Trabalho 1 – SSC0108

Descrição

Uma Unidade Lógica e Aritmética (ULA) combina uma variedade de operações lógicas e matemáticas dentro de uma única unidade. Por exemplo, uma ULA típica pode realizar adição, subtração, comparação de magnitude e operações AND e OR.

Neste trabalho, você deverá implementar uma ULA usando dois registradores em sua entrada de dados, A e B, sendo que o registrador A tambem serve de acumulador. As operações serão definidas da seguinte maneira:

- Operações com 1 operando: A = op A
- Operações com 2 operandos: A = A op B

O projeto deverá conter as seguintes entradas e saídas:

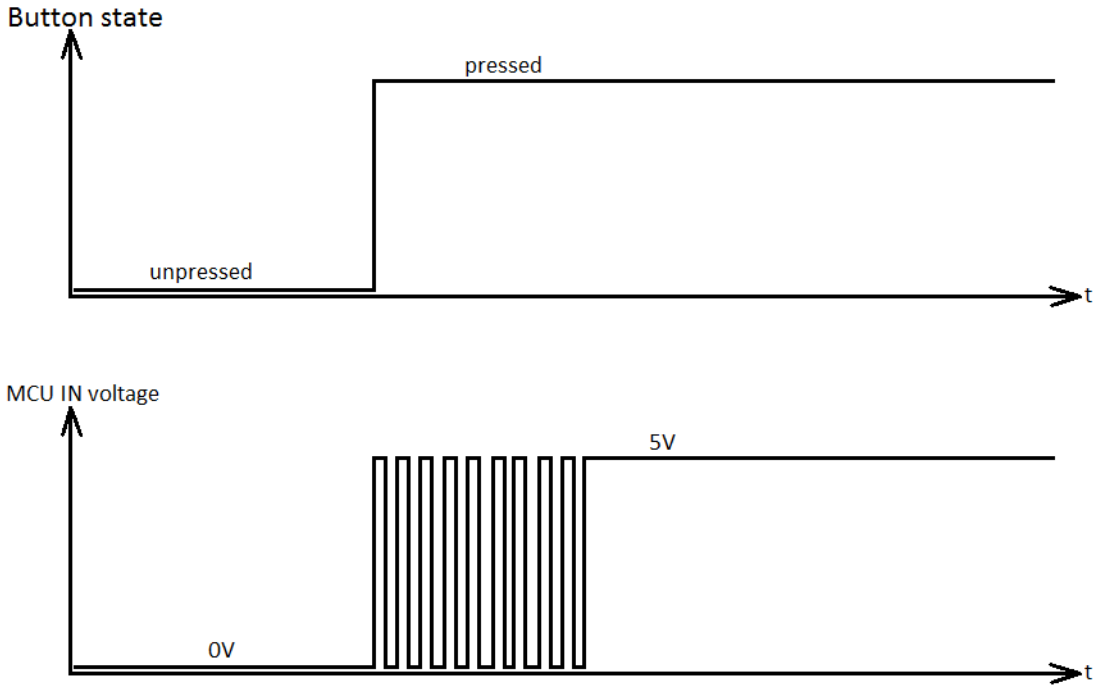
- KEY[0]: reseta os registradores para o valor zero
  - Utilize o pino CLEAR dos flip-flops
  - Convencionalmente sinais de reset são ativados em 0 (porém não obrigatório)
- SW[7..0]: operandos ou operação, 8 bits
- SW[8]: configura a operação do registrador A (ver KEY[0])
- KEY[1]: opera o registrador A de acordo com SW[8]:
  - Se SW[8] = 0 e pressionar KEY[1], o valor de SW[7..0] é carregado ao registrador A
  - Se SW[8] = 1 e pressionar KEY[1], é realizada a operação presente em SW[7..0] (ver tabela de operações)
- KEY[2]: carrega o valor de SW[7..0] ao registrador B
- HEX3[6..0] e HEX2[6..0]: valor presente no registrador A
- HEX1[6..0] e HEX0[6..0]: valor presente no registrador B

O diagrama no final deste documento apresenta a estrutura esperada para este projeto. Os módulos presentes são:

- Register8B: registrador de 8 bits
  - in[7..0]: entrada do registrador
  - out[7..0]: saída do registrador
  - clock: clock manual
- ULA: unidade lógica aritmética
  - A[7..0]: operando A
  - B[7..0]: operando B
  - OP[3..0]: operação a ser realizada
- mux2: selecionador de entrada para o registrador A. É este multiplexador que seleciona o que deve ser carregado no registrador A: o valor de SW[7..0] ou o resultado da ULA
- bin2sevenseg: conversor de valor binário de 8 bits para dois displays de sete segmentos
  - hex1: display de sete segmentos a mostrar os 4 bits mais significativos
  - hex0: display de sete segmentos a mostrar os 4 bits menos significativos

Debouncer

Os botões da placa, ao serem pressionados/ativados, geram um ruído antes de entrar em um estado estável. Isto é uma característica física dos botões. Segundo a imagem abaixo, ao apertar um botão, esperamos uma onda igual ao “Button state”. Porém, o que ocorre de verdade é uma onda igual ao “MCU IN voltage”.



Fonte: https://industrialcircuits.files.wordpress.com/2014/07/contact_bouncing.png

Para evitar tal problema, precisamos de um debouncer (um filtro para a remoção do ruído). O debouncer será fornecido na wiki como um módulo Verilog (arquivo com extensão .v) e a sua caixinha (arquivo com extensão .bsf). Adicione os dois arquivos no seu projeto Quartus e utilize-o para todos os pinos KEY[x]. Este debouncer é obrigatório, do contrário o seu circuito pode não funcionar como esperado!

Este debouncer funciona tanto para botões ativos em 0 quanto para botões ativos em 1.

O debouncer contém as seguintes entradas e saídas:

- clk: clock utilizado pelo filtro. Utilizar o clock de 50 MHz da placa (CLOCK_50):
  - PIN_Y2 na placa DE2-115
  - PIN_M9 na placa DE0-CV
- rst_n: reset do debouncer, conectar diretamente ao KEY[0]
- inb: entrada com ruído (botão KEY[x])
- outb: saída filtrada do botão

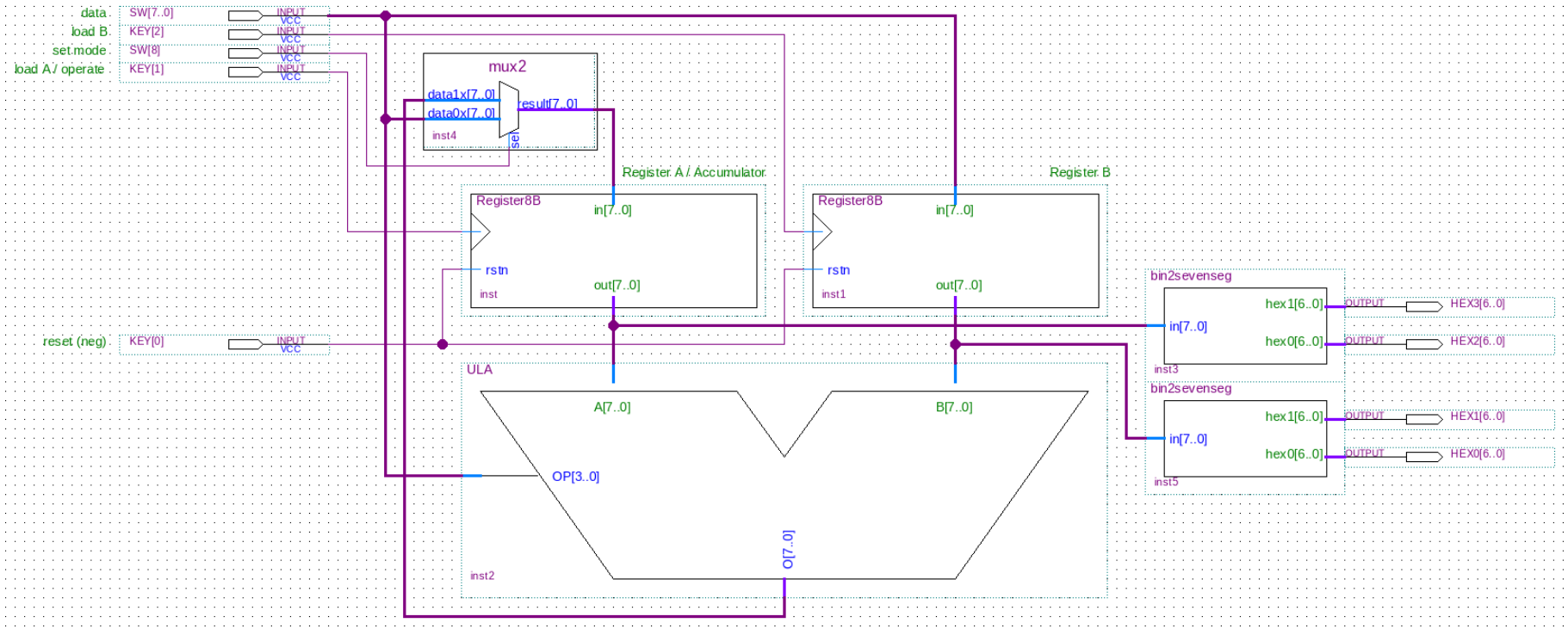
Tabela de instruções

SW[3..0]	Operação
0000	Negação (A = not A)
0001	E (A = A and B)
0010	OU (A = A or B)
0011	OU exclusivo (A = A xor B)
0100	Soma (A = A + B)*
0101	Subtração (A = A - B)*
0110	Multiplicação (A = A * B)*
0111	Divisão (A = A / B)*
1000	Shift lógico esquerda (A = A << B)
1001	Shift lógico direita (A = A >> B)
1010	Shift aritmético esquerda (A = A <<< B)*
1011	Shift aritmético direita (A = A >>> B)*
1100	Rotação para esquerda (A = A rotl B)
1101	Rotação para direita (A = A rotr B)
1110	----
1111	----

* A e B devem aceitar complemento de 2

Observações

- Os trabalhos serão avaliados no horário de aula:
  - Terça: 26/09
- Dupla de 2 alunos (também conhecido como dupla)
- Colar = 0
- O projeto será executado nas placas (utilizar os switches SW e botões KEY para entrada de dados e displays de sete segmentos HEX para o resultado)
- USE O DEBOUNCER
- UTILIZEM O DEBOUNCER
- Está liberado utilizar os seguintes módulos prontos do Quartus:
  - dff: flip-flop do tipo D com preset e clear
  - lpm_mux: multiplexador parametrizável


data
load B
set mode
load A / operate
reset (neg)
SW[7..0]
KEY[2]
SW[8]
KEY[1]
KEY[0]
INPUT
VCC
mux2
data1x[7..0]
data0x[7..0]
result[7..0]
sel
inst4
Register A / Accumulator
Register B
Register8B
in[7..0]
rstn
out[7..0]
inst
inst1
ULA
A[7..0]
B[7..0]
OP[3..0]
O[7..0]
inst2
bin2sevenseg
hex1[6..0]
hex0[6..0]
OUTPUT
HEX3[6..0]
HEX2[6..0]
HEX1[6..0]
HEX0[6..0]
inst3
inst5