

TXAMADRM - A LINUX PROGRAM FOR SENDING DIGITAL SSTV

Ties Bos - PA0MBO

December 20th, 2013

Abstract

A linux program was developed for the transmission of digital SSTV pictures conforming to the HAM DRM standard as developed by HB9TLK. The program is a modification of the Dream software that was developed by Alexander Kurpiers, Volker Fischer, Stephane Fillod, Andrea Russo, Julian Cable and Doyle Richard [1]. It generally follows the original modifications by Cesco (HB9TLK). Burst error protection is effected by Reed-Solomon coding using the rsbep-program developed by Phil Karn (KA9Q) [2]

Keywords: Digital SSTV - Hamradio - DRM - Linux Reed-Solomon coding

1 INTRODUCTION

The program **txamadr**m can be used to transmit (picture) files in the HAMDREAM standard as developed by HB9TLK [3]. In contrast to the programs based on HB9TLK's code like WINDRM [4], EASYPAL [5] and DIGTRX [6], **txamadr**m runs under Linux and its source code is available. It is published under the GNU General Public License (GPL), so you can freely modify and improve it.

The DRM-signal generating routines in the Dream software package version 1.12b as downloaded from sourceforge.net were adapted to the HAM DRM standard. The modified package can be configured and compiled to give a standalone executable called **drm**. All user interfacing from the original package was bypassed.

The program **drm** can be run standalone in a command terminal and accepts the name of the files to be transmitted in the DRM stan-

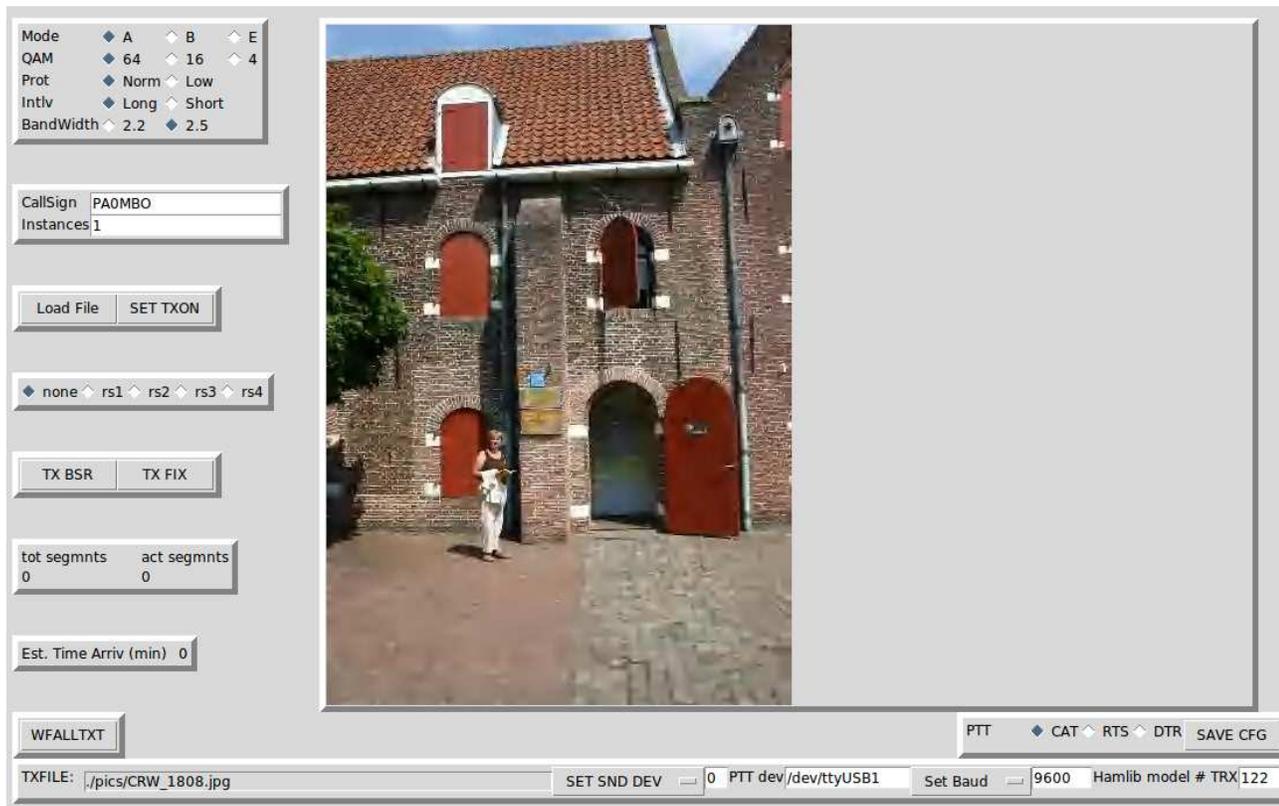


Figure 1: GUI with picture loaded

dard from the commandline and uses the content of a configuration file to obtain the mode settings for the transmission.

A Graphical User Interface (GUI) was developed as a separate program in the scripting language Tk/Tcl and provides facilities for selection of the file to be transmitted, the drm mode settings, transmission time estimate and PTT-operation of the TRX using hamlib CAT control. Communication between the GUI and the program **drm** is via standard I/O and the configuration file. The GUI allows for sending the files with more or less redundancy to obtain burst error protection. To this end separate Reed/Solomon coding programs were developed that can be invoked from the GUI. For compatibility with the Windows program EasyPal a bad segment request/fix system is incorporated. Figure 1 shows the user interface after a picture to be sent was loaded.

2 DESCRIPTION OF THE PROGRAMS

2.1 drm - DRM signal generation

The heart of the software is the Dream software version 1.12b as obtained from sourceforge.net. Its main routine was changed to start in tx mode and a number of files were modified to conform to the ham

drm standard. The main routine reads the file **txamadrn.ini** from the \$HOME directory to obtain the settings for the callsign, the sound device to be used and all mode parameters. When using **txamadrn** for the first time, this file will not yet be present and a version from the distribution files will be used. The **txamadrn.ini** file is plain ascii and can be setup by any plain text editor. If using the graphical user interface it is written automatically using the settings of the radio buttons.

As no use is made of the user interface code of Dream, Qt is not needed. The same goes for the libraries faad2 and faac as the modified code does not implement the drm transmission of speech. The output of the soundcard audio signal is programmed using the alsa system. Dream uses the Qt thread routines, but as the use of Qt in txamadrn was not wanted, threading was realized with the pthread library.

2.2 txamadrn.tcl - the GUI

The graphical frontend **txamadrn.tcl** automates a number of the steps needed in the standalone use of **drm**. The program starts with loading the required packages Tk, Expect and Img. Then follow the definition of a number of procedures. WriteIni writes the configuration file **txamadrn.ini** from which a number of settings can be retained from session to session. The procedure **rsencode** sees to it that the file to be sent will be coded with the correct Reed/Solomon code chosen by radiobuttons in the GUI. Next the main window is defined followed by the setting of all variables from the configuration file **txamadrn.ini**. The rest of the code is a straight forward setting up of the GUI.

Worth mentioning is that in the command for the button "Load File" a rough estimate of the time of arrival of the file is calculated based on the mode parameters.

The real work is started in the command of the button "SET TXON" where the PTT is operated by a CAT command via hamlibs' **rigctl**, the **txamadrn.ini** file is written with the current values of all the important GUI settings, the file to be sent is Reed/Solomon coded if wanted and the separate program **drm** is spawned. Some feedback from **drm**, i.e. the total number of segments to be sent and the number of the segment that is actually being transmitted, is shown in the GUI.

2.3 rs1encode, rs2encode, etc. - Reed/Solomon encoding

The Reed/Solomon encoding programs **rs1encode**, **rs2encode**, **rs3encode** and **rs4encode** use routines developed by Phil Karn (KA9Q) [2].

Table 1 shows the parameters for the four Reed/Solomon decoders.

Table 1: Reed/Solomon parameters

type	block length	data length	increase tx time %
rs1	255	224	14
rs2	255	192	33
rs3	255	160	60
rs4	255	128	100

2.4 wfaltxt

The generation of waterfall text is provided by the program **wfaltxt**. The main GUI has a button labeled "WFALLTXT", pressing it provides the user with a new window having buttons to generate canned messages and an text editing window that can hold three lines of 10 characters each. It has a "SET TXON" command button that generates an ASCII-file called **wfal.txt** and spawns the program **wfaltxt** which does the actual sending. **wfaltxt** generates the sound that eventually shows up as text in the receiving stations waterfall. Its crude font only has the upper case alphabetic characters, the numbers 0..9 and the slash. The fonts are constructed from a 10x10 dot matrix and it will not be too difficult to design a more elegant font and/or more characters yourself. The source code is also in the directory. The "WFALLTXT" window also has a button that generates the three pilot frequencies and operates PTT.

2.5 bsr /fix

As of version 0.9 **txamadr**m can handle bad segment request files (**bsr.bin**) received by **rxamadr**m and sent by the other stations in case they failed to decode a picture when it missed too many segments. When the "TX FIX" button is operated, the GUI spawns the main program **drm** which reads this bsr.bin file from the directory **/trxamadr/m/linux/bsrfix** where it was placed by **rxamadr**m upon

its receipt and uses it to transmit only the requested segments of the last file it has sent.

The GUI **txamadm.tcl** also handles the situation when **rxamadm** has received an incomplete file that it cannot decode. In this situation **rxamadm** places a `bsr.bin` file in the directory `~/trxamadm/linux/pics`. The button "TX BSR" on the GUI will transmit this file on the air. Before tx-ing it resets all rs-encoding, as this file should be sent unencoded.

2.6 hybrid mode

Starting with version v3.4 **trxamadm** now also provides sending in the so-called "Hybrid" mode using a slight modification of the main executable **drm**. This executable has the name **hybrid** and repeatedly sends a header segment containing the name of the picture file which itself is uploaded to a fileserver. The code for the upload by ftp is in the script **txamadm.tcl**. Both the ftp upload as well as the transmission of the filename by radio is activated by a single Tk command button labeled **TX HYBRID**.

3 INSTALLATION

TXAMADRM now is part of the archive **trxamadrmv3.4.tgz** For installation instructions see the document **trxamadm.pdf**.

4 OPERATION

4.1 Preliminaries

The collection of pictures or text files that you are planning to transmit should be transferred to the directory:

```
~/trxamadrmv3.4/txamadm/linux/pics
```

In principle any type of file can be sent with **txamadm**, but if you want your pictures to be shown in the right hand side pane of **txamadm.tcl** they should be of the types supported by the **Img** package of Tk, i.e. jpeg, bmp, gif, ico, pcx, pixmap, png, ppm, ps, sgi, sun, tga, tiff, xbm and xpm. Files with the extension.txt will be shown in text mode. Unsupported file types popup an error screen for the fact that they cannot be displayed, but ignoring this error by clicking on OK in the error screen will nevertheless allow sending the file.

Picture files should be converted to a reasonable size to keep their sending time in check (20 - 30 kB).

To be able to use CAT commands to operate the PTT of your transmitter you need to know the hamlib model number of your TRX. If hamlib is installed correctly this number can be found by giving the command:

```
rigctl -l
```

in a command terminal.

Test the correct working of the CAT PTT in a command terminal with `rigctl -m <modelnr> -r <devptt> -s <catbaud> T 1`

this should switch your trx to TX, whereas

```
rigctl -m <modelnr> -r <devptt> -s <catbaud> T 0
```

should switch the TX off. The `< modelnr >` etc. are the parameters you have to substitute in the given commands.

4.2 Sending pictures

Assuming that the GUI `txamadm` has been started as described in the document `trxamdrm.pdf` you should now work through this checklist to prepare txamadm:

1. Make a choice for PTT control between CAT / RTS / DTR by clicking the right radiobutton
2. If using CAT type the hamlib model number of your transceiver in the bottom right entry called "**Hamlib model # TRX**".
3. If using CAT use the pull down menu "**SET CAT Baud**" and make the choice that corresponds to the setting of your transceiver.
4. Use the "**SET SOUND DEVICE**" pull down menu to set the alsa device number of the soundcard that is used to drive the transmitter with audio.
5. Change the entry "**CallSign**" at the left handside of the GUI to your own callsign.
6. Fill the entry with the label "**CAT dev**" with the device code of the serial port of your computer that you have connected to the CAT-interface port of your transceiver, i.e. `/dev/ttyS0`. For USB to serial converters use `/dev/ttyUSB0` or higher numbers if you have more than one. If using DTR or RTS give the name of the serial device you want to use for PTT in the entrybox that is now called **PTT dev**.
7. Press the button "SAVE CFG"

Now you have to make your choices for the radiobuttons at the upper left of the GUI. These buttons set the sending parameters. Successful reception of the files you send depends very much on the right

choice of the sending parameters. Mode A can only be used on very good quality channels. On hf mode B or E generally are used either in QAM-16 or QAM-4 when the going gets tough. The modes A, B and E differ in the distance and number of carriers placed in the bandwidth of 2.2 or 2.5 kHz. With more carriers the data will be transferred faster, but QRM will impair the transmission much more. The same goes for the number of digital states the carrier can be in (QAM64, QAM16, QAM4). Table 2 shows the influence of these parameters. The software provides an estimated transmission time for the some combinations of transmission parameters. The influence of bandwidth, protection level and interleaving is neglected in the calculation of this estimate.

Table 2: Transmission times for various modes

mode	BW kHz	QAM	carriers	est. tx time m:s	actual tx time m:s
A	2.5	64	57	0:20	0:25
A	2.5	16	57	0:20	0:25
A	2.5	4	57	0:41	0:50
A	2.2	64	53	0:20	0:25
B	2.5	64	51	0:20	0:27
B	2.5	16	51	0:41	0:53
B	2.5	4	51	1:02	1:15
B	2.2	64	45	1:02	1:17
E	2.5	64	31	0:41	0:49
E	2.5	16	31	0:41	0:53
E	2.5	4	31	1:23	1:40
E	2.2	64 * 29	1:23	1:45	

On the hf bands under good conditions mode B with QAM4 is a good choice, but generally I use mode E with QAM4. My bandwidth of choice is 2.5 kHz, except in cases when there is qrm of stations at the edges of the passband, then I reduce the bandwidth of the SSB-filter on my transceiver and use the 2.2 kHz setting.

When conditions are very good it is sometimes possible to use B/16 (S/N \geq 12 dB).

In heavy qsb or qrm situations burst error protection via Reed Solomon coding can be switched on by checking one of the rs-buttons before hitting the TX-ON button. The higher the rs-number the more protection, but the longer it takes to transmit a file (see 1).

When all these choices have been made you are ready to load the file to be transmitted by clicking the **Load File** button and pick the right filename and filetype in the window that now pops up. Finally hit the **SET TXON button** and watch the actual segment numbers being sent. When the name of the **SET TXOFF** button changes back to **SET TXON** the work is done. If you want to abort a transmission just hit the **SET TXOFF** button.

The settings in the GUI are stored in a configuration file called **txamadrn.ini** when the "SET TXON" button is hit and will be retained between different sessions with TXAMADRM.

For good results it is very important not to overdrive your transmitter. The audio out level cannot be set from within the GUI. You have to use the program **alsamixer** to see to it that the ALC-level during transmission is very moderate.

4.3 Sending text files

The only difference of sending text files instead of picture files consists of loading the correct text file. When a file with the extension .txt is loaded its content will be shown in the GUI and can be edited there on the spot. Don't forget to click the "SAVE TXT" button before hitting the "SET TXON" button otherwise the original content will be transmitted. The text file will be transmitted repeatedly until the "SET TXOFF" button is clicked. Rsendcoding will be switched off, as text-files are always sent in the clear.

Having a number of "canned" messages as txt-files in the **pics** sub-directory will be very helpful. The files **stndescr.txt**, **73pa0mbo.txt**, etc. can serve as examples to prepare your own often used texts. Give them some meaningful names with the .txt-extension so that you can find them easily using the "Load File" button in the GUI and changing the "Files of type" entry from the window that opens to "Text Files (*.txt)".

4.4 Waterfall text

You can use the **WFALLTXT** button to send a signal that shows up as text in the receiving stations' waterfall display, i.e. to acknowledge the successful receipt of a picture, to announce the start of your transmission, etc. Left clicking on this button opens a text window that can hold three lines of 10 characters each. You can enter the wanted text (only uppercase letters, the numbers 0 .. 9 and the slash) from

your keyboard. There are some buttons for precanned messages, i.e. for signing off with 73 and your callsign, with the text "GUD CPY" followed by your callsign, etc.

Pressing the **SET TXON** button in this window puts your set in the TX-mode and starts the transmission. When all text has been sent the window disappears and the normal GUI unfreezes. As of version **txamadrnv11** the waterfall text window has a button labeled "TUNE". Pressing this button will transmit the three pilot tones for a short time.

4.5 BSR / FIX

The button "TX BSR" loads the bsr.bin file created by **rxamadrnv** upon incomplete reception of a (picture) file and automatically starts its transmission. The file will be transmitted 6 times or until you hit "SET TXOFF". It is common to send these short bsr.bin files at least several times and it is up to you to decide how long to continue this transmission. To make sure it will come across, send it at least 4 times. You can count this by watching the rollover of the "act segmnts" counter. If left alone the bsr.bin file will be sent 6 times.

If you successfully receive a bad segment request from another station after you have sent a picture you can repair his incomplete file by pressing "TX FIX". This will transmit his missing segments plus some overhead. Don't use this button if the received bsr.bin file regarding you last picture transmission was not complete. In that case you should retransmit the last picture completely.

4.6 Sending pictures in hybrid mode

To be able to use hybrid mode you and the station(s) you want to qso with have to agree upon which files server to use. So one of you has to provide the files server facility and its access data (ip-address or internet name, username and password and the directory name). This data should be entered in the corresponding fields of the form that pops up when pressing the "HYBRID" button in the receiving programs RXAMADRM window.

To be able to qso with EasyPal users you should enter the access data for EasyPals Hybrid files server (ask VK4AES).

After loading a picture file sending in hybrid mode is started by pressing the "TX HYBRID" mode. You will notice that it takes some time before the PTT of the transceiver is activated: the scripts has to wait for the transfer of the file to the files server by ftp. Generally the mode E and 4-QAM are used in these transmission as they do not

take much time.

5 RESULTS

5.1 Computer to computer testing

During the development of **txamadr**m the audio signal generated was received by **rxamadr**m on a second computer. The audio line output of the soundcard in the computer running **txamadr**m was connected to the microphone input of the Linux system running **rxamadr**m. All DRM-modes A,B and E with various combinations of interleave (long or short), protection (Normal or Low) and all possible QAM-modes were tested. SNR was over 38 dB in this set up. Transmitted files were jpg's. Under all circumstances the reception was without missing segments.

Testing with the popular windows program Easy-Pal on the receiving end produced similar results, also with Reed-Solomon coding switched on.

6 DISCUSSION

The main reason to develop **txamadr**m was to provide a sending companion program to **rxamadr**m. Together this set of programs offers SSTV drm transmission and receiving under Linux.

This set of programs is compatible with the now very popular windows program EasyPal.

7 ACKNOWLEDGEMENT

The author is very much indebted to David Ranch (KI6ZHD). Without Davids help **txamadr**m would still be a barebones DSP program for sending DRM SSTV. Its current features, i.e. the estimation of the sending time of a file, CAT control of PTT via hamlib and the setting of the sound device via the GUI were all his ideas. Moreover he also tested **txamadr**m's features extensively. Bugs remaining are of course my responsibility.

References

- [1] Volker Fischer et.al., <http://www.sourceforge.net/projects/drm>.
- [2] Phil Karn, <http://www.ka9q.net>.

- [3] Cesco HB9TLK, <http://www.qslnet.de/member/hb9tlk>.
- [4] <http://n1su.com/windrm>.
- [5] <http://www.vk3evl.com>.
- [6] <http://www.qsl.net/py4zbz/index.htm>.
- [7] Frank Simpleton (VK3FCS) and Stéphane Fillod (F8CFS),
<http://sourceforge.net/projects/hamlib>.