

COMPUTER SYSTEMS ORGANIZATION

ALU Design -- Spring 2010 -- IIIT-H -- Suresh Purini

MIPS Architecture

- ❑ Introduced in 1981 (by John Hennessy & Co. in Stanford University).
- ❑ Initially a 32-bit processor. 64-bit versions available from 1991 onwards.
- ❑ ISA follows RISC philosophy
- ❑ 32-bit (4 GB) Address Space
- ❑ 31 General Purpose Registers (No condition code register, like CPSR in ARM)
- ❑ Load-Store Architecture
- ❑ Only few addressing modes
 - ❑ Register, immediate, register+offset, pc-relative addressing in branch instructions.

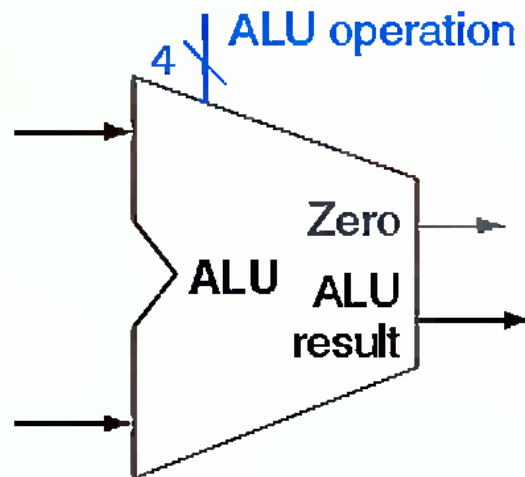
MIPS ISA

Our Focus: How to implement the following 8 instructions of MIPS ?

Instructions	Functionality
lw r_1 , 24(r_2)	$r_1 = \text{mem}[r_2 + 24]$
sw r_1 , 24(r_2)	$\text{mem}[r_2 + 24] = r_1$
add r_1, r_2, r_3	$r_1 = r_2 + r_3$ (signed)
sub r_1, r_2, r_3	$r_1 = r_2 - r_3$ (signed)
and r_1, r_2, r_3	$r_1 = r_2 \& r_3$
or r_1, r_2, r_3	$r_1 = r_2 \mid r_3$
slt r_1, r_2, r_3	if $r_2 < r_3$ then $r_1 = 1$ else $r_1 = 0$ (signed)
beq $r_1, r_2, 25$	if $r_1 == r_2$ then $\text{pc} = \text{pc} + 4 + 100$

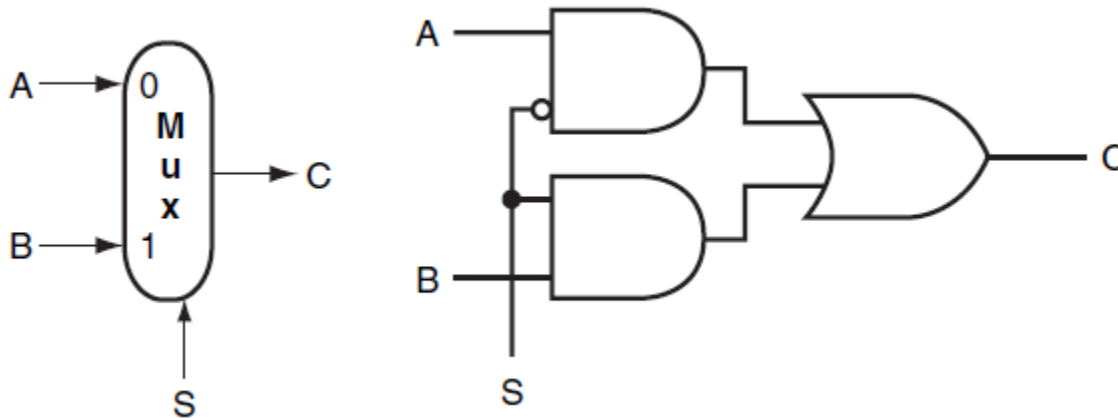
Designing a 32-bit ALU

Idea: Build the 32-bit ALU using 32 1-bit ALU



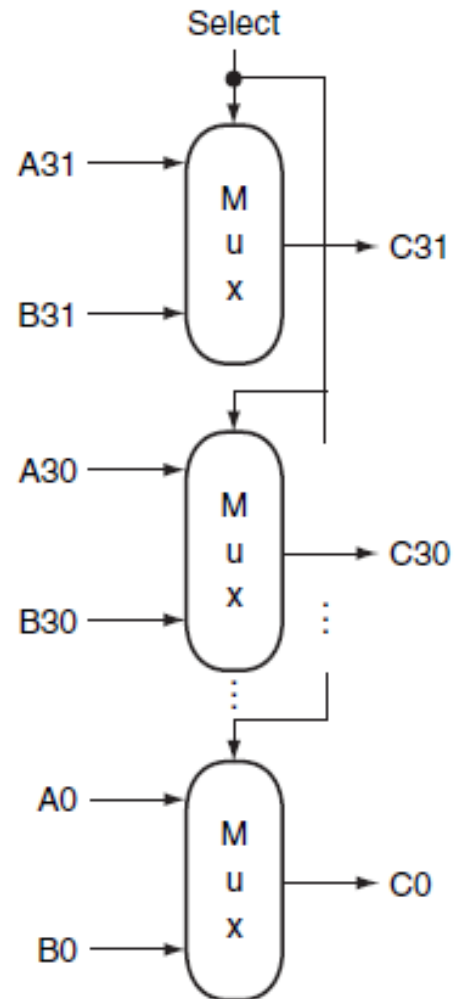
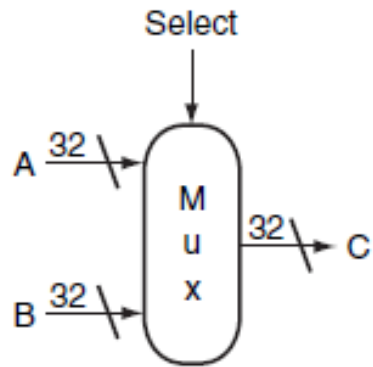
Review: Multiplexor

- A 2-to-1 multiplexor.



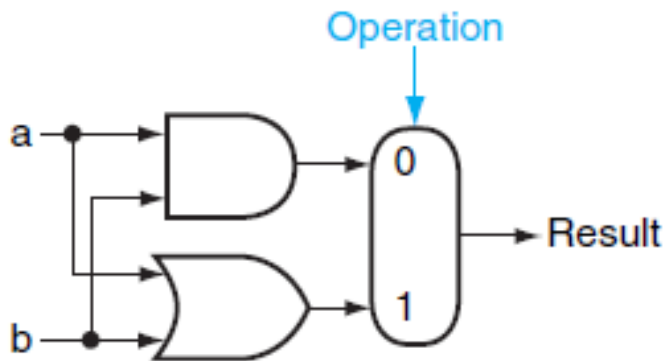
- How to use a multiplexer to build an ALU?

A 32-bit wide 2-to-1 Multiplexor

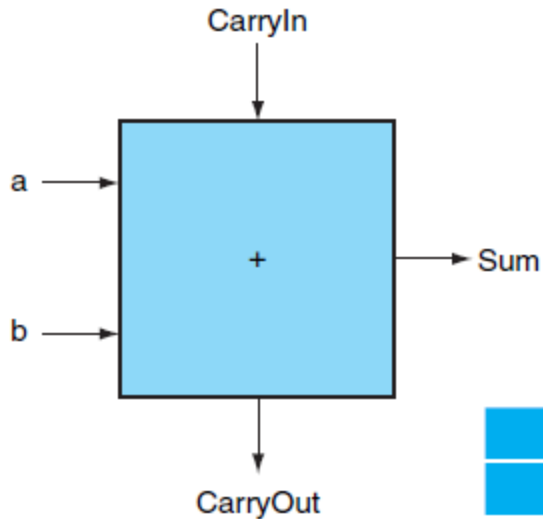


1-bit ALU

- A 1-bit ALU that can perform AND/OR operation.



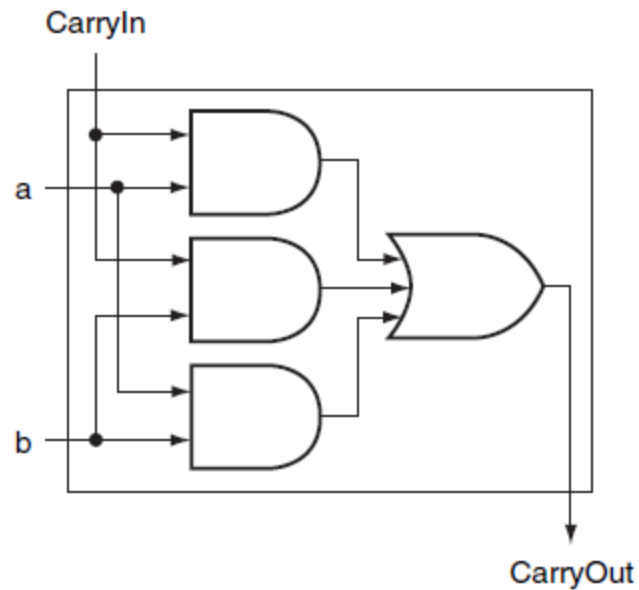
1-bit Addition Operation



Inputs			Outputs		Comments
a	b	CarryIn	CarryOut	Sum	
0	0	0	0	0	$0 + 0 + 0 = 00_{\text{two}}$
0	0	1	0	1	$0 + 0 + 1 = 01_{\text{two}}$
0	1	0	0	1	$0 + 1 + 0 = 01_{\text{two}}$
0	1	1	1	0	$0 + 1 + 1 = 10_{\text{two}}$
1	0	0	0	1	$1 + 0 + 0 = 01_{\text{two}}$
1	0	1	1	0	$1 + 0 + 1 = 10_{\text{two}}$
1	1	0	1	0	$1 + 1 + 0 = 10_{\text{two}}$
1	1	1	1	1	$1 + 1 + 1 = 11_{\text{two}}$

1-bit Addition Operation

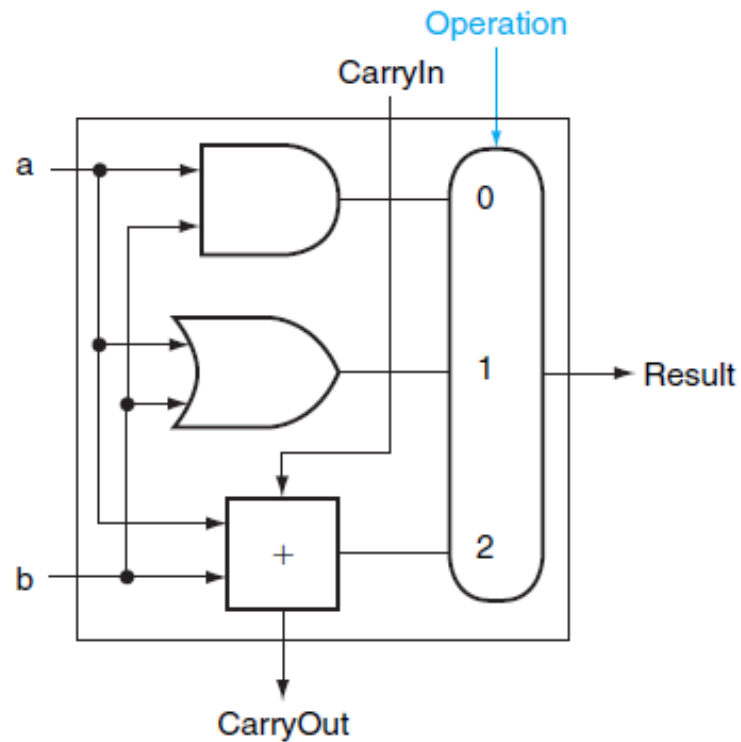
$$\text{CarryOut} = (b \cdot \text{CarryIn}) + (a \cdot \text{CarryIn}) + (a \cdot b)$$



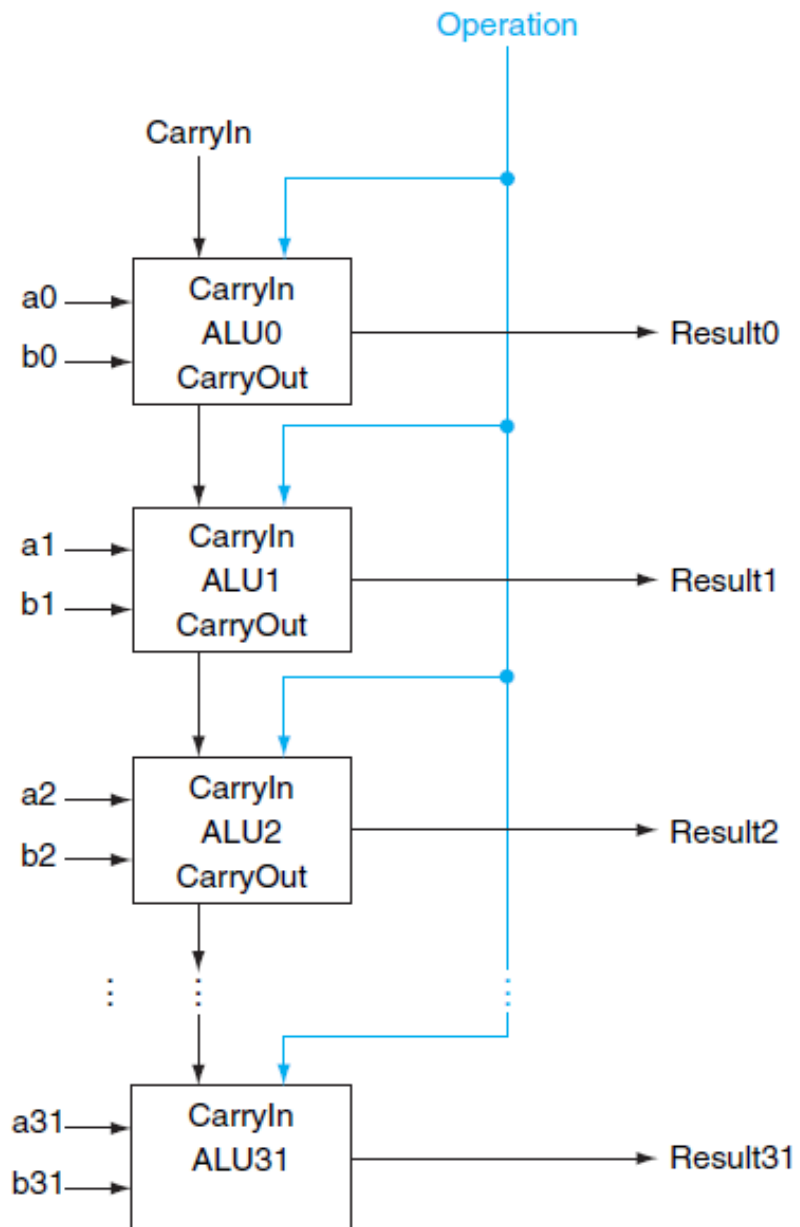
$$\text{Sum} = (a \cdot \bar{b} \cdot \overline{\text{CarryIn}}) + (\bar{a} \cdot b \cdot \overline{\text{CarryIn}}) + (\bar{a} \cdot \bar{b} \cdot \text{CarryIn}) + (a \cdot b \cdot \text{CarryIn})$$

1-bit ALU

- A 1-bit ALU that can perform AND, OR and Addition.

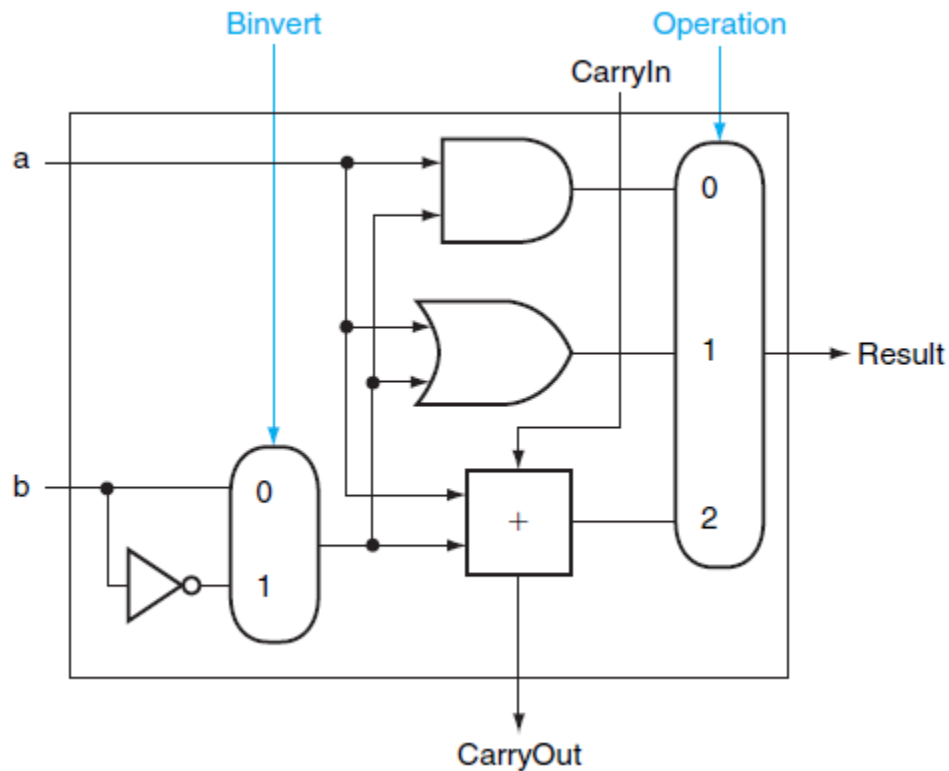


32-bit ALU



1-bit ALU

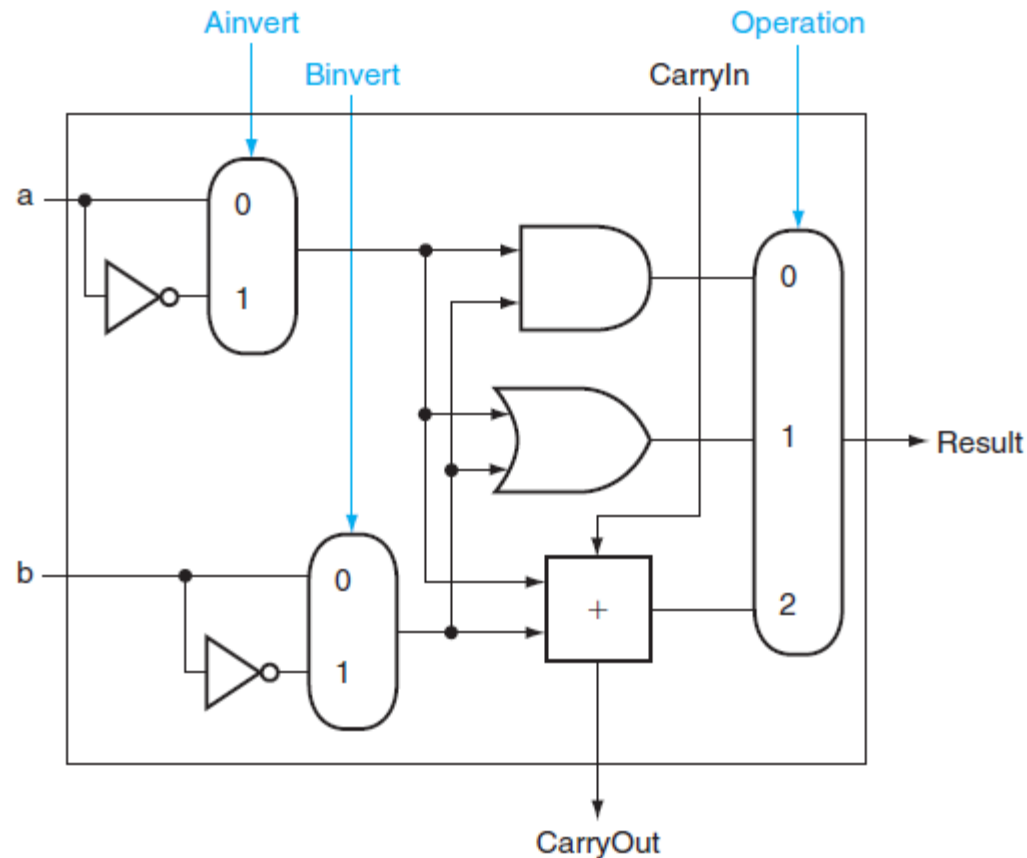
- A 1-bit ALU that can perform AND, OR, ADD (a , b or a , b')



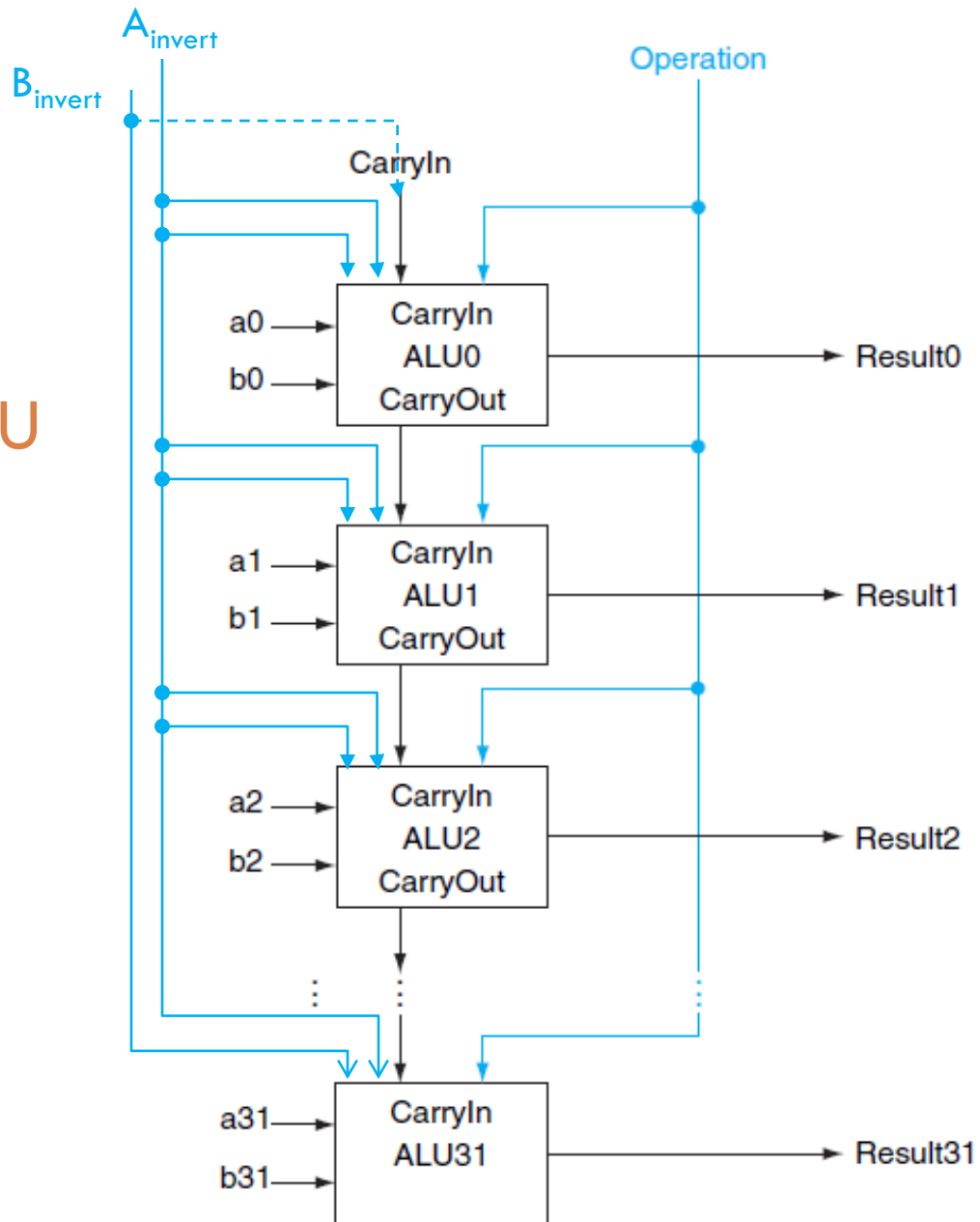
Can you think of how to build 32-bit ALU that can do subtraction also using 32 of these 1-bit ALUs?

1-bit ALU

- A 1-bit that can perform AND, OR, NOR and ADD

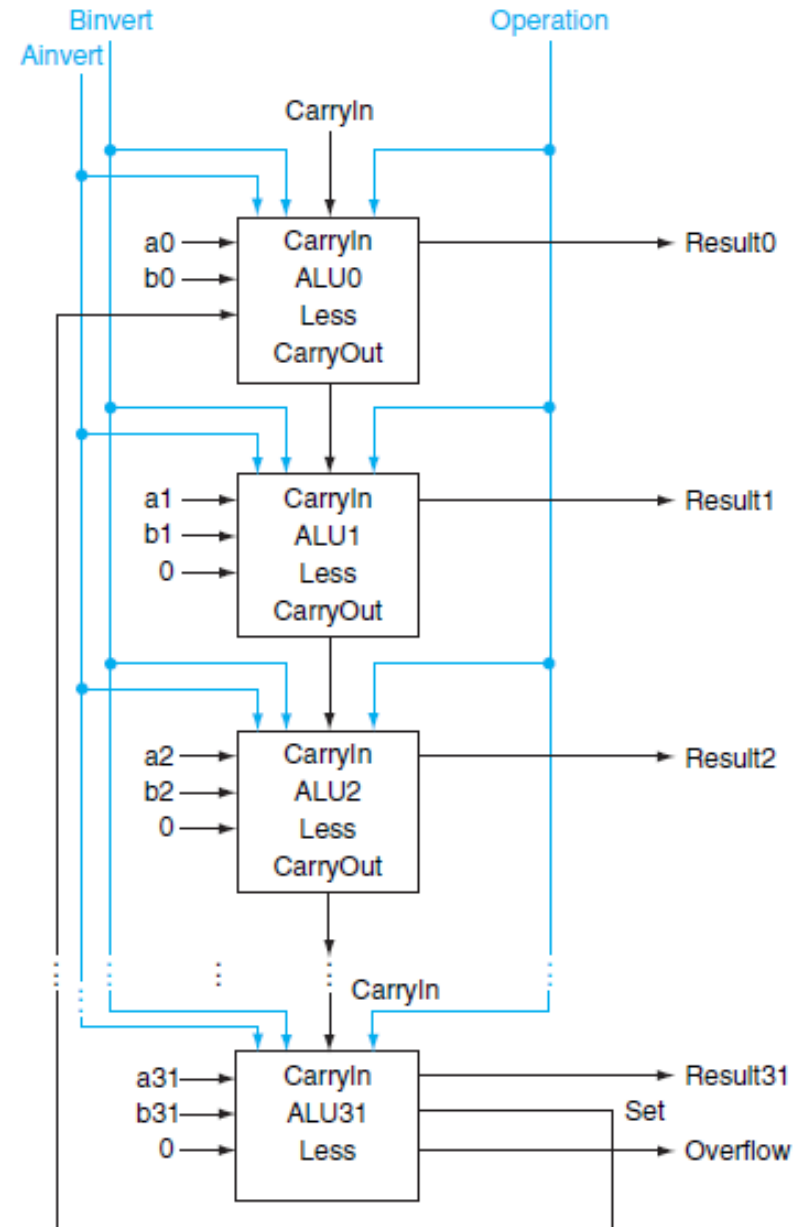


32-bit ALU



Does this ALU takes care of the **slt** instruction?

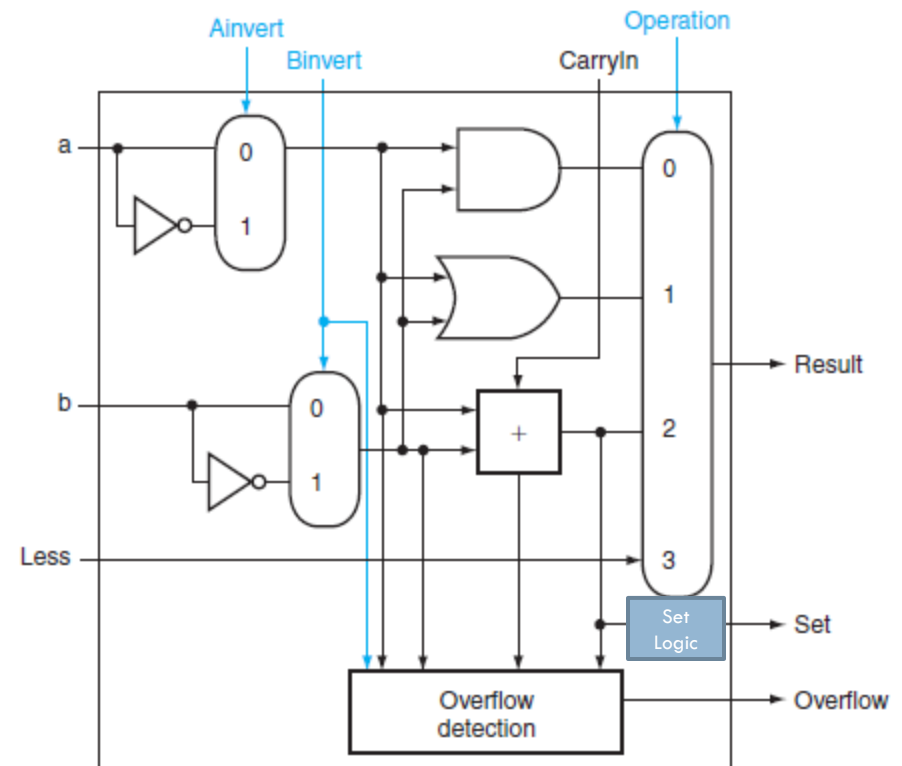
a_{31}	b_{31}	Set
0	0	MSB
0	1	0
1	0	1
1	1	MSB



ALU-31

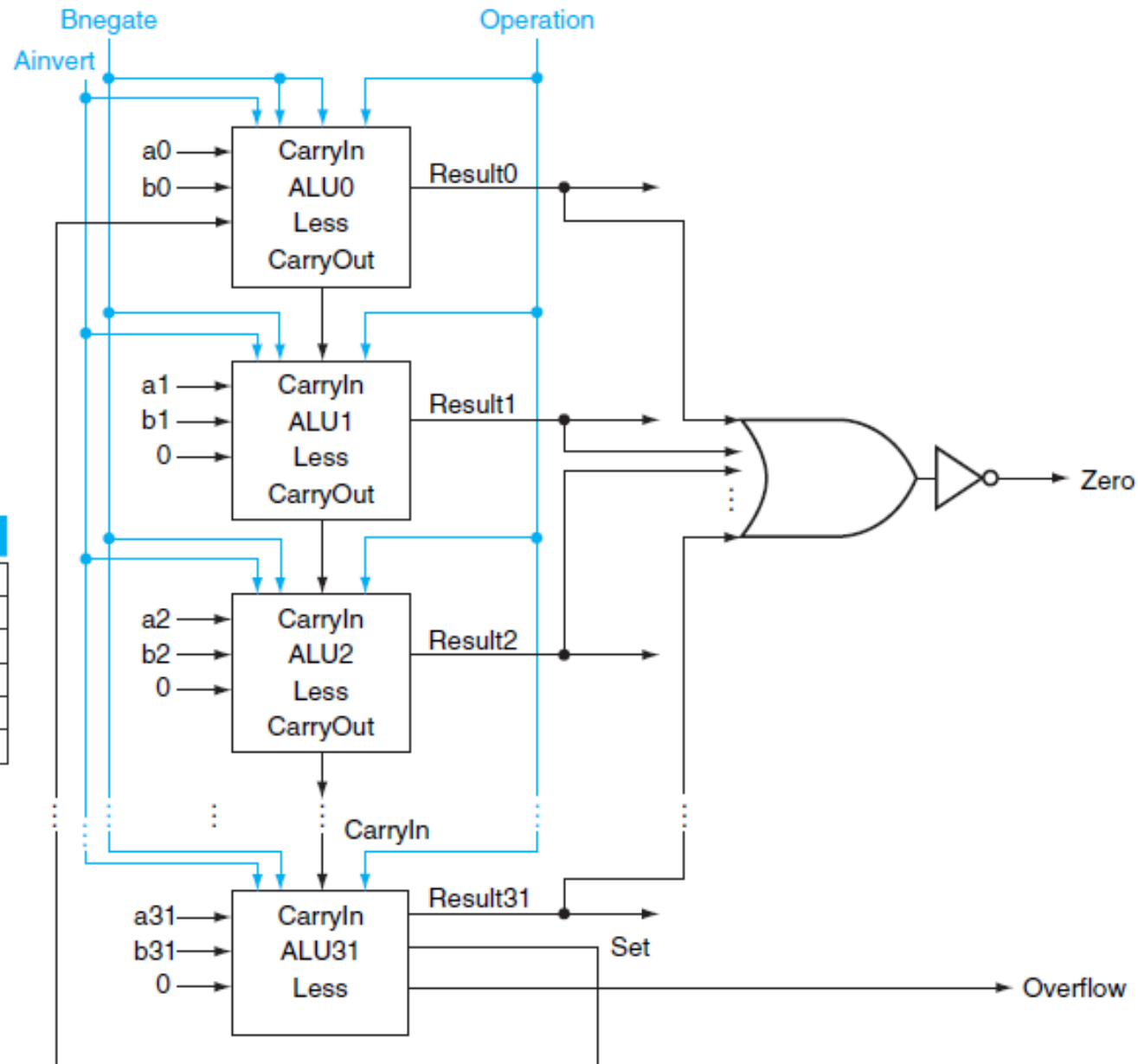
- Can you come up with overflow detection logic for signed numbers?

a_{31}	b_{31}	Set
0	0	MSB
0	1	0
1	0	1
1	1	MSB



32-bit ALU with Zero Flag Logic

ALU control lines	Function
0000	AND
0001	OR
0010	add
0110	subtract
0111	set on less than
1100	NOR



Final Block Diagram of 32-bit ALU

