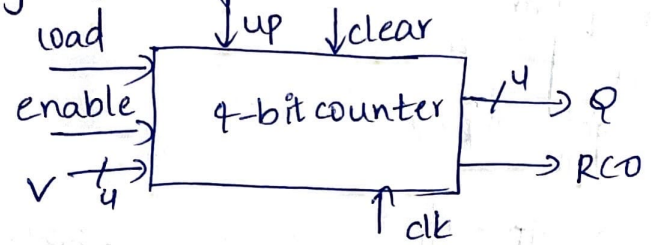
④

(a) Design 4-bit counter



so from above

ip's are
1. load
2. enable
3. clk
4. up
5. clear
6. [3:0] V
   4 bit

op's are
1. [3:0] Q
2. RCO

Conditions given → For Q

1. en = 0 op = previous value
2. en = 1 && ld = 1
   Q = V
3. en = 1 && ld = 0
   up ⇒ Q = Q + 1
   dow ⇒ Q = Q - 1

clear = 0

condition given For RCO

enable = 1, up = 1, Q = 1111 → RCO = 1

enable = 1, up = 0, Q = 0000

// declaring i/ps and o/ps
module counter_4bit(Q, RCO, ld, en, clk, up, clear, V);
output reg [3:0] Q; // as we want to use assign statement
output reg RCO;
input [3:0] V;
input ld, en, clk, up, clear;

always @(posedge clk)
begin // as there are multiple statements
if(~clear) // as it stated clear=0 this works
begin
Q <= 4'd0;
end
else if(en) // when en = 1
begin
if(ld)
begin
Q <= V; // Given when en=1; ld=1 Q=V
end
else
if(up)
Q <= Q+1 // Given up Q = Q+1
else
Q <= Q-1 // Given down Q = Q - 1
end
end
end

```
// For RCO output
always @(*)
begin
   if(en && up && (Q==4'b1111))
      RCO=1;
   else if(en && ~up && (Q==4'b0000))
      RCO=1;
   else
      RCO=0;
end
```

For test bench to test

Given

① count up for 10 clock cycles
② holds same for 5 clock cycles
③ counts down for 20 clock cycles
④ clears counter
⑤ loads the value 0xC into counter
⑥ count up for 11 clock cycles

```
module counter_4bit_tb;
reg ld,en, clk ,up, clear; //declaring as reg i/ps
reg [3:0]v;                  because it stores values
                               to test
wire [3:0]Q;
wire RCO;

counter_4bit dd (Q, RCO, ld, en, clk, up, clear, v);

//First we will generate clock

  clk = 1'b1;
  forever clk = #50 ~clk; //repeats continuously
  clear
//Initializing
   clear = 1;
    en = 0;
     ld = 0;
     up = 1;
      v = 4'd0;

//Given count up for 10 clock cycles
   #10 clear = 0   //After 10ns clear = 0
// In order to count up en = 1, ld = 0
   #5 en = 1;
   #5 ld = 0
//For 10 clock cycles we can use for loop
   for(int i=0; i<10; i++)
      #5;
```

```
// hold same for 5 clock cycles
// Already en=1 if ld=1 then ip is same as op.
#5 ld = 1;
for(int i=0; i<5; i++)
    #5;

// count down test 20 clock cycles
// Already en=1 if ld=0 and up =0 then it
   can be performed
#5 ld=0;
up =0;
for(int i=0; i<20; i++)
   #5;

// clears counter
#5 clear=0;
#5 clear=1;

// loads value 0xc into counter
#5 v = 4'hc; // given ip as 0xc
#5 ld = 1;

// count up for 11 clock cycles
// if en=1 ld=0 up=1 then countup as given
   up = 1;
#5 ld=0;
for(int i=0; i<11; i++)
   #5;
$finish
endmodule
```