Lab 7: AUDIO CODEC

I. Mục tiêu:

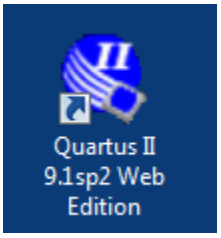
+Thiết kế hệ thống với Nios II Processor thực hiện công việc sau:

+ Thu âm khoảng 5 giây khi nhấn KEY[1], và phát lại đoạn thu âm đó khi nhấn KEY[2].

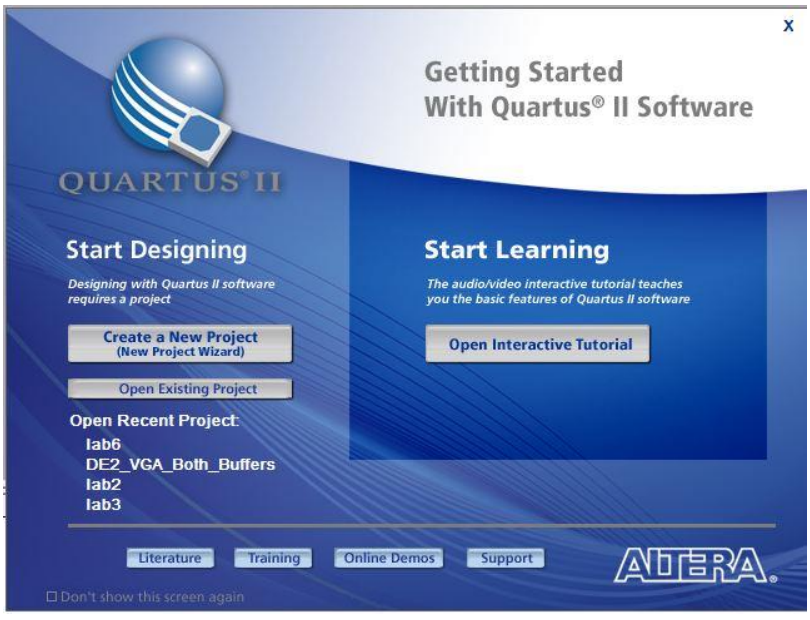
II. Tạo New Project Quartus II:

Thực hiện theo thứ tự các bước sau:

1. Tạo 1 file mới New folder với tên **lab7**
2. Mở Quartus II.



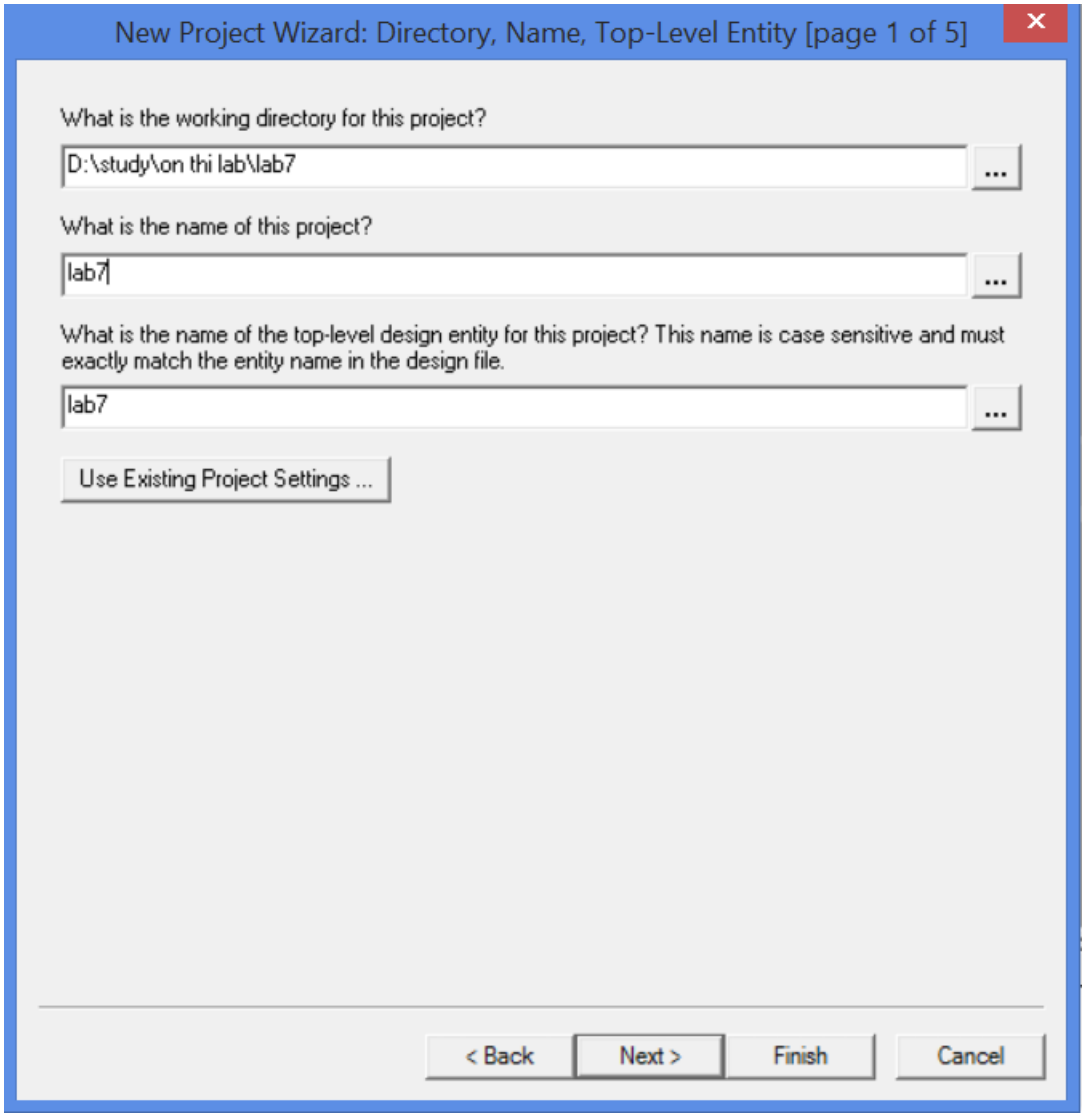
3. Trên Quartus II menu bar chọn File -> New Project Wizard.



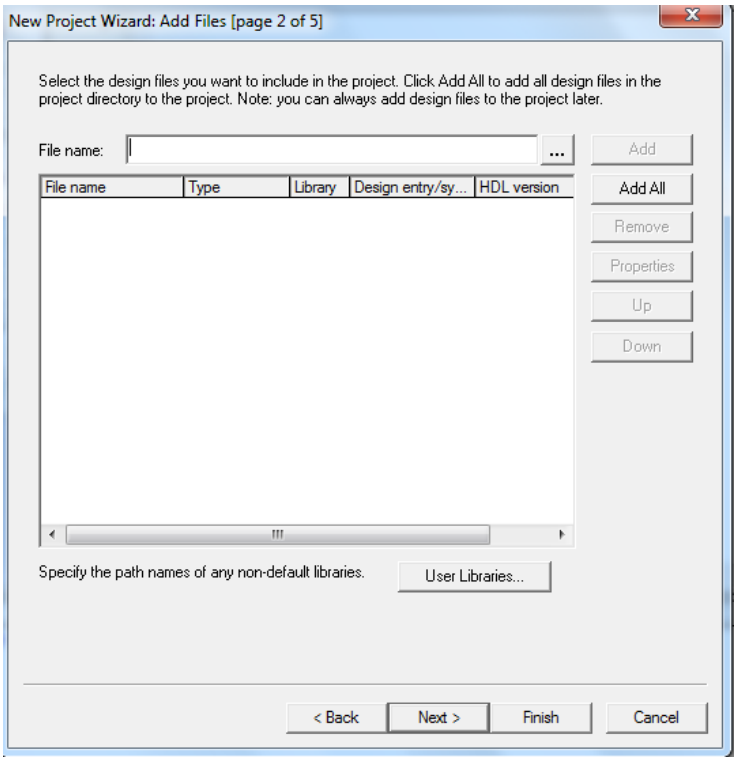
4. Trong khung thứ nhất chọn đường dẫn vào thư mục vừa tạo mang tên **lab7.**

Tên project phải trùng với tên thư mục là **lab7.**

Click Next



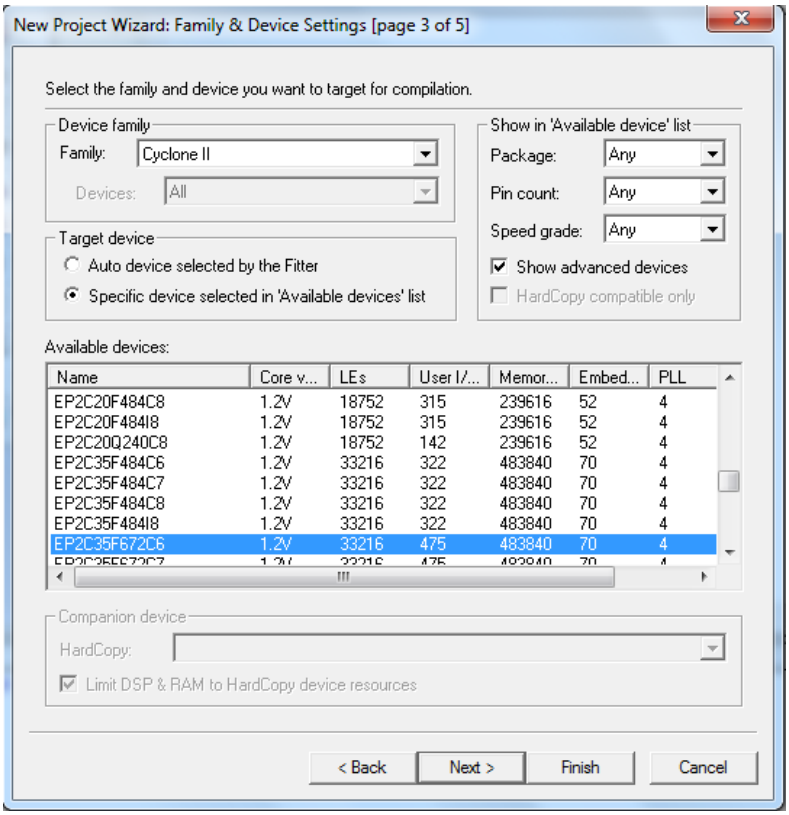
5. Click Next



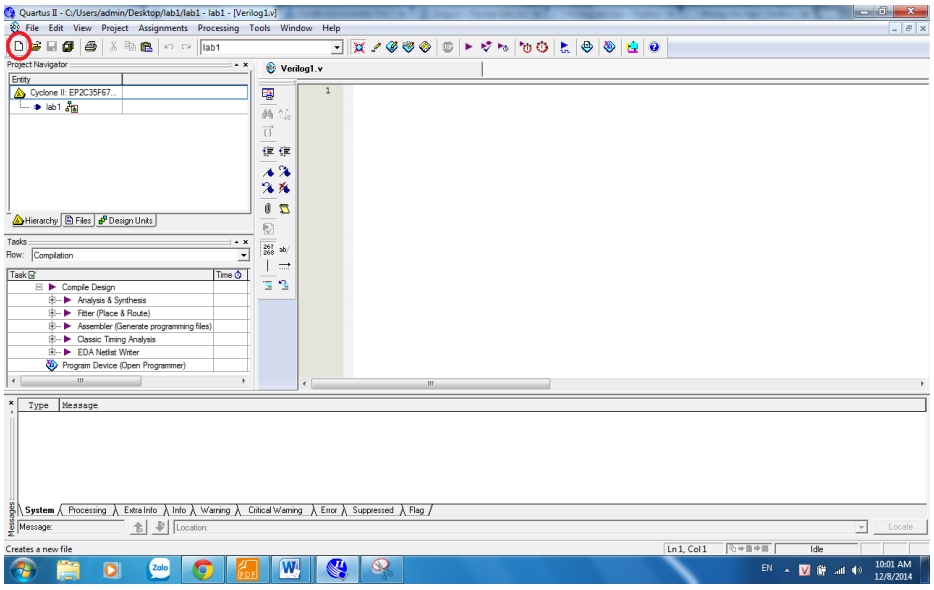
6. Chọn Cyclone II.

Available devices: **Chọn EP2C35F672C6.**

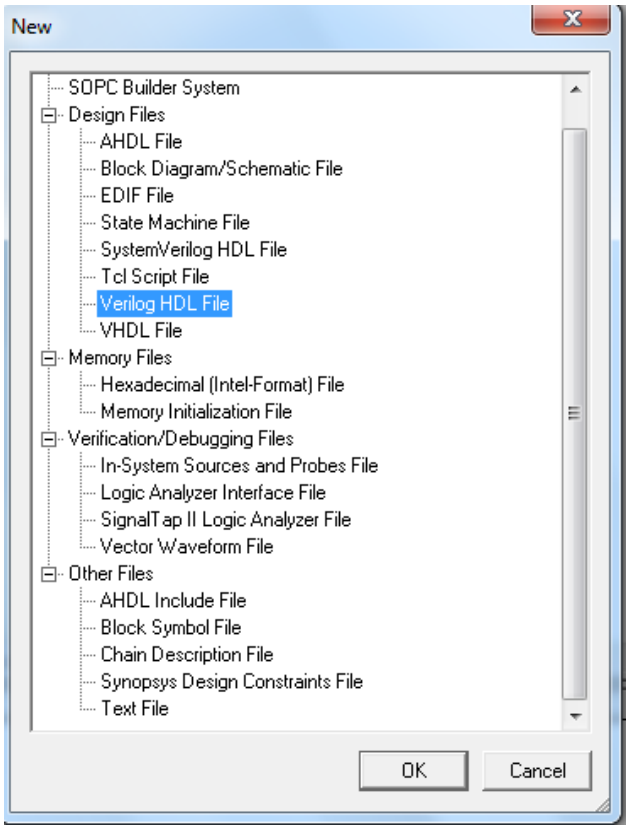
Click Next



7. Click Finish.
8. Click New

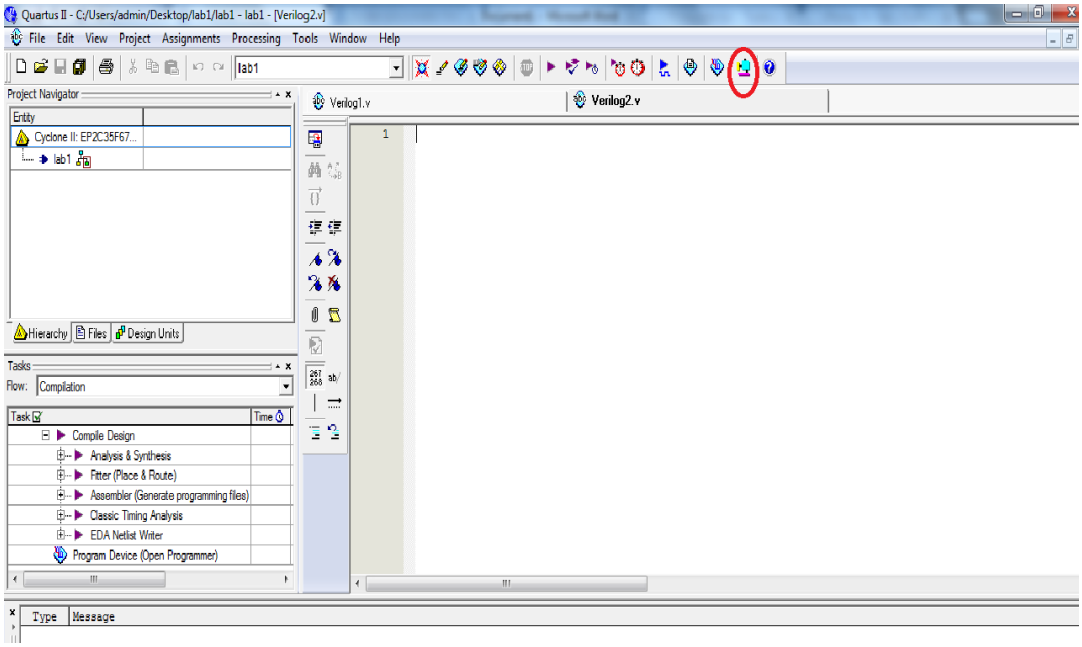


9. Chọn **Verilog HDL File -> click OK**

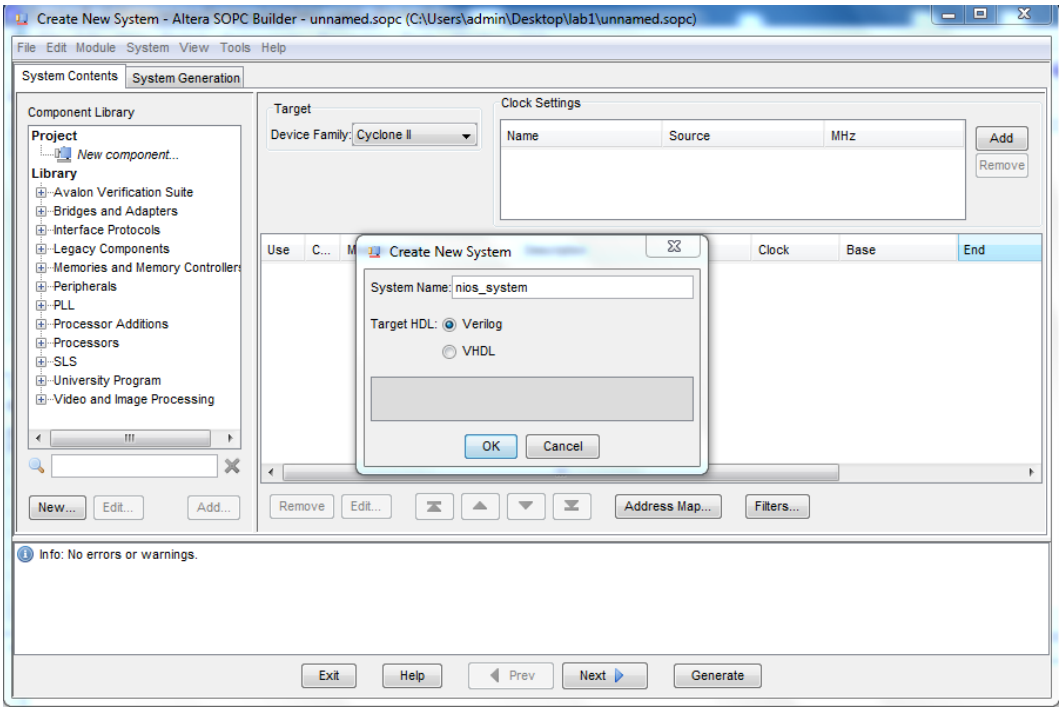


III. TẠO SOPC:

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



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

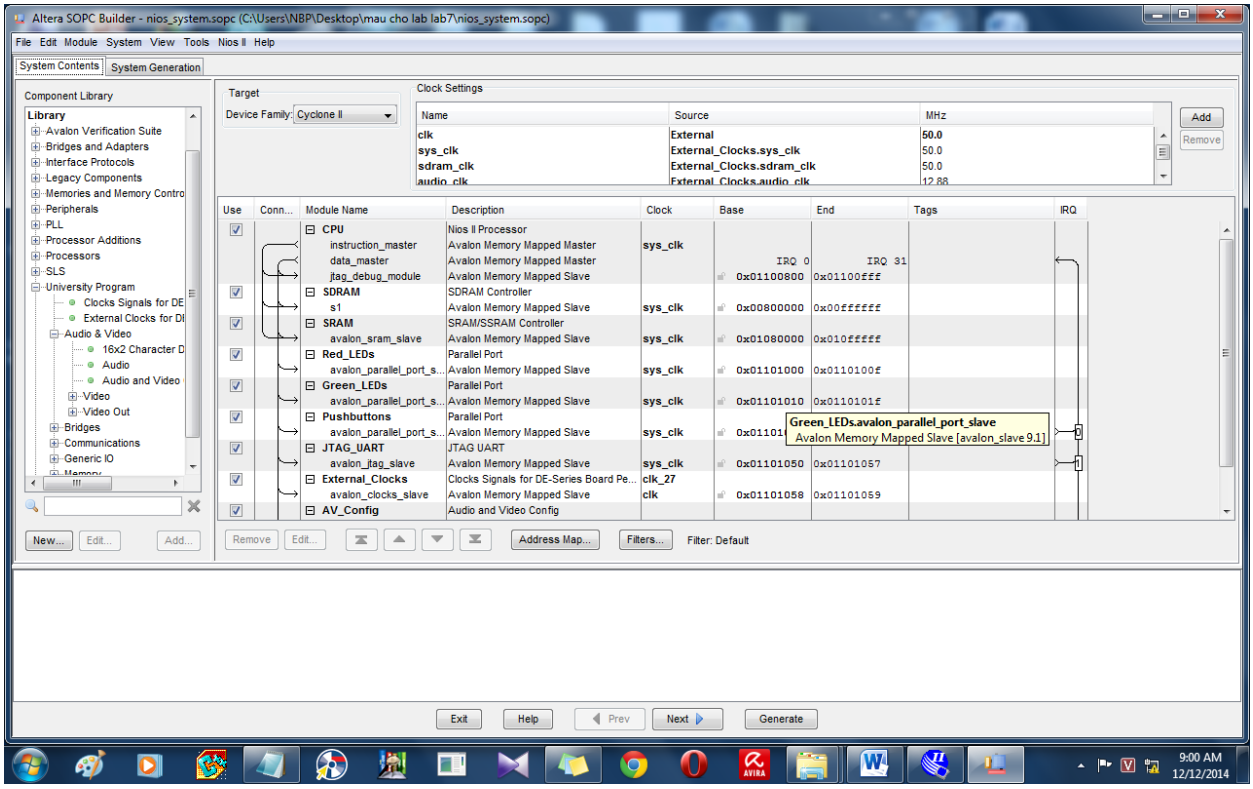


3. Building **SoPC:**
   - **Processors -> Nios II Processor.**
   - **Memories and Memory Control -> SDRAM-> SDRAM Controller:** Chọn Custom, memory size 8 Mbytes.
   - **University Program -> Memory -> SRAM/SSRAM Controller:** Chọn DE2, Use as a pixel buffer for video out.
   - **University Program -> Generic IO -> Parallel Port:** Chọn DE2 và LEDs.
   - **University Program -> Generic IO -> Parallel Port:** Chọn DE2 và LEDs.
   - **University Program -> Generic IO -> Parallel Port:** Chọn DE2 và Pushbuttons.
   - **Interface Protocols -> Serial -> JTAG UART.**
   - **University Program -> Audio & Video -> Audio and Video Config:**
     - **Audio/Video Device:** On-Board Peripherals.
     - **Auto Initialize Device(s):** Select.
     - **Audio in Path:** Microphone to ADC.
     - **Audio Out – Enable DAC Output:** Select.
     - **Audio – Line in Bypass:** Select.
     - **Data Format:** Left Justified.
     - **Bit Length:** 32.

- **Base Over-Sampling Rate:** 250fs/256fs.
- **Sampling Rate:** 0

- **University Program -> Audio & Video -> Audio:** Audio In Out: Select and Data width 32.

Và kết quả của Build SOPC của “ **lab7**” là:



IV. Verilog code:

```
module lab7 (
	// 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

// Internal Registers
```

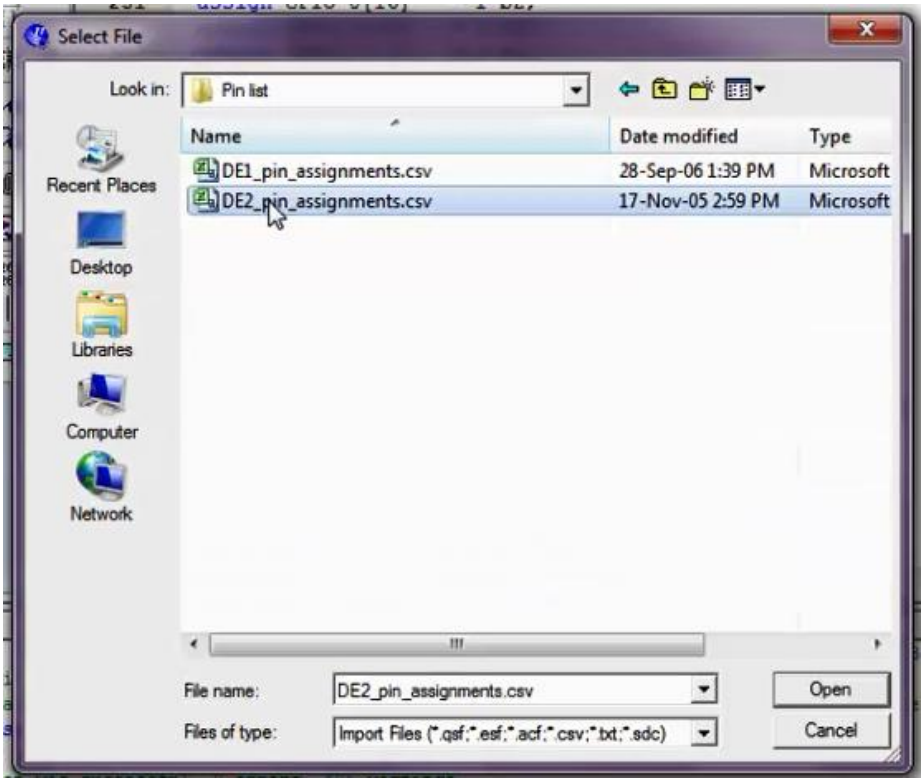
```
// State Machine Registers

/*****************************************************************************
 *                   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 Nios_II (
                // 1) global signals:
```

```
        .audio_clk
(AUD_XCK),
        .clk
(CLOCK_50),
        .clk_27
(CLOCK_27),
                .reset_n
(KEY[0]),
                .sys_clk
(),
                .sdram_clk
(DRAM_CLK),


        // the_AV_Config
         .I2C_SDAT_to_and_from_the_AV_Config    (I2C_SDAT),
                .I2C_SCLK_from_the_AV_Config
(I2C_SCLK),

        // the_Audio
        .AUD_ADCDAT_to_the_Audio
(AUD_ADCDAT),
                .AUD_BCLK_to_and_from_the_Audio
(AUD_BCLK),
                .AUD_ADCLRCK_to_and_from_the_Audio
(AUD_ADCLRCK),
                .AUD_DACLRCK_to_and_from_the_Audio
(AUD_DACLRCK),
                .AUD_DACDAT_from_the_Audio
(AUD_DACDAT),

        // the_Green_LEDs
         .LEDG_from_the_Green_LEDs
(LEDG),

        // the_Pushbuttons
       .KEY_to_the_Pushbuttons
({KEY[3:1], 1'b1}),
        // the_Red_LEDs
```

```
        .LEDR_from_the_Red_LEDs
    (LEDR),
        // the_SDRAM
         .zs_addr_from_the_SDRAM
    (DRAM_ADDR),
            .zs_ba_from_the_SDRAM
    ({DRAM_BA_1, DRAM_BA_0}),
            .zs_cas_n_from_the_SDRAM
    (DRAM_CAS_N),
            .zs_cke_from_the_SDRAM
    (DRAM_CKE),
            .zs_cs_n_from_the_SDRAM
    (DRAM_CS_N),
            .zs_dq_to_and_from_the_SDRAM
    (DRAM_DQ),
            .zs_dqm_from_the_SDRAM
    ({DRAM_UDQM, DRAM_LDQM}),
            .zs_ras_n_from_the_SDRAM
    (DRAM_RAS_N),
            .zs_we_n_from_the_SDRAM
    (DRAM_WE_N),

        // the_SRAM
        .SRAM_DQ_to_and_from_the_SRAM
    (SRAM_DQ),
            .SRAM_ADDR_from_the_SRAM
    (SRAM_ADDR),
            .SRAM_LB_N_from_the_SRAM
    (SRAM_LB_N),
            .SRAM_UB_N_from_the_SRAM
    (SRAM_UB_N),
            .SRAM_CE_N_from_the_SRAM
    (SRAM_CE_N),
            .SRAM_OE_N_from_the_SRAM
    (SRAM_OE_N),
            .SRAM_WE_N_from_the_SRAM
    (SRAM_WE_N)
        )
;
endmodule.
```

1.Save lại vào thư mục project của mình.

2.Vào Assignments → Import Assignments →Chọn file DE2_pin_assignments.csv →Open

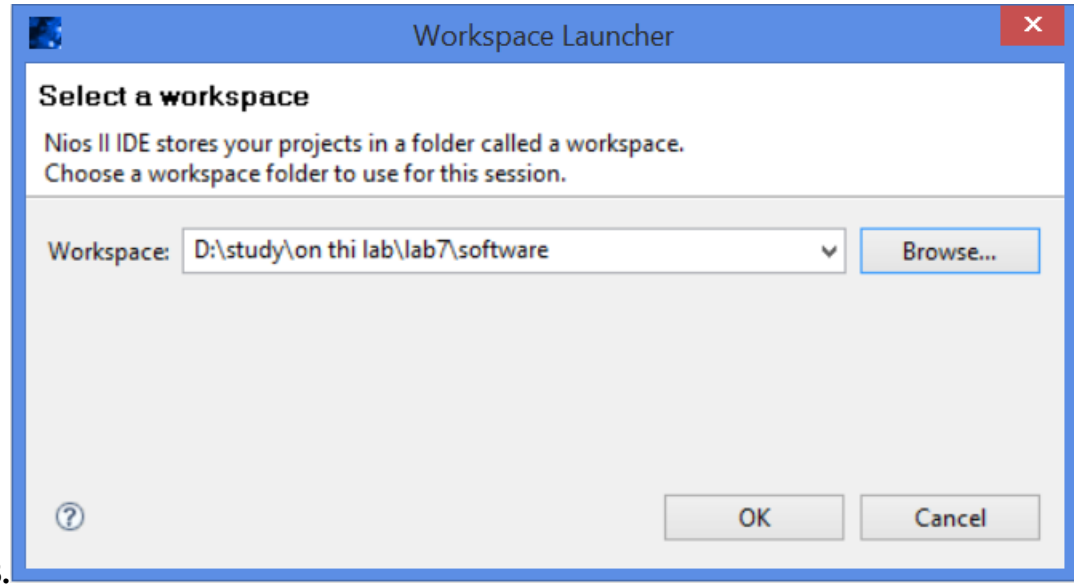


3.Start compile.

V. C code trên NIOS II 9.1 IDE

1.Mở Nios II 9.1 IDE

2.File-> Switch Workspace

3. 

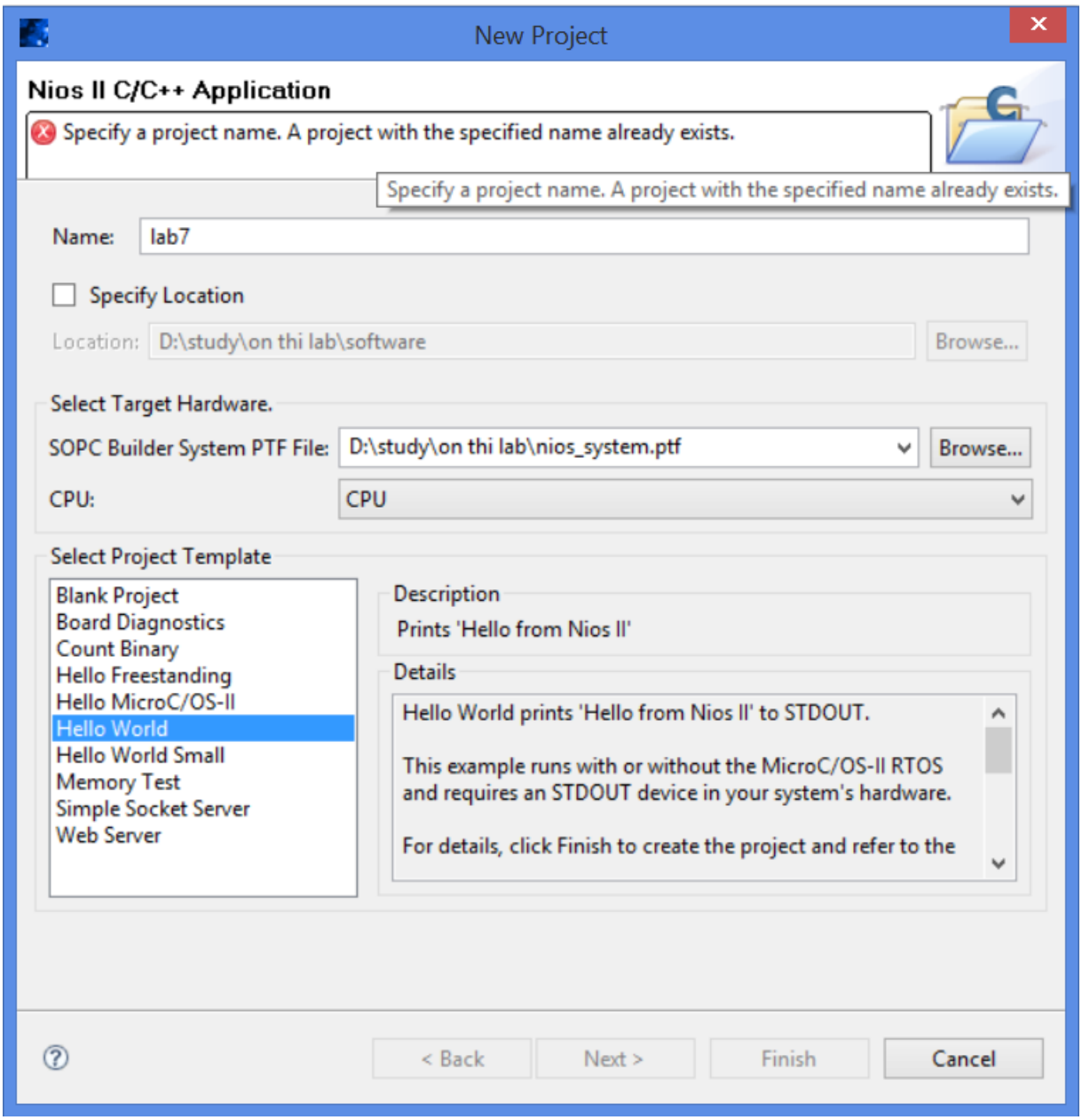
4.Chọn **File → New → Nios II C/C++ Application**

5.Đặt tên cho project.

Chọn **Blank Project.**

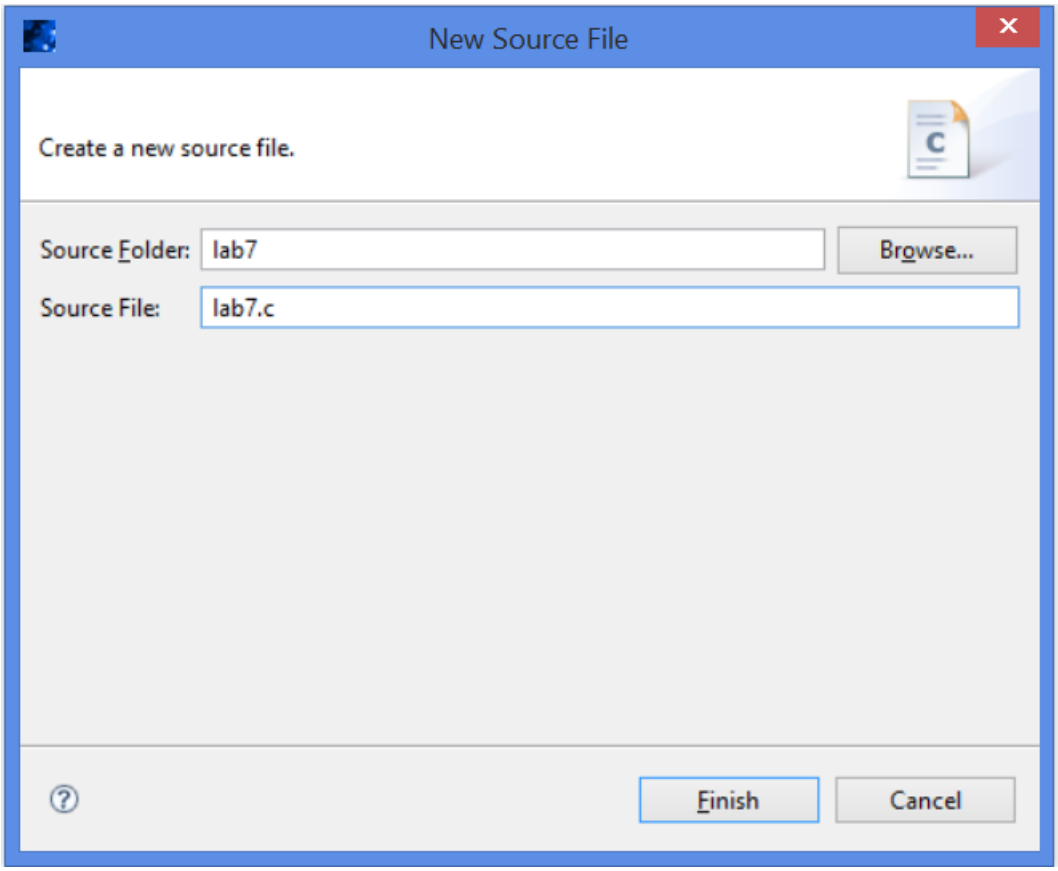
Chọn đường dẫn để đến file **nios_system.ptf** (vừa tạo được ở các bước trên) ở mục **SOPC Builder System PTF File**

Sau đó chọn **Finish.**



6.Click chuột phải vào thư mục **lab7_syslib[nios_system] -> Build Project**

7.Click chuột phải vào thư mục **lab7 → New → C Source File.** Đặt tên source file giống với tên project mình đặt



Lập trình code C:

```
#define AUDIO_BASE_ADR  0x10003040
#define GREEN_LEDS_BASE_ADR 0x10000010
#define RED_LEDS_BASE_ADR   0x10000000
#define KEYs_BASE_ADR   0x10000050

#define BUF_SIZE    250000   // about 5 seconds (@ 48K samples/sec)

int main(void)
{
/* Declare volatile pointers to I/O registers */
volatile int * AUDIO_ptr      = (int *) AUDIO_BASE_ADR;
volatile int * GREEN_LEDS_ptr = (int *) GREEN_LEDS_BASE_ADR ;
volatile int * RED_LEDS_ptr   = (int *) RED_LEDS_BASE_ADR ;
volatile int * KEYS_ptr       = (int *) KEYs_BASE_ADR;

/* used for audio record/playback */
```

```
int fifospace, leftdata, rightdata;
int buffer_index = 0;

/* reserve memory for audio buffers */
int left_buffer     [BUF_SIZE];
int right_buffer    [BUF_SIZE];
   /* record and playback audio data */
while(1)
   {
      if ( (*(KEYS_ptr + 0x3) & 0x8) > 0x0) // if KEY3 is detected
      {
*(RED_LEDS_ptr) = 0x01; // LEDR0 on
buffer_index = 0;
/* record until buffers are full */
      while (buffer_index < BUF_SIZE) // while buffer is not full
      {
/* read the audio port fifospace register */
      fifospace = *(AUDIO_ptr + 1);
      if ( fifospace & 0x000000FF )
      {
         left_buffer[buffer_index]   = *(AUDIO_ptr + 2);
         right_buffer[buffer_index]  = *(AUDIO_ptr + 3);
         ++buffer_index;
         }
}
*(KEYS_ptr+0x3) = 0x0;  // Clear the KEY3
*(RED_LEDS_ptr) = 0x00; // LEDR0 off
}
else if ( (*(KEYS_ptr + 0x3) & 0x4) > 0x0)
{
*(GREEN_LEDS_ptr) = 0x80; // LEDG7 on
buffer_index = 0;
while (buffer_index < BUF_SIZE) // while buffer is not full
      {
/* read the audio port fifospace register */
      fifospace = *(AUDIO_ptr + 1);
      if ( fifospace & 0x00FF0000 )
      {
         *(AUDIO_ptr + 2) = left_buffer[buffer_index];
         *(AUDIO_ptr + 3) = right_buffer[buffer_index];
```

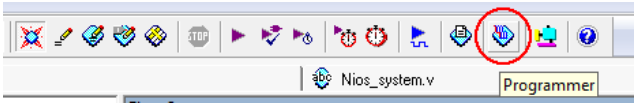
```
            ++buffer_index;

            }
    }
    *(KEYS_ptr+0x3) = 0x0;  // Clear the KEY3
    *(GREEN_LEDS_ptr) = 0x00;   // LEDG7 off
    }
    else if ( (*(KEYS_ptr + 0x3) & 0x2) > 0x0) // if KEY1 is detected
    {
    buffer_index = 0;
    /* clear samples from left_buffer and right_buffer */
    while (buffer_index < BUF_SIZE)
    {
       left_buffer[buffer_index]   = 0;
       right_buffer[buffer_index]  = 0;
       buffer_index++;
    }
    /* clear record FIFOs and playback FIFOs */
    *(AUDIO_ptr)    = 0x0C;
    *(AUDIO_ptr)    = 0x00;
    *(KEYS_ptr+0x3) = 0x0;  // Clear the KEY3
    }
    }
    }
```

8. Save lại và Click chuột phải vào **lab7 -> Build Project**

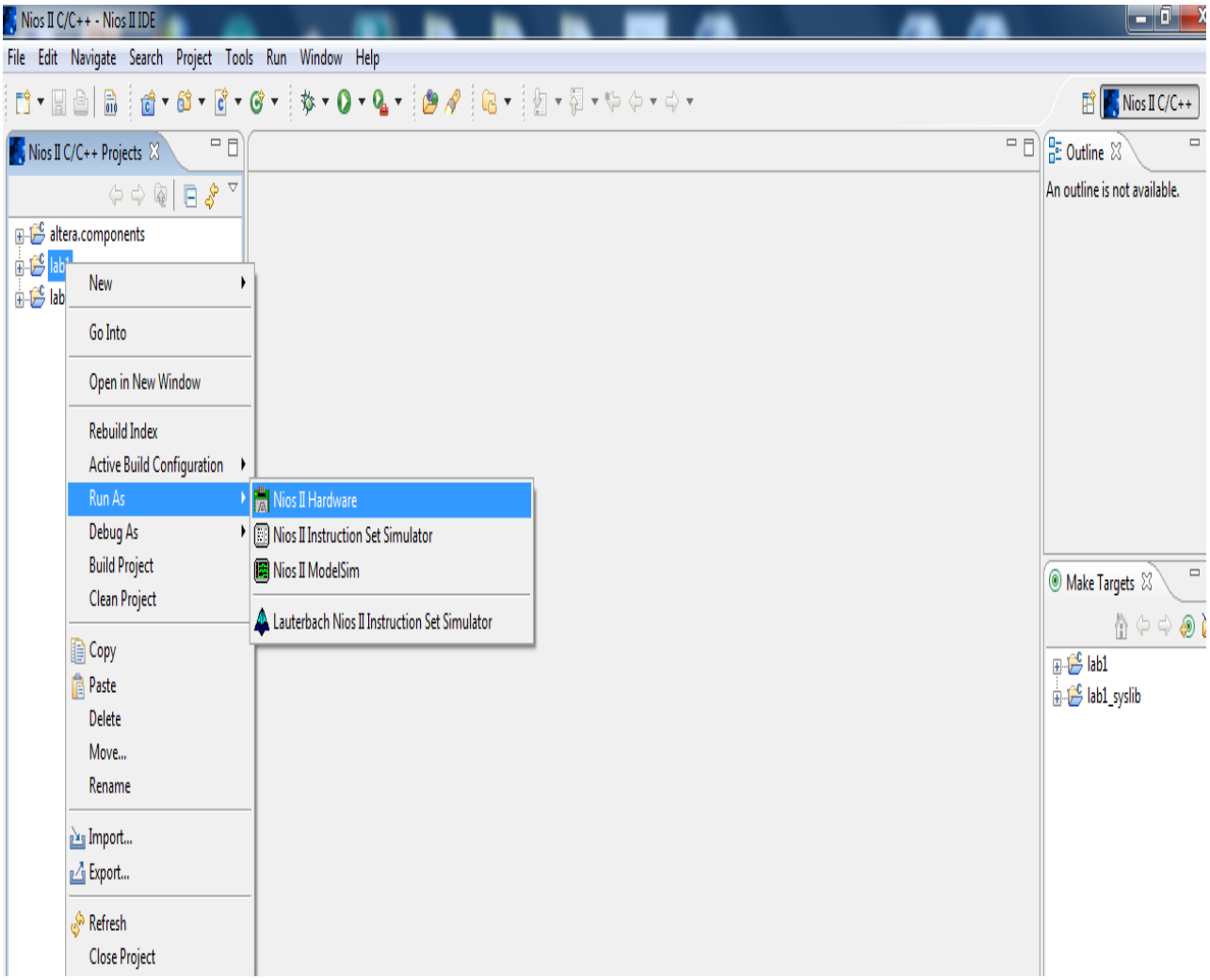
VI. Run Hardware on DE2 board:

1. USB Blaster:

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



2. Run


Nios II C/C++ - Nios II IDE
File Edit Navigate Search Project Tools Run Window Help
Nios II C/C++
Nios II C/C++ Projects
altera.components
New
Go Into
Open in New Window
Rebuild Index
Active Build Configuration
Run As
Debug As
Build Project
Clean Project
Copy
Paste
Delete
Move...
Rename
Import...
Export...
Refresh
Close Project
Nios II Hardware
Nios II Instruction Set Simulator
Nios II ModelSim
Lauterbach Nios II Instruction Set Simulator
Outline
An outline is not available.
Make Targets
lab1
lab1_syslib