ARQUITETURA DE COMPUTADORES

Laboratório 2 – Rede Neuronal Artificial em Assembly

Este laboratório destina-se a consolidar os conhecimentos de programação em assembly da arquitetura ARM. No laboratório será utilizado o emulador de um processador ARM designado de VisUAL. Este emulador implementa uma versão simplificada da *Unified Assembly Language (UAL). Contudo, no contexto do trabalho de laboratório, apenas o subconjunto de instruções indicado no guia de consulta rápida será suportado*. O emulador está disponível no link VisUAL, também disponível na página da cadeira. Para além da informação disponível na página da cadeira, deverá consultar o *User Guide* do emulador para realizar o laboratório.

O trabalho deve ser realizado fora do horário de laboratório, destinando-se este à demonstração e avaliação do trabalho realizado. No final da aula de laboratório deverá submeter o código Assembly no Fénix.

Para garantir a correção da solução, deverá validar a sua solução do laboratório no emulador (VisUAL) e confirmar os resultados, observando os valores finais dos registos ou o conteúdo da memória (Tools / View Memory Content). Pode igualmente colocar pontos de paragem no código (de tal forma que o emulador parará a execução sempre que atingir um destes pontos) clicando no número da linha do código.

Introdução

Uma rede neuronal artificial é um sistema computacional que imita as redes de neurónios que formam o sistema nervoso central dos animais. Estas são formadas por unidades elementares designadas de neurónios. Um neurónio artificial calcula uma saída y a partir de um conjunto de valores de entrada x_i usando a expressão:

$$y = f\left(\sum_{i=1}^{N} w_i x_i + b\right).$$

Em que w_i são os pesos das entradas, b é um termo de bias e $f(x)$ é uma função que depende do tipo de neurónio e é, em geral, não linear. Pode ser, por exemplo, a função sigmoide $(1/(1+e^x))$, a função de retificação $(\max(0,x))$, entre outras. Neste laboratório vamos utilizar a função degrau,

$$f(x) = \begin{cases} 1, \text{if } x \geq 0 \\ 0, \text{if } x < 0 \end{cases}$$

Na figura 1 apresenta-se o diagrama de blocos correspondente a um neurónio com duas entradas.

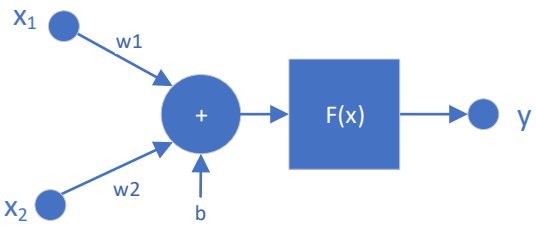

Figura 1 - Neurónio artificial com duas entradas.

Uma rede neuronal pode ser configurada para implementar funções arbitrárias das entradas, dada um número suficiente de neurónios e uma arquitetura adequada. No laboratório, pretende-se implementar uma pequena rede de 2 entradas que implementa a função XOR. Esta rede está representada na figura 2. Deve-se considerar que as entradas A e B tomam apenas os valores 0 ou 1.

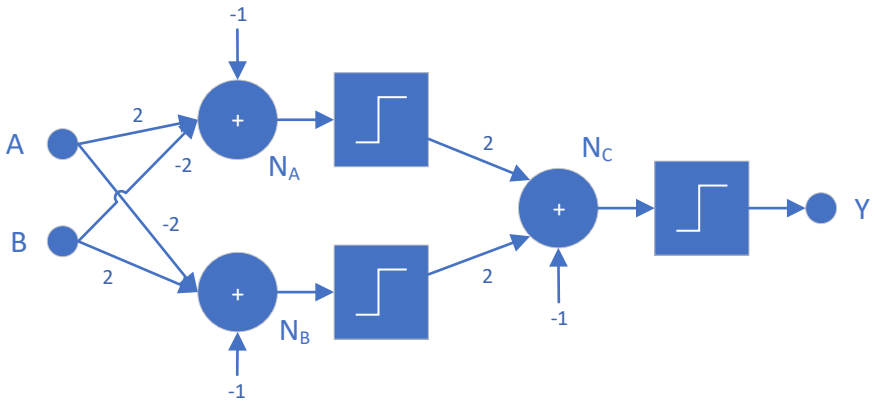

Figura 2 - Pequena rede neuronal que implementa o operador lógico XOR.

O neurónio N_A implementa a função lógica $A\bar{B}$, o neurónio N_B implementa a função lógica $\bar{A}B$ e o neurónio N_C implementa a função logica OR. Daqui resulta que o sinal na saída será $y = A\bar{B} + \bar{A}B = A \oplus B$.

Esta pequena rede neuronal é implementada pelo seguinte código em C:

```
int neuronio(int x1, int x2, int w1, int w2, int b) {
	int s;

	s = x1 * w1 + x2 * w2 + b;

	if (s >= 0) return 1;
	else return 0;
}

int rede_neuronal_xor(int a, int b) {
	int c, d, y;
	c = neuronio(a, b, 2, -2, -1);
	d = neuronio(a, b, -2, 2, -1);
	y = neuronio(c, d, 2, 2, -1);

	return y;
}
```

Exercício 1

a) Implemente, em assembly, a rede neuronal da figura 2, baseando-se no código C apresentado. Para isso deve utilizar 3 funções em assembly:

1. Função “multiplica” – Recebe dois valores em R0 e R1 e retorna o resultado em R3. Caso o multiplicador (R1) seja positivo deve calcular o resultado por somas sucessivas do multiplicando (R0) utilizando um ciclo que itera R1 vezes. Caso o multiplicador seja negativo deve inverter o sinal de ambos os argumentos e proceder da mesma forma.
2. Função “neuronio” – Deve corresponder à função em C de forma a implementar um neurónio **genérico**, chamando a função multiplica.
3. Função “rede_neuronal_xor” – Deve corresponde à função em C e chamar a função neurónio.

Não serão consideradas soluções que não obedeçam a esta estrutura.

b) Escreva o código para testar a rede implementada, invocando a função “rede_neuronal_xor” para os quatro valores possíveis da entrada: 00, 01, 10, e 11. Execute o programa e chame o docente para verificar o resultado.

Exercício 2

Este é um exercício surpresa que será divulgado pelo docente durante a aula.