Relatório do Projeto

O projeto consiste na elaboração de um jogo na linguagem Assembly do processador P3. O jogo é uma versão do famoso R-Type, no qual o jogador controla uma nave, num ambiente 2D, e tem de se desviar de asteroides e buracos negros que aparecem e se movimentam na sua direção, com a opção de destruir os asteróides com tiros laser que pode disparar.
A nave é controlada através dos interruptores presentes na placa do P3. É também nela mostrada a posição da nave em linha coluna

O programa consiste num ciclo principal de jogo, onde são feitos vários testes de modo a controlar as rotinas de tratamento das interrupções. Isto foi implementado usando flags, que são inicializadas com o valor 1 e são modificadas para o valor 0 aquando da execução das respetivas rotinas de tratamento das interrupções (i.e. INT_Baixo modifica a flag F_Baixo, etc.).
O ciclo principal é composto por vários testes, que comparam o valor das flags acima referidas com 0. Caso um desses testes seja verdade, o fluxo do programa é encaminhado para uma rotina que controla a execução da respetiva tarefa e repõe a flag ao seu estado inicial.
Antes do ciclo principal está uma pedaço de código que é executado no início do programa, que faz todas as inicializações necessárias e desenha a janela de jogo inicial. Este código está assinalado com uma label para ser fácil a sua execução também quando o jogo é reiniciado, quer por meio da colisão da uma nave com um asteróide/buraco negro, quer pela interrupção IE.
Para efetuar as movimentações da nave e dos objetos de jogo (asteróides, buracos negros e tiros laser), foram escritas rotinas que apagam todos os caracteres do respetivo objeto, modificam as suas posições e voltam a desenhá-los. Antes de os desenhar, são feitos os testes relativamente às colisões, que eliminam os objetos necessários, variando nos diferentes tipos de colisão: colisão asteróide/laser apaga ambos, colisão laser/buraco negro apaga o laser e colisão nave/buraco negro ou asteróide, que encaminha o jogo para a rotina GameOver que mostra o resultado no ecrã (em asteróides destruídos), e espera pelo input do utilizador para prosseguir e reiniciar o jogo, voltando para a Label anteriormente referida, que reinicializa todas as variáveis e flags e de seguida, para o ciclo principal, redesenhando o ecrã de jogo com as configurações iniciais.

Em relação à implementação dos lasers, em primeiro lugar reservamos 78 espaços na memória para as posições dos lasers, numa tabela (TAB). Quando é pressionado o botão correspondente à interrupção que faz a nave disparar, é criado um laser na posição a seguir à do "canhão" da nave. Esta criação é feita adicionando à primeira posição livre na tabela reservada uma posição (que corresponde à posição do canhão da nave + 1) e coloca o um caractere "especial" escolhido por nós na posição seguinte, que simboliza o fim da "lista de objetos". Isto poupa bastantes passos na execução do programa, uma vez que, se só houver 5 lasers por exemplo, em vez de serem percorridos os 78 espaços reservados, serão apenas percorridos os espaços até ser encontrado o caractere especial ('@' neste caso). A posição é fornecida a esta rotina pelo Stack (ou pilha).
De seguida nós escrevemos uma rotina que escreve-se um símbolo, passado como parâmetro pelo Stack, numa determinada posição da janela de texto, passada também pelo Stack. Para mover os lasers criámos uma rotina que apaga os lasers já criados, ou seja colocámos um " " na posição dos lasers, de seguida, incrementamos uma coluna na posição do laser e usamos a mesma rotina para escrever o laser nessa posição.
Estas rotinas são reutilizadas na implementação dos lasers, com a diferença na movimentação dos lasers, que em vez de ser feita incrementando a posição dos objetos, é feita a sua decrementação (-1).

Em relação à implementação da nave, para criar a nave considerámos a posição inicial dada no enunciado e guardamos a posição do "canhão" representado por um ">" numa WORD (um único espaço na memória). Para desenhar a nave e a movimentar utilizámos uma rotina que escreve os caracteres correspondentes à nave na janela de texto e uma que as apaga da mesma forma. Como a representação interna da nave é diferente que as dos restantes objetos, calculamos as restantes

posições dos outros caracteres da nave a partir do “canhão”. A atualização da posição da nave é feita aquando da execução das rotinas de tratamento de interrupção.
Quando é efetuada a movimentação da nave (apagar e desenhar), é também testada a posição da nave relativamente às posições dos asteróides/buracos negros. Caso a posição de algum dos componentes da nave seja igual a alguma das posições dos asteróides ou dos buracos negros, o fluxo do jogo é encaminhado para a rotina GameOver acima descrita.

Na criação e movimentação dos asteróides e buracos negros usámos também o mesmo processo que na implementação dos lasers com a diferença de aquando da criação dos asteróides ou buracos negros definimos uma rotina que gerasse uma linha aleatoriamente para a posição de criação do caractere. Para além disso, definimos que de 3 em 3 asteróides fosse criado um buraco negro utilizando flags que são atualizadas em todas as interrupções por parte do temporizador. Estas flags controlam quando é tempo de movimentar ou criar asteróides e buracos negros.

Quanto às colisões laser/asteroide nós em primeiro lugar realizamos um teste para todos os lasers na janela de texto que consiste em comparar a próxima posição do laser com a posição de cada um dos asteróides/buracos negros. Caso alguma dessas sejam iguais, são eliminados os objetos já descritos acima. Este teste é feito quando é efetuada a movimentação dos lasers. Em contrapartida também realizamos um teste para todos os asteroides/buracos negros na janela de texto que consiste em comparar a próxima posição com a posição dos lasers/nave. Caso alguma dessas sejam iguais, são eliminados os objetos já descritos acima. Caso algum objeto embata com a nave, o fluxo do programa é encaminhado para a rotina GameOver também descrita acima.

Num balanço geral, o jogo foi implementado da forma descrita no enunciado. Escolhemos a opção de o jogo não terminar quando a nave embate nos limites da janela de jogo. Além disso implementámos a possibilidade de disparar múltiplos tiros, como estava descrito na página da disciplina como sendo um extra, sem que haja uma alteração significativa na execução do jogo.

André João Beja Guerra nº 186382
Pedro Miguel Sousa Bernardo nº 186500

Grupo 19