Grupo de Electrónica e Instrumentação

Electrónica Analógica e Digital

Trabalho de Laboratório 2 (v2-Março de 2009)

Objectivo e introdução

O objectivo deste trabalho é promover a familiarização dos alunos com ferramentas de desenvolvimento de hardware digital implementado em FPGAs. Recorre-se à utilização do ISE Webpack da Xilinx (ISE) para sintetizar, a partir de uma descrição em Verilog, alguns projectos de pequena dimensão.

Pode ser utilizada qualquer versão do ISE a partir da 6.3 (aquela que está instalada nos PCs do laboratório de Electrónica.) Actualmente o ISE vai na versão 10.1, e para descarregar os correspondentes 2,45 Gbytes (que a Xilinx disponibiliza a custo zero) é necessário fazer uma inscrição na página Web apropriada. No entanto, na página de EAD são disponibilizados os ficheiros necessários à instalação da versão 6.3, bem como alguns tutoriais sobre a sua utilização. Esta versão 6.3, embora mais antiga, apenas tem cerca de 500 Mbytes de binário de instalação e é perfeitamente adequada aos nossos (modestos) propósitos.

O ISE Webpack

O ISE está instalado nos PCs do laboratório. Clique no ícone (mostrado acima) do *"Project Navigator"* da Xilinx, que inicia o ISE, crie um novo projecto seguindo as instruções do "Project Wizard" que entretanto é lançado e escreva o Verilog correspondente ao módulo que pretende implementar. Guarde o ficheiro e no painel do ISE que contém o fluxo de projecto carregue, mesmo no final, em *"Generate Programming File"*. Se tudo correr bem, poderá examinar em ficheiros criados pela ferramenta o resultado das várias fases da síntese do projecto (o diagrama RTL do módulo, o *layout* na FPGA, a velocidade de funcionamento, etc...). Na fig. 1 vê-se a janela do ISE com a interface de um módulo já definida, estando o sistema pronto para que seja especificada a sua funcionalidade em Verilog.

Plano de trabalho

Implemente cada um dos blocos descritos de seguida no ISE e observe, em cada caso, o circuito que o sintetizador cria a partir deles (clique na linha *"RTL Schematic"* para ver os circuitos). Pode também examinar a sua implantação na FPGA (use o *floorplanner*). Embora possa implementar todos os módulos num mesmo projecto, sugere-se que crie um projecto para cada um deles.

Somador completo

O somador completo é uma das células básicas da aritmética digital. Soma (em binário) as entradas X e Y e coloca o resultado em S e a carga (*"carry"*) em C. O bloco de Verilog que se segue implementa-o de uma forma quase integralmente estrutural (a excepção é a linha começada por `assign`).

```
module FULLADDER(X, Y, C, S);
  input X, Y;
  output C, S;

  nand NANDA(S3 ,X, Y);
  nand NANDB(S1 ,X, S3);
  nand NANDC(S2 , S3 ,Y);
  nand NANDD(S , S1 , S2);
  assign C = ~S3;
endmodule
```

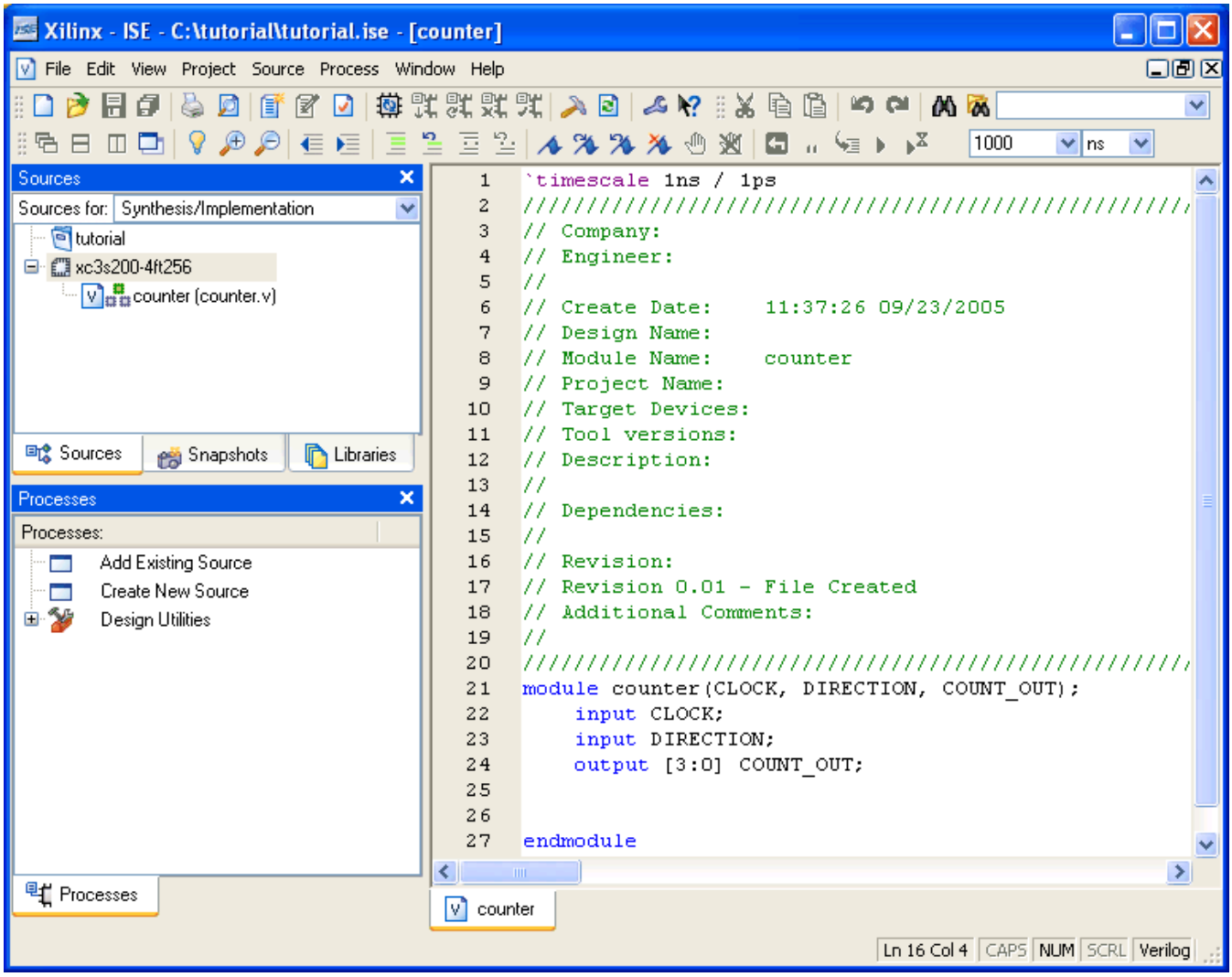

Fig. 1: Panorâmica da janela do ISE da Xilinx.

Divisor de frequência de relógio

O seguinte código implementa um contador de 28 bits. A função deste componente nas nossas realizações será a geração de um sinal de relógio de frequência submúltipla da frequência do relógio de entrada, f_{clk}. Com efeito, a frequência de `bigcount[0]` é $f_{clk}/2$; a frequência de `bigcount[1]` é $f_{clk}/4$; generalizando, a frequência de `bigcount[k]` é $f_{clk}/2^{k+1}$. Neste módulo, em particular, os 4 bits da saída `count` iriam ligar aos 4 LEDs da placa da Cesys que existe no laboratório. Como correspondem aos 4 bits mais significativos do contador, a sua frequência já é suficientemente baixa para que se possa ver os LEDs a piscar (se a frequência fosse muito elevada, eles pareceriam estar sempre acesos "a meia luz".)

```
module CONTADOR(clk ,count);
  input clk;
  output [3:0] count;
  reg [27:0] bigcount;

  always @(posedge clk)
    bigcount <= bigcount+1 ;
  assign count = bigcount[27:24];
endmodule
```

Máquina de estados com código (quase integralmente) estrutural

Neste último exemplo vai ser realizada uma máquina de estados, denominada `LAB2`[1] sem, porém, utilizar uma descrição comportamental de ME em Verilog. Usa-se um FF do tipo D, segundo a descrição dada na *"template"* da Xilinx, e alguns `assign` para realizar a lógica combinatória correspondente às equações de excitação dos FFs.

O código correspondente ao FF é:

```
module FFD(C, D, CLR, Q);
  input C, D, CLR;
  output Q;
  reg Q;

  always @(posedge C or posedge CLR) begin
    if (CLR)
      Q <= 1'b0;
    else
      Q <= D;
    end
endmodule
```

e a máquina de estados global será descrita pelo seguinte módulo:

```
module LAB2(C, CLR, IN, OUT);
  input C, IN, CLR;
  output OUT;

  FFD flip0 (C, D0 , CLR, Q0);
  FFD flip1 (C, D1 , CLR, Q1);
  assign D1 = IN & (Q1 | Q0);
  assign D0 = IN & (Q1 | ~Q0);
  assign OUT = Q1 & ~Q0;
endmodule
```

Note que esta máquina instancia por duas vezes o flip-flop D anteriormente especificado, o que significa que usa dois flip-flops. Estes módulos podem ser ambos escritos num mesmo ficheiro, ou escritos em ficheiros diferentes: o "gestor de projectos" do ISE lida bem com qualquer dos casos.

Finalmente, deverá tomar nota de que o ISE., e outras ferramentas equiparadas, aceitam descrições comportamentais de máquinas de estados. Se forem respeitados os respectivos *"templates"* sugeridos pela ferramenta garante-se que o sintetizador reconhecerá correctamente as máquinas. Esta capacidade facilita sobremaneira a descrição de MEs complexas e/ou de grandes dimensões.

Exercícios (facultativos, para treinar depois da aula)

1. Fazendo uma pequena modificação no módulo FULLADDER, implemente-o segundo uma descrição integralmente estrutural.
2. Faça as tabelas de verdade das saídas do módulo FULLADDER.
3. A partir do módulo FULLADDER, implemente um módulo somador que tem mais uma entrada, CIN, correspondente à carga proveniente de um módulo somador anterior, e mantém as saídas (C e S).
4. Implemente um *reset* assíncrono, activo a '1', no módulo contador.
5. Realize a máquina de estados usando Verilog comportamental de acordo com um dos formatos (*"templates"*) dados na documentação da Xilinx (e também acessíveis nos menus do ISE).

[1] É a máquina já anteriormente especificada no primeiro trabalho de laboratório.