Fedora 14

Amateur Radio Guide

A guide for users of Fedora amateur radio software

The Fedora Documentation Project

Abstract

Fedora includes a wide range of applications relevant to amateur radio operators. This guide describes the use of some of those applications

1. Introduction

Amateur radio and Linux go hand-in-hand. Both allow users to experiment to the extent of their knowledge and to learn more along the way. With new digital technologies being used everyday open source software is the best way to stay cutting e dge in this ever-changing hobby.

Fedora has packaged dozens of software to make it easy for Fedora users to obtain and setup. Within seconds any user w ill be able to have the tools they to enhance their amateur radio experience.

2. Sound Card Modes

This section describes the sound card modes.

2.1. fldigi

fldigi is one of the most robust soundcard modem software around. It supports CW, DominioEX, Feld-Hell, MFSK, MT-63 PSK, OLIVIA, RTTY, Thor, and Throb. fldigi also includes, as part of the software, a logger which is smart enough to almost complete itself.

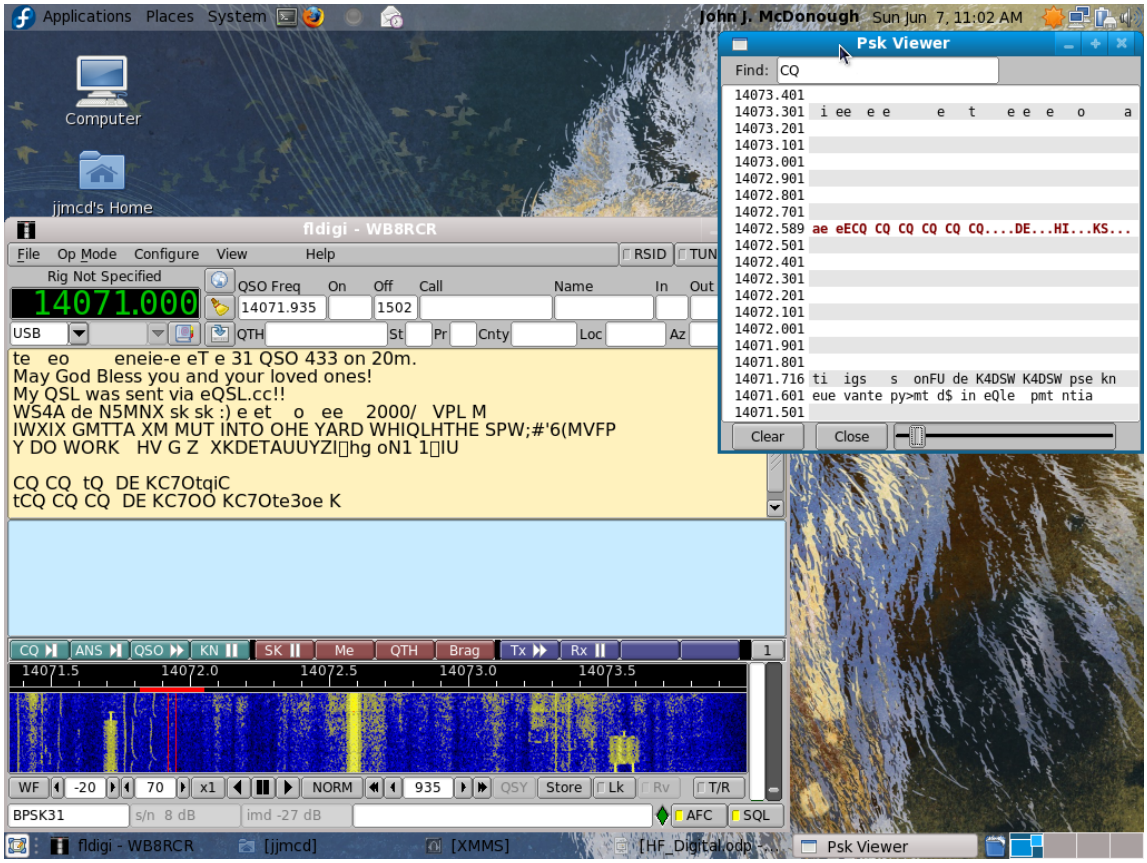

Figure 1. fldigi

3. Logging and related applications

.

3.1. qle

qle stands for QSO Logger and Editor. It is a simple yet flexible logging program. *qle* uses a lightweight *sqlite* database that can be manipulated using standard tools. The application is easily customized, so you can have the logging program behave the way you want. It also interfaces with *hamlib*, so information may be automatically retrieved from your rig with the appropriate hardware.

3.1.1. Installing qle

qle can be installed with yum like any other package:

```
sudo yum install qle
```

However, *qle* requires some initial setup before it may be used.

3.1.2. Configuring qle

The install process creates a configuration file **`/etc/qle/qle.conf`** which must be edited. This can be done with your favorite text editor, however, the file is protected against writing by a non-admin user. The file might be edited with something like:

```
sudo gedit /etc/qle/qle.conf &
```

There are two lines that must be changed. At line 63 of the file, you will find the lines:

```
#
debug = 0
#
myCall = N0CAL
#
```

Be sure that the **`debug`** line is set to zero and change the **`myCall`** line to reflect your callsign.

The second line that must be changed is at line 75 where you will find:

```
# Filename of SQLite DB with full path.
# This file requires sufficient RW access for the DB to work...
#
db = foo3.db
#
# Name of the table that you want to log into.
# Is probably case-sensitive:
#
tableName = mycall
#
```

You must change the name of the database to your desired name and location.

qle is set up for a single user system, so all users share the same database. You must place the database in a location where it can be accessed by any users requiring it. If you always log on with the same usercode, you might choose to put it in a hidden subdirectory off your logon directory, for example, **`~/.qle`**. This is the simplest approach, but in some circumstances, you may prefer a more "global" location, for example, **`/etc/qle`**. In this case, you need to take care to give the file appropriate protections.

For simplicity, we will assume that *qle* will only ever be run from a single usercode and we will put the database there. Reflect that location and name in **`qle.conf`**, for example:

```
db = /home/usercode/.qle/qle.sqlite
```

Note that you cannot use the tilde (**~**) within the config file, you must enter the entire path.

There are many things you may wish to change. For example, at line 101:

```
#
useRig = 1
#
```

determines whether you want to use the rig control library, *hamlib*, which can be a great convenience if you have the appropriate hardware.

At line 225:

```
#
noCwDaemon = 0
#
```

determines whether you wish *qle* to have the capability of keying the transmitter.

To avoid describing *hamlib* settings and hardware setup, we will assume these are both disabled for now, that is, **`useRig=0`** and **`noCwDaemon=1`**.

After editing **`qle.conf`**, you need to create the database. There is a sample database in **`/usr/share/qle`** so we can copy that to the location we have specified for our database:

```
cp /usr/share/qle/foo3.db ~/.qle/qle.sqlite
```

This file has some test data which we will delete after some initial testing.

3.1.3. Running qle for the first time

The first time you run *qle*, it should be done from the command line in debug mode to be sure you made no errors in the config file:

```
qle --debug=1
```

If there were errors editing the configuration file, they will appear in the window from which you started *qle*. If all went well, this should result in seeing the logging windows with the test data displayed:

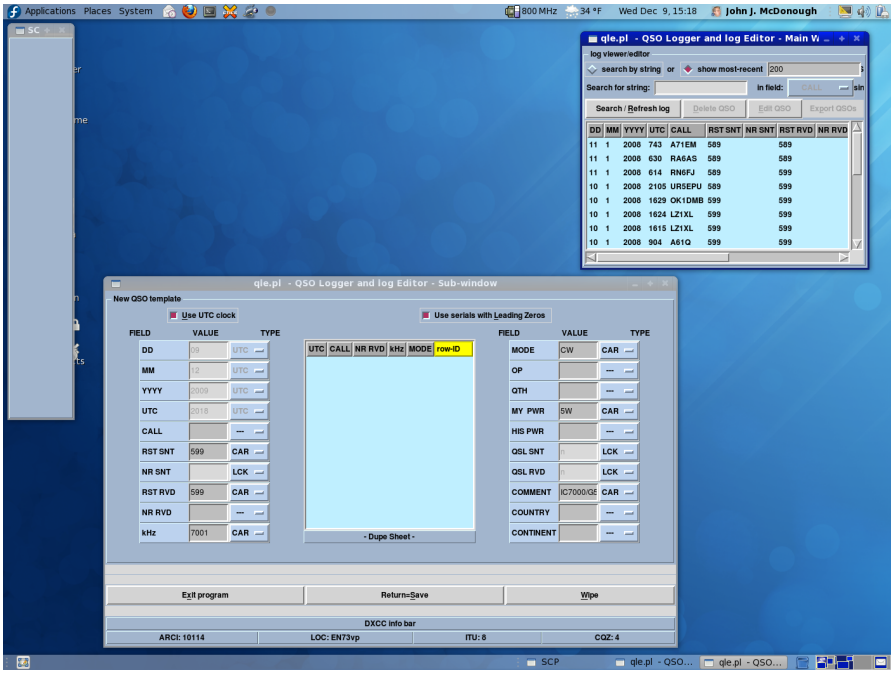

Figure 2. Running qle the first time

The *qle* "Main Window" shows the QSOs that have been logged so far. It will contain QSOs from the sample database. We will delete those QSOs later.

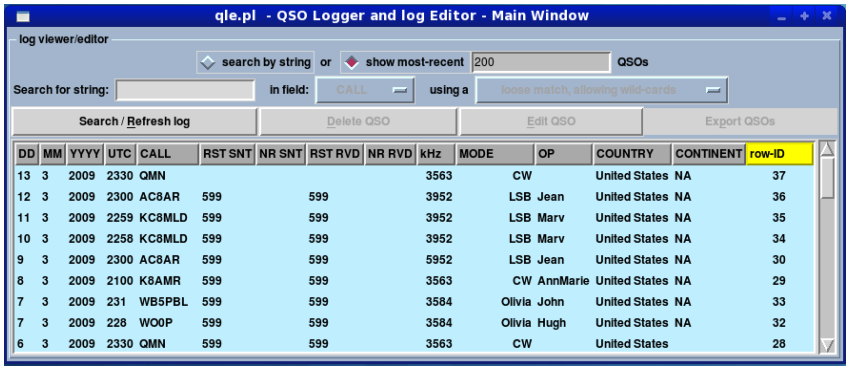

Figure 3. qle Main Window

The "Sub-Window" is actually where the data will be entered for each QSO. Some fields are provided automatically, such as the date and time. These will be a lighter color than the other fields. Each field has a button to the right indicating how that field is to be treated. Fields that have the label **CAR** will be carried over from QSO to QSO. These can be overwritten, but will initially be filled in with data from the previous QSO. These are things like Mode, Power, etc that tend not to change.

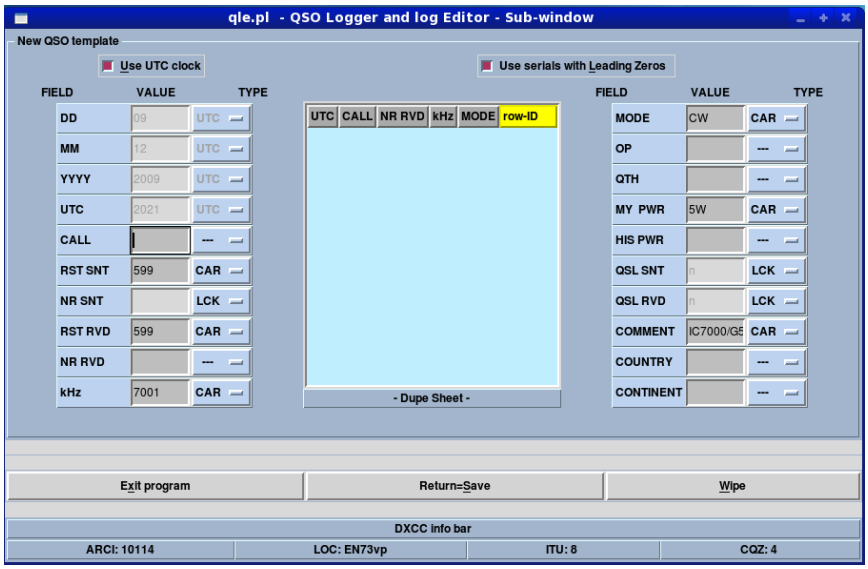

Figure 4. qle Data Entry Window

If you wish to change the data in a field that has the label **LCK**, you may simply click on **LCK** and select another choice from the dropdown. Normally, you might choose **--**, but if you are contesting, the **NR SENT** field includes a **+1** choice.`

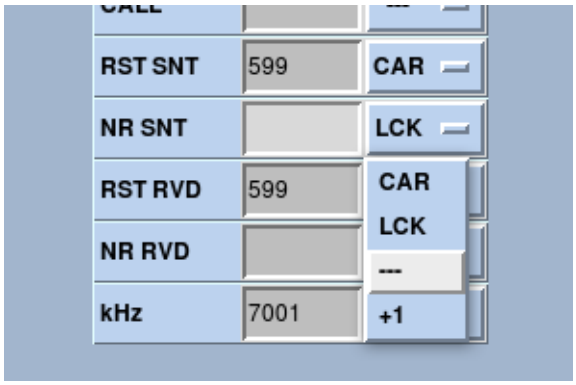

Figure 5. Changing Field Attributes

If you double-click an existing QSO in the Main Window, an Editing Window will appear, allowing you to make changes to the existing QSO.

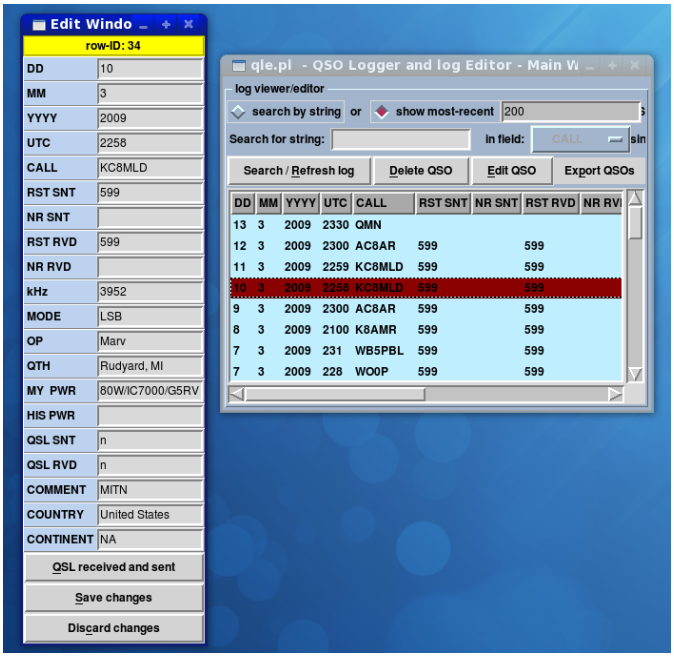

Figure 6. Editing an existing QSO

To exit the program, click the **Exit Program** button at the lower left of the subwindow.

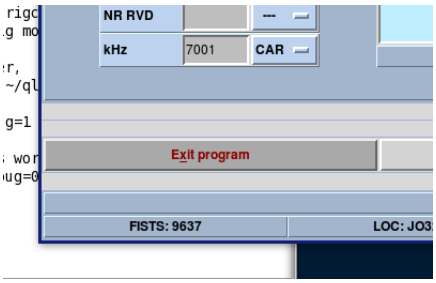

Figure 7. Exit Program Button

qle will ask for confirmation when exiting.

3.1.4. Deleting Sample Data

Now that you are familiar with the basic operation of *qle*, you will want to delete the original sample data so the database only contains your QSOs. Since the data is in a *sqlite* database, we merely need to use some simple commands to do that:

```
[jjmcd@Aidan .qle]$ sqlite3 ~/.qle/qle.sqlite
SQLite version 3.6.20
Enter ".help" for instructions
Enter SQL statements terminated with a ";"
sqlite> DELETE FROM mycall;
sqlite> .quit
[jjmcd@Aidan .qle]$
```

If you are familiar with SQL, you can also use *sqlite* to make other changes and queries.

You are now ready to begin using *qle*. By default, *qle* is available on the **Other** submenu of the **Applications** menu.

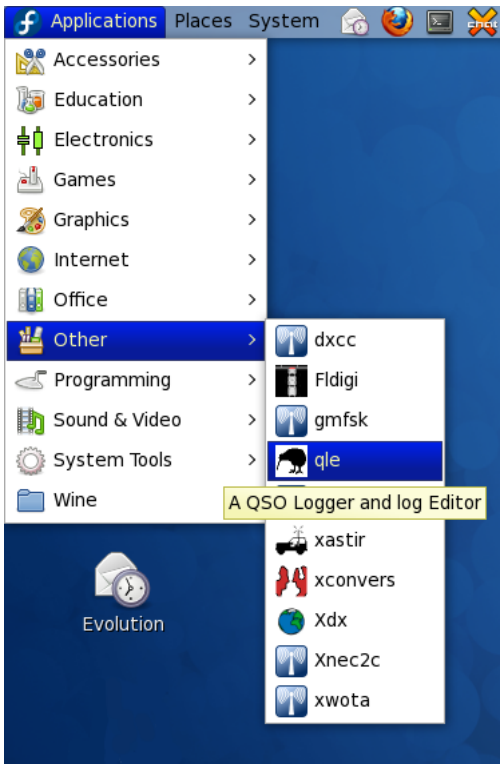

Figure 8. Menu selection

3.1.5. Duplicate Checking

qle provides duplicate checking as the call is entered. When the first character is entered, any calls in the log that match are shown in the duplicate window:

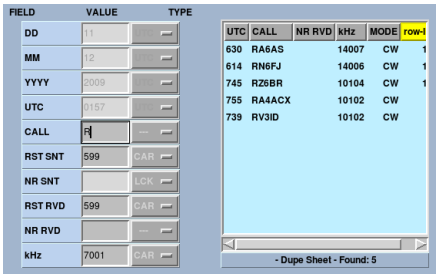

Figure 9. Dupe Sheet - first character

As additional characters are entered, the list gets shorter:

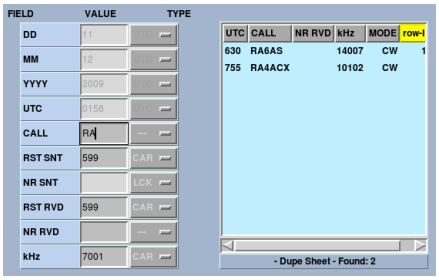

Figure 10. Dupe Sheet - second character

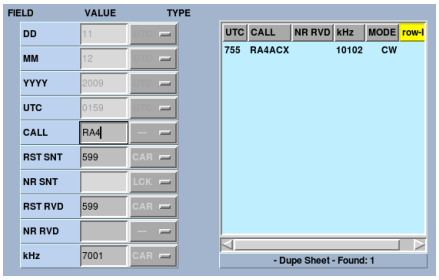

Figure 11. Dupe Sheet - third character

In addition to the current log, the file **`/usr/share/qle/master.scp`** contains a list of calls to check. These are shown in a separate SCP window:

Figure 12. SCP Window

Like the dupes window, this list gets shorter as you type. Edit **`master.scp`** to include the calls you want.

3.1.6. Some customizations you probably want

qle is highly customizable. In the previous sections, we described a few things that definitely need to be changed. There are quite a few more you probably want to change.

At the bottom of the secondary screen you will notice a bar with some information:

Figure 13. Subwindow Info Bar

This bar is simply a reminder for some random data you may want to keep handy. You can change the information by editing the lines starting at line 1051 in the configuration file:

```
infoString = "ARCI: 10114"
infoString = "LOC: EN73vp"
infoString = "ITU: 8 "
infoString = "CQZ: 4 "
```

You may find that you want the default field types to be different. For example, suppose you constantly change power and you don't want the previous power to be shown by default. At line 384 of the configuration file is a line for each field with the default type. You can simply change this to make *qle* start with the type you desire:

```
fieldTypes =  "---"           # mypwr
```

You can even adjust individual colors to make them as appealing (or as horrible) as you prefer:

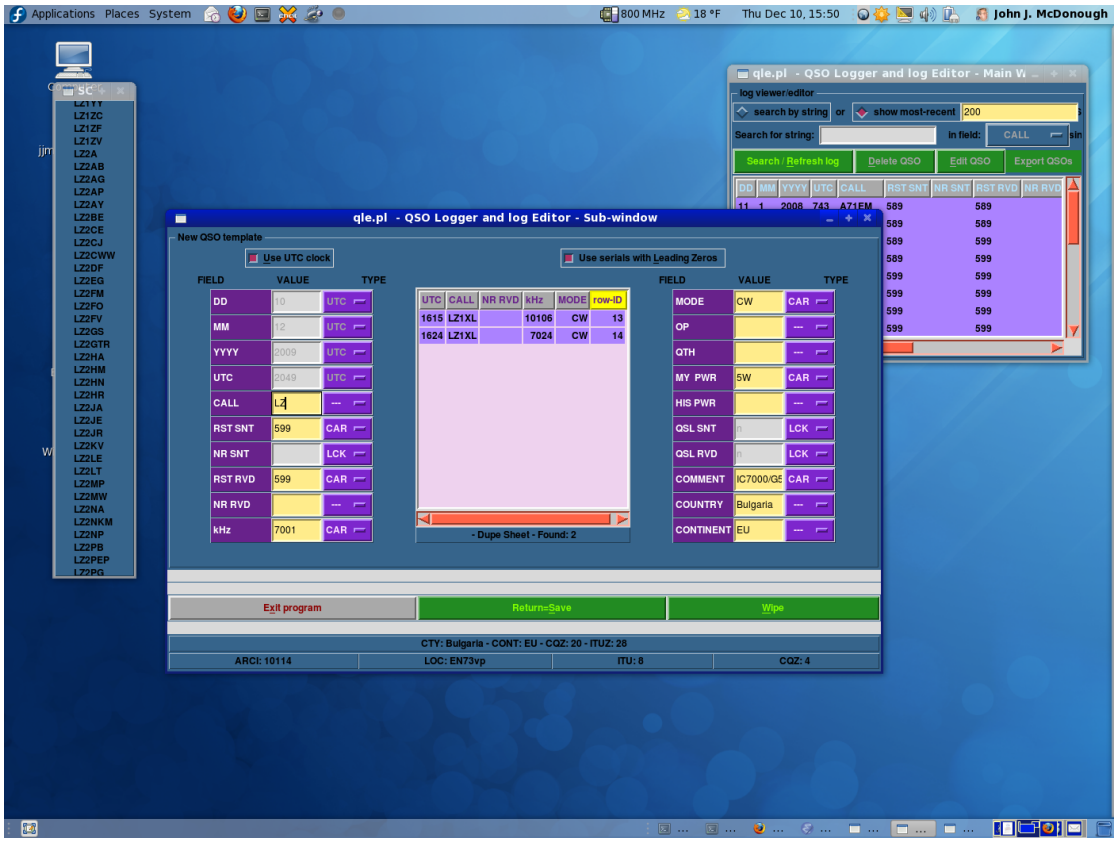

Figure 14. Colors only a mother could love

There are literally hundreds of adjustments you can make to tailor your logger to behave exactly the way you want. Just be sure to carefully review the comments in **`qle.conf`** so you don't get unexpected results.

3.2. xlog

xlog is a wonderful logging program that allows the user to create multiple logs, import and export in various formats, and maintain a count on the various awards such as WAS, WAC, DXCC, WAZ, and IOTA. Future versions will contain code to hook into the ARRL's LoTW.

3.2.1. Installing xlog

xlog is simply installed like most applications:

```
sudo yum install xlog
```

3.2.2. Starting xlog

xlog may be started from the menu by selecting **Applications->Other->xlog** or from the command line by typing **`xlog`**.

3.2.3. Setting up xlog

xlog is extremely easy to get setup. After starting xlog, select **Settings->Preferences**. This will bring up a preferences box where you can configure most options for *xlog*.

The **General** tab contains basic information on how the log will be setup including the modes and bands you operate. You can change these at anytime but it is good to go ahead and add or remove the modes and bands you don't operate to simplify the operation of the logging later. You can also enable the clock on the status bar and recording of azimuth and distance when you enter in the location of the station. You can also control out data from external programs, such as gmfsk and ktrack, are handled.

The **Info** tab contains information on your station and preference to miles or kilometers and where you want the software to look up a callsign. It is recommended that you enter your callsign and your coordinates into the fields located on this tab so the log can appropriately annotate your callsign where necessary and can provide azimuth and distance to a station upon entry of the state or grid. If you don't know your latitude and longitude you can just enter your grid locator and the software will populate a rough location for your station.

The **Hamlib** tab allows you to setup *xlog* to read your radio so your log will automatically record the frequency and mode. *xlog* will also display the S-meter on the status bar for your convinence.

The **Logs** tab allow you to setup the logs themselves. This includes where to store the logs, which logs to start automatically upon starting *xlog*, when to save the log, and the font. By default, *xlog* stores your logs in ~/.xlog. This can be changed by providing the appropriate path. If you have multiple logs you can type in the names of each log separated by a comma in the next field and *xlog* will load those logs each time using tabs at the top of the main screen. The next field asks if you want *xlog* to save the log whenever you write a log entry or every x minutes. You can also establish a backup of your logs in a separate directory which you can provide in the backup entry. The last field is used to select the font you would like to use for your logs.

4. Antenna and Propagation Modeling

.

4.1. splat

`splat` is a Surface Path Length And Terrain analysis application which can perform path loss calculations as well as generate coverage maps. Primarily intended for VHF/UHF, it can help plan repeater coverage or plan emergency communications strategies.

4.1.1. Installation and setup

Installing **`splat`** is straightforward:

```
su -c 'yum install splat'
```

4.1.1.1. Obtaining Terrain Files

Before it can be useful, **`splat`** requires files that describe the terrain around the station to be modelled. First, determine the latitude and longitude of the station. Then download the nine terrain files centered on that latitude and longitude from *http://e0srp01u.ecs.nasa.gov/srtm/version2/SRTM3/*.

Unzip the nine files and convert them from **`hgt`** files to **`sdf`** with the **`srtm2sdf`** utility. For example:

```
srtm2sdf N41W082.hgt
```

Do this for each of the nine files. Those files can now be placed in a directory where you wish to store terrain files, or they can be placed in the directory where you wish to work with **splat**

If you will be modelling stations over a wide geographic area, you may wish to download and convert additional files. **splat** will select those files it requires for a particular calculation.

4.1.1.2. Obtaining cartographic boundary files

splat will work with just the terrain files. However, for path loss maps, the resulting maps can be more useful if they are marked with political boundaries and names of towns and cities. For the United States, county outlines can be downloaded from *http://www.census.gov/geo/www/cob/co2000.html#ascii* and 'census designated areas' from *http://www.census.gov/geo/www/cob/pl2000.html#ascii*.

For each of these, there are two files, an **xxyy_d00.dat** and **xxyy_d00a.dat**, where xx is 'co' for county and 'pl' for place, and yy is a state number. A file of place names can be generated from the 'a' file with the **citydecoder** utility. For example:

```
citydecoder pl37 >cities.dat
```

The **cities.dat** file is simply a list of names followed by latitude and longitude. You may edit the file with a text editor to insert additional places which will be marked on the map with a red dot.

4.1.2. Using SPLAT!

splat can perform calculations for a particular path, or generate a map showing path loss or signal strength over a region. In any case **splat** needs at least one file identifying the transmitter location. For a specific path, it needs an identical file for the receiver. If you would like signal strength calculations, you will need another file with more details about the transmitter.

4.1.2.1. The QTH file

You tell **splat** about a particular station (transmitter or receiver) with a **qth** file. This file has four lines:

1. The name of the station
2. The latitude of the station
3. The longitude of the station
4. The antenna height above ground

Here is an example **qth** file:

```
W8KEA-4
43 38 05
84 15 41
124.0
```

The **qth** file should be named for the station. The name of the file in the above example would be **W8KEA-4.qth**.

By default, **splat** uses British units; heights are in feet, distances are in miles. However, invoking **splat** with the **-metric** switch will cause it to use metric units.

4.1.2.2. The LRP file

If you would like **splat** to calculate signal strengths, it needs to know a little more about the transmitter. You provide this information in a file whose name matches that of the **qth** file but has an extension of **lrp**.

The **lrp** file has 9 lines:

1. Earth Dielectric Constant. If you do not have measured data available, the **splat** man page has a table that can help you estimate a value.
2. Earth Conductivity
3. Atmospheric Bending Constant
4. Frequency
5. Radio Climate. This is a code describing the terrain. See the table in the man page
6. Polarization
7. Fraction of situations. This and the following line reflect how the Longley-Rice calculations are to be carried out. In the example below, **splat** will calculate the maximum path loss experienced 50% of the time in 50% of the situations.
8. Fraction of time
9. Effective radiated power - power out less feedline loss times antenna gain

```
15.000  ; Earth Dielectric Constant (Relative permittivity)
0.005   ; Earth Conductivity (Siemens per meter)
301.000 ; Atmospheric Bending Constant (N-Units)
145.090 ; Frequency in MHz (20 MHz to 20 GHz)
5       ; Radio Climate
1       ; Polarization (0 = Horizontal, 1 = Vertical)
0.50    ; Fraction of situations
0.50    ; Fraction of time
126.00  ; ERP
```

You may leave out the last line in which case **splat** will calculate only path loss.

4.1.2.3. Making a map of coverage

.

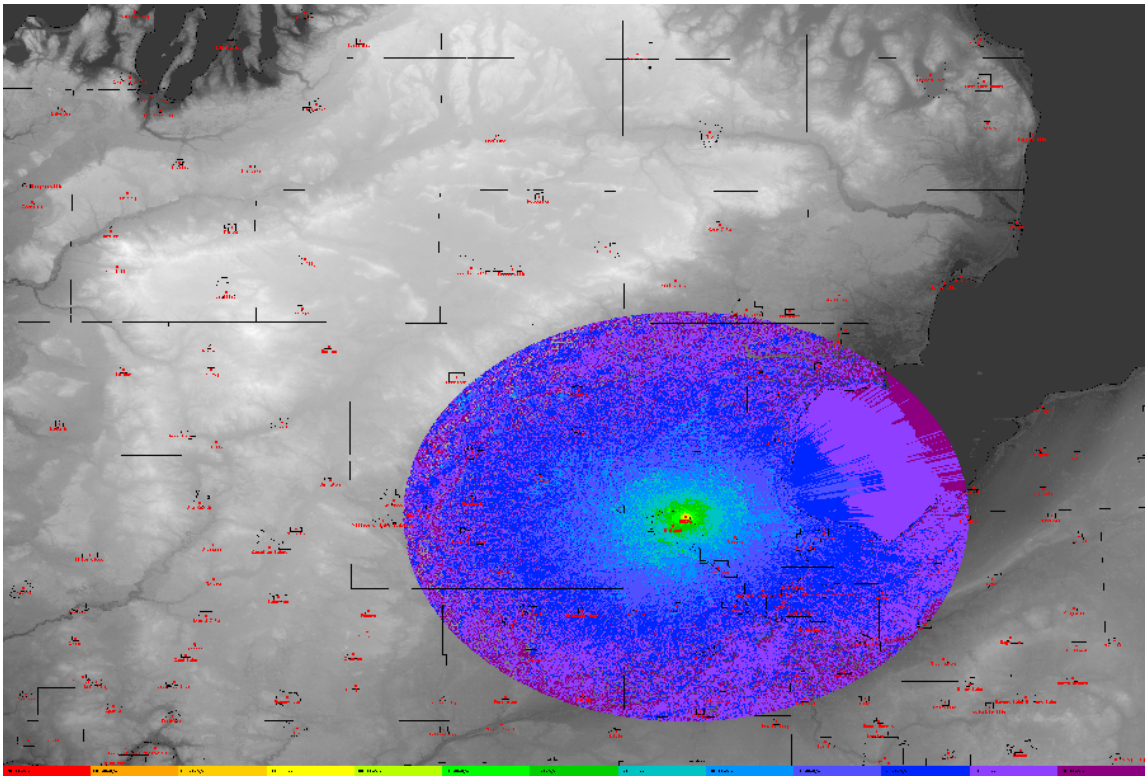

Figure 15. Coverage map with constrained distance

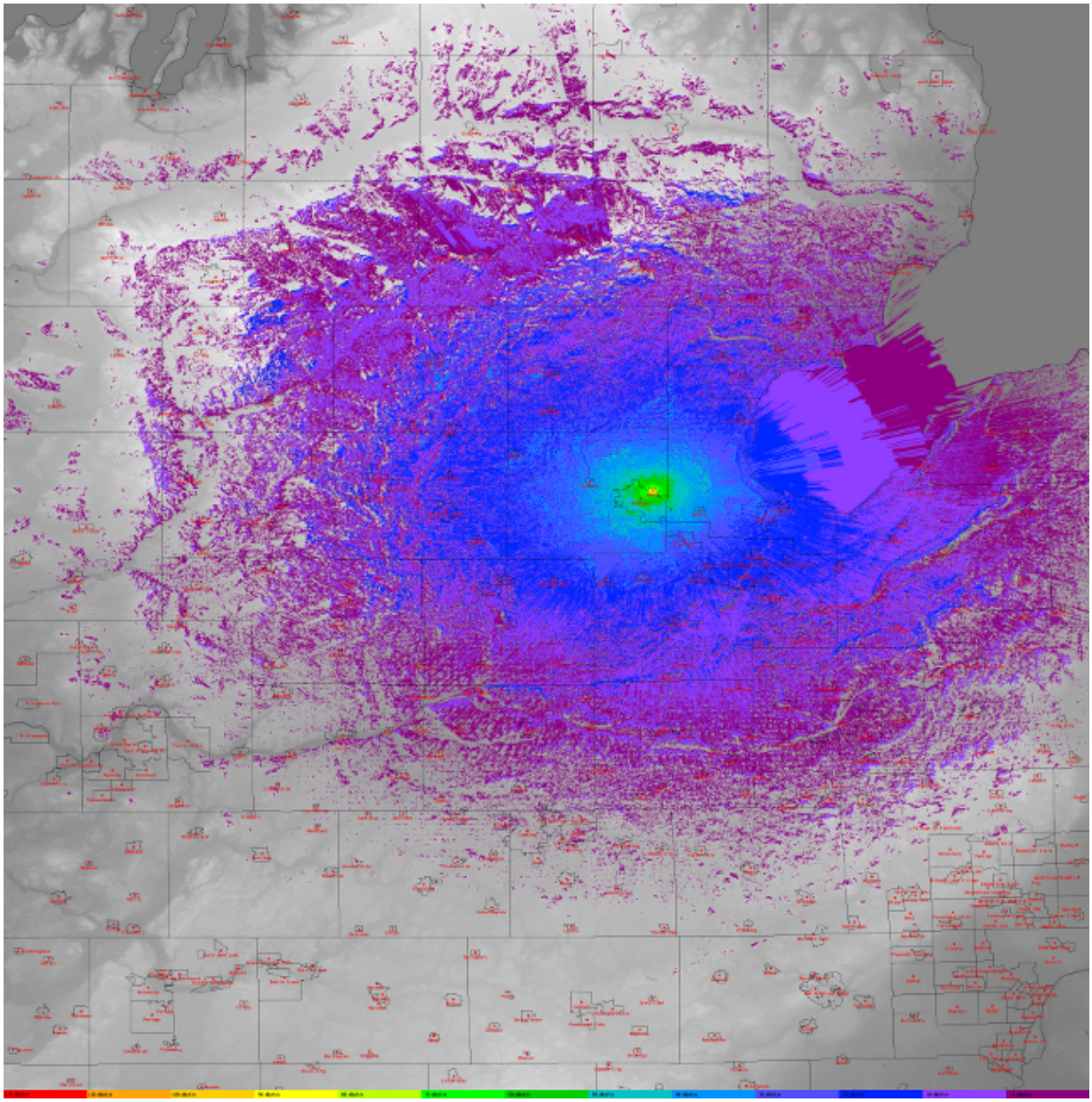

Figure 16. Coverage map with unconstrained distance

Figure 17. Signal Strength Legend

4.1.2.4. Calculating a point-to-point path

.

4.2. xnec2c

yum info as placeholder

```
Description: xnec2c is a GUI interactive application that (in its current form)
           : reads NEC2 input files but presents output data in graphical form,
           : e.g. as wire frame drawings of the radiation pattern or near E/H
```

```
: field, graphs of maximum gain, input impedance, vswr etc against
: frequency and simple rendering of the antenna structure, including
: color code representation of currents or charge densities.
:
: These results are only calculated and drawn on user demand via
: menu items or buttons, e.g. xnec2c is interactive and does not
: execute NEC2 "commands" in batch style as the original does.
```

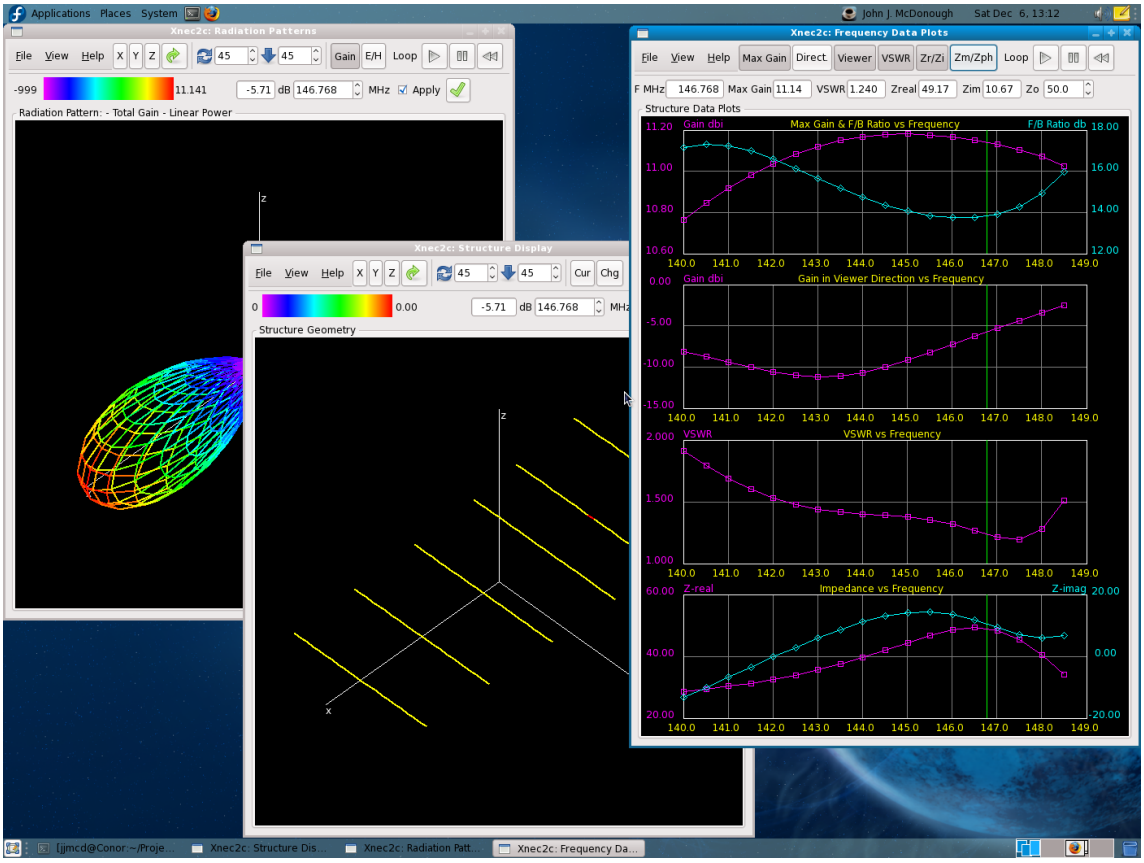

Figure 18. xnec2c - yagi

5. Packet and APRS

.

5.1. colrdx

Colrdx is a simple client for amateur radio dx-clusters. In a split-screen display you can type commands for the cluster in the bottom part. Messages from the dx-cluster will appear in the main window. There is also a status line at the top with some basic information.

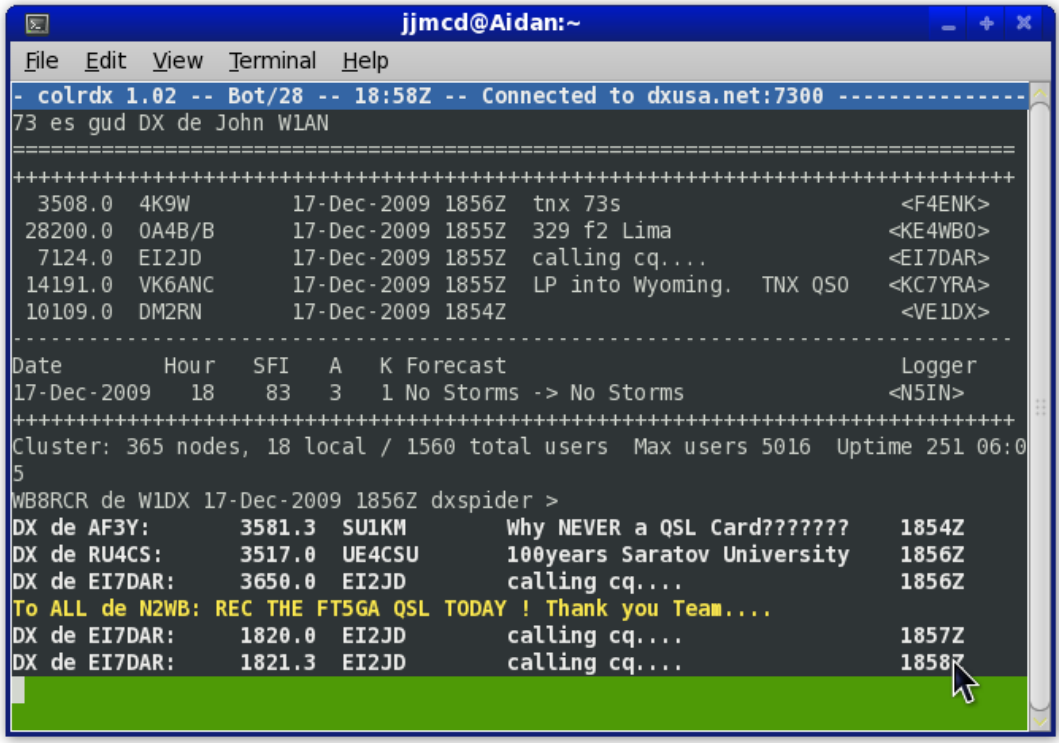

Figure 19. DX Cluser Client

To start *colrdx*, open a terminal window and type the command. You must provide your callsign and the name of the packet cluster. Optionally, you may also wish to provide the port:

```
colrdx -c <call> <nodename> [<port>]
```

You will see some introductory information from the cluster and spots will beging to appear. You may type commands to the cluster (dependent on the particular cluster). To exit type **`quit`**.

There is a manpage with additional details.

5.2. xconvers

xconvers is a client for packet based CONVerse bridges. When selected, *xconvers* will present a blank screen. Choosing **Open...** from the **Host** will open a dialog allowing the user to enter the name and port of the host.

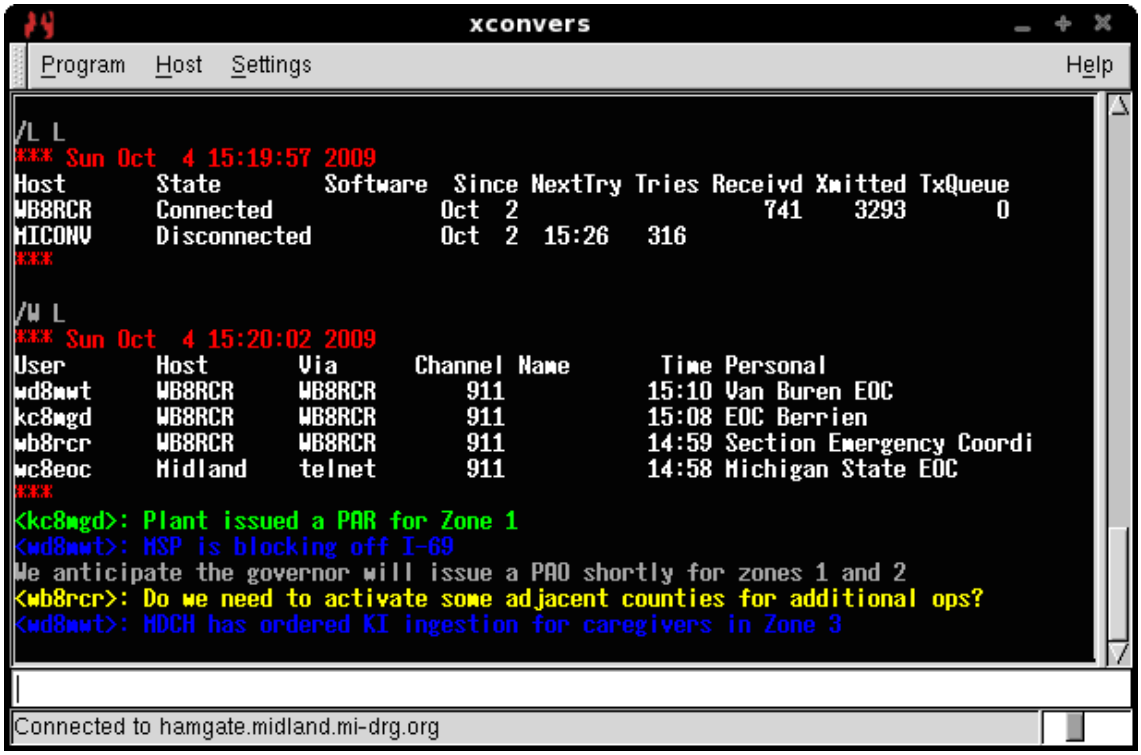

Figure 20. xconvers

Once connected, the user will see conversation on the channel. User input is seen in the lower part of the window and entered into the CONVerse bridge when Return is pressed.

Input from different users is seen in different colors. The colors and fonts may be adjusted by selecting **Preferences...** from the **Settings** menu.

5.3. xdx

xdx is a GUI client for packet clusters. When first launching *xdx*, select **Connect** from the **Host** menu and enter the desired host and port.

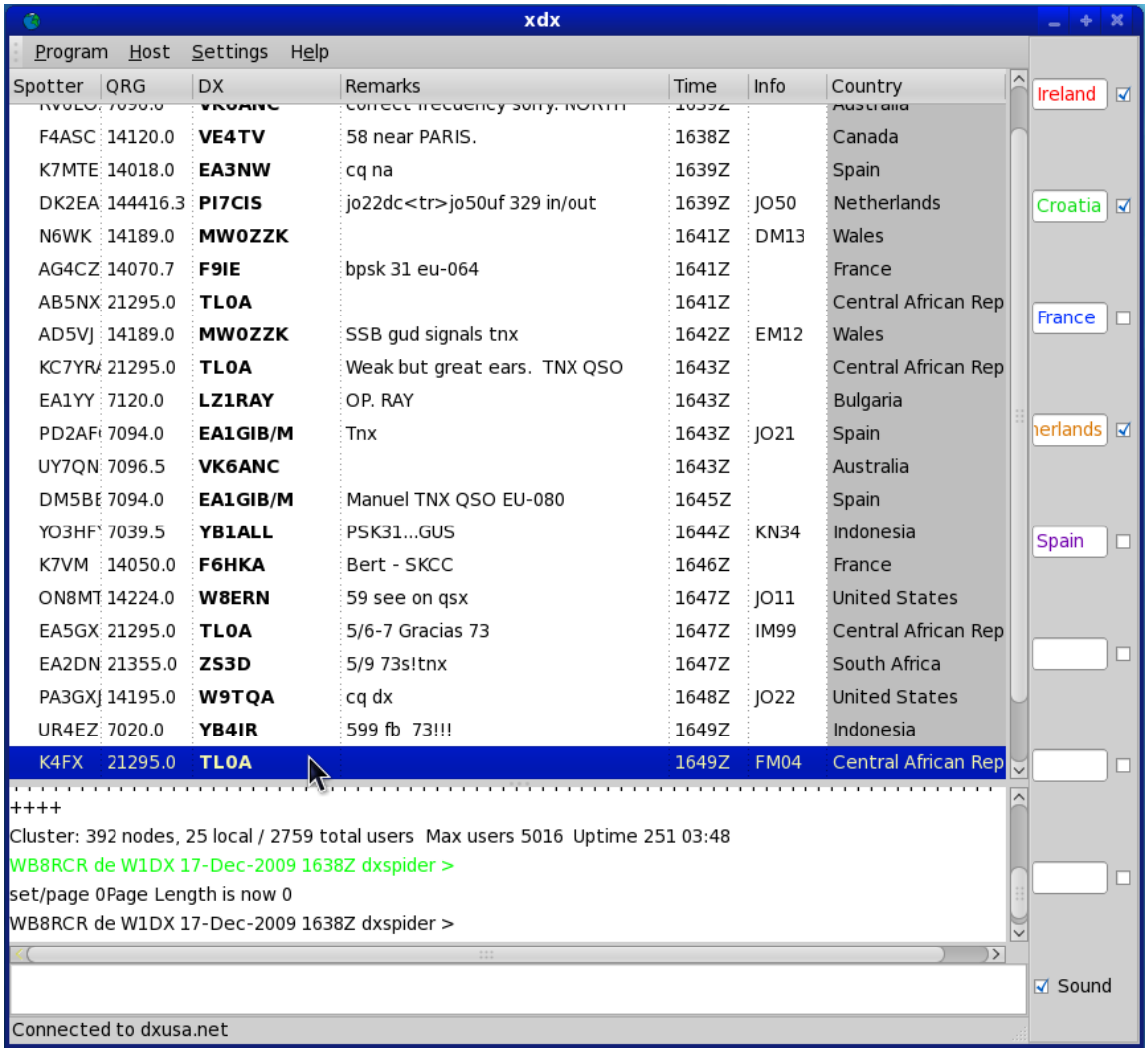

Figure 21. xdx cluster client

there are three panes in the display window. The split between the upper two may be adjusted by dragging.

5.4. xastir

xastir is an APRS application that allows users to send and receive position reports, messages, weather data, and other information over packet radio. Data received is displayed on a map allowing the user to get real-time information about a certain area. The user may also fetch trails from findu.com and display them on the map when connected to the Internet.

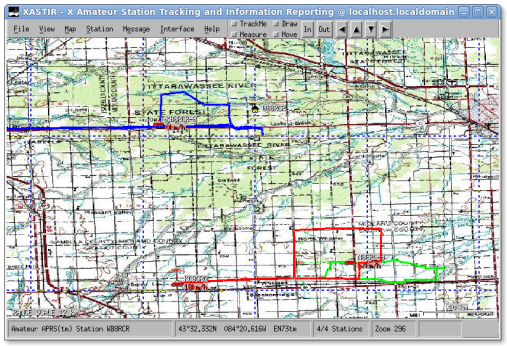

Figure 22. xastir

The map may be selected from a large number of sources. Facilities are also provided for drawing fixed items on the map as well as making measurements. The user may make specific queries to weather stations and Igates.

6. Circuit Design and Simulation

.

6.1. gEDA

gEDA is a collection of packages for schematic capture, netlist generation, circuit cimulation and PCB layout. Included in the geda suite are:
geda-docs - Documentation and example files
geda-gattrib - gEDA attribute editor
geda-gnetlist - Generates a netlist from a gEDA schematic
geda-symbols - A library of symbols for gEDA
geda-gschem - The gEDA schematic capture application
geda-gsymcheck - A symbol checker for schematics
In addition to the *geda-utils* utilities package, *geda-gaf* design automation package, and *libgeda* the gEDA library.

Closely tied into gEDA and mentioned elsewhere in this guide are:
pcb - The printed circuit board layout application
gerbv - Gerber viewer
gwave - The waveform viewer
ngspice - The circuit simulator
gspiceui - A GUI interface for *ngspice*

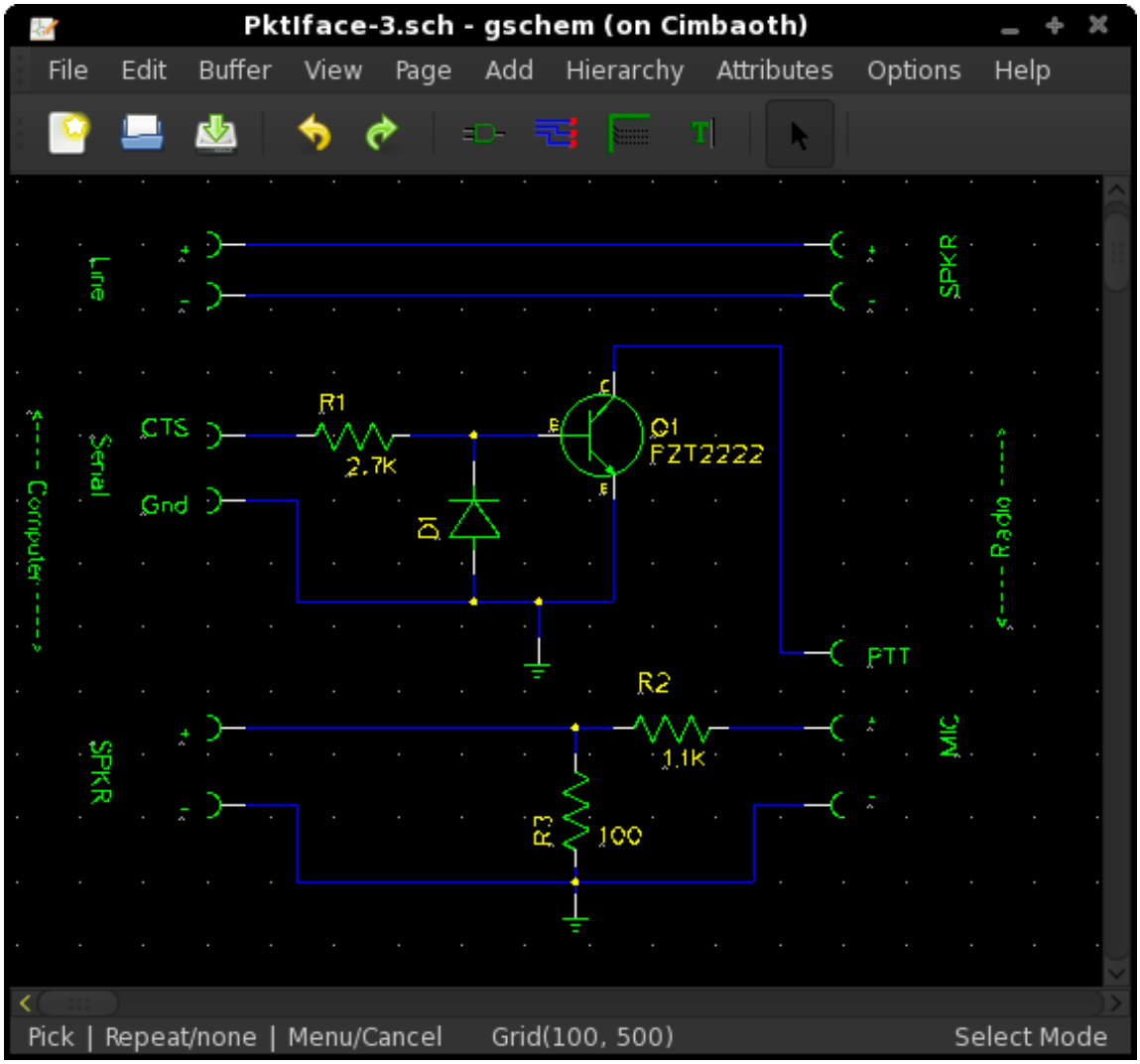

Figure 23. geda - gschem

Users wishing to take full advantage of gEDA should consider installing the *Electronic Lab* group which includes all the above components as well as the *electronics-menu* package and a number of other useful applications.

6.2. gerbv

grebv is a viewer for Gerber files.

In addition to selectively viewing and coloring Gerber layers, *gerbv* allows the user to export the image in a number of image formats for publication, as well as RS-274X compliant Gerbers and Excellon drill files.

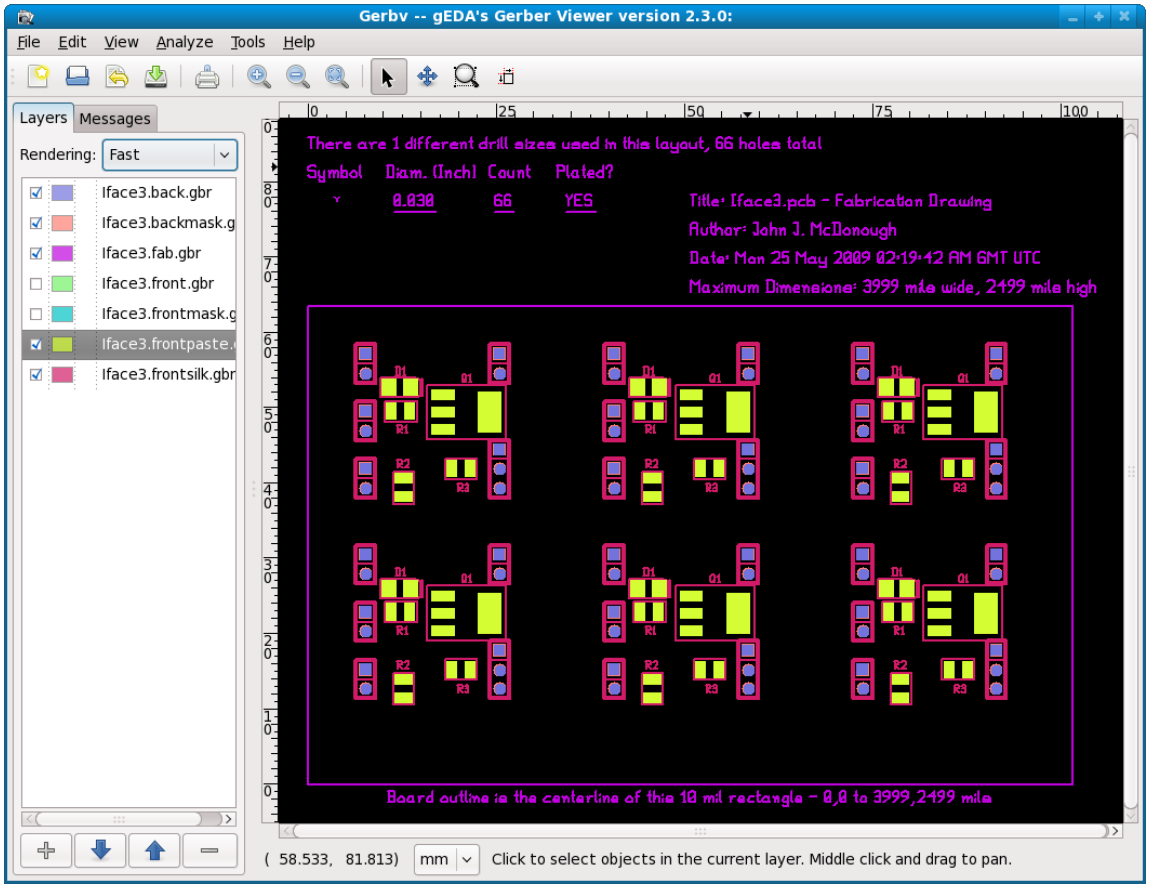

Figure 24. gerbv

6.3. pcb

pcb allows for the capture of printed circuit board layouts.

In addition to purely manual layout, *pcb* can import netlists from gschem. A large number of footprints are available or the user may develop his own.

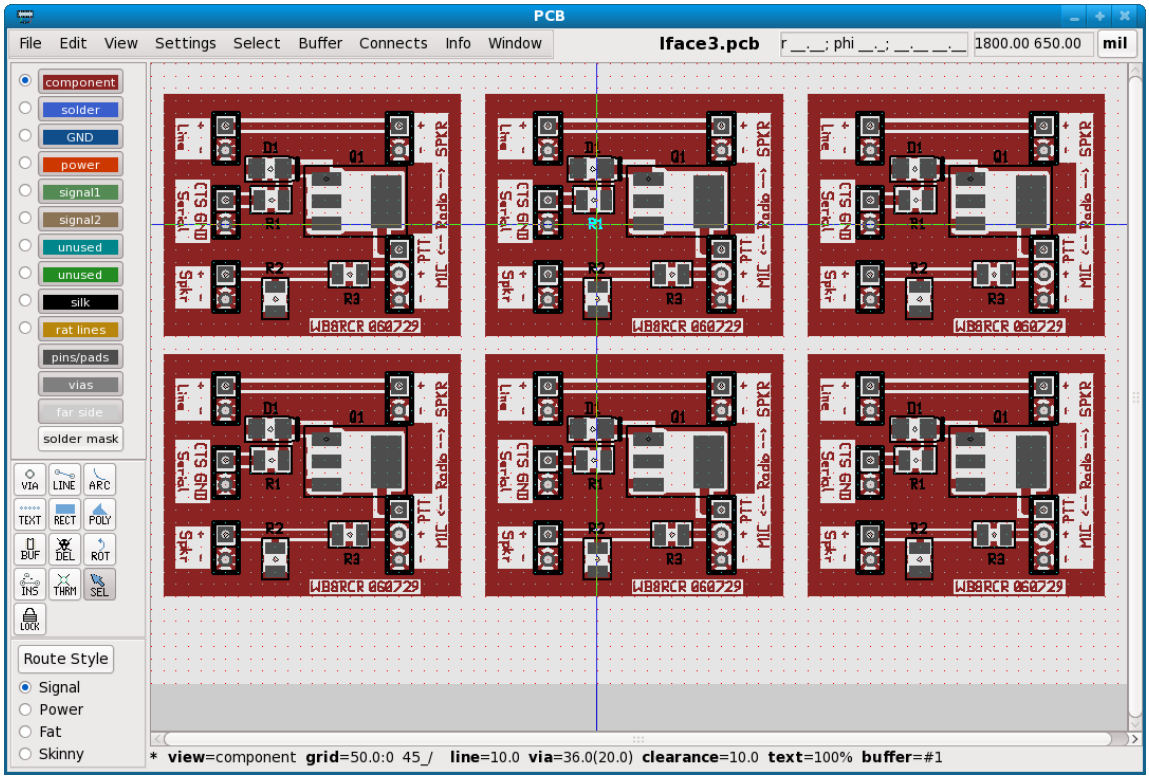

Figure 25. pcb

The application can generate a bill of material and drill file, and in addition to printing various layers can export in a number of popular formats.

7. Miscellaneous Applications

.

7.1. callgit

callgit is a simple application for callsign lookup. Enter the callsign and press **Search** and the callsign information is fetched from the web without the need to start a browser or download flashing ads.

In addition, callsign information can be saved in a second tab in case the user wishes to transfer the information at a later time.

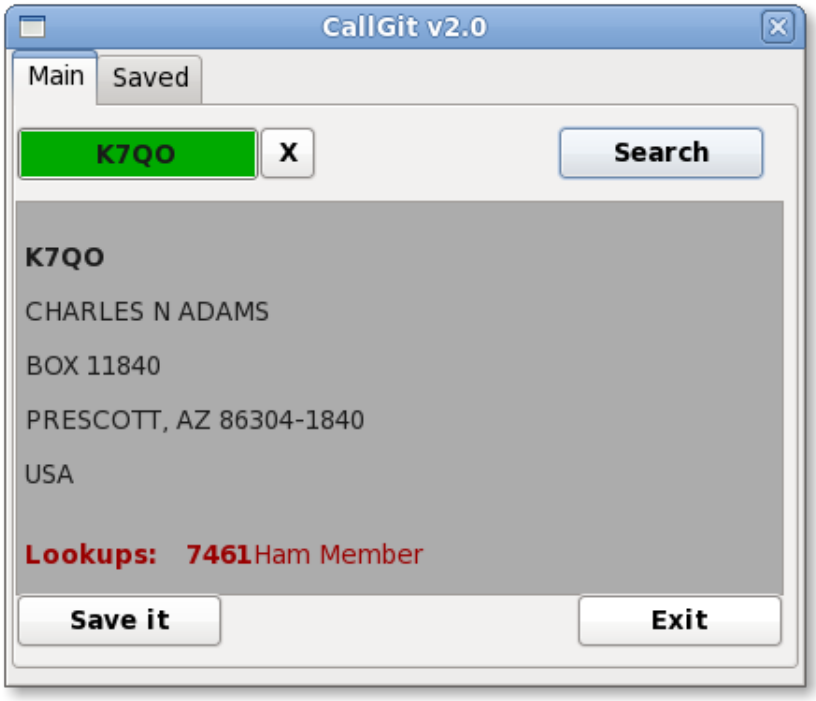

Figure 26. CallGit

7.2. dxcc

dxcc is a simple application to give quick information about a DXCC entity. Simply enter the callsign and see the country, WAZ and ITU zones, and other useful information.

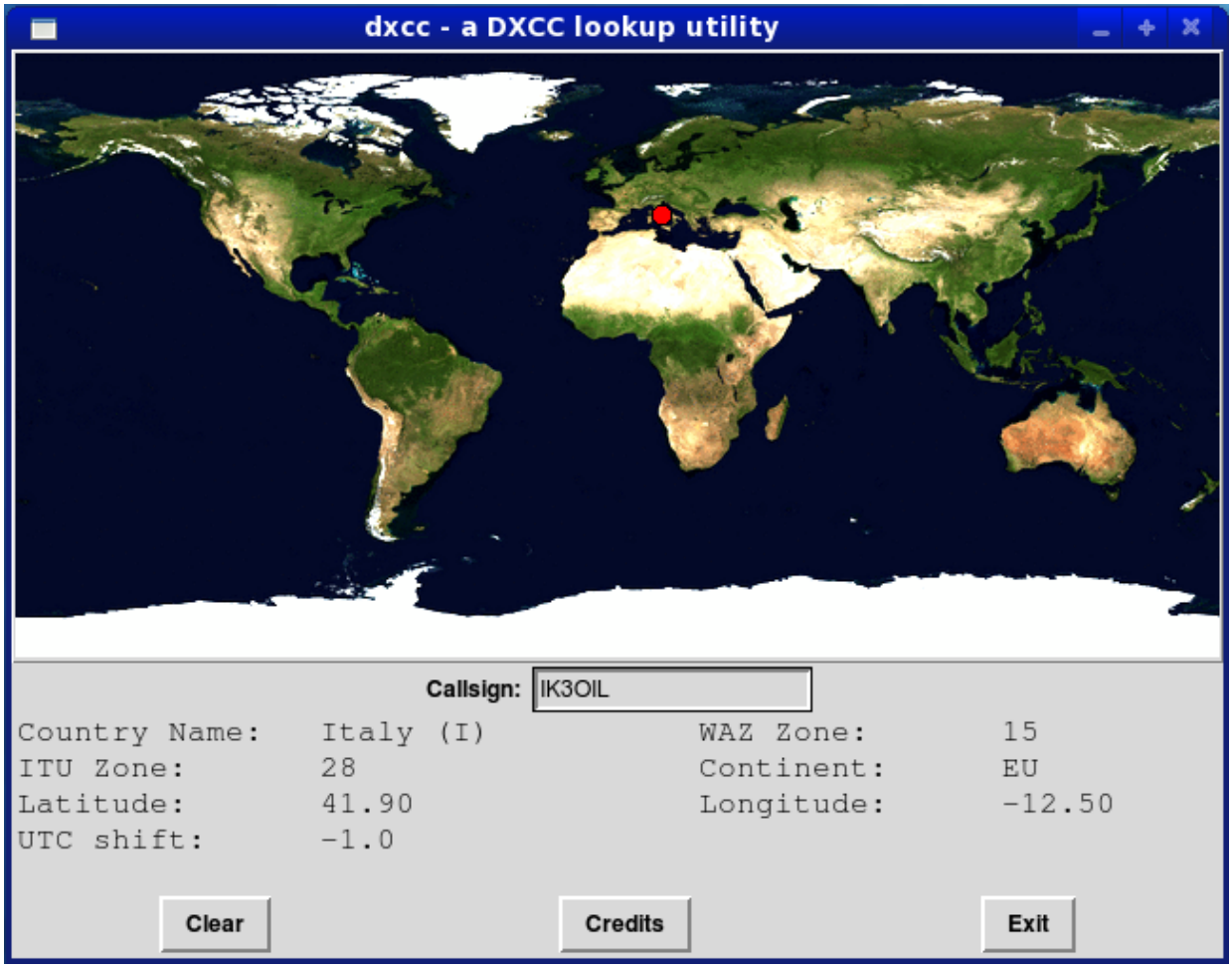

Figure 27. dxcc

7.3. gresistor

gresistor is a simple application for decoding resistor color codes. Select the number of bands on the resistor and each of the colors of the bands, and the resistor value and tolerance are displayed.

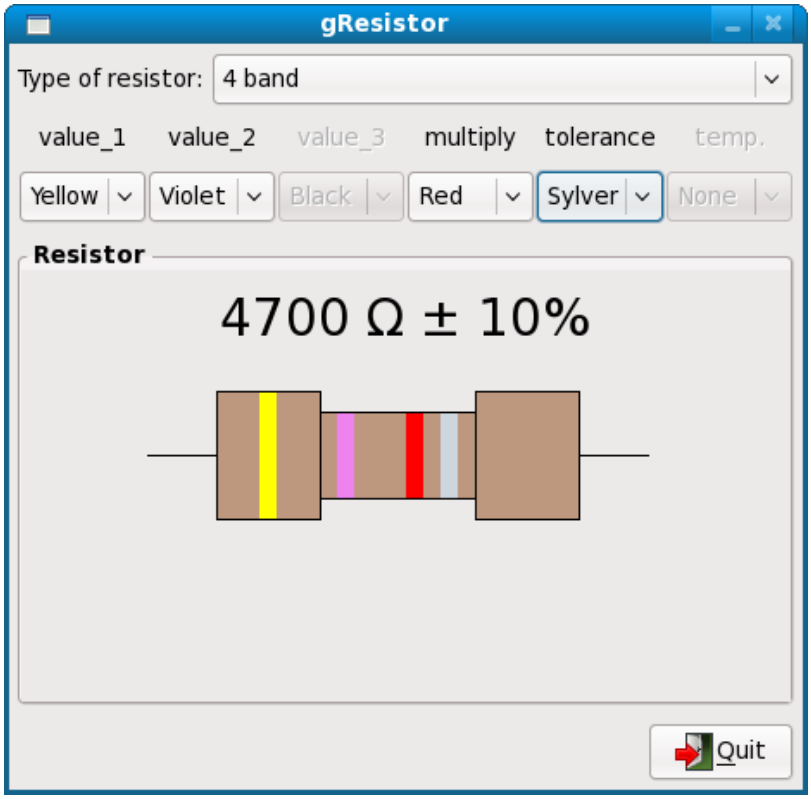

Figure 28. gresistor

7.4. ibp

ibp is a simple application that shows beacons which are part of the International Beacon Project. A number of beacons around the world transmit at predetermined times. The *ibp* application shows you which beacons are currently transmitting.

7.4.1. Installing ibp

ibp is simply installed like most applications:

```
sudo yum install ibp
```

No additional configuration is required, however, *ibp* expects that the time on the system is correct. Synchronizing your system with one of the many timeservers is recommended.

7.4.2. Starting ibp

ibp may be started from the menu by selecting **Applications->Other->ibp** or from the command line by typing **`ibp`**.

When *ibp* is started, by default, two windows will open. The first is a simple text screen showing a list of beacons with the currently transmitting beacons highlighted:

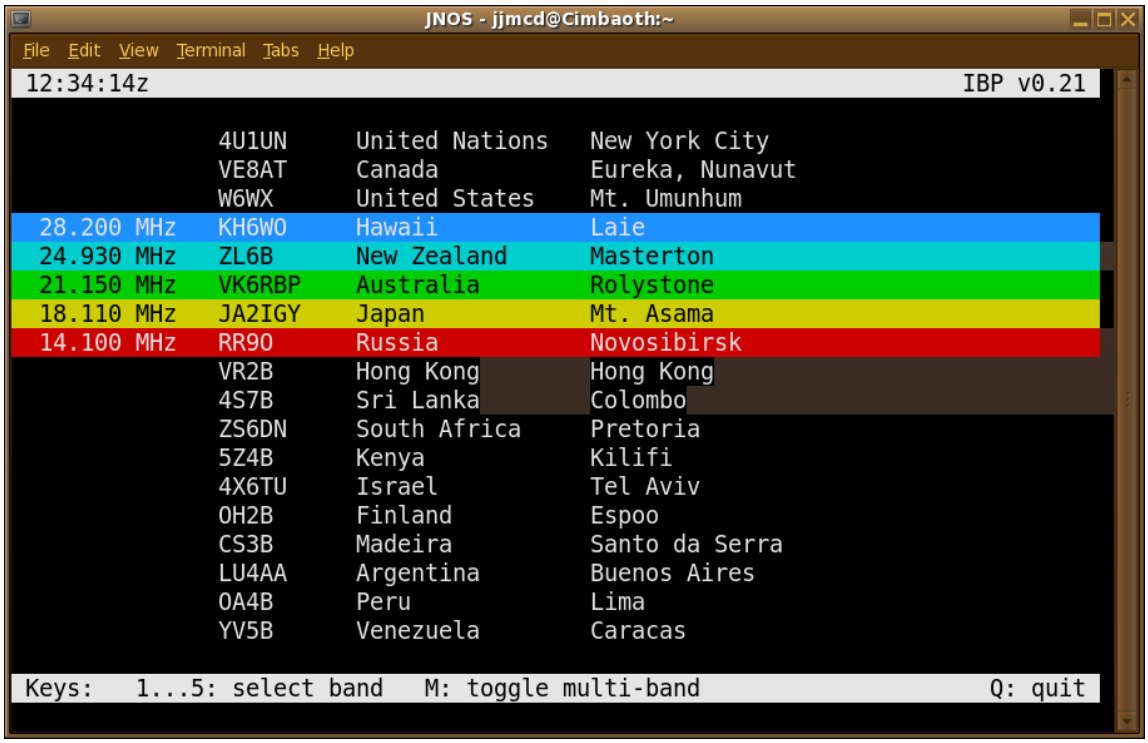

Figure 29. ibp - text screen

The second window shows a map of the world with a colored dot for each transmitting beacon:

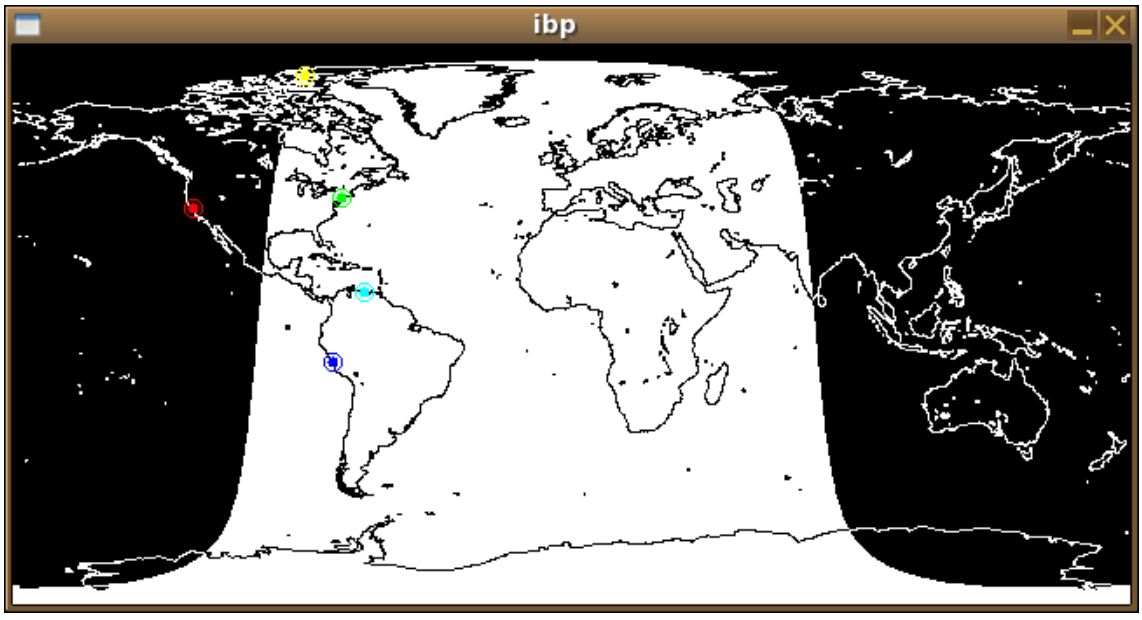

Figure 30. ibp - map

There are a number of arguments you may specify to affect how ibp behaves when it is started from the command line:

- `-c`, `--nocolor` - causes the text window to be displayed only in monochrome. The graph window is still in color.
- `-m`, `--morse` - In single beacon mode, causes the callsign of the transmitting beacon to be displayed at the bottom of the text window in Morse.

- `-x`, `--nograph` - Don't display the map window.

7.4.3. Running ibp

While *ibp* is running, the highlighted lines on the text display and the dots on the map will periodically change as different beacons take on the transmitting task.

There are several commands you can enter into the text screen to affect the behavior of *ibp*:

- digits **`1`** through **`5`** - causes only one band to be displayed. Since one is normally only monitoring a single band at a time this can lead to faster identification of the beacon of interest. This is also useful for visually challenged operators.
- **`M`** - toggles between single band and multi band mode. If a single band was displayed, typing **`M`** will cause all five bands to be displayed. If five bands were displayed, the previously selected single band will be displayed.
- **`Q`** - causes *ibp* to exit.

7.5. rcrpanel

rcrpanel is a command line application wich allows layout of panels for electronic equipment. *rcrpanel* accepts as input a text file describing the panel. It produces as output a Postscript stream of an image of the panel. The Postscript stream may be redirected to a file, a Postscript printer, or piped to another application such as Ghostscript.

By taking a text description of the panel, *rcrpanel* allows precise placement of controls and annotation, which can be difficult to achieve with a GUI interface. *rcrpanel* provides scripting elements for text, controls of various sizes, and even calibrated dials.

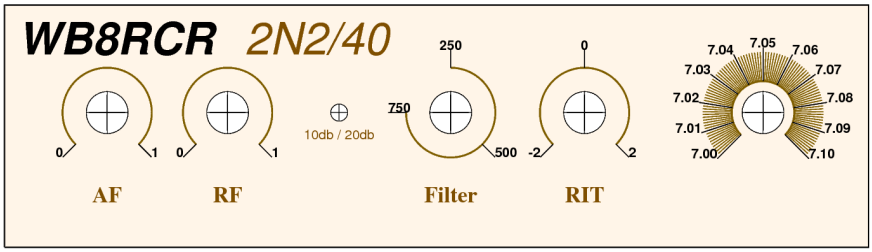

Figure 31. Example Panel

7.5.1. Running rcrpanel

rcrpanel accepts a single command line parameter, the input file containing the description of the panel. It produces its output on **`stdout`**, which means that in most cases, the user will redirect the output to a file. For example:

```
rcrpanel mypanel.txt >mypanel.ps
```

There are no command line switches available.

The output image will be centered on a standard size page. The smallest page on which the panel will fit is selected from the following list, in order:
216x179 mm - U.S. Letter
210x297 mm - A4
216x279 mm - U.S. Legal
297x420 mm - A3

279x432 mm - Tabloid
594x841 mm - A1
559x894 mm - D
841x1189 mm - A0
1000X1414 mm - B0

7.5.2. The Input File

The input file contains lines describing the various controls. Most lines are of the form

```
Command = something
```

where the spaces around the equal sign are significant, and the command itself is case-sensitive.

Measurements are in units of millimeters. Angles are in degrees. Colors are given as 24 bit C style integers where each byte repesents the amount of red, green, or blue.

In general, the order of commands makes no difference. However, the **`Text`** command must be immediately followed by a line containing the text to be displayed, and those commands affecting the appearance of a **`Dial`** affect the preceding **`Dial`** command.

7.5.2.1. Background

This command takes a single color following the equal sign. The entire panel will be filled with this color.

```
Background = 0xfff5e8
```

Note, however, that the interior of controls will not be filled with this color, allowing the alignment marks to be viewed for drilling, even if the panel were filled with a dark color.

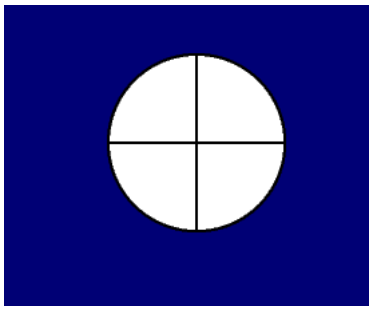

Figure 32. ControlPhone on a dark background

7.5.2.2. ControlLarge

This is used for large diameter controls such as large pots and the like. It takes 2 values after the equal sign representing the position of the control on the panel.

```
ControlLarge = 23.0 30.0
```

7.5.2.3. ControlLED

This command generates an outline for a 5 mm LED. Like the other control commands, it takes 2 values, the X and Y positions on the panel of the center of the LED.

7.5.2.4. ControlPhone

This is used for 1/4" phone jacks and similar controls. The 2 values after the equal sign represent the position on the panel.

7.5.2.5. ControlSmall

This command generates an outline for a 3.5 mm phone jack. The two values are the X and Y positions of the jack on the panel.

7.5.2.6. ControlTiny

This command generates an outline for a 2.5 mm phone jack. The two values are the X and Y positions of the jack on the panel.

```
ControlTiny = 75.0 30.0
```

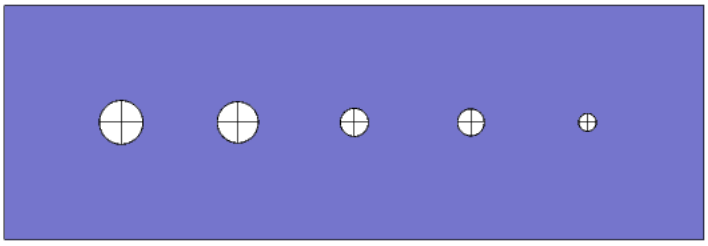

Figure 33. Large, Phone, LED, Small and Tiny controls

7.5.2.7. Panel

This command defines the size of the panel. The 2 dimensions are the width and height of the panel.

```
Panel = 193.675 53.975
```

7.5.2.8. Reverse

This command takes no arguments. If this command appears anywhere in the input file, the resulting PostScript will be flipped left to right (for printing on the reverse side of transfer media or transparency material).

7.5.2.9. Text

This command is somewhat different from the others. After the equal sign, it takes 3 floating point numbers, a color, and a text string. The first 2 floating point numbers are the X, Y position of the text on the panel. The third number is the height of the text. The color represents the color of the text, and the text string represents the font to be used. No checking is done before preparing the PostScript; you are responsible for ensuring that the font is availble on your printer.

This command is then followed by another line containing the text to be displayed.

```
Text = 100.0 10.0 5.0 0x7f4f00 Times-Roman-Bold
Filter
```

7.5.2.10. Dial

This command introduces a new dial. The **`Dial`** command describes the X,Y center of the dial. The following commands then further refine the details of this particular dial. This relationship between the **`Dial`** command and it's successors is the only place where the order of the commands within the file matters.

```
Dial = 170.0 30.0
```

7.5.2.11. Radius

This command takes a single value which is the radius of the circle which forms the inside of the tick marks. This command refers to the current **`Dial`** command.

```
Radius = 7.0
```

7.5.2.12. Span

This command describes the angle over which the control may operate. Typically, this would be 270 for a potentiometer and 180 for a variable capacitor. This command refers to the current **`Dial`** command.

7.5.2.13. NumTicks

This command describes the total number of tick marks, large and small, to be drawn. This is usually an odd number since the starting and ending values are counted. Typically this will be 11, 101, or a similar number. This command refers to the current **`Dial`** command.

```
NumTicks = 101
```

7.5.2.14. BigPer

This command tells the program how many small tick marks there are per large tick mark. This command refers to the current **`Dial`** command.

```
BigPer = 10
```

7.5.2.15. SizeTicks

This command describes the length of the small tick marks. This command refers to the current **`Dial`** command.

```
SizeTicks = 6.5
```

7.5.2.16. SizeBig

This command describes the length of the large tick marks. This command refers to the current **`Dial`** command.

```
```

```
SizeBig = 7.5
```

7.5.2.17. StartingIndicator

This command describes the value to be placed on the furthest counterclockwise large tick mark. This command refers to the current **`Dial`** command.

7.5.2.18. IncrementPerBigTick

This command tells rcrpanel how much to increment the value in **`StartingIndicator`** for each succeding large tick mark. This command refers to the current **`Dial`** command.

7.5.2.19. SizeFont

This command describes how large to make the annotation on the ticks. This command refers to the current **`Dial`** command.

7.5.2.20. ColorCircle

This command takes a single color as an argument, which is used to draw the inner circle. This command refers to the current **`Dial`** command.

7.5.2.21. ColorTickMarks

This command permits setting the color to draw the small tick marks. This command refers to the current **`Dial`** command.

7.5.2.22. ColorBigTickMarks

This command permits setting the color to draw the large tick marks. This command refers to the current **`Dial`** command.

7.5.2.23. ColorText

This command accepts a single color which will be used for the annotation. This command refers to the current **`Dial`** command.

7.5.2.24. StartAngle

By default, *rcrpanel* arranges dials so the dead spot on the control is straight down. This is the desired behavior in almost all cases. However, sometimes you may want to rotate a control to some other orientation. The single argument to **`StartAngle`** is the number of degrees clockwise to rotate the control. This command refers to the current **`Dial`** command.

7.5.3. Example Dials

7.5.3.1. Frequency Markings for a VCO

```
Dial = 25.0 25.0
Radius = 7.0
SizeTicks = 4.5
ColorTickMarks = 0xff7777
SizeBig = 7.5
```

```
ColorBigTickMarks = 0x000000
StartingIndicator = 7.0
IncrementPerBigTick = 0.01
NumTicks = 51
BigPer = 5
ColorCircle = 0xff7777
SizeFont = 3.0
```

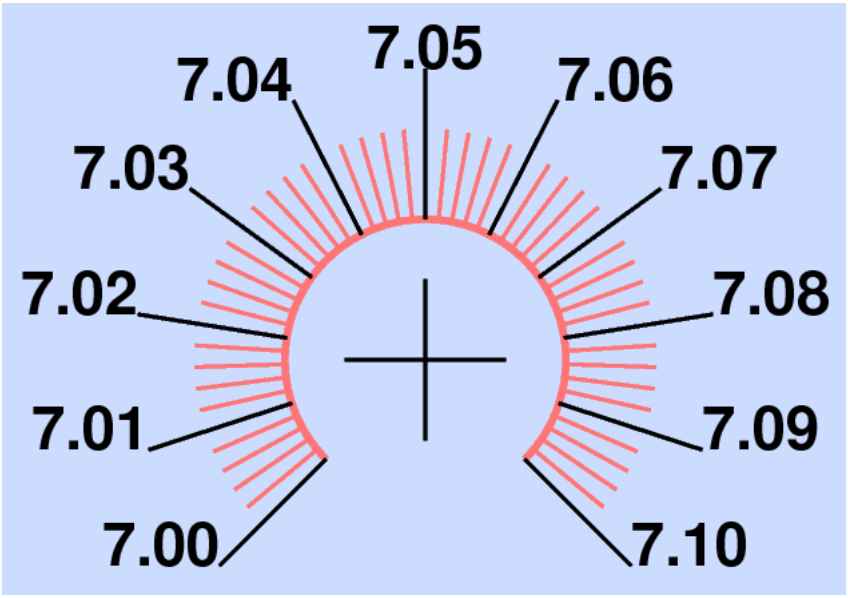

Figure 34. 40 meter dial, 270 degrees

7.5.3.2. Markings for a volume control

```
ControlLarge = 25.0 25.0
Dial = 25.0 25.0
Radius = 7.0
SizeTicks = 1.0
ColorTickMarks = 0xaaddaa
SizeBig = 2.0
ColorBigTickMarks = 0x007f00
StartingIndicator = 0
IncrementPerBigTick = 2
NumTicks = 11
BigPer = 2
ColorCircle = 0xccffcc
SizeFont = 3.0
```

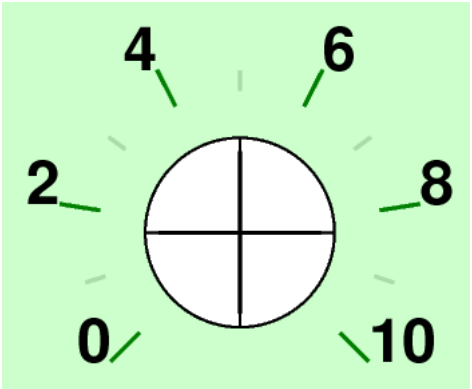

Figure 35. Volume Control

7.5.3.3. Markings for a VFO (capacitor based)

```
ControlLarge = 25.0 25.0
Dial = 25.0 25.0
Radius = 7.0
SizeTicks = 3.5
ColorTickMarks = 0x777777
SizeBig = 5.5
ColorBigTickMarks = 0xffffff
StartingIndicator = 3.5
IncrementPerBigTick = 0.01
NumTicks = 41
BigPer = 4
ColorCircle = 0x777777
SizeFont = 2.0
ColorText = 0xffffff
Span = 180.0
Text = 25.0 15.0 2.0 0xff0000 Century-Schoolbook
Frequency
```

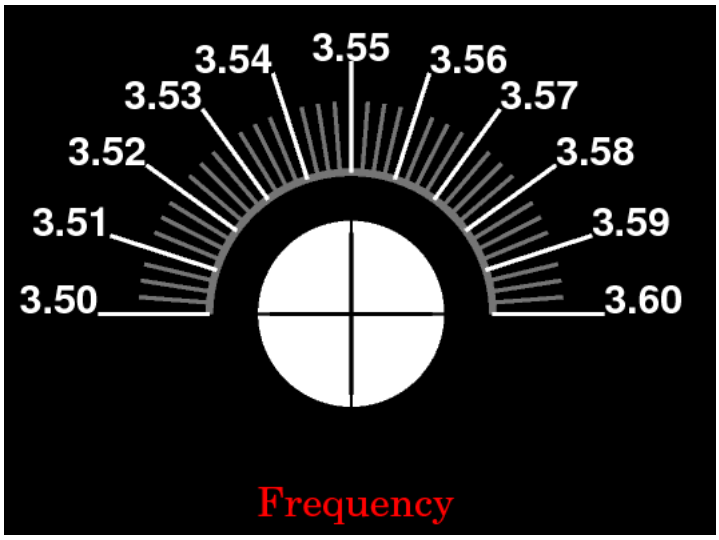

Figure 36. 80 meter VFO

7.6. xgridloc

xgridloc is an application which will translate a latitude/longitude into a Maidenhead grid square. It will also calculate the Great Circle distance and bearing between two locations.

7.6.1. Installing xgridloc

xgridloc is installed like most applications in Fedora:

```
sudo yum install xgridloc
```

7.6.2. Setting up xgridloc

xgridloc uses a small configuration file, **`~/.xgridlocrc`**. Before using *xgridloc* you should replace the default location in the file with your station location using your favorite text editor:

```
######### Runtime config file for 'xgridloc' #########
```

```
#
### Blank lines and those starting with a # are ignored ###
#
# The 'Home' location's position.
# (East Longitude and North Latitude)
# Format is "East/ddd:mm:ss North/dd:mm:ss"
West/084:11:59 North/43:38:06
#
# The name of the 'Home' location
Midland
#
```

7.6.3. Using xgridloc

xgridloc may be started by selecting the menu item **Applications->Other->xgridloc** or by issuing the **`xgridloc`** command from the command line.

Clicking the **Default Home Position** button will cause the top location to be filled in with the location you specified in the configuration file.

If you enter a latitude and longitude in either the "Home" or "Remote" location and press **`Enter`**, the corresponding **Locator** box will be filled in with the Maidenhead grid square for that location.

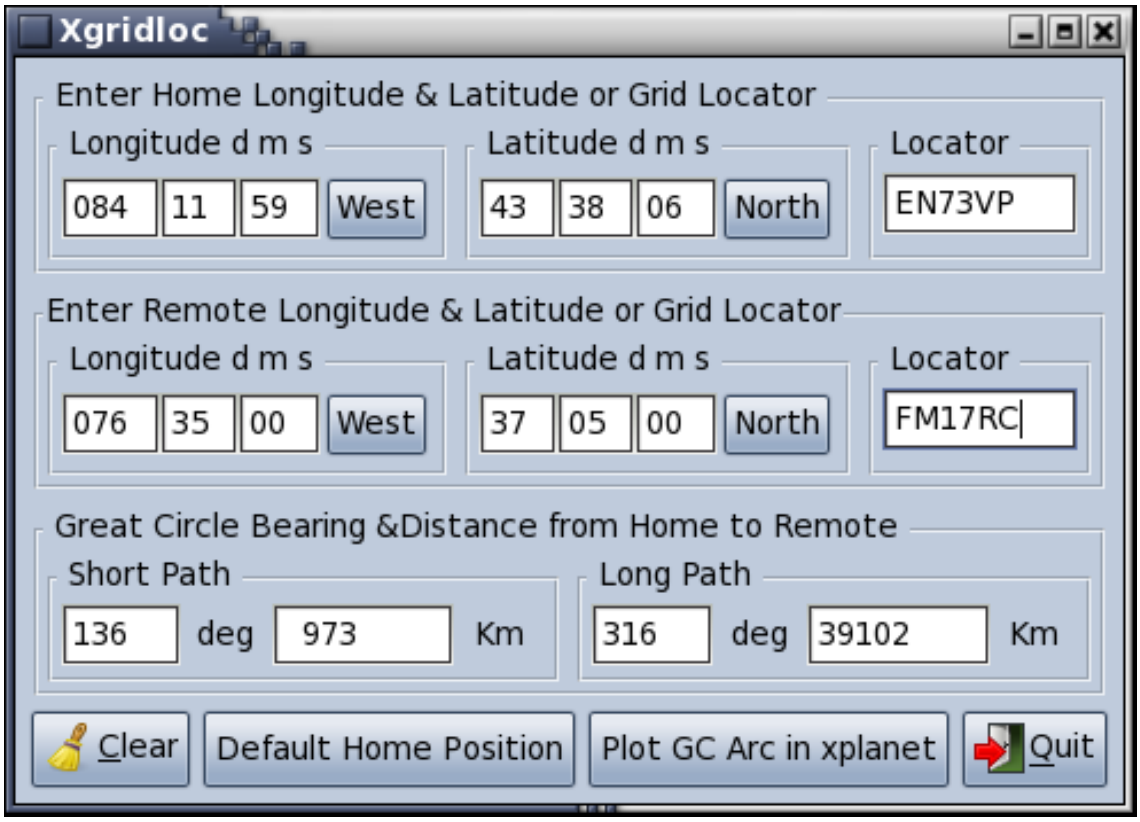

Figure 37. xgridloc

If both locations are filled, the Great Circle bearing and distance will appear at the bottom of the window.

7.7. xwota

xwota allows for monitoring and querying the WOTA database. It operates much like a DX Cluster client, except it uses the database rather than a cluster. Refer to *http://www.wotadb.org* for information about the WOTA database.

Selecting **Connect** from the **Xwota** menu will connect to the database. New reports will appear on the screen as they arrive.

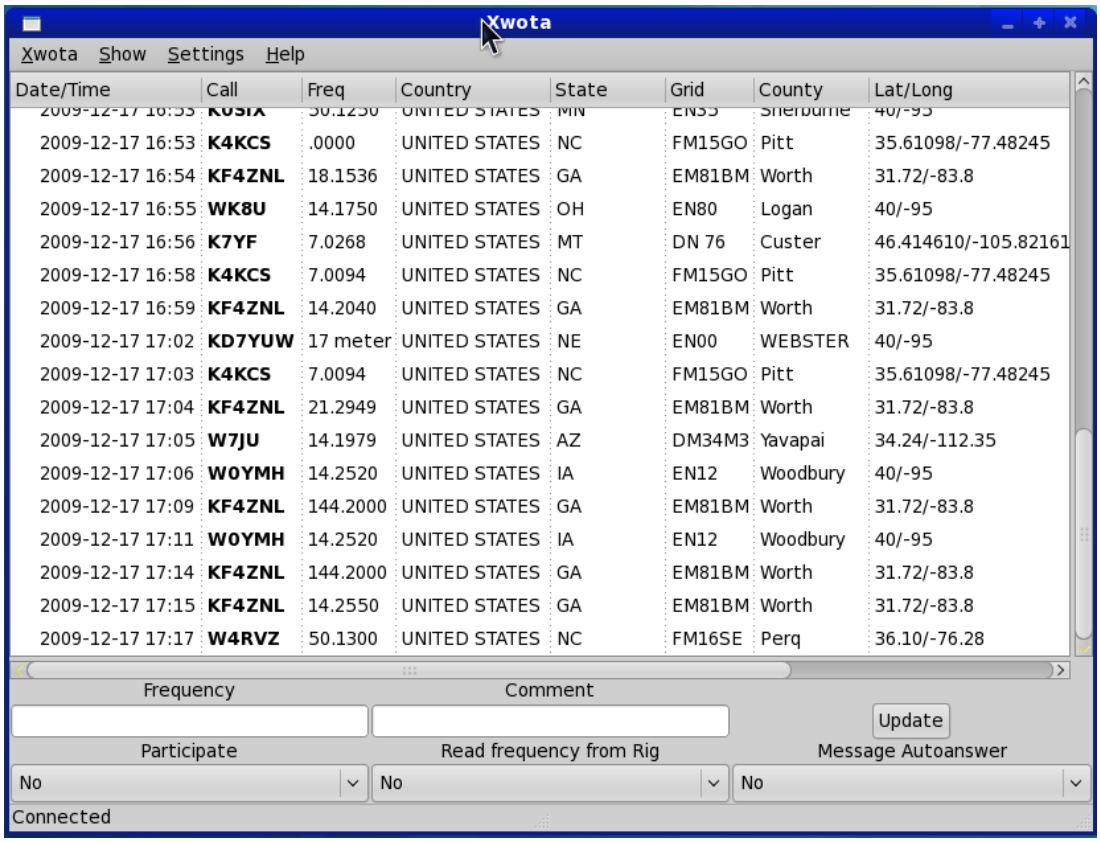

Figure 38. Who is on the air

The user may also query the database by selecting **Query** from the **Show** menu. A dialog will appear allowing the user to enter specific location, frequency, and/or call to be searched for. Clicking **Send** will then cause the results to be returned at the bottom of the main window.

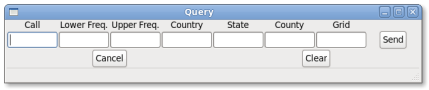

Figure 39. Xwota query window

The user may also enter his own report. Before doing this, station information should be entered by selecting **Station Info** from the **Settings** menu and filling in the dialog:

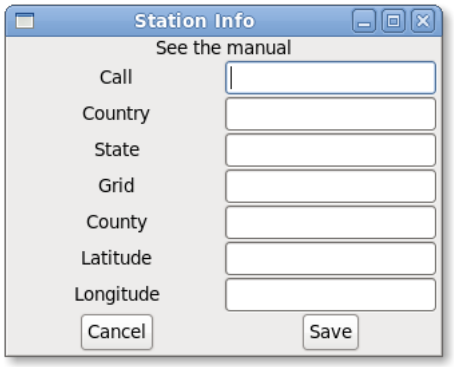

Figure 40. Station Info window

The user may then enter the frequency and optionally a comment in the main window, and click the **Update** button to cause the report to be sent to the database.

A. Installing Software on Fedora

There are basically two ways to install software from the Fedora repositories; from the GUI using *PackageKit* and from the command line using *yum*. Because the *yum* approach is simpler, throughout this document we describe that method. However, there are a number of details with respect to *yum* that are useful to know, and some users feel more comfortable with a graphical user interface. Hence, this appendix.

A.1. Installing Software with the GUI

Launch the *PackageKit* application by selecting **Administration->Add/Remove Software** from the **System** menu:

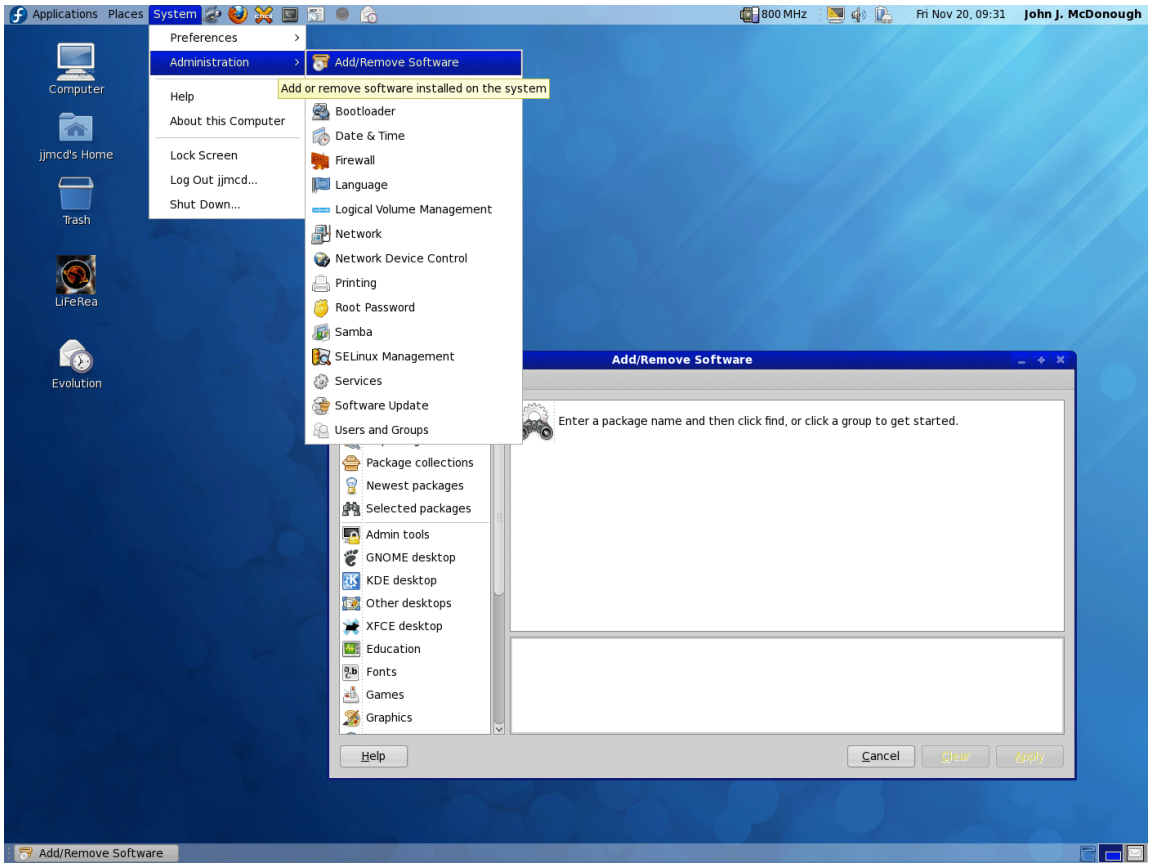

Figure A.1. Launching PackageKit

There may be a delay before the controls on the window that appears may be used. This delay may be brief or lengthy, depending on what you have done earlier. The data used by PackageKit is cached, and if it is stale, new data will be downloaded, which may take several minutes.

Type in the name of the package you wish to install in the text box at the upper left, and select the **Find** button. There may be a delay, and you may need to click the **Find** button a second time.

The package (perhaps with several others) will appear to the right. The closed box indicates that the package is not installed on your system.

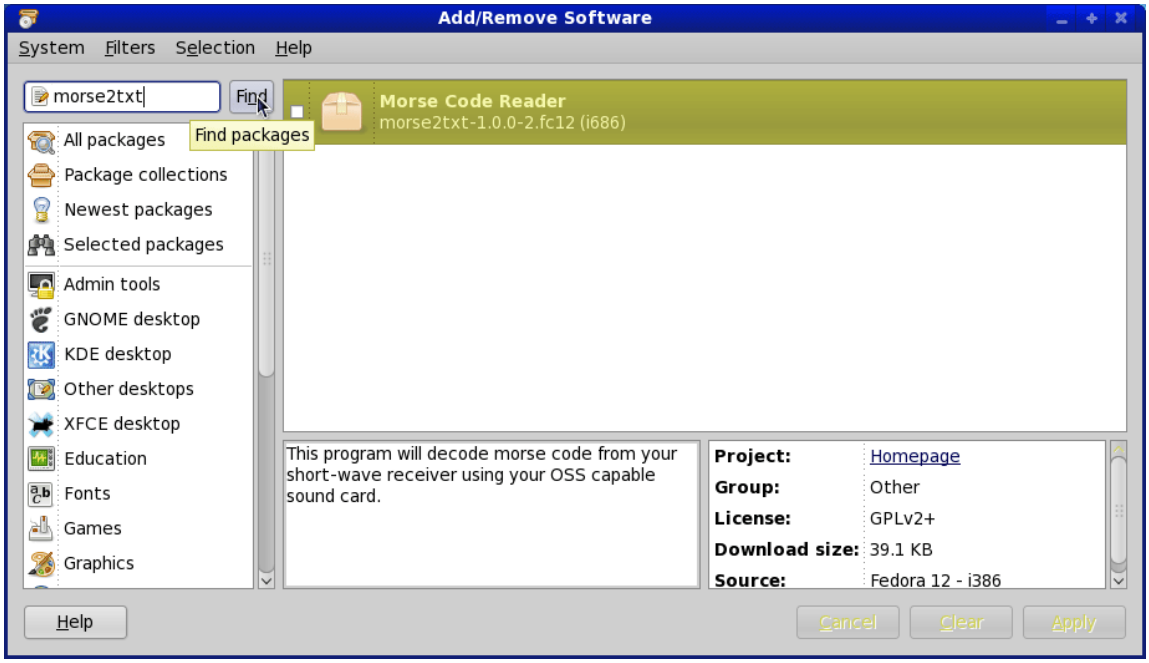

Figure A.2. Searching for Software

Click on the check box next to the package you wish to install. A blue plus sign will appear over the box indicating that is has been selected to install.

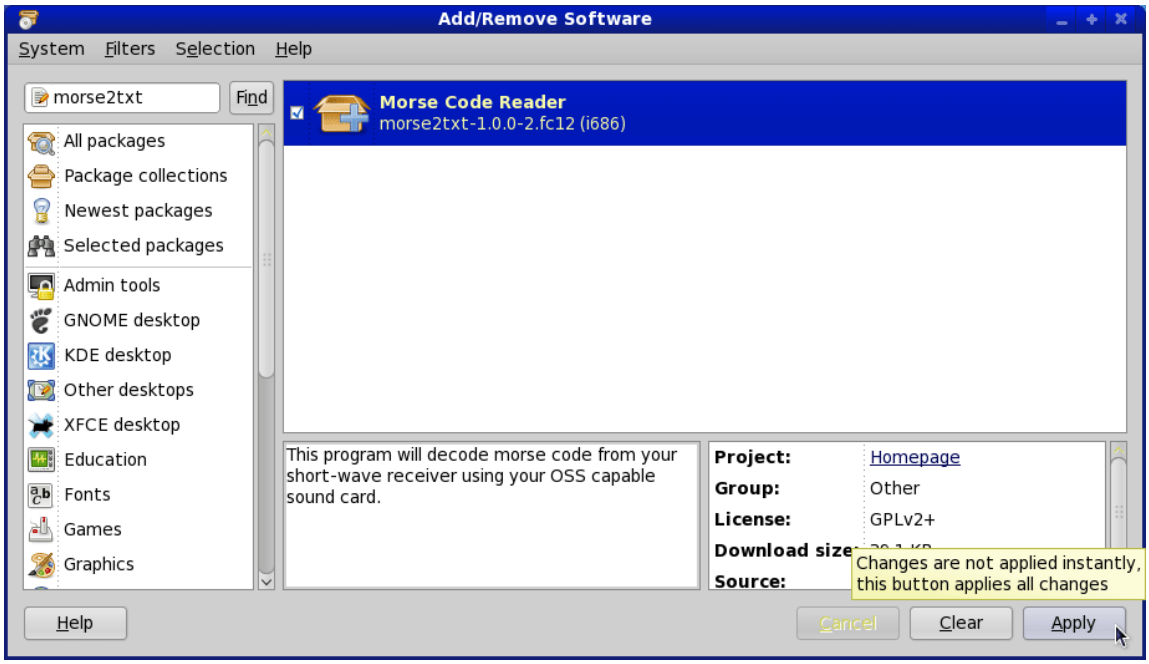

Figure A.3. Selecting a package to install

You may repeat the process if you wish to install additional packages. When you have selected the packages you wish, click the **Apply** button at the lower right to install the packages.

Depending on how your system has been configured, you may be prompted to enter the password for the administrative user. If the package is unsigned, or is from a repository you have not used before, you may be prompted for this password again. As a general rule, only administrators may install software.

Figure A.4. Authorizing the installation

Enter the root password and click **Authenticate**.

The package will then be installed. When the installation is complete, the package wll be shown as an open box, indicating that the package is installed. For some applications, the application's icon may be displayed instead of the open box. The **Apply** button will be disabled, indicating that there are no pending actions to apply.

Figure A.5. Package is installed

The **Find** button searches not only the name of the package, but also the package description. Some common words may reveal several packages, some perhaps not those intended. For example, if you typed in "circuit" with the intent of finding circuit design applications, you may also find applications referring to wired communications circuits as well as racing circuits!

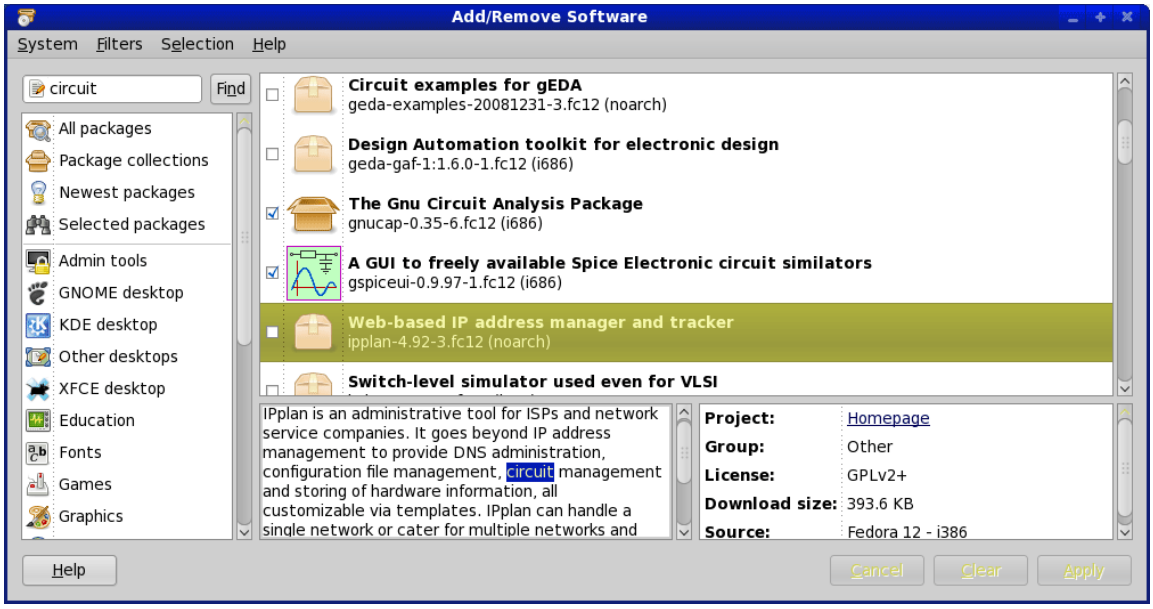

Figure A.6. Searching the description

Sometimes you may want to try several variations of a word or phrase. If, for example, you are looking for an SSTV application, trying "slowscan" or "slow scan" will fail:

Figure A.7. Searching the description - failure

The developer of the slow scan application used the word "slow-scan" in his description. *PackageKit* is not smart enough to guess what you meant or what the original developer was thinking.

Figure A.8. Searching the description - success

A.2. Installing Software with yum

Unlike PackageKit, where the system administrator may choose to allow non-administrative users to install software, yum requires administrator authentication. There are three ways this can be done:

- You may switch to the root user with the **su** command:

```
[jjmcd@Cimbaoth ~]$ su -
Password:
[root@Cimbaoth ~]# yum install xastir
Loaded plugins: presto, refresh-packagekit
  ...
```

This is the least desireable method. You must enter the root password, and you can easily forget that you are operating as root. As the root user you can do unlimited damage.

- You may use the **su** command with the **-c** switch. This allows you to enter the single **yum** command as root, but immediately switches back to your normal user:

```
[jjmcd@Cimbaoth ~]$ su - 'yum install fldigi'
Password:
Loaded plugins: presto, refresh-packagekit
  ...
```

  Notice that you must surround the command with quotation marks or apostrophes. This still requires you to type the password, and is somewhat more annoying to type, but does not leave you as root ready to do damage.

- If the adminstrator has set you up in the **sudoers** file, you may use the **sudo** command:

```
[jjmcd@Cimbaoth ~]$ sudo yum install wxapt
Loaded plugins: presto, refresh-packagekit
  ...
```

  This has several advantages; you don't need to type the password, you are not left in a dangerous position, and if desired, the administrator can limit you to a select set of commands so you do not inadvertently cause damage.

  Because this is the preferred approach, the examples in this guide use this method. However, it does require setup ahead of time.

yum may determine that additional packages must be installed. **yum** will list these packages and calculate the total size of the download. It will then ask you whether you want to actually download and install this package or group of packages:

```
[jjmcd@Cimbaoth ~]$ sudo yum install trustedqsl
Loaded plugins: presto, refresh-packagekit
Setting up and reading Presto delta metadata
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package trustedqsl.i386 0:1.11-3.fc10 set to be updated
--> Processing Dependency: tqsllib >= 1.2 for package: trustedqsl-1.11-3.fc10.i386
--> Processing Dependency: libtqsllib.so.1 for package: trustedqsl-1.11-3.fc10.i386
--> Running transaction check
---> Package tqsllib.i386 0:2.0-5.fc10 set to be updated
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package             Arch          Version               Repository        Size
================================================================================
Installing:
 trustedqsl          i386          1.11-3.fc10           updates          557 k
Installing for dependencies:
 tqsllib             i386          2.0-5.fc10            updates          167 k

Transaction Summary
================================================================================
Install      2 Package(s)
Update       0 Package(s)
Remove       0 Package(s)
```

```
Total download size: 723 k
Is this ok [y/N]:
```

Answer **y** or **N** depending on whether you want to download and install the group of packages.

A.2.1. Searching for Software

yum gives you a number of choices for locating software you desire. To find information about a package you do not need to provide credentials. Any user may look up information about a package. You may search for specific words in the description using **yum search**:

```
[jjmcd@Cimbaoth ~]$ yum search APRS
Loaded plugins: presto, refresh-packagekit
Setting up and reading Presto delta metadata
================================ Matched: APRS ================================
aprsd.i386 : Internet gateway and client access to amateur radio APRS packet
           : data
xastir.i386 : Amateur Station Tracking and Reporting system for amateur radio
[jjmcd@Cimbaoth ~]$
```

yum will return the names of any package with the specified phrase in its description, and a short description. You may get a more detailed description of the package with the **yum info** command:

```
[jjmcd@Cimbaoth ~]$ yum info xastir
Loaded plugins: presto, refresh-packagekit
Setting up and reading Presto delta metadata
Installed Packages
Name       : xastir
Arch       : i386
Version    : 1.9.4
Release    : 5.fc10
Size       : 4.0 M
Repo       : installed
Summary    : Amateur Station Tracking and Reporting system for amateur radio
URL        : http://www.xastir.org
License    : GPLv2+
Description: Xastir is a graphical application that interfaces HAM radio
           : and internet access to realtime mapping software.
           :
           : Install XASTIR if you are interested in APRS(tm) and HAM radio
           : software.

[jjmcd@Cimbaoth ~]$
```

Notice that **yum** also tells you whether the package is installed. Yum also gives you the address of the upstream website so you may learn more about the package before installing it.

B. Revision History

Revision 0.9 **November 9, 2010** **John McDonough**
jjmcd@fedoraproject.org

- Documentation for dxcc
- Documentation for gresistor
- Documentation for callgit
- Correct typo in colrdx

Correct typos in xwota

Revision 0.8 **November 7, 2010** **John McDonough** *jjmcd@fedoraproject.org*

Documentation for rcrpanel
Documentation for colrdx
Documentation for xconvers
Documentation for xdx
Documentation for xastir
Documentation for gEDA
Documentation for gerbv
Documentation for pcb
Documentation for xwota

Revision 0.7 **November 20, 2009** **John McDonough** *jjmcd@fedoraproject.org*

Installing Software appendix

Revision 0.7 **October 31, 2009** **Eric Christensen** *sparks@fedoraproject.org*

Added xlog installation and setup procedures.

Revision 0.6 **October 29, 2009** **John McDonough** *jjmcd@fedoraproject.org*

ibp

Revision 0.5 **October 29, 2009** **Randy Berry** *randyn3lrx@gmail.com*

cwirc

Revision 0.4 **October 28, 2009** **Eric Christensen** *sparks@fedoraproject.org*

xlog screenshot
Additional info on fldigi

Revision 0.3 **October 6, 2009** **John McDonough** *jjmcd@fedoraproject.org*

xgridloc

Revision 0.2 **October 4, 2009** **John McDonough** *jjmcd@fedoraproject.org*

Beginning of SPLAT! instructions

Revision 0.1 **October 1, 2009** **John McDonough**
jjmcd@fedoraproject.org

Document skeleton