Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ

по дисциплине Архитектура вычислительных систем

Студент Глубоков Г.В.

Группа АИ-20

Руководитель Болдырихин О. В.

Липецк 2022 г.

Задание кафедры

Изучить материал по выбранной теме: техническую документацию, периодическую, учебную и другую литературу.

Разработать программу для экспериментального исследования по выбранной теме.

Выбрать варьируемые параметры и задать их диапазон.

Определить, на что влияет и как изменение варьируемых параметров.

Сделать выводы.

Цель работы

Рассмотреть устройства FPU (Floating Point Unit) процессора x86. Изучить регистры, их устройство, разрядность и назначение. Изучить систему команд сопроцессора, функции и назначение команд, их операнды. Разработать программу на языке assembler для экспериментального исследования FPU процессора, составить таблицы состояния системы, отражающие работу и изменение регистров сопроцессора.

1 Теоретическая информация

В состав блока FPU процессоров входят восемь регистров данных R7-R0 (ST0 – ST7), регистр тегов TW, регистры управления FPCR и состояния FPSR, указатели команды FIP и данных FDP (рисунок 1).

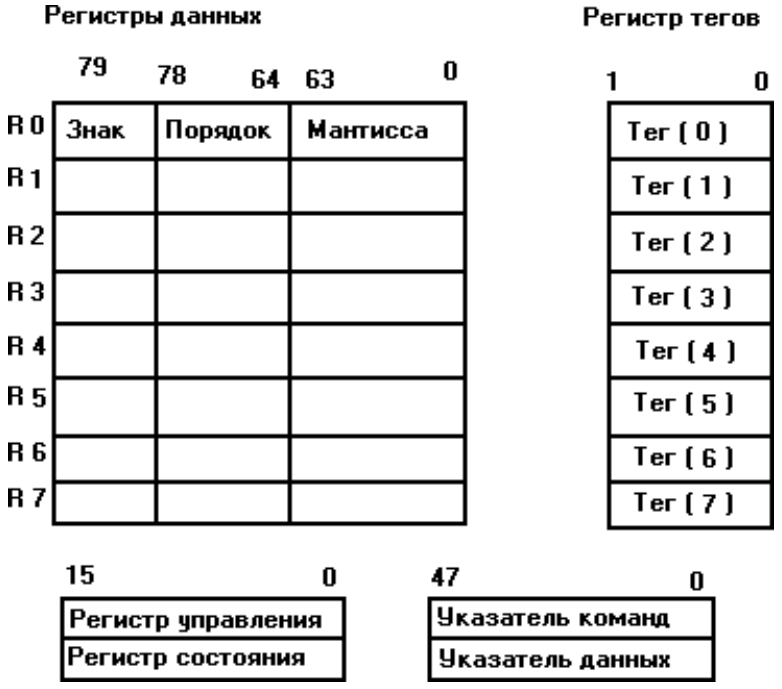

Рисунок 1 – Регистры FPU

Регистры данных R7-R0 содержат по 80 разрядов, разбитых на три поля: знак, порядок и мантисса, в соответствии с форматом представления чисел с плавающей точкой. Набор этих регистров организован в виде кольцевого стека, вершина которого определяется содержимым поля TOP в регистре состояния FPSR

Регистр тегов TW содержит восемь пар бит, описывающих содержание каждого регистра данных, - биты 15 – 14 описывают регистр R7, 13 – 12 - R6 и т.д. Если пара бит (тег) равна 11, соответствующий регистр пуст. 00 означает, что регистр содержит число, 01 - ноль, 10 - не число, бесконечность, денормализованное число, неподдерживаемое число (Рис. 2).

Регистры FIP и FDP содержат адрес последней выполненной команды (кроме FINIT, FCLEX, FLDCW, FSTCW, FSTSW, FSTSWAX, FSTENV, FLDENV, FSAVE, FRSTOR и FWAIT) и адрес ее операнда соответственно и

используются в обработчиках исключений для анализа вызвавшей его команды.

Регистр состояний FPSR содержит слово состояния FPU (Рис. 2):

Бит 15: B - занятость FPU - этот флаг существует для совместимости с 8087, и его значение всегда совпадает с ES.

Бит 14: C3 - условный флаг 3.

Биты 13 – 11: TOP - число от 0 до 7, показывающее, какой из регистров данных R0 – R7 в настоящий момент является вершиной стека.

Бит 10: C2 - условный флаг 2.

Бит 9: C1 - условный флаг 1.

Бит 8: C0 - условный флаг 0.

Бит 7: ES - общий флаг ошибки - равен 1, если произошло хотя бы одно немаскированное исключение.

Бит 6: SF - ошибка стека. Если C1 = 1, произошло переполнение (команда пыталась писать в непустую позицию в стеке), если C1 = 0, произошло антипереполнение (команда пыталась считать число из пустой позиции в стеке).

Бит 5: PE - флаг неточного результата - результат не может быть представлен точно.

Бит 4: UE - флаг антипереполнения - результат слишком маленький.

Бит 3: OE - флаг переполнения - результат слишком большой.

Бит 2: ZE - флаг деления на ноль - выполнено деление на ноль.

Бит 1: DE - флаг денормализованного операнда - выполнена операция над денормализованным числом.

Бит 0: IE - флаг недопустимой операции - произошла ошибка стека (SF = 1) или выполнена недопустимая операция.

Регистр управления FPCR (Рис. 2):

Биты 15 – 13 - зарезервированы.

Бит 12 «IC» - управление бесконечностью (поддерживается для совместимости с 8087 и 80287 - вне зависимости от значения этого бита $+\infty > -\infty$.

Биты 11 – 10 «RC» - управление округлением (Табл. 1).

Биты 9 – 8 «PC» - управление точностью (Табл. 2).

Биты 7 – 6 - зарезервированы.

Бит 5 «PM» - маска неточного результата.

Бит 4 «UM» - маска антипереполнения.

Бит 3 «OM» - маска переполнения.

Бит 2 «ZM» - маска деления на ноль.

Бит 1 «DM» - маска денормализованного операнда.

Бит 0 «IM» - маска недействительной операции.

Таблица 1 - Способы округления результатов команд FPU

Значение RC	Способ округления
00	к ближайшему числу
01	к отрицательной бесконечности
10	к положительной бесконечности
11	к нулю

Таблица 2 - Точность результатов команд

Значение PC	Точность результатов
00	одинарная точность (32-битные числа)
01	зарезервировано
10	двойная точность (64-битные числа)
11	расширенная точность (80-битные числа)

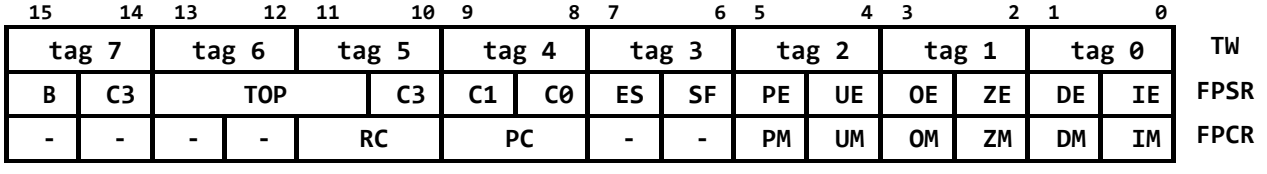

Рисунок 2 – Формат содержимого регистров TW, FPCR, FPSR

Система команд FPU содержит около 80 машинных команд, включающих в себя:

– команды передачи данных;

– команды сравнения данных;

– арифметические команды;

– команды трансцендентных функций;

– команды управления сопроцессором.

А таблицах 3-8 содержится описание основных команд FPU. Мнемоническое обозначение команд сопроцессора характеризует особенности их работы:

1. Все мнемонические обозначения начинаются с символа f (float).

2. Вторая буква мнемонического обозначения определяет тип операнда в памяти (i - целое двоичное число; b - целое десятичное число; отсутствие буквы - вещественное число).

3. Последняя буква мнемонического обозначения команды p означает, что последним действием команды обязательно является извлечение операнда из стека.

4. Последняя или предпоследняя буква r (reversed) означает реверсивное следование операндов при выполнении команд вычитания и деления, так как для них важен порядок следования операндов.

Таблица 3 - Команды загрузки констант

Команда	Описание
FLDZ	Загрузка 0
FLD1	Загрузка 1
FLDPI	Загрузка π
FLDL2T	Загрузка $\log_2 10$
FLDL2E	Загрузка $\log_2 e$
FLDLN2	Загрузка ln 2

Таблица 4 - Команды передачи данных

Команда	Операнды	Описание
FLD	src	Загрузка операнда в вершину стека
FST	dst	Сохранение вершины стека в память
FSTP	dst	Сохранение вершины стека в память с выталкиванием
FXCH	ST(i)	Обмен значений ST(0) и ST(i)

Таблица 5 - Команды сравнения данных вещественного типа

Команда	Операнды	Описание
FCOM FUCOM	src	Вещественное сравнение
FCOMP FUCOMP	src	Вещественное сравнение с выталкиванием
FCOMPP	—	Вещественное сравнение с двойным выталкиванием
FCOMI FUCOMI	ST, ST(i)	Вещественное сравнение с модификацией EFLAGS
FCOMIP FUCOMIP	ST, ST(i)	Вещественное сравнение с выталкиванием с модификацией EFLAGS
FXAM	—	Анализ ST(0)

Таблица 6 – Установка флагов регистра FPSR командами сравнения

Условие	C3	C2	C0
ST(0) > src	0	0	0
ST(0) < src	0	0	1
ST(0) = src	1	0	0
Недопустимая операция (#IA)	1	1	1

Таблица 7 – Арифметические команды вещественного типа

Команда	Операнды	Описание
FADD	dst, src	Сложение вещественное
FADDP	ST(i), ST(0)	Сложение вещественное с выталкиванием
FSUB	dst, src	Вычитание вещественное
FSUBP	ST(i), ST(0)	Вычитание вещественное с выталкиванием
FSUBR	dst, src	Вычитание вещественное реверсивное
FSUBRP	ST(i), ST(0)	Вычитание вещественное реверсивное с выталкиванием
FMUL	dst, src	Умножение вещественное
FMULP	ST(i), ST(0)	Умножение вещественное с выталкиванием
FDIV	dst, src	Деление вещественное
FDIVP	ST(i), ST(0)	Деление вещественное с выталкиванием
FDIVR	dst, src	Деление вещественное реверсивное
FDIVRP	ST(i), ST(0)	Деление вещественное реверсивное с выталкиванием

Таблица 8 – Дополнительные арифметические команды

Команда	Описание
FSQRT	Вычисление квадратного корня
FABS	Вычисление модуля
FCHS	Изменение знака
FXTRACT	Выделение порядка и мантиссы
FSCALE	Масштабирование по степеням 2
FRNDINT	Округление ST(0)
FPREM	Частичный остаток от деления

2 Листинг программы

```
;seg.asm
.386
stck segment
        dw 10h dup(0)
stck ends
data segment use16
        ten dd 10.0
        buffer dw 0
        ErrorString db "Error input", 24h
data ends
code segment use16
assume cs:code, ds:data, ss:stck
begin:
        mov dx, data
        mov ds, dx      ; Инициализация сегмента данных
        mov dx, 0B800h
        mov es, dx      ; Инициализация сегмента видеопамяти
; Очистка экрана
        mov ax,03h
        int 10h
        xor di, di
        finit
        call input      ; Ввод 1 числа
        fadd            ; Сложение st(0) и st(1)
        mov ah, 02 ; Функция DOS 02h - установка позиции каретки
        mov bh, 00
        mov dh, 1
        int 10h         ; Установка каретки на 2 строку
        call input      ; Ввод 2 числа
        mov ah, 02 ; Функция DOS 02h - установка позиции каретки
        mov bh, 00
        mov dh, 2
        int 10h
        fadd            ; Сложение st(0) и st(1)
        fxch st(1)      ; Обмен значениями st(0) <-> st(1)
        fincstp         ; Выталкивание верхушки стека
        fcom st(1) ; Сравнение st(0) и st(1) с установкой регистра Flags
        fstsw   ax              ; Считывание соятояния FPU в ax
        sahf                    ; Запись содержимого AH в младший байт регистра
флагов (SF, ZF, AF, PF и CF )
        jb add_numbers
        fsub  st(0),  st(1)
        jmp output
add_numbers:
        fadd st(0), st(1)
output:
        mov ah, 02 ; Функция DOS 02h - установка позиции каретки
        mov bh, 00
        mov dh, 3
        int 10h
        call Output_number
```

```
        jmp pause

input proc ; Подпрограмма ввода числа
   mov cx, 20
   fld ten
   fldz
input_loop:
   mov ah, 01h          ; Функция DOS 01h - ввод символа
   int 21h
   cmp al, 0Dh
   je exit_loop
   cmp al, 2Eh; Проверка на символ '.'
   je inp_order
   cmp al, 48
   jb error_inp
   cmp al, 57
   ja error_inp
   sub al, 48    ; Преобразование ASCI в число
   xor ah, ah
   fmul st(0),st(1)
   mov buffer, ax
   fiadd buffer
   dec cx
   jz exit_loop
   jmp input_loop
inp_order:
        fldz
input_order_loop:
        mov ah, 01h
   int 21h
   cmp al, 0Dh
   je exit_loop
   cmp al, 48
   jb error_inp
   cmp al, 57
   ja error_inp
   sub al, 48
   xor ah, ah
   mov buffer, ax
   fiadd buffer
   fdiv st(0), st(2)
   dec cx
   jz exit_loop
   jmp input_order_loop
exit_loop:
   ret
error_inp:
        call Error
        jmp pause
input endp

Error proc              ; Подпрограмма вывода ошибки
```

```
                mov ah, 02
                mov bh, 00
                mov dh, 3
                int 10h
                mov dx, offset ErrorString
                mov ah, 09h
                int 21h
                ret
        Error endp

        Output_number proc
                mov bp, sp
                fld1                    ; Помещение в стек 1 (1 - 52.34)
                fld st(1)               ; (52.34 - 1 - 52.34)
                fprem                   ; Остаток от st(0) % st(1) (0.34 - 1 - 52.34)
                fsub st(2),st(0); (0.34 - 1 - 52)
                fxch st(2)              ; Обмен с st(2) (52 - 1 - 0.34)
                xor cx, cx              ; Обнуление счетчика цифр
        loop_whole:
                fld ten                 ; (10 - 52 - 1 - 0.34)
                fdivp                   ; (5.2 - 1 - 0.34)
                fxch st(1)              ; (1 - 5.2 - 0.34)
                fld st(1)               ; (5.2 - 1 - 5.2 - 0.34)
                fprem                   ; (0.2 - 1 - 5.2 - 0.34)
                fsub st(2),st(0); (0.2 - 1 - 5 - 0.34)
                fld ten                 ; (10 - 0.2 - 1 - 5 - 0.34)
                fmulp                   ; (2 - 1 - 5 - 0.34)
                fistp buffer    ; Помещение верхушки в память (1 - 5 - 0.34)
                push buffer
                inc cx
                fxch st(1)              ; (5 - 1 - 0.34)
                ftst                    ; Сравнение с нулем st(0)
                fstsw   ax              ; Считывание соятояния FPU в ax
                sahf                    ; Запись содержимого AH в младший байт регистра
флагов (SF, ZF, AF, PF и CF )
                jnz loop_whole
        outp_whole:                     ; Цикл вывода целой части
                mov ah, 02h
                pop dx
                add dx, 48
                int 21h
                loop outp_whole         ; Далее вывод дробной части
                fincstp                 ; Выталкивание верхушки стека (1 - 0.34)
          fxch st(1)    ; (0.34 - 1)
          ftst
          fstsw ax
          sahf
          jz exit_output ; Если дробной части нет
          mov ah, 02h
          mov dl, 2Eh                   ; Вывод символа '.'
          int 21h
        loop_fract:
```

```
    fld ten                     ; (10 - 0.34 - 1)
        fmulp                   ; (3.4 - 1)
    fxch st(1)      ; (3.4 - 1 - 1)
    fld st(1)       ; (3.4 - 1 - 3.4)
    fprem                       ; (0.4 - 1 - 3.4)
    fsub st(2),st(0); (0.4 - 1 - 3)
    fxch st(2)              ; (3 - 1 - 0.4)
    fistp buffer ; Помещение верхушки в память (1 - 0.4)
    mov dx, buffer ; Вывод цифры из дробной части
    add dx, 48
    mov ah, 02h
    int 21h
    fxch st(1)      ; (0.4 - 1)
    ftst                    ; Сравнение с нулем st(0)
    fstsw ax
    sahf
    jnz loop_fract
exit_output:
        ret
Output_number endp

pause:
        in al, 60h
        cmp al, 01h
        jne pause
exit:
        mov ax, 4c00h
        int 21h
code ends
end begin
```

На рисунке 3 представлен результат выполнения данной программы для случайно введенных вещественных чисел.

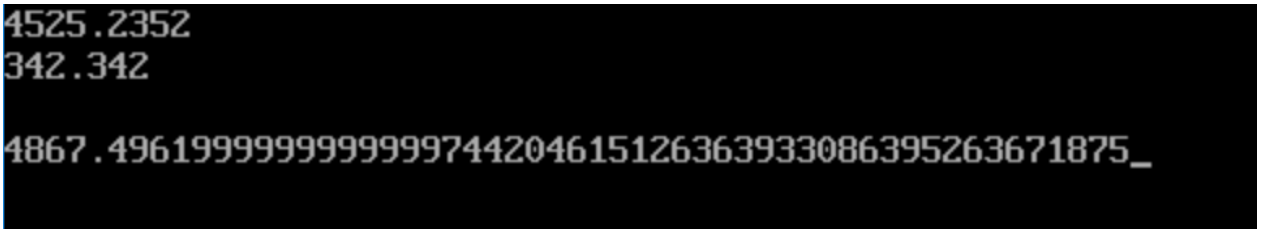

Рисунок 3 – Результат выполнения программы

3 Таблицы состояния системы

В таблице 9 приведены состояния регистров FPU в момент выполнения подпрограммы ввода числа с плавающей точкой (input proc).

Таблица 9 – Состояния системы при выполнении подпрограммы input

№	Адрес команды	Команда на машинном языке	Команда на языке ассемблера	Содержание изменившихся регистров		
				Наим.	Значение	TW
1	004E	B9 0014	mov cx, 20	cx: 0014h		
2	0051	D9 06 0000r	fld ten	ST(0)	10	Valid
				ST(1)	-	Empty
				…	-	Empty
3	0055	D9 EE	fldz	ST(0)	0	Zero
				ST(1)	10	Valid
				ST(2)	-	Empty
				…	-	Empty
Ввод целой части числа						
4	0057	B4 01	mov ah, 01h	ax: 0100h		
5	0059	CD 21	int 21h	ax:0131h		
6	005B	3C 0D	cmp al, 0Dh	z:0 a:1		
7	005D	74 42	je exit_loop	-		
8	005F	3C 2E	cmp al, 2Eh ;	-		
9	0061	74 1A	je inp_order	-		
10	0063	3C 30	cmp al, 48	p:1 a:0		
11	0065	72 3B	jb error_inp	-		
12	0067	3C 39	cmp al, 57	c:1 s:1 a:1		
13	0069	77 37	ja error_inp	-		
14	006B	2C 30	sub al, 48	ax:0101h		
15	006D	32 E4	xor ah, ah	ax:0001h		
16	006F	D8 C9	fmul st(0),st(1)	ST(0)	ST(0) = 0*10	Zero
				ST(1)	10	Valid
				ST(2)	-	Empty
				…	-	Empty
17	0071	A3 0004r	mov buffer, ax	buffer:00001h		
18	0074	DE 06 0004r	fiadd buffer	ST(0)	1	Valid
				ST(1)	10	Valid
				ST(2)	-	Empty
				ST(3)	-	Empty
				…	-	Empty
19	0078	49	dec cx	cx:0013h		
20	0079	74 26	jz exit_loop	-		
21	007B	EB DA	jmp input_loop	ip: 0057h		

№	Адрес команды	Команда на машинном языке	Команда на языке ассемблера	Содержание изменившихся регистров		
				Наим.	Значение	TW
Ввод вещественной части числа						
22	007D	D9 EE	fldz	ST(0)	0	Zero
				ST(1)	1	Valid
				ST(2)	10	Valid
				ST(3)	-	Empty
				…	-	Empty
23	007F	B4 01	mov ah, 01h	ax:0101h		
24	0081	CD 21	int 21h	ax:0137h		
25	0083	3C 0D	cmp al, 0Dh	z:0 a:1		
26	0085	74 1A	je exit_loop	-		
27	0087	3C 30	cmp al, 48	p:1 a:0		
28	0089	72 17	jb error_inp	-		
29	008B	3C 39	cmp al, 57	c:1 s:1 a:1		
30	008D	77 13	ja error_inp	-		
31	008F	2C 30	sub al, 48	ax: 0107h		
32	0091	32 E4	xor ah, ah	ax:0007h		
33	0093	A3 0004r	mov buffer, ax	buffer:0007h		
34	0096	DE 06	fiadd buffer	ST(0)	1.7 (дробн. ч.)	Valid
				ST(1)	14(цел. числ.)	Valid
				ST(2)	10	Valid
				ST(3)	-	Empty
				…	-	Empty
35	009A	D8 F2	fdiv st(0), st(2)	ST(0)	0.17 (дробн. ч.)	Valid
				ST(1)	14 (цел. числ.)	Valid
				ST(2)	10	Valid
				ST(3)	-	Empty
				…	-	Empty
36	009C	49	dec cx	cx:0011h		
37	009D	74 02	jz exit_loop	-		
38	009F	EB DE	jmp input_order_loop	ax: 4C00; ip:0020		
39	00A1	C3	ret	-		

Таблица 10 – Состояния системы в случае не правильного кода

№	Адрес команды	Команда на машинном языке	Команда на языке ассемблера	Содержание изменившихся регистров		
				Наим.	Значение	TW
Вывод целой части числа						
1	00B8	8B EC	mov bp, sp	bp:FFFEh		
2	00BA	D9 E8	fld1	ST(0)	1	Valid
				ST(1)	22.84 (вывод)	Valid
				ST(2)	14.71	Valid
				ST(3)	10	Valid
				ST(4)	-	Empty
				…	-	Empty
3	00BC	D9 C1	fld st(1)	ST(0)	22.84	Valid
				ST(1)	1	Valid
				ST(2)	22.84	Valid
				ST(3)	14.71	Valid
				ST(4)	10	Empty
				…	-	Empty
4	00BE	D9 F8	fprem	ST(0)	0.84	Valid
				ST(1)	1	Valid
				ST(2)	22.84	Valid
				ST(3)	14.71	Valid
				ST(4)	10	Empty
				…	-	Empty
5	00C0	DC EA	fsub st(2),st(0)	ST(0)	0.84	Valid
				ST(1)	1	Valid
				ST(2)	22	Valid
				ST(3)	14.71	Valid
				ST(4)	10	Empty
				…	-	Empty
6	00C2	D9 CA	fxch st(2)	ST(0)	22	Valid
				ST(1)	1	Valid
				ST(2)	0.84	Valid
				ST(3)	14.71	Valid
				ST(4)	10	Empty
				…	-	Empty
7	00C4	33 C9	xor cx, cx	cx:0000h		
8	00C6	D9 06 0000r	fld ten	ST(0)	10	Valid
				ST(1)	22	Valid
				ST(2)	1	Valid
				ST(3)	0.84	Valid
				ST(4)	14.71	Valid
				ST(5)	10	Empty
				…	-	Empty
9	00CA	DE F9	fdivp	ST(0)	2.2	Valid
				ST(1)	1	Valid
				ST(2)	0.84	Valid
				ST(3)	14.71	Valid
				ST(4)	10	Valid
				…	-	Empty
10	00CC	D9 C9	fxch st(1)	ST(0)	1	Valid
				ST(1)	2.2	Valid
				ST(2)	0.84	Valid
				ST(3)	14.71	Valid
				ST(4)	10	Valid
				…	-	Empty

№	Адрес команды	Команда на машинном языке	Команда на языке ассемблера	Содержание изменившихся регистров		
				Наим.	Значение	TW
11	00CE	D9 C1	fld st(1)	ST(0)	2.2	Valid
				ST(1)	1	Valid
				ST(2)	2.2	Valid
				ST(3)	0.84	Valid
				ST(4)	14.71	Valid
				ST(5)	10	Valid
				…	-	Empty
12	00D0	D9 F8	fprem	ST(0)	0.2	Valid
				ST(1)	1	Valid
				ST(2)	2.2	Valid
				ST(3)	0.84	Valid
				ST(4)	14.71	Valid
				ST(5)	10	Valid
				…	-	Empty
13	00D2	DC EA	fsub st(2),st(0)	ST(0)	0.2	Valid
				ST(1)	1	Valid
				ST(2)	2	Valid
				ST(3)	0.84	Valid
				ST(4)	14.71	Valid
				ST(5)	10	Valid
				…	-	Empty
14	00D4	D9 06 0000r	fld ten	ST(0)	10	Valid
				ST(1)	0.2	Valid
				ST(2)	1	Valid
				ST(3)	2	Valid
				ST(4)	0.84	Valid
				ST(5)	14.71	Valid
				ST(6)	10	Valid
				…	-	Empty
15	00D8	DE C9	fmulp	ST(0)	2	Valid
				ST(1)	1	Valid
				ST(2)	2	Valid
				ST(3)	0.84	Valid
				ST(4)	14.71	Valid
				ST(5)	10	Valid
				…	-	Empty
16	00DA	DF 1E 0004r	fistp buffer	ST(0)	1	Valid
				ST(1)	2	Valid
				ST(2)	0.84	Valid
				ST(3)	14.71	Valid
				ST(4)	10	Valid
				…	-	Empty
17	00DE	FF 36 0004r	push buffer	buffer:0002h		
18	00E2	41	inc cx	cx:0001h		
19	00E3	D9 C9	fxch st(1)	ST(0)	2	Valid
				ST(1)	1	Valid
				ST(2)	0.84	Valid
				ST(3)	14.71	Valid
				ST(4)	10	Valid
				…	-	Empty
20	00E5	D9 E4	ftst	c:010b		
21	00E7	9B DF E0	fstsw ax	ax:1820h		

№	Адрес команды	Команда на машинном языке	Команда на языке ассемблера	Содержание изменившихся регистров		
				Наим.	Значение	TW
22	00EA	9E	sahf	a:1		
23	00EB	75 D9	jnz loop_whole	-		
24	00ED	B4 02	mov ah, 02h	ax:0200h		
25	00EF	5A	pop dx	sp:FFFC dx:0001h		
26	00F0	83 C2 30	add dx, 48	dx:0001h		
27	00F3	CD 21	int 21h	dx:0031h		
28	00F5	E2 F6	loop outp_whole	-		
29	00F7	D9 F7	fincstp	ST(0)	1	Valid
				ST(1)	0.84	Valid
				ST(2)	14.71	Valid
				ST(3)	10	Valid
				ST(4)	-	Empty
				…	-	Empty
30	00F9	D9 C9	fxch st(1)	ST(0)	0.84	Valid
				ST(1)	1	Valid
				ST(2)	14.71	Valid
				ST(3)	10	Valid
				ST(4)	-	Empty
				…	-	Empty
31	00FB	D9 E4	ftst	c:010b		
32	00FD	9B DF E0	fstsw ax	ax:2020		
33	0100	9E	sahf	-		
34	0101	74 2F	jz exit_output	-		
Вывод вещественной части						
35	0103	B4 02	mov ah, 02h	ax:0200h		
36	0105	B2 2E	mov dl, 2Eh	dx:002Eh		
37	0107	CD 21	int 21h	-		
38	0109	D9 06 0000r	fld ten	ST(0)	10	Valid
				ST(1)	0.84	Valid
				ST(2)	1	Valid
				ST(3)	14.71	Valid
				ST(4)	-	Empty
				…	-	Empty
39	010D	DE C9	fmulp	ST(0)	8.4	Valid
				ST(1)	1	Valid
				ST(2)	14.71	Valid
				ST(3)	-	Empty
				…	-	Empty
40	010F	D9 C9	fxch st(1)	ST(0)	1	Valid
				ST(1)	8.4	Valid
				ST(2)	14.71	Valid
				ST(3)	-	Empty
				…	-	Empty
41	0111	D9 C1	fld st(1)	ST(0)	8.4	Valid
				ST(1)	1	Valid
				ST(2)	8.4	Valid
				ST(3)	14.71	Valid
				…	-	Empty

№	Адрес команды	Команда на машинном языке	Команда на языке ассемблера	Содержание изменившихся регистров		
				Наим.	Значение	TW
42	0113	D9 F8	fprem	ST(0)	0.4	Valid
				ST(1)	1	Valid
				ST(2)	8.4	Valid
				ST(3)	14.71	Valid
				ST(4)	-	Empty
				…	-	Empty
43	0115	DC EA	fsub st(2), st(0)	ST(0)	0.4	Valid
				ST(1)	1	Valid
				ST(2)	8	Valid
				ST(3)	14.71	Valid
				ST(4)	-	Empty
				…	-	Empty
44	0117	D9 CA	fxch st(2)	ST(0)	8	Valid
				ST(1)	1	Valid
				ST(2)	0.4	Valid
				ST(3)	14.71	Valid
				ST(4)	-	Empty
				…	-	Empty
45	0119	DF 1E 0004r	fistp buffer	buffer:0008h		
46	011D	8B 16 0004r	mov dx, buffer	dx:0008h		
47	0121	83 C2 30	add dx, 48	dx:0038h		
48	0124	B4 02	mov ah, 02h	ax:0200h		
49	0126	CD 21	int 21h	-		
50	0128	D9 C9	fxch st(1)	ST(0)	0.4	Valid
				ST(1)	1	Valid
				ST(2)	14.71	Valid
				ST(3)	-	Empty
				…	-	Empty
51	012A	D9 E4	ftst	c:0101b		
52	012C	9B DF E0	fstsw ax	ax:0238h		
53	012F	9E	sahf	-		
54	0130	75 D7	jnz loop_fract	ip:0109h		
55	0132	C3	ret	-		

4 Вывод

В ходе выполнения данной лабораторной работы были исследована система команд и назначения регистров FPU. Написана программа ввода- вывода вещественного числа с помощью FPU, а также проанализированы состояния FPU в ходе выполнения. В ходе исследования были выяснены следующее:

1. FPU не является полноценным процессором, так как не умеет делать многих необходимых для этого операций (например, не умеет работать с программой и вычислять адреса памяти), являясь всего лишь придатком центрального процессора.

2. FPU организует свои регистры не как массив, как большинство других архитектур, а как регистровый стек. Это означает, что в один момент времени только два верхних регистра доступны для проведения операций, а доступ к другим регистрам требует манипуляций со стеком.

3. После получения команды и необходимых данных сопроцессор начинает её выполнение. Пока сопроцессор выполняет команду, центральный процессор выполняет программу дальше, параллельно с вычислениями сопроцессора. Если следующая команда также является командой сопроцессора, процессор останавливается и ожидает завершения выполнения сопроцессором предыдущей команды. Данная остановка осуществляется с помощью специальной команды ожидания (FWAIT).

4. Внутри FPU числа хранятся в 80-битном формате с плавающей запятой, для записи же или чтения из памяти могут использоваться:

– Вещественные числа в трёх форматах: коротком (32 бита), длинном (64 бита) и расширенном (80 бит).

– Двоичные целые числа в трёх форматах: 16, 32 и 64 бита.