Instruction	Description
FADD	Floating-point addition
FDIV	Floating-point division
FDIVR	Reverse floating-point division
FMUL	Floating-point multiplication
FSUB	Floating-point subtraction
FSUBR	Reverse floating-point subtraction

`FADD source`: Add a 32- or 64-bit value from memory to the `ST0` register

`FADD %st(x), %st(0)`: Add `st(x)` to `st(0)` and store the result in `st(0)`

`FADD %st(0), %st(x)`: Add `st(0)` to `st(x)` and store the result in `st(x)`

`FADDP %st(0), %st(x)` : Add `st(0)` to `st(x)`, store the result in `st(x)`, and pop `st(0)`

`FADDP`: Add `st(0)` to `st(1)`, store the result in `st(1)`, and pop `st(0)`

`FIADD source`: Add a 16- or 32-bit integer value to `st(0)` and store the result in `st(0)`

Instruction	Description
F2XM1	Computes 2 to the power of the value in ST0, minus 1
FABS	Computes the absolute value of the value in ST0
FCHS	Changes the sign of the value in ST0
FCOS	Computes the cosine of the value in ST0
FPATAN	Computes the partial arctangent of the value in ST0
FPREM	Computes the partial remainders from dividing the value in ST0 by the value in ST1
FPREM1	Computes the IEEE partial remainders from dividing the value in ST0 by the value in ST1
FPTAN	Computes the partial tangent of the value in ST0
FRNDINT	Rounds the value in ST0 to the nearest integer
FSCALE	Computes ST0 to the ST1st power
FSIN	Computes the sine of the value in ST0
FSINCOS	Computes both the sine and cosine of the value in ST0
FSQRT	Computes the square root of the value in ST0
FYL2X	Computes the value ST1 * log ST0 (base 2 log)
FYL2XP1	Computes the value ST1 * log (ST0 + 1) (base 2 log)

Instruction	Description
FCOM	Compare the ST0 register with the ST1 register.
FCOM ST(x)	Compare the ST0 register with another FPU register.
FCOM source	Compare the ST0 register with a 32- or 64-bit memory value.
FCOMP	Compare the ST0 register with the ST1 register value and pop the stack.
FCOMP ST(x)	Compare the ST0 register with another FPU register value and pop the stack.
FCOMP source	Compare the ST0 register with a 32 or 64-bit memory value and pop the stack.
FCOMPP	Compare the ST0 register with the ST1 register and pop the stack twice.
FTST	Compare the ST0 register with the value 0.0.

The result of the comparison is set in the C0, C2, and C3 condition code bits of the status register. The possible results from the comparison are shown in the following table.

Condition	C3	C2	C0
ST0 > source	0	0	0
ST0 < source	0	0	1
ST0 = source	1	0	0

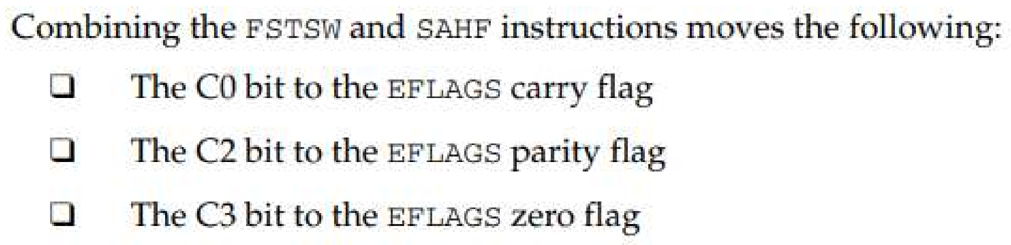
The control register uses a 16-bit register, with the bits shown in the following table.

Control Bits	Description
0	Invalid operation exception mask
1	Denormal operand exception mask
2	Zero divide exception mask
3	Overflow exception mask
4	Underflow exception mask
5	Precision exception mask
6–7	Reserved
8–9	Precision control
10–11	Rounding control
12	Infinity control
13–15	Reserved

FSTSW : Save status register value from FPU to AX

And

SAHF: Moves bit 0,2,4,6 and 7 to Carry, parity, aligned zero and sign flag respectively



Combining the `FSTSW` and `SAHF` instructions moves the following:

- ❑ The C0 bit to the `EFLAGS` carry flag
- ❑ The C2 bit to the `EFLAGS` parity flag
- ❑ The C3 bit to the `EFLAGS` zero flag