INTRODUÇÃO À ARQUITETURA DE COMPUTADORES

LEIC
IST-TAGUSPARK

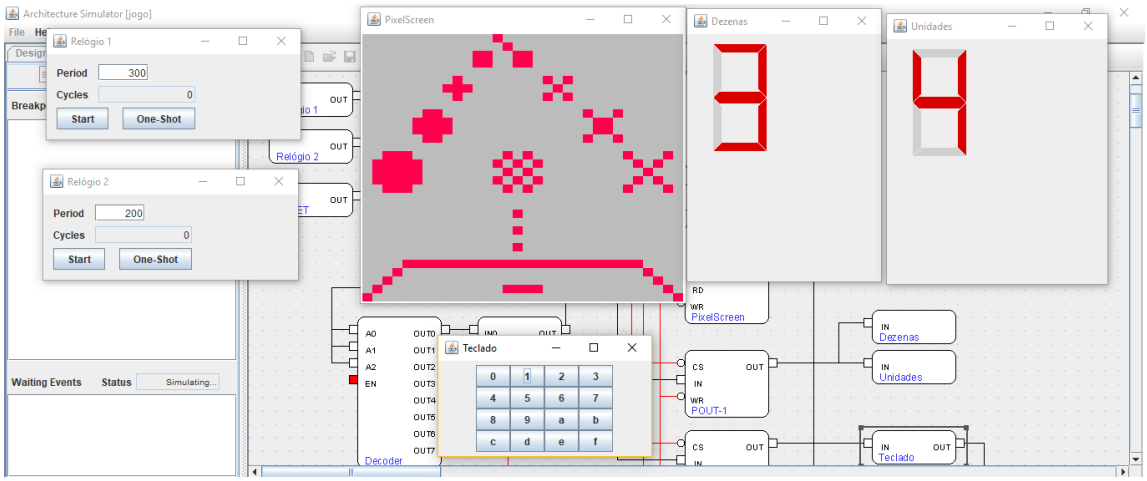
MODELO DO RELATÓRIO DO PROJETO

GRUPO 34

90736 – João Sousa
90747 – Luís Guilherme Silva
90767 – Pedro Cabral

1. Manual de utilizador

Campo de Asteroides é um jogo de simulação do voo de uma nave espacial através de um campo de asteroides.



O jogo é composto por asteroides, bons e maus, uma nave, com um volante e um dispara míssil. Há dois tipos de asteroides, os maus, que não podem colidir com a nave e os asteroides bons, que ao colidir com a nave aumenta a pontuação. Os processos cooperativos são acionados por um grupo de teclas, tais como:

- Tecla C: Começa o jogo, colocando os displays da pontuação a zero, limpando o ecrã inicial, e desenhando a nave com o volante em frente.

```
161 volante:
162     STRING TRESNUM,QUATRONUM
163     STRING 0,0,0,0
164     STRING 1,1,1,1
165     STRING 0,0,0,0
166
```

- Tecla E: Termina o jogo, pintando o ecrã com uma imagem.

- Tecla D: Pausa o jogo, impossibilitando o jogador de fazer qualquer ação.

- Tecla 0: Vira à esquerda, apagando o volante da nave e pinta o volante para a esquerda. Enquanto é pressionada a tecla 0, o campo descola todos os objetos para a esquerda.

```
CALL limpa_volante
MOV R3, VINTNOVE
MOV R4, CATORZE
MOV R9, volante_esquerda
CALL pintar_objecto
```

```
167 volante_esquerda:
168     STRING TRESNUM,QUATRONUM
169     STRING 0,0,0,1
170     STRING 0,1,1,0
171     STRING 1,0,0,0
172
```

- Tecla 1 e Tecla 2: Dispara o míssil na vertical, fazendo com que este destrua o asteroide que esta na sua trajetória.

```
MOV R3, 24
MOV R4, QUINZE
MOV R0, missil_1
MOVB R1, [R0]
CMP R1, 1
JZ fim_1
MOV R9, missil
CALL pintar_objecto
MOV R1, 1
MOVB [R0], R1
```

```
Tecla 2:
CALL tecla_1
```

- Tecla 3: Vira à direita, apagando o volante da nave e pinta o volante para a direita. Enquanto é pressionada a tecla 3, o campo descola todos os objetos para a direita.

```
CALL limpa_volante
MOV R3, VINTNOVE
MOV R4, CATORZE
MOV R9, volante_direita
CALL pintar_objecto
```

```
173 volante_direita:
174     STRING TRESNUM,QUATRONUM
175     STRING 1,0,0,0
176     STRING 0,1,1,0
177     STRING 0,0,0,1
```

2. Conclusões

1. Foi possível concluir a rotina teclado. Esta rotina fazia o varrimento do teclado pela linha, percorrendo depois a coluna. Retornava a tecla para um registo na memória, *var_ultima_tecla_pressa,* mantendo também em registo, R5.

2. Também foi possível concluir a rotina para processar a última tecla pressionada/ atual, **ciclo_compara**. Utilizando uma variável registada na memória, e depois comparada com as vária etiquetas definidas por EQU. São feitas várias comparações, nomeadamente com a etiqueta de início de jogo, C, com uma restrição em relação às outras teclas.

   ```
   MOV R1, var_ultima_tecla_pressa
   CMP R1, C
   JNZ outras    ; salta para as outras opções
   ```

   Se a tecla C for pressionada, o jogo começa atualizando em memória a variável *tecla_c_premida* para um valor diferente de zero. Caso a tecla não tenha sido carregada, sai da rotina **ciclo_compara** e volta a pedir uma nova tecla, impedindo qualquer ação sobre outra tecla senão a C.

   Foi possível concluir a rotina **soma_primos**, que tem como objetivo gerar asteroides com aleatórias características. Através de uma mascara, MASK_BIT_3 é feito, AND R1(nº Primo) , R0(Mask_bit_6), que retorna 0 - não criado ou 1 – criado, para saber se o asteroide em questão já foi criado. Se foi criado, vamos ver à tabela de outro asteroide, se este mesmo já foi criado também. Para ser mais simples, normalmente tem um asteroide bom criado e um asteroide mau não criado. Assim independentemente de ser um asteroide ou outro, o que não está criado, vai ser chamada uma rotina para criar uma tabela inicial, **cria_tab_inicial**. A rotina **cria_tab_inicial** vai receber uma tabela do objeto a criar, num registo, e também vai receber o tipo de asteroide e a sua direção. Nesta rotina é definida a direção do asteroide, com o valor -1 para a esquerda, 0 frente e 1 para a direita. A direção do asteroide vai ser conseguido através dos últimos dois bits to número ramdom gerado pelo **soma_primos**.

   Foi conseguido a rotina **Hex_to_dec** que implementa os displays de 3 em 3, consoante os asteroides bons que colidem com a nave.

   Foi conseguido a rotina **pinta_apaga_pixel** que define se é para pintar ou apagar o pixel, consoante uma variável definida na memoria, se o valor for 0 – apaga, se for 1 – pinta. Esta rotina usa a rotina **apaga_pixel** e **pinta_pixel**.

   Foi conseguido a rotina para movimentar o asteroide, **movimentar**, verificando as tabelas de asteroides e chamando a rotina **movimenta_tabs**, que vai ler as informações do asteroide que estão inseridos numa tabela. A tabela contem cinco String's, String para ver se

o asteroide está **criado**, com 0 – não criado e 1 – criado, a **posição**, contendo o y – linhas do asteroide e o x – colunas, o **tipo**, 0 – mau, 1 – bom, a **direção**, -1 – esquerda ,0 – frente e 1 – direita e a **forma**, que vai sendo incrementada à medida que vai ser lida pela rotina **busca_forma**. Nesta rotina também é utilizada a rotina **atualiza_forma** que vai atualizar a forma do asteroide. Esta rotina é controlada por uma interrupção, **rot_int_0**, verificada por **verifica_int**.

```
tab_asteroide_1:
        STRING 0       ; 1 - criado , 0 - não criado
        STRING 0, 14   ; posicao y,x
        STRING 0       ; tipo 0 - Mau(asteroide) - 1 (minério)
        STRING 0       ; varia entre -1, 0 , 1
        STRING 0       ; forma 0(embrião) ate 5 (idoso)
```

Também foi implementada a rotina para desenhar o míssil e movimentar, **movimenta_missil**, e respetiva interrupção **rot_int_1**, verificada por **verifica_miss**.

São usados relógios para controlar as interrupções, sendo ativadas no inico do jogo.

3. O funcionamento do jogo Campo de Asteroide ficou com algumas funcionalidades por implementar, nomeadamente a **verifica_colisao, atualiza_displays, desloca_objeto, desloca_ecra.**

   O jogo inicia com a rotina **teclado** onde a tecla C tem de ser pressa. Assim que existe uma **tecla pressionada**, a rotina teclado devolve um **valor** que é **registado na memória**. De seguida é chamada a rotina **ciclo_compara**, que vai comparar a tecla pressionada, em memoria, com as opções do jogo, nomeadamente, **a tecla C**, para começar. Se for a tecla C, é chamada uma rotina para a **tecla_C** e esta vai chamar a rotina para **limpar o ecrã todo**, desenha a nave e de seguida pinta o volante em frente. Depois destes passos, os displays são resetados com auxílio da rotina **reset.** A rotina **reset** vai colocar os dois displays a zero. Após esta rotina o jogo está pronto a jogar. Com as interrupções ligadas, os asteroides vão começar a movimentar em direção à nave. Como não está a fazer a **deteta_colisão,** os asteroides sao para pela nave e sao apagar a nave. Não são atualizados os displays porque a rotina **atualiza_displays** não está implementada. Quando é pressionada a tecla 0 e a tecla 3 o volante vira mas a nava não vira nem os objetos movem, pois a rotina **desloca_objeto** e **desloca_ecra.** A **verifica_colisao** é uma **rotina crucial**, pois é necessária para verificar os **limites do ecrã**, que limita o asteroide, **detetar** os asteroides ao **colidir** com a nave e a **colisão** entre o míssil e o asteroide. Para implementar a **deteta_colisão** o asteroide tinha de verificar a posição da nave ,os limites do ecrã e os pixéis do míssil.