```
`timescale 1ns / 1ps
//////////////////////////////////////////////////////////////////////////////////
// Company: 
// Engineer: 
// 
// Create Date: 04/27/2022 03:54:09 PM
// Design Name: 
// Module Name: countUD16L
// Project Name: 
// Target Devices: 
// Tool Versions: 
// Description: 
// 
// Dependencies: 
// 
// Revision:
// Revision 0.01 - File Created
// Additional Comments:
// 
//////////////////////////////////////////////////////////////////////////////////


module countUD16L(
    input clk,
    input Up,
    input Dw,
    input LW,
    input [15:0] d,
    
    output [15:0] q,
    output UTC,
    output DTC
    
    );
    wire [3:0] utc;
    wire [3:0] dtc;
    //making 4 count4dl which creates a 16 biy flip flop
    //in the future shouldve made some wires for the logic and made these look much 
cleaner like i did in the countUD4L
    countUD4L cnt0to3(.clk(clk), .Up(Up), .Dw(Dw), .LW(LW), .d(d[3:0]), .q(q[3:0]), 
.UTC(utc[0]), .DTC(dtc[0]));
    countUD4L cnt4to7(.clk(clk), .Up(Up & utc[0]), .Dw(Dw & dtc[0]), .LW(LW), 
.d(d[7:4]), .q(q[7:4]), .UTC(utc[1]), .DTC(dtc[1]));
    countUD4L cnt8to11(.clk(clk), .Up(Up & utc[0] & utc[1]), .Dw(Dw & dtc[0] & 
dtc[1]), .LW(LW), .d(d[11:8]), .q(q[11:8]), .UTC(utc[2]), .DTC(dtc[2]));
    countUD4L cnt12to15(.clk(clk), .Up(Up & utc[0] & utc[1] & utc[2]), .Dw(Dw & 
dtc[0] & dtc[1] & dtc[2]), .LW(LW), .d(d[15:12]), .q(q[15:12]), .UTC(utc[3]),
.DTC(dtc[3]));
    //when all true it returns left most dot is going to be on
    assign UTC = utc[0] & utc[1] & utc[2] & utc[3];
    //dtc is gotten from the count4dl which is why non of these are inverted
    //the inversion for dtc is already made in count4dl
    assign DTC = dtc[0] & dtc[1] & dtc[2] & dtc[3];
endmodule
```