Лабораторная работа 3. АРХИТЕКТУРА И ПРОГРАММИРОВАНИЕ АРИФМЕТИЧЕСКОГО СОПРОЦЕССОРА

В процессорах Intel операции с плавающей запятой выполняет специальное устройство – арифметический сопроцессор, который начиная с процессора 80486 встраивается в основной процессор.

Сопроцессор работает параллельно с целочисленным процессором. Параллельная работа уменьшает время обработки, позволяя математическому сопроцессору производить математические вычисления, в то время как процессор продолжает выполнять другие функции.

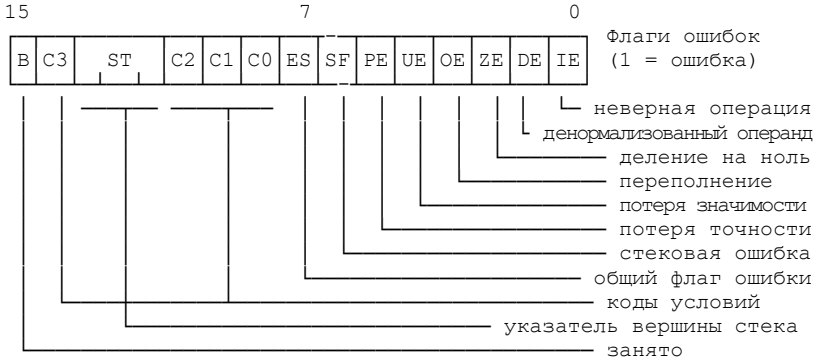
Для программиста сопроцессор состоит из восьми 80-битных регистров, регистра управления, регистра состояния, регистра признаков и указателей особых случаев. 80-битные регистры используются для хранения констант и промежуточных результатов в процессе вычислений, уменьшая, таким образом, число обращений к памяти и повышая одновременно и скорость, и доступность магистрали. Регистровое пространство может быть использовано как стек или как фиксированный набор регистров. Данные в регистрах хранятся во временном вещественном формате, который используется при всех вычислениях.

Поле ST в слове состояния определяет текущий верхний регистр стека. Операция загрузки уменьшает ST на единицу и загружает новую величину в верхний регистр стека. Операция извлечения и сохранения переписывает в нужное место величину верхнего регистра стека и увеличивает ST на единицу. Таким образом регистровый стек сопроцессора растет в направлении от старшего регистра к младшему.

Команды могут адресовать регистры прямо и косвенно. Команды, которые работают с вершиной стека, осуществляют косвенную адресацию регистра, на который указывает ST. Например, команда FSQRT замещает число в вершине стека его квадратным корнем; эта команда не получает операндов, так как в качестве операнда используется верхний регистр стека. Прямая адресация регистров зависит от вершины стека. Выражение ST

определяет текущую вершину стека, а ST(i) ссылается на i-й регистр от ST. Например, если ST содержит двоичное 011 (регистр 3 находится в вершине стека), то команда FADD ST,ST(2) будет складывать регистры 3 и 5.

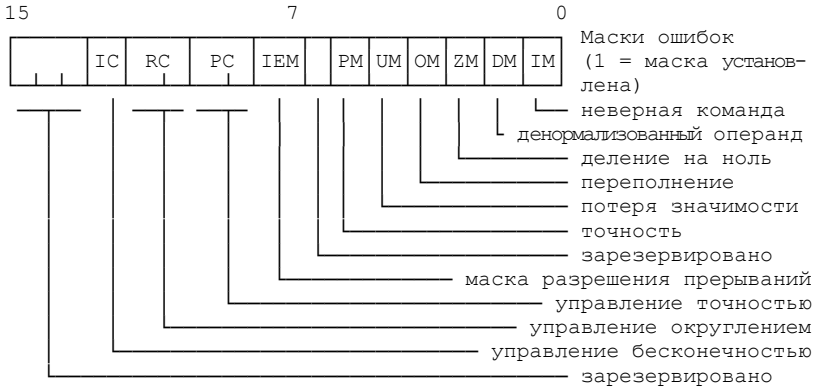
Слово состояния отражает все условия сопроцессора. Оно может быть сохранено в памяти по команде сопроцессора и затем проверено процессором. Слово состояния делится на несколько полей:



Некоторые инструкции, например инструкции сравнения, передают свои результаты в качестве кодов условий (биты 14, 10-8). Основное назначение кодов условий – для условного ветвления.

Биты с 13 по 11 слова состояния указывают на регистр сопроцессора, который является текущей вершиной стека. Бит 7 – это поле запроса на прерывание, а биты 5-0 служат для индикации того, что устройство обработки чисел сопроцессора обнаружило ошибку при обработке команды.

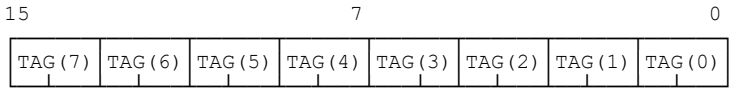
Слово управления сопроцессора определяет его режим работы: точность вычислений, способ округления, реакцию на ошибки и др.:



Слово управления загружается из памяти специальной командой, кроме того, содержимое регистра управления устанавливается при инициализации сопроцессора и может быть оставлено по умолчанию.

Маска разрешения прерываний: 0 – прерывания разрешены, 1 – прерывания запрещены (маскированы). Управление точностью: 00 – 24 бита, 01 – зарезервировано, 10 – 53 бита, 11 – 64 бита. Управление округлением: 00 – округлять до ближайшего или четного, 10 – округлять вверх, 01 – округлять вниз, 11 – отбрасывать. Управление бесконечностью: 0 – проективная, 1 – афинная.

Слово признаков (тэгов) предназначено для пометки содержимого каждого регистра:



Основная его функция – оптимизация работы сопроцессора под несколькими управляющими программами. Величины признаков: 00 – конечное ненулевое число, 01 – истинный нуль, 10 – специальное число (не-число, бесконечность или денормализованное), 11 – пусто.

Указатели на ошибку необходимы для написанных пользователем программ обработки ошибок. Когда сопроцессор выполняет команду, то устройство управления хранит в регистрах указателя на ошибку адреса команды и операнда. Подпрограмма обработки ошибок может сохранить эти указатели в памяти и определить, какая команда привела к возникновению условия ошибки.

Арифметический сопроцессор работает с несколькими типами числовых данных, разделяя их на три класса: двоичные целые (три типа), десятичные целые (один тип), вещественные числа (три типа):

Основные характеристики чисел приведены в табл. 1. Короткие, длинные и временные вещественные типы данных соответствуют вещественным данным с одинарной, двойной и расширенной точностью.

Таблица 1

Характеристики типов данных сопроцессора

Тип данных	Биты	Диапазон значений (десятичный)
word_integer	16	-32768…+32767
short_integer	32	-2E+9…+2E9
long_integer	64	-9E18…+9E18
bcd	80	-99..99…+99..99
short_real	32	-8,43E-37…+3,37E+38
long_real	64	-4,19E-307…+1,67E+308
temp_real	80	-3,4E-4932…+1,2E+4932

Система команд сопроцессора описана в прил. 2.

Рассмотрим примеры разработки программ.

Задание. Составить программы вычисления функции $z=\exp(x)$ для следующих случаев: а) с использованием трансцендентных команд сопроцессора; б) путем разложения функции в ряд

$$\exp(x) = 1 + x/1! + x^2/2! + ... + x^n/n! + ...$$

В случае (б) считать, что требуемая точность *eps* достигнута, если очередное слагаемое по модулю меньше *eps*.

Решение.

```
;ВЫЧИСЛЕНИЕ ЭКСПОНЕНТЫ С ПОМОЩЬЮ ФУНКЦИЙ СОПРОЦЕССОРА

cseg     segment
         assume  cs:cseg,ds:cseg

x     dd      1.2    ; аргумент функции
z     dd      ?      ; результат вычисления функции

beg:  mov     ax,cs          ; настроить сегментные
      mov     ds,ax          ;  регистры
      finit                  ; инициализировать
                             ;  сопроцессор
      fld     x              ; загрузить x
      fldl2e                 ; загрузить log2 e
      fmulp   st(1),st(0)    ; вычислить y=x*log2 e
      fld     st(0)          ; образовать копию y
      frndint                ; округлить до целого y1
      fsub    st(1),st(0)    ; выделить дробную часть
                               ;  y2=y-y1
```

```
      fxch                  ; обменять y2 и y1
      f2xm1                 ; вычислить 2**(y2)-1
      fld1                  ; загрузить 1
      faddp   st(1),st      ; вычислить 2**(y2)
      fscale                ; домножить на 2^y1
      fstp    st(1)         ; удалить y1
      fstp    z             ; сохранить результат
      mov     ax,4c00h      ; вернуться в DOS
      int     21h
Cseg  ends
      end     beg

; ВЫЧИСЛЕНИЕ ЭКСПОНЕНТЫ С ПОМОЩЬЮ РАЗЛОЖЕНИЯ В РЯД
; Алгоритм вычисления:
;  1. n=0; Delta=1; S=1
;  2. n=n+1; Delta=Delta*x/n; S=S+Delta
;  3. Если ABS(Delta)>eps идти к 2, иначе - закончить
; Распределение регистров:
;  ST(0) - рабочий, ST(1) - S, ST(2) - Delta,
;  ST(3) - n, ; ST(4) - x, ST(5) - 1.0, ST(6) - eps

Cseg    segment
        assume  cs:Cseg,ds:Cseg

x       dq      1.0     ; аргумент функции
sum     dq      ?       ; результат вычисления функции
eps     dq      1.0E-05; точность вычисления
                        ;
start:  mov     ax,cs   ; настройка сегмента данных на cs
        mov     ds,ax
        finit           ; инициализация сопроцессора
; реализация первого шага алгоритма
        Fld     eps
        fld1
        fld     x
        fldz
        fld1
        fld1
        fld1
; шаг 2
calc:   fxch    st(3)       ;
        fadd    st(0),st(5) ; n=n+1
```

```
        fst     st(3)
        fdivr   st(0),st(2) ; Delta/n
        fmul    st(0),st(4) ; (Delta/n)*x
        fst     st(2)
        fadd    st(1),st(0) ; S=S+Delta
; шаг 3
        fabs                ;, ABS(Delta)
;       fcomi   st(6)       ; ABS(Delta)>eps
        db      0dbh,0f6h; машинный код команды
        ja      calc        ; переход к шагу 2
        fstp    st
        fstp    sum         ; запоминание результата
        finit
        mov     ax,4c00h ; выход из программы
         int     21h
Cseg     ends
         end     start
```

Задание к лабораторной работе

1. Дано целое число $1 \le n \le 12$, вещественные числа t, a_0, a_1, ..., a_n. Вычислить значение многочлена

$$a_0 \cdot x^n + a_1 \cdot x^{n-1} + \ldots + a_{n-1} \cdot x + a_n$$

и его производной в точке t.

2. Вычислить $y = \cos(x) + \cos^2(x) + \cos^3(x) + \ldots + \cos^n(x)$, где $1 \le n \le 32$.

3. Вычислить y – первое из чисел $\sin(x)$, $\sin(\sin(x))$, $\sin(\sin(\sin(x)))$, ... , меньшее по модулю заданного числа *eps*.

4. Вычислить функцию $y = \mathrm{sh}(x)$ двумя способами: а) с использованием трансцендентных команд сопроцессора; б) путем разложения функции в ряд $y = x + x^3/3! + x^5/5! + \ldots$.

5. Вычислить функцию $y = \cos(x)$ двумя способами: а) с использованием трансцендентных команд сопроцессора; б) путем разложения функции в ряд $y = 1 - x^2/2! + x^4/4! - \ldots$.

6. Вычислить функцию $y = \ln(1+x)$ при $|x|<1$ двумя способами: а) с использованием трансцендентных команд сопроцессора; б) путем разложения функции в ряд $y = x - x^2/2 + x^3/3 - \ldots$.

7. Вычислить функцию $y = \arctan(x)$ при $|x|<1$ двумя способами: а) с использованием трансцендентных команд сопроцессора; б) путем разложения функции в ряд $y = x - x^3/3 + x^5/5 - \ldots$.

8. Вычислить интеграл

$$\int_a^b \ln(2+\sin(x))dx,$$

используя формулу прямоугольников

$$\int_a^b f(x)dx = h\cdot[f(x_1)+f(x_2)+\ldots+f(x_n)],$$

где $n = 100$, $h = (b - a)/n$, $x_i = a + i\cdot h - h/2$.

9. Дано n вещественных чисел, $n \le 32$. Найти порядковый номер того из них, которое наиболее близко к какому-нибудь целому числу.

10. Даны целое $1< n \le 32$ и вещественные числа $x_1, x_2,..., x_n$. Вычислить

$$M = \sum x_i / n, \quad D = \sqrt{\sum (x_i - M)^2 /(n-1)}.$$

11. Дана вещественная матрица размером 4×4 все элементы которой различны. Найти скалярное произведение строки, в которой находится наибольший элемент матрицы, на столбец с наименьшим элементом.

12. Даны две квадратные вещественные матрицы 4-го порядка. Получить квадрат той из них, в которой наименьший след (сумма диагональных элементов).

13. Даны длины a, b и c сторон некоторого треугольника. Найти медианы треугольника, сторонами которого являются медианы исходного треугольника (длина медианы, проведенной к стороне a, равна $0.5\sqrt{2b^2 + 2c^2 - a^2}$).

14. По заданным вещественным числам c и d ($c < d$) вычислить

$$\int_c^d \sin(e^{-10x})dx,$$

используя формулу трапеций при $n = 40$

$$\int_a^b f(x)dx = h\cdot[f(a)/2 + \sum_{i=1}^{n-1} f(a+i\cdot h) + f(b)/2], \text{ где } h = (b-a)/2.$$

15. Даны вещественные коэффициенты многочленов P(x) и Q(x) 8-й степени и вещественное число a. Вычислить величину P(a+Q(a)P(a+1)).

6. По вещественному числу $a > 0$ вычислить величину

$(\sqrt[3]{a} - \sqrt[6]{a^2 + 1}) / (1 + \sqrt[7]{3 + a})$.

17. По вещественному числу t вычислить величину

$\sqrt[4]{1 - \cos^4(t)/4} + \sqrt[5]{1 + arctg(t)/2} \cdot \sqrt[9]{1/(3 + t^2)}$.

18. Дано комплексное число z (пара вещественных чисел). Вычислить значение комплексной функции $y = \sin(z)$.

19. Дано комплексное число z (пара вещественных чисел). Вычислить значение комплексной функции $y = \cos(z)$.

20. Найти корни квадратного трехчлена с заданными комплексными коэффициентами.

Порядок выполнения работы

1. Изучить основные сведения по работе, при необходимости обратиться к математическому справочнику.

2. Разработать алгоритм и программу решения задачи на языке ассемблера, подобрать контрольные примеры. Показать содержимое регистров стека сопроцессора после выполнения каждой его команды.

3. Выполнить ввод, трансляцию, построение кода программы.

4. Используя отладчик TD, отладить программу, выполнить контрольные примеры и записать их результаты.

Содержание отчета

1. Цель работы.

2. Текст задания, схема реализации, блок-схема или программа на языке высокого уровня.

3. Текст программы.

4. Результаты работы программы на контрольных примерах.

5. Выводы по работе.

Контрольные вопросы

1. С какими форматами данных может работать сопроцессор? Привести примеры их определения на языке ассемблера.

2. Что такое неявный бит мантиссы и смещенный порядок в формате вещественных чисел сопроцессора? Почему применяется такой формат (достоинства и недостатки)?

3. В чем отличия регистрового стека сопроцессора от стека, который реализуется в ОЗУ, например, МП 8086?

4. Каково будет содержимое регистров стека сопроцессора после выполнения команды FSTP ST(3), если до выполнения ST(0)=0.1, ST(1)=0.2, ST(2)=0.3, ST(3)=0.4, ST(4)=0.5?

5. Объясните необходимость наличия у сопроцессора команд загрузки констант.

6. Рассмотрите все особые случаи, которые могут возникнуть в сопроцессоре при выполнении команды FADD mem. Какой результат вернет сопроцессор, если прерывания запрещены?

7. Как осуществляется условный переход по результатам сравнения чисел в сопроцессоре?

8. В чем заключается особенность обратных форм команд вычитания и деления в сопроцессоре? Как смоделировать эти формы, если бы указанных команд не было?

9. Какие действия выполняются в сопроцессоре по команде FINIT?

10. Какие трансцендентные функции вычисляет сопроцессор? Каковы особенности трансцендентных команд?

СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ

1. Андреева А.А. Основы программирования персонального компьютера на языке ассемблера: лабораторный практикум. Чебоксары. Изд-во Чуваш. ун-та, 2013. 84 с.
2. Брэй Б. Микропроцессоры Intel: 8086/8088…80486, Pentium: пер. с англ. СПб: BHV-Петербург, 2005. 1328 с
3. Зубков С.В. Ассемблер для DOS, Windows и UNIX. М.: ДМК-Пресс, 2013. 638 с.
4. Юров В.И. Assembler: учебник для вузов. СПб.: Питер, 2011. 640 с.
5. Юров В.И. Assembler: практикум: учеб. Пособие для вузов. СПб.: Питер, 2007. 400 с.

ПРИЛОЖЕНИЕ

Набор команд сопроцессора

Операнды любых команд могут быть закодированы несколькими способами. Например, команда FADD (сложение вещественное) может быть записана без операндов, только с операн-

дом-источником, а также с источником и приемником. При описании команд для разделения альтернативных форм представления операндов используется наклонная черта, причем черта без последующей спецификации означает отсутствие явно задаваемых операндов. Таким образом, для команды FADD возможные комбинации операндов будут представлены следующим образом:

//источник/приемник, источник

Это означает, что, например, команда FADD может быть записана в одной из следующих форм:

FADD

FADD источник

FADD приемник, источник.

Важно помнить, что операнды в памяти могут быть закодированы в любом из режимов адресации памяти процессора.

В мнемониках команд приняты следующие соглашения:

– первая буква всегда F (Floating) и обозначает команду сопроцессора;

– вторая буква I (Integer) обозначает операцию с целыми числами, буква B (BCD) – с целыми в BCD-формате, отсутствие букв I и B – операцию с вещественными числами;

– предпоследняя или последняя буква R (Reversed) указывает обратную операцию;

– последняя буква P (Pop) идентифицирует команду, заканчивающуюся извлечением из стека.

Для некоторых команд управления работой сопроцессора имеются альтернативные мнемоники, второй буквой которых является N (например, FSAVE/FNSAVE). Мнемоники такого типа сообщают ассемблеру, автоматически вставляющему в программе для процессора 8086 перед каждой командой сопроцессора команду WAIT (ожидания), что перед этой командой вставлять WAIT не нужно.

Система команд сопроцессора

Мнемоника	Операнды	Описание
Команды пересылки данных		
FLD	ST(i)	Загрузка новой вершины стека
	short_real	
	long_real	
	temp_real	

Мнемоника	Операнды	Описание
FBLD	bcd	
FILD	word_integer	
	short_integer	
	long_integer	
FSTP	ST(i)	Извлечение из стека
	short_real	
	long_real	
	temp_real	
FISTP	word_integer	
	short_integer	
	long_integer	
FBSTP	bcd	
FST	ST(i)	Копирование вершины стека
	short_real	
	long_real	
FIST	word_integer	
	short_integer	
FXCH	//ST(i)	Обмен вершины стека с заданным регистром стека
FLDLG2		Загрузка новой вершины стека константой lg2
FLDLN2		Загрузка новой вершины стека константой ln2
FLDL2E		Загрузка новой вершины стека константой $\log_2 e$
FLDL2T		Загрузка новой вершины стека константой $\log_2 10$
FLDPI		Загрузка новой вершины стека константой π
FLDZ		Загрузка новой вершины стека нулем
FLD1		Загрузка новой вершины стека единицей
Команды арифметических операций		
Базовые арифметические команды		
FADD	//ST, ST(i) / ST(i), ST	Сложение
	short_real	
	long_real	
FADDP	ST(i), ST	

Мнемоника	Операнды	Описание
FIADD	word_integer	
	short_integer	
FSUB	//ST, ST(i) / ST(i), ST	Вычитание
	short_real	
	long_real	
FSUBP	ST(i),ST	
FISUB	word_integer	
	short_integer	
FSUBR	//ST, ST(i) / ST(i), ST	
	short_real	
	long_real	
FSUBRP	ST(i), ST	
FISUBR	word_integer	
	short_integer	
FMUL	//ST, ST(i) / ST(i), ST	Умножение
	short_real	
	long_real	
FMULP	ST(i), ST	
FIMUL	word_integer	
	short_integer	
FDIV	//ST, ST(i) /ST(i), ST	Деление
	short_real	
	long_real	
FDIVP	ST(i), ST	
FIDIV	word_integer	
	short_integer	
FDIVR	//ST, ST(i) /ST(i), ST	
	short_real	
	long_real	
FDIVRP	ST(i), ST	
FIDIVR	word_integer	
	short_integer	
Дополнительные арифметические команды		
FABS		Нахождение модуля
FCHS		Смена знака
FSQRT		Извлечение квадратного корня
FRNDINT		Округление до целого
FPREM		Нахождение частичного остатка
FSCALE		Масштабирование

Мнемоника	Операнды	Описание
FXTRACT		Нахождение мантиссы и несмещенного порядка
Команды сравнения		
FCOM	// ST(i)	Сравнение
	short_real	
	long_real	
FCOMP	//ST(i)	Сравнение с извлечением из стека
	short_real	
	long_real	
FCOMPP		Сравнение с двойным извлечением из стека
FICOM	word_integer	Сравнение с целым операндом в памяти
	short_integer	
FICOMP	word_integer	Сравнение с целым операндом в памяти и извлечение из стека
	short_integer	
FCOMI	ST,ST(i)	Сравнение и установка флагов CF, ZF центрального процессора (начиная с Pentium II). Машинный код: db 0dbh, 0fih.
FCOMIP		Сравнение с извлечением из стека и установка флагов CF, ZF центрального процессора (начиная с Pentium II). Машинный код: db 0dfh, 0fih.
FTST		Сравнение с нулем
FXAM		Проверка содержимого st(0)
Трансцендентные команды		
FPATAN		Вычисление частичного арктангенса
FPTAN		Вычисление частичного тангенса
FCOS		Вычисление косинуса (начиная с 80387)
FSINCOS		Вычисление синуса и косинуса (начиная с 80387)
F2XM1		Вычисление функции $2^x - 1$
FYL2X		Вычисление функции $y \log_2 x$
FYL2XP1		Вычисление функции $y \log_2 (x+1)$

Мнемоника	Операнды	Описание
Команды управления		
FCLEX		Сброс флагов ошибок, флага запроса на прерывание и флага занятости в слове состояния
FNCLEX		
FDECSTP		Уменьшение на 1 указателя стека в слове состояния
FDISI		Запрещение прерываний в слове состояния
FNDISI		
FENI		Разрешение прерываний в слове состояния
FNENI		
FFREE	ST(i)	Освобождение регистра в слове тэгов
FINCSTP		Увеличение на 1 указателя стека в слове состояния
FINIT		Инициализация (сброс) сопроцессора
FNINIT		
FLDCW	word	Загрузка слова управления
FLDENV	address	Загрузка программной среды сопроцессора (всех регистров, кроме стековых)
FNOP		Пустая операция
FRSTOR	address	Загрузка полного состояния сопроцессора (всех регистров, включая стековые)
FSAVE	address	Сохранение полного состояния сопроцессора (всех регистров, включая стековые)
FNSAVE	address	
FSTCW	word	Сохранение слова управления
FSTENV	address	Сохранение программной среды сопроцессора (всех регистров, кроме стековых)
FSTSW	word	Сохранение слова состояния в памяти
FNSTSW	word	
FSTSW AX		Сохранение слова состояния в регистре AX центрального процессора (начиная с 80287)
FWAIT		Ожидание (альтернативная мнемоника команды центрального процессора WAIT)