Apunte de circuitos

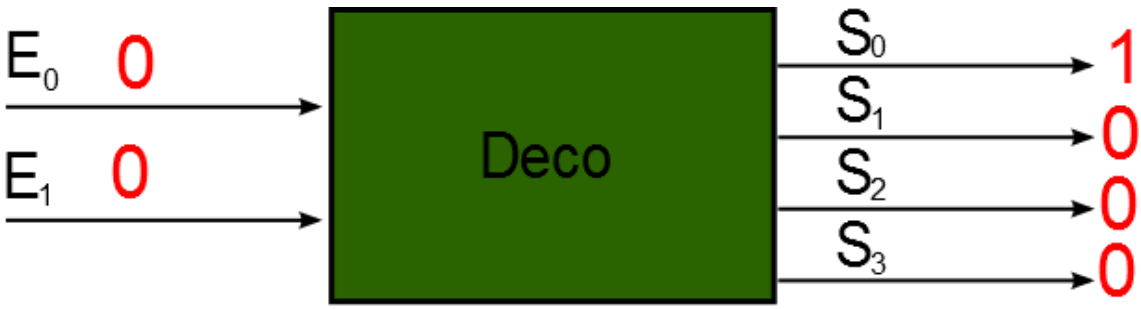
En este apunte veremos cómo armar algunos circuitos comunes. Estos circuitos no tienen nada de especial con respecto a los vistos en la última clase, pero tienen nombre y apellido porque se usan en varios contextos o bien porque computan funciones útiles.

Decodificador:

Un decodificador es un circuito que tiene 2 entradas y en base al valor que ingresa por ellas se activa una de las cuatro salidas. La idea es que si llamamos ***e1e0*** al valor de la entrada, que puede ser 00, 01, 10, 11, y llamamos ***s0, s1, s2 y s3*** a las salidas; la salida que se va a activar es la ***si*** tal que si interpretamos el valor de la entrada, obtenemos ***i***. En criollo, si las entradas valen 0 y 0, se activa la salida 0; si valen 0 y 1, se activa la salida 1, si valen 1 y 0, la salida 2 y si valen 1 y 1, la salida 3.

Por ejemplo, podemos pensar que un deco es algo así:

Si E0 y E1 valen 0, la cadena de entrada es 00. El valor de 00 es 0, por lo cual se activa la salida 0.



Si E0 y E1 valen 1, la cadena de entrada es 11. El valor de 11 es 3, por lo cual se activa la salida 3.

A partir de esto, podemos construir la tabla de verdad de este circuito. La misma va a tener 6 columnas y 4 filas:

E1	E0	S0	S1	S2	S3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Para construir el circuito, deberíamos obtener la formula, por ejemplo usando suma de productos. Hasta ahora, todos nuestros circuitos tenían una única salida; pero el decodificador tiene 4. ¿Cómo hacemos para construir el circuito entonces? Fácil, obtenemos la fórmula que le corresponde a cada salida por separado y luego usando esas fórmulas armamos el circuito.

La fórmula de cada salida es (usando suma de productos):

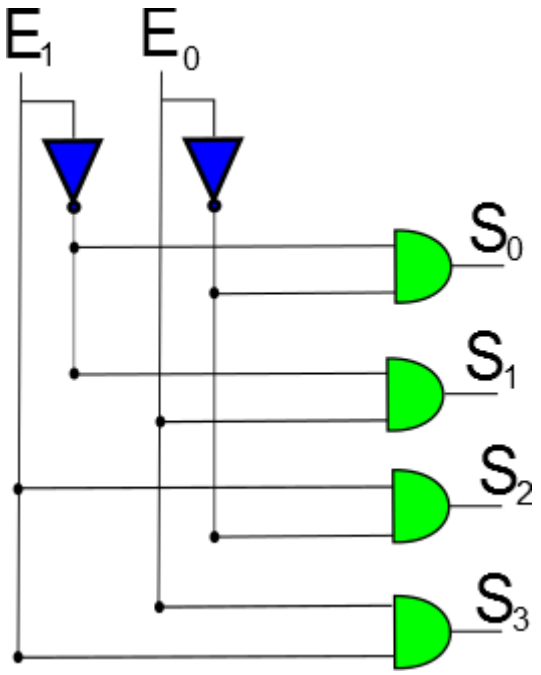
S0 = ¬E1 * ¬E0

S1 = ¬E1 * E0

S2 = E1 * ¬E0

S3 = E1 * E0

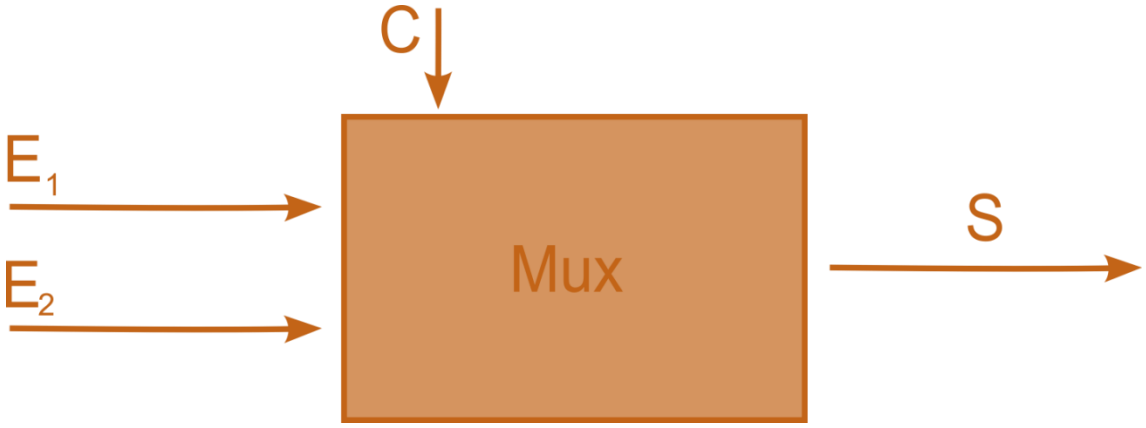
Usando la formula, podemos armar el circuito como vimos en clase:



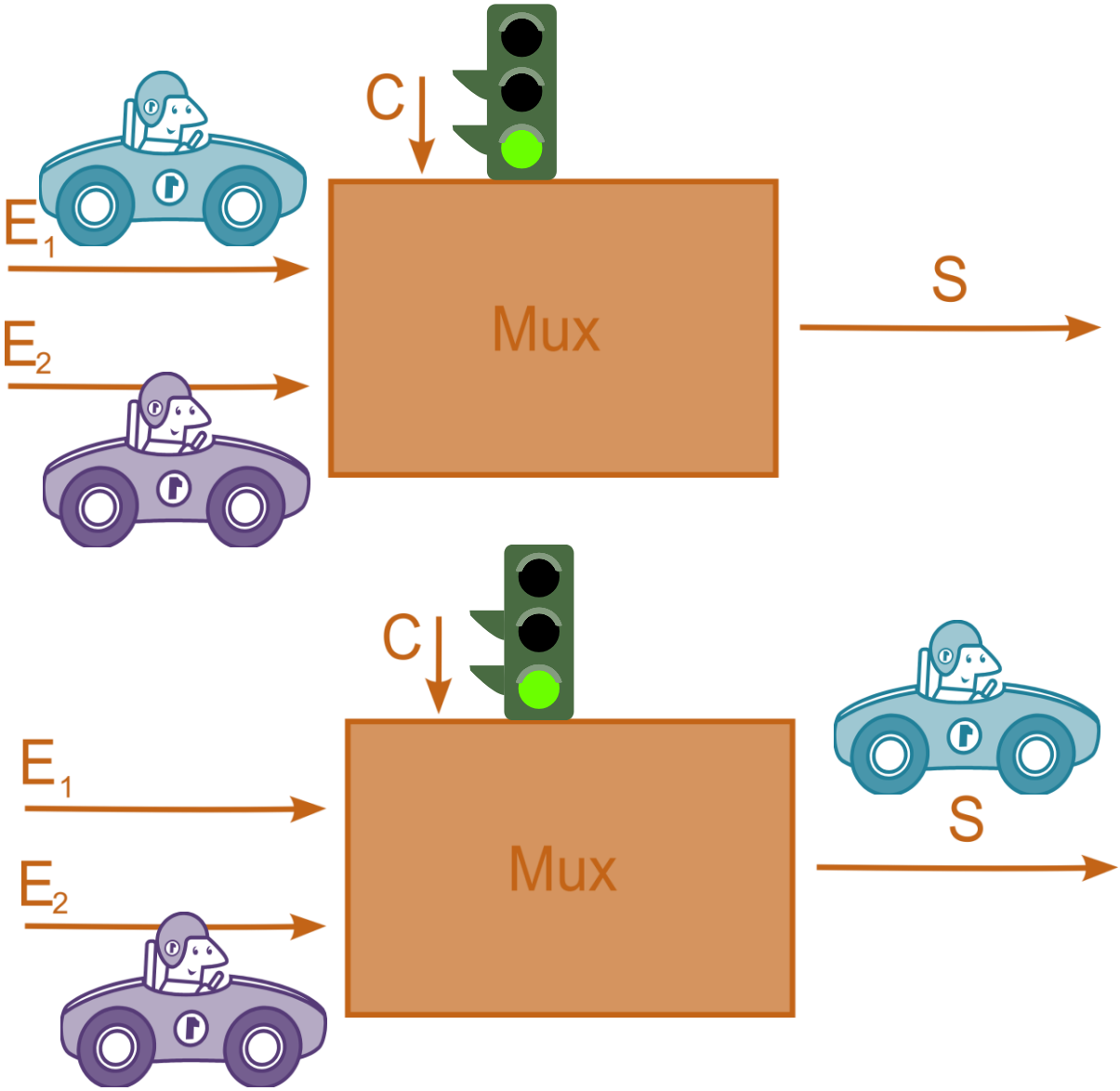
Notar que del circuito salen 4 cables de salida. Eso es correcto porque el circuito tiene 4 salidas. No hay que unirlas con un or, un and, xor, nada de nada; porque son efectivamente 4 salidas distintas.

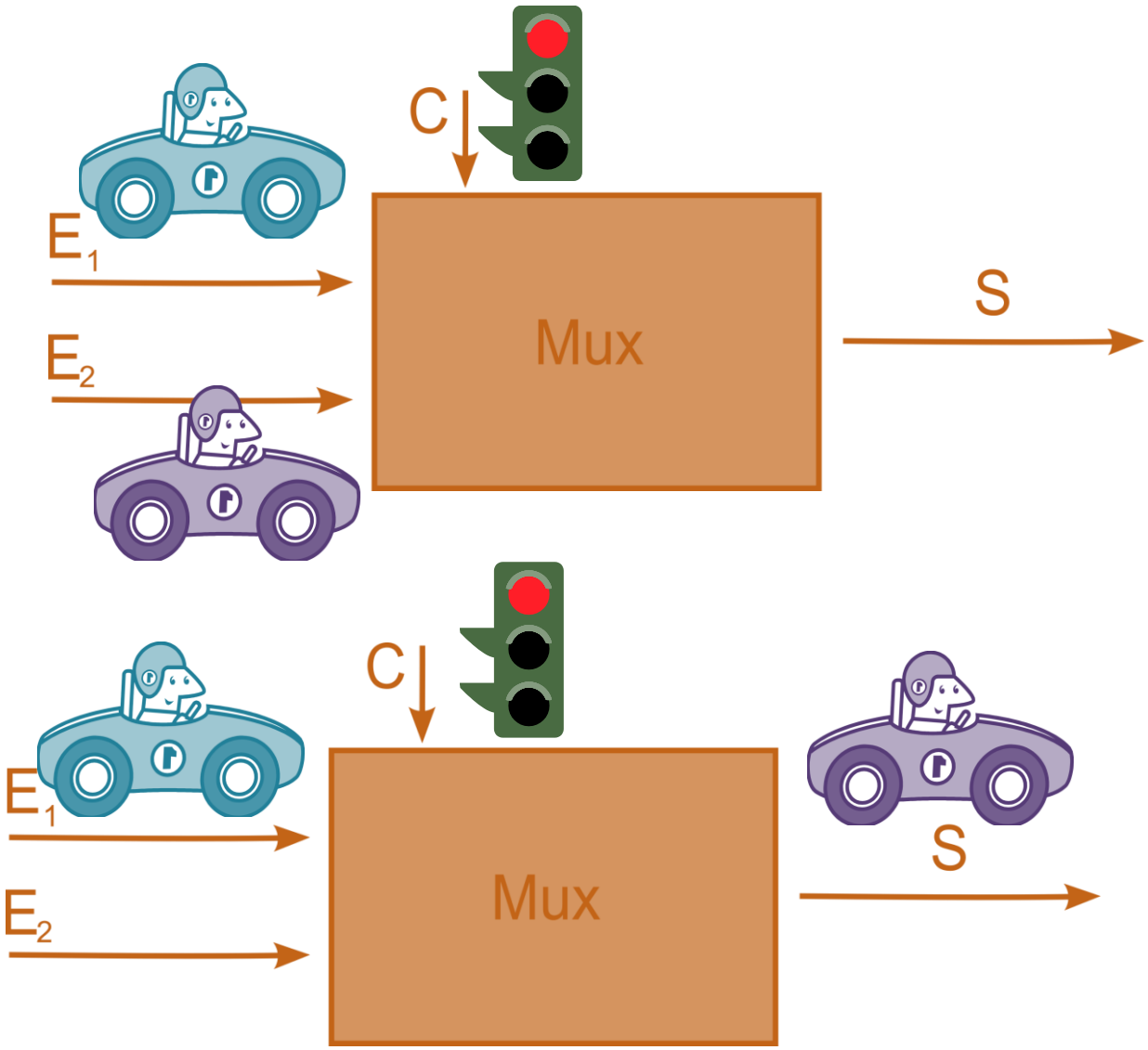
Multiplexor:

Un multiplexor es un circuito que tiene 3 entradas: 2 de datos y una llamada de control (solo son nombres, son entradas comunes y corrientes) y 1 salida. La idea es que la línea de control decide cuál de las líneas de entrada es la que sale:



Podemos pensar la línea de control como un semáforo. Cuando esté en verde, va a pasar lo que está en E1 y cuando esté en rojo va a darle paso a lo que está en E2





Podríamos pensar en una tabla de verdad abreviada con la siguiente pinta:

C	S
0	E1
1	E2

Esta tabla no es la tabla completa que nos va a permitir armar el circuito. Construyamos la tabla completa:

C	E1	E2	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

A partir de esta tabla, podemos armar la formula (por ejemplo con suma de productos) y luego el circuito:

S = ¬C * E1 * ¬E2 + ¬C * E1 *E2 + C*¬E1 *E2 + C*E1*E2

Una vez que tenemos la formula, podemos armar el circuito. Esto lo dejamos de tarea.

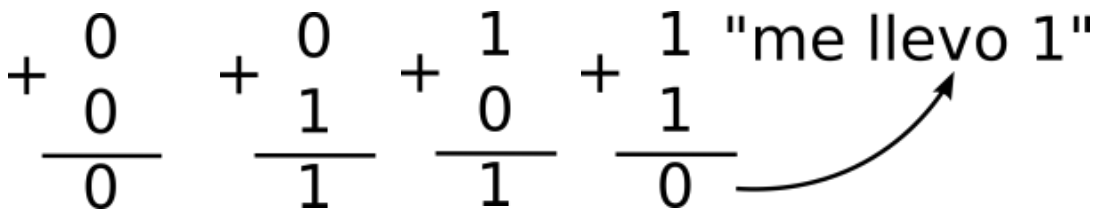
Circuitos Aritméticos:

Llamamos circuitos aritméticos a aquellos que implementan funciones aritméticas, como por ejemplo la suma. Tengamos en cuenta que son circuitos como cualquier otro, pero la función que calculan tiene un interés particular para nosotros.

Half adder

Llamamos half adder al circuito que dado dos bits de entrada, calcula el resultado de la suma y el carry (es decir, el "me llevo uno").

Cuando vimos la suma, planteamos que existen 4 casos distintos:



En base a estos casos, podemos construir la tabla de verdad:

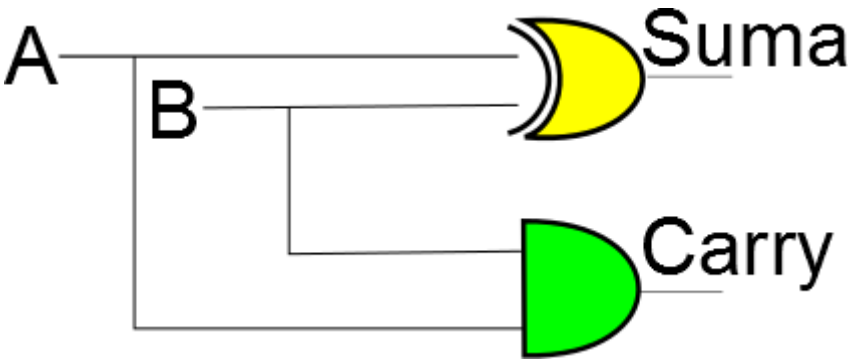
A	B	Suma	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Armemos las formulas:

Carry = A * B

Suma = ¬A*B + A*¬B → Esto es igual a A *xor* B (Probar!)

Con esta fórmula podemos armar el circuito:



El problema con el half adder es que muchas veces cuando sumamos dos números, tenemos que tener en cuenta que el anterior nos pudo pasar 1, y ese 1 hay que sumarlo.

Full adder

Un full adder implementa la suma entre dos bits, pero agregando una entrada más que representa el carry de entrada. Las salidas nuevamente son dos: suma y carry.

Pensemos entonces la tabla de verdad:

A	B	Carry de entrada	Suma	Carry de salida
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

Armemos las formulas:

Suma = ¬A*¬B*Carry_de_entrada + ¬A*B*¬Carry_de_entrada + A*¬B*¬Carry_de_entrada + A*B*Carry_de_entrada

Carry_de_salida = ¬A*B*Carry_de_entrada + A*¬B*Carry_de_entrada + A*B*¬Carry_de_entrada + A*B*Carry_de_entrada

Dejamos como tarea armar el circuito y responder la siguiente pregunta: El full adder nos permite sumar dos números de 1 bit. ¿Cómo podríamos hacer si quisiéramos sumar números de 2 bits? ¿Y de 3? ¿Y de 32 bits?