

Міністерство освіти і науки України

Національний технічний університет України

«Київський політехнічний інститут»

**Лабораторна робота №3**

***з дисципліни «Комп'ютерна схемотехніка»***

**«Реалізація операцій на мікроасемблері обчислювальної системи з мікропрограмним керуванням»**

Виконав студент III курсу

групи: КВ-11

ПІБ: Терентьєв Іван Дмитрович

Перевірив: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Київ 2024**

*Завдання для лабораторної роботи*

1. Розробити мікроалгоритм виконання заданої операції (за варіантом) над числами зі знаком.
2. Написати мікропрограму виконання заданої операції на мікроасемблері.
3. Виконати операцію для кількох наборів чисел. Результати подати у 16-ковій формі.

*Порядок виконання роботи*

За допомогою редактора набрати мікропрограму, виконати її та перевірити правильність роботи на контрольних прикладах.

***Варіант за списком групи №23(0010111)***

*Таблиця 1.* Варіант

|  |  |
| --- | --- |
|  | Операція |
| 1 1 1 | Добування квадратного кореня з числа з плаваючою крапкою |

***Завдання №1-2***

macro mov reg1, reg2:{or reg1, reg2, z;}

macro inc reg: {add reg,reg,z,nz;}

macro dec reg: {sub reg,reg, z,z;}

macro double reg: {add reg, reg, reg, z;}

macro shiftR reg: {add sra, reg, reg, z;}

macro shiftL reg: {add sla, reg, reg, z;}

link l1: rdm

link l2: ct

accept r0: 04BFh

\ accept r1:20bh \

\ r0 is input first byte or P

\ r1 is input second byte or X

\ r15 is output

\ r1 is PX

\ r2 is PD

\ r3 is PDelta

\ r4 is PZ

\ r5 is i

\ r6 reserved for -2 in i + 1

\ r7 reserved for 2\*PZ

\ r8 reserved for PDeltai

\ r9 reserved for masking

\ r10 reserved for PZi

\ r11 reserved for loops

\ r12 for check number for double

\ only if 16-bit mode\

{mov r1, r0;}

{and r1, r1, 0000000011111111%;}

{and r0, r0, 1111111100000000%;}

\ for X \

\ PX = X = r1 \

{mov r2, 0000001100000000%;} \ PD = 11,00000000

{xor r3,r3;} \ PDelta = 0

{mov r5, 8h;} \ i = 8

{mov r9, 0000001111111111%;} \ 10 bit out

{mov r6, 0000010000000000%;} \ -2 in i +1

{and r12, r0, 1000000000000000%;} \ mask first bit

{cjp not zo, end; or nil, r12, z;} \ if number is not positive exit

{mov r12, r0;}

{push nz, 7h;}

{shiftR r12;}

{rfct;}

{and r12, r0, 0000000000000001%;} \ mask last bit

{cjp not zo, CalcX; or nil, r12, z;} \ if number is not double

{shiftR r1;}    \ shift second byte, X once right

{add r0, r0, 100h;}    \ add one to first byte, to P

{and r12, r0, 1000000000000000%;} \ mask first bit

{cjp not zo, end; or nil, r12, z;} \ if number is not positive exit,

                                    \ it is just buffer overflow \

CalcP{shiftR r0;} \ shift first byte, P once right

{and r0, r0, 1111111100000000%;} \ mask it

CalcX{push nz, 8h;} \ loop begin get X

GetX{cjs nz, GetXLoop;} \ calc PZi

{rfct;} \ get X loop i >= 0

{and r1, r1, 0000000011111111%;} \ mask it

{cjp nz, exit;} \ exit program

org 300h

GetXLoop{double r1;} \ PX\*2

{cjp zo, home; or nil, r5, z;}

{dec r5;} \ i--

{add sra, r2, r2, r6, z;} \ PD shift right -2 in -i-1

{mov r3, r1;} \ PDelta = PX

{mov r7, r4;} \ r7 = PZ

{double r7;}  \ r7 = 2\*PZ

{sub r3, r3, r7, nz;} \ PDelta = PX - 2\*PZ

{add r3, r3, r2, z;} \ PDelta = PX - 2\*PZ + PD

\ Masking \

{and r3, r3, r9;} \ cleaning PDelta

{mov r8, r3;} \ r8 = PDelta

{mov r11, 9h;} \ r11 = 9

{push;} \ loop begin

{shiftR r8;} \ shift value right

{dec r11;} \ r11--

{loop zo;or nil,r11,z;} \ if r11 > 0

{cjs zo, addOne; or nil, r8, z;} \ if r8 == 0 go addOne

{mov r4, r15;} \ PZ = output

{crtn nz;} \ return

org 400h

addOne {mov r10, 1h;} \ r10 = 1h

{mov r11, r5;} \ r11 = i

{cjp zo, skip; or nil, r11,z;} \ if i == 0 no need to shift

{push;} \ loop begin

{shiftL r10;} \ shift value left

{dec r11;} \ r11--

{loop zo;or nil,r11,z;} \ if r11 > 0

skip{add r15, r10;} \ PZ += PZi

{mov r1, r3;} \ PX = PDelta

home{crtn nz;} \ return

exit{}

\ cleanup \

{xor r1,r1;}

{xor r2,r2;}

{xor r3,r3;}

{xor r4,r4;}

{xor r5,r5;}

{xor r6,r6;}

{xor r7,r7;}

{xor r8,r8;}

{xor r9,r9;}

{xor r10,r10;}

{xor r11,r11;}

{xor r12,r12;}

\{mov r8, r15;} for 32-bit \

{or r0, r15;} \ for 16-bit

{xor r15, r15;}

\ output in r0 and r8, r0 is P, r8 is X if 32-bit\

\ output in r0 if 16-bit\

end{}

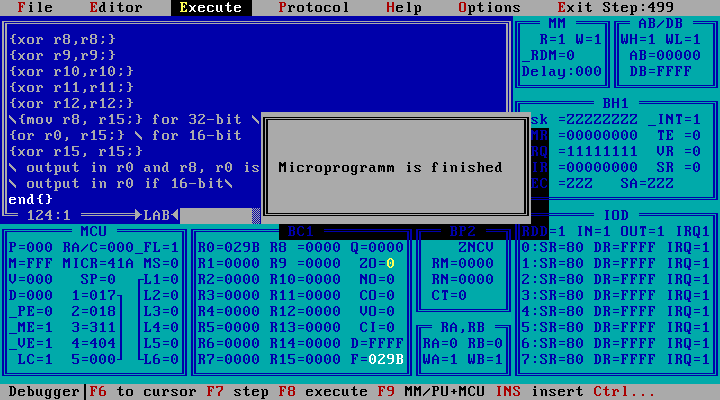
***Завдання №3***

**

*Рис. 1* – Виконання мікропрограми(при початковому значенні 5C8, 0000 0



*Рис. 2* – Виконання мікропрограми(при початковому значенні 32B, 0000 00

*Рис. 3* – Виконання мікропрограми(при початковому значенні 04BF, 0000 0