Sheet 1

Q1-The exclusive-OR circuit of fig has gates with a delay of 3ns for an inverter,a 6ns for delay for an AND gate,and a 8ns for an OR gate.The input of the circuit goes from xy=00 to xy=01.

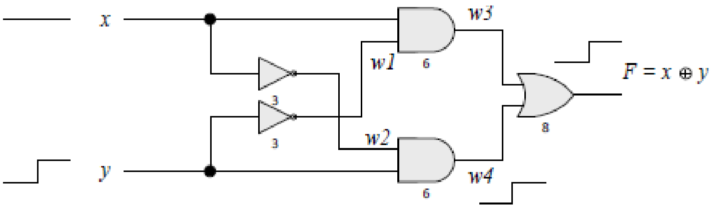
(a) Determine the signals at the output of each gate from t=0 to t=50ns.

(b) Write a Verilog gate-level description of the circuit,including the delays.

(c) Write a stimulus module and simulate the circuit to verify the answer in part (a).

ANS

(a) Block diagram:-



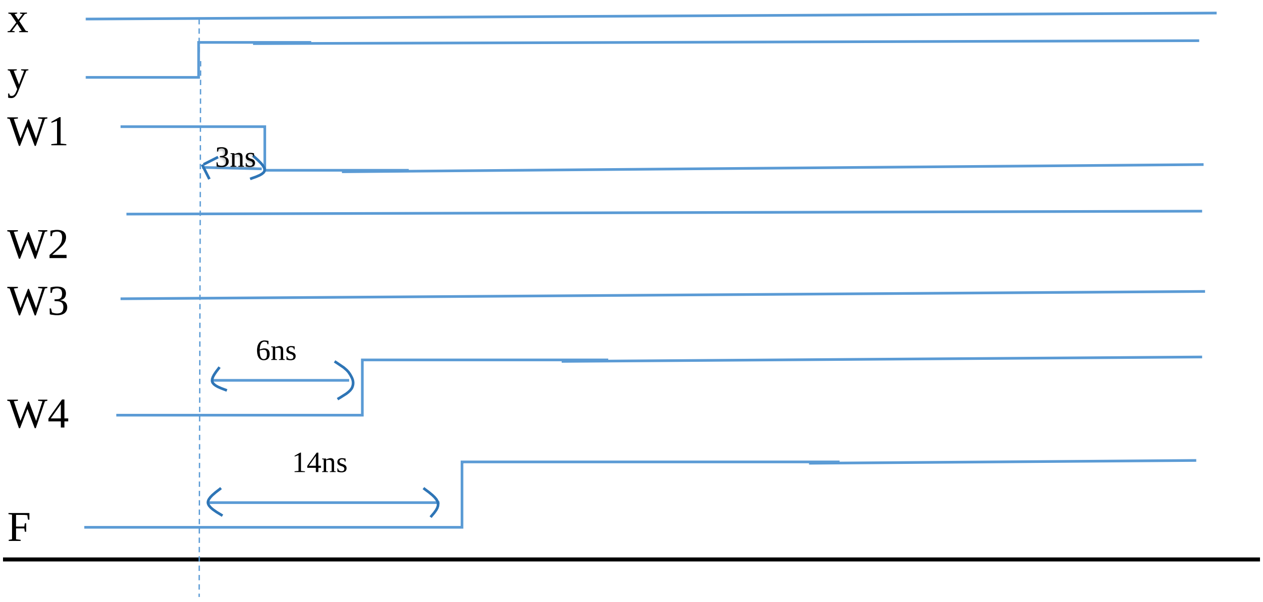
Initially when xy=00,W1=W2=1,W3=W4=0,and F=0.

xy=00→xy=11:-

After 3ns from change →W1=0,and W2=1.

After 6ns from change →W3=0,and W4=1.

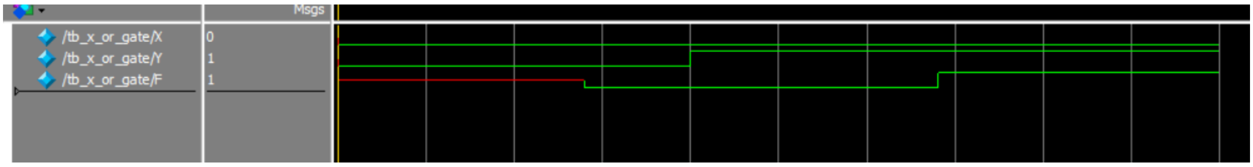
After 8ns from change →F=1.



(b&c) Code:-

```
`timescale 1ns/1ps
module x_or_gate (output F,input X,Y);
wire W1,W2,W3,W4;
not #3 (W1,Y);
not #3 (W2,X);
and #6 (W3,X,W1);
and #6 (W4,Y,W2);
or #8 (F,W3,W4);
endmodule

module tb_x_or_gate ();
reg X,Y;
wire F;
x_or_gate tb(F,X,Y);
initial
begin
X=0;
Y=0;
#20 Y=1;
end
endmodule
```

Simulation:-



Q2-Using continues assignment,write a Verilog description of the circuit specified by the following Boolean functions:

Out_1 = (A+B`)C`(C+D)

Out_2 = (C`D+BCD+CD`)(A`+B)

Out_3 = (AB+C)D+B`C

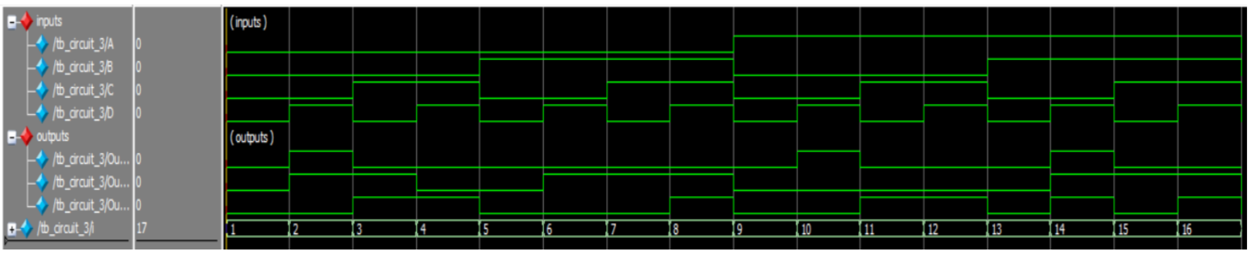
ANS

Code:-

```
module circuit_3 (output Out_1,Out_2,Out_3,input A,B,C,D);
wire not_A,not_B,not_C,not_D;
assign not_A=~A;
assign not_B=~B;
assign not_C=~C;
assign not_D=~D;
assign Out_1 = (A|not_B)&not_C&(C|D);
assign Out_2 = ((not_C&D)|(B&C&D)|(C&not_D))&(not_A|B);
assign Out_3 = ((A&B|C)&D)|(not_B&C);
endmodule
`timescale 1ns/1ps
```

```
module tb_circuit_3 ();
reg A,B,C,D;
wire Out_1,Out_2,Out_3;
integer i;
circuit_3 tb (Out_1,Out_2,Out_3,A,B,C,D);
initial
begin
A=0;
B=0;
C=0;
D=0;
for (i=0;i<17;i=i+1)
#5 {A,B,C,D}={A,B,C,D}+1;
End
endmodule
```

Simulation:-



Q3-Find the syntax errors in the following declarations (note that names for primitive gates are optional):

```
module Exmpl-3(A,B,C,D,F)                              //line 1
Inputs          A,B,C,Output D,F,                      //line 2
output           B                                     //line 3
and             g1(A,B,D);                             //line 4
not              (D,A,C),                              //line 5
OR              (F,B,C);                               //line 6
endmodule;                                              //line 7
```

ANS

Line 1: (-) not allowed to be in the module name , So use (_) instead of (-) , there is (;) required after).
Line 2: input not inputs , must be output not Output and there is required (;) after C, There is (;) required after F.
Line 3: B cant be declared as input and output.
Line 4: A cant be output because it’s declared as input in port list,D cant be input because it’s declared as output in port list.
Line 5: too many ports entire of not gate,there is (;) required afer).
Line 6: OR must be or.
Line 7: no (;) after endmodule.
So) The code after editing should be:-

```
module Exmpl_3(A,B,C,D,F);
input              A,B,C;
output             D,F;
and                 g1(D,A,B);
not                 (D,A ( /*or C*/))
```

```
Or                    (F,B,C);
endmodule
```

Q4-A majority logic function is a Boolean function that is equal to 1 if the majority of the variables are equal to 1, equal to 0 otherwise.

(a) Write a truth table for a four-bit majority function.

(b) Write a Verilog user defined primitive for a four-bit majority function.

ANS

(a) Truth table:-

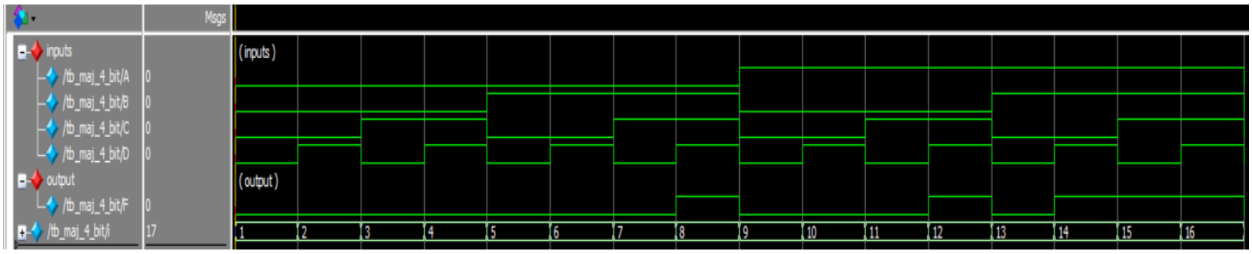
A	B	C	D	F
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	1
1	1	1	1	1

(b) Code:-

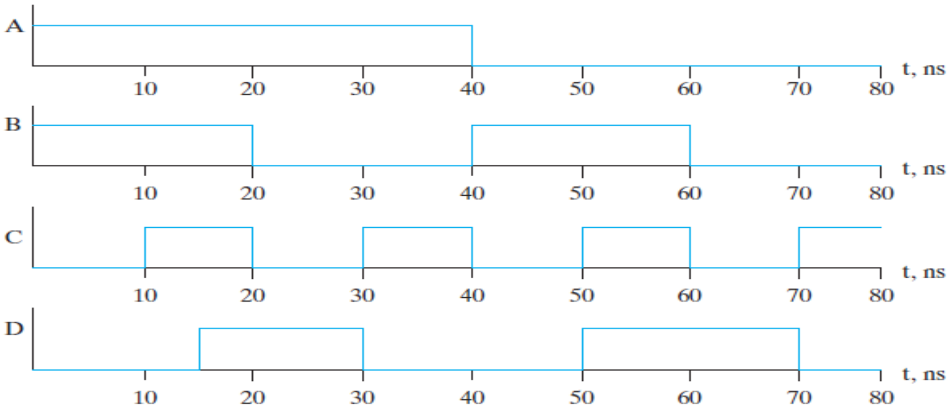
```
primitive maj_UDP (output F,input A,B,C,D);
table
0 0 0 0   : 0;
0 0 0 1   : 0;
0 0 1 0   : 0;
0 0 1 1   : 0;
0 1 0 0   : 0;
0 1 0 1   : 0;
0 1 1 0   : 0;
0 1 1 1   : 1;
1 0 0 0   : 0;
1 0 0 1   : 0;
1 0 1 0   : 0;
1 0 1 1   : 1;
1 1 0 0   : 0;
1 1 0 1   : 1;
1 1 1 0   : 1;
1 1 1 1   : 1;
```

```
endtable
endprimitive

`timescale 1ns/1ps

module tb_maj_4_bit ();
reg A,B,C,D;
wire F;
integer i;
maj_UDP M(F,A,B,C,D);
initial
begin
A=0;
B=0;
C=0;
D=0;
for (i=1;i<17;i=i+1)
#5 {A,B,C,D}={A,B,C,D}+1;
end
endmodule
```

(c) Simulation:-

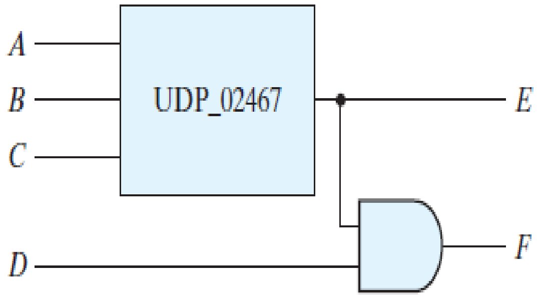


Q5-Simulate the behaviour of circuit_with_UDP_02467, using the stimulus waveforms shown in fig.



ANS

(a) Block diagram:-

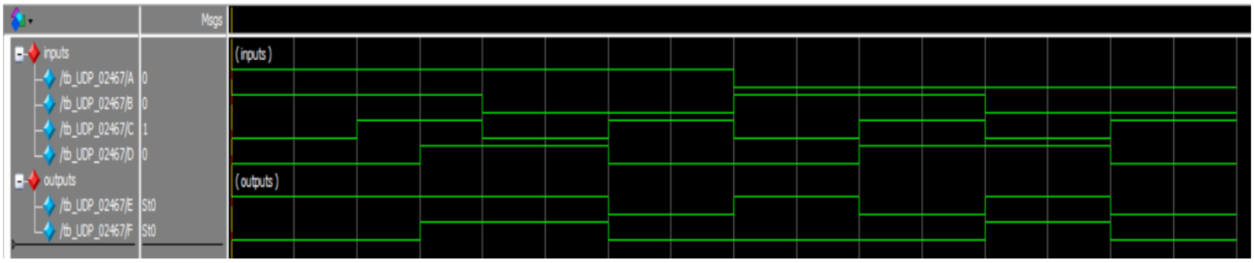


(b) Code:-

```
primitive UDP_02467 (output E,input A,B,C);
table
0 0 0 : 1;
0 0 1 : 0;
0 1 0 : 1;
0 1 1 : 0;
1 0 0 : 1;
1 0 1 : 0;
1 1 0 : 1;
1 1 1 : 1;
endtable
endprimitive

module circuit_UDP_02467 (output E,F,input A,B,C,D);
UDP_02467 DP (E,A,B,C);
and (F,E,D);
endmodule

`timescale 1ns/1ps

module tb_UDP_02467 ();
reg A,B,C,D;
wire E,F;
circuit_UDP_02467 UD (E,F,A,B,C,D);
initial
begin
```

```
    A=1;B=1;C=0;D=0;
#10 A=1;B=1;C=1;D=0;
#5  A=1;B=1;C=1;D=1;
#5  A=1;B=0;C=0;D=1;
#10 A=1;B=0;C=1;D=0;
#10 A=0;B=1;C=0;D=0;
#10 A=0;B=1;C=1;D=1;
#10 A=0;B=0;C=0;D=1;
#10 A=0;B=0;C=1;D=0;
end
endmodule
```

(c) Simulation:-


Msgs
inputs
/tb_UDP_02467/A 0
/tb_UDP_02467/B 0
/tb_UDP_02467/C 1
/tb_UDP_02467/D 0
outputs
/tb_UDP_02467/E St0
/tb_UDP_02467/F St0
(inputs)
(outputs)