

MC404

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

2009

Prof. Paulo Cesar Centoducatte

ducatte@ic.unicamp.br

www.ic.unicamp.br/~ducatte

MC404

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

**“Macros,
Montagem Condicional de Código
e Uso da EEPROM”**

Macros

Sumário

- **Macros**
- **Montagem Condicional de Código**
- **Uso da EEPROM**

Macros

**; Troca o conteúdo de dois registradores sem usar
; variável auxiliar!:**

.MACRO SWAP ;(Rd, Rs)

eor @0, @1

eor @1, @0

eor @0, @1

.ENDMACRO

Macro

;Soma uma constante de 8 bits a um registrador

```
.MACRO ADDI ;(Rd, k)  
    subi    @0, -(@1)  
.ENDMACRO
```

**; Soma uma constante de 16 bits a um par de
;registradores (X, Y ou Z)**

```
.MACRO ADDIW ;(RdL:RdH, k)  
    subi    @0L, LOW(-@1)  
    sbci    @0H, HIGH(-@1)  
.ENDMACRO
```

Macro - Montagem Condicional

; I/O

.macro input

.if @1 < 0x40

in @0, @1

.else

lds @0, @1

.endif

.endmacro

Macro - Montagem Condicional

; I/O

```
.macro output  
  .if @0 < 0x40  
    out @0, @1  
  .else  
    sts @0, @1  
  .endif  
.endmacro
```

Macro - Montagem Condicional

; Branch if Bit in I/O-Register is Set

```
.macro bbis ;port,bit,target
```

```
.if @0 < 0x20
```

```
    sbic    @0, @1
```

```
    rjmp    @2
```

```
.elif @0 < 0x40
```

```
    in      zl, @0
```

```
    sbrc    zl, @1
```

```
    rjmp    @2
```

```
.else
```

```
    lds     zl, @0
```

```
    sbrc    zl, @1
```

```
    rjmp    @2
```

```
.endif
```

```
.endmacro
```


Macro e Montagem Condicional

Branch if Bit in I/O-Register is Cleared

```
.macro bbic ;port,bit,target
    .if @0 < 0x20
        sbis @0, @1
        rjmp @2
    .elif @0 < 0x40
        in    zl, @0
        sbrs zl, @1
        rjmp @2
    .else
        lds  zl, @0
        sbrs zl, @1
        rjmp @2
    .endif
.endmacro
```

Montagem Condicional

```
.EQU clock=4000000
```

```
.IF clock>4000000
```

```
    .EQU divider=4
```

```
.ELSE
```

```
    .EQU divider=2
```

```
.ENDIF
```

Mais exemplos de uso de IF e ENDIF

Macros Aninhadas

```
.macro mult8 ; macro para multiplica dois números  
            ; de 8 bits sem sinal  
            ; Parâmetros de entrada:  
            ; @0 e @1: dois registradores quaisquer  
            ; (números para multiplcar)  
            ; destroi: r0, r1, @2  
  
            mul @0,@1  
.endmacro
```

Macros Aninhadas

```
.macro callmult8 ; macro para multiplicar dois números  
                ; de 8 bits sem sinal
```

```
; Parâmetros de entrada:
```

```
; @0 e @1: dois registradores quaisquer
```

```
; (números para multiplicar)
```

```
; @2 um dos pares X, Y ou Z: endereço na RAM
```

```
; para colocar o produto no formato little endian
```

```
; destroi: r0, r1, @2
```

```
    mult8 @0,@1 ; invoca a macro mult8 repassando os  
                ; parametros @0 e @1
```

```
    st @2+,r0
```

```
    st @2, r1
```

```
.endmacro
```

Uso de Macro

```
RESET:                                ; início do programa
    ldi r16,low(RAMEND)                ; fim da RAM: definido em
                                        ; m88def.inc
    out SPL,r16                        ; inicializa Stack Pointer
    ldi r16,high(RAMEND)
    out SPH, r16
    ldi yh, high(SRAM_START)           ; Área de RAM onde será
                                        ; colocado o resultado
    ldi yl, low(SRAM_START)
    ldi r16,2
    ldi r17,5
    callmult8 r16,r17, y                ; Expande as duas macros

    rjmp PC
```

Uso da EEPROM

- **EEPROM Data Memory**
 - **The ATmega48/88/168 contains 256/512/512 bytes of data EEPROM memory.**
 - **It is organized as a separate data space, in which single bytes can be read and written.**
 - **The EEPROM has an endurance of at least 100,000 write/erase cycles.**
 - **The access between the EEPROM and the CPU is described in the following, specifying the:**
 - **EEPROM Address Registers,**
 - **EEPROM Data Register, and**
 - **EEPROM Control Register.**

EEPROM

The EEPROM Address Register – EEARH and EEARL

Bit	15	14	13	12	11	10	9	8	
	-	-	-	-	-	-	-	EEAR8	EEARH
	EEAR7	EEAR6	EEAR5	EEAR4	EEAR3	EEAR2	EEAR1	EEAR0	EEARL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	X	
	X	X	X	X	X	X	X	X	

- **Bits 15..9 – Res: Reserved Bits**
- **Bits 8..0 – EEAR8..0: EEPROM Address**

EEPROM

The EEPROM Data Register – EEDR

Bit	7	6	5	4	3	2	1	0	
	MSB							LSB	EEDR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7..0 – EEDR7.0: EEPROM Data**

EEPROM

The EEPROM Control Register – EECR

Bit	7	6	5	4	3	2	1	0	
	-	-	EPM1	EPM0	EERIE	EEMPE	EEPE	EERE	EECR
Read/Write	R	R	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	X	X	0	0	X	0	

- **Bits 7..6 – Res: Reserved Bits**

These bits are reserved bits in the ATmega48/88/168 and will always read as zero.

- **Bits 5, 4 – EPM1 and EPM0: EEPROM Programming Mode Bits**

EEPROM

Bits 5, 4 – EEPM1 and EEPM0: EEPROM Programming Mode Bits

Table 5-1. EEPROM Mode Bits

EEP1	EEP0	Programming Time	Operation
0	0	3.4 ms	Erase and Write in one operation (Atomic Operation)
0	1	1.8 ms	Erase Only
1	0	1.8 ms	Write Only
1	1	-	Reserved for future use

EEPROM

- Bit 3 – EERIE: EEPROM Ready Interrupt Enable

Quando igual a 1, habilita a interrupção READY da EEPROM se o bit I do SREG estiver setado. Quando zero, desabilita a interrupção. A interrupção READY da EEPROM gera uma interrupção constante quando EEPE estiver zerado. A interrupção não será gerada durante a escrita na EEPROM ou SPM

- Bit 2 – EEMPE: EEPROM Master Write Enable

• Quando EEMPE é igual a 1, setando EEPE o dado será escrito na EEPROM dentro de 4 ciclos, no endereço selecionado. Se for zero, , setando EEPE, não termos nenhuma ação. EEMPE é setado por software e zerado por hardware, após 4 ciclos

EEPROM

- Bit 1 – EEPE: EEPROM Write Enable
É um sinal de strobe de escrita para a EEPROM. Quando o endereço e o dado estão corretamente inicializados, o EEPE deve ser setado para que seja feita a escrita na EEPROM. O bit EEMPE deve ser setado antes de EEPE, senão a escrita não será feita.
Procedimento pra escrita na EEPROM:
 1. Esperar até EEPE ser zero
 2. Esperar até SELPRGEN em SPMCSR ser zero
 3. Escrever o endereço em EEAR (opcional)
 4. Escrever o novo dado em EEDR (opcional)
 5. Escrever um no EMPE enquanto escreve zero em EEPE no EECR
 6. Em quatro ciclos após setar EEMPE, escrever um em EEPE

Uso da EEPROM

e2prom-rotinas.asm