MSDL
Modelling, Simulation and Design Lab

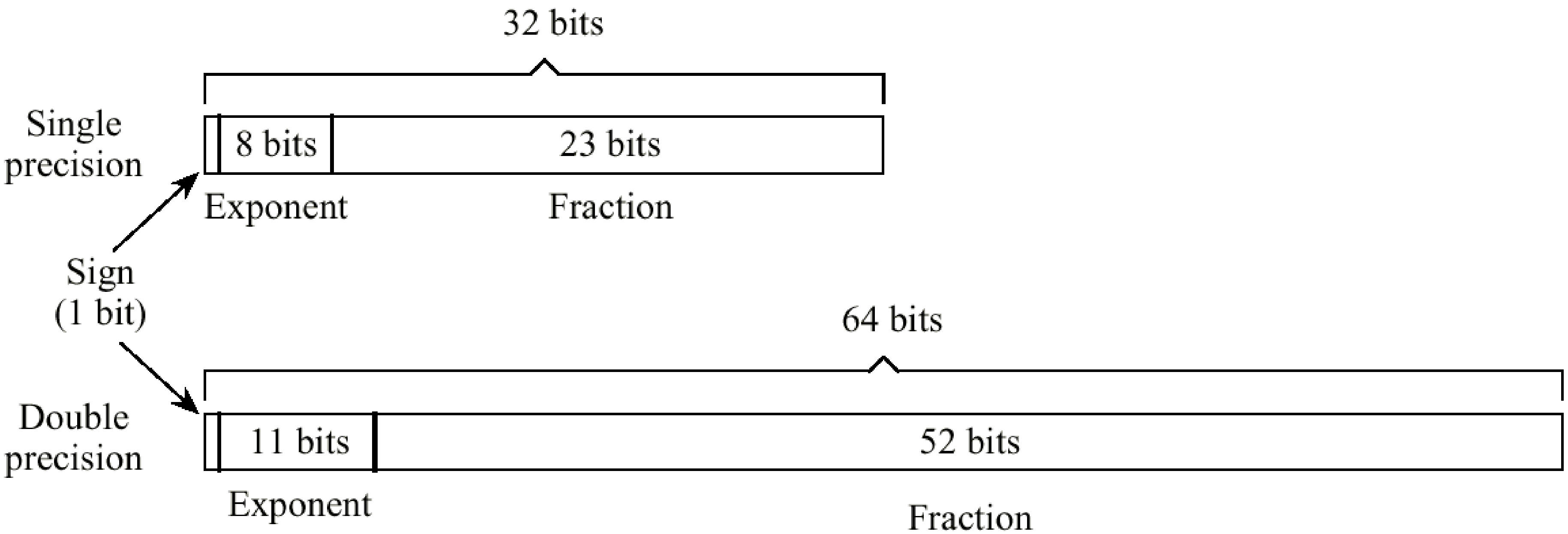
Floating Point – IEEE-754

October 20/23, 2009

Computersystemen en –architectuur

2009 –2010

Floating Point Formats



exponent:
- excess 127
- ! not 128 (why?)

fraction:
- implicitly preceded by “1.” (why?)

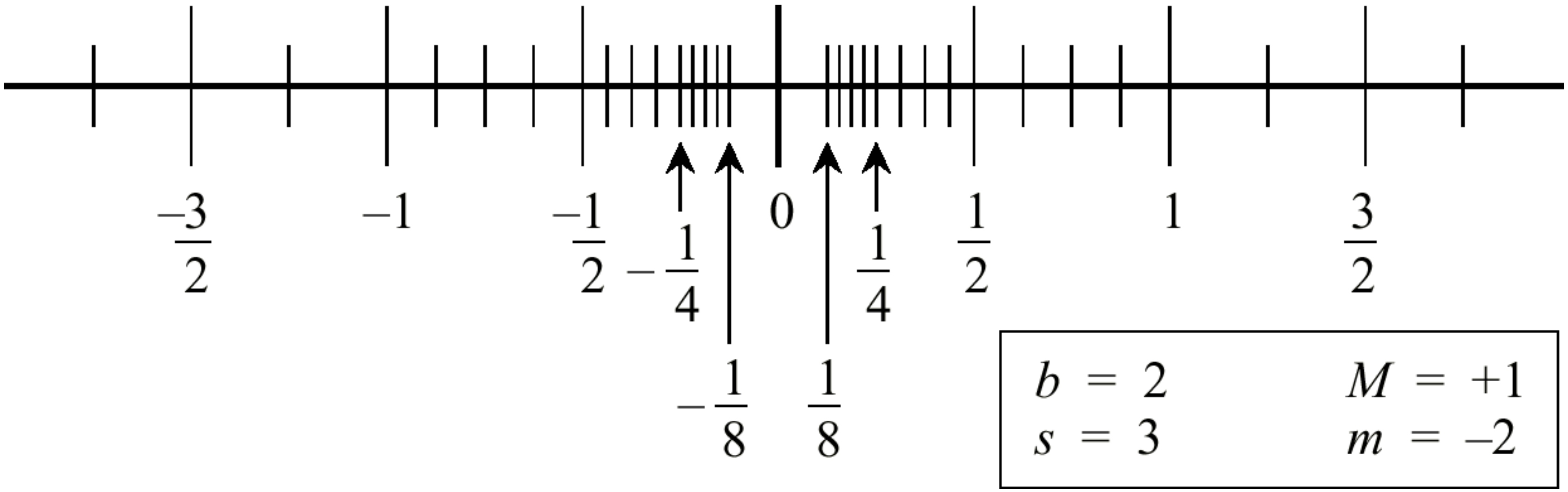
Conversion Example

- Represent -12.625_{10} in single precision IEEE-754 format.
- Step #1: Convert to target base. $-12.625_{10} = -1100.101_2$
- Step #2: Normalize. $-1100.101_2 = -1.100101_2 \times 2^3$
- Step #3: Fill in bit fields.
- Sign is negative, so sign bit is 1. Exponent is in excess 127 (not excess 128!), so exponent is represented as the unsigned integer $3 + 127 = 130$. Leading 1 of significand is hidden, so final bit pattern is:

  1 1000 0010 . 1001 0100 0000 0000 0000 000

Filling the gap: denormalized numbers

- Normalization
  - Drawback: gap between 0 and “most precize numbers next to 0”



- Solution: allow denormalized fractions
  - implicitly preceeded by 0
  - “virtual” exponent is smallest possible
  - apply this for “some chosen exponent” -> 0

Types of numbers

Type	Exponent	Fraction
Zeroes	0	0
Denormalized numbers	0	non zero
Normalized numbers	1 to $2^e - 2$	any
Infinities	$2^e - 1$	0
NaNs	$2^e - 1$	non zero

Examples

	Value	Bit Pattern		
		Sign	Exponent	Fraction
(a)	$+1.101 \times 2^{5}$	0	1000 0100	101 0000 0000 0000 0000 0000
(b)	-1.01011×2^{-126}	1	0000 0001	010 1100 0000 0000 0000 0000
(c)	$+1.0 \times 2^{127}$	0	1111 1110	000 0000 0000 0000 0000 0000
(d)	$+0$	0	0000 0000	000 0000 0000 0000 0000 0000
(e)	-0	1	0000 0000	000 0000 0000 0000 0000 0000
(f)	$+\infty$	0	1111 1111	000 0000 0000 0000 0000 0000
(g)	$+2^{-128}$	0	0000 0000	010 0000 0000 0000 0000 0000
(h)	+NaN	0	1111 1111	011 0111 0000 0000 0000 0000
(i)	$+2^{-128}$	0	011 0111 1111	0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000 0000