Saarland University
Department 6.2 – Computer Science
Dr. M. A. Hillebrand
Prof. Dr. W. J. Paul

Computer Architecture I – WS 06/07

Exercise Sheet 6 **(due: 04.12.06)**

Exercise 1: (processor correctness) **(3+4+4+4+4=19 points)**

In class, you have seen the principle how to proof the correctness of the sequential DLX implementation versus the instruction set architecture with the simulation relation. In this exercise, you are asked to proof the correctness for five cases with the notation from the lecture. Hence, for each of the following cases, you have to determine the correct path or paths through the control automaton and compute the updates of all implementation registers. Then argue about the correctness and the ISA equivalence.

Proof the correctness for the following cases:

1. The add case, i. e. for an instruction $I(c) = add\ RD\ RS1\ RS2$.
2. The load case, i. e. for an instruction $I(c) = x\ RD\ RS1\ imm$ for $x \in \{lb, lh, lw, lbu, lhu\}$.
3. The store case, i. e. for an instruction $I(c) = x\ RD\ RS1\ imm$ for $x \in \{sb, sh, sw\}$.
4. The jump-and-link case, i. e. for an instruction $I(c) = x\ RD\ RS1\ imm$ for $x \in \{jal, jalr\}$.
5. The branch case, i. e. for an instruction $I(c) = x\ RD\ RS1\ imm$ for $x \in \{beqz, bnez\}$.

Exercise 2: (self-modifying code) **(4 points)**

Write an ISA program that writes in each memory cell from address 1024 to address 32767 the address of the current memory cell modulo 2^8. Thereby, fulfill the following constraints:

- You are only allowed to use absolute addresses, i. e. on a memory access the effective address $ea(c)$ have to be computed as $(R0 + imm)$.
- Your code consists of eight or less instructions (it is possible with 6).

You may assume that there is a common instruction and data memory and your code starts at memory address 0.

Exercise 3: (program execution) **(2+3+2=7 points)**

Consider the following ISA code:

0: $add(R2, R0, R3)$
4: $add(R1, R0, R4)$
8: $beqz(R1, 16)$
12: $add(R2, R2, R1)$
16: $subi(R1, R1, 1)$
20: $j(-12)$

Where the notation from class is used, i. e. $instr.(RD, RS1, R2/imm)$. Assume the processor stops the execution after the last code line.

1. What does the program compute? Shortly explain your answer.
2. Assume, we execute the code on the *pipelined* implementation of the processor. Moreover, assume that the general purpose register is only written in the write back stage and we do not use forwarding. Determine an execution table as in class starting at time t until $t + 15$. Make sure that you insert the so-called 'bubbles' at situations where hardware interlock occurs.
3. Assume the following initial values for $GPR(3)$ and $GPR(4)$:

   $$\langle GPR(3)\rangle = \langle GPR(4)\rangle = 2$$

   Perform an execution time analysis for a program execution on the *sequential* implementation based on the definition of execution cycles $s(c)$ for an instruction on page 40 of the lecture notes revision 1.50.