8086 opcode map

Operand order: MOV DST,SRC

https://hackaday.io/project/193288-improved-8086-opcode-map

20231023 RoelH

	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F
0x00	ADD b r/m reg	ADD w r/m reg	ADD b reg r/m	ADD w reg r/m	ADD b AL imm	ADD w AX imm	PUSH w ES	POP w ES	OR b r/m reg	OR w r/m reg	OR b reg r/m	OR w reg r/m	OR b AL imm	OR w AX imm	PUSH w CS	
0x10	ADC b r/m reg	ADC w r/m reg	ADC b reg r/m	ADC w reg r/m	ADC b AL imm	ADC w AX imm	PUSH w SS	POP w SS	SBB b r/m reg	SBB w r/m reg	SBB b reg r/m	SBB w reg r/m	SBB b AL imm	SBB w AX imm	PUSH w DS	POP w DS
0x20	AND b r/m reg	AND w r/m reg	AND b reg r/m	AND w reg r/m	AND b AL imm	AND w AX imm	PREFIX ES:	DAA	SUB b r/m reg	SUB w r/m reg	SUB b reg r/m	SUB w reg r/m	SUB b AL imm	SUB w AX imm	PREFIX CS:	DAS
0x30	XOR b r/m reg	XOR w r/m reg	XOR b reg r/m	XOR w reg r/m	XOR b AL imm	XOR w AX imm	PREFIX SS:	AAA	CMP b r/m reg	CMP w r/m reg	CMP b reg r/m	CMP w reg r/m	CMP b AL imm	CMP w AX imm	PREFIX DS:	AAS
0x40	INC w AX	INC w CX	INC w DX	INC w BX	INC w SP	INC w BP	INC w SI	INC w DI	DEC w AX	DEC w CX	DEC w DX	DEC w BX	DEC w SP	DEC w BP	DEC w SI	DEC w DI
0x50	PUSH w AX	PUSH w CX	PUSH w DX	PUSH w BX	PUSH w SP	PUSH w BP	PUSH w SI	PUSH w DI	POP w AX	POP w CX	POP w DX	POP w BX	POP w SP	POP w BP	POP w SI	POP w DI
0x60																
0x70	JO signed overflow	JNO signed no overflow	JB unsign below	JNB unsign not below	JZ signed zero	JNZ signed not zero	JBE unsign below or equal	JA unsign above	JS signed sign	JNS signed not sign	JPE signed parity even	JPO signed parity odd	JL signed less	JGE signed greater or eq	JLE signed less or equal	JG signed greater
0x80	IMM b r/m imm8	IMM w r/m imm16	IMM b r/m imm8	IMM w, ext r/m imm8	TEST b reg r/m	TEST w reg r/m	XCHG b reg r/m	XCHG w reg r/m	MOV b r/m reg	MOV w r/m reg	MOV b reg r/m	MOV w reg r/m	MOV w r/m sreg	LEA w reg r/m	MOV w sreg r/m	POP w r/m
0x90	NOP	XCHG w CX AX	XCGH w DX AX	XCHG w BX AX	XCHG w SP AX	XCGH w BP AX	XCHG w SI AX	XCHG w DI AX	CBW ext (byte to word)	CWD ext (word to dword)	CALL far addr16	WAIT	PUSHF	POPF	SAHF (AH to flags)	LAHF (flags to AH)
0xA0	MOV b AL addr	MOV w AX addr	MOV b addr AL	MOV w addr AX	MOVSB	MOVSW	CMPSB	CMPSW	TEST b AL imm8	TEST w AX imm16	STOSB	STOSW	LODSB	LODSW	SCASB	SCASW
0xB0	MOV b AL imm8	MOV b CL imm8	MOV b DL imm8	MOV b BL imm8	MOV b AH imm8	MOV b CH imm8	MOV b DH imm8	MOV b BH imm8	MOV w AX imm16	MOV w CX imm16	MOV w DX imm16	MOV w BX imm16	MOV w SP imm16	MOV w BP imm16	MOV w SI imm16	MOV w DI imm16
0xC0			RET imm16	RET	LES w r/m	LDS w r/m	MOV b r/m imm8	MOV w r/m imm16			RET far imm16	RET far	INT 3	INT b vector	INTO int on overflow	IRET
0xD0	SHIFT b r/m 1	SHIFT w r/m 1	SHIFT b r/m CL	SHIFT w r/m CL	AAM r/m	AAD r/m		XLAT	ESC r/m	ESC r/m	ESC r/m	ESC r/m	ESC r/m	ESC r/m	ESC r/m	ESC r/m
0xE0	LOOPNZ displ	LOOPZ displ	LOOP displ	JCXZ displ	IN b AL port	IN w AX port	OUT b port AL	OUT w port AX	CALL w disp16	JMP disp16	JMP far disp16	JMP disp8	IN AL DX	IN AX DX	OUT DX AL	OUT DX AX
0xF0	LOCK prefix		REPNZ prefix	REP/Z prefix	HLT	CMC	Grp1 b r/m	Grp1 w r/m	CLC	STC	CLI	STI	CLD	STD	Grp2b b r/m	Grp2w w r/m

instruction	
prefix	(opt)
opcode	
r/m	(opt)
DispL	(opt)
DispH	(opt)
ImmL	(opt)
ImmH	(opt)

r/m		
mod	reg / op	R/M
0-3	0-7	0-7

AX:	AH	AL
CX:	CH	CL
DX:	DH	DL
BX:	BH	BL

mod 0	mod 1	mod 2	mod 3
A	A+d8	A+d16	reg
A	A+d8	A+d16	reg
A	A+d8	A+d16	reg
A	A+d8	A+d16	reg
A	A+d8	A+d16	reg
A	A+d8	A+d16	reg
addr	A+d8	A+d16	reg
A	A+d8	A+d16	reg

d8 will be sign-extended to d16

R/M	A
0	(BX)+(SI)
1	(BX)+(DI)
2	(BP)+(SI)
3	(BP)+(DI)
4	(SI)
5	(DI)
6	(BP)
7	(BX)

reg	byte	word	sreg
0	AL	AX	ES
1	CL	CX	CS
2	DL	DX	SS
3	BL	BX	DS
4	AH	SP	
5	CH	BP	
6	DH	SI	
7	BH	DI	

op	Grp1	Grp2b	Grp2w	IMM	SHIFT
0	TEST	INC	INC	ADD	ROL
1		DEC	DEC	OR	ROR
2	NOT		CALL	ADC	RCL
3	NEG		CALLfar	SBB	RCR
4	MUL		JMP	AND	SHL
5	IMUL		JMP far	SUB	SHR
6	DIV		PUSH	XOR	
7	IDIV			CMP	SAR

d=0	reg is source
d=1	reg is destination
s=0	no sign extension
s=1	sign extension (ext)
r/m	Has ModRM
	Flow control
	String instruction
	MOV
sreg	segment register