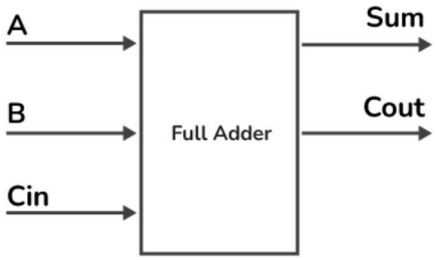
Propuestas Laboratorio 2: Lógica Combinacional y Aritmética

Estudiantes: Ana Melissa Vásquez Rojas, Dario Garro Moya

La calculadora parametrizable debe implementar las operaciones: suma, resta, multiplicación, división, módulo, and, or, xor, shift left y shift right. Para las operaciones de suma, resta, multiplicación y módulo se empleará un enfoque estructural, mientras que el resto se realizará mediante operadores de HDL.

Para ambas propuestas se utilizará el mismo diseño de sumador completo.

Suma: Se realizará un sumador completo de 1 bit:



La tabla se muestra a continuación:

C_{in}	**A**	**B**	**S**	C_{out}
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

La ecuación booleana se obtiene a partir de los mintérminos.

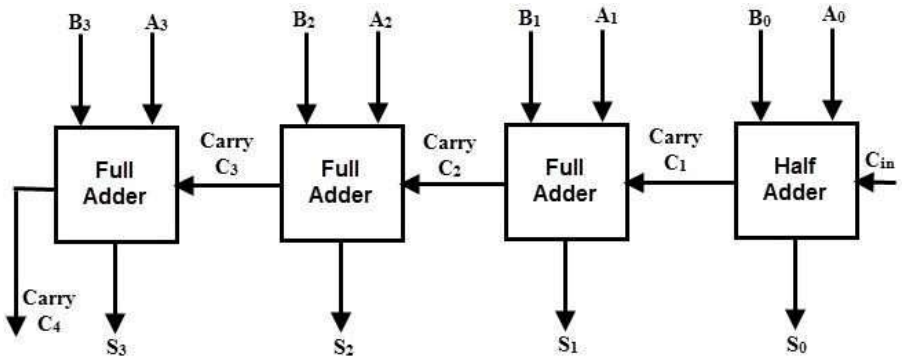
Primero para S (S = 1):

- $S = \overline{AB}C_{in} + \overline{A}B\overline{C_{in}} + A\overline{BC_{in}} + ABC_{in}$
- Ya que $\overline{A}B + A\overline{B}$ = $A \oplus B$, se simplifica como
-
- $S =_{in} \overline{(A \oplus B)} + \overline{C_{in}}(A \oplus B)$, siendo la expresión final
- $S = A \oplus B \oplus C_{in}$

Luego para C_{out} (C_{out} = 1):

- $C_{out} = \overline{A}BC_{in} + A\overline{B}C_{in} + AB\overline{C_{in}} + ABC_{in}$, esto se simplifica como
- $C_{out} = AB(\overline{C_{in}} + C_{in}) + C_{in}(\overline{A}B + A\overline{B})$, siendo la siguiente la expresión final
- $C_{out} = AB + C_{in}(A \oplus B)$

Para hacer el sumador de 4 bits o más, lo que se hace es colocarlo en cascada, como se muestra a continuación.



Propuesta 1 - Restador: El sumador completo permite operar tanto con números positivos como negativos mediante el uso de complemento a dos. Para implementar la resta, se invierte el signo del segundo operando aplicando su complemento a uno y, posteriormente, sumándole 1. De esta manera, la operación de resta puede resolverse utilizando el mismo bloque del sumador completo.

C_{in}	**A**	**B**	**S**	C_{out}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	0
0	1	1	0	0
1	0	0	1	1
1	0	1	0	1
1	1	0	0	0
1	1	1	1	1

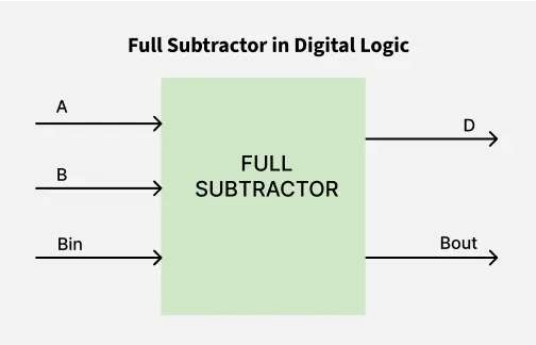
Para conseguir $Y = A - B$

- $Y = A + \overline{B} + 1 = A - B$

La salida (S) se mantiene igual, solo cambia el C_{out}:

- $C_{out} = \overline{AB}C_{in} + \overline{A}B\overline{C_{in}} + \overline{A}BC_{in} + ABC_{in}$, sería la ecuación aplicando mintérminos
- $C_{out} = \overline{A}B + \overline{A}C_{in} + BC_{in}$, siendo la expresión final simplificada
- $C_{out} = \overline{A}(B + C_{in}) + BC_{in}$

Propuesta 2 – Restador: Se realiza el restador completo desde 0.



Para este diseño borrow in se mantiene en 0 al inicio, y además no se requiere asignación de switch.

A continuación, se muestra la tabla del restador.

A	**B**	C_{in}	**S**	C_{out}
1	0	0	0	0
1	0	1	1	1
1	1	0	1	1
1	1	1	0	1
0	0	0	1	0
0	0	1	0	0
0	1	0	0	0
0	1	1	1	1

Se realiza el mapa de Karnaugh para la salida S

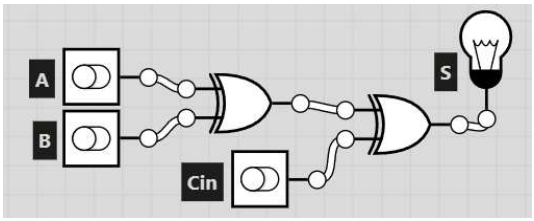
A / B C_{in}	00	01	11	10
0	0	1	0	1
1	1	0	1	0

La expresión sin simplificar es la siguiente:

- $S = \overline{AB}C_{in} + \bar{A}B\overline{C_{in}} + A\overline{BC_{in}} + ABC_{in}$
- $S = C_{in}(\overline{AB} + AB) + C_{in}(\bar{A}B + A\bar{B})$
- $S = C_{in}(\overline{A \oplus B}) + \overline{C_{in}}(A \oplus B)$

Por lo que se obtiene lo siguiente:

- $S = C_{in} \oplus (A \oplus B)$



Luego se extrae de la tabla de verdad los mintérminos para C_{out}.

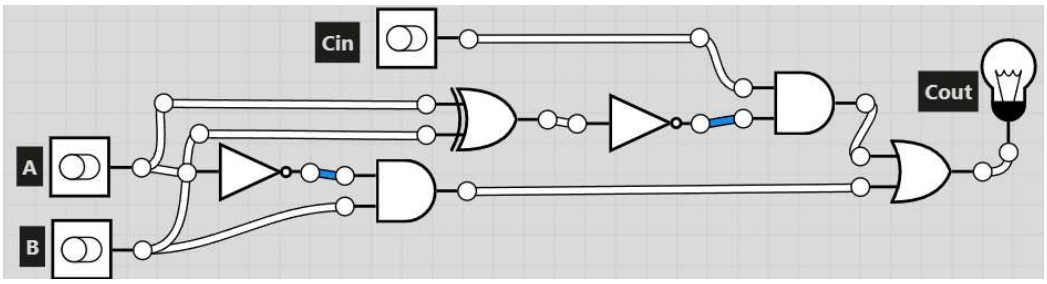
- $C_{out} = \overline{AB}C_{in} + \bar{A}B\overline{C_{in}} + \bar{A}BC_{in} + ABC_{in}$

  Factorizando los términos:

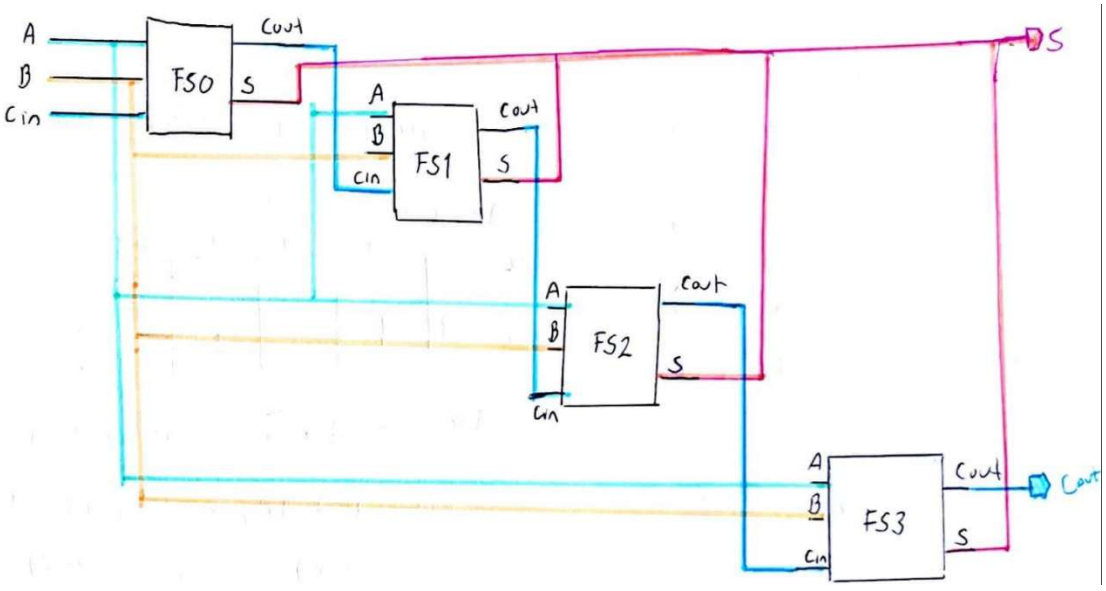
- $C_{out} = C_{in}(AB + \overline{AB}) + \bar{A}B(C_{in} + \overline{C_{in}})$

  Usando las propiedades de xor y xnor:

- $C_{out} = C_{in}(\overline{A \oplus B}) + \bar{A}B$



Y para hacerlo de 4 bits o más se coloca en cascada, como se muestra a continuación:



Propuesta 2 - Multiplicación: Se requiere implementar un multiplicador de 4×4 bits mediante un enfoque estructural. El resultado completo será de 8 bits. Para ello:

Se generan los productos parciales mediante compuertas AND.

Dichos productos parciales se suman por columnas utilizando sumadores completos (Full Adders, FA) y medios sumadores (Half Adders, HA).

La estructura general se representa de la siguiente manera:

				A_3	A_2	A_1	A_0
			x	B_3	B_2	B_1	B_0
				$A_3\ B_0$	$A_2\ B_0$	$A_1\ B_0$	$A_0\ B_0$
			$A_3\ B_1$	$A_2\ B_1$	$A_1\ B_1$	$A_0\ B_1$	
		$A_3\ B_2$	$A_2\ B_2$	$A_1\ B_2$	$A_0\ B_2$		
+	$A_3\ B_3$	$A_2\ B_3$	$A_1\ B_3$	$A_0\ B_3$			
P_7	P_6	P_5	P_4	P_3	P_2	P_1	P_0

Para realizar un multiplicador de N-bit Multiplicación, se necesitan N^2 compuertas AND, N medios sumadores, y N*(N-2) sumadores completos.

Donde para el sumador completo de 1 bit se utiliza el que se realizó anteriormente.

El diseño del medio sumador, necesario para la implementación, se basa en la siguiente tabla de verdad:

A	**B**	**Sum**	**Carry**
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

A continuación, se muestra el mapa de Karnaugh para Sum

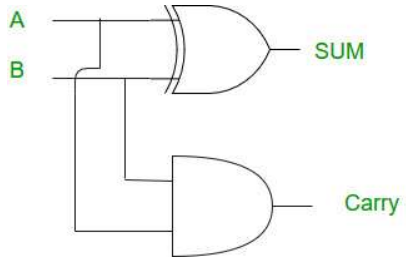
B\A	0	1
0	0	1
1	1	0

$$Sum = A \oplus B$$

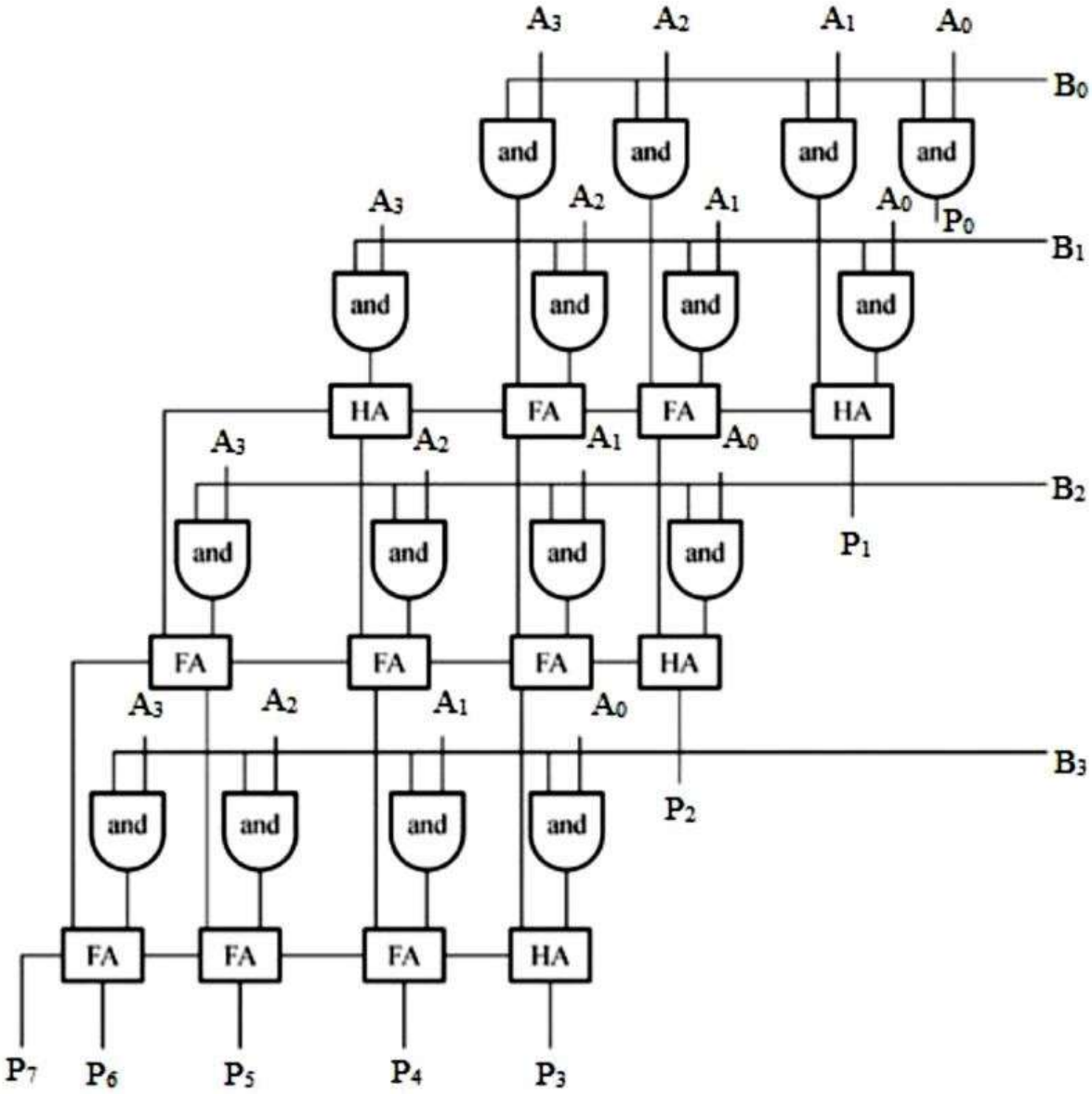
Luego, para el carry la expresión lógica sería:

$$Carry = AB$$

Por lo que el circuito lógico para el medio sumador es el siguiente:



Posteriormente, para implementar el multiplicador, se estructurará de la siguiente manera: los bloques identificados como FA corresponden a Full Adders y los bloques señalados como HA corresponden a Half Adders.



Propuesta 1 – División: La división sería equivalente a la siguiente ecuación:

$$N = Q \times D + R$$

- N = dividendo
- D = divisor
- Q = cociente
- R = resto

Todos siendo de 4 bits de tamaño. Se compara N con D, si N es mayor o igual a D se restan una vez y se actualiza N, se anota 1 en el cociente, si N es menor que D, el bit del cociente es 0. Este ciclo se repite varias veces siempre "bajando" o desplazando un bit del dividendo. Para 4 bits, se deben realizar 4 verificaciones.

Este ciclo se divide en cuatro etapas que debe seguir el número

1. Restador n-bit: Este calcula N - D
2. Comparador: Este indica si N es mayor o igual a B
3. Multiplexor: Escoge entre "restar" el número o "mantener" el valor.
4. Resultado: Este hace que los resultados intermedios "bajen".

La división tiene algunas reglas, la siguiente tabla las muestra

Regla	Resultado
0/0	Un cero dividido por cero será indefinido: ∞
0/1	Un cero dividido por un uno será cero: 0
1/0	Un uno dividido por un cero será indefinido: ∞
1/1	Un uno dividido por otro uno será uno: 1

Ya que utilizaremos de manera constante la resta entre binarios se necesita cumplir con la tabla de verdad:

A	**B**	**S**	**Carry**
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

En la Propuesta 1, se plantea un diseño donde la suma se implementa con un sumador completo estructural. La resta se obtiene a partir de este mismo sumador, aplicando el complemento a dos sobre el segundo operando. Además, esta propuesta incluye una división estructural, basada en un algoritmo de comparación y resta sucesiva para calcular el cociente y el residuo, lo cual añade un nivel considerable de complejidad al diseño.

La Propuesta 2 también utiliza el sumador completo para la operación de suma, pero propone un restador diseñado desde cero, con su propia tabla de verdad, mapas de Karnaugh y ecuaciones lógicas. Esto hace que el módulo sea independiente y más modular. A esto se le suma la multiplicación estructural de 4x4 bits, construida a partir de productos parciales con compuertas AND y su suma mediante Full Adders y Half Adders.

Después de comparar las dos alternativas, decidimos elegir la Propuesta 2, porque nos da más modularidad y permite que cada bloque sea más independiente. Aunque es un poco compleja de implementar, consideramos que es la opción que mejor se ajusta a los objetivos del laboratorio y que, además, facilita trabajar con un diseño más claro y escalable.

Además de los módulos seleccionados para ser implementados de manera estructural, el resto de las operaciones se realizarán empleando directamente los operadores de HDL:

- División: Se utiliza el operador /
- Módulo: Se utiliza el operador %
- AND: Se crea un módulo aparte utilizando &
- OR: Se crea un módulo aparte utilizando ||
- XOR: Se crea un módulo aparte utilizando ^
- Shift Right: Se utiliza el operador >>
- Shift Left: Se utiliza el operador <<

Las banderas de estado de la ALU se visualizarán en los primeros cuatro LEDs, en el siguiente orden: Negativo (N), Cero (Z), Acarreo (C) y Desbordamiento (V).

Finalmente, el resultado de cada operación se mostrará en el display de 7 segmentos.