Laboratorio 06

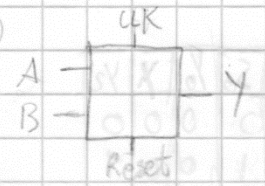
Ejercicio 01

Brayan José Castillo Alvarado, 19780

Laboratorio #6

Ejercicio .01

(1)



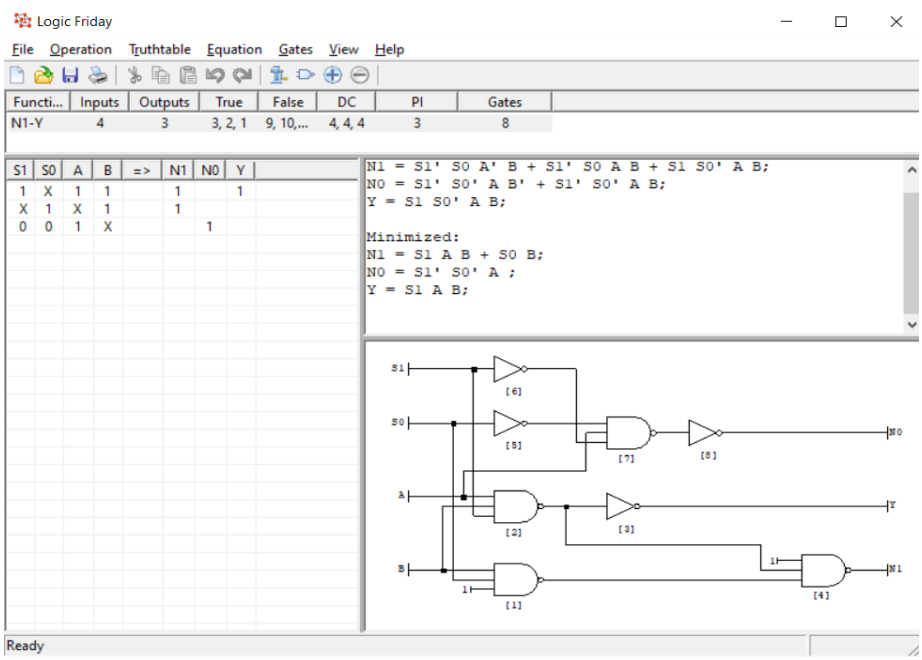
(2)

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

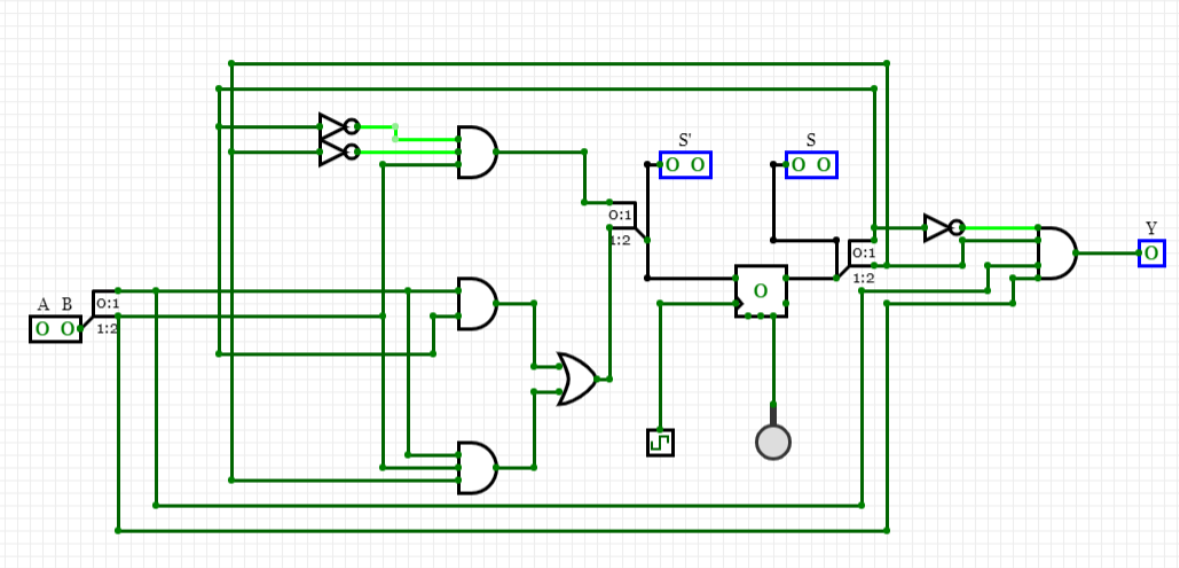
(3)

S_1	S_0	A	B	S_1'	S_0'	Y
0	0	0	X	0	0	0
0	0	1	X	0	1	0
0	1	X	0	0	0	0
0	1	X	1	1	0	0
1	0	1	1	1	0	1
1	0	0	X	0	0	0
1	0	X	0	0	0	0

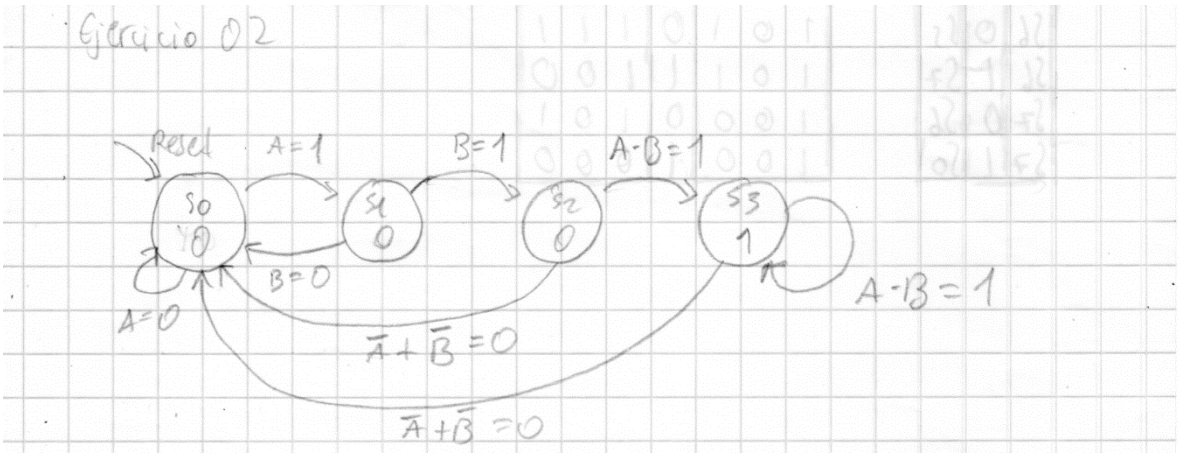
Logic Friday y Ecuaciones booleanas



Circuitverse

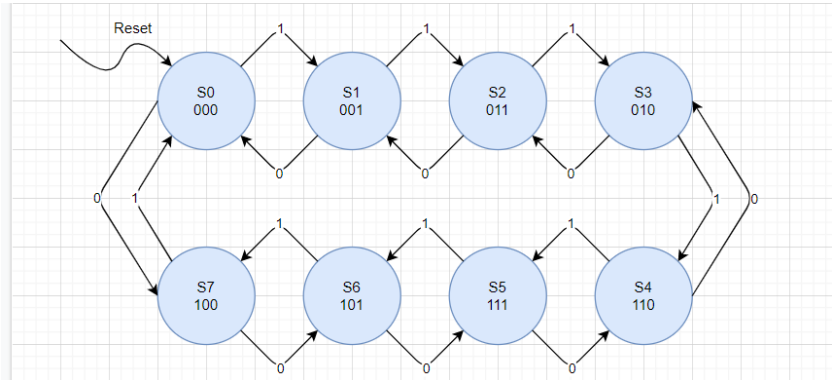


Ejercicio 02

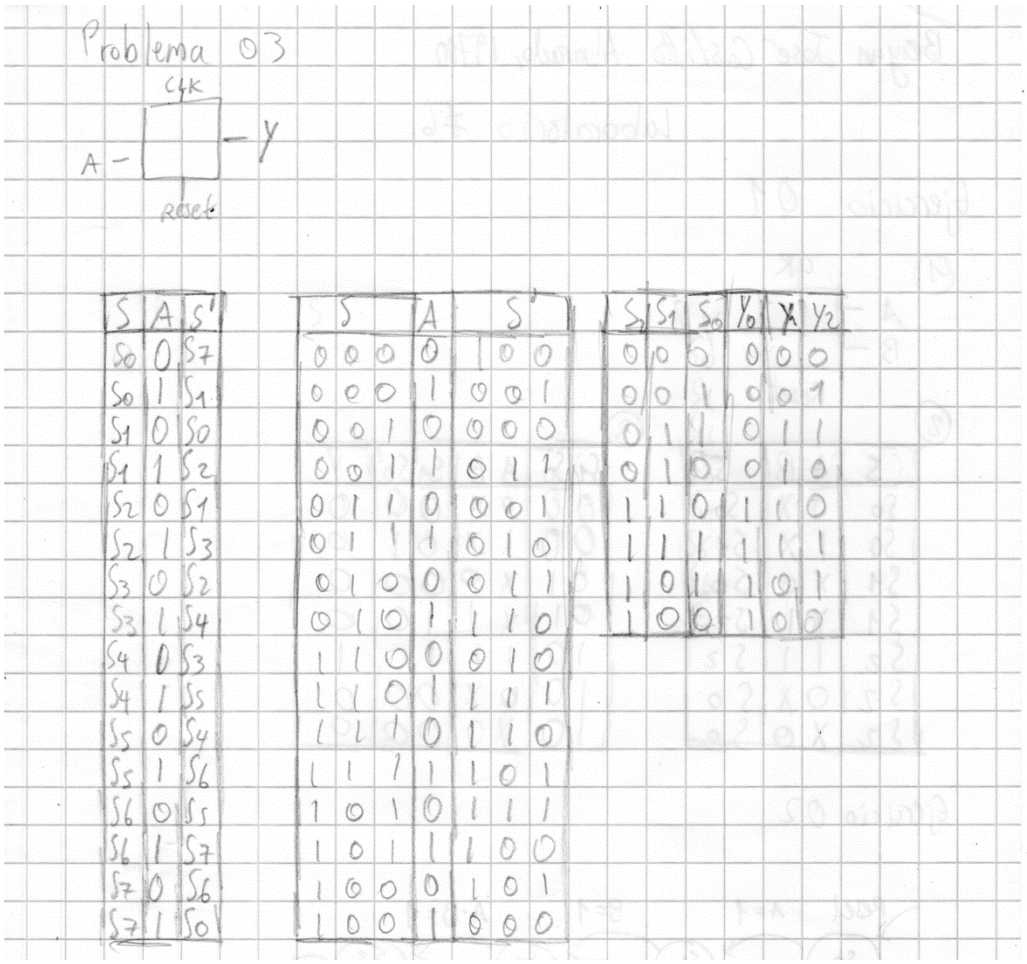


Ejercicio 03

Diagrama



Caja negra, tablas sin codificar y tablas codificadas

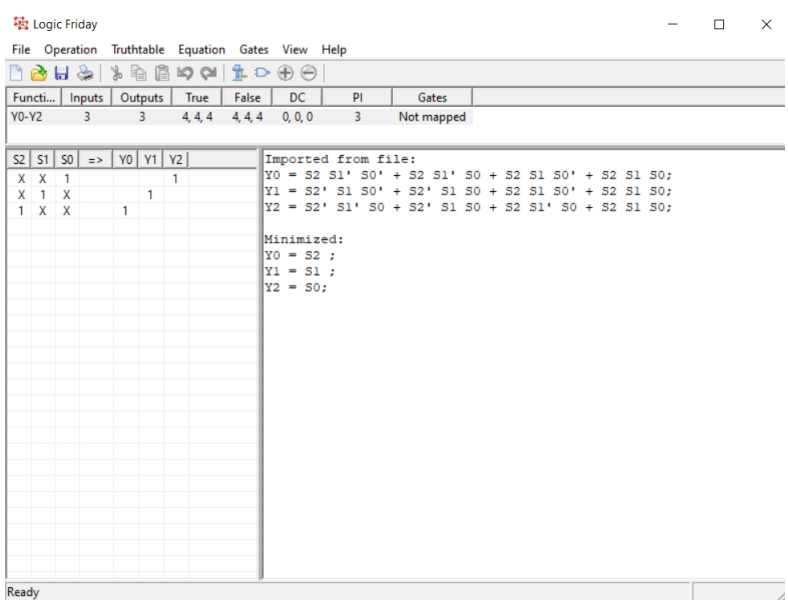


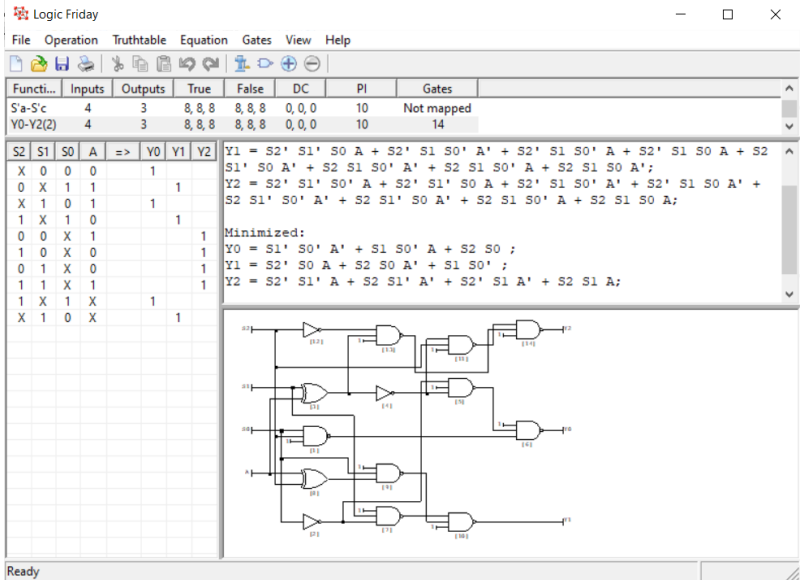
S	A	S'
S_0	0	S_7
S_0	1	S_1
S_1	0	S_0
S_1	1	S_2
S_2	0	S_1
S_2	1	S_3
S_3	0	S_2
S_3	1	S_4
S_4	0	S_3
S_4	1	S_5
S_5	0	S_4
S_5	1	S_6
S_6	0	S_5
S_6	1	S_7
S_7	0	S_6
S_7	1	S_0

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

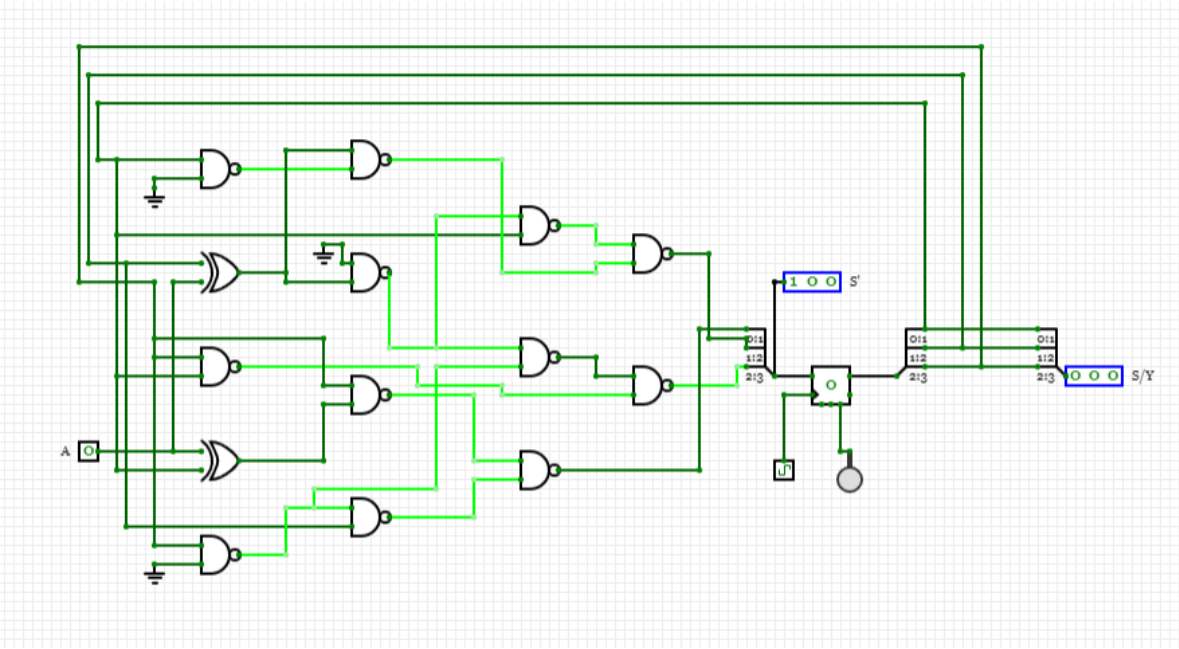
S_2	S_1	S_0	Y_0	Y_1	Y_2
0	0	0	0	0	0
0	0	1	0	0	1
0	1	1	0	1	1
0	1	0	0	1	0
1	1	0	1	1	0
1	1	1	1	1	1
1	0	1	1	0	1
1	0	0	1	0	0

Logic Friday y Ecuaciones booleanas





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 assingment y el blocking assignmente 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

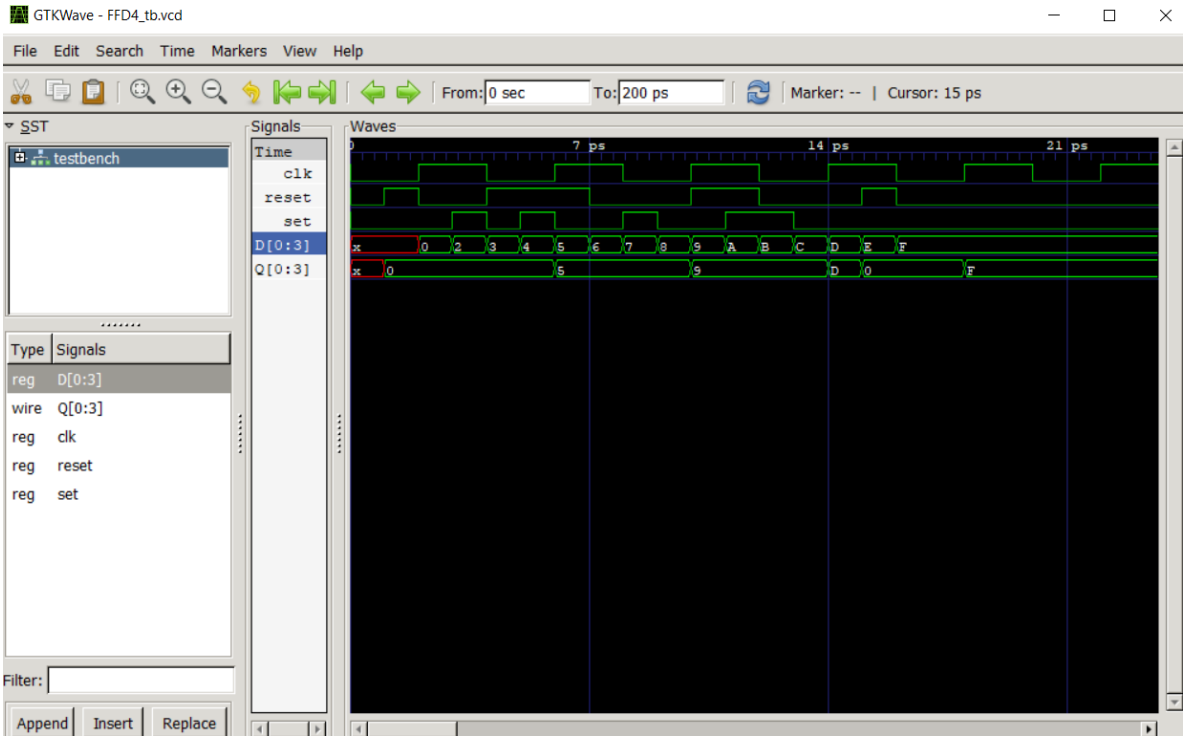
Código archivo.v

```
// Ejercicio05 Flip Flop tipo D de 4 bits

module FFD4A(input clk , input reset, input set, input [3:0]d, output reg[3:0]q);

    always@(posedge clk, posedge reset)
    if(reset)q<=4'b0000;
    else     q<=d;

    always@(posedge clk)
    if(set)q<=4'b1111;
    else     q<=d;
endmodule
```

Código archivo _tb.v

```
module testbench();
    reg clk, reset, set;
    reg [0:3]D;
    wire [0:3]Q;

    FFD4A G1(clk, reset, set, D, Q);

    initial begin
    clk = 0;
    reset = 0;
    set = 0;

    #1 clk = 0; reset = 1;
    #1 reset = 0;D = 1;D = 0;
    #1 set = 1; D = 2;
    #1 D = 3;reset = 1; set = 0;
    #1 D = 4;set = 1;
    #1 D = 5;set = 0;
    #1 D = 6; reset = 0;
    #1 D = 7;set = 1;
    #1 D = 8;set = 0;
    #1 D = 9;reset = 1;
    #1 D = 10;set = 1;
    #1 D = 11;reset = 0;
    #1 D = 12;set = 0;
    #1 D = 13;
    #1 D = 14;reset = 1;
    #1 D = 15;reset = 0;
    end
    always
    #2 clk = ~clk;
    initial
    #200 $finish;
    initial begin
    $dumpfile("FFD4_tb.vcd");
    $dumpvars(0, testbench);
    end

endmodule
```

Diagrama de Timing



Ejercicio 06

Ejercicio 06-01

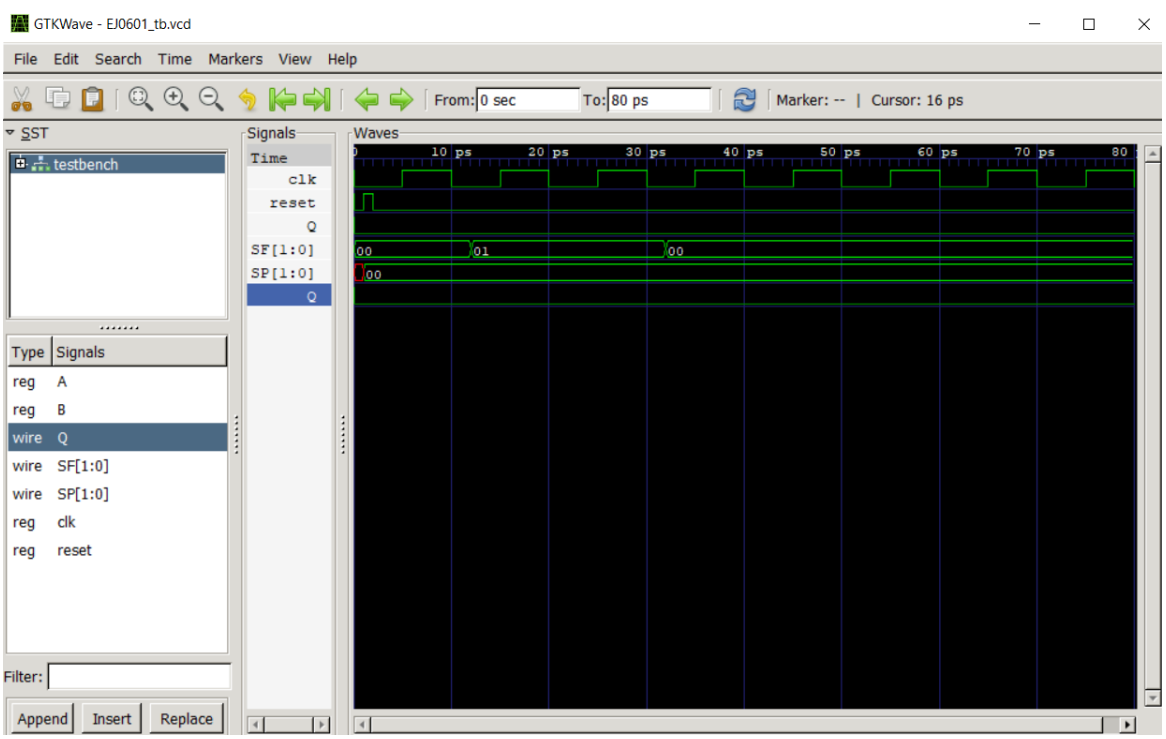
Archivo .v

```
//Flip Flop
module FF (input clk, reset, D, output reg Q);
    always @ (posedge clk or posedge reset)begin
    if (reset)
        Q <= 1'b0;
    else
        Q <= D;
    end
endmodule

//Ejercicio 01 en Verilog
module EJ01(input clck, reset, A, B, output wire Q, output wire [1:0]SF, SP);
        //wire SP0, SP1, SF0, SF1;

        assign SF[0] = (~SP[1] & ~SP[0] & A);
        assign SF[1] = (SP[0] & B) | (SP[1] & A & B);
        assign Q = (SP[1] & ~SP[0] & A & B);

            FF M1(.clk(clk), .reset(reset), .D(SF[1]), .Q(SP[1]));
            FF M0(.clk(clk), .reset(reset), .D(SF[0]), .Q(SP[0]));
endmodule
```

Archivo _tb.v

```
module testbench();
    reg clk, reset, A, B;
    wire Q;
    wire [1:0]SF, SP;
    EJ01 D1(clk, reset, A, B, Q, SF, SP);

    initial begin
    $display("\n");
        $display(" Ejercicio 01");
    $display("clk reset A B SF SP | Y");
    $display("--------------------|--");
    $monitor("%b %b %b %b %b %b | %b", clk, reset, A, B, SF, SP, Q);
    end

    initial begin
        clk = 0;
        reset = 0;
        A = 0;
        B = 0;
        #1 reset = 1;
        #1 reset = 0;
        #10
        A = 1;
        B = 0;
        #20
        A = 0;
        B = 0;
    end

    always
    #5 clk = ~clk;

    initial
    #80 $finish;

    initial begin
    $dumpfile("EJ0601_tb.vcd");
    $dumpvars(0, testbench);
    end

endmodule
```

Diagrama de Timing



Ejercicio 06-01

Archivo .v

```
//Flip Flop
module FF (input clk, reset, D, output reg Q);
    always @ (posedge clk or posedge reset)begin
    if (reset)
        Q <= 1'b0;
    else
        Q <= D;
    end
endmodule

//Ejercicio 03
module EJ03(input A, clk, reset, output wire Y1, Y2, Y3);
    wire S0, S1, S2, S00, S11, S22;

    assign S00 = (~S1 & ~S0 & ~A) | (S1 & ~S0 & A) | (S2 & S0);
    assign S01 = (~S2 & S0 & A) | (S2 & S0 & ~A) | (S1 & ~S0);
    assign S22 = (~S2 & ~S1 & A) | (S2 & ~S1 & ~A) | (~S2 & S1 & ~A) | (S2 & S1 & A);

    FF U1(clk, reset, S00, S0);
    FF U2(clk, reset, S11, S1);
    FF U3(clk, reset, S22, S2);

    assign Y1 = S0;
    assign Y2 = S1;
    assign Y3 = S2;

endmodule
```

Archivo_tb.v

```
module testbench();
    reg clk, reset, A;
    wire Y1, Y2, Y3;
    EJ03 D1(A, clk, reset, Y1, Y2, Y3);

    initial begin
    $display("\n");
    $display(" Ejercicio 01");
    $display("clk reset A B SF SP | Y");
    $display("--------------------|--");
    $monitor("%b %b %b %b %b %b | %b", clk, reset, A, Y1, Y2, Y3);
    end

    initial begin
    clk = 0;
    reset = 0;
    A = 0;
    #1 reset = 1;
    #1 reset = 0;
    #10
    A = 1;
    #20
    A = 0;
    #30
    A = 1;
    #40
    A = 0;
    #50
    A = 1;
    #60
    A = 0;
    #70
    A = 1;
    end
    always
    #5 clk = ~clk;
    initial
    #150 $finish;
    initial begin
    $dumpfile("EJ0603_tb.vcd");
    $dumpvars(0, testbench);
    end

endmodule
```

Diagrama de Timing