VHDL

Sinais e Tipos de Dados

Rafael Corsi Ferrão - IMT

rafael.corsi@maua.br
http://www.maua.br

9 de outubro de 2014

Conteúdo

Classe de objetos

Ieee Library

1 Classe de objetos

2 Ieee Library

Conteúdo

1 Classe de objetos

2 Ieee Library

Classe de objetos

Ieee Library

Definição

Contantes é um objeto com um valor estático.

Exemplo:

```
constant nome_a : tipo_x := valor_inicial
```

```
entity SS_controller is
 port(
       .........
  );
end SS_controller;

architecture Behavioral of SS_controller is

constant vectora: STD_LOGIC_VECTOR (7 downto 0) := X"0F";
constant vectorb: STD_LOGIC_VECTOR (7 downto 0) := "0101_0101";
constant teste  : STD_LOGIC := '1';

begin
 ....
end Behavioral;
```

Classe de objetos

Ieee Library

Definição

Sinais são utilizados para a ligação interna de uma entidade, não possuem definição de direção

Sinais são análogos a trilhas em placa de circuito impresso. Utiliza-se para conectar blocos, receber valores de entrada e propagar valores para saídas.

Classe de objetos
Ieee Library

Declara-se os sinais entre a definição da arquitetura e o começo da descrição (begin):

```
entity SS_controller is
 port(
       .........
  );
end SS_controller;

architecture Behavioral of SS_controller is

    signal s_SSEG_CA : STD_LOGIC_VECTOR (7 downto 0);
    signal s_SSEG_AN : STD_LOGIC_VECTOR (7 downto 0);
    signal cnt       : STD_LOGIC := '1';

begin
 ....
end Behavioral;
```

Classe de objetos
Ieee Library

Os sinais podem ser inicializados em qualquer valor (respeitando o padrão ieee) :

```
signal vectora: STD_LOGIC_VECTOR (7 downto 0) := X"0F";
signal vectorb: STD_LOGIC_VECTOR (7 downto 0) := "0101_0101";
signal teste  : STD_LOGIC := '1';
```

Um sinal pode receber outro sinal ou uma porta de entrada:

```
vectorb    <= vectora;
vectora(1) <= teste;
```

Classe de objetos

Ieee Library

Outras classes que vermos futuramente:

- FILE
- VARIABLE

Classe de objetos

Ieee Library

1 Classe de objetos

2 Ieee Library

Classe de objetos

Ieee Library

Define-se no padrão std_logic_1164 os seguintes tipos (principais):

- **bit** : valores ('0', '1');
- **bit_vector** : Um vetor de bits
- **boolean** : (false, true);

Classe de objetos

Ieee Library

Define-se no padrão std_logic_1164 os seguintes tipos (principais):

- **bit** : valores ('0', '1');
- **bit_vector** : Um vetor de bits
- **boolean** : (false, true);
- **std_logic** : ('U','X','0','1','Z','W','L','H','-')
- **std_logic_vector** : Um vetor de std_logic

Sendo que o tipo **std_logic** pode assumir os seguintes valores :

- U : Uninitialized
- X : Forcing unknown
- 0 : Forcing 0
- 1 : Forcing 1
- Z : High impedance
- W : Weak unknown
- L : Weak 0
- H : Weak 1
- - : Don't care

As operações definidas em std_logic_1164 são:

- lógicas:
  - and, nand, or, nor, xnor, not
- conversões:
  - To_bit()
  - To_std_logic()
  - To_bit_vector()
  - To_std_logic_vector()

Define-se no pacote numeric_std os seguintes tipos (principais):

- **integer** : -2_147_483_647 to 2_147_483_647;
- **natural** : 0 to integer'HIGH;

Define-se no pacote numeric_std os seguintes tipos (principais):

- **integer** : -2_147_483_647 to 2_147_483_647;
- **natural** : 0 to integer'HIGH;
- **real** : TYPE real IS RANGE -2_147_483_647.0 TO 2_147_483_647.0;

Define-se no pacote numeric_std os seguintes tipos (principais):

- **integer** : -2_147_483_647 to 2_147_483_647;
- **natural** : 0 to integer'HIGH;
- **real** : TYPE real IS RANGE -2_147_483_647.0 TO 2_147_483_647.0;
- **UNSIGNED** : É um **VETOR** que não possui definição de sinal, todos os bits são usados para armazenar o valor do dado;
- **SIGNED** : É um **VETOR** que possui definição de sinal (positivo,negativo)

Bits	Unsigned Value	Signed Value
011	3	3
010	2	2
001	1	1
000	0	0
111	7	-1
110	6	-2
101	5	-3
100	4	-4

Classe de objetos

Ieee Library

No caso dos números inteiros, reais e naturais podemos limitar os seus tamanhos :

```
signal <integer_name> : integer range <low> to <high>;
signal <natural_name> : natural range <low> to <high>;
signal <real_name>    : real range <low> to <high>;
```

Exemplo de utilização :

```
signal cnt1 : integer range -2 to 2; -- 2 bits
                                     -- + 1bit para o sinal
signal cnt2 : natural range 0 to 16; -- 4 bits
signal cnt3 : integer range 0.0 to 1000.0; -- ???
```

Classe de objetos

Ieee Library

Podemos criar vetores com os tipos UNSIGNED e SIGNED

```
signal <unsigned_name> : UNSIGNED( <low> to <high> )
signal <signed_name>   : SIGNED  ( <low> to <high> )
```

Exemplo de utilização :

```
signal rs_SUM_RESULT : signed(4 downto 0)   := (others => '0');
signal ru_SUM_RESULT : unsigned(4 downto 0) := (others => '0');
```

Os seguintes tipos não são sintetizáveis, ou seja, não podem ser implementando numa FPGA/ASIC e sevem puramente em ambientes simulados:

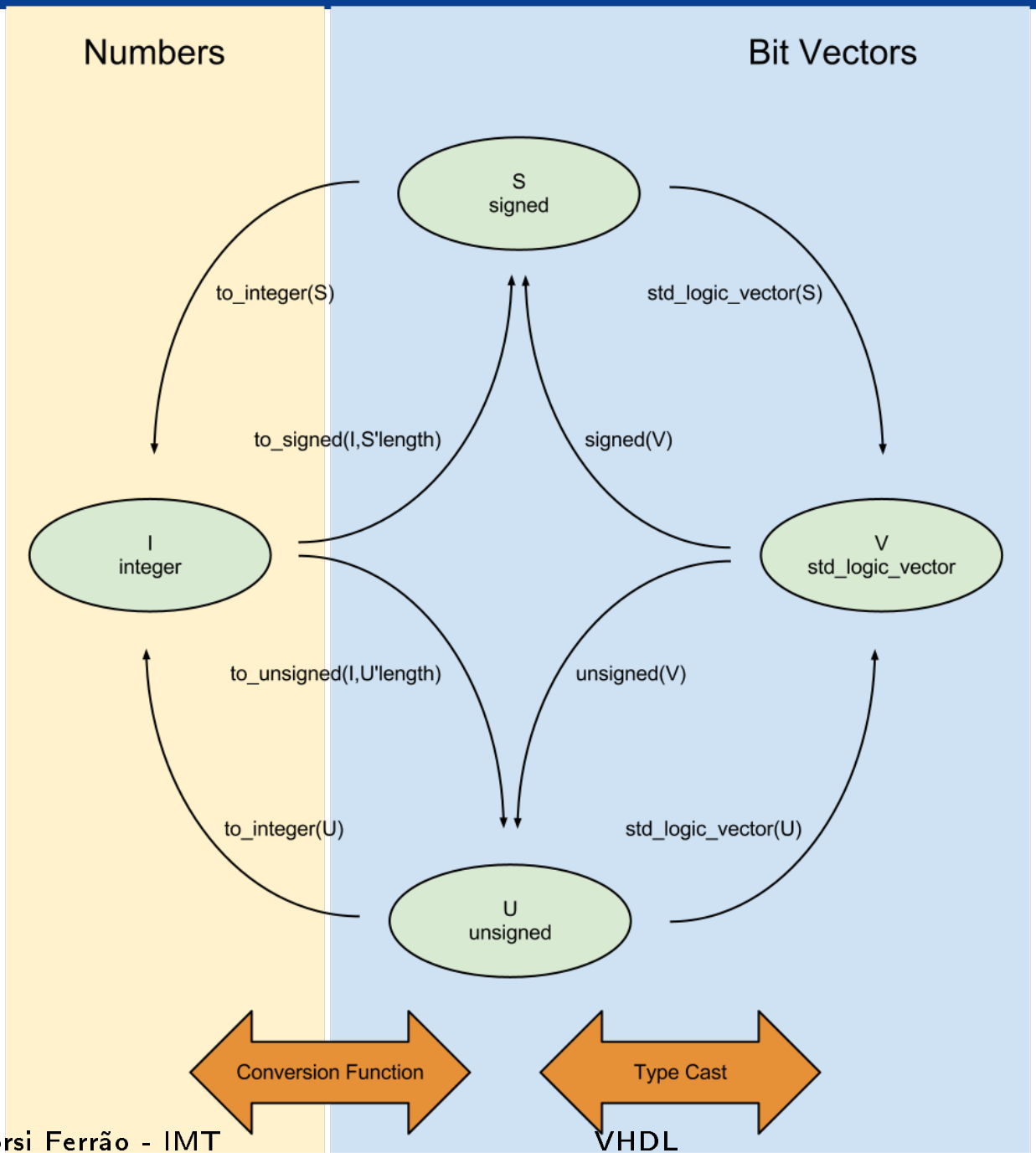
- REAL : Devido a complexidade de implementação
- TIME : Intervalo de tempo

As principais operações definidas em numeric_std são:

- + : soma
- - : subtração
- * : multiplicação
- / : divisão
- shift_left : rotaciona para esquerda
- shifit_right : rotaciona para direita

Verificar o arquivo **Material de Apoio/ VHDL/ numeric_std.tex** para todas as possibilidades

Conversões


Numbers
Bit Vectors
S
signed
to_integer(S)
std_logic_vector(S)
to_signed(I,S'length)
signed(V)
I
integer
V
std_logic_vector
to_unsigned(I,U'length)
unsigned(V)
to_integer(U)
std_logic_vector(U)
U
unsigned
Conversion Function
Type Cast

Classe de objetos

Ieee Library

Exemplo de conversão de dados :

```
architecture bhv of entidade1 is

signal ex1 : SIGNED(3 downto 0);
signal ex2 : STD_LOGIC_VECTOR(3 downto 0);
signal ex3 : INTEGER RANGE 0 to 7;

begin
  ex2 <= STD_LOGIC_VECTOR(ex1);
  ex3 <= to_integer(SIGNED(ex2));
  ex1 <= to_unsigned(i, 4);
end bhv;
```

- Porque eu não posso converter de STD_LOGIC_VECTOR para INTEGER direto ?
- Note que devemos dizer o tamanho do vetor na conversão de Inteiro para -> UNSIGNED/ SIGNED