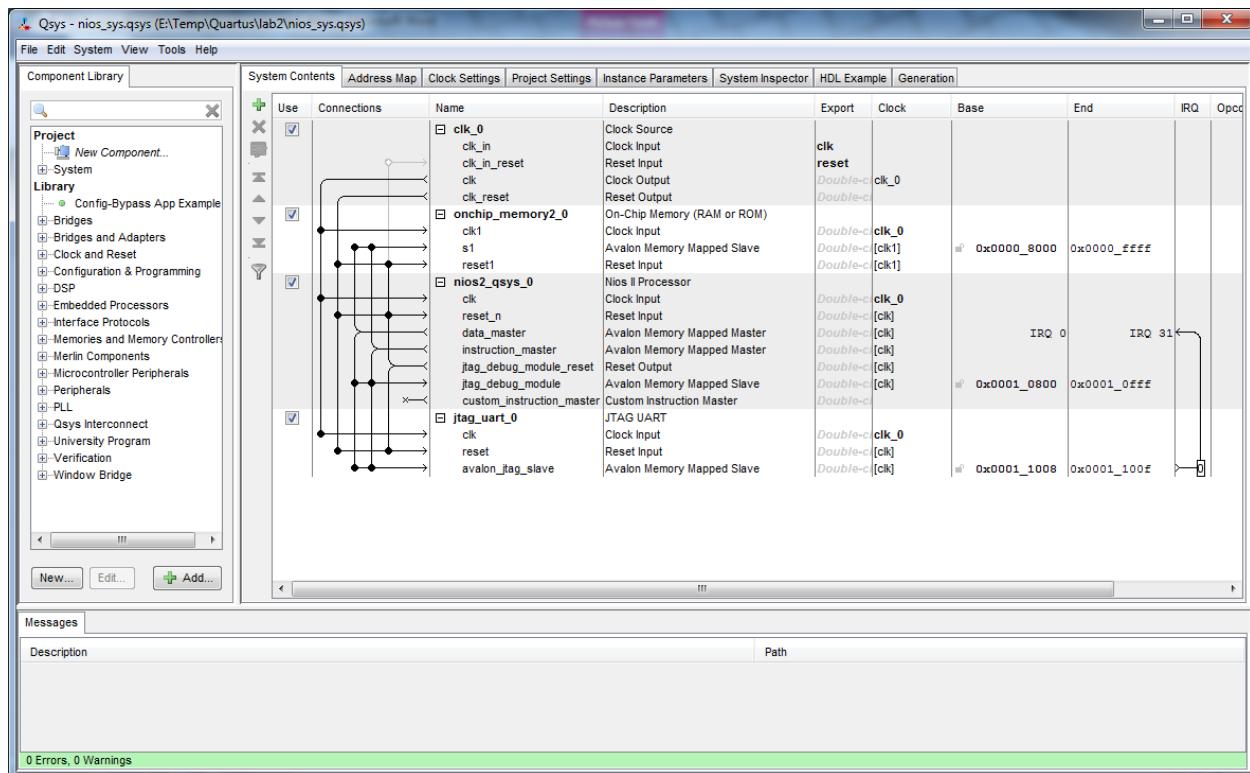
Tạo project Quartus tên là “lab2”. Lưu ý đường dẫn thư mục project không được có khoảng trắng.

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

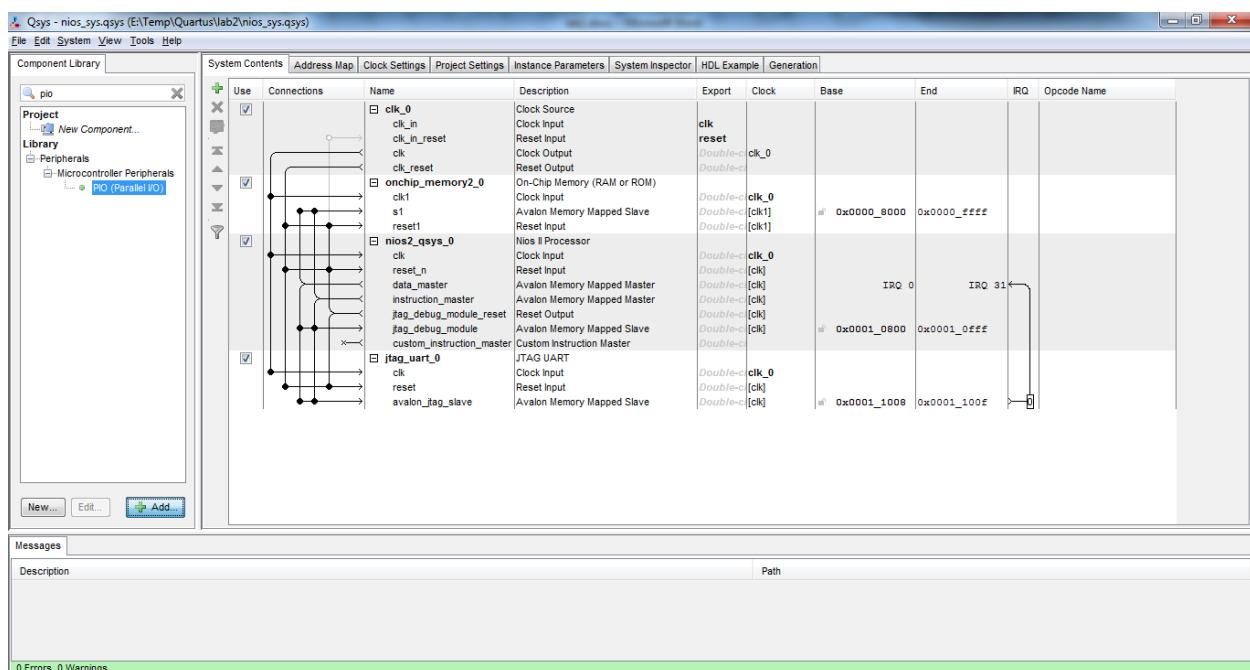
Nếu sử dụng board DE2, chọn Family là Cyclone II, device là EP2C35F672C6.

2.2.2. Xây dựng hệ thống Qsys

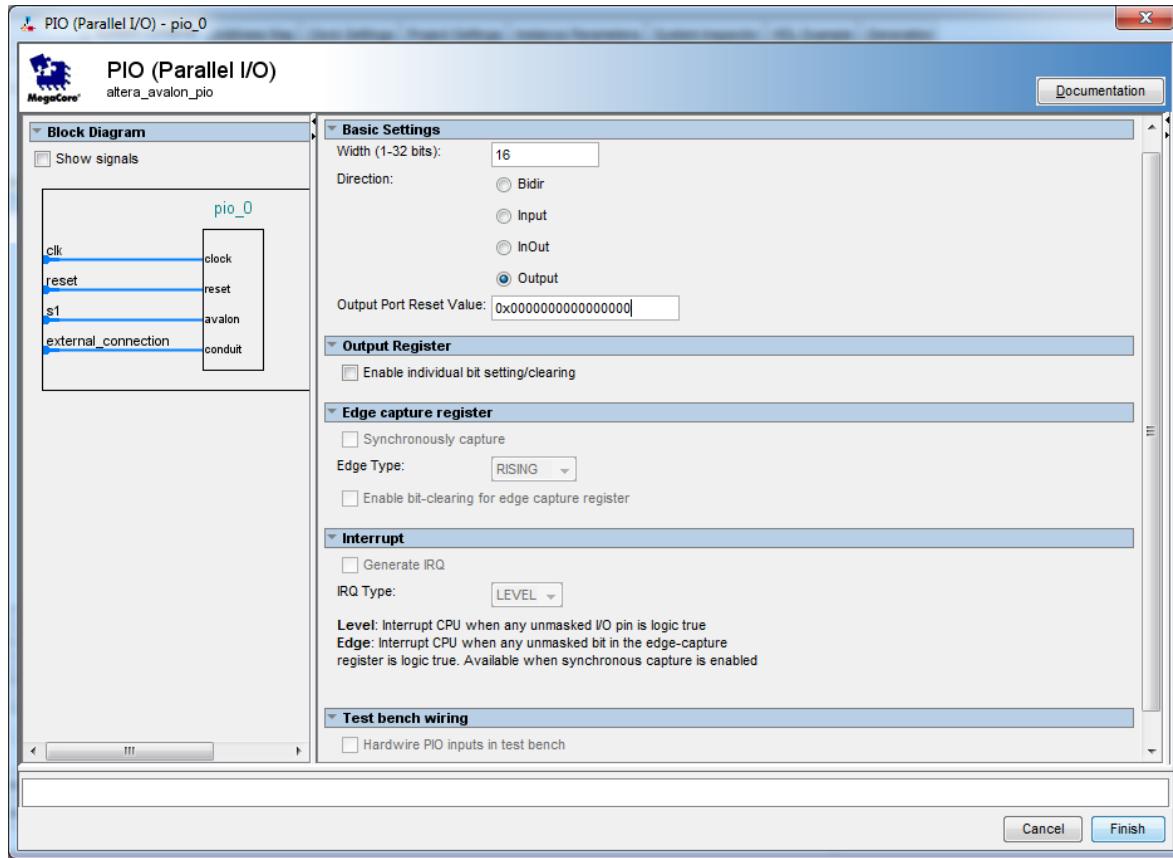
Xây dựng hệ thống phần cứng tương tự như bài thực hành trước, ta được như hình bên dưới.



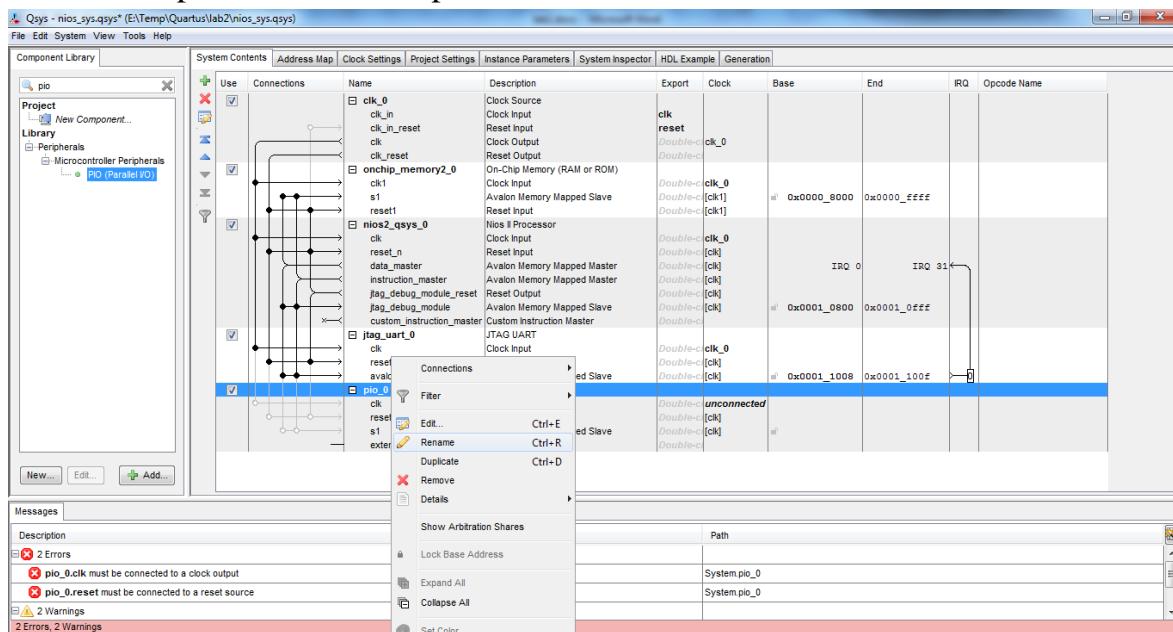
Gõ vào ô tìm kiếm module “PIO” và thêm vào hệ thống.



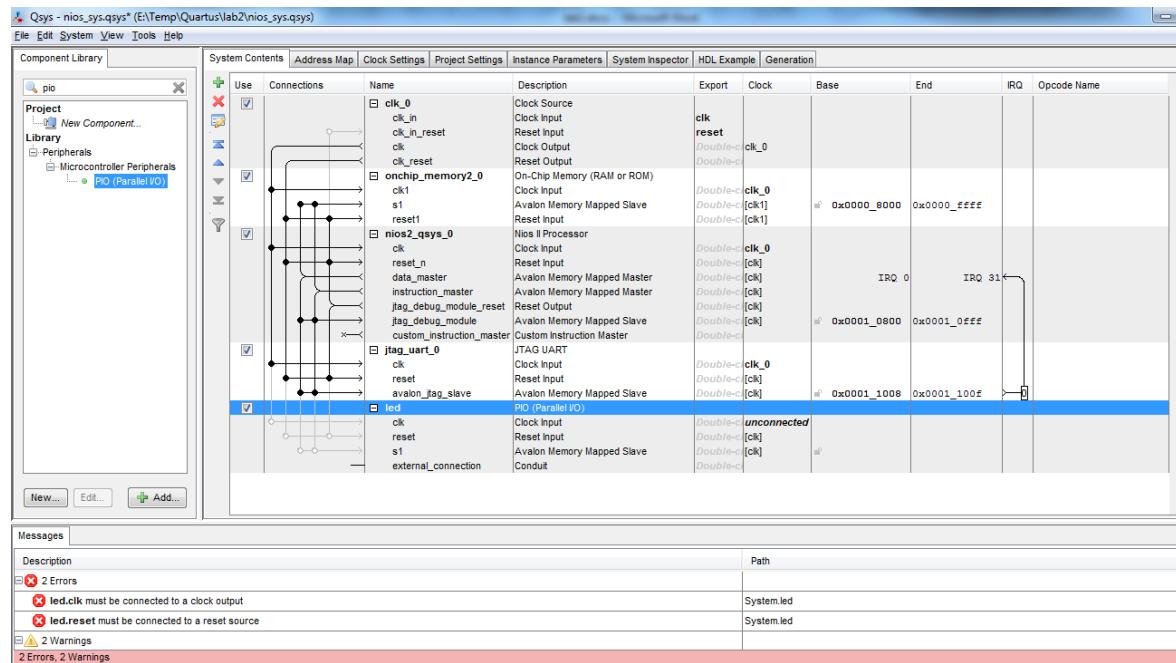
Cấu hình như bên dưới, click “Finish”.



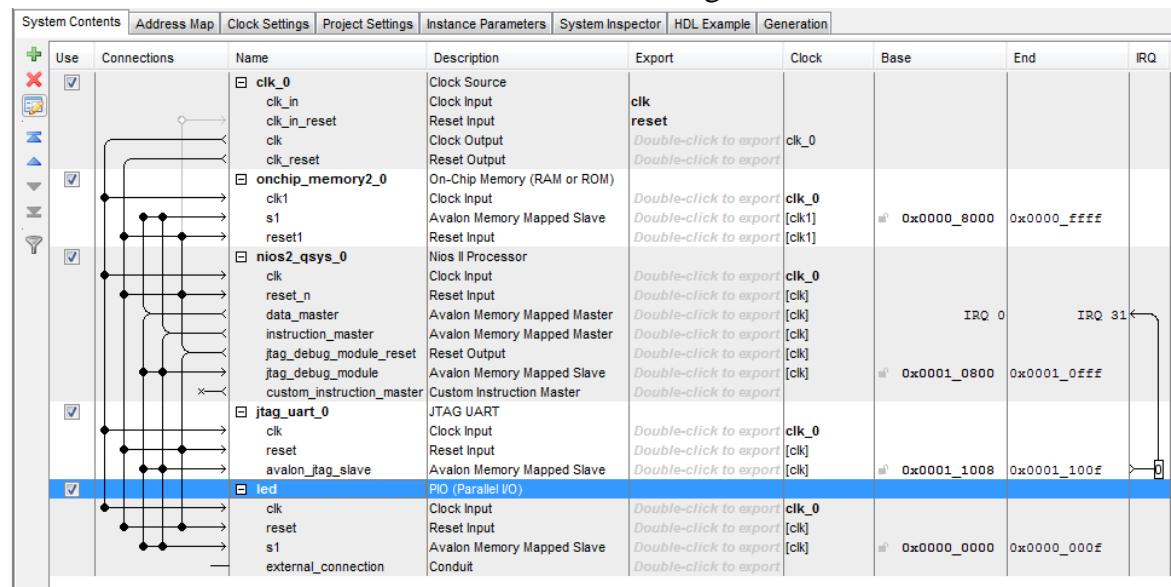
Click chuột phải vào module “pio_0”, chọn “Rename”.



Đổi tên module này thành “led”.



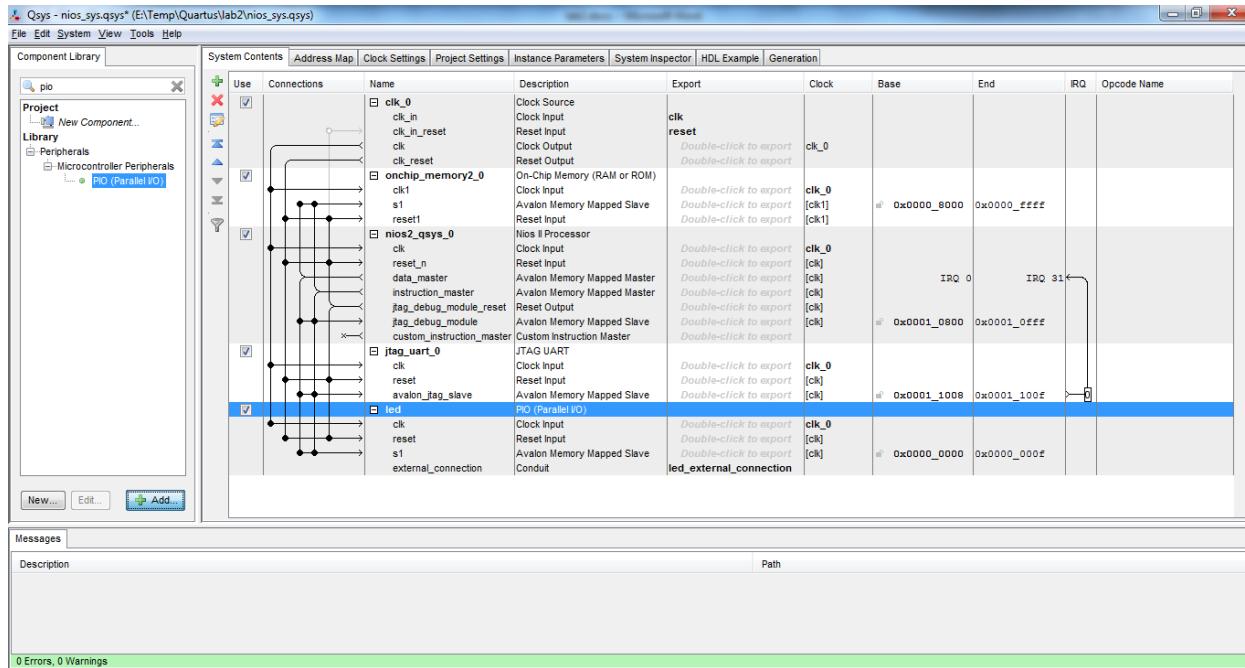
Kết nối các tín hiệu của module “led” vào hệ thống.



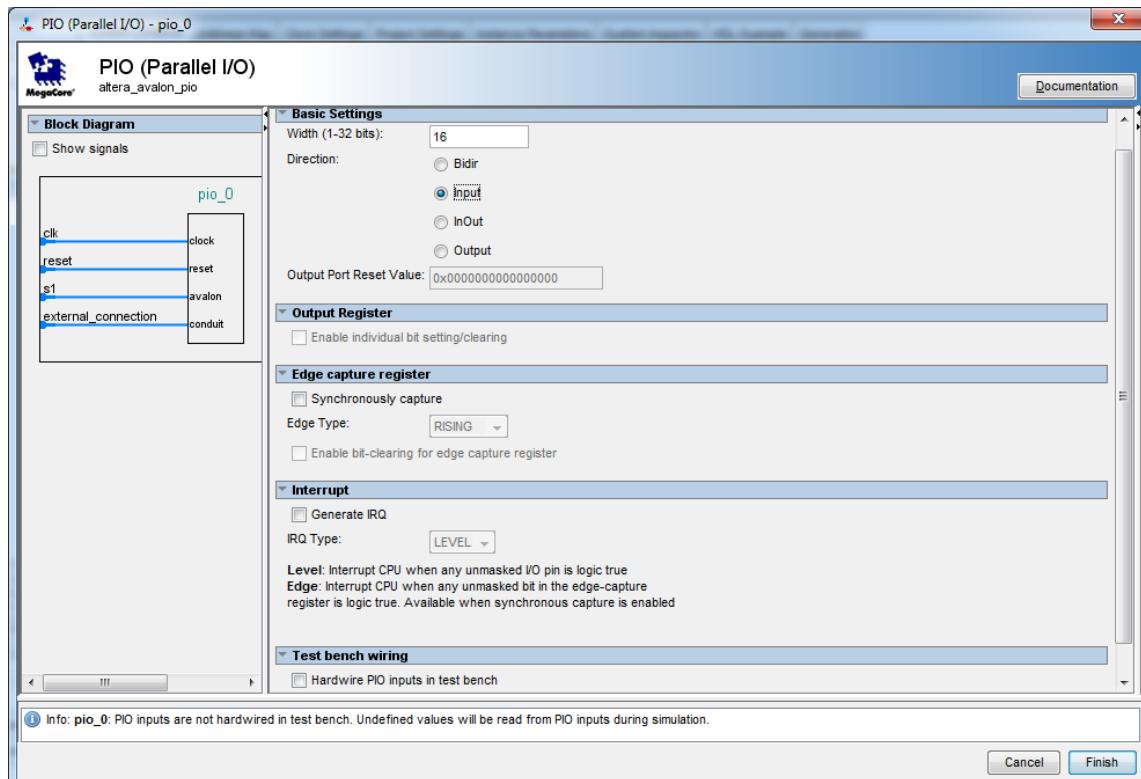
Tại interface “external_connection” của module “led”, double-click vào cột “Export”.

Use	Connections	Name	Description	Export	Clock	Base	End
		led	PIO (Parallel I/O)	Double-click to export	clk_0		
		clk	Clock Input	Double-click to export	[clk]		
		reset	Reset Input	Double-click to export	[clk]		
		s1	Avalon Memory Mapped Slave	Double-click to export	[clk]	0x0000_0000	0x0000_000f
		external_connection	Conduit	Double-click to export	led_external_connection		

Gõ vào ô tìm kiếm module “PIO” và thêm vào hệ thống.



Câu hình như bên dưới, click “Finish”.



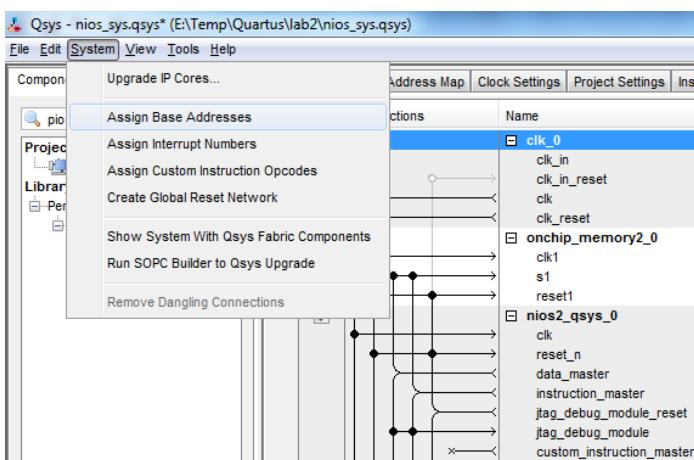
Đổi tên module PIO mới thêm thành “switch”. Kết nối các tín hiệu của module này vào hệ thống như bên dưới.

Use	Connections	Name	Description	Export	Clock	Base	End	IRQ
<input checked="" type="checkbox"/>		clk_0	Clock Source					
		clk_in	Clock Input					
		clk_in_reset	Reset Input					
		clk	Clock Output					
		clk_reset	Reset Output					
<input checked="" type="checkbox"/>		onchip_memory2_0	On-Chip Memory (RAM or ROM)					
		clk1	Clock Input					
		s1	Avalon Memory Mapped Slave					
		reset1	Reset Input					
<input checked="" type="checkbox"/>		nios2_qsys_0	Nios II Processor					
		clk	Clock Input					
		reset_n	Reset Input					
		data_master	Avalon Memory Mapped Master					
		instruction_master	Avalon Memory Mapped Master					
		jtag_debug_module_reset	Reset Output					
		jtag_debug_module	Avalon Memory Mapped Slave					
		custom_instruction_master	Custom Instruction Master					
<input checked="" type="checkbox"/>		jtag_uart_0	JTAG UART					
		clk	Clock Input					
		reset	Reset Input					
		avalon_jtag_slave	Avalon Memory Mapped Slave					
<input checked="" type="checkbox"/>		led	PIO (Parallel I/O)					
		clk	Clock Input					
		reset	Reset Input					
		s1	Avalon Memory Mapped Slave					
		external_connection	Conduit					
<input checked="" type="checkbox"/>		switch	PIO (Parallel I/O)					
		clk	Clock Input					
		reset	Reset Input					
		s1	Avalon Memory Mapped Slave					
		external_connection	Conduit					

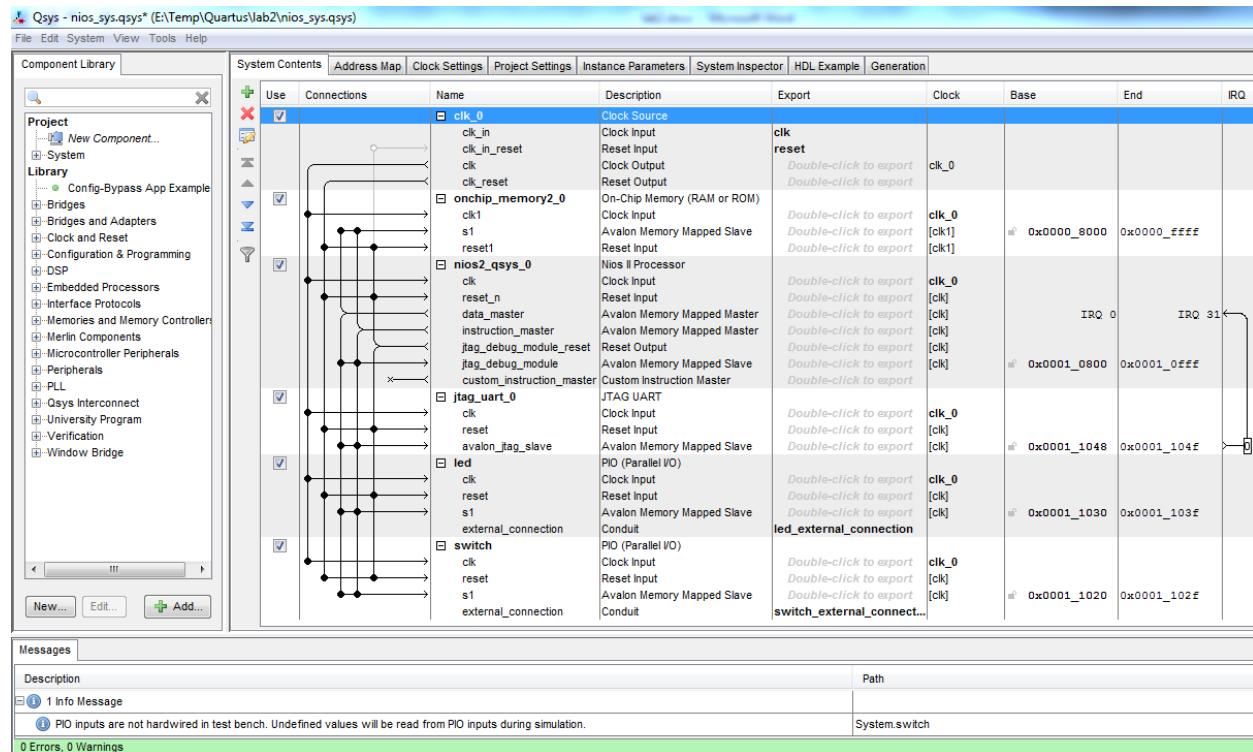
Tại interface “external_connection” của module “switch”, double-click vào cột “Export”.

Use	Connections	Name	Description	Export	Clock	Base	End
<input checked="" type="checkbox"/>		switch	PIO (Parallel I/O)				
		clk	Clock Input				
		reset	Reset Input				
		s1	Avalon Memory Mapped Slave				
		external_connection	Conduit	switch_external_connect..			

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”

2.2.3. Tích hợp hệ thống Qsys vào project Quatus

Thực hiện tương tự như bài thực hành trước.

Tạo file top module, đặt tên là “lab2.v” với nội dung như sau.

```

nios_sys/synthesis/lab2.v
1  module lab2(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   .led_external_connection_export(LEDR),
12   .switch_external_connection_export(SW)
13 );
14
15 endmodule
16

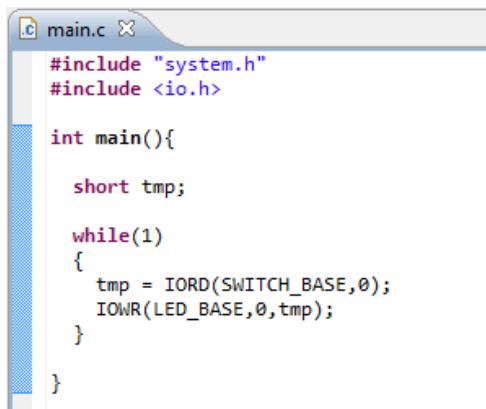
```

Build project Quartus và download hệ thống phần cứng xuống board

2.3. Lập trình phần mềm

Thực hiện tương tự như bài thực hành trước. Đặt tên project là “lab2”.

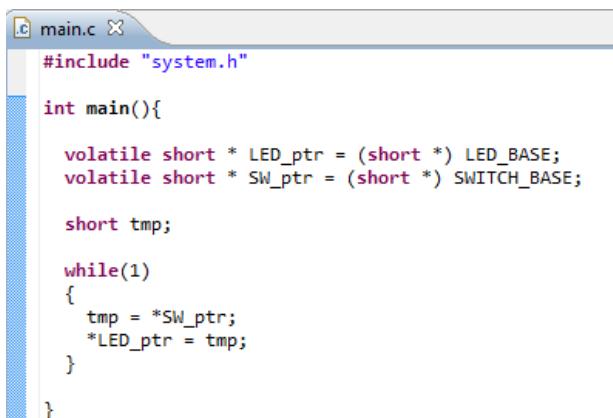
Thêm file “lab2.c” vào project “lab2”. File “lab2.c” có nội dung như bên dưới. Build project và dowload phần mềm xuống board. Đổi trạng thái của Switch trên board và quan sát LED.



```
#include "system.h"
#include <io.h>

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

Đổi nội dung của file “lab2.c” thành bên dưới. Build lại project và dowload phần mềm xuống board. Đổi trạng thái của Switch trên board và quan sát LED.



```
#include "system.h"
#include <io.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;
    }
}
```

BÁO CÁO THỰC HÀNH

BÀI 2: GIAO TIẾP IO

Sinh viên:

.....

Lớp: Nhóm:

Bài 1: Tiến hành xây dựng hệ thống hiển thị dữ liệu 4 bits từ Switches (PIO 4 bits) ra Led 7 đoạn HEX0 (thực hiện giải mã Led 7 đoạn bằng code C trên phần mềm).