1-bit ALU

Inputs			Outputs	
a	b	C_{in}	C_{out}	Sum
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

K-map simplification for carry and sum

For Carry (C_{out})

A \ BC_{in}	00	01	11	10
0	0	0	1	0
1	0	1	1	1

$C_{out} = AB + A C_{in} + B C_{in}$

For Sum

A \ BC_{in}	00	01	11	10
0	0	1	0	1
1	1	0	1	0

$Sum = \bar{A}\bar{B}C_{in} + \bar{A}B\bar{C}_{in} + A\bar{B}\bar{C}_{in} + ABC_{in}$

$C_{out} = ab + a c_{in} + b c_{in}$

$Sum = a\bar{b}\bar{c}_{in} + \bar{a}b\bar{c}_{in} + \bar{a}\bar{b}c_{in} + abc_{in}$

Sum = a xor b xor c

Slide 6

One bit ALU that perform AND, OR, addition.

Slide 7

$a - b = a + (-b)$

To generate ⇒ 2's complement

$a - b = a + (-b) = a + (\bar{b} + 1)$

invert ↑

LSB has a carryin signal (even though it is not necessary for addition)

If we set Carryin to 1

→ adder will calculate $a+b+1$

If b invert selected ⇒ $a+\overline{b}+1$

↓

$(a-b)$

Slide 8

Add NOR in hardware

$\overline{(a+b)} = \overline{a}\,\overline{b}$

($\overline{a}$ should be added)

Slide 9

slt $s1, $s2, $s3

{ If rs < rt produce 1 = 000....01

{ otherwise produce 0 = 000...00

$a \overset{?}{<} b \Rightarrow (a-b) < b-b$

$(a-b) \overset{?}{<} 0$

Connect sign bit from the adder output to the least significant bit to get SLT.

Result output of MSB ALU is not the output of adder
↳ it is input value "less"

1-bit ALU MSB → an extra output bit (adder output)

Observation

Every time we want to ALU to Subtract
→ Cin & Binvert set to 1.

add/and/or → both 0

Combine Cin & Binvert → Bnegate to a sngle line

Slide 12

Equality for conditional branches (if equal or not equal)

$a \overset{?}{=} b \Rightarrow (a-b) \overset{?}{=} 0$

1. Compute Result = a − b
2. OR all outputs & send that signal through an inverter

$$Zero = \overline{(Result_{31} + Result_{30} + \ldots + Result_0)}$$

equal to 1 when result is zero. (if two regs equal)