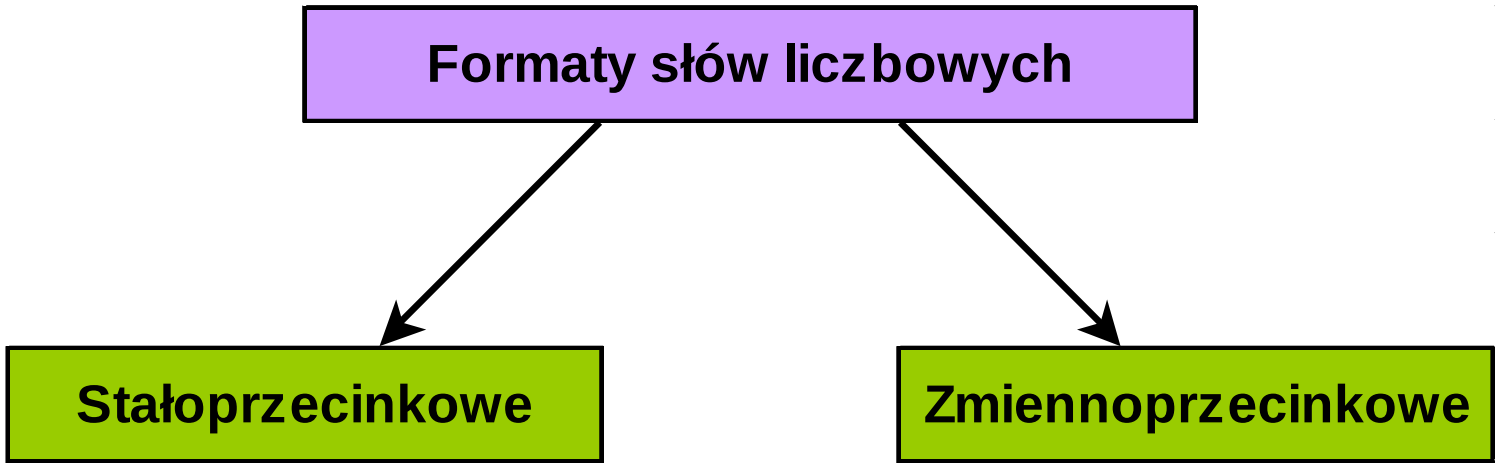
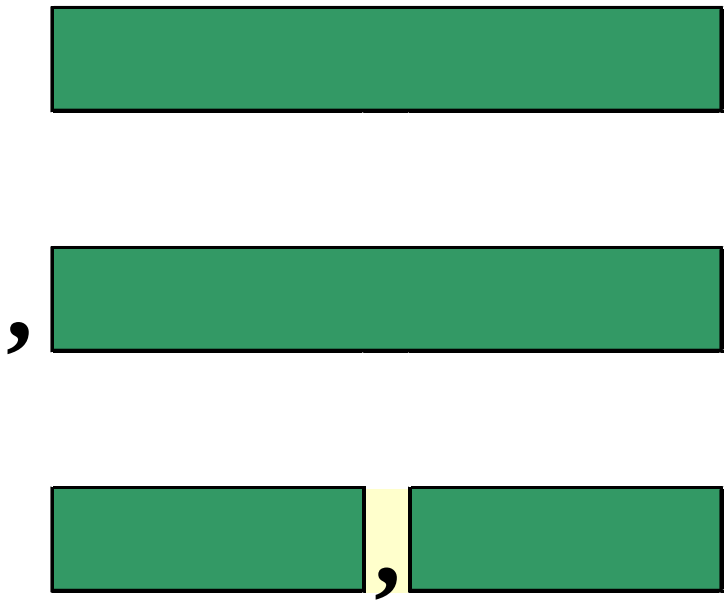
Reprezentacja liczb w komputerze



Formaty stałoprzecinkowe



W formatach stałopozycyjnych przecinek umieszczony jest w stałej pozycji

Liczby stałoprzecinkowe mogą występować:

- z przecinkiem na skrajnej prawej pozycji (reprezentacja *całkowitoliczbowa*)
- z przecinkiem na skrajnej lewej pozycji (reprezentacja *ułamkowa*)
- jako liczba zawierająca część całkowitą i ułamkową stałego rozmiaru

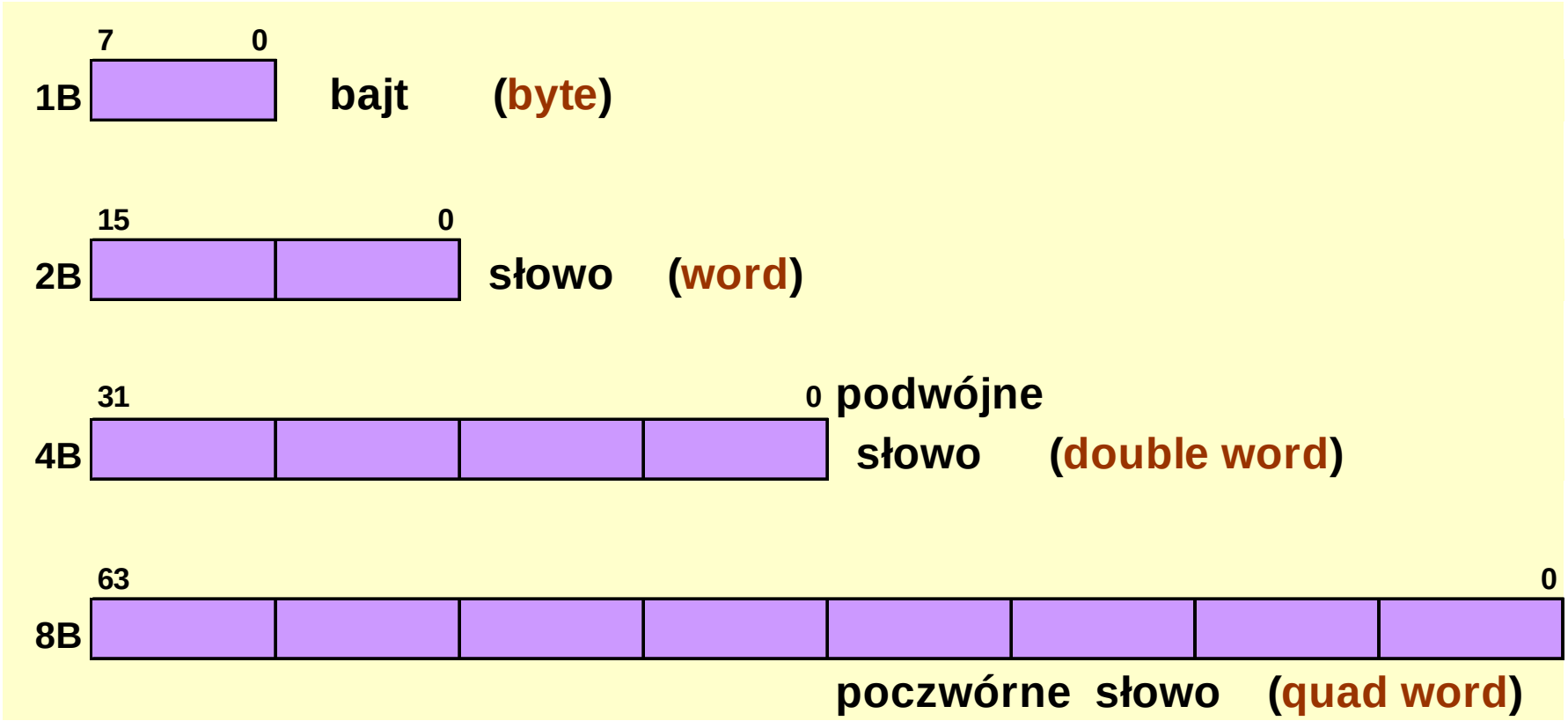
Formaty zmiennoprzecinkowe

Zapis wykładniczy liczby:

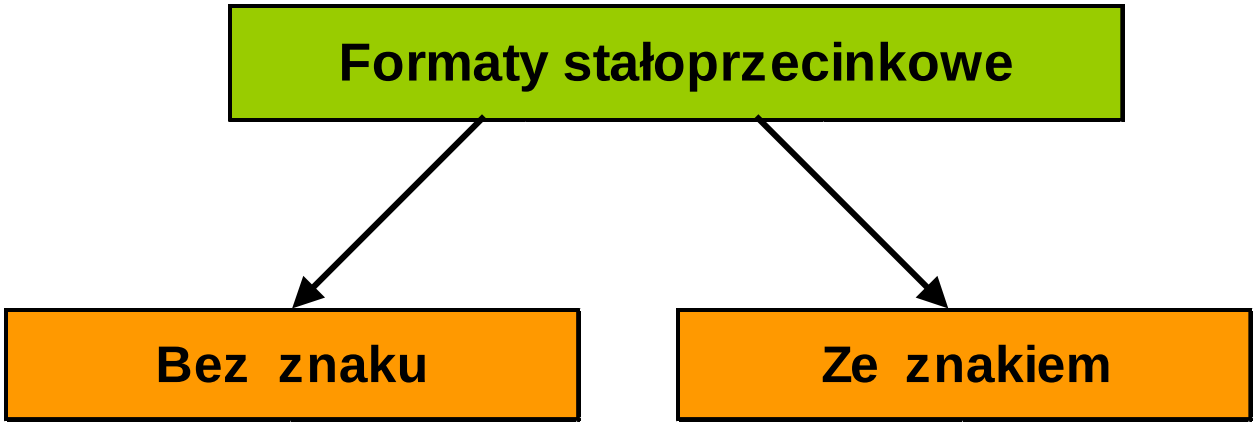
$$L = (-1)^S \cdot M \cdot 2^E$$

Formaty słów maszynowych

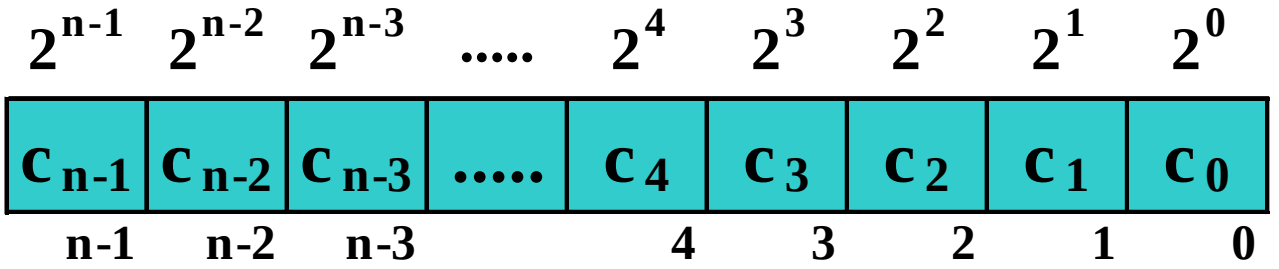
Szerokość słowa maszynowego zwykle wynosi 1, 2, 4 lub 8 bajtów



Formaty stałoprzecinkowe



Liczby całkowite bez znaku

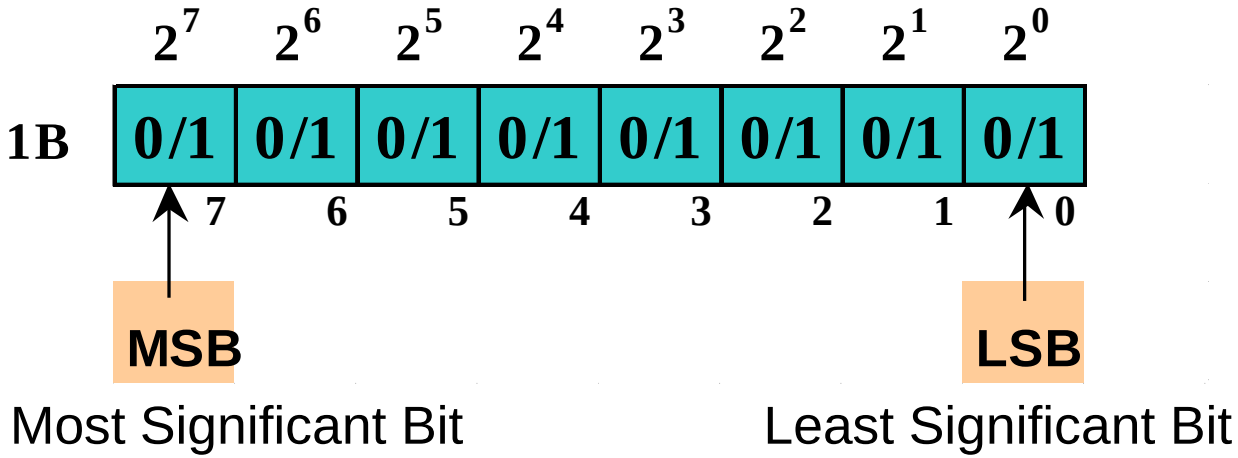


- Liczby nieujemne przechowują się w naturalnym kodzie binarnym – ***NKB***
- Wartość liczby całkowitej **bez znaku** zapisanej w n-bitowym słowie maszynowym wynosi:

$$L = c_0 \cdot 2^0 + c_1 \cdot 2^1 + c_2 \cdot 2^2 + \ldots + c_{n-2} \cdot 2^{n-2} + c_{n-1} \cdot 2^{n-1} =$$

$$= \sum_{i=0}^{n-1} c_i 2^i$$

Zakres liczb całkowitych bez znaku

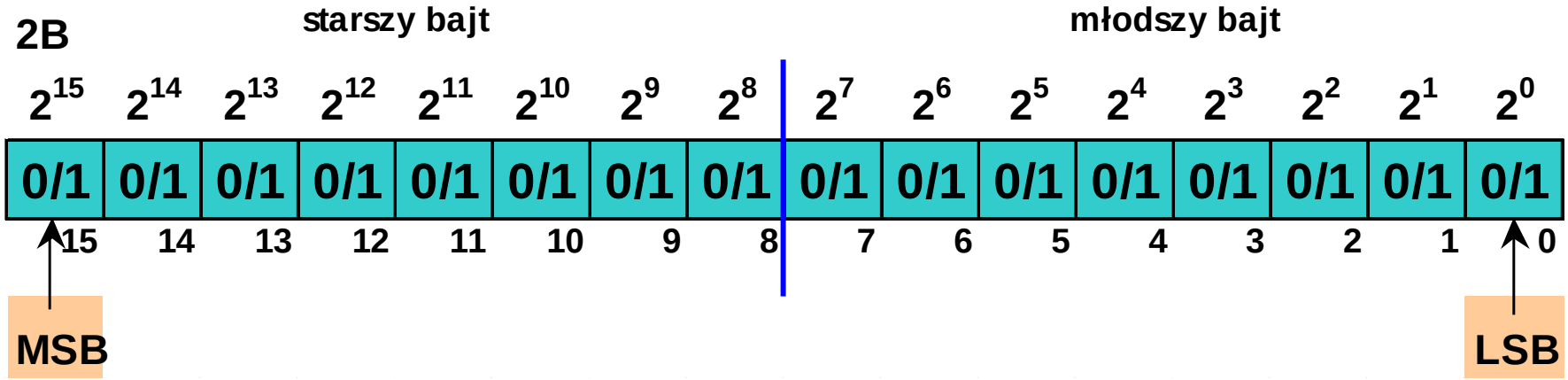


- Zakres liczb wynosi: $0 \leq L \leq 2^n-1$
- Zakres liczb całkowitych bez znaku przechowywanych w jednym bajcie (1B) wynosi

0 ... 255

- $00000000_{(2)}$ = 0 (minimalna liczba)
- $11111111_{(2)}$ = 2^8-1 = 255 (maksymalna liczba)

Zakres liczb całkowitych bez znaku



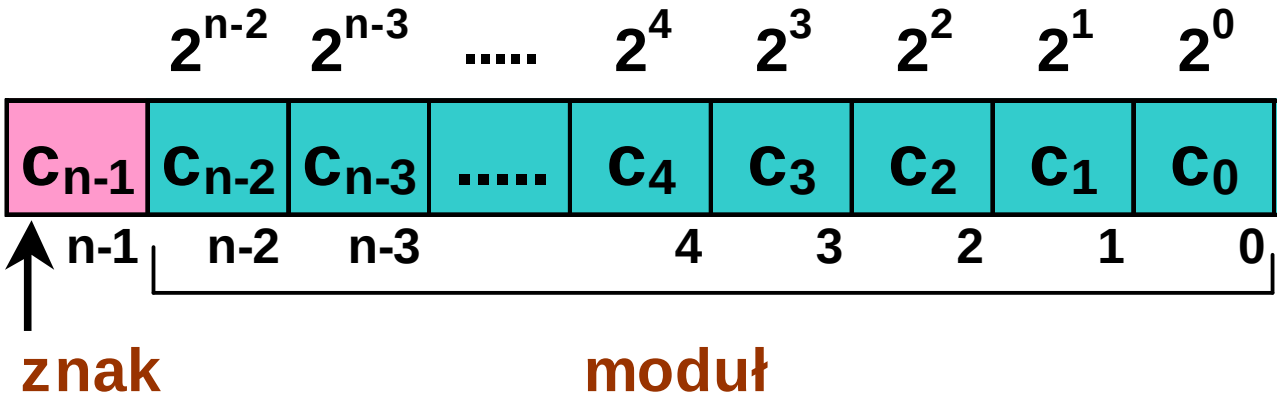
- Zakres liczb całkowitych bez znaku przechowywanych w dwubajtowym słowie (word) wynosi **0 ... 65 535**

- $00000000\ 00000000_{(2)} = 0$
- $11111111\ 11111111_{(2)} = 2^{16}-1 = 65\ 535$

Liczby całkowite ze znakiem

- Do przedstawienia liczb całkowitych ze znakiem stosowane są następujące kody:
  - ZM (znak-moduł)
  - U1 (uzupełnienie do 1)
  - U2 (uzupełnienie do 2)
  - Kod spolaryzowany (z przesunięciem BIAS)

Kod Znak-Moduł



- W kodzie znak-moduł wszystkie bity liczby poza najstarszym mają takie same znaczenie jak w kodzie ***NKB***
- Najstarszy bit jest bitem znaku: 0 - liczba dodatnia,
  1 - liczba ujemna
- Wartość liczby wynosi:

$$L = (-1)^{c_{n-1}} \sum_{i=0}^{n-2} c_i 2^i$$

Reprezentacja liczb w kodzie ZM

Liczba(10)	Kod ZM	Liczba(10)	Kod ZM
+0	00000000	-0	10000000
1	00000001	-1	10000001
2	00000010	-2	10000010
3	00000011	-3	10000011
4	00000100	-4	10000100
5	00000101	-5	10000101
6	00000110	-6	10000110
7	00000111	-7	10000111
8	00001000	-8	10001000
9	00001001	-9	10001001
10	00001010	-10	10001010

Kod Znak-Moduł

- Występują dwie reprezentacje zera: +0 (00000000)
  –0 (10000000)

- Zakres liczb w formacie ZM: $-2^{n-1}+1 \leq L \leq 2^{n-1}-1$

- Zakres liczb *8-bitowych* w kodzie ZM: **-127 ...+127**.
  $11111111_{(ZM)} = -2^7+1 = -127$ (minimalna)
  $01111111_{(ZM)} = 2^7-1 = 127$ (maksymalna)

- Zakres liczb *16-bitowych*: **-32 767 ...+32 767**

$11111111\ 11111111_{(ZM)} = -2^{15}+1 = -32\ 767$ (minimalna)
$01111111\ 11111111_{(ZM)} = 2^{15}-1 = 32\ 767$ (maksymalna)

Przedstawienie liczby dziesiętnej w kodzie ZM

- znak liczby zakodować w starszym bicie słowa maszynowego
- moduł liczby przedstawić w kodzie NKB
- rozszerzyć liczbę zerami z lewej strony do formatu słowa maszynowego

$-117_{(10)} = L_{(ZM)}$

$|-117_{(10)}| = 117_{(10)} = 1110101_{(NKB)}$

$\mathbf{1}1110101_{(ZM)}$ (w formacie 8-bitowym)

$\mathbf{1}0000000\ 01110101_{(ZM)}$ (w formacie 16-bitowym)

Zamiana ZM → 10

$$L = (-1)^{c_{n-1}} \sum_{i=0}^{n-2} c_i 2^i$$

10101010$_{(ZM)}$ = L$_{(10)}$

L$_{(10)}$ = **(-1)^{1}** · (0·2^6 + 1·2^5 + 0·2^4 + 1·2^3 + 0·2^2 + 1·2^1 + 0·2^0) = -(32+8+2) = **- 42$_{(10)}$**

01001001$_{(ZM)}$ = L$_{(10)}$

L$_{(10)}$ = **(-1)^{0}** · (1·2^6 + 0·2^5 + 0·2^4 + 1·2^3 + 0·2^2 + 0·2^1 + 1·2^0) = 64+8+1 = **73$_{(10)}$**

Uzupełnienia liczb

W pozycyjnym systemie liczbowym o podstawie P dla liczby ***L*** definiuje się dwa rodzaje uzupełnień:

- uzupełnienie P-te:

$$\overline{\overline{L}} = P^n - L \qquad \text{dla } L \neq 0$$

$$\overline{\overline{L}} = 0 \qquad \text{dla } L = 0$$

- uzupełnienie (P-1)-sze:

$$\overline{L} = P^n - L - P^i$$

i – numer pozycji najmłodszej cyfry liczby

n – liczba cyfr w części całkowitej

Uzupełnienia liczb

- Uzupełnienie P-te można uzyskać przez dodanie do uzupełnienia (P-1)-szego wartości P^i :

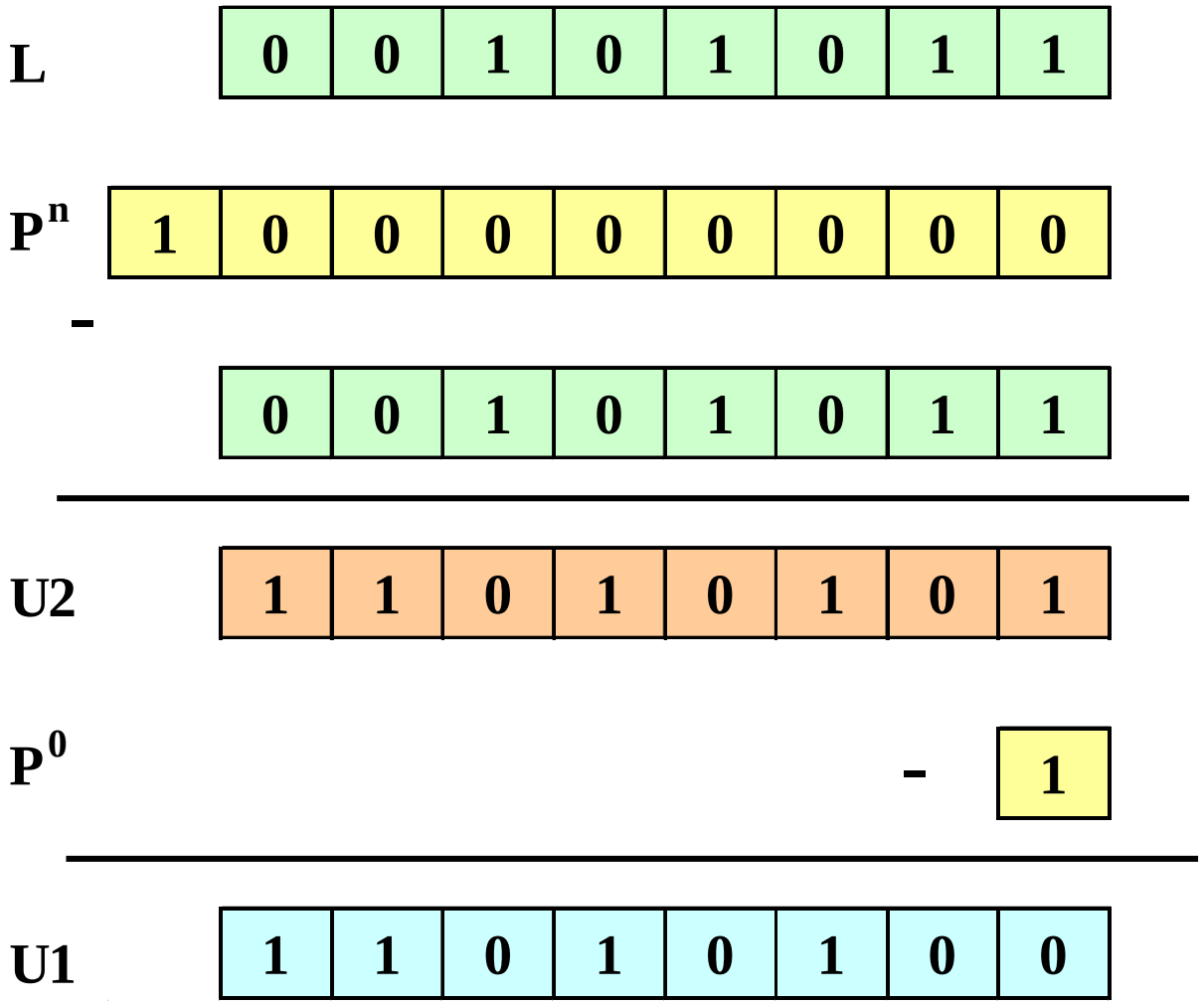
$$\overline{\overline{L}} = \overline{L} + P^i$$

- Dwukrotne uzupełnienie liczby pozwala uzyskać jej pierwotną wartość:

$$\overline{\overline{P^n - L}} = P^n - (P^n - L) = L$$

$$\overline{\overline{P^n - L - P^i} - P^i} = P^n - (P^n - L - P^i) - P^i = L$$

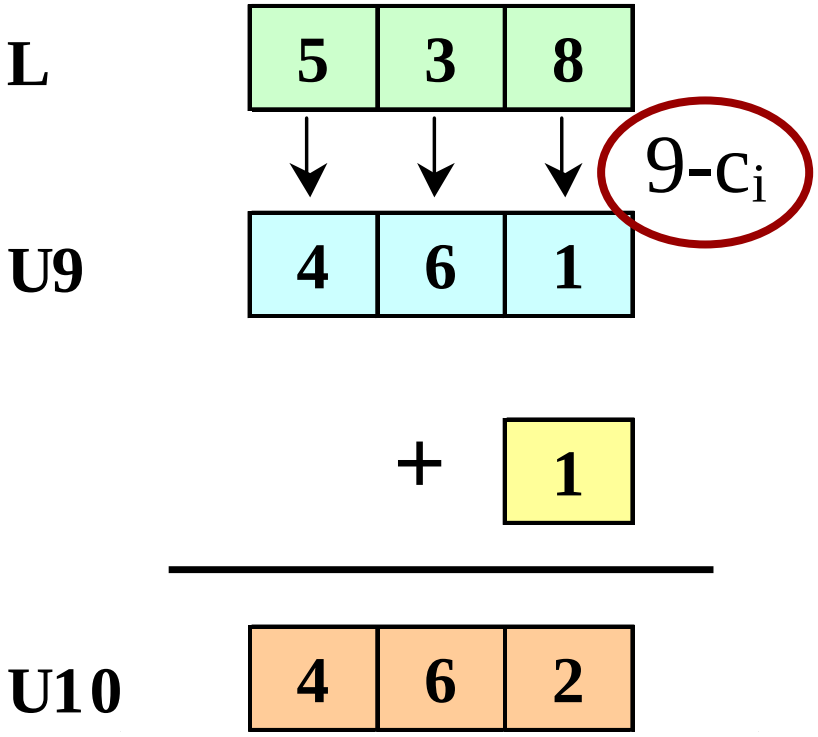
$00101011_{(2)}$ → U2, U1

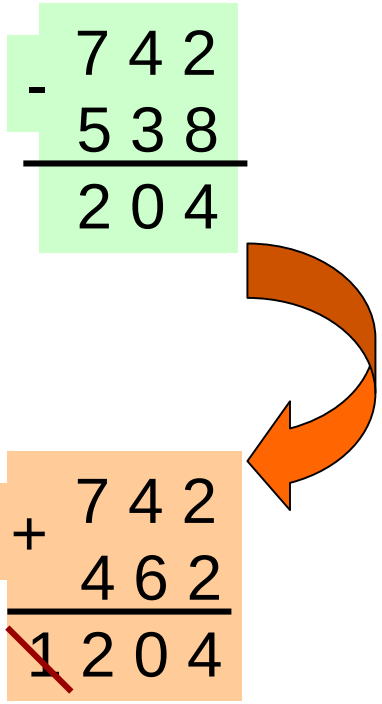


Uzupełnienia liczb

- Uzupełnienie (P-1)-sze można utworzyć przez odjęcie każdej cyfry liczby od (P-1)
- Dla uzupełnienia P-go do tak obliczonej wartości należy jeszcze dodać 1 do najmniej znaczącej pozycji liczby

$$538_{(10)} \rightarrow \mathbf{U10, U9}$$

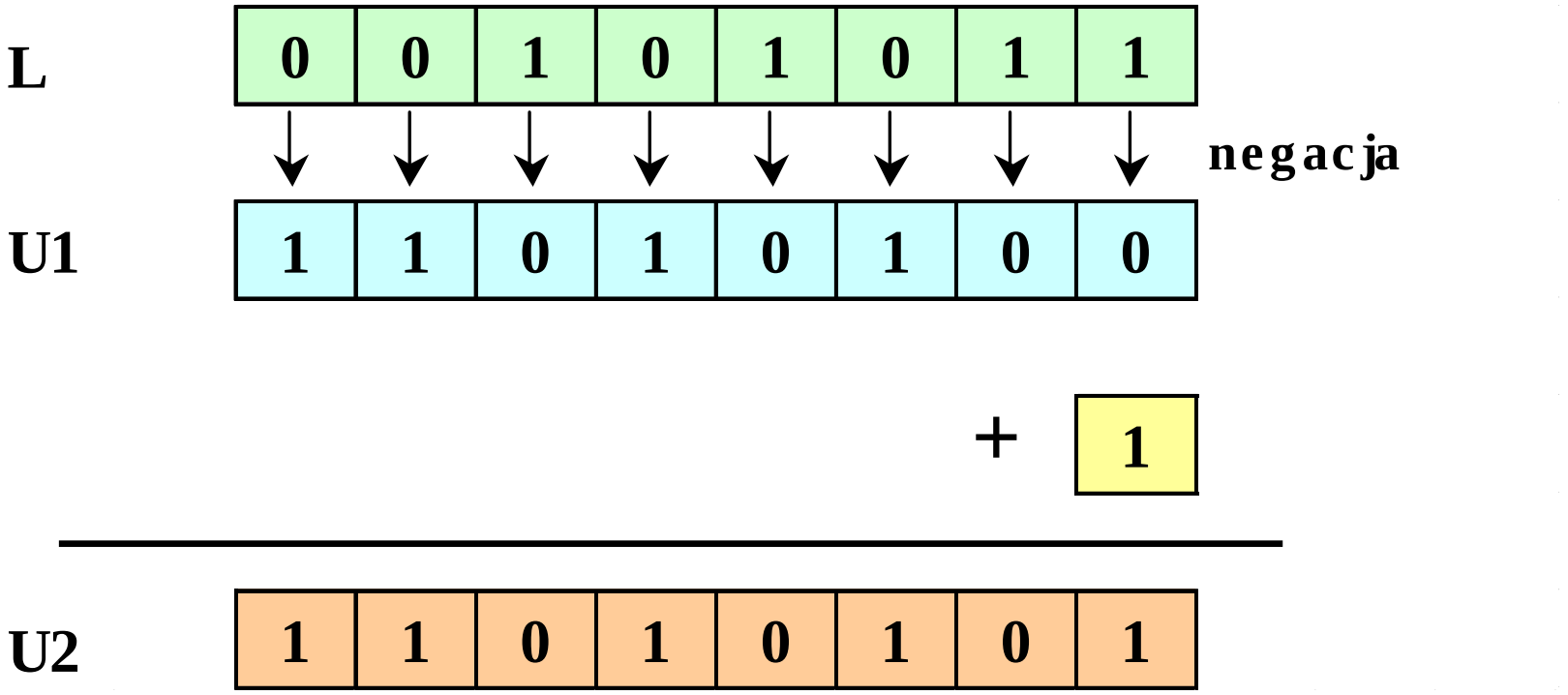




Uzupełnienia U1, U2

- Dla systemu binarnego obliczenie uzupełnień P-1 cyfr sprowadza się do negacji poszczególnych bitów

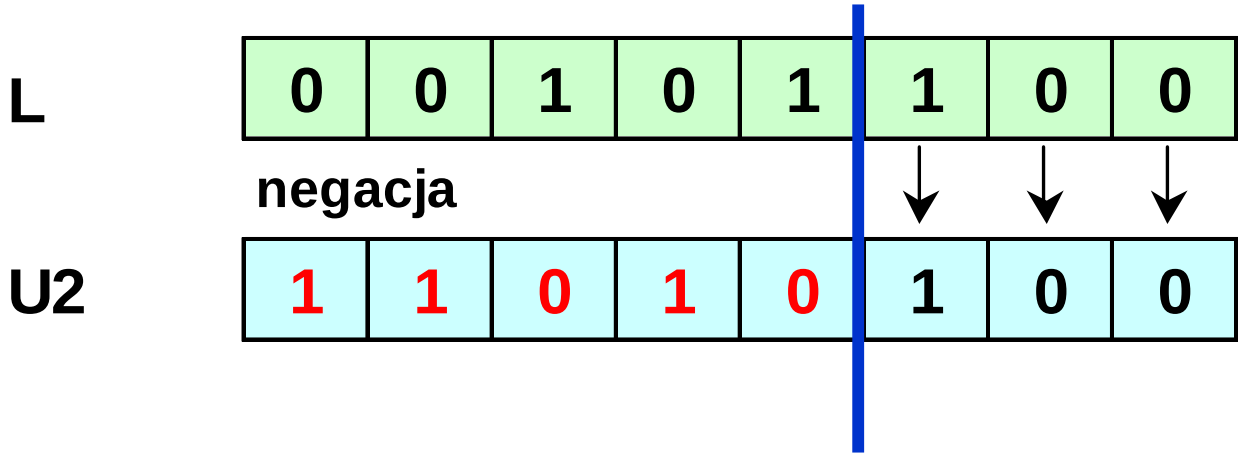
$$00101011_{(2)} \rightarrow \text{U2, U1}$$



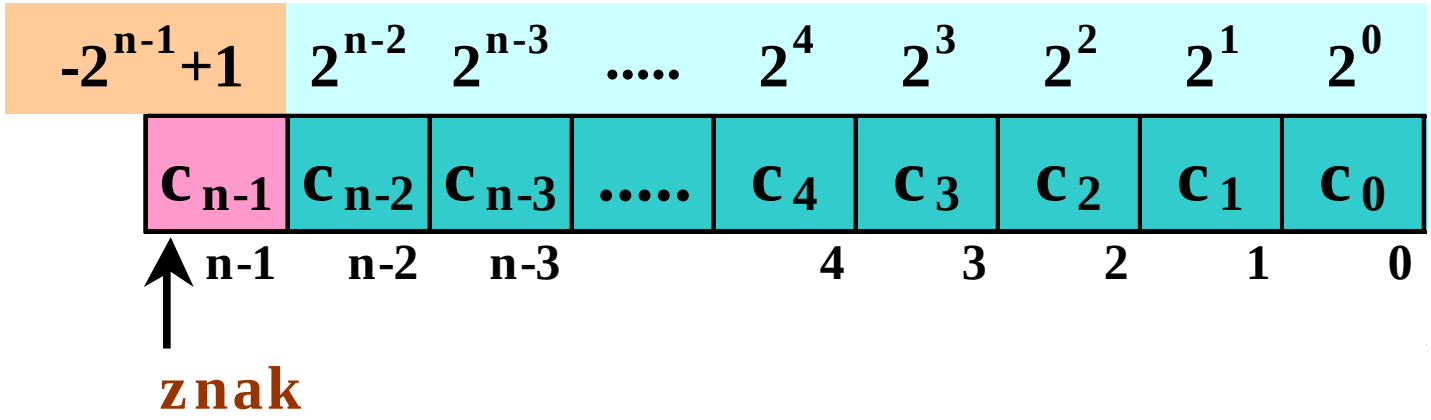
Obliczenie uzupełnienia U2

- Zaczynając od bitu najmniej znaczącego (LSB) przepisać cyfry binarne do napotkania pierwszej 1
- Pozostałe bity zanegować

$$00101100_{(2)} \rightarrow U2$$

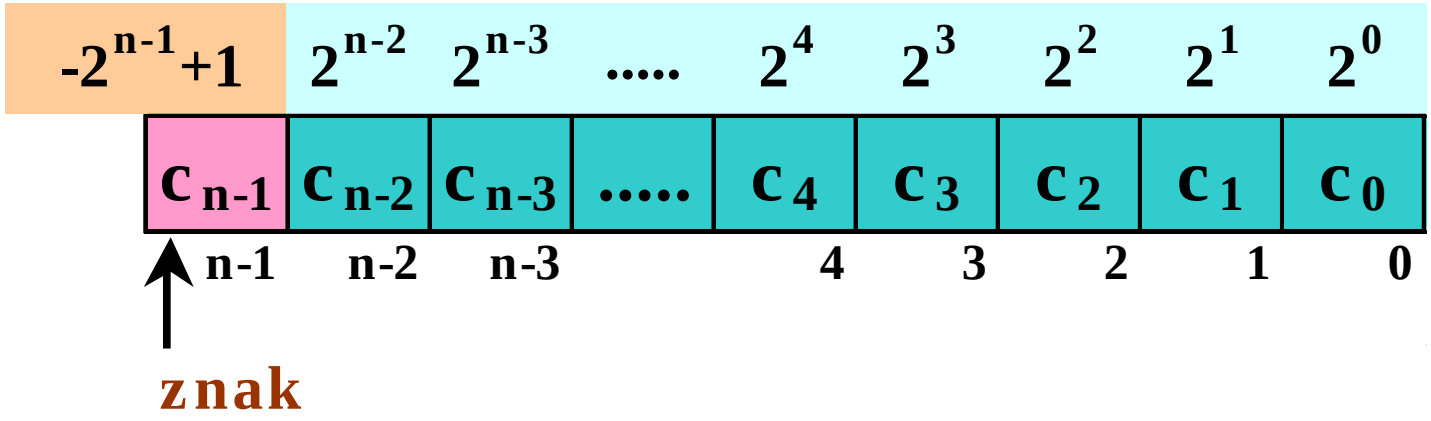


Reprezentacja liczb w kodzie U1



- Najstarszy bit jest bitem znaku: 0 - liczba dodatnia, 1 - liczba ujemna
- W kodzie U1 liczby ***dodatnie*** zapisywane są tak samo jak w NKB, ale najbardziej znaczący bit traktowany jest jako bit znaku
- Liczby ***ujemne*** otrzymywane są poprzez **bitową negację** danej liczby, bit znakowy przyjmuje wtedy wartość 1

Reprezentacja liczb w kodzie U1



- Starszy bit słowa ma wagę **$-2^{n-1}+1$**
- Wartość liczby wynosi:

$$L = c_{n-1}(-2^{n-1}+1) + \sum_{i=0}^{n-2} c_i 2^i$$

Reprezentacja liczb w kodzie U1

Liczba(10)	Kod U1	Liczba(10)	Kod U1
+0	00000000	**-0**	11111111
1	00000001	**-1**	11111110
2	00000010	**-2**	11111101
3	00000011	**-3**	11111100
4	00000100	**-4**	11111011
5	00000101	**-5**	11111010
6	00000110	**-6**	11111001
7	00000111	**-7**	11111000
8	00001000	**-8**	11110111
9	00001001	**-9**	11110110
10	00001010	**-10**	11110101

Reprezentacja liczb w kodzie U1

- Występują dwie reprezentacje zera: +0 (00000000) i
  –0 (11111111)

- Zakres liczb w formacie U1: $\mathbf{-2^{n-1}+1 \leq L \leq 2^{n-1}-1}$

- Zakres liczb *8-bitowych* w kodzie U1: **-127 ...+127**
  $10000000_{(U1)} = -2^7+1 = -127$ (minimalna)
  $01111111_{(U1)} = 2^7-1 = 127$ (maksymalna)

- Zakres liczb *16-bitowych* : **-32 767 ...+32 767**
  $10000000\ 00000000_{(U1)} = -2^{15}+1 = -32\ 767$
  $01111111\ 11111111_{(U1)} = 2^{15}-1 = 32\ 767$

Zamiana 10 → U1

- Znak liczby zakodować w starszym bicie słowa maszynowego
- Moduł liczby przedstawić w kodzie NKB
- Rozszerzyć moduł zerami z lewej strony do formatu słowa maszynowego
- Jeśli liczba jest ujemna zanegować wszystkie bity modułu liczby

$-117_{(10)} = L_{(U1)}$

$|-117_{(10)}| = 117_{(10)} = 1110101_{(NKB)}$

$\mathbf{1}0001010_{(U1)}$ (format 8-bitowy)

$\mathbf{11111111}\ \mathbf{1}0001010_{(U1)}$ (format 16-bitowy)

Zamiana U1 → 10

$$L = c_{n-1}(-2^{n-1}+1) + \sum_{i=0}^{n-2} c_i 2^i$$

$1\,0101010_{(U1)} = L_{(10)}$

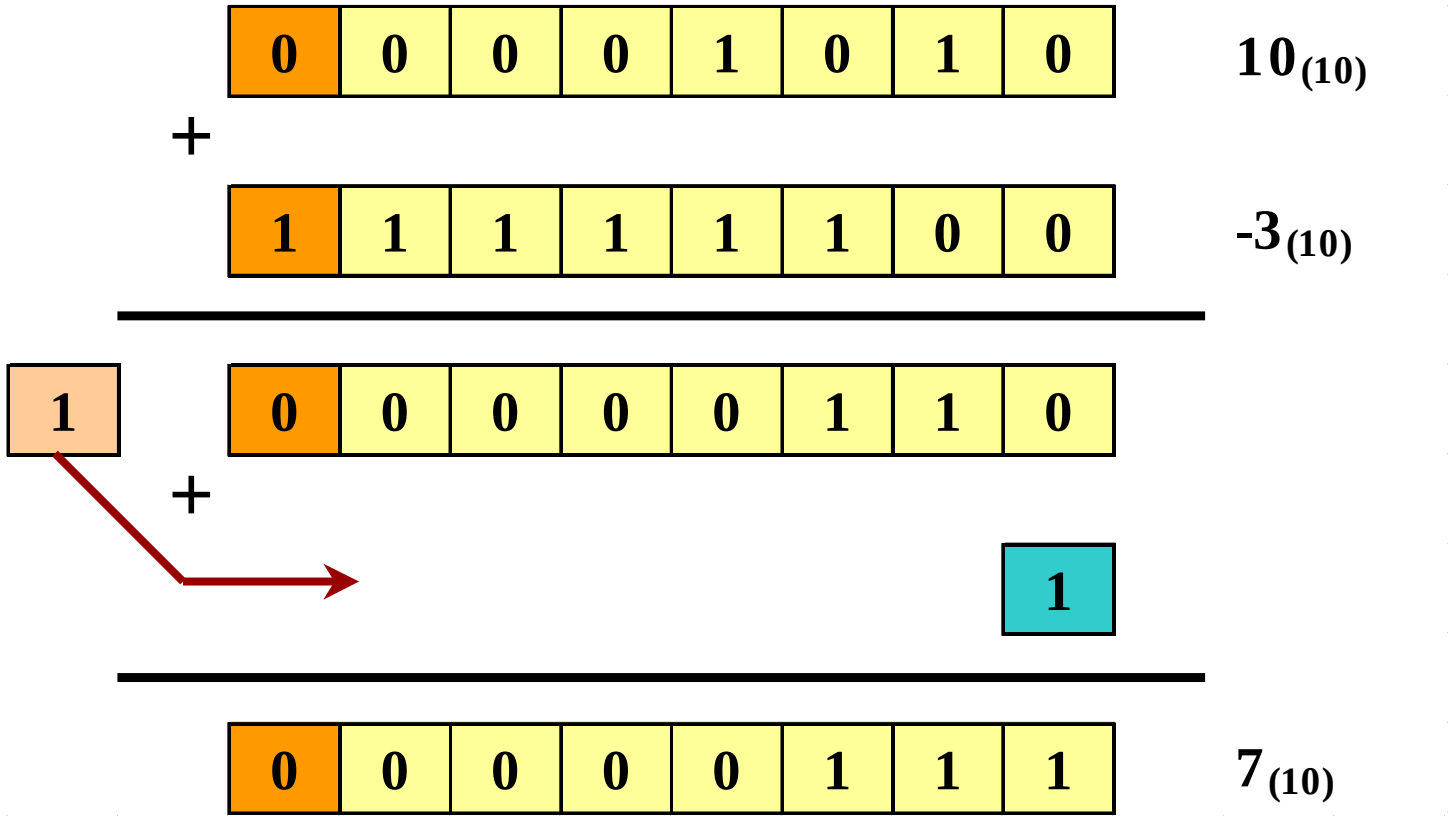
$L_{(10)} = \mathbf{1 \cdot (-2^7+1)} + (0\cdot 2^6 + 1\cdot 2^5 + 0\cdot 2^4 + 1\cdot 2^3 +$
$+ 0\cdot 2^2 + 1\cdot 2^1 + 0\cdot 2^0) =$
$= -127 + (32+8+2) = \mathbf{-85_{(10)}}$

$0\,1001001_{(U1)} = L_{(10)}$

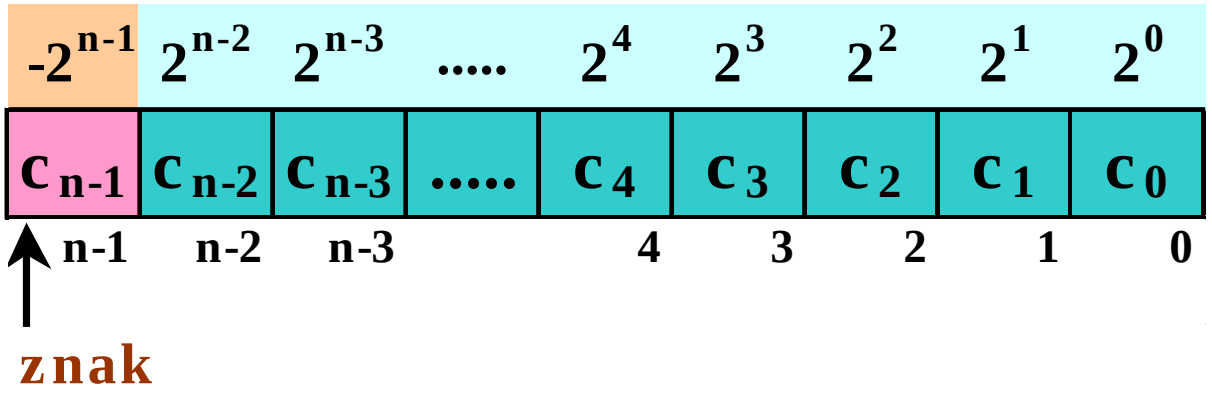
$L_{(10)} = \mathbf{0 \cdot (-2^7+1)} + (1\cdot 2^6 + 0\cdot 2^5 + 0\cdot 2^4 + 1\cdot 2^3 + + 0\cdot 2^2$
$+ 0\cdot 2^1 + 1\cdot 2^0) = 64+8+1 = \mathbf{73_{(10)}}$

Dodawanie liczb w kodzie U1

- Dodawanie w kodzie U1 polega na zwykłym dodawaniu bitowym
- Jeśli na najstarszym bicie wystąpi przeniesienie, to należy je dodać do końcowego wyniku

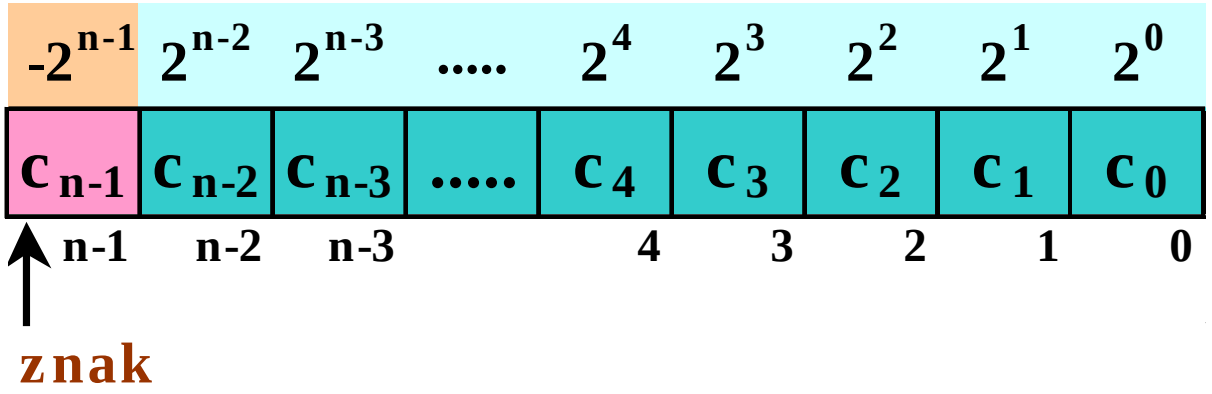


Reprezentacja liczb w kodzie U2



- Najstarszy bit jest bitem znaku: 0 - liczba dodatnia,
  1 - liczba ujemna
- W kodzie U2 liczby ***dodatnie*** zapisywane są tak samo jak w NKB, ale najbardziej znaczący bit traktowany jest jako bit znaku
- Liczby ***ujemne*** otrzymywane są poprzez **bitową negację** danej liczby oraz dodanie do zanegowanej liczby jedynki

Reprezentacja liczb w kodzie U2



- Starszy bit słowa ma wagę -2^{n-1}

- Wartość liczby wynosi:

$$L = c_{n-1}(-2^{n-1}) + \sum_{i=0}^{n-2} c_i 2^i$$

Reprezentacja liczb w kodzie U2

Liczba(10)	Kod U2	Liczba(10)	Kod U2
+0	00000000		
1	00000001	**-1**	11111111
2	00000010	**-2**	11111110
3	00000011	**-3**	11111101
4	00000100	**-4**	11111100
5	00000101	**-5**	11111011
6	00000110	**-6**	11111010
7	00000111	**-7**	11111001
8	00001000	**-8**	11111000
9	00001001	**-9**	11110111
10	00001010	**-10**	11110110

Reprezentacja liczb w kodzie U2

- Występuje jedna reprezentacja zera: 00000000
- Zakres liczb w formacie U2: $-2^{n-1} \leq L \leq 2^{n-1}-1$
- Jest niesymetryczny dla górnej i dolnej granicy
- Nie istnieje liczba przeciwna do najmniejszej -2^{n-1}
- Zakres liczb *8-bitowych* w kodzie U1: **-128 ...+127**

  $10000000_{(U2)} = -2^7 = -128$ (minimalna)

  $01111111_{(U2)} = 2^7-1 = 127$ (maksymalna)

Reprezentacja liczb w kodzie U2

- Zakres liczb 16-bitowych w kodzie U2:

-32 768 ...+32 767

$10000000\ 00000000_{(U2)} = -2^{15} = -32\ 768$ (minimalna)

$01111111\ 11111111_{(U2)} = 2^{15}-1 = 32\ 767$ (maksymalna)

Zamiana 10 → U2

- Moduł liczby przedstawić w kodzie NKB
- Rozszerzyć moduł zerami z lewej strony do formatu słowa maszynowego
- Jeśli liczba jest ujemna zanegować wszystkie bity liczby
- Do wyniku dodać 1

$-117_{(10)} = L_{(U2)}$

$|-117_{(10)}| = 117_{(10)} =$ **01110101** – NKB

10001010 – negacja bitów

+ 1

10001011 – U2

$10001011_{(U2)}$ (format 8-bitowy)

$11111111\ 10001011_{(U2)}$ (format 16-bitowy)

Zamiana U2 → 10

$$L = c_{n-1}(-2^{n-1}) + \sum_{i=0}^{n-2} c_i 2^i$$

$10101010_{(U2)} = L_{(10)}$

$L_{(10)} = 1 \cdot (-2^7) + (0 \cdot 2^6 + 1 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 +$
$+ 1 \cdot 2^1 + 0 \cdot 2^0) = -128 + (32+8+2) = -86_{(10)}$

$01001001_{(U2)} = L_{(10)}$

$L_{(10)} = 0 \cdot (-2^7) + (1 \cdot 2^6 + 0 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 +$
$+ 0 \cdot 2^1 + 1 \cdot 2^0) = 64+8+1 = 73_{(10)}$

Kody z przesunięciem BIAS

- Kod liczby tworzy się przez dodanie do niej pewnej wartości przesunięcia (BIAS)

  Kod=Liczba+BIAS

- Dla przesunięcia równego $2^{n-1}-1$ (gdzie n – liczba bitów w słowie kodowym) zakres kodowanych wartości wynosi:

  od $-2^{n-1}+1$ do 2^{n-1}

- Kod pozwala uniknąć przechowywania znaku liczby

- W formacie *8-bitowym* (BIAS=127) można zapisać wartości :
  od –127 (00000000)
  do +128 (11111111)

Kody z przesunięciem BIAS

Liczba = Kod – BIAS

Dla formatu 4-bitowego (BIAS=7):

-7	0000
-6	0001
-5	0010
...	
-1	0110
0	0111
1	1000
...	
7	1110
8	1111

Monotoniczność zapisu z przesunięciem

DEC	BIAS	U2	ZM
+128	FFh	-	-
+127	FEh	7Fh	7Fh
+126	FDh	7Dh	7Dh
...	...	...	...
+1	80h	01h	01h
0	7Fh	00h	00h,80h
-1	7Eh	FFh	81h
...	...	...	...
-126	01h	82h	FEh
-127	00h	81h	FFh
-128	-	80h	-