Name: Samun Islam Ahmed

ID: 2019380182

Assignment-9

page-321

Problem – 5.1

Sol^n: The following is the circuit diagram.

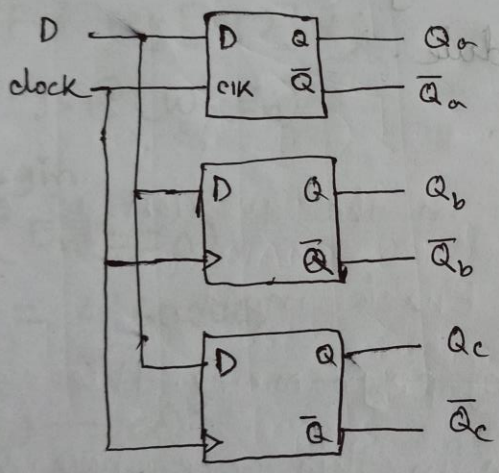


Fig: 2

In figure-2, notice that there are three different D flip-flops, the first element is a normal 'D' latch. It follows the 'D' input provided the clock is high. The second D flip-flop is a positive clock edge modifies from 0 to 1.

The last 'D' flip-flop is a negative clock edge triggered one. This device responds only when the clock input modifies from 1 to 0. The output timing diagram of the three outputs Q_a, Q_b, and Q_c is shown in Figure-3.

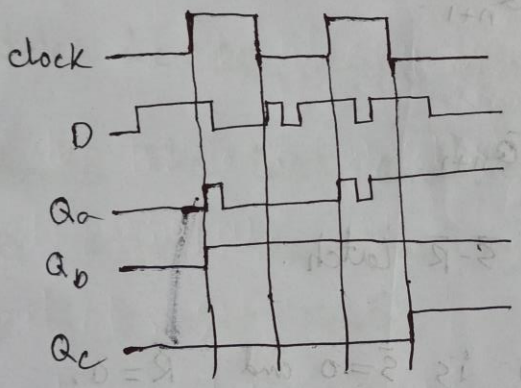


Fig-3

Problem- 5.2

Soln:

NAND based $\bar{S}$ $\bar{R}$ Latch.

Another basic latch circuit constructed using cross-coupled NAND gates is shown in Fig 7.5. The operation of NAND $\bar{S}$-$\bar{R}$ latch. To understand the operation of NAND-based $\bar{S}$-$\bar{R}$ latch is

shown in Table 7.2 which is different from that of a NOR-based S-R latch. This latch is called as $\bar{S}$-$\bar{R}$ latch, i.e, here $\bar{S}=0$ & $\bar{R}=1$ will set the latch.

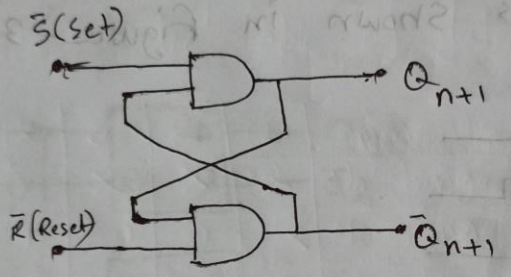


Fig: NAND based $\bar{S}$-$\bar{R}$ latch.

Case 1: The first condition is $\bar{S}=0$ and $\bar{R}=0$. When latch inputs go to 0, latch outputs go to 1, i.e, $Q_{n+1}=1$ and $\bar{Q}_{n+1}=1$. This condition is ambiguous & should not be used.

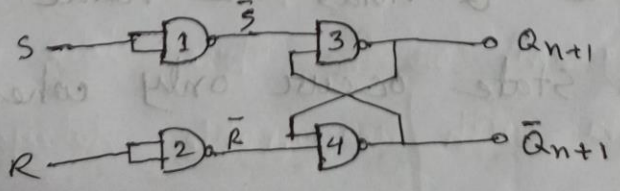
Case 2: The condition $\bar{S}=0$ and $\bar{R}=1$ always produces $Q_{n+1}=1$ regardless of the present state of the latch output. This condition sets the state of the latch, i.e, as shown $Q_{n+1}=1$ & $\bar{Q}_{n+1}=0$

Case 3: The condition $\bar{S}=1$ and $\bar{R}=0$ forces the lower NAND gate output to 1; i.e, $\bar{Q}_{n+1}=1$. Now both the inputs of upper NAND gate are 1, and therefore the output of upper NAND gate is LOW, i,e, $Q_{n+1}=0$, regardless of the prior State of the latch. This condition resets (clear) the latch i.e. $Q_{n+1}=0$ and $\bar{Q}_{n+1}=1$

Case 4: The last condition $\bar{S}=1$ and $\bar{R}=1$ does not affect the state of the latch. It remains in it's prior state.

Inputs		Outputs		Action
$\bar{S}$	$\bar{R}$	Q_{n+}	$\bar{Q}_{n+1}$	
0	0	?	?	Forbidden
0	1	1	0	Set
1	0	0	1	Reset
1	1	Q_n	$\bar{Q}_n$	change

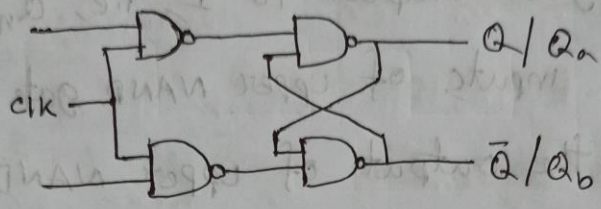
Table 7.2→ Truth table of NAND-gates $\bar{S}$-$\bar{R}$ latch.



S-R latch using 4 NAND gates.

Problem - 5.3

Soln:



characteristic table of gated SR latch.

clk	S	R	Q^+	Comments
0	X	X	Q	No change
1	0	0	Q	No change
1	0	1	0	Reset
1	1	0	1	SET
1	1	1	X	

↓
Avoid it

Gated SR latch is basic latch. If S=1 and R=0, which forces the latch into the state Q=1. If S=0 and R=1, which causes Q=0. If S=0 and R=0, which cause 'Q' holds the value. of course the changes in state occur only when clk = 1.

Problem – 5.4

Soln°. The asynchronous ~~sequential~~ sequential circuit is implemented using T flip flops. The circuit generates 50-MHz at Q_0 & 25 MHz at Q_1 & 12.5 MHz at Q_2 for a 100 MHz clock signal. At T=1, conversion circuit becomes,

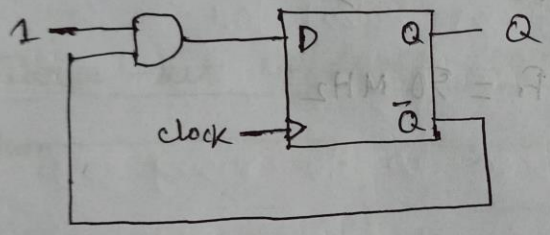
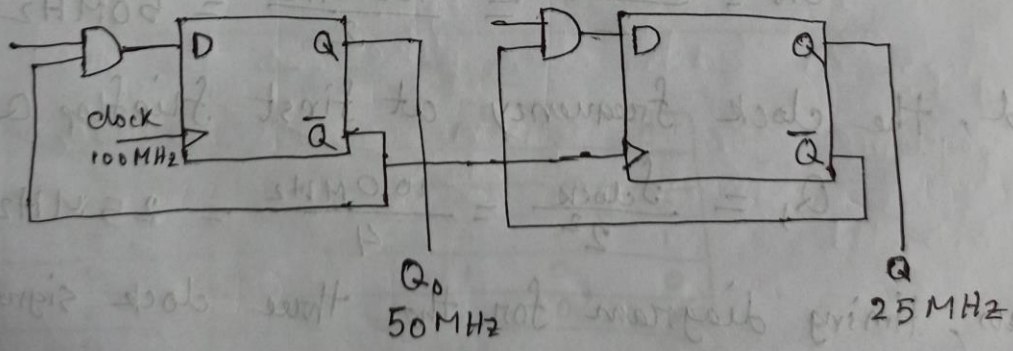
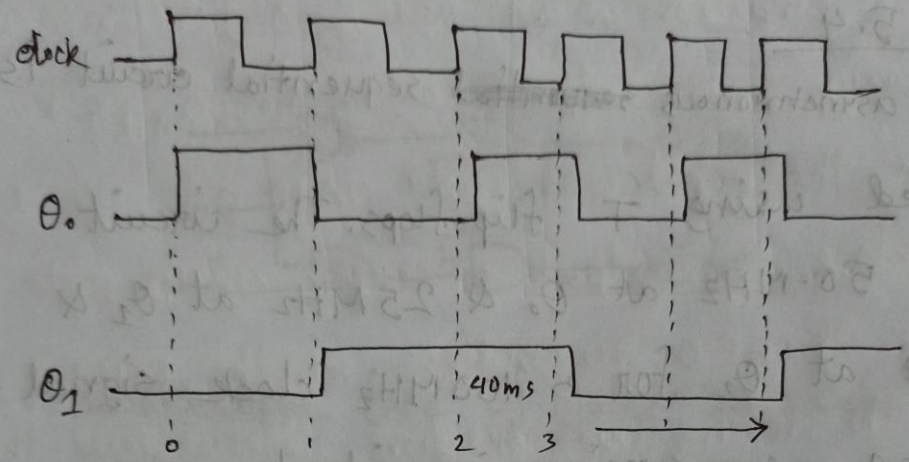


Fig: 1

This Fig-1, to replace 't' flip flop with D flip flop,



circuit is operated in the raising edge of clock.



$T_{clock} = 10ns$ $T_0 = 20ns$.

$F_{clock} = 100MH_2$ $F_0 = 50\ MH_2$

$T_1 = 40ns$

$F_1 = 25MH_2$

The clock frequency at first flip flop, Q is

$$Q_0 = \frac{f_{clock}}{2} = \frac{100MH_2}{2} = 50MH_2$$

And, the clock frequency at first flipflop Q_1 is

$$Q_1 = \frac{f_{clock}}{2^2} = \frac{100MH_2}{4} = 25MH_2$$

Thus, timing diagram for the three clock signals has been drawn with reasonable delays.

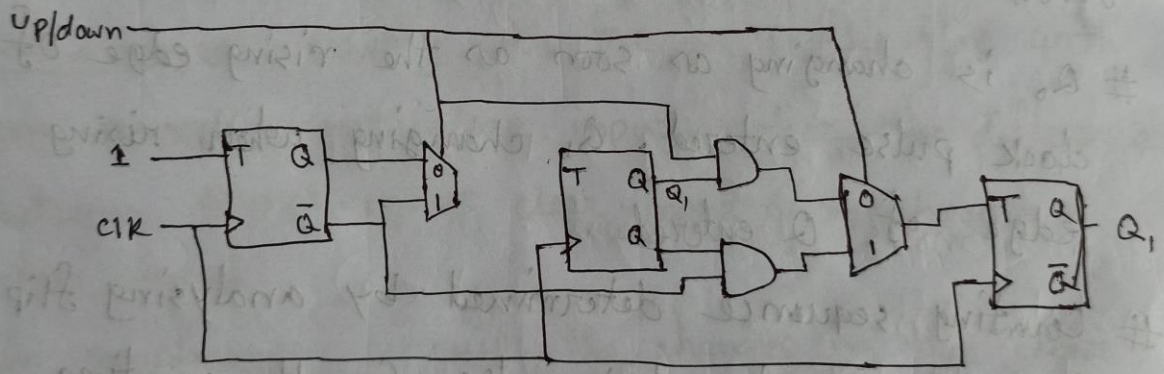
Problem - 5.15

Soln: We have to design a 3-bit up/down counter by using T flip-flops

If up/down = 0, then the circuit behave as an up-counter.

If up/down = 1, then the circuit behave as a down-counter.

Three-bit up/down counter using T flip-flops.



Truth Table

up/Down	A	B	C	A^+	B^+	C^+
0	0	0	0	0	0	1
0	0	0	1	0	1	0
0	0	1	0	0	1	1
0	0	1	1	1	0	0
0	1	0	0	1	0	1
0	1	0	1	1	1	0
0	1	1	0	1	1	1
0	1	1	1	0	0	0
1	0	0	0	1	1	1
1	0	0	1	0	0	0

Up/Down	A	B	C	A^+	B^+	C^+
1	0	1	0	0	0	1
1	0	1	1	0	1	0
1	1	0	0	0	1	1
1	1	0	1	1	0	0
1	1	1	0	1	0	1
1	1	1	1	1	1	0

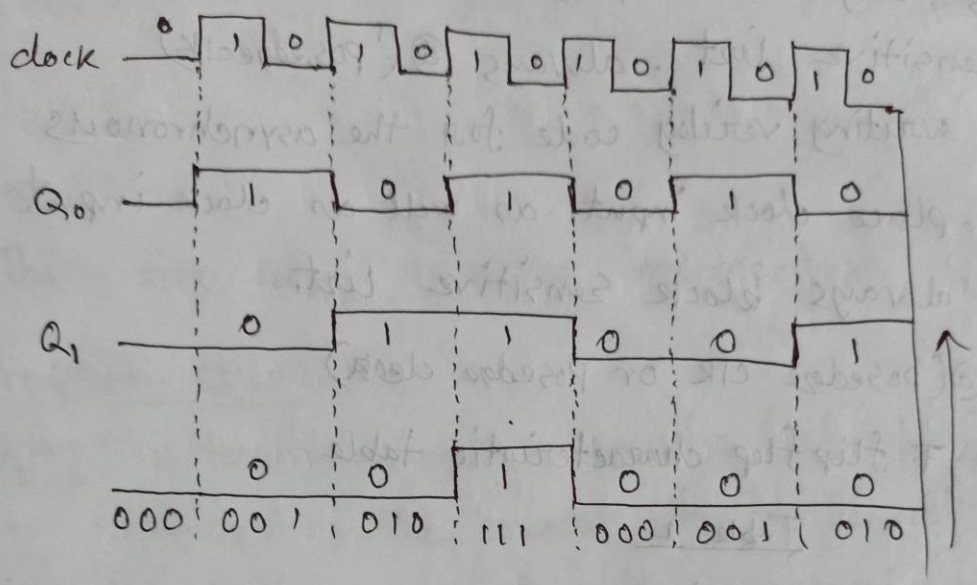
Problem - 5.17

Sol^{n}. Couter work based on clock pulse.

Timing diagram is given below for the counter given.

Q_0 is changing as soon as the rising edge of clock pulse entered. Q_1 changing when rising edge of Q entered.

Counting sequence determined by analysing flip flops in each clock pulse. So, the counting sequence for the given counter is 000, 001, 010, 111



Problem - 5.9

Soln^{n} Before writing code for a T flip-flop with an asynchronous clear input, notice the difference between asynchronous clear input & synchronous clear input.

In synchronous circuits, the changes in the clear input signal occur shortly after an action clock edge which means synchronous clear input, flip-flop to be cleared to '0' shortly after the edge of clock. So, in synchronous circuits 'clear input' is depends on the clock edge indirectly.

While writing Verilog code for the synchronous

circuits, only clock inputs should be in the always block sensitive list. always @(pos,edgeclk) whereas writing verilog code for the asynchronous circuits, place clock input as well as clock input in the always block sensitive list.

always @(posedge clk or posedge clear)

Recall T flip flop characteristic table

Table 1

T	Q(t+1)	
0	Q(t)	nothing to do, holds the value
1	$\bar{Q}(t)$	Complement the Q(t)

Verilog code representing a T flip-flop with an asynchronous clear input.

```
module tff_async_clear(T, clk, clear, q)
input T, clk, clear;
output q;
reg q;
always@(posedge clk or posedge clear)
If (clear)
begin
 q<= 0'b0;
 end
else if (T)
```

```
begin
q<=!q;
end
end module
```

Thus, desired T flip-flop implemented

Problem – 5.20

Soln: The term synchronous counter refers to changes in flip-flops of the counter are clocked at the same time by a common clock pulse. Input "Reset" assertion is also performed at time of clock pulse.

MOD-12 counter counts as follows:-

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 0 (repeat)

Write Verilog code (behavioural model) that represents a modulo-12-up counter with synchronous reset.

```
module Modulo_12_up_counter(clk,rst,Q,Qn);
input clk, rst;
output [3:0] Q, Qn;
reg [3:0] Q;
always @(posedge clk)
if (rst || Q == 4'b1011)
 Q<=4'b0000;
else
```

```
Q <= Q+1'b1;
assign Qn = ~Q;
end module;
Write test_modulo_12_up_counter;
//Input
reg clk;
reg rst;
//outputs
wire [3:0] Q;
wire [3:0] Qn;
//Instantiate the Unit Under Test (UUT)
Modulo_12_up_counter uut(
                .clk(clk)
                .rst(rst)
                .Q(Q)
                .Qn(Qn)
);
initial begin
//Initialize Inputs
clk = 0;
rst = 1;
//wait 100ns for global reset to finish
#10
```

```
clk = 0;
rst = 0;
//wait 100ns for global reset to finish
#10;
//Add stimulus here
end
always
#5 clk = !clk;
end module
```

Problem - 5.21

Sol^n: Refer to the sequential circuit is Fig 7.25 in the textbook. The longest path (critical path) delay in the circuit is from the output of FF_0 to the input of FF_3.

From the output of FF_0 to the input of FF_3. There are 3 AND gates + one XOR gate + one 2 to 1 multiplexer.

The propagation delay of each AND gate, XOR gate and 2 to 1 multiplexer gates is 1ns.

So total longest delay in the circuit is $3(1ns)_{\text{AND gate}} + 1(1ns)_{\text{XOR gate}} + 1(1ns)_{\text{2 to 1 multiplexer}} = 5ns$.

Setup and hold of a flip-flop is 3ns & 1ns respectively. Let us check if they are any hold time violations in the circuit. In this case, we need to examine the shortest possible delay from a positive edge to a change in the value of the 'D' input. The shortest path through the circuit are from each flip-flop to itself through and XOR gate.

$$t_{cQ(min)} = t_{p(flip-flop)} + t_{XOR\ gate}$$
$$= 1ns + 1ns$$
$$= 2ns > t_h$$
$$= 1ns$$

There is no hold time violation so, no need to consider hold time in maximum clock frequency. Therefore, the delay of critical path include the propagation delay through three AND gates, one XOR-gate delay, 2 to 1 multiplexer & propagation delay of flip-flop. Must also account for the setup time of flip-flop Q_3. This gives,

$$T_{min} = t_{p(flip-flop)} + 3t_{p(AND\ gate)} + t_{p(XOR\ gate)} +$$
$$+ t_{p(2\ to\ 1\ multiplexer)} + t_{su}$$

$= 1ns + 3(1ns) + 1ns + 1ns + 3ns$

$= 1ns + 3ns + 1ns + 1ns + 3ns$

$\therefore T_{min} = 9ns$

$\therefore$ Find Maximum clock frequency,

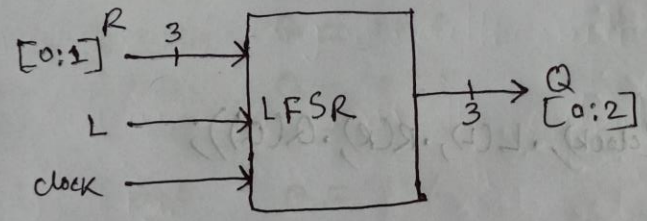
$$F_{max} = \frac{1}{T_{min}} = \frac{1}{9ns} = \frac{1}{9\times10^{-9}} = 0.111\times10^{9}$$

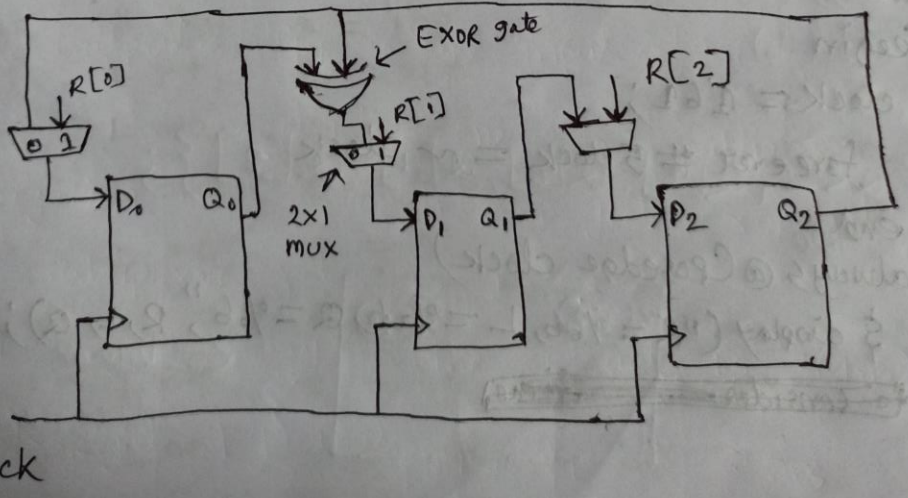
$= \boxed{111\ MHz}$.

Problem - 5.28

Sol^{n}:

LFSR diagram





LFSR Structural Diagram


```
module lfsr(R,L, clock, Q)
  input [0:2] R
  input L, clock ;
  output reg [0:2] Q;
  always @(posedge clock)
    if L
     Q<=R
    else
    Q<= {Q[2], Q[0]^Q[2], Q[1]};
end module
 module lfsr_td;
 reg [0:2]R ;
 reg L, clock;
 wire [0:2]Q;
 lfsr m1(.clock(clock), .L(L), .R(R), .Q(Q));
 initial
  begin
   clock = 1'b1;
    forever #5 clock = ~ clock
  end
  always @(posedge clock)
   $ display ("R = %b, L = %b, Q = %b", R, L, Q);
```

~~to consider done~~

```
initial
begin
Q = 001;
L = 1'b1;
@(negedge clock)
L = 1'b0;
repeat [20]
@(negedge clock);
$ finish
end
end module
```

Simulation Result

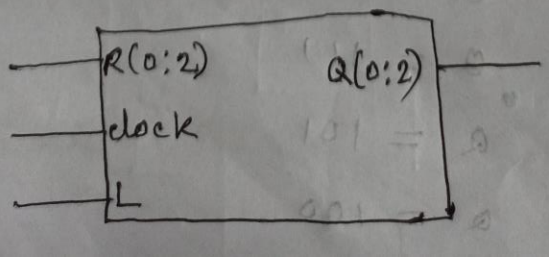
R=001, L=0, Q = 001
R=001, L=0, Q = 110
R=001, L=0, Q = 011
R=001, L=0, Q = 111
R=001, L=0, Q = 101
R=001, L=0, Q = 100
R=001, L=0, Q = 010
R=001, L=0, Q = 001
R=001, L=0, Q = 110
R=001, L=0, Q = 011
R=001, L=0, Q = 111
R=001, L=0, Q = 101
R=001, L=0, Q = 100
R=001, L=0, Q = 010

R = 001, L = 0, Q = 001
R = 001, L = 0, Q = 110
R = 001, L = 0, Q = 011
R = 001, L = 0, Q = 111
R = 001, L = 0, Q = 101
R = 001, L = 0, Q = 100

Problem – 5.29

Soln: Consider the following Verilog code of a linear-feedback shift register:

```
module ifsr(R,L,clock,Q);
  input [0:2] R;
  input L, clock;
  output reg [0:2] Q;
  always @(posedge clock)
    if(L)
    Q<=R
    else
    Q<={Q[2], Q[0], Q[1]^Q[2]};
endmodule
```



⇐ Drawn the diagram