

Conceitos de sinais e variáveis

Disciplina:
Laboratório de Sistemas Digitais

Professor:
Jhonattan Córdoba Ramírez

Alunos:
João Pedro Copelli - 2021014414
João Victor Gomes - 2020072690
Marcelle Christine Aquino Silva - 2021014546

7 de outubro de 2022

Sumário

1	Introdução	3
2	Parte Teórica	3
2.1	Qual a diferença entre os tipos bit e std_logic?	3
2.2	Quais pacotes da library “ieee” são extensões definidas por desenvolvedores de ferramentas e devem ser evitadas por não serem pacotes padrão definidos pelo IEEE?	3
2.3	Como realizar operações aritméticas (soma, subtração,...) usando std_logic_vector?	3
2.4	Onde um signal deve ser declarado?	4
2.5	Onde uma variable deve ser declarada?	4
2.6	Qual a diferença entre um signal e uma variable?	4
2.7	Qual a diferença entre um signal e uma variável de uma linguagem de programação, como C, por exemplo?	5
2.8	Qual a diferença entre uma variable e uma variável de uma linguagem de programação, como C?	5
3	Parte Prática	5
4	Conclusão	9

1 Introdução

Neste relatório serão discutidos conceitos sobre Tipos de dados básicos em VHDL e conversão de tipos, Conceitos de sinais e variáveis, Comandos para atribuição de sinais e variáveis.

2 Parte Teórica

2.1 Qual a diferença entre os tipos bit e std_logic?

bit	Assume valores '0' ou '1'. <code>x: in bit;</code>
bit_vector	Vetor de bits. <code>x: in bit_vector(7 downto 0);</code> <code>x: in bit_vector(0 to 7);</code>
std_logic	<code>x: in std_logic;</code>
std_logic_vector	<code>x: in std_logic_vector(7 downto 0);</code> <code>x: in std_logic_vector(0 to 7);</code>
boolean	Assume valores TRUE ou FALSE
real	

Figura 1: Tabela com os tipos de dados

2.2 Quais pacotes da library “ieee” são extensões definidas por desenvolvedores de ferramentas e devem ser evitadas por não serem pacotes padrão definidos pelo IEEE?

2.3 Como realizar operações aritméticas (soma, subtração,...) usando std_logic_vector?

A maneira padrão de se fazer operações aritméticas com bit_vector ou std_logic_vector é com o uso dos pacotes padrão para síntese (Padrão IEEE 1076.3).

Embora haja um pacote numeric_bit, o pacote numeric_std é mais usado. Ambos pacotes definem os tipos unsigned e signed como vetores do tipo base (bit ou std_logic) Assim é possível se definir um sinal ou uma variável e se poder fazer atribuições e comparações.

2.4 Onde um signal deve ser declarado?

SIGNAL(Sinal): representa sinais lógicos sobre um fio no circuito, os quais interligam componentes. Um sinal não tem memória, portanto se a fonte do sinal é removida, o sinal não terá um valor. PORTS são exemplos de sinais. Podem ser declarados na entidade, na arquitetura ou package. Não podem ser declarados em processos, mas podem ser utilizados em seu interior. Sinais internos declarados na arquitetura devem ser declarados no corpo da arquitetura antes entre as declarações “ARCHITECTURE” e “BEGIN”.

2.5 Onde uma variable deve ser declarada?

VARIABLE (Variável): Um objeto VARIABLE lembra seu conteúdo e é usado para cálculos de modelos comportamental. São utilizadas em processos e devem ser declaradas neles. São atualizadas imediatamente e não correspondem à implementação física, como no caso dos sinais.

2.6 Qual a diferença entre um signal e uma variable?

Uma diferença fundamental entre variáveis e sinais é o atraso da atribuição: as variáveis mudam sem atraso e os sinais mudam com um pequeno atraso. Sinais são temporizados e variáveis sem temporização. Variáveis só podem ser usadas dentro de processos, sinais podem ser usados dentro ou fora de processos. Qualquer variável que é criada em um processo não pode ser usada em outro processo, os sinais podem ser usados em vários processos, embora só possam ser atribuídos em um único processo. Sinal é um objeto com um histórico de valores passado, enquanto uma variável é um objeto com um único valor atual.

2.7 Qual a diferença entre um signal e uma variável de uma linguagem de programação, como C, por exemplo?

2.8 Qual a diferença entre uma variable e uma variável de uma linguagem de programação, como C?

3 Parte Prática

Criamos um novo projeto no Quartus II e adicionamos o arquivo *alu.vhd*. Com a compilação finalizada, temos a visualização RTL e o TMV abaixo:

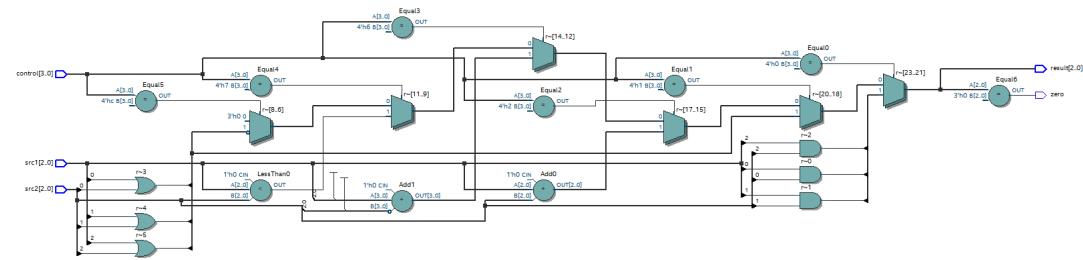


Figura 2: Visualização RTL da ALU

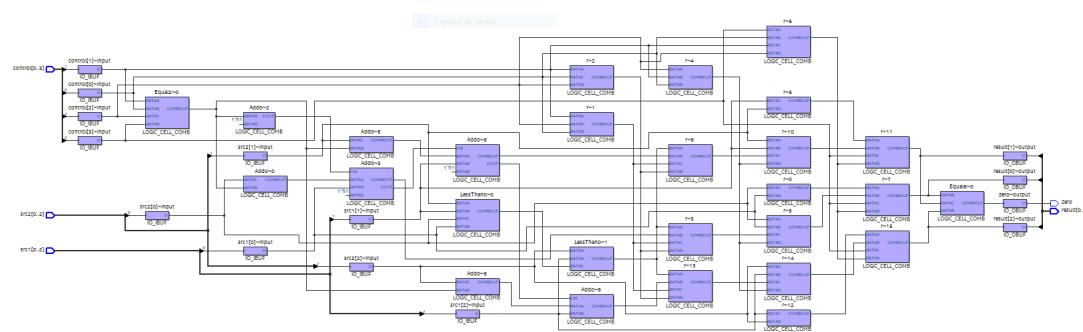


Figura 3: Visualização do TMV da ALU

Com tudo verificado, fizemos a configuração do *testbench* e executamos a simulação pelo ModelSim. Com a simulação também funcional, podemos fazer o planejamento de quais pinos serão utilizados na placa real. Definimos que os pinos utilizados serão:

- Entradas SRC1:
 - Chave SW0 - PIN_C10
 - Chave SW1 - PIN_C11
 - Chave SW2 - PIN_D12
- Entradas SRC2:
 - Chave SW3 - PIN_C12
 - Chave SW4 - PIN_A12
 - Chave SW5 - PIN_B12
- Seleção ALU:
 - Chave SW6 - PIN_A13
 - Chave SW7 - PIN_A14
 - Chave SW8 - PIN_B13
 - Chave SW9 - PIN_F15
- Saída Result:
 - LED0 - PIN_A8
 - LED1 - PIN_A9
 - LED2 - PIN_A10
- Saída Z:
 - LED9 - PIN_B11

Com isso, realizamos a gravação da placa para os testes.

Com a placa gravada, realizamos o seguinte teste:

- Teste 1
 - Inserimos o valor '000' na entrada SCR1
 - Inserimos o valor '000' na entrada SCR2
 - Inserimos o valor '0000' na seleção da ALU
 - Esperamos que o valor na saída Result seja '000'
 - Esperamos que o valor na saída Z seja '1'
- Teste 2
 - Inserimos o valor '110' na entrada SCR1

- Inserimos o valor '100' na entrada SCR2
- Inserimos o valor '0000' na seleção da ALU
- Esperamos que o valor na saída Result seja '100'
- Esperamos que o valor na saída Z seja '0'
- Teste 3
 - Inserimos o valor '000' na entrada SCR1
 - Inserimos o valor '001' na entrada SCR2
 - Inserimos o valor '0001' na seleção da ALU
 - Esperamos que o valor na saída Result seja '001'
 - Esperamos que o valor na saída Z seja '0'
- Teste 4
 - Inserimos o valor '110' na entrada SCR1
 - Inserimos o valor '100' na entrada SCR2
 - Inserimos o valor '0111' na seleção da ALU
 - Esperamos que o valor na saída Result seja '000'
 - Esperamos que o valor na saída Z seja '1'

Os resultados obtidos foram esses:



Figura 4: Resultado do Teste 1



Figura 5: Resultado do Teste 2

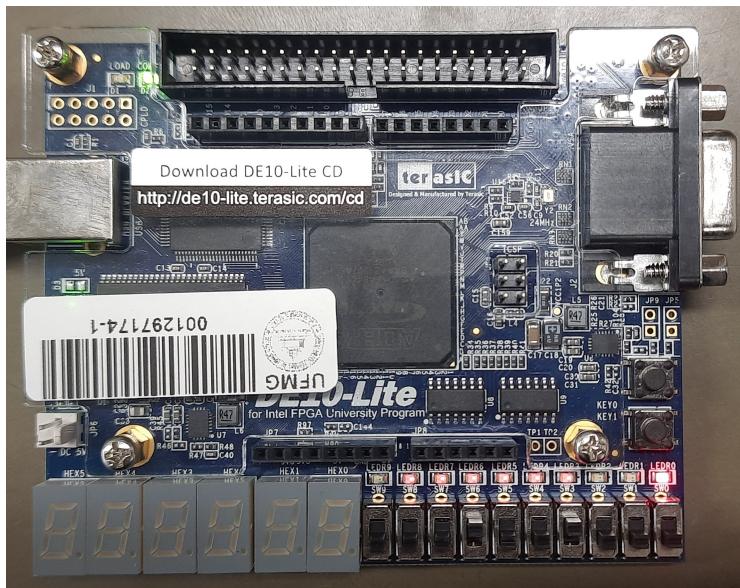


Figura 6: Resultado do Teste 3

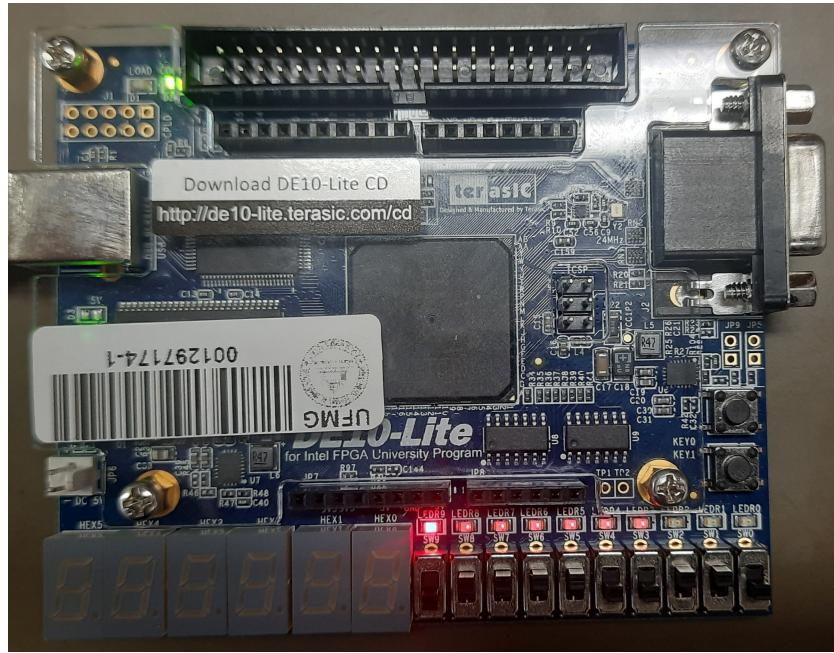


Figura 7: Resultado do Teste 4

4 Conclusão

Podemos observar pelas figuras acima que os resultados obtidos foram iguais aos resultados esperados em todos os teste.