Implementação de uma Unidade Lógica Aritmética com Portas Lógicas Básicas
Elementos de Lógica Digital 2018/02
Prof. João Paulo Andrade Almeida

3 de setembro de 2018

O objetivo deste trabalho é praticar os conceitos básicos do projeto de circuitos lógicos combinacionais com uma ferramenta computacional (Logisim evolution, download em: http://reds-data.heig-vd.ch/logisim-evolution/logisim-evolution.jar com instruções de execução em: https://github.com/reds-heig/logisim-evolution). O trabalho deverá ser realizado em duplas. Plágio (cópia entre diferentes grupos) não será tolerado, e grupos com trabalhos copiados (mesmo que parcialmente) receberão nota mínima no trabalho como um todo.

Além da corretude, a organização dos seus circuitos é um critério importante de avaliação. Use a modularidade de circuitos que o Logisim permite para estruturar os seus circuitos, separando aqueles que executam diferentes sub-funções e lançando mão do reuso de circuitos modulares (reuso de circuitos como caixas pretas).

Além do circuito, você deve incluir um relatório em que explique o seu projeto, as decisões tomadas, e os testes executados (capture as telas dos testes no logisim). O relatório deve conter os esquemas de todos os circuitos utilizados, apresentando-os progressivamente de forma modularizada.

Você deve usar apenas as portas lógicas básicas, sem recorrer a componentes prontos do Logisim (como somadores, multiplexadores). Você deve construir todos esses circuitos que precisar a partir de portas lógicas básicas.

O prazo para entrega é dia 5 de outubro de 2018.

1 Unidade Lógica Aritmética (Arithmetic Logic Unit – ALU)

Uma unidade lógica aritmética (ULA ou ALU) é um elemento central em uma CPU. Ela executa operações lógicas e aritméticas diversas em dados de entrada. Neste trabalho, o seu objetivo é projetar uma ALU que opere sobre dados de 8 bits.

2 Especificação

Sua ALU deve implementar as seguintes operações em cima das entradas A e B de 8 bits:

- ADD soma (A + B);
- SUB subtração (A – B);
- XOR bit a bit (A XOR B);
- OR bit a bit (A OR B);
- AND bit a bit (A AND B);
- SHL Shift left (A << 1, desloca todos os bits de A para a esquerda, multiplicando o número por 2)
- SHR Shift right (A >> 1, desloca todos os bits de A para a direita, dividindo o número por 2)

Além dos operandos A e B de 8 bits, sua ALU deve ter uma entrada OP de 3 bits que indica a operação a ser realizada com os operandos. A ALU deve seguir a seguinte tabela para a operação OP:

Valor de OP	Saída
000	ADD soma (A + B)
001	SUB subtração (A – B)
010	XOR bit a bit (A XOR B)
011	OR bit a bit (A OR B)
100	AND bit a bit (A AND B)
101	SHL Shift left (A << 1)
110	SHR Shift right (A >> 1)
111	não usada

Sua ALU também deve ter um *carry* de entrada (Cin), cujo funcionamento permitirá depois concatenar duas ALUs de 8 bits para formar uma ALU de 16 bits).

Além de uma saída *s* de 8 bits com o resultando da operação, sua ALU deve ter:

- uma saída *c* indicando *carry* (1 em *c* indica que houve *carry*) (no caso de SUB indica *borrow*, no caso da operação SHL, o carry recebe o bit mais significativo do byte sendo deslocado; no caso de SHR, recebe o bit menos significativo),
- uma saída *z* que indica se todos os bits da saída *s* são iguais a 0 (1 em *z* indica que todos os bits da saída *s* são 0).

Um arquivo `.circ` básico do Logisim será fornecido para que você o use como base para a definição da sua ALU.

3 Calculadora

Você deve usar esta ALU para construir uma calculadora cuja saída deve ser representada em um display de 7 segmentos com 2 dígitos hexadecimais. (Você deverá construir também o decodificador para o display de 7 segmentos, *não* podendo lançar mão do display de 7 segmentos existente no logisim chamado “Hex Digit Display”. Você deve usar o “7-Segment Display”.)

4 ALU de 16 bits

Use a ALU de 8 bits para construir uma ALU de 16 bits com as mesmas operações.

5 Condições de Entrega

Inclua o arquivo `.circ`, e o relatório (PDF) um arquivo `.zip` nomeado com `alu` seguido dos nomes dos componentes do grupo separados por hífen. (Exemplo: `alu-JoseSilva-MariaVieira.zip`). Envie o arquivo para jpalmeida@inf.ufes.br com o nome do arquivo no campo “subject”.