Intel 8086 Family Architecture. . . . . . . . . . . . . . . . . . . . . 3

Instruction Clock Cycle Calculation . . . . . . . . . . . . . . . . . . 3

8088/8086 Effective Address (EA) Calculation . . . . . . . . . . . . . 3

Task State Calculation. . . . . . . . . . . . . . . . . . . . . . . . . 4

FLAGS - Intel 8086 Family Flags Register. . . . . . . . . . . . . . . . 4

MSW - Machine Status Word (286+ only) . . . . . . . . . . . . . . . . . 5

8086/80186/80286/80386/80486 Instruction Set. . . . . . . . . . . . . . 6

AAA - Ascii Adjust for Addition. . . . . . . . . . . . . . . . . . 6

AAD - Ascii Adjust for Division. . . . . . . . . . . . . . . . . . 6

AAM - Ascii Adjust for Multiplication. . . . . . . . . . . . . . . 6

AAS - Ascii Adjust for Subtraction . . . . . . . . . . . . . . . . 6

ADC - Add With Carry . . . . . . . . . . . . . . . . . . . . . . . 7

ADD - Arithmetic Addition. . . . . . . . . . . . . . . . . . . . . 7

AND - Logical And. . . . . . . . . . . . . . . . . . . . . . . . . 7

ARPL - Adjusted Requested Privilege Level of Selector (286+ PM). . 7

BOUND - Array Index Bound Check (80188+) . . . . . . . . . . . . . 8

BSF - Bit Scan Forward (386+). . . . . . . . . . . . . . . . . . . 8

BSR - Bit Scan Reverse (386+) . . . . . . . . . . . . . . . . . . 8

BSWAP - Byte Swap (486+) . . . . . . . . . . . . . . . . . . 8

BT - Bit Test (386+) . . . . . . . . . . . . . . . . . . 9

BTC - Bit Test with Compliment (386+). . . . . . . . . . . . . . . 9

BTR - Bit Test with Reset (386+) . . . . . . . . . . . . . . . . . 9

BTS - Bit Test and Set (386+) . . . . . . . . . . . . . . . . . . 9

CALL - Procedure Call. . . . . . . . . . . . . . . . . . . . . . . 10

CBW - Convert Byte to Word . . . . . . . . . . . . . . . . . . . . 10

CDQ - Convert Double to Quad (386+). . . . . . . . . . . . . . . . 10

CLC - Clear Carry. . . . . . . . . . . . . . . . . . . . . . . . . 11

CLD - Clear Direction Flag . . . . . . . . . . . . . . . . . . . . 11

CLI - Clear Interrupt Flag (disable) . . . . . . . . . . . . . . . 11

CLTS - Clear Task Switched Flag (286+ privileged). . . . . . . . . 11

CMC - Complement Carry Flag. . . . . . . . . . . . . . . . . . . . 11

CMP - Compare. . . . . . . . . . . . . . . . . . . . . . . . . . . 12

CMPS - Compare String (Byte, Word or Doubleword) . . . . . . . . . 12

CMPXCHG - Compare and Exchange . . . . . . . . . . . . . . . . . . 12

CWD - Convert Word to Doubleword . . . . . . . . . . . . . . . . . 12

CWDE - Convert Word to Extended Doubleword (386+). . . . . . . . . 13

DAA - Decimal Adjust for Addition. . . . . . . . . . . . . . . . . 13

DAS - Decimal Adjust for Subtraction . . . . . . . . . . . . . . . 13

DEC - Decrement. . . . . . . . . . . . . . . . . . . . . . . . . . 13

DIV - Divide . . . . . . . . . . . . . . . . . . . . . . . . . . . 13

ENTER - Make Stack Frame (80188+) . . . . . . . . . . . . . . . . 14

ESC - Escape . . . . . . . . . . . . . . . . . . . . . . . . . . . 14

HLT - Halt CPU . . . . . . . . . . . . . . . . . . . . . . . . . . 14

IDIV - Signed Integer Division . . . . . . . . . . . . . . . . . . 14

IMUL - Signed Multiply . . . . . . . . . . . . . . . . . . . . . . 15

IN - Input Byte or Word From Port. . . . . . . . . . . . . . . . . 15

INC - Increment. . . . . . . . . . . . . . . . . . . . . . . . . . 16

INS - Input String from Port (80188+) . . . . . . . . . . . . . . 16

INT - Interrupt. . . . . . . . . . . . . . . . . . . . . . . . . . 16

INTO - Interrupt on Overflow . . . . . . . . . . . . . . . . . . . 17

INVD - Invalidate Cache (486+). . . . . . . . . . . . . . . . . . 17

INVLPG - Invalidate Translation Look-Aside Buffer Entry (486+) . . 17

IRET/IRETD - Interrupt Return. . . . . . . . . . . . . . . . . . . 17

Jxx - Jump Instructions Table. . . . . . . . . . . . . . . . . . . 18

JCXZ/JECXZ - Jump if Register (E)CX is Zero. . . . . . . . . . . . 18

JMP - Unconditional Jump . . . . . . . . . . . . . . . . . . . . . 19

LAHF - Load Register AH From Flags . . . . . . . . . . . . . . . . 19

LAR - Load Access Rights (286+ protected). . . . . . . . . . . . . 19

LDS - Load Pointer Using DS. . . . . . . . . . . . . . . . . . . . 20

LEA - Load Effective Address . . . . . . . . . . . . . . . . . . . 20

LEAVE - Restore Stack for Procedure Exit (80188+). . . . . . . . . 20

LES - Load Pointer Using ES. . . . . . . . . . . . . . . . . . . . 20

LFS - Load Pointer Using FS (386+) . . . . . . . . . . . . . . . . 21

LGDT - Load Global Descriptor Table (286+ privileged). . . . . . . 21

LIDT - Load Interrupt Descriptor Table (286+ privileged) . . . . . 21

LGS - Load Pointer Using GS (386+) . . . . . . . . . . . . . . . . 21

LLDT - Load Local Descriptor Table (286+ privileged) . . . . . . . 22

LMSW - Load Machine Status Word (286+ privileged). . . . . . . . . 22

LOCK - Lock Bus. . . . . . . . . . . . . . . . . . . . . . . . . . 22

LODS - Load String (Byte, Word or Double). . . . . . . . . . . . . 22

LOOP - Decrement CX and Loop if CX Not Zero. . . . . . . . . . . . 23

LOOPE/LOOPZ - Loop While Equal / Loop While Zero . . . . . . . . . 23

LOOPNZ/LOOPNE - Loop While Not Zero / Loop While Not Equal . . . . 23

LSL - Load Segment Limit (286+ protected). . . . . . . . . . . . . 23

LSS - Load Pointer Using SS (386+) . . . . . . . . . . . . . . . . 24

LTR - Load Task Register (286+ privileged) . . . . . . . . . . . . 24

MOV - Move Byte or Word. . . . . . . . . . . . . . . . . . . . . . 24

MOVS - Move String (Byte or Word). . . . . . . . . . . . . . . . . 25

MOVSX - Move with Sign Extend (386+) . . . . . . . . . . . . . . . 25

MOVZX - Move with Zero Extend (386+) . . . . . . . . . . . . . . . 25

MUL - Unsigned Multiply. . . . . . . . . . . . . . . . . . . . . . 25

NEG - Two's Complement Negation. . . . . . . . . . . . . . . . . . 26

NOP - No Operation (90h) . . . . . . . . . . . . . . . . . . . . . 26

NOT - One's Compliment Negation (Logical NOT). . . . . . . . . . . 26

OR - Inclusive Logical OR. . . . . . . . . . . . . . . . . . . . . 26

OUT - Output Data to Port. . . . . . . . . . . . . . . . . . . . . 27

OUTS - Output String to Port (80188+) . . . . . . . . . . . . . . 27

POP - Pop Word off Stack . . . . . . . . . . . . . . . . . . . . . 27

POPA/POPAD - Pop All Registers onto Stack (80188+). . . . . . . . 28

POPF/POPFD - Pop Flags off Stack . . . . . . . . . . . . . . . . . 28

PUSH - Push Word onto Stack. . . . . . . . . . . . . . . . . . . . 28

PUSHA/PUSHAD - Push All Registers onto Stack (80188+) . . . . . . 28

PUSHF/PUSHFD - Push Flags onto Stack . . . . . . . . . . . . . . . 29

RCL - Rotate Through Carry Left. . . . . . . . . . . . . . . . . . 29

RCR - Rotate Through Carry Right . . . . . . . . . . . . . . . . . 29

REP - Repeat String Operation. . . . . . . . . . . . . . . . . . . 30

REPE/REPZ - Repeat Equal / Repeat Zero . . . . . . . . . . . . . . 30

REPNE/REPNZ - Repeat Not Equal / Repeat Not Zero . . . . . . . . . 30

RET/RETF - Return From Procedure . . . . . . . . . . . . . . . . . 31

ROL - Rotate Left. . . . . . . . . . . . . . . . . . . . . . . . . 31

ROR - Rotate Right . . . . . . . . . . . . . . . . . . . . . . . . 31

SAHF - Store AH Register into FLAGS. . . . . . . . . . . . . . . . 32

SAL/SHL - Shift Arithmetic Left / Shift Logical Left . . . . . . . 32

SAR - Shift Arithmetic Right . . . . . . . . . . . . . . . . . . . 32

SBB - Subtract with Borrow/Carry . . . . . . . . . . . . . . . . . 33

SCAS - Scan String (Byte, Word or Doubleword) . . . . . . . . . . 33

SETAE/SETNB - Set if Above or Equal / Set if Not Below (386+). . . 33

SETB/SETNAE - Set if Below / Set if Not Above or Equal (386+). . . 33

SETBE/SETNA - Set if Below or Equal / Set if Not Above (386+). . . 34

SETE/SETZ - Set if Equal / Set if Zero (386+). . . . . . . . . . . 34

SETNE/SETNZ - Set if Not Equal / Set if Not Zero (386+). . . . . . 34

SETL/SETNGE - Set if Less / Set if Not Greater or Equal (386+) . . 34

SETGE/SETNL - Set if Greater or Equal / Set if Not Less (386+) . . 35

SETLE/SETNG - Set if Less or Equal / Set if Not greater or Equal (386+) 35

SETG/SETNLE - Set if Greater / Set if Not Less or Equal (386+) . . 35

SETS - Set if Signed (386+). . . . . . . . . . . . . . . . . . . . 35

SETNS - Set if Not Signed (386+) . . . . . . . . . . . . . . . . . 36

SETC - Set if Carry (386+) . . . . . . . . . . . . . . . . . . . . 36

SETNC - Set if Not Carry (386+). . . . . . . . . . . . . . . . . . 36

SETO - Set if Overflow (386+). . . . . . . . . . . . . . . . . . . 36

SETNO - Set if Not Overflow (386+) . . . . . . . . . . . . . . . . 36

SETP/SETPE - Set if Parity / Set if Parity Even (386+). . . . . . 37

SETNP/SETPO - Set if No Parity / Set if Parity Odd (386+). . . . . 37

SGDT - Store Global Descriptor Table (286+ privileged) . . . . . . 37

SIDT - Store Interrupt Descriptor Table (286+ privileged). . . . . 37

SHL - Shift Logical Left . . . . . . . . . . . . . . . . . . . . . 37

SHR - Shift Logical Right. . . . . . . . . . . . . . . . . . . . . 38

SHLD/SHRD - Double Precision Shift (386+). . . . . . . . . . . . . 38

SLDT - Store Local Descriptor Table (286+ privileged). . . . . . . 38

SMSW - Store Machine Status Word (286+ privileged) . . . . . . . . 38

STC - Set Carry. . . . . . . . . . . . . . . . . . . . . . . . . . 39

STD - Set Direction Flag . . . . . . . . . . . . . . . . . . . . . 39

STI - Set Interrupt Flag (Enable Interrupts). . . . . . . . . . . 39

STOS - Store String (Byte, Word or Doubleword). . . . . . . . . . 39

STR - Store Task Register (286+ privileged). . . . . . . . . . . . 39

SUB - Subtract . . . . . . . . . . . . . . . . . . . . . . . . . . 40

TEST - Test For Bit Pattern. . . . . . . . . . . . . . . . . . . . 40

VERR - Verify Read (286+ protected). . . . . . . . . . . . . . . . 40

VERW - Verify Write (286+ protected) . . . . . . . . . . . . . . . 40

WAIT/FWAIT - Event Wait. . . . . . . . . . . . . . . . . . . . . . 41

WBINVD - Write-Back and Invalidate Cache (486+). . . . . . . . . . 41

XCHG - Exchange. . . . . . . . . . . . . . . . . . . . . . . . . . 41

XLAT/XLATB - Translate . . . . . . . . . . . . . . . . . . . . . . 41

XOR - Exclusive OR . . . . . . . . . . . . . . . . . . . . . . . . 42

Intel 8086 Family Architecture

General Purpose Registers Segment Registers

AH/AL AX (EAX) Accumulator CS Code Segment

BH/BL BX (EBX) Base DS Data Segment

CH/CL CX (ECX) Counter SS Stack Segment

DH/DL DX (EDX) Data ES Extra Segment

(FS) 386 and newer

(Exx) indicates 386+ 32 bit register (GS) 386 and newer

Pointer Registers Stack Registers

SI (ESI) Source Index SP (ESP) Stack Pointer

DI (EDI) Destination Index BP (EBP) Base Pointer

IP Instruction Pointer

Status Registers

FLAGS Status Flags (see FLAGS)

Special Registers (386+ only)

CR0 Control Register 0 DR0 Debug Register 0

CR2 Control Register 2 DR1 Debug Register 1

CR3 Control Register 3 DR2 Debug Register 2

DR3 Debug Register 3

TR4 Test Register 4 DR6 Debug Register 6

TR5 Test Register 5 DR7 Debug Register 7

TR6 Test Register 6

TR7 Test Register 7

Register Default Segment Valid Overrides

BP SS DS, ES, CS

SI or DI DS ES, SS, CS

DI strings ES None

SI strings DS ES, SS, CS

- see CPU DETECTING Instruction Timing

Instruction Clock Cycle Calculation

Some instructions require additional clock cycles due to a "Next

Instruction Component" identified by a "+m" in the instruction

clock cycle listings. This is due to the prefetch queue being

purge on a control transfers. Below is the general rule for

calculating "m":

88/86 not applicable

286 "m" is the number of bytes in the next instruction

386 "m" is the number of components in the next instruction

(the instruction coding (each byte), plus the data and

the displacement are all considered components)

8088/8086 Effective Address (EA) Calculation

Description Clock Cycles

Displacement 6

Base or Index (BX,BP,SI,DI) 5

Displacement+(Base or Index) 9

Base+Index (BP+DI,BX+SI) 7

Base+Index (BP+SI,BX+DI) 8

Base+Index+Displacement (BP+DI,BX+SI) 11

Base+Index+Displacement (BP+SI+disp,BX+DI+disp) 12

- add 4 cycles for word operands at odd addresses

- add 2 cycles for segment override

- 80188/80186 timings differ from those of the 8088/8086/80286

Task State Calculation

"TS" is defined as switching from VM/486 or 80286 TSS to one of

the following:

ÚÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ?

? New Task ?

ÃÄÄÄÄÄÄÄÂÄÄÄÄÄÄÄÂÄÄÄÄÄÄÄÂÄÄÄÄÄÄÄÂÄÄÄÄÄÄÄ?

ÚÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ?86 TSS?86 TSS?86 TSS?86 TSS?86 TSS?

? Old Task ?(VM=0)?(VM=1)?(VM=0)?(VM=1)? ?

ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÅÄÄÄÄÄÄÄÅÄÄÄÄÄÄÄÅÄÄÄÄÄÄÄÅÄÄÄÄÄÄÄÅÄÄÄÄÄÄÄ?

386 TSS (VM=0) ? ? ? 309 ? 226 ? 282 ?

ÃÄÄÄÄÄÄÄÅÄÄÄÄÄÄÄÅÄÄÄÄÄÄÄÅÄÄÄÄÄÄÄÅÄÄÄÄÄÄÄ?

386 TSS (VM=1) ? ? ? 314 ? 231 ? 287 ?

ÃÄÄÄÄÄÄÄÅÄÄÄÄÄÄÄÅÄÄÄÄÄÄÄÅÄÄÄÄÄÄÄÅÄÄÄÄÄÄÄ?

386 CPU/286 TSS ? ? ? 307 ? 224 ? 280 ?

ÃÄÄÄÄÄÄÄÅÄÄÄÄÄÄÄÅÄÄÄÄÄÄÄÅÄÄÄÄÄÄÄÅÄÄÄÄÄÄÄ?

486 CPU/286 TSS ? 199 ? 177 ? ? ? 180 ?

ÀÄÄÄÄÄÄÄÁÄÄÄÄÄÄÄÁÄÄÄÄÄÄÄÁÄÄÄÄÄÄÄÁÄÄÄÄÄÄÄ?

Miscellaneous

- all timings are for best case and do not take into account wait

states, instruction alignment, the state of the prefetch queue,

DMA refresh cycles, cache hits/misses or exception processing.

- to convert clocks to nanoseconds divide one microsecond by the

processor speed in MegaHertz:

(1000MHz/(n MHz)) = X nanoseconds

- see 8086 Architecture

FLAGS - Intel 8086 Family Flags Register

?1?0?????????????????

? ????????????????ÀÄÄÄ CF Carry Flag

? ???????????????ÀÄÄÄ 1

? ??????????????ÀÄÄÄ PF Parity Flag

? ?????????????ÀÄÄÄ 0

? ????????????ÀÄÄÄ AF Auxiliary Flag

? ???????????ÀÄÄÄ 0

? ??????????ÀÄÄÄ ZF Zero Flag

? ?????????ÀÄÄÄ SF Sign Flag

? ????????ÀÄÄÄ TF Trap Flag (Single Step)

? ???????ÀÄÄÄ IF Interrupt Flag

? ??????ÀÄÄÄ DF Direction Flag

? ?????ÀÄÄÄ OF Overflow flag

? ???ÀÄÁÄÄÄ IOPL I/O Privilege Level (286+ only)

? ??ÀÄÄÄÄÄ NT Nested Task Flag (286+ only)

? ?ÀÄÄÄÄÄ 0

? ÀÄÄÄÄÄ RF Resume Flag (386+ only)

ÀÄÄÄÄÄ? VM Virtual Mode Flag (386+ only)

- see PUSHF POPF STI CLI STD CLD

MSW - Machine Status Word (286+ only)

?1?0-5?????? Machine Status Word

? ? ????ÀÄÄÄ?Protection Enable (PE)

? ? ???ÀÄÄÄÄÄ Math Present (MP)

? ? ??ÀÄÄÄÄÄ?Emulation (EM)

? ? ?ÀÄÄÄÄÄÄÄ Task Switched (TS)

? ? ÀÄÄÄÄÄÄÄ?Extension Type (ET)

? ÀÄÄÄÄÄÄÄÄÄ?Reserved

ÀÄÄÄÄÄÄÄÄÄÄÄÄÄ Paging (PG)

Bit 0 PE Protection Enable, switches processor between

protected and real mode

Bit 1 MP Math Present, controls function of the WAIT

instruction

Bit 2 EM Emulation, indicates whether coprocessor functions

are to be emulated

Bit 3 TS Task Switched, set and interrogated by coprocessor

on task switches and when interpretting coprocessor

instructions

Bit 4 ET Extension Type, indicates type of coprocessor in

system

Bits 5-30 Reserved

bit 31 PG Paging, indicates whether the processor uses page

tables to translate linear addresses to physical

addresses

- see SMSW LMSW

8086/80186/80286/80386/80486 Instruction Set

AAA - Ascii Adjust for Addition

Usage: AAA

Modifies flags: AF CF (OF,PF,SF,ZF undefined)

Changes contents of AL to valid unpacked decimal. The high order

nibble is zeroed.

Clocks Size

Operands 808x 286 386 486 Bytes

none 8 3 4 3 1

AAD - Ascii Adjust for Division

Usage: AAD

Modifies flags: SF ZF PF (AF,CF,OF undefined)

Used before dividing unpacked decimal numbers. Multiplies AH by

10 and the adds result into AL. Sets AH to zero. This instruction

is also known to have an undocumented behavior.

AL := 10\*AH+AL

AH := 0

Clocks Size

Operands 808x 286 386 486 Bytes

none 60 14 19 14 2

AAM - Ascii Adjust for Multiplication

Usage: AAM

Modifies flags: PF SF ZF (AF,CF,OF undefined)

AH := AL / 10

AL := AL mod 10

Used after multiplication of two unpacked decimal numbers, this

instruction adjusts an unpacked decimal number. The high order

nibble of each byte must be zeroed before using this instruction.

This instruction is also known to have an undocumented behavior.

Clocks Size

Operands 808x 286 386 486 Bytes

none 83 16 17 15 2

AAS - Ascii Adjust for Subtraction

Usage: AAS

Modifies flags: AF CF (OF,PF,SF,ZF undefined)

Corrects result of a previous unpacked decimal subtraction in AL.

High order nibble is zeroed.

Clocks Size

Operands 808x 286 386 486 Bytes

none 8 3 4 3 1

ADC - Add With Carry

Usage: ADC dest,src

Modifies flags: AF CF OF SF PF ZF

Sums two binary operands placing the result in the destination.

If CF is set, a 1 is added to the destination.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,reg 3 2 2 1 2

mem,reg 16+EA 7 7 3 2-4 (W88=24+EA)

reg,mem 9+EA 7 6 2 2-4 (W88=13+EA)

reg,immed 4 3 2 1 3-4

mem,immed 17+EA 7 7 3 3-6 (W88=23+EA)

accum,immed 4 3 2 1 2-3

ADD - Arithmetic Addition

Usage: ADD dest,src

Modifies flags: AF CF OF PF SF ZF

Adds "src" to "dest" and replacing the original contents of "dest".

Both operands are binary.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,reg 3 2 2 1 2

mem,reg 16+EA 7 7 3 2-4 (W88=24+EA)

reg,mem 9+EA 7 6 2 2-4 (W88=13+EA)

reg,immed 4 3 2 1 3-4

mem,immed 17+EA 7 7 3 3-6 (W88=23+EA)

accum,immed 4 3 2 1 2-3

AND - Logical And

Usage: AND dest,src

Modifies flags: CF OF PF SF ZF (AF undefined)

Performs a logical AND of the two operands replacing the destination

with the result.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,reg 3 2 2 1 2

mem,reg 16+EA 7 7 3 2-4 (W88=24+EA)

reg,mem 9+EA 7 6 1 2-4 (W88=13+EA)

reg,immed 4 3 2 1 3-4

mem,immed 17+EA 7 7 3 3-6 (W88=23+EA)

accum,immed 4 3 2 1 2-3

ARPL - Adjusted Requested Privilege Level of Selector (286+ PM)

Usage: ARPL dest,src

(286+ protected mode)

Modifies flags: ZF

Compares the RPL bits of "dest" against "src". If the RPL bits

of "dest" are less than "src", the destination RPL bits are set

equal to the source RPL bits and the Zero Flag is set. Otherwise

the Zero Flag is cleared.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,reg - 10 20 9 2

mem,reg - 11 21 9 4

BOUND - Array Index Bound Check (80188+)

Usage: BOUND src,limit

Modifies flags: None

Array index in source register is checked against upper and lower

bounds in memory source. The first word located at "limit" is

the lower boundary and the word at "limit+2" is the upper array bound.

Interrupt 5 occurs if the source value is less than or higher than

the source.

Clocks Size

Operands 808x 286 386 486 Bytes

reg16,mem32 - nj=13 nj=10 7 2

reg32,mem64 - nj=13 nj=10 7 2

- nj = no jump taken

BSF - Bit Scan Forward (386+)

Usage: BSF dest,src

Modifies flags: ZF

Scans source operand for first bit set. Sets ZF if a bit is found

set and loads the destination with an index to first set bit. Clears

ZF is no bits are found set. BSF scans forward across bit pattern

(0-n) while BSR scans in reverse (n-0).

Clocks Size

Operands 808x 286 386 486 Bytes

reg,reg - - 10+3n 6-42 3

reg,mem - - 10+3n 7-43 3-7

reg32,reg32 - - 10+3n 6-42 3-7

reg32,mem32 - - 10+3n 7-43 3-7

BSR - Bit Scan Reverse (386+)

Usage: BSR dest,src

Modifies flags: ZF

Scans source operand for first bit set. Sets ZF if a bit is found

set and loads the destination with an index to first set bit. Clears

ZF is no bits are found set. BSF scans forward across bit pattern

(0-n) while BSR scans in reverse (n-0).

Clocks Size

Operands 808x 286 386 486 Bytes

reg,reg - - 10+3n 6-103 3

reg,mem - - 10+3n 7-104 3-7

reg32,reg32 - - 10+3n 6-103 3-7

reg32,mem32 - - 10+3n 7-104 3-7

BSWAP - Byte Swap (486+)

Usage: BSWAP reg32

Modifies flags: none

Changes the byte order of a 32 bit register from big endian to

little endian or vice versa. Result left in destination register

is undefined if the operand is a 16 bit register.

Clocks Size

Operands 808x 286 386 486 Bytes

reg32 - - - 1 2

BT - Bit Test (386+)

Usage: BT dest,src

Modifies flags: CF

The destination bit indexed by the source value is copied into the

Carry Flag.

Clocks Size

Operands 808x 286 386 486 Bytes

reg16,immed8 - - 3 3 4-8

mem16,immed8 - - 6 6 4-8

reg16,reg16 - - 3 3 3-7

mem16,reg16 - - 12 12 3-7

BTC - Bit Test with Compliment (386+)

Usage: BTC dest,src

Modifies flags: CF

The destination bit indexed by the source value is copied into the

Carry Flag after being complimented (inverted).

Clocks Size

Operands 808x 286 386 486 Bytes

reg16,immed8 - - 6 6 4-8

mem16,immed8 - - 8 8 4-8

reg16,reg16 - - 6 6 3-7

mem16,reg16 - - 13 13 3-7

BTR - Bit Test with Reset (386+)

Usage: BTR dest,src

Modifies flags: CF

The destination bit indexed by the source value is copied into the

Carry Flag and then cleared in the destination.

Clocks Size

Operands 808x 286 386 486 Bytes

reg16,immed8 - - 6 6 4-8

mem16,immed8 - - 8 8 4-8

reg16,reg16 - - 6 6 3-7

mem16,reg16 - - 13 13 3-7

BTS - Bit Test and Set (386+)

Usage: BTS dest,src

Modifies flags: CF

The destination bit indexed by the source value is copied into the

Carry Flag and then set in the destination.

Clocks Size

Operands 808x 286 386 486 Bytes

reg16,immed8 - - 6 6 4-8

mem16,immed8 - - 8 8 4-8

reg16,reg16 - - 6 6 3-7

mem16,reg16 - - 13 13 3-7

CALL - Procedure Call

Usage: CALL destination

Modifies flags: None

Pushes Instruction Pointer (and Code Segment for far calls) onto

stack and loads Instruction Pointer with the address of proc-name.

Code continues with execution at CS:IP.

Clocks

Operands 808x 286 386 486

rel16 (near, IP relative) 19 7 7+m 3

rel32 (near, IP relative) - - 7+m 3

reg16 (near, register indirect) 16 7 7+m 5

reg32 (near, register indirect) - - 7+m 5

mem16 (near, memory indirect) - 21+EA 11 10+m 5

mem32 (near, memory indirect) - - 10+m 5

ptr16:16 (far, full ptr supplied) 28 13 17+m 18

ptr16:32 (far, full ptr supplied) - - 17+m 18

ptr16:16 (far, ptr supplied, prot. mode) - 26 34+m 20

ptr16:32 (far, ptr supplied, prot. mode) - - 34+m 20

m16:16 (far, indirect) 37+EA 16 22+m 17

m16:32 (far, indirect) - - 22+m 17

m16:16 (far, indirect, prot. mode) - 29 38+m 20

m16:32 (far, indirect, prot. mode) - - 38+m 20

ptr16:16 (task, via TSS or task gate) - 177 TS 37+TS

m16:16 (task, via TSS or task gate) - 180/185 5+TS 37+TS

m16:32 (task) - - TS 37+TS

m16:32 (task) - - 5+TS 37+TS

ptr16:16 (gate, same privilege) - 41 52+m 35

ptr16:32 (gate, same privilege) - - 52+m 35

m16:16 (gate, same privilege) - 44 56+m 35

m16:32 (gate, same privilege) - - 56+m 35

ptr16:16 (gate, more priv, no parm) - 82 86+m 69

ptr16:32 (gate, more priv, no parm) - - 86+m 69

m16:16 (gate, more priv, no parm) - 83 90+m 69

m16:32 (gate, more priv, no parm) - - 90+m 69

ptr16:16 (gate, more priv, x parms) - 86+4x 94+4x+m 77+4x

ptr16:32 (gate, more priv, x parms) - - 94+4x+m 77+4x

m16:16 (gate, more priv, x parms) - 90+4x 98+4x+m 77+4x

m16:32 (gate, more priv, x parms) - - 98+4x+m 77+4x

CBW - Convert Byte to Word

Usage: CBW

Modifies flags: None

Converts byte in AL to word Value in AX by extending sign of AL

throughout register AH.

Clocks Size

Operands 808x 286 386 486 Bytes

none 2 2 3 3 1

CDQ - Convert Double to Quad (386+)

Usage: CDQ

Modifies flags: None

Converts signed DWORD in EAX to a signed quad word in EDX:EAX by

extending the high order bit of EAX throughout EDX

Clocks Size

Operands 808x 286 386 486 Bytes

none - - 2 3 1

CLC - Clear Carry

Usage: CLC

Modifies flags: CF

Clears the Carry Flag.

Clocks Size

Operands 808x 286 386 486 Bytes

none 2 2 2 2 1

CLD - Clear Direction Flag

Usage: CLD

Modifies flags: DF

Clears the Direction Flag causing string instructions to increment

the SI and DI index registers.

Clocks Size

Operands 808x 286 386 486 Bytes

none 2 2 2 2 1

CLI - Clear Interrupt Flag (disable)

Usage: CLI

Modifies flags: IF

Disables the maskable hardware interrupts by clearing the Interrupt

flag. NMI's and software interrupts are not inhibited.

Clocks Size

Operands 808x 286 386 486 Bytes

none 2 2 3 5 1

CLTS - Clear Task Switched Flag (286+ privileged)

Usage: CLTS

Modifies flags: None

Clears the Task Switched Flag in the Machine Status Register. This

is a privileged operation and is generally used only by operating

system code.

Clocks Size

Operands 808x 286 386 486 Bytes

none - 2 5 7 2

CMC - Complement Carry Flag

Usage: CMC

Modifies flags: CF

Toggles (inverts) the Carry Flag

Clocks Size

Operands 808x 286 386 486 Bytes

none 2 2 2 2 1

CMP - Compare

Usage: CMP dest,src

Modifies flags: AF CF OF PF SF ZF

Subtracts source from destination and updates the flags but does

not save result. Flags can subsequently be checked for conditions.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,reg 3 2 2 1 2

mem,reg 9+EA 7 5 2 2-4 (W88=13+EA)

reg,mem 9+EA 6 6 2 2-4 (W88=13+EA)

reg,immed 4 3 2 1 3-4

mem,immed 10+EA 6 5 2 3-6 (W88=14+EA)

accum,immed 4 3 2 1 2-3

CMPS - Compare String (Byte, Word or Doubleword)

Usage: CMPS dest,src

CMPSB

CMPSW

CMPSD (386+)

Modifies flags: AF CF OF PF SF ZF

Subtracts destination value from source without saving results.

Updates flags based on the subtraction and the index registers

(E)SI and (E)DI are incremented or decremented depending on the

state of the Direction Flag. CMPSB inc/decrements the index

registers by 1, CMPSW inc/decrements by 2, while CMPSD increments

or decrements by 4. The REP prefixes can be used to process

entire data items.

Clocks Size

Operands 808x 286 386 486 Bytes

dest,src 22 8 10 8 1 (W88=30)

CMPXCHG - Compare and Exchange

Usage: CMPXCHG dest,src (486+)

Modifies flags: AF CF OF PF SF ZF

Compares the accumulator (8-32 bits) with "dest". If equal the

"dest" is loaded with "src", otherwise the accumulator is loaded

with "dest".

Clocks Size

Operands 808x 286 386 486 Bytes

reg,reg - - - 6 2

mem,reg - - - 7 2

- add 3 clocks if the "mem,reg" comparison fails

CWD - Convert Word to Doubleword

Usage: CWD

Modifies flags: None

Extends sign of word in register AX throughout register DX forming

a doubleword quantity in DX:AX.

Clocks Size

Operands 808x 286 386 486 Bytes

none 5 2 2 3 1

CWDE - Convert Word to Extended Doubleword (386+)

Usage: CWDE

Modifies flags: None

Converts a signed word in AX to a signed doubleword in EAX by

extending the sign bit of AX throughout EAX.

Clocks Size

Operands 808x 286 386 486 Bytes

none - - 3 3 1

DAA - Decimal Adjust for Addition

Usage: DAA

Modifies flags: AF CF PF SF ZF (OF undefined)

Corrects result (in AL) of a previous BCD addition operation.

Contents of AL are changed to a pair of packed decimal digits.

Clocks Size

Operands 808x 286 386 486 Bytes

none 4 3 4 2 1

DAS - Decimal Adjust for Subtraction

Usage: DAS

Modifies flags: AF CF PF SF ZF (OF undefined)

Corrects result (in AL) of a previous BCD subtraction operation.

Contents of AL are changed to a pair of packed decimal digits.

Clocks Size

Operands 808x 286 386 486 Bytes

none 4 3 4 2 1

DEC - Decrement

Usage: DEC dest

Modifies flags: AF OF PF SF ZF

Unsigned binary subtraction of one from the destination.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 3 2 2 1 2

mem 15+EA 7 6 3 2-4

reg16/32 3 2 2 1 1

DIV - Divide

Usage: DIV src

Modifies flags: (AF,CF,OF,PF,SF,ZF undefined)

Unsigned binary division of accumulator by source. If the source

divisor is a byte value then AX is divided by "src" and the quotient

is placed in AL and the remainder in AH. If source operand is a word

value, then DX:AX is divided by "src" and the quotient is stored in AX

and the remainder in DX.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 80-90 14 14 16 2

reg16 144-162 22 22 24 2

reg32 - - 38 40 2

mem8 (86-96)+EA 17 17 16 2-4

mem16 (150-168)+EA 25 25 24 2-4 (W88=158-176+EA)

mem32 - - 41 40 2-4

ENTER - Make Stack Frame (80188+)

Usage: ENTER locals,level

Modifies flags: None

Modifies stack for entry to procedure for high level language.

Operand "locals" specifies the amount of storage to be allocated

on the stack. "Level" specifies the nesting level of the routine.

Paired with the LEAVE instruction, this is an efficient method of

entry and exit to procedures.

Clocks Size

Operands 808x 286 386 486 Bytes

immed16,0 - 11 10 14 4

immed16,1 - 15 12 17 4

immed16,immed8 - 12+4(n-1) 15+4(n-1) 17+3n 4

ESC - Escape

Usage: ESC immed,src

Modifies flags: None

Provides access to the data bus for other resident processors.

The CPU treats it as a NOP but places memory operand on bus.

Clocks Size

Operands 808x 286 386 486 Bytes

immed,reg 2 9-20 ? 2

immed,mem 2 9-20 ? 2-4

HLT - Halt CPU

Usage: HLT

Modifies flags: None

Halts CPU until RESET line is activated, NMI or maskable interrupt

received. The CPU becomes dormant but retains the current CS:IP

for later restart.

Clocks Size

Operands 808x 286 386 486 Bytes

none 2 2 5 4 1

IDIV - Signed Integer Division

Usage: IDIV src

Modifies flags: (AF,CF,OF,PF,SF,ZF undefined)

Signed binary division of accumulator by source. If source is a

byte value, AX is divided by "src" and the quotient is stored in

AL and the remainder in AH. If source is a word value, DX:AX is

divided by "src", and the quotient is stored in AL and the

remainder in DX.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 101-112 17 19 19 2

reg16 165-184 25 27 27 2

reg32 - - 43 43 2

mem8 (107-118)+EA 20 22 20 2-4

mem16 (171-190)+EA 38 30 28 2-4 (W88=175-194)

mem32 - - 46 44 2-4

IMUL - Signed Multiply

Usage: IMUL src

IMUL src,immed (286+)

IMUL dest,src,immed8 (286+)

IMUL dest,src (386+)

Modifies flags: CF OF (AF,PF,SF,ZF undefined)

Signed multiplication of accumulator by "src" with result placed

in the accumulator. If the source operand is a byte value, it

is multiplied by AL and the result stored in AX. If the source

operand is a word value it is multiplied by AX and the result is

stored in DX:AX. Other variations of this instruction allow

specification of source and destination registers as well as a

third immediate factor.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 80-98 13 9-14 13-18 2

reg16 128-154 21 9-22 13-26 2

reg32 - - 9-38 12-42 2

mem8 86-104 16 12-17 13-18 2-4

mem16 134-160 24 12-25 13-26 2-4

mem32 - - 12-41 13-42 2-4

reg16,reg16 - - 9-22 13-26 3-5

reg32,reg32 - - 9-38 13-42 3-5

reg16,mem16 - - 12-25 13-26 3-5

reg32,mem32 - - 12-41 13-42 3-5

reg16,immed - 21 9-22 13-26 3

reg32,immed - 21 9-38 13-42 3-6

reg16,reg16,immed - 2 9-22 13-26 3-6

reg32,reg32,immed - 21 9-38 13-42 3-6

reg16,mem16,immed - 24 12-25 13-26 3-6

reg32,mem32,immed - 24 12-41 13-42 3-6

IN - Input Byte or Word From Port

Usage: IN accum,port

Modifies flags: None

A byte, word or dword is read from "port" and placed in AL, AX or

EAX respectively. If the port number is in the range of 0-255

it can be specified as an immediate, otherwise the port number

must be specified in DX. Valid port ranges on the PC are 0-1024,

though values through 65535 may be specified and recognized by

third party vendors and PS/2's.

Clocks Size

Operands 808x 286 386 486 Bytes

accum,immed8 10/14 5 12 14 2

accum,immed8 (PM) 6/26 8/28/27 2

accum,DX 8/12 5 13 14 1

accum,DX (PM) 7/27 8/28/27 1

- 386+ protected mode timings depend on privilege levels.

first number is the timing if: CPL ?IOPL

second number is the timing if: CPL > IOPL or in VM 86 mode (386)

CPL ?IOPL (486)

third number is the timing when: virtual mode on 486 processor

- 486 virtual mode always requires 27 cycles

INC - Increment

Usage: INC dest

Modifies flags: AF OF PF SF ZF

Adds one to destination unsigned binary operand.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 3 2 2 1 2

reg16 3 2 2 1 1

reg32 3 2 2 1 1

mem 15+EA 7 6 3 2-4 (W88=23+EA)

INS - Input String from Port (80188+)

Usage: INS dest,port

INSB

INSW

INSD (386+)

Modifies flags: None

Loads data from port to the destination ES:(E)DI (even if a

destination operand is supplied). (E)DI is adjusted by the size

of the operand and increased if the Direction Flag is cleared and

decreased if the Direction Flag is set. For INSB, INSW, INSD no

operands are allowed and the size is determined by the mnemonic.

Clocks Size

Operands 808x 286 386 486 Bytes

dest,port - 5 15 17 1

dest,port (PM) - 5 9/29 10/32/30 1

none - 5 15 17 1

none (PM) - 5 9/29 10/32/30 1

- 386+ protected mode timings depend on privilege levels.

first number is the timing if: CPL ?IOPL

second number is the timing if: CPL > IOPL

third number is the timing if: virtual mode on 486 processor

INT - Interrupt

Usage: INT num

Modifies flags: TF IF

Initiates a software interrupt by pushing the flags, clearing the

Trap and Interrupt Flags, pushing CS followed by IP and loading

CS:IP with the value found in the interrupt vector table. Execution

then begins at the location addressed by the new CS:IP

Clocks Size

Operands 808x 286 386 486 Bytes

3 (constant) 52/72 23+m 33 26 2

3 (prot. mode, same priv.) - 40+m 59 44 2

3 (prot. mode, more priv.) - 78+m 99 71 2

3 (from VM86 to PL 0) - - 119 82 2

3 (prot. mode via task gate) - 167+m TS 37+TS 2

immed8 51/71 23+m 37 30 1

immed8 (prot. mode, same priv.) - 40+m 59 44 1

immed8 (prot. mode, more priv.) - 78+m 99 71 1

immed8 (from VM86 to PL 0) - - 119 86 1

immed8 (prot. mode, via task gate) - 167+m TS 37+TS 1

INTO - Interrupt on Overflow

Usage: INTO

Modifies flags: IF TF

If the Overflow Flag is set this instruction generates an INT 4

which causes the code addressed by 0000:0010 to be executed.

Clocks Size

Operands 808x 286 386 486 Bytes

none: jump 53/73 24+m 35 28 1

no jump 4 3 3 3

(prot. mode, same priv.) - - 59 46 1

(prot. mode, more priv.) - - 99 73 1

(from VM86 to PL 0) - - 119 84 1

(prot. mode, via task gate) - TS 39+TS 1

INVD - Invalidate Cache (486+)

Usage: INVD

Modifies flags: none

Flushes CPU internal cache. Issues special function bus cycle

which indicates to flush external caches. Data in write-back

external caches is lost.

Clocks Size

Operands 808x 286 386 486 Bytes

none - - - 4 2

INVLPG - Invalidate Translation Look-Aside Buffer Entry (486+)

Usage: INVLPG

Modifies flags: none

Invalidates a single page table entry in the Translation

Look-Aside Buffer. Intel warns that this instruction may be

implemented differently on future processors.

Clocks Size

Operands 808x 286 386 486 Bytes

none - - - 12 2

- timing is for TLB entry hit only.

IRET/IRETD - Interrupt Return

Usage: IRET

IRETD (386+)

Modifies flags: AF CF DF IF PF SF TF ZF

Returns control to point of interruption by popping IP, CS

and then the Flags from the stack and continues execution at

this location. CPU exception interrupts will return to the

instruction that cause the exception because the CS:IP placed

on the stack during the interrupt is the address of the offending

instruction.

Clocks Size

Operands 808x 286 386 486 Bytes

iret 32/44 17+m 22 15 1

iret (prot. mode) - 31+m 38 15 1

iret (to less privilege) - 55+m 82 36 1

iret (different task, NT=1) - 169+m TS TS+32 1

iretd - - 22/38 15 1

iretd (to less privilege) - - 82 36 1

iretd (to VM86 mode) - - 60 15 1

iretd (different task, NT=1) - - TS TS+32 1

- 386 timings are listed as real-mode/protected-mode

Jxx - Jump Instructions Table

Mnemonic Meaning Jump Condition

JA Jump if Above CF=0 and ZF=0

JAE Jump if Above or Equal CF=0

JB Jump if Below CF=1

JBE Jump if Below or Equal CF=1 or ZF=1

JC Jump if Carry CF=1

JCXZ Jump if CX Zero CX=0

JE Jump if Equal ZF=1

JG Jump if Greater (signed) ZF=0 and SF=OF

JGE Jump if Greater or Equal (signed) SF=OF

JL Jump if Less (signed) SF != OF

JLE Jump if Less or Equal (signed) ZF=1 or SF != OF

JMP Unconditional Jump unconditional

JNA Jump if Not Above CF=1 or ZF=1

JNAE Jump if Not Above or Equal CF=1

JNB Jump if Not Below CF=0

JNBE Jump if Not Below or Equal CF=0 and ZF=0

JNC Jump if Not Carry CF=0

JNE Jump if Not Equal ZF=0

JNG Jump if Not Greater (signed) ZF=1 or SF != OF

JNGE Jump if Not Greater or Equal (signed) SF != OF

JNL Jump if Not Less (signed) SF=OF

JNLE Jump if Not Less or Equal (signed) ZF=0 and SF=OF

JNO Jump if Not Overflow (signed) OF=0

JNP Jump if No Parity PF=0

JNS Jump if Not Signed (signed) SF=0

JNZ Jump if Not Zero ZF=0

JO Jump if Overflow (signed) OF=1

JP Jump if Parity PF=1

JPE Jump if Parity Even PF=1

JPO Jump if Parity Odd PF=0

JS Jump if Signed (signed) SF=1

JZ Jump if Zero ZF=1

Clocks Size

Operands 808x 286 386 486 Bytes

Jx: jump 16 7+m 7+m 3 2

no jump 4 3 3 1

Jx near-label - - 7+m 3 4

no jump - - 3 1

- It's a good programming practice to organize code so the

expected case is executed without a jump since the actual

jump takes longer to execute than falling through the test.

- see JCXZ and JMP for their respective timings

JCXZ/JECXZ - Jump if Register (E)CX is Zero

Usage: JCXZ label

JECXZ label (386+)

Modifies flags: None

Causes execution to branch to "label" if register CX is zero. Uses

unsigned comparision.

Clocks Size

Operands 808x 286 386 486 Bytes

label: jump 18 8+m 9+m 8 2

no jump 6 4 5 5

JMP - Unconditional Jump

Usage: JMP target

Modifies flags: None

Unconditionally transfers control to "label". Jumps by default

are within -32768 to 32767 bytes from the instruction following

the jump. NEAR and SHORT jumps cause the IP to be updated while FAR

jumps cause CS and IP to be updated.

Clocks

Operands 808x 286 386 486

rel8 (relative) 15 7+m 7+m 3

rel16 (relative) 15 7+m 7+m 3

rel32 (relative) - - 7+m 3

reg16 (near, register indirect) 11 7+m 7+m 5

reg32 (near, register indirect) - - 7+m 5

mem16 (near, mem indirect) 18+EA 11+m 10+m 5

mem32 (near, mem indirect) 24+EA 15+m 10+m 5

ptr16:16 (far, dword immed) - - 12+m 17

ptr16:16 (far, PM dword immed) - - 27+m 19

ptr16:16 (call gate, same priv.) - 38+m 45+m 32

ptr16:16 (via TSS) - 175+m TS 42+TS

ptr16:16 (via task gate) - 180+m TS 43+TS

mem16:16 (far, indirect) - - 43+m 13

mem16:16 (far, PM indirect) - - 31+m 18

mem16:16 (call gate, same priv.) - 41+m 49+m 31

mem16:16 (via TSS) - 178+m 5+TS 41+TS

mem16:16 (via task gate) - 183+m 5+TS 42+TS

ptr16:32 (far, 6 byte immed) - - 12+m 13

ptr16:32 (far, PM 6 byte immed) - - 27+m 18

ptr16:32 (call gate, same priv.) - - 45+m 31

ptr16:32 (via TSS) - - TS 42+TS

ptr16:32 (via task state) - - TS 43+TS

m16:32 (far, address at dword) - - 43+m 13

m16:32 (far, address at dword) - - 31+m 18

m16:32 (call gate, same priv.) - - 49+m 31

m16:32 (via TSS) - - 5+TS 41+TS

m16:32 (via task state) - - 5+TS 42+TS

LAHF - Load Register AH From Flags

Usage: LAHF

Modifies flags: None

Copies bits 0-7 of the flags register into AH. This includes flags

AF, CF, PF, SF and ZF other bits are undefined.

AH := SF ZF xx AF xx PF xx CF

Clocks Size

Operands 808x 286 386 486 Bytes

none 4 2 2 3 1

LAR - Load Access Rights (286+ protected)

Usage: LAR dest,src

Modifies flags: ZF

The high byte of the of the destination register is overwritten by

the value of the access rights byte and the low order byte is zeroed

depending on the selection in the source operand. The Zero Flag is

set if the load operation is successful.

Clocks Size

Operands 808x 286 386 486 Bytes

reg16,reg16 - 14 15 11 3

reg32,reg32 - - 15 11 3

reg16,mem16 - 16 16 11 3-7

reg32,mem32 - - 16 11 3-7

LDS - Load Pointer Using DS

Usage: LDS dest,src

Modifies flags: None

Loads 32-bit pointer from memory source to destination register

and DS. The offset is placed in the destination register and the

segment is placed in DS. To use this instruction the word at the

lower memory address must contain the offset and the word at the

higher address must contain the segment. This simplifies the loading

of far pointers from the stack and the interrupt vector table.

Clocks Size

Operands 808x 286 386 486 Bytes

reg16,mem32 16+EA 7 7 6 2-4

reg,mem (PM) - - 22 12 5-7

LEA - Load Effective Address

Usage: LEA dest,src

Modifies flags: None

Transfers offset address of "src" to the destination register.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,mem 2+EA 3 2 1 2-4

- the MOV instruction can often save clock cycles when used in

place of LEA on 8088 processors

LEAVE - Restore Stack for Procedure Exit (80188+)

Usage: LEAVE

Modifies flags: None

Releases the local variables created by the previous ENTER

instruction by restoring SP and BP to their condition before

the procedure stack frame was initialized.

Clocks Size

Operands 808x 286 386 486 Bytes

none - 5 4 5 1

LES - Load Pointer Using ES

Usage: LES dest,src

Modifies flags: None

Loads 32-bit pointer from memory source to destination register

and ES. The offset is placed in the destination register and the

segment is placed in ES. To use this instruction the word at the

lower memory address must contain the offset and the word at the

higher address must contain the segment. This simplifies the loading

of far pointers from the stack and the interrupt vector table.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,mem 16+EA 7 7 6 2-4 (W88=24+EA)

reg,mem (PM) - - 22 12 5-7

LFS - Load Pointer Using FS (386+)

Usage: LFS dest,src

Modifies flags: None

Loads 32-bit pointer from memory source to destination register

and FS. The offset is placed in the destination register and the

segment is placed in FS. To use this instruction the word at the

lower memory address must contain the offset and the word at the

higher address must contain the segment. This simplifies the loading

of far pointers from the stack and the interrupt vector table.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,mem - - 7 6 5-7

reg,mem (PM) - - 22 12 5-7

LGDT - Load Global Descriptor Table (286+ privileged)

Usage: LGDT src

Modifies flags: None

Loads a value from an operand into the Global Descriptor Table

(GDT) register.

Clocks Size

Operands 808x 286 386 486 Bytes

mem64 - 11 11 11 5

LIDT - Load Interrupt Descriptor Table (286+ privileged)

Usage: LIDT src

Modifies flags: None

Loads a value from an operand into the Interrupt Descriptor Table

(IDT) register.

Clocks Size

Operands 808x 286 386 486 Bytes

mem64 - 12 11 11 5

LGS - Load Pointer Using GS (386+)

Usage: LGS dest,src

Modifies flags: None

Loads 32-bit pointer from memory source to destination register

and GS. The offset is placed in the destination register and the

segment is placed in GS. To use this instruction the word at the

lower memory address must contain the offset and the word at the

higher address must contain the segment. This simplifies the loading

of far pointers from the stack and the interrupt vector table.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,mem - - 7 6 5-7

reg,mem (PM) - - 22 12 5-7

LLDT - Load Local Descriptor Table (286+ privileged)

Usage: LLDT src

Modifies flags: None

Loads a value from an operand into the Local Descriptor Table

Register (LDTR).

Clocks Size

Operands 808x 286 386 486 Bytes

reg16 - 17 20 11 3

mem16 - 19 24 11 5

LMSW - Load Machine Status Word (286+ privileged)

Usage: LMSW src

Modifies flags: None

Loads the Machine Status Word (MSW) from data found at "src"

Clocks Size

Operands 808x 286 386 486 Bytes

reg16 - 3 10 13 3

mem16 - 6 13 13 5

LOCK - Lock Bus

Usage: LOCK

LOCK: (386+ prefix)

Modifies flags: None

This instruction is a prefix that causes the CPU assert bus lock

signal during the execution of the next instruction. Used to

avoid two processors from updating the same data location. The

286 always asserts lock during an XCHG with memory operands. This

should only be used to lock the bus prior to XCHG, MOV, IN and

OUT instructions.

Clocks Size

Operands 808x 286 386 486 Bytes

none 2 0 0 1 1

LODS - Load String (Byte, Word or Double)

Usage: LODS src

LODSB

LODSW

LODSD (386+)

Modifies flags: None

Transfers string element addressed by DS:SI (even if an operand is

supplied) to the accumulator. SI is incremented based on the size

of the operand or based on the instruction used. If the Direction

Flag is set SI is decremented, if the Direction Flag is clear SI

is incremented. Use with REP prefixes.

Clocks Size

Operands 808x 286 386 486 Bytes

src 12/16 5 5 5 1

LOOP - Decrement CX and Loop if CX Not Zero

Usage: LOOP label

Modifies flags: None

Decrements CX by 1 and transfers control to "label" if CX is not

Zero. The "label" operand must be within -128 or 127 bytes of the

instruction following the loop instruction

Clocks Size

Operands 808x 286 386 486 Bytes

label: jump 18 8+m 11+m 6 2

no jump 5 4 ? 2

LOOPE/LOOPZ - Loop While Equal / Loop While Zero

Usage: LOOPE label

LOOPZ label

Modifies flags: None

Decrements CX by 1 (without modifying the flags) and transfers

control to "label" if CX != 0 and the Zero Flag is set. The

"label" operand must be within -128 or 127 bytes of the instruction

following the loop instruction.

Clocks Size

Operands 808x 286 386 486 Bytes

label: jump 18 8+m 11+m 9 2

no jump 5 4 ? 6

LOOPNZ/LOOPNE - Loop While Not Zero / Loop While Not Equal

Usage: LOOPNZ label

LOOPNE label

Modifies flags: None

Decrements CX by 1 (without modifying the flags) and transfers

control to "label" if CX != 0 and the Zero Flag is clear. The

"label" operand must be within -128 or 127 bytes of the instruction

following the loop instruction.

Clocks Size

Operands 808x 286 386 486 Bytes

label: jump 19 8+m 11+m 9 2

no jump 5 4 ? 6

LSL - Load Segment Limit (286+ protected)

Usage: LSL dest,src

Modifies flags: ZF

Loads the segment limit of a selector into the destination register

if the selector is valid and visible at the current privilege level.

If loading is successful the Zero Flag is set, otherwise it is

cleared.

Clocks Size

Operands 808x 286 386 486 Bytes

reg16,reg16 - 14 20/25 10 3

reg32,reg32 - - 20/25 10 3

reg16,mem16 - 16 21/26 10 5

reg32,mem32 - - 21/26 10 5

- 386 times are listed "byte granular" / "page granular"

LSS - Load Pointer Using SS (386+)

Usage: LSS dest,src

Modifies flags: None

Loads 32-bit pointer from memory source to destination register

and SS. The offset is placed in the destination register and the

segment is placed in SS. To use this instruction the word at the

lower memory address must contain the offset and the word at the

higher address must contain the segment. This simplifies the loading

of far pointers from the stack and the interrupt vector table.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,mem - - 7 6 5-7

reg,mem (PM) - - 22 12 5-7

LTR - Load Task Register (286+ privileged)

Usage: LTR src

Modifies flags: None

Loads the current task register with the value specified in "src".

Clocks Size

Operands 808x 286 386 486 Bytes

reg16 - 17 23 20 3

mem16 - 19 27 20 5

MOV - Move Byte or Word

Usage: MOV dest,src

Modifies flags: None

Copies byte or word from the source operand to the destination

operand. If the destination is SS interrupts are disabled except

on early buggy 808x CPUs. Some CPUs disable interrupts if the

destination is any of the segment registers

Clocks Size

Operands 808x 286 386 486 Bytes

reg,reg 2 2 2 1 2

mem,reg 9+EA 3 2 1 2-4 (W88=13+EA)

reg,mem 8+EA 5 4 1 2-4 (W88=12+EA)

mem,immed 10+EA 3 2 1 3-6 (W88=14+EA)

reg,immed 4 2 2 1 2-3

mem,accum 10 3 2 1 3 (W88=14)

accum,mem 10 5 4 1 3 (W88=14)

segreg,reg16 2 2 2 3 2

segreg,mem16 8+EA 5 5 9 2-4 (W88=12+EA)

reg16,segreg 2 2 2 3 2

mem16,segreg 9+EA 3 2 3 2-4 (W88=13+EA)

reg32,CR0/CR2/CR3 - - 6 4

CR0,reg32 - - 10 16

CR2,reg32 - - 4 4 3

CR3,reg32 - - 5 4 3

reg32,DR0/DR1/DR2/DR3 - 22 10 3

reg32,DR6/DR7 - - 22 10 3

DR0/DR1/DR2/DR3,reg32 - 22 11 3

DR6/DR7,reg32 - - 16 11 3

reg32,TR6/TR7 - - 12 4 3

TR6/TR7,reg32 - - 12 4 3

reg32,TR3 3

TR3,reg32 6

- when the 386 special registers are used all operands are 32 bits

MOVS - Move String (Byte or Word)

Usage: MOVS dest,src

MOVSB

MOVSW

MOVSD (386+)

Modifies flags: None

Copies data from addressed by DS:SI (even if operands are given) to

the location ES:DI destination and updates SI and DI based on the

size of the operand or instruction used. SI and DI are incremented

when the Direction Flag is cleared and decremented when the Direction

Flag is Set. Use with REP prefixes.

Clocks Size

Operands 808x 286 386 486 Bytes

dest,src 18 5 7 7 1 (W88=26)

MOVSX - Move with Sign Extend (386+)

Usage: MOVSX dest,src

Modifies flags: None

Copies the value of the source operand to the destination register

with the sign extended.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,reg - - 3 3 3

reg,mem - - 6 3 3-7

MOVZX - Move with Zero Extend (386+)

Usage: MOVZX dest,src

Modifies flags: None

Copies the value of the source operand to the destination register

with the zeroes extended.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,reg - - 3 3 3

reg,mem - - 6 3 3-7

MUL - Unsigned Multiply

Usage: MUL src

Modifies flags: CF OF (AF,PF,SF,ZF undefined)

Unsigned multiply of the accumulator by the source. If "src" is

a byte value, then AL is used as the other multiplicand and the

result is placed in AX. If "src" is a word value, then AX is

multiplied by "src" and DX:AX receives the result. If "src" is

a double word value, then EAX is multiplied by "src" and EDX:EAX

receives the result. The 386+ uses an early out algorithm which

makes multiplying any size value in EAX as fast as in the 8 or 16

bit registers.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 70-77 13 9-14 13-18 2

reg16 118-113 21 9-22 13-26 2

reg32 - - 9-38 13-42 2-4

mem8 (76-83)+EA 16 12-17 13-18 2-4

mem16 (124-139)+EA 24 12-25 13-26 2-4

mem32 - - 12-21 13-42 2-4

NEG - Two's Complement Negation

Usage: NEG dest

Modifies flags: AF CF OF PF SF ZF

Subtracts the destination from 0 and saves the 2s complement of

"dest" back into "dest".

Clocks Size

Operands 808x 286 386 486 Bytes

reg 3 2 2 1 2

mem 16+EA 7 6 3 2-4 (W88=24+EA)

NOP - No Operation (90h)

Usage: NOP

Modifies flags: None

This is a do nothing instruction. It results in occupation of both

space and time and is most useful for patching code segments.

(This is the original XCHG AL,AL instruction)

Clocks Size

Operands 808x 286 386 486 Bytes

none 3 3 3 1 1

NOT - One's Compliment Negation (Logical NOT)

Usage: NOT dest

Modifies flags: None

Inverts the bits of the "dest" operand forming the 1s complement.

Clocks Size

Operands 808x 286 386 486 Bytes

reg 3 2 2 1 2

mem 16+EA 7 6 3 2-4 (W88=24+EA)

OR - Inclusive Logical OR

Usage: OR dest,src

Modifies flags: CF OF PF SF ZF (AF undefined)

Logical inclusive OR of the two operands returning the result in

the destination. Any bit set in either operand will be set in the

destination.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,reg 3 2 2 1 2

mem,reg 16+EA 7 7 3 2-4 (W88=24+EA)

reg,mem 9+EA 7 6 2 2-4 (W88=13+EA)

reg,immed 4 3 2 1 3-4

mem8,immed8 17+EA 7 7 3 3-6

mem16,immed16 25+EA 7 7 3 3-6

accum,immed 4 3 2 1 2-3

OUT - Output Data to Port

Usage: OUT port,accum

Modifies flags: None

Transfers byte in AL,word in AX or dword in EAX to the specified

hardware port address. If the port number is in the range of 0-255

it can be specified as an immediate. If greater than 255 then the

port number must be specified in DX. Since the PC only decodes 10

bits of the port address, values over 1023 can only be decoded by

third party vendor equipment and also map to the port range 0-1023.

Clocks Size

Operands 808x 286 386 486 Bytes

immed8,accum 10/14 3 10 16 2

immed8,accum (PM) - - 4/24 11/31/29 2

DX,accum 8/12 3 11 16 1

DX,accum (PM) - - 5/25 10/30/29 1

- 386+ protected mode timings depend on privilege levels.

first number is the timing when: CPL ?IOPL

second number is the timing when: CPL > IOPL

third number is the timing when: virtual mode on 486 processor

OUTS - Output String to Port (80188+)

Usage: OUTS port,src

OUTSB

OUTSW

OUTSD (386+)

Modifies flags: None

Transfers a byte, word or doubleword from "src" to the hardware

port specified in DX. For instructions with no operands the "src"

is located at DS:SI and SI is incremented or decremented by the

size of the operand or the size dictated by the instruction format.

When the Direction Flag is set SI is decremented, when clear, SI is

incremented. If the port number is in the range of 0-255 it can

be specified as an immediate. If greater than 255 then the port

number must be specified in DX. Since the PC only decodes 10 bits

of the port address, values over 1023 can only be decoded by third

party vendor equipment and also map to the port range 0-1023.

Clocks Size

Operands 808x 286 386 486 Bytes

port,src - 5 14 17 1

port,src (PM) - - 8/28 10/32/30 1

- 386+ protected mode timings depend on privilege levels.

first number is the timing when: CPL ?IOPL

second number is the timing when: CPL > IOPL

third number is the timing when: virtual mode on 486 processor

POP - Pop Word off Stack

Usage: POP dest

Modifies flags: None

Transfers word at the current stack top (SS:SP) to the destination

then increments SP by two to point to the new stack top. CS is not

a valid destination.

Clocks Size

Operands 808x 286 386 486 Bytes

reg16 8 5 4 4 1

reg32 4 - - 4 1

segreg 8 5 7 3 1

mem16 17+EA 5 5 6 2-4

mem32 5 - - 6 2-4

POPA/POPAD - Pop All Registers onto Stack (80188+)

Usage: POPA

POPAD (386+)

Modifies flags: None

Pops the top 8 words off the stack into the 8 general purpose 16/32

bit registers. Registers are popped in the following order: (E)DI,

(E)SI, (E)BP, (E)SP, (E)DX, (E)CX and (E)AX. The (E)SP value popped

from the stack is actually discarded.

Clocks Size

Operands 808x 286 386 486 Bytes

none - 19 24 9 1

POPF/POPFD - Pop Flags off Stack

Usage: POPF

POPFD (386+)

Modifies flags: all flags

Pops word/doubleword from stack into the Flags Register and then

increments SP by 2 (for POPF) or 4 (for POPFD).

Clocks Size

Operands 808x 286 386 486 Bytes

none 8/12 5 5 9 1 (W88=12)

none (PM) - - 5 6 1

PUSH - Push Word onto Stack

Usage: PUSH src

PUSH immed (80188+ only)

Modifies flags: None

Decrements SP by the size of the operand (two or four, byte values

are sign extended) and transfers one word from source to the stack

top (SS:SP).

Clocks Size

Operands 808x 286 386 486 Bytes

reg16 11/15 3 2 1 1

reg32 - - 2 1 1

mem16 16+EA 5 5 4 2-4 (W88=24+EA)

mem32 - - 5 4 2-4

segreg 10/14 3 2 3 1

immed - 3 2 1 2-3

PUSHA/PUSHAD - Push All Registers onto Stack (80188+)

Usage: PUSHA

PUSHAD (386+)

Modifies flags: None

Pushes all general purpose registers onto the stack in the following

order: (E)AX, (E)CX, (E)DX, (E)BX, (E)SP, (E)BP, (E)SI, (E)DI. The

value of SP is the value before the actual push of SP.

Clocks Size

Operands 808x 286 386 486 Bytes

none - 19 24 11 1

PUSHF/PUSHFD - Push Flags onto Stack

Usage: PUSHF

PUSHFD (386+)

Modifies flags: None

Transfers the Flags Register onto the stack. PUSHF saves a 16 bit

value while PUSHFD saves a 32 bit value.

Clocks Size

Operands 808x 286 386 486 Bytes

none 10/14 3 4 4 1

none (PM) - - 4 3 1

RCL - Rotate Through Carry Left

Usage: RCL dest,count

Modifies flags: CF OF

ÚÄ? ÚÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ?

ÚÄþ??þÄÄþ? <ÄÄÄÄÄÄÄÄÄÄ 0?Ä¿

? ÀÄ? ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ? ?

ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ?

Rotates the bits in the destination to the left "count" times with

all data pushed out the left side re-entering on the right. The

Carry Flag holds the last bit rotated out.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,1 2 2 9 3 2

mem,1 15+EA 7 10 4 2-4 (W88=23+EA)

reg,CL 8+4n 5+n 9 8-30 2

mem,CL 20+EA+4n 8+n 10 9-31 2-4 (W88=28+EA+4n)

reg,immed8 - 5+n 9 8-30 3

mem,immed8 - 8+n 10 9-31 3-5

RCR - Rotate Through Carry Right

Usage: RCR dest,count

Modifies flags: CF OF

ÚÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ? ÚÄ?

ÚÄ>? þÄÄÄÄÄÄÄÄ? 0³þÄÄ??³þÄ¿

? ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ? ÀÄ? ?

ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ?

Rotates the bits in the destination to the right "count" times with

all data pushed out the right side re-entering on the left. The

Carry Flag holds the last bit rotated out.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,1 2 2 9 3 2

mem,1 15+EA 7 10 4 2-4 (W88=23+EA)

reg,CL 8+4n 5+n 9 8-30 2

mem,CL 20+EA+4n 8+n 10 9-31 2-4 (W88=28+EA+4n)

reg,immed8 - 5+n 9 8-30 3

mem,immed8 - 8+n 10 9-31 3-5

REP - Repeat String Operation

Usage: REP

Modifies flags: None

Repeats execution of string instructions while CX != 0. After

each string operation, CX is decremented and the Zero Flag is

tested. The combination of a repeat prefix and a segment override

on CPU's before the 386 may result in errors if an interrupt occurs

before CX=0. The following code shows code that is susceptible to

this and how to avoid it:

again: rep movs byte ptr ES:[DI],ES:[SI] ; vulnerable instr.

jcxz next ; continue if REP successful

loop again ; interrupt goofed count

next:

Clocks Size

Operands 808x 286 386 486 Bytes

none 2 2 2 1

REPE/REPZ - Repeat Equal / Repeat Zero

Usage: REPE

REPZ

Modifies flags: None

Repeats execution of string instructions while CX != 0 and the Zero

Flag is set. CX is decremented and the Zero Flag tested after

each string operation. The combination of a repeat prefix and a

segment override on processors other than the 386 may result in

errors if an interrupt occurs before CX=0.

Clocks Size

Operands 808x 286 386 486 Bytes

none 2 2 2 1

REPNE/REPNZ - Repeat Not Equal / Repeat Not Zero

Usage: REPNE

REPNZ

Modifies flags: None

Repeats execution of string instructions while CX != 0 and the Zero

Flag is clear. CX is decremented and the Zero Flag tested after

each string operation. The combination of a repeat prefix and a

segment override on processors other than the 386 may result in

errors if an interrupt occurs before CX=0.

Clocks Size

Operands 808x 286 386 486 Bytes

none 2 2 2 1

RET/RETF - Return From Procedure

Usage: RET nBytes

RETF nBytes

RETN nBytes

Modifies flags: None

Transfers control from a procedure back to the instruction address

saved on the stack. "n bytes" is an optional number of bytes to

release. Far returns pop the IP followed by the CS, while near

returns pop only the IP register.

Clocks Size

Operands 808x 286 386 486 Bytes

retn 16/20 11+m 10+m 5 1

retn immed 20/24 11+m 10+m 5 3

retf 26/34 15+m 18+m 13 1

retf (PM, same priv.) - 32+m 18 1

retf (PM, lesser priv.) - 68 33 1

retf immed 25/33 15+m 18+m 14 3

retf immed (PM, same priv.) 32+m 17 1

retf immed (PM, lesser priv.) 68 33 1

ROL - Rotate Left

Usage: ROL dest,count

Modifies flags: CF OF

ÚÄ? ÚÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ?

??þÂÄþ? <ÄÄÄÄÄÄÄÄÄÄ 0?Ä¿

ÀÄ? ? ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ? ?

ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ?

Rotates the bits in the destination to the left "count" times with

all data pushed out the left side re-entering on the right. The

Carry Flag will contain the value of the last bit rotated out.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,1 2 2 3 3 2

mem,1 15+EA 7 7 4 2-4 (W88=23+EA)

reg,CL 8+4n 5+n 3 3 2

mem,CL 20+EA+4n 8+n 7 4 2-4 (W88=28+EA+4n)

reg,immed8 - 5+n 3 2 3

mem,immed8 - 8+n 7 4 3-5

ROR - Rotate Right

Usage: ROR dest,count

Modifies flags: CF OF

ÚÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ? ÚÄ?

ÚÄ>? þÄÄÄÄÄÄÄÄ? 0³þÄÂ???

? ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ? ? ÀÄ?

ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ?

Rotates the bits in the destination to the right "count" times with

all data pushed out the right side re-entering on the left. The

Carry Flag will contain the value of the last bit rotated out.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,1 2 2 3 3 2

mem,1 15+EA 7 7 4 2-4 (W88=23+EA)

reg,CL 8+4n 5+n 3 3 2

mem,CL 20+EA+4n 8+n 7 4 2-4 (W88=28+EA+4n)

reg,immed8 - 5+n 3 2 3

mem,immed8 - 8+n 7 4 3-5

SAHF - Store AH Register into FLAGS

Usage: SAHF

Modifies flags: AF CF PF SF ZF

Transfers bits 0-7 of AH into the Flags Register. This includes

AF, CF, PF, SF and ZF.

Clocks Size

Operands 808x 286 386 486 Bytes

none 4 2 3 2 1

SAL/SHL - Shift Arithmetic Left / Shift Logical Left

Usage: SAL dest,count

SHL dest,count

Modifies flags: CF OF PF SF ZF (AF undefined)

ÚÄ? ÚÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ? ÚÄ?

??ÄÄÄþ? <ÄÄÄÄÄÄÄÄÄÄ 0?ÄÄÄþ??

ÀÄ? ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ? ÀÄ?

Shifts the destination left by "count" bits with zeroes shifted

in on right. The Carry Flag contains the last bit shifted out.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,1 2 2 3 3 2

mem,1 15+EA 7 7 4 2-4 (W88=23+EA)

reg,CL 8+4n 5+n 3 3 2

mem,CL 20+EA+4n 8+n 7 4 2-4 (W88=28+EA+4n)

reg,immed8 - 5+n 3 2 3

mem,immed8 - 8+n 7 4 3-5

SAR - Shift Arithmetic Right

Usage: SAR dest,count

Modifies flags: CF OF PF SF ZF (AF undefined)

ÚÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ? ÚÄ?

ÚÄþ? ÄÄÄÄÄÄÄÄÄÄ> 0³ÄÄÄþ>??

? ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ? ÀÄ?

ÀÄÄÄ^

Shifts the destination right by "count" bits with the current sign

bit replicated in the leftmost bit. The Carry Flag contains the

last bit shifted out.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,1 2 2 3 3 2

mem,1 15+EA 7 7 4 2-4 (W88=23+EA)

reg,CL 8+4n 5+n 3 3 2

mem,CL 20+EA+4n 8+n 7 4 2-4 (W88=28+EA+4n)

reg,immed8 - 5+n 3 2 3

mem,immed8 - 8+n 7 4 3-5

SBB - Subtract with Borrow/Carry

Usage: SBB dest,src

Modifies flags: AF CF OF PF SF ZF

Subtracts the source from the destination, and subtracts 1 extra if

the Carry Flag is set. Results are returned in "dest".

Clocks Size

Operands 808x 286 386 486 Bytes

reg,reg 3 2 2 1 2

mem,reg 16+EA 7 6 3 2-4 (W88=24+EA)

reg,mem 9+EA 7 7 2 2-4 (W88=13+EA)

reg,immed 4 3 2 1 3-4

mem,immed 17+EA 7 7 3 3-6 (W88=25+EA)

accum,immed 4 3 2 1 2-3

SCAS - Scan String (Byte, Word or Doubleword)

Usage: SCAS string

SCASB

SCASW

SCASD (386+)

Modifies flags: AF CF OF PF SF ZF

Compares value at ES:DI (even if operand is specified) from the

accumulator and sets the flags similar to a subtraction. DI is

incremented/decremented based on the instruction format (or

operand size) and the state of the Direction Flag. Use with REP

prefixes.

Clocks Size

Operands 808x 286 386 486 Bytes

string 15 7 7 6 1 (W88=19)

SETAE/SETNB - Set if Above or Equal / Set if Not Below (386+)

Usage: SETAE dest

SETNB dest

(unsigned, 386+)

Modifies flags: none

Sets the byte in the operand to 1 if the Carry Flag is clear

otherwise sets the operand to 0.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 - - 4 3 3

mem8 - - 5 4 3

SETB/SETNAE - Set if Below / Set if Not Above or Equal (386+)

Usage: SETB dest

SETNAE dest

(unsigned, 386+)

Modifies flags: none

Sets the byte in the operand to 1 if the Carry Flag is set

otherwise sets the operand to 0.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 - - 4 3 3

mem8 - - 5 4 3

SETBE/SETNA - Set if Below or Equal / Set if Not Above (386+)

Usage: SETBE dest

SETNA dest

(unsigned, 386+)

Modifies flags: none

Sets the byte in the operand to 1 if the Carry Flag or the Zero

Flag is set, otherwise sets the operand to 0.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 - - 4 3 3

mem8 - - 5 4 3

SETE/SETZ - Set if Equal / Set if Zero (386+)

Usage: SETE dest

SETZ dest

Modifies flags: none

Sets the byte in the operand to 1 if the Zero Flag is set,

otherwise sets the operand to 0.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 - - 4 3 3

mem8 - - 5 4 3

SETNE/SETNZ - Set if Not Equal / Set if Not Zero (386+)

Usage: SETNE dest

SETNZ dest

Modifies flags: none

Sets the byte in the operand to 1 if the Zero Flag is clear,

otherwise sets the operand to 0.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 - - 4 3 3

mem8 - - 5 4 3

SETL/SETNGE - Set if Less / Set if Not Greater or Equal (386+)

Usage: SETL dest

SETNGE dest

(signed, 386+)

Modifies flags: none

Sets the byte in the operand to 1 if the Sign Flag is not equal

to the Overflow Flag, otherwise sets the operand to 0.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 - - 4 3 3

mem8 - - 5 4 3

SETGE/SETNL - Set if Greater or Equal / Set if Not Less (386+)

Usage: SETGE dest

SETNL dest

(signed, 386+)

Modifies flags: none

Sets the byte in the operand to 1 if the Sign Flag equals the

Overflow Flag, otherwise sets the operand to 0.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 - - 4 3 3

mem8 - - 5 4 3

SETLE/SETNG - Set if Less or Equal / Set if Not greater or Equal (386+)

Usage: SETLE dest

SETNG dest

(signed, 386+)

Modifies flags: none

Sets the byte in the operand to 1 if the Zero Flag is set or the

Sign Flag is not equal to the Overflow Flag, otherwise sets the

operand to 0.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 - - 4 3 3

mem8 - - 5 4 3

SETG/SETNLE - Set if Greater / Set if Not Less or Equal (386+)

Usage: SETG dest

SETNLE dest

(signed, 386+)

Modifies flags: none

Sets the byte in the operand to 1 if the Zero Flag is clear or the

Sign Flag equals to the Overflow Flag, otherwise sets the operand

to 0.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 - - 4 3 3

mem8 - - 5 4 3

SETS - Set if Signed (386+)

Usage: SETS dest

Modifies flags: none

Sets the byte in the operand to 1 if the Sign Flag is set, otherwise

sets the operand to 0.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 - - 4 3 3

mem8 - - 5 4 3

SETNS - Set if Not Signed (386+)

Usage: SETNS dest

Modifies flags: none

Sets the byte in the operand to 1 if the Sign Flag is clear,

otherwise sets the operand to 0.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 - - 4 3 3

mem8 - - 5 4 3

SETC - Set if Carry (386+)

Usage: SETC dest

Modifies flags: none

Sets the byte in the operand to 1 if the Carry Flag is set,

otherwise sets the operand to 0.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 - - 4 3 3

mem8 - - 5 4 3

SETNC - Set if Not Carry (386+)

Usage: SETNC dest

Modifies flags: none

Sets the byte in the operand to 1 if the Carry Flag is clear,

otherwise sets the operand to 0.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 - - 4 3 3

mem8 - - 5 4 3

SETO - Set if Overflow (386+)

Usage: SETO dest

Modifies flags: none

Sets the byte in the operand to 1 if the Overflow Flag is set,

otherwise sets the operand to 0.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 - - 4 3 3

mem8 - - 5 4 3

SETNO - Set if Not Overflow (386+)

Usage: SETNO dest

Modifies flags: none

Sets the byte in the operand to 1 if the Overflow Flag is clear,

otherwise sets the operand to 0.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 - - 4 3 3

mem8 - - 5 4 3

SETP/SETPE - Set if Parity / Set if Parity Even (386+)

Usage: SETP dest

SETPE dest

Modifies flags: none

Sets the byte in the operand to 1 if the Parity Flag is set,

otherwise sets the operand to 0.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 - - 4 3 3

mem8 - - 5 4 3

SETNP/SETPO - Set if No Parity / Set if Parity Odd (386+)

Usage: SETNP dest

SETPO dest

Modifies flags: none

Sets the byte in the operand to 1 if the Parity Flag is clear,

otherwise sets the operand to 0.

Clocks Size

Operands 808x 286 386 486 Bytes

reg8 - - 4 3 3

mem8 - - 5 4 3

SGDT - Store Global Descriptor Table (286+ privileged)

Usage: SGDT dest

Modifies flags: none

Stores the Global Descriptor Table (GDT) Register into the

specified operand.

Clocks Size

Operands 808x 286 386 486 Bytes

mem64 - 11 9 10 5

SIDT - Store Interrupt Descriptor Table (286+ privileged)

Usage: SIDT dest

Modifies flags: none

Stores the Interrupt Descriptor Table (IDT) Register into the

specified operand.

Clocks Size

Operands 808x 286 386 486 Bytes

mem64 - 12 9 10 5

SHL - Shift Logical Left

See: SAL

SHR - Shift Logical Right

Usage: SHR dest,count

Modifies flags: CF OF PF SF ZF (AF undefined)

ÚÄ? ÚÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ? ÚÄ?

?³ÄÄÄþ>? ÄÄÄÄÄÄÄÄÄÄ> 0³ÄÄÄþ>??

ÀÄ? ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ? ÀÄ?

Shifts the destination right by "count" bits with zeroes shifted

in on the left. The Carry Flag contains the last bit shifted out.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,1 2 2 3 2

mem,1 15+EA 7 7 2-4 (W88=23+EA)

reg,CL 8+4n 5+n 3 2

mem,CL 20+EA+4n 8+n 7 2-4 (W88=28+EA+4n)

reg,immed8 - 5+n 3 3

mem,immed8 - 8+n 7 3-5

SHLD/SHRD - Double Precision Shift (386+)

Usage: SHLD dest,src,count

SHRD dest,src,count

Modifies flags: CF PF SF ZF (OF,AF undefined)

SHLD shifts "dest" to the left "count" times and the bit positions

opened are filled with the most significant bits of "src". SHRD

shifts "dest" to the right "count" times and the bit positions

opened are filled with the least significant bits of the second

operand. Only the 5 lower bits of "count" are used.

Clocks Size

Operands 808x 286 386 486 Bytes

reg16,reg16,immed8 - - 3 2 4

reg32,reg32,immed8 - - 3 2 4

mem16,reg16,immed8 - - 7 3 6

mem32,reg32,immed8 - - 7 3 6

reg16,reg16,CL - - 3 3 3

reg32,reg32,CL - - 3 3 3

mem16,reg16,CL - - 7 4 5

mem32,reg32,CL - - 7 4 5

SLDT - Store Local Descriptor Table (286+ privileged)

Usage: SLDT dest

Modifies flags: none

Stores the Local Descriptor Table (LDT) Register into the

specified operand.

Clocks Size

Operands 808x 286 386 486 Bytes

reg16 - 2 2 2 3

mem16 - 2 2 3 5

SMSW - Store Machine Status Word (286+ privileged)

Usage: SMSW dest

Modifies flags: none

Store Machine Status Word (MSW) into "dest".

Clocks Size

Operands 808x 286 386 486 Bytes

reg16 - 2 10 2 3

mem16 - 3 3 3 5

STC - Set Carry

Usage: STC

Modifies flags: CF

Sets the Carry Flag to 1.

Clocks Size

Operands 808x 286 386 486 Bytes

none 2 2 2 2 1

STD - Set Direction Flag

Usage: STD

Modifies flags: DF

Sets the Direction Flag to 1 causing string instructions to

auto-decrement SI and DI instead of auto-increment.

Clocks Size

Operands 808x 286 386 486 Bytes

none 2 2 2 2 1

STI - Set Interrupt Flag (Enable Interrupts)

Usage: STI

Modifies flags: IF

Sets the Interrupt Flag to 1, which enables recognition of all

hardware interrupts. If an interrupt is generated by a hardware

device, an End of Interrupt (EOI) must also be issued to enable

other hardware interrupts of the same or lower priority.

Clocks Size

Operands 808x 286 386 486 Bytes

none 2 2 2 5 1

STOS - Store String (Byte, Word or Doubleword)

Usage: STOS dest

STOSB

STOSW

STOSD

Modifies flags: None

Stores value in accumulator to location at ES:(E)DI (even if operand

is given). (E)DI is incremented/decremented based on the size of

the operand (or instruction format) and the state of the Direction

Flag. Use with REP prefixes.

Clocks Size

Operands 808x 286 386 486 Bytes

dest 11 3 4 5 1 (W88=15)

STR - Store Task Register (286+ privileged)

Usage: STR dest

Modifies flags: None

Stores the current Task Register to the specified operand.

Clocks Size

Operands 808x 286 386 486 Bytes

reg16 - 2 2 2 3

mem16 - 3 2 3 5

SUB - Subtract

Usage: SUB dest,src

Modifies flags: AF CF OF PF SF ZF

The source is subtracted from the destination and the result is

stored in the destination.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,reg 3 2 2 1 2

mem,reg 16+EA 7 6 3 2-4 (W88=24+EA)

reg,mem 9+EA 7 7 2 2-4 (W88=13+EA)

reg,immed 4 3 2 1 3-4

mem,immed 17+EA 7 7 3 3-6 (W88=25+EA)

accum,immed 4 3 2 1 2-3

TEST - Test For Bit Pattern

Usage: TEST dest,src

Modifies flags: CF OF PF SF ZF (AF undefined)

Performs a logical AND of the two operands updating the flags

register without saving the result.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,reg 3 2 1 1 2

reg,mem 9+EA 6 5 1 2-4 (W88=13+EA)

mem,reg 9+EA 6 5 2 2-4 (W88=13+EA)

reg,immed 5 3 2 1 3-4

mem,immed 11+EA 6 5 2 3-6

accum,immed 4 3 2 1 2-3

VERR - Verify Read (286+ protected)

Usage: VERR src

Modifies flags: ZF

Verifies the specified segment selector is valid and is readable

at the current privilege level. If the segment is readable,

the Zero Flag is set, otherwise it is cleared.

Clocks Size

Operands 808x 286 386 486 Bytes

reg16 - 14 10 11 3

mem16 - 16 11 11 5

VERW - Verify Write (286+ protected)

Usage: VERW src

Modifies flags: ZF

Verifies the specified segment selector is valid and is ratable

at the current privilege level. If the segment is writable,

the Zero Flag is set, otherwise it is cleared.

Clocks Size

Operands 808x 286 386 486 Bytes

reg16 - 14 15 11 3

mem16 - 16 16 11 5

WAIT/FWAIT - Event Wait

Usage: WAIT

FWAIT

Modifies flags: None

CPU enters wait state until the coprocessor signals it has finished

its operation. This instruction is used to prevent the CPU from

accessing memory that may be temporarily in use by the coprocessor.

WAIT and FWAIT are identical.

Clocks Size

Operands 808x 286 386 486 Bytes

none 4 3 6+ 1-3 1

WBINVD - Write-Back and Invalidate Cache (486+)

Usage: WBINVD

Modifies flags: None

Flushes internal cache, then signals the external cache to write

back current data followed by a signal to flush the external cache.

Clocks Size

Operands 808x 286 386 486 Bytes

none - - - 5 2

XCHG - Exchange

Usage: XCHG dest,src

Modifies flags: None

Exchanges contents of source and destination.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,reg 4 3 3 3 2

mem,reg 17+EA 5 5 5 2-4 (W88=25+EA)

reg,mem 17+EA 5 5 3 2-4 (W88=25+EA)

accum,reg 3 3 3 3 1

reg,accum 3 3 3 3 1

XLAT/XLATB - Translate

Usage: XLAT translation-table

XLATB (masm 5.x)

Modifies flags: None

Replaces the byte in AL with byte from a user table addressed by

BX. The original value of AL is the index into the translate table.

The best way to discripe this is MOV AL,[BX+AL]

Clocks Size

Operands 808x 286 386 486 Bytes

table offset 11 5 5 4 1

XOR - Exclusive OR

Usage: XOR dest,src

Modifies flags: CF OF PF SF ZF (AF undefined)

Performs a bitwise exclusive OR of the operands and returns

the result in the destination.

Clocks Size

Operands 808x 286 386 486 Bytes

reg,reg 3 2 2 1 2

mem,reg 16+EA 7 6 3 2-4 (W88=24+EA)

reg,mem 9+EA 7 7 2 2-4 (W88=13+EA)

reg,immed 4 3 2 1 3-4

mem,immed 17+EA 7 7 3 3-6 (W88=25+EA)

accum,immed 4 3 2 1 2-3