Processor Architecture Assignment (Learning)

CSCI 389: Computer Systems

Fall 2021

This assignment is an opportunity to test your understanding of processor architecture and receive feedback. Point values are assigned so that you can differentiate between large and small mistakes, but this assignment does not affect your grade.

Due Date: Friday, February 11th at 10:00 am.

1. (8 points) **Decoding Assembly.** Using the ALPHA instruction set (from the Processor Architecture Slides), decode the following hexadecimal instructions. This is a four step process: 1. Convert the instruction to binary. 2. Map the binary bits to the different fields of the instruction (use the opcode to determine instruction type). 3. Convert the values of the different fields of the instruction from binary to hexadecimal. 4. Check the opcode (and perhaps func field) for the type of operation and the arguments. Specify the operation type. The arguments can be left in hexadecimal, but specify input vs output.

   (a) (4 points) 0x44720805

   (b) (4 points) 0xb532bad4

   A.

   Binary is: 01000100011100100000100000000101

   Opfield = $010001_2 = 11_{16}$

   RA = $00011_2 = 3$

   twelth bit = 0

   RB = $10010_2 = 12_{16}$

   Funct = $1000000_2 = 40$ (XOR)

   RC = $00101_2 = 05$

   Translated XOR R3 with R12 place in R5.

   B.

   Binary is 10110101001100101011101011010100

   Opfield is $101101_2 = 2D_{16}$

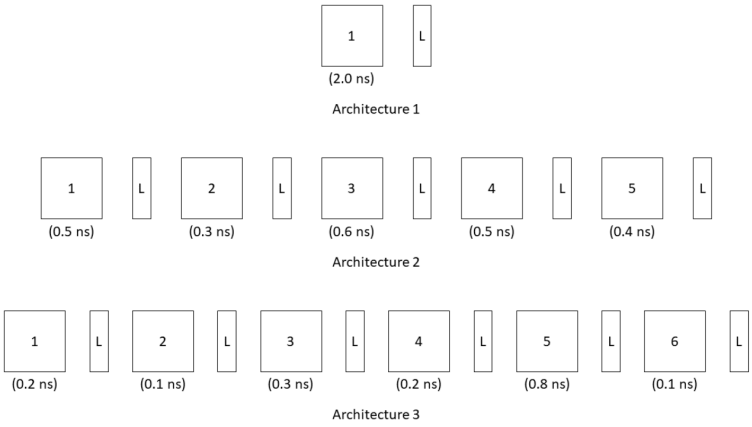
   RA is $01001_2 = 19_{16}$

   RB = $10010_2 = 14_{12}$

   Offset is: $1011101011010100_2 = BAD4_{16}$

   Translated is Store R19 in R19 offset by $BAD4$

2. (12 points) **Latency and Bandwidth.** Consider the pipeline architectures shown below:



Compute the latency and bandwidth for each architecture, assuming that latches (registers) take 0.1 ns.

(a) (4 points) Architecture 1.

(b) (4 points) Architecture 2.

(c) (4 points) Architecture 3.

latency = num stages $\cdot$ (max stage time)

bandwidth = $\frac{1}{\text{Max stage time}}$

A. Latency = $1 \cdot (2 + 0.1) = 2.1$ ns

bandwidth = $\frac{1}{2.1} = 0..47$ Ghz

B. Latency = $5 \cdot (0.6 + 0.1) = 5 \cdot 0.7 = 3.5$ ns

bandwidth = $\frac{1}{0.7} = 1.43$ Ghz

C. Latency = $6 \cdot (0.8 + 0.1) = 6 \cdot 0.9 = 5.4$ ns

bandwidth = $\frac{1}{0.9} = 1.\overline{11}$ Ghz

3. (20 points) **Analyzing Code Execution.** Consider the following instructions:

```
00 loop:
01 r1 * 8 => r4
02 r2 + r4 => r5
03 r3 + r4 => r6
04 load[r5 + 0] => r7
05 load[r6 + 0] => r8
06 r7 * r8 => r5
07 r1 * 2 => r6
08 r5 + r6 => r5
09 store r5 => [r4 + 0]
10 r1 + 1 => r1
11 compare r1, r10
12 branch if not zero => loop
```

(a) (4 points) List the data hazards for the instruction 04. For each hazard, label its type.

(b) (4 points) List the data hazards for the instruction 09. For each hazard, label its type.

(c) (12 points) Draw a pipeline diagram showing the progression of one iteration of these instructions through a basic 5-stage pipeline that has EX-EX, MEM-MEM, and MEM-EX forwarding, but lacks more advanced parallelism techniques. Make sure to label all instances of forwarding.

A.

RAW hazard from line 2

B.

RAW hazard. R5 is written on line 8 but can be prempticely read on line 9

WAW R4 can be clobbered on line 1

4. (5 points) **Loop Unrolling.** Consider the following instructions:

```
loop:
load[r1 + 0] => r3
r2 + r3 => r2
store r2 => [r1 + 0]
r1 + 8 => r1
branch if r1 < r4 => loop
```

Unroll this loop so that 4 iterations happen between branches. Make sure that your unrolled loop is correct.

```
loop:
load[r1 + 0] => r3
r2 + r3 => r2
store r2 => [r1 + 0]
r1 + 8 => r1
load[r1 + 0] => r3
r2 + r3 => r2
store r2 => [r1 + 0]
r1 + 8 => r1
branch if r1+1 < r4 => loop

load[r1 + 0] => r3
r2 + r3 => r2
store r2 => [r1 + 0]
r1 + 8 => r1
```