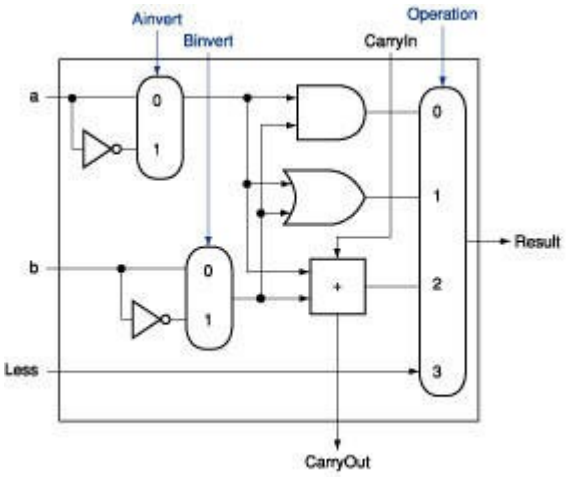
Computer Organization

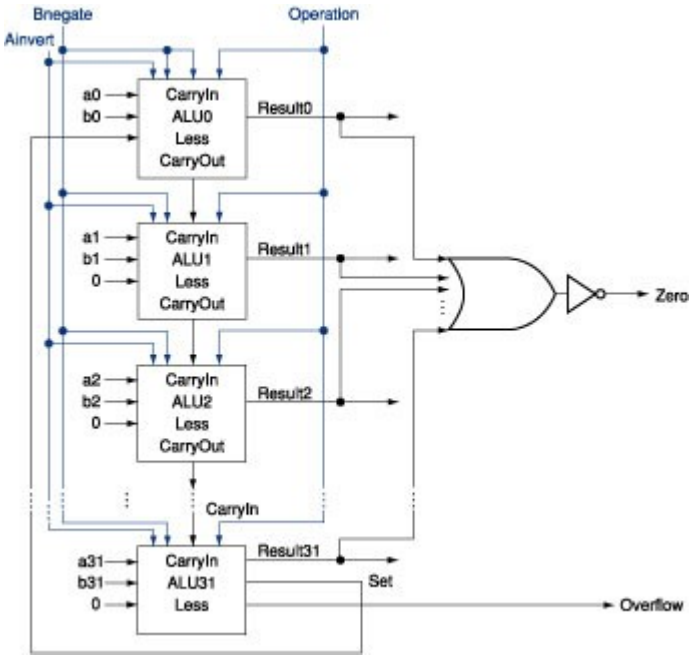
0616058_0617052

Architecture diagram:

1-bit ALU



32-bit ALU



Detailed description of the implementation:

ALU action	Name	ALU control input
And	And	0000
Or	Or	0001
Add	Addition	0010
Sub	Subtract	0110
Nor	Nor	1100
Nand	Nand	1101
Slt	Set less than	0111

Basic Control:

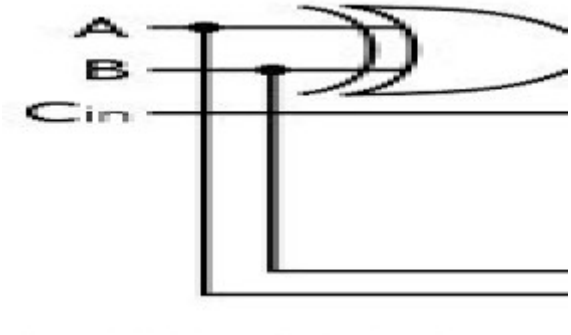
1. And,Or,Nand,Nor Implementation :

   做出 And 和 Or 後, 可以透過控制 invert bit 來做出 Nand 和 Nor

2. Add,Sub Implementation :

   做出 adder 後, 同理可以透過控制 invert bit 來完成 Substrator

   Adder:



3. Slt implementation :

   同號時考慮減法出來的結果, MSB = 1 則 lt = 1,

   異號時只要 A MSB = 1 則 lt = 1

ZCV control signal :

1. Zero :

   檢查(Result 是否 == 0), 是的話 Zero =1 ;反之 Zero =0

2. Carryout :

   在每一個 adder 或 substractor 算出 Cout 後, 下一個 adder 或 Substractor 才會有 Cin, 因此需要等待一個 bit 一個 bit 做完

3. Overflow :

   考慮加法會 overflow 的情況, 當兩數同號的時候有可能會出現 overflow

   兩數為正：MSB = 1, 第 32 bit 有 Cin 沒有 Cout

   兩數為負：MSB = 0, 第 32 bit 有 Cout 沒有 Cin

   其他：第 32 bit 有 Cin 就會有 Cout

   結論：當第 32 bit Cin 與 Cout 沒有同時出現就是 overflow

Problems encountered and solution:

Problems : Zero 和 Carryout 在 Slt 本來是錯誤的

Solution : 後來我們將 result 丟到新的 wire(result_n) 就成功了

Lesson learnt (if any):

ALU 的功能以及基本構造

Reference :

https://www.cs.drexel.edu/~jjohnson/2012-13/fall/cs281_fa12/labs/vhdl_32-bit_ALU.html

https://github.com/chengchingwen/slt_alu

https://zh.wikipedia.org/wiki/%E5%8A%A0%E6%B3%95%E5%99%A8

http://programmermagazine.github.io/201310/htm/article4.html

https://github.com/xatier/COlab2/blob/master/ALU.v

http://web.cse.ohio-state.edu/~teodorescu.1/download/teaching/cse675.au08/

Cse675.02.F.ALUDesign_part1.pdf