Arquitetura e Organização de Computadores
Turma C 2021/01

Projeto e Simulação de uma
ULA em VHDL

Objetivo: projetar, simular e sintetizar uma versão da ULA do RISC-V de 32 bits no ambiente ModelSim-Altera ou EdaPlayground.

Características:

- Duas entradas de dados: *A* e *B*
- Uma Saída de dados: *Z*
- Sinal *cond*: *true* ou *false*, válido para as operações de comparação (*SLT, SLTU, SGE, SGEU, SEQ, SNE*)
- Operações:

Operação	Significado	OpCode
ADD A, B	Z recebe a soma das entradas A, B	0000
SUB A, B	Z recebe A - B	0001
AND A, B	Z recebe a operação lógica A and B, bit a bit	0010
OR A, B	Z recebe a operação lógica A or B, bit a bit	0011
XOR A, B	Z recebe a operação lógica A xor B, bit a bit	0100
SLL A, B	Z recebe a entrada A deslocada B bits à esquerda	0101
SRL A, B	Z recebe a entrada A deslocada B bits à direita sem sinal	0110
SRA A, B	Z recebe a entrada A deslocada B bits à direita com sinal	0111
SLT A, B	Z = 1 se A < B, com sinal	1000
SLTU A, B	Z = 1 se A < B, sem sinal	1001
SGE A, B	Z = 1 se A ≥ B, com sinal	1010
SGEU A, B	Z = 1 se A ≥ B, sem sinal	1011
SEQ A, B	Z = 1 se A == B	1100
SNE A, B	Z = 1 se A != B	1101

Interface:

```
entity ulaRV is
      generics (WSIZE : natural := 32);
      port (
            opcode    : in std_logic(3 downto 0)
            A, B      : in std_logic_vector(WSIZE-1  downto 0);
            Z         : out std_logic_vector(WSIZE-1  downto 0)
            cond      : out std_logic);
end ulaRV;
```

Utilizar como referência inicial o código visto em aula.

onde:

- *opcode* indica a operação a ser realizada
- *A e B*: operandos, 32 bits.
- *Z*: saída, 32 bits
- *cond*: indicação resultado da comparação

Bibliotecas: utilizar a biblioteca *numeric_std* do VHDL para as operações aritméticas sobre os vetores lógicos.

Simulação e Verificação: simular o funcionamento da ULA de forma a verificar o funcionamento de cada uma das suas operações. Verificar igualmente a geração do sinal *cond*. Utilizar o ModelSim Altera para a simulação, desenvolvendo um *testbench* para acionamento dos sinais.

Testar todas as operações da ULA.

Estrutura do *testbench*:

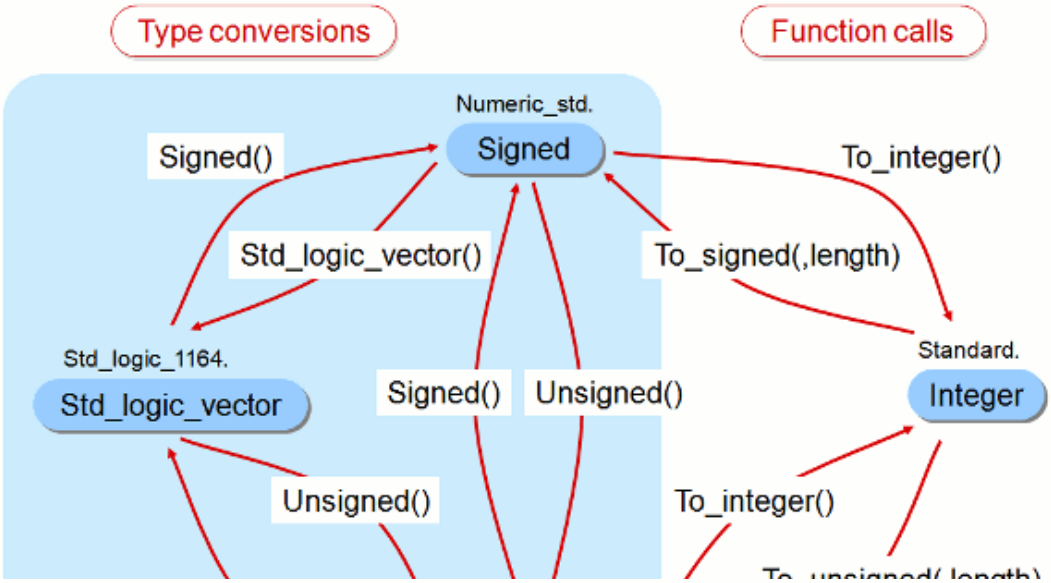
```
LIBRARY ieee;
USE ieee.std_logic_1164.all;
use ieee.numeric_std.all;

ENTITY ula_tb IS
END ula_tb;

ARCHITECTURE tb_arch OF ula_tb IS
■  declaração de sinais
■  declaração do componente ULA
   begin
■  instanciação da ULA
■  process
       ○  gera estímulos
       ○  wait for
   end process
end tb_arch;
```

A verificação deve incluir a execução de ao menos um teste para cada operação da ULA. As operações aritméticas devem ser testadas para resultado zero, negativo, positivo.

Entrega:

- Um relatório contendo:
  - Uma breve descrição do trabalho
  - Explique em suas palavras a diferença entre as comparações com e sem sinal.
  - Telas da simulação com as formas de onda dos sinais
  - Incluir o código da ULA e do *testbench*

Numeric Std Conversions



Summary of NUMERIC_STD

```
+  -  *  /   rem  mod
<  <=  >  >=  =  /=
```

UNSIGNED	UNSIGNED
UNSIGNED	NATURAL
NATURAL	UNSIGNED
SIGNED	SIGNED
SIGNED	INTEGER
INTEGER	SIGNED

```
sll srl rol ror
```

UNSIGNED	INTEGER
SIGNED	INTEGER

```
not and or nand nor
xor xnor
```

UNSIGNED	UNSIGNED
SIGNED	SIGNED

```
TO_INTEGER   [UNSIGNED          ] return INTEGER
TO_INTEGER   [SIGNED            ] return INTEGER
TO_UNSIGNED  [NATURAL,  NATURAL] return UNSIGNED
TO_SIGNED    [INTEGER,  NATURAL] return SIGNED
RESIZE       [UNSIGNED, NATURAL] return UNSIGNED
RESIZE       [SIGNED,   NATURAL] return SIGNED
```