Chapter 4

Introduction to PLDs and QUARTUS II

Key Terms

Programmable Logic Device (PLD) A digital integrated circuit that can be programmed by the user to implement any digital logic function.

Fixed-Function Logic A digital logic device or circuit whose function is determined when it is manufactured and, hence, cannot be changed.

Complex PLD (CPLD) A digital device consisting of several programmable sections with internal interconnections between the sections.

Schematic Entry A technique of entering CPLD design information by using a CAD (computer aided design) tool to draw a logic circuit as a schematic. The schematic can then be interpreted by design software to generate programming information for the CPLD.

Quartus II CPLD design and programming software owned by Altera Corporation.

Compile The process used by CPLD design software to interpret design information (such as a drawing or text file) and create required programming information for a CPLD.

Introduction -1

In the first three chapters of this book, we examined logic gates and Boolean algebra. These basic foundations of combinational circuitry, as well as the sequential logic circuits we will study in a later chapter, form the fundamental building blocks of many digital integrated circuits (ICs).

In the past, such digital ICs were fixed in their logic functions; it was not possible to change designs without changing the chips in a circuit. Programmable logic offers the digital circuit designer the possibility of changing design function even after it has been built. A programmable logic device (PLD) can be programmed, erased, and reprogrammed many times, allowing easier prototyping and design modification. (The industry marketing buzz often refers to "rapid prototyping" and "reduced time to market.") The number of IC packages required to implement a design with one or more PLDs is often reduced, compared to a design fabricated using standard fixed-function ICs.

Introduction -2

FPGA：元件可程式邏輯閘陣列（FPGA, Field Programmable Gate Array），是一個含有可編輯元件的邏輯電路設計平台，可供使用者程式化的邏輯閘元件。

目前以硬體描述語言（**Verilog** 或 **VHDL**）所完成的電路設計，可以經過簡單的綜合與佈局，快速的燒錄至 FPGA 上進行測試，是現代IC設計驗證的技術主流。這些可編輯元件可以被用來實現一些基本的**邏輯閘電路**（比如**AND**、**OR**、**XOR**、**NOT**）或者更複雜一些的組合功能比如解碼器或數學方程式。在大多數的FPGA裡面，這些可編輯的元件裏也包含記憶元件例如**正反器**（Flip－flop）或者其它更加完整的記憶塊。系統設計師可以根據需要通過可編輯的連接把FPGA內部的邏輯塊連接起來，就好像一個電路試驗板被放在了一個晶片裏。一個出廠後的成品FPGA的邏輯塊和連接可以按照設計者而改變，所以FPGA可以完成所需要的邏輯功能。FPGA一般來說比**ASIC**（專用集成晶片）的速度要慢，無法完成複雜的設計，而且消耗更多的電能。但是他們也有很多的優點比如可以快速成品，可以被修改來改正程序中的錯誤和更便宜的造價。廠商也可能會提供便宜的但是編輯能力差的FPGA。因為這些晶片有比較差的可編輯能力，所以這些設計的開發是在普通的FPGA上完成的，然後將設計轉移到一個類似於ASIC的晶片上。在一些技術更新比較快的行業，FPGA幾乎是電子系統中的必要部件，因為在大批量供貨前，必須迅速搶佔市場，這時FPGA方便靈活的優勢就顯得很重要。

Introduction -3

- 與CPLD的比較：

  另外一種方法是用**CPLD**（複雜可程式邏輯器件備）。

  早在**1980年代**中期，FPGA已經在**PLD**設備中紮根。 **CPLD**和**FPGA**包括了一些相對大數量的可以編輯邏輯單元。CPLD邏輯閘的密度在幾千到幾萬個邏輯單元之間，而FPGA通常是在幾萬到幾百萬。

  CPLD和FPGA的主要區別是他們的系統結構。CPLD是一個有點限制性的結構。這個結構由一個或者多個可編輯的結果之和的邏輯組列和一些相對少量的鎖定的暫存器。這樣的結果是缺乏編輯靈活性，但是卻有可以預計的延遲時間和邏輯單元對連接單元高比率的優點。而FPGA卻是有很多的連接單元，這樣雖然讓它可以更加靈活的編輯，但是結構卻複雜的多。

  CPLD和FPGA另外一個區別是大多數的FPGA含有高層次的內置模塊（比如加法器和乘法器）和內置的存儲器。一個因此有關的重要區別是很多新的FPGA支持完全的或者部分的系統內重新配置。允許他們的設計隨著系統升級或者動態重新配置而改變。一些FPGA可以讓設備的一部分重新編輯而其他部分繼續正常運行。

- 廠商

  Xilinx 和**Altera** 是目前 FPGA 的領導廠商

4.1 Programmable Logic Device (PLD)

- Supplied with no predetermined logic function.
- Programmed by user to implement any digital logic function.
- Requires specialized computer software for designing and programming.

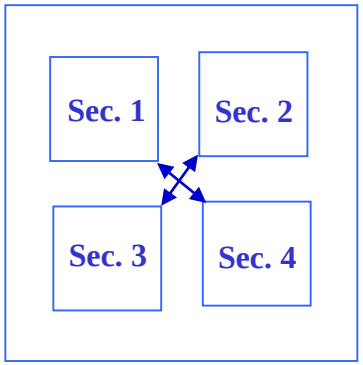
Advantages of Using PLDs

- Saves on number of chips used.
- Saves on different types of chips used.
- Shortens the design process.
- Creates design flexibility.

Complex PLD (CPLD)

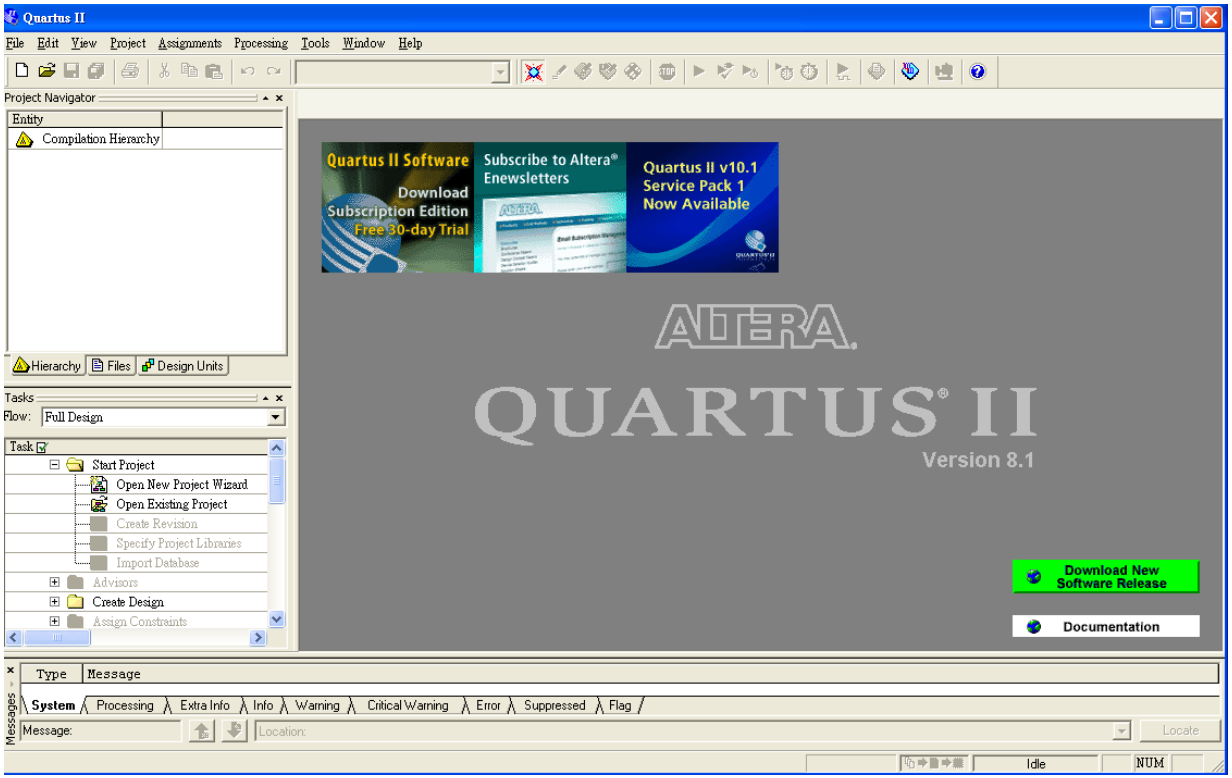
- A PLD that has several programmable sections with internal interconnections between the sections.

- In effect, several interconnected PLDs on a single chip.



PLD Programming

- Requires specialized computer software (e.g., Altera's Quartus II)
- Programmed through a series of steps known as the PLD Design Cycle.
- CPLD design and programming software owned by Altera Corporation.



Example 4.1 -1 Majority Vote Circuit

Figure 4.1 shows a majority vote circuit, as described in Problem 3.4 of Chapter 3. This circuit will produce a HIGH output when two out of three inputs are HIGH. Write the Boolean equation for the circuit and state the minimum number and type of 74HC devices required to build the circuit. How many packages would be required to build two such circuits?

Plot the majority vote circuit and write the boolean equation:

Sol:

Example 4.1 -2 Majority Vote Circuit

Figure 4.2 shows the 74HC devices required to build the majority vote circuit: one 74HC08A quad 2-input AND gate and one 74HC4075 triple 3-input OR gate. Figure 4.2 also shows connections between the devices. Note that unused gate inputs are grounded and unused outputs are left open.

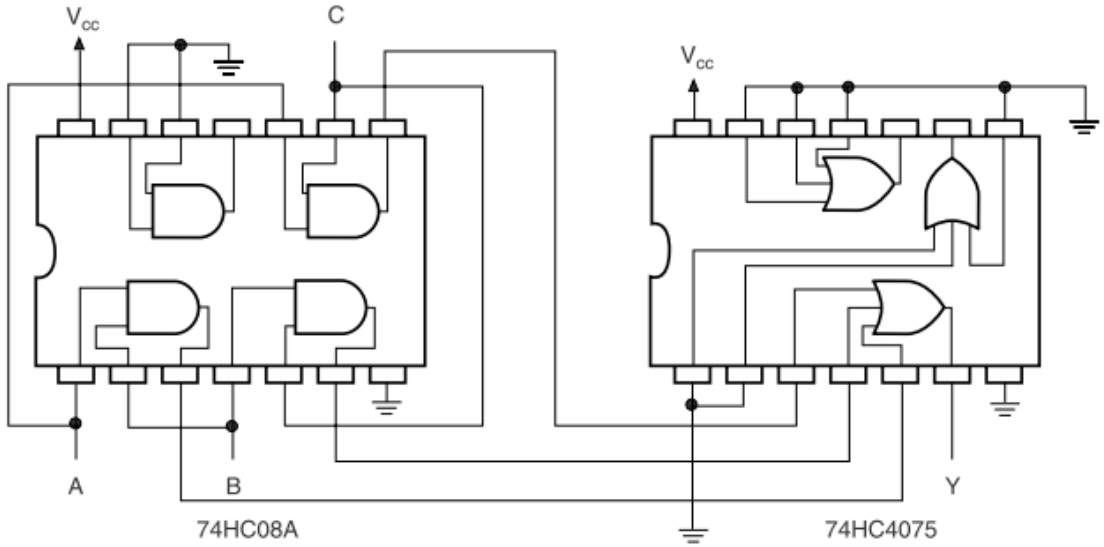


FIGURE 4.2
74HC Devices Required to Build a Majority Vote Circuit

Two majority vote circuits would require 6 ANDs and two ORs. This requires one more 74HC08A package.

Example 4.2 PLD Programming

Show how a CPLD can be programmed with a majority vote function, using a **schematic entry** tool. State how many CPLDs would be required to build two majority vote circuits.

■ Solution

A CPLD can be programmed by entering the schematic directly, using PLD programming software such as Altera Corporation's **Quartus II.** Figure 4.3 shows the circuit as entered in a Quartus II Block Diagram File.

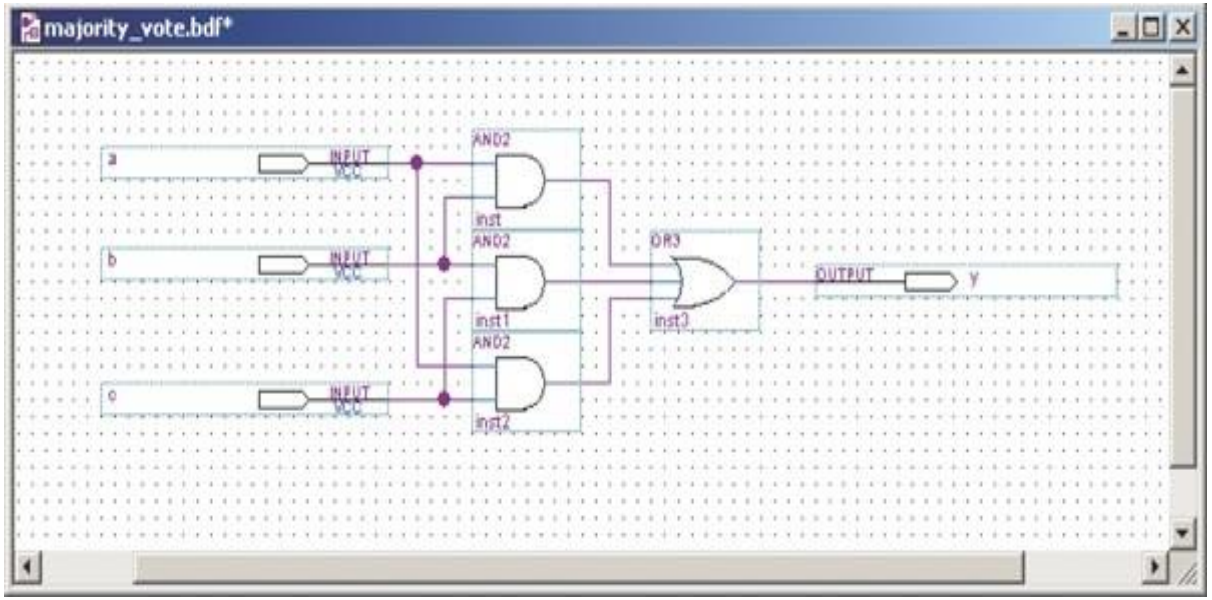


FIGURE 4.3 Example 4.2: Quartus II Block Diagram File of Majority Vote Circuit

PLD Key Terms -1

- **Design entry –** enter the circuit design.

  Design entry The process of using software tools to describe the design requirements of a PLD. Design entry can be done by entering a schematic or a text file that describes the required digital function.

- **Simulation –** verify that the circuit outputs correctly respond to the inputs.

  Simulation Verifying design function by specifying a set of inputs and observing the resultant outputs. Simulation is generally shown as a series of input and output waveforms.

- **Compilation**（編譯）**–** create the required design information for programming the CPLD.

PLD Key Terms -2

- **Fitting** – determine which portions of the CPLD to assign as circuitry for the required design.

> **Fitting** Assigning internal PLD circuitry, as well as input and output pins, for a PLD design.

- **Programming** – configures the CPLD to perform the desired logic function.

> **Programming** Transferring design information from the computer running PLD design software to the actual PLD chip.

4.2 Programmable Sum-of-Product (SOP) Array - 1

- Consists of AND gates and OR gates organized in an SOP array.
- Connections are made or broken by a matrix of fused links.
- Intact（完整無缺的）fuse connection is made.
- Intact fuse lines are indicated by ‘X’.
- Blown fuse（保險絲等燒斷）connection is open.

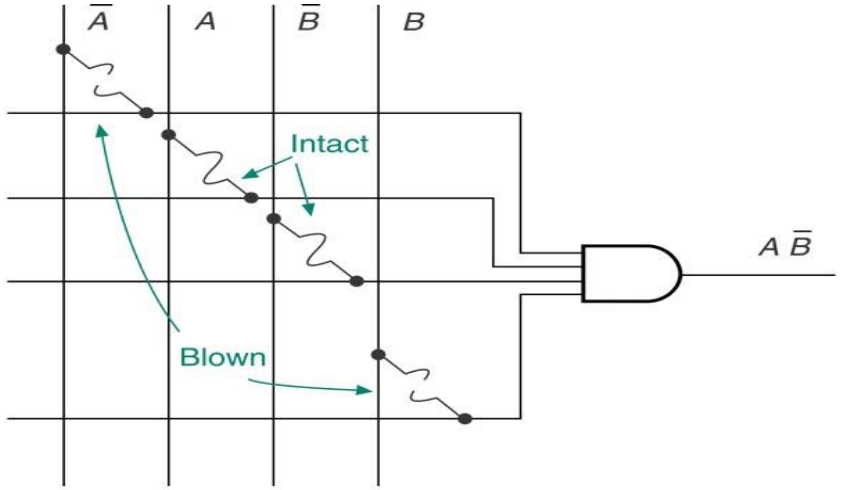
Key Terms

Product Line A single line on a logic diagram used to represent all inputs to an AND gate (i.e., one product term) in a PLD sum-of-products array.

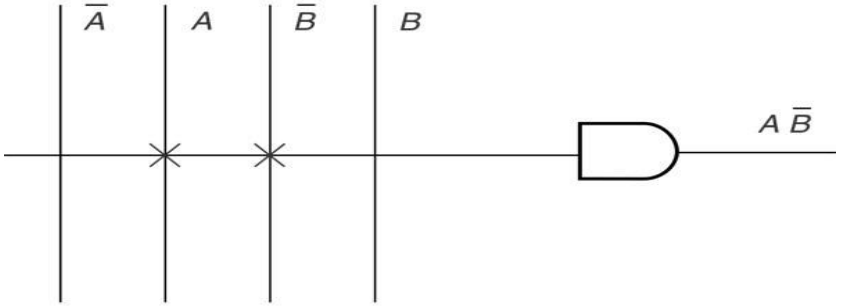
Input Line A line that applies the true or complement form of an input variable to the AND matrix of a PLD.

PAL Programmable array logic. Programmable logic with a fixed OR matrix and a programmable AND matrix.

Programmable SOP Array - 2

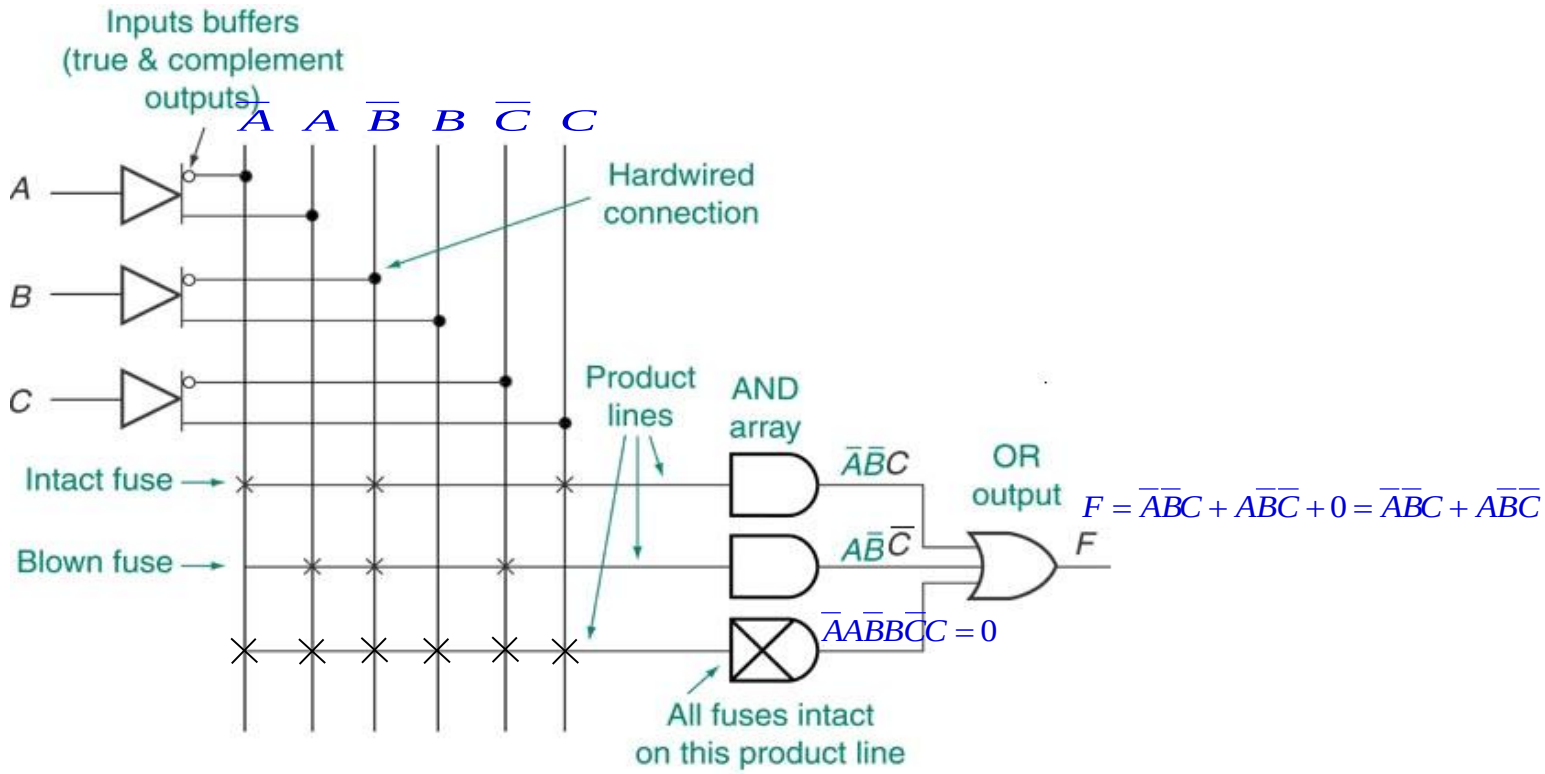


a. Crosspoint fuse matrix (*A* and $\overline{B}$ intact)



b. PLD notation for fuse matrix

PAL Fuse Matrix & Combinational Outputs



- The configuration of the above figure, with a programmable AND matrix and a hardwire OR connection, is called PAL (programmable array logic) architecture.

4.3 PAL (Programmable Array Logic) Fuse Matrix & Combinational Outputs

- Fuse assignments done with special software.
- The software takes inputs such as Boolean equations, truth tables, or other forms.
- Software produces the simplest solution to the problem.

Key Terms

JEDEC Joint Electron Device Engineering Council

JEDEC File An industry-standard form of text file indicating which fuses are blown and which are intact in a programmable logic device.

Text File An ASCII-coded document stored electronically.

Cell A programmable location in a PLD, specified by the intersection of an input line and a product line.

Product Line First Cell Number The lowest cell number on a particular product line in a PAL AND matrix where all cells are consecutively numbered.

Input Line Number A number assigned to a true or complement input line in a PAL AND matrix.

Checksum An error-checking code derived from the accumulated sum of the data being checked.

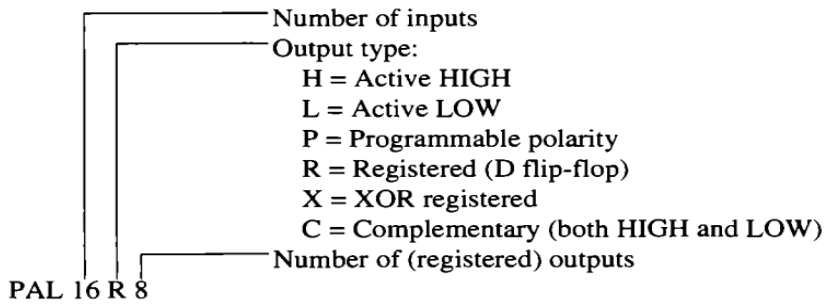
Multiplexer A circuit that selects one of several signals to be directed to a single output.

PAL Circuit

Figure 4.6 shows the logic diagram of a PAL16L8 PAL circuit. This device can produce up to eight different sum-of-products expressions, one for each group of AND and OR gates. The device has active-LOW tristate outputs, as indicated by the "L" in the part number. The output enable of each tristate buffer is controlled by a product line from the related AND matrix.

The pins that can be used only as inputs or outputs are marked "I" or "O," respectively. Six of the pins can be used as inputs or outputs and are marked "I/O." The I/O pins can also feed back a derived Boolean expression into the matrix, where it can be employed as part of another function. A detail of an I/O section is shown in Figure 4.7

The part number of a PAL device gives the designer information about the number of inputs and outputs and their configurations, as follows:



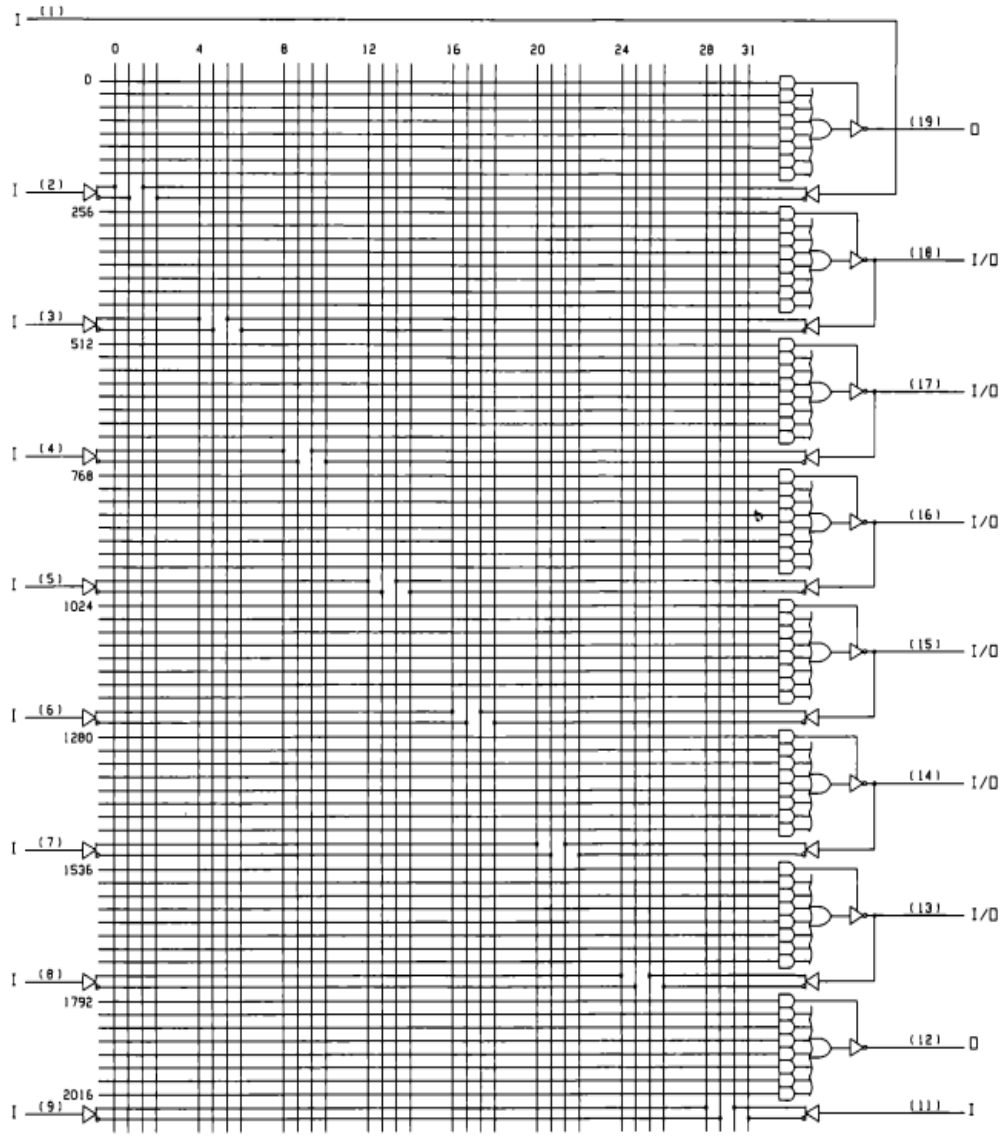
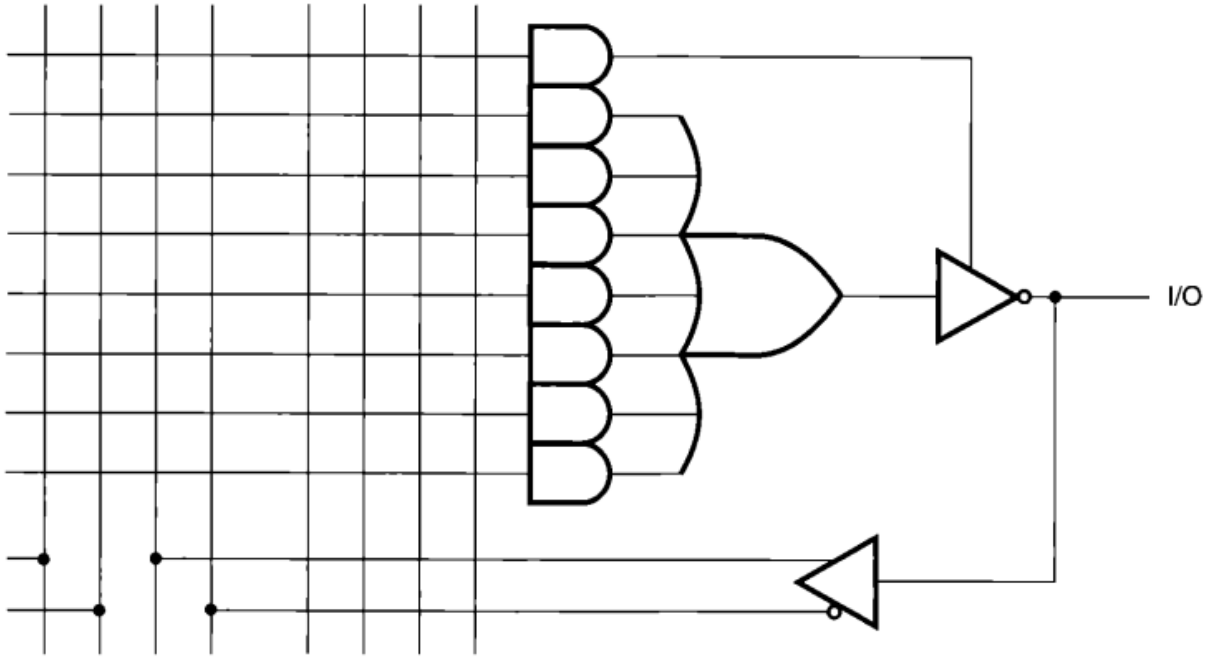


FIGURE 4.6 Unprogrammed PAL16L8

PAL16L8 I/0 Section

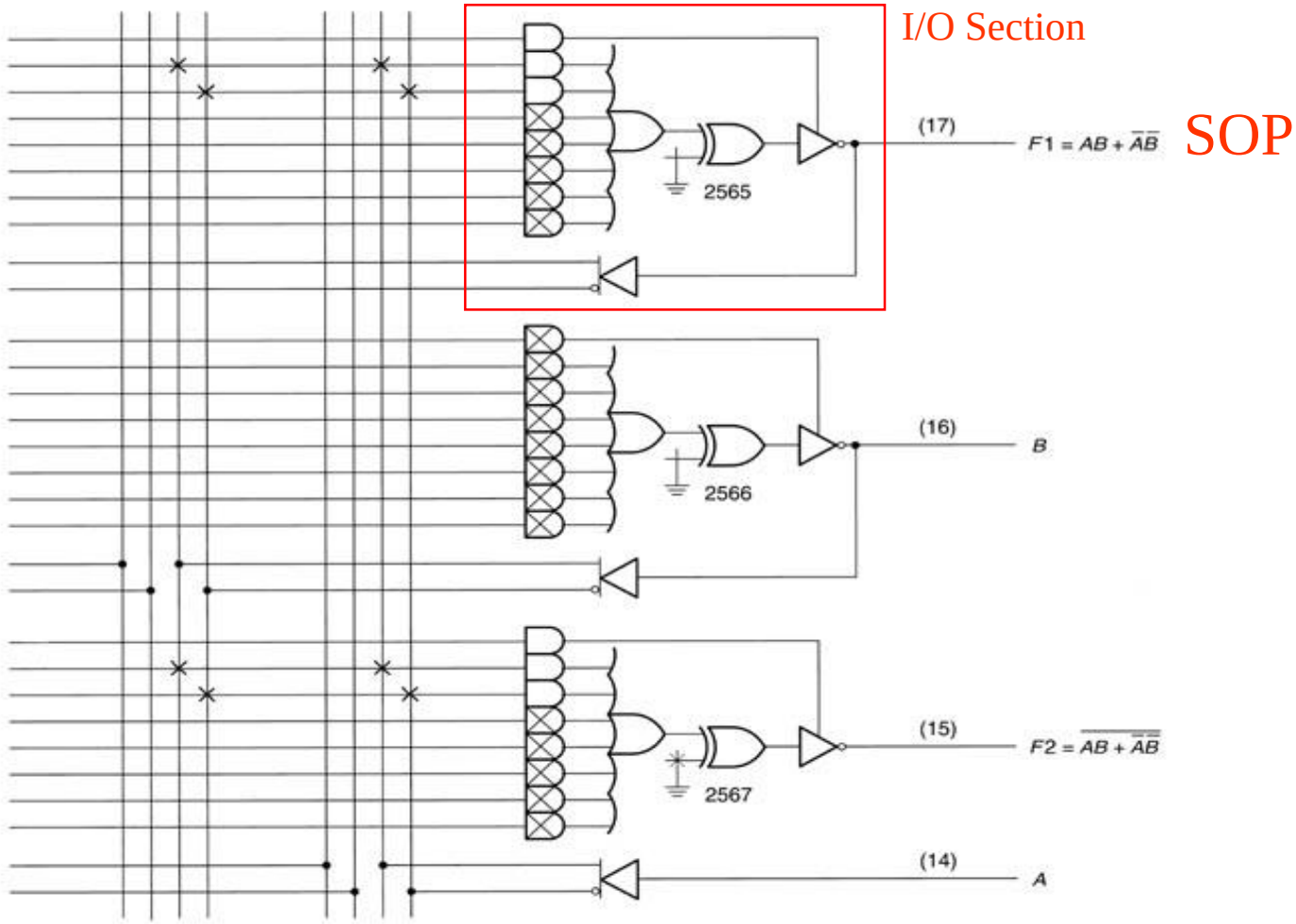


SOP Expression

Outputs with Programmable Polarities -1

- Allows for flexibility in the final design.
- Each output has a programmable XOR gate associated with the output.
- The XOR gate can be programmed to act as either an inverter or a buffer.

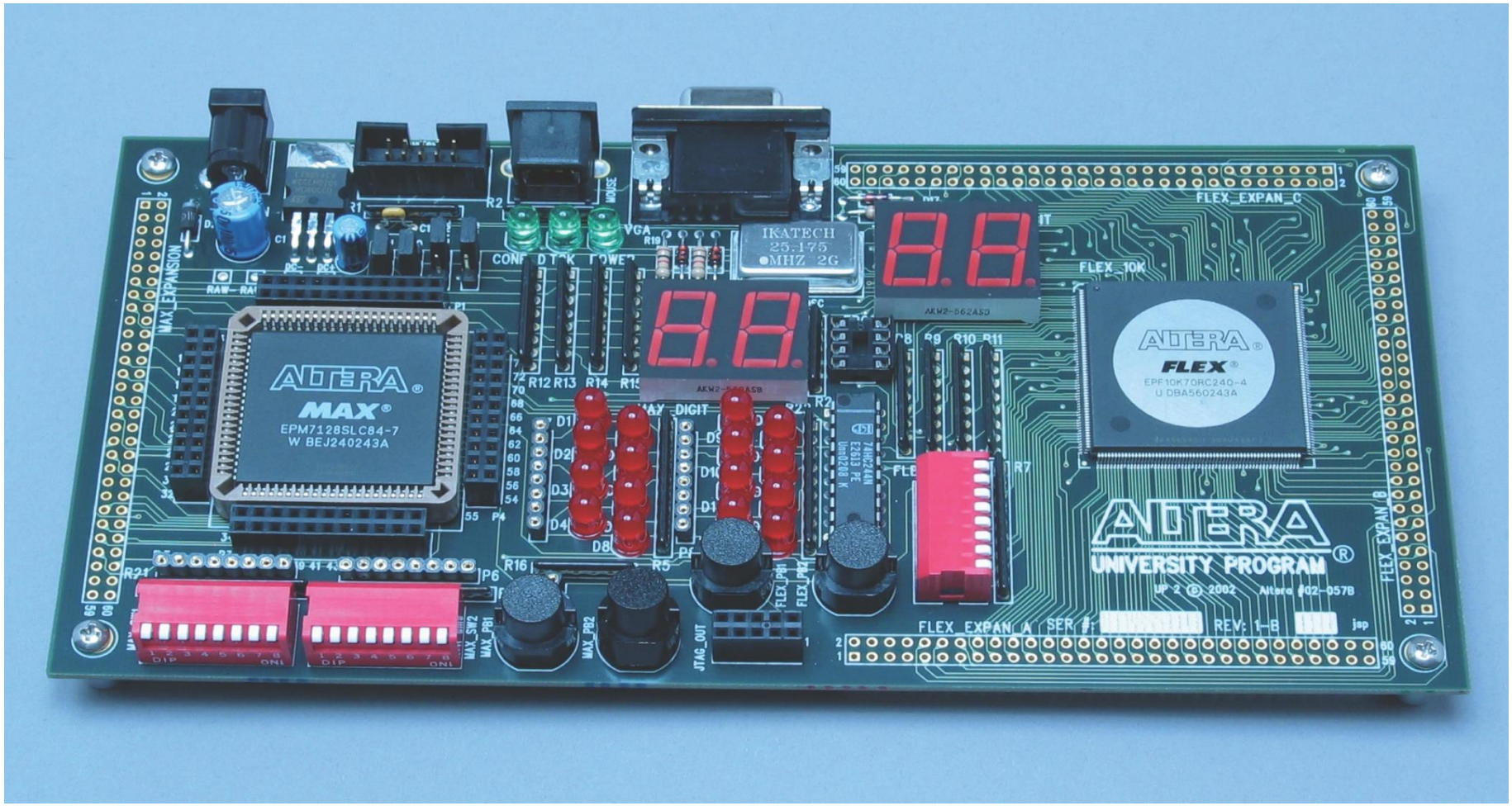
Outputs with Programmable Polarities -2



CPLD Development Boards

- Altera UP-2
- DeVry eSoc
- RSR Electronics PLDT-2
- HVW Technologies

CPLD Development Boards



Altera's UP-2 Board

- Contains two target devices:
  - MAX7000S family–EPM7128SLC84-7, a nonvolatile(非揮發性的) CPLD.
  - FLEX10K family–EPF10K70RC240-4, a volatile (揮發性的) CPLD.

Volatile vs. Nonvolatile

- Volatile–information (programming) is available only as long as power is applied to the device.

- Nonvolatile–information (programming) is available even after power is removed and then reapplied.

Quartus II Definitions

- Project–a collection of files associated with a PLD design.

- Block Diagram File–A design file in which PLD design information is entered as a schematic or as a block diagram.

Quartus II Design Flow

- Design Entry (Block or Text Editor).
- Create Project.
- Assign Target Device (PLD).
- Compile.
- Simulate (if the simulation produces errors, make corrections and recompile).
- Assign input and output pin numbers.
- Recompile.
- Program the target CPLD.

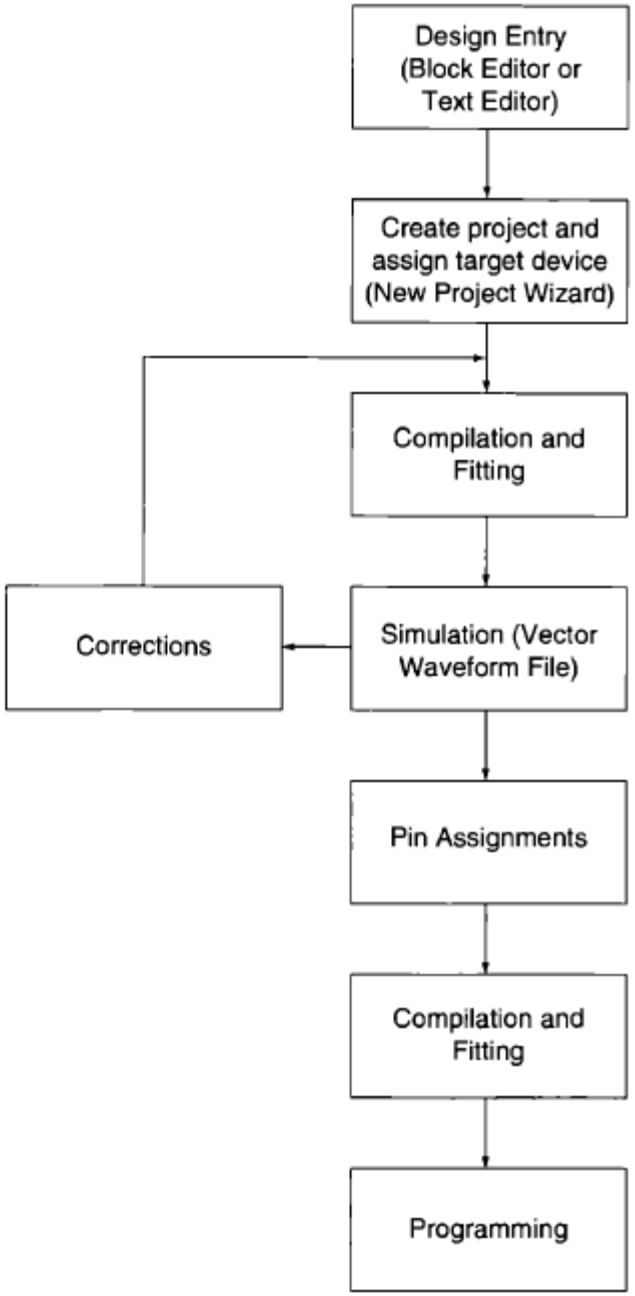
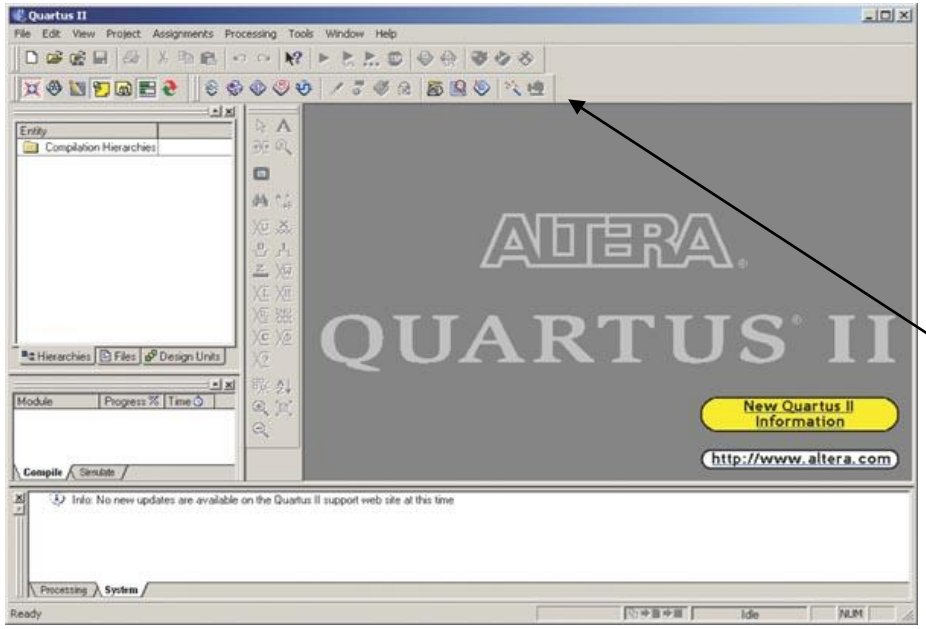
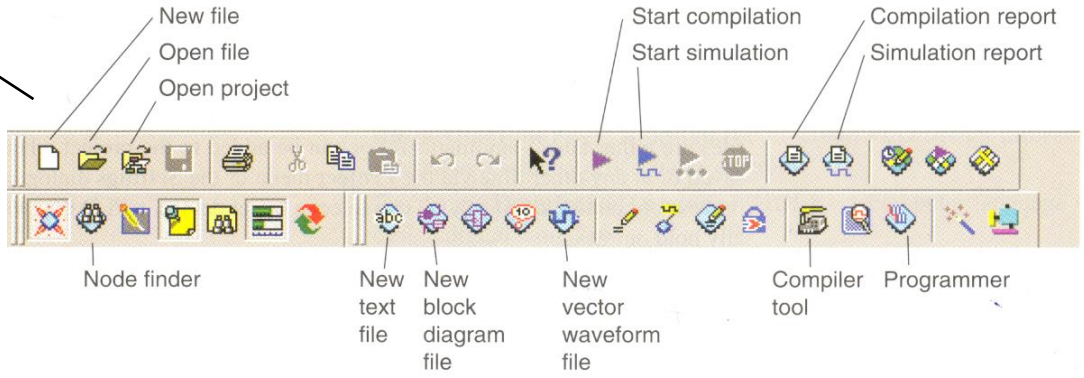


FIGURE 4.15 Simplified Design Flow for Quartus II

Quartus II Start-Up Screen

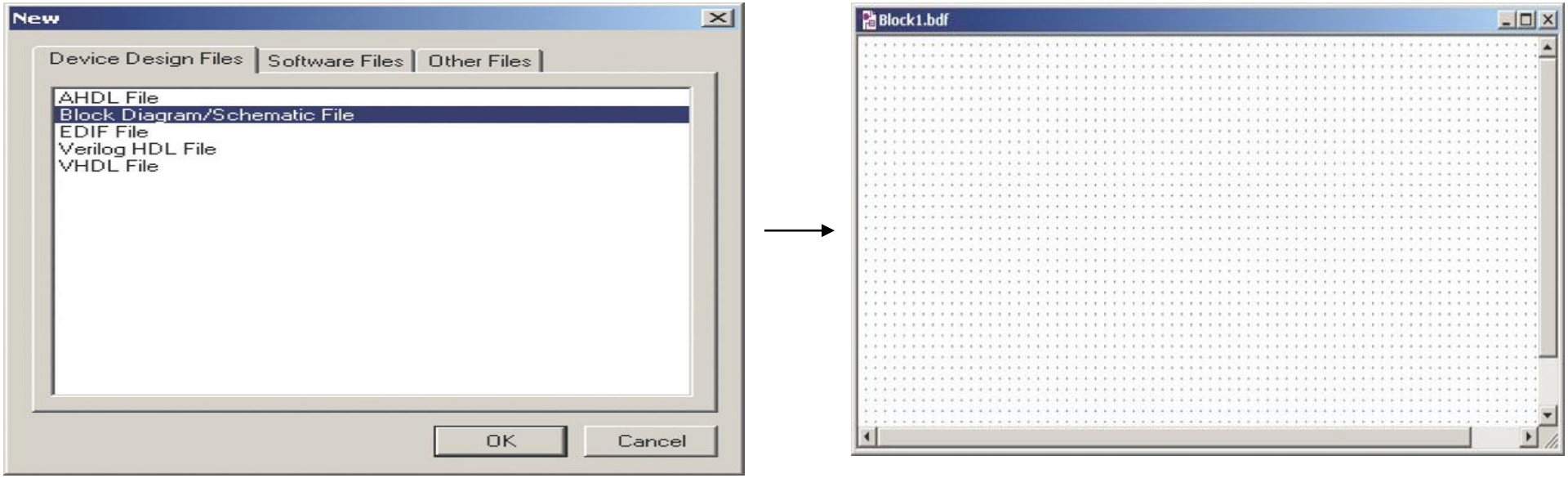
- From the Quartus start-up screen, you can open all other Quartus applications.
- Toolbar provides shortcuts for frequently used functions.





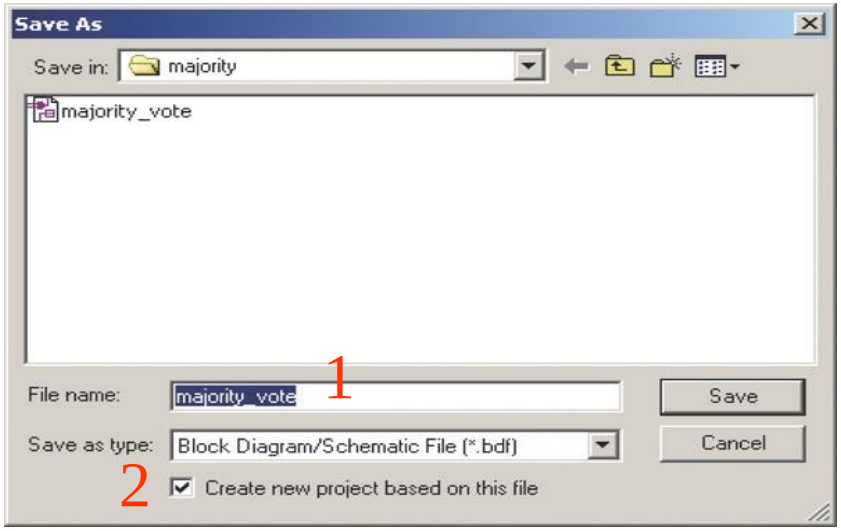
Creating a Block Diagram File

- Select **New** from the **File** menu.
- From the dialog box, choose **Device Design Files** tab.
- Select **Block Diagram/Schematic File**.



Creating a New Project -1

- Must be done before entering any design information.
- Use the **Save As** dialog box.
- Enter the file name.
- Save as type **Block Diagram/Schematic File (*.bdf)**.
- Check the **Create new project based on this file** box.



Creating a New Project -2

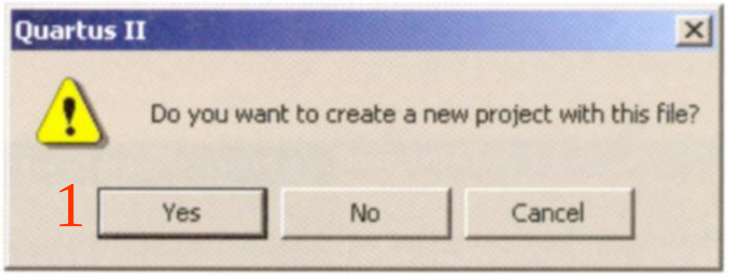
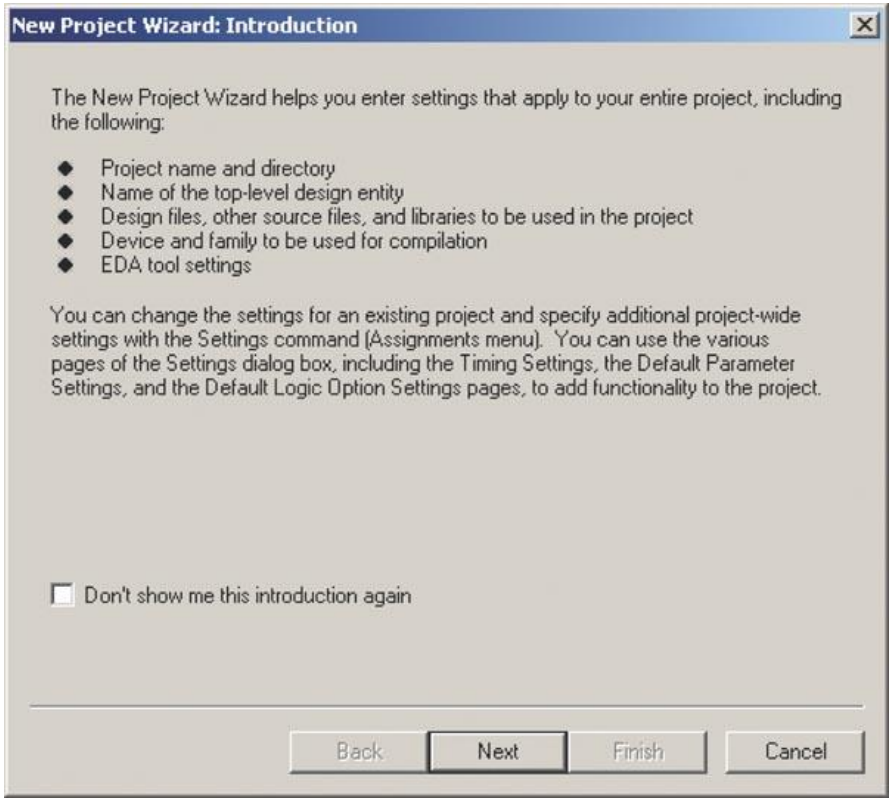


FIGURE 4.21 New Project Query

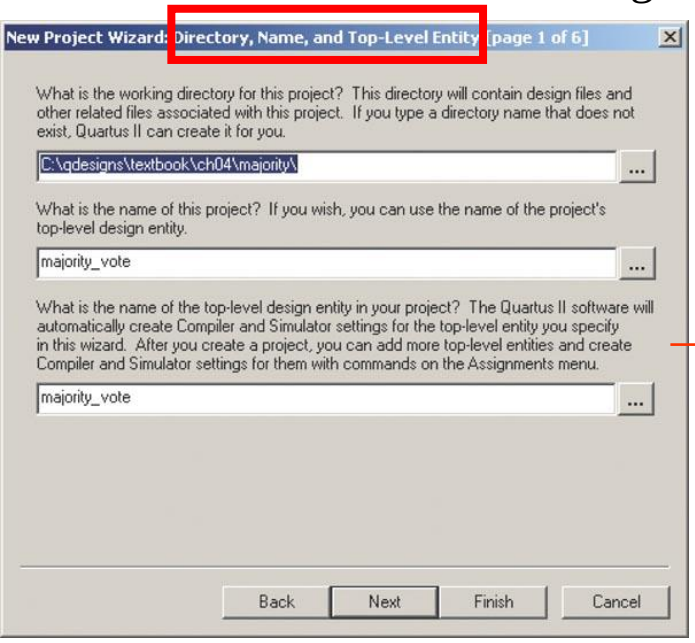
New Project Wizard - 1

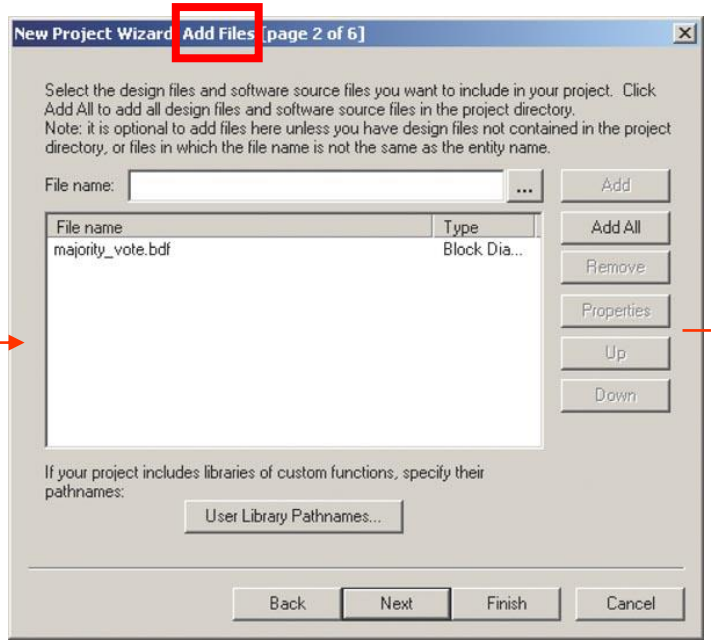
- Allows the user to easily create a new project and assign some of its basic settings.
- Helps the user to establish the basic settings for the project, as well as to add files and user libraries to, or remove them from, the project.

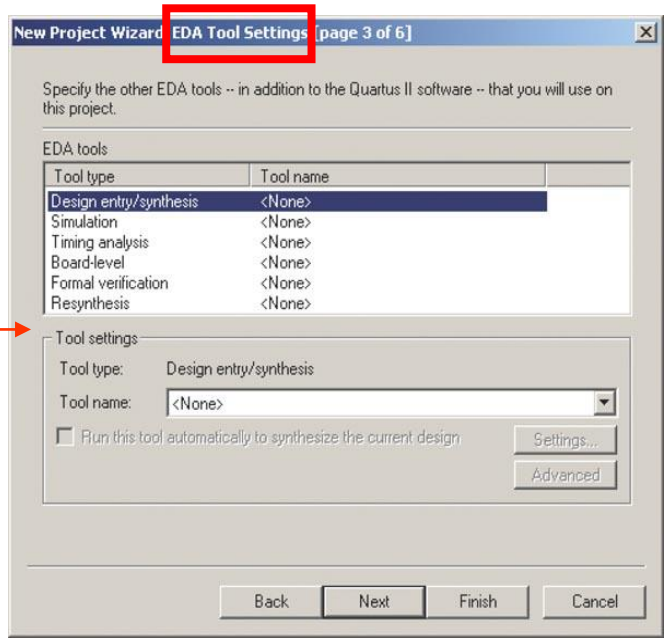


New Project Wizard - 2

- The user can enter:
  - Project name and directory.
  - Name of the top-level design entity.
  - Design files, other source files, and libraries to be used in the project.
  - Device and family to be used.
  - EDA tool settings.







New Project Wizard - 3

The screens in Figures 4.26 and 4.27 select the device family (MAX 7000S) and target device (EPM7128SLC84-7) for the project. The MAX 7000S is a family of sum-of-products CPLDs manufactured by Altera. The EPM7128SLC84-7 is a device in this family with 128 user-programmable locations.

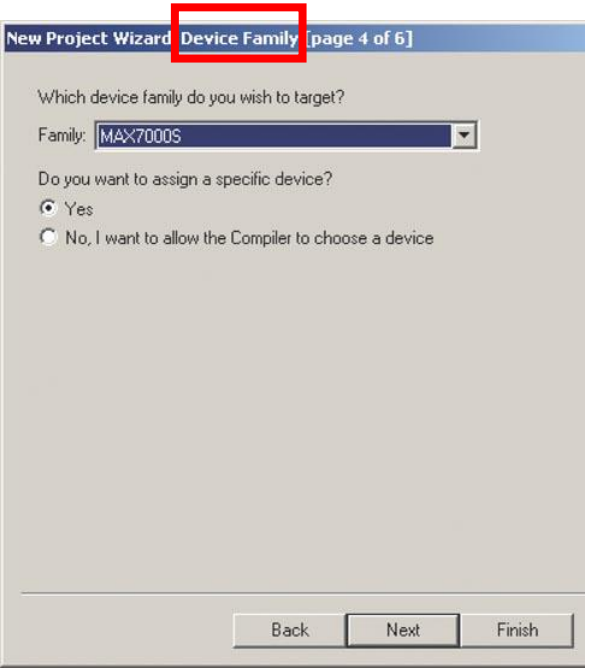


Fig. 4.26

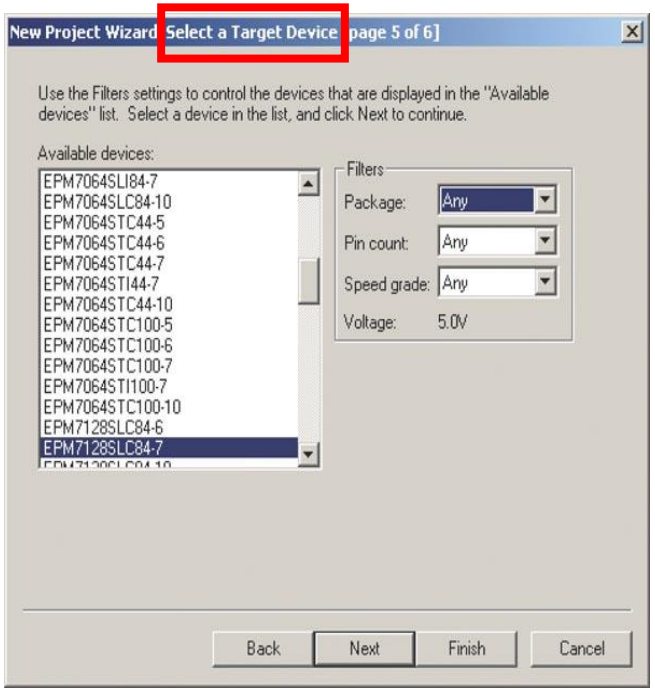


Fig. 4.27

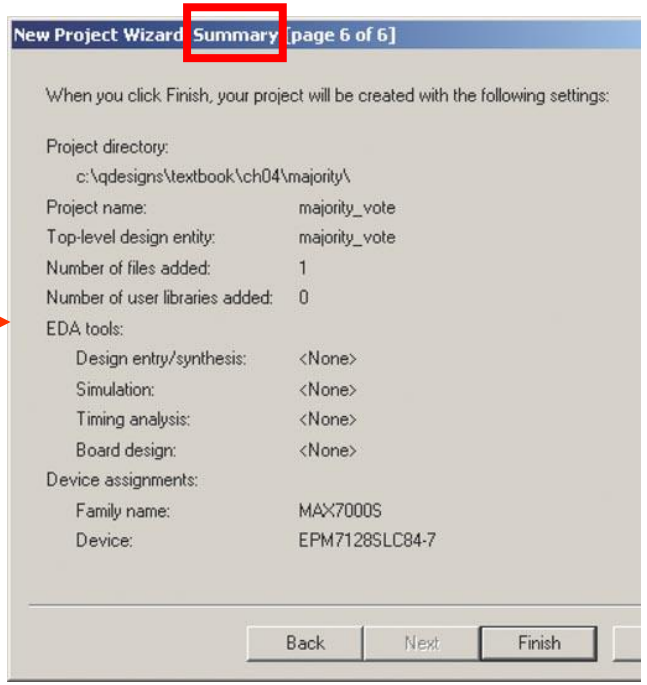
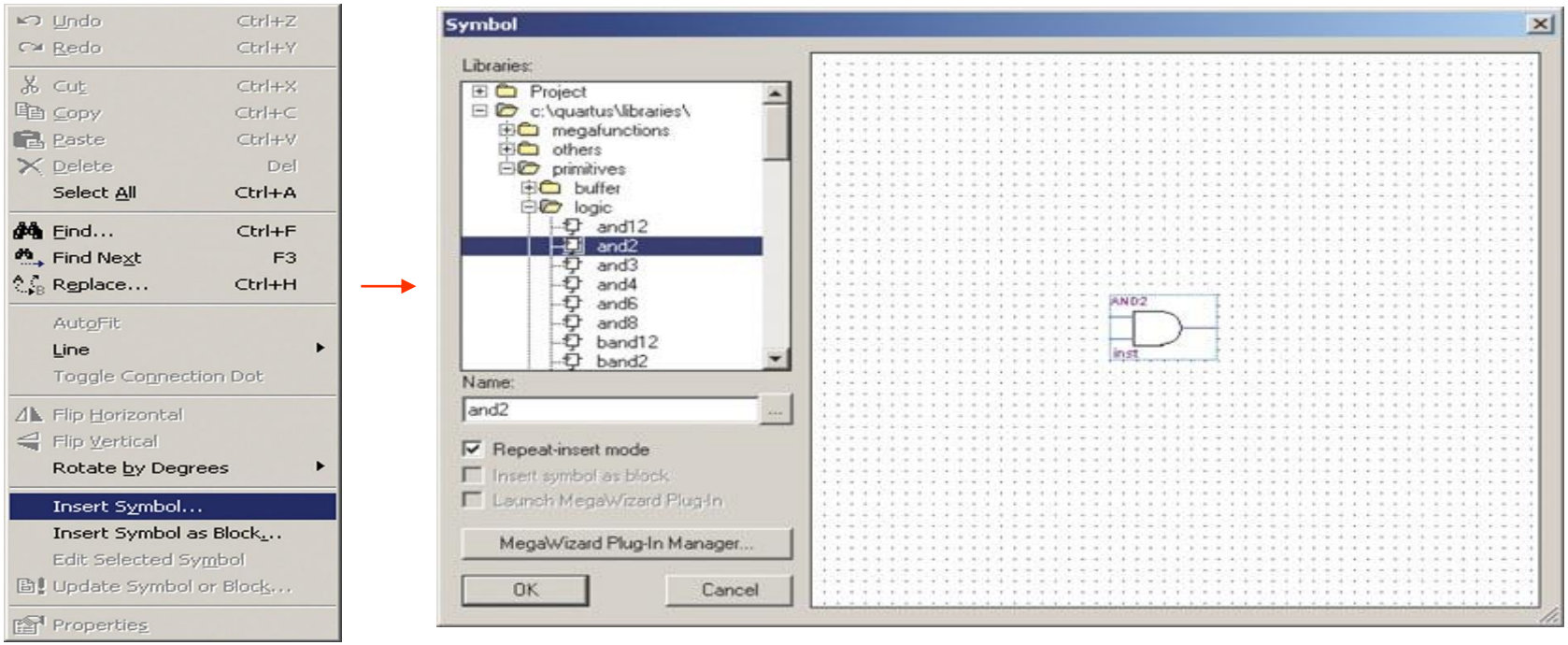


Fig. 4.28

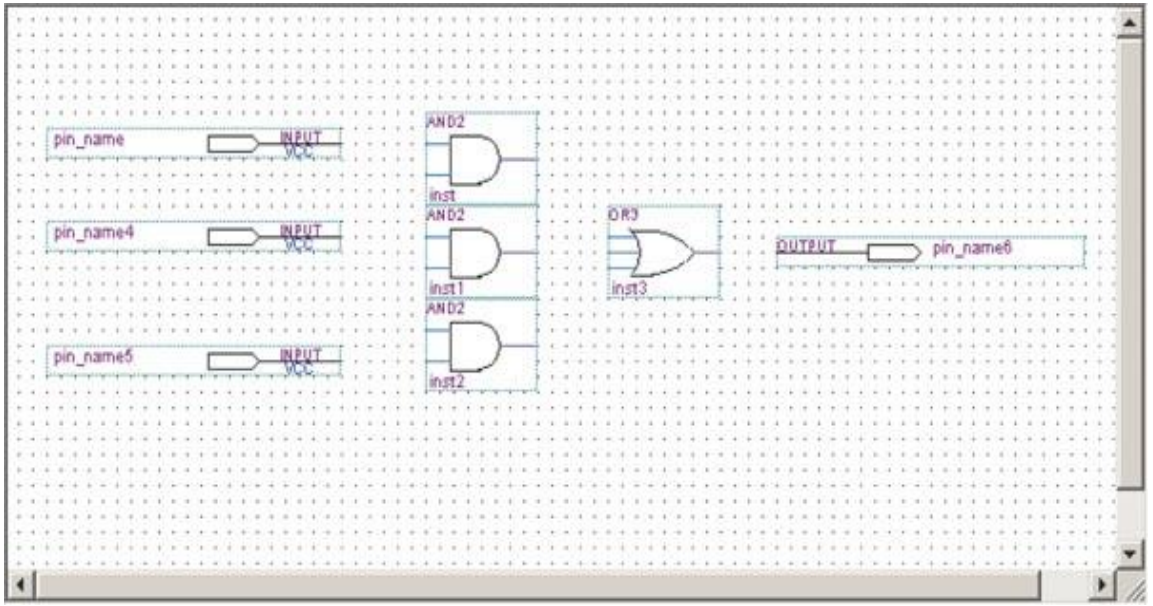
Entering Components

- Open the **Edit** menu and select **Insert Symbol**, or
- Double-click on the **Block Editor** desktop.
- Enter the component name in the **Symbol** dialog box (e.g., ***and2***).



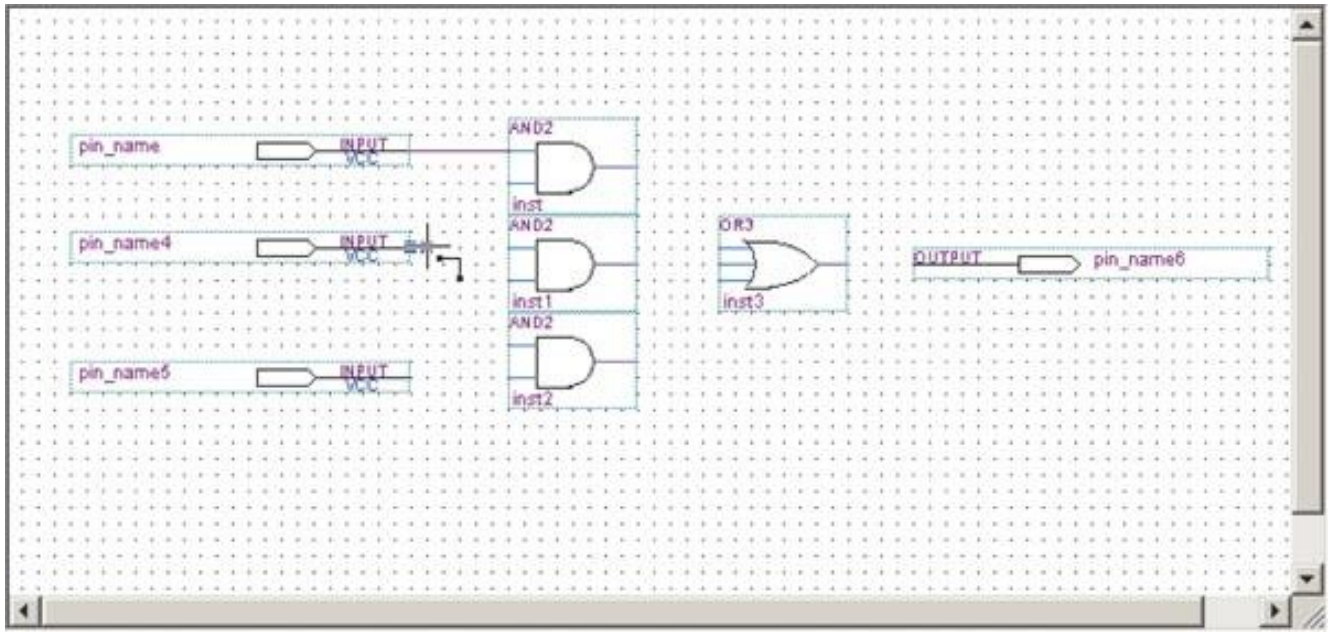
Aligning Components

- Highlight each component.
- Drag each one to the desired location.



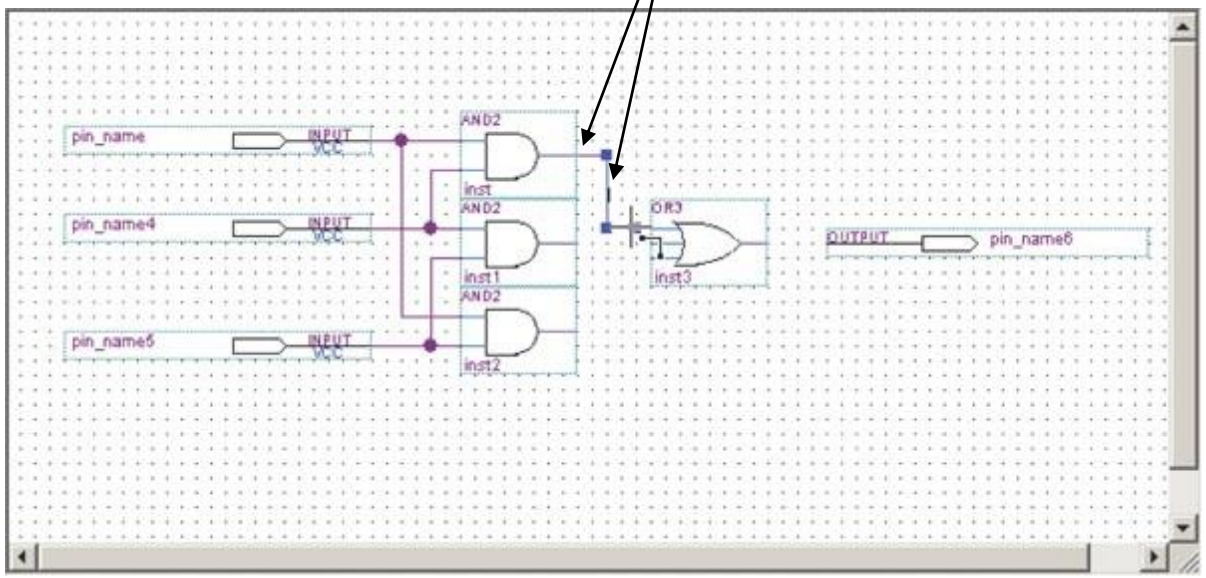
Connecting Components -1

- Components are connected by clicking over one end of one component and dragging a line to one end of a second component.
- Hover over a line causes the cursor to change from an arrow to a cross-hair with a right angle symbol.



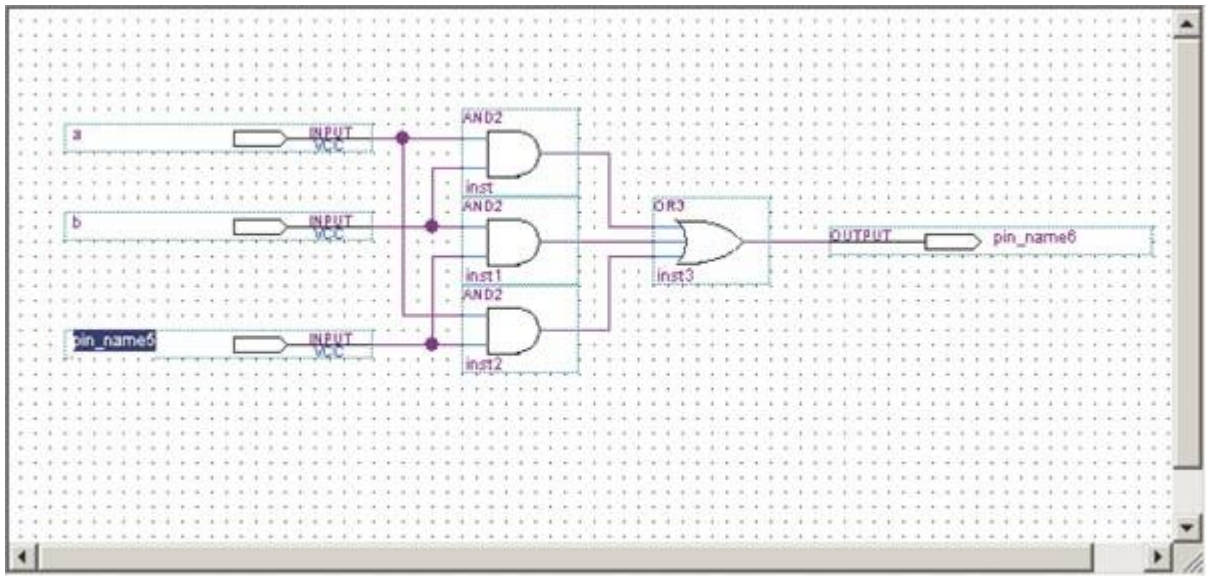
Connecting Components -2

Two 90° bends require two separate lines



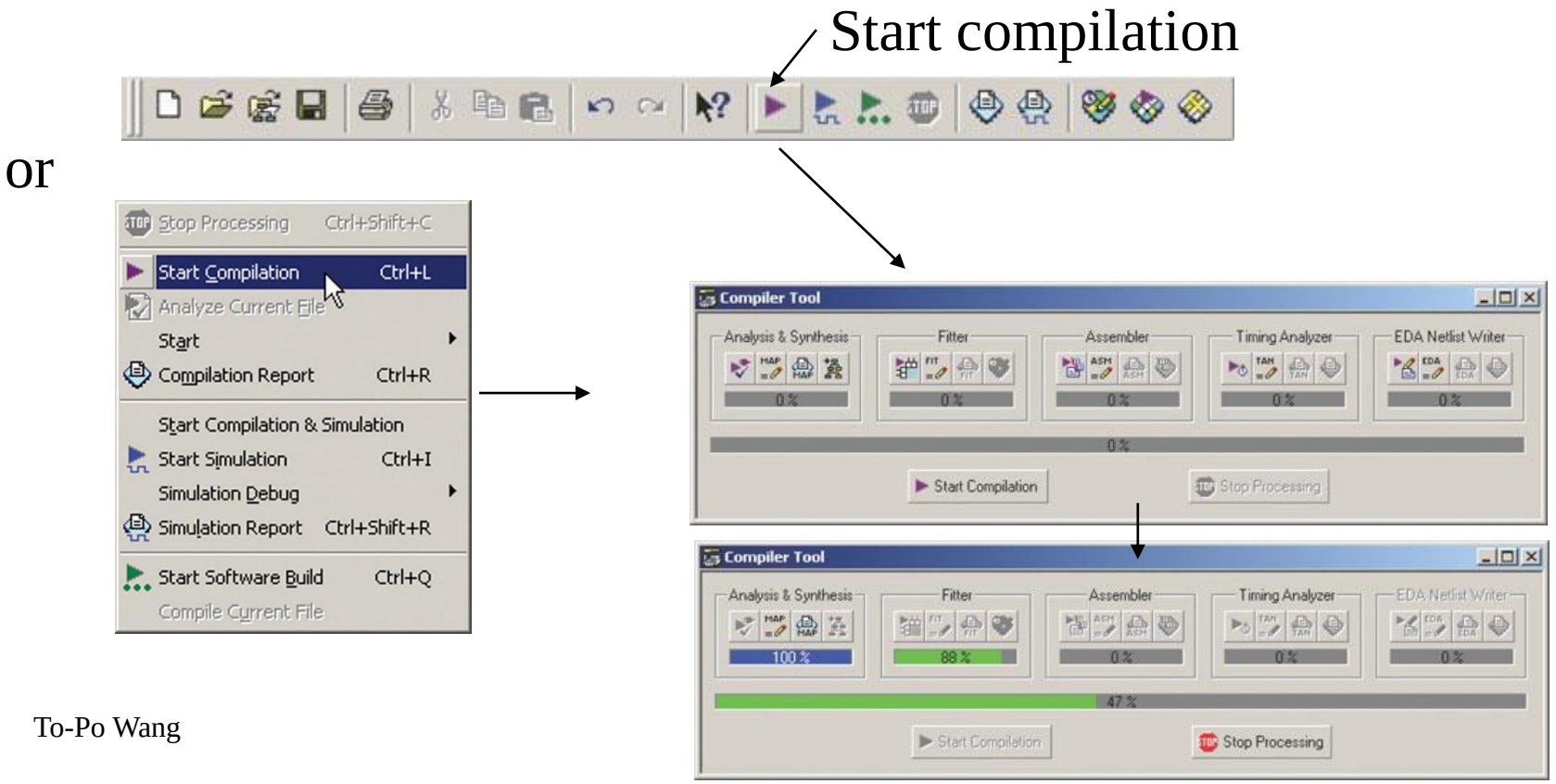
Assigning Pin Names

- Inputs and outputs must be assigned names.
- Double-click the pin name to highlight the name.
- Type in the pin name.



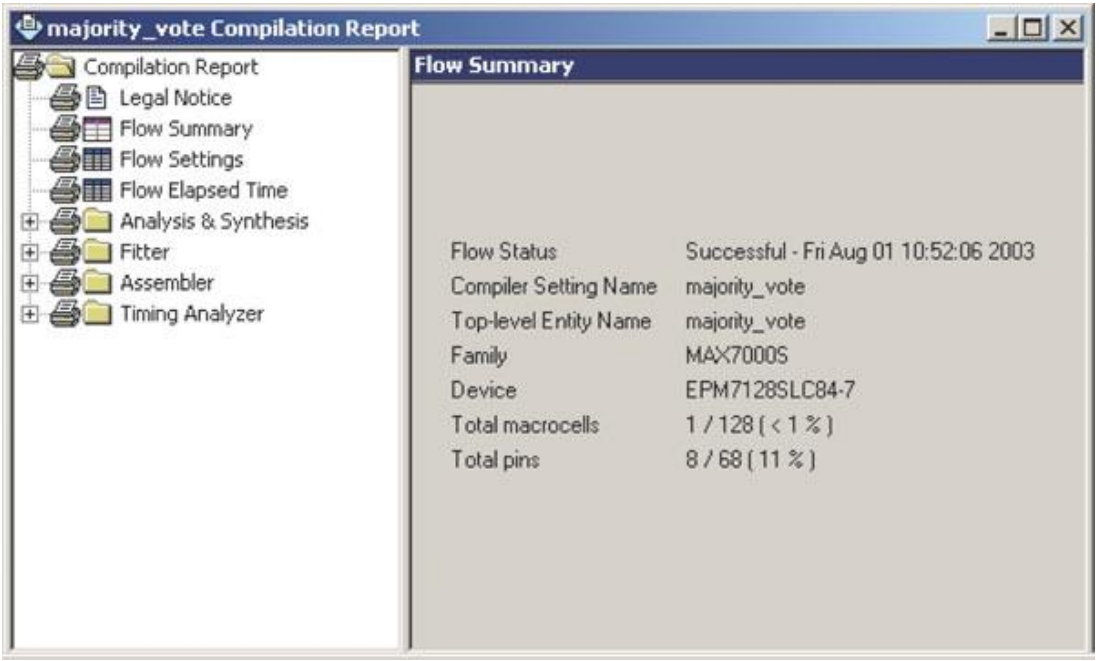
4.8 Compiling and Simulating a Design in Quartus II

- Click the **Start Compilation** button on the toolbar, or
- Select **Start Compilation** from the **Processing** Menu, or
- Open the Compiler tool from the **Tools** menu, and click **Start Compilation**.



Compilation Report

- A summary of the compilation.
- More details are available under the various folders.
- Legal Notice, Flow Setting, etc.
- Folders can be expanded by clicking on the (+) symbol.



Simulating a Design in Quartus II

- Simulation is based on a Vector Waveform file (vwf).
- The Vector Waveform file contains simulation input and output values defined as graphical waveforms.
- The input and output values are defined by the designer.

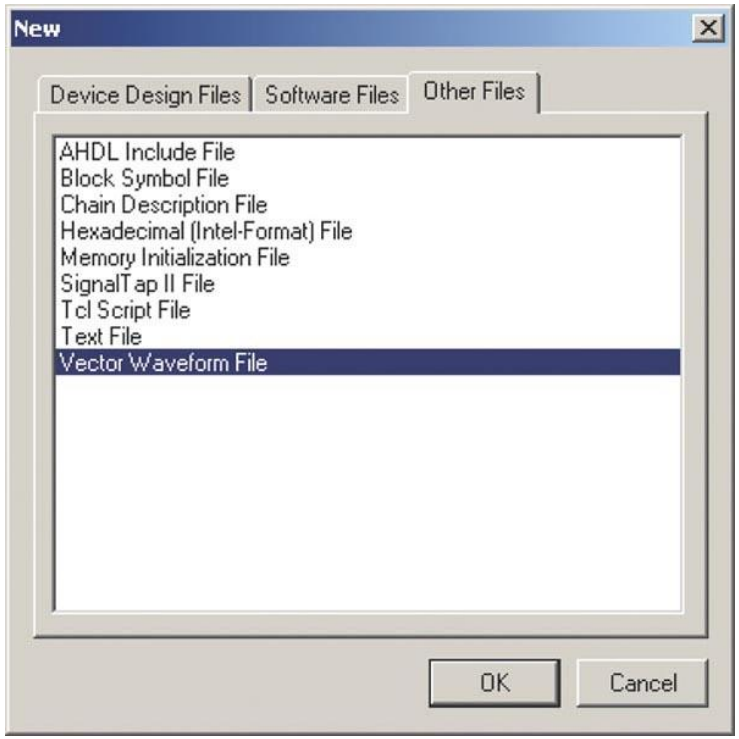
The Quartus II Simulator

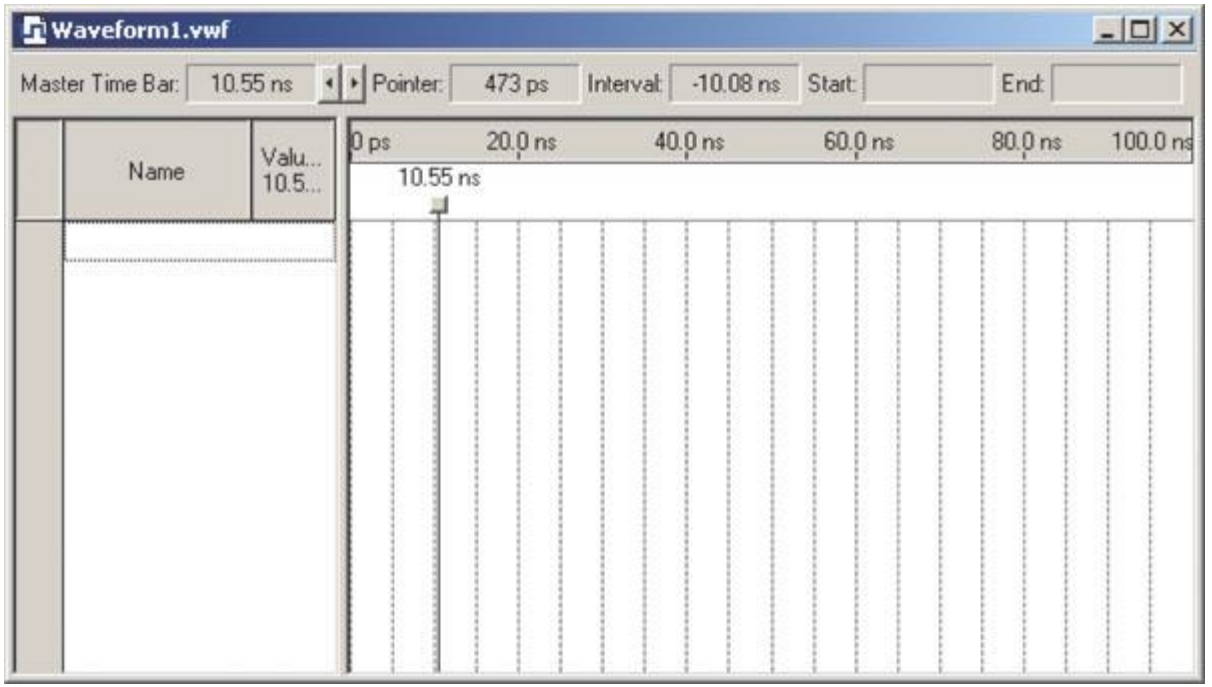
- Performs a functional simulation to test the logical operation of your design, or

- Performs a timing simulation to test both the logical operation and the worst-case timing for the design in the target device.

Creating a Vector Waveform File

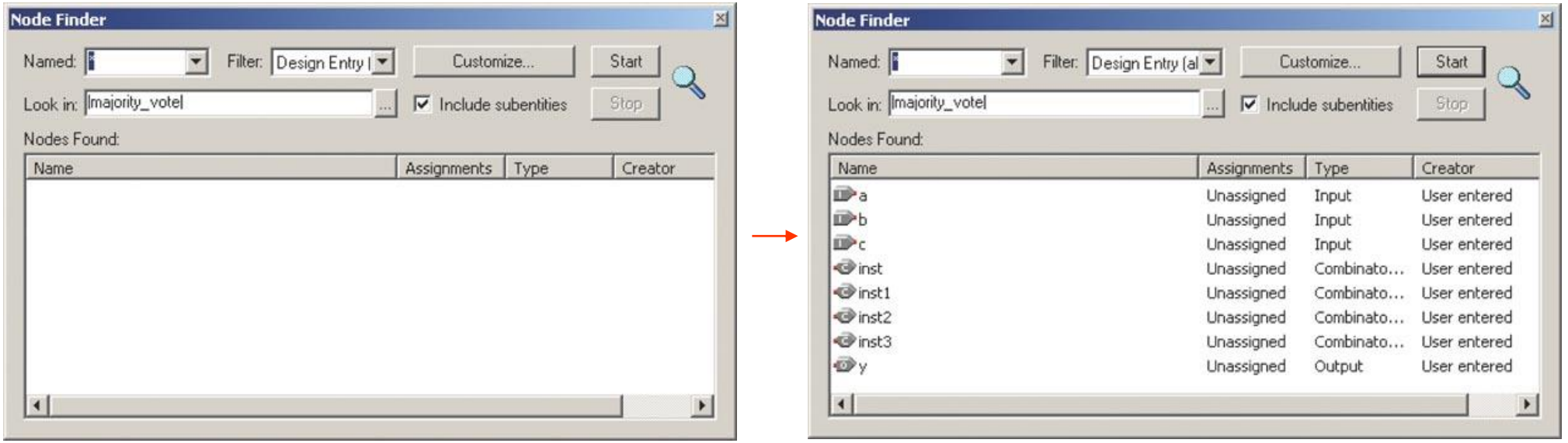
- Click **New** from the **File** menu or select the appropriate toolbar button.
- In the **New File** dialog box, select the **Other Files** tab.
- Select **Vector Waveform File**.





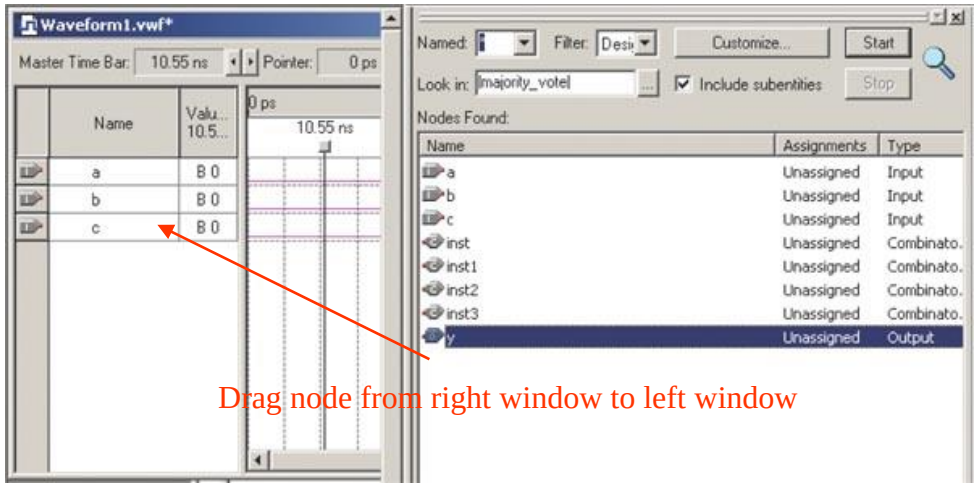
Entering Waveforms in the Editor

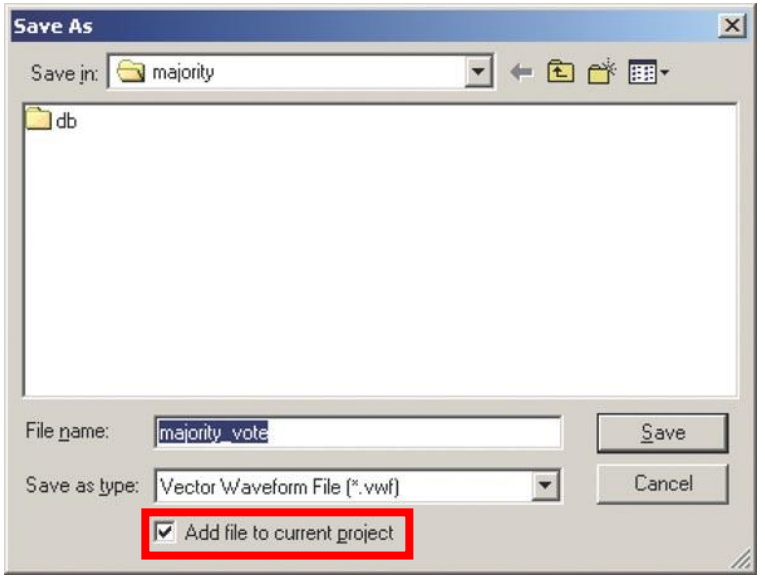
- Start the **Node Finder** by clicking on the toolbar button or by selecting **Utility Windows/Node Finder** from the **View** menu.
- Click **Start** to display the list of nodes.



Adding the Required Nodes

- In the **Node Finder**, click on the waveform to highlight it.
- Drag and drop the waveform in place in the **Waveform Editor** window.
- Once all waveforms have been added, **Save** and **Close** the **Node Finder** window.



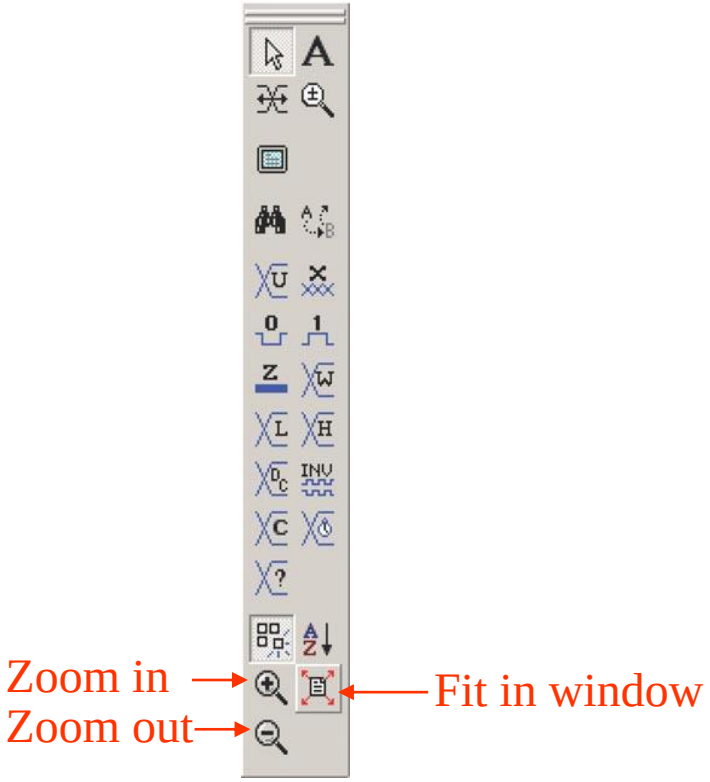


Simulation Time

- Determine the length of time our simulation should run.
- The default time is 100 ns.
- Each CPLD has a delay time from input to output specified in nanoseconds.
- This is indicated by the number at the end of the CPLD's part number.
- The simulation time should be long compared to the input-output delay time of the CPLD.
- Select **End Time** from the **Edit** menu.
- In the **End Time** dialog box, change the unit from nS (nanoseconds) to μS (microseconds).

Viewing the Waveform

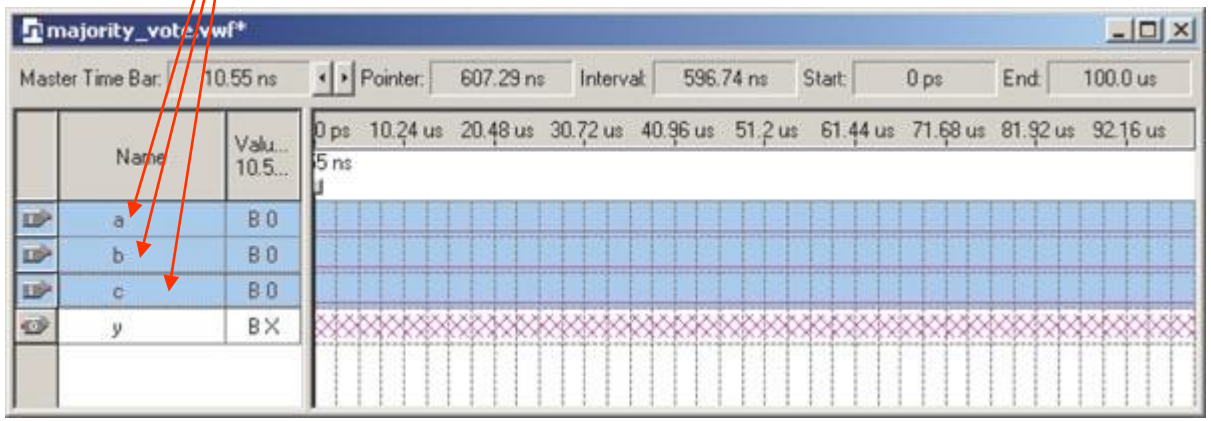
- The entire waveform can be viewed by selecting **Fit in Window** from the **View** menu, or
- Using the **Zoom In**, **Zoom Out**, and **Fit in Window** buttons on the **Waveform Editor** toolbar.



Grouping Waveforms – 1

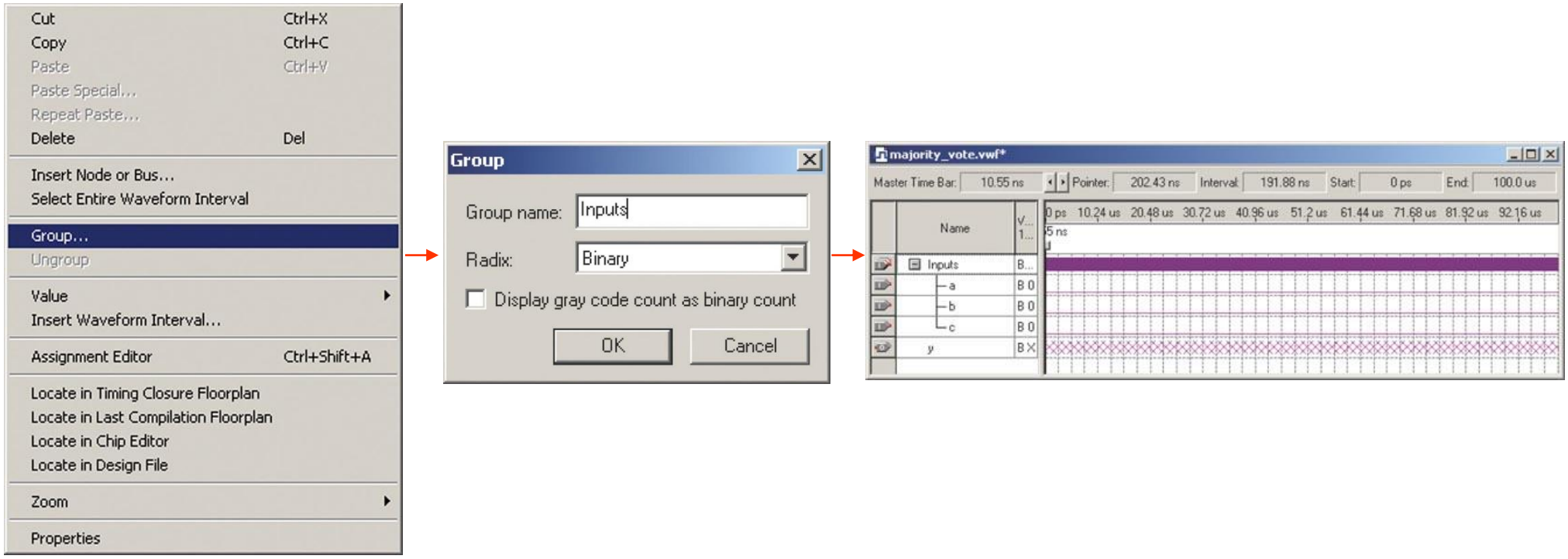
- Waveforms can be entered individually or as a group of waveforms.
- To highlight a group, click on the top waveform and then drag the cursor to the last waveform of the group.

Click an drag from a to c to highlight the group



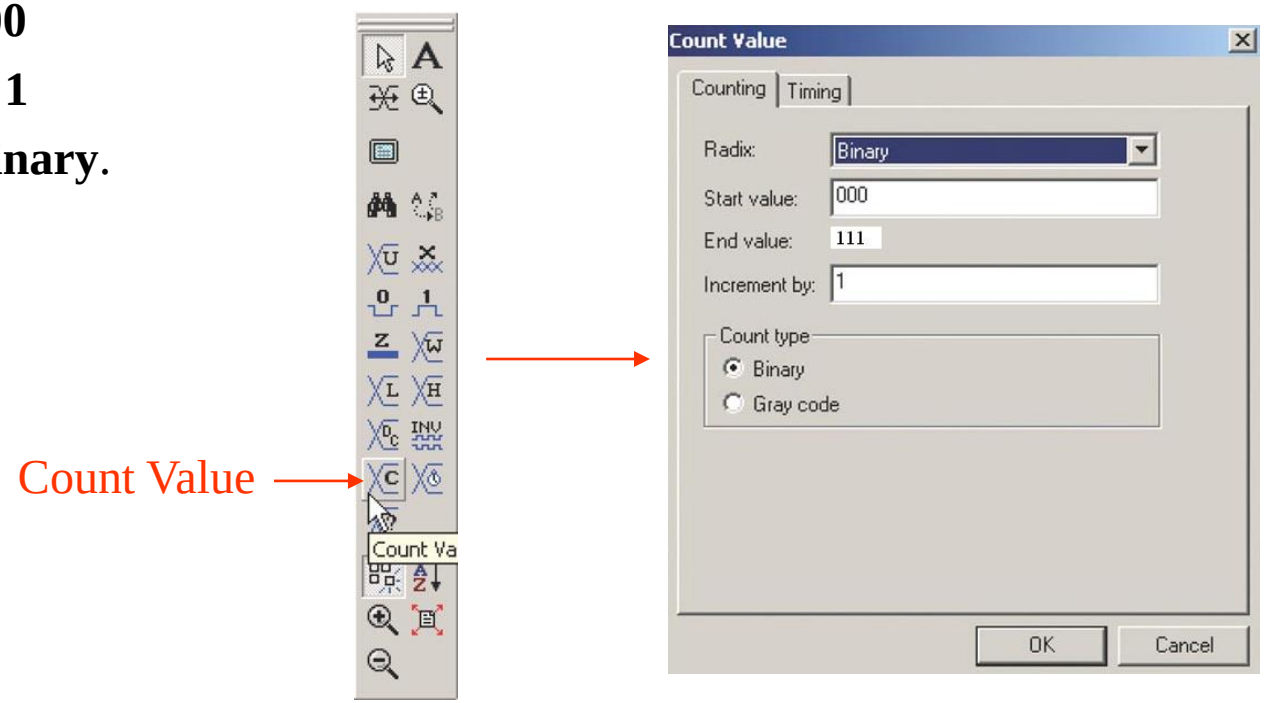
Grouping Waveforms – 2

- Right-click on the selected group of waveforms and select **Group**.
- Enter the group name **Inputs**.
- Select the radix as **Binary**.
- Click **OK**.



Counting Waveform

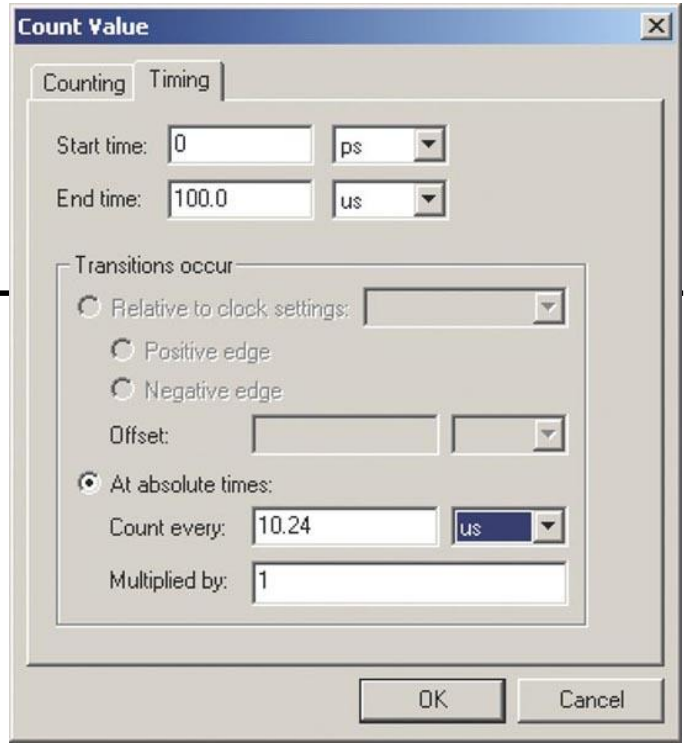
- Click on the waveform group **Inputs**.
- Click the **Count Value** toolbar button.
- In the **Counting** tab of the **Count Value** dialog box, select
  - **Radix: Binary**
  - **Start value: 000**
  - **Increment by: 1**
  - **Count type: Binary**.

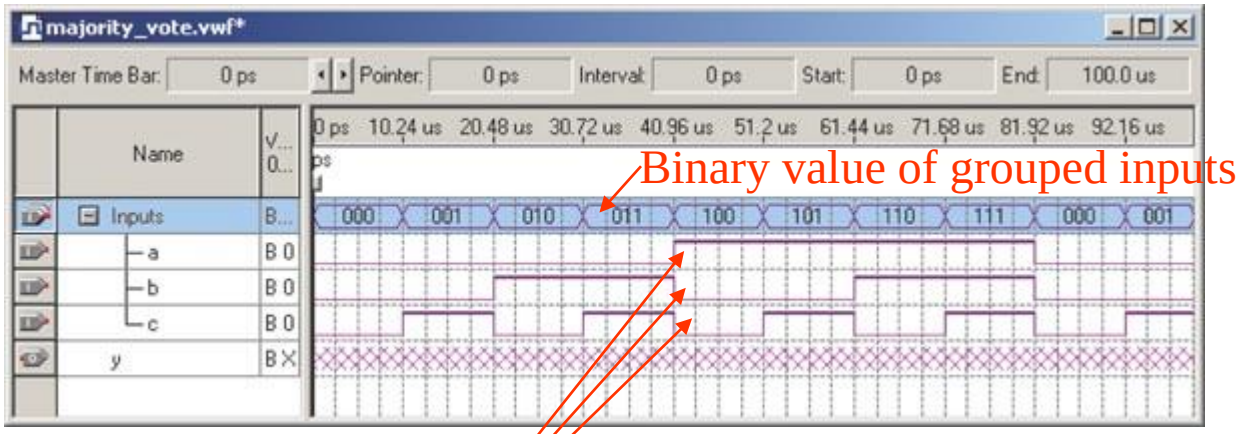


Timing Waveform

- Select
  - **Start time: 0 ps**
  - **End time: 100 µS**
  - **Count every: 10.24 µS**
  - **Multiplied by: 1**

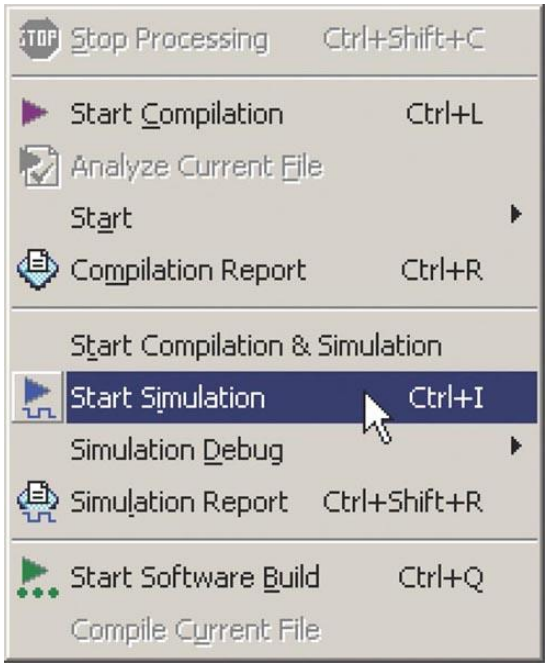
- Count interval of 10.24 µS matches the spacing of the Waveform Editor timing grid.





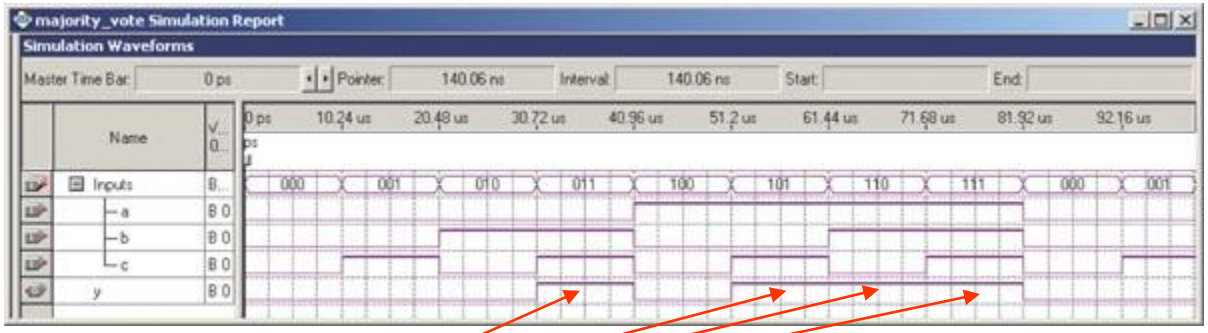
Starting the Simulation

- Select **Start Simulation** from the toolbar, or
- Select **Start Simulation** from the **Processing** menu.



or

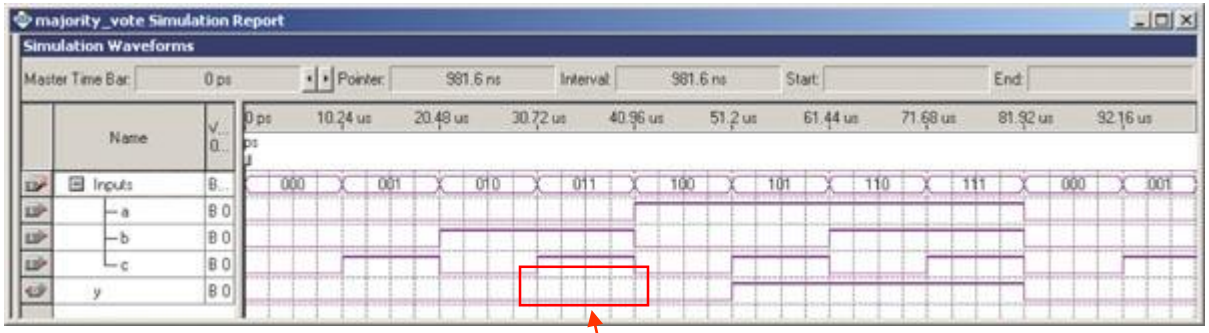
Simulation Results



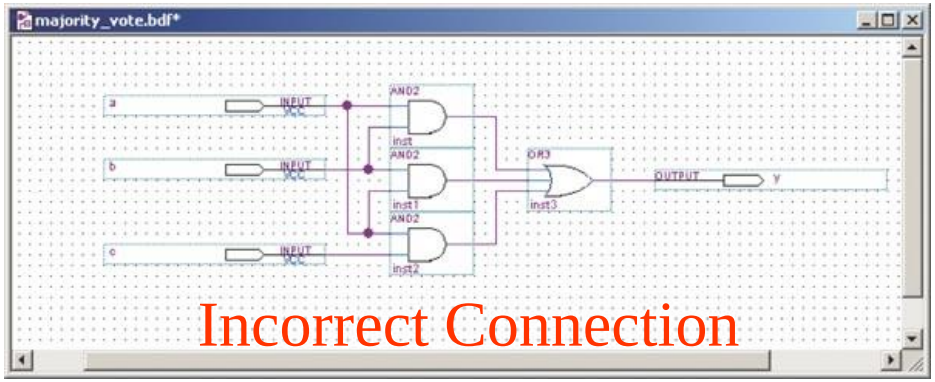
Output HIGH when two or more inputs are HIGH

Using the Simulator for Troubleshooting

- Simulation is a valuable tool that provides the designer with the means of finding errors in the design entry of a project.
- Should errors be identified through the simulation process, they must then be corrected before programming the design on a CPLD.



Output should be HIGH



Incorrect Connection

4.9 Transferring a Design to a CPLD

- Assign pin numbers to each assigned pin name.
- Assign the Device.
- Recompile the file.
- Use the Quartus programming tool to transfer the design from the PC to the CPLD.

HW