Date: 9-11-2022

Assignment – 10

Aim: To understand about the conversions from one flipflop to another.

Table of Contents

=> Use the table 1, describing the input and output states of various flip-flops for your reference.

Present State	Next State	SR flip-flop inputs		D flip-flop input	JK flip-flop inputs		T flip-flop input
Q t	Q $t+1$	S	R	D	J	K	T
0	0	0	x	0	0	x	0
0	1	1	0	1	1	x	1
1	0	0	1	0	x	1	1
1	1	x	0	1	x	0	0

Q1) The J-K flip-flop is basically a gated S-R flip-flop with the addition of a clock input circuitry that prevents the illegal or invalid output condition that can occur when both inputs S and R are equal to logic level "1".

1) Use the table-1 to develop the conversion table for S-R to J-K flip-flop and generate the conversion expression between JK and SR via K map.

2) Use this conversion expression to Modify the S-R flip flop created in last lab into JK flip flop.

3) Verify the functionality of J-K flip flop using suitable Testbench.

Step 1:

Q_n	Q_{n+1}	S	R
0	0	0	X
0	1	1	0
1	0	0	1
1	1	X	0

Excitation table of SR Flipflop

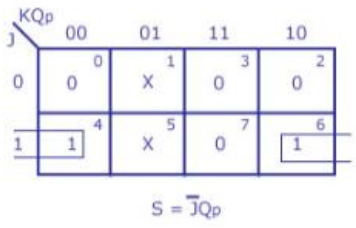
Step 2: Characteristic table of jk flipflop

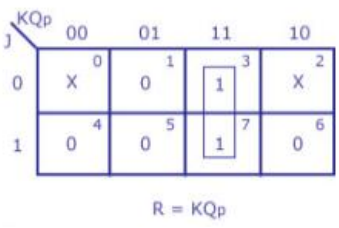
JK flip-flop			
Characteristic table			
J	**K**	**Comment**	Q_{next}
0	0	hold state	Q
0	1	reset	0
1	0	set	1
1	1	toggle	$\overline{Q}$

Step 3: conversion table from sr to jk FF

J-K Inputs		Outputs		S-R Inputs	
J	K	Qp	Qp+1	S	R
0	0	0	0	0	X
0	0	1	1	X	0
0	1	0	0	0	X
0	1	1	0	0	1
1	0	0	1	1	0
1	0	1	1	X	0
1	1	0	1	1	0
1	1	1	0	0	1

k-map conversion:





Verilog Code:

```
module tb_jk_ff;
 reg j, k, CLK;
 wire Q, QBAR;

   jk_ff dut (j, k, CLK,Q,QBAR) ;
initial
begin
j= 0; k= 0; CLK=1; # 1;
   $display ( "CLK = %b, j = %b, k = %b, Q = %b, QBAR = %b",CLK,j, k,
Q, QBAR);
j= 0; k= 1; CLK=1; # 1;
   $display ( "CLK = %b, j = %b, k = %b, Q = %b, QBAR = %b",CLK,j, k,
Q, QBAR);
   j= 1; k= 0; CLK=1; # 1;
   $display ( "CLK = %b, j = %b, k = %b, Q = %b, QBAR = %b",CLK,j, k,
Q, QBAR);
   j= 1; k= 1; CLK=1; # 1;
   $display ( "CLK = %b, j = %b, k = %b, Q = %b, QBAR = %b",CLK,j, k,
Q, QBAR);
$finish;

end
initial
begin
$dumpfile("dump.vcd");
$dumpvars (1);
end
endmodule
```

```
module sr_ff(s, r, q, qbar);
    input s, r;
    output q, qbar;
    nand(q, s, qbar);
    nand(qbar, r, q);
endmodule


module jk_ff (j, k, clk, q, qbar);
    input j, k, clk;
    output q, qbar;
    wire sn = clk & qbar & j;
    wire rn = clk & q    & k;
    sr_ff SRL (sn, rn, q, qbar);
endmodule
```

Output:

```
[2022-11-12 22:37:56 EST] iverilog '-Wall' design.sv testb
VCD info: dumpfile dump.vcd opened for output.
CLK = 1, j = 0, k = 0, Q = 1, QBAR = 1
CLK = 1, j = 0, k = 1, Q = 1, QBAR = 0
CLK = 1, j = 1, k = 0, Q = 0, QBAR = 1
CLK = 1, j = 1, k = 1, Q = 0, QBAR = 1
Finding VCD file...
./dump.vcd
[2022-11-12 22:37:57 EST] Opening EPWave...
Done
```

Output:

(figure 1.1: EPWave output)

Q2) Now lets convert the J-K flipflop created in question 1 into D flipflop. One can use the table 1 to perform similar steps as per previous solution for this conversion.

1) Develop the behavioural Verilog module of D flip-flop, converting it from a J-K flip flop. You may utilize if-else statements to develop your verilog code.

2) Verify the functionality via a suitable test bench code.

Step 1: Excitation table of jk ff

Q_N	Q_{N+1}	J	K
0	0	0	X
0	1	1	X
1	1	X	0
1	0	X	1

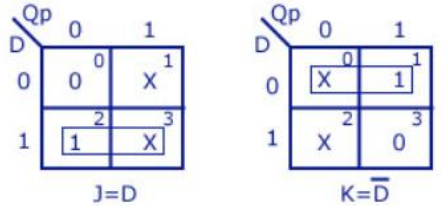
Step 2: Characteristic table of d ff

D	Q(t+1)	
0	0	Reset
1	1	Set

Step 3: Conversion table

D Input	Outputs Qp	Outputs Qp+1	J-K Inputs J	J-K Inputs K
0	0	0	0	X
0	1	0	X	1
1	0	1	1	X
1	1	0	X	0

K-map conversion:



Verilog Code:

```
//21BCP341
module tb_d_ff;
 reg d, CLK;
 wire Q, QBAR;

   d_ff dut (d, CLK,Q,QBAR) ;
initial
begin
d = 0; CLK=1; # 1;
   $display ( "CLK = %b, d = %b, Q = %b, QBAR = %b",CLK,d, Q, QBAR);
d = 1; CLK=1; # 1;
   $display ( "CLK = %b, d = %b, Q = %b, QBAR = %b",CLK,d, Q, QBAR);
$finish;

end
initial
begin
$dumpfile("dump.vcd");
$dumpvars (1);
end
endmodule

```

```
module sr_ff(s, r, q, qbar);
    input s, r;
    output q, qbar;
    nand(q, s, qbar);
    nand(qbar, r, q);
endmodule

module jk_ff (j, k, clk, q, qbar);
    input j, k, clk;
    output q, qbar;
    wire sn = clk & qbar & j;
    wire rn = clk & q & k;
    sr_ff SRL (sn, rn, q, qbar);
endmodule

module d_ff(input d,input clk, output q, output qbar);
  wire t1,t2;
  assign t1 = d;
  assign t2 = ~d;
  jk_ff jkf(t1,t2,clk,q,qbar);
endmodule

```

Output:

```
[2022-11-12 22:49:04 EST] iverilog '-Wall' desig
VCD info: dumpfile dump.vcd opened for output.
CLK = 1, d = 0, Q = 1, QBAR = 0
CLK = 1, d = 1, Q = 0, QBAR = 1
Done
```

Output:

(figure 2.1: EPWave output)

Q3) Develop a characteristic table of T flip-flop along with the excitation inputs of JK flip flop from table 1.

1) Use K map to develop the conversion relation between T and JK flip-flop.

2) Develop a behavioural Verilog module for JK flip-flop using case statements. Modify it to act like a T flip-flop.

3) Validate the modification via a suitable test bench.

Step 1: Excitation table of jk ff:

Q_N	Q_{N+1}	J	K
0	0	0	X
0	1	1	X
1	1	X	0
1	0	X	1

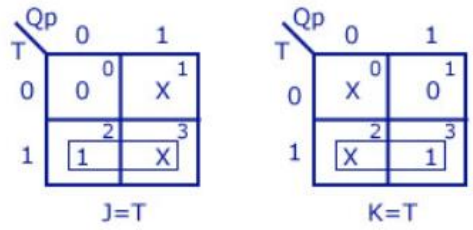
Step 2: Characteristic table of T ff:

T	Q(t+1)	
0	Q(t)	No change
1	Q(t)'	Toggle

Step 3: Conversion table:

T Input	Outputs Qp	Outputs Qp+1	J-K Inputs J	J-K Inputs K
0	0	0	0	X
0	1	1	X	0
1	0	1	1	X
1	1	0	X	1

K-map conversion:



Verilog Code:

```
module test;
reg clk=0;
reg t=0;
reg reset=1;
wire q, qnot;

  jkff dut(reset, clk,t,q,qnot);

initial
  begin
    $dumpfile("dump.vcd");
    $dumpvars(1);
    t=1'b0;
#5  reset=1'b0;
#25 $finish;
  end

always #1 clk=~clk;

endmodule
```

```
module jkff(input reset, input clk, input t,output reg q,output
qnot);

  assign j=t;
  assign k=t;
  assign qnot=~q;
always @(posedge clk)
  if (reset) q<=1'b0; else
    case ({t})
1'b0: q<=q;
1'b1: q<=~q;
 endcase
endmodule
```

Output:

(figure 3.1: EPWave output)