Assignment 2
Due: Thursday, October 24, 2013

- There are 12 questions and you are required to answer 10 questions. Last two questions are exercise purpose only.
- **Show all your work clearly and legibly for all your answers.**
- The numbers under [ ] indicate the marks allotted to each question.
- The assignment (written or printed) must be stapled and put in a folder
- You may hand in your assignment
  - to the instructor on due date by end of the class or
  - deposit it on hand-in lockers on 4th floor of the E2 building besides the elevator by 430 pm on the due date

---

1. [10] Given the bit pattern: 1000 1111 1110 1111 1100 0000 1111 1111 what does it represent, assuming that it is

   a. [2] A two's complement integer?
   b. [3] An unsigned integer?
   c. [5] A MIPS instruction?

Solution:

a) [2]: The sign bit is 1, so this is a negative number. We first take its 2's complement.

A = 1000 1111 1110 1111 1100 0000 1111 1111
-A = 1's complement of A +1
= 0111 0000 0001 0000 0011 1111 0000 0000 + 1
= 0111 0000 0001 0000 0011 1111 0000 0001 (a positive integer)

$= 2^{30} + 2^{29} + 2^{28} + 2^{20} + 2^{13} + 2^{12} + 2^{11} + 2^{10} + 2^9 + 2^8 + 2^0$
= 1073741824 + 536870912 + 268435456 + 1048576 + 8192 + 4096 +2048 + 1024+ 512 + 256 + 1
= - 1,880,112,897

b) [3]

A = 1000 1111 1110 1111 1100 0000 1111 1111

= 8FEFC0FF
= 8 * 16^{7} + 15 * 16^{6} + 14 * 16^{5} + 15 * 16^{4} + 12*16^{3} + 15 * 16^1 + 15 * 16^0
= 2,147,483,648 + 251,658,240 + 14,680,064 + 983,040 + 49,152 + 240 + 15
= 2 414 854 399

c) [5]
opcode (6 bits) = 100011 = 35 = lw (this should immediately tell you that the lower 16 bits are for address field)
rs (5 bits) = 11111 = 31
rt (5 bits) = 01111 = 15
address = 1100 0000 1111 1111
Since the address is negative, we have to take its 2's complement.
2's complement of address = 0011 1111 0000 0001

Therefore, address = - (2^{13}+2^{12}+2^{11}+2^{10}+2^9+2^8+2^0) =
= - (8192 + 4096+2048+1024+512+256+1) = -16129
Therefore, the instruction is: lw 15, -16129(31) OR lw $t7. -16129($ra)

2. [7] Assume a=$s0; b=$s1; c=$s2; d=$s3; e=$s4; f=$s5;
Convert the following high level language instruction in to MIPS instruction using minimum number of instructions and use the registers provided above.
a. [1]

a = b – e;

b. [2]

f= c + (d – 7);

c. [4] What is the final value of a and f if b, c, d and e have values 1, 2,3 and 4 respectively. Provide your answers in 32-bit 2's complement representation.

Solution:

a. [1] sub $s0, $s1, $s4 (1-4 = -3)
b. [2] addi $s5, $s3, -7 (3-7 = -4)

add $s5, $s2, $s5 (-4+2 = - 2)

c. [4] a = -3 = 1111 1111 1111 1111 1111 1111 1111 1101

f = -2 = 1111 1111 1111 1111 1111 1111 1111 1110

3. [5] Two friends, A and B, are arguing. A says, "All integers greater than zero and exactly divisible by six have exactly two 1s in their binary representation." B disagrees. She says, "No, but all such numbers have an even number of 1s in their representation." Do you agree with A or with B, or with neither?
(Hint: Look for counterexamples)

Solution:

6 = 4+2 = 0110
12 = 8+4 = 1100
18 = 16+2 = 10010
24 = 16+8 = 11000
30 = 16+8+4+2 = 111010 (here more than two 1s - therefore, A is wrong)
36 = 32+4 = 100100
42 = 32+8+2 = 101010 (odd number of 1s - B is wrong)

4. [10] The table below shows the values of an array stored in memory. Assume that base address of the array is stored in register $s2 and offset it with respect to the base address of the array. The data shown represents the variable *Data*, which is an array of type *int*, such that Data[0]=14 (that is, the first data of the array is the first element of the array). Assume the machine is byte-addressable and a word consists of 4 bytes.

Address	data
20	14
24	15
28	13
32	12
36	11

a. [8] For the memory locations in the above array, write a MIPS code to reorder the data from lowest to highest; that is to place the smallest data value in the earliest (address) memory location. Use a minimum number

of MIPS instructions. (Hint: first write the code in high level language with minimum number of instructions and then translate it to MIPS)

b. [1] How many (minimum) MIPS instructions are required for this code.

c. [1] List the MIPS instructions you have used and their type

Solution:

```
temp = Array[0];
temp2 = Array[1];
Array[0] = Array[4];
Array[1] = Array[3];
Array[3] = temp;
Array[4] = temp2;
```

```
lw   $t0, 0($s2)
lw   $t1, 4($s2)
lw   $t2, 16($s2)
sw   $t2, 0($s2)
lw   $t2, 12($s2)
sw   $t2, 4($s2)
sw   $t0, 12($s2)
sw   $t1, 16($s2)
```

b. [1] 8

c. [1] lw and sw (I – Format)

5. [4] Assume that the register $s0 =$(a0800000)_{16}$ and $s1 has the values given below. If the instruction *sub $s0, $s0, $s1* is executed, will there be overflow in these two values of $s1? Show your work.

   a. $s1= $(-1)_{10}$

   b. $s1 = $(2048)_{10}$

Solution:

a. **no overflow** (subtracting two operands of same sign)
0xa0800000 - 0xFFFFFFFF = 0xa0800000 + 0x00000001 = 0xa0800001.
Carry into the most significant bit is equal to the carry out (0) so there is no overflow.

b. **overflow** (subtracting two operands of opposite sign)
0xa0800000 - 0x00000800 = 0xa0800000 + 0xFFFFF800 = 0xFFFFF800. In this case, there was overflow.

6. [4] Assume that the register $s0 =$(7FFFFFFF)_{16}$ and $s1 has the values given below. If the instruction *sub $s0, $s0, $s1* is executed, will there be overflow in these two values of $s1?

   a. $s1= $(-1)_{10}$

   b. $s1 = $(1024)_{10}$

Solution:

a. **overflow** (subtracting two operands of opposite sign)
0x7FFFFFFF - (-1) = 0x7FFFFFFF + 0x00000001 = 0x80000000. This is overflow; The most significant bit had a carry in of 1 and a carry out of 0.

b. **no overflow** (subtracting two operands of same sign)
0x7FFFFFFF - (1024) = 0x7FFFFFFF + 0x7FFFC00 = 0x7FFFBFF.
The carry in and carry out on the most significant bit match (1), so there is no overflow.

7. [3] If the instruction set of the MIPS architecture is modified, the instruction format must also be changed. If the MIPS architecture has 64 registers, how will the bit fields of the R-type instructions would change. List all the fields and their corresponding bit lengths.

8. [3] If the instruction set of the MIPS architecture is modified, the instruction format must also be changed. If the MIPS architecture has 64 registers, how will the bit fields of the I-type instruction would chang

Solution for 7 and 8 :

In a 32-register MIPS architecture, we have 5 bits for register fields (rs, rt and rd) to select one of the 32 registers

With 64 registers, the bit fields for source, second source and destination registers should have enough bits to represent one of the 64 registers.
Therefore, for a 64-register MIPS architecture**, we would need 6 bits for register (rs, rt and rd) fields. Other fields do not change.**

	a. [3] R- type	**b. [3] I-type**
opcode:	6	6
rs field	**6**	**6**
rt field	**6**	**6**
rd field	**6**	NA
shamt field	5	NA
function field	6	16
Total bits	35	34

9. [10] If the instruction set of the MIPS architecture is modified, the instruction format must also be changed. If the MIPS architecture has two times the number of instructions, how will the bit fields of the following instruction type would change
   a. [5] R-type instructions
   b. [5] I-type instructions

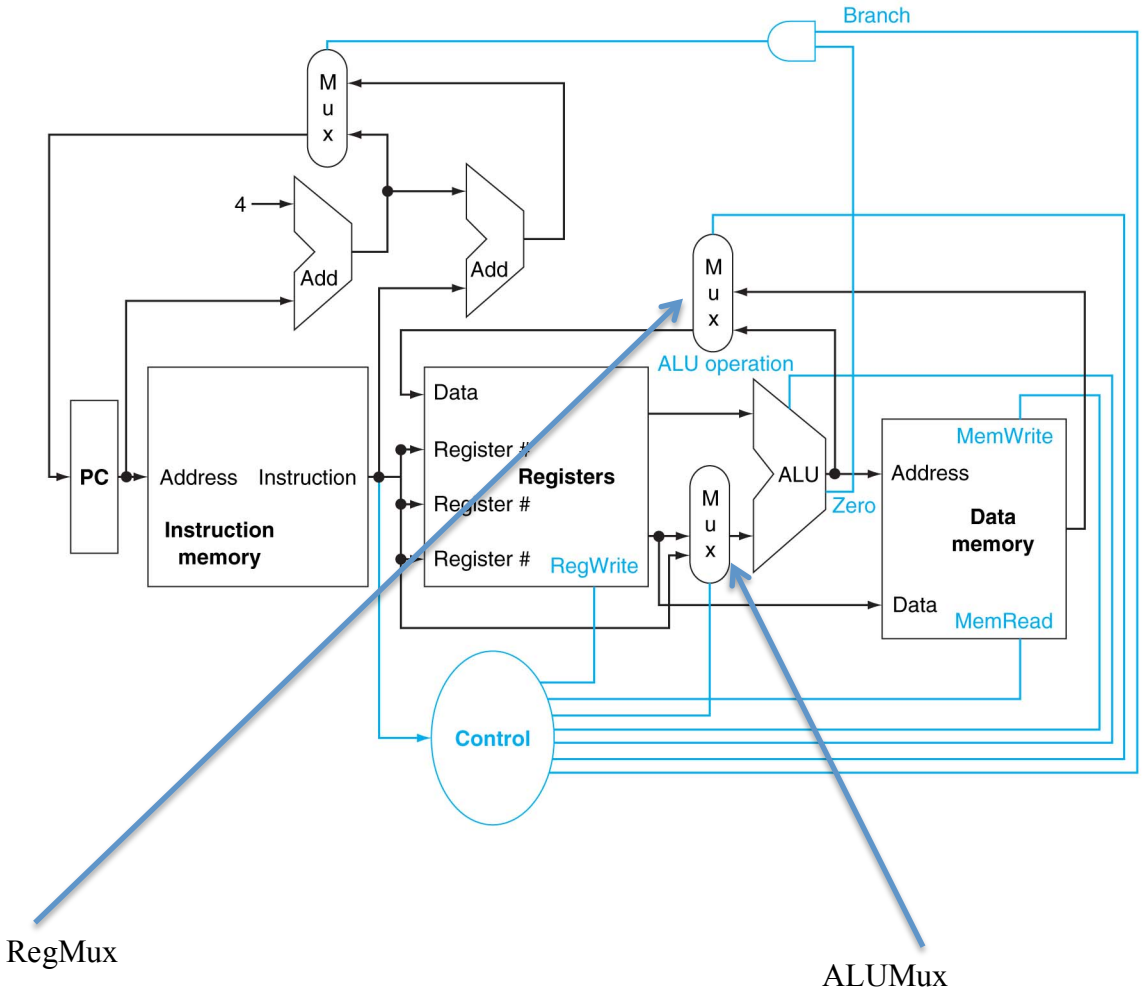
Solution:

The number of instructions are represented by the opcode in the instruction format. With 6 bits allotted for the opcode we would have 2^6 instructions (64). Two times as many instructions implies 64*2=128 instructions. To represent one of these 128 instructions **we would need 7 bits for the opcode** ($2^7 = 128$) for both R and I-type instructions. **Other fields do not change.**

	a. [5] R- type	**b. [5] I-type**
opcode:	**7**	**7**
rs field	5	5
rt field	5	5
rd field	5	NA
shamt field	5	NA
function field	6	16
Total bits	33	33

Refer to the following figure for answering questions 10 through 12 (figure 4.2 in the 4th edition revised printing of the book: Computer Organization and Design - Patterson and Hennessy) (also we have seen this in class).
The resource units in this figure include ALUs, AND gate, Control, Counter (PC), Memory (data and instruction), Muxes, and Register file. Control Unit generates seven signals viz, RegWrite, MemRead, ALUMux, MemWrite, ALUop, RegMux, and Branch.

Basic implementation of the MIPS instructions subset



10. [12] Consider the instruction AND rd. rs, rt.
    a. [7] Which of the signals from the Control Unit will be *asserted* and which will be *deasserted* to execute this instruction? (in case of ALUMux provide the actual operation to be selected as well).
    b. [3] Which of the resource units are used in executing this instruction?
    c. [2] Among the units used for this instruction, the output from some of them is not used subsequently. Which of the resource output are not used?

Solution:

a. [7] **Asserted**: RegWrite, RegMux (ALU), ALUOp (AND)

Deasserted: MemRead, ALUMux (Reg), MemWrite, and Branch

b. [3] All except data memory and branch Add unit
c. [2] Branch Add (No output from data memory)

11. [12] Consider the instruction SW rt, offset(rs).

    a. [7] Which of the signals from the Control Unit will be *asserted* and which will be *deasserted* to execute this instruction? (in case of ALUMux provide the actual operation to be selected as well).
    b. [3] Which of the resource units are used in executing this instruction?
    c. [2] Among the units used for this instruction, the output from some of them is not used subsequently. Which of the resource output are not used?

Solution:

a. [7] **Asserted**: ALUMux (imm), MemWrite, ALUop (ADD)
Deasserted: RegWrite, MemRead, Branch
Don't care: RegMux
b. [3] All except branch ADD unit and Write port of the registers
c. [2] Branch Add and Write port of the registers (all units produce outputs)

12. [4] Different execution units and blocks of digital logic have different latencies (time needed to do their work). In the above figure, there are seven (7) major blocks. **Critical path** is the path with longest latency. It determines minimum time needed to complete an instruction. Below is a list of latencies for a typical implementation: I-Mem: 200ps; Add: 70ps; Mux: 20ps; ALU: 90ps; Regs: 90ps; D-Mem: 250ps; Control:40ps
    a. What is the Critical path for an MIPS AND instruction?
    b. What is the Critical path for an MIPS load (lw) instruction?

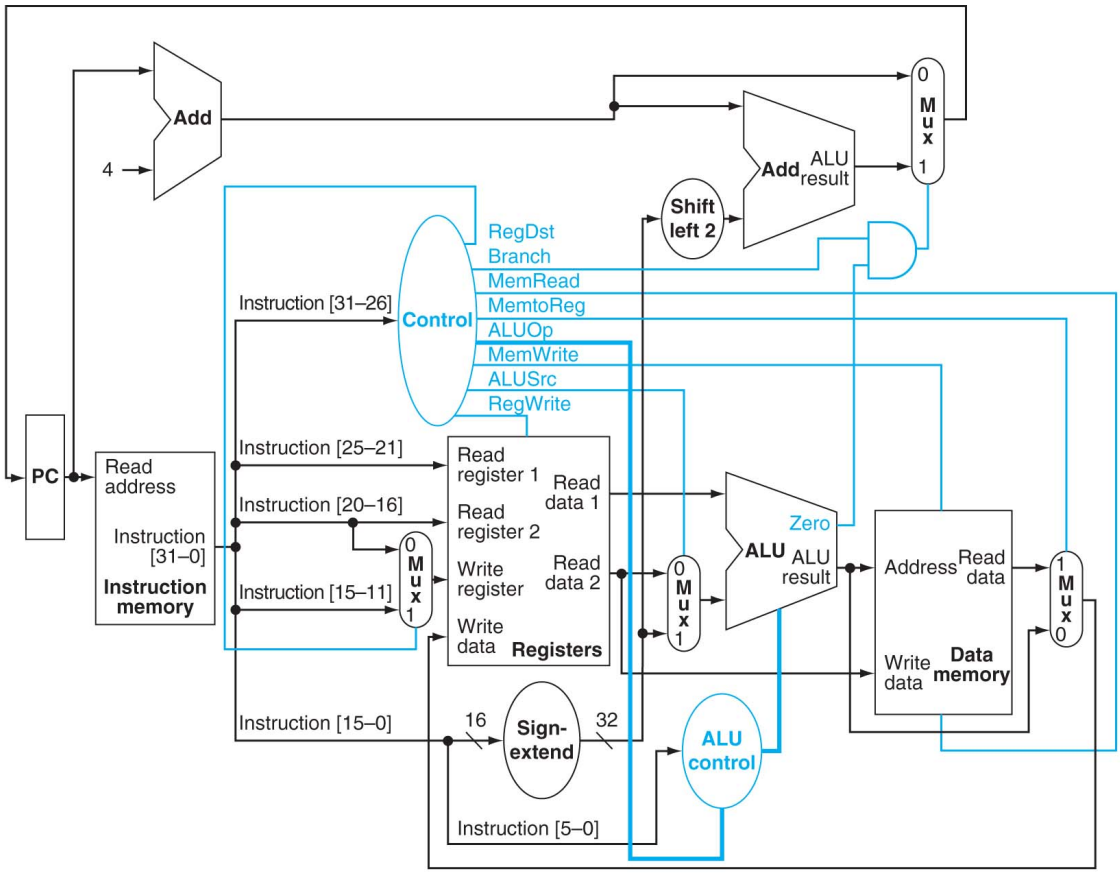
Solution:

a. [2] 200+90+20+90+20++20+90=530ps
b. [2] 200+90+20+90+250+20+20+90=780ps

Refer to the following figure (we have seen in class) below (figure 4.17 in the 4th edition revised printing of the book: Computer Organization and Design -

Patterson and Hennessy) of simple datapath with Control Unit for answering questions 13 and 14.

The input to the control unit is the 6-bit opcode field from the instruction. The outputs of the control unit consist of three 1-bit signals that are used to control multiplexors (RegDst, ALUSrc, and MemtoReg), three signals for controlling reads and writes in the register file and data memory (RegWrite, MemRead, and MemWrite), a 1-bit signal used in determining whether to possibly branch (Branch), and a 2-bit control signal for the ALU (ALUOp). An AND gate is used to combine the branch control signal and the Zero output from the ALU; the AND gate output controls the selection of the next PC. Notice that PCSrc is now a derived signal, rather than one coming directly from the control unit. Thus, we drop the signal name.



13. [8] Describe the effect that a single stuck-at-0 fault (i.e., regardless of what it should be, the signal is always 0) would have on the multiplexors in the single-cycle datapath in this figure. Which instructions, if any, would still work?

Consider each of the following faults separately: RegDst = 0, ALUSrc = 0, MemtoReg = 0, Zero = 0.

Solution:

NOTE: The solution provided here is those instructions that will NOT work, though I have asked for those instruction will still work. If the student has given answer about those instructions that will work, give mark

- [2] If RegDst =0, R Format will not work, because we specify the wrong register to write
- [2] If ALUSrc = 0, all I format (except branch) will not work, because we would omit the sign-extended 16 bits into the ALU
- [2] If MemtoReg =0, load (lw) will not work
- [2] If zero=0, branch instruction will never branch, even when it should

14. [8] Describe the effect that a single stuck-at-1 fault (i.e., regardless of what it should be, the signal is always 1) would have on the multiplexors in the single-cycle datapath in this figure. Which instructions, if any, would still work? Consider each of the following faults separately: RegDst = 1, ALUSrc = 1, MemtoReg = 1, Zero = 1.

Solution:

NOTE: The solution provided here is those instructions that will NOT work, though I have asked for those instruction will still work. If the student has given answer about those instructions that will work, give mark

- [2] If RegDst =1, loads will not work because of bad destinations
- [2] If ALUSrc = 1, all R format and branch will not work properly because we cannot get the 2nd register read into the ALU
- [2] If MemtoReg =1, all R format instructions will not write the correct data into the register file
- [2] If zero=1, all branches will be taken all the time, even if they should not be

PCSrc=0; MemRead=0; MemWrite=0;