

Arquitectura de Computadores

IIC1005 - Computación: Ciencia y Tecnología del Mundo Digital

Nicolás Elliott B. (nicolas.elliott@uc.cl)

Vicente Domínguez (vidomingez@uc.cl)

Luis Ramírez (llramirez@uc.cl)



DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
ESCUELA DE INGENIERÍA
PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

(II/2019)

Introducción

Buscamos responder

- ¿Qué es un computador?
- ¿Cómo funciona?
- ¿Cómo se construye?
- ¿Cómo se programa?
- ¿Cómo se mejora/optimiza su rendimiento?

¿Computador?



Computador

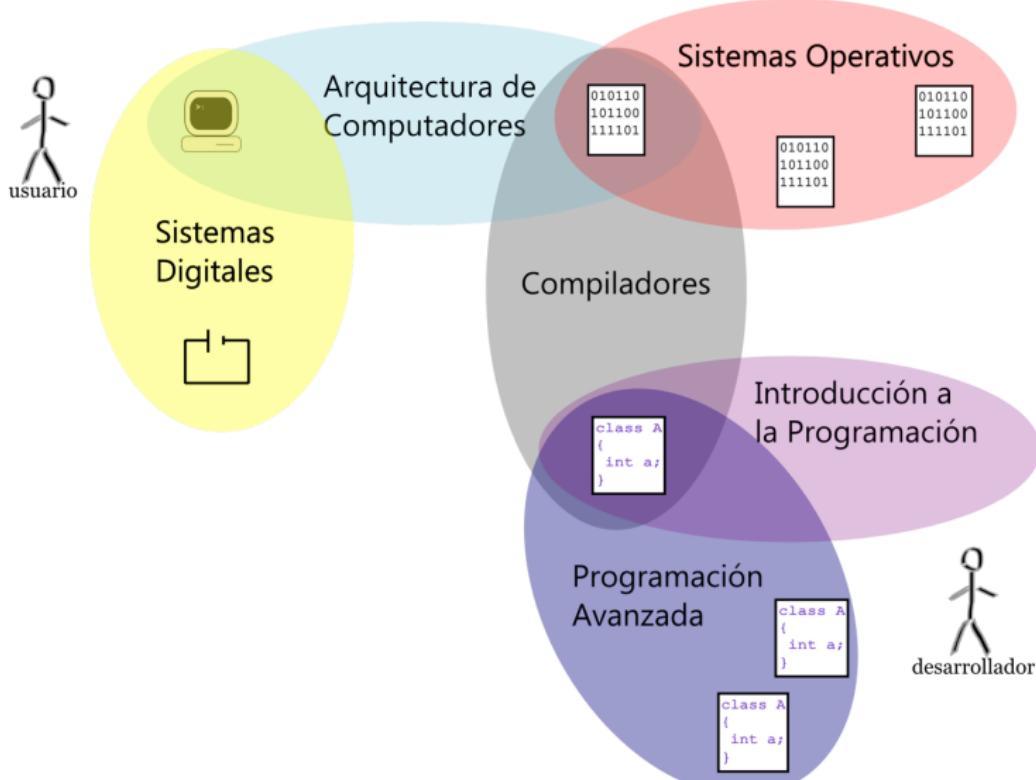
Definición:

- ① Máquina electrónica capaz de almacenar información y tratarla automáticamente mediante operaciones matemáticas y lógicas controladas por programas informáticos.
- ② A.k.a Máquina programable que ejecuta programas.

Programa:

- Secuencia de instrucciones.

¿Cómo?

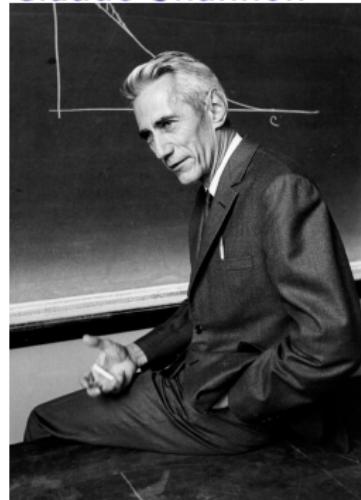


Historia

Claude Shannon

- 1916 - 2001
- Aplica el álgebra de Boole al análisis y la síntesis de la conmutación y de los circuitos digitales

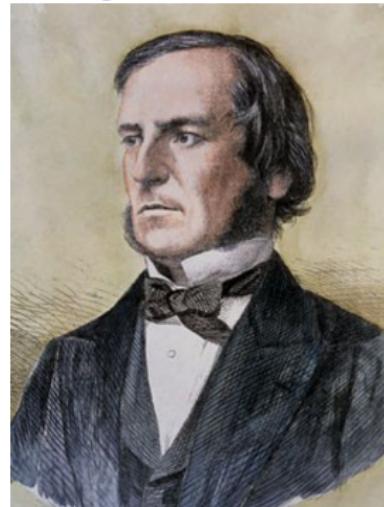
Claude Shannon



George Boole

- 1815 - 1864
- Estudio matemática y la lógica
- Álgebra de Boole, usa solo variables del tipo V o F

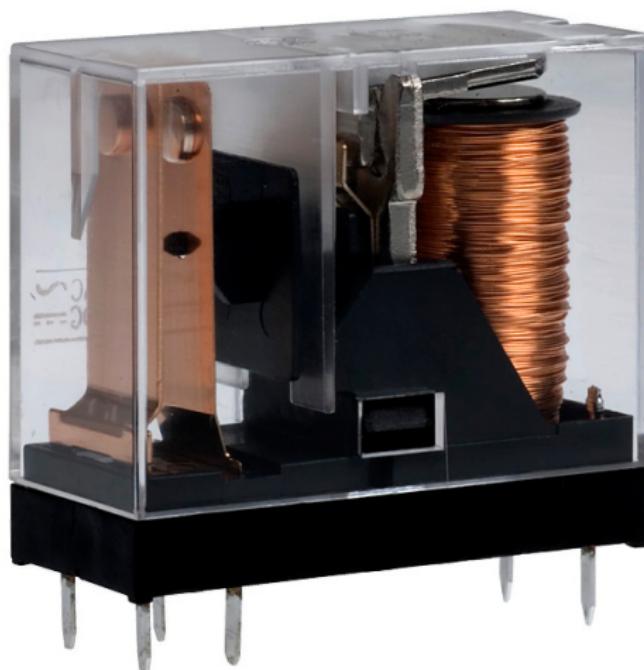
George Boole



Implementación

Generación 0 (1830)

Relés



Generación 1 (1900)

Tubos



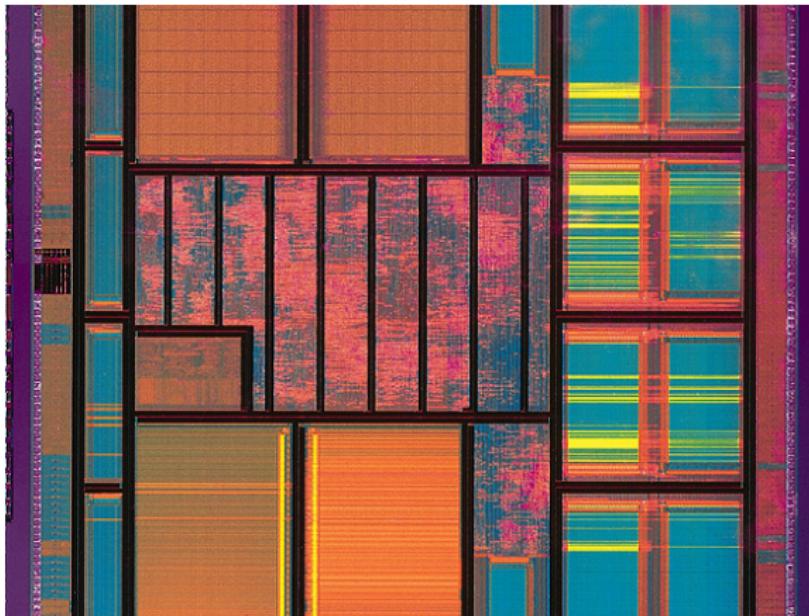
Generación 2 (1947)

Transistores

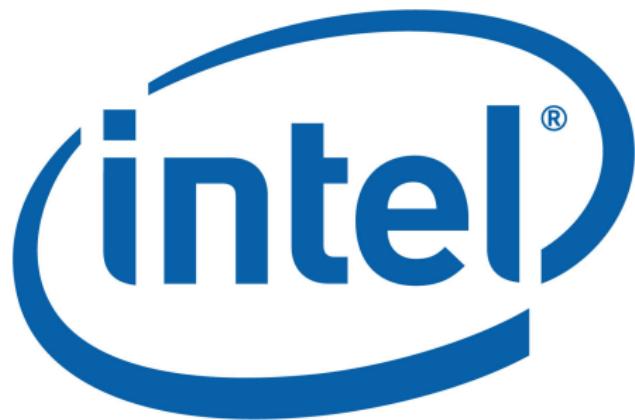


Generación 3 (1958)

Transistores

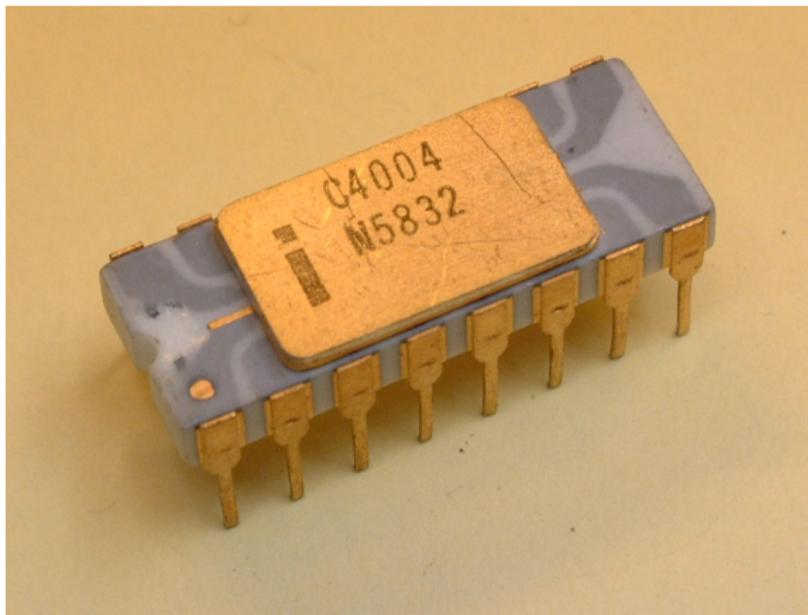


Intel (1968)



Intel

Intel 4004 (1971)



Intel

Gordon Moore y Robert Noyce



Historia

Implementación

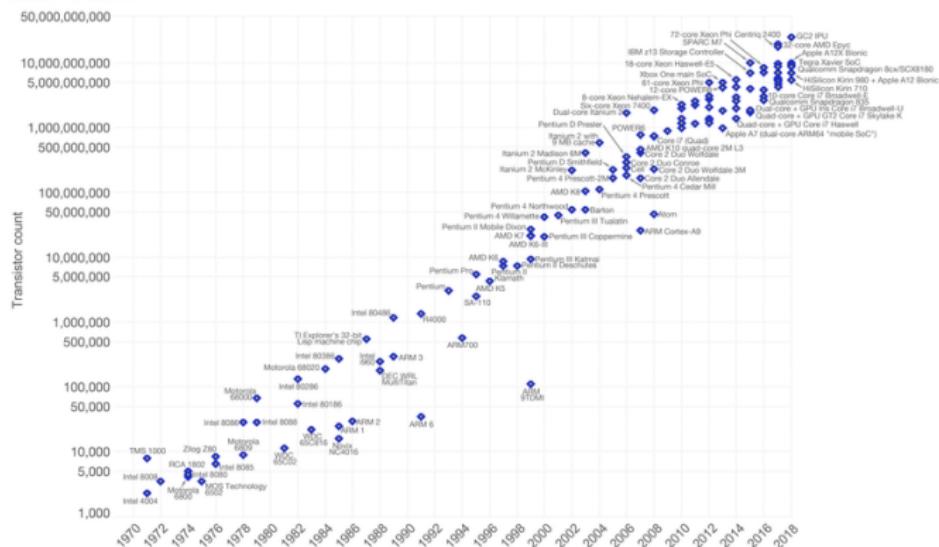
Ley de Moore

Doble cada 24 meses

Moore's Law – The number of transistors on integrated circuit chips (1971–2018)

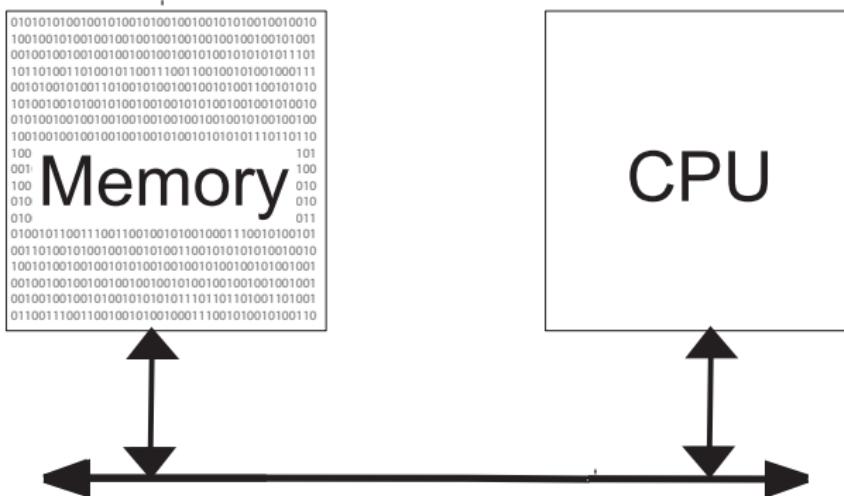
Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important as other aspects of technological progress – such as processing speed or the price of electronic products – are linked to Moore's law.

OurWorld
in Data

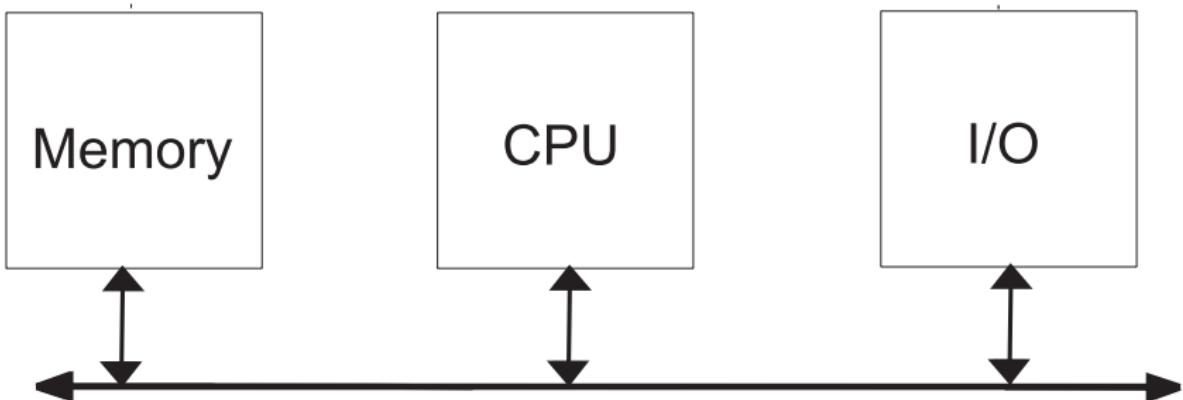


Computador

¿Qué tiene por dentro?



¿Qué tiene por dentro?

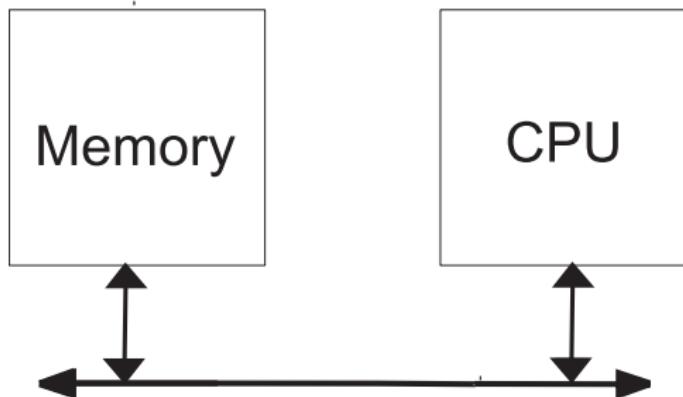


¿Y las instrucciones?

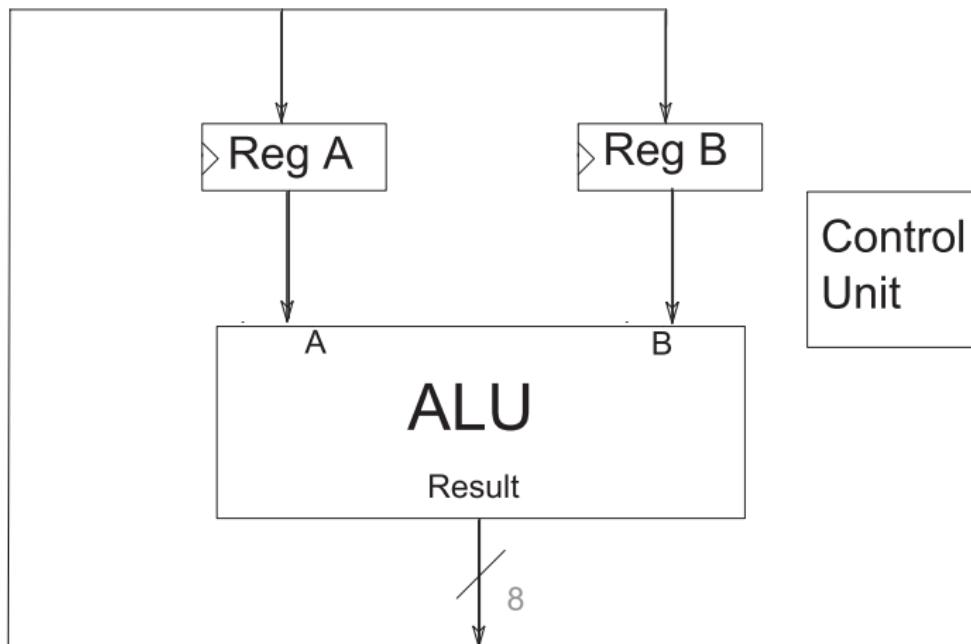
Son:

- Almacenadas en la memoria (RAM)
- Secuencia de 0s y 1s
- Recibidas por la unidad de control de la CPU
- Cada una es una cadena única de 0s y 1s

¿Cómo?



CPU



Flujo de una instrucción

- Lectura de la instrucción desde memoria (Fetch).
- Comprender la instrucción (Decode).
- Obtener la información requerida (Mem)
- Realizar la operación (Execute).
- Guardar el resultado (Write Back).

Microarquitectura e ISA

Microarquitectura

- Los distintos componentes de hardware que componen el computador.

Arquitectura del set de instrucciones (ISA)

- Se refiere al tipo, formato, características, etc., de las instrucciones soportadas por el computador.

Arquitecturas

Hardvard

- Memorias de datos e instrucciones independientes.

von Neumann

- Una sola memoria para datos e instrucciones.

ISAs

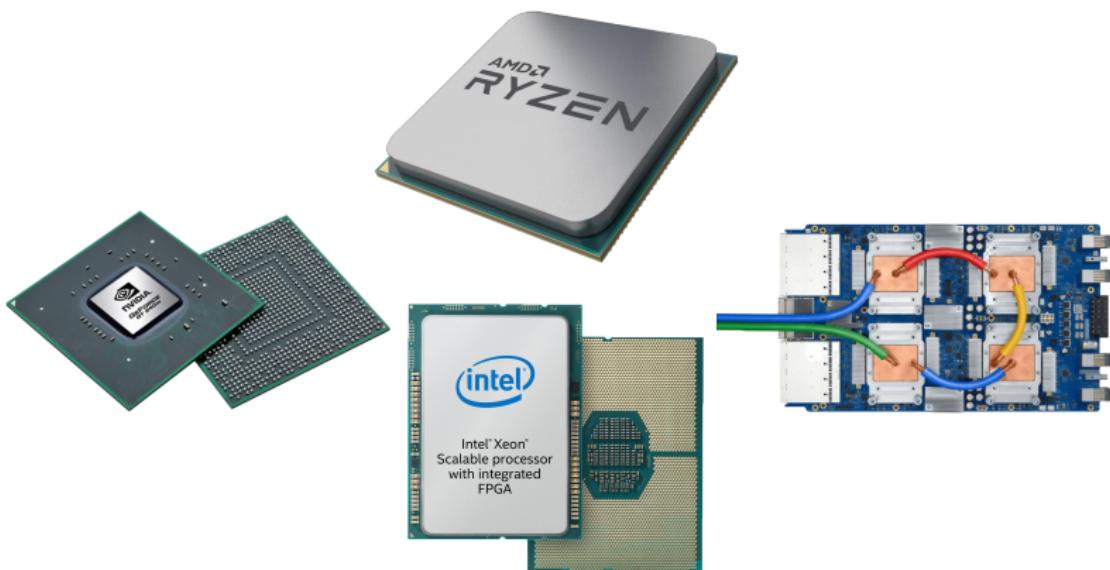
RISC

- Pocas instrucciones pequeñas y simples
- minimiza complejidad en hardware

CISC

- Muchas instrucciones de alta complejidad.

Presente y Futuro



Decimal

Regla:

- Número:

$$(1209)_{10}$$

- Posiciones:

$$\begin{array}{rcccc} & 10^3 & 10^2 & 10^1 & 10^0 \\ \hline & 1 & 2 & 0 & 9 \end{array}$$

$$\sum_{k=0}^{n-1} s_k \times 10^k$$

Base b

Regla:

- Número:

$$(wxyz)_b$$

- Posiciones:

$$\begin{array}{r} b^3 & b^2 & b^1 & b^0 \\ \hline w & x & y & z \end{array}$$

$$\sum_{k=0}^{n-1} s_k \times b^k$$

Álgebra de Boole

Operador No (\neg):

A	not(A)
F	V
V	F

Operador O (\vee):

A	B	A or B
F	F	F
F	V	V
V	F	V
V	V	V

Operador Y (\wedge):

A	B	A and B
F	F	F
F	V	F
V	F	F
V	V	V

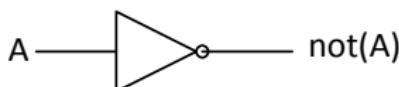
Operador O exclusivo (\oplus):

$$A \oplus B = (A \wedge \neg B) \vee (B \wedge \neg A)$$

Compuertas Lógicas

NOT

No (\neg):



C

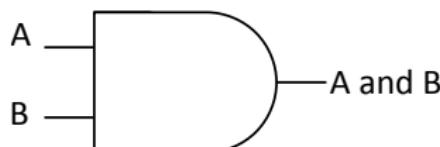
Tabla de valores:

not_a = $\sim a$;

A	not(A)
0	1
1	0

AND

Y (\wedge):



C

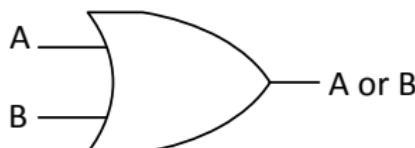
Tabla de valores:

```
a_and_b = a & b;
```

A	B	A and B
0	0	0
0	1	0
1	0	0
1	1	1

OR

O (\vee):



C

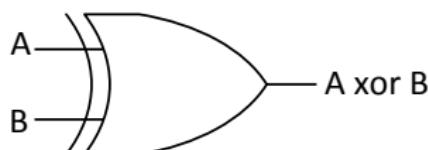
Tabla de valores:

a_or_b = a | b;

A	B	A or B
0	0	0
0	1	1
1	0	1
1	1	1

XOR

O Exclusivo (\oplus):



C

Tabla de valores:

```
a_xor_b = a ^ b;
```

A	B	A xor B
0	0	0
0	1	1
1	0	1
1	1	0

Sumador

Sumador de 1 Bit

HalfAdder

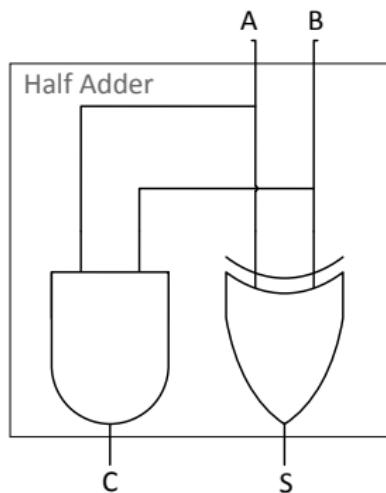


Tabla de valores

A	B	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Sumador de 1 Bit intermedio

FullAdder

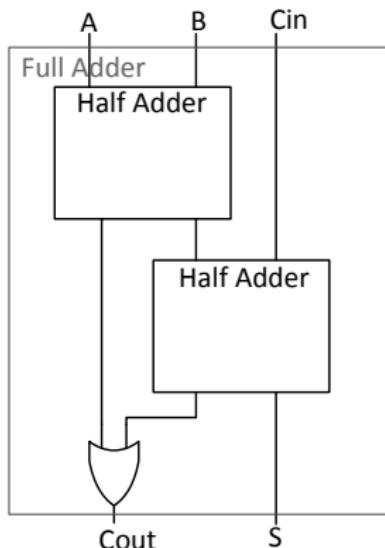


Tabla de valores

A	B	Cin	Cout	S
0	0	0	0	0
0	1	0	0	1
1	0	0	0	1
1	1	0	1	0
0	0	1	0	1
0	1	1	1	0
1	0	1	1	0
1	1	1	1	1

Sumador de 4 Bit

4-bit Adder

