5. Systemy maszynowe kodowania liczb.

5.1. Kody binarne

Ze wszystkich systemów pozycyjnych, system pozycyjny przy podstawie $p = 2$ wymaga najmniejszej liczby cyfr do kodowania liczb, czyli tylko dwóch.

Wadą tego systemu są długie reprezentacje liczb. Do zalet zaś należy wymienić łatwość wykonywania operacji arytmetycznych oraz fakt, że cyfry są reprezentowane tylko przez dwa znaki.

Problem długości reprezentacji jest technicznie prostszy do rozwiązania w porównaniu z techniczną realizacją działań arytmetycznych. Z tego względu konstruowane współcześnie maszyny cyfrowe oparte są na systemie pozycyjnym dwójkowym. Przetwarzają one informacje zakodowane w postaci słów binarnych, czyli dwuwartościowych ciągów o ustalonych długościach. Te dwie wartości z reguły utożsamiamy z cyframi ”0” i ”1” systemu dwójkowego.

Zapis informacji przy użyciu słów binarnych nazywamy kodowaniem binarnym, a ciągi zapisów złożonych z ”0” i ”1” nazywamy kodami binarnymi. W maszynach cyfrowych

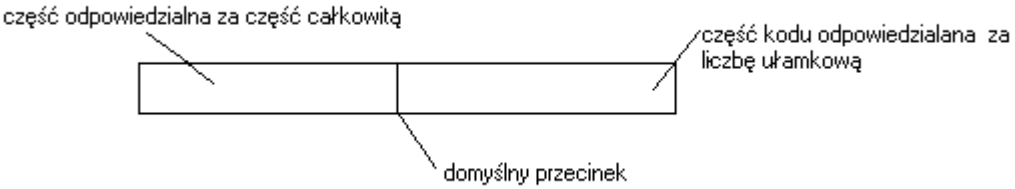
oprócz kodowania liczb zachodzi potrzeba kodowania obiektów nieliczbowych, np. znaków alfabetu języka naturalnego, znaków matematycznych i innych znaków specjalnych używanych przy redakcji tekstów. Służą temu specjalne tabele konwersji kodów binarnych jak np. kod ASCII.

W dalszym ciągu zajmiemy się wyłącznie problemem kodowania binarnego liczb mając do dyspozycji kody binarne o ustalonej długości. Bezpośrednie przeniesienie systemu dwójkowego dla potrzeb kodowania binarnego liczb napotyka trudności spowodowane określoną długością kodu binarnego, potrzebą zakodowania znaku liczby oraz potrzebą zakodowania znaku rozdzielającego część całkowitą liczby od części ułamkowej. Z uwagi na znak rozdzielający stosuje się zazwyczaj dwa systemy binarne kodowania liczb zwane systemami kodowania: stałopozycyjnego oraz zmiennopozycyjnego.

5.2. Kody stałopozycyjne

Kody stałopozycyjne, zwane również kodami stałoprzecinkowymi, charakteryzują się ustaloną pozycją

przecinka w kodzie binarnym. Zakładając, że mamy ustaloną długość kodu binarnego *n*, ustalamy domyślnie położenie przecinka rozdzielającego kod na dwie części:



Oznaczając przez n_c liczbę bitów części całkowitej, zaś przez n_u liczbę bitów części ułamkowej otrzymujemy zależność $n_c + n_u = n$. W dalszym ciągu pozycja przecinka będzie ustalana przez parametr całkowity *m* z domyślną wartością 0. Oznacza ona, że wszystkie *n* bitów słowa binarnego przeznaczone są na zakodowanie części całkowitej, czyli $n_c = n$ i $n_u = 0$. W przypadku, gdy $n_u = m > 0$ przyjmujemy, że słowo binarne koduje liczbę całkowitą pomnożoną przez czynnik skalujący 2^{-m}. Odpowiada to przesunięciu przecinka z pozycji domyślnej ($n_u = 0$) o *m* bitów w lewo. Zasadę przemnażania przez czynnik skalujący 2^{-m} rozszerzamy na przypadek dowolnej wartości całkowitej parametru *m*, czyli stosujemy ją nawet wtedy, gdy $m > n$ lub $m < 0$.

5.2.1 Naturalny kod binarny NKB (NBC) [1]

Funkcją kodującą związaną w naturalny sposób z funkcją kodującą L_2 systemu pozycyjnego dwójkowego jest funkcja NKB o dwóch argumentach opcjonalnych [n] i [m] przyjmujących w razie ich pominięcia wartości domyślne oraz argumencie obowiązkowym (a), którego wartością jest ciąg binarny $a = "a_{n-1}...a_1a_0"$ traktowany jako łańcuch znaków ”0” lub ”1” o długości n. Wartości funkcji NKB określone są wzorem

$$\mathrm{NKB}[n](a)[m] := L_2(a)\cdot 2^{-m} = 2^{-m}\sum_{k=0}^{n-1} L_2("a_k")\cdot 2^k$$

$$= 2^{-m}\sum_{k=0}^{n-1} a_k \cdot 2^k$$

Stąd zakres liczb reprezentowanych przez kod NKB jest opisywany przez zbiór $\{k\cdot 2^{-m} : k \in \mathrm{Z}, 0 \le k \le 2^n - 1\}$. Przyjęte tutaj konwencje dotyczące argumentów m, n i a będą obowiązywać do końca rozdziału 5.2.

---

[1] Naturalny Kod Binarny NKB (ang. *Natural binary code* – NBC*)*

5.2.2. Reprezentacja znak – moduł ZM [2]

Najbardziej naturalnym sposobem kodowania liczb zarówno dodatnich jak i ujemnych jest przyjęcie, że bit najbardziej znaczący $a_{n\text{-}1}$ reprezentuje znak liczby, zaś pozostałe bity reprezentują wartość bezwzględną liczby. Realizuje to funkcja kodująca ZM określona wzorem

$$\text{ZM}\ [n](a)[m] = (-1)^{a_{n-1}} L_2("a_{n-2}...a_1a_0") \cdot 2^{-m}$$

$$= (-1)^{a_{n-1}}\ \text{NKB}\ [n-1]("a_{n-2}...a_1a_0") \cdot 2^{-m}$$

Stąd zakres liczb reprezentowanych przez kod ZM jest opisywany przez zbiór $\{k \cdot 2^{-m} : k \in Z, | k | \leq 2^{n-1} - 1\}$. Z definicji funkcji kodującej wynika ponadto, że

$\text{ZM}[n]("a_{n-1}...a_1a_0")[m] = 0$ wtedy i tylko wtedy gdy $a_k = 0$ dla k = 0,1,2,...,n-1, lub

$a_{n-1} = 1$ i $a_k = 0$ dla k=0,1,2,…,n-2. Zatem liczba 0 może być reprezentowana podwójnie

---

[2] Znak – Moduł (ang. *Sign – magnitude* – SM)

przez ciąg binarny ”00...00” złożony z samych zer, bądź ciąg binarny ”10...00” z jedynką na pierwszym miejscu i zerami na pozostałych miejscach. Pozostałe liczby z zakresu muszą więc być reprezentowane jednoznacznie. Ponadto

$$|\,\mathrm{ZM}[n](\text{"}a_{n-1}\ldots a_1a_0\text{"})[m]\,| = L_2(\text{"}a_{n-2}\ldots a_1a_0\text{"})\cdot 2^{-m}$$

i w konsekwencji przy założeniu, że ZM[n](*a*)[m]≠0 mają miejsce równoważności:

$$\mathrm{ZM}[n](\text{"}a_{n-1}\ldots a_1a_0\text{"})[m] = |\,\mathrm{ZM}[n](\text{"}a_{n-1}\ldots a_1a_0\text{"})[m]\,| > 0$$

wtedy i tylko wtedy $a_{n-1}=0$

oraz

$$\mathrm{ZM}[n](\text{"}a_{n-1}\ldots a_1a_0\text{"})[m] = -|\,\mathrm{ZM}[n](\text{"}a_{n-1}\ldots a_1a_0\text{"})[m]\,| < 0$$

wtedy i tylko wtedy $a_{n-1}=1$.

Zatem istotnie najbardziej znaczący bit a_{n-1} reprezentuje znak kodowanej liczby, zaś pozostałe bity reprezentują wartość bezwzględną kodowanej liczby. Mimo swojej naturalności z punktu widzenia człowieka kod ZM nie jest obecnie stosowany w maszynach cyfrowych z powodu podwójnej reprezentacji zera oraz specjalnego traktowania bitu a_{n-1}, co utrudnia tworzenie algorytmów wykonujących działania arytmetyczne.

5.2.3. Reprezentacja uzupełnienia do jedności U1 [3]

Wady kodu ZM częściowo eliminuje kod U1. Podobnie jak w przypadku kodu ZM bit najbardziej znaczący $a_{n\text{-}1}$ reprezentuje znak liczby, ale tym razem pozostałe bity nie reprezentują wartości bezwzględnej liczby. Dzieje się tak dlatego, że bit $a_{n\text{-}1}$ jest traktowany podobnie jak pozostałe bity tyle, że z inną wagą $-(2^{n\text{-}1}-1)$. Dokładniej funkcja kodująca U1 określona jest wzorem

$$\mathrm{U1}[n](a)[m] := (-a_{n-1}\cdot(2^{n-1}-1)+L_2("a_{n-2}\ldots a_1a_0"))\cdot 2^{-m}$$
$$= -a_{n-1}\cdot(2^{n-1}-1)\cdot 2^{-m}+\mathrm{NKB}[n-1]("a_{n-2}\ldots a_1a_0")[m]$$

Stąd zakres liczb reprezentowanych przez kod U1 jest opisywany przez zbiór

$\{k\cdot 2^{-m} : k\in Z, |k|\le 2^{n-1}-1\}$. Z definicji funkcji kodującej wynika ponadto, że

$\mathrm{U1}[n]("a_{n-1}\ldots a_1a_0")[m]=0$ wtedy i tylko wtedy gdy $a_k=0$ dla k= 0,1,2,...,n-1, lub $a_k=1$ dla k=0,1,2,…,n-1. Zatem liczba 0 może być reprezentowana podwójnie

---

[3] U1- uzupełnienia do jedności (ang. *1's complement* U1)

przez ciąg binarny ”00...00” złożony z samych zer, bądź ciąg binarny ”11...11” złożony z samych jedynek. Pozostałe liczby z zakresu muszą więc być reprezentowane jednoznacznie. Ponadto $0 \le L_2("a_{n-2}...a_1a_0") \le 2^{n-1}-1$ i w konsekwencji przy założeniu, że U1[*n*](*a*)[*m*]≠0 mają miejsce równoważności:

$$\mathrm{U1}[n]("a_{n-1}...a_1a_0")[m] = |\,\mathrm{U1}[n]("a_{n-1}...a_1a_0")[m]\,|$$
$$= L_2("a_{n-2}...a_1a_0")2^{-m} > 0$$

wtedy i tylko wtedy $a_{n-1} = 0$ oraz

$$\mathrm{U1}[n]("a_{n-1}...a_1a_0")[m] = -|\,\mathrm{U1}[n]("a_{n-1}...a_1a_0")[m]\,|$$
$$= -(2^{n-1}-1-L_2("a_{n-2}...a_1a_0"))2^{-m} < 0$$

wtedy i tylko wtedy $a_{n-1} = 1$. Zatem istotnie najbardziej znaczący bit a_{n-1} reprezentuje znak kodowanej liczby. Wadą tego kodu jest w dalszym ciągu podwójna reprezentacja zera, zaś zaletą prosty mechanizm uzyskiwania reprezentacji liczby przeciwnej –U1[*n*](*a*)[*m*] przez zanegowanie wszystkich bitów reprezentacji *a*.

5.2.4. Reprezentacja uzupełnienia do dwóch U2 [4]

[4] U2- uzupełnienia do dwóch (ang. *2's complement U*2)

Podwójną reprezentację zera w kodzie U1 eliminuje kod U2. Efekt ten uzyskuje się przez zastąpienie wagi $-(2^{n-1}-1)$ wagą -2^{n-1} dla najbardziej znaczącego bitu a_{n-1}. Dokładniej funkcja kodująca U2 określona jest wzorem

$$\mathrm{U}2[n]("a_{n-1}a_{n-2}...a_1a_0")[m] := (-a_{n-1}2^{n-1} + L_2("a_{n-1}...a_1a_0")) \cdot 2^{-m}$$
$$= -a_{n-1}2^{n-1} \cdot 2^{-m} + \mathrm{NKB}[n-1]("a_{n-1}...a_1a_0")[m]$$

Stąd zakres liczb reprezentowanych przez kod U2 jest opisywany przez zbiór

$\{k \cdot 2^{-m} : k \in Z, -2^{n-1} \le k \le 2^{n-1}-1\}$, a więc każda liczba z tego zakresu musi być reprezentowana w dokładnie jeden sposób. W szczególności $U2[n]("a_{n-1}...a_1a_0")[m] = 0$ wtedy i tylko wtedy gdy $a_k = 0$ dla k = 0,1,2,...,n-1. Zatem liczba 0 jest reprezentowana przez dokładnie jeden ciąg binarny ”00...00” złożony z samych zer. Ponadto $0 \le L_2("a_{n-2}...a_1a_0") \le 2^{n-1}-1$ i w konsekwencji mają miejsce równoważności:

$$U2[n]("a_{n-1}...a_1a_0")[m] = |U2[n]("a_{n-1}...a_1a_0")[m]|$$
$$= L_2("a_{n-2}...a_1a_0")2^{-m} \ge 0$$

wtedy i tylko wtedy $a_{n-1} = 0$ oraz

$$U2[n]("a_{n-1}...a_1a_0")[m] = -|U2[n]("a_{n-1}...a_1a_0")[m]|$$
$$= -(2^{n-1} - L_2("a_{n-2}...a_1a_0"))2^{-m} < 0$$

wtedy i tylko wtedy $a_{n-1} = 1$. Zatem najbardziej znaczący bit a_{n-1} reprezentuje znak kodowanej liczby. Zaletą tego kodu jest pojedyncza reprezentacja zera i tym samym jednoznaczna reprezentacja każdej liczby postaci $k \cdot 2^{-m}$, gdzie $k \in Z$ i $-2^{n-1} \leq k \leq 2^{n-1}-1$. Podobnie jak w przypadku kodu U1 kod U2 oferuje stosunkowo prosty mechanizm uzyskiwania reprezentacji liczby przeciwnej $-U2[n](a)[m]$ przez zanegowanie wszystkich bitów reprezentacji *a*, a następnie dodanie binarne jedynki. Dodawanie binarne oznacza operację wykonywaną na ciągach binarnych *x* i *y*, której wartością jest ciąg binarny *z* spełniający zależność $L_2(z) = L_2(x) + L_2(y)$. *Z* uwagi na powyższe własności kod U2 jest obecnie powszechnie stosowany w maszynach cyfrowych.

5.2.5. Reprezentacja przesunięta KP (BIAS) [5]

[5] KS - kod spolaryzowany (ang. *bias*)

Oprócz kodu U2 własność jednoznaczności reprezentacji liczb ma również kod przesunięty, który polega na wykorzystaniu kodu NKB z przesunięciem w dół o połowę zakresu liczb reprezentowanych w tym kodzie. Dokładniej funkcja kodująca KP określona jest wzorem

$$\mathrm{KP}[n]("a_{n-1}a_{n-2}...a_1a_0")[m] := (L_2("a_{n-1}...a_1a_0") - 2^{n-1}) \cdot 2^{-m}$$
$$= (\mathrm{NKB}[n]("a_{n-1}...a_1a_0") - 2^{n-1}) \cdot 2^{-m}$$

Stąd zakres liczb reprezentowanych przez kod KP jest opisywany przez zbiór

$\{k \cdot 2^{-m} : k \in Z, -2^{n-1} \le k \le 2^{n-1} - 1\}$, a więc każda liczba z tego zakresu musi być reprezentowana w dokładnie jeden sposób. W szczególności $KP[n]("a_{n-1}...a_1a_0")[m] = 0$ wtedy i tylko wtedy gdy $a_{n-1} = 1$ i $a_k = 0$ dla k = 0,1,2,...,n-2. Zatem liczba 0 jest reprezentowana przez dokładnie jeden ciąg binarny ”10...00” z jedynką na pierwszym miejscu i zerami na pozostałych miejscach. Ponadto $0 \le L_2("a_{n-2}...a_1a_0") \le 2^{n-1} - 1$ i w konsekwencji mają miejsce równoważności:

$$KP[n]("a_{n-1}...a_1a_0")[m] = |KP[n]("a_{n-1}...a_1a_0")[m]|$$
$$= L_2("a_{n-2}...a_1a_0")2^{-m} \ge 0$$

wtedy i tylko wtedy $a_{n-1} = 1$ oraz

$$KP[n]("a_{n-1}...a_1a_0")[m] = -|KP[n]("a_{n-1}...a_1a_0")[m]|$$
$$= -(2^{n-1} - L_2("a_{n-2}...a_1a_0"))2^{-m} < 0$$

wtedy i tylko wtedy $a_{n-1} = 0$. Zatem najbardziej znaczący bit a_{n-1} reprezentuje znak kodowanej liczby z tym, że tym razem bit ”0” oznacza znak minus zaś bit ”1” oznacza znak plus. Kod KP jest stosowany przy kodowaniu cechy liczby w reprezentacji zmiennopozycyjnej.

5.2.6. Reprezentacja binarno-dziesiętna BKD (BCD) [6]

Istota kodowania binarno-dziesiętnego polega na użyciu dziesiętnego systemu pozycyjnego (używanego w życiu codziennym), gdzie kolejne cyfry znaczące rozwinięcia dziesiętnego są reprezentowane przy użyciu 4-bitowego kodu NKB z dołączonym ewentualnie bitem znaku. Zatem dla dowolnie ustalonych liczb n ∈ N i m ∈ Z otrzymujemy:

$$BKD[4n]("a_{4n-1}a_{4n-2}a_{4n-3}a_{4n-4}...a_3a_2a_1a_0")[m] :=$$
$$= 10^{-m} \sum_{k=0}^{n-1} L_2("a_{4k+3}a_{4k+2}a_{4k+1}a_{4k}")10^k =$$

[6] BKD – binarny kod dziesiętny (ang. *binary coded decimal* BCD)

$$= 10^{-m} \sum_{k=0}^{n-1} NKB[4]("a_{4k+3}a_{4k+2}a_{4k+1}a_{4k}")[0] \cdot 10^k$$

oraz

$$BKD[4n+1]("a_{4n}a_{4n-1}a_{4n-2}a_{4n-3}a_{4n-4}...a_3a_2a_1a_0")[m] :=$$

$$= (-1)^{a_{4n}} 10^{-m} \sum_{k=0}^{n-1} L_2("a_{4k+3}a_{4k+2}a_{4k+1}a_{4k}") \cdot 10^k =$$

$$= (-1)^{a_{4n}} 10^{-m} \sum_{k=0}^{n-1} NKB[4]("a_{4k+3}a_{4k+2}a_{4k+1}a_{4k}")[0] \cdot 10^k$$

Z powyższych równości wynika zależność

$$BKD[4n+1]("a_{4n}a_{4n-1}a_{4n-2}...a_3a_2a_1a_0")[m] =$$

$$= (-1)^{a_{4n}} BKD[4n]("a_{4n-1}a_{4n-2}...a_3a_2a_1a_0")[m]$$

Oznacza ona, że bit ”a_{4n}” jest w 4n+1 bitowym kodzie BKD bitem znaku liczby, przy czym bit ”0” oznacza znak plus a bit ”1” znak minus. Podobnie jak w przypadku kodu ZM liczba zero jest reprezentowana podwójnie przez ciąg binarny złożony z samych zer bądź przez ciąg binarny z jedynką na pierwszym miejscu i zerami na pozostałych miejscach. Bity od ”a_0” do ”a_{4n-1}” reprezentują wartość bezwzględną kodowanej liczby. Każde z czterobitowych słów ”$a_{4k+3}a_{4k+2}a_{4k+1}a_{4k}$” reprezentuje w kodzie NKB jedną z liczb całkowitych od 0 do 9. Zatem nie wszystkie 4

bitowe słowa są dopuszczalne. Dopuszczalne słowa prezentuje następująca tabela.

X	NKB[4](x)[0]
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9

Tabela ta daje jednocześnie kody konwersji z systemu dziesiętnego na reprezentacje w kodzie BKD. Zakres liczb reprezentowanych przez 4*n* bitowy ciąg w kodzie BKD jest opisywany przez zbiór $\{k \cdot 10^{-m} : k \in \mathrm{Z}, 0 \le k \le 10^{n-1} - 1\}$ W przypadku reprezentacji 4*n*+1 bitowych jest to zbiór $\{k \cdot 10^{-m} : k \in Z, | k | \le 10^{n-1} - 1\}$.

5.2.7. Reprezentacja minus-dwójkowa (KMD) [7]

Reprezentacja minus-dwójkowa ma dzisiaj jedynie historyczne znaczenie. Umożliwia ona kodowanie zarówno liczb dodatnich jak i ujemnych przez formalne zastąpienie podstawy 2 w funkcji kodującej L_2 przez podstawę –2. Precyzuje to definicja

$$KMD[n]("a_{n-1}a_{n-2}...a_1a_0")[m] := 2^{-m}\sum_{k=0}^{n-1} L_2(a_k)\cdot(-2)^k$$

Przykładowo wartość liczbowa słowa ”11001” w kodzie minus-dwójkowym będzie równa

$$KMD[5](11001)[0]$$
$$= 1\times(-2)^4 + 1\times(-2)^3 + 0\times(-2)^2 + 0\times(-2)^1 + 1\times(-2)^0$$
$$= 16 - 8 + 1 = 9$$

[7] KMD – kod minus – dwójkowy.