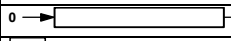
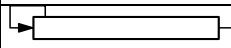
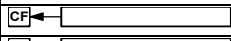
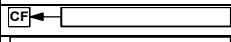
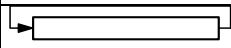
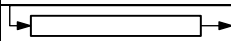
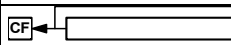
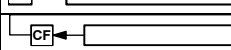
Transferência de dados	
MOV destino,fonte	**destino ← fonte**
PUSH fonte	coloca **fonte** na pilha
POP destino	retira da pilha para **destino**
XCHG oper1,oper2	Troca o conteúdo de **oper1** com **oper2**
BSWAP r32	Inverte a ordem dos bytes do registrador **r32**
Transferência sobre endereços	
LEA r16/r32 , operando	Carrega o EA do **operando** para o registro **r16/r32**
LDS r16/r32 , operando	Carrega o EA do **operando** para **r16/r32** e o segmento para o **DS**
LES r16/r32 , operando	Carrega o EA do **operando** para **r16/r32** e o segmento para o **ES**
LFS r16/r32 , operando	Carrega o EA do **operando** para **r16/r32** e o segmento para o **FS**
LGS r16/r32 , operando	Carrega o EA do **operando** para **r16/r32** e o segmento para o **GS**
XLAT	AX ← [BX+AL]
Entrada e Saída	
IN acumulador, porta	Leitura de um byte (acumulador=AL), word (acumulador=AX) ou dword (acumulador=EAX) na **porta**
OUT porta, acumulador	Escreve na **porta** um byte (acumulador=AL), word (acumulador=AX) ou dword (acumulador=EAX)
Instruções aritméticas	
ADD destino, fonte	destino = destino + fonte
ADC destino, fonte	destino = destino + fonte + **carry**
SUB destino, fonte	destino = destino – fonte
SBB destino, fonte	destino = destino – fonte - borrow
CMP destino, fonte	Seta os códigos de condição, de acordo com a diferença “destino – fonte”
INC destino	destino = destino + 1
DEC destino	destino = destino – 1
NEG destino	destino = 0 - destino
MUL fonte	Multiplicação **sem** sinal de bytes (AX=AL*fonte), word (DX:AX=AX*fonte) e dword (EDX:EAX=EAX*fonte)
IMUL fonte	Multiplicação **com** sinal de bytes (AX=AL*fonte), word (DX:AX=AX*fonte) e dword (EDX:EAX=EAX*fonte)
DIV fonte	Divisão **sem** sinal de bytes (AL=AL / fonte; AH=resto), word (AX=DX:AX / fonte, DX=resto) e dword (EAX=EDX:EAX / fonte, EDX=resto)
IDIV fonte	Divisão **com** sinal de bytes (AL=AL / fonte; AH=resto), word (AX=DX:AX / fonte, DX=resto) e dword (EAX=EDX:EAX / fonte, EDX=resto)
Conversão e Ajuste	
CBW	Converte byte (AL) para word (AX)
CWD	Converte word (AX) para dword (DX:AX)
CWDE	Converte word (AX) para dword (EAX)
CDQ	Converte dword (EAX) para qword (EDX:EAX)
DAA	Ajuste decimal após a adição (AL)
DAS	Ajuste decimal após a subtração (AL)
AAA	Ajuste ASCII após a adição (AL)
AAS	Ajuste ASCII após a subtração (AL)
AAM	Ajuste ASCII após a multiplicação (AX)
AAD	Ajuste ASCII após a divisão (AX)
MOVSX r16/r32, fonte	Move **fonte** para **r16/r32** com extensão de sinal
MOVZX r16/r32, fonte	Move **fonte** para **r16/r32** com extensão de zeros
Manipulação de bits	
NOT destino	“NÃO” lógico
AND destino, fonte	“E” lógico
OR destino, fonte	“OU” lógico

XOR destino, fonte	"OU-EXCLUSIVO" lógico
TEST destino, fonte	Seta os flags de acordo com o "E" lógico (não altera o **destino**)
SHR reg/mem, nbits	 **nbits**=const. ou CL
SAR reg/mem, nbits	 **nbits**=const. ou CL
SHL reg/mem, nbits	 **nbits**=const. ou CL
SAL reg/mem, nbits	 **nbits**=const. ou CL
ROR reg/mem, nbits	 **nbits**=const. ou CL
RCR reg/mem, nbits	 **nbits**=const. ou CL
ROL reg/mem, nbits	 **nbits**=const. ou CL
RCL reg/mem, nbits	 **nbits**=const. ou CL
Instruções de Desvio	
JMP destino	Desvia para o endereço **destino**
LOOPE destino	Decrementa CX (ECX) e desvia para **destino** se for zero
LOOPZ destino	Idem
LOOPNE destino	Decrementa CX (ECX) e desvia para **destino** se não for zero
LOOPNZ destino	Idem
Instruções de Desvio Condicional (operações com sinal)	
JG/JNLE destino	Desvia se **maior / não menor nem igual**
JGE/JNL destino	Desvia se **maior ou igual / não menor**
JL/JNGE destino	Desvia se **menor / não maior nem igual**
JLE/JNG destino	Desvia se **menor ou igual / não maior**
JO destino	Desvia se **overflow ligado**
JS destino	Desvia se **sinal ligado**
JNO destino	Desvia se **overflow desligado**
JNS destino	Desvia se **sinal desligado**
Instruções de Desvio Condicional (operações sem sinal)	
JÁ/JNBE destino	Desvia se **acima / não abaixo nem igual**
JAE/JNB destino	Desvia se **acima ou igual / não abaixo**
JB/JNAE destino	Desvia se **abaixo / não acima nem igual**
JBE/JNA destino	Desvia se **abaixo ou igual / não acima**
Instruções de Desvio Condicional	
JC destino	Desvia se carry=1
JNC destino	Desvia se carry=0
JE/JZ destino	Desvia se zero=1
JNE/JNZ destino	Desvia se zero=0
JP/JPE destino	Desvia se paridade=1 (even)
JNP/JPO destino	Desvia se paridade=0 (odd)
Subrotinas	
CALL endereço	Chama a subrotina do **endereço**
RET valor_opcional	Retorna da subrotina. Retorna **valor_opcional**
Manipulação de strings	
MOVSx (x=B,W ou D)	**ES:DI ← DS:SI** INC SI; INC DI
CMPSx (x=B,W ou D)	Seta flags de acordo com **(DS:SI - EX:DI)**; INC SI; INC DI
SCASx (x=B,W ou D)	Seta flags de acordo com **(AL,AX ou EAX – ES:DI)** INC SI; INC DI
LODSx (x=B,W ou D)	**AL,AX ou EAX ← DS:SI** INC SI; INC DI
STOSx (x=B,W ou D)	**ES:DI ← AL,AX ou EAX** INC SI; INC DI