**UNIVERSIDADE DE BRASÍLIA**

Adrianne Alves da Silva - 160047595
Ana Paula Gomes de Matos - 160023629

**TURMA "F"**
**PRÁTICA DE ELETRÔNICA DIGITAL - 119466**
Experimento 01: Portas Lógicas

Brasília - DF
31 de março de 2017

UNIVERSIDADE DE BRASÍLIA

# PRÁTICA DE ELETRÔNICA DIGITAL - 119466

Relatório referente ao primeiro experimento da disciplina Prática de Eletrônica Digital, ministrada pela professora Lourdes Mattos Brasil, realizado no dia 24 de março de 2017.

**Alunas:**

Adrianne Alves da Silva

Ana Paula Gomes de Matos

Brasília – DF

**SUMÁRIO**

# 1 INTRODUÇÃO

## 1.1 Portas lógicas

Em meados do século XIX, o inglês George Boole desenvolveu o primeiro sistema para raciocínio lógico, ele acreditava na possibilidade de utilizar o cálculo na resolução de problemas em diferentes situações. A álgebra booleana tornou possível o avanço da tecnologia computacional até a velocidade da eletrônica, de forma menos complexa. [1]

No início da eletrônica, todos os problemas eram solucionados por intermédio de sistemas analógicos, contudo, o avanço tecnológico tornou possível a resolução por meio da eletrônica digital. Nesta, os sistemas agregam um conjunto de circuitos eletrônicos básicos, conhecidos como portas lógicas, as quais possuem uma ou mais entradas e saída única. A estes pinos são atribuídos valores de tensão, associados às variáveis booleanas "**0**" e "**1**", equivalentes a uma potência de **0 volts** e **5 volts,** respectivamente. [2]

A porta lógica **NOT,** ou inversora**,** é considerada a porta mais simples, pois possui apenas uma entrada e sua saída é o complemento desta, conforme a figura abaixo.

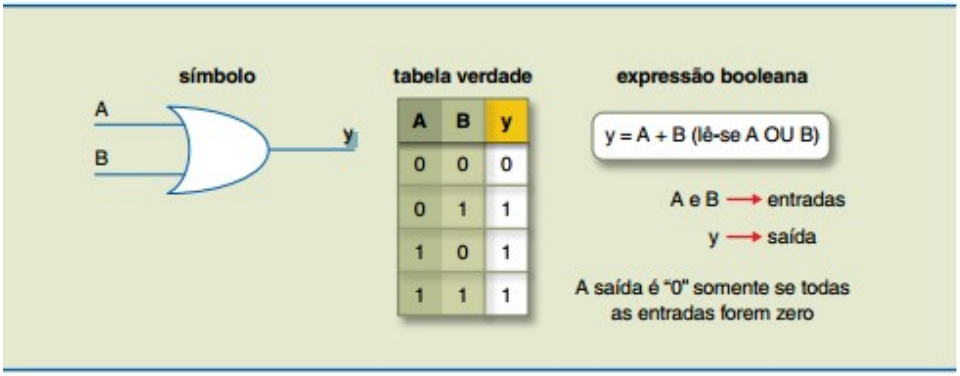
**Figura 01:** Porta NOT


**Fonte:** www.albertoferes.com.br

Conforme apresenta a **figura 01**, a cada porta lógica está associado um símbolo, que a representa em um circuito, uma expressão booleana que descreve a atividade desta, cujos valores assumidos restringem-se a **0** e **1** , e por fim, uma tabela verdade onde constam todas as possibilidades de entradas e saídas. O conjunto desses dados fornece um completo domínio para a montagem de circuitos lógicos e simulação em softwares apropriados [3]. Com base nesses conceitos, todas as outras portas lógicas podem ser caracterizadas.

A porta **OR**, por sua vez, pode possuir duas ou mais entradas e sua saída sempre apresentará o valor "1" se pelo menos umas das entradas for igual a "1", em contrapartida, se todas as entradas forem iguais a "0", sua saída também será "0", conforme a tabela verdade demonstrada na figura abaixo. Salienta-se que o resultado da saída é associado à operação de soma entre as entradas, entretanto, não se comporta como a soma aritmética.[2]

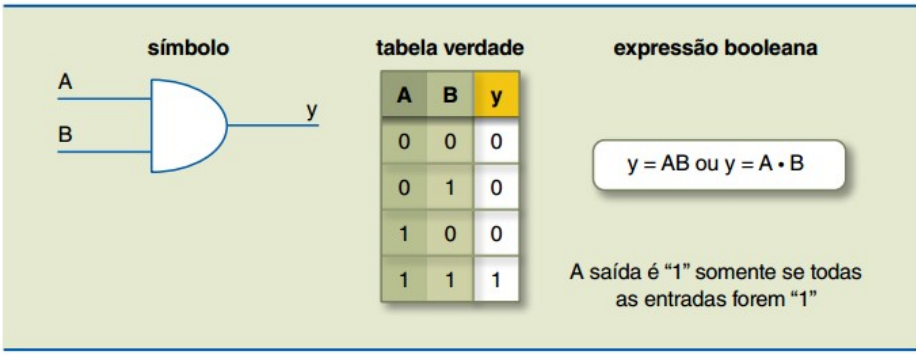
**Figura 02:** Porta OR




**Fonte:** www.albertoferes.com.br

Abaixo, a porta lógica denominada **AND**, também pode possuir duas ou mais entradas, entretanto, a sua saída só será "1" quando todas as entradas também forem iguais a "1" e em decorrência disto ela é associada à operação de multiplicação.

**Figura 03:** Porta AND

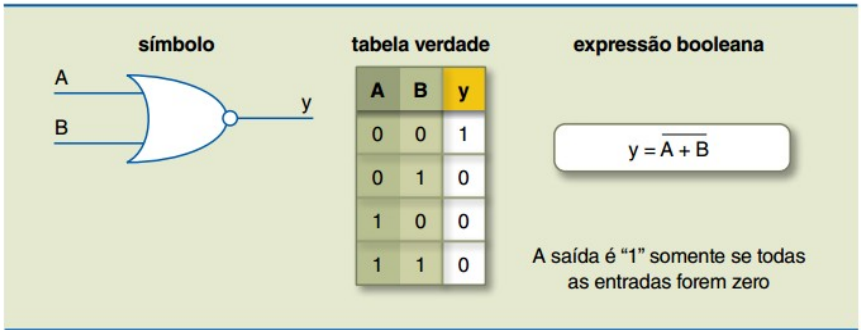



**Fonte:** www.albertoferes.com.br

As três portas que foram apresentadas anteriormente são as principais dentro da eletrônica, a combinação ou derivação delas dá origem a outras que são tradicionalmente usadas em diferentes projetos, e por esse motivo, recebem nomes próprios.

A combinação das portas NOT e OR dá origem a **NOR**, cuja saída será "1" somente se todas as entradas forem iguais a "0", ou seja, funciona de maneira inversa à tabela verdade da porta OR, e essa característica é expressa  no símbolo por meio do inversor.

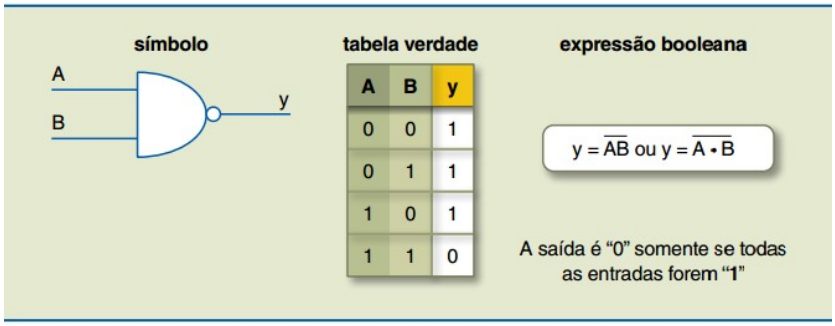
**Figura 04:** Porta NOR




**Fonte:** www.albertoferes.com.br

A combinação das portas NOT e AND dá origem a **NAND**, cuja saída será “0” apenas se todas as entradas forem iguais a “1”, como expresso por meio da tabela verdade da figura abaixo.

**Figura 05:** Porta NAND




**Fonte:** www.albertoferes.com.br

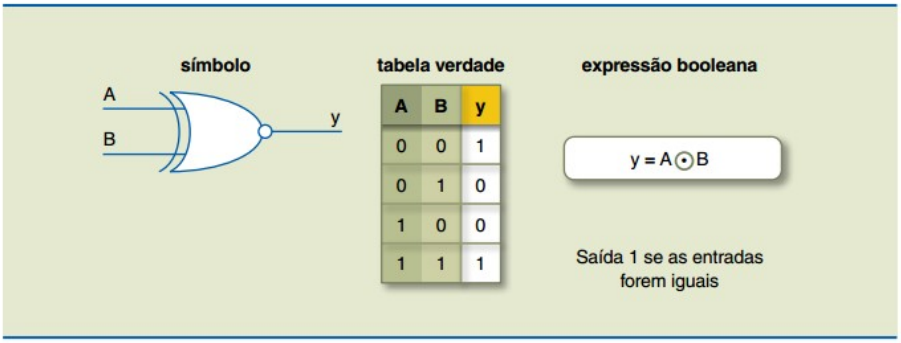
Por sua vez, a porta **XOR** pode apresentar uma ou mais entradas, mas a sua saída só será “1” quando suas entradas forem diferentes.

**Figura 06:** Porta XOR


**Fonte:** www.albertoferes.com.br

A porta lógica XOR com saída investida recebe o nome de **XNOR**, assim, sua saída só será “1” se todas as suas entradas forem iguais.

**Figura 07:** Porta XNOR




**Fonte:** www.albertoferes.com.br

## 1.2 VHDL

O VHDL é a linguagem adotada pelo IEEE (*Institute of Electrical and Electronics Engineers*) e foi estabelecida como padrão 1076-1993. Por ser uma linguagem complexa e compreensiva, o uso de seu potencial requer muito esforço e prática.

Ele prevê três tipos de abordagens básicas na descrição de um código: procedimento, fluxo de dados e estrutura.

Os operadores lógicos em VHDL são: AND, OR, NOT, NAND, NOR, XOR e XNOR, os quais já foram descritos anteriormente. Os principais elementos dessa linguagem são a identidade e a arquitetura, a primeira faz referência à descrição da função lógica em termos de suas portas e a segunda descreve a operação interna da função. O elemento entidade é composto por três declarações: a primeira faz a associação de um nome à função lógica, a segunda especifica as entradas e saídas, a última é a declaração End. [4]

A declaração de em VHDL de identidade para uma porta AND com duas entradas é a seguinte:

```
entity AND_GATE2 is
        port (A, B: in bit; X: out bit);
end entity AND_GATE2;
```

O VHDL reserva as palavras em negrito, os demais termos são identificadores associados pelo usuário, os parênteses, vírgulas e ponto-e-vírgulas compõem a sintaxe da linguagem. [3]

A arquitetura do código em VHDL para a porta AND com duas entradas é descrito pela seguinte identidade:

```
architecture LogicFunction of AND_Gate2 is
begin
X ⇐ A and B;
end architecture LogicFunction;
```

Novamente, as palavras em negrito são reservadas pela linguagem, a pontuação faz parte da síntese necessária, vale ressaltar que a primeira declaração do elemento em uma estrutura tem que ser diferente do nome da identidade. Em seguida, a entidade e a arquitetura são combinadas em um único comando VHDL para então descrever a porta lógica desejada no caso, a porta AND. [4]

### 1.3 Objetivo

Proporcionar ao aluno a compreensão do funcionamento das portas lógicas por meio de simulações no *software* Vivado e a familiarização do mesmo com o ambiente laboratorial.

### 1.4 Justificativa

Compreender o que é uma porta lógica e suas aplicações dentro da eletrônica é de extrema importância no atual contexto tecnológico, da mesma forma, entender o funcionamento de *software* voltados à esse tipo de abordagem e saber realizar simulações é também um diferencial no mercado de trabalho. O experimento realizado no dia 24 de março de 2017, no laboratório de Prática de Eletrônica Digital, conseguiu unir a relevância das portas lógicas para a eletrônica e a atual tendência de mercado, o que possibilita maior preparação para o estudante e melhor fixação da teoria . Sendo assim, é de suma importância o registro de todo o processo, o que torna crucial a produção de relatórios para que o professor avalie se de fato o aluno entendeu o objetivo do experimento, assim como possibilitar a comparação dos resultados em simulações realizadas em diferentes programas computacionais.

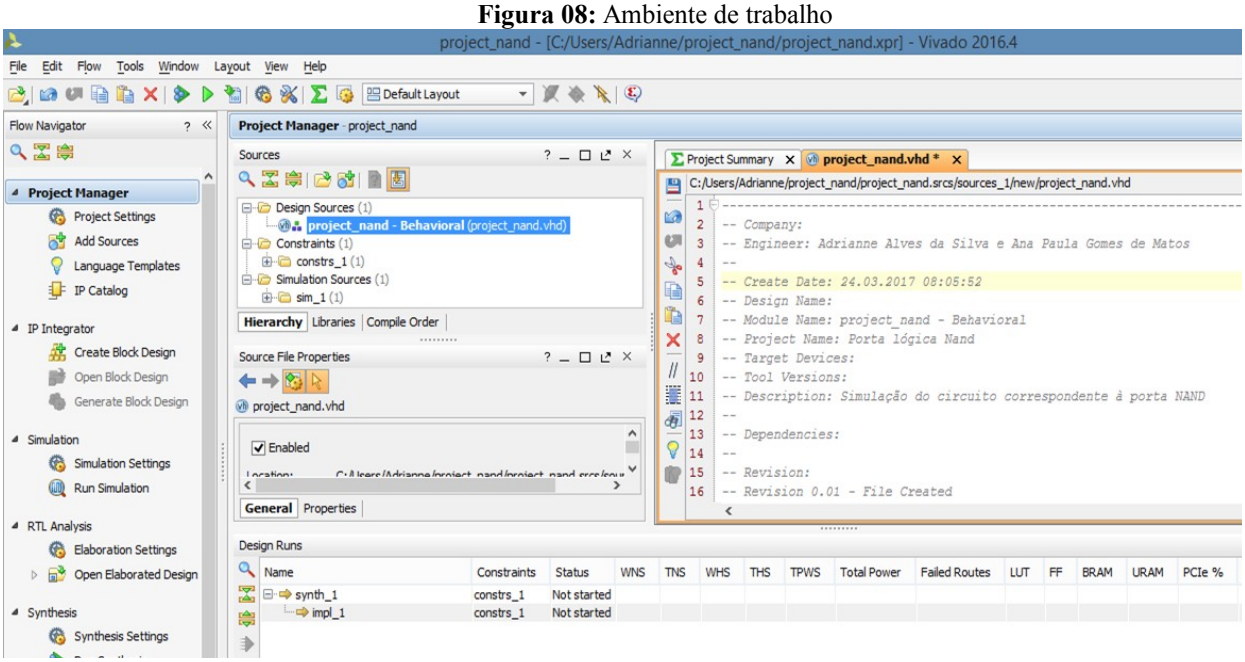
# 2 PARTE EXPERIMENTAL

## 2.1 Preparação do ambiente de simulação

O passo inicial para a realização do experimento foi criar um novo projeto no *software* Vivado. A configuração do mesmo foi realizada adicionando a placa Basys 3 ao projeto, por meio da aba "*add or create constraints*". Após esses passos foi necessário apenas criar o arquivo relativo ao *design* , adicionando o arquivo com o tipo vhdl, para que o código pudesse ser implementado nesta linguagem. O módulo foi então definido de acordo com o número de portas de entrada e saída do circuito referente à cada atividade solicitada, nomeadas as entradas como A,B e C e Saída com S.

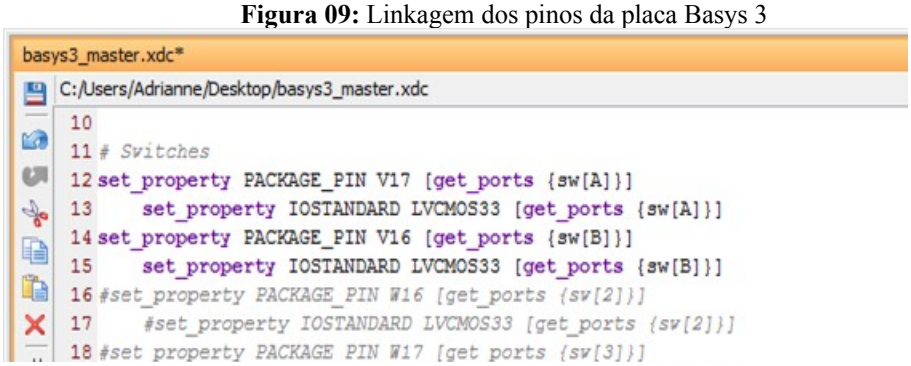
É necessário então ativar na aba de *project settings* , *bitstream* o arquivo bin_file que possibilita escrita binária sem uso do cabeçalho. Assim, obteve-se o ambiente demonstrado na figura a seguir.

**Figura 08:** Ambiente de trabalho



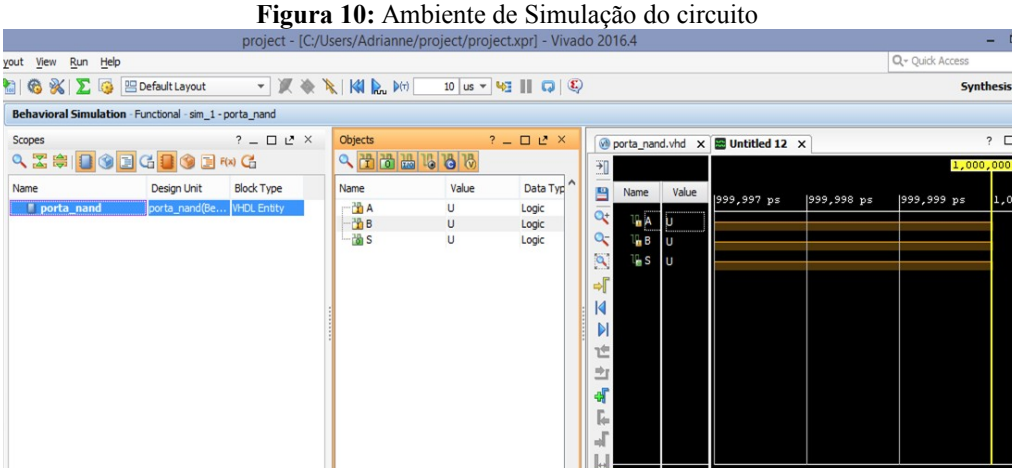

Sendo assim, foi possível abrir o arquivo vhdl definido e implementá-lo, identificando com os nomes das alunas responsáveis, data e horário de desenvolvimento. Para cada atividade, forneceu-se ao arquivo o código e a expressão booleana adequada. Por sequência do passo-a-passo fornecido, foram linkados os pinos de entrada e saída da placa Basys com os dispositivos de entrada e saída implementadas. No arquivo basys3.xdc (**figura 9**), isso foi feito por meio da retirada do comentário das linhas referentes ao número de entradas e saídas presentes em cada atividade, além da adequada atribuição de cada uma das portas, segundo a nomenclatura adotada (A,B,C,S), essa atividade seria essencial para a verificação diretamente numa FPGA.

**Figura 09:** Linkagem dos pinos da placa Basys 3



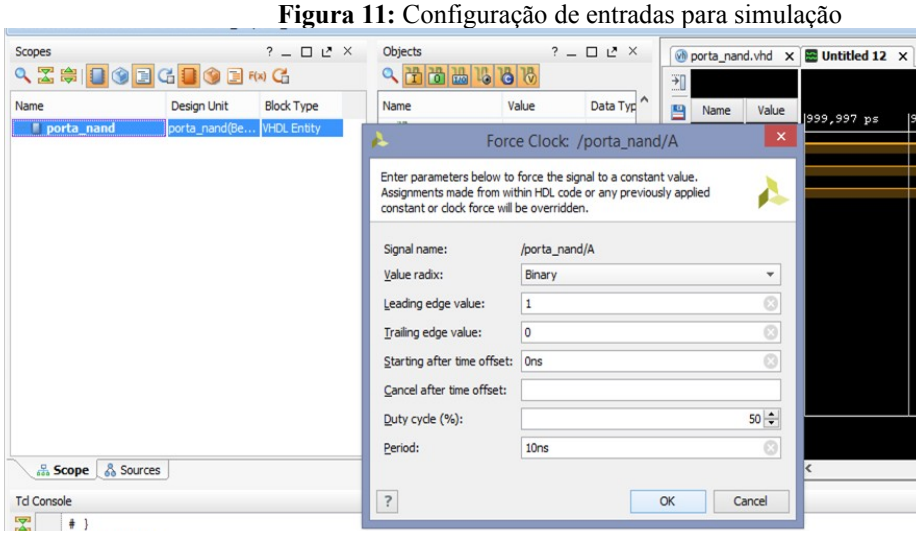

Após todos esses passos foi necessário sintetizar o circuito por meio do *Run Synthesis* , gerando um esquemático facilmente visualizado no *RTL Analysis* . Nesse ponto, é que tornou-se possível realizar a simulação do experimento, bastando clicar em *run simulation* que gera um ambiente de simulação (**figura 10**).

**Figura 10:** Ambiente de Simulação do circuito




Nesse passo, é possível configurar os valores referentes à cada entrada, assim como o tempo de mudança de sinal ( 0 ou 1) para a geração de ondas . Isso é feito clicando sobre cada entrada com o botão direito, na opção de *Force Clock*, conforme a figura a seguir.

**Figura 11:** Configuração de entradas para simulação



A primeira configuração realizada é a modificação do argumento *value* para o tipo binário informando entradas diferenciadas para os argumento *Leading edge value* e *Trailing edge value*, que seja 0 ou 1 . Define-se um período para as ondas, diferentes para cada entrada. É legal se o valor de uma for o dobro da outra, por exemplo. E então, basta executar a simulação.

### 2.2 Experimento

O experimento consistiu na simulação de diferentes portas lógicas utilizando o software Vivado, por meio do qual, através da implementação de código em VHDL, é possível construir e testar o funcionamento de circuitos. A primeira porta trabalhada foi a **NAND** , à qual fora solicitada na **atividade 1** e **atividade 3** do experimento.

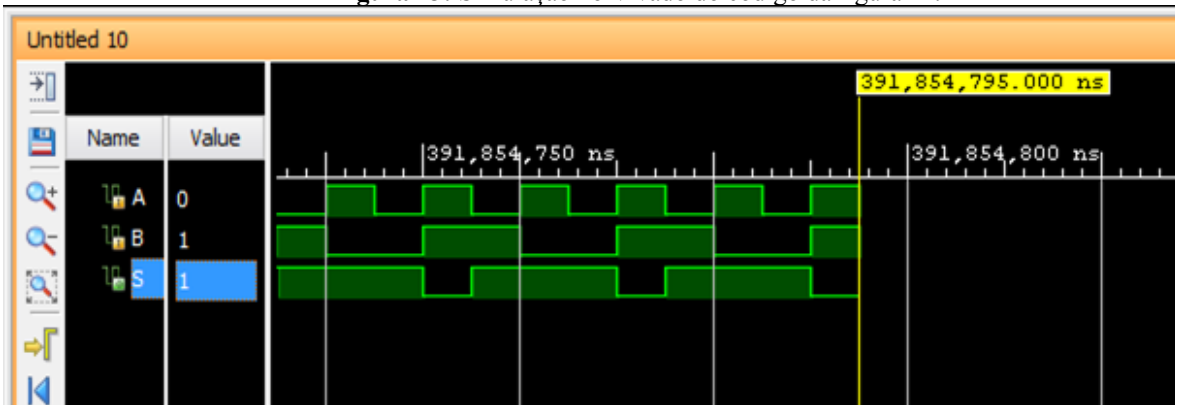
Optou-se por implementar a porta NAND de duas diferentes formas, sendo a primeira delas utilizando a função pronta do VHDL, segundo a figura a seguir .

**Figura 12:** Código em VHDL da porta lógica NAND com função pronta

```
-- Declaração das bibliotecas necessárias

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

--Descrição dos pinos de entrada e saída
-- Tipo bit para os pinos pois trabalharemos nesse caso apenas com entradas do tipo 0 e 1

entity project_nand is
    Port ( A : in BIT;
           B : in BIT;
           S : out BIT);
end project_nand;

-- Definição da arquitetura e/ou funcionamento da porta NAND

architecture Behavioral of project_nand is

begin
    S <= A NAND B; -- Expressão lógica que caracteriza a porta NAND
end Behavioral;
```

Antes que a simulação fosse realizada, foi feita a síntese do programa. Nesta etapa, a compilação foi mais lenta que o previsto e o código apresentou alguns erros desconhecidos, sendo necessário muitas vezes, fazer a síntese novamente, entretanto, foram devidamente corrigidos com ajustes na configuração por meio do auxílio do monitor da disciplina. Após esse processo, foi criado o diagrama esquemático do circuito (verificar no final deste documento) e a simulação, com o intuito de identificar se a lógica estava correta conforme a porta lógica solicitada. A figura 13 representa a simulação do código apresentado na figura 12.

**Figura 13:** Simulação no Vivado do código da figura 11.



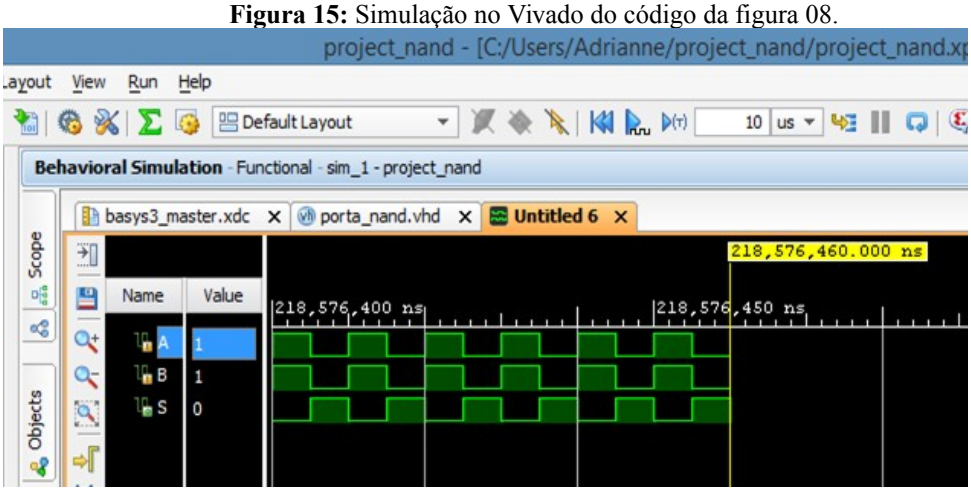

Se observarmos a tabela da verdade, referente à essa porta lógica (**figura 5**), vemos que o formato de ondas obedece fielmente a relação proposta, e o mesmo acontece se o código for implementado segundo uma expressão booleana própria (**figura 14**).

**Figura 14:** Código em VHDL, implementado pela dupla, da porta lógica NAND

```
20
21  -- Declaração das bibliotecas necessárias
22
23  library IEEE;
24  use IEEE.STD_LOGIC_1164.ALL;
25
26  --Descrição dos pinos de entrada e saída
27  -- Tipo bit para os pinos pois trabalharemos nesse caso apenas com entradas do tipo 0 e 1
28
29  entity project_nand is
30      Port ( A : in bit;
31             B : in bit;
32             S : out bit);
33  end project_nand;
34
35  -- Definição da arquitetura e/ou funcionamento da porta NAND
36
37  architecture Behavioral of project_nand is
38
39  begin
40      S<= not(A and B); -- Expressão lógica que caracteriza a porta NAND
41  end Behavioral;
42
```

A onda obtida através desta última implementação funciona perfeitamente, conforme a tabela verdade (**figura 15**).

**Figura 15:** Simulação no Vivado do código da figura 08.



A segunda implementação realizada foi sobre a porta lógica **XNOR** , cujo código VHDL, conforme **atividade 3** solicitada, foi o seguinte :

**Figura 16:** Código em VHDL da porta lógica XNOR.

```
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25  -- Uncomment the following library declaration if using
26  -- arithmetic functions with Signed or Unsigned values
27  --use IEEE.NUMERIC_STD.ALL;
28
29  -- Uncomment the following library declaration if instantiating
30  -- any Xilinx leaf cells in this code.
31  --library UNISIM;
32  --use UNISIM.VComponents.all;
33
34  entity porta_xnor is
35      Port ( A : in BIT;
36             B : in BIT;
37             S : out BIT);
38  end porta_xnor;
39
40  architecture Behavioral of porta_xnor is
41  begin
42      S <= A XNOR B;
43  end Behavioral;
```

Após a sintetização e checagem do esquema gerado ( ver no fim do arquivo), obteve-se uma onda coerente com a simulação.

**Figura 17:** Simulação no Vivado do código da figura 16.


Como solicitado na atividade 2 (**figura 18**) , observa-se que de fato as ondas estão corretas.

**Figura 18:** Tabela verdade da porta XNOR

| A | B | S |
|---|---|---|
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

A última etapa do experimento foi a execução da **atividade 4**, que consiste na projeção de um circuito que realiza a função AND com três variáveis, isto é, no Vivado e em um simulador diferente, para que seja possível realizar uma comparação entre eles, comparando a facilidade de uso, a flexibilidade de apresentação de resultados e a possibilidade de implementação utilizando o VHDL.

A implementação deste circuito seguiu a lógica a seguir :

**Figura 19:** Código em VHDL da porta lógica AND com três entradas

```
22  library IEEE;
23  use IEEE.STD_LOGIC_1164.ALL;
24
25
26  entity porta_and is
27      Port ( A : in BIT;
28             B : in BIT;
29             C : in BIT;
30             S : out BIT);
31  end porta_and;
32
33  architecture Behavioral of porta_and is
34
35  begin
36  S<= A AND B AND C;
37
38  end Behavioral;
39
```

Tendo sido o processo de síntese tranquilo, a simulação obtida foi também tranquila, apresentando as ondas presentes na **figura 20**, que pode ser conferida com a tabela verdade presente na **figura 21.**

**Figura 20:** Simulação no Vivado do código da figura 18.

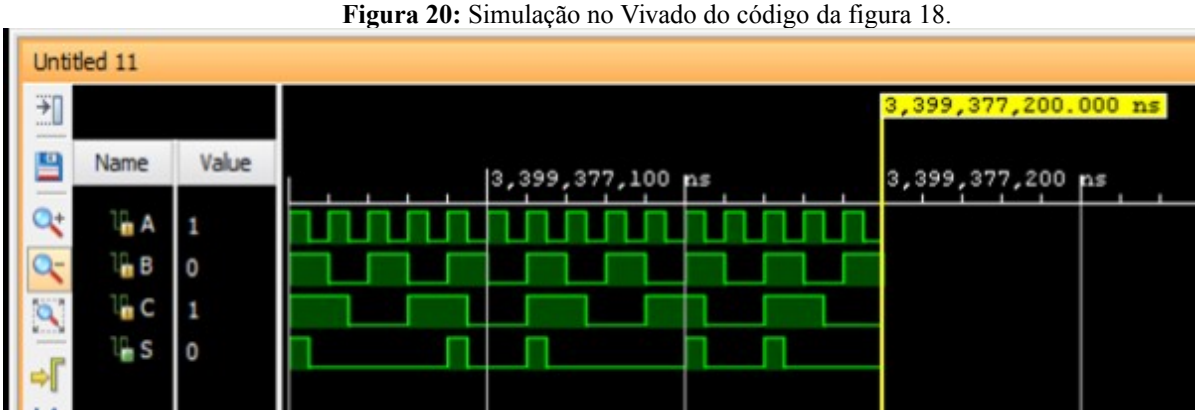



**Figura 21:** Tabela verdade da porta AND.

| A | B | C | S |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

Esse mesmo circuito foi implementado ainda em outro simulador, para que os resultados e a experiência de uso fossem comparados. O programa utilizado para comparação foi **Circuit Maker** o qual apresenta maior facilidade de uso quando se trata de circuitos pequenos, porém não possibilita a utilização de VHDL nem a visualização da tabela verdade. Dessa forma, todo o projeto é confeccionado de forma direta, ou seja, sem a implementação de um código, mas sim adição de componentes.

As figuras abaixo, referentes à experiência com o Circuit Maker, exibem a porta AND. A primeira imagem (**figura 22**), por exemplo, com entradas “1”, “1” e “0”, e saída “0”, apresenta o LED desligado, entretanto, na imagem seguinte (**figura 23**), com entradas “1”, “1” e “1”, ou seja, todas as entradas possuem nível de tensão alto, sua saída também apresenta nível alto, isto explica o LED aceso.

**Figura 22:** Simulação da porta AND com três pinos no Circuit Maker (LED apagado)

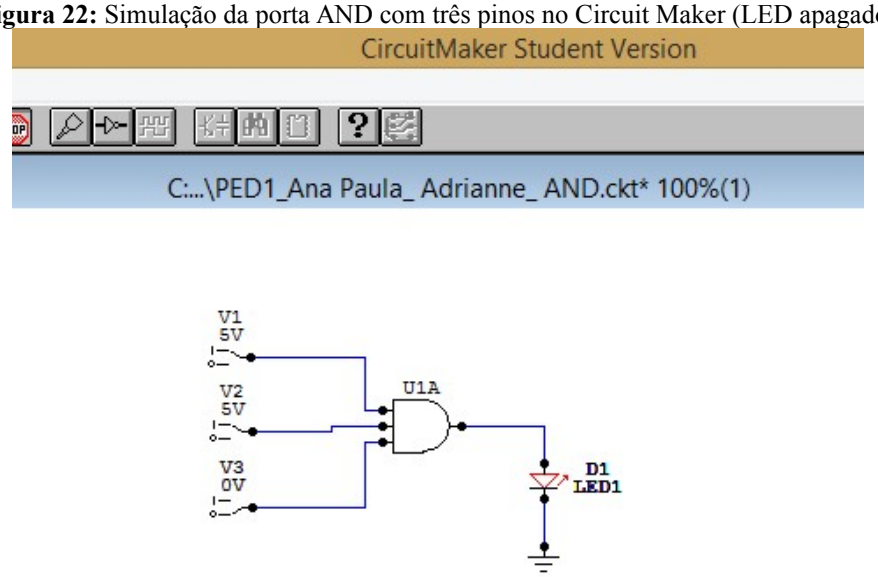
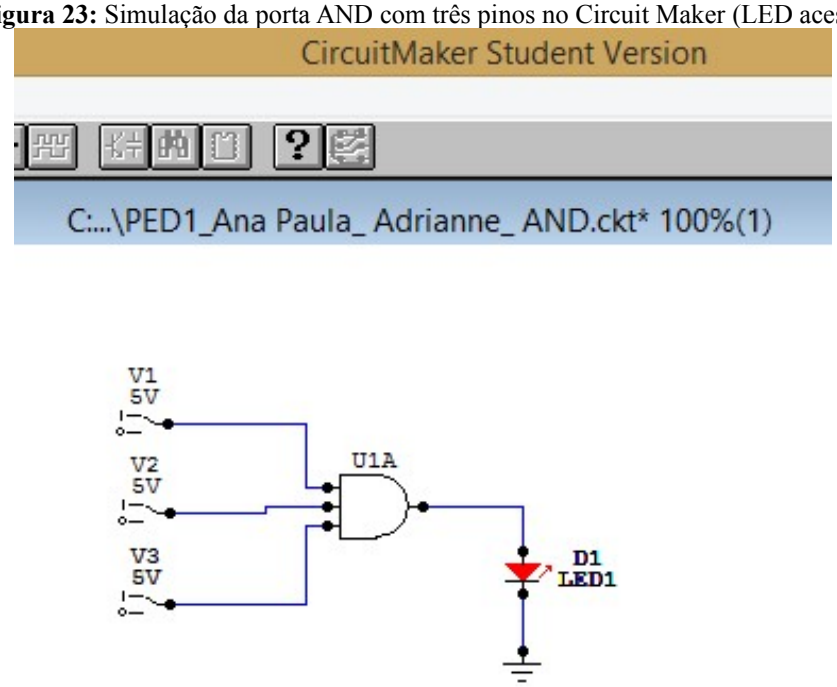



**Figura 23:** Simulação da porta AND com três pinos no Circuit Maker (LED aceso)



## 3 DISCUSSÃO

A simulação das portas lógicas, da maneira como foi proposta, envolve todo o conceito interligado à lógica booleana e expressões lógicas relacionadas. O raio dessa discussão, vai além do senso comum, e introduz a utilização de técnicas nas quais a lógica se estabeleça.

Mesmo portas lógicas simples como a AND da qual a NAND deriva, possui um parâmetro de fundamentação.

Na proposta da atividade 1 a implementação da porta lógica NAND de duas diferentes formas nos dá a certeza de que a utilização das técnicas relacionadas à lógica booleana é exata. Não há dois resultados diferentes para a mesma relação, de forma binária, é verdadeiro ou falso. Sendo a NAND uma espécie de complemento da porta AND, seu funcionamento é semelhante, baseado apenas numa inversão da saída esperada.

No primeiro experimento, utilizou-se, com finalidade de observação, os valores 0 para a entrada A e 1 para a entrada B, para determinar o início da onda, da qual se obteve, estabelecendo o período de A como 10 ns e o período em B como o dobro deste , uma gama variada de combinações que deixam mais explícitas as relações estabelecidas na tabela verdade. Se compararmos esta porta com a porta XNOR, por exemplo, notamos que esta só assume saída ‘1’ se as entradas forem iguais, enquanto a NAND só possui entrada 1 se ambas forem nulas, o que as dá tabela verdade e expressão lógica diferenciadas.

Na simulação da porta lógica XNOR foram usados 1 para a entrada A e 1 para entrada B, da mesma forma que o experimento anterior, foi usado A como 10 ns e B como 20 ns, os valores de saída de uma porta XNOR sempre assumirá o valor 1 se suas entradas forem iguais, consequentemente, os valores obtidos em sua tabela verdade não têm relação com os valores obtidos nas tabelas das portas NAND e AND.

No experimento seguinte, utilizou-se como entradas os valores 1 para A, 0 para B e 1 para C, isto para simular a porta AND que é corriqueiramente associada à operação de multiplicação (não que ambas realmente tenham alguma relação funcional) e dá origem à porta do primeiro experimento. Os valores obtidos na simulação de ondas mostram claramente que todas as entradas que continham o valor 0 obtiveram saídas iguais a zero, o que respeita a estrutura da tabela verdade da porta AND.

Por fim, a simulação da porta AND com três entradas foi realizada também no *software* Circuit Maker o qual apresenta uma configuração menos complexa que o Vivado e é consideravelmente mais intuitivo, porém também é bem mais limitado, pois não permite a obtenção direta da tabela verdade e nem simulações de ondas como faz o outro programa usado para a realização dos experimentos.

## 4 CONCLUSÕES

Diante do que foi apresentado, nota-se que todas as atividades solicitadas no experimento foram executadas com resultados satisfatórios, ou seja, foi possível simular as portas NAND, XNOR e AND no software Vivado, logo, o objetivo do trabalho foi alcançado, contudo, sob muitas limitações.

Como exemplo, pode-se citar o mau funcionamento de alguns computadores do laboratório, a demora na compilação dos códigos, etapas importantes do experimento que não existem no roteiro passo a passo e um ambiente virtual pouco intuitivo.

O ideal seria turmas menores para melhor aproveitamento do professor, orientação básica a respeito do que é solicitado nas atividades e, principalmente, uma excelente preparação dos alunos a respeito do assunto.

## REFERÊNCIAS BIBLIOGRÁFICAS

[1] FONSECA FILHO, Cléuzio. **História da computação: O Caminho do Pensamento e da Tecnologia**. EDIPUCRS, 2007. p. 56-58.

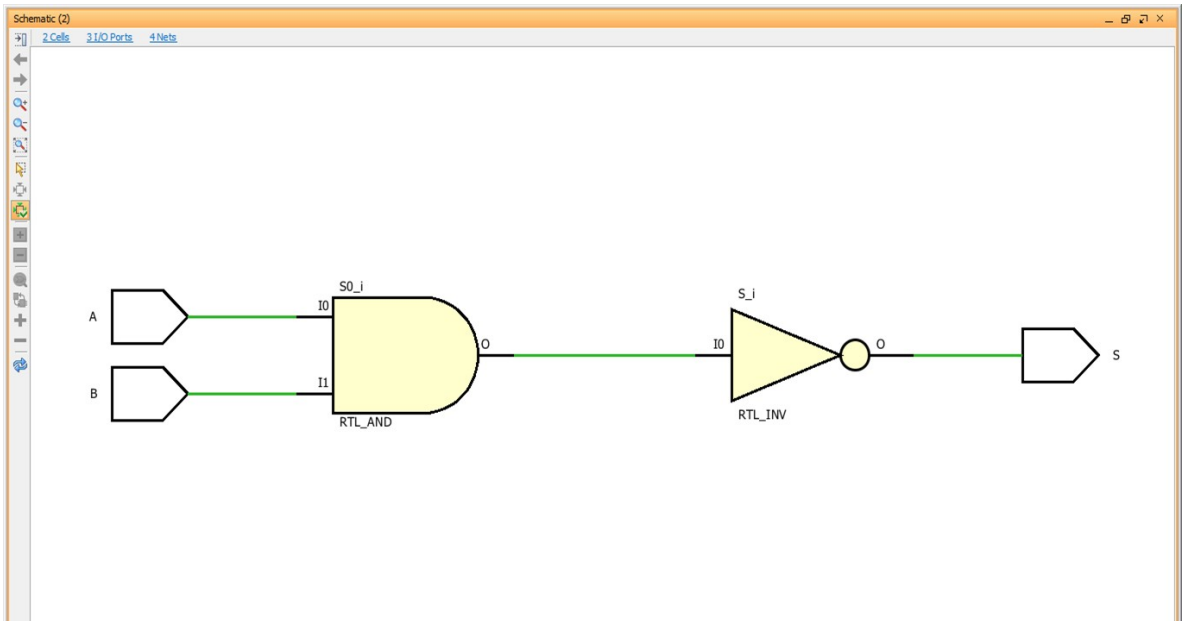
[2] AMARAL, Valder Moreira. **Eletrônica Digital**. São Paulo: Fundação Padre Anchieta, 2011. p. 31-33. 4 v.

[3] FLOYD, Thomas. Portas lógicas. In _____. **Sistemas digitais: fundamentos e aplicações**. Bookman Editora, 2009. Cap. 3, p. 134-135.

[4] _____. Álgebra booleana e simplificação lógica. In _____. **Sistemas digitais: fundamentos e aplicações**. Bookman Editora, 2009. Cap. 4, p. 244-245.

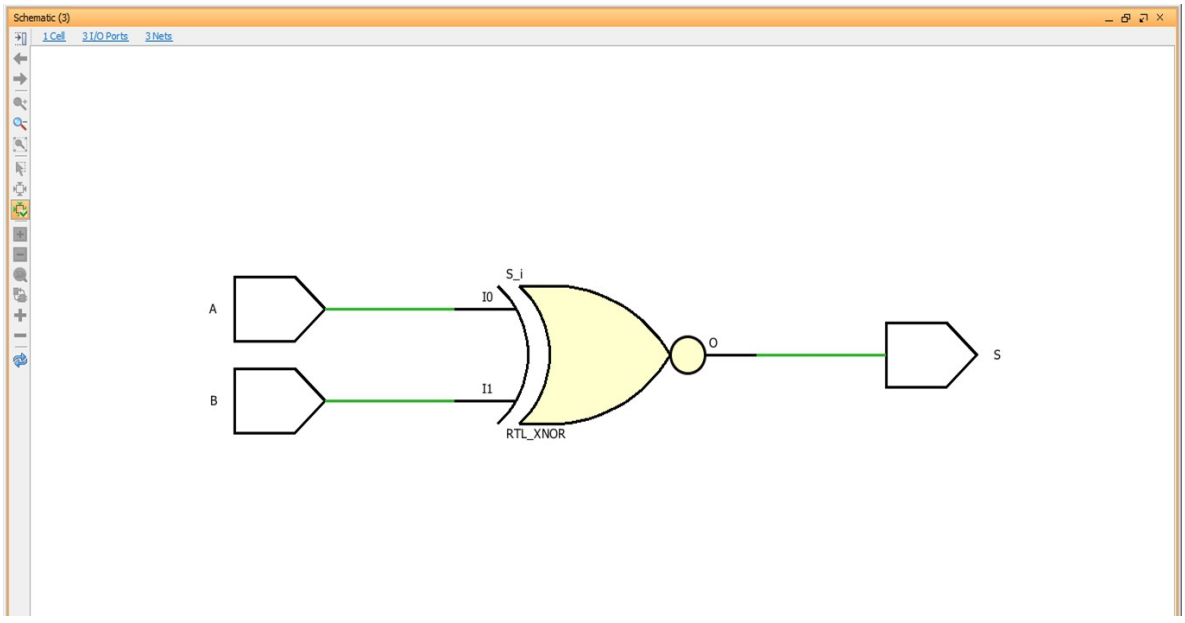
# DIAGRAMAS ESQUEMÁTICOS

**Figura 24:** Diagrama esquemático da porta NAND



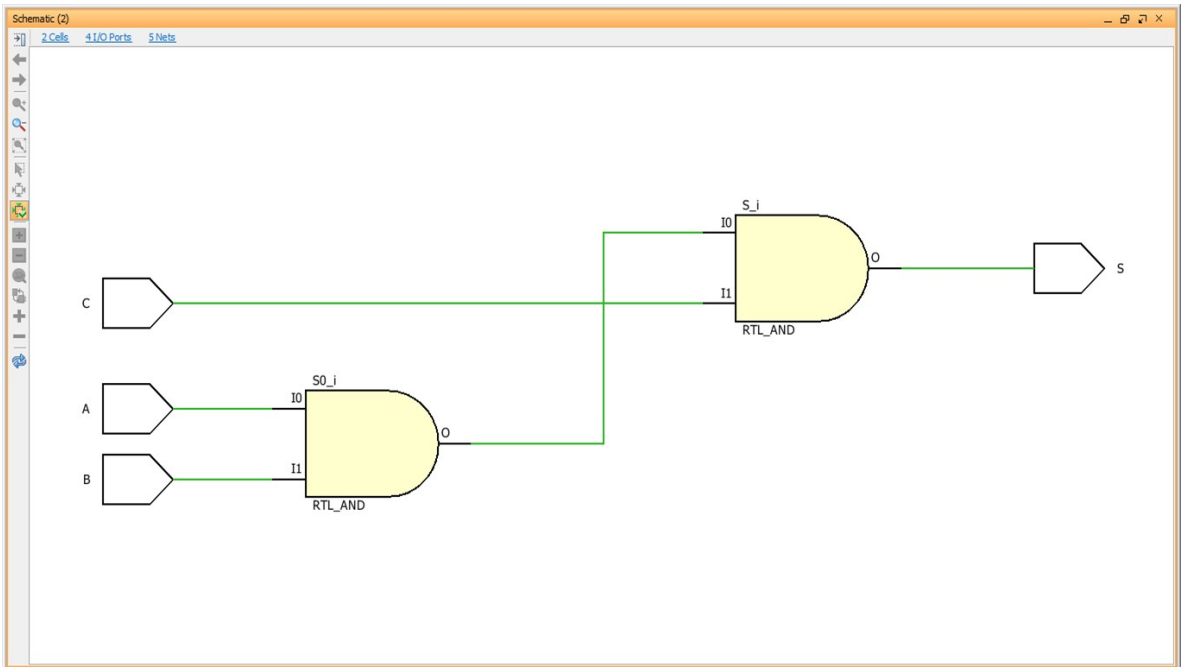

| UnB - FGA | Prancha 1 | Versão | Data | Elaborador por: | Adrianne Alves da Silva |
|---|---|---|---|---|---|
| LAB PED 1 | Portas Lógicas | 1.0 | 24/03/2017 | Revisado por: | Ana Paula Gomes de Matos |

**Figura 25:** Diagrama esquemático da porta XNOR




| UnB - FGA | Prancha 1 | Versão | Data | Elaborador por: | Adrianne Alves da Silva |
|---|---|---|---|---|---|
| LAB PED 1 | Portas Lógicas | 1.0 | 24/03/2017 | Revisado por: | Ana Paula Gomes de Matos |

**Figura 26:** Diagrama esquemático da porta AND com três entradas




| UnB - FGA | Prancha 1 | Versão | Data | Elaborador por: | Adrianne Alves da Silva |
|---|---|---|---|---|---|
| LAB PED 1 | Portas Lógicas | 1.0 | 24/03/2017 | Revisado por: | Ana Paula Gomes de Matos |