FT8Call de KN4CRD

2018-09-2 - v0.5.2 - Pre-Release

FT8 has taken over the airwaves as *the* digital communication mode for making QSOs over HF/VHF/UHF. The mode has been widely popular as the latest offering in K1JT's WSJT-X application. FT8 stands on the shoulders of JT65, JT9, and WSPR modes for weak signal communication, but transmits much faster with only slightly reduced sensitivity.

While FT8 is an incredibly robust weak signal mode, it is designed heavily to take advantage of short band openings on HF/VHF/UHF and only offers a minimal QSO framework. However, many operators are using these weak signal qualities to make successful QSOs on the HF bands where other modes fail.

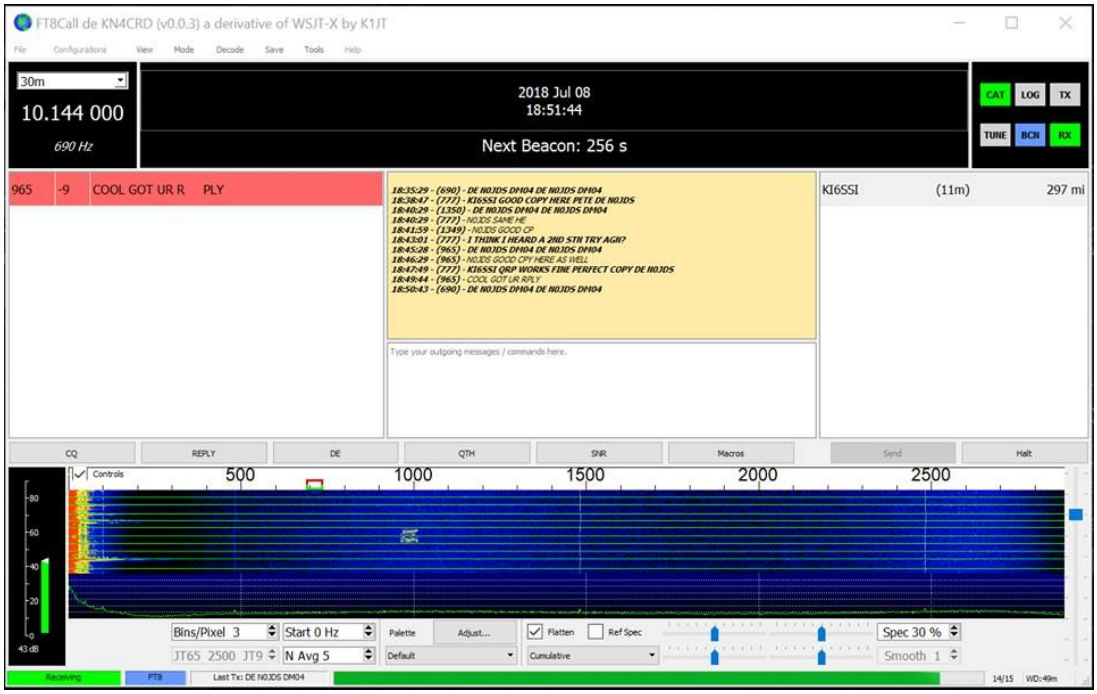
FT8Call is an ***experiment*** to test the feasibility of a digital mode with the robustness of FT8, combined with a messaging and network protocol layer for weak signal *communication* on HF, using keyboard-to-keyboard style interface. FT8Call is heavily inspired by WSJT-X, Fldigi, and FSQCall and would not exist without the hard work and dedication of the many developers in the amateur radio community.

FT8Call stands on the shoulder of giants...the takeoff angle is better up there.

Read more on the design inspiration here: https://github.com/jsherer/ft8call

For release announcements, join the FT8Call mailing list here: https://groups.io/g/ft8call

---



Notice

FT8Call is a derivative of the WSJT-X application, restructured and redesigned for keyboard-to-keyboard message passing. It is not supported by nor endorsed by the WSJT-X development group. While the WSJT-X group maintains copyright over the original work and code, FT8Call is a derivative work licensed under and in accordance with the terms of the GPLv3 license. Source code can be found in this public repository: https://bitbucket.org/widefido/wsjtx/

FT8Call is and will always be **free** software (free as in beer and free as in speech).

Download & Install

FT8Call currently comes in a variety of builds.

- Desktop Linux (64-bit x86_64, Ubuntu 18.04 AppImage) v0.5.2
- Desktop Linux (64-bit x86_64, deb) v0.5.2
- Desktop Linux (32-bit i386) v0.5.2: [not yet available]
- Raspbian Stretch (armv7, AppImage) v0.5.2
- Raspbian Stretch (armv7, deb) v0.5.2
- Windows 10 (win32_64) v0.5.2
    - Windows 10 is the only officially supported Windows build at this time, even though the application works all the way back to Windows XP.
- Mac OSX (x86_64) v0.5.2

For the most up-to-date download links, check:

- FT8Call Release Announcements
- FT8Call Release Download Links

Of course, you are always free to take a look at the source code as well!

The application is distributed using the WSJT-X installer on Windows. For Linux, it is distributed as an AppImage, a single file executable that can be run portably on your Linux desktop or RaspberryPi, as well as a .deb. AppImage is an untraditional approach to distributing Linux software so if you've never dealt with an AppImage before, all you need to do is:

1. Download the AppImage for your platform
2. Set the AppImage file to be executable
3. Run the AppImage file

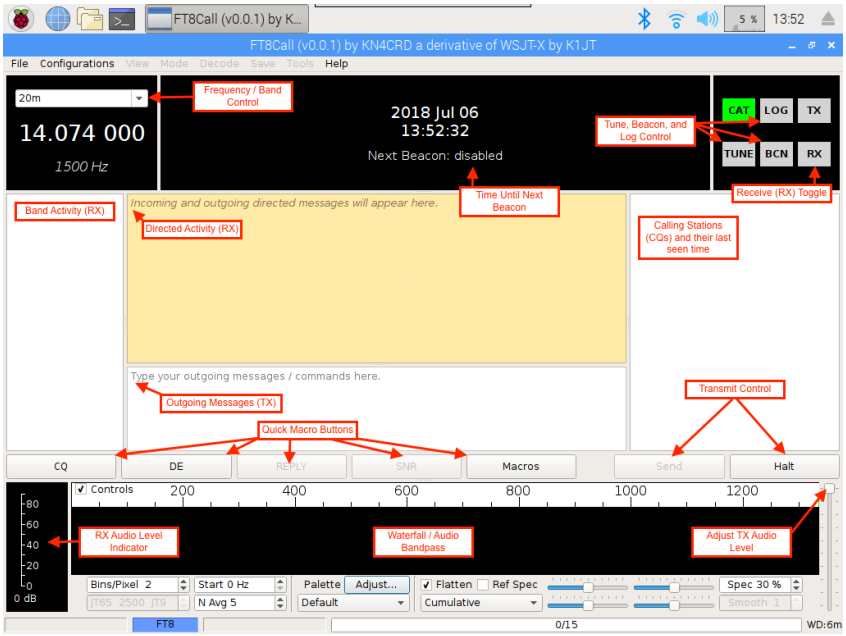
For more information on running AppImages, please check out:
https://docs.appimage.org/user-guide/run-appimages.html

NOTE: Until the general release of FT8Call, development versions will only carry a 10 day lifespan. After expiration, you'll be required to upgrade to the latest version of the application.

Operating FT8Call

If you've used Fldigi or WSJT-X before, you'll feel right at home with FT8Call. The premise is that FT8Call changes the encoding structure of FT8 modulated messages, breaking up long freetext messages into multiple back to back transmission frames / cycles.

Here's what you'll see when you start up the application:



Clock Sync / Timing

In the application you can see the current time reported by your PC in UTC format. An accurate clock is important with FT8Call, as the decoder operates within a 15-second window of transmission (frames). Your clock being off greater than 2 seconds from UTC can cause messages to not decode at your station. It is best to use an Internet, NTP or GPS time source for synchronizing your clock as accurately as possible.

Band Activity

Band activity is displayed on the left. Callsigns you've heard CQing are on the right. Clicking either of those will move your RX/TX offset to that audio frequency (QSY).

There is a waterfall at the bottom of the screen to show you the signals in your audio passband.

Messages

The top yellow text box shows you messages that are either on the frequency offset you're on or who have directed a message to you (they sent a message that included your callsign).

You type into the white box on the bottom to prepare a message for transmission. Normal FT8 character restrictions **do not** apply! The extended character set includes all printable uppercase ASCII (A-Z 0-9 Space

./?+-`~!@#$%^&*()_=[]\{}|;':",<>). The message structure is variable encoded, so the most common characters take the least amount of space, and special characters take longer to send.

As you type your message you'll see the send button display the number of frames (15 second transmit cycles) it'll take to send your complete message. All you have to do is click send (or hit enter) to start transmitting on the next interval. As each frame is transmitted one after the other, the button will update with the number of frames left.

Because of this special variable encoding, messages in FT8Call cannot be decoded by WSJT-X. The same is also true, WSJT-X messages will not be shown in FT8Call.

Messages come in two forms:

1. standard FT8Call free text messages
2. directed FT8Call messages

Standard Messages

Standard messages are freetext messages that do not start with a callsign or a directed command. These messages will only print at another's station location if they align their receive offset within 10Hz of your transmit offset. This is operation similar to other keyboard-to-keyboard digital modes, like Olivia, RTTY, and PSK.

Directed Messages

Directed messages are special FT8Call transmissions that automatically prefix your message with your callsign, similar to how FSQCall operates. Directed messages are useful for communicating in that you do not have to include your callsign in your message, allowing you to use more of the transmission frame(s) for actual message text, as well as alerting the recipient that a message was sent to them. As long as you are in the same passband, you do not have to be on the same frequency offset to receive a directed message.

To send a directed message, all you need to do is include the callsign of the receiving station as the first word in the message. There is a special "ALLCALL" callsign that you can use to send the message to anybody who is able to receive your message. Some examples:

- DR4CNK HELLO HOW ARE YOU JIM?
    - Will be sent as: **KN4CRD: DR4CNK HELLO HOW ARE YOU JIM?** ⌁
- ALLCALL HELLO NET PSE QSY 14300
    - Will be sent as: **KN4CRD: ALLCALL HELLO NET PSE QSY 14300** ⌁

You'll notice a special character at the end of the message, ala ⌁. It is a symbol to indicate the End of Transmission, the last frame of the message has been transmitted with nothing else to follow. This means you get a visual indicator that the transmission is done and you can begin transmitting a reply.

There are special directed messages that you can send to stations to have them automatically reply if they have AUTO enabled. They are comprised in the form of [CALLSIGN][COMMAND].

Available commands:

- ? - What is my SNR?
- @ - What is your QTH (station location)?
- & - What is your QTC (station message)?
- % - What is your station power?
- $ - What stations are you hearing? (Will transmit the top 4 ranked by SNR)
- QSO [CALLSIGN]? - Can you communicate directly with CALLSIGN?
- |message - Please ACK and retransmit the following message
    - The message is retransmitted by the receiving station verbatim with the addition of "DE [CALLSIGN]" added to the end of the message...meaning you do not need to add it to your message.
- !message - Please display this message in a alert dialog and ACK if acknowledged
- #message - Please ACK if you receive this message in its entirety
- AGN? - Have the station automatically retransmit their last message
- ---
- QTC - Send station message
- QTH - Send location message
- GRID - Send a long-form grid locator (to be spotted)
- ---
- QSL? - Did you receive my last transmission?
- QSL - I received your last transmission
- YES - I confirm your last inquiry
- NO - I negative confirm your last inquiry
- HW CPY? - How do you copy?
- RR - Roger. Received. I copy.
- FB - Fine Business
- 73 - I send my Best Regards / End of Contact

Examples:

If we wanted to ask DR4CNK what his QTH was, we'd send:

- DR4CNK@
    - And he would respond with a directed message back to me: "DR4CNK: KN4CRD MY QTH IS SOUTH OF FRANCE", automatically if AUTO reply is enabled.

You can also use "ALLCALL" with the "?" command and all stations that receive your message will respond on a random frequency offset, example:

- ALLCALL?
    - And all stations who heard your message would respond with your signal strength: "DR4CNK: KN4CRD SNR +21"

If we wanted to transmit a message to OH8STN through DR4CNK, we could use the retransmit command and send:

- DR4CNK|OH8STN HELLO JULIAN!
    - During retransmit, at each hop the originating sender's call is appended to the message.
    - The command above would be received by OH8STN, they would send an ACK back, then retransmit the message, like so:
        - KN4 station sends:

            KN4CRD: DR4CNK|OH8STN HELLO JULIAN!
        - DR4 station acks and retransmits:

            DR4CNK: KN4CRD ACK

            DR4CNK: OH8STN HELLO JULIAN! DE KN4CRD

If we wanted to alert a message to LB9YH through OH8STN through DR4CNK, we could send:

- DR4CNK|OH8STN|LB9YH!HEY KEN!

You can also mix and match standard and free text messages, but most of the time you won't need to.

BCN - Beacons

There is an automated beacon that transmits on an interval once turned on (BCN button on the top right). This interval can be changed in the settings. There's no protection against the beacon transmitting over a message you're receiving, so you'll have to keep an eye on it. All beacons are transmitted on your current frequency unless it is not free, in which case a random (unused) frequency free offset between 250Hz-1500Hz.

The intent of beaconing is not to report on propagation...it's to help populate your heard list (on the right) so you know who's likely to be reachable so you can try to make contact. You can't work them if you can't "hear" them.

AUTO - Automatic Replies

While AUTO is enabled, the software will automatically respond to directed queries, like "?", "@", and "&". When AUTO is turned off, FT8Call will buffer responses to directed queries in the send message textbox until you are ready to send the replies manually. Responses to ALLCALL queries are not buffered while AUTO is turned off.

LOG - Station Log

There's a log button in the top right. The software will do its best effort to pre-populate log fields. However, you'll likely have to fill out some missing information manually since the QSO is freetext and not automated. The log is stored in ft8call.log & ft8call.adif in the log directory (which you can find by clicking "Open log directory" in the file menu).

SPOT - Callsign Spotting

FT8Call will report callsigns you hear (or your callsign if heard by other stations) to PSKReporter under the “FT8CALL” mode.

MACRO - Quick Macro Text

There are a few quick macro buttons for entering common messages. You can edit these in the settings window. Just be mindful that long macros will take a while to send.

Frequencies

Most operators testing the application can be found +/- 4-8kHz from the standard FT8 frequencies. It is essential to avoid the main FT8 frequencies, as that will cause confusion among WSJT-X operators. Here are some suggested frequencies to use:

- 160m: 1.842 MHz // 2kHz above FT8
- 80m: 3.578 MHz // 5kHz above FT8
- 40m: 7.078 MHz // 4kHz above FT8
- 30m: 10.130 MHz // 6kHz below FT8
- 20m: 14.078 MHz // 4kHz above FT8
- 17m: 18.104 MHz // 4kHz above FT8
- 15m: 21.078 MHz // 4kHz above FT8
- 12m: 24.922 MHz // 9kHz above FT8
- 10m: 28.078 MHz // 4kHz above FT8
- 6m: 50.318 MHz // 5kHz above FT8

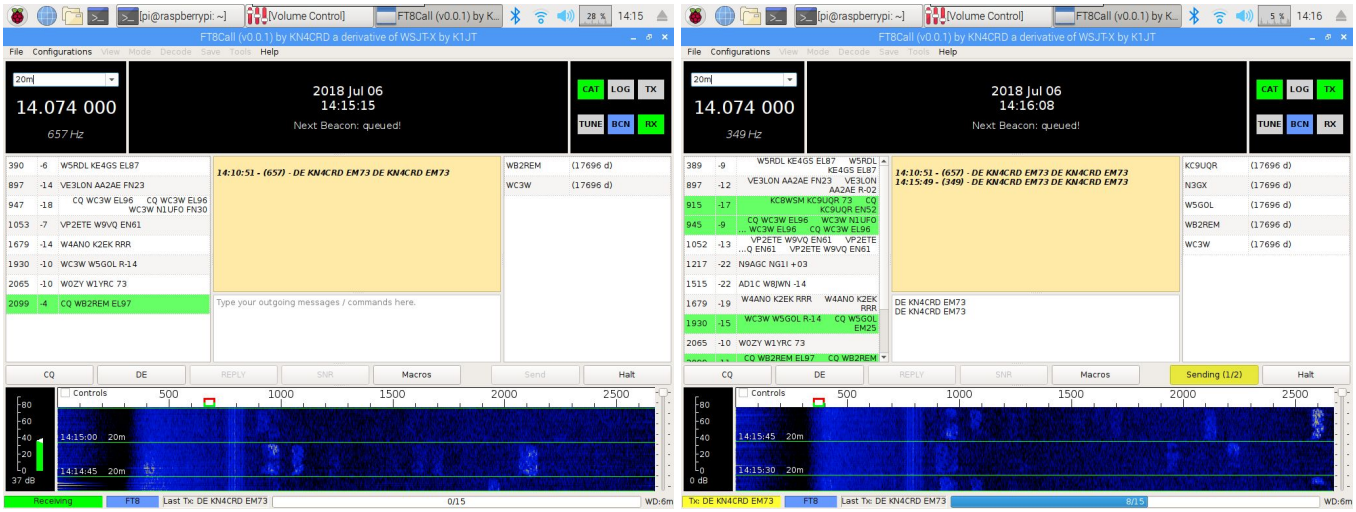
You might notice a few of these being close to the JT9 frequencies. Don’t grab your pitchforks! FT8Call blocks out the lower 500Hz of the passband, which is enough room for 25 simultaneous JT9 signals.

But also, please keep in mind these are only suggested frequencies and **are subject to change while in development**. We all have VFOs, so please use them. Just remember to be good operators and prevent from interfering with other signals on our shared bands.

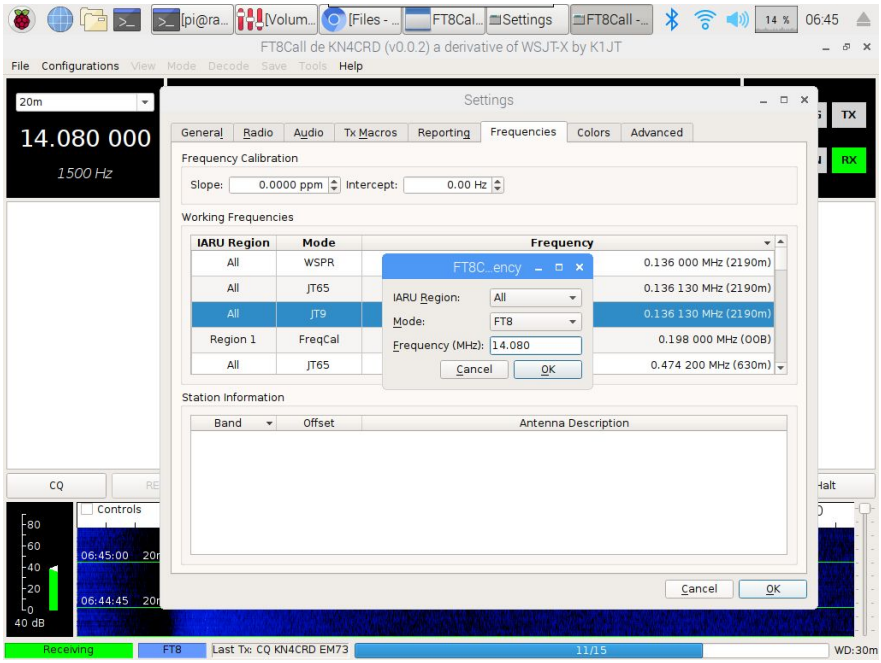
You **CAN** type into the frequency dropdown list to change to a different frequency. FT8Call will not limit which frequencies you can transmit on.

You can use the mailing list Sked Chat or the Facebook group to schedule on other frequencies with test operators.

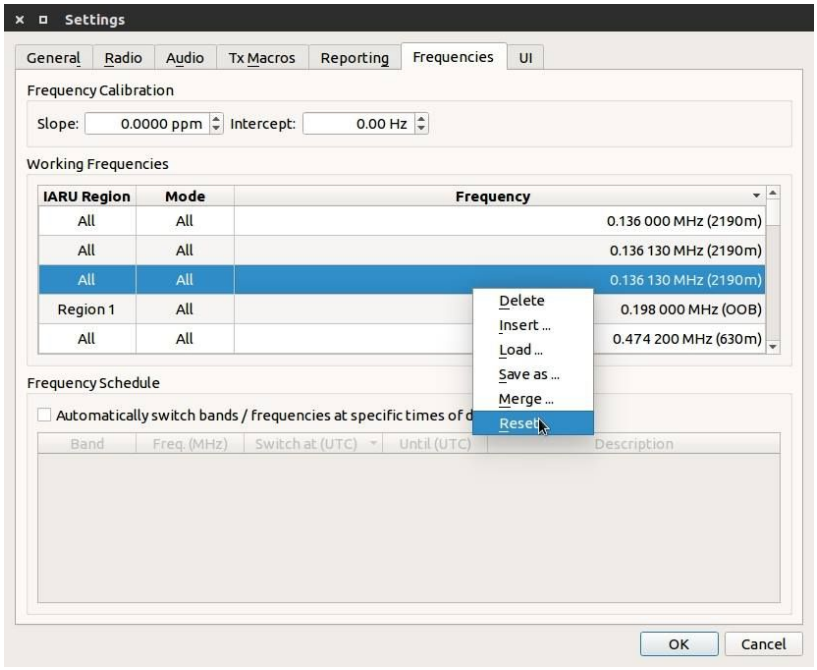
If you want to transmit on a non-standard frequency (recommended) you can either modify the frequencies list in the settings, or you can type directly into the band dropdown box in the top left of the screen.



If you'd like to add custom frequencies for FT8Call, you can do so in the settings:



If you'd like to reset to the suggested frequencies, right click the frequencies box and click Reset.


Settings
General
Radio
Audio
Tx Macros
Reporting
Frequencies
UI
Frequency Calibration
Slope: 0.0000 ppm
Intercept: 0.00 Hz
Working Frequencies
IARU Region
Mode
Frequency
All All 0.136 000 MHz (2190m)
All All 0.136 130 MHz (2190m)
All All 0.136 130 MHz (2190m)
Region 1 All 0.198 000 MHz (OOB)
All All 0.474 200 MHz (630m)
Delete
Insert ...
Load ...
Save as ...
Merge ...
Reset
Frequency Schedule
Automatically switch bands / frequencies at specific times of d
Band
Freq (MHz)
Switch at (UTC)
Until (UTC)
Description
OK
Cancel

Tips & Tricks

- An example QSO:
  - →**KN4CRD: CQCQCQ EM73**
  - ←DR4CNK: KN4CRD SNR +01 GOOD SIGNAL
  - →**KN4CRD: DR4CNK SNR -12 TU 4 CALL RIG IS KX2 5W DIPOLE**
  - ←*DR4CNK: KN4CRD RR -22 FB KX3 100W VERT*
  - →**KN4CRD: DR4CNK RR FB REALLY ENJOYING THE CHAT MODE WITH LONG MESSAGES. BUT HEY LET'S TRY A RELAY**
  - *… (and on, and on, if you want)*
  - →**KN4CRD: DR4CNK 73**
  - →**KN4CRD: CQCQCQ EM73**
- You do not need to include your callsign when initiating your directed replies. They will be prefixed to your message automatically.
- You do not have to reply on the same frequency offset as the caller. But, if you're calling another station off their frequency, you do need to include their callsign at the beginning of the the message so it is directed to them and will show up in their yellow directed activity window.
- Directed messages pack as much data as standard FT8 frames. The following examples are all 1 transmit cycle long
  - Example:
    - KN4CRD/P: CQCQCQ EM73 (1 transmit frame)
    - KN4CRD: ALLCALL? (1 transmit frame)
    - DR4CNK: KN4CRD SNR +15 (1 transmit frame)
    - DR4CNK: KN4CRD AGN? (1 transmit frame)
- For replying to a station's CQ, double click their call in the call activity window, then either choose a directed command or type a message to them:
  - DR4CNK: KN4CRD HW CPY?
  - DR4CNK: KN4CRD SNR +12
  - DR4CNK: KN4CRD YES
  - DR4CNK: KN4CRD NO
  - DR4CNK: KN4CRD RR
  - DR4CNK: KN4CRD 73
  - DR4CNK: KN4CRD HELLO MY FRIEND GREAT TO HEAR YOU!
- You can send freetext at any time! That's what FT8Call was designed for:
  - HI JIM TU 4 CALL UR -12 INTO ATLANTA BTU DE KN4CRD (4 transmit frames)
- It might be helpful to learn some of the morse code prosign/abbreviations and psk31 abbreviations:
  - https://en.wikipedia.org/wiki/Prosigns_for_Morse_code
  - http://www.hamblog.co.uk/common-psk31-abbreviations/
  - Examples:
    - K - over
    - BTU - back to you
    - WX - weather
    - FB - fine business
    - HW? - how do you copy?

- Etc

Frequently Asked Questions

- What are the random three (or six) characters at the end of retransmit and alert commands?
    - These are a checksum for the message added to ensure all of the message frames were delivered correctly before retransmitting / alerting. If received in its entirety by the receiving station, these checksums will not be displayed to them.
- Beacons transmit back to back (30 seconds). Why? Is this normal?
    - Yes. They currently transmit for 30 seconds to compensate for band fading, qrn, or stations transmitting over each other.
    - In the future, beacons that transmit for 30 seconds may also be able to offer extra information during the beacon (things like compound callsigns, extra long grid locators for telemetry, station power, etc)
- You said that all printable ASCII characters can be used. Do some take more time to send than others?
    - Yes. The characters that are sent in the messages are variable encoded, ranging from 3 to 19 bits in length based on their probability of being used in a sentence. The most common characters take the least amount of space, allowing us to send more than 13 characters per transmission cycle on average.
    - Example: Space and E are only 3 bits in length. You could send about 23 (!!) of them in a single transmission. Whereas a character like { is 14 bits in length, you could only send 4 of those. (But really, how frequently do you use that character?)
    - Here are some examples of phrases that could be sent in one 15 second transmit cycle:
        - EEEEEEEEEEEEEEEEEEEEEE (23 characters)
        - I HAVE EATEN A SHOE (19 characters)
        - WHICH WAY TO OHIO (17 characters)
        - NEVER HAVE I EVER (17 characters)
        - TU UR 599 INTO (14 characters)
        - Etc
- How fast does FT8Call transmit?
    - FT8Call uses the same 15-second transmission cycle as FT8. What is different is that due to the variable encoding of the characters, FT8Call can transmit up to 23 characters per transmission frame. For average sentences, FT8Call can pack words very tightly, at around 10-15WPM.
    - Example:
        - "WE HOLD THESE TRUTHS TO BE SELF-EVIDENT THAT ALL MEN ARE CREATED EQUAL THAT THEY ARE ENDOWED BY THEIR CREATOR WITH CERTAIN UNALIENABLE RIGHTS THAT AMONG THESE ARE LIFE LIBERTY AND THE PURSUIT OF HAPPINESS."
            - This phrase is 35 words. It would take 13 transmission cycles to send (3 minutes 15 seconds). That is just under 11 WPM.
        - “A SUCCESSFUL MAN IS ONE WHO CAN LAY A FIRM FOUNDATION WITH THE BRICKS OTHERS HAVE THROWN AT HIM”
            - This phrase is 19 words. It would take 6 transmission cycles to send (1 minute 30 seconds). That is just over 12.5WPM.
        - CW has a neat way of calculating WPM, timing how long it takes to transmit the word PARIS. In FT8Call, PARIS is encoded into 24 bits (4.8 bits/character). Each transmission cycle can pack up to 69 character bits. That equates to about 11.5 WPM. (69/24=2.875 words / 15seconds * 4)
- Isn’t 10-15 WPM too slow to have a conversation?
    - If propagation is good enough for a faster mode, you should be using it instead! But, with poor conditions like we have experienced at solar minimum, FT8Call might just be the best balance.

    - It may seem really slow (and it is, relatively speaking). However, FT8 modulation is able to decode (theoretically) down to -24dB below the noise floor. Not many modes can say this, especially at faster speeds. What does this mean? FT8Call may work when other modes cannot. Receiving messages slowly is better than not receiving them at all.
- What is the FT8Call Relay Challenge?
    - This is a friendly competition to maximize the number of continents one can pass a message back and forth to using the retransmit command.
    - We'll be giving away an award (and prize) to the first team of operators to successfully relay a message from one continent across two other continents (NA, SA, EU, AF, AS, OC, AN) and relay an ACK back to the original station using FT8Call. All you need to do is submit your logs from each station and optionally photographic/video documentation of your effort.
    - For example, this is what the outgoing and incoming messages could be:
        - LB9YH|KN4CRD|VK1MIC QSL?
        - VK1MIC|KN4CRD|LB9YH QSL
- How do I submit logs for FT8Call?
    - Currently, the logging function in FT8Call will log each contact under the FT8CALL mode. We have not petitioned ADIF yet for inclusion in the mode tables, but will plan to do so before the general release.
    - In the meantime, you can either submit the mode as FT8 (with a submode of FT8CALL) or as DATA.
    - e.g., <MODE:3>FT8 <SUBMODE:7>FT8CALL
    - e.g., <MODE:4>DATA

Bug Reports

You can send bug reports to Jordan Sherer (KN4CRD) at kn4crd@gmail.com or in the troubleshooting chatroom in the groups.io page: https://groups.io/g/ft8call/chat/1423

Inside Jokes

Many of the testers have come to use the term “Baconing” instead of “Beaconing”.

API Definition

FT8Call uses a JSONRPC API offered through UDP. It is currently highly experimental and subject to drastic change in the future (like, for instance, moving to a HTTP or XMLRPC implementation instead).

Once released, the API will allow you to:

- `RIG.GET_FREQ - Get the current Frequency`
- `RIG.SET_FREQ - Set the current Frequency`

- `STATION.GET_CALLSIGN - Get the current callsign`
- `STATION.GET_GRID - Get the current grid locator`
- `STATION.SET_GRID - Set the current grid locator`
- `STATION.GET_QTC - Get the current station message`
- `STATION.SET_QTC - Set the current station message`

- `RX.GET_CALL_ACTIVITY - Get the current heard list`
- `RX.GET_BAND_ACTIVITY - Get the current activity in the band activity list`
- `RX.GET_TEXT - Get the text from the yellow rx box`

- `TX.GET_TEXT - Get the text in the tx box`
- `TX.SET_TEXT - Set the text in the tx box`
- `TX.SEND_MESSAGE - Send a message`

- `LOG.QSO - QSO has been added to the FT8Call log`

- `WINDOW.RAISE - Bring the window to the foreground`

Example implementations will be provided at a future date.

Technical Implementation

FT8Call is under active development and details about the technical implementation are subject to change. Detail will be added here as the implementation stabilizes. Until then, the code is the source of truth for the implementation.

Modulation

FT8Call uses FT8 modulation as the base transport for data. Being a derivative of WSJT-X, FT8Call heavily leverages the work by the WSJT-X Development Group on the FT8 mode. Very few modifications have been made (see source code for the exhaustive list) to the base FT8 modulation, with the exception of two important items:

1. Modifying the CRC algorithm to prevent FT8Call from interfering with FT8 signals
2. And allowing all 75 bits to be utilized for data transport

Protocol

The FT8Call protocol sits at a layer above the base transport. Much of the implementation is inspired by the design document: https://github.com/jsherer/ft8call with a few deviations from the original proposal.

Messages in FT8Call are transmitted in 15-second intervals (frames), with each frame being classified as one of 6 types:

1. Beacon
2. Compound Callsign Partial
3. Compound Callsign Directed Command
4. Directed Command
5. Data Padded
6. Data Unpadded

Further, each frame includes a flag identifying it as:

1. Default Frame (any frame)
2. First Frame (first frame of the transmission)
3. Last Frame (last frame of the transmission)
4. Reserved (for future use)

And finally, there are special encodings for:

1. Callsigns
2. Callsign Prefix/Suffixes
3. Signal Reports
4. Power Levels
5. Grids

Beacon

Beacon frames are comprised of:

- Beacon Type (BEACON or CQ)
- Compound Callsign
- Grid

Compound Callsign Partial

Compound callsign partials are used as one-half of a 2-frame compound transmission when one of the stations includes a compound callsign. Compound callsign partials are always the 1st frame in a 2-frame compound transmission, encoding the “from” portion of a directed command with compound callsigns.

The frame includes:

- Callsign
- 4 character alphanumeric prefix or suffix (A-Z 0-9)
- Grid or Numeric Value (SNR or PWR)

Compound Callsign Directed Command

Compound callsign directed commands are a special case for compound callsign partials where the numeric value encodes a directed command to be used with a compound directed message. It is one-half of a 2-frame compound transmission. Compound callsign directed commands are always the 2nd frame in a 2-frame compound transmission, encoding the “to” portion of a directed command with compound callsigns.

The frame includes:

- Callsign
- 4 character alphanumeric prefix or suffix (A-Z 0-9)
- Directed Command

Directed Command

Standard callsigns can send a directed command in one frame.

The frame includes:

- From Callsign
- To Callsign
- Directed Command
- Numeric Value

Data

Data frames are the backbone for long-form messages in FT8Call. They are 75-bit frames that use a variable encoding to pack character data into the smallest transmission possible.

Data frames come in two flavors:

- Unpadded: All bits are used for character data, with no pad bits at the end.
- Padded: The character data is less than the frame size, so pad bits are added. The pad bits are a series of 1-bits up until the first 0-bit. The rest of the bits from the beginning of the frame to that point are data.

Data frames may need to include pad bits because of the variable encoding that character data uses for packing. The variable encoding used is a modified Huffman code that represents the most common characters (based on their frequency of observation in most texts) in fewer bits than less common characters, with the option to shift in alternate alphabets.

The complete modified Huffman code is located in Appendix A.

Callsigns

Callsigns are encoded in 28-bits as described in: EME 2000 - http://www.ka9q.net/papers/eme-2000.ps.gz

Callsign Prefix / Suffix

Prefixes and suffixes are 4 character alphanumeric encoded in 21-bits with a 1-bit flag to indicate whether or not it is a prefix or suffix. Alphanumeric digits can each be encoded in 5.25 bits (there are only 1,874,161 combinations of 4 character alphanumeric prefix/suffix, which is less than can be represented in a 21-bit number 2^{21} = 2,097,152)

Grids

Grids are encoded in 15-bits as described in:
http://physics.princeton.edu/pulsar/k1jt/wsjtx-doc/wsjtx-main-1.7.0.html#PROTOCOL_OVERVIEW

Appendix A: Huffman Code Table

Character code weighted by frequency

```
3 bits
" "      "000"
"E"      "001"

4 bits
"T"      "1100"
"A"      "1010"
"O"      "0111"
"I"      "0101"
"N"      "0100"

5 bits
"S"      "11111"
"H"      "11110"
"R"      "11101"
"D"      "10111"
"L"      "10110"

6 bits
"C"      "111001"
"U"      "111000"
"M"      "110111"
"W"      "110110"
"F"      "110100"
"G"      "100111"
"Q"      "100110"
"Y"      "011010"
"P"      "011001"
"B"      "011000"

7 bits
"\\"     "0110111"    (this is an escape character)
"."      "1000000"
"0"      "1000001"
"1"      "1000010"
"2"      "1000011"
"3"      "1000100"
"4"      "1000101"
"5"      "1000110"
"6"      "1000111"
"7"      "1001000"
"8"      "1001001"
"9"      "1001010"
"?"      "1001011"
"/"      "1101010"
"V"      "0110110"

8 bits
"K"      "11010111"

10 bits
"J"      "1101011010"
"X"      "1101011001"

11 bits
"Z"      "11010110110"
":"      "11010110000"

12 bits
"+"      "110101100011"
"-"      "110101101110"
"!"      "110101101111"
"\x04"   "110101100010"  (this is the EOT character)
```

character escapes for alternate alphabets, special characters, or trigrams / quadgrams

```
10 bits
"\\E"    ","

11 bits
"\\T"    "&"
"\\A"    "@"
"\\O"    "#"
"\\I"    "$"
"\\N"    "%"

12 bits
"\\S"    "\'"
"\\H"    "\""
"\\R"    "("
"\\D"    ")"
"\\L"    "|"

13 bits - trigram / quadgram efficiency
"\\C"    "YOU"  (16 bits - 3 bit savings)
"\\U"    "THAT" (17 bits - 4 bit savings)
"\\M"    "THER" (17 bits - 4 bit savings)
"\\W"    "WITH" (18 bits - 5 bit savings)
"\\F"    "TION" (16 bits - 3 bit savings)
"\\G"    "HERE" (16 bits - 3 bit savings)
"\\Q"    "OULD" (20 bits - 7 bit savings)
"\\Y"    "IGHT" (19 bits - 6 bit savings)
"\\P"    "HAVE" (19 bits - 6 bit savings)
"\\B"    "HICH" (20 bits - 7 bit savings)

14 bits
"\\1"    "<"
"\\2"    ">"
"\\3"    "["
"\\4"    "]"
"\\5"    "{"
"\\6"    "}"
"\\7"    "*"
"\\8"    "="
"\\9"    ";"
"\\?"    "WHIC" (21 bits - 7 bit savings)
"\\/"    "THIS" (18 bits - 4 bit savings)
"\\V"    "FROM" (21 bits - 7 bit savings)

15 bits - quadgram efficiency
"\\K"    "OUGH" (21 bits - 6 bit savings)

18 bits
"\\Z"    "^"
"\\:"    "~"

19 bits
"\\+"    "`"
"\\-"    "_"

special case :)
"\\!"    "FT8CALL" (37 bits - 18 bit savings)
```