Digital ICs Lab 3

Manchester Carry-Chain Adder

Contents

List of Figures

1 Preparatory Exercises

Start by reading through this compendium. Use the information found here and the course book (Pages: 568-573) to fill in the answers to the preparatory exercises. In order to be well prepared for this exercise we suggest that you read through the corresponding chapter in the main textbook, i.e. chapter 11, especially the parts discussing adders. The preparatory exercises MUST be completed before you arrive to the lab exercise. If you fail to do so you may not be allowed to participate.

- Consider a Manchester carry-chain adder in "dynamic" implementation. Having the inputs $A < 0 : 7 >$, $B < 0 : 7 >$, and C_{in}, write down the equations for Generate (G_i) and Propagate (P_i) as functions of inputs i.e. A_i, B_i and C_{in}.

  Further on, write down the equations for Sum (S_i), and Carry (C_i) as functions of intermediate signals (i.e. G_i, P_i, C_{i-1}, etc.).

- Draw the schematic of the Generate (G), Propagate (P), Sum (S_i), and Carry (C_i) functions in gate level (logic AND, OR, XOR etc.)

  Also draw the 8-bit Manchester "carry-chain" circuit in transistor level. Assume that both inverted and non-inverted inputs exist!

  Try to analyze and understand how the the circuit works in precharge and evaluation phases of the clock.

- Where is the critical path (worst-case delay) in the 8-bit Manchester carry-chain adder? And what input code can be applied to simulate and measure the critical path delay?

- Think through how to size the transistors in the carry-chain to reduce the delay. Which transistors should be sized (and how) and which transistors are preferably minimum size?

Show your preparatory exercises to the Lab assistant!

Date & LiU-ID of Student	Signature of Lab Assistant

2 Theory Regarding the Sizing of the Manchester Carry-Chain

The worst-case delay of a four-bit dynamic carry chain (figure 11-8 in the course book) can be modeled by the linear RC-network shown in Fig. 1 below.

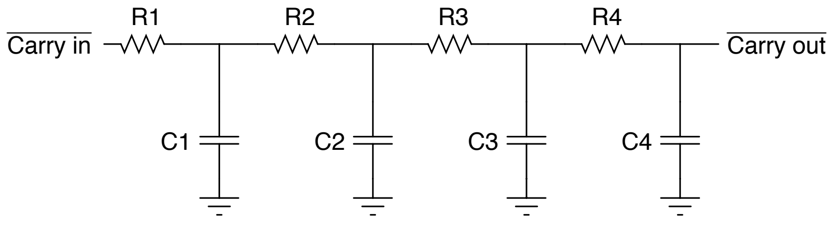

Figure 1: Linear RC-network modeling the worst-case delay of a dynamic carry chain.

The resistance R_i at each node of the chain comes from the corresponding pass transistor, and the capacitance C_i is the linearized joint capacitance from all the devices connected to node i. The delay of the RC–network can be determined by the expression (Elmore delay formula)

$$t_p = 0.69 \cdot (R_1 \cdot C_1 + C_2 \cdot (R_1 + R_2) + C_3 \cdot (R_1 + R_2 + R_3) + C_4 \cdot (R_1 + R_2 + R_3 + R_4)).$$

Since R_1 occurs 4 times in the expression, it makes sense to minimize this contribution by making the first transistor larger than the other ones, or by using progressive scaling. First consider the case where all transistors are minimum-size (all R and C are equal). We have

$$t_p = 0.69 \cdot C\ (4R + 3R + 2R + R) = 0.69 \cdot 10 \cdot RC.$$

graphics Now assume that the stages are made progressively larger, starting from a minimum-size transistor at the output. The (W/L) of the next-to-last transistor is scaled by a factor k ($k > 1$), which means that its resistance is divided by k, while its associated capacitances are approximately increased by a factor k. The following expression holds:

$$C_i = k \cdot C_{i+1},$$

$$R_i = R_{i+1}/k,$$

where $C_4 = C$ and $R_4 = R$.
This yields the following expression for t_p

$$t_p = 0.69 \cdot R \cdot C(1 + 2k + 3k^2 + 4k^3)/k^3.$$

If we for example use $k = 2$, we get the total propagation delay:

$$t_p = 0.69 \cdot 6.125 \cdot RC$$

This new propagation delay is almost 40% lower than for the unsized Manchester carry-chain according to our two expressions above. On the other hand, note that the area increases dramatically with k, which effectively excludes the use of large scaling factors. Thus, for k larger than approximately 2, progressive scaling is not very efficient. Large values of k also affects the driving capability of the Generate and Propagate signals since they will be heavily loaded by the large transistors in the carry-chain. This might result in a need for up-sizing these gates as well, which increases the area even more.

3 Design and Simulation of an 8-bit Manchester Carry-Chain adder

3.1 Purpose of this exercise

During this laboratory exercise you will complete the design of an 8-bit Manchester carry-chain adder. You will then simulate your design and try to improve the performance. You will also look into how the delay depends on supply voltage and temperature.

3.2 The IC design tool

The software to be used is called Cadence, which is a tool commonly used in industry for development of full custom integrated circuits. All designs in Cadence are organized in libraries. Reference libraries include basic building blocks such as transistors, resistors, capacitors etc. Design libraries include your designs in which you have instantiated components from the reference libraries. The Library Manager, which pops up when you start Cadence, shows a tree structure with the names of all the libraries available. You can add and/or remove libraries from the Library Manager. Each library includes cells and their different views. A cell is for example an inverter and the view is either a schematic, a symbol, or a layout representation of the cell. The simulation tool is called Analog Environment and is invoked from the menus in the schematic tool. All the supply sources and signal sources are inserted directly into the schematic. In this way we save a lot of work compared to writing all the netlists manually. Analog Environment uses the Cadence Spectre-simulator, which is a simulator similar to SPICE. The simulation results are presented as waveforms in a new window. The signals can be post-processed and plotted/printed by the calculator tool that is available.

3.3 Initialization of the lab

If you have done the first lab you have already set up the tool and have copied proper directories with the script. Then what you need is just to start the Cadence as you did in lab 1 and just go to the library "manchesterAdder8bit".

Just to recall: log in to the Cadence server with

```
ssh -X ssh.edu.liu.se

ssh -X naum.ad.liu.se
```

and start the tool in your home directory with the commands

```
tcsh

source /sw/defaults/dotfiles/.tcshrc

module add kurs/eks

module add DIGIC

source .DIGIC_rc

cad
```

3.4 Getting started

In this section you will learn how to use the schematic editor in Cadence and handle its most elementary functions. In the library "*manchesterAdder8bit*" some cells are prepared for you. Go to the window called Library Manager. Click at the library called "*manchesterAdder8bit*" with the left mouse button (lmb), now you should be able to see what cells there are in this library. To make your work easier we have prepared some cells for you, **however some of them are not completed and to finish them will be your first task.** Open the schematic view of the cell called testbench by the following procedure:

In the Library Manager, choose the testbench cell by clicking at it with the lmb. Then press rmb at the *schematic* view and choose "open" from the pop up menu, or choose "File ⇒ open" from the File menu in the Library Manager.

Now you have opened the design that you will work with. Look at the different instances and try to identify the different components. The 8bit-adder you see in the testbench consists of several sub-blocks (instances). To be able to view these instances and to be able to edit them, we have to descend into them. To descend into an instance, do as follows: Highlight the symbol of the 8bit-adder by clicking at it, you should get a white rectangle around it. Then,

Edit ⇒ Hierarchy ⇒ Descend Edit... [E], (E is the short command: Shift-e)

and press OK. Now you see the schematic view of the 8bit-adder that consists of eight adder-cells and one carry-chain. To descend into these cells you repeat what you just did. To return to a higher level in the hierarchy (for example back to the testbench) do as follows:

Edit ⇒ Hierarchy ⇒ Return [B]

repeat until you are back at the testbench level. Explore the design hierarchy further until you know how the adder is built up.

3.5 Completing the 8-bit adder

Hopefully, you have seen that there are some parts missing in the adder design. In the adder cell we have left out the Generate function (G). Descend into the adder cell or open it from the library manager. Look at how the other functions in that cell are implemented. Now implement the Generate function (G) according to the preparatory exercise.

To add a transistor do the following:

Create ⇒ Instance [i]

fill in the form or use the browser. Browse in library manager to:

- *Library* ⇒gpdk45
- *Cell* ⇒ *pmos_1v_lvt* (or nmos)
- *View* ⇒ *symbol*

Give the transistor the minimum allowed dimensions (default) and add it to the schematic,

press cancel [Esc].

Another way of adding a transistor is that you can copy one of the other transistors in the schematic by:

Highlight the item you want to copy by clicking on it

Press [c]

Click on the item you want to copy and place the new item where you want it.

You can of course copy other components and wires in the same way.

To add a wire you do as follows:

Create ⇒ Wire (narrow) [*w*].

Click with the lmb at the starting point.

Drag the marker and click with the lmb at the ending point.

Press [Esc] to leave the add wire mode.

To add a name to a wire you do as follows:

Create ⇒ wire name [*l*]

Write the wire name in the new window that pops up.

Click with the lmb at the wire which you want to name.

Press [Esc] to leave the add wire name mode.

Now you can complete the design of the "*adderCell*". When you are finished you save your work by:

File ⇒ Check and Save [X]

The next step is to implement the Manchester carry-chain that you have drawn when you did the preparatory exercises. Descend into the carryChain symbol or open it from the library manager. All that exists in this schematic view is the pins. Do not rename the pins since that will force you to build a new symbol that contains the new pin names. You do not have to draw all inverters that you need yourself, you can instantiate them from your design library: Create ⇒ Instance [i]. Fill in the form or use the browser, Browse button.

- *Library ⇒ manchesterAdder8bit*
- *Cell ⇒ inv*
- *View ⇒ symbol*

Press cancel [Esc]. Complete the design and save it, make sure no warnings and/or errors appears when saving. Close the schematic.

3.6 Simulating your design

In this section you will learn how to use the Cadence simulator environment called Analog Design Environment (ADE). Open the schematic view of the "testbench" cell. To invoke the simulator, issue the following command from the testbench schematic window:

Launch ⇒ ADE L

Under the section Analysis in the Virtuoso Analog Design Environment, you specify what type/types of analysis that should be made. In this lab we will only use the transient analysis. The arguments in the transient analysis form are: start time, stop time and accuracy. The conservative option gives the highest accuracy in your simulation. You can change the start/stop time and accuracy by clicking the button with the text "AC,TRAN,DC" on it. Fill in the stop time 5n (means 5 ns!) and click OK.

You should also load the design variables from testbench. For that, select "Variables ⇒ Copy from cellview". Set the "Tp" (clock period) to 500p second and use power supply of 1.2 volts for "vccr12". Under the section Outputs you specify the voltages, currents or expressions that you would like to plot or calculate. To add outputs to plot, do as follows:

Outputs ⇒ To Be Plotted ⇒ Select On Schematic

Now click at the nodes that you would like to plot. If you click at a wire, the voltage is plotted. If you are interested in plotting the current, click at the terminal of the instance (the red contact box). Select the nodes you would like to plot, press ESC to exit this mode.

Now we are ready to run a simulation and this is done by clicking the green play button or by

Simulation ⇒ Netlist and Run

– red stop button interrupts the ongoing simulation.

You have now run a simulation on an 8-bit Manchester carry-chain adder! Edit the input pattern, A_i and B_i, to check the functionality of your adder. To edit a signal source do as follows:

Highlight the symbol of the signal source by clicking at it, you should get a white rectangle around it.

Edit ⇒ Properties ⇒ objects [q]

Do not forget to save your design before you run a new simulation!

To measure the delay in the Waveform Window, use the markers available in the menu. Markers are available through: Marker ⇒ Create Marker... in the waveform window.

Run a simulation with input pattern signals (according to your preparatory exercise) that enables you to measure the critical path.

- What is the critical path delay?

$t_{critical-path} =$ ____________

- What is the delay through the carry-chain?

$t_{carry-chain} =$ ____________

3.7 Optimizing your design

Try to optimize your design and reduce the critical path by sizing the transistors in the carry-chain according to your preparatory exercise. Note that you need to should consider the whole stage in sizing (and not only the propagate transistor). You can use a variable as "k" in the sizing and sweep it in the analog design environment. Note that "kˆn" is expressed as "pow(k, n)" in the transistor property dialogs.

- What is the new worst-case delay and carry-chain delay?

$t_{critical-path} =$ ____________

$t_{carry-chain} =$ ____________

As you may have already noticed there exist a maximum size that a transistor can have with only one finger. To increase the size further you should probably use multiple fingers.

3.8 Delay vs. supply voltage and temperature variations

Vary the supply voltage (change it in the design variables) and note what happens to the critical path delay. Simulate with Vdd=1.1 V and Vdd=1.3 V.

Vdd (V)	$t_{critical-path}$
1.1	
1.2	
1.3	

Reset Vdd to 1.2 V. The on-chip temperature in real world can vary over a wide range and we will now investigate how this might affect the performance. Simulate at -40^o C , 45^o C, 70^o C, and 125^o C. To change the simulation temperature, go to the ADE window: *Setup* ⇒ *Temperature*. Fill in the new temperature and press OK.

Temperature	$t_{critical-path}$
-40	
45	
70	
125	

3.9 Worst case

Simulate the delay assuming we have the worst possible operating condition according to what you discovered above.

- What is the critical path delay now? Do you have any comments on the result?

- We have taken voltage and temperature variation into account to get the performance in worse case. Can you name some other variation that can lower the performance? What would be the solution to ensure that the fabricated design fulfills the specifications under these conditions?

Show your results to the lab assistant!

Date & LiU-ID of Student

Signature of Lab Assistant