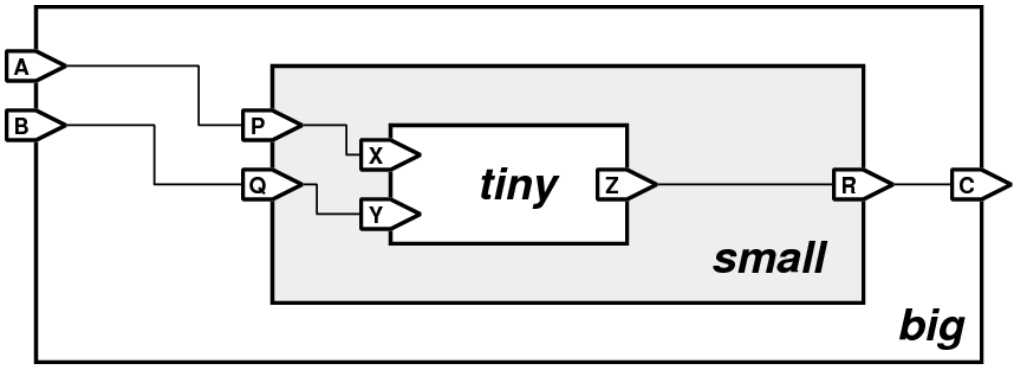
4. In this question for each part there will be two Verilog code snippets. For each part you will have to say whether both, only one, or none of the code snippets fulfill what is being asked. All code snippets are syntactically correct. They will compile and produce either a sequential circuit or a combinational circuit. (2 points each)

a) Which code snippet(s) realizes the following hierarchy of three instances given in the figure below? (Note the function "tiny" realizes a simple AND function)



Code Snippet (A)	Code Snippet (B)

`module big (input A,B,`
`output C);`
`module small (input P,Q`
`output R);`
`module tiny (input X,Y,`
`output Z);`
 	
`assign Z = X & Y;`
`assign R = Z;`
`assign C = R;`
 	
`endmodule`
`endmodule`
`endmodule`
	`module tiny (input X,Y,`
	`output Z);`
	`assign Z = X & Y;`
	`endmodule;`
	
	`module small (input P,Q`
	`output R);`
	`tiny tim (P,Q,R);`
	`endmodule`
	
	`module big (input A,B,`
	`output C);`
	`small sam (A,B,C);`
	`endmodule`

☐ Only A ☐ Only B ☐ Both A and B ☐ None

b) Which code snippet(s) will produce a four input multiplexer?

Code Snippet (A)	Code Snippet (B)

`assign z = sel[0] ?`
`(sel[1] ? c : d )`
`:`
`(sel[1] ? b : a);`
	`always @ (*)`
	`case (sel)`
	`2'b00: z = a;`
	`2'b10: z = b;`
	`2'b11: z = c;`
	`default: z = d;`
	`endcase`

☐ Only A ☐ Only B ☐ Both A and B ☐ None

c) Which code snippet(s) will produce a 8-bit value which is composed of (from MSB to LSB), $c_2c_1d_0d_0d_0001$ (c and d are both 8-bit values)?

Code Snippet (A)	Code Snippet (B)

`always @ (*)`
`  begin`
`  z  <= 8'b00000001;`
`  c2 <= c << 6;`
`  d2 <= d << 3;`
`  z  <= z & c2 & d2;`
`  end`
	`assign z = { c[2:1],`
	`  {3{d[0]}},`
	`  3b'001};`

☐ Only A ☐ Only B ☐ Both A and B ☐ None

d) Which code snippet(s) will produce a sequential circuit?

Code Snippet (A)	Code Snippet (B)

`always @ (some, signal)`
`  if (signal)`
`  lone <= some;`
	`always @ (posedge clk)`
	`  en <= data;`

☐ Only A ☐ Only B ☐ Both A and B ☐ None

e) Which code snippet(s) will produce a falling edge triggered D-type flip-flop with an asynchronous reset?

Code Snippet (A)	Code Snippet (B)

`always @ (posedge clk)`
`  if (reset) q <= 1'b0`
`  else  q <= data;`
	`always @ (negedge clk)`
	`  if (reset) q <= data;`

☐ Only A ☐ Only B ☐ Both A and B ☐ None