# 1 Number Systems on the Computer

## 1.1 Binary Numbers

Humans like to use the *decimal number* system for representing numbers. Decimal numbers are *base* 10 meaning that a decimal number consists of a sequence of digits separated by a *decimal point*, where each *digit* can have values $d \in \{0, 1, 2, \ldots, 9\}$ and the weight of each digit is proportional to its place in the sequence of digits with respect to the decimal point, i.e., the number $357.6 = 3 \cdot 10^2 + 5 \cdot 10^1 + 7 \cdot 10^0 + 6 \cdot 10^{-1}$, or in general, for a number consisting of digits $d_i$ with $n+1$ and $m$ digits to the left and right of the decimal point, the value $v$ is calculated as:

· decimal number
· base
· decimal point
· digit

$$v = \sum_{i=-m}^{n} d_i 10^i. \tag{1.1}$$

The basic unit of information in almost all computers is the binary digit, or *bit* for short. A *binary number* consists of a sequence of binary digits separated by a decimal point, where each digit can have values $b \in \{0, 1\}$, and the base is 2. The general equation is,

· bit
· binary number

$$v = \sum_{i=-m}^{n} b_i 2^i, \tag{1.2}$$

and examples are $1011.1_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} = 11.5$. Notice that we use subscript 2 to denote a binary number, while no subscript is used for decimal numbers. The left-most bit is called the *most significant bit*, and the right-most bit is called the *least significant bit*. Due to typical organisation of computer memory, 8 binary digits is called a *byte*, and the term *word* is not universally defined but typically related to the computer architecture, a program is running on, such as 32 or 64 bits.

· most significant bit
· least significant bit
· byte
· word

Other number systems are often used, e.g., *octal numbers*, which are base 8 numbers and have digits $o \in \{0, 1, \ldots, 7\}$. Octals are useful short-hand for binary, since 3 binary digits map to the set of octal digits. Likewise, *hexadecimal numbers* are base 16 with digits $h \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f\}$, such that $a_{16} = 10$, $b_{16} = 11$ and so on. Hexadecimals are convenient, since 4 binary digits map directly to the set of hexadecimal digits. Thus $367 = 101101111_2 = 557_8 = 16f_{16}$. A list of the integers 0–63 in various bases is given in Table 1.1.

· octal number
· hexadecimal number

## 1.2 IEEE 754 Floating Point Standard

The set of real numbers, also called *reals*, includes all fractions and irrational numbers. It is infinite in size both in the sense that there is no largest nor smallest number, and that between any 2 given numbers there are infinitely many numbers. Reals are widely used for calculation, but since any computer only has finite memory, there are infinitely many numbers which cannot be represent on a computer. Hence, any computation performed on a

· reals

| Dec | Bin | Oct | Hex | Dec | Bin | Oct | Hex |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 32 | 100000 | 40 | 20 |
| 1 | 1 | 1 | 1 | 33 | 100001 | 41 | 21 |
| 2 | 10 | 2 | 2 | 34 | 100010 | 42 | 22 |
| 3 | 11 | 3 | 3 | 35 | 100011 | 43 | 23 |
| 4 | 100 | 4 | 4 | 36 | 100100 | 44 | 24 |
| 5 | 101 | 5 | 5 | 37 | 100101 | 45 | 25 |
| 6 | 110 | 6 | 6 | 38 | 100110 | 46 | 26 |
| 7 | 111 | 7 | 7 | 39 | 100111 | 47 | 27 |
| 8 | 1000 | 10 | 8 | 40 | 101000 | 50 | 28 |
| 9 | 1001 | 11 | 9 | 41 | 101001 | 51 | 29 |
| 10 | 1010 | 12 | a | 42 | 101010 | 52 | 2a |
| 11 | 1011 | 13 | b | 43 | 101011 | 53 | 2b |
| 12 | 1100 | 14 | c | 44 | 101100 | 54 | 2c |
| 13 | 1101 | 15 | d | 45 | 101101 | 55 | 2d |
| 14 | 1110 | 16 | e | 46 | 101110 | 56 | 2e |
| 15 | 1111 | 17 | f | 47 | 101111 | 57 | 2f |
| 16 | 10000 | 20 | 10 | 48 | 110000 | 60 | 30 |
| 17 | 10001 | 21 | 11 | 49 | 110001 | 61 | 31 |
| 18 | 10010 | 22 | 12 | 50 | 110010 | 62 | 32 |
| 19 | 10011 | 23 | 13 | 51 | 110011 | 63 | 33 |
| 20 | 10100 | 24 | 14 | 52 | 110100 | 64 | 34 |
| 21 | 10101 | 25 | 15 | 53 | 110101 | 65 | 35 |
| 22 | 10110 | 26 | 16 | 54 | 110110 | 66 | 36 |
| 23 | 10111 | 27 | 17 | 55 | 110111 | 67 | 37 |
| 24 | 11000 | 30 | 18 | 56 | 111000 | 70 | 38 |
| 25 | 11001 | 31 | 19 | 57 | 111001 | 71 | 39 |
| 26 | 11010 | 32 | 1a | 58 | 111010 | 72 | 3a |
| 27 | 11011 | 33 | 1b | 59 | 111011 | 73 | 3b |
| 28 | 11100 | 34 | 1c | 60 | 111100 | 74 | 3c |
| 29 | 11101 | 35 | 1d | 61 | 111101 | 75 | 3d |
| 30 | 11110 | 36 | 1e | 62 | 111110 | 76 | 3e |
| 31 | 11111 | 37 | 1f | 63 | 111111 | 77 | 3f |

Table 1.1: A list of the integers 0–63 in decimal, binary, octal, and hexadecimal.

computer with reals must rely on approximations. *IEEE 754 double precision floating-point format (binary64)*, known as a *double*, is a standard for representing an approximation of reals using 64 bits. These bits are divided into 3 parts: sign, exponent and fraction,

· IEEE 754 double precision floating-point format
· binary64
· double

$$s\, e_1 e_2 \dots e_{11}\, m_1 m_2 \dots m_{52},$$

where $s$, $e_i$, and $m_j$ are binary digits. The bits are converted to a number using the equation by first calculating the exponent $e$ and the mantissa $m$,

$$e = \sum_{i=1}^{11} e_i 2^{11-i}, \tag{1.3}$$

$$m = \sum_{j=1}^{52} m_j 2^{-j}. \tag{1.4}$$

I.e., the exponent is an integer, where $0 \leq e < 2^{11}$, and the mantissa is a rational, where $0 \leq m < 1$. For most combinations of $e$ and $m$, the real number $v$ is calculated as,

$$v = (-1)^s\, (1+m)\, 2^{e-1023} \tag{1.5}$$

with the exceptions that

| | $m=0$ | $m \neq 0$ |
|---|---|---|
| $e=0$ | $v=(-1)^s\,0$ (signed zero) | $v=(-1)^s\,m2^{1-1023}$ (subnormals) |
| $e=2^{11}-1$ | $v=(-1)^s\,\infty$ | $v=(-1)^s\,$NaN (not-a-number) |

· subnormals
· NaN
· not-a-number

where $e = 2^{11}-1 = 11111111111_2 = 2047$. The largest and smallest number that is not infinity is thus

$$e = 2^{11} - 2 = 2046, \tag{1.6}$$

$$m = \sum_{j=1}^{52} 2^{-j} = 1 - 2^{-52} \simeq 1, \tag{1.7}$$

$$v_{\max} = \pm\left(2 - 2^{-52}\right) 2^{1023} \simeq \pm 2^{1024} \simeq \pm 10^{308}. \tag{1.8}$$

The density of numbers varies in such a way that when $e - 1023 = 52$, then

$$v = (-1)^s \left(1 + \sum_{j=1}^{52} m_j 2^{-j}\right) 2^{52} \tag{1.9}$$

$$= \pm \left(2^{52} + \sum_{j=1}^{52} m_j 2^{-j} 2^{52}\right) \tag{1.10}$$

$$= \pm \left(2^{52} + \sum_{j=1}^{52} m_j 2^{52-j}\right) \tag{1.11}$$

$$\overset{k=52-j}{=} \pm \left(2^{52} + \sum_{k=51}^{0} m_{52-k} 2^{k}\right), \tag{1.12}$$

which are all integers in the range $2^{52} \leq |v| < 2^{53}$. When $e - 1023 = 53$, then the same calculation gives

$$v \overset{k=53-j}{=} \pm \left(2^{53} + \sum_{k=52}^{1} m_{53-k} 2^{k}\right), \tag{1.13}$$

which are every second integer in the range $2^{53} \leq |v| < 2^{54}$, and so on for larger values of $e$. When $e - 1023 = 51$, the same calculation gives,

$$v \overset{k=51-j}{=} \pm \left(2^{51} + \sum_{k=50}^{-1} m_{51-k} 2^{k}\right), \tag{1.14}$$

which is a distance between numbers of 1/2 in the range $2^{51} \leq |v| < 2^{52}$, and so on for smaller values of $e$. Thus we may conclude that the distance between numbers in the interval $2^n \leq |v| < 2^{n+1}$ is $2^{n-52}$, for $-1022 = 1 - 1023 \leq n < 2046 - 1023 = 1023$. For subnormals, the distance between numbers is

$$
\begin{aligned}
v &= (-1)^s \left( \sum_{j=1}^{52} m_j 2^{-j} \right) 2^{-1022} && (1.15)\\
&= \pm \left( \sum_{j=1}^{52} m_j 2^{-j} 2^{-1022} \right) && (1.16)\\
&= \pm \left( \sum_{j=1}^{52} m_j 2^{-j-1022} \right) && (1.17)\\
&\overset{k=-j-1022}{=} \pm \left( \sum_{j=-1023}^{-1074} m_{-k-1022} 2^{k} \right), && (1.18)
\end{aligned}
$$

which gives a distance between numbers of $2^{-1074} \simeq 10^{-323}$ in the range $0 < |v| < 2^{-1022} \simeq 10^{-308}$.