



DEEC

DEPARTAMENTO DE ENGENHARIA
ELETROTÉCNICA E DE COMPUTADORES

TÉCNICO LISBOA

Arquitectura de Computadores

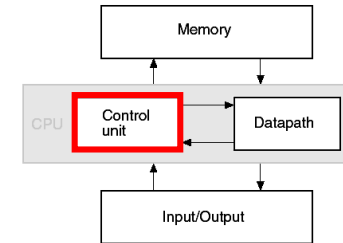
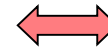
MEEC (2016/17 – 2º Sem.)

P3 – Processador e Periféricos

Prof. Nuno Horta

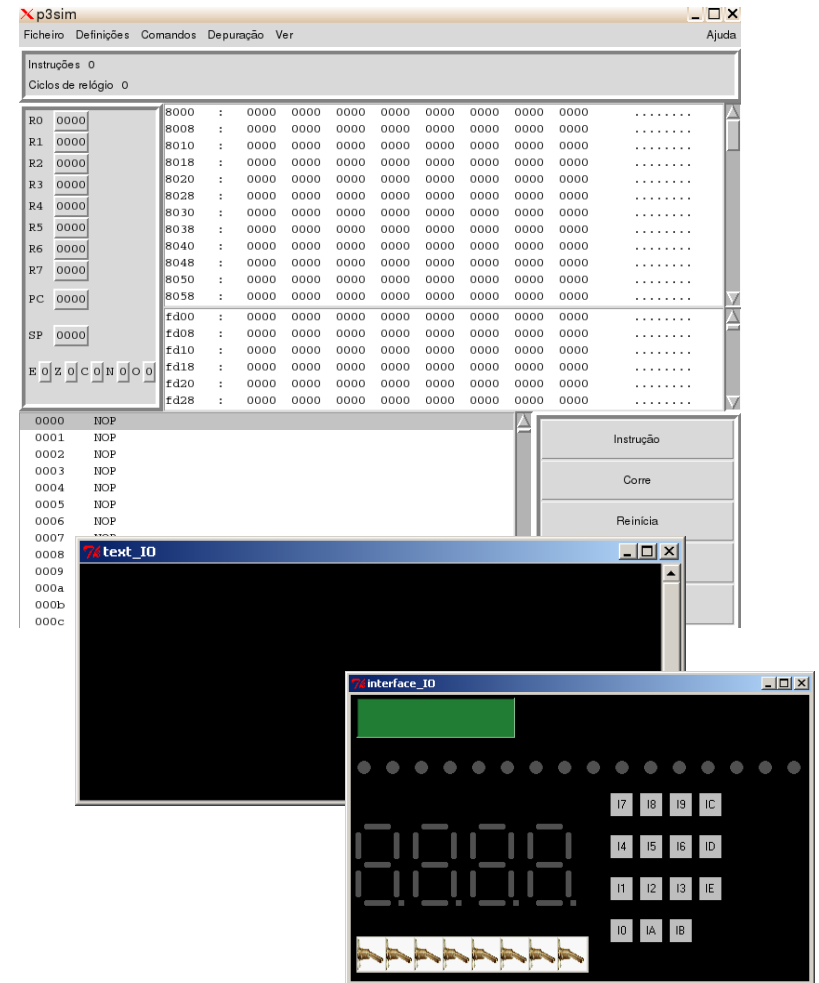
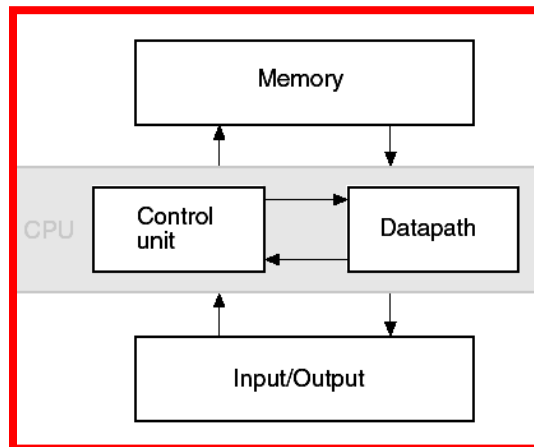
PLANEAMENTO

- ❑ *Introdução*
- ❑ *Unidade de Processamento*
- ❑ *Unidade de Controlo*
- ❑ *Arquitectura do Conjunto de Instruções*
- ❑ *Unidade Central de Processamento (CPU)*
- ❑ ***P3 – Processador e Periféricos***
- ❑ *Unidade de Entrada/Saída (I/O)*
- ❑ *Unidade de Memória*



SUMÁRIO

- ❑ **P3 - Processador**
 - ❑ **Arquitectura do Processador P3**
 - ❑ **Desenvolvimento de Programas**
 - ❑ **Simulador**
 - ❑ **Periféricos**



P3 - PROCESSADOR

Arquitectura do Processador P3

1. Registos

- **R0-R7 – Uso geral (R0 = 0)**
- **PC – Program Counter**
- **SP – Stack Pointer**
- **RE – Registo de Estado (EZCNO)**

2. Memória

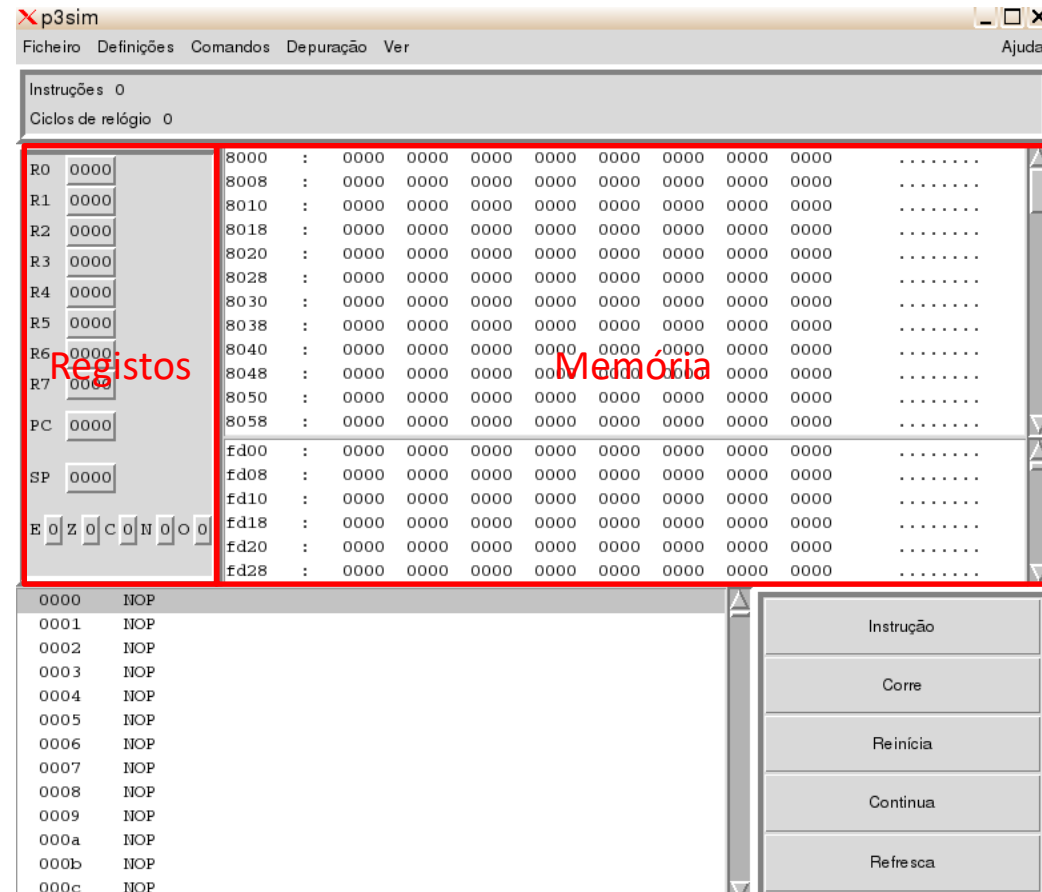
- **Espaço de endereçamento:**
64 KPalavras (2^{16})
- **Dimensão das palavras:**
16 bits

3. Entradas/Saídas

- **Endereçamento de I/O:**
Memory Mapped I/O
de FF00h em diante

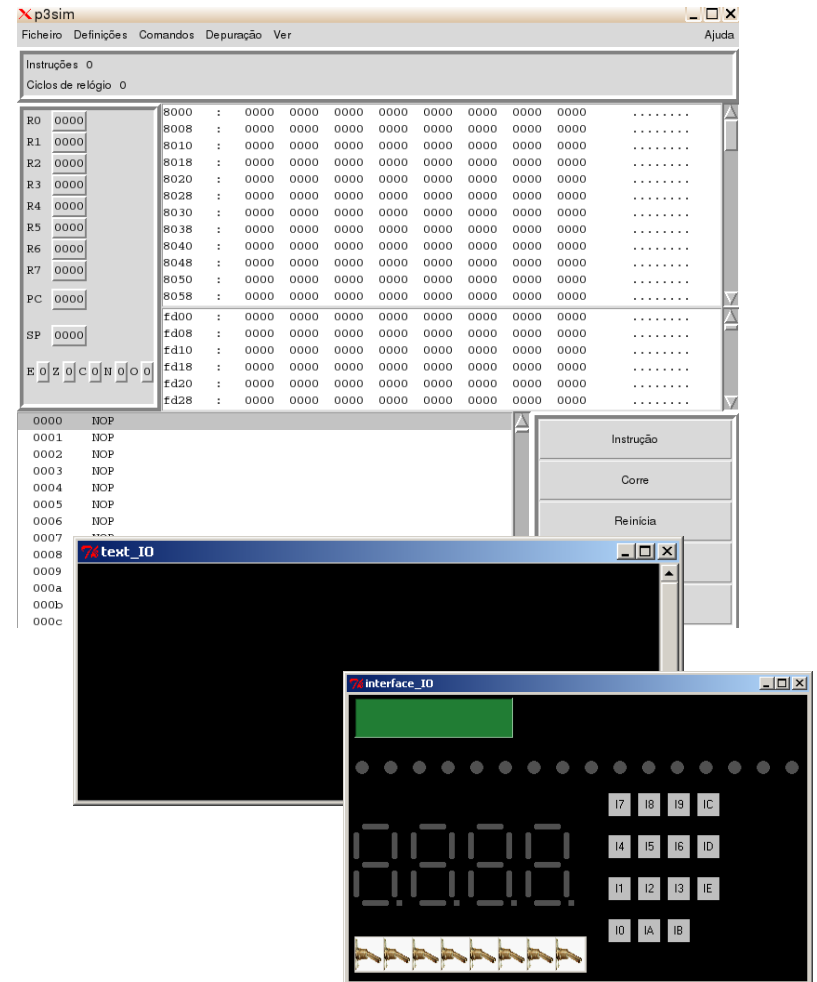
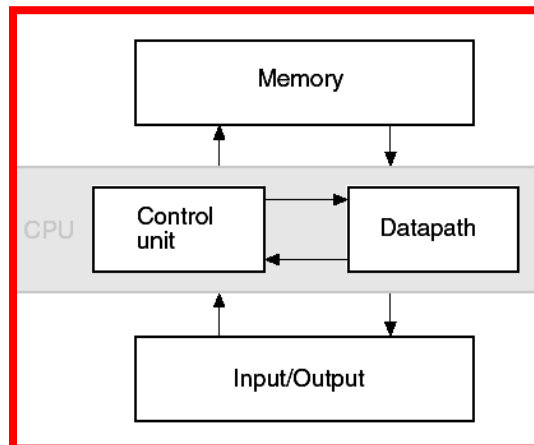
4. Interrupções

- **Máscara de Interrupções: FFFAh**
- **Tabela de Vectores de Interrupção: FE00h**



SUMÁRIO

- ❑ **P3 - Processador**
 - ❑ **Arquitectura do Processador P3**
 - ❑ **Desenvolvimento de Programas**
 - ❑ **Simulador**
 - ❑ **Periféricos**



P3 - PROCESSADOR

Desenvolvimento de Programas

1. Desenvolvimento de Programas (cont.):

1.3. Conjunto de Instruções

Pseudo	Aritméticas	Lógicas	Deslocamento	Controlo	Transfer.	Genéricas
ORIG	NEG	COM	SHR	BR	MOV	NOP
EQU	INC	AND	SHL	BR. cond	MVEH	ENI
WORD	DEC	OR	SHRA	JMP	MVEL	DSI
STR	ADD	XOR	SHLA	JMP. cond	XCH	STC
TAB	ADDC	TEST	ROR	CALL	PUSH	CLC
	SUB		ROL	CALL. cond	POP	CMC
	SUBB		RORC	RET		
	CMP		ROLC	RETN		
	MUL			RTI		
	DIV			INT		

1.4. Modos de Endereçamento

- **Registo** $op = Rx$
- **Registo Indirecto** $op = M[Rx]$
- **Imediato** $op = W$
- **Directo** $op = M[W]$
- **Indexado** $op = M[Rx+W]$
- **Relativo** $op = M[PC+W]$
- **Baseado** $op = M[SP+W]$

1.5. Etiquetas (Label):

Nome seguido de ':' utilizado para referenciar uma dada posição de memória.

```

=====
; ZONA III: Código
; conjunto de instruções Assembly, ordenadas de forma a realizar
; as funções pretendidas
=====

ORIG    0000h
JMP     inicio

;=====
; BubbleSort: Implementa algoritmo de ordenação "Bubble sort"
; Entradas: Sentido da Ordenação
; Sidas: ---
; Efeitos: Ordena vector de dados em memória
;=====
BubbleSort:  MOV     R6, M[SP+2] ; Sentido da ordenação
              MOV     R2, M[VDados]; Comprimento do Vector de Dados
              ADD     R2, VDados ; Último Elemento

SortLoop:    MOV     R3, 0 ; Indicador de troca (0 - não houve
troca, 1 - houve troca)

              MOV     R1, VDados ; Índice do vector de dados
              ADD     R1, 2

SwapLoop:    MOV     R4, M[R1-1] ; Inicia loop de trocas e lê dados
              MOV     R5, M[R1]
              CMP     R6, 1 ; Verifica tipo de ordenação
              BR.Z    IncOrder

DecOrder:    CMP     R4, R5
              BR.N    Swap
              BR      NoSwap

IncOrder:    CMP     R4, R5
              BR.P    Swap
              BR      NoSwap

Swap:        XCH     R4, R5 ; Executa troca
              MOV     M[R1-1], R4
              MOV     M[R1], R5
              MOV     R3, 1

NoSwap:      INC     R1

trocas       CMP     R1, R2 ; Verifica se chegou ao fim do loop de

              BR.NP   SwapLoop
              CMP     R3, 1 ; Verifica se vector está ordenado
              BR.Z    SortLoop
              RETN    1

```

```

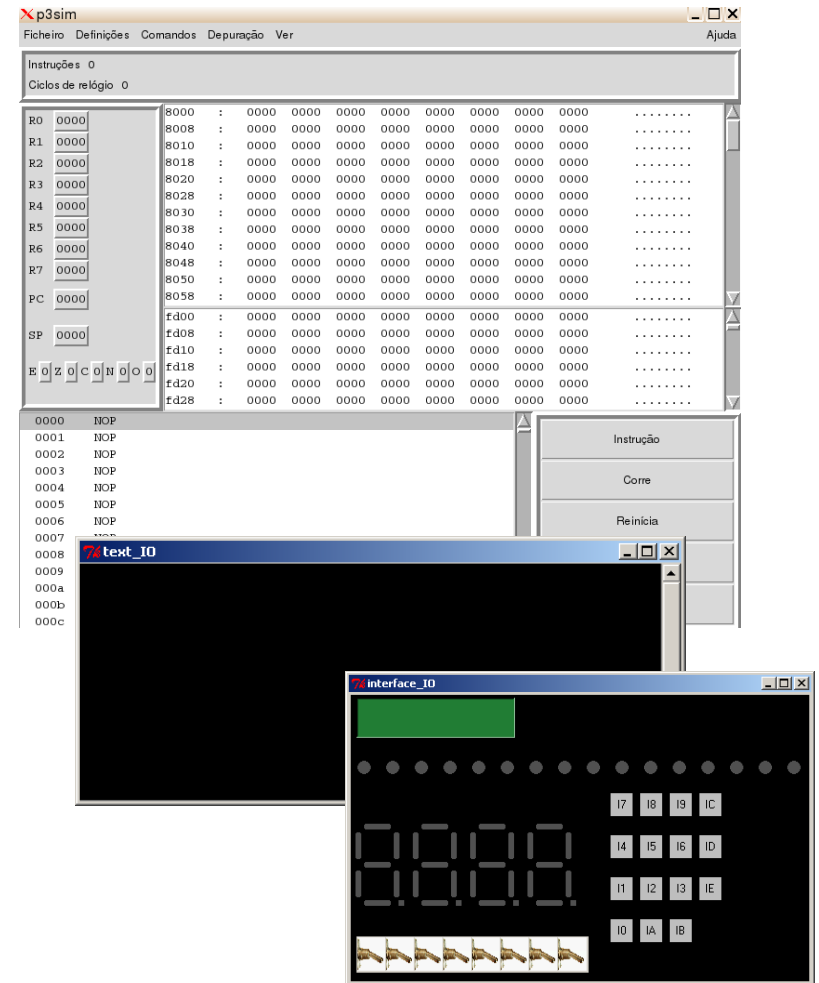
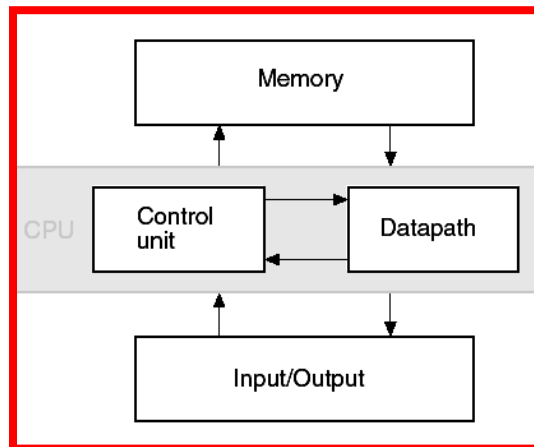
=====
; Programa principal
;=====
inicio:      MOV     R1, SP_INICIAL
              MOV     SP, R1

Orderna:     PUSH    1 ; Ordenação crescente
              CALL    BubbleSort
              PUSH    -1 ; Ordenação decrescente
              CALL    BubbleSort
              BR      Orderna

```

SUMÁRIO

- ❑ **P3 - Processador**
 - ❑ **Arquitectura do Processador P3**
 - ❑ **Desenvolvimento de Programas**
 - ❑ **Simulador**
 - ❑ **Periféricos**



P3 - PROCESSADOR

Simulador

2. Assemblador

2.1. Entrada

- Ficheiro com código Assembly – **filename.as**.

2.2. Comando

- **>p3as filename.as**

2.3. Saídas

- Ficheiro **filename.lis** contém o valor atribuído às referências usadas no programa assembly.
- Ficheiro **filename.exe** contém o código binário pronto a ser executado pelo simulador **p3sim**.

filename.as

ASSEMBLER

filename.lis

filename.exe

SIMULADOR

```
.....
; Programa BubbleSort.as
;
; Descrição: Implementação do Algoritmo de ordenação Bubble Sort para Assembly
; do P3
;
;   (1) Inicialização de Dados em Memória
;   (2) Ordenação de dados por ordem crescente
;   (3) Ordenação de dados por ordem decrescente
;
;
; Autor: Nuno Horta
; Data: 04/2013
; Última Alteração: 22/04/2013
;
;=====
; ZONA I: Definição de constantes
;
;   Pseudo-instrução: EQU
;=====
; STACK POINTER
SP_INICIAL    EQU    FFFFh
;=====
; ZONA II: Definição de variáveis
;
;   Pseudo-instrução: WORD - palavra (16 bits)
;   STR - sequência de caracteres.
;   Cada carácter ocupa 1 palavra
;=====
;
;   ORG 8000h
;
; VDados     STR    20,1,3,5,7,9,2,4,6,8,10,12,14,16,18,20,11,13,15,17,19
;=====
```

BubbleSort.lis

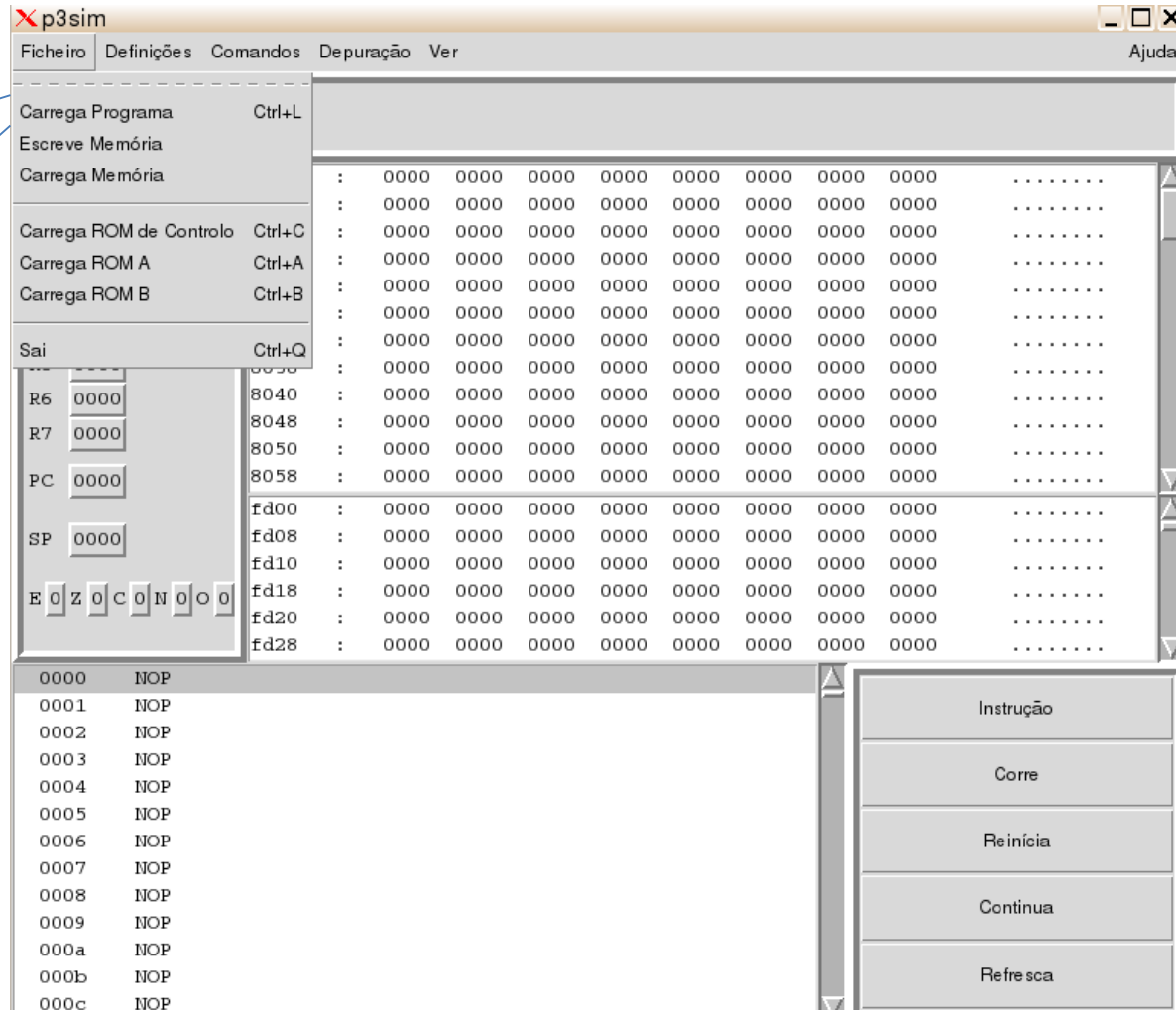
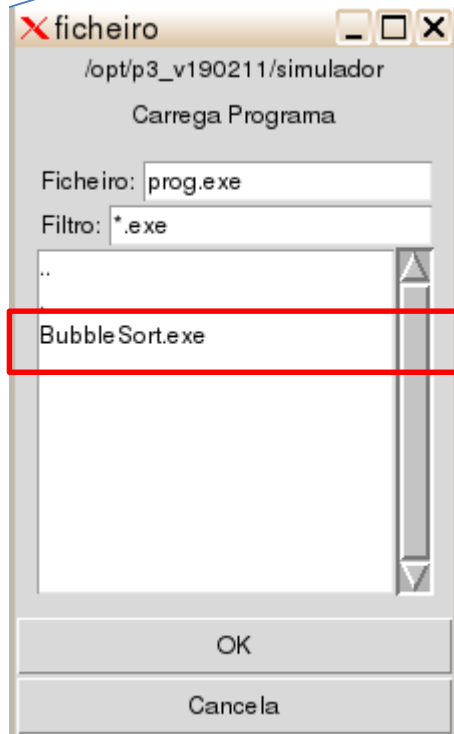
GLOBAL REFERENCES

Name	Value	Type
=====	=====	=====
SP_INICIAL	FDDF	CONSTANT
VDados	8000	STRING
BubbleSort	0002	LABEL
SortLoop	0008	LABEL
SwapLoop	000E	LABEL
DecOrder	0014	LABEL
IncOrder	0017	LABEL
Swap	001A	LABEL
NoSwap	0020	LABEL
inicio	0027	LABEL
Ordena	002A	LABEL

P3 - PROCESSADOR

Simulador

- Carregamento de Programa



P3 - PROCESSADOR

Simulador

Carregamento de Programa

```

; Programa BubbleSort.as
;
; Descricao: Implementação do Algoritmo de ordenação Bubble Sort para Assembly
do P3
;
; (1) Inicialização de Dados em Memória
; (2) Ordenação de dados por ordem crescente
; (3) Ordenação de dados por ordem decrescente
;
;
; Autor: Nuno Horta
; Data: 04/2013
; Última Alteração: 22/04/2013
;
;-----
; ZONA I: Definição de constantes
;
; Pseudo-instruções: EQU
;
;-----
; STACK POINTER
SP_INICIAL EQU FDFH
;
;-----
; ZONA II: Definição de variáveis
;
; Pseudo-instruções: WORD - palavra (16 bits)
; STR - sequência de caracteres.
;
; Cada carácter ocupa 1 byte
;
;-----
; Dados
ORIG 8000h
STR 20,1,3,5,7,9,2,4,6,8,10,12,14,16,18,20,11,13,15,17,19

```

DADOS

```

; ZONA III: Código
;
; conjunto de instruções Assembly, ordenadas de forma a realizar
; as funções pretendidas
;
;-----
;
; ORIG 0000h
; JMP inicio
;
;-----
; BubbleSort: Implementa algoritmo de ordenação "Bubble sort"
; Entradas: Sentido da Ordenação
; Saídas: ---
; Efeitos: Ordena vector de dados em memória
;
BubbleSort: MOV R6, M[SP+2] ; Sentido da ordenação
;
; MOV R6, M[SP+2] ; Sentido da ordenação
; ADD R2, VDados ; Último elemento do vector de Dados
;
SortLoop: MOV R3, 0 ; Indicador de troca (0 - não houve troca,
; 1 - houve troca)
;
; MOV R1, VDados ; Índice do vector de dados
; ADD R1, 2
;
SwapLoop: MOV R4, M[R1-1] ; Inicia loop de trocas e lê dados
; MOV R5, M[R1]
;
; CME R6, 1 ; Verifica tipo de ordenação
; BR.Z IncOrder
;
DecOrder: CMP R4, R5
; BR.N Swap
; BR NoSwap

```

PROGRAMA

x3sim

Ficheiro Definições Comandos Depuração Ver Ajuda

Instruções 0

Ciclos de relógio 0

R0	0000	8000	:	0014	0001	0003	0005	0007	0009	0002	0004
R1	0000	8008	:	0006	0008	000a	000c	000e	0010	0012	0014
R2	0000	8010	:	000b	000d	000f	0011	0013	0000	0000	0000
R3	0000	8018	:	0000	0000	0000	0000	0000	0000	0000	0000
R4	0000	8020	:	0000	0000	0000	0000	0000	0000	0000	0000
R5	0000	8028	:	0000	0000	0000	0000	0000	0000	0000	0000
R6	0000	8030	:	0000	0000	0000	0000	0000	0000	0000	0000
R7	0000	8038	:	0000	0000	0000	0000	0000	0000	0000	0000
PC	0000	8040	:	0000	0000	0000	0000	0000	0000	0000	0000
SP	0000	8048	:	0000	0000	0000	0000	0000	0000	0000	0000
		8050	:	0000	0000	0000	0000	0000	0000	0000	0000
		8058	:	0000	0000	0000	0000	0000	0000	0000	0000
		fd00	:	0000	0000	0000	0000	0000	0000	0000	0000
		fd08	:	0000	0000	0000	0000	0000	0000	0000	0000
		fd10	:	0000	0000	0000	0000	0000	0000	0000	0000
		fd18	:	0000	0000	0000	0000	0000	0000	0000	0000
		fd20	:	0000	0000	0000	0000	0000	0000	0000	0000
		fd28	:	0000	0000	0000	0000	0000	0000	0000	0000

DADOS

0000	JMP	0027
0002	MOV	R6, M[SP+2]
0004	MOV	R2, M[8000]
0006	ADD	R2, 8000
0008	MOV	R3, 0000
000a	MOV	R1, 8000
000c	ADD	R1, 0002
000e	MOV	R4, M[R1-1]
0010	MOV	R5, M[R1]
0011	CMP	R6, 0001
0013	BR.Z	3
0014	CMP	R4, R5
0015	BR.N	4

PROGRAMA

Instrução
Corre
Reinicia
Continua
Refresca

P3 - PROCESSADOR Simulador

- O código carregado no simulador substitui todas as **constantes** e **etiquetas** pelos respectivos valores.

	JMP	inicio	0000	JMP	0027
BubbleSort:	MOV	R6, M[SP+2]	0002	MOV	R6, M[SP+2]
	MOV	R2, M[VDados]	0004	MOV	R2, M[8000]
	ADD	R2, VDados	0006	ADD	R2, 8000
SortLoop:	MOV	R3, 0	0008	MOV	R3, 0000
	MOV	R1, VDados	000a	MOV	R1, 8000
	ADD	R1, 2	000c	ADD	R1, 0002
SwapLoop:	MOV	R4, M[R1-1]	000e	MOV	R4, M[R1-1]
	MOV	R5, M[R1]	0010	MOV	R5, M[R1]
	CMP	R6, 1	0011	CMP	R6, 0001
	BR.Z	IncOrder	0013	BR.Z	3
DecOrder:	CMP	R4, R5	0014	CMP	R4, R5
	BR.N	Swap	0015	BR.N	4
	BR	NoSwap	0016	BR	9
IncOrder:	CMP	R4, R5	0017	CMP	R4, R5
	BR.P	Swap	0018	BR.P	1
	BR	NoSwap	0019	BR	8
Swap:	XCH	R4, R5	001a	XCH	R4, R5
	MOV	M[R1-1], R4	001b	MOV	M[R1-1], R4
	MOV	M[R1], R5	001d	MOV	M[R1], R5
	MOV	R3, 1	001e	MOV	R3, 0001
NoSwap:	INC	R1	0020	INC	R1
	CMP	R1, R2	0021	CMP	R1, R2
	BR.NP	SwapLoop	0022	BR.NP	-21
	CMP	R2, 1	0023	CMP	R3, 0001
	BR.Z	SortLoop	0025	BR.Z	-30
	RETN	1	0026	RETN	1
inicio:	MOV	R1, SP_INICIAL	0027	MOV	R1, fdff
	MOV	SP, R1	0029	MOV	SP, R1
Ordena:	PUSH	1	002a	PUSH	0001
	CALL	BubbleSort	002c	CALL	0002
	PUSH	-1	002e	PUSH	ffff
	CALL	BubbleSort	0030	CALL	0002
	BR	Ordena	0032	BR	-9
			0033	NOP	

PROGRAMA

CÓDIGO
CARREGADO NO P3

P3 - PROCESSADOR

Simulador

- Execução e Debug do Programa

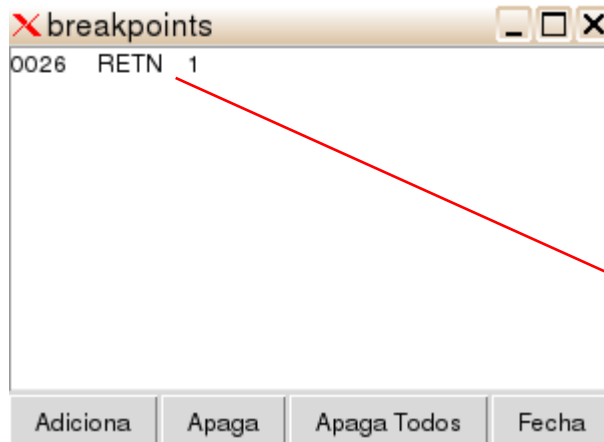
Execução Completa

Execução Instrução-a-Instrução

Pontos de Paragem

Exemplo:

ordenação ascendente



The screenshot shows the main p3sim simulator window. It has a menu bar with 'Ficheiro', 'Definições', 'Comandos', 'Depuração', and 'Ver'. The interface is divided into several sections:

- Registers (REGISTOS):** A vertical list on the left showing registers R0 through R7 and the PC (Program Counter) and SP (Stack Pointer). R0 is 0000, R1 is 8015, R2 is 8014, R3 is 0000, R4 is 0013, R5 is 0014, R6 is 0001, R7 is 0000, and PC is 0026. SP is fdff.
- Data Memory (DADOS):** A table on the right showing memory addresses from 8000 to fd28. Each address is followed by a colon and a 16-bit hexadecimal value. The values are mostly 0000, with some non-zero values at 8008 (0008), 8010 (0010), 8012 (0012), 8014 (0000), 8018 (0000), 8020 (0000), 8028 (0000), 8030 (0000), 8038 (0000), 8040 (0000), 8048 (0000), 8050 (0000), 8058 (0000), fd00 (0000), fd08 (0000), fd10 (0000), fd18 (0000), fd20 (0000), and fd28 (0000).
- Instruction Memory (PROGRAMA):** A table at the bottom showing instruction addresses from 0021 to 0033. Each address is followed by a colon and the instruction. The instructions are: 0021: CMP R1, R2; 0022: BR.NP -21; 0023: CMP R3, 0001; 0025: BR.Z -30; 0026: RETN 1 (highlighted in red); 0027: MOV R1, fdff; 0029: MOV SP, R1; 002a: PUSH 0001; 002c: CALL 0002; 002e: PUSH ffff; 0030: CALL 0002; 0032: BR -9; 0033: NOP.
- Control Panel:** A vertical stack of buttons on the right: 'Instrução', 'Corre', 'Reinicia', 'Continua', and 'Refresca'.

**Número de Instruções
Executadas**

**Número de Ciclos de Relógio
durante a execução do
Programa (Arquitetura de
Ciclo Múltiplo, porquê?)**

Exemplo:

ordenação descendente

The screenshot shows the p3sim simulator interface. The top menu bar includes 'Ficheiro', 'Definições', 'Comandos', 'Depuração', and 'Ver'. The status bar at the top right shows 'Ajuda'.

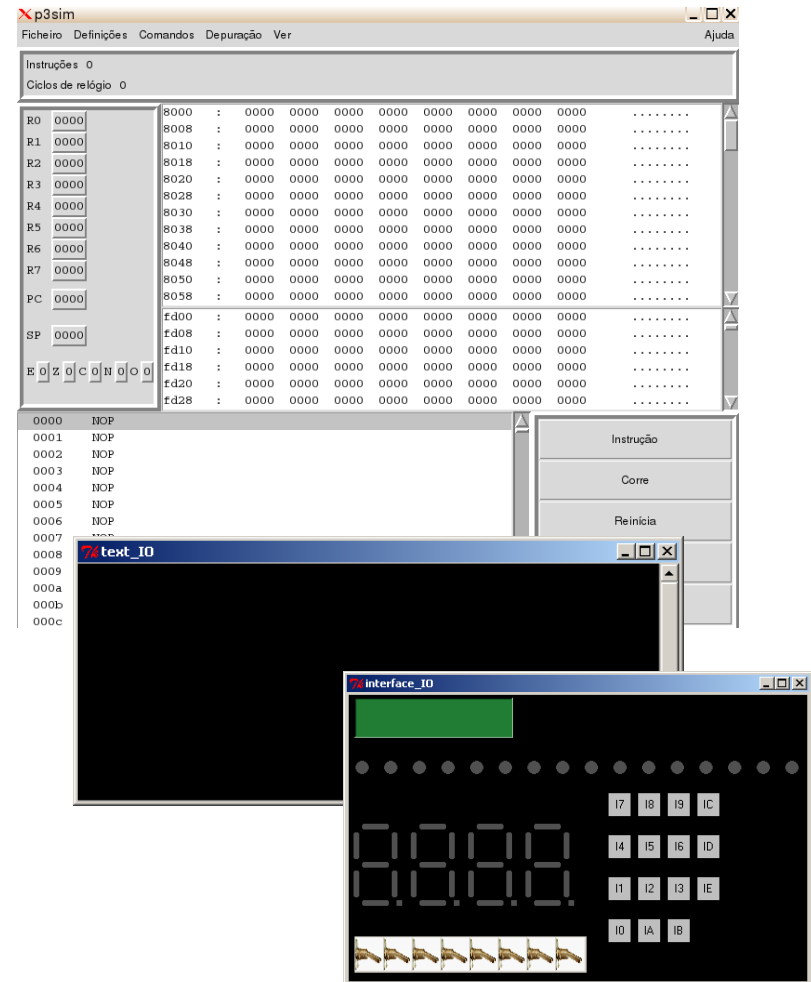
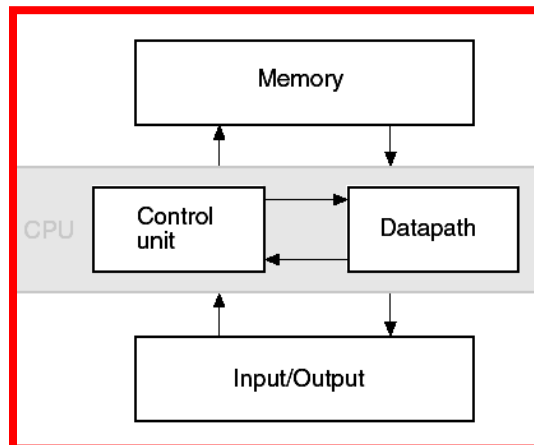
The main display area is divided into several sections:

- Registers:** A list of registers (R0 to R7) with their current values. R0 is 0000, R1 is 8015, R2 is 8014, R3 is 0000, R4 is 0002, R5 is 0001, R6 is ffff, and R7 is 0000. The PC (Program Counter) is 0026 and the SP (Stack Pointer) is fdff.
- Memory:** A table showing memory addresses and their contents. The addresses range from 8000 to fd28. The contents are shown in hexadecimal. A red box highlights the memory area, and the word 'DADOS' is written in red over it.
- Instruction Execution:** A table showing the sequence of instructions being executed. The instructions are:
 - 0021: CMP R1, R2
 - 0022: BR.NP -21
 - 0023: CMP R3, 0001
 - 0025: BR.Z -30
 - >>0026: RETN 1 (highlighted in red)
 - 0027: MOV R1, fdff
 - 0029: MOV SP, R1
 - 002a: PUSH 0001
 - 002c: CALL 0002
 - 002e: PUSH ffff
 - 0030: CALL 0002
 - 0032: BR -9
 - 0033: NOP
- Control Panel:** A vertical panel on the right side with buttons for 'Instrução', 'Corre', 'Reinicia', 'Continua', and 'Refre sca'.

A red box highlights the instruction execution table, and the word 'PROGRAMA' is written in red over it.

SUMÁRIO

- ❑ **P3 - Processador**
 - ❑ **Arquitectura do Processador P3**
 - ❑ **Desenvolvimento de Programas**
 - ❑ **Simulador**
 - ❑ **Periféricos**



P3 - PROCESSADOR

Periféricos

1. Registos

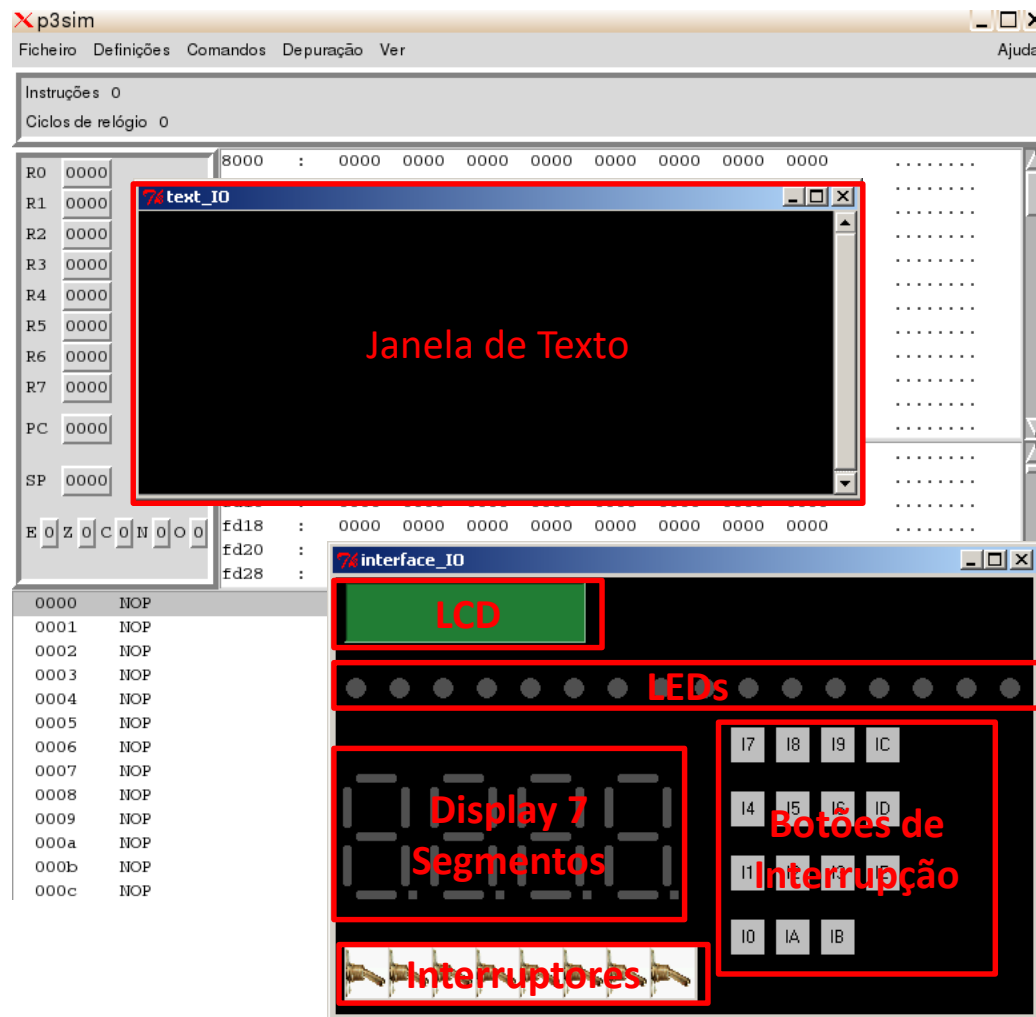
2. Memória

3. Entradas/Saídas

- **Endereçamento de I/O:**
Memory Mapped I/O de FF00h em diante
- **Janela de Texto:** FFFCh a FFFFh
- **Interruptores:** FFF9h
- **LEDs:** FFF8h
- **LCD:** FFF4h a FFF5h
- **Display de 7 Segmentos:**
FFF0h a FFF3h
- **Temporizador:** FFF6h a FFF7h

4. Interrupções

- **Máscara de Interrupções:** FFFAh
- **Tabela de Vectors de Interrupção:**
FE00h a FE0Fh



P3 - PROCESSADOR

Periféricos – Janela de Texto

Janela de Texto: FFFCh a FFFFh

- **FFFCh** - Permite colocar o cursor numa dada posição da janela.
- **FFFDh** - Permite testar se houve alguma tecla premida.
- **FFFEh** - Permite escrever um caracter na janela.
- **FFFFh** - Permite ler a última tecla premida.

XY para escrita
(Linha Coluna)

```
=====
; Programa Demol.as
;
; Descricao: Demonstracao da utilização da janela de texto
;
; Autor: Nuno Horta
; Data: 28/05/2013                               Ultima Alteracao:28/05/2013 |
=====

; ZONA I: Definicao de constantes
; Pseudo-instrucao : EQU
=====
; STACK POINTER
SP_INICIAL      EQU      FDFFFh

; I/O a partir de FF00H
IO_CURSOR       EQU      FFFCh
IO_WRITE        EQU      FFFEh

LIMPAR_JANELA   EQU      FFFFh
XY_INICIAL      EQU      0614h
FIM_TEXTO       EQU      '@'

=====
; ZONA II: Definicao de variaveis
; Pseudo-instrucoes : WORD - palavra (16 bits)
;                      STR  - sequencia de caracteres.
; Cada caracter ocupa 1 palavra
=====
ORIG      8000h
```

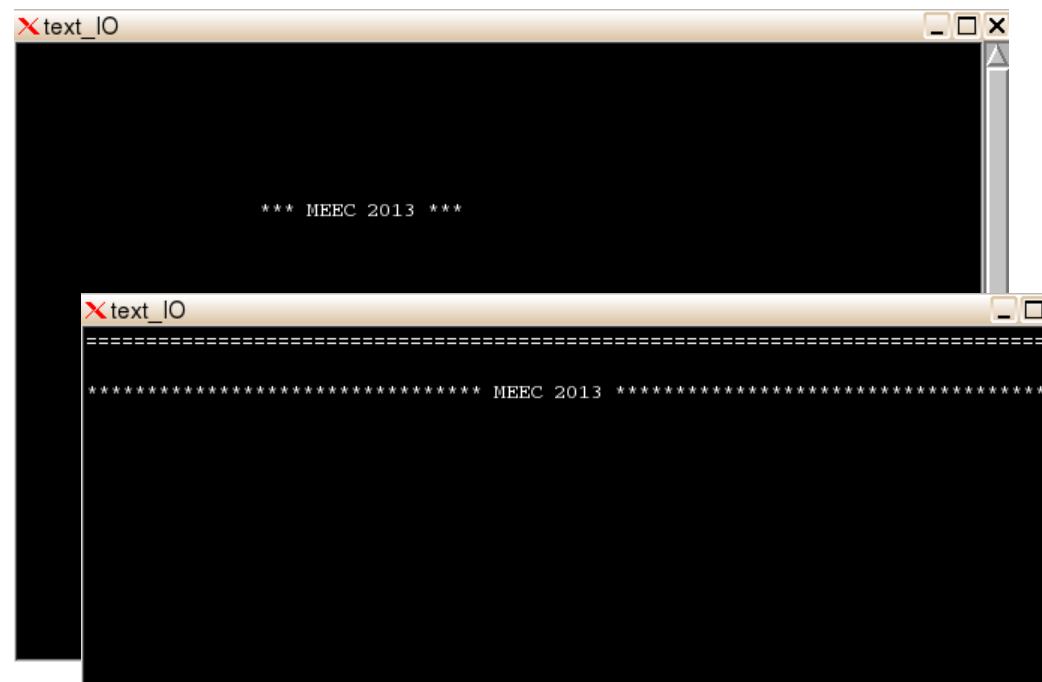
P3 - PROCESSADOR

*Zona de Dados inicializada com texto a
Enviar para a janela de texto.*

- *Utilização de símbolo para definir
fim da string.*
- *Etiquetas/Labels utilizados para
identificar endereço de começo da string.*
- *Ilustração do output a visualizar
durante a execução do programa de teste.*

Periféricos – Janela de Texto

```
=====
; ZONA II: Definicao de variaveis
;           Pseudo-instrucoes : WORD - palavra (16 bits)
;                               STR  - sequencia de caracteres.
;           Cada caracter ocupa 1 palavra
=====
                                ORIG      8000h
VarTexto1                      STR        '*** MEEC 2013 ***',FIM_TEXTO
VarTexto2                      STR        '=====
FIM_TEXTO
VarTexto3                      STR        '***** MEEC 2013 *****
FIM_TEXTO
```



P3 - PROCESSADOR

Periféricos – Janela de Texto

Zona de Código:

- **Rotina LimpaJanela utilizada para apagar todos os caracteres da janela.**
- **Rotina EscCar para escrita do carater na posição da janela de texto onde se encontra o cursor.**

(Passagem de parâmetros por Registo)

(PUSHs e POPs para preservar e recuperar as variáveis alteradas durante a rotina)

```
=====
; ZONA III: Codigo
; conjunto de instrucoes Assembly, ordenadas de forma a realizar
; as funcoes pretendidas
=====

        ORIG    0000h
        JMP     inicio

=====
; LimpaJanela: Rotina que limpa a janela de texto.
; Entradas: --
; Sidas: ---
; Efeitos: ---
=====
LimpaJanela:    PUSH R2
                MOV     R2, LIMPAR_JANELA
                MOV     M[IO_CURSOR], R2
                POP  R2
                RET

=====
; EscCar: Rotina que efectua a escrita de um carater para o ecran.
; 0 carater pode ser visualizado na janela de texto.
; Entradas: R1 - Carater a escrever
; Sidas: ---
; Efeitos: alteracao da posicao de memoria M[IO]
=====
EscCar:         MOV     M[IO_WRITE], R1
                RET

=====
```

Zona de Código (cont.):

- **Rotina EscStr para escrita de uma string de caracteres na posição da janela de texto onde se encontra o cursor.**

(Passagem de parâmetros pelo STACK)

(PUSHs e POPs para preservar e recuperar as variáveis alteradas durante a rotina)

```

=====
; EscString: Rotina que efectua a escrita de uma cadeia de caracter, terminada
; pelo caracter FIM_TEXTO, na janela de texto numa posicao
; especificada. Pode-se definir como terminador qualquer caracter
; ASCII.
; Entradas: pilha - posicao para escrita do primeiro carater
;           pilha - apontador para o inicio da "string"
; Saidas: ---
; Efeitos: ---
=====
EscString:    PUSH    R1
              PUSH    R2
              PUSH    R3
              MOV     R2, M[SP+6]    ; Apontador para inicio da "string"
              MOV     R3, M[SP+5]    ; Localizacao do primeiro carater
Ciclo:        MOV     M[IO_CURSOR], R3
              MOV     R1, M[R2]
              CMP     R1, FIM_TEXTO
              BR.Z    FimEsc
              CALL    EscCar
              INC     R2
              INC     R3
              BR      Ciclo
FimEsc:       POP     R3
              POP     R2
              POP     R1
              RETN    2              ; Actualiza STACK
=====
    
```

Programa Principal:

- *Inicialização do STACK, obrigatória antes da sua utilização.*
- *Escreve 1 String > Limpa Janela > Escreve 2 Strings*

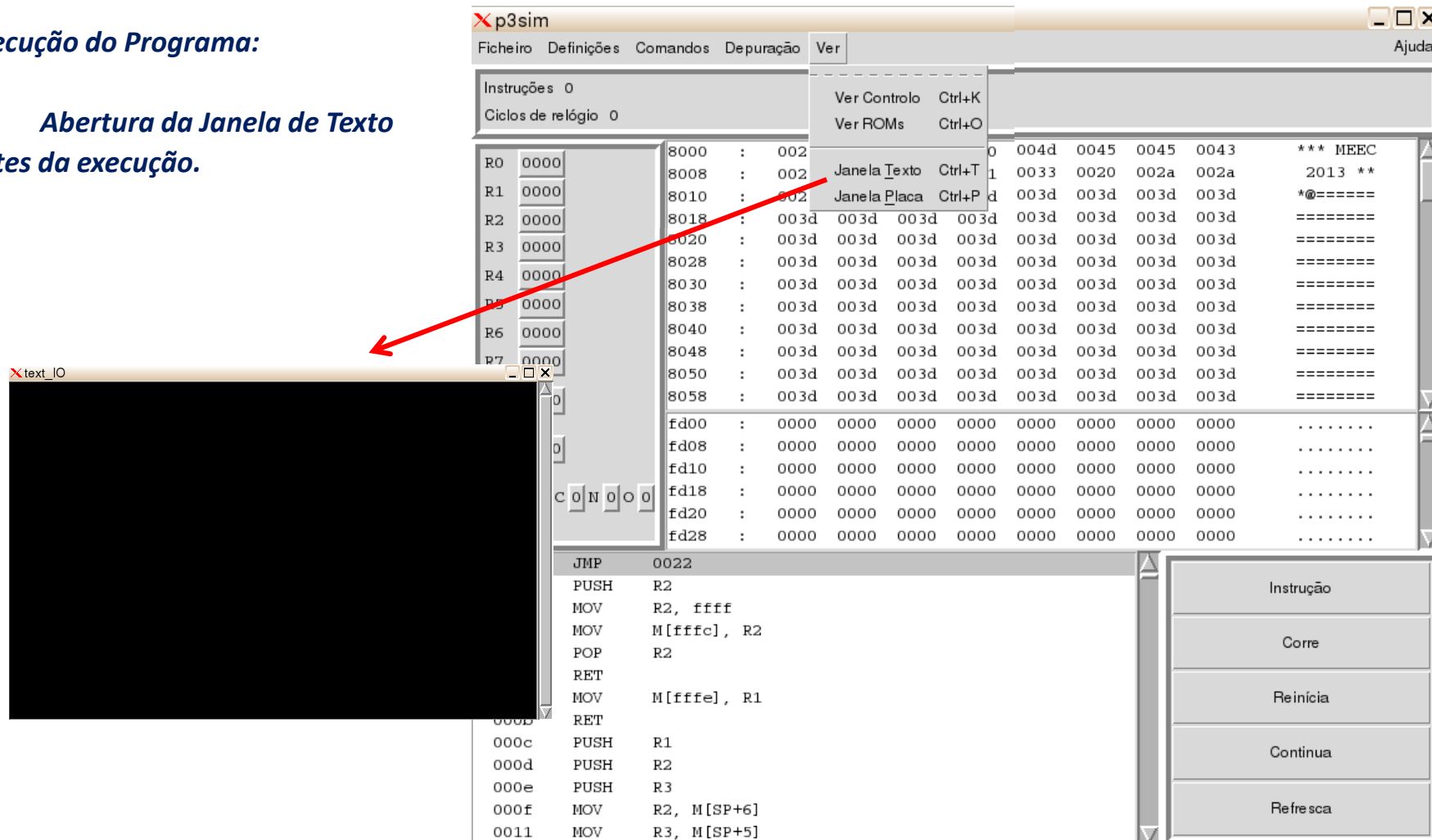
```
=====
;
;                               Programa principal
;
=====
inicio:      MOV     R1, SP_INICIAL
             MOV     SP, R1
             CALL    LimpaJanela
             MOV     R1, XY_INICIAL
             PUSH    VarTextol           ; Passagem de parametros pelo STACK
             PUSH    R1                 ; Passagem de parametros pelo STACK
             CALL    EscString
             CALL    LimpaJanela
             MOV     R1, 0000h
             PUSH    VarTexto2          ; Passagem de parametros pelo STACK
             PUSH    R1                 ; Passagem de parametros pelo STACK
             CALL    EscString
             ADD     R1, 0200h
             PUSH    VarTexto3          ; Passagem de parametros pelo STACK
             PUSH    R1                 ; Passagem de parametros pelo STACK
             CALL    EscString
Fim:         BR      Fim
=====
```

P3 - PROCESSADOR

Periféricos – Janela de Texto

Execução do Programa:

- **Abertura da Janela de Texto antes da execução.**

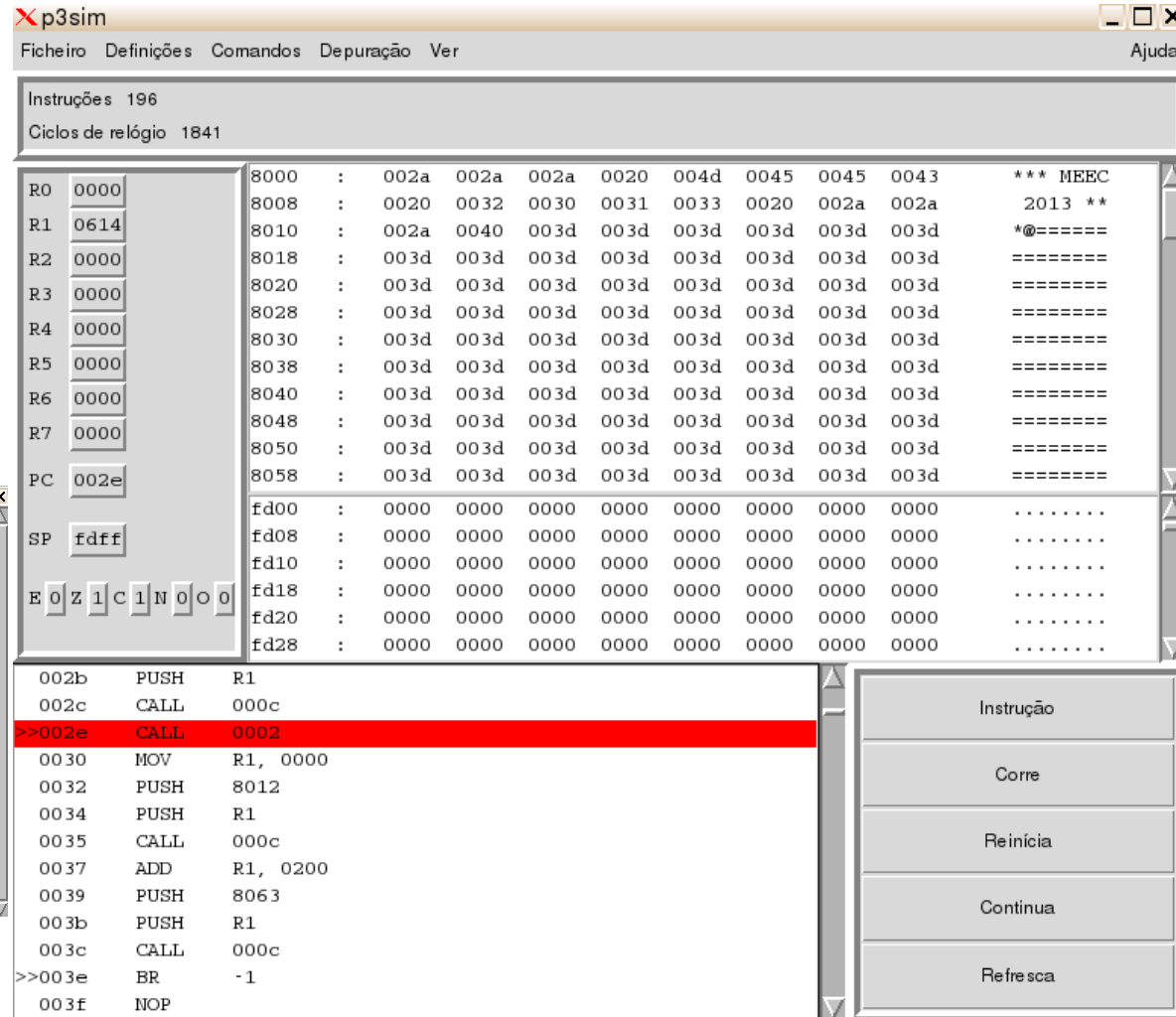
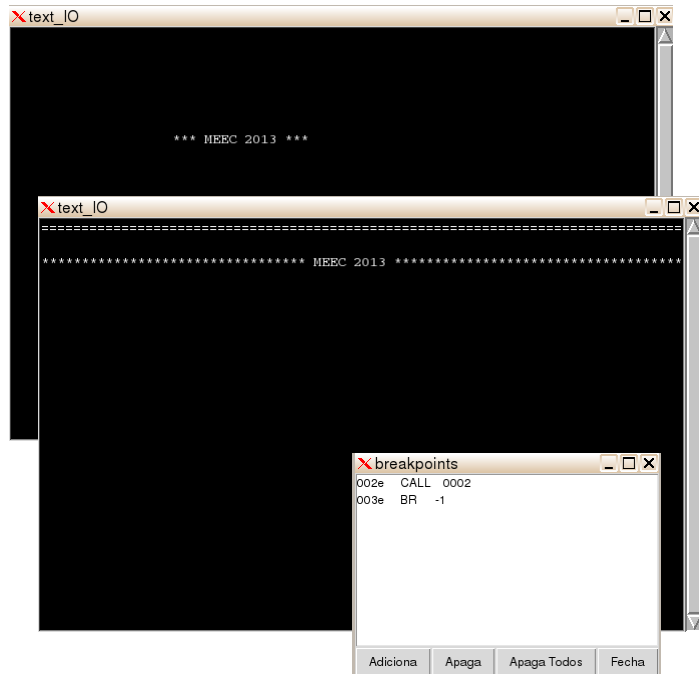


P3 - PROCESSADOR

Periféricos – Janela de Texto

Execução do Programa (cont):

- Utilização de breakpoints.



P3 - PROCESSADOR

Periféricos – Display 7 Seg.

Display de 7 Segmentos: FFF0h a FFF3h

- **FFF0h – (Display 7 segmentos 0)**
Permite escrever no display de 7 segmentos mais à direita. Só são considerados os 4 bits menos significativos escritos no endereço.
- **FFF1h – (Display 7 segmentos 1)**
- **FFF2h – (Display 7 segmentos 2)**
- **FFF3h – (Display 7 segmentos 3)**

```
=====
; Programa Display7seg.as
;
; Descricao: Demonstracao da utilizacao dos interruptores
;
; Autor: Nuno Horta
; Data: 28/05/2013
;
;=====
; ZONA I: Definicao de constantes
; Pseudo-instrucao : EQU
;=====

; TEMPORIZACAO
DELAYVALUE      EQU      0300h

; STACK POINTER
SP_INICIAL      EQU      FDFh

; I/O a partir de FF00h
DISP7S1         EQU      FFF0h
DISP7S2         EQU      FFF1h
DISP7S3         EQU      FFF2h
LCD_WRITE       EQU      FFF5h
LCD_CURSOR      EQU      FFF4h
LEDS            EQU      FFF8h
INTERRUPTORES   EQU      FFF9h
```


P3 - PROCESSADOR

Zona de Código:

- **Rotina EscDisplay utilizada para escrever no display selecionado.**
- **Rotina Delay utilizada para provocar um atraso (ver interrupções para soluções Alternativas - Temporizador).**

Programa Principal:

- **Ciclo para escrita dos valores 1, 2, 4 e 8 no display da direita.**

Periféricos – Display 7 Seg.

```
=====
; ZONA III: Código
; conjunto de instruções Assembly, ordenadas de forma a realizar
; as funções pretendidas
=====

                ORIG    0000h
                JMP     inicio

=====
; EscDisplay: Rotina que efectua escrita no DISPLAY de 7 segmentos
; Entradas: R1 - Valor a enviar para o porto do DISPLAY
;           R2 - Porto do DISPLAY a utilizar
; Sidas: ---
; Efeitos: alteração da posição de memória/porto M[R2]
=====
EscDisplay:     MOV     M[R2], R1
                RET

=====
; EscDisplay: Rotina que permite gerar um atraso
; Entradas: ---
; Sidas: ---
; Efeitos: ---
=====
Delay:          PUSH    R1
                MOV     R1, DELAYVALUE
DelayLoop:      DEC     R1
                BR.NZ   DelayLoop
                POP     R1
                RET

=====
; Programa principal
=====
inicio:         MOV     R1, SP_INICIAL
                MOV     SP, R1
                MOV     R2, DISP7S1
                MOV     R3, DISP7S2

ResetCont:      MOV     R1, 0001h
CicloCont:      CALL    EscDisplay          ; Passagem de parâmetros por registo
                CALL    Delay
                ROL     R1, 1
                TEST    R1, 0010h
                BR.Z    CicloCont
                BR      ResetCont
```


Interruptores: FFF9h

- **FFF9h** - Permite ler, nos 8 bits menos significativos, o valor definido pela posição dos interruptores. O interruptor da direita corresponde ao bit menos significativo.

Execução do Programa:

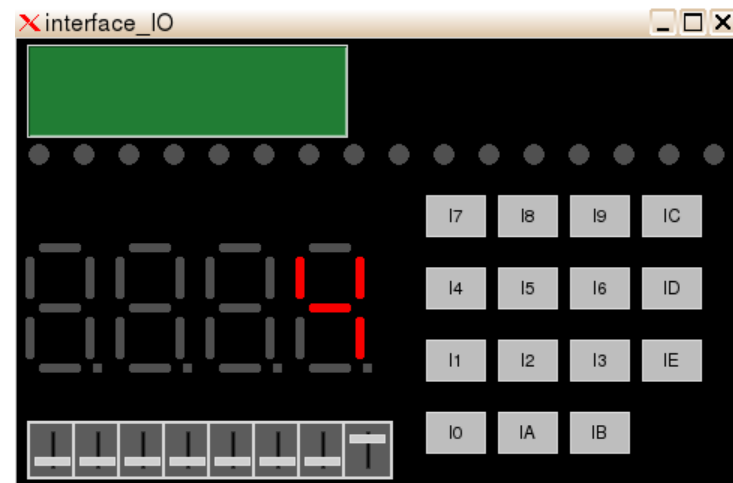
- A ativação do interruptor da direita bloqueia a contagem no display de 7 segmentos.

```

=====
;
;                                     Programa principal
;=====
inicio:      MOV     R1, SP_INICIAL
             MOV     SP, R1
             MOV     R2, DISP7S1
             MOV     R3, DISP7S2

ResetCont:   MOV     R1, 0001H
CicloCont:   CALL    EscDisplay          ; Passagem de parametros por registo
             CALL    Delay

Stop:        MOV     R5, M[INTERRUPTORES]
             TEST    R5, 0001H
             BR.NZ   Stop
             ROL     R1, 1
             TEST    R1, 0010H
             BR.Z    CicloCont
             BR      ResetCont
=====
    
```



P3 - PROCESSADOR

Periféricos – LEDs

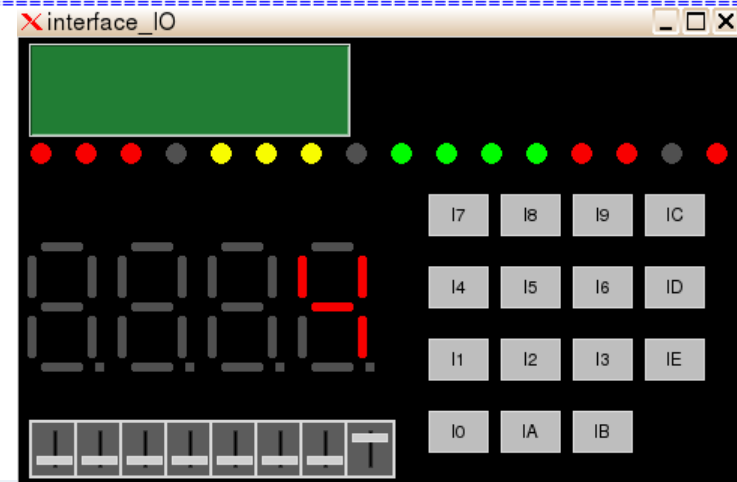
LEDs: FFF8h

- *Permite acender os LEDs correspondentes ao valor em binário que se escreve no endereço. O LED da direita corresponde ao bit menos significativo.*

Execução do Programa:

- *A contagem em R4 é representada nos LEDs*

```
=====
;
;                               Programa principal
;
=====
inicio:      MOV     R1, SP_INICIAL
             MOV     SP, R1
             MOV     R2, DISP7S1
             MOV     R3, DISP7S2
             MOV     R4, 0001H
ResetCont:   MOV     R1, 0001H
CicloCont:   CALL    EscDisplay           ; Passagem de parametros por registo
             CALL    Delay
Stop:        MOV     R5, M[INTERRUPTORES]
             INC     R4
             MOV     M[LEDS], R4
             TEST    R5, 0001H
             BR.NZ   Stop
             ROL     R1, 1
             TEST    R1, 0010H
             BR.Z    CicloCont
             BR      ResetCont
=====
```



P3 - PROCESSADOR

Periféricos – LCD

LCD: FFF4h a FFF5h

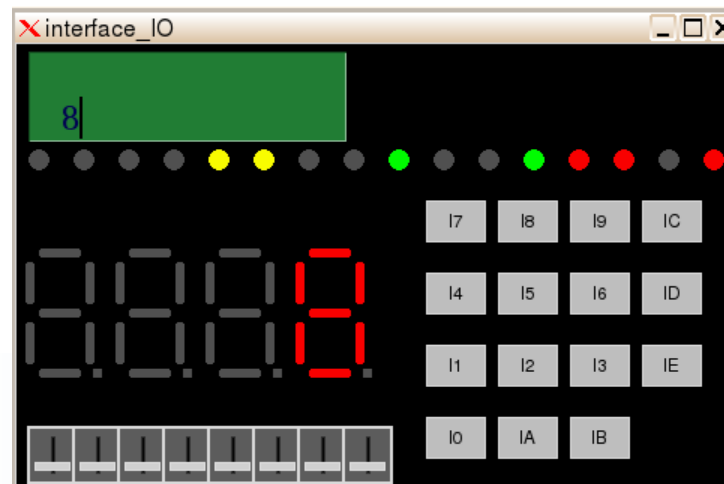
- **FFF4h** – Permite enviar sinais de controlo para o LCD. (Bit 15 ativa LCD, Bit 5 Limpa LCD, Bit 4 representa a linha, Bits 3-0 indicam a coluna).
- **FFF5h** – Permite escrever um caracter no LCD cujo código ASCII foi escrito no endereço.

Zona de Código:

- Rotina **CLRLCD** utilizada para limpar LCD
- Rotina **EscLCD** utilizada para escrever no LCD

```
=====
; CLRLCD: Rotina para limpar LCD
;
;      Entradas:
;      Sidas:
;      Efeitos:
=====
CLRLCD:      PUSH    R1
             MOV     R1, 8020h
             MOV     M[LCD_CURSOR], R1
             POP     R1
             RET

=====
; ESCLCD: Rotina para escrita no LCD
;
;      Entradas: R4, R1
;      Sidas:
;      Efeitos:
=====
EscLCD:      PUSH    R1
             PUSH    R2
             MOV     R1, M[SP+4]
             MOV     R2, M[SP+5]
             MOV     M[LCD_CURSOR], R1
             ADD     R2, 0030h      ; Converte valor para ASCII
             MOV     M[LCD_WRITE], R2
             POP     R2
             POP     R1
             RETN    2
```

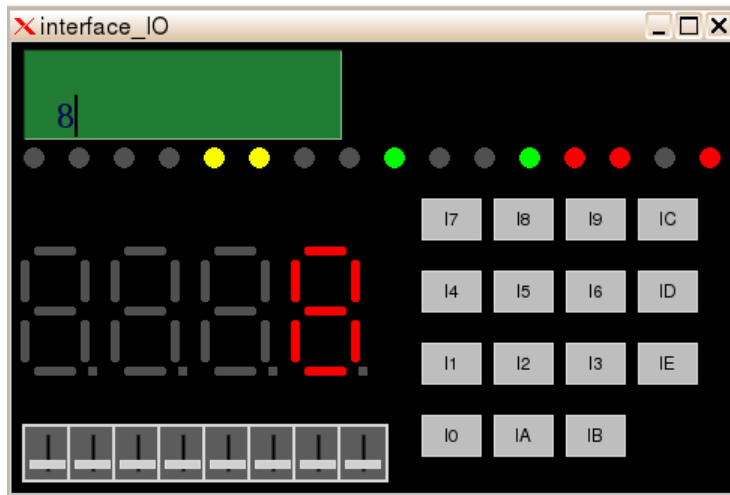


P3 - PROCESSADOR

Periféricos – LCD

Execução do Programa

- *Escreve no LCD o mesmo valor que é escrito no display de 7 segmentos.*



```
=====
;
;                               Programa principal
;
=====
inicio:      MOV     R1, SP_INICIAL
             MOV     SP, R1
             MOV     R2, DISP7S1
             MOV     R3, DISP7S2
             MOV     R4, 0001H

ResetCont:   MOV     R1, 0001H
CicloCont:   CALL    EscDisplay          ; Passagem de parametros por registo
             CALL    Delay
Stop:        MOV     R5, M[INTERRUPTORES]
             INC     R4
             MOV     M[LEDS], R4
             TEST    R5, 0001H
             BR.NZ   Stop
             PUSH    R1
             PUSH    8014h
             CALL    EscLCD
             ROL     R1, 1
             TEST    R1, 0010H
             BR.Z    CicloCont
             CALL    CLR_LCD
             BR       ResetCont
=====
```

Interrupções e Temporizador

- As **interrupções são simuladas através dos botões de pressão** (interrupções de 0 a 14) e do temporizador (interrupção 15).
- A cada interrupção ativa deve corresponder **uma rotina de atendimento** da interrupção.
- A localização (endereço) das rotinas de interrupção deve ser guardado na **Tabela de Vectores de Interrupção (TVI)**. A TVI corresponde às posições de memória FE00h a FEOFh, correspondendo o endereço FE00h à posição de memória onde é guardado o endereço de atendimento da interrupção **0**, e assim sucessivamente.
- A **Máscara de Interrupções** (FFFAh) permite seleccionar as interrupções que se pretendem aceitar (cada interrupção é permitida colocando a 1 o respetivo bit da posição de memória indicada por este endereço)
- As interrupções só são aceites depois de se fazer o **enable das interrupções**, executando a instrução ENI.
- **Temporizador: FFF6h a FFF7h**
 - FFF6h - uma escrita para este endereço define o número de unidades de contagem, cada com a duração de 100ms. Por exemplo, para se ter um intervalo de 1s, deve ser escrito para endereço o valor 10. Uma leitura deste endereço permite obter o valor atual de contagem;
 - FFF7h - este porto permite dar início ou parar uma contagem por escrita, respetivamente, de um 1 ou um 0 no bit menos significativo (os restantes bits são ignorados). Uma leitura deste endereço indica, no bit menos significativo, o estado do temporizador, em contagem ou parado.

Zona de Código:

- **Inicialização da TVI** – Coloca os endereços das rotinas nas respetivas posições da TVI
- **Inicialização da Máscara de Interrupções** – Permite apenas as interrupções 15 (temporizador) e as interrupções 0 e 1 dos botões de pressão.
- **Inicialização do Temporizador** – inicializa o temporizador com 2s e inicia a contagem.
- **Enable das Interrupções** – só depois desta instrução é que são aceites interrupções.

```

=====
;InitInt: Inicializa TVI, Mascara de INT e Temporizador
;
;      Entradas: ---
;      Sidas: ---
;      Efeitos: ---
=====
InitInt:      MOV      R1, RotinaInt0
              MOV      M[TAB_INT0], R1
              MOV      R1, RotinaInt1
              MOV      M[TAB_INT1], R1
              MOV      R1, RotinaIntTemp
              MOV      M[TAB_INTTemp], R1
              MOV      R1,8003h
              MOV      M[MASCARA_INT], R1
              MOV      R1,0020h
              MOV      M[TempValor], R1
              MOV      R1,0001h
              MOV      M[TempControlo], R1
              ENI
              RET
=====
    
```


Zona de Código:

- **RotinaInt0** – Troca o display de 7 segmentos onde se faz a escrita do valor da contagem.
- **RotinaInt1** – Acende todos os LEDs
- **RotinaIntTemp** – Acende os LEDs centrais e reprograma o temporizador.

```

=====
; RotinaInt0: Rotina de interrupcao 0
;           Entradas: R2, R3 - Portos dos Display de 7 Segmentos
;           Saidas:   R2, R3
;           Efeitos: (1) Comutacao de R2 e R3
;
=====
RotinaInt0:  XCH      R2, R3
            RTI

;
=====
; RotinaInt1: Rotina de interrupcao 1
;           Entradas: R4
;           Saidas:   R4
;           Efeitos: (1) Acende todos os LEDS
;
=====
RotinaInt1:  MOV      R4, FFFFh
            MOV      M[LEDS], R4
            RTI

;
=====
; RotinaIntTemp: Rotina de interrupcao do temporizador|
;           Entradas: R4
;           Saidas:   R4
;           Efeitos: (1) Activa Leds com valor de R4
;                   (2) Reprograma temporizador
;
=====
RotinaIntTemp:  PUSH    R1
               MOV      R4, 0FF0h
               MOV      M[LEDS], R4
               MOV      R1, 0020h
               MOV      M[TempValor], R1
               MOV      R1, 0001h
               MOV      M[TempControlo], R1
               POP      R1
               RTI
=====

```

Programa Principal

- *Inicialização do STACK*
- *Inicialização das interrupções*
- *Escrita na Janela de Texto*
- *Escrita no Display de 7 seg.*
- *Escrita no LCD*
- *Utilização de Interruptores*
- *Ativação dos LEDs*

```

=====
;
;                                     Programa principal
;
=====
inicio:      MOV     R1, SP_INICIAL
             MOV     SP, R1
             CALL    InitInt
             CALL    LimpaJanela
             MOV     R1, 0000h
             PUSH    VarTexto2           ; Passagem de parametros pelo STACK
             PUSH    R1                 ; Passagem de parametros pelo STACK
             CALL    EscString
             ADD     R1, 0200h
             PUSH    VarTexto3           ; Passagem de parametros pelo STACK
             PUSH    R1                 ; Passagem de parametros pelo STACK
             CALL    EscString
             MOV     R2, DISP7S1
             MOV     R3, DISP7S2
             MOV     R4, 0001H
ResetCont:   MOV     R1, 0001H
CicloCont:   CALL    EscDisplay           ; Passagem de parametros por registo
             CALL    Delay
Stop:        MOV     R5, M[INTERRUPTORES]
             INC     R4
             MOV     M[LEDS], R4
             TEST    R5, 0001H
             BR.NZ   Stop
             PUSH    R1
             PUSH    8014h
             CALL    EscLCD
             ROL     R1, 1
             TEST    R1, 0010H
             BR.Z    CicloCont
             CALL    CLRLCD
             BR      ResetCont
=====

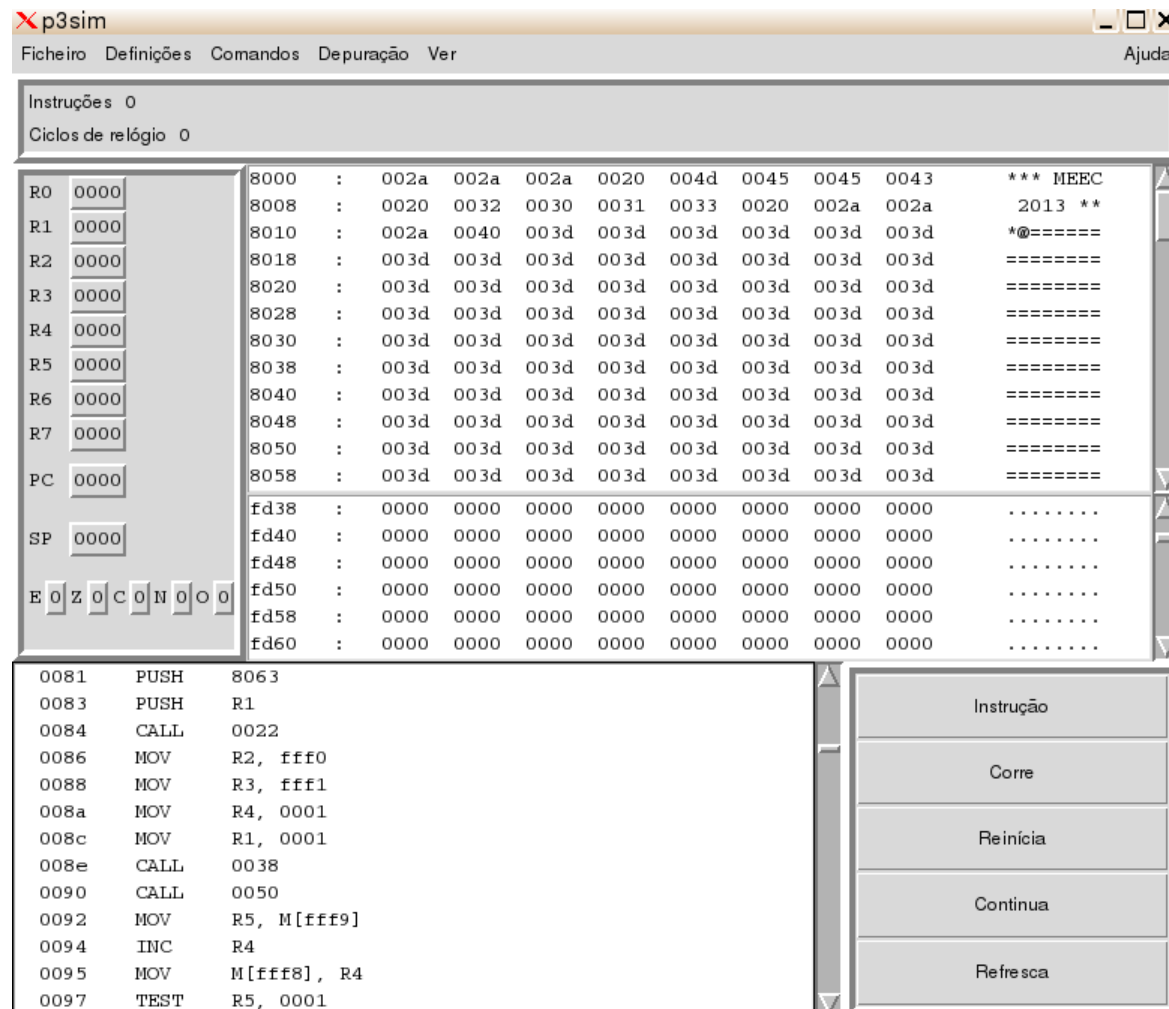
```

P3 - PROCESSADOR

Interrupções

Execução do Programa:

- Definição de Breakpoints**



P3 - PROCESSADOR

Interrupções

Execução do Programa:

- *Entrada na rotina de inicialização das interrupções*

p3sim

Ficheiro Definições Comandos Depuração Ver Ajuda

Instruções 3

Ciclos de relógio 27

R0	0000	8000	:	002a	002a	002a	0020	004d	0045	0045	0043	*** MEEC
R1	fdff	8008	:	0020	0032	0030	0031	0033	0020	002a	002a	2013 **
R2	0000	8010	:	002a	0040	003d	003d	003d	003d	003d	003d	*@=====
R3	0000	8018	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R4	0000	8020	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R5	0000	8028	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R6	0000	8030	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R7	0000	8038	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
PC	0074	8040	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
SP	fdff	8048	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
E	0	8050	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
Z	0	8058	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
C	0	fd38	:	0000	0000	0000	0000	0000	0000	0000	0000
N	0	fd40	:	0000	0000	0000	0000	0000	0000	0000	0000
O	0	fd48	:	0000	0000	0000	0000	0000	0000	0000	0000
O	0	fd50	:	0000	0000	0000	0000	0000	0000	0000	0000
O	0	fd58	:	0000	0000	0000	0000	0000	0000	0000	0000
O	0	fd60	:	0000	0000	0000	0000	0000	0000	0000	0000

006f	ENI	
0070	RET	
0071	MOV	R1, fdff
0073	MOV	SP, R1
>>0074	CALL	0057
>>0076	CALL	0018
0078	MOV	R1, 0000
007a	PUSH	8012
007c	PUSH	R1
007d	CALL	0022
007f	ADD	R1, 0200
0081	PUSH	8063
0083	PUSH	R1

Instrução

Corre

Reinicia

Continua

Refresca

P3 - PROCESSADOR

Interrupções

Execução do Programa:

- *TVI Programada na rotina de inicialização das interrupções.*
- *FLAG E ativa*

Flags: E 1 Z 0 C 0 N 0 O 0

Registers: R0: 0000, R1: 0001, R2: 0000, R3: 0000, R4: 0000, R5: 0000, R6: 0000, R7: 0000, PC: 0076, SP: fdff

Instructions:

Address	Instruction	Value
8000	:	002a 002a 002a 0020 004d 0045 0045 0043 *** MEEC
8008	:	0020 0032 0030 0031 0033 0020 002a 002a 2013 **
8010	:	002a 0040 003d 003d 003d 003d 003d 003d *@=====
8018	:	003d 003d 003d 003d 003d 003d 003d 003d =====
8020	:	003d 003d 003d 003d 003d 003d 003d 003d =====
8028	:	003d 003d 003d 003d 003d 003d 003d 003d =====
8030	:	003d 003d 003d 003d 003d 003d 003d 003d =====
8038	:	003d 003d 003d 003d 003d 003d 003d 003d =====
8040	:	003d 003d 003d 003d 003d 003d 003d 003d =====
8048	:	003d 003d 003d 003d 003d 003d 003d 003d =====
8050	:	003d 003d 003d 003d 003d 003d 003d 003d =====
8058	:	003d 003d 003d 003d 003d 003d 003d 003d =====
fd00	:	0002 0004 0000 0000 0000 0000 0000 0000v
fd08	:	0000 0000 0000 0000 0000 0000 0000 0009v
fd10	:	0000 0000 0000 0000 0000 0000 0000 0000v

Instruction List:

Address	Instruction	Value
0070	RET	
0071	MOV	R1, fdff
0073	MOV	SP, R1
>>0074	CALL	0057
>>0076	CALL	0018
0078	MOV	R1, 0000
007a	PUSH	8012
007c	PUSH	R1
007d	CALL	0022
007f	ADD	R1, 0200
0081	PUSH	8063
0083	PUSH	R1
0084	CALL	0022

Control Panel:

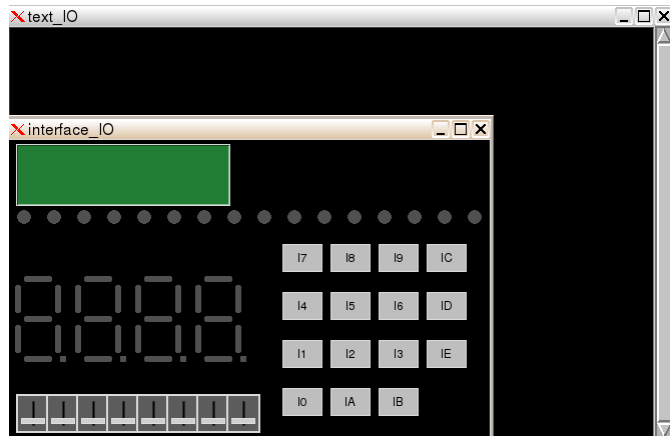
- Instrução
- Corre
- Reinicia
- Continua
- Refresca

P3 - PROCESSADOR

Interrupções

Execução do Programa:

- **Entrada na rotina de atendimento do temporizador.**
- **Flag E a 0, Interrupções desativadas enquanto executa rotina de atendimento a uma interrupção.**



p3sim

Ficheiro Definições Comandos Depuração Ver Ajuda

Instruções 19

Ciclos de relógio 203

R0	0000	8000	:	002a	002a	002a	0020	004d	0045	0045	0043	*** MEEC
R1	0001	8008	:	0020	0032	0030	0031	0033	0020	002a	002a	2013 **
R2	0000	8010	:	002a	0040	003d	003d	003d	003d	003d	003d	*@=====
R3	0000	8018	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R4	0000	8020	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R5	0000	8028	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R6	0000	8030	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R7	0000	8038	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
PC	0009	8040	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
SP	fdfc	8048	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
E	0	8050	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
Z	0	8058	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
C	0	fd38	:	0000	0000	0000	0000	0000	0000	0000	0000
N	0	fd40	:	0000	0000	0000	0000	0000	0000	0000	0000
O	0	fd48	:	0000	0000	0000	0000	0000	0000	0000	0000
O	0	fd50	:	0000	0000	0000	0000	0000	0000	0000	0000
O	0	fd58	:	0000	0000	0000	0000	0000	0000	0000	0000
O	0	fd60	:	0000	0000	0000	0000	0000	0000	0000	0000

```

>>0003 RTI
>>0004 MOV R4, ffff
0006 MOV M[fff8], R4
>>0008 RTI
>>0009 PUSH R1
000a MOV R4, 0ff0
000c MOV M[fff8], R4
000e MOV R1, 0020
0010 MOV M[fff6], R1
0012 MOV R1, 0001
0014 MOV M[fff7], R1
0016 POP R1
>>0017 RTI
    
```

Instrução

Corre

Reinicia

Continua

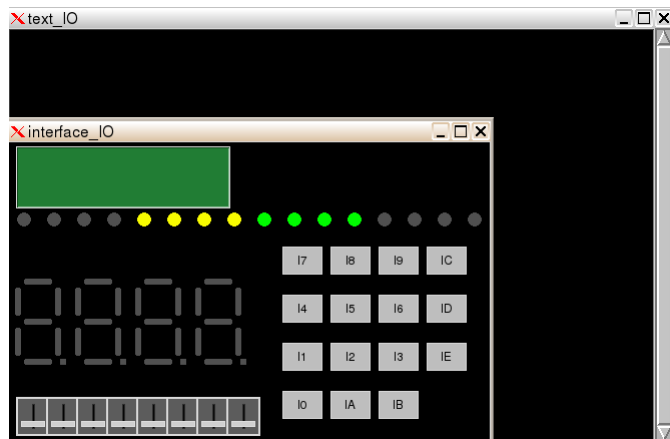
Refre sca

P3 - PROCESSADOR

Interrupções

Execução do Programa:

- Ativação dos LEDS centrais e reprogramação do temporizador.



p3sim

Ficheiro Definições Comandos Depuração Ver Ajuda

Instruções 27
Ciclos de relógio 288

R0	0000	8000	:	002a	002a	002a	0020	004d	0045	0045	0043	*** MEEC
R1	0001	8008	:	0020	0032	0030	0031	0033	0020	002a	002a	2013 **
R2	0000	8010	:	002a	0040	003d	003d	003d	003d	003d	003d	*@=====
R3	0000	8018	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R4	0ff0	8020	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R5	0000	8028	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R6	0000	8030	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R7	0000	8038	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
PC	0017	8040	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
SP	fdfc	8048	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
E	0	8050	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
Z	0	8058	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
C	0	fd38	:	0000	0000	0000	0000	0000	0000	0000	0000
N	0	fd40	:	0000	0000	0000	0000	0000	0000	0000	0000
O	0	fd48	:	0000	0000	0000	0000	0000	0000	0000	0000
O	0	fd50	:	0000	0000	0000	0000	0000	0000	0000	0000
O	0	fd58	:	0000	0000	0000	0000	0000	0000	0000	0000
O	0	fd60	:	0000	0000	0000	0000	0000	0000	0000	0000

0010	MOV	M[fff6], R1
0012	MOV	R1, 0001
0014	MOV	M[fff7], R1
0016	POP	R1
>>0017	RTI	
0018	PUSH	R2
0019	MOV	R2, ffff
001b	MOV	M[fffc], R2
001d	POP	R2
001e	RET	
001f	MOV	M[fffe], R1
0021	RET	
0022	PUSH	R1

Instrução

Corre

Reinicia

Continua

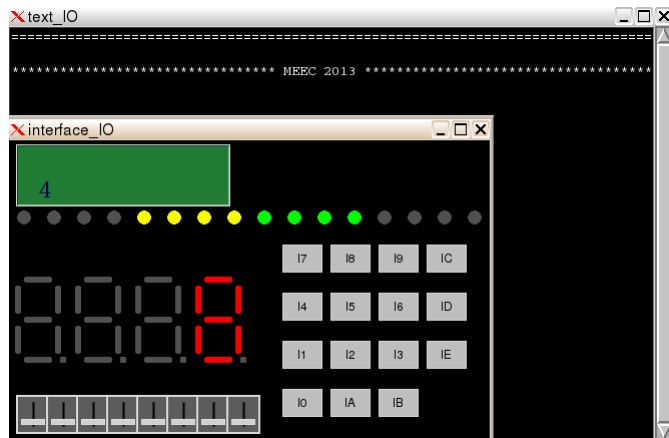
Refre sca

P3 - PROCESSADOR

Interrupções

Execução do Programa:

- *Entrada na rotina de atendimento do temporizador, após início da contagem (ver Display de 7 seg. e LCD).*



p3sim

Ficheiro Definições Comandos Depuração Ver Ajuda

Instruções 18957

Ciclos de relógio 196829

R0	0000	8000	:	002a	002a	002a	0020	004d	0045	0045	0043	*** MEEC
R1	02f4	8008	:	0020	0032	0030	0031	0033	0020	002a	002a	2013 **
R2	ffff0	8010	:	002a	0040	003d	003d	003d	003d	003d	003d	*@======
R3	ffff1	8018	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R4	0ff0	8020	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R5	0000	8028	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R6	0000	8030	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R7	0000	8038	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
PC	0017	8040	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
SP	fdfb	8048	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
		8050	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
		8058	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
		fd38	:	0000	0000	0000	0000	0000	0000	0000	0000
		fd40	:	0000	0000	0000	0000	0000	0000	0000	0000
		fd48	:	0000	0000	0000	0000	0000	0000	0000	0000
		fd50	:	0000	0000	0000	0000	0000	0000	0000	0000
		fd58	:	0000	0000	0000	0000	0000	0000	0000	0000
		fd60	:	0000	0000	0000	0000	0000	0000	0000	0000

E 0 Z 0 C 0 N 0 O 0

0010	MOV	M[fff6], R1
0012	MOV	R1, 0001
0014	MOV	M[fff7], R1
0016	POP	R1
>0017	RTI	
0018	PUSH	R2
0019	MOV	R2, ffff
001b	MOV	M[fffc], R2
001d	POP	R2
001e	RET	
001f	MOV	M[fffe], R1
0021	RET	
0022	PUSH	R1

Instrução

Corre

Reinicia

Continua

Refresca

P3 - PROCESSADOR

Interrupções

Execução do Programa:

- Entrada na rotina de atendimento à interrupção 1.

The screenshot displays the p3sim simulator interface with the following components:

- Registers and PC:**
 - R0: 0000, R1: 006f, R2: fff0, R3: fff1, R4: 0ff5, R5: 0000, R6: 0000, R7: 0000
 - PC: 0004, SP: fdfb
- Memory Dump:**

8000	:	002a	002a	002a	0020	004d	0045	0045	0043	*** MEEC
8008	:	0020	0032	0030	0031	0033	0020	002a	002a	2013 **
8010	:	002a	0040	003d	003d	003d	003d	003d	003d	*@=====
8018	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
8020	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
8028	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
8030	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
8038	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
8040	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
8048	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
8050	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
8058	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
fd00	:	0000	0000	0000	0000	0000	0000	0000	0000
fd08	:	0000	0000	0000	0000	0000	0000	0000	0000
fd10	:	0000	0000	0000	0000	0000	0000	0000	0000
fd18	:	0000	0000	0000	0000	0000	0000	0000	0000
fd20	:	0000	0000	0000	0000	0000	0000	0000	0000
fd28	:	0000	0000	0000	0000	0000	0000	0000	0000
- Instruction List:**

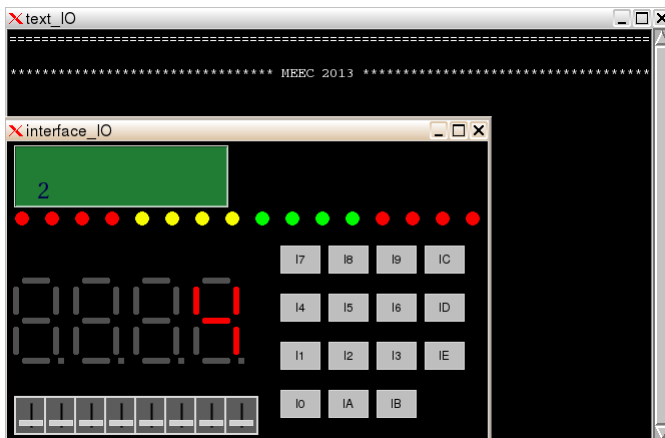
0000	JMP	0071
>>0002	XCH	R2, R3
>>0003	RTI	
>>0004	MOV	R4, ffff
0006	MOV	M[fff8], R4
>>0008	RTI	
>>0009	PUSH	R1
000a	MOV	R4, 0ff0
000c	MOV	M[fff8], R4
000e	MOV	R1, 0020
0010	MOV	M[fff6], R1
0012	MOV	R1, 0001
0014	MOV	M[fff7], R1
- I/O Devices:**
 - text_IO:** Displays "MEEC 2013".
 - interface_IO:** Shows a green display with the number "2", a row of 10 LEDs (4 yellow, 6 green), and a 4x4 grid of buttons labeled I7-IC, I4-ID, I1-IE, and I0-IB.
- Control Panel:**
 - Instrução
 - Corre
 - Reinicia
 - Continua
 - Refresca

P3 - PROCESSADOR

Interrupções

Execução do Programa:

- Ativação de todos os LEDs.



xp3sim

Ficheiro Definições Comandos Depuração Ver Ajuda

Instruções 56338
Ciclos de relógio 588445

Register	Value
R0	0000
R1	006f
R2	fff0
R3	fff1
R4	ffff
R5	0000
R6	0000
R7	0000
PC	0008
SP	fdfb

E 0 Z 0 C 0 N 0 O 0

Address	Instruction	Comment
8000	: 002a 002a 002a 0020 004d 0045 0045 0043	*** MEEC
8008	: 0020 0032 0030 0031 0033 0020 002a 002a	2013 **
8010	: 002a 0040 003d 003d 003d 003d 003d 003d	*@=====
8018	: 003d 003d 003d 003d 003d 003d 003d 003d	=====
8020	: 003d 003d 003d 003d 003d 003d 003d 003d	=====
8028	: 003d 003d 003d 003d 003d 003d 003d 003d	=====
8030	: 003d 003d 003d 003d 003d 003d 003d 003d	=====
8038	: 003d 003d 003d 003d 003d 003d 003d 003d	=====
8040	: 003d 003d 003d 003d 003d 003d 003d 003d	=====
8048	: 003d 003d 003d 003d 003d 003d 003d 003d	=====
8050	: 003d 003d 003d 003d 003d 003d 003d 003d	=====
8058	: 003d 003d 003d 003d 003d 003d 003d 003d	=====
fd00	: 0000 0000 0000 0000 0000 0000 0000 0000
fd08	: 0000 0000 0000 0000 0000 0000 0000 0000
fd10	: 0000 0000 0000 0000 0000 0000 0000 0000
fd18	: 0000 0000 0000 0000 0000 0000 0000 0000
fd20	: 0000 0000 0000 0000 0000 0000 0000 0000
fd28	: 0000 0000 0000 0000 0000 0000 0000 0000

```

>>0002 XCH R2, R3
>>0003 RTI
>>0004 MOV R4, ffff
0006 MOV M[fff8], R4
>>0008 RTI
>>0009 PUSH R1
000a MOV R4, 0ff0
000c MOV M[fff8], R4
000e MOV R1, 0020
0010 MOV M[fff6], R1
0012 MOV R1, 0001
0014 MOV M[fff7], R1
0016 POP R1
    
```

Instrução

Corre

Reinicia

Continua

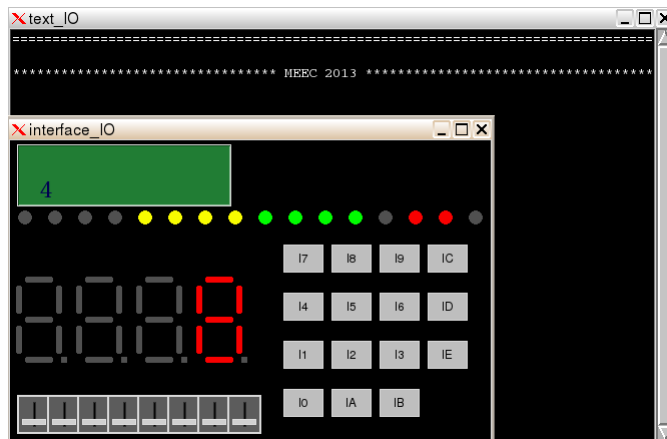
Refresca

P3 - PROCESSADOR

Interrupções

Execução do Programa:

- Entrada na rotina de atendimento à interrupção 0 para troca do Display de 7 seg.



p3sim

Ficheiro Definições Comandos Depuração Ver Ajuda

Instruções 88296
Ciclos de relógio 923197

R0	0000	8000	:	002a	002a	002a	0020	004d	0045	0045	0043	*** MEEC
R1	0257	8008	:	0020	0032	0030	0031	0033	0020	002a	002a	2013 **
R2	ffff0	8010	:	002a	0040	003d	003d	003d	003d	003d	003d	*@=====
R3	ffff1	8018	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R4	0fff6	8020	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R5	0000	8028	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R6	0000	8030	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R7	0000	8038	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
PC	0002	8040	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
SP	fdfb	8048	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
		8050	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
		8058	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
		fd00	:	0000	0000	0000	0000	0000	0000	0000	0000
		fd08	:	0000	0000	0000	0000	0000	0000	0000	0000
		fd10	:	0000	0000	0000	0000	0000	0000	0000	0000
		fd18	:	0000	0000	0000	0000	0000	0000	0000	0000
		fd20	:	0000	0000	0000	0000	0000	0000	0000	0000
		fd28	:	0000	0000	0000	0000	0000	0000	0000	0000

E 0 Z 0 C 0 N 0 0 0

0000	JMP	0071
>>0002	XCH	R2, R3
>>0003	RTI	
>>0004	MOV	R4, ffff
0006	MOV	M[fff8], R4
>>0008	RTI	
>>0009	PUSH	R1
000a	MOV	R4, 0fff
000c	MOV	M[fff8], R4
000e	MOV	R1, 0020
0010	MOV	M[fff6], R1
0012	MOV	R1, 0001
0014	MOV	M[fff7], R1

Instrução

Corre

Reinicia

Continua

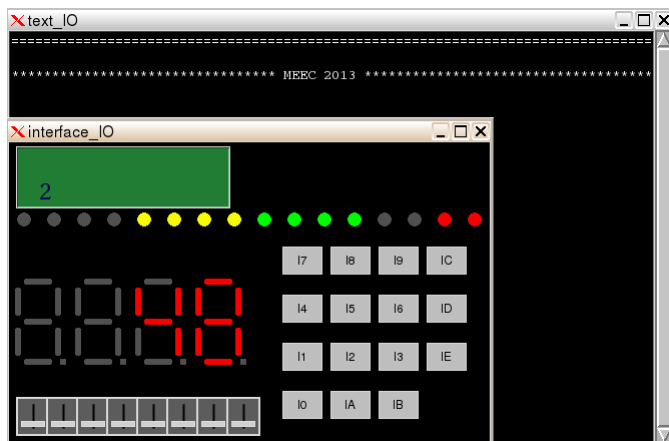
Refresca

P3 - PROCESSADOR

Interrupções

Execução do Programa:

- Escrita no 2º Display e FLAG E ativa fora da rotina de interrupção.



p3sim

Ficheiro Definições Comandos Depuração Ver

Instruções 124338
Ciclos de relógio 1300741

R0	0000	8000	:	002a	002a	002a	0020	004d	0045	0045	0043	*** MEEC
R1	026a	8008	:	0020	0032	0030	0031	0033	0020	002a	002a	2013 **
R2	ffff1	8010	:	002a	0040	003d	003d	003d	003d	003d	003d	*@=====
R3	ffff0	8018	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R4	0ff3	8020	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R5	0000	8028	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R6	0000	8030	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
R7	0000	8038	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
PC	0054	8040	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
SP	fdfd	8048	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
E	1	8050	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
Z	0	8058	:	003d	003d	003d	003d	003d	003d	003d	003d	=====
C	1	fd00	:	0000	0000	0000	0000	0000	0000	0000	0000
N	0	fd08	:	0000	0000	0000	0000	0000	0000	0000	0000
O	0	fd10	:	0000	0000	0000	0000	0000	0000	0000	0000
0	0	fd18	:	0000	0000	0000	0000	0000	0000	0000	0000
0	0	fd20	:	0000	0000	0000	0000	0000	0000	0000	0000
0	0	fd28	:	0000	0000	0000	0000	0000	0000	0000	0000

004f	RETN	2
0050	PUSH	R1
0051	MOV	R1, 0300
0053	DEC	R1
0054	BR.NZ	-2
0055	POP	R1
0056	RET	
0057	MOV	R1, 0002
0059	MOV	M[fe00], R1
005b	MOV	R1, 0004
005d	MOV	M[fe01], R1
005f	MOV	R1, 0009
0061	MOV	M[fe0f], R1

Instrução

Corre

Reinicia

Continua

Refresca

• Bibliografia

[1] M. Morris Mano, Charles R. Kime, “Logic and Computer Design Fundamentals”, Prentice-Hall International, Inc. (Capítulos 9, 10 e 11)

[2] N. Horta, “Arquitecturas de Computadores”, Aulas Teóricas, 2013.

[3] G.Arroz, J.C.Monteiro, A.Oliveira, “Manual do Simulador do P3”, IST, 2007

[4] G.Arroz, J.C.Monteiro, A.Oliveira, “Introdução aos Sistemas Digitais e Microprocessadores”, IST Press, 2007.