

Pocket FT8: A Palm Size FT 8 Transceiver

Scope

This document is written to help you build your own self contained FT8 Rig. Listed below are the main features of the project.

- Small Size, 3.5" X 2.75" X 1.125"
- 100 mW power output @ 50 ohm load
- 1 uVolt Receiver Sensitivity
- Single 5 volt power input, battery or wall wart
- Silicon Labs Technology, Si4735 SSB Receiver & Si5351 Transmit FSK Clock
- SD Card Contact Logging
- 320 X 480 Resistive Color Touch Screen

Attributions

This project is based on two significant software projects:

Si4735 Library developed by Ricardo Caritti: <https://github.com/pu2clr/SI4735>

FT8 Decoding Library by Karlis Goba: https://github.com/kgoba/ft8_lib

DSP Audio Architecture

Decoding FT8 requires significant data storage and processing speed.

In order to optimize both program storage and processing speed requirements so that the Teensy 3.6 is not over taxed, the Teensy Audio Library has been modified to allow Analog to Digital conversion to be run at the rate of 6400 samples per second. This allows audio data processing to be done at 3200 Hz. The 3200 Hz audio processing with a 2048 FFT to process the received audio for FT8 decoding yields a bin spacing of 3.125 Hz.

The algorithms developed by Karlis Goba use the 3.125 Hz spaced FFT bins to be screened in both frequency and time so that errors in symbol frequency and time reception can be overcome to provide really great FT8 decoding. The end spacing of the FT8 algorithms is 6.25 Hz.



Overall Architecture

The project consists of the following hardware modules:

Teensy 3.6

<https://www.pjrc.com/store/teensy36.html>

Adafruit 320 X 480 TFT Touchscreen

<https://learn.adafruit.com/adafruit-3-5-color-320x480-tft-touchscreen-breakout>

Adafruit Si5351 Clock Breakout Board

<https://www.adafruit.com/product/2045>

Si4735 Radio Receiver IC:

<https://www.semiconductorstore.com/cart/pc/viewPrd.asp?idproduct=51832>

Microchip MCP 3422 Analog to Digital IC

<https://www.microchip.com/en-us/product/MCP3422>

Mini-Circuits GVA84++ Monolithic RF Amplifier

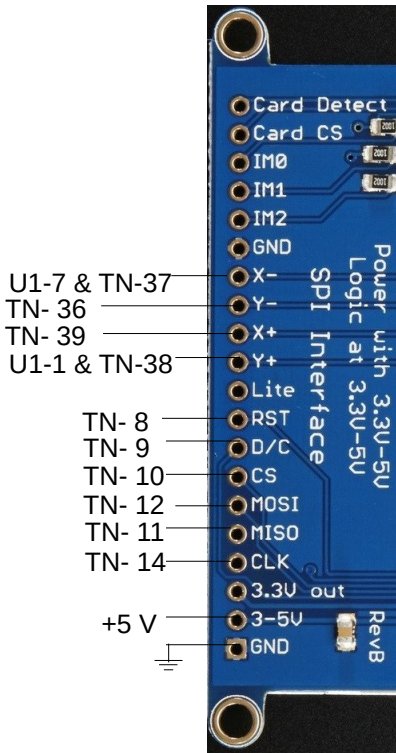
<https://www.minicircuits.com/pdfs/GVA-84+.pdf>

The Teensy 3.6 communicates with the Si5351 Clock, Si4735 Receiver and MCP 3244 A/D via a common I2C Bus and the Touchscreen Display via a SPI Bus.

The Microchip MCP 3422 Analog to Digital IC is used to interface the Touch Screen touch point coordinates to the Teensy processor. One may ask “Why not use two of the Analog Pins on the Teensy?”. An attempt was made to do just this. However, the high rate of A to D sampling required by the Teensy Audio Library swamps out polling of the touch screen analog data. So, the MCP3422 is used so the touch point data may be fed to the Teensy 3.6 via the I2C Bus.

The project connections are shown on two drawings shown below. One drawing shows the connections between the modules listed above and the other shows the low pass filters and PTT control associated with the GVA-84+ transmit RF amplifier. The values for the low pass filers are set for 17 Meter (18.100 mHz) operation.

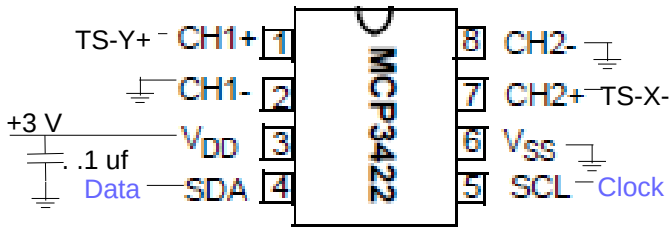
Touch Screen (TS)



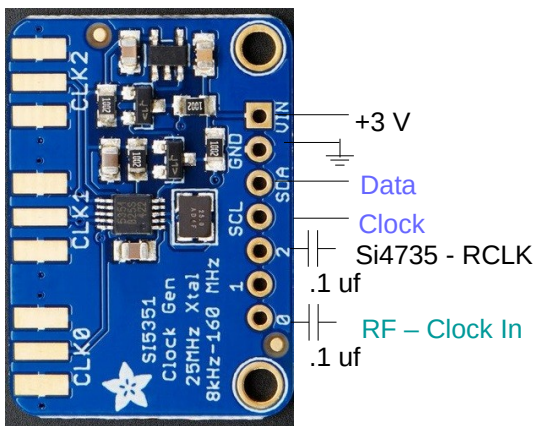
U1-7 & TN-37
TN- 36
TN- 39
U1-1 & TN-38

TN- 8
TN- 9
TN- 10
TN- 12
TN- 11
TN- 14

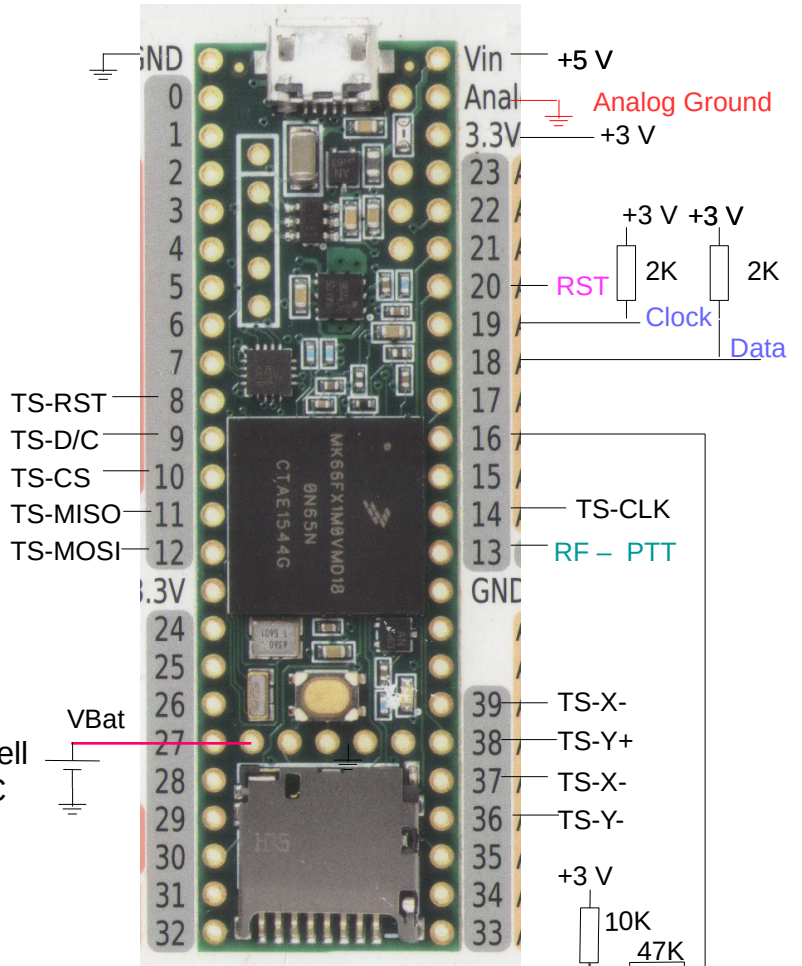
U1
MCP3422
MSOP, SOIC



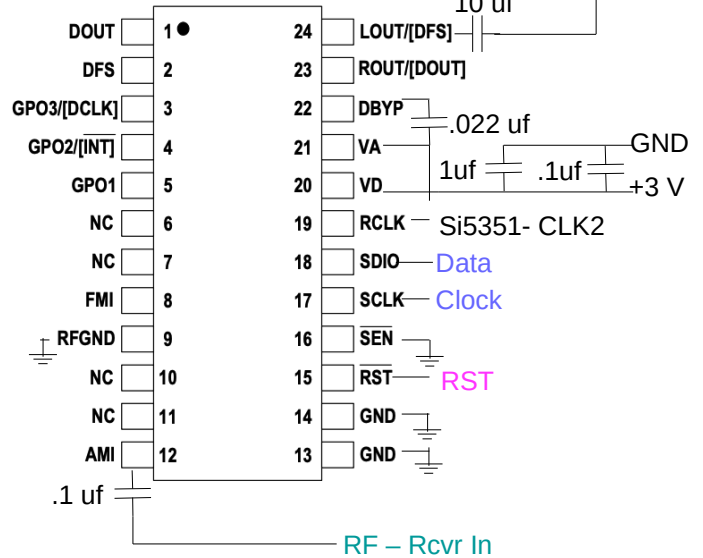
Si5351 Breakout

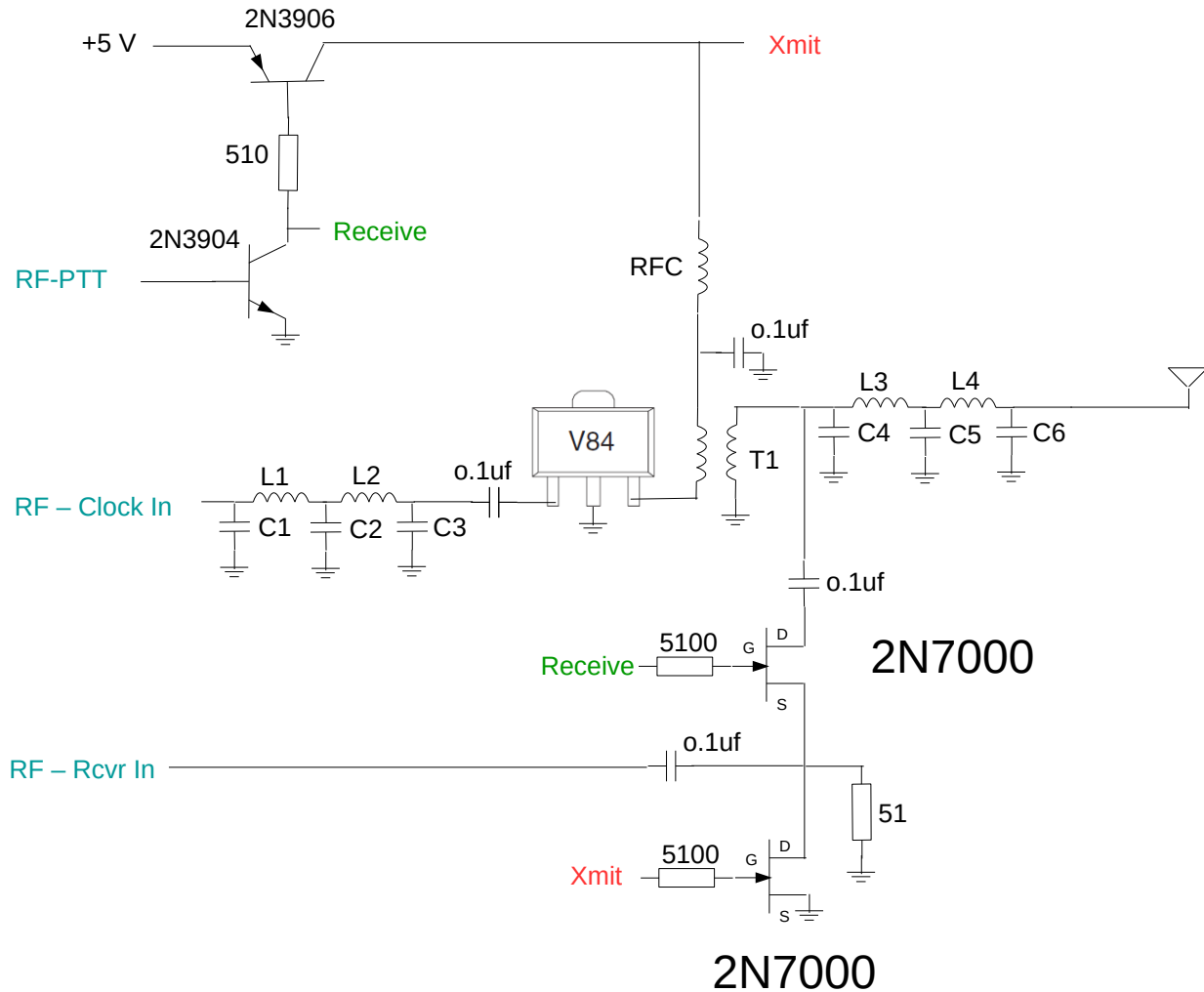


Teensy 3.6 (TN)



U2
Si4735





L1, L2, L3, L4, L5, L6: 0.5 uh 14 turns # 26 on T25-6 core

C1, C3, C4, C6: 75 pf

C2, C5: 243 pf

V84: Mini-Circuits GVA-84+ Monolithic 100 mW Amplifier

RFC: 8 turns # 26 on BN-43-2402 Core

T1: Pri- 2 turns, Sec- 4 turns, #26 on BN-61-2402 Core

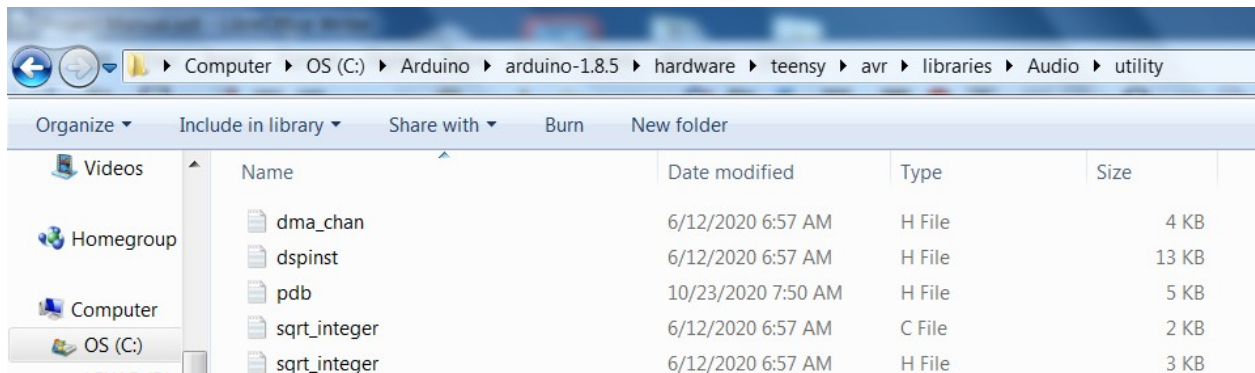
Building the Firmware

The firmware is built using the Arduino IDE along with the Teensy Libraries. Please use this link for instructions on installing the IDE : https://www.pjrc.com/teensy/td_download.html

As stated above, the Teensy Audio Library will need to be modified to support 6400 sample per second A to D sampling. To do this follow this path:

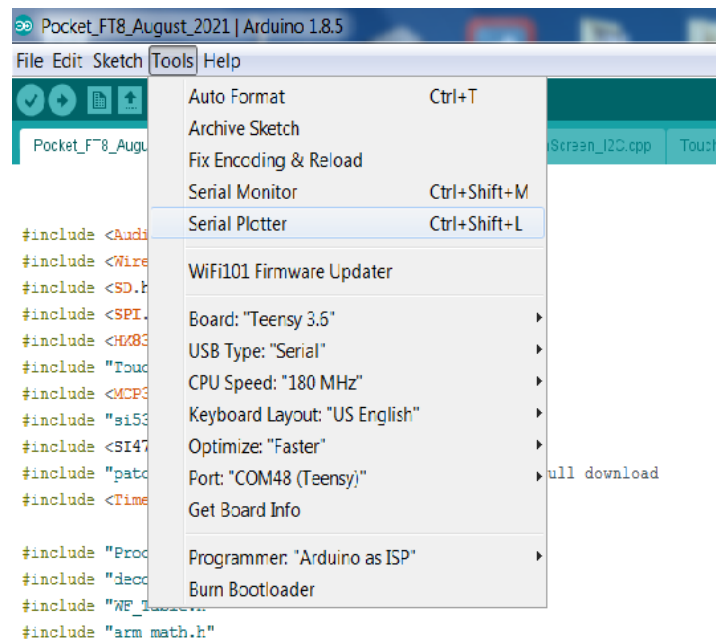
C:\Arduino\arduino-1.8.5\hardware\teensy\avr\libraries\Audio\utility

to the “utility” folder as shown below:



Replace the pdb.h file with the one provided with this project archive.

Shown below is the board selection for the Teensy 3.6 board:



The following libraries are required for the project. These libraries are provided in the Project Driver Files folder:

Etherkit_Si5351
MCP342X

HX8357_t3-master
Si4735-master

Real Time Clock


The Teensy Real Time Clock (RTC) is used to display relative time and to mark log entries with date – timegroups. The RTC is not used to synchronize the reception of FT8 Traffic.

Instead, the software is manually synchronized using the touch screen button marked “Sy”. This technique is copied from the work done by Karlis Goba and is a very effective way to sync with the current FT8 traffic.

A 3 volt coin cell battery is required to be connected to the Vbat pin on the Teensy 3.6. In order to set the date and time of the RTC please use the <https://www.unixtimestamp.com/> to calculate a target sync time along with the Time.pde file included in the project archive to set the Teensy RTC.

Setting Your Call and Location

Before you build your project please modify the Pocket_FT8_August_2021.ino file, Lines 58 & 59 to use your call and your location Maidenhead Locator group. Shown below are Lines 58 & 59 in the .ino file



```
Pocket_FT8_August_2021 | Arduino 1.8.5
File Edit Sketch Tools Help
Pocket_FT8_August_2021 $ Process_DSP.cpp Process_DSP.h TouchScreen_I2C.cpp To
AudioConnection patchCord2(adcl, queue1);

q15_t dsp_buffer[3*input_gulp_size] __attribute__((aligned (4)));
q15_t dsp_output[FFT_SIZE *2] __attribute__((aligned (4)));
q15_t input_gulp[input_gulp_size] __attribute__((aligned (4)));

char Station_Call[] = "UrCall"; //six character call sign + /0
char Locator[] = "UrLo"; // four character locator + /0
```

Setting Transmit Offset

Although both the transmit and receive functions are derived from the same Si5351 there will be an offset between the transmit frequency shown by the Red Line in the spectrum display and the actual transmit frequency of the FT_8 FSK signal. This is the result of the receiver using Clock 2 of the Si5351 and the transmitter using Clock 1 of the Si5351. To provide synchronization of the transmit frequency with the receive frequency a Serial interface has been included in the project firmware.

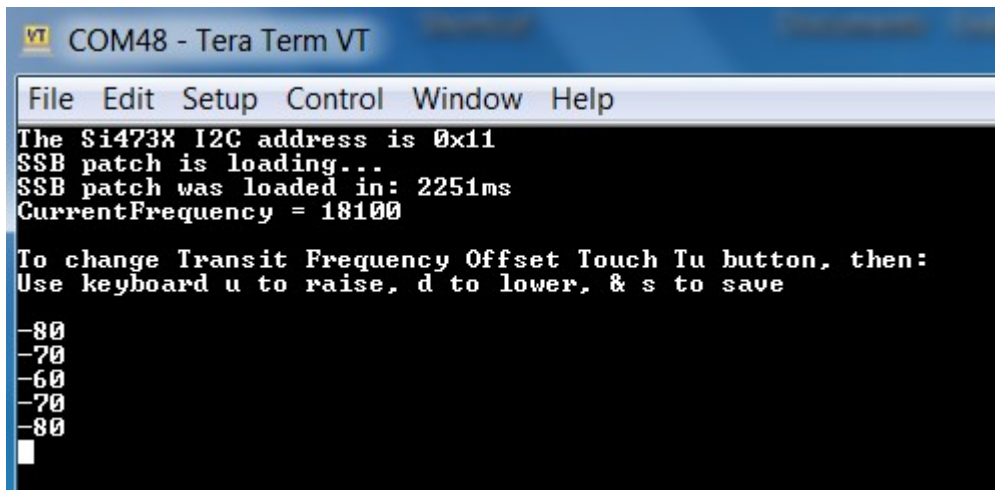
To use this interface install a copy of Tera Term: https://en.wikipedia.org/wiki/Tera_Term

Use Tera Term to establish a Serial connection to the Teensy 3.6 at 9600 Baud using the USB port on the Teensy. This is the same USB port used to program the Teensy. Then reboot the Teensy to see the entire start up message. You will see the instructions for changing the Transmit Frequency Offset in the Tera Term serial monitor window as shown below.

Then touch the Tu button to place the transceiver in the tune mode. Use the “u” key to move the transmitted signal up in frequency and the “d” button to move the transmitted frequency down.

The movement of the transmitted frequency may be viewed in real time on the spectrum display.

When the transmitted frequency is under the red line, use the “s” key to save the modified Transmit Frequency Offset value to the Teensy EEPROM. When you reboot the Teensy, the value stored in the EEPROM is recalled so that the transmitted signal shows up under the red line.



```
COM48 - Tera Term VT
File Edit Setup Control Window Help
The Si473X I2C address is 0x11
SSB patch is loading...
SSB patch was loaded in: 2251ms
CurrentFrequency = 18100

To change Transit Frequency Offset Touch Tu button, then:
Use keyboard u to raise, d to lower, & s to save

-80
-70
-60
-70
-80
█
```

Pocket_FT8 User Operation



The user is presented with a number of data displays, touch buttons and touch areas as shown above.

Display Items

Starting at the top left hand, the received audio / FT8 spectrum is shown. The vertical red line on the spectrum display shows the FT8 transmit frequency.

At the top right hand side, the RTC time, selected station call sign, base FT8 RF Frequency, and RTC date are shown. In the figure above, the selected station call sign is blank since no station has been selected yet.

In the left center pane the received FT8 traffic is shown.

In the right center pane the FT8 messages wherein your station is being called are listed.

Touch Areas

Two touch areas are provided.

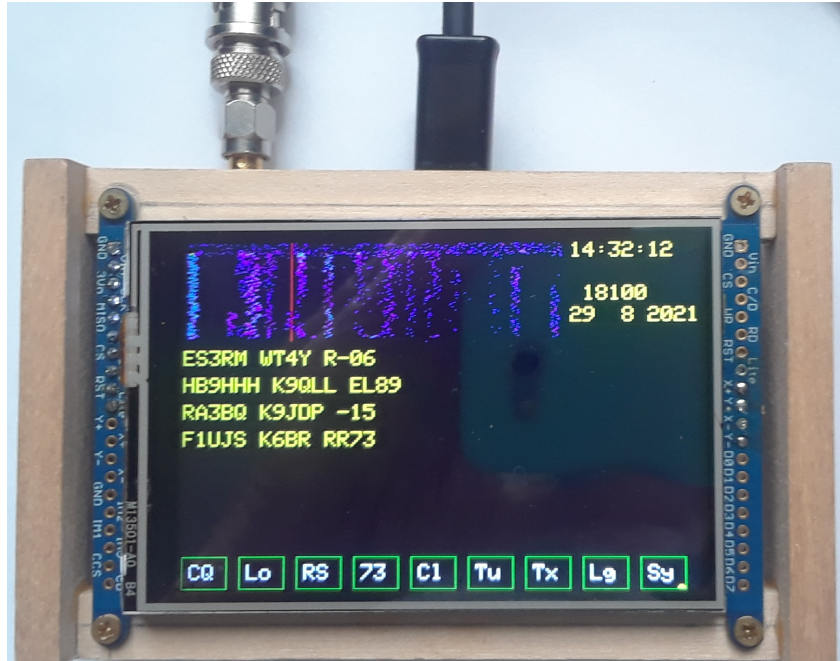
One is the FT8 spectrum. Touching this area moves the FT8 transmit frequency. The red vertical line moves to show the selected FT8 transmit frequency.

The other touch area is the left center pane where the FT8 traffic is listed. You may select one of the FT8 messages so that you may compose a message to the transmitting station. The call of the selected transmitting station is shown in the top right hand display area as described above.

The figure below illustrates the results of touching the message "IZ2KWV NK1I FN43" where the calling station call's "NK1I" is displayed at the top right hand corner right below the displayed RTC time.



Touch Buttons



Nine touch buttons are displayed at the bottom of the screen.

CQ toggles between the beacon (CQing) mode and the QSO mode of operation. Green means the QSO mode is selected. Red means CQ mode is selected.

Lo generates a location message; e.g., W5ITU W5BAA EM00

RS generates a signal strength message; e.g, W5ITU W5BAA -7

73 generates a 73 message; e.g, W5ITU W5BAA RR73

The message composed and queued for transmission is shown in white on the left hand side of the screen right above the touch buttons.

Cl clears the user composed FT8 message described right above.

Tu puts the transceiver in transmit with with a steady tone at the frequency shown by the vertical red line in the spectrum display.

Tx is used in the QSO mode so that the user composed FT8 message is transmitted on the next available FT8 time slot.

Lg is used to toggle the traffic logging on to the SD card on or off. Red means logging is on.

Sy allows the user to synchronize the FT8 receiver with the rest of the FT8 world.

Shown below is the result of touching the Lo button with the selected call of **NK1I**

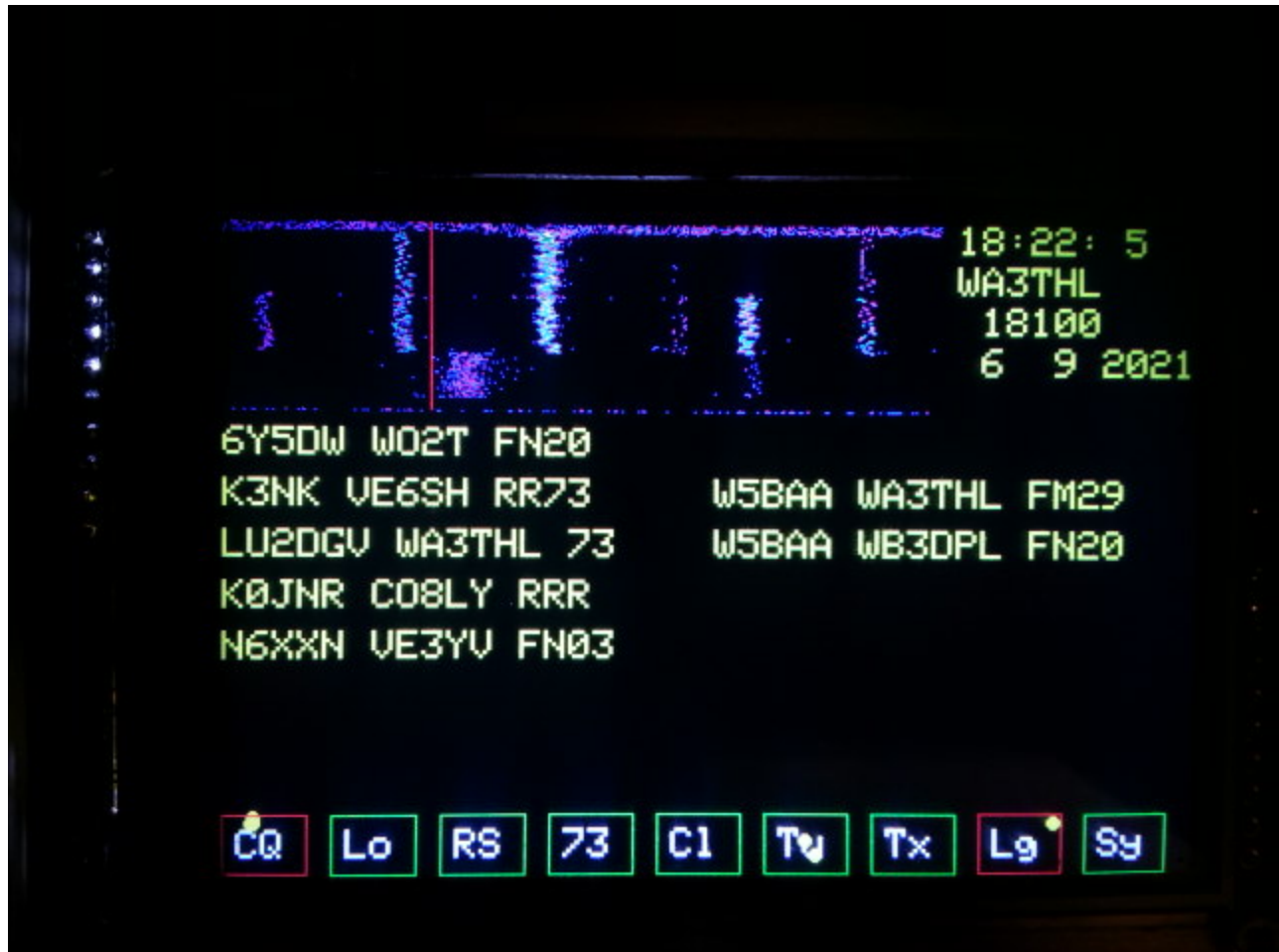


The other two message compose touch buttons, **RS** and **73** operate in the same way.

The FT8 has a traffic manager that works in either a QSO mode or CQ (beacon) mode.

In the QSO mode the user first selects a station to call by touching one of the FT8 traffic messages as described above. Then, by using the Lo, RS, or 73 touch buttons the user may compose the message to send next. Finally, the user must touch the Tx button so that the displayed composed message is transmitted in the next available FT8 time slot.

In the CQ mode the user simply touches the CQ button and the button turns red and then the user may observe the traffic. As stations respond to your CQ call, their FT8 messages are displayed in the center right hand pane as shown below.



Further, if the Lg, Log button, is touched the button turns red and the traffic logging to SD card is turned on. Shown below are the log entries for the CQ message responses are shown above.

6/ 9/2021 17:28:39 W5BAA WB3DPL FN20
6/ 9/2021 18:11:39 W5BAA WA3THL FM29

The messages are logged along with the date and time of the received message. The date and time are taken from the Teensy Real Time Clock (RTC).

That's All Folks!