Gath Aher
CompArc HW #5

① Feedback Survey ☑

② Combinational Logic Practice – Implement SLT (<) 32-bit input
• assume inputs are signed

A \ B	-2	-1	0	1	2
-2	F	T	T	T	T
-1	F	F	T	T	T
0	F	F	F	T	T
1	F	F	F	F	T
2	F	F	F	F	F

Is A < B?

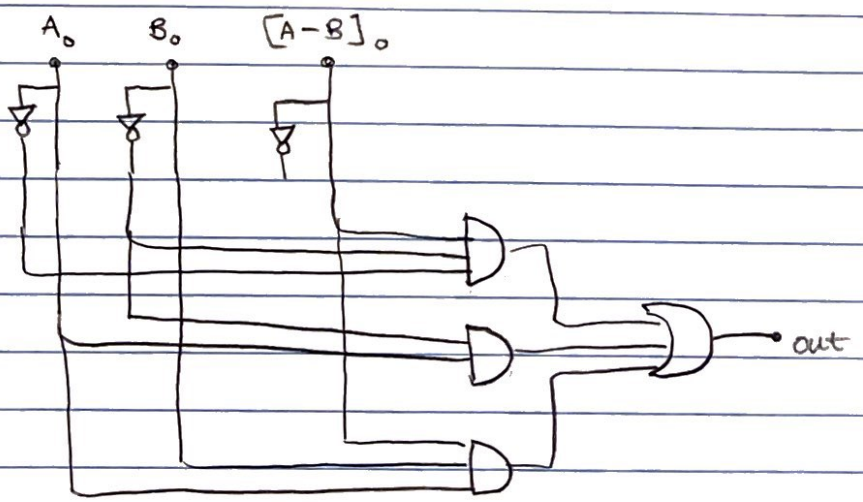
-1 -1 : A - B
(-1) + (+2) = 1 (F)
OR
(-2) + (+1) = -1 (T)

-1 1 (T)

1 -1 (F)

1 1 : A - B
1 - 2 = -1 (T)
OR
2 - 1 = 1 (F)

A_0	B_0	$[A-B]_{[1]}$	Out
0	0	0	0
		1	1
0	1	X	0
1	0	X	1
1	1	0	0
		1	1



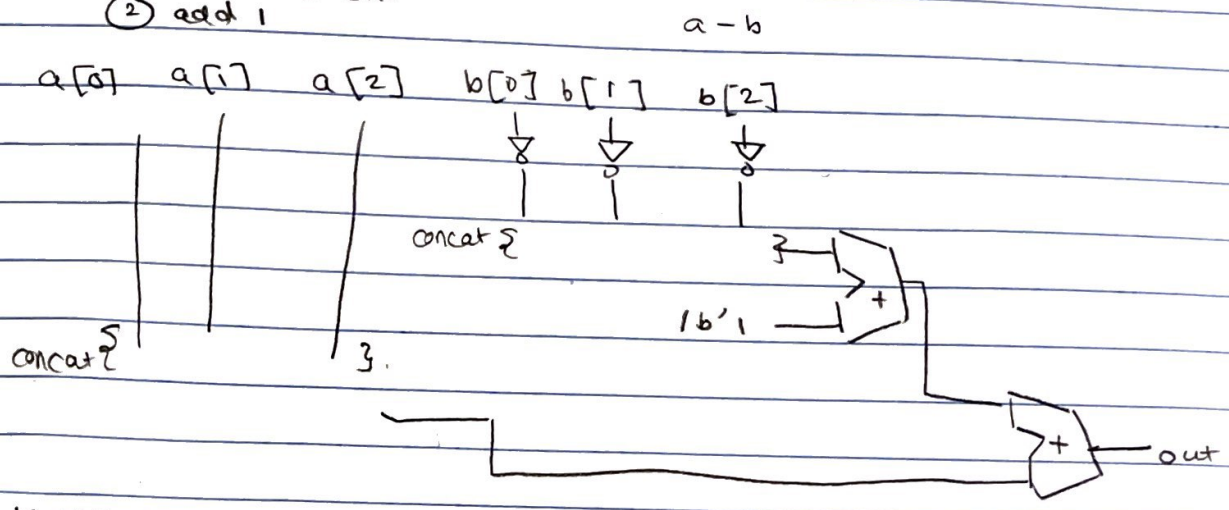
Gath Ather

Full Subtractor

subtraction is performed by taking the two's complement of the second number, then adding

two's complement

① invert the bits
② add 1



Note: all tests pass for 32 bit input & output

(3) Finish Lab 2 Part A

- pulse generator
- triangle generator
- PWM

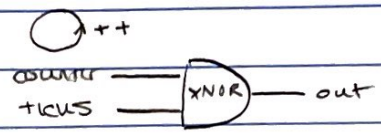
pulse_generator.sv and **test/**

- module outputs high for exactly one clock cycle out of every N ticks

count	output
0	0
1	0
2	0
3	0
4	1
0	0
1	0
2	0
3	0
4	1

① make a counter that goes from 0 to N-1 every iteration increment by 1
② if counter == N-1 output 1 and counter r=0
else output 0

checked w/ gtkwave



triangle_generator and **test/**

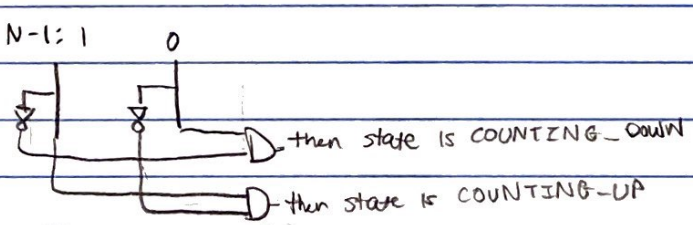
- generates triangle waves
- counts from 0 to 2^N-1 then back down again
- if ena is low hold value
  else increment/decrement

$N=2$ $2^N-1 = 4-1 = 3$

counter	state
00	0
01	0
10	0
11	1
10	1
01	1
00	0

transition here so ready for next posClk — if all bits on state ⇐ 1

transition here so ready for next posClk — if all bits off state ⇐ 0



pwm.sv

```
if (step):
    if (ena == 0)
        out ⇐ 0
    if (duty == 0)
        out ⇐ 0
    if (duty = 2^N - 1)
        out ⇐ highest
```

(A) use counter

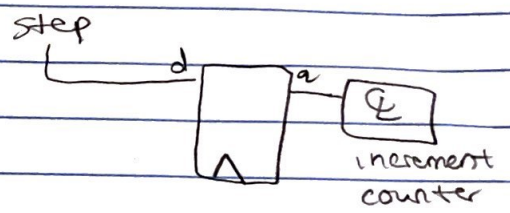
```
if counter == duty
    counter ⇐ 0
    out ⇐ 1
else
    out ⇐ 0
    counter ++
```

counter	duty	out
0	2	0
1 ← change here	2	0
2	2	1
0	2	0
1 ← change here	2	0
2	2	1
0	2	0
1	2	0
2	2	1

11.0

— pos Clk

— pos



Note: step isn't on regularly enough for any logic to make sense (in the tests, step isn't on for enough times for counter to reach duty :))

incrementing counter must happen in sequential logic

checking must also happen in sequential logic and d ⇒ q

⇒ everything in sequential logic