DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
ESCUELA DE INGENIERÍA
PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

IIC2343 - Arquitectura de Computadores (I/2025)

Ayudantía 2

Ayudantes: Daniela Ríos (danielaarp@uc.cl), Joaquín Peralta (jperaltaperez@uc.cl), Gonzalo Bastías (gbastias.o@uc.c)

Pregunta 1: Diseño de un Circuito

En el curso de Arquitectura de Pokemones existen diferentes criterios que determinan la aprobación del curso:

- **Requisito 1 (R1):** Asistencia al curso (1 si la asistencia es del 85 % o más, 0 si es menor).
- **Requisito 2 (R2):** Entrega del proyecto final (1 si el proyecto es entregado, 0 si no).
- **Requisito 3 (R3):** Resultado de las interrogaciones (1 si el promedio es 4.0 o superior, 0 si es menor).

Para aprobar el curso (salida A = 1), se debe cumplir **al menos** una de las siguientes condiciones:

1. Se cumple R1 y R2.
2. Se cumple R1 y R3.
3. Se cumple R2 y R3.

En cualquier otro caso, el estudiante no aprueba el curso (A = 0).

A partir de esta descripción, responda:

(a) Elabore la tabla de verdad que describa el comportamiento del circuito.

(b) Diseñe el circuito lógico que implemente el comportamiento descrito, utilizando las siguientes compuertas: AND, OR, NOT, NAND o NOR.

Solución:

a) A continuación, la tabla de verdad para A a partir de $R_3R_2R_1$:

R_3	R_2	R_1	A
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

A partir de ella, se puede hacer uso de *minterms* o *maxterms* para obtener la expresión que representa a la salida A.

- Minterms:

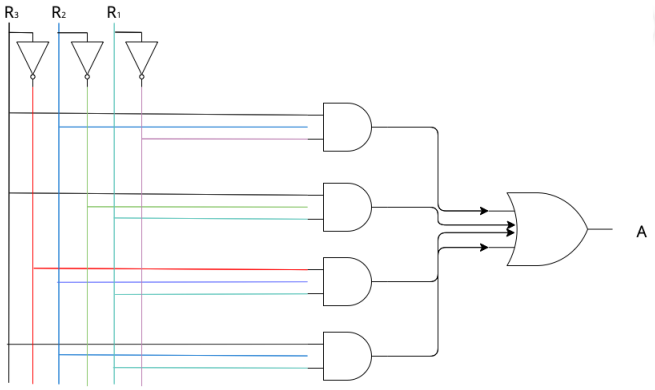
$$A = (R_3 \wedge R_2 \wedge \bar{R_1}) \vee (R_3 \wedge \bar{R_2} \wedge R_1) \vee (R_3 \wedge R_2 \wedge R_1) \vee (\bar{R_3} \wedge R_2 \wedge R_1)$$

- Maxterms:

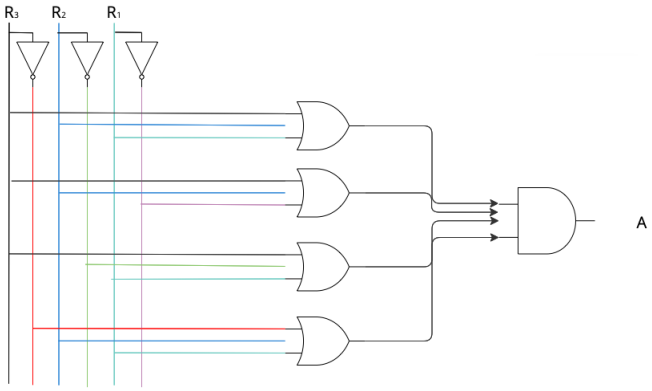
$$A = (R_3 \vee R_2 \vee R_1) \wedge (R_3 \vee R_2 \vee \bar{R_1}) \wedge (R_3 \vee \bar{R_2} \vee R_1) \wedge (\bar{R_3} \vee R_2 \vee \bar{R_1})$$

b) Finalmente, se puede diseñar un circuito a partir de estas expresiones:

Minterms:

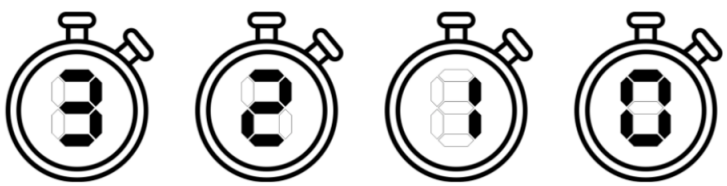


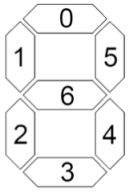
Maxterms:



Pregunta 2: Circuitos digitales (I1-2024-2)

Pronto se llevarán a cabo las DCCarreras de sacos dieciocheras y, para ahorrar recursos, le piden a usted que diseñe un *timer* para marcar el inicio de cada carrera. A continuación, una imagen de referencia del *timer* esperado y del *display* de 7 segmentos que tendrá integrado para desplegar el número:





Para construir el timer:

Diseñe, para cada segmento S_i del *display*, un circuito con una señal de entrada de 2 bits $I_1 I_0$ y una señal de salida de 1 bit. La entrada corresponderá al número a desplegar en el *timer* (3, 2, 1 o 0), mientras que la salida indica si el segmento S_i se prende (1) o no (0).

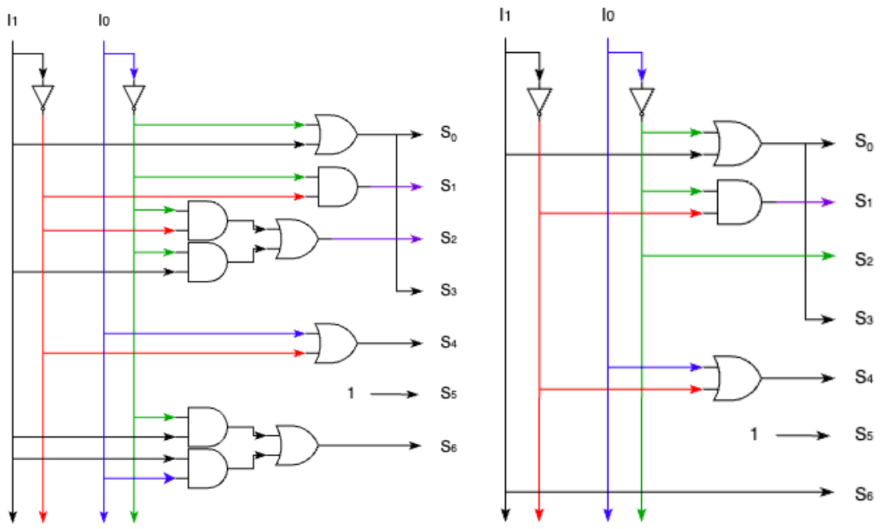
Solución: A continuación, la tabla de verdad para los segmentos S_i a partir de $I_1 I_2$:

I_1	I_0	S_0	S_1	S_2	S_3	S_4	S_5	S_6
0	0	1	1	1	1	1	1	0
0	1	0	0	0	0	1	1	1
1	0	1	0	1	1	0	1	1
1	1	1	0	0	1	1	1	1

A partir de ella, se puede hacer uso de minterms y maxterms para obtener la expresión que representa cada salida. A continuación, se listan expresiones válidas para cada segmento, junto con su reducción a través de equivalencias lógicas.

$$
\begin{aligned}
S_0 &= S_3 = I_1 \vee \bar{I}_0 \\
S_1 &= \bar{I}_1 \wedge \bar{I}_0 \\
S_2 &= (\bar{I}_1 \wedge \bar{I}_0) \vee (I_1 \wedge \bar{I}_0) = \bar{I}_0 \wedge (I_1 \vee \bar{I}_1) = \bar{I}_0 \wedge 1 = \bar{I}_0 \\
S_4 &= \bar{I}_1 \vee I_0 \\
S_5 &= 1 \\
S_6 &= (I_1 \wedge \bar{I}_0) \vee (I_1 \wedge I_0) = I_1 \wedge (\bar{I}_0 \vee I_0) = I_1 \wedge 1 = I_1
\end{aligned}
$$

Finalmente, se puede diseñar un circuito a partir de estas expresiones. Se incluye un circuito para las primeras expresiones obtenidas con minterms y maxterms (izquierda), y otro reducido a partir de equivalencias lógicas (derecha).



Al diseñar circuitos lógicos, no es necesario utilizar una compuerta OR para cada par de entradas; una sola compuerta OR puede combinar múltiples señales de entrada y se considera correcto.

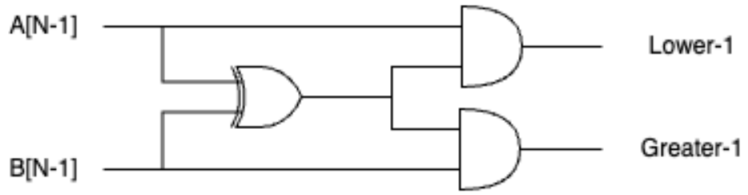
Pregunta 3: Operaciones aritméticas y lógicas (T1-2023-1)

Un comparador de números es un circuito que, dados dos números A y B en representación posicional, indica cuál es el mayor, o si estos son iguales. El circuito posee tres salidas, donde la primera entrega un 1 solo si A es el mayor, la segunda un 1 solo si ambos son iguales, y la tercera un 1 solo B es mayor. Haciendo uso de las compuertas lógicas vistas en clases, diseñe un comparador de números **enteros** de N bits, explicando la funcionalidad de cada uno de los circuitos que elabore.

Solución: Las tres salidas de nuestro circuito se llamarán *Greater, Equal y Lower* según lo descrito en el enunciado. Luego, para facilitar la construcción de nuestro circuito, vemos que existen cuatro casos a evaluar:

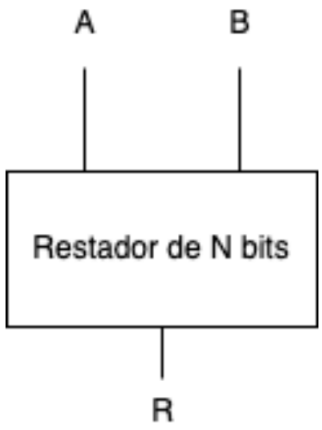
- **Caso 1:** $A \geq 0, B \geq 0$
- **Caso 2:** $A < 0, B < 0$
- **Caso 3:** $A \geq 0, B < 0$
- **Caso 4:** $A < 0, B \geq 0$

Los casos 3 y 4 se pueden resolver rápidamente haciendo uso del bit de signo: Verificamos que A y B posean signo distinto a través de una compuerta XOR (cuya salida es 1 solo si ambas señales son distintas) y este resultado lo conectamos con compuertas AND que nos ayudarán a determinar si se cumple que la condición *Greater-1 o Lower-1* (primer caso para identificar si el resultado es mayor o menor).



Se observa en el circuito que si $A_{N-1} = 0$,$B_{N-1} = 1$, entonces la señal *Greater-1* estará activa y la señal *Lower-1* no, mientras que se dará el caso contrario para $A_{N-1} = 1$, $B_{N-1} = 0$. Si $A_{N-1} = B_{N-1}$ ninguna de las señales se activará, independiente del signo de estos números.

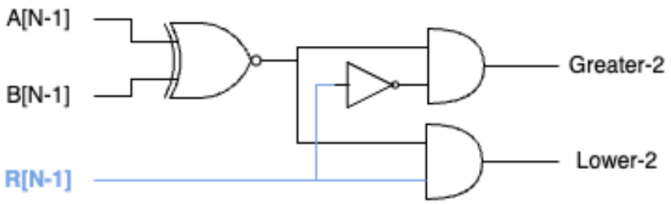
Para resolver los casos 1 y 2, haremos uso de un restador de N bits para obtener un nuevo bus de N bits R:



R nos sirve por las siguientes observaciones:

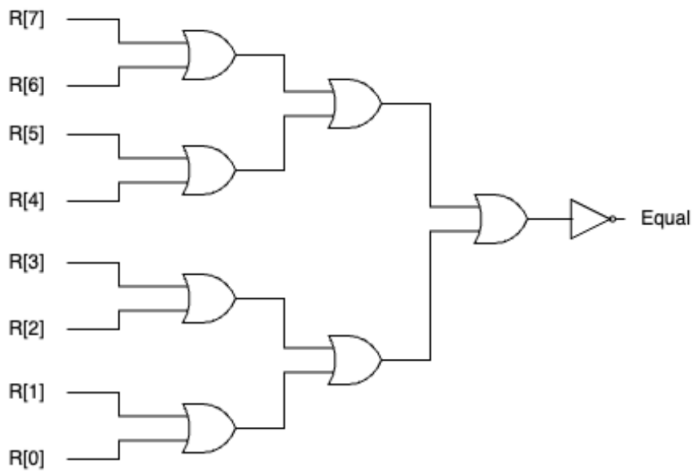
- Si $A - B > 0 \Rightarrow A > B \Rightarrow (A - B)_{N-1} = R_{N-1} = 0$
- Si $A - B < 0 \Rightarrow A < B \Rightarrow (A - B)_{N-1} = R_{N-1} = 1$

Notamos entonces que el bit más significativo de R nos indica si A >B o A <B, lo que podemos verificar con el siguiente circuito:



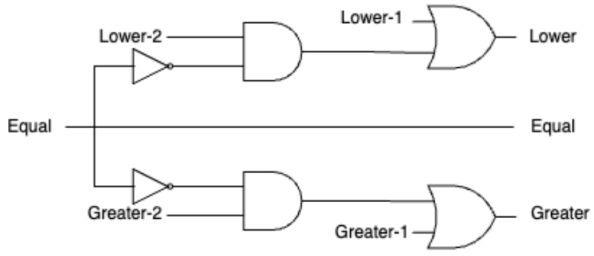
En contraste con el circuito anterior, aquí hacemos uso de una compuerta **XNOR** para verificar que los signos de A y B sean **iguales**. Observamos entonces que si $R_{N-1} = 0$, entonces se activará la señal *Greater-2* que indica que A >B y, en contraste, si $R_{N-1} = 1$ entonces se activará la señal *Lower-2* que indica que A <B (siempre que se cumpla A XNOR B = 1 para ambos casos).

El caso anterior no está del todo completo, dado que no estamos considerando la posibilidad donde A = B. Para ello, definimos un nuevo circuito que verifique que **todos los bits de R sean iguales a cero**:



En este caso se muestra para N = 8 pero su extensión a cualquier N es trivial. El circuito activa la señal Equal si, y solo si ninguna compuerta OR entrega como resultado 1 (que se da solo si alguno de los bits de R es igual a 1).

Usamos este último resultado para entregar la señal faltante Equal y para que los casos *Greater-2 y Lower-2* sean válidos solo si Equal = 0:



De esta forma, *Greater, Lower y Equal* se activan correctamente según los casos señalados al comienzo.

1. Feedback ayudantía

Escanee el QR para entregar feedback sobre la ayudantía.