Number representation

Bit Pattern	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Unsigned	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Sign & Magnitude	+0	+1	+2	+3	+4	+5	+6	+7	−0	−1	−2	−3	−4	−5	−6	−7
1s Complement	+0	+1	+2	+3	+4	+5	+6	+7	-7	-6	-5	-4	-3	-2	-1	-0
2s Complement	+0	+1	+2	+3	+4	+5	+6	+7	-8	-7	-6	-5	-4	-3	-2	-1
Excess-8	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7
BCD	0	1	2	3	4	5	6	7	8	9	-	-	-	-	-	-

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Number representation Excess-n

- **Excess-n TO Decimal number: convert to decimal, substract the n from the decimal**
- **Decimal number TO Excess-n: add the n to the decimal and convert result to binary**

-3 in Excess-8?
-3 + 8 = 5
5 in unsigned = 0101 = 5 in one-complement = 5 in two-complement = -3 in Excess 8

5 in Excess-8?
5 + 8 = 13
13 in unsigned: 1101 (beyond 2s complement rang ft like a circular linked list in 2s complement!). No further processing necessa

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

-7 in excess-6?
-7 + 6 = -1 = -1 in 1s complement: 0001 -> (negative number bit inversion rule) -> 1110 =
-1 in 2s complement: 1110 + 1 = 1111 = -7 in excess-6

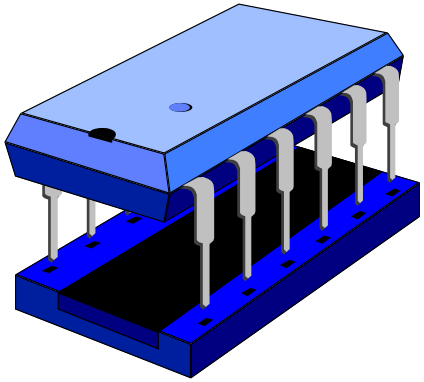
-8 in excess-6?
-8+6 = -2 ->
2 in unsigned: 0010 -> 1s complement = 0010 -> (negative number bit inversion rule) -> 1101
In 2s complement = 1110 = -8 in excess-6

Number representation

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Bit Pattern	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Unsigned	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Sign & Magnitude	+0	+1	+2	+3	+4	+5	+6	+7	−0	−1	−2	−3	−4	−5	−6	−7
1s Complement	+0	+1	+2	+3	+4	+5	+6	+7	-7	-6	-5	-4	-3	-2	-1	-0
2s Complement	+0	+1	+2	+3	+4	+5	+6	+7	-8	-7	-6	-5	-4	-3	-2	-1
Excess-8	-8	-7	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7
BCD	0	1	2	3	4	5	6	7	8	9	-	-	-	-	-	-
Excess-6	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6	7	-8	-7

FLOATING POINT NUMBERS



Assignment Project Exam Help

Introduction

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Bernhard Kainz (with thanks to **N. Dulay** and **E. Edwards**)

b.kainz@imperial.ac.uk

Why do we need this: large, small and fractional numbers

World population >7, 200, 000, 000 people

One light year 9, 130, 000, 000, 000 km

One solar mass 2, 000, 000, 000, 000, 000, 000, 000, 000, 000 kg

Electron diameter 00, 01 m

Electron mass , 000, 000, 000, 000, 9 kg

Smallest measurable length of time 0.000, 000, 000, 000, 000, 00 0, 000, 000, 00, 1 sec

Pi (to 14 decimal places) 3.14159 26535 8979...

Standard rate of VAT 20%

Googol 1 followed by a 100 zeros ☺

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Large integers

Example: How can we represent integers up to 30 decimal digits long?

Assignment Project Exam Help

- **Binary**: $2^X = 10$ 100 bits (1 decimal digit $\approx$

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

- **BCD**: $30 \times 4 = 120$ bits

- **ASCII**: $30 \times 8 = 240$ bits

Floating point numbers

Recall scientific notation:

$M \times 10^E$ Decimal

$M \times 2^E$ Binary

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

This is the basis for ation schemes

- *M* is the **coefficient** (aka. **significan** r **mantissa**)
- *E* is the **exponent** (aka. **characteristic**)
- *10* (or for binary, *2*) is the **radix** (aka. **base**)

- No. of bits in **exponent** determines the **range** (**bigness/smallness**)
- No. of bits in **coefficient** determines the **precision** (**exactness**)

Real vs. floating point numbers

	Mathematical real	Floating point number
Range	$-\infty \ldots + \infty$	Finite
No. of values	(Uncountably) infinite	Finite
Spacing	?	Gap between numbers varies
Errors		Incorrect results are possible

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Some questions **(assume signed 3-digit coefficient** and a **signed 2-digit exponent as before):**

- What are the **closest** floating point numbers to $\mathbf{.001 \times 10^{-99}}$? What is the **gap** between this number and them?
- What about $\mathbf{.001 \times 10^{-50}}$?

Zones of expressibility

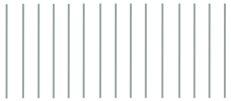
- **Example**: assume numbers are formed with a **signed 3-digit coefficient** and a **signed 2-digit exponent**

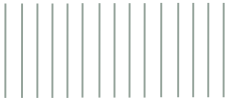
Assignment Project Exam Help

- **Zones of expre**

https://eduassistpro.github.io/

Add WeChat edu_assist_pro





Normalised floating point numbers

- Depending on how you interpret the coefficient, floating point numbers **can have multiple forms**, e.g.:

$$0.023 \times 10^4 = 0.230 \times 10^3$$

Assignment Project Exam Help

$$0.5_2$$

https://eduassistpro.github.io/

- For hardware implementations ble for each number to have a unique floati presentation, a **normalised form**

Add WeChat edu_assist_pro

- We'll normalise coefficients in the **range** $[\mathbf{1}, \ldots \boldsymbol{R})$ where $\boldsymbol{R}$ is the base, e.g.:

$[1, \ldots, 10)$ for decimal

$[1, \ldots, 2)$ for binary

Normalised forms (base 10)

Number	Normalised form
23.2×10^4	

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Normalised forms (base 10)

Number	Normalised form
23.2×10^4	2.32×10^5

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Normalised forms (base 10)

Number	Normalised form
23.2×10^4	2.32×10^5
$-4.01 \times$	

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Normalised forms (base 10)

Number	Normalised form
23.2×10^4	2.32×10^5
$-4.01 \times$	$.01 \times 10^{-3}$

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Normalised forms (base 10)

Number	Normalised form
23.2×10^4	2.32×10^5
$-4.01 \times$	$.01 \times 10^{-3}$
$343\,000 \times 10$	3×10^5
$0.000\,000\,098\,9 \times 10^0$	

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Normalised forms (base 10)

Number	Normalised form
23.2×10^4	2.32×10^5
$-4.01 \times$	$.01 \times 10^{-3}$
$343\,000 \times 10$	3×10^5
$0.000\,000\,098\,9 \times 10^0$	$\times 10^{-8}$

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Normalised forms (base 2)

Number	Normalised form
100.01×2^1	1.0001×2^3
1010.11	1011×2^5
0.00101×2	1×2^{-5}
1100101×2^{-2}	101×2^4

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Binary fractions

Binary	Decimal
0.1	

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Binary fractions

Binary	Decimal
0.1	0.5

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Binary fractions

Binary	Decimal
0.1	0.5
0.01	

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Binary fractions

Binary	Decimal
0.1	0.5
0.01	0.25

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Binary fractions

Binary	Decimal
0.1	0.5
0.01	0.25
0.00	

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Binary fractions

Binary	Decimal
0.1	0.5
0.01	0.25
0.00	0.125

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Binary fractions

Binary	Decimal
0.1	0.5
0.01	0.25
0.00	0.125
0.11	0.75

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Binary fractions

Binary	Decimal
0.1	0.5
0.01	0.25
0.00	0.125
0.11	0.75

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Binary fractions

Binary	Decimal
0.1	0.5
0.01	0.25
0.00	0.125
0.11	0.75
0.111	

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Binary fractions

Binary	Decimal
0.1	0.5
0.01	0.25
0.00	0.125
0.11	0.75
0.111	0.875

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Binary fractions

Binary	Decimal
0.1	0.5
0.01	0.25
0.001	0.125
0.11	0.75
0.111	0.875
0.011	

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Binary fractions

Binary	Decimal
0.1	0.5
0.01	0.25
0.001	0.125
0.11	0.75
0.111	0.875
0.011	0.375

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Binary fractions

Binary	Decimal
0.1	0.5
0.01	0.25
0.001	0.125
0.11	0.75
0.111	0.875
0.011	0.375
0.101	

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Binary fractions

Binary	Decimal
0.1	0.5
0.01	0.25
0.00	0.125
0.11	0.75
0.111	0.875
0.011	0.375
0.101	0.625

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

Binary fraction to decimal fraction

- What is the binary value **0.01101** in decimal?

	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}
.	0	1	1	0	1

Assignment Project Exam Help

- $\frac{1}{4}+\frac{1}{8}+\frac{1}{32}=\frac{13}{32}=$

https://eduassistpro.github.io/

32	16	8	4	2	1
.	0	1	1		1

Add WeChat edu_assist_pro

- $\frac{8+4+1}{2^5}=\frac{13}{32}$
- What about 0.000 110 011?
- Answer: $\frac{32+16+2+1}{2^9}=\frac{51}{512}=0.099\,609\,375$

Decimal fraction to binary fraction

- What is the decimal value **0.6875** in binary?

$$0.6875 = \frac{1.375}{2} = \frac{1}{2} + \frac{0.375}{2} = \frac{1}{2} + \frac{0.75}{4} = \frac{1}{2} + \frac{1.5}{8}$$

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

$$= \frac{1}{2} + \frac{1}{8} + \frac{1}{8}$$

So the answer is **0.1011**

- What is the decimal value **0.1** in binary?

$$0.1 = \frac{1.6}{16} = \frac{1}{16} + \frac{0.6}{16} = \frac{1}{16} + \frac{1.2}{32} = \frac{1}{16} + \frac{1}{32} + \frac{0.2}{32} = \frac{1}{16} + \frac{1}{32} + \frac{1.6}{256}$$

...

Floating point multiplication

$$N_1 \times N_2 = \left(M_1 \times 10^{E_1}\right) \times \left(M_2 \times 10^{E_2}\right)$$
$$= (M_1 \times M_2) \times \left(10^{E_1} \times 10^{E_2}\right)$$
$$= (M_1 \times M_2) \times \left(10^{E_1+E_2}\right)$$

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

- That is, we **mul** nd **add the exponents**
- Example:

$$\left(2.6 \times 10^{6}\right) \times \left(5.4 \times 10^{-3}\right) = (2.6 \times 5.4) \times \left(10^{3}\right)$$
$$= 14.04 \times 10^{3}$$

- We must also **normalise the result**, so final answer is 1.404×10^{4}

Truncation and rounding

- For many computations, the result of a floating point operation is **too large to store in the coefficient**
- Example (with a **2-digit coefficient**):

$$(2.3 \times 10^{2}) \times (2.3 \times 10^{0}) = 5.29 \times 10^{2}$$

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

- **Truncation** ➔ 5.2×10^{2} rror)
- **Rounding** ➔ 5.3×10^{2} (unbiased error)

Floating point addition

- A floating point addition such as $4.5 \times 10^3 + 6.7 \times 10^2$ is not a simple coefficient addition, unless the exponents are the same. Otherwise, we need to align them first

Assignment Project Exam Help

$$N_1 + \left(\quad\right)\left(\quad 10^{E_2}\right)$$

https://eduassistpro.github.io/

$$\left(\quad\right) \times 10^{E_1}$$

Add WeChat edu_assist_pro

- To align, **choose the number wit er exponent** and **shift its coefficient** the corresponding number of digits to the right

$$4.5 \times 10^3 + 6.7 \times 10^2 = 4.5 \times 10^3 + 0.67 \times 10^3$$
$$= 5.17 \times 10^3 = 5.2 \times 10^3$$

(rounded)

Exponent overflow and underflow

- **Exponent overflow** occurs when the result is **too large** i.e. when the **result's exponent > maximum exponent**

- **Example:** if max exponent is 99 then $10^{99} \times 10^{99} = 10^{198}$ **(overflow)**

Assignment Project Exam Help

To handle **overf**https://eduassistpro.github.io/ raise an exception

Add WeChat edu_assist_pro

- **Exponent underflow** occurs when t **e small** i.e. when the **result's exponent < smallest exponent**

- **Example:** if min exponent is -99 then $10^{-99} \times 10^{-99} = 10^{-198}$ **(underflow)**

To handle **underflow**, set value as zero or raise an exception

Comparing floating point values

- Because of the potential for producing **inexact results**, comparing floating point values should **account for close results**
- If we know the **desired magnitude and precision** of results, we can adjust for example:

Assignment Project Exam Help
https://eduassistpro.github.io/
Add WeChat edu_assist_pro

$a = b$ $\quad (b - \epsilon) < a < (b +$

$a = 1$ $\quad 1 - 0.000005 < a < 1 + 0.000005$

$0.999995 < a < 1.000005$

- A more general approach is to calculate closeness of two numbers based on the **relative size** of the two numbers being compared