

INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA

Licenciatura em Engenharia Informática e de Computadores



Relatório do 2.º Laboratório de Lógica e Sistemas Digitais

Circuitos Aritméticos

Trabalho realizado por:

Nome: Denis Orlov N° 48250

Nome: Pedro Malafaia N° 49506

Nome: Roberto Petrisoru N° 49418

Docente: Mário Vestias

24 de novembro de 2021

1 Introdução

Este segundo laboratório para avaliação de Lógica e Sistemas Digitais tem como objetivo o desenvolvimento de um circuito aritmético (ALU – Unidade Lógica e Aritmética) em VHDL. Para além disso é também necessário simular o circuito no programa Quartus prime e, por fim, implementar o circuito na placa de desenvolvimento DE10-Lite da Intel.

Este circuito (ALU) realiza várias operações, das quais: soma, subtração, AND, OR, XOR, NOT e soma e subtração com o primeiro elemento dividido por 2.

Ou seja, no fundo, este circuito recebe 2 valores (W e Y) que são os operandos, ambos com 4 bits e, de seguida, com base na entrada OP vai ser seleccionada a operação que se vai fazer com os 2 valores. Como saída este circuito vai apresentar o resultado da operação, bem como o Carry/Borrow, Overflow, Zero, Paridade e Greater or Equal.

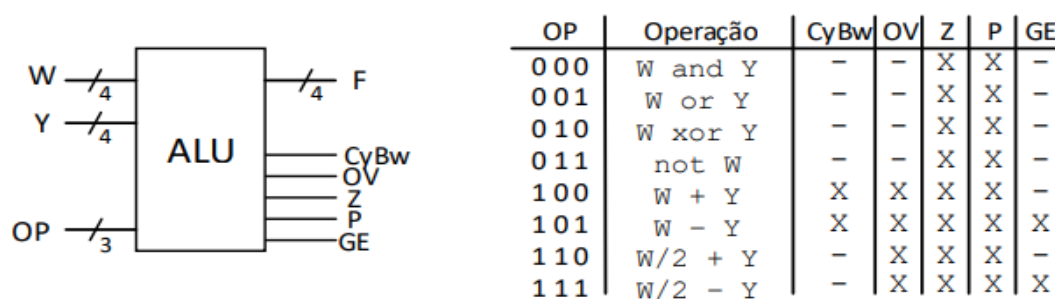


Imagem 1 – Circuito de topo e escolha da operação pelo OP

2 Desenvolvimento do Trabalho

2.1 Descrição das funções lógicas e os respectivos diagramas lógicos

2.1.1 Módulo Lógica

Dentro deste módulo é onde se realizam as operações AND, OR, XOR e NOT. Para tal, este módulo recebe como entrada o W e o Y (dois números com 4 bits), bem como OP com 2 bits que é o seletor do multiplexer que se encontra dentro do circuito lógico. Como saída este módulo vai ter S que corresponde ao resultado da operação lógica. O valor de S é obtido a partir de um multiplexer em que OP vai ser o seletor que vai decidir qual a operação que vai ser apresentada como saída. As funções lógicas que realizam as operações são as seguintes:

O0 = W and Y;

O1 = W or Y;

O2 = W xor Y;

O3 = not W

O0, O1, O2 e O3 foi o nome que demos às entradas no multiplexer. Recorrendo ao seletor OP, o multiplexer irá decidir qual é a operação que vai sair em forma de S.

O circuito lógico tem o seguinte aspeto:

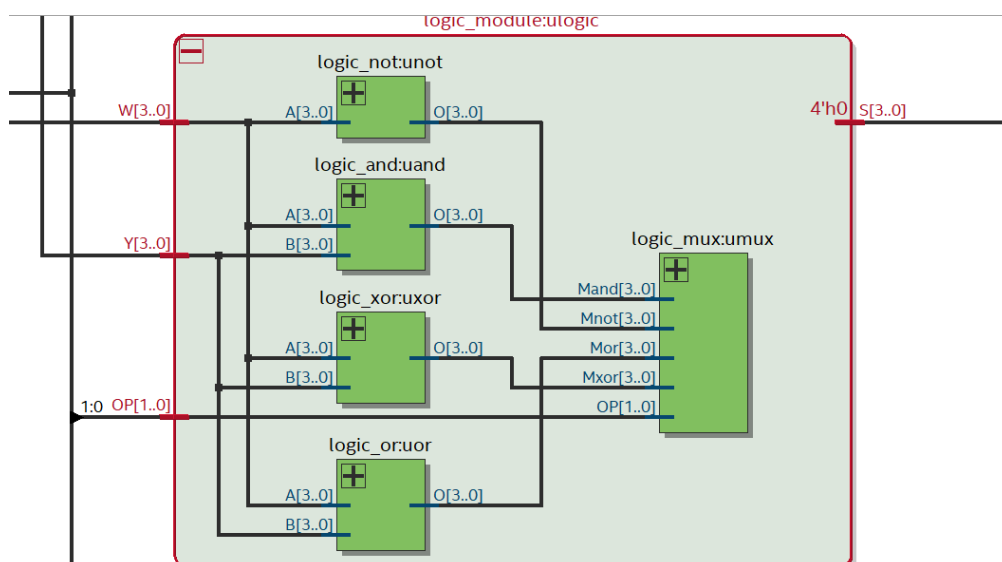


Imagem 2 – Diagrama do módulo lógica (logic_module)

O multiplexer que tem como função utilizando o OP como seletor, escolher e realizar a função desejada, sendo elas AND, OR, XOR, NOT (este exclusivo ao W) tem como entradas o W e Y de 4 bits cada, e OP de 2 bits, e como saída S de 4 bits.

$$S(0) = (OP(0) \text{ and } Mand(0)) \text{ or } (\text{not } OP(0) \text{ and } Mor(0)) \text{ or } (OP(1) \text{ and } Mxor(0)) \text{ or } (\text{not } OP(1) \text{ and } Mnot(0));$$

$$S(1) = (OP(0) \text{ and } Mand(1)) \text{ or } (\text{not } OP(0) \text{ and } Mor(1)) \text{ or } (OP(1) \text{ and } Mxor(1)) \text{ or } (\text{not } OP(1) \text{ and } Mnot(1));$$

$$S(2) = (OP(0) \text{ and } Mand(2)) \text{ or } (\text{not } OP(0) \text{ and } Mor(2)) \text{ or } (OP(1) \text{ and } Mxor(2)) \text{ or } (\text{not } OP(1) \text{ and } Mnot(2));$$

$$S(3) = (OP(0) \text{ and } Mand(3)) \text{ or } (\text{not } OP(0) \text{ and } Mor(3)) \text{ or } (OP(1) \text{ and } Mxor(3)) \text{ or } (\text{not } OP(1) \text{ and } Mnot(3));$$

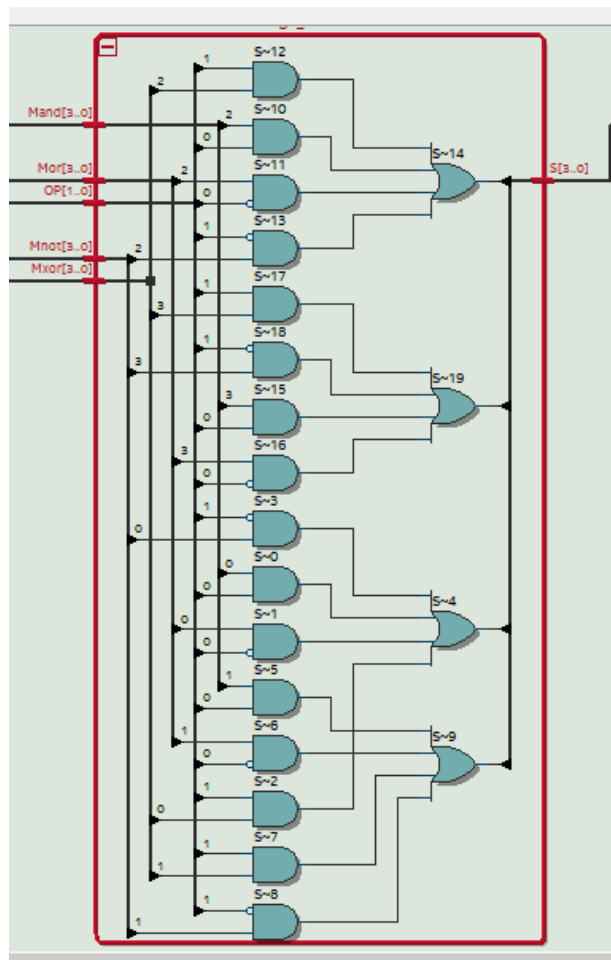


Imagem 3 – Diagrama do multiplexer do modulo de lógica (logic_mux)

2.1.2 W_W/2

Dentro deste módulo é onde ocorre a divisão de W, nos casos em que é necessário a realização da mesma. Para isso, este módulo recebe como entrada W, com 4 bits e OP, com apenas 1 bit. OP vai decidir se a divisão efetuada ou não. Se a divisão não for efetuada, W mantém o valor. Como saída, este módulo tem A que vai corresponder ao valor de W após a divisão, caso tenha ocorrido. As operações lógicas que determinam o valor de A são as seguintes:

$$A(0) = (W(0) \text{ and not}(OP)) \text{ or } (W(1) \text{ and } OP);$$

$$A(1) = (W(1) \text{ and not}(OP)) \text{ or } (W(2) \text{ and } OP);$$

$$A(2) = (W(2) \text{ and not}(OP)) \text{ or } (W(3) \text{ and } OP);$$

$$A(3) = W(3)$$

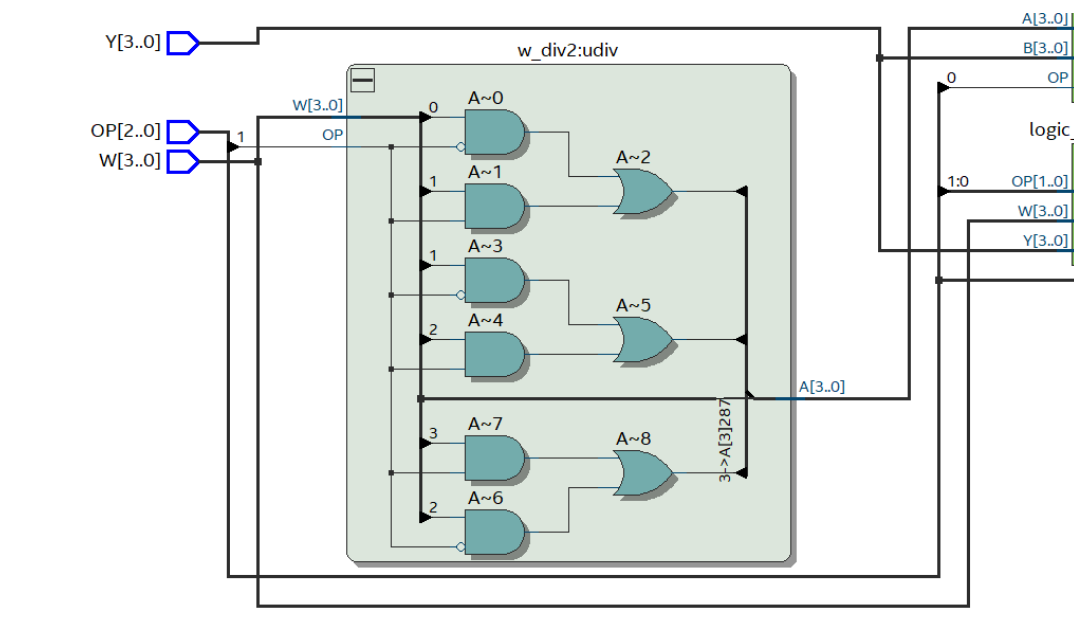


Imagem 4 – Diagrama W_div2

2.1.3 Módulo LAB4 (m_main)

Este modulo vai realizar as seguintes operações: $W + Y$, $W - Y$, $W/2 + Y$ e $W/2 - Y$. Para tal, recebe como entrada A e Y, ambos com 4 bits (A corresponde a W após ter passado pelo módulo da divisão por 2), bem como OP com 1 bit apenas. OP vai novamente ser o seletor que decide que operação vai ser dada como saída. Dentro deste módulo temos ainda dois submódulos: um adder (composto este por 4 submódulos fulladder) que vai realizar a soma ou a subtração, e as flags, que nos vão indicar o Carry/Borrow e o Overflow.

$$S = A \text{ xor } Y \text{ xor } \text{Cin};$$

$$\text{Cout} = (A \text{ and } Y) \text{ or } (\text{Cin and } (A \text{ xor } Y));$$

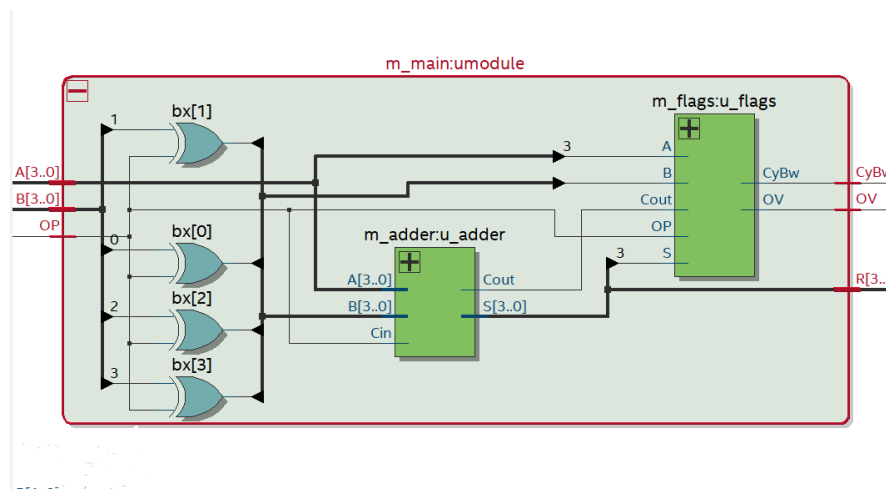


Imagem 5 – Diagrama m_main

Cada fulladder tem como entradas A, B e Cin(CarryIn) com 1 bit apenas e como saída terá S, Cout(CarryOut) com 1 bit também.

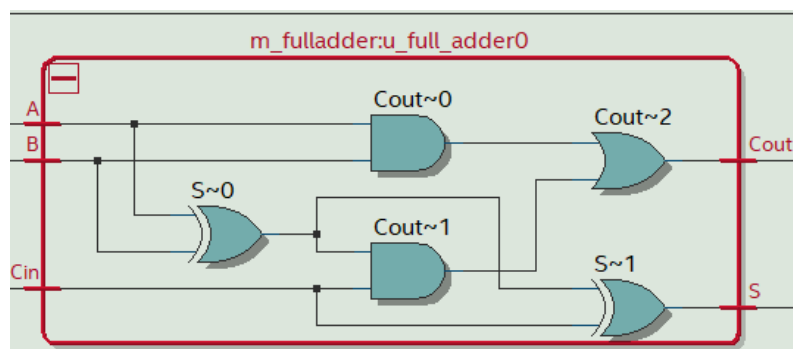


Imagem 6 – Diagrama m_fulladder

O adder é composto por 4 fulladders, sendo o Cin de cada um o Cout do anterior.

$A \Rightarrow A(x)$,

$Y \Rightarrow Y(x)$,

$Cin \Rightarrow Cin$,

$S \Rightarrow S(x)$,

$Cout \Rightarrow c(x+1)$

$x=\{0,1,2,3\}$

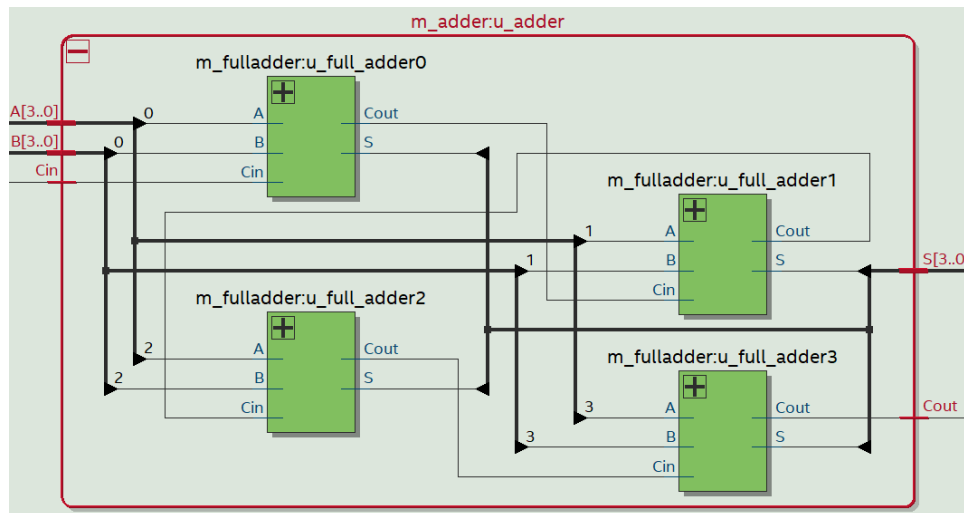


Imagem 7 – Diagrama m_adder

As flags têm como entrada S, A e Y, com 4 bits, e OP e CarryOut, ambas com 1 bit, e CyBw e OV como saída ambas com 1 bit.

$CyBw \Leftarrow Cout \text{ xor } OP$;

$OV \Leftarrow (\text{not}(A) \text{ and } \text{not}(Y) \text{ and } S) \text{ or } (A \text{ and } Y \text{ and } \text{not}(S))$;

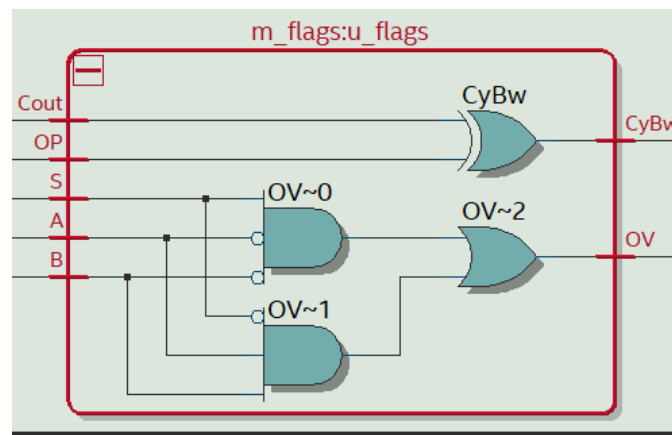


Imagem 8 – Diagrama m_flags

2.1.4 MUX

O módulo MUX da ALU tem como função, utilizando o 3o bit de OP, escolher qual dos módulos entre LAB4 (m_main) e o modulo lógico (logic_mux) utilizar no momento.

O mesmo conta com 2 entradas R e S de 4 bits provenientes de cada um dos módulos citados anteriormente, e o sel (3º bit de OP) de 1 bit. E com apenas 1 saída F de 4 bits com o resultado do multiplexer.

$$F(0) = (\text{sel and } R(0)) \text{ or } (\text{not sel and } S(0));$$

$$F(1) = (\text{sel and } R(1)) \text{ or } (\text{not sel and } S(1));$$

$$F(2) = (\text{sel and } R(2)) \text{ or } (\text{not sel and } S(2));$$

$$F(3) = (\text{sel and } R(3)) \text{ or } (\text{not sel and } S(3));$$

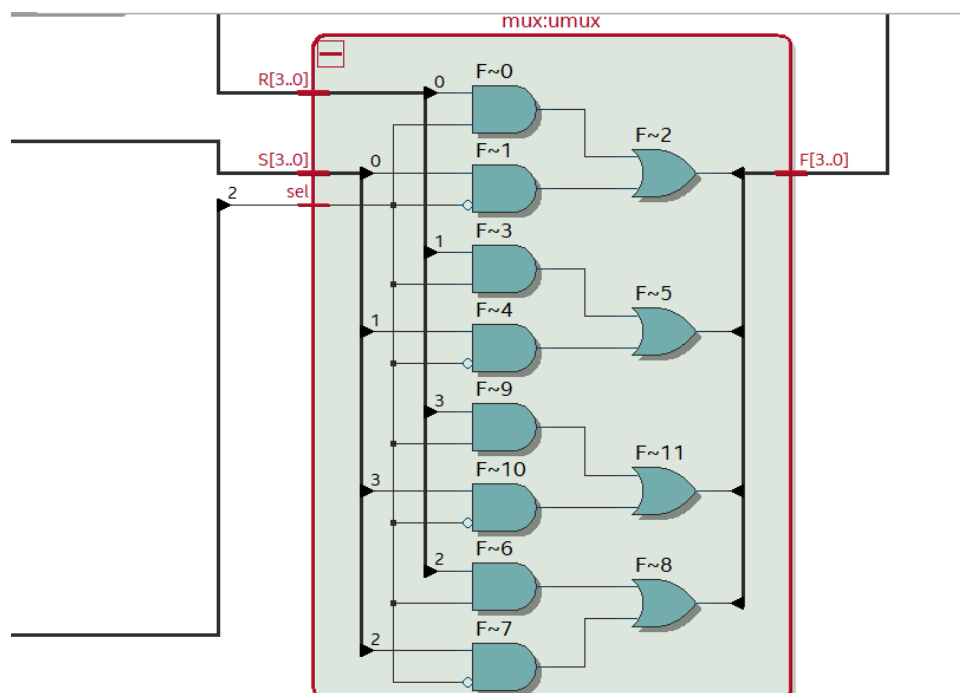


Imagem 9 – Diagrama do multiplexer (mux)

2.1.5 Flags

O módulo das flags tem como função receber o resultado do módulo MUX, sendo a entrada do mesmo denominada de F de 4 bits, e o A e B (sendo o A o CyBw e B o OV do módulo LAB4) ambos de 1 bit, e como saída o CyBw, OV, Z, P e GE todos de 1 bit só.

$CyBw = A;$

$OV = B;$

$Z = (\text{not}(F(3)) \text{ or } F(2) \text{ or } F(1) \text{ or } F(0));$

$P = (F(0) \text{ xor } F(1) \text{ xor } F(2) \text{ xor } F(3));$

$GE = B \text{ xnor } F(3);$

O CyBW (Carry Borrow) e o OV (Overflow) são adquiridos diretamente do módulo LAB4, sendo o CyBw ativa quando o resultado excede o domínio dos números naturais, e OV ativa quando excede o domínio de números relativos.

O Z (Zero), ou seja verifica se o valor é nulo ou não.

O P (Paridade), verifica se o valor é par ou não. O GE (Greater or Equal), verifica se o primeiro operando é maior do que o segundo, considerando-se apenas na representação de números relativos.

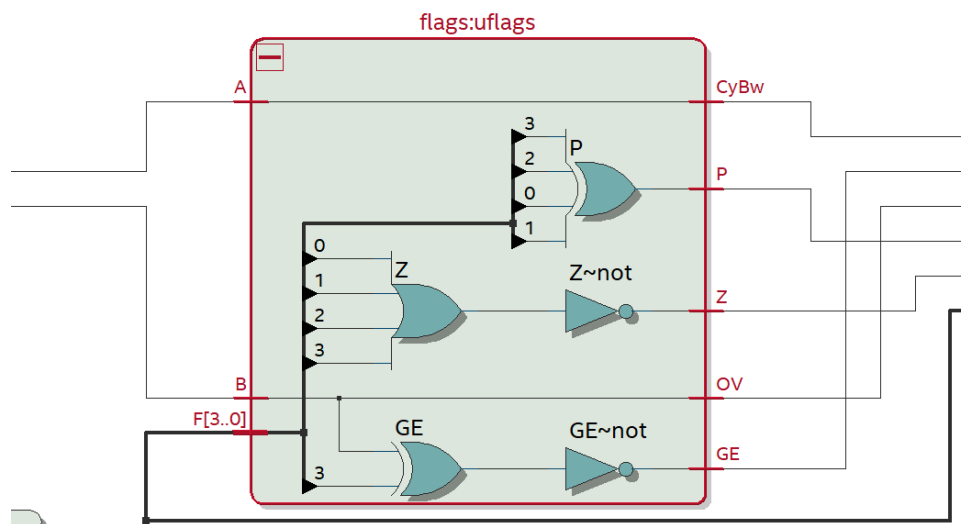


Imagem 10 – Diagrama flags

3 Montagem e Teste Laboratorial

Após termos todos os módulos necessários, passámos à montagem da entidade de topo onde ligámos todos os módulos uns aos outros de forma a obtermos um circuito funcional e com a finalidade de ser implementado e testado na placa de desenvolvimento DE-Lite da Intel. A entidade de topo recebe como entrada W e Y, ambos com 4 bits e OP com 3 bits. De saída, a entidade de topo apresenta o F com 4 bits, que corresponde ao resultado da operação que foi realizada, bem como todas as flags: Carry/Borrow, Greater or Equal, Overflow, Parity e Zeros, todas estas com apenas 1 bit. O circuito completo ficou com o seguinte aspeto:

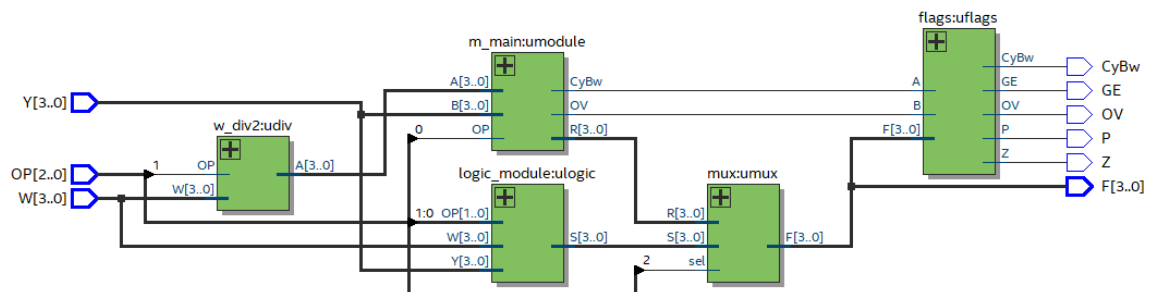


Imagem 10 – Entidade de topo ALU

De seguida, utilizando a ferramenta de simulação do programa Quartus Prime verificamos os valores que obtemos o que nos permitiu preencher a seguinte tabela de resultados:

							Resultado Teórico								Resultado Experimental							
OP	W(2)	W(N)	W(Z)	Y(2)	Y(N)	Y(Z)	F(2)	F(N)	F(Z)	CyBw	OV	Z	P	GE	F(2)	F(N)	F(Z)	CyBw	OV	Z	P	GE
1 0 0	1010	10	12	0101	5	5	1111	15	17	0	0	0	0	0	1111	15	17	0	0	0	0	0
1 0 0	1010	10	12	1101	13	15	0111	7	7	1	1	0	1	0	0111	7	7	1	1	0	1	0
1 0 0	0110	6	6	0101	5	5	1011	11	13	0	1	0	1	1	1011	11	13	0	1	0	1	1
1 0 0	1010	10	12	1010	10	12	0100	4	4	1	1	0	1	0	0100	4	4	1	1	0	1	0
1 0 1	1010	10	12	0101	5	5	0101	5	5	0	1	0	0	0	0101	5	5	0	1	0	0	0
1 0 1	1010	10	12	1101	13	15	1101	13	15	1	0	0	1	0	1101	13	15	1	0	0	1	0
1 0 1	0110	6	6	0101	5	5	0001	1	1	0	0	0	1	1	0001	1	1	0	0	0	1	1
1 0 1	1010	10	12	1010	10	12	0000	0	0	0	0	1	0	1	0000	0	0	0	0	1	0	1
1 1 0	1010	10	12	0101	5	5	0010	2	2	1	0	0	1	1	0010	2	2	1	0	0	1	1
1 1 0	1010	10	12	1101	13	15	1010	10	12	1	0	0	0	0	1010	10	12	1	0	0	0	0
1 1 0	0110	6	6	0101	5	5	1000	8	8	0	1	0	1	1	1000	8	8	0	1	0	1	1
1 1 0	1010	10	12	1010	10	12	0111	7	7	1	1	0	1	0	0111	7	7	1	1	0	1	0
1 1 1	1010	10	12	0101	5	5	1000	8	8	0	0	0	1	0	1000	8	8	0	0	0	1	0

1 1 1	1010	10	12	1101	13	15	0000	0	0	0	0	1	0	1	0000	0	0	0	0	1	0	1
1 1 1	0110	6	6	0101	5	5	1110	14	16	1	0	0	1	0	1110	14	16	1	0	0	1	0
1 1 1	1010	10	12	1010	10	12	0011	3	3	0	0	0	0	1	0011	3	3	0	0	0	0	1
0 0 0	1011	13	15	1101	13	15	1111	15	17	1	0	0	0	0	1111	15	17	1	0	0	0	0
0 0 1	1011	11	13	1101	13	15	1101	13	15	1	0	0	1	0	1101	13	15	1	0	0	1	0
0 1 0	1011	11	13	1101	13	15	1111	15	17	1	0	0	0	0	1111	15	17	1	0	0	0	0
0 1 1	1011	11	13	1101	13	15	1111	15	17	0	0	0	0	0	1111	15	17	0	0	0	0	0

Tabela 1 – Resultados Teóricos e Resultados Experimentais

4 Conclusão

Concluindo o circuito aritmético ALU serviu como uma base para um aprofundamento e exploração de circuitos aritméticos utilizando VHDL. Através do mesmo conseguiu-se implementar e aplicar diferentes conceitos sobre aritmética, como o somador binário e a geração de diferentes flags resultado desse somador para além de também simular o circuito no programa Quartus prime e, por fim, implementar o circuito na placa de desenvolvimento DE10-Lite da Intel, completando assim a tabela utilizada para anotar os resultados teóricos e confirmar os mesmos com os resultados práticos obtidos.