Київський національний університет імені Тараса Шевченка
Факультет радіофізики, електроніки та компютерних систем

Лабораторна робота №2
Арифметичні операції над двійковими числами

Виконав студент
3 курсу СА-КІ
Глушко Гліб

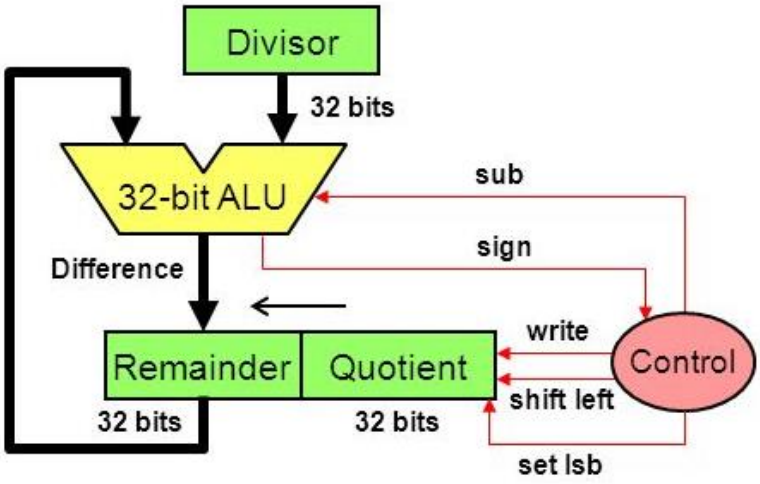
Алгоритм Бута

00 – NOP
10 – SUB
11 – NOP
01 – ADD

```
Enter 16 bits signed multiplicand:
4
Enter 16 bits signed multiplier:
9
Booth's algorithm:
  Step 1:
        Add S:   1111 1111 1111 1100 0000 0000 0000 0000 0
        To P:    0000 0000 0000 0000 0000 0000 0000 1001 0
        Shift right:     1111 1111 1111 1100 0000 0000 0000 1001 0
                         1111 1111 1111 1110 0000 0000 0000 0100 1
  Step 2:
        Add A:   0000 0000 0000 0100 0000 0000 0000 0000 0
        To P:    1111 1111 1111 1110 0000 0000 0000 0100 1
        Shift right:     0000 0000 0000 0010 0000 0000 0000 0100 1
                         0000 0000 0000 0001 0000 0000 0000 0010 0
  Step 3:
        Shift right:     0000 0000 0000 0001 0000 0000 0000 0010 0
                         0000 0000 0000 0000 1000 0000 0000 0001 0
  Step 4:
        Add S:   1111 1111 1111 1100 0000 0000 0000 0000 0
        To P:    0000 0000 0000 0000 1000 0000 0000 0001 0
        Shift right:     1111 1111 1111 1100 1000 0000 0000 0001 0
                         1111 1111 1111 1110 0100 0000 0000 0000 1
  Step 5:
        Add A:   0000 0000 0000 0100 0000 0000 0000 0000 0
        To P:    1111 1111 1111 1110 0100 0000 0000 0000 1
        Shift right:     0000 0000 0000 0010 0100 0000 0000 0000 1
                         0000 0000 0000 0001 0010 0000 0000 0000 0
  Step 6:
        Shift right:     0000 0000 0000 0001 0010 0000 0000 0000 0
                         0000 0000 0000 0000 1001 0000 0000 0000 0
  Step 7:
        Shift right:     0000 0000 0000 0000 1001 0000 0000 0000 0
                         0000 0000 0000 0000 0100 1000 0000 0000 0
  Step 8:
        Shift right:     0000 0000 0000 0000 0100 1000 0000 0000 0
                         0000 0000 0000 0000 0010 0100 0000 0000 0
  Step 9:
        Shift right:     0000 0000 0000 0000 0010 0100 0000 0000 0
                         0000 0000 0000 0000 0001 0010 0000 0000 0
  Step 10:
        Shift right:     0000 0000 0000 0000 0001 0010 0000 0000 0
                         0000 0000 0000 0000 0000 1001 0000 0000 0
  Step 11:
        Shift right:     0000 0000 0000 0000 0000 1001 0000 0000 0
                         0000 0000 0000 0000 0000 0100 1000 0000 0
  Step 12:
        Shift right:     0000 0000 0000 0000 0000 0100 1000 0000 0
                         0000 0000 0000 0000 0000 0010 0100 0000 0
  Step 13:
        Shift right:     0000 0000 0000 0000 0000 0010 0100 0000 0
                         0000 0000 0000 0000 0000 0001 0010 0000 0
  Step 14:
        Shift right:     0000 0000 0000 0000 0000 0001 0010 0000 0
                         0000 0000 0000 0000 0000 0000 1001 0000 0
  Step 15:
        Shift right:     0000 0000 0000 0000 0000 0000 1001 0000 0
                         0000 0000 0000 0000 0000 0000 0100 1000 0
  Step 16:
        Shift right:     0000 0000 0000 0000 0000 0000 0100 1000 0
                         0000 0000 0000 0000 0000 0000 0010 0100 0
 Answer is:
       In decemal: 36
       In binary:  0000 0000 0000 0000 0000 0000 0010 0100
```

```
Enter 16 bits signed multiplicand:
1488
Enter 16 bits signed multiplier:
228
Booth's algorithm:
  Step 1:
        Shift right:     0000 0000 0000 0000 0000 0000 1110 0100 0
                         0000 0000 0000 0000 0000 0000 0111 0010 0
  Step 2:
        Shift right:     0000 0000 0000 0000 0000 0000 0111 0010 0
                         0000 0000 0000 0000 0000 0000 0011 1001 0
  Step 3:
        Add S:   1111 1010 0011 0000 0000 0000 0000 0000 0
        To P:    0000 0000 0000 0000 0000 0000 0011 1001 0
        Shift right:     1111 1010 0011 0000 0000 0000 0011 1001 0
                         1111 1101 0001 1000 0000 0000 0001 1100 1
  Step 4:
        Add A:   0000 0101 1101 0000 0000 0000 0000 0000 0
        To P:    1111 1101 0001 1000 0000 0000 0001 1100 1
        Shift right:     0000 0010 1110 1000 0000 0000 0001 1100 1
                         0000 0001 0111 0100 0000 0000 0000 1110 0
  Step 5:
        Shift right:     0000 0001 0111 0100 0000 0000 0000 1110 0
                         0000 0000 1011 1010 0000 0000 0000 0111 0
  Step 6:
        Add S:   1111 1010 0011 0000 0000 0000 0000 0000 0
        To P:    0000 0000 1011 1010 0000 0000 0000 0111 0
        Shift right:     1111 1010 1110 1010 0000 0000 0000 0111 0
                         1111 1101 0111 0101 0000 0000 0000 0011 1
  Step 7:
        Shift right:     1111 1101 0111 0101 0000 0000 0000 0011 1
                         1111 1110 1011 1010 1000 0000 0000 0001 1
  Step 8:
        Shift right:     1111 1110 1011 1010 1000 0000 0000 0001 1
                         1111 1111 0101 1101 0100 0000 0000 0000 1
  Step 9:
        Add A:   0000 0101 1101 0000 0000 0000 0000 0000 0
        To P:    1111 1111 0101 1101 0100 0000 0000 0000 1
        Shift right:     0000 0101 0010 1101 0100 0000 0000 0000 1
                         0000 0010 1001 0110 1010 0000 0000 0000 0
  Step 10:
        Shift right:     0000 0010 1001 0110 1010 0000 0000 0000 0
                         0000 0001 0100 1011 0101 0000 0000 0000 0
  Step 11:
        Shift right:     0000 0001 0100 1011 0101 0000 0000 0000 0
                         0000 0000 1010 0101 1010 1000 0000 0000 0
  Step 12:
        Shift right:     0000 0000 1010 0101 1010 1000 0000 0000 0
                         0000 0000 0101 0010 1101 0100 0000 0000 0
  Step 13:
        Shift right:     0000 0000 0101 0010 1101 0100 0000 0000 0
                         0000 0000 0010 1001 0110 1010 0000 0000 0
  Step 14:
        Shift right:     0000 0000 0010 1001 0110 1010 0000 0000 0
                         0000 0000 0001 0100 1011 0101 0000 0000 0
  Step 15:
        Shift right:     0000 0000 0001 0100 1011 0101 0000 0000 0
                         0000 0000 0000 1010 0101 1010 1000 0000 0
  Step 16:
        Shift right:     0000 0000 0000 1010 0101 1010 1000 0000 0
                         0000 0000 0000 0101 0010 1101 0100 0000 0
  Answer is:
        In decemal: 339264
```

Частка та залишок в одному регістрі




```
Enter 16 bits signed dividend:
1488
Enter 16 bits signed divisor:
228
```

```
        Answer is:
                Remainder:          0 0000 0000 0111 1000  (in decimal: 120)
                Quotient:             0000 0000 0000 0110 (in decimal: 6)
```

```
Enter 16 bits signed dividend:
18
Enter 16 bits signed divisor:
5
        Register:
                        0 0000 0000 0000 0000 0000 0000 0001 0010
        Shift left:
                    0 0000 0000 0000 0000 0000 0000 0010 0100
Substract divisor: 1 1111 1111 1111 1011
        Register:
                    1 1111 1111 1111 1011 0000 0000 0010 0100
        Set last quotient bit to 0:
                    1 1111 1111 1111 1011 0000 0000 0010 0100
        Shift left:
                    1 1111 1111 1111 0110 0000 0000 0100 1000
      Add divisor: 0 0000 0000 0000 0101
        Register:
                    1 1111 1111 1111 1011 0000 0000 0100 1000
        Set last quotient bit to 0:
                    1 1111 1111 1111 1011 0000 0000 0100 1000
        Shift left:
                    1 1111 1111 1111 0110 0000 0000 1001 0000
      Add divisor: 0 0000 0000 0000 0101
        Register:
                    1 1111 1111 1111 1011 0000 0000 1001 0000
        Set last quotient bit to 0:
                    1 1111 1111 1111 1011 0000 0000 1001 0000
        Shift left:
                    1 1111 1111 1111 0110 0000 0001 0010 0000
      Add divisor: 0 0000 0000 0000 0101
        Register:
                    1 1111 1111 1111 1011 0000 0001 0010 0000
        Set last quotient bit to 0:
                    1 1111 1111 1111 1011 0000 0001 0010 0000
        Shift left:
                    1 1111 1111 1111 0110 0000 0010 0100 0000
      Add divisor: 0 0000 0000 0000 0101
        Register:
                    1 1111 1111 1111 1011 0000 0010 0100 0000
        Set last quotient bit to 0:
                    1 1111 1111 1111 1011 0000 0010 0100 0000
        Shift left:
                    1 1111 1111 1111 0110 0000 0100 1000 0000
      Add divisor: 0 0000 0000 0000 0101
        Register:
                    1 1111 1111 1111 1011 0000 0100 1000 0000
        Set last quotient bit to 0:
                    1 1111 1111 1111 1011 0000 0100 1000 0000
        Shift left:
                    1 1111 1111 1111 0110 0000 1001 0000 0000
      Add divisor: 0 0000 0000 0000 0101
        Register:
                    1 1111 1111 1111 1011 0000 1001 0000 0000
        Set last quotient bit to 0:
                    1 1111 1111 1111 1011 0000 1001 0000 0000
```

```
        Set last quotient bit to 0:
                    1 1111 1111 1111 1011 0010 0100 0000 0000
        Shift left:
                    1 1111 1111 1111 0110 0100 1000 0000 0000
      Add divisor: 0 0000 0000 0000 0101
        Register:
                    1 1111 1111 1111 1011 0100 1000 0000 0000
        Set last quotient bit to 0:
                    1 1111 1111 1111 1011 0100 1000 0000 0000
        Shift left:
                    1 1111 1111 1111 0110 1001 0000 0000 0000
      Add divisor: 0 0000 0000 0000 0101
        Register:
                    1 1111 1111 1111 1011 1001 0000 0000 0000
        Set last quotient bit to 0:
                    1 1111 1111 1111 1011 1001 0000 0000 0000
        Shift left:
                    1 1111 1111 1111 0111 0010 0000 0000 0000
      Add divisor: 0 0000 0000 0000 0101
        Register:
                    1 1111 1111 1111 1100 0010 0000 0000 0000
        Set last quotient bit to 0:
                    1 1111 1111 1111 1100 0010 0000 0000 0000
        Shift left:
                    1 1111 1111 1111 1000 0100 0000 0000 0000
      Add divisor: 0 0000 0000 0000 0101
        Register:
                    1 1111 1111 1111 1101 0100 0000 0000 0000
        Set last quotient bit to 0:
                    1 1111 1111 1111 1101 0100 0000 0000 0000
        Shift left:
                    1 1111 1111 1111 1010 1000 0000 0000 0000
      Add divisor: 0 0000 0000 0000 0101
        Register:
                    1 1111 1111 1111 1111 1000 0000 0000 0000
        Set last quotient bit to 0:
                    1 1111 1111 1111 1111 1000 0000 0000 0000
        Shift left:
                    1 1111 1111 1111 1111 0000 0000 0000 0000
      Add divisor: 0 0000 0000 0000 0101
        Register:
                    0 0000 0000 0000 0100 0000 0000 0000 0000
        Set last quotient bit to 1:
                    0 0000 0000 0000 0100 0000 0000 0000 0001
        Shift left:
                    0 0000 0000 0000 1000 0000 0000 0000 0010
Substract divisor: 1 1111 1111 1111 1011
        Register:
                    0 0000 0000 0000 0011 0000 0000 0000 0010
        Set last quotient bit to 1:
                    0 0000 0000 0000 0011 0000 0000 0000 0011
        Answer is:
                Remainder:          0 0000 0000 0000 0011  (in decimal: 3)
                Quotient:             0000 0000 0000 0011 (in decimal: 3)
```

Робота з IEEE 754 Floating Point (Представити лише ключові кроки при виконанні операцій)

Додавання

i. Align binary points
ii. Add significands
iii. Normalize result

```
Enter first float signed value:
6,7
Enter second float signed value:
8,2
Adding 8,2 (a), to 6,7 (b)
  Convert "a" to binary (without exponent and normalization):
        0 | 00000000 | 00110011001100110011000
  Convert "b" to binary (without exponent and normalization):
        0 | 00000000 | 10110011001100110011000
  Normalize "a":
        0 | 10000010 | 00000110011001100110011
  Normalize "b" :
        0 | 10000001 | 10101100110011001100110
  Shift left "b" on 1:
        0 | 10000001 | 11010110011001100110011
  Adding "a" to "b":
        0 | 10000010 | 00000110011001100110011
      + 0 | 10000001 | 11010110011001100110011
  Answer is:
        In decemal: 14,9
        In binary: 0 | 10000010 | 11011100110011001100110
```

```
Enter first float signed value:
14,88
Enter second float signed value:
22,8
Adding 22,8 (a), to 14,88 (b)
  Convert "a" to binary (without exponent and normalization):
        0 | 00000000 | 11001100110011001100000
  Convert "b" to binary (without exponent and normalization):
        0 | 00000000 | 11100001010001111011000
  Normalize "a":
        0 | 10000011 | 01101100110011001100110
  Normalize "b" :
        0 | 10000010 | 11011100001010001111011
  Shift left "b" on 1:
        0 | 10000010 | 11101110000101000111101
  Adding "a" to "b":
        0 | 10000011 | 01101100110011001100110
      + 0 | 10000010 | 11101110000101000111101
  Answer is:
        In decemal: 37,68
        In binary: 0 | 10000100 | 00101101011100001010001
```

Висновок: Під час лабораторної работи, було досліджено і реалізовано алгортим Бута, ділення, частка і залишок в одному регістрі та додавання дробових чисел.

https://github.com/GlebGlushko/CompSys.git