C12

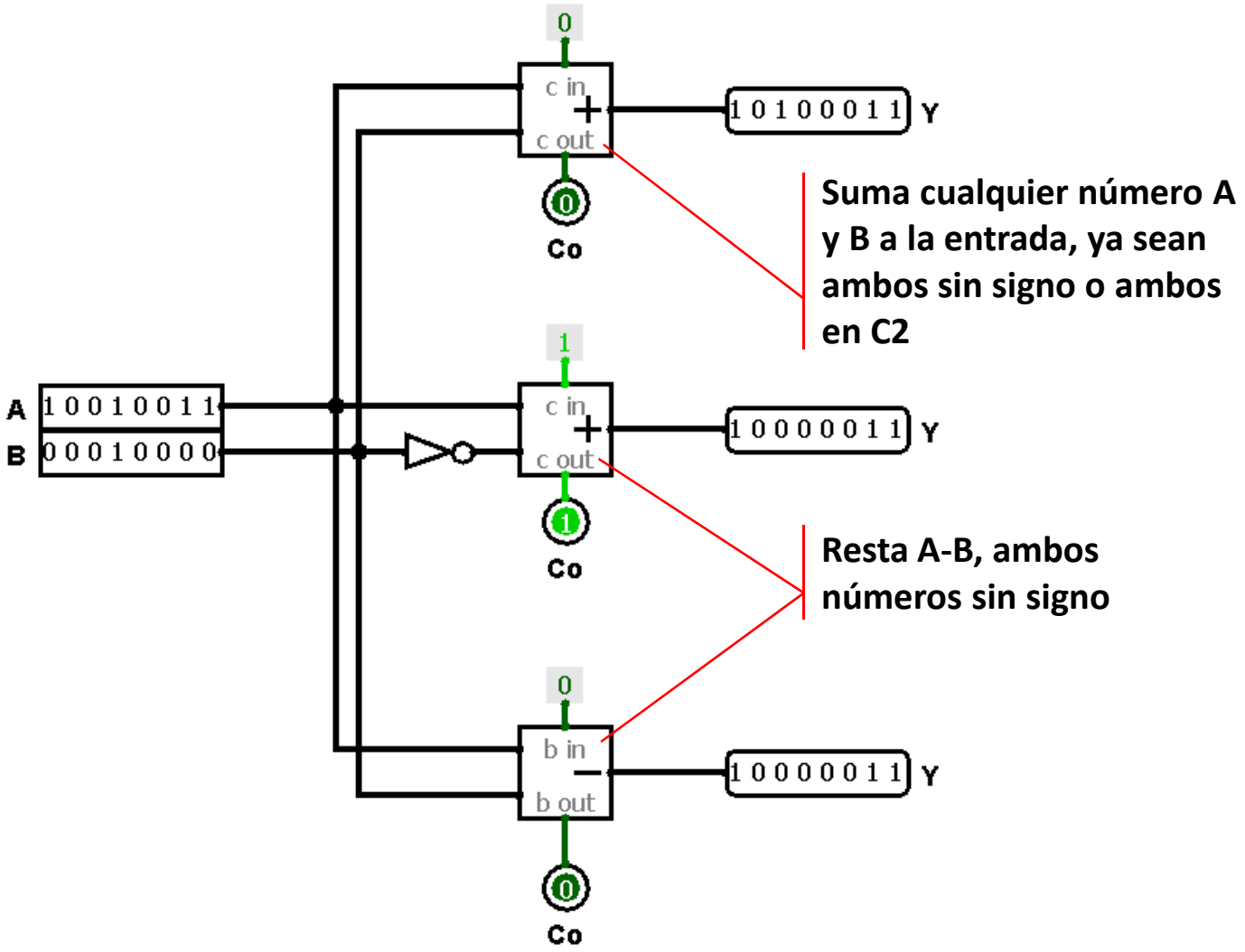
Combinacionales específicos

ALU

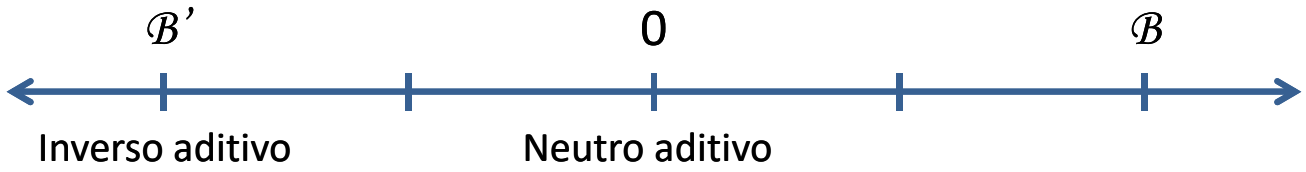
Índice

Restadores

Suma y resta


0
c in
c out
1 0 1 0 0 0 1 1
Y
0
Co
Suma cualquier número A y B a la entrada, ya sean ambos sin signo o ambos en C2
1
A
1 0 0 1 0 0 1 1
B
0 0 0 1 0 0 0 0
c in
c out
1 0 0 0 0 0 1 1
Y
1
Co
Resta A-B, ambos números sin signo
0
b in
b out
1 0 0 0 0 0 1 1
Y
0
Co

Inverso aditivo



Si existe un inverso aditivo del número $\mathcal{B}$ tal que

$$\mathcal{B} + \mathcal{B}' = 0 \rightarrow -\mathcal{B} = \mathcal{B}'$$

entonces puedo reescribir cualquier resta como una suma algebraica

$$\mathcal{A} - \mathcal{B} = \mathcal{A} + \mathcal{B}'$$

Por otra parte el complemento a 2 de un numero está definido como:

$$f_{C2}(\mathcal{B}) = 2^n - |\mathcal{B}|$$

Pero como en un sistema de representación de n bits: $2^n = 0$ entonces:

El complemento a la base de un número será su complemento aditivo.

$$f_{C2}(\mathcal{B}) = -\mathcal{B} \qquad \mathcal{A} - \mathcal{B} = \mathcal{A} + f_{C2}(\mathcal{B})$$

Decodificador (DECO)

Todos los posibles circuitos con 2 variables

Deco-4

T	F	L
0	0	
0	1	
1	0	
1	1	

S_1	S_0	D_0
0	0	1
0	1	
1	0	
1	1	

S_1	S_0	D_1
0	0	
0	1	1
1	0	
1	1	

T	F	L
0	0	1
0	1	1
1	0	
1	1	

S_1	S_0	D_2
0	0	
0	1	
1	0	1
1	1	

T	F	L
0	0	1
0	1	
1	0	1
1	1	

T	F	L
0	0	
0	1	1
1	0	1
1	1	

T	F	L
0	0	1
0	1	1
1	0	1
1	1	

S_1	S_0	D_3
0	0	
0	1	
1	0	
1	1	1

T	F	L
0	0	1
0	1	
1	0	
1	1	1

T	F	L
0	0	
0	1	1
1	0	
1	1	1

T	F	L
0	0	1
0	1	1
1	0	
1	1	1

T	F	L
0	0	
0	1	
1	0	1
1	1	1

T	F	L
0	0	1
0	1	
1	0	1
1	1	1

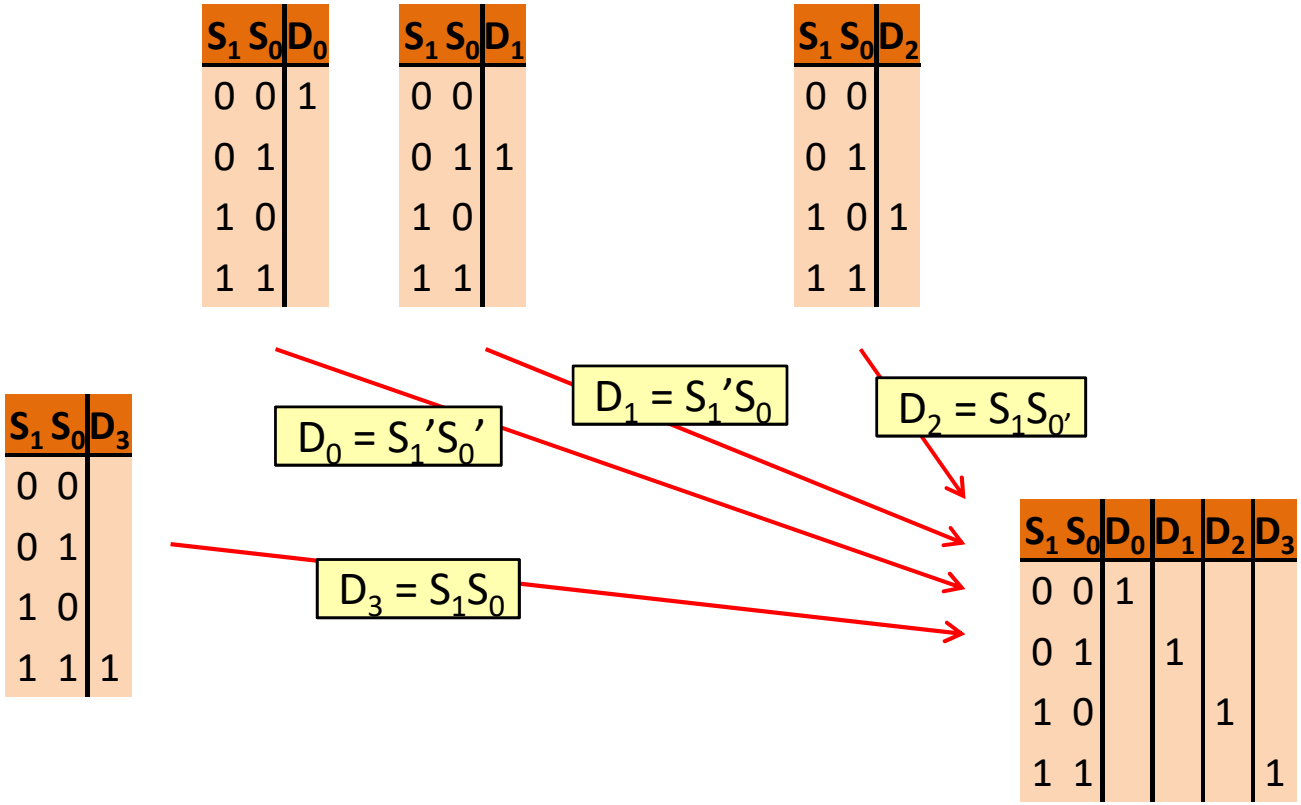
T	F	L
0	0	
0	1	1
1	0	1
1	1	1

T	F	L
0	0	1
0	1	1
1	0	1
1	1	1

$D_0 = S_1'S_0'$ $D_1 = S_1'S_0$ $D_2 = S_1S_0'$ $D_3 = S_1S_0$

Todos los posibles circuitos con 2 variables

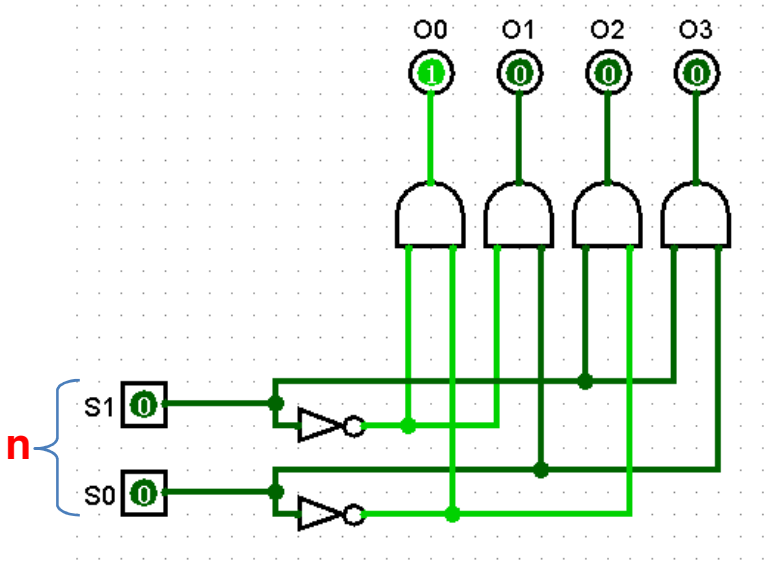
Deco-4

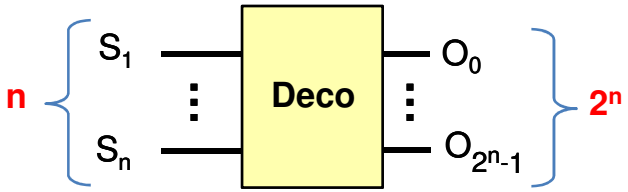


Decodificador

Los **decodificadores** poseen **n** entradas por donde reciben un número binario de n bits que van a decodificar en decimal activando una y solo una de las 2^n salidas posibles.

Son fundamentales para seleccionar una posición de memoria a partir de una dirección codificada en binario. También vamos a poder reconocerlos formando parte del circuito interno de un multiplexor.



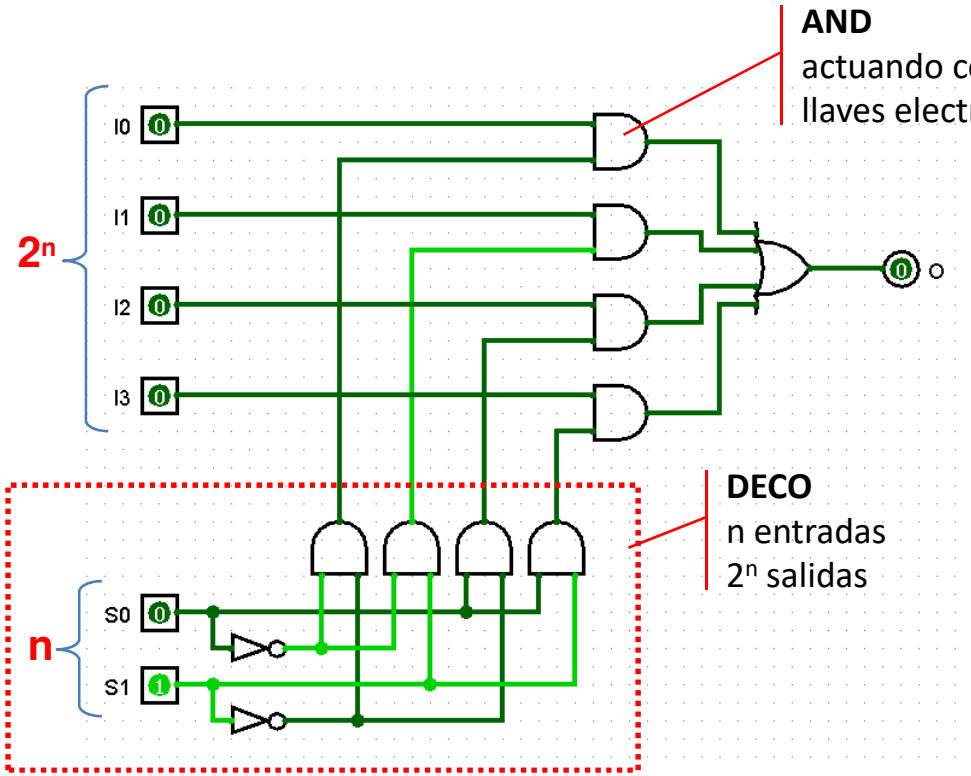


S_1	S_0	O_0	O_1	O_2	O_3
0	0	1			
0	1	0	1		
1	0	0		1	
1	1	0			1

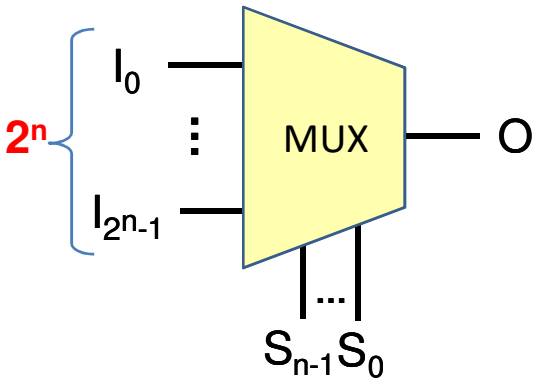
Multiplexor (MUX)

Multiplexor (MUX)

Selecciona **1** de entre 2^n entradas (I_0 … I_{2^n-1}) mediante **n** líneas de selección (S_0 … S_{n-1}) y la conecta a la única salida de datos (O)



S_1	S_0	O
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3



Demultiplexor (DEMUX)

Demultiplexor

Conectan la única entrada **I** con alguna de las **2^n** salidas **O** según la selección realizada mediante "**n**" líneas de selección

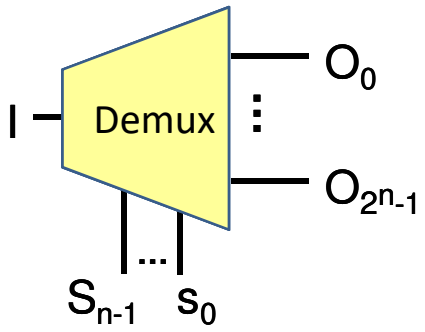
2 salidas → 1 línea de selección
4 salidas → 2 líneas de selección
8 salidas → 3 líneas de selección
2n salidas → n líneas de selección

TV completa

S_1	S_0	I	O_0	O_1	O_2	O_3
0	0	0	**0**			
0	0	1	**1**			
0	1	0		**0**		
0	1	1		**1**		
1	0	0			**0**	
1	0	1			**1**	
1	1	0				**0**
1	1	1				**1**

TV simplificada

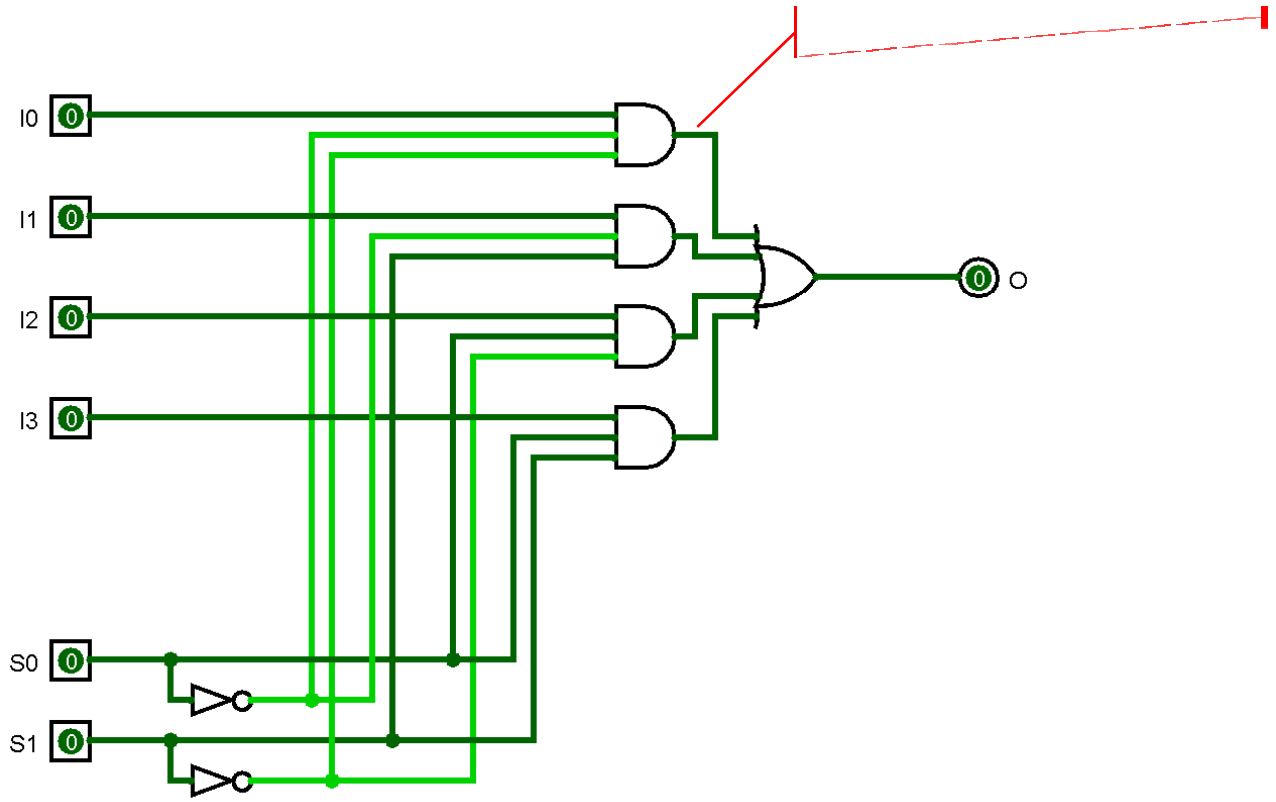
S_1	S_0	O_0	O_1	O_2	O_3
0	0	I			
0	1		I		
1	0			I	
1	1				I



Hay un DECO en cada MUX y en cada DEMUX

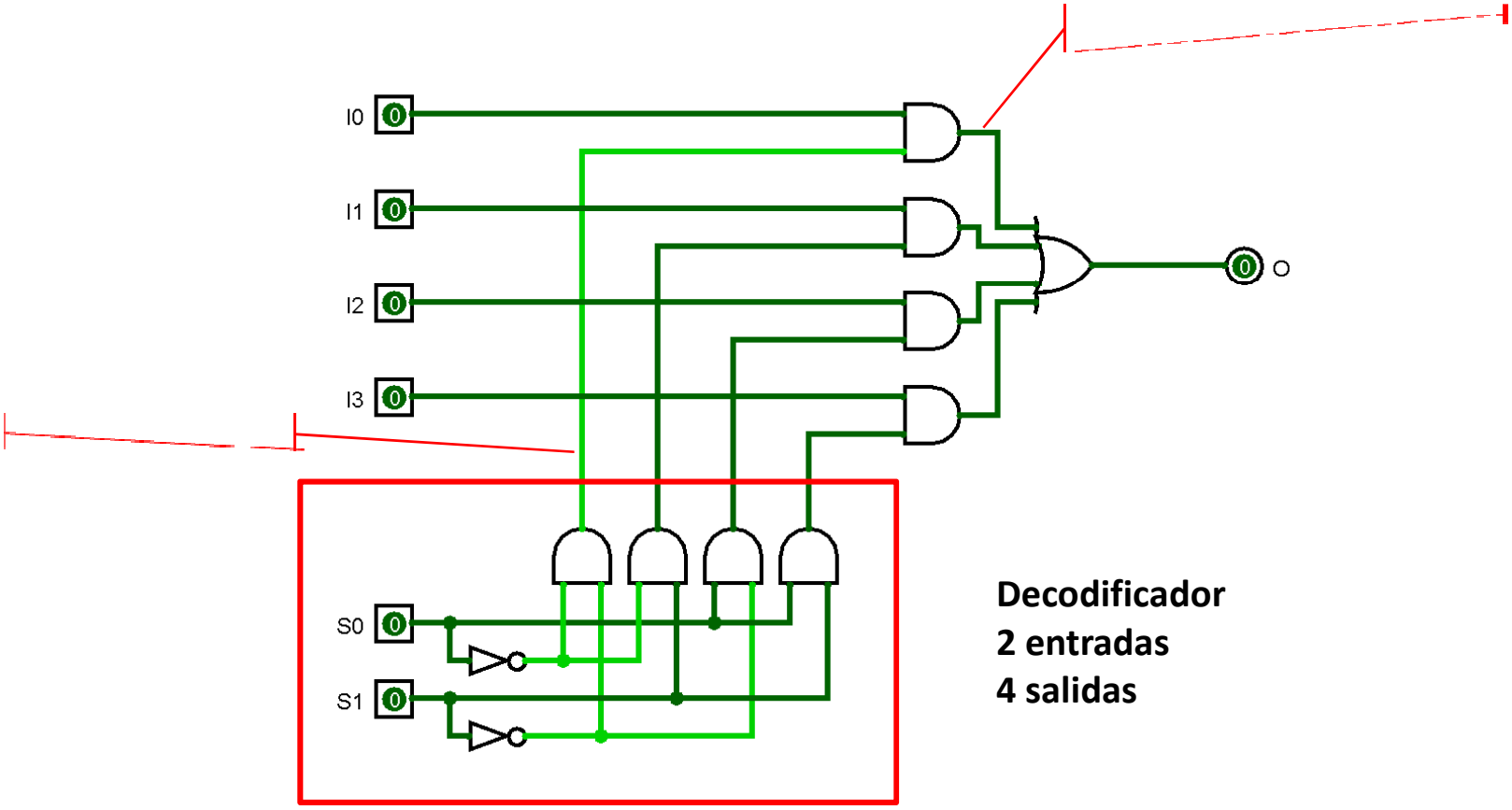
MUX 4 entradas – Ver. 1

$$O = I_0 \overline{S1}\,\overline{S0} + I_1 \overline{S1}\,S0 + I_2 S1\,\overline{S0} + I_3 S1\,S0$$



MUX 4 entradas – Ver. 2

$$O = I_0 (\overline{S1}\,\overline{S0}) + I_1 (\overline{S1}\,S0) + I_2 (S1\,\overline{S0}) + I_3 (S1\,S0)$$



Decodificador
2 entradas
4 salidas

Lógicas positiva y negativa

Lógica digital

Los circuitos digitales operan con
dos valores eléctricos (por ejemplo 5v y 0v)

Estos valores pueden ser interpretados como
1 y **0** y pensarlos como **información**

También pueden ser interpretados como
verdadero o falso y pensarlos como **valores lógicos**

Activo alto

Si 1 = Verdadero/True

Lógica positiva

Variable activa en alto

e inactiva en bajo

Activo bajo

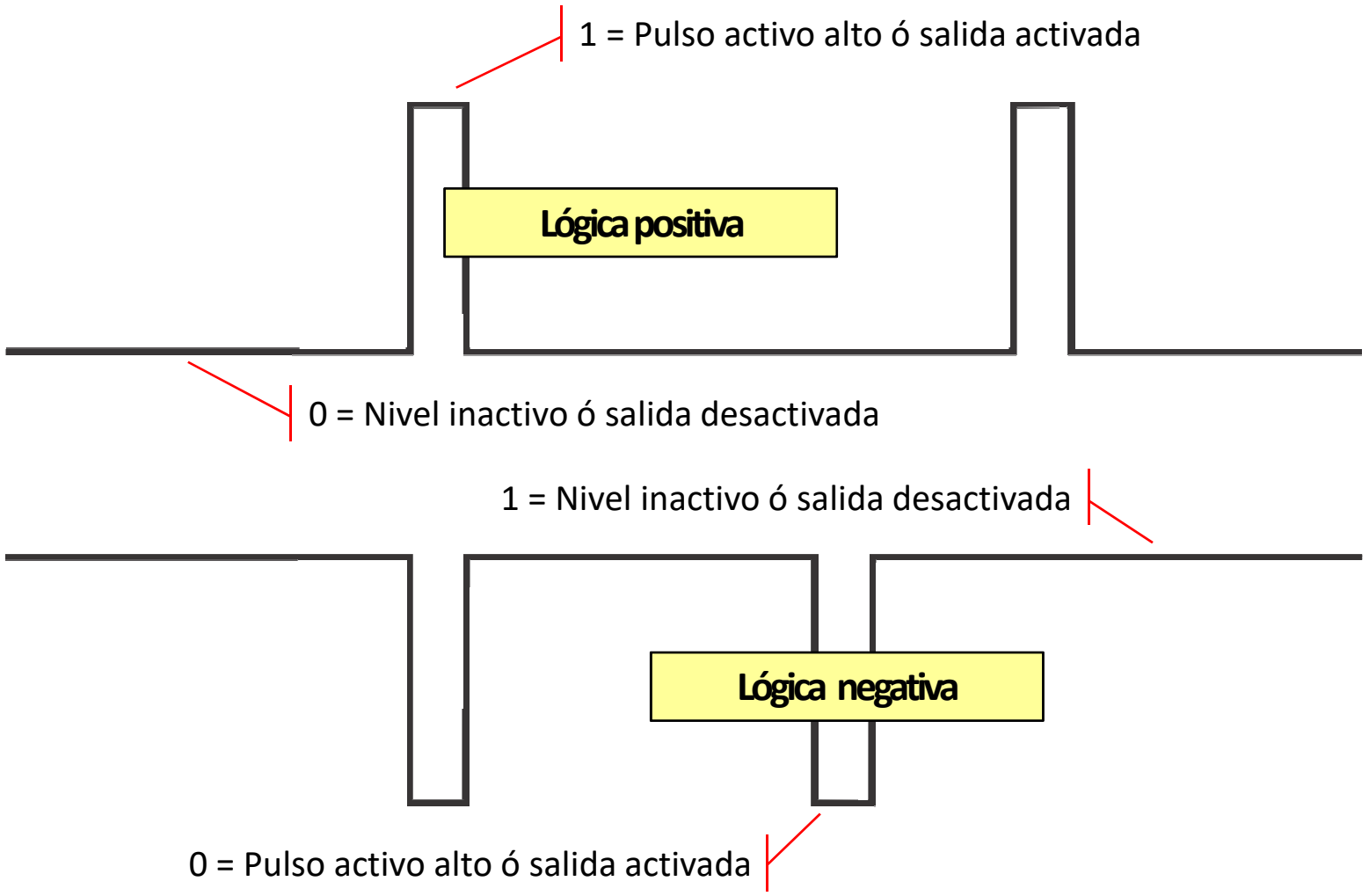
Si 0 = Verdadero/True

Lógica negativa

Variable activa en bajo

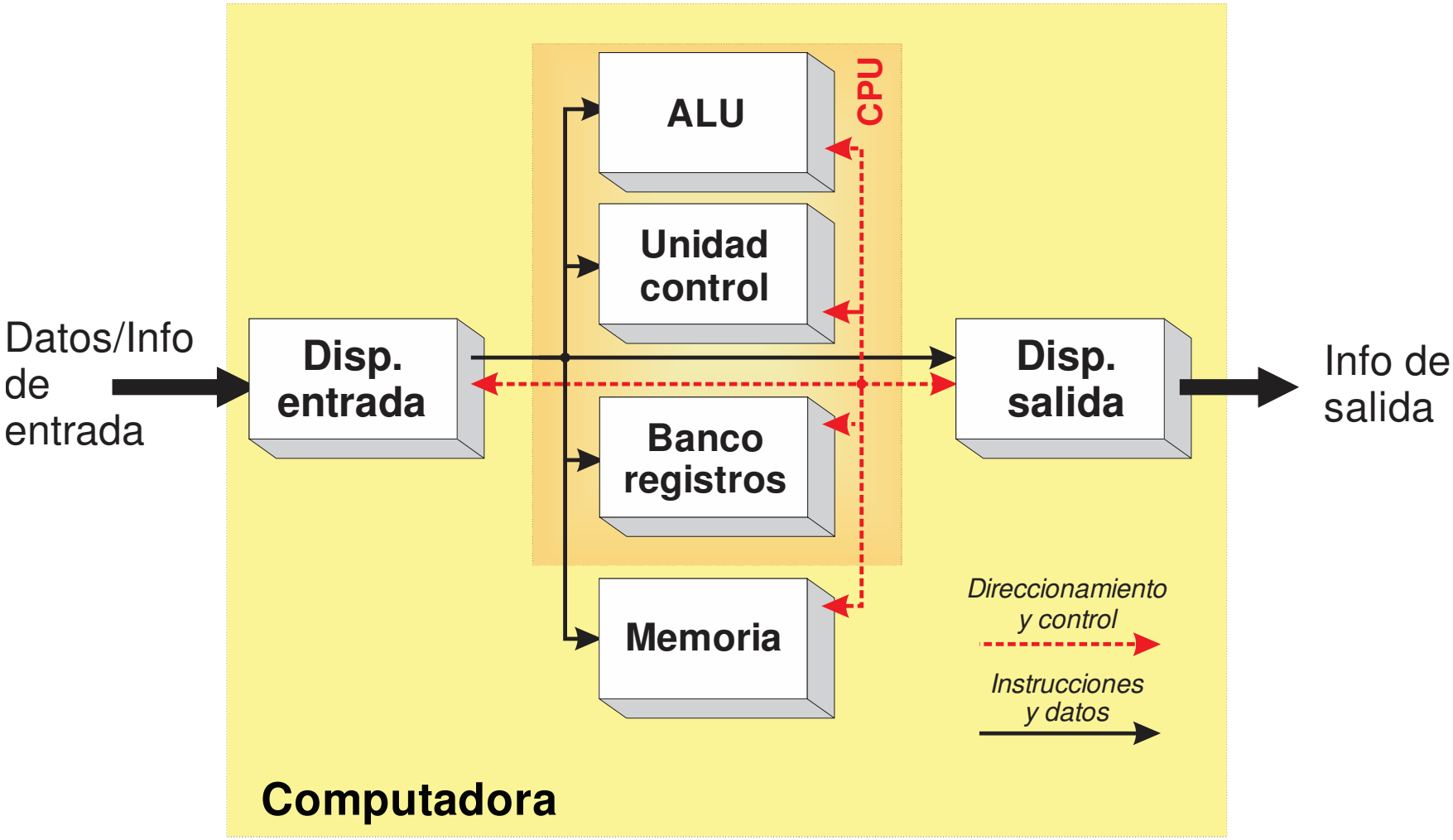
e inactiva en alto

Pulso


1 = Pulso activo alto ó salida activada
Lógica positiva
0 = Nivel inactivo ó salida desactivada
1 = Nivel inactivo ó salida desactivada
Lógica negativa
0 = Pulso activo alto ó salida activada

Unidades funcionales

Computadora



ALU: Unidad aritmético-lógica

ALU de “n” bits

La unidad aritmético-lógica (ALU) es el corazón de cualquier CPU ya que es allí donde se realizan todas las operaciones de dicho tipo.

Siempre tendrá dos entradas de “n” bits de ancho, que llamaremos **A** y **B** (por ejemplo) y por donde se ingresarán o el único operando de operaciones *unarias* o los dos operandos de operaciones *binarias*.

El ancho de palabra “n” dependerá de cada arquitectura en particular, pero son habituales valores de 16, 32, 64 y más bits. Una vez que la ALU realice la operación solicitada volcará el resultado en la salida **Y**, también de “n” bits.

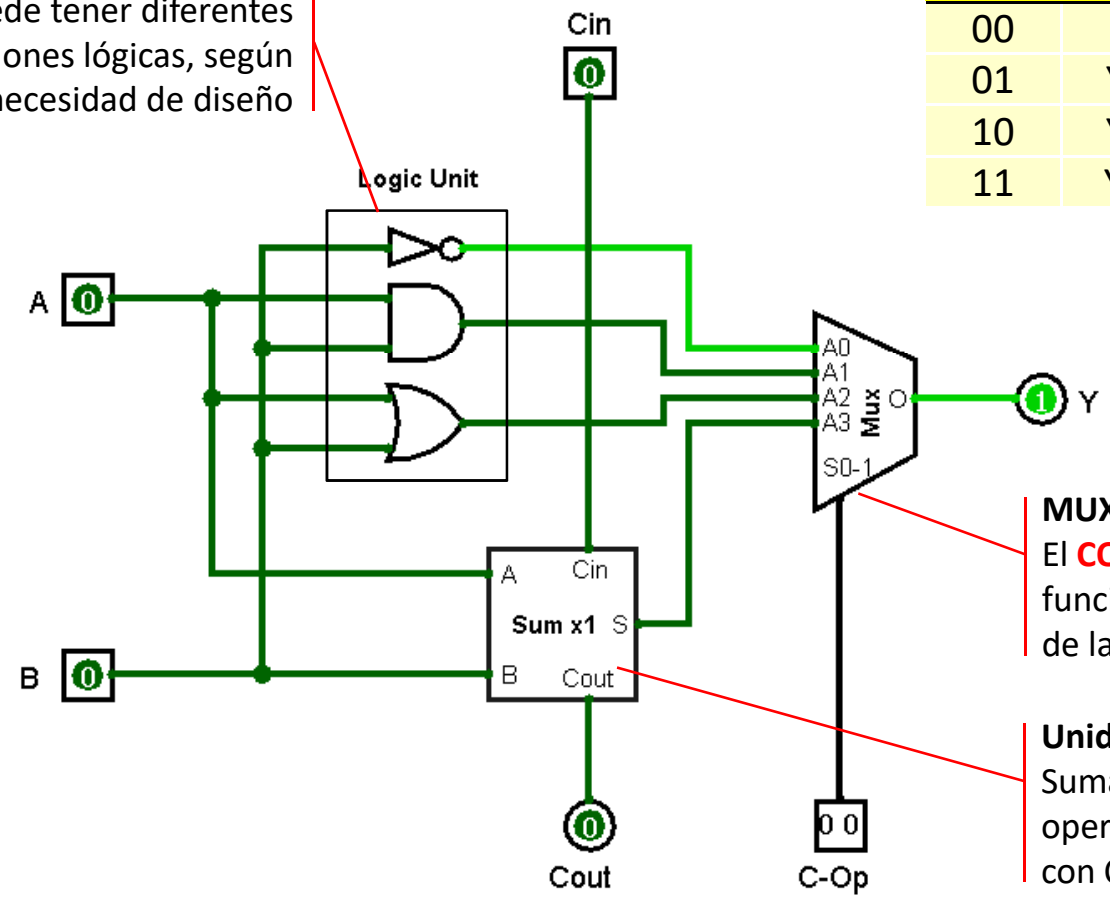
Necesariamente existirá otro grupo de entradas “COp” de ancho “m” bits por donde se le informará de manera codificada en binario que tipo de operación realizar, según surja de interpretar la instrucción en código de máquina que se esté ejecutando.

Además tendrá algunas salidas adicionales de 1 solo bit llamadas “banderas” o “flags” donde informará a la unidad de control algunas situaciones particulares de la última operación aritmético-lógica realizada, tal como: si hubo “overflow” al operar algebraicamente operandos sin signo (flag **C**) o en 2C2 (flag **V**), si el resultado fue “0” (flag **Z**), si el resultado fue negativo (flag **N**), si A fue mayor, menor o igual a B, etc..

ALU elemental de 1 bit (ALU-1 4Fn)

Unidad lógica
Puede tener diferentes funciones lógicas, según necesidad de diseño

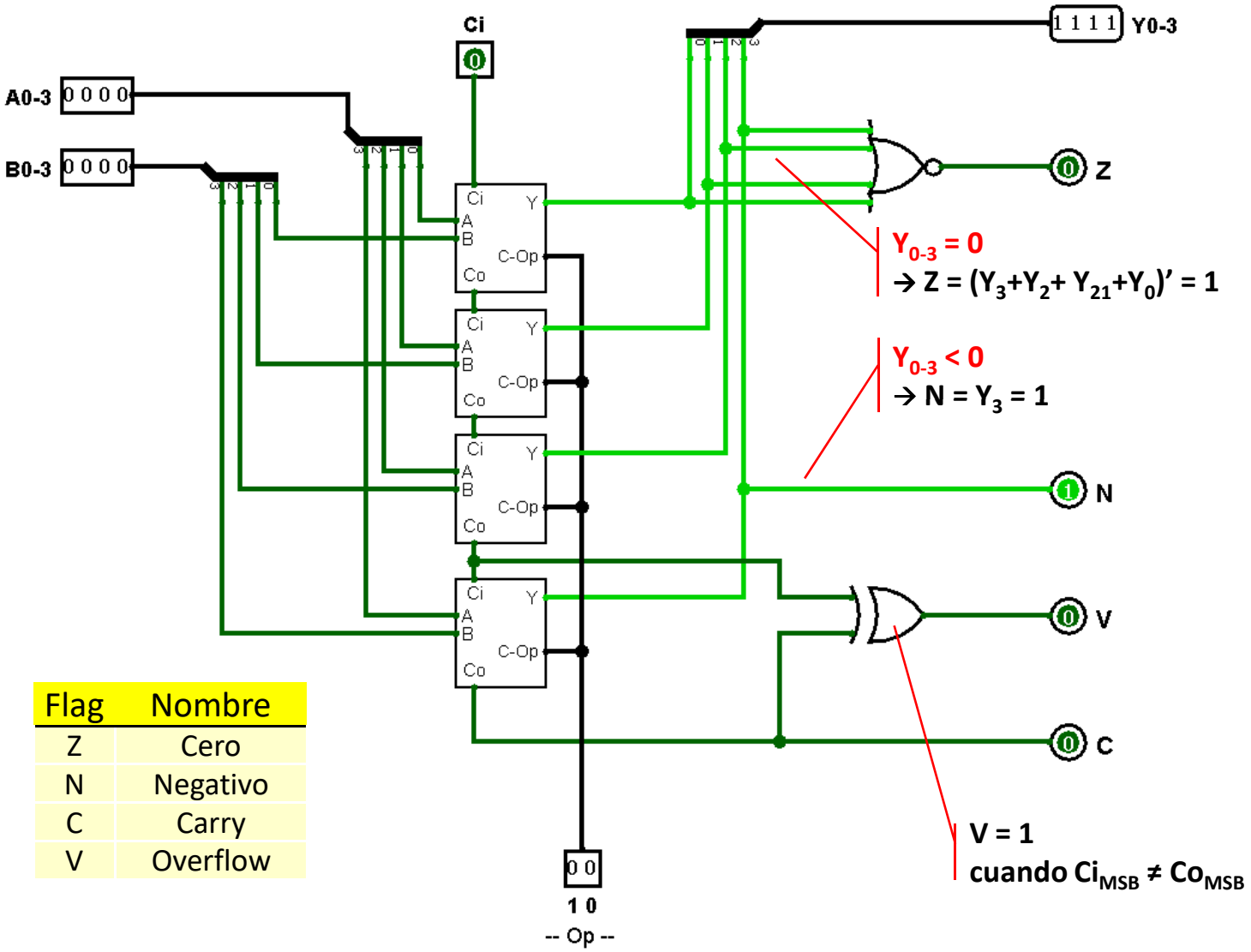
COp	Función	Tipo
00	Y=B’	Lógica
01	Y=A∧B	Lógica
10	Y=A∨B	Lógica
11	Y=A+B	Aritm.



MUX de salida de la ALU
El **COp** selecciona la función de salida a través de las entradas **S**

Unidad aritmética
Sumador para dos operandos de 1 bit con Carry in

La ALU multibit y sus Flags



Flag	Nombre
Z	Cero
N	Negativo
C	Carry
V	Overflow

Resumen de Flags

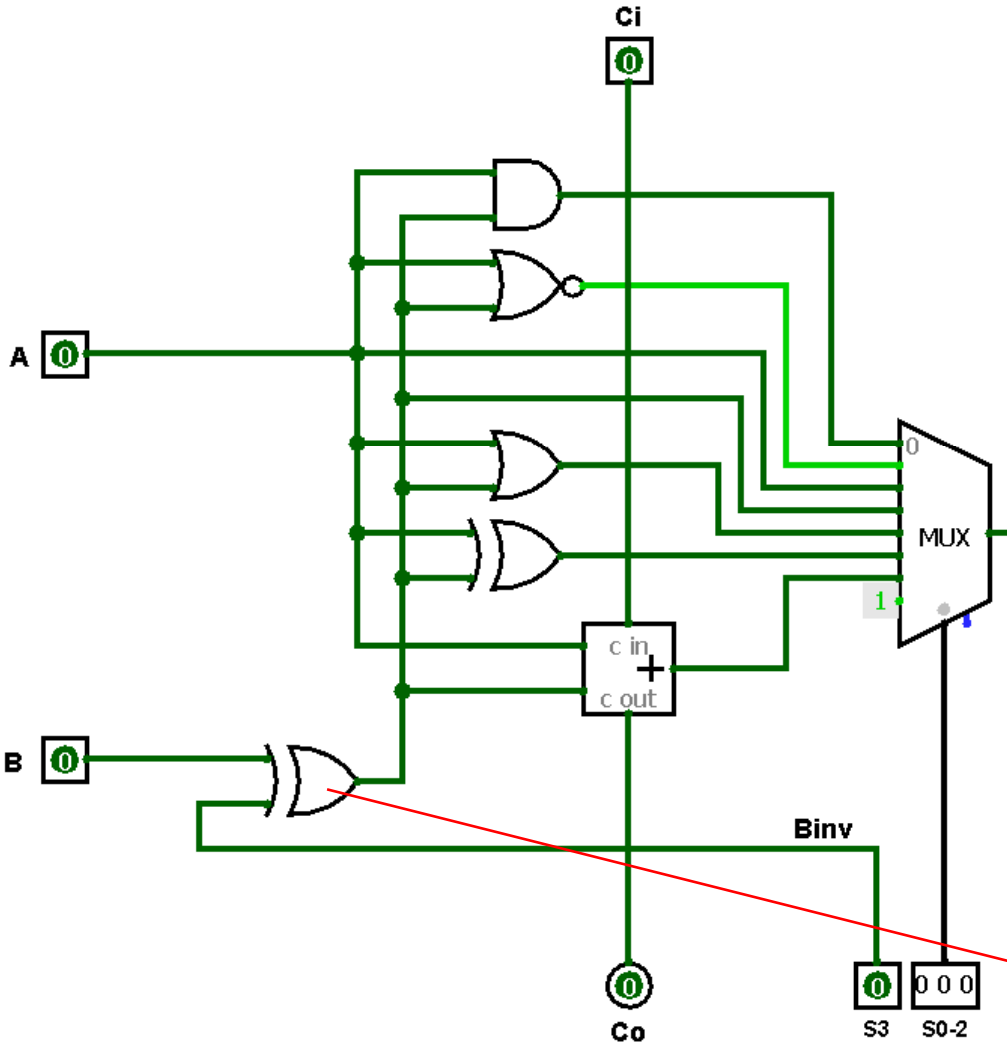
Al calcular los flags sobre la salida de la ALU los mismos se activarán tanto cuando esta realice operaciones aritméticas como lógicas.

C, V y N solo tienen sentido para operaciones aritméticas, que operan con datos multibit o palabras de datos (word-wise). No tiene sentido hablar de negativo o de desborde si el dato de salida no es numérico.

Z podría ser útil tanto en operaciones aritméticas como lógicas (bit-wise), ya que además de necesitar saber si una operación aritmética dio cero también nos podrías ser útil saber si todos los bits de una operación lógica son cero

ALU-1 16Fn

ALU-1 16 funciones



COp	Función	Tipo
0000	Y=A∧B	Lóg
0001	Y=(A∨B)’	Lóg
0010	Y=A	Lóg
0011	Y=B	Arit
0000	Y=A∨B	Lóg
0101	Y=A⊕B	Log
0110	Y = A+B+Cin	Arit
0111	Y=1	—
1000	Y=A∧B’	Lóg
1001	Y=(A∨B’)’	Lóg
1010	Y=A	Lóg
1011	Y=B’	Arit
1000	Y=A∨B’	Lóg
1101	Y=A⊕B’	Log
1110	Y = A+B’+Cin	Arit
1111	Y=1’	—

Inversor controlado
El MSB del **COp** selecciona si se niega o no la entrada B