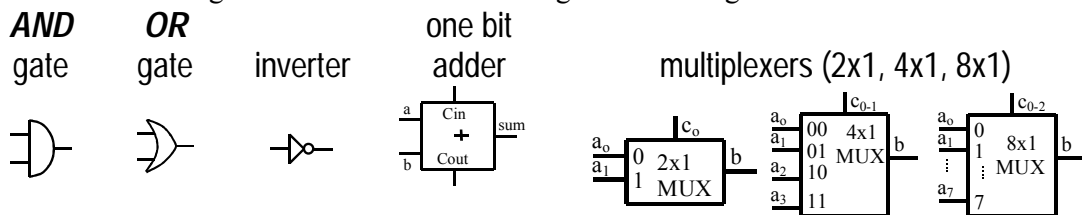


## Problems and Solutions

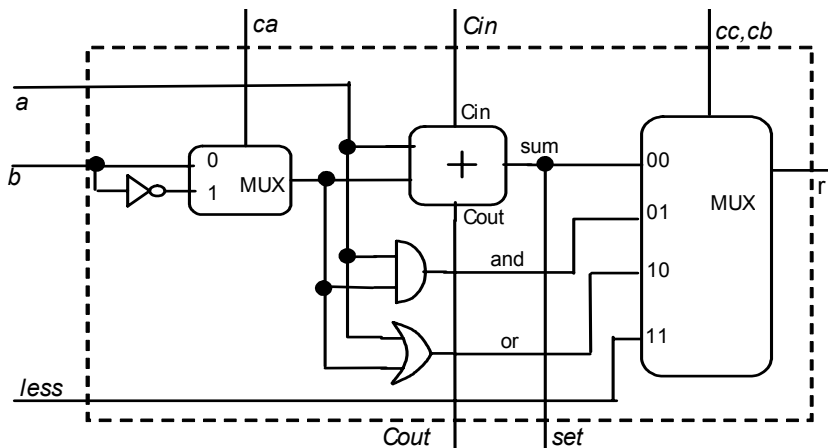
**Q1)** Consider the following control signals and the corresponding ALU operation of a 1-bit ALU with the operand inputs  $a$  and  $b$ , outputs  $C_{out}$  and  $r$ , and other required inputs/outputs such as *less* and *set*.

Control Signals			ALU operation
$cc$	$cb$	$ca$	
0	0	0	$(C_{out}, r) = sum(a, b, C_{in})$ (for add)
0	1	0	$r = AND(a, b)$
1	0	0	$r = OR(a, b)$
0	0	1	$(C_{out}, r) = sum(a, b', C_{in})$ (for sub)
1	1	1	$r = less$ (for slt): if $A - B < 0$ , then $R = 1$ ,
all other cases			don't care

- a) i) Which of the control signals from  $ca$ ,  $cb$ , or  $cc$  is most suitable for  $b_{invert}$  signal of this 1-bit ALU?  
 ii) Draw a circuit diagram for this 1-bit ALU using the following circuit elements.



<< solution



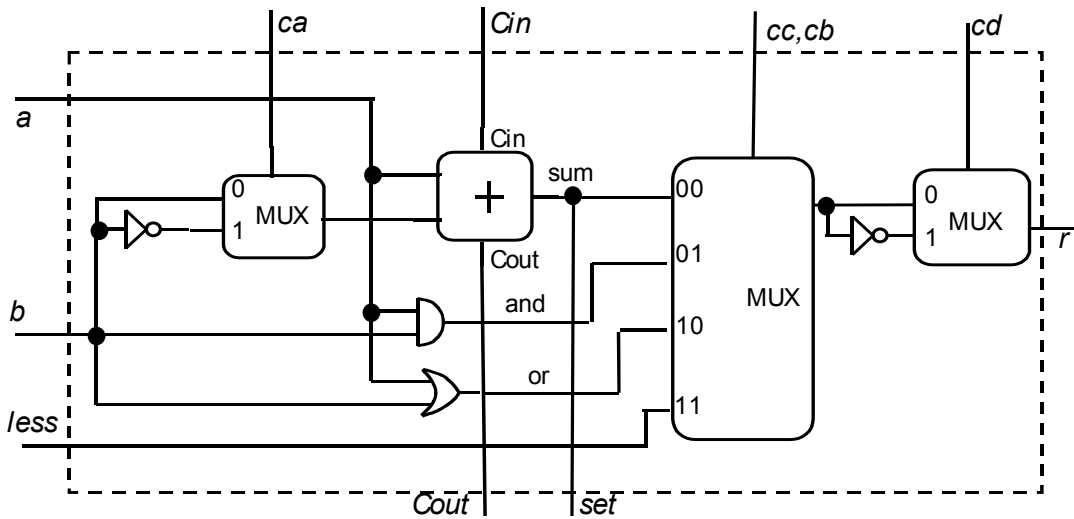
- i) only  $ca$  is suitable for  $b_{invert}$ , because only  $ca = 0$  for *add*, and  $ca=1$  for both *sub* and *less*.  
 ii) *AND* and *OR* gate inputs can also be connected to the inputs  $a$  and  $b$ .  
 >>

**b)** Extend your 1-bit ALU to perform the following operations with an additional control signal  $cd$  (sge: set if greater or equal).

Try to use as smaller multiplexer units as possible (i.e., one 4x1 plus one 2x1 multiplexer is twice faster and cheaper than one 8x1 multiplexer)

Control Signals				ALU operation
<i>cd</i>	<i>cc</i>	<i>cb</i>	<i>ca</i>	<i>r</i>
0	0	0	0	$(C_{out}, r) = sum(a, b, C_{in})$ (for add)
0	0	1	0	$r = AND(a, b)$
0	1	0	0	$r = OR(a, b)$
0	0	0	1	$(C_{out}, r) = sum(a, b', C_{in})$ (for sub)
0	1	1	1	$r = less$ (for slt): if $A - B < 0$ , then $R = 1$ ,
1	0	1	0	$r = NAND(a, b)$
1	1	0	0	$r = NOR(a, b)$
1	1	1	1	$\overline{less}$ (for sge): if $A - B \geq 0$ , then $R = 1$
all other combinations				don't care

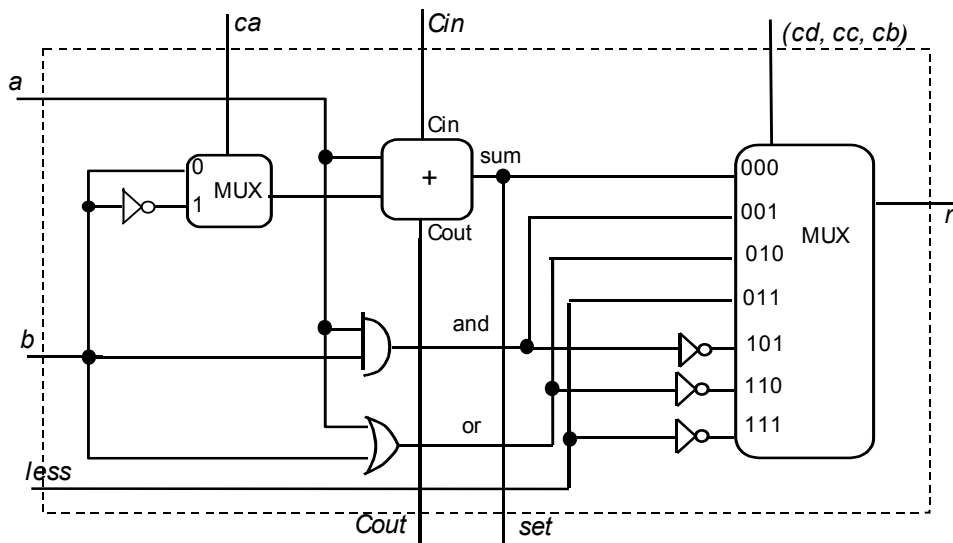
<<

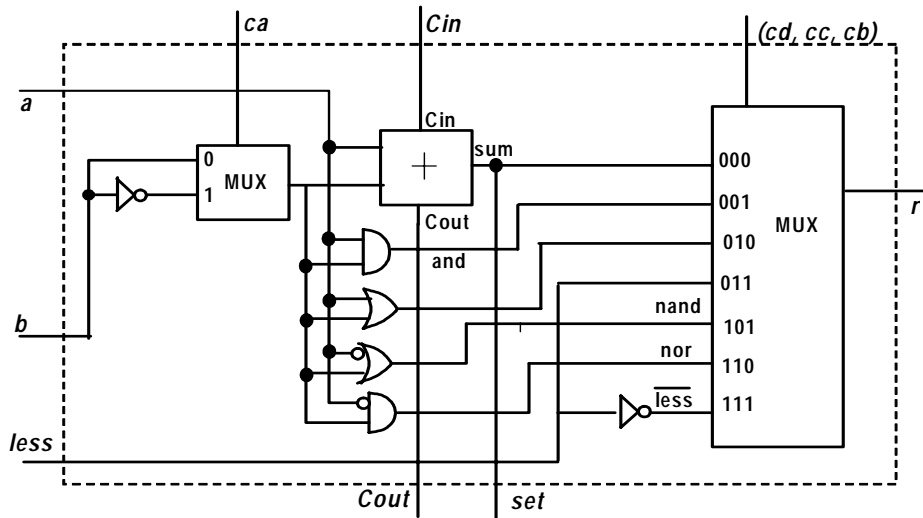


Use an inverter at the output of the MUX to satisfy **NAND** and **NOR** functions with the codes **cc,cb** = 01 and 10.

But, do not invert *b* of **AND** and **OR**. Connect them directly *b*.

Other possible solutions:





The other solutions require more gates and larger multiplexers.>>

c) Draw a 3-bit ALU that can execute *add, sub, and, or, nand, nor, slt* and *sge* instructions, using 1-bit ALU units, with the required circuit blocks in the most-significant-bit unit to generate an overflow signal.

<< solution

