TRABALHO DE LABORATÓRIO IV
CIRCUITO DE PROCESSAMENTO DE DADOS

Trabalho realizado por: Diogo Martins Alves Nº 86980

Diogo Moura Nº 86976

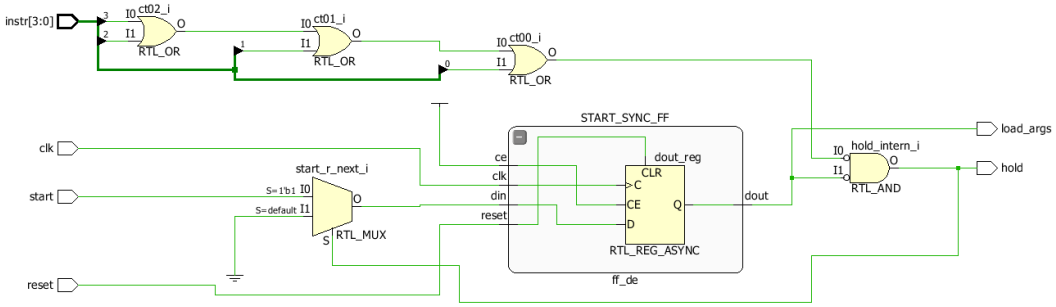
Dia: 02/12/2016 Hora: 13h00 Lab: 4 Grupo: 68 Docente:_____

1. Verifique o funcionamento do registo para números de vírgula flutuante (float_reg) através de uma simulação. Utilize o testbench especificado no ficheiro tb_float_register.vhd para o efeito (não coloque o resultado da simulação). Descreva como é implementado o deslocamento da mantissa.

Sendo B(7:0) uma variável de 8 bits introduzida na entrada num_in(7:0) de um dos registos RA ou RB em que B(7:3) é uma mantissa e B(2:0) é um expoente, o deslocamento aritmético para a direita (SRA) da mantissa B(7:3) é feito através da seguinte sequência de operações introduzidas na entrada op(2:0) do registo:
101 – Load (carrega o valor da entrada B(7:0) na saída num(7:0))
111 - SRA M; Ld E– carrega o valor da entrada arithmetic_in no expoente e faz o deslocamento para a direita aritmético da mantissa. Em VHDL, esta ultima operação utiliza a seguinte linha de código para fazer o deslocamento da mantissa:

```
mant_SRA <= mant_intern(4) & mant_intern(4 downto 1);
```

2. Apresente o logigrama do circuito CTR_CONTROL, tendo por base a descrição em VHDL fornecida em cntr_control.vhd (incluindo todos os sinais externos e internos, portas lógicas, componentes e largura em bits de cada ligação). Explique sucintamente qual é a função dos componentes dentro do CTR_CONTROL.



O componente start_r_next_i é um multiplexer 1:2 em que a entrada de seleção está ligada ao sinal "hold" – quando este é "0", a saída do multiplexer é "0" e quando é "1", a saída do multiplexer toma o valor da entrada "start". O componente START_SYNC_FF é um flip-flop tipo D positive edge-triggered, em que a entrada D está ligada à saída do multiplexer e o enable está ligado ao valor lógico "1".

3. Apresente o resultado e os valores intermédios do cálculo efetuados manualmente.

N1+N2 = 86976 N1+N2 = 1 0101 0011 1100 0000_b

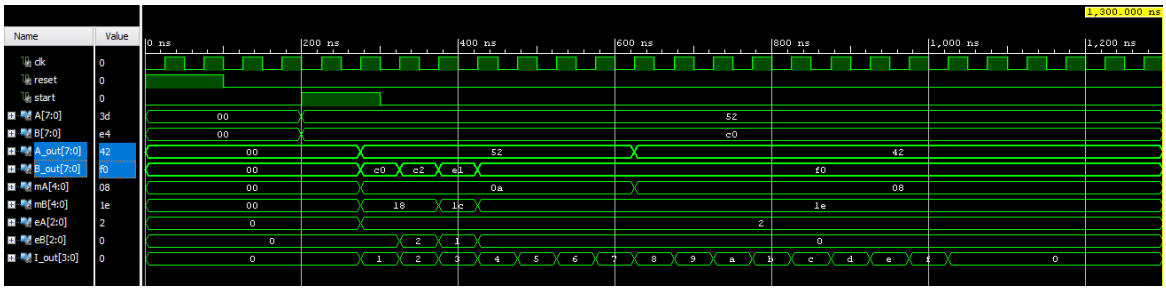
X =11000_b x 2^{000b} Y =01010_b x 2^{010b} X_{norm} =11000_b x 2^{000b} Y_{norm} = 01010_b x 2^{010b}

A =01010_b x 2^{010b} B =11000_b x 2^{000b} (depois da eventual troca)

A =01010_b x 2^{010b} B =11110_b x 2^{010b} (depois de ajustar o expoente)

A =01000_bx 2^{010b} (depois da soma)

4. Apresente um *snapshot* da simulação da simulação completa do circuito da Figura 2 e compare o resultado esperado com o resultado obtido através da simulação no VIVADO. Comente o resultado.



Analisando a simulação podemos ver que:

Valores iniciais: A = $52_{(16)}$ = $01010\_010_{(2)}$ B = $C0_{(16)}$ = $11000\_000_{(2)}$

Depois de ajustar o expoente: A = $52_{(16)}$ = $01010\_010_{(2)}$ B = $F0_{(16)}$ = $11110\_000_{(2)}$

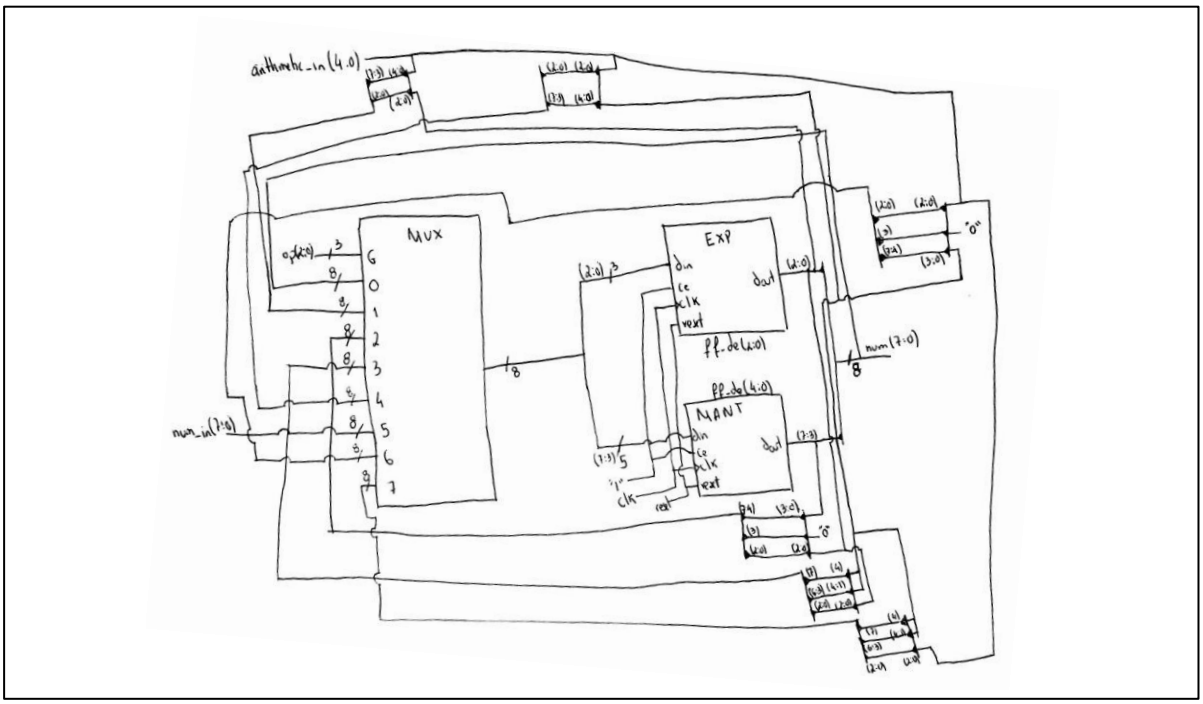
Depois da soma: A = $42_{(16)}$ = $01000\_010_{(2)}$

Concluímos assim que a simulação nos deu o resultado esperado para o valor da soma de A e B, uma vez que os valores tirados a partir desta são iguais aos valores calculados manualmente.

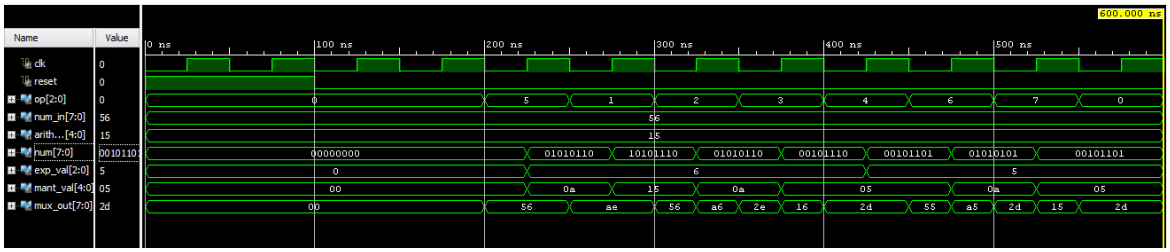
5. Efetue a mesma operação usando o circuito implementado na placa de prototipagem. Comente o funcionamento (**procedimento realizado na aula de laboratório**).

Na placa de prototipagem, o valor do display do lado esquerdo (op) foi percorrendo os valores de $0_{(16)}$ a $F_{(16)}$ por ordem e os dois displays do meio (mA) alteraram entre $0A_{(16)}$ e $08_{(16)}$ quando o valor de op mudou de $7_{(16)}$ para $8_{(16)}$. O display do lado direito (eA) mateve-se contante e igual a $2_{(16)}$. Ao compararmos estes valores com os da simulação, observamos que são exatemente iguais, logo a operação foi realizada como seria de esperar.

6. Apresente o logigrama do registo para números em virgula flutuante que projetou.



7. Apresente um snapshot da simulação do registo para números em virgula flutuante que projetou. TODAS as operações possíveis devem ser testadas na simulação.



8. Preencha a tabela de controlo (em baixo) de modo a realizar as duas instruções correspondentes ao seu turno.

I	RZ	RZ_CE	RA_OP(2:0)	RB_OP(2:0)	ALU_OP(1:0)
I1	X	1	111	000	11

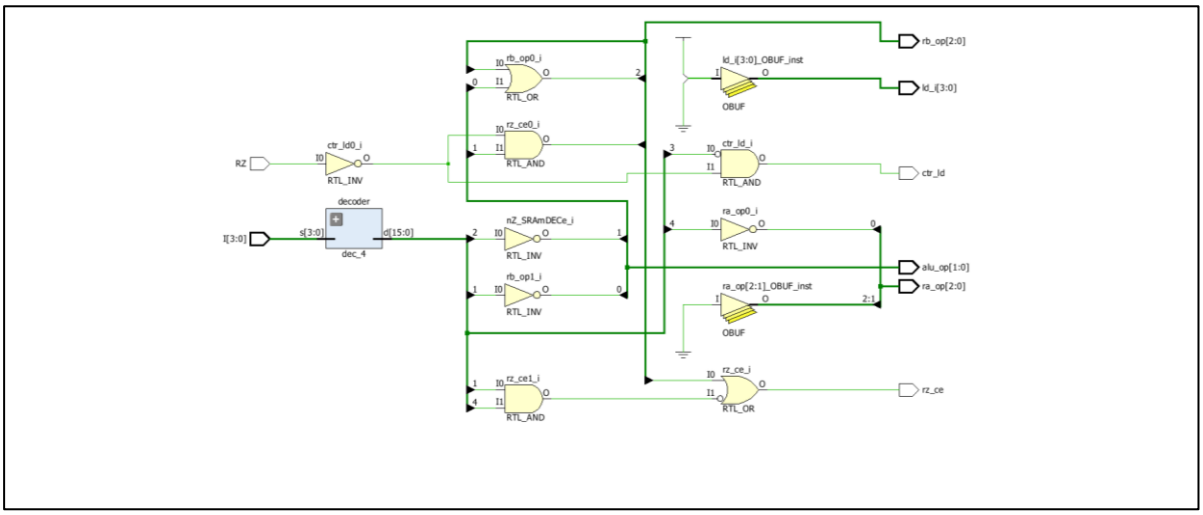
I	RZ	RZ_CE	RA_OP(2:0)	RB_OP(2:0)	ALU_OP(1:0)
I2	0	X	XXX	XXX	XX
	1	1	000	100	11

9. Complete a tabela com os sinais necessários para criar a instrução 3, de carregamento (nZ_LOAD_2), que carrega no contador 2 enquanto o valor de RZ for '0', e os valores que as saídas CTR_LD e LD_I(3:0) devem tomar para não afetar as restantes instruções.

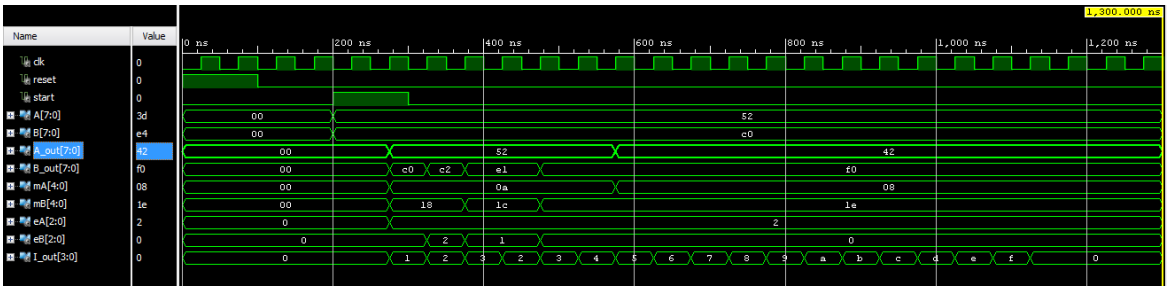
Tabela 1 – Tabela do controlo.

I(3:0)	RZ	CTR_LD	LD_I(3:0)	RZ_CE	RA_OP(2:0)	RB_OP(2:0)	ALU_OP(1:0)	Descrição
0000	**X**	0	XXXX	X	XXX	XXX	XXX	Reserved
0001	**X**	0	XXXX	1	000	100	01	SUBe
0010	**0**	0	XXXX	1	000	111	10	nZ_SRAmDECe
	1	0	XXXX	0	000	000	XX	NOP
0011	**0**	1	0010	0	000	000	XX	nZ_LOAD_2
	1	0	XXXX	0	000	000	XX	NOP
0100	**X**	0	XXXX	1	001	000	00	ADD (mA, mB)
0101	**X**	0	XXXX	0	000	000	XX	NOP
...	**X**	0	XXXX	0	000	000	XX	NOP
1111	**X**	0	XXXX	0	000	000	XX	NOP

10. Apresente o logigrama do controlador (controlo_v2.vhd) correspondente à nova implementação da sequência de instruções.



11. Apresente um snapshot da simulação do sistema completo utilizando o novo controlador nas mesmas condições da pergunta 4.



12. Efetue a operação usando o circuito implementado na placa de prototipagem. Comente o funcionamento (**procedimento realizado na aula de laboratório**).

Na placa de prototipagem, o valor do display do lado esquerdo (op) foi percorrendo os valores de $0_{(16)}$ a $F_{(16)}$, sendo que quando chegou a $3_{(16)}$, voltou até $2_{(16)}$ e a partir daí continuou normalmente até $F_{(16)}$. Observámos que o valor de mA alterou entre $0A_{(16)}$ e $08_{(16)}$ quando o valor de op mudou de $4_{(16)}$ para $5_{(16)}$. O display do lado direito (eA) mateve-se contante e igual a $2_{(16)}$. Ao compararmos estes valores com os da simulação, observamos que são exatemente iguais, logo a operação foi realizada como seria de esperar.

13. Complete a tabela com os sinais necessários para tratar os casos de overflow na soma das mantissas. Considere a sequência de operações da pergunta 9 mesmo que não a tenha resolvido (deixe as valores de CTR_LD e LD_I(3:0) em branco).

Tabela 2 – Tabela do controlo 2.

I(3:0)	RV	RZ	CTR_LD	LD_I	RZ(V)_CE	RA_OP(2:0)	RB_OP(2:0)	ALU_OP(1:0)	Descrição
0000	**X**	**X**	0	XXXX	X	XXX	XXX	XX	Reserved
0001	**X**	**X**	0	XXXX	1	000	100	01	SUBe
0010	**X**	**0**	0	XXXX	1	000	111	10	nZ_SRAmDECe
	X	**1**	0	XXXX	0	000	000	XX	NOP
0011	**X**	**0**	1	0010	0	000	000	XX	nZ_LOAD_2
	X	**1**	0	XXXX	0	000	000	XX	NOP
0100	**X**	**X**	0	XXXX	1	000	000	00	TEST mA + mB
0101	**0**	**X**	0	XXXX	1	001	000	00	ADD (mA, mB)
	1	**X**	0	XXXX	0	000	101	XX	Load(B)
0110	**0**	**X**	0	XXXX	0	000	000	XX	NOP
	1	**X**	0	XXXX	0	111	000	11	nV_SRAmA_INCeA
0110	**0**	**X**	0	XXXX	0	000	000	XX	NOP
	1	**X**	1	0010	0	000	000	XX	nZ_LOAD_2
1000	**0**	**X**	0	XXXX	0	000	000	XX	NOP
	1	**X**	0	XXXX	0	000	000	XX	NOP
…			0	XXXX	0	000	000	XX	NOP
1111	**0**	**X**	0	XXXX	0	000	000	XX	NOP
	1	**X**	0	XXXX	0	000	000	XX	NOP