Lab#7
Design of a sequential circuit

Digital design principles.

Teacher:
Rogelio Hernández Hernández

Work made by:
Christian Aaron Ortega Blanco

Hermosillo, Son. 12 de mayo de 2022.

The state diagram present on figure 1 can be represented as a D flipflop circuit, where each state represents the output of 2 flip-flops, and the output is just the combinational result of one state.

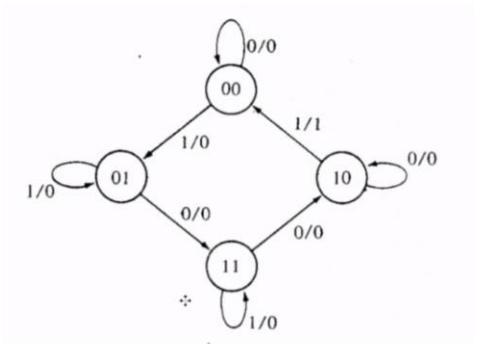
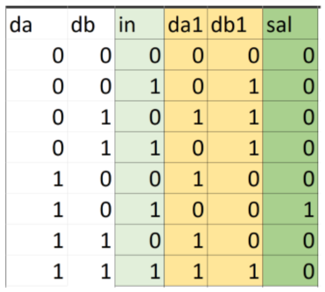


Figure 1 State diagram

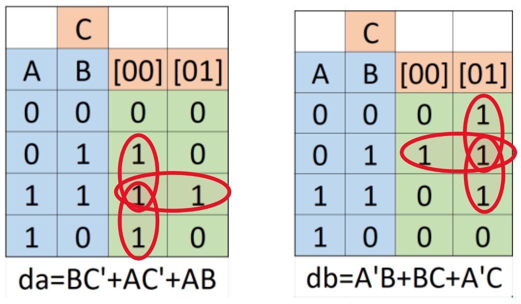
Since we have 3 bits to represent the inputs and outputs, the next state table, where in is the input and da, db are states, needs all the possible combinations in order to obtain the Boolean expressions.



da	db	in	da1	db1	sal
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	1	1	0
0	1	1	0	1	0
1	0	0	1	0	0
1	0	1	0	0	1
1	1	0	1	0	0
1	1	1	1	1	0

Figure 2 next state table

from the next state table, the Boolean expressions were obtained assuming that each da1 and db1 are the flipflop outputs and the next state is equal to the design table of the DFlip-flop.



	C		
A	B	[00]	[01]
0	0	0	0
0	1	1	0
1	1	1	1
1	0	1	0
da=BC'+AC'+AB			

a)

	C		
A	B	[00]	[01]
0	0	0	1
0	1	1	1
1	1	0	1
1	0	0	0
db=A'B+BC+A'C			

b)

Figure 3 a) da expression b) db expression.

The next step is drawing the circuit and test it. using the program Proteus its possible to draw and simulate without problems.

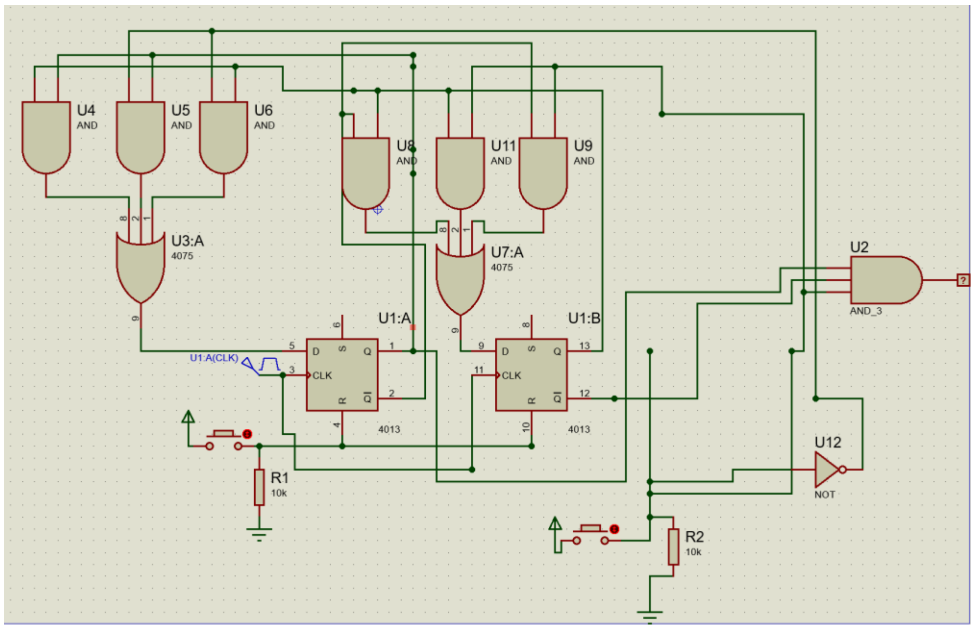


Figure 4 Circuit

Then if every state is implemented correctly, the circuit is converted into a Verilog scrip as bellow;

Module definition

```
module state(clk,rst,in,count);
  input rst,clk,in;
  output [2:0] count;
  wire n1,n2,n3,n4,n5,s,DA,DB,DC,QA,QB,QC,QANEG,QBNEG,QCNEG;


  Dflipflop FFA(clk,rst,DA,QA,QANEG);
  Dflipflop FFB(clk,rst,DB,QB,QBNEG);

  And_gate out(QA,QBNEG,in,s);

  Or_gate U1(n1,n2,n3,DA);
  And_gate_2 u2(QB,~in,n1);
  And_gate_2 u3(QA,~in,n2);
  And_gate_2 u4(QA,QB,n3);

  Or_gate U5(n4,n5,n6,DB);
  And_gate_2 u6(QANEG,QB,n4);
  And_gate_2 u7(QB,in,n5);
  And_gate_2 u8(QANEG,in,n6);



  assign count = {QA,QB,s};
```

Bench test

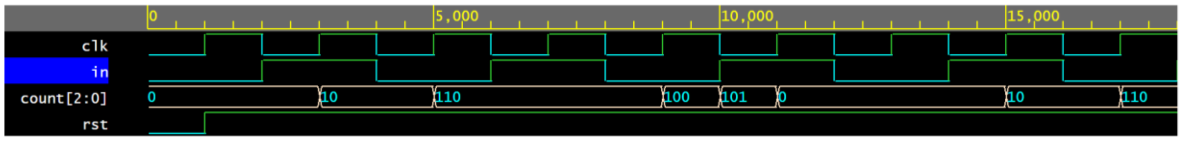
```
`timescale 1ns/1ps
`include "And_gate_2.sv"
`include "And_gate.sv"
`include "Or_gate_2.sv"
`include "Or_gate.sv"
`include "Dflipflop.sv"
module state_TB;
  reg clk,rst,in;
  wire [2:0] count;

  state UUT(clk,rst,in,count);
  initial
    begin
      $dumpfile("state.vcd");
      $dumpvars(1,state_TB);
      $display("clk,rst,count");
      $monitor ("%b %b",in, count);

      rst=1'b0;
      clk=1'b0;
      #1;
      rst=1'b1;


      for(int i=0;i<17;i++)begin
        /*if (clk) begin $display("%0b %0b %b",clk,rst,count);
        end*/
        clk=~clk;
        #1;

      end

      $finish;
    end

  initial begin
    in=1'b1;
        for(int i=0;i<17;i++)begin

        in=~in;

        #2;

      end
  end
  endmodule;
```

D- flipflop module

```
module Dflipflop(input clk,input rst, input D,output reg Q,output
reg Qneg);
  always_ff @(posedge clk , rst) begin
    if(~rst) begin
       Q <= 1'b0;
       Qneg <= 1'b1;
    end else begin
       Q <= D ;
       Qneg <= ~D ;
    end
  end
endmodule
```

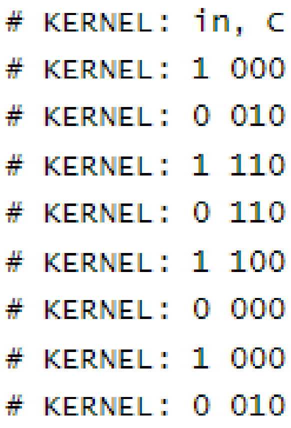
Results

Running the scrib, the resulting signals are as the follow chart 1.



1 State results

One way to analyze the results of the Circuit, is viewing the value of the input in every high pulse of the clock ("clk"), and compare it with the state diagram. The output and the state are described on the "Count" array in the form [da,db,output]. Other form is displaying the states and the output of every input combination as bellow.



```
# KERNEL: in, C
# KERNEL: 1 000
# KERNEL: 0 010
# KERNEL: 1 110
# KERNEL: 0 110
# KERNEL: 1 100
# KERNEL: 0 000
# KERNEL: 1 000
# KERNEL: 0 010
```

Figure 5 States of every positive edge-clock