

## Instruções assembly da família Intel x86-64 - Sintaxe AT&T

### Registadores

64 bits	32 bits	16 bits	8 bits		
%rax	%eax	%ax	%ah	%al	valor de retorno
%rbx	%ebx	%bx	%bh	%bl	callee-saved
%rcx	%ecx	%cx	%ch	%cl	4º argumento
%rdx	%edx	%dx	%dh	%dl	3º argumento
%rsi	%esi	%si		%sil	2º argumento
%rdi	%edi	%di		%dil	1º argumento
%r8	%r8d	%r8w		%r8b	5º argumento
%r9	%r9d	%r9w		%r9b	6º argumento
%r10	%r10d	%r10w		%r10b	
%r11	%r11d	%r11w		%r11b	
%r12	%r12d	%r12w		%r12b	callee-saved
%r13	%r13d	%r13w		%r13b	callee-saved
%r14	%r14d	%r14w		%r14b	callee-saved
%r15	%r15d	%r15w		%r15b	callee-saved
%rbp					frame pointer
%rsp					stack pointer

### Instruções

Todas as instruções contendo mais de um operando, guardam seu resultado em op2.

Para especificar o tamanho do(s) operando(s), acrescentar sufixo na instrução: **B,W,L** ou **Q**.

MOV	op1	op2	copia para op2 o valor de op1
MOVB	op1	op2	copia para op2 (8 bits) o valor de op1 (constante de 8 bits)
MOVW	op1	op2	copia para op2 (16 bits) o valor de op1 (constante de 16 bits)
MOVQ	op1	op2	copia para op2 (64 bits) o valor de op1 (constante de 64 bits)
MOVBABSQ	op1	op2	copia para op2 (64 bits) o valor de op1 (constante de 64 bits)
ADD	op1	op2	guarda em op2 o resultado de op2 + op1
SUB	op1	op2	guarda em op2 o resultado de op2 - op1
IMUL	op1	op2	guarda em op2 resultado de op1 * op2
INC	op1		incrementa o valor de op1
DEC	op1		decrementa o valor de op1
NEG	op1		complemento a 2 do valor de op1
AND	op1	op2	guarda em op2 o resultado de op1 & op2
OR	op1	op2	guarda em op2 o resultado de op1   op2
XOR	op1	op2	guarda em op2 o resultado de op1 ^ op2
NOT	op1		guarda em op1 o resultado de ~op1
CMP	op1	op2	compara op2 com op1 (atualiza flags)
TEST	op1	op2	atualiza flags conforme resultado de op2 AND op1 (não altera op2)
SHL	constante	op 1	desloca op1 o número de bits indicado para a esquerda
SHR	constante	op 1	desloca op1 o número de bits indicado para a direita (shift lógico)
SAR	constante	op 1	desloca o número de bits indicado para a direita (shift aritmético)
LEAQ	op1	op2	guarda em op2 o ENDEREÇO de op1 ("load effective address")

### Extensão considerando sinal

MOVSB[W][L][Q]	op1	op2	guarda em op2 (w,l,q) o valor estendido de op1 (8 bits)
MOVSW[L][Q]	op1	op2	guarda em op2 (l,q) o valor estendido de op1 (16 bits)
MOVSLQ	op1	op2	guarda em op2 (64 bits) o valor estendido de op1 (32 bits)

### Extensão sem sinal

MOVZB[W][L][Q]	op1	op2	guarda em op2 (w,l,q) o valor estendido de op1 (8 bits)
MOVZW[L][Q]	op1	op2	guarda em op2 (l,q) o valor estendido de op1 (16 bits)

### Desvio condicional

JE/JZ	label	transfere para label caso o flag ZF (resultado zero) esteja setado
JNE/JNZ	label	transfere para label caso o flag ZF não esteja setado

### Desvio condicional para operações sem sinal

JA/JNBE	label	transfere se superior (“jump if above”)
JAE/JNB	label	transfere se superior ou igual (“jump if above or equal”)
JB/JNAE	label	transfere se inferior (“jump if below”)
JBE/JNA	label	transfere se inferior ou igual (“jump if below or equal”)

### Desvio condicional considerando sinal

JG/JNLE	label	transfere se superior (“jump if greater”)
JGE/JNL	label	transfere se superior ou igual (“jump if greater or equal”)
JL/JNGE	label	transfere se inferior (“jump if less”)
JLE/JNG	label	transfere se inferior ou igual (“jump if less or equal”)

### Desvio incondicional

JMP	label	transfere para label
CALL	label	transfere para label, empilhando endereço de retorno
RET		transfere para endereço retirado do topo da pilha
INT	valor	transfere para o tratador da interrupção indicada

## Instruções de Ponto Flutuante

<b>Registadores:</b>	%xmm0 a %xmm15
<b>Argumentos:</b>	%xmm0 a %xmm7
<b>Valor de Retorno:</b>	%xmm0

### Movimentação e Conversão de Dados

MOVS[S][D]	op1	op2	copia para op2 o valor de op1
CVTSS2SD	op1	op2	converte float em op1 para double em op2 (reg)
CVTSD2SS	op1	op2	converte double em op1 para float em op2 (reg)
CVTSI2SS	op1	op2	converte inteiro em op1 para float em op2 (reg)
CVTSI2SD	op1	op2	converte inteiro em op1 para double em op2 (reg)
CVTTSS2SI	op1	op2	converte float em op1 para inteiro em op2 (reg)
CVTTSD2SI	op1	op2	converte double em op1 para inteiro em op2 (reg)
CVTTSS2SIQ	op1	op2	converte float em op1 para long em op2 (reg)
CVTTSD2SIQ	op1	op2	converte double em op1 para long em op2 (reg)

### Operações Aritméticas

ADDS[S][D]	op1	op2	guarda em op2 (reg) o resultado de op2 + op1
SUBS[S][D]	op1	op2	guarda em op2 (reg) o resultado de op2 - op1
MULS[S][D]	op1	op2	guarda em op2 (reg) o resultado de op2 * op1
DIVS[S][D]	op1	op2	guarda em op2 (reg) o resultado de op2 / op1

### Comparação

UCOMIS[S][D]	op1	op2	compara op2 com op1 (testar resultado com condições de comparação sem sinal: a, ae, b, be)
--------------	-----	-----	---