INF245 – Laboratorio 3
Don Bit y Bitópolis

Sebastián Richiardi Pérez – 20203055-2 – 201
Gabriel Alejandro Toro Varela – 202204557-4 – 201

02 de junio de 2025

1. Introducción

En este laboratorio se implementa un sistema digital en Logisim para simular los recorridos dentro de la ciudad digital Bitópolis. Se emplea lógica secuencial sin memorias ROM/RAM, únicamente con compuertas lógicas y flip-flops tipo D, mostrando paso a paso el trayecto en un display de 7 segmentos.

2. Descripción del problema

Don Bit construyó una ciudad digital donde los vehículos se guían por un código de 4 bits que define un nodo inicial en un grafo. Cada uno de los 16 códigos posibles activa una ruta única y secuencial hacia el nodo final común: F.

3. Diseño del sistema

La FSM consta de:
1. Un registro de 4 bits que almacena el "estado actual" (el nodo en el grafo). 2. Lógica combinacional de **próximo estado** (Next–State) que, para cada contenido actual del registro, produce los 4 bits del nodo siguiente. 3. Un decodificador a 7 segmentos que, a partir de los 4 bits del estado, activa las líneas del display para mostrar la letra del nodo (A–P). 4. Una señal de "Enable" que detiene la máquina en cuanto se alcanza el nodo final (F).

3.1. Algoritmo general

- **Entrada inicial:** El usuario ingresa un código de 4 bits (b3 b2 b1 b0), que corresponde a un número decimal entre 0 y 15. Cada valor 0–15 identifica un nodo inicial del grafo (`A`, `B`, . . . , `P`).
- **Estado inicial:** Al presionar "Reset", el registro de estado carga directamente el código de 4 bits ingresado, es decir, el nodo de partida.
- **Transiciones:** En cada flanco de subida del reloj, si Enable = 1, el registro pasa de "Estado_actual" a "Estado_siguiente" de acuerdo con la ruta predefinida. Esa ruta

se deriva del grafo de Bitópolis: cada estado (nodo) tiene exactamente un sucesor, salvo el nodo `F` (código 5), que es el "destino final" y se mantiene en sí mismo.

- **Detención:** Cuando el registro de estado contiene el código del nodo `F` (decimal 5), la señal Enable pasa a 0 y la máquina deja de avanzar (queda "retenida" en `F`).
- **Visualización:** El decodificador 7 segmentos traduce los 4 bits del "Estado_actual" a los segmentos a–g para que se muestre la letra correspondiente.

Donde:

- El bloque **Registro_Estado (4 D–FF)** guarda el nodo actual.
- El bloque **Next_State (Lógica combinacional 4 → 4)** calcula, a partir del contenido del registro, el código de los 4 bits del nodo siguiente.
- El bloque **Decodificador_7seg (4 → 7)** traduce el código binario del nodo (0–15) a los 7 bits que activan cada segmento del display.
- La señal **Enable** se genera mediante comparación (Estado_actual == 5) y se emplea para inhabilitar el reloj al llegar a `F`.

3.2. Tablas de verdad

Dec	Binario	Nodo	a	b	c	d	e	f	g
0	0000	A	1	1	1	0	1	1	1
1	0001	B	1	1	1	1	1	1	0
2	0010	C	1	0	0	1	1	1	1
3	0011	D	0	1	1	1	1	0	1
4	0100	E	1	0	0	1	1	1	1
5	0101	F	1	0	0	0	1	1	1
6	0110	G	1	1	0	1	1	1	1
7	0111	H	0	1	1	1	1	1	1
8	1000	I	1	1	1	1	1	1	0
9	1001	J	0	1	1	1	1	1	1
10	1010	K	1	1	0	1	1	1	1
11	1011	L	1	0	0	1	1	1	1
12	1100	M	1	1	1	0	1	1	1
13	1101	N	1	1	1	1	1	1	1
14	1110	O	1	1	1	1	1	1	1
15	1111	P	1	1	1	1	1	1	0

Cuadro 1: Tabla de verdad del decodificador de 7 segmentos para nodos A–P.

Mapa K-map de segmento a

	$b_1b_0 = 00$	01	11	10
$b_3b_2 = 00$	1	1	1	0
01	1	1	1	0
11	1	1	1	0
10	1	1	1	0

Mapa K-map de segmento b

	$b_1b_0 = 00$	01	11	10
$b_3b_2 = 00$	1	1	0	1
01	0	1	1	1
11	1	1	1	1
10	1	1	1	1

Mapa K-map de segmento c

	$b_1b_0 = 00$	01	11	10
$b_3b_2 = 00$	1	1	1	1
01	0	1	0	1
11	1	0	1	1
10	1	1	1	0

Mapa K-map de segmento d

	$b_1b_0 = 00$	01	11	10
$b_3b_2 = 00$	0	1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

Mapa K-map de segmento e

	$b_1b_0 = 00$	01	11	10
$b_3b_2 = 00$	1	1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

Mapa K-map de segmento f

	$b_1b_0 = 00$	01	11	10
$b_3b_2 = 00$	1	1	1	1
01	1	1	1	1
11	1	1	1	1
10	1	1	1	1

Mapa K-map de segmento g

	$b_1b_0 = 00$	01	11	10
$b_3b_2 = 00$	1	0	1	0
01	1	0	1	0
11	1	0	1	0
10	1	0	1	0

3.3. Mapas de Karnaugh

3.4. Subcircuitos

- Nombre
- Función
- Entradas y salidas
- Interacción con otros módulos

4. Simulación en Logisim

5. Pruebas

6. Supuestos

- Las rutas están predefinidas y son únicas por código de entrada.
- No se requiere retroceso ni manejo de errores en tiempo real.
- El sistema parte desde reposo y no repite el ciclo una vez finalizado.

7. Conclusión