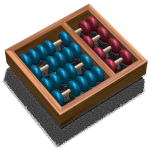

UNICAMP - Instituto de Computação

Disciplinas e
MC602 - Circuitos Lógicos e Organização de Computadores
MC613 - Laboratório de Circuitos Lógicos

Tutorial de Projeto Hierárquico

PED: Caio Hoffman
Professor: Mario Lúcio Côrtes

Setembro de 2011

Sumário

Lista de Figuras

Lista de Tabelas

Conteúdo do Tutorial

1. Projeto hierárquico (utilizando símbolos).
2. Organização dos sinais de uma simulação.
   (a) Agrupamento e desagrupamento de sinais.
   (b) Configuração dos valores dos sinal de entrada.

Materiais

- Quartus 9.1 sp2 da Altera.

1 Projeto Hierárquico

1.1 Gerando um símbolo

É comum no projeto de circuitos lógicos que uma parte do circuito esteja replicada várias vezes em todo o projeto. Esta parte, que é um circuito com uma função específica dentro do projeto, pode ser visto como uma caixa preta. Para o projetista isso é uma facilidade, pois acaba só lhe interessando quais são as entradas e saídas e a função deste circuito, permitindo modularizar o projeto.

Esse processo de modularização, composto por funções pequenas englobadas por funções maiores é chamado hierarquização. Para ilustrar um projeto hierárquico no *Quartus*, vamos supor que não exista a porta lógica XOR pré-definida no Quartus. Na Figura 1 temos a implementação de uma XOR usando portas AND, OR e NOT. O projeto criado chama-se ***tutorial*** e o diagrama de blocos ***myxor.bdf***.

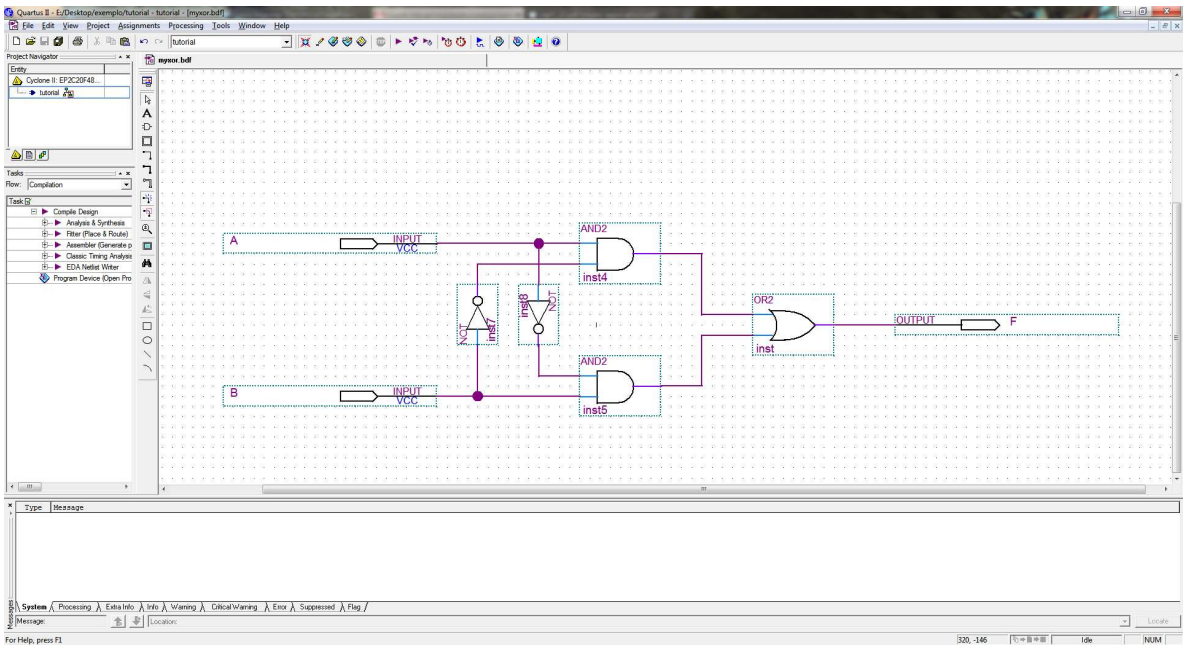

Figura 1: Porta XOR criada com portas AND, OR e NOT.

O próximo passo é criar o simbolo, veja na Figura 2 como isto é feito. O nome do arquivo ficou como ***myxor.bsf***.

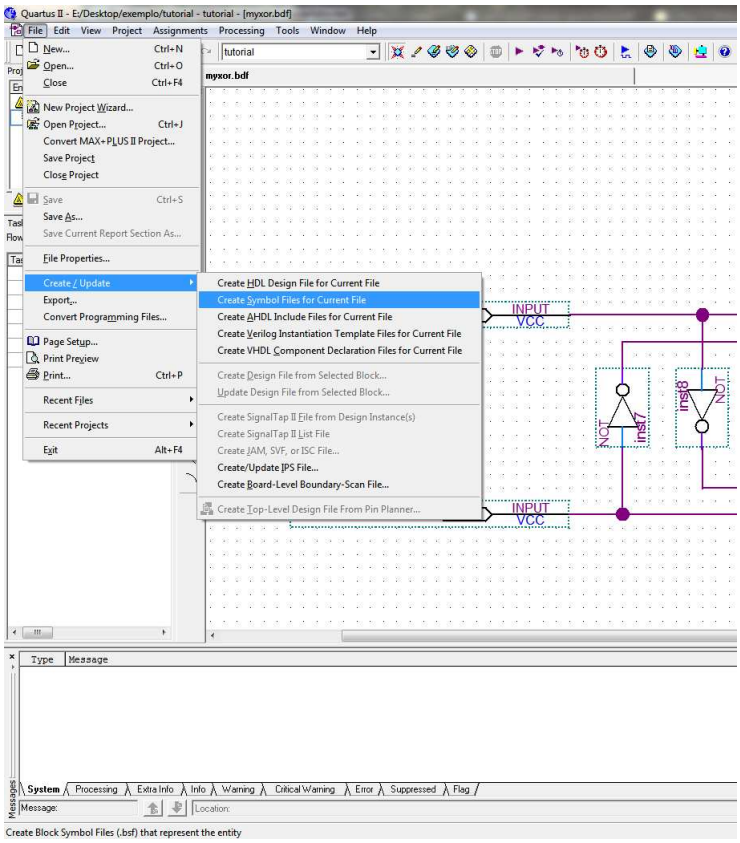

Figura 2: Gerando o símbolo do circuito XOR da 1.

1.2 Utilizando um símbolo

Prosseguindo com o projeto, suponha que sua especificação seja de um verificador de paridade ímpar de 4 bits. Para cumprir a especificação vamos usar a porta XOR já desenhada. O primeiro passo é criar um novo diagrama de blocos. Então, chamamos a janela de inserção de símbolos. Após a tela ter sido invocada seleciona-se a pasta ***Project***, que pode ser vista na Figura 3, clica-se no mais, seleciona-se ***myxor*** e, daí, é dado OK.

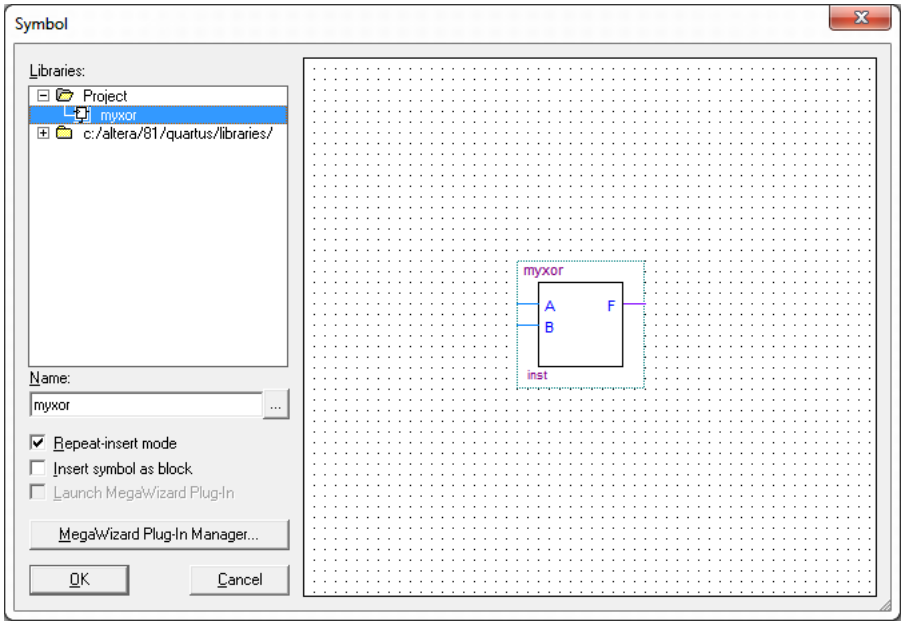

Figura 3: Inserindo um símbolo do projeto.

Como a porta XOR usada só possui duas entradas, será necessário que três portas sejam usadas para o desenho do circuito. O novo diagrama de blocos foi saldo no arquivo ***tutorial.bsf***. Veja como a implementação do verificador de paridade de 4 bits ficou na Figura 4.

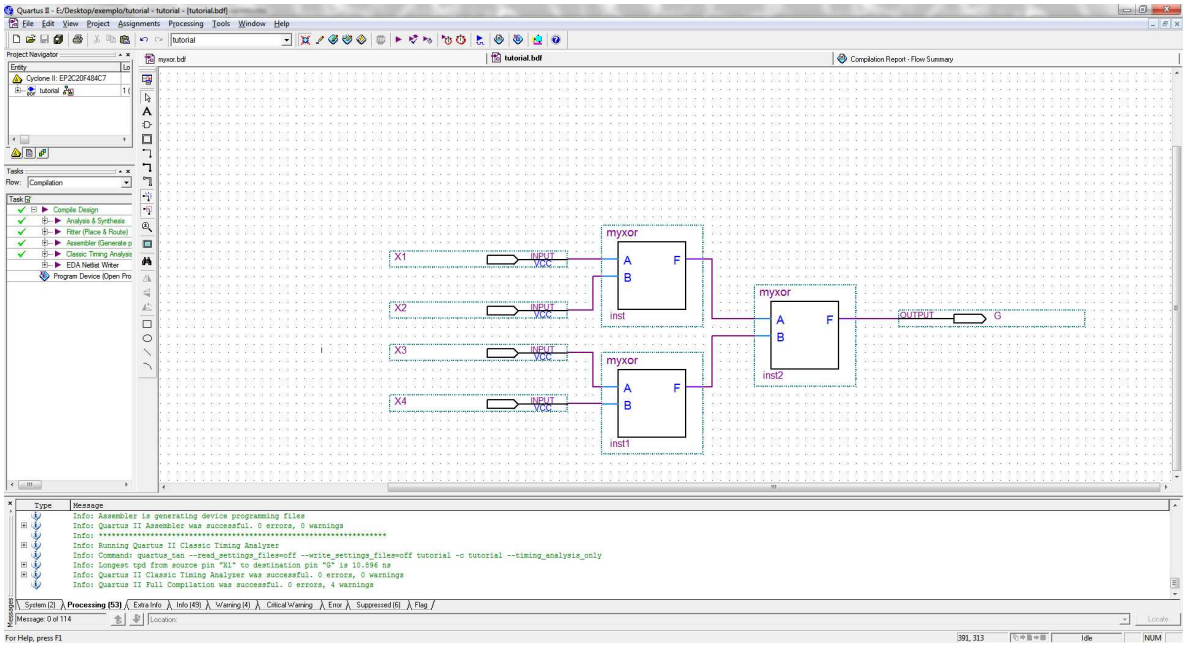

Figura 4: Esquemático do verificador de paridade de 4 bits.

2 Organização dos Sinais na Simulação

2.1 Agrupamento e Desagrupamento de Sinais

Para verificar a corretude do verificador de paridade ímpar, vamos simular as possíveis entradas do projeto e verificar se elas condizem com o valores da Tabela 1.

Tabela 1: Tabela verdade para a função de verificação da paridade ímpar.

$(x_1x_2x_3x_4)_{10}$	f
0	0
1	1
2	1
3	0
4	1
5	0
6	0
7	1
8	1
9	0
10	0
11	1
12	0
13	1
14	1
15	0

Depois do projeto ter sido compilado, adicionamos um *Vector Waveform file* ao projeto, salvo como ***tutorial.vwf***. Então adicionamos os sinais x_1, x_2, x_3,x_4 e f. Para facilitar a comparação do resultado da simulação com a Tabela 1, agruparemos os sinais de entrada para serem exibidos em decimal.

Com mouse selecionamos os sinais que queremos agrupar, nesse caso os sinais de entrada x_1, x_2, x_3 e x_4. Clica-se com o botão direito do mouse em qualquer item selecionado, seleciona-se a opção ***Grouping→Group***, como na Figura 5.

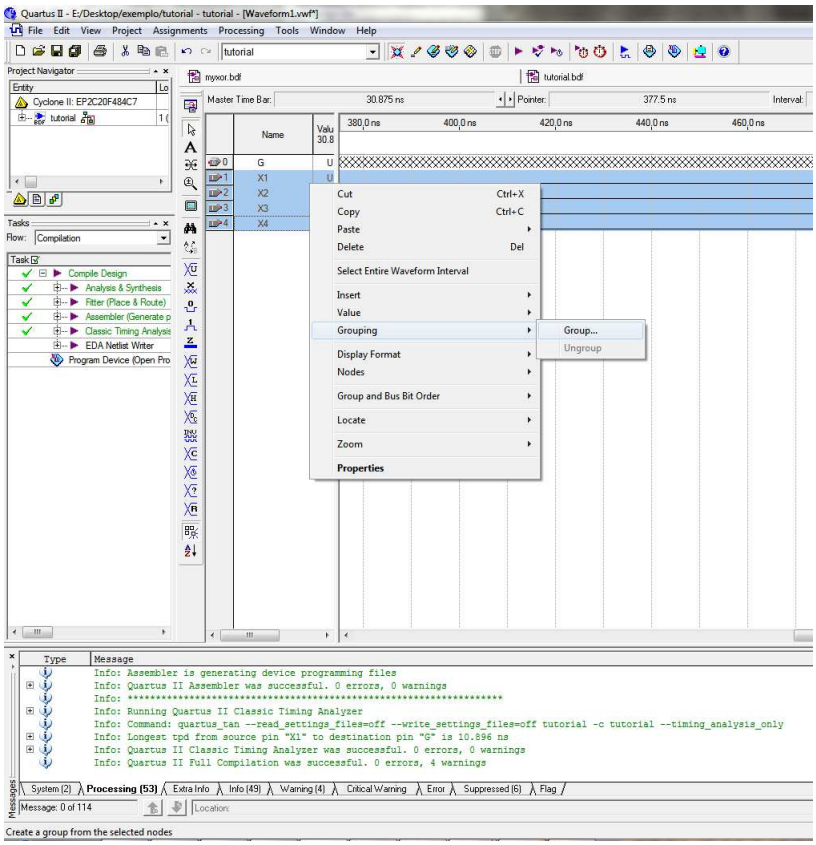

Figura 5: Acesso a opção de agrupamento dos sinais.

A tela por conseguinte é vista na Figura 6. Após dar o nome de **Entrada**, seleciona-se ***Unsigned Decimal*** no menu de opções ***Radix*** (Figura 7). OK finaliza essa etapa.

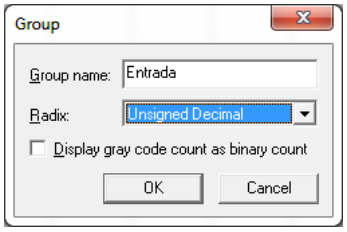

Figura 6: Configurando o agrupamento de sinais.

Note que na opção *Radix* podemos selecionar diversas maneiras de exibir um conjunto de sinais, como números binários, números octais ou até mesmo em hexadecimal.

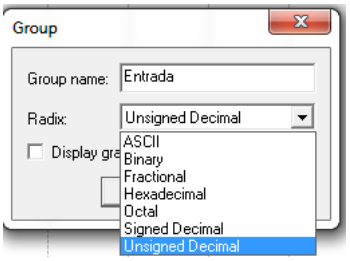

Figura 7: Selecionando o formato de exibição dos sinais.

Para modificar o modo de exibição, uma das alternativas é clicar com o botão direito do mouse sobre o sinal **Entrada** (veja na Figura 8) e ao invés de selecionarmos ***Group*** na opção ***Grouping***, no menu visto na Figura 5, selecionamos ***Ungroup***. Então, repetimos o processo descrito nas Figuras 5, 6 e 7.

2.2 Configuração dos Sinais de Entrada

Antes de simular, selecionamos o sinal **Entrada** e, após o clique com o botão direito do mouse, escolhemos a opção ***Value*→*Count Value*** (veja na Figura 8). A janela que será apresentada está na Figura 9.

Note que existem várias opções de configurar os sinais de entrada (tanto em conjunto quanto individualmente), como visto no menu apresentado na Figura 8. Contudo, queremos simular todas as possibilidades de um grupo de sinais e configurá-los como um contador é o caminho mais simples.

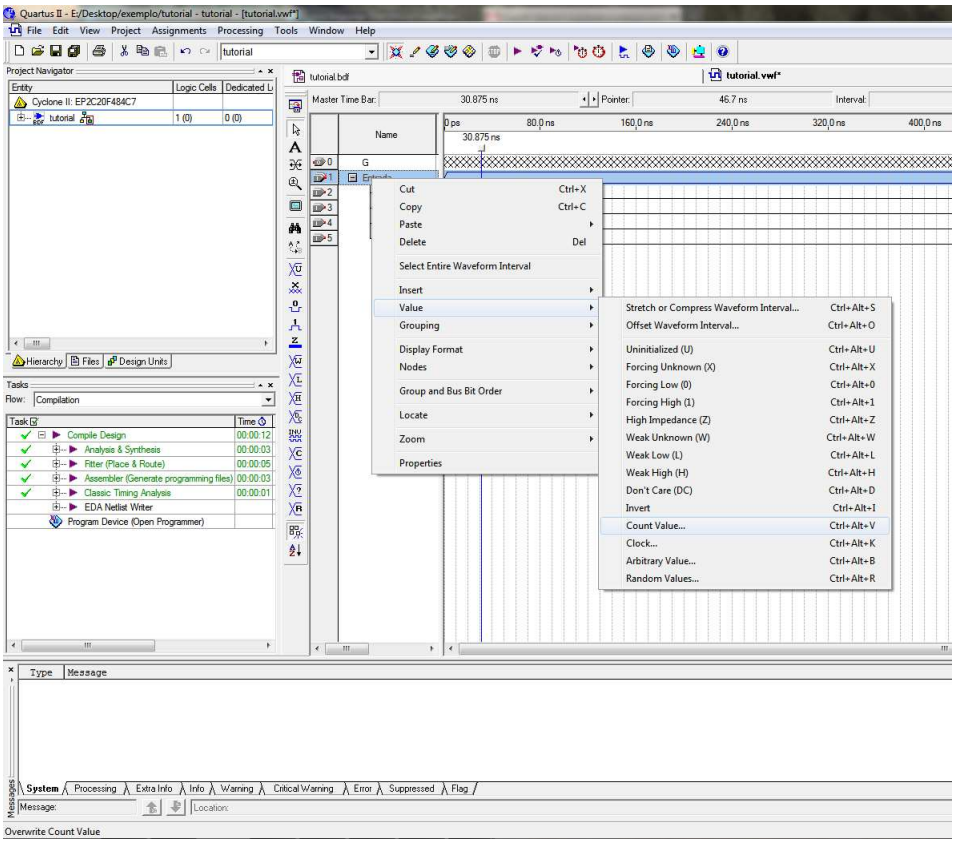

Figura 8: Menu de configuração de valores para os sinais.

A opção ***Radix*** no nosso caso, não deve ser alterada, pois queremos a exibição dos números como decimais. O ***Start value*** deve ser 0, ***Increment by*** igual 1 e ***Count type*** como ***Binary***. O valor *End value* será automaticamente atualizado depois que os parâmetros da aba ***Timing***, Figura 10, forem corretamente modificados.

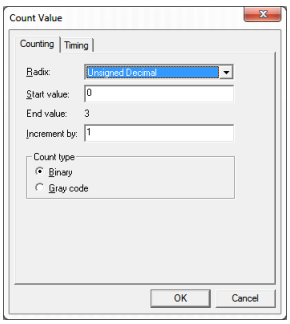

Figura 9: Janela de configuração para um sinal no formato de contador.

Na aba ***Timing*** vamos manter os valores de ***Start time*** e ***End time***, porque só estamos interessados na simulação de contagem de valores. O ajuste desses tempos

só seria necessário se o sinal **Entrada** tivesse que apresentar comportamentos distintos durante a simulação. No campo ***Count every*** é inserido o tempo de 50 ns, isto é, o contador será incrementado a cada 50 ns. Dessa forma, é possível apresentar todos os dezesseis valores (para cobrir todas as possíveis combinações dos quatro sinais de entrada), visto que $16 \cdot 50$ ns $= 800$ ns < 1 μs (***End Time***).

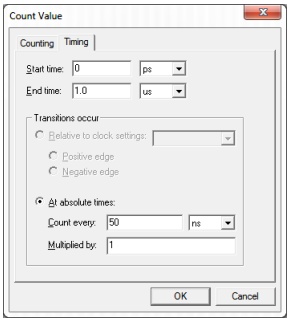

Figura 10: Configurando a contagem do sinal.

Finalmente, na Figura 8 vemos o resultado da simulação (funcional) que ocorreu como previsto.

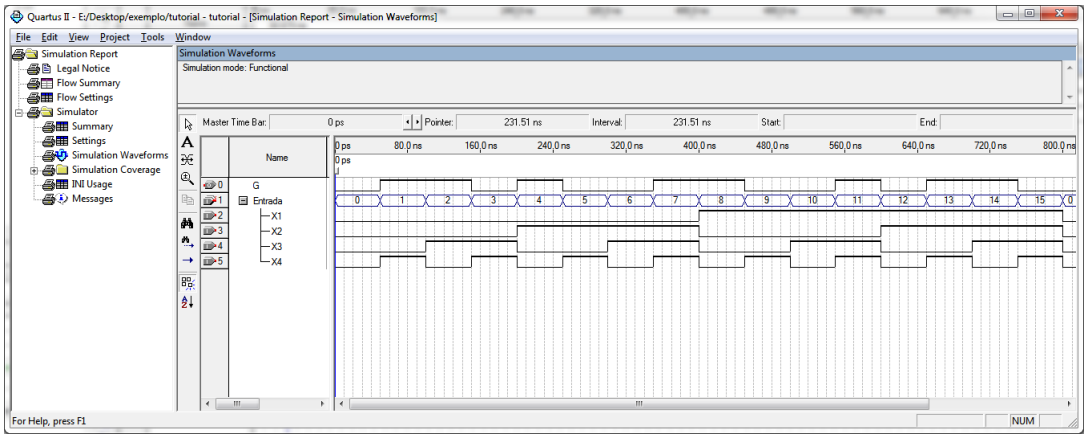

Figura 11: Simulação do Projeto.

Observações

- Esse tutorial é simplificado e não contém todas as formas de executar os passo aqui descritos.
- Pressupõe-se que o usuário já tenha tido um contato inicial com Quartus.