80x86 Opcodes

Here is a list of instructions and opcodes used by Intel, AMD, Cyrix and Nexgen. *Gdb* was used get all the info out of the processors. This page was made by Asbjørn Leth Vonsild (alv@imada.ou.dk) and Jesper Pedersen (jews@imada.ou.dk). Please send any comments, questions directly to the original authors.
Last updated : 03.10.1996 at 11:30:00

Table of contents

Technical Specifications

	Introduction Date	Clock Speeds	Bus Width	Number of Transistors	Addressable Memory	Virtual Memory	Brief Description
4004	11/15/71	108 KHz	4 bits	2,300 (10 microns)	640 bytes		First microcomputer chip, Arithmetic manipulation
8008	4/1/72	108 KHz	8 bits	3,500	16 KBytes		Data/character manipulation
8080	4/1/74	2 MHz	8 bits	6,000 (6 microns)	64 KBytes		10X the performance of the 8008
8086	6/8/78	5 MHz 8 MHz 10 MHz	16 bits	29,000 (3 microns)	1 Megabyte		10X the performance of the 8080
8088	6/1/79	5 MHz 8 MHz	8 bits	29,000 (3 microns)			Identical to 8086 except for its 8-bit external bus
80286	2/1/82	8 MHz 10 MHz 12 MHz	16 bits	134,000 (1.5 microns)	16 Megabytes	1 gigabyte	3-6X the performance of the 8086
Intel386(TM)DX Microprocessor	10/17/85	16 MHz 20 MHz 25 MHz 33 MHz	32 bits	275,000 (1 micron)	4 gigabytes	64 terabytes	First X86 chip to handle 32-bit data sets
Intel386(TM)SX Microprocessor	6/16/88	16 MHz 20 MHz	16 bits	275,000 (1 micron)	4 gigabytes	64 terabytes	16-bit address bus enabled low-cost 32-bit processing
Intel486(TM)DX Microprocessor	4/10/89	25 MHz 33 MHz 50 MHz	32 bits	1,200,000 (1 micron, .8	4 gigabytes	64 terabytes	Level 1 cache on chip

				micron with 50 MHz)			
Intel486(TM)SX Microprocessor	4/22/91	16 MHz 20 MHz 25 MHz 33 MHz	32 bits	1,185,000 (.8 micron)	4 gigabytes	64 terabytes	identical in design to Intel486(TM) DX but without math coprocessor
Pentium® Processor	3/22/93	60MHz 66MHz 75MHz 90MHz 100MHz 120MHz 133MHz 150MHz 166MHz	32 bits	3.1 million (.8 micron)	4 gigabytes	64 terabytes	superscaler architecture brought 5X the performance of the 33-MHz Intel486 DX processor
Pentium® Pro Processor	3/27/95	150MHz 180MHz 200MHz	32 bits	5.5 million (.32 micron)	4 gigabytes	64 terabytes	dynamic execution architecture drives high-performing processor

---

Instructions and opcodes

oo : Function

- 00 : If mmm = 110, then a displacement follows the operation; otherwise, no displacement is used
- 01 : An 8-bit signed displacement follows the opcode
- 10 : A 16-bit signed displacement follows the opcode
- 11 : mmm specifies a register, instead of an addressing mode

mmm : Function

- 000 : DS:[BX+SI]
- 001 : DS:[BX+DI]
- 010 : SS:[BP+SI]
- 011 : SS:[BP+DI]
- 100 : DS:[SI]
- 101 : DS:[DI]
- 110 : SS:[BP]
- 111 : DS:[BX]

rrr : W=0 : W=1 : reg32

- 000 : AL : AX : EAX
- 001 : CL : CX : ECX
- 010 : DL : DX : EDX
- 011 : BL : BX : EBX
- 100 : AH : SP : ESP
- 101 : CH : BP : EBP
- 110 : DH : SI : ESI
- 111 : BH : DI : EDI

sss : Segment Register

- 000 : ES
- 001 : CS
- 010 : SS
- 011 : DS
- 100 : FS (Only 386+)
- 101 : GS (Only 386+)

rrr : Index Register

- 000 : EAX
- 001 : ECX
- 010 : EDX
- 011 : EBX
- 100 : No Index
- 101 : EBP
- 110 : ESI
- 111 : EDI

32 bit addressing-mode

oo	mmm	rrr	Description
00	000		DS:[EAX]
00	001		DS:[ECX]
00	010		DS:[EDX]
00	011		DS:[EBX]
00	100	000	DS:[EAX+scaled_index]
00	100	001	DS:[ECX+scaled_index]
00	100	010	DS:[EDX+scaled_index]
00	100	011	DS:[EBX+scaled_index]
00	100	100	SS:[ESP+scaled_index]
00	100	101	DS:[disp32+scaled_index]
00	100	110	DS:[ESI+scaled_index]
00	100	111	DS:[EDI+scaled_index]
00	101		DS:disp32
00	110		DS:[ESI]
00	111		DS:[EDI]
01	000		DS:[EAX+disp8]
01	001		DS:[ECX+disp8]
01	010		DS:[EDX+disp8]
01	011		DS:[EBX+disp8]
01	100	000	DS:[EAX+scaled_index+disp8]
01	100	001	DS:[ECX+scaled_index+disp8]
01	100	010	DS:[EDX+scaled_index+disp8]
01	100	011	DS:[EBX+scaled_index+disp8]

01	100	100	SS:[ESP+scaled_index+disp8]
01	100	101	SS:[EBP+scaled_index+disp8]
01	100	110	DS:[ESI+scaled_index+disp8]
01	100	111	DS:[EDI+scaled_index+disp8]
01	101		SS:[EBP+disp8]
01	110		DS:[ESI+disp8]
01	111		DS:[EDI+disp8]
10	000		DS:[EAX+disp32]
10	001		DS:[ECX+disp32]
10	010		DS:[EDX+disp32]
10	011		DS:[EBX+disp32]
10	100	000	DS:[EAX+scaled_index+disp32]
10	100	001	DS:[ECX+scaled_index+disp32]
10	100	010	DS:[EDX+scaled_index+disp32]
10	100	011	DS:[EBX+scaled_index+disp32]
10	100	100	SS:[ESP+scaled_index+disp32]
10	100	101	SS:[EBP+scaled_index+disp32]
10	100	110	DS:[ESI+scaled_index+disp32]
10	100	111	DS:[EDI+scaled_index+disp32]
10	101		SS:[EBP+disp32]
10	110		DS:[ESI+disp32]
10	111		DS:[EDI+disp32]

Main Instructions

A || B || C || D || E || H || I || J || L || M || N || O || P || R || S || T || V || W || X

Name	Regs	Opcode	Proc	Description
AAA		00110111	8086	ASCII Adjust After Addition
AAD	Imm8	11010101	Pentium	ASCII Adjust Register AX Before Division
		1101010100001010	8086	ASCII Adjust Register AX Before Division

AAM	Imm8	11010100	Pentium	ASCII Adjust AX Register After Multiplication
		1101010000001010	8086	ASCII Adjust AX Register After Multiplication
AAS		00111111	8086	ASCII Adjust AL Register After Substraction
ADC	Reg,Reg	0001001woorrrmmm	8086	Add Integers with Carry
	Mem,Reg	0001000woorrrmmm	8086	Add Integers with Carry
	Reg,Mem	0001001woorrrmmm	8086	Add Integers with Carry
	Acc,Imm	0001010w	8086	Add Integers with Carry
	Reg,Imm8	1000001woo010mmm	8086	Add Integers with Carry
	Mem,Imm8	1000001woo010mmm	8086	Add Integers with Carry
	Reg,Imm	1000000woo010mmm	8086	Add Integers with Carry
	Mem,Imm	1000000woo010mmm	8086	Add Integers with Carry
ADD	Reg,Reg	0000001woorrrmmm	8086	Add Integers
	Mem,Reg	0000000woorrrmmm	8086	Add Integers
	Reg,Mem	0000001woorrrmmm	8086	Add Integers
	Acc,Imm	0000010w	8086	Add Integers
	Reg,Imm8	1000001woo000mmm	8086	Add Integers
	Mem,Imm8	1000001woo000mmm	8086	Add Integers
	Reg,Imm	1000000woo000mmm	8086	Add Integers
	Mem,Imm	1000000woo000mmm	8086	Add Integers
AND	Reg,Reg	0010001woorrrmmm	8086	Logical AND
	Mem,Reg	0010000woorrrmmm	8086	Logical AND
	Reg,Mem	0010001woorrrmmm	8086	Logical AND
	Acc,Imm	0010010w	8086	Logical AND
	Reg,Imm8	1000001woo100mmm	8086	Logical AND
	Mem,Imm8	1000001woo100mmm	8086	Logical AND

	Reg,Imm	1000000woo100mmm	8086	Logical AND
	Mem,Imm	1000000woo100mmm	8086	Logical AND
ARPL	Reg16,Reg16	01100011oorrrmmm	80286	Adjust Requester Privilege Level of Selector
	Mem16,Reg16	01100011oorrrmmm	80286	Adjust Requester Privilege Level of Selector
BOUND	Reg16,Mem32	01100010oorrrmmm□	80186	Check Array Index Against Bounds
	Reg32,Mem64	01100010oorrrmmm	80386	Check Array Index Against Bounds
BSF	RegWord,RegWord	0000111110111100oorrrmmm	80386	Bit Scan Forward
	RegWord,MemWord	0000111110111100oorrrmmm	80386	Bit Scan Forward
BSR	RegWord,RegWord	0000111110111101oorrrmmm	80386	Bit Scan Reverse
	RegWord,MemWord	0000111110111101oorrrmmm	80386	Bit Scan Reverse
BSWAP	RegWord	0000111111001rrr	80486	Byte swap
BT	RegWord,Imm8	0000111110111010oo100mmm	80386	Bit Test
	MemWord,Imm8	0000111110111010oo100mmm	80386	Bit Test
	RegWord,RegWord	0000111110100011oorrrmmm	80386	Bit Test
	MemWord,RegWord	0000111110100011oorrrmmm	80386	Bit Test
BTC	RegWord,Imm8	0000111110111010oo111mmm	80386	Bit Test and Complement
	MemWord,Imm8	0000111110111010oo111mmm	80386	Bit Test and Complement
	RegWord,RegWord	0000111110111011oorrrmmm	80386	Bit Test and Complement
	MemWord,RegWord	0000111110111011oorrrmmm	80386	Bit Test and Complement
BTR	RegWord,Imm8	0000111110111010oo110mmm	80386	Bit Test and Reset
	MemWord,Imm8	0000111110111010oo110mmm	80386	Bit Test and Reset
	RegWord,RegWord	0000111110110011oorrrmmm	80386	Bit Test and

				Reset
	MemWord,RegWord	0000111110110011oorrrmmm	80386	Bit Test and Reset
BTS	RegWord,Imm8	0000111110111010oo101mmm	80386	Bit Test and Set
	MemWord,Imm8	0000111110111010oo101mmm	80386	Bit Test and Set
	RegWord,RegWord	0000111110101011oorrrmmm	80386	Bit Test and Set
	MemWord,RegWord	0000111110101011oorrrmmm	80386	Bit Test and Set
CBW		10011000□	8086	Convert Byte to Word
CDQ		10011001	80386	Convert Doubleword to Quad-Word
CLC		11111000	8086	Clear Carry Flag (CF)
CLD		11111100	8086	Clear Direction Flag (DF)
CLI		11111010	8086	Clear Interrupt Flag (IF)
CLTS		0000111100000110	80286	Clear Task-Switched Flag in Control Register Zero
CMC		11110101	8086	Complementer Carry Flag (CF)
CMOVcc	Reg,Reg	000011110100ccccoorrrmmm	PentiumPro	Conditional Move
	Reg,Mem	000011110100ccccoorrrmmm	PentiumPro	Conditional Move
CMP	Reg,Reg	0011101woorrrmmm	8086	Compare
	Mem,Reg	0011100woorrrmmm	8086	Compare
	Reg,Mem	0011101woorrrmmm	8086	Compare
	Acc,Imm	0011110w	8086	Compare
	Reg,Imm8	1000001woo111mmm	8086	Compare
	Mem,Imm8	1000001woo111mmm	8086	Compare
	Reg,Imm	1000000woo111mmm	8086	Compare
	Mem,Imm	1000000woo111mmm	8086	Compare
CMPSB		10100110	8086	Compare

				String - Byte
CMPSW		10100111□	8086	Compare String - Word
CMPSD		10100111	80386	Compare String - Doubleword
CMPXCHG	Reg,Reg	000011111011000woorrrmmm	80486	Compare and Exchange
	Mem,Reg	000011111011000woorrrmmm	80486	Compare and Exchange
CMPXCHG8B	Mem64	0000111111000111oo001mmm	Pentium	Compare and Exchange 8 Bytes
CPUID		0000111110100010	Pentium	CPU Identification code to EAX
CWD		10011001□	8086	Convert Word to Doubleword
CWDE		10011000	80386	Convert Word to Extended Doubleword
DAA		00100111	8086	Decimal Adjust Register After Addition
DAS		00101111	8086	Decimal Adjust AL Register After Substraction
DEC	RegWord	01001rrr	8086	Decrement by One
	Reg	1111111woo001mmm	8086	Decrement by One
	Mem	1111111woo001mmm	8086	Decrement by One
DIV	Reg	1111011woo110mmm	8086	Unsigned Integer Divide
	Mem	1111011woo110mmm	8086	Unsigned Integer Divide
ENTER	Imm16,Imm8	11001000	80186	Make Stack Frame for Procedure Parameter
HLT		11110100	8086	Halt
IDIV	Reg	1111011woo111mmm	8086	Signed Divide

	Mem	1111011woo111mmm	8086	Signed Divide
IMUL	RegWord,RegWord,Imm8	01101011oorrrmmm	80186	Signed Integer Multiply
	RegWord,MemWord,Imm8	01101011oorrrmmm	80186	Signed Integer Multiply
	RegWord,RegWord,Imm	01101001oorrrmmm	80186	Signed Integer Multiply
	RegWord,MemWord,Imm	01101001oorrrmmm	80186	Signed Integer Multiply
	RegWord,Imm8	0110101111rrrqqq	80186	Signed Integer Multiply
	RegWord,Imm	0110100111rrrqqq	80186	Signed Integer Multiply
	RegWord,RegWord	0000111110101111oorrrmmm	80386	Signed Integer Multiply
	RegWord,MemWord	0000111110101111oorrrmmm	80386	Signed Integer Multiply
	Reg	1111011woo101mmm	8086	Signed Integer Multiply
	Mem	1111011woo101mmm	8086	Signed Integer Multiply
IN	Acc,Imm8	1110010w	8086	Input from Port
	Acc,DX	1110110w	8086	Input from Port
INC	RegWord	01000rrr	8086	Increment by 1
	Reg	1111111woo000mmm	8086	Increment by 1
	Mem	1111111woo000mmm	8086	Increment by 1
INSB		01101100	80186	Input Byte
INSW		01101101□	80186	Input Word
INSD		01101101	80386	Input DoubleWord
INT	3	11001100	8086	Call to Interrupt Procedure
	Imm8	11001101	8086	Call to Interrupt Procedure
INTO		11001110	8086	Interrupt on Overflow

INVD		0000111100001000	80486	Invalidate data cache
INVLPG	Mem	0000111100000001oo111mmm	80486	Invalidate TBL entry
IRET		11001111□	8086	Return from Interrupt
IRETD		11001111	80386	Return from Interrupt - 32-bit Mode
LAHF		10011111	8086	Load Flags into AH Register
LAR	RegWord,RegWord	0000111100000010oorrrmmm	80286	Load Access Rights Byte
	RegWord,MemWord	0000111100000010oorrrmmm	80286	Load Access Rights Byte
LDS	Reg16,Mem32	11000101oorrrmmm□	8086	Load Pointer Using DS
	Reg32,Mem64	11000101oorrrmmm	80386	Load Pointer Using DS
LES	Reg16,Mem32	11000100oorrrmmm□	8086	Load Pointer Using ES
	Reg32,Mem64	11000100oorrrmmm	80386	Load Pointer Using ES
LFS	Reg16,Mem32	0000111110110100oorrrmmm□	80386	Load Pointer Using FS
	Reg32,Mem64	0000111110110100oorrrmmm	80386	Load Pointer Using FS
LGS	Reg16,Mem32	0000111110110101oorrrmmm□	80386	Load Pointer Using GS
	Reg32,Mem64	0000111110110101oorrrmmm	80386	Load Pointer Using GS
LSS	Reg16,Mem32	0000111110110010oorrrmmm□	80386	Load Pointer Using SS
	Reg32,Mem64	0000111110110010oorrrmmm	80386	Load Pointer Using SS
LEA	RegWord,Mem	10001101oorrrmmm	8086	Load Effective Address
LEAVE		11001001	80186	High Level Procedure Exit
LGDT	Mem64	0000111100000001oo010mmm	80286	Load Global Descriptor Table

LIDT	Mem64	0000111100000001oo011mmm	80286	Load Interrupt Descriptor Table
LLDT	Reg16	0000111100000000oo010mmm	80286	Load Local Descriptor Table
	Mem16	0000111100000000oo010mmm	80286	Load Local Descriptor Table
LMSW	Reg16	0000111100000001oo110mmm	80286	Load Machine Status Word
	Mem16	0000111100000001oo110mmm	80286	Load Machine Status Word
LODSB		10101100	8086	Load Byte
LODSW		10101101□	8086	Load Word
LODSD		10101101	80386	Load Doubleword
LSL	RegWord,RegWord	0000111100000011oorrrmmm	80286	Load Segment Limit
	RegWord,MemWord	0000111100000011oorrrmmm	80286	Load Segment Limit
LTR	Reg16	0000111100000000oo011mmm	80286	Load Task Register
	Mem16	0000111100000000oo011mmm	80286	Load Task Register
MOV	MemOfs,Acc	1010001w	8086	Move Data
	Acc,MemOfs	1010000w	8086	Move Data
	Reg,Imm	1011wrrr	8086	Move Data
	Mem,Imm	1100011woo000mmm	8086	Move Data
	Reg,Reg	1000101woorrrmmm	8086	Move Data
	Reg,Mem	1000101woorrrmmm	8086	Move Data
	Mem,Reg	1000100woorrrmmm	8086	Move Data
	Reg16,Seg	10001100oosssmmm	8086	Move Data
	Seg,Reg16	10001110oosssmmm	8086	Move Data
	Mem16,Seg	10001100oosssmmm	8086	Move Data
	Seg,Mem16	10001110oosssmmm	8086	Move Data
	Reg32,CRn	00001111001000011sssrrr	80386	Move Data
	CRn,Reg32	00001111001000111sssrrr	80386	Move Data
	Reg32,DRn	00001111001000011sssrrr	80386	Move Data
	DRn,Reg32	00001111001000111sssrrr	80386	Move Data
	Reg32,TRn	00001111001001011sssrrr	80386	Move Data

	TRn,Reg32	0000111100100110 11sssrrr	80386	Move Data
MOVSB		10100100	8086	Move Byte
MOVSW		10100101	8086	Move Word
MOVSD		10100101	80386	Move Doubleword
MOVSX	RegWord,Reg8	0000111110111110oorrrmmm	80386	Move with Sign Extension
	RegWord,Mem8	0000111110111110oorrrmmm	80386	Move with Sign Extension
	RegWord,Reg16	0000111110111111oorrrmmm	80386	Move with Sign Extension
	RegWord,Mem16	0000111110111111oorrrmmm	80386	Move with Sign Extension
MOVZX	RegWord,Reg8	0000111110110110oorrrmmm	80386	Move with Zero Extension
	RegWord,Mem8	0000111110110110oorrrmmm	80386	Move with Zero Extension
	RegWord,Reg16	0000111110110111oorrrmmm	80386	Move with Zero Extension
	RegWord,Mem16	0000111110110111oorrrmmm	80386	Move with Zero Extension
MUL	Reg	1111011woo100mmm	8086	Unsigned Integer Multiply of AL, AX or EAX
	Mem	1111011woo100mmm	8086	Unsigned Integer Multiply of AL, AX or EAX
NEG	Reg	1111011woo011mmm	8086	Negate(Two's Complement)
	Mem	1111011woo011mmm	8086	Negate(Two's Complement)
NOP		10010000	8086	No Operation
NOT	Reg	1111011woo010mmm	8086	Negate(One's Complement)

	Mem	1111011woo010mmm	8086	Negate(One's Complement)
OR	Reg,Reg	0000101woorrrmmm	8086	Logical Inclusive OR
	Mem,Reg	0000100woorrrmmm	8086	Logical Inclusive OR
	Reg,Mem	0000101woorrrmmm	8086	Logical Inclusive OR
	Acc,Imm	0000110w	8086	Logical Inclusive OR
	Reg,Imm8	1000001woo001mmm	8086	Logical Inclusive OR
	Mem,Imm8	1000001woo001mmm	8086	Logical Inclusive OR
	Reg,Imm	1000000woo001mmm	8086	Logical Inclusive OR
	Mem,Imm	1000000woo001mmm	8086	Logical Inclusive OR
OUT	Imm8,Acc	1110011w	8086	Output To Port
	DX,Acc	1110111w	8086	Output To Port
OUTSB		01101110	80186	Output Byte
OUTSW		01101111□	80186	Output Word
OUTSD		01101111	80386	Output Doubleword
POP	RegWord	01011rrr	8086	Pop a Word from the Stack
	MemWord	10001111oo000mmm	8086	Pop a Word from the Stack
	SegOld	00sss111	8086	Pop a Word from the Stack
	Seg	0000111110sss001	80386	Pop a Word from the Stack
POPA		01100001□	80186	POP All Registers
POPAD		01100001	80386	POP All Registers - 32-bit Mode
POPF		10011101□	8086	POP Stack into FLAGS
POPFD		10011101	80386	POP Stack into EFLAGS

PUSH	RegWord	01010rrr	8086	Push Operand onto Stack
	MemWord	11111111oo110mmm	8086	Push Operand onto Stack
	SegOld	00sss110	8086	Push Operand onto Stack
	Seg	0000111110sss000	80386	Push Operand onto Stack
	Imm8	01101010	80186	Push Operand onto Stack
	Imm	01101000	80186	Push Operand onto Stack
PUSHW	Imm16	01101000□	80286	PUSH Word
PUSHD	Imm32	01101000	80386	PUSH Double Word
PUSHA		01100000□	80186	PUSH All Registers
PUSHAD		01100000	80386	PUSH All Registers - 32-bit Mode
PUSHF		10011100□	8086	PUSH FLAGS
PUSHFD		10011100	80386	PUSH EFLAGS
RCL	Reg,1	1101000woo010mmm	8086	Rotate Left through Carry - Uses CF for Extension
	Mem,1	1101000woo010mmm	8086	Rotate Left through Carry - Uses CF for Extension
	Reg,CL	1101001woo010mmm	8086	Rotate Left through Carry - Uses CF for Extension
	Mem,CL	1101001woo010mmm	8086	Rotate Left through Carry - Uses CF for Extension
	Reg,Imm8	1100000woo010mmm	80186	Rotate Left through Carry - Uses CF for Extension
	Mem,Imm8	1100000woo010mmm	80186	Rotate Left through Carry

				- Uses CF for Extension
RCR	Reg,1	1101000woo011mmm	8086	Rotate Right through Carry - Uses CF for Extension
	Mem,1	1101000woo011mmm	8086	Rotate Right through Carry - Uses CF for Extension
	Reg,CL	1101001woo011mmm	8086	Rotate Right through Carry - Uses CF for Extension
	Mem,CL	1101001woo011mmm	8086	Rotate Right through Carry - Uses CF for Extension
	Reg,Imm8	1100000woo011mmm	80186	Rotate Right through Carry - Uses CF for Extension
	Mem,Imm8	1100000woo011mmm	80186	Rotate Right through Carry - Uses CF for Extension
RDMSR		0000111100110010	Pentium	Read from Model Specific Register
RET	NEAR	11000011	8086	Return from subprocedure
RET	imm NEAR	11000010	8086	Return from subprocedure
RET	FAR	11001011	8086	Return from subprocedure
RET	imm FAR	11001010	8086	Return from subprocedure
RDPMC		0000111100110011	PentiumPro	Read Performance Monitor Counter
ROL	Reg,1	1101000woo000mmm	8086	Rotate Left through Carry - Wrap bits around
	Mem,1	1101000woo000mmm	8086	Rotate Left through Carry

				- Wrap bits around
	Reg,CL	1101001woo000mmm	8086	Rotate Left through Carry - Wrap bits around
	Mem,CL	1101001woo000mmm	8086	Rotate Left through Carry - Wrap bits around
	Reg,Imm8	1100000woo000mmm	80186	Rotate Left through Carry - Wrap bits around
	Mem,Imm8	1100000woo000mmm	80186	Rotate Left through Carry - Wrap bits around
ROR	Reg,1	1101000woo001mmm	8086	Rotate Right through Carry - Wrap bits around
	Mem,1	1101000woo001mmm	8086	Rotate Right through Carry - Wrap bits around
	Reg,CL	1101001woo001mmm	8086	Rotate Right through Carry - Wrap bits around
	Mem,CL	1101001woo001mmm	8086	Rotate Right through Carry - Wrap bits around
	Reg,Imm8	1100000woo001mmm	80186	Rotate Right through Carry - Wrap bits around
	Mem,Imm8	1100000woo001mmm	80186	Rotate Right through Carry - Wrap bits around
RSM		0000111110101010	Pentium	Return from System Management mode
SALC		11010110	Pentium Pro	Set AL on Carry
SAHF		10011110	8086	Load Flags into AH

				Register
SAL	Reg,1	1101000woo100mmm	8086	Shift Arithmetic Left
	Mem,1	1101000woo100mmm	8086	Shift Arithmetic Left
	Reg,CL	1101001woo100mmm	8086	Shift Arithmetic Left
	Mem,CL	1101001woo100mmm	8086	Shift Arithmetic Left
	Reg,Imm8	1100000woo100mmm	80186	Shift Arithmetic Left
	Mem,Imm8	1100000woo100mmm	80186	Shift Arithmetic Left
SAR	Reg,1	1101000woo111mmm	8086	Shift Arithmetic Right
	Mem,1	1101000woo111mmm	8086	Shift Arithmetic Right
	Reg,CL	1101001woo111mmm	8086	Shift Arithmetic Right
	Mem,CL	1101001woo111mmm	8086	Shift Arithmetic Right
	Reg,Imm8	1100000woo111mmm	80186	Shift Arithmetic Right
	Mem,Imm8	1100000woo111mmm	80186	Shift Arithmetic Right
SETcc	Reg8	000011111001ccccoo000mmm	80386	Set Byte on Condition Code
	Mem8	000011111001ccccoo000mmm	80386	Set Byte on Condition Code
SHL	Reg,1	1101000woo100mmm	8086	Shift Logic Left
	Mem,1	1101000woo100mmm	8086	Shift Logic Left

	Reg,CL	1101001woo100mmm	8086	Shift Logic Left
	Mem,CL	1101001woo100mmm	8086	Shift Logic Left
	Reg,Imm8	1100000woo100mmm	80186	Shift Logic Left
	Mem,Imm8	1100000woo100mmm	80186	Shift Logic Left
SHR	Reg,1	1101000woo101mmm	8086	Shift Logic Right
	Mem,1	1101000woo101mmm	8086	Shift Logic Right
	Reg,CL	1101001woo101mmm	8086	Shift Logic Right
	Mem,CL	1101001woo101mmm	8086	Shift Logic Right
	Reg,Imm8	1100000woo101mmm	80186	Shift Logic Right
	Mem,Imm8	1100000woo101mmm	80186	Shift Logic Right
SBB	Reg,Reg	0001101woorrrmmm	8086	Substract Integers with Borrow
	Mem,Reg	0001100woorrrmmm	8086	Substract Integers with Borrow
	Reg,Mem	0001101woorrrmmm	8086	Substract Integers with Borrow
	Acc,Imm	0001110w	8086	Substract Integers with Borrow
	Reg,Imm8	1000001woo011mmm	8086	Substract Integers with Borrow
	Mem,Imm8	1000001woo011mmm	8086	Substract Integers with Borrow
	Reg,Imm	1000000woo011mmm	8086	Substract Integers with Borrow
	Mem,Imm	1000000woo011mmm	8086	Substract Integers with Borrow
SCASB		10101110	8086	Compare Byte

SCASW		10101111□	8086	Compare Word
SCASD		10101111	80386	Compare Doubleword
SGDT	Mem64	0000111100000001oo000mmm	80286	Store Global Descriptor Table
SHLD	RegWord,RegWord,Imm8	0000111110100100oorrrmmm	80386	Double Precision Shift Left
	MemWord,RegWord,Imm8	0000111110100100oorrrmmm	80386	Double Precision Shift Left
	RegWord,RegWord,CL	0000111110100101oorrrmmm	80386	Double Precision Shift Left
	MemWord,RegWord,CL	0000111110100101oorrrmmm	80386	Double Precision Shift Left
SHRD	RegWord,RegWord,Imm8	0000111110101100oorrrmmm	80386	Double Precision Shift Right
	MemWord,RegWord,Imm8	0000111110101100oorrrmmm	80386	Double Precision Shift Right
	RegWord,RegWord,CL	0000111110101101oorrrmmm	80386	Double Precision Shift Right
	MemWord,RegWord,CL	0000111110101101oorrrmmm	80386	Double Precision Shift Right
SIDT	Mem64	0000111100000001oo001mmm	80286	Store Interrupt Descriptor Table
SLDT	Reg16	0000111100000000oo000mmm	80286	Store Local Descriptor Table Register (LDTR)
	Mem16	0000111100000000oo000mmm	80286	Store Local Descriptor Table Register (LDTR)
SMSW	Reg16	0000111100000001oo100mmm	80286	Store Machine Status Word
	Mem16	0000111100000001oo100mmm	80286	Store Machine Status Word
STC		11111001	8086	Set Carry

				Flag(CF)
STD		11111101	8086	Set Direction Flag(DF)
STI		11111011	8086	Set Interrupt Flag(IF)
STOSB		10101010	8086	Store String Data Byte
STOSW		10101011□	8086	Store String Data Word
STOSD		10101011	80386	Store String Data DoubleWord
STR	Reg16	0000111100000000oo001mmm	80286	Store Task Register
	Mem16	0000111100000000oo001mmm	80286	Store Task Register
SUB	Reg,Reg	0010101woorrrmmm	8086	Subtract
	Mem,Reg	0010100woorrrmmm	8086	Subtract
	Reg,Mem	0010101woorrrmmm	8086	Subtract
	Acc,Imm	0010110w	8086	Subtract
	Reg,Imm8	1000001woo101mmm	8086	Subtract
	Mem,Imm8	1000001woo101mmm	8086	Subtract
	Reg,Imm	1000000woo101mmm	8086	Subtract
	Mem,Imm	1000000woo101mmm	8086	Subtract
TEST	Reg,Reg	1000010woorrrmmm	8086	Test Operands
	Mem,Reg	1000010woorrrmmm	8086	Test Operands
	Reg,Mem	1000010woorrrmmm	8086	Test Operands
	Acc,Imm	1010100w	8086	Test Operands
	Reg,Imm	1111011woo000mmm	8086	Test Operands
	Mem,Imm	1111011woo000mmm	8086	Test Operands
VERR	Reg16	0000111100000000oo100mmm	80286	Verify Read
	Mem16	0000111100000000oo100mmm	80286	Verify Read
VERW	Reg16	0000111100000000oo101mmm	80286	Verify Write
	Mem16	0000111100000000oo101mmm	80286	Verify Write
WAIT		10011011	8086	Wait for FPU
WBINVD		0000111100001001	80486	Write Back and Invalidate Data Cache
WRMSR		0000111100110000	Pentium	Write to Model

				Specific Register
XADD	Reg,Reg	000011111100000woorrrmmm	80486	Exchange and Add
	Mem,Reg	000011111100000woorrrmmm	80486	Exchange and Add
XCHG	AccWord,RegWord	10010rrr	8086	Exchange
	RegWord,AccWord	10010rrr	8086	Exchange
	Reg,Reg	1000011woorrrmmm	8086	Exchange
	Mem,Reg	1000011woorrrmmm	8086	Exchange
	Reg,Mem	1000011woorrrmmm	8086	Exchange
XLAT		11010111	8086	Translate
XOR	Reg,Reg	0011001woorrrmmm	8086	Exclusive-OR
	Mem,Reg	0011000woorrrmmm	8086	Exclusive-OR
	Reg,Mem	0011001woorrrmmm	8086	Exclusive-OR
	Acc,Imm	0011010w	8086	Exclusive-OR
	Reg,Imm8	1000001woo110mmm	8086	Exclusive-OR
	Mem,Imm8	1000001woo110mmm	8086	Exclusive-OR
	Reg,Imm	1000000woo110mmm	8086	Exclusive-OR
	Mem,Imm	1000000woo110mmm	8086	Exclusive-OR
CALL	MemFar	11111111oo011mmm	8086	Call a Procedure
	Near	11101000	8086	Call a Procedure
	Far	10011010	8086	Call a Procedure
	RegWord	11111111oo010mmm	8086	Call a Procedure
	MemNear	11111111oo010mmm	8086	Call a Procedure
Jcc	Short	0111cccc	8086	Jump on Some Condition Code
	Near	000011111000cccc	80386	Jump on Some Condition Code
JCXZ	Short	11100011□	8086	
JCXE	Short	11100011□	8086	
JECXZ	Short	11100011	8086	

JECXE	Short	11100011	8086	
JMP	MemFar	11111111oo101mmm	8086	
	Short	11101011	8086	
	Near	11101001	8086	
	Far	11101010	8086	
	RegWord	11111111oo100mmm	8086	
	MemNear	11111111oo100mmm	8086	
LOOP	Short	11100010	8086	Loop Control While ECX Counter Not Zero
LOOPZ	Short	11100001	8086	Loop while Zero
LOOPE	Short	11100001	8086	Loop while Equal
LOOPNZ	Short	11100000	8086	Loop while Not Zero
LOOPNE	Short	11100000	8086	Loop while Not Equal
LOCK		11110000	8086	Assert Lock# Signal Prefix
LOCK:		11110000	8086	Assert Lock# Signal Prefix
REP		11110011	8086	Repeat Following String Operation
REPE		11110011	8086	Repeat while Equal
REPZ		11110011	8086	Repeat while Zero
REPNE		11110010	8086	Repeat while Not Equal
REPNZ		11110010	8086	Repeat while Not Zero
CS:		00101110	8086	CS segment override prefix
DS:		00111110	8086	DS segment override prefix
ES:		00100110	8086	ES segment override prefix

FS:		01100100	80386	FS segment override prefix
GS:		01100101	80386	GS segment override prefix
SS:		00110110	8086	SS segment override prefix

Co-processor instructions :

Name	Regs	Opcode	Proc	Description
F2XM1		1101100111110000	8087	2^X-1
FABS		1101100111100001	8087	Absolute Value of ST
FADD	ST(0),ST(n)	1101100011000rrr	8087	Addition
	ST(n),ST(0)	1101110011000rrr	8087	Addition
	ST(0),Mem32	11011000oo000mmm	8087	Addition
	Mem32	11011000oo000mmm	8087	Addition
	ST(0),Mem64	11011100oo000mmm	8087	Addition
	Mem64	11011100oo000mmm	8087	Addition
	ST(n)	1101110011000rrr	8087	Addition
		1101110011000001	8087	Addition
FADDP	ST(n),ST(0)	11011110oo000mmm	8087	Addition and Pop
	ST(n)	11011110oo000mmm	8087	Addition and Pop
		1101111011000001	8087	Addition and Pop
FIADD	ST(0),Mem16	11011110oo000mmm	8087	Addition (Integer)
	Mem16	11011110oo000mmm	8087	Addition (Integer)
	ST(0),Mem32	11011010oo000mmm	8087	Addition (Integer)
	Mem32	11011010oo000mmm	8087	Addition (Integer)
FCMOVcc	ST(0),ST(n)	1101101n110ccrrr	PentiumPro	Conditional Move
	ST(n)	1101101n110ccrrr	PentiumPro	Conditional Move
		1101101n110cc001	PentiumPro	Conditional Move
FCHS		1101100111100000	8087	Change Sign
FCLEX		[FWAIT] 1101101111100010	8087	Clear Errors
FNCLEX		1101101111100010	8087	Clear Errors

FCOM	ST(0),Mem64	11011100oo010mmm	8087	Compare
	Mem64	11011100oo010mmm	8087	Compare
	ST(0),ST(n)	11011000oo010mmm	8087	Compare
	ST(0),Mem32	11011000oo010mmm	8087	Compare
	ST(n)	11011000oo010mmm	8087	Compare
	Mem32	11011000oo010mmm	8087	Compare
		1101100011010001	8087	Compare
FCOMP	ST(0),Mem64	11011100oo011mmm	8087	Compare and Pop
	Mem64	11011100oo011mmm	8087	Compare and Pop
	ST(0),ST(n)	11011000oo011mmm	8087	Compare and Pop
	ST(0),Mem32	11011000oo011mmm	8087	Compare and Pop
	ST(n)	11011000oo011mmm	8087	Compare and Pop
	Mem32	11011000oo011mmm	8087	Compare and Pop
		1101100011011001	8087	Compare and Pop
FICOM	ST(0),Mem16	11011110oo010mmm	8087	Compare (Integer)
	Mem16	11011110oo010mmm	8087	Compare (Integer)
	ST(0),Mem32	11011010oo010mmm	8087	Compare (Integer)
	Mem32	11011010oo010mmm	8087	Compare (Integer)
FICOMP	ST(0),Mem16	11011110oo011mmm	8087	Compare (Integer) and Pop
	Mem16	11011110oo011mmm	8087	Compare (Integer) and Pop
	ST(0),Mem32	11011010oo011mmm	8087	Compare (Integer) and Pop
	Mem32	11011010oo011mmm	8087	Compare (Integer) and Pop
FCOMI	ST(0),ST(n)	1101101111110rrr	PentiumPro	Compare Integer (EFLAGS)
	ST(n)	1101101111110rrr	PentiumPro	Compare Integer (EFLAGS)
		1101101111110001	PentiumPro	Compare Integer (EFLAGS)
FCOMIP	ST(0),ST(n)	1101111111110rrr	PentiumPro	Compare Integer and Pop (EFLAGS)
	ST(n)	1101111111110rrr	PentiumPro	Compare Integer and Pop (EFLAGS)
		1101111111110001	PentiumPro	Compare Integer and Pop (EFLAGS)
FUCOMI	ST(0),ST(n)	1101101111101rrr	PentiumPro	Unordered Compare Integer (EFLAGS)
	ST(n)	1101101111101rrr	PentiumPro	Unordered Compare Integer (EFLAGS)
		1101101111101001	PentiumPro	Unordered Compare Integer (EFLAGS)
FUCOMIP	ST(0),ST(n)	1101111111101rrr	PentiumPro	Unordered Compare Integer

				(EFLAGS)
	ST(n)	1101111111101rrr	PentiumPro	Unordered Compare Integer (EFLAGS)
		1101111111101001	PentiumPro	Unordered Compare Integer (EFLAGS)
FUCOMPP		1101111011010101	8086	Compare and Pop and Pop
FCOS		1101100111111111	80387	Cosine
FDECSTP		1101100111110110	8087	Decrement Stack Pointer
FDISI		[FWAIT] 1101101111100001	8087	Disable Interrupts
FNDISI		1101101111100001	8087	Disable Interrupts
FDIV	ST(0),ST(n)	1101100011110rrr	8087	Division
	ST(n),ST(0)	1101110011111rrr	8087	Division
	ST(0),Mem32	11011000oo110mmm	8087	Division
	Mem32	11011000oo110mmm	8087	Division
	ST(0),Mem64	11011100oo110mmm	8087	Division
	Mem64	11011100oo110mmm	8087	Division
	ST(n)	1101110011111rrr	8087	Division
		1101110011111001	8087	Division
FDIVP	ST(n),ST(0)	11011110oo111mmm	8087	Division and Pop
	ST(n)	11011110oo111mmm	8087	Division and Pop
		1101111011111001	8087	Division and Pop
FIDIV	ST(0),Mem16	11011110oo110mmm	8087	Division (Integer) and Pop
	Mem16	11011110oo110mmm	8087	Division (Integer) and Pop
	ST(0),Mem32	11011010oo110mmm	8087	Division (Integer) and Pop
	Mem32	11011010oo110mmm	8087	Division (Integer) and Pop
FDIVR	ST(0),ST(n)	1101100011111rrr	8087	Division Reversed
	ST(n),ST(0)	1101110011110rrr	8087	Division Reversed
	ST(0),Mem32	11011000oo111mmm	8087	Division Reversed
	Mem32	11011000oo111mmm	8087	Division Reversed
	ST(0),Mem64	11011100oo111mmm	8087	Division Reversed
	Mem64	11011100oo111mmm	8087	Division Reversed
	ST(n)	1101110011110rrr	8087	Division Reversed
		1101110011110001	8087	Division Reversed
FDIVRP	ST(n),ST(0)	11011110oo110mmm	8087	Division Reversed and Pop
	ST(n)	11011110oo110mmm	8087	Division Reversed and Pop
		1101111011110001	8087	Division Reversed and Pop

FIDIVR	ST(0),Mem16	11011110oo111mmm	8087	Division Reversed (Integer)
	Mem16	11011110oo111mmm	8087	Division Reversed (Integer)
	ST(0),Mem32	11011010oo111mmm	8087	Division Reversed (Integer)
	Mem32	11011010oo111mmm	8087	Division Reversed (Integer)
FENI		[FWAIT] 1101101111100000	8087	Disable Interrupts
FNENI		1101101111100000	8087	Disable Interrupts
FFREE	ST(n)	1101110111000rrr	8087	Free Register
		1101110111000001	8087	Free Register
FINCSTP		1101100111110111	8087	Increment Stack Pointer
FINIT		[FWAIT] 1101101111100011	8087	Initialize FPU
FNINIT		1101101111100011	8087	Initialize FPU
FLD	ST(0),Mem32	11011001oo000mmm	8087	Load Data
	Mem32	11011001oo000mmm	8087	Load Data
	ST(0),Mem64	11011101oo000mmm	8087	Load Data
	Mem64	11011101oo000mmm	8087	Load Data
	ST(0),Mem80	11011011oo101mmm	8087	Load Data
	Mem80	11011011oo101mmm	8087	Load Data
	ST(0),ST(n)	11011001oo000mmm	8087	Load Data
	ST(n)	11011001oo000mmm	8087	Load Data
		1101100111000001	8087	Load Data
FILD	ST(0),Mem16	11011111oo000mmm	8087	Load Data (Integer)
	Mem16	11011111oo000mmm	8087	Load Data (Integer)
	ST(0),Mem32	11011011oo000mmm	8087	Load Data (Integer)
	Mem32	11011011oo000mmm	8087	Load Data (Integer)
	ST(0),Mem64	11011111oo101mmm	8087	Load Data (Integer)
	Mem64	11011111oo101mmm	8087	Load Data (Integer)
FBLD	ST(0),Mem80	11011111oo100mmm	8087	Load Data (BCD)
	Mem80	11011111oo100mmm	8087	Load Data (BCD)
FLD1		1101100111101000	8087	Load + 1.0
FLDZ		1101100111101110	8087	Load + 0.0
FLDPI		1101100111101011	8087	Load PI
FLDL2E		1101100111101010	8087	Load log2 e
FLDL2T		1101100111101001	8087	Load log2 10
FLDLG2		1101100111101100	8087	Load log10 2
FLDLN2		1101100111101101	8087	Load loge 2

FLDCW	Mem16	11011001oo101mmm	8087	Load Control Register
FLDENV	Mem	11011001oo100mmm	8087	Load Enviroment
FMUL	ST(0),ST(n)	1101100011001rrr	8087	Multiplication
	ST(n),ST(0)	1101110011001rrr	8087	Multiplication
	ST(0),Mem32	11011000oo001mmm	8087	Multiplication
	Mem32	11011000oo001mmm	8087	Multiplication
	ST(0),Mem64	11011100oo001mmm	8087	Multiplication
	Mem64	11011100oo001mmm	8087	Multiplication
	ST(n)	1101110011001rrr	8087	Multiplication
		1101110011001001	8087	Multiplication
FMULP	ST(n),ST(0)	11011110oo001mmm	8087	Multiplication and Pop
	ST(n)	11011110oo001mmm	8087	Multiplication and Pop
		1101111011001001	8087	Multiplication and Pop
FIMUL	ST(0),Mem16	11011110oo001mmm	8087	Multiplication (Integer)
	Mem16	11011110oo001mmm	8087	Multiplication (Integer)
	ST(0),Mem32	11011010oo001mmm	8087	Multiplication (Integer)
	Mem32	11011010oo001mmm	8087	Multiplication (Integer)
FNOP		1101100111010000	8087	No Operation
FPATAN		1101100111110011	8087	Partial Arctangent
FPREM		1101100111111000	8087	Partial Remainder
FPREM1		1101100111110101	80387	Partial Remainder (IEEE)
FPTAN		1101100111110010	8087	Partial Tangent
FRNDINT		1101100111111100	8087	Round to Integer
FRSTOR	Mem112	11011101oo100mmm	8087	Restore State
FSAVE	Mem112	[FWAIT] 11011101oo110mmm	8087	Save Machine State
FNSAVE	Mem112	11011101oo110mmm	8087	Save Machine State
FSCALE		1101100111111101	8087	Scale
FSETPM		1101101111100100	80287	Set Protected Mode
FSIN		1101100111111110	80387	Sine
FSINCOS		1101100111111011	80387	Sine and Cosine
FSQRT		1101100111111010	8087	Square Root
FST	Mem32,ST(0)	11011001oo010mmm	8087	Store
	Mem32	11011001oo010mmm	8087	Store
	Mem64,ST(0)	11011101oo010mmm	8087	Store
	Mem64	11011101oo010mmm	8087	Store

	ST(n),ST(0)	11011101oo010mmm	8087	Store
	ST(n)	11011101oo010mmm	8087	Store
		1101110111010001	8087	Store
FSTP	Mem32,ST(0)	11011001oo011mmm	8087	Store and Pop
	Mem32	11011001oo011mmm	8087	Store and Pop
	Mem64,ST(0)	11011101oo011mmm	8087	Store and Pop
	Mem64	11011101oo011mmm	8087	Store and Pop
	Mem80,ST(0)	11011011oo111mmm	8087	Store and Pop
	Mem80	11011011oo111mmm	8087	Store and Pop
	ST(n),ST(0)	1101110111011rrr	8087	Store and Pop
	ST(n)	1101110111011rrr	8087	Store and Pop
		1101110111011001	8087	Store and Pop
FIST	Mem16,ST(0)	11011111oo010mmm	8087	Store (Integer)
	Mem16	11011111oo010mmm	8087	Store (Integer)
	Mem32,ST(0)	11011011oo010mmm	8087	Store (Integer)
	Mem32	11011011oo010mmm	8087	Store (Integer)
FISTP	Mem16,ST(0)	11011111oo011mmm	8087	Store (Integer) and Pop
	Mem16	11011111oo011mmm	8087	Store (Integer) and Pop
	Mem32,ST(0)	11011011oo011mmm	8087	Store (Integer) and Pop
	Mem32	11011011oo011mmm	8087	Store (Integer) and Pop
	Mem64,ST(0)	11011111oo111mmm	8087	Store (Integer) and Pop
	Mem64	11011111oo111mmm	8087	Store (Integer) and Pop
FBSTP	Mem80,ST(n)	11011111oo110mmm	8087	Store (BCD) and Pop
	Mem80	11011111oo110mmm	8087	Store (BCD) and Pop
FSTCW	Mem16	[FWAIT] 11011001oo111mmm	8087	Store Control Register
FNSTCW	Mem16	11011001oo111mmm	8087	Store Control Register
FSTENV	Mem	[FWAIT] 11011001oo110mmm	8087	Store Environment
FNSTENV	Mem	11011001oo110mmm	8087	Store Environment
FSTSW	Mem16	[FWAIT] 11011101oo111mmm	8087	Store Status Register
	AX	[FWAIT] 1101111111100000	8087	Store Status Register
FNSTSW	Mem16	11011101oo111mmm	8087	Store Status Register
	AX	1101111111100000	8087	Store Status Register
FSUB	ST(0),ST(n)	1101100011100rrr	8087	Subtraction
	ST(n),ST(0)	1101110011101rrr	8087	Subtraction

	ST(0),Mem32	11011000oo100mmm	8087	Subtraction
	Mem32	11011000oo100mmm	8087	Subtraction
	ST(0),Mem64	11011100oo100mmm	8087	Subtraction
	Mem64	11011100oo100mmm	8087	Subtraction
	ST(n)	1101110011101rrr	8087	Subtraction
		1101110011101001	8087	Subtraction
FSUBP	ST(n),ST(0)	11011110oo101mmm	8087	Subtraction and Pop
	ST(n)	11011110oo101mmm	8087	Subtraction and Pop
		1101111011101001	8087	Subtraction and Pop
FISUB	ST(0),Mem16	11011110oo100mmm	8087	Subtraction (Integer)
	Mem16	11011110oo100mmm	8087	Subtraction (Integer)
	ST(0),Mem32	11011010oo100mmm	8087	Subtraction (Integer)
	Mem32	11011010oo100mmm	8087	Subtraction (Integer)
FSUBR	ST(0),ST(n)	1101100011101rrr	8087	Reverse Subtraction
	ST(n),ST(0)	1101110011100rrr	8087	Reverse Subtraction
	ST(0),Mem32	11011000oo101mmm	8087	Reverse Subtraction
	Mem32	11011000oo101mmm	8087	Reverse Subtraction
	ST(0),Mem64	11011100oo101mmm	8087	Reverse Subtraction
	Mem64	11011100oo101mmm	8087	Reverse Subtraction
	ST(n)	1101110011100rrr	8087	Reverse Subtraction
		1101110011100001	8087	Reverse Subtraction
FSUBRP	ST(n),ST(0)	11011110oo100mmm	8087	Reverse Subtraction and Pop
	ST(n)	11011110oo100mmm	8087	Reverse Subtraction and Pop
		1101111011100001	8087	Reverse Subtraction and Pop
FISUBR	ST(0),Mem16	11011110oo101mmm	8087	Reverse Subtraction (Integer)
	Mem16	11011110oo101mmm	8087	Reverse Subtraction (Integer)
	ST(0),Mem32	11011010oo101mmm	8087	Reverse Subtraction (Integer)
	Mem32	11011010oo101mmm	8087	Reverse Subtraction (Integer)
FTST		1101100111100100	8087	Compare with 0.0
FUCOM	ST(0),ST(n)	1101110111100rrr	80387	Unordered Compare
	ST(n),ST(0)	1101110111100rrr	80387	Unordered Compare
	ST(n)	1101110111100rrr	80387	Unordered Compare
		1101110111100001	80387	Unordered Compare
FUCOMP	ST(0),ST(n)	1101110111101rrr	80387	Unordered Compare and Pop
	ST(n),ST(0)	1101110111101rrr	80387	Unordered Compare and Pop
	ST(n)	1101110111101rrr	80387	Unordered Compare and Pop

		1101110111101001	80387	Unordered Compare and Pop
FUCOMPP	ST(0),ST(n)	1101101011101rrr	80387	Unordered Compare and Pop and Pop
	ST(n),ST(0)	1101101011101rrr	80387	Unordered Compare and Pop and Pop
	ST(n)	1101101011101rrr	80387	Unordered Compare and Pop and Pop
		1101101011101001	80387	Unordered Compare and Pop and Pop
FWAIT		10011011	8086	Wait for FPU
FXAM		1101100111100101	8087	Examine
FXCH	ST(0),ST(n)	1101100111001rrr	8087	Exchange with Register
	ST(n),ST(0)	1101100111001rrr	8087	Exchange with Register
	ST(n)	1101100111001rrr	8087	Exchange with Register
		1101100111001001	8087	Exchange with Register
FXTRACT		1101100111110100	8087	Extract Components
FYL2X		1101100111110001	8087	ST(1) * log2 ST(0)
FYL2XP1		1101100111111001	8087	ST(1) * log2 (ST(0)+1.0)

Condition codes.

CCCC	Name	Means	In short
0000	O	overflow	o=1
0001	NO	Not overflow	o=0
0010	C/B/NAE	Carry, below, not above nor equal	c=1
0011	NC/AE/NB	Not carry, above or equal, not below	c=0
0100	E/Z	Equal, zero	z=1
0101	NE/NZ	Not equal, not zero	z=0
0110	BE/NA	Below or equal, not above	c=1 or z=1
0111	A/NBE	Above, not below nor equal	c=0 and z=0
1000	S	Sign (negative)	s=1
1001	NS	Not sign	s=0
1010	P/PE	Parity, parity even	p=1
1011	NP/PO	Not parity, parity odd	p=0
1100	L/NGE	Less, not greater nor equal	s<>o
1101	GE/NL	Greater or egual, not less	s=o

1110	LE/NG	Less or equal, not greater	z=1 or s<>o
1111	G/NLE	Greater, not less nor equal	z=0 and s=o

Condition codes for FCMOVcc.

cc	n	Name	Means	In short
00	0	C/B/NAE	Carry, below, not above nor equal	c=1
00	1	NC/AE/NB	Not carry, above or equal, not below	c=0
01	0	E/Z	Equal, zero	z=1
01	1	NE/NZ	Not equal, not zero	z=0
10	0	BE/NA	Below or equal, not above	c=1 or z=1
10	1	A/NBE	Above, not below nor equal	c=0 and z=0
11	0	U	Unordered	C2=1
11	1	NU	Not Unordered	C2=0