PC Speaker

From OSDev Wiki

The PC Speaker is the most primitive sound device available on PC compatible systems. It is characterized by the distinctive "beeps" and "squeaks" that it can be made to produce and is therefore sometimes referred to as the "PC Beeper" or the "PC Squeaker".

Contents

The Raw Hardware

The speaker itself has two possible positions, "in" and "out". This position can be set through bit 1 of port 0x61 on the Keyboard Controller. If this bit is set (=1), the speaker will move to the "out" position, if it is cleared (=0) then the speaker will move to the "in" position. It is important to note that the speaker will not instantly jump to the desired position. Rather it will move towards the desired position ("in" or "out") over a short period of time. This allows the speaker to produce actual beeps and whistles instead of just clicks.

Modes of Operation

Through the Programmable Interval Timer (PIT)

The PC Speaker can be connected directly to the output of timer number 2 on the Programmable Interval Timer by setting bit 0 of port 0x61 (=1). In this mode, when the timer goes "high" (=1) the speaker will move to the "out" position. Likewise, when the timer goes "low" (=0) the speaker will move to the "in" position. By changing the frequency at which timer 2 "ticks", the PC Speaker can be made to output sound of the same frequency. This mode is very popular because it is easy to program and because it is asynchronous from the rest of the computer's operation, meaning that it takes very little CPU time. It should also be noted that this is the "official" way to program the PC Speaker and, if a sound card is present, should be the only way that the PC Speaker is programmed.

Pulse Width Modulation

In the absence of a real sound card, the PC Speaker can be used to output low quality digital sound. As stated earlier, the speaker itself has only two possible positions, "in" and "out". More positions are needed though in order to play digital sound. Typically 256 positions (8 bits) are considered adequate to play comprehensible audio. One popular method used by many early PC games to overcome this limitation is called pulse-width modulation. This technique uses the physical properties of the speaker to allow it to output the (relatively) complex sounds that exist in 8-bit audio.

How It Works

The PC Speaker takes approximately 60 millionths of a second to change positions. This means that if the position of the speaker is changed from "in" to "out" and then changed back in less than 60 milliseconds, the speaker did not have enough time to fully reach the "out" position. By precisely adjusting the amount of time that the speaker is left "out", the speaker's position can be set to anywhere between "in" and "out", allowing the speaker to form more complex sounds.

Setting It Up

You should first have the PIT interrupt you each time the speaker's position needs to be changed. To play standard audio, the CPU needs to be interrupted 44100 times every second. The easiest way to do this is to (re)program timer zero (0) to interrupt you at 44100Hz.

You also need a way to actually change the speaker's position. This could be done simply through bit 1 of port 0x61 but this option is too slow to be practical. Instead, it is best to (re)program timer 2 to properly set the position of the speaker for you while you wait for the next interrupt (or do other, more useful things with any spare CPU time).

Playing the Audio

When the PIT interrupts you you need to program PIT timer 2 so that it puts the PC Speaker in the "out" position for a fraction of 60 milliseconds. For 8 bit audio the number of milliseconds can be calculated like this:

```
milliseconds = (sample * 60) / 255;
```

For more information see the Wikipedia article Pulse-width modulation. (http://en.wikipedia.org/wiki/Pulse-width_modulation) There is also sample code posted in the forum here (http://forum.osdev.org/viewtopic.php?f=13&t=17293) .

Sample Code

This will work on real hardware, but may not on emulators. The code also changes the PIT timer 2 frequency, so you will have to reset that when you're done "beep"ing :)

```
//Play sound using built in speaker
static void play_sound(uint32_t nFrequence) {
       uint32_t Div;
       uint8_t tmp;

```

```
        //Set the PIT to the desired frequency
        Div = 1193180 / nFrequence;
        outb(0x43, 0xb6);
        outb(0x42, (uint8_t) (Div) );
        outb(0x42, (uint8_t) (Div >> 8));

        //And play the sound using the PC speaker
        tmp = inb(0x61);
        if (tmp != (tmp | 3)) {
                outb(0x61, tmp | 3);
        }
}

//make it shutup
static void nosound() {
        uint8_t tmp = inb(0x61) & 0xFC;

        outb(0x61, tmp);
}

//Make a beep
void beep() {
         play_sound(1000);
         timer_wait(10);
         nosound();
          //set_PIT_2(old_frequency);
}
```

Retrieved from "http://wiki.osdev.org/index.php?title=PC_Speaker&oldid=17168"
Category: Sound

- This page was last modified on 1 December 2014, at 08:51.
- This page has been accessed 32,728 times.