Лекция 3

1.2.2. Контроллер прерываний

1.2.2.1. Механизм прерываний

Для обработки событий, происходящих асинхронно по отношению к выполнению программы, лучше всего подходит механизм прерываний. Прерывание можно рассматривать как некоторое особое событие в системе, требующее моментальной реакции. Например, хорошо спроектированные системы повышенной надежности используют прерывание по аварии в питающей сети для выполнения процедур записи содержимого регистров и оперативной памяти на магнитный носитель, с тем чтобы после восстановления питания можно было продолжить работу с того же места.

Кажется очевидным, что возможны самые разнообразные прерывания по самым различным причинам. Поэтому прерывание рассматривается не просто как таковое, с ним связывают число, называемое номером типа прерывания или просто номером прерывания. С каждым номером прерывания связывается то или иное событие. Система умеет распознавать, какое прерывание, с каким номером произошло и запускает соответствующую этому номеру процедуру.

Программы могут сами вызывать прерывания с заданным номером. Для этого они используют команду ***INT***. Это так называемые программные прерывания. Программные прерывания не являются асинхронными, так как вызываются из программы (а она-то знает, когда она вызывает прерывание!).

Программные прерывания удобно использовать для организации доступа к отдельным, общим для всех программ модулям. Например, программные модули операционной системы доступны прикладным программам именно через прерывания, и нет необходимости при вызове этих модулей знать их текущий адрес в памяти. Прикладные программы могут сами устанавливать свои обработчики прерываний для их последующего использования другими программами. Для этого встраиваемые обработчики прерываний должны быть резидентными в памяти.

Аппаратные прерывания вызываются физическими устройствами и приходят асинхронно. Эти прерывания информируют систему о событиях, связанных с работой устройств, например о том, что наконец-то завершилась печать символа на принтере и неплохо было бы выдать следующий символ, или о том, что требуемый сектор диска уже прочитан его содержимое доступно программе.

Использование прерываний при работе с медленными внешними устройствами позволяют совместить ввод/вывод с обработкой данных в центральном процессоре и в результате повышает общую производительность системы.

Некоторые прерывания (первые пять в порядке номеров) зарезервированы для использования самим центральным процессором на случай каких-либо особых событий вроде попытки деления на ноль, переполнения и т.п.

Иногда желательно сделать систему нечувствительной ко всем или отдельным прерываниям. Для этого используют так называемое маскирование прерываний, о котором мы еще будем подробно говорить. Но некоторые прерывания замаскировать нельзя, это немаскируемые прерывания.

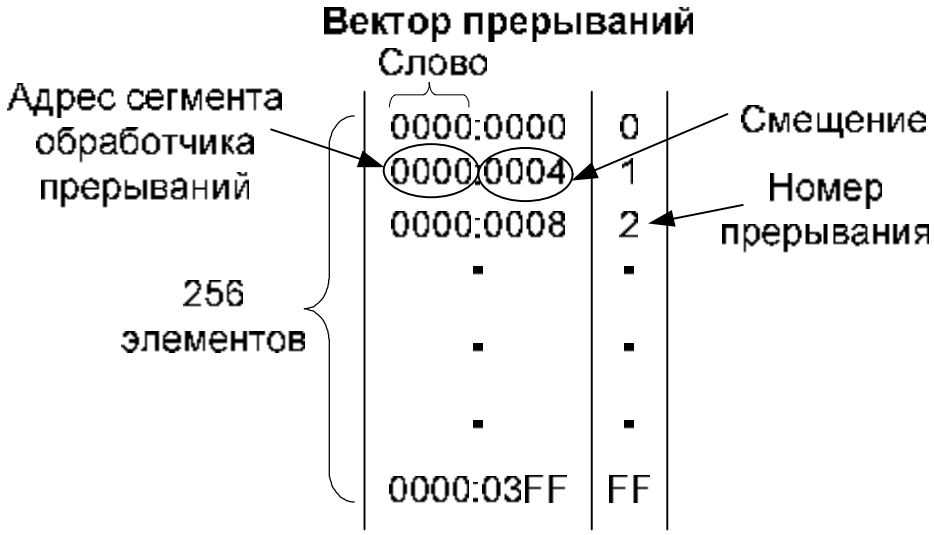
Заметим еще, что обработчики прерываний могут сами вызывать программные прерывания, например, для получения доступа к сервису BIOS или DOS (сервис BIOS также доступен через механизм программных прерываний).

1.2.2.2. Таблица векторов прерываний

Для того чтобы связать адрес обработчика прерывания с номером прерывания, используется таблица векторов прерываний, занимающая первый килобайт оперативной памяти - адреса от 0000:0000 до 0000:03FF. Таблица состоит из 256 элементов - FAR-

адресов обработчиков прерываний. Эти элементы называются векторами прерываний. В первом слове элемента таблицы записано смещение, а во втором - адрес сегмента обработчика прерывания.

Прерыванию с номером 0 соответствует адрес 0000:0000, прерыванию с номером 1 - 0000:0004 и т.д.



Инициализация таблицы происходит частично BIOS после тестирования аппаратуры и перед началом загрузки операционной системой, частично при загрузке DOS. DOS может переключить на себя некоторые прерывания BIOS.

Приведем назначение некоторых наиболее важных векторов:

Таблица

Назначение некоторых наиболее важных векторов

Номер	Описание
0	**Ошибка деления.** Вызывается автоматически после выполнения команд DIV или IDIV, если в результате деления происходит переполнение (например, при делении на 0). DOS обычно при обработке этого прерывания выводит сообщение об ошибке и останавливает выполнение программы. Для процессора 8086 при этом адрес возврата указывает на следующую после команды деления команду, а в процессоре 80286 - на первый байт команды, вызвавшей прерывание
1	**Прерывание пошагового режима.** Вырабатывается после выполнения каждой машинной команды, если в слове флагов установлен бит пошаговой трассировки TF. Используется для отладки программ. Это прерывание не вырабатывается после выполнения команды MOV в сегментные регистры или после загрузки сегментных регистров командой POP
2	**Аппаратное немаскируемое прерывание.** Это прерывание может использоваться по-разному в разных машинах. Обычно вырабатывается при ошибке четности в оперативной памяти и при запросе прерывания от сопроцессора
3	**Прерывание для трассировки.** Это прерывание генерируется при выполнении однобайтовой машинной команды с кодом CCh и обычно используется отладчиками для установки точки прерывания
4	**Переполнение.** Генерируется машинной командой INTO, если установлен флаг OF. Если флаг не установлен, то команда INTO выполняется как NOP. Это прерывание используется для обработки ошибок при выполнении арифметических операций
5	**Печать копии экрана.** Генерируется при нажатии на клавиатуре клавиши PrtScr. Обычно используется для печати образа экрана. Для процессора 80286

	генерируется при выполнении машинной команды BOUND, если проверяемое значение вышло за пределы заданного диапазона
6	Неопределенный код операции или длина команды больше 10 байт (для процессора 80286)
7	Особый случай отсутствия математического сопроцессора (процессор 80286)
8	***IRQ0*** - прерывание интервального **таймера**, возникает 18,2 раза в секунду
9	***IRQ1*** - прерывание от **клавиатуры**. Генерируется при нажатии и при отжатии клавиши. Используется для чтения данных от клавиатуры
A	***IRQ2*** - используется для каскадирования аппаратных прерываний в машинах класса AT
B	***IRQ3*** - прерывание асинхронного порта **COM2**
C	***IRQ4*** - прерывание асинхронного порта **COM1**
D	***IRQ5*** - прерывание от контроллера жесткого диска для XT
E	***IRQ6*** - прерывание генерируется контроллером флоппи-диска после завершения операции
F	***IRQ7*** - прерывание **принтера**. Генерируется принтером, когда он готов к выполнению очередной операции. Многие адаптеры принтера не используют это прерывание
10	Обслуживание **видеоадаптера**
11	**Определение конфигурации** устройств в системе
12	**Определение размера оперативной памяти** в системе
13	Обслуживание **дисковой** системы
14	Последовательный **ввод/вывод**
15	Расширенный сервис для AT-компьютеров
16	Обслуживание клавиатуры
17	Обслуживание принтера
18	Запуск BASIC в ПЗУ, если он есть
19	Загрузка операционной системы
1A	Обслуживание часов
1B	Обработчик прерывания Ctrl-Break
1C	Прерывание возникает 18.2 раза в секунду, вызывается программно обработчиком прерывания таймера
1D	**Адрес видео таблицы** для контроллера видеоадаптера 6845
1E	Указатель на таблицу параметров дискеты
1F	Указатель на графическую таблицу для символов с кодами ASCII 128-255
20-5F	Используется DOS или зарезервировано для DOS
60-67	Прерывания, зарезервированные для пользователя
68-6F	Не используются
70	***IRQ8*** - прерывание от часов реального времени
71	***IRQ9*** - прерывание от контроллера EGA
72	***IRQ10*** - зарезервировано
73	***IRQ11*** - зарезервировано
74	***IRQ12*** - зарезервировано
75	***IRQ13*** - прерывание от математического сопроцессора
76	***IRQ14*** - прерывание от контроллера жесткого диска
77	***IRQ15*** – зарезервировано
78 - 7F	Не используются
80-85	Зарезервированы для BASIC
86-F0	Используются интерпретатором BASI
F1-FF	Не используются

IRQ0-IRQ15 - это аппаратные прерывания, о них будет рассказано позже.

1.2.2.3.. Маскирование прерываний

Часто при выполнении критических участков программ для того, чтобы гарантировать выполнение определенной последовательности команд целиком приходится запрещать прерывания. Это можно сделать командой ***CLI***. Ее нужно поместить в начало критической последовательности команд, а в конце расположить команду ***STI***, разрешающую процессору воспринимать прерывания. Команда **CLI** запрещает только маскируемые прерывания, немаскируемые всегда обрабатываются процессором.

1.2.2.4. Изменение таблицы векторов прерываний

Вашей программе может потребоваться организовать обработку некоторых прерываний. Для этого программа должна переназначить вектор на свой обработчик. Это можно сделать, изменив содержимое соответствующего элемента таблицы векторов прерываний.

Очень важно не забыть перед завершением работы восстановить содержимое измененных векторов в таблице прерываний, т.к. после завершения работы программы память, которая была ей распределена, считается свободной и может быть использована для загрузки другой программы. Если вы забыли восстановить вектор и пришло прерывание, то система может разрушиться - вектор теперь указывает на область, которая может содержать что угодно.

Поэтому последовательность действий для нерезидентных программ, желающих обрабатывать прерывания, должна быть такой:

1. Прочитать содержимое элемента таблицы векторов прерываний для вектора с нужным вам номером;
2. Запомнить это содержимое (адрес старого обработчика прерывания) в области данных программы;
3. Установить новый адрес в таблице векторов прерываний так, чтобы он соответствовал началу Вашей программы обработки прерывания;
4. Перед завершением работы программы прочитать из области данных адрес старого обработчика прерывания и записать его в таблицу векторов прерываний.

Для облегчения работы по замене прерывания DOS предоставляет в ваше распоряжение специальные функции для чтения элемента таблицы векторов прерывания и для записи в нее нового адреса. Если вы будете использовать эти функции, DOS гарантирует, что операция по замене вектора будет выполнена правильно. вам не надо заботиться о непрерывности процесса замены вектора прерывания.

Для чтения вектора используйте функцию 35h прерывания 21h. Перед ее вызовом регистр **AL** должен содержать номер вектора в таблице. После выполнения функции в регистрах **ES:BX** будет искомый адрес обработчика прерывания.

Функция 25h прерывания 21h устанавливает для вектора с номером, который находится в **AL**, обработчик прерывания **DS:DX**.

Разумеется, вы можете непосредственно обращаться к таблице векторов прерываний, но тогда при записи необходимо замаскировать прерывания командой **CLI**, не забыв разрешить их после записи командой **STI**.

1.2.2.5. Особенности обработки аппаратных прерываний.

Аппаратные прерывания вырабатываются устройствами компьютера, когда возникает необходимость их обслуживания. Например, по прерыванию таймера соответствующий обработчик прерывания увеличивает содержимое ячеек памяти, используемых для хранения времени. В отличие от программных прерываний,

вызываемых запланировано самой прикладной программой, аппаратные прерывания всегда происходят асинхронно по отношению к выполняющимся программам. Кроме того, может возникнуть одновременно сразу несколько прерываний!

Для того, чтобы система "не растерялась", решая какое прерывание обслуживать в первую очередь, существует специальная схема приоритетов. Каждому прерыванию назначается свой уникальный приоритет. Если происходит одновременно несколько прерываний, то система отдает предпочтение самому высокоприоритетному, откладывая на время обработку остальных прерываний.

Таблица

Аппаратные прерывания, расположенные в порядке приоритета

Номер	Описание
8	IRQ0 прерывание интервального таймера, возникает 18,2 раза в секунду.
9	IRQ1 прерывание от клавиатуры. Генерируется при нажатии и при отжатии клавиши. Используется для чтения данных с клавиатуры.
A	IRQ2 используется для каскадирования аппаратных прерываний в машинах класса AT.
70	IRQ8 прерывание от часов реального времени.
71	IRQ9 прерывание от контроллера EGA.
72	IRQ10 зарезервировано.
73	IRQ11 зарезервировано.
74	IRQ12 зарезервировано.
75	IRQ13 прерывание от математического сопроцессора.
76	IRQ14 прерывание от контроллера жесткого диска.
77	IRQ15 зарезервировано.
B	IRQ3 прерывание асинхронного порта COM2.
C	IRQ4 прерывание асинхронного порта COM1.
D	IRQ5 прерывание от контроллера жесткого диска для XT.
E	IRQ6 прерывание генерируется контроллером флоппи диска после завершения операции
F	IRQ7 прерывание принтера. Генерируется принтером, когда он готов к выполнению очередной операции. Многие адаптеры принтера не используют это прерывание.

Из таблицы видно, что самый высокий приоритет у прерываний от интервального таймера, затем идет прерывание от клавиатуры.

Для управления схемами приоритетов необходимо знать внутреннее устройство контроллера прерываний 8259. Поступающие прерывания запоминаются в регистре запроса на прерывание ***IRR***. Каждый бит из восьми в этом регистре соответствует прерыванию. После проверки на обработку в настоящий момент другого прерывания запрашивается информация из регистра обслуживания ***ISR***. Перед выдачей запроса на прерывание в процессор проверяется содержимое восьмибитового регистра маски прерываний ***IMR***. Если прерывание данного уровня не замаскировано, то выдается запрос на прерывание.

1.3 Представление данных в ЭВМ

Данные, обрабатываемые МП 8086, можно разделить на три вида:

1. физические и логические коды;
2. числа с фиксированной точкой;
3. числа с плавающей точкой.

С точки зрения размерности (***физическая интерпретация***) микропроцессор аппаратно поддерживает следующие основные типы данных.

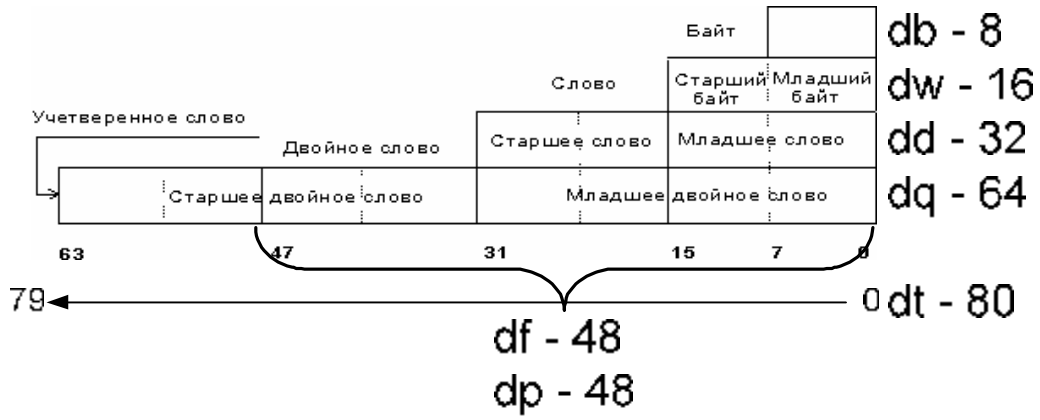
1.3.1. Типы данных

байт — восемь последовательно расположенных битов, пронумерованных от 0 до 7, при этом бит 0 является самым младшим значащим битом;

слово — последовательность из двух байт, имеющих последовательные адреса. Размер слова — 16 бит; биты в слове нумеруются от 0 до 15.

двойное слово — последовательность из четырех байт (32 бита), расположенных по последовательным адресам. Нумерация этих бит производится от 0 до 31. Адресом двойного слова считается адрес его младшего слова. Адрес старшего слова может быть использован для доступа к старшей половине двойного слова.

учетверенное слово — последовательность из восьми байт (64 бита), расположенных по последовательным адресам. Адресом учетверенного слова считается адрес его младшего двойного слова. Адрес старшего двойного слова может быть использован для доступа к старшей половине учетверенного слова.



С точки зрения их разрядности, микропроцессор на уровне команд поддерживает **логическую** интерпретацию:

1.3.2. Логические коды

Логическими кодами представляются: символы, числа без знака, битовые величины.

Все устройства ввода/вывода обмениваются данными из ЭВМ символами. Любой текст или число при этом рассматриваются как последовательность символов.

Каждый символ кодируется кодом ACSII (КОИ -- 7) и в памяти ЭВМ занимает один байт. Основная часть символов (латинские буквы, цифры, знаки препинаний) - 128 символов кодируется 7 битами, 8-й всегда равняется нулю. Остальные символы (кирилличные буквы и проч.) кодируются расширенной частью кода. (8-ю битами). Например, большая латинская буква А имеет код 65_{10}, буква С - 67_{10}. Потому АС будет иметь вид:

128	64	32	16	8	4	2	1	128	64	32	16	8	4	2	1
0	1	0	0	0	0	1	1	0	1	0	0	0	0	0	1
			С								А				

Цифра 0 имеет код 48_{10} = 30h:

0	0	1	1	0	0	0	0

Адрес ячейки памяти или байта является числом без знака, которое подается так же, как и числом с фиксированной точкой, только без знака.

При работе с внешними устройствами, превращении данных придется иметь дело с отдельными битами слова. В этом случае содержание ячейки рассматривается как определен код, или битовая величина.

1.3.3. Числа с фиксированной точкой

Целый тип со знаком — двоичное значение со знаком, размером 8, 16 или 32 бита. Знак в этом двоичном числе содержится в 7, 15 или 31-м бите соответственно. Ноль в этих битах в операндах соответствует положительному числу, а единица — отрицательному. Отрицательные числа представляются в дополнительном коде. Числовые диапазоны для этого типа данных следующие:

8-разрядное целое — от –128 до +127;

16-разрядное целое — от –32 768 до +32 767;

32-разрядное целое — от -2^{31} до $+2^{31}-1$.

Дополнительный код отрицательного числа формируется таким образом:

1. представить в соответствующей системе абсолютную величину числа N;
2. для каждого разряда взять его дополнение (если в двоичной системе, то просто заменить 1 на 0, и наоборот;);
3. прибавить единицу, пренебрегая при этом возможным перенесение из старшего разряда.

Например, число -1607_{10} будет иметь такой вид: -1607_{10}

1. $|N| = 0000011001000111$;
2. $\overline{N} = 1111100110111000$;
3. $\widetilde{N} = 1111100110111001$.

Целый тип без знака — двоичное значение без знака, размером 8, 16 или 32 бита. Числовой диапазон для этого типа следующий:

байт — от 0 до 255;

слово — от 0 до 65 535;

двойное слово — от 0 до $2^{32}-1$.

Цепочка (***байтовая строка***)— представляющая собой некоторый непрерывный набор байтов, слов или двойных слов максимальной длины до 4 Гбайт.

Битовое поле представляет собой непрерывную последовательность бит, в которой каждый бит является независимым и может рассматриваться как отдельная переменная. Битовое поле может начинаться с любого бита любого байта и содержать до 32 бит.

Неупакованный двоично-десятичный тип — байтовое представление десятичной цифры от 0 до 9. Неупакованные десятичные числа хранятся как байтовые значения без знака по одной цифре в каждом байте. Значение цифры определяется младшим полубайтом.

Упакованный двоично-десятичный тип представляет собой упакованное представление двух десятичных цифр от 0 до 9 в одном байте. Каждая цифра хранится в своем полубайте. Цифра в старшем полубайте (биты 4–7) является старшей.

Указатель на память двух типов:

ближнего типа — 32-разрядный логический адрес, представляющий собой относительное смещение в байтах от начала сегмента. Эти указатели могут также использоваться в сплошной (плоской) модели памяти, где сегментные составляющие одинаковы;

дальнего типа — 48-разрядный логический адрес, состоящий из двух частей: 16-разрядной сегментной части — селектора, и 32-разрядного смещения.

В памяти ЭВМ целое число представляются в двоичной системе исчисления.

Для удобства реализации арифметических операций целые числа внутри ЭВМ представляются в *дополнительном двоичном коде*.

Положительные числа имеют обычное двоичное представление и могут занимать одно слово или байт. Потому максимальным обычным числом будет 2^{15}-1=32767_{10}. В байт можно записать не больше, чем $2^7 - 1 = 127_{10}$.

Ясно, что в старшем разряде отрицательного числа всегда будет 1, а в положительных - 0. Поэтому этот разряд называется знаковым.

В дополнительном двоичном коде +0 и -0 представляются одинаково.

Обычные целые числа в памяти ЭВМ могут изменяться в диапазоне $-32768_{10} \le N \le 32767_{10}$

Использование дополнительного двоичного кода позволяет заменить операцию вычитания операцией добавления.

Кроме отмеченного представления целых чисел используется еще специальное представление десятичных чисел для двоично-десятичной арифметики. При этом одна десятичная цифра записывается в 8 (НЕупакованная форма) или 4 бита (упакованная форма) в двоичной форме.

1.3.4. Числа с плавающей точкой

В МП 8086 непосредственно не предусматривается действий с плавающей точкой.

Обычные десятичные числа можно записывать как число с порядком, например:

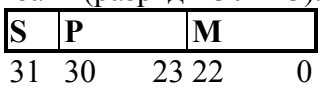
$1234.5 = 123.45*10 = 12.345*10^2 = 1.2345*10^3$

Такая форма выходит неоднозначной. Однако, если число изображать так, чтобы оно было меньше единицы и первая цифра за десятичной точкой была значимой (то есть не нуль), то такое представление будет однозначно и будет называться *нормализованным*. Нормализованное число в памяти подается *мантиссой и порядком*.

Для МП 8086 нормализованным в двоичной системе считается число, целая часть мантиссы которого равняется единице.

Такая форма удобна для компактного представления очень малых или очень больших чисел. Если же число имеет много разрядов, то сохранить их все не удается. Поэтому числа с плавающей точкой представляются в ЭВМ приближенно, а с фиксированной - точно.

В ЭВМ обычное вещественное число представляются в форме с плавающей точкой и занимает два слова. Старший бит первого слова является знаковым (S). Порядок (P) занимает байт (разряды 30 - 23). Мантисса (M) расположена в 23 битах.



Отметим особенности представления порядка и мантиссы числа. Чтобы не отводить один разряд под знак порядка, в эти разряды записывается не настоящий порядок, а со сдвигом $+127_{10}$.

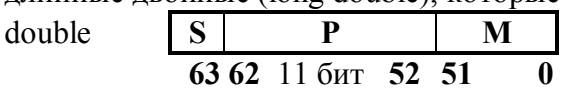
Поскольку мантисса числа записывается в нормализованном виде, то в двоичном представлении целая ее часть будет всегда равняться единице. Эту единицу в памяти НЕ записывают (сдвигают мантиссу на один разряд влево).

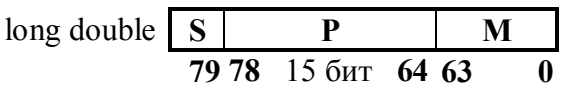
Например:

Число $15.375_{10} = 1111.011_2$.
В нормализованной форме оно будет иметь вид:
$1.111011*2^{11}{}_2$
Внутреннее представление числа:
$S = 0$; $P = 3 + 127 = 130_{10} = 10000010_2$; $M = 1110110\ldots0$.

В отличие от чисел с фиксированной точкой отрицательные вещественные числа отличаются от положительных лишь знаком.

Порядок числа определяет его диапазон, а мантисса - точность (количество значимых разрядов). Кроме обычных чисел можно использовать двойные (double) и длинные двойные (long double), которые имеют структуру:





В длинных двойных мантисса записывается полностью, а в двойных первая единица мантиссы сохраняется неявно.

Свойства чисел с плавающей точкой:

Тип	Размер (бит)	Диапазон		Точность десятичных розрядов
		Минимальный	Максимальный	
Float	32	$3.4*10^{-38}$	$3.4*10^{38}$	7
Double	64	$1.7*10^{-308}$	$1.7*10^{308}$	15
Long double	80	$3.4*10^{-4932}$	$3.4*10^{4932}$	19

В самом МП 8086 действий над числами с плавающей точкой не предусмотрено. Они выполняются в сопроцессоре 8087, 8187, 8287, или моделируются программно (эмулируются).