Computer Systems and -architecture

Project 7: Datapath in Use

1 Ba INF 2016-2017

Bart Meyers
bart.meyers@uantwerpen.be

Don't hesitate to contact the teaching assistant of this course. You can reach him in room M.G.3.17 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: **December, 21 2016, 23u55**
- Evaluation and feedback: **December, 23 2016**

Project

Read section 4.9 of Chapter 4. You can use all Logisim libraries for this assignment.

1. Exceptions are a very important part of a datapath and control. In this exercise, you will add a basic form of exception handling to your datapath: when an exception is detected, your program counter should halt at the instruction that caused the exception. Arithmetic overflow should be detected and supported.

   Think about enhanced versions of exception control. What is necessary in order to add a more advanced form of exception handling to a datapath with our instruction set?

2. How would you translate the following pseudo-instructions to assembler, using the instruction set of your datapath? Demonstrate on your datapath:

   (a) A 'zero rd' instruction that sets \$rd to 0.

   (b) A 'li rd imm' (load immediate) instruction, loading a 16-bit constant imm in register rd.

   (c) A 'inv rd rs' (numerical inverse) instruction, setting \$rd to -\$rs.

(d) A 'j imm' (jump) instruction, that sets $pc to a 16-bit constant imm.

(e) A 'bgte rs rt imm' (branch if greater than or equal) instruction, that jumps to $pc+1+imm if $rs >= $rt.

3. Demonstrate the proper operation of your datapath by providing a number of small assembler programs. Try to use subroutines at least once. Don't forget to initialize the stack pointer.
   Provide the programs below.

   (a) A program that calculates the Fibonacci numbers (`http://en.wikipedia.org/wiki/Fibonacci_numbers`) and stores them in memory. After which number does overflow occur?

   (b) A program that finds an element in a sorted array (with given size). The index is written to a register of your choice, and -1 is written if it does not occur. Use the binary search algorithm (`https://en.wikipedia.org/wiki/Binary_search_algorithm`).