Module: R3: DLD + DSD
Section: Sequential Circuits **Task:** Design Problem

Design Problem

Sequential Circuits

➢ **Question: Design a stopwatch using counters:**

a. **Design:**

I have used four counters to make this stopwatch. The stopwatch can show up to 9 minutes, 59 seconds, and 9 tenths of a seconds. So, three counters go up to 9, and one counter goes up to 5.

First, we count from 0 to 9 (for tenths of a second). When we reach 9, it sends a signal to the next counter (for seconds). This next counter also counts from 0 to 9. When it reaches 9, it sends a signal to the third counter

This counter goes up to 5. When it reaches 5, it sends a signal to the last counter (for minutes), which counts up to 9.

So, each counter sends a signal to the next one when it reaches its maximum count. This way, we keep track of time accurately.

The truth tables will be as follows:

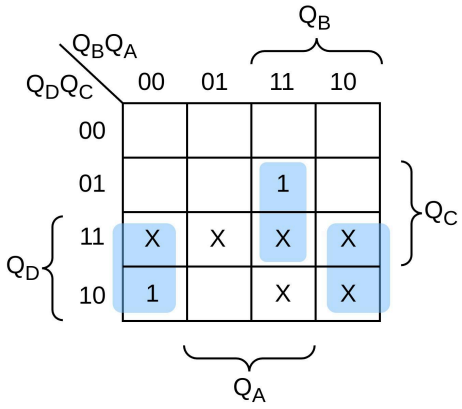
b. **Truth Table:**

- **Counter 9**

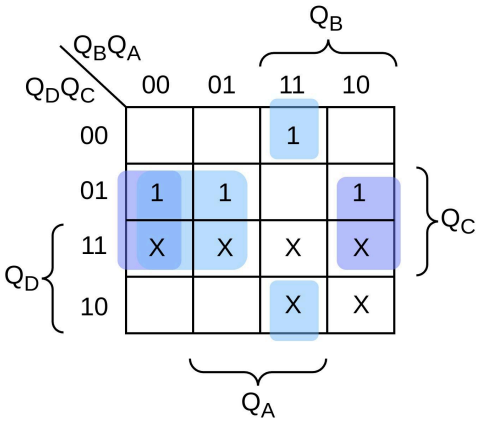
Present State				Next State				FF Inputs			
Q_D	Q_C	Q_B	Q_A	QD (t+1)	Q_C (t+1)	Q_B (t+1)	Q_A (t+1)	D_D	D_C	D_B	D_A
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	1	0	0	1	0	0	0	1	0
0	0	1	0	0	0	1	1	0	0	1	1
0	0	1	1	0	1	0	0	0	1	0	0
0	1	0	0	0	1	0	1	0	1	0	1
0	1	0	1	0	1	1	0	0	1	1	0
0	1	1	0	0	1	1	1	0	1	1	1
0	1	1	1	1	0	0	0	1	0	0	0
1	0	0	0	1	0	0	1	1	0	0	1

1	0	0	1	0	0	0	0	0	0	0	0

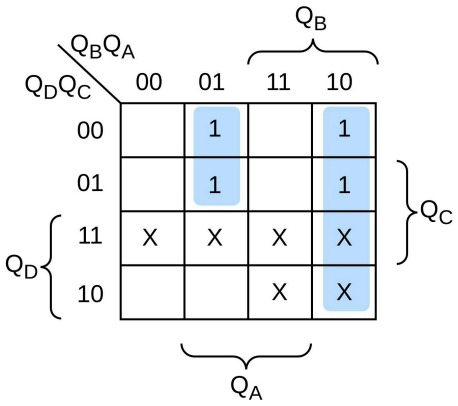
Using K-Maps:



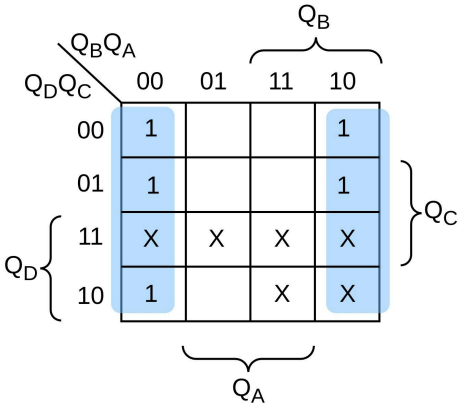
$D_D = Q_A' Q_D + Q_C Q_B Q_A$



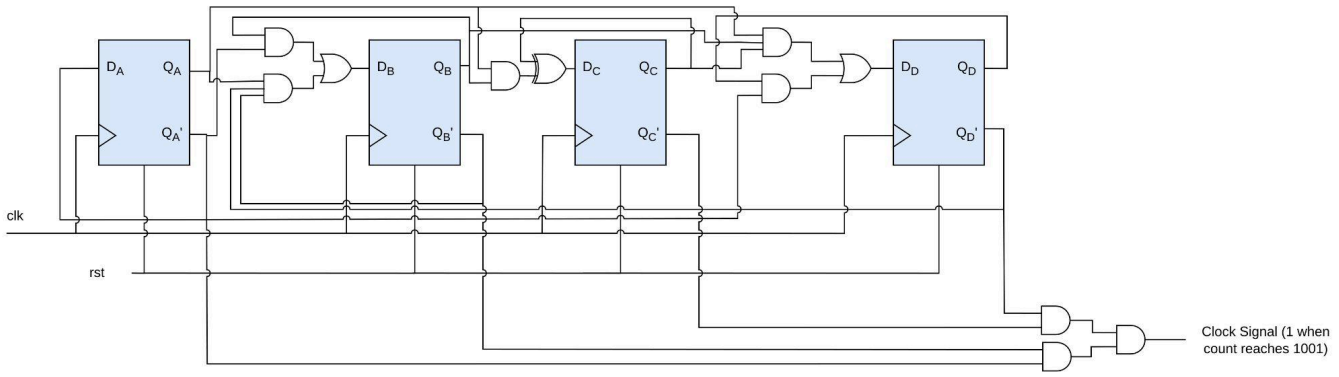
$D_C = Q_C \wedge (Q_B Q_A)$



$D_B = Q_D' Q_B' Q_A + Q_B Q_A'$



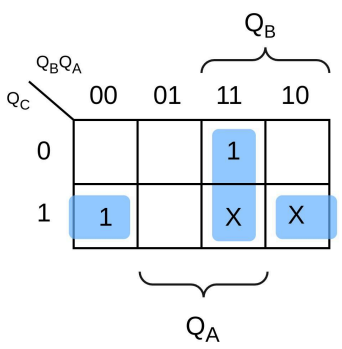
$D_A = Q_A'$



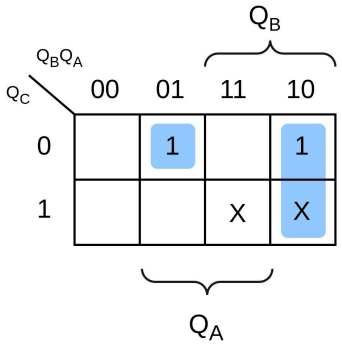
■ Counter 5

Present State			Next State			FF Inputs		
Q_C	Q_B	Q_A	Q_C (t+1)	Q_B (t+1)	Q_A (t+1)	D_C	D_B	D_A
0	0	0	0	0	1	0	0	1
0	0	1	0	1	0	0	1	0
0	1	0	0	1	1	0	1	1
0	1	1	1	0	0	1	0	0
1	0	0	1	0	1	1	0	1
1	0	1	0	0	0	0	0	0

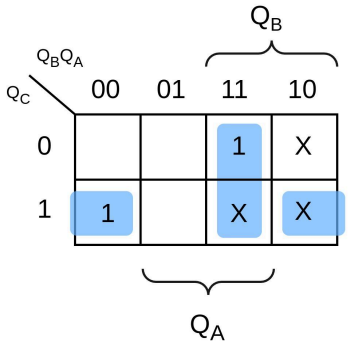
Using K-Maps:



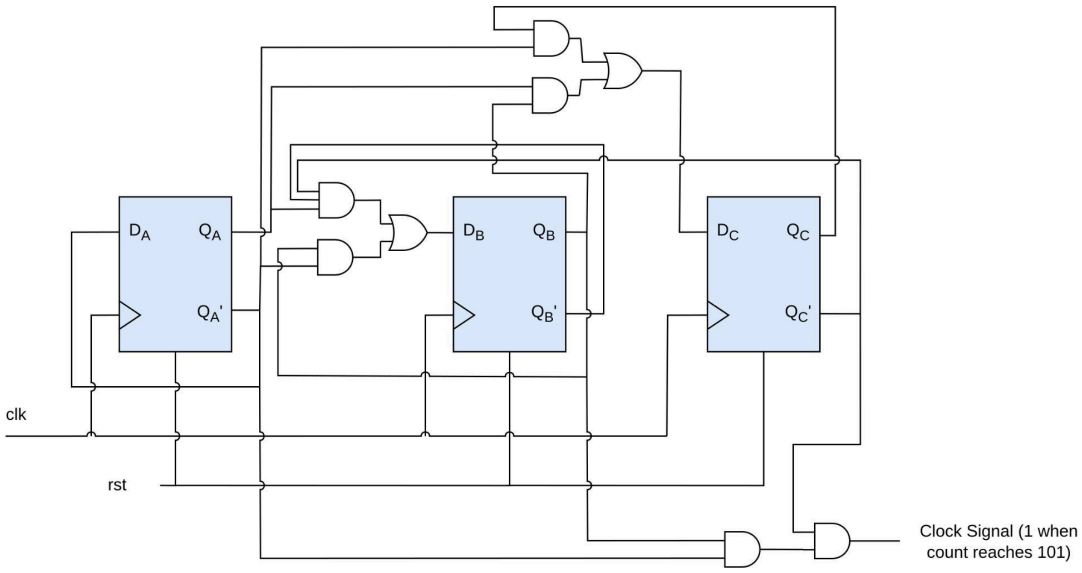
$D_C = Q_CQ_A' + Q_BQ_A$



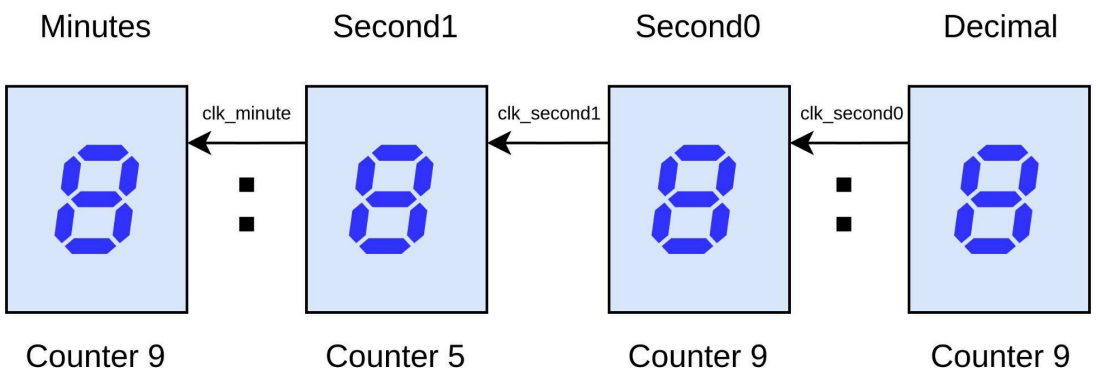
$D_B = Q_C'Q_B'Q_A + Q_BQ_A'$



$D_A = Q_A'$



Final circuit will look like following:



c. Verilog Code:

```
module stopwatch (input clk, clr, output [3:0] min, sec0, deci,
output [2:0] sec1);

//Deci
counter9 decimal (.clk(clk), .rst(clr), .out(deci));

//Second_0 (LSB)
wire clk_sec0;
assign clk_sec0 = (~deci[3] & ~deci[2] & ~deci[1] & ~deci[0]);
counter9 second_0 (.clk(clk_sec0), .rst(clr), .out(sec0));

//Second_1 (MSB)
wire clk_sec1;
assign clk_sec1 = (~sec0[3] & ~sec0[2] & ~sec0[1] & ~sec0[0]);
counter5 second_1 (.clk(clk_sec1), .rst(clr), .out(sec1));

//Minutes
wire clk_minute;
assign clk_minute = (~sec1[2] & ~sec1[1] & ~sec1[0]);
counter9 minute (.clk(clk_minute), .rst(clr), .out(min));

endmodule


    module counter9 (input clk, rst, output [3:0] out);
```

```
        wire D_3, D_2, D_1, D_0;
        // D_d = D_3 = Q0' Q3 + Q2 Q1 Q0
        assign D_3 = (~out[0] & out[3]) | (out[2] & out[1] &
out[0]);

        // D_c = D_2 = Q2 ^ (Q1 Q0)
        assign D_2 = out[2] ^ (out[1] & out[0]);

        // D_b = D_1 = (Q3' Q1' Q0) + (Q1 Q0')
        assign D_1 = (~out[3] & ~out[1] & out[0]) | (out[1] &
~out[0]);

        // D_a = D_0 = Q0'
        assign D_0 = ~out[0];

        d_ff D (.d(D_3), .clk(clk), .rst(rst), .q(out[3]));
        d_ff C (.d(D_2), .clk(clk), .rst(rst), .q(out[2]));
        d_ff B (.d(D_1), .clk(clk), .rst(rst), .q(out[1]));
        d_ff A (.d(D_0), .clk(clk), .rst(rst), .q(out[0]));

    endmodule



    module counter5 (input clk, rst, output [2:0] out);

        wire D_2, D_1, D_0;
        // D_c = D_2 = Q1 Q0 + Q2 Q0'
        assign D_2 = (out[1] & out[0]) | (out[2] & ~out[0]);

        // D_b = D_1 = Q1 Q0' + Q2' Q1' Q0
        assign D_1 = (out[1] & ~out[0]) | (~out[2] & ~out[1] &
out[0]);

        // D_a = D_0 = Q0'
        assign D_0 = ~out[0];


        d_ff C (.d(D_2), .clk(clk), .rst(rst), .q(out[2]));
        d_ff B (.d(D_1), .clk(clk), .rst(rst), .q(out[1]));
        d_ff A (.d(D_0), .clk(clk), .rst(rst), .q(out[0]));
```

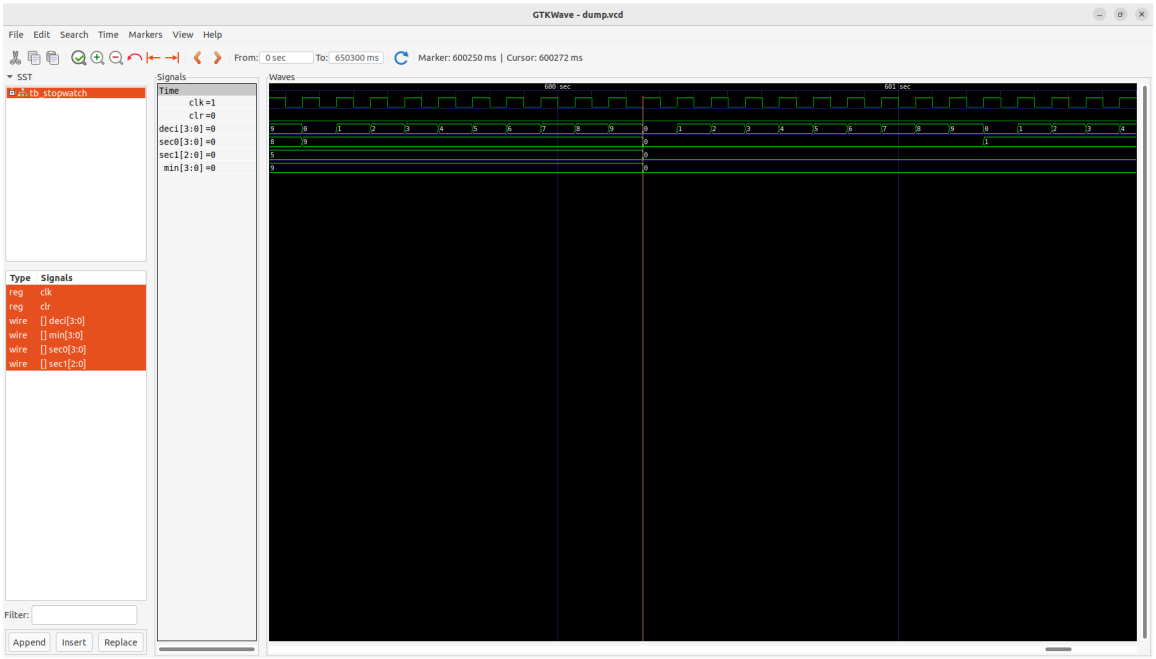
```
    endmodule


    module d_ff (input d, clk, rst, output reg q);

    always @(posedge clk or posedge rst) begin

        if (rst)
            q <= 1'b0;
        else
            q <= d;
    end
    endmodule
```

d. Testbench:

```
    `timescale 1ms/100us
    module tb_stopwatch;

    //input clk, clr, output reg [3:0] min, sec0, deci, output reg
[2:0] sec1
    reg clk, clr;
    wire [3:0] min, sec0, deci;
    wire [2:0] sec1;

    stopwatch dut (.clk(clk), .clr(clr), . min(min), .sec0(sec0),
.deci(deci), .sec1(sec1));

    always #50 clk = ~clk;
    initial begin

        $dumpvars;
        clk = 0;
        clr = 0;

        #20;
        clr = 1;

        #300;
        clr = 0;
```

```
        #650000;
        $finish;
    end

    initial begin
        $monitor("%d : %d%d : %d",min,sec1,sec0,deci);
    end

    endmodule
```

e. **Output:**



f. **Duty Cycle and Frequency:**

i. **Pulse-width:** From the waveform, it is evident that the pulse-width is 50 ms. Since we are toggling the clock every 50 time units in our code and our timescale is in ms (1ms = time unit).

ii. **Duty Cycle:**

```
Duty Cycle = (Time the signal is high/Total time period) * 100
Duty Cycle = (50 ms/100 ms) * 100 = 50%
```

iii. **Frequency:**

```
Frequency = 1/time period
```

```
Frequency = 1/100 ms = 10 Hz
```