Ensaios Sugeridos (programas) para o Computador de 8-bits

Autor: Eng. Wagner Rambo

Data: Novembro de 2017

www.wrkits.com.br

Teste da instrução LDA Assembly

```
.org    0h      ;origem no endereço 0h de memória
lda     9h      ;acc = (9h) *entre parênteses, indica conteúdo da
                ;posição de memória
out             ;move conteúdo de acc para o registrador de saída
hlt             ;para o processamento
.end            ;final do programa
```

Teste da instrução LDA Código Objeto

Address	Program
0000	00011001
0001	11100000
0010	11110000
.	Data
1001	00101010

Resultado esperado: “042” no display decimal

Teste da instrução LDI Assembly

```
.org    0h      ;origem no endereço 0h de memória
ldi     9h      ;acc = 9h
out             ;move conteúdo de acc para o registrador de saída
hlt             ;para o processamento
.end            ;final do programa
```

Teste da instrução LDI Código Objeto

Address	Program
0000	00101001
0001	11100000
0010	11110000

Resultado esperado: “009” no display decimal

Resolvendo a operação “35+127” Assembly

```
.org    0h      ;origem no endereço 0h de memória
lda     Ch      ;acc = (Ch)
add     Dh      ;acc = acc + (Dh)
out             ;move conteúdo de acc para o registrador de saída
hlt             ;para o processamento
.end            ;final do programa
```

Resolvendo a operação “35+127” Código Objeto

Address	Program
0000	00011100
0001	01001101
0010	11100000
0011	11110000
.	Data
1100	00100011
1101	01111111

Resultado esperado: “162” no display decimal

Resolvendo a operação “4+129” Assembly

```
.org    0h      ;origem no endereço 0h de memória
ldi     4h      ;acc = 4h
add     Dh      ;acc = acc + (Dh)
out             ;move conteúdo de acc para o registrador de saída
hlt             ;para o processamento
.end            ;final do programa
```

Resolvendo a operação “4+129” Código Objeto

Address	Program
0000	00100100
0001	01001101
0010	11100000
0011	11110000
.	Data
1101	10000001

Resultado esperado: “133” no display decimal

Resolvendo a operação “224-63” Assembly

```
.org    0h      ;origem no endereço 0h de memória
lda     Dh      ;acc = (Dh)
sub     Eh      ;acc = acc - (Eh)
out             ;move conteúdo de acc para o registrador de saída
hlt             ;para o processamento
.end            ;final do programa
```

Resolvendo a operação “224-63” Código Objeto

Address	Program
0000	00011101
0001	01011110
0010	11100000
0011	11110000
.	Data
1101	11100000
1110	00111111

Resultado esperado: “161” no display decimal

Resolvendo a operação “10-13+58” Assembly

```
.org    0h      ;origem no endereço 0h de memória
ldi     Ah      ;acc = 10d
sub     Fh      ;acc = acc - (Fh)      (Fh) = 13d
add     Eh      ;acc = acc + (Eh)      (Eh) = 58d
out             ;move conteúdo de acc para o registrador de saída
hlt             ;para o processamento
.end            ;final do programa
```

Resolvendo a operação “10-13+58” Código Objeto

Address	Program
0000	00101010
0001	01011111
0010	01001110
0011	11100000
0100	11110000
.	Data
1110	00111010
1111	00001101

Resultado esperado: “055” no display decimal

Resolvendo a operação AND entre dois números Assembly

```
.org    0h      ;origem no endereço 0h de memória
lda     Bh      ;acc = (Bh)
and     Ch      ;acc = acc AND (Ch)
out             ;move conteúdo de acc para o registrador de saída
hlt             ;para o processamento
.end            ;final do programa
```

Resolvendo a operação AND entre dois números Código Objeto

Address	Program
0000	00011011
0001	01101100
0010	11100000
0011	11110000
.	Data
1011	11000011
1100	10101011

Resultado esperado: “10000011” no registrador de saída

Resolvendo a operação OR entre dois números Assembly

```
.org    0h      ;origem no endereço 0h de memória
lda     Dh      ;acc = (Dh)
orl     Eh      ;acc = acc OR (Eh)
out             ;move conteúdo de acc para o registrador de saída
hlt             ;para o processamento
.end            ;final do programa
```

Resolvendo a operação OR entre dois números Código Objeto

Address	Program
0000	00011101
0001	01111110
0010	11100000
0011	11110000
.	Data
1101	11000011
1110	10101011

Resultado esperado: “11101011” no registrador de saída

Resolvendo a operação XOR entre dois números Assembly

```
.org    0h      ;origem no endereço 0h de memória
lda     Eh      ;acc = (Eh)
xor     Fh      ;acc = acc XOR (Fh)
out             ;move conteúdo de acc para o registrador de saída
hlt             ;para o processamento
.end            ;final do programa
```

Resolvendo a operação XOR entre dois números Código Objeto

Address	Program
0000	00011110
0001	10001111
0010	11100000
0011	11110000
.	Data
1110	11000011
1111	10101011

Resultado esperado: “01101000” no registrador de saída

Teste da instrução NOT Assembly

```
.org    0h      ;origem no endereço 0h de memória
lda     7h      ;acc = (7h)
not             ;acc = NOT acc
out             ;move conteúdo de acc para o registrador de saída
hlt             ;para o processamento
.end            ;final do programa
```

Teste da instrução NOT Código Objeto

Address	Program
0000	00010111
0001	10010000
0010	11100000
0011	11110000
.	Data
0111	11000011

Resultado esperado: “00111100” no registrador de saída

Calcular complemento de dois do número “14” Assembly

```
.org    0h      ;origem no endereço 0h de memória
ldi     1h      ;carrega literal 1 em acc
sta     Fh      ;armazena acc no endereço Fh
ldi     Eh      ;carrega literal 14d em acc
not             ;resolve o complemento de um de acc
add     Fh      ;soma 1 em acc’ (complemento de dois)
out             ;envia resultado para o registrador de saída
hlt             ;para o processamento
.end            ;final do programa
```

Calcular complemento de dois do número “14” Código Objeto

Address	Program
0000	00100001
0001	00111111
0010	00101110
0011	10010000
0100	01001111
0101	11100000
0110	11110000

Calcular complemento de dois do número “14” Saída de Dados

Output Register: 11110010b

Decimal Display: 242d (unsigned)

Decimal Display: -014d (signed)

Teste de Loop (incremento de 2 em 2) Assembly

```
.org    0h      ;origem no endereço 0h de memória
ldi     2h      ;carrega literal 2 em acc
sta     Fh      ;armazena acc no endereço Fh
ldi     0h      ;carrega literal 0d em acc
out             ; envia resultado para o registrador de saída
add     Fh      ;soma (Fh) em acc
jmp     3h      ;desvio incondicional para endereço 3
.end            ;final do programa
```

Teste de Loop (incremento de 2 em 2) Código Objeto

Address	Program
0000	00100010
0001	00111111
0010	00100000
0011	11100000
0100	01001111
0101	10100011

Teste de Loop (incremento de 2 em 2) Saída de Dados

Output Register: Incremento de 00000010b em 00000010b

Decimal Display: Incremento de 2 em 2

Dica: habilite a chave “signed” do display e interprete o efeito que
ocorre

Teste de Loop (decremento de 4 em 4) Assembly

```
.org    0h      ;origem no endereço 0h de memória
ldi     4h      ;carrega literal 4 em acc
sta     Fh      ;armazena acc no endereço Fh
ldi     0h      ;carrega literal 0d em acc
out             ; envia resultado para o registrador de saída
sub     Fh      ;subtrai (Fh) em acc
jmp     3h      ;desvio incondicional para endereço 3
.end            ;final do programa
```

Teste de Loop (decremento de 4 em 4) Código Objeto

Address	Program
0000	00100100
0001	00111111
0010	00100000
0011	11100000
0100	01011111
0101	10100011

Teste de Loop (decremento de 4 em 4) Saída de Dados

Output Register: Decremento de 00000100b em 00000100b

Decimal Display: Decremento de 4 em 4

Dica: habilite a chave “signed” do display e interprete o efeito que
ocorre

Deslocamento para Esquerda Assembly

```
.org    0h      ;origem no endereço 0h de memória
ldi     1h      ;carrega literal 1 em acc
out             ; envia resultado para o registrador de saída
sta     Eh      ;armazena acc no endereço Eh
add     Eh      ; soma (Fh) em acc
jmp     1h      ;desvio incondicional para endereço 1
.end            ;final do programa
```

Deslocamento para Esquerda Código Objeto

Address	Program
0000	00100001
0001	11100000
0010	00111110
0011	01001110
0100	10100001

Deslocamento para Esquerda Saída de Dados

Output Register: 00000001b .. 00000010b .. 00000100b ...

Decimal Display: Efeito de multiplicação por 2 (1.. 2 .. 4 .. 8 .. 16 ...)

Exercício proposto: após atingir o valor “10000000b”, o registrador de saída apresentará o valor “00000000b” e permanecerá assim. Analise o porquê disto estar ocorrendo.