

EE261 Computer Project 2: Binary Adder

Introduction

In this project you will use **Mentor Graphics** to examine the characteristics of the ripple carry adder. You saw in the last project that logic circuits do not react instantaneously. When the input to a gate changes, there is invariably some delay before the output changes. This project will demonstrate the consequences of these delays.

Simulating a One Bit Adder

Log into an HP and start **Mentor Graphics**. Using the goto button (four arrows), navigate to:

```
/usr/local/classwork/ee261/mentor/project2
```

Select the “261Circuit2” icon and, using the method from project 1, copy it to your `mgc` directory. Navigate back to your `mgc` directory and start **QuickSimII** on “261Circuit2”.

When **QuickSimII** comes up, open a schematic view of the circuit. Select the adder, then set the timing mode to *Full Delays* max. Select the two inputs (A and B) and the two outputs (Cout and Sum). Note that the inputs are on the right and the outputs are on the left, the reverse of the usual placement. Now open Trace and List windows containing these signals. Using the waveform editing tools you used in project 1, draw stimulus for inputs A and B so that all possible logical combinations of A and B occur. When you are done, type `run` and press return to start the simulation.

In **QuickSimII**, you can run a simulation for a specific amount of time. If you wanted to run it for 100 ns, you would just type `run 100`. Just replace the 100 with however long you want to simulate. Now, use the view menu to zoom in around the changes on the trace or examine the List window.

Questions:

1. In the worst case, how long does it take, after the last input change, before the Sum output correctly indicates the sum of the inputs A and B?
2. In the worst case, how long does it take the carry output, Cout, to reflect changes in the input?

Hint: use the `DELTA`? tool from the palette. (This tool is found on several palettes. To change between palettes, use the brown palette selector buttons.) To use the `DELTA`? tool, click on its icon and then click near a waveform change. When prompted, click near a second change

later in time. Information about the changes you clicked on and the delta time (time difference) between them is reported in the message area. When you are done gathering data and any print-outs you may need, close **QuickSimII**.

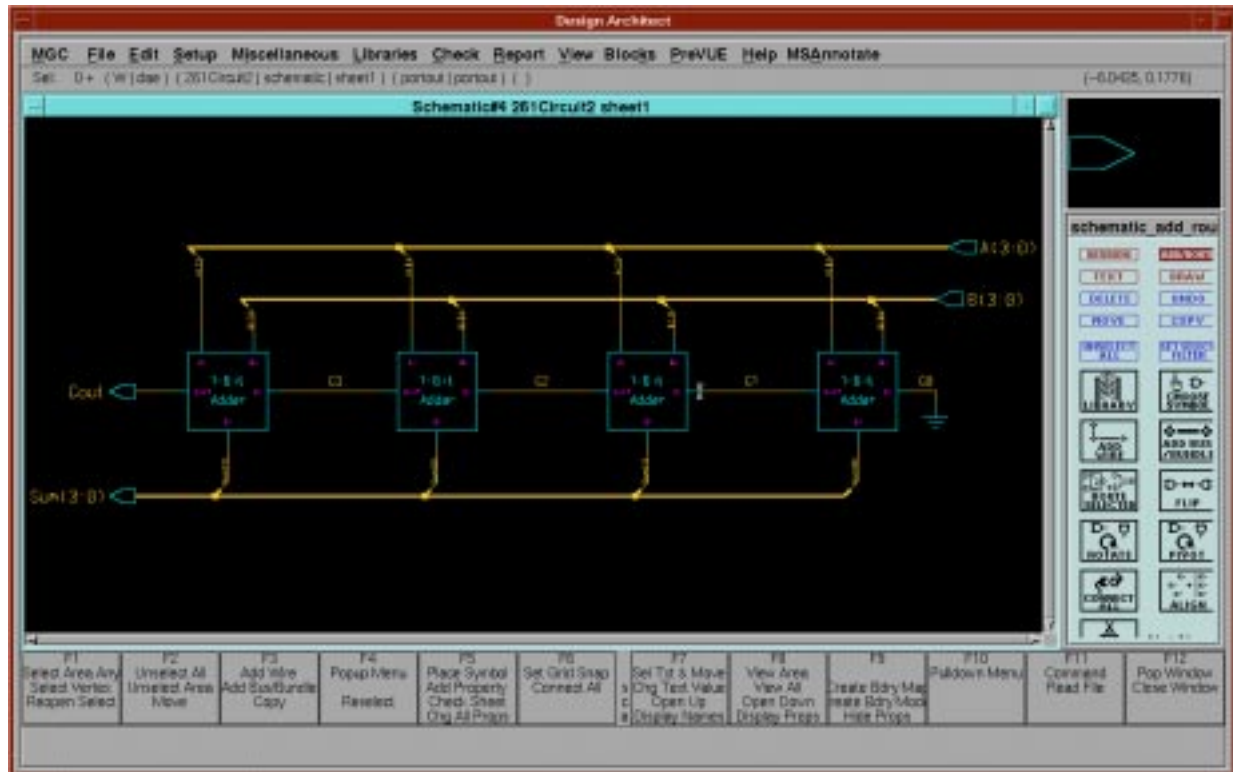


FIGURE 1. Four-Bit Adder for Project 2

Building a Four-Bit Adder

In this section, you will use copies of the one-bit adder you were given in the last part to build a four-bit adder. A suggestion for the layout of the circuit is shown in Figure 1. Remember, you have to edit the circuit from **Design Architect**. From **Design Manager**, select the `261Circuit2` icon, then start **Design Architect** on it from the pop-up menu. Once **Design Architect** has loaded, follow the steps below to modify your circuit.

1. Select all the wires connected to the one-bit adder. Delete these wires. After this step, you should have the one-bit adder, a “ground,” four ports (A, B, Cout and Sum), and no wires left in your circuit.
2. The component called `1-Bit Adder` is not in your libraries. Instead of picking it out of a library, you will need to copy the one that’s already in the circuit. Do this by selecting the one-bit adder component. (Click anywhere inside the box so it becomes highlighted.) Then choose `Copy⇒Multiple:` from the pop-up menu or the menu bar. Enter 3 in the prompt box and press return. Only one ghost image of the part will appear. Click to place it to the left of the original adder and the Cout port. The other two will be placed automatically. You may need to zoom out to see all four adders. You could also use `View⇒All` from the menu bar.

3. Unselect everything, then select just the Cout and Sum output ports. Use the `Move` command from the pop-up menu, the menu bar or the palette to move them to the far left of your circuit. See Figure 1 for placement of the output ports.
4. Next, you will begin to wire up your circuit. Look at Figure 1. Notice how the wires coming from inputs A and B and the wire going to output Sum are thicker? These lines are busses (bundles of wires). In this case, there are four wires in each bus. To add these you will need to use the `ADD BUS/BUNDLE` tool from the `ADD/ROUTE` palette. This tool works just like the `ADD WIRE` tool. Draw a straight line for each bus. Cancel the tool when you are done drawing.
5. Mentor now knows the wires you just drew are busses, but it doesn't know how many wires they contain. You have to give each bus a specially formatted name to provide this information. Make sure everything is unselected, then select all of these port names now by click on each one while holding down the `shift` key. This allows you to select properties. The things you should have selected are A, B, and Sum. Go to the `TEXT` palette and click `CHANGE VALUE`. A prompt box will appear in the lower left with the current value of the property you are changing highlighted. Simply enter the new value in the `New Value` box and press return. You will automatically be prompted about the next property value to change until they are all changed. You must change A, B and Sum to `A(3:0)`, `B(3:0)` and `Sum(3:0)` respectively. This tells Mentor that the wire is a bus with bit 3 most significant and bit 0 least significant (a four-bit bus).
6. Now that you have changed the names so Mentor knows the size of the busses, you can add the other wires to your circuit. In this step, you will draw wires from the adders to the busses. When you draw a wire that attaches to a bus, a bus ripper is automatically installed. (A bus ripper is just a device to pull one wire or bit out of the bundle). When the ripper is placed, you will be prompted to enter what bit of the bus the ripper is to pull out. Enter the bit number and `OK` the form. The label is automatically placed. Referring to Figure 1, you will see that the far left adder corresponds to bit 3 of the bus and the far right adder is bit 0. Draw the bus connections for each adder and enter the proper bit numbers, following Figure 1's numbering pattern. When you are done, draw the wires between the adders and the connection to the Cout port. Note that when you're drawing the wires that attach to the busses, you must start at the other end and finish on the bus. Otherwise, a bus ripper will not be installed.
7. You will want to give a name to the internal carry wires, so that you can watch their values during simulation. To do this, unselect all (`F2`). Join the carry ins and the carry outs of the adders using normal wires. Unselect all again. Select the first wire, the one that connects the carry out of bit 0 and carry in of bit 1. Goto the Text Pallette. Choose `NAME NET` and enter a name (`C1`, `C2`, etc.) for the wire in the 'Property Value' box and do NOT press enter yet. Click the '+' button of the 'At Location' and move the pointer near the wire where you want the label to be. You'll see a line joining the concerned wire and the mouse pointer. Place the label by clicking the left mouse button. Repeat the same for the other carry wires.
8. When you are done wiring your circuit, select `Check⇒Sheet` from the menu bar. The report may have a warning about the ground connection, and possibly warnings about dangles on the busses. These are OK. If you have any other warnings or errors, check your work, correct the problem and check the sheet again. Remember, you can use cross-selection to help find your errors. Highlight the handle name (i.e. `N$7` or `I$16`) in the check sheet report and the corresponding part will be highlighted in your circuit.

- When your sheet checks with no errors or warnings, save it. From the menu bar choose `File⇒Save Sheet As...` You'll want to enter a new name so you don't overwrite the original circuit. Enter a new name in the `Component Name:` box and OK the form.

Make a printout of your completed circuit before you go on. Remember, you *must* setup your printer each time you print in any tool during a Mentor session. Close **Design Architect** when you are done editing your circuit and have a printout. You are now ready to simulate your circuit.

Simulating a Four-Bit Adder

Use the four-bit adder to compute $14+1=15$ (in hexadecimal, this is expressed as $E+1=F$). In **Design Manager**, update the Navigator window and start **QuickSimII** on your new circuit. Once **QuickSimII** has loaded, open a schematic view of your circuit. Also open Trace and List windows with the inputs, the outputs, C1, C2 and C3 in them. Select all four adders and change the timing mode to *Full Delays max*. Create some input for the circuit by using the `ADD FORCE` tool from the `STIMULUS` palette. In the prompt box that appears, enter a value and when you want the signal to take on that value. The default value type is hex so you can enter a number (0-9) or a letter (A-F) directly. OK the form when you are done. Do the same thing for the other input signal. Begin with both inputs set to 0, then make input A(3:0) go to 14 (decimal) or E (hexadecimal), then, at a later time, make input B(3:0) go to 1 (decimal or hex). Having A take on its value at 400 ns and B take on its value at 600 ns works well. When you are done editing the stimulus, run the simulation by typing `run`.

Note: Use hex inputs for all simulations in the project. If you are asked to input a decimal or binary number, do the conversion manually. If you are asked for results in binary or decimal, manually convert the hex output Mentor gives you.

Questions:

- What is the result of this addition, in binary?
- How long did it take, from the time that input B changed, until the outputs were stable and giving the correct final result?

Make a printout of the timing diagram for this part. Label on the printout the point at which the last input changed and the point at which the output stabilized. Indicate the elapsed time between the two events.

Further Simulation

Reset the simulation state by using the `Reset` button on the palette. Now use the four-bit adder to compute $15+1$. Remember, you will need to select inputs A and B and `Delete`

Forces so you will have a clean slate for the new inputs. Begin by setting both inputs to 0, then set input A to 15 (decimal), then, at a later time, set input B to 1. Run the simulation and examine the output trace. You will probably have to zoom in around the areas of changes.

Questions:

5. What is the result of this addition, in hex and binary?
6. What incorrect results, if any, are shown on the hex output display before the correct result appears?
7. How long did it take, from the time that input B changed, until the outputs were stable and giving the correct final result? Did this addition take more or less time than $14+1$?
8. Why do you think this is called a *ripple carry* adder?

Make a printout of the timing diagram for this part. Label it as you did the previous diagram.

Final Simulation

Use the four-bit adder to compute the sum of the binary quantities 1111 and 0101. Begin by setting both inputs to 0, then set input A to 1111 (binary), then, later, set input B to 0101 (binary).

Questions:

9. What is the result of this addition, in hex and binary?
10. How long did this addition take? Did this addition take more or less time than $14+1$ and $15+1$? Why?
11. For an N-bit ripple carry adder, what will be the *maximum* (worst-case) delay from input to output? (Hint: consider the sum of $11\dots111$ and $00\dots001$.)

Make a printout of the timing diagram for this part. Label it as you did the previous two.

Report

The report (one per person or group) should be typed and follow the report format handed out in class. Be sure to include:

- Answers to the questions (1-11)
- A printout of the 4-bit adder circuit
- Trace printouts showing the results of $14+1$, $15+1$, and $1111+0101$