

## MCTA024 – Sistemas Digitais – 2017-2a

José Artur Quilici-Gonzalez – CMCC - UFABC

### Lab 3 – Placa DE1 da Altera

Nesta experiência, vamos aprender a criar uma **configuração** para o projeto de um **Contador Binário**, baixar a configuração para a placa **DE1** da **Altera**, e depois fazer o **teste** do circuito.

### Contador Binário

Primeiramente crie uma pasta com seu nome, e dentro dela crie um novo projeto no **Quartus II** cuja **“entity”** receba o nome de **“ContadorBinario”** (sem aspas e sem acento em Binário).

Como agora vamos testar o circuito em uma placa de *hardware* (e não apenas uma simulação virtual), é muito importante especificar exatamente o código da **FPGA** da placa. Para a placa **DE1** que temos em nossos labs, a **FPGA** é da família **“Cyclone II”**, e o dispositivo (**“device”**) é o **“EP2C20F484C7”**, conforme ilustra a Fig. 1.

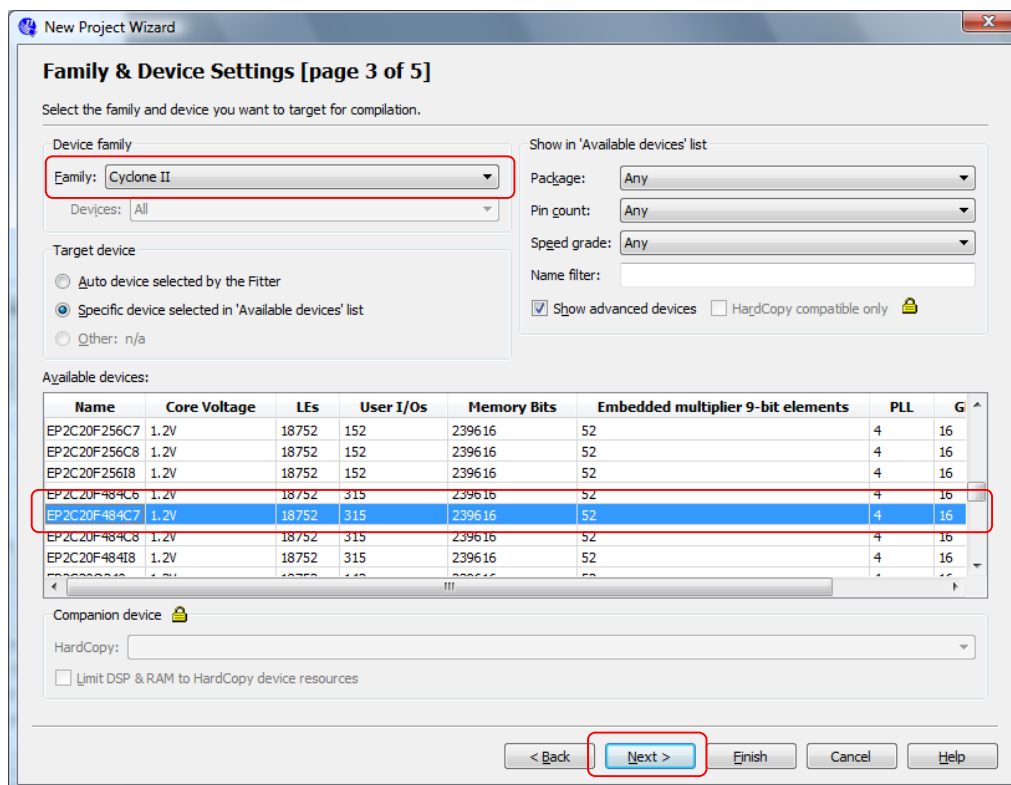


Fig. 1 – Escolha da FPGA “Cyclone II”, com o código “EP2C20F484C7”.

A seguir tome a opção **“Next”** desta página e das próximas até que o novo projeto seja criado.

Vá em **"File"** e tome a opção **"New..."**, escolha a opção **"VHDL File"** e dê um **"OK"**. Para não perdermos tempo com digitações, o arquivo **"ContadorBinario.vhd"** foi anexado à pasta **Lab3** desta experiência. Basta copiá-lo para o editor de VHDL do **Quartus II** e salvar com o nome **"ContadorBinario"** e extensão do tipo **"vhd"**.

Agora vamos fornecer ao **Quartus II** o arquivo que atribui nomes aos elementos físicos da placa DE1, como as chaves, os LEDs etc. Clique em **"Assignments"**, depois em **"Import Assignments"** e quando aparecer o quadro **"Import Assignments"**, na opção **"File name:"** carregue o arquivo **"DE1.qsf"**, também anexado na pasta **Lab3**, conforme ilustra a Fig. 2.

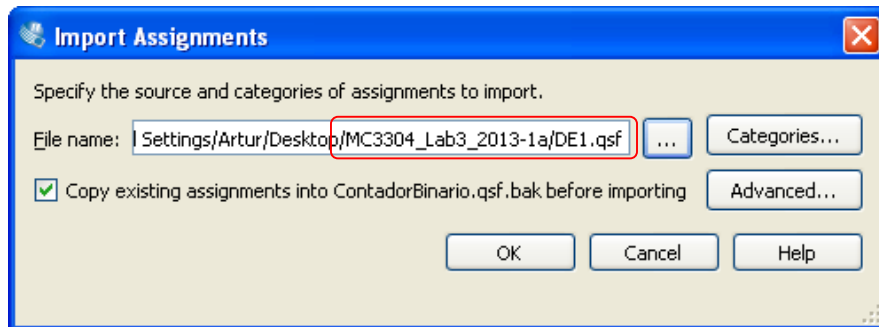


Fig. 2 – O arquivo **"DE1.qsf"** atribui nomes aos elementos físicos da placa DE1.

Depois destas etapas, podemos fazer a compilação do projeto. Dê um clique no triângulo roxo (**"Start Compilation"**) que aparece na parte superior do **Quartus II** (veja Fig. 3).

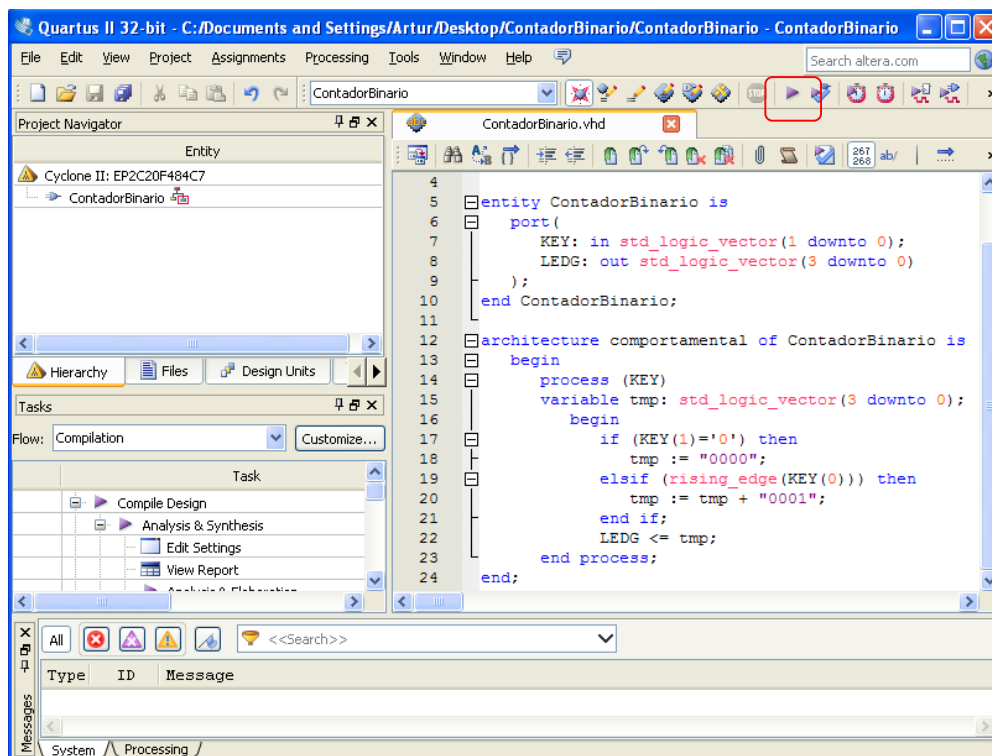


Fig. 3 – Para compilar o projeto, clicar no triângulo roxo (**"Start Compilation"**).

Inicialmente poderá aparecer a mensagem **“Data not available”**, mas ela logo desaparecerá. Se não houver nenhum erro no projeto, a mensagem de **“Full Compilation was successful (453 warnings)”** deve aparecer (Fig. 4). Clique no botão **“OK”**.

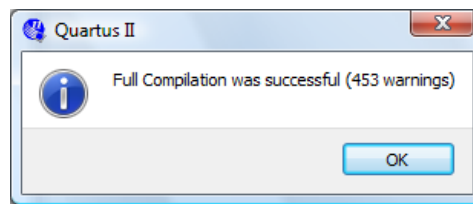


Fig. 4 – Depois da compilação devem aparecer muitos **“warnings”**, mas nenhum **“Error”**.

Faça a conexão entre a placa DE1 e o PC através do cabo USB, e ligue a placa apertando (com cuidado) o botão vermelho próximo ao cabo. Se os LEDs se acenderem e os **“displays seven segments”** iniciarem uma contagem crescente, a placa está funcionando bem.

Vamos agora “programar” ou “configurar” a FPGA, isto é, criar um programa para injetar 1s e 0s na FPGA através de um caminho serial chamado **“JTAG”**. Na parte esquerda do **Quartus II**, na seção **“Task”**, dê dois cliques na opção **“Program Device (Open Programmer)”**, como ilustra a Fig. 5.

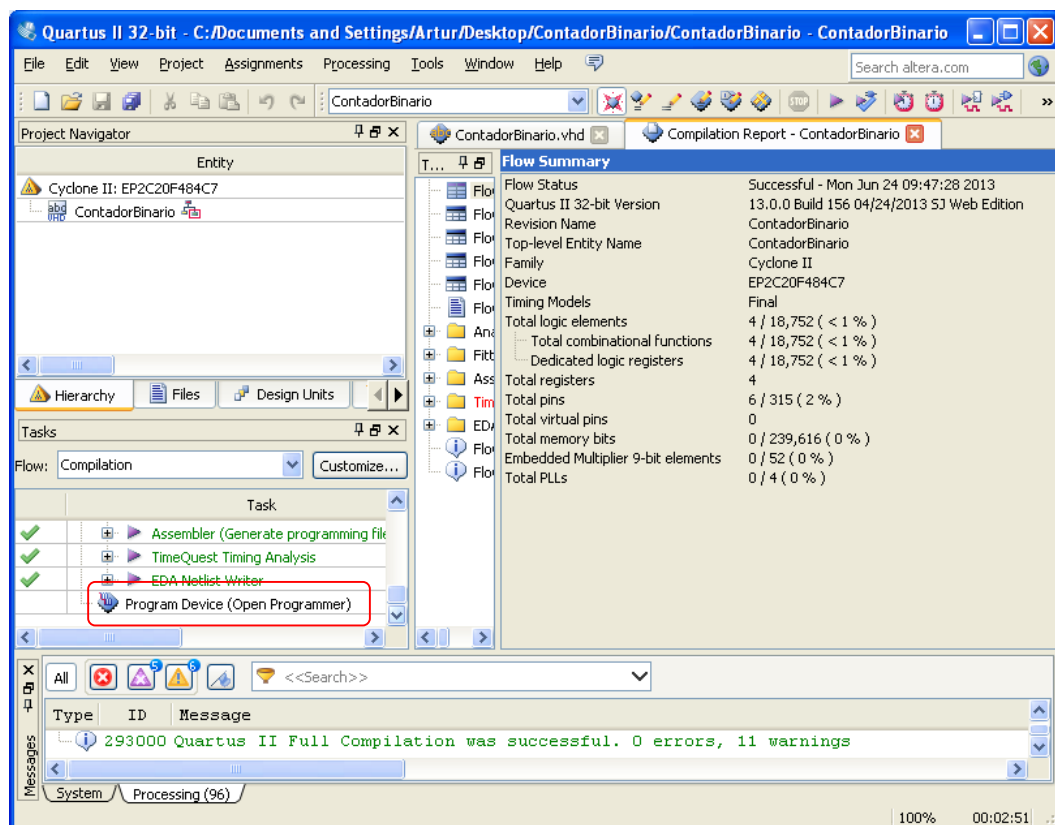


Fig. 5 – Depois da compilação, é possível criar a configuração para a FPGA.

A tela do **“Programmer”** deve aparecer (Fig. 6). Certifique-se que as opções **“USB-Blaster”** e **“Mode: JTAG”** estejam ajustadas.

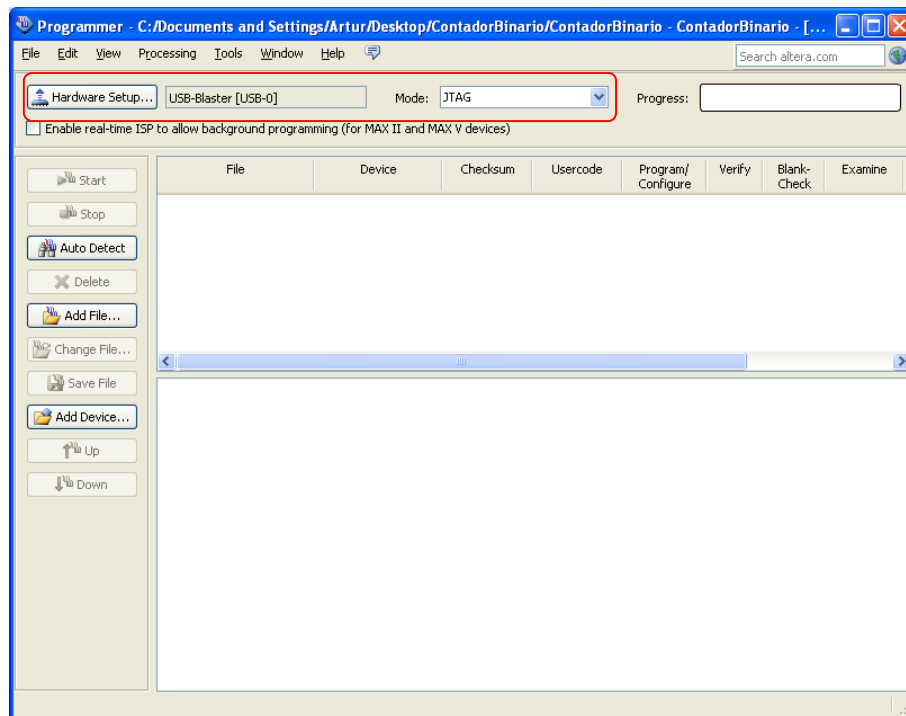


Fig. 6 – A tela do programador (“Programmer”) da FPGA.

Caso não apareça a opção **“USB-Blaster [USB-0]”** (na primeira vez que a placa **DE1** for ligada), clique em **“Hardware Setup ...”** e, na janela de **“Hardware Setup”** (Fig. 7), ajuste a opção **“USB-Blaster [USB-0]”** no campo **“Currently selected hardware”**. Depois, clique em **“Close”**.

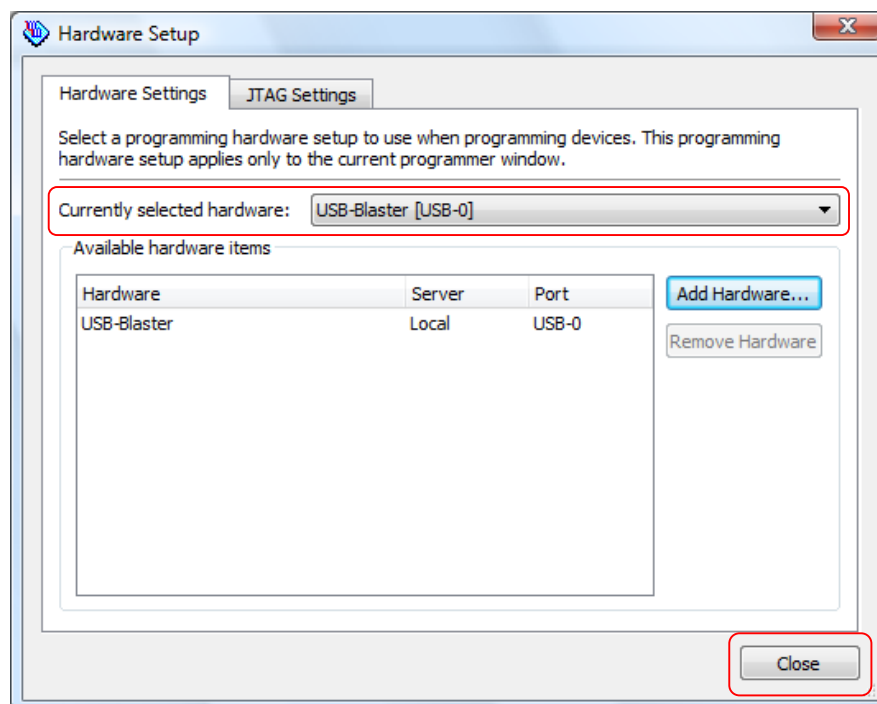
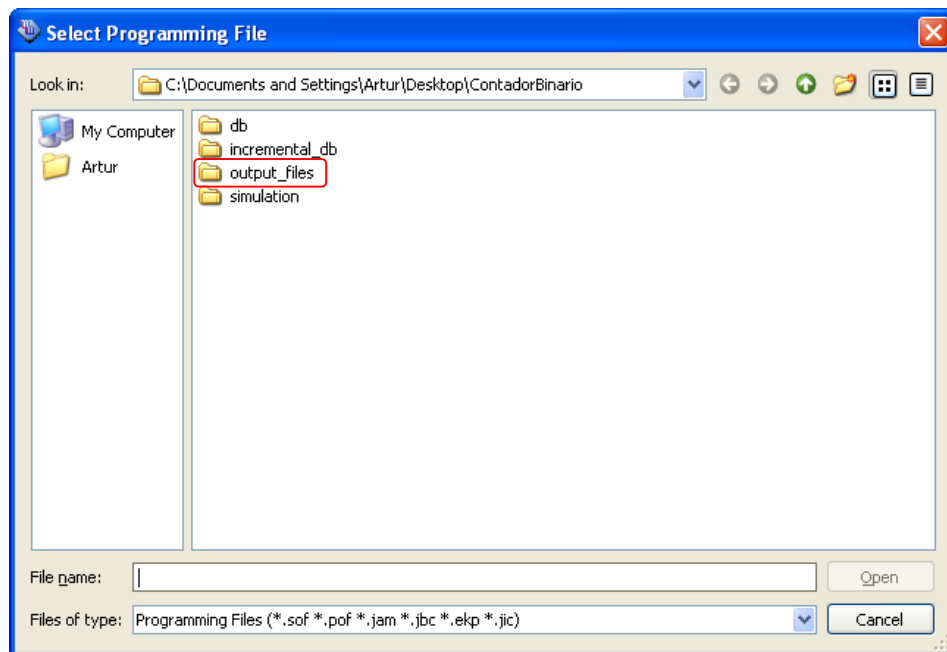


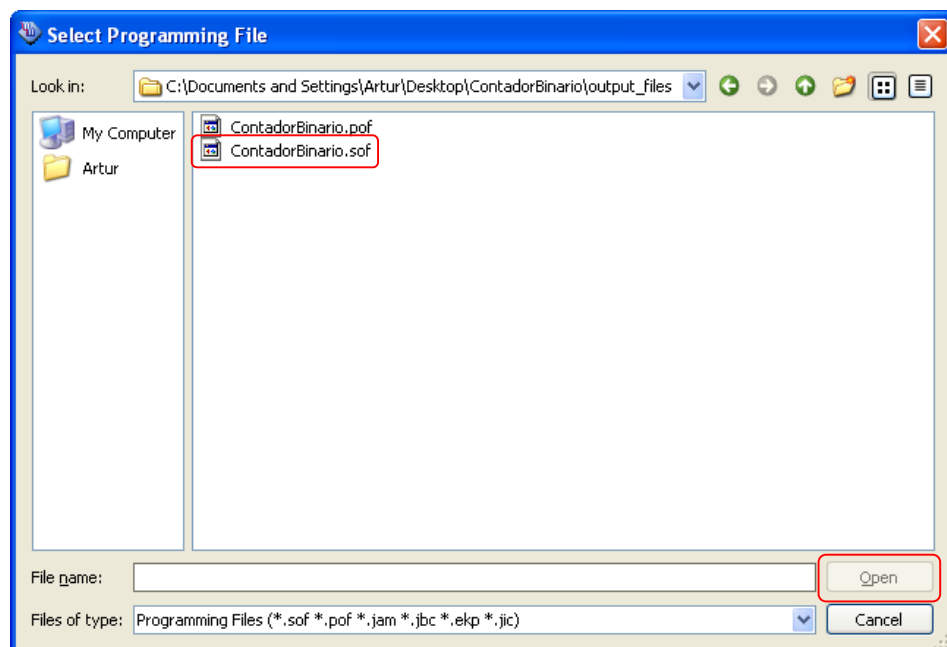
Fig. 7 – Ajuste do hardware **“USB-Blaster [USB-0]”**

Voltando à tela da Fig. 6, clique em **“Add File”** e, na tela subsequente que se abre, escolha **“output\_files”** (Fig. 8) e **“ContadorBinario.sof”**-(Fig. 9).



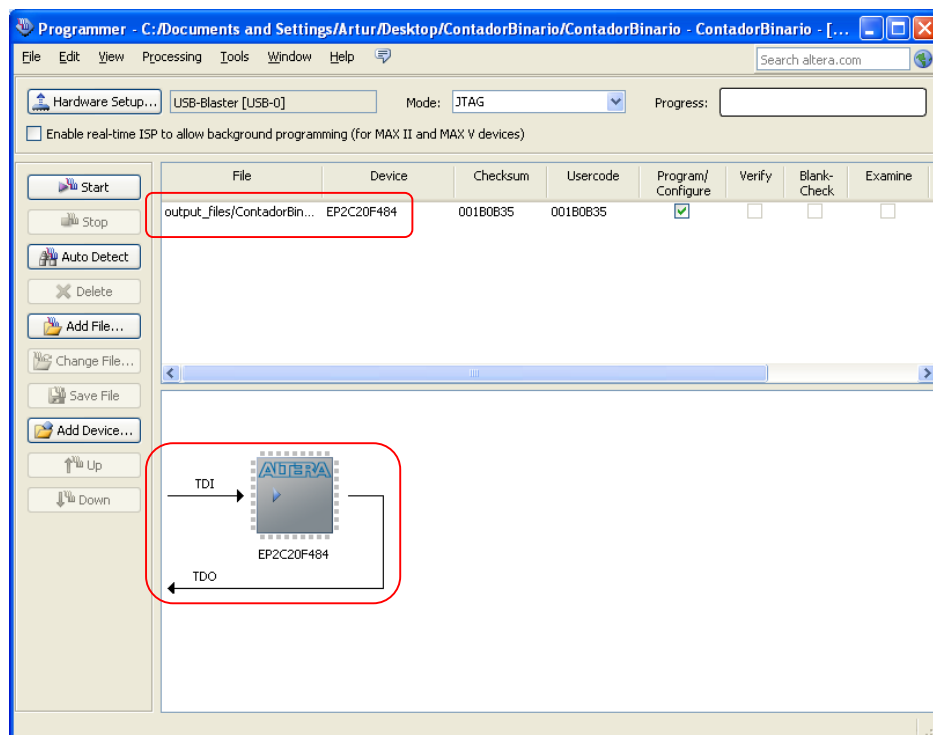
**Fig. 8 – Os arquivos de configuração gerados estão em “output\_files”.**

Depois de escolher **“ContadorBinario.sof”**, dê um clique em **“Open”** (Fig. 9).



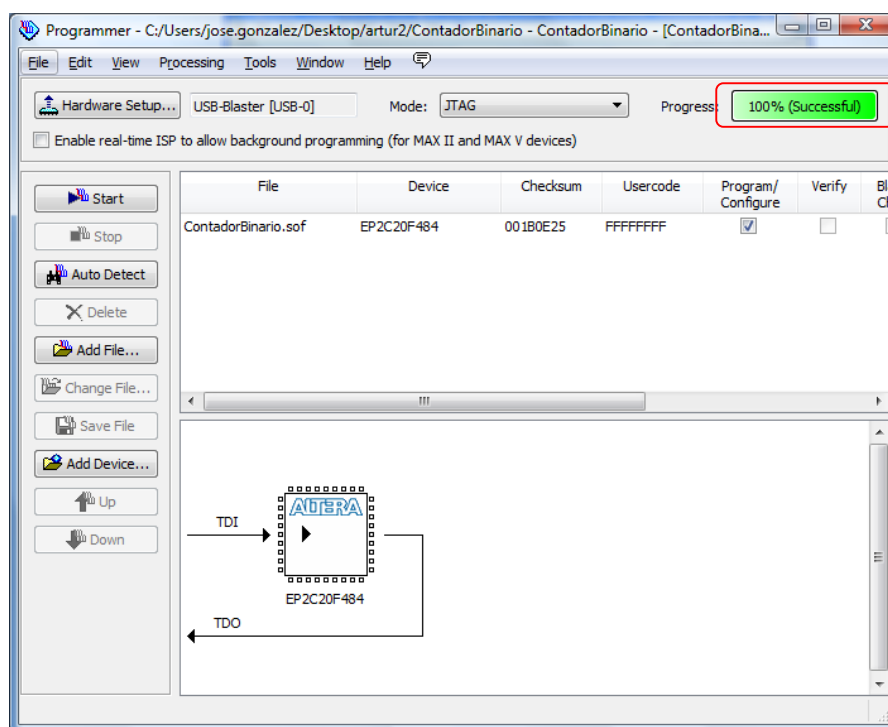
**Fig. 9 – O arquivo de configuração da FPGA é o “ContadorBinario.sof”.**

Depois de escolher **“ContadorBinario.sof”** e de dar um clique em **“Open”** (Fig. 9), devem aparecer o arquivo e o dispositivo da Altera (Fig. 10).



**Fig. 10 – Dispositivo da Altera foi detectado através do cabo USB.**

Com a placa “DE1 Altera” ligada, dê um clique em “Start”. Depois de algum tempo deve aparecer o aviso “*Progress: 100% (Successful)*” (Fig. 11).



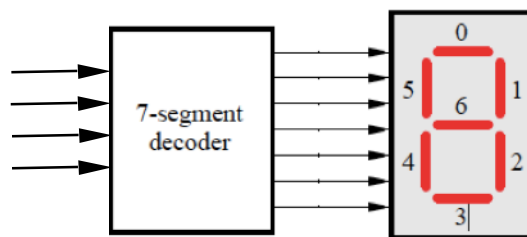
**Fig. 11 – O aviso “*Progress: 100% (Successful)*” indica que a configuração está terminada.**

Feche a janela da Fig. 11 (não há necessidade de salvar). Agora o circuito já pode ser testado na placa **DE1** através das chaves, LEDs, display sete segmentos etc. Ao apertar sucessivamente a chave **"KEY0"**, os LEDs verdes devem começar a acender em ordem binária crescente. A chave **"KEY1"** funciona como **"Reset"** do contador.

### **Seven Segment Display**

Uma forma simples de acionar o **"Seven Segment Display"** (Fig. 12) da placa **DE1** é através de um **decodificador**, como mostrado na Fig. 13. (Este código VHDL foi adaptado do livro **FPGA Prototyping by VHDL Examples**, Pong P. Chu, originalmente concebido para ser implementado numa placa da Xilinx).

O decodificador para o **"Seven Segment Display"** recebe 4 bits em suas entradas e os decodifica em 7 bits, que acionam o *display* (Fig. 12). Cada um dos segmentos recebe uma numeração de 0 a 6, e são ativos em "0", ou seja, ele se apaga quando recebe "1" no respectivo segmento.



**Fig. 12 – Decodificador e "Seven Segment Display".**

Na placa DE1 da Altera há 4 displays, identificados como HEX0, HEX1, HEX2 e HEX3, sendo cada um deles declarado como um **"HEX0: out std\_logic\_vector(0 to 6).**

No código VHDL mostrado na Fig. 13, o vetor foi declarado em ordem inversa (ou seja, **6 downto 0**), desta forma introduzindo propositalmente "erros" para que você possa corrigi-los e entender como é acionado um display de sete segmentos. Os primeiros seis algarismos estão corretamente decodificados, mas o restante precisa ser corrigido.

```

1  -- Baseado no código de FPGA Prototyping by VHDL Examples - Pong P. Chu
2  -- Adaptacao para a placa DE1 da Altera
3  library ieee;
4  use ieee.std_logic_1164.all;
5
6  entity SevenSegment is
7  port(
8      SW: in std_logic_vector(3 downto 0);
9      --dp: in std_logic;
10     HEX0: out std_logic_vector(6 downto 0)
11 );
12 end SevenSegment;
13
14 architecture arch of SevenSegment is
15 begin
16     with SW select
17         HEX0(6 downto 0) <=
18             "1000000" when "0000",
19             "1111001" when "0001",
20             "0100100" when "0010",
21             "0110000" when "0011",
22             "0011001" when "0100",
23             "0010010" when "0101",
24             "0100000" when "0110",
25             "0001111" when "0111",
26             "0000000" when "1000",
27             "0000100" when "1001",
28             "0001000" when "1010", --a
29             "1100000" when "1011", --b
30             "0110001" when "1100", --c
31             "1000010" when "1101", --d
32             "0110000" when "1110", --e
33             "0111000" when "1111", --f
34             --sseg(7) <= dp;
35 end arch;

```

Fig. 12 – Código VHDL simplificado para o “Seven Segment Display” da placa DE1.

Corrija o código da Fig. 12 para que apareça no display *HEX0* os 16 caracteres hexadecimais, ou seja, 0, 1, 2, ..., 9, A, b, C, d, E, e F. (Para não gastar tempo com digitação, o código “SevenSegment.vhd” foi anexado a esta pasta Lab3).

## Contador Hexadecimal

Fazendo com que a saída do **Contador Binário** seja a entrada do “Seven Segment Display”, é possível implementar um **Contador Hexadecimal**. Para isso, é preciso alterar a saída do **Contador Binário** (i.e., desconectá-la dos LEDGs) e a entrada do “Seven Segment Display” (i.e., desconectá-la das chaves SW) e conectar ambos através de um “signal”.

Implemente o **Contador Hexadecimal** de tal forma que pressionando-se *KEY0*, o display *HEX0* mostre todos os caracteres hexadecimais em ordem crescente.



## Bibliografia

Chu, P. P. *FPGA Prototyping para VHDL Examples*. Hoboken, NJ, John Wiley & Sons, Inc., 2008.

Altera. *Laboratory Exercise 1*. In

[ftp://ftp.altera.com/up/pub/Altera\\_Material/11.1/Laboratory\\_Exercises/Digital\\_Logic/DE1/vhdl/lab1\\_VHDL.pdf](ftp://ftp.altera.com/up/pub/Altera_Material/11.1/Laboratory_Exercises/Digital_Logic/DE1/vhdl/lab1_VHDL.pdf) . Acessado em 28.04.14.