ECE 201 Spring 2022 - Project 3

Assigned by Prof. Xiaochen Guo

Due Mar. 31st, 2022 23:59pm

In this project, you will design and simulate a **single-cycle datapath** by running instructions in the *swap* function of the bubble sort program discussed in Lecture 7 (page 13) **on paper by hand**. The datapath is illustrated in Figure 4.23 on the textbook (and page 27 of Lecture 12 slides). However, this datapath does not support the branch register instruction.

1. (30 points) Please modify the datapath in Figure 4.23 so that the datapath can support the branch register instruction (e.g., BR X30). In the report, please include a drawing of the modified part of the datapath.

2. (70 points, 10 points per instruction) Simulate your datapath from part 1 on paper by hand. The initial register file contains the following values in the corresponding registers, the memory contains the following values at the corresponding addresses (assume *little endian*; data and instructions are in data cache and instruction cache already), and the initial value of the PC is 0x80 0000 (*this is a 64-bit datapath*, the non-specified upper bits are all zeros).

Register file:

X0	0x1000 0000
X1	0x5
X28	0x7F FFFF FC00
X29	0x7F FFFF FC20
X30	0x40 101C

Memory:

Address	Data	Address	Data
0x80 0000	0xD360 0C2A	0x80 0018	0xD600 03C0
0x80 0004	0x8B0A 000A	…	
0x80 0008	0xF840 0149	0x1000 0028	0xFFFF 8888
0x80 000C	0xF840 814B	0x1000 002C	0x4444 0000
0x80 0010	0xF800 014B	0x1000 0030	0x2222 CCCC
0x80 0014	0xF800 8149	0x1000 0034	0x1111 1111

In the report, for each instruction, please include:
(1) the value of the PC, and **changed values** in register file and memory (data cache) *right after* the positive edge of the clock in **hexadecimal**.
(2) the **changed values** (*i.e.,* the values that are related to the corresponding instruction in that cycle) of the following wires (please use the wire names in the following list) in **hexadecimal** *right before* the positive edge of the clock of the corresponding cycle. The wires are grouped

with respect to the microarchitecture structures. If you would like to make modifications and addition to this list due to the modification in Part 1, please specify and explain in your report.

PC+4 Adder: PcPlus4 (output)
Instruction memory: Instruction address (input), Instruction (output)
Control: Opcode (input), Reg2Loc Uncondbranch (output), Branch (output), MemRead (output), MemtoReg (output), ALUOp (output), MemWrite (output), ALUSrc (output), RegWrite (output)
Register file: Read register 1 (input), Read register 2 (input), Write register (input), Write data (input), Read data 1 (output), Read data 2 (output)
Sign-extend: Extended data (output)
Branch target adder: Branch target (ALU result of this adder, output)
ALU control: ALUctrl (output)
ALU: ALU result (output), Zero (output)
Data memory: Data address (input), Load memory data (output)
PC: Next PC (input)