```
;Ana Sofia Apararicio n 81105
;Joana Godinho n 81478
;Nuno Fernandes n 80774

; DefiniÃ§ao de constantes

        INT_MASK                EQU     FFFAh
        INT_1                   EQU     1000101010000011b       ; para ativar as
 quatro interrupÃ§oes e a interrupÃ§ao do temporizador
        FIM_TEXTO       EQU     '@'
        IO_WRITE        EQU     FFFEh
        IO_APONTADOR    EQU             FFFCh
        IO_DISPLAY      EQU     FFF0h
        IO_Leds                 EQU             FFF8h
        SP_INICIAL      EQU     FDFFh
        NIBBLE_MASK     EQU     000fh
        NUM_NIBBLES     EQU     4               ; sao 4 valores que formam o rel
ogio
        BITS_PER_NIBBLE EQU     4               ;cada valor do relogio Ã© formad
o por 4 bits
        LCD                             EQU FFF4h
        LCD_Write               EQU     FFF5h


                                ORIG    8000h
        StringIntro1    STR     'Bem-Vindo ao TRON',FIM_TEXTO
        StringIntro2    STR     'Pressione I1 para comecar',FIM_TEXTO
        VarString1              STR     '+--------------------------------------
----------+',FIM_TEXTO
        VarString2              STR     '|
          |',FIM_TEXTO
        VarString3              STR     '|'
        VarString4              STR     '+--------------------------------+',FIM
_TEXTO
        VarString5              STR     '|                                |',FIM
_TEXTO
        Linhas                  WORD 20d
        Colunas                 WORD 48d
        IniMatriz               WORD 010fh              ;valor onde comeÃ§a a ma
triz (para corresponder as posiÃ§oes da matriz as posiÃ§oes de memoria)
        PosJ1                   WORD 0D18h
        PosJ2                   WORD 0D37h
        PrimeiroJogador STR     'X',FIM_TEXTO
        SegundoJogador  STR     '#',FIM_TEXTO
        StringFim1              STR     ' Fim de jogo ',FIM_TEXTO
        StringFim2              STR     ' Pressione I1 para recomecar',FIM_TEXTO
        StringLimpa             STR '
        ',FIM_TEXTO
        Contador                WORD 0000h
        Vetor1                  WORD 0001h
        Vetor2                  WORD FFFFh
        FlagI1                  WORD 0000h
        Nivel                   WORD 7d
        FlagInt                 WORD 0000h
        FlagTemp1               WORD 0000h
        FlagTemp2               WORD 0000h
        FlagEsqJ1               WORD 0000h
        FlagDirJ1               WORD 0000h
        FlagEsqJ2               WORD 0000h
        FlagDirJ2               WORD 0000h
        FlagChoque              WORD 0000h
```



```
        StringLCD1              STR     'Temp max:',FIM_TEXTO
        StringLCD2              STR     'J1',FIM_TEXTO
        StringLCD3              STR     'J2',FIM_TEXTO
        StringLCD4              STR 's',FIM_TEXTO
        FlagCiclo               WORD 0000h                              ;utiliza
da no escreveceLcd
        TempMax                 WORD 0000h
        FlagJ1                  WORD 0000h
        FlagJ2                  WORD 0000h


;Tabela de interrupÃ§oes:

                                ORIG    FE00h
INT0                    WORD    EsquerdaJ1
INT1                    WORD    Inicializar
                                ORIG    FE07h
INT7                    WORD    EsquerdaJ2
                                ORIG    FE09h
INT9                    WORD    DireitaJ2
                                ORIG    FE0Bh                           ;posiÃ§a
o da interrupÃ§ao 11(IB)
INTB                    WORD    DireitaJ1
                                ORIG    FE0Fh
INT15                   WORD    Temporizador

;Codigo:
                                        ORIG    0000h
                                        JMP             2000h

                                        ORIG    2000h
                                        JMP             Inicio
; ==================================================================
;       EsquerdaJ1: Funcao de servico a interrupcao I0;
;       Entradas: --
;       Saidas: --
;       Efeitos:  ativa a flag de movimento para esquerda do jogador 1

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


EsquerdaJ1:             PUSH    R3
                                MOV             R3,0001h
                                MOV             M[FlagEsqJ1],R3
                                POP             R3
                                RTI
; ==================================================================
;       DireitaJ1: Funcao de servico a interrupcao IB
;       Entradas: --
;       Saidas: --
;       Efeitos: ativa a flag de movimento para a direita do jogador 1

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

DireitaJ1:              PUSH    R3
                                MOV             R3,0001h
                                MOV             M[FlagDirJ1],R3
                                POP             R3
                                RTI
; ==================================================================
;       EsquerdaJ2:  Rotina de serviÃ§o a interrupÃ§ao I7
;       Entradas: --
```

```
;       Saidas: --
;       Efeitos: ativa a flag de movimento para a esquerda do jogador 2

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


EsquerdaJ2:             PUSH    R3
                                MOV             R3,0001h
                                MOV             M[FlagEsqJ2],R3
                                POP             R3
                                RTI

; ==============================================================
;       DireitaJ2:  Rotina de serviÃ§o a interrupÃ§ao I9
;       Entradas: --
;       Saidas: --
;       Efeitos: ativa a flag de movimento para a direita do jogador 2

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

DireitaJ2:              PUSH    R3
                                MOV             R3,0001h
                                MOV             M[FlagDirJ2],R3
                                POP             R3
                                RTI

; ==============================================================
;       Inicializar:  Rotina de serviÃ§o a interrupÃ§ao I1
;       Entradas: --
;       Saidas: --
;       Efeios: ativa a flag de inicializacao permitindo o comeco do jogo

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

Inicializar:    PUSH    R3
                                MOV             R3,1
                                MOV             M[FlagI1],R3
                                POP             R3
                                RTI

; ==============================================================
;       Temporizador: coloca a flagInt a 1
;       Entradas: --
;       Saidas: --
;       Efeitos:  reativa o temporizador apos a sua interrupcao

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

Temporizador:   PUSH    R6
                                MOV             R6, 0001h
                                MOV             M[FlagInt],R6
                                POP             R6
                                RTI

; ==============================================================
;       Relogio: responsavel pela contagem do tempo de jogo
;       Entradas: --
;       Saidas: --
;       Efeitos: altera os digitos do display de 7 segmentos (relogio) e
;                       altera a variavel Nivel consoante o tempo no display

; ==============================================================
```



```
Relogio:                Push    R3
                                MOV             M[FlagTemp2],R0
                                CALL    VerificaNivel
                                CaLL    ContaDigitos
                                CaLL    EscCont
                                POP             R3
                                RET

; ==============================================================
==========
;       EscCont: escreve no display de 7segmentos
;       Entradas: --
;       Saidas: --
;       Efeitos: impreme no display de 7 segmentos os digitos que estao na varia
lvel Contador
; ==============================================================
==========
EscCont:        PUSH    R1
                PUSH    R2
                MOV     R2, NUM_NIBBLES
                MOV     R3, IO_DISPLAY

Ciclo2:                 MOV     R1, M[Contador]                 ; este ciclo vai
 servir para escrever cada um dos 4 numeros do relogio
                AND     R1, NIBBLE_MASK
                MOV     M[R3], R1
                ROR     M[Contador], BITS_PER_NIBBLE    ; faz um deslocamento pa
ra a direita para ir retirando os 4 bits correspondentes ao valor colocado anter
iormente
                INC     R3
; aumenta o porto de controlo dos display para ir mudando de display
                DEC     R2

                BR.NZ   Ciclo2
                POP     R2
                POP     R1
                RET

; ==============================================================
===
;       ContaDigitos: altera os valores escritos no display de 7 segmentos
;       Entradas: --
;       Saidas: --
;       Efeitos: vai aumentar o valor da variavel Contador pondo este em decimal

; ==============================================================
===
ContaDigitos:   PUSH    R1
                                PUSH    R2
                                MOV             R1,M[Contador]
                                MOV     R2, 000Fh
                                MOV             R3, R2
                                AND             R1,R2
        ;para so obter o ultimo digito
                                CMP             R1,0009h
                                JMP.NZ          Soma
                                MOV     R2, FFF0h
        ;para por a zero o numero mais a direita
                                AND     M[Contador], R2
                                MOV     R2, 00F0h
                                MOV     R3,R2
```

```
                        AND     R2, M[Contador]
                        CMP     R2,     0090h
                        JMP.NZ  Soma
                        MOV     R2, FF0Fh
                        AND     M[Contador], R2
                        MOV     R2, 0F00h
                        MOV     R3,R2
                        AND     R2, M[Contador]
                        CMP     R2,     0900h
                        JMP.NZ  Soma
                        MOV     R2, F0FFh
                        AND     M[Contador], R2
                        MOV     R2, F000h
                        MOV     R3,R2
                        AND     R2, M[Contador]
                        CMP     R2,     9000h
                        JMP.NZ  Soma

FIM7:           POP     R2
                        POP             R1
                        RET

Soma:           MOV     R1, 1111h                               ;para es
colher o digito a que se vai somar 1(o primeiro, o segundo...)
                        AND     R1,R3

                        ADD             M[Contador],R1
;esta funÃ§ao soma vai somar 1 ao digito enquanto este nao esta a 9
                        JMP             FIM7
                        RET

; ===========================================================================
;       VerificaNivel: vai comparar o valor do contador com o valor da
;                      variavel nivel
;       Entradas: --
;       Saidas: --
;       Efeitos: altera o valor da variavel nivel aos 10s,20s,40s,60s

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

VerificaNivel:  PUSH    R3
                        PUSH    R4
                        MOV             R3,M[Contador]
                        CMP             R3,0009h
                        JMP.NZ  Nivel2
                        MOV             M[FlagTemp1], R0
                        MOV             R3,3d
                        MOV             M[Nivel],R3
                        MOV             R4,000Fh
                        MOV             M[IO_Leds],R4

Nivel2:         MOV             R3,M[Contador]
                        CMP             R3,0019h
                        BR.NZ   Nivel3
                        MOV             M[FlagTemp1], R0
                        MOV             R3,3d
                        MOV             M[Nivel],R3
                        MOV             R4,00FFh
                        MOV             M[IO_Leds],R4

Nivel3:         MOV             R3,M[Contador]
                        CMP             R3,0039h
```



```
                        BR.NZ   Nivel4
                        MOV             M[FlagTemp1], R0
                        MOV             R3,2d
                        MOV             M[Nivel],R3
                        MOV             R4,0FFFh
                        MOV             M[IO_Leds],R4

Nivel4:         MOV             R3,M[Contador]
                        CMP             R3,0059h
                        BR.NZ   NFim
                        MOV             M[FlagTemp1], R0
                        MOV             R3,1d
                        MOV             M[Nivel],R3
                        MOV             R4,FFFFh
                        MOV             M[IO_Leds],R4

NFim:           POP             R4
                        POP             R3
                        RET


; ===========================================================================
=======
;       GuardaTecto: vai fazer com que as posiÃ§oes de memoria correspondentes a
s posiÃ§oes
;                                    do teto na matriz sejam ocupadas
;       Entradas: valor da primeira posicao da matiz; posicao onde
;            comeca a string
;       Saidas:--
;   Efeitos: as posicoes de memoria referidas deixam de estar a zero

; ===========================================================================
=======
GuardaTecto:            MOV             R3,M[R2]
                                        CMP             R3,FIM_TEXTO
                                        JMP.Z   FIM_TECTO
                                        MOV             M[R1],R3
                                        INC             R2
                                        INC             R1
                                        BR              GuardaTecto
FIM_TECTO:                      RET

; ===========================================================================
====
;       GuardaParede: guarda os limites laterais da matriz
;       Entradas: --
;       Saidas: --
;   Efeitos: as posicoes de memoria correspondentes ao limites laterais ficam "o
cupadas"
; ===========================================================================
====
GuardaParede:           MOV             R3,M[R2]
                                        MOV             M[R1],R3
; guarda '|' na posiÃ§ao de memoria correspondente a posiÃ§ao de '|' na matriz
                                        ADD             R1,0100h
                                        DEC             R4

                                        JMP.Z   FIM_PAREDE
                                        BR              GuardaParede
FIM_PAREDE:                     RET

; ===========================================================================
```

```
===
;       GuardaMatriz: vai guardar nas posicoes de memoria correspondentes os lim
ites
;                                 da matriz (serve para indicar que estao ocupad
as)
;       Entradas: --
;       Saidas: --
;   Efeitos: as posicoes de memoria correspondentes ao limites da matriz ficam "
ocupadas"
; ==========================================================================
===
GuardaMatriz:           MOV             R1,M[IniMatriz]
                                        MOV             R2,VarString1
                                        MOV             R3,0001h
                                        TEST    M[FFF9h],R3
; caso o interruptor 0 esteja ligado
                                        BR.Z    Continua3
                                        MOV             R2,VarString4
Continua3:                      PUSH    R1
;serve para guardar o valor 010fh na pilha para depois mudar de linha
                                        MOV             R1,M[SP+1]
;para aceder ao valor de R1
                                        CALL    GuardaTecto
                                        POP             R1
                                        ADD             R1,0100h
                                        PUSH    R1
                                        MOV             R1,M[SP+1]
                                        MOV             R2,VarString3
                                        MOV             R4,M[Linhas]
; R4 vai funcionar como um contador das linhas
                                        CALL    GuardaParede
                                        POP             R1
        ; obter o valor do comeÃ§o da primeira linha do espaÃ§o de jogo
                                        MOV             R5,M[Colunas]
                                        INC             R5

                                        ADD             R1,R5
; incrementa-se o numero de colunas porque entre as paredes estao M[Colunas] vaz
ias, logo na M[Colunas]+1 esta a segunda parede
                                        MOV             R4,M[Linhas]
                                        CALL    GuardaParede
                                        SUB             R1,R5
; subtrai-se 31h para obter a posiÃ§ao inicial do contorno de baixo da matriz (a
pos a funÃ§ao GuardaParede R1 refere-se a ultima posiÃ§ao da linha 22
                                        MOV             R2,VarString1
                                        MOV             R3,0001h
                                        TEST    M[FFF9h],R3
; caso o interruptor 0 esteja ligado
                                        BR.Z    Continua4
                                        MOV             R2,VarString4
Continua4:                      CALL    GuardaTecto
                                        RET


; ==========================================================================
=======
;       EscreveStr: serve para escrever na janela de texto as strings
;       Entradas: posicao do inicio da string; R5 posicao onde se escreve na jan
ela
;       Saidas: --
;   Efeitos: Imprime na janela de texto as frases\carateres

; ==========================================================================
```



```
=======

EscreveStr:             PUSH    R1
                                PUSH    R7
                                MOV             R7,R5
Ciclo:                  MOV             M[IO_APONTADOR],R7              ; para i
r alterando a posiÃ§ao onde se escreve a medida q se avanÃ§a
                                MOV             R1,M[R2]
;move o que esta na string para depois comparar
                                CMP     R1, FIM_TEXTO
                BR.Z    FimEsc
                CALL    EscLinha
                INC     R2
                BR      Ciclo

EscLinha:               PUSH    R1                                      ; para g
uardar o carater que se pretende escrever
                                MOV             R1,M[SP+1]
;+1 porque e na posicao sp+1 da pilha que esta o que se pretende escrever
                                MOV     M[IO_WRITE], R1
                                INC             R7
                                POP     R1
                                RET


FimEsc:         POP     R7
                POP     R1
                                ADD             R5,0100h
; para mudar de linha
                RET



; ==========================================================================
;       EscreveMatriz: vai produzir uma matriz cujas dimensoes dependem se os
;                                 interruptores 0 e 1 estao accionados ou
nao
;       Entradas: --
;       Saidas: --
;       Efeitos: matriz impressa na janela de texto

; ==========================================================================
EscreveMatriz:  PUSH    R1
                                PUSH    R3
                                MOV             R5,M[IniMatriz]
                                MOV             R2,VarString1
                                MOV             R3,VarString2
                                MOV             R6,VarString1
                                MOV             R1,0001h
                                TEST    M[FFF9h],R1
; caso o interruptor 0 esteja ligado
                                BR.Z    Continua2
                                MOV             R2,VarString4
                                MOV             R3,VarString5
                                MOV             R6,VarString4
Continua2:              CALL    EscreveStr
                                MOV             R4,M[Linhas]
                                CaLL    CicloLinhas
                                MOV     R2,R6
                                CALL    EscreveStr
                                MOV             R5,M[PosJ1]
                                MOV             R2,PrimeiroJogador
                                CALL    EscreveStr
```

```
                        MOV     R5,M[PosJ2]
                        MOV     R2,SegundoJogador
                        CALL    EscreveStr
                        POP     R3
                        POP     R1
                        RET

CicloLinhas:    MOV     R2,R3
                        CALL    EscreveStr
                        DEC     R4
;serve para verificar quando e que as 20 linhas da matriz foram escritas
                        JMP.NZ  CicloLinhas
                        RET
;=====================================
;Inicio: Funcao Principal
;inclui a movimentacao dos jogadores
;=====================================

Inicio:         MOV     R7,SP_INICIAL
                        MOV     SP, R7
                        MOV     R7,FFFFh
                        MOV     M[IO_APONTADOR],R7
                        MOV     R5,0b1fh
                        MOV     R2,StringIntro1
                        CALL    EscreveStr
                        MOV     R5,0c1ah
;para escrever a string 2 na janela
                        MOV     R2,StringIntro2
                        CALL    EscreveStr
                        MOV     R7,INT_1
                        MOV     M[INT_MASK],R7          ; serve
para ativar as interrupcoes I0,I1,I7,I9 e IB
                        CALL    ResetMemoria
                        ENI
CMPFlag:        CMP             M[FlagI1],R0            ; vai verificand
o se ocorreu alguma interrupÃ§ao
                        BR.Z    CMPFlag
                        CALL    EscLCD
Reinicio:       CALL    LimpaJanela
                        CALL    CompInt
                        CaLL    GuardaMatriz

                        CaLL    EscreveMatriz
                        DSI
                        MOV             R3,1d
                        MOV             R1,M[PosJ1]
                        MOV             M[R1],R3
                        MOV             R1,M[PosJ2]
                        MOV             M[R1],R3
                        MOV             M[FFF6h],R3
                        MOV             R3,1h
                        MOV             M[FFF7h],R3
                        MOV             R1,M[PosJ1]             ; ao lon
go do jogo os registos R1 e R2 vao corresponder as posiÃ§oes, respetivamente do
jogador1(X) e do jogador2(#)
                        MOV             R2,M[PosJ2]
                        ENI
                        JMP             Comp

CicloJogo:      MOV             M[FlagInt], R0
                        MOV             R3,0080h
                        TEST    M[FFF9h],R3
```



```
                        CALL.NZ VerPausa
                        MOV             R3,1d
                        MOV             M[FFF6h],R3
                        MOV             R3,1h
                        MOV             M[FFF7h],R3
                        INC             M[FlagTemp1]
                        INC             M[FlagTemp2]    ;R7 vai funcio
nar como um contador do numero de vezes que ocorrem as interrupÃ§oes do relogio
                        MOV             R7,M[FlagTemp1]
                        CMP     R7, M[Nivel]
                        BR.Z    Movimento
CMPFlagTemp:    MOV             R7,M[FlagTemp2]
                        CMP             R7,10d
                        JMP.NZ  Comp
                        CaLL    Relogio
                        JMP             Comp

Movimento:              MOV             M[FlagTemp1],R0         ; para reiniciar
 o contador
                        DSI
                        CALL    EscolheVet1
                        CALL    EscolheVet2
                        ADD             R1,M[Vetor1]
                        ADD             R2,M[Vetor2]
                        ENI
                        CALL    VerChoque
                        MOV             R5,R1
                        PUSH    R2
; para guardar na pilha o valor da posiÃ§ao do jogador2
                        MOV             R2,PrimeiroJogador
                        CALL    EscreveStr
                        POP             R2
                        MOV             R5,R2
                        Push    R2
                        MOV             R2,SegundoJogador
                        CALL    EscreveStr
                        POP             R2
                        JMP             CMPFlagTemp
Comp:                   ENI
                        MOV             R6,M[FlagInt]
                        CMP     R6, 0001h
                        JMP.NZ  Comp
                        JMP             CicloJogo
                        RET


; ==========================================================================
===
;       EscolheVet1: dependendo da interrupcao feita vai escolher o vetor
;                                a somar a posicao do jogador 1
;       Entradas: --
;       Saidas: --
;       Efeitos: altera o valor da variavel vetor1 caso tenha ocorrido uma inter
rupcao

; ==========================================================================
===
EscolheVet1:    CMP             M[FlagEsqJ1],R0
                        BR.Z    DirJ1
                        MOV             M[FlagEsqJ1],R0
                        CALL    VetorJ1
                        JMP             FIM2
DirJ1:                  CMP             M[FlagDirJ1],R0
```

```
                                BR.Z    FIM2
                                MOV             M[FlagDirJ1],R0
                                CALL    VetorJ1

                                NEG             M[Vetor1]               ;a movim
entaÃ§ao para a direita Ã© o simetrico da movimentaÃ§ao para a esquerda
FIM2:                   RET

; ====================================================================
;       VetorJ1: vai escolher o vetor a somar a posicao do J1 dependendo
;                       do vetor anterior (direcao)
;       Entradas: --
;       Saidas: --
;       Efeitos: altera o valor da variavel vetor2

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

VetorJ1:                PUSH    R4
                                MOV             R4,0001h
                                CMP             R4,M[Vetor1]
                                BR.NZ   UM
                                MOV             R4,FF00h
                                MOV             M[Vetor1],R4
                                JMP             FIM3
UM:                             MOV             R4,FF00h
                                CMP             R4,M[Vetor1]
                                JMP.NZ  DOIS
                                MOV             R4,FFFFh
                                MOV             M[Vetor1],R4
                                JMP             FIM3
DOIS:                   MOV             R4,FFFFh
                                CMP             R4,M[Vetor1]
                                JMP.NZ  TRES
                                MOV             R4,0100h
                                MOV             M[Vetor1],R4
                                JMP             FIM3
TRES:                   MOV             R4,0001h
                                MOV             M[Vetor1],R4
FIM3:                   POP             R4
                                RET

; ====================================================================
==================
;       EscolheVet2: dependendo da interrupcao feita vai escolher o vetor
;                               a somar a posicao do jogador 2
;       Entradas: --
;       Saidas: --
;       Efeitos: altera o valor da variavel vetor2 caso tenha ocorrido uma inter
rupcao
; ====================================================================
==================


EscolheVet2:    CMP             M[FlagEsqJ2],R0
                                JMP.Z   DirJ2
                                MOV             M[FlagEsqJ2],R0
                                CALL    VetorJ2
                                JMP             FIM2
DirJ2:                  CMP             M[FlagDirJ2],R0
                                BR.Z    SFIM2
                                MOV             M[FlagDirJ2],R0
                                CALL    VetorJ2
```



```
                                NEG             M[Vetor2]       ;a movimentaÃ§ao
 para a direita Ã© o simetrico da movimentaÃ§ao para a esquerda
SFIM2:                  RET

; ====================================================================
==
;       VetorJ2: vai escolher o vetor a somar a posicao do J2 dependendo
;                               do vetor anterior (direcao)
;       Entradas: --
;       Saidas: --
;       Efeitos: altera o valor da variavel vetor2 caso tenha ocorrido uma inter
rupcao
; ====================================================================
===

VetorJ2:                PUSH    R4
                                MOV             R4,0001h
                                CMP             R4,M[Vetor2]
                                JMP.NZ  SUM
                                MOV             R4,FF00h
                                MOV             M[Vetor2],R4
                                JMP             SFIM3
SUM:                    MOV             R4,FF00h
                                CMP             R4,M[Vetor2]
                                JMP.NZ  SDOIS
                                MOV             R4,FFFFh
                                MOV             M[Vetor2],R4
                                JMP             SFIM3
SDOIS:                  MOV             R4,FFFFh
                                CMP             R4,M[Vetor2]
                                JMP.NZ  STRES
                                MOV             R4,0100h
                                MOV             M[Vetor2],R4
                                JMP             SFIM3
STRES:                  MOV             R4,0001h
                                MOV             M[Vetor2],R4
SFIM3:                  POP             R4
                                RET

; ====================================================================
;       VerChoque: vai verificar se a posiÃ§ao de cada um dos jogadores esta ocu
pada
;       Entradas:       posicao do jogador 1; zposicao do jogador 2
;       Saidas: --
;       Efeitos: se as posicos nao estiverem coloca-se 0001h na memoria dessas p
osicoes
;                       para indicar que esta ocupada

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

VerChoque:              PUSH    R1
                                PUSH    R2
                                Push    R3
                                MOV             R1,M[SP+3]
                                MOV             R2,M[SP+2]
                                MOV             R3,1h
                                CMP             M[R1],R0
                                BR.NZ   PosOcupada1
                                MOV             M[R1],R3
;se nao houve choque, essa posiÃ§ao pode ser ocupada,
                                BR              Choque2
```

```
;logo coloca-se um valor qualquer excepto 0 (pois significa que esta vazia)
PosOcupada1:    INC             M[FlagChoque]
Choque2:                CMP             M[R2],R0
                        JMP.NZ  PosOcupada2
                        MOV             M[R2],R3
                        BR              FimChoque
PosOcupada2:    MOV             R3,0002h
                        ADD             M[FlagChoque],R3
FimChoque:              MOV             R3,0003h
;caso de empate
                        CMP             M[FlagChoque],R3
                        JMP.Z   FimJogo
                        MOV             R3,0002h
                        CMP             M[FlagChoque],R3
                        JMP.Z   PerdeJ2
                        MOV             R3,0001h
                        CMP             M[FlagChoque],R3
                        JMP.Z   PerdeJ1
                        POP             R3
                        POP             R2
                        POP             R1
                        RET



PerdeJ1:                DSI
                        MOV             R3,0100h                ; na esc
rita no lcd os primeiros 2 quartetos de bits da FlagJ1 correspondem ao numero de
 vitorias
                        ADD             M[FlagJ2],R3
                        MOV             R1,M[FlagJ2]
                        CALL    ConvDecimal
                        MOV             M[FlagJ2],R1
                        JMP             FimJogo

PerdeJ2:                DSI
                        MOV             R3,0100h

                        ADD             M[FlagJ1],R3
                        MOV             R1,M[FlagJ1]
                        CALL    ConvDecimal
                        MOV             M[FlagJ1],R1

FimJogo:                DSI
                        MOV             M[FFF7h],R0
                        MOV             R2,StringFim1
                        MOV             R5,0c20h
                        CaLL    EscreveStr
                        MOV             R2,StringFim2
                        MOV             R5,0d19h
                        CaLL    EscreveStr
                        CALL    EscLCD
                        CALL    ResetMemoria
                        CALL    ResetVar
                        MOV             M[FlagI1],R0            ;para de
sativar a flagI1 para esta estar a zero no fim do jogo
                        ENI
Fim1:                   CMP             M[FlagI1],R0
                        BR.Z    Fim1
                        MOV             M[FlagI1],R0
                        JMP             Reinicio

; ==============================================================================
```



```
===
;       EscLCD: permite a escrita no lcd do t
;       Entradas: --
;       Saidas: --
;       Efeitos: tempo maximo e da pontuacao dos jogadores impressos no Lcd

; ==============================================================================
===
EscLCD:                 MOV             R2,StringLCD1
                                MOV             R5,8000h            ;8000h ativa
o Lcd e indica que a escrita se inicia na coluna 0 e linha 0
                                CALL    EscStrLCD
                                MOV             R2,M[Contador]
                                CMP             R2,M[TempMax]
                                BR.N    Continua
                                MOV             M[TempMax],R2
Continua:               MOV             R2,M[TempMax]
                                MOV             R4,4
                                MOV             M[FlagCiclo],R4
                                CALL    EscNumLCD
                                MOV             R2,StringLCD4
                                INC             R5
                                CALL    EscStrLCD
                                MOV             R5,8010h
                                MOV             R2,StringLCD2
                                CALL    EscStrLCD
                                MOV             R2,M[FlagJ1]
                                MOV             R4,2
                                MOV             M[FlagCiclo],R4
                                CALL    EscNumLCD
                                ADD             R5,0003h
                                MOV             R2,StringLCD3
                                CALL    EscStrLCD
                                MOV             R2,M[FlagJ2]
                                MOV             R4,2
                                MOV             M[FlagCiclo],R4
                                CALL    EscNumLCD
                                RET

; ==============================================================================
=========
;       EscString:  permite a escrita no lcd de strings
;       Entradas: Posicao inicial da string a escrever; Posicao onde se pretende

;                       escrever no LCD
;       Saidas: --
;       Efeitos: string impressa no LCD

; ==============================================================================
==========
EscStrLCD:              MOV             M[LCD],R5
                                MOV             R3,M[R2]
                                CMP             R3,FIM_TEXTO
                                JMP.Z   FIM4
                                MOV             M[LCD_Write],R3
                                INC             R5
                                INC             R2
                                JMP             EscStrLCD
FIM4:                   RET

; ==============================================================================
==========
```

```
;       Inicio: escreve numeros no LCD
;       Entradas: conjunto de numeros a escrever e posicao na janela onde se pre
tende escrever
;       Saidas: --
;       Efeitos: no LCD vao aparecer os valores refenrentes as pontuacoes e
;                       ao tempo maximo

; ================================================================================
===========
EscNumLCD:              INC             R5
CicloLCD:       MOV             R3,R2
                AND     R3,F000h
                        SHR             R3,12
                        ADD             R3,'0'
                        MOV             M[LCD],R5
                        MOV             M[LCD_Write],R3
                        INC             R5
                        SHL             R2,4    ; para ir retirando os d
igitos que foram escritos
                        DEC             M[FlagCiclo]
                        JMP.NZ  CicloLCD
                        RET
; ================================================================================
==========
;       ResetMemoria:  vai limpar a memoria de forma a que nunhuma posicao estej
a
;                               "ocupada" quando se reinicia o jogo.
;       Entradas: --
;       Saidas: --
;   Efeitos: as posiÃ§oes de memoria correspondentes as posicoes da janela de te
xto
;                       ficam a 0
; ================================================================================
==========

ResetMemoria:   PUSH    R1
                        PUSH    R2
                        PUSH    R3
                        Push    R4
                        MOV             R1,010fh        ; a primeira pos
iÃ§ao ocupavel pelos jogadores Ã© uma linha e uma coluna a mais de onde se comeÃ
§ou a escrever a matriz M[IniMatriz]
                        MOV             R3,22d
CicloMemLinha:  MOV             R2,50d
                        MOV             R4,R1           ;para guardar a
primeira posiÃ§ao de cada linha para opter a primeira posiÃ§ao da linha seguinte
CicloMemCol:    MOV             M[R1],R0
                        INC     R1
                        DEC             R2
                        CMP             R2,R0
                        BR.NZ   CicloMemCol
                        MOV             R1,R4
                        ADD             R1,0100h
                        DEC             R3
                        BR.NZ   CicloMemLinha
                        POP             R4
                        POP             R3
                        POP             R2
                        POP             R1
                        RET


; ================================================================================
```



```
================
;       ResetVar:  coloca nas variaveis que se alteraram ao longo do jogo os seu
s valores iniciais
;       Entradas: --
;       Saidas: --
;       Efeitos: as variaveis utilizadas no jogo ficam com os valores pretendido
s para
;                       que se possa realizar outro jogo

; ================================================================================
================
ResetVar:               Push    R1
                        MOV             R1,7d
                        MOV             M[Nivel],R1
                        MOV             M[Contador],R0
                        MOV             M[IO_Leds],R0
                        MOV             M[FlagTemp1],R0
                        MOV             M[FlagTemp2],R0
                        MOV             M[FlagChoque],R0
                        MOV             R1,48d
                        MOV             M[Colunas],R1
                        MOV             R1,20d
                        MOV             M[Linhas],R1
                        MOV             R1,010fh
                        MOV             M[IniMatriz],R1
                        MOV             R1,0D18h
                        MOV             M[PosJ1],R1
                        MOV             R1,0D37h
                        MOV             M[PosJ2],R1
                        POP             R1
                        RET


; ================================================================================
=======
;       LimpaJanela: limpa a janela de texto
;       Entradas: --
;       Saidas: --
;       Efeitos:  apaga o que esta escrito na janela de texto

; ================================================================================
=======
LimpaJanela:    MOV             R5,010fh
                MOV             R4,22d
CicloLimpa:             MOV             R2,StringLimpa  ;na funÃ§ao EscreveStr j
a ocorre a soma de R5 com 0100h
                        CALL    EscreveStr

                        DEC             R4
                        BR.NZ   CicloLimpa
                        RET


; ================================================================================
===========
;       CompInt: verifica se algum dos interruptores foi accionado e quais.

;       Entradas: --
;       Saidas: --
;       Efeitos: Consoante os interruptores que estao para cima altera o valor d
a variavel
;                       colunas e\ou linhas.Se nenhum estiver para cima as varia
veis mantÃ©m-se iguais.
; ================================================================================
```

```
===========
CompInt:                PUSH    R1
                                MOV             R1,0001h
                                TEST    M[FFF9h],R1
                                BR.Z    IntLinhas
                                MOV             R1,32d
                                MOV             M[Colunas],R1
                                MOV             R1,0008h
                                ADD             M[PosJ1],R1
                                SUB             M[PosJ2],R1
                                ADD             M[IniMatriz],R1
IntLinhas:              MOV             R1,0002h                ; verifica se o
interruptor I1 esta para cima ou para baixo
                                TEST    M[FFF9h],R1
                                BR.Z    FimInt
                                MOV             R1,15d
                                MOV             M[Linhas],R1
                                MOV             R1,0200h
                                SUB             M[PosJ1],R1
                                SUB             M[PosJ2],R1
                                ADD             M[IniMatriz],R1
FimInt:                 POP             R1
                                RET

; ================================================================================
============
; VerPausa: vai verificar se o interruptor 7 esta para cima.
; Entradas: --
; Saidas: --
; Efeitos: Se o interruptor 7 estiver para cima desativa o temporizador
;                 e reinicia as flags relacionadas com o movimento e o relogio

; ================================================================================
=============
VerPausa:               MOV             R3,0080h
                                TEST    M[FFF9h],R3
                                BR.Z    FimPausa
                                MOV             M[FFF7h],R0
                                MOV             M[FlagTemp1],R0
                                MOV             M[FlagTemp2],R0
                                JMP             VerPausa
FimPausa:               MOV             M[FlagEsqJ1],R0
                                MOV             M[FlagDirJ1],R0
                                MOV             M[FlagEsqJ2],R0
                                MOV             M[FlagDirJ2],R0
                                RET

; ================================================================================
=================
;       ConvDecimal:  converte valores de hexadecimal para decimal
;       Entradas: R1 (pontuaÃ§ao dos jogadores)
;       Saidas: --
;   Efeitos: converte a pontuacao dos jogadores de hexadecimal para decimal

; ================================================================================
================
ConvDecimal:    PUSH    R2
                                PUSH    R4
                                MOV     R2, 0F00h
                                MOV             R3, R2
                                AND             R2,R1
;para so obter o ultimo digito da pontuacao
```



```
                                CMP             R2,0a00h
                                JMP.NZ          FimConv
                                MOV             R2, F000h
;para por a zero o numero mais a direita
                                AND             R1, R2
                                ADD             R1,1000h

FimConv:                POP     R4
                                POP             R2
                                RET
```