



FEDERAL UNIVERSITY
OF SANTA CATARINA

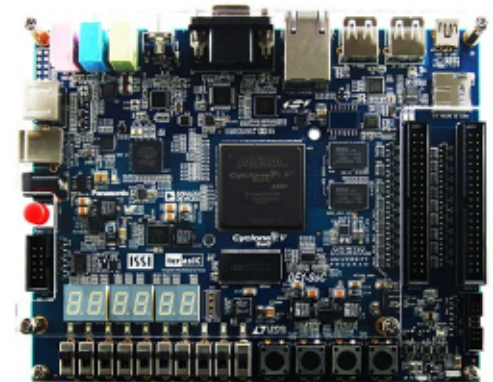
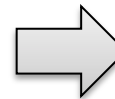
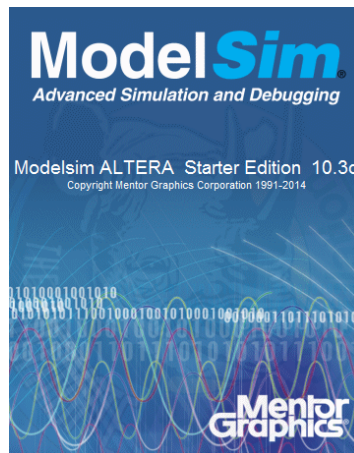
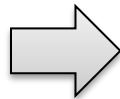
Laboratório Introdução a VHDL

Parte 1: Introdução à VHDL e Modelsim

EEL7123/EEL510457 – Tópico Avançado em Sistemas Digitais/Circuitos
Aritméticos

Objetivos

- Apresentar uma visão geral sobre **VHDL** e estudar **exemplos de descrição de hardware** em VHDL
- Familiarização com o simulador utilizado na disciplina **ModelSim** e **emulador** de placa **FPGA**.
- Compreensão do fluxo de projeto de sistemas digitais: Projeto, **simulação** e **emulação** na placa **FPGA**.



Introdução à VHDL

Uso de Quartus e Modelsim

Emulação na placa DE2

Introdução à VHDL

- **VHDL - Visão Geral**
 - **VHDL** é uma linguagem para descrição de hardware
 - **VHDL** = **V**HSIC **H**ardware **D**escription **L**anguage
 - No final da década de 80, **VHDL** se tornou uma linguagem padrão para o **IEEE** (*Institute of Electrical and Electronic Engineers*).
 - Existem diversas ferramentas para simular e sintetizar (gerar hardware) circuitos descritos em **VHDL**.
 - Outras linguagens de descrição de hardware: Verilog, SystemC, AHDL, Handel-C, System Verilog, Abel, Ruby, ...

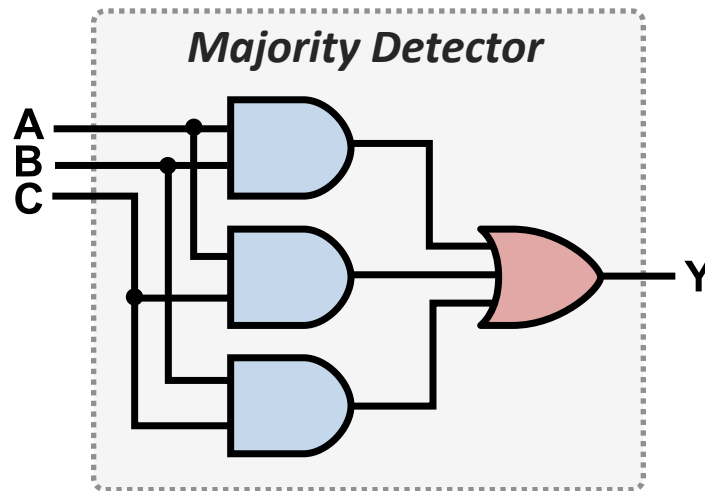
Introdução à VHDL

- **VHDL - Visão Geral**

- O projeto de um circuito digital pode ser descrito em **VHDL** em diversos níveis de abstração (ex.: **estrutural**, **comportamental**).
- Descrições em VHDL podem ser utilizadas para gerar **hardware** (arquivo para configuração de um FPGA, por exemplo).
- Descrições em VHDL podem ser **simuladas** (executadas em um simulador).
- A geração de estímulos para simulação VHDL é realizada por intermédio de **testbenches**. Um **testbench** define os estímulos externos a serem utilizados como entrada para o circuito.

Introdução à VHDL

- VHDL – Exemplo de código: *Majority Detector*



Introdução à VHDL

- VHDL – *Majority Detector*

LIBRARIES

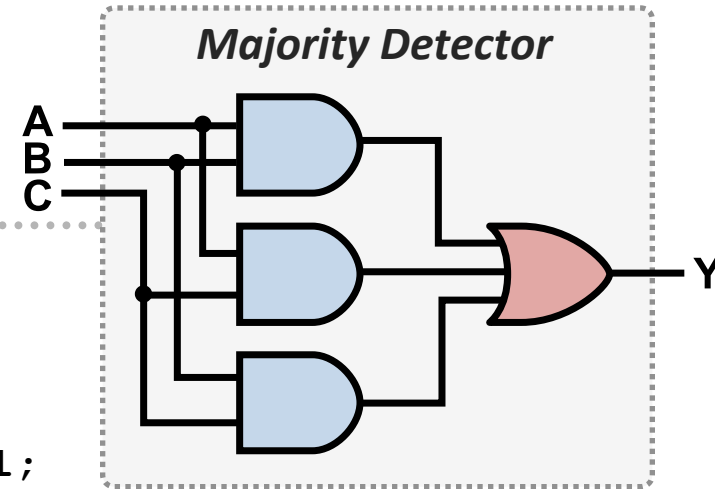
```
library IEEE;  
use IEEE.Std_Logic_1164.all;
```

ENTITY

```
entity majority is  
  port (A: in std_logic;  
        B: in std_logic;  
        C: in std_logic;  
        Y: out std_logic  
        );  
end majority;
```

ARCHITECTURE

```
architecture circuito_logico of majority is  
  signal D,E,F: std_logic;  
begin  
  Y <= D or E or F;  
  D <= A and B;  
  E <= A and C;  
  F <= B and C;  
end circuito_logico;
```



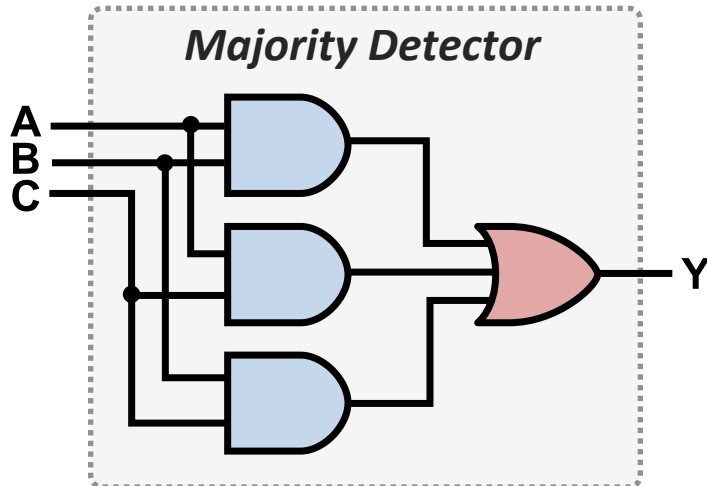
Introdução à VHDL

- VHDL – *Majority Detector*
 - **LIBRARIES** : bibliotecas necessárias

```
library IEEE;  
use IEEE.Std_Logic_1164.all;
```


Introdução à VHDL

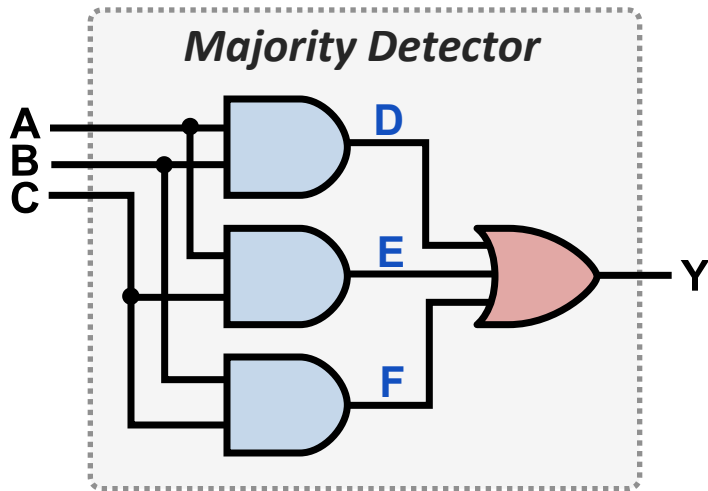
- VHDL – *Majority Detector*
 - **ENTITY** : define os **pinos** do circuito digital, ou seja, a **interface** entre a lógica implementada e o mundo externo.



```
entity majority is
port (A: in std_logic;
      B: in std_logic;
      C: in std_logic;
      Y: out std_logic
      );
end majority;
```

Introdução à VHDL

- VHDL – *Majority Detector*
 - **ARCHITECTURE** : define a funcionalidade do circuito digital, utilizando os **pinos** de entrada e saída listados na **ENTITY**, além de **signals** para fazer as conexões internas.



```
architecture circuito of majority is
    signal D,E,F: std_logic;
begin
    Y <= D or E or F;
    D <= A and B;
    E <= A and C;
    F <= B and C;
end circuito;
```

Introdução à VHDL

Uso de Quartus e Modelsim

Emulação na placa DE2

Uso de Quartus e Modelsim

Dicas:

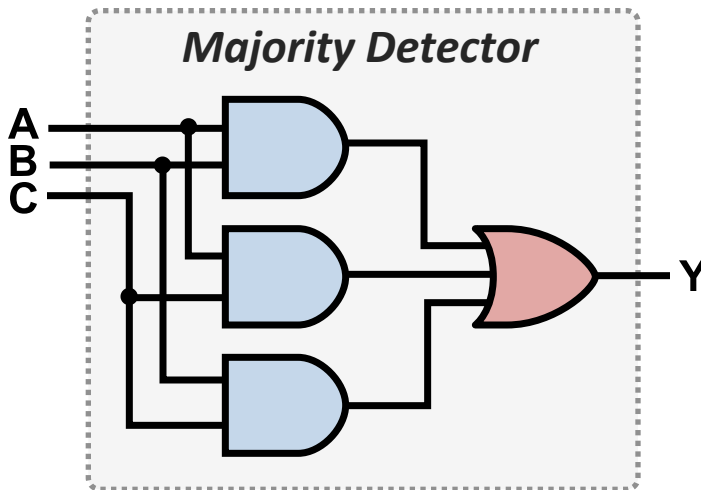
1. Se instalou o Quartus corretamente no computador, para inicia-lo simplesmente clique duas vezes o executável (sugestão de instalação Quartus versão 15.0)
2. Se quiser usar o Quartus do laboratório de forma remota o aluno deve seguir o passos do tutorial apresentado no Moodle da disciplina. O aluno deve ter instalado antes o software x2goclient e VPN da UFSC.
3. A explicação da tarefa vai ser feita usando a placa DE2-SoC.

Uso de Quartus e Modelsim

4. Uma vez aberto o Quartus. Note que o menu **File** tem opções diferentes para o Quartus project (**New Project**, **Open Project**, **Close Project**, **Save Project**) e para os arquivos (**New**, **Open**, **Close**, **Save**)
5. Quando definir o nomes de pastas, projeto ou arquivos, nunca use caracteres especiais, nem **espaços**, **ç** ou **acentos**.
6. Seja organizado, salve seu “Quartus project” em uma pasta dentro do Desktop (já seja no seu computador ou no computador do laboratório acessando remotamente).

Uso de Quartus e Modelsim

- Para fazer uso das ferramentas vamos nesse tutorial, projetar, **simular** e implementar um *Majority Detector*
 - Majority Detector*: saída em nível lógico alto sempre que a maioria dos bits de entrada estiver em nível lógico alto



$$Y = (A \text{ and } B) \text{ or } (A \text{ and } C) \text{ or } (B \text{ and } C)$$

Tabela verdade

A B C	Y
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

Uso de Quartus e Modelsim

- **Exemplo: Majority Detector**
 - **Passo 1:** Criar projeto no **Quartus II**
 - Acessar **File -> New Project Wizard** e criar um projeto como feito na aula anterior.
 - Sugestão de nome do projeto: **Lab2**
 - **Passo 2:** Criar arquivo do tipo **VHDL** dentro do projeto.
 - **File -> New -> Design Files -> VHDL File**
 - **File -> Save As** com algum nome desejado (ex.: “**Majority.vhd**”)
 - Com arquivo aberto: **Project -> Set As Top Level Entity**



Atenção: nome do arquivo deve ser igual ao da **ENTITY** que você vai criar.

Uso de Quartus e Modelsim

- **Passo 3:** Escreva o código **VHDL** do **Majority** no arquivo criado.

- *Libraries:*

```
library IEEE;  
use IEEE.Std_Logic_1164.all;
```

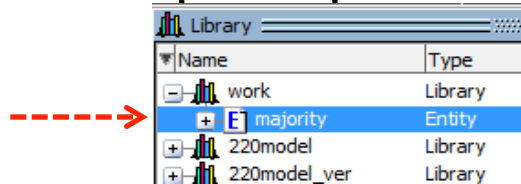
- *Entity:*

```
entity majority is  
port (A: in std_logic;  
      B: in std_logic;  
      C: in std_logic;  
      Y: out std_logic  
      );  
end majority;
```

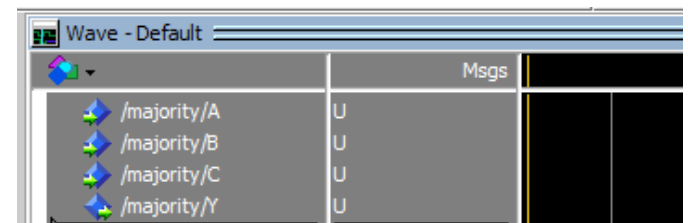
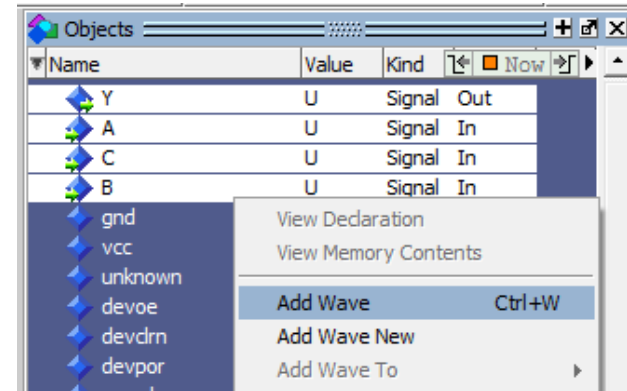
- Escrever a *Architecture* (ver slide 10)
- **Passo 4:** Compilar o projeto. Clique então em **Processing -> Start Compilation** para fazer a síntese do seu projeto

Uso de Quartus e Modelsim

- **Passo 5:** Simulação no *ModelSim*
 - Para abrir o simulador **Modelsim** use **Tools -> Run Simulation Tool -> RTL Simulation** no **Quartus II**
 - Dê um duplo clique no módulo a ser simulado no grupo **work**:

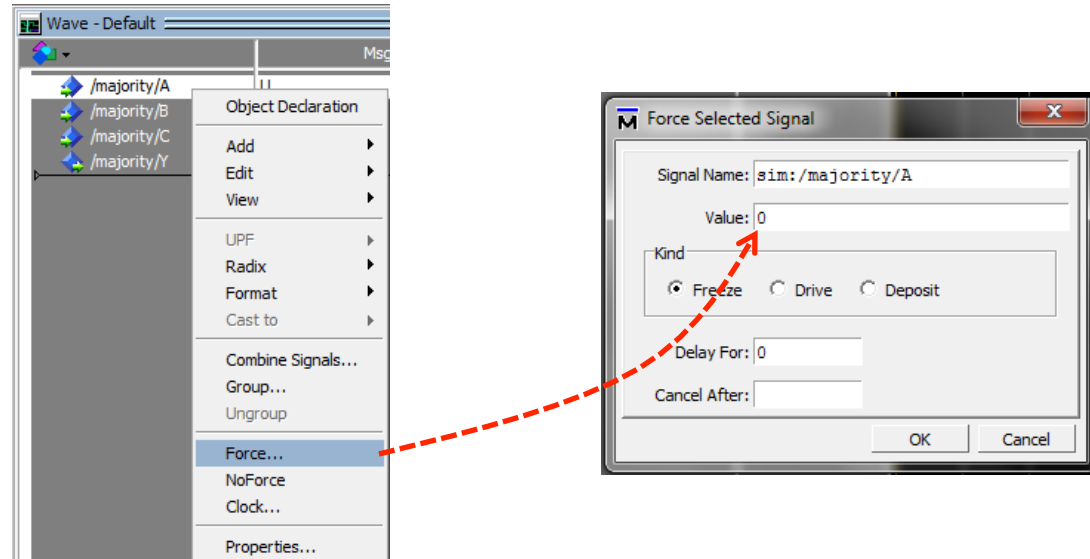


- Selecione os sinais de interesse (**A**, **B**, **C** e **Y**) na janela **Objects**, e os adicione à simulação usando a opção **Add Wave**:
- Com isso, os sinais de interesse deverão aparecer na janela **Wave**:



Uso de Quartus e Modelsim

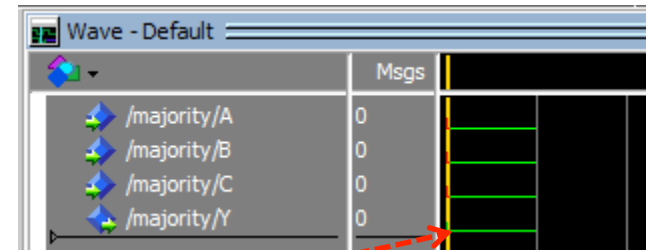
- **Passo 5:** Simulação no *ModelSim*
 - Utilize a opção **Force** para atribuir valores às entradas **A**, **B** e **C** do circuito:



- Atribua inicialmente os valores **A = B = C = 0** e, em seguida, clique em **Simulate -> Run** (ou simplesmente pressione **F9**) para simular o circuito com tais valores nas entradas.

Uso de Quartus e Modelsim

- **Passo 5:** Simulação no *ModelSim*
 - Como resultado, você deverá observar uma linha verde em nível lógico baixo para a saída **Y**, uma vez que, para o circuito considerado, **Y = 0** para **A = B = C = 0**:



- Simule então agora o circuito para todas as combinações possíveis de valores das entradas e complete a **tabela verdade**:

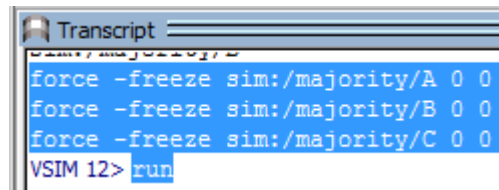
A B C			Y
0	0	0	0
0	0	1	
0	1	0	
0	1	1	
1	0	0	

Uso de Quartus e Modelsim

- **Passo 5:** Simulação no *ModelSim*
 - Finalmente, compare a **tabela verdade obtida** com a **tabela verdade esperada para o seu circuito** e verifique se ele está funcionando corretamente.

Uso de Quartus e Modelsim

- **Passo 6: Automatizando a Simulação**
 - O procedimento de simulação pode ser automatizado com a criação de um **script de simulação (testbench)**.
 - Para tal, force **A = B = C = 0**, rode sua simulação (pressionando **F9**) e, na janela **Transcript**, você poderá observar a sequencia de quatro comandos utilizada para forçar valores e rodar a simulação:



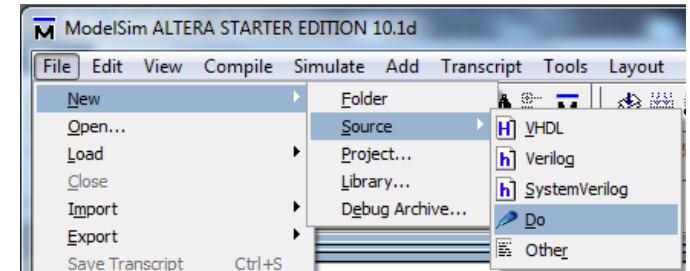
```
Transcript
sim:/majority/B
force -freeze sim:/majority/A 0 0
force -freeze sim:/majority/B 0 0
force -freeze sim:/majority/C 0 0
VSIM 12> run
```

- Para criar um **script** a partir de tais comandos, inicialmente você deve selecionar tais comandos e copiá-los (**Ctrl+C**).

Uso de Quartus e Modelsim

- **Passo 6: Automatizando a Simulação**

- Cria agora um arquivo de **script** usando **File -> New -> Source -> Do**

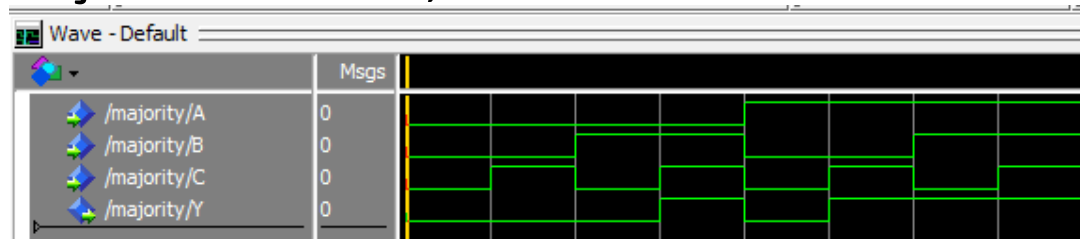


- Cole os comandos copiados e modifique-os para que a simulação seja executada com diferentes combinações de valores para **A**, **B** e **C**:

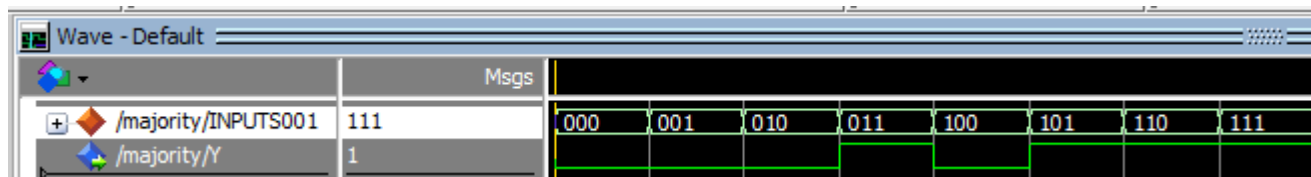
```
C:/altera/15.0/Untitled-1.do - Default *
Ln#
1    force -freeze sim:/majority/A 0 0
2    force -freeze sim:/majority/B 0 0
3    force -freeze sim:/majority/C 0 0
4    run
5    force -freeze sim:/majority/A 0 0
6    force -freeze sim:/majority/B 0 0
7    force -freeze sim:/majority/C 1 0
8    run
9    force -freeze sim:/majority/A 0 0
10   force -freeze sim:/majority/B 1 0
11   force -freeze sim:/majority/C 0 0
12   run
13   force -freeze sim:/majority/A 0 0
14   force -freeze sim:/majority/B 1 0
15   force -freeze sim:/majority/C 1 0
16   run
```

Uso de Quartus e Modelsim

- **Passo 6: Automatizando a Simulação**
 - Salve seu script (por exemplo, como **sim1.do**) e, para executá-lo, digite o seguinte comando na janela **Transcript**: **do sim1.do**
 - Como resultado, sua simulação deverá rodar para todas as combinações de valores, resultando em



- Para facilitar a visualização dos resultados, você pode selecionar os três sinais de entrada (**A**, **B** e **C**) e usar a opção **Combine Signals** para criar um agrupamento de nome **INPUTS**, resultando em:



Introdução à VHDL

Uso de Quartus e Modelsim

Emulação na placa DE2

Emulação na placa DE2

Passo 7: Fazer a emulação na placa DE2 do *Majority Detector*

Com a emulação, o fluxo de **projeto**→**simulação**→**emulação** será completado. Os passos a seguir para a emulação se mostram a seguir:

- **7.1:** Renomear o arquivo como “usertop.vhd”.
- **7.2:** A *entity* deve ser a seguinte:

```
entity usertop is
port(
  CLOCK_50:in std_logic;
  CLK_500Hz:in std_logic;
  RKEY:in std_logic_vector(3 downto 0);
  KEY:in std_logic_vector(3 downto 0);
  RSW:in std_logic_vector(17 downto 0);
  SW:in std_logic_vector(17 downto 0);
  LEDR:out std_logic_vector(17 downto 0);
  HEX0,HEX1,HEX2,HEX3,HEX4,HEX5,HEX6,HEX7: out std_logic_vector(6 downto 0));
end usertop;
```

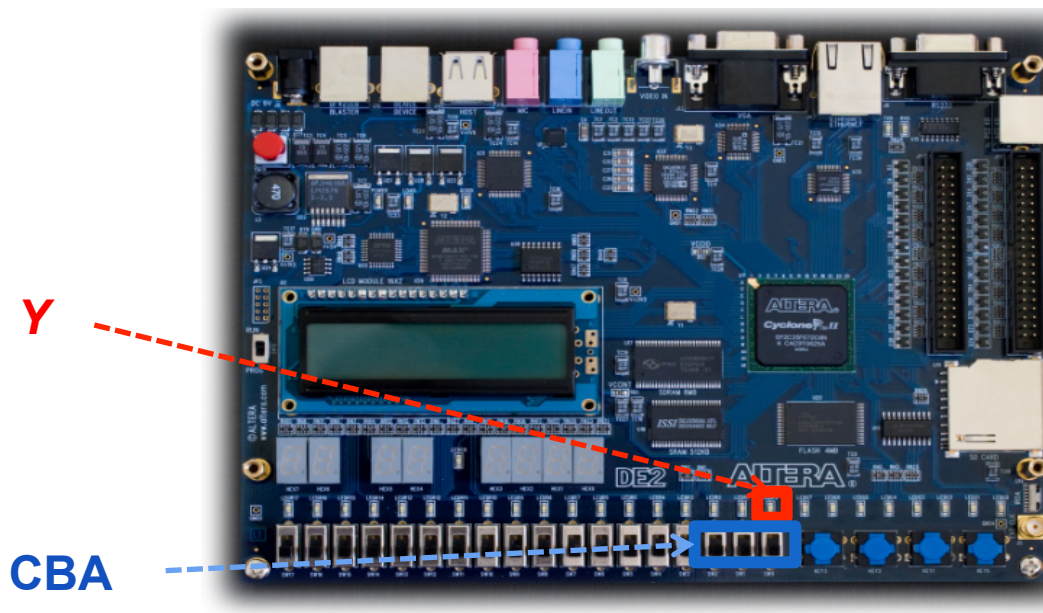
```
architecture circuito of usertop is
begin
...
```

← Não se esqueçam de trocar!

Emulação na placa DE2

Passo 7: Fazer a emulação na placa DE2 do *Majority Detector*

- **7.3:** Temos de associar **A**, **B**, **C**, e **Y** a **SW(0)**, **SW(1)**, **SW(2)** e **LEDR(0)** na “architecture” do VHDL. Troque **A** por **SW(0)**, **B** por **SW(1)**, **C** por **SW(2)**, **B** por **LEDR(0)**.



Emulação na placa DE2

Passo 7: Fazer a emulação na placa DE2 do *Majority Detector*. Existem **duas opções** que serão apresentadas nos apartados 7.4.a e 7.4.b.

- **7.4.a (Opção emulador usando x2go e VPN):** Para o uso desse emulador foi feito um tutorial onde é mostrado passo a passo como emular na placa DE2 de um exemplo “circuito 1.vhd”. Uma vez testado “cirtcuito1.vhd” faça a emulação do circuito majority.vhd.
- Com as devidas adaptações feitas no arquivo majority.vhd→usertop.vhd feito no passo 7.1 abra o terminal de comandos (ver tutorial de acesso remoto ao Quartus no Moodle) e digite:

```
cd /path
```

- onde “path” é o caminho até os arquivos VHDL que deseja utilizar na emulação. Em seguida, compile com o comando

```
fpgacompile usertop.vhd
```

- Para emular o circuito, na mesma pasta, digite

```
./fpgatest
```

Emulação na placa DE2

Passo 7: Fazer a emulação na placa DE2 do *Majority Detector*. Existem **duas opções** que serão apresentadas nos apartados 7.4.a e 7.4.b.

7.4.a (Opção emulador usando x2go e VPN) continuação: Assim que fizer isso, uma janela com a emulação do seu circuito abrirá. Quando desejar terminar a emulação, feche a janela e, em seguida, use Ctrl+C para liberar o terminal. Ao final da utilização, não se esqueça de efetuar o logout no X2Go Client.

7.4.b (Opção usando emulador on-line sem necessidade de ter o VPN da UFSC ligado): Pode ver uns exemplos de emulação na apresentação *Introdução ao emulador* disponível no Moodle. Com as devidas adaptações feitas nos arquivos abra o emulador <http://150.162.54.54:5000/> , e faça o cadastro em *Sign up*. Uma vez cadastrado, faça *Upload* dos arquivos VHDL que deseja utilizar na emulação. Uma vez os arquivos carregados pressione *Compile* e apos verificar que não existem erros de emulação pressione *Go to emulation* e no emulador, inicie a emulação e verifique o funcionamento.

Emulação na placa DE2

Passo 7: Fazer a emulação na placa DE2 do *Majority Detector*. Existem **duas opções** que serão apresentadas nos apartados 7.4.a e 7.4.b.

- **7.4.a (Opção emulador usando x2go e VPN):** Com as devidas adaptações feitas nos arquivos abra o terminal de comandos (ver tutorial de acesso remoto ao Quartus no Moodle) e digite:
`cd /path`
- onde “path” é o caminho até os arquivos VHDL que deseja utilizar na emulação. Em seguida, compile com o comando
`fpgacompile usertop.vhd`
- Para emular o circuito, na mesma pasta, digite
`./fpgatest`
- Assim que fizer isso, uma janela com a emulação do seu circuito abrirá. Quando desejar terminar a emulação, feche a janela e, em seguida, use Ctrl+C para liberar o terminal.
- Ao final da utilização, não se esqueça de efetuar o logout no X2Go Client.

Emulação na placa DE2

Passo 7: Fazer a emulação na placa DE2 do *Majority Detector*. Existem **duas opções** que serão apresentadas nos apartados 7.4.a e 7.4.b.

7.4.b (Opção usando emulador on-line sem necessidade de ter o VPN da UFSC ligado): Está disponível no Moodle **exemplos de uso do emulador online na apresentação *Introdução ao emulador***. Com as devidas adaptações feitas nos arquivos abra o emulador <http://150.162.54.54:5000/> e faça **UPLOAD** dos arquivos VHDL que deseja utilizar na emulação. Em seguida, vai para o emulador, inicie a emulação e verifique o funcionamento.