Universidade Federal do Rio Grande do Norte 

Centro de Tecnologia

Departamento de Engenharia de Computação e Automação

DCA0212.1 - Circuitos Digitais

Docente: Tiago Barros

Discentes: Carlos Eduardo Medeiros da Silva, Victor de Alcantara Gouveia, Marcantoni Leite de Sousa Melo e

Vilner César Oliveira.

Laboratório 1

**1. Introdução:**

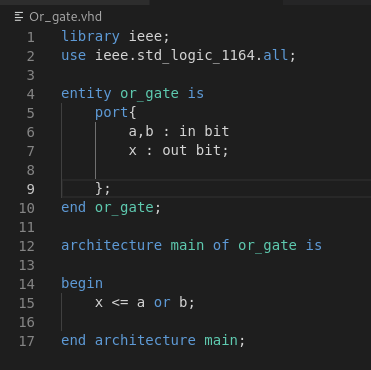
Este relatório abrange uma introdução aos princípios fundamentais dos circuitos digitais, abordando os conceitos e propriedades das portas lógicas AND, OR e NOT, bem como os fundamentos da álgebra booleana. A estrutura do documento segue a divisão clássica em metodologia, resultados e conclusão. Na metodologia, serão detalhados os métodos e procedimentos utilizados para explorar os conceitos dos circuitos digitais, incluindo tanto a análise teórica quanto a prática das portas lógicas e suas aplicações. Os resultados apresentarão os principais achados obtidos durante as simulações das propriedades das portas lógicas e da álgebra booleana, demonstrando sua utilidade e funcionamento por meio de simulações. Por fim, a conclusão reunirá as descobertas e insights obtidos, ressaltando a importância dos circuitos digitais na computação e eletrônica, além de sugerir possíveis direções para pesquisas futuras.

**2. Metodologia:**

Com o objetivo de compreender o funcionamento das portas lógicas e demonstrar todas as propriedades fundamentais da álgebra booleana, foram desenvolvidos códigos em VHDL para simular circuitos digitais.

**2.1 Porta OR:**

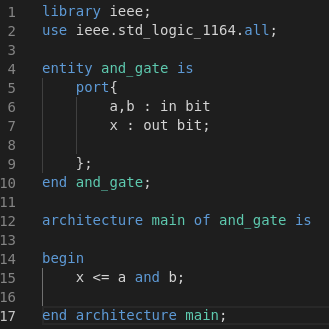
Para realização da simulação, foi usado o seguinte código em vhdl para uma porta OR com duas entradas.



No código é apresentada a existência de duas entradas a e b, que após aplicado operador lógico OR, resulta numa saída x.

**2.2 Porta AND:**

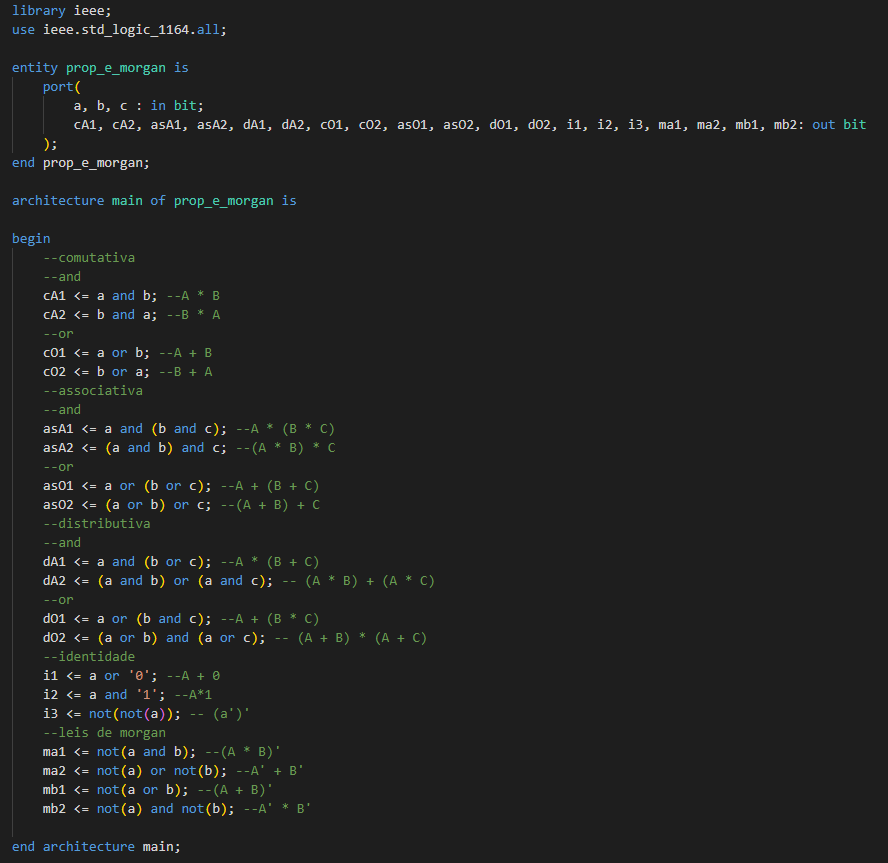
Para realização da simulação, foi usado o seguinte código em vhdl para uma porta AND com duas entradas.



Assim as entradas a e b retornam um valor de x com o operador lógico AND aplicado a elas.

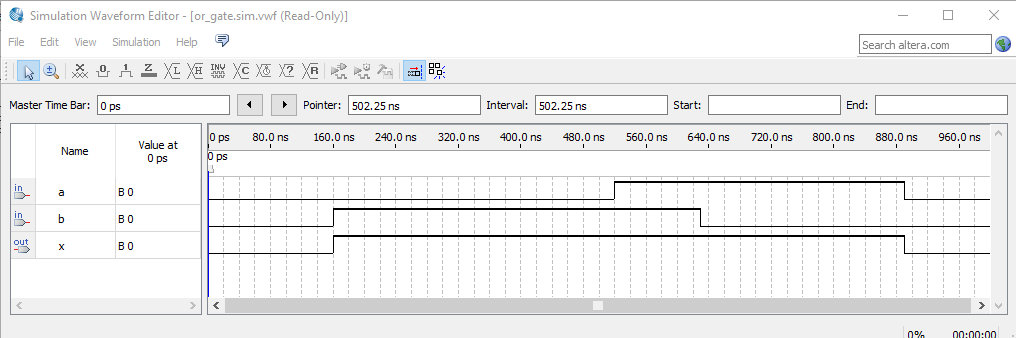
**2.3 Propriedades da Álgebra Booleana e Leis de Morgan:**

Para realização da simulação, foi usado o seguinte código em vhdl, onde existem três entradas nomeadas “a”, “b” e “c”, enquanto as saídas são organizadas da seguinte forma: as portas “asA1”, “asA2”, “asO1” e “asO2” são saídas referentes à propriedade associativa, “cA1”, “cA2”, “cO1” e “cO2” à propriedade comutativa e “dA1”, “dA2”, “dO1” e “dO2” à propriedade distributiva e as variáveis “i1”, “i2” e “i3” são referentes à identidade, de forma que as variáveis com mesmas letras maiusculas são comparáveis entre si, por exemplo, se seguir corretamente a propriedade distributiva, a variável “asA1” deve ser igual a “asA2” e assim por diante. As saídas “ma1”, “ma2”, “mb1” e “mb2” são referentes às Leis de Morgan.



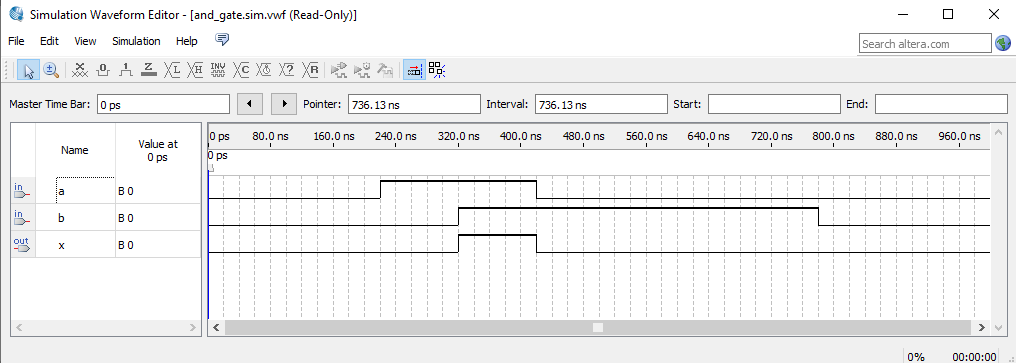
**3. Resultados:**

**3.1 Porta OR:**

****

A partir do experimento, foi observado que na porta OR no intervalo de 0 ms a 160 ms o valor de A e de B são 0 e o resultado é 0 também. Já no intervalo de 160 ms a 520ms o valor de A é 0 e o valor de B é 1, o que fez o valor do resultado ser 1 também. No intervalo de 520ms a 625 ms os valores de A e B são 1 e o resultado se manteve 1. No intervalo de 625 ms a 870 ms o valor de B foi 0 e o de A foi 1, e o resultado foi 1 também. Após 870 ms os valores tanto de A como de B foram 0, e o resultado ficou em 0 também. A partir dessa análise, é possível concluir que a porta OR, retorna 1 quando pelo menos uma das entradas for 1, e só resultará em 0 quando todas as entradas forem 0.

**3.2 Porta AND:**

****

No caso da porta AND, no intervalo de 0 ms a 240 ms os valores de A e B foram 0 e o resultado foi 0 também. No intervalo de 240 ms a 320 ms o valor de A foi 1, o de B foi 0 e o resultado foi 0 também. Já no intervalo de 320 ms a 420 ms, tanto o valor de A como o valor de B foi 1 e o resultado foi 1 também. Entre 420 ms e 780 ms o valor de A foi 0 e o de B foi 1, o que resultou em 0. A partir dessa análise, é possível concluir que a porta AND resultou em 1 apenas quando todas as entradas foram 1 e caso apenas uma entrada seja 1, o resultado será 0.

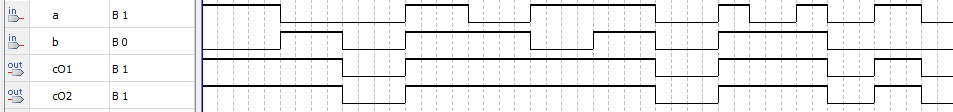
**3.3 Propriedades da Álgebra Booleana e Leis de Morgan:**

**3.3.1: Comutativa:**

Abaixo seguem os resultados obtidos ao testar a propriedade comutativa das operações.

**3.3.1.1 Adição:**

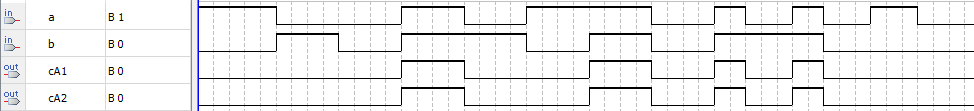
Para implementar essa propriedade foram criadas duas portas OR. A variável cO1 recebe a saída de A OR B e cO2 recebe o resultado de B OR A. Logo:



Para a=1 e b=0 vemos que ambos resultados cO1 e cO2 são iguais a 1. Portanto a propriedade A + B = B + A está provada.

**3.3.1.2 Multiplicação:**

Para provar essa propriedade, criamos duas portas AND contendo nas variáveis cA1 e cA2 os resultados correspondentes a A\*B e B\*A respectivamente.



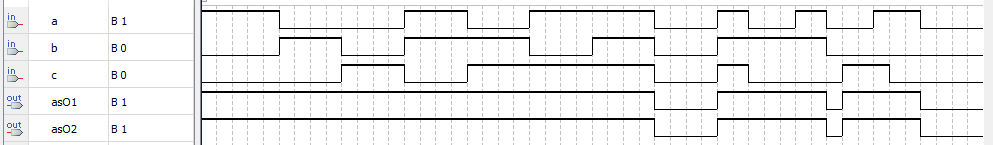
Na imagem acima, testamos as entradas de acordo com o esperado e podemos conferir que os resultados de cA1 e cA2 são iguais a 1 quando a=1 e b=1 de acordo o esperado da operação. Isso prova que A\*B = B\*A

**3.3.2: Associativa:**

Abaixo seguem os resultados obtidos ao testar a propriedade associativa das operações.

**3.3.2.1 Adição:**

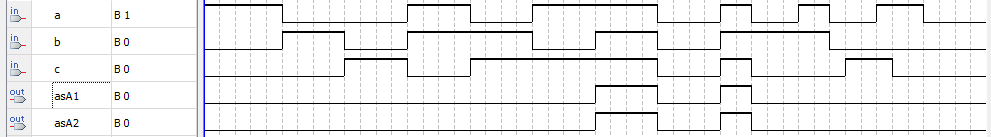
Para implementar essa propriedade foram criadas duas funções booleanas. A variável asO1 recebe a saída de A + (B + C) e asO2 recebe o resultado de (A + B) + C. Logo:



Observa-se então que asO1 é igual a asO2 e se comporta da forma esperada para ambas as funções booleanas, portanto a propriedade (A + B) + C = A + (B + C) está provada.

**3.3.2.2 Multiplicação:**

Para provar essa propriedade, criamos duas funções booleanas contendo nas variáveis asA1 e asA2 os resultados correspondentes a A \* (B \* C) e (A \* B) \* C respectivamente.



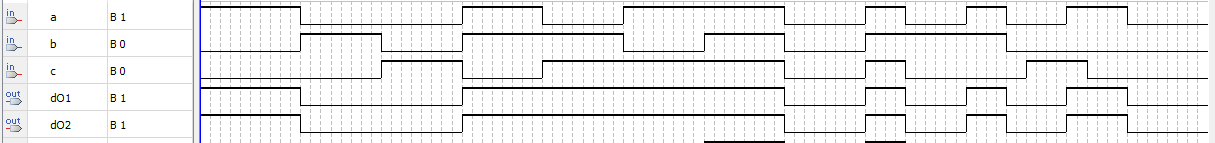
Na imagem acima, testamos as entradas de acordo com o comportamento esperado e podemos conferir que os resultados de asA1 e asA2 são iguais, provando que A \* (B \* C) = (A \* B) \* C

**3.3.3: Distributiva:**

Abaixo seguem os resultados obtidos ao testar a propriedade distributiva das operações.

**3.3.3.1 Adição:**

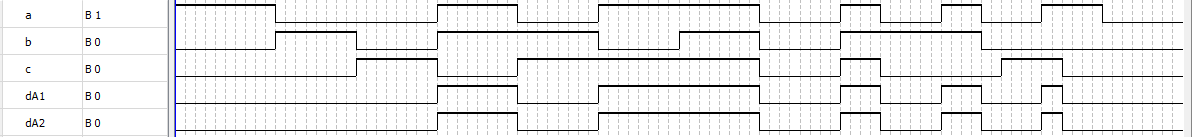
Para implementar essa propriedade foram criadas duas funções booleanas. A variável dO1 recebe a saída de A \* (B + C) e dO2 recebe o resultado de (A \* B) + (A \* C). Logo:



Observa-se então que asO1 é igual a asO2 e se comporta da forma esperada para ambas as funções booleanas, portanto a propriedade (A \* B) + (A \* C) = A \* (B + C) está provada.

**3.3.3.2 Multiplicação:**

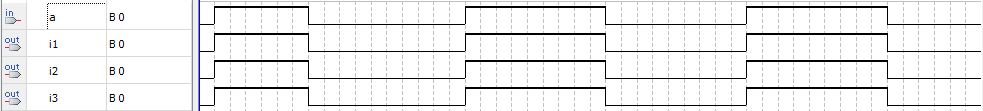
Para provar essa propriedade, criamos duas portas AND contendo nas variáveis asA1 e asA2 os resultados correspondentes a A + (B \* C) e (A + B) \* (A + C) respectivamente.



Na imagem acima, testamos as entradas de acordo com o comportamento esperado e podemos conferir que os resultados de dA1 e dA2 são iguais, provando que A + (B \* C) = (A + B) \* (A + C).

**3.3.4: Identidade:**

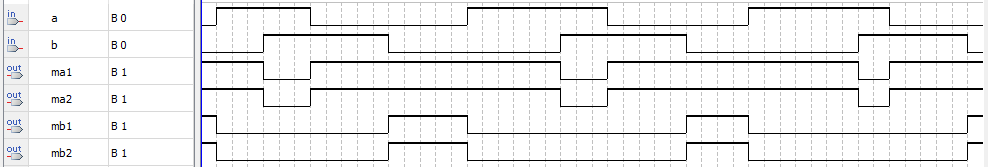
Para provar essa propriedade, bastou usar a entrada “a” como base paras as saídas i1, i2 e i3 que são, respectivamente, A OR 0, A AND 1 e (A’)’.



Como todas as saídas foram iguais à entrada, comprova-se que a propriedade foi implementada com sucesso.

**3.3.5: Leis de Morgan:**

Para provar as Leis de Morgan, foram usadas as entradas a e b para realizar as operações, representadas nas saídas ma1 e ma2 para a primeira lei, onde esta é a operação A’ + B’, e aquela, (A \* B)’, e mb1 e mb2 para a segunda lei, sendo elas, respectivamente, (A + B)’ e A’ \* B’, resultando no seguinte:



Percebe-se, então, que ma1 possui o mesmo resultado que ma2, logo, (A \* B)’ = A’ + B’, comprovando a primeira lei, enquanto mb1 é igual a mb2, provando a segunda lei onde (A + B)’ = A’ \* B’

**4. Conclusão:**

Dessa forma, é possível concluir que os estudos teóricos realizados em laboratório foram devidamente aplicados durante esse experimento, além de ter sido desenvolvido com sucesso um circuito que simula a aplicação das propriedades das operações booleanas e das Leis de Morgan, comprovando seu funcionamento.