

Laboratório de Sistemas Digitais

Ano Letivo 2022/23

Projeto Final – enunciado 2

Filtro de Média Móvel

1. Introdução

Um filtro de média móvel é um componente frequente em sistemas de processamento de sinal e ou imagem. O sinal de saída, como consequência do resultado da filtragem, é uma versão “suavizada” do sinal de entrada em que os efeitos de variações bruscas são de alguma forma minimizados tal como se pretende representar na figura 1. Neste trabalho, pretende projetar-se e implementar na FPGA e testar no kit Terasic DE2-115 um sistema de filtragem de média móvel de largura 4. Para efeitos de teste o sinal de entrada estará disponível numa ROM de 256x8 bits e o sinal de saída deverá ser escrito numa RAM com as mesmas dimensões.

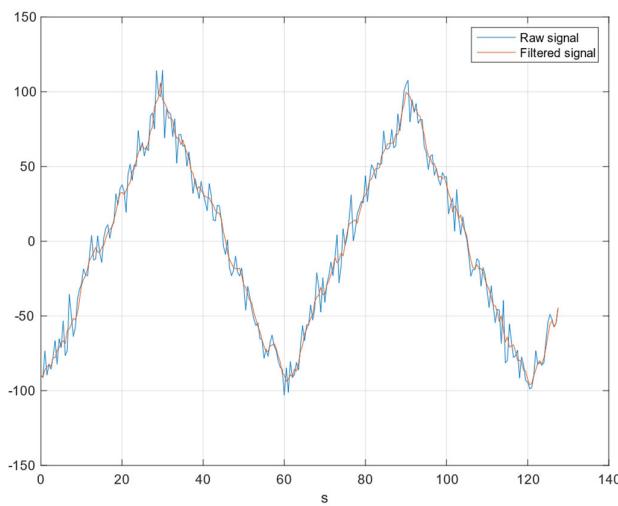


Figura 1: Filtro de média móvel de largura 4.

2. Descrição do funcionamento e requisitos

De uma forma geral, este trabalho consiste em modelar em VHDL, simular, sintetizar, implementar na FPGA e testar no kit Terasic DE2-115 um sistema digital de acordo com as principais especificações seguintes:

- O sistema deve ler um sinal de teste escrito numa ROM 256x8 bits. Este sinal é baseado numa onda triangular com ruído. O período da onda triangular é de 60s. A frequência de amostragem é de 2Hz e como tal a duração total do sinal gravado na ROM é de 128 s. Este componente será previamente fornecido.
- O conteúdo da ROM é do tipo *signed* com 8 bits.
- A RAM deverá ter as mesmas dimensões da ROM, com 1 porto para escrita síncrona e leitura assíncrona.

- O sistema deve mostrar sequencialmente em 4 displays Hexadecimais o conteúdo da ROM (sinal de entrada)
- O sistema deve mostrar sequencialmente em 4 displays Hexadecimais o conteúdo da RAM (sinal filtrado)
- Cada amostra do sinal filtrado y_k deverá ser calculada por uma unidade aritmética que implemente a seguinte fórmula que engloba 4 amostras do sinal de entrada x

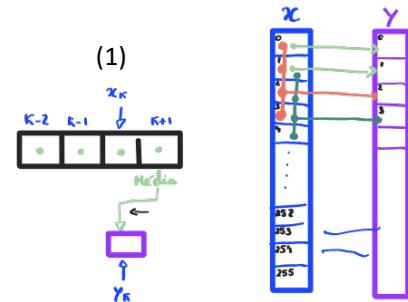
$$y_k = \frac{x_{k-2} + x_{k-1} + x_k + x_{k+1}}{4}, \quad k = 2, \dots, 254,$$

$$y_0 = x_0, y_1 = x_1, y_{255} = x_{255}$$

Values mantêm!

O sistema deverá ter os seguintes sinais de controlo:

- START: KEY0
- FILTER_ON: SW(0),
- RESET_RAM: KEY1, reset da RAM
- RESET: KEY2, reset geral do sistema
- HEX3...HEX0: Conteúdo da ROM (sinal da entrada)
- HEX7...HEX4: Conteúdo da RAM (sinal de saída)
- O *top-level* do circuito deverá ser implementado com recurso a representação estrutural em VHDL.



3. Sugestões para implementação

A implementação deste sistema deve seguir uma estratégia faseada, de acordo com a descrição que se sugere e de acordo com a proposta de arquitetura da figura 2:

Fase 1 (2 valores): Implementar um subsistema de geração de endereços que permita ler na forma *signed* o conteúdo da ROM. Utilize o ficheiro auxiliar `NoisyTriangSignal_ROM_256x8.vhd` donde poderá ler o sinal de entrada.

(8) -> 8 bits de endereço

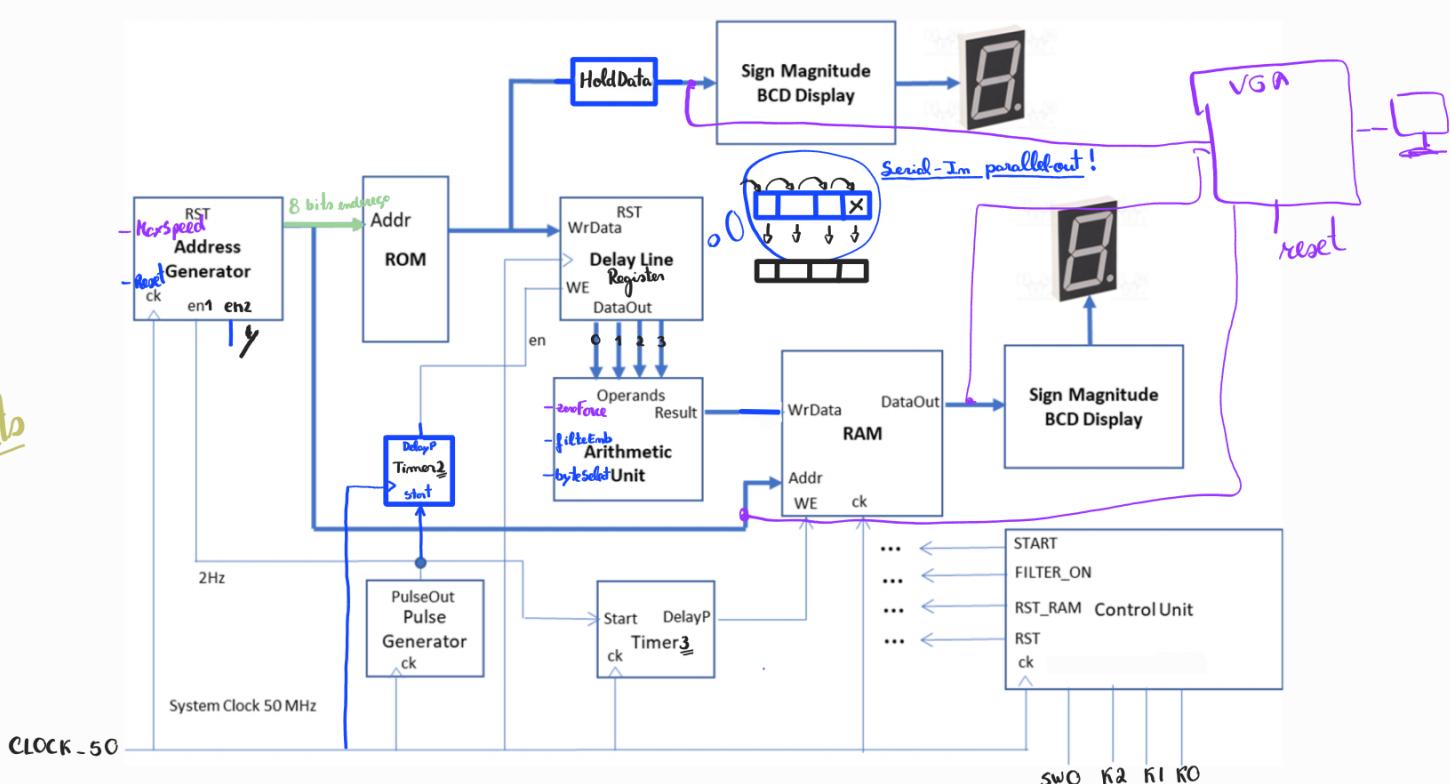
Fase 2 (2 valores): Implementar um subsistema de geração de endereços que permita ler na forma *signed* o conteúdo da RAM. Implemente o modo de funcionamento RESET_RAM que permita preencher a RAM com x"00". Visualize o resultado nos displays respetivos. Neste modo, utilize a frequência de 50 MHz para a geração dos endereços da RAM.

Fase 3 (4 valores): Implemente um subsistema intermédio de armazenamento que tenha como sáida as 4 amostras do sinal de entrada envolvidas no cálculo. Sugere-se que estes valores sejam sequencialmente lidos da Rom e escritos num banco de 4 registos que proporcionem um mecanismo de deslocamento ao nível das amostras de 8 bits. Na prática, trata-se duma linha de atraso em modo serial-in – parallel-out para palavras de 8 bits. → Total de bits : $8 \times 4 = 32$

Fase 4 (4 valores): Uma unidade aritmética deverá ler em paralelo o conteúdo dos 4 registos e implementar a fórmula (1). O resultado deverá ser escrito na RAM com endereçamento adequado.

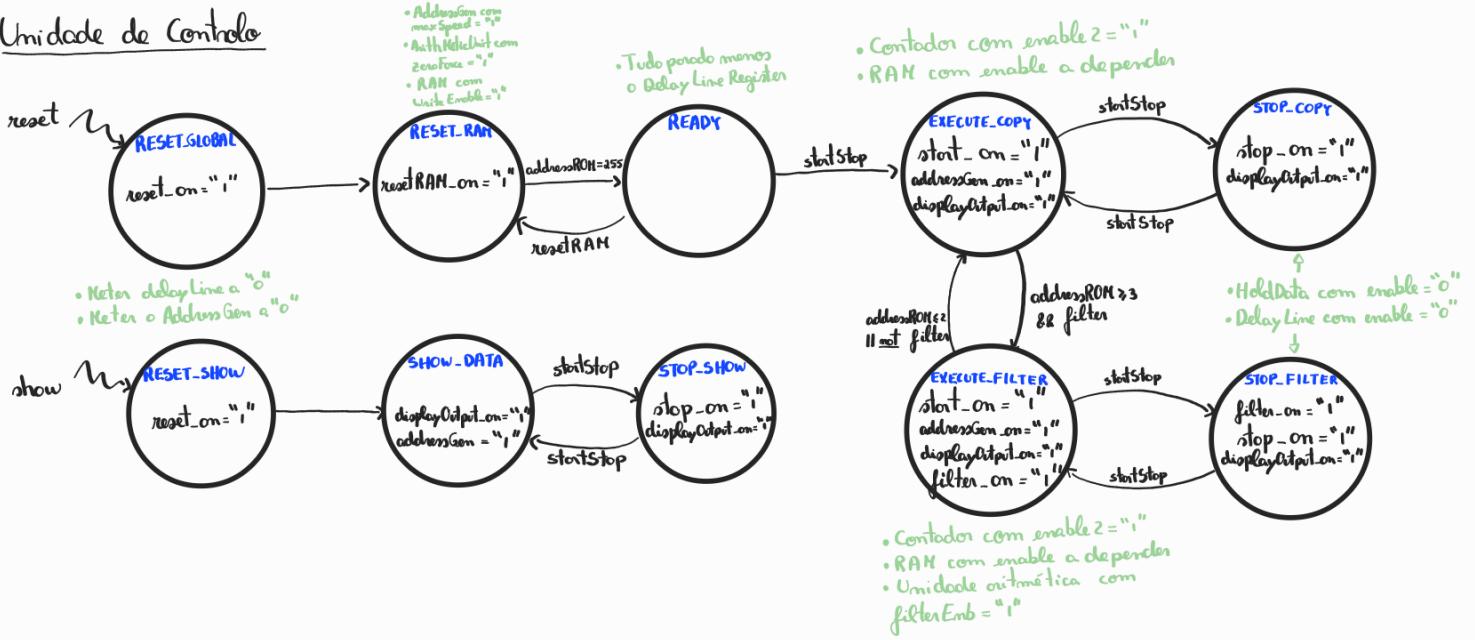
FASE 5 (4 valores): Implementar uma unidade de controlo (máquina de estados) que permita o funcionamento de acordo com a interface proposta. O sistema deve necessariamente começar por fazer RESET à RAM e só depois evoluir para o processo de filtragem.

Fase 6 (4 valores): Interligar os blocos anteriores (*top-level*) e visualizar a filtragem em tempo real. Para controlo de qualidade, as sequências geradas devem ser seguir as sequências que constam no ficheiro auxiliar: `LSD_MAF_Signals.xlsx`



Address ROM	Data Line	Operação	Saida	Address RAM	
0	$\rightarrow X_0 X_{255} X_{254} X_{253}$	1 byte	$Y_{255} = X_{255}$	255	$\Rightarrow \text{Sincronismo do Sistema}$
1	$\rightarrow X_1 X_0 X_{255} X_{254}$	1 byte	$Y_0 = X_0$	0	
2	$\rightarrow X_2 X_1 X_0 X_{255}$	1 byte	$Y_1 = X_1$	1	
3	$\rightarrow X_3 X_2 X_1 X_0$	CalcEnable	$Y_2 = \text{Med}(.)$	2	
.	⋮	⋮	⋮	⋮	
254	$\rightarrow X_{254} X_{253} X_{252} X_{251}$	CalcEnable	$Y_{253} = \text{Med}(.)$	253	
255	$\rightarrow X_{255} X_{254} X_{253} X_{252}$	CalcEnable	$Y_{254} = \text{Med}(.)$	254	
0	$\rightarrow X_0 X_{255} X_{254} X_{253}$	1 byte	$Y_{255} = X_{255}$	255	
1	$\rightarrow X_1 X_0 X_{255} X_{254}$	1 byte	$Y_0 = X_0$	0	

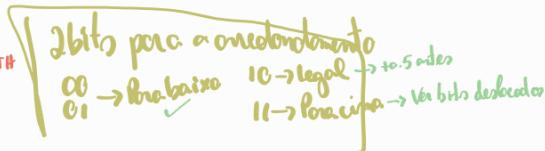
Unidade de Controle



Problemas:

- Arredondamento (VGA e ALU)

• Ajustar os pixels
pixels pixel x 156 ≈ width

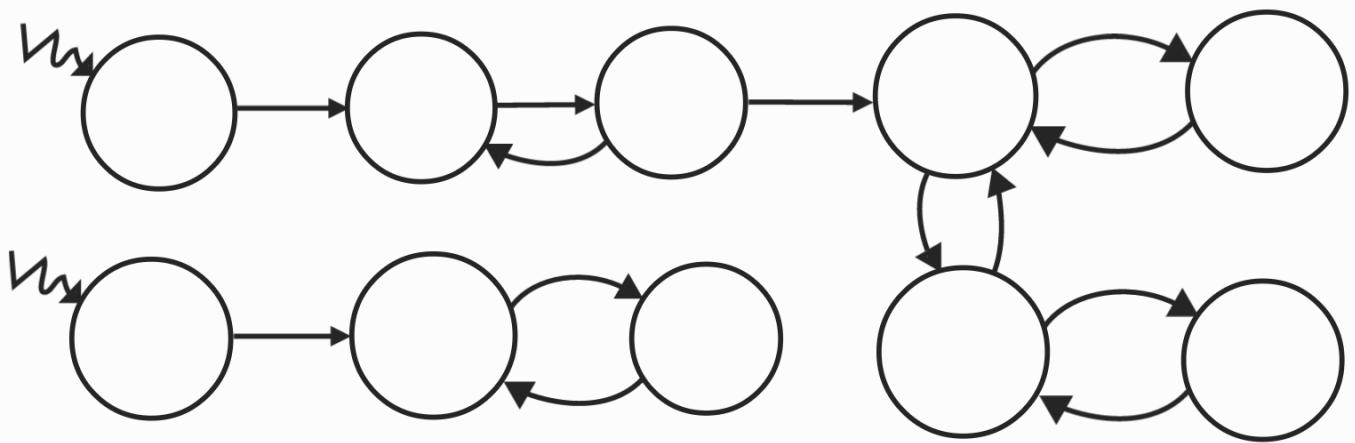
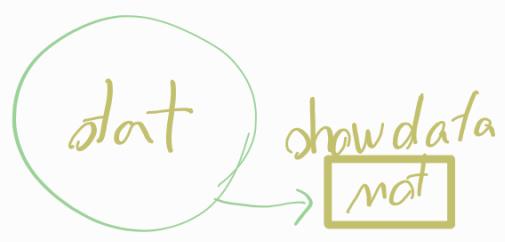


Filter ✓ delayLine_on = not s_stop_on
addressGen = start_on or showData_on

2 enables para HoldData e DelayLine

WriteForce → RAM = not RAM_on → Ultimo!

RAM_write_on DisplayOutput = dad_dsp



Média de 4 números signed de 3 bits: (exemplo) [-4, 3]

$$\begin{array}{l} 100 = -4 \\ 110 = -2 \\ 001 = 1 \\ 011 = 3 \end{array}$$

Média é $-0,5$

$$\begin{array}{l} 011 = 3 \\ 011 = 3 \\ 011 = 3 \\ 011 = 3 \end{array}$$

Média é 3

$$\begin{aligned} 000000 + (-4) &= 111100 = -4 \\ + (-2) &= 111010 = -6 \\ + 1 &= 111011 = -5 \\ + 3 &= \underbrace{111110}_{\text{Soma}} = -2 \end{aligned}$$

$\Rightarrow 111110 \lll 2$

$$= 111111 = -1 //$$

$$\begin{aligned} 0000000 + 3 &= 0000111 = 3 \\ + 3 &= 0001110 = 6 \\ + 3 &= 0010001 = 9 \\ + 3 &= \underbrace{001100}_{\text{Soma}} = 12 \end{aligned}$$

$\Rightarrow 001100 \lll 2$

$$= 0000\underbrace{11}_{\text{Soma}} = 3 //$$

Aredondamento:

0X: Aredondamento para baixo

\hookrightarrow Soma $>> 2$

10: Aredondamento normal: \rightarrow Média não negativa!

\hookrightarrow Soma $+ 2 >> 2$ \rightarrow Aredondamento para baixo

$$\begin{aligned} \text{Ex: } \text{Soma} &= 147_{10} = 10010011_2 \\ + 2 &= 149_{10} = 10010101_2 \\ \text{Soma} &>> 2 = 00100101_2 = 37 \end{aligned}$$

$$\frac{147}{4} = 36,75$$

Aredondamento para cima

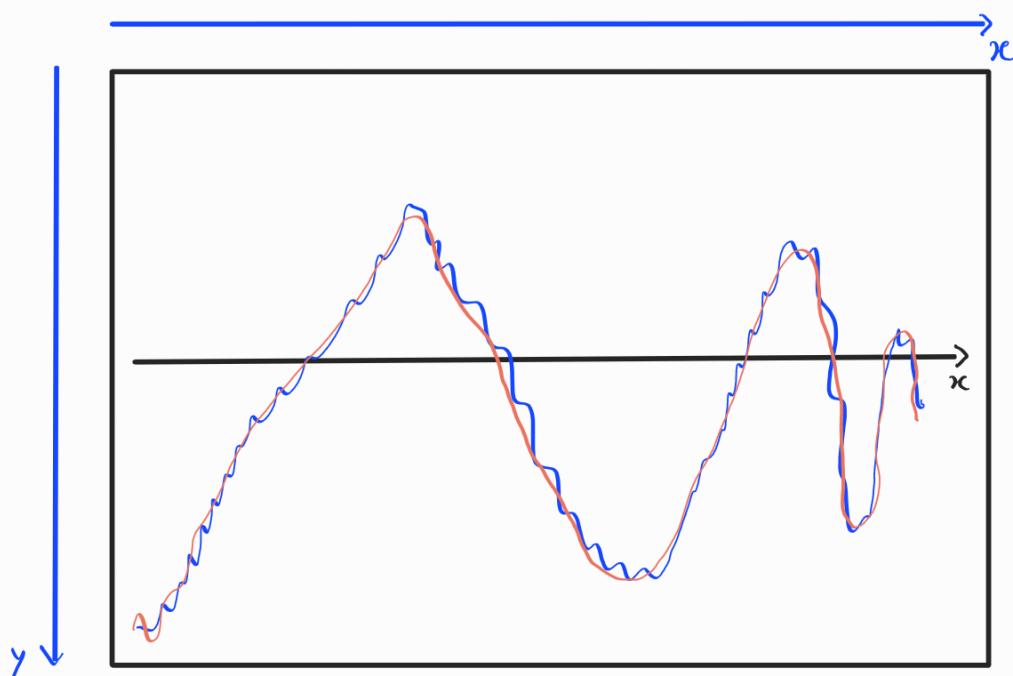
$$\begin{aligned} \text{Soma} &= 145_{10} = 10010001_2 \\ + 2 &= 148_{10} = 10010100_2 \\ \frac{\text{Soma}}{4} &= 00100100_2 = 36 \checkmark \\ \frac{145}{4} &= 36,25 \checkmark \end{aligned}$$

11: Aredondamento para cima:

\hookrightarrow Soma $>> 2 + 1$ \rightarrow Aredondamento para baixo

Se $B(0) = '1'$ or $B(1) = '1'$

$$\begin{array}{r} 0000 144 \rightarrow 36 \\ 0001 145 \rightarrow 37 \\ 0010 146 \rightarrow 37 \\ 0011 147 \rightarrow 37 \\ 0100 148 \rightarrow 37 \end{array}$$



(127)

(0) 256 dados
 $y = 256 - (x + 128)$

(-128)

(0)

256 dados

(255)

0x:

0 →	-77
1 →	-114
2 →	-85,5 → 110 1010 1010
3 →	-84,75 → 110 1010 11,01
4 →	-80,25 → 110 1011 11,11

10:

-77
-114
-86
-85
-80

11:

-77
-114
-86 - 86
-85
-84
-80