Universidade de Brasília

INSTITUTO DE CIÊNCIAS EXATAS – IE
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO – CIC

Disciplina: CIC 116394 – Organização e Arquitetura de Computadores – Turma C
Semestre: 2019/2
Professor: Marcelo Grandi Mandelli

TRABALHO 2 – ULA e GERADOR DE IMEDIATOS RISC-V

OBJETIVOS

- Compreender o processo de desenvolvimento e simulação de projetos de hardware utilizando linguagens de descrição de hardware
- Compreender sobre do projeto de um processador RISC-V
- Compreender o funcionamento e o projeto de uma Unidade Lógica e Aritmética (ULA)
- Compreender sobre formato de instruções do RISC-V, além do projeto de um gerador de imediatos

ESPECIFICAÇÃO DO TRABALHO

Este trabalho consiste no desenvolvimento e simulação de projetos de hardware utilizando uma linguagem de descrição de hardware. Para isso, as ferramentas Quartus II e/ou Modelsim serão utilizadas. **Este trabalho consiste no desenvolvimento e simulação de dois módulos utilizando uma linguagem de descrição de hardware: a Unidade Lógica e Aritmética (ULA) do RISC-V; e o gerador de imediatos do RISC-V.**

Unidade Lógica e Aritmética (ULA)

Nesse trabalho você deve desenvolver e simular em linguagem de descrição de hardware uma Unidade Lógica e Aritmética (ULA) com as seguintes características:

- O tamanho das palavras de dados da ULA (operandos e resultado) será parametrizável, definido por uma constante DATA_WIDTH. Na linguagem VHDL, essa constante pode ser definida na entidade da ULA utilizando a primitiva ***generic***.
- Por padrão, a constante DATA_WIDTH terá tamanho 32, indicando que a ULA fará operações de dados de 32 bits.
- A ULA terá duas entradas de dados **A** e **B**, as quais terão tamanho de palavra DATA_WIDTH.
- As entradas **A** e **B** serão os operandos da operação a ser realizada pela ULA.
- A operação a ser realizada pela ULA será definida pela entrada **op** de tamanho 4 bits.
- A ULA terá uma saída **result**, a qual terá tamanho de palavra DATA_WIDTH.
- A saída **result** da ULA apresentará o resultado da operação definida em **op** entre os operandos definidos por **A** e **B.**
- **As operações com sinal serão realizadas em complemento de 2.**

- As operações que serão realizadas pela ULA são apresentadas na tabela a seguir:

Operação	Significado	**op**
ADD	result recebe A + B	0000
SUB	result recebe A - B	0001
SLL	result recebe A deslocado em B bits à esquerda	0010
SLT	result = 1 se A < B, com sinal	0011
SLTU	result = 1 se A < B, sem sinal	0100
XOR	result recebe a operação lógica A xor B, bit a bit	0101
SRL	result recebe A deslocado em B bits à direita sem manter o sinal	0110
SRA	result recebe A deslocado em B bits à direita mantendo o sinal	0111
OR	result recebe a operação lógica A or B, bit a bit	1000
AND	result recebe a operação lógica A and B, bit a bit	1001
SEQ	result = 1 se A == B	1010
SNE	result = 1 se A != B	1011
SGE	result = 1 se A ≥ B, com sinal	1100
SGEU	result = 1 se A ≥ B, sem sinal	1101

O seguinte código VHDL, que implementa uma ULA simples, pode ser tomado como base para a implementação desse trabalho.

```
library ieee;
```

```
                    std_logic_vector(signed(A) - signed(B)); --operação default
end alu;
```

Dicas para implementação da ULA:

- Algumas operações necessitam de palavras com ou sem sinal. Para isso converta de std_logic_vector para **signed** ou **unsigned.** Exemplo: signed(A)
- Algumas operações (ex.: +, -) não funcionarão com std_logic_vector. Para isso, será necessário converter para **signed** ou **unsigned**, realizar a operação, e depois reconverter pra std_logic_vector. Exemplo:

  output <= std_logic_vector(signed(A) <operação> signed(B));

- Para as operações sll, srl e sra utilize as funções shift_left ou shift_right. **Será necessário ter cuidado com conversões de tipo!**

GERADOR DE IMEDIATOS RISC-V

O gerador de imediatos a ser implementado deve ser capaz de gerar o imediato de todos os formatos de instruções presentes na ISA RV32I: tipo-R, tipo-I, tipo-S, tipo-B, tipo-U e tipo-J. A figura abaixo apresenta os formatos de instruções e suas divisões por campos, onde o imediato de cada formato está marcado em verde:

Os principais campos presentes nos formatos são:

- **funct7**: campo de 7 bits auxiliar na identificação da instrução
- **rs2**: registrador fonte da operação
- **rs1**: registrador fonte da operação
- **rd**: registrados destino da operação
- **funct3**: campo de 3 bits auxiliar na identificação da instrução
- **opcode**: campo de 7 bits que identifica a instrução
- **imediato (i):** valor numérico utilizada para a instrução

A formação de imediato, para cada formato, é apresentada a seguir:

Formato	
tipo-R	Não há imediato
tipo-I	imediato = { 20{imm[11]}, imm[11:0] }
tipo-S	imediato = { 20{imm[11]}, imm[11:5], imm[4:0] }
tipo-B	imediato = { 20{imm[12]}, imm[11:1], 0 }}
tipo-U	imediato = { imm[31:12], 000000000000 }
tipo-J	imediato = {12{imm[20]}, imm[19:1], 0 }

Exceto no tipo-U, como vemos nas fórmulas acima, todos os imediatos têm extensão de sinal. O tipo-U não possui extensão de sinal, visto que o imediato do formato já corresponde aos bits mais significativos dos 32 bits.
Os imediatos dos tipos B e J sempre são múltiplos de 2, por isso, como podemos ver nas fórmulas acima, o último bit de seus imediatos é sempre 0. **No gerador de imediato deste trabalho, os imediatos dos tipos B e J devem já ser gerados sendo múltiplos de 2.**

Os formatos são definidos pelos seguintes opcodes:

Formato	**opcode**
tipo-R	0x33
tipo-I	0x03 ou 0x13 ou 0x67
tipo-S	0x23
tipo-B	0x63
tipo-U	0x17 ou 0x37
tipo-J	0x6F

O gerador de imediatos deve possuir como entrada uma instrução de tamanho 32 bits em algum dos formatos apresentados acima. Como saída, o gerador de imediato deve gerar o valor do dado imediato, de tamanho 32 bits, de acordo com o formato de instrução da entrada. No caso do formato tipo-R, que não possui imediato, deve ser gerado o valor zero na saída de 32 bits.

Dessa forma, as entradas e saídas do gerador de imediato devem ser similares a ***entity*** abaixo em VHDL:

```
library ieee;

                inst   : in std_logic_vector(32 downto 0);

end entity genImm32;
```

Para verificar o funcionamento do módulo gerador de imediatos, pode ser utilizado as seguintes instruções do RISC-V, entre outras:

Instrução RISC-V	Código	Formato	Imediato
add t0, zero, zero	0x000002b3	tipo-R	inexiste: 0
lw t0, 16(zero)	0x01002283	tipo-I	16
addi t1, zero, -100	0xf9c00313	tipo-I	-100
xori t0, t0, -1	0xfff2c293	tipo-I	-1
addi t1, zero, 354	0x16200313	tipo-I	354
jalr zero, zero, 0x18	0x01800067	tipo-I	0x18 / 24
lui s0, 2	0x00002437	tipo-U	0x2000
sw t0, 60(s0)	0x02542e23	tipo-S	60
bne t0, t0, main	0xfe5290e3	tipo-B	-32
jal rot	0x00c000ef	tipo-J	0xC / 12

<u>Dicas para implementação do gerador de imediatos:</u>

- O operador de concatenação em VHDL pode ser muito útil
- Exemplo:

```
architecture a of genImm32 is
signal
begin
                                                16) & '0';
...
end gemImm32;
```

Simulação e Verificação: simular o funcionamento da ULA e gerador de imediatos de forma a verificar o funcionamento. Utilizar o Quartus/ ModelSim para a simulação, desenvolvendo um *testbench* para cada um dos módulos a serem testados (ULA e Gerador de Imediatos) para acionamento das entradas. Os testbenchs devem incluir a execução de ao menos um teste para cada operação da ULA e para cada formato de instrução no gerador de imediatos.

GRUPOS

O trabalho deverá ser realizado em grupos de 2 ou 3 alunos.

ENTREGA

Deverá ser entregue:

- **código em linguagem de descrição de hardware comentado** da ULA e do gerador de imediatos
- **testbench comentado** da ULA e do gerador de imediatos
- **relatório mostrando os resultados da simulação (imagens)** da ULA e do gerador de imediatos. Nesse relatório você colocará as imagens das formas de onda obtidas em simulação, descrevendo os testes que estão sendo realizados em cada imagem.

Entregar um arquivo compactado em formato .zip no moodle (aprender.unb.br) até às 23h55 do dia 10/11/2019

O nome do arquivo deve conter as matrículas dos integrantes do grupo:
matriculaaluno1_matriculaaluno2_matriculaaluno3.zip

ENTREGA EM ATRASO: o número de pontos descontados por dia em atraso na entrega do trabalho é dado pela equação: $2^n+(n-1)$, sendo n o número de dias em atraso. (1 dia → -2 pontos, 2 dias → -5 pontos, 3 dias → -10 pontos)

CRITÉRIOS DE AVALIAÇÃO

Funcionamento da Unidade Lógica e Aritmética – 40% da nota

Funcionamento do Gerador de Imediatos – 40% da nota

Comentários nos códigos e relatório de simulação – 20% da nota

Esta especificação pode ser atualizada para se efetuar correções de texto ou alterações para se deixar mais claro o que está sendo pedido.

Caso a especificação sofrer uma atualização, os alunos serão informados via moodle (aprender.unb.br).

Última atualização: 24/10/2019 às 17:00