PROYECTO 2

Presentado por:

Isabella Ocampo Sánchez

Curso:

Organización de computadores

Profesor:

Jose Luis Montoya Pareja

Universidad EAFIT

Ingeniería de Sistemas

Medellín, Colombia

2026

UNIVERSIDAD EAFIT

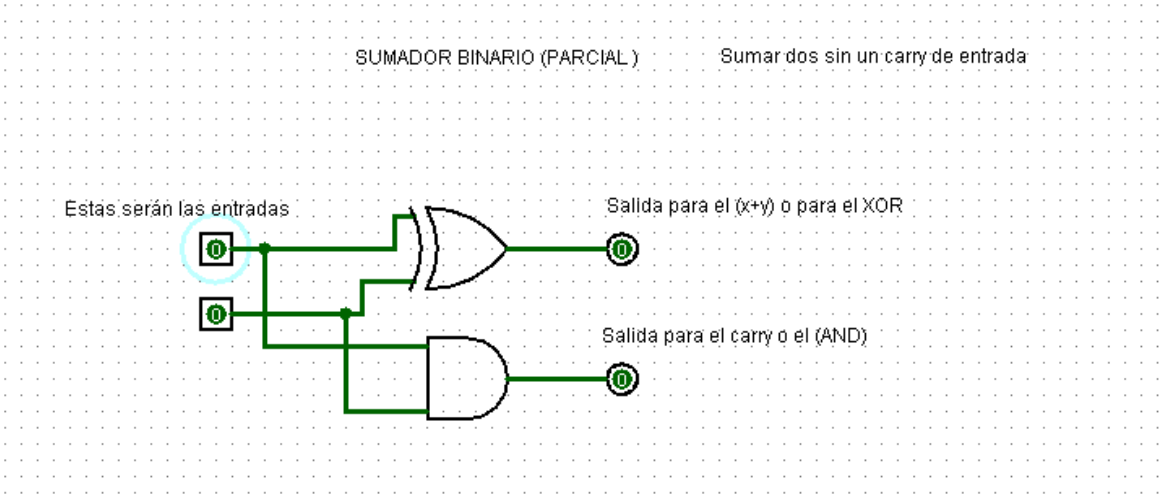
Tabla de contenido

CHIP HalfAdder

· HalFadder

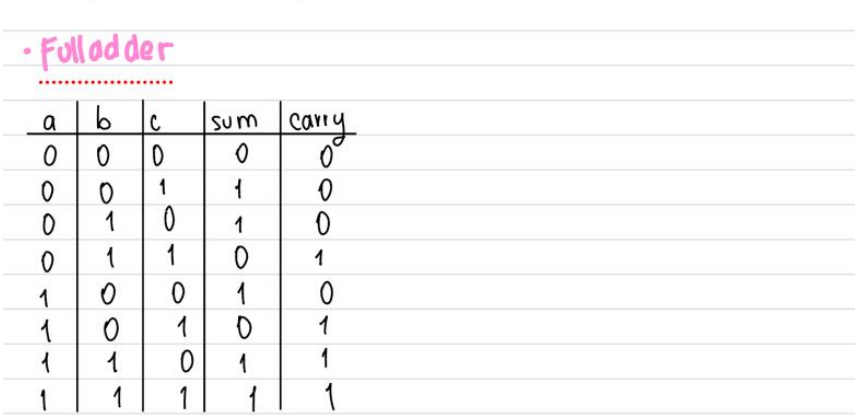
a	b	sum (XOR)	carry (AND)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

```
CHIP HalfAdder {

   IN a, b;    // 1-bit inputs

   OUT sum,    // Right bit of a + b

       carry;  // Left bit of a + b


   PARTS:

   Xor(a= a, b= b , out=sum );

   And(a= a, b= b, out= carry);

}
```



UNIVERSIDAD EAFIT

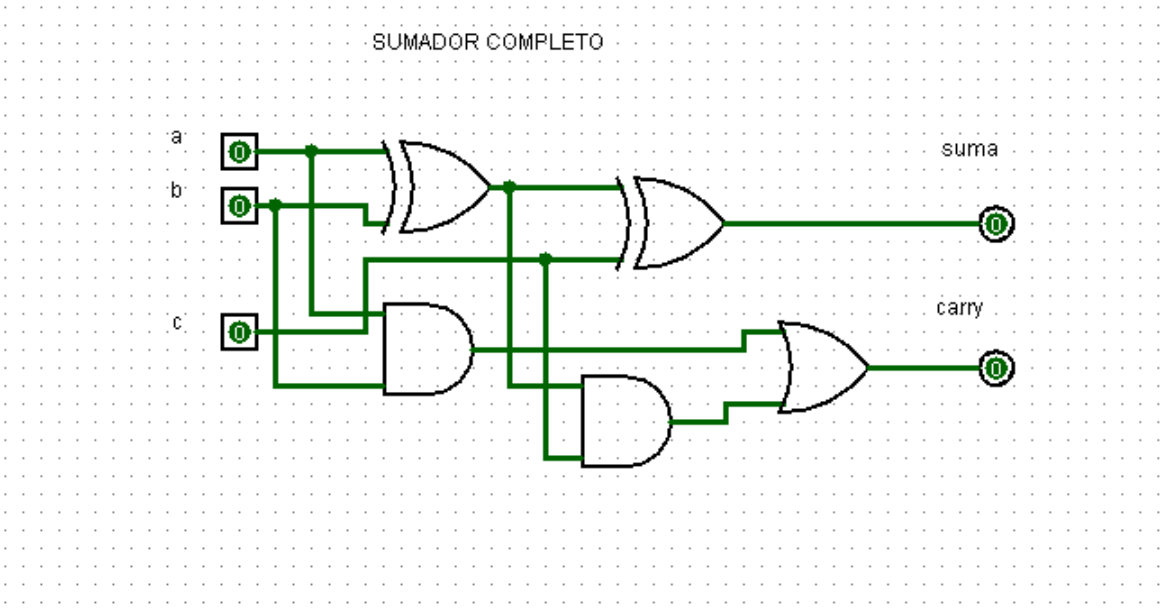
CHIP FullAdder



- Fulladder

a	b	c	sum	carry
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	1	1
1	1	1	1	1

```
CHIP FullAdder {

   IN a, b, c;  // 1-bit inputs

   OUT sum,     // Right bit of a + b + c

      carry;   // Left bit of a + b + c


   PARTS:

   Xor(a= a, b= b, out= Xor1);

   And(a= a, b=b , out=And1 );

   Xor(a= Xor1, b= c, out=sum );

   And(a=Xor1 , b= c, out=And2 );

   Or(a=And1 , b= And2, out= carry);


}
```



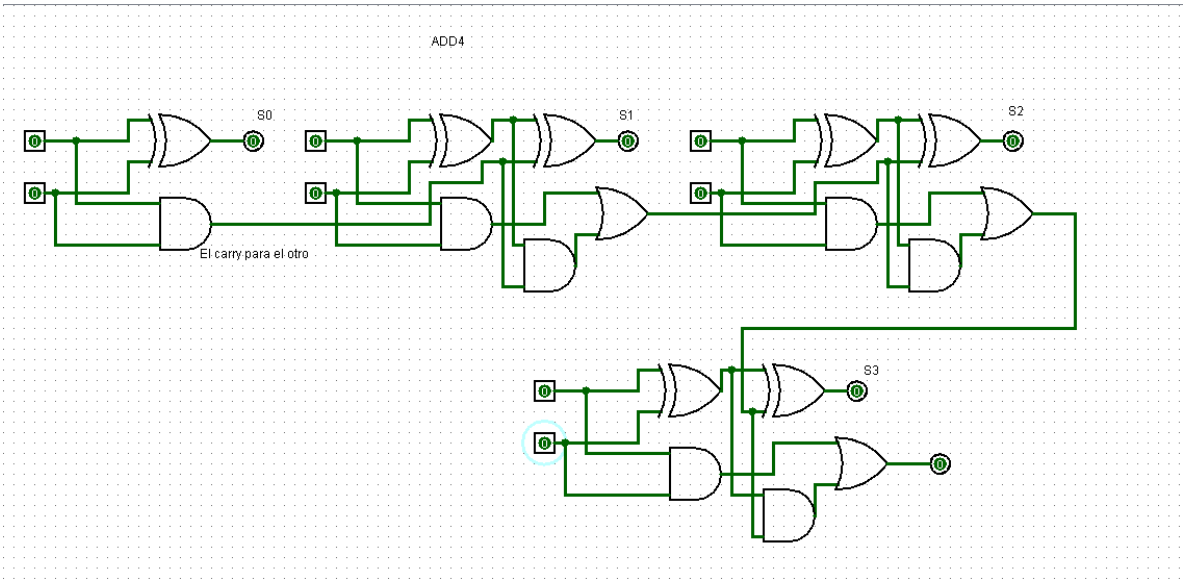
CHIP Add4

- Add 4

Bit	Componente	a[i]	b[i]	carry In	SumOut	Carry Out
0	HalfAdder	a[0]	b[0]	—	Out[0]	C_0
1	FullAdder	a[1]	b[1]	C_0	Out[1]	C_1
2	FullAdder	a[2]	b[2]	C_1	Out[2]	C_2
3	FullAdder	a[3]	b[3]	C_2	Out[3]	C_8 (descartado)

```
CHIP Add4 {

    IN a[4], b[4];

    OUT out[4], carry;


    PARTS:

    // Bit 0: El inicio (puedes usar HalfAdder o FullAdder con c=false)

    HalfAdder(a=a[0], b=b[0],sum=out[0], carry=c0);

    // Bit 1: Recibe el acarreo c0
```

```
    FullAdder(a=a[1], b=b[1], c=c0, sum=out[1], carry=c1);

    // Bit 2: Recibe el acarreo c1

    FullAdder(a=a[2], b=b[2], c=c1, sum=out[2], carry=c2);

    // Bit 3: Recibe el acarreo c2 y bota el carry final

    FullAdder(a=a[3], b=b[3], c=c2, sum=out[3], carry=carry);

}
```



CHIP Add16

. Add 16 (no tiene tabla de verdad completa por tener 2^{32} combinaciones)

Tabla genérica — El bit (0) utiliza un HalfAdder (no recibe acarreo de entrada), mientras que los bits 1 a 15 utilizan FullAdders en cascada

Bit	a[i]	b[i]	Carry In	Sum Out	Carry Out
0	a_0	b_0	0	out_0	$carry_0$
1	a_i	b_1	$carry_{i-1}$	Out_i	$carry_1$

Bit	Componente	a[i]	b[i]	Carry In	Sum Out	Carry Out
0	HalfAdder	a_0	b_0	—	out_0	$carry_0$
1	FullAdder	a_1	b_1	$carry_0$	out_1	$carry_1$
2	FullAdder	a_2	b_2	$carry_1$	Out_2	$Carry_2$
...	...	...	...	...	...	...
15	FullAdder	a_{15}	b_{15}	$carry_{14}$	Out_{15}	$Carry_{15}$ (descartado)

```
CHIP Add16 {

   IN a[16], b[16];

   OUT out[16];


   PARTS:

   HalfAdder(a=a[0],  b= b[0], sum=out[0] , carry= carry0);

   FullAdder(a=a[1] , b= b[1], c=carry0 , sum= out[1], carry= carry1);

   FullAdder(a=a[2] , b= b[2], c=carry1 , sum= out[2], carry= carry2);

   FullAdder(a=a[3] , b= b[3], c=carry2 , sum= out[3], carry= carry3);

   FullAdder(a=a[4] , b= b[4], c=carry3 , sum= out[4], carry= carry4);

   FullAdder(a=a[5] , b= b[5], c=carry4 , sum= out[5], carry= carry5);

   FullAdder(a=a[6] , b= b[6], c=carry5 , sum= out[6], carry= carry6);

   FullAdder(a=a[7] , b= b[7], c=carry6 , sum= out[7], carry= carry7);

   FullAdder(a=a[8] , b= b[8], c=carry7 , sum= out[8], carry= carry8);

   FullAdder(a=a[9] , b= b[9], c=carry8 , sum= out[9], carry= carry9);
```

FullAdder(a=a[10] , b= b[10], c=carry9 , sum= out[10], carry= carry10);

FullAdder(a=a[11] , b= b[11], c=carry10 , sum= out[11], carry= carry11);

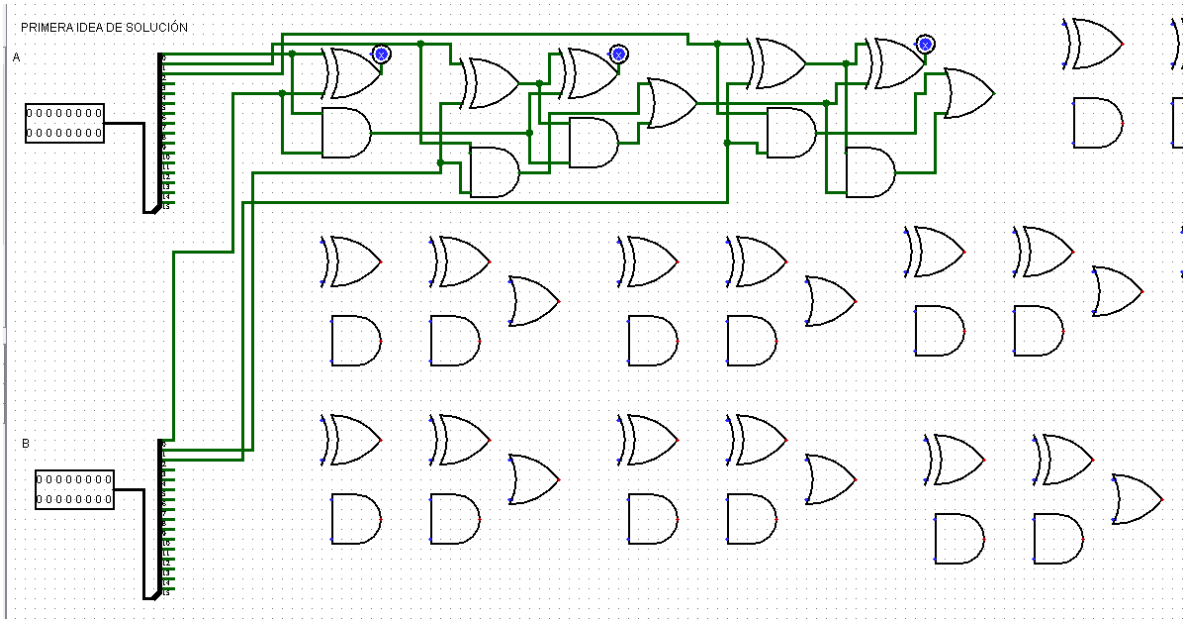
FullAdder(a=a[12] , b= b[12], c=carry11, sum= out[12], carry= carry12);

FullAdder(a=a[13] , b= b[13], c=carry12, sum= out[13], carry= carry13);

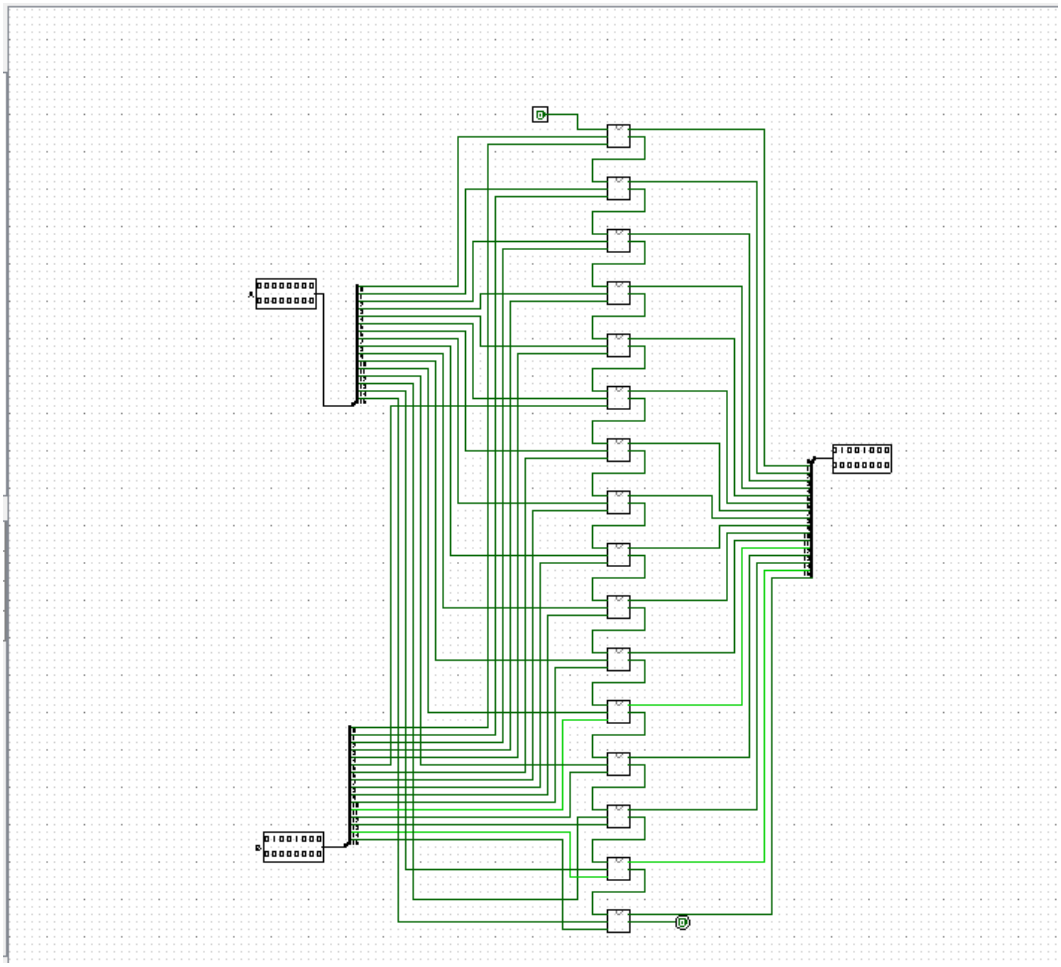
FullAdder(a=a[14] , b= b[14], c=carry13, sum= out[14], carry= carry14);

FullAdder(a=a[15] , b= b[15], c=carry14, sum= out[15], carry= carry15);

}



SOLUCIÓN FINAL Y MEJORADA

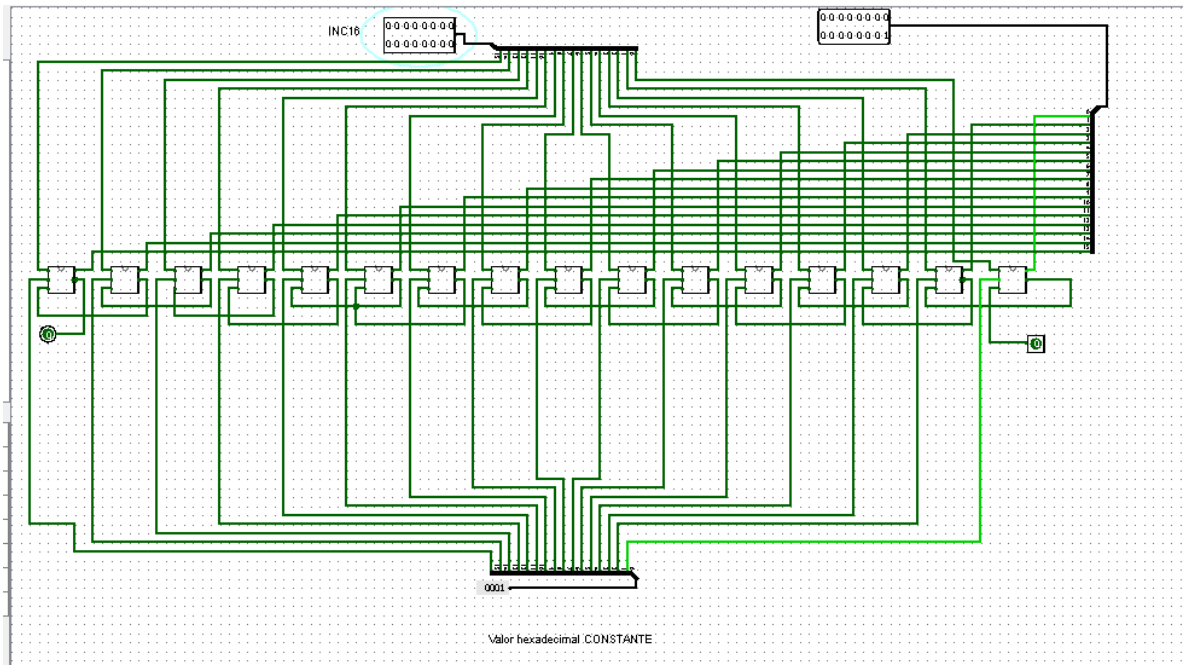


CHIP Inc16

- Inc16

Bit	Componente	Entrada	Carry In	Salida	Carry Out
0	HalfAdder	in[0]	1	out[0]	C_0
1	HalfAdder	in[1]	C_0	out[1]	C_1
2	HalfAdder	in[2]	C_1	out[2]	C_2
3	HalfAdder	in[3]	C_2	out[3]	C_3
4	HalfAdder	in[4]	C_3	out[4]	C_4
5	HalfAdder	in[5]	C_4	out[5]	C_5
6	HalfAdder	in[6]	C_5	out[6]	C_6
7	HalfAdder	in[7]	C_6	out[7]	C_7
⋮	HalfAdder	⋮	⋮	⋮	⋮
15	HalfAdder	in[15]	C_{14}	out[15]	ignored

```
CHIP Inc16 {
    IN in[16];
    OUT out[16];

    PARTS:
    HalfAdder(a=in[0], b=true, sum=out[0], carry=c0);
    HalfAdder(a=in[1], b=c0, sum=out[1], carry=c1);
    HalfAdder(a=in[2], b=c1, sum=out[2], carry=c2);
    HalfAdder(a=in[3], b=c2, sum=out[3], carry=c3);
    HalfAdder(a=in[4], b=c3, sum=out[4], carry=c4);
    HalfAdder(a=in[5], b=c4, sum=out[5], carry=c5);
    HalfAdder(a=in[6], b=c5, sum=out[6], carry=c6);
    HalfAdder(a=in[7], b=c6, sum=out[7], carry=c7);
    HalfAdder(a=in[8], b=c7, sum=out[8], carry=c8);
    HalfAdder(a=in[9], b=c8, sum=out[9], carry=c9);
    HalfAdder(a=in[10], b=c9, sum=out[10], carry=c10);
    HalfAdder(a=in[11], b=c10, sum=out[11], carry=c11);
    HalfAdder(a=in[12], b=c11, sum=out[12], carry=c12);
    HalfAdder(a=in[13], b=c12, sum=out[13], carry=c13);
    HalfAdder(a=in[14], b=c13, sum=out[14], carry=c14);
    HalfAdder(a=in[15], b=c14, sum=out[15], carry=ignored);
}
```


INC16
0001
Valor hexadecimal CONSTANTE