```
;=================================================================
;               Aquitectura de Computadores - Projecto Pac-Man
;               Grupo 33
;               André Silva - 68707
;               Mariana Azevedo - 72595
;=================================================================

; Definição de constantes

Topo_Pilha      EQU     FDFFh
Int_Mask        EQU     FFFAh

TimerValue      EQU     FFF6h                   ;Recebe o valor a "contar"
                                                ;em intervalos de 100ms
TimerControl    EQU     FFF7h                   ;1 = conta, 0 = pára

INI_Int_Mask    EQU     1000000000001111b       ;Permite ints. 0,1,2,3,15

N_LIN           EQU     14                      ;Altura do mapa de jogo
N_COL           EQU     21                      ;Largura do mapa de jogo

N_LIN_m1        EQU     13                      ;N_LIN - 1
N_COL_m1        EQU     20                      ;N_COL - 1

ENABLE          EQU     1                       ;Constante de uso geral

COD_Niv1        EQU     0000000000000000b       ;Código associado ao Nível 1
COD_Niv2        EQU     0000000000000001b       ;Código associado ao Nível 2
COD_Niv3        EQU     0000000000000010b       ;Código associado ao Nível 3

ALA_Estado      EQU     FFF9h                   ;Estado actual das alavancas
ALA_Mask        EQU     0000000000000011b       ;Máscara do valor relevante
                                                ;do estado das alavancas

PORTO_Estado    EQU     FFFDh                   ;Porto de estado
PORTO_Ctrl      EQU     FFFCh                   ;Porto de controlo
PORTO_Esc       EQU     FFFEh                   ;Porto de escrita
INI_Index       EQU     FFFFh                   ;M[INI_Index] funciona como
                                                ;porto de leitura

LEDs            EQU     FFF8h                   ;Endereço dos LEDs da placa
Display         EQU     FFF0h                   ;Endereço do display de
                                                ;7 segmentos

LCD_Ctrl        EQU     FFF4h                   ;Porto de controlo (Display LCD)
LCD_Esc         EQU     FFF5h                   ;Porto de escrita (Display LCD)
LCD_Ini         EQU     1000000000000010b       ;Estado inicial do porto
                                                ;de controlo (LCD)

INC_Col         EQU     0001h                   ;Somando à posição actual passa
                                                ;para a coluna seguinte
INC_Lin         EQU     0100h                   ;Somando à posição actual passa
                                                ;para a linha seguinte

CIMA            EQU     0                       ;Códigos correspondentes a cada
BAIXO           EQU     1                       ;direcção de movimento
ESQ             EQU     2
DIR             EQU     3
INDEF           EQU     4

DIF_Niv1        EQU     10                      ;Níveis de dificuldade do jogo
DIF_Niv2        EQU     9                       ;Correspondem aos valores a
```



```
DIF_Niv3        EQU     8                       ;colocar na variável TimeLong
DIF_Niv4        EQU     7
DIF_Niv5        EQU     6
DIF_Niv6        EQU     5
DIF_Niv7        EQU     4
DIF_Niv8        EQU     3

LIM_DifPts      EQU     20                      ;Dificuldade aumenta a cada
                                                ;'LIM_DifPts' segundos

LIM_Pontuacao   EQU     9999                    ;Define como pontuação máxima
                                                ;9999 (para evitar overflow da
                                                ;pontuação na janela da placa)

LEDs1           EQU     0000000000000001b       ;Valores cujo número na etiqueta

LEDs2           EQU     0000000000000011b       ;indica o número de LEDs ligados

LEDs3           EQU     0000000000000111b       ;na placa.
LEDs4           EQU     0000000000001111b       ;Correspondem aos valores a
LEDs5           EQU     0000000000011111b       ;colocar no endereço definido
LEDs6           EQU     0000000000111111b       ;pela constante LEDs
LEDs7           EQU     0000000001111111b
LEDs8           EQU     0000000011111111b

PtsPonto        EQU     2                       ;Número de pontos que cada
PtsBanana       EQU     10                      ;bónus atribui
PtsPera         EQU     20
PtsGelado       EQU     30

ASCII_Int       EQU     0030h                   ;Somado a um inteiro devolve
                                                ;o código ASCII desse inteiro

Fim_Menu_L3     EQU     800Bh                   ;Posição em memória do final das
Fim_Menu_L5     EQU     801Dh                   ;strings, para se saber quando
Fim_Menu_L7     EQU     804Bh                   ;parar de escrever.
Fim_Menu_L8     EQU     8068h                   ;Utilizar constantes em vez de
                                                ;guardar em memória caracteres
Fim_MJogo_L7    EQU     806Eh                   ;indicadores de final de texto
Fim_MJogo_L10   EQU     8083h                   ;evita a ocupação desnecessária
Fim_MJogo_L11   EQU     8092h                   ;de posições de memória



; Tabela de Interrupcoes

                ORIG    FE00h
INT0            WORD    IntCima
INT1            WORD    IntBaixo
INT2            WORD    IntEsq
INT3            WORD    IntDir

                ORIG    FE0Fh
INT15           WORD    IntTimer

; Início do armazenamento em memória

                ORIG    8000h

; Armazenamento das mensagens de texto em memória
; Mensagem do menu de jogo
```

```
Menu_L3         STR     'Jogo Pac-Man'          ;Usado também nas mens. de jogo
Menu_L5         STR     'Pontuacao maxima: '
Menu_L7         STR     'Use as alavancas para escolher o nivel inicial'
Menu_L8         STR     'e em seguida prima uma tecla.'

; Mensagens dentro do jogo

MJogo_L7        STR     'Nivel '
MJogo_L10       STR     'Premir uma tecla para'
MJogo_L11       STR     'iniciar o jogo.'
MJogo_N1        STR     '1'
MJogo_N2        STR     '2'
MJogo_N3        STR     '3'

; Armazenamento dos mapas em memória
; Mapa correspondente ao nível 1

                ;       01234567890123456789
Nivel1_L0       STR     '######################'
Nivel1_L1       STR     '#&........M........%#'
Nivel1_L2       STR     '#.#####. #.# .#####.#'
Nivel1_L3       STR     '#.#   #. #.# .#   #.#'
Nivel1_L4       STR     '#.# # #. #.# .# # #.#'
Nivel1_L5       STR     '#...................#'
Nivel1_L6       STR     '##.#### # ) # #### ##'
Nivel1_L7       STR     '##.#### ##### #### ##'
Nivel1_L8       STR     '# .................#'
Nivel1_L9       STR     '# # # #  # #  # # #.#'
Nivel1_L10      STR     '# #   #  # #  #   #.#'
Nivel1_L11      STR     '# #####  # #  #####.#'
Nivel1_L12      STR     '#         @........#'
Nivel1_L13      STR     '######################'

; Mapa correspondente ao nível 2

                ;       01234567890123456789
Nivel2_L0       STR     '######################'
Nivel2_L1       STR     '#.........@........#'
Nivel2_L2       STR     '#.#####..###..#####.#'
Nivel2_L3       STR     '#.#...#..#.#..#...#.#'
Nivel2_L4       STR     '#.#.#.#..#.#..#.#.#.#'
Nivel2_L5       STR     '#...................M#'
Nivel2_L6       STR     '##.####.#####.####.##'
Nivel2_L7       STR     '##.#.##.#.).#.##.#.##'
Nivel2_L8       STR     '#M..................#'
Nivel2_L9       STR     '#.########.########.#'
Nivel2_L10      STR     '#.#.&.#..#.#..#.%.#.#'
Nivel2_L11      STR     '#.##.##..#.#..##.##.#'
Nivel2_L12      STR     '#.........M........#'
Nivel2_L13      STR     '######################'

; Mapa correspondente ao nível 3

                ;       01234567890123456789
Nivel3_L0       STR     '######################'
Nivel3_L1       STR     '#..M.............M..#'
Nivel3_L2       STR     '#.#####..#.#..#####.#'
Nivel3_L3       STR     '#.#......#.#......#.#'
Nivel3_L4       STR     '#.#####..###..#####.#'
Nivel3_L5       STR     '#M..................#'
Nivel3_L6       STR     '##.#..####@####..#.##'
Nivel3_L7       STR     '##.####..###..####.##'
```



```
Nivel3_L8       STR     '#...................M#'
Nivel3_L9       STR     '#######.#.#.#.#######'
Nivel3_L10      STR     '#&#...#.#.).#.#...#%#'
Nivel3_L11      STR     '#.##.##.#####.##.##.#'
Nivel3_L12      STR     '#.........M.........#'
Nivel3_L13      STR     '######################'

; Mapa correspondente à animação "Game Over" (funcionalidade extra)

                ;       01234567890123456789
GameOver_L0     STR     '#####    .-.    #####'
GameOver_L1     STR     '#####   | OO|   #####'
GameOver_L2     STR     '#####   |   |   #####'
GameOver_L3     STR     '#####   '^^^'   #####'
GameOver_L4     STR     '#####           #####'
GameOver_L5     STR     '##### GAME OVER #####'
GameOver_L6     STR     '#####           #####'
GameOver_L7     STR     '######################'
GameOver_L8     STR     '#                    #'
GameOver_L9     STR     '#Pressione uma tecla#'
GameOver_L10    STR     '#para voltar ao menu#'
GameOver_L11    STR     '#                    #'
GameOver_L12    STR     '######################'
GameOver_L13    STR     '######################'

; Definição de variáveis

RandomVar       WORD    A5A5h                   ;Conterá número semi-aleatório
MapaActual      WORD    COD_Niv1                ;Conterá COD_NivX
NumPontos       WORD    0000h                   ;Núm. de pontos no mapa actual
NumMonstros     WORD    0000h                   ;Núm. de monstros no mapa actual
PosPacManT      WORD    0000h                   ;Pos. actual Pac-Man
                                                ;na janela de texto
PosPacManM      WORD    0000h                   ;Pos. actual Pac-Man em memória
PosMonstro1T    WORD    0000h                   ;Pos. actual Monstro 1
                                                ;na janela de texto
PosMonstro1M    WORD    0000h                   ;Pos. actual Monstro 1
                                                ;em memória
PosMonstro2T    WORD    0000h                   ;Pos. actual Monstro 2
                                                ;na janela de texto
PosMonstro2M    WORD    0000h                   ;Pos. actual Monstro 2
                                                ;em memória
PosMonstro3T    WORD    0000h                   ;Pos. actual Monstro 3
                                                ;na janela de texto
PosMonstro3M    WORD    0000h                   ;Pos. actual Monstro 3
                                                ;em memória
PosMonstro4T    WORD    0000h                   ;Pos. actual Monstro 4
                                                ;na janela de texto
PosMonstro4M    WORD    0000h                   ;Pos. actual Monstro 4
                                                ;em memória
PosMonstro5T    WORD    0000h                   ;Pos. actual Monstro 5
                                                ;na janela de texto
PosMonstro5M    WORD    0000h                   ;Pos. actual Monstro 5
                                                ;em memória
Pontuacao       WORD    0000h                   ;Pontuacao actual do jogador
PontMaxima      WORD    0000h                   ;High score
Direccao        WORD    INDEF                   ;Direcção de movim. do Pac-Man
DirMonstro1     WORD    INDEF                   ;Direcção do movim. do Monstro 1
DirMonstro2     WORD    INDEF                   ;Direcção do movim. do Monstro 2
DirMonstro3     WORD    INDEF                   ;Direcção do movim. do Monstro 3
DirMonstro4     WORD    INDEF                   ;Direcção do movim. do Monstro 4
DirMonstro5     WORD    INDEF                   ;Direcção do movim. do Monstro 5
```

```
BaixoMon1       WORD    ' '                     ;O que está por baixo do Mons. 1
BaixoMon2       WORD    ' '                     ;O que está por baixo do Mons. 2
BaixoMon3       WORD    ' '                     ;O que está por baixo do Mons. 3
BaixoMon4       WORD    ' '                     ;O que está por baixo do Mons. 4
BaixoMon5       WORD    ' '                     ;O que está por baixo do Mons. 5
CondTimer       WORD    0                       ;1 após INT15.
CondGameOver    WORD    0                       ;1 após o Pac-Man ser capturado
TimeLong        WORD    DIF_Niv1                ;Valor a colocar no
                                                ;endereço TimerValue

; Reserva de espaço em memória para o jogo actual

EspacoJogo      TAB     294                     ;Cada mapa contém 294 células


; Início do programa

                ORIG    0000h
                JMP     Inicio

;=================================================================

; EsperaTecla: Rotina que espera que o utilizador carregue numa
;              tecla antes de prosseguir.
;              Incrementa também M[RandomVar], para um valor
;              ainda mais aleatório
;=================================================================

EsperaTecla:    PUSH    R1
EsperaTecla2:   INC     M[RandomVar]
                CMP     R0, M[PORTO_Estado]     ;Foi pressionada uma tecla?
                BR.Z    EsperaTecla2            ;Se não, verifica outra vez
                MOV     R1, M[INI_Index]        ;Recoloca porto de estado a 0
                POP     R1
                RET

;=================================================================

; EscolheNivel: Rotina que lê o estado das alavancas para
;               escolher um nível
;=================================================================

EscolheNivel:   PUSH    R1
                MOV     R1, M[ALA_Estado]
                AND     R1, ALA_Mask            ;Atribui relevância apenas aos
                                                ;dois interruptores da direita
                MOV     M[MapaActual], R1       ;Regista o mapa actual
                CMP     R1, COD_Niv1            ;Compara o estado das alavancas
                BR.Z    SelecNiv1               ;com o código dos níveis,
                CMP     R1, COD_Niv2            ;de forma a seleccionar
                BR.Z    SelecNiv2               ;o mapa apropriado
                CMP     R1, COD_Niv3
                BR.Z    SelecNiv3
                MOV     R1, M[INI_Index]        ;Recoloca porto de estado a 0
                CALL    EsperaTecla
                BR      EscolheNivel
SelecNiv1:      POP     R1                      ;Dependendo do nível escolhido
                PUSH    Nivel1_L0               ;passa como parâmetro para
                CALL    EscreveMapa             ;a rotina EscreveMapa
                BR      FimEscNivel             ;o primeiro endereço
SelecNiv2:      POP     R1                      ;onde o mapa do nível desejado
                PUSH    Nivel2_L0               ;se encontra armazenado em
```



```
                CALL    EscreveMapa             ;memória
                BR      FimEscNivel
SelecNiv3:      POP     R1
                PUSH    Nivel3_L0
                CALL    EscreveMapa
FimEscNivel:    RET

;=================================================================
; EscreveMapa: Rotina que escreve um mapa na janela de texto,
;              copia-o para memória, e chama rotinas relevantes
;              quando encontra determinados caracteres.
; Parâmetro passado a esta rotina pela pilha:
;              M[SP+7] <- NivelX_L0 (X=1,2,3) ou GameOver_L0
;=================================================================

EscreveMapa:    DSI
                PUSH    R1
                PUSH    R2
                PUSH    R3
                PUSH    R4
                PUSH    R5
                MOV     R1, INI_Index
                MOV     M[PORTO_Ctrl], R1       ;Limpa a janela de texto
                MOV     R4, R0
                MOV     M[PORTO_Ctrl], R4       ;Coloca cursor na posição (0,0)
                MOV     R5, EspacoJogo
                MOV     M[NumPontos], R0        ;Num. pontos inicial = 0
                MOV     M[NumMonstros], R0      ;Num. monstros inicial = 0
                MOV     R3, M[SP+7]             ;Pos. da primeira célula do mapa
EscreveMais:    MOV     R1, M[R3]
                MOV     M[PORTO_Esc], R1        ;Escreve o caracter na janela
                MOV     M[R5], R1               ;Coloca o caracter no EspacoJogo
                CMP     R1, '@'                 ;Se for escrito o Pac-Man...
                CALL.Z  RegPosPacMan
                CMP     R1, '.'                 ;Se for escrito um ponto...
                CALL.Z  IncNumPontos
                CMP     R1, 'M'                 ;Se for escrito um monstro...
                CALL.Z  IncNumMonstros
                INC     R3                      ;Aponta para próximo caracter
                INC     R5                      ;Aponta para próxima posição
                                                ;no EspacoJogo
                MOV     R2, R0
                MVBL    R2, R4
                CMP     R2, N_COL_m1
                BR.Z    IncLinha                ;Se já estiver na última coluna
                MOV     R2, INC_Col
                ADD     R4, R2
                MOV     M[PORTO_Ctrl], R4
                JMP     EscreveMais

IncLinha:       MOV     R2, R0
                MVBH    R2, R4
                SHR     R2, 8
                CMP     R2, N_LIN_m1
                BR.Z    FimEscrita              ;Se escrita a última linha
                MOV     R2, INC_Lin
                ADD     R4, R2
                MOV     M[PORTO_Ctrl], R4       ;Incrementa linha
                MVBL    R4, R0
                MOV     M[PORTO_Ctrl], R4       ;Volta à coluna 0
                JMP     EscreveMais
```

```
FimEscrita:     POP     R5
                POP     R4
                POP     R3
                POP     R2
                POP     R1
                CALL    EscMensJogo
                MOV     R1, M[INI_Index]        ;Recoloca porto de estado a 0
                CALL    EsperaTecla
                CALL    ApagMensJogo
                ENI
                RETN    1

;===============================================================
; RegPosPacMan: Rotina que guarda em variáveis a posição inicial
;               do Pac-Man na janela de texto e em memória
; Parâmetros passados a esta rotina por registo:
;               R5 <- Posição do Pac-Man em memória
;               R4 <- Posição do Pac-Man na janela de texto
;===============================================================


RegPosPacMan:   MOV     M[PosPacManM], R5
                MOV     M[PosPacManT], R4
                RET

;===============================================================
; IncNumPontos: Rotina que incrementa o contador de número de
;               pontos '.' inicial num mapa
;===============================================================

IncNumPontos:   INC     M[NumPontos]
                RET

;===============================================================
; DecNumPontos: Rotina que decrementa o número de pontos no mapa
;               durante o jogo, após um ponto ter sido capturado
;===============================================================

DecNumPontos:   DEC     M[NumPontos]
                RET

;===============================================================
; IncNumMonstros: Rotina que incrementa o contador do número de
;                 monstros 'M' inicial num mapa, regista a
;                 posição deles na janela de texto e em memória
;                 e atribui-lhes uma direcção de movimento
;                 inicial. Também declara que os monstros se
;                 encontram em cima de uma casa vazia
; Parâmetros passados a esta rotina por registo:
;                 R5 <- Posição de um monstro em memória
;                 R4 <- Posição de um monstro na janela de texto
;===============================================================

IncNumMonstros: PUSH    R1
                PUSH    R2
                INC     M[NumMonstros]          ;+1 monstro no mapa
                MOV     R1, M[NumMonstros]      ;Verifica, dependendo do
                CMP     R1, 1                   ;número de monstros já tratados,
                BR.Z    RegPosMon1              ;qual vai ser o monstro a
                CMP     R1, 2                   ;tratar neste chamamento da
                BR.Z    RegPosMon2              ;rotina
                CMP     R1, 3
```



```
                JMP.Z   RegPosMon3
                CMP     R1, 4
                JMP.Z   RegPosMon4
                JMP     RegPosMon5

RegPosMon1:     MOV     M[PosMonstro1M], R5
                MOV     M[PosMonstro1T], R4
                MOV     R2, ' '
                MOV     M[BaixoMon1], R2
                PUSH    DirMonstro1
                CALL    RandomMon
                JMP     FimIncMons

RegPosMon2:     MOV     M[PosMonstro2M], R5
                MOV     M[PosMonstro2T], R4
                MOV     R2, ' '
                MOV     M[BaixoMon2], R2
                PUSH    DirMonstro2
                CALL    RandomMon
                JMP     FimIncMons

RegPosMon3:     MOV     M[PosMonstro3M], R5
                MOV     M[PosMonstro3T], R4
                MOV     R2, ' '
                MOV     M[BaixoMon3], R2
                PUSH    DirMonstro3
                CALL    RandomMon
                BR      FimIncMons

RegPosMon4:     MOV     M[PosMonstro4M], R5
                MOV     M[PosMonstro4T], R4
                MOV     R2, ' '
                MOV     M[BaixoMon4], R2
                PUSH    DirMonstro4
                CALL    RandomMon
                BR      FimIncMons

RegPosMon5:     MOV     M[PosMonstro5M], R5
                MOV     M[PosMonstro5T], R4
                MOV     R2, ' '
                MOV     M[BaixoMon5], R2
                PUSH    DirMonstro5
                CALL    RandomMon

FimIncMons:     POP     R2
                POP     R1
                RET

;===============================================================
; EscMensJogo: Rotina que escreve as mensagens iniciais no mapa
;              de jogo
;===============================================================

EscMensJogo:    PUSH    R1
                PUSH    R3
                PUSH    R4
                MOV     R1, M[CondGameOver]
                CMP     R1, ENABLE              ;Se estiver a escrever a
                                                ;animação "Game Over"
                JMP.Z   FimEscMenJog            ;não escreve a mensagem de jogo
EscMJogoL1:     MOV     R4, 011Dh               ;Começamos a escrever em (1,29)
                MOV     M[PORTO_Ctrl], R4
```

```
                MOV     R3, Menu_L3
CicloMJogoL1:   MOV     R1, M[R3]
                MOV     M[PORTO_Esc], R1        ;Escreve o caracter na janela
                CMP     R3, Fim_Menu_L3
                BR.Z    EscMJogoL7
                INC     R3
                INC     R4
                MOV     M[PORTO_Ctrl], R4       ;Põe o cursor na coluna seguinte
                BR      CicloMJogoL1
EscMJogoL7:     MOV     R4, 071Fh               ;Começamos a escrever em (7,31)
                MOV     M[PORTO_Ctrl], R4
                MOV     R3, MJogo_L7
CicloMJogoL7:   MOV     R1, M[R3]
                MOV     M[PORTO_Esc], R1        ;Escreve o caracter na janela
                CMP     R3, Fim_MJogo_L7
                BR.Z    EscNumNivel             ;Para escrever o num. do nível
                INC     R3
                INC     R4
                MOV     M[PORTO_Ctrl], R4       ;Põe ocursor na coluna seguinte
                BR      CicloMJogoL7
EscNumNivel:    INC     R4                      ;Põe o cursor a seguir a "Nivel"
                MOV     M[PORTO_Ctrl], R4       ;Põe o cursor na coluna seguinte
                MOV     R1, COD_Niv1
                CMP     M[MapaActual], R1       ;Verifica qual o mapa em que
                BR.Z    EscNumNivel1            ;se vai jogar, de modo a
                MOV     R1, COD_Niv2            ;escrever o número do nível
                CMP     M[MapaActual], R1       ;na mensagem de jogo
                BR.Z    EscNumNivel2
                MOV     R1, COD_Niv3
                CMP     M[MapaActual], R1
                BR.Z    EscNumNivel3
EscNumNivel1:   MOV     R1, M[MJogo_N1]
                MOV     M[PORTO_Esc], R1        ;Escreve 1 a seguir a "Nivel "
                BR      EscMJogoL10
EscNumNivel2:   MOV     R1, M[MJogo_N2]
                MOV     M[PORTO_Esc], R1        ;Escreve 2 a seguir a "Nivel "
                BR      EscMJogoL10
EscNumNivel3:   MOV     R1, M[MJogo_N3]
                MOV     M[PORTO_Esc], R1        ;Escreve 3 a seguir a "Nivel "
EscMJogoL10:    MOV     R4, 0A1Ah               ;Começamos a escrever em (10,26)
                MOV     M[PORTO_Ctrl], R4
                MOV     R3, MJogo_L10
CicloMJogoL10:  MOV     R1, M[R3]
                MOV     M[PORTO_Esc], R1        ;Escreve o caracter na janela
                CMP     R3, Fim_MJogo_L10
                BR.Z    EscMJogoL11
                INC     R3
                INC     R4
                MOV     M[PORTO_Ctrl], R4       ;Põe o cursor na coluna seguinte
                BR      CicloMJogoL10
EscMJogoL11:    MOV     R4, 0B1Ah               ;Começamos a escrever em (11,26)
                MOV     M[PORTO_Ctrl], R4
                MOV     R3, MJogo_L11
CicloMJogoL11:  MOV     R1, M[R3]
                MOV     M[PORTO_Esc], R1        ;Escreve o caracter na janela
                CMP     R3, Fim_MJogo_L11
                BR.Z    FimEscMenJog
                INC     R3
                INC     R4
                MOV     M[PORTO_Ctrl], R4       ;Põe cursor na coluna seguinte
                BR      CicloMJogoL11
FimEscMenJog:   POP     R4
```



```
                POP     R3
                POP     R1
                RET

;=================================================================
; ApagMensJogo: Rotina que apaga a mensagem
;               "Premir uma tecla para iniciar o jogo."
;=================================================================

ApagMensJogo:   PUSH    R1
                PUSH    R3
                PUSH    R4
                MOV     R1, ' '                 ;R1 conterá caracter 'espaço'
ApaMJogoL10:    MOV     R4, 0A1Ah               ;Começamos a apagar em (10,26)
                MOV     M[PORTO_Ctrl], R4
                MOV     R3, MJogo_L10
CApaMJogoL10:   MOV     M[PORTO_Esc], R1        ;Escreve um espaço na janela
                CMP     R3, Fim_MJogo_L10       ;Já apagámos a linha toda?
                BR.Z    ApaMJogoL11
                INC     R3
                INC     R4
                MOV     M[PORTO_Ctrl], R4       ;Põe o cursor na coluna seguinte
                BR      CApaMJogoL10
ApaMJogoL11:    MOV     R4, 0B1Ah               ;Começamos a apagar em (11,26)
                MOV     M[PORTO_Ctrl], R4
                MOV     R3, MJogo_L11
CApaMJogoL11:   MOV     M[PORTO_Esc], R1        ;Escreve um espaço na janela
                CMP     R3, Fim_MJogo_L11       ;Já apagámos a linha toda?
                BR.Z    FimApaMenJog
                INC     R3
                INC     R4
                MOV     M[PORTO_Ctrl], R4       ;Põe o cursor na coluna seguinte
                BR      CApaMJogoL11
FimApaMenJog:   POP     R4
                POP     R3
                POP     R1
                RET

;=================================================================

; EscreveMenu: Rotina que escreve o menu principal do jogo
;              na janela de texto
;=================================================================

EscreveMenu:    PUSH    R1
                PUSH    R3
                PUSH    R4
                MOV     R1, INI_Index
                MOV     M[PORTO_Ctrl], R1       ;Limpa a janela de texto
EscMenuL3:      MOV     R4, 0300h               ;Começamos a escrever na linha 3
                MOV     M[PORTO_Ctrl], R4
                MOV     R3, Menu_L3
CicloMenuL3:    MOV     R1, M[R3]
                MOV     M[PORTO_Esc], R1        ;Escreve o caracter na janela
                CMP     R3, Fim_Menu_L3         ;Já escrevemos a linha toda?
                BR.Z    EscMenuL5
                INC     R3
                INC     R4
                MOV     M[PORTO_Ctrl], R4       ;Põe o cursor na coluna seguinte
                BR      CicloMenuL3
EscMenuL5:      MOV     R4, 0500h               ;Começamos a escrever na linha 5
                MOV     M[PORTO_Ctrl], R4
```

```
                MOV     R3, Menu_L5
CicloMenuL5:    MOV     R1, M[R3]
                MOV     M[PORTO_Esc], R1        ;Escreve o caracter na janela
                CMP     R3, Fim_Menu_L5         ;Já escrevemos a linha toda?
                BR.Z    EscMenuL7
                INC     R3
                INC     R4
                MOV     M[PORTO_Ctrl], R4       ;Põe o cursor na coluna seguinte
                BR      CicloMenuL5
EscMenuL7:      CALL    EscrevePont             ;Escreve o high score na janela
                MOV     R4, 0700h               ;Começamos a escrever na linha 7
                MOV     M[PORTO_Ctrl], R4
                MOV     R3, Menu_L7
CicloMenuL7:    MOV     R1, M[R3]
                MOV     M[PORTO_Esc], R1        ;Escreve o caracter na janela
                CMP     R3, Fim_Menu_L7         ;Já escrevemos a linha toda?
                BR.Z    EscMenuL8
                INC     R3
                INC     R4
                MOV     M[PORTO_Ctrl], R4       ;Põe o cursor na coluna seguinte
                BR      CicloMenuL7
EscMenuL8:      MOV     R4, 0800h               ;Começamos a escrever na linha 7
                MOV     M[PORTO_Ctrl], R4
                MOV     R3, Menu_L8
CicloMenuL8:    MOV     R1, M[R3]
                MOV     M[PORTO_Esc], R1        ;Escreve o caracter na janela
                CMP     R3, Fim_Menu_L8         ;Já escrevemos a linha toda?
                BR.Z    FimEscMenu
                INC     R3
                INC     R4
                MOV     M[PORTO_Ctrl], R4       ;Põe o cursor na coluna seguinte
                BR      CicloMenuL8
FimEscMenu:     POP     R4
                POP     R3
                POP     R1
                RET


;=============================================================
; EscrevePont: Rotina que escreve a pontuação máxima à frente da
;              mensagem "Pontuacao maxima: " no menu principal.
;              A pontuação é transformada em caracteres ASCII
;              e escrita na janela de texto
;=============================================================

EscrevePont:    PUSH    R1
                PUSH    R2
                PUSH    R3
                PUSH    R4
                PUSH    R5
                PUSH    R6
                MOV     R1, M[PontMaxima]       ;Obtém a pontuação máxima em R1
                MOV     R2, 10
                DIV     R1, R2                  ;Primeiro dígito menos
                                                ;significativo em R2
                MOV     R3, R2
                ADD     R3, ASCII_Int           ;Código ASCII desse dígito em R3
                MOV     R2, 10
                DIV     R1, R2                  ;Segundo dígito menos
                                                ;significativo em R2
                MOV     R4, R2
                ADD     R4, ASCII_Int           ;Código ASCII desse dígito em R4
```



```
                MOV     R2, 10
                DIV     R1, R2                  ;Terceiro dígito menos
                                                ;significativo em R2
                MOV     R5, R2
                ADD     R5, ASCII_Int           ;Código ASCII desse dígito em R5
                MOV     R2, 10
                DIV     R1, R2                  ;Dígito mais significativo em R2
                MOV     R6, R2
                ADD     R6, ASCII_Int           ;Código ASCII desse dígito em R6
EscritaPont:    MOV     R1, 0512h               ;Escrever na linha 5, coluna 18
                MOV     M[PORTO_Ctrl], R1
                MOV     M[PORTO_Esc], R6        ;Escrevemos o dígito
                                                ;mais significativo
                INC     R1                      ;Saltamos para a coluna seguinte
                MOV     M[PORTO_Ctrl], R1
                MOV     M[PORTO_Esc], R5        ;Escrevemos o terceiro dígito
                                                ;menos significativo
                INC     R1                      ;Saltamos para a coluna seguinte
                MOV     M[PORTO_Ctrl], R1
                MOV     M[PORTO_Esc], R4        ;Escrevemos o segundo dígito
                                                ;menos significativo
                INC     R1                      ;Saltamos para a coluna seguinte
                MOV     M[PORTO_Ctrl], R1
                MOV     M[PORTO_Esc], R3        ;Escrevemos o primeiro dígito
                                                ;menos significativo
                POP     R6
                POP     R5
                POP     R4
                POP     R3
                POP     R2
                POP     R1
                RET

;=============================================================
; EscreveLCD: Rotina que escreve a mensagem "Jogo Pac-Man"
;             no display LCD da janela de placa
;             (funcionalidade extra)
;=============================================================

EscreveLCD:     PUSH    R1
                PUSH    R2
                PUSH    R3
                MOV     R2, Menu_L3             ;Endereço da mensagem a escrever
                MOV     R1, LCD_Ini
EscMaisLCD:     MOV     M[LCD_Ctrl], R1         ;Liga display, posiciona cursor

                MOV     R3, M[R2]
                MOV     M[LCD_Esc], R3          ;Escreve caracter no display
                CMP     R2, Fim_Menu_L3         ;Já escreveu tudo?
                BR.Z    FimEscLCD
                INC     R2                      ;Aponta para próximo caracter
                INC     R1                      ;Escreve-se na coluna seguinte
                BR      EscMaisLCD
FimEscLCD:      POP     R3
                POP     R2
                POP     R1
                RET

;=============================================================
; ActMapa: Rotina que actualiza mapa de jogo tendo em conta
;          movimentos, pontos comidos, etc.
;          Para o movimento dos monstros, os parâmetros são
```

```
;          passados às rotinas MonCima, MonBaixo, MonEsq e
;          MonDir pela pilha
;===============================================================

ActMapa:        DSI
                PUSH    R1
TrataMovPacMan: MOV     R1, M[Direccao]
                CMP     R1, CIMA                ;Verifica a direcção actual
                CALL.Z  PacCima                 ;do Pac-Man para saber qual
                CMP     R1, BAIXO               ;a sub-rotina a chamar
                CALL.Z  PacBaixo
                CMP     R1, ESQ
                CALL.Z  PacEsq
                CMP     R1, DIR
                CALL.Z  PacDir
TrataMovMons1:  MOV     R1, M[DirMonstro1]
                PUSH    DirMonstro1             ;Passa os parâmetros referidos
                PUSH    BaixoMon1               ;acima...
                PUSH    PosMonstro1M
                PUSH    PosMonstro1T
                CMP     R1, CIMA                ;...para a sub-rotina
                CALL.Z  MonCima                 ;correspondente à direcção
                CMP     R1, BAIXO               ;do monstro 1
                CALL.Z  MonBaixo
                CMP     R1, ESQ
                CALL.Z  MonEsq
                CMP     R1, DIR
                CALL.Z  MonDir
                MOV     R1, M[PosMonstro2T]
                CMP     R1, R0
                JMP.Z   FimActMapa
TrataMovMons2:  MOV     R1, M[DirMonstro2]
                PUSH    DirMonstro2             ;Passa os parâmetros referidos
                PUSH    BaixoMon2               ;acima...
                PUSH    PosMonstro2M
                PUSH    PosMonstro2T
                CMP     R1, CIMA                ;...para a sub-rotina
                CALL.Z  MonCima                 ;correspondente à direcção
                CMP     R1, BAIXO               ;do monstro 2
                CALL.Z  MonBaixo
                CMP     R1, ESQ
                CALL.Z  MonEsq
                CMP     R1, DIR
                CALL.Z  MonDir
                MOV     R1, M[PosMonstro3T]
                CMP     R1, R0
                JMP.Z   FimActMapa
TrataMovMons3:  MOV     R1, M[DirMonstro3]
                PUSH    DirMonstro3             ;Passa os parâmetros referidos
                PUSH    BaixoMon3               ;acima...
                PUSH    PosMonstro3M
                PUSH    PosMonstro3T
                CMP     R1, CIMA                ;...para a sub-rotina
                CALL.Z  MonCima                 ;correspondente à direcção
                CMP     R1, BAIXO               ;do monstro 3
                CALL.Z  MonBaixo
                CMP     R1, ESQ
                CALL.Z  MonEsq
                CMP     R1, DIR
                CALL.Z  MonDir
                MOV     R1, M[PosMonstro4T]
                CMP     R1, R0
```



```
                JMP.Z   FimActMapa
TrataMovMons4:  MOV     R1, M[DirMonstro4]
                PUSH    DirMonstro4             ;Passa os parâmetros referidos
                PUSH    BaixoMon4               ;acima...
                PUSH    PosMonstro4M
                PUSH    PosMonstro4T
                CMP     R1, CIMA                ;...para a sub-rotina
                CALL.Z  MonCima                 ;correspondente à direcção
                CMP     R1, BAIXO               ;do monstro 4
                CALL.Z  MonBaixo
                CMP     R1, ESQ
                CALL.Z  MonEsq
                CMP     R1, DIR
                CALL.Z  MonDir
                MOV     R1, M[PosMonstro5T]
                CMP     R1, R0
                BR.Z    FimActMapa
TrataMovMons5:  MOV     R1, M[DirMonstro5]
                PUSH    DirMonstro5             ;Passa os parâmetros referidos
                PUSH    BaixoMon5               ;acima...
                PUSH    PosMonstro5M
                PUSH    PosMonstro5T
                CMP     R1, CIMA                ;...para a sub-rotina
                CALL.Z  MonCima                 ;correspondente à direcção
                CMP     R1, BAIXO               ;do monstro 5
                CALL.Z  MonBaixo
                CMP     R1, ESQ
                CALL.Z  MonEsq
                CMP     R1, DIR
                CALL.Z  MonDir

FimActMapa:     POP     R1
                ENI
                RET

; MOVIMENTO PAC-MAN PARA CIMA
;===============================================================
; Conjunto de rotinas que prevêem e tratam o movimento ascendente
; do Pac-Man
;===============================================================

PacCima:        PUSH    R2
                PUSH    R3
                PUSH    R4
                PUSH    R5
                PUSH    R6
                PUSH    R7
                MOV     R2, M[PosPacManM]
                SUB     R2, N_COL
                MOV     R3, M[R2]               ;R3 contém agora a posição
                                                ;acima do Pac-Man
                CMP     R3, 'M'                 ;Se essa posição for um monstro
                CALL.Z  PacCapMonstro           ;chama a interacção com monstro
                CMP     R3, '#'                 ;Caso contrário, e se não for
                CALL.NZ PacCimaGeral            ;uma parede, chama caso geral
                POP     R7
                POP     R6
                POP     R5
                POP     R4
                POP     R3
                POP     R2
                RET
```

```
PacCimaGeral:   MOV     R4, M[PosPacManT]
                MOV     R5, ' '
                MOV     R6, M[PosPacManM]
                MOV     R7, '@'
                MOV     M[PORTO_Ctrl], R4
                MOV     M[PORTO_Esc], R5        ;Escreve um espaço na posição
                                                ;que o Pac-man deixa livre
                MOV     M[R6], R5               ;Escreve um espaço na posição
                                                ;de memória que o pacman...
                SUB     R4, INC_Lin
                SUB     R6, N_COL
                MOV     M[PosPacManT], R4       ;Actualiza a posição do
                                                ;Pac-Man na janela
                MOV     M[PosPacManM], R6       ;Actualiza a posição do
                                                ;Pac-Man em memória
                MOV     M[PORTO_Ctrl], R4
                MOV     M[PORTO_Esc], R7        ;Escreve Pac-Man na nova
                                                ;posição na Janela
                MOV     M[R6], R7               ;Escreve Pac-Man na nova
                                                ;posição em Memória
                CMP     R3, '.'
                CALL.Z  PacCapPonto
                CMP     R3, ')'
                CALL.Z  PacCapBanana
                CMP     R3, '&'
                CALL.Z  PacCapPera
                CMP     R3, '%'
                CALL.Z  PacCapGelado
                RET

; MOVIMENTO PAC-MAN PARA BAIXO
;=============================================================
; Conjunto de rotinas que prevêem e tratam o movimento
; descendente do Pac-Man
;=============================================================

PacBaixo:       PUSH    R2
                PUSH    R3
                PUSH    R4
                PUSH    R5
                PUSH    R6
                PUSH    R7
                MOV     R2, M[PosPacManM]
                ADD     R2, N_COL
                MOV     R3, M[R2]               ;R3 contém agora a posição
                                                ;abaixo do Pac-Man
                CMP     R3, 'M'                 ;Se essa posição for um monstro
                CALL.Z  PacCapMonstro           ;chama a interacção com monstro
                CMP     R3, '#'                 ;Caso contrário, e se não for
                CALL.NZ PacBaixoGeral           ;uma parede, chama caso geral
                POP     R7
                POP     R6
                POP     R5
                POP     R4
                POP     R3
                POP     R2
                RET


PacBaixoGeral:  MOV     R4, M[PosPacManT]
```



```
                MOV     R5, ' '
                MOV     R6, M[PosPacManM]
                MOV     R7, '@'
                MOV     M[PORTO_Ctrl], R4
                MOV     M[PORTO_Esc], R5        ;Escreve um espaço na posição
                                                ;que o Pac-man deixa livre
                MOV     M[R6], R5               ;Escreve um espaço na posição
                                                ;de memória que o pacman...
                ADD     R4, INC_Lin
                ADD     R6, N_COL
                MOV     M[PosPacManT], R4       ;Actualiza a posição do
                                                ;Pac-Man na janela
                MOV     M[PosPacManM], R6       ;Actualiza a posição do
                                                ;Pac-Man em memória
                MOV     M[PORTO_Ctrl], R4
                MOV     M[PORTO_Esc], R7        ;Escreve Pac-Man na nova
                                                ;posição na Janela
                MOV     M[R6], R7               ;Escreve Pac-Man na nova
                                                ;posição em Memória
                CMP     R3, '.'
                CALL.Z  PacCapPonto
                CMP     R3, ')'
                CALL.Z  PacCapBanana
                CMP     R3, '&'
                CALL.Z  PacCapPera
                CMP     R3, '%'
                CALL.Z  PacCapGelado
                RET

; MOVIMENTO PAC-MAN PARA A ESQUERDA
;=============================================================
; Conjunto de rotinas que prevêem e tratam o movimento lateral
; esquerdo do Pac-Man
;=============================================================

PacEsq:         PUSH    R2
                PUSH    R3
                PUSH    R4
                PUSH    R5
                PUSH    R6
                PUSH    R7
                MOV     R2, M[PosPacManM]
                DEC     R2
                MOV     R3, M[R2]               ;R3 contém agora a posição
                                                ;à esquerda do Pac-Man
                CMP     R3, 'M'                 ;Se essa posição for um monstro
                CALL.Z  PacCapMonstro           ;chama a interacção com monstro
                CMP     R3, '#'                 ;Caso contrário, e se não for
                CALL.NZ PacEsqGeral             ;uma parede, chama caso geral
                POP     R7
                POP     R6
                POP     R5
                POP     R4
                POP     R3
                POP     R2
                RET

PacEsqGeral:    MOV     R4, M[PosPacManT]
                MOV     R5, ' '
                MOV     R6, M[PosPacManM]       ;Saltamos para a coluna seguinte
                MOV     R7, '@'
                MOV     M[PORTO_Ctrl], R4
```

```
                MOV     M[PORTO_Esc], R5        ;Escreve um espaço na posição
                                                ;que o Pac-man deixa livre
                MOV     M[R6], R5               ;Escreve um espaço na posição
                                                ;de memória que o pacman...
                SUB     R4, INC_Col
                DEC     R6
                MOV     M[PosPacManT], R4       ;Actualiza a posição do
                                                ;Pac-Man na janela
                MOV     M[PosPacManM], R6       ;Actualiza a posição do
                                                ;Pac-Man em memória
                MOV     M[PORTO_Ctrl], R4
                MOV     M[PORTO_Esc], R7        ;Escreve Pac-Man na nova
                                                ;posição na Janela
                MOV     M[R6], R7               ;Escreve Pac-Man na nova
                                                ;posição em Memória
                CMP     R3, '.'
                CALL.Z  PacCapPonto
                CMP     R3, ')'
                CALL.Z  PacCapBanana
                CMP     R3, '&'
                CALL.Z  PacCapPera
                CMP     R3, '%'
                CALL.Z  PacCapGelado
                RET

; MOVIMENTO PAC-MAN PARA A DIREITA
;=============================================================
; Conjunto de rotinas que prevêem e tratam o movimento lateral
; direito do Pac-Man
;=============================================================

PacDir:         PUSH    R2
                PUSH    R3
                PUSH    R4
                PUSH    R5
                PUSH    R6
                PUSH    R7
                MOV     R2, M[PosPacManM]
                INC     R2
                MOV     R3, M[R2]               ;R3 contém agora a posição
                                                ;à direita do Pac-Man
                CMP     R3, 'M'                 ;Se essa posição for um monstro
                CALL.Z  PacCapMonstro           ;chama a interacção com monstro
                CMP     R3, '#'                 ;Caso contrário, e se não for
                CALL.NZ PacDirGeral             ;uma parede, chama caso geral
                POP     R7
                POP     R6
                POP     R5
                POP     R4
                POP     R3
                POP     R2
                RET

PacDirGeral:    MOV     R4, M[PosPacManT]
                MOV     R5, ' '
                MOV     R6, M[PosPacManM]
                MOV     R7, '@'
                MOV     M[PORTO_Ctrl], R4
                MOV     M[PORTO_Esc], R5        ;Escreve um espaço na posição
                                                ;que o Pac-man deixa livre
                MOV     M[R6], R5               ;Escreve um espaço na posição
                                                ;de memória que o pacman...
```



```
                ADD     R4, INC_Col
                INC     R6
                MOV     M[PosPacManT], R4       ;Actualiza a posição do
                                                ;Pac-Man na janela
                MOV     M[PosPacManM], R6       ;Actualiza a posição do
                                                ;Pac-Man em memória
                MOV     M[PORTO_Ctrl], R4
                MOV     M[PORTO_Esc], R7        ;Escreve Pac-Man na nova
                                                ;posição na Janela
                MOV     M[R6], R7               ;Escreve Pac-Man na nova
                                                ;posição em Memória
                CMP     R3, '.'
                CALL.Z  PacCapPonto
                CMP     R3, ')'
                CALL.Z  PacCapBanana
                CMP     R3, '&'
                CALL.Z  PacCapPera
                CMP     R3, '%'
                CALL.Z  PacCapGelado
                RET

; ROTINAS GERAIS DE MOVIMENTO DO PAC-MAN
;=============================================================
; Quando à frente do Pac-Man há pontos ou bónus...
;=============================================================

PacCapPonto:    PUSH    PtsPonto                ;Passa núm. pontos pela pilha
                CALL    AumPontos
                CALL    DecNumPontos
                RET

PacCapBanana:   PUSH    PtsBanana               ;Passa núm. pontos pela pilha
                CALL    AumPontos
                RET

PacCapPera:     PUSH    PtsPera                 ;Passa núm. pontos pela pilha
                CALL    AumPontos
                RET

PacCapGelado:   PUSH    PtsGelado               ;Passa núm. pontos pela pilha
                CALL    AumPontos
                RET

;=============================================================
; Quando à frente do Pac-Man há um monstro...
;=============================================================

PacCapMonstro:  MOV     R4, M[PosPacManT]
                MOV     R5, ' '
                MOV     R6, M[PosPacManM]
                MOV     M[PORTO_Ctrl], R4
                MOV     M[PORTO_Esc], R5        ;Apaga o Pac-Man, simbolizando
                                                ;a sua morte
                MOV     R5, ENABLE
                MOV     M[CondGameOver], R5     ;Activa a var. condição para
                                                ;o "Game Over"
                RET

; MOVIMENTO MONSTRO PARA CIMA
;=============================================================
; Conjunto de rotinas que prevêem e tratam o movimento ascendente
; dum Monstro.
```

```
; Parâmetros passados a estas rotinas pela pilha:
;               M[SP+11] <- DirMonstroX
;               M[SP+10] <- BaixoMonX
;               M[SP+9]  <- PosMonstroXM
;               M[SP+8]  <- PosMonstroXT
;===============================================================

MonCima:        PUSH    R2
                PUSH    R3
                PUSH    R4
                PUSH    R5
                PUSH    R6
                PUSH    R7
                MOV     R2, M[SP+9]
                MOV     R2, M[R2]
                SUB     R2, N_COL
                MOV     R3, M[R2]               ;R3 contém agora a posição
                                                ;acima do Monstro
                CMP     R3, 'M'                 ;Se nessa posição estiver um
                JMP.Z   MonCimaObstac           ;monstro ou uma parede...
                CMP     R3, '#'
                JMP.Z   MonCimaObstac
                CMP     R3, '@'                 ;Se nessa posição estiver o
                JMP.Z   MonCimaPacMan           ;Pac-Man...

MonCimaGeral:   MOV     R4, M[SP+8]             ;...caso contrário, é executado
                MOV     R4, M[R4]               ;o caso geral
                MOV     R5, M[SP+10]
                MOV     R5, M[R5]
                MOV     R6, M[SP+9]
                MOV     R6, M[R6]
                MOV     M[PORTO_Ctrl], R4
                MOV     M[PORTO_Esc], R5        ;Restora o conteúdo da posição
                                                ;que o Monstro deixa livre
                MOV     M[R6], R5               ;Restora a posição de memória
                                                ;que o Monstro deixa livre
                MOV     R7, R4
                SUB     R7, INC_Lin
                MOV     M[PORTO_Ctrl], R7
                MOV     R4, M[SP+8]
                MOV     M[R4], R7               ;Actualiza a posição do
                                                ;Monstro na janela
                MOV     R7, R6
                SUB     R7, N_COL
                MOV     R6, M[SP+9]
                MOV     M[R6], R7               ;Actualiza a posição do
                                                ;Monstro em memória
                MOV     R5, M[SP+10]
                MOV     M[R5], R3               ;Actualiza o que fica por
                                                ;baixo do Monstro

                MOV     R5, R7
                MOV     R7, 'M'
                MOV     M[PORTO_Esc], R7        ;Escreve Monstro na nova
                                                ;posição na Janela
                MOV     M[R5], R7               ;Escreve Monstro na nova
                                                ;posição em Memória
                JMP     FimMonCima

MonCimaObstac:  MOV     R4, M[SP+11]
                PUSH    R4
                CALL    RandomMon
```



```
                JMP     FimMonCima

MonCimaPacMan:  MOV     R4, M[SP+8]
                MOV     R4, M[R4]
                MOV     R5, M[SP+10]
                MOV     R5, M[R5]
                MOV     R6, M[SP+9]
                MOV     R6, M[R6]
                MOV     M[PORTO_Ctrl], R4
                MOV     M[PORTO_Esc], R5        ;Restora o conteúdo da posição
                                                ;que o Monstro deixa livre
                MOV     M[R6], R5               ;Restora a posição de memória
                                                ;que o Monstro deixa livre
                MOV     R7, R4
                SUB     R7, INC_Lin
                MOV     M[PORTO_Ctrl], R7
                MOV     R4, M[SP+8]
                MOV     M[R4], R7               ;Actualiza a posição do
                                                ;Monstro na janela
                MOV     R7, R6
                SUB     R7, N_COL
                MOV     R6, M[SP+9]
                MOV     M[R6], R7               ;Actualiza a posição do
                                                ;Monstro em memória
                MOV     R5, R7
                MOV     R7, 'M'
                MOV     M[PORTO_Esc], R7        ;Escreve Monstro em cima do
                                                ;Pac-Man na Janela
                MOV     M[R5], R7               ;Escreve Monstro em cima do
                                                ;Pac-Man em Memória
                MOV     R5, ENABLE
                MOV     M[CondGameOver], R5     ;Activa a var. condição para
                                                ;o "Game Over"

FimMonCima:     POP     R7
                POP     R6
                POP     R5
                POP     R4
                POP     R3
                POP     R2
                RETN    4

; MOVIMENTO MONSTRO PARA BAIXO
;===============================================================
; Conjunto de rotinas que prevêem e tratam o movimento
; descendente dum Monstro.
; Parâmetros passados a estas rotinas pela pilha:
;               M[SP+11] <- DirMonstroX
;               M[SP+10] <- BaixoMonX
;               M[SP+9]  <- PosMonstroXM
;               M[SP+8]  <- PosMonstroXT
;===============================================================

MonBaixo:       PUSH    R2
                PUSH    R3
                PUSH    R4
                PUSH    R5
                PUSH    R6
                PUSH    R7
                MOV     R2, M[SP+9]
                MOV     R2, M[R2]
                ADD     R2, N_COL
```

```
                MOV     R3, M[R2]               ;R3 contém agora a posição
                                                ;abaixo do Monstro
                CMP     R3, 'M'                 ;Se nessa posição estiver um
                JMP.Z   MonBaixoObstac          ;monstro ou uma parede...
                CMP     R3, '#'
                JMP.Z   MonBaixoObstac
                CMP     R3, '@'                 ;Se nessa posição estiver o
                JMP.Z   MonBaixoPacMan          ;Pac-Man...

MonBaixoGeral:  MOV     R4, M[SP+8]             ;...caso contrário, é executado
                MOV     R4, M[R4]               ;o caso geral
                MOV     R5, M[SP+10]
                MOV     R5, M[R5]
                MOV     R6, M[SP+9]
                MOV     R6, M[R6]
                MOV     M[PORTO_Ctrl], R4
                MOV     M[PORTO_Esc], R5        ;Restora o conteúdo da posição
                                                ;que o Monstro deixa livre
                MOV     M[R6], R5               ;Restora a posição de memória
                                                ;que o Monstro deixa livre
                MOV     R7, R4
                ADD     R7, INC_Lin
                MOV     M[PORTO_Ctrl], R7
                MOV     R4, M[SP+8]
                MOV     M[R4], R7               ;Actualiza a posição do Monstro
                                                ;na janela
                MOV     R7, R6
                ADD     R7, N_COL
                MOV     R6, M[SP+9]
                MOV     M[R6], R7               ;Actualiza a posição do
                                                ;Monstro em memória
                MOV     R5, M[SP+10]
                MOV     M[R5], R3               ;Actualiza o que fica por
                                                ;baixo do Monstro

                MOV     R5, R7
                MOV     R7, 'M'
                MOV     M[PORTO_Esc], R7        ;Escreve Monstro na nova
                                                ;posição na Janela
                MOV     M[R5], R7               ;Escreve Monstro na nova
                                                ;posição em Memória
                JMP     FimMonBaixo

MonBaixoObstac: MOV     R4, M[SP+11]
                PUSH    R4
                CALL    RandomMon
                JMP     FimMonBaixo

MonBaixoPacMan: MOV     R4, M[SP+8]
                MOV     R4, M[R4]
                MOV     R5, M[SP+10]
                MOV     R5, M[R5]
                MOV     R6, M[SP+9]
                MOV     R6, M[R6]
                MOV     M[PORTO_Ctrl], R4
                MOV     M[PORTO_Esc], R5        ;Restora o conteúdo da posição
                                                ;que o Monstro deixa livre
                MOV     M[R6], R5               ;Restora a posição de memória
                                                ;que o Monstro deixa livre
                MOV     R7, R4
                ADD     R7, INC_Lin
                MOV     M[PORTO_Ctrl], R7
```



```
                MOV     R4, M[SP+8]
                MOV     M[R4], R7               ;Actualiza a posição do
                                                ;Monstro na janela
                MOV     R7, R6
                ADD     R7, N_COL
                MOV     R6, M[SP+9]
                MOV     M[R6], R7               ;Actualiza a posição do
                                                ;Monstro em memória
                MOV     R5, R7
                MOV     R7, 'M'
                MOV     M[PORTO_Esc], R7        ;Escreve Monstro em cima
                                                ;do Pac-Man na Janela
                MOV     M[R5], R7               ;Escreve Monstro em cima
                                                ;do Pac-Man em Memória
                MOV     R5, ENABLE
                MOV     M[CondGameOver], R5     ;Activa a var. condição
                                                ;para o "Game Over"

FimMonBaixo:    POP     R7
                POP     R6
                POP     R5
                POP     R4
                POP     R3
                POP     R2
                RETN    4

; MOVIMENTO MONSTRO PARA A ESQUERDA
;==============================================================
; Conjunto de rotinas que prevêem e tratam o movimento lateral
; esquerdo dum Monstro.
; Parâmetros passados a estas rotinas pela pilha:
;               M[SP+11] <- DirMonstroX
;               M[SP+10] <- BaixoMonX
;               M[SP+9]  <- PosMonstroXM
;               M[SP+8]  <- PosMonstroXT
;==============================================================

MonEsq:         PUSH    R2
                PUSH    R3
                PUSH    R4
                PUSH    R5
                PUSH    R6
                PUSH    R7
                MOV     R2, M[SP+9]
                MOV     R2, M[R2]
                DEC     R2
                MOV     R3, M[R2]               ;R3 contém agora a posição
                                                ;à esquerda do Monstro
                CMP     R3, 'M'                 ;Se nessa posição estiver um
                JMP.Z   MonEsqObstac            ;monstro ou uma parede...
                CMP     R3, '#'
                JMP.Z   MonEsqObstac
                CMP     R3, '@'                 ;Se nessa posição estiver o
                JMP.Z   MonEsqPacMan            ;Pac-Man...

MonEsqGeral:    MOV     R4, M[SP+8]             ;...caso contrário, é executado
                MOV     R4, M[R4]               ;o caso geral
                MOV     R5, M[SP+10]
                MOV     R5, M[R5]
                MOV     R6, M[SP+9]
                MOV     R6, M[R6]
                MOV     M[PORTO_Ctrl], R4
```

```
                MOV     M[PORTO_Esc], R5        ;Restora o conteúdo da posição
                                                ;que o Monstro deixa livre
                MOV     M[R6], R5               ;Restora a posição de memória
                                                ;que o Monstro deixa livre
                MOV     R7, R4
                SUB     R7, INC_Col
                MOV     M[PORTO_Ctrl], R7
                MOV     R4, M[SP+8]
                MOV     M[R4], R7               ;Actualiza a posição do
                                                ;Monstro na janela
                MOV     R7, R6
                DEC     R7
                MOV     R6, M[SP+9]
                MOV     M[R6], R7               ;Actualiza a posição do
                                                ;Monstro em memória
                MOV     R5, M[SP+10]
                MOV     M[R5], R3               ;Actualiza o que fica por
                                                ;baixo do Monstro

                MOV     R5, R7
                MOV     R7, 'M'
                MOV     M[PORTO_Esc], R7        ;Escreve Monstro na nova
                                                ;posição na Janela
                MOV     M[R5], R7               ;Escreve Monstro na nova
                                                ;posição em Memória
                JMP     FimMonEsq

MonEsqObstac:   MOV     R4, M[SP+11]
                PUSH    R4
                CALL    RandomMon
                JMP     FimMonEsq

MonEsqPacMan:   MOV     R4, M[SP+8]
                MOV     R4, M[R4]
                MOV     R5, M[SP+10]
                MOV     R5, M[R5]
                MOV     R6, M[SP+9]
                MOV     R6, M[R6]
                MOV     M[PORTO_Ctrl], R4
                MOV     M[PORTO_Esc], R5        ;Restora o conteúdo da posição
                                                ;que o Monstro deixa livre
                MOV     M[R6], R5               ;Restora a posição de memória
                                                ;que o Monstro deixa livre
                MOV     R7, R4
                SUB     R7, INC_Col
                MOV     M[PORTO_Ctrl], R7
                MOV     R4, M[SP+8]
                MOV     M[R4], R7               ;Actualiza a posição do
                                                ;Monstro na janela
                MOV     R7, R6
                DEC     R7
                MOV     R6, M[SP+9]
                MOV     M[R6], R7               ;Actualiza a posição do
                                                ;Monstro em memória
                MOV     R5, M[SP+10]
                MOV     R7, 'M'
                MOV     M[PORTO_Esc], R7        ;Escreve Monstro em cima
                                                ;do Pac-Man na Janela
                MOV     M[R5], R7               ;Escreve Monstro em cima
                                                ;do Pac-Man em Memória
                MOV     R5, ENABLE
                MOV     M[CondGameOver], R5     ;Activa a var. condição para
```



```
                                                ;o "Game Over"

FimMonEsq:      POP     R7
                POP     R6
                POP     R5
                POP     R4
                POP     R3
                POP     R2
                RETN    4

; MOVIMENTO MONSTRO PARA A DIREITA
;===============================================================
; Conjunto de rotinas que prevêem e tratam o movimento lateral
; direito dum Monstro.
; Parâmetros passados a estas rotinas pela pilha:
;               M[SP+11] <- DirMonstroX
;               M[SP+10] <- BaixoMonX
;               M[SP+9]  <- PosMonstroXM
;               M[SP+8]  <- PosMonstroXT
;===============================================================

MonDir:         PUSH    R2
                PUSH    R3
                PUSH    R4
                PUSH    R5
                PUSH    R6
                PUSH    R7
                MOV     R2, M[SP+9]
                MOV     R2, M[R2]
                INC     R2
                MOV     R3, M[R2]               ;R3 contém agora a posição
                                                ;à direita do Monstro
                CMP     R3, 'M'                 ;Se nessa posição estiver um
                JMP.Z   MonDirObstac            ;monstro ou uma parede...
                CMP     R3, '#'
                JMP.Z   MonDirObstac
                CMP     R3, '@'                 ;Se nessa posição estiver o
                JMP.Z   MonDirPacMan            ;Pac-Man...

MonDirGeral:    MOV     R4, M[SP+8]             ;...caso contrário, é executado
                MOV     R4, M[R4]               ;o caso geral
                MOV     R5, M[SP+10]
                MOV     R5, M[R5]
                MOV     R6, M[SP+9]
                MOV     R6, M[R6]
                MOV     M[PORTO_Ctrl], R4
                MOV     M[PORTO_Esc], R5        ;Restora o conteúdo da posição
                                                ;que o Monstro deixa livre
                MOV     M[R6], R5               ;Restora a posição de memória
                                                ;que o Monstro deixa livre
                MOV     R7, R4
                ADD     R7, INC_Col
                MOV     M[PORTO_Ctrl], R7
                MOV     R4, M[SP+8]
                MOV     M[R4], R7               ;Actualiza a posição do
                                                ;Monstro na janela
                MOV     R7, R6
                INC     R7
                MOV     R6, M[SP+9]
                MOV     M[R6], R7               ;Actualiza a posição do
                                                ;Monstro em memória
                MOV     R5, M[SP+10]
```

```
                MOV     M[R5], R3               ;Actualiza o que fica por
                                                ;baixo do Monstro

                MOV     R5, R7
                MOV     R7, 'M'
                MOV     M[PORTO_Esc], R7        ;Escreve Monstro na nova
                                                ;posição na Janela
                MOV     M[R5], R7               ;Escreve Monstro na nova
                                                ;posição em Memória
                JMP     FimMonDir

MonDirObstac:   MOV     R4, M[SP+11]
                PUSH    R4
                CALL    RandomMon
                JMP     FimMonDir

MonDirPacMan:   MOV     R4, M[SP+8]
                MOV     R4, M[R4]
                MOV     R5, M[SP+10]
                MOV     R5, M[R5]
                MOV     R6, M[SP+9]
                MOV     R6, M[R6]
                MOV     M[PORTO_Ctrl], R4
                MOV     M[PORTO_Esc], R5        ;Restora o conteúdo da posição
                                                ;que o Monstro deixa livre
                MOV     M[R6], R5               ;Restora a posição de memória
                                                ;que o Monstro deixa livre
                MOV     R7, R4
                ADD     R7, INC_Col
                MOV     M[PORTO_Ctrl], R7
                MOV     R4, M[SP+8]
                MOV     M[R4], R7               ;Actualiza a posição do
                                                ;Monstro na janela
                MOV     R7, R6
                INC     R7
                MOV     R6, M[SP+9]
                MOV     M[R6], R7               ;Actualiza a posição do
                                                ;Monstro em memória
                MOV     R5, R7
                MOV     R7, 'M'
                MOV     M[PORTO_Esc], R7        ;Escreve Monstro em cima do
                                                ;Pac-Man na Janela
                MOV     M[R5], R7               ;Escreve Monstro em cima do
                                                ;Pac-Man em Memória
                MOV     R5, ENABLE
                MOV     M[CondGameOver], R5     ;Activa a var. condição para
                                                ;o "Game Over"

FimMonDir:      POP     R7
                POP     R6
                POP     R5
                POP     R4
                POP     R3
                POP     R2
                RETN    4

;===============================================================
; RandomMon: Rotina que, aplicando a instrução I1OP (Random) a
;            um valor (inicialmente uma semente), gera um valor
;            semi-aleatório entre 0 e 3, correspondente a uma
;            direcção de movimento. O resultado é guardado na
;            variável de direcção de um monstro, passada como
```



```
;            parâmetro pela pilha
; Parâmetro passado a esta rotina pela pilha:
;            M[SP+4] <- DirMonstroX (X=1,2,3,4,5)
;===============================================================

RandomMon:      PUSH    R6
                PUSH    R7
Random:         I1OP    M[RandomVar]
                MOV     R6, M[RandomVar]
                MOV     R7, 4                   ;Quatro direcções possíveis
                DIV     R6, R7                  ;O resto, valor entre 0 e 3,
                                                ;ficará em R7
                MOV     R6, M[SP+4]             ;Passa end. DirMonstroX para R6
                CMP     M[R6], R7               ;Se a nova direcção for igual à
                BR.Z    Random                  ;anterior repete a geração de
                                                ;número aleatório
                MOV     M[R6], R7               ;Guarda a nova direcção em M[R6]
                POP     R7
                POP     R6
                RETN    1

;===============================================================
; AumPontos: Rotina que soma pontos obtidos à pontuação actual
;            do jogador
; Parâmetro passado a esta rotina pela pilha:
;            M[SP+3] <- PtsPonto, PtsBanana, PtsPera ou PtsGelado
;===============================================================

AumPontos:      PUSH    R7
                MOV     R7, M[SP+3]
                ADD     M[Pontuacao], R7
                MOV     R7, M[Pontuacao]
                CMP     R7, LIM_Pontuacao       ;Se pontuação for maior que 9999
                BR.NP   ContAumPontos
                MOV     R7, LIM_Pontuacao       ;força-se pontuação = 9999
                MOV     M[Pontuacao], R7
ContAumPontos:  MOV     R7, M[Pontuacao]
                PUSH    R7
                CALL    ActDisplay
                PUSH    R7
                CALL    ActDific
                POP     R7
                RETN    1

;===============================================================
; ActDisplay: Rotina que, recebendo como parâmetro pela
;             pilha a pontuação actual do jogador, coloca
;             esse valor no display de 7 segmentos
; Parâmetro passado a esta rotina pela pilha:
;             M[SP+5] <- M[Pontuacao] em registo
;===============================================================

ActDisplay:     PUSH    R1
                PUSH    R2
                PUSH    R3
                MOV     R1, Display             ;Passa endereço Display para R1
                MOV     R2, M[SP+5]             ;Passa pontuação para R2
                MOV     R3, 10
                DIV     R2, R3                  ;Obtém e escreve dígito menos
                MOV     M[R1],  R3              ;significativo no display
                MOV     R3, 10
                DIV     R2, R3                  ;Obtém e escreve segundo dígito
```

```
                MOV     M[R1+1], R3             ;menos significativo no display
                MOV     R3, 10
                DIV     R2, R3                  ;Obtém e escreve terceiro dígito
                MOV     M[R1+2], R3             ;menos significativo no display
                MOV     R3, 10
                DIV     R2, R3                  ;Obtém e escreve dígito mais
                MOV     M[R1+3], R3             ;significativo no display
                POP     R3
                POP     R2
                POP     R1
                RETN    1

;===============================================================
; ActDific: Rotina que, recebendo como parâmetro pela
;           pilha a pontuação actual do jogador, actualiza
;           a dificuldade (velocidade) do jogo
; Parâmetro passado a esta rotina pela pilha:
;           M[SP+5] <- M[Pontuacao] em registo
;===============================================================

ActDific:       PUSH    R1
                PUSH    R2
                PUSH    R3
                MOV     R2, M[SP+5]             ;Passa pontuação para R2
                MOV     R3, LIM_DifPts          ;Passa margens de dific. para R3
                MOV     R1, 1
                MUL     R1, R3
                CMP     R2, R3                  ;As comparações aqui efectuadas
                JMP.N   ActDifNiv1              ;destinam-se a verificar onde
                MOV     R3, LIM_DifPts          ;a pontuação actual se encaixa
                MOV     R1, 2                   ;dentro das margens de
                MUL     R1, R3                  ;pontuação que definem a
                CMP     R2, R3                  ;dificuldade.
                JMP.N   ActDifNiv2              ;Estas margens podem ser
                MOV     R3, LIM_DifPts          ;conferidas na página 7
                MOV     R1, 3                   ;do relatório do projecto
                MUL     R1, R3
                CMP     R2, R3
                JMP.N   ActDifNiv3
                MOV     R3, LIM_DifPts
                MOV     R1, 4
                MUL     R1, R3
                CMP     R2, R3
                JMP.N   ActDifNiv4
                MOV     R3, LIM_DifPts
                MOV     R1, 5
                MUL     R1, R3
                CMP     R2, R3
                JMP.N   ActDifNiv5
                MOV     R3, LIM_DifPts
                MOV     R1, 6
                MUL     R1, R3
                CMP     R2, R3
                JMP.N   ActDifNiv6
                MOV     R3, LIM_DifPts
                MOV     R1, 7
                MUL     R1, R3
                CMP     R2, R3
                JMP.N   ActDifNiv7
                JMP     ActDifNiv8

ActDifNiv1:     MOV     R1, DIF_Niv1            ;Dependendo da pontuação,
```



```
                MOV     M[TimeLong], R1         ;a variável TimeLong
                MOV     R1, LEDs1               ;(correspondente à velocidade
                MOV     M[LEDs], R1             ;de jogo) e o número de LEDs
                JMP     FimActDific             ;acesos são alterados

ActDifNiv2:     MOV     R1, DIF_Niv2
                MOV     M[TimeLong], R1
                MOV     R1, LEDs2
                MOV     M[LEDs], R1
                JMP     FimActDific

ActDifNiv3:     MOV     R1, DIF_Niv3
                MOV     M[TimeLong], R1
                MOV     R1, LEDs3
                MOV     M[LEDs], R1
                JMP     FimActDific

ActDifNiv4:     MOV     R1, DIF_Niv4
                MOV     M[TimeLong], R1
                MOV     R1, LEDs4
                MOV     M[LEDs], R1
                JMP     FimActDific

ActDifNiv5:     MOV     R1, DIF_Niv5
                MOV     M[TimeLong], R1
                MOV     R1, LEDs5
                MOV     M[LEDs], R1
                BR      FimActDific

ActDifNiv6:     MOV     R1, DIF_Niv6
                MOV     M[TimeLong], R1
                MOV     R1, LEDs6
                MOV     M[LEDs], R1
                BR      FimActDific

ActDifNiv7:     MOV     R1, DIF_Niv7
                MOV     M[TimeLong], R1
                MOV     R1, LEDs7
                MOV     M[LEDs], R1
                BR      FimActDific

ActDifNiv8:     MOV     R1, DIF_Niv8
                MOV     M[TimeLong], R1
                MOV     R1, LEDs8
                MOV     M[LEDs], R1

FimActDific:    POP     R3
                POP     R2
                POP     R1
                RETN    1

;===============================================================
; MudaNivel: Rotina que trata a mudança de nível quando o jogador
;            come os pontos todos do mapa
;===============================================================

MudaNivel:      DSI
                PUSH    R2
                MOV     R1, M[MapaActual]
                CMP     R1, COD_Niv1            ;Se estiver no Nível 1
                BR.Z    MudaNiv2                ;Muda para o Nível 2
                BR      MudaNiv3                ;Caso contrário, carrega Nível 3
```

```
MudaNiv2:       MOV     R1, COD_Niv2
                MOV     M[MapaActual], R1
                PUSH    Nivel2_L0               ;Passa o endereço do mapa
                                                ;do Nível 2
                CALL    EscreveMapa             ;para a rotina EscreveMapa
                BR      FimMudaNivel

MudaNiv3:       MOV     R1, COD_Niv3
                MOV     M[MapaActual], R1
                PUSH    Nivel3_L0               ;Passa o endereço do mapa
                                                ;do Nível 3
                CALL    EscreveMapa             ;para a rotina EscreveMapa

FimMudaNivel:   POP     R2
                ENI
                MOV     R1, INDEF               ;Colocar estas duas instruções
                MOV     M[Direccao], R1         ;aqui evita que o Pac-Man se
                                                ;mexa devido a interrupções
                                                ;pendentes

                RET

;===============================================================
; RotGameOver: Rotina que trata de reinicializar os parâmetros
;              de jogo  necessários em caso de "Game Over".
;              Actualiza também a pontuação máxima,
;              se necessário
;===============================================================

RotGameOver:    PUSH    R2
                MOV     R1, M[Pontuacao]
                MOV     R2, M[PontMaxima]
                CMP     R2, R1
                BR.NN   ContGameOver            ;Se pontuação não for maior que
                                                ;o high score
                MOV     M[PontMaxima], R1       ;Caso contrário, actualiza o
                                                ;high score

ContGameOver:   PUSH    GameOver_L0             ;Passa o endereço da animação
                CALL    EscreveMapa             ;"Game Over"para a rotina
                                                ;EscreveMapa (func. opcional)
                MOV     R1, INDEF
                MOV     M[PosMonstro1T], R0     ;Reinicializa a posição dos
                MOV     M[PosMonstro1M], R0     ;monstros em memória e na
                                                ;janela de texto (escondidos)
                MOV     M[DirMonstro1], R1      ;Coloca igualmente a direcção
                MOV     M[PosMonstro2T], R0     ;inicial de todos eles como
                MOV     M[PosMonstro2M], R0     ;indefinida, de forma a estarem
                                                ;prontos para o próximo jogo
                MOV     M[DirMonstro2], R1
                MOV     M[PosMonstro3T], R0
                MOV     M[PosMonstro3M], R0
                MOV     M[DirMonstro3], R1
                MOV     M[PosMonstro4T], R0
                MOV     M[PosMonstro4M], R0
                MOV     M[DirMonstro4], R1
                MOV     M[PosMonstro5T], R0
                MOV     M[PosMonstro5M], R0
                MOV     M[DirMonstro5], R1
                MOV     R1, INDEF
                MOV     M[Direccao], R1         ;Reinicializa a direcção inicial
```



```
                                                ;do Pac-Man
                MOV     R1, DIF_Niv1
                MOV     M[TimeLong], R1         ;Reinicializa a dif. do jogo
                MOV     M[Pontuacao], R0        ;Reinicializa a pontuação
                MOV     M[CondGameOver], R0     ;Coloca a var. condição para
                                                ;o "Game Over" a 0
                POP     R2
                RET


; Rotinas de tratamento da interrupção
;===============================================================
; Rotinas correspondentes às teclas de direcção
; Alteram a variável de direcção do Pac-Man
;===============================================================

IntCima:        PUSH    R1
                MOV     R1, CIMA
                MOV     M[Direccao], R1
                POP     R1
                RTI

IntBaixo:       PUSH    R1
                MOV     R1, BAIXO
                MOV     M[Direccao], R1
                POP     R1
                RTI

IntEsq:         PUSH    R1
                MOV     R1, ESQ
                MOV     M[Direccao], R1
                POP     R1
                RTI

IntDir:         PUSH    R1
                MOV     R1, DIR
                MOV     M[Direccao], R1
                POP     R1
                RTI

;===============================================================
; IntTimer: Activa variável de condição do temporizador
;           e reinicia a contagem
;===============================================================

IntTimer:       PUSH    R1
                MOV     R1, M[TimeLong]
                MOV     M[TimerValue], R1
                MOV     R1, ENABLE
                MOV     M[CondTimer], R1
                MOV     M[TimerControl], R1
                POP     R1
                RTI

;===============================================================
; Programa principal
;===============================================================

Inicio:         MOV     R1, Topo_Pilha
                MOV     SP, R1                  ;Inicialização da pilha, usando
                                                ;a posição mais alta possível!
                MOV     R1, INI_Int_Mask
```

```
                MOV     M[Int_Mask],R1          ;Definição da máscara de
                                                ;interrupções
                CALL    EscreveLCD
                BR      IniMenu
GameOver:       DSI
                CALL    RotGameOver             ;Reinicializar os parâmetros
IniMenu:        CALL    EscreveMenu             ;Mostrar menu principal
                MOV     R1, M[INI_Index]        ;Recoloca porto de estado a 0
                CALL    EsperaTecla
                CALL    EscolheNivel
InicioJogo:     MOV     R1, DIF_Niv1
                MOV     M[TimeLong], R1         ;Inicializa velocidade de jogo
                MOV     R1, LEDs1
                MOV     M[LEDs], R1             ;Acende o primeiro LED
                MOV     R1, R0
                PUSH    R1
                CALL    ActDisplay              ;Coloca pontuação inicial a 0
                MOV     R1, M[TimeLong]
                MOV     M[TimerValue], R1       ;Passa velocidade para o Timer
                MOV     R1, ENABLE
                MOV     M[TimerControl], R1     ;Activa o temporizador
                ENI
                MOV     R1, INDEF               ;Colocar estas duas instruções
                MOV     M[Direccao], R1         ;aqui evita que o Pac-Man se
                                                ;mexa devido a interrupções
                                                ;pendentes
CicloJogo:      MOV     R1, M[CondTimer]
                CMP     R1, ENABLE              ;Verifica se a var. de condição
                BR.NZ   CicloJogo               ;do temporizador foi activada
                CALL    ActMapa
                MOV     M[CondTimer], R0        ;Desactiva variável condição do
                                                ;temporizador
                MOV     R1, M[NumPontos]
                CMP     R1, R0                  ;Se tiver comido os pontos todos
                CALL.Z  MudaNivel
                MOV     R1, M[CondGameOver]
                CMP     R1, ENABLE              ;Se tiver sido apanhado por
                                                ;um monstro...
                JMP.Z   GameOver                ;..trata do Game Over e
                                                ;reinicializa
                BR      CicloJogo
```