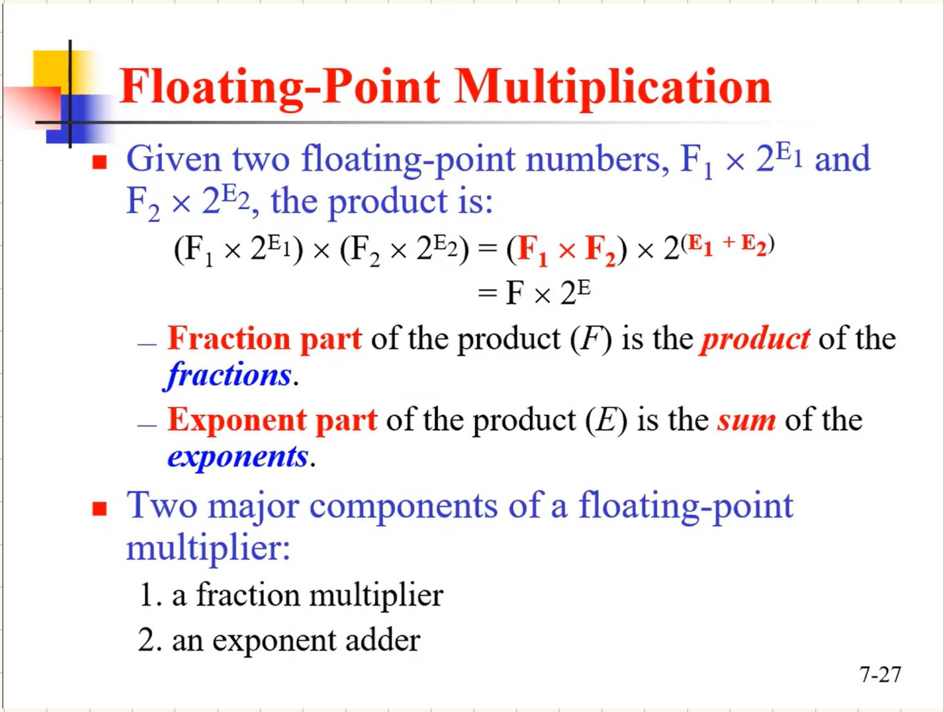
7.2 FLP Multiplication

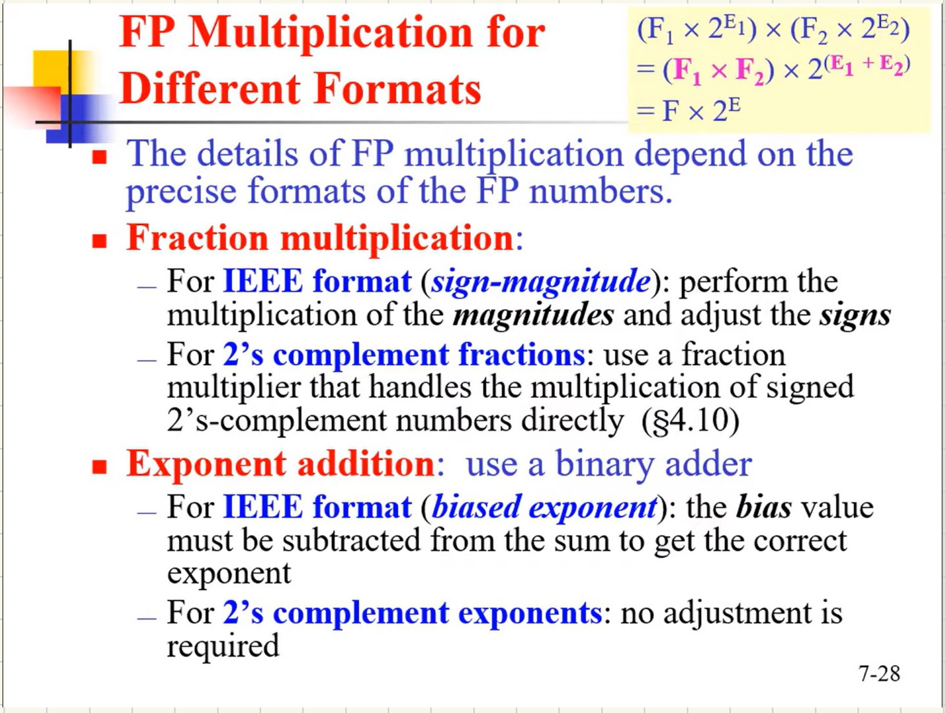


Floating-Point Multiplication

- Given two floating-point numbers, $F_1 \times 2^{E_1}$ and $F_2 \times 2^{E_2}$, the product is:

  $$(F_1 \times 2^{E_1}) \times (F_2 \times 2^{E_2}) = (F_1 \times F_2) \times 2^{(E_1 + E_2)}$$
  $$= F \times 2^E$$

  - **Fraction part** of the product (*F*) is the ***product*** of the ***fractions***.
  - **Exponent part** of the product (*E*) is the ***sum*** of the ***exponents***.
- Two major components of a floating-point multiplier:
  1. a fraction multiplier
  2. an exponent adder



FP Multiplication for Different Formats

$(F_1 \times 2^{E_1}) \times (F_2 \times 2^{E_2})$
$= (F_1 \times F_2) \times 2^{(E_1 + E_2)}$
$= F \times 2^E$

- The details of FP multiplication depend on the precise formats of the FP numbers.
- **Fraction multiplication**:
  - For **IEEE format** (***sign-magnitude***): perform the multiplication of the ***magnitudes*** and adjust the ***signs***
  - For **2's complement fractions**: use a fraction multiplier that handles the multiplication of signed 2's-complement numbers directly (§4.10)
- **Exponent addition**: use a binary adder
  - For **IEEE format** (***biased exponent***): the ***bias*** value must be subtracted from the sum to get the correct exponent
  - For **2's complement exponents**: no adjustment is required

- The ***2's complement system*** has several interesting properties for performing arithmetic.
  - Many FP arithmetic units convert the IEEE notation to 2's complement and then use the 2's complement internally for carrying out the FP operations.
  - Then the final result is converted back to IEEE standard notation.

General Proce
Multiplication

* Assumptions:
1. The two numbers are properly ***normalized*** to start with.
2. The final result has to be ***normalized***.

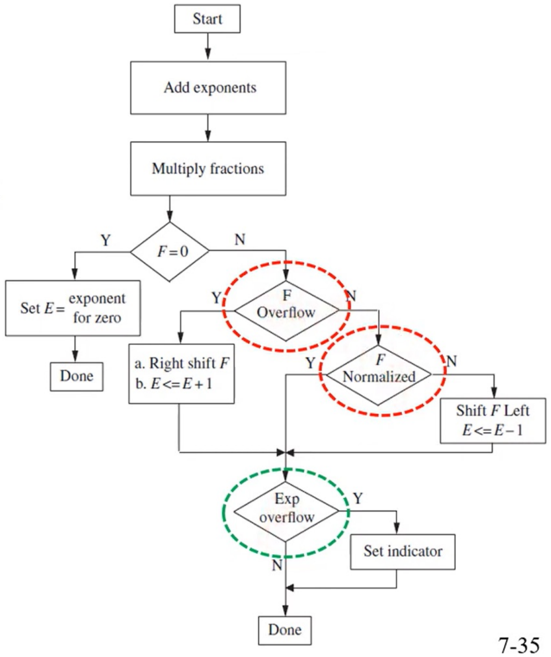
- General procedure f

$$(F_1 \times 2^{E_1}) \times (F_2 \times 2^{E_2}) = (\mathbf{F_1} \times \mathbf{F_2}) \times 2^{(\mathbf{E_1} + \mathbf{E_2})}$$

1. Add the two exponents.
2. Multiply the two fractions (significands).
3. If the product is 0, adjust the representation to the proper representation for 0.
4. Normalization:
   a. If the product fraction is too big, normalize by shifting it right and incrementing the exponent.
   b. If the product fraction is too small, normalize by shifting left and decrementing the exponent.
5. If an exponent underflow or overflow occurs, generate an exception or error indicator.
6. Round to the appropriate # of bits. If rounding resulted in loss of normalization, go to step 4 again.

Flowchart

- Flowchart for floating-point multiplier:


Start
Add exponents
Multiply fractions
$F=0$
Y
N
Set $E=$ exponent for zero
Done
F Overflow
Y
N
a. Right shift F
b. $E<=E+1$
F Normalized
Y
N
Shift F Left
$E<=E-1$
Exp overflow
Y
N
Set indicator
Done