Sampling IR remote controls with a sound card

Oded Noam oded@no.am

Disclaimer

The technique proposed in this document is the one I used to sample the signals of my remote controls, and it worked for me without causing any damage. However, this is just my hobby, and I give this information free of charge and allow it to be distributed; therefore I cannot be liable for any damages caused to people or equipment or anything at all, due to usage of the proposed technique. I am not an electrical engineer thus I may not even be aware to all risks involved; in other words – use at your own risk.

What you need

- 1 photoresistor (almost any photoresistor will do, I used a 13-900KΩ resistor)
- 1 AA (1.5v) battery
- 1 mini-PL (3.5) audio plug
- 1 computer
- 1 simple audio card (this will not work with DSP or 3D audio cards, such as “Creative SBLive!” and compatibles)
- 1 waveform editing software (such as CoolEdit, SoundForge etc.)

How and why it works

This sampler can sample coding made by an IR remote control, but it cannot sample modulation. That is because modulation frequency is at around 40KHz and sound cards can only sample waves of up to 22 or 24 Khz (sample rate must be at least twice the sampled waveform frequency).

But, if you know the modulation frequency of your IR remote (on most remote controls, you can assume it is 40KHz), all you need to sample is the coding of the remote.

How IR coding works

A remote control send commands as a series of modulated signals and stops. Unlike other

communications mechanisms, its signal can only exist or not exist (either it sends a 40KHz wave or not), and the signal content is determined by the lengths of the signal and the length of the stop. Every remote control has a method of converting digital instructions into a long series of signals and stops; hence – the coding.

How sound cards work

Sound is physically vibrations of the air.

An analog audio signal, as can be created by a microphone, for example, is simply a change in voltage that represents the movements of the air. When the air pressure rises, positive voltage will be created by the microphone; when it drops, negative voltage will be created. When its rise is steep, the voltage will be higher.

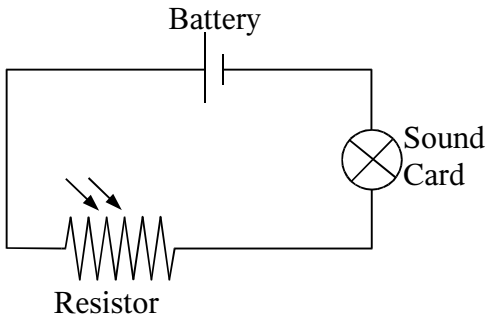
A sound card simply converts this electronic representation of sound – by measuring the line current – to digital values. If we sample the line current many times and do this conversion, we will get digital representation of the sound wave.

How can a sound card sample IR remote control

A sound card is designed to sample sound waves; however simple sound cards do not care whether they sample sound waves or simply line current. If we create a circuit in which line current represents something other than sound, most sound cards can sample it just as easily.

So, we create a circuit that converts light (including infrared light) into voltage (and current) changes. We use a battery to give constant voltage, that flows through a resistor and the sound card. The resistor changes its resistance when it is hit by light, thus changing the current being sampled at the sound card.

Notes for building the circuit



Use a power cord or a telephone cord to connect the resistor to the other components; you will need some distance in order to place the resistor conveniently in front of the remote control being sampled.

1. Use a mini-PL plug to connect to your sound card. You can connect either to the audio-in or the microphone socket; you will probably get better samples using the microphone port.
2. Polarity doesn't matter – so you cannot connect the battery, the resistor or the sound card incorrectly.
3. If you use stereo (3-connectors) mini-PL plug, connect its innermost connector and its outermost (shield) connector. Connecting only the two inner connections will not work

4. Don't try and replace the battery with a transformer or another power supply; it will create noise that will affect the sample.

Sampling

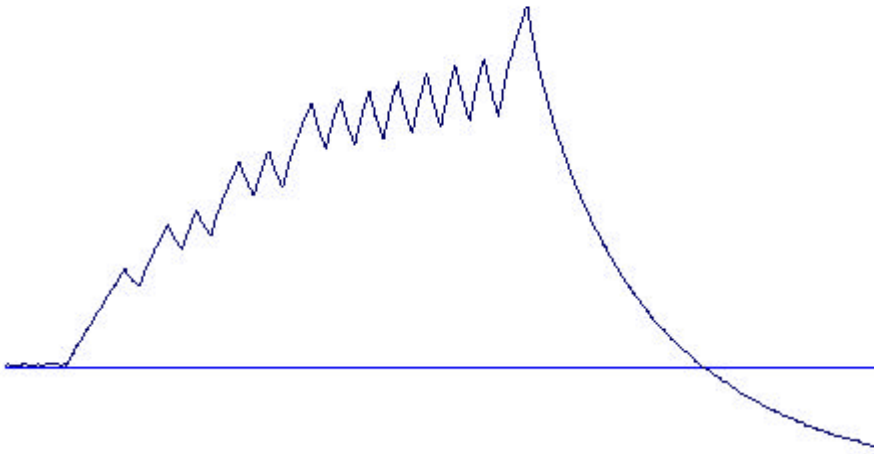
After you have everything connected, use your sound editing software to do the sample. Since the resistor reacts to any light source (not only infrared), you will need to do it in the dark – no windows, no lamps, not even your computer monitor. Practice using only the keyboard to do the recording, and turn off the screen before actually sampling.

Now that you have the sample, chances are your sound editing software will show you no signal. That is because the resistance changes are very subtle; you will need to zoom in in order to actually see a sample.

Reading the sample

The sound card samples the derivative of the voltage. Therefore, the waveform you receive will have sections going up and sections going down; these represents signal and no signal. Depending on the battery polarity, “up” can mean signal or no signal, and “down” can mean no signal or signal.

Most sound editing software will show you the length in milliseconds of selected range on the waveform; all you have to do now is calculate the lengths of the signal and no-signal segments.



Example: Reading the sample of the “power” button in Sony RM-SMD1 Remote Control

As we can see, before the sample the line level is around 0. There is no signal at this point; therefore the first change is when there is signal – and the change is upwards. The line level goes upward for about 2.4 milliseconds, then goes down for 0.6 milliseconds. This means the IR signal was on for 2.4 milliseconds and then off for 0.6 millisecond. The whole signal was:

Signal length ***(millisecond)***	***No-signal length*** ***(millisecond)***	***Suggested Meaning***
2.4	0.6	START signal
1.2	0.6	'1' bit
0.6	0.6	'0' bit
1.2	0.6	'1' bit
0.6	0.6	'0' bit
1.2	0.6	'1' bit
0.6	0.6	'0' bit
0.6	0.6	'0' bit
0.6	0.6	'0' bit
0.6	0.6	'0' bit
0.6	0.6	'0' bit

Signal length ***(millisecond)***	***No-signal length*** ***(millisecond)***	***Suggested Meaning***
0.6	0.6	'0' bit
1.2	0.6	'1' bit

Reading the whole signal, we got: (start)-101010000001-(end).
This is a 12 bit command coded in Sony SIRCS protocol. In SIRCS, the signal bits are coded backwards; 5 bits are for device-ID and 7 bits are for the command. Here, the device-id is 0b10000 (16 decimal, meaning “system, cassette deck or tuner”), and the command is 0b0010101 (21 decimal, meaning “power toggle”).

Attention SBLive! Users

The first time I built this circuit, I tested it on my SBLive! card, and to my great disappointment, I got this sample:

As it turns out, the DSP on the SBLive! expects audio signal, and recreates the waveform by analyzing its frequencies. The signal I created was not sound, thus re-creating its waveform ruined the whole sample. Luckily I had another sound card, of simpler kind, which worked just fine.