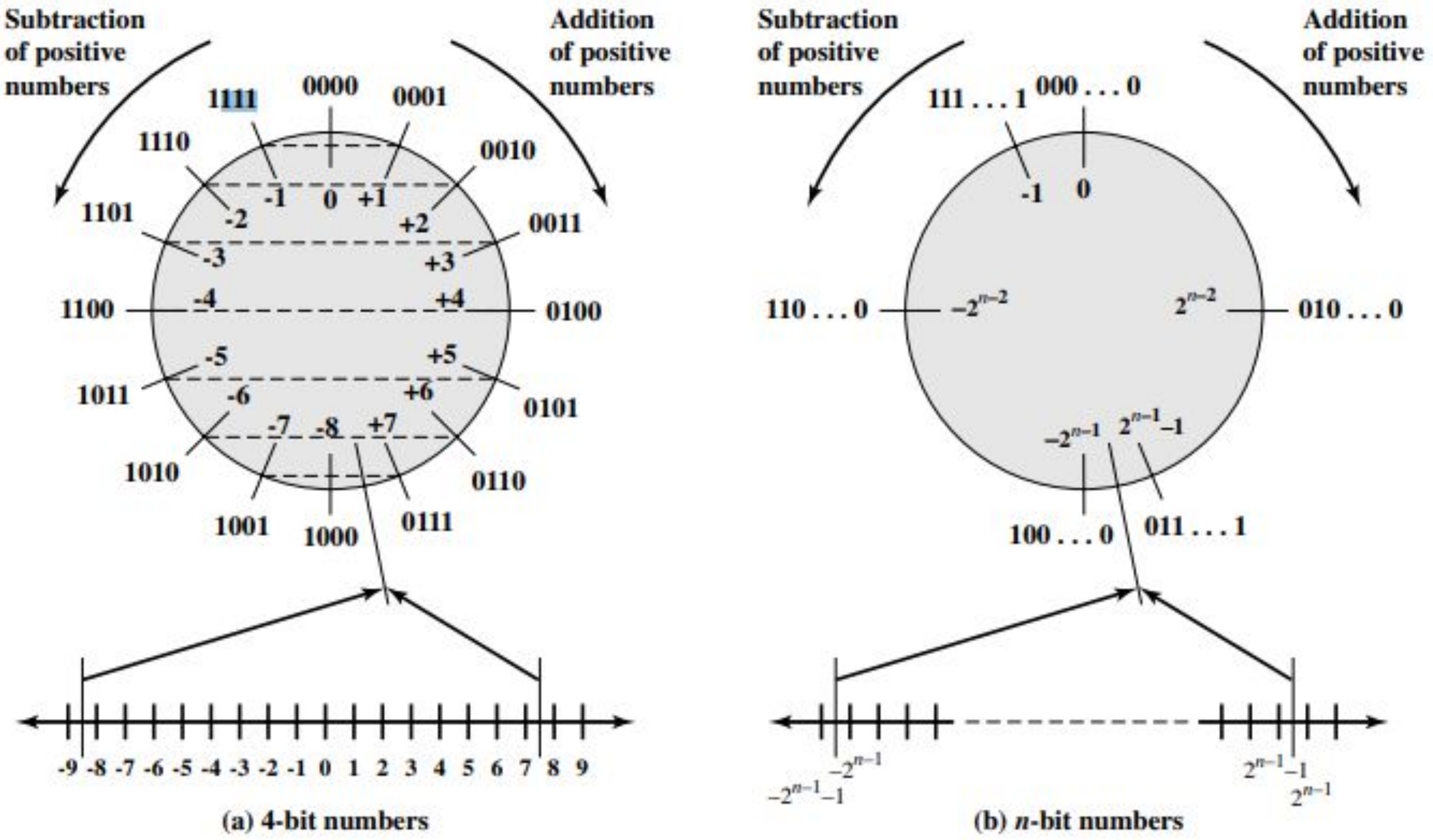
IFT 307
Computer Organization and Architecture

Mr. Ibrahim Lawal



Geometric Depiction of Twos Complement Integers

Fixed points

- _ _ _ _ . _ _ (b^2 b^1 b^0. b^{-1} b^{-2})(1/2,1/3…) in this case you just need to know where the decimal point is.
- Now if you decide that your decimal point falls here _ _._ _ (_ _ _ _ .)all you need to do is just to take your 2's complement number then you divide by 2 places which will be 2 to the power of 4.

Fixed point varies from 2’s complement because you need to know where the decimal lies.

Fixed points are generally on systems that have a lot of flexibility about how many bits they are using to represent numbers, then your computer may just say it is an integer or a 32bit integer or a long or a 4 bit integer number with two bits after the decimal point.

Floating points

- To represent numbers that are very large or very small and you want a dynamic range.
- They are almost scientific notations.

$$\underline{3.\ 158} \quad . \quad \underline{10^{100}}$$

Mantissa exponent

- floats are actually 32bits in size and 64 bits which are called doubles.
- For 32bit in IEEE standard is to use 23 mantissa and 8 bit exponential but there is a limitation of representing a large number.

E.g. if you have a 32 bit represent you will only be able to represent 23 bits into the mantissa unless it will be overwritten.

THE IEEE 754 FLOATING POINT STANDARD

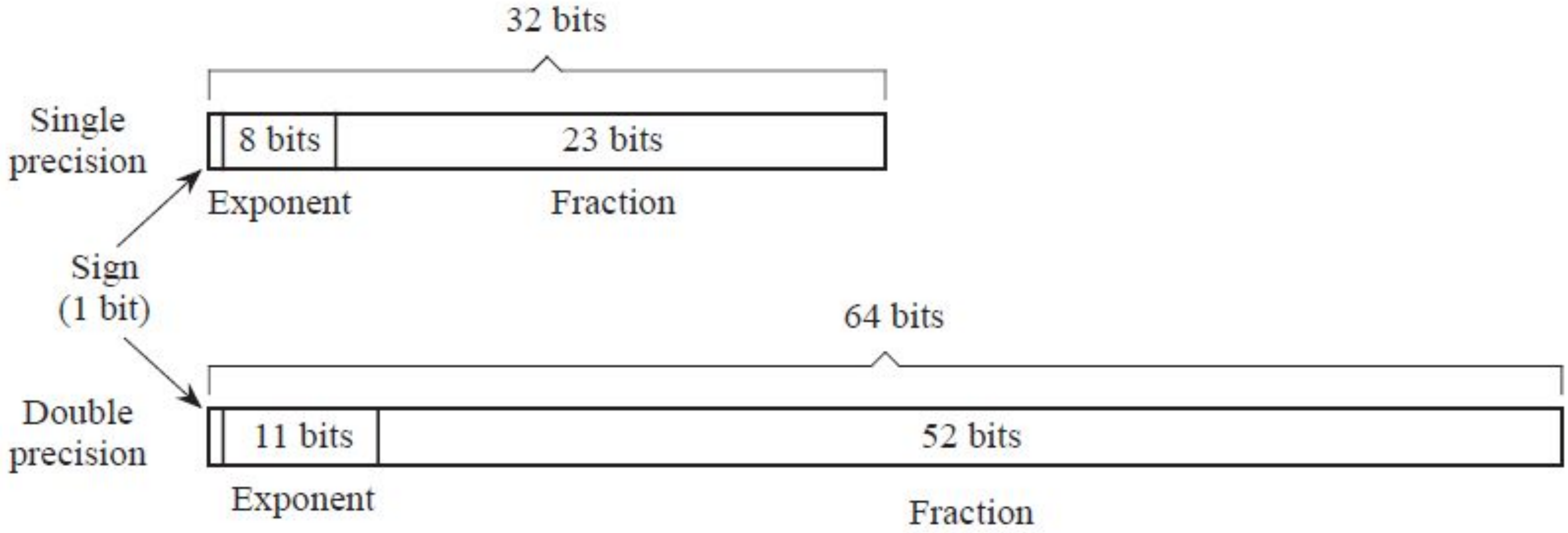
The IEEE 754 standard as described below must be supported by a computer *system*, and not necessarily by the hardware entirely.

That is, a mixture of hardware and software can be used while still conforming to the standard.

Formats

There are two primary formats in the IEEE 754 standard:

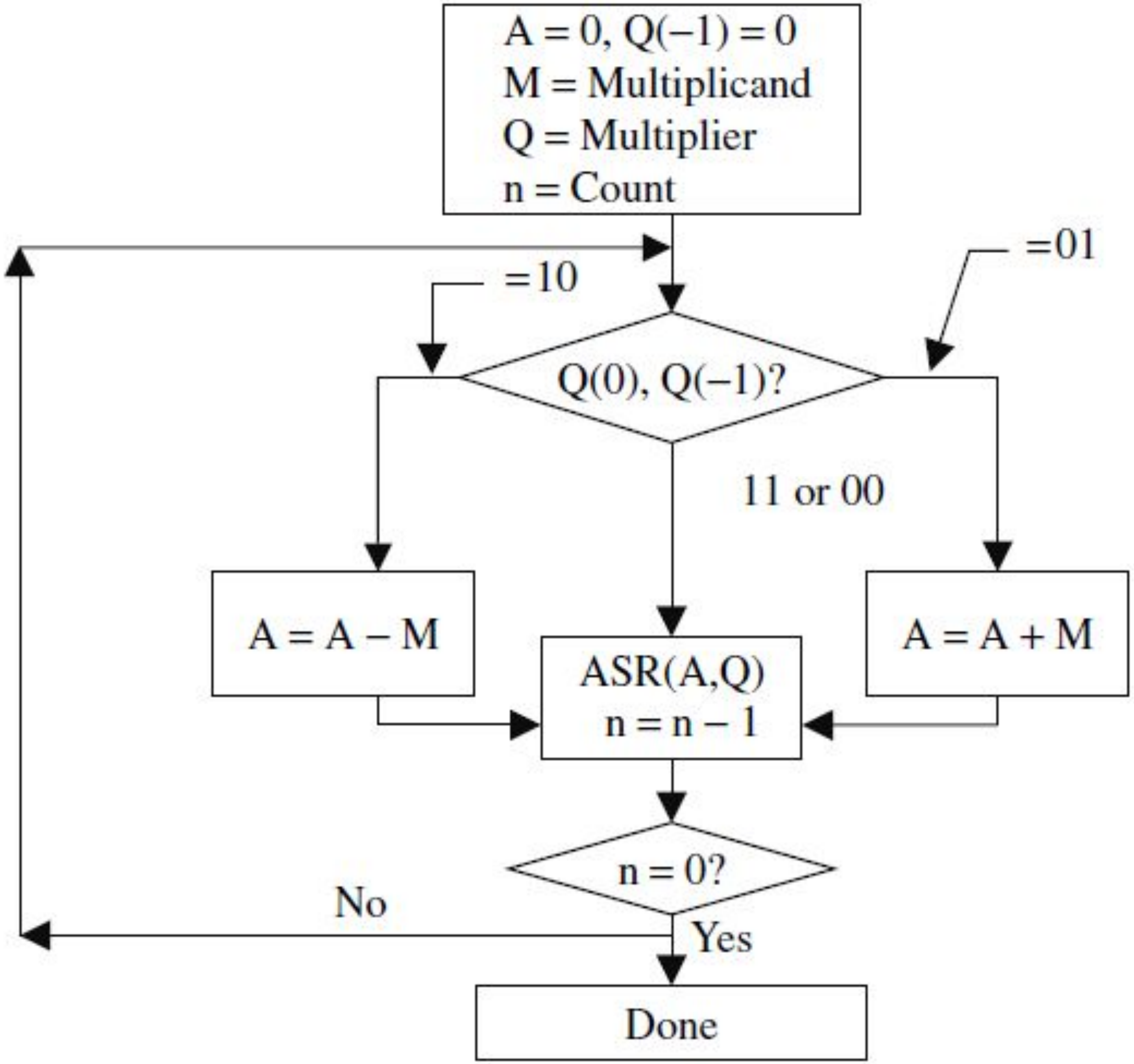
- single precision and
- double precision.


32 bits
Single precision
8 bits
23 bits
Exponent
Fraction
Sign (1 bit)
64 bits
Double precision
11 bits
52 bits
Exponent
Fraction

- Take the following examples

The Booth’s algorithm

The following examples show how to apply the steps of the Booth’s algorithm.



A = 0, Q(−1) = 0
M = Multiplicand
Q = Multiplier
n = Count
=10
=01
Q(0), Q(−1)?
11 or 00
A = A − M
A = A + M
ASR(A,Q)
n = n − 1
n = 0?
No
Yes
Done

Carry in →	0	0	0	0	1	1	1	1
Operands →	0	0	1	1	0	0	1	1
Operands →	+ 0	+ 1	+ 0	+ 1	+ 0	+ 1	+ 0	+ 1
	0 0	0 1	0 1	1 0	0 1	1 0	1 0	1 1
	↑ Carry out ↑ Sum							

Example:

Carry		1	1	1	1	0	0	0	0	
Addend: A		0	1	1	1	1	1	0	0	$(124)_{10}$
Augend: B	+	0	1	0	1	1	0	1	0	$(90)_{10}$
Sum		1	1	0	1	0	1	1	0	$(214)_{10}$

Example Consider the multiplication of the two positive numbers M = 0111 (7) and Q = 0011 (3) and assuming that n = 4. The steps needed are tabulated below.

M	A	Q	$Q(-1)$		
0111	0000	0011	0	Initial value	
0111	1001	0011	0	$A = A - M$	
0111	1100	1001	1	ASR	End cycle #1
0111	1110	0100	1	ASR	End cycle #2
0111	0101	0100	1	$A = A + M$	
0111	0010	1010	0	ASR	End cycle #3
0111	0001	0101	1	ASR	End cycle #4
	$\underbrace{0001\ 0101}_{}$				

+21 (correct result)

Example Consider the multiplication of the two numbers M = 0111 (7) and Q = 1101 (23) and assuming that n = 4. The steps needed are tabulated below.

M	A	Q	$Q(-1)$		
0111	0000	1101	0	Initial value	
0111	1001	1101	0	$A = A - M$	
0111	1100	1110	1	ASR	End cycle #1
0111	0011	1110	1	$A = A + M$	
0111	0001	1111	0	ASR	End cycle #2
0111	1010	1111	0	$A = A - M$	
0111	1101	0111	1	ASR	End cycle #3
0111	1110	1011	1	ASR	End cycle #4

$\underbrace{1110 \quad 1011}_{-21 \text{ (correct result)}}$

- 6 x -2 = -12

- 6 = 0110

- 2 = 0010
- -2 = 1110

$$\begin{array}{r} 1101 \\ +\quad 1 \\ \hline 1110 \\ \hline \end{array}$$

0	**0**	**No Operation**
0	1	Addition
1	0	Subtraction
1	1	No Operation

Decimal	Unsigned	Sign–Mag.	1's Comp.	2's Comp.	Excess 4
7	111	–	–	–	–
6	110	–	–	–	–
5	101	–	–	–	–
4	100	–	–	–	–
3	011	011	011	011	111
2	010	010	010	010	110
1	001	001	001	001	101
+0	000	000	000	000	100
-0	–	100	111	000	100
-1	–	101	110	111	011
-2	–	110	101	110	010
-3	–	111	100	101	001
-4	–	–	–	100	000

3-bit Integer Representations

Thank You