RELATÓRIOS - EXERCÍCIOS

SD112 - Introdução a Verilog

DOCENTE Felipe Gustavo de Freitas **Rocha**

DISCENTE André Francisco Ribeiro **Bezerra**

DATA DE ENTREGA **07 de novembro de 2025 (prazo máximo)**

A-001	A-002	A-003
A-004	A-005	A-006
A-007	A-008	A-009
A-010	A-011	A-012
A-013	A-014	

23 de outubro

Formulário para envio das Atividades

SUMÁRIO

REFERÊNCIAS

[1] Digital Systems ; Authors, Ronald Tocci, Neal Widmer, Greg Moss ; Edition, 12 ; Publisher, Pearson Education, 2016 ; ISBN, 0134220145, 9780134220147;

[2] Frank Vahid. 2010. Digital Design with RTL Design, Verilog and VHDL (2nd. ed.). Wiley Publishing.;

☐ A-013: Primitivas

Conceitual

- [x] 1 O que são primitivas em Verilog e quais são os dois tipos principais?
- [x] 2 Qual a diferença entre primitivas predefinidas e definidas pelo usuário (UDPs)?
- [x] 3 3. Qual é a função de uma primitiva condicional como *bufif0* e *notif0*?
- [x] 4 Explique a diferença da primitiva sensível à borda e sensível ao nível.
- [x] 5 Qual a diferença entre primitivas unidirecionais e bidirecionais?
- [x] 6 O que caracteriza uma UDP combinacional e uma UDP sequencial?

Prático

- [x] 1 Pesquise e implemente em Verilog, utilizando primitivas built-in de transistores, o modelo de uma porta lógica AND.
- [x] 2 Escreva um código em Verilog que implemente um MUX 2:1, utilizando primitivas built-in.
- [x] 3 Implemente um flip-flop tipo D com reset assíncrono como uma UDP.
- [x] 4 Crie uma testbench, simule e valide a implementação do exercício anterior.

A-013: Primitivas (Conceitual)

1- Primitivas em Verilog são blocos de construção lógicos fundamentais, pré-definidos na linguagem. Servem como base para a modelagem ESTRUTURAL (gate-level modeling). Representação do hardware básico (portas lógicas e transistores), sem a necessidade de uma descrição comportamental interna.

Os principais tipos são (dois):
→ GATE PRIMITIVES - Modelam portas lógicas combinacionais padrão (AND, OR, NAND, NOR, XOR, XNOR) além dos buffers (BUF) e inversores (NOT).
→ SWITCH PRIMITIVES - Modelam transistores que atuam como chaves para controlar o fluxo de sinais (nmos, pmos, cmos, tran e rtran).

2 - As principais diferenças residem na origem e flexibilidade da primitiva:

a. Predefinidas são componentes lógicos padrão (and, or, nand, buf etc) incorporados à linguagem Veribg. funcionalidade nativa e reconhecida pelo simulador, sem a necessidade de uma definição comportamental.

b. Definidas pelo usuário (UDP) são componentes definidas pelo projetista para modelar um bloco lógico customizado/personalizado. São construídos apartir da palavra-chave PRIMITIVE e uma tabela verdade que descreve o comportamento. A principal desvantagem é que uma UDP só pode conter apenas uma saída.

A-013: Primitivas (prático)
> código nos arquivos de suporte enviados no formulário

REPOSITÓRIO DO GITHUB

github.com/ci-digital-inatel/SD112-INTRO-VERILOG

A-013 Primitivas (continuação)

III

3 - A função das primitivas condicionais (bufif0 notif0 - e suas análogas bufif1, notif1) é a implementação lógica de três estados (tri-state), essencial para o controle de barramentos compartilhados.

Atuam como um buffer ou inversor padrão, mas somente quando uma condição de controle é atendida:

a. bufif0 - Funciona como buffer (saída = entrada) se o sinal de controle for 0.

b. notif0 - Funciona como um inversor (saída = ~entrada) se o sinal de controle for 0.

Se a condição de controle não for atendida (ou seja, for 1), a saída de ambas as primitivas é colocada em estado de alta impedância (Z), efetivamente se desconectar do barramento e permitir que outro dispositivo o utilize.

IV

4 - A principal diferença está no gatilho/ativação que provoca a atualização da saída da primitiva.

Uma primitiva sensível ao nível (level-sensitive), análoga a um latch, atualiza sua saída sempre que o nível lógico (0 ou 1) de qualquer uma das suas entradas mudem. Sua saída depende diretamente do estado atual das entradas.

Uma primitiva sensível a borda (edge-sensitive), análoga a um flip-flop, só atualiza sua saída no instante de uma transição específica em um entrada (geralmente o clock), como uma borda de subida (0 para 1) ou descida (1 para 0). Mudanças nas outras entradas são ignoradas fora do evento.

V

5 - A diferença principal está na direção do fluxo do sinal e no propósito da modelagem das primitivas:

a. Unidirecionais (and, or, buf, etc) possuem portas de entrada e saída definidas (fixadas). O sinal flui em uma única direção, das entradas para a saída, para executar uma função lógica específica.

b. Bidirecionais (tran, rtranif0, tranif1, etc) não tem entradas ou saídas fixadas, mas terminais.

(continuação)
As bidirecionais modelam um caminho ou uma chave física, ao permitir que o sinal passe em qualquer direção entre seus terminais. São usadas para modelagem em nível de chaveamento (switch-level), como barramentos bidirecionais ou células de memória.

VI → 6 - A característica principal que diferencia é a presença ou ausência de memória (estado interno).

Uma UDP combinacional não possui memória. Sua saída é uma função direta e exclusiva das suas entradas atuais. Sua tabela verdade simplesmente mapeia cada combinação de valores de entrada para um valor de saída.

Uma UDP sequencial possui memória. Sua saída (que representa o próximo estado) depende tanto das entradas atuais quanto do estado anterior da primitiva. Sua tabela verdade precisa incluir uma coluna para o estado atual e pode ser definida como sensível (latch) ou à borda (flip-flop).