ARRL 100 YEARS

QEX

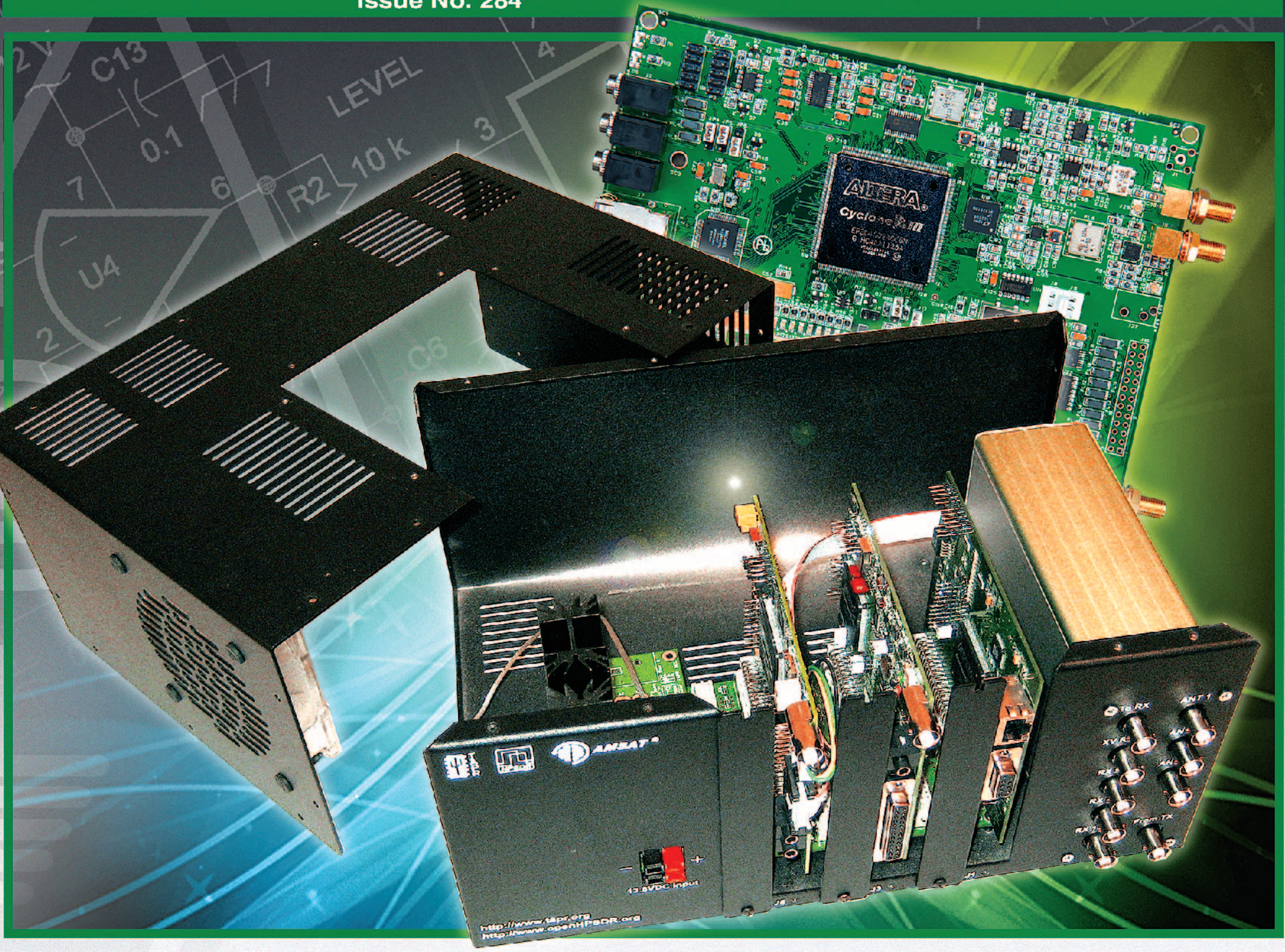
$5

May/June 2014

www.arrl.org

A Forum for Communications Experimenters

Issue No. 284



WA2DFI updates us on "The High Performance Software Defined Radio Project." From a collection of boards that fit nicely into a custom chassis, to a single-board transmitter and receiver, this project is engineered to be a top performer.

ARRL The national association for AMATEUR RADIO™
225 Main Street
Newington, CT USA 06111-1494

QEX

QEX (ISSN: 0886-8093) is published bimonthly in January, March, May, July, September, and November by the American Radio Relay League, 225 Main Street, Newington, CT 06111-1494. Periodicals postage paid at Hartford, CT and at additional mailing offices.

POSTMASTER: Send address changes to: QEX, 225 Main St, Newington, CT 06111-1494

Issue No 284

Offices

225 Main St, Newington, CT 06111-1494 USA
Telephone: 860-594-0200
Fax: 860-594-0259 (24 hour direct line)
e-mail: **qex@arrl.org**

Subscription rate for 6 issues:

In the US: ARRL Member $24, nonmember $36;

US by First Class Mail: ARRL member $37, nonmember $49;

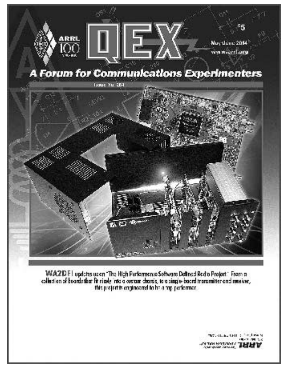
International and Canada by Airmail: ARRL member $31, nonmember $43;

Members are asked to include their membership control number or a label from their QST when applying.

In order to ensure prompt delivery, we ask that you periodically check the address information on your mailing label. If you find any inaccuracies, please contact the Circulation Department immediately. Thank you for your assistance.

About the Cover

Scotty Cowling, WA2DFI, tells us about "The High Performance Software Defined Radio Project (HPSDR). This project is a collection of circuit boards designed to help interested builders experiment with the hardware needed to assemble a top performing radio. The latest development is the Hermes Board — a single-board transmitter and receiver that integrates onto one board most of the functions of the boards previously developed for HPSDR.



In This Issue

Features

3 **The High Performance Software Defined Radio Project**
Scotty Cowling, WA2DFI

14 **RF Power Amplification Using a High Voltage, High Current IGBT**
Pete Horowitz

21 **Locked VCXOs for Stable Microwave Local Oscillators with Low Phase Noise**
Paul Wade, W1GHZ

32 **Android Wireless Project Control Part 1 — Android GUI Software Development**
Thomas M. Alldread, VA7A

38 **Upcoming Conferences**

Index of Advertisers

ARRL: ....Cover III
Array Solutions: 31
Down East Microwave Inc:.......................... 20
Kenwood Communications: ....Cover II
M²:... 40
Nemal Electronics International, Inc: 20
Quicksilver Radio Products............... Cover IV
RF Parts:... 37, 39
Tucson Amateur Packet Radio: 40

The American Radio Relay League

The American Radio Relay League, Inc, is a noncommercial association of radio amateurs, organized for the promotion of interest in Amateur Radio communication and experimentation, for the establishment of networks to provide communications in the event of disasters or other emergencies, for the advancement of the radio art and of the public welfare, for the representation of the radio amateur in legislative matters, and for the maintenance of fraternalism and a high standard of conduct.

ARRL is an incorporated association without capital stock chartered under the laws of the state of Connecticut, and is an exempt organization under Section 501(c)(3) of the Internal Revenue Code of 1986. Its affairs are governed by a Board of Directors, whose voting members are elected every three years by the general membership. The officers are elected or appointed by the Directors. The League is noncommercial, and no one who could gain financially from the shaping of its affairs is eligible for membership on its Board.

"Of, by, and for the radio amateur," ARRL numbers within its ranks the vast majority of active amateurs in the nation and has a proud history of achievement as the standard-bearer in amateur affairs.

A *bona fide* interest in Amateur Radio is the only essential qualification of membership; an Amateur Radio license is not a prerequisite, although full voting membership is granted only to licensed amateurs in the US.

Membership inquiries and general correspondence should be addressed to the administrative headquarters:

ARRL
225 Main Street
Newington, CT 06111 USA
Telephone: 860-594-0200
FAX: 860-594-0259 (24-hour direct line)

Officers

President: KAY C. CRAIGIE, N3KN
570 Brush Mountain Rd, Blacksburg, VA 24060

Chief Executive Officer: DAVID SUMNER, K1ZZ

The purpose of *QEX* is to:

1) provide a medium for the exchange of ideas and information among Amateur Radio experimenters,

2) document advanced technical work in the Amateur Radio field, and

3) support efforts to advance the state of the Amateur Radio art.

All correspondence concerning *QEX* should be addressed to the American Radio Relay League, 225 Main Street, Newington, CT 06111 USA. Envelopes containing manuscripts and letters for publication in *QEX* should be marked Editor, *QEX*.

Both theoretical and practical technical articles are welcomed. Manuscripts should be submitted in word-processor format, if possible. We can redraw any figures as long as their content is clear.
Photos should be glossy, color or black-and-white prints of at least the size they are to appear in *QEX* or high-resolution digital images (300 dots per inch or higher at the printed size). Further information for authors can be found on the Web at **www.arrl.org/qex/** or by e-mail to **qex@arrl.org**.

Larry Wolfgang, WR1B

Empirical Outlook

Learning Opportunities

Most of us would agree that education is a life-long process. It certainly doesn't end with a high school diploma. By the time we earn an Associates or Bachelors degree in our chosen field we have come to realize that the more we learn, the more we want to learn. Going on to a Masters degree or even a Doctorate will not end the learning process.

Amateur Radio is an excellent example of this ongoing need to learn. As new integrated circuits are developed we want to learn about how to use those new components. As new communications techniques are developed we want to learn how we, too, can participate or even help lead the way. How do we learn about all this new information? Certainly there are many ways we can learn new tricks.

I am seeing more and more notices about webinars, and being invited to attend them. Various manufacturers, such as Texas Instruments, Linear Technology, and National Instruments all send newsletters with articles about the latest applications for their devices and software. I receive several other newsletters with information from a variety of companies. I am sure many of you, our readers, receive these same e-mails and more, depending upon your professional interests and connections. They all provide many learning opportunities. How many of them do you take advantage of? How many do you have time to read/study/observe? It can seem a bit overwhelming at times. If you are not receiving similar e-mails, and would like to, I encourage you to visit various manufacturer's websites and sign up for their newsletters and other information.

In the weeks running up to a major even, such as the National Association of Broadcasters Show and others, the e-mails increase even more. It seems that many of the major companies who will attend the show want to make sure I will stop by their booth to see their latest offering. I'm sure it would be great fun to attend one of those shows sometime.

My point is that we have many opportunities to learn new things, whether it be by attending a webinar, studying information newsletters from various manufacturers, finding a book about a topic of interest or reading the articles in *QEX*. I try to take as many of these opportunities as time and cost allow. How about you?

Another opportunity that many readers take advantage of, but more probably should, is attending the various Amateur Radio technical conferences held around the country each year. We try to highlight many conferences in the Upcoming Conferences column in just about every issue of *QEX*. I have periodically mentioned some of them in my editorials, and encouraged readers to attend one or more of these conferences each year. There are some conferences that tend to be regional in nature. Most of these move to different locations over the years, in an attempt to make it easier for local hams to attend occasionally. For example, the Eastern States VHF Conference, the Central States VHF Conference and others provide great learning opportunities. Other conferences are offered on a national scale, such as the ARRL/TAPR Digital Communications Conference, the AMSAT Symposium, and Microwave Update. The technical presentations are top notch, and you get the opportunity to rub elbows with the experts. Sure you can always pick up a copy of the *Conference Proceedings*, and read the papers there, but by attending the conference you can ask questions of the authors. You can discuss their techniques and gain further insight into their projects. Often you can hold the actual project in your hands and inspect the construction details. It is hard to imagine a better learning opportunity.

In the March/April 2014 issue of *QEX*, we reprinted "78 GHz LNA Wrap-Up" by Tom Williams, WA1MBA, from the *Proceedings of Microwave Update2013*. In this issue we have picked up an article by Paul Wade, W1GHZ, also from the *Proceedings of Microwave Update2013* — "Locked VCXOs for Stable Microwave Local Oscillators with Low Phase Noise." Both articles are excellent project descriptions and could have been "*QEX* Exclusive" articles, but the authors chose to present them first at MUD, to share them with other attendees there.

In the past we have occasionally picked up articles from the *DCC Proceedings* and other conferences. One reason for doing that is that they are excellent papers worthy of being shared with a wider audience. Another reason is to show our readers the quality of presentations that are offered at these conferences. I hope we can inspire some new attendees at one or more of the many great technical conferences offered each year. Let me know if you attend a conference for the first time, and tell me about your experience. Perhaps I will use it in the Letters column.

If you know of other conferences that we have not mentioned here or in the Upcoming Conferences column, please let me know about them. Provide some details at least 5 to 6 months in advance (the earlier the better) so we can be sure to include a timely notice in the pages of *QEX*.

Scotty Cowling, WA2DFI

PO Box 26843, Tempe, AZ 85285; **scotty@tonks.com**

The High Performance Software Defined Radio Project

Here is a summary of the history and current status of the HPSDR project.

Since its inception in 2005, the High Performance Software Defined Radio (HPSDR) project has produced over a dozen building blocks that can be used to assemble a high-grade 100 kHz to 55 MHz software-defined radio. See Figure 1.

The openHPSDR project, as it is known today, began in March 2006 from the merger of the HPSDR Yahoo group and the Xylo-SDR e-mail reflector. The first piece of hardware produced was the Atlas backplane. Eric Ellison, AA4SW, paid for the initial run of 400 circuit boards and shipped them to individuals from his dining room table in May of 2006. He collected enough money (entirely on the honor system) from these early adopters to pay for the initial circuit board run. Mainly because of Eric's efforts, Tucson Amateur Packet Radio (TAPR) became involved on the production side in June 2006 and was able to help augment the many HPSDR designers' efforts with early volume production and storefront retail sales. While TAPR offers financial support to the designers to help defray some or all of the costs of building prototypes for testing, TAPR and the HPSDR project always were and remain independent entities. The HPSDR project changed its name to openHPSDR in April 2009 in order to more accurately reflect the open-source nature of the project.

In fact, the openHPSDR project was the impetus for creating the TAPR Open Hardware License ("OHL"). The openHPSDR developers wanted to create a community around their designs, much like the GNU General Public License for software, and invited TAPR to work with them to develop a license for open hardware designs. The OHL itself was the result of an open design process that included a public comment period. It was released in May, 2007, and is available for use by any open hardware project.[1]

Figure 1 — The openHPSDR Transmitter/Receiver. From left to right, Linear Power Unit (LPU), Mercury receiver, Pennylane transmitter, and Metis Ethernet interface are plugged into the Atlas backplane. Alex filters (in the aluminum enclosure) are on the right. All boards fit within the Pandora enclosure.

TAPR is a non-profit corporation that provides resources for the purpose of advancing the state of the radio art, especially the *digital* radio art.[2] What could be more digital than a software defined radio with an A-to-D conversion practically at the antenna?

The openHPSDR project is a community (currently over 1000 strong) of designers, developers and users that design, build and experiment with high-performance radios.[3] The openHPSDR domain hosts an active e-mail reflector where new hardware is proposed, software is discussed and where users can get (and offer) openHPSDR system help and operating tips.[4]

System Architecture

From the beginning, the openHPSDR project was designed to be modular and expandable. This type of architecture makes the system a bit more costly and complex because common interface circuitry must

[1]Notes appear on page 13.

openHPSDR Names Quick Reference

Each openHPSDR board has a project leader. (You can visit each board's Wiki page to identify the culprit.) While this person interacts with the openHPSDR community, ultimately the design decisions and name selection rest with him. Early in the project, mythological names seemed to be the preference, and the tradition stuck. Here is a quick guide to openHPSDR names.

- Atlas = Backplane
- Magister = USB interface
- Metis = Ethernet interface
- Mercury = Receiver
- Penelope = Original transmitter, replaced by Pennylane
- Pennylane = Transmitter
- LPU = Power Supply, or Linear Power Unit
- Excalibur = Frequency Reference
- Janus = Baseband A/D, D/A card
- Pinocchio = Extender
- Pandora = Enclosure
- Pennywhistle = Power Amplifier
- Alexiares = LP/HP Filter Set
- Hermes = Single Board SDR
- Apollo = Filters and PA accessory to Hermes

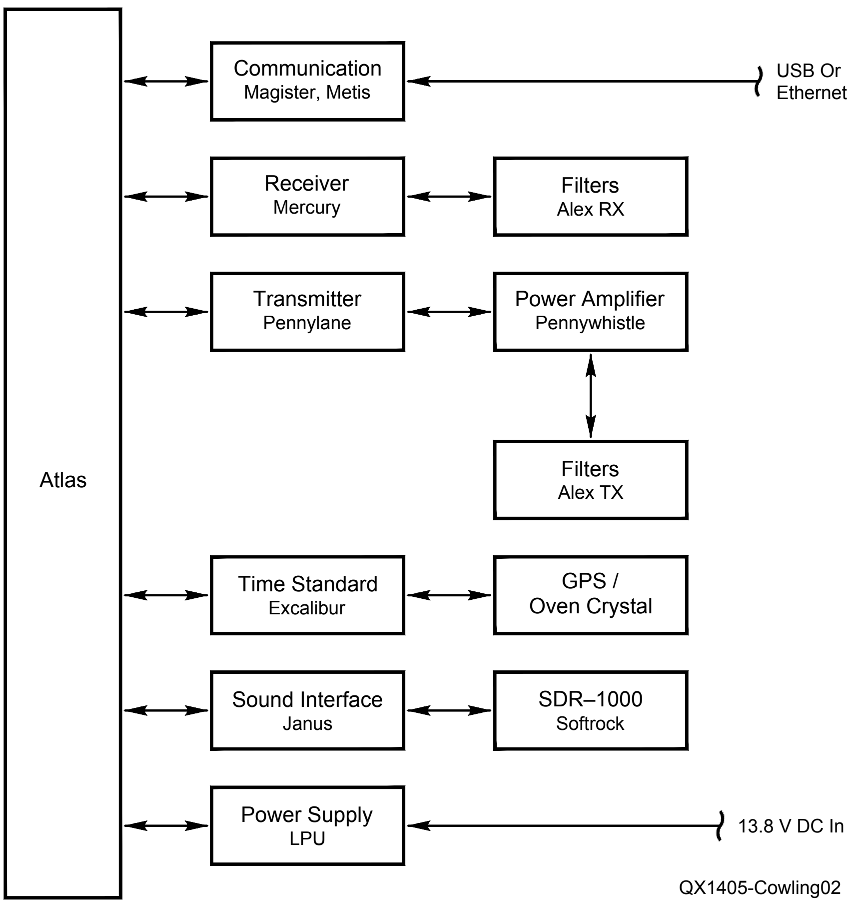


Figure 2 — openHPSDR System Overview.

be duplicated on each module. The resulting system is, however, inherently upgradeable and flexible; these two features are highly desirable from an experimenter's point of view. From the openHPSDR perspective, performance generally takes precedence over cost.

An example of the value of the openHPSDR upgrade path is in order. A production run of the Penelope transmitter board was made by TAPR in May 2008. Penelope was a good transmitter, but it had two shortcomings. First, the power output fell off rather quickly above 30 MHz because of the design of the PA output stage. Second, there was no hardware power-output level control. The only way to reduce power output was by scaling the data values sent to the DAC, resulting in increasing quantization errors (and thus, more distortion in the output waveform) as the output power was decreased. In August of 2011, both of these shortcomings were addressed with the production of the Pennylane transmitter board. Pennylane simply replaces Penelope, uses the same firmware and software, but performs better. Interestingly enough, due to the open source nature of this project, Pennylane was produced by iQuadLabs and not by TAPR; more on this later.[5] The example illustrated here is that drop-in hardware enhancements are possible with a modular architecture that would not be possible with a single-board SDR.

One other hardware feature worth noting is that all but one of the openHPSDR boards that plug into the Atlas backplane are a standard size (100 mm by 120 mm) and all use a standard connector (96-pin DIN41612). This makes a common enclosure for all systems feasible.

I have teased you with mysterious talk of Atlas, Penelope and Pennylane long enough. Let's move on to some hardware details. Please follow the link for each hardware component for more detail, schematics, parts lists, and layouts, as appropriate. I will also indicate a source for purchasing bare circuit boards, assembled and tested boards or kits, depending on what is available. Most of the hardware described below is released under TAPR OHL. A few designs are under the TAPR non-commercial license (NCL — now deprecated), but will be moved to OHL when possible.[6]

It is helpful to refer to Figure 2 while reading the board descriptions below to see how each board fits into the complete openHPSDR system.

Hardware — Main Atlas Backplane Components

These components consist of the Atlas 6-slot backplane and the three basic boards required for a functioning transmitter/receiver. (It is not a transceiver in the classic sense, since the transmitter and receiver are separate and can operate at the same time, or in full duplex.) All openHPSDR systems must have a communications interface. A receiver, a transmitter or both is also required.

Atlas Backplane – The Heart of it All

The Atlas 6-slot backplane (Figure 3) consists of six 96-pin DIN connectors bused to an unterminated 48-bit (6-byte) wide bus. In addition to the bus, six pins of each connector are daisy-chained to the adjacent DIN connectors (3 pins to each side). Power connections are provided to each DIN connector for one common ground and five power supplies: +12 V, +5 V, +3.3 V, –5 V and –12 V. Paralleled pins allow each power supply connection to carry in excess of 2.5 A. The power input connector is a standard ATX computer motherboard connector; an off-the-shelf ATX power supply can be connected here, but be aware that most PC power supplies are very RF noisy and may compromise receiver small-signal performance. LEDs are provided for each power rail, and a header is provided for remote power control of an ATX supply. Power rail bypassing is abundant.

TAPR offers the Atlas 6-slot backplane as a kit only. The DIN and ATX connectors are through-hole, and the remaining parts are relatively easy to assemble 0805 size surface mount (SMT) parts.

Atlas documentation is available on the openHPSDR Atlas web page.[7] Atlas kits are available from TAPR.[8]

Communications Interface – Two to Choose From

The communications interface is the openHPSDR endpoint of the data path between the PC and the radio. Magister uses USB 2.0 as its interface, while Metis uses Gigabit Ethernet for the same function. Speeds and protocols differ between the two boards, but the function is the same.

Note that these two boards are ***alternates***; you cannot use both interfaces at the same time.

Magister USB Interface

Magister (Figure 4) is a high-speed USB 2.0 interface built around a Cypress CY7C68013A FX2 microcontroller and an Altera Cyclone II EP2C8 FPGA (field programmable gate array). The FX2 provides the USB 2.0 interface to the PC and a FIFO interface to the FPGA. The FPGA formats the data to/from the various openHPSDR components via the Atlas bus.

Magister documentation is available on the openHPSDR Magister web page.[9] Fully assembled and tested Magister boards are available from TAPR.[10]

Metis Gigabit Ethernet Interface

Metis (Figure 5) is a 100M/1000M Ethernet interface built around a Micrel KSZ9021RL Gigabit PHY and a large Altera Cyclone III EP3C40 FPGA. The FPGA is the largest Cyclone III part that is available in a leaded (240-pin QFP) package. There are 12 FPGA-controlled LEDs, an LVTTL-level serial port, 512 kbytes of SRAM as well as four digital outputs and three digital inputs.

Metis can use an IP (Internet Protocol) address obtained via DHCP (Dynamic Host Configuration Protocol); lacking a DHCP server on the network, it will use an automatic private IP address (APIPA) of the form 169.254.x.x, where x.x is determined by the board's media access control (MAC) address. Each Metis board has an on-board

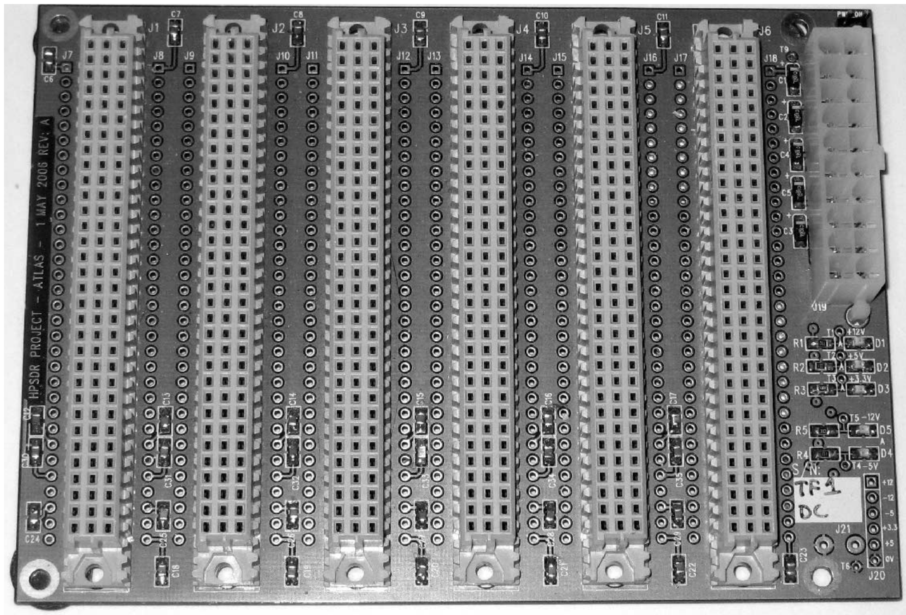


Figure 3 — Atlas 6-Slot Backplane.

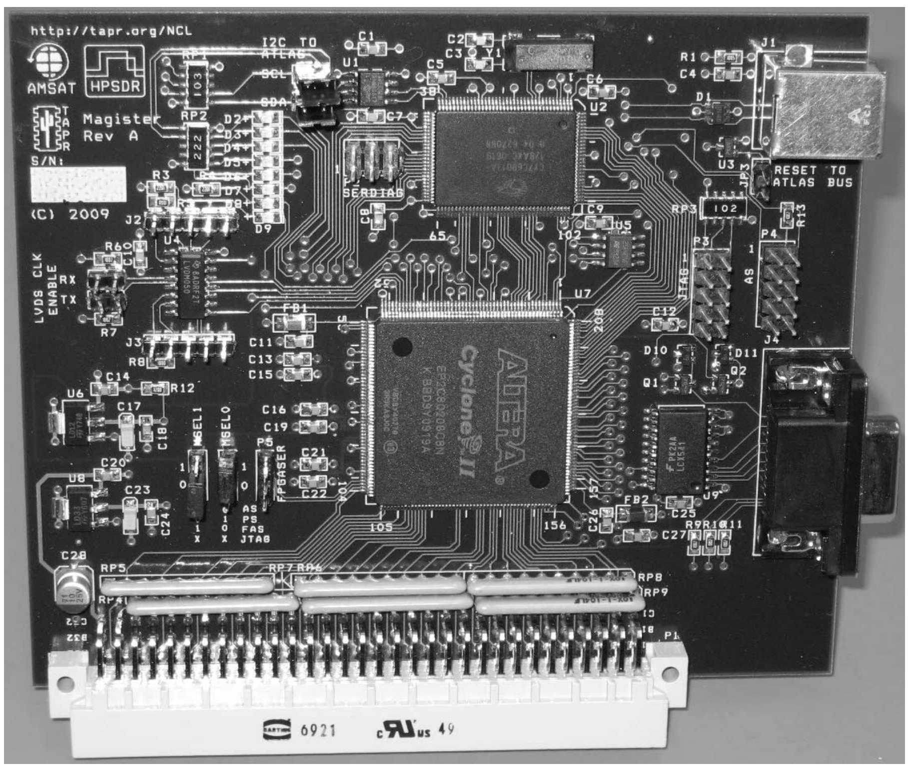


Figure 4 — Magister USB Interface.

EEPROM that is pre-programmed with a unique MAC address. A fixed IP address can be optionally stored in this EEPROM as well. Data from the Atlas bus (from a Mercury receiver, for example) is formatted by the logic in the FPGA and sent to the PC via user datagram protocol (UDP) packets. In the opposite direction, UDP data from the PC is formatted and sent to the Atlas bus (to a Pennylane transmitter, for example).

Metis documentation is available on the openHPSDR Metis web page.[11] Assembled and tested Metis boards are available from TAPR.[12]

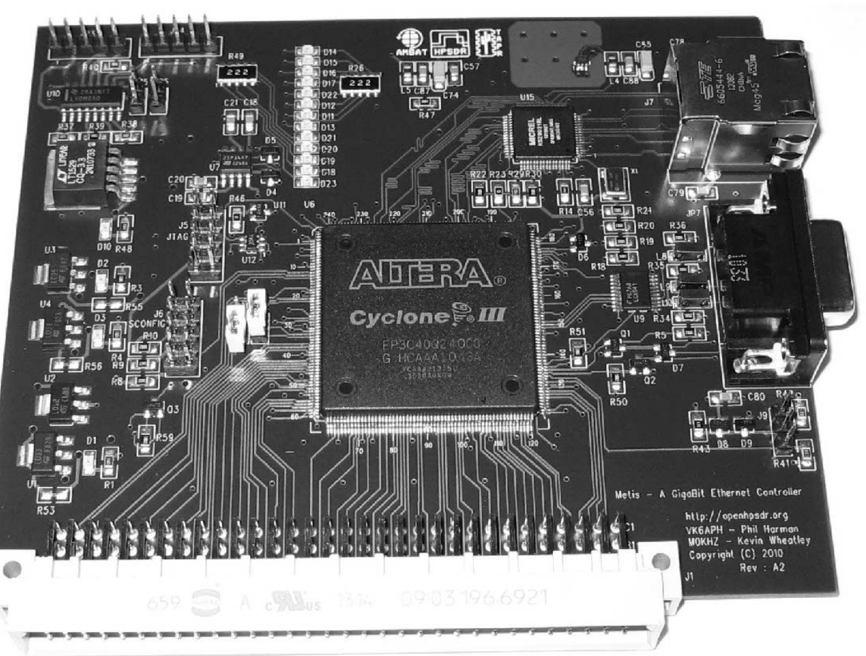


Figure 5 — Metis Gigabit Ethernet Interface.

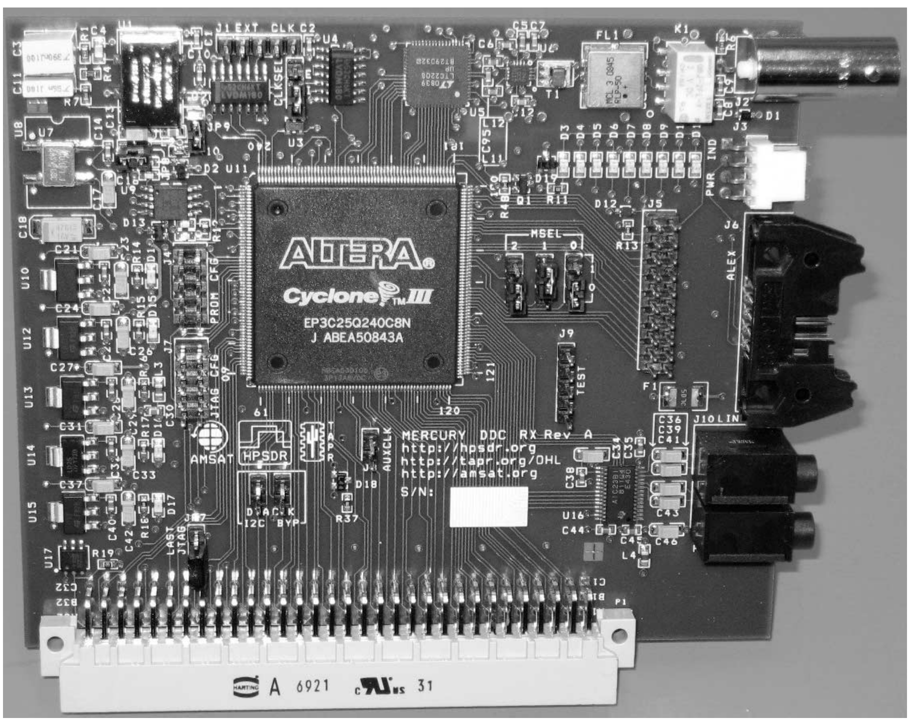


Figure 6 — Mercury Direct-Sampling Receiver.

Mercury Direct Sampling Receiver

Figure 6 shows the Mercury high-speed, direct-sampling receiver board. The Mercury front-end consists of a switched 20 dB attenuator followed by a 20 dB low noise amplifier (LNA) — an LTC6400-20 high-speed differential amplifier — and a low-pass filter (LPF). The LPF feeds an LTC2208 700 MHz bandwidth, 16-bit analog to digital converter (ADC) clocked at 122.88 MHz. The digitized data from the ADC is fed to an Altera Cyclone III EP3C25 FPGA where it is processed and sent to the communications interface (Magister or Metis) via the Atlas bus. This "processing" consists of combined filtering and decimation to reduce the amount of data sent across the Atlas backplane and, eventually, to the PC for demodulation and/or display. For those interested in the inner workings of the Mercury FPGA code, Mercury development history is available on-line.[13]

Mercury is a very high performance receiver, with a minimum discernable signal (MDS) of about –138 dBm and a blocking dynamic range (BDR) of about 119 dB. The overload point of the ADC at –12 dBm (+8 dBm with attenuator switched in) determines the BDR, which is not phase-noise limited. An excellent evaluation of Mercury performance can be found on-line.[14]

Mercury documentation is available on the openHPSDR Mercury web page.[15] Assembled and tested Mercury boards are available from TAPR.[16]

Pennylane Direct Up-Conversion (DUC) 500 mW Transmitter

As mentioned above, there are two openHPSDR transmitter boards. The new improved Pennylane transmitter has superseded the original Penelope transmitter board. The function of the two boards is identical; Pennylane just does the job a bit better than Penelope.

Pennylane is a 500 mW direct up-conversion transmitter, shown in Figure 7. The transmit data stream from the Atlas bus is processed by an Altera Cyclone II EP2C8 FPGA and fed to a high-speed 14-bit DAC (AD9744ARU) clocked at 122.88 MHz. The analog waveform from the DAC is filtered and then amplified by a two-stage 500 mW RF power amplifier (PA). Other features of Pennylane are on-board analog output level detection, four general-purpose analog inputs, three PWM outputs for future class E amplifier support, seven open-collector digital outputs and a CODEC for microphone audio input and auxiliary audio output. The on-board 122.88 MHz voltage controlled crystal oscillator (VCXO) is the same ultra-low phase noise type that Mercury uses.

Pennylane documentation is available on the openHPSDR Pennylane web page.[17] Assembled and tested Pennylane boards are available from TAPR.[18]

Hardware — Other Atlas Backplane Components

These components consist of a power supply and various other boards that provide additional openHPSDR functions. Excalibur provides enhanced frequency accuracy capability, Janus provides baseband A/D and D/A capability and Pinocchio allows openHPSDR cards to be "extended" above the backplane for debug tasks.

LPU — Linear Power Unit

Figure 8 shows the linear regulated power supply, LPU, designed to power an openHPSDR radio from a regulated 13.8 V bench supply. It provides 2 A at +12 V, 1.5 A at +5 V and 100 mA at –12 V from a 12.5 V to 14.5 V input. LPU can also supply 1 A at 3.3 V if optional parts are installed. The –12 V regulator is an inverting switch-mode regulator, and can be disabled to reduce switching noise when –12 V is not required. (Janus is the only openHPSDR board that uses –12 V.)

LPU passes the regulated input connection through to an internal connector for use by a power amplifier. LPU also provides a header for a 12 V dc fan, which is almost always required because of the large amount of heat generated by the linear nature of the LPU regulators. LPU plugs directly into the power connector on the Atlas backplane, without any cables.

LPU was intended to be a temporary solution until a custom openHPSDR switching power supply could be designed. The switching supply solution has not materialized thus far. LPU is inefficient, but it is also very RF-quiet; this is a good thing for the sensitive Mercury receiver.

LPU documentation is available on the openHPSDR LPU web page.[19] TAPR offers the LPU power supply as a kit only.[20] The SMT parts are relatively easy to solder 0805 or larger size.

Excalibur 10MHz Frequency Reference

Excalibur, shown in Figure 9, offers two options for generating a precision 10 MHz reference clock source for openHPSDR boards. The first option is Excalibur's on-board high-stability 10 MHz TCXO, which can be phase-locked to an external input. The second option is an external GPS-disciplined or other precision oscillator. The 10 MHz oscillators on Pennylane and Mercury have a rated stability of between ±50 ppm and ±100 ppm. Thus the 10 MHz clock error can be up to 1 kHz at temperature extremes. Excalibur's TCXO is rated at ±1 ppm, or 10 Hz at temperature extremes. At room temperature, the error is typically less than 1 Hz.

If an external high-performance GPS disciplined oscillator is used, typical

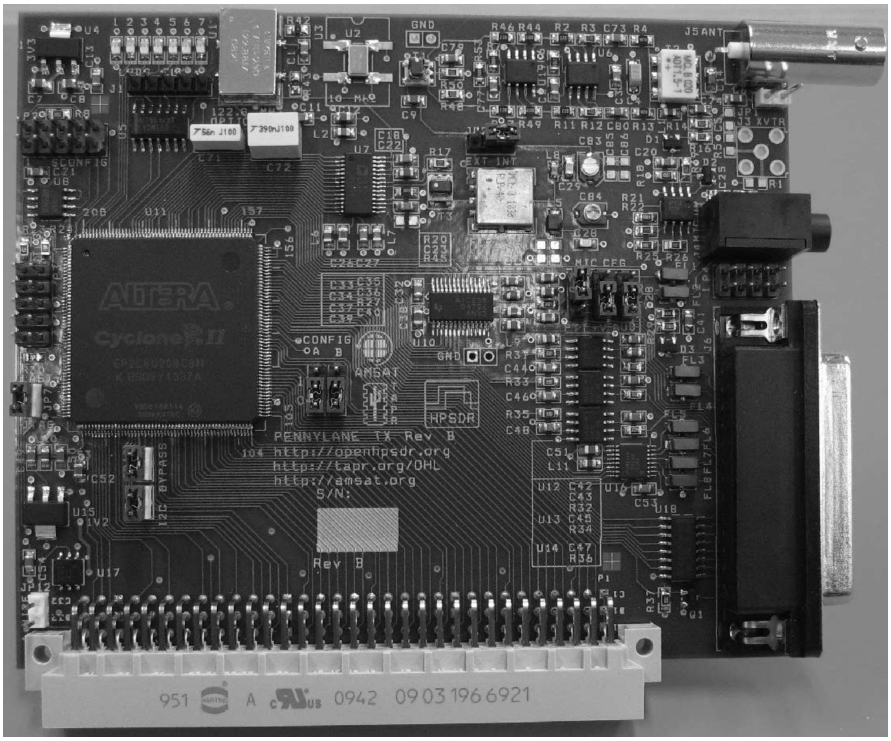


Figure 7 — Pennylane 500 mW Direct Up-Conversion (DUC) transmitter.

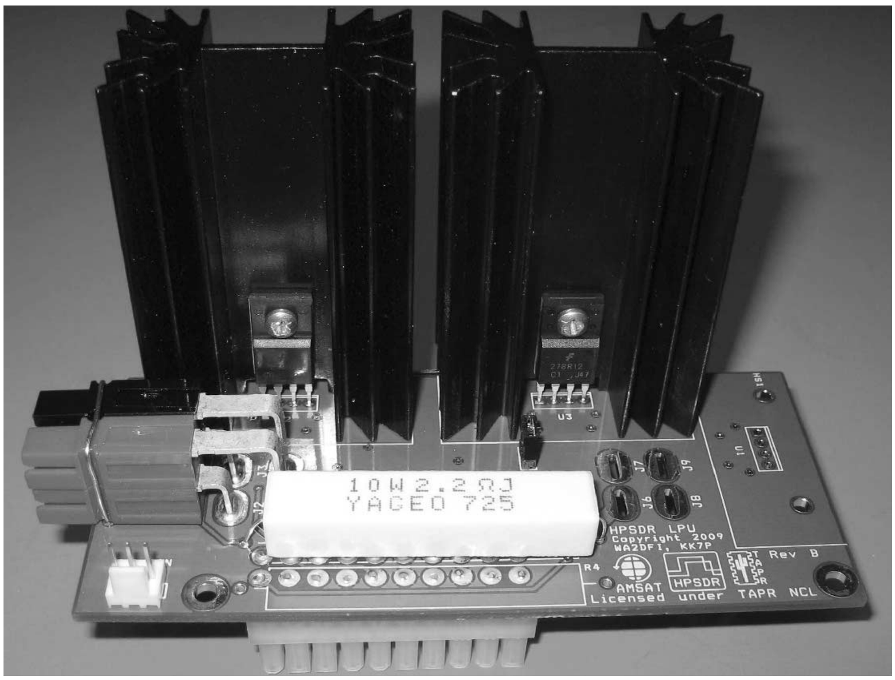


Figure 8 —Linear Power Unit (LPU).

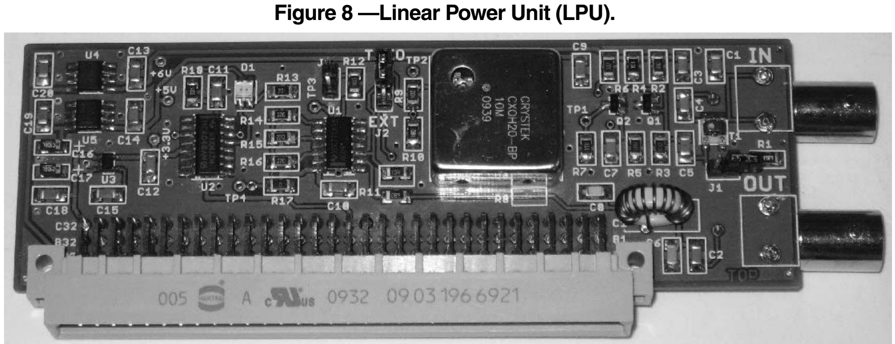


Figure 9 — Excalibur Frequency Reference.

accuracies of ±0.0001 ppm can be reached. This is one milliHertz at 10 MHz! The time nuts have lots of fun with Excalibur.[21]

Excalibur documentation is available on the openHPSDR Excalibur web page.[22]

TAPR offers Excalibur as a kit only.[23] The surface mount (SMT) parts are mostly easy to solder 0805 size, but there are a few smaller ICs. There is one evil toroid to wind.

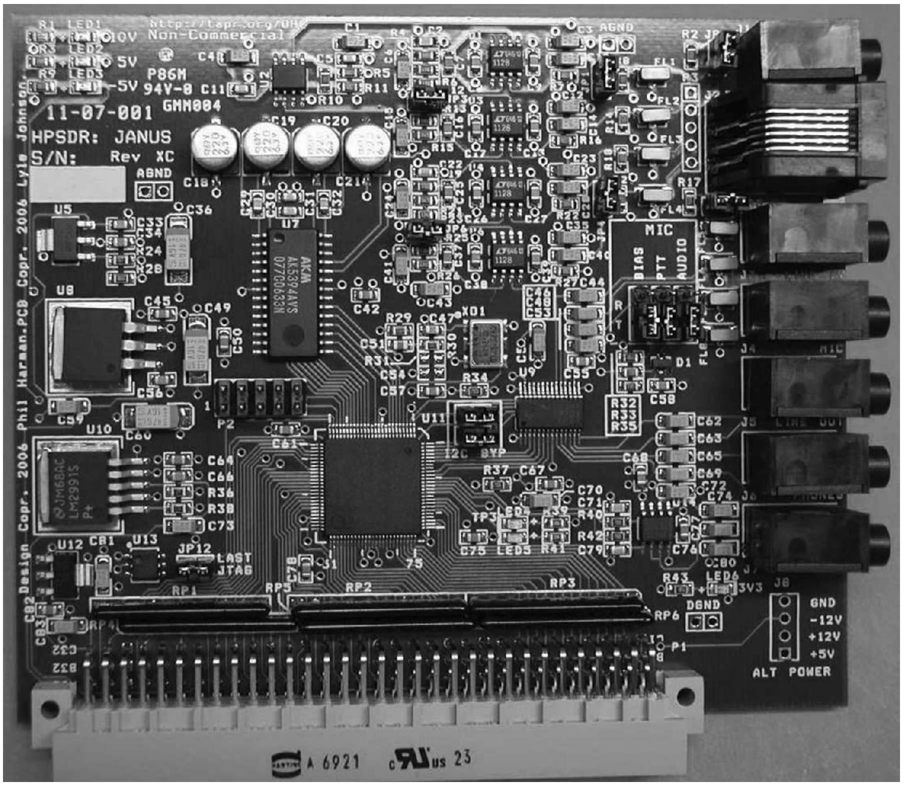


Figure 10 — Janus A/D and D/A Converter.

Janus Baseband A/D and D/A Converter

Figure 10 shows the Janus board. Janus is a very high-performance baseband A/D and D/A, or sound card. It uses a high-performance, 24 bit, 192 kilosamples per second (ksps) ADC (AKM AK5394) for baseband input, and a stereo CODEC (TI TLV320AIC23B) for mic/line input and headphones/line output. Janus is intended to be used with a source of I/Q data from a QSD-based receiver. Two examples of such receivers are the Softrock series from Tony Parks, KB9YIG and the SDR-1000 from FlexRadio Systems®.[24, 25] The CODEC output from Janus can also drive the QSE-based transmitter section of these same radios.

The performance of Janus equals or exceeds all but the very highest performance (read: expensive) PC sound cards. There is currently no *Windows* sound card driver for Janus, however. It can only be used with software that supports it directly, such as the openHPSDR version of PowerSDR™.

Janus documentation is available on the openHPSDR Janus web page.[26] Assembled and tested Janus boards are available from TAPR.[27]

Pinocchio Extender Card

While Pinocchio (Figure 11) was designed to raise any Atlas plug-in card up and into the open so it can be probed, it also has other uses. Since every Atlas bus signal is available on the surface of the

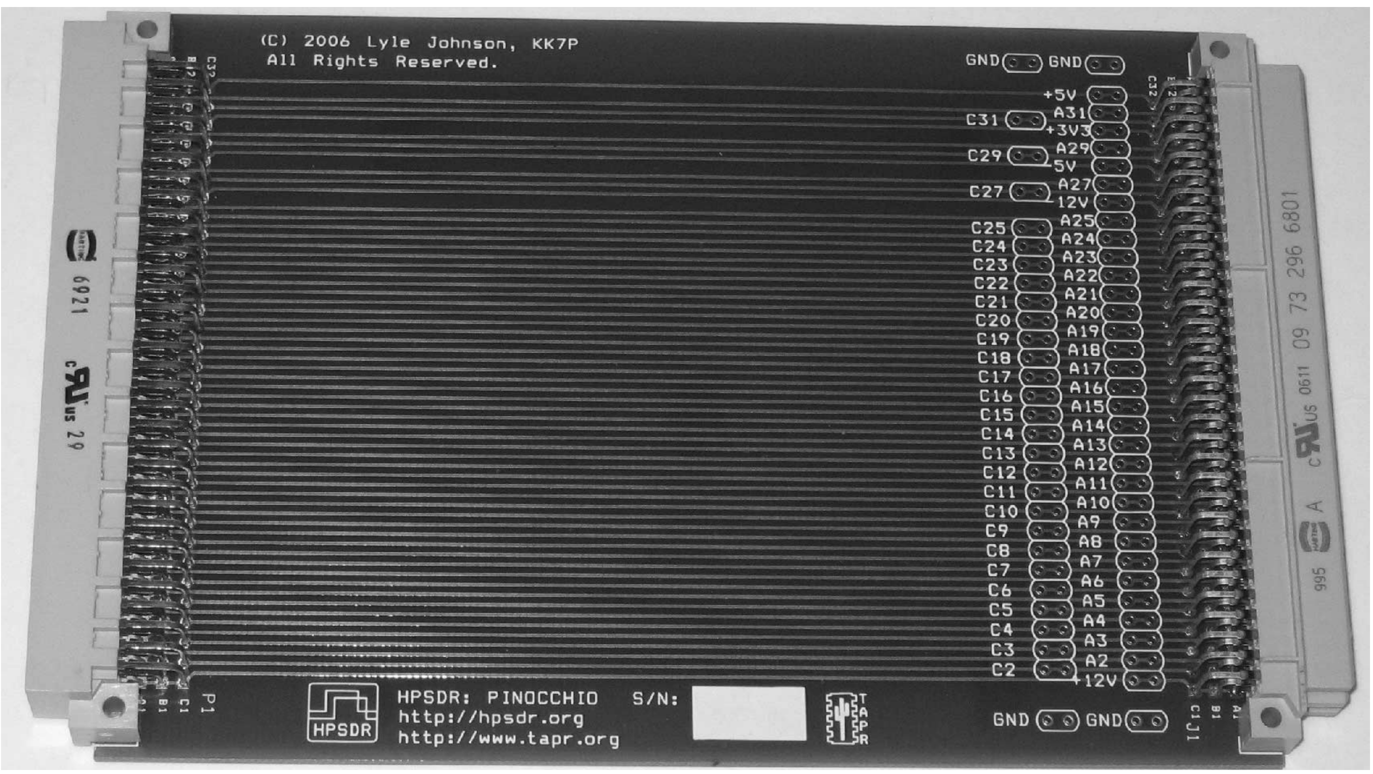


Figure 11 — Pinocchio Extender.

card, Pinocchio can make an excellent base for prototyping new hardware. It is a very simple kit, with through-hole right-angle 96-pin DIN connectors on each end of a circuit board.

Pinocchio documentation is available on the openHPSDR Pinocchio web page.[28] TAPR offers the Pinocchio extender as a kit only.[29]

Hardware — Non-Atlas Backplane Components

These components do not plug into the Atlas backplane, but are useful and/or necessary to build up a complete openHPSDR radio.

Pandora Chassis Enclosure

Pandora, shown in Figure 12, is an enclosure for openHPSDR components. It is large enough to house all of the components necessary for a 20 W (or more) HF/6 m transceiver. There are provisions for a fan, an Atlas backplane fully loaded with six boards, an LPU, a power amplifier (Pennywhistle or other model) and a set of Alex filters in a shielded sub-chassis. Pandora has a black, powder-coated finish and is made of aluminum for easy modification. It is pre-punched and drilled for all of the above components. Blank filler panels are included to block off unused Atlas slots.

Pandora documentation is available on the openHPSDR Pandora web page.[30] TAPR offers Pandora as a bolt-together enclosure complete with hardware and blank filler panels.[31]

Figure 12 — Pandora Enclosure.

Pennywhistle 20 W Power Amplifier

Pennywhistle is an RF power amplifier that produces up to 20 W of RF output from 250 mW of drive. See Figure 13. It uses two RD15HVF1 power MOSFETs in a push-pull output stage and delivers about 19 dB of gain. Some kind of low-pass filtering is required (such as Alex, described below) to meet FCC regulatory requirements for harmonic emissions.

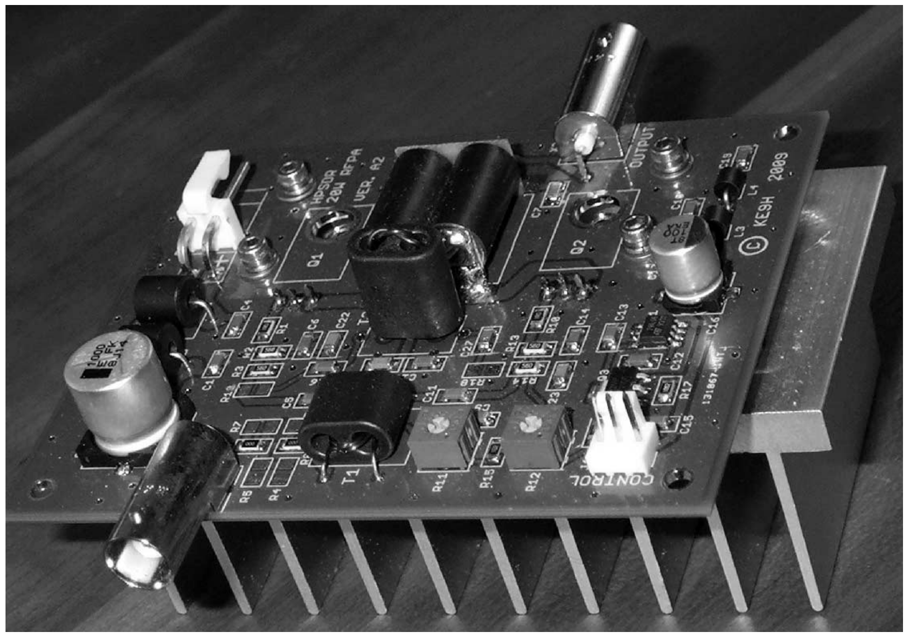


Figure 13 — Pennywhistle 20W RF Power Amplifier.

Pennywhistle documentation is available on the openHPSDR Pennywhistle web page.[32] TAPR offers Pennywhistle as a kit only.[33] The SMT parts are easy to solder 1206 size, and there are a few simple transformers to wind.

Alexiares Transmit/Receive Filters

Alexiares (Alex for short) is a set of filter boards for the openHPSDR project, but these two boards offer much more than just filtering.

The Alex-TX board, shown in Figure 14, not only contains six switched 100 W transmit low-pass filters; it has a transmit/receive (T/R) relay, a 6 m LPF, a directional coupler for power measurements and relays

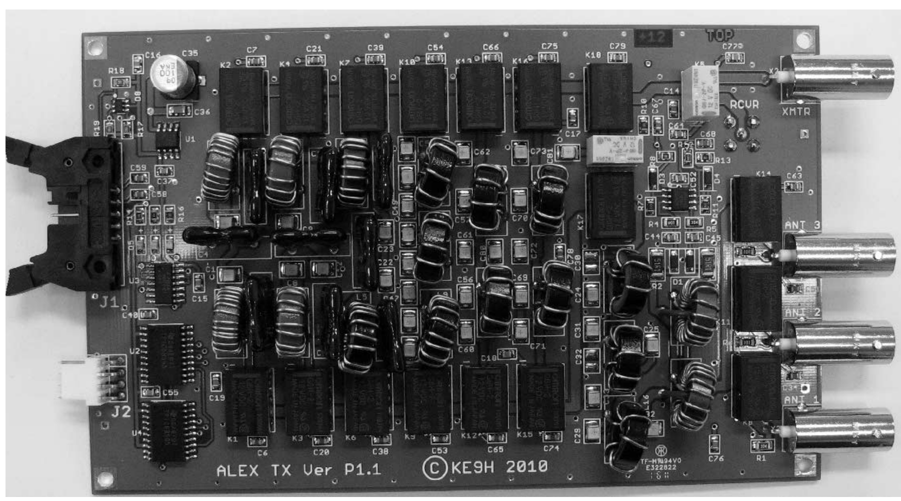


Figure 14 — Alex-TX Low-Pass Filter Board.

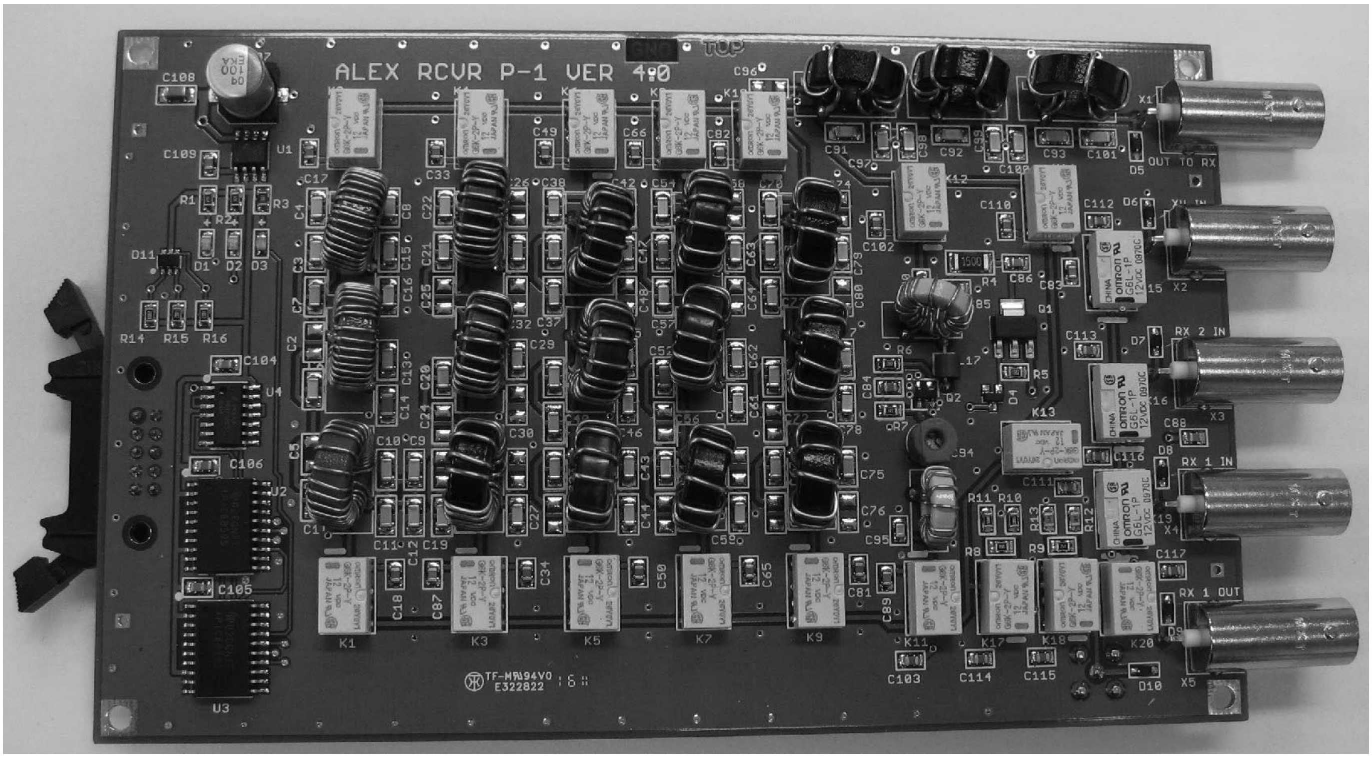


Figure 15 — Alex-RX High-Pass Filter Board.

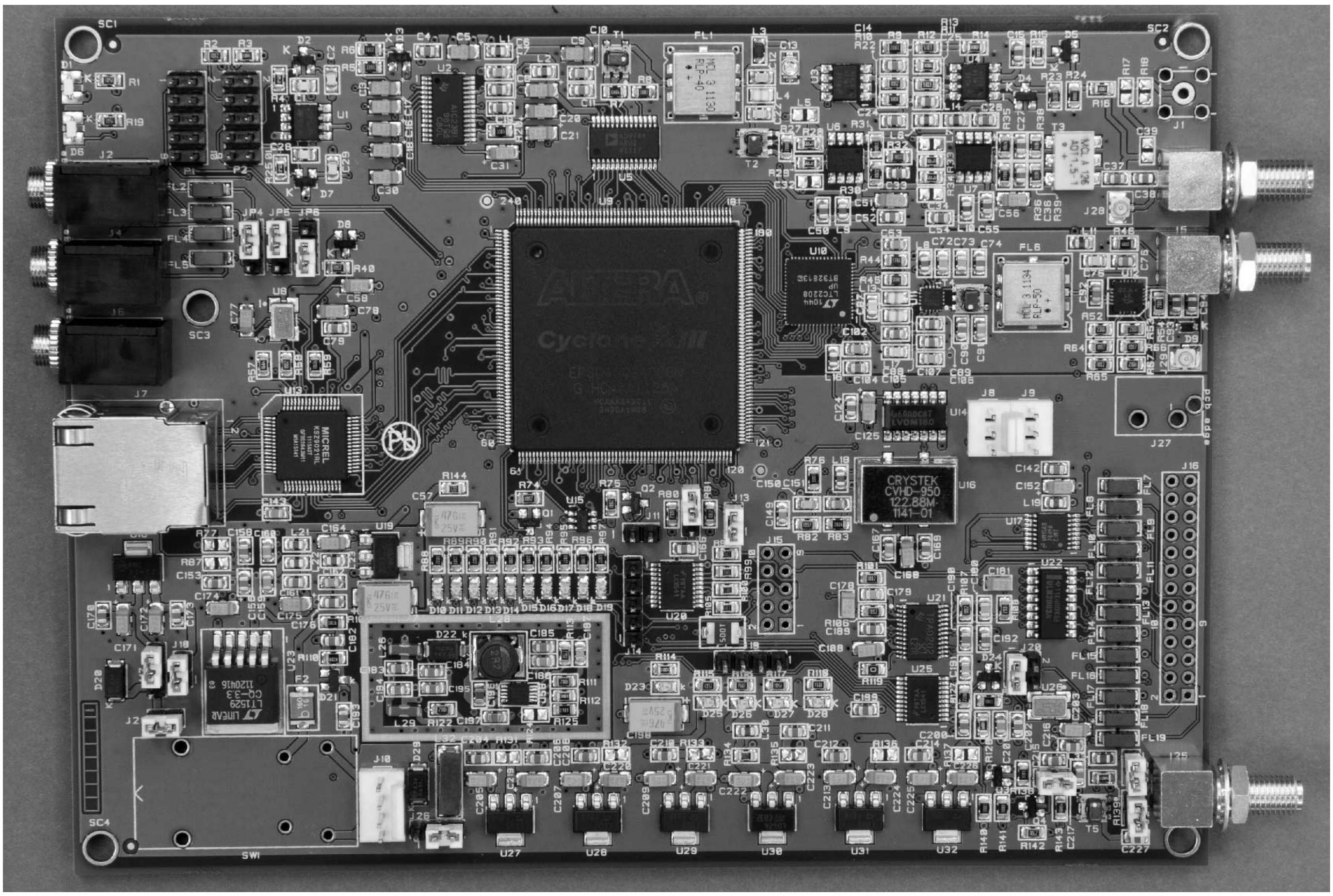


Figure 16 — Hermes Transmitter/Receiver.

to select one of three separate antennas.

The Alex-RX board contains five switched receive high-pass filters, a 6 m LNA, a 55 MHz LPF and a switched 0/10/20/30 dB attenuator. See Figure 15. Alex-RX also has connections for a transverter, two separate receive antennas and an external filter or preamplifier.

Alex-TX and Alex-RX daisy chain together on a 10 pin ribbon cable that supplies power and serial control from an interface on the Mercury receiver board. An off-the-shelf extruded aluminum enclosure with custom end plates is available to mount and completely shield the pair of boards in one enclosure. The Alex-TX and Alex-RX boards are mounted back-to-back in the enclosure to provide as much shielding as possible between the two boards within the enclosure; the circuit board layers are arranged on each board to shield the transmit components from the receive components. John Ackerman, N8UR, has tested Alex filters using laboratory-grade test equipment and published his results.[34]

Alexiares documentation is available on the openHPSDR Alexiares web page.[35] TAPR offers Alex-TX and Alex-RX boards fully assembled and tested.[36] TAPR also offers an enclosure with custom end plates for proper shielding. Note that an enclosure is necessary for Alex boards even if they are mounted within the Pandora enclosure for RF shielding reasons.

Hardware — Single Board openHPSDR

It is not quite a single board when you include the Apollo 15 W power amplifier and automatic Antenna Tuning Unit (ATU), but Hermes does include both the transmitter and receiver on one board. The combination of Hermes and Apollo fits in a standard Eurocard enclosure, yielding a compact and complete 15 W high-performance software defined radio. Just how does it compare to the openHPSDR Atlas system? Read on.

Hermes 500 mW DUC Transmitter/ DDC Receiver

The Hermes receiver section uses the same front-end filter, preamp (LTC6400-20) and ADC (LTC2208) that Mercury uses. See Figure 16. The Hermes transmitter section uses the same DAC (AD9744ARU), filter and RF power amplifier (OPA2674) that Pennylane uses, as well as the same audio CODEC (TLV320AIC23B) and analog input circuit (ADC78H90). The Hermes Ethernet interface uses the same PHY (KSZ9021RL) that Metis uses. A single EP3C40 FPGA replaces the three FPGAs from Metis, Mercury and Pennylane (EP3C40, EP3C25 and EP2C8, respectively). The new layout is really the only variable, and preliminary testing indicates that Hermes is actually quieter on receive than Mercury, and has transmit performance equivalent to Pennylane.

Hermes documentation is available on the openHPSDR Hermes web page.[37] TAPR had built a small production run of assembled and tested Hermes boards but they are no longer available from TAPR.[38] Assembled and tested Hermes boards are available from Apache Labs in India.[39]

Apollo 15 W Power Amplifier with LP Filter Bank and Automatic Antenna Tuner

Apollo, shown in Figure 17, is a companion board to Hermes, and boosts the 500 mW RF output from Hermes to 15 W with a pair of RD15HVF1 MOSFETs in a push-pull amplifier configuration. Apollo contains a set of low-pass filters to reduce transmitter harmonic energy. These LP filters are low-power versions of the filters on the 100 W Alex-TX board. Apollo also has an

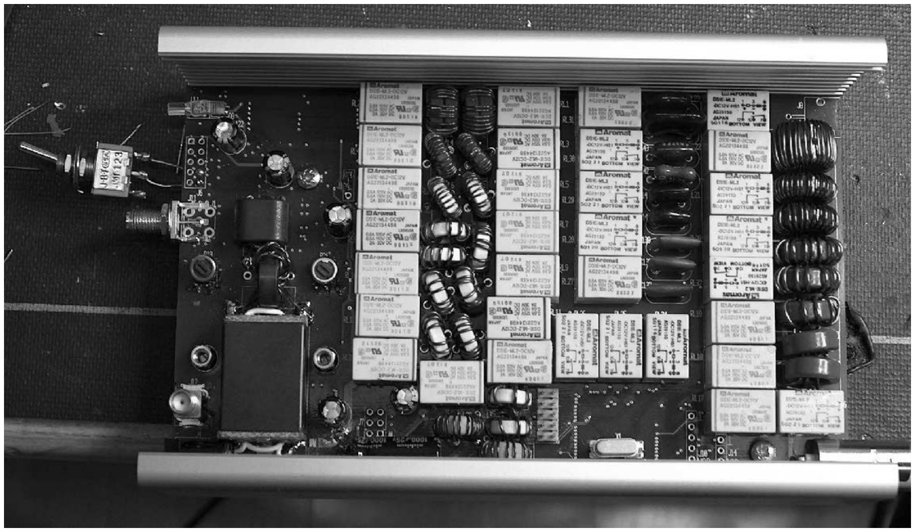


Figure 17 — Prototype Apollo 15 W Power Amplifier, Low Pass Filters, and Antenna Tuning Unit (PA/LPF/ATU).

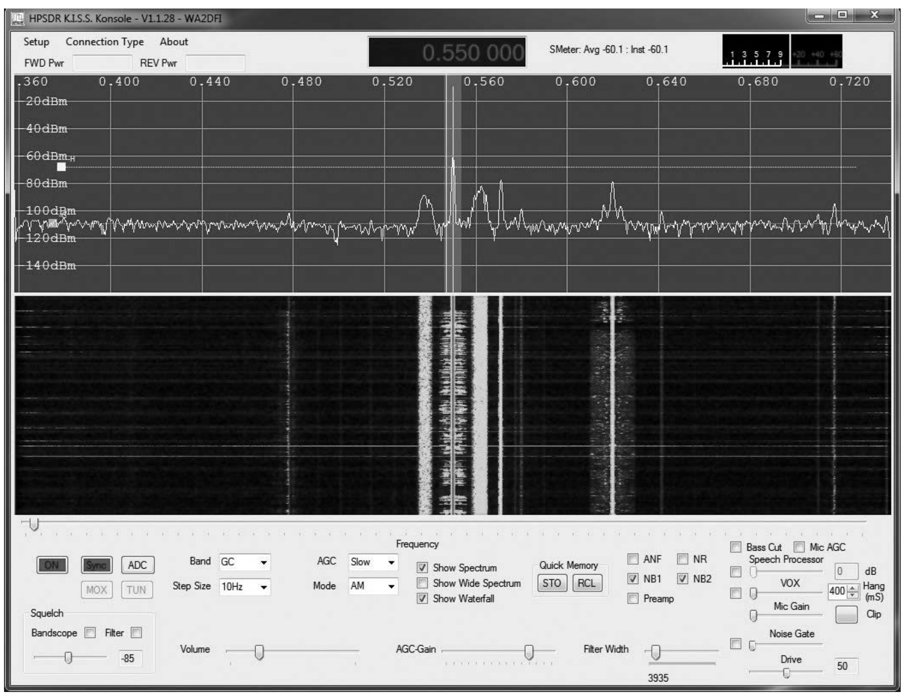


Figure 18 — KISS Konsole Screen.

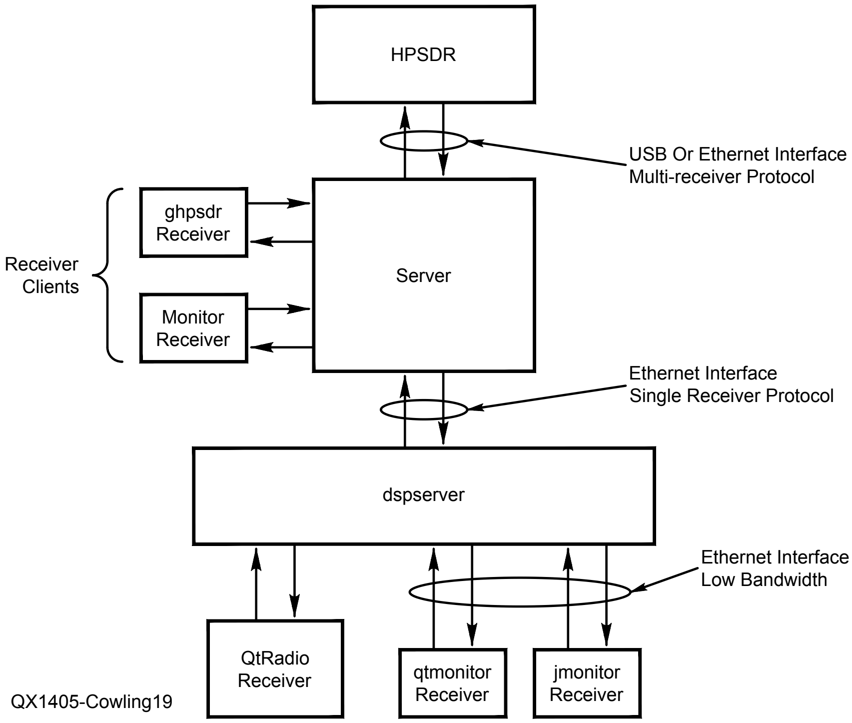


Figure 19 — ghpsdr3-alex server-client.

automatic antenna tuner (ATU) that uses an Atmel AT90 microcontroller in conjunction with an on-board directional coupler to determine the output mismatch and then switch in capacitance and inductance to correct it. Switching is done with latching relays to conserve power. The result is a power amplifier correctly matched and harmonically filtered.

Apollo documentation is available on the openHPSDR Apollo web page.[40] When this article was written, the Apollo board was still under development.

openHPSDR Software

The focus of this article has obviously been on the hardware, but since openHPSDR is a ***Software*** Defined Radio, it stands to reason that there must be *some* software involved. Several good programs are available that allow almost everyone to play, whatever your computing persuasion.

For *Windows* PC users, you can use a derivative of *PowerSDR*™, the GPL software that was developed by FlexRadio Systems® for their product line. Bill Tracey, KD5TFD, originally modified this software to support openHPSDR hardware. The software is currently supported by Doug Wigley, W5WC, and is at revision 3.2.10. It is a full featured program, and works very well in the *Windows XP* and *Windows 7* environments. Joe Martin, K5SO, has a modified version of *PowerSDR*™ 2.2.3 that works with multiple Mercury boards for diversity reception and beam steering. More information on all of these variants of *PowerSDR*™ for openHPSDR can be found on the openHPSDR *PowerSDR* web page.[41]

Kiss Konsole (KK for short) is a basic program for beginners to get their feet wet in SDR and DSP programming. See Figure 18. Written by Phil Harman, VK6PH, KK is written for *Windows* in *C#* using the free VS 2008 IDE, and is currently at version 1.1.28. It is heavily commented and is a good starting point for new SDR programmers. Further information on KISS Konsole can be found on the openHPSDR Kiss Konsole web page.[42]

For Linux users, ghpsdr3-alex is a client-server based software system that supports several types of SDR hardware, including openHPSDR. Alex Lee has modified ghpsdr3, originally written by John Melton, GØORX/N6LYT. The CLI-based server software (see Figure 19) interfaces the HPSDR hardware to the dspserver I/Q data processing software. The dspserver software supports simultaneous connections to multiple clients. Clients run on many different devices (smart phones, tablet PCs, desktop PCs) and are written for different operating systems (*Android*, *iOS*, *Ubuntu*, and *Windows*).

The dspserver and client can be on the same machine or on separate machines. A common scenario is to run the server and dspserver software on one machine, while allowing clients to connect to the dspserver over the Internet. The client PC "controls" the remote radio over the network. The ghpsdr3-alex software was developed on the Ubuntu version of Linux. You can find more information about ghpsdr3-alex on-line.[43]

MAC users are not left out in the cold. Jeremy McDermond, NH6Z, has written a version of openHPSDR software just for *MacOS*: *Heterodyne*. Formerly called *MacHPSDR*, this software runs on *Snow Leopard* (MacOS X 10.6) or later for Intel systems. More information can be found on the openHPSDR Heterodyne web page.[44]

There is other SDR software that will work well with the openHPSDR hardware. A popular *Windows* program is *HDSDR*, derived from WinRAD. *HDSDR* uses an interface file called an ExtIO.dll to support many types of hardware.[45] *HDSDR* ExtIO.dll files are available for the openHPSDR Atlas systems as well as for the Hermes single board radio.[46]

Another interesting program, geared more toward experimenters, is GNURadio.[47] This software runs under *Linux* and has a diverse library of DSP functions and other radio building blocks. Using a graphical tool called GRC (for GNU Radio Companion) you can select, interconnect, and configure blocks from the library to create highly sophisticated radio systems. Tom McDermott, N5EG, has written a source block (GNURadio's name for an input interface) for openHPSDR hardware, providing the gateway from openHPSDR hardware into GNURadio.[48] GNURadio software is all open-source and is constantly growing and expanding in functionality.

Future Evolution

Even though some of the openHPSDR hardware is on the far side of seven years old, it has aged well. The receiver and transmitter front-ends can still be considered state-of-the-art, even though the Atlas bus is imposing its own limitations on system performance. To be fair, Atlas was never designed to support multiple Mercury receiver cards, each running five virtual receivers within their own FPGAs! So how, then, are we to get all of this data over to Metis in order to send it out the Gigabit Ethernet port? A new openHPSDR gen2 (as I am tentatively calling it) backplane, to replace the passive, unterminated Atlas backplane, could be the solution. The new hardware must:

- Provide support for existing gen1 receiver and transmitter cards at moderately improved performance.

• Provide support for new gen2 receiver and transmitter cards at much higher performance.

• Support multiple synchronous receivers and transmitters (both gen1 and gen2) in order to implement diversity reception, pre-distortion transmitter linearization, network analyzers, spectrum analyzers and more.

• Solve the power and heat problems of full Atlas systems, while retaining modularity.

These are lofty goals, but ultimately they will have to be met if we hope to keep the "HP" in HPSDR.

Conclusion

The openHPSDR project is an ongoing evolution of ideas and implementation. There are several places to jump in. Subscribe to the openHPSDR e-mail list or visit the Wiki.[49, 50] Another great place to meet and listen to the developers in real time is the Teamspeak audio session.[51] Everyone is welcome to listen and/or contribute. An audio recording of each week's session is posted on-line every week, just in case you are shy (or live in a time zone that makes listening inconvenient).[52] Please come join us!

Acknowledgements

A project the size and complexity of openHPSDR has many contributors. Take a look at the project pages on the web site and you will see how large and diverse this group really is. Without this dedicated group of innovative and skilled individuals and their selfless contributions of their time and talent, openHPSDR could not exist. I would like to say thank you to all of the past and current contributors, and hope that this article will encourage future contributors to get involved.

Scotty Cowling, WA2DFI, was first licensed in 1967 as WN2DFI, and has been continuously active since that time. An Extra Class licensee, ARRL Life Member, and ARRL Technical Advisor, Scotty is active while mobile on HF CW and on APRS. He is an advisor for Explorer Post 599, a BSA affiliated ham club for teens in the Phoenix, Arizona area. He also enjoys minimalist QRP operating. He has participated in every ARRL Field Day since 1968!

Scotty has been involved in the openHPSDR project for the last 8 years, and has served on the TAPR Board of Directors (2006-2012) and as TAPR Vice President (2011-2012). He is active in the production of openHPSDR components and with other TAPR projects. He is a co-founder of iQuadLabs, LLC, a supplier of openHPSDR systems and other Software Defined Radio components, and is President of Zephyr Engineering, Inc, an engineering consulting firm.

Scotty's professional specialty is FPGA and embedded systems hardware design. He designed his first project with a microprocessor in 1975 and his first FPGA project was in 1987. He holds a BSEE from Rensselaer Polytechnic Institute and an MSEE from Arizona State University.

Notes

[1]TAPR Open Hardware License (OHL): **www.tapr.org/ohl**
[2]TAPR website: **www.tapr.org**
[3]openHPSDR website: **www.openhpsdr.org**
[4]openHPSDR discussion list information: **www.openhpsdr.org/reflector.php**
[5]iQuadlabs web store: **www.iQuadLabs.com**
[6]TAPR Noncommercial Hardware License (NCL, deprecated): **www.tapr.org/NCL**
[7]Atlas documentation: **www.openhpsdr.org/atlas.php**
[8]Atlas kits are available from TAPR: **www.tapr.org/kits_atlas**
[9]Magister documentation: **www.openhpsdr.org/magister.php**
[10]Magister boards are available from TAPR: **www.tapr.org/kits_magister**
[11]Metis documentation: **www.openhpsdr.org/metis.php**
[12]Metis boards are available from TAPR: **www.tapr.org/kits_metis**
[13]Mercury development history: **www.openhpsdr.org/wiki/index.php?title=Mercury_-_Development_History**
[14]An evaluation of Mercury performance: **www.openhpsdr.org/wiki/index.php?title=Mercury_-_intermodulation_(IMD)_tests**
[15]Mercury documentation: **www.openhpsdr.org/mercury.php**
[16]Mercury boards are available from TAPR: **www.tapr.org/kits_merc**
[17]Pennylane documentation: **www.openhpsdr.org/penny.php**
[18]Pennylane boards are available from TAPR: **www.tapr.org/kits_PL**
[19]LPU documentation: **www.openhpsdr.org/LPU.php**
[20]LPU kits are available from TAPR: **www.tapr.org/kits_lpu**
[21]Time nuts web page: **www.leapsecond.com/time-nuts.htm**
[22]Excalibur documentation: **www.openhpsdr.org/excalibur.php**
[23]Excalibur kits are available from TAPR: **www.tapr.org/kits_excalibur**
[24]Softrock boards are available from Five Dash Inc: **www.fivedash.com**
[25]Flex Radio Systems SDR radios: **www.flex-radio.com**
[26]Janus documentation: **www.openhpsdr.org/janus.php**
[27]Janus boards are available from TAPR: **www.tapr.org/kits_janus**
[28]Pinocchio documentation: **www.openhpsdr.org/pinocchio.php**
[29]Pinocchio kits are available from TAPR: **www.tapr.org/kits_pinocchio**
[30]Pandora documentation: **www.openhpsdr.org/pandora.php**
[31]Pandora chassis are available from TAPR: **www.tapr.org/kits_pandora**
[32]Pennywhistle documentation: **www.openhpsdr.org/pennywhistle.php**
[33]Pennywhistle kits are available from TAPR: **www.tapr.org/kits_pw**
[34]Alex test results from N8UR: **www.febo.com/pages/hpsdr/alex**
[35]Alexiares documentation: **www.openhpsdr.org/alex.php**
[36]Alexiares boards and enclosures are available from TAPR: **www.tapr.org/kits_alex**
[37]Hermes documentation: **www.openhpsdr.org/hermes.php**
[38]TAPR had produced an initial run of Hermes boards, but they are no longer available from TAPR. You can find more information about the boards at: **www.tapr.org/kits_hermes**
[39]Assembled and tested Hermes boards are available from Apache Labs in India: **www.apache-labs.com/al-products/1022/OpenHPSDR-Hermes-Transceiver-Card-Assembled--Tested.html**
[40]Apollo documentation: **www.openhpsdr.org/wiki/index.php?title=APOLLO**
[41]openHPSDR *PowerSDR*™ web page: **www.openhpsdr.org/wiki/index.php?title=PowerSDR**
[42]openHPSDR Kiss Konsole web page: **www.openhpsdr.org/wiki/index.php?title=KISS_Konsole**
[43]ghpsdr3-alex client-server software page: **www.napan.ca/ghpsdr3/index.php/Main_Page**
[44]openHPSDR *Heterodyne* web page: **www.openhpsdr.org/wiki/index.php?title=MacHPSDR**
[45]*HDSDR* web page: **www.hdsdr.de**
[46]*HDSDR* Hardware support page: **www.hdsdr.de/hardware**
[47]GNURadio web page: **www.gnuradio.org/redmine/projects/gnuradio/wiki**
[48]GRC Module for openHPSDR Hermes/Metis, Tom McDermott N5EG, DCC 2013: **www.tapr.org/~n5eg/index_files/DCC_2013_Gnuradio_Presentation%20-%20Rev%205.pdf**
[49]openHPSDR e-mail reflector: **www.openhpsdr.org/reflector.php**
[50]openHPSDR Wiki: **www.openhpsdr.org/wiki/index.php?title=HPSDRwiki:Community_Portal**
[51]Teamspeak audio setup instructions: **www.openhpsdr.org/teamspeak.php**
[52]Teamspeak audio archives: **www.hamsdr.com/dnld.aspx**

Pete Horowitz

136 Golden Gate Point #501, Sarasota, FL 34236; phorowitz@live.com

RF Power Amplification Using a High Voltage, High Current IGBT

New insulated gate bipolar transistors offer some amazing power amplifier capabilities, as the author's experiments show.

This article describes experiments, calculations, modeling, and analysis of a specific insulated gate bipolar transistor (IGBT) device. Originally, a casual examination of datasheets turned up several recently available IGBTs in one device family that looked surprisingly capable of very high peak RF power and amplification at useable average power levels, if both the maximum voltage and current capabilities of the device could be used simultaneously. Figure 1 shows three International Rectifier transistors, along with a 360 V dc power supply.

Given their moderate internal device parasitic element makeup, and charge supply requirements to the IGBT gate, circuit design using these bipolar junction devices additionally looked attractive. In fact, the AUIRG4045D, as the smallest device in the family, is rated at 600 V and 6 A. It comes in a diminutive D-Pak package (costing about a dollar each in lots of 100). This transistor proved capable of delivering 3342 W of pulsed power into a 50 Ω load at 20 MHz when driven by 3 or 4 W with no reactive input current correction.[1] See Figure 2.

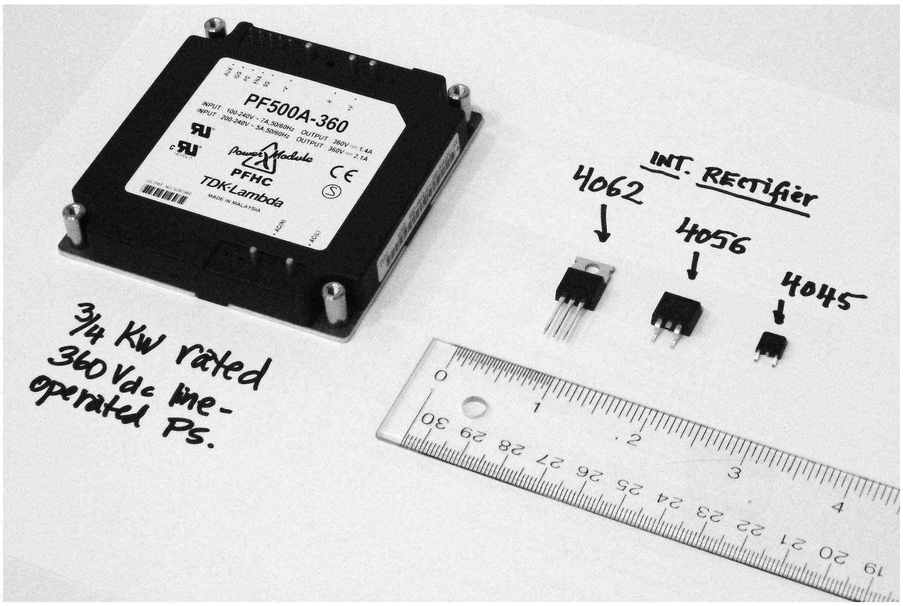


Figure 1 — This photo shows a 360 V dc power supply used in these experiments along with a family of International Rectifier insulated gate bipolar transistors (IGBT) evaluated for the experiments.

The sine wave output to the 50 Ω (actually 48 Ω) load as shown in Figure 3 has been attenuated by 4.9 dB to protect the Tektronix high voltage probe from damage — the actual voltage is 1133 V_{p-p}, which corresponds to 400.5 V_{rms} across the 48 Ω load. That is an output power of 3342 W, with a second harmonic component of –30 dBc = 3.3 W.

The important takeaway here is the somewhat squared-up shape of the IGBT collector voltage waveform shown in Figure 4, which occurs as a result of overdriving this Class A amplifier with a 20 MHz square wave. Since the average power output capability of the device is what counts for Amateur Radio work, what we want to do with these devices is to get the efficiency up. The goal is to shape the collector voltage and current waveforms, so that regions of higher voltage in the collector RF waveform correspond with low current in the IGBT and vice versa.

It can be readily shown that a linear Class A amplifier with transducer efficiency approaching 100% is possible using a square wave input. In this regard, if the Pi network in the present circuit is replaced with a network that in effect terminates the IR4045 collector with 22 Ω of resistance at odd harmonics (f_0, 3 f_0, 5 f_0, and so on) efficiency here will improve and the IR4045 can then operate without ever saturating or

[1]Notes appear on page 20.

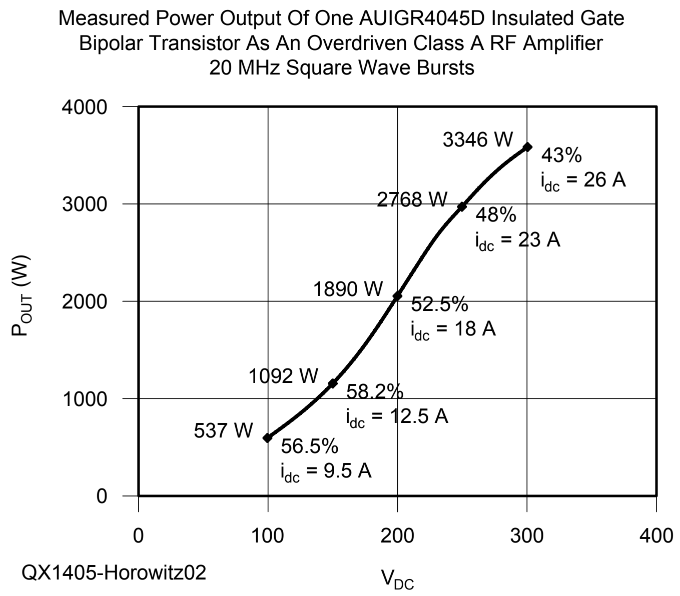


Figure 2 — This graph shows the measured power output of a single AUIRGR4045D IGBT device as an RF amplifier with 20 MHz bursts.

having to turn off on part of each RF cycle (it does both now because of overdrive).[2] The bipolar IR4045 has large 20 MHz RF Power capability, but is not a 20 MHz switch and stores some excess charge near saturation.

Interestingly, this device can also deliver high average power at very high peak power levels, partly because the 4045 has meaningful heat capacity in addition to thermal conductivity when the energy comes in short bunches. This can be elicited by re-labeling the device safe operating area (SOA) curves in joules per pulse, and in interval; and also by examination of the transient thermal impedance curves shown in Figure 25 on the manufacturer's data sheet.[3] When energy is dissipated more continuously, only the thermal conductivity is involved.[4] Figure 5 exemplifies such a situation for a clocked 10 kHz pulse train, where average power above 50 W is maintained for pulse powers up to 5 kW at 50% dc to RF conversion efficiency. The largest device in this family, which has a price tag of $3.33 each in 100 piece lots, calculates out to be able to deliver over 200 W CW and many kilowatts pulsed power also at the 200 W average power level when the dc to RF conversion efficiency is 50%. The high device current capability is not used when not operating at high peak power, however the high voltage property alone may be useful for simplifying circuit designs. Design tradeoffs including cost in comparing IGBTs here with MOSFETs for narrowband use should be done.

Initially reading the manufacturer's documentation turned up statements such as "Ultrafast 8 -30 kHz," and "Application — Air Conditioning Compressor;" hence I decided to buy the smallest version, and to first run a frequency response sweep of the device transconductance. In fact when I first powered it up, the device oscillated strongly at 200 MHz. (That was an exciting moment!) I immediately worked to find simple stabilization methods — a small ferrite ring core or two around the wire to the gate terminal or a 10 Ω resistor in series with the gate. (This was lossy; I did not subsequently use this technique, and it is not necessary at high power levels.)

High peak RF power (watts per pulse) of itself in a transistor or diode is not generally useful; technically sophisticated device modalities, such as TRAPATT action using a 1N5408 diode produces high pulsed power periodic transients at no significant energy (joules) or average power (average watts = joules/pulse × pulses/second) levels. A TRAPATT diode is a PN junction diode, similar to the IMPATT diode, but characterized by the formation of a trapped space-charge plasma within the junction region; it is used in the generation and amplification of microwave power. The name is derived from the description: trapped plasma avalanche transit time diode.

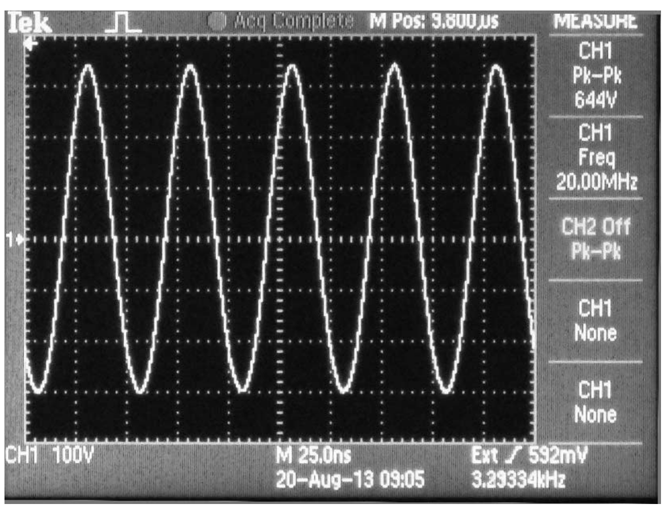


Figure 3 — This oscilloscope pattern shows the output from a single AUIRGR4045D IGBT device, with the signal attenuated 4.9 dB across a 48 Ω load. The second harmonic is –30 dBc (3.3 W).

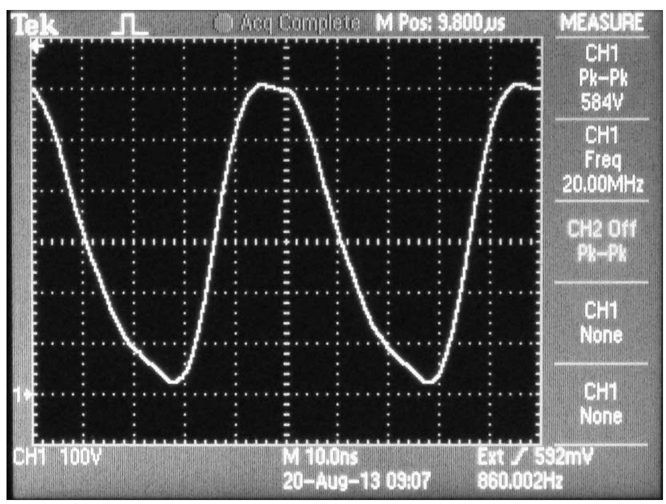


Figure 4 — This oscilloscope pattern shows the voltage waveform across the collector to emitter terminals of the single AUIRGR4045D device.

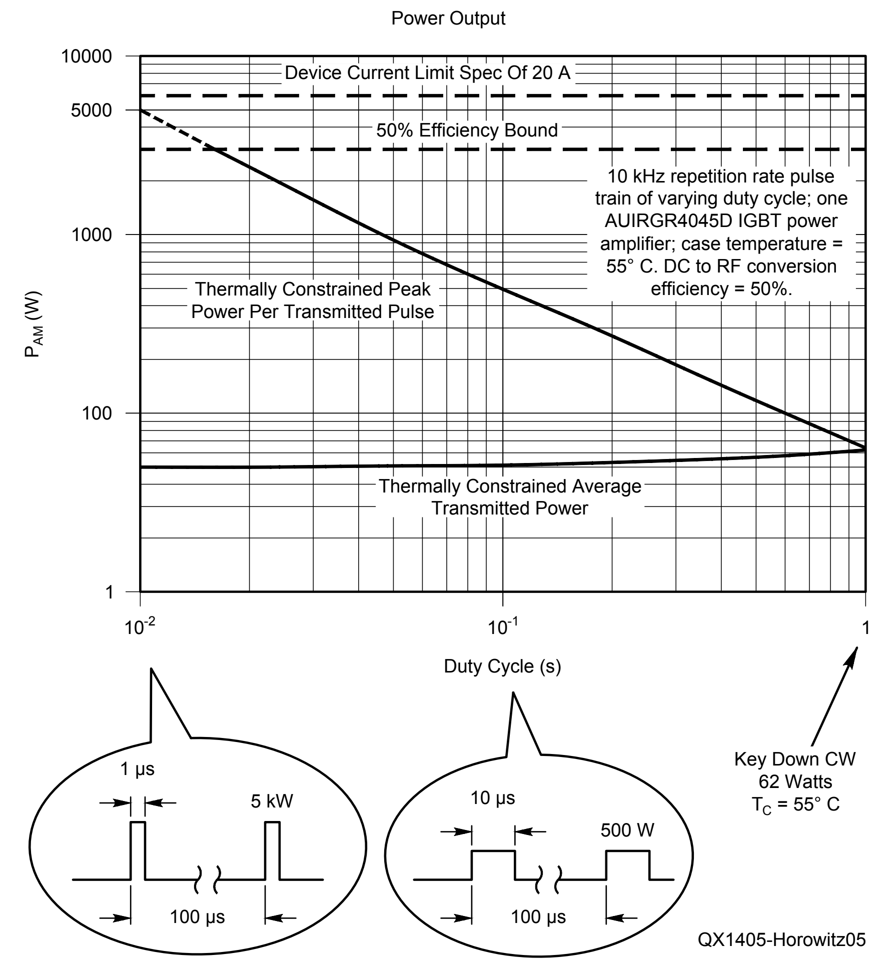


Figure 5 — This graph plots the peak power per transmitted pulse and average power for a 10 kHz repetition rate pulse train of varying duty cycle.

IMPATT diodes with heat sinks, for example, are capable of delivering 3 kW CW at 3 GHz and higher frequencies as oscillators, but have high phase noise and also cannot amplify. There are useful applications for these sorts of high power devices, but not typically in Amateur Radio, where for example, Gunn diodes operating at low CW powers are familiar for microwave work.

Once a device can amplify and then deliver average power it potentially becomes useful for communication systems, minimizing bandwidth to use spectrum space efficiently.

The Experimental Hardware and Design

The hardware shown in Figure 6 consists of the IGBT device and driver pair on the center whiteboard containing the large Pi network. The dc pulsed power source and RF amp bias supply (Instek Power Supply) are shown in black polyethylene boxes towards the right side of the photo. At the back, starting from the left, there is a Rigol waveform generator, a Tektronix oscilloscope in the center, and an Instek triple output dc power supply — also used to supply 20 V dc to the RF driver device. I also used a Tektronix P5100A high voltage/ high frequency/ high impedance scope probe.

This layout forms my test setup. Figure 7 shows a close-up shot of the Pi network and RF head. Figure 8 shows a close-up of the high voltage power supply for driving the RF head. Hopefully this setup will be flexible

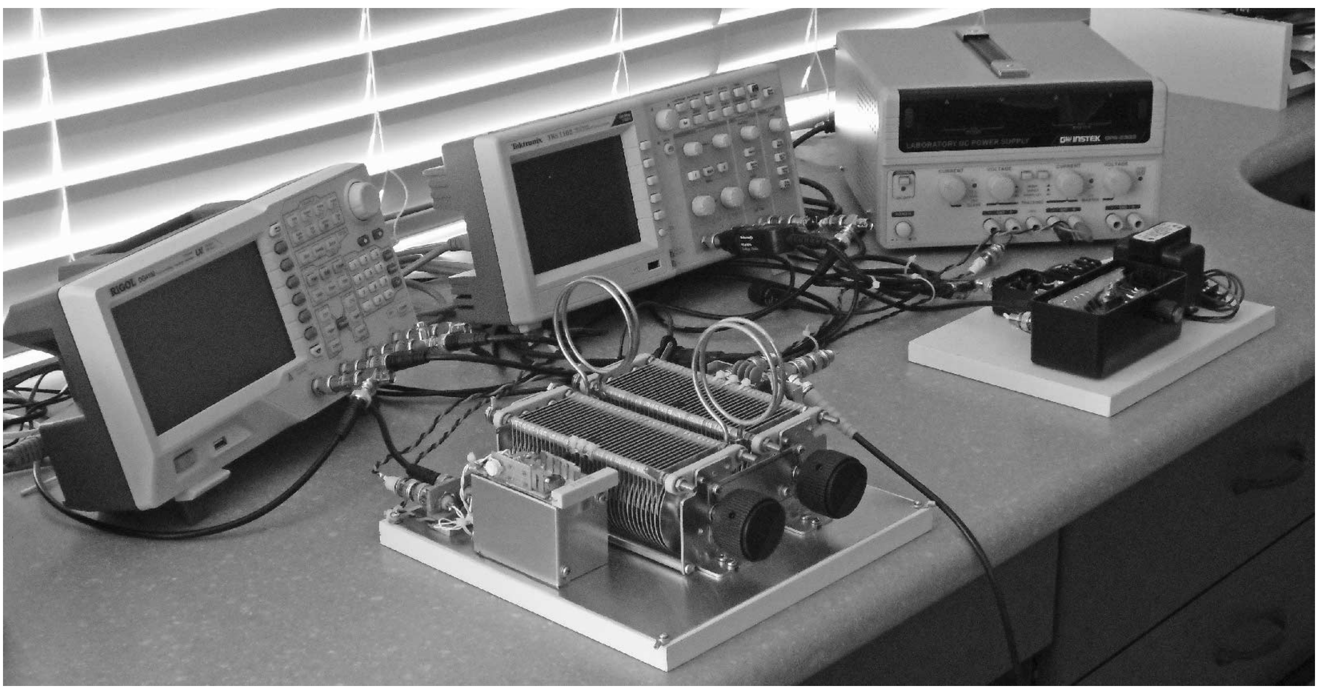


Figure 6 — This photo shows the test setup for the experiments.

enough to accommodate different devices and operating modes as they "suggest themselves." Figure 9 is a schematic diagram of the hardware test setup.

Output Network and 10 kW Pulsed Power Supply

The Pi network and inductor feed along with bypass capacitors and voltage divider output (to protect the high voltage Tektronix probe from damage) have an aggregate unloaded *Q* of 350, which is necessary to keep power losses below 100 W at the 3.3 kW operating level here with resistive ratios below 10 to 1. The present Pi network is capable of greater than 10 kW output.

The pulsed power supply is capable of about 10 kW output at 300 V regulated for 100 μs pulses. At lower output voltages, power capability is correspondingly reduced because of increased power dissipation and transient thermal limits within the AUIRGDC0250 series pass transistor.

RF Driver Design

A 6 W capable laterally diffused metal oxide semiconductor (LDMOS) transistor with f_t = 7 GHz is used as a source follower to present a 3 Ω drive resistance to the IGBT gate out to 150 MHz. This is used to control the effect of harmonic currents generated by the nonlinear charge versus gate voltage transfer property of the IR4045.

The low impedance drive also insures stable amplification by the IGBT. The LDMOS operates at low quiescent dc current, but can produce high current positive RF pulses, which drive the IGBT effectively. The LDMOS transistor used here can be operated at an average power of 3 W or so, and supply adequate drive to result in IGBT 3.3 kW P_{out} levels at 20 MHz.

The 7 GHz LDMOS transistor circuit, though very simple, needs to be constructed with microwave components to avoid instability. Measurement gear may not indicate such high frequency oscillations, but their existence can be readily monitored during use by the dc current draw. As used here, the device dc current drain never changes value despite the wide variation in drive levels (milliwatts to watts) because the pulsed input square wave duty cycle is low.

There is no need to correct the reactive input current to the IGBT, since its power gain of 1000 or so allows the designer the freedom to tailor the drive waveform to help achieve higher efficiency amplification. The Rigol DG4102 Arbitrary Waveform Generator can, in principle, be used to synthesize a waveform point by point to drive the IGBT gate in pursuit of higher RF efficiency

Shielding to Control High Level RF Fields

It is necessary to deal with the high local field strengths to avoid upsetting measurements and the low level circuitry; the 150 pF bypass capacitor from the base to emitter of the 2N3904 suffices in this system. The pictures of the test bed (Figures 6 and 7) have been photographed with the P5100A probe shield removed.

Figure 10 is a photo of the shielded TEK P5100A high voltage probe used to make accurate RF power output measurements in the presence of the strong ambient RF field. I added this aluminum box to the test setup when I realized that the plastic-encased high voltage probe may have been affected by the RF field generated by the amplifier.

Gate Protection Circuitry

I stripped out all of the gate protection circuitry in the interest of speedy testing. For

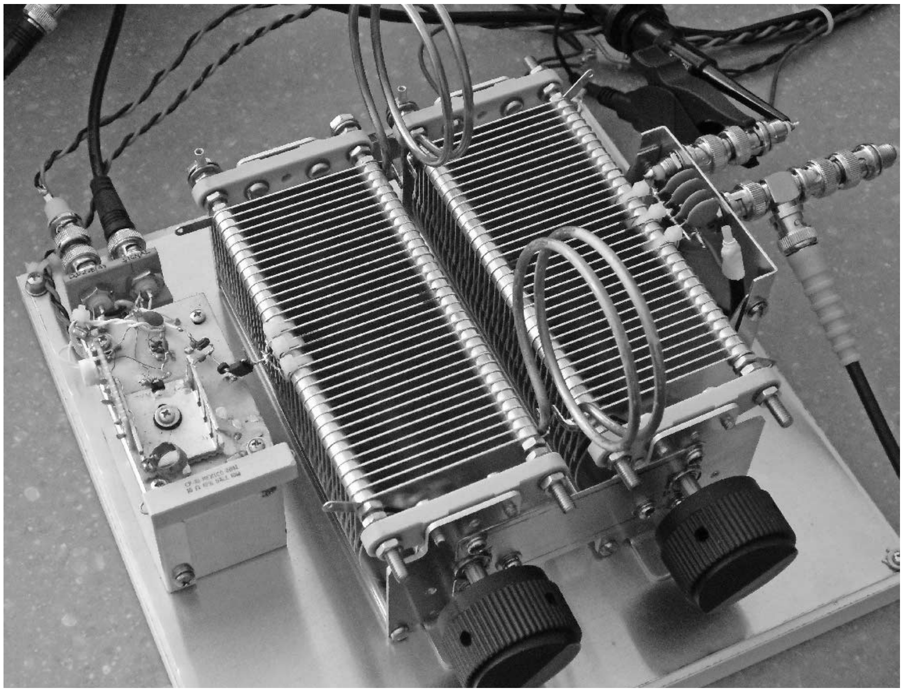


Figure 7 — Here is a close-up view of the Pi network and RF head assembly.

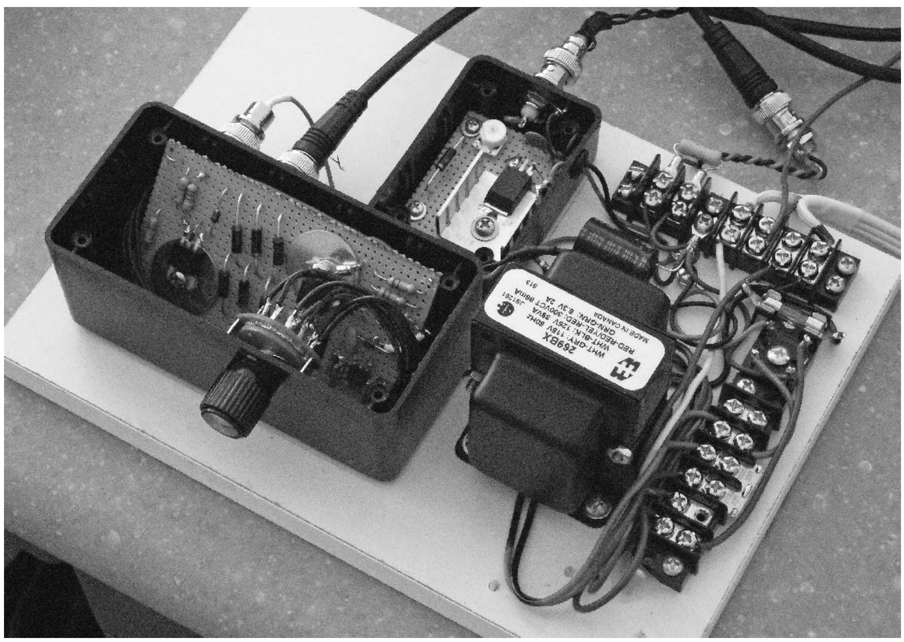


Figure 8 — This photo shows a close-up view of the 10 kW pulsed power supply for the IGBT RF head.

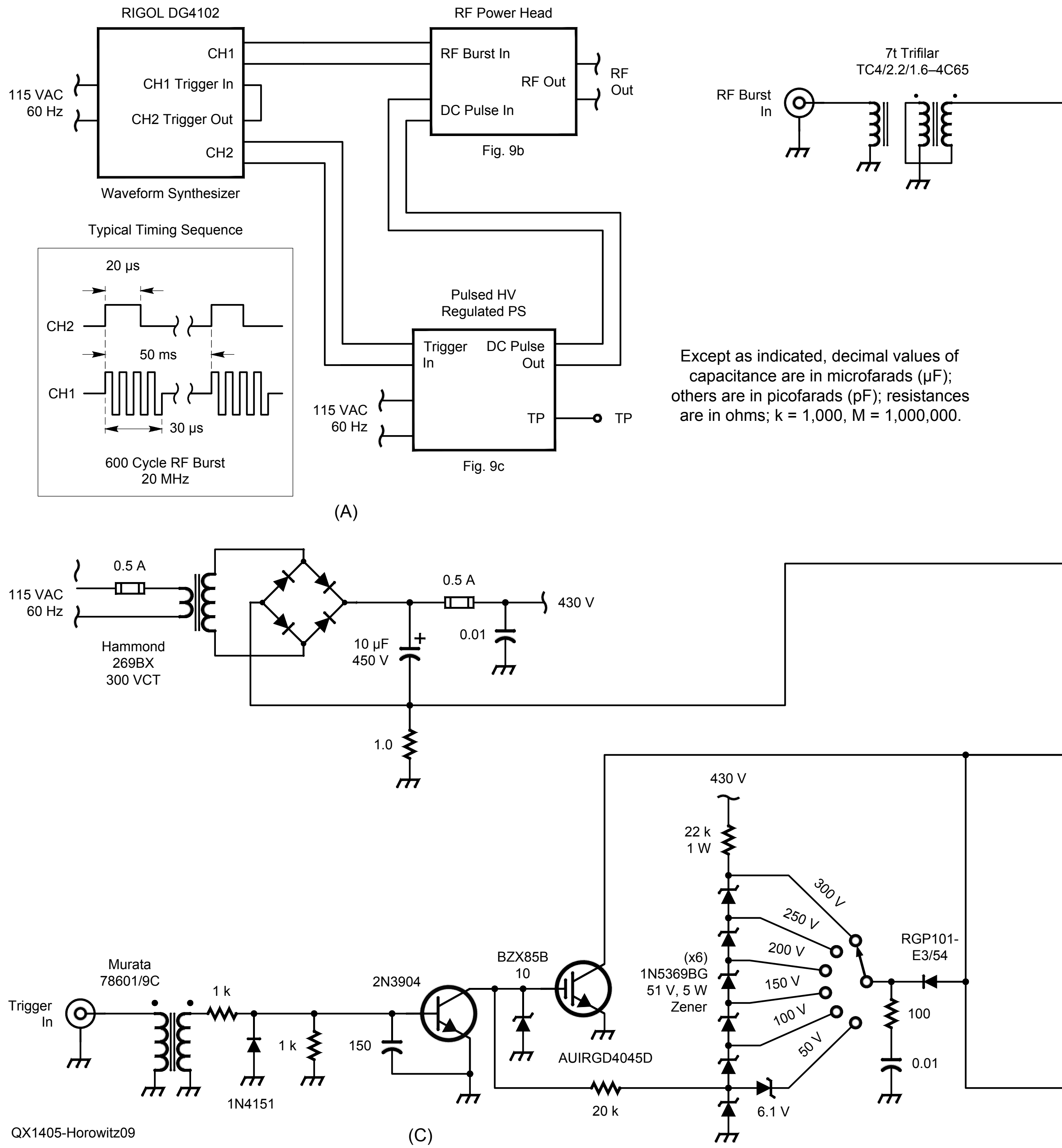
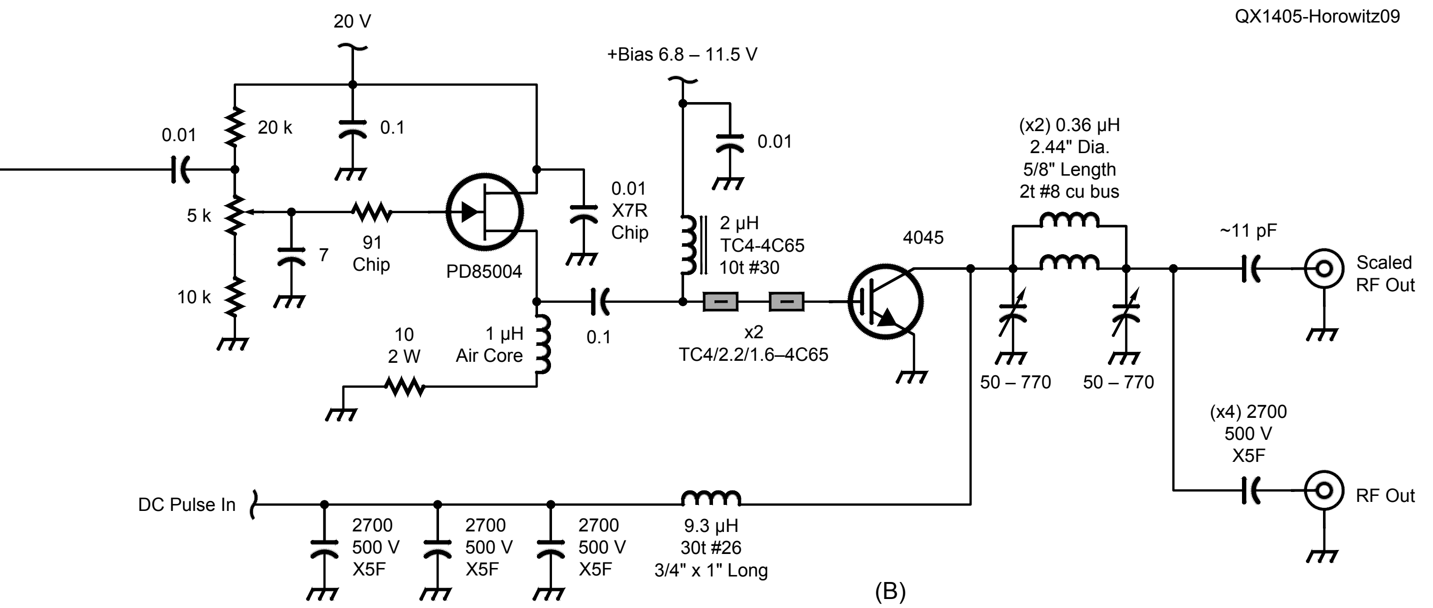
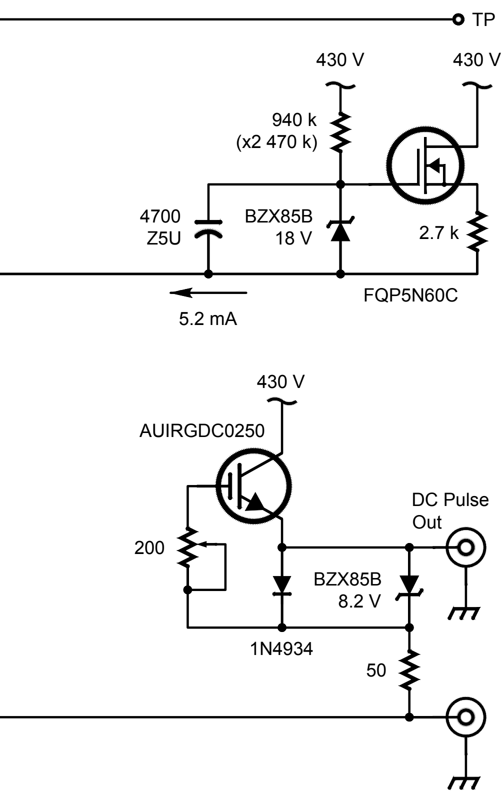


Figure 9 — Here is a detailed schematic diagram of the IGBT RF amplifier test setup.


QX1405-Horowitz09
20 V
+Bias 6.8 – 11.5 V
0.01
20 k
0.1
0.01
(x2) 0.36 µH
2.44" Dia.
5/8" Length
2t #8 cu bus
5 k
91
Chip
7
PD85004
0.01
X7R
Chip
2 µH
TC4-4C65
10t #30
4045
~11 pF
Scaled
RF Out
10 k
10
2 W
1 µH
Air Core
0.1
x2
TC4/2.2/1.6–4C65
50 – 770
50 – 770
(x4) 2700
500 V
X5F
RF Out
DC Pulse In
2700
500 V
X5F
2700
500 V
X5F
2700
500 V
X5F
9.3 µH
30t #26
3/4" x 1" Long

(B)


TP
430 V
430 V
940 k
(x2 470 k)
4700
Z5U
BZX85B
18 V
2.7 k
5.2 mA
FQP5N60C
430 V
AUIRGDC0250
DC Pulse
Out
200
BZX85B
8.2 V
1N4934
50

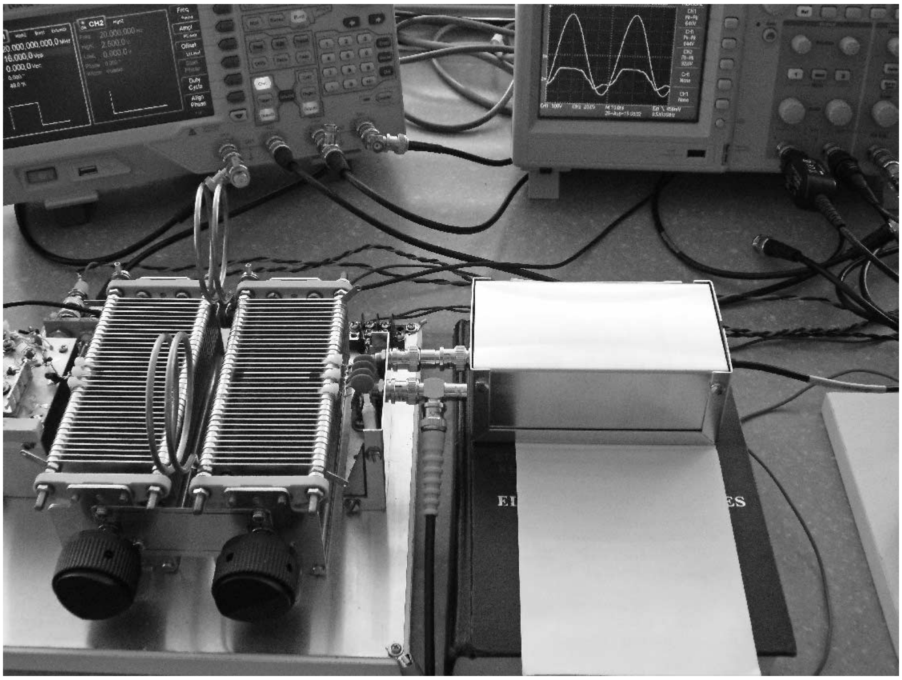


Figure 10 — This photo shows the shielded TEK P5100A high voltage probe used to make accurate RF power output measurements in the presence of the strong ambient RF field.

example, burnouts are quick to troubleshoot when there is a minimum of solid state devices in the circuit. I have not had gates or devices damaged in assembly or tests lately, when the tests have been done with methodical care and attention because the system operation now is familiar to me.

Input Reactive Drive Power Calculation Method

To calculate the input reactive drive power for the IR4045, refer to the manufacturer's data sheet Figure 24. Extend the curve to span –20 to +20 V. This corresponds to charge levels of –7.3 to +16.2 nanocoulombs. Integrate VdQ over this trajectory. At 20MHz this is 100 nanocoulomb volts times 20 MHz = 2 W. A lot of this reactive power can be recovered on the return trajectory with the 1 μH inductor shown in Figure 9 from the PD85004 LDMOS Source to 10 Ω 2 W resistor.

Acknowledgement

Dave Brougham of Fourth Dimension Engineering, Columbia, Maryland did the wideband Spice simulations of the PD85004 LDMOS driver circuit.

Peter Horowitz is an ARRL Member who founded EVI, Inc in 1983. This is a high-tech business specializing in communications systems. He earned a BS in Electrical Engineering from MIT and an MS in Electrical Engineering from New York University. His specialties include RF communications, acoustics and magnetics, prototype hardware design and thick film hybrid manufacturing. He has written many technical articles and contributed to several books, including The Art of Electronics *by Paul Horowitz and Winfield Hill, Cambridge University Press. He holds several patents. Pete is a member of the Columbia Amateur Radio Association, Columbia, Maryland.*

Notes

[1]This data point and the 2.76 kW point, Figure 2, involve RF collector currents considerably exceeding the manufacturer's bound of 20 A for the 4045: There is no data on the datasheet for I_c > 20 A, however the manufacturer's Figure WF.4 does indicate greater capability. The 20 A bound is probably for long term device reliability; we never saw any evidence of P_{out} degradation at the 3.3 kW level, even after hours of continuous pulsing.

[2]D. M. Snyder, "A Theoretical Analysis and Experimental Confirmation of the Optimally Loaded and Overdriven RF Power Amplifier," *IEEE Transactions on Electron Devices*, Volume ED-14, No. 12, Dec 1967.

3 The data sheet for the 4045 transistor is available for download at: **www.irf.com/product-info/datasheets/data/irgr4045dpbf.pdf**.

[4]This description perhaps oversimplifies the situation to some extent. For example, the 4062 manifests somewhat more compli-

Paul Wade, W1GHZ

582 Dubray Rd, Cabot, VT 05647; **w1ghz@arrl.net**

Locked VCXOs for Stable Microwave Local Oscillators with Low Phase Noise

This Article is Reprinted from The Proceedings of Microwave Update 2013.

A good local oscillator (LO) has been a perennial problem for microwave operators. An ideal LO would be stable, provide frequency accuracy, have low phase noise, and be free from birdies and other unwanted artifacts.

Early microwave work relied on inefficient diode multipliers — it was a real challenge just generating enough power to drive a diode mixer. They also required critical tuning. The advent of MMICs, providing cheap gain, solved the power problem and made "no-tune" transverters possible.

The next problem was frequency stability and accuracy. Crystals drift and age and large multiplication factors increase the effect. Microwave operation always involved lots of tuning to find signals. Some operators would bury the main oscillator underground, to keep the temperature constant, and run it continuously. Recent developments in frequency synthesizer chips have made synthesized local oscillators popular. The synthesizers are readily locked to an accurate frequency reference, like a Rubidium standard or a GPS-disciplined oscillator, so that we are able operate on microwaves without a need for tuning.

However, the phase noise generated by almost all synthesizers is significantly worse than a good crystal oscillator. For very weak signals, my experiments suggest that the difference in Minimum Detectable Signal is about 2 dB.[1] On the other hand, 10 GHz MDS tests at the NEWS (North East Weak Signal group — **www.newsvhf.com**) picnic over several years suggest that knowing the frequency of a very weak signal can improve the MDS by up to 5 dB, when listening by ear. The addition of an SDR waterfall display eliminates the unknown frequency problem — all signals appear on the screen — so minimizing phase noise can help to hear very weak signals.

VCXO

Is it possible to have a crystal oscillator, for low phase noise, and frequency accuracy as well? A VCXO, Voltage Controlled Crystal Oscillator, can be adjusted in frequency by varying a control voltage, so we can compensate for drift and aging. One way to do this would be to simply adjust the frequency at the beginning of each operating session to match some reference. Or we can add circuitry to do this automatically and continuously, so that our frequency is locked to some reference.

Why is a VCXO better than a typical synthesizer? Most synthesizers use a VCO, Voltage Controlled Oscillator, a free-running oscillator with wide tuning range. The VCO has poor inherent stability without the synthesizer controlling it, and the wide tuning range makes it very sensitive to noise.

As an example, a typical local oscillator frequency for 1296 MHz operation is 1152 MHz. A synthesizer for this frequency might use a VCO like the Minicircuits ROS-1285-119+, which tunes from 1115 to 1285 MHz for a tuning voltage of 0 to 5 V. Tuning sensitivity is specified as about 56 MHz per volt — one microvolt of noise will move the frequency by 56 Hz, and a millivolt of noise will move the frequency by 56 kHz. A VCXO has much lower tuning sensitivity; I measured a 96 MHz VCXO to vary 16.5 kHz for a tuning voltage of 0 to 3.3 V, so the tuning sensitivity is about 5 kHz per volt, roughly 10,000 times less sensitive. When multiplied by 12 to 1152 MHz, the tuning sensitivity is about 60 kHz per volt,

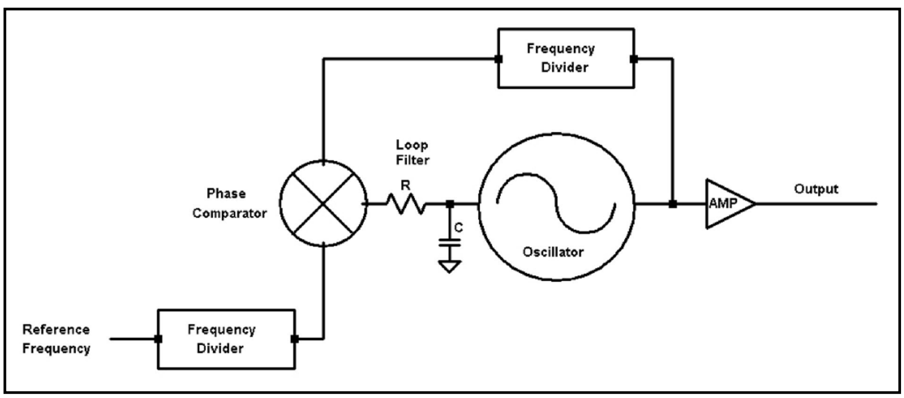


Figure 1 — Simplified Frequency Synthesizer

[1]Notes appear on page 29.

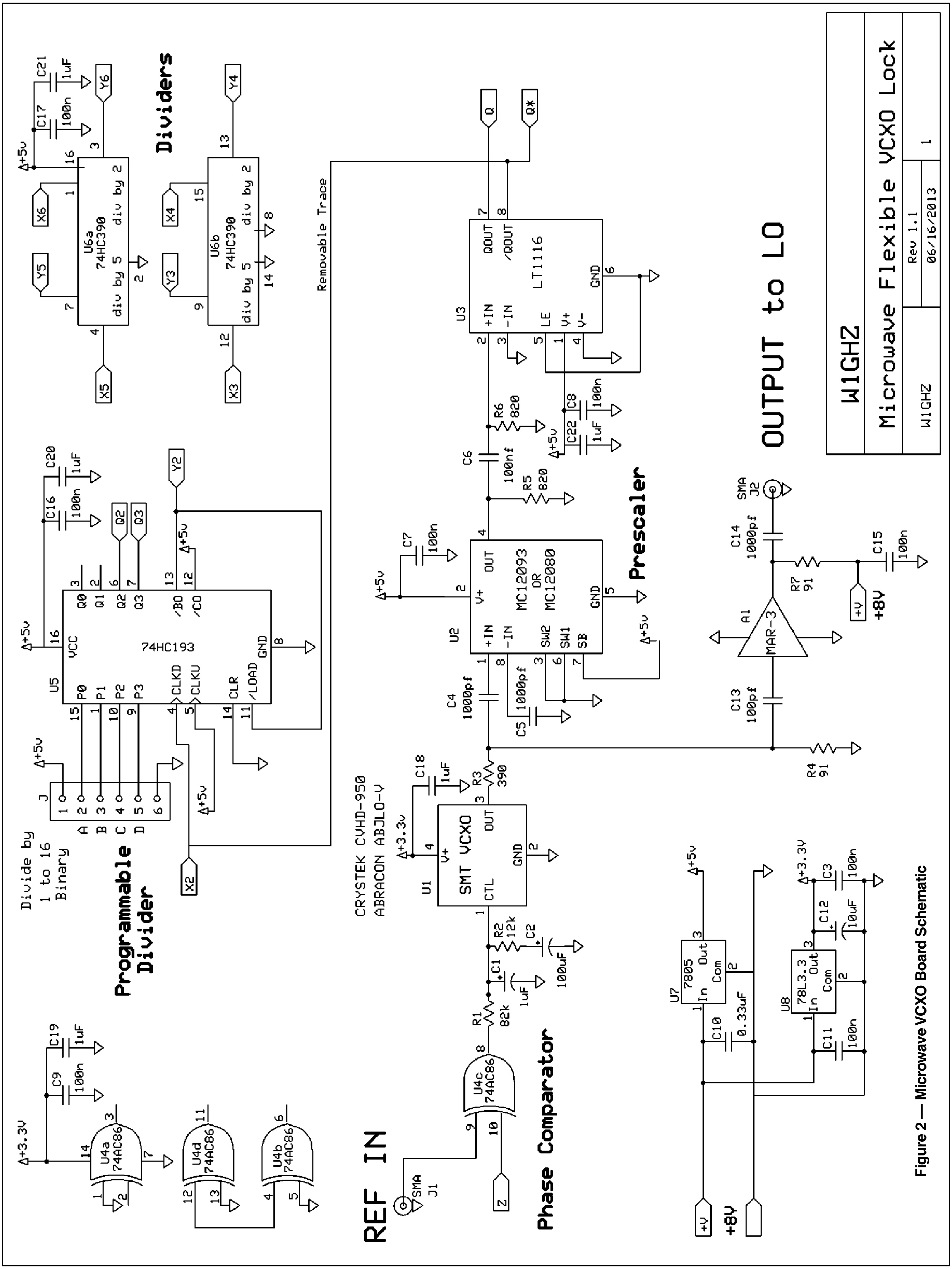


Figure 2 — Microwave VCXO Board Schematic

still 1000 times better than the VCO. One millivolt of noise will move the VCXO frequency by 60 Hz, or the VCO frequency by 56 kHz. Even one microvolt of noise will move the VCO frequency by 56 Hz. Any noise voltage is constantly moving the oscillator frequency around — we call it phase noise. In real equipment, it is easy to keep noise well below a millivolt, but getting below a microvolt is a lot harder. It is really hard to keep the noise on the tuning voltage low enough to eliminate it as a source of phase noise (of course, there are other sources of noise).

Frequency Synthesizers

A frequency synthesizer generates a desired frequency by comparing the frequency of the oscillator in the synthesizer with a reference frequency and correcting the oscillator frequency until it is on the desired frequency. Since the desired frequency is not the same as the reference (otherwise the synthesizer is not required), the frequencies must be converted to a common frequency for comparison. One technique is to generate harmonics of the reference and do the comparison at the oscillator frequency. A more common and versatile technique, shown in Figure 1, is to digitally divide one or both frequencies to a lower common frequency and then compare.

Some of the division schemes can get pretty fancy, in order to generate a wide range of frequencies with very small step increments, and to change frequencies quickly. Most of them also use some form of processor chip. All of this complexity tends to generate digital noise.

For the VCXO, we are making a very simple synthesizer, hard-wired for one specific frequency. The goal is to generate a clean, stable signal. For simplicity, only integer dividers are used, using ordinary CMOS logic devices. This limits the choice of frequencies, but only a limited number of VCXO devices are readily available; however, several of them are quite useful for microwave Local Oscillators.

The perceived wisdom for minimizing phase noise in synthesizers includes:

• A stable oscillator with small tuning range
• High comparison frequency
• Integer dividers
• Slow loop filter time constant
• Clean, stable power supply

I have tried to incorporate all of these in the VCXO circuit.

VCXO Board

The schematic of the VCXO board is shown in Figure 2. The board assumes a packaged VCXO, but it could also control a homebrew one to operate at some other frequency. The VCXO frequency range is roughly 50 to 200 MHz, and the CMOS logic is only guaranteed to operate up to about 30 MHz, so the first divider is a prescaler chip. Several prescalers fit in this location, allowing divide ratios of 2 to 80, selected by three selection pins. The prescalers will operate up to at least 1100 MHz, so higher frequency oscillators are also possible. After the prescaler, two CMOS logic chips are available — U5 can be wired for any divisor between 2 and 16, while U6 has two sections each with fixed divisors of 2 and 5. To divide by 28, for example, one chip would divide by 14 and the other by 2. All the dividers are programmed by jumper wires — simple, effective, and fine for fixed frequency operation.

The programmable divider sections may be used for either the oscillator frequency

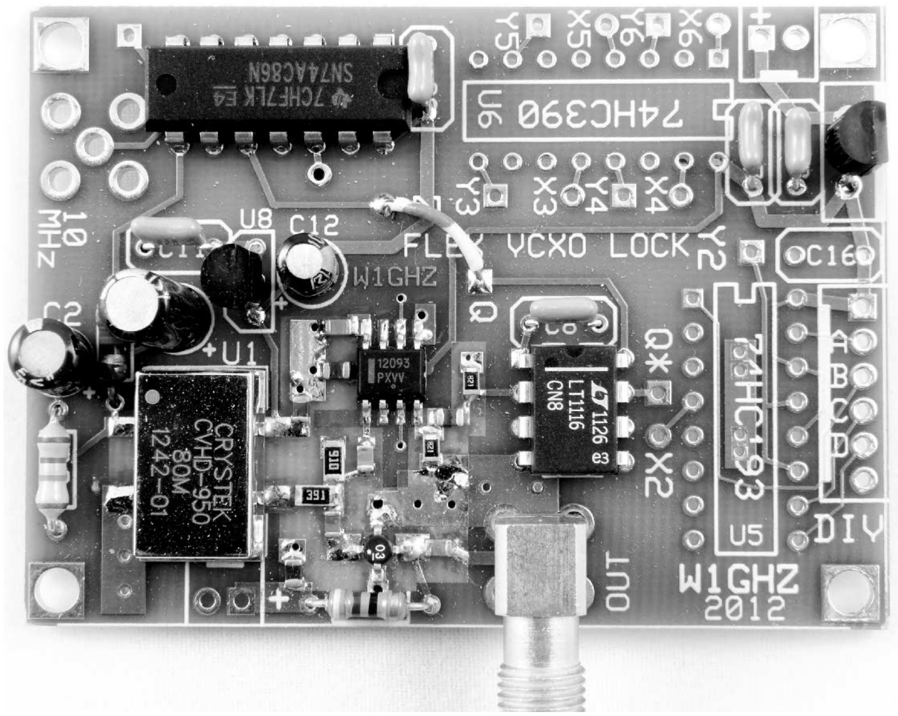


Figure 3 — 80 MHz VCXO board

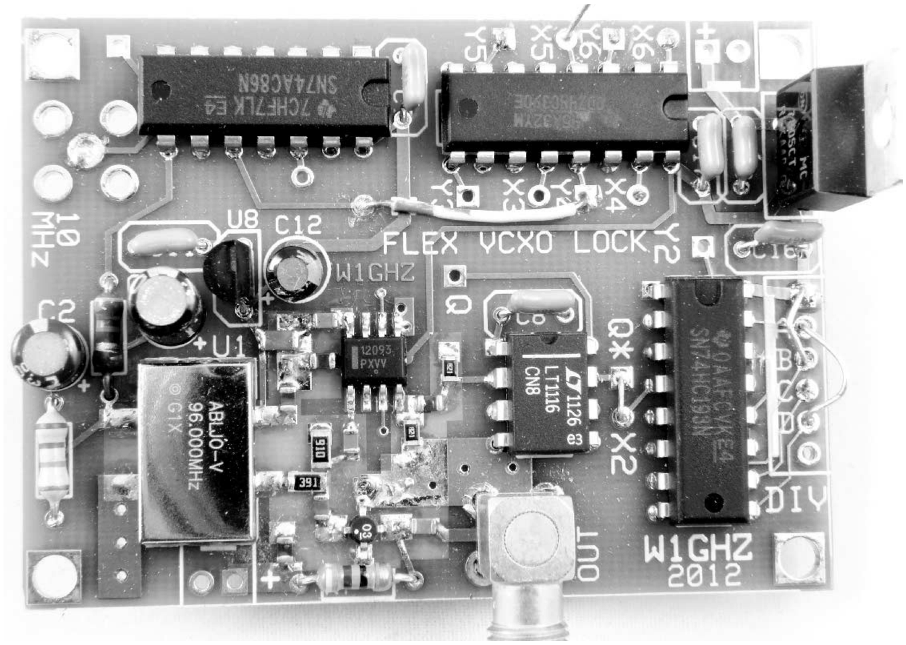


Figure 4 — 96 MHz VCXO board

or the reference frequency, in order to arrive at a common frequency. Examples will be shown for all the useful VCXO frequencies that I have found. At the common frequency, both signals go to the comparator — this is a logical Exclusive-OR (XOR) gate. The XOR gate output is high when both signals are the same and low when they are different. When the frequencies are the same, the output will be high for part of each cycle and low for part of each cycle. The output is averaged to create a DC tuning voltage, which just happens to be the voltage for the desired frequency. If the frequencies were different, the voltage would be higher or lower, forcing the oscillator toward the desired frequency.

The output averaging is done in the loop filter, an RC filter with a long time constant, so that the oscillator is gently guided onto frequency. During testing, I can see the oscillator waveform on an oscilloscope shift frequency over several seconds to line up with the reference waveform.

Finally, there is a MMIC buffer amplifier for the oscillator output.

While all my examples use a 10 MHz reference frequency, other references can also work. If you have a good reference oscillator at another frequency, use it. Only integer dividers are available, so an integer VCXO requires an integer reference. On the other hand, to lock an odd frequency, perhaps for a beacon, an odd reference is required that can achieve a common reference frequency. A couple of possibilities are a programmable Rubidium source or a reverse DDS source.

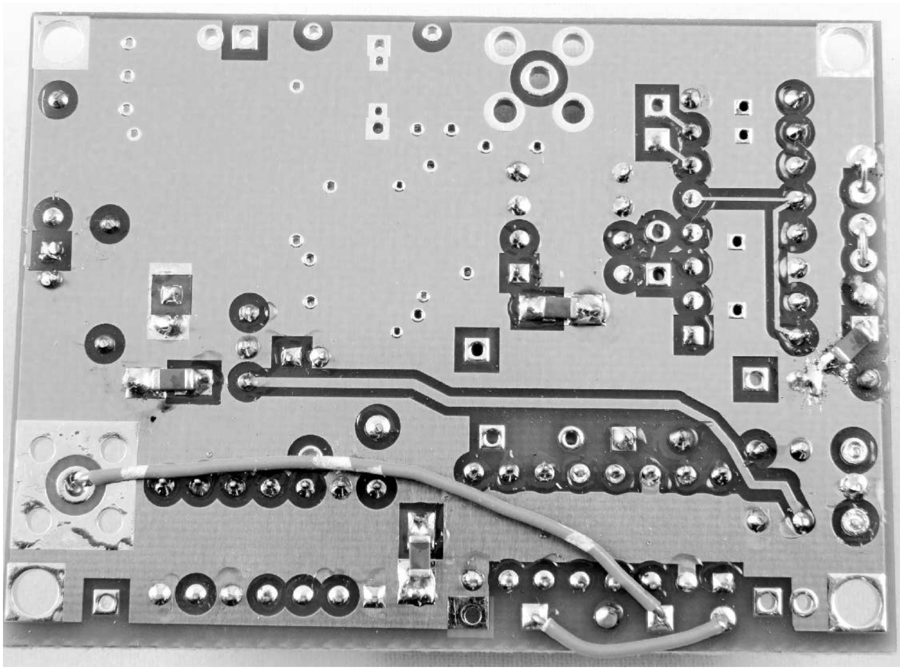


Figure 5 — Bottom side of 96 MHz VCXO board

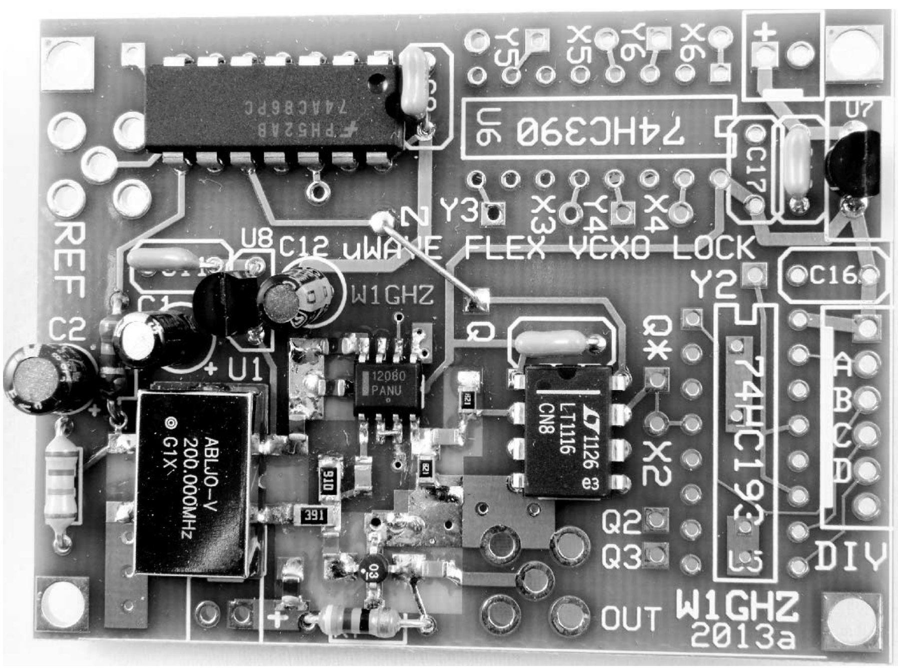


Figure 6 — 200 MHz VCXO board

VCXO Examples

80 MHz — The simplest example is an 80 MHz VXCO, useful as the LO source to multiply to 2160 MHz (2304), 3600 MHz (3456), and 10800 MHz (10368). The board is shown in Figure 3 — 80 MHz is divided by 8 in the MC12093 prescaler chip (select ÷ 8 by grounding pins 3 and 6) to 10 MHz, then compared directly with the 10 MHz reference. This board locks right up with a clean output.

96 MHz — Another very useful frequency is 96 MHz, which is multiplied to 1152 MHz for the popular LO for 1296, with harmonics providing markers on many microwave bands. On the board shown in Figure 4, 96 MHz is divided by 8 in the MC12093 prescaler chip to 12 MHz. Then we must divide by 12 to 1 MHz, and divide the 10 MHz reference by 10 to compare at 1 MHz. I tried dividing by 12 directly in the 74HC193 chip, but the output was very asymmetric, with only a narrow pulse — this would not work well with the XOR comparator. Instead, the 74HC193 is wired to divide by six, followed by a divide by two in the 74HC390 chip — a divide-by-two always generates a square wave output. Other sections of the 74CH390 divide the 10 MHz reference by five, then two, to provide a square wave as well. This board also locks right up; the output has small spurs 1 MHz on each side of the main output.

The divisor of the74HC193 chip, U5, is programmed by the ABCD jumpers on the right side or the board with the binary version of the divisor. For instance, a divisor of 12 = 1100 binary, so jumpers A and B are wired HIGH, to +5 V, and jumpers C and D are wired LOW, to ground. The board in Figure 4 is wired for a divisor of 6 = 0110 binary.

The bottom of the board, shown in Figure 5, has the wiring connections to divide the 10 MHz reference by 10 to 1 MHz. The input for the 10 MHz reference is at location X5.

200 MHz — provides a stable LO for 222 MHz; once you get away from 28 MHz, any IF is good. This one, shown in Figure 6, is nearly as simple as the 80 MHz, requiring only a prescaler. The prescaler is changed to a MC12080 programmed to divide by 20 for 10 MHz output. Selection of divide-by-20 requires changing pin 3 from ground to high, by cutting the PCB trace and adding a jumper wire to pin 2. This board locks right up with a clean output.

120 MHz — provides a stable LO for 144 MHz. As shown in Figure 7, the prescaler is a MC12080 programmed to divide by 10, to 12 MHz. Selection of divide-by-10 requires changing both pins 3 and 6 from ground to high, by cutting the PCB traces and adding jumper wires from pin 3 to pin 2 and from pin 6 to pin 7. The rest of the wiring is the same as the 96 MHz board. This board also locks right up; the output has small spurs 1 MHz on each side of the main output.

108 MHz — I haven't yet found a VCXO for this frequency, but WA1MBA says it would be a useful frequency, and it provides another example. The prescaler must divide by 9 to 12 MHz; the MC12026 can be programmed for this. The rest of the wiring is the same as the 96 MHz board.

Multiplier Board

To multiply from the VCXO board to the microwave LO frequency, I use the LO boards for my simple rover transverters, replacing the oscillator with the signal from the VCXO board.[2] The 80 MHz VCXO drives the 720 MHz LO board — the frequency is further multiplied in the 2304 or 3456 MHz transverter. The multiplier board in photo in Figure 8 shows the input from the VCXO bypassing the oscillator section.

The 1152 MHz LO board originally used a 64 MHz crystal oscillator. To operate with 96 MHz input from the VCXO, the filter after the first tripler must be tuned to 288 MHz. With the 64 MHz oscillator, the combline filter was tuned to 192 MHz by capacitive loading of 36 pF (two 18 pF capacitors in parallel for lower loss). From the chart, Figure 7, in my transverter article, I estimated the required capacitance for 288 MHz as 22 pF.[2] The filter response in Figure 9 shows that the filter is tuned slightly high, with 288 MHz at the edge of the response. Increasing the capacitance to 23 pF, an18 pF in parallel with 5 pF, centered the filter near 288 MHz. The parallel capacitors slightly reduced the loss. The input connection to this multiplier board, shown in Figure 10, is similar to the one in Figure 8.

Phase Noise

While there is no reason to expect the VCXO to be more stable than a synthesizer,

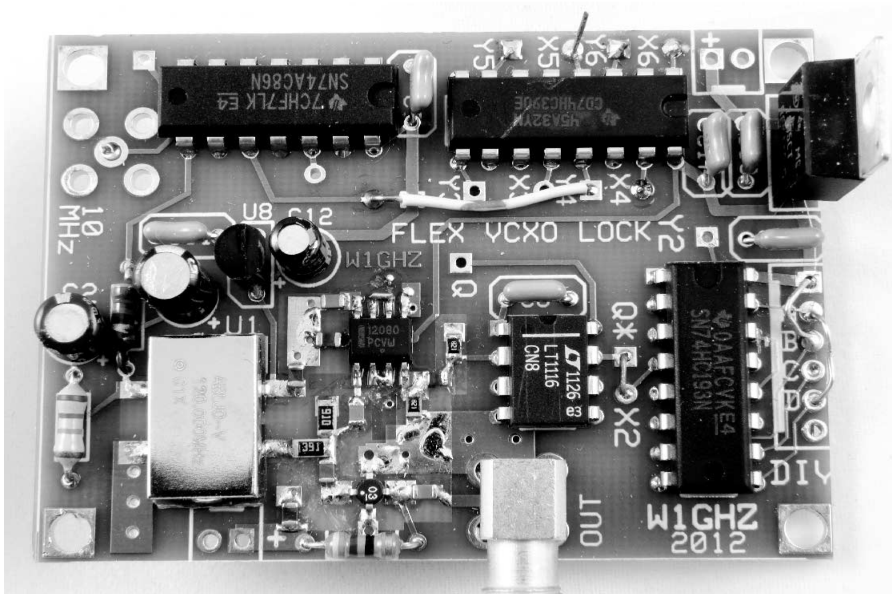


Figure 7 — 120 MHz VCXO board

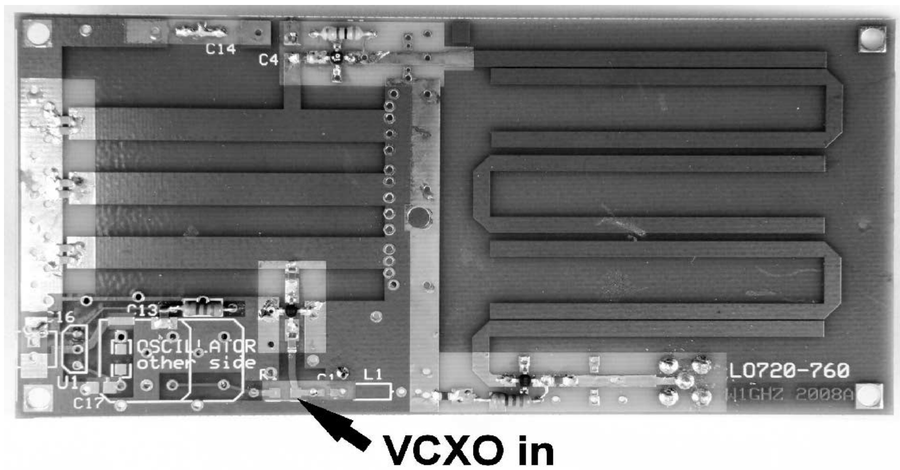


Figure 8 — 720 MHz LO Multiplier board configured for external VCXO

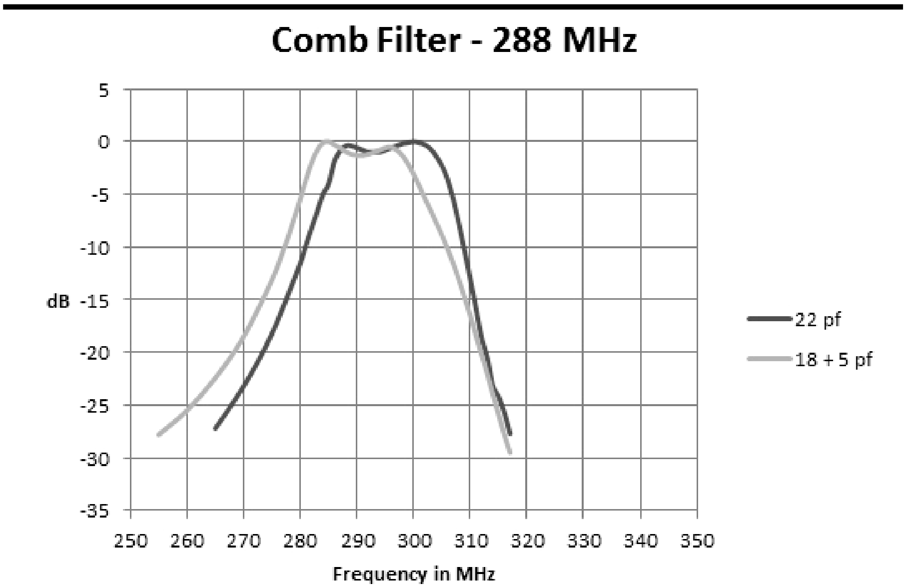


Figure 9 — Response of Combline Filter for 288 MHz

our goal is to reduce phase noise. Figure 11 shows the phase noise starting with a pretty good TCXO, the 80 MHz VCXO, and the VCXO multiplied to 720 MHz. Frequency multiplication increases phase noise by 20 log_{10} (N) dB, where N is the multiplication factor. Thus, we expect an 8 times multiplication to add about 18 dB of phase noise — the 80 MHz VCXO has about 18 dB more phase noise in the mid-range, but is somewhat worse at very low offset frequencies. A further 9 times multiplication to 720 MHz should add another 19 dB, which we can see in the mid-range as well. The 720 MHz curve generally has the same shape as the 10 MHz source, so most of the difference is due to frequency multiplication. Figure 12 compares the VCXO-based 720 MHz LO with one using an ordinary crystal oscillator. The VCXO is a few dB worse, but still really good.

An LO for 1152 MHz, starting with a 96 MHz VCXO, also looks very good. In Figure 13, we can see the phase noise for the same 10 MHz TCXO, the locked 96 MHz VCXO, and the 1152 MHz output from the multiplier board. The same phase noise multiplication with frequency multiplication is evident.

The two VCXO-base LO chains are compared in Figure 14. We would expect the 1152 MHz phase noise to be about 4 dB worse due to the greater frequency multiplication. However, the 1152 MHz phase noise is lower than the 720 MHz version. A possible explanation is different crystals — the 80 MHz VCXO is made by Crystek, while the 96 MHz VCXO is made by Abracon.

How do the VCXO-based LO chains compare with a synthesized LO? Figure 15 shows a comparison with several popular synthesizers, all operating at 1152 MHz. The VCXO is at least 10 dB better than the best synthesizers in the important frequency offset range, the SSB passband. The most common synthesizers, the ApolLO (curve marked N5AC A32) and the Qualcomm, have phase noise nearly 40 dB worse than the VCXO.

The ApolLO synthesizer was used for the MDS tests that showed a 2 dB difference from a crystal oscillator LO — and the difference in phase noise is ~40 dB.[1] Would some of the better synthesizers, only 10 or 20 dB worse than a crystal source, be any different in MDS? Clearly, more work is needed.

One problem is that I do not have equipment for phase noise measurement — measurement of the very low phase noise of a good oscillator is difficult. I rely on test equipment at various conferences. The results here were measured at the Eastern VHF/UHF Conference in 2012 and 2013, and at Microwave Update 2012. A few oscillators are measured at all three sessions

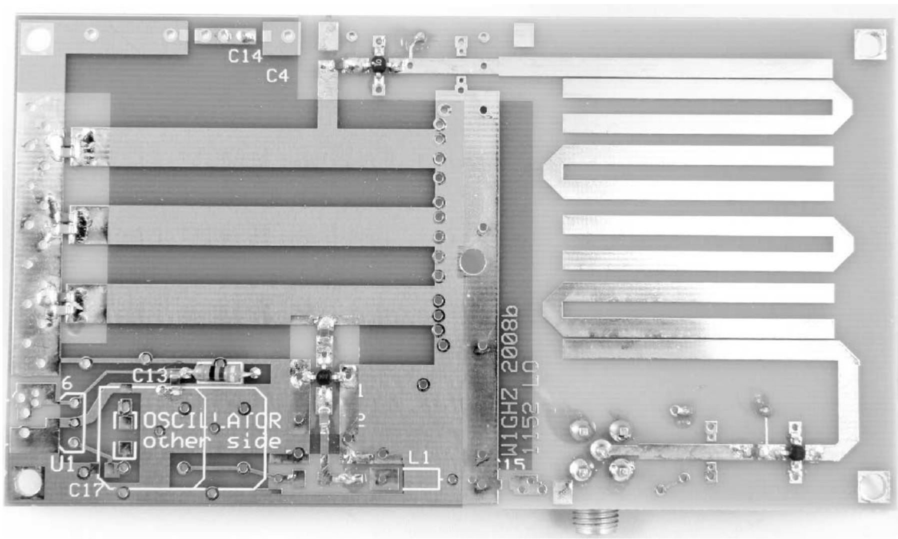


Figure 10 —1152 MHz multiplier board configured for external 96 MHz VCXO input

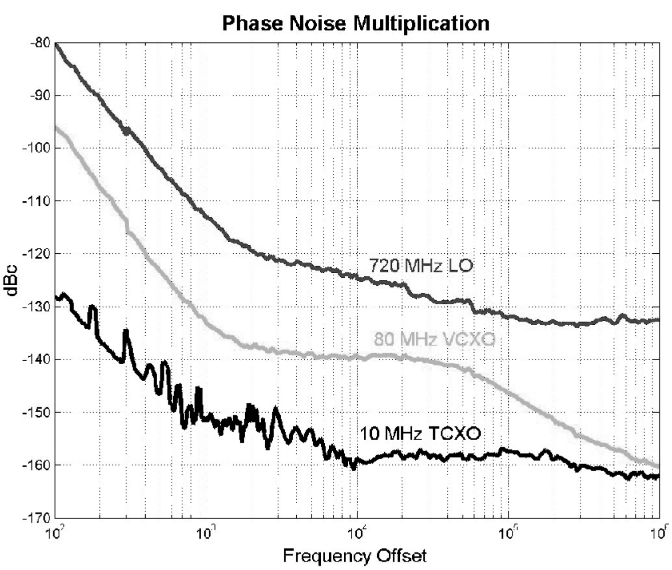


Figure 11 —Phase Noise of VCXO LO, showing increase with Frequency Multiplication

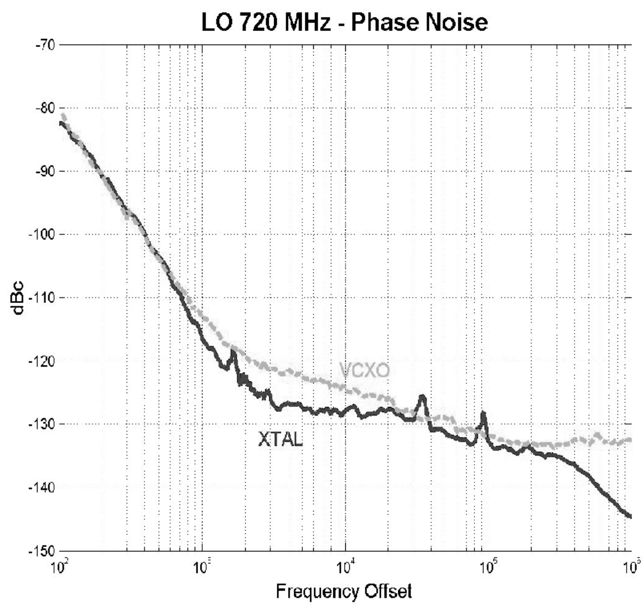


Figure 12 — Comparison of 720 MHz LO with Crystal and VCXO as source.

to insure that the results are comparable. Special thanks are due to the folks who provide and operate this specialized test equipment, particularly Greg Bonaguide, WA1VUG, of Rohde & Schwarz.

Spurious

On the versions that divide down to 1 MHz comparison frequency, like the 96 MHz version, I found small spurs 1 MHz on each side of the VCXO frequency. The spurs were about 65 dB down, but grew significantly larger after multiplication to 1152 MHz. I suspected inadequate bypassing — the 0.1 μF capacitors at each chip are not enough to be effective at low frequencies. A quick experiment showed that the problem was at the VCXO power pin rather than the tuning voltage.

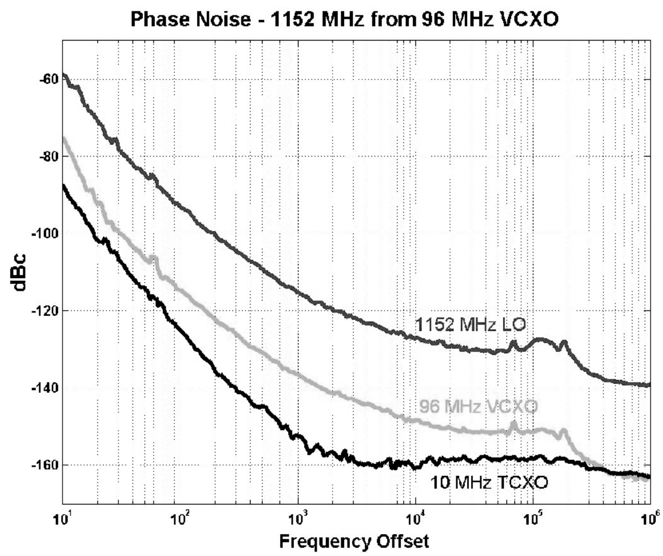


Figure 13 — Phase Noise of 1152 MHz VCXO from locked 96 MHz VCXO

I added a 1 μF chip capacitor at the VCXO and at each IC with a 1 MHz signal. The spurs were reduced to about 72 dB down, and reduced a similar amount after multiplication. This still isn't good enough, so I put the VCXO on a separate board with separate voltage regulator in a separate enclosure, shown in Figure 16. I added a Minicircuits directional coupler (door prize at some conference) to pick off some signal for the prescaler on the other board

Another choice might be to use a 12 MHz reference, eliminating the 1 MHz component entirely. This could be provided by locking a 12 MHz VCXO to GPS.[3]

The complete 1152 MHz LO is shown in Figure 17, packaged up as the start of a new 1296 MHz transverter. When I did the initial assembly, the power output at 1152 MHz was quite low. After some experimentation, I found that the harmonics of 96 MHz in the VCXO output were causing low output from the multiplier board. A low-pass filter eliminates the harmonics — I found one in the junkbox for 115 MHz, which works just fine, but it wouldn't be hard to make an adequate filter with a few coils and capacitors.

I did not notice any problem with VCXO harmonics in the 720 MHz LO chain, shown in Figure 18, but it used an early prototype board. I'll have to check this out further.

Update

I thought more about the 1 MHz spurs around the 96 MHz VCXO output. Since the problem seems to be with the power pin to the VCXO, I tried adding a separate 3.3 V regulator like U8 for the 74AC86 phase comparator U4, cutting the trace from U8, so that U8 only powers the VCXO. This moved the 1 MHz spurs below the noise on my spectrum analyzer.

I then packaged this unit up in a die-cast box, shown in Figure 19 and connected the output to the input of the low-pass filter in Figure 17. The output at 1152 MHz is fairly clean, with spurs at 12 MHz away that are 43 dB down. Smaller spurs are found 4, 6 and 8 MHz away, but more than 50 dB down. The only other visible signal is at 288 MHz, 40 dB down. While this LO version isn't as clean as the separate oscillator version in Figures 16 and 17, I'll bet that many transverters in use today are not any better.

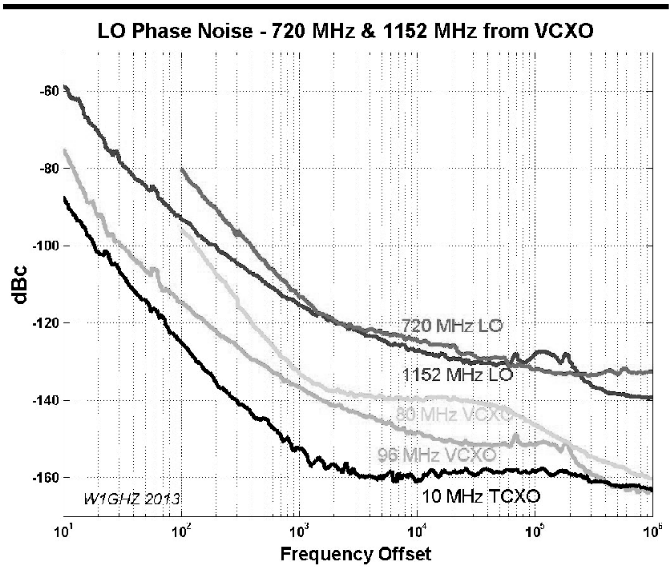


Figure 14 — Phase noise of both VCXO-based LO chains

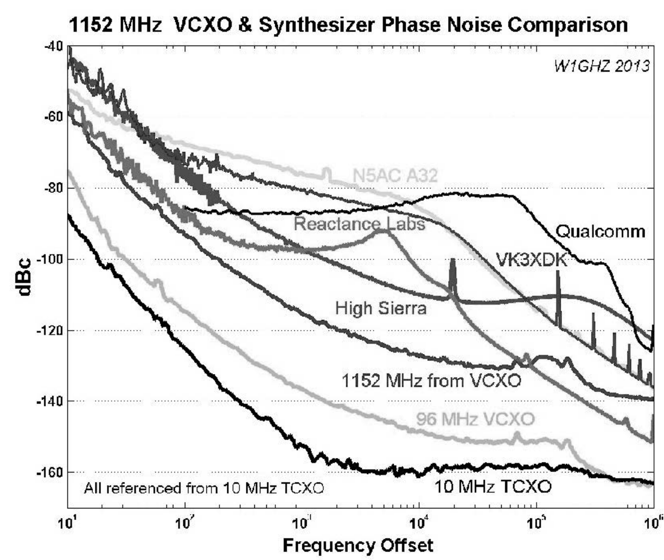


Figure 15 — Comparison of VCXO LO with several Synthesizers

Construction

Construction is fairly straightforward, with almost all of the components on the top side of the board, except for a few 1 μF chip capacitors on the bottom side to for better bypassing of the 1 MHz signal in some of the dividers. The VCXO and prescaler ICs are only available in surface-mount versions, and a few of the higher frequency components around them are also surface mount. The rest of the components are traditional thru-hole.

The Crystek VCXO shown in Figure 3 has four solder pads, matching the PC board, while the Abracon ABLJO-V VCXO used at the other frequencies have six pads. However, the middle ones are unused, so both types fit the PC board. I put a tiny dab of rosin paste flux on the VCXO pads before soldering. A VCXO is moderately expensive, but so are quality crystals.

Not all the ICs are needed for all options, and different prescalers are needed, as mentioned above. The final step is programming by soldering iron, adding the wires to select the appropriate divisors to bring the VCXO and reference frequencies to a common comparison frequency, with details in Appendix A. Appendix B is a parts list.

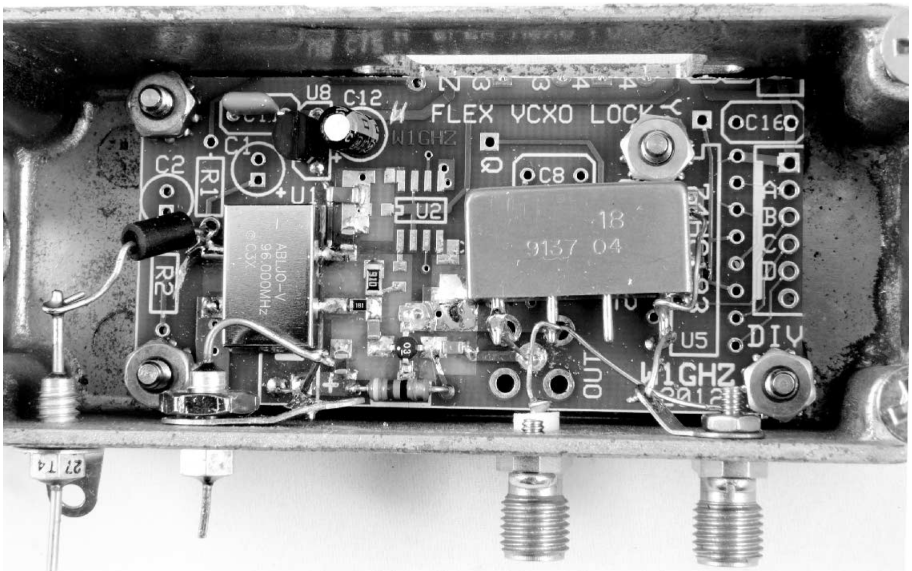


Figure 16 — VCXO on separate board with buffer amp and directional coupler

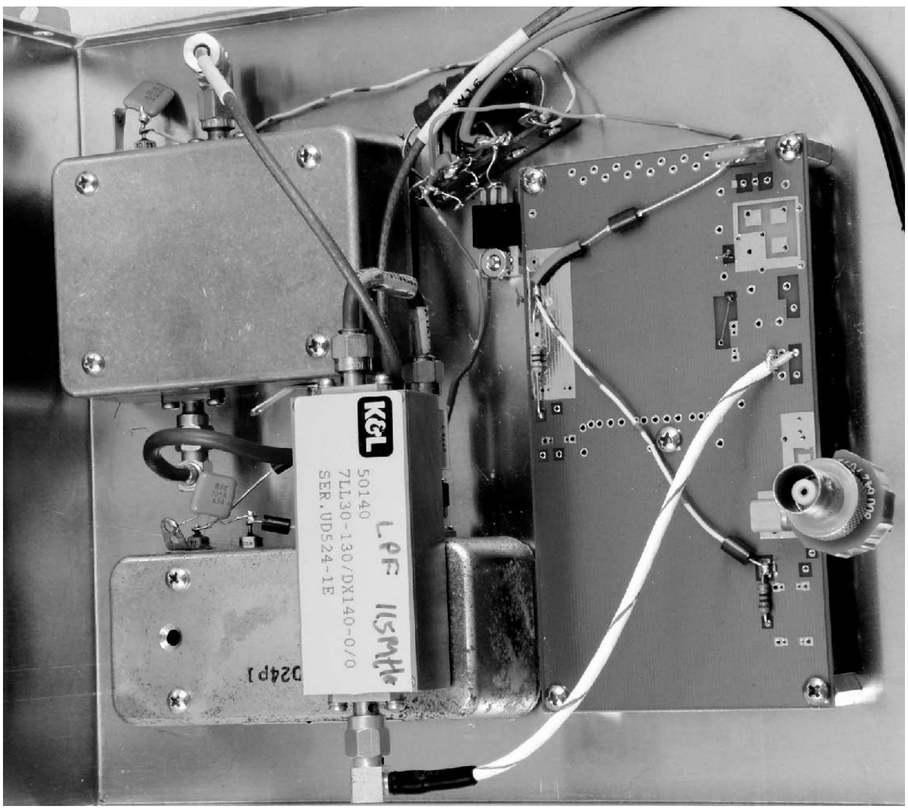


Figure 17 — Complete packaged 1152 MHz LO based on locked 96 MHz VCXO

The 1 μF chip capacitors, C18 thru C22, reduce the switching noise and resultant spurious signals in the versions with a 1 MHz comparison frequency, like 96 MHz. C18 can be seen in Figure 4, attached to the PC trace between U8 and the VCXO; the green solder mask must be scraped off the trace before it can be soldered. C19-22 are on the back side of the board, shown in Figure 5.

Other frequencies are certainly possible, but may require homebrewing your own VCXO. For instance, Down East Microwave has instructions for the crystal oscillator in some of their older transverters; this may be useful since crystals do tend drift slightly with aging. For odd frequencies that are not easily divided to 10 MHz, an odd reference frequency may be provided by a programmable Rubidium standard or by a reverse-DDS scheme.[4]

Output power at J2 is typically +3 to +5 dBm. It may be increased a bit by reducing the value of R3 — in my two piece version, R3 is 180 Ω and the output is about +10 dBm. This also increases the drive voltage to the prescaler, which should be 1000 mV max peak-to-peak (and at least 400 mV p-p); I used a directional coupler in the two-piece version to reduce the prescaler drive voltage. PC boards are available.

Summary

A local oscillator chain sourced from a locked VCXO can provide the frequency accuracy and stability of a good reference oscillator while maintaining the low phase noise of a crystal oscillator. Measured phase noise is 20 to 40 dB better than available synthesized local oscillators. Good packaged VCXOs are available for a few desirable frequencies at a cost comparable to a quality quartz crystal. Further experimentation is needed to determine whether the final system performance is significantly better than a good synthesizer.

Notes

[1]Paul Wade, W1GHZ, "Phase Noise and MDS," *Proceedings of Microwave Update 2009*, ARRL, 2009, pp. 193-196.

[2]Paul Wade, W1GHZ, "Multiband Microwave Transverters for the Rover — Simple and Cheap," *Proceedings of Microwave Update 2008*, ARRL, 2008, pp. 40-67. This article is available for download at: **www.w1ghz.org**.

[3]Paul Wade, W1GHZ, "A Flexible VCXO Locking Board," *Proceedings of Microwave Update 2012*, ARRL, 2012, pp. 101-113. This article is available for download at: **www.w1ghz.org**.

[4]See more details about the reverse DDS scheme at: **myweb.tiscali.co.uk/g4nns/RevDDS.html** and **www.microwaves.dsl.pipex.com/RDDS/RDDSINDX.htm**

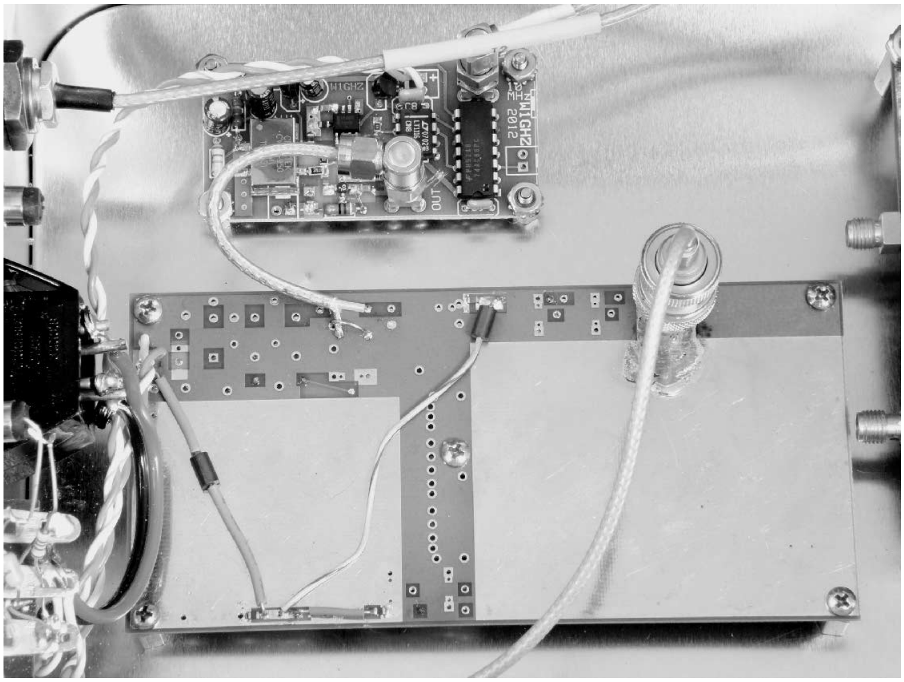


Figure 18 — Complete packaged 720 MHz LO based on locked 80 MHz VCXO

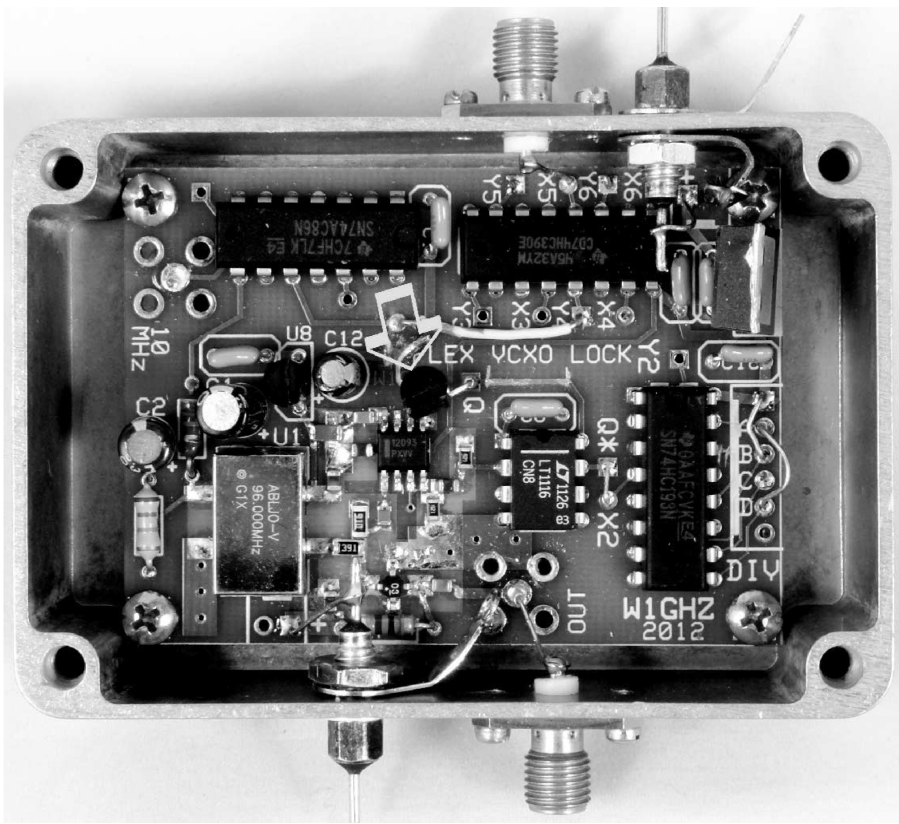


Figure 19 — 96 MHz locked VCXO in box. Arrow points to added voltage regulator.

Appendix A
Microwave VCXO Board Programming

Frequency (MHz)	*VCXO*	*DigiKey*	*Prescaler*	*Prescale*	*Divide (74HC193)*	*ABCD**	*÷ 2 (74HC390)*	*Signal*	*Comparison Frequency (MHz)*	*Ref ÷ 5 (74HC390)*	*Ref ÷ 2 (74HC390)*	*Reference*
80	Crystek	744-1214-ND	MC12093	÷ 8	na	na	na	Q->Z	10	na	na	10 MHz -> REF
80	Abracon	535-11429-ND	MC12093	÷ 8	na	na	na	Q->Z	10	na	na	10 MHz -> REF
96	Abracon	535-11431-ND	MC12093	÷ 8	÷ 6	LHHL	Q2->X4	Y2->Z	1	10 MHz->X5	Y5->X6	Y6->REF
100	Abracon	535-11433-ND	MC12080	÷ 10	na	na	na	Q->Z	10	na	na	10 MHz->REF
120	Abracon	535-11437-ND	MC12080	÷ 10	÷ 6	LHHL	Q2->X4	Y2->Z	1	10 MHz->X5	Y5->X6	Y6->REF
200	Abracon	535-11451-ND	MC12080	÷ 20	na	na	na	Q->Z	10	na	na	10 MHz->REF
108	Homebrew?		MC12026	÷ 9	÷ 6	LHHL	Q2->X4	Y2->Z	1	10 MHz->X5	Y5->X6	Y6->REF

NOTES
Y5->X6 means add a wire from point Y5 to point X6, etc.
*ABCD = LHHL: L is ground, H is +5 V
na means not applicable — chip may be left out
MC12093 ÷ 8: pins 3 and 6 grounded (unmodified board) DigiKey MC12080DGOS-ND
MC12080 ÷ 10: pins 3 and 6 high (jumpers to pins 2 and 7) DigiKey MC12093DGOS-ND
MC12080 ÷ 20: pin 3 grounded (unmodified) and pin 6 high (jumper to pin 7)
Crystek VCXO is CVHD-950 series
Abracon VCXO are ABLIO-V series
Comparator 74AC86 compares Z and REF Inputs

PC Boards marked 2013a are good for all frequencies
PC Boards marked 2012 are good for 80, 100, and 200 MHz. A wire is needed from Pin 1 of U4 (LT116) to +5 V

Appendix B
Parts List

Reference Designation	*Value*	*Description*	*Digikey*	*Mouser*
A1	MAR-3SM	MMIC	Minicircuits	
C1	1 μF	Electrolytic	P14054-ND	
C2	100 μF	Electrolytic	P5123-ND	
C3,7,8,9,11,16,17	0.1 μF			581-SR215C104K
C4,5,14	1000 pf	Chip		
C6,15	0.1 μF	Chip		
C10	0.33 μF		478-3194-ND	581-SR215E334M
C12	10 μF		P813-ND	
C13	100 pf	Chip		
C18-22	1 μF	Chip	478-5906-1-ND	
R1	82 k	¼ W	CF14JT82K0CT-ND	
R2	12 k	¼ W		
R3	390	Chip		
R4	91	Chip		
R5,6	820	Chip		
R7	91	¼ W		
U1	VCXO		See Appendix A	
U2	Prescaler		See Appendix A	
U3	LT1116		LT1116CN8*PBF-ND	
U4	74AC86		296-4345-5-ND	512-74AC86PC
U5	74HC193		296-8262-5-ND	
U6	74HC390		296-9199-5-ND	
U7	7805	78L05 adequate for 80 MHz version		
U8	78L3.3		497-7287-1-ND	

Thomas M. Alldread, VA7TA

7056 Railway Ave, Courtenay, BC V9J 1N4, Canada; **VA7TA@telus.net**

Android Wireless Project Control Part 1 — Android GUI Software Development

The author describes his entry into the world of Android device programming.

The popularity of tablets as personal computing devices has provided us with a powerful and convenient alternative platform for controlling Amateur Radio equipment. Ever since I started experimenting with microcontroller-based projects a few decades ago, I have wished for a means to obtain a powerful color graphical user interface (GUI) control platform that is convenient to use around the workbench and in the field. Although traditional desktop and laptop computers certainly have the power to do the job I find, in many situations, that they are awkward to use. Compared to tablets, conventional PCs are relatively bulky, are tethered by cables, often incur lengthy start-up delays and usually don't lend themselves very well to rough and tough environments.

The Refreshing Tablet Alternative

I find the tablet concept quite a refreshing alternative. The tablet is light, portable, and in my opinion it is about the right size for displaying test instrumentation and radio gear graphics. The touch screen control eliminates the need for a separate keyboard surface and is very easy to operate without the traditional need for point and click cursor control — just a quick finger touch is typically all that is needed to trigger a desired result. The tablets are battery-power-efficient devices with almost instant-on wake-up capability. Thus, tablets alleviate the traditional wake-up delay disruptions typically encountered with conventional PCs when, after a dormant command entry period, a new instrument setting needs to be changed or a new control command needs to be sent.

The tablet is designed for a fairly rough mobile environment, and thus it is a much better fit for use around the workbench and in the field. Tablets have built in wireless links for both project and Internet connectivity, plus they have data storage memory capability for saving acquired data. Last, but certainly not least, is that most modern day tablets deliver advanced, high resolution, bright color graphics along with stereo sound capabilities all controlled by a powerful operating system driven by a blindingly fast CPU. Figure 1 shows Jim, VE5FP, making a swept frequency SWR response for his 6 m antenna. The application I wrote, which Jim is looking at on my tablet, is controlling a NimbleSig III signal generator to perform the frequency sweep of the antenna.[1]

A final consideration is that tablets have become quite popular for Internet and other more general purpose activities, so many radio amateurs may already own a suitable device that could also be used as an Amateur Radio project controller.

Remote Control Wireless Project Portability

Tablets with Bluetooth (BT) wireless connectivity capability don't even need to be

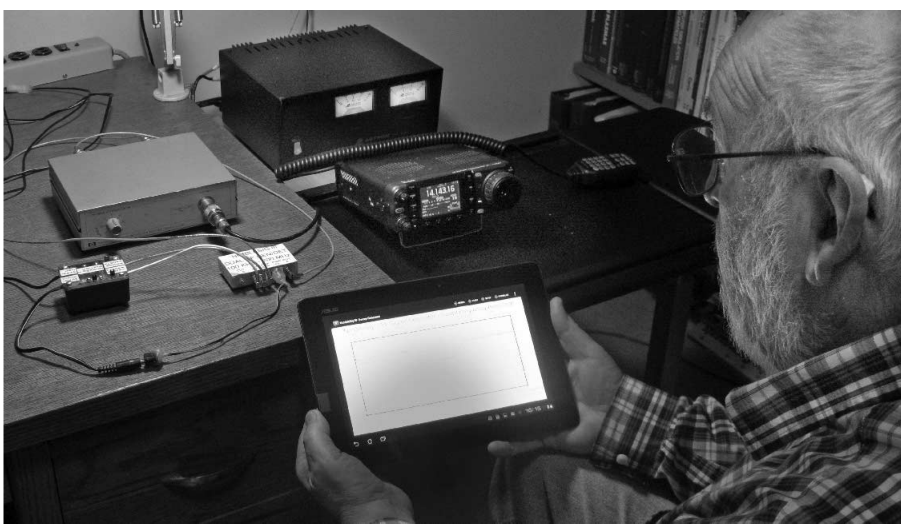


Figure 1 — Jim Koehler, VE5FP, performing a 6 m antenna swept frequency SWR response via wireless tablet Control.

[1]Notes appear on page 37.

cable tethered to the target equipment. At the time of this writing the additional transceiver module that is needed within a radio or instrument to provide BT wireless serial data link capability costs less than a computer serial cable with the required USB/RS232 adapter. Thus, the use of a BT wireless link is a cost effective alternative that offers wireless convenience and portability.

Data Storage and Wireless Internet Connectivity

A nice benefit that comes with the use of a tablet for an instrument display is that the operating system provides support for saving acquired data locally and/or sharing data via WiFi access to the Internet. For example, I sent the Figure 2 screen capture for inclusion in this article from my tablet to my main desktop PC via an email attachment. Figure 2 shows the feed line match coefficient frequency response measurement for Jim Koehler's (VE5FP) 6 m antenna. See the "Measuring Antenna Response" sidebar. The results of this measurement can be easily filed for reference for future routine follow-up testing of the antenna system.

Keep it Simple Factor

Keep in mind that for projects where you only need a text readout of a few lines and a few push button switches for control, a low cost, alpha-numeric LCD display should still be given consideration as possibly the best option. These traditional displays are inexpensive, become an integral part of the equipment and are usually simple to get going because they directly accept ASCII text character data. If minimal cost is a priority and additional firmware driver complexity is acceptable then very inexpensive graphics LCD displays are currently available on eBay. For example the tiny 84 × 48 pixel Nokia 5011 back-lit LCD cell phone graphics display (Nokia 5011 open source alpha-numeric drivers are available for some popular microcontrollers) can, at the time of this writing, be purchased for roughly $3.00, shipping included.

Good Fit Tablet Applications

The tablet option described here offers a solution for projects requiring a complex user interface with multiple menus, a need for graphics and/or the need to store or share acquired information. In these cases some form of PC or at least a touch screen LCD display would be needed anyway to support the complex project. Thus, considerable software development time would be involved to provide the user interface in any case. Keep in mind that the tablet can also easily be used for the wireless control of projects that only require text communications with the operator. For such cases there is no need to do any GUI software development for the tablet. There are multiple serial ASCII terminal apps freely available for the Android tablet that can be used to control a target project via a wireless BT link.[2] These terminal programs can be used quite conveniently to wirelessly connect to and text communicate with a microcontroller based project while offering the additional benefits of a relatively large, more brilliant screen, data storage, screen capture and outside world connectivity.

The Project Interface Standard

A practical compatibility consideration is the use of standard ASCII text characters for the project serial interface command sequences. I usually stick to the general command line format used by operating systems, where the target project sends out a prompt when it is ready to receive a new command. The commands consist of standard ASCII character sequences terminated by a carriage return (CR). The use of a CR termination provides a user friendly acknowledgement key stroke for triggering each command. This approach for the serial interface protocol paves the road for testing with common terminal applications and modern day logic analyzers, which typically have built in ASCII character decode capability.

Figure 3 illustrates how a tablet can be used for wireless project control without any need for Android software development. This screen was captured from my tablet, which was connected via a wireless Bluetooth link, to a signal generator module using a freeware app called *BlueTerm*. As shown, projects

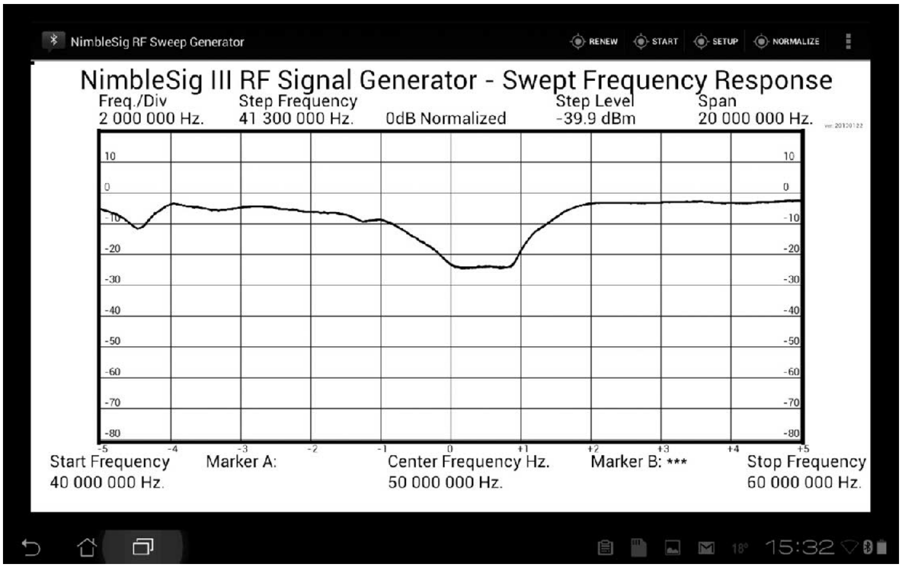


Figure 2 — Tablet screenshot of VE5FP 6 m beam feed line match coefficient swept frequency response.

Measuring Antenna Response

For the measurement of Jim (VE5FP) Koehler's 6 m antenna, we used his classy, multi-gigahertz-bandwidth HP return loss bridge connected to a NimbleSig III generator module. You should understand, however, that a relatively inexpensive Minicircuits ZFDC series directional coupler or even a home brew bridge could have been used to produce similar results for this rather low-demand test.

Figure 2 in the article shows the 50 to 52 MHz feed line match coefficient frequency response, as measured in Jim's ham shack. Note that over the 50 to 52 MHz range, the returned signal was about 24 dB below the forward signal, which is equivalent to an SWR of about 1.2:1. Also of interest is the low out of band response. At 60 MHz, where the antenna is not resonant, the returned signal is only about 2 dB below the forward signal. Since at 60 MHz the antenna SWR is extremely high, almost all the power is reflected. This provides a convenient method for keeping an eye on the feed line insertion loss. Since in this case the round trip loss is about 2 dB, the insertion loss must be 1 dB or less. Jim uses hardline for his 6 m antenna feed line, so this very nice low insertion loss estimation seems appropriate.

that provide a command line style text based interface can be readily controlled wirelessly with off-the-shelf applications.

In comparison to text, the convenience of a slick graphical interface with touch screen control makes for a much more user friendly controller. In the case of the example signal generator project described here, there was clear justification for the development of a custom graphical user interface (GUI).

The Android Operating System Choice

Google's rather open Android operating system appeals to me. Android runs on a securely designed, multi-tasking, multi-user *Linux* platform. Android applications are written in the well established *JAVA* high level language. Of particular importance to most hobbyists is that the Android development system tools, which use the Eclipse IDE shown in Figure 4, are free. Additionally there is a good tutor website with many examples provided by Google. There are also many open source code snippets available for download from other sites on the Internet that can be used to provide a quick start. In general there is much freely available support for Android.

Help for Some Android Hurdles

During the process of developing this app I encountered a number of road blocks that stopped my progress until I gained a better understanding of the Android system. I found that I needed to spend a lot of time searching around the Internet for answers. I hope the various links provided here will ease the effort needed to find relevant resources of information on the Internet.

Android App Development — Can I Manage to Do It?

Before diving into the deep end of a new software development pool, radio amateurs may wish to consider whether or not the desired result is reachable within the time available to dedicate to the project at hand. Suffice to say that I am not an advanced programmer; however thanks to the free tools, example code and support information on the Internet, I was still able to manage the development of an app that I consider quite rewarding. The graphics and Bluetooth communications modules used for this app are based on open source code provided in tutorials written by others to whom I wish to express my gratitude.[3] I think the best way for someone to test the Android waters is to work through some of the examples provided on the Android developers website.[4]

Define the Target Project

Prior to diving into Android controller development the target project needs to be developed, built and tested. As mention above it is important to design a project that uses a standard ASCII text interface which lends itself to testing by freely available terminal applications. As the Bluetooth transceivers used here provide the common UART asynchronous, 3 V, TXD/RXD logic level data interface the target project needs to be designed with the same type of interface. A typical desktop PC running your favorite serial communications terminal program connected via an inexpensive USB to UART logic level converter module works well for the initial testing. This simplifies the early phase of project development by minimizing the complexity of the communications link during the project debugging stage. The Bluetooth link can be introduced later after the project itself is known to be working properly.

Figure 3 shows the command line style ASCII text character interface used by my signal generator project. The example shows the setting of the VFOA frequency to 100 MHz with the character sequence "FAM100" followed by the *Enter* key which terminates the command with the end-of-line carriage return (CR, 0x0D) character. The "LA100" command that followed triggered the RF detector level measurement, which responded with –69.2 dBm. By boot-up default the firmware in this project echoes back the characters as they are sent and then, after each command entry, it responds with a '>' prompt character at the beginning of a new line. The character echo and the prompt provide necessary acknowledgement feedback for human-to-machine control but can be cumbersome to deal with for machine-to-machine communications. Consequently there are commands provided in the signal generator firmware for disabling and re-enabling the echo and prompt features. It is wise to design any project that is intended for both human and machine control to have this adaptability.

The Android Development System

As usual the first step towards getting started with developing an app for a new platform is to get the associated software development tools up and running on the software development station PC. There are Android Software Development Kit (SDK) versions available for *Linux*, *Mac OS* and *Windows* for both 32 and 64 bit platforms. These download packages, called "ADT-Bundles," are provided on the Android Developer web site along with installation instructions.[5] Note that at the time of this writing Google is previewing a new

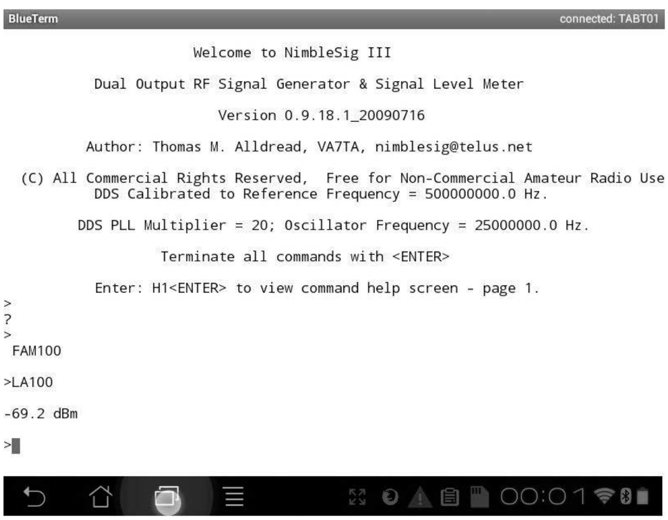


Figure 3 — Tablet screenshot (cropped) of typical ASCII human-machine interface via an Android Bluetooth freeware terminal program application called *BlueTerm*.

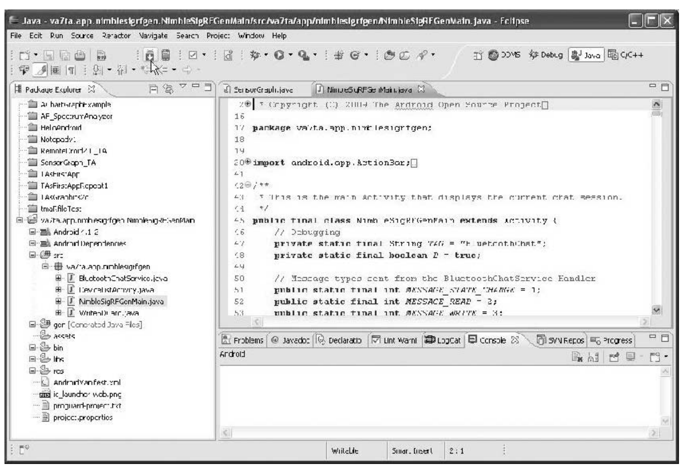


Figure 4 — Eclipse integrated development environment (IDE) — Android software development tools main screen.

generation of Android tools called "Android Studio." These tools, which are still under development, look very promising and I have the first impression that as they become more mature they will be the better choice.

The Android SDK uses a popular Integrated Development Environment (IDE) program called Eclipse, shown in Figure 4. The Eclipse IDE integrates access to all the Android SDK tools into one GUI. Eclipse is an IDE intended to be adaptable to almost any software language for almost any platform. This multipurpose IDE is adapted to different targets through the use of associated plug-ins which, in this case, are within the Android SDK. Google has provided single download bundles for each supported platform, named ADT-Bundle-xxxxx where xxxxx is the name of the intended PC platform. These bundles provide all the installation files needed for installing both Eclipse and the Android tools. The bundles also include installation files for an emulator that can be used for a code test platform. It can emulate, with some limitations, multiple variations of Android devices. This emulator provides a means to run and test new code eliminating the need, in many cases, for providing an external Android tablet (or other Android based device) for testing.

As my development station PC is set up to boot either *Windows XP* or *Linux* Mint 32 bit platforms I chose to install both the *Windows* and *Linux* SDT packages to compare the SDK performance running on the two platform alternatives. Since I found that the Android device emulator running under *Linux* loaded apps faster, I chose *Linux* for my development platform. I am quite confident, however, that either of the other platforms could be used effectively. As I ended up using my tablet for testing my new code, the emulator program load time consideration turned out to be irrelevant for development of my particular application.

After initially getting set up to use the emulator, I discovered (the hard way) that the emulator does not, at least at the time of this writing, support Bluetooth wireless linking. Thus, for any application that uses BT, you must have an external BT equipped Android device that can be connected to a USB port via a suitable cable.

The USB interface built into my Asus Transformer tablet has proven to work well with the Android SDK software. A further benefit for using the tablet USB alternative for the code test platform is that the application reload time can be quite a bit faster compared to using the emulator. I concluded that for my development station the tablet will probably be my preferred test platform choice.

An advantage that the emulator offers over a single external tablet test platform is that it can provide different screen size views. This feature facilitates perspective viewing of the various GUI screens as would be seen on devices with different screen resolutions.

As the preliminary version of my NimbleSig III app described here is currently just intended for a 10 inch screen my tablet has provided all the test platform capabilities that I have needed to date for this endeavor.

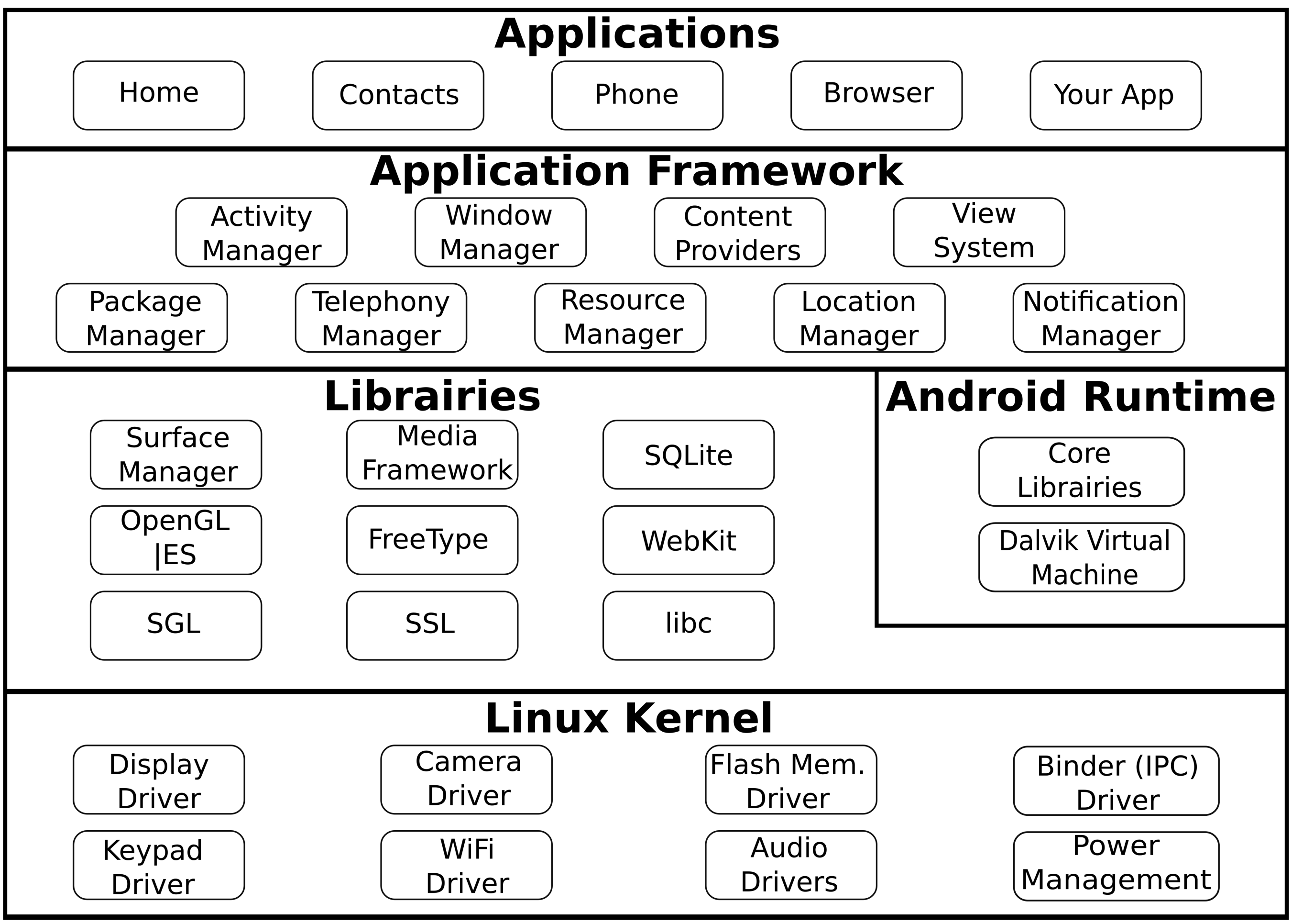


Figure 5 — Android system architecture (Courtesy android-app-market.com).

Android Operating System — Tutorials

Google has provided an excellent Android training section on their android developers web site in the "Getting Started" section.[4, 6] Initially I didn't know about the Google Android Internet tutor, so I started out studying a published book on Android programming. Although the book was surely a helpful start, with hindsight I think I would have progressed faster had I started with Google's free online tutor. The tutor provides instructions for both the Eclipse IDE as described here or alternative command line tools, which I have never used.

Android Operating System Adaptability

A feature I found particularly different with Android code is the additional complexity needed to permit software to adapt to widely varying, sometimes very resource limited, target devices. My past experience was either producing code for PCs with an abundance of resources, or alternately for a microcontroller project with clearly defined hardware. The Android software structure provides the necessary additional capability with associated complexity that permits app compatibility with the limited and varying resources of portable Android devices. These devices have varying features, varying amounts of memory and varying screen sizes, which in turn can have various resolutions with differing pixel densities.

Android Operating System Versions

The Android operating system has spiraled since its inception a half dozen years ago. New releases have typically been emerging biannually. A large number of active mobile devices employ various earlier versions of the Android platform. To provide support for these earlier devices, the SDK includes separate Eclipse plug-in packages for each earlier version of the operating system. During the installation of the SDK, the programmer needs to decide on the Android versions for which to install support. I chose version 4.1 used by my Asus Transformer tablet, which is quite recent and thus supports the majority of Android's latest features.

Android Application Components

Many descriptions of the classic block diagram of the Android operating system, shown in Figure 5, can be found on the Internet. For the purpose of this article, suffice to say that Android programs are based on four component types: Activities, Services, Content Providers and Broadcast Receivers, which are described in detail on the Android Developer web site.[7]

A significant difference between an Android app compared to conventional embedded firmware is that each screen displayed by an app is assigned its own independent code block (called an Activity). Whenever there is a screen change, a new independent activity is initialized. The previous activity code block, associated with the previous screen, is either put on hold within a background stack or becomes overwritten should there be a need for memory space by higher priority foreground activities. If there is sufficient memory then there may be several previous activity code blocks stacked on hold in the background that can be quickly brought back to the foreground if one of the previous screens is recalled. This is a slick system that makes it possible for large complex programs to run on a device with limited resources.

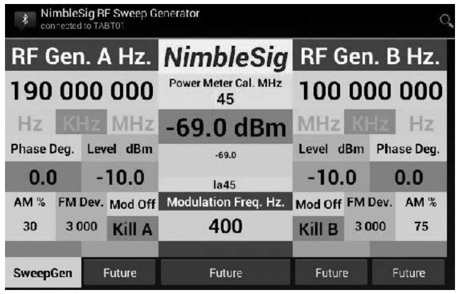


Figure 6 — NimbleSig generator control GUI.

This concept differs from what most modern day microcontroller programmers are accustomed to. In the case of the microcontroller program, all sub routines typically reside in program memory all the time and share a common memory for exchanging data.

Since multiple activities are not active at the same time, an added complexity of the Android scheme is the need for the implementation of special techniques to permit intercommunication between activities. Android provides a special support module (class) named Handler, which in this case delivers data streams to and from the Bluetooth service. Another support class named Intents has multiple uses, one of which is to provide the application with access to sharable general purpose activities, which belong to other application packages. This can, in many cases, eliminate much duplication of resident code on the tablet for common tasks such as messaging, sound level control, editing, and so on.

Android Manifest File

All the components and associated resources used by an application must be declared in a file named "AndroidManifest.xml." This manifest file must reside in the root of the application source code directory tree. Details for the manifest file can be found on the Android developer web site.[7]

Android Styles and Themes

The Android operating system uses styles and themes to define the appearance of the various screens. The use of styles and themes permits automatic control of screen appearance for matching the graphics standards used by different mobile devices. Themes can be used to define the appearance of an entire app or an activity. Styles are used to define the appearance of specific graphic items (called Views in Android) such as icons and widgets. There are a huge number of Android platform standard styles and themes to choose from. These can be customized with selective inheritance techniques. As the range of possibilities seems immense, you could spend a great deal of time studying this aspect of Android programming alone. That is beyond the scope of this article. Detailed information can be found on the Android web site.[9] Fortunately, the default theme selection done by the SDK wizard is all that is needed for most controller type GUIs.

My first puzzling exposure to the significance of themes occurred when I first tried integrating the graphics module used for the swept frequency response feature into my signal generator controller app. I found that when I ran the graphics code independently, it provided the desired full screen display, but when I started the same graphics code from my app it popped up in a reduced size window. I eventually learned that this was because the theme that was automatically assigned by default to my new sweep generator activity was limiting the screen size. A modification of the theme definition, which appears in the manifest file, resolved the problem. This issue provided me with an appreciation of the power of styles and themes in Android programming.

Designing the GUI Screen

The Android development system, similar to many modern day high level development tool packages (such as Netscape), provides a drag and drop layout editor that you can use to drag and drop widgets such as buttons and text windows onto the GUI canvas. The GUI screen that was captured for Figure 6 was built using this tool.[10]

The Android widget placement tool permits you to choose how the locations of the various widgets on the screen are defined in code. I chose the powerful relative layout mode, which defines the location of the widgets by referencing previously placed widgets. A downside of the relative layout concept is that if you move a reference widget the other associated widget positions are shifted as well. If you change the name of a reference widget then dependent widgets loose their position reference, which can result in a scrambled screen that can be time consuming to reconstruct. To prevent this from happening, you need to update all other occurrences of the changed name in the layout XML file. This is best done by using the find and replace search tool. It is important that all name occurrences are updated prior to going back to the graphics view screen, because that action would otherwise trigger the possible scrambling of the widget locations.

As you place widgets on the canvas, the layout editor generates the Extensible Markup Language (XML) source code that defines the screen. XML, which is essentially a format standard for plain text data storage, is intended to be easily readable by both humans and machines. Thus it is a quick study.[11]

Parts 2 and 3 to Follow

The second part of this article will describe the example NimbleSig III signal generator Bluetooth wireless app, which appears here in some of the Figures. This example application provides a GUI based controller for all the basic functions needed to fully control a NimbleSig III RF Signal Generator (NS3) module. (See Note 1.) In addition, this package creates a wirelessly controlled swept frequency response display system based on the NS3 module.

The third part of this article will focus on the Bluetooth link used for the wireless connection. A description of the BT transceiver and the associated configuration setup will be provided.

Tom Alldread, VA7TA, became interested in electronics while still in grade school, repaired radio and television sets in his teens, obtained his Amateur Radio ticket at the age of 19, and went on to obtain commercial radio operator certification a year later. He subsequently graduated from the Capitol Radio Engineering Institute Engineering Technology program. His career was in the telecommunications industry where he worked in microwave and VHF radio equipment maintenance, training, engineering standards and design, and long distance network operations management.

Tom is now retired and lives on Vancouver Island with his wife Sylvia, VA7SA. Tom has a history in repeater building and emergency communications, and he enjoys operating CW, building equipment and is currently the net manager for the 20 m Trans Canada Net. He is interested in microcontroller development projects associated with RF technology. He had the good fortune of winning second place in the 2006 Design Stellaris and first place in the 2011 Renesas RX international microcontroller design contests. Other hobbies include computing, hobby farming and bicycling.

Notes

[1]Thomas Alldread, VA7TA, "NimbleSig III — Parts 1, 2, and 3," QEX, Jan/Feb, Mar/Apr, May/Jun 2009. The article and more details are available on the author's website at: **www3.telus.net/ta/NimbleSig%20III/**.

[2]There are a number of Bluetooth terminal programs available at the Google store: **https://play.google.com/store/search?q=BlueTerm&c=apps**

[3]Tutorials and sample graphics and Bluetooth communications modules used in this project were found on several websites:
https://thenewcircle.com/s/post/1036/android_2d_graphics_example
developer.android.com/guide/topics/connectivity/bluetooth.html
developer.android.com/tools/samples/index.html

[4]The examples provided on the Android developer's website are very helpful for learning how to program in this environment: **developer.android.com/training/index.html**

[5]You can find Android developer tools and the Software Development Kit at:
developer.android.com/tools/index.html
developer.android.com/sdk/index.html

[6]The Android developer's website provides a wealth of useful information: **developer.android.com**

[7]For more information about the four Android component types, go to:
developer.android.com/guide/components/index.html

[8]For details about the Manifest file go to:
http://developer.android.com/guide/topics/manifest/manifest-intro.html

[9]There are many Android platform styles and themes that you can use to customize the various Views in your program:
developer.android.com/guide/topics/ui/themes.html#PlatformStyles

[10]The Android development system provides a drag and drop layout editor that is very useful for creating GUI screens such as the one shown in Figure 6:
http://developer.android.com/training/basics/firstapp/building-ui.html

[11]For more information about the XML source code generated by the layout editor, go to:
www.w3schools.com/xml/xml_whatis.asp

Upcoming Conferences

2014 Annual Conference, Society of Amateur Radio Astronomers

June 29 – July 4, 2014
National Radio Astronomy Observatory
Green Bank, WV

The Society of Amateur Radio Astronomers (SARA) will hold its 2014 Annual Meeting and Technical Conference Sunday June 29 through Friday July 4, 2014, at the National Radio Astronomy Observatory (NRAO), Green Bank WV. This is a joint conference with the Radio JOVE Project. Papers will be presented on radio astronomy hardware, software, education, research strategies, and philosophy. Dr Joe Taylor, K1JT, will be our keynote speaker at the Green Bank SARA Conference

For more information go to **www.radio-astronomy.org**.

Central States VHF Society

July 25 – 27, 2014
Austin Marriott South
Austin, Texas
4415 So. IH35
Austin, TX 78744
Hotel Reservation Phone:
888-253-1628

The Central States VHF Society, Inc. is soliciting papers, presentations, and poster displays for the 48th Annual CSVHFS Conference on July 25–27, 2014. Papers, presentations, and posters on all aspects of weak-signal VHF and above Amateur Radio are requested. You do not need to attend the conference, nor present your paper, to have it published in the Proceedings. Posters will be displayed during the two days of the Conference.

The papers will be published in the *Conference Proceedings*, which will be available at the conference. You do not have to attend the conference nor present the paper to have it published in the *Proceedings*. Posters describing your project can be displayed during the 2-day conference.

Presentations and Posters at the conference may be technical or non-technical but will cover the full breadth of amateur weak signal VHF/UHF activities. The presentations generally vary from 15 to 45 minutes, covering the highlights with details in the *Proceedings* paper. Topics of Interest include:

- VHF/UHF Antennas, including modeling/design, arrays and control
- Construction of Equipment – such as transmitters, receivers and transverters
- RF power amplifiers – including single and multi-band, vacuum tube and solid state
- Preamplifiers (low noise)
- Regulatory topics
- Software defined radio (SDR)
- Test equipment – including homebrew, using and making measurements
- Operating — including Contesting, Roving and DXpeditions
- Propagation – including ducting, sporadic E, tropospheric and meteor scatter
- Digital Modes – WSJT, JT65 and others
- EME (Moon Bounce).
- Digital Signal Processing (DSP)

Non weak signal topics such as FM, repeaters and packet radio are generally not considered, although there are exceptions. If you have any questions about the suitability of a topic, contact K5TRA.

If you would like to contribute a paper, presentation, or poster, please contact Tom Apel, K5TRA, *as soon as possible* with the title and a short description. Author Guidelines and other details are available at the Society website: **www.csvhfs.org/2014conference/2014callforpapers.html**

Tom Apel, K5TRA, 7221 Covered Bridge Dr, Austin, TX 78736-3344; **csvhfs2014@gmail.com**

Submissions Deadlines:

Proceedings – April 23, 2014
Presentations – June 27, 2014
Posters – June 27, 2014

Banquet Speaker

The Saturday evening Banquet Speaker will be Jimmy Treybig, W6JKV.

The 33rd Annual ARRL and TAPR Digital Communications Conference

Austin, Texas
September 5-7, 2014
Austin Marriott South
4415 S. IH-35
Austin, Texas 78704
Hotel Reservation Phone:
512-441-7900

Now is the time to start making plans to attend the premier technical conference of the year, the 33rd Annual ARRL and TAPR Digital Communications Conference. This year's DCC will be held September 5 – 7, 2014 in Austin Texas, at the Austin Marriott South. This is the same hotel as the Central States VHF Society Conference. Regular attendees will note that the conference is a couple of weeks earlier than normal this year. It is the weekend after Labor Day.

The ARRL and TAPR Digital Communications Conference is an international forum for radio amateurs to meet, publish their work, and present new ideas and techniques. Presenters and attendees will have the opportunity to exchange ideas and learn about recent hardware and software advances, theories, experimental results, and practical applications.

Topics include, but are not limited to: Software defined radio (SDR), digital voice (D-Star, P25, WinDRM, FDMDV, G4GUO), digital satellite communications, Global Position System (GPS), precision timing, Automatic Packet Reporting System® (APRS), short messaging (a mode of APRS), Digital Signal Processing (DSP), HF digital modes, Internet interoperability with Amateur Radio networks, spread spectrum, IEEE 802.11 and other Part 15 license-exempt systems adaptable for Amateur Radio, using TCP/IP networking over Amateur Radio, mesh and peer to peer wireless networking, emergency and Homeland Defense backup digital communications, using Linux in Amateur Radio, updates on AX.25 and other wireless networking protocols and any topics that advance the Amateur Radio art.

This is a three-Day Conference (Friday, Saturday, and Sunday). Technical sessions will be presented all day Friday and Saturday. In addition there will be introductory sessions on various topics on Saturday.

Join others at the conference for a Friday evening social get together. A Saturday evening banquet features an invited speaker and concludes with award presentations and prize drawings.

The ever-popular Sunday Seminar has not be finalized yet, but is sure to be an excellent program. This is an in-depth four-hour presentation, where attendees learn from the experts. Check the TAPR website for more information: **www.tapr.org**.

Call for Papers

Technical papers are solicited for presentation and publication in the *Digital Communications Conference Proceedings*. Annual conference proceedings are published by the ARRL. Presentation at the conference is not required for publication. Submission of papers are due by 31 July 2014 and should be submitted to: Maty Weinberg, ARRL, 225 Main Street, Newington, CT 06111, or via the Internet to **maty@arrl.org**. There are full details and specifications about how to format and submit your paper for publication on the TAPR website.

Even if you are not presenting a paper at the conference, plan to bring a project or two to display and talk about in the popular Demonstration Room, or "Play Room" as it is commonly known.

AMSAT Symposium

According to the AMSAT website (**ww2.amsat.org/?page_id=1555**) the 2014 AMSAT Symposium and General Membership Meeting will be held in the Baltimore/Washington DC area the weekend of October 10-12 2014. Check the AMSAT website for updated information. Mark those dates on your calendar now, and plan to attend.

Whether you are a seasoned satellite operator or think you might like to give it a try, the Symposium will have plenty of presentations of interest. Catch up on the latest news about AMSAT's various satellite projects as well as presentations about equipment and operating practices.

Microwave Update

This year's Microwave Update Conference will be held in Rochester, NY. There are no further details on the MUD website (**www.microwaveupdate.org**) at the time of this writing, so check the website for details as they are filled in. The Northern Lights Radio Society website (**www.nlrs.org/home/contest-activities**) has the conference listed for October 24-26, 2014, so you can mark that time off on your calendar.

MUD is the top technical conference for anyone interested in our microwave frequencies. There are sure to be plenty of technical presentations about equipment construction, operating techniques and other topics of interest to beginners as well as long-time operators.