Universidad Nacional Autónoma de México

Facultad de Ciencias

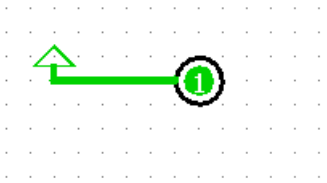
Organización y Arquitectura de Computadoras

Tarea 04

Arriaga Santana Estela Monserrat
Castañon Maldonado Carlos Emilio
Fernández Blancas Melissa Lizbeth

1 De las siguientes funciones dibuje los diagramas lógicos correspondientes.

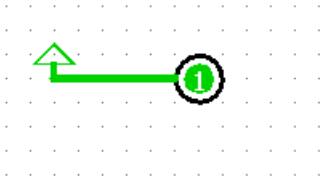
- $\neg(p \wedge q) \leftrightarrow \neg p \vee \neg q$
  El diagrama lógico se ve así



  Esto se debe a que es una tautología

p	q	¬	(	p	∧	q	)	↔	(	¬	p	∨	¬	q	)
1	1	0	1	1	1		1		0	1	0	0	1		
1	0	1	1	0	0		1		0	1	1	1	0		
0	1	1	0	0	1		1		1	0	1	0	1		
0	0	1	0	0	0		1		1	0	1	1	0		

- $p \vee (p \wedge q) \leftrightarrow p$
  El diagrama lógico se ve así



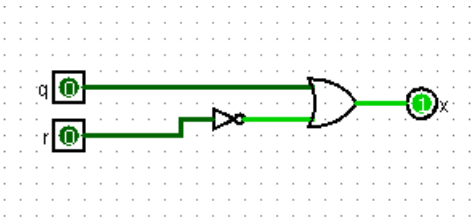
  Esto se debe a que es una tautología

p	q	(	p	∨	(	p	∧	q	))	↔	p
1	1		1	1		1	1	1		1	1
1	0		1	1		1	0	0		1	1
0	1		0	0		0	0	1		1	0
0	0		0	0		0	0	0		1	0

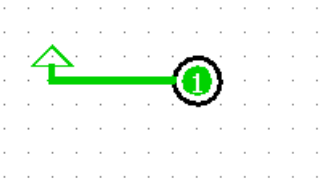
- $p \wedge q \vee r \rightarrow q$
  Al minimizar la expresión obtenemos que

P \ QR	00	01	11	10
0	1	0	1	1
1	1	0	1	1

  Que significa $q + \overline{r}$, así que el diagrama lógico se ve



- $((p \to q) \wedge (q \to r)) \to (p \to r)$
  El diagrama lógico se ve así



  Esto se debe a que es una tautología

p	q	r	((p	→	q)	∧	(q	→	r))	→	(p	→	r)
1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	0	1	1	1	0	1	0	0	1	1	0	0
1	0	1	1	0	0	0	0	1	1	1	1	1	1
1	0	0	1	0	0	0	0	1	0	1	1	0	0
0	1	1	0	1	1	1	1	1	1	1	0	1	1
0	1	0	0	1	1	0	1	0	0	1	0	1	0
0	0	1	0	1	0	1	0	1	1	1	0	1	1
0	0	0	0	1	0	1	0	1	0	1	0	1	0

2 **Dibuje los diagramas lógicos para AND, OR y NOT usando compuertas NAND.**

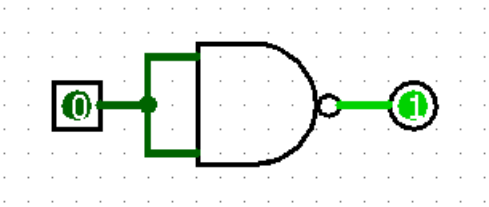
Notemos que el el NAND tiene la siguiente tabla de verdad:

p	q	p	|	q
1	1	1	0	1
1	0	1	1	0
0	1	0	1	1
0	0	0	1	0

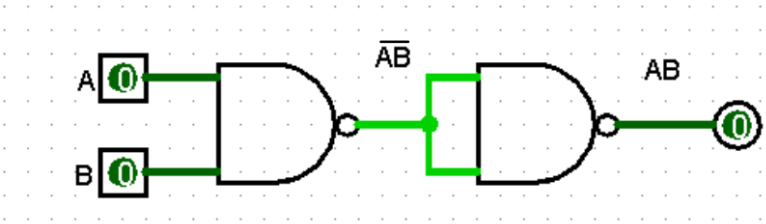
- NOT: Notemos que al conectar la entrada que queremos negar a las dos entradas del NAND tendremos los casos:

p	p	|	p
1	1	0	1
0	0	1	0

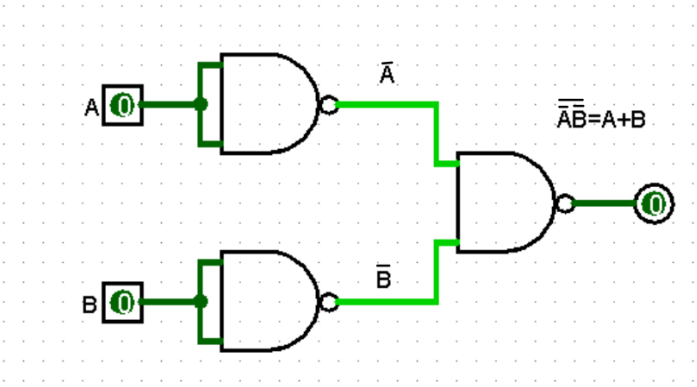
  Lo cual es equivalente al not, por lo tanto, el diagrama lógico se ve así:



- AND: Sabemos que $pNANDq = \overline{pq}$, por lo que para obtener pq colocaremos una compuerta NAND y la salida la negaremos usando el NOT que hicimos anteriormente.



- OR: Sabemos que $p + q = \overline{(\bar{p}\bar{q})} = \bar{p}Nand\bar{q}$, por lo que negaremos primero a p y a q usando el NOT realizado anteriormente y con esto haremos un NAND de la siguiente manera:



3 **Un circuito puede enviar una señal para cambiar de color una serie de leds de un semáforo: "R" para rojo y "G" para verde. Este circuito cuenta con cuatro entradas, una independiente de cambio de color "r", "g" y un interruptor "e" que de acuerdo a las siguientes condiciones:**

- Si se pulsa un solo color, el encendido del color depende de "e" de forma que:
  - Si "e" esta activado, el led pulsado esta encendido
  - Si "e" esta desactivado, el led pulsado esta apagado
- Si se pulsa dos botones al mismo tiempo, el color depende de "e" de forma que:
  - Si "e" esta activado, el led verde esta encendido
  - Si "e" esta desactivado, el led rojo esta encendido

Establece la tabla de verdad, las funciones lógicas y simplificadas, y su circuito lógico mediante compuertas.

Haciendo la tabla de verdad usando las condiciones dadas obtenemos lo siguiente.

r	q	e	R	G
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	0	1
1	0	0	0	0
1	0	1	1	0
1	1	0	1	0
1	1	1	0	1

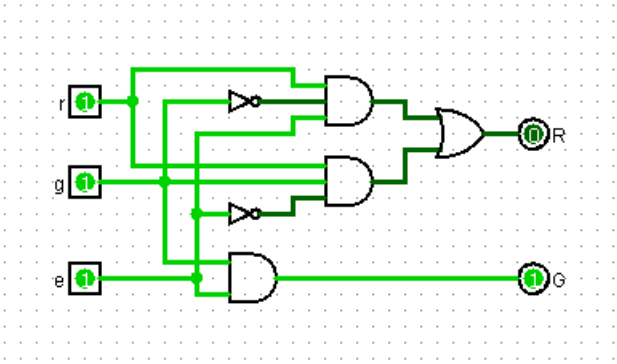
Para obtener las expresiones reducidas utilizaremos mapas de Karnaugh de la siguiente manera.
Para R:

R \ GE	00	01	11	10
0	0	0	0	0
1	0	1	0	1

Entonces $R = r\bar{g}e + rg\bar{e}$
Para G:

R \ GE	00	01	11	10
0	0	0	1	0
1	0	0	1	0

Entonces $G = ge$
Por lo que el circuito lógico es



4 **La computadora de un coche tiene tres medidores en caso de que falte aceite "A", el nivel de las llantas este bajo "N" y el calor del motor se eleve a niveles peligrosos "T". Todas estas son alertas que, de no reportarse podrían causar un fallo grave al sistema general del automóvil. Diseñe un circuito de control mediante compuertas lógicas que cumpla las siguientes condiciones de funcionamiento:**

- Si no se activa ninguna alerta, el motor se enciende.
- Si se enciende la alerta del calor del motor, el motor se apaga, pero una lampara adicional de emergencia sé enciende también.
- Si se encienden cualesquiera dos alertas, el motor se enciende y la lampara de emergencia.
- Si se enciende solo una alerta que no sea del calor del motor, el motor se enciende pero no se activa la lámpara de emergencias.
- Si todas las alertas se encienden, el motor se apaga y la lampara se enciende.

Establece la tabla de verdad, las funciones lógicas y simplificadas, y su circuito lógico mediante compuertas.

Notemos que en el circuito requerido tenemos tres entradas, "A", "N" y "T" que nos darán dos salidas, las cuales serán el que el motor se encienda y el que se encienda la lampara adicional de emergencia. Por lo cual tendremos dos funciones que dependen de las mismas tres variables pero que cuyo resultado no interviene en el resultado de la otra función.

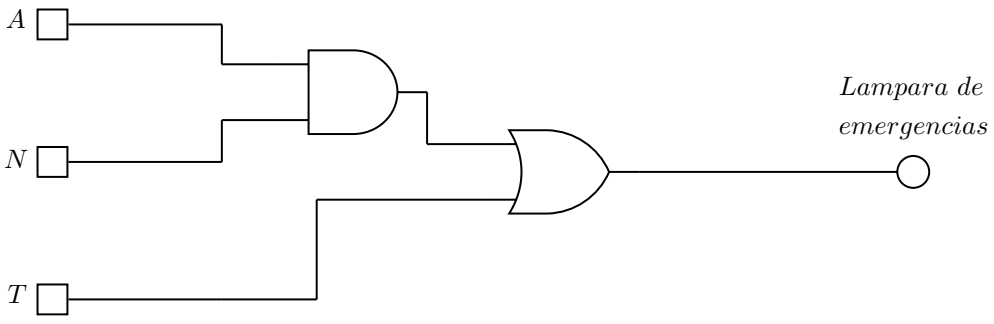
Dicho esto, empezaremos con el diseño de la función que prende o apaga la lámpara adicional de emergencia. Notemos por el segundo enunciado sin importar lo que pase con las otras alertas, si se activa la de "T" se debe activar la lámpara, por el tercer enunciado sabemos que si dos alertas se encienden, entonces se debe activar la lampara, por el cuarto enunciado sabemos que si se enciende solo una alerta (que no sea "T") la lampara no se debe activar, por el primer y quinto enunciado tenemos algo evidente pero no por eso menos importante y es el que si tenemos todas nuestras alertas apagadas no debe encenderse la lámpara y el que si tenemos todas las alarmas prendidas se debe prender la lámpara.

Estas son principalmente nuestras condiciones con las cuales procedemos a realizar una Función Booleana que la satisfaga, empezaremos por nuestra condición de que se debe prender si tenemos a "A" y "N" prendidas se debe prender la lampara, esto es $A \cdot N$ ya que si una de las dos (o las dos) se encuentra apagada dará como salida 0 (no se prenderá la lámpara) pero si ambas están prendidas entonces la lámpara se va a prender, ahora, debemos recordar nuestra condición de que sin importar el estado de los demás medidores, si "T" esta prendido, se debe prender la lampara, por lo que nuestra expresión quedaría como:

$$F_1(A, N, T) = (A \cdot N) + T$$

A	N	T	$(A \cdot N)$	F_1
0	0	0	0	0
0	0	1	0	1
0	1	0	0	0
0	1	1	0	1
1	0	0	0	0
1	0	1	0	1
1	1	0	1	1
1	1	1	1	1

Notemos que con esto ya cumplimos los enunciados, segundo, tercero, cuarto y también el primero/quinto ya que si no tenemos ninguna alerta encendida tendríamos $(0 \cdot 0) + 0 = 0$ y si tuviéramos todas prendidas entonces tendríamos $(1 \cdot 1) + 1 = 1$
Por lo tanto su circuito seria:



Ya que tenemos la primera función que determina el estado de la lámpara de emergencias, procedemos a realizar la función que determina el encendido o apagado del motor por lo que procedemos a analizar las condiciones.

Comenzamos por los enunciados primero y quinto, los cuales nos dicen que si las tres entradas de alerta están activas no debemos de permitir el encendido del motor y por ende, si las tres están desactivadas el motor debe de prenderse; por el segundo enunciado sabemos que si "T" esta activada no debemos de permitir el encendido del motor.

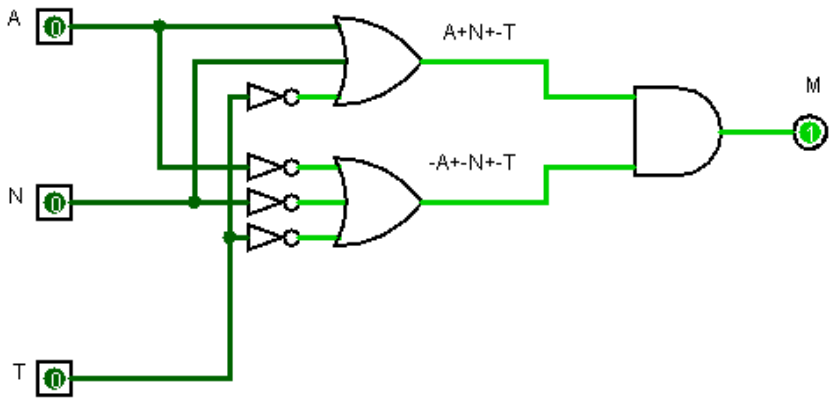
Con esto solo nos falta analizar el enunciado tercero y cuarto, el cuarto nos dice que si se enciende una sola alerta (que no incluyan a "T") el motor debe encenderse, el tercer enunciado en cambio nos dice que aunque se enciendan dos alertas el motor también debe encenderse aunque este encendida "T" provocando que seguramente termine en el mecánico el carro en cuestión, esto sumado a la condición del quinto enunciado nos deja la siguiente tabla de verdad que determina su comportamiento:

A	N	T	F_2
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

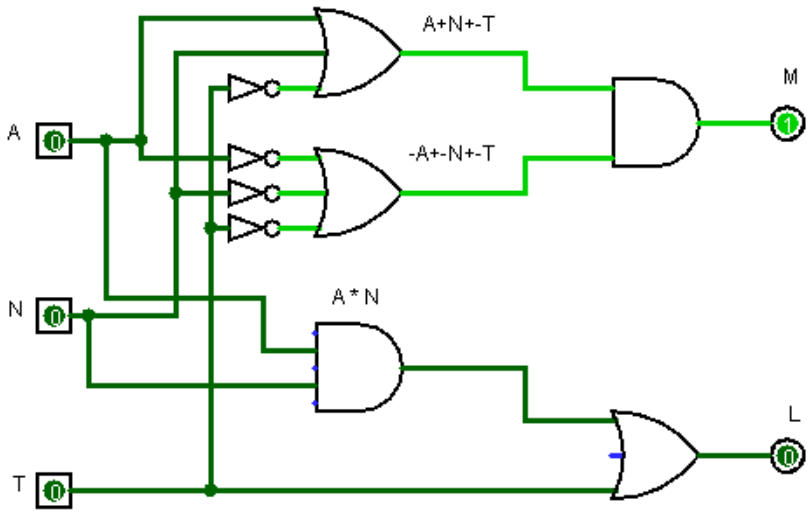
Usando Maxtérminos tendremos que la función que del funcionamiento del motor será:

$$F_2 = (A + N + \bar{T})(\bar{A} + \bar{N} + \bar{T})$$

Por lo que su circuito sería:



Por lo tanto el circuito de control que modela todo lo descrito anteriormente será:



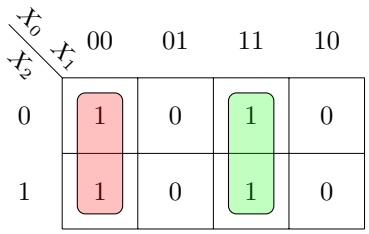
5 **Dibuje los diagramas lógicos de las siguientes funciones. En caso de de que la función no sea óptima, utilice el método a conveniencia visto en la tarea anterior para reducirla.**

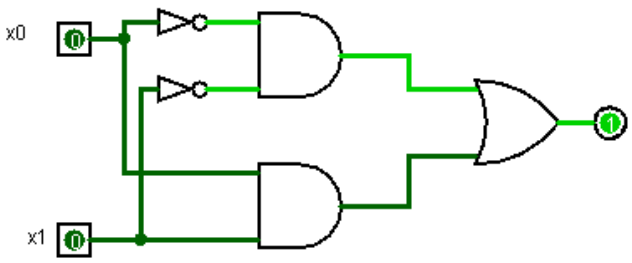
➢ $F(x_0, x_1, x_2) = x_0x_1x_2 + \bar{x_0}\bar{x_1}\bar{x_2} + x_0x_1\bar{x_2} + \bar{x_0}\bar{x_1}x_2$

Procedemos a realizar la tabla de verdad.

x_0	x_1	x_2	$\bar{x_0}$	$\bar{x_1}$	$\bar{x_2}$	$x_0x_1x_2$	$\bar{x_0}\bar{x_1}\bar{x_2}$	$x_0x_1\bar{x_2}$	$\bar{x_0}\bar{x_1}x_2$	F
0	0	0	1	1	1	0	1	0	0	1
0	0	1	1	1	0	0	0	0	1	1
0	1	0	1	0	1	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0	0
1	0	0	0	1	1	0	0	0	0	0
1	0	1	0	1	0	0	0	0	0	0
1	1	0	0	0	1	0	0	1	0	1
1	1	1	0	0	0	1	0	0	0	1

Procedemos a hacer el mapa de Karnaugh:





$\therefore F = \bar{x_0}\bar{x_1} + x_0x_1$

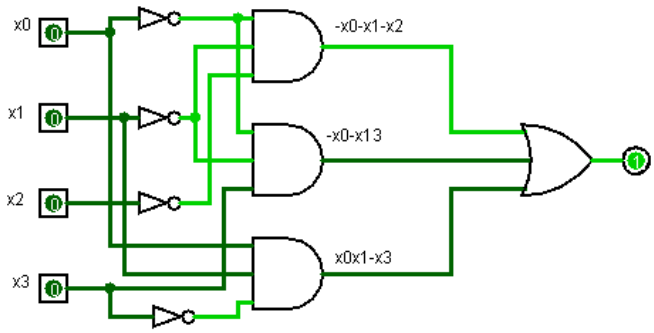
➢ $F(x_0, x_1, x_2, x_3) = \bar{x_0}\bar{x_1}\bar{x_2}\bar{x_3} + \bar{x_0}\bar{x_1}\bar{x_2}x_3 + \bar{x_0}\bar{x_1}x_2x_3 + x_0x_1\bar{x_2}\bar{x_3} + x_0x_1x_2\bar{x_3}$

Procedemos a realizar la tabla de verdad.

x_0	x_1	x_2	x_3	$\bar{x_0}$	$\bar{x_1}$	$\bar{x_2}$	$\bar{x_3}$	$\bar{x_0}\bar{x_1}\bar{x_2}\bar{x_3}$	$\bar{x_0}\bar{x_1}\bar{x_2}x_3$	$\bar{x_0}\bar{x_1}x_2x_3$	$x_0x_1\bar{x_2}\bar{x_3}$	$x_0x_1x_2\bar{x_3}$	F
0	0	0	0	1	1	1	1	1	0	0	0	0	1
0	0	0	1	1	1	1	0	0	1	0	0	0	1
0	0	1	0	1	1	0	1	0	0	0	0	0	0
0	0	1	1	1	1	0	0	0	0	1	0	0	1
0	1	0	0	1	0	1	1	0	0	0	0	0	0
0	1	0	1	1	0	1	0	0	0	0	0	0	0
0	1	1	0	1	0	0	1	0	0	0	0	0	0
0	1	1	1	1	0	0	0	0	0	0	0	0	0
1	0	0	0	0	1	1	1	0	0	0	0	0	0
1	0	0	1	0	1	1	0	0	0	0	0	0	0
1	0	1	0	0	1	0	1	0	0	0	0	0	0
1	0	1	1	0	1	0	0	0	0	0	0	0	0
1	1	0	0	0	0	1	1	0	0	0	1	0	1
1	1	0	1	0	0	1	0	0	0	0	0	0	0
1	1	1	0	0	0	0	1	0	0	0	0	1	1
1	1	1	1	0	0	0	0	0	0	0	0	0	0

Procedemos a hacer el mapa de Karnaugh:

x_0x_1 \ x_2x_3	00	01	11	10
00	1	1	1	0
01	0	0	0	0
11	1	0	0	1
10	0	0	0	0

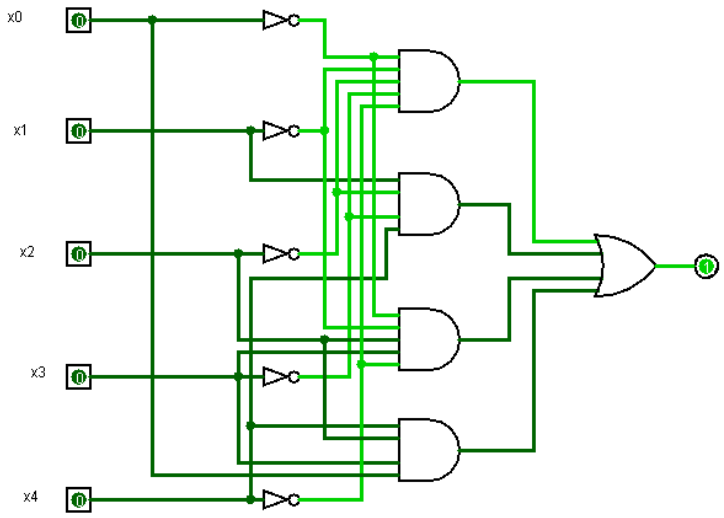


$\therefore F = \bar{x_0}\bar{x_1}\bar{x_2} + \bar{x_0}\bar{x_1}x_3 + x_0x_1\bar{x_3}$

➢ $F(x_0, x_1, x_2) = x_0x_1x_2 + \bar{x_0}\bar{x_1}\bar{x_2} + x_0x_1\bar{x_2} + \bar{x_0}\bar{x_1}x_2$
Como tenemos 5 variables y una suma de productos, realizaremos el mapa de karnaugh directamente para no hacer una tabla de verdad como la del anterior inciso, aprovechando las propiedades sobre la suma de productos.

x_0x_1 \ $x_2x_3x_4$	000	001	011	010	110	111	101	100
00	1	0	0	1	1	0	0	0
01	0	1	0	0	0	0	0	0
11	0	1	0	0	0	1	0	0
10	0	0	0	0	0	1	0	0

$\therefore F = \bar{x_0}\bar{x_1}\bar{x_2}\bar{x_3}\bar{x_4} + x_1\bar{x_2}\bar{x_3}x_4 + \bar{x_0}\bar{x_1}x_2x_3\bar{x_4} + x_0x_2x_3x_4$



6 **Dibuje el diagrama lógico de un sumador completo de 4 bits.**

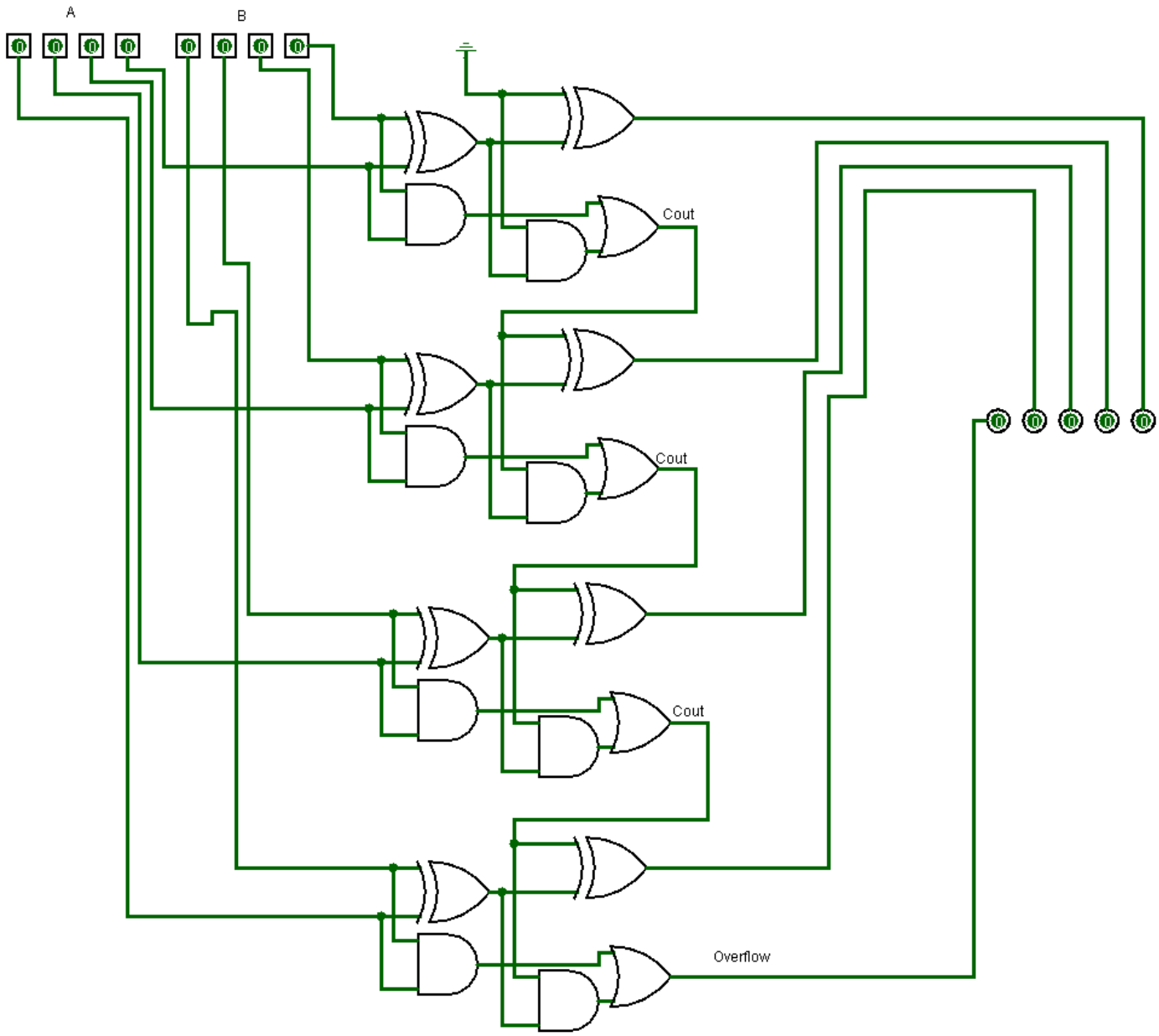
Para hacer un sumador completo de 4 bits , conectaremos en cascada 4 sumadores, con los cuales tenemos dos opciones: conectar 4 sumadores completos y conectar un ground en el acarreo de entrada del primer sumador o conectar 1 half-adder como primer sumador seguido de 3 sumadores completos.

En los dos casos anteriores, el primer sumador hará la suma de los bits menos significativos de cada entrada y generará un acarreo de salida que pasará como acarreo de entrada del siguiente sumador y de esta manera se conectan todos los sumadores hasta llegar al cuarto sumador que hará la suma de los bits más significativos junto con el acarreo de salida del sumador anterior y generará el resultado de la suma y un acarreo de salida que será el overflow.

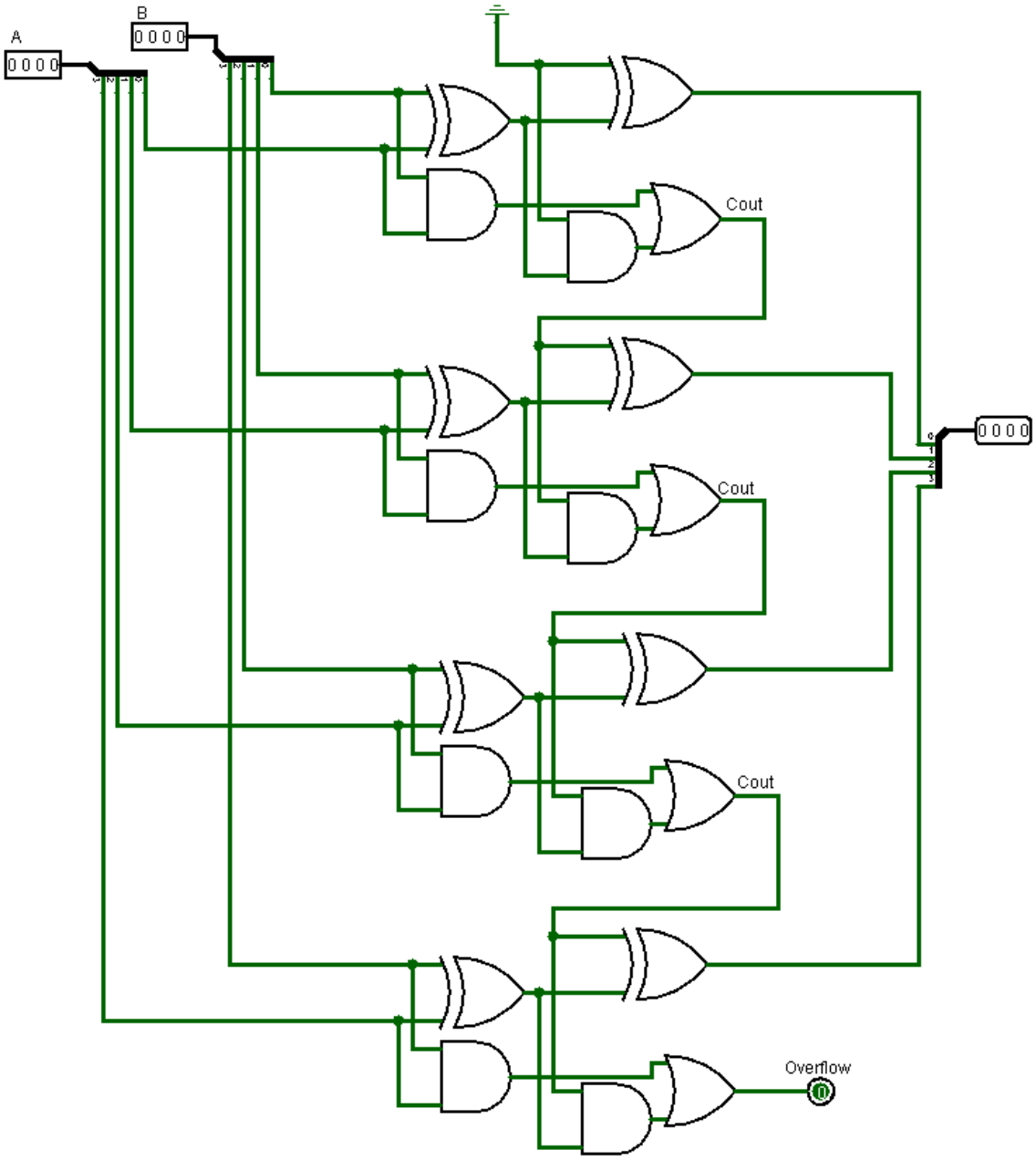
Los diagramas lógicos quedan de la siguiente manera:

Con 4 sumadores completos:

Circuito sin separadores

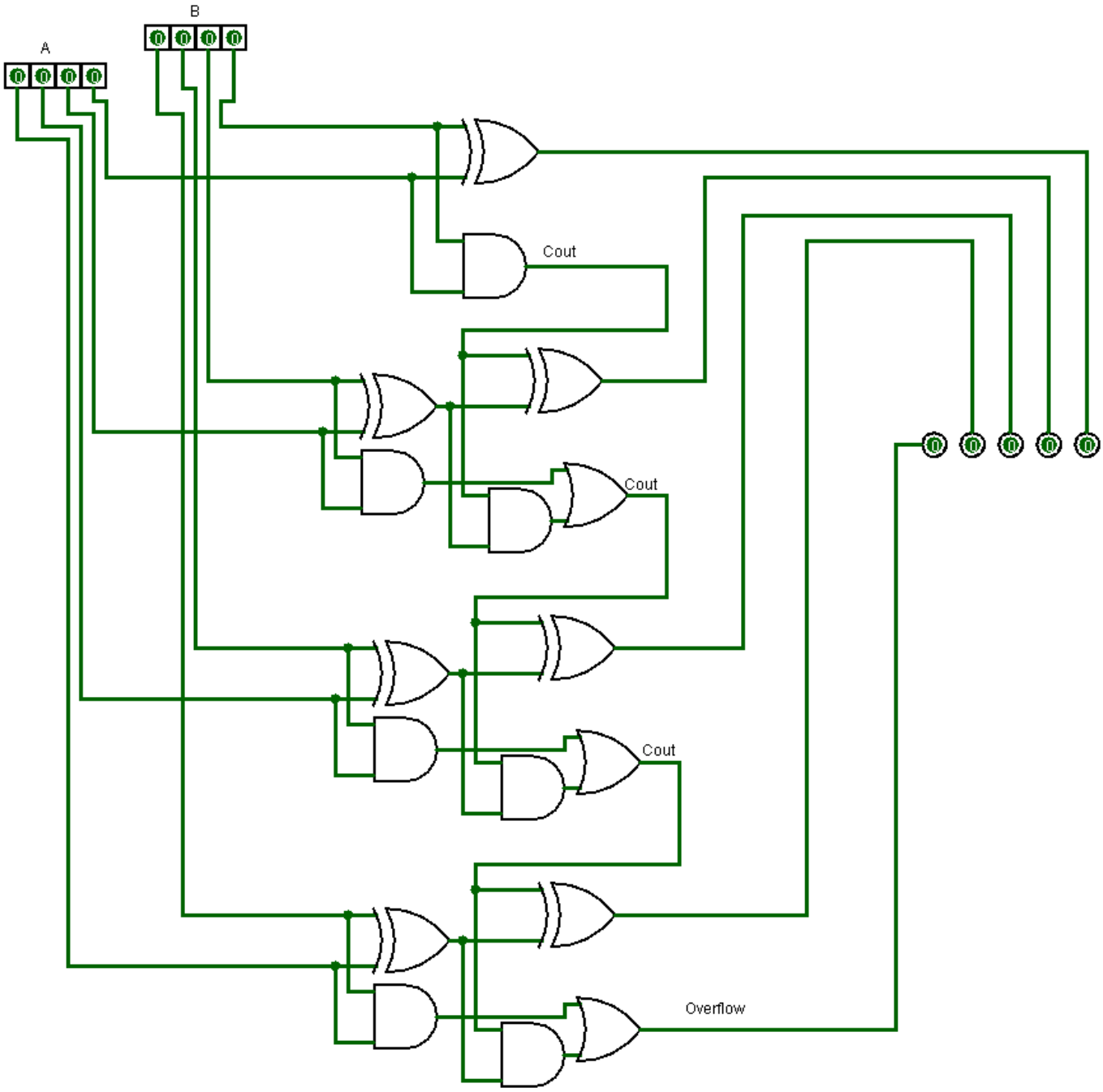


Circuito con separadores



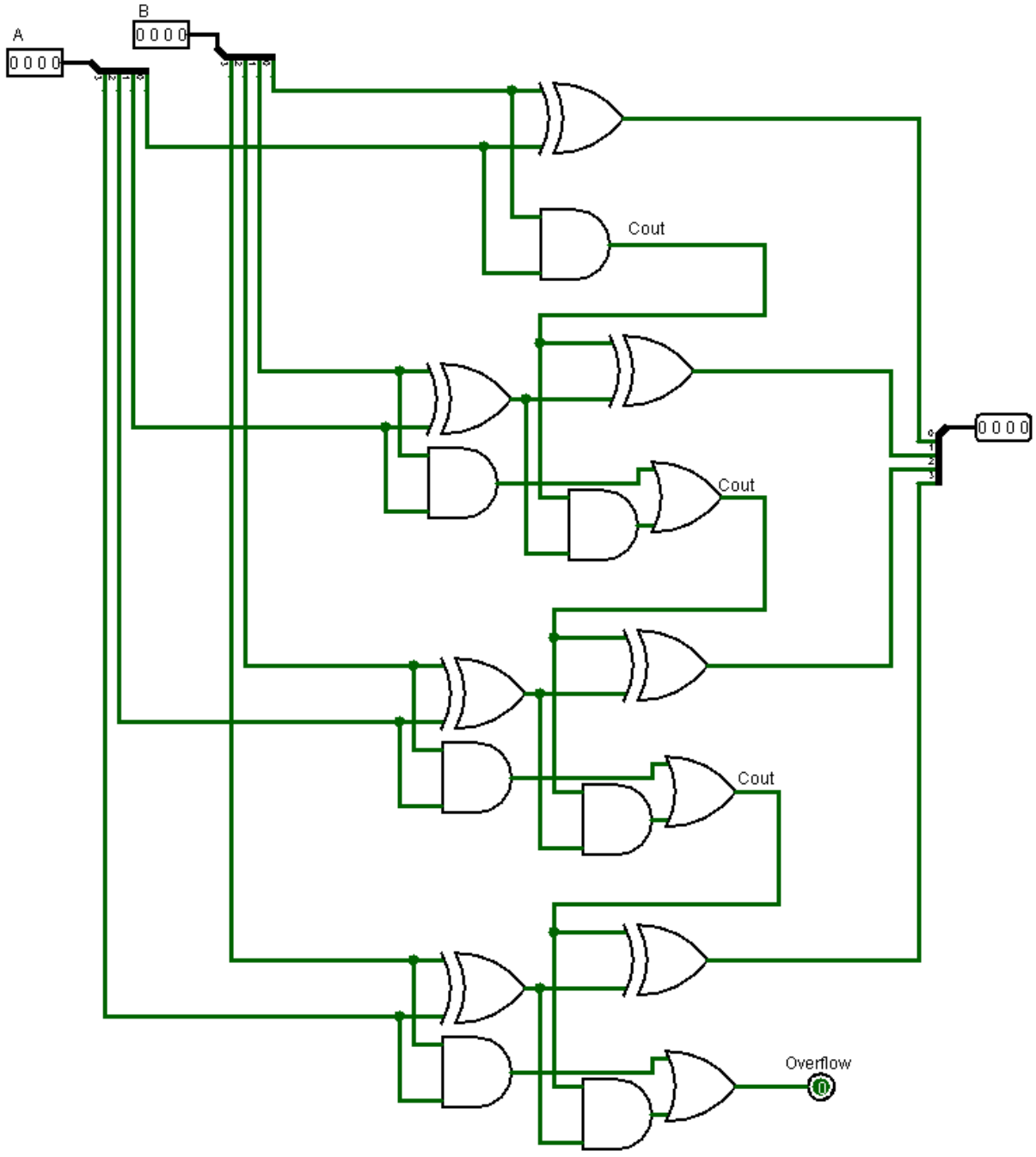
Con 1 half-adder y 3 sumadores completos:

Circuito sin separadores



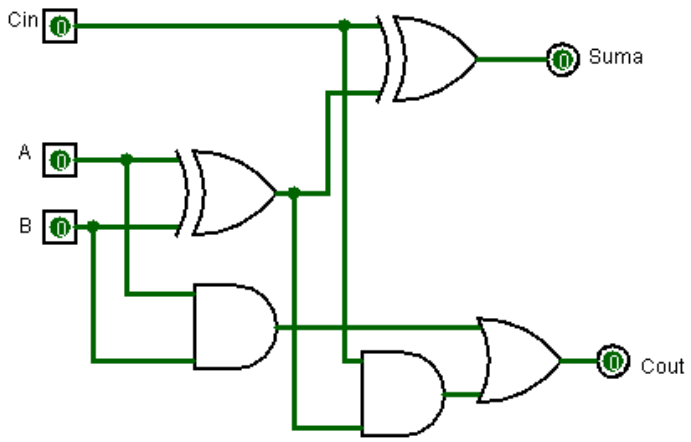
Con 1 half-adder y 3 sumadores completos:

Circuito con separadores



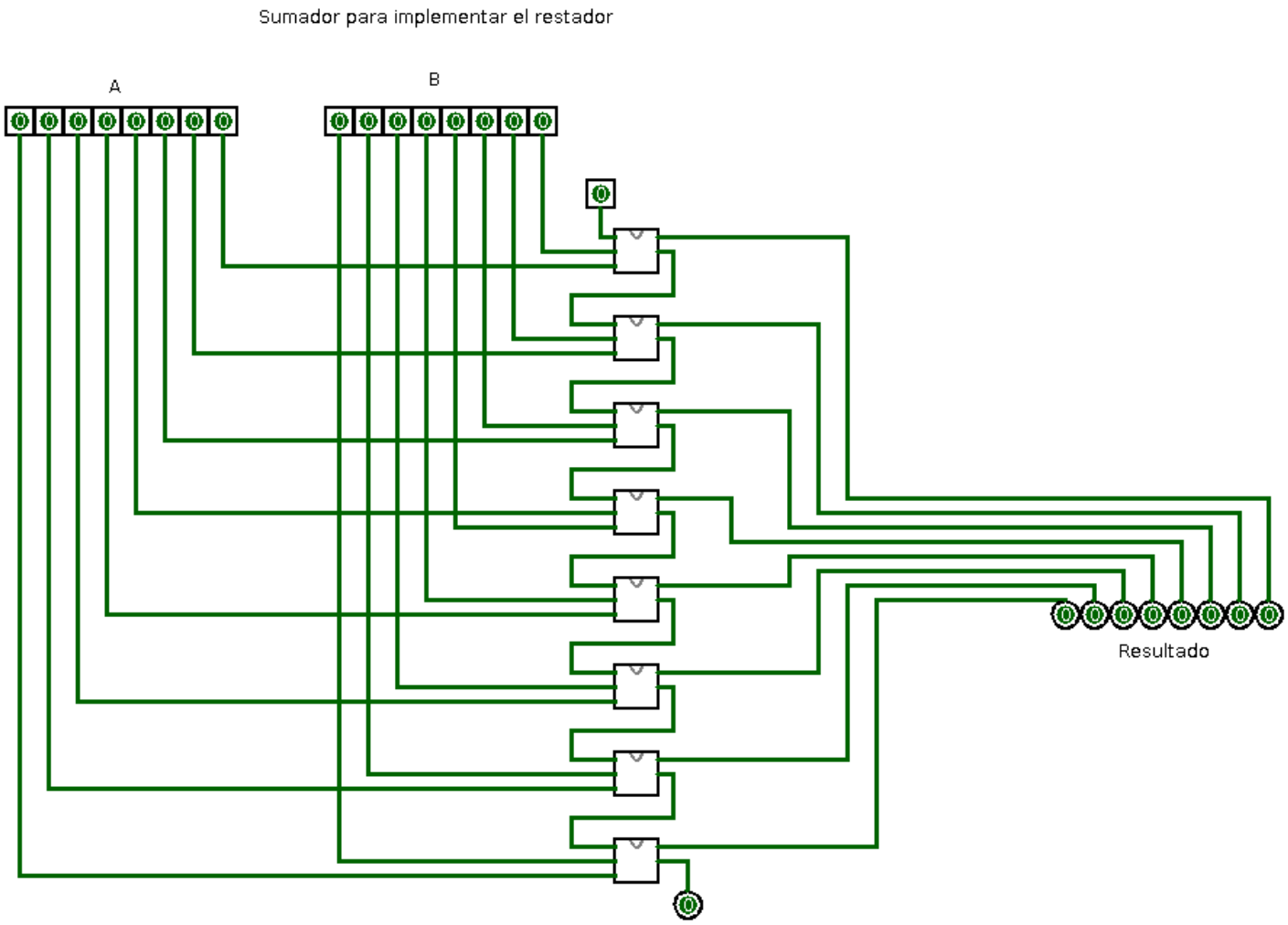
7 **Utilizando sumadores y semi-sumadores, diseña un circuito digital que realice resta entre numero de 8 bits.**

Para hacer un restador de 8 bits utilizaremos un sumador completo de 8 bits compuesto por 8 sumadores completos de 1 bit conectados en cascada.
El sumador completo de 1 bit que usaremos es el siguiente:

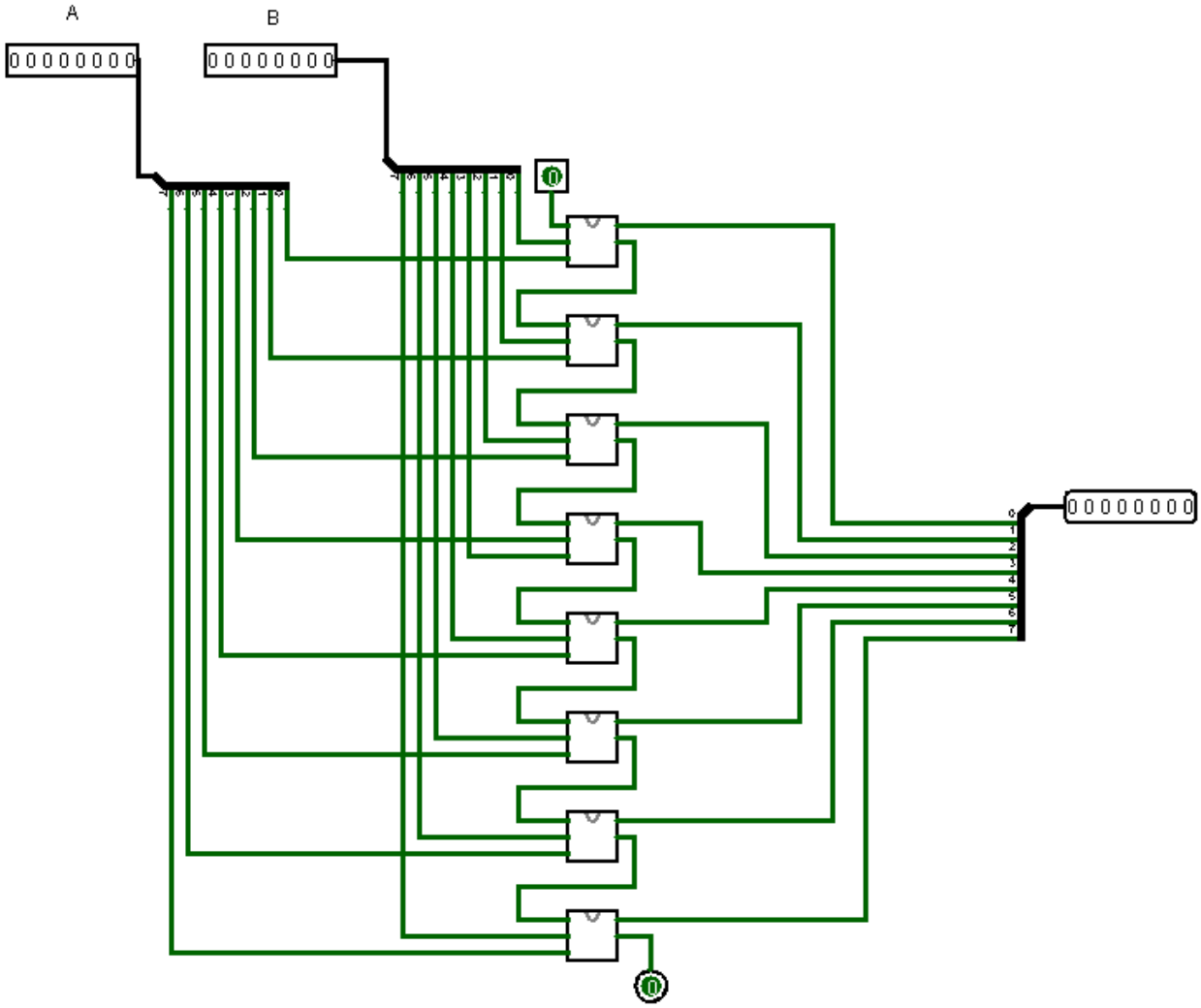


El sumador de 8 bits que usaremos será el siguiente.

Circuito sin separadores

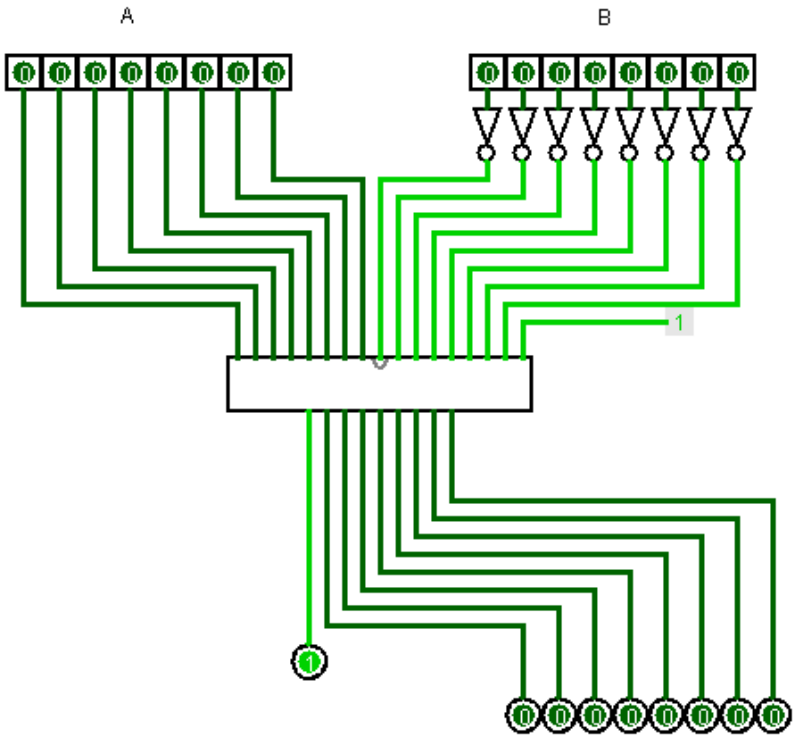


Circuito con separadores

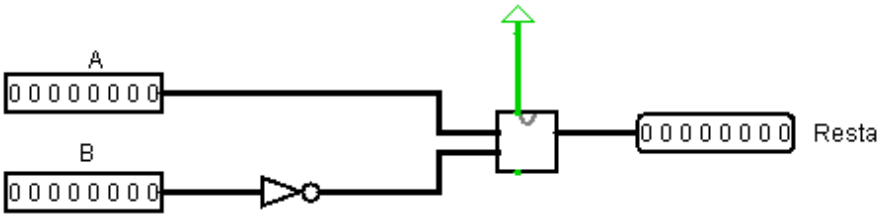


Para poder hacer la resta realizaremos el complemento a dos del sustraendo, el cual se sumará con el minuendo. Para hacer esto, sólo necesitamos negar todos los bits del sustraendo y al acarreo de entrada del sumador de 8 bits le conectamos un power de la siguiente manera:

Circuito sin separadores



Circuito con separadores



8 **Utilizando sumadores y semi-sumadores, diseña un circuito digital que realice multiplicación entre numeros de 4 bits.**

Para hacer una multiplicación de 4 bits lo hacemos de forma similar a como lo hacemos en decimal, es decir:

$$
\begin{array}{cccccccc}
 & & & & & & A_3A_2A_1A_0 & \\
 & & & & & \times & B_3B_2B_1B_0 & \\
\hline
 & & & & A_3B_0 & A_2B_0 & A_1B_0 & A_0B_0 \\
 & & & A_3B_1 & A_2B_1 & A_1B_1 & A_0B_1 & \\
 & + & A_3B_2 & A_2B_2 & A_1B_2 & A_0B_2 & & \\
 & A_3B_3 & A_2B_3 & A_1B_3 & A_0B_3 & & & \\
\hline
M_7 & M_6 & M_5 & M_4 & M_3 & M_2 & M_1 & M_0
\end{array}
$$

Por lo cual primero debemos hacer multiplicaciones entre bits y y luego sumar el resultado de esas multiplicaciones de forma adecuada.

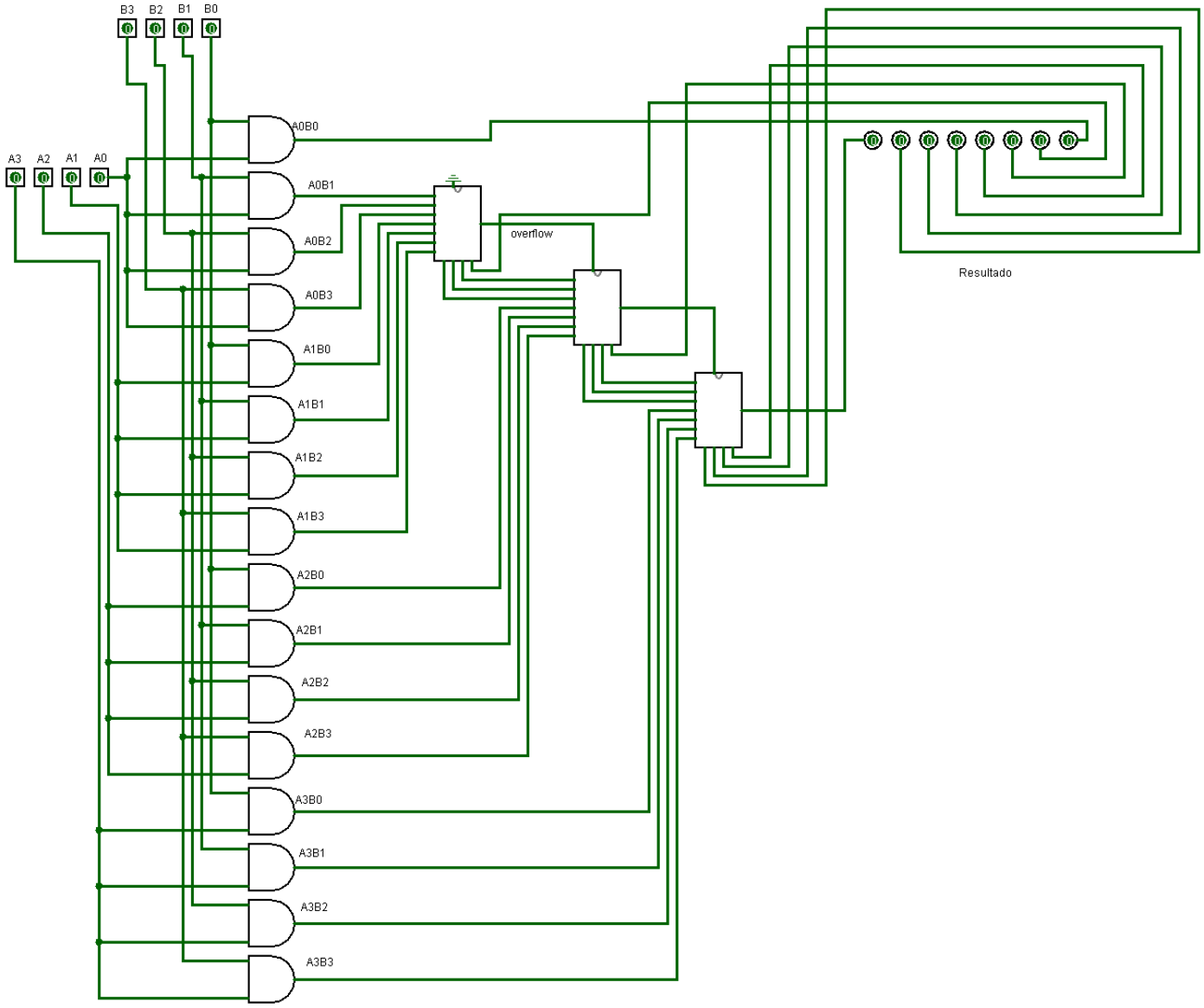
Notemos que la tabla de verdad de la multiplicación es:

p	q	p	*	q
1	1	1	1	1
1	0	1	0	0
0	1	0	0	1
0	0	0	0	0

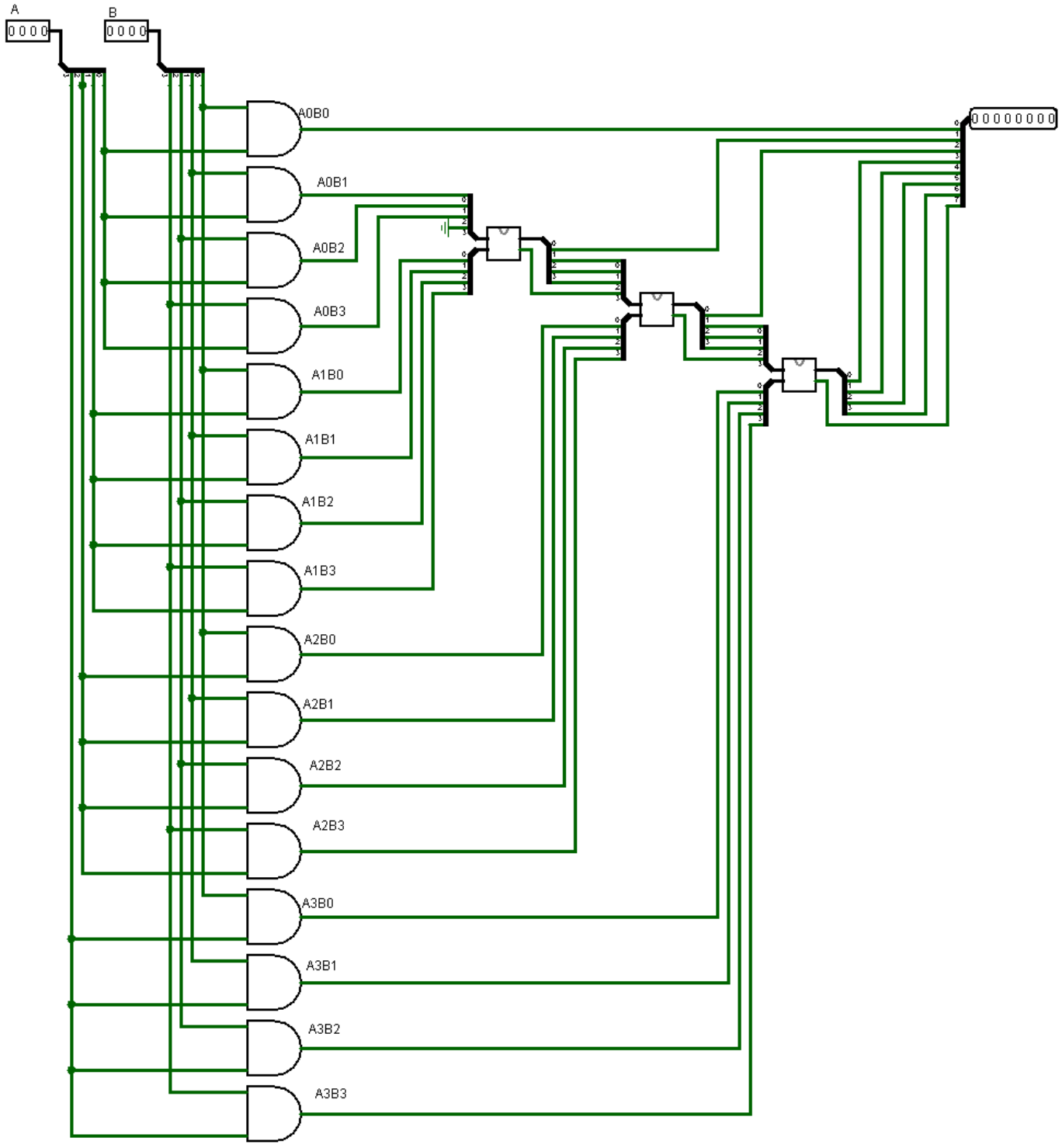
Podemos ver que ésta coincide con la salida de una compuerta AND.
Por lo tanto haremos las multiplicaciones usando compuertas AND (necesitaremos $4^2 = 16$ compuertas AND para realizar todas las multiplicaciones necesarias). Una vez que ya tengamos las multiplicaciones, notemos que debemos hacer 3 sumas de 4 bits , por lo cual usaremos 3 sumadores de 4 bits (los realizados en el ejercicio 6).

El bit A_0B_0 será el bit menos significativo del resultado, por lo que el primer sumador tendrá como entradas $A = 0\ A_3B_0\ A_2B_0\ A_1B_0$, $B = A_3B_1\ A_2B_1\ A_1B_1\ A_0B_1$ y de esta suma se obtendrá M_1. Los demás bits (junto con el acarreo)se pasarán a la entrada A del siguiente sumador , cuya entrada $B = A_3B_2\ A_2B_2\ A_1B_2\ A_0B_2$, del cual se obtiene M_2 y los demás bits (junto con el acarreo) se pasan a la entrada A del tercer sumador, que tiene como entrada $B = A_3B_3\ A_2B_3\ A_1B_3\ A_0B_3$ y nos da como resultado los 5 bits restantes del resultado. El circuito lógico es el siguiente:

Circuito sin separadores



Circuito con separadores

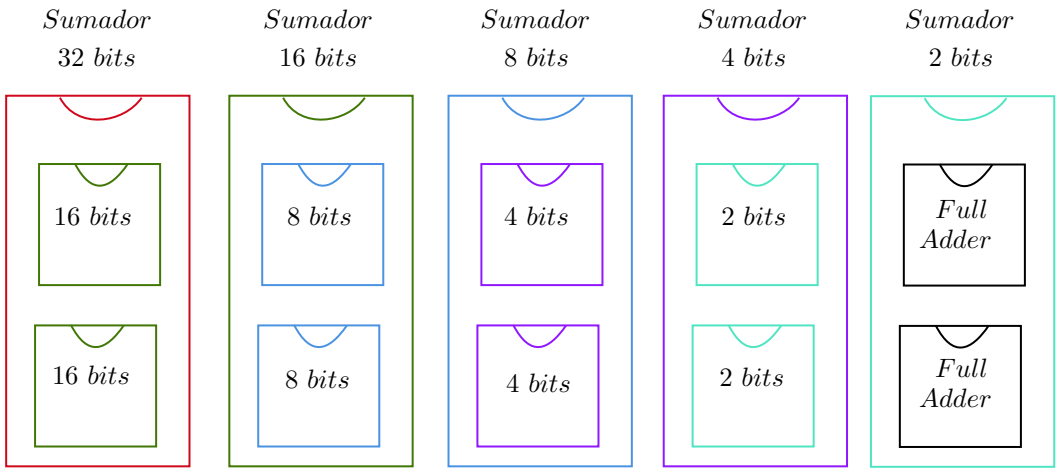


9 **Minimizando la cantidad de sumadores completos ¿Cómo quedaría la suma entre dos números de 32 bits? ¿Cuántos sumadores completos se requerirían?**

Caso de sumadores conectados en cascada:

Primero debemos recordar unos cuantos conceptos importantes: el primero es que la base para sumar una cantidad n de bits son los full adders (generalmente, aunque podemos usar también half-adders), los cuales solo están diseñados para operar con dos bits, además, debemos hacer énfasis en que nuestras compuertas lógicas en su nivel mas mínimo solo están hechas para soportar operaciones entre bits uno a uno para tener solo una salida de un bit, por lo que como hemos podido notar a lo largo de este trabajo, para una cantidad n de bits, necesitamos de una cantidad de n de full adders (o como vimos en el ejercicio 6, podemos usar un half-adder al inicio y 31 full-adders) ya que si por ejemplo estamos sumando un número A de 8 bits con otro número B de 8 bits, entonces tendremos que nuestro bit menos significativo de nuestro numero A se va a operar con el bit menos significativo de nuestro numero B y así sucesivamente hasta llegar al bit más significativo de ambos números, sin embargo como debe haber algo que opere a cada uno de esos 8 bits y sus acarreos, tendremos que ese algo serán 8 full adders (o 1 half-adder y 7 full-adders).

Ahora, si nosotros quisiéramos minimizar la cantidad de sumadores completos nos encontraríamos con que si o si necesitamos de esta dualidad, numero de bits - numero de full adders, ya que si le quitamos full adders a esta operación de 32 bits, tendríamos que hay bits que no se lograran operar respectivamente con sus pares en la otra entrada, sin embargo aunque no podemos "deshacernos"de los full adders, para tener un circuito más legible o no tan grande, lo que si podemos hacer es minimizar el circuito haciendo uso de equivalencias para que en vez de sumar 32 bits con 32 sumadores completos (o con 31 full-adders y un half-adder) lo hagamos con dos sumadores de 16 bits que a su vez estén compuestos ambos de sumadores de 8 bits y estos a su vez compuestos de sumadores de 4 bits y estos a su vez estén compuestos por sumadores de 2 bits y a su vez estos esten compuestos de dos full adders, los 32 (o 31) full adders siguen presentes pero con esto tendremos un circuito más minimizado.



Representación de un sumador de 32 bits usando full adders y sumadores de menor capacidad

Por lo tanto, el número mínimo de sumadores completos que tenemos para este caso son 31. Sin embargo, recordemos que podemos hacer un sumador completo utilizando dos half-adders y una compuerta or, por lo cual podríamos sustituir todos los full-adders por sumadores completos hechos con half-adders. Esto haría un sumador de 32 bits usando $31 * 2 + 1 = 63$ half adders y sin sumadores completos explícitamente. No obstante esto no nos quita el hecho de que estamos simulando el comportamiento de sumadores completos.

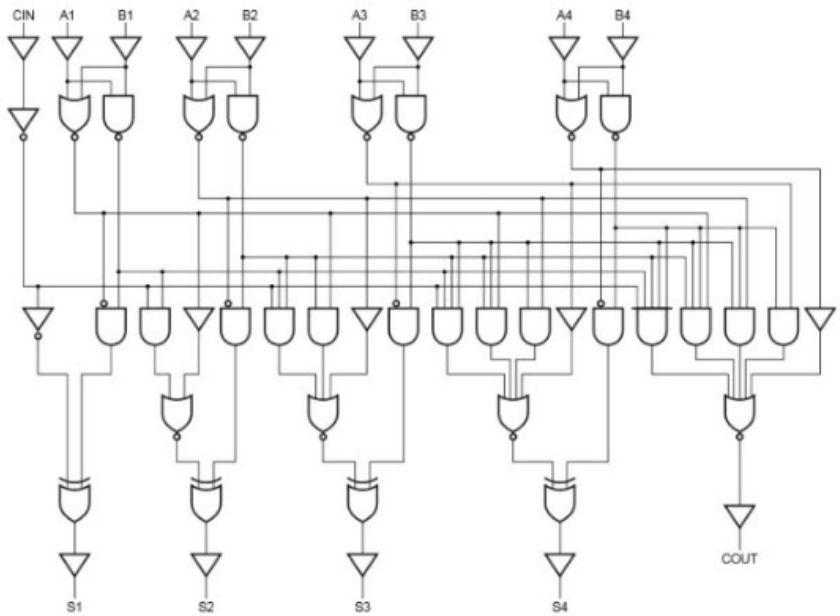
Caso de sumadores con acarreo anticipado:
Los sumadores con acarreo anticipado usa compuertas lógicas para ver si se generará o no un acarreo de orden superior.
Existen el acarreo de propagación (P_i) y el acarreo de generación (G_i). El bit P_i se propaga a la siguiente etapa y el bit G_i se usa para generar el bit de acarreo de salida, independientemente del bit de acarreo de entrada.

Notemos que teniendo la siguiente tabla de verdad:

A	B	C_i	C_{i+1}
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

Tenemos que $P_i = A \oplus B_i$ y $G_i = A_i B_i$, por lo que la suma y el acarreo son $S_i = P_i \oplus C_i$ y $C_{i+1} = G_i + P_i C_i$.
Aquí G_i es un acarreo que produce el acarreo cuando A_i y B_i con independientes del acarreo de entrada y P_i se asocia con la propagación de C_i y C_{i+1}.
Generalizando tenemos entonces que $C_{n+1} = G_n + \sum_{k=1}^{n}(G_{k-1}\Pi_{i=k}^{n} Pi) + C_0\Pi_{i=1}^{n} Pi$.
Con esto, nnotemos que podemos hacer los circuitos de los acarreos de salida conectando compuertas and seguidas de una compuerta or.

Primero veamos el caso de 4 bits usando el acarreo anticipado: en cada suma se usan dos compuertas xor. uno de ellos produce P_i y un and produce G_i. Notemos que aquí tenemos dos niveles de compuertas que producen P_i y G_i.

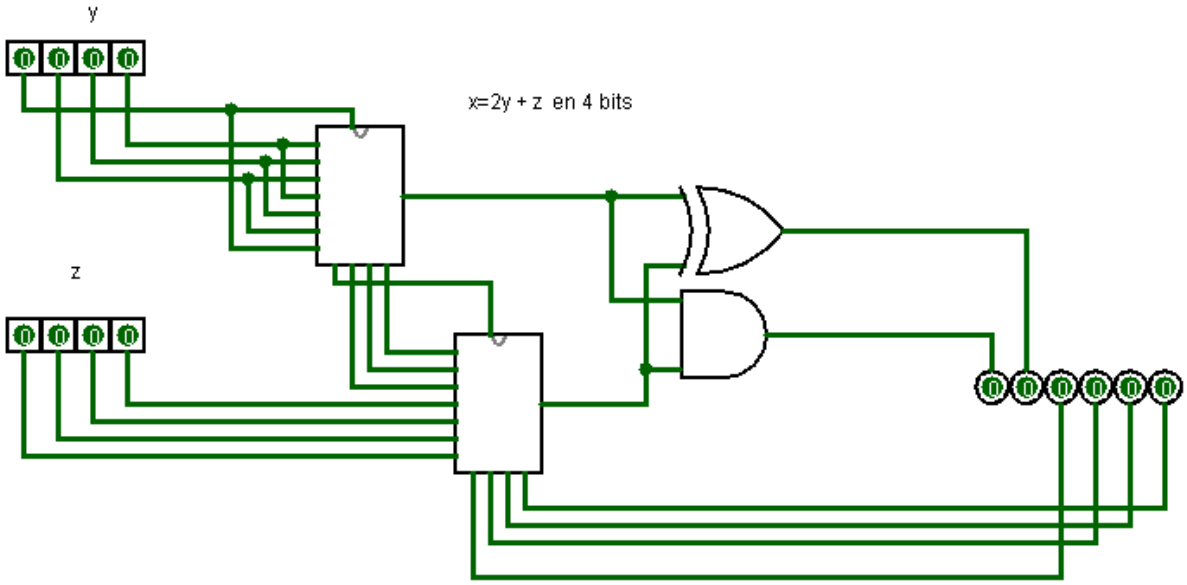


Por lo tanto, para construir un sumador de 32 bits con el esquema de acarreo anticipado, requerimos conectar en cascada dos sumadores carry look ahead de 16 bits, en los cuales se conectan en cascada 4 sumadores carry look ahead de 4 bits.

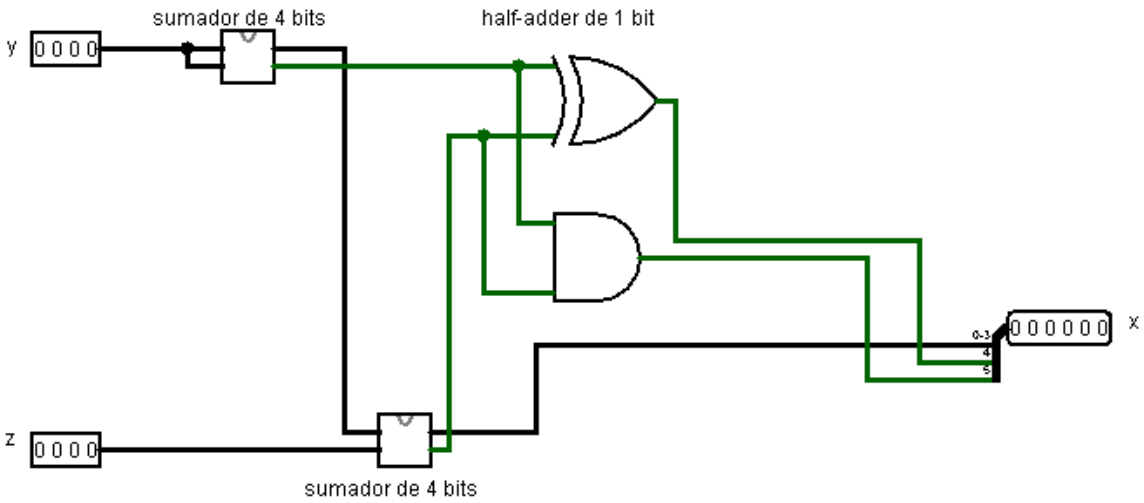
10 **Diseña un circuito digital que resuelva la siguiente ecuación:** $x = 2y + z$.

Haremos un circuito que opere con entradas de 4 bits.
Tendremos son entradas, que serán z y y y una salida x.
Al observar lo requerido nos damos cuenta que debemos realizar operaciones sencillas, en primera instancia sumaremos una entrada z con dos veces la entrada y. Notemos que se puede que resolver esto con circuitos sumatorios, por lo que si sumamos la entrada y consigo misma, obtendremos el resultado de $2y$ y si a eso le sumamos la entrada z ya tendríamos nuestro resultado (no debemos olvidar que también debemos sumar el acarreo de la operación $2y$ con el acarreo de la suma de $2y + z$) teniendo entonces que el resultado de todo lo anterior sería:

Circuito sin separadores



Circuito con separadores



REFERENCIAS

1. Quora. (n.d.). How do you construct a logic gate using a NAND gate only for the expression? https://www.quora.com/How-do-you-construct-a-logic-gate-using-a-NAND-gate-only-for-the-expression-x-A-B-C

2. Electronica, W. (2018, 12 septiembre). Multiplicador. Wilaeba Electronica. Recuperado 29 de abril de 2023, de https://wilaebaelectronica.blogspot.com/2017/09/multiplicador-de-dos-numeros-de-4-bits.html

PDF Tema 11: Sistemas combinacionales - Free Download PDF. (s. f.). https://silo.tips/download/tema-11-sistemas-combinacionales

3. Administrator. (2017). Carry-Lookahead Adder. ElectronicsHub. https://www.electronicshub.org/carry-look-ahead-adder/

4. WatElectronics. (2022, 14 julio). Carry Lookahead Adder: Truth Table, Circuit, Advantages and Applications. WatElectronics.com. https://www.watelectronics.com/carry-lookahead-adder/