**Київський національний університет імені Тараса Шевченка**

**факультет радіофізики, електроніки та комп’ютерних систем**

Звіт з предмету «Комп'ютерні системи»  
Лабораторна робота №2

Роботу виконав

студент 3 курсу

Комп’ютерної Інженерії

Стансков Володимир Володимирович

Київ 2023р

**Мета роботи:** Дослідити алгоритми, що використовуються в мікропроцесорах для множення та ділення цілих чисел та підходи до роботи з дійсними числами

**Хід роботи:**

Варіант c, a, b

1. Множення двійкових чисел (Множник в правій частині регістру)

Код функції, що виконує множення двійкових чисел (за двійкові числа приймається массив булевих змінних):

function multiply(a, b) {

  if (a.length != b.length) {

    return [false];

  }

  let result = new Array(a.length).fill(false).concat(b);

  for (let i = 0; i < b.length; i++) {

    if (result[result.length - 1]) {

      result = add(result, a.concat(new Array(b.length).fill(false)))

    }

  result.pop();

  result.unshift(false);

  }

  return result;

}

Код допоміжної функції add, що додає два двійкових числа:

function add(a, b) {

  if (a.length != b.length) {

    return [false];

  }

  let carry = false;

  const result = Array(a.length).fill(false);

  for (let i = a.length - 1; i >= 0; i--) {

    if (!xor(a[i], b[i])) {

      if (a[i]) {

        result[i] = carry;

        carry = true;

      }

      else{

        result[i] = carry;

          carry = false;

      }

    }

    else{

      if (carry) {

        result[i] = false;

      }

      else{

        result[i] = true;

      }

    }

  }

  if (carry) {

    result.unshift(true);

  }

  return result;

}

Фрагмент коду, що добоможе показати роботу програми:

let aInt = 11;

let bInt = 10;

let aBinary = intToBinaryArray(aInt, 4);

let bBinary = intToBinaryArray(bInt, 4);

let productBinary = multiply(aBinary, bBinary);

let productInt = binaryArrayToInt(productBinary);

console.log(`${aInt} \* ${bInt} = ${productInt}`);

console.log(`${binaryArrayToString(aBinary)} \* ${binaryArrayToString(bBinary)} = ${binaryArrayToString(productBinary)}`);

Вивід консолі при запуску програми:

PS D:\labs\cs\Lab2> node .\Task1.js

11 \* 10 = 110

1011 \* 1010 = 01101110

2. Ділення двійкових чисел (Ділення як є (в стовпчик))

Код функції, що виконує ділення двійкових:

function div(a, b) {

  let quotient = [];

  let remainder = [...a];

  let i = 0;

  let test = [...b];

  while(test[test.length - 1] == false){

    test.pop();

  }

  while(isGreaterOrEqual(remainder,new Array(remainder.length - test.length).fill(false).concat(test))){

    let subtr;

    if (i===0) {

      subtr = b.concat(new Array(remainder.length - b.length).fill(false))

    }

    subtr = new Array(i).fill(false).concat(b.concat(new Array(remainder.length - b.length - i).fill(false)));

    if (isGreaterOrEqual(remainder, subtr)) {

      quotient.push(true);

      remainder = sub(remainder, subtr);

    }

    else{

      quotient.push(false);

    }

    i++;

  }

  if (i%2==0) {

    quotient.push(false);

  }

  return [quotient, remainder];

}

Код допоміжної функції sub, що знаходить різницю двох двійкових чисел:

function sub(a, b) {

  if (!isGreaterOrEqual(a,b)) {

    return[false]

  }

  let borrow = false;

  let result = a.slice();

  for (let i = a.length - 1; i >= 0; i--) {

    if (!xor(result[i],b[i])) {

      result[i] = borrow;

    }

    else{

      if (result[i]) {

        if (borrow) {

          result[i] = false;

          borrow = false;

        }

      }

      else{

        result[i] = !borrow;

        borrow = true;

      }

    }

  }

  return result;

}

Код допоміжної функції isGreaterOrEqual, що перевіряє чи є перше число більше за друге:

function isGreaterOrEqual(a, b) {

  if (a.length > b.length) {

    return true;

  } else if (a.length < b.length) {

    return false;

  }

  for (let i = 0; i < a.length; i++) {

    if (xor(a[i],b[i])) {

      if (b[i]==true) {

        return false;

      }

      else{

        return true;

      }

    }

  }

  return true;

}

Фрагмент коду, що добоможе показати роботу програми:

let aInt = 55;

let bInt = 6;

let aBinary = intToBinaryArray(aInt, 16);

let bBinary = intToBinaryArray(bInt, 8);

let [quotientBinary, remainderBinary] = div(aBinary, bBinary);

let quotientInt = binaryArrayToInt(quotientBinary);

let remainderInt = binaryArrayToInt(remainderBinary);

console.log(`${aInt} / ${bInt} = ${quotientInt} and ${remainderInt}`);

console.log(`${binaryArrayToString(aBinary)} / ${binaryArrayToString(bBinary)} = ${binaryArrayToString(quotientBinary)} and ${binaryArrayToString(remainderBinary)}`);

Вивід консолі при запуску програми:

PS D:\labs\cs\Lab2> node .\Task2.js

55 / 6 = 9 and 1

0000000000110111 / 00000110 = 000001001 and 0000000000000001

3. Робота з IEEE 754 Floating Point (представити лише ключові кроки при виконанні операцій)

Множення

1. Compute exponents

Код функції, що виконує калькулювання експонент:

function addExponents(exponent1, exponent2) {

    let b127 = [false, true, true, true, true, true, true, true];

    let diff = isGreaterOrEqual(exponent1,b127) ? sub(exponent1,b127) : sub(b127, exponent1);

    return add(isGreaterOrEqual(exponent1,b127) ? add(exponent2, diff) : sub(diff, exponent2),[false, false, false, false, false, false, false, true]);

}

1. Multiply significands

Код функції, що виконує множення мантис:

function multiplyBinaryNumbers (binaryString1, binaryString2){

    let int1 = binaryArrayToInt([true].concat(binaryString1));

    let int2 = binaryArrayToInt([true].concat(binaryString2));

    return intToBinaryArray(int1 \* int2, 48);

}

1. Normalize result

Строка коду, що виконує нормалізування:

newMantissa = multiplyBinaryNumbers(mantissa1,mantissa2).slice(1,24);

iv. Set sign

Строка коду, що визначає знак:

newSign = xor(sign1, sign2);

Код допоміжної функції xor:

function xor(bit1, bit2) {

  return (bit1 && !bit2) || (!bit1 && bit2);

}

Код функції, що виконує множення IEEE754 чисел:

function multiplyIEEE754(num1, num2) {

    let sign1 = num1[0];

    let sign2 = num2[0];

    let exponent1 = num1.slice(1, 9);

    let exponent2 = num2.slice(1, 9);

    let mantissa1 = num1.slice(9);

    let mantissa2 = num2.slice(9);

    let newSign;

    let newExponent;

    let newMantissa;

    newSign = xor(sign1, sign2);

    newExponent = addExponents(exponent1,exponent2);

    newMantissa = multiplyBinaryNumbers(mantissa1,mantissa2).slice(1,24);

    return [newSign].concat(newExponent).concat(newMantissa);

}

Код допоміжних функцій decimalToIEEE754 та IEEE754ToDecimal, що проводять конвертацю між десятковим записом чиста та булевим масивом, що репрезентує собою IEEE754 число:

function decimalToIEEE754(decimal) {

    const buffer = new ArrayBuffer(4);

    const floatView = new Float32Array(buffer);

    const intView = new Uint32Array(buffer);

    floatView[0] = decimal;

    const binaryString = intView[0].toString(2).padStart(32, '0');

    return binaryString.split('').map((bit) => bit === '1');

}

function IEEE754ToDecimal(binaryArray) {

    const binaryString = binaryArray.map((bit) => bit ? '1' : '0').join('');

    const buffer = new ArrayBuffer(4);

    const intView = new Uint32Array(buffer);

    const floatView = new Float32Array(buffer);

    intView[0] = parseInt(binaryString, 2);

    return floatView[0];

}

Фрагмент коду, що добоможе показати роботу програми:

const decimal1 = 123.456;

const binary1 = decimalToIEEE754(decimal1);

const decimal2 = 123.456;

const binary2 = decimalToIEEE754(decimal2);

const productBinary = multiplyIEEE754(binary1,binary2);

const productInt = IEEE754ToDecimal(productBinary)

console.log(`${decimal1} \* ${decimal2} = ${productInt}`);

console.log(`${binaryArrayToString(binary1)} \* ${binaryArrayToString(binary2)} = ${binaryArrayToString(productBinary)}`);

Вивід консолі при запуску програми:

PS D:\labs\cs\Lab2> node .\Task3.js

123.456 \* 123.456 = 15241.3837890625

01000010111101101110100101111001 \* 01000010111101101110100101111001 = 01000110011011100010010110001001

**Покликання на код на ГітЛаб:** https://gl.vlabs.knu.ua/frecs/ce/cs/2022-2023/Stanskov/CS\_lab2

**Висновок:** Під час виконання лабораторной роботи було створено декілка програм, які повторяють відомі алгоритми додавання, віднямання, множення та ділення двійкових чисел. Алгоритми були створені з всіми труднощами використання регістрів та логічних елементів. Також було імплементовано алгоритм множення чисел IEEE754.