In various forms, 3-D printing has been part of manufacturing for decades. Most 3-D printers apply material—often molten plastic—layer by layer to create prototypes or even finished parts. But these industrial machines tend to be expensive and large. Even a basic model can cost upwards of US $100 000 and be the size of a refrigerator, Pettis says, well out of reach for hobbyists.

With some angel investing, Pettis and his friends introduced MakerBot Industries' first printer kit in March 2009. The tabletop device could make small structures out of plastic—or pretty much any material that could be made to flow steadily through a nozzle onto a moving platform. "We created a tool head for the machine [we] called a frostruder, which was an extruder for frosting," says Pettis. "It worked really well with Nutella." The kits started selling briskly. What started as a small hacking project has ballooned over the past four years into a small empire, fed in part by Pettis's energetic media presence and his plethora of photogenic printed plastic gewgaws.

Pettis now oversees more than 250 employees at MakerBot's New York City headquarters and production facility. The company has shipped nearly 20 000 printers (now as finished units instead of kits), and in November 2012, it opened its first retail store. Earlier this year, Pettis debuted a 3-D laser scanner that could be used to map physical objects, creating a digital model suitable for printing. That's in addition to the company's website, Thingiverse.com, a compendium of user-submitted files that has expanded to include more than 80 000 designs for 3-D printed objects.

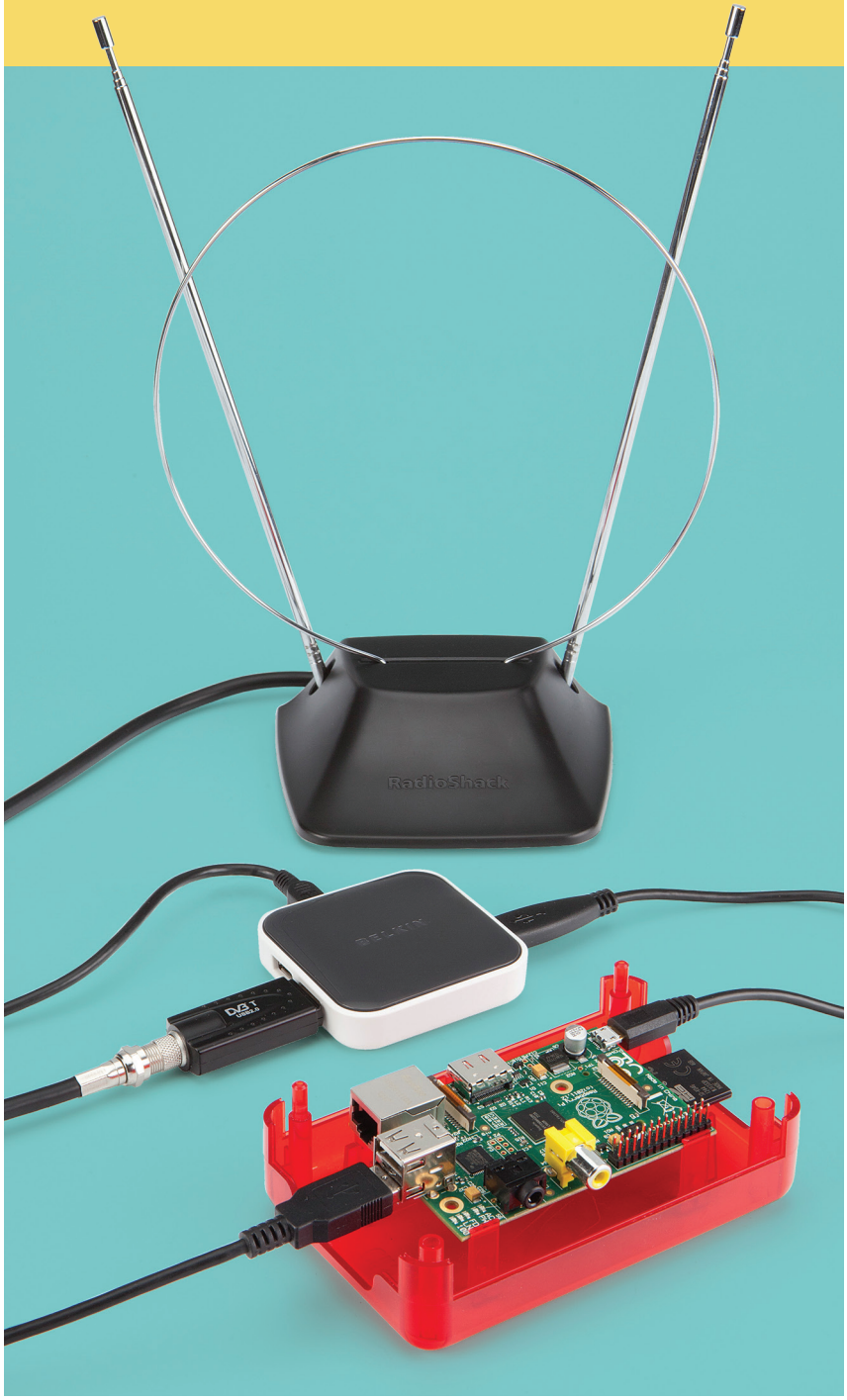
Pettis got his start as an art teacher in Seattle before leaving to make how-to videos for *Make* magazine and the craft-selling site Etsy. He sees few limits to the market for 3-D printers. In 5 years, he says, you'll know someone who has a 3-D printer; in 10, you'll likely have access to one—if not in your home, then at the local library.

The increasing availability of 3-D printers has already turned up some surprises. In May, for example, a project called Robohand uploaded designs to the Thingiverse for 3-D printed fingers, which could be used to build inexpensive but functional pulley-driven prosthetics. "We're going to see all sorts of superinteresting things happen as the technology shifts from being something just for the elite to something that's the price of a laptop," Pettis says.

—RACHEL COURTLAND

A $40 SOFTWARE-DEFINED RADIO

A REPURPOSED TV TUNER CAN REVEAL A WIDE SWATH OF SPECTRUM



PHOTOGRAPH BY NICHOLAS EVELEIGH

The last time I ventured into the waters of software-defined radio (SDR) was seven years ago, when I reviewed Matt Ettus's Universal Software Radio Peripheral. While it's an excellent product, the basic motherboard at the time cost US $550; daughterboards for different frequency ranges cost $75 to $275 [see "Hardware for Your Software Radio," *IEEE Spectrum*, October 2006]. And I spent more than a few frustrating hours compiling the needed software on my MacBook Pro. This time I was able to get my feet wet for about $40—and the software took about 2 minutes to download, install, and run.

This minor miracle was made possible by Finnish engineering student and Linux developer Antti Palosaari. Last year, he discovered an unexpected feature of the RTL2832U demodulator chip made by Taiwan's Realtek: Intended for decoding European HDTV broadcasts in inexpensive USB dongle-type receivers, the RTL2832U chip can also output a raw digital stream describing the amplitude and phase (so-called I/Q data) of signals over a wide range of frequencies.

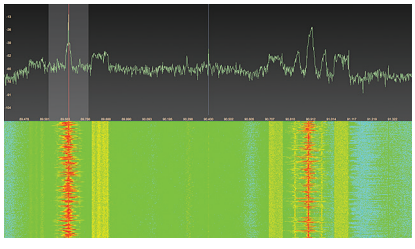
Digital radio enthusiasts immediately began adapting open-source tools that can translate I/Q information into audio and data streams. The result is a low-cost SDR that can pick up a huge variety of transmissions with different modulation schemes, including stereo FM from broadcasters, digital data packets from aircraft transponders, and suppressed-sideband (SSB) dispatches from amateur radio operators. Of course, the system isn't as sensitive as purpose-built SDRs and is incapable of transmitting a signal, but it's enough to see what's going on across a huge chunk of spectrum.

Different receiver dongles pair the RTL2832U with different radio tuners, so the exact range of frequencies that can be received varies. I purchased a Freeview P250 dongle from a Chinese supplier on Amazon.com for $20, which included shipping. The P250 combines the RTL2832U with an Elonics E4000 tuner, allowing it to pick up signals from about 52 megahertz to 2.2 gigahertz, with a gap from about 1.1 to 1.25 GHz.

A small antenna came with the receiver, but I replaced it with a set of $15 rabbit ears from RadioShack. An adapter to connect the rabbit ears' U.S. coaxial cable to the dongle's European socket cost a couple of dollars.

To use the receiver with my MacBook Pro, I downloaded Elias Önal's port of the Gqrx software receiver. Önal's port is precompiled for OS X, so installation was simply a matter of downloading it to my hard disk. The application automatically detected my receiver, and I was off.

Gqrx centers on an oscilloscope-like display, showing a slice of the radio spectrum (along with a waterfall-type display beneath that tracks the last 30 seconds or so). Gqrx allows you to set how wide the slice should be, from 1 to 2.4 MHz. You select the frequency that's passed to the software demodulator by clicking the mouse on that frequency in the oscilloscope display. Demodulation modes include AM, narrowband FM, mono and stereo FM, SSB, and continuous wave (used for Morse code).



WATCHING RADIO: This screenshot from Gqrx shows two FM stations. The central spikes are the analog stereo broadcast, while the squared-off signals on either side are digital radio transmissions. You can tune in and listen to a station by clicking on its center frequency. The gray stripe indicates the bandwidth of the user-selected software demodulator.

Because the receiver can see so much spectrum at once, you can use it to monitor activity on many channels simultaneously. For example, in Boston, where I live, there are 17 narrowband-FM police channels between 460.025 and 460.500 MHz, covering various districts, et cetera. A spike on the display shows when any of those channels is in use, and a click of the mouse has its audio playing over my speakers.

Which brings us to regulatory issues. In some countries, it's illegal to receive any frequency you don't have a license for, apart from public broadcast frequencies. In the United States, you're free to pick up nearly all the signals you can receive. There are, however, important exceptions to this general rule, such as a ban on listening to cellphone frequencies, or operating equipment capable of picking up police signals while you're in a vehicle (the latter is permitted with a ham license).

I soon discovered that having the dongle and TV antenna attached to my laptop is cumbersome, and besides, my home office doesn't always get great reception. So I spent another $35 and purchased a Model B Raspberry Pi microcontroller [see "The Gift Guide: Basic Bytes," *Spectrum*, December 2012].

The Raspberry Pi is an ARM-based, Ethernet-capable microcontroller with USB connectors that can run a number of variants of Linux. Following instructions on the Ham Radio Science website (http://www.hamradioscience.com), I was able to download and compile some support software to use the Pi with the dongle (connected via a powered USB hub) in about 30 minutes. In turn, I connected the Pi to the home network hub in my front room via an Ethernet cable. Using the Pi lets me place the receiver farther away from local radio sources (such as my hub's Wi-Fi transmitter) and also allows multiple machines to access the receiver easily; the Pi acts as a centralized SDR server, thanks to a command-line utility called rtl_tcp.

With the Pi running, I was able to call up Gqrx on my Wi-Fi-connected MacBook, feed it the Pi's network address, and then control and decode signals as if the tuner were plugged directly into the laptop. (Admittedly, this represents a degree of engineering overkill when it comes to listening to a local FM station.)

Now that I've got the basic system up and running, I'd like to extend the bottom of my receiver's range to longer wavelength bands, such as the popular amateur 20-meter band between 14.00 and 14.35 MHz. This will require either modifying the dongle or buying or building a frequency converter. But either approach will be pointless unless I swap my rabbit ears for a long-wavelength antenna, which is a whole other kettle of home-brew fish. —STEPHEN CASS