MC404

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

2010

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

MC404

ORGANIZAÇÃO BÁSICA DE COMPUTADORES E LINGUAGEM DE MONTAGEM

"Passagens de Parâmetros para Subrotinas"

Passagens de Parâmetros
Sumário

- Por registradores
- Pela pilha
- Pela memória

Passagens de Parâmetros

- **Por registradores**

  É o modo mais simples. Para isto devemos carregar os valores, que serão utilizados nas rotinas, nos registradores. O número de parâmetros é restrito ao número de registradores disponíveis. Pelas características de acesso aos registradores, tende a tornar a execução mais rápida

Passagens de Parâmetros por registradores

```
        .nolist
        .include "m88def.inc"
        .list .def r =r16          ; usado na inicialização dos apontadores da pilha e inicio da RAM
start: rjmp RESET                  ; salta sobre eventuais vetores de interrupção ; se  tivermos rotinas de   interrupção os
                                     vetores (saltos para as rotinas) estariam aqui ; vetor 1 ; etc
mult8:              ; subrotina para multiplica dois números de 8 bits sem sinal. Parâmetros de entrada:  r16  e r17 :
                      números para multiplcar ; par Y: endereço na RAM para colocar o resultado no  formato little
                       endian, destroi: r0, r1, Y
        mul r16,r17
        st y+,r0
        st y, r1
        ret
RESET: ; início do programa
        ldi r,low(RAMEND)          ; fim da RAM: definido em m88def.inc
        out SPL,r                  ; inicializa Stack Pointer: redundante no caso do ATmega88
        ldi r,high(RAMEND)         ; pois ao dar partida a CPU inicializa o SPL
        out SPH, r                 ; e o SPH com o end  da ultima posicao da RAM
        ldi yh, high(SRAM_START)   ; no inicio da RAM será colocado o resultado
        ldi yl, low(SRAM_START)
        ldi r16,2                  ; vamos multiplicar 2
        ldi r17,5                  ; por 5, o resultado vai para (r0,r1) (low,high)
        rcall mult8                ; resultado no formato little endian (low, high)= 0a00
        rjmp PC                    ; laço infinito
```

Passagens de Parâmetros

- **Pela memória**

Para isto devemos atribuir às posições de memória, os valores a serem utilizados nas rotinas. Pelas características de acesso à memória, tende a tornar a execução mais lenta.

Passagens de Parâmetros pela memória

```
        .nolist
        .include "m88def.inc"
        .list .def r =r16           ; usado na inicialização dos apontadores da pilha e inicio da RAM
start: rjmp RESET
mult8:
        ld  r16, y+
        ld  r17, y+
        mul r16,r17
        st y+,r0
        st y, r1
        ret
RESET: ; início do programa
        ldi r,low(RAMEND)                   ; fim da RAM: definido em m88def.inc
        out SPL,r                           ; inicializa Stack Pointer: redundante no caso do ATmega88
        ldi r,high(RAMEND)     ; pois ao dar partida a CPU inicializa o SPL
        out SPH, r                           ; e o SPH com o end da ultima posicao da RAM
         ldi yh, high(SRAM_START)           ; no inicio da RAM será colocado o resultado
         ldi yl, low(SRAM_START)
        push r28                            ;salvo apontador para  o incio da RAM
        push r29
        ldi r18,2
        ldi r19,5
        st y+,r18                           ; vamos multiplicar 2
        st y, r19                           ; por 5, o resultado vai para (r0,r1) (low,high)
        pop r29
        pop r28
        rcall mult8                         ; resultado no formato little endian (low, high)= 0a00
         rjmp PC                            ; laço infinito
```

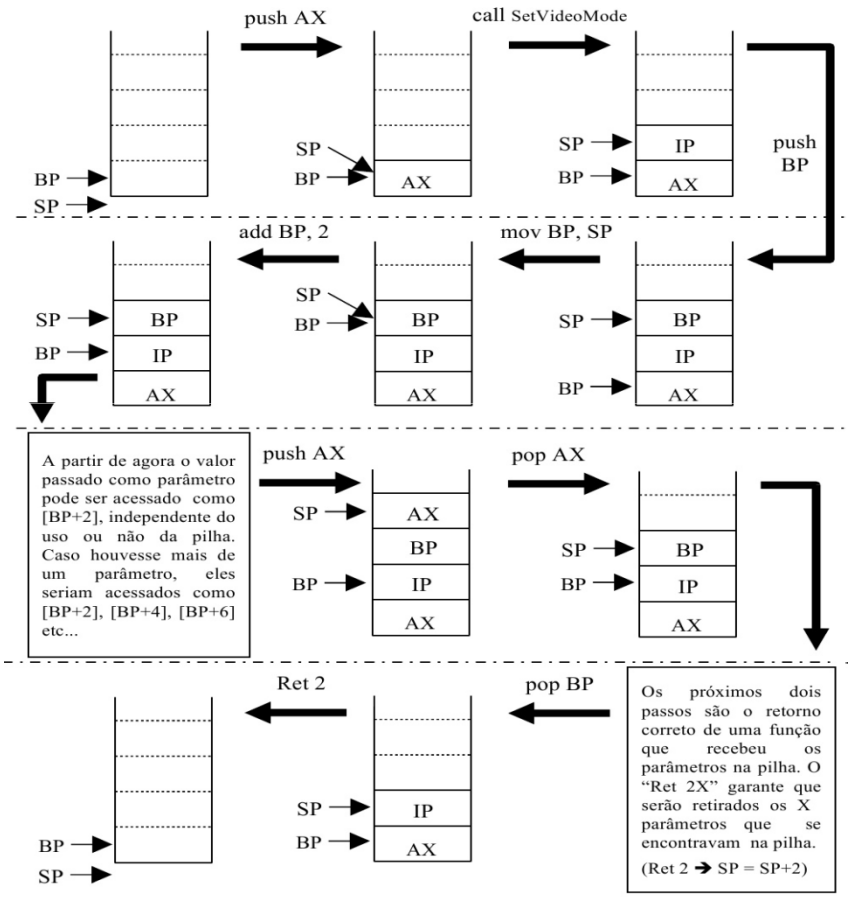
Passagens de Parâmetros

- **Passagem pela pilha**

  É o modo mais flexível, mas que exige alguns cuidados, pois o endereço de retorno também estará armazenado na pilha.

Passagem pela pilha

Passagem de Parâmetro Por Pilha:



Passagens de Parâmetros pela pilha

```
        .nolist
        .include "m88def.inc"
        .list .def r =r16       ; usado na inicialização dos apontadores da pilha e inicio da RAM
start: rjmp RESET               ; salta sobre eventuais vetores de interrupção ; se  tivermos rotinas de   interrupção os
                                  vetores (saltos para as rotinas) estariam aqui ; vetor 1 ; etc
mult8:
        in r26,SPL
        in r27, SPH
        subi r26,-3
        ld r17, x+
        ld r16, x
        mul r16,r17
        st y+,r0
        st y, r1
        ret
RESET: ; início do programa
        ldi r,low(RAMEND)               ; fim da RAM: definido em m88def.inc
        out SPL,r                       ; inicializa Stack Pointer: redundante no caso do ATmega88
        ldi r,high(RAMEND)              ; pois ao dar partida a CPU inicializa o SPL
        out SPH, r                       ; e o SPH com o end   da ultima posicao da RAM
        ldi yh, high(SRAM_START)        ; no inicio da RAM será colocado o resultado
        ldi yl, low(SRAM_START)
        ldi r16,2                       ; vamos multiplicar 2
        ldi r17,5                       ; por 5, o resultado vai para (r0,r1) (low,high)
        push r16
        push r17
        rcall mult8                     ; resultado no formato little endian (low, high)= 0a00
        rjmp PC                         ; laço infinito
```