Software-defined radios don't necessarily have to come in commercially produced packages. If you're handy with a soldering iron, and especially if you have software skills that can contribute to the state of the art in SDR, the OpenHPSDR Project has a place—and a radio—for you!

High-Performance Software-Defined Radio for Experimenters

BY JEREMY McDERMOND,* NH6Z

Software Defined Radio, or SDR, has become of increasing interest to amateur radio in recent years. There has been an explosion of manufacturers offering SDR products such as the Elecraft K3, the Flex Radio 5000 (and siblings), and the RFSpace SDR-IQ. But while there has been some activity around homebrew experimenter SDR platforms that use standard computer sound cards such as Tony Parks's SoftRock kits, there has not been much progress on a truly high-performance software-defined radio (HPSDR) for experimenters. The OpenHPSDR project is an attempt to provide such a platform to allow homebrewers a venue to push the state-of-the-art in software defined radio for hams.

The OpenHPSDR Project

The project is organized as a loose conglomeration of individuals who each contribute their talents. This is similar to many open-source software projects on the internet. The members of the project are committed to providing both open-source hardware and software to the SDR community. Hardware designs generally are provided under the TAPR Open Hardware License[1] which makes schematics, block diagrams, and manufacturing files publicly available. Firmware files and most of the software packages are available under open-source licenses such as the GNU Public License (GPL).

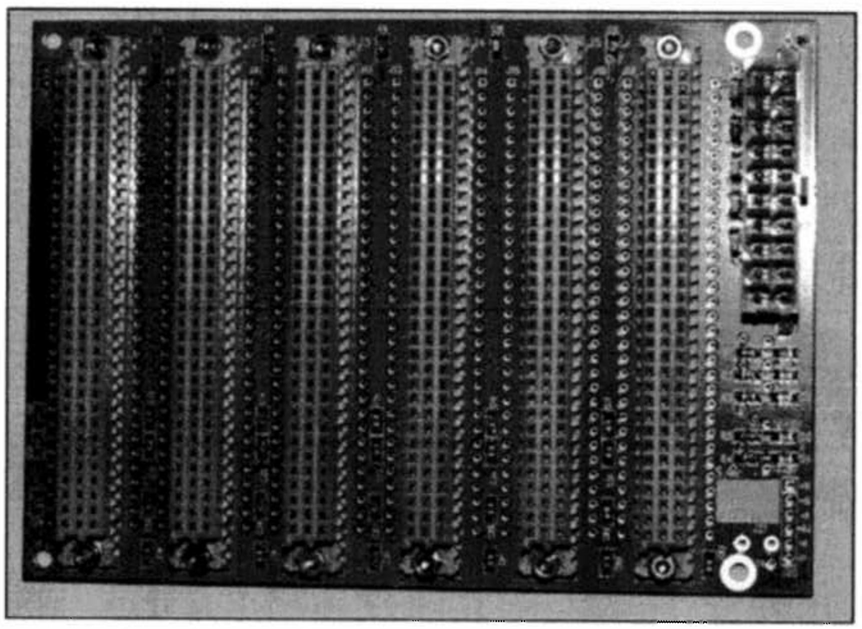

Photo A– The Atlas backplane board, into which all other modules of the OpenHPSDR radio connect. (Photos courtesy the OpenHPSDR Project)

Project members are supported by TAPR (Tucson Amateur Packet Radio) and AMSAT (the Radio Amateur Satellite Corp.). AMSAT has provided a plethora of design resources such as PCB layout programs and development environments to assist project leaders in their hardware designs. TAPR provides seed capital to allow project leaders to acquire development boards, prototype new designs, and prepare boards for release to a wider audience. However, because one-off prototypes don't necessarily contribute greatly to the state-of-the-art of SDR, TAPR also helps the project manufacture limited runs of production boards for sale to the ham community at large. This puts these boards into the hands of software developers and others who can contribute their skills to advance the project's goals.

Hardware

There are two generations of OpenHPSDR hardware available. The Atlas[2]-based systems were the original designs which are intended to provide a maximum amount of flexibility and extendibility. Project hardware designers then used the knowledge gained

*2171 NW Sunview Drive, Corvallis, OR 97330
e-mail: <nh6z@nh6z.net>*

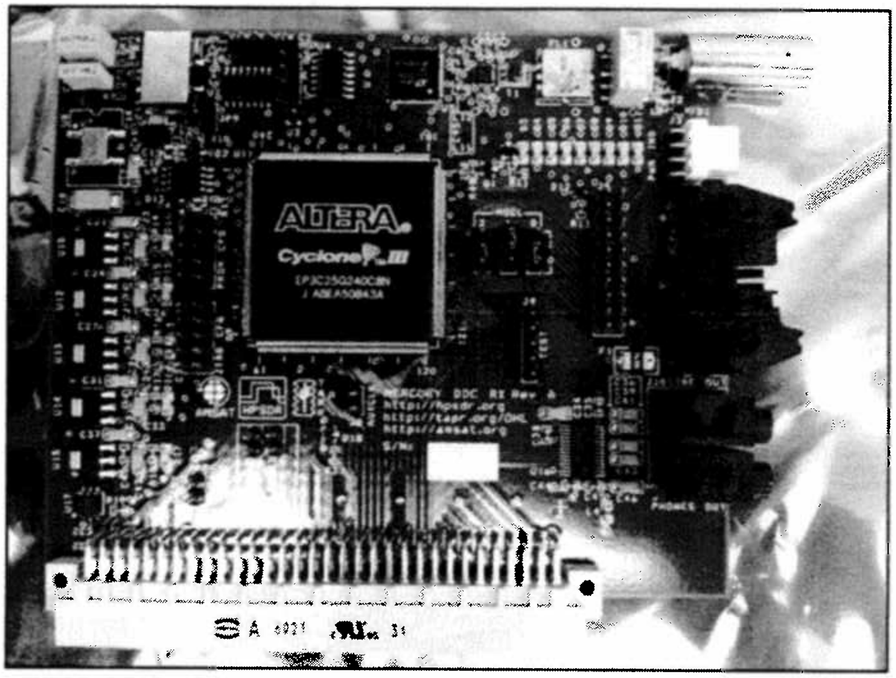

Photo B– An OpenHPSDR Mercury receiver board. The Altera Cyclone FPGA is in the center and the Linear Technologies ADC is at the top center.

from the Atlas-based systems to create a second generation SDR called Hermes. Hermes has not deprecated Atlas, as each platform has its respective strengths and weaknesses.

The Atlas system is based around a central passive backplane called, appropriately, Atlas, since it supports all the other modules (see photo A). It has an ATX-style connector for power and a set of connectors to hold additional cards. The cards provide the fundamental capabilities of the SDR such as receiving, transmitting, and communications with the computer.

Mercury (photo B) is the receiver card that plugs into the Atlas bus. It contains a Linear Technologies LTC2208 analog-to-digital converter (ADC) chip that is capable of 130-million 16-bit samples per second. This allows the Mercury receiver to use direct down conversion (DDC) to continuously sample the entire RF spectrum from DC to 65 MHz. Since computer interfaces to transfer this amount of data are prohibitively expensive, Mercury also contains an Altera Cyclone III field programmable gate array (FPGA) to filter and decimate the samples to an appropriate data rate and format for the computer to process. As noted above, all of this FPGA source code is available under an open-source license, allowing others to modify the firmware to expand its uses. Project members have used this capability to

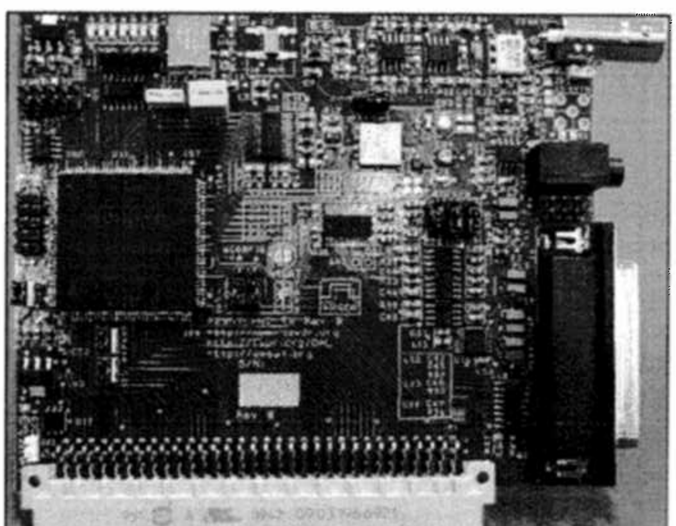

Photo C– The new Pennylane transmitter board uses knowledge gained from the Penelope transmitter project to provide greater capabilities.

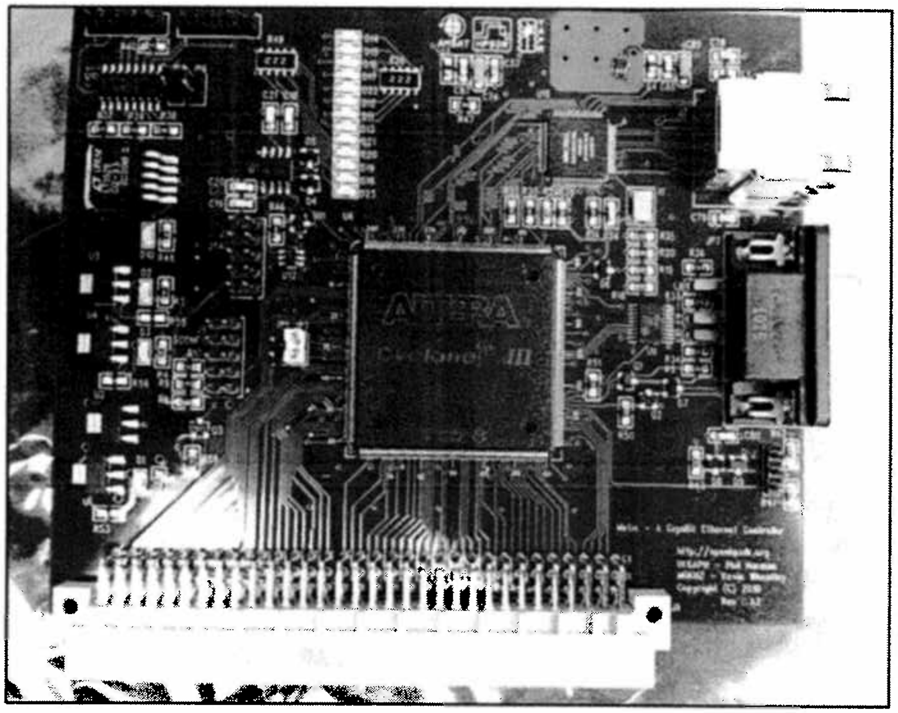

Photo D– The Metis computer interface board. The Micrel Gigabit Ethernet PHY is to the top right of the Altera FPGA.

implement capabilities such as a self-contained receiver in Mercury's FPGA.

The second important component in the Atlas systems is the Penelope transmitter board. It is based around an Analog Devices AD9744 digital-to-analog converter (DAC), which is a 14-bit device that is clocked at 122.88-million samples per second to match Mercury. This essentially makes Penelope a mirror image of Mercury's capabilities. This wide-band capability has been used to allow a Penelope to act as a WSPR[3] beacon on multiple bands simultaneously. Penelope also has an Altera Cyclone FPGA to provide sample processing before being fed to the DAC.

There is now an updated version of Penelope called PennyLane (photo C) which uses knowledge gleaned from the Penelope project to improve the capabilities of the board. Both Penelope and PennyLane produce 0.5W output and are very clean transmitters. Wide-band spurious emissions are over 50 dB down from the carrier.

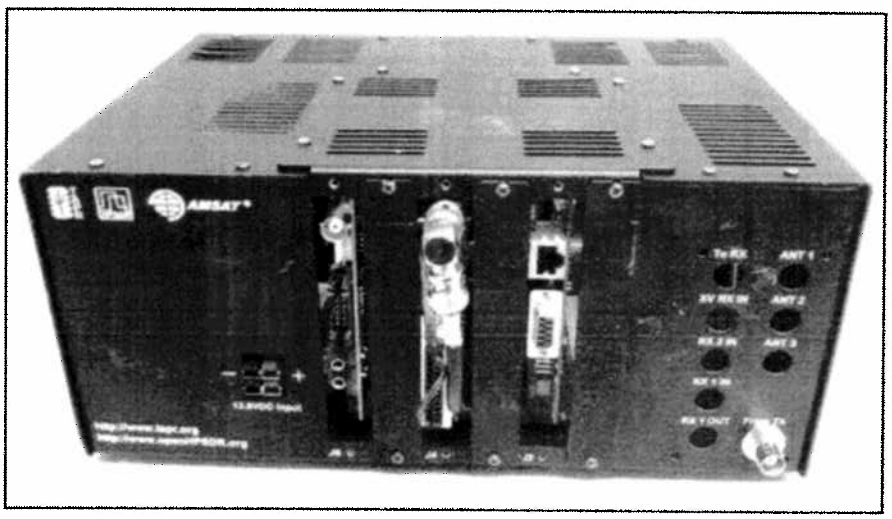

Photo E– An assembled OpenHPSDR transceiver in a Pandora case. The boards are, from left to right: Mercury, Penelope and Metis. Inside the case are an LPU and Pennywhistle (see text for details).

The final critical component in an Atlas system is the interface to the computer. The original computer interface—called Ozymandias, or Ozy—was based on USB 2.0 technology using the Cypress FX2 USB chip. While USB was an adequate technology for a first-generation computer interface, limitations were quickly encountered. The speed of USB is not particularly fast and would not allow for increased bandwidths and capabilities to push the limits of what was possible with software-defined ham radios. Additionally, USB usually requires drivers that would have to be written for every operating system and supported by the project. These drivers were also problematic because of the requirements of 64-bit Windows® systems for signed drivers which were difficult for an open-source project such as OpenHPSDR to produce.

For these reasons the project decided to pursue a different technology for connectivity to the computer. The current generation interface is called Metis (photo D) and provides a full-duplex 1-Gigabit per second ethernet connection to the computer. Metis uses a Micrel KSZ9021RL ethernet PHY and an Altera Cyclone III FPGA. The large FPGA allows for plenty of space for future expansion of code. The board's firmware has been programmed to use a standard UDP/IP packet format and supports configuration through familiar tools such as DHCP and Link Local addressing. While not supporting a full TCP/IP stack, the firmware allows for a variety of network configurations to fit most operators' needs.

The decision to use ethernet as a transport for sample data has had many advantages. Issues with driver compatibility have virtually disappeared, as Metis will talk to any device that has an ethernet driver. The network architecture also allows novel configurations such as multiple computers talking to a single piece of RF hardware and even multiple pieces of RF hardware talking to a single computer. There are also efforts to implement a "soft core" microprocessor in the FPGA to allow a real-time operating system to run directly on the Metis board.

In addition to Mercury, Metis, and PennyLane, there are a variety of add-on boards to help make the Atlas-based systems into fully-functioning transceivers. Production has recently begun on Alexiares, also known as Alex, a transmitter low-pass filter bank and receiver preselector. This board will allow for better rejection of out-of-band signals at the receiver input and reduce any aliasing that occurs in either the transmit or receive path.

PennyWhistle is a 20-watt power amplifier (PA) for OpenHPSDR systems. It is meant to take the 0.5W output from PennyLane or Penelope and amplify it to a reasonable power level. The spurious emissions levels for this card do not comply with FCC regulations in the absence of additional filtering so a solution such as Alex is required.

Excalibur is a 10-MHz frequency reference that can either use an on-board ±1ppm TCXO (temperature-compensated crystal oscillator) or input from an external 10-MHz source such as a GPS-DO or rubidium standard. The 10-MHz signal is then used to phase-lock the various clocks on Mercury, PennyLane, and other boards.

Pandora, an enclosure, and LPU, a linear power supply, are also available for Atlas systems. The OpenHPSDR Wiki[4] has more information on hardware components of Atlas systems. A complete OpenHPSDR transceiver is shown in photo E.

Hermes

The second generation of OpenHPSDR systems is called Hermes and has been developed to design a single board solution with performance levels equal to or exceeding the Atlas systems. Rather than placing multiple FPGAs on the board, project designers found that there was enough room in a single large Altera Cyclone III to contain the entirety of the code for Mercury, PennyLane, and Metis. While early prototypes of Hermes used the USB circuitry from Ozymandias, the project decided after the huge success of Metis to incorporate its ethernet interface into the Hermes project.

In addition to the FPGA, Hermes has the same ADC and DAC as Mercury and PennyLane, respectively. Hermes is laid out on an 8-layer Eurocard-size board which is intended to fit into a standard Hammond enclosure. It has an on-board switch-mode power supply which is designed to run from 12 volts DC. Even with a switcher, spurs on Hermes are better than on Mercury.

As a companion to Hermes, the project designed the Apollo board which is a 15W PA, filter bank, and automatic antenna tuner. Like Hermes, Apollo is designed to fit into a standard Hammond enclosure. This allows a 2-slot enclosure to contain a complete 15W OpenHPSDR transceiver. This portability and simplicity will not only make it easier for new project users to obtain a working transceiver, it will also allow for new usage possibilities such as portable and remote operations.

Photo F– The PowerSDR software developed for FlexRadio SDR transceivers is commonly used by OpenHPSDR Project participants.

Hermes is in its final design stages and is being readied for production. It is hoped that preparations can be complete and that it will be offered for sale before the end of this year.

Software

Because of the open nature of the hardware designs and the ready availability of specifications and source code, a robust ecosystem of software packages has grown up around the OpenHPSDR project. These software packages are available on a variety of operating systems and represent many different interpretations of what an amateur software-defined radio should be.

The most popular software platform for OpenHPSDR experimenters is PowerSDR from Flex Radio Systems (photo F). Flex Radio has been kind enough to open-source itsr source code

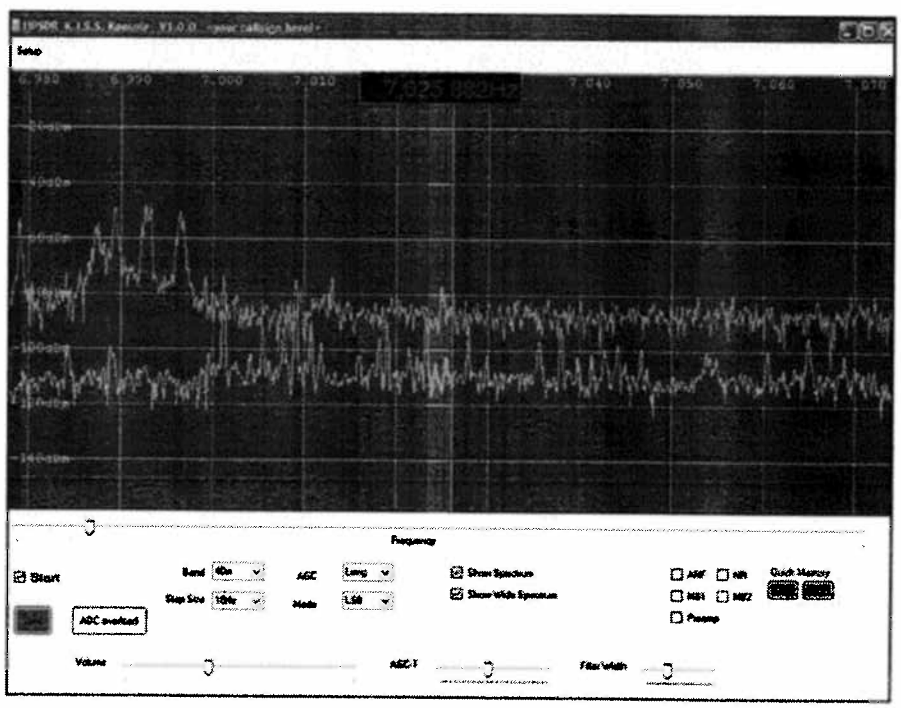

Photo G– The Kiss Konsole was developed by project members using Microsoft's .NET framework. It supports both the USB and ethernet-based OpenHPSDR systems.

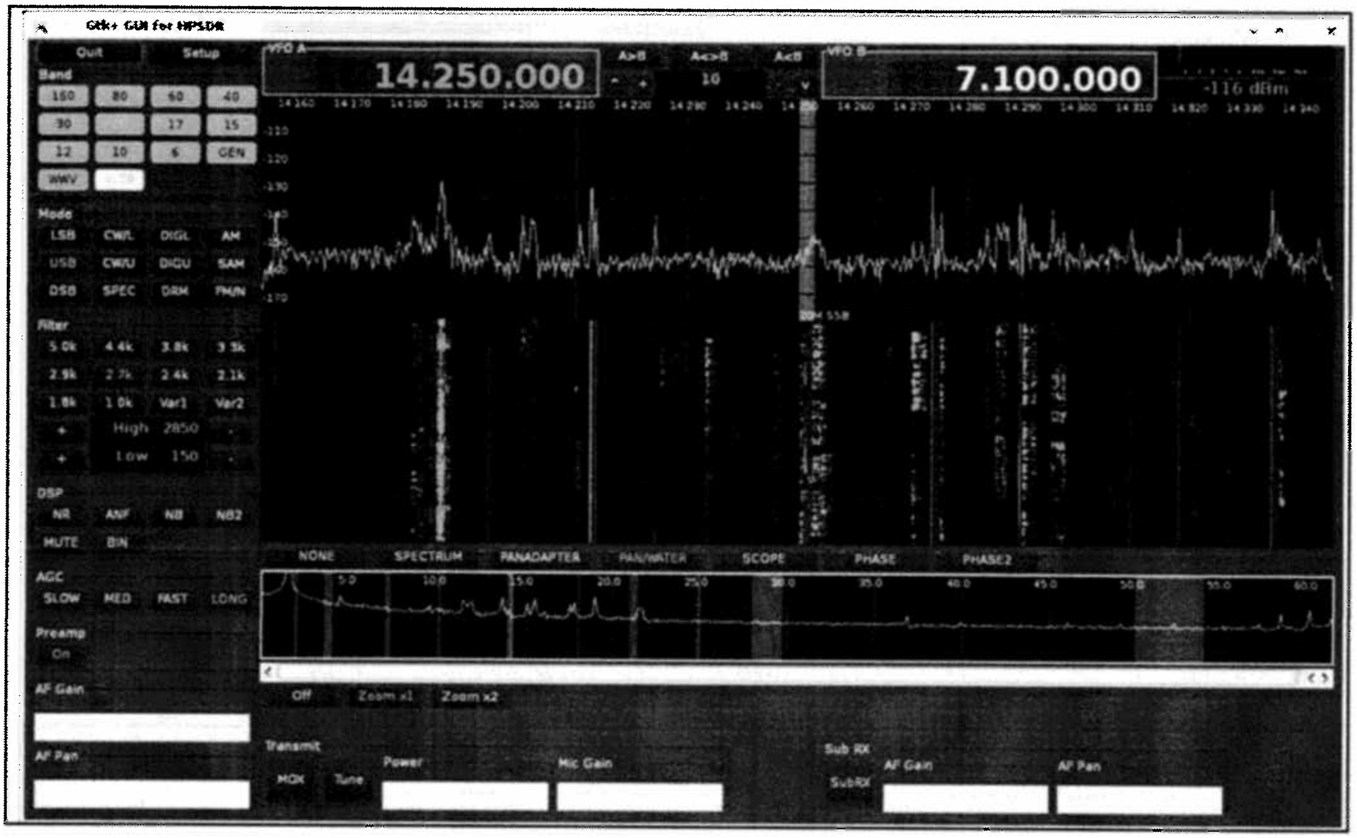

Photo H– GHPSDR is a popular HPSDR program among Linux users. It splits the workload of a software-defined radio into multiple components that can be assembled and run on different computers

for the software piece of their excellent commercially available software-defined radios. The OpenHPSDR project, like many SDR projects, has modified the base PowerSDR source to create a program compatible with the OpenHPSDR hardware platforms. The latest revision is 1.19.3 and is available from the OpenHPSDR website.

Various experimenters have used this base PowerSDR distribution to provide extended functionality to the OpenHPSDR platform. One of the more notable projects is the diversity receive work by Joe Martin, K5SO. The diversity versions of PowerSDR allow an Atlas-based system with multiple Mercury cards to process inputs from multiple spatially or polarity-diverse antennas to achieve increased signal levels and beam steering or nulling. Joe is working on versions of his software that will permit three and four antenna inputs for implementations of antenna arrays such as an electronically steered "four square" antenna.

In addition to PowerSDR, OpenHPSDR project members have developed Kiss Konsole (photo G), a piece of Windows® software written in the C# language using Microsoft's .NET frameworks. Kiss Konsole is designed to have source code that is easily understandable and modifiable to encourage experimentation. Many new hardware features are first supported in Kiss Konsole before they are ported to other platforms. The program has recently been fitted with a unified architecture that supports both the USB and Ethernet-based OpenHPSDR systems in one executable.

Pushing the Boundaries

For those users of the Linux operating system, John Melton, N6LYT, has developed a program called *ghpsdr* (photo H). The newest version of ghpsdr, 3.0, is based on a modular concept that splits the workload of a software-defined radio into multiple components that can be assembled and run on different computers. A hardware server provides samples to a DSP server that performs all of the mathematically intense processing. This DSP server then talks to a front end, which provides audio, control, and visual displays to the user. Users of ghpsdr 3.0 have had great success providing internet-connected radios to hams around the world. Additionally, John has fashioned clients that run on a multitude of different hardware platforms including the iPhone and Android. John's approach shows great promise in how to fully utilize the capabilities of the OpenHPSDR hardware.

Another project that is pushing the boundaries in SDR is *cuSDR* by Hermann von Hasseln, DL3HVH. Hermann has been experimenting with nVidia's CUDA, which is a system that allows a computer to use its graphics card for general-purpose computing. It turns out that the processor architecture of a graphics card is uniquely suited to the digital signal processing required for SDR radios because of its ability to perform the same operation in parallel to an array of inputs. Hermann is developing a DSP library and client that can leverage this power to drive the OpenHPSDR hardware. This experimentation promises not only faster SDR radios, but ones that have the ability to demodulate and decode an increasing amount of the bandwidth of the RF hardware.

Finally, for the Macintosh lovers out there I have written an OpenHPSDR compatible package called Heterodyne (photo I). It is a fully

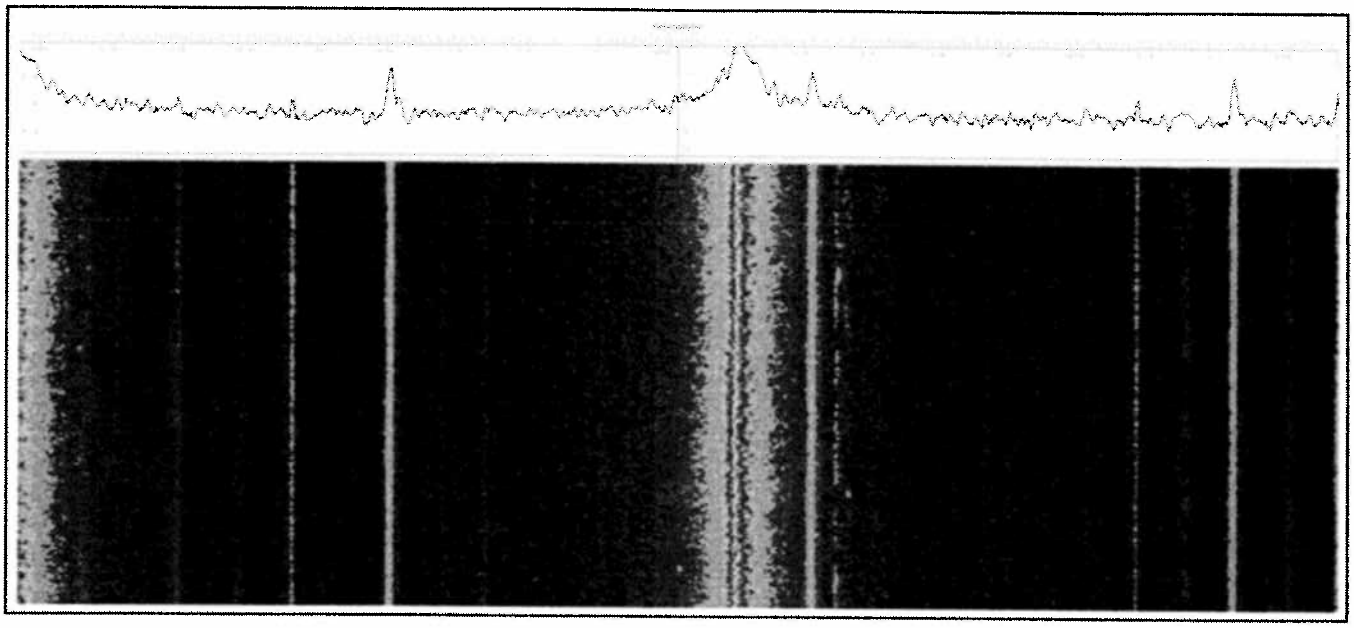

Photo I– The Heterodyne software for MacOS X tuned to the 20-meter phone band. The band is relatively dead, but an SSB signal can be seen just below 14,280 kHz.

native Cocoa-based Macintosh program written in Objective C. Heterodyne supports both the ethernet and USB OpenHPSDR connections by providing a driver layer that should theoretically allow it to support many disparate types of hardware. The program is intended to be easy to install and use and to conform to as many Macintosh platform customs and conventions as possible. Due to licensing restrictions it is not available on Apple's Mac App Store right now, but I hope to have it available there soon.

All of these programs are open-source and have source code available for third-party use and modification. All of the project leaders encourage contributions and comments on the functionality and implementation of the code. Most software code is available on the OpenHPSDR source code control system.

Getting Involved

The OpenHPSDR project has no formal membership requirements, dues, or defined roles. Anyone is welcome to join the group and contribute his or her talents. The main method of communication among project members is a weekly meeting on Teamspeak, an internet voice conferencing solution. These meetings take place on Saturdays at 0100 UTC during the U.S.'s daylight savings time, and 0200 UTC during standard time. More information on the Teamspeak session is available on our website at <http://www.openhpsdr.org/teamspeak.php>.

Another important method of communication for the OpenHPSDR project is our e-mail list. The list, like the Teamspeak session, is open to anyone who would like to participate. There are also searchable list archives for research purposes. More information on the discussion list is available on our website at <http://www.openhpsdr.org/reflector.php>.

There is also a wealth of information on the project via the web. The project maintains a website at <http://www.openhpsdr.org> which contains a Wiki with a trove of user-contributed content. This serves as an entry point to further research on the multitude of projects ongoing within the OpenHPSDR community.

The OpenHPSDR project provides an excellent platform for those wanting to experiment and homebrew with SDR technology. The project continues to grow, expand, and push the boundaries of what is possible with amateur software-defined radio. Through the contributions of the many exceptionally gifted hams in the project, we hope to continue the tradition of a high-performance software-defined radio that is available for tinkering and exploring.

Notes

1. See http://www.tapr.org/ohl.html
2. OpenHPSDR boards are traditionally named after mythical figures or creatures such as the Greek and Roman gods or the Griffin (Gryphon).
3. Weak Signal Propagation Reporter, see <http://www.wsprnet.org>
4. See <http://www.openhpsdr.org/wiki>