Bob Simmons, WB6EYV

1268 Veronica Springs Rd, Santa Barbara, CA 93105; **pelican2@silicon.com**

APRS Unveiled

All the sneaky bit-level details of APRS messages...with an example packet.

Anyone attempting to create APRS equipment "from scratch" has immediately confronted the lack of a complete, detailed summary of all the APRS message requirements. Finding these details requires considerable effort, more than a little "luck," and a vocabulary that an APRS "novice" simply won't have. The lack of a simple, concise (and complete) summary has probably thwarted many attempts to create APRS technology.

This article (hopefully) addresses that problem. It provides a detailed description of a typical APRS message string, byte by byte, with a complete example provided. Basically, there is enough "message protocol" information to enable the creation of a circuit that can plug into the MIC jack of a non-APRS radio, and generate APRS packets that will successfully propagate across an APRS network.

The guidelines described here have been tested and proven. I used them to develop a 2 m APRS beacon transmitter design that has been in service for 2 years, and which has been used by several people. You can find more information about the beacon itself on my website: **www.silcom.com/~pelican2/PicoDopp/XDOPP.htm#MBCN**.

The Bell 202 Modulation Method

APRS data is transmitted at a 1200 baud data rate, typically on the (US) national APRS channel of 144.390 MHz. The data is frequency-modulated onto the RF carrier with two audio tones (1200 / 2200 Hz) that comply with a modified version of the Bell 202 modem standard. At the moment of data bit transition, a logic "zero" data bit is signified by "flipping" between tones, (for example, 1200 to 2200, or vice versa) whereas a logic "one" data bit is signified by no "flip" (steady frequency, either 1200 or 2200 Hz)

The Bell 202 standard specifies that the tone "flip" must be "phase contiguous," which basically means that the transition between tones must be as smooth as possible. The phase angle of the audio waveform (at the instant of tone switching) must be preserved and used as the "starting" phase angle for the new tone waveform. This minimizes switching transients and reduces the required bandwidth of the signal, resulting in improved signal to noise ratio (SNR). An example modulation waveform is shown in Figure 1.

Octets Versus Bytes

APRS message bytes are transmitted with no START or STOP bits, (each byte is called an octet) which is different from regular RS232 bytes. Most of the message data is encoded using ordinary ASCII characters, but there are some exceptions to this rule, described later. In all cases, octet (byte) data bits are transmitted least significant bit (LSB) first. (The bit order is B0 to B7.)

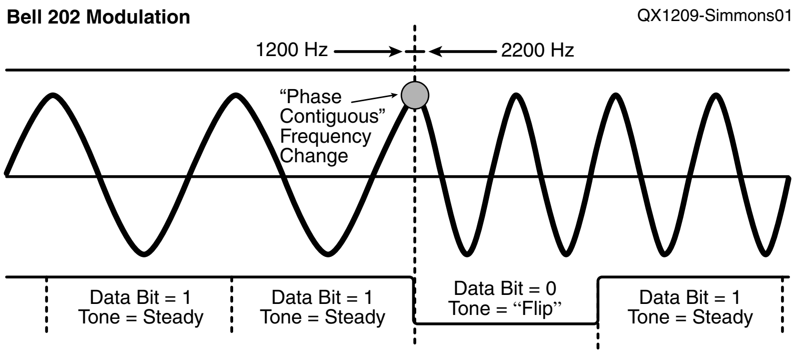

Bell 202 Modulation
QX1209-Simmons01
1200 Hz
2200 Hz
"Phase Contiguous" Frequency Change
Data Bit = 1 Tone = Steady
Data Bit = 1 Tone = Steady
Data Bit = 0 Tone = "Flip"
Data Bit = 1 Tone = Steady

Figure 1 — This graph illustrates the Bell 202 Modem modulation specification for the transition between tone frequencies to represent a 0 data bit.

APRS Position Reports: The Message Format

The APRS "example" message provided in this article complies with the format identified on page 33 of the APRS specification document, version 1.0.1, dated 29 August 2000.[1] In that document, (page 33) this message format is identified as: "Lat/Long Position Report Format with Data Extension (no Timestamp)" Byte by byte, the transmitted message has this format:

FDDDDDDdSSSSSSsVVVVVVvVVVVVVvCPsLLLLLLLHsLLLLLLLLHsCCCsSSSCccFF

With each byte defined as shown below:
(begin transmission)
(NOTE: Tone modulation begins IMMEDIATELY when transmission begins… no "dead carrier" time is provided.)

Starting Flag Bytes	(1 byte minimum, usually several identical bytes)	
F	Flag byte, always = 0x7E	
Preamble Bytes		
DDDDDDd	Destination addr	(7 bytes)("APDF00"+ SSID = "0" in this example)
SSSSSSs		
Source addr	(7 bytes = 6 call sign bytes + 1 SSID byte)	
VIA addr (0 - 8 addr = 0 - 56 bytes)	(7 bytes each = 6 call sign + 1 SSID byte) (NOTE: this example uses only 2 VIAs, shown below)	
VVVVVVv	VIA 1 address + SSID (optional, 6 + 1 bytes)	
VVVVVVv	VIA 2 address + SSID (optional, 6 + 1 bytes)	
(end of preamble bytes)		
Control and Protocol Bytes		
C	Control field	(1 byte, always = 0x03)
P	Protocol field	(1 byte, always = 0xF0)
(end of control and protocol bytes)		
Information Field	(APRS Position report, no timestamp, no messaging)	
s	Symbol	(1 byte)(APRS message type identifier)
	(! = exclamation mark = no APRS messaging, no time stamp)	
LLLLLLLH	Latitude	(8 bytes, XXxx.xxH) (XX = degrees latitude, 00 to 89) (xx.xx = minutes + dp + decimal minutes latitude) (H = hemisphere, N or S)
s	Symbol	(1 byte)(= primary or alternate APRS symbol table)
	(This identifies the type of APRS map icon to be displayed)	
LLLLLLLLH	Longitude	(9 bytes, XXXxx.xxH) (XXX = degrees longitude, 000 to 179) (xx.xx minutes + dp + decimal minutes longitude) (H = hemisphere, E or W)
s	Symbol	(1 byte)(= map symbol displayed on APRS screens)
(data extension begins here: COURSE and SPEED)		
CCC	Course	(3 bytes, xxx = 001-359, true degrees, 000 = stationary)
s	Symbol	(1 byte)(delimiter = "/")
SSS	Speed	(3 bytes, xxx = 000-999 knots)
(end of data extension)		
C	Comment	(0 - 36 bytes, 1 byte shown here)
(end of information field)		
Frame Checksum Bytes		
cc	FCS field	(2 bytes, CRC checksum, sent low byte / high byte)
(end of frame checksum bytes)		
Ending Flag Bytes		
FF	Flag	(2 bytes minimum)
(end of message…. end transmission)		

NOTE: As a courtesy to the receiving decoder (to make its job easier) it is not unusual to send several bytes of 0x00 data before sending the first FLAG byte. The pattern of several successive "0" bits causes the Bell 202 tones to constantly flip between tones, which simplifies the detection of the boundary between successive data bits, at the receiving decoder. For similar reasons, it is not unusual to send several FLAG bytes at the beginning (and end) of an APRS message, even though the APRS spec states only one FLAG byte is required.

Bear in mind that the receiver "at the other end" probably is an ordinary voice radio, with ordinary squelch circuits that will require 50 to 100 milliseconds (or more) of time to detect the presence of a signal on the channel, before any speaker audio is generated. At 1200 baud, 100 milliseconds of time equates to 15 transmitted bytes of message data… so the typical "courtesy" practice of transmitting several starting flag bytes is (probably) more important than the APRS specification indicates.

[1]Ian Wade, G3NRW, Editor, *Automatic Position Reporting System APRS Protocol Reference*, TAPR, 2000, p 33: **www.aprs.org/doc/APRS101.PDF**. [Yes, that should be "Automatic Packet Reporting System," but the title of the document was not changed. — Ed.]

Flag Bytes and Bit Stuffing

The lack of START and STOP bits in APRS messages means that some other method must be provided for an APRS decoder to "synchronize" itself with the bitstream of arriving messages.

The boundary between successive data bits can be identified by observing the 1200 / 2200 Hz tones, but the boundary between successive BYTES (end of one byte and start of next byte) must be identified by some other means. This is accomplished with special octets called FLAG bytes, consisting of a bit pattern of "01111110." (hex 0x7e = ASCII "tilde" character: ~)

This pattern (six consecutive "one" data bits) is reserved EXCLUSIVELY for FLAG bytes in the APRS specification. Therefore, any "accidental" occurrence of the same pattern (in the transmitted data) must be detected and prevented, but the data itself must (somehow) be preserved and recovered at the receiver. To accomplish this, a method is employed called "bit stuffing."

With "bit stuffing," each transmitted message is examined (bit by bit) as it is transmitted, to detect any (accidental) occurrence of five consecutive "1" bits. If such an event is detected, the sixth data bit (which might be either "1" or "0") is delayed, and a "0" bit is sent, ("stuffed" into the data stream) followed immediately by the (delayed) sixth data bit.

At the receiving end of the message, detection of 5 consecutive "1" data bits will alert the software that the following (6th) bit will determine if the data is a FLAG byte, or simply part of a regular message byte...if the 6th bit is a "1," the byte is judged to be a FLAG byte... otherwise, the 6th bit (which is a "0") will be ignored and discarded from the bitstream.

The Preamble: General Description

The message preamble includes the source, destination and VIA address bytes, and their associated SSID bytes. According to the APRS specification, the number of VIAs (which are user specified) can vary from zero to eight, but in the example provided in this article, the number is limited to two VIASs.

The SOURCE address is actually the FCC call sign of the transmitter operator, (6 characters, always spelled with CAPITOL letters) with an SSID byte appended to the end (7 bytes total...more info about SSID bytes later). If the call sign is less than 6 characters long, it is left-justified and padded with trailing ASCII "blank" characters, (0x20) followed by the SSID byte.

The DESTINATION address is not actually used in APRS… it is a legacy of packet communications, but APRS is a specialized subset of packet that does not employ this data field. Instead, it is filled with a fixed string of characters that identifies the type of software used to generate the APRS message. The string provided in this article's example was assigned to the author by Bob Bruninga, (developer of APRS) and consists of the text string "APDF00," with an SSID character of zero. This "assignment" is a matter of social courtesy, so that any problems in the resulting APRS messages can be traced back to the software author, and corrected. Anyone creating their own software should therefore contact Bob Bruninga for a similar assignment.

The VIA call signs (and their SSID characters) are optional (two are provided in this article's example). These are supplemental identifiers that provide information about the preferred signal path or direction for the message to take, and/or the preferred recipients for the message.

Typically these two VIAs are "WIDE1" (with SSID = 1) and "WIDE2." (with SSID = 2) These particular VIAs are actually requests for automatic "message relays" by any digipeater station that hears the messages.

The Preamble: SSID Bytes

SSID stands for "Secondary Station ID" (secondary station identification) SSID is encoded as a single byte that can express a number ranging from 0 to 15. Various (somewhat complex) "rules" for selection of SSID numbers are included in the APRS specification, but their actual values do not seem to be critical to message detection / propagation through the APRS network. In this article's example, the SSID for the DESTINATION station (= APDF00) is zero. For a WIDE1 VIA, this SSID should be one, and for a WIDE2 VIA, this SSID should be 2.

It is important to mention that the SSID values shown in ordinary computer displays (and in published articles) always include a hyphen character, so that "W6XYZ-0" indicates station W6XYZ with an SSID of zero, but in the actual transmitted message, no hyphen character is transmitted.

Furthermore, the SSID character is *not* an ASCII character; the SSID number is a 4-bit BINARY number, encoded into an 8-bit byte. The remaining bits are employed for other purposes and a description of the bits is provided below:

SSID bit 0 = extension bit (= 1 for last PREAMBLE field, = 0 otherwise)

SSID bits 1 to 4 = secondary station identification number (0 to 15, = "SSID" number)

SSID bits 5 and 6 = reserved, always = 1

SSID bit 7 = "control info," (C-bit) always = 0

The Preamble: Extension Bits and Byte Rotation

The APRS specification allows zero to eight VIA stations to be identified in a message, so some method must be provided to indicate how many VIAs are actually contained in any specific message. (This signals the end of the preamble block of data.) This is accomplished with the least significant bit (LSB) in ALL the preamble bytes. If the LSB (= bit 0) in a preamble SSID byte equals zero, then more preamble bytes remain in the message. If this bit equals one, no more preamble bytes remain. This bit is called the "extension bit"

This bit is often used in ordinary ASCII codes, and therefore it is not normally available for this purpose. To deal with this conflict, the ASCII codes used in the preamble (but *not* in the main message body) are limited to the 7-bit ASCII codes only (high order bit = always zero). This includes all "printable" ASCII characters, which are a subset of the entire ASCII set.

Furthermore, each ASCII byte (in the preamble only) is "rotated left" by one bit position, which is (arithmetically) equivalent to multiplying the character's binary value by two. This can be done without loss of information because the top bit of all 7-bit ASCII codes always equals zero. As a result of this "rotation," the LSB in each preamble byte is "liberated" for use as an APRS "extension bit."

For example, ASCII character "3" would normally be expressed as hex number 0x33, but in an APRS preamble, (due to the byte rotation) this would be transmitted as hex number 0x66, (if the extension bit = 0) or as hex number 0x67 (if the extension bit = 1). A table of ASCII characters with their regular and "rotated" values is in included in the APRS specification, in Appendix 3, Part 2.

ASCII "3" character	= 0x33	= 00110011	
Rotated character	= 0x66	= 01100110	(if extension bit = 0)
	= 0x67	= 01100111	(if extension bit = 1)

This "byte rotation" method is *only* applied to the preamble bytes — *not* to the entire contents of the APRS transmission. The first "rotated" byte is the first byte of the destination address, and the last "rotated" byte is the last byte of the last VIA address (SSID byte of the last VIA). If no VIAs are used, then the last "rotated" byte would be the SSID byte of the source address.

Summarizing, the extension bit in all PREAMBLE characters must be zero, EXCEPT for the VERY LAST character in the PREAMBLE, in which the extension bit must equal one.

Control and Protocol Characters

The control and protocol characters consist of two octets trans-

Table 1
Sample APRS Example Message

NAME	*VALUE*	*HEX DATA TRANSMITTED*								
(begin transmission)										
NULLS	(5X <nul>)	0x00	0x00	0x00	0x00	0x00				
FLAGS	(5X <tilde>)	0x7e	0x7e	0x7e	0x7e	0x7e				
(begin CRC calculation here)										
(begin bit stuffing here)										
(NOTE: The following bytes are left-rotated one bit position to provide bit 0 = extension bit)										
DESTINATION	APDF00	0x82	0xa0	0x88	0x8c	0x60	0x60			
DEST SSID	<SSID = 0>	0x60								
SOURCE	W6XYZ<sp>	0xae	0x6c	0xb0	0xb2	0xb4	0x40			
SRC SSID	<SSID=15>	0x7e								
VIA1		WIDE1<sp>		0xae	0x92	0x88	0x8a	0x62	0x40	
VIA1 SSID	<SSID=1>	0x62								
VIA2		WIDE2<sp>		0xae	0x92	0x88	0x8a	0x64	0x40	
VIA2 SSID	<SSID=2>	0x65								
(end left-rotation)										
CTRL CHAR	<control>	0x03								
PROTO CHAR	<protocol>	0xf0								
MSG TYPE	<msg type>	0x21								
LATITUDE	3426.22N	0x33	0x34	0x32	0x36	0x2e	0x32	0x32	0x4e	
SYMB TABLE	<primary>	0x2f								
LONGITUDE	11943.57W	0x31	0x31	0x39	0x34	0x33	0x2e	0x35	0x37	0x57
SYMB CODE	<car>	0x3e								
COURSE	264	0x32	0x36	0x34						
DELIMITER	/	0x2f								
SPEED	000	0x30	0x30	0x30						
COMMENT	COMMENT	0x43	0x4f	0x4d	0x4d	0x35	0x4e	0x54		
(end CRC calculation)										
CRC LSB	<CRC lo byte>	0xf9								
CRC MSB	<CRC hi byte>	0x3c								
(end bit stuffing)										
FLAGS	(5X < tilde>)	0x7e	0x7e	0x7e	0x7e	0x7e				
(end of transmission)										
(total = 81 bytes = 540 ms at 1200 baud)										

mitted immediately after the preamble. The control octet is transmitted first, and always consists of 0x0f. The protocol octet is transmitted next, always consisting of 0xf0. (No explanation is offered here for their purpose.)

Message Body (Information Field)

The message body (called the "information field" in the APRS spec) has various forms, depending on the type of APRS message being transmitted. The format of the data contained in this field is identified by the very first character, ("symbol") and different APRS messages use different characters for this field. (Refer to the APRS specification.)

APRS Map Symbol

The APRS map symbol is identified with two ASCII bytes located in the message body. One is located immediately after the latitude data field, the other immediately after the longitude data field. These two bytes are defined in the APRS specification, in Appendix 2.

The first character identifies one of two "symbol tables" in the appendix, (PRIMARY or ALTERNATE) each containing 93 "symbols" that will be shown on a map display when the message is received. The second character identifies one of the 93 symbols in the associated table.

Data Extensions

Data extensions are optional 7-byte fields that (if employed) express additional information, as described in Chapter 7 of the APRS specification. In this example, a data extension is employed to express the COURSE and SPEED of the reporting station.

Comment Field

The COMMENT field is optional. The maximum allowed length of the COMMENT field varies depending on the type of APRS message being sent. (See the details in the APRS specification for a particular message type.)

Frame Checksum

The frame checksum is calculated using a CRC calculation method. CRC refers to "Cyclic Redundancy Check," which consists of a special two byte "checksum" that allows the integrity of the message data to be tested, after it is received. The CRC checksum (= frame checksum) is generated when each message is transmitted, and evaluated at the destination, when the message is received.

The CRC checksum generation is performed by examining each byte in the transmitted message, using a special "formula" that is applied to each bit in the message. The result of this special "formula" is a two byte number that expresses the CRC (frame) checksum.

Bits that are added to the bitstream as a result of "bit stuffing" are *not* included in the calculation of the CRC checksum. The two-byte checksum itself is also excluded from the calculation.

CRC checksum calculation begins with the first byte in the PREAMBLE block, (immediately after the last starting FLAG) and ends with the last byte in the COMMENT block. The two CRC bytes are then transmitted LSB / MSB (low byte first, then high byte).

Rather than re-explaining it here in the author's own words, I defer to the source where I learned of it myself — many thanks to Scott Miller, N1VG, for posting this simple and concise explanation of the CRC checksum calculation method on his website:

Frame Check Sequence

One detail of the AX.25 format that deserves attention is the Frame Check Sequence (FCS) checksum. This is a two-byte checksum added to the end of every frame. It's generated using the CRC-CCITT polynomial, and is sent low-byte first.

The CRC-CCITT algorithm has plenty of published code examples, but the one I needed, and had trouble finding, was the algorithm for calculating the FCS one bit at a time, rather than a byte at a time. That algorithm is as follows:

Start with the 16-bit FCS set to 0xffff. For each data bit sent, shift the FCS value right one bit. If the bit that was shifted off (formerly bit 1) was not equal to the bit being sent, exclusive-OR the FCS value with 0x8408. After the last data bit, take the ones complement (inverse) of the FCS value and send it low-byte first.

NOTE: this text (and more useful information) can be found at Scott Miller's website, at: **http://n1vg.net/packet/index.php**

Those who choose to double-check this information against the AX.25 protocol specification, AX.25.2.2, dated July 1998, will find in section 3.8 that the order of bit transmission for the FCS data bits is opposite to that for the rest of the packet data, that is, the FCS bits (in the spec) should be transmitted most significant bit first (bit order B15 to B0 for the two FCS bytes) whereas the bit order for all other packet bytes should be sent least significant bit first (bit order = B0 to B7).

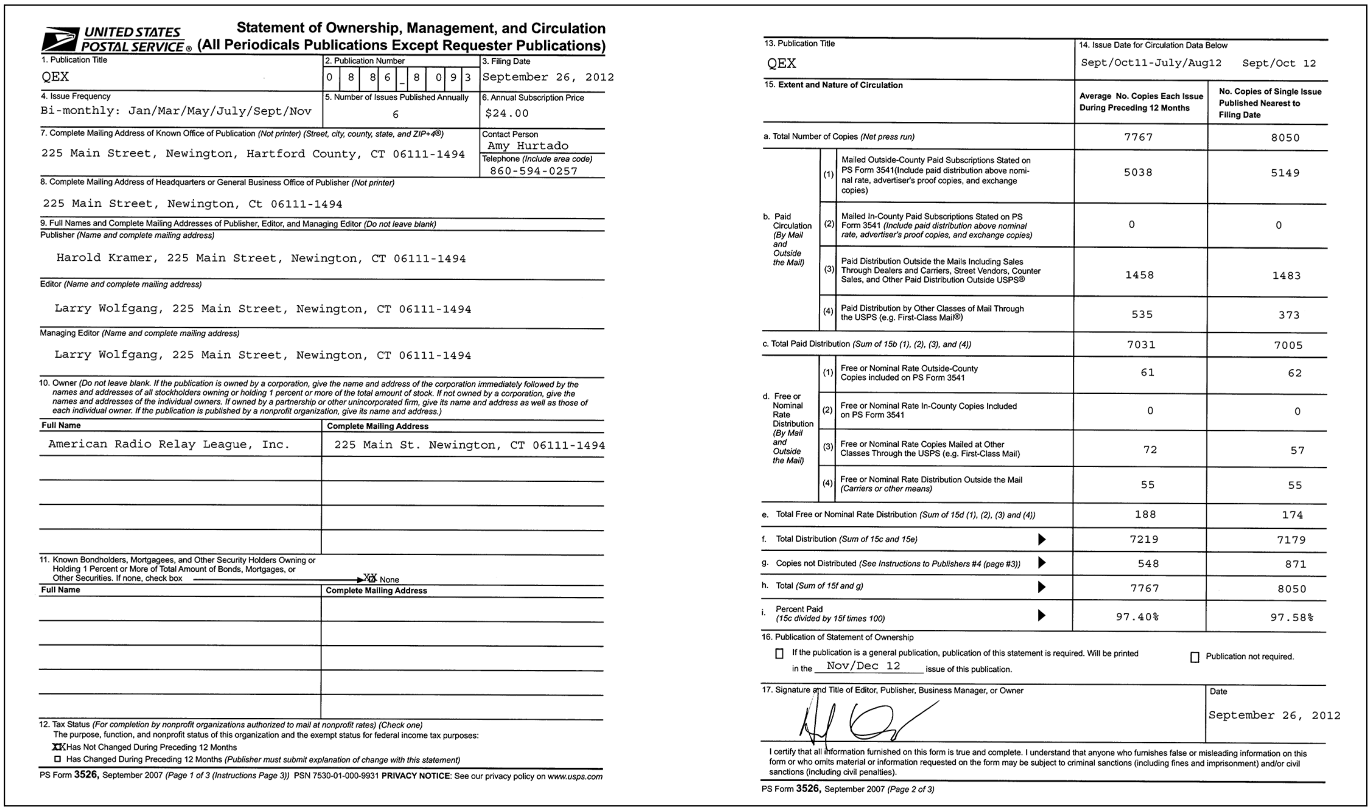
This contradicts the author's experience, in which successful on-air tests (and iGate postings) of the beacon transmitter's APRS packets used FCS data transmitted LSB first, just like the rest of the APRS packet data. There also is no mention of this "reversed" bit order in Scott Miller's comments on the topic, so it seems that the AX.25 spec is "suspect," on this point.

A Message Example

The message example given in Table 1 expresses a complete APRS message generated in compliance with the guidelines described in this article. For clarity, bits added as a result of "bit stuffing" are not shown in this data. Because a few of the bytes consist of unprintable ASCII characters, the data here is expressed in hexadecimal notation.

Bob Simmons, WB6EYV, was first licensed as a novice in 1964 at age 13, and remained licensed (more or less) constantly ever since. He also earned a commercial FCC license in 1967. He served Naval Reserve duty as a radar technician (ETR2) with about 6 months of total sea time. He spent several years of civilian work in nautical and marine electronics in Los Angeles harbor, as well as doing some land mobile radio work, followed by 5 years in flight line avionics, working on business jets. He moved to Santa Barbara, CA in 1992 and worked on vacuum deposition systems for 5 years, and held assorted odd engineering jobs at other times.

Presently, Bob is self employed and runs a website making and selling radio direction finding equipment and modules, with a majority of his "new" work spent creating embedded software / hardware and developing technologies to enable Internet-linked remote DF stations. His primary interest is developing and applying new technologies to old problems, and pushing the DF "art" forward.



UNITED STATES POSTAL SERVICE® **Statement of Ownership, Management, and Circulation (All Periodicals Publications Except Requester Publications)**

1. Publication Title QEX	2. Publication Number 0 8 8 6 _ 8 0 9 3	3. Filing Date September 26, 2012
4. Issue Frequency Bi-monthly: Jan/Mar/May/July/Sept/Nov	5. Number of Issues Published Annually 6	6. Annual Subscription Price $24.00

7. Complete Mailing Address of Known Office of Publication *(Not printer) (Street, city, county, state, and ZIP+4®)*
225 Main Street, Newington, Hartford County, CT 06111-1494

Contact Person
Amy Hurtado
Telephone *(Include area code)*
860-594-0257

8. Complete Mailing Address of Headquarters or General Business Office of Publisher *(Not printer)*
225 Main Street, Newington, Ct 06111-1494

9. Full Names and Complete Mailing Addresses of Publisher, Editor, and Managing Editor *(Do not leave blank)*

Publisher *(Name and complete mailing address)*
Harold Kramer, 225 Main Street, Newington, CT 06111-1494

Editor *(Name and complete mailing address)*
Larry Wolfgang, 225 Main Street, Newington, CT 06111-1494

Managing Editor *(Name and complete mailing address)*
Larry Wolfgang, 225 Main Street, Newington, CT 06111-1494

10. Owner *(Do not leave blank. If the publication is owned by a corporation, give the name and address of the corporation immediately followed by the names and addresses of all stockholders owning or holding 1 percent or more of the total amount of stock. If not owned by a corporation, give the names and addresses of the individual owners. If owned by a partnership or other unincorporated firm, give its name and address as well as those of each individual owner. If the publication is published by a nonprofit organization, give its name and address.)*

Full Name	Complete Mailing Address
American Radio Relay League, Inc.	225 Main St. Newington, CT 06111-1494

11. Known Bondholders, Mortgagees, and Other Security Holders Owning or Holding 1 Percent or More of Total Amount of Bonds, Mortgages, or Other Securities. If none, check box → [✓ None]

Full Name	Complete Mailing Address

12. Tax Status *(For completion by nonprofit organizations authorized to mail at nonprofit rates) (Check one)*
The purpose, function, and nonprofit status of this organization and the exempt status for federal income tax purposes:
[✓ Has Not Changed During Preceding 12 Months]
☐ Has Changed During Preceding 12 Months *(Publisher must submit explanation of change with this statement)*

PS Form **3526,** September 2007 *(Page 1 of 3 (Instructions Page 3))* PSN 7530-01-000-9931 **PRIVACY NOTICE**: See our privacy policy on *www.usps.com*

13. Publication Title QEX		14. Issue Date for Circulation Data Below Sept/Oct11-July/Aug12	Sept/Oct 12
15. **Extent and Nature of Circulation**		**Average No. Copies Each Issue During Preceding 12 Months**	**No. Copies of Single Issue Published Nearest to Filing Date**
a. Total Number of Copies *(Net press run)*		7767	8050
b. Paid Circulation *(By Mail and Outside the Mail)*	(1) Mailed Outside-County Paid Subscriptions Stated on PS Form 3541(Include paid distribution above nominal rate, advertiser's proof copies, and exchange copies)	5038	5149
	(2) Mailed In-County Paid Subscriptions Stated on PS Form 3541 *(Include paid distribution above nominal rate, advertiser's proof copies, and exchange copies)*	0	0
	(3) Paid Distribution Outside the Mails Including Sales Through Dealers and Carriers, Street Vendors, Counter Sales, and Other Paid Distribution Outside USPS®	1458	1483
	(4) Paid Distribution by Other Classes of Mail Through the USPS (e.g. First-Class Mail®)	535	373
c. Total Paid Distribution *(Sum of 15b (1), (2), (3), and (4))*		7031	7005
d. Free or Nominal Rate Distribution *(By Mail and Outside the Mail)*	(1) Free or Nominal Rate Outside-County Copies included on PS Form 3541	61	62
	(2) Free or Nominal Rate In-County Copies Included on PS Form 3541	0	0
	(3) Free or Nominal Rate Copies Mailed at Other Classes Through the USPS (e.g. First-Class Mail)	72	57
	(4) Free or Nominal Rate Distribution Outside the Mail *(Carriers or other means)*	55	55
e. Total Free or Nominal Rate Distribution *(Sum of 15d (1), (2), (3) and (4))*		188	174
f. Total Distribution *(Sum of 15c and 15e)* ▶		7219	7179
g. Copies not Distributed *(See Instructions to Publishers #4 (page #3))* ▶		548	871
h. Total *(Sum of 15f and g)* ▶		7767	8050
i. Percent Paid *(15c divided by 15f times 100)* ▶		97.40%	97.58%

16. Publication of Statement of Ownership
☐ If the publication is a general publication, publication of this statement is required. Will be printed in the Nov/Dec 12 issue of this publication.
☐ Publication not required.

17. Signature and Title of Editor, Publisher, Business Manager, or Owner
[illegible]

Date
September 26, 2012

I certify that all information furnished on this form is true and complete. I understand that anyone who furnishes false or misleading information on this form or who omits material or information requested on the form may be subject to criminal sanctions (including fines and imprisonment) and/or civil sanctions (including civil penalties).

PS Form **3526,** September 2007 *(Page 2 of 3)*