**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №5

дисциплина: Архитектура вычислительных систем

тема: «Команды сопроцессора»

Выполнил: ст. группы ПВ-31

Донцов А.А.

Проверил: Осипов О.В.

Белгород 2019

Цель работы: изучение команд сопроцессора для выполнения арифметических операций.

**Вариант 1**

****

.386 ; Тип\_процессора

.model flat, stdcall ; Модель\_памяти\_и\_стиль\_вызова\_подпрограмм

option casemap: none ; Чувствительность\_к\_регистру

; Подключение\_файлов\_с\_кодом\_макросами\_константами\_прототипами\_функций\_и\_тд

include c:\masm32\include\windows.inc

include c:\masm32\include\kernel32.inc

include c:\masm32\include\user32.inc

include c:\masm32\include\msvcrt.inc

; Подключаемые\_библиотеки

includelib c:\masm32\lib\user32.lib

includelib c:\masm32\lib\kernel32.lib

includelib c:\masm32\lib\msvcrt.lib

; Сегмент\_данных

.DATA

CONST\_1 dq 1.0

CONST\_2 dq 2.0

CONST\_3 dq 3.0

CONST\_4 dq 4.0

CONST\_Q dq 23.14069263277926

n dd 10

fmt db "n = %lf", 10, 0

.code

pow proc

FINIT ;инициализация сопроц.

FLD QWORD PTR [ESP + 12] ;st(0) = y

FLDZ ;st(0) = 0, st(1) = y

DB 0DBh, 0F0h+1 ;FCOMI 0, y (сравнение 0 и y)

FLD1 ;st(0) = 1, st(1) = 0, st(2) = y

JE exit ;if(0 == y) -> выход st0 = 1

FLD QWORD PTR [ESP + 4] ;st(0) = x, st(1) = 1, st(2) = 0, st(3) = y

FXCH ST(2) ;st(0) = 0, st(1) = 1, st(2) = x, st(3) = y

DB 0DBh, 0F0h+2 ;FCOMI 0, x (сравнение 0 и x)

JE exit ;if(0 == x) -> выход, st0 = 0

FINIT ;отчистка+инициализация

FLD QWORD PTR [ESP + 4] ;st(0) = x

FLD ST(0) ;st(0) = x, st(1) = x

FABS ;st(0) = abs(x), st(1) = x

FDIV ST(0), ST(1) ;st(0) = знак х, st(1) = x

FLD1 ;st(0) = 1, st(1) = знак х, st(2) = x

;DF 0DBh, 0F0h+1 ;FCOMIP -1, знак х

;st(0) = знак х, st(1) = x

SUB ESP, 4

DB 0DBh, 0F0h+1

fstp DWORD PTR [ESP] ;st(0) = знак х, st(1) = x

FXCH ST(1) ;Обмен значений ST(0) и ST(1)

;st(0) = x, st(1) = знак х

FABS ;st(0) = abs(x), st(1) = знак х

JA qw ;if(1 > знак х)

ADD ESP, 4

JMP contin ;знак менять не нужно

qw:

ADD ESP, 4

FLD QWORD PTR [ESP + 12];st(0) = y, st(1) = x, st(2) = знак х

FABS

FLD QWORD PTR [ESP + 12];st(0) = y, st(1) = y, st(2) = x, st(3) = знак х

FRNDINT ;округлить y

FABS

DB 0DBh, 0F0h+1 ;сравнить округл. y и y

JA qw1 ;if(ST(0) > ST(1))

JMP qw2

qw1:

FLD1 ;st(0) = 1, st(1) = округл.y, st(2) = y,

;st(3) = x, st(4) = знак х

FSUBP ST(1), ST(0) ;ST(i-1)=ST(i)-ST(0)

;ST(0)=целая часть y, ST(1)=y, ST(2)=x, ST(3)=знак х

JMP qw2

qw2:

PUSH EAX

PUSH ECX

PUSH EDX

SUB ESP, 4

fistp DWORD PTR [ESP];ST(0)=y, ST(1)=x, ST(2)=знак х

MOV EAX, [ESP] ;EAX = целая часть y

ADD ESP, 4

MOV ECX, 2

XOR EDX, EDX

DIV ECX

CMP EDX, 0

POP EDX

POP ECX

POP EAX

JE revers\_znak ;если четная степень

SUB ESP, 4

fstp DWORD PTR [ESP];ST(0)=x, ST(1)=знак х

ADD ESP, 4

JMP contin

revers\_znak:

SUB ESP, 4

fstp DWORD PTR [ESP];ST(0)=x, ST(1)=знак х

ADD ESP, 4

FXCH ST(1) ;ST(1)=знак х, ST(2)=х

FCHS ;изменить знак

FXCH ST(1) ;ST(0)=x, ST(1)=знак х

JMP contin

contin:

FLD QWORD PTR [ESP + 12];st(0) = y, ST(1)=x, ST(2)=знак х

FXCH ST(1) ;Обмен значений ST(0) и ST(1)

;st(0) = x, st(1) = y, st(2) = знак х

FYL2X ;ST(0) = ST(1)·log2(ST(0))

;выталкивает значения из ST(0) и ST(1)

;st(0) = y\*log2(x), st(1) = знак х

FLD1 ;st(0) = 1, st(1) = y\*log2(x), st(2) = знак х

FSCALE ;Масштабирование по степеням 2

;ST(0) = ST(0) · 2^ST(1)

;st(0) = 2^y\*log2(x),

;st(1) = y\*log2(x), st(2) = знак х

FLD1 ;st(0) = 1, st(1) = 2^y\*log2(x),

;st(2) = y\*log2(x), st(3) = знак х

FLD ST(2) ;st(0) = y\*log2(x), st(1) = 1, st(2) = 2^y\*log2(x),

;st(3) = y\*log2(x), st(4) = знак х

FPREM ;получение частичного остатка от деления

;ST(0) = ST(0) - Q\*ST(1)

;Q – целочисленное частное от деления

F2XM1 ;ST(0) = 2^ST(0) - 1

FADD ST(0), ST(1)

FMUL ST(0), ST(2)

FMUL ST(0), ST(4) ;st(0) = |x|^y \* знак х

exit:

SUB ESP, 8

fstp QWORD PTR [ESP]

FINIT ;отчистка+инициализация

FLD QWORD PTR [ESP] ;ST(0)= x^y

ADD ESP, 8

RET 16

pow endp

start:

MOV ECX, n ;ecx = n

MOV ESI, 0 ;i

FLDZ ;st(0) = 0

SUB ESP, 8 ;выделение памяти

FSTP QWORD PTR [ESP] ;запомнить res

;цикл====================================================

loop1:

INC ESI ;увеличение n

CMP ESI, ECX ;сравнение, если ecx==esi -> выход

JG exit1

SUB ESP, 24

;1-ое слагаемое

PUSH ESI

FILD DWORD PTR [ESP] ;ST(0) = i

POP ESI

FLD CONST\_Q ;ST(0) = q, ST(1) = i

SUB ESP, 16 ;выделение памяти в стеке

FSTP QWORD PTR [ESP] ;передать в стек q

FSTP QWORD PTR [ESP + 8] ;передать в стек i

CALL pow ;вызов pow, st(0) = q^i

FSUB CONST\_1

SUB ESP, 8

FSTP QWORD PTR[ESP]

FLD CONST\_3

FDIV QWORD PTR [ESP]

ADD ESP, 8

FSTP QWORD PTR[ESP]

;2-ое слагаемое

PUSH ESI

FLD QWORD PTR[CONST\_2]

FILD DWORD PTR [ESP] ;ST(0) = i

FMUL

POP ESI

FLD CONST\_Q ;ST(0) = q, ST(1) = i

SUB ESP, 16 ;выделение памяти в стеке

FSTP QWORD PTR [ESP] ;передать в стек q

FSTP QWORD PTR [ESP + 8] ;передать в стек i

CALL pow ;вызов pow, st(0) = q^i

FSUB CONST\_1

SUB ESP, 8

FSTP QWORD PTR[ESP]

FLD CONST\_4

FDIV QWORD PTR [ESP]

ADD ESP, 8

FSTP QWORD PTR[ESP + 8]

;3-ое слагаемое

PUSH ESI

FLD QWORD PTR[CONST\_4]

FILD DWORD PTR [ESP] ;ST(0) = i

FMUL

POP ESI

FLD CONST\_Q ;ST(0) = q, ST(1) = i

SUB ESP, 16 ;выделение памяти в стеке

FSTP QWORD PTR [ESP] ;передать в стек q

FSTP QWORD PTR [ESP + 8] ;передать в стек i

CALL pow ;вызов pow, st(0) = q^i

FSUB CONST\_1

SUB ESP, 8

FSTP QWORD PTR[ESP]

FLD CONST\_1

FDIV QWORD PTR [ESP]

ADD ESP, 8

FSTP QWORD PTR[ESP + 16]

FLD QWORD PTR [ESP + 16]

FLD QWORD PTR [ESP + 8]

FLD QWORD PTR [ESP]

ADD ESP, 16

SUB ESP, 8

FSUBRP ST(1), ST(0)

FADDP ST(1), ST(0)

FSTP QWORD PTR[ESP]

MOV EAX, [ESP+4]

MOV EDX, [ESP]

ADD ESP, 8

PUSH EAX

PUSH EDX

MOV EBX, ECX

PUSH offset fmt

CALL crt\_printf

ADD ESP, 8

MOV ECX, EBX

JMP loop1

;цикл====================================================

exit1:

call crt\_\_getch ; Задержка ввода

push 0

call ExitProcess

end start