## 80x86 Integer Instruction Set Timings (8088 - Pentium)

Source: http://www.quantasm.com/opcode i.html

```
Legend:
General
      = AL, AX or EAX unless specified otherwise
acc
      = any general register
      = 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
     = 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
         Instruction formats, clock cycles and Pentium® Pairing info
AAA
       ASCII adjust after addition
                                       286
                                               386
               bytes
                       8088
                               186
                                                       486
                                                               Pentium
                        8
                                8
                                        3
                                                4
                                                        3
                                                                    NΡ
                                                                3
                1
        Example:
                       aaa
       ASCII adjust AX before division (second byte is multiplier)
AAD
               bytes 8088
                               186
                                       286
                                               386
                                                       486
                                                               Pentium
                       60
                               15
                                               19
                                       14
                                                       14
```

Example: aad

AAM	ASCII	adjust AX bytes 2	<b>after</b> 8088 83	multiply 186 19	(second 286 16	<b>byte</b> 386 17	is	divisor) 486 15	Pent 18	ium NP
	Examp		aam							
AAS	ASCII	adjust AL	after	subtracti	ion					
		bytes	8088	186	286	386		486	Pent	ium
		1	8	7	3	4		3	3	NP
	Examp	le:	aas							
ADC	Integ	er add wit	h carry	У						
	operands	bytes	8088	186	286	386		486	Pent	ium
	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)	23+EA	16	7	7		3	3	PU*
		+i(1,2)						_	_	
	acc, imm	1+1(1,2) ot pairable	4 = if +1	4	3	2 mant	and	1	1	PU
	Examp		adc	eax, el		EIIICIIC	anu	IIIIIIEGIA	LE	
	-			•						
ADD		er addition								
	operands	bytes	8088	186	286	386		486	Pent	
	reg, reg	2	3	3	2	2		1	1	UV
		2+d(0,2)			7	7		3	3	UV
		2+d(0,2)			7	6		2 1	2	UV
		2+i(1,2)		4 16	3 7	2 7		3	1 3	UV*
	mem, Imm	2+d(0,2) +i(1,2)	23+EA	10	/	/		3	3	0 0 "
	acc, imm		4	4	3	2		1	1	UV
	* = n	ot pairable		nere is a	displace	ement	and	immedia	te	
	Examp	le:	add	eax, ek	OX					
AND	Logic	al AND								
	operands	bytes	8088	186	286	386		486	Pent	ium
	req, req	2	3	3	2	2		1	1	UV
	mem, reg	2+d(0,2)	24+EA	10	7	7		3	3	UV
	reg, mem		13+EA		7	6		2	2	UV
	reg, imm	2+i(1,2)	4	4	3	2		1	1	UV
	mem, imm	2+d(0,2)	23+EA	16	7	7		3	3	UV*
		+i(1,2)	4		2	0		-		
	acc, imm		4 - :	4	3	2		1	1	UV
	r = n Examp	ot pairable		eax, el		ement	and	Illilledia	ce	
	ьхашр	īe.	and	eax, ex	)X					
ARPI	L Adjus	t RPL field	d of se	elector (2	286+)					
	operands				286	386			Pent	ium
	reg, reg				10	20		9	7	NP
		2+d(0-2)			11	21		9	7	NP
	Examp	le:	arpl	ax, bx						
BOU	ND Check	array inde	ex aga:	inst bound	ds (186+	)				
		s bytes		186	286	386		486	Pent	ium
	reg, me	<del>-</del>		35	13	10		7	8	NP
	•	le:	bound		ray					
D.C.=	<b>-</b> 1.	<b>6</b>	a (20c							
BSF		can forward bytes	ı (386-	+ )		386		486	Pent	i 11m
	operands	nytes				300		100	rent	± uill

r16, r16 3 r32, r32 3 r16, m16 3+d(0,1,2) r32, m32 3+d(0,1,2,4) Example: bsf	eax, [esi]	10+3n 10+3n 10+3n 10+3n	6-42 7-43	6-42 NP 6-35 NP
BSR Bit scan reverse (386+) operands bytes r16, r16 3 r32, r32 3 r16, m16 3+d(0,1,2) r32, m32 3+d(0,1,2,4) Example: bsr			6-103 7-104 6-103	7-71 NP 7-40 NP
BSWAP Byte swap (486+) operand bytes r32 2 Example: bswap	eax		486 1	Pentium 1 NP
BIT Bit test (386+) operands bytes reg, reg 3 mem, reg 3+d(0,1,2,4) reg, imm8 3+i(1) mem, imm8 3+d(0,1,2,4)+i(1 Example: bt	) eax, 4	386 3 12 3 6	486 3 8 3 3	Pentium 4 NP 9 NP 4 NP 4 NP
operands bytes reg, reg 3 mem, reg 3+d(0,1,2,4) reg, imm8 3+d(0,1,2,4)+i(1 Example: btc		386 6 13 6 8	486 6 13 6 8	Pentium 7 NP 13 NP 7 NP 8 NP
BTR Bit test and reset (386 operands bytes reg, reg 3 mem, reg 3+d(0,1,2,4) reg, imm8 3+d(0,1,2,4)+i(1 Example: btr		386 6 13 6 8	486 6 13 6 8	Pentium 7 NP 13 NP 7 NP 8 NP
BTS Bit test and set (386+)  operands bytes  reg, reg 3  mem, reg 3+d(0,1,2,4)  reg, imm8 3+d(0,1,2,4)+i(1  Example: bts		386 6 13 6 8	486 6 13 6 8	Pentium 7 NP 13 NP 7 NP 8 NP
CALL         Call subroutine           operand         bytes         8088           near         3         23           reg         2         20           mem16         2+d(0-2)         29+EA           far         5         36           mem32         2+d(0-2)         53+EA           pr         operand         bytes           far         5	186 286 14 7+m 13 7+m 19 11+m 23 13+m 38 16+m rotected Mode 286 26+m	386 7+m 7+m 10+m 17+m 22+m 386 34+m	486 3 5 5 18 17 486 20	Pentium 1 PV 2 NP 2 NP 4 NP 4 NP 4 NP Pentium 4-13 NP

me		2+d(0-2) rcles not	shown fo	or calls my_func		38+m call and	20 task		NP
CDI	<b>a</b>			(37	3.75)				
CBW	Conver	t byte to			-	206	106	<b>.</b>	
		bytes 1	8088 2	186 2	286 2	386 3	486 3	Pent 3	ıum NP
	Exampl	<del>-</del>	cbw	2	2	3	3	3	NP
	тхашрт	.e •	CDW						
CWDE	Conver	t word to	dword	(386+) (	'AY> F	EAX)			
CNDE	COHVEL	bytes	awora	(3001)	, AA.	386	486	Pent	iıım
		1				3	3	3	NP
	Exampl		cwde			3	3	3	111
CWD	Conver	t word to	double	(AX>	DX:AX)				
		bytes	8088	186	286	386	486	Pent	ium
		1	5	4	2	2	3	2	NP
	Exampl	.e:	cwd						
CDQ	Conver	t double	to quad	(EAX -	> EDX:				
		bytes				386	486	Pent	
		1	,			2	3	2	NP
	Exampl	.e:	cdq						
CT C	C1	tho ======	£1						
CLC	Clear	the carry	8088	106	286	386	106	Pent	ium
		bytes 1	2	186 2	2	2	486 2	2	NP
	Exampl		clc	2	2	2	4	۷	NP
CLD	_	the direc		ad (set t	o forwar	rd direct	ion)		
СПР	CICAL	bytes	8088	186	286	386	486	Pent	iıım
		1	2	2	2	2	2	2	NP
	Exampl	<del>-</del>	cld	2	_	_	-	-	111
CLI	-	the inter		ag (disak	ole inter	rrupts)			
		bytes	8088	186	286	386	486	Pent	ium
		1	2	2	3	3	5	7	NP
	Exampl	.e:	cli						
CLTS	Clear	task swit	ched fla	ag in CRO	(286+)				
		bytes			286	386	486	Pent	ium
		2			2	5	7	10	NP
	Exampl		clts						
CMC	Comple	ment carr							
		bytes	8088	186	286	386	486	Pent	
		1	2	2	2	2	2	2	NP
	Exampl	.e <b>:</b>	CMC						
CMP	Compan	e two ope	randa						
	rands	bytes	8088	186	286	386	486	Pent	iıım
_	, reg	2	3	3	2	2	1	1	UV
	, reg	2+d(0,2)	13+EA	10	7	5	2	2	UV
		2+d(0,2)	13+EA	10	6	6	2	2	UV
_		2+i(1,2)	4	4	3	2	1	1	UV
_		2+d(0,2)	14+EA	10	6	5	2	2	UV*
		+i(1,2)							
acc	, imm	1+i(1,2)	4	4	3	2	1	1	UV
	* = no	t pairabl	e if the	ere is a	displace	ement and	immed	liate	
	Exampl	.e:	cmb	eax, 3					
		SW/CMPSD		are strir					
varia		bytes	8808	186	286	386	486	Pent	
cmpsb									
		1	30	22	8	10	8	5	NP
cmpsw		1 1	30 -	22 -	8 –	10 10	8 8	5 5	NP NP

cmpsd		1	_	_	_	10	8	5
repX o	cmpsb	2	9+30n	5+22n	5+9n	5+9n	7+7n*	9+4n
repX o	cmpsw	2	9+30n	5+22n	5+9n	5+9n	7+7n*	9+4n
repX o	cmpsd	2	_	-	_	5+9n	7+7n*	9+4n
rep:			repne or	repnz				
	* : 5 if Example:		repne d	cmpsb				
CMPXCHG	_		Exchange		)			
	operands		bytes		-		486	Penti
	reg, reg	3	3				6	5
	mem, reg	•	3+d(0-2)				7-10	6
	Example:		cmpxchg	g ebx,	edx			
CMPXCHG	<b>8B Compa</b> operar		Exchange bytes	e 8 byte	es (Pent:	ium+)		Penti
	mem, r		3+d(0-2)	١				10
	Example:		cmpxch		vl edv			10
					A], edx			
CPUID	CPU ider	<b>ntifica</b> bytes	tion (Per	ntium+)				Penti
		2						14
	Example:		cpuid					
DAA	Decimal	_	AL after			206	406	D + -
		bytes 1	8088 4	186 4	286 3	386 4	486 2	Penti
	Example:		daa	4	3	4	2	3
DAS	Dogimal	ad ingt	AL after	c gubtr	action			
DAS	Decimal	bytes	8088	186	286	386	486	Penti
		1	4	4	3	4	2	3
	Example:	_	das	1	J	1	2	J
DEC	Decremen	nt						
_	rand	bytes	8808	186	286	386	486	Penti
r8		2	3	3	2	2	1	1
r16		1	3	3	2	2	1	1
r32		1	3	3	2	2	1	1
mem		-d(0,2)		15	7	6	3	3
	Example:		dec	eax				
DIV	Unsigned			106	206	206	406	D t -
opei r8	rand	bytes 2	8088 80-90	186 29	286 14	386	486 16	Pent: 17
					22	14 22		
r16			144-162	38			24	25
r32		2	- 06 06 17	- \ 2E	- 1 7	38 17	40	41
mem: mem:		(0-2)	86-96+EA		17 25	17	16	17
mem. mem:			150-168+E	SA 44 -	25 -	25 41	24 40	25 41
mem.	5∠ ∠+α(	(0-2)	_	-	_	41	40	4⊥
impl: divi		perand	quo	otient	remain	der		
AX		/te	=	AL	AH			
DX:A		ord	=	AX	DX			
EDX:	•	vord	=	EAX	EDX			
	Example:		div	ebx				
ENTER			me for pr					_
	perands		8808	186	286	386	486	Penti
ir	mm16, 0	3	_	15	11	10	14	11

```
- 25 15 12 17 15 NP
- 22+16n 12+4n 15+4n 17+3i 15+2i NP
    imm16, 1 4
    imm16, imm8 4
                 n = imm8-1; i = imm8
      Example:
                 enter 1, 0
     Escape
escape opcodes D8 - DF are used by floating point instructions
      Halt
                 8808
                       186
                              286
                                    386
                                          486
            bytes
                                               Pentium
                  2
                        2
            1
                              2
                                    5
                                           4
                                                4 NP
      Example:
                  hlt
IDIV Signed divide
   operand bytes 8088
                         186 286 386
                                          486
                                                Pentium

    44-52
    17
    19

    53-61
    25
    27

            2 101-112
   r8
                                          19
                                                22 NP
   r16
            2
                165-184
                                         27
                                                30 NP
   46 NP
                                              22 NP
30 NP
                                               46 NP
                  quotient remainder
  implied operand
  dividend
  AX / byte
                 =
                       AL
                            AH
  DX:AX / word
EDX:EAX / dword
                 =
                             DX
                       AX
                 =
                       EAX
                             EDX
     Example:
                 idiv ebx
IMUL
     Signed multiply
                  Accumulator Multiplies
            bytes 8088 186 286 386
   operand
                                          486
                                                Pentium
                        25-28 13
                                   9-14
                 80-98
   r8
           2
                                          13-18
                                                11 NP
   11
                                                    NP
                                                10 NP
                                                11 NP
                                                11 NP
                                                10 NP
   implied
            operand
                      result
  multiplicand (multiplier)
     AL * byte = AX
         * word = DX:AX
* dword = EDX:EAX
      ΑX
      EAX
                 imul ebx
      Example:
                  2 and 3 operand Multiplies
              bytes 186 286 386 486
                                                Pentium
   operands
           2+i(1,2)
2+i(1,2)
                     -
-
   r16, imm
                           21 9-14/9-22 13-18/13-26 10 NP
   r32, imm
                           - 9-38
                                                10
                                      13-42
   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) 25/32 24 12-17/12-25 13-18/13-26 10 NP
              +i(1,2)
                              12-41
   r32,m32,imm
             2+d(0-2)+i(1,2)
                                        13-42
                                                10 NP
   9-22
                                      13-18/13-26 10 NP
                                      13-42
                                                10 NP
                                      13-18/13-26 10 NP
```

ESC

HLT

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

13-42

10 NP

dest, src1, src2

INVLPG Invalidate TLB entry (486+)

		c, SICI,		00	h 10				
	Example	<b>2</b> •	imul	eax, e	ebx, 10				
		_							
IN	_	from port							
ope	rands	bytes	8808	186	286	386	486	Pent	ium
al,	imm8	2	14	10	5	12	14	7	NP
ax,	imm8	2	14	10	5	12	14	7	NP
eax	i, imm8	2	_	_	_	12	14	7	NP
	dx	1	12	8	5	13	14	7	NP
		1	12	8	5	13	14	7	NP
	dx			0	- -	_			
eax	, dx	1	_			13	14	7	NP
			I	Protected	d mode				
oper	ands	bytes				386	486	Pent	ium
acc,	imm	2			6/	/26/26 9	9/29/27	4/21/	19 NP
acc,	dx	1			7/	/27/27 8	3/28/27	4/21/	19 NP
		cycl	es for:	CPL <=	IOPL / CE	PL > IOPI	_ / V86		
	Example		in	al, dx					
	211011111111111111111111111111111111111	-		GI , GI	-				
INC	Increme								
		_	0000	100	006	206	406		
	rand		8088	186	286	386	486	Pent	
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	2+d(0,2)	23+EA	15	7	6	3	3	UV
	Example	<b>:</b>	inc	ebx					
	Liampic	-	1110	0.011					
TNG /TNG	D/TNGW/	INCD	T						
	B/INSW/I				to strin		406		
	iations	-	8088	186	286	386	486	Pent	
ins	d	1	-	14	5	15	17	9	NP
ins	W	1	_	14	5	15	17	9	NP
ins	d	1	-	_	-	15	17	9	NP
			Ι	Protected	d Mode				
		bytes				386	486	Pent	ium
		1			9 /	/29/29 10			
			og for	· CDI /-	IOPL / CE			0/21/2	2 111
	Erromple				TOPH / CF	- 10F1	1 / VOO		
	Example	<b>:</b> •	rep ir	ISD					
			_						
INT		nterrupt :	_						
0	perands	bytes	8808	186	286	386	486	Pent	ium
	3	1	72	45	23+m	33	26	13	NP
	imm8	2	71	47	23+m	37	30	16	NP
				Protected					
		bytes		186		386	486	Pent	iıım
		1	-		(40-78)+n				
					(40-70)+11	11 39-99	44-/1	27-02	INP
	Example	<b>:</b> •	int	21h					
		_	-						
INTO	Call in	nterrupt :							
		bytes	8808	186	286	386	486	Pent	ium
		1	4/73	4/48	3/24+m	3/35	3/28	4/13	NP
			I	Protected	d mode				
		bytes				386	486	Pent	ium
		1			(40-78)+n				
		-	Тасі	e awitah	clocks no		/ _	2, 50	111
	Esca-7 :	· ·		7 DWILCII	CIOCKP IIC	C SHOWII			
	Example	<b>:</b> •	into						
INVD	Invalid	date data							
		bytes	8808	186	286	386	486	Pent	ium
		2	-	_		-	4	15	NP
	Example		invd						
		_							

0	mem32	_						486 12	Pent 25	ium NP
	Examp]	le:	invlpg	[eax]	]					
IRET	Return	n from int	errupt							
11111	RCCULI	bytes	8088	186	28	6	386	486	Pent	ium
		1	44	28	17	+m	22	15	8-27	NP
		T	ask swit	ch clo	ocks n	ot sh	ıown			
	Exampl	le:	iret							
TDEMD	22 644	t return f			(206)	`				
IRETD	32-DI	bytes	TOIL THE	errupt	(300+	,	386	486	Pent	iım
		1					22		10-27	
		T	ask swit	ch clo	ocks n	ot sh	lown			
	Examp]	le:	iretd							
	_									
Jcc	_	on conditi		100	20	_	206	406	Dant	
ope nea	rand	bytes 2	8088 4/16	186 4/13	28	o 7+m	386 3/7+m	486 1/3	Pent 1	PV
	r16	3	-	<del>1</del> /13	<i>3/</i>	/ <b>+</b> 111	3/7+m	· ·	1	PV
iica	110	3	cycles i		no ium	p/ium	-	1/3	_	ı v
		cc	ndition							
ja		if above			jnbe	jump	if not	below o	r equa	1
jae	jump i	if above c	r equal		jnb	jump	if not	below		
jb	-	if below			jnae		if not		r equa	1
jbe		if below o			jna		if not			
jg		if greater		_	jnle		if not		equal	
jge		if greater	or equa	al	jnl		if not			_
jl ::		if less	,		jnge		if not			[ual
jle 		if less or	equal		jng 		if not			
je jne		if equal if not equ	<b>~</b> 1		jz jnz		o if zero o if not			
jc		if carry	.aı		jnc		if not			
js		if sign			jns		if not	_		
jnp		if no pari	ty (odd	)	jpo		if pari			
jo		if overflo		•	jno		if not		W	
ąį		if parity			jpe		if pari			
	Examp]	le:	jne	not_e	equal					
JCXZ/JE	_	Jump if C			0.0	_	206	406		
-	erand st	bytes 2	8088	186 5/16	28		386	486	Pent	
	st	2	6/18	5/10	4/	8+m	5/9+m 5/9+m	5/8 5/8	5/6 5/6	NP NP
ue	SC	2	cycles i	- for: r	– ישוד סי	n/ilim	•	5/6	5/0	NP
	Exampl	le:	jcxz		s_zero		ıP			
	-			_	_					
JMP	Uncond	ditional j	ump							
ope	rand	bytes	8808	186	28	6	386	486	Pent	ium
sho		2	15	13	7+1		7+m	3	1	PV
nea		3	15	13	7+1		7+m	3	1	PV
far		5	15	13	11+1		12+m	17	3	NP
r16 mem		2 2+d(0,2)	11 18+F7	11 17	7+ı 11+ı		7+m 10+m	5 5	2 2	NP NP
mem		2+d(0,2) 2+d(4)	18+EA 24+EA	26	15+1		10+m 12+m	13	4	NP NP
r32		2+4(4)	∠¬⊤ĽA		15+	ııı	7+m	5	2	NP NP
mem		2+d(0,2)	_	_	_		10+m	5	2	NP
mem		2+d(0,2) 2+d(6)	_	_	_		10+m	13	4	NP
mem		cycles for	jumps t	hronat		gate			1	747
	Examp		jmp		et_add:					
	-			5	_					

LAHF	Load	flags into	АН					
		bytes 1	8088 4	186 2	286 2	386 2	486 3	Pentium 2 NP
	Exam	ple:	4 lahf	2	2	2	3	Z NP
	_							
LAR	<b>Load</b> rands	access rigitation bytes	hts byte	(286+)	286	386	486	Pentium
-	, r16	-			14	15	11	8 NP
	, r32					15	11	8 NP
	, m16				16	16	11	8 NP
	, m32		-		-	16	11	8 NP
	Exam	ple:	lar	eax, eb	X			
LDS	Load	far pointe	r					
ope	rands	bytes	8808	186		386		Pentium
reg	, mem	2+d(2)	24+EA		7	7	6	4 NP
	Exam	ole:	las	si, ptr				
LES	Load	far pointe	r					
		bytes				386		Pentium
reg	-	2+d(2)				7	6	4 NP
	Exam	ole:	les	ai, ptr	_2			
LFS	Load	far pointe	r (386+)					
		bytes				386	486	Pentium
reg	, mem	3+d(2,4)				7	6	4 NP
	Exam	ole:	lis	sı, ptr	_3			
LGS	Load	far pointe	r (386+)					
ope	rands	bytes				386	486	Pentium
		3+d(2,4)				7	6	4 NP
	Exam	ole:	Igs	sı, ptr	_4			
LSS	Load	stack segm	ent and	offset				
		bytes				386		Pentium
reg		3+d(2,4)		hn nta	Е	7	6	4 NP
	Exalli	ple:	155	pp, ptr	_၁			
		effective a						
	rands			186	286	386	486	Pentium
	, mem	, ,	2+EA -	6 -	3	2	1-2 1-2	1 UV 1 UV
132	Exam		lea		ax+ebx*2	_	1-2	1 00
				, -				
LEAVE	High	level proc	edure ex			206	406	D
		bytes 1		186 8	286 5	386 4	486 5	Pentium 3 NP
	Exam	<del>-</del>	leave	O	5	1	J	5 NI
LGDT		global des	criptor (	table re	_		100	Dan + 1
-	rand m48	bytes 5			286 11	386 11	486 11	Pentium 6 NP
ilic	Exam	-	lgdt	descrip	tor[ebx]			0 111
	•							
LIDT		interrupt	descripto	or table			106	Dow to t
_	rand m48	bytes 5			286 12	386 11	486 11	Pentium 6 NP
ille	Exam	-	lidt	descrip	tor[ebx]	<b></b>	<b></b>	O INF
	•	-						
LLDT		local desc	riptor ta	able reg			100	Dorti
ope	rand	bytes			286	386	486	Pentium

r16 mem16	3 3+d(0-2)			17 19	20 24	11 11	9 9	NP NP
Exam	ple:	lldt	ax					
LMSW Load	machine st	atus wor	d (286+)					
operand	bytes			286	386	486	Pent	
r16	3			3	10	13	8	NP
mem16	3+d(0-2)	lmsw		6	13	13	8	NP
Exam	bre.	IIISW	ax					
LOCK Lock	bus on nex		_					
	bytes	8088	186	286	386		Pent	
/27   1	1	2	2	0	0	1	1	NP
	g always is	locked			ecillea	or not)		
Exam	bie.	TOCK	mov	mem, 1				
LODS/LODSB/L	ODSW/LODSD	Load	string o	perand				
variatio	ns bytes	8808	186	286	386	486	Pent	ium
lodsb	1	16	10	5	5	5	2	NP
lodsw	1	16	10	5	5	5	2	NP
lodsd	1	-	_	-	5	5	2	NP
Exam	ple:	lodsb						
LOOP Loop	control wi	th CX co	unter					
operan	d bytes	8088	186	286	386	486	Pent	ium
short	2	5/17	5/15	4/8+m	11+m	6/7	5/6	NP
loopw short								
loopd short								
Exam	ple:	loop	loop_sta	art				
LOOPE/LOOPZ	Loop while	e equal	(or zero	)				
operan	=	8088	186	<b>,</b> 286	386	486	Pent	ium
short	2	6/18	5/16	4/8	11+m	6/9	7/8	NP
loopew short	(uses CX	in 32-bi	t mode)					
loopzw short	(uses CX	in 32-bi	t mode)					
looped short								
loopzd short			-					
Exam	ple:	loope	loop_sta	art				
LOOPNE/LOOPN	Z Loop whi	le not e	gual (or	not zero	o)			
	and bytes	8088	186	286	386	486	Pent	ium
shor	t 2	5/19	5/16	4/8	11+m	6/9	7/8	NP
loopnew shor	t (uses CX	in 32-b	it mode)					
loopnzw shor			-					
loopned shor								
loopnzd shor								
Exam	ple:	loopne	loop_sta	art				
LSL Load	segment li	mit (286	+)					
operands	bytes			286	386	486	Pent	ium
r16, r16				14	20/25	10	8	NP
r32, r32				-	20/25	10	8	
r16, m16				16	21/26	10	8	
r32, m32			,	-	21/26	10	8	
Exam	bīe:	lsl	eax, eb	X				
LTR Load	task regis	ter (286	+)					
operand	bytes			286	386	486	Pent	ium
r16	3			17	23	20	10	NP
mem16	3+d(0,2)			19	27	20	10	
Exam	ple:	ltr	ax					

	Move d	lata							
	operands	bytes	8808	186	286	386	486	Pent	ium
	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
	•	3	14	9	3	2	1	1	UV
	mem, acc	s ot pairabl		-	_	_	_	_	UV
	Exampl		mov	eax, e		silicite att	a Illilicai		
			Segme	nt Regis Real Mo	ter Moves	5			
	operands	bytes	8088	186	286	386	486	Pent	ium
	seg, r16	2	2	2	2	2	3	2-11	NP
	seg, m16	2+d(0,2)		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
	Exampl	Le:	mov	ds, ax					
	-				Differenc	ces			
	operands	bytes			286	386	486	Pent	iıım
	_	2			17	18	9	2-11*	
	seg, r16								
	seg, m16	2+d(0,2)		1 0 1 5	19	19	9	3-12*	NP
					ew descri			SS	
			E to/fr	om speci	al regist				
	operands	bytes				386	486	Pent	ium
	r32, cr32	3				6	4	4	NP
	cr32, r32	3				4/10*	4/16*	12/22	* NP
	r32, dr32	3				14/22*	· ·	2/12	
	dr32, r32	3				16/22*		11/12	
						•		•	
	70 ( ) + 70 ( )					12	3/4*	_	NP
	r32, tr32	3							
	tr32, tr32	3	_			12	4/6*	-	NP
	tr32, r32	3 * = cycle					-	-	NP
	tr32, r32 Exampl	3 * = cycle Le:	mov	cr0, e	ax	al regis	ter	-	ΝP
MOV	tr32, r32  Exampl  S/MOVSB/MOV	* = cycle Le: //SW/MOVSD	mov <b>Move</b>	cr0, e	ax <b>om strin</b> g	al regis g to str	ing		
<b>M</b> OV:	tr32, r32 Exampl	3 * = cycle le: /SW/MOVSD	mov <b>Move</b> 8088	cr0, e	ax <b>om strin</b> g 286	al regis y to str 386	ing 486	- Pent	
MOV	tr32, r32  Exampl  S/MOVSB/MOV	* = cycle Le: //SW/MOVSD	mov <b>Move</b>	cr0, e	ax <b>om strin</b> g	al regis g to str	ing		
<b>M</b> OV:	tr32, r32  Exampl  S/MOVSB/MOV  variations	3 * = cycle Le: 7SW/MOVSD s bytes	mov <b>Move</b> 8088	cr0, e	ax <b>om strin</b> g 286	al regis y to str 386	ing 486	Pent	ium
MOV	Exampl  S/MOVSB/MOV  variations  movsb	3 * = cycle Le: 7SW/MOVSD s bytes 1	Move 8088 18	cr0, e.  data fro	om string 286 5	al regis y to str 386 7	ing 486 7	Pent. 4	ium NP
<b>M</b> OV:	Exampl  S/MOVSB/MOV  variations  movsb  movsw  movsd	3 * = cycle Le: /SW/MOVSD s bytes 1 1 1	Move 8088 18 26	cr0, e.  data from 186 9 9	ax om string 286 5 5	to str 386 7 7	ing 486 7 7 7	Pent 4 4 4	ium NP NP NP
MOV:	Exampl  S/MOVSB/MOV  variations  movsb  movsw  movsd  rep movsb	3 * = cycle le:  /SW/MOVSD s bytes 1 1 1 2	Move 8088 18 26 - 9+17n	cr0, e.  data from 186 9 9 - 8+8n	ax  om string  286  5  -  5+4n	to str 386 7 7 7 7+4n	ing 486 7 7 7 12+3n*	Pent. 4 4 4 3+n	ium NP NP NP
MOV	Exampl  S/MOVSB/MOV  variations  movsb  movsw  movsd  rep movsw  rep movsw	3 * = cycle le:  /SW/MOVSD s bytes 1 1 2 2	Move 8088 18 26	cr0, e.  data from 186 9 9	om string 286 5 5 - 5+4n 5+4n	to str 386 7 7 7 7+4n 7+4n	ing 486 7 7 7 12+3n* 12+3n*	Pent. 4 4 4 3+n 3+n	ium NP NP NP NP
MOV	Exampl  S/MOVSB/MOV  variations  movsb  movsw  movsd  rep movsb	3 * = cycle le:  /SW/MOVSD s bytes 1 1 1 2	Move 8088 18 26 - 9+17n 9+25n	cr0, e.  data fr  186  9  -  8+8n  8+8n -	om string 286 5 5 - 5+4n 5+4n	to str 386 7 7 7 7+4n 7+4n 7+4n	ing 486 7 7 7 12+3n*	Pent. 4 4 4 3+n 3+n	ium NP NP NP NP
<b>M</b> OV:	Exampl  S/MOVSB/MOV  variations  movsb  movsw  movsd  rep movsw  rep movsw	<pre>3 * = cycle le: /SW/MOVSD s bytes 1 1 2 2 2 2</pre>	Move 8088 18 26 - 9+17n 9+25n - * = 5	cr0, e.  data from 186 9 9 - 8+8n 8+8n - if n=0,	om string 286 5 5 - 5+4n 5+4n - 13 if n=1	to str 386 7 7 7 7+4n 7+4n 7+4n	ing 486 7 7 7 12+3n* 12+3n*	Pent. 4 4 4 3+n 3+n	ium NP NP NP NP
MOV	Exampl  S/MOVSB/MOV  Variations  movsb  movsw  movsd  rep movsb  rep movsw  rep movsw  rep movsw	* = cycle Le:  /SW/MOVSD s bytes 1 1 2 2 2 (n = c	Move 8088 18 26 - 9+17n 9+25n - * = 5 ount of	cr0, e  data from 186 9 9 - 8+8n 8+8n - if n=0, bytes,	om string 286 5 5 - 5+4n 5+4n	to str 386 7 7 7 7+4n 7+4n 7+4n	ing 486 7 7 7 12+3n* 12+3n*	Pent. 4 4 4 3+n 3+n	ium NP NP NP NP
MOV	Exampl  S/MOVSB/MOV  variations  movsb  movsw  movsd  rep movsw  rep movsw	* = cycle Le:  /SW/MOVSD s bytes 1 1 2 2 2 (n = c	Move 8088 18 26 - 9+17n 9+25n - * = 5	cr0, e  data from 186 9 9 - 8+8n 8+8n - if n=0, bytes,	om string 286 5 5 - 5+4n 5+4n - 13 if n=1	to str 386 7 7 7 7+4n 7+4n 7+4n	ing 486 7 7 7 12+3n* 12+3n*	Pent. 4 4 4 3+n 3+n	ium NP NP NP NP
	Example S/MOVSB/MOV variations movsb movsd rep movsb rep movsd rep movsd SX Move w	<pre>3 * = cycle te: /SW/MOVSD s bytes 1 1 2 2 2 (n = content with sign-</pre>	Move 8088 18 26 - 9+17n 9+25n - * = 5 ount of rep mo	cr0, e.  data from 186 9 9 - 8+8n 8+8n - if n=0, bytes, vsb	om string 286 5 5 - 5+4n 5+4n - 13 if n=1	to str 386 7 7 7+4n 7+4n 7+4n dwords)	ing 486 7 7 7 12+3n* 12+3n*	Pent. 4 4 4 3+n 3+n 3+n	ium NP NP NP NP NP
	Example S/MOVSB/MOV variations movsb movsd rep movsb rep movsd rep movsd SExample Example SX Move voperands	* = cycle Le:  /SW/MOVSD s bytes 1 1 2 2 2 (n = c) Le:  with sign- s bytes	Move 8088 18 26 - 9+17n 9+25n - * = 5 ount of rep mo	cr0, e.  data from 186 9 9 - 8+8n 8+8n - if n=0, bytes, vsb	om string 286 5 5 - 5+4n 5+4n - 13 if n=1	to str 386 7 7 7+4n 7+4n 7+4n dwords)	ing 486 7 7 7 12+3n* 12+3n* 12+3n*	Pent. 4 4 4 3+n 3+n 3+n	ium NP NP NP NP NP
	Example S/MOVSB/MOV variations movsb movsd rep movsb rep movsd rep movsd SX Move w	* = cycle Le:  /SW/MOVSD s bytes 1 1 2 2 2 (n = c) Le:  with sign- s bytes	Move 8088 18 26 - 9+17n 9+25n - * = 5 ount of rep mo	cr0, e.  data from 186 9 9 - 8+8n 8+8n - if n=0, bytes, vsb	om string 286 5 5 - 5+4n 5+4n - 13 if n=1	to str 386 7 7 7+4n 7+4n 7+4n dwords)	ing 486 7 7 12+3n* 12+3n* 12+3n*	Pent. 4 4 4 3+n 3+n 3+n	ium NP NP NP NP NP
	Example S/MOVSB/MOV variations movsb movsd rep movsb rep movsd rep movsd SExample Example SX Move voperands	<pre>3 * = cycle Le: /SW/MOVSD s bytes 1 1 2 2 2 (n = c) Le: //SW/MOVSD s bytes 3</pre>	Move 8088 18 26 - 9+17n 9+25n - * = 5 ount of rep mo	cr0, e.  data from 186 9 9 - 8+8n 8+8n - if n=0, bytes, vsb	om string 286 5 5 - 5+4n 5+4n - 13 if n=1	to str 386 7 7 7+4n 7+4n 7+4n dwords)	ing 486 7 7 7 12+3n* 12+3n* 12+3n*	Pent. 4 4 4 3+n 3+n 3+n	ium NP NP NP NP NP
	Exampl  S/MOVSB/MOV  variations  movsb  movsd  rep movsb  rep movsw  rep movsd  Exampl  SX Move w  operands  reg, reg, men	3 * = cycle Le:  /SW/MOVSD s bytes 1 1 2 2 2 (n = c Le:  with sign- s bytes 3 1 3+d(0,	Move 8088 18 26 - 9+17n 9+25n - * = 5 ount of rep mo extend 1,2,4)	cr0, e.  data fr  186 9 9 - 8+8n 8+8n - if n=0, bytes, vsb  (386+)	om string 286 5 5 - 5+4n 5+4n - 13 if n=1	to str 386 7 7 7+4n 7+4n 7+4n dwords)	ing 486 7 7 7 12+3n* 12+3n* 12+3n*	Pent. 4 4 3+n 3+n 3+n 3+n 3+n	ium NP NP NP NP NP
MOV:	Example  S/MOVSB/MOV  Variations  movsb  movsd  rep movsb  rep movsw  rep movsd  Example  SX Move w  operands  reg, reg,  reg, men  (Note:	<pre>3 * = cycle Le: /SW/MOVSD s bytes 1 1 2 2 2 (n = c le: with sign- s bytes g 3 n 3+d(0, c destinat</pre>	Move 8088 18 26 - 9+17n 9+25n - * = 5 ount of rep mo extend 1,2,4) ion reg	cr0, e.  data fr  186 9 9 - 8+8n 8+8n - if n=0, bytes, vsb  (386+)	om string 286 5 5 - 5+4n 5+4n - 13 if n=1 words or	to str 386 7 7 7+4n 7+4n 7+4n dwords)	ing 486 7 7 7 12+3n* 12+3n* 12+3n*	Pent. 4 4 3+n 3+n 3+n 3+n 3+n	ium NP NP NP NP NP
	Exampl  S/MOVSB/MOV  variations  movsb  movsd  rep movsb  rep movsw  rep movsd  Exampl  SX Move w  operands  reg, reg, men	<pre>3 * = cycle Le: /SW/MOVSD s bytes 1 1 2 2 2 (n = c le: with sign- s bytes g 3 n 3+d(0, c destinat</pre>	Move 8088 18 26 - 9+17n 9+25n - * = 5 ount of rep mo extend 1,2,4)	cr0, e.  data fr  186 9 9 - 8+8n 8+8n - if n=0, bytes, vsb  (386+)	om string 286 5 5 - 5+4n 5+4n - 13 if n=1 words or	to str 386 7 7 7+4n 7+4n 7+4n dwords)	ing 486 7 7 7 12+3n* 12+3n* 12+3n*	Pent. 4 4 3+n 3+n 3+n 3+n 3+n	ium NP NP NP NP NP
MOV	Exampl  S/MOVSB/MOV  Variations  movsb  movsd  rep movsb  rep movsw  rep movsd  Exampl  SX Move w  operands  reg, reg  reg, men  (Note:  Exampl	3 * = cycle le:  /SW/MOVSD s bytes 1 1 2 2 2 (n = c le:  with sign- s bytes 3 n 3+d(0, destinat le:	Move 8088 18 26 - 9+17n 9+25n - * = 5 ount of rep mo extend 1,2,4) ion reg movsx	cr0, e.  data fr. 186 9 9 - 8+8n 8+8n - if n=0, bytes, vsb  (386+)  is 16 o ebx, a.	om string 286 5 5 - 5+4n 5+4n - 13 if n=1 words or	to str 386 7 7 7+4n 7+4n 7+4n dwords)	ing 486 7 7 7 12+3n* 12+3n* 12+3n*	Pent. 4 4 3+n 3+n 3+n 3+n 3+n	ium NP NP NP NP NP
MOV	Example S/MOVSB/MOV variations movsb movsd rep movsd rep movsd rep movsd Example SX Move woperands reg, reg, men (Note: Example ZX Move woperands reg, reg, men (Note: Example XX Move woperands reg, reg, reg, reg, reg, reg, reg, reg,	3 * = cycle le:  /SW/MOVSD s bytes 1 1 2 2 2 (n = c le:  with sign- s bytes 3 n 3+d(0, destinat le:  with zero-	Move 8088 18 26 - 9+17n 9+25n - * = 5 ount of rep mo extend 1,2,4) ion reg movsx	cr0, e.  data fr. 186 9 9 - 8+8n 8+8n - if n=0, bytes, vsb  (386+)  is 16 o ebx, a.	om string 286 5 5 - 5+4n 5+4n - 13 if n=1 words or	to str 386 7 7 7+4n 7+4n 7+4n dwords)	ing 486 7 7 12+3n* 12+3n* 12+3n*	Pent. 3 3 ar 16 b.	ium NP NP NP NP NP
MOV	Example  S/MOVSB/MOV  Variations  movsb  movsd  rep movsb  rep movsw  rep movsd  Example  SX Move w  operands  reg, reg, men  (Note: Example  ZX Move w  operands	3 * = cycle le:  /SW/MOVSD s bytes 1 1 2 2 2 (n = c le:  with sign- s bytes 3 1 3+d(0, destinat le:  with zero- s bytes	Move 8088 18 26 - 9+17n 9+25n - * = 5 ount of rep mo extend 1,2,4) ion reg movsx	cr0, e.  data fr. 186 9 9 - 8+8n 8+8n - if n=0, bytes, vsb  (386+)  is 16 o ebx, a.	om string 286 5 5 - 5+4n 5+4n - 13 if n=1 words or	to str 386 7 7 7+4n 7+4n 7+4n dwords) 386 3 6 3 6	ing 486 7 7 12+3n* 12+3n* 12+3n* 486 3 a se is 8 co	Pent.  4 4 3+n 3+n 3+n 3+n Pent. 3 3 Pent.	ium NP NP NP NP NP
MOV	Example S/MOVSB/MOV variations moved moved rep moved rep moved rep moved Example SX Move woperands reg, reg, men (Note: Example ZX Move woperands reg, reg, reg, reg, men (note: Example ZX Move woperands reg, reg, reg, reg, men (note: Example ZX Move woperands reg, reg	3 * = cycle le:  /SW/MOVSD s bytes 1 1 2 2 2 (n = c le:  vith sign- s bytes 3 1 3+d(0, c destinat le:  vith zero- s bytes 3 3 4 3 5 6 6 7 7 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8	Move 8088 18 26 - 9+17n 9+25n * = 5 ount of rep moderated 1,2,4) ion reg movsx extend	cr0, e.  data fr. 186 9 9 - 8+8n 8+8n - if n=0, bytes, vsb  (386+)  is 16 o ebx, a.	om string 286 5 5 - 5+4n 5+4n - 13 if n=1 words or	to str 386 7 7 7+4n 7+4n 7+4n dwords) 386 3 6 3; sourc	ing 486 7 7 12+3n* 12+3n* 12+3n* 486 3 a is 8 c	Pent. 3 3 3 16 b. Pent. 3	ium NP NP NP NP IUM NP its)
MOV	Exampl  S/MOVSB/MOV  Variations  movsb  movsd  rep movsb  rep movsd  Exampl  SX Move w  operands  reg, reg,  reg, men  (Note:  Exampl  ZX Move w  operands  reg, reg  reg, men  cy, reg  reg, men  cy, reg  reg, men  cy, reg  reg, men  cy, reg  reg, men	<pre>3 * = cycle te: /SW/MOVSD s bytes 1 1 2 2 2 (n = c) te: /sth sign- s bytes 3 1 3+d(0, 1 destinate) /sth zero- s bytes 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3</pre>	mov  Move 8088 18 26 - 9+17n 9+25n - * = 5 ount of rep mod extend  1,2,4) ion reg movsx extend  1,2,4)	cr0, e.  data fr  186 9 9 - 8+8n 8+8n - if n=0, bytes, vsb  (386+)  is 16 o ebx, a: (386+)	om string 286 5 5 5 5+4n 5+4n - 13 if n=1 words or	**To str 386	ing 486 7 7 12+3n* 12+3n* 12+3n*  486 3 3 e is 8 c 486 3 3	Pent. 3 3 3 Pent. 3 3 3	ium NP NP NP NP ium NP its)

Example: movzx ebx, ax

MUL	Unsign	ned multi	ply						
ope	rand	bytes	8088	186	286	386	486	Pent	ium
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
mem	8 2-	+d(0-2)	76-83+EA	32-34	16	12-17	13-18	11	NP
mem		+d(0-2) 1			24	12-25	13-26	11	NP
mem	32 2-	+d(0-2)	_	_	_	12-41	13-42	10	NP
im	plied	oper	and	result					
mult	iplica	nd (multi	plier)						
	AL	* byte	=	AX					
	AX	* word	=	DX:AX					
	EAX	* dword	=	EDX: EAX					
	Examp:	le:	mul	ebx					
NEG		compleme:	_						
ope	rand	bytes	8808	186	286	386	486	Pent	ium
reg		2	3	3	2	2	1	1	NP
mem		2+d(0-2)	24+EA	13	7	6	3	3	NP
	Examp.	le:	neg	eax					
MOD									
NOP	ио оре	eration	0000	106	286	386	106	Dont	- <del>-</del>
		bytes 1	8088 3	186 3	⊿oo 3	3	486 1	Pent 1	UV
	Evano	_	_	3	3	3	Τ.		UV
	Examp.	īe.	nop						
NOT	One's	compleme	nt negati	.on					
ope	rands	bytes	8088	186	286	386	486	Pent	ium
reg		2	3	3	2	2	1	1	NP
mem		2+d(0-2)	24+EA	13	7	6	3	3	NP
	Examp.	le:	not	eax					
0.0		. 1 . 2 1							
OR	_	al inclus		106	206	206	406	D t	
	ands	bytes 2	8088 3	186	286	386 2	486	Pent	
reg,			_	3	2 7	7	1 3	1	UV
•	reg	2+d(0,2)		10 10	7	, 6	2	3 2	UV
		2+d(0,2)	13+EA 4	4	3	2	1	1	UV
mem,		2+i(1,2) 2+d(0,2)	4 23+EA	16	3 7	7	3	3	UV UV*
mem,	±111111	+i(1,2)	ZSTEA	10	/	,	3	3	0 0
200	imm		4	4	3	2	1	1	UV
acc,	imm	1+i(1,2)				acement ar			UV
	Examp		or	eax, el		acement an	ia immeai	100	
			-	75,					
OUT	Output	t to port							
-	rands	bytes	8088	186	286	386	486	Pent	ium
	8, al	2	14	9	3	10	16	12	NP
	8, ax	2	14	9	3	10	16	12	NP
imm	8, 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
	_		Pr	otected	Mode		40-	_	
	rands	bytes				386	486	Pent	
	8, acc						11/31/29		
dx,	acc	1	c . ===		, ~		10/30/29	9/26/	24 NP
	Escome '	-			/ Chr	> IOPL /	V86		
	Examp.	TG.	out	dx, al					

variation outsb	TSW/OUTSD	Outpu	t string	g to po	rt			
	s bytes	_	186	286	386	486	Pent	ium
OULSD	1		14	5	14	17	13	NP
outsw	1		14	5	14	17	13	NP
outsd	1		_	_	14	17	13	NP
oacsa	_	Dz		Modo	11	Ι,	13	INI
	1 .	PI	otected	моае	206	406	<b>.</b>	
	bytes				386	486	Pent	
	1				8/28/28 1		10/27/	25 NP
	cycles f			/ CPL	> IOPL /	V86		
Examp	ole:	rep out	SW					
			_					
	word/dwor	<b>d from t</b> 8088	he stac 186	<b>k</b> 286	386	486	Pent	ium
operand	bytes	12						
reg	1		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
		Pr	otected	Mode				
operand	bytes			286	386	486	Pent	ium
CS/DS/ES	1			20	21	9	3-12	NP
SS	1			20	21	9	8-17	NP
FS/GS	2			20	21	9	3-12	NP
,	_			_	21	9	3-12	NP
Examp	ote:	pop	eax					
POPA/POPAD	Pop all (	186+)/Pc	p all d	ouble (	386+)			
variation		,,	186	286	386	486	Pent	iıım
popa	1		51	19	24	9	5	NP
	1		31	10	24	9	5	NP
popad	=				24	9	5	NP
popa = pop								
popad = pop				eax, e	cx, eax			
<del>-</del>	and esp a	re disca	rded)					
Examp	ole:	popa						
DODE / DODED	D 61/	D 61	الطييمات سي	- (206)	`			
	Pop flags/							
variation	-	8808	186	286	386	486	Pent	
popf	1	12	8	5	5	9	6	NP
popfd	1	-	-				O	NP
		Dr		-	5	9	6	NP
			otected		5	9	-	
	bytes		otected		5 386	9 486	-	NP
qoq	-		otected	Mode 286	386	486	6 Pent	NP ium
popf popfd	1	11	otected	Mode	386 5	486 6	6 Pent 4	NP ium NP
popfd	1 1		otected	Mode 286	386	486	6 Pent	NP ium
	1 1	popf	otected	Mode 286	386 5	486 6	6 Pent 4	NP ium NP
popfd Examp	1 1 ole:	popf		Mode 286	386 5	486 6	6 Pent 4	NP ium NP
popfd Examp	1 1 ole: word/dwor	popf <b>d to the</b>	stack	Mode 286 5 -	386 5 5	486 6 6	6 Pent 4 4	NP ium NP NP
popfd Examp  PUSH push a operand	1 1 ole: word/dwor bytes	popf <b>d to the</b> 8088	stack	Mode 286 5 -	386 5 5 386	486 6 6	6 Pent 4 4	NP ium NP NP
popfd Examp  PUSH push a operand reg	1 1 ole: word/dwor bytes 1	popf <b>d to the</b> 8088 15	stack 186 10	Mode 286 5 - 286 3	386 5 5 386 2	486 6 6 486 1	6 Pent 4 4 Pent 1	NP ium NP NP
popfd Examp  PUSH push a operand reg mem	1 1 0le: word/dwor bytes 1 2+d(0-2)	popf <b>d to the</b> 8088 15 24+EA	stack 186 10 16	Mode 286 5 - 286 3 5	386 5 5 386 2 5	486 6 486 1 4	Pent 4 4 Pent 1 2	NP ium NP NP  ium UV NP
popfd Examp  PUSH push a operand reg mem seg	1 1 0le: word/dwor bytes 1 2+d(0-2)	popf <b>d to the</b> 8088 15 24+EA 14	stack 186 10	Mode 286 5 - 286 3 5 3	386 5 5 386 2 5	486 6 6 486 1 4	Pent 4 4 Pent 1 2 1	NP ium NP NP  ium UV NP NP
popfd Examp  PUSH push a operand reg mem seg imm	1 1 1 0le: word/dwor bytes 1 2+d(0-2) 1 1+i(1,2)	popf <b>d to the</b> 8088 15 24+EA	stack 186 10 16	Mode 286 5 - 286 3 5	386 5 5 386 2 5 2	486 6 6 486 1 4 3	Pent 4 4 Pent 1 2 1 1	NP ium NP NP  ium UV NP NP NP
popfd Examp  PUSH push a operand reg mem seg imm FS/GS	1 1 1 ole: word/dwor bytes 1 2+d(0-2) 1 1+i(1,2) 2	popf  d to the  8088 15 24+EA 14 -	stack 186 10 16 9 -	Mode 286 5 - 286 3 5 3	386 5 5 386 2 5	486 6 6 486 1 4	Pent 4 4 Pent 1 2 1	NP ium NP NP  ium UV NP NP
popfd Examp  PUSH push a operand reg mem seg imm	1 1 1 ole: word/dwor bytes 1 2+d(0-2) 1 1+i(1,2) 2	popf <b>d to the</b> 8088 15 24+EA 14	stack 186 10 16	Mode 286 5 - 286 3 5 3	386 5 5 386 2 5 2	486 6 6 486 1 4 3	Pent 4 4 Pent 1 2 1 1	NP ium NP NP  ium UV NP NP NP
popfd Examp  PUSH push a operand reg mem seg imm FS/GS Examp	1 1 1 ole: word/dwor bytes 1 2+d(0-2) 1 1+i(1,2) 2	popf  d to the 8088 15 24+EA 14 - push	stack 186 10 16 9 - - eax	Mode 286 5 - 286 3 5 3 -	386 5 5 386 2 5 2 2	486 6 6 486 1 4 3	Pent 4 4 Pent 1 2 1 1	NP ium NP NP  ium UV NP NP NP
popfd Examp  PUSH push a operand reg mem seg imm FS/GS Examp	1 1 2)le: word/dwor bytes 1 2+d(0-2) 1 1+i(1,2) 2	popf  d to the 8088 15 24+EA 14 - push	* stack 186 10 16 9 - eax	Mode 286 5 - 286 3 5 3 -	386 5 5 386 2 5 2 2 2	486 6 6 486 1 4 3 1 3	Pent 4 4 4 4 Pent 1 2 1 1 1	NP ium NP NP ium UV NP NP NP
popfd Examp  PUSH push a operand reg mem seg imm FS/GS Examp  PUSHA/PUSHAD variation	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	popf  d to the 8088 15 24+EA 14 - push	stack 186 10 16 9 - eax Push al	Mode 286 5 - 286 3 5 3 - 1 <b>doub1</b> 6 286	386 5 5 386 2 2 2 2 2	486 6 6 486 1 4 3 1 3	Pent 4 4 Pent 1 2 1 1 Pent	NP ium NP NP ium UV NP NP NP
popfd Examp  PUSH push a operand reg mem seg imm FS/GS Examp  PUSHA/PUSHAD variation pusha	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	popf  d to the 8088 15 24+EA 14 - push	* stack 186 10 16 9 - eax	Mode 286 5 - 286 3 5 3 -	386 5 5 5 2 2 2 2 2 8 6 (386+) 386 18	486 6 6 486 1 4 3 1 3	Pent 4 4 4 Pent 1 2 1 1 1 Pent 5	NP ium NP NP ium UV NP NP NP
popfd Examp  PUSH push a operand reg mem seg imm FS/GS Examp  PUSHA/PUSHAD variation pusha pushad	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	popf  d to the 8088 15 24+EA 14 - push (186+)/	stack 186 10 16 9 - eax Push al	Mode 286 5 - 286 3 5 3 - 1 <b>doubl</b> 286 17 -	386 5 5 5 386 2 2 2 2 2 2 8 6 18 18	486 6 6 486 1 4 3 1 3	Pent 4 4 Pent 1 2 1 1 Pent	NP ium NP NP ium UV NP NP NP
popfd Examp  PUSH push a operand reg mem seg imm FS/GS Examp  PUSHA/PUSHAD variation pusha pushad pusha = pu	1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	popf  d to the 8088 15 24+EA 14 - push (186+)/	stack 186 10 16 9 - eax Push al 186 36 - sp, bp	Mode 286 5 - 286 3 5 3 - 1 double 286 17 - , si, d	386 5 5 5 386 2 2 2 2 2 2 18 18	486 6 6 486 1 4 3 1 3	Pent 4 4 4 Pent 1 2 1 1 1 Pent 5	NP ium NP NP ium UV NP NP NP
popfd Examp  PUSH push a operand reg mem seg imm FS/GS Examp  PUSHA/PUSHAD variation pusha pushad pushad = pu pushad = pu	l le: word/dwor bytes 1 2+d(0-2) 1 1+i(1,2) 2 de: Push all as bytes 1 1 1sh ax, cx, ash eax, ec	popf  d to the 8088 15 24+EA 14 - push (186+)/	stack 186 10 16 9 - eax Push al 186 36 - sp, bp	Mode 286 5 - 286 3 5 3 - 1 double 286 17 - , si, d	386 5 5 5 386 2 2 2 2 2 2 18 18	486 6 6 486 1 4 3 1 3	Pent 4 4 4 Pent 1 2 1 1 1 Pent 5	NP ium NP NP ium UV NP NP NP
popfd Examp  PUSH push a operand reg mem seg imm FS/GS Examp  PUSHA/PUSHAD variation pusha pushad pusha = pu	l le: word/dwor bytes 1 2+d(0-2) 1 1+i(1,2) 2 de: Push all as bytes 1 1 1sh ax, cx, ash eax, ec	popf  d to the 8088 15 24+EA 14 - push (186+)/	stack 186 10 16 9 - eax Push al 186 36 - sp, bp	Mode 286 5 - 286 3 5 3 - 1 double 286 17 - , si, d	386 5 5 5 386 2 2 2 2 2 2 18 18	486 6 6 486 1 4 3 1 3	Pent 4 4 4 Pent 1 2 1 1 1 Pent 5	NP ium NP NP ium UV NP NP NP

		0+100	a bretoa	0000	106	206	206	106	Dont	:
			s bytes 1	8088 14	186 9	286 3	386 4	486 4	Pent: 9	NP
	push push		1	T.4	<i>9</i> –	- -	4	4	9	NP
	pusi	iiu	Т.	- Dro		ed Mode	4	4	9	NP
			bytes	FIC	JUECU	286	386	486	Pent:	i 11m
	push	n <b>f</b>	1			3	4	3	3	NP
	push		1			_	4	3	3	NP
	Publ		le:	pushf			-	3	3	141
		Litamp	10	Pabiii						
RCL		Rotat	e bits lef	t with C	7					
		ands	bytes	8088	186	286	386	486	Pent	ium
	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		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
		Examp	le:	rcl	eax,	16				
					_					
RCR			e bits rig			006	206	406		
	-		bytes		186	286	386	486	Pent:	
	reg,		2	2	2	2 7	9	3 4	1 3	PU
		1	2+d(0,2) 2	23+EA 8+4n	15 5+n	-	10 9	=	3 7-24	PU
		cl cl	2+d(0,2)	_	_	_	10	8-30 9-31	9-26	
		imm	3		5+n		9	8-31	8-25	
		imm	3+d(0,2)	_	17+n	8+n	10	9-31	10-27	
	mem,	Examp		rcr	eax,		10	J J±	10 27	INI
		Litamp	10	101	carry	10				
ROL		Rotat	e bits lef	t						
				0000	186	286	386	486	Pent	ium
	oper	ands	bytes	8088	T 0 0		300	100	- 0	
	oper reg,		bytes 2	2	2	2	3	3	1	PU
	reg,		2	2						PU PU
	reg,	1	2 2+d(0,2) 2	2 23+EA 8+4n	2 15 5+n	2 7 5+n	3	3 4 3	1	
	reg, mem, reg, mem,	1 1 cl cl	2 2+d(0,2) 2	2 23+EA 8+4n	2 15 5+n	2 7 5+n	3 7 3 7	3 4 3 4	1 3	PU
	reg, mem, reg, mem,	1 1 cl cl imm	2 2+d(0,2) 2 2+d(0,2) 3	2 23+EA 8+4n	2 15 5+n 17+n 5+n	2 7 5+n 8+n 5+n	3 7 3 7 3	3 4 3 4 2	1 3 4 4 1	PU NP NP PU
	reg, mem, reg, mem, reg,	1 1 cl cl imm imm	2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2)	2 23+EA 8+4n 28+EA+4n -	2 15 5+n 17+n 5+n 17+n	2 7 5+n 8+n 5+n 8+n	3 7 3 7 3	3 4 3 4 2 4	1 3 4 4 1 3	PU NP NP
	reg, mem, reg, mem, reg, mem,	1 1 cl cl imm imm = not	2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) pairable	2 23+EA 8+4n 28+EA+4n - - if there	2 15 5+n 17+n 5+n 17+n is a	2 7 5+n 8+n 5+n 8+n displace	3 7 3 7 3	3 4 3 4 2 4	1 3 4 4 1 3	PU NP NP PU
	reg, mem, reg, mem, reg, mem,	1 1 cl cl imm imm	2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) pairable	2 23+EA 8+4n 28+EA+4n -	2 15 5+n 17+n 5+n 17+n	2 7 5+n 8+n 5+n 8+n displace	3 7 3 7 3	3 4 3 4 2 4	1 3 4 4 1 3	PU NP NP PU
ROR	reg, mem, reg, mem, reg, mem,	1 cl cl imm imm = not Examp	2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) pairable le:	2 23+EA 8+4n 28+EA+4n - if there rol	2 15 5+n 17+n 5+n 17+n is a	2 7 5+n 8+n 5+n 8+n displace	3 7 3 7 3	3 4 3 4 2 4	1 3 4 4 1 3	PU NP NP PU
ROR	reg, mem, reg, mem, reg, mem,	1 cl cl imm imm = not Examp	2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) pairable le:	2 23+EA 8+4n 28+EA+4n - if there rol	2 15 5+n 17+n 5+n 17+n is a eax,	2 7 5+n 8+n 5+n 8+n displace	3 7 3 7 3 7 ement and	3 4 3 4 2 4 immediat	1 3 4 4 1 3	PU NP NP PU PU*
ROR	reg, mem, reg, mem, reg, mem,	1 cl cl imm imm = not Examp  Rotat ands	2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) pairable le: e bits rig	2 23+EA 8+4n 28+EA+4n - if there rol ght 8088	2 15 5+n 17+n 5+n 17+n is a eax,	2 7 5+n 8+n 5+n 8+n displace 16	3 7 3 7 3 7 ement and	3 4 3 4 2 4 immediat	1 3 4 4 1 3 ce	PU NP NP PU PU*
ROR	reg, mem, reg, mem, reg, mem, *	1 cl cl imm imm = not Examp  Rotat ands 1	2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) pairable le: e bits rig bytes 2	2 23+EA 8+4n 28+EA+4n - if there rol ght 8088 2	2 15 5+n 17+n 5+n 17+n is a eax,	2 7 5+n 8+n 5+n 8+n displace 16	3 7 3 7 3 7 ement and	3 4 3 4 2 4 immediat	1 3 4 4 1 3 ce	PU NP NP PU PU*
ROR	reg, mem, reg, mem, reg, mem, *	1 1 cl cl imm imm = not Examp  Rotat rands 1 1	2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) pairable le: e bits rig bytes 2 2+d(0,2)	2 23+EA 8+4n 28+EA+4n - if there rol wht 8088 2 23+EA	2 15 5+n 17+n 5+n 17+n is a eax, 186 2 15	2 7 5+n 8+n 5+n 8+n displace 16 286 2 7	3 7 3 7 3 7 ement and 386 3	3 4 3 4 2 4 immediat	1 3 4 4 1 3 ce	PU NP NP PU*
ROR	reg, mem, reg, mem, reg, mem, *	1 cl cl imm imm = not Examp  Rotat ands 1 cl	2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) pairable le: e bits rig bytes 2 2+d(0,2)	2 23+EA 8+4n 28+EA+4n - if there rol sht 8088 2 23+EA 8+4n	2 15 5+n 17+n 5+n 17+n is a eax, 186 2 15 5+n	2 7 5+n 8+n 5+n 8+n displace 16 286 2 7 5+n	3 7 3 7 3 7 ement and 386 3 7	3 4 3 4 2 4 immediat	1 3 4 4 1 3 ce	PU NP NP PU PU*
ROR	reg, mem, reg, mem, reg, mem, *	1 1 cl cl imm imm = not Examp  Rotat cands 1 cl cl	2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) pairable le: e bits rig bytes 2 2+d(0,2) 2 2+d(0,2)	2 23+EA 8+4n 28+EA+4n - if there rol ght 8088 2 23+EA 8+4n 28+EA+4n	2 15 5+n 17+n 5+n 17+n is a eax, 186 2 15 5+n	2 7 5+n 8+n 5+n 8+n displace 16 286 2 7 5+n 8+n	3 7 3 7 3 7 ement and 386 3 7	3 4 3 4 2 4 immediat	1 3 4 4 1 3 ce Pent: 1 3	PU NP NP PU PU*
ROR	reg, mem, reg, mem, * oper reg, mem, reg, mem, reg,	1 1 cl cl imm imm = not Examp  Rotat rands 1 cl cl imm	2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) pairable le: e bits rig bytes 2 2+d(0,2) 2 2+d(0,2)	2 23+EA 8+4n 28+EA+4n - if there rol ght 8088 2 23+EA 8+4n 28+EA+4n	2 15 5+n 17+n 5+n 17+n is a eax, 186 2 15 5+n 17+n	2 7 5+n 8+n 5+n 8+n displace 16 286 2 7 5+n 8+n 5+n	3 7 3 7 3 7 ement and 386 3 7 3	3 4 3 4 2 4 immediat 486 3 4 3 4	1 3 4 4 1 3 ce Pent:	PU NP PU PU*
ROR	reg, mem, reg, mem, *  oper reg, mem, reg, mem, reg, mem,	1 1 cl cl imm imm = not Examp  Rotat rands 1 cl cl imm imm	2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) pairable le: e bits rig bytes 2 2+d(0,2) 2 2+d(0,2) 3	2 23+EA 8+4n 28+EA+4n - if there rol ght 8088 2 23+EA 8+4n 28+EA+4n -	2 15 5+n 17+n 5+n 17+n is a eax, 186 2 15 5+n 17+n 5+n 17+n	2 7 5+n 8+n 5+n 8+n displace 16 286 2 7 5+n 8+n 5+n 8+n	3 7 3 7 3 7 ement and 386 3 7 3 7 3	3 4 3 4 2 4 immediat 486 3 4 3 4 2 4	1 3 4 4 1 3 2ee Pent: 1 3 4 4 1 3	PU NP PU PU*
ROR	reg, mem, reg, mem, *  oper reg, mem, reg, mem, reg, mem,	1 1 cl cl imm imm = not Examp  Rotat rands 1 cl cl imm imm	2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) pairable le:  e bits rig bytes 2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) t pairable	2 23+EA 8+4n 28+EA+4n - if there rol ght 8088 2 23+EA 8+4n 28+EA+4n -	2 15 5+n 17+n 5+n 17+n is a eax, 186 2 15 5+n 17+n 5+n 17+n	2 7 5+n 8+n 5+n 8+n displace 16 286 2 7 5+n 8+n 5+n 8+n a displace	3 7 3 7 3 7 ement and 386 3 7 3 7 3	3 4 3 4 2 4 immediat 486 3 4 3 4 2 4	1 3 4 4 1 3 2ee Pent: 1 3 4 4 1 3	PU NP PU *  Lum PU PU NP NP PU
	reg, mem, reg, mem, *  oper reg, mem, reg, mem,	l l cl cl cmm imm = not Examp  Rotat cands l cl cl imm imm f = no Examp	2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) pairable le:  e bits rig bytes 2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) t pairable le:	2 23+EA 8+4n 28+EA+4n - if there rol sht 8088 2 23+EA 8+4n 28+EA+4n - e if there	2 15 5+n 17+n 5+n 17+n is a eax, 186 2 15 5+n 17+n 5+n 17+n e is a eax,	2 7 5+n 8+n 5+n 8+n displace 16 286 2 7 5+n 8+n 5+n 8+n a displace	3 7 3 7 3 7 ement and 386 3 7 3 7 3 7	3 4 3 4 2 4 immediat 486 3 4 3 4 2 4	1 3 4 4 1 3 2ee Pent: 1 3 4 4 1 3	PU NP PU *  Lum PU PU NP NP PU
ROR	reg, mem, reg, mem, *  oper reg, mem, reg, mem,	l l cl cl cmm imm = not Examp  Rotat cands l cl cl imm imm f = no Examp	2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) pairable le:  e bits rig bytes 2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) t pairable le:  from model	2 23+EA 8+4n 28+EA+4n - if there rol sht 8088 2 23+EA 8+4n 28+EA+4n - e if there	2 15 5+n 17+n 5+n 17+n is a eax, 186 2 15 5+n 17+n 5+n 17+n e is a eax,	2 7 5+n 8+n 5+n 8+n displace 16 286 2 7 5+n 8+n 5+n 8+n a displace	3 7 3 7 3 7 ement and 386 3 7 3 7 3 7	3 4 3 4 2 4 immediat 486 3 4 3 4 2 4	1 3 4 4 1 3 3 4 4 1 3	PU NP PU PU*
	reg, mem, reg, mem, *  oper reg, mem, reg, mem,	l l cl cl mm imm = not Examp  Rotat ands l cl cl imm imm f = no Examp	2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) pairable le:  e bits rig bytes 2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) t pairable le:  from model bytes	2 23+EA 8+4n 28+EA+4n - if there rol sht 8088 2 23+EA 8+4n 28+EA+4n - e if there	2 15 5+n 17+n 5+n 17+n is a eax, 186 2 15 5+n 17+n 5+n 17+n e is a eax,	2 7 5+n 8+n 5+n 8+n displace 16 286 2 7 5+n 8+n 5+n 8+n a displace	3 7 3 7 3 7 ement and 386 3 7 3 7 3 7	3 4 3 4 2 4 immediat 486 3 4 3 4 2 4	1 3 4 4 1 3 4 4 1 3 ate	PU NP PU PU*
	reg, mem, reg, mem, *  oper reg, mem, reg, mem,	l l cl cl imm imm = not Examp  Rotat cands l cl imm imm = no Examp	2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) pairable le:  e bits rig bytes 2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) t pairable le:  from model bytes 2	2 23+EA 8+4n 28+EA+4n - if there rol  (ht 8088 2 23+EA 8+4n 28+EA+4n - if there ror  specific	2 15 5+n 17+n 5+n 17+n is a eax, 186 2 15 5+n 17+n 5+n 17+n e is a eax,	2 7 5+n 8+n 5+n 8+n displace 16 286 2 7 5+n 8+n 5+n 8+n a displace	3 7 3 7 3 7 ement and 386 3 7 3 7 3 7	3 4 3 4 2 4 immediat 486 3 4 3 4 2 4	1 3 4 4 1 3 3 4 4 1 3	PU NP PU PU*
	reg, mem, reg, mem, *  oper reg, mem, reg, mem,	l l cl cl imm imm = not Examp  Rotat cands l cl imm imm = no Examp	2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) pairable le:  e bits rig bytes 2 2+d(0,2) 2 2+d(0,2) 3 3+d(0,2) t pairable le:  from model bytes	2 23+EA 8+4n 28+EA+4n - if there rol sht 8088 2 23+EA 8+4n 28+EA+4n - e if there	2 15 5+n 17+n 5+n 17+n is a eax, 186 2 15 5+n 17+n 5+n 17+n e is a eax,	2 7 5+n 8+n 5+n 8+n displace 16 286 2 7 5+n 8+n 5+n 8+n a displace	3 7 3 7 3 7 ement and 386 3 7 3 7 3 7	3 4 3 4 2 4 immediat 486 3 4 3 4 2 4	1 3 4 4 1 3 4 4 1 3 ate	PU NP PU PU*

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
retn 1 20 16 11+m 10+m 5
retn imm16 1+d(2) 24 18 11+m 10+m 5
retf 1 34 22 15+m 18+m 13
retf imm16 1+d(2) 33 25 15+m 18+m 14 Pentium NP 3 NP 4 NP 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/ 286 386 486 Pentium operands bytes 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 83 NP 2 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 3 PU\* \* = not pairable if there is a displacement and immediate sal = shift arithmetic left
shl = shift left (same as sal)
sar = shift arithmetic right
shr = shift right Example: shl eax, 1 Store AH into flags SAHF bytes 8088 186 286 386 486 Pentium 1 4
Example: sahf 3 2 2 3 2 NP Integer subtraction with borrow SBB operands bytes 8088 186 286 386 486 Pentium reg, reg 2 3 3 2 2
mem, reg 2+d(0,2) 24+EA 10 7 7
reg, mem 2+d(0,2) 13+EA 10 7 6
reg, imm 2+i(1,2) 4 4 3 2
mem, imm 2+d(0,2) 23+EA 16 7 7 1 3 2 2 PU 1 PU 3 3 PU\* +i(1,2)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

פטעפ	/ccaeb/cc	ASW/SCASD	Saan	string	data			
				-		206	106	Dontium
	variation		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 scas	b 2	9+15n	5+15n	5+8n	5+8n	7+5n*	8+4n NP
	repX scas	w 2	9+19n	5+15n	5+8n	5+8n	7+5n*	8+4n NP
	repX scas		_	_	_	5+8n	7+5n*	8+4n NP
	_	pe or repz	or rep	ne or re	epn <i>z</i>			
•			= 5 if		21111			
					ag wor	ds or dwo	orde)	
	Errama	le:	repne	_	Jes, Wol	as or awa	oras /	
	Examp	16.	repile	scasb				
SET	Set h	yte to 1 c	n condi	tion ale	20 COT T	o 0 (386	- )	
SEI			ni condi	CION EIS	se set t	386	486	Dontium
	operand	-						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		
se	tCC = one	of:						
	seta	setae se	etb s	etbe s	setc	sete		
	setg	setge se	etl s	etle s	setna	setnae		
		setnbe se	tnc s	etne s	setng	setnge		
	setnl	setnle se	etno s		_	setnz		
				-		setz		
	Examp	-	setne	al	, , ,	2002		
	Liamp	10	DCCIIC	41				
SCDT	Store	global de	ecripto	r table	registe	r (286+)		
BGDI	operand		BCI IPCO	I Cable	286	386	486	Pentium
	_	=						
	mem48	5	3.		11	9	10	4 NP
	Examp	le:	sgat	aescri	iptor[eb	x J		
	<b>a</b> .						>	
SIDT		interrupt	descri	ptor tar				
	operand	-			286	386	486	Pentium
	mem48	5			12	9	10	4 NP
	Examp	le:	sidt	descri	iptor[eb	x]		
SHLD		e precisio		left (3	386+)			
	operands		rtes			386	486	Pentium
	reg, reg	, imm 4	Ŀ			3	2	4 NP
	mem, reg	, imm 4+	-d(0-2)			7	3	4 NP
	reg, reg	, cl 4	<u> </u>			3	3	4 NP
	mem, reg		-d(0-2)			7	4	5 NP
	Examp		shld	eax, e	ebx, 16			
				,	,			
SHRD	Doubl	e precisio	n shift	right (	(386+)			
	operands		tes	J '	,	386	486	Pentium
	reg, reg					3	2	4 NP
	mem, reg	•	-d(0-2)			7	3	4 NP
						3	3	
	reg, reg							4 NP
	mem, reg		-d(0-2)			7	4	5 NP
	Examp	le:	shrd	eax, e	ebx, 16			
a	۵.	11		4 - 1 - 3		(005:)		
SLDT		local des	criptor	table 1				
	operands				286	386	486	Pentium
	r16	3			2	2	2	2 NP
	mem16	3+d(0-2)			3	2	3	2 NP
	Examp	le:	sldt	ax				
SMSW	Store	machine s	status w	ord (286	5+)			
	_	1			286	386	486	Pentium
	operands	bytes			200	300	100	I CIICI am
	operands r16	bytes 3			2	2	2	4 NP

mem16 $3+d(0-2)$			3	3	3	4	NP
Example:	smsw	ax					
STC Set the carry f							
bytes	8088	186	286	386	486	Pent	
1	2	2	2	2	2	2	NP
Example: STD Set direction f	stc	. +0 *0		ina dina	ation)		
STD Set direction f	8088	. <b></b> 186	286	386	486	Pent	iım
1	2	2	2	2	2	2	NP
Example:	std	_	_	_	_	_	
STI Set interrupt f	lag (ena	ble)					
bytes	8808	186	286	386	486	Pent	ium
1	2	2	2	3	5	7	NP
Example:	sti						
STOS/STOSB/STOSW/STOSD		strin	_	206	406		
variations bytes	8088	186	286	386	486	Pent	
stosb 1 stosw 1	11 15	10 10	3 3	4 4	5 5	3 3	NP
stosw 1 stosd 1	_	_	- -	4	5 5	3	NP NP
rep stosb 2	9+10n	6+9n	4+3n	5+5n	7+4n*	-	NP
rep stosb 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 i	f n=0,	13 if n=				
(n = c)			words or				
Example:	rep	stosd					
STR Store task regi	ster (28	86+)					
operand bytes			286	386	486	Pent	
r16 3			2	2	2	2	NP
mem16 $3+d(0-2)$		1	3	2	3	2	NP
Example:	str	bx					
SUB Integer subtrac	tion						
operands bytes	8088	186	286	386	486	Pent	ium
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)	23+EA	16	7	7	3	3	UV*
+i(1,2)							
acc, imm $1+i(1,2)$	4	4	3	2	1	1	UV
* = not pairable				ment and	limmediat	ce	
Example:	sub	eax,	ebx				
TEST Logical compare							
operands bytes	8088	186	286	386	486	Pent	iıım
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 $2+d(0,2)$	11+EA	10	6	5	2	2	UV*
+i(1,2)							
acc, imm $1+i(1,2)$	4	4 .	3	2	1	1	UV
* = not pairabl				ement ar	ıd immedia	ate	
Example:	sub	eax,	ebx				
VERR Verify a segmen	t for m	adina	(286±)				
VERR Verify a segmen operand bytes	re TOL LE	aurng	286	386	486	Pent	iım
r16 3			14	10	11	7	NP
						/	NIP

	mem16	3+d(0,2)			16	11	11	7	NP
	Examp		verr	ax					
VERW	VERW Verify a segment for writing (286+) operand bytes 286 386 486 Pentium								
	-	bytes				386			
	r16	3			14	15	11	7	NP
	mem16	3+d(0,2)			16	16	11	7	NP
	Examp	le:	verr	ax					
WAIT	Wait	for co-pro	cessor						
		bytes	8088	186	286	386	486	Pent	ium
		1	4	6	3	6	1-3	1	NP
	Examp		wait	· ·	J	· ·	_ 0	_	
	платр	101	Walt						
WBIN	/D Write	-back and	invalida	te data	cache (4	186+)			
		bytes					486	Pent	ium
		2					5 2	000+	NP
	Examp	le:	wbinvd						
WRMSI	R Write	to model	specific	registe	r (PENTI	TTM+)			
	200	bytes				/		Pent	ium
		2						30-45	
	Examp		wrmsr						
	Litamp		WIMDI						
XADD	Excha	nge and ad	d (486+)						
	operands	bytes					486	Pent	ium
	reg, reg	3					3	3	NP
	mem, reg	3+d(0-2	)				4	4	NP
	Examp	le:	xadd	eax, ek	ЭX				
XCHG		nge regist							
	operands	-	8088	186	286	386	486	Pent	ium
	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					
	Examp	le:	xchg	ax, dx					
VT Nm	/VI 7 TO 1	Table lest		alation					
ALAT,	/XLATB	Table look bytes	<b>-up tran</b> 8088	186	286	386	486	Pent	ium
		bytes 1	11	11	∠00 5	5	4	4	NP
	Erromo		xlat	11	5	5	4	4	NP
	Examp	ıe.	XIAL						
XOR	Logic	al exclusi	ve or						
(	operands	bytes	8088	186	286	386	486	Pent	ium
]	reg, reg	2	3	3	2	2	1	1	UV
r	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
		2+d(0,2)	23+EA	16	7	7	3	3	UV*
	110011			± 0	,	,	J	J	J v
	nem, imm								
	•	+i(1,2)	4	4	3	2	1	1	UV
	acc, imm	+i(1,2) 1+i(1,2)			_				UV
	acc, imm	+i(1,2) 1+i(1,2) ot pairable			displace				UV