ISEL
Departamento de Engenharia
Eletrónica e Telecomunicações
e de Computadores

Licenciatura em Engenharia Informática e de Computadores
e
Licenciatura em Engenharia Informática, Redes e Telecomunicações

Circuitos aritméticos e lógicos
(*2º Trabalho de Laboratório*)

Lógica e Sistemas Digitais
2024 / 2025 inverno

10 de outubro de 2024

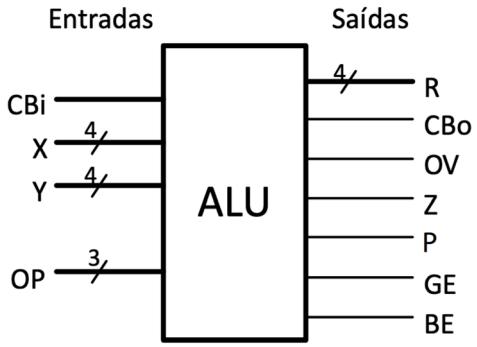
1 Objetivo

O objetivo deste trabalho é descrever um circuito aritmético e lógico (ALU – *Aritmetic and Logic Unit*) com base em *VHDL* estrutural, simulá-lo e implementá-lo na placa de desenvolvimento *DE10-Lite* da *Intel*. Este trabalho é obrigatório e contabilizado para a classificação prática.

2 Descrição do circuito a desenvolver

Pretende-se projetar uma unidade aritmética e lógica que realize as operações aritméticas adição ($W + Y + CBi$), subtração ($W - Y - CBi$), incremento ($W + CBi$) e decremento ($W - CBi$) e as operações de deslocamento ($W >>> 1$, $W >> 1$ e $W <<< 1$, e a operação lógica NAND ($\overline{W.Y}$), sobre operandos de 4 bits. O resultado tem 4 bits e deve gerar os indicadores (*flags*) *Carry/Borrow* (CBo), *Overflow* (OV), *Zero* (Z), *Greater or Equal* (GE), *Below or Equal* (BE) e *Parity* (P).

As entradas e saídas do sistema, bem como as operações, estão representadas na Figura 1.



OP	Operação	CBo	OV	Z	P	GE	BE
0 0 0	$X + CBi$	X	X	X	X	–	–
0 0 1	$X - CBi$	X	X	X	X	–	–
0 1 0	$X + Y + CBi$	X	X	X	X	–	–
0 1 1	$X - Y - CBi$	X	X	X	X	X	X
1 0 0	$X >>> 1$	X	–	X	X	–	–
1 0 1	$X >> 1$	X	–	X	X	–	–
1 1 0	$X <<< 1$	X	–	X	X	–	–
1 1 1	$\overline{X.Y}$	–	–	X	X	–	–

Figura 1 – Especificação da ALU a desenvolver

As entradas X e Y são os operandos de 4 bits e a entrada OP de 3 bits seleciona a operação a realizar. Para as operações aritméticas, considere que os operandos estão representados em números naturais ou relativos (inteiros sem e com sinal, respetivamente). A saída R também de 4 bits é o resultado da operação, no mesmo domínio dos operandos. As operações de deslocamento >>> (*Logical Shift Right* – LSR), >> (*Aritmetic Shift Right* – ASR) e <<< (*Logical Shift Left* – LSL) deslocam para a direita (LSR e ASR) ou para a esquerda (LSL) o valor do operando X em 1 bit na respetiva direção. As operações LSR e LSL introduzem zeros no resultado R enquanto a operação ASR introduz 1 bit com o valor do bit de sinal do operando X.

Adicionalmente, são geradas ainda as seis *flags*:

- CBo: Representa o *carry* de saída da operação de soma ou o *borrow* de saída da operação de subtração. Fica ativa quando o resultado excede o domínio dos números naturais; representa, igualmente, no âmbito das operações deslocamento (LSR, ASR e LSL), o valor do bit deslocado de X;

- OV: Fica ativa quando o resultado excede o domínio dos números relativos;
- Z: Fica ativa quando o resultado é igual a zero;
- P: Fica ativa quando o resultado tem um número ímpar de 1's;
- GE: Fica ativa quando o primeiro operando (X) é maior ou igual do que o segundo (Y + CBi), considerando-se apenas na representação de números relativos;
- BE: Fica ativa quando o primeiro operando (X) é menor ou igual do que o segundo (Y + CBi), considerando-se apenas na representação de números naturais.

Em algumas operações, o valor das *flags* não tem significado, representado na tabela da Figura 1 com o carater '−'. Nesses casos, o valor das *flags* pode assumir qualquer valor.

3 Projeto do Circuito

O circuito deverá ser implementado de acordo com o diagrama de blocos da Figura 2.

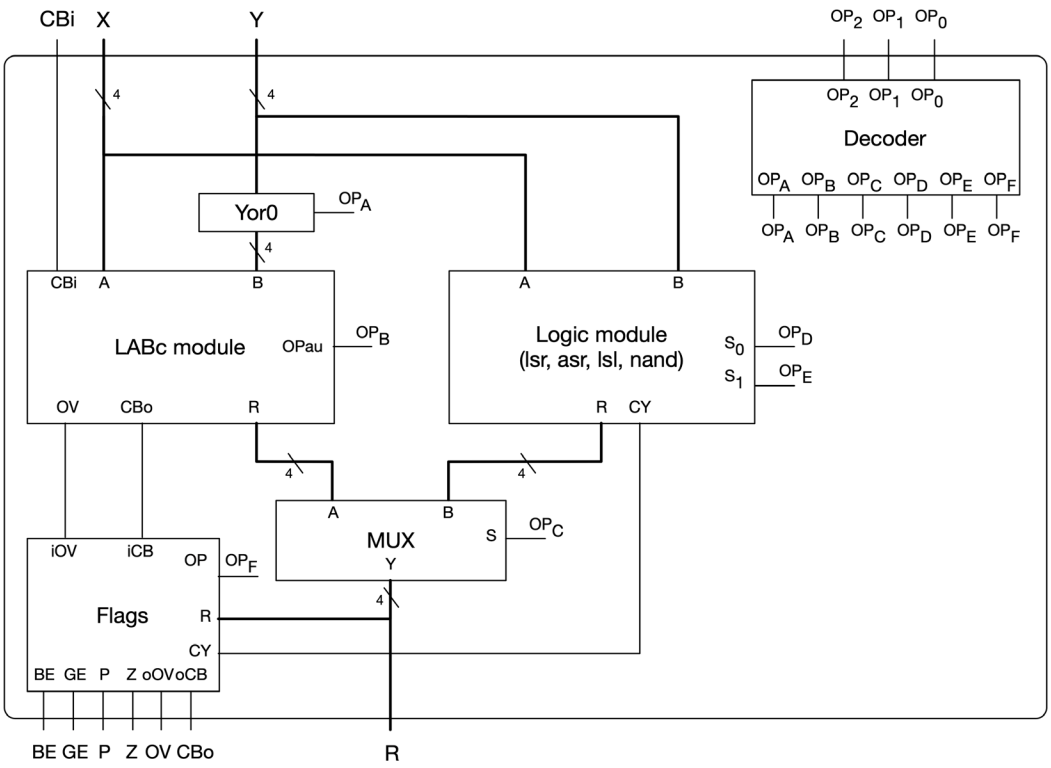


Figura 2 – Diagrama de blocos da ALU

Para o projeto da ALU deverá elaborar os seguintes passos:

1. Considere o módulo aritmético desenvolvido no laboratório LABc (*LABc module*);
2. Desenvolva e descreva em *VHDL* o módulo de lógica com base no módulo desenvolvido no LABb;
3. Desenvolva e descreva em *VHDL* o módulo *Flags* com base nas *flags* geradas pelo módulo LABc, no resultado R e na *flag* CY gerada pelo módulo de lógica para gerar as seis *flags* do circuito ALU;

4. Desenvolva o módulo Decoder em VHDL. Estabeleça a correspondência entre os sinais internos OP_A ... OP_F e os 3 bits de entrada $OP_{2..0}$. Note que vários sinais OP_A ... OP_F podem ligar ao mesmo bit de $OP_{2..0}$;
5. Reúna as unidades referidas nos pontos anteriores e a unidade MUX numa entidade de topo com o nome ALU que corresponde à descrição completa da ALU;
6. Simule o circuito (considere o ficheiro de teste anexo ao trabalho);
7. Implemente o circuito na placa DE10-Lite considerando a utilização dos 10 interruptores ($SW_{9..0}$) para definição dos valores de entrada ($X_{3..0} = SW_{3..0}$, $Y_{3..0} = SW_{7..4}$). A operação é definida na placa pelos dois $SW_{9..8}$ para definição dos bits $OP_{1..0}$ e o botão de pressão 0 (BTN_0) é usado para definição do bit mais significativo de OP ($OP_2 = BTN_0$); O sinal Carry-In (CB_i) é definido pelo estado do botão de pressão 1 (CBi = BTN_1).
8. Defina 4 combinações de entrada que servirão para testar o circuito implementado (duas para testar operações aritméticas – #21 e #22 – e duas para testar operações lógicas – #23 e #24).
9. Determine para cada uma das combinações de entrada na tabela do anexo A qual o resultado esperado (resultado teórico).
10. Confirme o funcionamento do circuito para as combinações de entrada na tabela do anexo A registando o valor obtido no circuito (resultado experimental) e comparando com o resultado esperado (resultado teórico).

4 Relatório

Deverá apresentar um relatório do trabalho desenvolvido com a seguinte estrutura:

1. Capa com a indicação do curso, unidade curricular, elementos do grupo (número e nome), nome do trabalho;
2. Introdução: breve descrição do trabalho a desenvolver e quais os objetivos;
3. Análise e Projeto: descrição de todas as funções lógicas e diagramas lógicos;
4. Montagem laboratorial: Resultados experimentais e confirmação dos resultados teóricos;
5. Conclusão: comentário sobre o trabalho desenvolvido e sobre os resultados obtidos;
6. Anexo: Código *VHDL*.

									Resultado Teórico									**Resultado Experimental**								
#	**OP**	**CB_i**	**$X_{(2)}$**	**$X_{(N)}$**	**$X_{(Z)}$**	**$Y_{(2)}$**	**$Y_{(N)}$**	**$Y_{(Z)}$**	**$R_{(2)}$**	**$R_{(N)}$**	**$R_{(Z)}$**	**CB_o**	**OV**	**Z**	**P**	**GE**	**BE**	**$R_{(2)}$**	**$R_{(N)}$**	**$R_{(Z)}$**	**CB_o**	**OV**	**Z**	**P**	**GE**	**BE**
1	000	0	0000			1111																				
2	000	1	0000			1111																				
3	000	1				0101			1000																	
4	000	1	1111			1010																				
5	001	1	0000			1111																				
6	001	1	0001			0101																				
7	001	1				1010				7																
8	010	0				1110					-1															
9	010	1	0001											1												
10	010	1	1000						1000																	
11	010	1	0010			0101																				
12	011	0	0011			0101																				
13	011	0	1110							2																
14	011	1	0110			0110																				
15	011	1				0111										1	1									
16	100	1	0001			1100																				
17	100	0	1101			1001																				
18	101	0	1010			1111																				
19	110	0	1001			0110																				
20	111	1	1111							11																
21																										
22																										
23																										
24																										