Kristy Nguyen
Dr. Ed Jorgensen
CS 219-1002: Computer Organization
22 February 2021

CS 219 – Assignment #5

Purpose: Become familiar with the MIPS architecture instruction formats, floating point data representation issues, and multiplication algorithms

Points: 100

Assignment:

1) According to the text (Ch 2), what are the three (3) underlying design principals? [8 pts]

The first of three underlying principles of hardware design is Simplicity favors regularity. The second principle is Smaller is faster. The final hardware design principle is Good design demands good compromises.

2) List the two architectural models and describe the key difference each. Include which is the most commonly used. *Note*, not in text. [5 pts]

The two architectural models are the Von Neumann architecture and the Harvard architecture. Von Neumann uses the same memory for instructions and data with a typical memory layout of program/data/stack. Harvard uses different memory for instructions and data. The most commonly used is the Von Neumann architecture.

3) List the six possible fields (include the name and meaning) in a MIPS instruction? [8 pts]

The six possible fields in a MIPS instruction are:

- op → operation / op field
- rs → 1st register source
- rd → 2nd register source
- rd → destination register
- shamt → shift amount (for logicals)
- funct → function (variant in op field)

4) What are the three basic instruction formats? Show the layout/fields for each. [3 pts]

R format:

- field: `op / rs / rt / rd / shamt / funct`
- bits: `6 / 5  / 5  /  5 /   5   / 6`

I format:

- field: `op / rs / rd / address/constant`
- bits: `6 / 5  / 5  /  16`

J format:

- field: `op / address`

Kristy Nguyen
Dr. Ed Jorgensen
CS 219-1002: Computer Organization
22 February 2021

- bits: 6 / 26

5) Referring to **mips.s** program from assignment #3

a) what is the opcode for the "**add $s0, $s0, $t5**" instruction? Show result in binary, grouped by field. [3 pts]

```
Hex: 020d8020
Binary: 000000/10000/01101/10000/00000/100000
R-format
op: 000000 → register based arithmetic operation
rs: 10000 → 16 → register $16 → $s0
rt: 01101 → 13 → register $13 → $t5
rd: 10000 → 16 → register $16 → $s0
shamt: 00000 → shift amount is N/A
funct: 100000 → add (20hex)
```

b) what is the value (in binary and decimal) for ***rd*** field and which register (name) is indicated for that field? [2 pt]

rd: 10000_2 , 16_{10} , register $s0

6) Given the hex opcode **0x00954020** what is the instruction (including register names)? Show field values. [5 pts]

```
000000/00100/10101/01000/00000/100000
R-format
op: 000000 → register based arithmetic operation
rs: 00100 → 4 → register $4 → $a0
rt: 10101 → 21 → register $21 → $s0
rd: 01000 → 8 → register $8 → $t0
shamt: 00000 → shift amount is N/A
funct: 100000 → add (20hex)
```

Instruction: add $t0, $a0, $s0

7) Given that $x = 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0101\ 1011_2$ and $y = 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 1101_2$ representing two's compliment signed integers, perform the following operations showing all work. [3 pts each]

a) x + y

```
                                  11 111
  0000 0000 0000 0000 0000 0000 0101 1011→   91₁₀
+ 0000 0000 0000 0000 0000 0000 0000 1101→  +13₁₀
  ---------------------------------------   ---
  0000 0000 0000 0000 0000 0000 0110 1000→  104₁₀
```

b) x – y → (x + (-y))

Kristy Nguyen

Dr. Ed Jorgensen

CS 219-1002: Computer Organization

22 February 2021

```
y: 0000 0000 0000 0000 0000 0000 0000 1101
```
→ decimal: $8+4+1=13_{10}$

Kristy Nguyen
Dr. Ed Jorgensen
CS 219-1002: Computer Organization
22 February 2021

```
          → negative 2's complement:
(y):      1111 1111 1111 1111 1111 1111 1111 0010
        + 0000 0000 0000 0000 0000 0000 0000 0001
          ---------------------------------------
(-y):     1111 1111 1111 1111 1111 1111 1111 0011

         11111 1111 1111 1111 1111 1111 111   11
(x):      0000 0000 0000 0000 0000 0000 0101 1011→  94₁₀
        + 1111 1111 1111 1111 1111 1111 1111 0011→ -13₁₀     :(-y)
          ---------------------------------------   ---
          0000 0000 0000 0000 0000 0000 0100 1110→  78₁₀
```

8) Given that x = 1111 1111 1111 1111 1001 1011 0011 1110$_2$ and y = 0000 0000 0000 0000 0000 0110 0010 0101$_2$ representing two's compliment signed integers, perform the following operations showing all work. [3 pts each]

a) x + y

```
                                11 11     111 1
          1111 1111 1111 1111 1001 1011 0011 1110→ -25794₁₀
        + 0000 0000 0000 0000 0000 0110 0010 0101→ + 1573₁₀
          ---------------------------------------   ------
          1111 1111 1111 1111 1010 0001 0110 0011→ -24221₁₀
```

b) x – y → (x + (-y))

```
y: 0000 0000 0000 0000 0000 0110 0010 0101
          → decimal: 1+4+32+512+1024=1573₁₀

          → negative 2's complement:
(y):      1111 1111 1111 1111 1111 1001 1101 1010
        + 0000 0000 0000 0000 0000 0000 0000 0001
          ---------------------------------------
(-y):     1111 1111 1111 1111 1111 1001 1101 1011

         11111 1111 1111 1111 1111  111 1111 11
(x):      1111 1111 1111 1111 1001 1011 0011 1110→  -25794₁₀
        + 1111 1111 1111 1111 1111 1001 1101 1011→+(- 1573₁₀) :(-y)
          ---------------------------------------   -------
          1111 1111 1111 1111 1001 0101 0001 1001→  -27367₁₀
```

Kristy Nguyen
Dr. Ed Jorgensen
CS 219-1002: Computer Organization
22 February 2021

9) Briefly describe what is meant by overflow and underflow conditions that might be produced after a floating point operation (for the exponent). [4 pts]
After shifting the smaller number to the right until its exponent matches the larger exponent, adding the significands, and normalizing the sum by either shifting right and incrementing the exponent or shifting left and decrementing the exponent, we must check for overflow or underflow. This means we make sure that the exponent still fits in its field, such as 127 ≥ exponent ≥ -126.

10) What are the basic elements/steps required for floating point addition? *Note*, answer with a short paraphrased description of each step. [4 pts]

1. Compare the exponents of the two numbers; shift the smaller number to the right until its exponent would match the larger exponent.
2. Add the significands
3. Normalize the sum by either shifting right and incrementing the exponent or shifting left and decrementing the exponent., checking for overflow or underflow.
4. Round the significand to the appropriate number of bits, normalize again if necessary.

11) What are the basic elements/steps required for floating point multiplication? *Note*, answer with a short paraphrased description of each step. [4 pts]

1. Add the biased exponents of the two numbers, subtracting the bias from the sum to get the new biased exponent.
2. Multiply the significands
3. Normalize the product if necessary, shifting it right and incrementing the exponent, and check for overflow or underflow.
4. Round the significand to the appropriate number of bits, normalize again if necessary.
5. Set the sign of the product to positive if the signs of the original operands are the same; if they differ make the sign negative.

12) Given the below data declarations (which are in a provided main):

```
 .data
# -----
# Single precision

fval1: .float 9.75
fval2: .float 5.5
fval3: .float 12.25
fval4: .float 3.6

fans1: .float 0.0
fans2: .float 0.0
```

Kristy Nguyen
Dr. Ed Jorgensen
CS 219-1002: Computer Organization
22 February 2021

```
fval: .float 0.001
fsum: .float 0.0

# -----
# Double precision

dval1: .double 9.75
dval2: .double 5.5
dval3: .double 12.25
dval4: .double 3.6

dans1: .double 0.0
dans2: .double 0.0

dval: .double 0.001
dsum: .double 0.0
```

Write a simple MIPS program to compute the following floating point formulas:

```
# fans1 = fval1 + fval2
# fans2 = fval3 + fval4
# fsum = 0.0
# do 1000 times
# fsum = fsum + fval

# dans1 = dval1 + dval2
# dans2 = dval3 + dval4
# dsum = 0.0
# do 1000 times
# dsum = dsum + dval
```

Show the displayed results of the program (i.e,. console output) and explain. [10 pts]

For this question:

1) Submit a copy of the console output (from the log file).
   a) It is ***not*** necessary to include the program source or text segment information in the log file.
2) Submit a copy of the source file.

13) Perform the following multiplication problems using Booths algorithm (see handout). Assume an 8-bit machine. [8 pts each]

a) 5 × 10
b) 6 × 7
c) 17 × -6
d) -14 × 4