

Universidad de las Américas Puebla



Sistemas Embebidos

Tarea 2: Informe Tecnico

Emilio Iván Jiménez López 179543

Otoño 2025

Planteamiento

En esta segunda tarea, realizaremos 2 circuitos secuenciales, ambos con el propósito de realizar un corrimiento a la derecha de un registro de bits, sin embargo, ambos tendrán un funcionamiento y complejidad muy diferentes.

El primer problema tendrá entrada y salida de registros paralelos de 8 bits, deberá además incluir un input para carga/corrimiento. El segundo problema es similar, sin embargo, deberá producir un output serial, no paralela, y de 4 bits en lugar de 8. Para ello, se deberán incluir también una señal de inicio de conversión, y una señal también de data_ready para señalar la conclusión del proceso.

Ambos deberán incluir un reloj y activación en el flanco positivo del reloj, un reset asíncrono, y se deberán hacer los testbenches correspondientes.

Procedimiento

Previo a la implementación en Verilog, se realizaron trabajos de prototipado en papel, los cuales pueden verse bajo solicitud al estudiante.

La implementación del problema 1 se ve de la siguiente forma:

```
4 //Problema 1: Registro paralelo-paralelo
5 module corrimiento_paralelo(
6     input wire [7:0] data_in,      // Entrada paralela de 8 bits
7     input wire control,           // Control: 0 = carga, 1 = corrimiento
8     input wire clk,               // Señal de reloj
9     input wire rst,               // Reset asíncrono
10    output reg [7:0] data_out      // Salida de 8 bits
11);
12
13 // Activación del ciclo por flanco positivo de reloj o estímulo reset
14 always @(posedge clk or posedge rst) begin
15     if (rst) begin
16         // Reinicia la salida a cero
17         data_out <= 8'b0;
18     end else begin
19         // Selección de operación según control
20         case (control)
21             1'b0: begin
22                 // Carga directa del dato de entrada
23                 data_out <= data_in;
24             end
25             1'b1: begin
26                 // Corrimiento lógico a la derecha
27                 data_out <= data_out[7:1];
28             end
29         endcase
30     end
31 end
32 endmodule
```

La implementación del problema 2 se ve de la siguiente forma:

```
34 //Problema 2: Registro paralelo-serial
35 module registro_paralelo_serie (
36     input wire clk,           // Señal de reloj
37     input wire rst,           // Reset asíncrono
38     input wire start,         // Señal de inicio de conversión
39     input wire [3:0] data_in, // Entrada paralela de 4 bits
40     output reg data_out,       // Salida serial de 1 bit
41     output reg data_ready     // Señal de conversión completa
42 );
43
44 // Registros internos
45 reg [3:0] shift_reg;          // Registro de desplazamiento
46 reg [2:0] bit_count;          // Contador de bits enviados
47 reg sending;                  // Estado de transmisión
48
49 // Ciclo activado por flanco positivo del reloj o estímulo reset
50 always @(posedge clk or posedge rst) begin
51     if (rst) begin
52         // Reinicio de todos los registros
53         shift_reg <= 4'b0000;
54         bit_count <= 3'b000;
55         data_out <= 1'b0;
56         data_ready <= 1'b0;
57         sending <= 1'b0;
58     end else begin
59         if (start && !sending) begin
60             // Inicio de transmisión: carga de datos
61             shift_reg <= data_in;
62             bit_count <= 3'b000;
63             sending <= 1'b1;
64             data_ready <= 1'b0;
65         end else if (sending) begin
66             // Transmisión bit a bit
67             data_out <= shift_reg[bit_count];
68             bit_count <= bit_count + 1;
69
70             if (bit_count == 3) begin
71                 // Ultimo bit transmitido
72                 sending <= 1'b0;
73                 data_ready <= 1'b1;
74             end
75         end else begin
76             // Limpieza de registro de listo cuando no se transmite
77             data_ready <= 1'b0;
78         end
79     end
80 end
81 endmodule
```

La implementación de los testbenches se realizó con apoyo de herramientas de inteligencia artificial.

Esta tarea explora temas que en clase no habíamos podido revisar adecuadamente, por tanto represento una gran oportunidad para aprender bastante acerca de el lenguaje Verilog, sus capacidades, el paradigma de programación para descripción de hardware, y en lo personal, me sirvió para aplicar mis habilidades programáticas al paradigma de los lenguajes de descripción.