INTRODUÇÃO À ARQUITETURA DE COMPUTADORES

LEIC-TP

IST-TAGUSPARK

RELATÓRIO DO PROJETO

"JOGO DO TETRIS INVADERS"

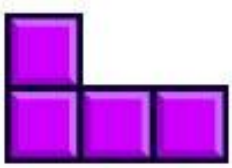
Grupo 14

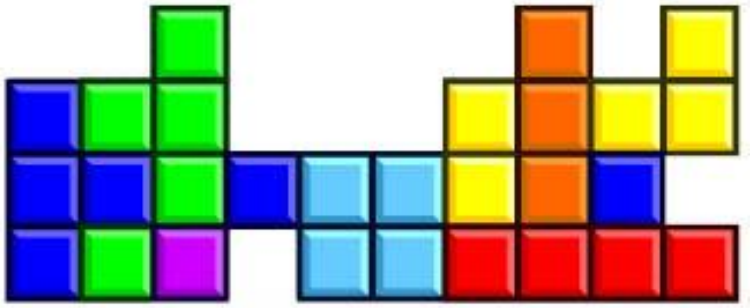
Francisco Sousa, 87657

Inês Albano, 87664

Viviana Bernardo, 87709

Prof. Rui Policarpo Duarte
Turno IAC3179





Este relatório está dividido em **4 secções** e apresenta a seguinte organização:

- A **secção 1** consiste na **Introdução** do projeto
- A **secção 2** consiste na **Conceção e Implementação** do projeto, estando esta secção dividida em **4 secções**.
    - Na **secção 2.1**. descreve-se a **Estrutura Geral** do projeto, onde falamos da estrutura envolvente, quer em termos de **Hardware** quer de **Software – Comunicação entre processos**.
    - Na **secção 2.2** apresenta-se os fluxogramas das principais rotinas do programa e um fluxograma geral.
    - Na **secção 2.3** apresenta-se o **mapa de endereçamento.**
    - Na **secção 2.4.** estão descritas as **interrupções utilizadas**.
- Na **secção 3** é feita a **conclusão do relatório**, referindo os objetivos cumpridos e os não cumpridos, como também o que há a melhorar.
- Na **secção 4** é uma secção de notas que associam as teclas premidas e as funções iteradas sobre o tetraminó.

1. Introdução

Tendo sido realizado no âmbito da cadeira de **Introdução à Arquitetura de Computadores**, este projeto tem como objetivo exercitar as componentes fundamentais desta cadeira, mais precisamente a programação em **linguagem Assembly**, os periféricos e as interrupções.

Desta forma, o projeto consiste na concretização do jogo do Tetris, com o jogo Space Invaders, onde, de forma sucinta, o jogador tem como objetivo intercalar vários tetraminós de modo a completar uma linha no ecrã de jogo de um lado ao outro, limpando essa linha de modo a ganhar pontos. Para além disso, o conjunto de peças não pode chegar à parte superior do ecrã de jogo, se tal acontecer o jogo é dado como terminado. A componente comum com o jogo Space Invaders é a existência de um monstro que se vai movimentando segundo uma trajetória horizontal de um lado ao outro do ecrã, se a peça do tetraminó intersetar o monstro o jogador ganha pontos, se o monstro atingir o lado esquerdo do ecrã de jogo, o jogo é dado por terminado.

De uma forma geral, é necessário compreender que, para conceber este jogo, devem ser acesos e apagados vários pixéis, de uma janela de 32x32 pixéis, conforme o movimento dos tetraminós.

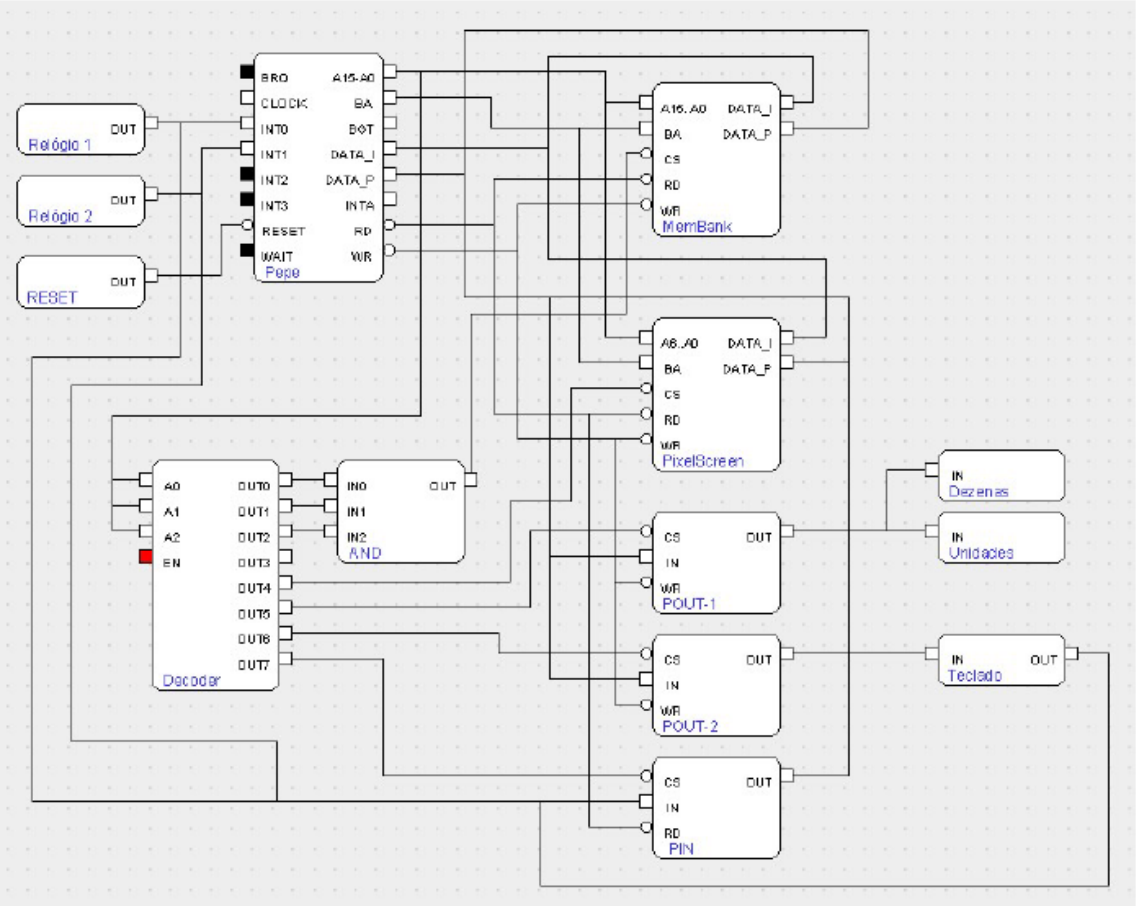
2. Conceção e Implementação

2.1 Estrutura Geral

O projeto fundamenta-se em duas partes, sendo estas o hardware (circuito montado, ficheiro **jogo.cmod**) e o software (código implementado).

2.1.1 Hardware

O seguinte esquema de blocos mostra o circuito utilizado no projeto.



- **PEPE (16 bits) -** Componente responsável pela lógica, aritmética e ainda pela interpretação do código e controlo de todos os componentes.
- **Relógio -** É utilizado como base para a temporização da descida dos tetraminós, como também para o controlo de tempo do movimento do monstro, que surge do lado direito do ecrã e movimenta-se segundo uma trajetória horizontal para o lado oposto.

- **PixelScreen -** Periférico com a finalidade de ser o ecrã de jogo. Serve para visualizar o movimento dos tetraminós e do monstro.

- **Teclado -** Periférico responsável movimento dos tetraminós, como também colocar o jogo em PAUSE e dar inicio ao jogo START

- **MemBank -** É o bloco de memória onde são guardadas as variáveis necessárias ao funcionamento do jogo, como a pontuação, tecla premida, etc.

- **POUT -** Os POUT têm como principal função a escrita da informação. No caso do POUT-1, é escrito no Hexa Display. Por outro lado, o POUT-2 ativa as linhas do teclado, para posteriormente, ao ser selecionada a tecla certa seja refletida a sua função no desenvolvimento do jogo, através do bloco PIN.

2.1.2 Software – Comunicação entre processos

A comunicação entre os diferentes processos recorre a chamadas às rotinas necessárias no momento, pelo que, de certa forma, estas acabam por estar encadeadas umas nas outras.

- **tecla_premida:** Rotina que faz o acesso ao teclado, verificando se houve alguma tecla pressionada.

- **limpa_screen:** Rotina que limpa os endereços do ecrã desde 8000H até 807FH. A rotina coloca todos os bits do PixelScreen a 0 e desenha um barra vertical no endereço 0008H.

- **gerador_tetra:** Rotina que gera um tetraminó no jogo de modo aleatório

- **teclado_funcoes:** Rotina que faz a associação das teclas do teclado às respetivas movimentações dos tetraminós (tecla_0 movimenta o tetraminó para a esquerda, tecla_1 faz rodar o tetraminó para as várias posições que este pode tomar, tecla_2 movimenta o tetraminó para a esquerda, tecla_5 faz descer rapidamente a o tetraminó, tecla_6 faz pausa no jogo, tecla_E dá início ao jogo, tecla_F faz terminar o jogo). A rotina chama a rotina tecla_premida para associar as teclas.

- **pintar_ponto:** Rotina que é chamada sempre que for necessário pintar um ponto no ecrã, devolvendo o byte do PixelScreen atualizado.

- **print_tetramino:** Rotina que é chamada para pintar um determinado tetraminó no PixelScreen. A rotina print_tetramino é chamada várias vezes em todo o código, uma vez que é sempre necessário desenhar outra peça no ecrã.

- **rodar_peca:** Rotina que é chamada para variar as diferentes posições que o tetraminó em jogo pode tomar. Na prática, a rotina rodar_peça apaga o tetraminó e devolve a posição seguinte do mesmo tetraminó chamando a rotina print_tetraminó, refletindo o encadeamento de rotinas.

- **escreve_movimento:** Rotina responsável pelos movimentos do tetraminó. A rotina escreve_movimento é chamada nas subrotinas move_dto, move_baixo e move_esq. Dentro desta rotina é chamada a rotina print_tetraminó. Assim, a rotina escreve_movimento é responsável por desenhar uma nova peça.

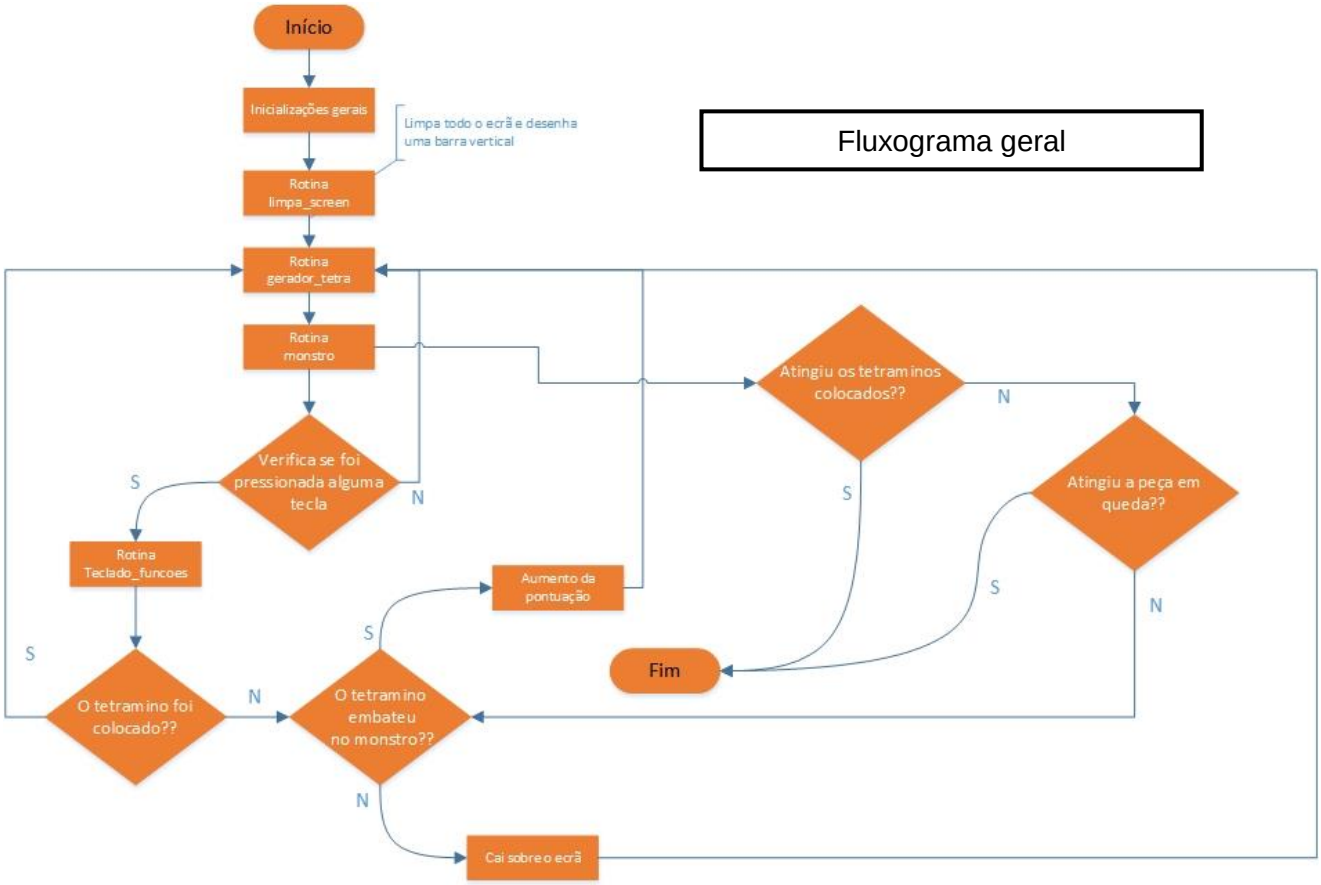
- **criar_peca:** Rotina responsável pelo desenho de um tetraminó, que é implementada na rotina gerador_tetra. Será criado um novo tetraminó sempre que for chamada a rotina gerador_tetra.

- **interrupcoes:** Rotina associada ao Relógio 1 que faz mover o tetraminó de acordo com o clock. Ou seja, é chamada a rotina escreve_movimento para desenhar o tetraminó na linha seguinte.

- **avalia_linha:** Rotina responsável por ver se a linha do ecrã de jogo está ou não pintada. Se sim a rotina limpar_linha é chamada e essa linha é removida.

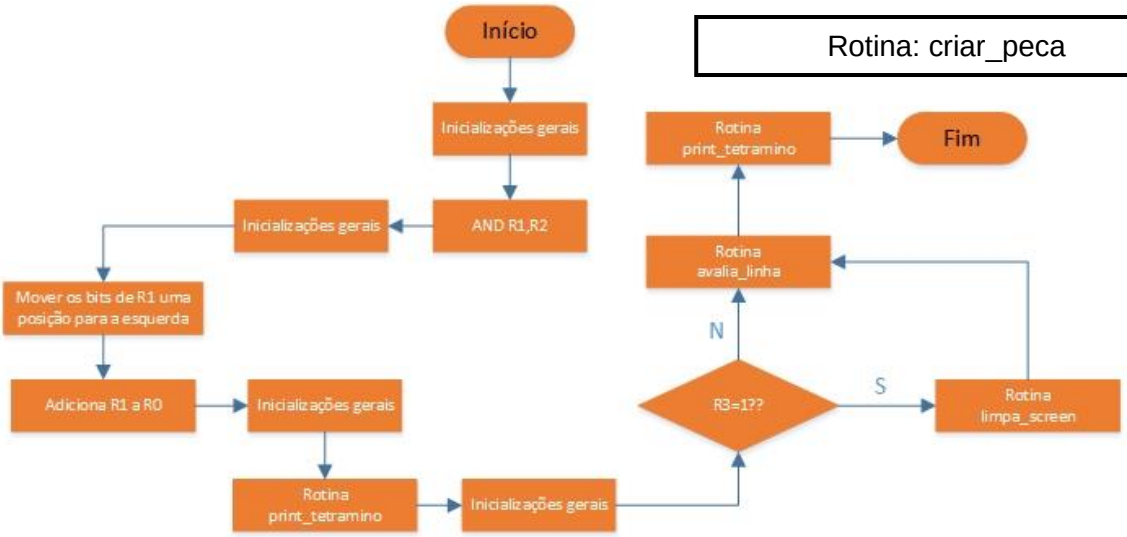
- **limpar_linha:** Rotina que limpa/remove a linha.

2.2 Fluxogramas

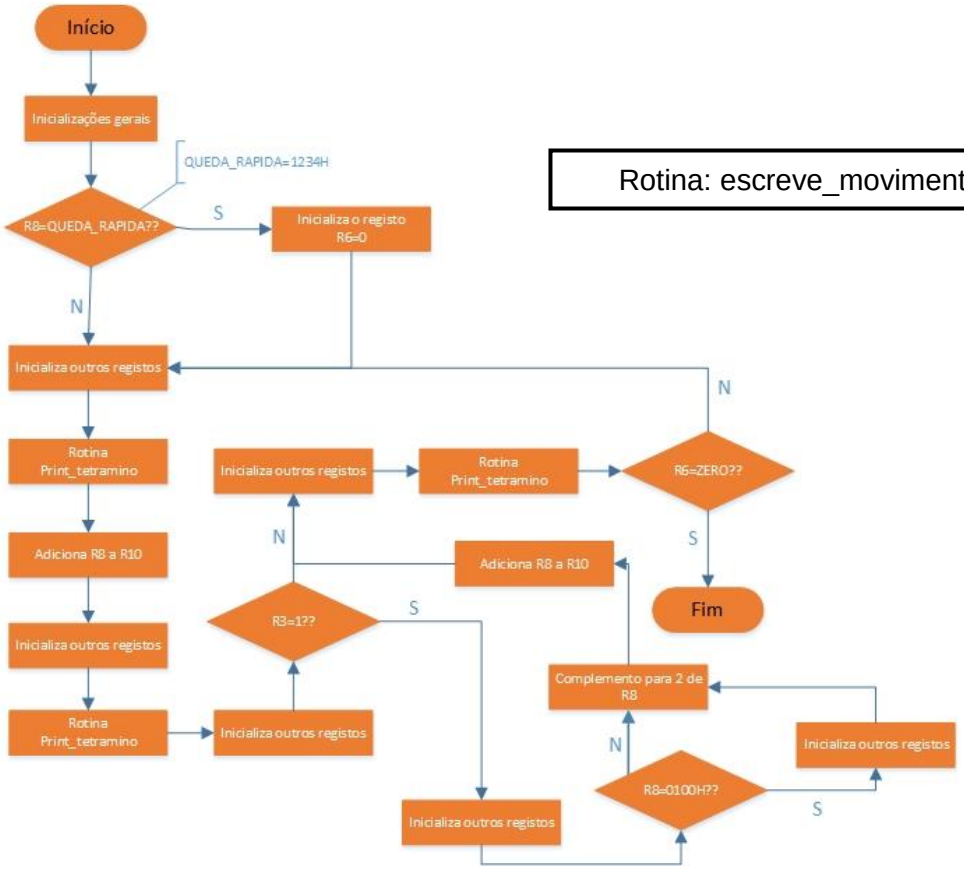


Fluxograma geral

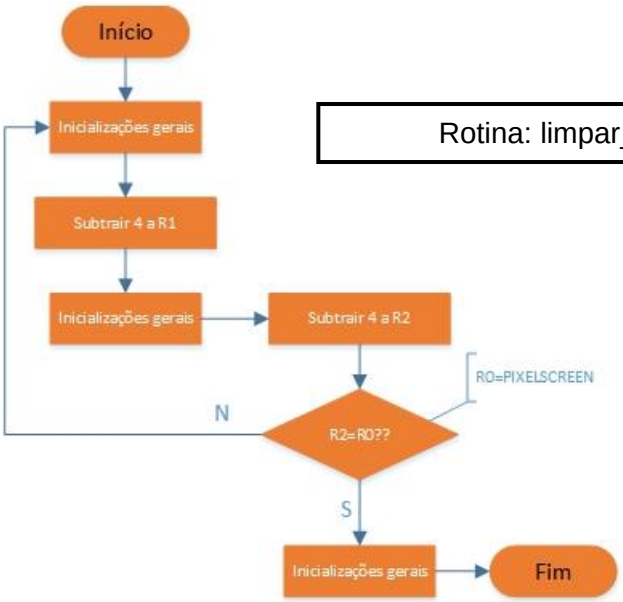
Rotina: criar_peca



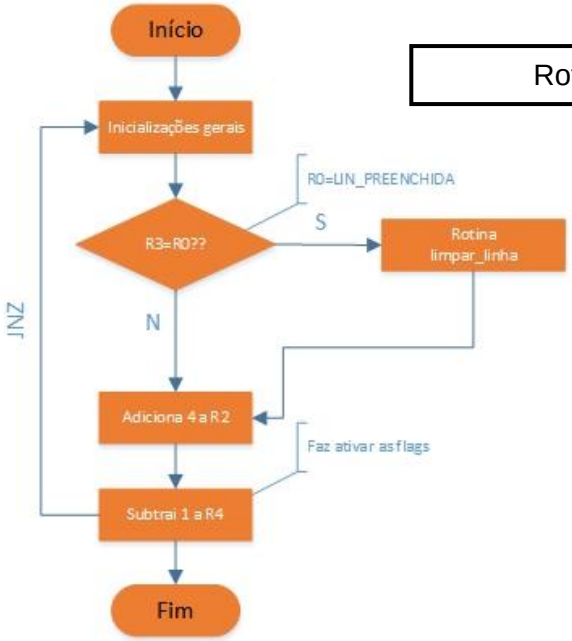
Rotina: escreve_movimento



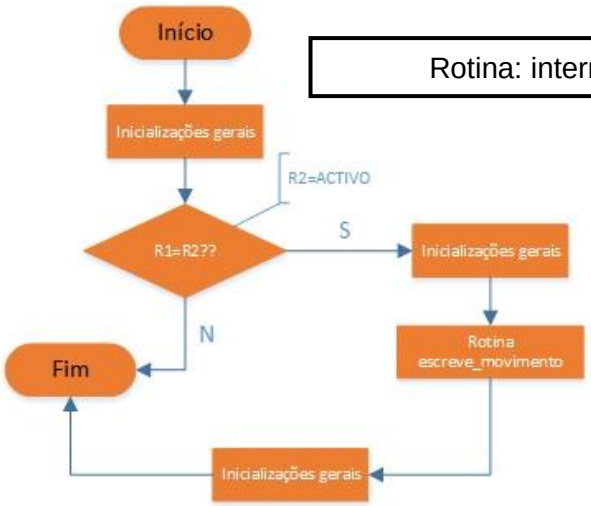
Rotina: limpar_linha



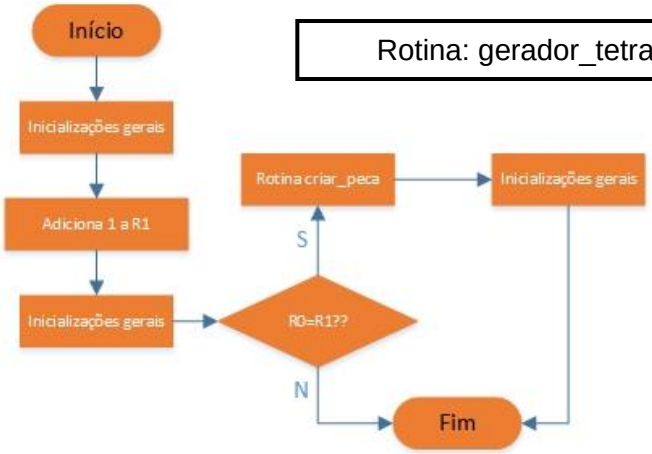
Rotina: avalia_linha



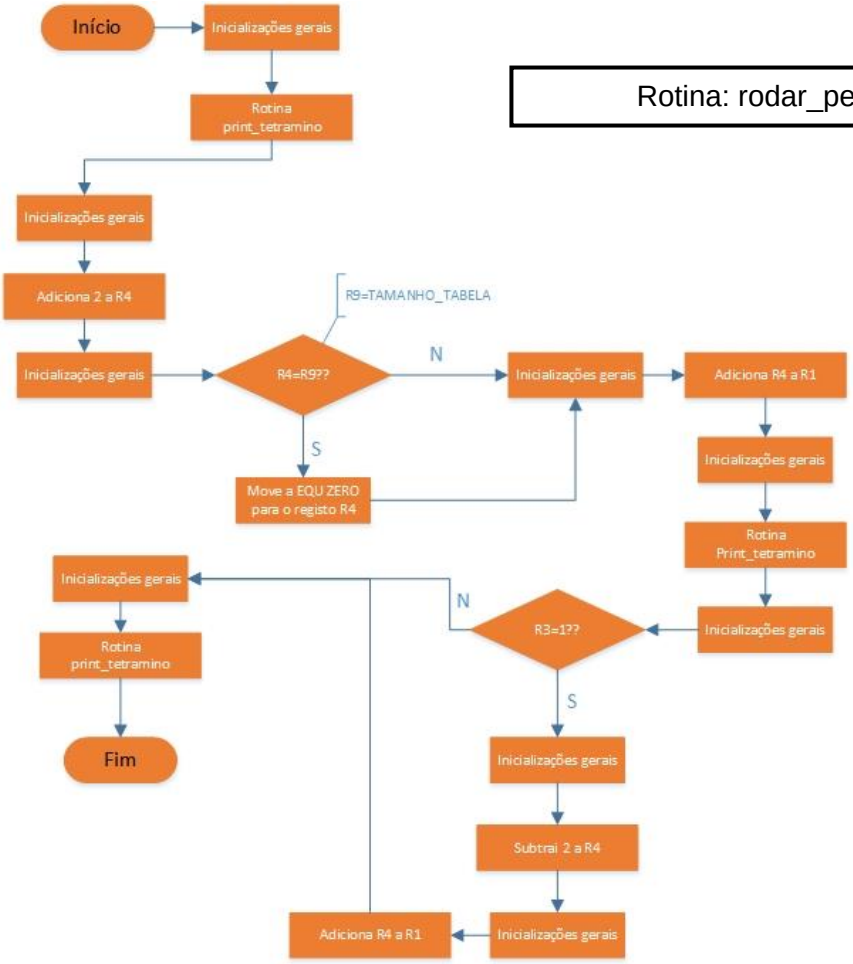
Rotina: interrupcoes



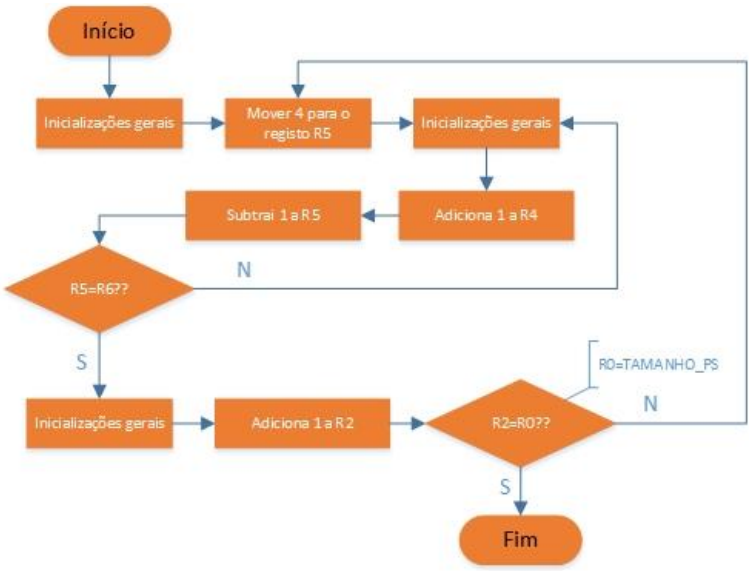
Rotina: gerador_tetramino



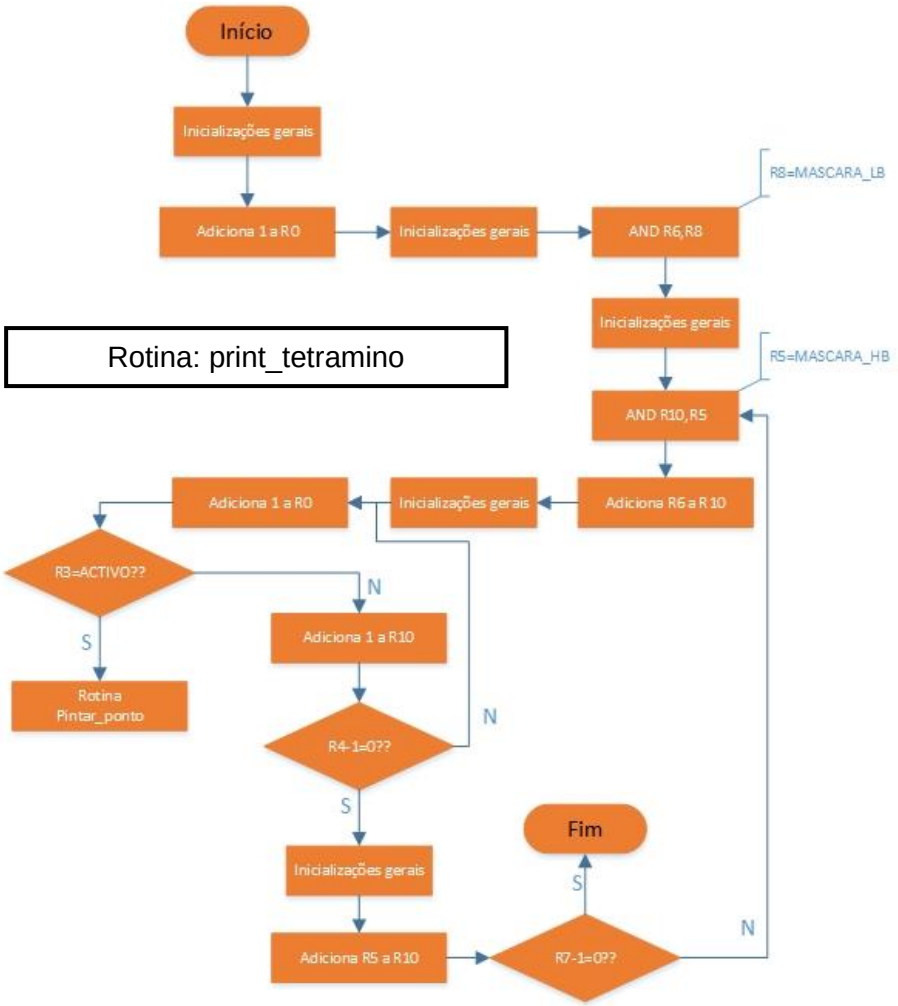
Rotina: rodar_peca

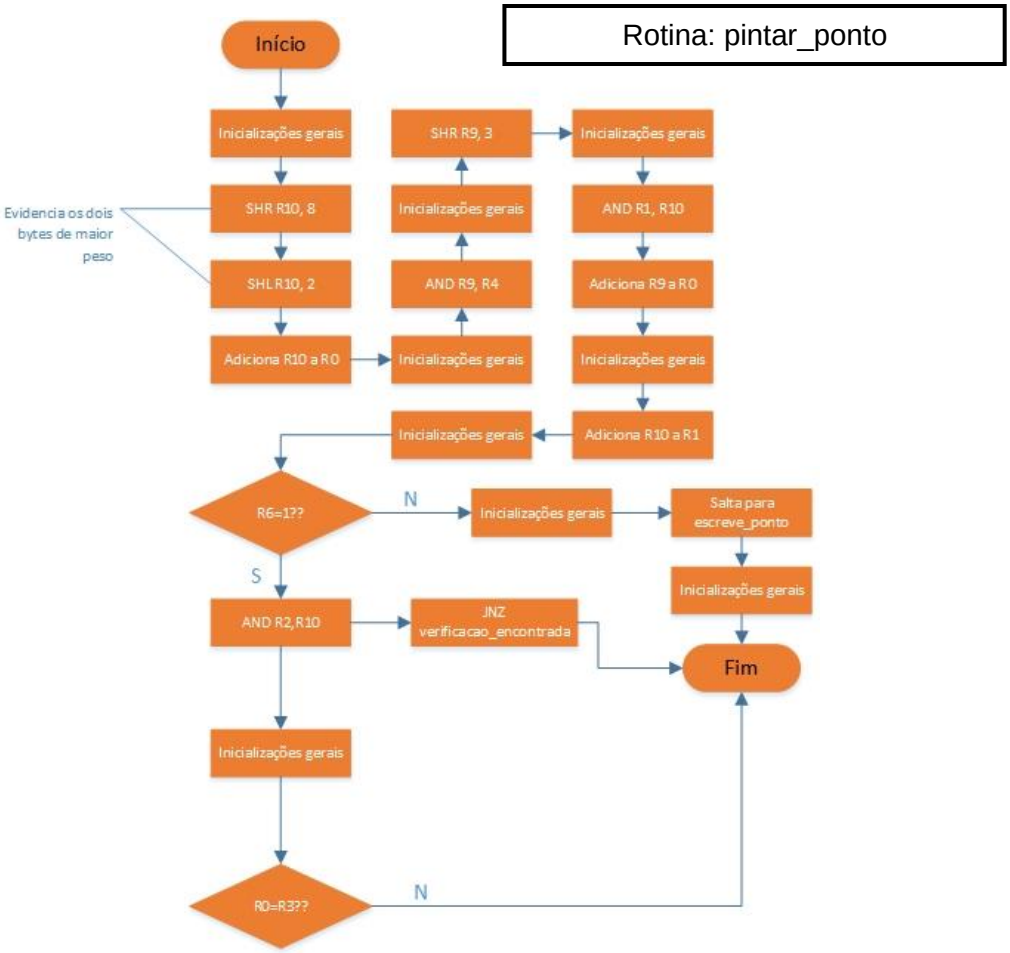


Rotina: limpa_screen



Rotina: print_tetramino





2.3 Mapa de Endereçamento

O mapa de endereçamento representado na tabela abaixo é referente aos endereços em que os dispositivos podem ser acedidos pelo **PEPE (16 bits).**

Dispositivo	Endereços
RAM (MemBank)	0000H a 5FFFH
PixelScreen	8000H a 807FH
POUT-1 (periférico de saída de 8 bits)	0A000H
POUT-2 (periférico de saída de 8 bits)	0C000H
PIN (periférico de entrada de 8 bits)	0E000H

2.3 Interrupções

Ao longo do projeto são utilizadas duas rotinas de interrupção: uma para movimentar o tetraminó até à base de jogo, outra para controlar o movimento do monstro.

3. Conclusões

Com este projeto, conseguimos desenvolver os nossos conhecimentos da linguagem Assembly, bem como compreender melhor o funcionamento do jogo utilizado e dos seus componentes, os periféricos.

O objetivo principal deste projeto era a concretização do jogo do Tetris Invaders em Assembly, o que foi possível, apesar de não totalmente da forma esperada, nomeadamente o aparecimento do monstro que surge 25% das vezes o número de tetraminós gerados e o momento correto lateral do monstro.

A parte mais complicada do jogo foi configurar as teclas PAUSE e o desenvolvimento correto do monstro. De resto conseguimos associar as teclas do teclado ao movimento dos tetraminós, apagar a linha quando esta está completa, associar os displays hexadecimais à pontuação gerada, fazer o aparecimento do monstro, bem como outros objetivos propostos.

O nosso projeto não está completamente de acordo com todas as exigências pedidas, no entanto, conseguimos obter um resultado bastante satisfatório. Para além disto, consideramos que o nosso código está bastante otimizado cumprindo vários objetivos propostos. Tivemos o cuidado de escrever várias rotinas que recorrem também a outras rotinas necessárias no momento, pelo que estas estão encadeadas umas nas outras, o que reflete bem a comunicação entre processos.

Na nossa opinião, deviam ter sido dadas mais diretrizes para a realização do projecto em aulas práticas, uma vez que até então nunca tínhamos trabalhado com linguagem Assembly, no entanto com a prática tornou-se mais fácil programar neste tipo de linguagem.

Problemas com o código:

- PAUSE – quando a tecla é premida é activado a pausa do jogo. Esta funciona, durante alguns momentos, no entanto não corre a função como deveria.

- MOVIMENTO DO MONSTRO – o monstro não possui a verificação quando colide com as peças dos tetraminós.

4. Notas teclado:

TECLA PREMIDA	FUNÇÃO ASSOCIADA
0	Move para a esquerda
1	Roda tetraminó
2	Move para a direita
5	Desce peça rapidamente
6	Pausa o jogo
E	Inicializa programa
F	Termina o jogo