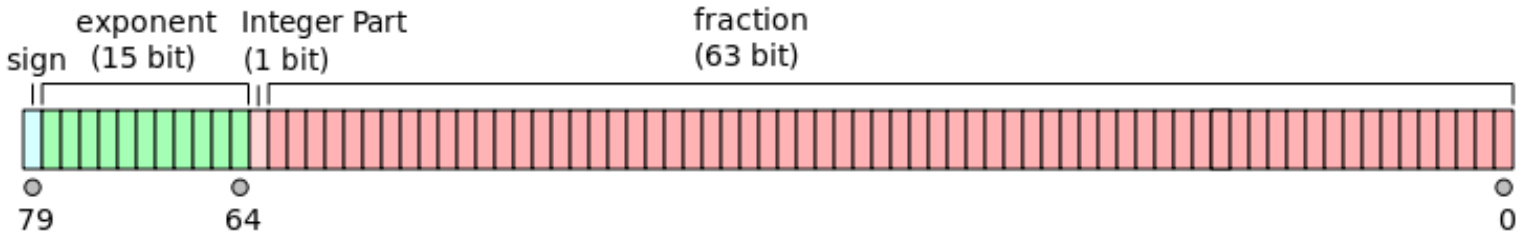
IA-32 Floating point Instructions

- Floating Point Unit (FPU) executes F.P. instructions
  - Has its own set of 8 registers
  - FPU & CPU execute their instructions concurrently
  - FPU uses 80-bit internal format for all operations
    - Converts IEEE-754 formatted operands to/from 80-bit format



Bit 79 is the sign bit (s)
Bits 64 – 78 give the 15-bit excess-16383 exponent (e)
Bit 63 = 1 for normalized values, = 0 for denormalized (i)
Bits 0 – 62 give the 63-bit fraction (f) [there is no hidden bit]
Number point is between bits 62 & 63
Value represented is $(-1)^s \times i.f \times 2^{e-16383}$

IA-32 Floating point Instructions

- Floating Point Stack
  - The 8 FPU registers are organized as a stack
    - ST(0) to ST(7) from top to bottom
    - Modulo-8 pointer defines top of stack
  - FLD copies its single memory operand onto stack ST(0)
  - FST writes contents of ST(0) into memory
    - FSTP does the same but also pops ST(0) from stack
  - FIST converts ST(0) to 32-bit integer & stores in memory
    - FISTP does same but also pops ST(0)

IA-32 Floating point Instructions

- Floating Point Load/Store examples
  - FLD DWORD PTR[EAX]
    - Converts 32-bit float in memory into 80-bit float in ST(0)
    - 80-bit value is pushed onto FPU stack
  - FST QWORD PTR [EDX + 8]
    - converts 80-bit float in ST(0) into 64-bit float in memory
  - FST writes contents of ST(0) into memory
    - FSTP does the same but also pops ST(0) from stack
  - FISTP [EDX + 8]
    - Pops ST(0) converts to 32-bit integer and stores in memory

IA-32 Floating point Instructions

- Floating Point Arithmetic Instructions
  - FADD QWORD PTR[EAX]
    - Converts 32-bit float in memory into 80-bit float & adds to ST(0)
  - FSUBP ST(1),ST(0)
    - puts ST(1)-ST(0) into ST(1) , pops ST(0), so result is now ST(0)
  - FSUBR ST(2),ST(0)
    - Puts ST(0) – ST(2) into ST(2) [reverse subtract]
    - FDIVR is similar
  - FISUB DWORD PTR [EDX + 8]
    - Converts 32-bit integer, sets ST(0) = ST(0) – float equivalent
    - FIADD, FIMUL & FIDIV are similar

IA-32 Floating point Instructions

- Floating Point Compare Instructions
  - FCOMI ST(0),ST(4)
    - Compares register with ST(0) and sets condition codes
    - ST(0) must be destination operand
  - FCOMIP ST(0),ST(2)
    - Similar to FUCOMI but also pops stack
  - Additional Float Instructions
    - FCHS changes sign of implicit operand ST(0)
    - FABS takes absolute value
    - FSQRT computes square root of ST(0)
    - FSIN & FCOS compute sine and cosine of ST(0)
    - FLDZ & FLD1 push 0.0 & 1.0 as 80-bit float
    - FLDPI pushes π as 80-bit float