COMP3401 Computer Organization

Fall 2023– HW#3 [CO2,3]

DUE: 7.12.2023

1) If A, B, C are 4 bit, write the result of the operations. State if the operation results an overflow

a) A=0100, B=0101 A and B are signed 2’s complement integers

A+B=?

A is 4 in decimal
B is 5 in decimal
A + B = 1001 is 9 in decimal overflow occurs

b) A=1010, B=0101 A and B are signed 2’s complement integers

A+B=?

A is -6 in decimal Why? We accept the first digit of the number A as the number that determines the sign. Since the first number of A is 1, the number A is negative. We use the 2's complement formula.
A -> 1010 then 0101 then 0101 + 1 = 0110
B is 5 in decimal
A+B = 1111 is -1 in decimal no overflow

c) A=0111, B=1111 A and B are signed 2’s complement integers

A-B=?
A = 0111 is 7 in decimal
B = 1111 is -1 in decimal (2's complement formula)
A – B = 1000 is 8 in decimal overflow occurs

d) A=0111, B=1101 A and B are **unsigned** integers

A+B = ?

A = 0111 is 7 in decimal

B = 1101 is 13 in decimal

A+B = 10100 is 20 in decimal Overflow occurs

COMP3401 Computer Organization
Fall 2023– HW#3 [CO2,3]

DUE: 7.12.2023

2. Manually create a 8 bit fp point representation -similar to IEEE754-, that has 1 sign bit, 3 bits of exponent bits, 5 fraction bits.

   a. Show the range of (min, max) fp numbers that can be represented by this encoding.
   b. Represent 3.14 in binary using your own representation.

a. (We created these values as if there was no reserve for the exponent.)
We create manually 1 111 11111
1->sign bit so our value is negative
111-> 7 exponent bit value in decimal
11111-> fraction part is = (1/2+1/4+1/8+1/16+1/32)=0,921875
(-1)^sign x(1+fraction)x2^(exponent-bias) according to that instruction :
We reach that :
(-1)^1x(1+0,921875)x(7-3) = - 30,75 minimum range

If we set the sign bit to 0 instead of 1, we can find the max range:
We create manually 0 111 11111
0->sign bit so our value is negative
111-> 7 exponent bit value in decimal
11111-> fraction part is = (1/2+1/4+1/8+1/16+1/32)=0,921875
(-1)^sign x(1+fraction)x2^(exponent-bias) according to that instruction :
We reach that :
(-1)^0x(1+0,921875)x(7-3) = + 30,75 minimum range

b. Our value is positive so that our sign bit must be 0

Firstly, we convert 3.14 to binary: 11.00100

Secondly, we normalize the binary representation: 1.100100×2^1.

Thirdly, we adjust the exponent: 1.100100×2^(1+3)=1100.100

Then we reach exponent bit's value 4 in decimal so now we placed it in the table in binary form -> 100

We obtained the binary equivalent of the fraction part from the binary equivalent of the number: 10010

So, the representation of 3.14 in this 8-bit format is:

0 100 10010

COMP3401 Computer Organization

Fall 2023– HW#3 [CO2,3]

DUE: 7.12.2023

3. Write the MIPS assembly code of the following C/JAVA code.

```
If (x>=5.0)
        z=2.0* (y-2.0)/(x-5.0);
else
        z= y;
```

where x, y, z are <u>all double precision floating point</u> numbers. Assume x, y, z are in memory locations $sp, $sp+8, $sp+16, respectively. You can use the following and other MIPS instructions in the reference card.

```
ldc1 $f1, $0($sp)  # Load a double precision value into register $f1 from the stack at offset 0.
ldc1 $f2, $8($sp)  # Load a double precision value into register $f2 from the stack at offset 8.
ldc1 $f3, $16($sp) # Load a double precision value into register $f3 from the stack at offset16.
ldc1 $f4, const2($gp) #Load a double precision value into register $f4 from global memory at address const2 + $gp.
ldc1 $f5, const5($gp) #Load a double precision value into register $f5 from global memory at location const5 + $gp.

c.le.d $f5, $f1 #Compare to see if $f5 and $f1 are equal or less.
bc1t Label1 #If the previous comparison holds true, branch to Label 1.
bc1f Label2 #If the previous comparison is not true, branch to Label 2.

Label1:
sub.d $f2, $f2, $f4 #Subtract $f4 from $f2.
sub.d $f1, $f1, $f5 #Subtract $f5 from $f1.
div.d $f3, $f2, $f1 #Divide $f2 by $f1 and store the result in $f3.
mul.d $f3, $f3, $f4 #Multiply $f3 by $f4.
j endLabel #Jump to endLabel

Label2:
sdc1  $f2, 8($sp) #At offset 8, place the value of $f2 in the stack.
ldc1 $f3, 8($sp) #A double precision value at offset 8 should be loaded into $f3 from the stack.
j endLabel #Jump to endLabel

endLabel:
sdc1 $f3, 16($sp) #At offset 16, place the value of $f3 in the stack.
```

MELIKE TEPELI 20COMP1012