Computer Systems and -architecture

Project 1: Gates and Wires

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.1 and B.2 of Appendix B. You can only use the following Logisim libraries for this assignment: Base, Wiring, Gates, Input/Output. Read `http://www.cburch.com/logisim/docs/2.7/en/html/guide/tutorial/` to get started with Logisim.

1. Prove De Morgan's theorems by composing a truth table. The theorems are
   (a) $\overline{A+B} = \overline{A} \cdot \overline{B}$
   (b) $\overline{A \cdot B} = \overline{A} + \overline{B}$

2. Prove by using Boolean algebra that

$$E = ((A \cdot B) + (A \cdot C) + (B \cdot C)) \cdot (\overline{A \cdot B \cdot C})$$
$$\Longleftrightarrow$$
$$E = (A \cdot B \cdot \overline{C}) + (A \cdot \overline{B} \cdot C) + (\overline{A} \cdot B \cdot C)$$

   In each step, write *each* used algebraic law.

   Remark: the last equation is a normalized *sum of products* representation, and will prove to be useful in implementing the behaviour of such algebraic equations.

3. Use Logisim to build a NAND port as a new component out of AND, OR an NOT gates.

   Like the set of AND, OR, NOT gates, the singleton NAND gate set is functionally complete. This means that any algebraic expression or truth table can be implemented by using only NAND gates. Show that NAND is functionally complete by building an AND, OR and NOT gate using only your own NAND component in Logisim.

4. Implement the following truth table in Logisim using AND, OR and NOT gates. To do this, first write the truth table as a Boolean algebraic expression. Use as few ports as possible. A, B and C are inputs, and X and Y are outputs.

A	B	C	X	Y
0	0	0	1	1
0	0	1	1	1
0	1	0	0	1
0	1	1	0	0
1	0	0	1	1
1	0	1	0	0
1	1	0	0	1
1	1	1	0	0

5. In Logisim, show the two-digit hexadecimal ASCII representation of the characters you type. For example, the character "n"'s ASCII representation is "6E". Use the *Keyboard* and *Hex Digit Display* components of the *Input/Output* library. You will also need to use *Splitter* components of the *Base* library. Check the documentation pages on how to use these components: `http://www.cburch.com/logisim/docs/2.7/en/html/libs/`

6. Build a circuit that shows the representation of numbers 0 to 7 on a LED display.

   - compose a truth table with binary outputs for every LED on the *7-Segment Display*.
   - find the Boolean algebraic expressions for this truth table. Discuss why these expressions are useful in the context of building a circuit.
   - implement the circuit in Logisim. Use the *7-Segment Display* of the *Input/Output* library.

   The LED display should display numbers as follows:

Decimal number	binary representation	LED
0	000	
1	001	
2	010	
3	011	
4	100	
5	101	
6	110	
7	111	

7. To prepare for the next lab session, read section B.3, B.5 and B.6 of Appendix B.