**Projeto Assembly Parte 2**

**Grupo 95**

**João Morais Nº 86876**

**Pedro Abreu Nº 92541**

No âmbito da cadeira de Introdução à Arquitetura de Computadores foi-nos pedido que desenvolvêssemos um programa que simula o movimento de uma bola num plano inclinado em Assembly P4. Para tal, foram criadas rotinas que utilizam o Terminal do Simulador do P4 e o Temporizador.

A simulação resume-se a um loop controlado pelo temporizador, neste loop é chamada a “Main” do programa.

Os valores que o utilizador pode alterar são: a constante VALORTEMPO, que determina o intervalo de tempo entre atualizações da posição e velocidade da bola, e as variáveis V\_INI, POSICAO\_INI, INTERVALO\_TEMPO.

O programa começa por “desenhar” no terminal a bola e o plano com auxílio da função TEXT\_WRITE. De seguida é ativado o temporizador que por sua vez faz com que a Main seja chamada. Na Main é chamada a função POSICAOX que calcula a posição da bola através da velocidade inicial, posição inicial e aceleração.

A aceleração é calculada pela fórmula a=x\*(g/255), onde x representa o valor lido no acelerómetro.

Após ser calculada a posição, é a chama a função ADD\_TO\_WINDOW que “apaga” a bola do terminal e coloca a bola na nova posição calculada. O valor POSICAO\_INI é atualizado com a nova posição calculada. De seguida é chamada a função VELOCIDADEX que calcula a “nova” velocidade da bola e logo a seguir é atualizado o valor V\_INI com a “nova” velocidade.

Também são guardados em duas tabelas os valores da velocidade e posição obtidos.

As tabelas contendo os valores das posições e velocidades encontram-se a partir do endereço 1000h da memória de dados, reservando 100 endereços para cada tabela.

As tabelas são atualizadas na função .addToMem, isto ajudou-nos na depuração do programa.

Qualquer multiplicação feita no programa recorre à função PRODUTO, que é uma alteração da função dada pelos docentes, que permite multiplicar números representados em vírgula fixa. Neste projeto usámos os 8 bits mais significativos para a parte inteira e os restantes 8 bits para a parte fracionária.