Программируемый таймер 8253.

Генерация звуков в компьютере PC выполняется с помощью программируемого таймера 8253, который применяется для управления колебаниями динамика. Управление колебаниями динамика определяется частотой, которая, в свою очередь, определяется содержимым различных внутренних регистров. Значения этих регистров устанавливаются при записи в определенные порты. Порт 66 используется для спецификации счетчика, который использует таймер при определении интервала колебаний динамика. Таймер работает в строгом соответствии с частотой системного таймера и специфицированным значением счетчика, определяющим колебания динамика. Затем, после обнуления счетчика происходит установка нового значения счетчика, и весь цикл функционирования программируемого таймера повторяется сначала. Значение счетчика определяется по следующей формуле:

count = 1,193,180/требуемая частота

где 1,193,180 есть тактовая частота системного таймера.

Регистр-счетчик таймера 8253 устанавливается в следующей последовательности (значение счетчика задается двухбайтным числом):

1. Выдать в порт 67 значение 182 (означающее, что будет устанавливаться счетчик).

2. Выдать в порт 66 младший байт числа, определяющего значение счетчика.

3. Выдать в порт 66 старший байт числа, определяющего значение счетчика.

Динамики большинства компьютеров класса PC не позволяют воспроизводить полный спектр частот, воспринимаемых человеческим слухом (от 20 Гц до 18.000 Гц). Однако динамик позволяет воспроизводить ноты лучше, чем динамики других компьютеров в пределах 12000 Гц и даже выше. В основном же динамик используется в пределах 100-5000 Гц.

Итак, таймер установлен. Однако динамик еще не будет воспроизводить звук, так как не включен. Таймер 8253 активен постоянно, а динамик требует дополнительной команды включения. Активизация динамика осуществляется путем установки значений битов 0 и 1 регистра программируемого периферийного интерфейса, задание значений которого выполняется через порт 97. Если значения этих двух битов установлены (равны 1), то динамик издает звук частотой, установленной счетчиком 8253. Если значения этих битов равны 0, то никакой звук генерироваться не будет. Остальные биты этого байта используются другими устройствами, поэтому интерпретация значения левых битов не может быть изменена. Таким образом, для установки значений управляющих динамиком бит

необходимо выполнить следующую последовательность действий:

1. Получить текущее значение регистра из порта 97.
2. Сравнить это значение с 3 или установить равным 3.
3. Записать результат в порт 97.

Для того, чтобы выключить динамик, необходимо переслать в порт значение 253.

Простейшим приемом, позволяющим читать и писать байт из или в порт, в Си является использование соответствующих функций. В Турбо Си - это функции inportb() и outportb(). В Microsoft Си - это функции inp() и outp(). Они имеют следующий общий формат:

```
int inportb(int port);
void outportb(int port, char value);
int inp(unsigned port);
int outp(unsigned port, int value);
```

В других компиляторах Си эти функции могут иметь иные названия, но обязательно будут присутствовать в вашей библиотеке, так как являются одними из базовых функций версий Си для ПЭВМ. В программах, приведенных в этом параграфе, используются функции Турбо Си.

Простейший способ проверки слуха.

Вы обладаете возможностью сделать несколько грубый, но эффективный тест слуха, который в состоянии обнаружить некоторые типы дефекта слуха. Как вы ранее узнали, динамик большинства компьютеров серии PC не воспроизводит звуки выше 12000 Гц. Однако ряд людей, у которых отмечены некоторые отклонения слуха, не могут услышать звук даже такой частоты. Фактически, тестируя свой слух, вы будете несколько удивлены тем, насколько высоким окажется звук с частотой 12000 Гц. (Предупреждение: тестирование слуха с помощью этого теста можно производить лишь ради шутки. Он, естественно, не позволяет действительно оценить слух испытуемого. Поэтому, если вы заметили у себя дефекты слуха или хотите действительно проверить свой слух, обратитесь лучше к своему врачу).

Для получения звука в тесте используется функция sound(), которая генерирует непродолжительное звучание специфицированной ноты. Как показано ниже, эта функция содержит все необходимое для того, чтобы сгенерировать любой звук с помощью динамика компьютера.

```
     /* Звучание динамика на заданной частоте */

void sound(freq)
int freq;
{
  unsigned i;
  union {
     long divisor;
     unsigned char c[2];
  } count;

  unsigned char p;

  count.divisor = 1193280 / freq; /* вычисление небходимого
                                     значения счетчика */
  outportb(67,182); /* обращение к таймеру 8253 после
                       установки счетчика */
  outportb(66,count.c[0]); /* пересылка младшего байта */
  outportb(66,count.c[1]); /* пересылка старшего байта */
  p = inportb(97); /* чтение существующего шаблона бит */
  outportb(97,p|3); /* установка битов 0 и 1 */

  for (i=0;i<64000;++i); /* цикл задержки */

  outportb(97,p); /* восстановление первоначального значения
                     шаблона бит для отключения динамика */
}
```

Заметим, что частота звучания ноты специфицирована как

аргумент функции. Цикл задержки необходим, так как без него вы бы услышали только мгновенный "щелчок" или "писк". Вы можете изменить частоту работы системного таймера процессора вашего компьютера. При этом, оформив его как параметр функции, вы добьетесь определенной эффективности вашей программы. Функция sound() может использоваться и для получения банального "пищания" компьютера.

Управляющая функция для программы теста слуха представлена ниже.

```
     /* Простейший тест слуха */

#include "dos.h"

void sound();

main()
{
     int freq;

     do {
```

```
            printf(" Введите частоту ( 0 - выход ): ");
            scanf("%d",&freq);
            if ( freq ) sound(freq);
          } while(freq);
      }
```

При использовании теста, в возрастающем порядке указывайте частоту звука до тех пор, пока звук воспринимается на слух. Для выхода введите 0.

Имитация звука сирены и взврывы.

Вы можете использовать возможность управления динамиком для создания различных звуковых эффектов, которые, в частности, делают видеоигры очень интересными и привлекательными. В основе всех звуковых эффектов лежит варьирование частоты звука - часто самым необычным образом.

Например, для создания эффекта звучания сирены вы должны варьировать частоту звука между двумя конечными точками. Высота звука должна изменяться от меньшей к большей, а затем уменьшаться от большей к меньшей. Функция siren(), представленная ниже, использует этот метод для создания эффекта звучания сирены.

```
#define DELAY 10000

/* Создание эффекта звучания сирены */
void siren()
{
  unsigned i,freq;
  union {
     long divisor;
     unsigned char c[2];
  } count;

  unsigned char p;

  p = inportb(97); /* чтение существующего шаблона бит */
  outportb(97,p|3); /* установка бит 0 и 1 */
     /* повышение звука сирены */
  for (freq = 1000;freq<3000;freq+=RATE) {
     count.divisor = 1193280 / freq; /* вычисление нужного
                                        значения счетчика */
     outportb(67,182); /* обращение к таймеру 8253 после
```

```
                                определения значения счетчика */
        outportb(66,count.c[0]); /* пересылка младшего байта */
        outportb(66,count.c[1]); /* пересылка старшего байта */

        for (i=0;i1000;freq-=RATE) {
        count.divisor = 1193280 / freq; /* вычисление нужного
                                           значения счетчика */
        outportb(67,182); /* обращение к таймеру 8253 после
                                определения значения счетчика */
        outportb(66,count.c[0]); /* пересылка младшего байта */
        outportb(66,count.c[1]); /* пересылка старшего байта */

        for (i=0;i1000;freq-=RATE) {
        count.divisor = 1193280 / freq; /* вычисление нужного
                  значения счетчика */
        outportb(67,182); /* обращение к таймеру 8253 после
                  определения значения счетчика */
        outportb(66,count.c[0]); /* пересылка младшего байта */
        outportb(66,count.c[1]); /* пересылка старшего байта */

        for (i = 0;i5000); /* после персонального
                                      прослушивания */
             sound(freq);
        } while (!kbhit());
}

/* звучание динамика на специфицированной частоте */
void sound(freq)
int freq;
{
  unsigned i;
  union {
     long divisor;
     unsigned char c[2];
  } count;

  unsigned char p;
  count.divisor = 1193280 / freq; /* вычисление нужного
                                     значения счетчика */
     outportb(67,182); /* обращение к таймеру 8253 после
                          определения значения счетчика */
     outportb(66,count.c[0]); /* пересылка младшего байта */
     outportb(66,count.c[1]); /* пересылка старшего байта */

     p = inportb(97); /* чтение существующего шаблона бит */
     outportb(97,p|3); /* установка бит 0 и 1 */
```