

MC 602

IC/Unicamp
2012s2

Prof Mario Côrtes

Introdução à programação baseada em VHDL

Conteúdo

- Programação da DE1 usando VHDL no Quartus
- Resumo do tutorial:
 - `tut_quartus_intro_vhdl.pdf`
- Grande parte do procedimento é igual ao usado no diagrama esquemático

Criação de novo projeto (ver diff)

- Abrir o Quartus
- File > New Project Wizard
 - Definir diretório onde o projeto será armazenado:
`tut_vhdl`
 - Escolher nome do projeto (2 próximos campos):
`light_vhdl`
 - Next
 - Next (mecanismo para adicionar arquivos)
 - Family device settings:
 - escolher Cyclone II EP2C20F484C7
 - Next (other EDA tools)
 - Finish

Circuito a ser criado

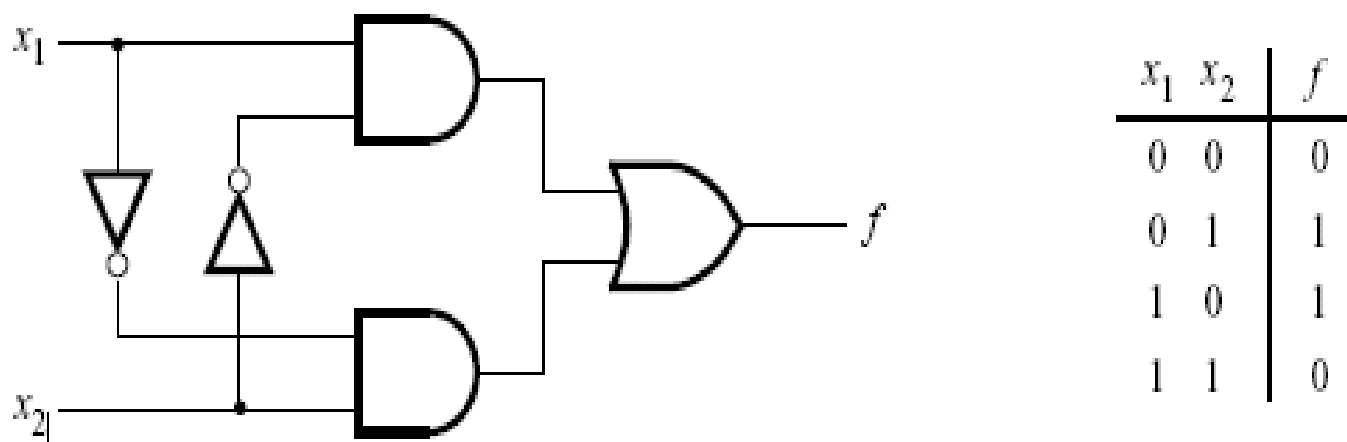


Figure 11. The light controller circuit.

Criação do código VHDL

- File > New > **VHDL File**
- File > Save as > **light_vhdl** > OK
 - Atenção: marcar checkbox “Add file to current project”
- Copiar e colar texto abaixo e salvar

```
LIBRARY ieee ;  
USE ieee.std_logic_1164.all ;
```

```
ENTITY light_vhdl IS  
    PORT ( x1, x2 : IN STD_LOGIC ;  
           f : OUT STD_LOGIC ) ;  
END light ;
```

```
ARCHITECTURE LogicFunction OF light_vhdl IS  
BEGIN  
    f <= (x1 AND NOT x2) OR (NOT x1 AND x2);  
END LogicFunction ;
```

Próximos passos

- são idênticos aos usados para diagr. esquemático
 - compilação
 - atribuição de pinos
 - criação de waveform
 - simulação
 - programação

Compilação e atribuição de pinos

- Compilação
 - Processing > Start Compilation, ou clicar no ícone ►
 - Verificar o relatório de compilação: Processing > Compilation Report, ou clicar no ícone apropriado
- Atribuição de pinos
 - (associar um pino da FPGA a um dispositivo de entrada/saída da placa)
 - (ver tabela de atribuição de pinos no manual de usuário ou no arquivo DE1pin assignments.odt)
 - Entradas: SW0 e SW1, associadas aos pinos PIN_L22 e PIN_L21
 - Saídas: LED verde DG0, associado ao pino PIN_U22
 - Assignments > Pins: selecionar os pinos acima e OK

Criação de waveform

- Recompilar
- Criação de waveform para simulação (ver tutorial):
 - File > New > Vector Waveform File
 - Clicar botão direito na coluna de nomes de sinais > Insert > Insert Node > Node Finder
 - List > selecionar os sinais e movê-los para o painel direito
 - Edit > End Time > 200 ns
 - Inserir forma periódica para as entradas
 - Selecionar linha x1; pressionar botão “Count Value” > iniciar 1, a cada 100ns
 - Repetir para x2, com iniciar 0, a cada 40ns
 - Salvar

Simulação

- Simulação funcional: atrasos não são levados em consideração
 - Assignment > Settings > Simulator Settings > Simulation mode = Functional
 - Geração de netlist: Processing > Generate Functional Simulation Netlist
 - Processing > Start Simulation (ou pressionar ícone apropriado)
 - Observar que não há atrasos para a saída
- Simulação com timing: há atrasos
 - Assignment > Settings > Simulator Settings > Simulation mode = Timing
 - Processing > Start Simulation (ou pressionar ícone apropriado)
 - Observar atrasos e glitches (hazards)

Programação

- Verificar se a char RUN / PROG está em RUN
- Tools > Programmer
- Verificar: HW Setup = USB Blaster, Mode = JTAG, CheckBox Program Configure = ON
- Selecionar arquivo (linha) e Start
- Programação concluída
- Testar funcionamento na placa