

Laboratório Circuitos Combinacionais

Prof. Fernando Gehm Moraes

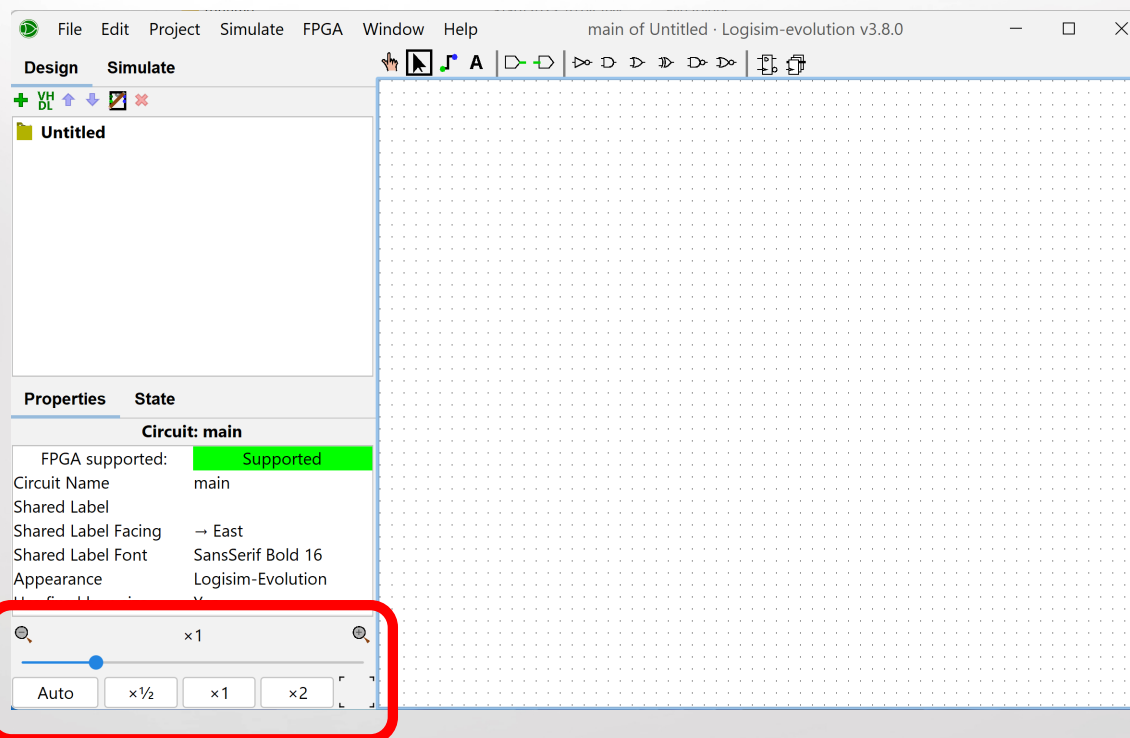


Baixar a aplicação

<https://github.com/logisim-evolution/logisim-evolution/releases/tag/v3.9.0>

<https://github.com/logisim-evolution/logisim-evolution/releases/download/v3.9.0/logisim-evolution-3.9.0-all.jar>

Tela Inicial



Zoom

PARTE I

FA: FULL ADDER

Primeiro módulo: **FA**

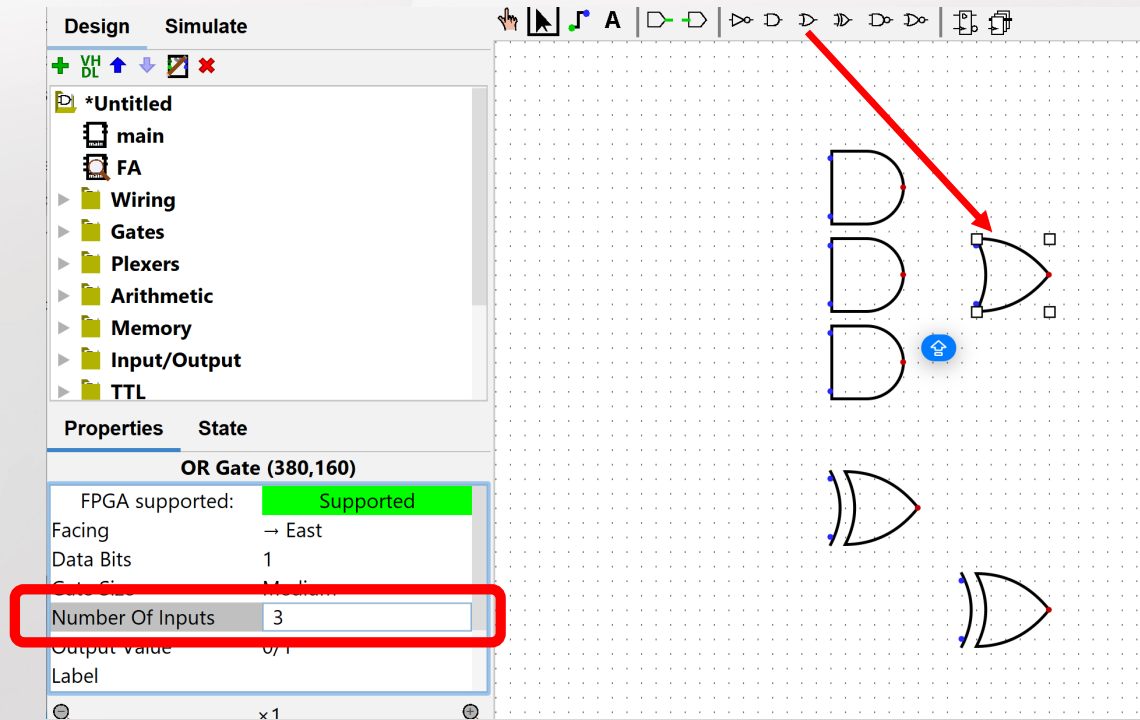
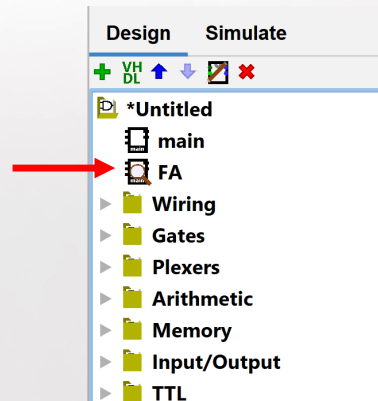
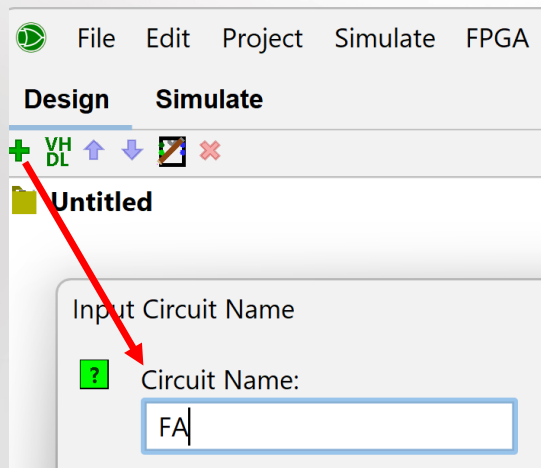
$$C_o = A.B + B.C_i + A.C_i$$

$$S = A \oplus B \oplus C_i$$

INSERIR PORTAS LOGICAS

Em destaque a seleção da or para 3 entradas

Clicar em **+** e definir o nome **FA**



Primeiro módulo: FA

$$C_o = A \cdot B + B \cdot C_i + A \cdot C_i$$

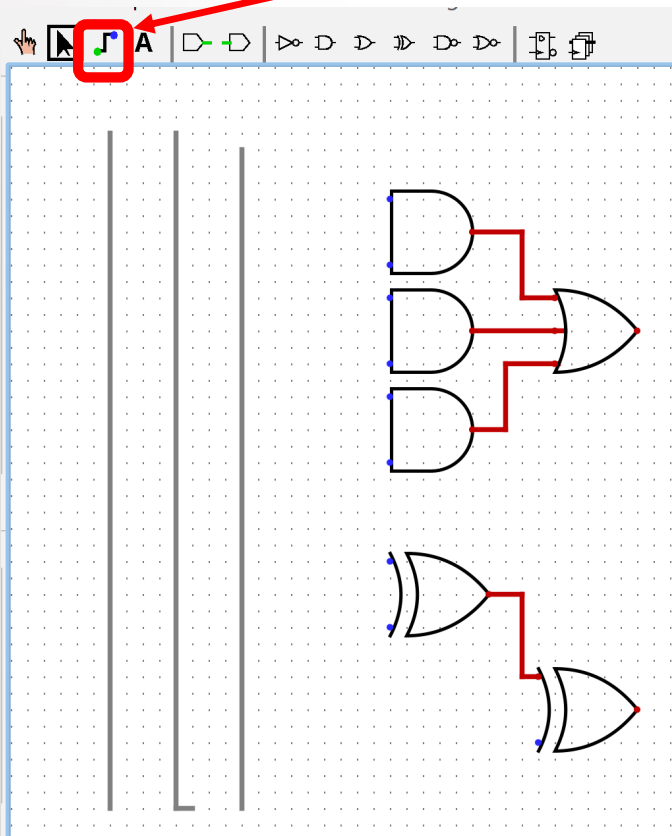
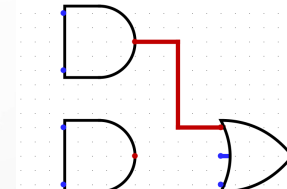
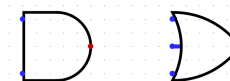
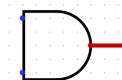
$$S = A \oplus B \oplus C_i$$

FIOS

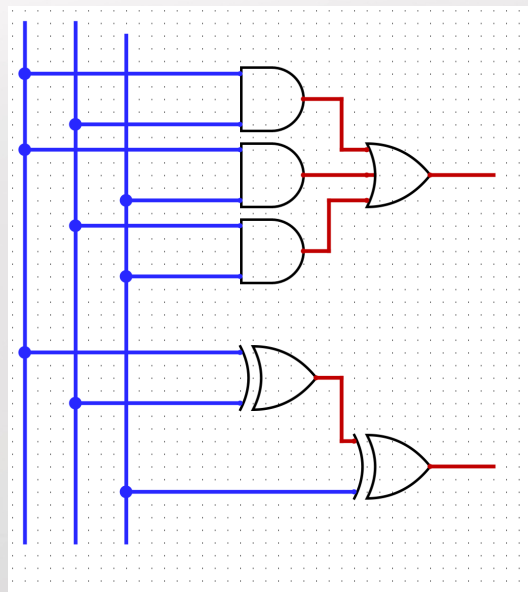
Desenhe três fios verticais para A, B, C

Lige as portas – dica: faça os fios por partes

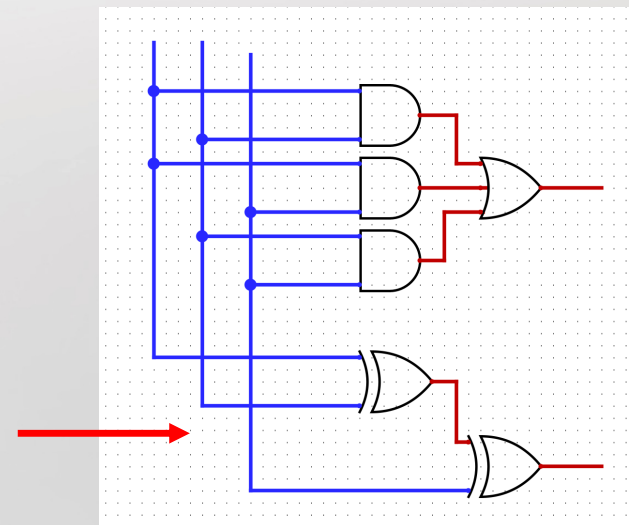
Use  para “parar” o fio



Complete as ligações



Remova os fios desnecessários (opcional)



Primeiro módulo: FA

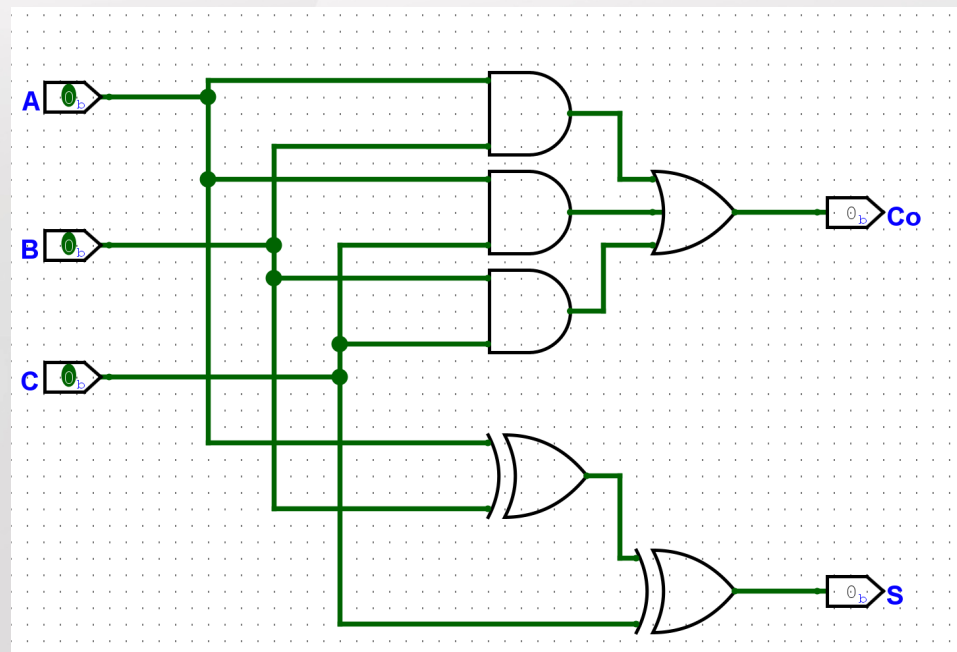
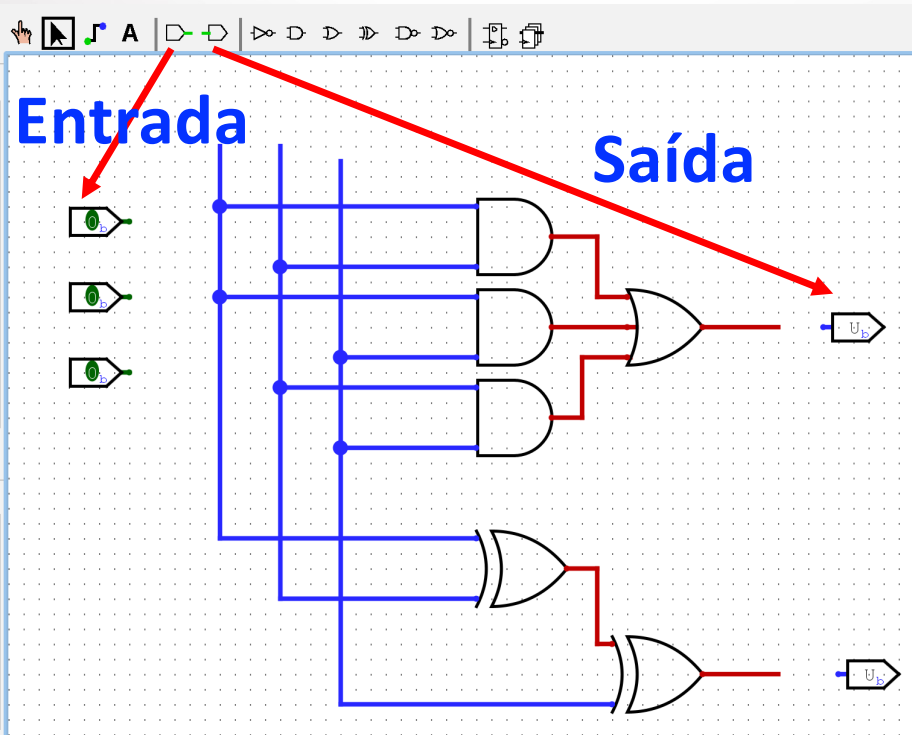
$$C_o = A \cdot B + B \cdot C_i + A \cdot C_i$$

$$S = A \oplus B \oplus C_i$$

Pinos de entrada e saída

Insira os pinos para A/B/C e Co/S

Duplo clique nos pinos para nomeá-los
Terminar as conexões
(opcional) Remover fios sobrando



Primeiro módulo: **FA**

$$C_o = A \cdot B + B \cdot C_i + A \cdot C_i$$

$$S = A \oplus B \oplus C_i$$

Testando



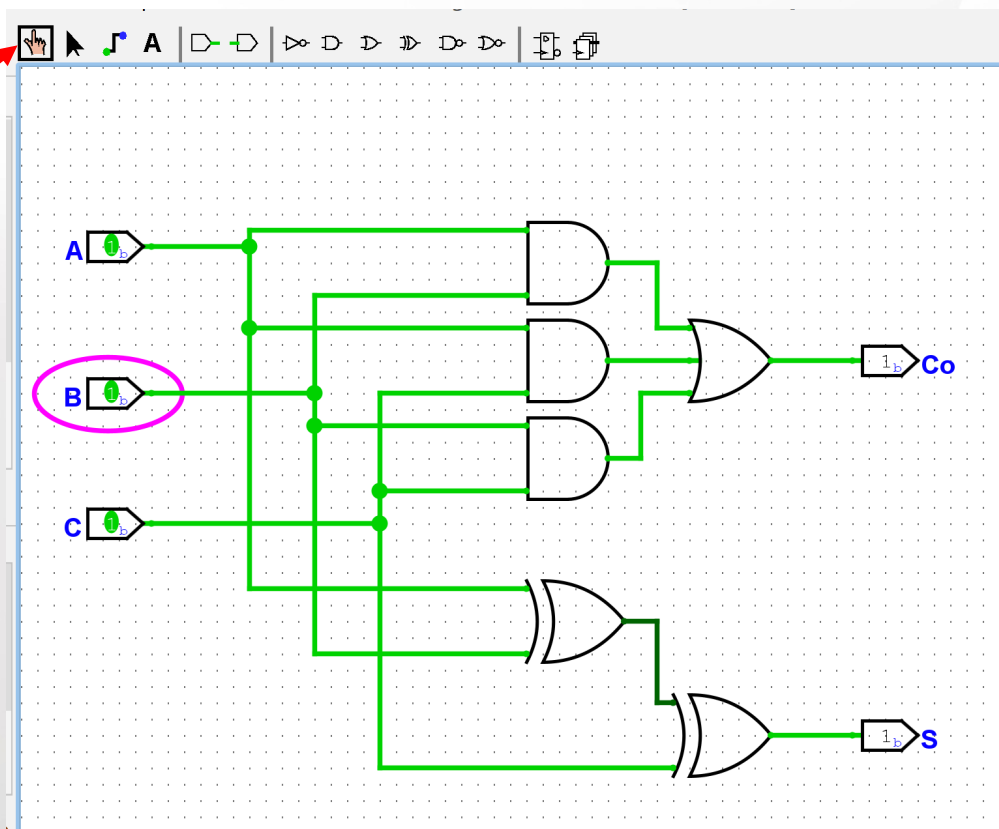
No exemplo temos

$$1 + 1 + 1$$

$$S = 1$$

$$C_o = 1$$

Testar o circuito de acordo com a tabela verdade

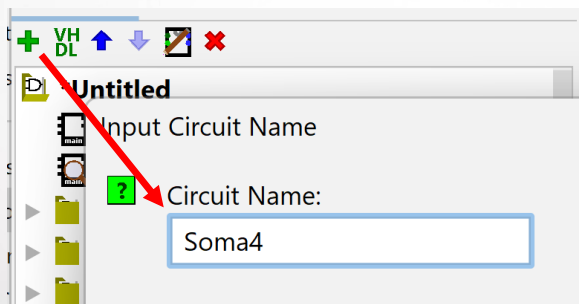


A	B	C _i	S	C _o
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

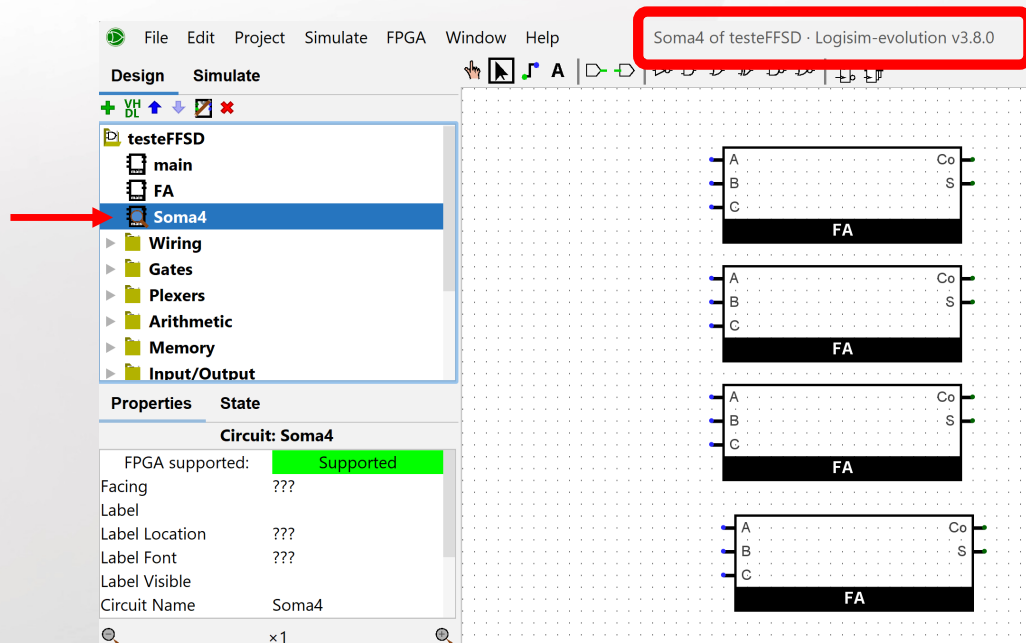
PARTE II

Soma4: Somador de 4 bits

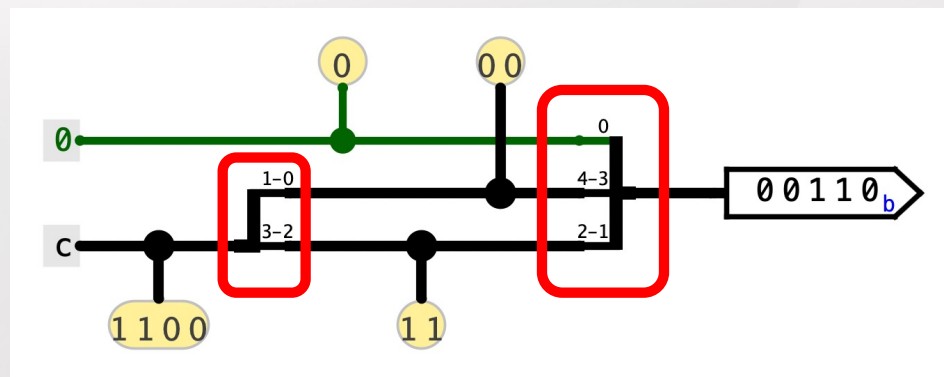
Clicar em **+** e definir o nome Soma4



Com o Soma4 selecionado, clicar no FA e depois na área de trabalho, posicionando 4 FA



Divisores (*Splitters*) permitem que você pegue um valor de múltiplos bits e o divida em partes menores ou combine vários valores de um ou mais bits em um único valor. Aqui, dividimos o número binário de 4 bits **1100** em **11** e **00**, invertendo suas posições e combinando-os com **0** para criar o número final de 5 bits: **00110**.



Clique em um divisor (*splitter*) (usando a ferramenta **Selecionar**) para ver seus atributos na barra lateral (canto inferior esquerdo). Você pode configurar atributos como o número de ramificações do divisor e a quantidade de bits em cada uma delas. Para o circuito anterior, os atributos dos divisores esquerdo e direito são os seguintes:

Properties State	
Selection: Splitter	
VHDL	Verilog
Facing	East
Fan Out	2
Bit Width In	4
Appearance	Centered
Spacing	2
Bit 0	0 (Top)
Bit 1	0 (Top)
Bit 2	1 (Bottom)
Bit 3	1 (Bottom)

Properties State	
Selection: Splitter	
VHDL	Verilog
Facing	West
Fan Out	3
Bit Width In	5
Appearance	Centered
Spacing	2
Bit 0	0 (Top)
Bit 1	2 (Bottom)
Bit 2	2 (Bottom)
Bit 3	1
Bit 4	1

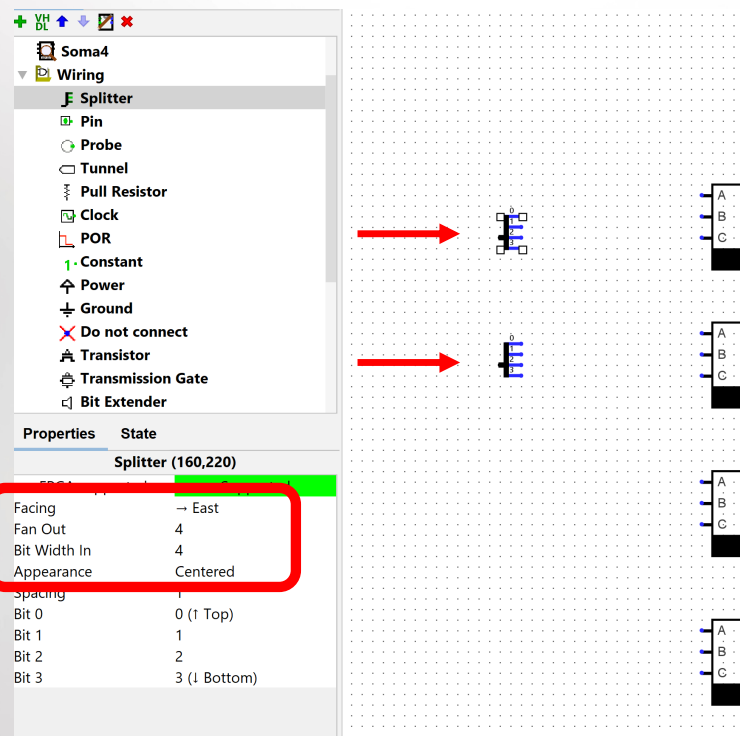
Observe que há um atributo chamado **Orientação (*Facing*)**. Você pode usá-lo para **girar o divisor**. No exemplo acima, o divisor à direita está orientado para o **Oeste**, enquanto o divisor à esquerda está orientado para o **Leste**.

Se você notar um **fio de erro em laranja**, isso significa que a **largura de bits de entrada** não corresponde à **largura de bits de saída**. Certifique-se de que, ao conectar dois componentes com um fio, você ajuste corretamente a largura de bits no menu desse componente.



Insira dois splitters (entra n fios e depois separa o n fios)

- observar a configuração destacada

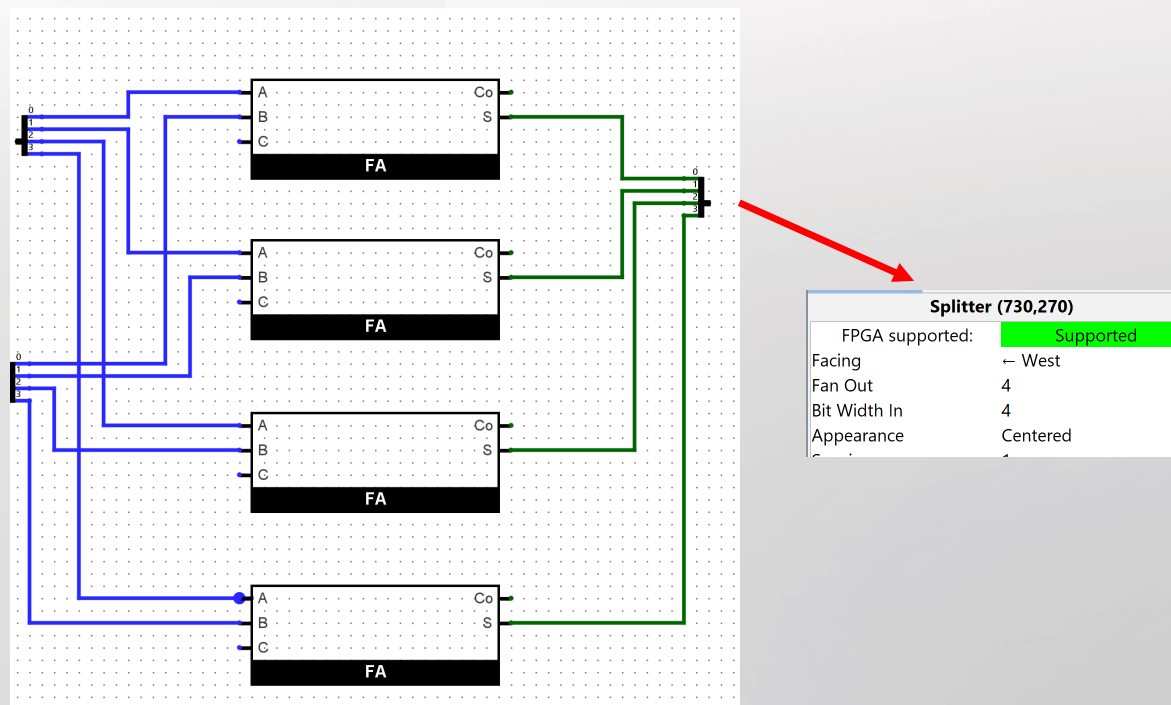


The screenshot shows the CADENCE tool interface. On the left, a component palette lists various components, including 'Splitter'. Below the palette, the 'Properties' tab for a 'Splitter (160,220)' is displayed. The properties are as follows:

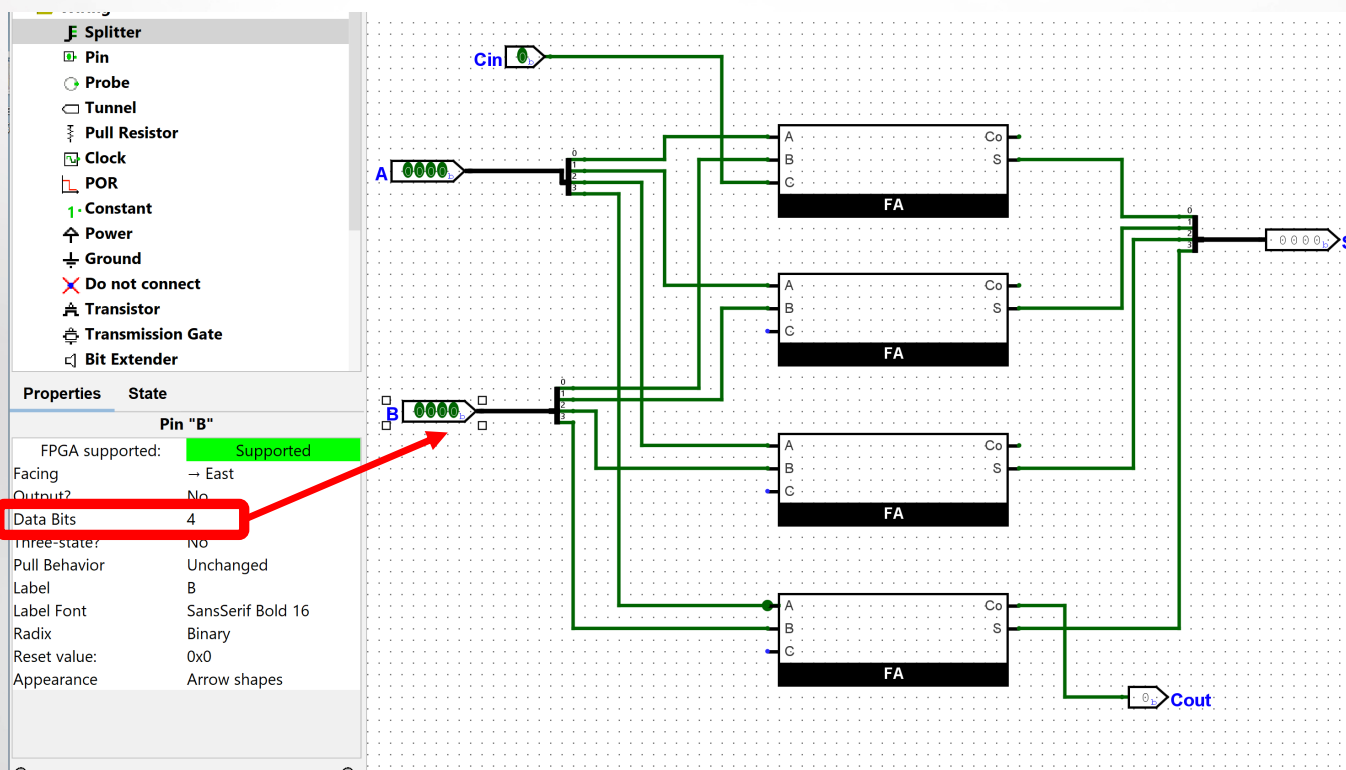
Property	Value
Facing	→ East
Fan Out	4
Bit Width In	4
Appearance	Centered
Spacing	1
Bit 0	0 (1 Top)
Bit 1	1
Bit 2	2
Bit 3	3 (1 Bottom)

The 'Facing' property is highlighted with a red rectangle.

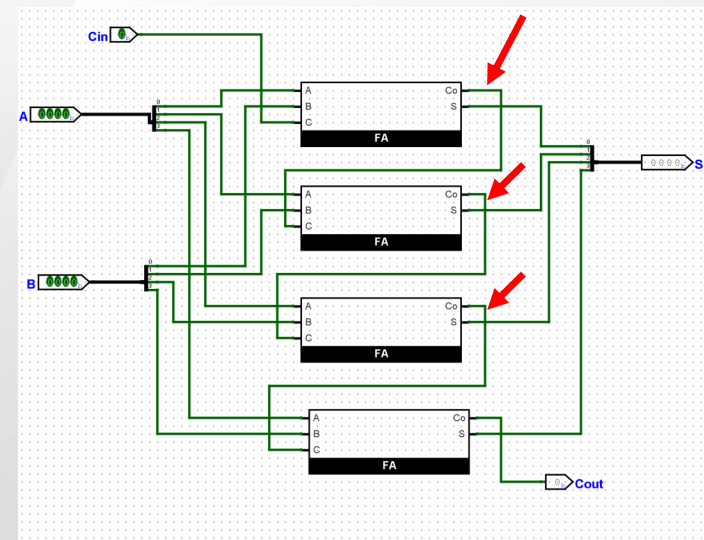
Insira um *splitter* para a soma e ligar os sinais A/B/S



Inseria os pinos Cin / A / B / S / Cout como abaixo.

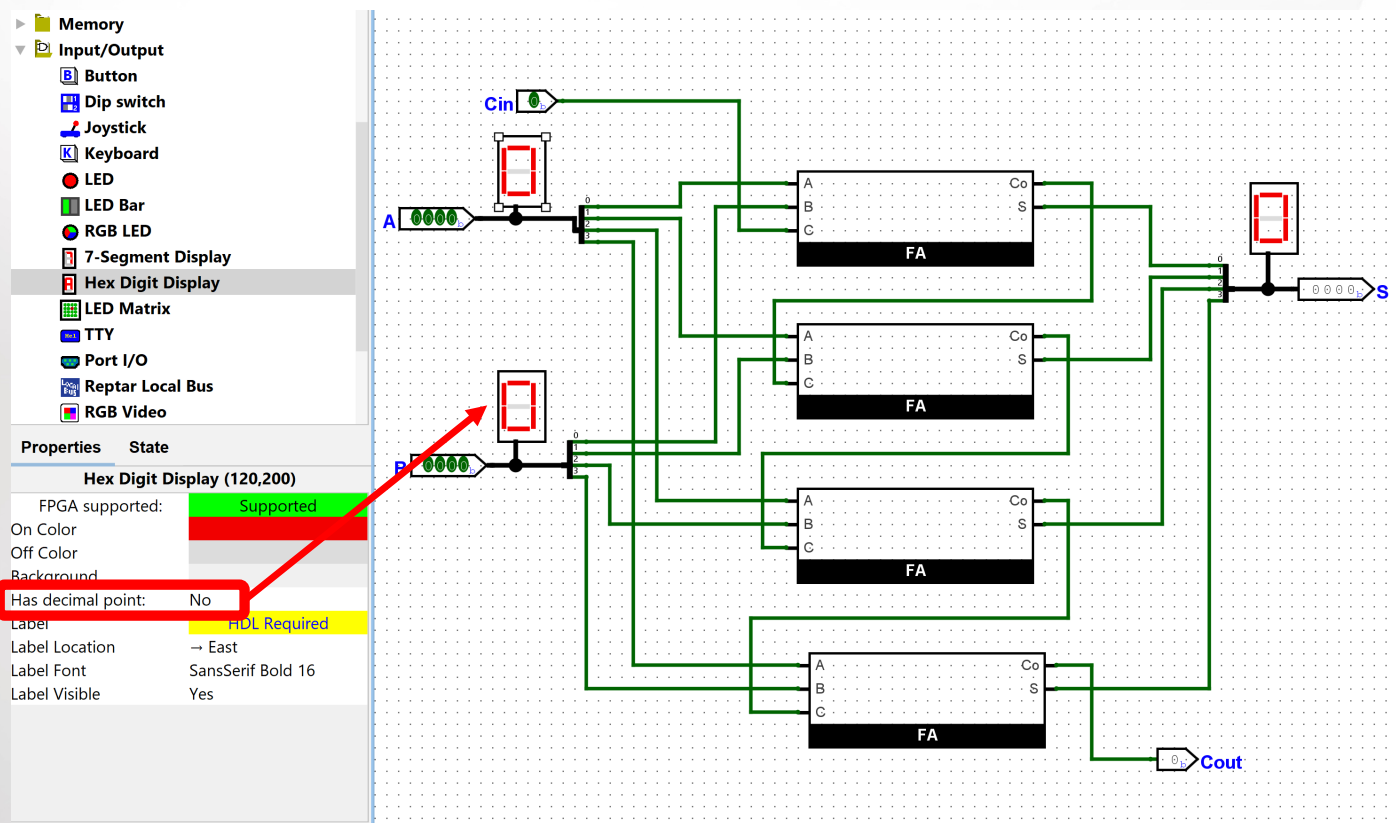


Conecte os Co nos C (Cin) como destacado



(atenção aos sinais de 4 bits)

Concluir o somador inserindo 3 “hex digit Display”, sem ponto decimal, conectando-os em A / B / S

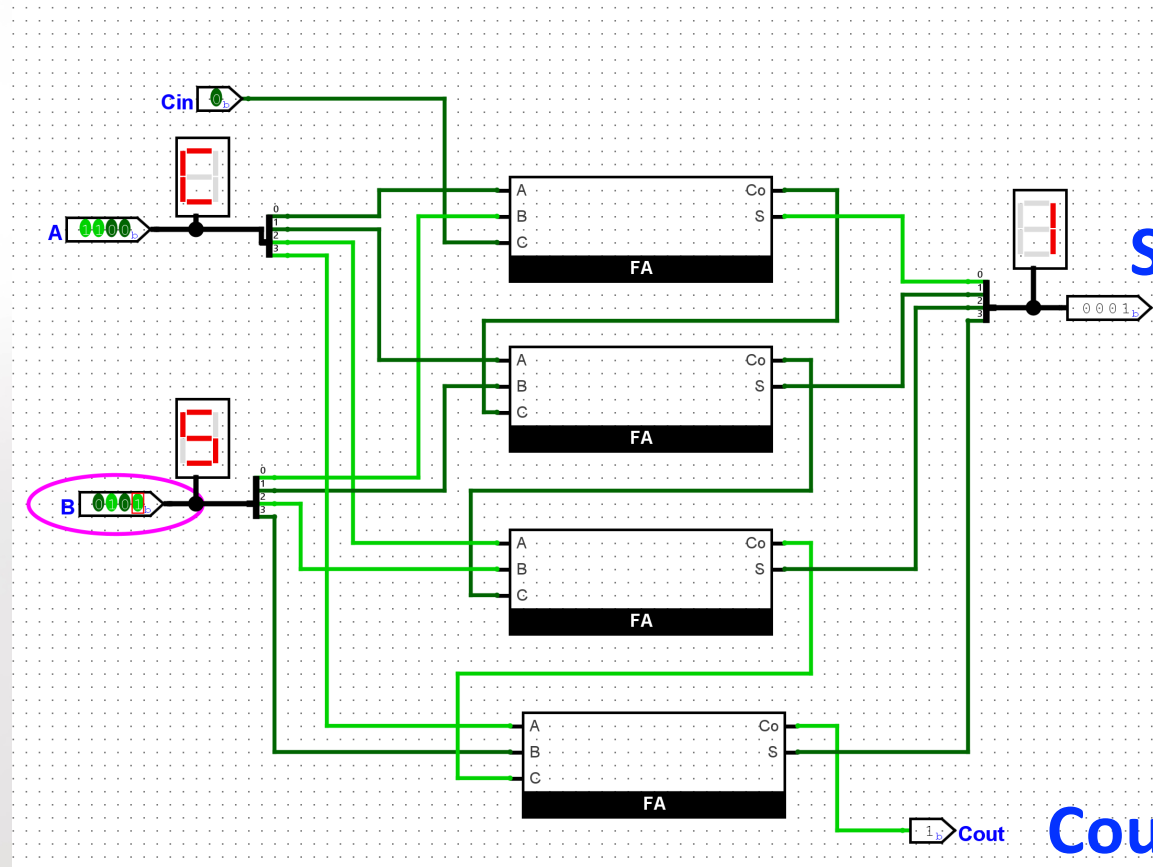


Testando



```

1
1100 (C)
+ 0101 (5)
-----
1 0001 (1)
    
```



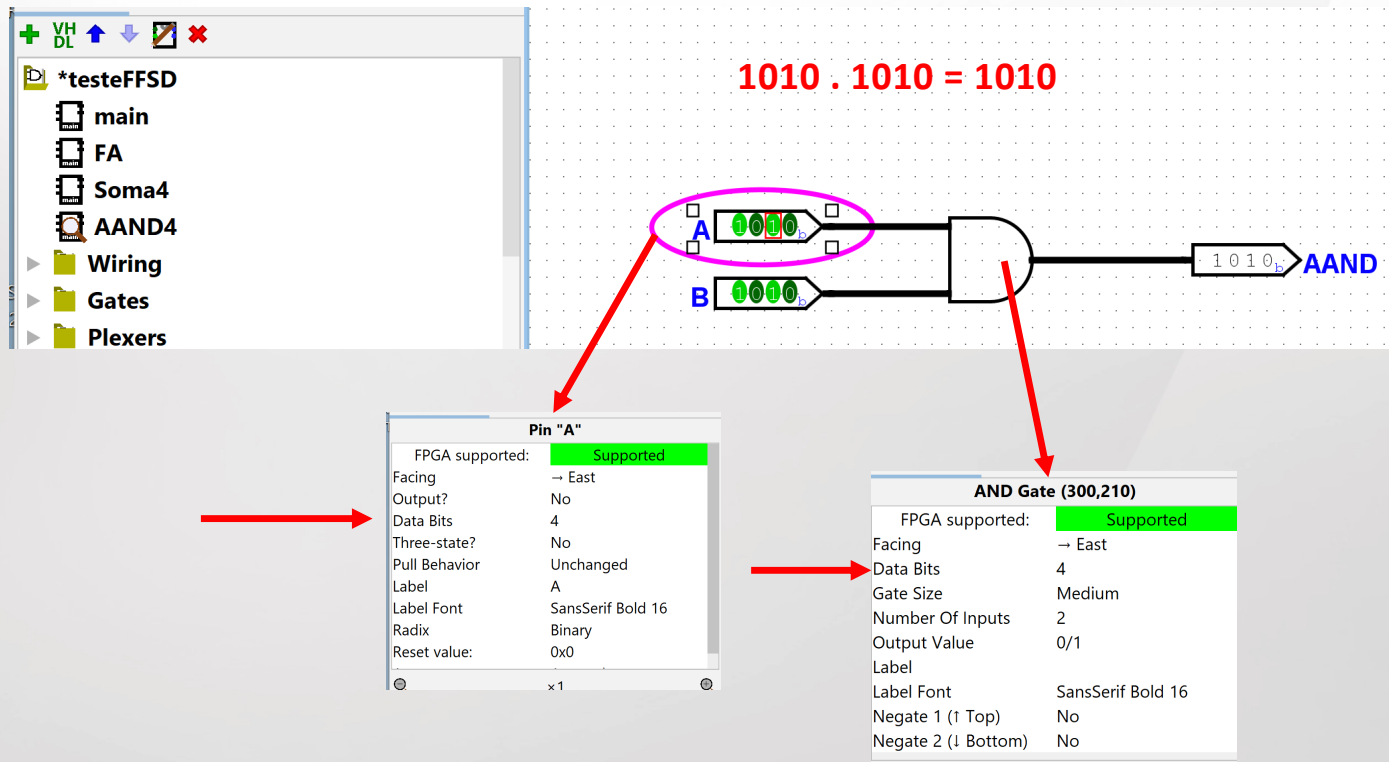
Soma=1

Cout=1

PARTE III

AND 4 bits

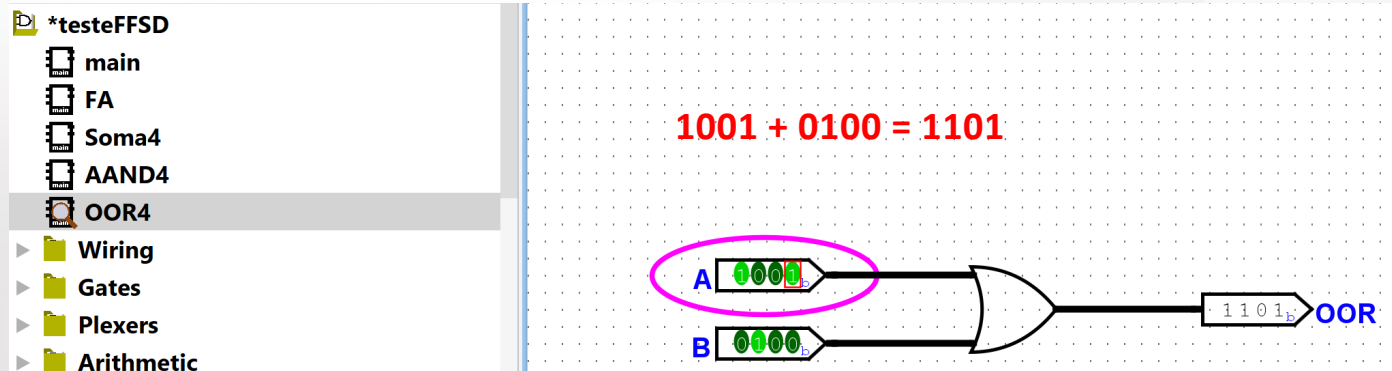
Criar um módulo AND de 4 bits como abaixo, nomeando-o AAND4



PARTE IV

OR 4 bits

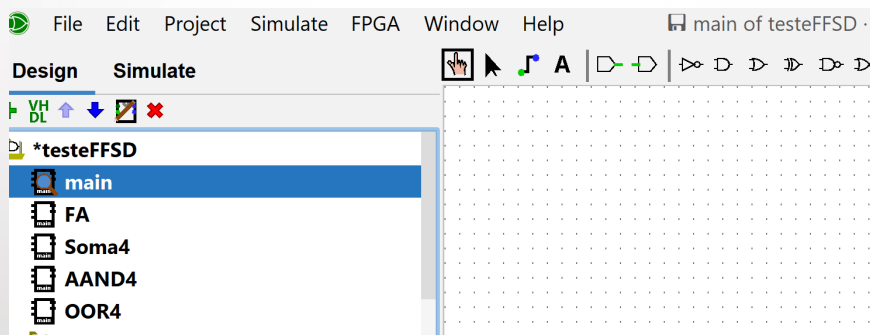
Criar um módulo OR de 4 bits como abaixo, nomeando-o OOR4



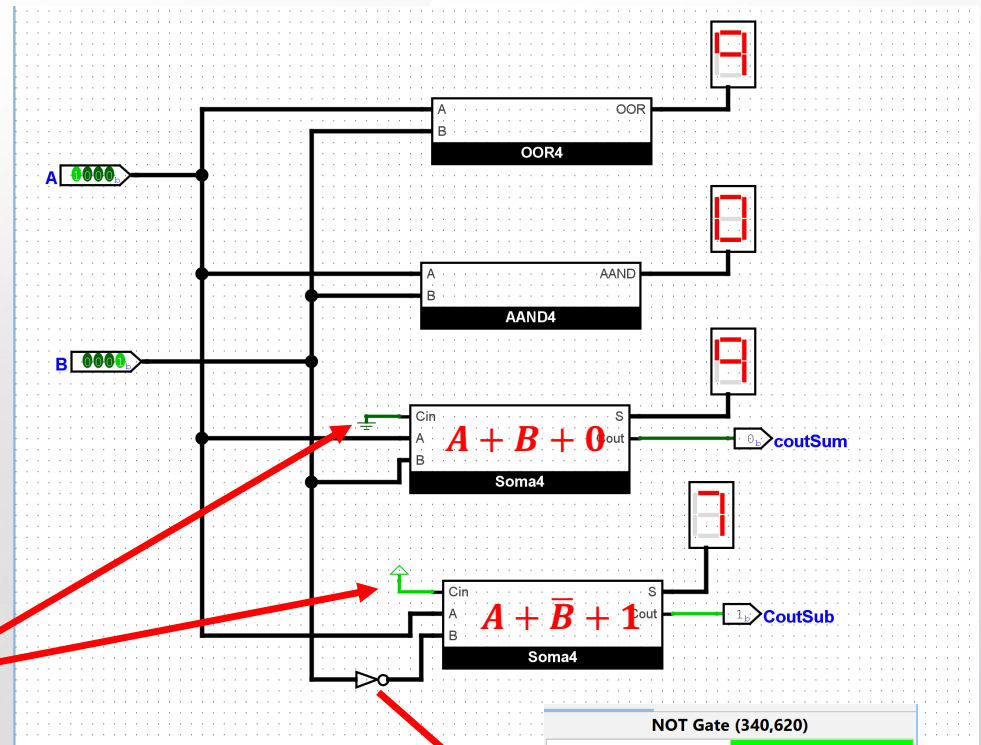
PARTE V

“TOP”

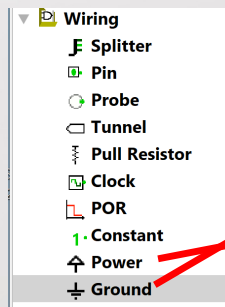
Duplo clique na “main” - a qual deve estar vazia



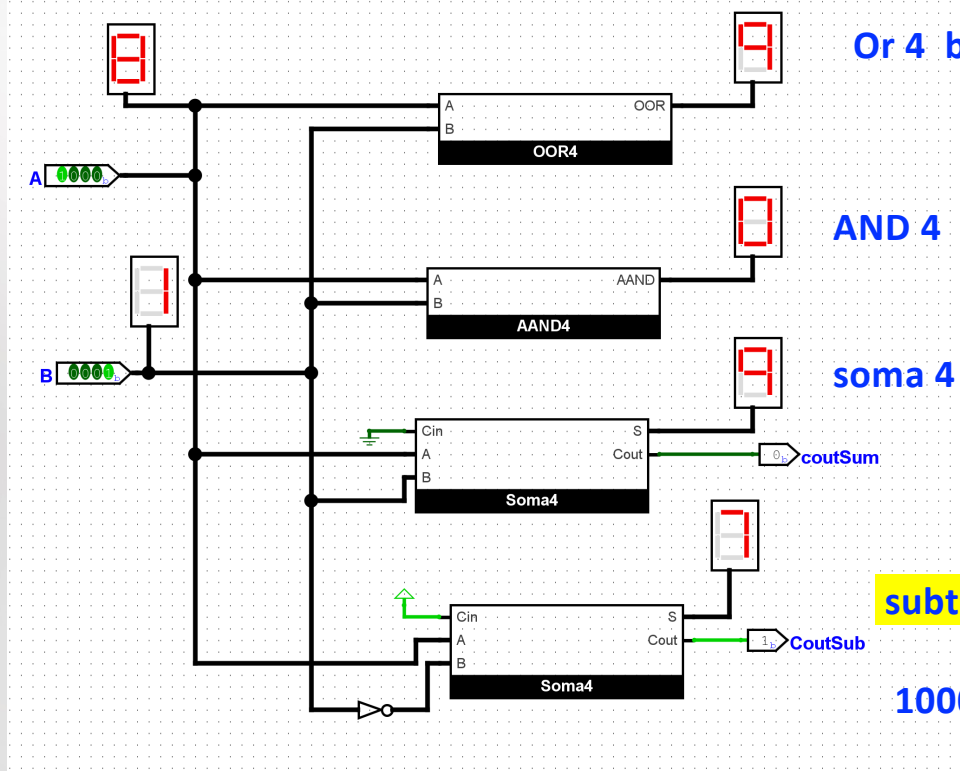
Montar o diagrama abaixo, com: OOR4, AAND4, 2x Soma4



Power: valor lógico '1'
Ground: valor lógico '0'



NOT Gate (340,620)	
FPGA supported:	Supported
Facing	→ East
Data Bits	4
Gate Size	Wide
Output Value	0/1
Label	
Label Font	SansSerif Bold 16



Or 4 bits 1000 or 0001 = 1001

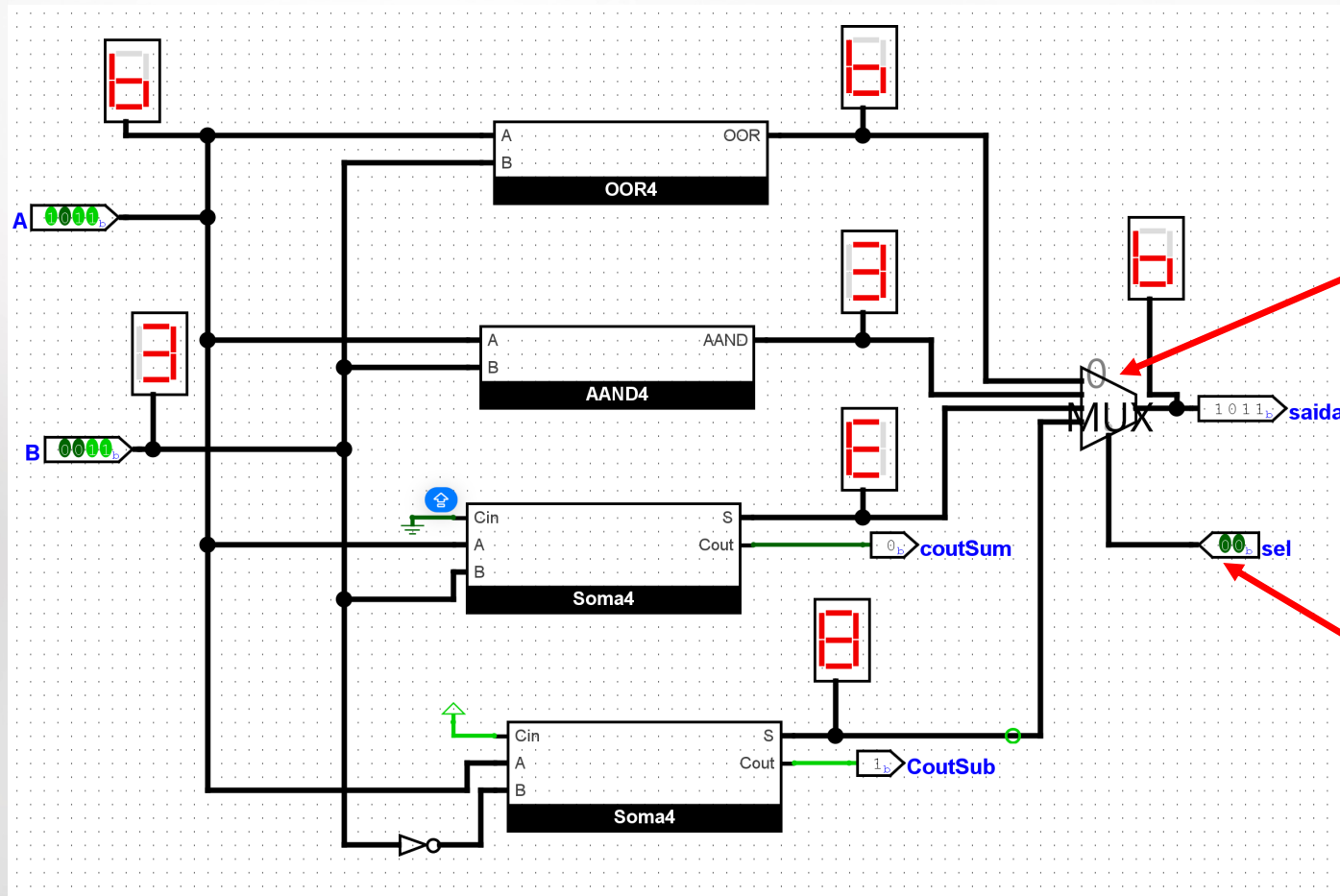
AND 4 bits 1000 and 0001 = 0000

soma 4 bits 1000 + 0001 + 0 = 1001 (9 e cout=0)

subtração 4 bits $A + \bar{B} + 1$

1000 + not(0001) + 1 = 1000 + 1110 + 1 = (1) 0111 → 7

Concluindo, com um multiplexador



Multiplexer (850,300)	
FPGA supported:	Supported
Facing	→ East
Gate Size	Wide
Select Location	Bottom/Left
Select Bits	2
Data Bits	4
Disabled Output	Zero
Include Enable?	No

Pin "sel"	
FPGA supported:	Supported
Facing	← West
Output?	No
Data Bits	2
Three-state?	No
Pull Behavior	Unchanged
Label	sel
Label Font	SansSerif Bold 16
Radix	Binary
Reset value:	0x0
Appearance	Arrow shapes