Лабораторна робота 2. Коди та їх характеристики

Мета: вивчення методів кодування інформації, принципів роботи з числами у цифрових пристроях.

Загальні відомості

Як у теорії інформації, так і на практиці застосування цифрової схемотехніки використовується багато різноманітних кодів.

Двійкове кодування чисел не є єдиним. При роботі з двійковими числами широко використовуються й інші коди, які в різних практичних ситуаціях мають свої переваги перед двійковим. Деякі з них для позитивних чисел в інтервалі 0…15 представлені у табл. 1.1.

Таблиця 1.1

$A10$	A_2 (двійковий)				B_2 (обернений)				D_2 (доповняльний)				Код Грея			
	a_3	a_2	a_1	a_0	b_3	b_2	b_1	b_0	d_3	d_2	d_1	d_0	g_3	g_2	g_1	g_0
0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
1	0	0	0	1	1	1	1	0	1	1	1	1	0	0	0	1
2	0	0	1	0	1	1	0	1	1	1	1	0	0	0	1	1
3	0	0	1	1	1	1	0	0	1	1	0	1	0	0	1	0
4	0	1	0	0	1	0	1	1	1	1	0	0	0	1	1	0
5	0	1	0	1	1	0	1	0	1	0	1	1	0	1	1	1
6	0	1	1	0	1	0	0	1	1	0	1	0	0	1	0	1
7	0	1	1	1	1	0	0	0	1	0	0	1	0	1	0	0
8	1	0	0	0	0	1	1	1	1	0	0	0	1	1	0	0
9	1	0	0	1	0	1	1	0	0	1	1	1	1	1	0	1
10	1	0	1	0	0	1	0	1	0	1	1	0	1	1	1	1
11	1	0	1	1	0	1	0	0	0	1	0	1	1	1	1	0
12	1	1	0	0	0	0	1	1	0	1	0	0	1	0	1	0
13	1	1	0	1	0	0	1	0	0	0	1	1	1	0	1	1
14	1	1	1	0	0	0	0	1	0	0	1	0	1	0	0	1
15	1	1	1	1	0	0	0	0	0	0	0	1	1	0	0	0

Прямий двійковий код A_2 також називають кодом 8-4-2-1 у відповідності з ваговими коефіцієнтами розрядів.

Обернений двійковий код $B_2 = b_3\, b_2\, b_1\, b_0$ отримується шляхом інверсії кожного розряду прямого коду

$$B_2 = b_3\, b_2\, b_1\, b_0 \;\; = \overline{A_2} \; = \overline{a_3}\, \overline{a_2}\; \overline{a_1}\, \overline{a_0}$$

При довільній основі P обернений код n розрядного числа N доповнює його до максимального можливого значення $P^n - 1$, $N_0 = P^n - 1 - N$.

При цьому, цифра кожного розряду оберненого кода N_0 доповнює відповідну цифру прямого коду N до найбільшого значення $P-1$ (наприклад, для десяткового коду це 9). Обернений код використовується як самостійно в логічних структурах цифрових систем, так і при виконанні арифметичних операцій для одержання доповнюючого коду D_2 . Останній застосовується при виконанні арифметичних операцій і знаходиться відповідно до формули:

$$D_2 = B_2 + 1 = \overline{A_2} + 1 = \overline{a_3}\,\overline{a_2}\,\overline{a_1}\,\overline{a_0} + 1,$$

де число 1 додається шляхом двійкової арифметики.

Код Грея, який часто називається *циклічним*, має ту особливість, що при переході з одного числа до сусіднього проходить зміна **0** на **1** або навпаки тільки в одному розряді. Як видно з таблиці, код, представлений двома, трьома або чотирма розрядами, завжди створює циклічну послідовність, тобто адекватну можливість переходу від самого старшого кодового значення числа до самого молодшого. Ця особливість дозволяє використовувати його при кодуванні кутових переміщень у перетворювачах кута повороту у цифровий код. Код Грея знаходить також широке використання у різних перетворювачах аналог - код, де його властивість дає можливість звести похибки неоднозначності при зчитуванні інформації до одиниці молодшого розряду.

Для одержання коду Грея в літературі описані декілька різних прийомів. Один з них дозволяє будувати код Грея безпосередньо з двійкового, використовуючи наступне правило: i-й біт коду Грея встановлюється в нуль, якщо i-й та $(i + 1)$-й біти відповідного двійкового коду однакові; у протилежному випадку біт $i = 1$. У тому випадку, коли $(i + 1)$-й біт виходить за рамки розрядності двійкового коду, його значення приймається рівним нулю.

При записі слів двійкового коду може використовуватись шістнадцяткове числення. При його використанні десяткові числа від 10 до 15 замінюються відповідно латинськими літерами A, B, C, D, E, F. Двійковий і шістнадцятковий коди легко взаємно переводяться.

Наприклад, $A_2 = 111011_2 = 0011\ 1011_2 = 3B_{16}$.

Для позначення цього коду у цифрових пристроях використовують букву h (*hexadecimal*), яку ставлять після його числового значення: $3B_{16} = 3Bh$.

У задачах обчислювальної та мікропроцесорної техніки часто виникає необхідність працювати з десятковими числами. Для цього використовують двійкове представлення кожної десяткової цифри окремо у відповідності з табл. 1.2. Наприклад, числу 937_{10} відповідає код **1001 0011 0111**. Таке представлення називається *двійково-десятковим кодом*. При виконанні арифметичних операцій з використанням двійково-десяткового коду виникають проблеми, пов'язані із переносом з одного розряду до іншого. Для їх вирішення використовуються *самодоповнюючі коди*. До них відносяться *код Айкена*, вагові коефіцієнти розрядів якого **2-4-2-1,** а також *код "з надлишком 3"*, який отримується з двійково-десяткового шляхом додавання числа **3**. Особливості цих кодів можна побачити з табл. 1.2.

Таблиця 1.2

Десяткове число	Двійково-десятковий код				Код Айкена 2-4-2-1				Код «з надлишком 3»			
A_{10}	a_3	a_2	a_1	a_0	h_3	h_2	h_1	h_0	c_3	c_2	c_1	c_0
0	0	0	0	0	0	0	0	0	0	0	1	1
1	0	0	0	1	0	0	0	1	0	1	0	0
2	0	0	1	0	0	0	1	0	0	1	0	1
3	0	0	1	1	0	0	1	1	0	1	1	0
4	0	1	0	0	0	1	0	0	0	1	1	1
5	0	1	0	1	1	0	1	1	1	0	0	0
6	0	1	1	0	1	1	0	0	1	0	0	1
7	0	1	1	1	1	1	0	1	1	0	1	0
8	1	0	0	0	1	1	1	0	1	0	1	1
9	1	0	0	1	1	1	1	1	1	1	0	0

Властивістю самодоповнюючих кодів є те, що їх зворотні комбінації доповнюють прямі до **9**. Ця особливість дозволяє мікропроцесорним пристроям виконувати арифметичні операції з десятковими числами.

У мікропроцесорній техніці двійково-десяткові коди широко використовуються для відображення цифрової інформації за допомогою знакосинтезуючих індикаторів. Тому необхідно вміти виконувати перетворення двійкового коду будь-якої розрядності у двійково-десятковий. Виконується ця операція у такій послідовності.

Оскільки, наприклад, однобайтовим двійковим словом може бути закодована цифра, яка у десятковому коді містить три знаки (до 255), то таке слово повинно бути переміщене з двійкового регістру R відповідними засобами у три чотирьохбітні регістри: регістр одиниць $R_О$, регістр десятків $R_Д$ і регістр сотень $R_С$.

$R_С \leftarrow R_Д \leftarrow R_О \leftarrow R$.

Виходячи з правил двійкової арифметики, при переміщенні кожного розряду з регістра R у регістри, розміщені зліва, необхідно контролювати значення чисел, що з'являються в регістрах $R_С$, $R_Д$, $R_О$.Якщо ці числа дорівнюють або перевищують $5_{10} = 0101_2$, то до них необхідно додати число $3_{10} = 0011_2$.

Контролюючи у такий спосіб вміст кожного регістра, за вісім тактів операції зсуву виконується повний цикл перетворення двійкового числа з регістру R у відповідні регістри $R_С$, $R_Д$, $R_О$. Наприклад, перетворити число 11011001 (217_{10}) у двійково-десятковий код.

№ етапу	Регістр сотень $R_С$				Регістр десятків $R_Д$				Регістр одиниць $R_О$				Задане число, регістр R							
	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0	0	1
1	0	0	0	0	0	0	0	0	0	0	0	1								
2	0	0	0	0	0	0	0	0	0	0	1	1								
3	0	0	0	0	0	0	0	0	0	1	1	0								
											1	1								
	0	0	0	0	0	0	0	0	1	0	0	1								
4	0	0	0	0	0	0	0	1	0	0	1	1								
5	0	0	0	0	0	0	1	0	0	1	1	1								
											1	1								
	0	0	0	0	0	0	1	0	1	0	1	0								
6	0	0	0	0	0	1	0	1	0	1	0	0								
							1	1												
	0	0	0	0	1	0	0	0	0	1	0	0								
7	0	0	0	1	0	0	0	0	1	0	0	0								
											1	1								
	0	0	0	1	0	0	0	0	1	0	1	1								
8	0	0	1	0	0	0	0	1	0	1	1	1								
	2_{10}				**1_{10}**				**7_{10}**											

Двійкові коди використовуються не тільки для кодування цифрової інформації. Широко відоме двійкове представлення букв, знаків, типових команд. У вітчизняній практиці такий код називається КОІ-7 (код для обміну інформацією, семирозрядний). Аналогічний код використовується в сучасній комп'ютерний техніці, який називається ASCII (*American Standart Code for Information Interchange*) і достатньо детально описаний у ряді літературних джерел.

1.4. Форми зображення чисел

У цифрових пристроях використовуються дві форми зображення чисел: *з фіксованою* і *плаваючою комою*.

Знак двійкового числа з фіксованою комою задається допоміжним розрядом, який встановлюється перед числовими. В додатних чисел значення допоміжного розряду рівне – **0**, для від'ємних – **1**. У табл. 1.3 приводяться три варіанти кодування додатних і від'ємних чисел чотирьохрозрядним двійковим кодом.

Таблиця 1.3

a_3	a_2	a_1	a_0	Прямий і доповняльний	Знакі величина	Прямий і обернений
0	0	0	0	+0	+0	+0
0	0	0	1	1	1	1
0	0	1	0	2	2	2
0	0	1	1	3	3	3
0	1	0	0	4	4	4
0	1	0	1	5	5	5
0	1	1	0	6	6	6
0	1	1	1	7	7	7
1	0	0	0	- 8	- 0	- 7
1	0	0	1	- 7	- 1	- 6
1	0	1	0	- 6	- 2	- 5
1	0	1	1	- 5	- 3	- 4
1	1	0	0	- 4	- 4	- 3
1	1	0	1	- 3	- 5	- 2
1	1	1	0	- 2	- 6	- 1
1	1	1	1	- 1	- 7	- 0

У першому варіанті, як витікає з таблиці, у кодовій двійковій послідовності мають місце додатній і від'ємний нулі, що призводить до появи проблем при виконанні арифметичних операцій.

Представлення від'ємних чисел у оберненому коді також не вирішує відміченої проблеми. Вона вирішується лише тоді, коли від'ємні числа представляються у *доповняльному коді*.

На рис. 1.12 приведена графічна інтерпретація зображення позитивних і негативних чисел відносно нуля з використанням прямого та доповняльного кодів. Як буде показано пізніше, така форма представлення десяткових чисел суттєво спрощує виконання арифметичних операцій.

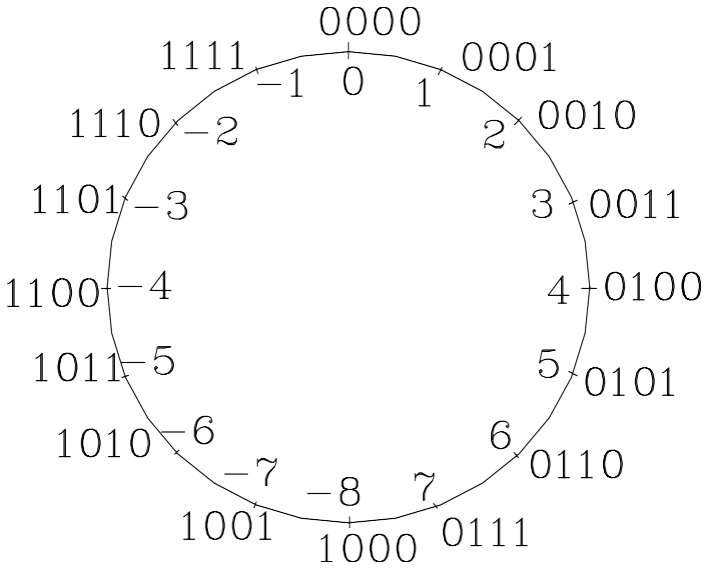

Рис.1. 12

Будь-яке число в цифрових системах зберігається спеціальними пристроями пам’яті, кожен рядок якого складаються з фіксованої кількості елементів. Кома, що відділяє в числі цілу частину від дробової, займає в рядку пам’яті фіксоване положення – перед старшим розрядом або після молодшого.

1	1	1	0	1	0	1	0	0	0

Рис. 1.13

У першому випадку абсолютне значення числа менше одиниці – наприклад, $0{,}110101_2$. Якщо рядок пам’яті призначений для десяти розрядів, то число в ньому запишеться так, як показано на рис. 1.13, де крайній лівий розряд відображає знак числа, а решта – розряди модуля. Вільні молодші розряди заповнюються нулями. Оскільки в розгляданому випадку в рядку пам’яті передбачається запис лише дробової частини числа, то і результати всіх операцій повинні бути з абсолютним значенням, меншим одиниці. Виконання цієї умови забезпечується вибором відповідних масштабних коефіцієнтів, на які помножуються вихідні дані. Якщо масштабний коефіцієнт вибраний невірно, то може з’явитись переповнення розрядів і

поява цілої частини, яка буде втрачена, оскільки в розрядній сітці не передбачена її поява. Все це приведе до похибки в результаті, що є недоліком такого способу.

0	0	0	0	0	1	0	0	1	1

Рис. 1.14

У другому випадку, коли кома фіксується після молодшого розряду, маємо справу з цілими числами. Тоді, наприклад, число 10011_2 в рядку пам'яті розміщується в відповідності з рис. 1.14, де лівий розряд знаковий, а слідуючі за ним справа вільні розряди заповнюються нулями. В цьому випадку величина модуля є обмеженою довжиною рядку пам'яті.

Числа з плаваючою комою передбачають зображення числа з використанням мантиси, що помножається на основу системи числення у ступені, який задається порядком. Наприклад, число 200 записується у вигляді $0{,}2 \cdot 10^3$, а число 0,000312 – як $0{,}312 \cdot 10^{-3}$. Відповідно записуються і двійкові числа. Мантиса і порядок зображаються у двійковому коді, а основою є двійка. Наприклад, число $0{,}111 \cdot 2^2 = 11.10_2$ в десятковій системі зображається як $0{,}875 \cdot 2^2 = 3{,}5_{10}$. В рядку пам'яті такі числа зберігаються у вигляді двох груп цифр: перша група – мантиса – визначає саме число, друга – порядок – місце коми в числі (рис.1.15).

0	1	1	1	0	0	0	0	0	0	1	0	0
0	1	2	3	4	5	6	7	8	9	10	11	12

Рис. 1.15

У нульовому елементі рядка пам'яті зображається знак числа (для приведеного вище двійкового числа, що записане у рядок пам'яті – **0**). Далі задаються вісім розрядів самого числа (стовпці 1…8). Якщо воно задається меншою кількістю розрядів, то вільні елементи пам'яті справа від числа заповнюються нулями. У дев'ятому розряді зображається знак порядку, а в решті, за аналогією з мантисою, – число, що визначає порядок. При використанні такої форми запису величина числа порядку задається так, щоб перша значуща цифра мантиси не дорівнювала –**0**. Така форма запису називається *нормальною*.

Мінімальне додатне число, що може бути записане при нормальній формі в рядку пам'яті, визначається мінімальною мантисою $0{,}1000..0_2$ та максимальним

від'ємним порядком $111..1_2$. При кількості k розрядів порядку мінімальне десяткове число, що може бути записаним, визначається формулою:

$$N_{\min} = \frac{1}{2} \cdot 2^{-\left(2^k - 1\right)} = 2^{-2^k}. \qquad (1.15)$$

Максимальне число матимемо при максимальному значенні мантиси $(0,111\ldots1)_2$ та максимальному додатному порядку $(111\ldots1_2) = 2^k - 1$, тобто

$$N_{\max} = 2^{2^k - 1}. \qquad (1.16)$$

Діапазон D чисел, представлених в нормальній формі, як витікає з формул (1.15) та (1.16), визначається лише числом k. Наприклад, для $k = 6$ знаходимо:

$$2^{-2^6} \le D \le 2^{2^6 - 1}; \qquad 2^{-64} \le D \le 2^{63}.$$

У сучасних цифрових системах для зображення чисел з плаваючою комою використовується рядок довжиною чотири байти. При цьому 23 розряди задають мантису, а 7 – величину порядку. Діапазон чисел, що зображуються, складає від $\pm 2^{127}$ до $\pm 2^{-127}$.

Використання чисел з плаваючою комою суттєво розширює і спрощує зображення чисел, але виконання операцій над такими числами більш складне, ніж над числами з фіксованою комою.

1.5. Виконання арифметичних операцій

Основною операцією, яка використовується в цифрових системах при виконанні різних обчислень, є операція *алгебраїчного додавання*. Вона виконується на основі правил виконання операцій у двійковій системі зображення чисел, які для однорозрядних чисел мають такий вигляд:

$$\begin{array}{r} 0 \\ +\ 0 \\ \hline 0 \end{array} \qquad \begin{array}{r} 1 \\ +\ 0 \\ \hline 1 \end{array} \qquad \begin{array}{r} 0 \\ +\ 1 \\ \hline 1 \end{array} \qquad \begin{array}{r} 1 \\ +\ 1 \\ \hline 10 \end{array}$$

Перенесення до старшого розряду виконується тоді, коли в одному розряді обох складових є одиниці. Операція знаходження суми в багаторозрядних числах виконується послідовно, починаючи з молодшого розряду. У зв'язку з цим, починаючи з другого розряду, виконується складання трьох цифр – двох розрядних складових і перенесення з молодшого розряду.

Приклад 1.13. Скласти два додатні двійкові числа $A_2 = 1001_2$; $B_2 = 1101_2$.

Розв'язання. При виконанні операції додавання мають місце переповнення у першому і четвертому розрядах і, відповідно, перенесення одиниці з першого розряду у другий і з четвертого у п'ятий.

$$\begin{array}{cccccccc} & & \overset{1}{\downarrow} & & \overset{1}{\downarrow} & & & \\ & A_2 & = & 1 & 0 & 0 & 1 & = 9_{10} \\ + & B_2 & = & 1 & 1 & 0 & 1 & = 13_{10} \\ \hline & (A+B)_2 & = 1 & 0 & 1 & 1 & 0 & = 22_{10} \end{array}$$

Операція віднімання в цифрових схемах виконується за допомогою операції додавання, зображуючи від'ємник у доповняльному коді.

Приклад 1.14. Знайти суму двох чисел $N = 854_{10}$ і $K = -387_{10}$ з використанням доповнюючого коду.

Розв'язання. При виконанні вказаної операції в десятковій системі числення необхідно для числа *K* знайти відповідний доповняльний код. Він знаходиться за тими ж правилами, що і в двійковій системі. Обернений код числа знаходиться як доповнення до дев'ятки цифри кожного розряду. Для числа $K = 387_{10}$ обернений код $B = 612_{10}$.

Доповняльний код для числа *K*

$$D = B + 1 = 612 + 1 = 613.$$

Виконаємо операцію додавання. При цьому введемо знакові розряди, які позначимо апострофом, що встановлюється після знакової цифри:

$$N + D = 0'\ 854 + 1'\ 613 = 10'\ 467 = 467.$$

Перенесення, що з'являється зі знакового розряду, відкидається.

Аналогічно виконується операція віднімання в двійковій системі числення.

Приклад 1.15. Додати два числа $N = 0'\ 11011_2 = 27_{10}$ і $K = 1'\ 01101_2 = -13_{10}$.

Розв'язання. Знаходимо доповняльний код від'ємного числа K :

$$D = 1'\ 10011.$$

Знаходимо суму:

$$\begin{array}{rrcccccc} & N & = & 0' & 1 & 1 & 0 & 1 & 1 \\ + & D & = & 1' & 1 & 0 & 0 & 1 & 1 \\ \hline & (N+D) & = & 10' & 0 & 1 & 1 & 1 & 0 \end{array}$$

Відкидаючи 1 переносу в знаковому розряді, отримуємо:

$$N + D = 0'\ 01110_2 = 14_{10}.$$

Приклад 1.16. Змінимо знаки обох чисел на зворотні:

$$N = -27_{10} = 1'\ 11011_2; \qquad K = 13_{10} = 0'\ 01101_2.$$

Розв'язання. Доповняльний код від'ємного числа N : $D = 1'\ 00101_2$.

Знаходимо суму:

$$\begin{array}{rrcccccc} & K & = & 0' & 0 & 1 & 1 & 0 & 1 \\ + & D & = & 1' & 0 & 0 & 1 & 0 & 1 \\ \hline & (K+D) & = & 1' & 1 & 0 & 0 & 1 & 0 \end{array}$$

Знаковий розряд показує, що результат операції від'ємний, а число зображене у доповняльному коді. Для отримання результату в прямому коді необхідно спочатку перейти до оберненого коду, віднімаючи одиницю від результату виконання арифметичної операції, а потім – інвертувати.

Знаходимо обернений код результату: $1'\ 10001_2$; прямий код: $1'\ 01110_2 = -14_{10}$.

Пряма операція віднімання з використанням операцій зайому зі старших розрядів застосовується лише при порівнянні двох кодів, адже відсутність чи наявність зайому зі старшого розряду дає можливість легко визначити більше з порівнюваних чисел.

Розглянемо тепер особливість виконання операцій у двійково-десятковій системі числення (код **8-4-2-1**). Процедура виконання операції здійснюється на основі правил двійкової арифметики. Якщо одержане число перевищує 10_{10} , то повинна формуватися одиниця переносу, яка передається до наступного десяткового розряду. Але результат відрізнятиметься від правильного, і в нього необхідно внести відповідну корекцію.

Приклад 1.17. Скласти числа $A = 5_{10}$ і $B = 3_{10}$ з використанням коду **8-4-2-1**.

Розв'язання. $A + B = 0101_2 + 0011_2 = 1000_2 = 8_{10}$.

Оскільки результат менше десяти, то корекція не потрібна.

Приклад 1.18. Скласти числа $A = 8_{10} = 1000_2$ і $B = 9_{10} = 1001_2$.

Розв'язання.

$$(A + B)_2 = 1000_2 + 1001_2 = 10001 = 17_{10} .$$

У даному випадку результат більше десяти, тому необхідно вводити корекцію.

Поява одиниці в п'ятому розряді означає, що число, представлене чотирма молодшими розрядами, збільшилось на 16 одиниць. Але, з іншого боку, одиниця передалась у старший десятковий розряд, що еквівалентно числу 10, тому в молодшому розряді недостає шістьох одиниць.

Це означає, що необхідно до результату додати корекцію – шість одиниць, тобто:

$$\begin{array}{rrcccccc} & 17_{10} & = & 1 & 0 & 0 & 0 & 1 \;{}_2 \\ + & 6_{10} & = & & & 1 & 1 & 0 \;{}_2 \\ \hline & \text{Сума} & = & 1 & 0 & 1 & 1 & 1 \;{}_2 \end{array} = 0001\ 0111_{2/10} = 17_{10} .$$

Результат представляється одиницею в розряді десятків і сімкою в розряді одиниць.

Приклад 1.19. Скласти числа $6_{10} + 7_{10}$.

Розв'язання.

$$\begin{array}{rrcccccc} & 6_{10} & = & 0 & 1 & 1 & 0 \;{}_2 \\ + & 7_{10} & = & 0 & 1 & 1 & 1 \;{}_2 \\ \hline & \text{Сума} & = & 1 & 1 & 0 & 1 \;{}_2 \end{array} = 13_{10}$$

У цьому прикладі переносу в п'ятий розряд немає, але результат перевершує дев'ятку, і тому необхідно ввести корекцію. Корекція необхідна не тільки для того, щоб скоригувати результат молодших розрядів, а й для того, щоб перенести одиницю в старший десятковий розряд. Знову додасться 10 і відніметься 16, і тому для корекції необхідно додати цифру 6:

$$\begin{array}{rrcccccc} & 13_{10} & = & & 1 & 1 & 0 & 1 \;{}_2 \\ + & 6_{10} & = & & 0 & 1 & 1 & 0 \;{}_2 \end{array}$$

$$\text{Сума} \quad = \quad 1 \quad 0 \quad 0 \quad 1 \quad 1 \quad _2 \quad = 0001\ 0011_{2/10} = 13_{10}\ .$$

З двох останніх прикладів бачимо, що в першому випадку при виконанні операції має місце перенесення в старшу тетраду, що і може виступати ознакою необхідності введення корекції результату. В другому випадку такого перенесення немає, і тому можливість такого результату або необхідно передбачати, або створювати допоміжні заходи для введення корекції. Як вихід з такої ситуації, можна запропонувати використання самокоригуючих кодів – наприклад, коду з надлишком 3.

Приклад 1.20. Виконати операцію знаходження суми двох чисел з попереднього прикладу при використанні коду з надлишком 3.

Розв'язання. Відповідні коди чисел: 6_{10} = 1001; 7_{10} = 1010. Після виконання операції додавання маємо результат:

$$1001 + 1010 = 10011_{2/10} = 0001\ 0011_{2/10} = 13_{10}\ .$$

При використанні інших кодів для зображення десяткових цифр правила знаходження суми зміняться, але логічний аналіз процедури легко дозволяє знайти правила корекції.

При знаходженні суми багаторозрядних двійково-десяткових чисел від'ємні числа зображуються у зворотному або доповняльному коді. При цьому зворотний код одержується доповненням до 9. Якщо використовувати не код **8-4-2-1**, а коди з надлишком **3** або **2-4-2-1**, то процедура формування доповнення до 9 значно спрощується.

Виконання операцій множення або ділення двійкових чисел на 2^j = 2, 4, 8, … досягається шляхом зсуву цифр числа відповідно вправо або вліво на j = 1, 2, 3, … розрядів.

Для довільних чисел виконання операції множення зводиться до послідовного виконання операцій додавання та зсуву.

Приклад 1.21. Виконати множення двох чисел: $A = 101_2$; $B = 011_2$.

Розв'язання.

$$\begin{array}{cccccccc}
 & A & = & & 1 & 0 & 1 & = & 5_{10} \\
\times & B & = & & 0 & 1 & 1 & = & 3_{10} \\
\hline
 & D_1 & = & & 1 & 0 & 1 & & \\
+ & D_2 & = & 1 & 0 & 1 & & & \\
\hline
 & \text{Сума} & = & 1 & 1 & 1 & 1 & = & 15_{10}
\end{array}$$

Спочатку виконується знаходження часткового добутку D_1 , потім здійснюється операція зсуву на один розряд вліво і знаходиться частковий добуток D_2. При наявності більшої кількості часткових добутків знаходяться часткові суми, після чого виконуються операції знаходження часткового добутку та їх зсуву.

Розглянемо ще два приклади. У першому з них операція знаходження суми виконується з переносом, починаючи з другого розряду. У другому демонструється той факт, що множення на нуль не виконується, а приписуються всі нулі справа після виконання операції.

```
          1 0 1 1
    ×       1 1 1
    ---------------
        1 0 1 1
      1 0 1 1
    1 0 1 1
    ---------------
  1 0 0 1 1 0 1
```

```
          1 0 1 0 0
    ×         1 0 1 0
    ---------------
        1 0 1
    1 0 1
    ---------------
  1 1 0 0 1 0 0 0
```

У будь-якому випадку, операція множення складається з операцій зсуву і додавання. Тому в цифровій схемотехніці і мікропроцесорних пристроях вона може виконуватись як на програмному рівні, так і на апаратному.

При виконанні операцій множення і ділення існують спеціальні алгоритми для безпосереднього виконання цих операцій з використанням доповняльних кодів, які описані у відповідній літературі з мікропроцесорної та комп'ютерної техніки. Методи ділення для двійкових чисел у деякій мірі комплементарні до методу двійкового множення. Типовий алгоритм ділення приймає (n + m) біт діленого та n біт дільника і створює m біт результату та n біт залишку. Операція ділення є переповнюючою, якщо дільник дорівнює нулю або результат може мати вираз більший, ніж m біт. У більшості цифрових пристроїв апаратно організовано, що n = m.

Виконання операцій ділення чисел зі знаком може бути виконано тим же шляхом. Результату приписується додатній знак, якщо знаки чисел, що беруть участь у виконанні операції, однакові, і від'ємний, якщо знаки чисел різні. Залишкові приписується той же знак, що і діленому.

2.1.5 Двійково-десяткові числа (коди)

Існує протиріччя між машинним представленням чисел (двійкова система числення) і представленням чисел у нашому повсякденному житті (десяткові числа). Перетворення між ними, у випадку великого об'єму вхідних даних і вихідних результатів, веде до помітних втрат процесорного часу. Разом с тим є велике коло задач, що характеризуються значними об'ємами числових даних і порівняно простою їх обробкою (економічні задачі, статистичні розрахунки, бухгалтерські викладення і т.п.). Тому потрібно буде розробити такі форми представлення чисел, в яких якимсь чином сполучалися б двійкова і десяткова системи числення. Такі форми отримали загальну назву *двійково-кодованого десяткового* (Binary – Coded Decimal) представлення або BCD – кодування. Загальна властивість цих форм виявляється в тому, що за основу береться десяткове число і кожна його цифра зображується тим або іншим двійковим еквівалентом. До теперішнього часу з багаточисельних систем двійково-десяткового кодування практичне застосування знаходять тільки дві: двійково-десяткові числа в упакованому і не упакованому форматі.

2.1.5.1 Двійково-десяткові числа в упакованому форматі

В упакованому форматі, байт містить дві десяткові цифри. Менша цифра займає праву тетраду (біти 3…0), старша – ліву тетраду (біти 7…4). Обидві цифри представлені своїми двійковими еквівалентами, названі також кодом 8421 (по двійковим вагам). Розміщення десяткових цифр у байті показано на **рисунку 2.1, а**.

Багаторозрядні упаковані десяткові числа займають декілька суміжних байтів. При необхідності роботи з знаковими числами старша тетрада (іноді – менша) старшого байта призначається для знака числа (**рисунок 2.1, б**). Для кодування знака можна використовувати шість заборонених тетрад 1010…1111 (або A…F), які не являють собою десяткових цифр. За звичай, для зображення знака “плюс” застосовується код 1100 (C), а знака “мінус” – код 1101 (D). Через необхідність зображення знака багатобайтові упаковані десяткові числа мають непарне число розрядів.

Під час програмування упаковані десяткові числа за звичай визначаються початковою адресою – DNUM (це адреса меншого байта) і числом байтів – N. Розміщення упакованого десяткового числа в пам'яті показано на **рисунку 2.1, в**.

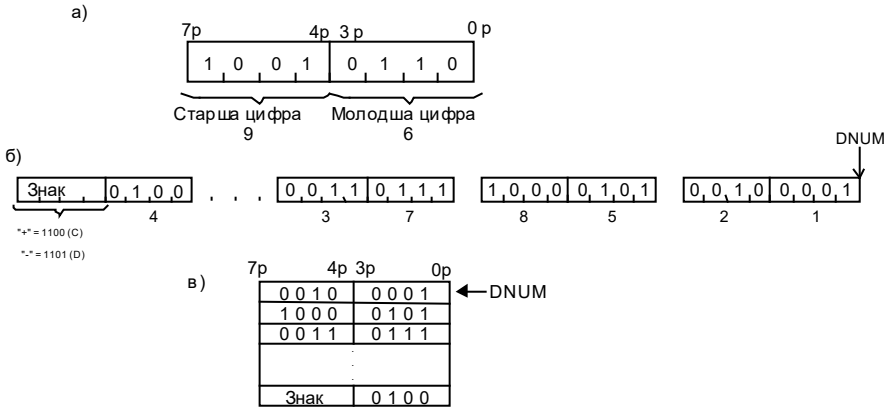

Рисунок 2.1 – Формат упакованих десяткових чисел
а – кодування байта, б- багаторозрядне число, в – розміщення багаторозрядного числа в пам'яті

2.1.5.2 Двійково-десяткові числа в неупакованому форматі

Неупакований двійково-десятковий формат наведений на **рисунку 2.2**. В цьому коді десятковим числам відповідають двійкові набори від 0000 0000 (цифра 0) до 0000 1001 (цифра 9), або в шістнадцятеричній системі від 00Н до 09Н. Отже, можна вважати, що власне значення десяткової цифри займає молодшу тетраду і представлено її двійковим еквівалентом, а в старшій тетраді знаходиться комбінація 0000.

Багаторозрядні неупаковані двійково-десяткові числа займають суміжні байти (**рисунок 2.2, б**). Для знака числа призначається старший байт, в якому комбінація 0000 0000 (00Н) означає знак плюс, а комбінація 0000 0001 (01Н) – знак мінус. Такі числа визначаються в програмах їх початковою адресою – DNUM і числом розрядів (байт) або довжиною – N. Розміщення неупакованого двійково-десяткового числа в пам'яті показано на **рисунку 2.2, в**.

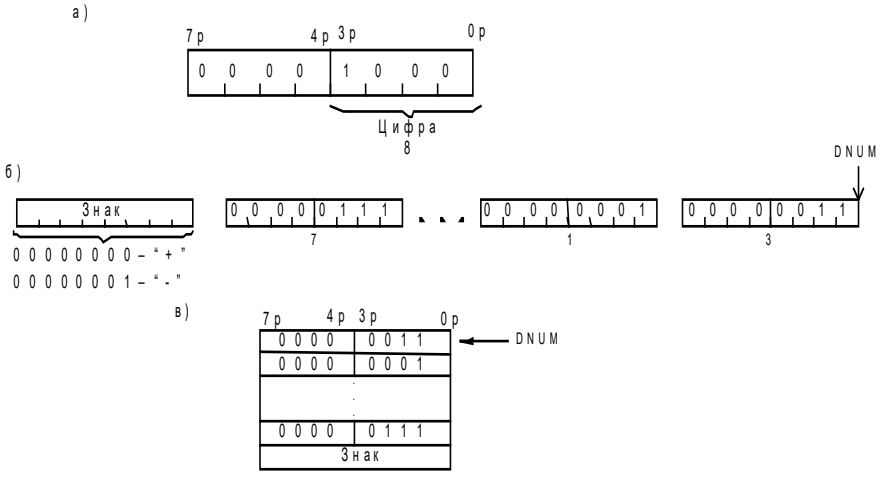

Рисунок 2.2 – Формат неупакованих двійково-десяткових чисел:

а – кодування байта, б – багаторозрядне число, в – розміщення його в пам'яті

Порядок виконання лабораторної роботи

1) Представити кожне число A_{10} B_{16} C_2 у: оберненому коді, доповняльному коді, двійковому коді Грея.

№ варіанту	A_{10}	B_{16}	C_2
1	158	1F	10010111
2	186	C8	11011010
3	129	5D	10000011
4	156	F3	10101100
5	251	FB	11001100
6	186	B9	10111101
7	235	D9	11011011
8	195	A5	11100001
9	178	8F	10010010
10	197	9A	10101010
11	234	D7	11011101
12	221	B8	11111011
13	178	A3	11011110
14	196	CF	10111110
15	251	BD	11111110
16	198	DA	11111101
17	242	F8	11011000
18	189	5C	10100010
19	164	C6	11000010
20	173	6D	10000100
21	129	D7	10001100
22	164	8F	10010100
23	245	1A	10100000
24	239	2B	11010010
25	182	B3	10001111
26	164	4C	10000101
27	171	D5	10101011
28	146	6E	10010101
29	183	E7	11101111
30	238	8E	11011111
31	215	A6	10111110
32	213	FF	10100001

2) Виконати арифметичні операції у двійковому коді: $N_{10} + 85_{10}$; $75_{10} - N_{10}$; $N_{10} - 85_{10}$; $(15_{10}+N_{10}) * 9_{10}$, де N_{10} – номер студента у журналі групи (номер варіанту). Перевірити, переводячи результати у десяткову систему числення.

3) Представити число D_{10} у двійково-десяткових кодах: коді 8-4-2-1, коді Айкена, коді з надлишком 3.

№ варіанта	D	№ варіанта	D
1	406821	17	102876
2	647065	18	638190
3	976395	19	824907
4	482575	20	935716
5	520786	21	824650
6	795046	22	150483
7	856954	23	159736
8	739460	24	651984
9	851526	25	489150
10	864894	26	230154
11	828202	27	108756
12	185470	28	783105
13	296581	29	468021
14	185692	30	792183
15	274580	31	832459
16	952831	32	824586

4) Виконати перетворення однобайтного числа «215+N (№ варіанта) на двійково-десятковий код 8-4-2-1 шляхом використання операцій зсуву.

5) Виконати арифметичні дії у двійково-десяткових кодах:

578+(N*15); код «8-4-2-1», код Айкена («2-4-2-1»), код з надлишком 3:

597-(N*15); код «8-4-2-1», код Айкена («2-4-2-1»), код з надлишком 3

257- (350+N*13); код «8-4-2-1», код Айкена («2-4-2-1»), код з надлишком 3

Примітка: дії у дужках звичайним десятковим множенням на номер варіанту.

6) Зобразити у 2-х байтному рядку пам’яті, де 10 біт відведено на мантису і 4 біта – порядок, формат з плаваючою комою числа: (187+N); (-255+N); (8,5*N/10000);

((N+50)/-0,57). Обчислити діапазон чисел, що можуть бути відображені у даному рядку пам’яті у форматі з плаваючою комою.

7) Розробити алгоритм, який забезпечував би перетворення двійкового коду у двійково-десятковий, і написати програму на будь-якій мові програмування.