80x86 Integer Instruction Set (8088 - Pentium)

Legend:

General

acc = AL, AX or EAX unless specified otherwise
reg = any general register
r8 = any 8-bit register
r16 = any general purpose 16-bit register
r32 = any general purpose 32-bit register
imm = immediate data
imm8 = 8-bit immediate data
imm16 = 16-bit immediate data
mem = memory address
mem8 = address of 8-bit data item
mem16 = address of 16-bit data item
mem32 = address of 32-bit data item
mem48 = address of 48-bit data item
dest = 16/32-bit destination
short = 8-bit destination

Integer instruction timings:

n - generally refers to a number of repeated counts
m - in a jump or call;
286: bytes in next instruction
386/486: number of components
(each byte of opcode) + 1 (if immed data) + 1 (if displacement)
EA = cycles to calculate the Effective Address
8088/8086:
base = 5 BP+DI or BX+SI = 7 BP+DI+disp or BX+SI+disp = 11
index = 5 BX+DI or BP+SI = 8 BX+DI+disp or BP+SI+disp = 12
disp = 6 segment override = +2
286 - 486:
base+index+disp = +1 all others, no penalty

instruction length:

The byte count includes the opcode length and length of any required displacement or immediate data. If the displacement is optional, it is shown as d() with the possible lengths in parentheses. If the immediate data is optional, it is shown as i() with the possible lengths in parentheses.

pairing categories for Pentium:

NP = not pairable
UV = pairable in the U pipe or V pipe
PU = pairable in the U pipe only
PV = pairable in the V pipe only
(end of legend)

Instruction formats, clock cycles and Pentium® Pairing info

AAA ASCII adjust after addition

	bytes	8088	186	286	386	486	Pentium

1	8	8	3	4	3	3	NP

Example: aaa

AAD ASCII adjust AX before division (second byte is divisor)

bytes	8088	186	286	386	486	Pentium	
2	60	15	14	19	14	10	NP

Example: aad

AAM ASCII adjust AX after multiply (second byte is divisor)

bytes	8088	186	286	386	486	Pentium	
2	83	19	16	17	15	18	NP

Example: aam

AAS ASCII adjust AL after subtraction

bytes	8088	186	286	386	486	Pentium	
1	8	7	3	4	3	3	NP

Example: aas

ADC Integer add with carry

operands	bytes	8088	186	286	386	486	Pentium	
reg, reg	2	3	3	2	2	1	1	PU
mem, reg	2+d(0,2)	24+EA	10	7	7	3	3	PU
reg, mem	2+d(0,2)	13+EA	10	7	6	2	2	PU
reg, imm	2+i(1,2)	4	4	3	2	1	1	PU
mem, imm	2+d(0,2) +i(1,2)	23+EA	16	7	7	3	3	PU*
acc, imm	1+i(1,2)	4	4	3	2	1	1	PU

* = not pairable if there is a displacement and immediate

Example: adc eax, ebx

ADD Integer addition

operands	bytes	8088	186	286	386	486	Pentium	
reg, reg	2	3	3	2	2	1	1	UV
mem, reg	2+d(0,2)	24+EA	10	7	7	3	3	UV
reg, mem	2+d(0,2)	13+EA	10	7	6	2	2	UV
reg, imm	2+i(1,2)	4	4	3	2	1	1	UV
mem, imm	2+d(0,2) +i(1,2)	23+EA	16	7	7	3	3	UV*
acc, imm	1+i(1,2)	4	4	3	2	1	1	UV

* = not pairable if there is a displacement and immediate

Example: add eax, ebx

AND Logical AND

operands	bytes	8088	186	286	386	486	Pentium	
reg, reg	2	3	3	2	2	1	1	UV
mem, reg	2+d(0,2)	24+EA	10	7	7	3	3	UV
reg, mem	2+d(0,2)	13+EA	10	7	6	2	2	UV
reg, imm	2+i(1,2)	4	4	3	2	1	1	UV
mem, imm	2+d(0,2) +i(1,2)	23+EA	16	7	7	3	3	UV*
acc, imm	1+i(1,2)	4	4	3	2	1	1	UV

* = not pairable if there is a displacement and immediate

Example: and eax, ebx

ARPL Adjust RPL field of selector (286+)

operands	bytes	286	386	486	Pentium	
reg, reg	2	10	20	9	7	NP
mem, reg	2+d(0-2)	11	21	9	7	NP

Example: arpl ax, bx

BOUND Check array index against bounds (186+)

operands	bytes	186	286	386	486	Pentium	
reg, mem	4	35	13	10	7	8	NP

Example: bound bx, array

BSF Bit scan forward (386+)

operands	bytes	386	486	Pentium	
r16, r16	3	10+3n	6-42	6-34	NP
r32, r32	3	10+3n	6-42	6-42	NP
r16, m16	3+d(0,1,2)	10+3n	7-43	6-35	NP
r32, m32	3+d(0,1,2,4)	10+3n	7-43	6-43	NP

Example: bsf eax, [esi]

BSR Bit scan reverse (386+)

operands	bytes	386	486	Pentium	
r16, r16	3	10+3n	6-103	7-39	NP
r32, r32	3	10+3n	7-104	7-71	NP
r16, m16	3+d(0,1,2)	10+3n	6-103	7-40	NP
r32, m32	3+d(0,1,2,4)	10+3n	7-104	7-72	NP

Example: bsr eax, [esi]

BSWAP Byte swap (486+)

operand	bytes	486	Pentium	
r32	2	1	1	NP

Example: bswap eax

BT Bit test (386+)

operands	bytes	386	486	Pentium
reg, reg	3	3	3	4 NP
mem, reg	3+d(0,1,2,4)	12	8	9 NP
reg, imm8	3+i(1)	3	3	4 NP
mem, imm8	3+d(0,1,2,4)+i(1)	6	3	4 NP

Example: bt eax, 4

BTC Bit test and complement (386+)

operands	bytes	386	486	Pentium
reg, reg	3	6	6	7 NP
mem, reg	3+d(0,1,2,4)	13	13	13 NP
reg, imm8	3+i(1)	6	6	7 NP
mem, imm8	3+d(0,1,2,4)+i(1)	8	8	8 NP

Example: btc eax, 4

BTR Bit test and reset (386+)

operands	bytes	386	486	Pentium
reg, reg	3	6	6	7 NP
mem, reg	3+d(0,1,2,4)	13	13	13 NP
reg, imm8	3+i(1)	6	6	7 NP
mem, imm8	3+d(0,1,2,4)+i(1)	8	8	8 NP

Example: btr eax, 4

BTS Bit test and set (386+)

operands	bytes	386	486	Pentium
reg, reg	3	6	6	7 NP
mem, reg	3+d(0,1,2,4)	13	13	13 NP
reg, imm8	3+i(1)	6	6	7 NP
mem, imm8	3+d(0,1,2,4)+i(1)	8	8	8 NP

Example: bts eax, 4

CALL Call subroutine

operand	bytes	8088	186	286	386	486	Pentium
near	3	23	14	7+m	7+m	3	1 PV
reg	2	20	13	7+m	7+m	5	2 NP
mem16	2+d(0-2)	29+EA	19	11+m	10+m	5	2 NP
far	5	36	23	13+m	17+m	18	4 NP
mem32	2+d(0-2)	53+EA	38	16+m	22+m	17	4 NP

Protected Mode

operand	bytes	286	386	486	Pentium
far	5	26+m	34+m	20	4-13 NP
mem32	2+d(0-2)	29+m	38+m	20	5-14 NP

cycles not shown for calls through call and task gates

Example: call my_function

CBW Convert byte to word (AL --> AX)

bytes	8088	186	286	386	486	Pentium
1	2	2	2	3	3	3 NP

Example: cbw

CWDE Convert word to dword (386+) (AX --> EAX)

bytes	386	486	Pentium
1	3	3	3 NP

Example: cwde

CWD Convert word to double (AX --> DX:AX)

bytes	8088	186	286	386	486	Pentium
1	5	4	2	2	3	2 NP

Example: cwd

CDQ Convert double to quad (EAX --> EDX:EAX)

bytes	386	486	Pentium
1	2	3	2 NP

Example: cdq

CLC Clear the carry flag

bytes	8088	186	286	386	486	Pentium
1	2	2	2	2	2	2 NP

Example: clc

CLD Clear the direction flag (set to forward direction)

bytes	8088	186	286	386	486	Pentium
1	2	2	2	2	2	2 NP

Example: cld

CLI Clear the interrupt flag (disable interrupts)

bytes	8088	186	286	386	486	Pentium
1	2	2	3	3	5	7 NP

Example: cli

CLTS Clear task switched flag in CR0 (286+)

bytes	286	386	486	Pentium
2	2	5	7	10 NP

Example: clts

CMC Complement carry flag

bytes	8088	186	286	386	486	Pentium
1	2	2	2	2	2	2 NP

Example: cmc

CMP Compare two operands

operands	bytes	8088	186	286	386	486	Pentium
reg, reg	2	3	3	2	2	1	1 UV
mem, reg	2+d(0,2)	13+EA	10	7	5	2	2 UV
reg, mem	2+d(0,2)	13+EA	10	6	6	2	2 UV
reg, imm	2+i(1,2)	4	4	3	2	1	1 UV
mem, imm	2+d(0,2) +i(1,2)	14+EA	10	6	5	2	2 UV*
acc, imm	1+i(1,2)	4	4	3	2	1	1 UV

* = not pairable if there is a displacement and immediate

Example: cmp eax, 3

CMPS/CMPSB/CMPSW/CMPSD Compare string operands

variations	bytes	8088	186	286	386	486	Pentium
cmpsb	1	30	22	8	10	8	5 NP
cmpsw	1	-	-	-	10	8	5 NP
cmpsd	1	-	-	-	10	8	5 NP
repX cmpsb	2	9+30n	5+22n	5+9n	5+9n	7+7n*	9+4n NP
repX cmpsw	2	9+30n	5+22n	5+9n	5+9n	7+7n*	9+4n NP
repX cmpsd	2	-	-	-	5+9n	7+7n*	9+4n NP

repX = repe, repz, repne or repnz
* : 5 if n = 0

Example: repne cmpsb

CMPXCHG Compare and Exchange (486+)

operands	bytes	486	Pentium
reg, reg	3	6	5 NP
mem, reg	3+d(0-2)	7-10	6 NP

Example: cmpxchg ebx, edx

CMPXCHG8B Compare and Exchange 8 bytes (Pentium+)

operands	bytes	Pentium
mem, reg	3+d(0-2)	10 NP

Example: cmpxchg8b [ebx], edx

CPUID CPU identification (Pentium+)

bytes	Pentium
2	14 NP

Example: cpuid

DAA Decimal adjust AL after addition

bytes	8088	186	286	386	486	Pentium
1	4	4	3	4	2	3 NP

Example: daa

DAS Decimal adjust AL after subtraction

bytes	8088	186	286	386	486	Pentium
1	4	4	3	4	2	3 NP

Example: das

DEC Decrement

operand	bytes	8088	186	286	386	486	Pentium
r8	2	3	3	2	2	1	1 UV
r16	1	3	3	2	2	1	1 UV
r32	1	3	3	2	2	1	1 UV
mem	2+d(0,2)	23+EA	15	7	6	3	3 UV

Example: dec eax

DIV Unsigned divide

operand	bytes	8088	186	286	386	486	Pentium
r8	2	80-90	29	14	14	16	17 NP
r16	2	144-162	38	22	22	24	25 NP
r32	2	-	-	-	38	40	41 NP
mem8	2+d(0-2)	86-96+EA	35	17	17	16	17 NP
mem16	2+d(0-2)	150-168+EA	44	25	25	24	25 NP
mem32	2+d(0-2)	-	-	-	41	40	41 NP

implied dividend		operand		quotient	remainder
AX	/	byte	=	AL	AH
DX:AX	/	word	=	AX	DX
EDX:EAX	/	dword	=	EAX	EDX

Example: div ebx

ENTER Make stack frame for procedure parameters (186+)

operands	bytes	8088	186	286	386	486	Pentium
imm16, 0	3	-	15	11	10	14	11 NP
imm16, 1	4	-	25	15	12	17	15 NP
imm16, imm8	4	-	22+16n	12+4n	15+4n	17+3i	15+2i NP

n = imm8-1; i = imm8

Example: enter 1, 0

ESC Escape

escape opcodes D8 - DF are used by floating point instructions

HLT Halt

bytes	8088	186	286	386	486	Pentium
1	2	2	2	5	4	4 NP

Example: hlt

IDIV Signed divide

operand	bytes	8088	186	286	386	486	Pentium
r8	2	101-112	44-52	17	19	19	22 NP
r16	2	165-184	53-61	25	27	27	30 NP
r32	2	-	-	-	43	43	46 NP
mem8	2+d(0-2)	107-118+EA	50-58	20	22	20	22 NP
mem16	2+d(0-2)	171-190+EA	59-67	28	30	28	30 NP
mem32	2+d(0-2)	-	-	-	46	44	46 NP

implied dividend		operand		quotient	remainder
AX	/	byte	=	AL	AH
DX:AX	/	word	=	AX	DX
EDX:EAX	/	dword	=	EAX	EDX

Example: idiv ebx

IMUL Signed multiply

Accumulator Multiplies

operand	bytes	8088	186	286	386	486	Pentium
r8	2	80-98	25-28	13	9-14	13-18	11 NP
r16	2	128-154	34-37	21	9-22	13-26	11 NP
r32	2	-	-	-	9-38	13-42	10 NP
mem8	2+d(0-2)	86-104+EA	32-34	16	12-17	13-18	11 NP
mem16	2+d(0-2)	134-160+EA	40-43	24	12-25	13-26	11 NP
mem32	2+d(0-2)	-	-	-	12-41	13-42	10 NP

implied multiplicand		operand (multiplier)		result
AL	*	byte	=	AX
AX	*	word	=	DX:AX
EAX	*	dword	=	EDX:EAX

Example: imul ebx

2 and 3 operand Multiplies

operands	bytes	186	286	386	486	Pentium
r16, imm	2+i(1,2)	-	21	9-14/9-22	13-18/13-26	10 NP
r32, imm	2+i(1,2)	-	-	9-38	13-42	10 NP
r16,r16,imm	2+i(1,2)	22/29	21	9-14/9-22	13-18/13-26	10 NP
r32,r32,imm	2+i(1,2)	-	-	9-38	13-42	10 NP
r16,m16,imm	2+d(0-2)+i(1,2)	25/32	24	12-17/12-25	13-18/13-26	10 NP
r32,m32,imm	2+d(0-2)+i(1,2)	-	12-41		13-42	10 NP

r16, r16	2+i(1,2)	-	-	9-22	13-18/13-26	10	NP
r32, r32	2+i(1,2)	-	-	9-38	13-42	10	NP
r16, m16	2+d(0-2)+i(1,2)	-	12-25	13-18/13-26	10	NP	
r32, m32	2+d(0-2)+i(1,2)	-	12-41	13-42	10	NP	

all forms: dest, src cycles for: byte/word
or dword
dest, src1, src2

Example: imul eax, ebx, 10

IN Input from port

operands	bytes	8088	186	286	386	486	Pentium
al, imm8	2	14	10	5	12	14	7 NP
ax, imm8	2	14	10	5	12	14	7 NP
eax, imm8	2	-	-	-	12	14	7 NP
al, dx	1	12	8	5	13	14	7 NP
ax, dx	1	12	8	5	13	14	7 NP
eax, dx	1	-	-	-	13	14	7 NP

Protected mode

operands	bytes	386	486	Pentium
acc, imm	2	6/26/26	9/29/27	4/21/19 NP
acc, dx	1	7/27/27	8/28/27	4/21/19 NP

cycles for: CPL <= IOPL / CPL > IOPL / V86

Example: in al, dx

INC Increment

operand	bytes	8088	186	286	386	486	Pentium
r8	2	3	3	2	2	1	1 UV
r16	1	3	3	2	2	1	1 UV
r32	1	3	3	2	2	1	1 UV
mem	2+d(0,2)	23+EA	15	7	6	3	3 UV

Example: inc ebx

INS/INSB/INSW/INSD Input from port to string

variations	bytes	8088	186	286	386	486	Pentium
insb	1	-	14	5	15	17	9 NP
insw	1	-	14	5	15	17	9 NP
insd	1	-	-	-	15	17	9 NP

Protected Mode

bytes	386	486	Pentium
1	9/29/29	10/32/30	6/24/22 NP

cycles for: CPL <= IOPL / CPL > IOPL / V86

Example: rep insb

INT Call interrupt procedure

operands	bytes	8088	186	286	386	486	Pentium	
3	1	72	45	23+m	33	26	13	NP
imm8	2	71	47	23+m	37	30	16	NP

Protected mode

bytes	8088	186	286	386	486	Pentium	
1	-	-	(40-78)+m	59-99	44-71	27-82	NP

Example: int 21h

INTO Call interrupt procedure if overflow

bytes	8088	186	286	386	486	Pentium	
1	4/73	4/48	3/24+m	3/35	3/28	4/13	NP

Protected mode

bytes	286	386	486	Pentium	
1	(40-78)+m	59-99	44-71	27-56	NP

Task switch clocks not shown

Example: into

INVD Invalidate data cache (486+)

bytes	8088	186	286	386	486	Pentium	
2	-	-	-	-	4	15	NP

Example: invd

INVLPG Invalidate TLB entry (486+)

operands	bytes	486	Pentium	
mem32	5	12	25	NP

Example: invlpg [eax]

IRET Return from interrupt

bytes	8088	186	286	386	486	Pentium	
1	44	28	17+m	22	15	8-27	NP

Task switch clocks not shown

Example: iret

IRETD 32-bit return from interrupt (386+)

bytes	386	486	Pentium	
1	22	15	10-27	NP

Task switch clocks not shown

Example: iretd

Jcc Jump on condition code

operand	bytes	8088	186	286	386	486	Pentium
near8	2	4/16	4/13	3/7+m	3/7+m	1/3	1 PV
near16	3	-	-	-	3/7+m	1/3	1 PV

cycles for: no jump/jump

conditional jump instructions:

ja	jump if above	jnbe	jump if not below or equal
jae	jump if above or equal	jnb	jump if not below
jb	jump if below	jnae	jump if not above or equal
jbe	jump if below or equal	jna	jump if not above
jg	jump if greater	jnle	jump if not less or equal
jge	jump if greater or equal	jnl	jump if not less
jl	jump if less	jnge	jump if not greater or equal
jle	jump if less or equal	jng	jump if not greater
je	jump if equal	jz	jump if zero
jne	jump if not equal	jnz	jump if not zero
jc	jump if carry	jnc	jump if not carry
js	jump if sign	jns	jump if not sign
jnp	jump if no parity (odd)	jpo	jump if parity odd
jo	jump if overflow	jno	jump if not overflow
jp	jump if parity (even)	jpe	jump if parity even

Example: jne not_equal

JCXZ/JECXZ Jump if CX/ECX = 0

operand	bytes	8088	186	286	386	486	Pentium
dest	2	6/18	5/16	4/8+m	5/9+m	5/8	5/6 NP
dest	2	-	-	-	5/9+m	5/8	5/6 NP

cycles for: no jump/jump

Example: jcxz cx_is_zero

JMP Unconditional jump

operand	bytes	8088	186	286	386	486	Pentium
short	2	15	13	7+m	7+m	3	1 PV
near	3	15	13	7+m	7+m	3	1 PV
far	5	15	13	11+m	12+m	17	3 NP
r16	2	11	11	7+m	7+m	5	2 NP
mem16	2+d(0,2)	18+EA	17	11+m	10+m	5	2 NP
mem32	2+d(4)	24+EA	26	15+m	12+m	13	4 NP
r32	2	-	-	-	7+m	5	2 NP
mem32	2+d(0,2)	-	-	-	10+m	5	2 NP
mem48	2+d(6)	-	-	-	12+m	13	4 NP

cycles for jumps through call gates not shown

Example: jmp target_address

LAHF Load flags into AH

bytes	8088	186	286	386	486	Pentium
1	4	2	2	2	3	2 NP

Example: lahf

LAR Load access rights byte (286+)

operands	bytes	286	386	486	Pentium
r16, r16	3	14	15	11	8 NP
r32, r32	3	-	15	11	8 NP
r16, m16	3	16	16	11	8 NP
r32, m32	3	-	16	11	8 NP

Example: lar eax, ebx

LDS Load far pointer

operands	bytes	8088	186	286	386	486	Pentium
reg, mem	2+d(2)	24+EA	18	7	7	6	4 NP

Example: lds si, ptr_1

LES Load far pointer

operands	bytes	8088	186	286	386	486	Pentium
reg, mem	2+d(2)	24+EA	18	7	7	6	4 NP

Example: les di, ptr_2

LFS Load far pointer (386+)

operands	bytes	386	486	Pentium
reg, mem	3+d(2,4)	7	6	4 NP

Example: lfs si, ptr_3

LGS Load far pointer (386+)

operands	bytes	386	486	Pentium
reg, mem	3+d(2,4)	7	6	4 NP

Example: lgs si, ptr_4

LSS Load stack segment and offset

operands	bytes	386	486	Pentium
reg, mem	3+d(2,4)	7	6	4 NP

Example: lss bp, ptr_5

LEA Load effective address

operands	bytes	8088	186	286	386	486	Pentium	
r16, mem	2+d(2)	2+EA	6	3	2	1-2	1	UV
r32, mem	2+d(2)	-	-	-	2	1-2	1	UV

Example: lea eax, [eax+ebx*2+3]

LEAVE High level procedure exit (186+)

bytes	186	286	386	486	Pentium	
1	8	5	4	5	3	NP

Example: leave

LGDT Load global descriptor table register (286+)

operand	bytes	286	386	486	Pentium	
mem48	5	11	11	11	6	NP

Example: lgdt descriptor[ebx]

LIDT Load interrupt descriptor table register (286+)

operand	bytes	286	386	486	Pentium	
mem48	5	12	11	11	6	NP

Example: lidt descriptor[ebx]

LLDT Load local descriptor table register (286+)

operand	bytes	286	386	486	Pentium	
r16	3	17	20	11	9	NP
mem16	3+d(0-2)	19	24	11	9	NP

Example: lldt ax

LMSW Load machine status word (286+)

operand	bytes	286	386	486	Pentium	
r16	3	3	10	13	8	NP
mem16	3+d(0-2)	6	13	13	8	NP

Example: lmsw ax

LOCK Lock bus on next instruction (prefix)

bytes	8088	186	286	386	486	Pentium	
1	2	2	0	0	1	1	NP

(Note: xchg always is locked whether it is specified or not)

Example: lock mov mem, 1

LODS/LODSB/LODSW/LODSD Load string operand

variations	bytes	8088	186	286	386	486	Pentium

lodsb	1	16	10	5	5	5	2 NP
lodsw	1	16	10	5	5	5	2 NP
lodsd	1	-	-	-	5	5	2 NP

Example: lodsb

LOOP Loop control with CX counter

operand	bytes	8088	186	286	386	486	Pentium
short	2	5/17	5/15	4/8+m	11+m	6/7	5/6 NP

loopw short (uses CX in 32-bit mode)
loopd short (uses ECX in 16-bit mode)

Example: loop loop_start

LOOPE/LOOPZ Loop while equal (or zero)

operand	bytes	8088	186	286	386	486	Pentium
short	2	6/18	5/16	4/8	11+m	6/9	7/8 NP

loopew short (uses CX in 32-bit mode)
loopzw short (uses CX in 32-bit mode)
looped short (uses ECX in 16-bit mode)
loopzd short (uses ECX in 16-bit mode)

Example: loope loop_start

LOOPNE/LOOPNZ Loop while not equal (or not zero)

operand	bytes	8088	186	286	386	486	Pentium
short	2	5/19	5/16	4/8	11+m	6/9	7/8 NP

loopnew short (uses CX in 32-bit mode)
loopnzw short (uses CX in 32-bit mode)
loopned short (uses ECX in 16-bit mode)
loopnzd short (uses ECX in 16-bit mode)

Example: loopne loop_start

LSL Load segment limit (286+)

operands	bytes	286	386	486	Pentium
r16, r16	3	14	20/25	10	8 NP
r32, r32	3	-	20/25	10	8
r16, m16	3+d(0,2)	16	21/26	10	8
r32, m32	3+d(0,2)	-	21/26	10	8

Example: lsl eax, ebx

LTR Load task register (286+)

operand	bytes	286	386	486	Pentium
r16	3	17	23	20	10 NP
mem16	3+d(0,2)	19	27	20	10

Example: ltr ax

MOV Move data

operands	bytes	8088	186	286	386	486	Pentium	
reg, reg	2	2	2	2	2	1	1	UV
mem, reg	2+d(0-2)	13+EA	9	3	2	1	1	UV
reg, mem	2+d(0-2)	12+EA	12	5	4	1	1	UV
mem, imm	2+d(0-2) +i(1,2)	14+EA	12-13	3	2	1	1	UV*
reg, imm	2+i(1,2)	4	3-4	2	2	1	1	UV
acc, mem	3	14	8	5	4	1	1	UV
mem, acc	3	14	9	3	2	1	1	UV

* = not pairable if there is a displacement and immediate

Example: mov eax, ebx

Segment Register Moves

Real Mode

operands	bytes	8088	186	286	386	486	Pentium	
seg, r16	2	2	2	2	2	3	2-11	NP
seg, m16	2+d(0,2)	12+EA	9	5	5	3	3-12	NP
r16, seg	2	2	2	2	2	3	1	NP
m16, seg	2+d(0,2)	13+EA	11	3	2	3	1	NP

Example: mov ds, ax

Protected Mode Differences

operands	bytes	286	386	486	Pentium	
seg, r16	2	17	18	9	2-11*	NP
seg, m16	2+d(0,2)	19	19	9	3-12*	NP

* = add 8 if new descriptor; add 6 if SS

MOVE to/from special registers (386+)

operands	bytes	386	486	Pentium	
r32, cr32	3	6	4	4	NP
cr32, r32	3	4/10*	4/16*	12/22*	NP
r32, dr32	3	14/22*	10	2/12*	NP
dr32, r32	3	16/22*	11	11/12*	NP
r32, tr32	3	12	3/4*	-	NP
tr32, r32	3	12	4/6*	-	NP

* = cycles depend on which special register

Example: mov cr0, eax

MOVS/MOVSB/MOVSW/MOVSD Move data from string to string

variations	bytes	8088	186	286	386	486	Pentium	
movsb	1	18	9	5	7	7	4	NP
movsw	1	26	9	5	7	7	4	NP

movsd	1	-	-	-	7	7	4 NP
rep movsb	2	9+17n	8+8n	5+4n	7+4n	12+3n*	3+n NP
rep movsw	2	9+25n	8+8n	5+4n	7+4n	12+3n*	3+n NP
rep movsd	2	-	-	-	7+4n	12+3n*	3+n NP

* = 5 if n=0, 13 if n=1
(n = count of bytes, words or dwords)

Example: rep movsb

MOVSX Move with sign-extend (386+)

operands	bytes	386	486	Pentium
reg, reg	3	3	3	3 NP
reg, mem	3+d(0,1,2,4)	6	3	3 NP

(Note: destination reg is 16 or 32-bits; source is 8 or 16 bits)

Example: movsx ebx, ax

MOVZX Move with zero-extend (386+)

operands	bytes	386	486	Pentium
reg, reg	3	3	3	3 NP
reg, mem	3+d(0,1,2,4)	6	3	3 NP

(Note: destination reg is 16 or 32-bits; source is 8 or 16 bits)

Example: movzx ebx, ax

MUL Unsigned multiply

operand	bytes	8088	186	286	386	486	Pentium
r8	2	70-77	26-28	13	9-14	13-18	11 NP
r16	2	118-133	35-37	21	9-22	13-26	11 NP
r32	2	-	-	-	9-38	13-42	10 NP
mem8	2+d(0-2)	76-83+EA	32-34	16	12-17	13-18	11 NP
mem16	2+d(0-2)	124-139+EA	41-43	24	12-25	13-26	11 NP
mem32	2+d(0-2)	-	-	-	12-41	13-42	10 NP

implied multiplicand		operand (multiplier)		result
AL	*	byte	=	AX
AX	*	word	=	DX:AX
EAX	*	dword	=	EDX:EAX

Example: mul ebx

NEG Two's complement negation

operand	bytes	8088	186	286	386	486	Pentium
reg	2	3	3	2	2	1	1 NP
mem	2+d(0-2)	24+EA	13	7	6	3	3 NP

Example: neg eax

NOP No operation

bytes	8088	186	286	386	486	Pentium	
1	3	3	3	3	1	1	UV

Example: nop

NOT One's complement negation

operands	bytes	8088	186	286	386	486	Pentium	
reg	2	3	3	2	2	1	1	NP
mem	2+d(0-2)	24+EA	13	7	6	3	3	NP

Example: not eax

OR Logical inclusive or

operands	bytes	8088	186	286	386	486	Pentium	
reg, reg	2	3	3	2	2	1	1	UV
mem, reg	2+d(0,2)	24+EA	10	7	7	3	3	UV
reg, mem	2+d(0,2)	13+EA	10	7	6	2	2	UV
reg, imm	2+i(1,2)	4	4	3	2	1	1	UV
mem, imm	2+d(0,2) +i(1,2)	23+EA	16	7	7	3	3	UV*
acc, imm	1+i(1,2)	4	4	3	2	1	1	UV

* = not pairable if there is a displacement and immediate

Example: or eax, ebx

OUT Output to port

operands	bytes	8088	186	286	386	486	Pentium	
imm8, al	2	14	9	3	10	16	12	NP
imm8, ax	2	14	9	3	10	16	12	NP
imm8, eax	2	-	-	-	10	16	12	NP
dx, al	1	12	7	3	11	16	12	NP
dx, ax	1	12	7	3	11	16	12	NP
dx, eax	1	-	-	-	11	16	12	NP

Protected Mode

operands	bytes	386	486	Pentium	
imm8, acc	2	4/24/24	11/31/29	9/26/24	NP
dx, acc	1	5/25/25	10/30/29	9/26/24	NP

cycles for: CPL <= IOPL / CPL > IOPL / V86

Example: out dx, al

OUTS/OUTSB/OUTSW/OUTSD Output string to port

variations	bytes	186	286	386	486	Pentium	
outsb	1	14	5	14	17	13	NP
outsw	1	14	5	14	17	13	NP
outsd	1	-	-	14	17	13	NP

Protected Mode

bytes	386	486	Pentium	
1	8/28/28	10/32/30	10/27/25	NP

cycles for: CPL <= IOPL / CPL > IOPL / V86

Example: rep outsw

POP Pop a word/dword from the stack

operand	bytes	8088	186	286	386	486	Pentium	
reg	1	12	10	5	4	1	1	UV
mem	2+d(0-2)	25+EA	20	5	5	6	3	NP
seg	1	12	8	5	7	3	3	NP
FS/GS	2	-	-	-	7	3	3	NP

Protected Mode

operand	bytes	286	386	486	Pentium	
CS/DS/ES	1	20	21	9	3-12	NP
SS	1	20	21	9	8-17	NP
FS/GS	2	-	21	9	3-12	NP

Example: pop eax

POPA/POPAD Pop all (186+)/Pop all double (386+)

variations	bytes	186	286	386	486	Pentium	
popa	1	51	19	24	9	5	NP
popad	1	-	-	24	9	5	NP

popa = pop di, si, bp, sp, bx, dx, cx, ax
popad = pop edi, esi, ebp, esp, ebx, edx, ecx, eax
(sp and esp are discarded)

Example: popa

POPF/POPFD Pop flags/Pop flags double (386+)

variations	bytes	8088	186	286	386	486	Pentium	
popf	1	12	8	5	5	9	6	NP
popfd	1	-	-	-	5	9	6	NP

Protected Mode

	bytes	286	386	486	Pentium	
popf	1	5	5	6	4	NP
popfd	1	-	5	6	4	NP

Example: popf

PUSH push a word/dword to the stack

operand	bytes	8088	186	286	386	486	Pentium	
reg	1	15	10	3	2	1	1	UV
mem	2+d(0-2)	24+EA	16	5	5	4	2	NP
seg	1	14	9	3	2	3	1	NP
imm	1+i(1,2)	-	-	3	2	1	1	NP
FS/GS	2	-	-	-	2	3	1	NP

Example: push eax

PUSHA/PUSHAD Push all (186+)/Push all double (386+)

variations	bytes	186	286	386	486	Pentium	
pusha	1	36	17	18	11	5	NP
pushad	1	-	-	18	11	5	NP

pusha = push ax, cx, dx, bx, sp, bp, si, di,
pushad = push eax, ecx, edx, ebx, esp, ebp, esi, edi

Example: pusha

PUSHF/PUSHFD Push flags/Push flags double (386+)

variations	bytes	8088	186	286	386	486	Pentium	
pushf	1	14	9	3	4	4	9	NP
pushfd	1	-	-	-	4	4	9	NP

Protected Mode

	bytes	286	386	486	Pentium	
pushf	1	3	4	3	3	NP
pushfd	1	-	4	3	3	NP

Example: pushf

RCL Rotate bits left with CF

operands	bytes	8088	186	286	386	486	Pentium	
reg, 1	2	2	2	2	9	3	1	PU
mem, 1	2+d(0,2)	23+EA	15	7	10	4	3	PU
reg, cl	2	8+4n	5+n	5+n	9	8-30	7-24	NP
mem, cl	2+d(0,2)	28+EA+4n	17+n	8+n	10	9-31	9-26	NP
reg, imm	3	-	5+n	5+n	9	8-30	8-25	NP
mem, imm	3+d(0,2)	-	17+n	8+n	10	9-31	10-27	NP

Example: rcl eax, 16

RCR Rotate bits right with CF

operands	bytes	8088	186	286	386	486	Pentium	
reg, 1	2	2	2	2	9	3	1	PU
mem, 1	2+d(0,2)	23+EA	15	7	10	4	3	PU
reg, cl	2	8+4n	5+n	5+n	9	8-30	7-24	NP
mem, cl	2+d(0,2)	28+EA+4n	17+n	8+n	10	9-31	9-26	NP
reg, imm	3	-	5+n	5+n	9	8-30	8-25	NP
mem, imm	3+d(0,2)	-	17+n	8+n	10	9-31	10-27	NP

Example: rcr eax, 16

ROL Rotate bits left

operands	bytes	8088	186	286	386	486	Pentium	
reg, 1	2	2	2	2	3	3	1	PU
mem, 1	2+d(0,2)	23+EA	15	7	7	4	3	PU
reg, cl	2	8+4n	5+n	5+n	3	3	4	NP
mem, cl	2+d(0,2)	28+EA+4n	17+n	8+n	7	4	4	NP
reg, imm	3	-	5+n	5+n	3	2	1	PU
mem, imm	3+d(0,2)	-	17+n	8+n	7	4	3	PU*

* = not pairable if there is a displacement and immediate

Example: rol eax, 16

ROR Rotate bits right

operands	bytes	8088	186	286	386	486	Pentium
reg, 1	2	2	2	2	3	3	1 PU
mem, 1	2+d(0,2)	23+EA	15	7	7	4	3 PU
reg, cl	2	8+4n	5+n	5+n	3	3	4 NP
mem, cl	2+d(0,2)	28+EA+4n	17+n	8+n	7	4	4 NP
reg, imm	3	-	5+n	5+n	3	2	1 PU
mem, imm	3+d(0,2)	-	17+n	8+n	7	4	3 PU*

* = not pairable if there is a displacement and immediate

Example: ror eax, 16

RDMSR Read from model specific register (Pentium+)

bytes	Pentium
2	20-24 NP

Example: rdmsr

REP Repeat string operation

See: MOVS (rep movs) move block
See: STOS (rep stos) fill block

REPE Repeat while equal (or zero) string operation

See: CMPS (repe cmps) find non-matching memory items
See: CMPS (repe scas) find non-acc matching byte in memory

REPNE Repeat while not equal (or not zero) string operation

See: CMPS (repne cmps) find first matching memory items
See: SCAS (repne scas) find first matching memory item to acc

RET/RETN/RETF Return from procedure

variations/ operands	bytes	8088	186	286	386	486	Pentium
retn	1	20	16	11+m	10+m	5	2 NP
retn imm16	1+d(2)	24	18	11+m	10+m	5	3 NP
retf	1	34	22	15+m	18+m	13	4 NP
retf imm16	1+d(2)	33	25	15+m	18+m	14	4 NP

RET is coded by the assembler as near or far based on the procedure declaration and program model, as:

RETN (return near)
RETF (return far)

Example: ret

Protected Mode

variations/ operands	bytes	286	386	486	Pentium
retf	1	25+m/55	32+m/62	18/33	4-13/23 NP
retf imm16	1+d(2)	25+m/55	32+m/68	17/33	4-13/23 NP

cycles for: same privilege level/lower privilege level

RSM Resume from system management mode (Pentium+)

bytes	Pentium
2	83 NP

Example: rsm

SAL/SHL/SAR/SHR Shift bits

operands	bytes	8088	186	286	386	486	Pentium
reg, 1	2	2	2	2	3	3	1 PU
mem, 1	2+d(0,2)	23+EA	15	7	7	4	3 PU
reg, cl	2	8+4n	5+n	5+n	3	3	4 NP
mem, cl	2+d(0,2)	28+EA+4n	17+n	8+n	7	4	4 NP
reg, imm	3	-	5+n	5+n	3	2	1 PU
mem, imm	3+d(0,2)	-	17+n	8+n	7	4	3 PU*

* = not pairable if there is a displacement and immediate

sal = shift arithmetic left sar = shift arithmetic right
shl = shift left (same as sal) shr = shift right

Example: shl eax, 1

SAHF Store AH into flags

bytes	8088	186	286	386	486	Pentium
1	4	3	2	3	2	2 NP

Example: sahf

SBB Integer subtraction with borrow

operands	bytes	8088	186	286	386	486	Pentium
reg, reg	2	3	3	2	2	1	1 PU
mem, reg	2+d(0,2)	24+EA	10	7	7	3	3 PU
reg, mem	2+d(0,2)	13+EA	10	7	6	2	2 PU
reg, imm	2+i(1,2)	4	4	3	2	1	1 PU
mem, imm	2+d(0,2)+i(1,2)	23+EA	16	7	7	3	3 PU*
acc, imm	1+i(1,2)	4	4	3	2	1	1 PU

* = not pairable if there is a displacement and immediate

Example: sbb eax, ebx

SCAS/SCASB/SCASW/SCASD Scan string data

variations	bytes	8088	186	286	386	486	Pentium

scasb	1	19	15	7	7	6	4 NP
scasw	1	19	15	7	7	6	4 NP
scasd	1	-	-	-	7	6	4 NP
repX scasb	2	9+15n	5+15n	5+8n	5+8n	7+5n*	8+4n NP
repX scasw	2	9+19n	5+15n	5+8n	5+8n	7+5n*	8+4n NP
repX scasd	2	-	-	-	5+8n	7+5n*	8+4n NP

repX = repe or repz or repne or repnz

* = 5 if n=0
(n = count of bytes, words or dwords)

Example: repne scasb

SET Set byte to 1 on condition else set to 0 (386+)

operand	bytes	386	486	Pentium
r8	3	4	4/3	1/2 NP
mem8	3+d(0-2)	5	3/4	1/2 NP

Cycles are for: true/false

setCC = one of:

seta	setae	setb	setbe	setc	sete
setg	setge	setl	setle	setna	setnae
setnb	setnbe	setnc	setne	setng	setnge
setnl	setnle	setno	setnp	setns	setnz
seto	setp	setpe	setpo	sets	setz

Example: setne al

SGDT Store global descriptor table register (286+)

operand	bytes	286	386	486	Pentium
mem48	5	11	9	10	4 NP

Example: sgdt descriptor[ebx]

SIDT Store interrupt descriptor table register (286+)

operand	bytes	286	386	486	Pentium
mem48	5	12	9	10	4 NP

Example: sidt descriptor[ebx]

SHLD Double precision shift left (386+)

operands	bytes	386	486	Pentium
reg, reg, imm	4	3	2	4 NP
mem, reg, imm	4+d(0-2)	7	3	4 NP
reg, reg, cl	4	3	3	4 NP
mem, reg, cl	4+d(0-2)	7	4	5 NP

Example: shld eax, ebx, 16

SHRD Double precision shift right (386+)

operands	bytes	386	486	Pentium
reg, reg, imm	4	3	2	4 NP
mem, reg, imm	4+d(0-2)	7	3	4 NP
reg, reg, cl	4	3	3	4 NP
mem, reg, cl	4+d(0-2)	7	4	5 NP

Example: shrd eax, ebx, 16

SLDT Store local descriptor table register (286+)

operands	bytes	286	386	486	Pentium
r16	3	2	2	2	2 NP
mem16	3+d(0-2)	3	2	3	2 NP

Example: sldt ax

SMSW Store machine status word (286+)

operands	bytes	286	386	486	Pentium
r16	3	2	2	2	4 NP
mem16	3+d(0-2)	3	3	3	4 NP

Example: smsw ax

STC Set the carry flag

bytes	8088	186	286	386	486	Pentium
1	2	2	2	2	2	2 NP

Example: stc

STD Set direction flag (set to reverse string direction)

bytes	8088	186	286	386	486	Pentium
1	2	2	2	2	2	2 NP

Example: std

STI Set interrupt flag (enable)

bytes	8088	186	286	386	486	Pentium
1	2	2	2	3	5	7 NP

Example: sti

STOS/STOSB/STOSW/STOSD Store string data

variations	bytes	8088	186	286	386	486	Pentium
stosb	1	11	10	3	4	5	3 NP
stosw	1	15	10	3	4	5	3 NP
stosd	1	-	-	-	4	5	3 NP
rep stosb	2	9+10n	6+9n	4+3n	5+5n	7+4n*	3+n NP
rep stosw	2	9+14n	6+9n	4+3n	5+5n	7+4n*	3+n NP
rep stosd	2	-	-	-	5+5n	7+4n*	3+n NP

* = 5 if n=0, 13 if n=1

(n = count of bytes, words or dwords)

Example: rep stosd

STR Store task register (286+)

operand	bytes	286	386	486	Pentium	
r16	3	2	2	2	2	NP
mem16	3+d(0-2)	3	2	3	2	NP

Example: str bx

SUB Integer subtraction

operands	bytes	8088	186	286	386	486	Pentium	
reg, reg	2	3	3	2	2	1	1	UV
mem, reg	2+d(0,2)	24+EA	10	7	7	3	3	UV
reg, mem	2+d(0,2)	13+EA	10	7	6	2	2	UV
reg, imm	2+i(1,2)	4	4	3	2	1	1	UV
mem, imm	2+d(0,2) +i(1,2)	23+EA	16	7	7	3	3	UV*
acc, imm	1+i(1,2)	4	4	3	2	1	1	UV

* = not pairable if there is a displacement and immediate

Example: sub eax, ebx

TEST Logical compare

operands	bytes	8088	186	286	386	486	Pentium	
reg, reg	2	3	3	2	2	1	1	UV
mem, reg	2+d(0,2)	13+EA	10	6	5	2	2	UV
reg, mem	2+d(0,2)	13+EA	10	6	5	2	2	UV
reg, imm	2+i(1,2)	5	4	3	2	1	1	UV
mem, imm	2+d(0,2) +i(1,2)	11+EA	10	6	5	2	2	UV*
acc, imm	1+i(1,2)	4	4	3	2	1	1	UV

* = not pairable if there is a displacement and immediate

Example: sub eax, ebx

VERR Verify a segment for reading (286+)

operand	bytes	286	386	486	Pentium	
r16	3	14	10	11	7	NP
mem16	3+d(0,2)	16	11	11	7	NP

Example: verr ax

VERW Verify a segment for writing (286+)

operand	bytes	286	386	486	Pentium	
r16	3	14	15	11	7	NP
mem16	3+d(0,2)	16	16	11	7	NP

Example: verr ax

WAIT Wait for co-processor

bytes	8088	186	286	386	486	Pentium
1	4	6	3	6	1-3	1 NP

Example: wait

WBINVD Write-back and invalidate data cache (486+)

bytes	486	Pentium
2	5	2000+ NP

Example: wbinvd

WRMSR Write to model specific register (PENTIUM+)

bytes	Pentium
2	30-45 NP

Example: wrmsr

XADD Exchange and add (486+)

operands	bytes	486	Pentium
reg, reg	3	3	3 NP
mem, reg	3+d(0-2)	4	4 NP

Example: xadd eax, ebx

XCHG Exchange register/memory with register

operands	bytes	8088	186	286	386	486	Pentium
reg, reg	2	4	4	3	3	3	3 NP
reg, mem	2+d(0-2)	25+EA	17	5	5	5	3 NP
mem, reg	2+d(0-2)	25+EA	17	5	5	5	3 NP
acc, reg	1	3	3	3	3	3	2 NP
reg, acc	1	3	3	3	3	3	2 NP

in above: acc = AX or EAX only

Example: xchg ax, dx

XLAT/XLATB Table look-up translation

bytes	8088	186	286	386	486	Pentium
1	11	11	5	5	4	4 NP

Example: xlat

XOR Logical exclusive or

operands	bytes	8088	186	286	386	486	Pentium
reg, reg	2	3	3	2	2	1	1 UV
mem, reg	2+d(0,2)	24+EA	10	7	7	3	3 UV

reg, mem	2+d(0,2)	13+EA	10	7	6	2	2	UV
reg, imm	2+i(1,2)	4	4	3	2	1	1	UV
mem, imm	2+d(0,2) +i(1,2)	23+EA	16	7	7	3	3	UV*
acc, imm	1+i(1,2)	4	4	3	2	1	1	UV

* = not pairable if there is a displacement and immediate

Example: xor eax, ebx