PODSTAWY TEORII UKŁADÓW CYFROWYCH

Kod U2

Opracował: Andrzej Nowak

Bibliografia: **Urządzenia techniki komputerowej,** K. Wojtuszkiewicz
http://pl.wikipedia.org/

System zapisu liczb ze znakiem opisany w poprzednim rozdziale nie jest zbyt wygodny dla obliczeń maszynowych.

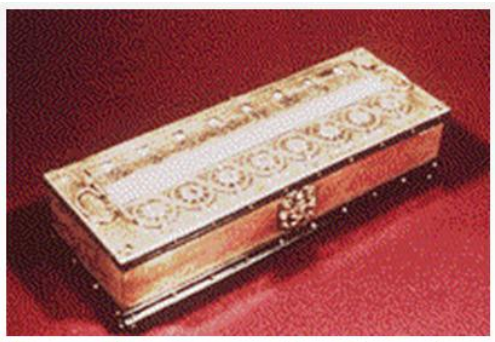
Już przy tak prostych operacjach jak dodawanie i odejmowanie musimy stosować dodatkową logikę obsługi znaków, aby otrzymywać poprawne wyniki.

Dlatego w obliczeniach komputerowych bardziej popularny jest inny system, zwany **systemem uzupełnień do dwóch** (w skrócie U2) lub uzupełnień do podstawy (w literaturze angielskiej nosi on nazwę Two's Complement Numbering System).

Idea systemu U2 nie jest nowa - wymyślił ją już Blaise Pascal, znany francuski fizyk i matematyk, który skonstruował w 1652 roku prostą maszynę arytmetyczną zdolną dodawać liczby dziesiętne.

Aby umożliwić również odejmowanie liczb, Pascal wprowadził tzw. uzupełnienie do podstawy 10.

Otrzymujemy je odejmując daną liczbę od podstawy podniesionej do potęgi równej największej ilości cyfr dodawanych liczb. Np. dla liczb z zakresu od 0..99 będziemy odejmować od $100 - 10^2$, dla liczb z zakresu od 0..999 od $1000 - 10^3$ itd.



Pascalina - sumator Pascala

Chcemy wykonać odejmowanie: 56 - 27.

W tym celu obliczamy uzupełnienie do podstawy 10 liczby 27:

$$- 27 = 100 - 27 = 73_{(U10)}$$

Teraz wykonujemy dodawanie liczby 56 oraz uzupełnienia $73_{(U10)}$:

$$56 + 73_{(U10)} = 129$$

Odrzucamy najstarszą jedynkę i mamy wynik 29. Zgadza się? /Pascal też był z tego zadowolony./ Na identycznej zasadzie utworzono system uzupełnień do podstawy U2. W systemie tym waga pozycji najstarszego bitu jest ujemna. Jeśli bit na tej pozycji będzie miał wartość 0, to otrzymamy liczbę dodatnią,

której wartość określają pozostałe bity. Gdy bit znaku przyjmie wartość 1, to liczba będzie ujemna. Wartość liczby obliczymy jako sumę wagi pozycji najstarszego bitu (jest ujemna) oraz pozostałej części liczby.

Wzór obliczeniowy jest następujący:

$$W_{U2} = c_{n-1} \times (- p^{n-1}) + c_{n-2} \times p^{n-2} + \ldots + c_1 \times p^1 + c_0 \times p^0$$

Obliczmy dla przykładu wartość liczb 4 bitowych w kodzie U2:

$0101_{(U2)} = 0 \times (- 2^3) + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$
$0101_{(U2)} = 0 \times (- 8) + 1 \times 4 + 0 \times 2 + 1 \times 1$
$0101_{(U2)} = 0 + 4 + 1$
$0101_{(U2)} = \mathbf{5}$

$1101_{(U2)} = 1 \times (- 2^3) + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$
$1101_{(U2)} = 1 \times (- 8) + 1 \times 4 + 0 \times 2 + 1 \times 1$
$1101_{(U2)} = (- 8) + 4 + 1$
$1101_{(U2)} = - 8 + 5$
$1101_{(U2)} = \mathbf{- 3}$

Zapamiętaj

Wartość liczby w kodzie U2 obliczamy bardzo podobnie do wartości liczby w naturalnym kodzie dwójkowym. Musimy tylko pamiętać, że waga najstarszej pozycji (pozycji znakowej) jest ujemna. Dla n bitowej liczby U2 możemy zapisać wzór:

$$W_{U2} = c_{n-1} \times (- 2^{n-1}) + \text{wartość reszty liczby w kodzie NBC}$$

Zakres liczb w kodzie U2

Największą liczbę w kodzie U2 otrzymamy, gdy bit znaku przyjmie wartość 0 (liczba dodatnia), a reszta cyfr będzie składała się z samych jedynek.

Reszta cyfr tworzy największą liczbę n-1 bitową w naturalnym kodzie binarnym, wobec tego:

$$Z_{U2max} = 2^{n-1} - 1$$

Najmniejszą liczbę w kodzie U2 otrzymamy dla bitu znaku równego 1 (liczba ujemna) oraz pozostałych bitów równych zero.

$$Z_{U2min} = -2^{n-1}$$

zakres n-bitowej liczby w kodzie U2

$$Z_{U2} = \langle -2^{n-1}, 2^{n-1} - 1 \rangle$$

Przykład

Obliczmy zakres 4 bitowych liczb w kodzie U2:

$Z_{U2max} = 2^{4-1} - 1 = 2^3 - 1 = 8 - 1 = 7$

$Z_{U2min} = -2^{4-1} = -2^3 = -8$

Dla czterech bitów $Z_{U2} =$ **<-8, 7>**

Obliczanie wartości przeciwnej w kodzie U2

Wartość przeciwna ma tą samą wartość bezwzględną, lecz znak przeciwny.

W kodzie U2 obliczamy ją następująco:

ZAPAMIĘTAJ

Aby znaleźć wartość przeciwną do danej w kodzie U2, wykonaj następujące dwie operacje:

- zmień wszystkie bity liczby na przeciwne (możesz wykorzystać do tego celu operację logiczną NOT).
- do tak uzyskanej liczby dodaj 1

Przykład

Oblicz wartość przeciwną do liczby $0011_{(U2)} = 3$:

NOT	0011
	1100
+	0001
	1101

Sprawdzenie

$1101_{(U2)} = 1 \times (-2^3) + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$

$1101_{(U2)} = 1 \times (-8) + 1 \times 4 + 0 \times 2 + 1 \times 1$

$1101_{(U2)} = -8 + 4 + 1$

$1101_{(U2)} = -3$

Znajdowanie reprezentacji liczby w kodzie U2

Mamy wartość dziesiętną W i chcemy zapisać ją w n-bitowym kodzie U2.

Najpierw musimy sprawdzić, czy zakres liczb U2 obejmuje wartość W. Jeśli tak, to:

- Dla W nieujemnego obliczamy liczbę dwójkową o tej wartości, a następnie dopełniamy ją zerami do formatu n-bitowego.
- Dla W ujemnego obliczamy uzupełnienie do podstawy 2 o wartości 2^n + W. Wynik kodujemy binarnie i otrzymujemy liczbę ujemną w kodzie U2.

Przykład

Znaleźć zapis liczby 92 w 8 bitowym kodzie U2.

Liczba jest dodatnia, więc znajdujemy jej zapis binarny:

$$92 = 01011100_{(U2}$$

92 : 2 = 46 i reszta $c_0 = 0$
46 : 2 = 23i reszta $c_1 = 0$
23 : 2 = 11i reszta $c_2 = 1$
11 : 2 = 5i reszta $c_3 = 1$
5 : 2 = 2i reszta $c_4 = 1$
2 : 2 = 1i reszta $c_5 = 0$
1 : 2 = 0i reszta $c_6 = 1$

Przykład

Znaleźć zapis liczby -107 w 8 bitowym kodzie U2:

Liczba jest ujemna, więc najpierw obliczamy

jej dopełnienie do podstawy:

$U = 2^n + W = 2^8 - 107 = 256 - 107 = 149$

$$-107 = 10010101_{(U2)}$$

Wyrażamy 149 binarnie
149 : 2 = 74 i reszta $c_0 = 1$
74 : 2 = 37 i reszta $c_1 = 0$
37 : 2 = 18 i reszta $c_2 = 1$
18 : 2 = 9 i reszta $c_3 = 0$
9 : 2 = 4 i reszta $c_4 = 1$
4 : 2 = 2 i reszta $c_5 = 0$
2 : 2 = 1 i reszta $c_6 = 0$
1 : 2 = 0 i reszta $c_7 = 1$

Arytmetyka liczb w kodzie U2

Dodawanie i odejmowanie

Zasady dodawania i odejmowania liczb w kodzie U2 nie różnią się od zasad wykonywania tych działań arytmetycznych w zwykłym systemie binarnym, co jest niewątpliwą zaletą kodu U2. Otrzymaliśmy poprawne wyniki operacji bez sprawdzania znaków przetwarzanych liczb. Jest to bardzo dużą zaletą kodu U2, dzięki której obliczenia są szybkie i sprawne.

5 + (-3)	2-(-3)
0101	0010
+ 1101	- 1101
1 0010	1 0101
Wynik 2	Wynik 5

Zapamiętaj
W systemie U2 dodawanie i odejmowanie wykonujemy wg poznanych zasad dla naturalnego kodu dwójkowego. Przeniesienia i pożyczki poza bit znaku ignorujemy.

Mnożenie

Mnożenie liczb w kodzie U2 różni się nieco od standardowego mnożenia liczb binarnych.

Przed wykonaniem tej operacji arytmetycznej musimy rozszerzyć znakowo obie mnożone liczby tak, aby ich długość (liczba bitów) wzrosła dwukrotnie (jeśli są różnej długości, to rozszerzamy znakowo względem dłuższej liczby).

Rozszerzenie znakowe polega na powielaniu bitu znaku na wszystkie dodane bity. Np.:

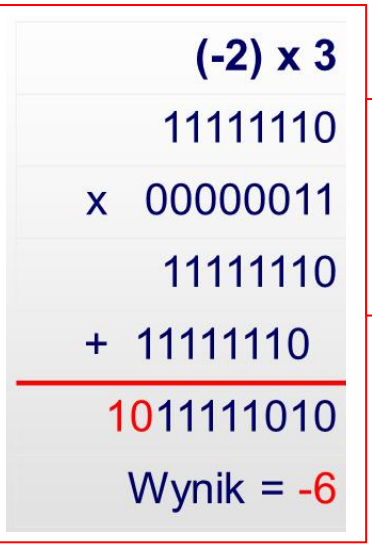
$0111_{(U2)} = 0000\ 0111_{(U2)}$ - rozszerzyliśmy znakowo liczbę 4 bitową do 8 bitowej
$1011_{(U2)} = 1111\ 1011_{(U2)}$ - to samo dla liczby ujemnej.

Rozszerzenie znakowe nie zmienia wartości liczby w kodzie U2.

Po wykonaniu rozszerzenia znakowego liczby mnożymy wg poznanych już zasad.

Dla przykładu pomnóżmy (-2) x 3:

$$-2 = 1110_{(U2)} = 1111\ 1110_{(U2)}$$
$$3 = 0011_{(U2)} = 0000\ 0011_{(U2)}$$



Wynik mnożenia może być liczbą o długości równej sumie długości mnożonych liczb.
Dlatego bity wykraczające w naszym przykładzie poza 8 bitów ignorujemy.
Pozostałe 8 bitów określa w kodzie U2 liczbę -6.

Dzielenie

Najprostszym rozwiązaniem jest zapamiętanie znaków dzielonych liczb, zamiana ich na liczby dodatnie, dokonanie dzielenia dla liczb naturalnych, a następnie zmiana znaku wyniku, jeśli znaki dzielnej i dzielnika różnią się.