

Laboratorio 06

Ejercicio 01

Bryan Jose Castillo Alvarado, 19710

Laboratorio #6

Ejercicio .01

(1)

(2)

S	A	B	S'
S ₀	0	X	S ₀
S ₀	1	X	S ₁
S ₁	X	0	S ₀
S ₁	X	1	S ₂
S ₂	1	1	S ₂
S ₂	0	X	S ₀
S ₂	X	0	S ₀

(3)

S	A	B	S'	S'	Y
0	0	X	0	0	0
0	0	1	0	1	0
0	1	X	0	0	0
0	1	1	0	1	0
1	0	X	1	0	1
1	0	1	1	0	1
1	1	X	1	1	0
1	1	1	1	1	0

Logic Friday y Ecuaciones booleanas

Logic Friday

File Operation Truthtable Equation Gates View Help

Function	Inputs	Outputs	True	False	DC	PI	Gates
N1-Y	4	3	3, 2, 1	9, 10, ...	4, 4, 4	3	8

S1	S0	A	B	=>	N1	N0	Y
1	X	1	1		1		1
X	1	X	1			1	
0	0	1	X				1

Minimized:

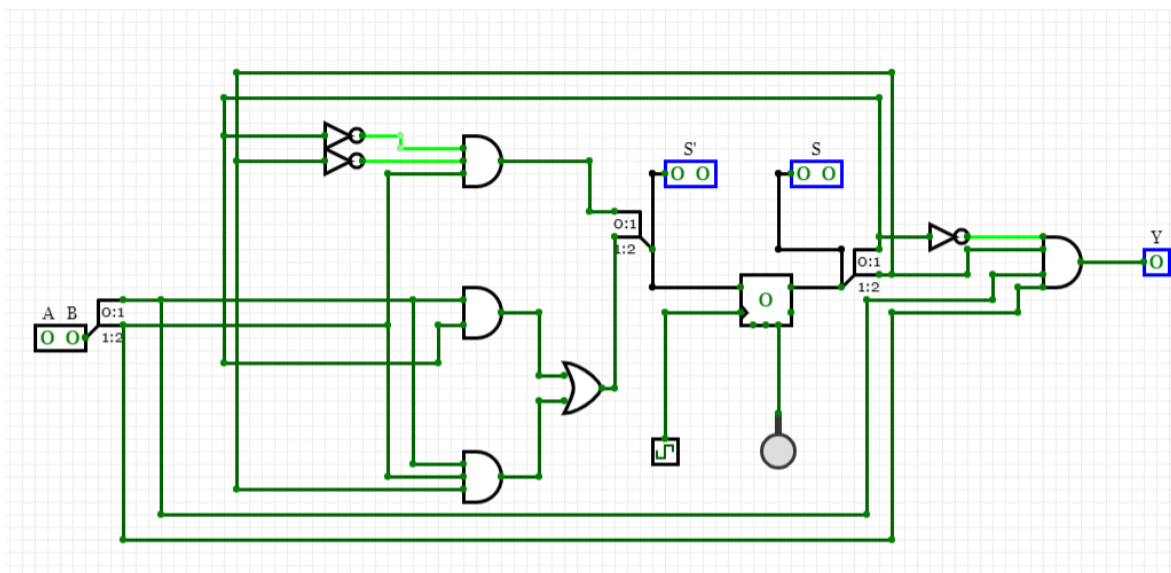
N1 = S1 A B + S0 B;

N0 = S1' S0' A B' + S1' S0' A B;

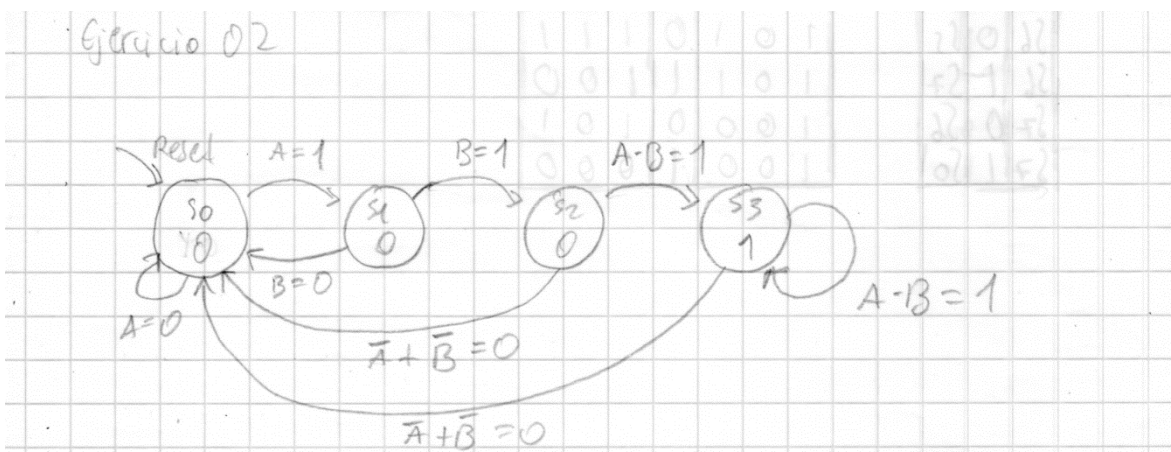
Y = S1 S0' A B;

Logic Diagram:

Circuitverse

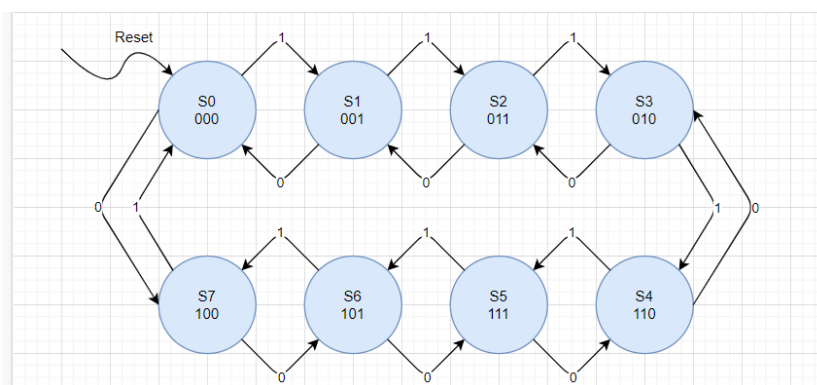


Ejercicio 02



Ejercicio 03


Diagrama



Caja negra, tablas sin codificar y tablas codificadas

Problema 03

clk

A -  - Y

Reset

S	A	S'
S0	0	S7
S0	1	S1
S1	0	S0
S1	1	S2
S2	0	S1
S2	1	S3
S3	0	S2
S3	1	S4
S4	0	S3
S4	1	S5
S5	0	S4
S5	1	S6
S6	0	S5
S6	1	S7
S7	0	S6
S7	1	S0

S	A	S'
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

S	S'	S0	Y0	X	Y2
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	1
0	1	0	0	1	0
1	0	0	1	1	0
1	0	1	1	1	1
1	1	0	1	0	1
1	1	0	1	0	0

Logic Friday y Ecuaciones booleanas

Logic Friday

File Operation Truthtable Equation Gates View Help

Funci... Inputs Outputs True False DC PI Gates

Y0-Y2 3 3 4, 4, 4 4, 4, 4 0, 0, 0 3 Not mapped

S2	S1	S0	=>	Y0	Y1	Y2
X	X	1				1
X	1	X			1	
1	X	X		1		

Imported from file:

Y0 = S2 S1' S0' + S2 S1' S0 + S2 S1 S0' + S2 S1 S0;

Y1 = S2' S1 S0' + S2' S1 S0 + S2 S1 S0' + S2 S1 S0;

Y2 = S2' S1' S0 + S2' S1 S0 + S2 S1' S0 + S2 S1 S0;

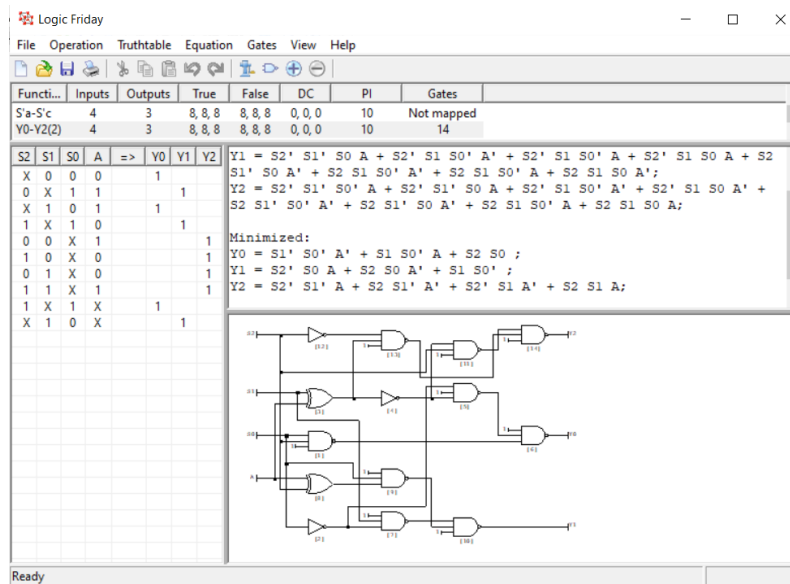
Minimized:

Y0 = S2 ;

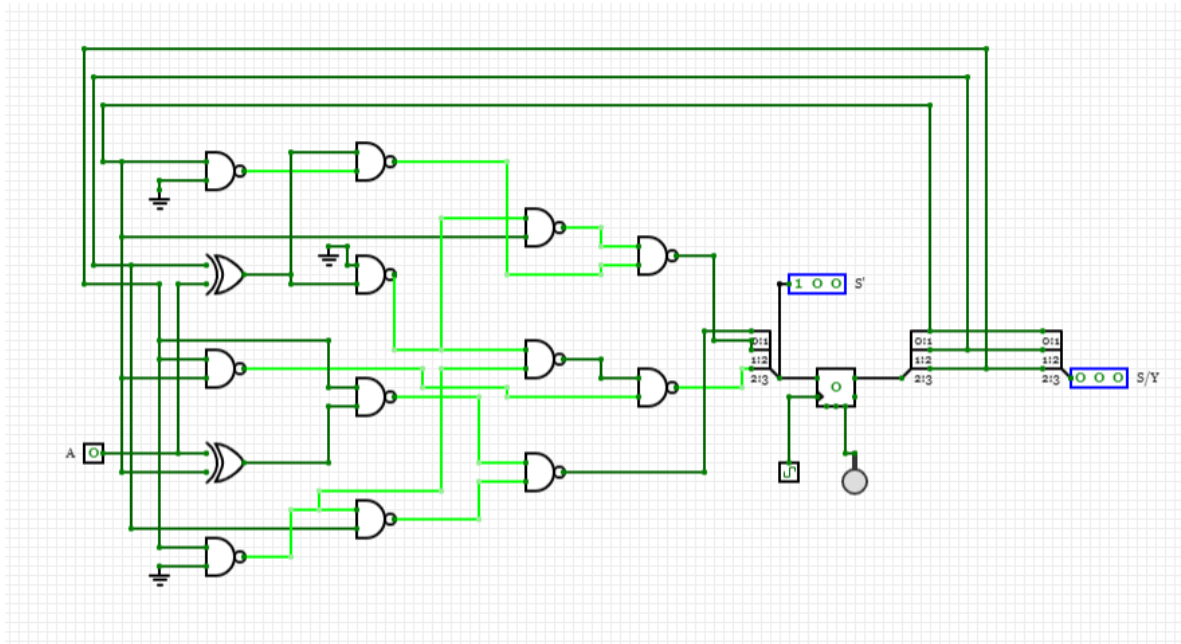
Y1 = S1 ;

Y2 = S0;

Ready



Circuitverse



Ejercicio 04

El non-blocking y blocking assignment sirven para emular nuestro circuito en la realidad debido a que puede funcionar en la simulación, pero puede tener un hardware incorrecto. La diferencia entre el non-blocking assignment y el blocking assignment es que el primero se utiliza para lógica secuencial, ya que se ejecuta en los tiempos que la persona programa, mientras que el blocking se usa para lógica combinacional donde todas las funciones se ejecutan en paralelo.

Ejercicio 05

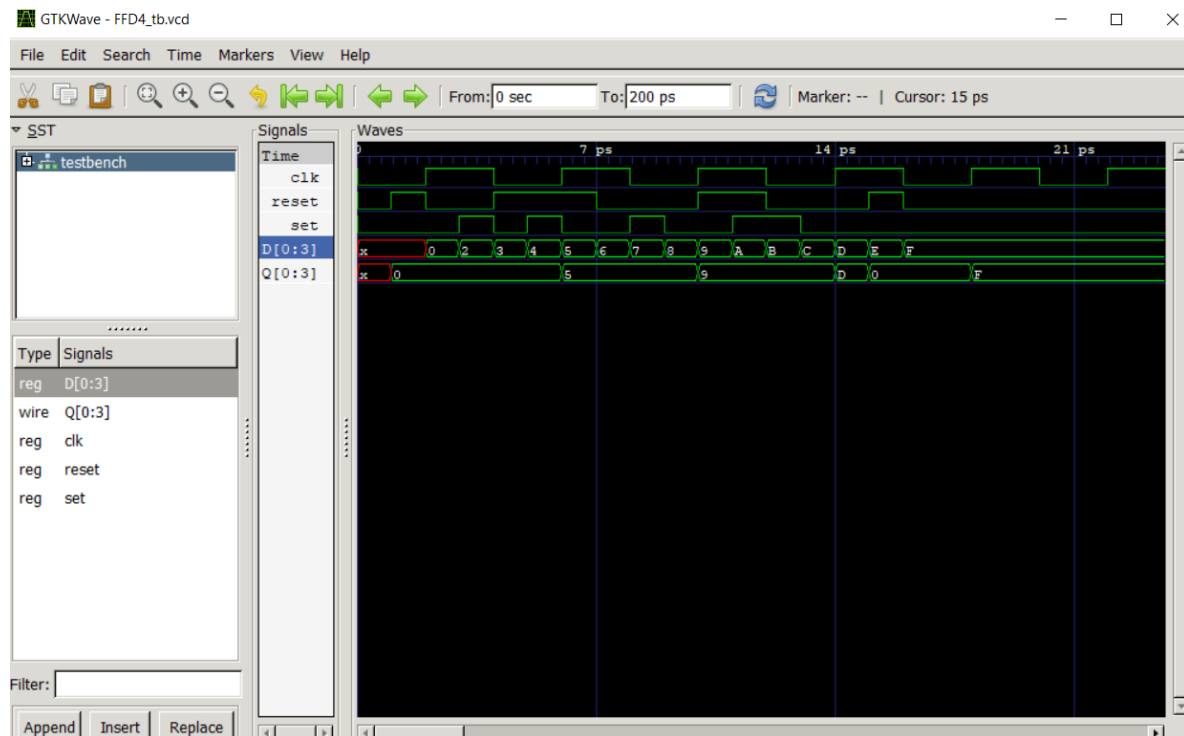
Código archivo.v

```
1 // Ejercicio05 Flip Flop tipo D de 4 bits
2
3 module FFD4A(input clk , input reset, input set, input [3:0]d, output reg[3:0]q);
4
5     always@(posedge clk, posedge reset)
6     if(reset)q<=4'b0000;
7     else    q<=d;
8
9     always@(posedge clk)
10    if(set)q<=4'b1111;
11    else   q<=d;
12 endmodule
13
14
```

Código archivo_tb.v

```
1 module testbench();
2     reg clk, reset, set;
3     reg [0:3]D;
4     wire [0:3]Q;
5
6     FFD4A G1(clk, reset, set, D, Q);
7
8     initial begin
9         clk = 0;
10        reset = 0;
11        set = 0;
12
13        #1 clk = 0; reset = 1;
14        #1 reset = 0; D = 1; D = 0;
15        #1 set = 1; D = 2;
16        #1 D = 3; reset = 1; set = 0;
17        #1 D = 4; set = 1;
18        #1 D = 5; set = 0;
19        #1 D = 6; reset = 0;
20        #1 D = 7; set = 1;
21        #1 D = 8; set = 0;
22        #1 D = 9; reset = 1;
23        #1 D = 10; set = 1;
24        #1 D = 11; reset = 0;
25        #1 D = 12; set = 0;
26        #1 D = 13;
27        #1 D = 14; reset = 1;
28        #1 D = 15; reset = 0;
29    end
30    always
31    #2 clk = ~clk;
32    initial
33    #200 $finish;
34    initial begin
35        $dumpfile("FFD4_tb.vcd");
36        $dumpvars(0, testbench);
37    end
38
39 endmodule
```

Diagrama de Timing



Ejercicio 06

Ejercicio 06-01

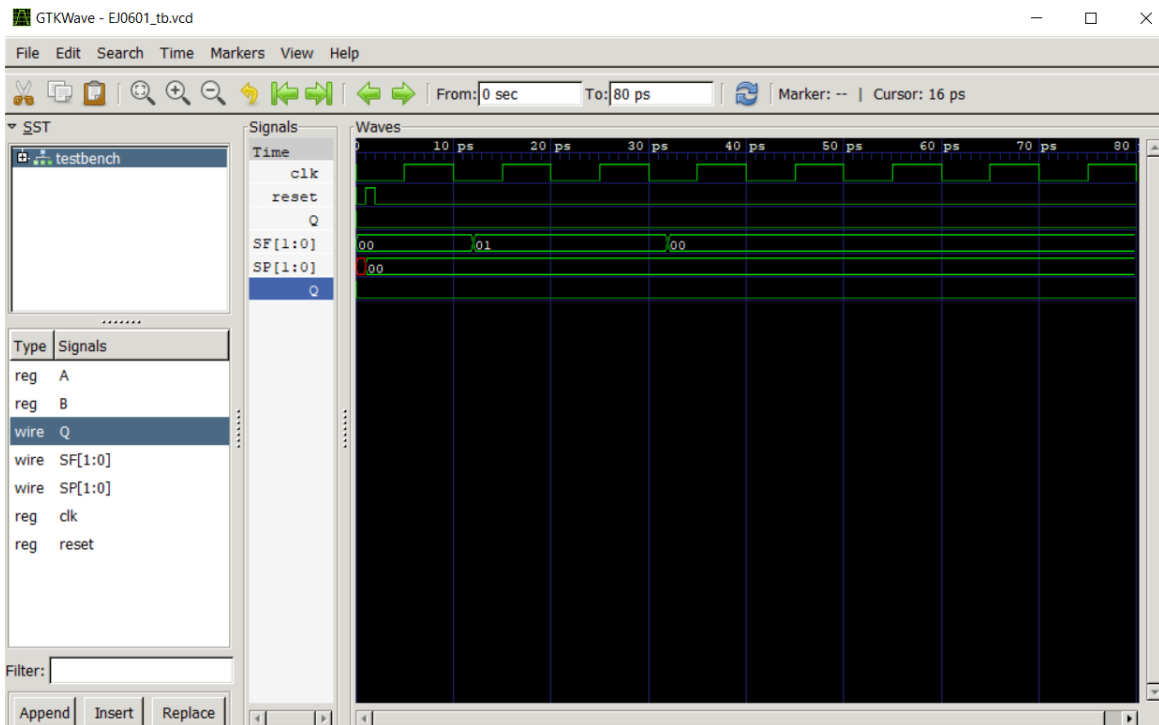
Archivo .v

```
1 //Flip Flop
2 module FF (input clk, reset, D, output reg Q);
3     always @ (posedge clk or posedge reset)begin
4         if (reset)
5             Q <= 1'b0;
6         else
7             Q <= D;
8         end
9     endmodule
10
11 //Ejercicio 01 en Verilog
12 module EJ01(input clk, reset, A, B, output wire Q, output wire [1:0]SF, SP);
13     //wire SP0, SP1, SF0, SF1;
14
15     assign SF[0] = (~SP[1] & ~SP[0] & A);
16     assign SF[1] = (SP[0] & B) | (SP[1] & A & B);
17     assign Q = (SP[1] & ~SP[0] & A & B);
18
19     FF M1(.clk(clk), .reset(reset), .D(SF[1]), .Q(SP[1]));
20     FF M0(.clk(clk), .reset(reset), .D(SF[0]), .Q(SP[0]));
21 endmodule
22
```

Archivo _tb.v

```
1 module testbench();
2     reg clk, reset, A, B;
3     wire Q;
4     wire [1:0] SF, SP;
5     EJ01 D1(clk, reset, A, B, Q, SF, SP);
6
7     initial begin
8         $display("\n");
9         $display(" Ejercicio 01");
10        $display("clk reset A B SF SP | Y");
11        $display("-----|---");
12        $monitor("%b %b %b %b %b %b | %b", clk, reset, A, B, SF, SP, Q);
13    end
14
15    initial begin
16        clk = 0;
17        reset = 0;
18        A = 0;
19        B = 0;
20        #1 reset = 1;
21        #1 reset = 0;
22        #10
23        A = 1;
24        B = 0;
25        #20
26        A = 0;
27        B = 0;
28    end
29
30    always
31        #5 clk = ~clk;
32
33    initial
34        #80 $finish;
35
36    initial begin
37        $dumpfile("EJ0601_tb.vcd");
38        $dumpvars(0, testbench);
39    end
40
41 endmodule
42
```

Diagrama de Timing



Ejercicio 06-01

Archivo .v

```
1 //Flip Flop
2 module FF (input clk, reset, D, output reg Q);
3     always @ (posedge clk or posedge reset)begin
4         if (reset)
5             Q <= 1'b0;
6         else
7             Q <= D;
8         end
9     endmodule
10
11 //Ejercicio 03
12 module EJ03(input A, clk, reset, output wire Y1, Y2, Y3);
13     wire S0, S1, S2, S00, S11, S22;
14
15     assign S00 = (~S1 & ~S0 & ~A) | (S1 & ~S0 & A) | (S2 & S0);
16     assign S01 = (~S2 & S0 & A) | (S2 & S0 & ~A) | (S1 & ~S0);
17     assign S22 = (~S2 & ~S1 & A) | (S2 & ~S1 & ~A) | (~S2 & S1 & ~A) | (S2 & S1 & A);
18
19     FF U1(clk, reset, S00, S0);
20     FF U2(clk, reset, S11, S1);
21     FF U3(clk, reset, S22, S2);
22
23     assign Y1 = S0;
24     assign Y2 = S1;
25     assign Y3 = S2;
26
27 endmodule
```

Archivo_tb.v

```
1 module testbench();
2     reg clk, reset, A;
3     wire Y1, Y2, Y3;
4     EJ03 D1(A, clk, reset, Y1, Y2, Y3);
5
6     initial begin
7         $display("\n");
8         $display(" Ejercicio 01");
9         $display("clk reset A B SF SP | Y");
10        $display("-----|---");
11        $monitor("%b %b %b %b %b %b | %b", clk, reset, A, Y1, Y2, Y3);
12    end
13
14    initial begin
15        clk = 0;
16        reset = 0;
17        A = 0;
18        #1 reset = 1;
19        #1 reset = 0;
20        #10
21        A = 1;
22        #20
23        A = 0;
24        #30
25        A = 1;
26        #40
27        A = 0;
28        #50
29        A = 1;
30        #60
31        A = 0;
32        #70
33        A = 1;
34    end
35    always
36        #5 clk = ~clk;
37    initial
38        #150 $finish;
39    initial begin
40        $dumpfile("EJ0603_tb.vcd");
41        $dumpvars(0, testbench);
42    end
43
44 endmodule
```


Diagrama de Timing

