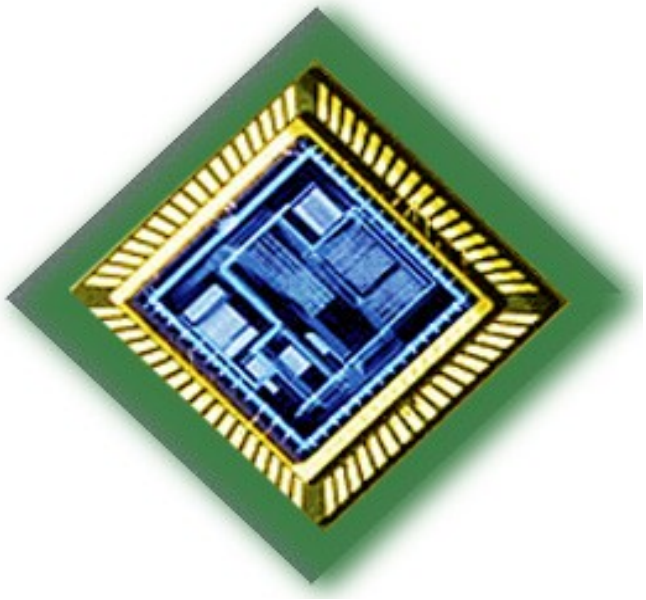


eTBc
Easy TestBench Creator

Curso do projeto Brazil-IP

Elmar Melcher
UFCG
elmar@dsc.ufcg.edu.br
http://lad.dsc.ufcg.edu.br/ip

Roteiro

- Introdução
  - Motivação
  - Objetivos
- Como usar a ferramenta?

Motivação

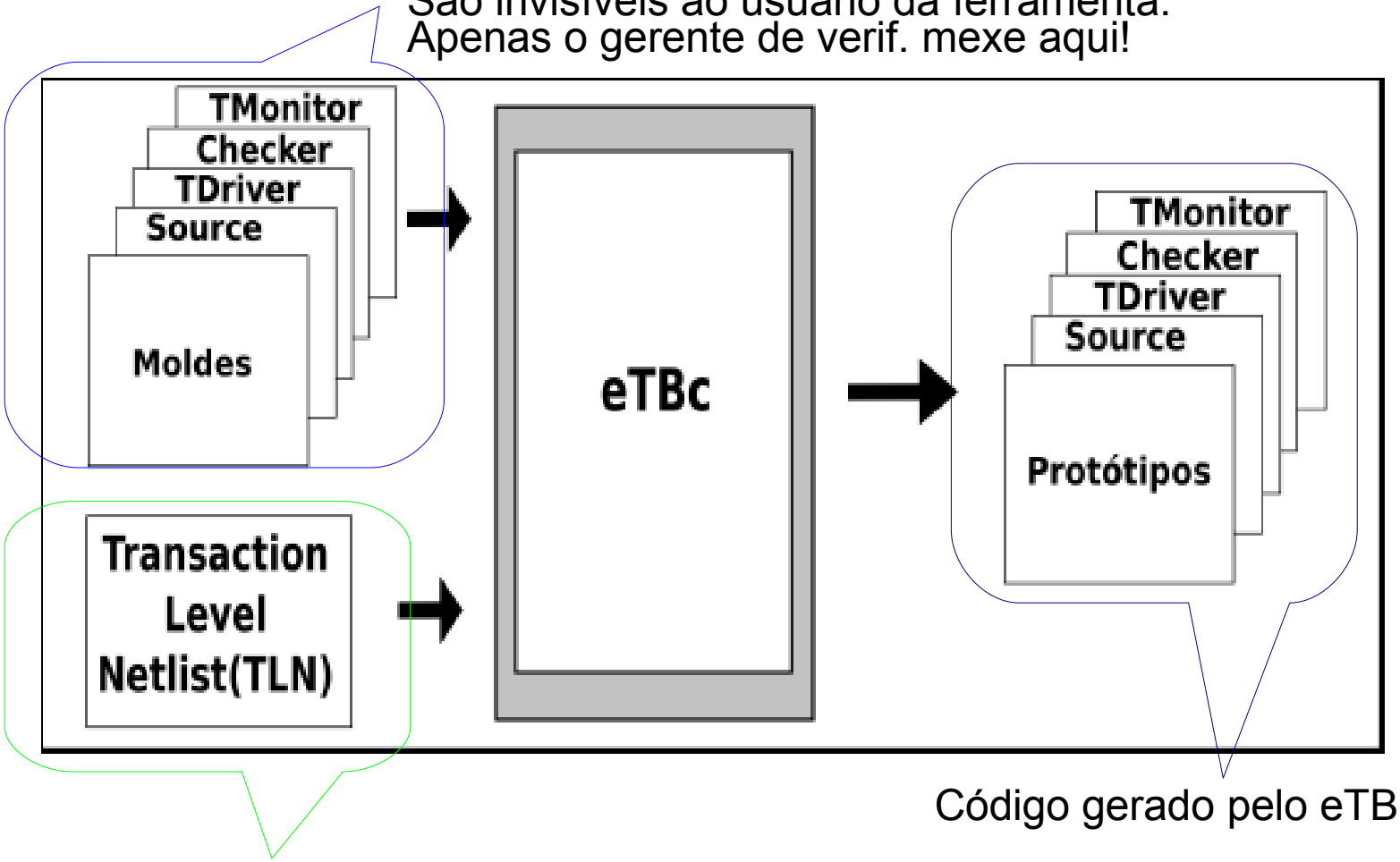
- A complexidade da verificação funcional tende a crescer exponencialmente com relação ao tamanho do hardware a ser verificado.
- O processo de verificação consome a maior parte dos recursos em um projeto de hardware.
  - Ferramentas que ajudem esse processo de verificação são bem vindas!

Motivação

- eTBc (easy TestBench creator)
  - Ferramenta de apoio à metodologia de verificação funcional BVM / VeriSC.
  - Ajuda a acelerar o processo de construção de testbenches.
  - Funciona como um gerador de código.

Introdução - eTBc

São invisíveis ao usuário da ferramenta.
Apenas o gerente de verif. mexe aqui!



Código gerado pelo eTBc

Código Fonte descrito pelo usuário de eTBc

Objetivos

- Os objetivos da ferramenta eTBc:
  - Suporte a metodologia BVM / VeriSC.
  - Automatizar o processo de construção de ambientes de simulação (testbenches)
  - Aumentar a velocidade em que as tarefas de verificação são executadas.

Objetivos

- Melhoria do processo de desenvolvimento de IP-cores.
- Geração semi-automática de protótipos de cada um dos elementos de um testbench além das ligações entre os mesmos.
  - Protótipos são os elementos gerados pelo eTBc.
  - São protótipos por estarem ainda incompletos considerando todo o plano de verificação funcional.
  - Precisam ser "completados" para ficarem bons para o uso.

eTBc - Tipos de Linguagens

- eTBc Design Language (eDL)
  - Linguagem usada para criar a TLN.
- eTBc Template Language (eTL)
  - Linguagem interna do eTBc
  - Os templates estão prontos e são transparentes ao desenvolvedor (engenheiro de verificação).
  - Apenas o gerente de verificação funcional mexe nesta linguagem.

eDL

- Tem como objetivo a modelagem de um IP-core em nível de transação.
  - Elaborar as TLNs (Transaction Level Netlist) referentes ao projeto de verificação funcional.
  - TLN - arquivo fornecido como entrada para o eTBc.

eDL – Palavras reservadas

- struct
  - Permite a definição de uma transação e de seus sinais.
  - A transação conjuntamente com os sinais representam o TLD (Transaction Level Data).
  - Dentro de struct:
    - trans
    - signals

eDL – Palavras reservadas

- trans
  - Usada no escopo de um "struct" e serve para declarar transações.
  - Dados do escopo de uma transação seguem a mesma sintaxe que os dados na linguagem C.
  - Exemplo dos tipos primitivos:
    - int, char ,short, long, float, double, bool e unsigned.
      - Combinações desses tipos também são permitidos de acordo com o padrão da linguagem C : long int, long long int, unsigned int...

eDL – Palavras reservadas

- Correspondência entre eDL e SystemVerilog
  - bool --> bit
  - char --> byte
  - short int --> shortint
  - int --> int
  - unsigned int --> int unsigned
  - long int --> int
  - long long int --> longint
  - float --> shortreal
  - double --> real

eDL – Palavras reservadas

- signals
  - Usada no escopo de um “struct” e serve para declarar sinais em nível RTL (Register Transfer Level).
  - Dentro de “signals” é permitido:
    - signed
    - unsigned
    - bool
    - invert

eDL – Palavras reservadas

- signed
  - Usado para a declaração de um identificador com sinal numérico.
- unsigned
  - Usado para a declaração de um identificador sem sinal numérico.

eDL – Palavras reservadas

- bool
  - Usado para a declaração de um identificador com um bit.
- invert
  - Serve para inverter o sentido do fluxo normal do sinal.
  - É usado para criar fios de protocolo, onde há sinais em sentidos opostos.

eDL – Palavras reservadas

- module
  - Usado para declaração de blocos funcionais.
  - Escopo de um module:
    - input
    - output
    - channel

eDL – Palavras reservadas

- input
  - Usado para criar as interfaces de entrada de um módulo.
  - Não confundir com entrada de sinal.
  - As interfaces de entrada são vias por onde entram transações em um módulo.

eDL – Palavras reservadas

- output
  - Usado para criar as interfaces de saída de um módulo.
  - As interfaces de saída são vias por onde saem transações de um módulo.

eDL – Palavras reservadas

- channel
  - Usado para fazer a comunicação entre dois módulos.
  - Função de ser o veículo das transações que trafegam entre dois módulos.

eDL- Exemplos

- Modelagem de um IP-core que converte YUV em RGB

eDL- Exemplo- Arquivo yub2rgb.tln

```
1.  struct yuv{
2.      trans {
3.        int y;
4.        int u;
5.        int v;
6.      }
7.      signals {
8.        signed[8] canal;
9.        bool valid;
10.       invert bool ready;
11.     }
12.}
```

Definição de uma estrutura chamada yuv

Definição de transações da estrutura yuv

Definição de sinais da estrutura yuv

Um sinal de 8 bits

Um sinal de um bit

Um sinal de um bit com sentido invertido

continuação...

```
12. struct rgb{
13.     trans {
14.       int r;
15.       int g;
16.       int b;
17.     }
18.     signals {
19.       signed[8] canal;
20.       bool valid;
21.       invert bool ready;
22.    }
23.}
```

```
24. module yuv2rgb{
25.
26.     input yuv entrada_yuv;
27.     output rgb saida_yuv;
28.}
```

Definição de um módulo funcional chamado yuv2rgb

Definição de uma interface de entrada cujo tipo é a estrutura yuv

Definição de uma interface de saida cujo tipo é a estrutura rgb

Como usar a ferramenta?

- Usar terminal de linhas de comando, obedecendo a sintaxe abaixo:

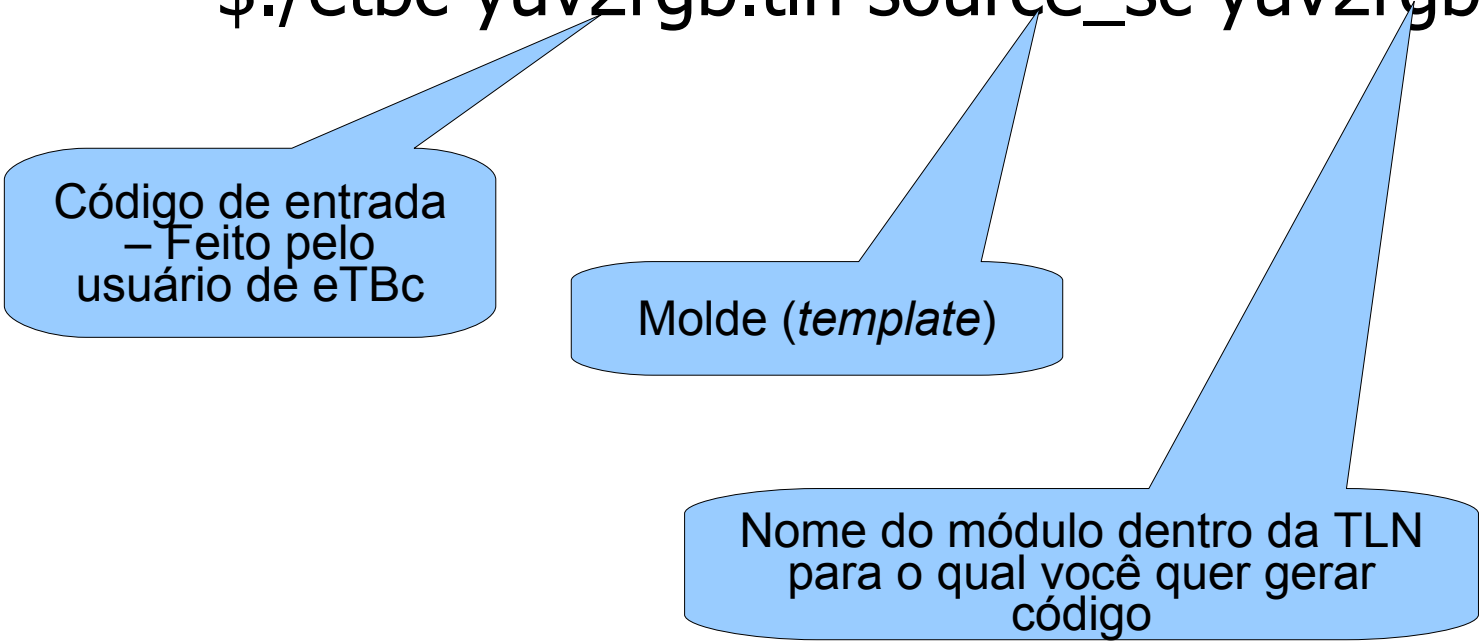
eTBc [tln] [molde] [modulo]

  - **tln:** O nome da TLN de entrada.
  - **molde:** O nome do molde (template) que será usado para a geração do protótipo (Source, TDriver, TMonitor, Checker ...) desejado.
  - **modulo:** O nome da instância ou do módulo do TLN para a qual os protótipos serão gerados.

Como usar a ferramenta?

- Exemplo para o yuv2rgb

$./etbc yuv2rgb.tln source_sc yuv2rgb



Página do eTBc

http://lad.dsc.ufcg.edu.br/pmwiki.php?n=Lad.ETBc