Lab 1: SWITCHES, LED.

I. Mục tiêu:

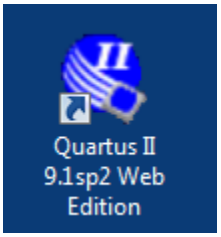
Thiết kế hệ thống với Nios II Processor thực hiện công việc sau: Sử dụng 8 Switches để bật tắt 8 LEDR.

II. Tạo New Project Quartus II:

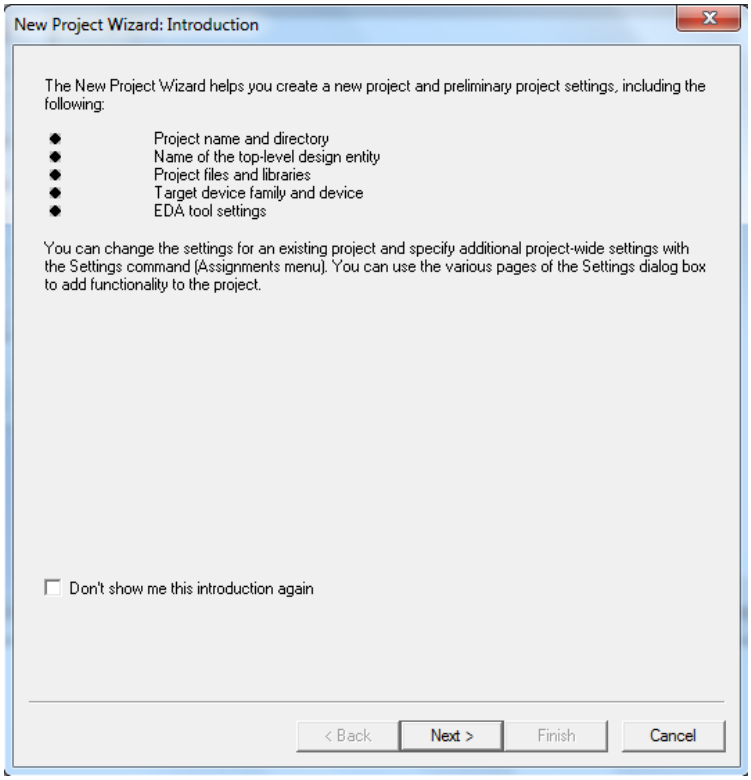
Thực hiện theo thứ tự các bước sau:

1. Tạo 1 file mới New folder với tên **lab1.**

2. Double click vào shortcut Quartus II trên Destop để mở giao diện làm việc.



3. Trên Quartus II menu bar chọn File -> New Project Wizard. Thiết lập các tùy chọn như bên dưới.

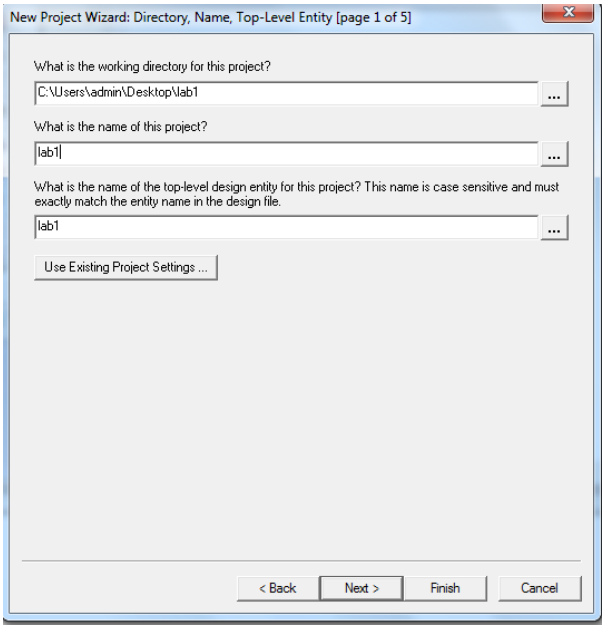


4. Click Next.

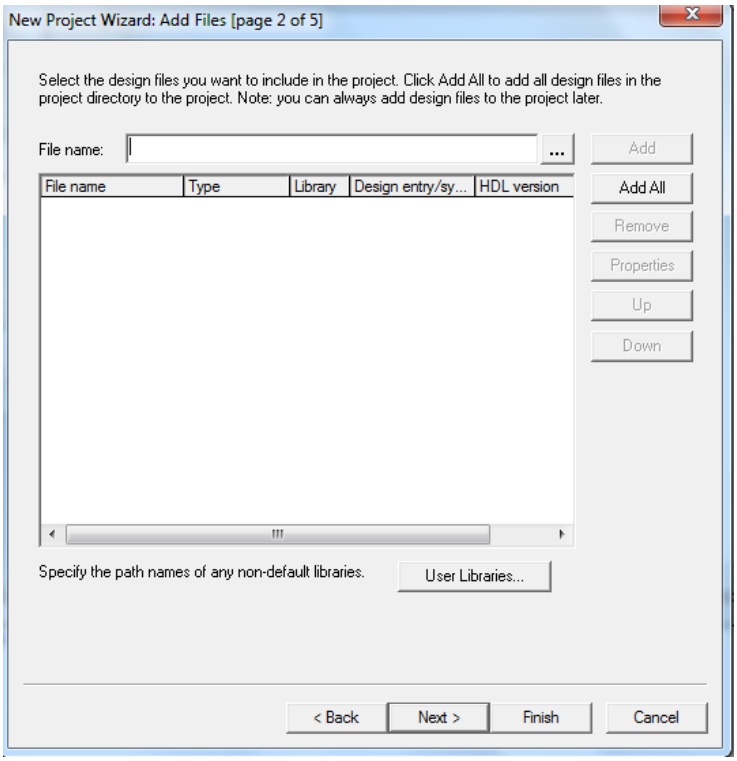
5. Trong khung thứ nhất chọn đường dẫn vào thư mục vừa tạo mang tên **lab1.**

   Tên project phải trùng với tên thư mục là **lab1.**

   Click Next



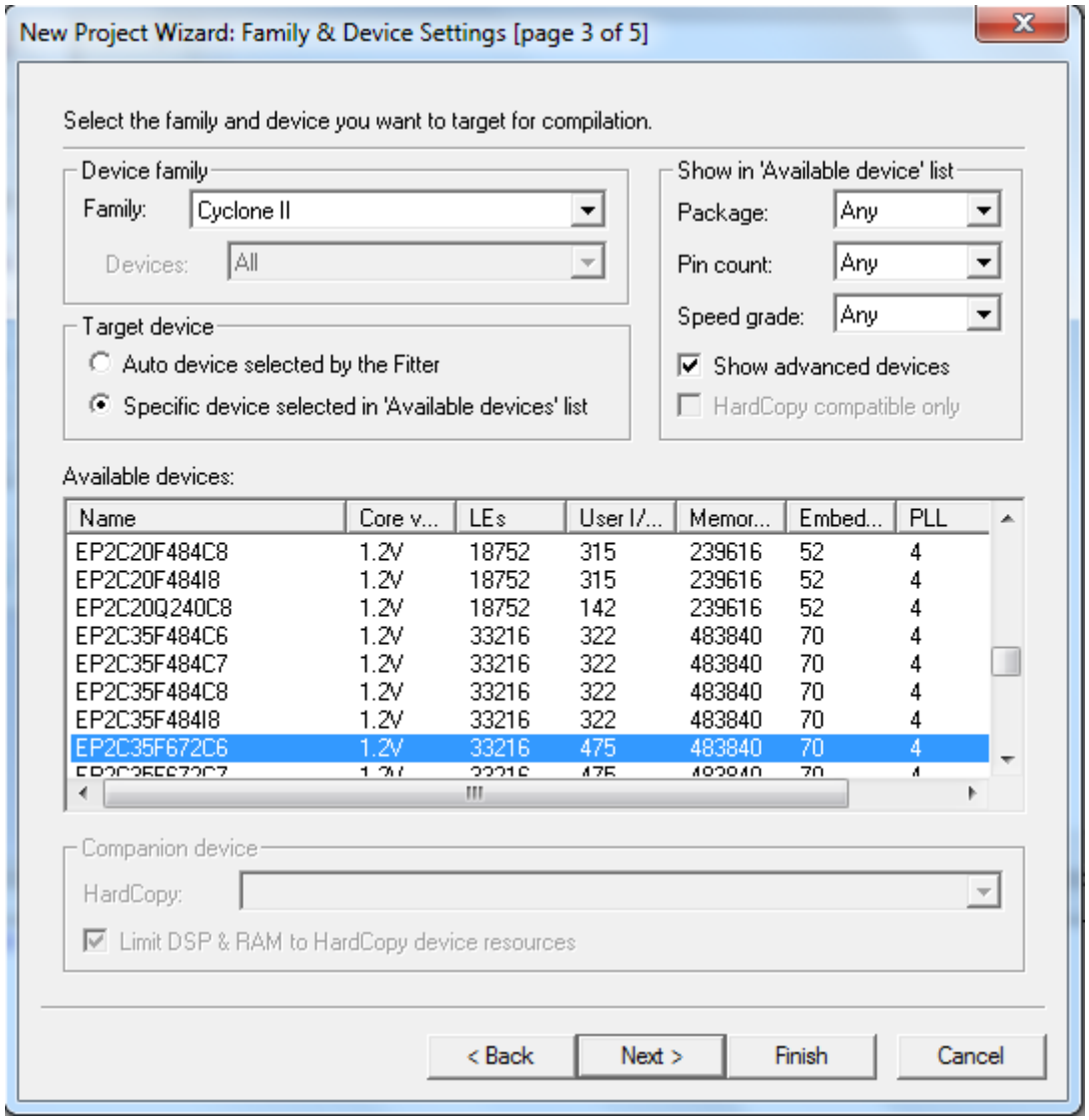
6. Click Next



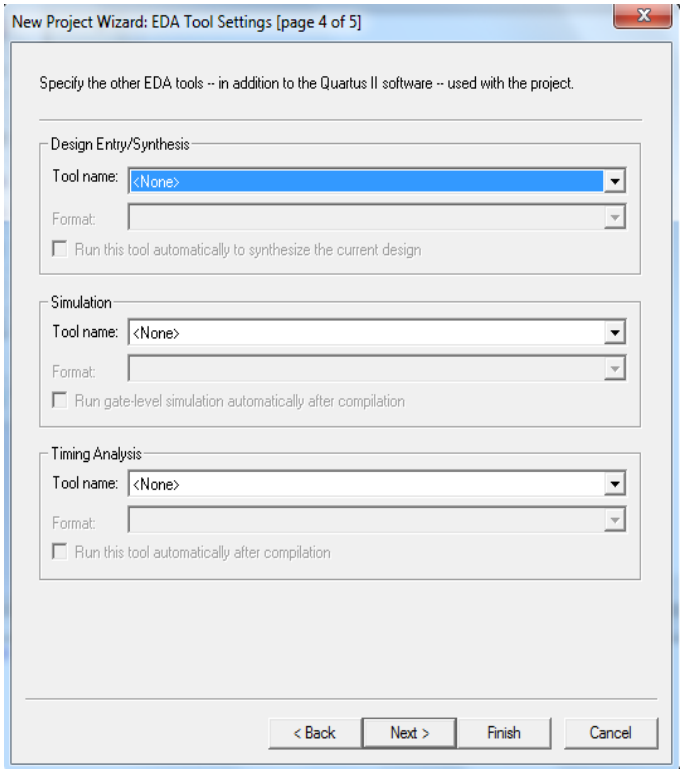
7. Chọn Cyclone II.

Available devices: **Chọn EP2C35F672C6.**

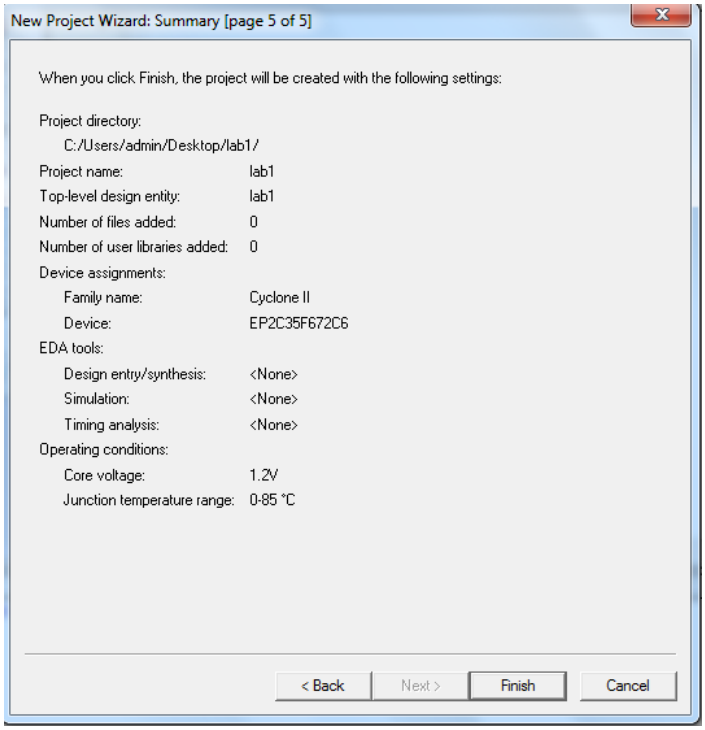
Click Next


New Project Wizard: Family & Device Settings [page 3 of 5]
Select the family and device you want to target for compilation.
Device family
Family: Cyclone II
Devices: All
Target device
Auto device selected by the Fitter
Specific device selected in 'Available devices' list
Show in 'Available device' list
Package: Any
Pin count: Any
Speed grade: Any
Show advanced devices
HardCopy compatible only
Available devices:
Name Core v... LEs User I/... Memor... Embed... PLL
EP2C20F484C8 1.2V 18752 315 239616 52 4
EP2C20F484I8 1.2V 18752 315 239616 52 4
EP2C20Q240C8 1.2V 18752 142 239616 52 4
EP2C35F484C6 1.2V 33216 322 483840 70 4
EP2C35F484C7 1.2V 33216 322 483840 70 4
EP2C35F484C8 1.2V 33216 322 483840 70 4
EP2C35F484I8 1.2V 33216 322 483840 70 4
EP2C35F672C6 1.2V 33216 475 483840 70 4
Companion device
HardCopy:
Limit DSP & RAM to HardCopy device resources
< Back
Next >
Finish
Cancel

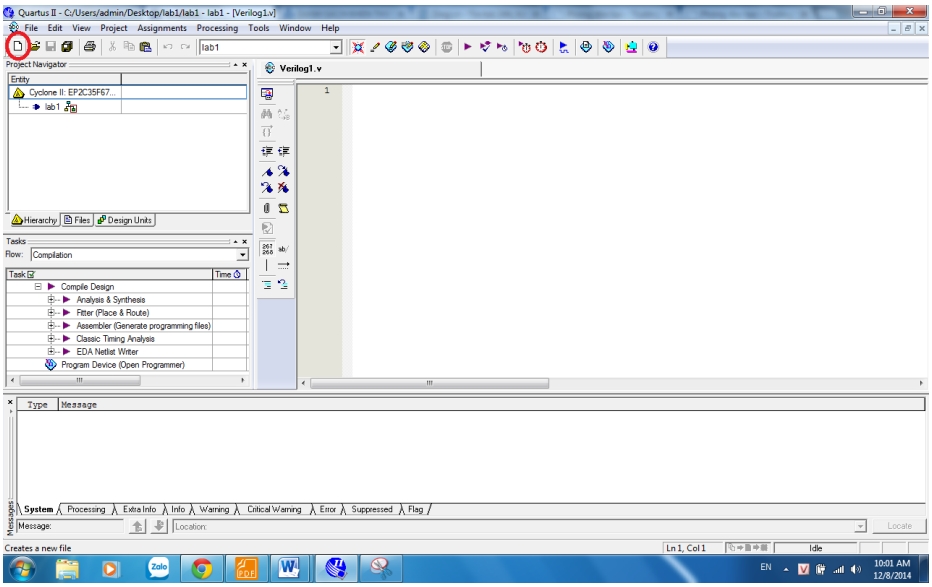
8. Click Next.



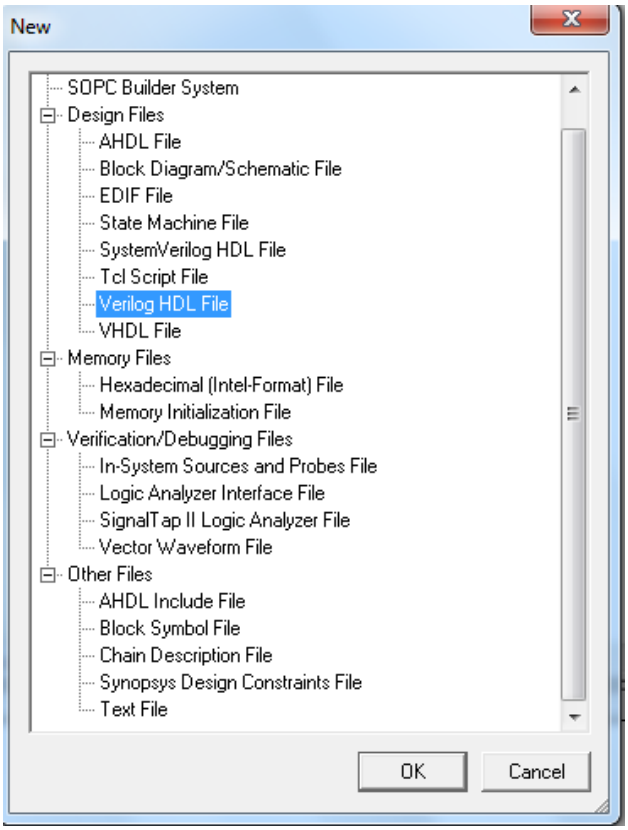
9. Click Finish.



10. Click New

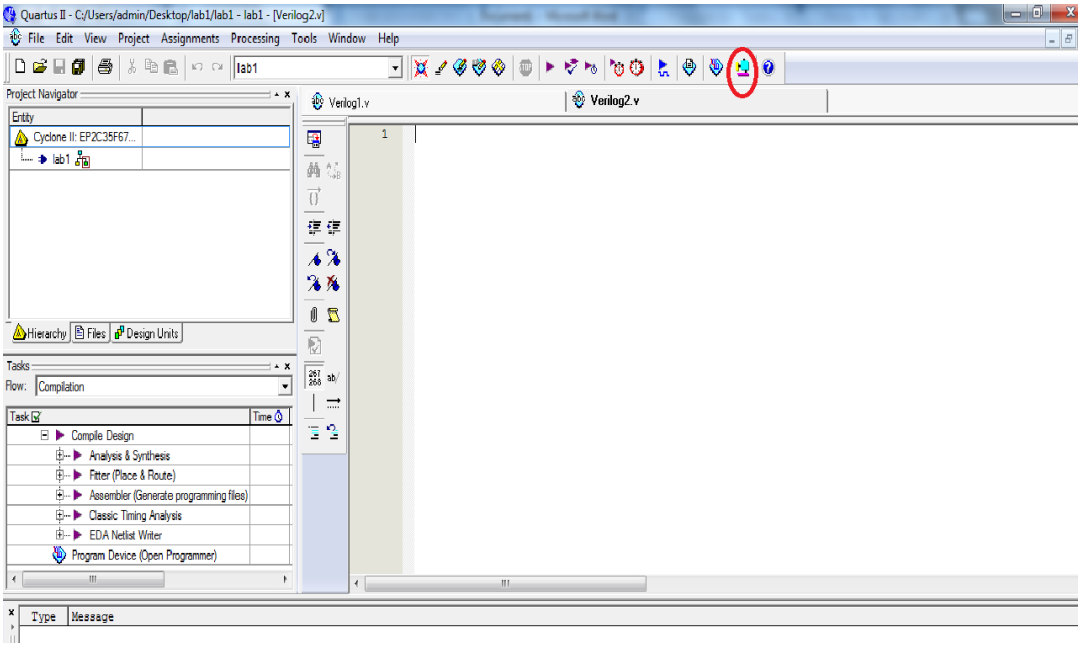


11. Chọn **Verilog HDL File -> click OK**

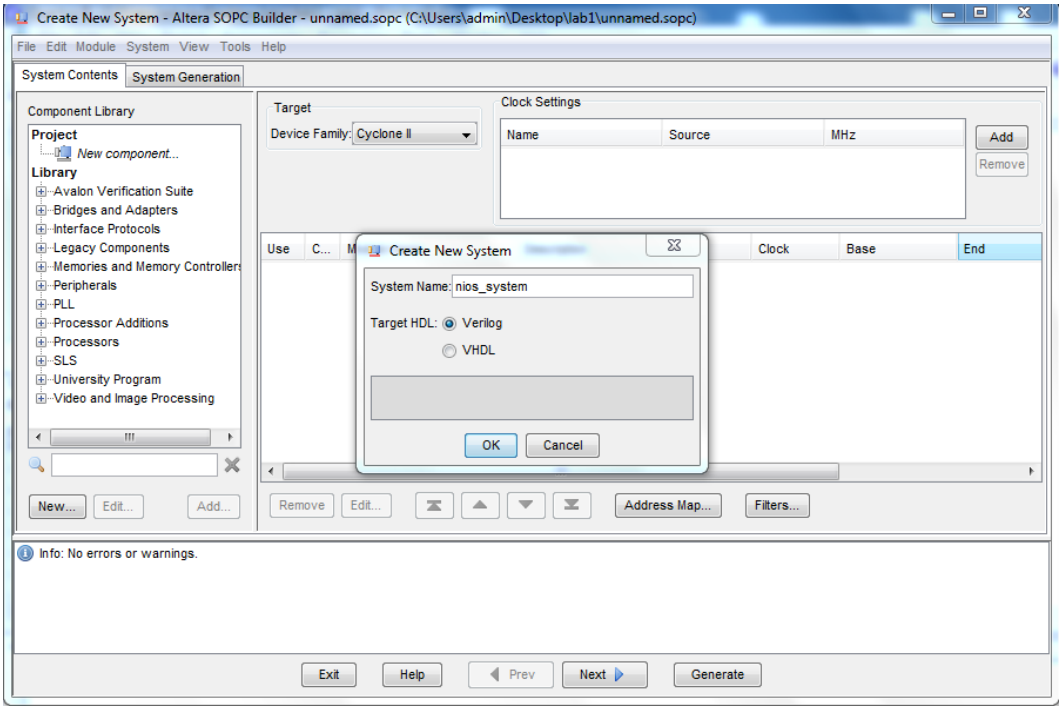


III. TẠO SOPC:

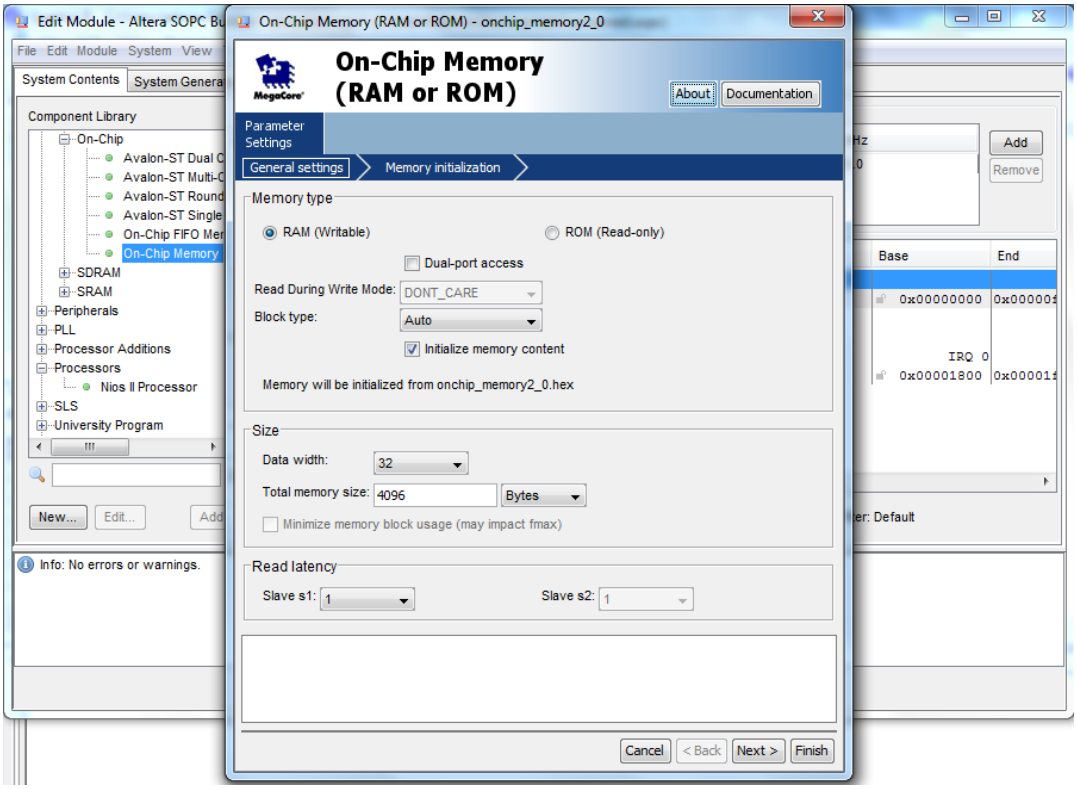
1. Click **SOPC Builder** để tạo file SOPC.



2. System name: **nios_system** -> Click **OK.**
   Target HDL: **Verilog**
   Sau đó chọn : **OK**

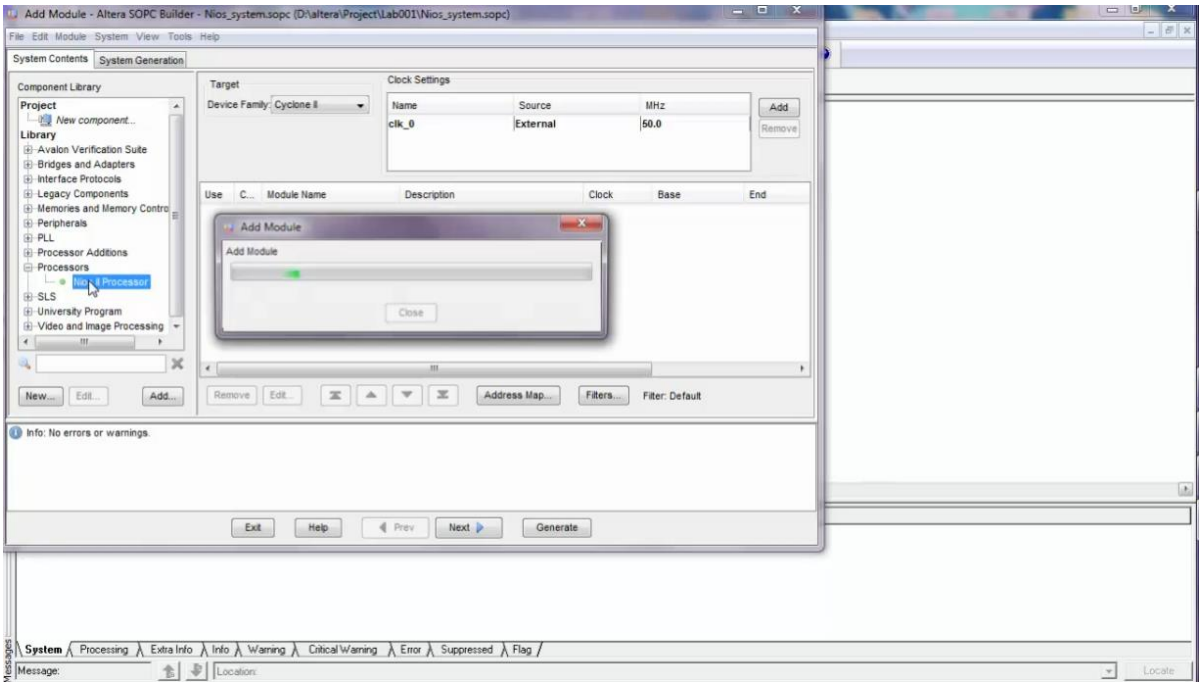


3. Trong Library: Click **Memmories and Memory Controlers -> On-Chip -> On-chip Memory(RAM or ROM)**

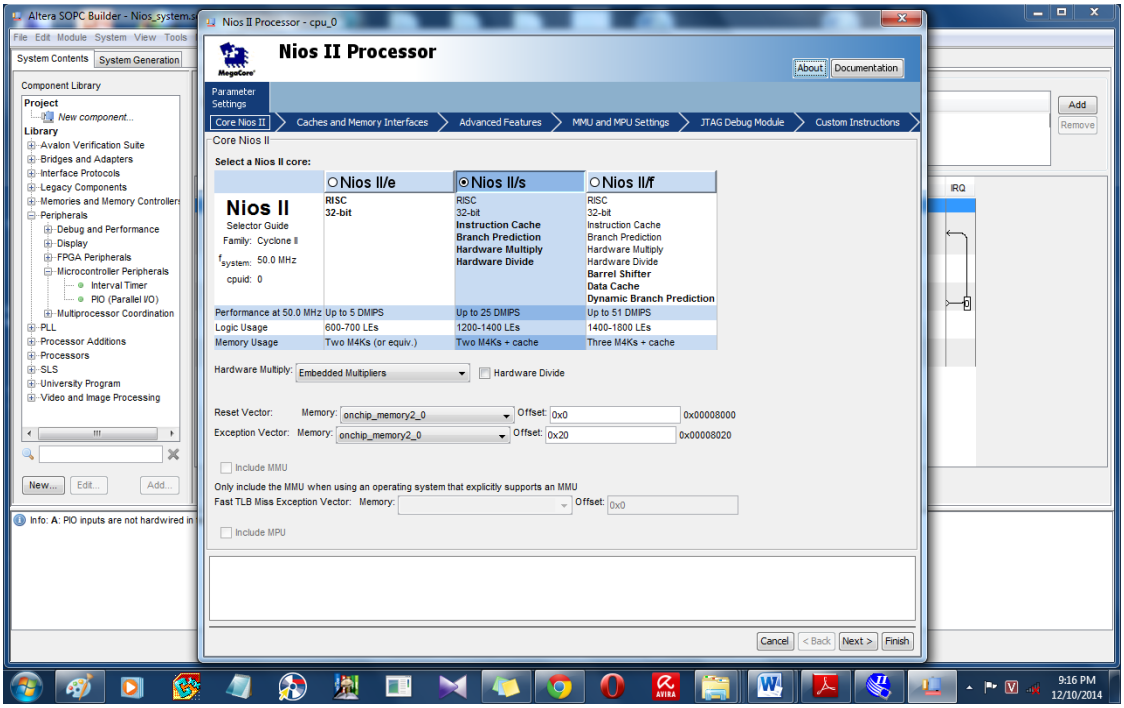


4. **Click Next -> Finish.**

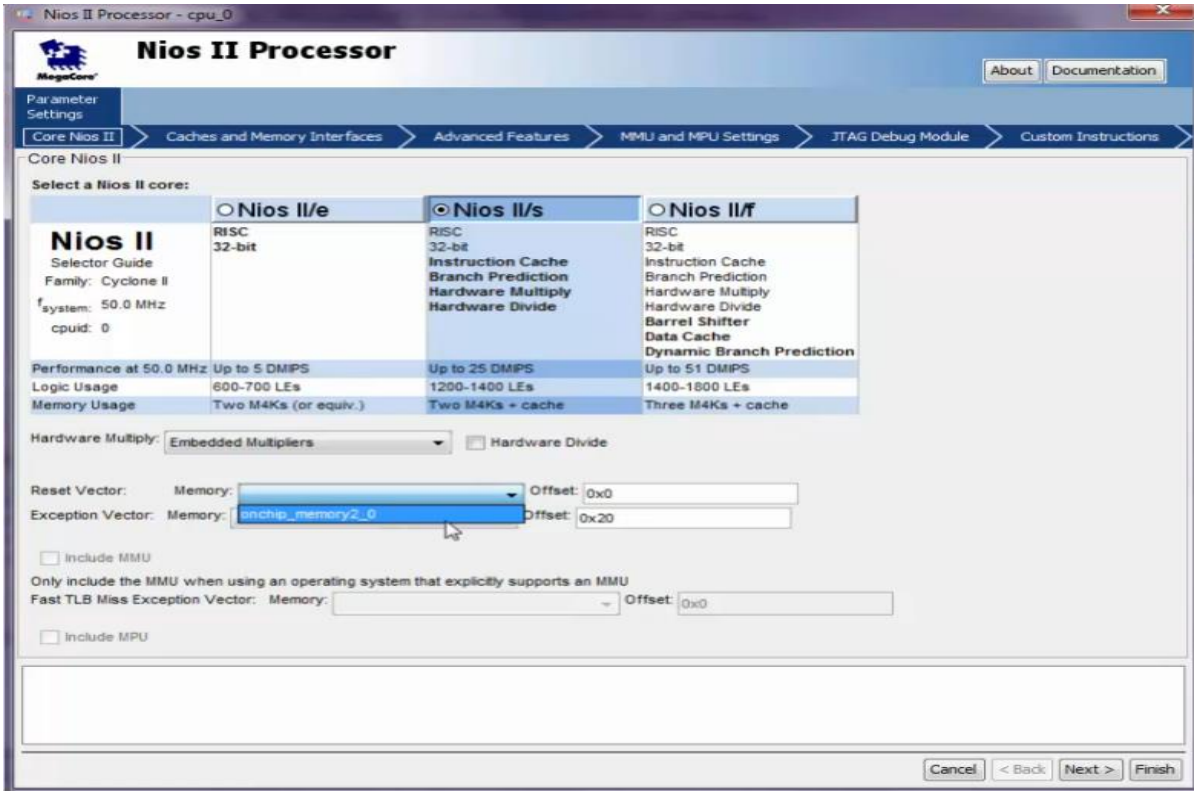
5. **Trong Library: Click Processors ->** chọn **Nios II Processor** để tạo CPU



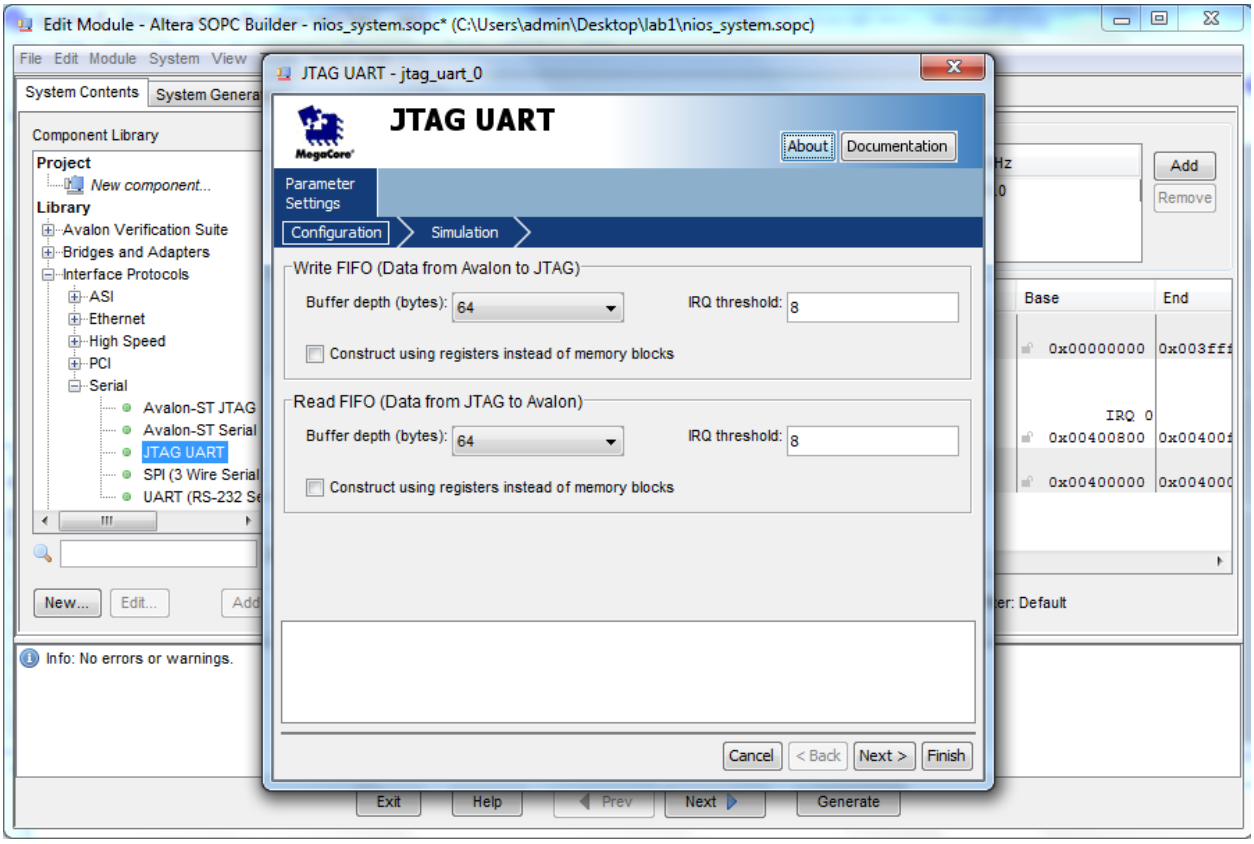
6. Chọn Nios II/s



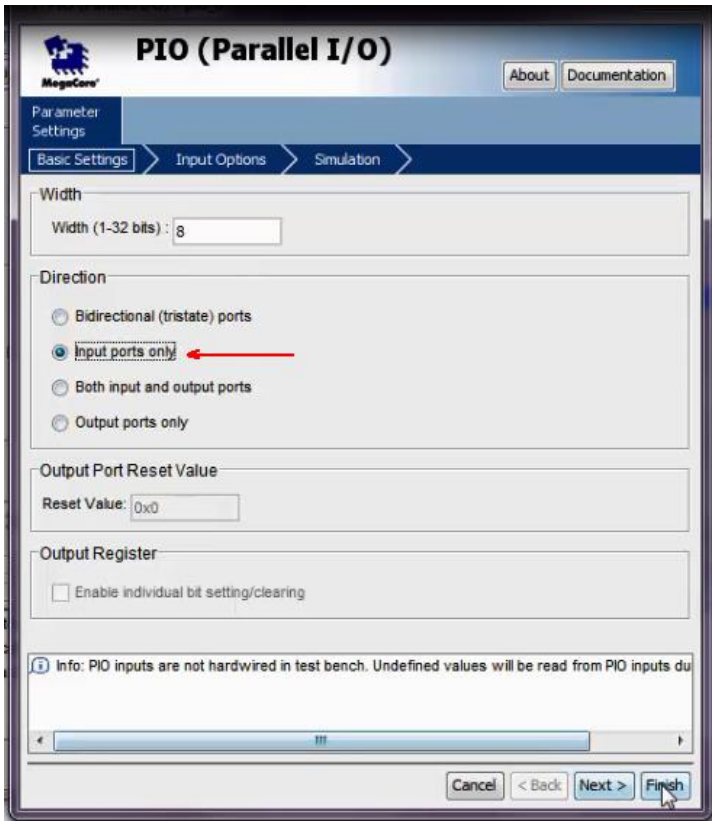
7. Trong **Reser Vector** và **Exception Vector** : chọn **Onchip_memory2_0 -> click Finish**



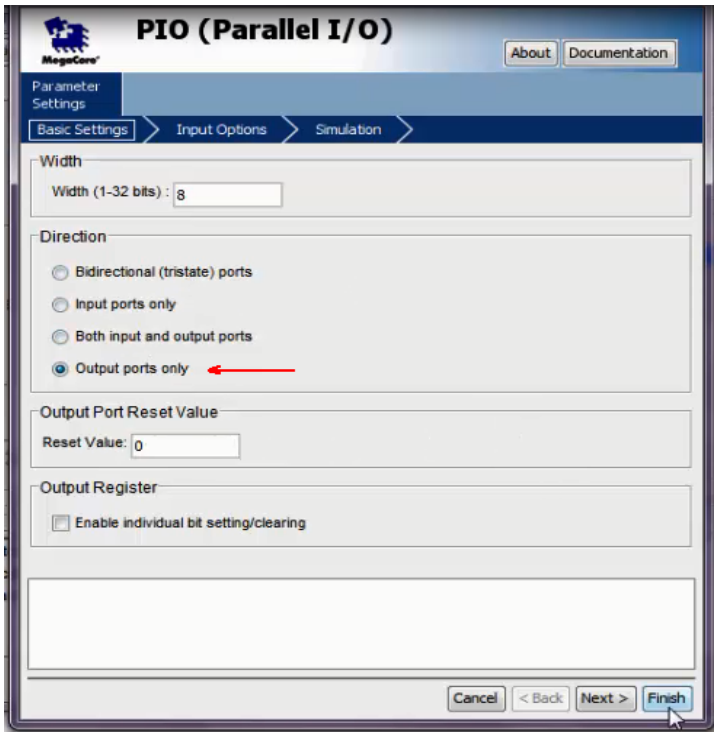
8. Trong Library: click **Interface Protocols -> Serial ->** chọn **JTAG UART,** sau đó chọn **Finish.**



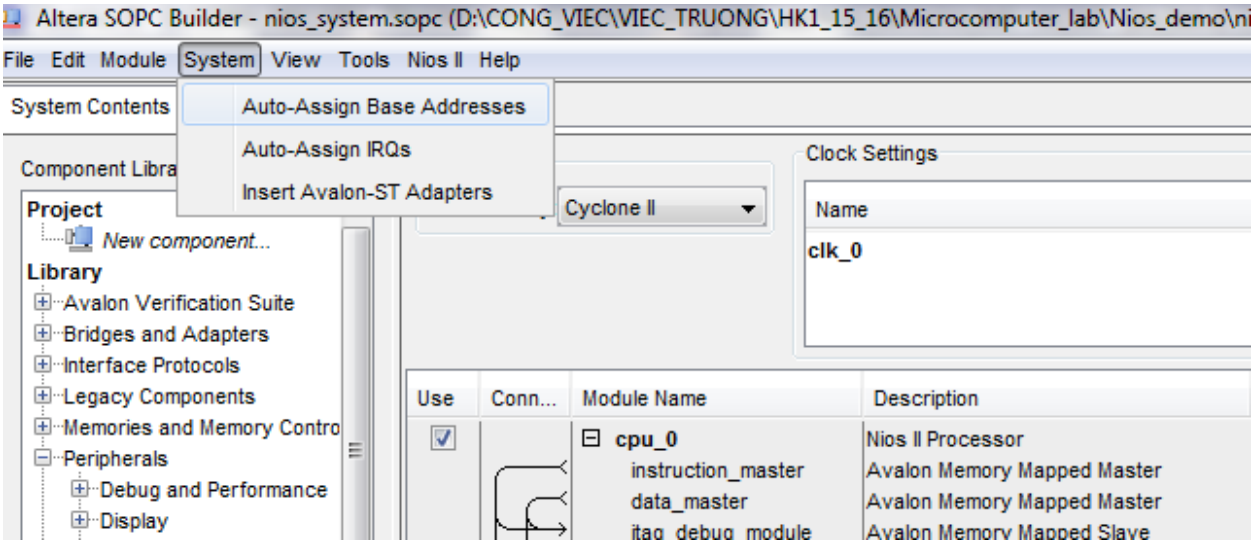
9. Tiếp theo, trong mục **Peripherals,** chọn **Microcontroller Peripherals,** chọn **PIO (Parallel I/O).** Chọn Width là **8 bit** và Direction là **Input ports only** và chọn **Finish.**



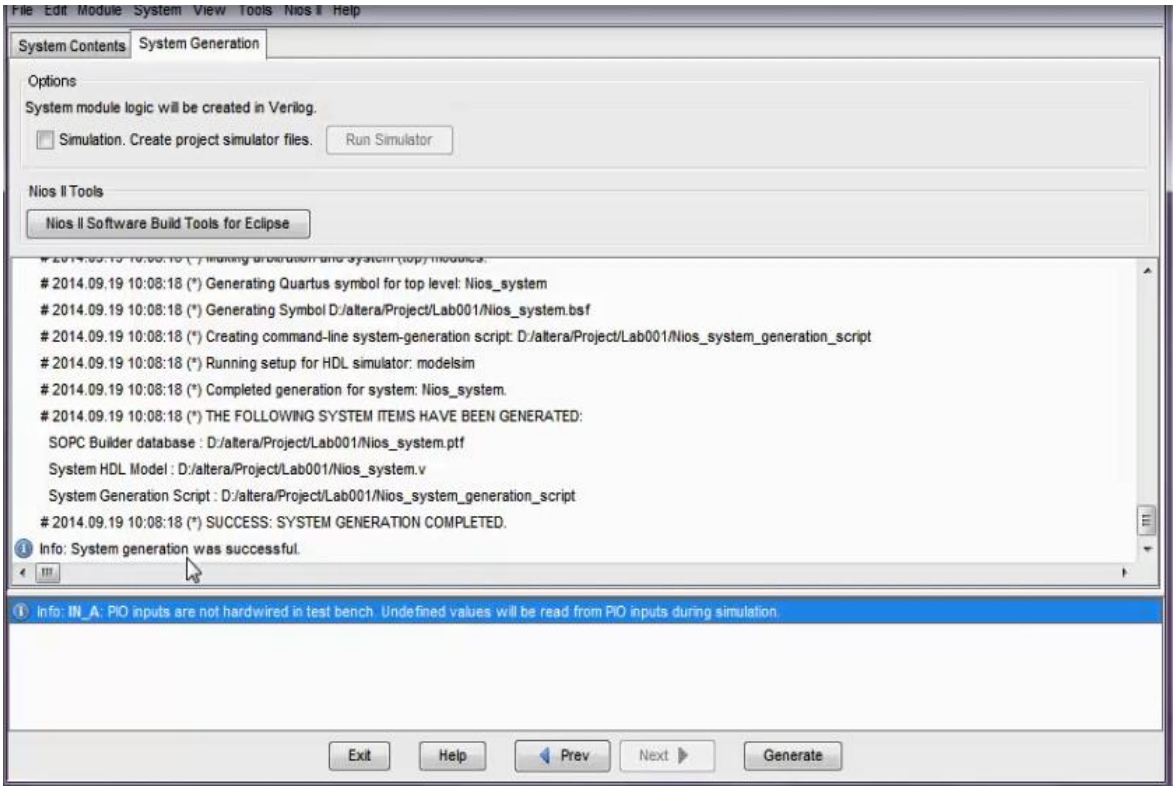
10. Chọn **Microcontroller Peripherals,** chọn **PIO (Parallel I/O).** Chọn Width là **8 bit** và Direction là **Output ports only** và chọn **Finish.**



11. Chọn **System -> Auto-Assign Base Addresses**.



12. Chọn **Generate.** Nếu **system generation was successful**, save lại và tắt SOPC builder.



IV. Verilog Code:

```
module lab1 (
	// Inputs
	CLOCK_50,
	CLOCK_27,
	EXT_CLOCK,
	KEY,
	SW,

	//  Communication
	UART_RXD,

	// Audio
	AUD_ADCDAT,

/*****************************************************************************/
	// Bidirectionals
	GPIO_0,
	GPIO_1,

	// Memory (SRAM)
	SRAM_DQ,

	// Memory (SDRAM)
	DRAM_DQ,

	// PS2 Port
	PS2_CLK,
	PS2_DAT,

	// Audio
	AUD_BCLK,
	AUD_ADCLRCK,
	AUD_DACLRCK,

	// Char LCD 16x2
	LCD_DATA,

	// AV Config
	I2C_SDAT,

/*****************************************************************************/
	// Outputs
	TD_RESET,

	//	Simple
	LEDG,
	LEDR,
```

```
HEX0,
HEX1,
HEX2,
HEX3,
HEX4,
HEX5,
HEX6,
HEX7,

//        Memory (SRAM)
SRAM_ADDR,

SRAM_CE_N,
SRAM_WE_N,
SRAM_OE_N,
SRAM_UB_N,
SRAM_LB_N,

//  Communication
UART_TXD,

// Memory (SDRAM)
DRAM_ADDR,

DRAM_BA_1,
DRAM_BA_0,
DRAM_CAS_N,
DRAM_RAS_N,
DRAM_CLK,
DRAM_CKE,
DRAM_CS_N,
DRAM_WE_N,
DRAM_UDQM,
DRAM_LDQM,

// Audio
AUD_XCK,
AUD_DACDAT,

// VGA
VGA_CLK,
VGA_HS,
VGA_VS,
VGA_BLANK,
VGA_SYNC,
VGA_R,
VGA_G,
VGA_B,
```

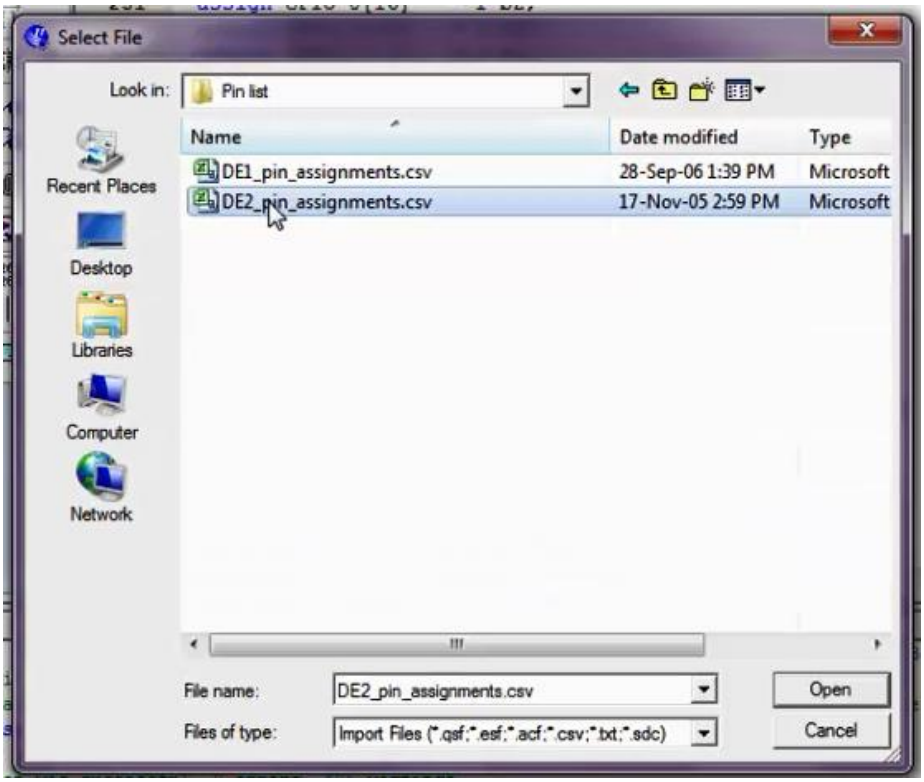
```
        // Char LCD 16x2
        LCD_ON,
        LCD_BLON,
        LCD_EN,
        LCD_RS,
        LCD_RW,

        // AV Config
        I2C_SCLK,
);

/*****************************************************************************
 *                  Parameter Declarations                   *
 *****************************************************************************/


/*****************************************************************************
 *                  Port Declarations                   *
 *****************************************************************************/
// Inputs
input                       CLOCK_50;
input                       CLOCK_27;
input                       EXT_CLOCK;
input           [3:0]   KEY;
input           [17:0]  SW;


//  Communication
input                       UART_RXD;

//  Audio
input                       AUD_ADCDAT;

// Bidirectionals
inout           [35:0]  GPIO_0;
inout           [35:0]  GPIO_1;

//        Memory (SRAM)
inout           [15:0]  SRAM_DQ;

//  Memory (SDRAM)
inout           [15:0]  DRAM_DQ;

//  PS2 Port
inout                       PS2_CLK;
inout                       PS2_DAT;
```

```
//  Audio
inout                           AUD_BCLK;
inout                           AUD_ADCLRCK;
inout                           AUD_DACLRCK;

//  AV Config
inout                           I2C_SDAT;

//  Char LCD 16x2
inout           [ 7: 0]   LCD_DATA;

// Outputs
output                          TD_RESET;

//      Simple
output          [8:0]     LEDG;
output          [17:0]    LEDR;

output          [6:0]     HEX0;
output          [6:0]     HEX1;
output          [6:0]     HEX2;
output          [6:0]     HEX3;
output          [6:0]     HEX4;
output          [6:0]     HEX5;
output          [6:0]     HEX6;
output          [6:0]     HEX7;

//      Memory (SRAM)
output          [17:0]    SRAM_ADDR;

output                          SRAM_CE_N;
output                          SRAM_WE_N;
output                          SRAM_OE_N;
output                          SRAM_UB_N;
output                          SRAM_LB_N;

//  Communication
output                          UART_TXD;

//  Memory (SDRAM)
output          [11:0]    DRAM_ADDR;

output                          DRAM_BA_1;
output                          DRAM_BA_0;
output                          DRAM_CAS_N;
output                          DRAM_RAS_N;
output                          DRAM_CLK;
output                          DRAM_CKE;
```

```
output                          DRAM_CS_N;
output                          DRAM_WE_N;
output                          DRAM_UDQM;
output                          DRAM_LDQM;

//  Audio
output                          AUD_XCK;
output                          AUD_DACDAT;

//  VGA
output                          VGA_CLK;
output                          VGA_HS;
output                          VGA_VS;
output                          VGA_BLANK;
output                          VGA_SYNC;
output          [ 9: 0]   VGA_R;
output          [ 9: 0]   VGA_G;
output          [ 9: 0]   VGA_B;

//  Char LCD 16x2
output                          LCD_ON;
output                          LCD_BLON;
output                          LCD_EN;
output                          LCD_RS;
output                          LCD_RW;

//  AV Config
output                          I2C_SCLK;

/*****************************************************************************
 *            Internal Wires and Registers Declarations              *
 *****************************************************************************/
// Internal Wires
// Used to connect the Nios 2 system clock to the non-shifted output of the PLL
wire                            system_clk;

// Internal Registers

// State Machine Registers

/*****************************************************************************
 *                 Finite State Machine(s)                   *
 *****************************************************************************/


/*****************************************************************************
 *                  Sequential Logic                 *
 *****************************************************************************/
```

```
/*****************************************************************************
 *                  Combinational Logic                *
 *****************************************************************************/

// Output Assignments
assign TD_RESET                 = 1'b1;
assign GPIO_0[ 0]         = 1'bZ;
assign GPIO_0[ 2]         = 1'bZ;
assign GPIO_0[16]         = 1'bZ;
assign GPIO_0[18]         = 1'bZ;
assign GPIO_1[ 0]         = 1'bZ;
assign GPIO_1[ 2]         = 1'bZ;
assign GPIO_1[16]         = 1'bZ;
assign GPIO_1[18]         = 1'bZ;

Nios_system NIOSII (
              // 1) global signals:
               .clk_0(CLOCK_50),
              .reset_n(KEY[0]),

              // the_A
               .in_port_to_the_A(SW[7:0]),

              // the_B
               .out_port_from_the_B(LEDR[7:0])
             )
;

endmodule
```

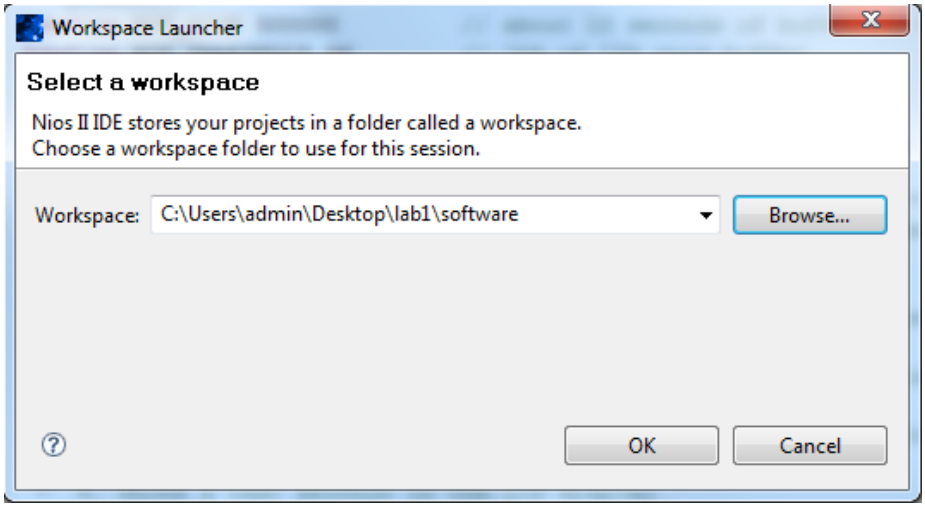
1. Save lại vào thư mục project của mình.
2. Vào **Assignments → Import Assignments →**Chọn file **DE2_pin_assignments.csv →Open**



3. Start compile.

V. C code trên NIOS II 9.1 IDE

1. Chọn **File -> chọn Switch workspace,** tạo 1 thư mục software mới trong thư mục project, sau đó tắt tab Welcome.



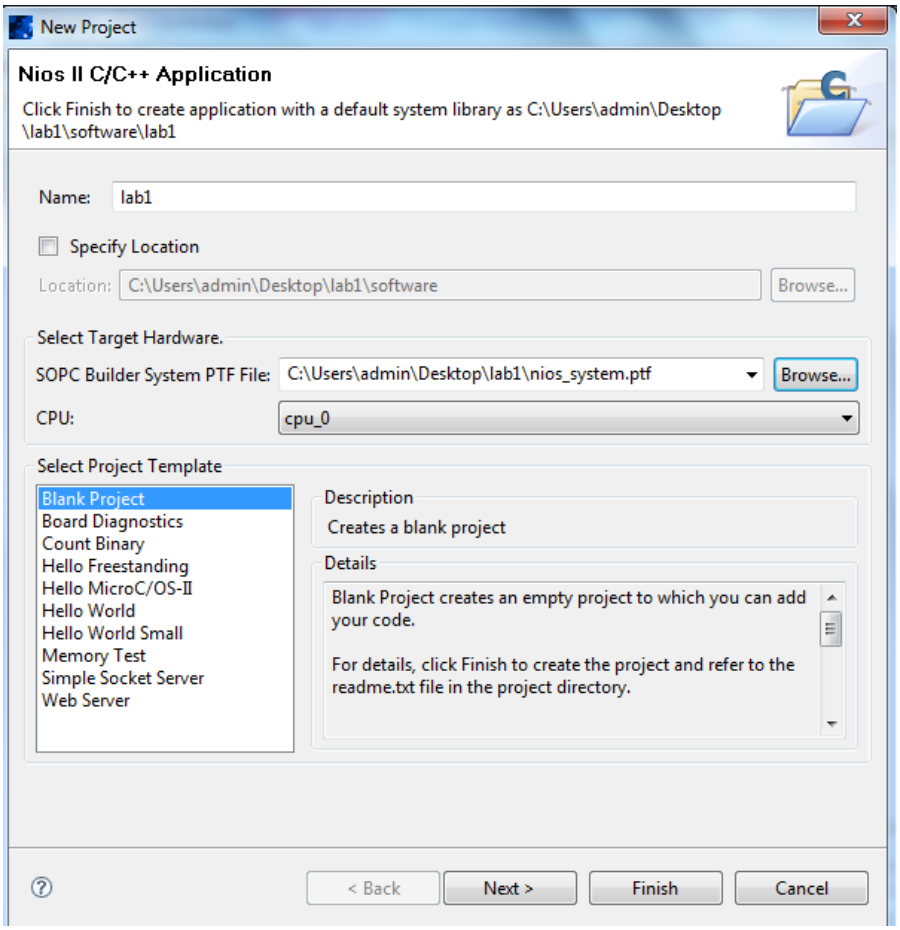
2. Chọn **File → New → Nios II C/C++ Application**

3. Đặt tên cho project.

   Chọn **Blank Project.**

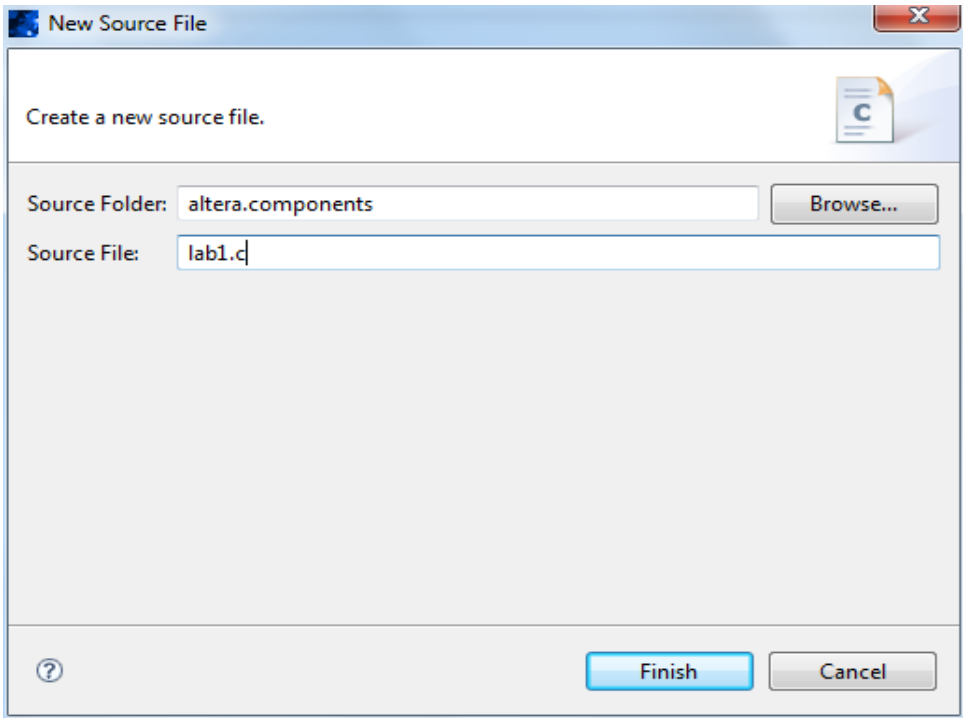
   Chọn đường dẫn để đến file **nios_system.ptf** (vừa tạo được ở các bước trên) ở mục **SOPC Builder System PTF File**

   Sau đó chọn **Finish.**



4. Click chuột phải vào **lab1_syslib[nios_system] -> Build Project**

5. Click chuột phải vào **lab1 → New → Source File.** Đặt tên source file giống với tên project mình đặt



6. Lập trình code C:

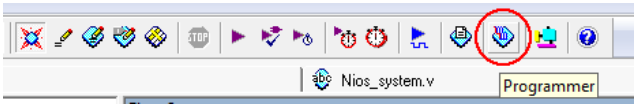
```
#include "stdio.h"

int * SW = 0x00011000;
int * LED = 0x00011010;

int main (void)
{
  while (1)
  {
    *LED = * SW;
  }
  return 0;
}
```

7. Save lại và Click chuột phải vào **lab1 -> Build Project**

VI. Run Hardware on DE2 board:

1. USB Blaster:

- In window Quartus II, click **Programmer** in taskbar



2. Run: