The NCDXF/IARU International Beacon Network—*Part 2*

In Part 1, the beacon network was described; this part explains the underlying technology including the use of a GPS receiver for accurate timing.

By John G. Troster, W6ISQ
Coordinator
IARU International Beacon Project
82 Belbrook Way
Atherton, CA 94027
Photos by John G. Troster, W6ISQ

and Robert S. Fabry, N6EK
Assistant Coordinator
IARU International Beacon Project
1175 Colusa Ave
Berkeley, CA 94707

N6EK with an entire multiband beacon unit.

Part 1, in last month's *QST*, presented a short history of the 15-year-old 14.1-MHz worldwide beacon network that we call Phase II. The Phase II network introduced two important innovations in beacons. First, by transmitting at four power levels it allowed casual listeners to learn much more about the robustness of the propagation path than had previously been possible. It's great fun to know you just heard a 0.1-W signal from halfway around the world! Second, by sharing a single frequency among many beacons, it not only cut down on the spectrum needed by the beacons, but it also simplified listening—the listener does not have to tune to a different frequency for each beacon. In the Phase II network, each beacon has a 1-minute time slot once every 10 minutes in which it is to transmit. The downside of this arrangement is that it takes 10 minutes to listen for 10 beacons; many people find this too long.

The soon-to-be-operational Phase III network will improve on Phase II by allowing more beacons. It does this by shortening the transmissions in order to make listening for the beacons faster. It also will allow the beacons to operate on five bands. The present and possible future locations of beacons in the Phase III network were shown in the map in Part 1 of this article.

Time Standard

In order for frequency-sharing to work, each beacon must know precisely when its time to transmit has come. Each Phase II beacon uses a 6-MHz temperature-compensated crystal oscillator (TCXO) for its time base. Each beacon transmits when it is first put on line, and every 10 minutes thereafter. Each beacon is started manually, and must be started within a fraction of a second of the correct time for its transmission.

In practice, the Phase II beacon's internal clock drifts 1 or 2 seconds per month. The transmission sequence allows 3 seconds of "guard time" between one beacon's transmission and the next; after a drift of 3 seconds, a beacon's transmission may overlap that of its neighbor. As a practical matter, it is necessary to reset each beacon's internal clock every four to six weeks, and this resynchronization is the most demanding operational aspect of the Phase II beacon network. We owe a great deal of appreciation to the individual amateurs and amateur organizations around the world who have undertaken the responsibility of faithfully operating the beacons for so many years.

The other operational problem with the Phase II network is that no distinction is made between an intentional synchronizing power-up and a randomly timed power-up after a power failure. If you have heard a beacon transmitting at a wildly incorrect time, a randomly timed power-up that has not been yet discovered by the beacon operator is the cause.

WWV Timing Solution

A first attempt to solve the synchronization problem used the 0.8-second 1500-Hz tone broadcast by WWV and WWVH at the start of each hour to keep the internal clock synchronized. The transceiver used for the beacon transmissions was switched to WWV, its audio output was fed into a 1500-Hz detector and the detector output was made available to the microprocessor. Spurious responses were minimized by measuring the precise duration of the detected tone and by knowing approximately when the start of the hour should occur. In tests in California this scheme worked well, but there was a concern that the scheme could not be depended on to work well in every corner of the world and in every phase of the sunspot cycle.

GPS Timing Solution

The final synchronization scheme uses the Global Positioning System (GPS) satellites to provide a timing standard. Rather than do this from scratch, it has proven cost-effective to use a commercial unit, the Accutime GPS receiver made by Trimble Navigation.[1] This unit combines the antenna and receiver in a small plastic housing, requires 2 W at 12 V dc and provides two outputs: a pulse at the beginning of every second that is specified to be accurate within 1 microsecond

[1]Notes appear at end.

Copyright © 1998 ARRL

Beacon Hunting from Northwest Connecticut

"Sure—I can monitor 14.100 MHz and log some beacons over the weekend for W6ISQ's article." It sounded like a piece of cake: log some beacons, write up a little blurb, and get back to my planned weekend fun of replacing the water line from the house to the road. But actually *hearing* the beacons proved to be more difficult than digging the 55-foot-long, 4-foot-deep trench for the water line.

It's not that I was trying to monitor the NCDXF beacons with a direct-conversion receiver hooked up to an unwound 10-foot reel of hook-up wire. Though perhaps modest to some, the equipment consisted of a Kenwood TS-440S/AT loaded with primo IRCI crystal IF filters, and the choice of a rotatable Mosley TA-53-M at 73 feet or a Cushcraft A3, fixed south, at 36 feet. The Yagis are switch-selectable.

What became immediately apparent was that packet stations—*strong* packet stations—on both sides of the supposedly *guarded* beacon frequency, made for tough going. I could hear 4U1UN, W6WX, CT3B and LU4AA regularly, as propagation supported the paths, of course. But copying the other five stations in the existing beacon network proved fruitless.

Amateurs with bare-bones stations trying to log beacons on 14.1 MHz will no doubt experience varying degrees of frustration. I believe that users of stock HF transceivers with only a 2.1-kHz IF filter are going to get eaten alive by packet stations on or near the beacon frequency. This isn't to say that basic HF stations won't be able to hear any of the beacons. But in order to make the most out of the NCDXF beacon system, here is a list of things that might help:

• A good receiver—robust front-end design that doesn't overload easily; quality IF crystal filters with steep skirts and negligible ringing; IF shift or passband tuning; audio filtering or tailoring.

• Rotatable, directional antenna or antennas. One Yagi or quad is nice; two of them is at least three times as nice as one. Trust me on this.

• A high degree of patience.—*Jeff Bauer, WA1MBK, Chief Operator, W1AW*

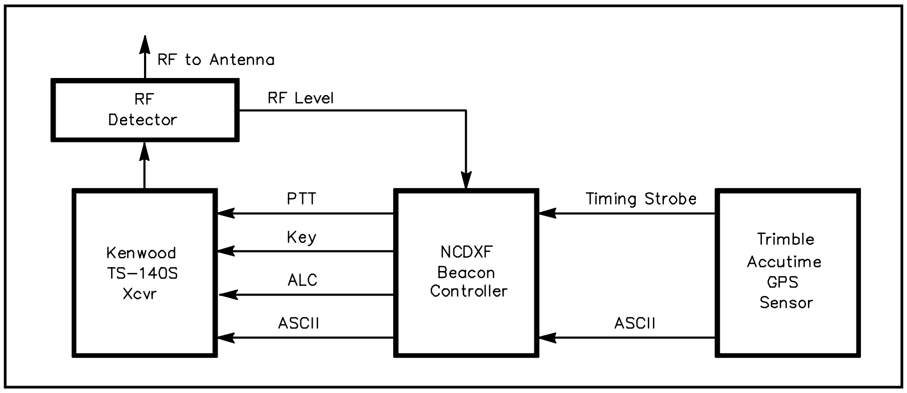

Figure 1—The block diagram of a Phase III beacon, showing the control-signal and RF paths.

(much better than we need!) and a serial-data line on which ASCII-character packets are sent.

The GPS receiver sends many different types of packets over the serial data line. Normally, a GPS receiver is used to provide location, as in the recent *QST* article on an automated direction finder[2] and the recent *QST* product review of the Trimble Scout GPS receiver that automatically displays Maidenhead grid locators.[3] Unlike those GPS applications, we ignore the packets that tell us our latitude and longitude, and use only the packets that specify the time (in UTC) and those that tell us that the GPS unit is "healthy."

Art Lange, W6RXQ, who works for Trimble, has provided technical guidance in the use of GPS timing. Within Trimble, Art championed the marketing of a self-contained stand-alone timing product. (Art was also the person who pushed internally to have the hand-held Scout unit display Maidenhead grid locators, a feature whose only known application is ham radio!)

Beacon Controller

A controller serves as the brain of each Phase III beacon (Figure 1). The other components are a Kenwood TS-140S transceiver, the Trimble Accutime GPS receiver and an RF detector that produces a voltage proportional to the RF output voltage. The major functions of the controller include receiving and interpreting ASCII timing packets from the GPS receiver, sending ASCII character strings to the transceiver to initiate frequency switching, controlling the power level of the transmissions from the beacon using the ALC of the transceiver and keying the transceiver to send Morse code. The controller uses the Intel 8748 computer-on-a-chip that was also used for the Phase II beacon controller, and which remains a cost-effective choice. It combines an 8-bit microprocessor with a 1024-byte EPROM (erasable, programmable read-only memory). The program for the 8748 is written in assembly language.

Timing

The once-a-second timing pulse from the GPS receiver is used to interrupt the microprocessor so it can add 1 to its internal clock and initiate a transmission when appropriate. The time packets from the GPS receiver are used to set or correct the clock's value. In addition to validating the parity on the characters from the GPS receiver, the controller requires two time packets in a row that agree as to the correct time before it sets or resets its internal clock. In practice, the time packets and pulses from the GPS receiver have rarely been garbled, but this arrangement means that a time drift from a lost or extra timing pulse will normally be corrected within 10 minutes and that nonsensical time packets will be ignored.

When the beacon is first powered up, the GPS receiver does not know where it is or what satellites to look for. It takes a few minutes for the GPS receiver to orient itself. The controller knows this. When first powered up, the controller waits until the GPS receiver declares itself "healthy," and then waits until it has received two time packets that agree. This process can take as much as 15 minutes.

Decoding the time messages is somewhat tricky. The program first assembles individual characters by watching an input connected to the serial-data line from the GPS receiver. It accumulates characters until it has a valid packet. If the packet is not a time packet, it is ignored. If a time packet arrives too close to the beginning of a second it is ignored, since there is an ambiguity as to whether it refers to the second just ending or to the second just starting. Unfortunately, the time-of-day information in the packet is provided as a floating-point number rather than as an integer. Converting the 48-bit floating-point number to an integer using 8-bit arithmetic is lots of fun. The entire algorithm for extracting the time of day from the character stream sent by the GPS receiver was written and tested as a C program on a PC before it was coded in assembly language.

Frequency Control

Setting the frequency of the transceiver

Copyright © 1998 ARRL

is relatively easy; the program sends the rig a string of ASCII characters. In the case of the Kenwood TS-140S, for example, one can set the frequency to 14.1 MHz by sending to the transceiver the characters "FA00014100000;" (the ";" is one of the characters that must be sent to the transceiver—it is not an ordinary punctuation mark). Each character must be bracketed with start and stop bits and the bits must have a specific duration on the wire, just as with computer serial ports and telephone modems. In this case, the program sets the value for the wire into one of the output bits and then loops, doing nothing until it is time to set the next value. (We might have used specialized UART chips for sending characters to the TS-140S and for assembling characters from the GPS receiver, but there was no need to add such hardware cost in this case.)

The W6WX/B housing on Mt Umunum. K6GSJ (l) and N6EK (r) are working on some of the coaxial cabling. The GPS unit is mounted on the pole at the far left corner of the housing.

Output Power Control

The output power level is controlled by adjusting the ALC input to the transceiver, using analog circuitry. A diode detector attached to the RF output of the transceiver produces a voltage proportional to the RF output. This voltage goes through a voltage divider, the ratio of which is controlled by the microprocessor. The output of the voltage divider is compared with a fixed reference voltage in an operational amplifier string that produces an ALC voltage for the transceiver to force the output of the voltage divider to match the reference voltage.

The tricky part of all this is to design the control circuitry so the keying envelope is smooth at all power levels and so the power level is accurate at all frequencies. It was necessary to provide a microprocessor-controlled ALC bias voltage so the ALC voltage is approximately correct for the desired power level before the transceiver starts to transmit.

Keying

The keying is accomplished electrically with one section of a 7417 open-collector driver chip. The program for sending Morse code, written by Jack Curtis, K6KU, was carried over from the original beacon. The messages to be sent by the beacon are stored in the memory as a string of ASCII characters, rather than as a string of dots and dashes. To send a character, the character is looked up in a table that specifies the dot and dash representation of the character. As one might imagine, a 1 bit in the specification stands for a dot and a 0 bit stands for a dash; the tricky part is to find some simple way to specify how many dots and dashes a particular character requires.

The solution is useful for any program that sends Morse code. The unused bit positions are filled with a 1 followed by enough 0s to fill the byte: A is encoded as 1010000 for dot-dash; B is encoded as 01111000 for dash-dot-dot-dot; and so on. If you have written any bit-twiddling programs, you can probably imagine the algorithm for sending the character specified in such a way: If the high-order bit of the specification is a 1, send a dot, otherwise send a dash. Shift the specification left one bit position, throwing away the old high-order bit and filling the low-order bit with a 0. If the specification is now 10000000, exit; otherwise, go back to send the next dot or dash.

Further Ideas

One can imagine building a computer-controlled beacon monitoring station and attaching it to the packet networks used by DXers. Such a device could include a computer-controlled transceiver for listening to the various frequencies and a GPS receiver for accurate timing. Perhaps DXers could call up the actual propagation history for the path from their location to particular beacons for the past hour, day, month or year.

The value of knowing that you can hear a particular power level is much more than the information provided by existing propagation prediction programs, because those programs do not fully account for atmospheric noise. Perhaps a systematic history of actual propagation, such as might be available with automated monitoring stations, could provide the raw data for refining these prediction programs so they could forecast signal-to-noise ratios instead of merely signal levels.

Another interesting possibility arises from the extremely precise timing of the beacon transmissions. Since the GPS receivers provide synchronization to the nearest microsecond, one could easily calculate the travel time for the radio signal from the beacon to the monitoring station to within a few microseconds. Would this allow one to determine the actual pattern of ionospheric reflections that occurred? Could this information shed light on unusual propagation modes such as skew paths? Fascinating possibilities exist, and need only be tested and developed by the users!

Conclusion

When complete, the Phase III beacon network will allow you to check for band openings on a particular band (any of the five bands from 20 to 10 meters) in 3 minutes. Or, you will be able to track the same beacon through five bands to determine the band that has the best propagation to that area. We are in for some interesting propagation experiences in the next several years as the sunspot count begins to move up from the approaching minimum. We can hardly wait!

Notes

[1]Trimble Navigation, OEM Sales, Post Office Box 3642, Sunnyvale, CA 94088, 408-481-8000.

[2]R. Flanagan and L. Calabrese, "An Automated Mobile Radio-Direction-Finding System," *QST*, Dec 1993, pp 51-55.

[3]M. Wilson, "Product Review: Trimble Scout GPS Hand-Held Global Positioning System Receiver," *QST*, Mar 1994, pp 77-ff.

Copyright © 1998 ARRL