SystemVerilog para Projetos

Curso Presencial IFPB/UFAL

Marcos Morais – Professor DEE
Projetos de Excelência em Microeletrônica – PEM
Centro de Engenharia Elétrica e Informática
Universidade Federal de Campina Grande

Sumário

- Níveis de abstração;
- Projetos bottom-up e top-down;
- Diagrama de blocos;
- Separação entre caminho de dados (datapath) e controle;
- Linguagens de descrição de hardware;
- Fluxos de projeto em microeletrônica e FPGA;
- Subconjunto sintetizável;
- Descrição comportamental e nível RTL;
- Módulos;
- Tipos de dados e valores literais;
- Operadores e expressões;
- Sentenças procedurais e de controle;
- Atribuição de variáveis;
- Procedimentos e processos;
- Tarefas e funções;
- Interfaces;
- Simulação e síntese lógica;
- Aplicações e exemplos;
- Laboratório - Projeto do flux-capacitor (vai e vem de LEDs);
- Laboratório - Projeto de máximo divisor comum

Sistemas Digitais Complexos com SystemVerilog HDL

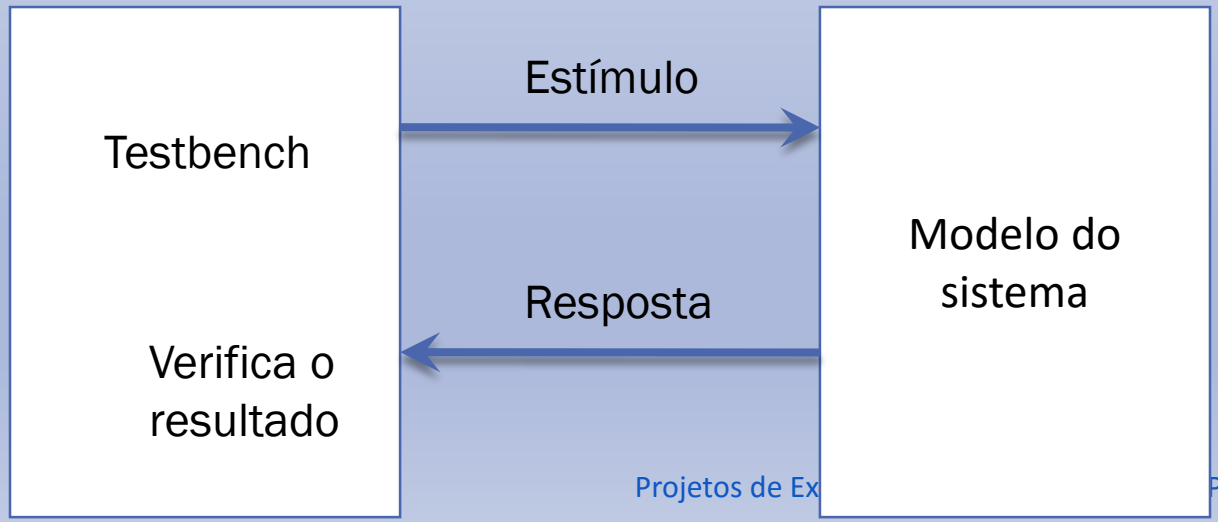
Nível COMPORTAMENTAL e RTL

Projetos de Excelência em Microeletrônica – PEM
Centro de Engenharia Elétrica e Informática
Universidade Federal de Campina Grande

Modelagem COMPORTAMENTAL

Usando Verilog para Simulação

- O testbench gera estímulos e verifica a resposta
- Acoplado ao modelo do sistema
- O par executa simultaneamente



Simulação de Eventos Discretos

- Idéia básica: somente trabalhe quando algo muda
- Centrado sobre uma fila de eventos que contêm eventos marcados com o tempo de simulação em que devem ser executados
- Paradigma básico da simulação
  - Executa todo evento para o tempo simulado atual
  - Fazendo isto muda o estado do sistema e pode escalonar eventos no futuro
  - Quando não há eventos restantes no tempo atual, avança o tempo simulado para o tempo de evento mais cedo da fila

Modelagem Comportamental

- Maneira muito mais fácil de escrever testbenches
- Também pode ser usado para fazer modelos abstratos dos circuitos
  - Mais fácil de escrever
  - Simula mais rápido
- Mais flexível
- Provê sequenciamento
- Verilog obteve sucesso em parte porque permitiu que o modelo e o testbench fossem descritos conjuntamente
- Aula posterior específica sobre simulação com verilog e verificação funcional

Dois componentes principais de Verilog: Comportamental

- Processos concorrentes, disparados por eventos (comportamental)
- Blocos initial e always
- Código imperativo que pode realizar tarefas padrão de manipulação de dados (atribuição, if-then, case)
- Processos ou estão executando (no caso eles podem produzir atrasos) ou estão esperando por um evento de disparo

Blocos Initial e Always

```
initial
   begin
      // sentenças
      // imperativas
end
```

- Roda quando a simulação começa
- Termina quando o controle atinge o end
- Bom para prover estímulos

```
always
   begin
      // sentenças
      // imperativas
end
```

- Roda quando a simulação começa
- Restarta quando o controle atinge o end
- Bom para modelagem ou para especificar hardware

Always e Initial

✖ Executam até encontrar um atraso

```
initial begin
  #10 a = 1; b = 0;
  #10 a = 0; b = 1;
end
```

✖ Ou uma espera por um evento

```
always @(posedge clk) q = d;

always begin
  wait(i);
  a = 0;
  wait(~i);
  a = 1;
end
```

Operadores de atribuição

- Atribuição bloqueante: =
- Atribuição não-bloqueante: <=

```
module assignment_test();
  reg [4:0] a,b,c,e;
  reg [7:0] d;
  initial
        begin
              a<=4'd2;
              $display ("A is %d", a);
              $display ("B is %d", b);
              $display ("D is %d", d);
              $display ("\n\n");
              b<=4'd3;
              $display ("A is %d", a);
              $display ("B is %d", b);
              $display ("D is %d", d);
              $display ("\n\n");
              d<=(a+b);
              $display ("A is %d", a);
              $display ("B is %d", b);
              $display ("D is %d", d);
              $display ("\n\n");
        end
endmodule
```

Resultado

Saída do compilador (atribuição bloqueante)	Saída do compilador (atribuição não-bloqueante)
A is 2 B is x D is x	A is x B is x D is x
A is 2 B is 3 D is x	A is x B is x D is x
A is 2 B is 3 D is 5	A is x B is x D is x

- Atribuição bloqueante: execução seqüencial
- Atribuição não-bloqueante: "simultânea" apenas no final

Como usar

- Basicamente:
  - Use não-bloqueante em lógica sequencial
  - Use bloqueante em lógica combinacional

SystemVerilog

SystemVerilog

- Em verilog *wire* é utilizado para conexões
- *input* e *output* são *wires*
- *reg* é utilizado para descrever memória
- Um *reg* pode desaparecer
- Um *reg* não é um registrador!

- Systemverilog simplifica tudo utilizando a palavra chave **logic**
- **logic** pode ser utilizado no lugar de um *wire*, no lugar de um *reg* e no lugar do par *wire*,*reg*.
- System verilog possui tipos de dados com sinal

Na prática

- Códigos fonte verilog tem a extensão .v
- Códigos em systemverilog tem a extensão .sv

O que vimos hoje

- Modelagem de circuitos lógicos no nível RTL usando descrições procedurais
- O uso do reg como indicador de variáveis
- O bloco always como indicador de hardware
- Conceitos de simulação com verilog

Circuitos sequenciais e controle

PRÓXIMA AULA

Contato

Marcos Morais
Professor DEE
morais@dee.ufcg.edu.br

Linguagem de Descrição de Hardware

Prof. Marcos Morais
DEE/UFCG

Projetos de Excelência em Microeletrônica – PEM
Centro de Engenharia Elétrica e Informática
Universidade Federal de Campina Grande

Sumário

- Níveis de abstração
- Fluxos de desenvolvimento
- Abordagens top-down e bottom-up
- Linguagens de descrição de hardware
- Verilog
- Descrição de circuitos combinacionais
- Atribuição contínua
- SystemVerilog: evolução de Verilog