Computer Systems and -architecture

Project 2: Adders

1 Ba INF 2018-2019

Brent van Bladel
brent.vanbladel@uantwerpen.be

Don't hesitate to contact the teaching assistant of this course. You can reach him in room M.G.305 or by e-mail.

Time Schedule

Projects are solved in pairs of two students. Projects build on each other, to converge into a unified whole at the end of the semester. During the semester, you will be evaluated three times. At these evaluation moments, you will present your solution of the past projects by giving a demo and answering some questions. You will immediately receive feedback, which you can use to improve your solution for the following evaluations.

For every project, you submit a small report of the project you made by filling in `verslag.html` completely. A report typically consists of 500 words and a number of drawings/screenshots. Put all your files in one tgz archive, as explained on the course's website, and submit your report to the exercises on Blackboard.

- Report deadline: **November 4, 2018, 23u55**
- Evaluation and feedback: **November 9, 2018**

Project

Read sections B.3, B.5 and B.6 of Appendix B. You can only use the following Logisim libraries for this assignment: Base, Wiring, Gates, Input/Output.

1. Build a decoder for a 3-bit input.
    - It has one 3-bit wide input and eight 1-bit wide outputs.
    - Only use the Wiring library (inputs, outputs, wires and splitters) and AND-, OR-, and NOT-ports.
2. Build a multiplexer for eight 8-bit inputs.
    (a) Create a new circuit in the same logisim file where you created your decoder. You can do this by choosing from the menu 'Project' - 'Add Circuit'. You can then choose which circuit you want to edit by double clicking it in the library menu on the left side. You can use a self made circuit as a building block in another circuit, in the same way as you use other blocks/gates in your circuit. The interface of your block is determined by the input and output ports you created in its circuit.

- Only use the Wiring library and AND-, OR-, and NOT-ports.
- Use your decoder of the previous exercise.

(b) What will be the size of your "select" input?

(c) In order to assess the performance of your multiplexer, find its *latency* by calculating the maximal path. This is the total number of gates that are maximally passed (do not count NOT-gates, their latency time can be ignored).

3. Build a 1-bit full adder (with carry in and carry out).

   (a) Determine the inputs and outputs of a 1-bit full adder and build a truth table.

   (b) Convert the truth table to Boolean algebra, and optimize the Boolean expression.

   (c) Implement the Boolean expression as a circuit called "1-Bit Adder" in Logisim.

4. Build a circuit of an 8-bit two's complement ripple carry adder.

   (a) Use 1-bit adders to create an 8-bit adder, that adds two 8-bit wide inputs.

   (b) Think about a way how overflow can be determined from carry outs. Overflow happens for example in these cases: 127 + 1 = -128 or -128 + (-1) = 127. Add an output bit denoting overflow to the circuit containing your 8-bit two's complement adder.

5. Build a circuit of an 8-bit two's complement carry lookahead adder using 2 4-bit adder blocks.

   (a) What are the "super propagates" and the "super generates", C1 and C2 values for the addition of numbers 1011 0011 and 1100 0100? Calculate the carry out of all 4-bit adder blocks, including the most significant bit (i.e. c_7).

   (b) Build a circuit for a 4-bit adder block. This block has input carryIn, a_0, a_1, a_2, a_3, b_0, b_1, b_2, b_3 and outputs s_0, s_1, s_2, s_3, P_0, G_0. Note that there is no output for carryOut, as a carry lookahead adder doesn't use c_{i-1}.

   (c) Optimise this 4-bit adder block to decrease the circuit latency (maximum number of ports traversed). Do this by expressing your block as a set of Boolean equations (for every output pin, in terms of input pins), and using Boolean algebra or a truth table to optimise these equations. What is always the maximum latency for combinatorial circuits? Build your optimised 4-bit adder block.

   (d) Build a circuit of an 8-bit two's complement carry lookahead adder by creating a "carry lookahead unit" that uses 2 of your own 4-bit adder blocks.

   (e) On this 8-bit adder circuit, create an extra output bit, denoting overflow.

   (f) Verify that your 8-bit two's complement carry lookahead adder is correct. Do this by connecting two 8-bit inputs to both the carry lookahead 8-bit adder and the ripple carry 8-bit adders, and comparing the outputs.

   (g) To compare the carry lookahead 8-bit adder and the ripple carry 8-bit adders, count the latency of both blocks.

6. To prepare for the next lab session, read sections B.5 and B.6 of Appendix B.