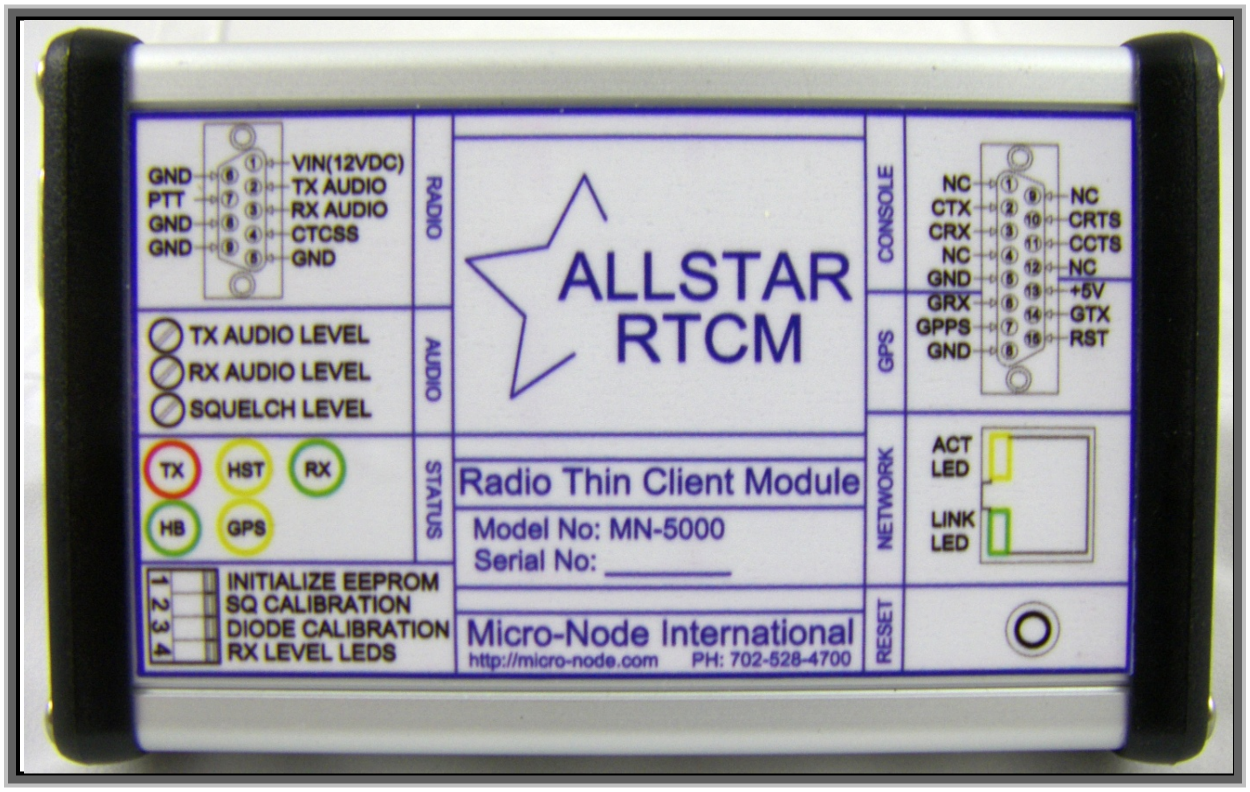
RTCM - RADIO THIN CLIENT MODULE

An Open-Source VOIP-Based Voting Multi-Receiver and Simulcast Transmit System



RADIO THIN CLIENT MODULE

An Open-Source VOIP-Based Voting Multi-Receiver and Simulcast Transmit System which can also function as an extremely simple, robust, low-power/complexity radio interface device for general purpose (non-voting/GPS-based) applications.

When used as a simple radio interface, it allows connection of a simplex or full duplex (repeater) radio system to a host-based radio control system via VOIP (Voice Over IP) and does so in a manner that will not require specialized network provisioning (such as Firewall/NAT port forwarding). In this mode, it does not require connection to a GPS receiver, although if position reporting for a portable application is desired, it does support such a connection.

The following introduction and description is mainly for use of the RTCM as a voting multi-receiver and/or simulcast transmitter system. Some of the information contains general concepts that are applicable to the RTCM regardless of its application.

INTRODUCTION

In many two-way radio applications, both for repeater systems and simplex base-stations, it is often difficult to have reliable reception when there is signal impairment due to terrain or other obstacles. One common way to solve this problem is by use of a voting multi-receiver system. Such a system is comprised of a number of receivers, located at diverse locations. The location of these receivers is chosen so that the combined coverage of all the receivers more or less fill-in the entire desired coverage area, even though it may take a number of receivers to accomplish this. All such receivers are connected ("linked") to a central location (generally the transmitter site), and a device compares each receiver's signal strength and selects the one with the best signal. This is called voting. Often in such a system, there is one single high-powered transmitter located at a central site that all mobile/portable stations would be able to receive. Sometimes, one single transmitter cannot be located where coverage of the entire desired area is possible. In such cases, multiple transmitters are deployed in several locations. Since all these transmitters are on the same frequency, they must be very precisely locked to the same frequency and be transmitting the exact same audio at the exact same time. Not doing so would severely degrade their reception. Traditionally, receivers that comprise a voting system are each connected to the central site via either a UHF or microwave link. Since this requires a receiver, link transmitter and link receiver for every receive site, implementation and continued cost of such a system typically becomes quite significant and in most cases prohibitive, and requires a high complexity of implementation. There are, however, many advantages of implementation of such a system that warrant the cost and effort required.

Recently, with the advent of VOIP -based interconnections for radio systems, it is now possible to implement a voter system using VOIP to link the receivers to the central site, and therefore reduce the system requirements and complexity and cost by eliminating the radio links between the receivers and the central site. There are a few commercial vendors of VOIP-based voting systems that offer a functional and reliable, yet proprietary and extremely high-cost solution. Although using VOIP-based technology significantly reduces the cost and complexity of implementation of a voting system, the cost of the commercial VOIP-based devices is still quite prohibitive for all but large commercial and/or government applications. Therefore, it seemed appropriate to offer a completely open-source, open-hardware solution that makes the cost and complexity of a VOIP-based voting system nearly trivial and gives almost anyone the ability to implement this type of a system that wishes to do so.

DESCRIPTION

In order to successfully implement a VOIP-based solution for this purpose, it is necessary to have a precise time reference in order to be able to re-assemble (time-wise) the audio streams at the central site, since VOIP has inherent inconsistencies in time delay (jitter and such). Clearly, the best and most reasonable and economic source of such precise timing data is GPS. Therefore, a GPS receiver (with a 1 pulse per second output) must be made available at each receiver location (along with the central site). Since precise timing is necessary, it precludes the use of any implementation which relies on a traditionally host-based operating system, such as Linux or UNIX, because of limitations of timing and scheduling precision and latency. Therefore, a dedicated hardware device with a simple embedded micro-controller was necessary to accomplish the necessary precise timing requirements. In addition, low cost, in comparison with a host based solution makes such a simple dedicated hardware device at each receiver location gives this solution a great advantage. Such a dedicated hardware solution significantly increases reliability, has a far lower power consumption, and is physically smaller (in comparison with a host-based solution). The "central site" functionality (the device which receives all of the VOIP-based audio streams from the receivers and selects which receiver's audio to use) is implemented in a host-based computer system (running Linux) as a channel driver (chan_voter) along with app_rpt under Asterisk PBX. The host-based system on which this runs, along with a dedicated hardware device (which gives access to the precise GPS-derived time data) is typically located with the system's transmitter (or at least one of them). Each dedicated hardware device is also capable of simultaneously generating time-synchronized audio out to drive a transmitter (or multiple transmitters in a simulcast system). The audio out of all the dedicated hardware devices at all locations will be identical at same time. There are facilities available for on line monitoring of the voting system (audio and signal strengths), in addition to recording (logging) facilities.

PROVISIONING AND CONFIGURATION
GENERAL SYSTEM PROVISIONING

There needs to be: Some number (> 1) of receiver sites that contain 1 of identical model receivers (or at least ones with identical characteristics) and 1 RTCM (Radio Thin Client Module), 1 GPS receiver and some sort of Internet connection (can be a private LAN). Some number of transmitter sites (typically 1, but not necessarily) that contain a transmitter, 1 RTCM, 1 GPS receiver, and some sort of Internet connection (can be a private LAN). (A site may have both a receiver and transmitter sharing the same RTCM, GPS receiver, and Internet connection). A Linux host running Asterisk with app_rpt and chan_voter, along with a RTCM (Radio Thin Client Module), a GPS receiver, and an Internet connection with sufficient bandwidth to accommodate all the packet traffic from/to all the receiver and/or transmitter sites. This location may also have a receiver and/or transmitter connected to the RTCM.

IP (INTERNET) NETWORK PLANNING AND PROVISIONING

The receiver/transmitter sites may be connected to the "master" site (the one containing the Linux system running chan_voter/app_rpt/Asterisk) via either a LAN (either physical or extended Local Area Network), or via a public Internet connection, or some combination thereof. Regardless of the interconnection method, the following criteria must be met: Sufficient bandwidth must be available at all times between each receiver/transmitter site and the "master" site. The latency of such connections must be within reasonable limits, and certainly must be within the limits configured in both the RTCM devices and chan_voter. The "master site" really should be on a static IP address. Although not 100% necessary, it could really make for an unreliable installation. Any of the devices (including the "master" site) may be behind NAT and/or a firewall. There is no specific requirement for the RTCM to have any sort of non-NATed access, since the actual IP address and UDP port that is public really makes no difference. All authentication is based upon content (CHAP-type tokens), and not upon IP address in any way. The "master" site may be behind NAT and/or a firewall, as long as its listening UDP port (generally 667) is exposed to the public (unless entire system is on an entirely private network). If the implementer wishes, telnet access may be configured for the RTCM, and in that case, the port selected for telnet must be made publicly accessible (since the RTCM firmware supports dynamic dns, it certainly may be on a dynamically-assigned public IP address when the telnet option is used, and its actual IP address really doesn't matter if telnet is not configured). If the JAVA client for monitoring is used, the server to which the clients connect must be on a fixed IP address (obviously.. A high-availability server on a dynamic IP address would be rather silly!). Typically, the IP bandwidth usage in each direction, when a signal is present is approx 100 kbps (including UDP/IP overhead, 1 direction only if just a receive or just a transmitter site, or both directions if a receive/transmit site). The "trunk" (or streaming) connection between the "master" site and the JAVA client distribution site (if used) uses approx 200 kbps of bandwidth when a signal is present (1 direction only). Each JAVA client requires approx 100 kbps for monitoring. On a public Internet connection, it’s a good rule-of-thumb to get a connection capable of 3 to 4 times the required bandwidth in the slowest direction (often, such connections are asymmetrical as far as their speeds are concerned). If possible, avoid wireless connections. DSL is your friend. It may be slow, but it tends to be more reliable and consistent then other forms of broadband connections. If a wireless connection is really required, use a high reliability solution, such as the devices from Ubiquiti, which are very high-quality, reliable, and inexpensive.

RECEIVER (AND/OR TRANSMITTER) SITE

A receiver and/or transmitter site consists of a receiver capable of providing direct access to the output of the FM discriminator (and output of CTCSS/CDCSS tone decoder, if applicable) and/or a transmitter, a RTCM, a GPS receiver capable of providing a precise 1 pulse per second signal (such as the Trimble Thunderbolt for transmit Simulcast),or Garmin LVC18 for receive voting only), and an Ethernet-based Internet connection.

"OFFLINE" (FAILOVER) MODES

If the IP-based communication between the RTCM and the host fails, the RTCM is capable of being configured to provide audible indication of such failure and/or restoral of communications (by transmitting a tone/CW message over the air).

In addition, for full-duplex repeater systems that have both a transmitter and a receiver connected to the RTCM, a mode is available whereby the RTCM acts as a VERY limited local repeater controller (passes audio from receiver to transmitter, has transmit hang-time, and provides a CW identifier, and can encode locally-generated CTCSS tone), in the event of IP-based communication failure.

RTCM CONFIGURATION SETTINGS

There are several categories of configuration parameters accessible through the RS232 and telnet console ports of the RTCM. There is a main menu for general configuration parameters, an "IP Parameter" menu for parameters dealing with IP networking, and "Offline Mode" menu for parameters dealing with "Offline" (failover) mode (See Appendix A).

IP SETTINGS – Standard IP setup parameters (IP Address, Netmask, Gateway, DHCP, etc).

RTCM SERVER – RTCM server address, specified as a Fully-Qualified Domain Name (FQDN), UDP port on the RTCM server, and (optionally) local UDP port (generally left at default), and client and server passwords.
The firmware in the RTCM does a DNS resolution once per minute and resolves the IP address for the RTCM server, based upon the FQDN specified in this configuration. Therefore, the actual IP address of the RTCM server may be changed (and its associated DNS entry), and the RTCM firmware will continue to be aware of its current IP address (thus not requiring re-configuration of the RTCM firmware if the IP address of the RTCM server changes).

The “client password” is the password, as configured in the server, that identifies the RTCM (client), and the “server password” is the common server password as configured in the server (which allows verification of the server's identity by the RTCM firmware).

TX BUFFER PARAMETERS – Transmit buffer length (and buffer delay). This allows for simultaneous time-consistent audio output at all RTCM’s in a system. Buffer length is set to allow for maximum network latency between “master site” and receive/transmit sites. The Buffer Delay parameter is included for development purposes currently, and will be removed in future releases.

GPS RECEIVER PARAMETERS – Baud Rate, Data Polarity and PPS Signal Polarity (and time offset, for GPS receiver.

TELNET PARAMETERS – Port Number, Username, and Password for Console Telnet Access.

DYNDNS PARAMETERS – Setup parameters necessary to make dynamic DNS work (if desired).

EXTERNAL TONE DECODE/COR TYPE – Sets mode for external tone decode input (enable/polarity), etc.

OFFLINE MODE (FAILOVER) PARAMETERS - Set various parameters associated with "Offline" (failover) mode, including its general mode of operation, CW messages and timing, CTCSS tone, etc.

BOOTLOADER ADDRESS – Sets IP Address of Boot loader.

The console gives access to a menu allowing setting of the above-mentioned parameters in addition to various system-related functions, reboot, save parameters, status, etc. In addition, the console displays important system status messages as various events occur (such as changes in GPS status, changes in DNS resolution, etc).

RTCM SETUP AND INSTALLATION

Appropriate connections need to be made between the receiver's discriminator and optionally its tone decode output and/or the transmitter and its line level (or MIC input, attenuated) and PTT signals. The GPS receiver needs to be connected, and the board needs to be connected via Ethernet to an Internet connection, and, of course, power (7-24Vdc) needs to be applied to the RTCM. (See Appendix B)

With no antenna connected to the receiver, the RTCM needs to have its “squelch calibration” procedure performed. Then the receive level needs to be set, either by putting the board in “level setting” mode (SW2-4 On), or with use of the console. Once the above procedures are performed, the board is ready to have its operating parameters configured and then it will be ready for use. Set the IP Setting, TX Buffer Parameters, GPS Parameters (if necessary), Telnet Parameters, DYNDNS Parameters, and External Tone Decode appropriately for the installation. The RTCM needs to be on an Ethernet segment (typically a router or switch or wireless bridging device of some sort) that has connectivity to the “master” site, either via public Internet or via private LAN (extended, perhaps). It may be on a network that allocates IP addresses (LAN and/or public IP address) dynamically. The VOTER protocol and the RTCM firmware are designed specifically to allow for not needing the IP address of any RTCM to be long-term consistent, or even important, for that matter. All authentication and identification is based upon tokens, **not** IP address or port. If telnet access is desired, the RTCM needs to at least have its telnet port publicly accessible (through a NAT/Firewall if used), otherwise it is not necessary. Even if telnet access is used, a dynamically-assigned public IP address will work, because dynamic DNS (dyndns.org) is supported in the RTCM firmware. This document assumes sufficient familiarity with IP and general networking concepts, and therefore, I will not go into specific details of network provisioning. The RTCM Server Settings on the RTCM will be covered in the next section.

CONFIGURATION AS A SIMPLE RADIO INTERFACE (non-Voting/Simulcast)

The RTCM may be configured and used as a simple radio interface, if multi-receiver voting/simulcast is not desired for the radio.

To accomplish this, set the PPS Polarity (main menu item 10) to "2 - NONE". This will indicate to the RTCM that it is not to operate in Voting/Simulcast mode.

You may still use the on-board squelch logic (hardware and software) and connect the RTCM's audio input to the receiver’s discriminator. If you do not wish to do so, and rely entirely on the receiver's COR/CTCSS decode circuitry, you may connect the RTCM's audio input to processed audio out from the receiver. If you do so, you must set main menu item 13 ("COR Type") to "1 - (IGNORE COR)". You therefore must connect a signal that represents valid COR *AND* CTCSS decode to the external CTCSS input, and set main menu item 12 ("External CTCSS") to the appropriate value.

OFFLINE (FAILOVER) MODE

The RTCM may be configured to allow for notification of loss of IP-based communication (and restoral thereof) with the host system.

When Offline mode is enabled, the RTCM sends a CW message (specified by "CW Offline String" - Offline menu item 5), when communications with the host is lost. If specified (by "Offline (CW ID) Period Time - Offline menu item 7), the RTCM will repeat this message every amount of time specified until communications is restored. When communications is restored, the RTCM will send a CW message (specified by "CW Online String" - Offline menu item 6).

The speed at which CW messages are sent is specified in Offline menu item 1 ("CW Speed"). This sets the length of 1 CW element in (1/8000 second periods). 400 would be 50ms per element (which is a good speed).

When the CW message is to be sent, the RTCM asserts PTT to the transmitter, waits a time period specified by offline menu item 3 ("Pre-CW Delay"), sends the CW message, waits a time period specified by offline menu item 4 ("Post-CW Time"), and de-asserts PTT to the transmitter.

The RTCM supports 4 different "Offline" modes (specified by offline menu item 1 "Offline Mode"):

- 0 - NONE ("Offline" Mode disabled)
- 1 - Simplex (Sends "Offline" message when off-line, and periodically while still off-line, if specified)
- 2 - Simplex w/Trigger (same as "Simplex", but re-triggers the "Offline" sequence when receive un-keys)
- 3 - Repeater (same as "Simplex", but also acts as local "dumb" repeater while offline)

For those applications where CTCSS tone is normally included in the host audio stream (and the transmitter is connected directly to the modulator), the RTCM is capable of locally-generating CTCSS tone when "offline". The offline menu items 9 and 10 specify the Tone and Level of this locally-generated CTCSS tone.

OVERALL SYSTEM / MAIN SITE CONFIGURATION

app_rpt/Asterisk uses a channel driver *"chan_voter"* , which implements the "master site" functionality, providing an Asterisk channel which app_rpt may open as a radio interface. This driver receives (and/or sends) UDP packets (typically on UDP port 667) from/to all associated RTCM's at receive and/or transmit sites. In addition, the channel driver (optionally) may be configured to send a stream of UDP packets to a predetermined IP address and UDP port. This stream is meant to be redistributed by the *votmond* program running on a high-availability, high-bandwidth Linux Server, and give users access to monitoring of the VOTER audio and signal strength information in real time. Chan_voter uses a configuration file, */etc/asterisk/voter.conf*, which contains typically several stanzas.

Here is an example of a *voter.conf* file:

```
[general]
port = 667
buflen = 500
password = BLAH

[1234]
client1 = pass1,transmit,master
client2 = pass2,transmit
client3 = pass3,transmit
client4 = pass4,transmit
client5 = pass5
streams = 12.34.56.78:1667
plfilter = y
txctcss = 100.0
txctcsslevel = 100
txtoctype = none
thresholds = 255,110=5
linger=6

[5678]
buflen=750
mobile1 = mobp1,adpcm,transmit,nodeemp
```

The *"general"* stanza specifies the UDP port that the RTCM clients are expected to send packets to (the "*port"* parameter), the receive buffer size in milliseconds (the *"buflen"* parameter), and the SERVER password (the *"password"* parameter).

Each additional stanza is named in an appropriate manner (generally, the associated node number in the *rpt.conf* file). This stanza name references the channel name (in this case "*Voter/1234"*) that is used when specifying the channel within the Asterisk system.

In this example, the [1234] stanza contains"client1" is designated as the "master" timing source (and is both a receiver and transmitter client), "client2" thru "client4" are receiver/transmitter clients, and "client5" is a receive-only client. The [5678] stanza is for a cellular (3G/4G) connected mobile non-voting client.

Within each named stanza (not the *"general"* stanza), the RTCM clients are defined (by specifying *DISPLAY_NAME = password[,options]* for each client), and operating parameters, such as *"streams"*, which specifies a list of *IP_ADDR:UDP_PORT* values indicating where to send the votmond redistribution stream associated with the specified system, *"plfilter"*, which, if set to true, enables a 300Hz low pass filter, used when a CTCSS/CDCSS tone is higher in frequency than 120 hertz, *"txctcss"*, *"txctcsslevel"*, and *"txtoctype"* (if transmit CTCSS is desired), and *"thresholds"* and *"linger"* which define the behavior of the voting selection algorithm, and "*buflen*" which optionally sets the default buffer length for the specified stanza only.

The “options” for each specified *DISPLAY_NAME* are as follows:

master – This specifies that this client is the Master Timing source (the RTCM client that is on the same LAN as the Asterisk server. There can **only** be 1 Master Timing source per entire Asterisk server.

t*ransmit* – This specifies that this client is intended to have transmit audio sent to it and will have a transmitter connected to it.

a*dpcm* – This specifies that this client is to be sent audio in ADPCM format, rather than Mu law.

nulaw - This specifies that this client is to be sent audio in 4000 samples/second Mulaw (Nulaw, as we call it), rather than the standard 8000 samples/second.

n*odeemp* – This specifies that the client is not to perform de-emphasis of the receiver audio. This is only to be used with non-voting clients.

b*uflen=valu*e – This specifies a buffer length for the specified client only.

gpsid[=value] – This specifies a gps identity to associate with the specified client (as referred in the */etc/asterisk/gps.conf* file).

If no options are specified for a client, then none of the options specified above are enabled for the particular client, and will be treated as a standard (non-master), receive-only, Mu law audio client.

On each RTCM board, the *“Client Password”* must be set to the associated value for the desired client as specified in the stanza of the *voter.conf* file. Using the above example, the “*CLIENT2*” VOTER board would be configured to have a *“Client Password”* of *“pass2”*. The *“CLIENT2”* designation is only used and referenced by chan_voter, and is not specified or known in any way by the RTCM client.

On each RTCM client, the *“Server Password”* must be set to the value specified in the *“password”* parameter in the *“general”* stanza of the *voter.conf* file.

The *“buflen”* parameter in the *“general”* stanza specifies the receiver buffer length in milliseconds. This parameter should be set to the maximum expected network latency, plus a little padding (100 milliseconds of padding is a good amount). The default is 500 milliseconds, and is what the value will be set to if not specified. Buffer length may be specified on a per-stanza and per-client basis, also.

The Asterisk server at the “main site” (in addition to all the ports that are canonically made public to operate a standard app_rpt installation) **must** have the port specified in the *“port”* parameter in the *“general”* stanza in the *voter.conf* file made publicly available, or at least available to all possible IP addresses from which its associated RTCM clients could possibly be operating.

The *“Tx Buffer Length”* parameter on each RTCM client, is similar to the *“buflen”* parameter for the receive side. It sets the transmit buffer length in samples (8000 samples per second, so 8 samples = 1 millisecond). Like the *“buflen”* parameter, this should be set to the maximum expected network latency plus a little padding. 800 samples is the minimum usable value. Setting the parameter to 3000 (375 ms) is probably a good place to start.

In both the receive and transmit cases, the buffer length is a fixed value. This will hard-set the delay time between actual reception and presentation time to the Asterisk channel (in the case of receive), and presentation from the Asterisk channel and actual transmission (in the case of transmit).

Since the overall system time delay (the amount of time between reception and re-transmission of the signal) is determined as the sum of the receiver buffer length, the (very minimal) latency within app_rpt/Asterisk and the Transmit buffer length, it is very necessary to keep these settings to a reasonable minimum (without sacrificing packet consistency/integrity by making it too short based upon network latency/jitter).

If transmit CTCSS is desired, the *“txctcss”* and *“txctcsslevel ”* (and optionally the *“txtoctype”*) parameters must be specified. In order to facilitate generation and transmission of CTCSS along with the normal system audio, the RTCM board (and host) generate “flat” (pre-emphasized/ limited) audio, which is intended to directly modulate a transmitter (to be applied at a direct modulation point, NOT a line level or microphone input).The *“txctcss”* parameter specifies the transmit CTCSS frequency in hertz.

The *“txctcsslevel”* specifies the level of the CTCSS tone (0-250). Optionally, the *“txtoctype”* parameter may be specified, which determines the “turn-off” (when transmitter un-keys) style for the CTCSS. Setting this parameter to *“none”* (which is default) means that the transmitter immediately stops transmitting when the system “un-keys”. Setting this parameter to *“phase”* means that the CTCSS tone will briefly be transmitted 120 degrees out of phase when the system “un-keys” (also known as “reverse-burst”, etc). Setting this parameter to “notone” means the transmitter will continue transmitting for a brief period with no CTCSS after the system “un-keys” (also known as “chicken-burst”).

If transmit CTCSS is **not** desired, **DO NOT** specify **ANY** of the *“txctcss”* ,*“txctcsslevel ”* or “*txtoctype”* parameters. The audio that is produced by the RTCM board (and the host) will be normal line-level (intended for a line or microphone input) and will NOT be pre-emphasized.

Since token-based authentication and identification is utilized, having the UDP port open to the public should not be a significant security risk.

There **must** be a RTCM client on the **same LAN** (very low latency) as the Asterisk server implementing the “master site”, which acts as the Master Timing source. This allows chan_voter to have a consistent, reliable, accurate timing source with which the timing information from all other inbound packets are compared and appropriately processed, and from which to generate accurate timing information for time-consistent transmission purposes.

VOTING SELECTION ALGORITHM AND ASSOCIATED PARAMETERS

Voting selection (the choice of client from which to take the audio stream) is based upon the RSSI (relative signal strength/quality) value associated with each particular client's received signal. The parameters associated with voting selection are the *“thresholds”* and *“linger”* parameters specified for each voter instance.

If **neither** of these parameters are specified, the voting selection algorithm will default to choosing the client with the highest RSSI number (and the last-most one listed in the *voter.conf* file, if a tie exists) each and every received audio frame (every 20 milliseconds). Although this is a functional option, **this is probably NOT what you really want.** Doing so will cause the received signal to have an audible "squelch-tail" at the end of a transmission if there is a receiver receiving the signal strongly in addition to a receiver receiving the signal at a low, noisy level. This, however might be appropriate in very rare instances where receivers are placed in locations that require very agile following of the strongest signal. It is very unlikely, however, that this is the case. It's far more likely that it is appropriate to configure a set of *"thresholds"* and optionally a *"linger"* time that is different than the default (which is 6 frame periods of 20 milliseconds).

The *"linger"* parameter allows specification of a default *"linger"* value (in 20 millisecond frames)of other then the system default value (6).

The *"thresholds"* parameter is a comma-separated list of RSSI threshold values each specified as follows:

MIN_RSSI[=REASSESS_FRAMES[:LINGER_FRAMES]]

MIN_RSSI is the minimum RSSI value for this threshold (1-255)

REASSESS_FRAMES is the number of 20ms frames for the client to remain selected at this threshold level before being re-assessed (may be specified as 0, meaning "re-assess at next frame").

LINGER_FRAMES is the number of 20ms frames for the client to remain selected after no longer being at this threshold (or any other) (may be specified as 0, meaning "do not linger")

If MIN_RSSI is specified without the other values, its REASSESS_FRAMES is considered to be infinite (it will not get re-assessed if its RSSI value remains at this threshold), and LINGER_FRAMES will be its default value (6).

If MIN_RSSI and REASSESS_FRAMES are specified, LINGER_FRAMES will be at its default value(6).

A specification for REASSESS_FRAMES **must** be present if there is a specification for LINGER_FRAMES.

For example:

thresholds = 255,110=5

Meaning:

If the RSSI level is 255, stay selected on that client as long as it still has an RSSI of 255, else if the RSSI level is at least 110, stay selected there for no more than 5 frames. If the RSSI drops below 110, re-assess every frame (since no value below 110 was specified).

Also, since there were threshold values specified and the "linger" value remained default, any client that meets any specified threshold, stays selected for 6 frames when no clients meet any thresholds any longer.

If, instead the example was:

thresholds = 255,110=5:10

Then:

The same would be true except that if a client meets the "110" threshold (and not the "255" threshold), and then no more clients meet any more thresholds, the client that met the "110" threshold would stay selected for 10 frames rather than the default 6.

This whole methodology is rather strange, but it seems to be appropriate and effective. In the future, it may be improved or modified in some way, but for now it is more than sufficient.

COMBINED USE OF VOTER/SIMULCAST AND NON-VOTER/SIMULCAST CLIENTS

In each chan_voter instance, there may be a combination of voter/simulcast and non-voter/simulcast clients. The receive audio from all individual non-voter/simulcast clients are "mixed" together along with the result of the voting receiver selection (of those receivers that are voting/simulcast clients). The result of that "mix" is presented to Asterisk as the voter instance's input (receive) audio.

All transmit clients receive the same transmit audio.

VOTERMON (JAVA MONITOR CLIENT) AND VOTMOND SERVER CONFIGURATION

A JAVA Client (applet) known as *"votermon"* is available, allowing real-time monitoring of the VOTER audio stream and relative signal strengths of all receivers in a system, as well as the result of the "voting" selection. It is a JAVA applet to be embedded into a web page served by an HTTP/HTTPS Web server. Based upon parameters specified on the web page in which it is embedded, it will connect to a server running the *"votmond"* program. This program takes a UDP stream from a running instance of chan_voter (as specified in its stanza's *"streams"* parameter), and makes it available for users of the JAVA applet to connect. The server should be at a high-availability, high-bandwidth location, and have a fixed IP address. It must be configured to allow public access to whatever TCP port the *"votmond"*program is set to listen to client connections on, and minimally, access from the IP address of the "main site" system on which the instance of chan_voter is running.

SUMMARY

This technology/project is a great asset to the two-way radio community in general, particularly Amateur Radio and other public-service related radio services, allowing inexpensive, general, and open access to what previously would have been impossible or otherwise unattainable. Systems of this type can greatly improve the ability to provide efficient and reliable communications, not to mention promoting usage of frequencies and modes that otherwise may have been underutilized or ignored.

APPENDIX A - RTCM Setup Console Description

Serial Console access is provided by an RS-232 serial port at 57600 baud, 8 bits, No parity, 1 Stop Bit. It is also made available remotely (off-board) via Telnet. When telnet-ing the default Username is “admin" and the password is "radios". When done with the console session, please use the 'q' command to disconnect. The local serial port and the Telnet (remote) user share the same (and only) console session.

Console Menus (commands):

The Console Menus have (configuration and administration) commands that are separated into the following categories:

Main Menu:

Serial Number (MAC Address),
RTCM Server Address, UDP Ports(for RTCM communications), RTCM authentication info
Tx Buffer Parameters
GPS Device Settings
Telnet authentication info, Telnet port setting
External CTCSS Configuration
Administration:
Set Debug Level
RX Level Display
System Status Info
Quit Remote Telnet Session
Reset (reboot) System

```
VOTER Client System version 0.64  12/30/2011, Jim Dixon WB6NIL
Select the following values to View/Modify:

1  - Serial # (1234) (which is MAC ADDR 0:0:0:0:0:0)
2  - VOTER Server Address (FQDN) (xxxxx.dyndns.org)
3  - VOTER Server Port (667),  4  - Local Port (Override) (0)
5  - Client Password (radios),  6  - Host Password (BLAH)
7  - Tx Buffer Length (1600)
8  - GPS Data Protocol (0=NMEA, 1=TSIP) (0)
9  - GPS Serial Polarity (0=Non-Inverted, 1=Inverted) (0)
10 - GPS PPS Polarity (0=Non-Inverted, 1=Inverted, 2=NONE) (2)
11 - GPS Baud Rate (4800)
12 - External CTCSS (0=Ignore, 1=Non-Inverted, 2=Inverted) (2)
13 - COR Type (0=Normal, 1=IGNORE COR, 2=No Receiver) (1)
14 - Debug Level (0)
97 - RX Level,  98 - Status,  99 - Save Values to EEPROM
i - IP Parameters menu, o - Offline Mode Parameters menu
q - Disconnect Remote Console Session, r - reboot system, d - diagnostics

Enter Selection (1-27,97-99,r,q,d) :
```

IP Parameters Menu:

IP/DNS Configuration
DYNDNS Settings

```
 IP Parameters Menu
Select the following values to View/Modify:

1  - (Static) IP Address (0.0.0.0)
2  - (Static) Netmask (0.0.0.0)
3  - (Static) Gateway (0.0.0.0)
4  - (Static) Primary DNS Server (0.0.0.0)
5  - (Static) Secondary DNS Server (0.0.0.0)
6  - DHCP Enable (1)
7  - Telnet Port (23)
8  - Telnet Username (admin)
9  - Telnet Password (radios)
10 - DynDNS Enable (0)
11 - DynDNS Username (xxxxxxxxx)
12 - DynDNS Password (xxxxxxxxx)
13 - DynDNS Host (xxxxx.dyndns.org)
14 - BootLoader IP Address (192.168.1.11) (OK)
99 - Save Values to EEPROM
x  - Exit IP Parameters Menu (back to main menu)
q  - Disconnect Remote Console Session, r - reboot system
Enter Selection (1-14,99,c,x,q,r) :
```

"Offline Mode" Menu:

CW Speed and timing
CW (Offline/Online) Messages
CTCSS Frequency/Level

```
Offline Mode Parameters Menu
Select the following values to View/Modify:
1  - Offline Mode (0=NONE, 1=Simplex, 2=Simplex w/Trigger, 3=Repeater) (3)
2  - CW Speed (400) (1/8000 secs)
3  - Pre-CW Delay (4000) (1/8000 secs)
4  - Post-CW Delay (4000) (1/8000 secs)
5  - CW "Offline" (ID) String (FAIL)
6  - CW "Online" String (OK)
7  - "Offline" (CW ID) Period Time (3000) (1/10 secs)
8  - Offline Repeat Hang Time (77) (1/10 secs)
9  - Offline CTCSS Tone (000.0) Hz
10 - Offline CTCSS Level (0-32767) (3000)
99 - Save Values to EEPROM
x  - Exit Offline Mode Parameter Menu (back to main menu)
q  - Disconnect Remote Console Session, r - reboot system
Enter Selection (1-9,99,c,x,q,r) :
```

Discussion of Various Parameters/Menu Functions:

Main Menu:

The "Serial Number" (menu item 1), is an unsigned 16 bit number (0-65535) which defines the last 2 bytes of the MAC address of the ethernet interface. Be sure that the MAC address of the device is unique on the LAN to which it is connected.

The "Voter Server Address" (menu item 2) is an ASCII string containing the FQDN of the VOTER server to which to board should communicate.

The "Voter Server Port" (menu item 3) allows specification of the UDP port on which the VOTER server operates. It is usually 667.

The "Local Port Override" (menu item 4) allows the specification of a UDP port number for the device to use locally. If specified as 0, it uses the same port as the "Voter Server Port" (see above).

The "Client Password" (menu item 5) allows the specification of the ASCII password for the client, as configured in the server (for that particular client).

The "Host Password" (menu item 6) allows the specification for the ASCII password the client expects to be receiving from the host.

The "Tx Buffer Length" (menu item 7) is the length, in samples (at 8000 per second), of the Transmit Delay buffer. This (plus the "Tx Buffer Delay", below) specifies the delay between the head end and all the clients on the transmit side. Must be set long enough to allow for worst-case delay and jitter in the Internet path between the client and the head end.

The GPS parameters (menu items 8-11) specify configuration for the GPS device.

The "External CTCSS" (menu item 12) allows specification of the configuration of the External CTCSS input. It may be ignored (value: 0), non-inverted (value: 1), or inverted value:2).
The “COR Type” (menu item 13) allows for connection to receivers that do not provide a connection directly to the discriminator output. In this case, a value of “1” for this parameter (“ignore COR”) would be appropriate. Also, if the RTCM is intended for use with no receiver whatsoever, a value of 2 (“no receiver”) would be appropriate. For normal operation, a value of 0 is to be used.

"Rx Level" (menu item 97) allows quasi-graphical display of the Rx input level.

"Status" (menu item 98) shows various operating parameters.

```
S/W Version: 0.64  12/30/2011
IP Address: 0.0.0.0
Netmask: 0.0.0.0
Gateway: 0.0.0.0
Primary DNS: 0.0.0.0
Secondary DNS: 0.0.0.0
DHCP: 1
VOTER Server IP: 0.0.0.0
VOTER Server UDP Port: 667
OUR UDP Port: 667
GPS Lock: 0
Connected: 1
COR: 1
EXT CTCSS IN: 1
PTT: 0
RSSI: 255
Current Samples / Sec.: 0
Current Peak Audio Level: 160
Squelch Noise Gain Value: 32, Diode Cal. Value: 41, SQL pot 544

Press The Any Key (Enter) To Continue
```

IP Parameters Menu:

The device may operate with DHCP-assigned IP information (menu item 6 set to 1), or Static IP information (menu item 6 set to 0). For Static configurations, menu items 1-5 allow specification of IP configuration information. The "Telnet Port" (menu item 7) specifies the TCP port on which Telnet (remote console) server listens. This is generally 23. It's a good idea to change it to something more unusual.

The Telnet authentication parameters (menu items 8 and 9) allow setting of the remote console (Telnet) user name and password.

The DYNDNS parameters (menu items 10-13) allow configuration of the DYNDNS service, if desired.

The “Bootloader Address” (menu item 14) allows specification of a default IP address for the Bootloader (different then the default address of 192.168.1.11).

"Offline Mode" Menu:

The "Offline Mode" (menu item 1) specifies the mode in which the RTCM operates when off-line:

0 - NONE ("Offline" Mode disabled)
1 - Simplex (Sends "Offline" message when off-line, and periodically while still off-line, if specified)
2 - Simplex w/Trigger (same as "Simplex", but re-triggers the "Offline" sequence when receive un-keys)
3 - Repeater (same as "Simplex", but also acts as local "dumb" repeater while offline)

The "CW Speed" (menu item 2) specifies the length of a CW element in 1/8000 second increments.

The "Pre-CW Delay" (menu item 3) and the "Post-CW Delay" (menu item 4) specify the amount of time amount and time after the PTT is asserted and the CW message starts, and the amount of time the ptt is asserted after the CW message completes.

The "CW "Offline" String" (menu item 5) and "CW "Online" String) menu item 6 specifies the CW messages that are sent when the RTCM detects an "Offline" or "Online" condition.

The ""Offline" (CW ID) Period)" (menu item 7) specifies a period after which the "CW Offline" message will be re-sent if the RTCM is still "Offline".

The "Offline Repeat Hang Time" (menu item 8) specifies the length of the "hang time" if "Offline repeater" mode is enabled.

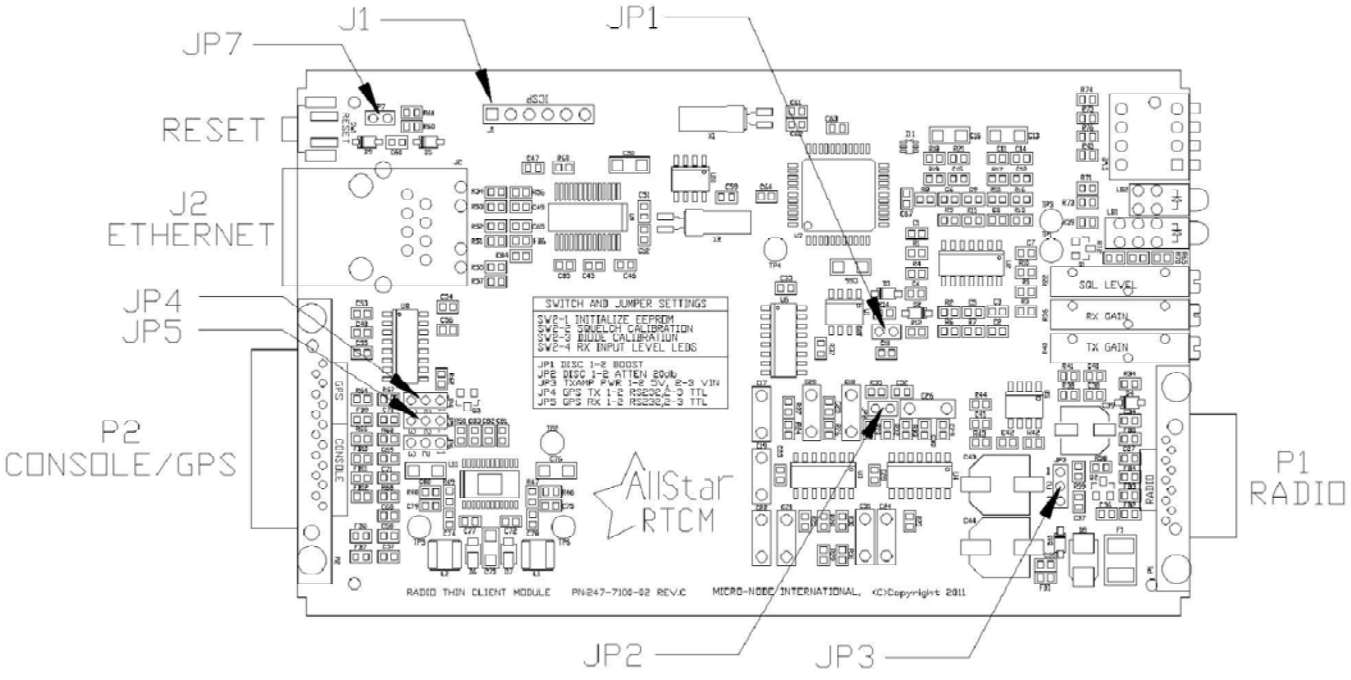
The "Offline CTCSS Tone" (menu item 9) and "Offline CTCSS Level" (menu item 10) specify a locally-generate CTCSS tone that will be added to the transmitted audio when in "offline" mode. Note: this is only applicable and should only be used if the RTCM is directly connected to the transmitter’s modulator. A value of 0 in either (or both) of these parameters disables the "Offline" CTCSS.

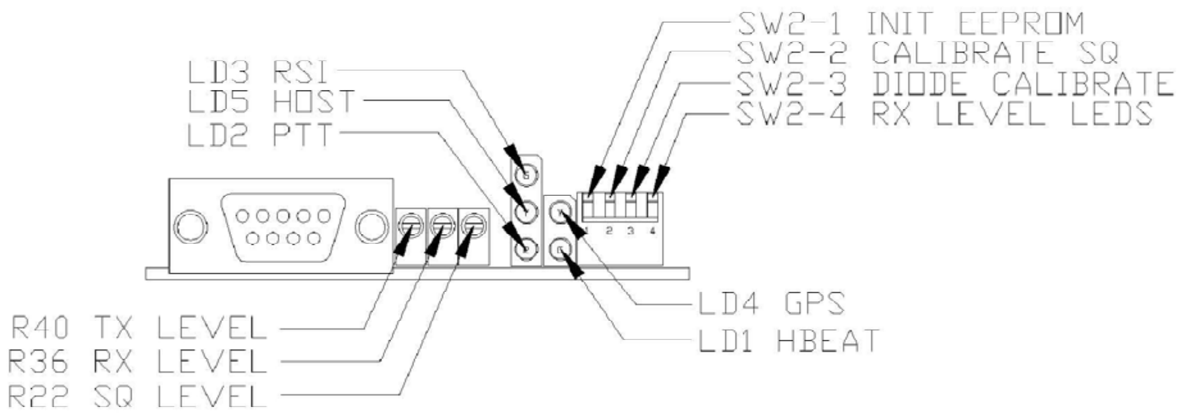
When using "Offline Repeater" mode, the "Offline String" mechanism is intended to be used as a CW identifier. Set the "CW Offline String" (menu item 5) to the repeater's callsign, and set the ""Offline" (CW ID) Period" (menu item 7) to the desired ID period time.

To disable any of the "Offline" functions associated with a numeric parameter, set the parameter to 0. For string parameters, set them to an empty string.

NOTE: The Ethernet must be connected for the firmware to start properly. Also, for GPS/PPS based-operation (normal voting) the GPS must be operational and parameters must be set properly for the rest of the system to work properly.

APPENDIX B - RTCM Hardware Setup Description:





Connectors:

J1 - MTA-100 6 pin connector for PIC ICPS programmer (PICKit3 or equiv).

1 - MCLR
2 - +3.3Vdc
3 - GND
4 - PGD (Program Data)
5 - PGC (Program Clock)
6 – NU

J2 - RJ45 Ethernet (10-Base-T).

P1 - Radio Interface (DB9M D-shell)

1 - + VIn (7-24 Vdc).
2 - Transmit Audio Out
3 - Receive (discriminator) Audio In
4 - External CTCSS Input (optional)
5 - Gnd
6 - Gnd
7 - PTT Out
8 - Gnd
9 - Gnd

P2 - DB15F D-Shell connector for the RS-232 (console @57600 bps, DCE) and GPS

1 - NC
2 - Console Transmit Data
3 - Console Receive Data
4 - NC
5 - Gnd
6 - GPS Receive Data
7 - PTT Out
8 - Gnd
9 - NC
10 - Console Request To Send (RTS)
11 - Console Clear To Send (CTS)
12 - NC
13 - GPS Power Output (5Vdc @ 800ma MAX)
14 - GPS Transmit Data
15 – External Reset

LED designations:

LD1 - (Green) System activity indicator
Should toggle every 500ms
LD2 - (Red) Transmit indicator
On indicates transmitter keyed
LD3 - (Green) Receive Signal indicator
On solid is valid Rx signal, flashing is without CTCSS
LD4 - (Yellow) GPS indicator
On solid is GPS received and locked, flashing is GPS received, lock in progress
LD5 - (Yellow) Host indicator
On indicates that connection was made with RTCM host

\

Switch function descriptions:

SW1 Momentary reset switch

SW2-1 "Initialize configuration parameters in EEPROM". If ON when firmware starts, the operating parameters in the EEPROM will be set to default values. The system activity LED (LD1, green) will stay off for aprox. 4 seconds, then stay on steady to indicate that the initialization process is complete. Afterwards, the switch may be TURNED OFF and the system will continue running normally. Note, if SW2-3 is ON during this procedure, the “Diode Calibration” process will also occur.

SW2-2 On to calibrate squelch. With the receiver connected and its antenna removed, switch on SW2-2. In the next few seconds the "Receive Signal Indicator" (LD3, Green) will flash On and off, then (hopefully) on steady. This indicates that the squelch calibration has occurred successfully. If unsuccessful, the LED will flash either fast to indicate that the discriminator noise level is too high, or slowly to indicate that the discriminator noise level is too low. Note, if SW2-3 is ON during this procedure, the "Diode Calibration" process will also occur.

SW2-3 On to perform "Diode Calibration". This may only be done in conjunction with a configuration parameter initialization (see SW2-1, above), or a "Squelch Calibration" (see SW2-2, above).

SW2-4 On to temporarily re-purpose LD4 and LD5 to allow for visual indication of RX input level (see "Rx Level Calibration", below).

Jumper Descriptions:

JP1 - Insert if low discriminator level. If squelch cannot self-calibrate with JP1 removed (too low), try with JP1 inserted.
JP2 - Insert to attenuate discriminator input level by 20db.
JP3 - Selects power source for output amplifier. 1-2 is to power it from the 5VDC power supply. 2-3 is to power it directly from Vin. DO *NOT* CHANGE THIS JUMPER WITH POWER ON. DAMAGE MAY OCCUR IF YOU DO!!
JP4 - GPS Serial transmit level. 1-2 RS232 Level, 2-3 TTL Level
JP5 - GPS Serial receive level. 1-2 RS232 Level, 2-3 TTL Level
JP6 – Not used
JP7 - Remove for programming.

Potentiometers:

R22 - Squelch adjustment
R36 - Rx Input Level
R40 - Tx Output Level

Squelch Setting:

Once the "Squelch Calibration" procedure has been performed (see SW2-2, above), the squelch adjustment (R22) needs to be properly set. Make sure that the "External CTCSS" is set to "ignore" (value: 0). Adjust R22 until the Receive Signal Indicator (LD3) is lit. Then advance R22 clockwise until LD3 is no longer lit. That is the minimum squelch setting. You probably will want to crank it up at least another turn clockwise, because if you leave it there it will open on a REALLY low level signal. After proper setting has been achieved, return the "External CTCSS" configuration to its original setting.

Rx Input Level Calibration:

Place a full-quieting saturated signal on the receiver modulated by 1000 Hertz sine wave at 3KHz deviation. With SW2-4 on, LD5 will indicate (by brightness) if the RX level is too low, and LD4 will indicate (by brightness) if the RX level is too high. So the idea is to tune R36 so that there is minimal brightness on both LD4 and LD5 (like a null, more or less). Alternatively, the "97" command from the console menu gives a more graphical method of setting the Rx input level.