insr.	x0	x1	x2	x3	x4	x5	x6	x7	x8	x9	xA	xB	xC	xD	xE	xF
0x	ADD (byte | word) reg/mem ← reg		ADD (byte | word) reg/mem → reg		ADD (byte | word) AL | eAX ← imm		PUSH ES | SS	POP ES | SS	OR (byte | word) reg/mem ← reg		OR (byte | word) reg/mem → reg		OR (byte | word) AL | eAX ← imm		PUSH CS | DS	⊕ 0F xx →
1x	ADC (byte | word) reg/mem ← reg		ADC (byte | word) reg/mem → reg		ADC (byte | word) AL | eAX ← imm				SBB (byte | word) reg/mem ← reg		SBB (byte | word) reg/mem → reg		SBB (byte | word) AL | eAX ← imm			POP DS
2x	AND (byte | word) reg/mem ← reg		AND (byte | word) reg/mem → reg		AND (byte | word) AL | eAX ← imm		seg.ES Prefix	DAA ⊕	SUB (byte | word) reg/mem ← reg		SUB (byte | word) reg/mem → reg		SUB (byte | word) AL | eAX ← imm		seg.CS Prefix	DAS ⊕
3x	XOR (byte | word) reg/mem ← reg		XOR (byte | word) reg/mem → reg		XOR (byte | word) AL | eAX ← imm		seg.SS Prefix	AAA ⊕	CMP (byte | word) reg/mem ← reg		CMP (byte | word) reg/mem → reg		CMP (byte | word) AL | eAX ← imm		seg.DS Prefix	AAS ⊕
4x	INC (word) eAX | eCX | eDX | eBX | eSP | eBP | eSI | eDI								DEC (word) eAX | eCX | eDX | eBX | eSP | eBP | eSI | eDI							
5x	PUSH (word) eAX | eCX | eDX | eBX | eSP | eBP | eSI | eDI								POP (word) eAX | eCX | eDX | eBX | eSP | eBP | eSI | eDI							
6x	PUSHA eAX → eDI	POPA eDI → eAX	BOUND ⊕	ARPL ⊕	seg.FS Prefix	seg.GS Prefix	o.size Prefix	o.addr Prefix	PUSH imm	IMUL (w) r/m*imm	PUSH imm8	IMUL (w) r/m*imm8	INSB byte | word [eDI] ← DX	INS	OUTSB byte | word [eSI] → DX	OUTS
7x	Jcc (short) O	NO	B/NAE/C	NB/AE/NC	Z/E	NZ/NE	BE/NA	NBE/A	S	NS	P/PE	NP/PO	L/NGE	NL/GE	LE/NG	NLE/G
8x	Immediate Group 1 r/m ← imm8 | imm		r/m ← imm8 | imm8		TEST (byte | word) reg/mem ↔ reg		XCHG (byte | word) reg/mem ↔ reg		MOV (byte | word) reg/mem ← reg		MOV (byte | word) reg/mem → reg		MOV (16) r/m ← sreg	LEA reg ← mem	MOV (16) r/m → sreg	Grp.1A Insr. Group
9x	XCHG eAX, reg (word) eAX (NOP)	eCX	eDX	eBX	eSP	eBP	eSI	eDI	CBW or CWDE	CWD or CDQ	CALL far	FWAIT or WAIT	PUSHF eFLAGS	POPF	SAHF AH ↔ eFLAGS	LAHF
Ax	MOV (byte | word) AL | eAX ← ptr[imm]		MOV (byte | word) AL | eAX → ptr[imm]		MOVSB byte | word [eDI] ← [eSI]	MOVS	CMPSB byte | word [eSI] ← [eDI]	CMPS	TEST (byte | word) AL | eAX ↔ imm		STOSB byte|word[eDI] ← AL|eAX	STOS	LODSB byte|word[eSI] → AL|eAX	LODS	SCASB AL|eAX ← byte|word[eDI]	SCAS
Bx	MOV (byte) AL | CL | DL | BL | AH | CH | DH | BH ← imm								MOV (word) eAX | eCX | eDX | eBX | eSP | eBP | eSI | eDI ← imm							
Cx	Shift Grp.2 byte | word ← imm8		RET (near) imm16	0	LES ⊕	LDS ⊕	Grp.11 (MOV) byte | word ← imm		ENTER ⊕	LEAVE	RET (far) 0	imm16	INT 3 (#BP)	imm8	INTO	IRET
Dx	Shift Grp.2 byte | word ← 1		Shift Grp.2 byte | word ← CL		AAM ⊕	AAD ⊕	SALC Undoc.	XLAT or XLATB	FPU Instructions 8087 Instruction Group							
Ex	LOOPNE	LOOPE	LOOP	JeCXZ short jump (imm8)	IN (imm8: byte | word) reg/mem ← I/O		OUT (imm8: byte | word) reg/mem → I/O		CALL near	JMP near	JMP far	JMP short	IN (DX: byte | word) reg/mem ← I/O		OUT (DX: byte | word) reg/mem → I/O	
Fx	LOCK Prefix	ICEBP Undoc.	REPNE Prefix	REPE or REP	HLT	CMC	Unary Grp.3 (b | w) r/m = ⊕(reg/mem)		CLC eFLAGS.CF ← 0 | 1	STC	CLI eFLAGS.IF ← 0 | 1	STI	CLD eFLAGS.DF ← 0 | 1	STD	Grp.4 INC/DEC	Grp.5 INC/DEC/...