GESTION DE ESTACIONAMIENTO

Alumnos
Agustin Barbero
Hugo Junior Albarenque

ETAPAS

01 Diagrama de estados FSM entrada/salida

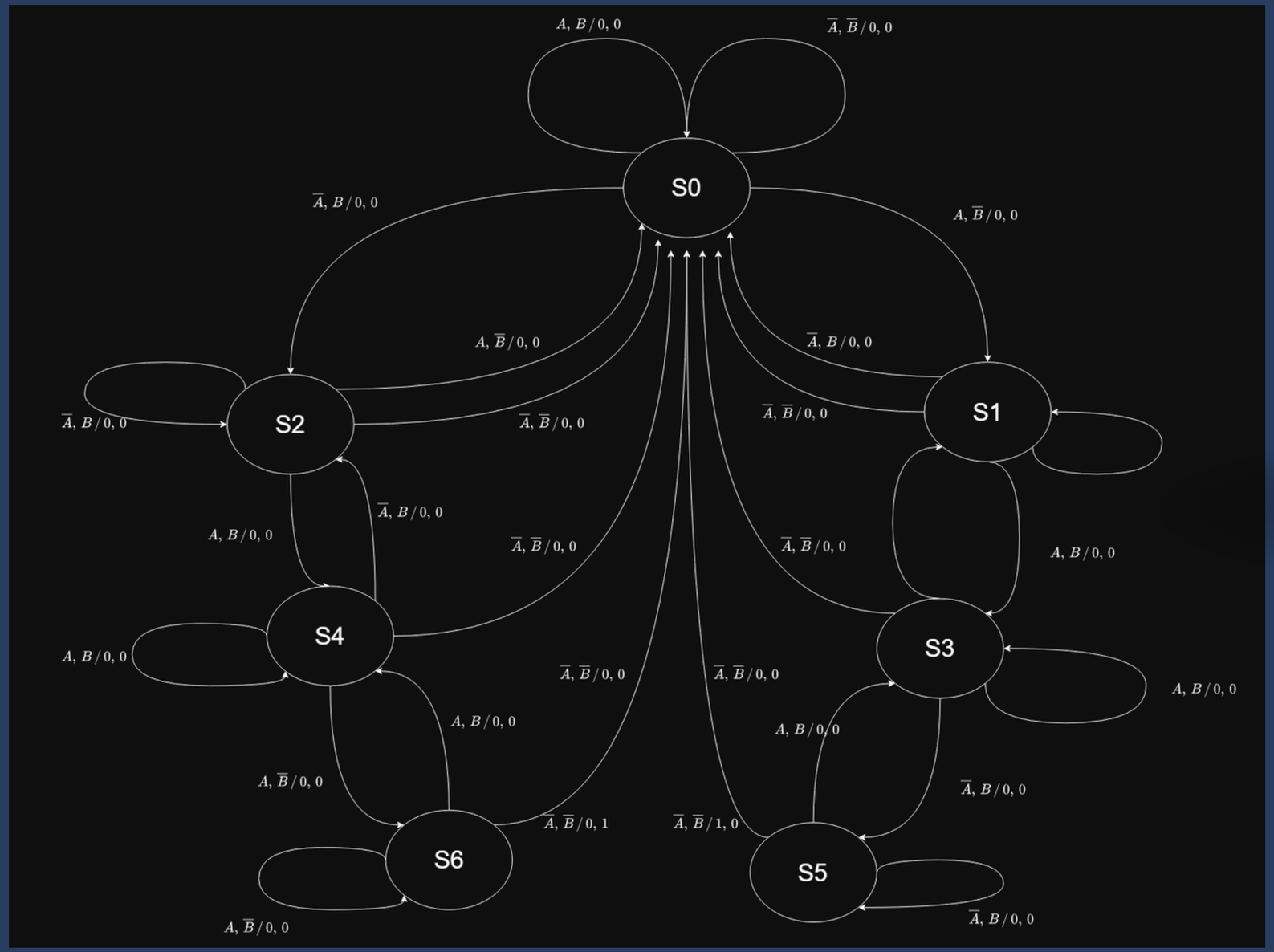
02 Implementación FSM entrada/salida

03 Diagrama de estados FSM contador

04 Modulo principal

05 Modulo antirrebote

FSM ENTRADA/ SALIDA



- 7 ESTADOS
- 2 ENTRADAS
- 2 SALIDAS

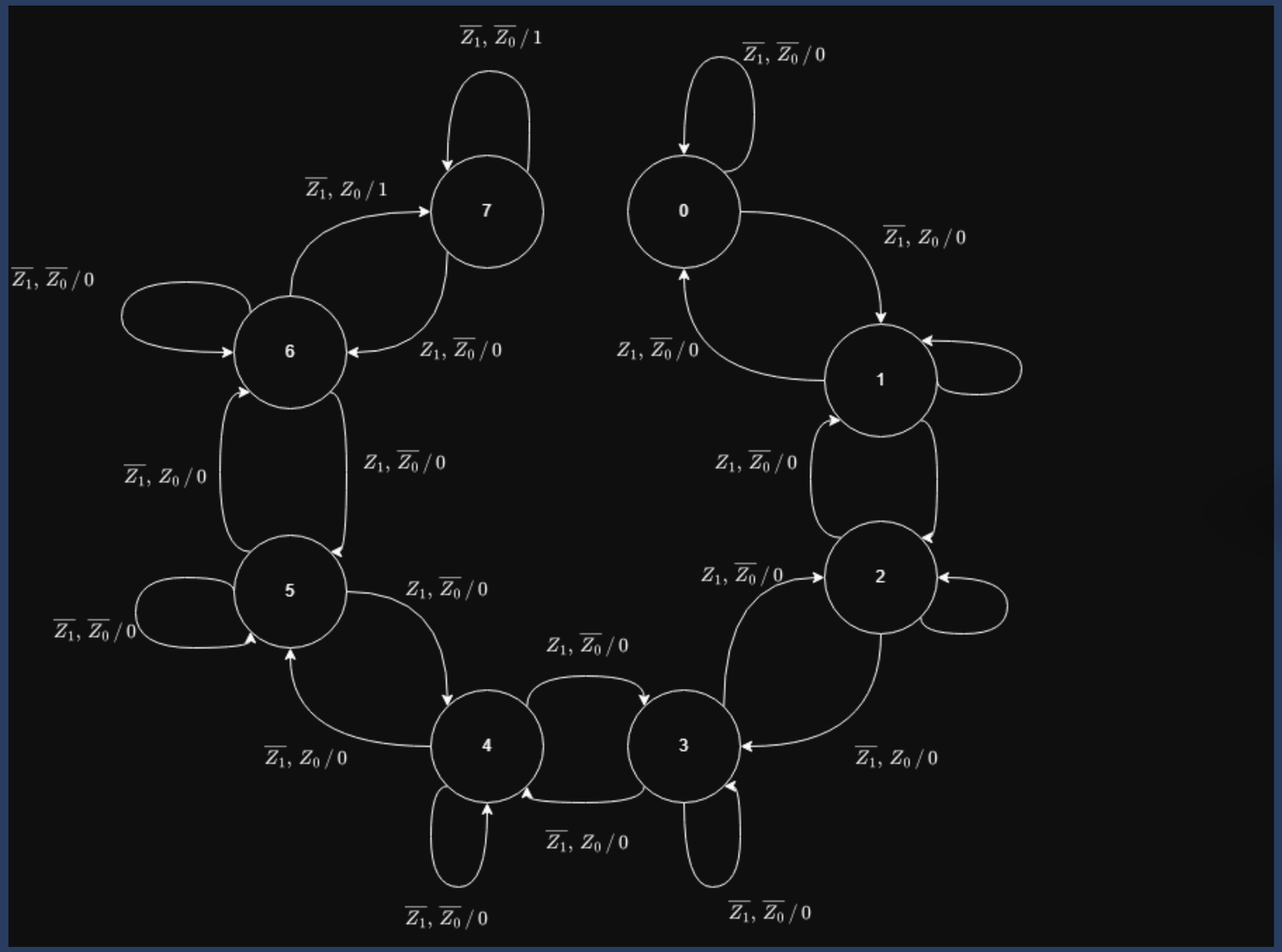
Implementacion de modulo entrada/salida

```
module entrada_salida(
    input clk,
    input wire A,
    input wire B,
    output wire [2:0]estado,
    output wire [1:0]salida
);

ffD flipflipD0(
    .D(D0),
    .Q(estado[0]),
    .Clk(clk)
);

ffD flipflipD1(
    .D(D1),
    .Q(estado[1]),
    .Clk(clk)
);

ffD flipflipD2(
    .D(D2),
    .Q(estado[2]),
    .Clk(clk)
);
```

```
wire Q0 = estado[0];
wire Q1 = estado[1];
wire Q2 = estado[2];
assign D2 = (~A & B & ~Q2 & Q1 & Q0) |
            ( A & B & Q1 & ~Q0) |
            ( A & Q2 & Q1 & ~Q0) |
            (~A & B & Q2 & ~Q1 & Q0) |
            ( A & Q2 & ~Q1 & ~Q0);

assign D1 = ( A & B & ~Q2 & Q0) |
            (~A & B & ~Q1 & ~Q0)      |
            (~A & B & ~Q2 & Q1 & ~Q0) |
            (A & ~B &  Q2 &  Q1 & ~Q0) |
            (A & ~B &  Q2 & ~Q1 & ~Q0) |
            (A & B &  Q2 & ~Q1 & Q0);

assign D0 = (B & Q2 & ~Q1 & Q0)|
            (B & ~Q2 & Q1 & Q0)|
            (A & ~Q2 & Q0)|
            (A & ~B & ~Q2 & ~Q1);

assign salida[0] = (Q0 & ~Q1 & Q2 & ~A & ~B);
assign salida[1] = (~Q0 & Q1 & Q2 & ~A & ~B);
```

Las expresiones logicas para las entradas de los flip-flops tipo D fueron obtenidas mediante mapas de Karnaugh

FSM CONTADOR



- 8 estados
- 2 entradas
- 1 salida

MODULO PRINCIPAL

```
module mostrarFPGA(
    input CLK,
    input RST,
    input wire BTN1,
    input wire BTN4,
    output wire LED0,
    output wire LED1,
    output wire LED2,
    output wire LED3
);

wire BTN2_clean;
wire BTN1_clean;
antirrebote_modulo rebote1(
    .clk(CLK),
    .btn(~BTN1),
    .btn_clean(BTN1_clean)
);
antirrebote_modulo rebote2(
    .clk(CLK),
    .btn(~BTN4),
    .btn_clean(BTN2_clean)
);
```

Al modulo antirebote le enviamos la señal de los botones negada debido a que los botones de la FPGA se encuentran conectados a una resistencia pull -UP

```
reg A = 0, B = 0;

wire [2:0]estado;
wire [1:0]salida;
entrada_salida gestion(
    .clk(CLK),
    .A(BTN1_clean),
    .B(BTN2_clean),
    .estado(estado),
    .salida(salida)
);

wire [2:0] num;
wire lleno;
contador cuenta(
    .clk(CLK),
    .reset(RST),
    .z1(salida[0]),
    .z2(salida[1]),
    .c(num),
    .lleno(lleno)
);
assign LED0 = num[0];
assign LED1 = num[1];
assign LED2 = num[2];
assign LED3 = lleno;
```

MODULO ANTIRREBOTE

```v
module antirrebote_modulo (
    input wire clk,
    input wire btn,     // entrada con rebotes
    output reg btn_clean     // salida limpia
);
    //El rebote mecánico típico de un pulsador dura entre 5 ms y 20ms
    //Como el reloj interno de nuestra  FPGA es de 12MHz,
    //es decir, 83,33ns por ciclo, como resultado si queremos esperar 20ms
    //debemos esperar 240000 ciclos de reloj.
    parameter tiempo_antirebote = 240000; // 20ms en ciclos de reloj a 12MHz
    reg [18:0] contador = 0;  // esto cuenta el tiempo que se encuentra activada la entrada
    reg estado = 0;           // estado interno
    always @(posedge clk) begin
        if (btn != estado) begin
            // cambio detectado en la entrada, se reinicia contador
            estado <= btn;
            contador <= 0;
        end else if (contador < tiempo_antirebote) begin
            // si no hay cambio se incrementa el contador hasta alcanzar el tiempo necesario
            contador <= contador + 1;
        end else begin
            // se cumplió el tiempo, se actualiza la salida
            btn_clean <= estado;
        end
    end
endmodule
```

FIN DE LA PRESENTACIÓN