

# Circuitos aritméticos - Meio somador

---

## Autores

Gabriel A. F. Souza, Gustavo D. Colletta, Leonardo B. Zoccal, Odilon O. Dutra
---

Unifei
--------

## Histórico de Revisões

10 de janeiro de 2025	1.0	Primeira versão do documento.
-----------------------	-----	-------------------------------

# Tópicos

---

- Soma Binária
- Circuitos Digitais para Soma Binária
- Implementações do circuito Meio Somador
- Exercícios



# Soma Binária

# O que é a soma binária?

---

A soma binária é uma operação fundamental em circuitos digitais e sistemas computacionais, sendo a base para a realização de operações aritméticas. Ela segue as mesmas regras básicas da adição decimal, mas utiliza o sistema de numeração binário, composto apenas pelos dígitos 0 e 1.

## Regras Básicas da Soma Binária

---

Bit 1	Bit 2	Soma ( $S$ )	Carry ( $C$ )
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Aqui:

- $S$  (Sum) é o bit de soma resultante.
- $C$  (Carry) é o bit de transporte, que é “carregado” para a próxima coluna de soma.

# Soma de Múltiplos Bits

---

Quando somamos números binários com mais de um bit, o transporte ( $C$ ) da soma de um par de bits é propagado para a próxima coluna, assim como na soma decimal. Por exemplo:

Considere somar dois números binários de 4 bits:

$$A = 1011 \quad \text{e} \quad B = 0111$$

# Soma de Múltiplos Bits

---

Realizando a soma bit a bit:

Posição	<i>A</i>	<i>B</i>	Carry In	Soma	Carry Out
Bit menos significativo (LSB)	1	1	0	0	1
Próximo bit	1	1	1	1	1
Próximo bit	0	1	1	0	1
Bit mais significativo (MSB)	1	0	1	0	1

O resultado final da soma é:

$$1011 + 0111 = 10010$$

Aqui, o resultado é um número de 5 bits, com o bit mais à esquerda (1) sendo o transporte final.

# Circuitos Digitais para Soma Binária

# 1. Meio Somador (Half Adder)

---

O meio somador é um circuito que realiza a soma de dois bits, mas não considera o transporte de entrada (Carry). Ele tem duas entradas ( $A$  e  $B$ ) e duas saídas:

- Soma ( $S = A \oplus B$ , onde  $\oplus$  é a operação XOR)
- Transporte ( $C = A \cdot B$ , onde  $\cdot$  é a operação AND)

## 2. Somador Completo (Full Adder)

---

O somador completo realiza a soma de três bits: os dois bits de entrada ( $A$  e  $B$ ) e um bit de transporte de entrada ( $C_{in}$ ). Ele tem duas saídas:

- Soma ( $S = A \oplus B \oplus C_{in}$ )
- Transporte ( $C_{out} = (A \cdot B) + (C_{in} \cdot (A \oplus B))$ )

Um somador completo pode ser construído conectando dois meios somadores e uma porta OR para o transporte.

### 3. Somadores de Múltiplos Bits

---

Para somar números binários com mais de um bit, os somadores completos podem ser conectados em cascata. O transporte de saída de um somador é propagado como transporte de entrada para o próximo.

# Implementações do circuito Meio Somador

# Descrição por Fluxo de Dados

---

```
1 // Módulo Half Adder (Meio Somador)
2 module half_adder (
3     input A,           // Primeiro bit de entrada
4     input B,           // Segundo bit de entrada
5     output Sum,        // Saída de soma (A + B)
6     output Carry       // Saída de transporte (A · B)
7 );
8
9 // Descrição por fluxo de dados
10 assign Sum = A ^ B;    // Soma é a operação XOR dos bits
    de entrada
11 assign Carry = A & B;  // Transporte é a operação AND dos
    bits de entrada
12
13 endmodule
```

## Diagrama esquemático

---

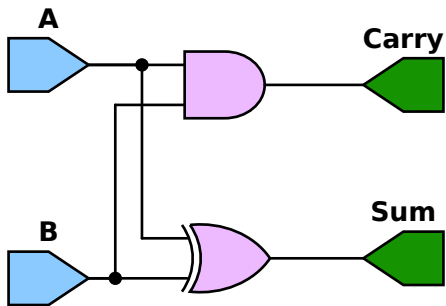


Figura 1: Esquemático do circuito meio somador.

# Descrição Estrutural

---

```
1 // Módulo Half Adder (Meio Somador) - Descrição Estrutural
2 module half_adder (
3     input A,           // Primeiro bit de entrada
4     input B,           // Segundo bit de entrada
5     output Sum,        // Saída de soma (A + B)
6     output Carry       // Saída de transporte (A · B)
7 );
8
9     //nome_da_porta (saída, entrada1, entrada2);
10    // Porta XOR para calcular a soma (Sum = A + B)
11    xor (Sum, A, B);
12
13    // Porta AND para calcular o transporte (Carry = A · B)
14    and (Carry, A, B);
15
16 endmodule
```

## Diagrama esquemático

---

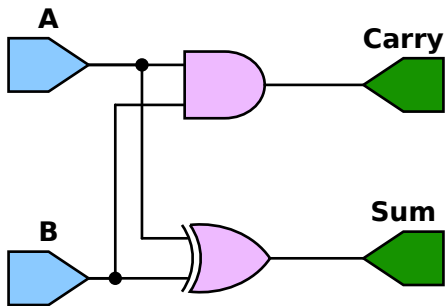


Figura 2: Esquemático do circuito meio somador.

# Descrição Comportamental 1

---

```
1 // Módulo Half Adder (Meio Somador) - Descrição Comportamental
2 module half_adder (
3     input A,           // Primeiro bit de entrada
4     input B,           // Segundo bit de entrada
5     output reg Sum,    // Saída de soma (A ^ B)
6     output reg Carry   // Saída de transporte (A & B)
7 );
8
9 // Bloco procedural always para calcular a saída
10 always @(*) begin
11     Sum = A ^ B;       // Soma (XOR)
12     Carry = A & B;     // Transporte (AND)
13 end
14
15 endmodule
```

# Diagrama esquemático 1

---

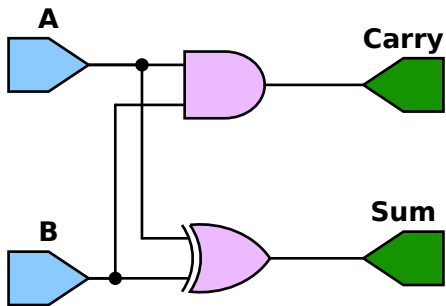


Figura 3: Esquemático do circuito meio somador.

## Descrição comportamental 2

---

```
1 // Módulo Half Adder (Meio Somador) - Descrição Comportamental
   usando operação de soma
2 module half_adder (
3     input A,           // Primeiro bit de entrada
4     input B,           // Segundo bit de entrada
5     output reg Sum,    // Saída de soma (A + B)
6     output reg Carry // Saída de transporte (A > B)
7 );
8
9 // Bloco procedural always para calcular a saída
10 always @(*) begin
11     {Carry, Sum} = A + B; // Soma binária com Carry e Sum
12 end
13
14 endmodule
```

## Diagrama esquemático 2

---

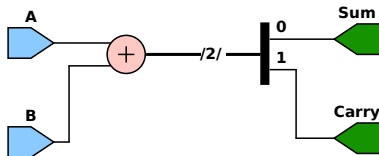


Figura 4: Esquemático do circuito meio somador.

- A operação  $+$  no Verilog é tratada como uma soma aritmética, e o Yosys otimiza essa operação para o símbolo de soma.
- Na primeira implementação, foram utilizadas portas lógicas explicitamente (XOR e AND), o que resulta em uma representação mais detalhada.
- O Yosys, ao encontrar a operação  $+$ , substitui essa operação por um símbolo de soma eficiente, ao invés de desenhar portas lógicas como na primeira implementação. Isso é uma otimização feita pelo Yosys.

# Exercícios

## Exercício único

---

Escreva um arquivo de *testbench* para testar as 4 descrições fornecidas para o circuito meio somador.

- ❶ Os 4 módulos devem ser instanciados. Portanto, as entradas serão conectadas à todos os módulos, porém as saídas devem ser separadas, ou seja, cada módulo deve ter uma saída de soma e uma de transporte (carry).
- ❷ O arquivo de *testbench* deve considerar todas as opções possíveis para as entradas.