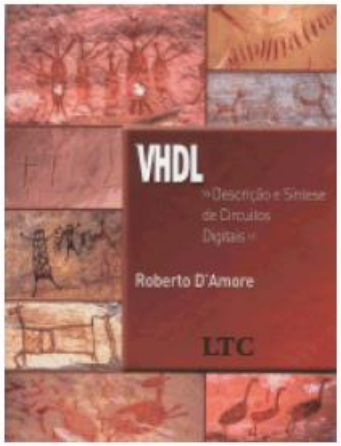
SEL5752/SEL0632 – Linguagens de Descrição de Hardware
Aula 01 – Apresentação

Prof. Dr. Maximiliam Luppe

Livro adotado:

VHDL - Descrição e Síntese de Circuitos Digitais
Roberto d'Amore

ISBN 85-216-1452-7
Editora LTC www.ltceditora.com.br



Para informações adicionais consulte: **www.ele.ita.br/~damore/vhdl**

Objetivos

- Introduzir novas abordagens no projeto de sistemas digitais com o uso de linguagens de descrição de *hardware* "HDL (*Hardware Description Languages*)" e "FPGA (*Field Programmable Gate Array*)".

Avaliação

- A avaliação será composta por listas de exercícios individuais (30%) e projeto de um sistema digital em grupo de até 3 alunos (70%). Não serão aceitos trabalhos individuais.
- O projeto será apresentado a partir da 7ª semana de aula e sua entrega terá como data máxima a 15ª semana de aula.

Introdução

Tópicos

- Histórico
- Aspectos gerais da linguagem
- Síntese de circuitos

- Entidade de projeto
- Classes de objetos: constante, variável e sinal
- Tipos
- Operadores
- Construção concorrente WHEN ELSE
- Construção concorrente WITH SELECT
- Processos e lista de sensibilidade
- Construção seqüencial IF ELSE
- Construção seqüencial CASE WHEN
- Circuitos síncronos

Histórico

- **Motivo do desenvolvimento:**
  - necessidade de uma ferramenta de projeto e documentação
    - projeto VHSIC *Very High Speed Integrated Circuit* do Departamento de Defesa dos Estados Unidos da América (DoD)

- **1981**
  - DoD patrocina o 1º encontro de especialistas
  - objetivo: discutir métodos para descrição de circuitos

- **1983**
  - DoD define os requisitos de uma linguagem padrão
  - Firmado contrato com IBM, Texas e Intermetrics
    - objetivo: desenvolvimento da linguagem e programas

- **1985**
  - apresentação da versão 7.2
  - direitos autorais do manual transferidos para o IEEE
    Institute of Electrical and Electronic Engineer
  - incumbências do IEEE:
    - definir um padrão para linguagem
    - manter futuros desenvolvimentos (novas versões etc.)

- **1987**
  - após revisões propostas por:
    - acadêmicos
    - representantes de industrias
    - governo dos Estados Unidos
      - definido o padrão IEEE 1076-1987

- **após 1987**
  - desenvolvidos os pacotes:
    - IEEE 1164 std_logic_1164
    - IEEE 1076.3 numeric_std numeric_bit
  - o que são pacotes (packages)?
    - local para armazenamento de informações de uso comum:
      - novos tipos de dados,
      - funções, etc.
  - motivos do desenvolvimento:
    - falta de um tipo de dado para modelar condições como:
      - alta impedância
      - níveis lógicos indeterminados
      - operações aritméticas com vetores (*arrays*)

Revisions

- IEEE 1076-1987 First standardized revision of ver 7.2 of the language from the United States Air Force.
- IEEE 1076-1993 (also published with ISBN 1-55937-376-8). Significant improvements resulting from several years of feedback. Probably the most widely used version with the greatest vendor tool support.
- IEEE 1076-2000. Minor revision. Introduces the use of *protected types*.
- IEEE 1076-2002. Minor revision of 1076-2000. Rules with regard to *buffer ports* are relaxed.
- IEEE 1076c-2007. Introduced VHPI, the VHDL Procedural Interface, which provides software with the means to access the VHDL model. The VHDL language required minor modifications to accommodate the VHPI.
- IEEE 1076-2008 (previously referred to as 1076-200x). Major revision released on 2009-01-26. Among other changes, this standard incorporates a basic subset of PSL (Property Specification Language), allows for generics on packages and subprograms and introduces the use of *external names*.
- IEEE 1076-2019. Major revision.

Related standards

- IEEE 1076.1 VHDL Analog and Mixed-Signal (VHDL-AMS)
- IEEE 1076.1.1 VHDL-AMS Standard Packages (stdpkgs)
- IEEE 1076.2 VHDL Math Package
- IEEE 1076.3 VHDL Synthesis Package (vhdlsynth) (numeric std)
- IEEE 1076.3 VHDL Synthesis Package – Floating Point (fphdl)
- IEEE 1076.4 Timing (VHDL Initiative Towards ASIC Libraries: vital)
- IEEE 1076.6 VHDL Synthesis Interoperability (withdrawn in 2010)
- IEEE 1164 VHDL Multivalue Logic (std_logic_1164) Packages

Aspectos gerais da linguagem

- **Suporta diversos níveis de hierarquia**
  - uma descrição pode ser: conjunto de descrições interligadas

- **Estilo de uma descrição**
  - um circuito pode ser descrito de diversas maneiras
    - níveis de abstração (exemplo: comportamental - rede de ligações)

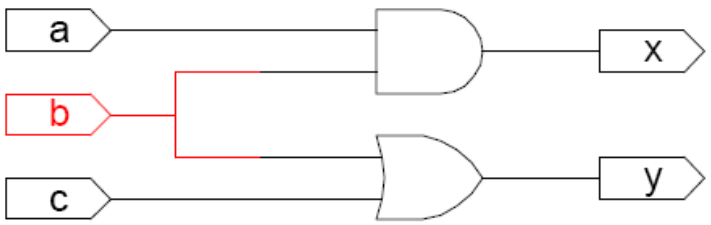
- **Uma descrição**
  - pode mesclar diferentes níveis de abstração (num mesmo código)

- **Ferramentas de síntese:**
  - suportam diferentes estilos de descrição
  - normalmente: modos preferenciais deve ser empregados

Aspectos gerais da linguagem

- **Linguagem concorrente**
  - ordem dos comandos: não importa
  - mudança de valor em um sinal:
    - acarreta a execução de todas os comandos envolvidos

- **Concorrência: exemplo**
  - alteração do valor em b:
    - execução conjunta dos comandos nas linhas 8 e 9
  - simulador
    - internamente executadas seqüencialmente
    - necessário um mecanismo interno para as avaliações
    - a ordem da avaliação dos comandos é irrelevante
    - resultado é sempre o mesmo



descrição VHDL

```
 1  ENTITY portas IS
 2    PORT (a, b, c : IN  BIT;
 3          x, y    : OUT BIT);
 4  END portas;
 5
 6  ARCHITECTURE teste OF portas IS
 7  BEGIN
 8    x <= a AND b;
 9    y <= c OR  b;
10  END teste;
```

- **Comandos seqüenciais:**
  - somente em regiões delimitadas no código
  - cada região é executada concorrentemente

- **Código seqüencial**
  - delimitado em regiões específicas:
    - subprogramas
    - processos
  - comandos próprios nestas regiões

- **Subprogramas**: procedimento (procedure) e função (function)
  - empregados em:
    - rotinas de conversão
    - outras operações não ligadas a síntese
  - síntese:
    - subprograma corresponde a um circuito
    - exemplo:
      multiplicador paralelo (composto de várias células interligadas)
      cada célula distinta → corresponde a um subprograma

- **Definição de biblioteca e pacote** (library) (package)
  - pacotes
    - armazenam:
      subprogramas, constantes, novos tipos etc.
    - evitam a repetição de definições comuns
    - exceto o pacote padrão (*standard package*),
      qualquer pacote deve ser declarado para se tornar visível
  - bibliotecas
    - local de armazenamento das informações compiladas
    - biblioteca padrão de armazenamento: work
    - diferentes locais podem ser definidos:
      - úteis para trabalhos em equipes (local comum de armazenamento)
      - organização do projeto

- **Tipos**
  - Exemplo de tipos pré-definidos no pacote padrão (STD):

    bit, boolean, integer

  - tipo bit

    exemplo de valores: 0 e 1

  - tipo boolean:

    exemplo de valores: true e false

  - tipo std_logic → um novo tipo definido no pacote std_logic_1164

    exemplo de valores:

    0 1 U (não inicializado) X (desconhecido) Z (alta impedância)

- **Definição de novos tipos**

  - usuário pode definir novos tipos

    exemplo: estados de uma máquina:

```
TYPE estado IS (parado, falha, correto, caso_1);
```

cada estado deve ser representado por um código

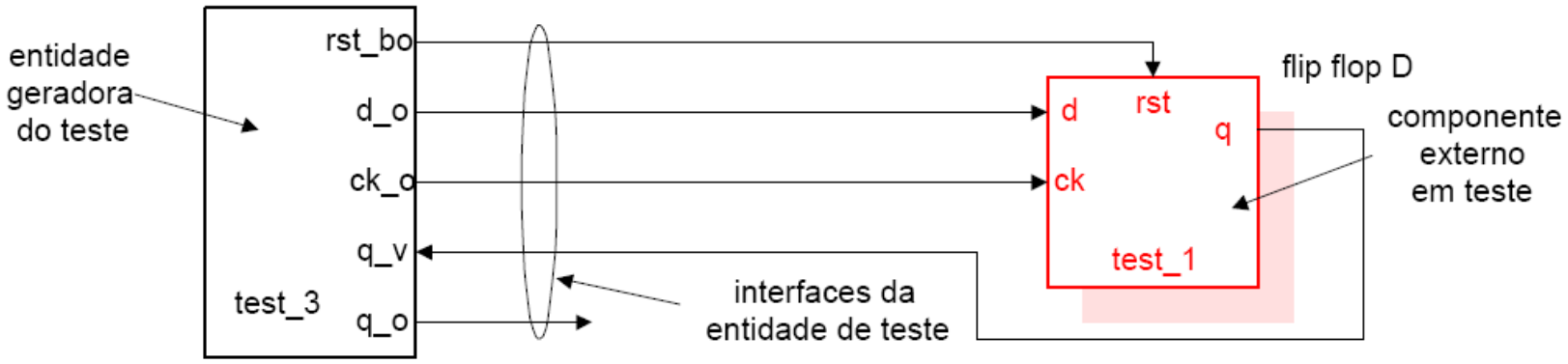
- exemplo de codificação:

parado = 0001, falha = 0010, correto = 0100, caso_1 = 1000

- forma de codificação dos estados:

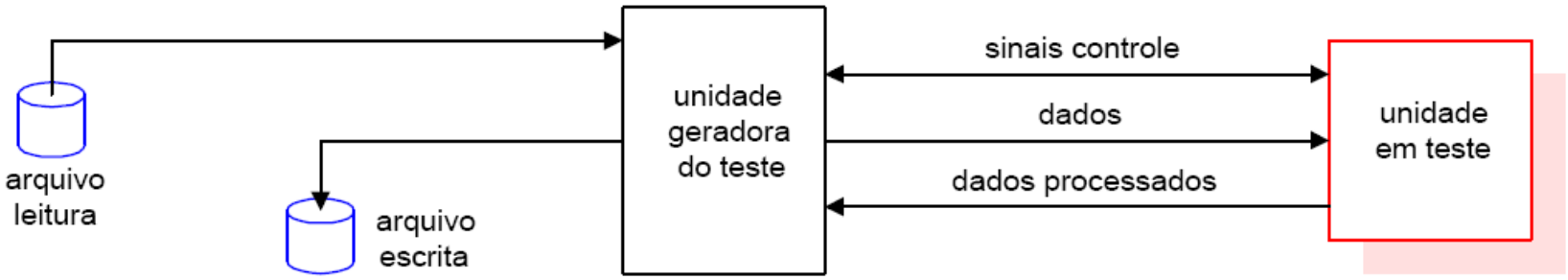
  executado pela ferramenta de síntese

- **Procedimentos de teste**
  - um descrição VHDL pode gerar estímulos de teste
    exemplo:



  - verificação funcional da descrição
  - após a síntese: temporização disponível
    - verificação de restrições como:
      hold time
      setup time

- **Operações com arquivos**
  - não suportadas por ferramentas de síntese
  - são por demais abstratas
  - aplicação: operações de teste de outras descrições



Síntese de circuitos

- **VHDL**: não foi concebida para síntese de circuitos
  - conseqüência:
    - nem todas construções da linguagem são suportadas na síntese

- **Motivos da limitação**:

  1- falta de correspondência entre: construção ⇔ circuito real

  exemplo: *flip flop* com dois terminais de relógio

  código pode ser simulado ⇔ não existe o *flip flop*

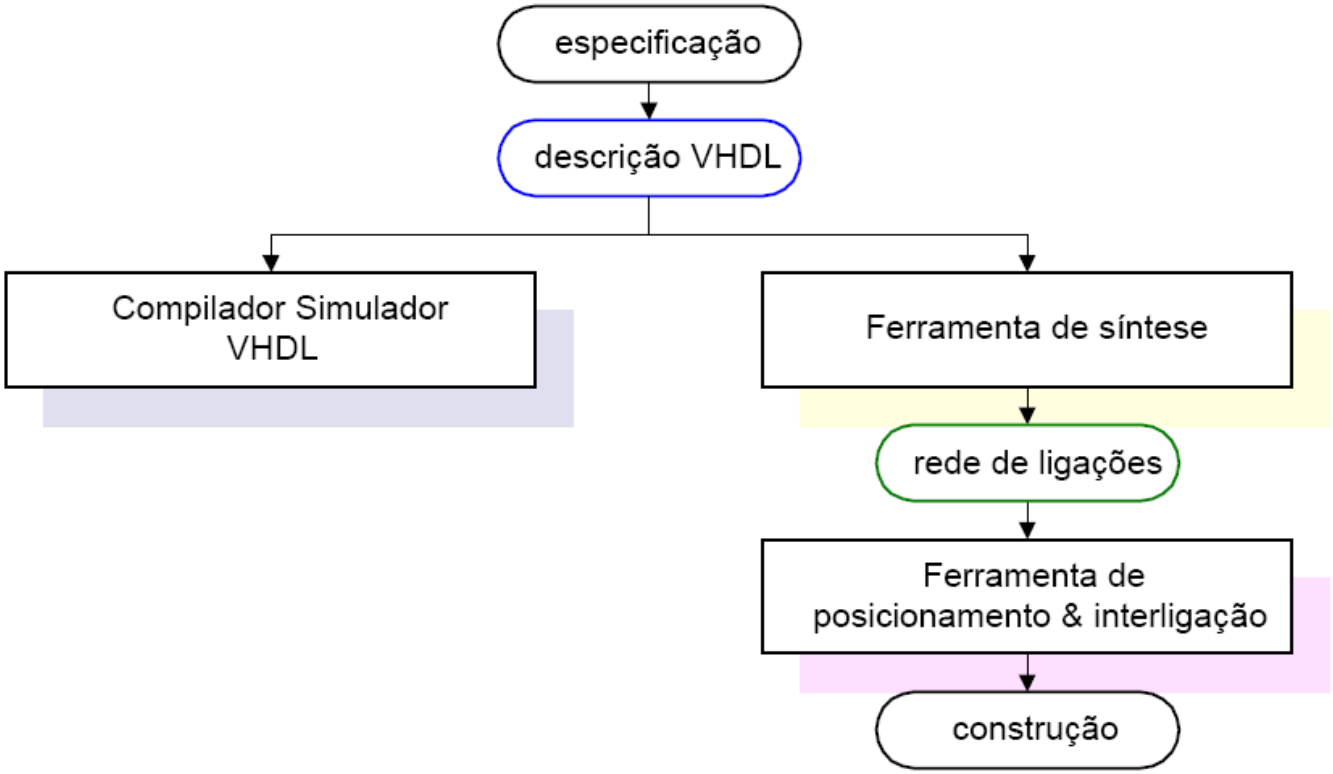
  2- impossibilidade da síntese direta

  exemplo: multiplicação de dois números reais

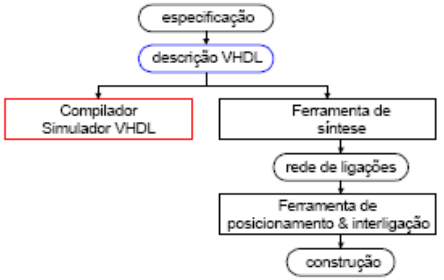
  código pode ser simulado ⇔ circuito muito complexo

Etapas gerais de um processo de síntese

- **Elaboração da descrição**
- **Síntese da descrição**
- **Posicionamento e interligação**



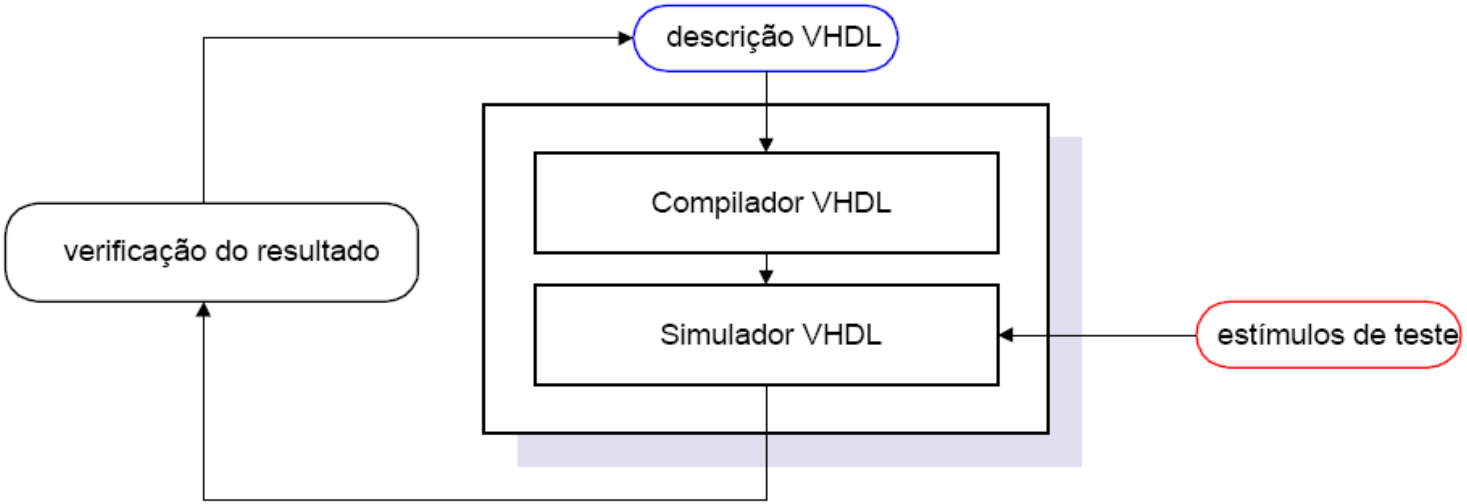
1- Elaboração da descrição



- um mesmo circuito:
  - várias formas de descrição possíveis
  - diferentes níveis de abstração

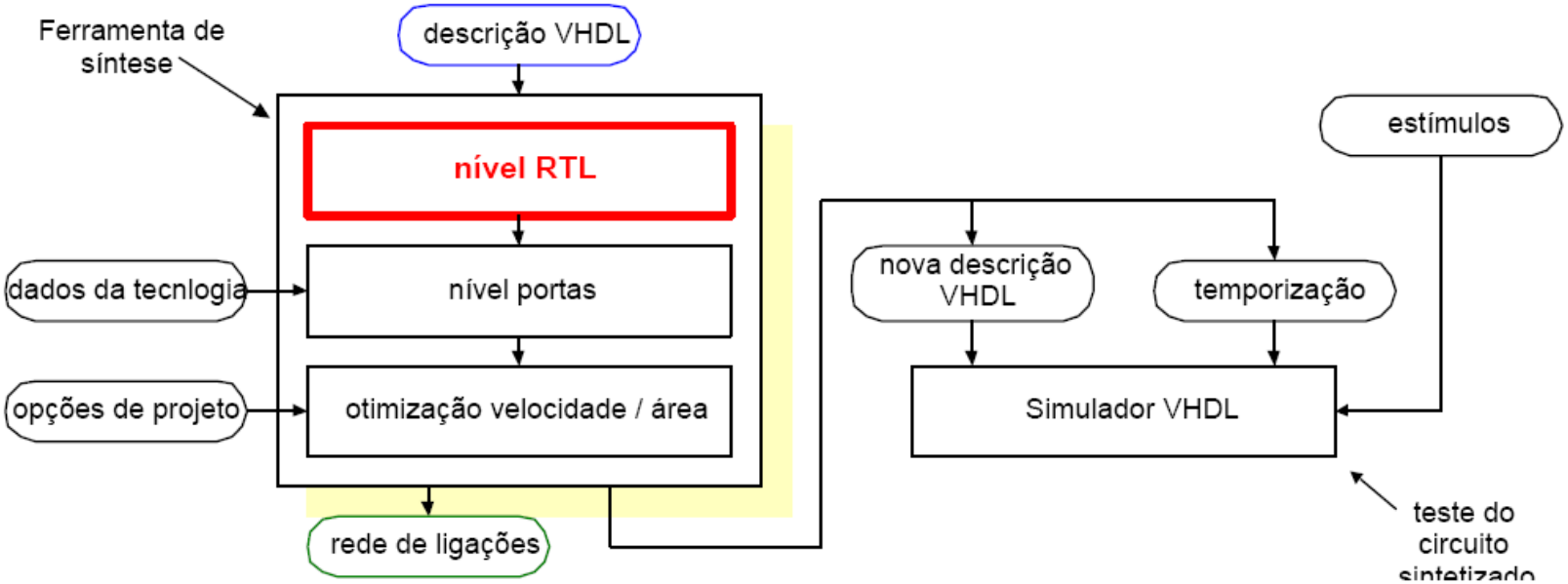
- proposta inicial da descrição:
  - pode conter estruturas muito abstratas para síntese direta

- processo interativo:
  - permite atingir o grau de detalhamento necessário para síntese



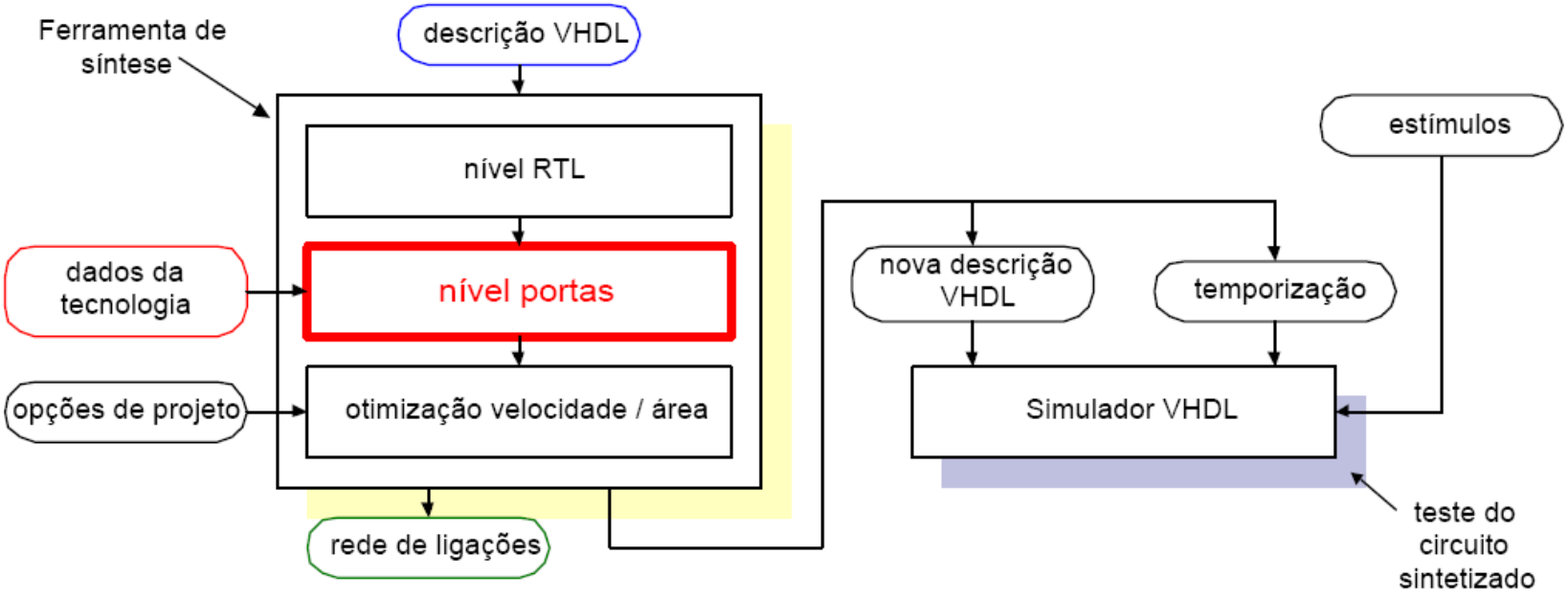
2- Síntese da descrição - (verificação de erros - nível RTL)

- verificação de erros de sintaxe
  - inferência das estruturas necessárias: gerado circuito nível RTL
    - emprega primitivas da ferramenta de síntese:
      (portas lógicas, somadores, comparadores etc.)
    - circuito nível RTL: não é associado a nenhuma tecnologia



2- Síntese da descrição - (nível portas)

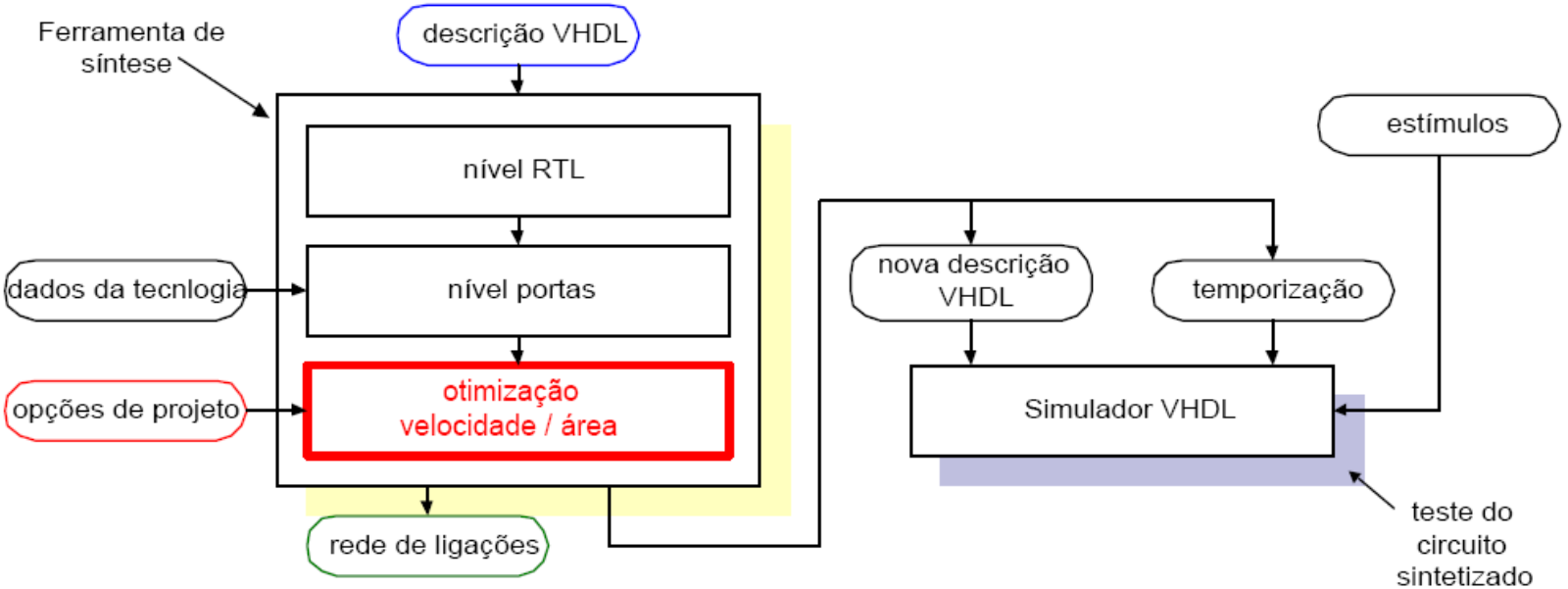
- novo circuito considerando a tecnologia utilizada
  - emprega: primitivas da tecnologia empregada
    módulos otimizados fornecidos pelo fabricante
    (exemplo de módulos: somadores, contadores)



2- Síntese da descrição - (otimização)

- definição de opções de projeto

velocidade / área (normalmente conflitantes)



2- Síntese da descrição - (ilustração das operações)

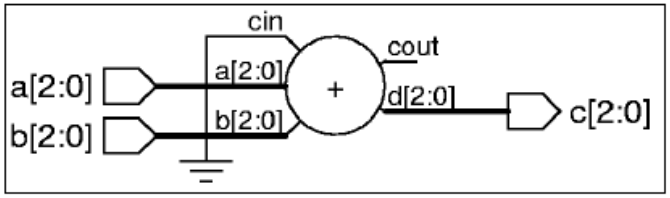
- descrição do circuito
- circuito nível RTL sintetizado
- circuito nível portas (final)

descrição VHDL

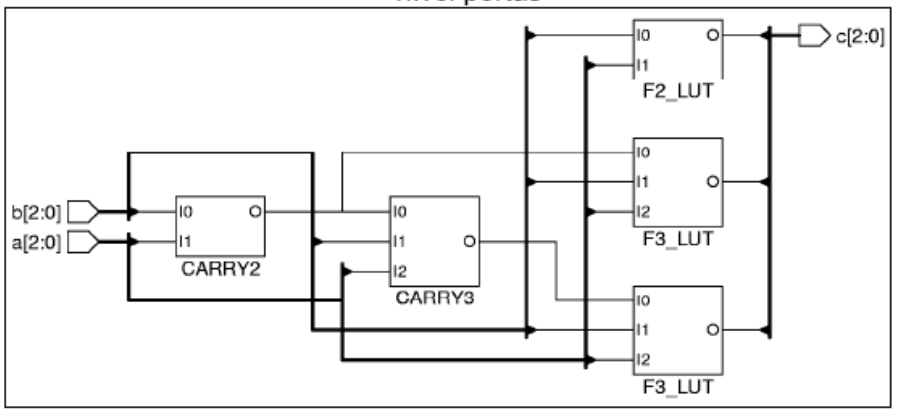
```
ENTITY soma IS
  PORT (a, b : IN  INTEGER RANGE 7 DOWNTO 0;
        c    : OUT INTEGER RANGE 7 DOWNTO 0);
END soma;

ARCHITECTURE teste OF soma IS
BEGIN
  c <= a+b;
END teste;
```

nível RTL



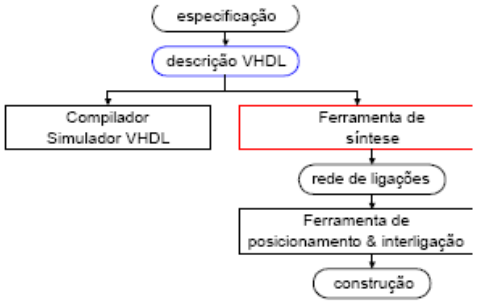
nível portas

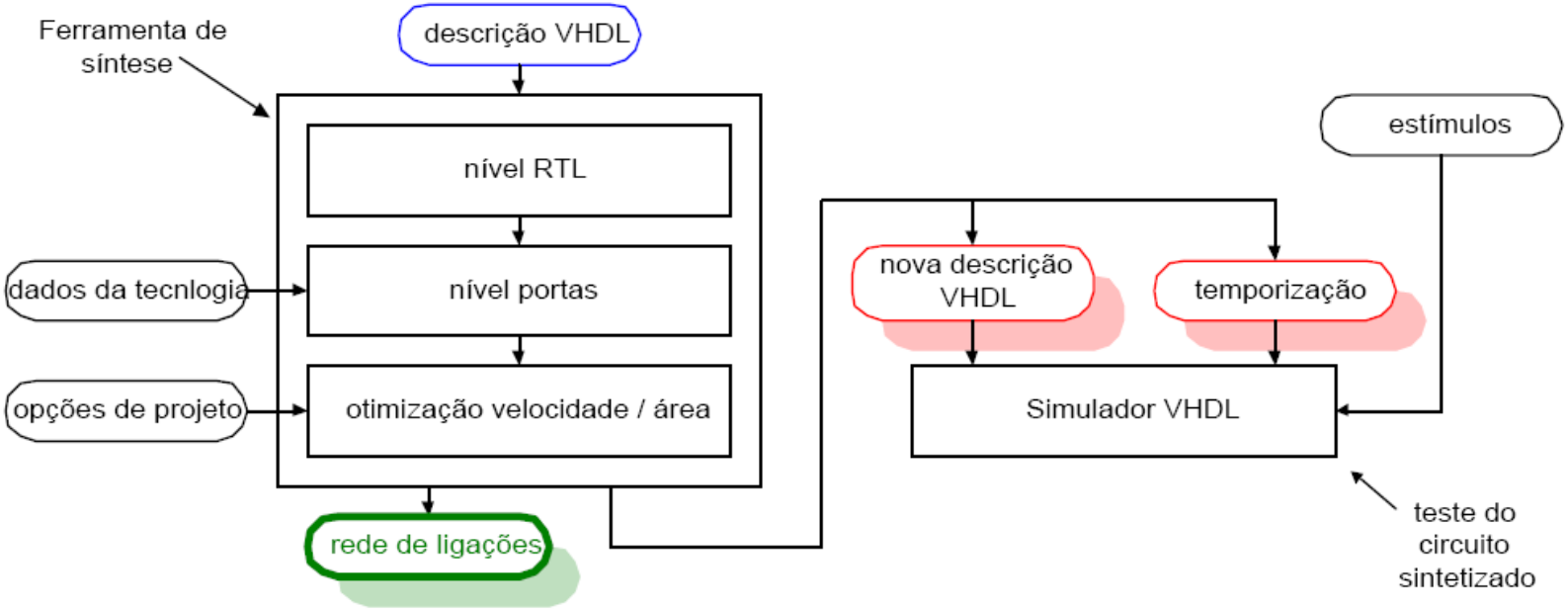


otimização velocidade / área

2- Síntese da descrição - (arquivos gerados)

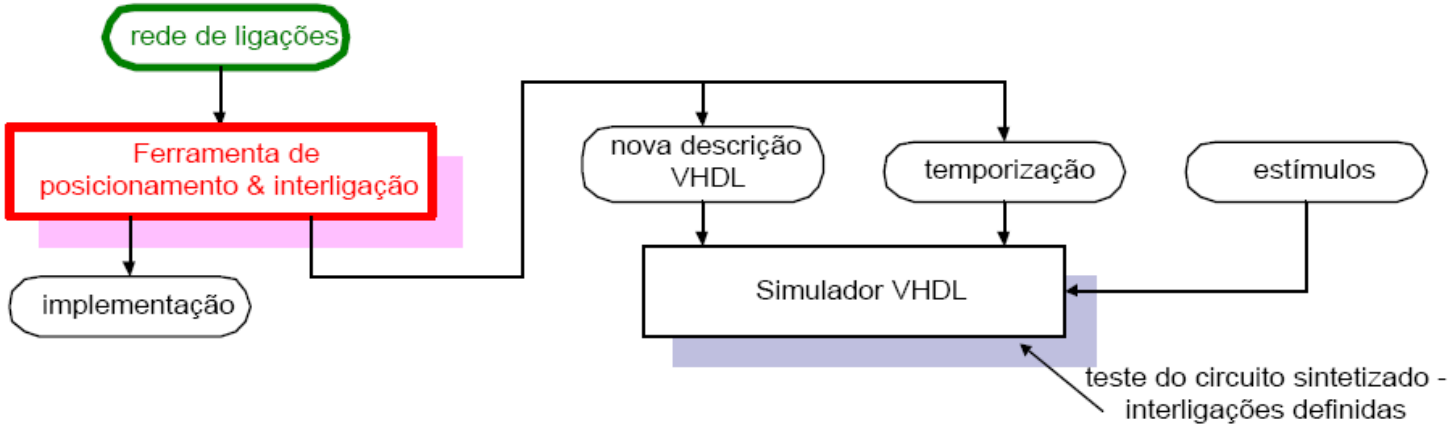
- rede de ligações
  - empregado pela próxima ferramenta
    - exemplo: formato EDIF
- nova descrição VHDL
  - contém informações preliminares de temporização
  - simulações contendo informações de temporização





3- Posicionamento e interligação (*placement routing*)

- ferramenta apropriada:
  - posiciona e interliga as primitivas / componentes
- dispositivo empregado na implementação:
  - *FPGA* - dispositivo lógico programável
  - *ASIC* - circuito integrado de aplicação específica
- arquivo de temporização mais preciso:
  interligações definidas (e atrasos por elas gerados)
- novas simulações podem ser executadas



Fim.