

# LSDI 2015/16 - Trabalho laboratorial 2

## Projeto de circuitos somadores

### 1 - Introdução

Este trabalho laboratorial tem por objetivos desenvolver circuitos combinacionais simples e explorar uma ferramenta computacional para projeto de sistemas digitais que permite o desenvolvimento, implementação e ensaio de circuitos digitais. Para tal, serão usadas técnicas de obtenção de circuitos lógicos a partir da descrição das suas funções e a linguagem de descrição de *hardware* Verilog. Como ambiente de desenvolvimento, será usado o *software* de projeto ISE (*Integrated Synthesis Environment*) da Xilinx. O ensaio dos circuitos far-se-á na placa de desenvolvimento S3Board ([www.digilentinc.com/Products/Detail.cfm?Prod=S3BOARD](http://www.digilentinc.com/Products/Detail.cfm?Prod=S3BOARD)), a qual inclui um dispositivo reconfigurável FPGA (*Field Programmable Gate Array*) com capacidade equivalente a 200000 portas lógicas. O *software* ISE, utilizado para o desenvolvimento deste e dos próximos trabalhos práticos da unidade curricular de Laboratório de Sistemas Digitais, é disponibilizado gratuitamente pelo fabricante do circuito FPGA, Xilinx ([www.xilinx.com](http://www.xilinx.com)). A versão instalada nos laboratórios (ISE 10.1) pode ser obtida em <http://www.xilinx.com/support/download/index.html/content/xilinx/en/downloadNav/design-tools/archive.html>.

Para simplificar a construção e ensaio dos circuitos em laboratório, é fornecido um arquivo com o projeto ISE parcialmente construído, LSDI2015\_lab2, que já inclui os elementos necessários para implementar interfaces com os dispositivos de entrada e saída de que a placa S3Board dispõe (LEDs, visualizadores de 7 segmentos, botões de pressão e interruptores). Como apoio à utilização do ISE estão disponíveis vídeos que mostram as principais etapas de projeto. Estes conteúdos estão disponíveis na pasta da unidade curricular no Sigarra

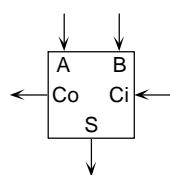
O trabalho é composto por duas partes: projeto e implementação. A componente de projeto deverá ser realizada previamente pelos estudantes como trabalho de casa. Nas duas aulas que este trabalho ocupa serão implementados e ensaiados os circuitos projetados, de acordo com os passos apresentados neste guião.

A preparação, como trabalho de casa, é fundamental para o cumprimento dos objetivos dos trabalhos laboratoriais e para a avaliação a realizar posteriormente na forma de fichas laboratoriais.

### 2 - Projeto

#### 2.1- Projeto de um circuito somador completo

Um somador completo (*full-adder*) é um circuito com três entradas e duas saídas que realiza a operação aritmética adição entre 3 números de 1 bit (cada um só pode ser 0 ou 1), produzindo na saída 2 bits que representam o valor da soma desses 3 bits (podendo ser 0, 1, 2 ou 3). As três entradas podem designar-se por A, B e Ci, e as duas saídas por Co e S, representando Co o bit mais significativo do resultado. A figura 1 mostra o símbolo deste circuito e a respetiva tabela de verdade.



A	B	Ci	Co	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Figura 1 - Somador completo: símbolo e tabela de verdade.

- Recorrendo a representações em mapas de Karnaugh, obtenha expressões minimizadas na forma soma de produtos ou produto de somas que realizem as funções  $Co(A,B,Ci)$  e  $S(A,B,Ci)$ .
- Desenhe os circuitos lógicos *sum* e *carryout* que realizam, respetivamente, as funções  $S(A,B,Ci)$  e  $Co(A,B,Ci)$  encontradas. Para tal, utilize portas lógicas *not*, *and* e *or*.
- Construa agora um circuito, *fulladder*, que implemente o somador completo recorrendo aos símbolos dos circuitos *sum* e *carryout* (figura 2), e respetivas interligações. Mantenha as entradas e as saídas do somador completo com os nomes A, B, Ci e Co, S, respetivamente.

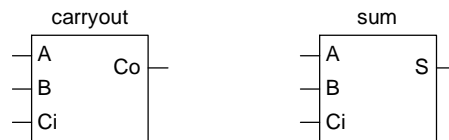


Figura 2 - Símbolos dos circuitos que realizam as funções  $Co(A,B,Ci)$  e  $S(A,B,Ci)$ .

## 2.2 - Projeto de um circuito somador/subtrator para números de 4 bits

O circuito desenvolvido nas alíneas anteriores é um componente fundamental para a construção de circuitos aritméticos. O circuito mostrado na figura 3 implementa um somador de dois números de  $n$  bits, obtido por interligação de  $n$  blocos do tipo somador completo. O operando A é representado pelos bits  $A_{n-1}, \dots, A_1$  e  $A_0$ , o operando B pelos bits  $B_{n-1}, \dots, B_1$  e  $B_0$ , e o resultado S é formado pelos bits  $S_{n-1}, \dots, S_1$  e  $S_0$ . Este circuito é designado por *ripple-carry adder* e produz o resultado da adição de forma semelhante à que é seguida quando se efetua a operação manualmente: a saída Co de um *full-adder* representa o bit de transporte para o andar somador seguinte, sendo recebido por este na entrada Ci. Note que a entrada Ci do andar menos significativo deverá ser ligada a zero e que a saída Co mais significativa representa o transporte resultante da soma dos bits mais significativos de A e B.

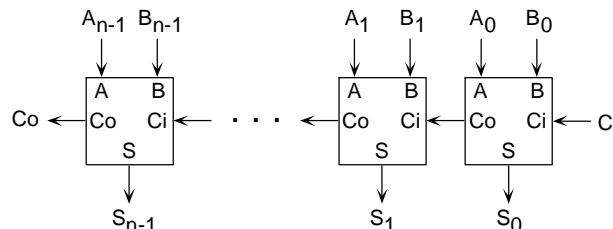


Figura 3 - Somador de 2 números de  $n$  bits utilizando  $n$  blocos do tipo *full-adder*.

- Com base na figura 3, desenhe o circuito somador de 4 bits, *adder4b*, por interligação de 4 circuitos somadores completos. Considere as entradas A3, A2, A1, A0, B3, B2, B1, B0 e Ci, e as saídas S3, S2, S1, S0 e Co, mostradas no símbolo apresentado na figura 4.

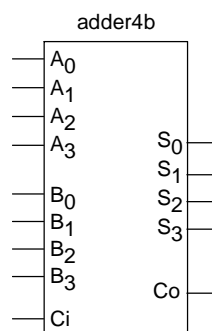


Figura 4 - Símbolo do somador de 4 bits.

- A operação de subtração  $A - B$  pode ser realizada como  $A + (-B)$ , onde  $(-B)$  é o simétrico de B em complemento para 2. Com base no circuito *adder4b* desenvolvido na alínea anterior, é possível realizar

adições ou subtrações, consoante o valor lógico que é colocado numa entrada adicional `add_sub` que permitirá escolher a operação a realizar. Se `add_sub` for 0, as saídas deverão apresentar o resultado da adição,  $A+B$ ; se `add_sub` for 1, o valor apresentado nas saídas deverá ser  $A-B$ , isto é,  $A+(-B)$ .

Comece por desenhar este circuito (`addsub4b`) capaz de realizar adições e subtrações de 4 bits, usando o somador de 4 bits representado na figura 4 e as portas lógicas adicionais que entenda serem necessárias. Seguidamente, escreva um módulo Verilog que implemente este somador/subtrator de 4 bits, por interligação das portas lógicas e do bloco `adder4b` obtido em a). Caso utilize portas lógicas XOR, considere que possuem um tempo de propagação de 9 ns. O módulo deverá ter a seguinte interface:

```
module addsub4b(add_sub, A3, A2, A1, A0, B3, B2, B1, B0, Co, R3, R2, R1, R0);
    input add_sub; // Se add_sub=0 faz A+B; se add_sub=1 faz A-B
    input A3, A2, A1, A0; // Operando A
    input B3, B2, B1, B0; // Operando B
    output Co;
    output R3, R2, R1, R0; // Resultado R
endmodule
```

### 3 - Implementação e ensaio

Após o projeto dos circuitos somadores, resultante da descrição feita na secção anterior, esta secção descreve a respetiva implementação e ensaio a realizar nas aulas laboratoriais. Para tal, irá utilizar o *software* ISE da Xilinx, para desenvolver os circuitos e implementá-los num dispositivo reconfigurável contido na placa S3board, podendo assim ensaiá-los.

Para simplificar o processo de desenvolvimento é fornecido o projeto `LSDI2015_Lab2` do ISE com os circuitos e o módulo Verilog que projetou na primeira parte do trabalho parcialmente construídos: relativamente aos circuitos estão desenhadas as entradas e as saídas a utilizar; para o módulo Verilog é fornecida a interface. Os ficheiros com estes conteúdos deverão ser editados e completados com os respetivos circuitos, de acordo com o projeto realizado. A tabela 1 identifica estes conteúdos. No projeto estão também incluídos outros módulos, que fazem a interface com a placa S3board, não devendo alterá-los.

Ficheiro	Módulo ou circuito implementado	Usado por
<code>sum.sch</code>	saída S do somador completo	<code>fulladder</code>
<code>carryout.sch</code>	saída Co do somador completo	<code>fulladder</code>
<code>fulladder.sch</code>	somador completo	<code>adder4b</code>
<code>Fulladder_tb.v</code>	simulação do somador completo	-
<code>adder4b.sch</code>	somador de 4 bits	<code>adder4b_tb</code> e <code>addsub4b</code>
<code>adder4b_tb.v</code>	simulação do somador de 4 bits	-
<code>addsub4b.v</code>	somador/subtrator de 4 bits	<code>addsub4b_tb</code> e <code>s3board_toplevel</code>
<code>addsub4b_tb.v</code>	simulação do somador/subtrator de 4 bits	-

Tabela 1 - Conteúdo do projeto ISE `LSDI2014_Lab2`.

Abra o projeto `LSDI2015_Lab2` no ISE fazendo duplo clique no ficheiro `LSDI2015_Lab2.ise` (ou, alternativamente, após iniciar o ISE execute a opção “*Open project ...*” e escolha `LSDI2015_Lab2.ise`).

Execute os passos a seguir descritos e responda às questões colocadas.

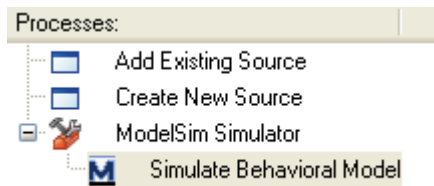
#### 3.1 - Implementação e simulação do somador completo

Edite os ficheiros `sum.sch`, `carryout.sch` e `fulladder.sch`, e complete-os com a implementação dos respetivos circuitos. Para tal, na janela “*Sources*”, faça um duplo clique sobre o nome de cada ficheiro a abrir. Na edição dos vários circuitos, utilize as portas lógicas NOT, AND e OR disponíveis em “*Categories, <C:/ ... /LSDI2015\_lab2>*”. O atraso de propagação destas portas lógicas é 2, 4 e 3 ns, respetivamente. Note que não tem disponíveis portas AND e OR com três entradas.

Para verificar se o somador completo funciona corretamente deve proceder à sua simulação. A simulação consiste na aplicação de valores às entradas do circuito e observação das correspondentes saídas. Comparando as saídas com os valores esperados verifica-se se o circuito apresenta o comportamento desejado. O ficheiro

fulladder\_tb.v fornecido, já completo, descreve uma sequência de valores a aplicar às entradas A, B e  $C_i$  do somador completo, as quais determinam os valores das saídas  $C_o$  e S.

Proceda então à simulação do somador completo: selecione o ficheiro fulladder\_tb.v e, na janela “Processes” (visualizada na figura 5), invoque o simulador ModelSim fazendo duplo clique em “Simulate Behavioral Model”.



**Figura 5** - Conteúdo da janela “Processes” para a simulação.

Expanda a janela de visualização das formas de onda (“Wave”) referentes às entradas e saídas do circuito e verifique se as saídas apresentam os valores esperados. Observe e analise os tempos de propagação com que as saídas vêm afetadas, não esquecendo que os atrasos considerados atrás se referem à versão de duas entradas do AND e do OR.

**Questão 1:** Considere  $t=214$  ns. Que valores apresentam  $C_o$  e S? Estes valores são os esperados quando  $A=0$ ,  $B=0$  e  $C_i=1$ ? E em  $t=202$  ns?

**Questão 2:** Qual o tempo de propagação máximo verificado?

### 3.2 - Implementação e simulação do somador de 4 bits

Edite o ficheiro adder4b.sch e complete-o com o circuito que obteve.

O ficheiro adder4b\_tb.v permite efetuar uma simulação lógica do somador de 4 bits. Nele é descrita uma sequência de valores a aplicar às entradas  $A=A_3A_2A_1A_0$ ,  $B=B_3B_2B_1B_0$  e  $C_i$  do somador, as quais determinam os valores das saídas  $C_o$  e  $S=S_3S_2S_1S_0$ . Selecione o ficheiro adder4b\_tb.v e invoque o simulador fazendo duplo clique em “Simulate Behavioral Model”. Expanda a janela de visualização das formas de onda referentes às entradas e saídas do circuito somador e verifique, atentamente, se as saídas apresentam os valores esperados. Observe e analise os tempos de propagação com que as saídas vêm afetadas.

**Questão 3:** Quando  $A_3A_2A_1A_0=0010$  e  $B_3B_2B_1B_0=0011$  a soma é  $S_3S_2S_1S_0=0101$ . Relativamente aos bits  $S_2$  e  $S_1$ , verifique qual deles surge primeiro e explique porquê?

**Questão 4:** Relativamente à operação  $1111+0000$  com  $C_i=1$ , que valores observa em  $C_o$  e  $S=S_3S_2S_1S_0$ ?

### 3.3 - Implementação e simulação do somador/subtrator de 4 bits

Relativamente ao somador/subtrator de 4 bits proceda de forma semelhante à praticada com os módulos anteriores, tendo em consideração que este módulo vai ser descrito em Verilog. Após ter completado a edição do código Verilog deste módulo, addsub4b(), selecione o respetivo ficheiro de simulação (addsub4b\_tb.v) e execute a simulação nele descrita invocando o simulador.

Analise atentamente os resultados da simulação observando as respetivas formas de onda, verificando se os resultados correspondem aos esperados.

**Questão 5:** Por que razão se usou um intervalo de tempo de 100 ns entre atribuições consecutivas de valores às entradas e não de, por exemplo, 10 ns?

### 3.4 - Ensaio do somador/subtrator de 4 bits

Implemente e experimente o circuito somador/subtrator de 4 bits, programando a placa S3board. Na lista de itens do projeto, o ficheiro s3board\_toplevel.v usa o módulo somador/subtrator de 4 bits, assim como outros módulos, para interface com os interruptores e visualizadores de 7 segmentos, permitindo a sua implementação na placa. Selecione s3board\_toplevel.v e execute o comando “Generate Programming File”, a partir da janela “Processes”, que permite obter o ficheiro de configuração do dispositivo reconfigurável contido na placa. A seguir execute “Configure Target Device”, o que permite transferir o ficheiro de configuração gerado, do

computador para a placa, e programar o dispositivo FPGA com o circuito projetado. Realize estas operações de acordo com as indicações dadas pelo docente.

A figura 6 identifica os recursos da placa usados pelo circuito. O operando A é definido pelo estado dos interruptores SW7 a SW4, isto é, os quatro da esquerda, e o operando B é definido por SW3 a SW0. O valor dos operandos A e B é apresentado em hexadecimal nos visualizadores de 7 segmentos (B no visualizador da direita e A no seguinte). O bit de seleção da operação (add\_sub) é definido pelo botão de pressão assinalado (btn0). O valor do resultado R, A+B ou A-B, é apresentado em hexadecimal no terceiro visualizador e a saída com o bit de transporte Co é apresentada no visualizador mais à esquerda.

Ensaie o circuito definindo vários valores para os operandos A e B através dos respectivos interruptores. Experimente e verifique a realização de operações de adição e subtração.

**Questão 6:** Supondo que não pressiona o botão add\_sub, como deverão estar posicionados os interruptores que definem A de modo a aparecer no conjunto dos visualizadores o conteúdo “0F...9”?

**Questão 7:** Considere que  $A_3A_2A_1A_0=1001$  e  $B_3B_2B_1B_0=0011$  estão representados em complemento para 2. Recorrendo ao circuito implementado, indique o valor decimal de A+B e A-B, justificando se ocorre *overflow*.

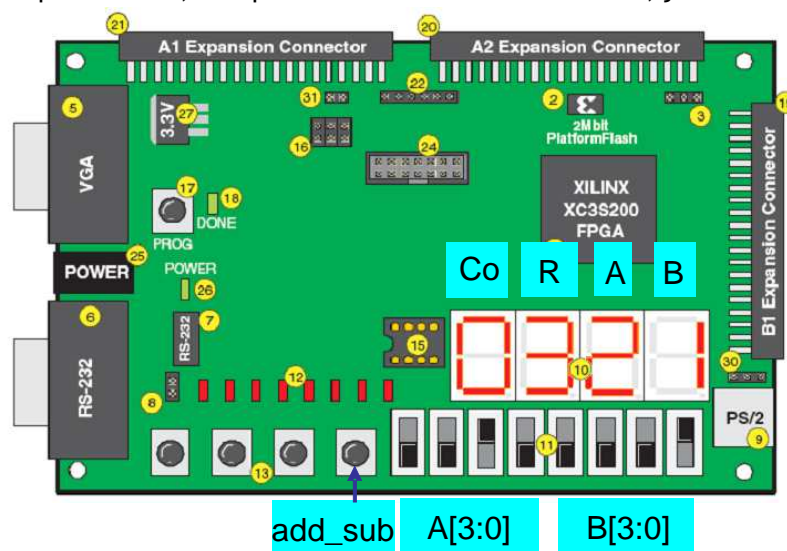


Figura 6 - Identificação das entradas e saídas do somador/subtrator de 4 bits na placa S3board.

#### 4. Tarefa complementar: detetor de *overflow*

**Nota:** Esta tarefa destina-se aos grupos que terminaram o trabalho mais cedo. Deverá portanto ser realizada só depois de ter concluído todos os passos anteriores e de ter verificado com sucesso o funcionamento do circuito na placa de prototipagem.

Tal como estudou, a deteção de *overflow* na adição ou subtração binárias em complemento para 2 pode ser realizada analisando os bits de sinal dos dois operandos e do resultado. Na adição binária ocorre *overflow* quando os dois operandos têm o mesmo sinal e o resultado possui sinal oposto (note que quando se somam operandos com sinais opostos nunca ocorre *overflow*). Como no circuito realizado a subtração é realizada adicionando ao diminuindo o simétrico do diminuidor, a situação de *overflow* acontece quando os dois operandos têm sinais opostos e o sinal do resultado é diferente do sinal do diminuindo.

Construa um circuito capaz de detetar o *overflow* em adições e subtrações, e verifique o seu funcionamento ligando-o ao circuito addsub4b que construiu. O detetor de *overflow* deverá ser escrito em Verilog e ter a interface mostrada a seguir. A saída que identifica *overflow* deverá ser ligada a um dos 8 LEDs disponíveis na placa (sinais ld0 a ld7), solicitando para isso o apoio do docente.

```
module detetor_overflow(signA, signB, signR, addsub, OVF);
input signA, signB, signR; // bits de sinal dos dois operandos e do resultado
input addsub;              // sinal que indica adição (0) ou subtração (1)
output OVF;                // saída que identifica com 1 a ocorrência de overflow
```