

MC404

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

2010

Prof. Paulo Cesar Centoducatte
Prof. Mario Lúcio Côrtes
Prof. Ricardo Pannain

MC404 – 2s2010

Organização Básica de Computadores e Linguagem de Montagem

3 - 1

MC404

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

"Diretivas de Montagem"

MC404 – 2s2010

Organização Básica de Computadores e Linguagem de Montagem

3 - 2

Diretivas de Montagem Sumário

- Diretivas do Montador
 - Segmento
 - Reserva de bytes e words
 - Definição de constantes
 - Definição de símbolos para registradores

MC404 – 2s2010

Organização Básica de Computadores e Linguagem de Montagem

3 - 3

Arquivo Fonte

- [Label:] diretiva [operandos] [comentários]
 - [Label:] instrução [operandos] [comentários]
 - Comentários
 - Linha em branco
-
- Comentários:
; [texto]

MC404 – 2s2010

Organização Básica de Computadores e Linguagem de Montagem

3 - 4

Diretivas do Montador

Directive	Description
BYTE	Reserve byte to a variable
CSEG	Code Segment
DB	Define constant byte(s)
DEF	Define a symbolic name on a register
DEVICE	Define which device to assemble for
DSEG	Data Segment
DW	Define constant word(s)
ENDMACRO	End macro
EQU	Set a symbol equal to an expression
ESEG	EEPROM Segment
EXIT	Exit from file
INCLUDE	Read source from another file
LIST	Turn listfile generation on
LISTMAC	Turn macro expansion on
MACRO	Begin macro
NOLIST	Turn listfile generation off
ORG	Set program origin
SET	Set a symbol to an expression

MC404 – 2s2010

Organização Básica de Computadores e Linguagem de Montagem

3 - 5

Diretivas do Montador

```
label: .EQU var1=100      ; Set var1 to 100 (Directive)
      .EQU var2=200      ; Set var2 to 200
test:  rjmp test          ; Infinite loop (Instruction)
                        ; Pure comment line
                        ; Another comment line
```

MC404 – 2s2010

Organização Básica de Computadores e Linguagem de Montagem

3 - 6

Diretivas do Montador Segmentos

- **.CSEG** ; Segmento de Memória na Flash
- **.DSEG** ; Segmento de Memória na SRAM
- **.ESEG** ; Segmento de Memória na EEPROM

Reserva de Byte(s) na SRAM

```
.DSEG
var1: .BYTE 1 ; reserve 1 byte to var1
table: .BYTE tab_size ; reserve tab_size bytes
.CSEG
ldi r30,low(var1) ; Load Z register low
ldi r31,high(var1) ; Load Z register high
ld r1,Z ; Load VAR1 into register 1
```

Reserva Byte(s) com inicialização

```
.CSEG
const: .DB 0, 255, 0b01010101, -128, 0xaa
.ESEG
eeconst: .DB 0xff
```

Nome Simbólicos para Registradores

Syntax:

```
.DEF Symbol=Register
```

Example:

```
.DEF temp=R16
.DEF ior=R0
.CSEG
ldi temp,0xf0 ; Load 0xf0 into temp register
in ior,0x3f ; Read SREG into ior register
eor temp,ior ; Exclusive or temp and ior
```

Reserva Word(s) com Inicialização

Syntax:

```
LABEL: .DW expressionlist
```

Example:

```
.CSEG
varlist: .DW 0,0xffff,0b100110001010101,-32768,65535
.ESEG
eevar: .DW 0xffff
```

Syntax:

```
.EQU label = expression
```

Example:

```
.EQU io_offset = 0x23
.EQU porta = io_offset + 2
.CSEG
clr r2 ; Start code segment ; Clear register 2
out porta,r2 ; Write to Port A
```

Syntax:

```
.ORG expression
```

Example:

```
.DSEG                ; Start data segment
.ORG 0x67             ; Set SRAM address to hex 67
    variable:.BYTE 1  ; Reserve a byte at SRAM
                      ; adr.67H
.ESEG                ; Start EEPROM Segment
.ORG 0x20             ; Set EEPROM location
                      ; counter
    eevar: .DW 0xfeff ; Initialize one word
.CSEG
.ORG 0x10             ; Set Program Counter to hex
                      ; 10
    mov     r0,r1     ; Do something
```

Expressões

The following functions are defined:

- LOW(expression) returns the low byte of an expression
- HIGH(expression) returns the second byte of an expression
- BYTE2(expression) is the same function as HIGH
- BYTE3(expression) returns the third byte of an expression
- BYTE4(expression) returns the fourth byte of an expression
- LWRD(expression) returns bits 0-15 of an expression
- HWRD(expression) returns bits 16-31 of an expression
- PAGE(expression) returns bits 16-21 of an expression
- EXP2(expression) returns $2^{\text{expression}}$
- LOG2(expression) returns the integer part of $\log_2(\text{expression})$