**Computer Architecture**

**Fall, 2019**

**Week 6**

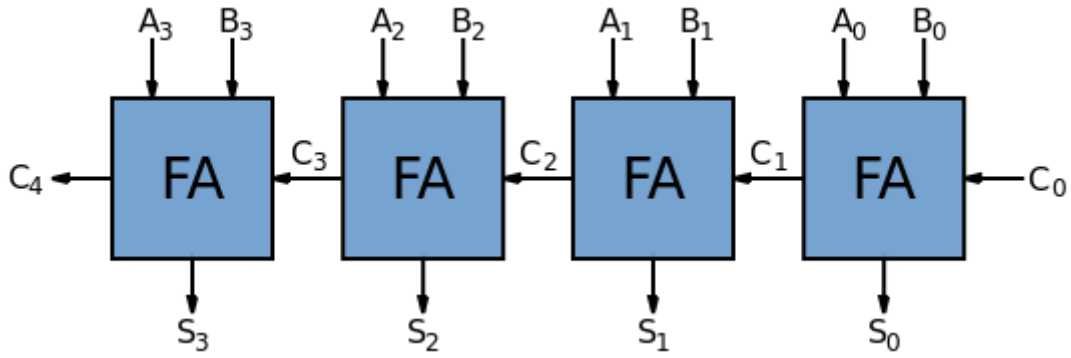
**2019.10.14**

組別：_____ 簽名：_____

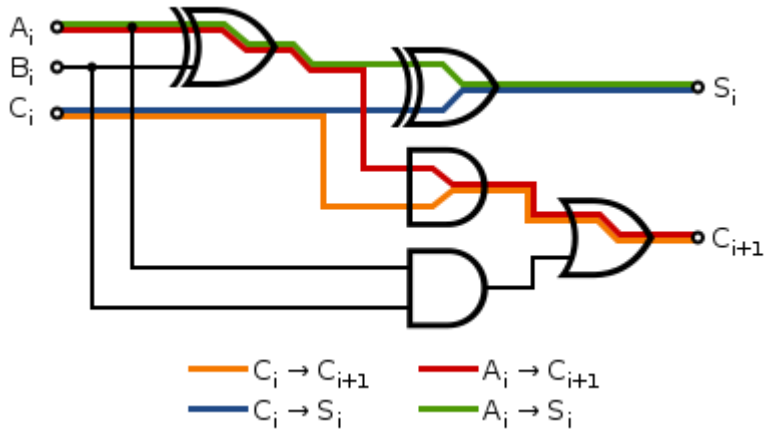
[group3] (對抗賽)

1. The propagation delay of an XOR gate is twice that of an AND/OR gate. The propagation delay of an AND/OR gate is 1.2 microseconds. (AND and OR gates have a nominal delay of 1 gate-delay, and XOR gates have a delay of 2, because they are really made up of a combination of ANDs and ORs.)

A 4-bit ripple-carry binary adder is implemented by using full adders. What is the total propagation time of this 4-bit binary adder in microseconds (in a worst case)?




Hint:




- 4 gate-delays from generating the first carry signal ($A_0/B_0 \rightarrow C_1$).
- 2 gate-delays per intermediate stage ($C_i \rightarrow C_{i+1}$).
- 2 gate-delays at the last stage to produce both the sum and carry-out outputs ($C_{n-1} \rightarrow C_n$ and $S_{n-1}$).

**(Ans)** 12 microseconds

Propagation Delay by n bit full adder is (2n + 2) from 4+2(n-2)+2.

**($A_0/B_0 \rightarrow C_1$).+ ($C_1 \rightarrow C_3$).+ ($C_3 \rightarrow C_4$ and $S_3$)**

The worst case propagation delay is (2*4+2)*1.2=12ms

[group4] (對抗賽)

2. 我們以 4 個 4bit Carry Lookahead Adder 當作基本單位，並串接起來形成 16bit Adder。計算下列 16bits 數字相加產生的$g_i, p_i, P_i, G_i$，並用$P_i, G_i$計算 Carryout15 的值。

   Hint: $G_i = g_{4i+3} + (p_{4i+3} * g_{4i+2}) + (p_{4i+3} * p_{4i+2} * g_{4i+1}) + (p_{4i+3} * p_{4i+2} * p_{4i+1} * g_{4i})$,

   $$P_i = p_{4i+3} * p_{4i+2} * p_{4i+1} * p_{4i}$$

a. $1000\ 0111\ 0101\ 0011_2$

b. $0011\ 1001\ 1101\ 0110_2$

Ans : gi : $0000\ 0001\ 0101\ 0010_2$

pi : $1011\ 1110\ 1000\ 0101_2$

P3 = 1*0*1*1 = 0

P2 = 1*1*1*0 = 0

P1 = 1*0*0*0 = 0

P0 = 0*1*0*1 = 0

G0 = g3+(p3*g2)+(p3*p2*g1)+(p3*p2*p1*g0)

= 0+(0*0)+(0*1*1)+(0*1*0*0)

= 0

G1 = g7+(p7*g6)+(p7*p6*g5)+(p7*p6*p5*g4)

= 0+(1*1)+(1*0*0)+(1*0*0*1)

= 1

G2 = g11+(p11*g10)+(p11*p10*g9)+(p11*p10*p9*g8)

= 0+(1*0)+(1*1*0)+(1*1*1*1)

= 1

G3 = g15+(p15*g14)+(p15*p14*g13)+(p15*p14*p13*g12)

= 0+(1*0)+(1*0*0)+(1*0*1*0)

= 0

c15 = C4

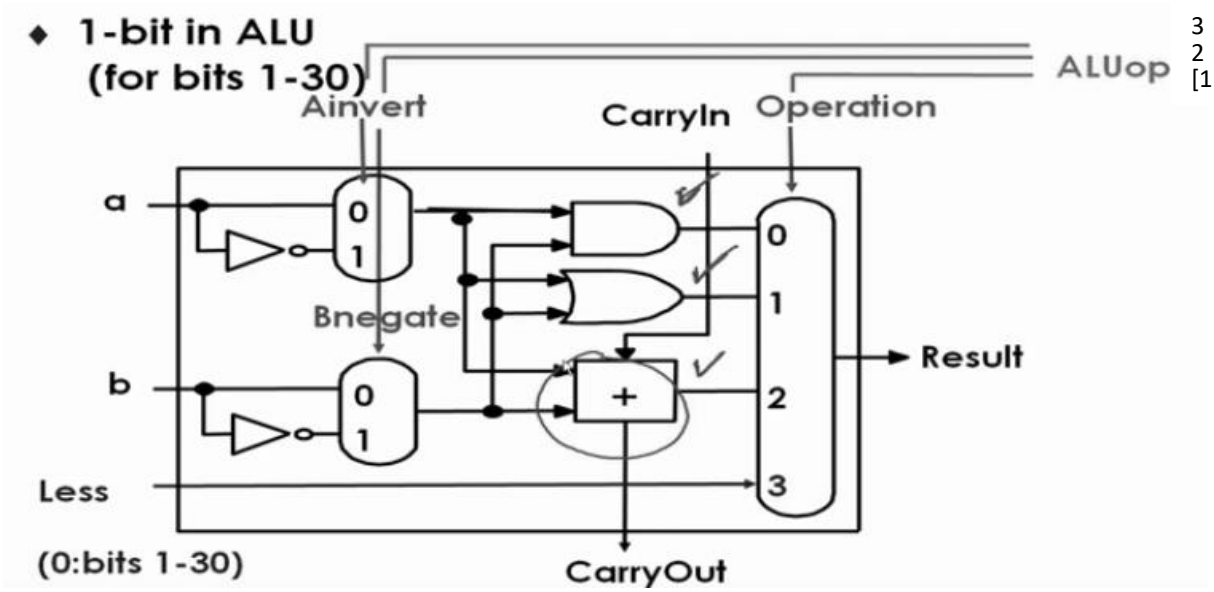
= G3+(P3*G2)+(P3*P2*G1)+(P3*P2*P1*G0)

=0+(0*1)+(0*0*1)+(0*0*0*1)

=0

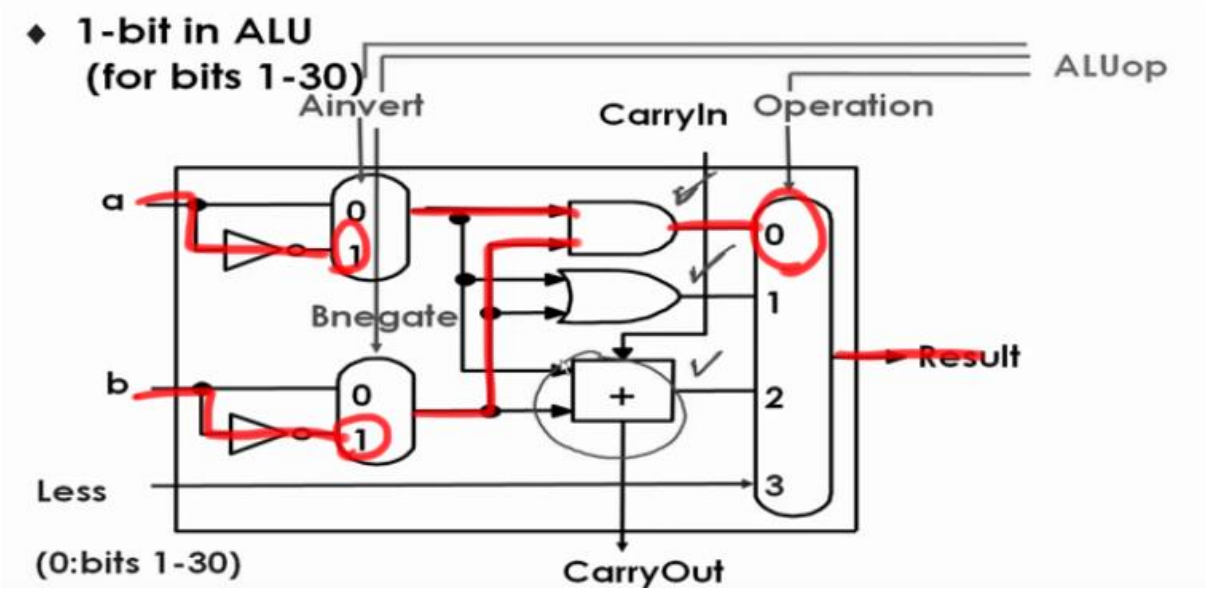
[group7] (對抗賽)

3.




請畫出當輸入指令為 1100 時的路線會怎麼走，並將 result 轉換成 Boolean 運算的表示。(ex: result = A add B)

A:




result = !A and !B = A nor B

[group5]

4. When performing arithmetic addition and subtraction, overflow might occur. Fill in the blanks in the following table of overflow conditions for addition and subtraction.

| Operation | Operand A | Operand B | Result indicating overflow |
|---|---|---|---|
| A+B | ≥0 | ≥0 | (1) |
| A+B | <0 | <0 | (2) |
| A-B | ≥0 | <0 | (3) |
| A-B | <0 | ≥0 | (4) |

Ans: (1) <0 (2) ≥0 (3) <0 (4) ≥0

[group8]

5. 下列有關 alu control (aluop) 的敘述何者正確? (參考 MOOCs: lec3.ppt, p. 19)

A. ALUOP 是一個 4 bits 且用來控制 ALU 的訊號
B. 前兩個 bit 控制兩個輸入是否做 not，若為 0 則 invert。
C. 後兩個 bit 控制 alu 要做的事情，如 and or add sub 等等
D. 輸出包含 result、zero 和 overflow

ans: A C D

[group11] (對抗賽)

6. When dealing with overflow in high level languages, some choose to ignore it, while some require raising an exception. What kind of MIPS instructions are used in the above two situations?

Ans:
Ignore overflow: addu, addui, subu instructions
Raise exception: add, addi, sub instructions

[group6] (對抗賽)

7. Consider a two-bit adder with its two operands Ai and Bi where i = 0, 1(0 is the least significant bit) and the carry-in bit C0 and carry-out bit C2. Derive the Boolean equation for the carry-out bit C2 using only Ai, Bi (where i = 0, 1) and C0.

Ans

C2 = A1A0B0 + A1A0C0 + A1B0C0 + B1A0B0 + B1A0C0 + B1B0C0 +A1B1

[group8] (對抗賽)

8. Let the decimal numbers A = 54, B = -77, give their 8-bit 2's complement representation:

(a) Compute A + B in 8-bit 2's complement.

(b) Compute A – B in 8-bit 2's complement, is it overflow? Why or why not?

Ans

(a) A+B = 54-77 =-23 在 127 ～ -128 之間所以沒有 overflow

00110110 + 10110011 = 11101001

(b) Overflow,因為 54+77 = 131 > 127 超過 8bits 的 2’s complement 可計算的範圍

A+(-B)

00110110 + 01001101 -> overflow