Computer Organization

605.204

Module Two

Part Three

Floating Point Numbers

Module Two

- Part Three

- In this presentation, we are going to talk about :

- Floating Point Numbers

Previously

- Previously we talked about:

- Integer Addition and Subtraction
- Overflow
- Integer Multiplication and Division

- Now: Floating Point Numbers

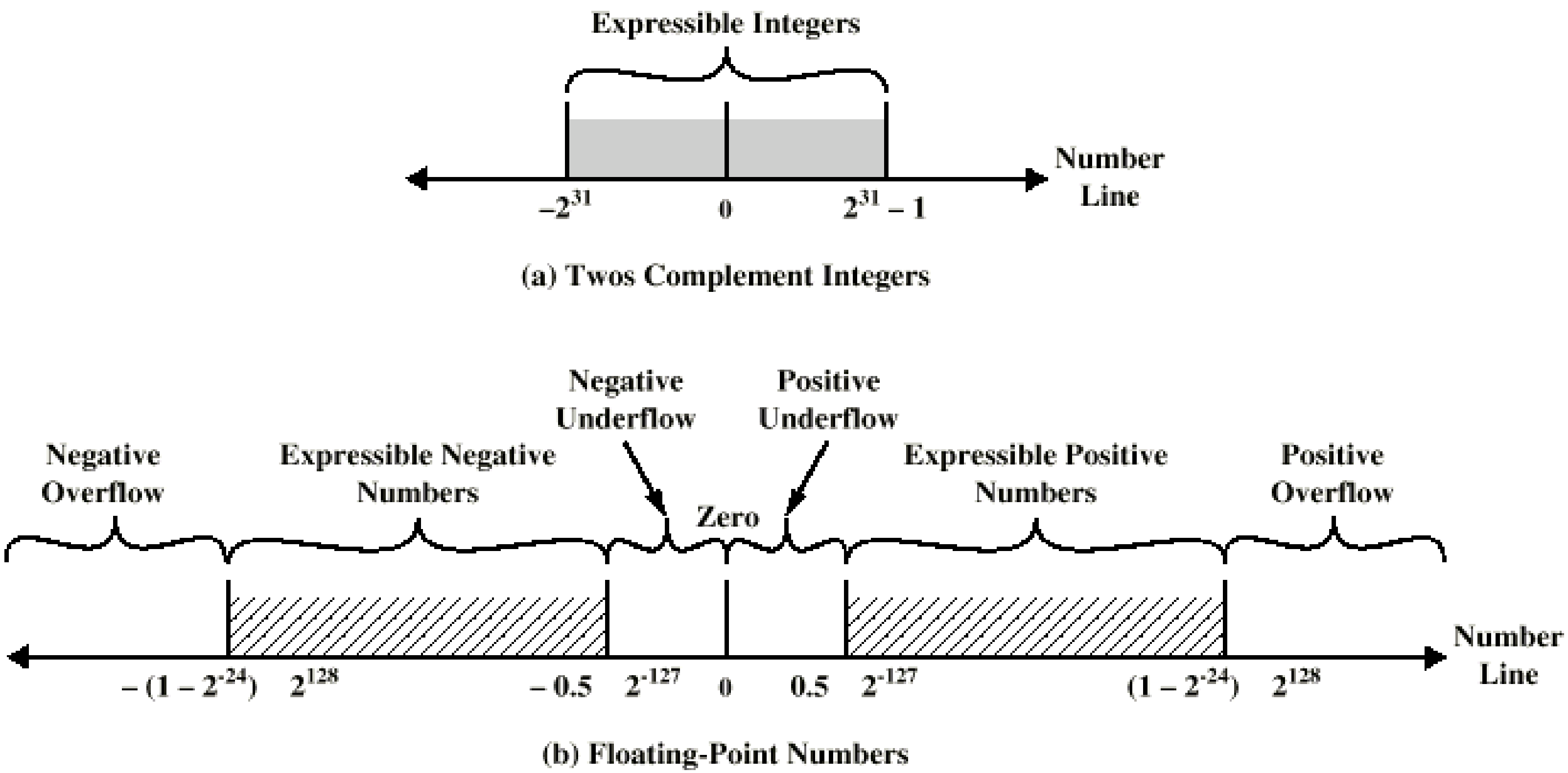
Floating Point

- We need a way to represent:
  - numbers with fractions, - 3.1415926
  - very small numbers, - .000000001
  - very large numbers, - 3,155,760,000,000 or $3.15576 * 10^{12}$
- Representation:
  - sign, exponent, fraction: $(-1)^{sign} * fraction * 2^{exponent}$

Sign	Biased Exponent	Fraction

  - more bits for fraction gives more accuracy
  - more bits for exponent increases range

Floating Point



Floating Point Formats

- IEEE 754
  - single precision: 8 bit exponent, 23 bit fraction
  - double precision: 11 bit exponent, 52 bit fraction
  - value: sign(-1) x 1+fraction x 2(exponent-127)

- VAX 11780
  - single precision: 8 bit exponent, 23 bit fraction
  - value: sign(-1) x fraction x 2(exponent-127)

IEEE 754 floating-point Standard

- Leading “1+” bit is implied
- Exponent is “biased” to make sorting easier
  - all 0s is smallest exponent all 1s is largest
  - bias of 127 for single precision and 1023 for double precision
- Value: $(-1)^{sign} * \underline{(1+\text{fraction})} * 2^{\text{exponent} - \text{bias}}$
- Example:
  - decimal: $-.75 = -3/4 = -3/2^2$
  - binary: $-.11 = -1.1 \times 2^{-1}$
  - floating point: exponent = 126 = 01111110
  - IEEE single precision:

```
1 01111110 10000000000000000000000
```

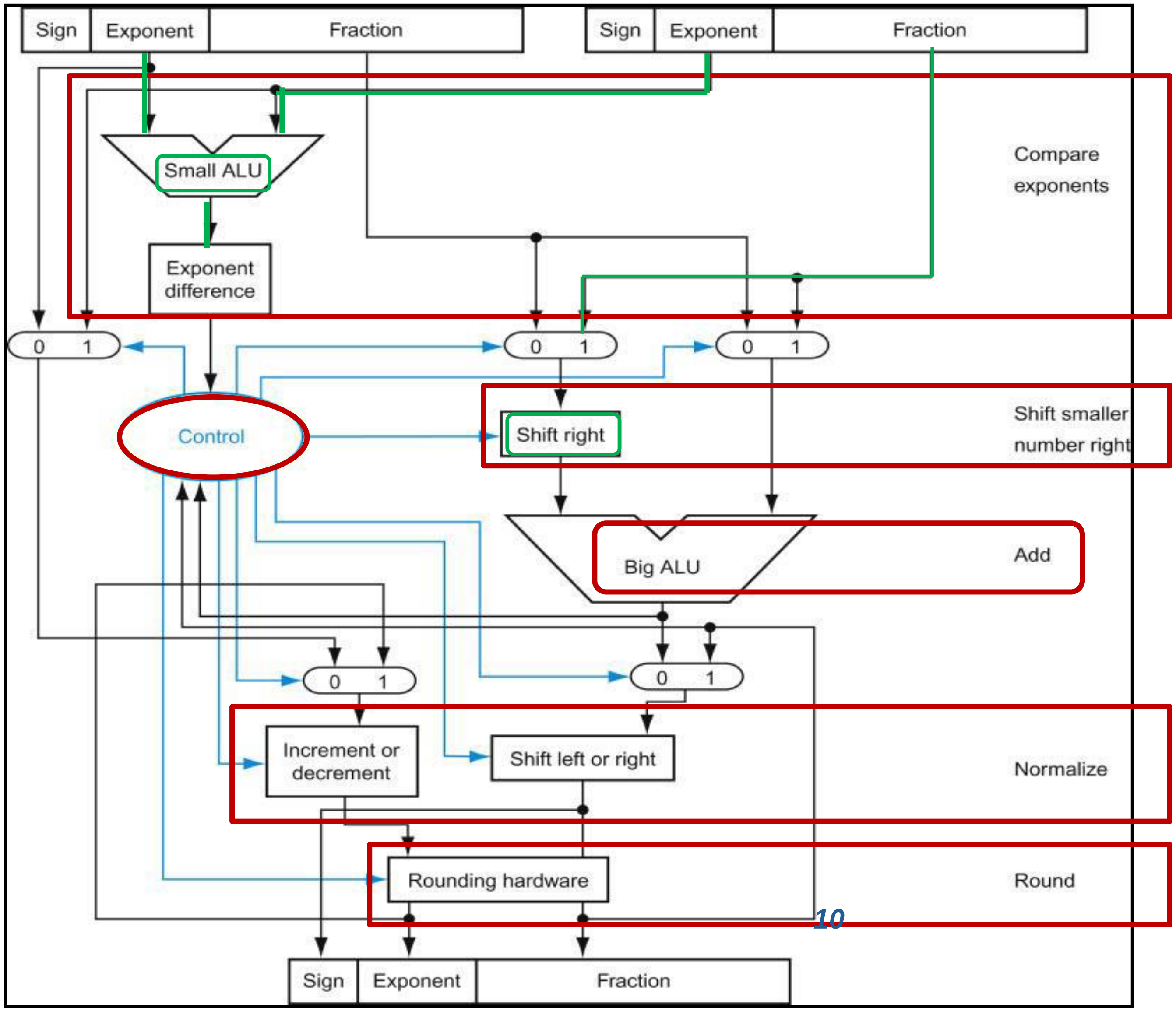
Normalization

- Floating Point numbers are usually normalized
- Exponent is adjusted so that leading bit (MSB) is 1
- Since it is always 1 there is no need to store it
- Scientific notation numbers are normalized to give a single digit before the decimal point. e.g. 3.123×10^3
- WHY Normalize ?
  - Standard representation of numerical value.
  - Simpler exchange of data.
  - Simpler algorithms and hardware.
  - Increases the accuracy of the information.

Floating Point Complexities

- Operations are more complicated,
  - Fields must be separated,
  - Exponents adjusted,
  - Fractions processed,   the actual calculations
  - Fields reassembled,
  - Value normalized.

Floating Point Calculations

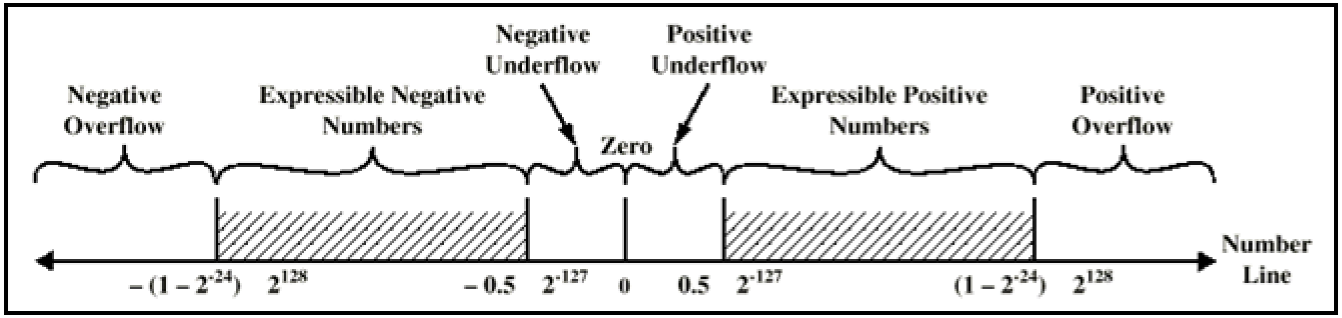


Addition

- Extract Fields
- Exponents
- Shift fraction
- Add fractions
- Normalize
- Round
- Iterate

Floating Point Complexities

- In addition to “overflow” we can have “underflow”
  - Exponent value too large > 255 - overflow
  - Exponent value too small < 0 - underflow



Floating Point Complexities

- There are as many numbers between zero and one as between one and infinity. Most values are just close approximations.
- Floating point addition is not Associative.
  - A + (B + C) =! (A + B) + C
- Implementing the IEEE 754 standard can be tricky
- Not using the standard can be even worse
  - See section 3.9, Fallacies and Pitfalls page 231, in the textbook for a description of Intel and Pentium bug! of July 1994 to December 1994.

Floating Point Complexities

- Accuracy can be a big problem
  - IEEE 754 keeps two extra bits, Guard and Round
  - Four rounding modes
    - Up
    - Down
    - Truncate
    - Toward nearest even number (rightmost bit is zero)
  - Positive divided by zero yields “infinity”
  - Zero divide by zero yields “not a number”
  - Other accuracy issues (See page 218 in the textbook)

More Floating Point Complexities

- Denormalized numbers
  - Exponent is all zeros Fraction non zero
  - Allows for gradual underflow
- Infinity
  - Exponent all ones Fraction all zeros
- Not a Number
  - Exponent all ones Fraction non zero
  - Positive divided by zero yields “infinity”
  - Zero divide by zero yields “not a number”

Floating Point Summary

- IEEE 754 format for real numbers.
- Calculations - Complex
- Overflow - Exponent value too large for number of digits
- Underflow - Exponent less than zero
- Accuracy - Normalize values
- Arithmetic - Not Associative

Summary

- Computer arithmetic is constrained by limited precision
- Bit patterns have no inherent meaning but standards do exist
  - Two’s Complement
  - IEEE 754 floating point
- Computer instructions determine “meaning” of the bit patterns
- Performance and accuracy are important, there are many complexities in real machines (algorithms and implementation).