ECE 85L Digital Logic Design Laboratory

Fresno State, Lyles College of Engineering
Electrical and Computer Engineering Department

Spring 2018

Laboratory 9 – HDL and Arithmetic Circuits

1. OBJECTIVES

- Introduce students to the Altera Quartus II CAD Tool
- Design and Construct a 1-Bit Full Adder Circuit
- Perform the Arithmetic Sum of Two Binary Numbers
- Perform the Subtraction of Binary Numbers using the 2's Complement Technique

2. DISCUSSION

In this experiment, you will construct and simulate a 1-Bit Full Adder Circuit, using the Altera Quartus II CAD Tool. When arithmetic operations are implemented in a computer, it is more convenient to use a system referred to as the signed-complement system for representing negative numbers. Therefore, in a digital system or in a computer, a negative number is indicated by its complement. Subtraction of two binary numbers when negative numbers are in 2's complement form is simple. The procedure can be stated as follows:

1. Take the 2's complement of the subtrahend (bottom number).
2. Add it to the minuend (top number).
3. Overflow indicates that the answer is positive. Ignore the overflow.
4. No overflow indicates that the answer is negative. Take the 2's complement of the original sum to obtain the true magnitude of the answer.

3. PRELAB

1. A half adder is a circuit that has two inputs, A and B, and two outputs, sum and carry. It adds A and B according to the rules of binary addition and outputs the sum and carry. Design a half-adder circuit using one XOR gate and one AND gate. Verify your design through truth table and with Multisim.

2. Whereas the half adder added two inputs A and B, the full adder adds three inputs together, A, B, and a carry from a previous addition, and outputs a sum and carry.

Design a full adder circuit to AND, OR, and XOR gates. Verify its operation through truth table and with Multisim.

4. LAB ASSIGNMENT

4.1 Field Programmable Gate Array (FPGA) Devices

This section introduces the student to the Altera Quartus II CAD Tool. Students will learn how to design and simulate circuits using the Quartus II CAD Tool environment, targeting a Field Programmable Gate Array (FPGA). An FPGA device is and integrated circuit which provides a circuit designer with the flexibility of modifying the implemented hardware through re-programming the FPGA device. This potential of changing hardware implementations with little effort is important for quick prototyping and testing of hardware designs, as well as, for applying new upgrades to the implemented hardware with minimal cost. A designer should target the specific FPGA device which exists in the used FPGA development board. For this lab experiment, there is no specific target device. The FPGA device used in this laboratory experiment is the Cyclone IV E series EP4CE115F29C7.

4.2 Design Flow of the Quartus II CAD Tool

The Quartus II CAD Tool will be used to enter and simulate designs of arithmetic circuits targeting the Cyclone IV E series EP4CE115F29C7 FPGA device. Students will use the tool to enter their HDL based design and simulate it. The following is a brief introduction to the design flow that will be used to implement circuits onto the FPGA device. Figure 4.1 presents the general flow of implementing circuits on an FPGA device using the Quartus II CAD Tool.

4.3 Quartus II 1-Bit Full Adder Implementation

In this section, students will be exposed to the Quartus II tool and will exercise the top part of the design flow presented in the previous section of this document (up to behavioral simulation), through designing and simulating a full bit adder circuit. The design in this section will be used for designing a 4-bit binary adder/subtractor in a later section.

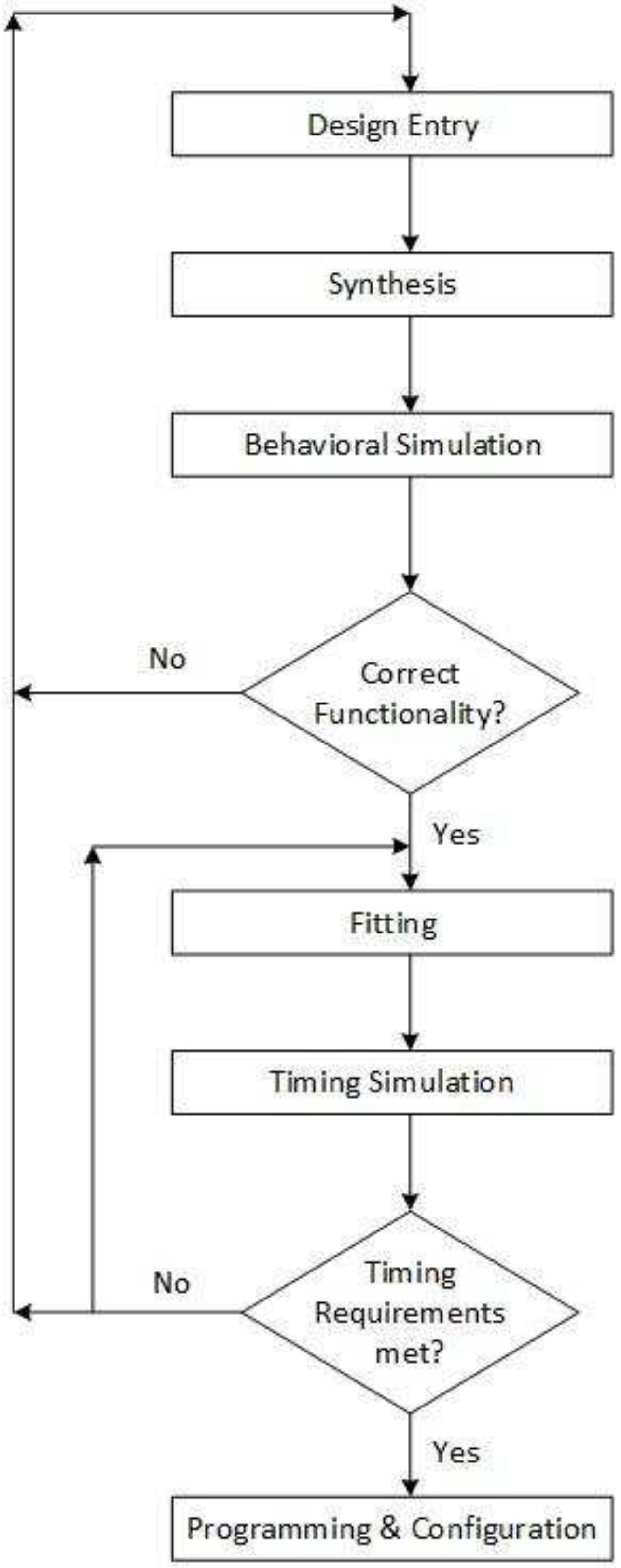


Figure 4.1: General Design Flow

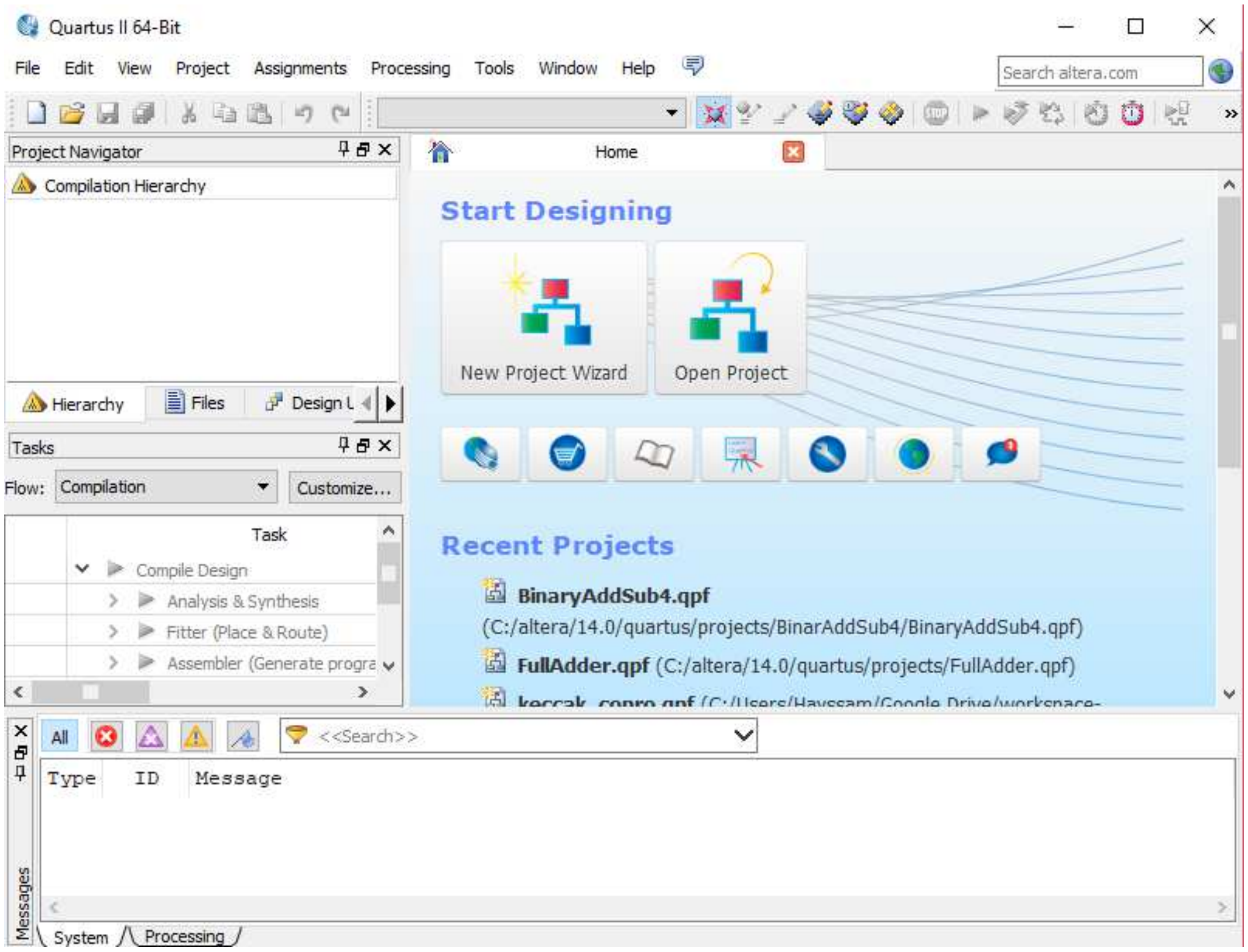


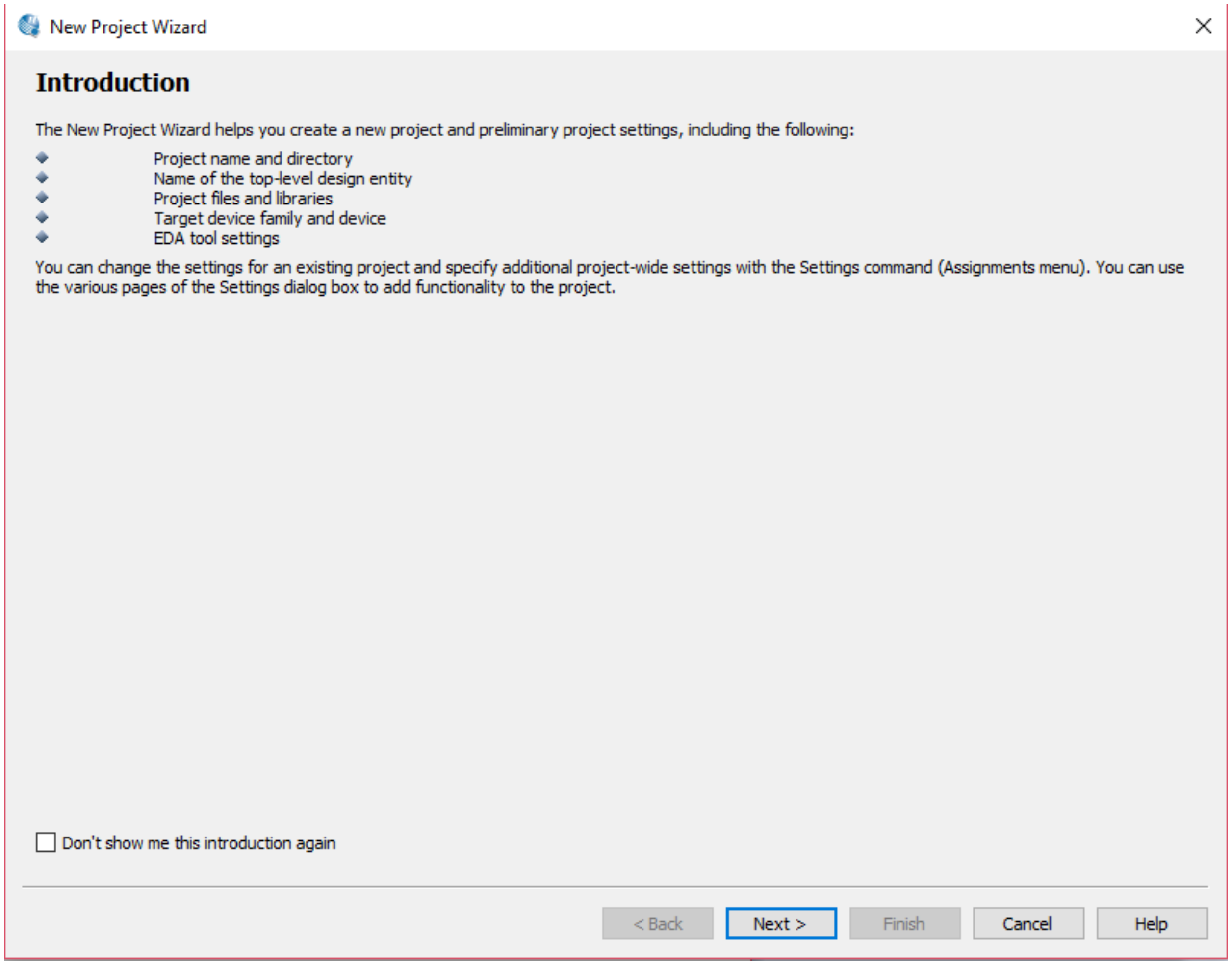
Figure 4.2: Quartus II IDE

4.3.1 Quartus II 1-Bit Full Adder Synthesis

The procedure for designing the 1-bit full adder is as follows:

a) Create a new project:

   a. Start the Quartus II CAD Tool. The following IDE will be displayed (Figure 4.2).


New Project Wizard

Introduction

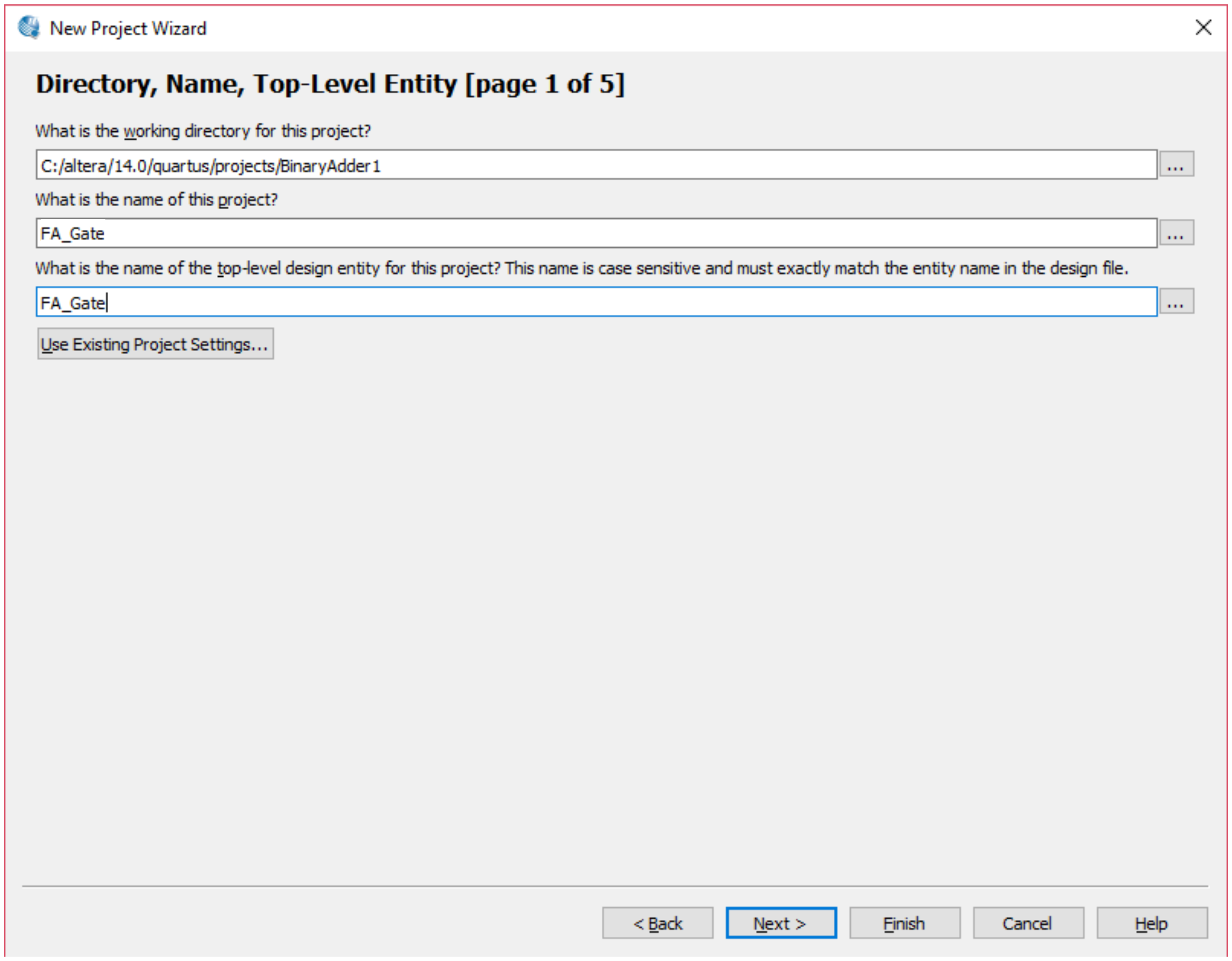
The New Project Wizard helps you create a new project and preliminary project settings, including the following:

- Project name and directory
- Name of the top-level design entity
- Project files and libraries
- Target device family and device
- EDA tool settings

You can change the settings for an existing project and specify additional project-wide settings with the Settings command (Assignments menu). You can use the various pages of the Settings dialog box to add functionality to the project.

Figure 4.3: Project Wizard Dialog Window

b. From the File menu, select the New Project Wizard option, this will open a new project wizard dialog window (Figure 4.3).



Directory, Name, Top-Level Entity [page 1 of 5]

What is the working directory for this project?

C:/altera/14.0/quartus/projects/BinaryAdder1

What is the name of this project?

FA_Gate

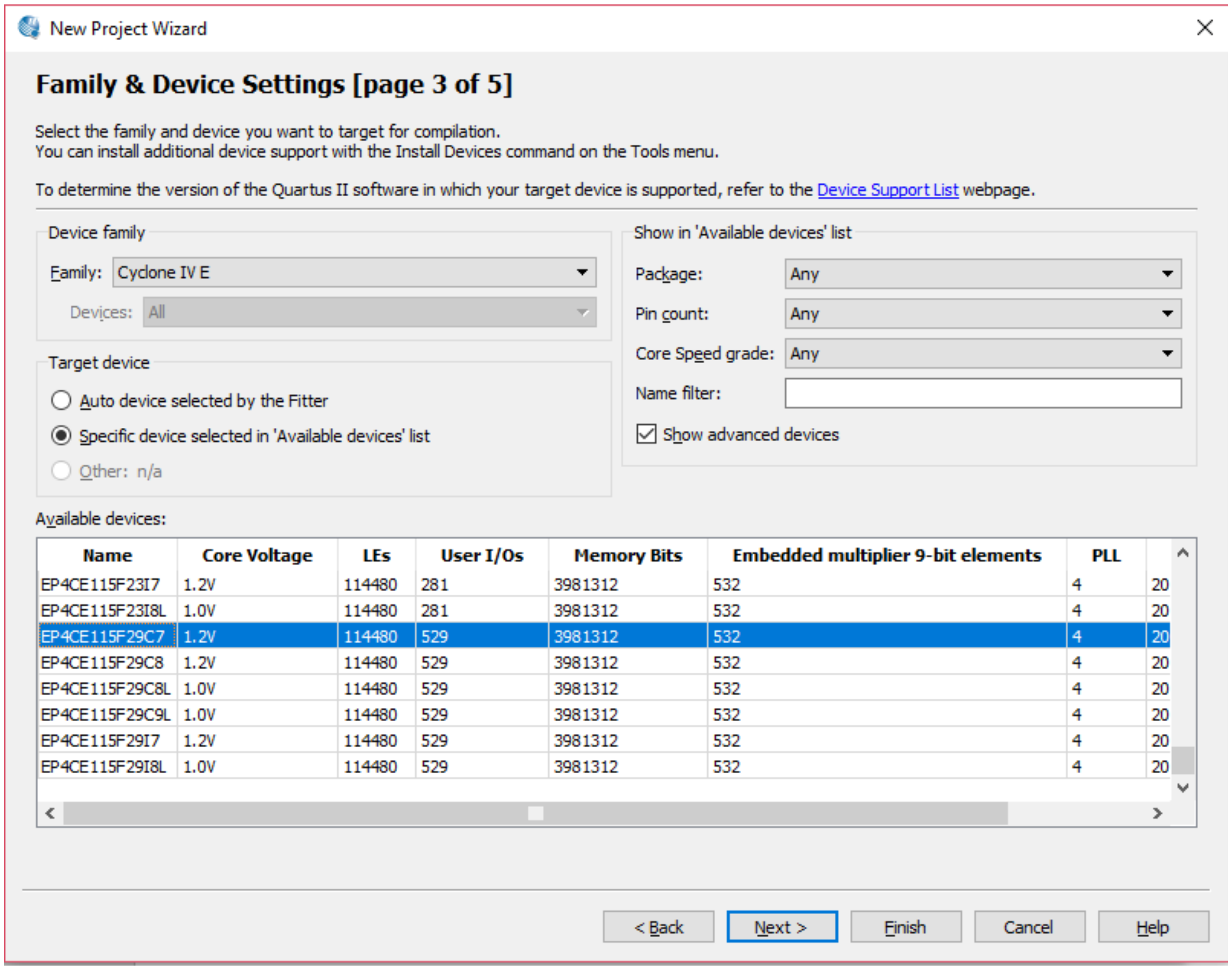
What is the name of the top-level design entity for this project? This name is case sensitive and must exactly match the entity name in the design file.

FA_Gate

Figure 4.4: Directory, Name, Top-Level Entity Project Wizard Window

c. Click Next on the Introduction page of the Wizard window, the Directory, Name, Top Level Entity page will be displayed (Figure 4.4).

d. Select the working directory (a location of your choice), assign a name to the project (FA_Gate), and indication the name of the top-level entity of your design (FA_Gate). Click Next.



Family & Device Settings [page 3 of 5]

Select the family and device you want to target for compilation.
You can install additional device support with the Install Devices command on the Tools menu.

To determine the version of the Quartus II software in which your target device is supported, refer to the Device Support List webpage.

Device family

Family: Cyclone IV E

Devices: All

Target device

( ) Auto device selected by the Fitter

(●) Specific device selected in 'Available devices' list

( ) Other: n/a

Show in 'Available devices' list

Package: Any

Pin count: Any

Core Speed grade: Any

Name filter:

[✓] Show advanced devices

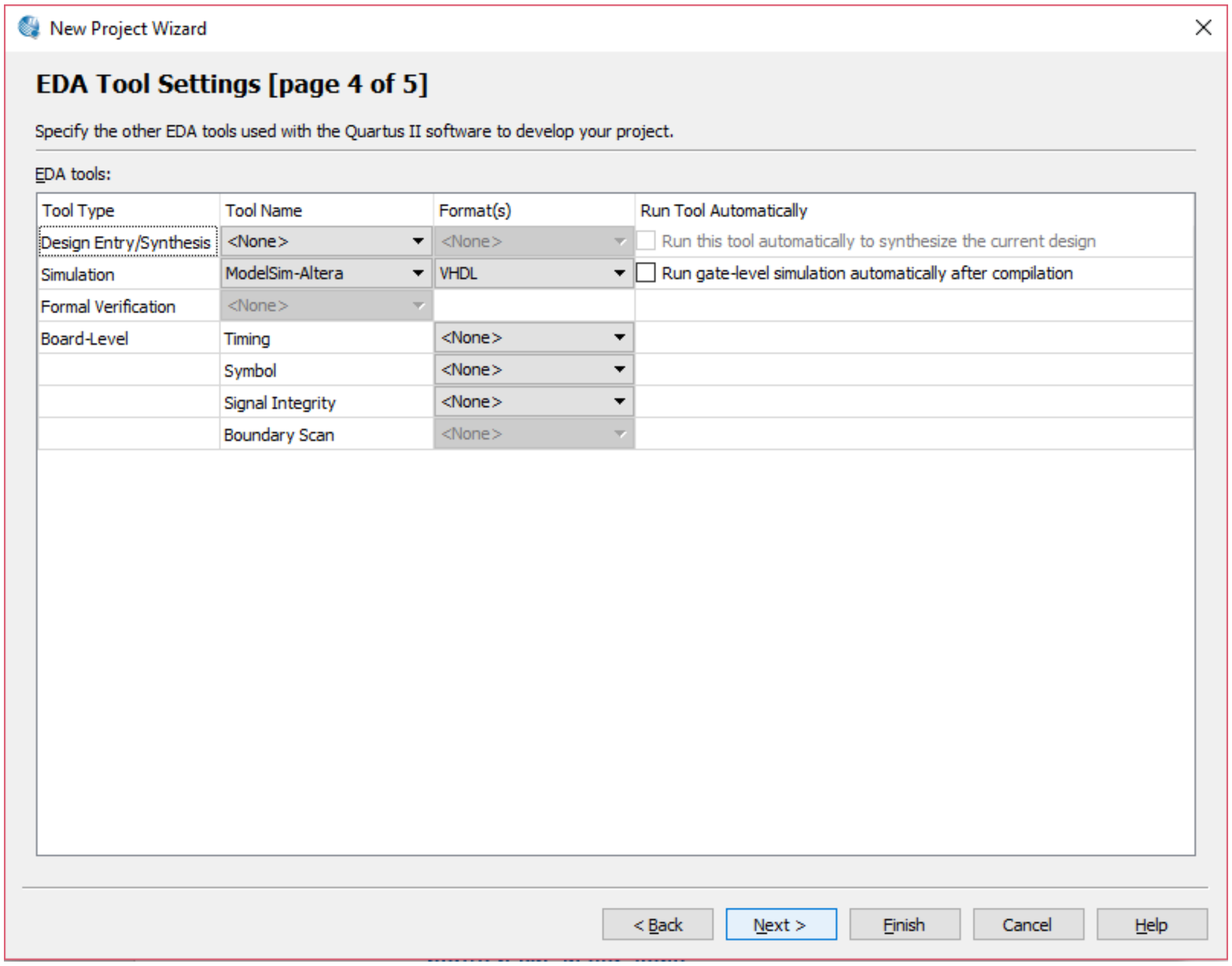
Available devices:

Name	Core Voltage	LEs	User I/Os	Memory Bits	Embedded multiplier 9-bit elements	PLL	
EP4CE115F23I7	1.2V	114480	281	3981312	532	4	20
EP4CE115F23I8L	1.0V	114480	281	3981312	532	4	20
EP4CE115F29C7	1.2V	114480	529	3981312	532	4	20
EP4CE115F29C8	1.2V	114480	529	3981312	532	4	20
EP4CE115F29C8L	1.0V	114480	529	3981312	532	4	20
EP4CE115F29C9L	1.0V	114480	529	3981312	532	4	20
EP4CE115F29I7	1.2V	114480	529	3981312	532	4	20
EP4CE115F29I8L	1.0V	114480	529	3981312	532	4	20

Figure 4.5: Family & Device Settings Project Wizard Window

e. In the add files page, Click Next, since we don't have any files to be added to the project.

f. In the family and Device Settings page, select the Cyclone IV E EP4CE115F29C7 device and click Next (Figure 4.5).


New Project Wizard

EDA Tool Settings [page 4 of 5]

Specify the other EDA tools used with the Quartus II software to develop your project.

EDA tools:

Tool Type	Tool Name	Format(s)	Run Tool Automatically
Design Entry/Synthesis	<None>	<None>	☐ Run this tool automatically to synthesize the current design
Simulation	ModelSim-Altera	VHDL	☐ Run gate-level simulation automatically after compilation
Formal Verification	<None>		
Board-Level	Timing	<None>	
	Symbol	<None>	
	Signal Integrity	<None>	
	Boundary Scan	<None>	

Figure 4.6: EDA Tool Settings Project Wizard Window

g. In the EDA Tools Settings page, leave default options and click Next (Figure 4.6).

h. Click Finish in the Summary page. Notice the .qpf file that is now created under the working directory of your project.

b) Add a Verilog design of the full adder to the project:

a. From the File menu, select New, this opens a dialog box showing a list of options. Select the Verilog File option under Design Files and click OK (Figure 4.7).

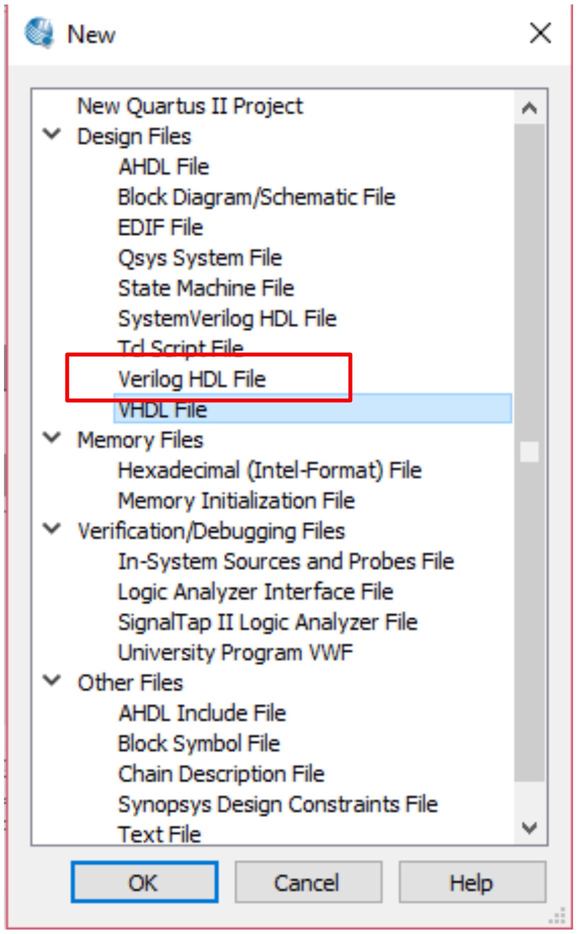


Figure 4.7: HDL File Options

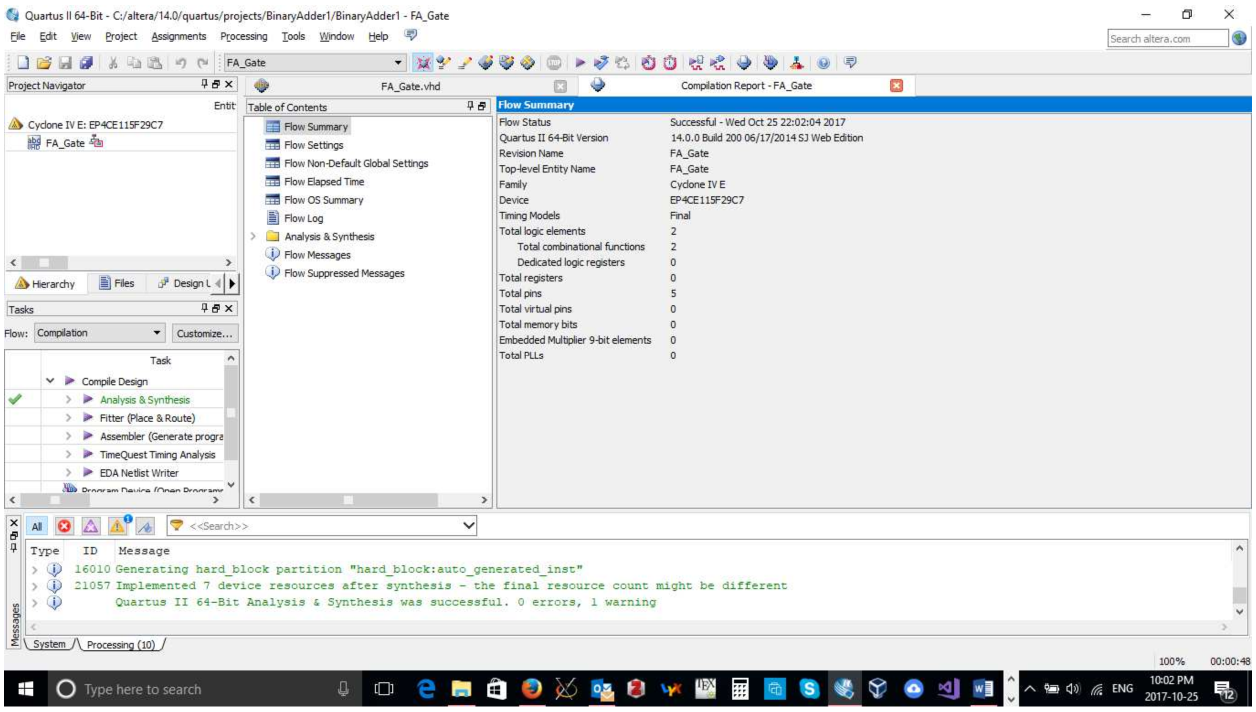
b. Add the following Verilog code to the newly added Verilog file:

```
module FA_Gate (carryout, sum, a, b, carryin);
        output carryout, sum;
        input a, b, carryin;

        assign sum = a ^ b ^ carryin;
        assign carryout = (a & b) | (carryin & a) | (carryin & b);
endmodule
```

c. Save the file as FA_Gate.v

c) Analyzing and Synthesizing the Design: From the Processing menu, select Start → Start Analysis & Synthesis. After Analysis & Synthesis is completed, a summary report will be displayed reporting some useful information about your design up to this point. In addition, At the bottom of the window, helpful messages will be generated specifying different information, errors, and warnings, if any. Usually, errors should be fixed, however, warnings can be ignored (Figure 4.8).



Flow Summary

Flow Status	Successful - Wed Oct 25 22:02:04 2017
Quartus II 64-Bit Version	14.0.0 Build 200 06/17/2014 SJ Web Edition
Revision Name	FA_Gate
Top-level Entity Name	FA_Gate
Family	Cyclone IV E
Device	EP4CE115F29C7
Timing Models	Final
Total logic elements	2
Total combinational functions	2
Dedicated logic registers	0
Total registers	0
Total pins	5
Total virtual pins	0
Total memory bits	0
Embedded Multiplier 9-bit elements	0
Total PLLs	0

```
Type  ID    Message
      16010 Generating hard_block partition "hard_block:auto_generated_inst"
      21057 Implemented 7 device resources after synthesis - the final resource count might be different
            Quartus II 64-Bit Analysis & Synthesis was successful. 0 errors, 1 warning
```

Figure 4.8: Analysis and Synthesis of the Design

<u>NOTE: The above code should not generate errors. In case you have any errors, you need to fix the errors and repeat this step.</u>

4.3.2 Behavioral (Functional Simulation)

This section utilizes the University Program VWF tool to conduct functional simulation of the full adder design in the previous section. More advanced tools such as ModelSim can also be used for this purpose. To conduct functional simulation on the full adder circuit, proceed as follows:

a) From the File menu, select New, this opens the new file dialog box (Figure 4.9).

b) Select University Program VWF option under Verification/Debugging and click OK.

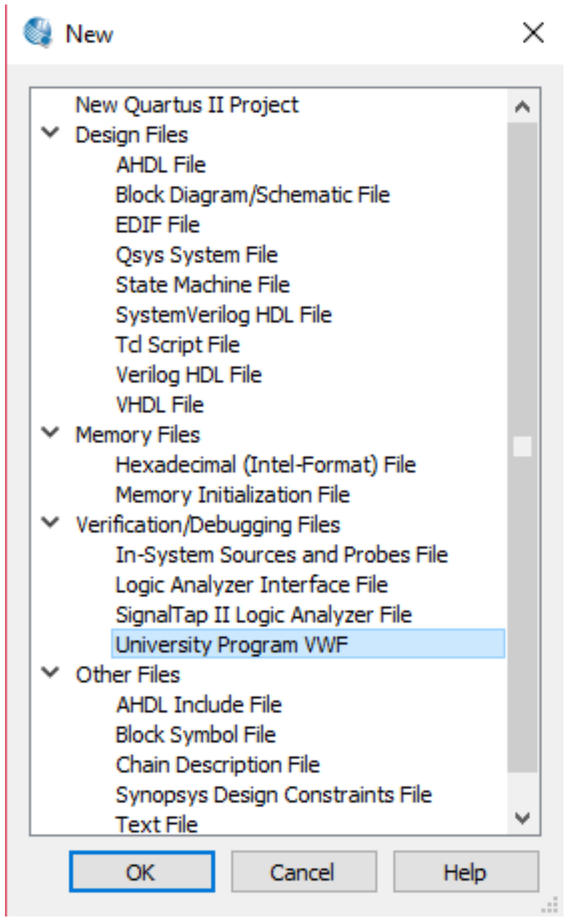


Figure 4.9: New File Dialog Box Window

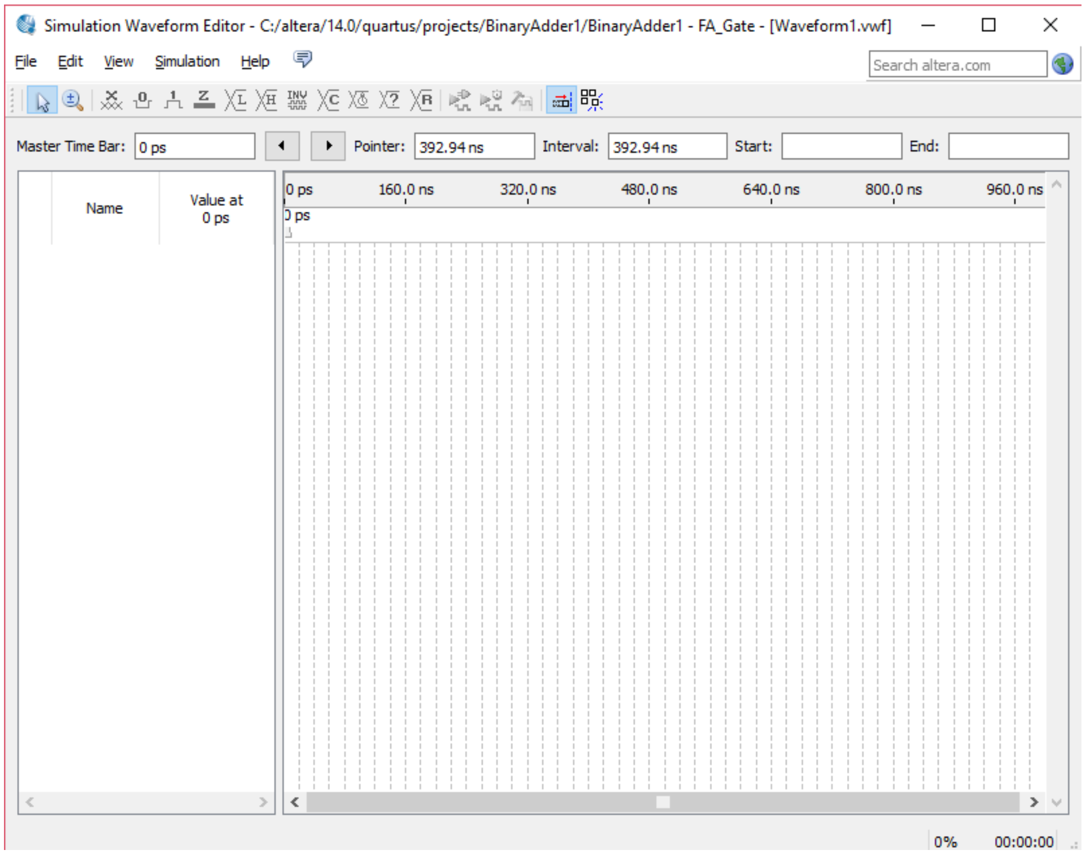
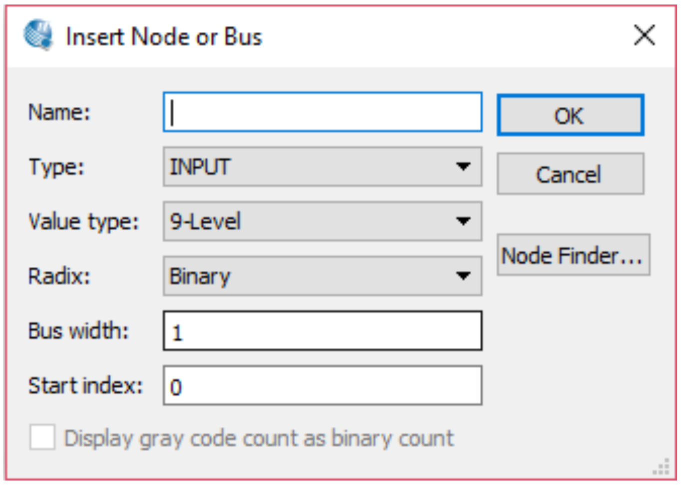


Figure 4.10: Simulation Waveform Editor Window

A new Simulation Waveform Editor window will open (Figure 4.10).

c) Right-click on the left pane of the Editor window and select Insert Node or Bus. An Insert Node or Bus dialog box will be displayed (Figure 4.11).



Name:

Type: INPUT

Value type: 9-Level

Radix: Binary

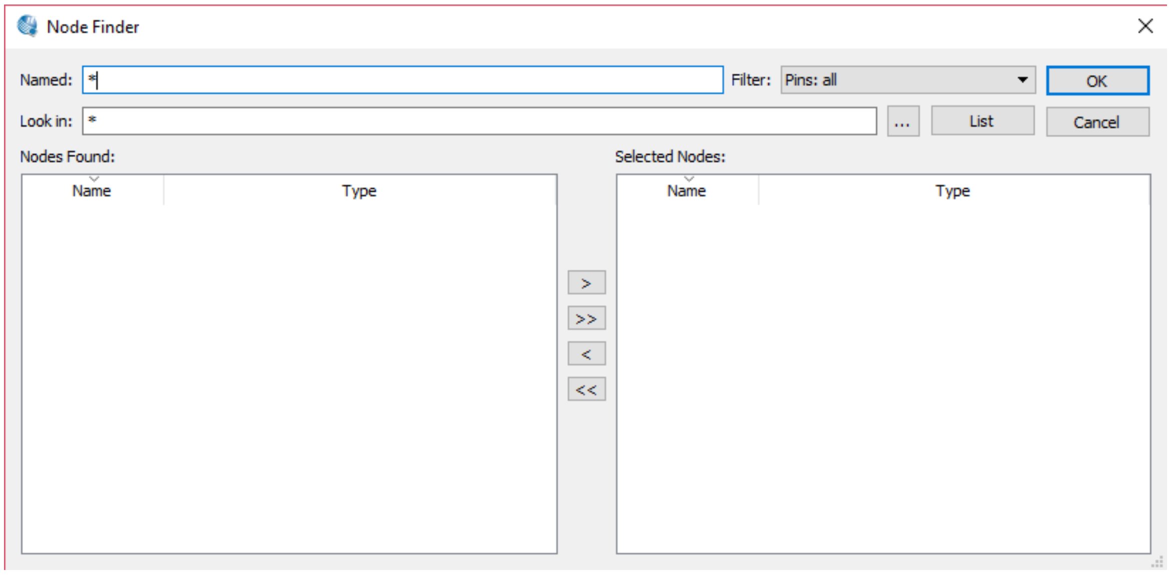
Bus width: 1

Start index: 0

☐ Display gray code count as binary count

Figure 4.11: Node or Bus Window

d) In the Insert Node or Bus dialog box, press the Node Finder button, a new Node Finder window will open (Figure 4.12).



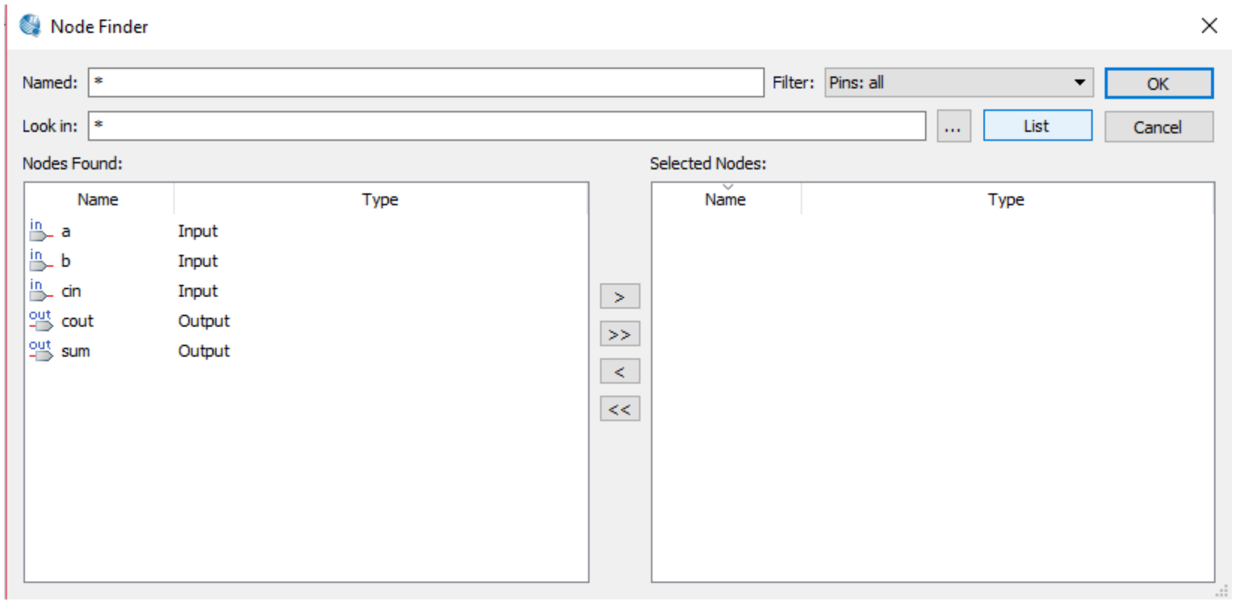
Named: *

Filter: Pins: all

Look in: *

Figure 4.12: Node Finder Window

e) In the Node Finder window, press the list button, this will display all inputs/outputs pins in your FA_Gate design (within the Nodes Found pane at the left side of this window) (Figure 4.13).



Named: *

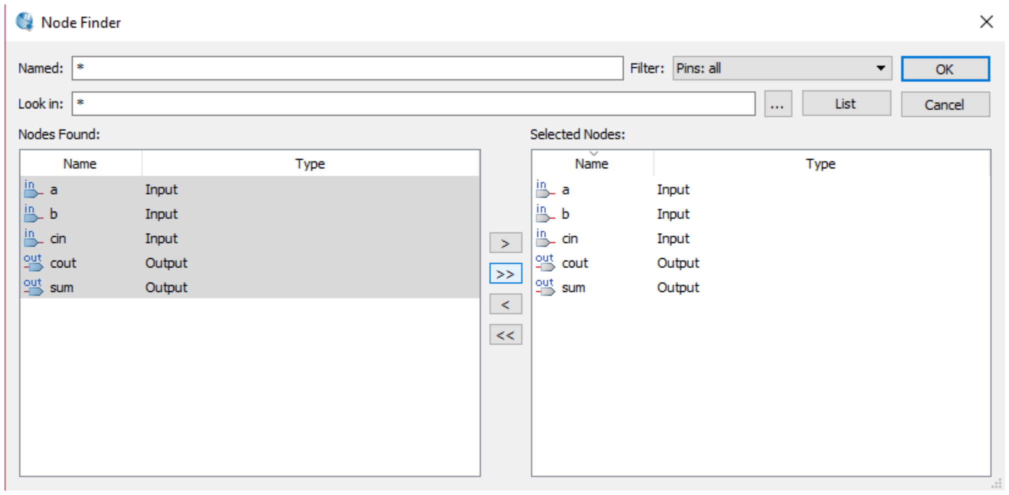
Filter: Pins: all

Look in: *

Name	Type
a	Input
b	Input
cin	Input
cout	Output
sum	Output

Figure 4.13: I/O Pin Selection in Node Finder Window

f) Select all the pins (i.e.: a, b, cin, sum, and cout), and add them to the right pane (Selected Nodes). Press OK twice (Figure 4.14).



Node Finder

Named: *   Filter: Pins: all

Look in: *

Nodes Found:

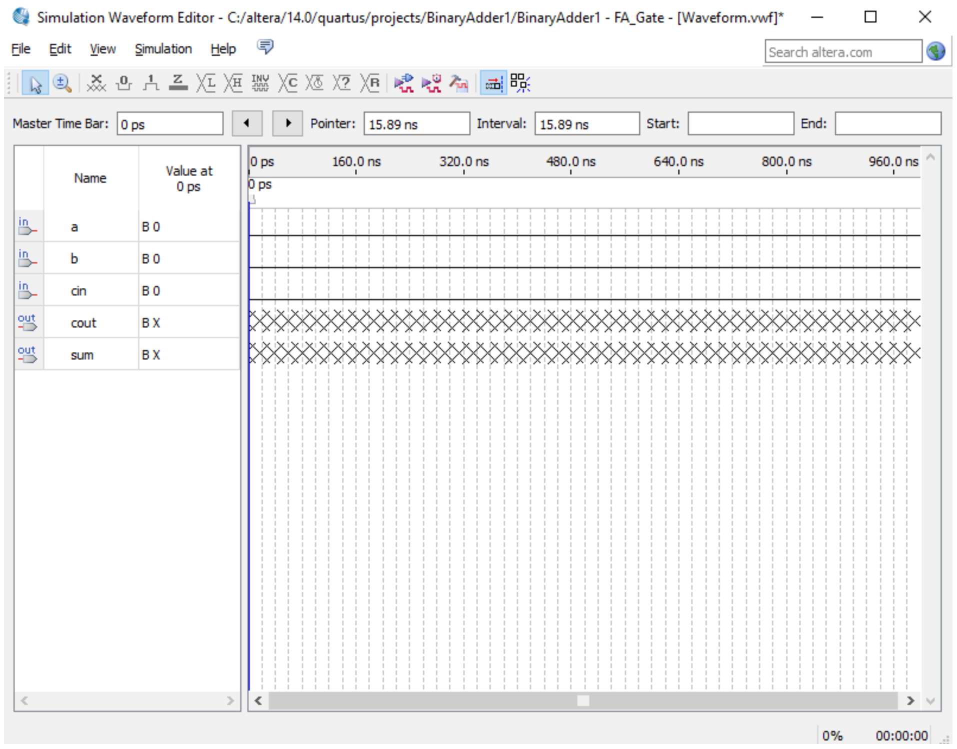
Name	Type
a	Input
b	Input
cin	Input
cout	Output
sum	Output

Selected Nodes:

Name	Type
a	Input
b	Input
cin	Input
cout	Output
sum	Output

Figure 4.14: Node Finder I/O Pin Configuration in Node Finder Window

This will add the a, b, cin, sum, cout pins to the left pane of the Simulation Editor (Figure 4.15).



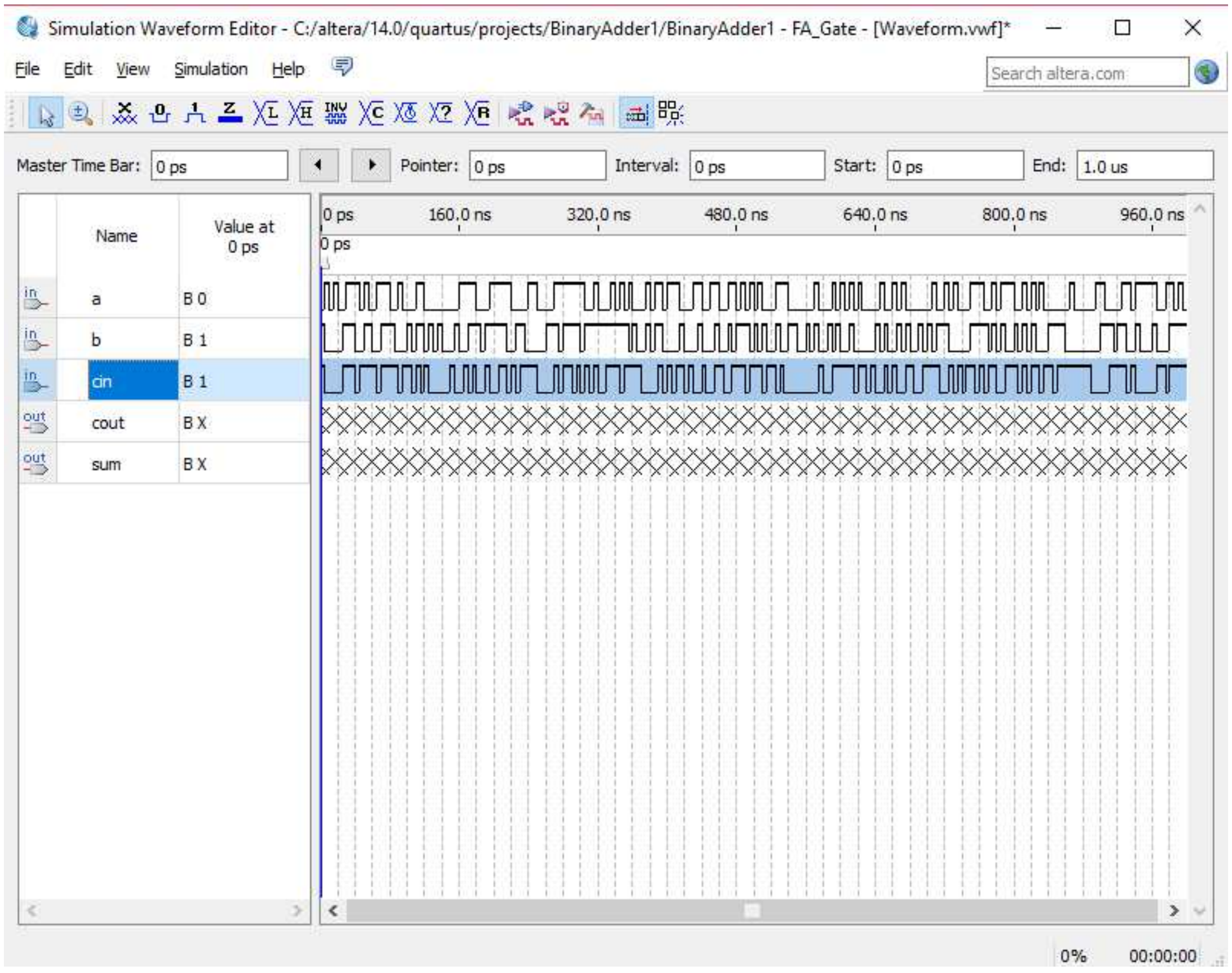
Simulation Waveform Editor - C:/altera/14.0/quartus/projects/BinaryAdder1/BinaryAdder1 - FA_Gate - [Waveform.vwf]*

Master Time Bar: 0 ps   Pointer: 15.89 ns   Interval: 15.89 ns   Start: End:

Name	Value at 0 ps
a	B 0
b	B 0
cin	B 0
cout	B X
sum	B X

Figure 4.15: Simulation Waveform Editor I/O Configuration

g) Highlight each one of the three input pins at a time (a, b, cin), and from the Edit menu, select Value → Random Value (choose the Every Grid option). This will assign a random wave form to each one of the three input pins (Figure 4.16).

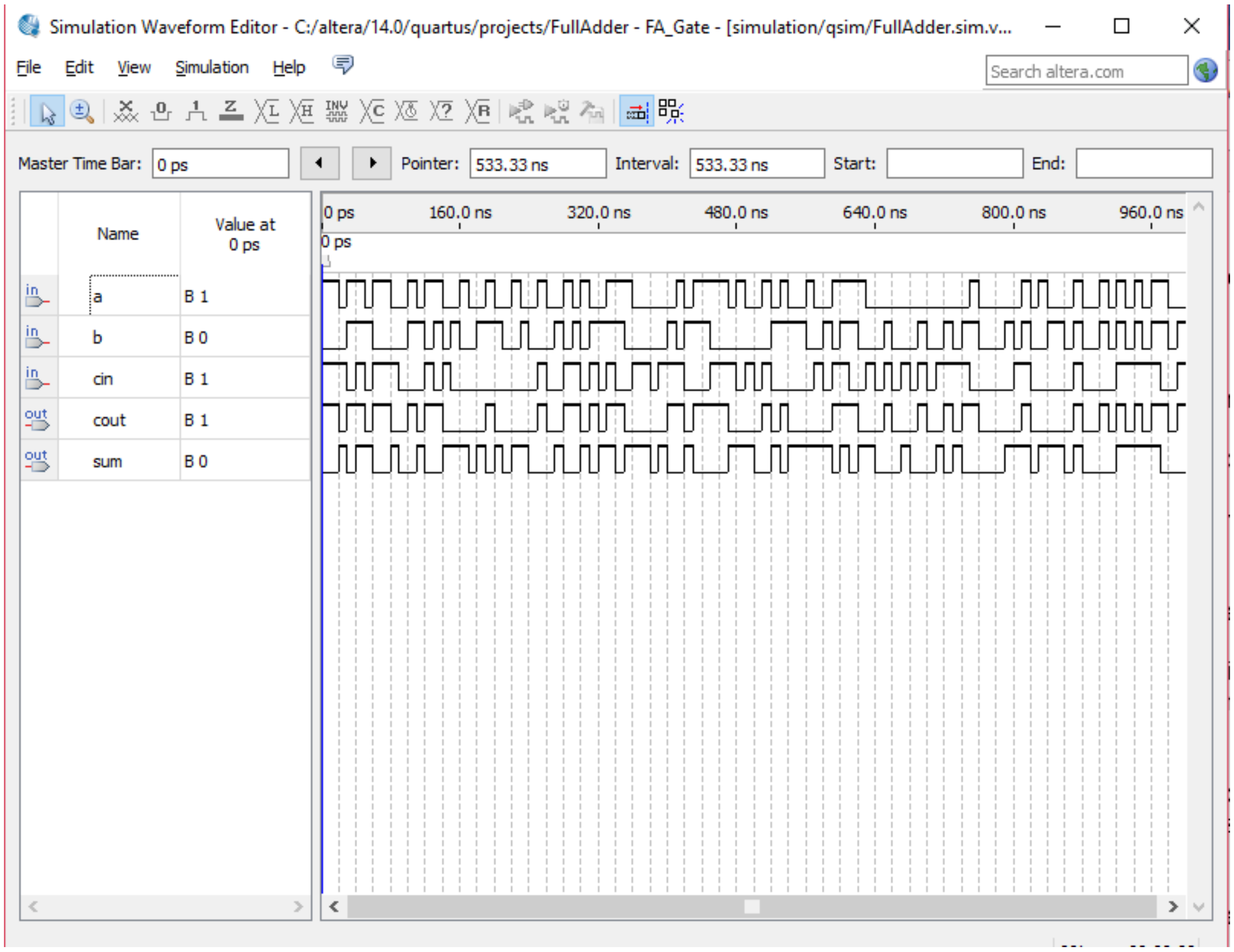


Simulation Waveform Editor - C:/altera/14.0/quartus/projects/BinaryAdder1/BinaryAdder1 - FA_Gate - [Waveform.vwf]*
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 0 ps
Interval: 0 ps
Start: 0 ps
End: 1.0 us
0 ps 160.0 ns 320.0 ns 480.0 ns 640.0 ns 800.0 ns 960.0 ns
0%
00:00:00

	Name	Value at 0 ps
in	a	B 0
in	b	B 1
in	cin	B 1
out	cout	B X
out	sum	B X

Figure 4.16: Random Input Waveform Generation

h) From the Simulation menu, select Run Functional Simulation (Figure 4.17). This will initiate the functional simulation of your design based on the selected random input waveform. You will be asked to save your file (extension is .vwf), go ahead and save it. A new Simulation Editor window will be displayed showing the sum and cout output waveform, as follows:


Simulation Waveform Editor - C:/altera/14.0/quartus/projects/FullAdder - FA_Gate - [simulation/qsim/FullAdder.sim.v...
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 533.33 ns
Interval: 533.33 ns
Start:
End:
0 ps 160.0 ns 320.0 ns 480.0 ns 640.0 ns 800.0 ns 960.0 ns

Name	Value at 0 ps
a	B 1
b	B 0
cin	B 1
cout	B 1
sum	B 0

Figure 4.17: Functional Waveform Simulation

a. Examine the output waveforms and make sure everything is correct. Notice that the outputs (sum and cout) change at the same time with the inputs. This is because in functional simulation no propagation delays are counted for the internal components of the circuit.

4.3.3 Timing Simulation

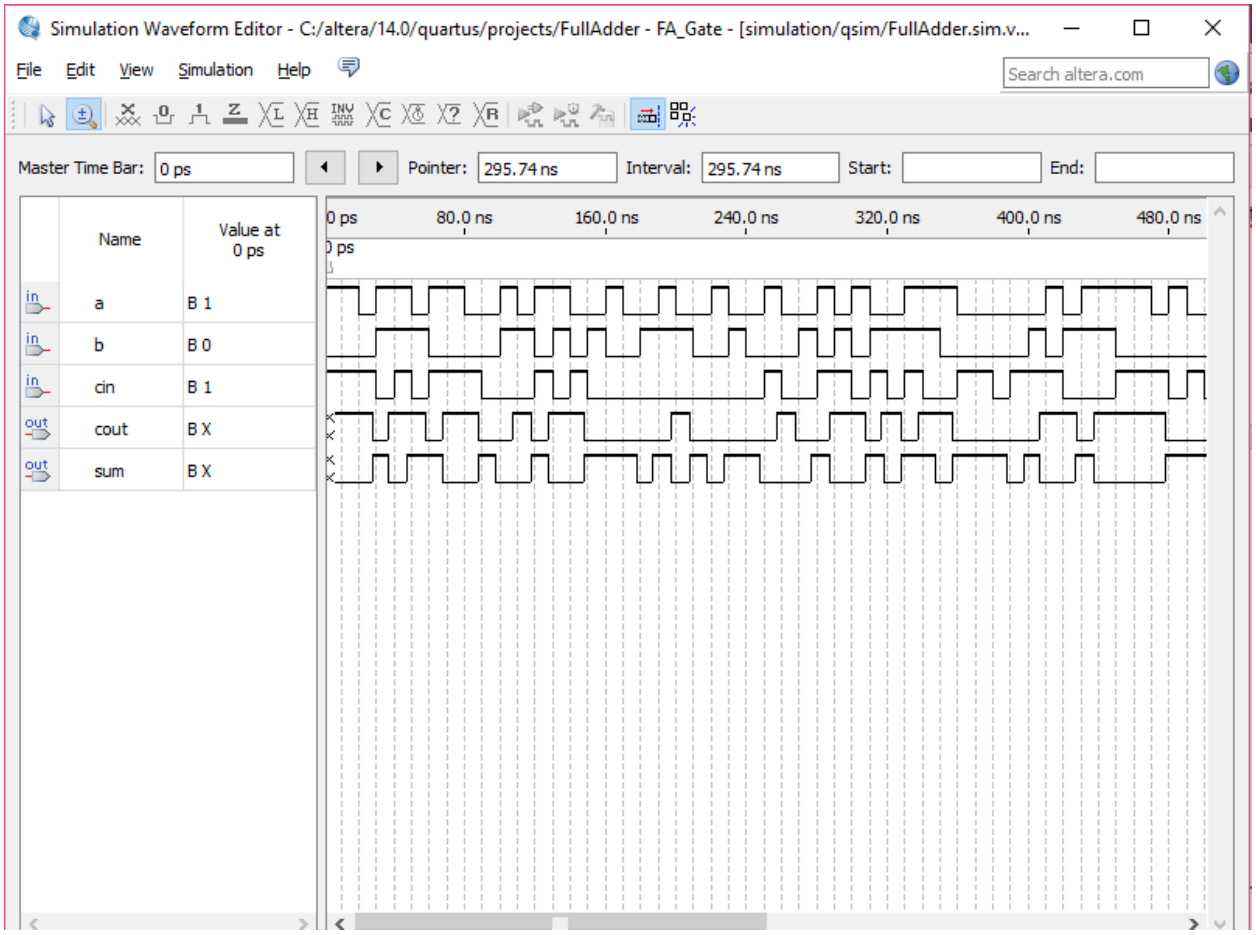
In behavioral simulation, timing information of the design under test is not included. In this section, timing simulation will be conducted on the full adder design. Timing simulation will enable the designer to watch for timing issues that might arise due to the inherent propagation delays of the used logic gates. To conduct timing simulation on the full adder circuit, proceed as follows:

a) Before timing simulation, the full adder design should be processed by the Fitter and TimeQuest Timing Analysis tools. To do so:

    a. Select the Processing → Start → Start Fitter from the Processing menu (Figure 4.18). After Fitter is complete, more detailed resource-consumption information will be generated and displayed on the screen. It is always important to check these generated reports against the expected resource utilization of the underlying design.

    b. After Fitter is complete, select the Processing → Start → Start TimeQuest Timing Analyzer from the Processing menu. By completing the TimeQuest Timing Analysis step, timing information, such as propagation delays, for the full adder design is now generated based on the specific target FPGA device (Cyclone IV E EP4CE115F29C7).

    c. Open the .vwf simulation file from step h of the behavioral simulation section. To do this, from the File menu, select Open and choose the proper .vwf file that was previously created during functional simulation. This will display the previously saved .vwf file in the Simulation Waveform Editor, with the same input waveforms assigned previously during functional simulation. For timing simulation, in the Simulation Waveform Editor, select Simulation → Run Timing Simulation. A read-only Simulation Waveform Editor window will open displaying the assigned input waveforms, in addition to the generated output waveforms. Notice the time delay between the change in inputs and the corresponding change in output.



Name	Value at 0 ps
a	B 1
b	B 0
cin	B 1
cout	B X
sum	B X

Figure 4.18: Timing Simulation

4.4 Implementing a 4-Bit Binary Adder/Subtractor

This section presents a design of a 4-bit binary adder/subtractor. The addition/subtraction module will be designed following a structural way through utilizing the 1-bit full adder (FA_Gate) module that was created in the previous section. Specifically, the subtraction operation will be conducted by adding the 2's complement of the subtrahend to the minuend. The 2's complement mechanism will be implemented by first obtaining the 1's complement of the subtrahend (bitwise complement), followed by adding a 1. This can be realized by using the first carry-in to the 4-bit adder together with four two-inputs XOR gates as shown in Figure 4.19.

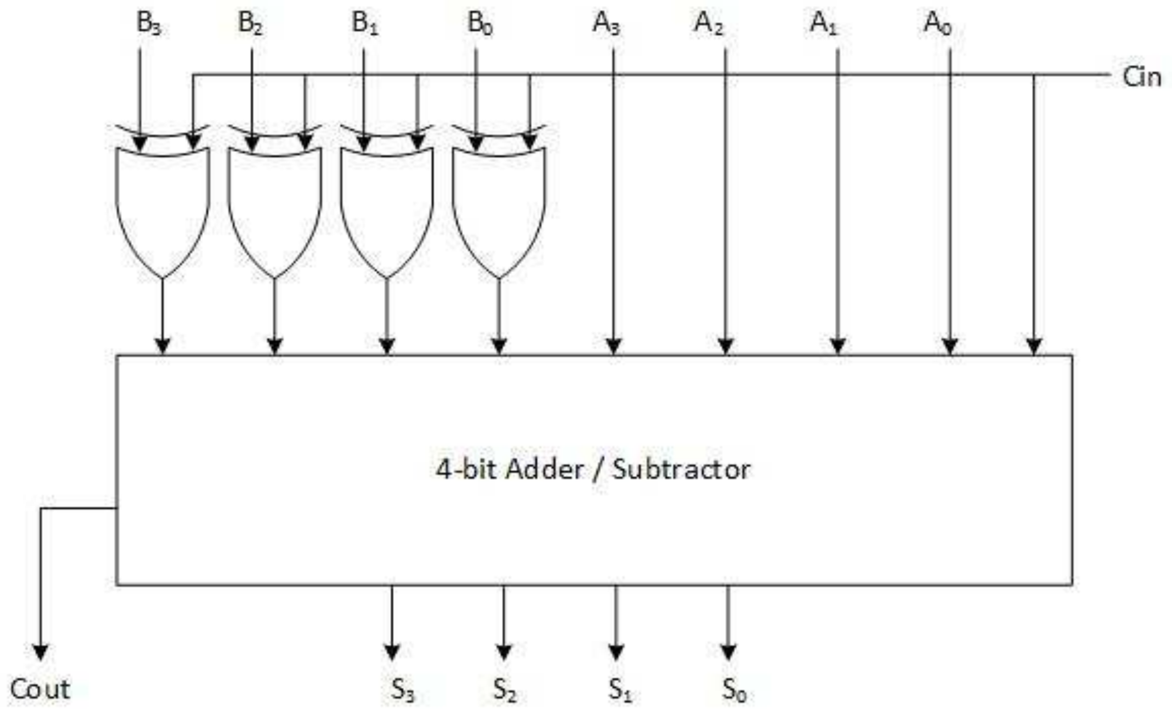


Figure 4.19: 4-bit binary adder / subtractor

To implement the 4-bit binary adder subtractor, follow the following steps:

a) Create a new project in Quartus II CAD Tool. Follow similar steps as in "Designing the 1-bit full adder" section. In this implementation, however, make sure to name the top-level entity as BinaryAddSub4. Also, you need to add the Verilog file of the FA_Gate entity to the project. To do this, in the Add Files dialog box window (during project creation phase), browse to the location of the required v file and select it. The file name will be displayed in the "Find name" textbox. Press the Add button to add this selected file to your project.

b) Add a new Verilog file to the project and name it BinaryAddSub4.v

c) Add the following code to the BinaryAddSub4.v file:

```
module BinaryAddSub4 (s, cout, a, b, cin);
        parameter N = 4;
        output [N-1:0] s;
        output cout;
        input [N-1:0] a, b;
        input cin;
        wire[N:0] c;
        genvar i;

        generate
                for (i=0; i<N; i=i+1)
                begin : bit_add_sub
                        FA_Gate FA (c[i+1], s[i], a[i], b[i] ^ cin, c[i]);
                end
        endgenerate

        assign c[0] = cin;
        assign cout = c[N];
endmodule
```

d) Using the same steps as in the “Implementing a Full Adder” section, run functional and timing simulations for the 4-bit adder/subtractor design in this section.

e) Examine the generated output. Do you notice the embedding of time delays? Notice that the output now change after some propagation delay time in response to the corresponding change in inputs.

f) Can you tell if the expected output, based on new inputs, can be read without any issues? In other words, is there enough time to read the output before any new change in inputs? If not, how can you address this issue?

Extra Credit
Design and simulate (functional and timing) the following Boolean function using a 4-to-10 BCD decoder:

$$f(a,b,c,d) = \sum (m_1, m_2, m_3, m_5, m_7)$$

What does this circuit imply?

5. REVIEW QUESTIONS

1. Explain the difference between a half adder and full adder.

2. Explain why a full adder requires a carry input, opposed to just a carry output.

3. Explain why 2’s complement addition is required opposed to 1’s complement addition to implement subtraction in digital circuits.

4. Explain generally how adder circuits can also be used to implement multiplication and division.