Lógica digital

Organización de computadoras

Universidad Nacional de Quilmes

http://orga.blog.unq.edu.ar

Repaso

1. Punto Flotante
    1. Motivación

Repaso

1. Punto Flotante
    1. Motivación
    2. Interpretar

Repaso

1. Punto Flotante
    1. Motivación
    2. Interpretar
    3. Rango

Repaso

1. Punto Flotante
    1. Motivación
    2. Interpretar
    3. Rango
    4. Resolución variable

Repaso

1. Punto Flotante
    1. Motivación
    2. Interpretar
    3. Rango
    4. Resolución variable
    5. Normalización (bit implícito)

Repaso

1. Punto Flotante
    1. Motivación
    2. Interpretar
    3. Rango
    4. Resolución variable
    5. Normalización (bit implícito)
    6. Representar

Niveles de abstracción

Los sistemas complejos pueden verse desde distintos puntos de vista

Niveles de abstracción

Los sistemas complejos pueden verse desde distintos puntos de vista

En particular, el funcionamiento de la computadora se puede separar en niveles de abstracción

Niveles de abstracción

Aplicaciones de Usuario	El usuario interactua con la computadora ejecutando herramientas como procesadores de texto, juegos o browsers.
Lenguaje de alto nivel	
Lenguaje Assembly	
Control	
Unidades funcionales	
Compuertas, transistores	

Niveles de abstracción

Aplicaciones de Usuario	
Lenguaje de alto nivel	El programador entiende el lenguaje (Gobstones, Java, C) pero no los detalles de implementación. El compilador es el encargado corresponder el programa en el lenguaje de alto nivel con el lenguaje específico para la arquitectura subyacente
Lenguaje Assembly	
Control	
Unidades funcionales	
Compuertas, transistores	

Niveles de abstracción

Aplicaciones de Usuario	
Lenguaje de alto nivel	
Lenguaje Assembly	El lenguaje assembly (Q1..Q5)tiene en cuenta los detalles de la arquitectura (registros, repertorio de operaciones, etc). Dado que programar en términos de 1s y 0s es tedioso y propenso a errores, en este nivel se provee un programa **ensamblador** que traduce las sentencias mnemotécnicas del lenguaje assembly al código máquina.
Control	
Unidades funcionales	
Compuertas,	

Niveles de abstracción

Aplicaciones de Usuario	
Lenguaje de alto nivel	
Lenguaje Assembly	
Control	Se interpretan las instrucciones máquina para producir la operación sobre los datos. Existen distintas formas de hacerlo: **Hardwiring** Las acciones se llevan a cabo mediante un conjunto de componentes digitales (compuertas) **Microprogramas** Cada instrucción máquina se traduce en un microprograma, escrito en un lenguaje de un nivel aún mas bajo que se implementa con hardware (firmware). El firmware es ejecutado por un microcontrolador.

Niveles de abstracción

Aplicaciones de Usuario	
Lenguaje de alto nivel	
Lenguaje Assembly	
Control	
Unidades funcionales	Registros, ALU, Memoria Principal
Compuertas, transistores	

Niveles de abstracción

Aplicaciones de Usuario	
Lenguaje de alto nivel	
Lenguaje Assembly	
Control	
Unidades funcionales	
Compuertas, transistores	Las unidades funcionales se contruyen a partir de compuertas lógicas y éstas a partir de transistores. Las compuertas realizan las operaciones lógicas fundamentales.

Niveles de abstracción

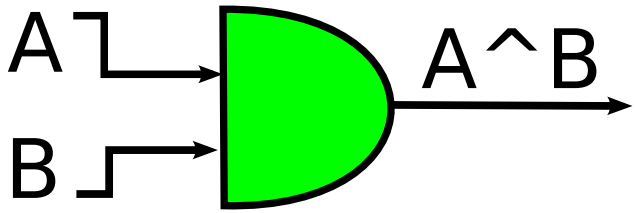
Como resultado de esta separación de niveles, es posible tener un conjunto de máquinas que difieran en la implementación a bajo nivel, pero que tengan el mismo set de instrucciones o superset. Así IBM 360 fue la primera familia de computadoras que garantizó compatibilidad "hacia adelante"

¿Qué son las compuertas?

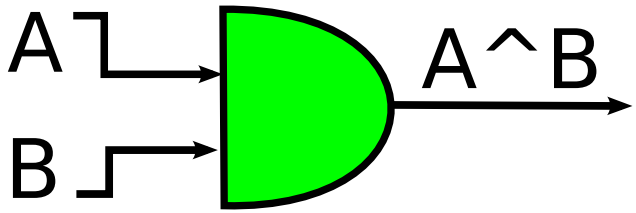
Compuertas lógicas

Compuerta lógica
es un dispositivo que implementa una función booleana simple. Traduce un conjunto de entradas (una o más) en una salida

Compuertas lógicas

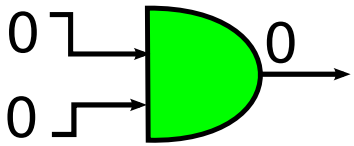


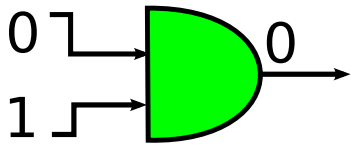
Compuertas lógicas

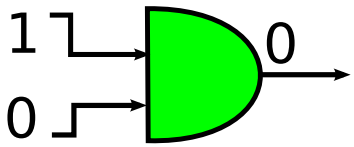


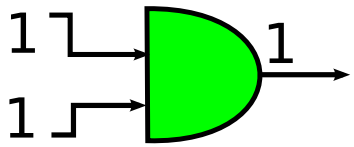
A	B	$A \wedge B$
0	0	0
0	1	0
1	0	0
1	1	1

Casos:

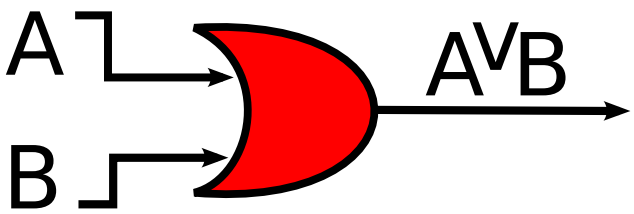




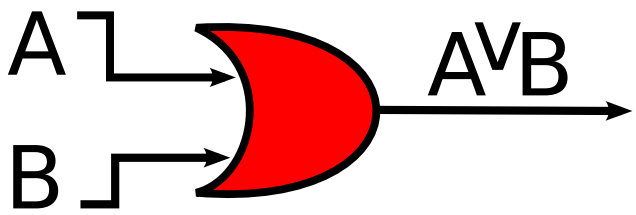




Compuertas lógicas

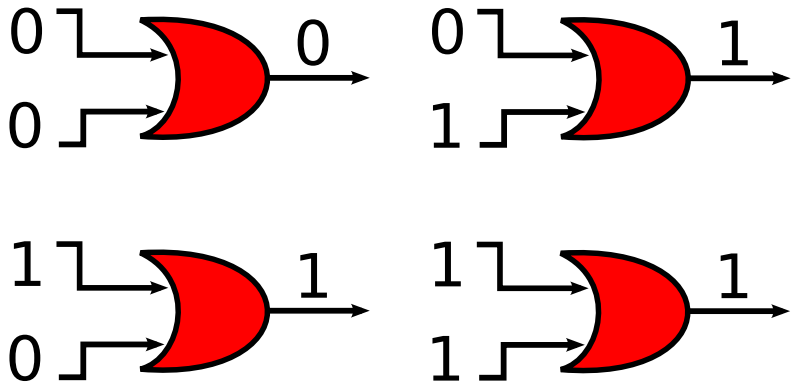


Compuertas lógicas

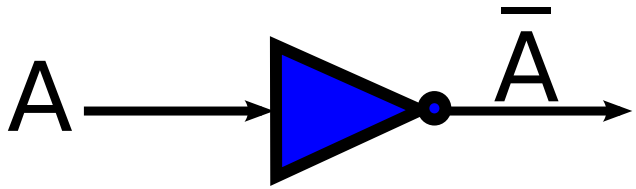


A	B	$A \vee B$
0	0	0
0	1	1
1	0	1
1	1	1

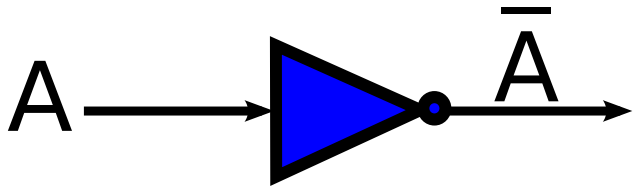
Casos:



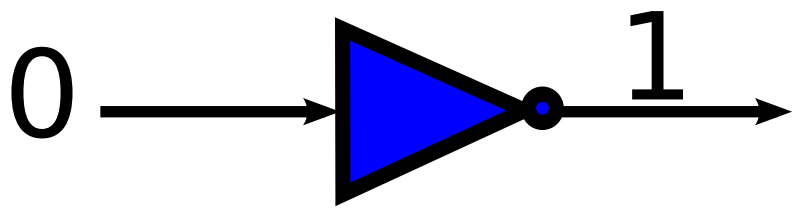
Compuertas lógicas



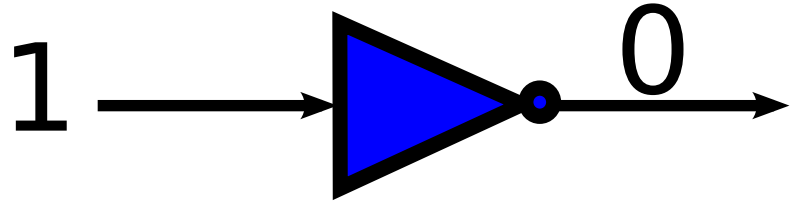
Compuertas lógicas



Casos:

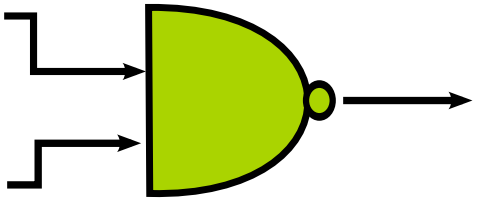


A	$\overline{A}$
0	1
1	0



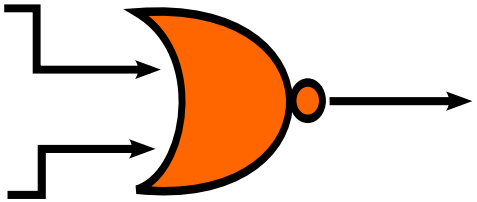
Compuertas lógicas

1. Compuerta NAND



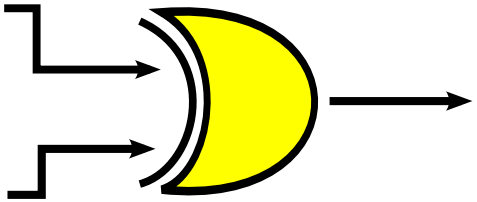
$$a \uparrow b = \overline{a \wedge b}$$

2. Compuerta NOR



$$a \downarrow b = \overline{a \vee b}$$

3. Compuerta XOR



$$a \oplus b = (\overline{a} \wedge b) \vee (a \wedge \overline{b})$$

¿Qué son los circuitos?

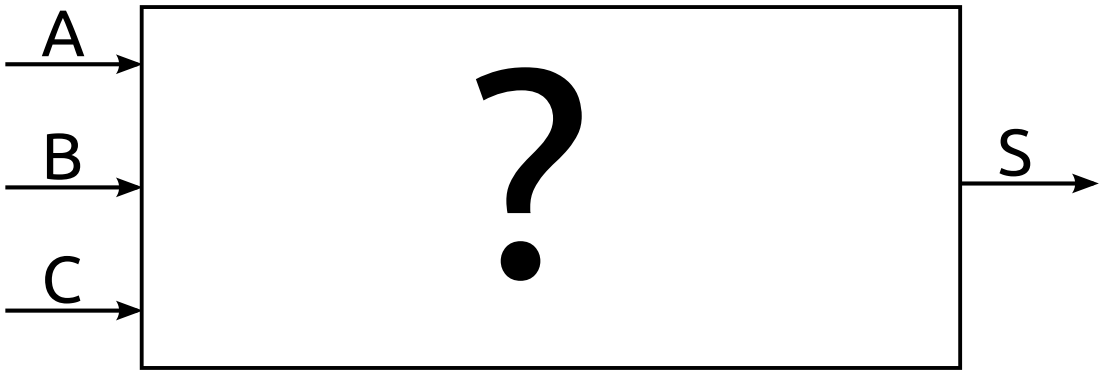
Compuertas y circuitos lógicos

Circuito lógico

- Composición de compuertas
- Traduce un conjunto de entradas en un conjunto de salidas de acuerdo a una o mas funciones booleanas
- Cada salida es estrictamente una función de las entradas
- Las salidas se actualizan de inmediato luego de que cambien las entradas

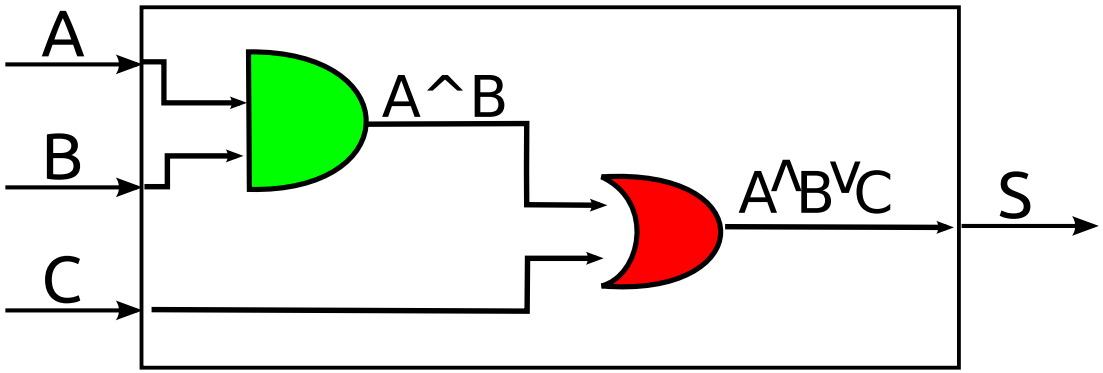
Circuitos lógicos

Ejemplo: ¿Cómo es el circuito de $A \wedge B \vee C$?



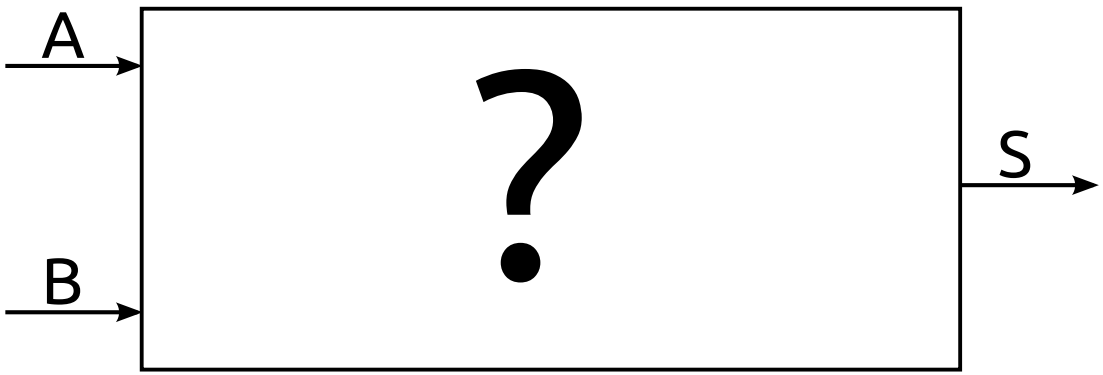
Circuitos lógicos

Ejemplo: ¿Cómo es el circuito de $A \wedge B \vee C$?



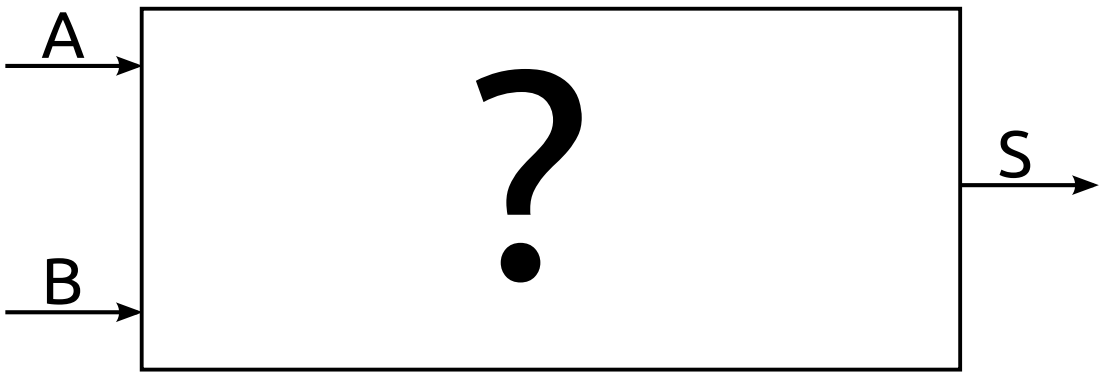
Circuitos lógicos

Ejercicio: ¿Cómo es el circuito de $(A \wedge \overline{B})$?



Circuitos lógicos

Ejercicio: ¿Cómo es el circuito de $(A \wedge \overline{B}) \vee (\overline{A} \wedge B)$?



Circuitos lógicos

Los circuitos se construyen a partir de...

a. Una tabla de verdad
b. Un enunciado en lenguaje natural
c. Una fórmula

Circuitos lógicos

Los circuitos se construyen a partir de...

a. Una tabla de verdad  fórmula
b. Un enunciado en lenguaje natural
c. Una fórmula

Circuitos lógicos

Los circuitos se construyen a partir de...

a. Una tabla de verdad  fórmula
b. Un enunciado en lenguaje natural  tabla  fórmula
c. Una fórmula

Circuitos lógicos

Los circuitos se construyen a partir de...

a. Una tabla de verdad  fórmula
b. Un enunciado en lenguaje natural  tabla  fórmula
c. Una fórmula

Hagamos un cirucito

Ejemplo de construcción de un circuito

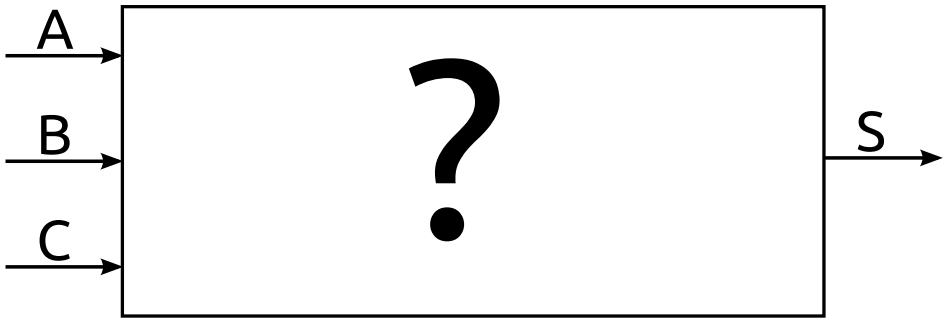
Realizar un circuito de 3 entradas que calcule la función mayoría:

- si dos o mas entradas valen 1: se obtiene un 1
- caso contrario: se obtiene un 0

Ejemplo de construcción de un circuito

Realizar un circuito de 3 entradas que calcule la función mayoría:

- si dos o mas entradas valen 1: se obtiene un 1
- caso contrario: se obtiene un 0



Ejemplo de construcción de un circuito

Ejemplo de construcción de un circuito

Ejemplo de construcción de un circuito

Lenguaje natural → Tabla de verdad → Fórmula booleana

Ejemplo de construcción de un circuito

Lenguaje natural ⇢ Tabla de verdad ⇢ Fórmula booleana

Realizar un circuito de 3 entradas que calcule la función mayoría:

- si dos o mas entradas valen 1: se obtiene un 1
- caso contrario: se obtiene un 0

E_1	E_2	E_3	S
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

Ejemplo de construcción de un circuito

Lenguaje natural  Tabla de verdad  Fórmula booleana

Ejemplo de construcción de un circuito

Lenguaje natural → Tabla de verdad → Fórmula booleana

1. Construir la tabla de verdad
2. Plantear la fórmula que describe cada caso donde la salida vale 1
3. Unir los casos con disyunción

E_1	E_2	E_3	S
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

Ejemplo de construcción de un circuito

Lenguaje natural → Tabla de verdad → Fórmula booleana

1. Construir la tabla de verdad
2. Plantear la fórmula que describe cada caso donde la salida vale 1
3. Unir los casos con disyunción

E_1	E_2	E_3	S	
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$(\overline{E}_1 \wedge E_2 \wedge E_3)$
1	0	0	0	
1	0	1	1	
1	1	0	1	
1	1	1	1	

Ejemplo de construcción de un circuito

Lenguaje natural ⇢ Tabla de verdad ⇢ Fórmula booleana

1. Construir la tabla de verdad
2. Plantear la fórmula que describe cada caso donde la salida vale 1
3. Unir los casos con disyunción

E_1	E_2	E_3	S	
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	
1	0	0	0	
1	0	1	1	$(E_1 \wedge \overline{E}_2 \wedge E_3)$
1	1	0	1	
1	1	1	1	

Ejemplo de construcción de un circuito

Lenguaje natural → Tabla de verdad → Fórmula booleana

1. Construir la tabla de verdad
2. Plantear la fórmula que describe cada caso donde la salida vale 1
3. Unir los casos con disyunción

E_1	E_2	E_3	S	
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	$(E_1 \wedge E_2 \wedge \overline{E}_3)$
1	1	1	1	

Ejemplo de construcción de un circuito

Lenguaje natural → Tabla de verdad → Fórmula booleana

1. Construir la tabla de verdad
2. Plantear la fórmula que describe cada caso donde la salida vale 1
3. Unir los casos con disyunción

E_1	E_2	E_3	S	
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	$(E_1 \wedge E_2 \wedge E_3)$

Ejemplo de construcción de un circuito

Lenguaje natural → Tabla de verdad → Fórmula booleana

1. Construir la tabla de verdad
2. Plantear la fórmula que describe cada caso donde la salida vale 1
3. Unir los casos con disyunción

E_1	E_2	E_3	S
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

$$s = (\overline{E}_1 \wedge E_2 \wedge E_3) \vee (E_1 \wedge \overline{E}_2 \wedge E_3) \vee (E_1 \wedge E_2 \wedge \overline{E}_3) \vee (E_1 \wedge E_2 \wedge E_3)$$

Ejemplo de construcción de un circuito

Lenguaje natural → Tabla de verdad → Fórmula booleana

1. Construir la tabla de verdad
2. Plantear la fórmula que describe cada caso donde la salida vale 1
3. Unir los casos con disyunción

E_1	E_2	E_3	S
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1 $(\overline{E}_1 \wedge E_2 \wedge E_3)$
1	0	0	0
1	0	1	1 $(E_1 \wedge \overline{E}_2 \wedge E_3)$
1	1	0	1 $(E_1 \wedge E_2 \wedge \overline{E}_3)$
1	1	1	1 $(E_1 \wedge E_2 \wedge E_3)$

$$s = (\overline{E}_1 \wedge E_2 \wedge E_3) \vee (E_1 \wedge \overline{E}_2 \wedge E_3) \vee (E_1 \wedge E_2 \wedge \overline{E}_3) \vee (E_1 \wedge E_2 \wedge E_3)$$

Suma de Productos

Suma de Productos

Obtener la Suma de Productos

Suma de Productos

Obtener la Suma de Productos

$$s = (\overline{E}_1 \wedge E_2 \wedge E_3) \vee (E_1 \wedge \overline{E}_2 \wedge E_3) \vee (E_1 \wedge E_2 \wedge \overline{E}_3) \vee (E_1 \wedge E_2 \wedge E_3)$$

SOP

(Suma de productos) Fórmula booleana compuesta por disyunciones ($\vee$) entre términos que son conjunciones ($\wedge$) de literales (a ó $\overline{a}$)

Suma de Productos

Obtener la Suma de Productos

$$s = \underbrace{(\overline{E}_1 \wedge E_2 \wedge E_3)}_{\text{término}} \vee \underbrace{(E_1 \wedge \overline{E}_2 \wedge E_3)}_{\text{término}}$$
$$\vee \underbrace{(E_1 \wedge E_2 \wedge \overline{E}_3)}_{\text{término}} \vee \underbrace{(E_1 \wedge E_2 \wedge E_3)}_{\text{término}}$$

SOP
(Suma de productos) Fórmula booleana compuesta por disyunciones ($\vee$) entre términos que son conjunciones ($\wedge$) de literales (a ó $\overline{a}$)

Ejemplo de construcción de un circuito

¿Es posible simplificar?

$$s = (\overline{E}_1 \wedge E_2 \wedge E_3) \vee (E_1 \wedge \overline{E}_2 \wedge E_3) \vee (E_1 \wedge E_2 \wedge \overline{E}_3) \vee (E_1 \wedge E_2 \wedge E_3) =$$

Ejemplo de construcción de un circuito

¿Es posible simplificar?

$$s = (\overline{E}_1 \wedge E_2 \wedge E_3) \vee (E_1 \wedge \overline{E}_2 \wedge E_3) \vee (E_1 \wedge E_2 \wedge \overline{E}_3) \vee (E_1 \wedge E_2 \wedge E_3) =$$

Por propiedad distributiva:

$$= ((\overline{E}_1 \wedge E_2) \vee (E_1 \wedge \overline{E}_2)) \wedge E_3 \vee (E_1 \wedge E_2) \wedge (E_3 \vee \overline{E}_3)$$

Ejemplo de construcción de un circuito

¿Es posible simplificar?

$$s = (\overline{E}_1 \wedge E_2 \wedge E_3) \vee (E_1 \wedge \overline{E}_2 \wedge E_3) \vee (E_1 \wedge E_2 \wedge \overline{E}_3) \vee (E_1 \wedge E_2 \wedge E_3) =$$

Por propiedad distributiva:

$$= ((\overline{E}_1 \wedge E_2) \vee (E_1 \wedge \overline{E}_2)) \wedge E_3 \vee (E_1 \wedge E_2) \wedge (E_3 \vee \overline{E}_3)$$

Por complemento en $\vee$:

$$= ((\overline{E}_1 \wedge E_2) \vee (E_1 \wedge \overline{E}_2)) \wedge E_3 \vee (E_1 \wedge E_2) \wedge 1$$

Ejemplo de construcción de un circuito

¿Es posible simplificar?

$$s = (\overline{E}_1 \wedge E_2 \wedge E_3) \vee (E_1 \wedge \overline{E}_2 \wedge E_3) \vee (E_1 \wedge E_2 \wedge \overline{E}_3) \vee (E_1 \wedge E_2 \wedge E_3) =$$

Por propiedad distributiva:

$$= ((\overline{E}_1 \wedge E_2) \vee (E_1 \wedge \overline{E}_2)) \wedge E_3 \vee (E_1 \wedge E_2) \wedge (E_3 \vee \overline{E}_3)$$

Por complemento en $\vee$:

$$= ((\overline{E}_1 \wedge E_2) \vee (E_1 \wedge \overline{E}_2)) \wedge E_3 \vee (E_1 \wedge E_2) \wedge 1$$

Por neutro de $\wedge$:

$$= ((\overline{E}_1 \wedge E_2) \vee (E_1 \wedge \overline{E}_2)) \wedge E_3 \vee (E_1 \wedge E_2)$$

Ejemplo de construcción de un circuito

¿Es posible simplificar?

$$s = (\overline{E}_1 \wedge E_2 \wedge E_3) \vee (E_1 \wedge \overline{E}_2 \wedge E_3) \vee (E_1 \wedge E_2 \wedge \overline{E}_3) \vee (E_1 \wedge E_2 \wedge E_3) =$$

Por propiedad distributiva:

$$= ((\overline{E}_1 \wedge E_2) \vee (E_1 \wedge \overline{E}_2)) \wedge E_3 \vee (E_1 \wedge E_2) \wedge (E_3 \vee \overline{E}_3)$$

Por complemento en $\vee$:

$$= ((\overline{E}_1 \wedge E_2) \vee (E_1 \wedge \overline{E}_2)) \wedge E_3 \vee (E_1 \wedge E_2) \wedge 1$$

Por neutro de $\wedge$:

$$= ((\overline{E}_1 \wedge E_2) \vee (E_1 \wedge \overline{E}_2)) \wedge E_3 \vee (E_1 \wedge E_2)$$

Por definición de $\oplus$:

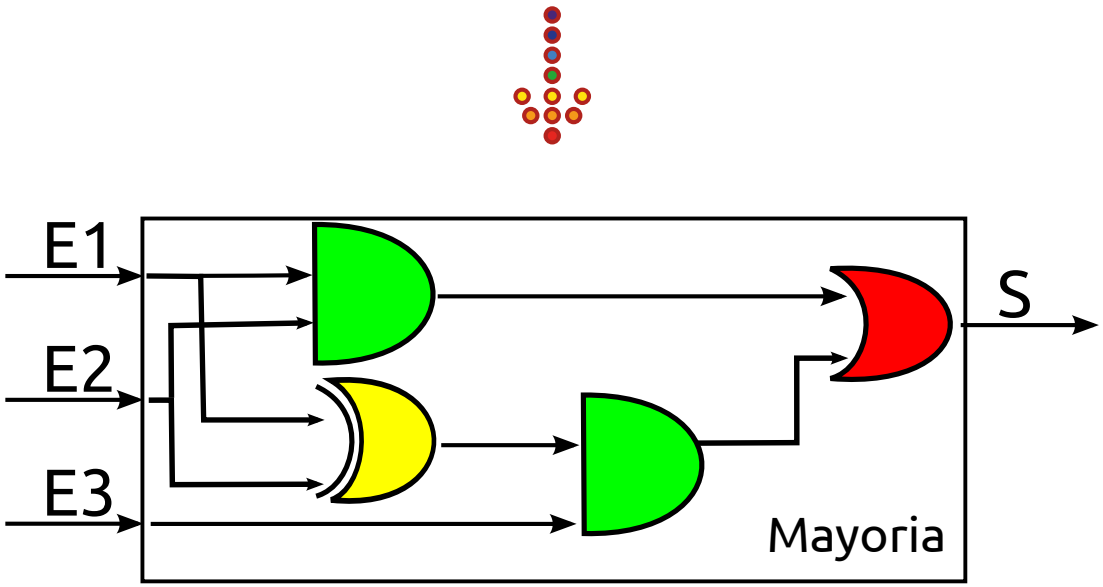
$$= (E_1 \oplus E_2) \wedge E_3 \vee (E_1 \wedge E_2)$$

Ejemplo de construcción de un circuito

$$(E_1 \oplus E_2) \wedge E_3 \vee (E_1 \wedge E_2)$$

Ejemplo de construcción de un circuito

$$(E_1 \oplus E_2) \wedge E_3 \vee (E_1 \wedge E_2)$$



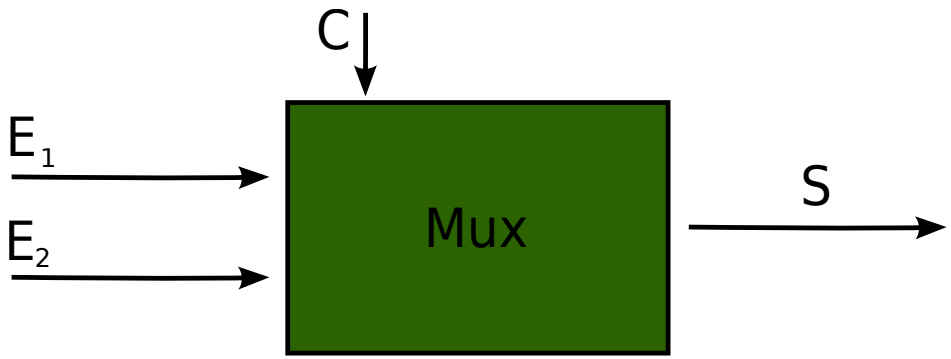
Circuitos mas usados

Multiplexor simple

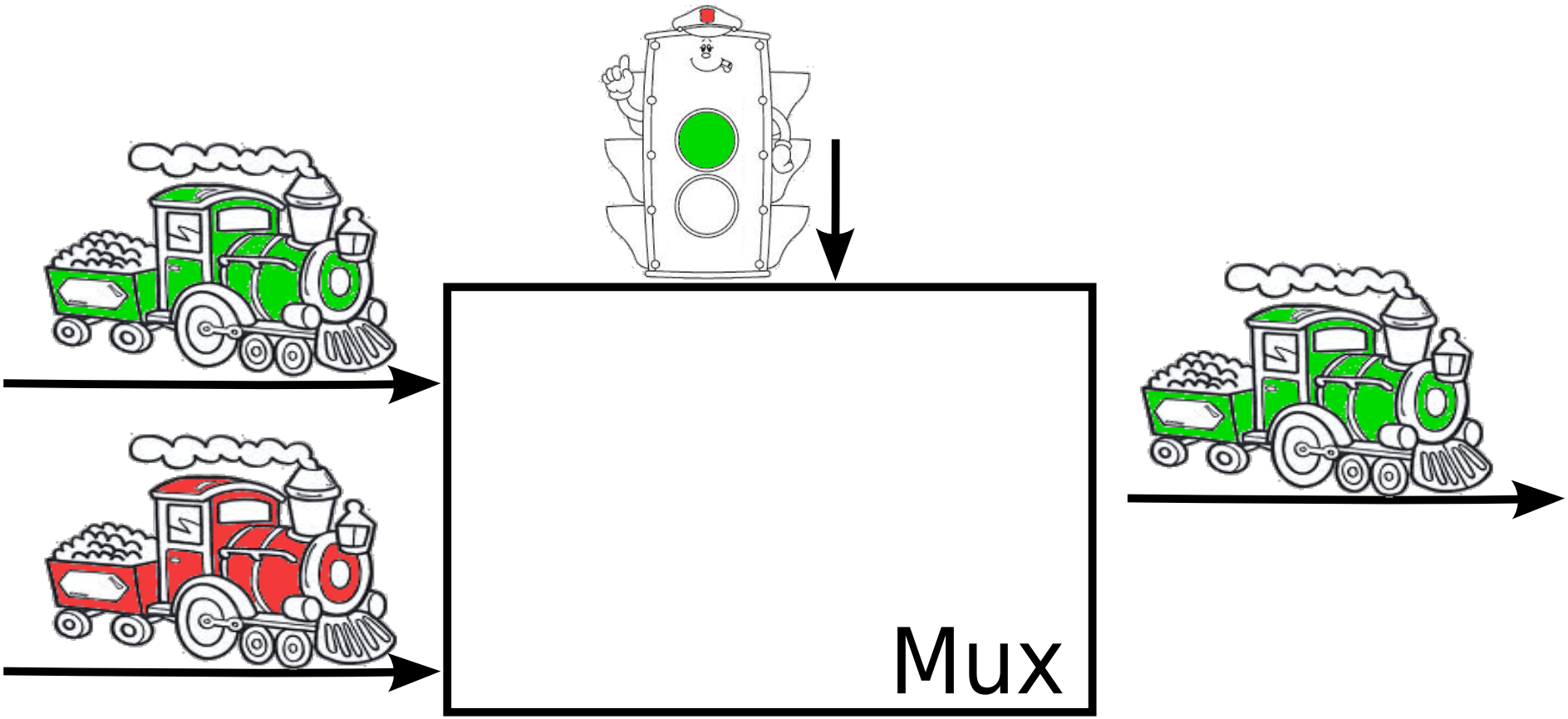
Objetivo Proyectar una de las entradas en la salida, a partir la configuración del control

Entradas 2 entradas, una línea de control

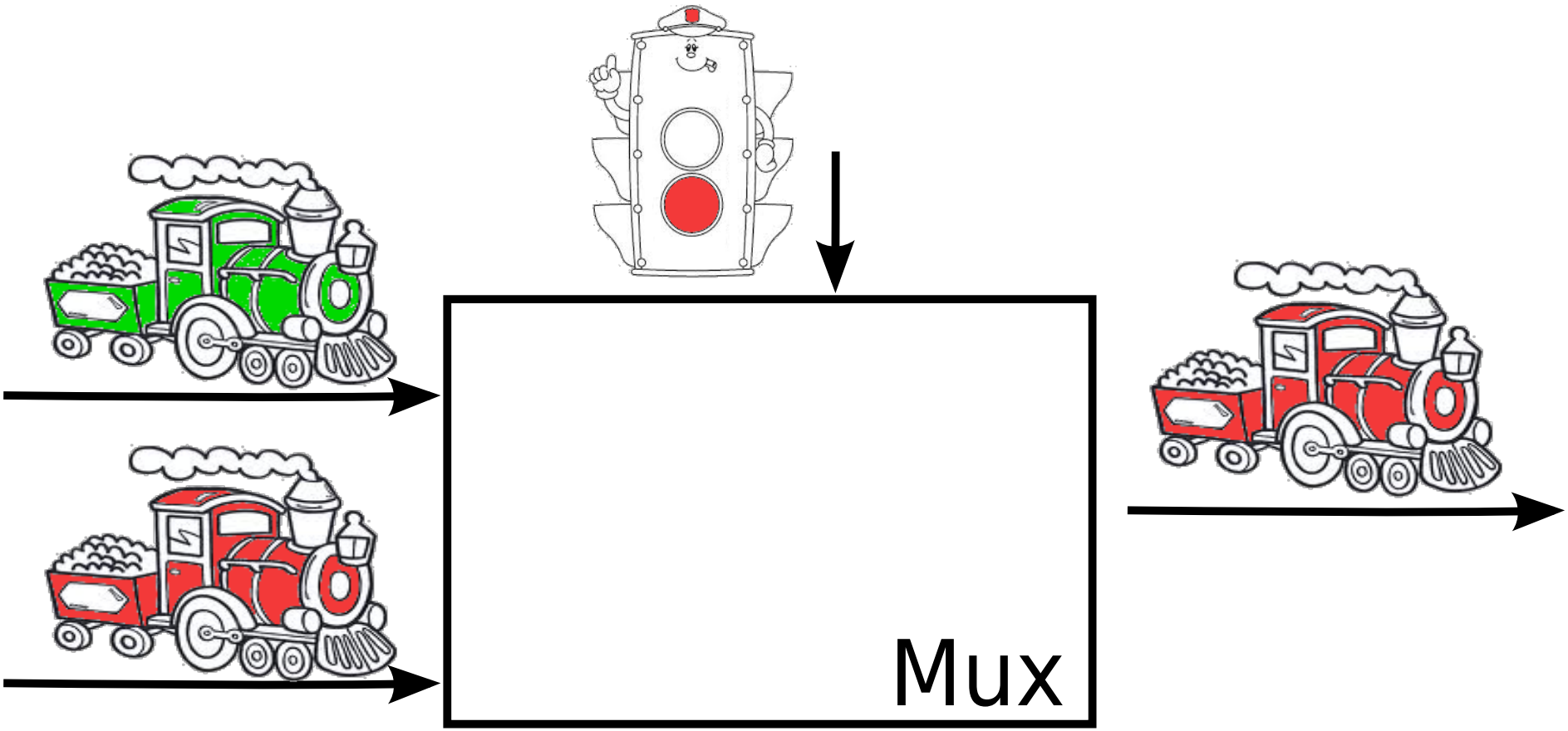
Salida 1 salida



Multiplexor simple: La idea



Multiplexor simple: La idea



Multiplexor simple

Lenguaje natural → Tabla de verdad → Fórmula booleana

Multiplexor simple

Lenguaje natural  Tabla de verdad

Tabla abreviada:

C	S
0	e_1
1	e_2

Multiplexor simple

Lenguaje natural  Tabla de verdad

Tabla abreviada:

C	S
0	e_1
1	e_2

Tabla de verdad:

C	E_1	E_2	S
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Multiplexor simple

Lenguaje natural → Tabla de verdad → Fórmula booleana

Multiplexor simple

Tabla de verdad → Fórmula booleana

C	E_1	E_2	S
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Multiplexor simple

Tabla de verdad → Fórmula booleana

C	E_1	E_2	S	
0	0	0	0	
0	0	1	0	
0	1	0	1	$(\overline{C} \wedge E_1 \wedge \overline{E_2})$
0	1	1	1	
1	0	0	0	
1	0	1	1	
1	1	0	0	
1	1	1	1	

Multiplexor simple

Tabla de verdad ⇢ Fórmula booleana

C	E_1	E_2	S	
0	0	0	0	
0	0	1	0	
0	1	0	1	
0	1	1	1	$(\overline{C} \wedge E_1 \wedge E_2)$
1	0	0	0	
1	0	1	1	
1	1	0	0	
1	1	1	1	

Multiplexor simple

Tabla de verdad → Fórmula booleana

C	E_1	E_2	S	
0	0	0	0	
0	0	1	0	
0	1	0	1	
0	1	1	1	
1	0	0	0	
1	0	1	1	$(C \wedge \overline{E}_1 \wedge E_2)$
1	1	0	0	
1	1	1	1	

Multiplexor simple

Tabla de verdad → Fórmula booleana

C	E_1	E_2	S	
0	0	0	0	
0	0	1	0	
0	1	0	1	
0	1	1	1	
1	0	0	0	
1	0	1	1	
1	1	0	0	
1	1	1	1	$(C \wedge E_1 \wedge E_2)$

Multiplexor simple

Tabla de verdad → Fórmula booleana

C	E_1	E_2	S	
0	0	0	0	
0	0	1	0	
0	1	0	1	$(\overline{C} \wedge E_1 \wedge \overline{E_2})$
0	1	1	1	$(\overline{C} \wedge E_1 \wedge E_2)$
1	0	0	0	
1	0	1	1	$(C \wedge \overline{E_1} \wedge E_2)$
1	1	0	0	
1	1	1	1	$(C \wedge E_1 \wedge E_2)$

$$s = (\overline{C} \wedge E_1 \wedge \overline{E_2}) \vee (\overline{C} \wedge E_1 \wedge E_2) \vee (C \wedge \overline{E_1} \wedge E_2) \vee (C \wedge E_1 \wedge E_2)$$

Multiplexor simple

¡Simplificar!

$$s = (\overline{C} \wedge E_1 \wedge \overline{E}_2) \vee (\overline{C} \wedge E_1 \wedge E_2) \vee (C \wedge \overline{E}_1 \wedge E_2) \vee (C \wedge E_1 \wedge E_2)$$

Multiplexor simple

¡Simplificar!

$$s = (\overline{C} \wedge E_1 \wedge \overline{E}_2) \vee (\overline{C} \wedge E_1 \wedge E_2) \vee (C \wedge \overline{E}_1 \wedge E_2) \vee (C \wedge E_1 \wedge E_2)$$

Por distributiva:

$$(\overline{C} \wedge E_1) \wedge (\overline{E}_2 \vee E_2) \vee (C \wedge \overline{E}_1 \wedge E_2) \vee (C \wedge E_1 \wedge E_2)$$

Multiplexor simple

¡Simplificar!

$$s = (\overline{C} \wedge E_1 \wedge \overline{E}_2) \vee (\overline{C} \wedge E_1 \wedge E_2) \vee (C \wedge \overline{E}_1 \wedge E_2) \vee (C \wedge E_1 \wedge E_2)$$

Por distributiva:

$$(\overline{C} \wedge E_1) \wedge (\overline{E}_2 \vee E_2) \vee (C \wedge \overline{E}_1 \wedge E_2) \vee (C \wedge E_1 \wedge E_2)$$

Por distributiva:

$$(\overline{C} \wedge E_1) \wedge (\overline{E}_2 \vee E_2) \vee (C \wedge E_2) \wedge (\overline{E}_1 \vee E_1)$$

Multiplexor simple

¡Simplificar!

$$s = (\overline{C} \wedge E_1 \wedge \overline{E}_2) \vee (\overline{C} \wedge E_1 \wedge E_2) \vee (C \wedge \overline{E}_1 \wedge E_2) \vee (C \wedge E_1 \wedge E_2)$$

Por distributiva:

$$(\overline{C} \wedge E_1) \wedge (\overline{E}_2 \vee E_2) \vee (C \wedge \overline{E}_1 \wedge E_2) \vee (C \wedge E_1 \wedge E_2)$$

Por distributiva:

$$(\overline{C} \wedge E_1) \wedge (\overline{E}_2 \vee E_2) \vee (C \wedge E_2) \wedge (\overline{E}_1 \vee E_1)$$

Por complemento de la $\vee$:

$$(\overline{C} \wedge E_1) \wedge 1 \vee (C \wedge E_2) \wedge 1$$

Multiplexor simple

¡Simplificar!

$$s = (\overline{C} \wedge E_1 \wedge \overline{E}_2) \vee (\overline{C} \wedge E_1 \wedge E_2) \vee (C \wedge \overline{E}_1 \wedge E_2) \vee (C \wedge E_1 \wedge E_2)$$

Por distributiva:

$$(\overline{C} \wedge E_1) \wedge (\overline{E}_2 \vee E_2) \vee (C \wedge \overline{E}_1 \wedge E_2) \vee (C \wedge E_1 \wedge E_2)$$

Por distributiva:

$$(\overline{C} \wedge E_1) \wedge (\overline{E}_2 \vee E_2) \vee (C \wedge E_2) \wedge (\overline{E}_1 \vee E_1)$$

Por complemento de la $\vee$:

$$(\overline{C} \wedge E_1) \wedge 1 \vee (C \wedge E_2) \wedge 1$$

Por neutro de la $\wedge$:

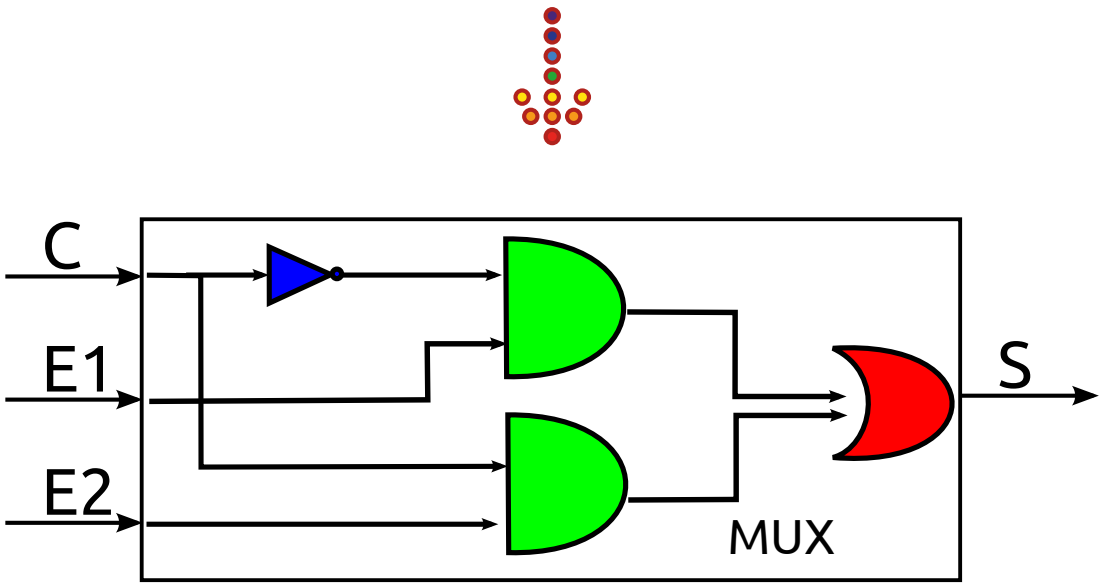
$$(\overline{C} \wedge E_1) \vee (C \wedge E_2)$$

Multiplexor simple

$$(\overline{C} \wedge E_1) \vee (C \wedge E_2)$$

Multiplexor simple

$$(\overline{C} \wedge E_1) \vee (C \wedge E_2)$$

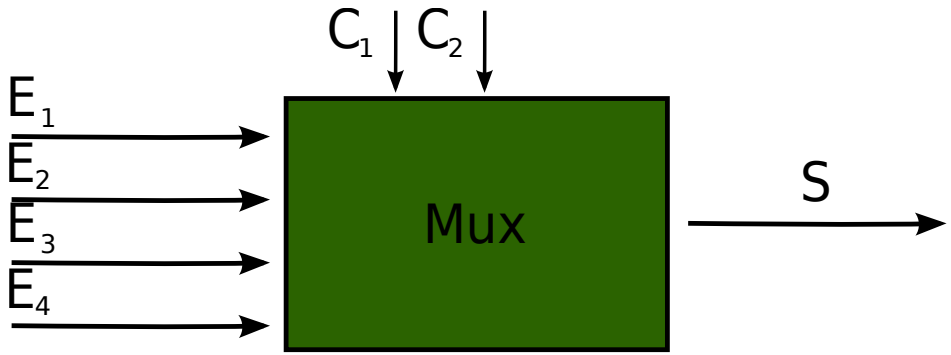


Multiplexor complejo

Objetivo Proyectar una de las entradas en la salida, a partir la configuración del control

Entradas 4 entradas

Salida 1 salida



Multiplexor complejo

Tabla abreviada:

C_1	C_2	S
0	0	e_1
0	1	e_2
1	0	e_3
1	1	e_4

Multiplexor complejo

Tabla abreviada:

C_1	C_2	S
0	0	e_1
0	1	e_2
1	0	e_3
1	1	e_4

Tabla de verdad:

C_1	C_2	E_1	E_2	E_3	E_4	S
0	0	0	0	0	0	0

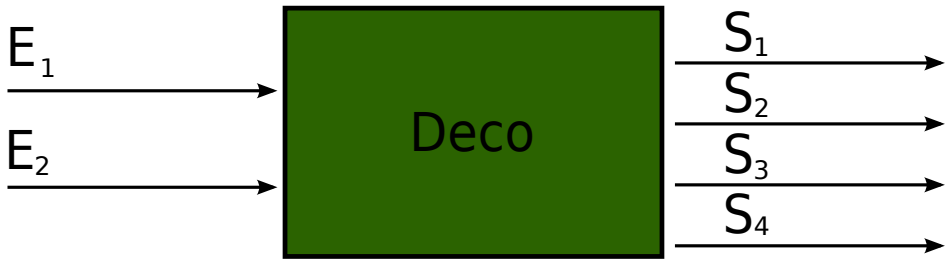
¡Completar de Tarea!

Decodificador

Objetivo Traduce un *código* de 2 bits en uno de 4 valores

Entrada 2 bits de la cadena de entrada (2 entradas)

Salida 4 líneas de salida



Decodificador

E_1	E_2	S_1	S_2	S_3	S_4

Tabla de verdad:

Decodificador

Tabla de verdad:

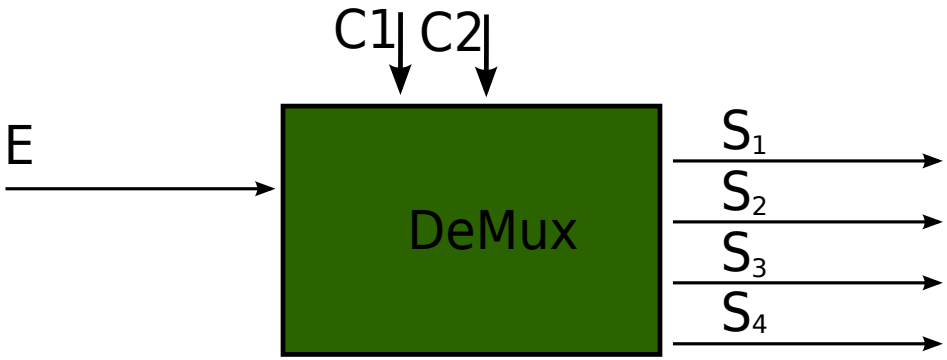
E_1	E_2	S_1	S_2	S_3	S_4
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Demultiplexor

Objetivo Permite configurar por qué salida se proyecta la entrada.

Entrada 1 línea de entrada, y dos líneas de control

Salida 4 líneas de salida



Demultiplexor

E	C_1	C_2	S_1	S_2	S_3	S_4

Tabla de verdad:

Demultiplexor

Tabla de verdad:

E	C_1	C_2	S_1	S_2	S_3	S_4
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

Demultiplexor

Tabla de verdad:

E	C_1	C_2	S_1	S_2	S_3	S_4
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	0
1	1	0	0	0	1	0
1	1	1	0	0	0	1

¿Cómo se contruye el circuito?

Circuitos aritméticos

Circuitos aritméticos

- La ALU se puede implementar mediante Hardwiring
- Cada operación aritmética podría resolverse con un circuito

Circuitos aritméticos: Half adder

Objetivo Suma 2 bits

Entradas Los bits a sumar

Salida El bit resultado y el bit de carry

Circuitos aritméticos: Half adder

Tabla de verdad del Half adder

A	B	R	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Circuitos aritméticos: Half adder

Tabla de verdad del Half adder

A	B	R	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$\begin{array}{r} 0 \\ +\ 0 \\ \hline 0 \end{array}$$

Circuitos aritméticos: Half adder

Tabla de verdad del Half adder

A	B	R	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$\begin{array}{r} 1 \\ +\ 0 \\ \hline 1 \end{array}$$

Circuitos aritméticos: Half adder

Tabla de verdad del Half adder

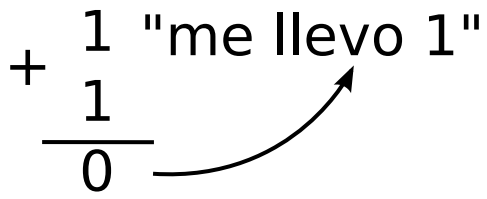
A	B	R	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$\begin{array}{r} 0 \\ +\ 1 \\ \hline 1 \end{array}$$

Circuitos aritméticos: Half adder

Tabla de verdad del Half adder

A	B	R	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



Circuitos aritméticos: Half adder

Fórmula del Half adder

Fórmula para el resultado:

A	B	R	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Fórmula para el carry:

A	B	R	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Circuitos aritméticos: Half adder

Fórmula del Half adder

Fórmula para el resultado:

A	B	R	C
0	0	0	0
0	1	1 $\overline{A} \wedge B$	0
1	0	1 $A \wedge \overline{B}$	0
1	1	0	1

Fórmula para el carry:

A	B	R	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Circuitos aritméticos: Half adder

Fórmula del Half adder

Fórmula para el resultado:

A	B	R	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$R = (\overline{A} \wedge B) \vee (A \wedge \overline{B})$$

Fórmula para el carry:

A	B	R	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Circuitos aritméticos: Half adder

Fórmula del Half adder

Fórmula para el resultado:

A	B	R	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$R = (\overline{A} \wedge B) \vee (A \wedge \overline{B})$$

Fórmula para el carry:

A	B	R	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1 $A \wedge B$

Circuitos aritméticos: Half adder

Fórmula del Half adder

Fórmula para el resultado:

A	B	R	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$R = (\overline{A} \wedge B) \vee (A \wedge \overline{B})$$

Fórmula para el carry:

A	B	R	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$C = (A \wedge B)$$

Circuitos aritméticos: Full Adder

Objetivo Suma 2 bits, considerando el carry anterior

Entradas Los bits a sumar, carry anterior

Salida El bit resultado y el bit de carry

Circuitos aritméticos: Full Adder

Tabla de verdad del Full adder

CAnt	*A*	*B*	*R*	*C*
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Circuitos aritméticos: Full Adder

Tabla de verdad del Full adder

CAnt	*A*	*B*	*R*	*C*
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

C anterior=0
0 + 0 = 0 C=0

C anterior=0
1 + 0 = 1 C=0

C anterior=0
0 + 1 = 1 C=0

C anterior=0
1 + 1 = 0 C=1

Circuitos aritméticos: Full Adder

Tabla de verdad del Full adder

CAnt	*A*	*B*	*R*	*C*
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

C anterior=0
$+\ \begin{matrix}0\\0\end{matrix}$
$\overline{}$ C=0
0

C anterior=0
$+\ \begin{matrix}1\\0\end{matrix}$
$\overline{}$ C=0
1

C anterior=0
$+\ \begin{matrix}0\\1\end{matrix}$
$\overline{}$ C=0
1

C anterior=0
$+\ \begin{matrix}1\\1\end{matrix}$
$\overline{}$ C=1
0

Circuitos aritméticos: Full Adder

Tabla de verdad del Full adder

CAnt	*A*	*B*	*R*	*C*
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

C anterior=0
$+\ \begin{matrix}0\\0\end{matrix}$
$\overline{}$ C=0
0

C anterior=0
$+\ \begin{matrix}1\\0\end{matrix}$
$\overline{}$ C=0
1

C anterior=0
$+\ \begin{matrix}0\\1\end{matrix}$
$\overline{}$ C=0
1

C anterior=0
$+\ \begin{matrix}1\\1\end{matrix}$
$\overline{}$ C=1
0

Circuitos aritméticos: Full Adder

Tabla de verdad del Full adder

CAnt	*A*	*B*	*R*	*C*
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

C anterior=0
$$\begin{array}{r} 0 \\ +\ 0 \\ \hline 0 \end{array}$$ C=0

C anterior=0
$$\begin{array}{r} 1 \\ +\ 0 \\ \hline 1 \end{array}$$ C=0

C anterior=0
$$\begin{array}{r} 0 \\ +\ 1 \\ \hline 1 \end{array}$$ C=0

C anterior=0
$$\begin{array}{r} 1 \\ +\ 1 \\ \hline 0 \end{array}$$ C=1

Circuitos aritméticos: Full Adder

Tabla de verdad del Full adder

CAnt	*A*	*B*	*R*	*C*
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

C anterior=1
0 + 0 = 1 C=0

C anterior=1
1 + 0 = 0 C=1

C anterior=1
0 + 1 = 0 C=1

C anterior=1
1 + 1 = 1 C=1

Circuitos aritméticos: Full Adder

Tabla de verdad del Full adder

CAnt	*A*	*B*	*R*	*C*
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

C anterior=1
0 + 0 = 1 C=0

C anterior=1
1 + 0 = 0 C=1

C anterior=1
0 + 1 = 0 C=1

C anterior=1
1 + 1 = 1 C=1

Circuitos aritméticos: Full Adder

Tabla de verdad del Full adder

CAnt	*A*	*B*	*R*	*C*
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

C anterior=1
0 + 0 = 1 C=0

C anterior=1
1 + 0 = 0 C=1

C anterior=1
0 + 1 = 0 C=1

C anterior=1
1 + 1 = 1 C=1

Circuitos aritméticos: Full Adder

Tabla de verdad del Full adder

CAnt	*A*	*B*	*R*	*C*
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

C anterior=1
0 + 0 = 1 C=0

C anterior=1
1 + 0 = 0 C=1

C anterior=1
0 + 1 = 0 C=1

C anterior=1
1 + 1 = 1 C=1

Circuitos aritméticos: Full adder

Fórmulas del Full adder

Fórmula para el resultado:

CAnt	*A*	*B*	*R*	*C*
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Fórmula para el carry:

CAnt	*A*	*B*	*R*	*C*
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Circuitos aritméticos: Full adder

Fórmulas del Full adder

Fórmula para el resultado:

$CAnt$	A	B	R	C
0	0	0	0	0
0	0	1	1 $\overline{CAnt}\wedge\overline{A}\wedge B$	0
0	1	0	1 $\overline{CAnt}\wedge A\wedge\overline{B}$	0
0	1	1	0	1
1	0	0	1 $CAnt\wedge\overline{A}\wedge\overline{B}$	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1 $CAnt\wedge A\wedge B$	1

Fórmula para el carry:

$CAnt$	A	B	R	C
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Circuitos aritméticos: Full adder

Fórmulas del Full adder

Fórmula para el resultado:

$CAnt$	A	B	R	C
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Fórmula para el carry:

$CAnt$	A	B	R	C
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	$1\ \overline{CAnt}^{\wedge} A^{\wedge} B$
1	0	0	1	0
1	0	1	0	$1\ CAnt^{\wedge}\overline{A}^{\wedge} B$
1	1	0	0	$1\ CAnt^{\wedge} A^{\wedge}\overline{B}$
1	1	1	1	$1\ CAnt^{\wedge} A^{\wedge} B$

Circuitos aritméticos

¿Cómo se suman cadenas de mas de un bit?

Circuitos aritméticos

¿Cómo se suman cadenas de mas de un bit?

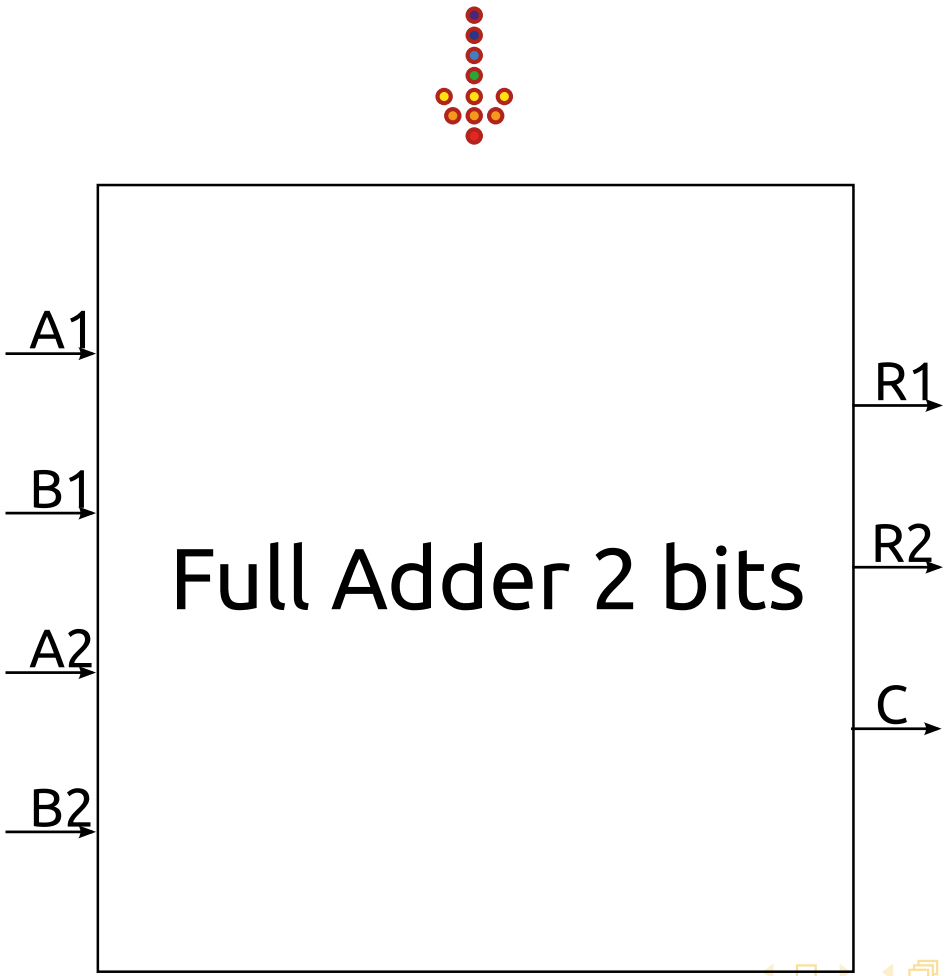
Usando múltiples Full Adders

Circuitos aritméticos

¿Cómo se suman cadenas de mas de un bit?

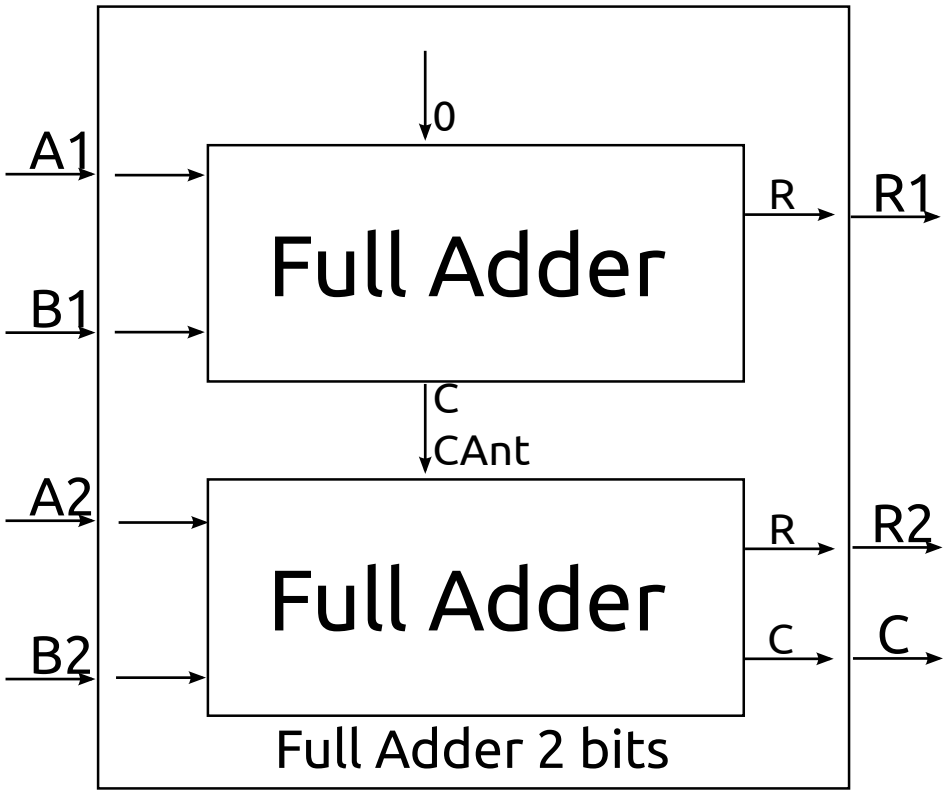
Circuitos aritméticos

¿Cómo se suman cadenas de mas de un bit?



Circuitos aritméticos

¿Cómo se suman cadenas de mas de un bit?



Circuitos aritméticos:restador de un bit

Completar la tabla de verdad

A	B	R	C

Circuitos aritméticos:restador de un bit

Completar la tabla de verdad

A	B	R	C
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Circuitos aritméticos:restador de un bit

Completar la tabla de verdad

A	B	R	C
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Completar las SOP

Redondeando...

¿Preguntas?