Laboratório 2
- ULA e FPULA -

Grupo 6
Alunos:

- Caio Couto Moreira (15/0155433)
- Gabriel Alcantara(15/0126701)
- Breno Felipe(16/0003318)
- João Pedro Ramos do Amaral(14/0056394)

Objetivos: Introduzir ao aluno a Linguagem de Descrição de Hardware Verilog; Familiarizar o aluno com a plataforma de desenvolvimento FPGA DE1-SoC e o software QUARTUS Prime da Intel; Desenvolver a capacidade de análise, síntese e caracterização de sistemas digitais usando HDL;

1) (0.0) Implementação de um driver para display de 7 segmentos e Multiplexadores
RESPOSTA: feita em sala

2) (1.0) Multiplexadores:

RESPOSTA:
a) 1 Bit

	N° de alms	Temporais (tpd) FF
Mux 2x1	**1**	**6.233 ns**
Mux 4x1	**2**	**6.293 ns**

b) 5 Bits

	N° de Alms	Temporais(tdp) FF
Mux 2X1	**3**	**7.148 ns**
Mux 4x1	**6**	**6.961 ns**
Mux 8x1	**13**	**11.482 ns**

c) 12 Bits

	N° de Alms	Temporais(tpd)FF
Mux 2x1	**7**	**10.677 ns**
Mux 4x1	**13**	**10.858 ns**

Mux 8x1	**31**	**11.604ns**
Mux 16x1	**61**	**12.644ns**

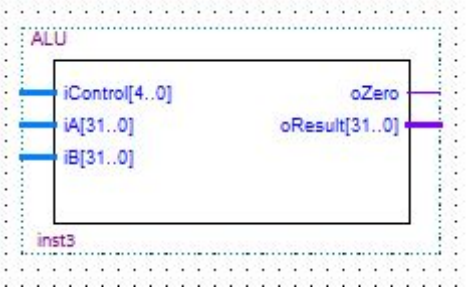
d)32 Bits

	N° de Alms	**Temporais(tpd)FF**
Mux 2x1	**17**	**10.389 ns**
Mux 4x1	**33**	**10.714 ns**
Mux 8x1	**49**	**11.740 ns**
Mux 16x1	**81**	**13.244 ns**
Mux 32x1	**178**	**15.367 ns**

3) (4.5) Unidade Lógico Aritmética de Inteiros:

a)Para a ULA de inteiros assíncrona fornecida, desenhe o diagrama em blocos, descreva suas funções e escreva a tabela verdade de seus códigos para cada operação
RESPOSTA:

Diagrama em blocos:



Dependendo do sinal de controle **iControl[4..0]**, as operações serão dadas pelos valores em **iA[31..0]** e pelo **iB[0..31]** sendo o output dado pelo **oResult[31..0]**. O **oZero** é simplesmente a resposta de valor zero.

A imagem abaixo conseguimos visualizar a ULA como um grande multiplexador.

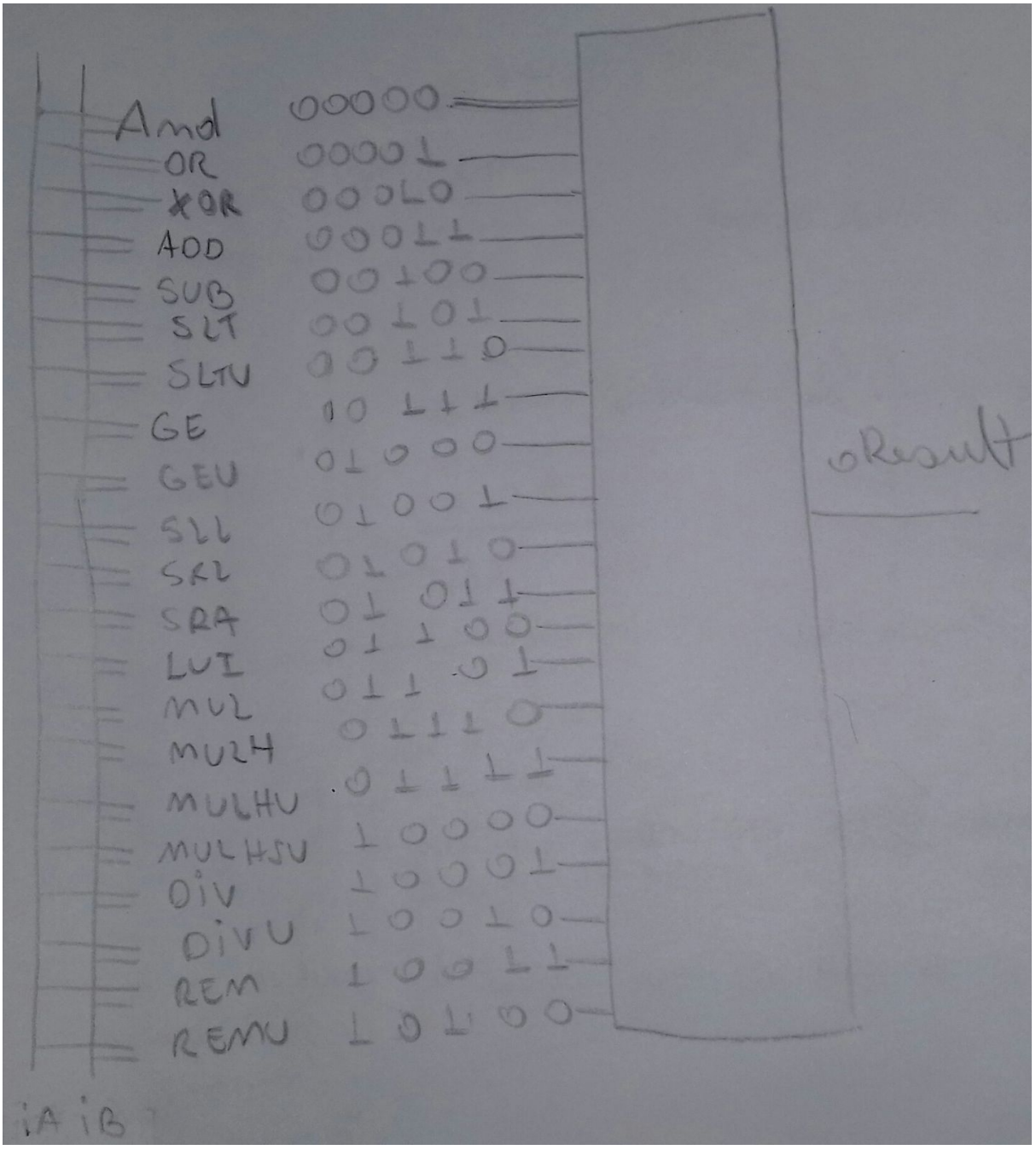


Tabela Verdade das operações:

```
-----------------

0  0  0  0  0  | OPAND= 5'd0,
0  0  0  0  1  | OPOR= 5'd1,
0  0  0  1  0  | OPXOR= 5'd2,
0  0  0  1  1  | OPADD= 5'd3,
0  0  1  0  0  | OPSUB= 5'd4,
0  0  1  0  1  | OPSLT= 5'd5,
0  0  1  1  0  | OPSLTU= 5'd6,
0  0  1  1  1  | OPGE= 5'd7,
0  1  0  0  0  | OPGEU= 5'd8,
0  1  0  0  1  | OPSLL= 5'd9,
0  1  0  1  0  | OPSRL= 5'd10,
0  1  0  1  1  | OPSRA= 5'd11,
0  1  1  0  0  | OPLUI= 5'd12,
0  1  1  0  1  | OPMUL= 5'd13,
0  1  1  1  0  | OPMULH= 5'd14,
0  1  1  1  1  | OPMULHU= 5'd15,
1  0  0  0  0  | OPMULHSU= 5'd16,
1  0  0  0  1  | OPDIV= 5'd17,
1  0  0  1  0  | OPDIVU= 5'd18,
1  0  0  1  1  | OPREM= 5'd19,
1  0  1  0  0  | OPREMU= 5'd20,
1  0  1  0  1  |
1  0  1  1  0  |
1  0  1  1  1  |
1  1  0  0  0  |
1  1  0  0  1  |
1  1  0  1  0  |
1  1  0  1  1  |
1  1  1  0  0  |
1  1  1  0  1  |
1  1  1  1  0  |
1  1  1  1  1  |
```

Para cada valor da tabela verdade de “b0 0 0 0 0 = d0” até o valor “b1 0 1 0 0 = d20”, será atribuído a uma operação que nossa ula é capaz de fazer, dessa forma quando executamos nossa ULA no Quartus, temos um switch case que seta esses valores em iControl[0..4] e consegue saber qual operação vamos utilizar..

b) Crie um testbech e verifique com o ModelSim cada operação implementada.
RESPOSTA:
Utilizamos o arquivo ALU_tb.v para descrever as operações e valores que seriam operados, abaixo é listado os valores utilizados:

```
IA = -32'd1;
IB = 32'd77;
ICONTROL = OPAND;

#10
IA = 32'd0;
IB = 32'd1;
ICONTROL = OPOR;

#10
IA = 32'd777;
IB = 32'd77;
ICONTROL = OPXOR;

#10
IA = 32'd4294967295
IB = 32'd1;
ICONTROL = OPADD;

#10
IA = 32'd0;
IB = 32'd1;
ICONTROL = OPSUB;

#10
IA = 32'd2;
IB = 32'd1;
ICONTROL = OPSLT;
```

```
#10
IA = -32'd2;
IB = 32'd1;
ICONTROL = OPSLTU;

#10
IA = 32'd0;
IB = 32'd0;
ICONTROL = OPGE;

#10
IA = -32'd1;
IB = 32'd1;
ICONTROL = OPGEU;

#10
IA = 32'd128;
IB = 32'd1;
ICONTROL = OPSLL;

#10
IA = 32'd128;
IB = 32'd1;
ICONTROL = OPSRL;

#10
IA = 32'd2;
IB = 32'd1;
ICONTROL = OPSRA;
```

```
#10
IA = 32'd16;
IB = 32'd1;
ICONTROL = OPLUI;

#10
IA = 32'd4294967294;
IB = 32'd2;
ICONTROL = OPMUL;

#10
IA = 32'd4294967294;
IB = 32'd2;
ICONTROL = OPMULH;

#10
IA = 32'd4294967294;
IB = -32'd2;
ICONTROL = OPMULHU;

#10
IA = 32'd4294967294;
IB = 32'd2;
ICONTROL = OPMULHSU;

#10
IA = 32'd2;
IB = 32'd0;
ICONTROL = OPDIV;
```

```
#10
IA = 32'd2;
IB = -32'd1;
ICONTROL = OPDIVU;

#10
IA = 32'd12;
IB = 32'd5;
ICONTROL = OPREM;

#10
IA = -32'd12;
IB = 32'd5;
ICONTROL = OPREMU;
```

Acima é possível ver o valor #10 colocado antes de inserir os valores IA, IB e ICONTROL,, esse #10 quer dizer que após 10 ciclos de clock ele fará a operação. Então uma sequência de comandos que queremos testar a ULA.

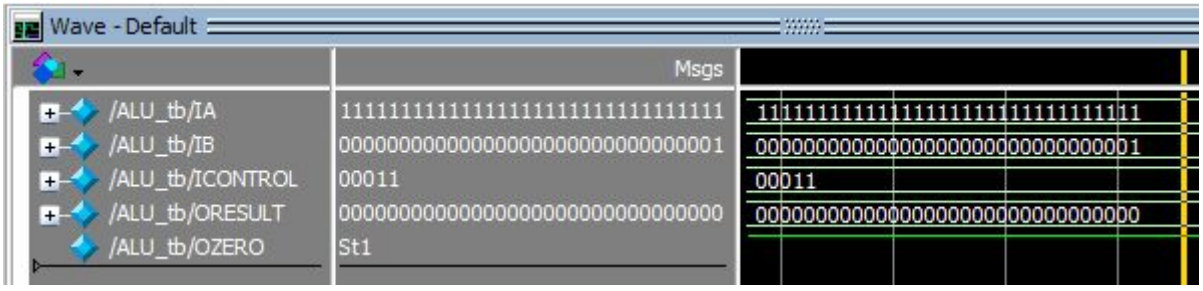
Feito isso quando abrimos o modelsim recebemos um resumo com a resposta de toda nossas operações no Transcript.

Transcript

```
# .main_pane.objects.interior.cs.body.tree
# run -all
#                    0 << Starting Simulation >>
#                time,    iA,      iB,      iControl, oResult, oZero
#                     0, ffffffff, 0000004d, 00, 0000004d, 0,
#                    10, 00000000, 00000001, 01, 00000001, 0,
#                    20, 00000309, 0000004d, 02, 00000344, 0,
#                    30, ffffffff, 00000001, 03, 00000000, 1,
#                    40, 00000000, 00000001, 04, ffffffff, 0,
#                    50, 00000002, 00000001, 05, 00000000, 1,
#                    60, fffffffe, 00000001, 06, 00000000, 1,
#                    70, 00000000, 00000000, 07, 00000001, 0,
#                    80, ffffffff, 00000001, 08, 00000001, 0,
#                    90, 00000080, 00000001, 09, 00000001, 0,
#                   100, 00000080, 00000001, 0a, 00000001, 0,
#                   110, 00000002, 00000001, 0b, 00000000, 1,
#                   120, 00000010, 00000001, 0c, 00010000, 0,
#                   130, fffffffe, 00000002, 0d, fffffffc, 0,
#                   140, fffffffe, 00000002, 0e, ffffffff, 0,
#                   150, fffffffe, fffffffe, 0f, fffffffc, 0,
#                   160, fffffffe, 00000002, 10, 00000001, 0,
#                   170, 00000002, 00000000, 11, xxxxxxxx, x,
#                   180, 00000002, ffffffff, 12, 00000000, 1,
#                   190, 0000000c, 00000005, 13, 00000002, 0,
#                   200, fffffff4, 00000005, 14, 00000004, 0,
#                   300<< Simulation Complete >>
# ** Note: $stop    : C:/Users/Gabri/Desktop/OAC/lab2/q3/ALU/ALU_tb.v(174)
#    Time: 300 ns  Iteration: 0  Instance: /ALU_tb
```

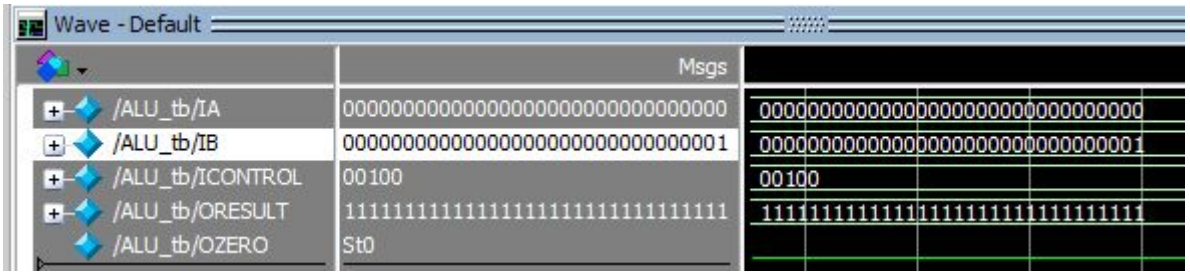
Abaixo alguns exemplos da wave que tem resultado das operações, geradas pelo ModelSim:

- **ADD:**

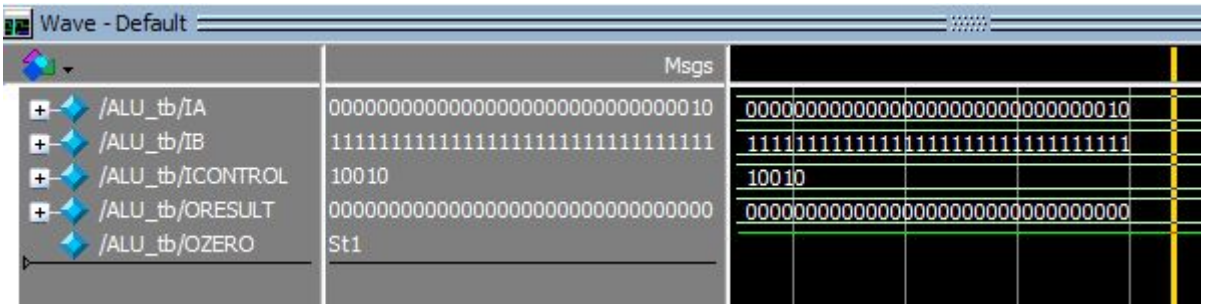


Os valores de entrada para soma de IA + IB, foram feita de forma que o resultado faça com que o registrador chegue no seu máximo de bits recomeçando em 0. Em IA é colocado o valor 4294967296(o valor na ula é -1), e em IB coloca-se 1.

- **SUB:**

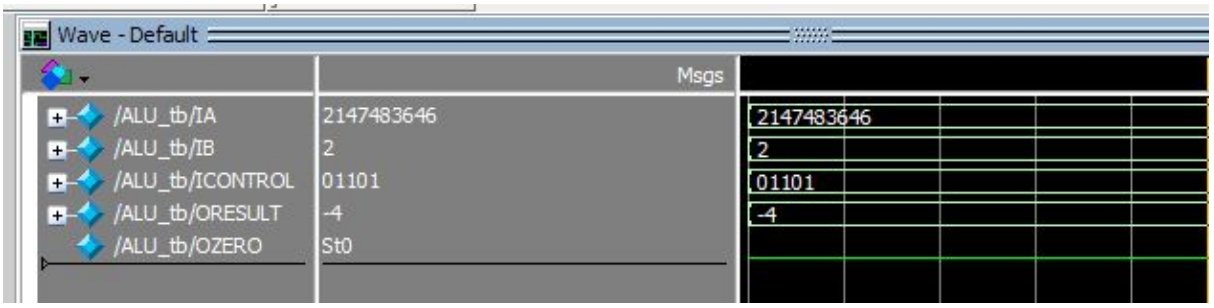


- **DIV(divisão por zero):**



Na divisao por zero, a ula está retornando zero.

- **MUL(multipicação com overflow)**



Deixando os valores de IA e IB em decimal na wave podemos ver que houve overflow, uma vez que um registrador de 32 bits cabe somente entre -2,147,483,647 até 2,147,483,647.

c) Indique os requisitos físicos da implementação da ULA total e para cada operação separadamente: i) Número de Elementos Lógicos (ALMs), ii) Número de Registradores e iii) Quantidade de bits de memória e iv) Número de blocos DSP usados;
RESPOSTA:

- **ADD:** 23 ALMs needed

Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins
39 (39)	0 (0)	0	0	97

- **AND: 27** ALMs needed

Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins	Virtual Pins
43 (43)	0 (0)	0	0	97	0

- **DIV: 646** ALMs needed

Compilation Hierarchy Node	Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins
|ALU	1273 (16)	0 (0)	0	0	97
|lpm_divide:Div0|	1257 (0)	0 (0)	0	0	0
|lpm_divide_rqo:auto_generated|	1257 (0)	0 (0)	0	0	0
|abs_divider_4dg:divider|	1257 (65)	0 (0)	0	0	0
|alt_u_div_o2f:divider|	1128 (1128)	0 (0)	0	0	0
|lpm_abs_4p9:my_abs_den|	32 (32)	0 (0)	0	0	0
|lpm_abs_4p9:my_abs_num|	32 (32)	0 (0)	0	0	0

- **DIVU: 572** ALMs needed

Compilation Hierarchy Node	Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins
|ALU	1137 (8)	0 (0)	0	0	97
|lpm_divide:Div1|	1129 (0)	0 (0)	0	0	0
|lpm_divide_3dm:auto_generated|	1129 (0)	0 (0)	0	0	0
|sign_div_unsign_9nh:divider|	1129 (0)	0 (0)	0	0	0
|alt_u_div_o2f:divider|	1129 (1129)	0 (0)	0	0	0

- **GE: 31** ALMs needed

Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins	Virtual Pins
40 (40)	0 (0)	0	0	97	0

- **GEU: 31** ALMs needed

Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins	Virtual Pins
40 (40)	0 (0)	0	0	97	0

- **LUI: 4** ALMs needed

Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins	Virtual Pins
4 (4)	0 (0)	0	0	97	0

- **MUL: 12** ALMs needed

Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins	Virtual Pins
21 (21)	0 (0)	0	2	97	0

- **MULH: 30** ALMs needed

Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins	Virtual Pins
53 (53)	0 (0)	0	3	97	0

- **MULHSU: 30** ALMs needed

Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins	Virtual Pins
53 (53)	0 (0)	0	3	97	0

- **MULHU: 30** ALMs needed

Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins	Virtual Pins
53 (53)	0 (0)	0	3	97	0

- **OR: 27** ALMs needed

Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins	Virtual Pins
43 (43)	0 (0)	0	0	97	0

- **REM: 678** ALMs needed

Compilation Hierarchy Node	Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins
|ALU	1336 (16)	0 (0)	0	0	97
|lpm_divide:Mod0|	1320 (0)	0 (0)	0	0	0
|lpm_divide_uio:auto_generated|	1320 (0)	0 (0)	0	0	0
|abs_divider_4dg:divider|	1320 (94)	0 (0)	0	0	0
|alt_u_div_o2f:divider|	1162 (1162)	0 (0)	0	0	0
|lpm_abs_4p9:my_abs_den|	32 (32)	0 (0)	0	0	0
|lpm_abs_4p9:my_abs_num|	32 (32)	0 (0)	0	0	0

- **REMU: 597** ALMs needed

Compilation Hierarchy Node	Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins
|ALU	1183 (15)	0 (0)	0	0	97
|lpm_divide:Mod1|	1168 (0)	0 (0)	0	0	0
|lpm_divide_65m:auto_generated|	1168 (0)	0 (0)	0	0	0
|sign_div_unsign_9nh:divider|	1168 (0)	0 (0)	0	0	0
|alt_u_div_o2f:divider|	1168 (1168)	0 (0)	0	0	0

- **SLL: 81** ALMs needed

Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins	Virtual Pins
106 (106)	0 (0)	0	0	97	0

- **SLT: 31** ALMs needed

Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins	Virtual Pins
40 (40)	0 (0)	0	0	97	0

- **SLTU: 31** ALMs needed

Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins	Virtual Pins
40 (40)	0 (0)	0	0	97	0

- **SRA: 86** ALMs needed

Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins	Virtual Pins
109 (109)	0 (0)	0	0	97	0

- **SRL: 82** ALMs needed

Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins	Virtual Pins
109 (109)	0 (0)	0	0	97	0

- **SUB: 23** ALMs needed

Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins	Virtual Pins
39 (39)	0 (0)	0	0	97	0

- **XOR: 27** ALMs needed

Combinational ALUTs	Dedicated Logic Registers	Block Memory Bits	DSP Blocks	Pins	Virtual Pins
43 (43)	0 (0)	0	0	97	0

d) Usando o TimeQuest, indique os requerimentos temporais para a ULA total e para cada operação separadamente**.**

RESPOSTA:
Para,
i) o caminho de maior atraso: estaremos observando a primeira e segunda coluna.
ii) maior tempo de atraso tpd: estaremos observando a última coluna.

Abaixo é descrito o **i)** e **ii)** de todas operações da ULA de inteiros.

- AND:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iA[7]	oResult[7]	11.453			11.973

- OR:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iA[7]	oResult[7]	11.453			11.973

- XOR:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iA[7]	oZero	11.442	12.192	11.826	12.611

- ADD:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iA[8]	oZero	14.304	14.185	14.696	14.577

- SUB:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iB[17]	oZero	14.850	15.390	15.474	16.004

- SLT:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iB[3]	oResult[0]	12.895	13.749	13.113	14.107

- SLTU:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iB[3]	oResult[0]	12.895	13.749	13.113	14.107

- GE:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iB[3]	oZero	12.832	13.643	13.050	14.001

- GEU:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iB[3]	oZero	12.832	13.643	13.050	14.001

- SLL:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iB[5]	oResult[28]	13.531			14.594

- SRL:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iB[18]	oResult[16]	16.082			19.094

- SRA:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iB[16]	oResult[8]	16.781			19.788

- LUI:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iA[12]	oResult[24]	9.704			10.028

- MUL:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iB[13]	oResult[25]	17.952	18.208	18.627	18.883

- MULH:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iA[30]	oResult[12]	19.027	19.753	19.912	20.638

- MULHU:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iB[15]	oResult[3]	19.941	20.840	20.925	21.824

- MULHSU:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iB[15]	oResult[3]	19.941	20.840	20.925	21.824

- DIV:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iB[8]	oZero	102.055	103.160	102.499	103.604

- DIVU:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iB[28]	oZero	96.660	96.986	98.109	98.435

- REM:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iB[15]	oResult[16]	103.793	105.070	104.167	105.444

- REMU:

Propagation Delay

	Input Port	Output Port	RR	RF	FR	FF
1	iB[27]	oResult[7]	94.250	95.134	96.006	96.890

e) Defina o arquivo TopDE.v como o toplevel, sintetize na DE1-SoC e filme o funcionamento comprovando seu correto funcionamento.
RESPOSTA:

O vídeo se encontra no seguinte link: https://www.youtube.com/watch?v=PO64S9fX47Y

4) (4.5) Unidade Aritmética de Ponto Flutuante:

a)Para a FPULA síncrona fornecida, desenhe o diagrama em blocos, descreva suas funções e escreva a tabela verdade de seus códigos para cada operação

RESPOSTA:
Diagrama de blocos:

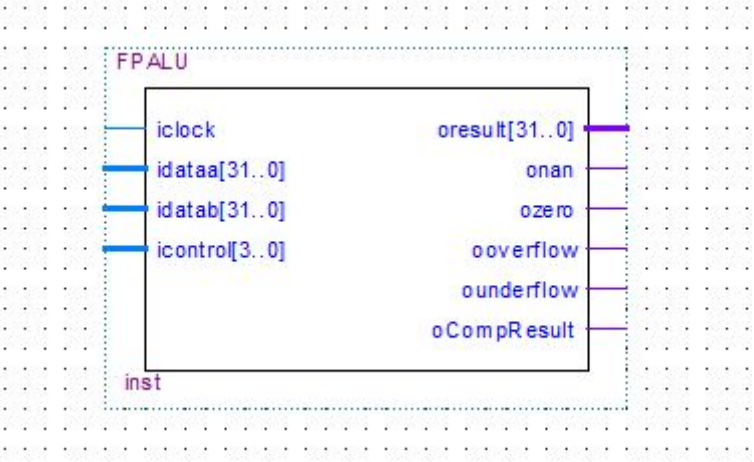


Tabela verdade:

Bit 3	Bit 2	Bit 1	Bit 0	Função
0	0	0	0	FOPADD = 4'd0,
0	0	0	1	FOPSUB = 4'd1,
0	0	1	0	FOPMUL = 4'd2,
0	0	1	1	FOPDIV = 4'd3,
0	1	0	0	FOPSQRT = 4'd4,
0	1	0	1	FOPABS = 4'd5,;
0	1	1	0	FOPNEG = 4'd6,
0	1	1	1	FOPCEQ = 4'd7,
1	0	0	0	FOPCLT = 4'd8,
1	0	0	1	FOPCLE = 4'd9,
1	0	1	0	FOPCVTSW = 4'd10,
1	0	1	1	FOPCVTWS = 4'd11

Funções:

FOPADD esta operação realiza uma soma entre dois números binários codificados como pontos flutuantes simples.

FOPSUB esta operação realiza uma subtração entre dois números binários codificados como pontos flutuantes simples.

FOPMUL esta operação realiza uma multiplicação entre dois números binários codificados como pontos flutuantes simples.

FOPDIV esta operação realiza uma divisão entre dois números binários codificados como pontos flutuantes simples.

FOPSQRT esta operação realiza uma raíz quadrada sobre um número binário codificado como ponto flutuante simples.

FOPABS esta operação retorna o valor absoluto de um número binário codificado em ponto flutuante simples, ou seja o número sem sinal.

FOPNEG esta operação realiza uma negação sobre um número binário codificado como ponto flutuante simples.

FOPCEQ esta operação realiza uma comparação entre dois números binários codificados como pontos flutuantes simples e diz se estes são iguais.

FOPCLT esta operação realiza uma comparação entre dois números binários codificados como pontos flutuantes simples e diz se um é menor que o outro.

FOPCLE esta operação realiza uma comparação entre dois números binários codificados como pontos flutuantes simples e diz se um é menor ou igual ao outro.

FOPCVTSW esta função converte um número binário codificado em ponto flutuante simples em uma word.

FOPCVTWS esta função converte uma wor em um número binário codificado em ponto flutuante simples.

b)Crie um testbech e verifique com o ModelSim cada operação implementada.

RESPOSTA:

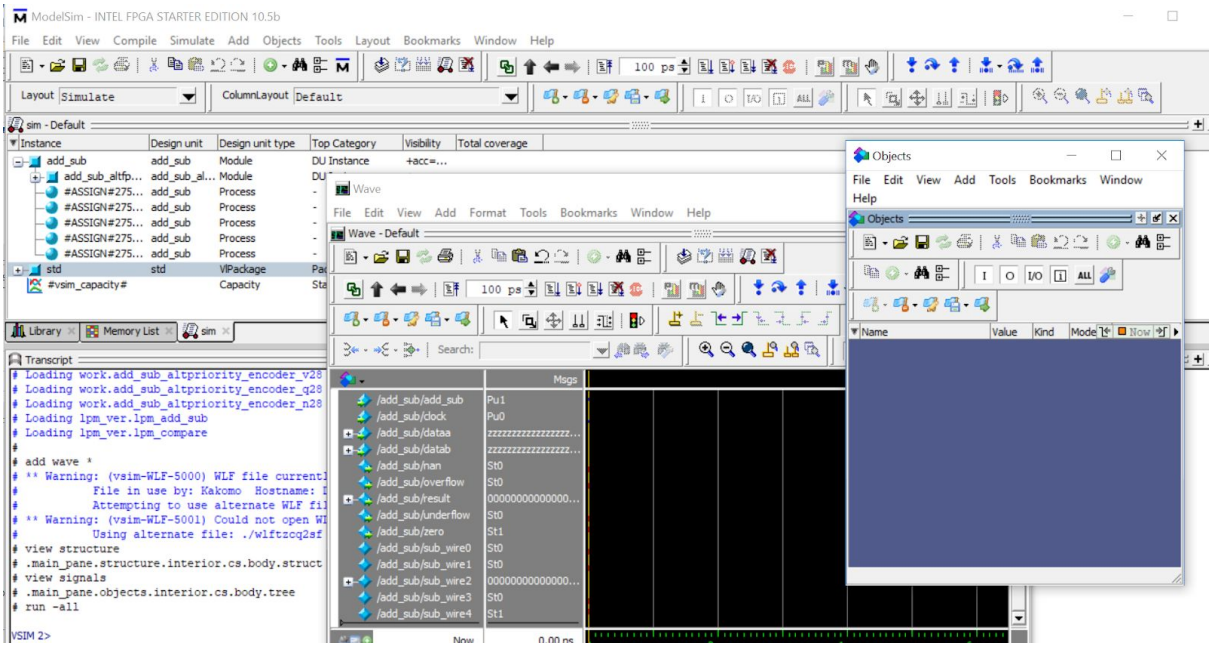
valores gerado no modelsim, com o testes no testbech da fpalu

Transcript

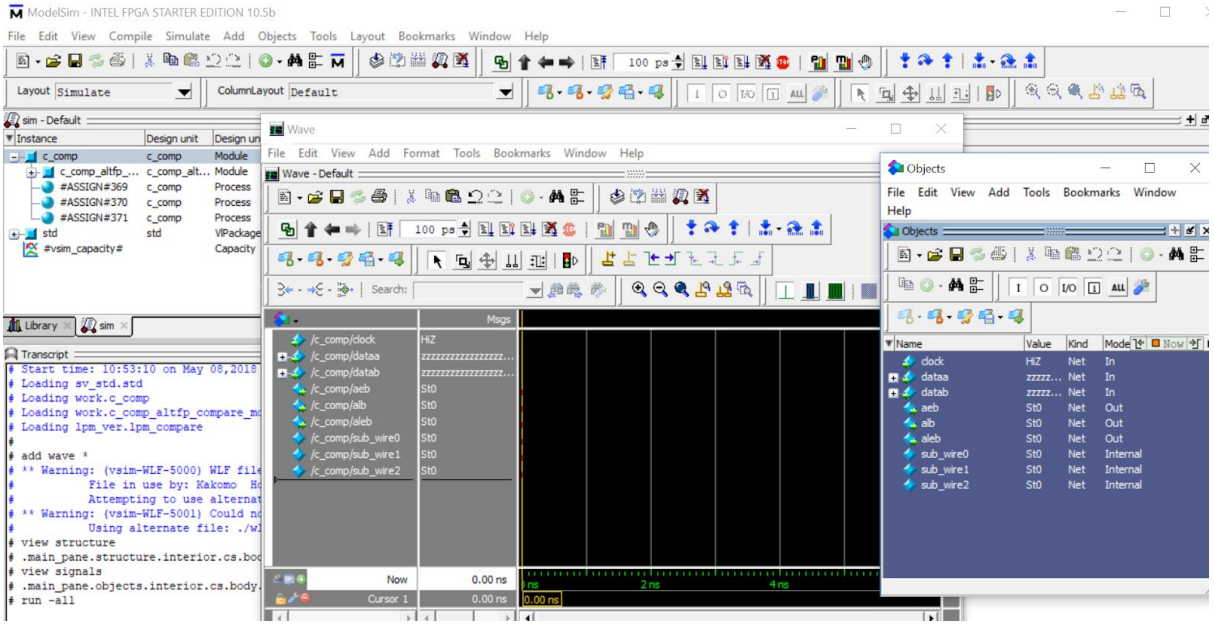
```
# ** Warning: (vsim-3722) C:/Users/Gabri/Desktop/OAC/lab2/q4/FPALU/FPALU.v(220): [TFMPC] - Missing connection for port
#
# add wave *
# view structure
# .main_pane.structure.interior.cs.body.struct
# view signals
# .main_pane.objects.interior.cs.body.tree
# run -all
#                    0 << InÃ-cio da SimulaÃ§Ã£o >>
#                  time, iA,       iB,       iControl, oResult,      oZero, oNaN, oVerflow, oUnderflow, oCompResult
#                    0,  3f800000, 40000000,  0,       00000000,      1,    0,    0,         0,          0
#                   65,  3f800000, 40000000,  0,       40400000,      0,    0,    0,         0,          0
#                  200,  40000000, 40800000,  1,       40400000,      0,    0,    0,         0,          0
#                  265,  40000000, 40800000,  1,       c0000000,      0,    0,    0,         0,          0
#                  400,  40400000, 3fc00000,  2,       41000000,      0,    0,    0,         0,          0
#                  445,  40400000, 3fc00000,  2,       40900000,      0,    0,    0,         0,          0
#                  600,  40400000, 40000000,  3,       40000000,      0,    0,    0,         0,          0
#                  655,  40400000, 40000000,  3,       3fc00000,      0,    0,    0,         0,          0
#                  800,  41500000, 00000000,  4,       3fddb3d7,      0,    0,    0,         0,          0
#                  955,  41500000, 00000000,  4,       4066c15a,      0,    0,    0,         0,          0
#                 1000,  bf800000, 00000000,  5,       3f800000,      0,    0,    0,         0,          0
#                 1020,  3f800000, 00000000,  6,       bf800000,      0,    0,    0,         0,          0
#                 1220,  3f800000, c0800000,  7,       00000000,      0,    0,    0,         0,          0
#                 1420,  3f800000, 40000000,  8,       00000000,      0,    0,    0,         0,          0
#                 1425,  3f800000, 40000000,  8,       00000000,      0,    0,    0,         0,          1
#                 1620,  3f800000, 40000000,  9,       00000000,      0,    0,    0,         0,          1
#                 1820,  00000004, 00000000, 10,       4e7e0000,      0,    0,    0,         0,          0
#                 1875,  00000004, 00000000, 10,       40800000,      0,    0,    0,         0,          0
#                 2020,  40800000, 00000000, 11,       00000000,      1,    0,    0,         0,          0
#                 2075,  40800000, 00000000, 11,       00000004,      0,    0,    0,         0,          0
#                 2220<< Final da SimulaÃ§Ã£o >>
# ** Note: $stop    : C:/Users/Gabri/Desktop/OAC/lab2/q4/FPALU/FPALU_tb.v(100)
#    Time: 2220 ns  Iteration: 0  Instance: /FPALU_tb
# Break in Module FPALU_tb at C:/Users/Gabri/Desktop/OAC/lab2/q4/FPALU/FPALU_tb.v line 100
```

Ativ

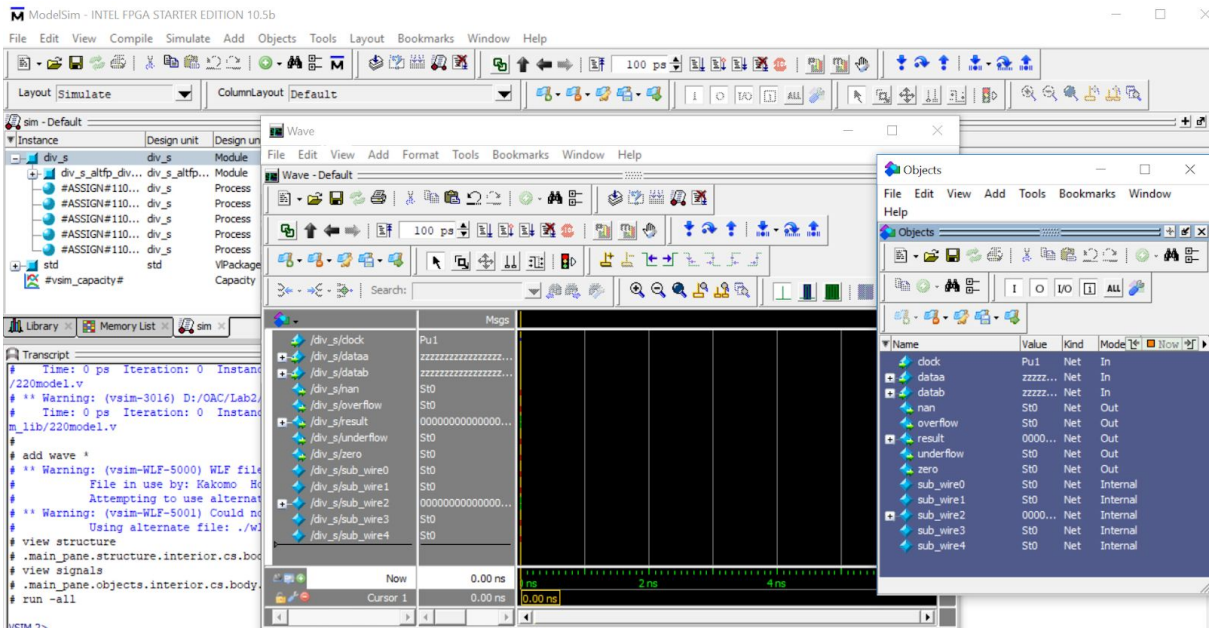
adição e subtração



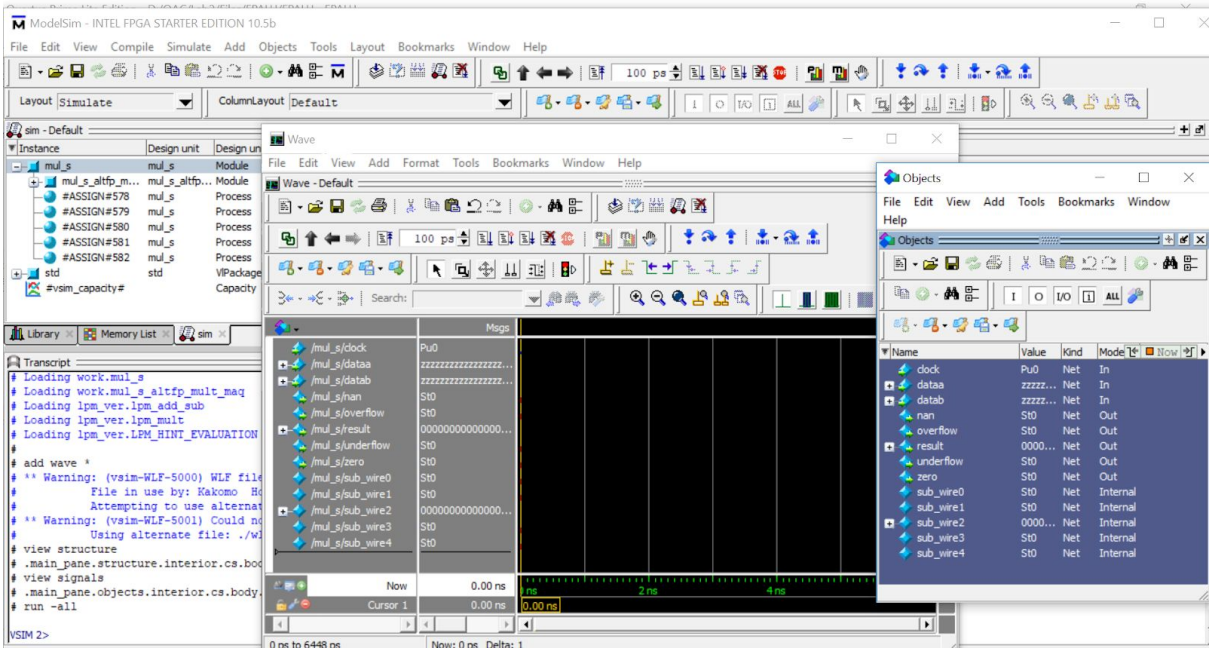
comparação



divisão



multiplexador



Os Teste Benchs foram complicados de fazer por conta de um bug no nome do projeto.
Não entendi como fazer para gerar resultados singulares.

c)Indique os requisitos físicos da implementação da FPULA total e para cada operação separadamente: i) Número de Elementos Lógicos (ALMs), ii) Número de Registradores e iii) Quantidade de bits de memória e iv) Número de blocos DSP usados;

RESPOSTA:

FOPADD:

	Resource	Usage
1	Estimate of Logic utilization (ALMs needed)	386
2		
3	Combinational ALUT usage for logic	604
4		
5	Dedicated logic registers	375
6		
7	I/O pins	102
8		
9	Total DSP Blocks	0

FOPSUB:

	Resource	Usage
1	Estimate of Logic utilization (ALMs needed)	386
2		
3	C… Estimate of Logic utilization (ALMs needed)	604
4		
5	Dedicated logic registers	375
6		
7	I/O pins	102
8		
9	Total DSP Blocks	0

FOPMUL:

	Resource	Usage
1	Estimate of Logic utilization (ALMs needed)	136
2		
3	Combinational ALUT usage for logic	156
4		
5	Dedicated logic registers	242
6		
7	I/O pins	102
8		
9	Total DSP Blocks	1

FOPDIV:

	Resource	Usage
1	Estimate of Logic utilization (ALMs needed)	247
2		
3	Combinational ALUT usage for logic	204
4		
5	Dedicated logic registers	419
6		
7	I/O pins	102
8	Total MLAB memory bits	0
9	Total block memory bits	4640
10		
11	Total DSP Blocks	6

FOPSQRT:

	Resource	Usage
1	Estimate of Logic utilization (ALMs needed)	291
2		
3	Combinational ALUT usage for logic	457
4		
5	Dedicated logic registers	507
6		
7	I/O pins	102
8	Total MLAB memory bits	0
9	Total block memory bits	143
10		
11	Total DSP Blocks	0

FOPABS:

	Resource	Usage
1	Estimate of Logic utilization (ALMs needed)	0
2		
3	Combinational ALUT usage for logic	0
4		
5	Dedicated logic registers	0
6		
7	I/O pins	102
8		
9	Total DSP Blocks	0

FOPNEG:

	Resource	Usage
1	Estimate of Logic utilization (ALMs needed)	0
2		
3	Combinational ALUT usage for logic	0
4		
5	Dedicated logic registers	0
6		
7	I/O pins	102
8		
9	Total DSP Blocks	0

FOPCEQ :

	Resource	Usage
1	Estimate of Logic utilization (ALMs needed)	30
2		
3	Combinational ALUT usage for logic	33
4		
5	Dedicated logic registers	1
6		
7	I/O pins	102
8		
9	Total DSP Blocks	0

FOPCLT :

	Resource	Usage
1	Estimate of Logic utilization (ALMs needed)	38
2		
3	Combinational ALUT usage for logic	64
4		
5	Dedicated logic registers	1
6		
7	I/O pins	102
8		
9	Total DSP Blocks	0

Estimate of Logic utilization (ALMs needed)

FOPCLE :

	Resource	Usage
1	Estimate of Logic utilization (ALMs needed)	39
2		
3	Combinational ALUT usage for logic	65
4		
5	Dedicated logic registers	1
6		
7	I/O pins	102
8		
9	Total DSP Blocks	0

FOPCVTSW:

	Resource	Usage
1	Estimate of Logic utilization (ALMs needed)	133
2		
3	Combinational ALUT usage for logic	187
4		
5	Dedicated logic registers	236
6		
7	I/O pins	102
8		
9	Total DSP Blocks	0

FOPCVTWS:

	Resource	Usage
1	Estimate of Logic utilization (ALMs needed)	157
2		
3	Combinational ALUT usage for logic	247
4	Combinational ALUT usage for logic	
5	Dedicated logic registers	255
6		
7	I/O pins	102
8		
9	Total DSP Blocks	0

d) Usando o TimeQuest, defina um clock de 50MHz, indique os requerimentos temporais para a ULA total e para cada operação separadamente: i) número de ciclos necessários, ii) caminho de maior atraso, iii) tempos th, tco, tsu e slacks, iv) máxima frequência de clock utilizável, e v) se há algum requerimento não atendido.

RESPOSTA:

IV) Máxima frequência de clock utilizável = 134.01 MHz

ADD/SUB:

Clock to Output Times

	Data Port	Clock Port	Rise	Fall	Clock Edge	Clock Reference
1	onan	clk	9.409	9.700	Rise	clk
2	ooverflow	clk	9.622	9.980	Rise	clk
3	▾ oresult[*]	clk	10.543	11.255	Rise	clk
1	oresult[0]	clk	9.400	9.633	Rise	clk
2	oresult[1]	clk	9.538	9.875	Rise	clk
3	oresult[2]	clk	9.356	9.561	Rise	clk
4	oresult[3]	clk	10.543	11.255	Rise	clk
5	oresult[4]	clk	9.411	9.687	Rise	clk
6	oresult[5]	clk	9.458	9.794	Rise	clk
7	oresult[6]	clk	10.067	10.464	Rise	clk
8	oresult[7]	clk	8.596	8.929	Rise	clk
9	oresult[8]	clk	10.436	10.907	Rise	clk
10	oresult[9]	clk	9.482	9.795	Rise	clk
11	oresult[10]	clk	9.543	9.817	Rise	clk
12	oresult[11]	clk	9.969	10.274	Rise	clk
13	oresult[12]	clk	8.842	9.408	Rise	clk
14	oresult[13]	clk	10.036	10.487	Rise	clk
15	oresult[14]	clk	8.133	8.563	Rise	clk
16	oresult[15]	clk	10.028	10.295	Rise	clk
17	oresult[16]	clk	8.331	8.687	Rise	clk
18	oresult[17]	clk	9.137	9.309	Rise	clk
19	oresult[18]	clk	8.348	8.641	Rise	clk
20	oresult[19]	clk	9.336	9.566	Rise	clk
21	oresult[20]	clk	9.473	9.760	Rise	clk
22	oresult[21]	clk	9.143	9.296	Rise	clk

MUL:

Clock to Output Times

	Data Port	Clock Port	Rise	Fall	Clock Edge	Clock Reference
1	onan	clk	8.238	8.330	Rise	clk
2	ooverflow	clk	9.669	9.973	Rise	clk
3	▾ oresult[*]	clk	10.258	10.741	Rise	clk
1	oresult[0]	clk	9.770	9.962	Rise	clk
2	oresult[1]	clk	9.155	9.287	Rise	clk
3	oresult[2]	clk	7.746	7.834	Rise	clk
4	oresult[3]	clk	7.732	7.775	Rise	clk
5	oresult[4]	clk	9.693	10.035	Rise	clk
6	oresult[5]	clk	9.302	9.515	Rise	clk
7	oresult[6]	clk	8.370	8.737	Rise	clk
8	oresult[7]	clk	9.659	9.802	Rise	clk
9	oresult[8]	clk	9.804	10.072	Rise	clk
10	oresult[9]	clk	7.610	7.671	Rise	clk
11	oresult[10]	clk	7.726	7.962	Rise	clk
12	oresult[11]	clk	10.258	10.741	Rise	clk
13	oresult[12]	clk	9.147	9.306	Rise	clk
14	oresult[13]	clk	8.323	8.575	Rise	clk
15	oresult[14]	clk	7.928	8.245	Rise	clk
16	oresult[15]	clk	8.587	9.049	Rise	clk
17	oresult[16]	clk	8.050	8.471	Rise	clk
18	oresult[17]	clk	9.289	9.465	Rise	clk
19	oresult[18]	clk	7.995	8.266	Rise	clk
20	oresult[19]	clk	10.181	10.466	Rise	clk
21	oresult[20]	clk	9.429	9.638	Rise	clk
22	oresult[21]	clk	9.212	9.351	Rise	clk

DIV:

Clock to Output Times

	Data Port	Clock Port	Rise	Fall	Clock Edge	Clock Reference
1	onan	clk	7.907	8.035	Rise	clk
2	ooverflow	clk	8.085	8.334	Rise	clk
3	oresult[*]	clk	14.189	16.715	Rise	clk
1	oresult[0]	clk	7.880	8.057	Rise	clk
2	oresult[1]	clk	8.127	8.417	Rise	clk
3	oresult[2]	clk	8.080	8.311	Rise	clk
4	oresult[3]	clk	8.280	8.629	Rise	clk
5	oresult[4]	clk	8.384	8.728	Rise	clk
6	oresult[5]	clk	8.212	8.394	Rise	clk
7	oresult[6]	clk	8.255	8.481	Rise	clk
8	oresult[7]	clk	8.014	8.234	Rise	clk
9	oresult[8]	clk	8.257	8.499	Rise	clk
10	oresult[9]	clk	8.427	8.841	Rise	clk
11	oresult[10]	clk	8.048	8.308	Rise	clk
12	oresult[11]	clk	7.636	7.712	Rise	clk
13	oresult[12]	clk	8.201	8.413	Rise	clk
14	oresult[13]	clk	8.313	8.631	Rise	clk
15	oresult[14]	clk	8.026	8.260	Rise	clk
16	oresult[15]	clk	7.931	8.134	Rise	clk
17	oresult[16]	clk	8.374	8.743	Rise	clk
18	oresult[17]	clk	8.380	8.733	Rise	clk
19	oresult[18]	clk	8.072	8.234	Rise	clk
20	oresult[19]	clk	8.075	8.186	Rise	clk
21	oresult[20]	clk	7.632	7.729	Rise	clk
22	oresult[21]	clk	8.132	8.312	Rise	clk

SQRT:

Clock to Output Times

	Data Port	Clock Port	Rise	Fall	Clock Edge	Clock Reference
1	onan	clk	9.779	10.049	Rise	clk
2	ooverflow	clk	9.763	10.022	Rise	clk
3	oresult[*]	clk	10.725	11.103	Rise	clk
1	oresult[0]	clk	7.918	8.021	Rise	clk
2	oresult[1]	clk	10.607	11.024	Rise	clk
3	oresult[2]	clk	8.818	9.405	Rise	clk
4	oresult[3]	clk	8.102	8.237	Rise	clk
5	oresult[4]	clk	8.656	8.882	Rise	clk
6	oresult[5]	clk	8.102	8.252	Rise	clk
7	oresult[6]	clk	10.725	11.103	Rise	clk
8	oresult[7]	clk	8.289	8.502	Rise	clk
9	oresult[8]	clk	8.453	8.736	Rise	clk
10	oresult[9]	clk	8.471	8.711	Rise	clk
11	oresult[10]	clk	8.445	8.739	Rise	clk
12	oresult[11]	clk	9.532	9.739	Rise	clk
13	oresult[12]	clk	8.239	8.462	Rise	clk
14	oresult[13]	clk	9.768	9.878	Rise	clk
15	oresult[14]	clk	8.577	8.828	Rise	clk
16	oresult[15]	clk	8.144	8.273	Rise	clk
17	oresult[16]	clk	8.534	8.815	Rise	clk
18	oresult[17]	clk	8.608	8.827	Rise	clk
19	oresult[18]	clk	8.022	8.324	Rise	clk
20	oresult[19]	clk	8.447	8.670	Rise	clk
21	oresult[20]	clk	9.532	9.695	Rise	clk
22	oresult[21]	clk	8.153	8.316	Rise	clk

C_EQ:

Clock to Output Times

	Data Port	Clock Port	Rise	Fall	Clock Edge	Clock Reference
1	oCompResult	clk	7.013	7.122	Rise	clk