MODULE 2: Data Representation

Lecture 2.4
Floating Point Representation

Prepared By:
- Scott F. Midkiff, PhD
- Luiz A. DaSilva, PhD
- Kendall E. Giles, PhD

Electrical and Computer Engineering
Virginia Tech

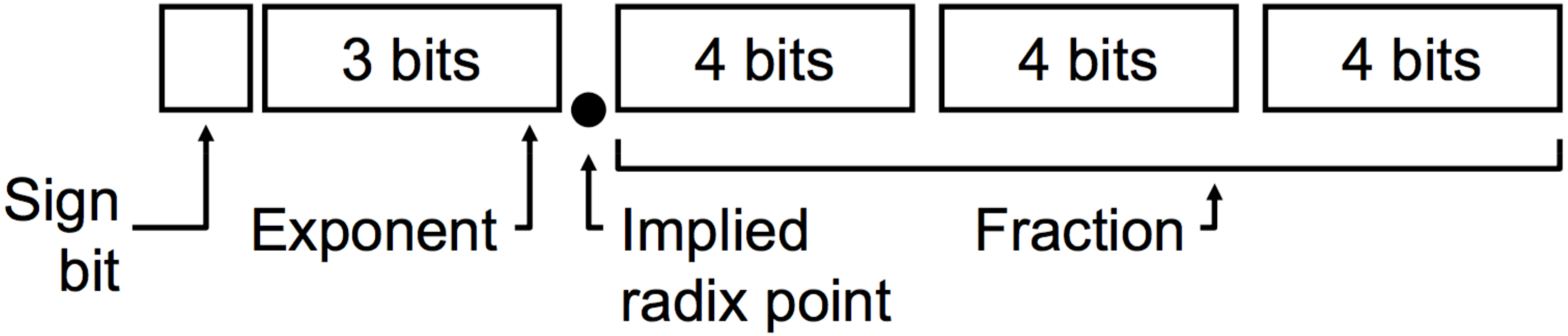
Lecture 2.4 Objectives

- Describe the rationale for using floating point representation
- Describe the role of the fraction in determining the precision, and of the exponent in determining the range of a floating point number
- Represent values as floating point numbers and interpret values stored as floating point numbers
- Describe the difference between single and double precision formats in the IEEE 754 standard
- Express a decimal number using the single precision IEEE 754 standard

Fixed versus Floating Point Numbers

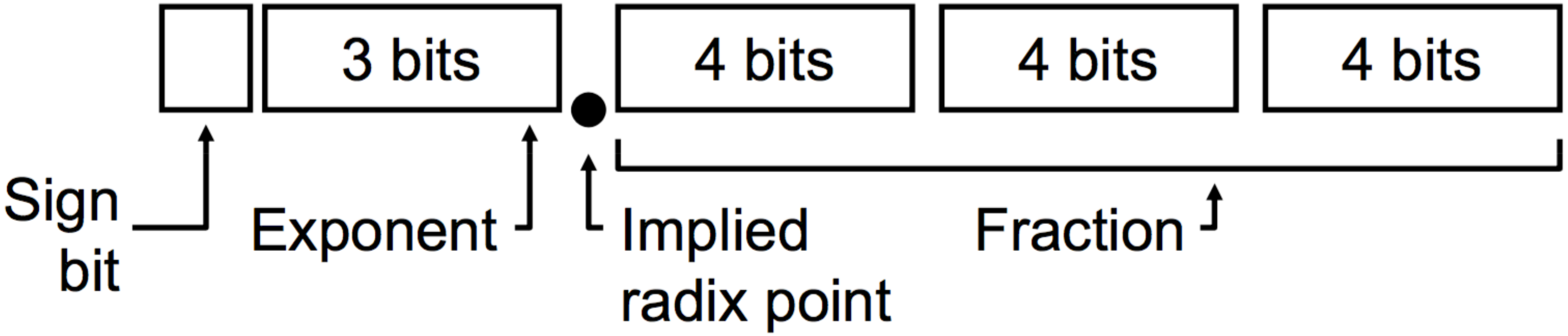
- Fixed point numbers have:
  - Fixed-size integer and fraction parts and, therefore,
  - Fixed range and precision
- Floating point numbers use a fraction and an exponent as two separate parts
  - Fraction provides precision
  - Exponent provides range

Example Representation (1)



- Sign bit indicates if the overall value is positive (S = 0) or negative (S = 1)
- Three-bit exponent uses an “excess-4” representation
  - Value stored is exponent + 4
  - Example: Exponent = -1 stored as (-1 + 4) = 3 = $(011)_2$

Example Representation (2)



- Fraction part of value stored as 3 radix-16 digits
- Implied radix point occurs to the left of all fraction digits
  - Normalized values: Most significant non-zero digit of the fraction is just to the right of the radix point

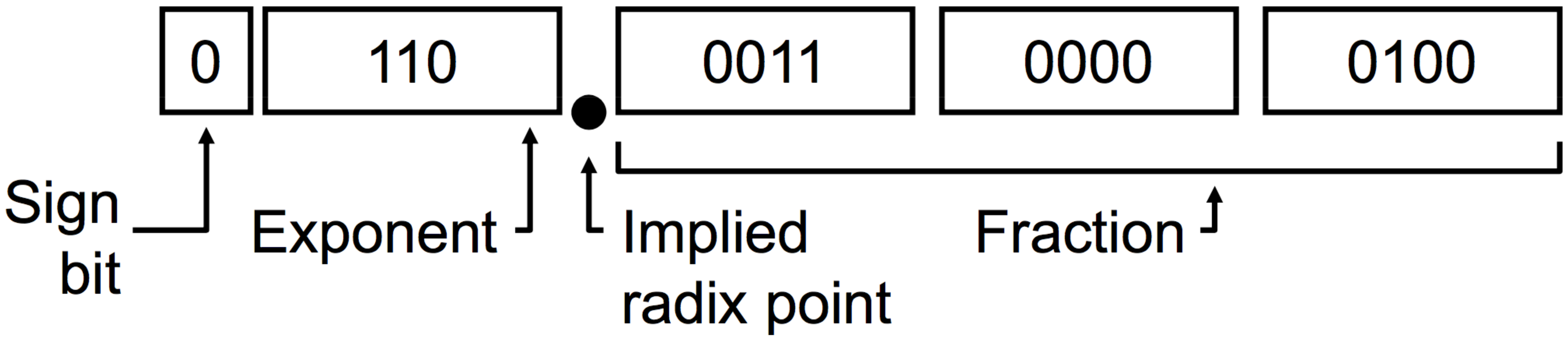
First Conversion Example (1)

- Let's represent 48.25 using the example floating point representation
- First, convert to binary representation
  - Integer part: $48 = (110000)_2$
  - Fraction part: $0.25 = (0.01)_2$
  - Value is $48.25 = (110000.01)_2$
- Second, form four-bit groups of 1's (radix-16 digits)
  - $48.25 = (0011\ 0000\ .\ 0100)_2 = (30.4)_{16}$

First Conversion Example (2)

- Third, normalize the value so that the most significant radix-16 digit is just to right of the radix point
  - $(30.4)_{16} = 0.304 \times 16^2$
  - In binary, fraction is 0011 0000 0100
- Fourth, determine binary representation of exponent (Exponent = 2)
  - Using excess-4 representation, represent exponent as $(2 + 4) = 6 = (110)_2$

First Conversion Example (3)



- Finally, put the sign (S = 0), exponent, and fraction together
  - Stored value is: 0 110 0011 0000 0100

CHECK POINT

As a checkpoint of your understanding, please pause the video and make sure you can do the following:

- Represent -48.25_{10} in our example floating point representation

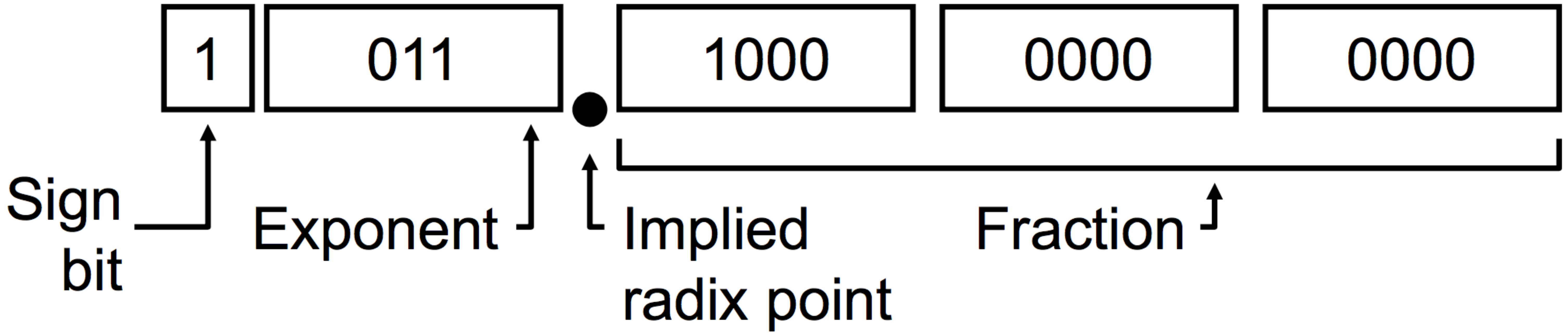
CHECK POINT

Answer:

- We determined 48.25_{10} is 0 110 0011 0000 0100 in our example floating point representation
- The magnitude of -48.25 is the same as 48.25. Since -48.25 is a negative number, the sign bit in our floating point representation will be 1
- So -48.25_{10} is 1 110 0011 0000 0100 in our example floating point representation

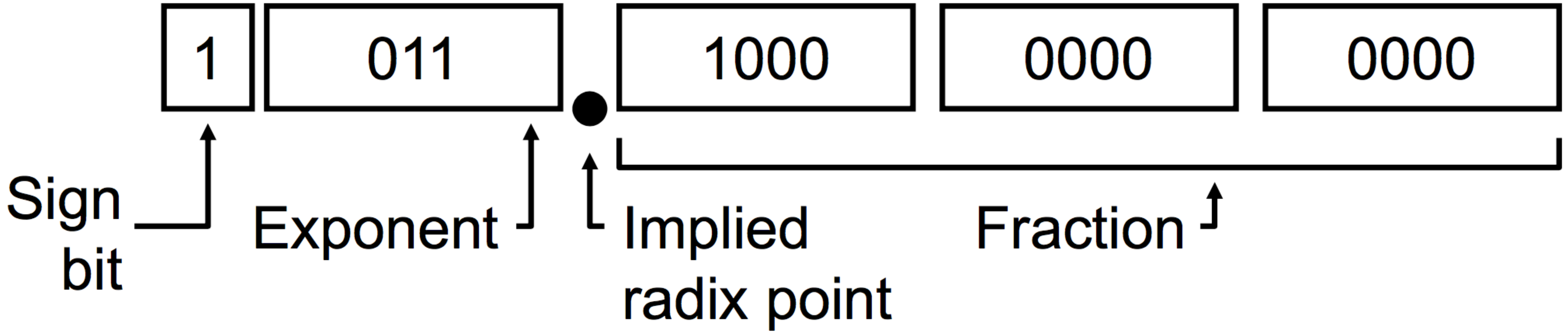
If you have any difficulties, please review the lecture video before continuing.

Second Conversion Example (1)



- Convert the following value to decimal: 1 011 1000 0000 0000
- Note that the sign bit is S = 1, so the final value will be negative

Second Conversion Example (2)



- Represented exponent value is $(011)_2$ = 3, so actual Exponent = (3-4) = -1
- Radix-16 fraction = $(800)_{16}$
- Value, as a radix-16 number = -0.800×16^{-1} or
- $(-0.08)_{16} = (-0\ .\ 0000\ 1000)_2$
- So, decimal value is $-2^{-5} = -1/32 = (-0.03125)_{10}$

IEEE 754 Floating Point Standard

- Two representations
  - 32-bit—“single precision”
  - 64-bit—“double precision”

IEEE 754 Single Precision Representation

- S: sign of number (1 bit)
- E: exponent (8 bits)
  - Excess-127 exponent—real exponent is E-127
- F: Fraction or mantissa or magnitude (23 bits)
  - Binary fraction with assumed leading 1

S	exponent (E)	fraction (F)
1	8	23

IEEE Standard: Example

$$(-17.5)_{10} = (-10001.1)_2$$

$$-10001.1_2 = -1.00011 \times 2^4$$

$$F = 0001100...000$$

$$E = 4+127 = 131 = 1000\ 0011$$

S	E	F
1	1000 0011	000 1100 0000 0000 0000 0000

CHECK POINT

As a checkpoint of your understanding, please pause the video and make sure you can do the following:

- What is 17_{10} in the IEEE 754 Single Precision Representation?

CHECK POINT

Answer:

- $17_{10} = 10001_2 = 1.0001 \times 2^4$
- S = 0
- E = 4 + 127 = 131 = 10000011_2
- F = 000100…000
- Therefore, 17_{10} is 0 1000 0011 000 1000 0000 0000 0000 0000

If you have any difficulties, please review the lecture video before continuing.

Summary

- Floating point representation provides a way to maintain precision over a large range of values
- Floating point representations typically contain a sign bit, an exponent, and a fraction
- Example representation presented based on radix-16 representation
- Most contemporary computers use the IEEE 754 floating point standard
    - Based on radix-2 representation

MODULE 2: Data Representation

Lecture 2.4
Floating Point Representation

Prepared By:

- Scott F. Midkiff, PhD
- Luiz A. DaSilva, PhD
- Kendall E. Giles, PhD

Electrical and Computer Engineering

Virginia Tech