CS2842 Computer Systems – Lecture V

Representing Numerical Data

Dr. Sapumal Ahangama
Department of Computer Science and Engineering

BINARY CODED DECIMAL

- BCD
  - Stored as a digit-by-digit binary representation of the original decimal integer

Decimal	Binary	BCD
68	= 0100 0100 $= 2^6 + 2^2 = 64 + 4 =$ **68**	= 0110 1000 $= 2^2 + 2^1 =$ **6** $2^3 =$ **8**
99 (largest 8-bit BCD)	= 0110 0011 $= 2^6 + 2^5 + 2^1 + 2^0 =$ $= 64 + 32 + 2 + 1 =$ **99**	= 1001 1001 $= 2^3 + 2^0$ $2^3 + 2^0$ = **9** **9**
255 (largest 8-bit binary)	= 1111 1111 $= 2^8 - 1 =$ **255**	= 0010 0101 0101 $= 2^1$ $2^2 + 2^0$ $2^2 + 2^0$ = **2** **5** **5**

VALUE RANGE

- Binary: 4 bits can hold 16 different values (0 to 15)
- BCD: 4 bits can hold only 10 different values (0 to 9)

- BCD range of values <conventional binary representation

No. of Bits	BCD Range		Binary Range	
4	0-9	1 digit	0-15	1+ digit
8	0-99	2 digits	0-255	2+ digits
12	0-999	3 digits	0-4,095	3+ digits
16	0-9,999	4 digits	0-65,535	4+ digits
20	0-99,999	5 digits	0-1 million	6 digits
24	0-999,999	6 digits	0-16 million	7+ digits
32	0-99,999,999	8 digits	0-4 billion	9+ digits
64	0-(10^{16}-1)	16 digits	0-16 quintillion	19+ digits

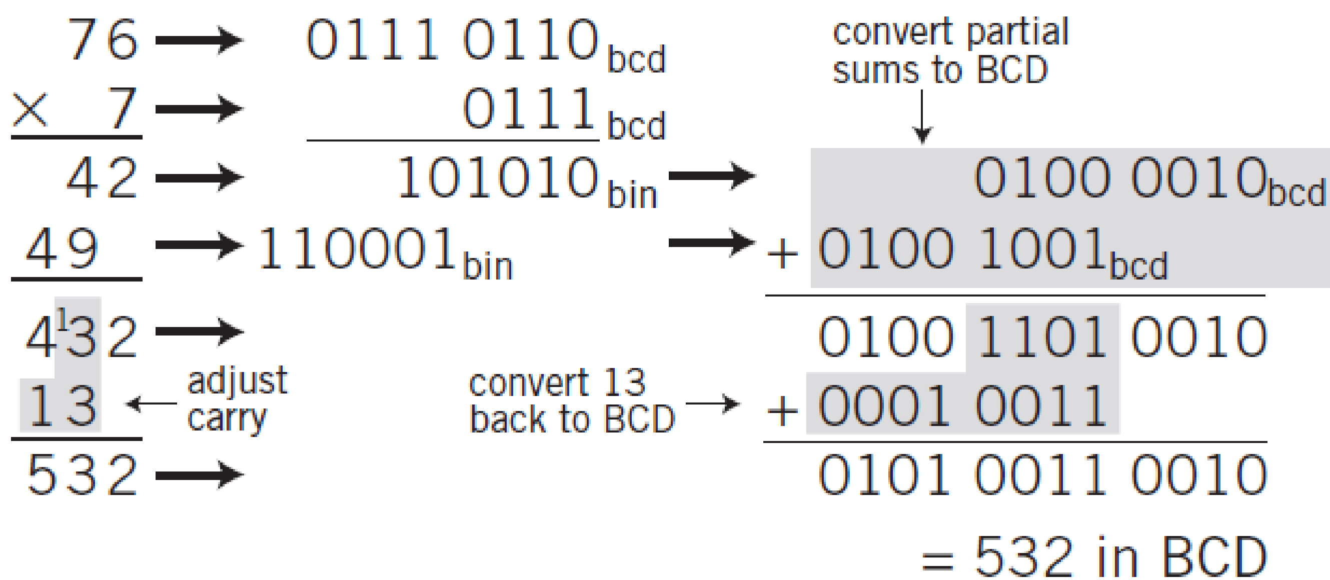
CONVENTIONAL BINARY VS. BCD

- Binary representation generally preferred
  - Greater range of value for given number of bits
  - Calculations are easier

- BCD often used in business applications to maintain decimal rounding and decimal precision
  - 0.2_{10} = .0011001100011… (Conventional Binary)
  - 0.2_{10} = 0.0010 (BCD)

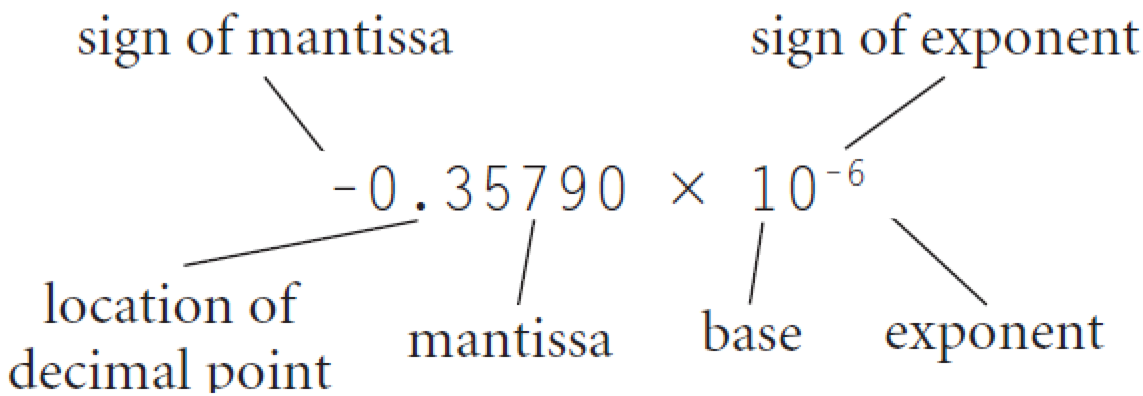
EXAMPLE

A Simple BCD Multiplication



EXPONENTIAL NOTATION

- Also called Scientific Notation



EXPONENTIAL NOTATION

- Example: 12345×10^0
- 4 specifications required for a number
  - Sign ("+" in example)
  - Magnitude or mantissa(12345)
  - Sign of the exponent ("+")
  - Magnitude of the exponent (5)
- Plus
  - Base of the exponent (10)
  - Location of decimal point (or other base) radix point

- 0.12345×10^5
- 123450000×10^{-4}

FORMAT SPECIFICATION

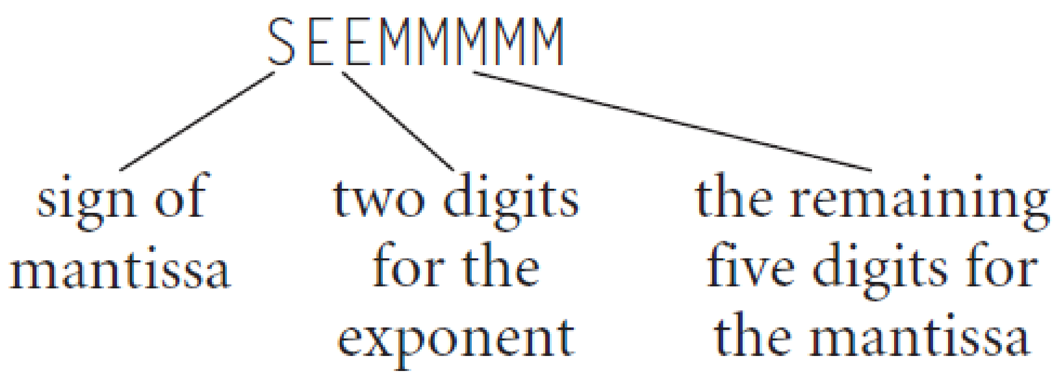
- For integers

SMMMMMMM

- For floating point numbers,
  - Must be divided: to sub parts
  - Part of the space is reserved for the exponent and its sign
  - The remainder is allocated to the mantissa and its sign
  - The base of the exponent and the implied location of the binary point are standardized as part of the format

FORMAT SPECIFICATION

- For floating point numbers,



- Increased range of values (two digits of exponent) traded for decreased precision (two digits of mantissa)

FORMAT

- Mantissa: sign-magnitude format with sign digit
- Assume decimal point located at beginning of mantissa
- Excess-N notation: Complementary notation
  - Pick middle value as offset where N is the middle value
  - Excess-50 notation

Representation	0	49	50	99
Exponent being represented	-50	-1	0	49

– Increasing value +

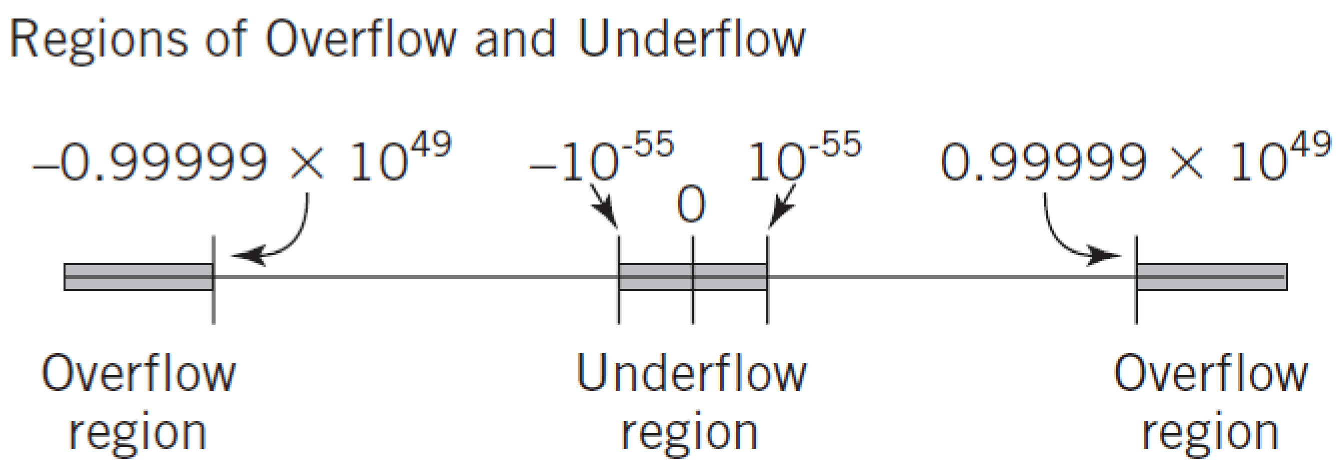
MAGNITUDE RANGE

$$0.00001 \times 10^{-50} < \text{number} < 0.99999 \times 10^{+49}$$

- Larger range than the integers
- Represent fractions

OVERFLOW AND UNDERFLOW

- Underflow,
  - The number is a decimal fraction of magnitude too small to be stored



EXAMPLES

$$05324657 = 0.24657 \times 10^{3} = 246.57$$

$$54810000 = -0.10000 \times 10^{-2} = -0.0010000$$

$$55555555 = -0.55555 \times 10^{5} = -55555$$

$$04925000 = 0.25000 \times 10^{-1} = 0.025000$$

- Convert 246.8035 to floating point representation

$$\overset{\text{the sign}}{0}\underbrace{53}_{\text{excess-50 exponent}}\overbrace{24680}^{\text{the mantissa}}$$

FLOATING POINT CALCULATIONS

- Addition and subtraction
  - Exponent and mantissa treated separately
  - Exponents of numbers must agree
    - Align decimal points
    - Least significant digits may be lost
  - Mantissa overflow requires exponent again shifted right

ADDITION AND SUBTRACTION

05199520
04967850

05199520
0510067850

(1)0019850

05210019(850)

05210020

ADDITION AND SUBTRACTION

$$
\begin{aligned}
05199520 &= 0.99520 \times 10^{1} = 9.9520 \\
04967850 &= 0.67850 \times 10^{-1} = \underline{0.06785}
\end{aligned}
$$

$$10.01985 = 0.1001985 \times 10^{2}$$

MULTIPLICATION AND DIVISION

$$\begin{array}{r} 05220000 \\ \times 04712500 \\ \hline \end{array}$$

$$52 + 47 - 50 = 49$$

$$0.20000 \times 0.12500 = 0.025000000$$

$$04825000$$

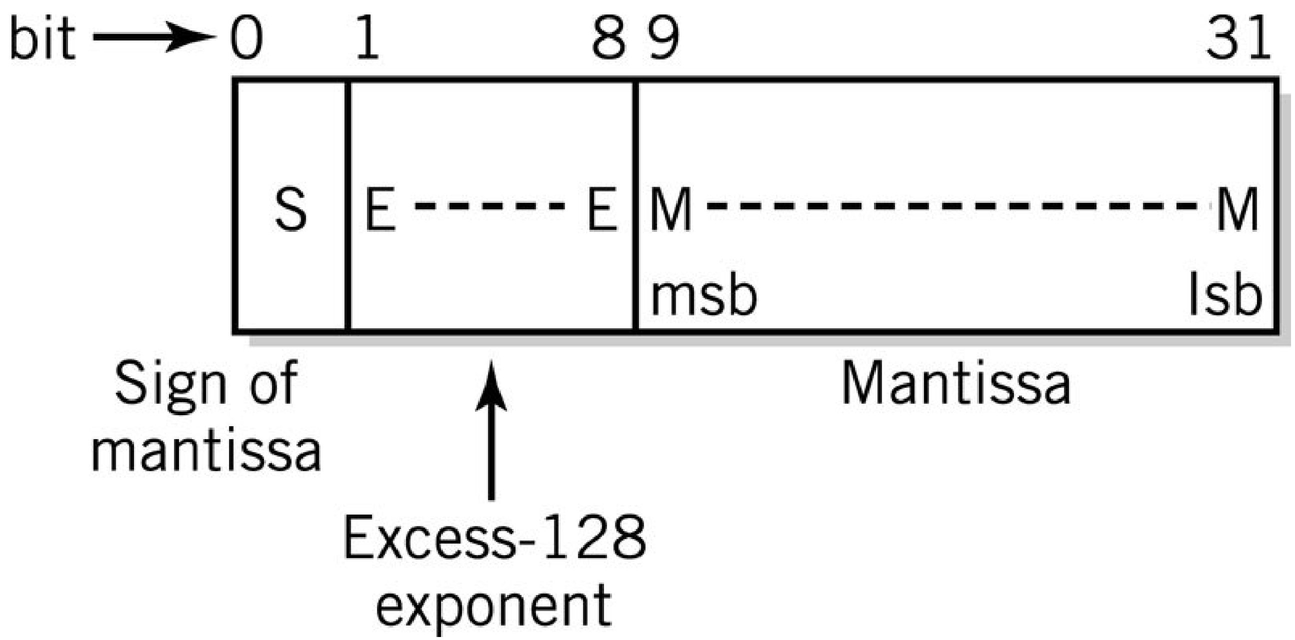
05220000 is equivalent to 0.20000×10^{2},
04712500 is equivalent to 0.12500×10^{-3}

$$0.025000000 \times 10^{-1} = 0.25000 \times 10^{-2}$$

FLOATING POINT IN THE COMPUTER

- 4, 8, or 16 bytes can be used to represent a floating point numbers
- Typical floating point format
  - 32 bits provide range ~10^{-38} to 10^{+38}
  - 8-bit exponent = 256 levels
    - Excess-128 notation
  - •23/24 bits of mantissa: approximately 7 decimal digits of precision

FLOATING POINT IN THE COMPUTER



IEEE 754 STANDARD

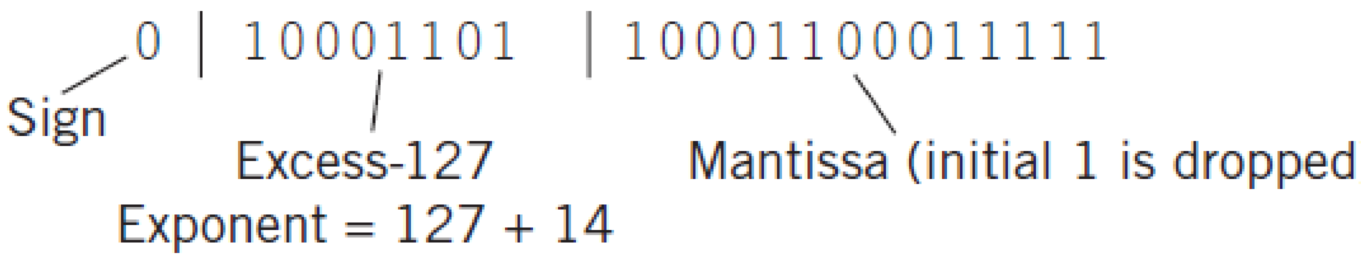
- Defines formats for 32-bit and 64-bit floating point arithmetic
  - Facilitates the portability of programs between different computers that support the standard
- 32 bits,
  - 1 sign bit
  - 8 bits exponent (2^{-126} to 2^{127})
  - 23 bits mantissa
  - Normalized

IEEE 754 STANDARD

Exponent	Mantissa	Value
0	± 0	0
0	not 0	$\pm 2^{-126} \times 0.M$
1-254	any	$\pm 2^{E-127} \times 1.M$
255	± 0	$\pm \infty$
255	not 0	NaN (Not a Number)

CONVERSION: BASE 10 AND BASE 2

- Convert 253.75_{10} to binary floating point form
  - Multiply number by 100 => 25375
  - Convert to binary equivalent 110 0011 0001 1111
  - $1.1000\ 1100\ 0111\ 11 \times 2^{14}$



PROGRAMMING CONSIDERATIONS

- Integer advantages
  - Easier for the computer to perform
  - Potential for higher precision
  - Faster to execute
  - Fewer storage locations to save time and space
- Most high-level languages provide 2 or more formats
  - Short integer (16 bits)
  - Long integer (64 bits)

PROGRAMMING CONSIDERATIONS

- Real numbers
  - Variable or constant has fractional part
  - Numbers take on very large or very small values outside integer range
  - Program should use least precision sufficient for the task
  - Packed decimal attractive alternative for business applications

THANK YOU

REFERNCES

- Chapter 5: REPRESENTING NUMERICAL DATA -The Architecture of Computer Hardware, Systems Software & Networking: An Information Technology Approach -4th Edition, Irv Englander -John Wiley and Sons