PCS3115 - Sistemas Digitais I - Simulado 1 - ULA em VHDL

por Bruno de Carvalho Albertini

05/06/2020

Neste trabalho você irá familiarizar-se com o juiz de HDL usado nas disciplinas de hardware do PCS, descrevendo uma ULA (Unidade Lógica e Aritmética). Este trabalho é um simulado, apesar de serem atribuídas notas elas não contam na disciplina. Leia todo o arquivo até o final antes de enviar sua solução.

Introdução

Um processador pode ser visto como um projeto que utiliza o paradigma de unidade de controle e fluxo de dados. A unidade de controle é responsável pelo ciclo de busca, decodificação, execução e gravação dos dados, ciclo este que rege o funcionamento de um computador de uso geral pela sua característica programável. Já o fluxo de dados é composto por elementos de memória (e.g. banco de registradores), componentes de controle de fluxo (quase sempre combinatórios) e **unidades funcionais**.

Não se preocupe se não entender alguns conceitos.

O cálculo computacional é realizado nas unidades funcionais, portanto não é exagero afirmar que são os componentes principais de um processador. É possível construir máquinas com um ou até mesmo sem registradores, mas uma máquina sem uma unidade funcional simplesmente não realiza computação alguma.

Como as unidades funcionais são muito comuns, é costume juntar as principais funções computacionais em uma única unidade, que chamamos de ULA (Unidade Lógica e Aritmética). Como o próprio nome diz, uma ULA reune em um único componente operações lógicas (e.g. AND, OR, XOR, etc.) e aritméticas (adição, subtração, divisão, multiplicação, etc.), além de operações de comparação (menor, maior, igual, etc).

ALU, *Arithmetic and Logic Unit*.

Atividades

As operações lógicas de uma ULA são feitas diretamente por portas lógicas. Para fazer um AND entre dois números binários, por exemplo, basta fazer um AND bit a bit. No entanto, as operações aritméticas são um pouco mais complexas e realizadas por unidades internas que realizam a operaçao desejada. A ULA que iremos construir possui 4 bits nas entradas de dados (representando um número binário em complemento de base) e suporta as operações da Tabela 1.

Sabemos então que a ULA possui efetivamente três unidades funcionais, a saber AND, OR e um somador, além da capacidade de inverter a entrada `B` antes de realizar qualquer operação. As operações `AND`, `OR` e adição são realizadas diretamente pelas unidades corres-

OP	Unidade	Descrição
000	AND	$A\&B$, bit a bit
001	OR	$A\|B$, bit a bit
010	adição	$A+B$
110	subtração	$A-B$

Tabela 1: Operações que podem ser realizadas pela ULA. O campo OP pode ser interpretado como ($Op_2Op_1Op_0$), onde: Op_2 inverte B e Op_1Op_0 escolhe a função entre: 00:AND, 01:OR, ou 10:+.

pondentes. A subtração pode ser obtida invertendo-se B e entrando-se com 1 no *carry-in*, realizando a operação de soma com o complemento de base do número, o que equivale a subtração.

Todos os números são interpretados como complemento de 2.

```
entity alu is
  port (
    A, B : in  bit_vector(3 downto 0);
    F    : out bit_vector(3 downto 0);
    S    : in  bit_vector(2 downto 0);
    Z    : out bit; -- zero flag
    Ov   : out bit; -- overflow flag
    Co   : out bit  -- carry out
    );
end entity alu;
```

Listagem 1: Entidade para a ULA

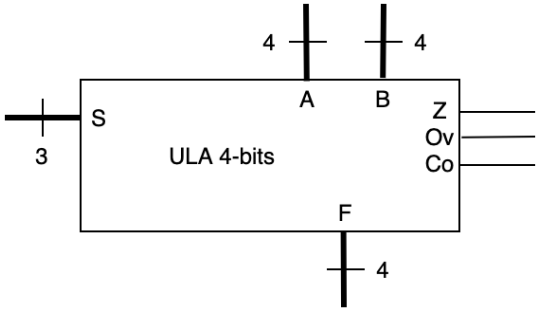

Figura 1: Diagrama da ULA.

As entradas de dados são A e B, a saída dados é F, e a entrada de controle S seleciona qual operação a ULA realizará. Os sinais de estado são as saídas que indicam que o resultado é zero ou que o resultado é *overflow*. Além dos sinais de estados, ainda há uma saída de *carry*. A Figura 1 mostra o símbolo para a ULA e a Listagem 1 a entidade correspondente em VHDL.

Se colocar no seletor de operações um valor não especificado na Tabela 1, o resultado é considerado inválido e pode ser qualquer coisa.

Note que não há entrada de *carry*.

S1A1 Implemente uma ULA seguindo **estritamente** a entidade fornecida, que suporte as operações de AND e OR. Esta atividade permite até 5 submissões e a nota será a maior nota dentre todas as suas submissões.

Simulado 1, Atividade 1, 5 envios, maior nota

Resposta: A resposta correta para este problema está no arquivo alu_andor.vhd. Analise o arquivo com cuidado e envie para o juiz. Para treinar, altere a entidade (e.g. mude o nome da entidade ou de uma das portas de entrada ou saída) e envie novamente. Lembre-se que você só tem 5 chances para acertar o problema, então certifique-se que ao menos uma das suas submissões esteja correta. No pacote que recebeu, há um arquivo chamado alu_andor_tb.vhd que pode ser usado para testar sua ULA localmente usando um simulador (e.g. GHDL). Se achar necessário inclua mais casos de teste. Quando esgotar a quantidade de envios permitida, tente enviar novamente.

O juiz não mostra quantas submissões você ainda tem na interface de envio, então você deve controlar isso.

S1A2 Altere sua ULA para suportar, além das operações já suportadas, a soma e a subtração. Esta atividade permite até 10 submissões e a nota é a da última submissão.

Simulado 1, Atividade 2, 10 envios, última nota

Resposta: A resposta para este problema está no arquivo (procure no pacote que recebeu) `alu_somador.vhd`, no entanto ela não está correta e você precisa corrigi-la antes de enviar. Use o *testbench* fornecido no arquivo `alu_somador_tb.vhd` para testar sua ULA antes de enviar. Comece enviando o arquivo fornecido e veja a resposta. Depois, encontre e conserte o erro na descrição e envie novamente. Lembre-se que a nota é a da última submissão, então envie por último uma solução correta.

O *testbench* está correto e testa todos os casos possíveis.

S1A3 (Desafio Baby Shark) Altere sua ULA para suportar, além das operações já suportadas, a saída do *flag* zero, que deve ser alto quando o resultado na saída é zero. Esta atividade permite até 10 submissões e a nota cai de acordo com o número de envios (i.e. seu primeiro envio corresponde à nota atribuída pelo juiz, mas se enviar novamente, sua nota será diminuída proporcionalmente ao número de envios restantes).

Simulado 1, Atividade 3, 10 envios, decaimento linear

Resposta: Este é um desafio, então não foi fornecida uma solução para esta atividade. Você precisa partir da solução anterior e adicionar lógica suficiente para resolver o problema. Para este problema, foi fornecido um *testbench* (correto e completo) que testa se sua ULA está correta. Note que a cada envio, a nota do juiz é multiplicada proporcionalmente pela quantidade de envios restantes, então se você receber 10 do juiz na quinta submissão, ficará com nota 5 no e-Disciplinas.

Não se preocupe pois a nota não conta na média da disciplina!

S1A4 (Desafio Chuck Norris) Altere sua ULA para suportar, além das operações já suportadas, a saída dos *flags* restantes, *carry-out* e *overflow*, que devem ser altos quando o resultado produz um *carry-out* ou um *overflow*, respectivamente. Caso aconteça um *overfow*, o resultado na saída deve condizer com a soma realizada pelo somador, mesmo errada. Esta atividade permite até 10 submissões e a nota é a mais alta.

Simulado 1, Atividade 4, 10 envios, maior nota

Resposta: Este desafio é para quem está se aprofundando na disciplina e não foi fornecida nenhuma resposta ou *testbench*! Você terá que fazer sua própria solução e sugere-se que faça também seu próprio *testbench* para testar sua solução antes de enviar para o juiz.

Instruções para Entrega

Para cada atividade deste simulado, há um *link* específico no e-Disciplinas. Acesse-o somente quando estiver confortável para enviar sua solução. Em cada atividade, você pode enviar apenas um único arquivo codificado em UTF-8. O nome do arquivo não importa, mas sim a descrição VHDL que está dentro. A entidade da implementação está na Listagem 1 e deve ser idêntica na sua solução ou o juiz não irá processar seu arquivo.

Qualquer editor de código moderno suporta UTF-8 (e.g. Atom, Sublime, Notepad++, etc).

Quando acessar o *link* no e-Disciplinas, o navegador abrirá uma janela para envio do arquivo. Selecione-o e envie para o juiz. Jamais recarregue a página de submissão pois seu navegador pode enviar o arquivo novamente, o que vai ser considerado pelo juiz como um novo envio e pode prejudicar sua nota final. Caso desista do envio, simplesmente feche a janela.

Depois do envio, a página carregará automaticamente o resultado do juiz, quando você poderá fechar a janela. A nota dada pelo juiz é somente para a submissão que acabou de fazer. Sua nota na atividade poderá ser vista no e-Disciplinas e pode diferir da nota dada pelo juiz dependendo da estratégia de atribuição de notas utilizada pelo professor que montou o problema (veja a atividade 3).

Pode demorar alguns segundos até o juiz processar seu arquivo.

Quando estiver elaborando sua solução, lembre-se que a ULA é um componente combinatório, então você não deve usar `process`. Para todos estes problemas, também não está permitido usar a biblioteca `std_logic_1164`, a `textio` e qualquer outra biblioteca não padronizada. Também certifique-se que não imprime nada na saída da simulação.

Atenção: não atualize a página de envio e não envie a partir de conexões instáveis (e.g. móveis) para evitar que seu arquivo chegue corrompido no juiz.