

CSE 321b

Computer Organization (2)

تنظيم الحاسب (2)



3rd year, Computer Engineering
Winter 2017

Lecture #10



Dr. Hazem Ibrahim Shehata

Dept. of Computer & Systems Engineering

Credits to Dr. Ahmed Abdul-Monem Ahmed for the slides

Adminstrivia

- Assignment #3:
 - To be released early next week

Website: <http://hshehata.github.io/courses/zu/cse321b/>

Office hours: Sunday 11:30am – 12:30pm

Chapter 10. Computer Arithmetic (*Cont.*)

Outline

- Integer Representation
 - Sign-Magnitude, Two's Complement, Biased
- Integer Arithmetic
 - Negation, Addition, Subtraction
 - Multiplication, Division
- Floating-Point Representation
 - IEEE 754
- Floating-Point Arithmetic
 - Addition, Subtraction
 - Multiplication, Division
 - Rounding

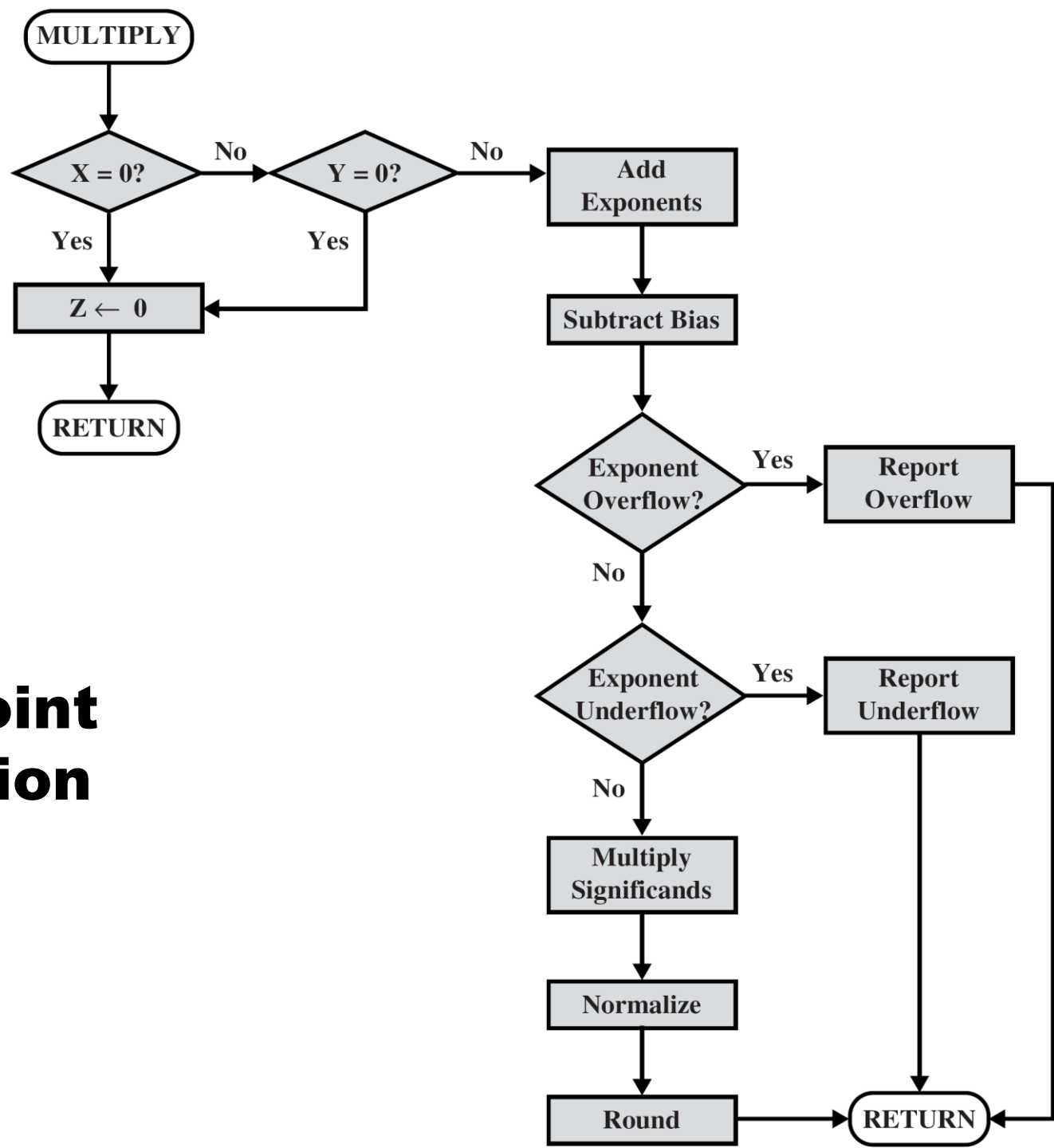
FP Arithmetic +/-

- Algorithm:
 1. Check for zeros (and other special cases, e.g., NaN).
 2. Align significands (adjusting exponents).
 3. Add or subtract significands.
 4. Normalize result.
 5. Round result.

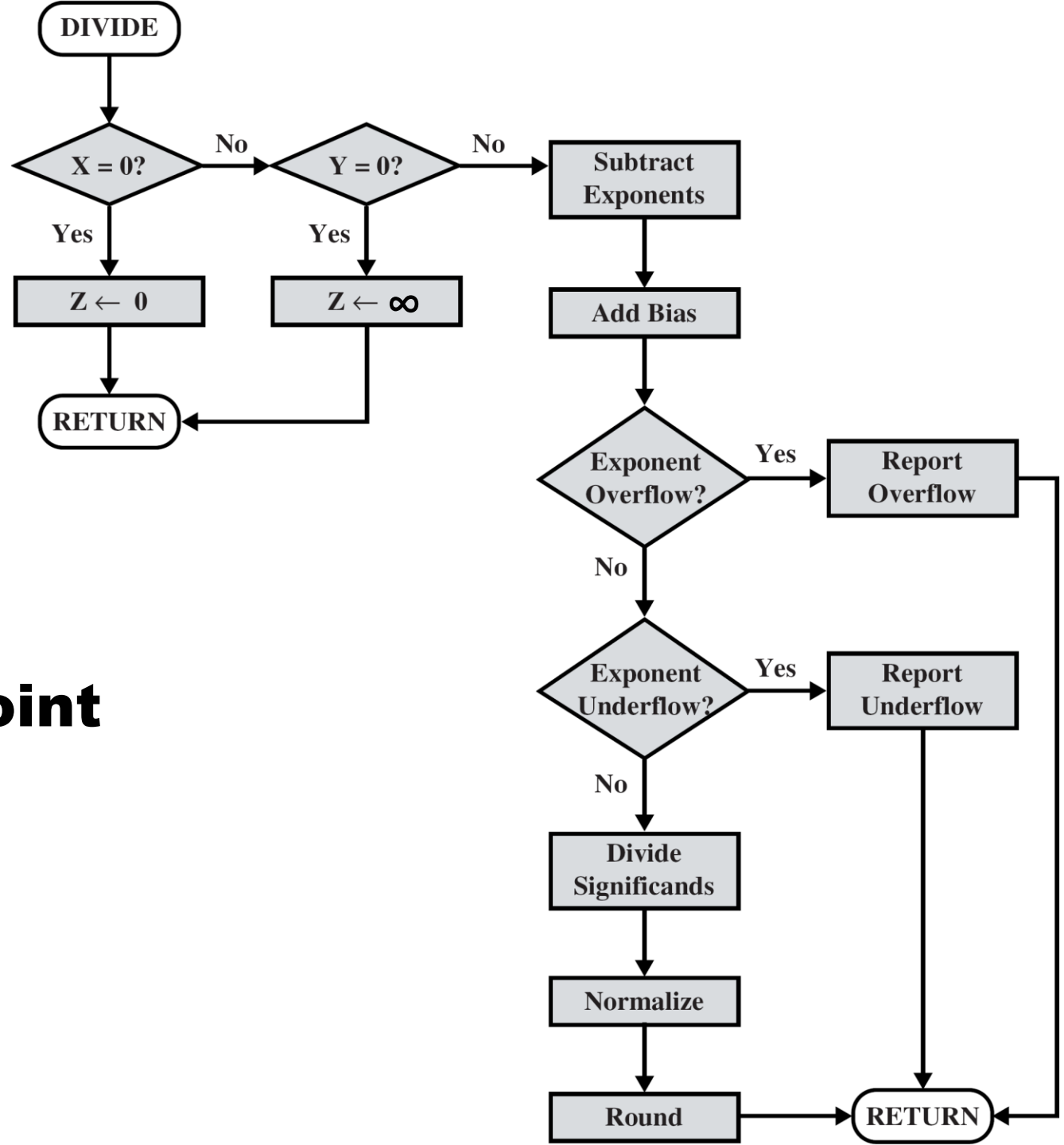
FP Arithmetic \times/\div

- Algorithm:
 1. Check for zeros (and other special cases, e.g., NaN).
 2. Add/subtract exponents.
 3. Multiply/divide significands (watch sign).
 4. Normalize result.
 5. Round result.
- All intermediate results should be in double length storage.

Floating Point Multiplication



Floating Point Division



Guard Bits

- Extra bits added to the right of the mantissa during intermediate calculations.
- Maintains good precision.

$$\begin{array}{r} 1.000 \dots 00 \times 2^1 \\ - 1.111 \dots 11 \times 2^0 \end{array}$$

$$\begin{array}{r} 1.000 \dots 00 \times 2^1 \\ - 0.111 \dots 11 \times 2^1 \\ \hline 0.000 \dots 01 \times 2^1 \\ = 2^{-23} \times 2^1 = \underline{\underline{2^{-22}}} \end{array}$$

$$\begin{array}{r} 1.000 \dots 00 \text{ } 0000 \times 2^1 \\ - 1.111 \dots 11 \text{ } 0000 \times 2^0 \end{array}$$

$$\begin{array}{r} 1.000 \dots 00 \text{ } 0000 \times 2^1 \\ - 0.111 \dots 11 \text{ } 1000 \times 2^1 \\ \hline 0.000 \dots 00 \text{ } 1000 \times 2^1 \\ = 2^{-24} \times 2^1 = \underline{\underline{2^{-23}}} \end{array}$$

Rounding

- The result of any operation on significands is stored in a longer register.
- When the result is to be stored as an FP number, extra bits have to be dropped off → rounding.
- Round to nearest representable number.
- Round toward $+\infty$: **round up** to the next number.
 - Ex.: $+1.1...001\ 001 \rightarrow +1.1...010$
 - Ex.: $-1.1...001\ 001 \rightarrow -1.1...001$
- Round toward $-\infty$: **round down** to the next number.
 - Ex.: $+1.1...001\ 001 \rightarrow +1.1...001$
 - Ex.: $-1.1...001\ 001 \rightarrow -1.1...010$
- Round toward zero: **truncate** the extra bits.
 - Ex.: $+1.1...001\ 001 \rightarrow +1.1...001$
 - Ex.: $-1.1...001\ 001 \rightarrow -1.1...001$

Round to Nearest

- Default technique listed in the IEEE standard.
- Deliver the representable value nearest to the infinitely precise result. If the two nearest representable values are equally near, the one with LSB 0 will be delivered.
- Examples:
 - If the guard bits are 10010 → they amount to more than one half of the last representable bit position → **Round away from zero.**
 - If the guard bits are 01111 → they amount to less than one half of the last representable bit position → **Truncate.**
 - If the guard bits are 10000 → midway
 - If we always truncate → biased toward zero.
 - If we choose randomly → not predictable/deterministic results.
 - IEEE standard:
 - + Force the result to be even.
 - + If last bit is 1, round away from zero, else, truncate.

Round to $\pm\infty$

- Useful in implementing interval arithmetic.
- **Interval arithmetic**: produce two values for every result. These two values correspond to the lower and upper endpoints of an interval that contains the true result.
- Used in monitoring and controlling errors.

Reading Material

- Stallings, Chapter 10:
 - Pages 352-356