* + 1. **Mnożenie binarne**

[https://www.youtube.com/watch?v=BCwPAdXQx-s](about:blank)

Mnożenie cyfr binarnych *ai \* bi* nie prowadzi do powstania przeniesienia na pozycję bardziej znaczącą. Stąd, reguły mnożenia mają następującą postać:

Mnożna *a*i 1 1 0 0

Mnożnik *b*i 0 1 0 1

----------------------------------------

Iloczyn *m*i 0 1 0 0

W przypadku mnożenia liczb w układzie binarnym zachodzi konieczność sumowania iloczynów cząstkowych. W wyniku mnożenia może pojawić się iloczyn zajmujący tyle bitów ile wynosi suma bitów zajmowanych przez mnożną i mnożnik. Jeśli mnożone liczby są ułamkami, to podczas mnożenia można pominąć przecinki w mnożnej i mnożniku, a następnie wstawić przecinek w otrzymanym iloczynie. Liczba bitów po przecinku w iloczynie jest równa łącznej liczbie bitów po przecinku w mnożnej i mnożniku.

**Przykład.** Pomnożyć pisemnie liczby binarne *A* = 111.01 = 7.25 i *B* = 110.1 = 6.5.

11101

\*1101

-------

11101

00000

11101

+11101

-----------

101111001

Uwzględniając trzy miejsca po przecinku uzyskuje się *M* = *A* \* *B* = 111.01 \* 110.1 =

101111.001 = 47.125. Sprawdzenie: 7.25 \* 6.5 = 47.125.

**Przykład.** Pomnożyć pisemnie liczby binarne *A* = 111.1 = 7.5 i *B* = 111.1 = 7.5.

1111

\*1111

-------

23321 przeniesienie

----------

1111

1111

1111

+1111

---------------

11100001

Uwzględniając dwa miejsca po przecinku uzyskuje się *M* = *A* \* *B* = 111.1 \* 111.1 =

111000.01 = 56.25. Sprawdzenie: 7.5 \* 7.5 = 56.25.

* + 1. **. Dzielenie binarne**

[https://www.youtube.com/watch?v=59SLPMkHnZg](about:blank)

**Przykład. 3.9.** Podzielić pisemnie liczby binarne *A* = 11101.01 = 29.25 i *B* = 10.1 = 2.5.

Przesunięcie przecinka A = 111010.1 oraz B = 101.

1011.

----------------

111010.1 : 101

-101

----------

1001

-101

----------

1000

-101

-------------

11.1 reszta

Można kontynuować dzielenie i w rezultacie otrzymać część ułamkową.

1011.1011001100 ...

----------------

111010.1 : 101

-101

----------

1001

-101

----------

1000

-101

-------------

11 1

- 10 1

-----------

1 000

- 101

-----------

0110

-101

-----------

1000

-101

--------------

...

W wyniku dzielenia otrzymano iloraz *E* = *A* : *B* = 11101.01 : 10.1 = 1011 reszta 11.1/101 =

11 reszta 3.5/5. Sprawdzenie: 29.25 : 2.5 = 11.7 = 1011.10 (1100) ... .

**Realizacja operacji arytmetycznych w systemach cyfrowych**

W systemach cyfrowych operacje arytmetyczne są wykonywane na liczbach o określonej liczbie bitów. W wyniku realizacji działań arytmetycznych pojawia się wynik, który jest zapamiętywany na określonej liczbie bitów. Na przykład w przypadku dodawania liczb n- bitowych często przyjmuje się, że wynik również zajmuje *n* bitów. Można zauważyć, że wykonanie określonej operacji arytmetycznej na liczbach o określonej liczbie bitów może dać wynik, który wykracza poza zakres wartości liczbowych reprezentowanych w danym kodzie na danej liczbie bitów. Wówczas powstaje nadmiar (ang. *overflow*), który musi być zasygnalizowany użytkownikowi układu cyfrowego. Powstawanie nadmiaru podczas dodawania liczb dodatnich w kodzie NKB ilustruje następujący przykład.

**Przykład.** Dodać dwie liczby 4-bitowe w naturalnym kodzie binarnym (NKB). Wynik sumowania zapamiętać na 4 bitach. Liczba *A* = 1100 = 12. Liczba *B* = 1010 = 10.

1100

+1010

------

C = 1 0110 C - przeniesienie

*S* = *A* + *B* = 1100 + 1010 = 10110 = 22. Sprawdzenie 12 + 10 = 22.

W wyniku operacji sumowania pojawia się przeniesienie *c*=1 na pozycję numer cztery. Oznacza to, że zakres wyniku został przekroczony. Rozwiązaniem tego problemu jest wyposażenie układu cyfrowego w sygnalizację przeniesienia lub zwiększenie zakresu wyniku o jeden bit.

W praktyce cyfrowe układy arytmetyczne wykonujące podstawowe działania są wyposażone w sygnał wyjściowy sygnalizujący nadmiar (np. jednostka arytmetyczno-logiczna procesora). Układy te sygnalizują także inne cechy wyniku jak: znak, wynik jest równy zero, wynik zawiera parzystą liczbę jedynek, itp. Bity te są nazywane w układach cyfrowych *znacznikami* (flagami, ang. flags).

2.3. Działania w systemach nie binarnych (o podstawie od 3 do 9)

2.4. Zadania – plik zadania 3.docx.