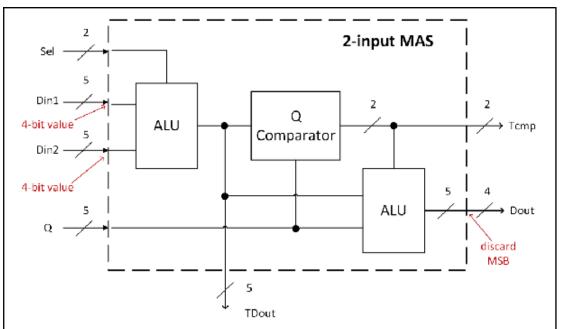
2024 Digital IC Design Homework 1

27.12.67	2024]	<u>Digital IC Desig</u>	<u>gn Homew</u>	vork I	
NAME	楊晴雯				
Student ID	P76114511				
Functional Simulation Result					
Stage 1	l Pass	Stage 2 Pa	ass	Stage	3 Pass
		Stage 1			
#	. ———				
#		Stage∙1			
#		-ALU-Simulatio	n Begin		
#					
#					
#		ALU Simulation	Success		
#					
#					
#		ALU Simulatio	n - End		
#					
		G. A			
Stage 2					
**					
**		Stago 3			
#	Com				
#	. —— . СОШ	parater Simula	rt Toll - Beg	TII - ——	
**					
#		anatan Cimulat	ion Succ		
**	· · —— · Collip	arater Simulat	.10II · Succ	622	
#					
**		mparater Simul	ation En	d	
**		iiiparacer - 31iilu t	acton En	u -	
		Stage 3			
	#	Stage-3			
	#				
	# 2-in #	put MAS Simulation	Begin		
	#				
		ut MAS Simulation S	Success - —		
	#				
	# · · 2-i	nput MAS Simulation	· End ·		
	#				
	**			· · · /1/1	
		 			
		Pass!	#### /^		
	#			^.^.^. w	
		finish . /+h ./		ml	
		finish : ./tb.v(007500 ps Iteratio		nce: /MAS tb	
		Description of yo	ur design		



The Modular Adder Subtractor is essentially a ALU (supporting addition and subtract), but will limit the output in a certain range [0, +Q). Given the control signal Sel, two numbers Din1 and Din2, the quotient Q, the LHS ALU performs the normal arithmetic operations based on input signal Sel on Din1 and Din2, and then the Q Comparator receives the result as **TDout**; it determines whether TDout >= Q, and whether it >= 0, and then write the 2-bit result **Tcmp** (determine if the result is within the range). Finally, this result is used as the signal for the RHS ALU to determine whether (1) to trim the value down or to add it up into [0, +Q) or (2) do nothing because the value itself is within the range.

- I write only 1 ALU and instantialize it twice.
- Since the RHS ALU needs 3 states (add, sub, noop) for the control signal, the Sel signal in the ALU module is designed to have 2-bit, despite that in LHS it needs to express 2 states only.

The design itself is simple to describe provided that the inputs are all controlled (eg. Din1, Din2 are all positive numbers of 4-bit...).

• I found that the testbench has small bugs; I think in the 2nd initial block, it should be

```
$display("ERROR: %d + %d should be %d ,not
%d\n",Din1,Din2,ALU_GOLD,TDout) instead of
$display("ERROR: %d + %d should be %d ,not
%d\n",Din1,Din2,GOLD,ALU_out) (the 2 signals are misnamed).
```