Semana de Engenharia

Introdução a Linguagem de Descrição de HW VHDL a ao Projeto Baseado em Lógica de Programação ALTERA

Prof. Daniel Barros Júnior (dbarros@ee.pucrs.br)

Bolsista: Dárcio Prestes (darcio@ee.pucrs.br)

- Existe um grande número de dispositivos para a implementação de projetos lógicos que em sua maioria são caracterizados por serem de difícil modificação nas suas especificações e em geral exigem um reprojeto para efetivar as modificações

- Uma alternativa para obter flexibilidade em uma implementação consiste em usar módulos programáveis.

- Estes módulos tem uma estrutura padrão e são personalizados para uma função particular.

- Os módulos programáveis são mais caros e lentos.
- Porém tornaram-se bastante populares, devido a grande flexibilidade.

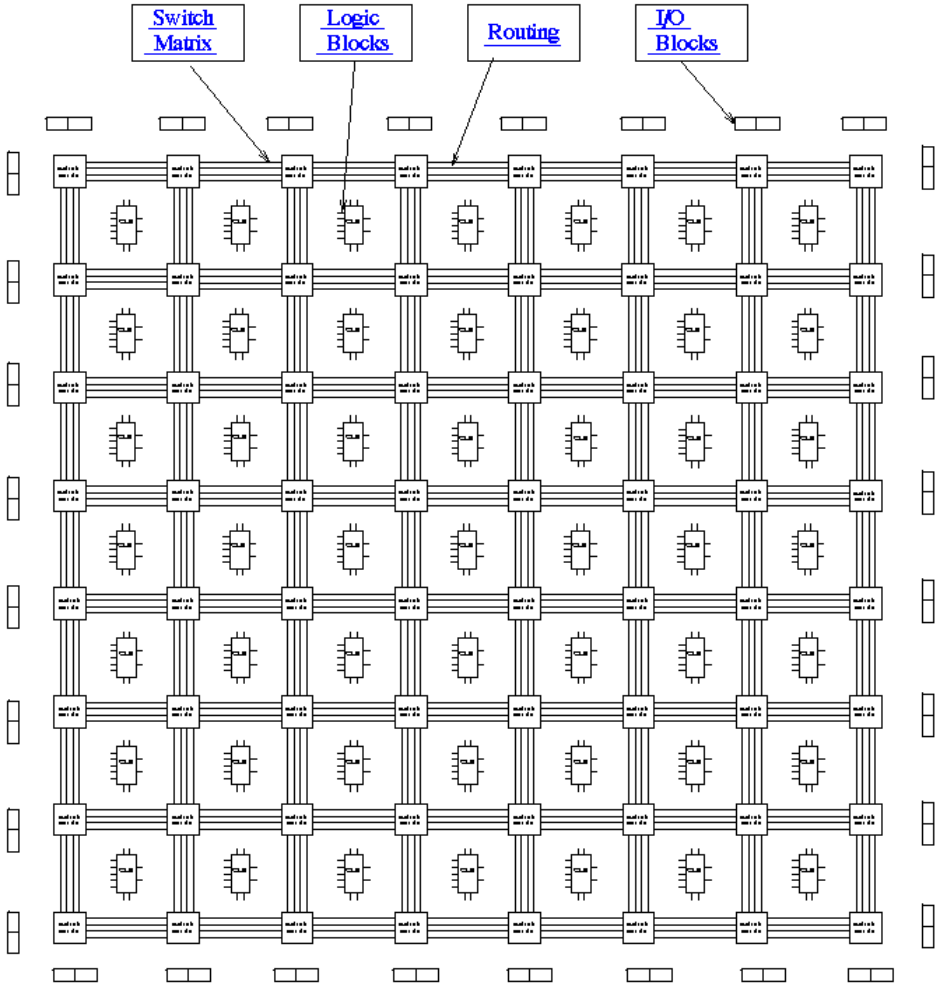
- Existem vários tipos de módulos programáveis tais como:
  - ROM - Memória Somente de Leitura
  - PAL - Matriz de Lógica Programável
  - CPLD - Dispositivos Lógicos Programáveis Complexos (Complex Programmable Logic Devices)
  - FPGA - Matriz de Portas Programáveis em Campo (Field Programmable Gate Array)

FPGAs e CPLDs

Grupo SISC Sistemas de Computação

- São módulos VLSI que podemos programar para implementar sistemas digitais com dezenas de milhares de portas.
- Os FPGAs consistem de três tipos de elementos:
  - Blocos Lógicos
  - Pontos de Interconexão
  - Blocos de Entrada e Saída
- Além disso, existem fios agrupados em canais horizontais e verticais.

Elementos do FPGA



- Dentro dos CPLDs e FPGAs tipicamente encontram-se múltiplas cópias.

Abordagens Básicas de Programabilidade

Grupo SISC Sistemas de Computação

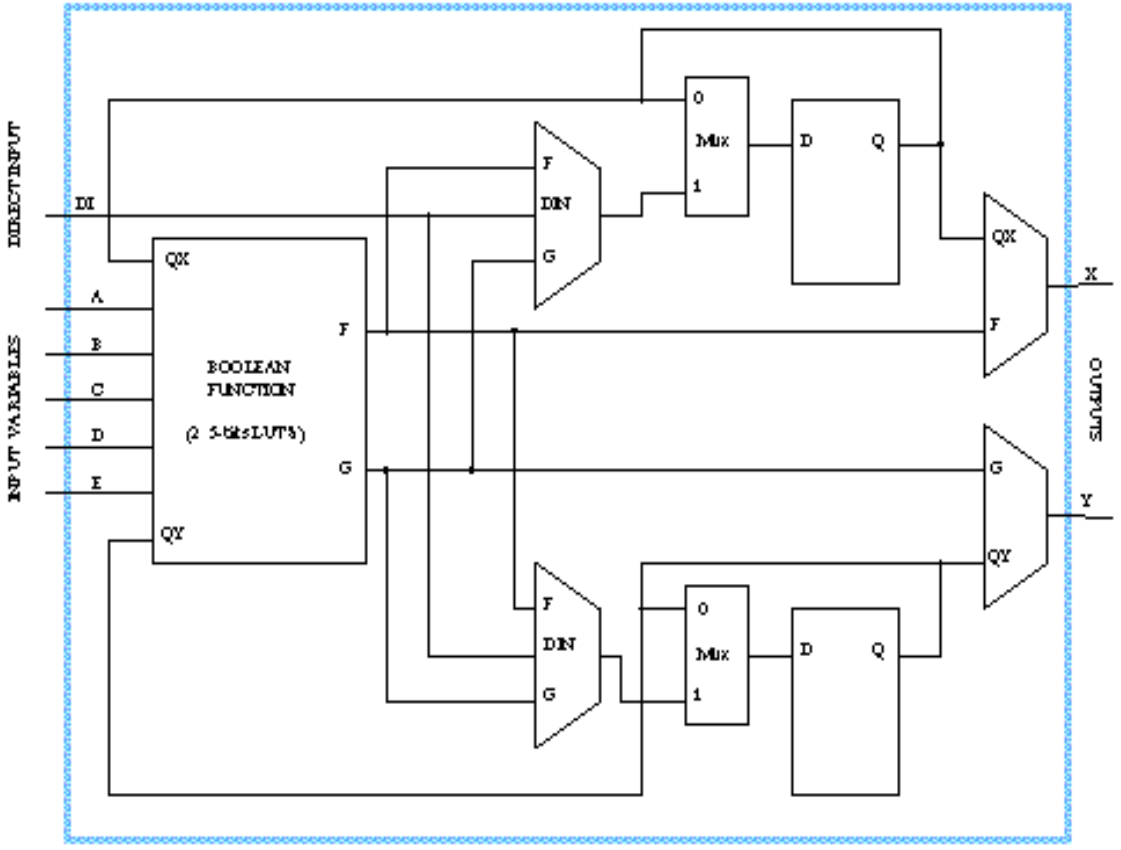
- FPGAs com memória RAM - Possuem uma maior capacidade mas perdem a programação quando desenergizados.
- FPGAs com memória PROM - Podem ser gravadas apenas uma vez, são mais rápidas do que as com RAM.
- FPGAs com EEPROM ou EPROM - não exigem memórias permanentes externas como as FPGAs com RAM, porém o processo de fabricação é mais complexo e os componentes tem menos capacidade.

Estrutura Interna de Um Bloco Lógico do FPGA

- A estrutura interna destes componentes varia conforme a família e conforme o fabricante.
- Mesmo a nomenclatura tem pequenas variações, vejamos abaixo:

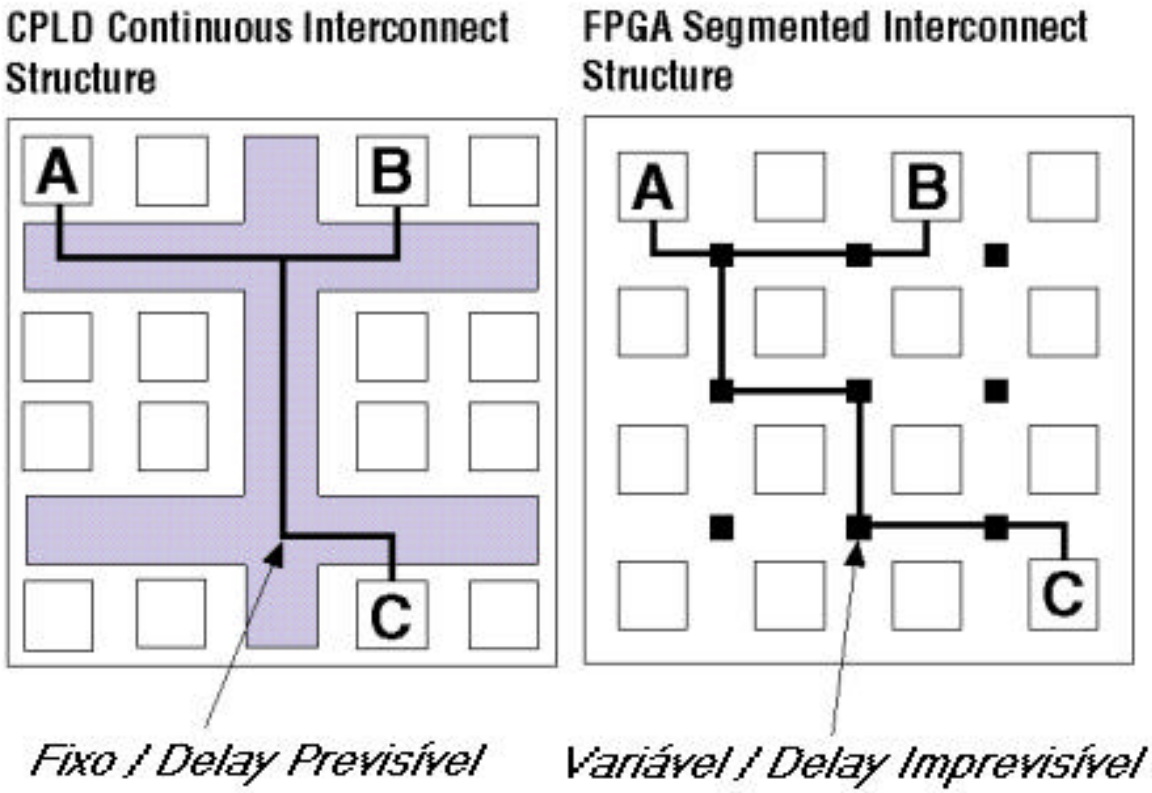
	XILINX	ALTERA
Bloco Lógico	CLB(Configurable Logic Block)	LE(Logic Element)
Pontos de Interconexão	SB (Switch Box)	--
Blocos de E/S	IOB (I/O Block)	IOE (I/O Element)
Memória RAM ou ROM	--	EAB (Embedde Array Block)

CLB da Família XC3000



- Diagrama em blocos do circuito interno de um CLB da família XC3000 da XILINX.

Roteamento



- A diferença básica entre os FPGAs e os CPLDs é mostrada na figura ao lado.

Técnicas de Programação

- Esquemático
- VHDL
- AHDL
- Handel C

O que é VHDL ?

Grupo SISC Sistemas de Computação

- **VHDL é uma forma de se descrever, através de um programa, o comportamento de um circuito ou componente digital.**
  - **V**ery High Speed Integrated Circuit (VHSIC)
  - **H**ardware
  - **D**escription
  - **L**anguage

Vantagens de Desvantagens de Utilizar VHDL

Grupo SISC Sistemas de Computação

- **Vantagens**
  - Código facilmente reutilizável;
  - Projeto independente da tecnologia;
  - Redução do tempo de projeto;
  - Eliminação de erros de baixo nível;
  - Portabilidade.

- **Desvantagens**
  - Hardware gerado é menos otimizado;
  - Nem todos os comandos são sintetizáveis.

Síntese

- Síntese é o processo de tradução e otimização baseado na tecnologia de componentes de biblioteca.
- VHDL foi inicialmente desenvolvido para simulação de circuitos digitais e não síntese.
- Existem vários circuitos descrito em VHDL que podem ser simulados porém devido as limitações das ferramentas de síntese não podem ser implementados.

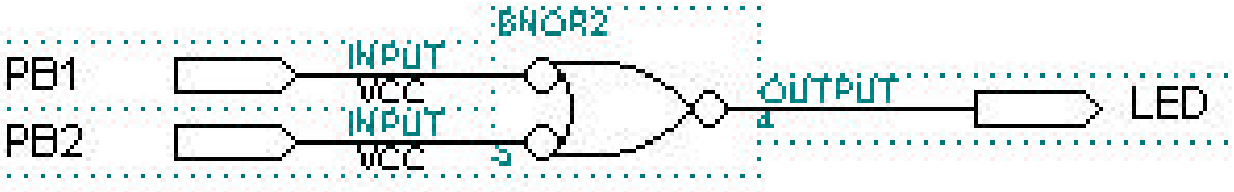
Estrutura de Um Projeto VHDL

Grupo SISC Sistemas de Computação

- **Projeto VHDL**
  - **Arquivos VHDL**
    - Package: Declara constante, tipos de dados, subprogramas. Objetivo: reutilização de código.
    - Entity: Declara as interfaces do projeto (pinos de entrada/saída).
    - Architecture: Define a implementação do projeto.
    - Configuration: Declara qual das arquiteturas será utilizada.

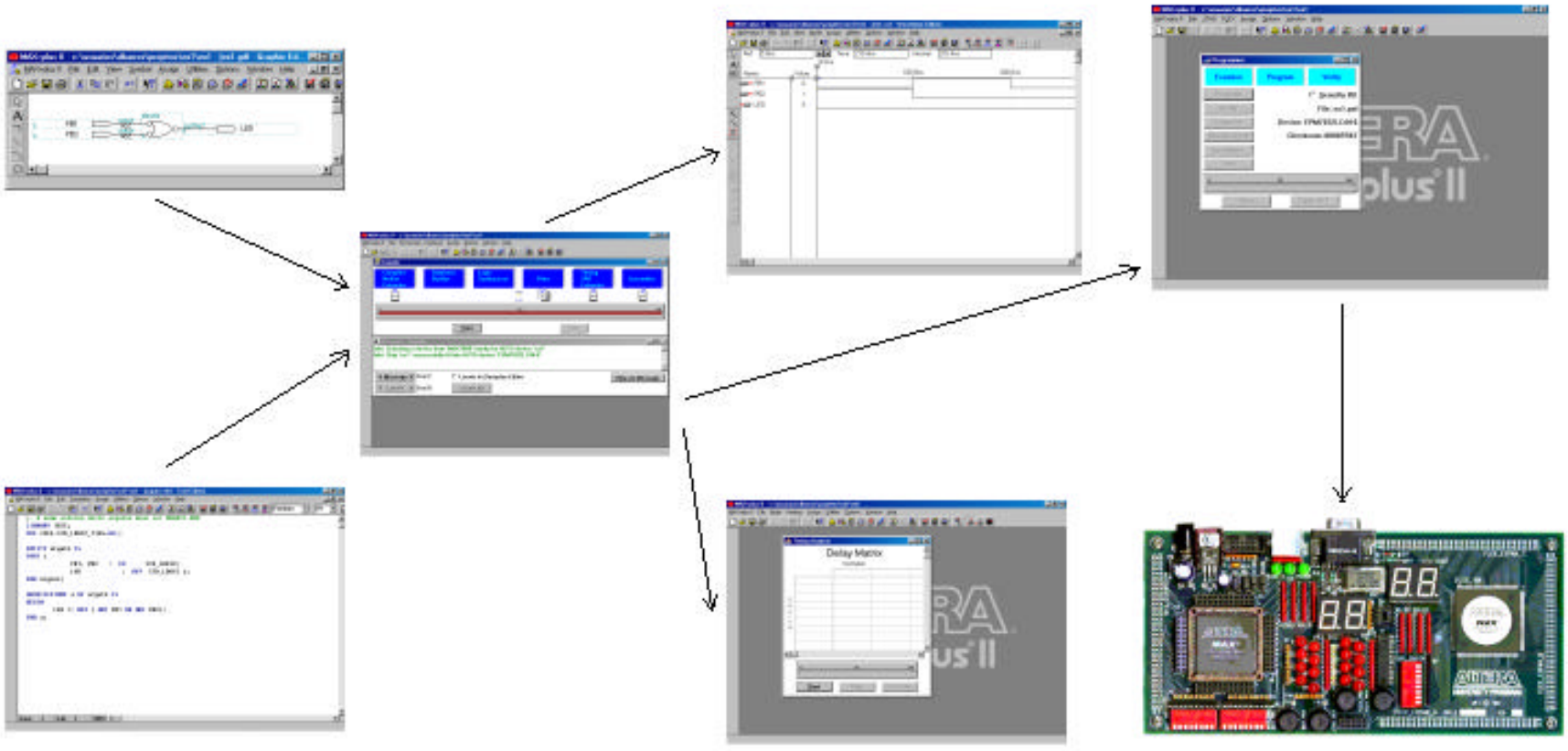
Programa Exemplo

```
-- O nome externo deste arquivo deve ser ORGATE.VHD
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY orgate IS
PORT (
        PB1, PB2   :  IN     STD_LOGIC;
        LED        :  OUT  STD_LOGIC );
END orgate;

ARCHITECTURE a OF orgate IS
BEGIN
     LED <= NOT ( NOT PB1 OR NOT PB2);
END a;
```



MAX-Plus II

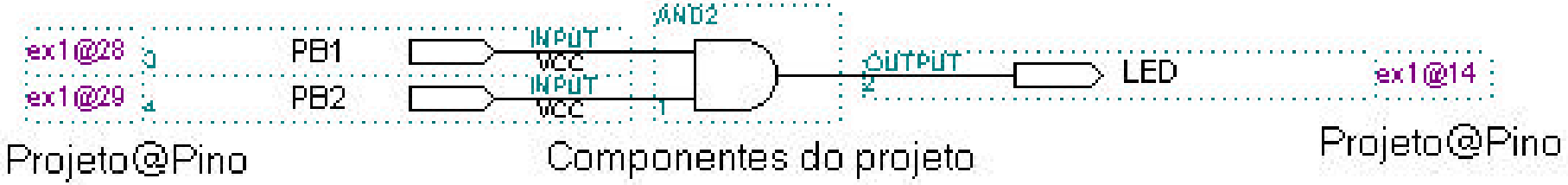
Projeto --> Compilação --> Simulação --> Verificação



Primeiro Projeto

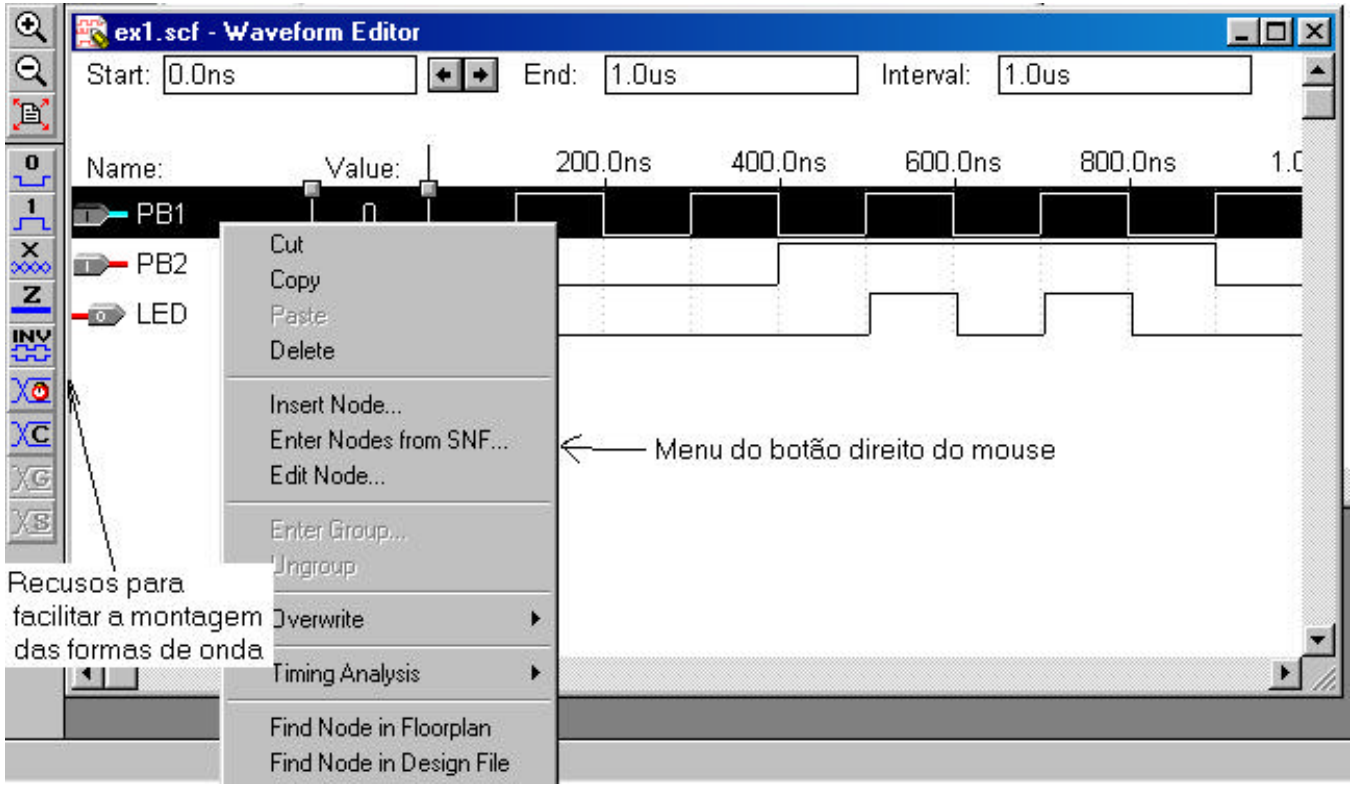
- Abrir o MAX-Plus II
- File -> New -> Graphic Editor -> OK
- Symbol -> Enter Symbol -> OK (AND2, OUTPUT, INPUT)
- Alterar nomes dos pinos Entradas PB1 e PB2, Saída LED
- File -> Save As -> EX1
- File -> Project -> Set Project to Current File
- File -> Project -> Save and Compile

- Assign -> Device -> EPF10K20RC240-4 -> OK
- Assign -> Pin/Location/Chip -> Search -> List -> Escolher um pino
- Ligar os pinos PB1 com 28, PB2 com 29 e LED com 14 (Usar botão ADD)



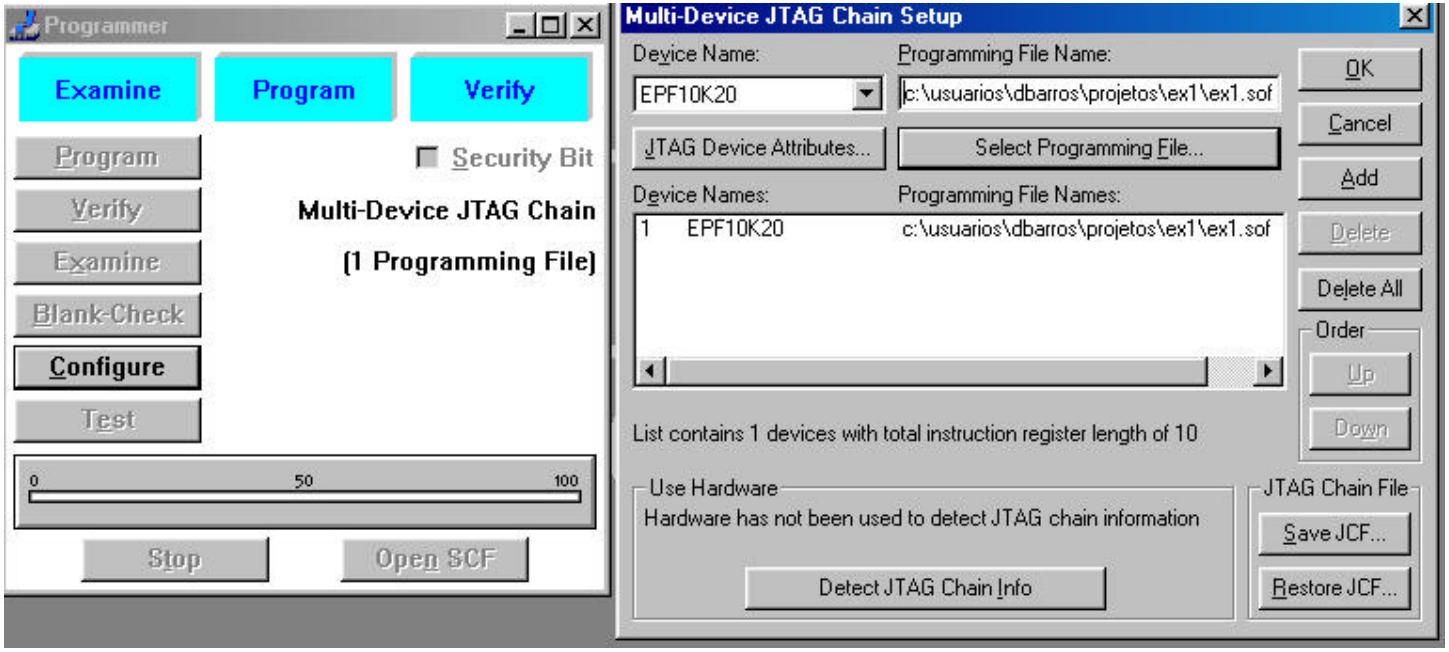
Simulação

- File -> New -> Waveform Editor File (scf)
- Node -> Enter Nodes from SNF -> List
- Montar formas de onda das entradas para a simulação



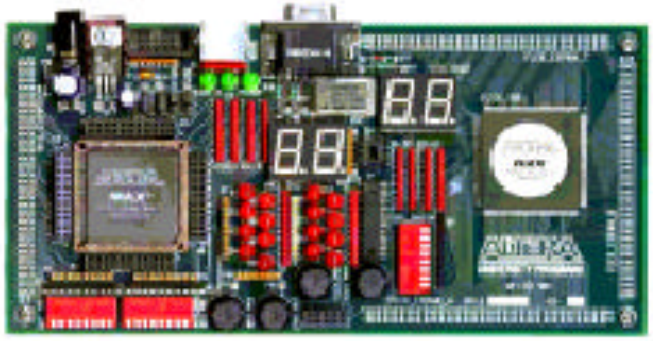
Download

- MAX-Plus II -> Programmer
- JTAG -> Multi-Device JTAG Chain
- JTAG -> Multi-Device JTAG Chain Setup
- Device name: EPF10K20
- Select Programming File (ex1.sof)
- Detect JTAG Chain Info



UP1 - University Program 1

- Possui 2 CPLDs
  - MAX EPM7128S 2.500 gates (eeprom)
  - FLEX EPF10K20 20.000 gates (sram)
- Clock de 25,175 MHz (pin 91)
- Conector de vídeo vga 640 x 480
- PS/2 mouse ou teclado
- chaves, botões, leds, displays



- Temos configurar para qual CPLD queremos fazer o download.
- Podemos conectar até 10 placas em série.

Mesmo exemplo em VHDL

- Abrir o MAX-Plus II
- File -> New -> Text Editor File -> OK
- Digitar o código VHDL ao lado
- Gravar com o nome de ex2.vhd
- File -> Project -> Set Project to Current File
- File -> Project -> Save and Compile

- Assign -> Device -> EPF10K20RC240-4 -> OK
- Assign -> Pin/Location/Chip -> Search -> List -> Escolher um pino
- Ligar os pinos PB1 com 28, PB2 com 29 e LED com 14 (Usar botão ADD)

- File -> New -> Waveform Editor File (scf)
- Node -> Enter Nodes from SNF -> List
- Montar formas de onda das entradas para a simulação

```
-- O nome externo do arquivo eh ex2.vhd
LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY ex2 IS
PORT (
        PB1, PB2   :  IN     STD_LOGIC;
        LED        :  OUT    STD_LOGIC );
END ex2;

ARCHITECTURE arquit OF ex2 IS
BEGIN
      LED <= PB1 and PB2;
END arquit;
```

Configurações Importantes

- MAX-Plus II -> Compiler
- Processing -> Fitter Settings -> Desmarcar “Use Quartus Fitter for Flex 10K ...”
- Interfaces -> VHDL Netlist Reader Settings -> VHDL Version: VHDL 1993
- Interfaces -> VHDL Netlist Writer Settings -> VHDL Version: VHDL 1993

Operadores VHDL

- \+ Adição
- \- Subtração
- * Multiplicação*
- / Divisão*
- & Concatena bits
- SLL deslocamento lógico para esquerda
- SRL deslocamento lógico para direita
- SLA deslocamento aritmético para esquerda
- SRA deslocamento aritmético para direita
- ROL rola para esquerda
- ROR rola para direita
- = igualdade
- /= desigualdade
- < , <= , > , >=
- Funções lógicas
  - NOT , AND, OR, NAND,NOR, XOR, XNOR

*** MAX-Plus II para potências de 2 (shift)**

Tipos de Dados

Grupo SISC Sistemas de Computação

- std_logic - usado para valores lógico
  - U - não utilizado
  - X - desconhecido
  - Z - tri-state ou alta impedância
  - W - fraco
  - L ou “0” - 0
  - H ou “1” - 1
  - - - don’t care
- std_logic_vector - um vetor de bits

Conversões de Dados

- **TO_STDLOGICVECTOR(vetor de bits) - Converte um vetor de bits em um std_logic_vector.**

TO_STDLOGICVECTOR(X"FFFF") - Gera um std_logic_vector de 16 bits

X para Hexadecimal e B para Binário

- **CONV_STD_LOGIC_VECTOR(inteiro, bits) - Converte um inteiro em um std_logic_vector.**

CONV_STD_LOGIC_VECTOR(7, 4) - Produz um std_logic_vector "0111"

- **CONV_INTEGER(std_logic_vector) - Converte um std_logic_vector em um inteiro.**

CONV_INTEGER("0111") - Produz um inteiro de valor 7

Display de 7 segmentos


```
library IEEE;
use  IEEE.STD_LOGIC_1164.all;
use  IEEE.STD_LOGIC_ARITH.all;
use  IEEE.STD_LOGIC_UNSIGNED.all;

ENTITY dec_7seg IS
      PORT( hex_digit     :  IN    STD_LOGIC_VECTOR(3 downto 0);
               segment_a, segment_b, segment_c, segment_d  :  out  std_logic;
               segment_e,  segment_f, segment_g                    :  out  std_logic);
END dec_7seg;
```

- Aqui temos as bibliotecas que são utilizadas e que em geral serão as mesmas.
- Na entidade definimos o nome do arquivo e as entradas e saídas com seus respectivos tipos de variáveis.

Display de 7 segmentos

```
ARCHITECTURE a OF dec_7seg IS
      SIGNAL segment_data : STD_LOGIC_VECTOR(6 DOWNTO 0);
BEGIN
PROCESS  (Hex_digit)
-- HEX to 7 Segment Decoder for LED Display
BEGIN
CASE Hex_digit IS
      WHEN "0000" => segment_data <= "1111110";
      WHEN "0001" => segment_data <= "0110000";
      WHEN "0010" => segment_data <= "1101101";
      WHEN "0011" => segment_data <= "1111001";
      WHEN "0100" => segment_data <= "0110011";
      WHEN "0101" => segment_data <= "1011011";
      WHEN "0110" => segment_data <= "1011111";
      WHEN "0111" => segment_data <= "1110000";
      WHEN "1000" => segment_data <= "1111111";
      WHEN "1001" => segment_data <= "1111011";
      WHEN "1010" => segment_data <= "1110111";
      WHEN "1011" => segment_data <= "0011111";
      WHEN "1100" => segment_data <= "1001110";
      WHEN "1101" => segment_data <= "0111101";
      WHEN "1110" => segment_data <= "1001111";
      WHEN "1111" =>  segment_data <= "1000111";
      WHEN OTHERS =>  segment_data <= "0111110";
END CASE;
END PROCESS;
```

```
-- extract segment data and LED driver is inverted
segment_a <= NOT segment_data(6);
segment_b <= NOT segment_data(5);
segment_c <= NOT segment_data(4);
segment_d <= NOT segment_data(3);
segment_e <= NOT segment_data(2);
segment_f <= NOT segment_data(1);
segment_g <= NOT segment_data(0);

END a;
```

- Na arquitetura definimos as váriaveis, sinais e processos.
- O comando CASE verifica o sinal hex_digit e coloca o valor correspondente em segment_data.
- Fora do processo as saídas são atualizadas.

ULA


```
LIBRARY IEEE;
USE  IEEE.STD_LOGIC_1164.all;
USE  IEEE.STD_LOGIC_ARITH.all;
USE  IEEE.STD_LOGIC_UNSIGNED.all;
ENTITY ALU IS
  PORT(-- Input Signals
       Op_code           : in std_logic_vector(2 DOWNTO 0);
       A_input, B_input  : in std_logic_vector(7 DOWNTO 0);
                    -- Output Signals
                    ALU_output      : out std_logic_vector(7 DOWNTO 0));
END ALU;

ARCHITECTURE behavior OF ALU IS
 -- declare signal(s) internal to module here
 SIGNAL temp_output: std_logic_vector(7 DOWNTO 0);

BEGIN
 PROCESS (Op_code, A_input, B_input)
  BEGIN
  -- Select Arithmetic/Logical Operation
       CASE Op_Code (2 DOWNTO 1) IS
      WHEN "00" => temp_output <= A_input + B_input;
      WHEN "01" => temp_output <= A_input - B_input;
      WHEN "10" => temp_output <= A_input AND B_input;
      WHEN "11" => temp_output <= A_input OR B_input;
      WHEN OTHERS => temp_output <= "00000000";
   END CASE;
   -- Select Shift Operation
   IF Op_Code(0) = '1' THEN
     -- Shift bits left with zero fill using concatination operator
     -- can also use VHDL 1076-1993 shift operator such as SLL
     Alu_output <= temp_output(6 DOWNTO 0) & '0';
   ELSE
     Alu_output <= temp_output;
   END IF;
  END PROCESS;
END behavior;
```

Como Obter o MAX-Plus II

- Entrar na página da Altera

  (www.altera.com)
- Escolher a opção University Program
  (www.altera.com/education/univ/unv-index.html)
- Escolher a opção Software

  (www.altera.com/education/univ/unv-software.html)
- Escolher a opção Download MAX+PLUS II Student Edition Software

  (www.altera.com/education/univ/unv-student_get.html)
- Entrar na primeira opção concordar com os termos de licença e fazer o download do software de aproximadamente 23 MBytes.

Como Obter a Licença do MAX-Plus II

- Entrar na página da Altera
  (www.altera.com)
- Escolher a opção University Program
  (www.altera.com/education/univ/unv-index.html)
- Escolher a opção Software
  (www.altera.com/education/univ/unv-software.html)
- Escolher a opção Download MAX+PLUS II Student Edition Software
  (www.altera.com/education/univ/unv-student_get.html)
- Entrar na segunda opção e escolher MAX+PLUS II Student Edition software Version 9.23 e Clicar em Continue
  (www.altera.com/support/licensing/lic-university.html).
- Digite o número do volume de seu HD 224f-15e7 (este número pode ser obtido digitando dir /p em modo MSDOS)
  (www.altera.com/cgi-bin/authcode91.pl)
- Na página seguinte preencha seus dados e a Altera que enviará em pouco tempo um arquivo licence.dat

Bibliografias

- HAMBLEN, James O. e FURMAN, Michael D., Rapid Prototyping of Digital Systems, KAP, 2000, com CD
- CHANG, K.C., Digital Design and Modeling with VHDL and Synthesis, IEEE, 1997, http://computer.org/books/bp07716/index.html
- ERCEGOVAC, Milos, LANG, Tomás e MORENO, Jaime H., Introdução aos Sistemas Digitais, Bookman, 1999, com CD
- TERROSO, Anderson R., Dispositivo Lógico Prog.(FPGA) Ling. de Descr. de HW (VHDL)
- MORAES, Fernando, Ling. de Descr. de HW VHDL
- SCARPINO, Frank, VHDL and AHDL Digital System Implementation, Prentice Hall, http://www.phptr.com/scarpino