INTRODUÇÃO À ARQUITETURA DE COMPUTADORES

LEIC

IST-TAGUSPARK

Projeto Tetris-Invaders

Grupo 22

Alunos:

João Freitas 87671

João Pimenta 87674

Vasco Pires 87708

1. Introdução

No âmbito da disciplina de Introdução à Arquitetura de Computadores foi realizado o projeto Tetris Invaders. Este projeto tinha como objetivo a recriação do jogo Tetris no processador PEPE, usando a linguagem Assembly, com a utilização de alguns periféricos como o teclado, o pixelscreen e o relógio, e de interrupções.

O jogo possui as mesmas especificações que o Tetris normal:

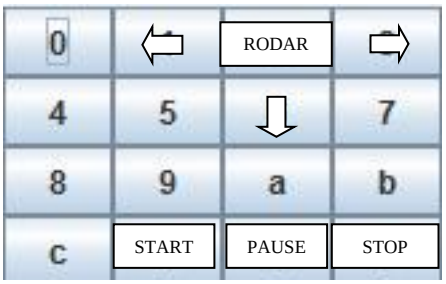
- Existem vários tetraminós que vão caindo um de cada vez sendo o objetivo completar várias linhas;
- Quando a linha é completada é ganha pontuação e os tetraminós acima dessa linha descem;
- Quando os tetraminós são empilhados de modo a atingir o limite superior do ecrã, o jogador perde;

Nesta recriação foi implementado uma variação no jogo:

- Um monstro que se desloca do lado direito do ecrã em linha reta até ao lado esquerdo;
- Se o tetraminó cair em cima do monstro o jogador ganha pontos, no entanto, se o monstro toca no tetraminó o jogo acaba.

Neste projeto são utilizados processos para criar o jogo sendo utilizadas diversas rotinas e acessos aos periféricos para produzir o efeito de jogo.

-O teclado controla o movimento das peças



-O pixelscreen de 32x32 pixeis é onde se visualiza o jogo;

-A pontuação é mostrada nos displays.

A nível gráfico era muito importante implementar a colisão dos tetraminós de modo a que não existisse a sobreposição de algum tetraminó e os restantes elementos do jogo. Era igualmente importante apagar o rasto dos pixéis durante o seu movimento.

A nível técnico era importante fazer corresponder as teclas premidas com os movimentos do tetraminó tal como a incrementação da pontuação do jogador sempre que uma linha seja completada ou que um monstro seja destruído.

2. Conceção e Implementação

Estrutura do software:

Loop principal:

1. Inicializa o BTE
2. Chama a rotina do gerador para a aleatoriedade do aparecimento dos tetraminós
3. Em seguida chama a rotina do teclado para detetar quando é que uma tecla é premida para iniciar o jogo
4. Depois chama a rotina do tetraminó para o fazer aparecer
5. Por fim chama a rotina “cleaner” que deteta quando é que as linhas estão preenchidas e volta ao início

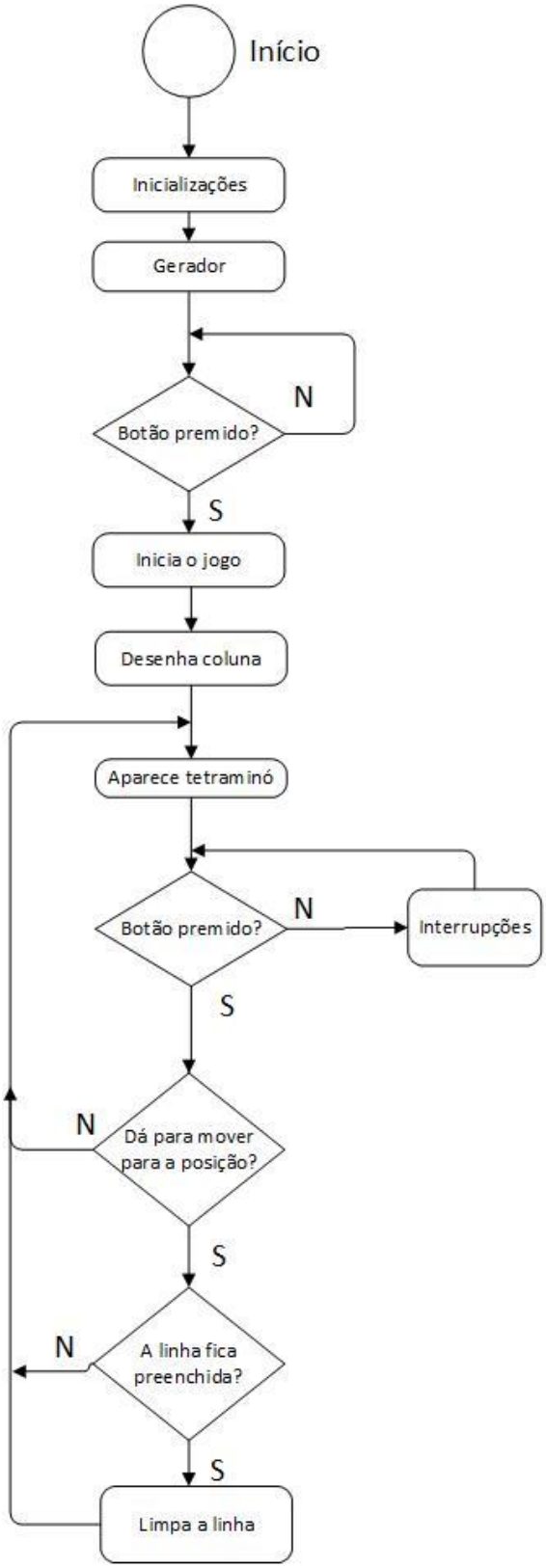
Interrupções:

INT1:

- Desloca o tetraminó uma linha para baixo

INT0:

- Desloca o monstro para a esquerda


Início
Inicializações
Gerador
Botão premido?
N
S
Inicia o jogo
Desenha coluna
Aparece tetraminó
Botão premido?
N
Interrupções
S
N
Dá para mover para a posição?
S
N
A linha fica preenchida?
S
Limpa a linha


Rotina de interrupção
INT1
Início
Desce o tetraminó uma posição
Retorna


Rotina de interrupção
INT0
Início
O monstro avança uma posição
Retorna

No que toca ao software, o fluxograma , mostra o funcionamento base do programa. Como é possível observar, o programa começa por chamar a rotina gerador, que serve de contador de vezes que o ciclo principal corre (utilizado depois por uma rotina para obter um numero pseudoaleatório) seguido da rotina TECLADO, que varre o teclado e obtém o valor de qualquer tecla pressionada, e da rotina de controlo, responsável por tratar das funções de pausa, gameover e reset, que verifica se o utilizador pressionou as teclas de start, pausa ou stop.

De seguida é chamada o função TETRA que irá não só desenhar os tetraminós como também move-los ao longo dos clock's, bem como move-los e rodá-los quando comandados pelo utilizador através do teclado e verificar possíveis colisões dos mesmos com a extremidade do pixel screen (no lado esquerdo), com a coluna da área de jogo (no lado direito) e com outros tetraminós.

A comunicação entre processos é dada através de "call's" e "jump's".

Para a implementação de funções como fazer com que o programa saiba se pode chamar um novo tetraminó ou para a rotina que escreve pixéis no pixel screen saber se tem que desenhar ou apagar ou para a rotina que move os tetraminós saber se os pode ou não mover utilizámos variáveis de estado como FIGLIGADA, APAGA e MOV_VALIDO, entre outras.

Tecnicamente, o desenrolar do software é concebido de forma cíclica entre os vários processos, sendo utilizado um ciclo principal que repete as funções explicadas anteriormente de forma contínua desencadeando um conjunto de respostas consoante os comandos verificados no teclado, dando ao utilizador a oportunidade de controlar os tetraminós e organizá-los da melhor forma, tentando não perder o jogo.

3. Conclusões

Tendo em conta os objetivos do trabalho foi conseguida:

- Criação de rotinas capazes de escrever um pixel e de apagá-lo. Deste modo foi conseguida a implementação de todos os tetraminós, da coluna e da queda dos tetraminós;

- Implementação do teclado que reconhece as teclas premidas pelo jogador e que faz corresponder as suas movimentações de acordo com a tecla;

- Os tetraminós reconhecem os limites do ecrã existindo colisão com as paredes e também com outros tetraminós.

- Os tetraminós são aleatórios devido ao gerador de números aleatórios;

- Existe uma interrupção para a queda dos tetraminós;

- Quando uma linha está cheia é limpa e todos os tetraminós descem uma linha e o jogador ganha dois pontos

Decidimos ainda adicionar ao projeto:

- Ecrãs de gameover/vitória, do modo pausa e um ecrã inicial para que o utilizador saiba como começar o jogo.

No entanto:

- Existe sempre forma de otimizar e simplificar o código e talvez a nossa forma de organizar as soluções implementadas não tenha sido a mais simples, sendo, no entanto, aquela que fez mais sentido no nosso ponto de vista.

- Algo que pudesse ter sido útil que poderíamos ter implementado seriam códigos (algo como developer options) de modo a poder por exemplo, incrementar a pontuação ao pressionar um conjunto especifico de teclas, o que não só seriam boas funcionalidades secretas para o utilizador como boas formas para que nós pudéssemos testar, ao longo da elaboração do projeto, se havíamos conseguido implementar com sucesso a funcionalidade em que estávamos a trabalhar.

- No desenrolar da elaboração do projeto surgiram varias ideias que acabamos por escolher não dar prioridade na nossa gestão de tempo:

  Umas das melhorias que poderiam ser implementadas seria fazer com que as pecas não piscassem quando se movem, ou seja, as pecas não seriam apagadas para serem reescritas num novo sitio, apenas os pixeis que seriam escritos espaços vazios.

Tendo em conta os objetivos iniciais apenas não foi possível a implementação do monstro.