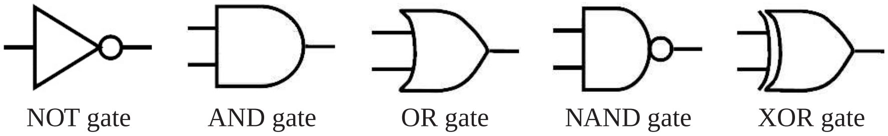
922 25300 Computer Organization and Assembly Languages **National Taiwan University**
Fall 2008 **Department of CSIE**

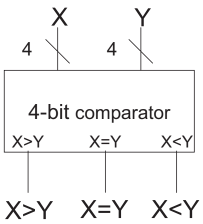
Homework 1

October 7, 2008 Due date: October 20, 2008

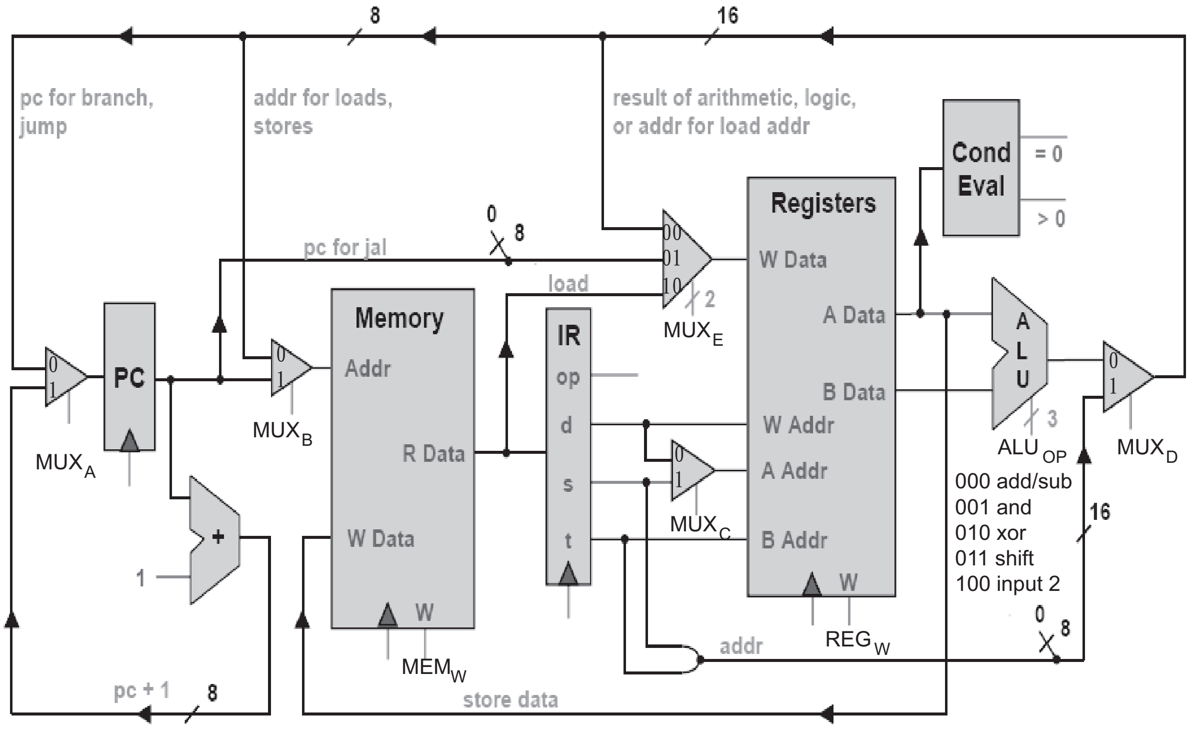
- *Signed integers are represented using two's complement representation.*
- *Please use the following notations for logic gates. Feel free to change the orientations of the gates and the positions of inputs and outputs. You are free to use other circuits. However, if they are not introduced in the class, you have to implement them before using them.*



1. (10%) What are the 8-bit binary representations of the following signed decimal integers? Convert them into hexadecimal.
   a. 37
   b. -18
2. (10%) Prove that (a) {NOT, AND} and (b) {NOR} are universal.
3. (10%) Let $Q = A + B(A + C) + AC$. Prove that (a) $Q = A + BC$. (b) Draw a circuit to implement Q.
4. (10%) (a) Create the truth table for the 3-input Boolean function, Ones, which returns the number of 1's in the input. For example, if the input $X_2X_1X_0 = 101$, then the output $Z_1Z_0 = 10$ as there are two 1's in the input. (b) Implement this function with logic gates AND, OR and NOT.
5. (10%) In the class, we show the design of the 4-bit left shifter. (a) Please create the truth table for the 4-bit right shifter (in the similar form we used for the left shifter). (b) Write down the corresponding logic expressions for each output.
6. (15%) As shown in the following diagram, design a 4-bit comparator which has two 4-bit unsigned integer inputs, $X_3X_2X_1X_0$ and $Y_3Y_2Y_1Y_0$, and a 3-bit output for the conditions of $X > Y$, $X = Y$ and $X < Y$, respectively. *Hint: design a 1-bit comparator first.*



7. (20%) Design a binary multiplier that multiplies two 3-bit unsigned integers, $X = X_2X_1X_0$ and $Y = Y_2Y_1Y_0$, and a 6-bit output $Z = Z_5Z_4 \ldots Z_0$ and $Z = X \times Y$, where X_0, Y_0 and Z_0 are LSBs. You may use the notation $X[n..m]$ to identify a portion of wires. For example, X[2..1] means the set of wires, X_2X_1.
8. (15%) Refer to the following figure for TOY architecture. Please specify the operations of MUX_A, MUX_B, MUX_C, MUX_D, MUX_E, REG_W, MEM_W and ALU_{OP} during the execution stage for the instructions "subtract", "load", and "jump register". For example, the answer for "jump and link" would be $MUX_A = 0, MUX_B = *, MUX_C = *, MUX_D = 1, MUX_E = 01, REG_W = 1, MEM_W = 0, ALU_{OP} = *$.


8
16
pc for branch, jump
addr for loads, stores
result of arithmetic, logic, or addr for load addr
Cond Eval
= 0
> 0
Registers
W Data
A Data
B Data
W Addr
A Addr
B Addr
W
REG$_W$
0
8
pc for jal
load
00
01
10
2
MUX$_E$
Memory
Addr
R Data
W Data
W
MEM$_W$
IR
op
d
s
t
PC
0
1
MUX$_A$
0
1
MUX$_B$
+
1
pc + 1
8
store data
addr
0
1
MUX$_C$
A L U
3
ALU $_{OP}$
000 add/sub
001 and
010 xor
011 shift
100 input 2
0
1
MUX$_D$
16
0
8