

Arquitetura de Computadores

Dispositivos Lógico Programáveis (*Hardware Reconfigurável*)
Arranjo de Portas Programável em Campo - FPGA

Rodolfo Labiapari Mansur Guimarães
Última Atualização: 21 de junho de 2017

rodolfolabiapari@gmail.com

Lattes: <http://goo.gl/MZv4Dc>

Departamento de Computação – Universidade Federal de Ouro Preto
Ouro Preto - MG – Brasil

Sobre esta Aula

Sobre esta Aula

“Não existe pergunta idiota, existe idiota que não pergunta”.

Sobre Mim

- **Formação:** Instituto Federa de Educação, Ciência e Tecnologia de Minas Gerais (IFMG) - Campus Formiga.
- **Pesquisas Aplicadas:**
 - Projeto e Desenvolvimento de um Carro Robô controlado por *Smartphone*, Utilizando a Plataforma Amaro (Google Android e Arduino);
 - Projeto e Desenvolvimento de um *Hardware* Reconfigurável de Criptografia para a Transmissão Segura de Dados.
- **Trabalho de Conclusão de Curso:**
 - Desenvolvimento de um *Device Driver* para Comunicação com um *Hardware* Reconfigurável.
- **Lattes:** <http://goo.gl/MZv4Dc>.

Matemática Discreta - Lógica Proposicional e Eletrônica Digital

- Chamamos **tabela verdade** de uma função booleana a tabela que apresenta **os valores da função** $y = f(A, B)$ para **todas as combinações possíveis** dos valores das variáveis.

Figura 1: Exemplo de tabela verdade com 2 entradas

- Tabela é **uma** de várias formas de representação de funções lógicas.

A	B	y
0	0	1
0	1	0
1	0	1
1	1	1

Fonte: Disponível em: [http://eletro.g12.br/
arquivos/materiais/electronica4.pdf](http://eletro.g12.br/arquivos/materiais/electronica4.pdf).

Figura 2: Exemplo de tabela verdade com 3 entradas

A	B	C	S
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Fonte: Disponível em: <http://eletro.g12.br/arquivos/materiais/eletronica4.pdf>.

Figura 3: Exemplo de tabela verdade com 3 entradas

A	B	C	F
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Diagram illustrating the output rows highlighted by orange boxes:

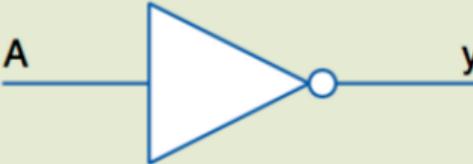
- Row 1 (highlighted): $\bar{A} \bar{B} \bar{C}$
- Row 5 (highlighted): $\bar{A} B C$
- Row 6 (highlighted): $A \bar{B} \bar{C}$
- Row 8 (highlighted): $A B C$

Fonte: Disponível em: <http://eletro.g12.br/arquivos/materiais/eletronica4.pdf>.

- Portas lógicas são **circuitos eletrônicos básicos** que possuem uma ou mais entradas e uma única saída.
- Nas entradas e na saída, podemos associar valores booleanos.
- Em eletrônica digital atribuímos valores de tensão
 - Associamos ao 5 V o estado “1” e ao 0 V, o estado “0”.

Lógica e Eletrônica - Portas Lógicas - Porta NOT

Figura 4: Representação da Porta Lógica NOT

símbolo	tabela verdade	expressão booleana						
	<table border="1"><thead><tr><th>A</th><th>y</th></tr></thead><tbody><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></tbody></table>	A	y	0	1	1	0	$y = \bar{A}$
A	y							
0	1							
1	0							

POR TA INVERSORA tem somente 1 entrada

Fonte: Disponível em: <http://eletro.g12.br/arquivos/materiais/electronica4.pdf>.

Lógica e Eletrônica - Portas Lógicas - Porta AND

Figura 5: Representação da Porta Lógica AND

símbolo	tabela verdade	expressão booleana															
	<table border="1"><thead><tr><th>A</th><th>B</th><th>y</th></tr></thead><tbody><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></tbody></table>	A	B	y	0	0	0	0	1	0	1	0	0	1	1	1	$y = AB \text{ ou } y = A \cdot B$ A saída é “1” somente se todas as entradas forem “1”
A	B	y															
0	0	0															
0	1	0															
1	0	0															
1	1	1															

Fonte: Disponível em: <http://eletro.g12.br/arquivos/materiais/electronica4.pdf>.

Lógica e Eletrônica - Portas Lógicas - Porta NAND

Figura 6: Representação da Porta Lógica NAND

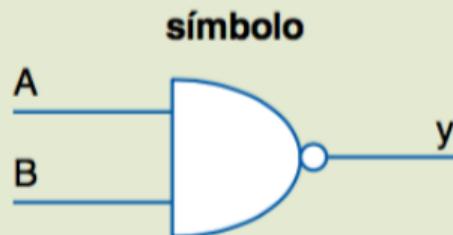


tabela verdade

A	B	y
0	0	1
0	1	1
1	0	1
1	1	0

expressão booleana

$$y = \overline{AB} \text{ ou } y = \overline{A} \cdot \overline{B}$$

A saída é "0" somente se todas as entradas forem "1"

Lógica e Eletrônica - Portas Lógicas - Porta OR

Figura 7: Representação da Porta Lógica OR

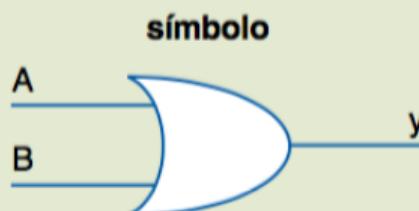


tabela verdade

A	B	y
0	0	0
0	1	1
1	0	1
1	1	1

expressão booleana

$$y = A + B \text{ (lê-se A OU B)}$$

A e B → entradas

y → saída

A saída é "0" somente se todas as entradas forem zero

Lógica e Eletrônica - Portas Lógicas - Porta NOR

Figura 8: Representação da Porta Lógica NOR

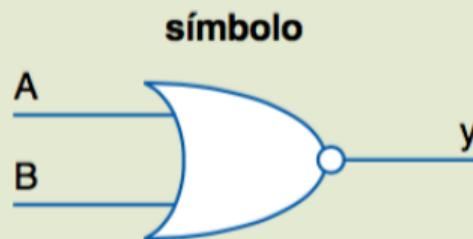


tabela verdade

A	B	y
0	0	1
0	1	0
1	0	0
1	1	0

expressão booleana

$$y = \overline{A + B}$$

A saída é “1” somente se todas as entradas forem zero

Lógica e Eletrônica - Portas Lógicas - Porta XOR

Figura 9: Representação da Porta Lógica XOR

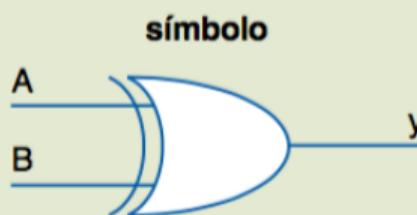


tabela verdade

A	B	y
0	0	0
0	1	1
1	0	1
1	1	0

expressão booleana

$$y = A \oplus B$$

Saída 1 se as entradas forem diferentes

Lógica e Eletrônica - Portas Lógicas - Porta XNOR

Figura 10: Representação da Porta Lógica XNOR

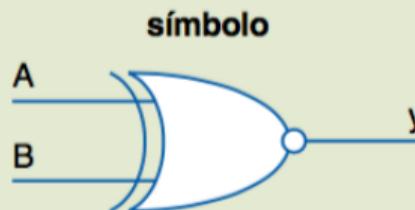


tabela verdade

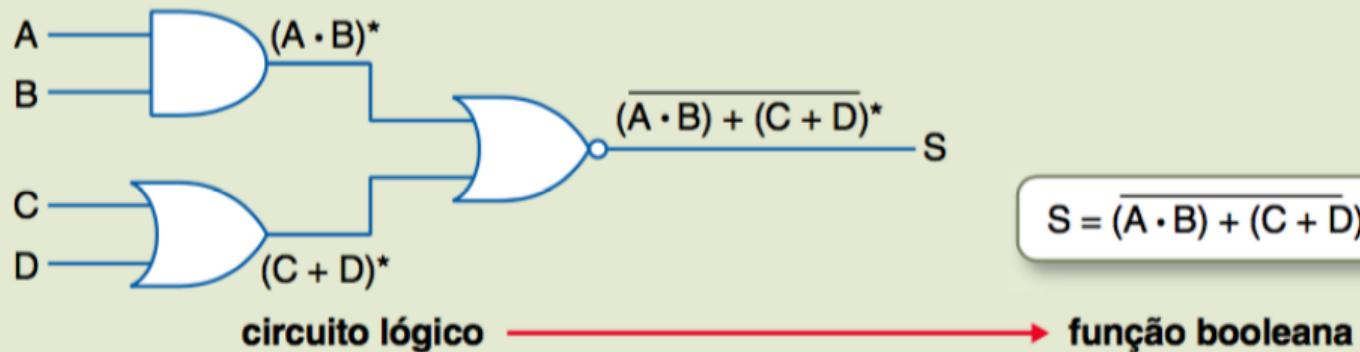
A	B	y
0	0	1
0	1	0
1	0	0
1	1	1

expressão booleana

$$y = A \odot B$$

Saída 1 se as entradas forem iguais

Figura 11: Representação de um circuito simples



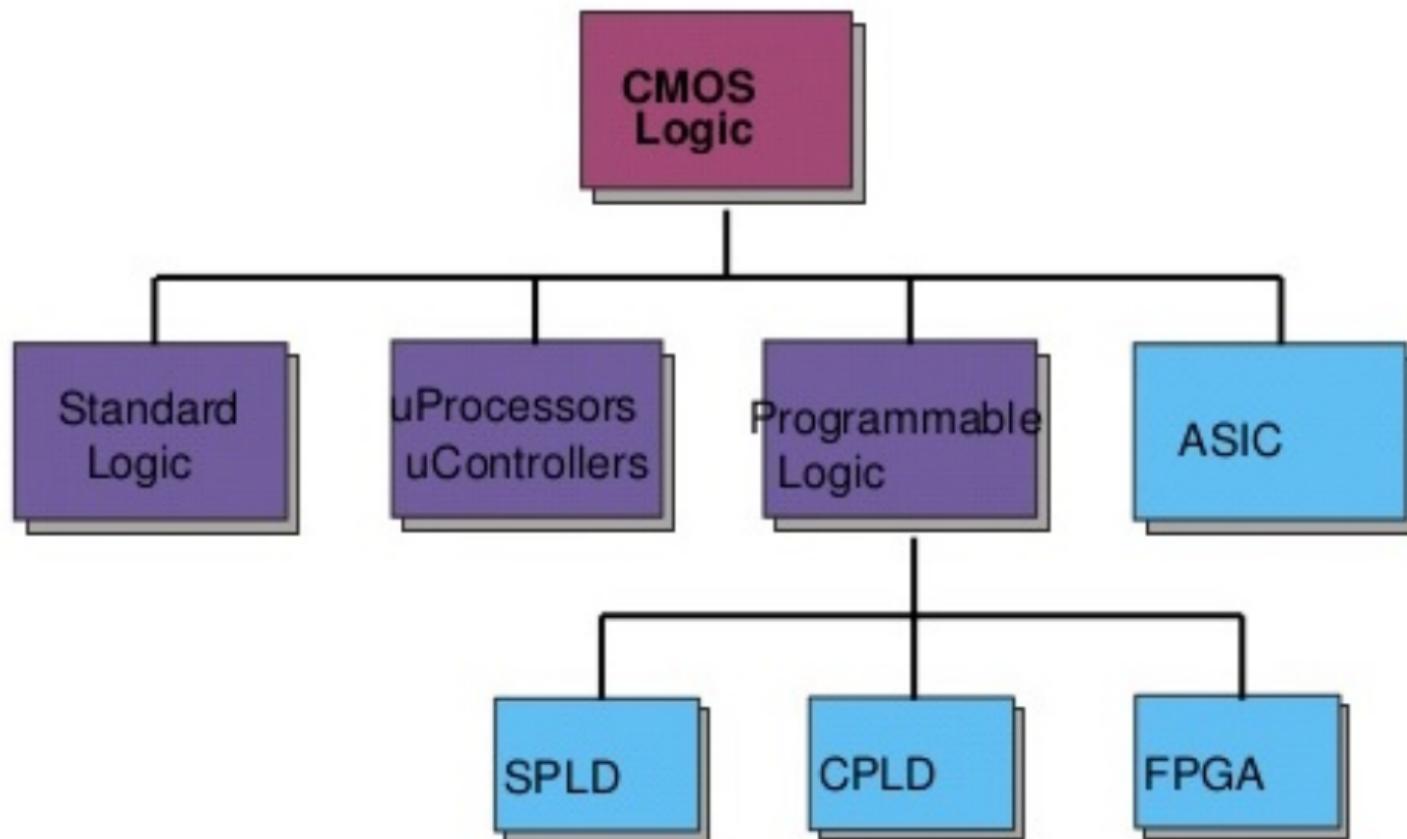
Fonte: Disponível em: <http://eletro.g12.br/arquivos/materiais/electronica4.pdf>.

Questão: Porque falar sobre Portas Lógicas se o assunto da aula é Hardware Reconfigurável?

- Circuitos comprado hoje, como um processador, não permitem que sejam alterados.
- E se

Dispositivos Lógico Programáveis

Dispositivos Lógico Programáveis



Dispositivos Lógico Programáveis

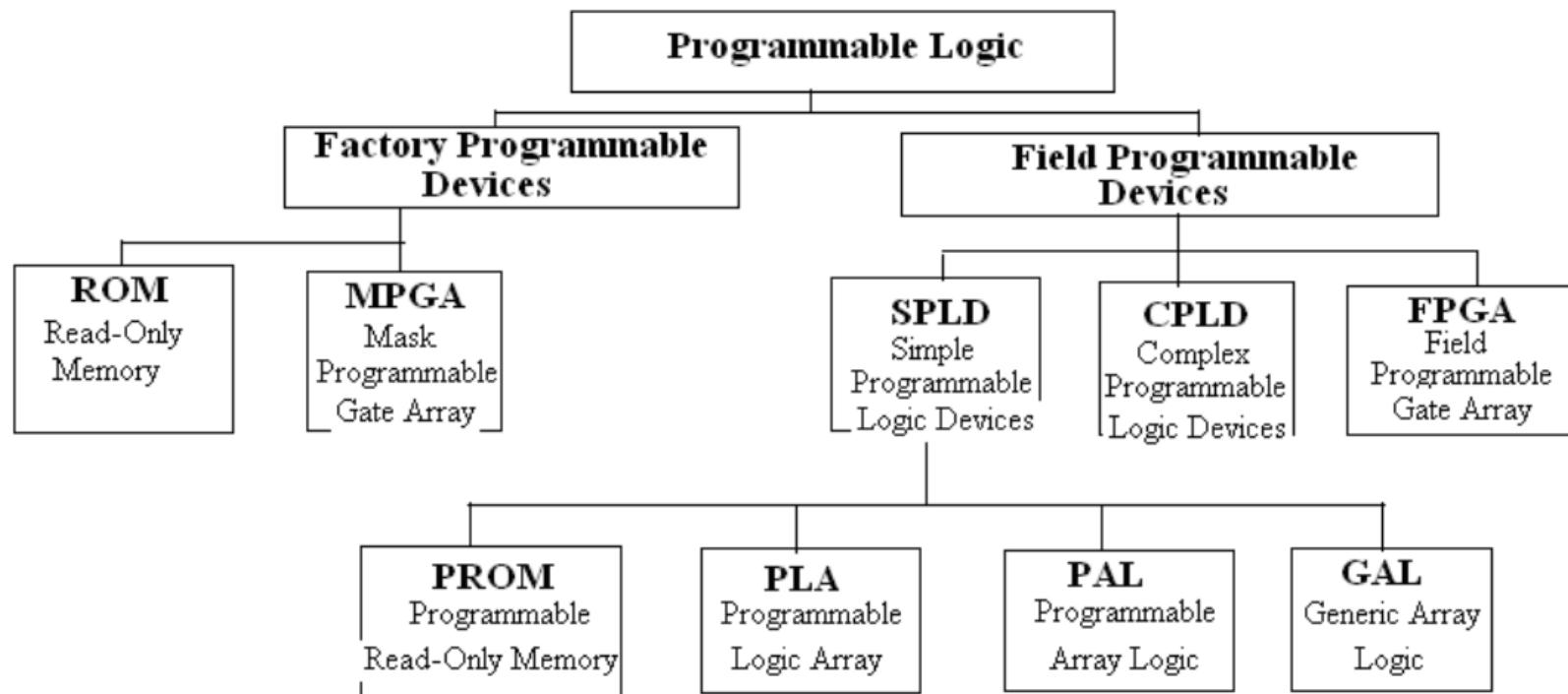


Figura 13: Programmable Logic Device.

Dispositivos Lógico Programáveis

- Tocci [7] cita que os dispositivos lógicos programáveis (PLD¹) são as “*maravilhas de flexibilidade de projeto*”.
- Ao contrário de uma porta lógica, que tem uma função fixa, um PLD tem uma função indefinida após a sua fabricação.
- Para utilizá-lo em um circuito, deve-se **programá-lo previamente**.
- A lógica programável proporciona ao projetista:
 - A possibilidade de se **adequar aos vários níveis de projetos** sendo estes:
 - Para desenvolvimento de protótipos ou mesmo circuitos finais.
 - **Fácil alteração** do projeto a **qualquer momento**;
 - Programação por meio de Álgebra Boole, Mapa de Karnaugh, e Linguagens de Descrição de Hardware.

¹do inglês *Programmable Logic Device*.

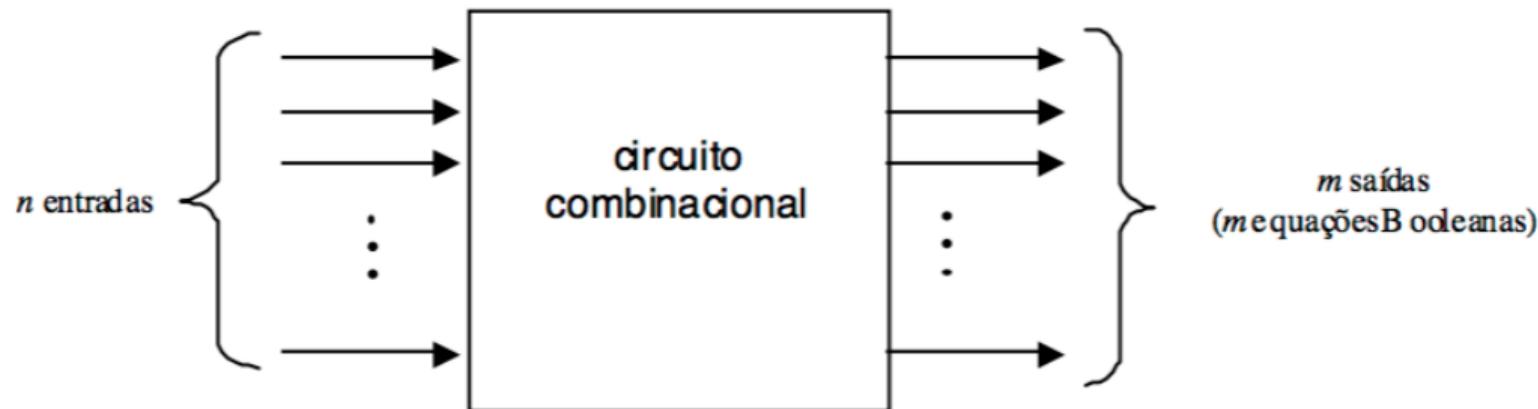


Figura 14: Exemplo de um circuito combinacional.

Dispositivos Lógico Programáveis - FPGA

- Em meados dos anos 80, a empresa **Xilinx** e Gerald Estrin anuncia uma nova arquitetura.
 - Isso foi um **choque** na época pois cria-se então um **novo paradigma de projetos de circuitos integrados**;
 - Entre vários tipos de PLD existe um tipo que é baseado na tecnologia *gate array* e é chamado de **FPGA**².
- O **FPGA** usa uma rede de portas lógicas cuja programação é feita pelo **projetista e não pelo fabricante**.
 - Sendo reprogramados, o usuário pode utilizá-lo num projeto e logo em seguida reprogramá-lo para que execute outro projeto.
 - Deixando mais claro, o termo “**campo**” na nomenclatura é apenas um termo da engenharia utilizado para indicar o **mundo de fora da fábrica**.

²Do inglês *Field-Programmable Gate Array*, Arranjo de Portas Programável em Campo.

- Excelentes para **desenvolver um protótipo de circuito integrado** a fim da realização de testes **antes da fabricação em massa** [6] e **ensino**.
- Seu custo em relação ao ASIC³ é menor;
- Seu tempo total gasto desde a especificação do projeto até sua sintetização também é menor
 - ASICs são produzidos geralmente em **larga escala**
 - O custo de fabricação de 1 circuito é caro;
 - O custo de fabricação em massa também é caro.
 - Qualquer erro no ASIC, deve-se reiniciar todo o processo de desenvolvimento
 - Sendo que no FPGA isso levaria talvez algumas horas/minutos e custo zero.
 - Circuitos feitos manualmente...

³Do inglês *Application Specific Integrated Circuits*.

1. Descrição em Alto Nível;
2. Simulação Lógica;

1. **Descrição em Alto Nível;**
2. **Simulação Lógica;**
3. Síntese e Mapeamento;
4. Distribuição dos Blocos Lógicos e Roteamento dos Sinais;
5. Análise Temporal e Simulação;

1. **Descrição em Alto Nível;**
2. **Simulação Lógica;**
3. Síntese e Mapeamento;
4. Distribuição dos Blocos Lógicos e Roteamento dos Sinais;
5. Análise Temporal e Simulação;
6. Configuração de Dispositivos de Lógica Programável;

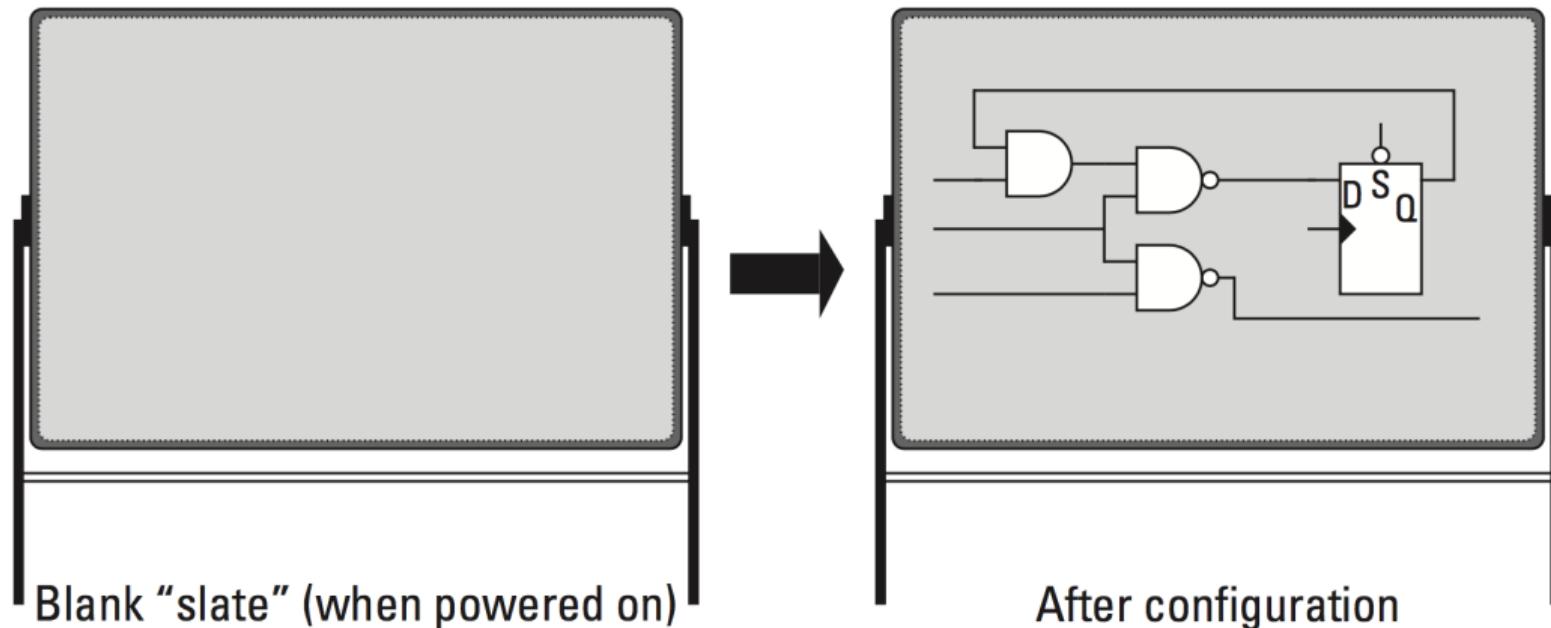
1. **Descrição em Alto Nível;**
2. **Simulação Lógica;**
3. Síntese e Mapeamento;
4. Distribuição dos Blocos Lógicos e Roteamento dos Sinais;
5. Análise Temporal e Simulação;
6. Configuração de Dispositivos de Lógica Programável;
7. **Teste de Avaliação.**

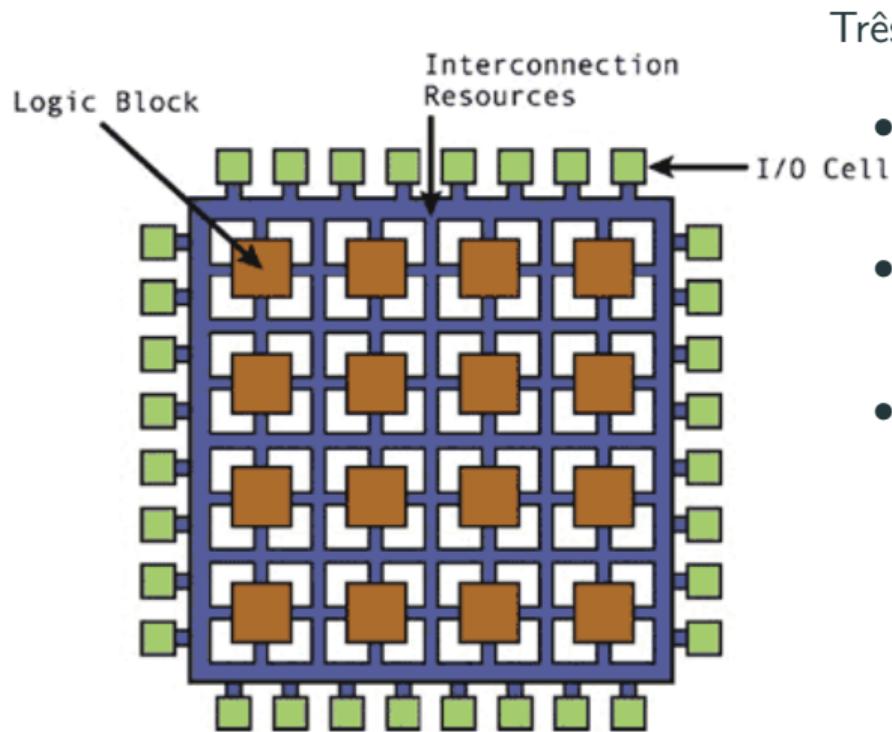
FPGA - Hardware

- Nos FPGAs, sua configuração é totalmente **volátil**
 - O projeto deve ser **recarregada** sempre que é **ligada**;
 - Para isso, a configuração é normalmente guardada numa memória não volátil como a EEPROM.
- Internamente:
 - Consiste num **arranjo de blocos lógicos e canais de roteamento**.
 - Significa que é capaz de alterar seus caminhos de dados/fluxos **habilitando/desabilitando módulos** [2].

FPGA - Hardware - Visão Geral

Figura 15: “FPGA é um *quadro negro* que pode ser *desenhado* qualquer circuito digital”.



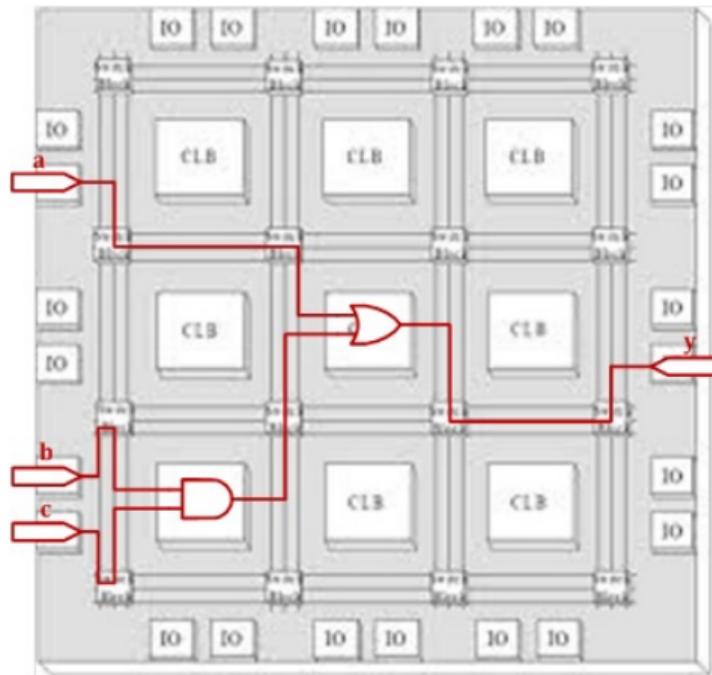


Três módulos:

- Blocos de Entrada e Saída;
- Blocos Lógicos Configuráveis;
- Chaves de Interconexão.

Figura 16: Exemplo de roteamento interno.

FPGA - Hardware - Roteamento de Blocos



Três módulos:

- Blocos de Entrada e Saída;
- Blocos Lógicos Configuráveis;
- Chaves de Interconexão.

Figura 17: Outro exemplo de roteamento interno no FPGA.

FPGA - Hardware - Roteamento de Blocos

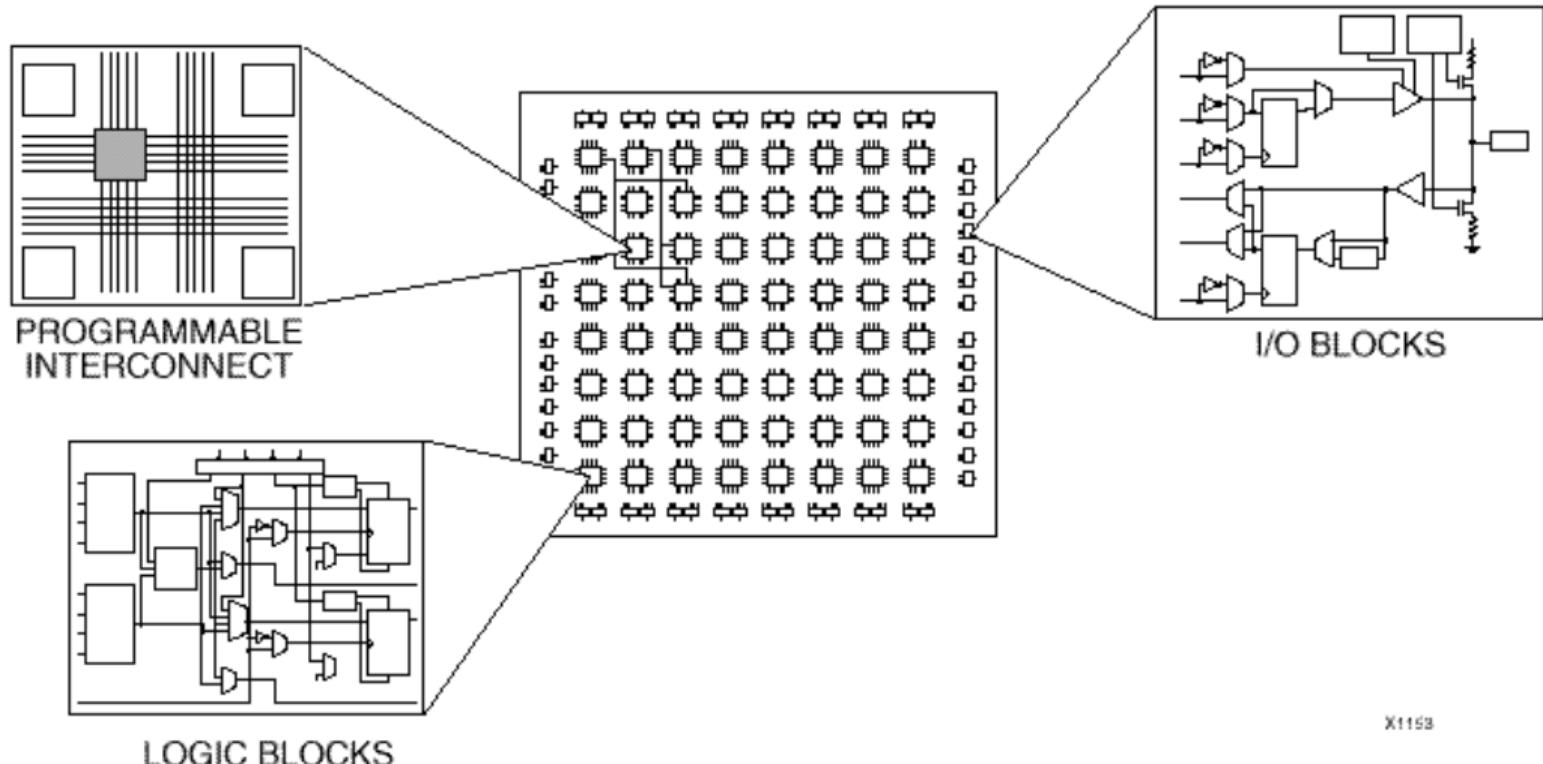


Figura 18: Demonstração mais complexa de síntese.

FPGA - Hardware - Roteamento de Blocos

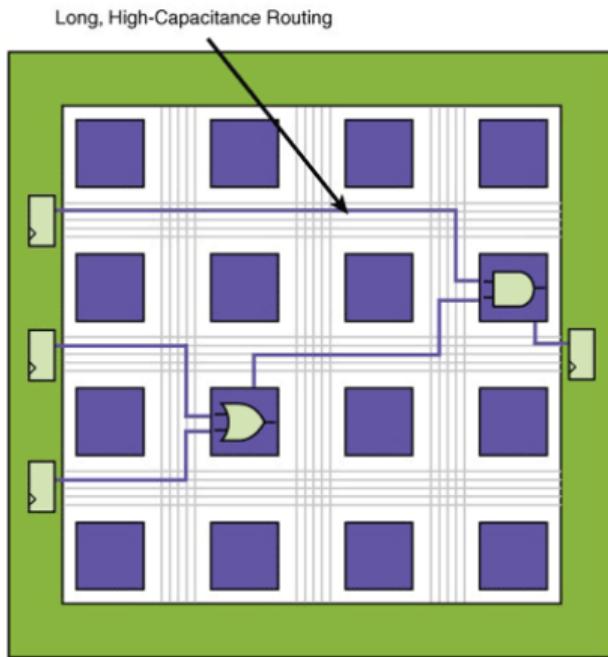


Figura 19: Exemplo de roteamento interno.

- O FPGA **não possui funções lógicas** como AND, OR...
 - Seu arranjo de células reconfiguráveis permite a **implementação** dessas portas/tabelas.
- Distribuições lógicas ruins e encaminhamentos mal feitos geram atrasos de processamento
 - Estudar qual a melhor maneira projetar cada circuito;
 - A implementação afeta diretamente seu seu design e por fim, no desempenho.

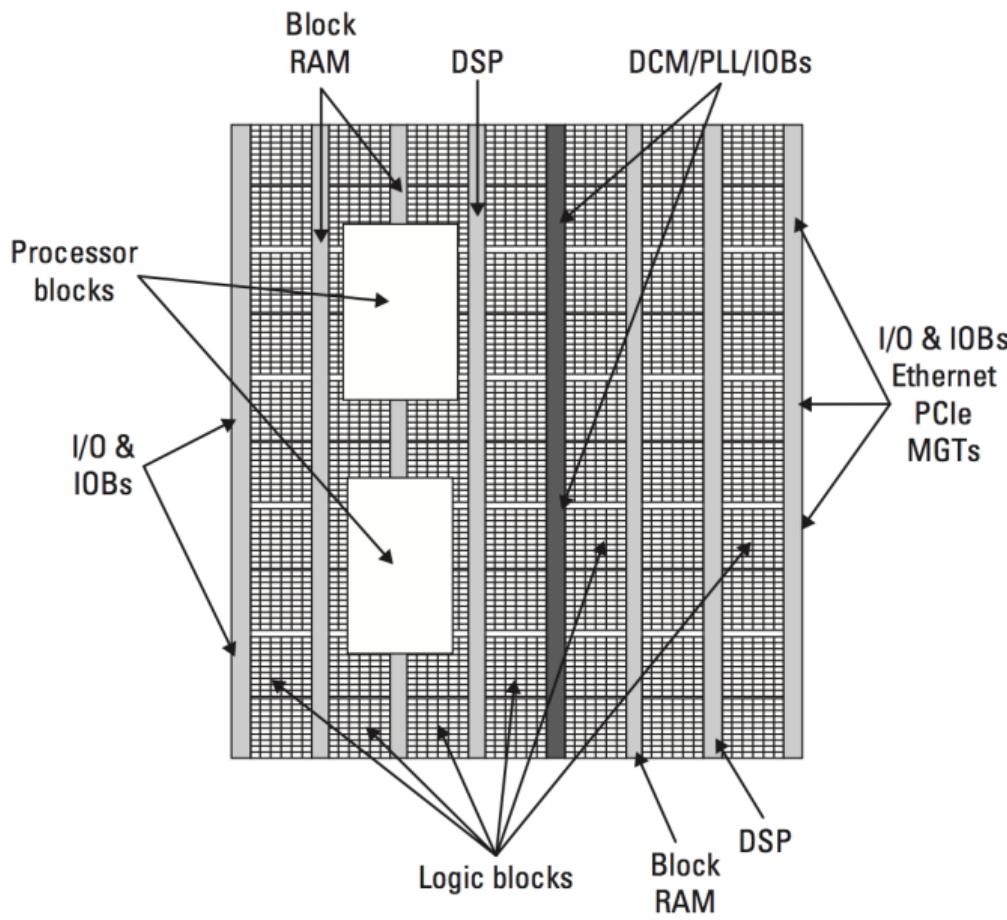


Figura 20: Abstração de um FPGA.

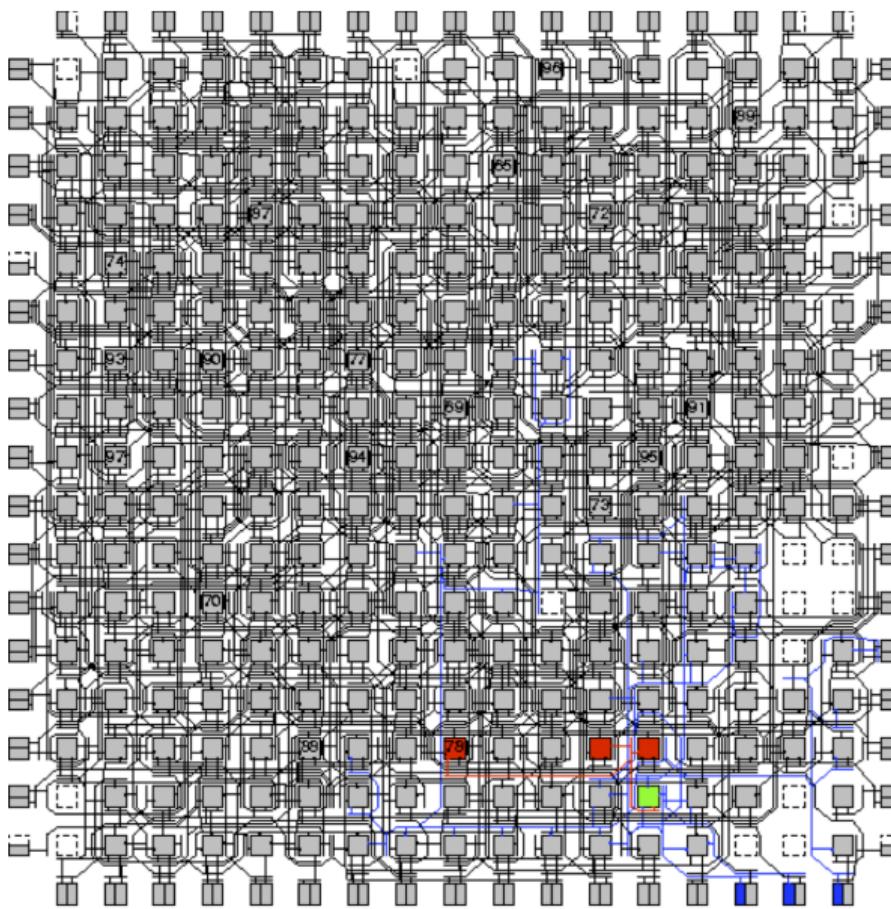


Figura 21: Exemplo de roteamento interno complexo no FPGA.

FPGA - Hardware - Placa

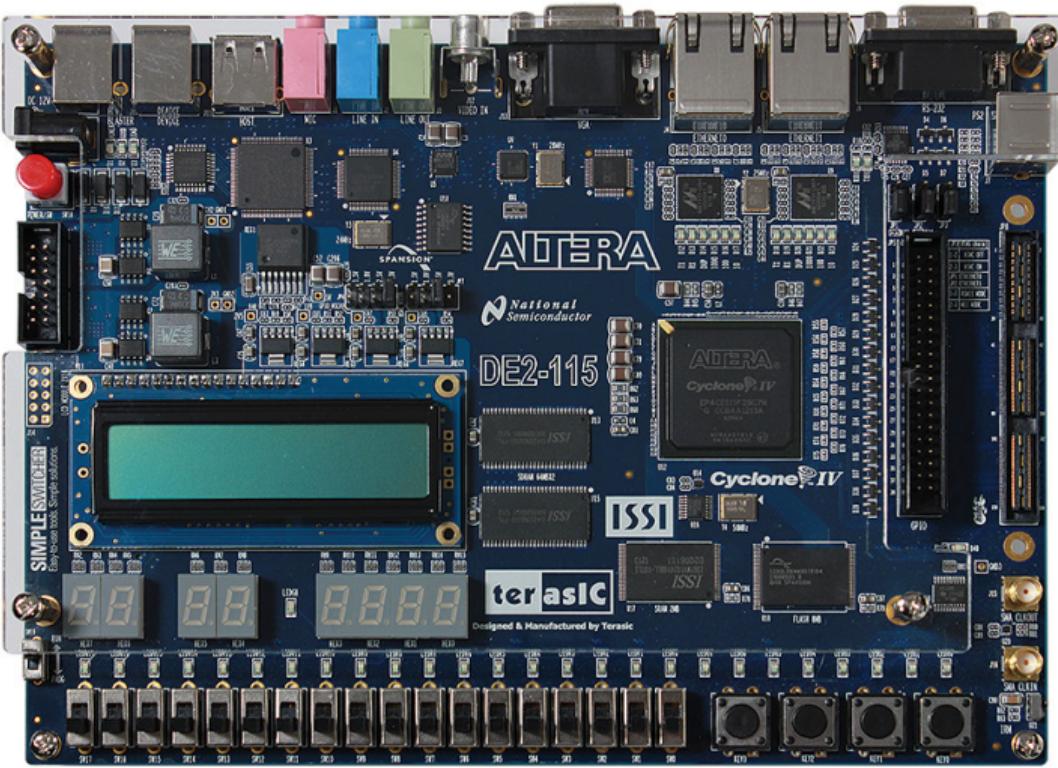
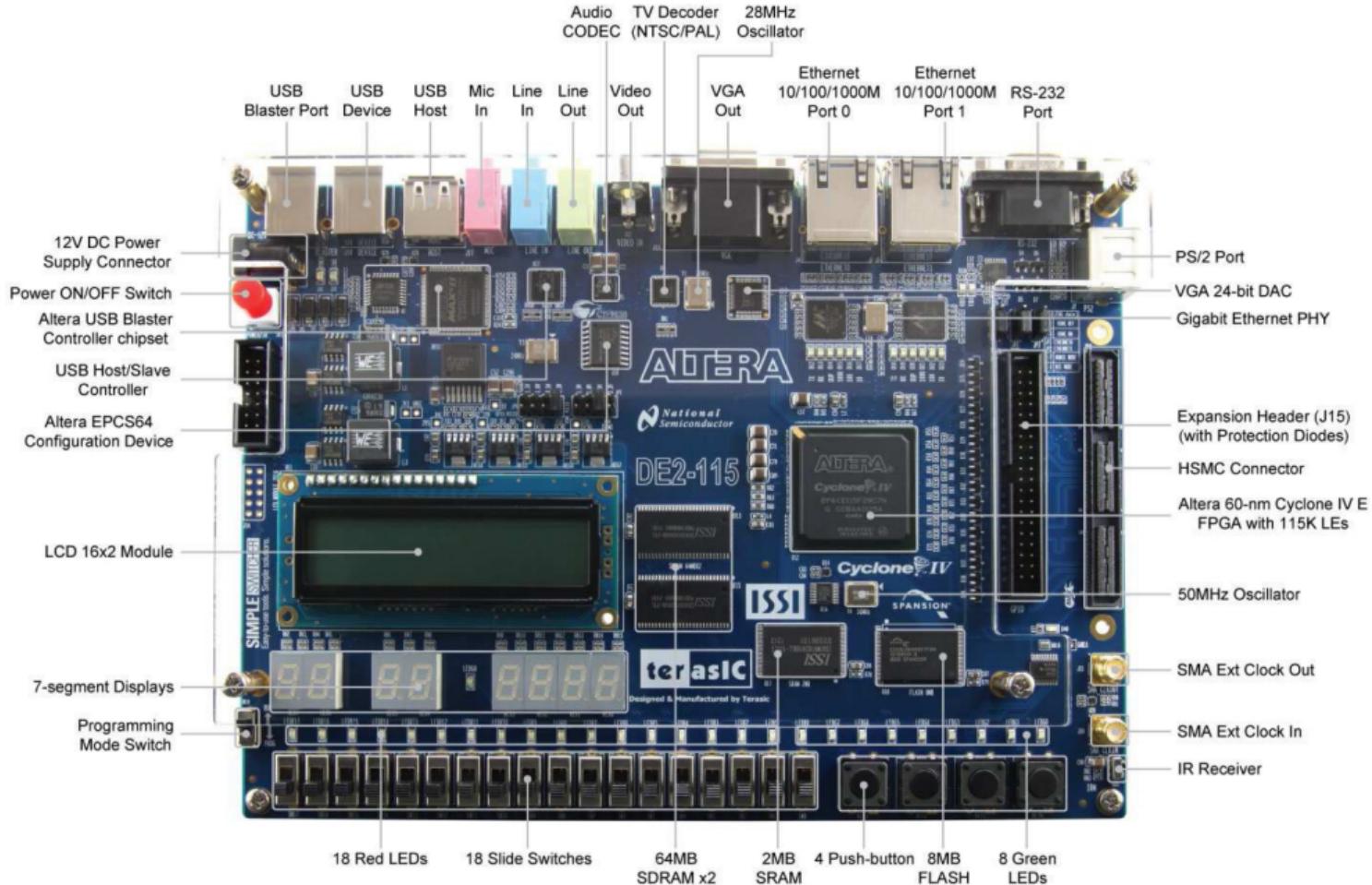


Figura 22: FPGA de Desenvolvimento Altera DE2-115.



FPGA - Software

FPGA - Software - IDE de Desenvolvimento

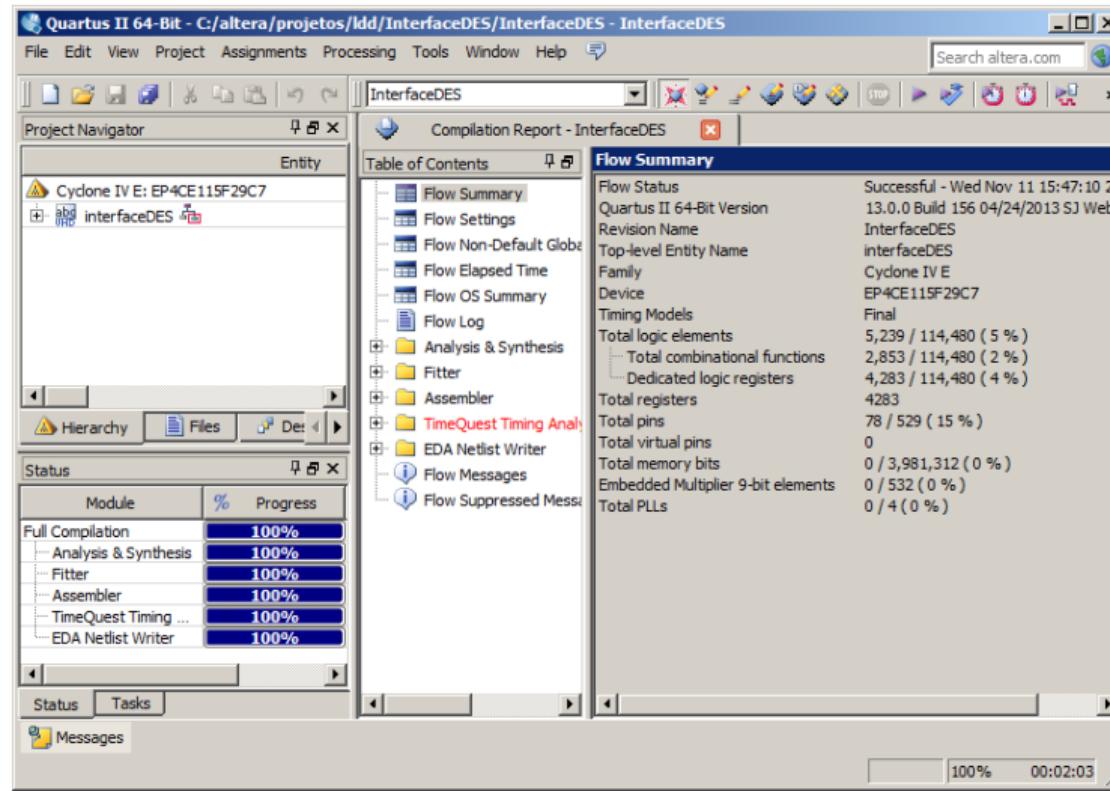


Figura 24: Altera Quartus II.

FPGA - Software - IDE de Desenvolvimento - Desenvolvendo um Projeto

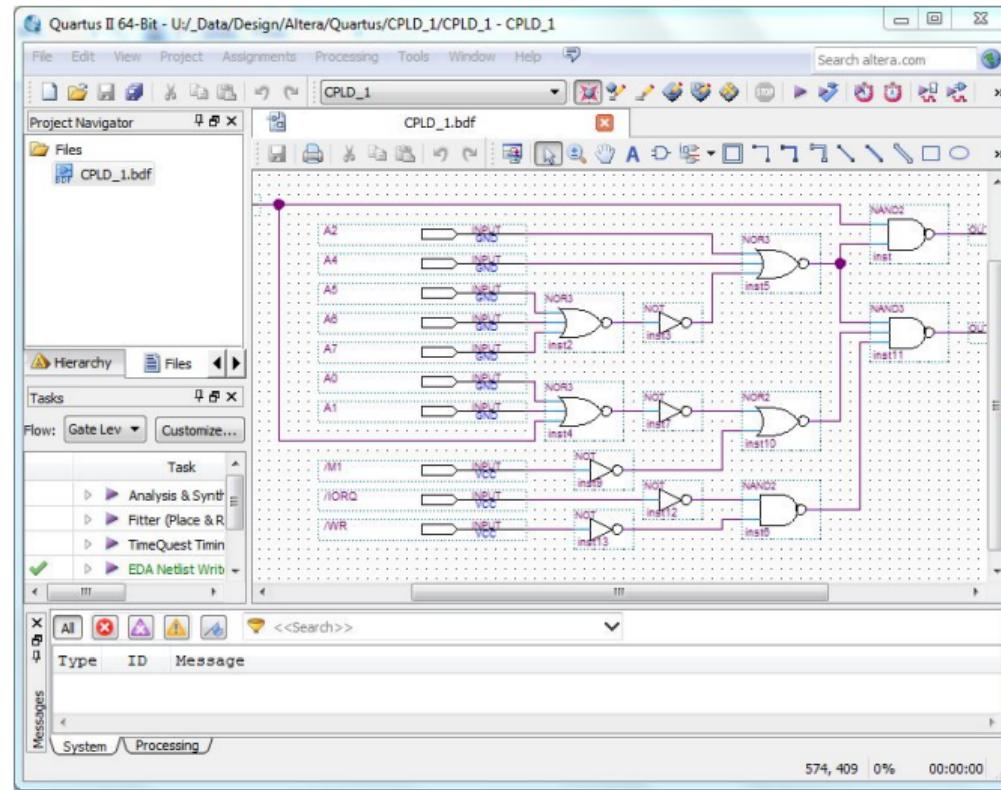


Figura 25: Projeto com assistência computacional usando portas lógicas.

FPGA - Software - IDE de Desenvolvimento - Desenvolvendo um Projeto

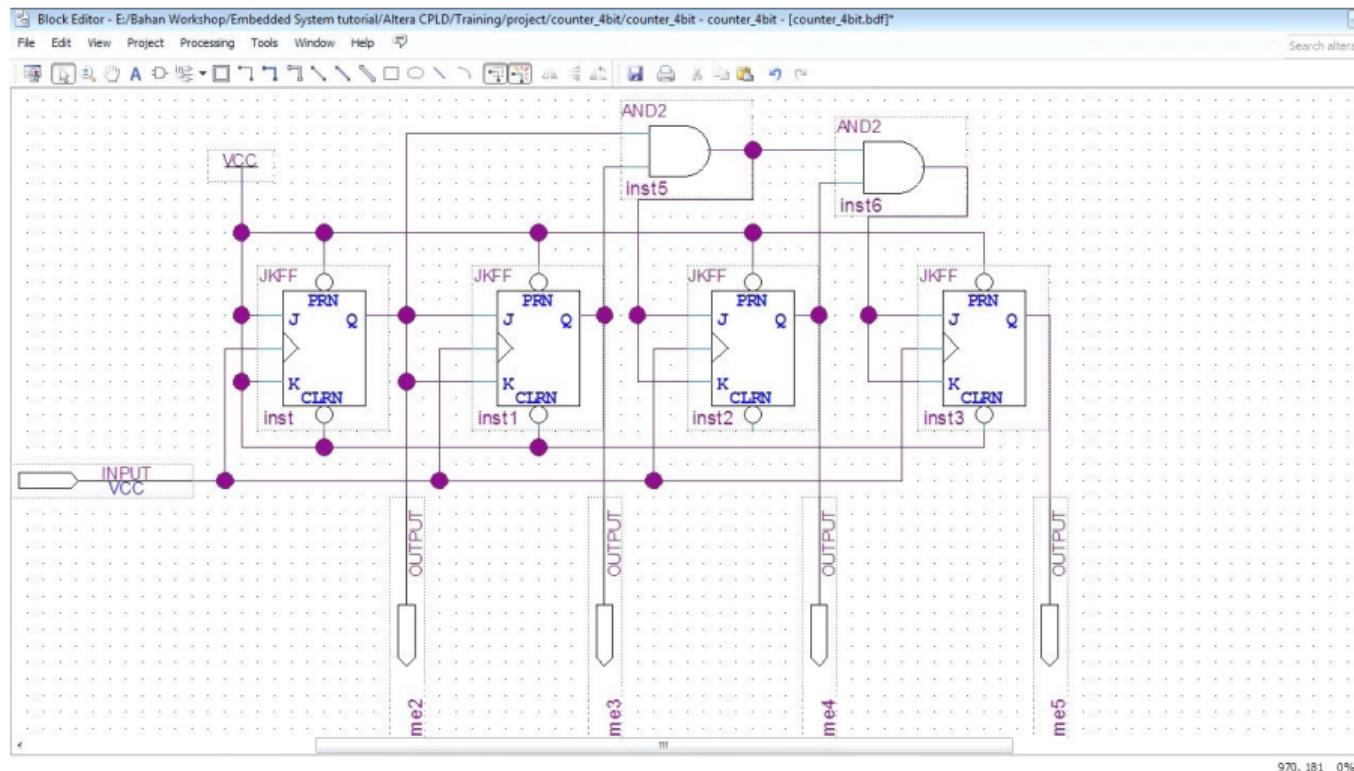


Figura 26: Projeto Contador 4-bit usando portas lógicas.

FPGA - Software - IDE de Desenvolvimento - Desenvolvendo um Projeto

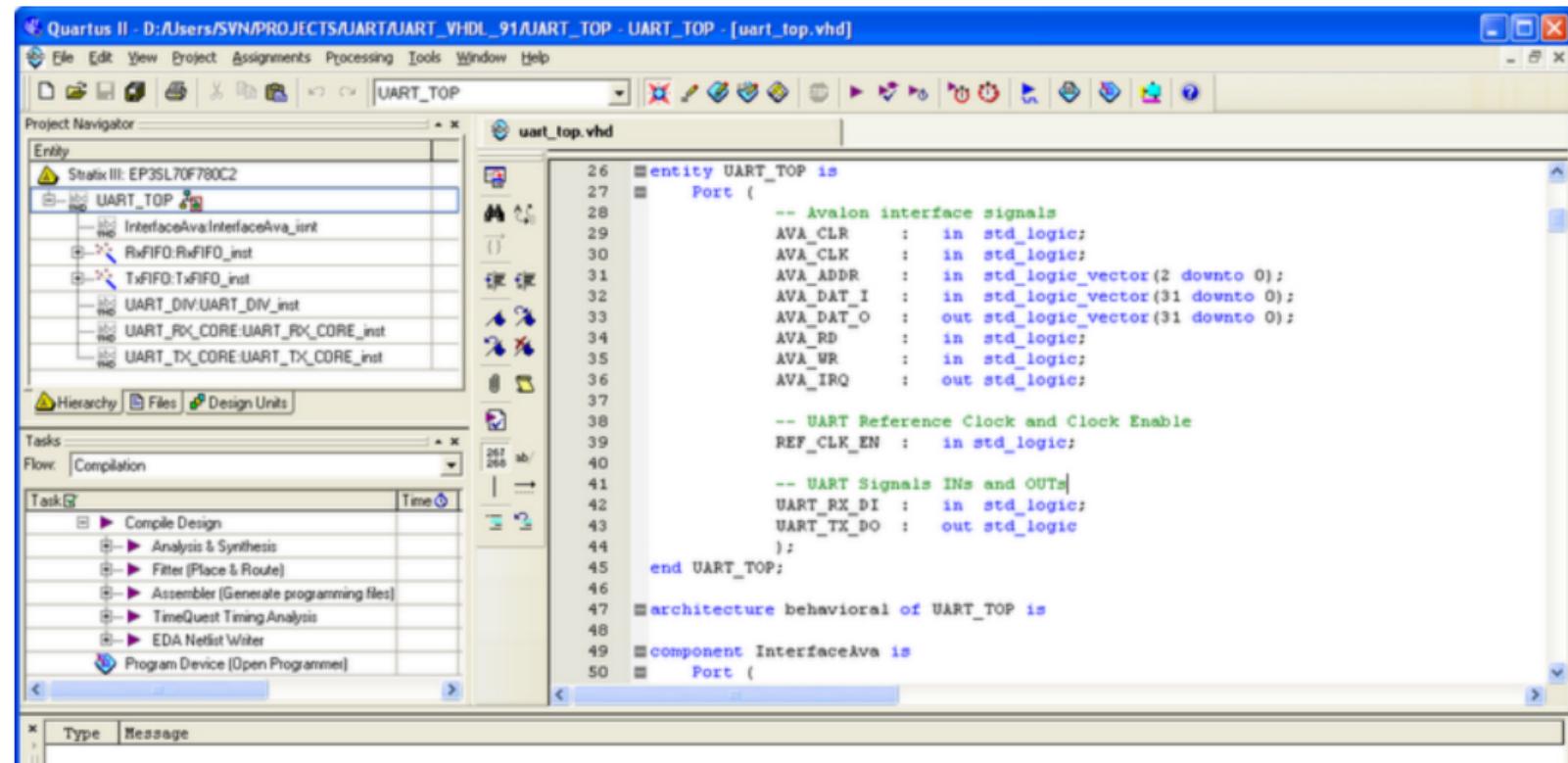


Figura 27: Projeto de uma interface UART usando VHDL.

Row	Inputs			Outputs		Comment
	x	y	c_{in}	c_{out}	s	
0	0	0	0	0	0	$0 + 0 + 0 = 00_2$
1	0	0	1	0	1	$0 + 0 + 1 = 01_2$
2	0	1	0	0	1	$0 + 1 + 0 = 01_2$
3	0	1	1	1	0	$0 + 1 + 1 = 10_2$
4	1	0	0	0	1	$1 + 0 + 0 = 01_2$
5	1	0	1	1	0	$1 + 0 + 1 = 10_2$
6	1	1	0	1	0	$1 + 1 + 0 = 10_2$
7	1	1	1	1	1	$1 + 1 + 1 = 11_2$

Figura 28: Somador Completo.

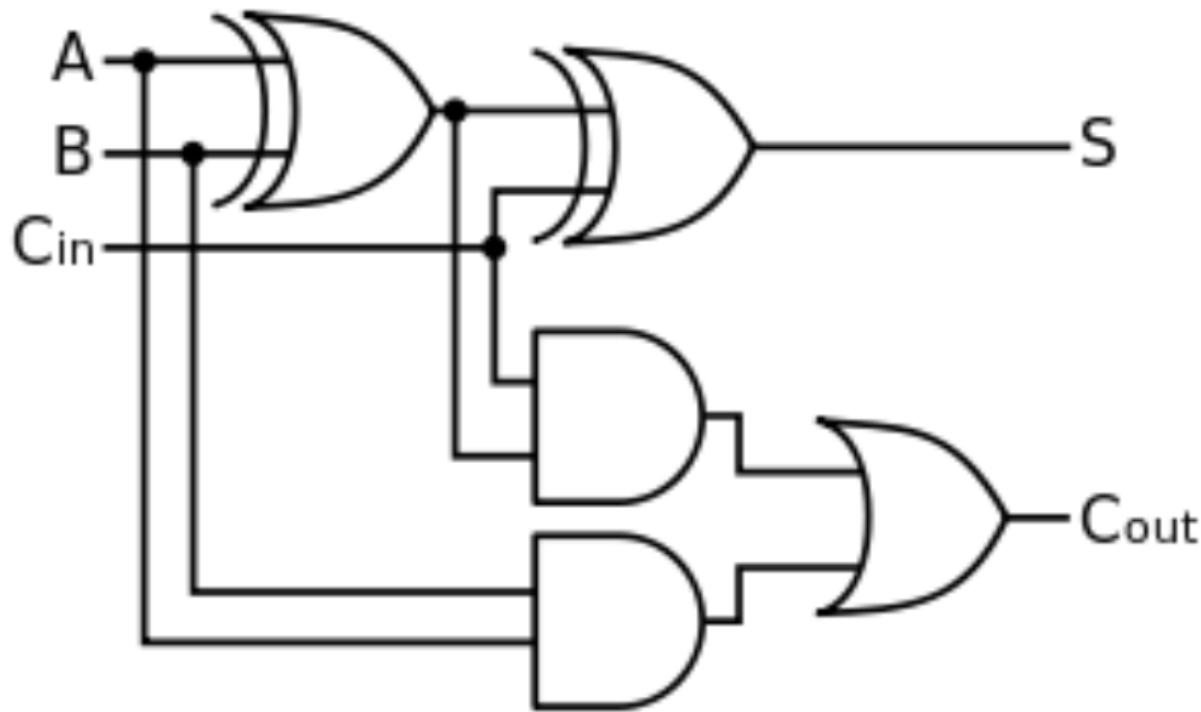


Figura 29: Representação do Somador Completo em Portas Lógicas.

```
library ieee;
use ieee.std_logic_1164.all;

entity fadder is port (
    a, b : in std_logic;
    cin  : in std_logic;
    sum   : out std_logic;
    cout  : out std_logic);
end fadder;

architecture beh of fadder is
begin
    sum <= a xor b xor cin;
    cout <= (a and b) or (b and cin) or (a and cin);
end beh;
```

Figura 30: Código do Somador Completo em VHDL.

FPGA - Software - IDE de Desenvolvimento - Desenvolvendo um Projeto

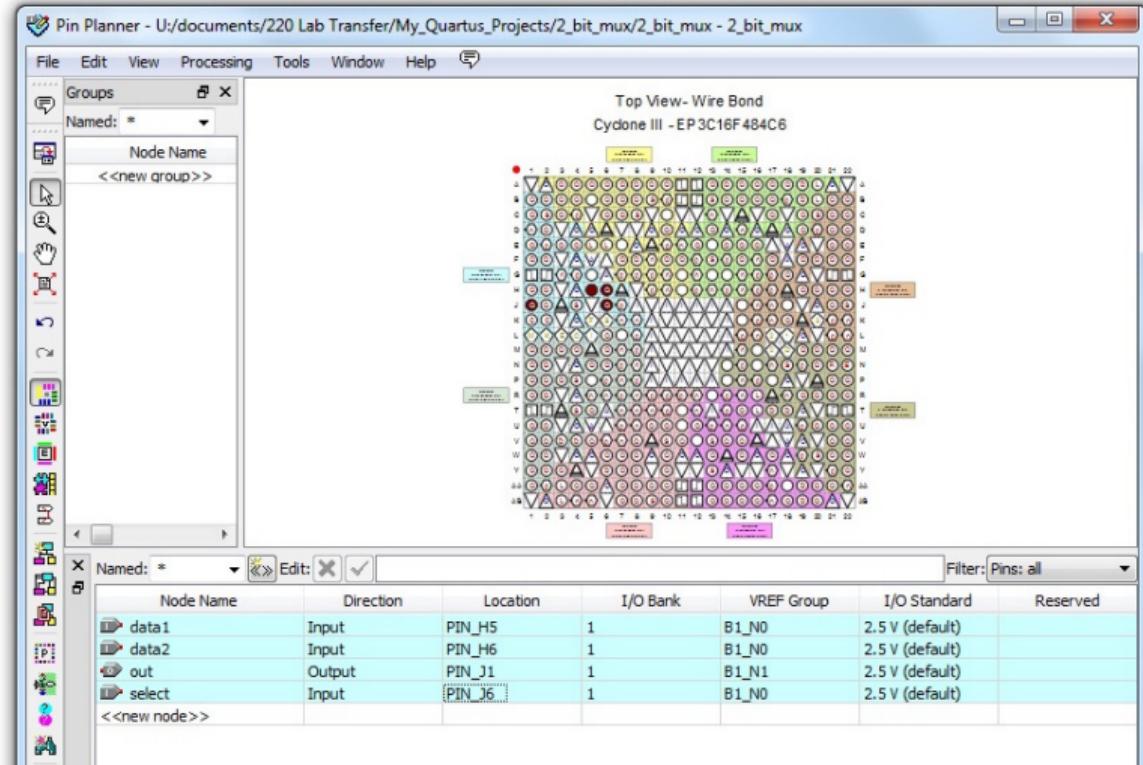
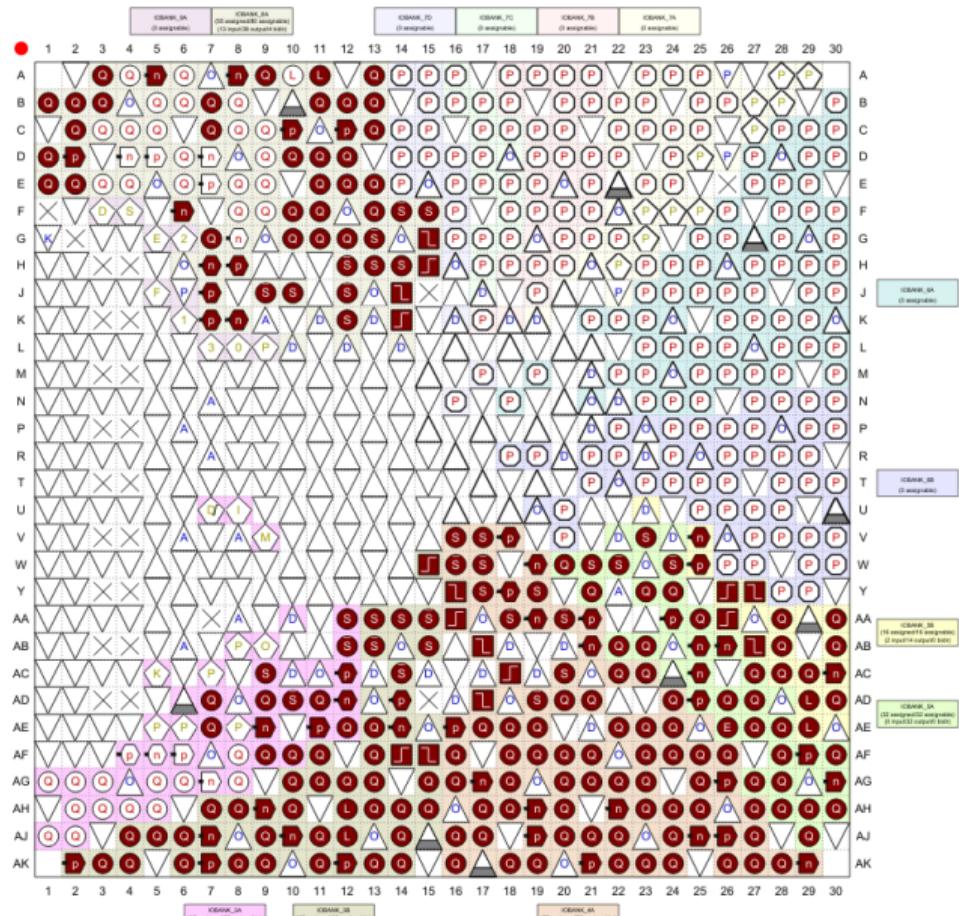


Figura 31: Pin Planner.



Cyclone V - 5CSEMA5F31C6



FPGA - Software - IDE de Desenvolvimento - Compilando um Projeto

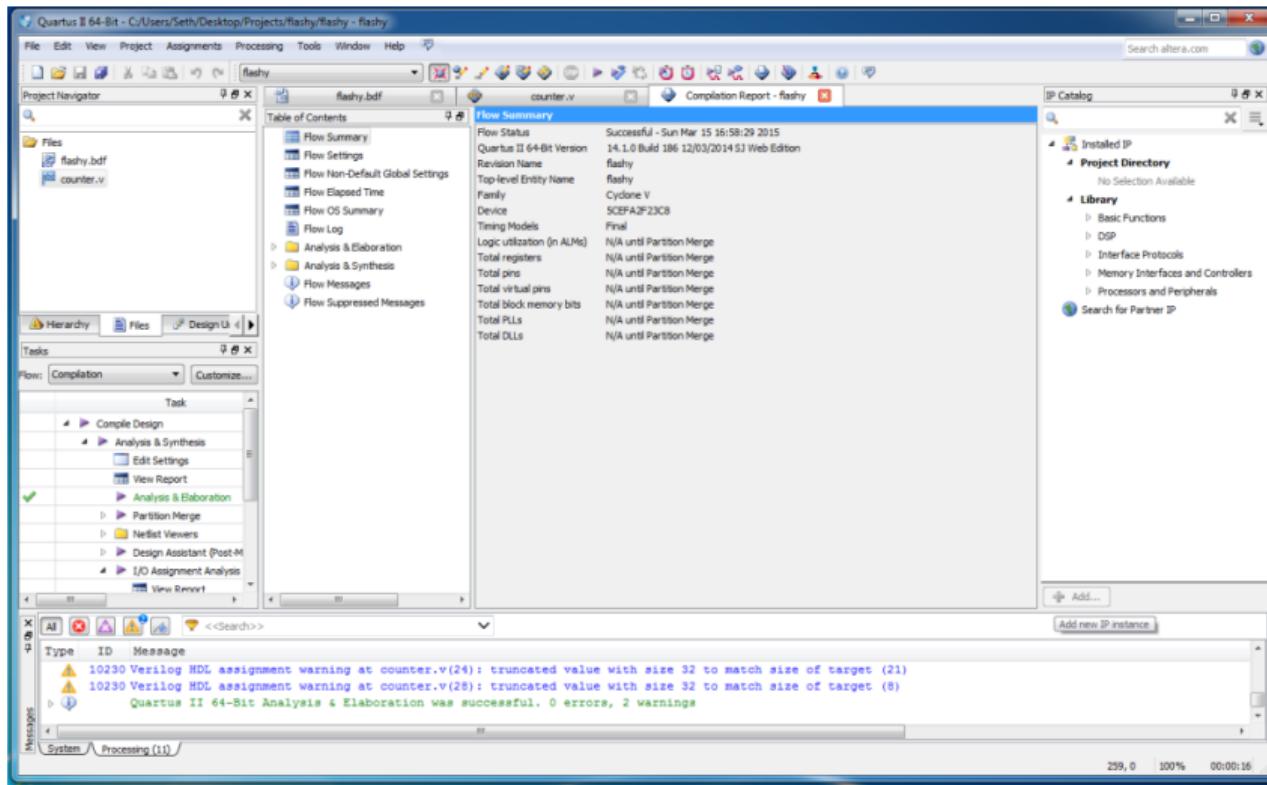


Figura 33: Início de síntese de um projeto.

FPGA - Software - IDE de Desenvolvimento

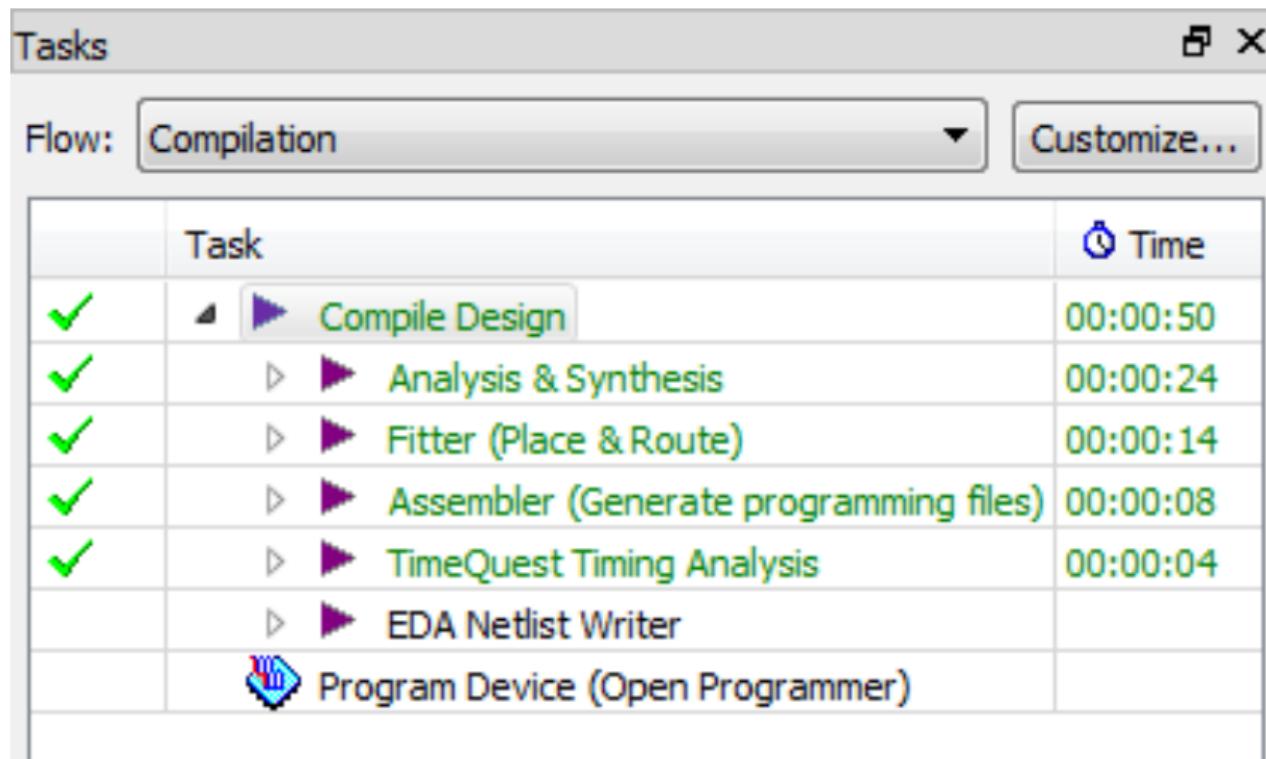


Figura 34: Término de síntese de um projeto.

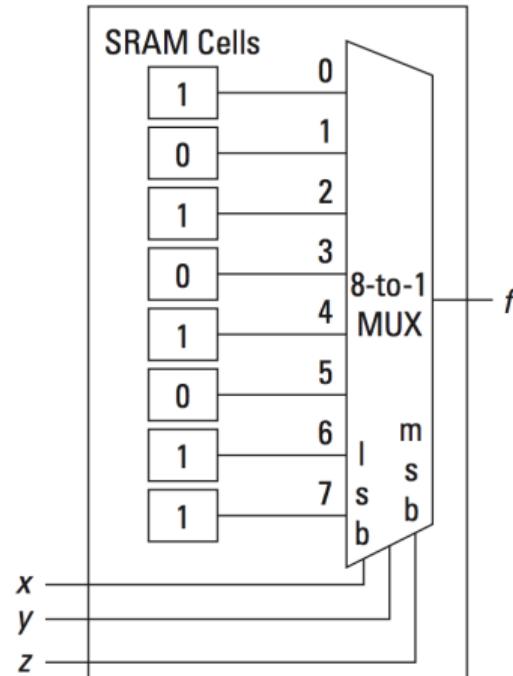
FPGA - Software - IDE de Desenvolvimento



Figura 35: Relatório de compilação final.

x	y	z	$xy + z'$
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

(a)



(b)

Figura 36: Gerador de Função. a) tabela verdade e b) Look-Up Table (LUT) com Flip-flops para armazenamento.

FPGA - Software - IDE de Simulação

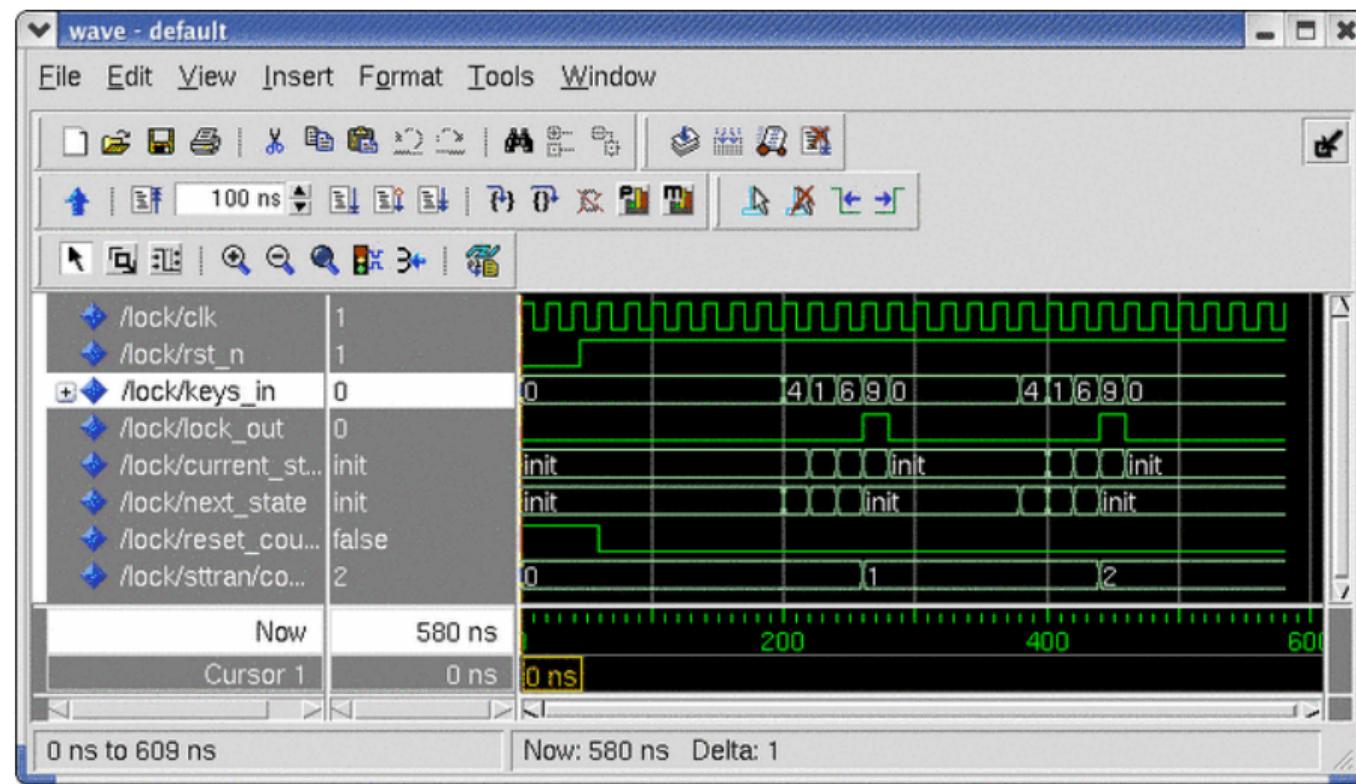


Figura 37: ModelSim.

FPGA - Programação

- A implementação na placa FPGA é feita por:
 - Linguagens de descrição chamadas de **Linguagens de Descrição de Hardware (HDL)**
 - São algumas linguagens: VHDL (*foto*), Verilog, ABEL, etc..
 - Ou por meio do desenvolvimento de **diagramas esquemáticos com ferramentas gráficas assistidas pelo computador** (*foto*).
 - Hoje, também é possível escrever por meio de **linguagens de alto nível** para como Handel-C, C-like ou mesmo o SystemC.
- Alguns conceitos de fundamentais sobre linguagens de descrição de *hardware*:
 - Estas linguagens permitem por exemplo *loops* infinitos ou portas lógicas com inúmeras entradas;
 - Essas são possível serem descritas em *software* mas impossível de implementar em *hardware*.

FPGA - Programação

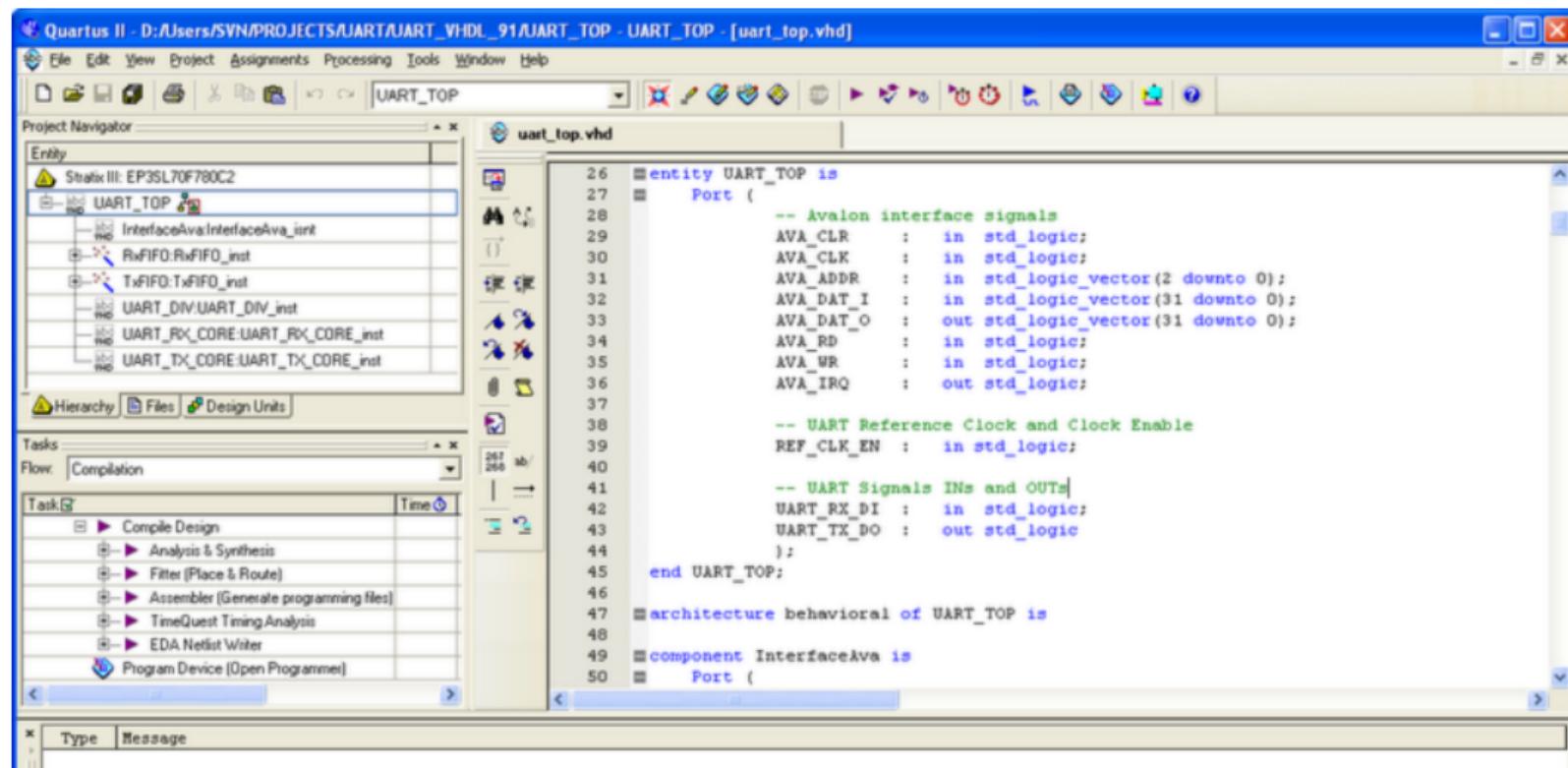


Figura 38: Projeto de uma interface UART usando VHDL.

- A implementação na placa FPGA é feita por:
 - Linguagens de descrição chamadas de **Linguagens de Descrição de Hardware (HDL)**
 - São algumas linguagens: VHDL, Verilog, ABEL, etc..
 - Ou por meio do desenvolvimento de **diagramas esquemáticos com ferramentas gráficas assistidas pelo computador**.
 - Hoje, também é possível escrever por meio de **linguagens de alto nível** para como Handel-C ou mesmo o SystemC.
- Alguns conceitos de fundamentais sobre linguagens de descrição de *hardware*:
 - Estas linguagens permitem por exemplo *loops* infinitos ou portas lógicas com inúmeras entradas;
 - Essas são possível serem descritas em *software* mas impossível de implementar em *hardware*.

FPGA - Programação

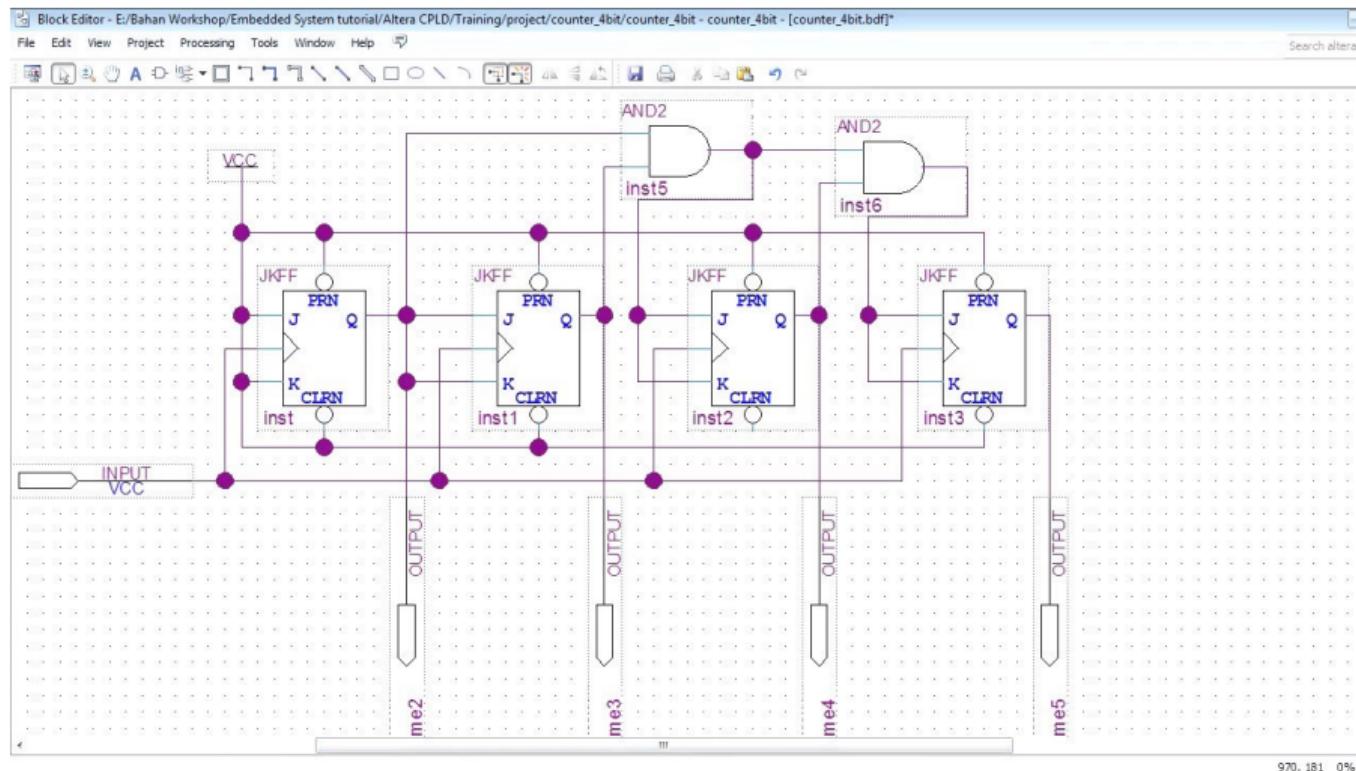


Figura 39: Projeto Contador 4-bit usando portas lógicas.

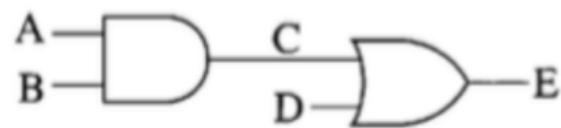
- A implementação na placa FPGA é feita por:
 - Linguagens de descrição chamadas de **Linguagens de Descrição de Hardware (HDL)**
 - São algumas linguagens: VHDL, Verilog, ABEL, etc..
 - Ou por meio do desenvolvimento de **diagramas esquemáticos com ferramentas gráficas assistidas pelo computador**.
 - Hoje, também é possível escrever por meio de **linguagens de alto nível** para como Handel-C ou mesmo o SystemC.
- Alguns conceitos de fundamentais sobre linguagens de descrição de *hardware*
 - Estas linguagens permitem por exemplo *loops* infinitos ou portas lógicas com inúmeras entradas;
 - Essas são possível serem descritas em *software* mas impossível de implementar em *hardware*.

- Linguagem para descrever o **comportamento e estrutura** de um sistema digital.
- A abreviatura VHDL significa VHSIC Hardware Description Language
 - Sendo que VHSIC significa Very High Speed Integrated Circuit.
- Ou seja, seu nome em português é **Linguagem de Descrição de Hardware de Circuitos Integrados com Altíssima Velocidade**.
- Ela tem propriedades suficientes para ser **independente tecnologicamente**
 - Ou seja, a mesma linguagem VHDL utilizada para descrever as tecnologias de hoje, poderá descrever as futuras tecnologias [4].

```
1 C <= A and B;  
2 E <= C or D;
```

```
1 C <= A and B;  
2 E <= C or D;
```

Figura 40: Altera Cyclone IV acoplado à placa da Terasic Altera DE2-115



FPGA - Bibliografia

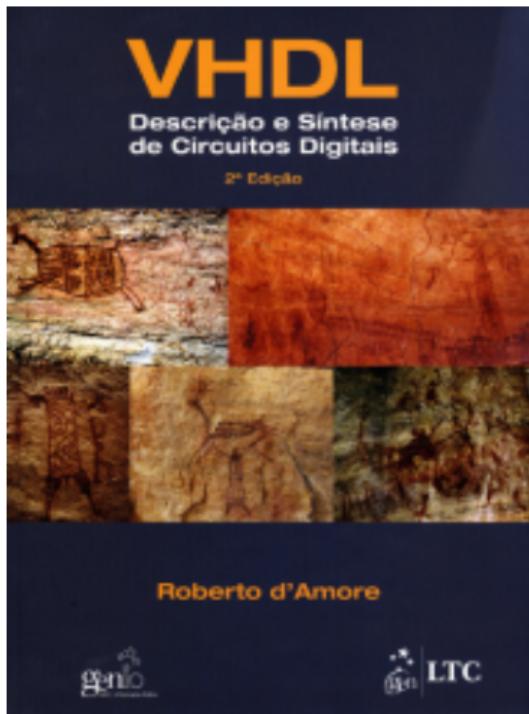


Figura 41: VHDL: Descrição e síntese de circuitos digitais [1].

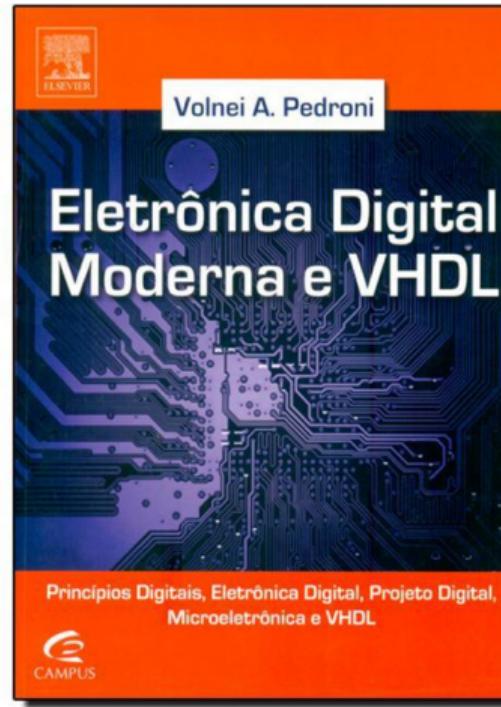


Figura 42: Eletrônica Digital Moderna e VHDL. [3].

FPGA - Bibliografia

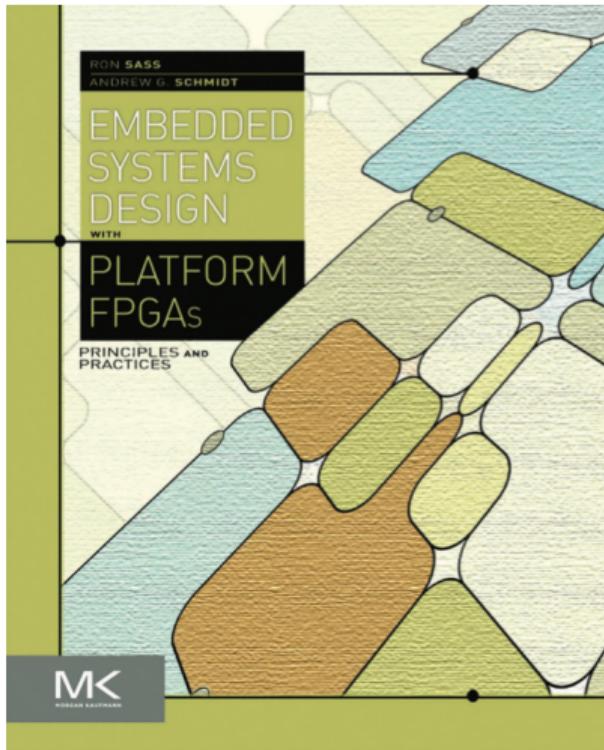


Figura 43: *Embedded Systems Design with Platform FPGAs, Principles and Practices* [5] .

FPGA - Vantagens e Desvantagens

- Vantagens:
 - Não é necessário uma ASIC pra realizar testes:
 - Com o uso do FPGA, é possível fazer os testes necessários prevendo erros;
 - É possível adicionar mais recursos em sua placa para que o projeto tente simular o mais próximo possível do produto final
 - Câmeras, telas *touch* ...
 - Amplamente usado para didática para a construção de circuitos integrados;
 - Aplicação em diversas áreas inclusive áreas de grande importância como projetos no qual necessita de sistema críticos
 - A Nasa...
 - A Intel... (*foto*)
- Desvantagens:
 - Seu custo é bastante elevado para a utilização em projetos pequenos;
 - Ele não possui capacidade de usar recurso analógicos;
 - Seu desempenho entra em desvantagem em comparação com placas ASIC.

Official At Last: Intel Completes \$16.7 Billion Buy of Altera

by Barb Darrow

@gigabarb

DECEMBER 28, 2015, 1:33 PM EDT



Photograph by Justin Sullivan—Getty Images

The acquisition is a game changer for the microprocessor king.

Figura 44: Noticiário. Fonte:

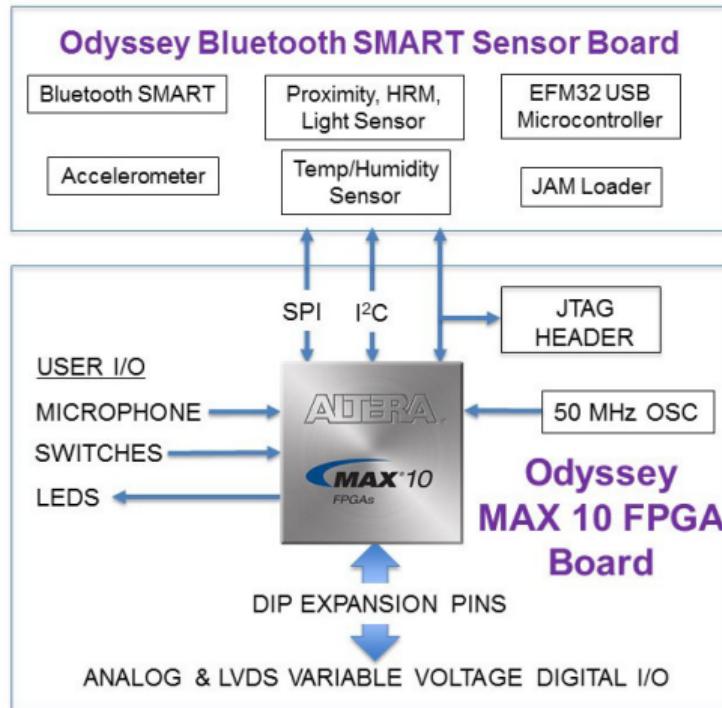
<http://fortune.com/2015/12/28/intel-completes-altera-acquisition/>

FPGA - Vantagens e Desvantagens

- Vantagens:
 - Não é necessário uma ASIC pra realizar testes:
 - Com o uso do FPGA, é possível fazer os testes necessários prevendo erros;
 - É possível adicionar mais recursos em sua placa para que o projeto tente simular o mais próximo possível do produto final
 - Câmeras, telas *touch* ...
 - Amplamente usado para didática para a construção de circuitos integrados;
 - Aplicação em diversas áreas inclusive áreas de grande importância como projetos no qual necessita de sistema críticos
 - A Nasa...
 - A Intel...
- Desvantagens:
 - Seu custo é bastante elevado para a utilização em projetos pequenos;
 - Ele não possui capacidade de usar recurso analógicos;
 - Seu desempenho entra em desvantagem em comparação com placas ASIC.

UFOP - Laboratório iMobilis

Kit Odyssey



- Características:
 - Altera FPGA 10M08;
 - 8 mil elementos lógicos
 - 169 pinos;
 - 55 nm.

Figura 45: Esquemático do Odyssey IV.

Kit Odyssey

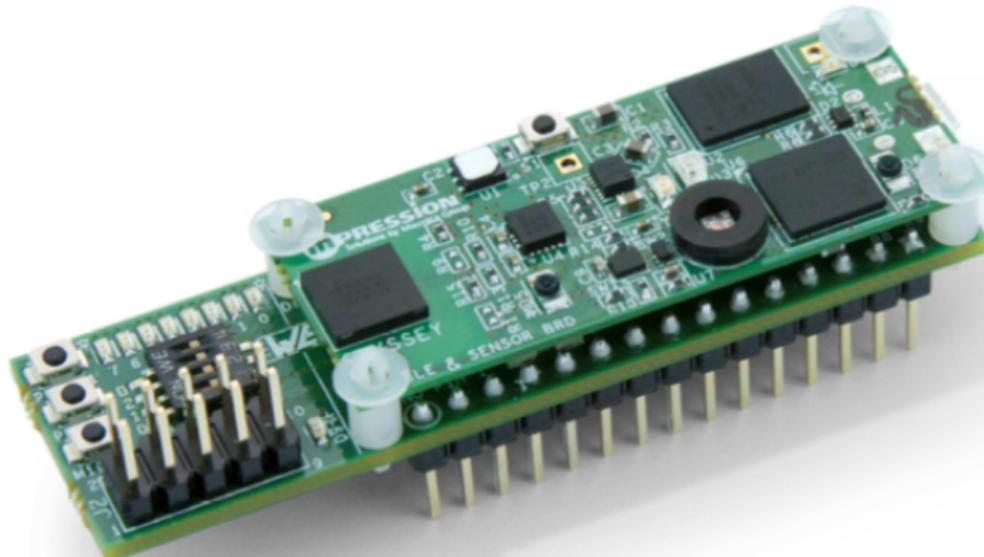
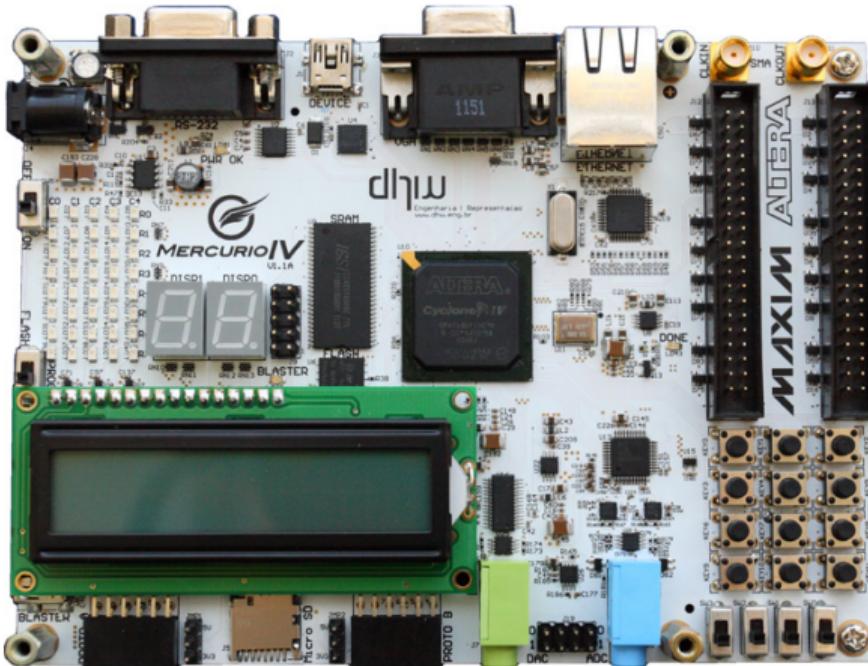
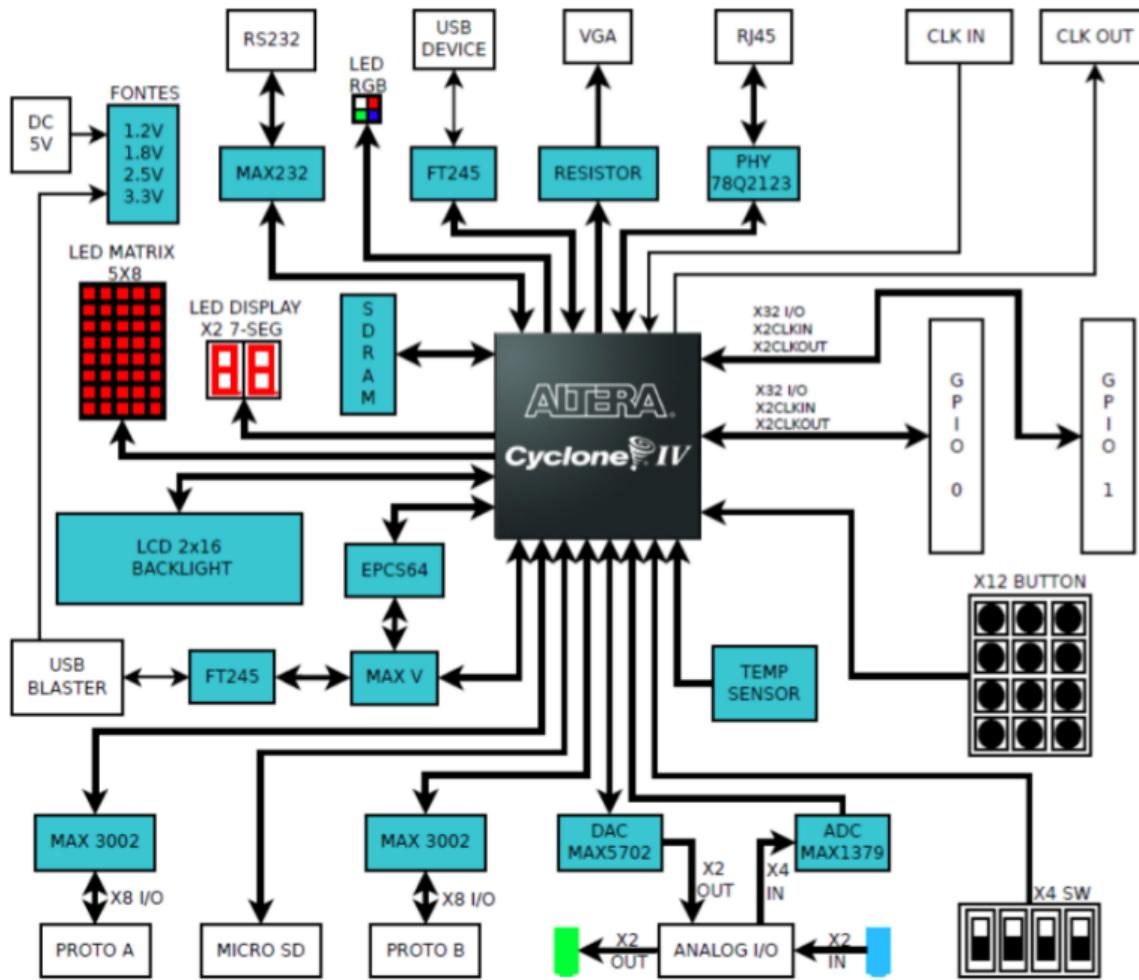


Figura 46: Odyssey IV.

Kit Mercurio IV

- Modelo EP4CE30F23 responsável possui cerca de 30 mil elementos lógicos, com 484 pinos de entrada e saída e utiliza *clock* de 50Mhz.





Kit Helio Cyclone V

- Fabricado em tecnologia de silício de 28nm.
- Este FPGA integra num único dispositivo um sistema em *hardware* baseado no processador ARM Cortex A9 dual-core e mais um grande conjunto de periféricos.
- Características:
 - Altera Cyclone V SoC – 5CSXFC6C6U23C8NES / 5CSXFC5C6U23C7N
 - HSMC expansion connector
 - DDR3-SDRAM
 - Micro SD Card
 - Interfaces: 1Gb Ethernet, USB OTG, UART...
 - Yocto Linux BSP

Kit Helio Cyclone V

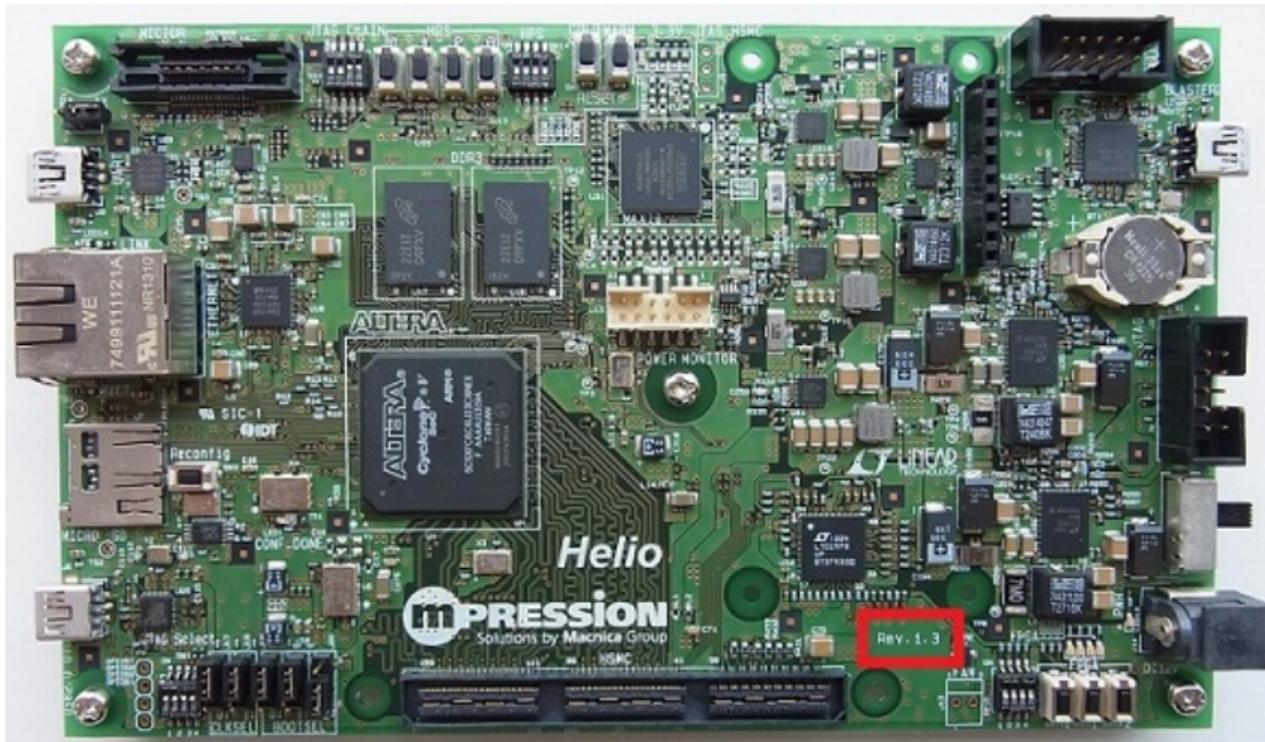
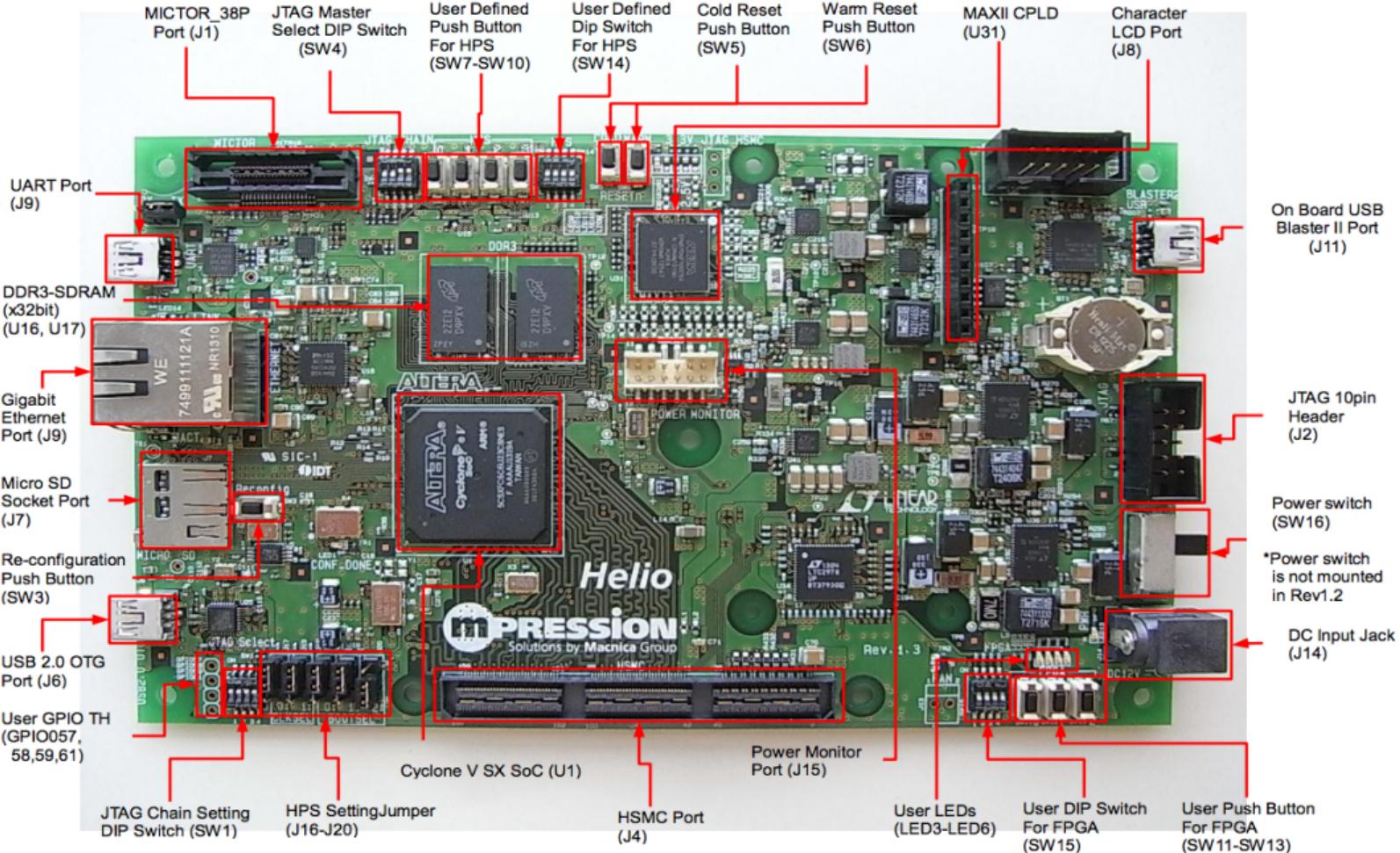
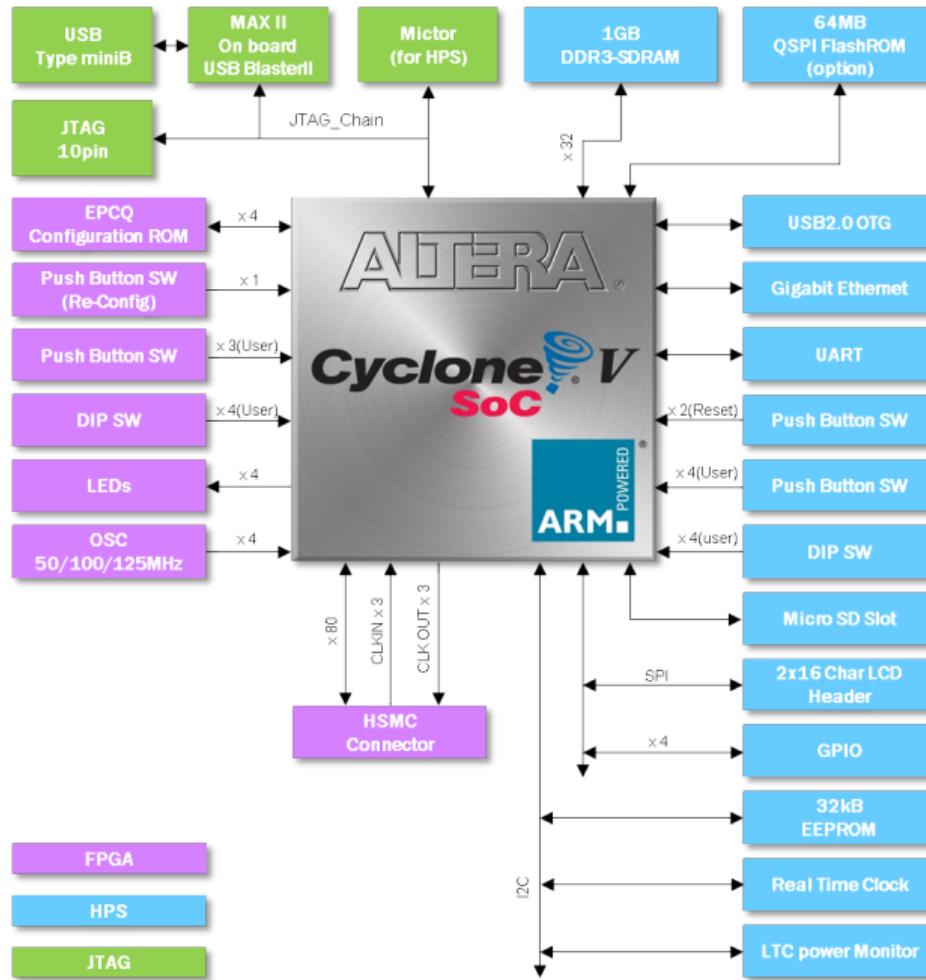


Figura 49: Componentes do Helio Cyclone V.





Arquitetura de Computadores

Dispositivos Lógico Programáveis (*Hardware Reconfigurável*)
Arranjo de Portas Programável em Campo - FPGA

Rodolfo Labiapari Mansur Guimarães

Última Atualização: 21 de junho de 2017

rodolfolabiapari@gmail.com

Lattes: <http://goo.gl/MZv4Dc>

Departamento de Computação – Universidade Federal de Ouro Preto
Ouro Preto - MG – Brasil



Bibliografia

 R. D' AMORE.

VHDL Descrição e Síntese de Circuitos Digitais.

Editora LTC, 2012.

 V. Moreira.

Plataforma reconfigurável para ensino e pesquisa em laboratório de sistemas digitais a distância.

Anais do Simpósio . . . , (Sbie):542–551, 2008.

 V. PEDRONI.

Eletrônica digital moderna e VHDL.

Campus, 1 edition, 2010.

 C. Roth.

Digital systems design using VHDL.

CL Engineering, 2 edition, 1998.

-  R. Sass and A. Schmidt.
Embedded Systems Design with Platform FPGAs Principles and Practices.
Morgan Kaufmann, 1 edition, 2010.
-  I. Skliarova and A. B. Ferrari.
Introdução à computação reconfigurável.
Electrónica e Telecomunicações, 4(1):103–119, 2012.
-  R. Tocci, N. Widmer, and G. Moss.
Sistemas digitais: princípios e aplicações.
Pearson, 11 edition, 2003.

Arquitetura de Computadores

Dispositivos Lógico Programáveis (*Hardware Reconfigurável*)
Arranjo de Portas Programável em Campo - FPGA

Rodolfo Labiapari Mansur Guimarães

Última Atualização: 21 de junho de 2017

rodolfolabiapari@gmail.com

Lattes: <http://goo.gl/MZv4Dc>

Departamento de Computação – Universidade Federal de Ouro Preto
Ouro Preto - MG – Brasil