1

Systemy liczbowe używane w technice komputerowej

Systemem liczenia nazywa się sposób tworzenia liczb ze znaków cyfrowych oraz zbiór reguł umożliwiających wykonywanie operacji arytmetycznych na liczbach. Podstawą systemów liczenia są systemy liczbowe dzielące się na pozycyjne i addytywne.

W systemach **addytywnych** liczbę tworzy się, sumując poszczególne wartości jej znaków cyfrowych. Do systemów addytywnych zaliczamy systemy: rzymski, hieroglificzny i alfabetyczny.

> **UWAGA**
>
> **Cyfry** systemu rzymskiego to: I (1), V (5), X (10), L (50), C (100), D (500), M (1000). **Liczby** są tworzone przez dodawanie poszczególnych cyfr w ciągu, np. XVI = 10+5+1 = 16. Jeżeli przed większą cyfrą pojawia się mniejsza, to przyjmuje ona wartość ujemną, np. XIV = 10–1+5 = 14.

W niniejszym rozdziale zostaną omówione następujące zagadnienia: pozycyjne systemy liczbowe, arytmetyka liczb binarnych, sposoby zapisu liczb binarnych ze znakiem oraz zapis liczb binarnych stało- i zmiennopozycyjnych.

1.1. Pozycyjne systemy liczbowe

Pozycyjny system liczbowy (ang. *positional numeral system*) to sposób zapisywania liczb za pomocą skończonego zbioru znaków (cyfry arabskie, litery alfabetu), w którym wartość liczbowa cyfry zależy od jej umiejscowienia (pozycji) względem sąsiednich znaków. System pozycyjny charakteryzuje liczba zwana **podstawą systemu pozycyjnego**, która jednocześnie określa liczbę używanych cyfr (znaków). Liczby są zapisywane za pomocą cyfr, które ustawia się na określonych pozycjach. Każda pozycja ma swoją wagę równą podstawie podniesionej do potęgi o wartości numeru pozycji. Wartość liczby uzyskujemy po zsumowaniu poszczególnych iloczynów wag i cyfr pozycji.

Załóżmy, że p oznacza podstawę systemu pozycyjnego. Dowolną liczbę l_p n-cyfrową można wówczas zapisać w następującej postaci (wielomianowy zapis liczby):

$$l_p = \sum_{i=0}^{n-1} a_i * p^i$$

$a_{n-1}\, a_{n-2} \ldots a_2\, a_1\, a_0 = a_{n-1} * p^{n-1} + a_{n-2} * p^{n-2} + \ldots + a_2 * p^2 + a_1 * p^1 + a_0 * p^0,$

gdzie: a_i to cyfry należące do zbioru $\{0, 1, \ldots, p-1\}$, p_i — waga, i — numer pozycji cyfry w ciągu liczbowym, n — liczba cyfr w ciągu, * — iloczyn.

Do najpopularniejszych pozycyjnych systemów liczbowych należą:

- system dziesiętny/decymalny (sposób oznaczenia liczb: $99_{10}/99_D$),
- system dwójkowy/binarny (sposób oznaczenia liczb: $0101_2/0101_B$),
- system szesnastkowy/heksadecymalny (sposób oznaczenia liczb: FF_{16}/FF_H),
- system ósemkowy/oktalny (sposób oznaczenia liczb: $77_8/77_O$).

UWAGA

Liczby w poszczególnych systemach są oznaczane za pomocą indeksu dolnego w postaci podstawy lub pierwszej litery nazwy angielskiej.

1.1.1. System dziesiętny (decymalny)

Ludzie posługują się najczęściej pozycyjnym systemem dziesiętnym (ang. *decimal* — decymalny), w którym podstawę stanowi liczba 10, a do zapisu liczb używa się dziesięciu cyfr arabskich: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Jeśli spróbujemy rozpisać dowolną liczbę dziesiętną z wykorzystaniem podanego przed chwilą wzoru, uzyskamy zapis wielomianowy:

$$p = 10,\ a_i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\},$$

pozycja setek *pozycja dziesiątek* *pozycja jedynek*

$$5\,4\,3_D = 5 * 100 + 4 * 10 + 3 * 1$$

$$5_2\, 4_1\, 3_0 = 5 * 10^2 + 4 * 10^1 + 3 * 10^0$$

cyfra *podstawa* *waga*

Każda cyfra w ciągu została ponumerowana, począwszy od prawej strony. Pozycji jedynek przyporządkowano 0, dziesiątek — 1, a setek — 2. Następnie każda cyfra z ciągu została pomnożona przez wagę, którą stanowi podstawa 10 podniesiona do potęgi równej pozycji.

1.1.2. System dwójkowy (binarny)

Cyfrowe urządzenia elektroniczne wykorzystują dwójkowy (ang. *binary* — binarny) pozycyjny system liczbowy, w którym podstawą jest liczba 2, a liczby zapisuje się za pomocą dwóch cyfr arabskich: 0, 1. Zapis liczby dwójkowej jest dłuższy niż dziesiętnej, jednak stosowanie tylko dwóch cyfr ułatwia budowanie układów półprzewodnikowych, w których w uproszczeniu np. 1 oznacza przepływ prądu, a 0 — brak przepływu. Trudno jest natomiast zbudować układ elektroniczny, który wydajnie i stabilnie reprezentowałby dziesięć stanów odpowiadających cyfrom: 0, 1, 2, ..., 9.

UWAGA

Przykładem praktycznego zastosowania systemu binarnego może być proces wyznaczania adresu sieci lub maski podsieci na podstawie adresu IP w notacji dwójkowej.

Liczba naturalna l_B w systemie dwójkowym ma postać $a_i \dots a_1 a_0$, gdzie a przyjmuje wartość 1 lub 0, np. 1100_B (jeden jeden zero zero, nie tysiąc sto!).

Aby dokonać konwersji liczby dwójkowej na postać dziesiętną, należy użyć zapisu wielomianowego:

$p = 2, a_i \in \{0, 1\}$,

$10101_B = 1_4\ 0_3\ 1_2\ 0_1\ 1_0 = 1*2^4+0*2^3+1*2^2+0*2^1+1*20 =$

$= 1*16+0*8+1*4+0*2+1*1 = 16+4+1 = 21_D$

Kolejne cyfry w liczbie binarnej należy ponumerować, począwszy od pierwszej (0) z prawej strony. Następnie każdą cyfrę mnoży się przez wagę otrzymaną z podstawy podniesionej do potęgi równej pozycji. Po przemnożeniu cyfr przez wagi należy je zsumować. Otrzymana liczba dziesiętna jest odpowiednikiem liczby binarnej. Liczba zapisana w systemie dwójkowym jako 10101_B odpowiada 21_D w systemie dziesiętnym.

Aby dokonać zamiany liczby dziesiętnej na postać binarną, należy wykonać cykliczne dzielenie z resztą. Dzielną jest liczba dziesiętna, a dzielnikiem — podstawa systemu binarnego, czyli 2. Wynik z pierwszego dzielenia ponownie jest dzielony przez 2, i tak aż do uzyskania 0. Liczba binarna powstaje na bazie reszt zapisanych w odwrotnej kolejności:

$25 : 2 = 12$	$r = 1$
$12 : 2 = 6$	$r = 0$
$6 : 2 = 3$	$r = 0$
$3 : 2 = 1$	$r = 1$
$1 : 2 = 0$	$r = 1$

$$25_D = 11001_B$$

Po przekształceniu dziesiętnej liczby 25_D uzyskujemy odpowiednik binarny 11001_B.

W celu szybkiego przekształcania liczb binarnych na postać dziesiętną dobrze jest zapamiętać krotności poszczególnych wag systemu binarnego zamieszczone poniżej.

2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
1024	**512**	**256**	**128**	**64**	**32**	**16**	**8**	**4**	**2**	**1**

Dzięki temu w prosty sposób możemy przekształcić liczbę binarną, sumując odpowiedniki dziesiętne wszędzie tam, gdzie w ciągu dwójkowym występują jedynki:

64	32	16	8	4	2	1
1	**1**	**0**	**0**	**1**	**0**	**1_B**
64	32	–	–	4	–	1

$$64 + 32 + 4 + 1 = 101_D$$

1.1.3. System szesnastkowy (heksadecymalny)

System szesnastkowy (ang. *hexadecimal* — heksadecymalny) najczęściej jest wykorzystywany do uproszczonego zapisu długich liczb binarnych.

> **UWAGA**
>
> *Ethernetowe* karty sieciowe mają 48-bitowy unikatowy adres sprzętowy zapisany w postaci szesnastkowej, np. 00:50:56:C0:00:08.

Podstawę systemu heksadecymalnego stanowi 16 cyfr. Pierwsze 10 to arabskie cyfry: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, pozostałe 6 to pierwsze litery alfabetu łacińskiego: A, B, C, D, E, F, oznaczające kolejno dziesiętne: 10, 11, 12, 13, 14, 15.

UWAGA

Oprogramowanie do wyszukiwania błędów w skompilowanych plikach binarnych przekształca pierwotny zapis danych dwójkowych na krótszy, szesnastkowy, ułatwiając w ten sposób analizę kodu. W systemie binarnym odpowiednik dziesiętnej liczby 15_D ma aż cztery cyfry: 1111_B, natomiast w szesnastkowym — tylko jedną: F_H.

Liczba naturalna l_H w systemie szesnastkowym ma postać: $a_i \ldots a_1 a_0$, gdzie a przyjmuje wartość 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, np. $1BF_H$.

Chcąc dokonać konwersji liczby szesnastkowej na postać dziesiętną, powinniśmy użyć zapisu wielomianowego:

$p = 16, a_i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F\}$,

$4C5_H = 4_2\, C_1\, 5_0 = 4*16^2 + C*16^1 + 5*16^0 = 4*256+12(C)*16+5*1 = 1221_D$

Kolejne cyfry w liczbie heksadecymalnej należy ponumerować, począwszy od pierwszej (0) z prawej strony. Następnie każdą cyfrę mnożymy przez wagę otrzymaną z podstawy (16) podniesionej do potęgi równej pozycji. Po przemnożeniu cyfr przez wagi (litery należy zamienić na odpowiedniki dziesiętne) wykonujemy sumowanie. Otrzymana liczba dziesiętna jest odpowiednikiem liczby szesnastkowej. Liczba zapisana w systemie szesnastkowym jako $4C5_H$ odpowiada 1221_D w systemie dziesiętnym.

Aby dokonać zamiany liczby dziesiętnej na postać szesnastkową, należy wykonać cykliczne dzielenie z resztą. Dzielną jest liczba dziesiętna, natomiast dzielnikiem — podstawa systemu heksadecymalnego, czyli 16. Wynik uzyskany z pierwszego dzielenia ponownie jest dzielony przez 16, i tak aż do uzyskania 0. Liczba szesnastkowa powstaje na bazie reszt zapisanych w odwrotnej kolejności. Wartości powyżej 9 koduje się za pomocą odpowiednich cyfr-liter, np. A:

1221 : 16 = 76	r = 5
76 : 16 = 4	r = 12 (C)
4 : 16 = 0	r = 4

$1221_D = 4C5_H$

UWAGA

W celu szybkiego obliczenia reszty z dzielenia, np. 1221:16 = 76,3125, należy pomnożyć część całkowitą wyniku, czyli 76, przez dzielnik 16. Wynik 1216 należy odjąć od dzielnej: 1221–1216, co da resztę 5.

Przy konwersji liczb szesnastkowych na postać binarną i odwrotnie najprościej posłużyć się poniższą tabelą.

Cyfra szesnastkowa	Cyfra dwójkowa	Cyfra szesnastkowa	Cyfra dwójkowa
0	0000	8	1000
1	0001	9	1001
2	0010	A	1010
3	0011	B	1011
4	0100	C	1100
5	0101	D	1101
6	0110	E	1110
7	0111	F	1111

Konwersję liczby binarnej na postać szesnastkową należy rozpocząć od pogrupowania ciągu po cztery cyfry. Grupowanie rozpoczynamy od prawej strony i kontynuujemy aż do uzyskania końca liczby. Jeżeli ostatnie cyfry w pogrupowanej liczbie mają mniej niż cztery znaki, należy uzupełnić puste pozycje zerami:

$$10111101011101000000101_B = 10 \mid 1111 \mid 0101 \mid 1101 \mid 0000 \mid 0101_B =$$

$$= \mathit{00}10 \mid 1111 \mid 0101 \mid 1101 \mid 0000 \mid 0101_B$$

Następnie, posługując się tabelą, należy wszystkie pogrupowane znaki zamienić na odpowiadające im cyfry heksadecymalne:

*00*10	1111	0101	1101	0000	0101
2	F	5	D	0	5

Po dokonaniu zamiany powstaje liczba szesnastkowa $2F5D05_H$ — prawda, że proste?

Konwersja z liczby szesnastkowej na binarną jest jeszcze prostsza. Wystarczy na podstawie tabeli zamienić cyfry heksadecymalne na czterocyfrowe ciągi binarne i połączyć je w jedną liczbę (np. dla $A4B9F0_H$):

1010	0100	1011	1001	1111	0000

Otrzymujemy liczbę binarną $101001001011100111110000_B$.

1.1.4. System ósemkowy (oktalny)

System ósemkowy (ang. *octal* — oktalny) jest pozycyjnym systemem liczbowym, w którym podstawą jest liczba 8, a liczby zapisuje się za pomocą ośmiu kolejnych cyfr arabskich: 0, 1, 2, 3, 4, 5, 6, 7. System ten jest rzadko stosowany; zastosowanie można zobaczyć w uniksowym poleceniu `chmod` (*służącym do zmiany uprawnień dostępu do plików i katalogów*).

Liczba naturalna l_O w systemie ósemkowym ma postać: $a_i \dots a_1 a_0$, gdzie a przyjmuje wartość 0, 1, 2, 3, 4, 5, 6, 7, np. 212_O.

Konwersję liczb ósemkowych na postać dziesiętną i odwrotnie wykonuje się analogicznie jak w przykładach poświęconych systemom binarnemu i szesnastkowemu.

1.2. Działania na liczbach binarnych

Liczby binarne umożliwiają wykonywanie operacji arytmetycznych (ang. *arithmetic operations on binary numbers*), takich jak suma, różnica, iloczyn i iloraz. Arytmetyką liczb binarnych rządzą pewne zasady, tzw. tabliczki: dodawania, odejmowania, mnożenia i dzielenia.

1.2.1. Dodawanie liczb binarnych

Dodawanie liczb binarnych (ang. *addition of binary numbers*) opiera się na prostej tabliczce dodawania, w której reprezentowane są cztery sumy cząstkowe:

0+0 = 0
0+1 = 1
1+0 = 1
1+1 = 0 i ***1 dalej***

Trzy pierwsze sumy nie wymagają komentarza. Czwarta suma, 1+1, daje wynik 0 w bieżącej kolumnie oraz przeniesienie (ang. *carry*) jedynki do następnej kolumny (w lewo), gdzie jest ona dodawana do stojącej tam liczby.

W celu przybliżenia szczegółów dodawania liczb binarnych rozpatrzmy przykład, w którym dodamy liczby binarne 1101_B i 1011_B.