(RTTY) Contesting on GNU/Linux with Win-Test

Serge Stroobandt, ON4AA

Introduction

RTTY is a great mode for contesting. There is no need to type a lot, nor do you need to shout your voice hoarse. What is more, you even do not need to listen to RTTY stations in order to work them. These characteristics render late-night RTTY contesting especially gentle towards family members and nearby neighbours. The required signal-to-noise ratio lies between a moderately low -5 to -9 dB @ 2500 Hz; in any case, much lower than the +10 dB for SSB!

However, getting set for RTTY contesting under GNU/Linux is kind of challenging.

The name may sound elusive, but Win-Test happens to be the only full-featured contest logging program offering some support for running on Linux, albeit under Wine and at the price of a licensing fee. The number of supported contests is fair, but could be better. Minor updates are free of charge for an unlimited period of time. Here are the release notes.

It was Ulf Schneid, DK5TX, who made me aware of this by courteously sharing his story on the net, accompanied by Wine instructions. Here, a slightly different set-up will be presented, as I prefer the use of PlayOnLinux — a front-end for Wine — over using bare metal Wine. PlayOnLinux offers the benefit of installing unrelated Windows™ programs on separate virtual drives, also called «bottles» in CrossOver, a commercial offering similar to the free (as in beer) PlayOnLinux. The use of virtual PlayOnLinux drives keeps things uncluttered and limits the damage when any Windows™ program goes awry.

microHAM Digi Keyer

If you own a modern transceiver with a USB port which presents audio as a sound device, you may skip this part.

Many transceiver interfaces work with GNU/Linux. However, as with all hardware under GNU/Linux, always check for Linux support on the web before acquiring a new interface! Because they are quite popular and I had one lying around, this article will focus on the excellent microHAM interface products.

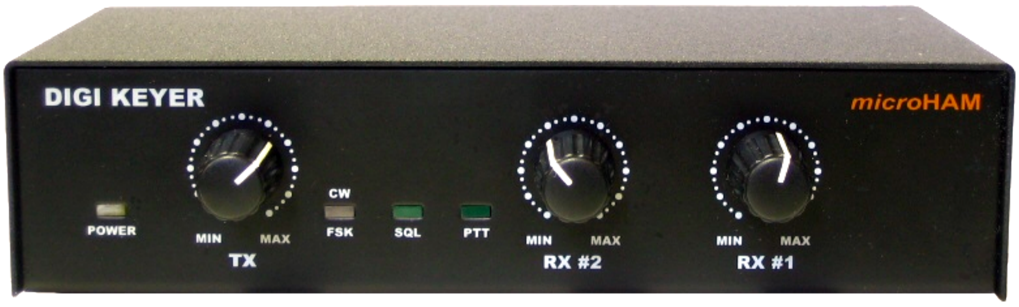


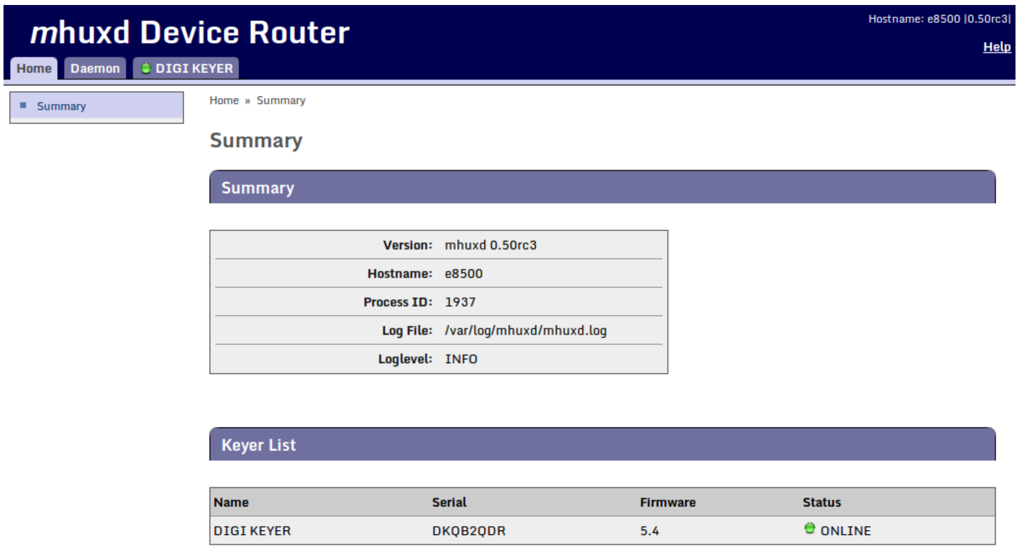
Figure 1: Front panel of the microHAM Digi Keyer

The good news is Matthias (Max) Moeller, DJ5QV, was so kind to code and share the `mhuxd` microHAM Unix daemon. In doing so, he could count with the help of Joe Subich, W4TV of microHAM for providing protocol specifications. The daemon should support all microHAM interface products.

On Ubuntu-derived systems, it is best to install `mhuxd` by adding its PPA repository. This will ensure updates are installed automatically when they become available. Type the following commands (without the `$` prompt) in a terminal window:

```
$ sudo add-apt-repository ppa:dj5qv/mhuxd-0.5
$ sudo apt-get update
$ sudo apt-get install mhuxd
```

The `mhuxd` daemon should immediately start running in the background. You can check that and perform the initial configuration by opening the following URL in your browser: `http://localhost:5052` Bookmark this URL as you may need it again occasionally. Note that microHAM interfaces are powered by the transceiver; not the computer.



Summary

Summary

Version:	mhuxd 0.50rc3
Hostname:	e8500
Process ID:	1937
Log File:	/var/log/mhuxd/mhuxd.log
Loglevel:	INFO

Keyer List

Name	Serial	Firmware	Status
DIGI KEYER	DKQB2QDR	5.4	● ONLINE

Figure 2: Summary screen of the `mhuxd` daemon available at `http://localhost:5052`, showing the keyer's firmware version.

Caveat: Upgrade to the latest firmware!

Once `mhuxd` is installed, first, **make absolutely sure your microHAM keyer is running the latest firmware!** Failing to do so, guarantees problems with MMTTY PTT keying over the FSK port —as I experienced first-hand.

The firmware for the micro/Digi/CW keyers (MK/CK/DK) goes under the name of `mmk`. Hidden deep within its change file `mmk_change_log.txt`, the November 22, 2007 entry reads:

«Some changes in keyer protocol and settings format was introduced to be compatible with router v5.1.0. Hence firmware MUST be upgraded and power-up settings MUST be again stored to EEPROM of keyer.»

Installing new keyer firmware is the one and only instance where you are probably quicker off borrowing a Windows™ computer. In less than 10 minutes, you get the job done. Just install the Windows™ microHAM USB Device Router and let this program upgrade your keyer to the latest version of the appropriate firmware. Keep your transceiver and cabling at hand too, because —remember— the keyer gets its power from the transceiver.

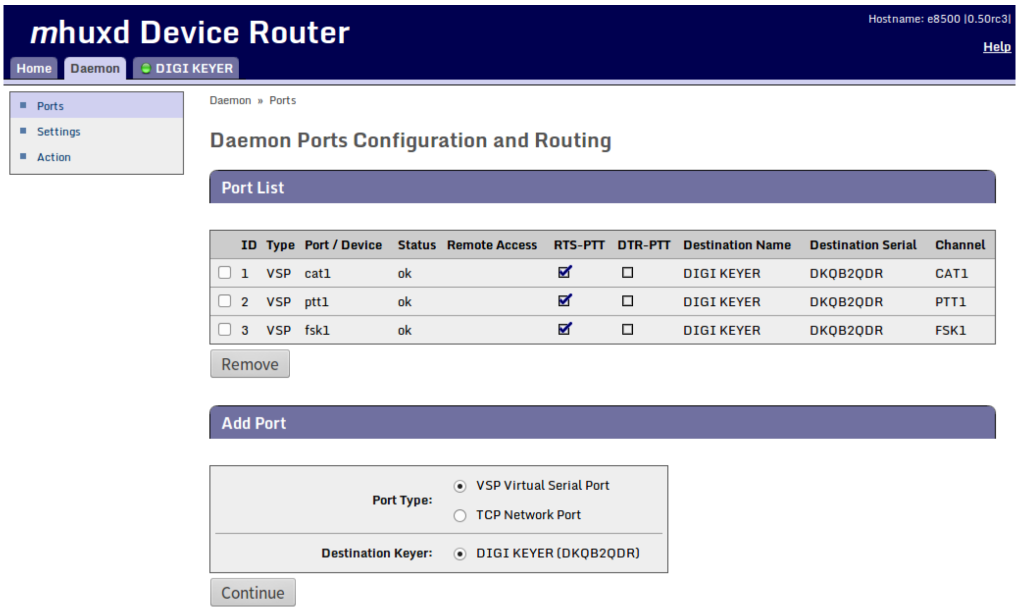
microHAM port routing

In this step, the CAT, PTT and FSK channels of the microHAM keyer will be assigned to a Linux device virtual serial port. Also make sure to tick off RTS-PTT for each of these virtual serial ports.

Here is why: MMTTY is only able to employ a single COM port for both PTT T/R and FSK tone keying. Instead of using the dedicated PTT1 port, MMTTY will need to key PTT with a RTS (ready to send) signal on the FSK1 port. (See below.) It is a good idea to also configure RTS-PTT for the other virtual serial ports for use with other programs.

Table 1: Digi Keyer port routing

channel	RTS-PTT	virtual serial port
CAT1	☑	`/dev/mhuxd/cat1`
PTT1	☑	`/dev/mhuxd/ptt1`
FSK1	☑	`/dev/mhuxd/fsk1`



Daemon Ports Configuration and Routing

Port List

	ID	Type	Port / Device	Status	Remote Access	RTS-PTT	DTR-PTT	Destination Name	Destination Serial	Channel
☐	1	VSP	cat1	ok		☑	☐	DIGI KEYER	DKQB2QDR	CAT1
☐	2	VSP	ptt1	ok		☑	☐	DIGI KEYER	DKQB2QDR	PTT1
☐	3	VSP	fsk1	ok		☑	☐	DIGI KEYER	DKQB2QDR	FSK1

Add Port

Port Type: [● VSP Virtual Serial Port] ○ TCP Network Port

Destination Keyer: [● DIGI KEYER (DKQB2QDR)]

Figure 3: Configuration and routing of the `mhuxd` deamon virtual serial ports for the microHAM Digi Keyer

To make the `mhuxd` device ports accessible from your account, add your username to the `mhuxd` group. Log out for this change to take effect.

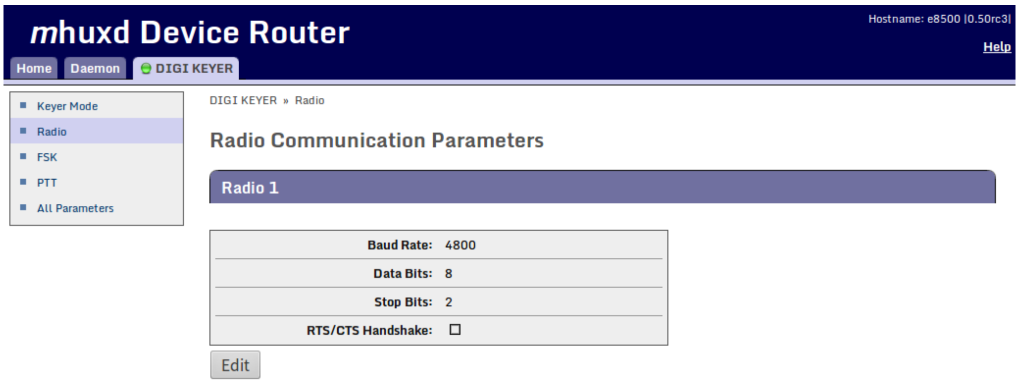
```
$ sudo adduser $USER mhuxd
```

In the rare event of a `mhuxd` daemon crash, there is no need to restart your computer. Typing the following command in a terminal window should do as long as your system is not using `systemd`.

```
$ sudo service mhuxd restart
```

Specifying the CAT protocol

The CAT1 port on the microHAM interface needs to be informed about the serial protocol parameters which are specific to your transceiver brand and model. Consult your transceiver manual for the proper settings. This CAT port will be used by Win-Test —not MMTTY— to read and control your transceiver.



Radio Communication Parameters

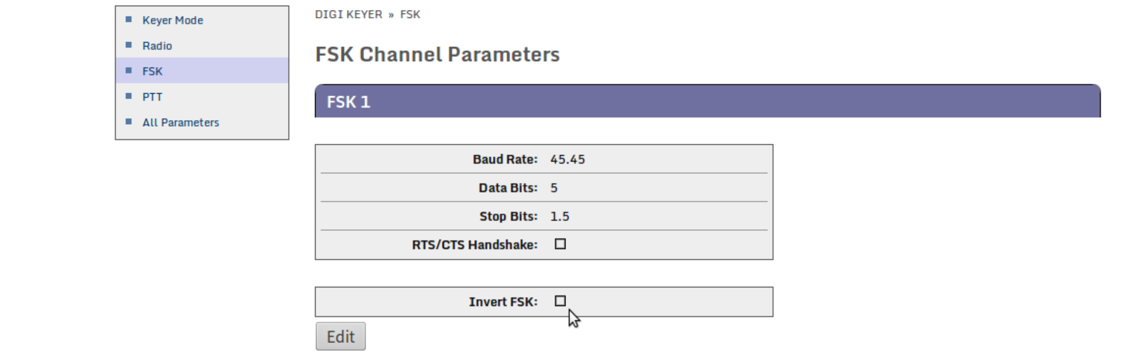
Radio 1

Baud Rate:	4800
Data Bits:	8
Stop Bits:	2
RTS/CTS Handshake:	☐

Figure 4: Computer-aided transceiver protocol parameters for interfacing with the Yaesu FT-990. Consult your transceiver manual for the proper settings.

FSK parameters

Standard FSK parameters for amateur radio TTY service are shown below. Other services (press, diplomatic, military) may require different parameters.

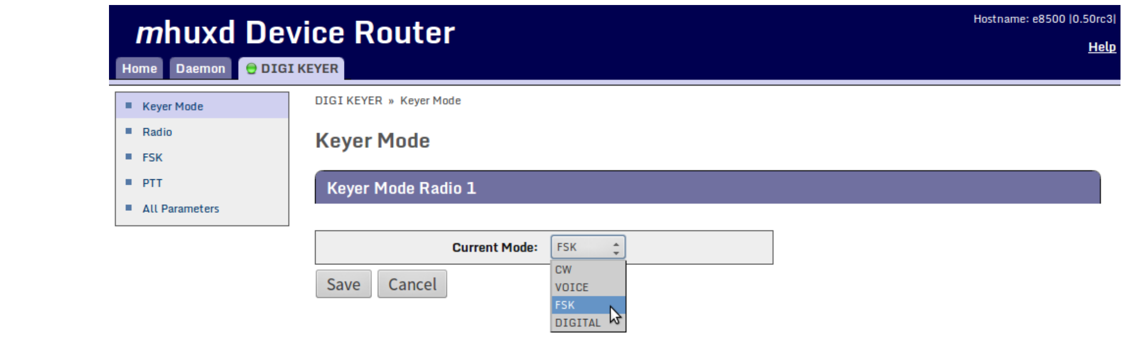


Baud Rate: 45.45
Data Bits: 5
Stop Bits: 1.5
RTS/CTS Handshake: ☐
Invert FSK: ☐

Figure 5: FSK parameter settings for RTTY, with an option to inverse the FSK stream on transmit.

Setting the keyer mode

Keyer mode is the only setting which requires changing each time one switches between operating modes (e.g. when switching from real `FSK` for MMTTY to `DIGITAL` AFSK for use with Fldigi). On newer microHAM interfaces (DK2, MK2 and MK2R+), it is more practical to opt for the `Keyer Mode Follows RIG` automation. This will eliminate once and for all the need to revisit this `mhuxd` configuration page.



Current Mode: FSK

Figure 6: Setting the keyer mode for real `FSK` keying with MMTTY.

Transceiver settings

- Whenever possible, always use truly keyed FSK on transmit. Because of confusing FT-990 CAT messaging, I am required to always operate my Yaesu FT-990 in `RTTY-USB` mode.
- To protect the transceiver PA from overheating, reduce the `RF PWR` to 50%.
- If one of the tones falls outside the (250 Hz) bandpass filter, use the transceiver's `SHIFT` knob to correct this.
- For fine tuning, employ `RX CLAR` and `TX CLAR` simultaneously on the FT-990.
- I prefer to listen to lower pitched "European" RTTY tones instead of the US default. On the FT-990, this is set using DIP switches.

Table 2: Mark and shift settings

style	MARK	shift	SPACE
US default	2125 Hz	170 Hz	2295 Hz
EU easy listening	1275 Hz	170 Hz	1445 Hz

Caveat: Reversed FSK on RX implies inversing FSK on TX!

My Yaesu FT-990 insists on designating `RTTY-LSB` as reversed or `RTTYR` in its CAT communication with my computer. This is cumbersome for logging QSOs on LoTW which does not know how to handle `RTTYR` as a mode. Hence, I operate this transceiver always in its `RTTY-USB` mode. This means Rev. on reception is always `on` in MMTTY **and** either my keyer or my transceiver is set to inverse the FSK transmission stream. I chose to do the latter by flipping a DIP switch on the FT-990. Remember: If nobody answers your calls, chances are you are sending in reverse!

PlayOnLinux

As mentioned earlier, PlayOnLinux is a neat front-end for Wine, offering the possibility to install unrelated Windows™ programs on segregated virtual drives. Being included with the Xubuntu LTS repositories, installing PlayOnLinux is straightforward:

```
$ sudo apt-get install playonlinux
```

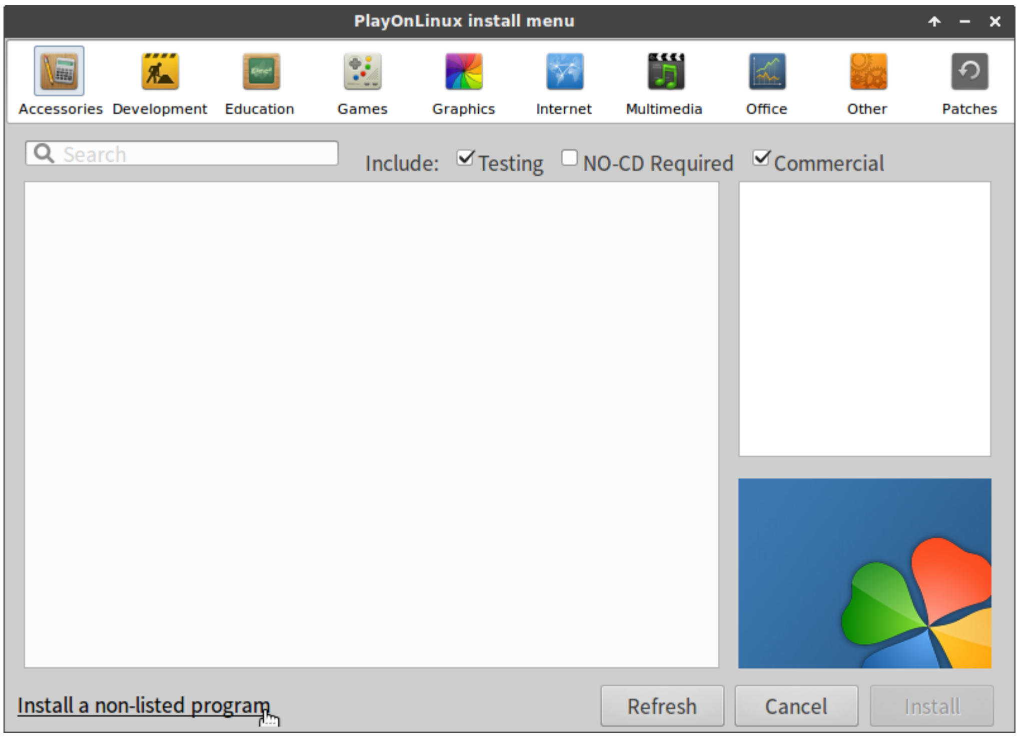


Figure 7: After clicking the + Install button, click on «Install a non-listed program» at the bottom of the window.

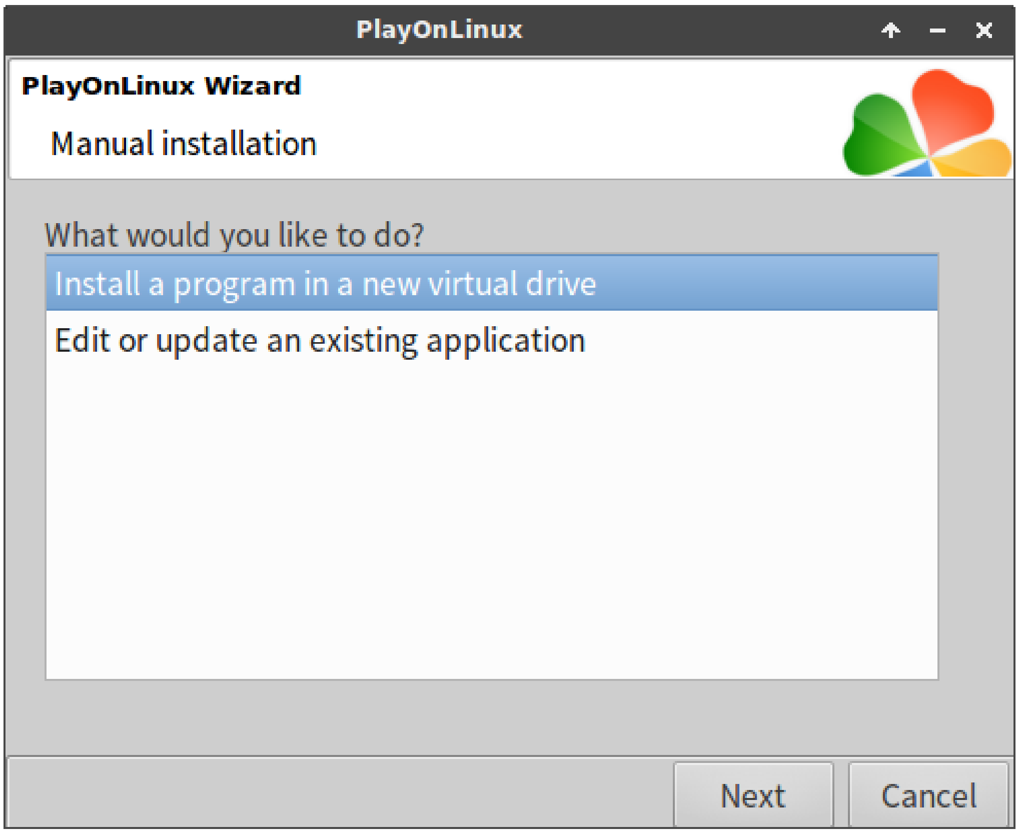
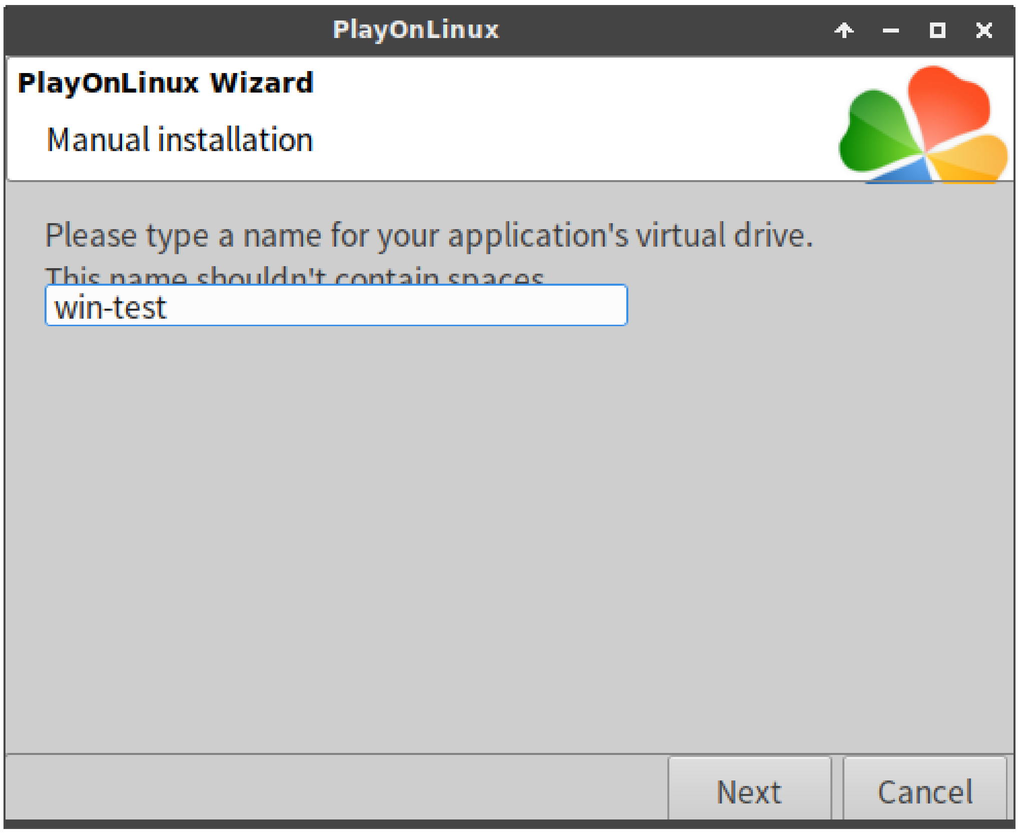

PlayOnLinux
PlayOnLinux Wizard
Manual installation
What would you like to do?
Install a program in a new virtual drive
Edit or update an existing application
Next
Cancel

Figure 8: Choose «Install a program in a new virtual drive» and click on Next.


PlayOnLinux

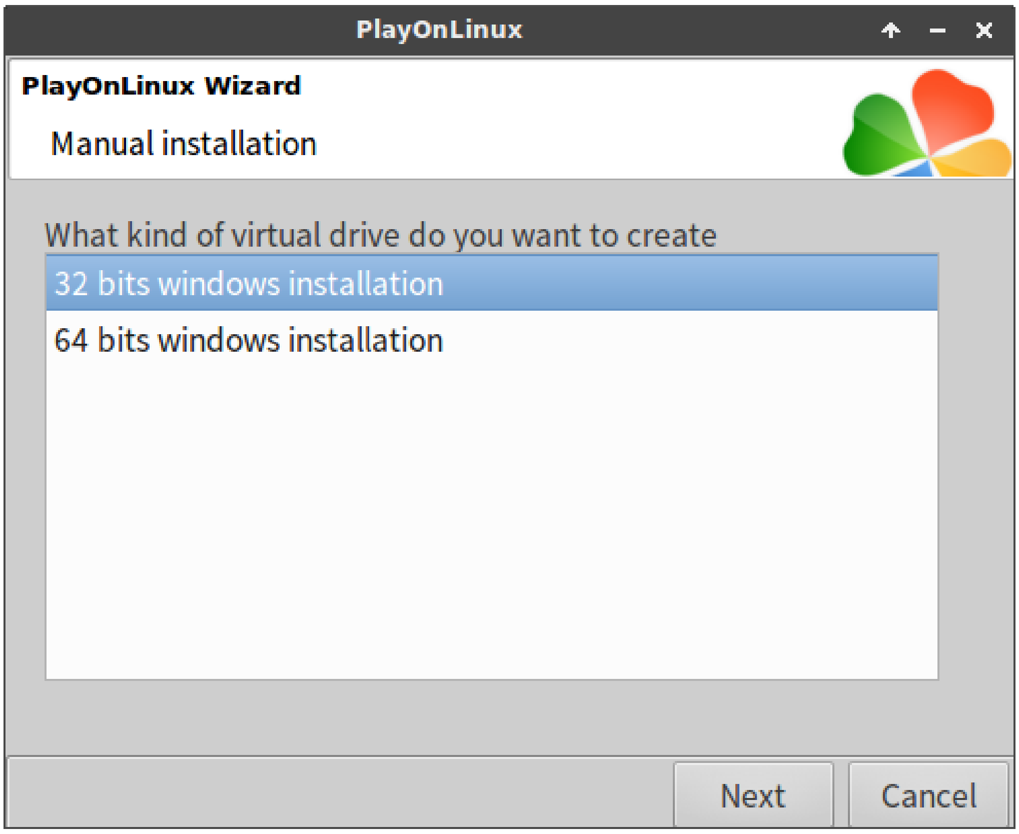
PlayOnLinux Wizard

Manual installation

Please type a name for your application's virtual drive.
This name shouldn't contain spaces

win-test

Figure 9: Type `win-test` and click on Next.


PlayOnLinux

PlayOnLinux Wizard

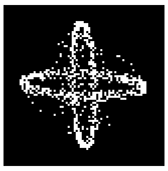
Manual installation

What kind of virtual drive do you want to create

32 bits windows installation
64 bits windows installation

Figure 10: Choose «32 bits windows installation» and click on Next.

MMTTY installation



MMTTY is installed first, on a PlayOnLinux virtual drive called `win-test`. MMTTY and Win-Test will share this virtual drive because both programs are related and need to be aware of oneanother. MMTTY allows for hard FSK keying, which in a contest environment is less prone to errors than modulating audio AFSK.

Caveat: Do not install EXTFSK!

On the web, you may find conflicting advice about installing the EXTFSK MMTTY extension. Here is the definitive answer by Joe Subich, W4TV, of microHAM:

«*NEVER* use EXTFSK with microHAM microKEYER, DigiKeyer, microKEYER II, DigiKeyer II, MK2R+ or micro2R. All of those interfaces provide a UART compatible, virtual port for FSK that handles 45, 50, 75 and 100 baud RTTY (baudot) directly. EXTFSK is not compatible with the FSK output of the microHAM "keyer" devices.»

Wine COM port mapping

Both MMTTY and Win-Test carry along their own drivers to communicate with the CAT interface of most common transceivers. These drivers function perfectly with PlayOnLinux, provided the Linux `mhuxd` device ports appear as `COM` ports to the Windows™ programs on the `win-test` virtual drive.

I reiterate that as a GNU/Linux user, one needs to make oneself member of the `mhuxd` group in order to use these virtual ports.

Old `wine` versions, prior to version 2.8

Up to `wine` version 2.8, symbolic links from the Linux device ports to the typical Windows™ `COM` ports can be created directly in the directory `~/.PlayOnLinux/wineprefix/win_test/dosdevices` on the `win_test` virtual drive:

```
$ cd ~/.PlayOnLinux/wineprefix/win_test/dosdevices
$ ln -s /dev/mhuxd/cat1 com1
$ ln -s /dev/mhuxd/ptt1 com2
$ ln -s /dev/mhuxd/fsk1 com3
```

New `wine` versions, since version 2.8

Since the advent of `wine` version 2.8, above solution no longer works because `wine` overwrites any symbolic link. However, the CAT1, PTT1 and FSK1 ports can be mapped to respectively `COM1`, `COM2` and `COM3` by adding new string entries to the `HKEY_LOCAL_MACHINE\Software\Wine\Ports` key of the `wine` registry on the `win-test` virtual drive.

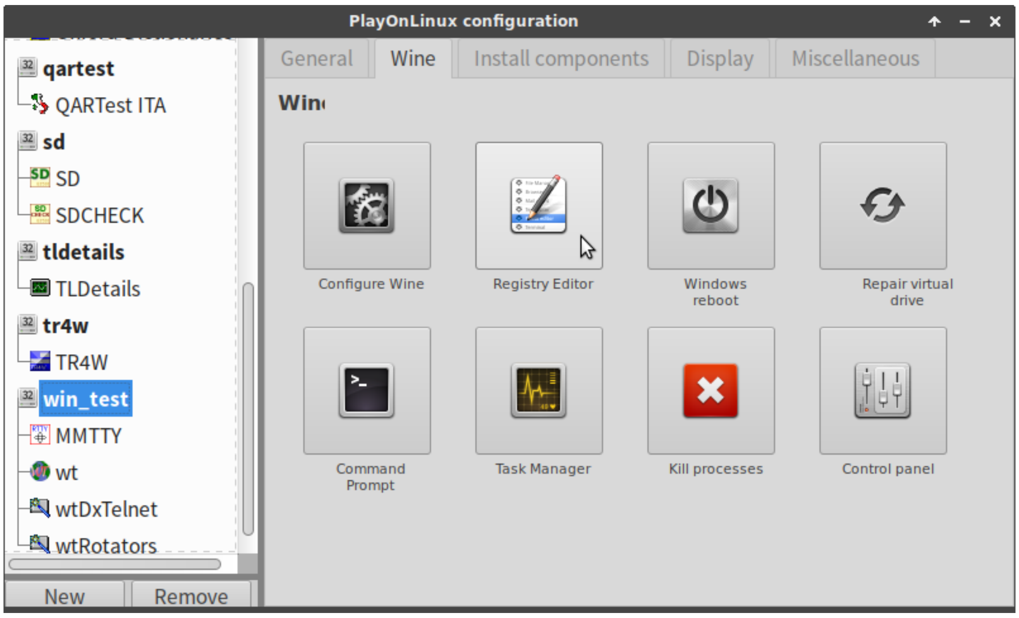
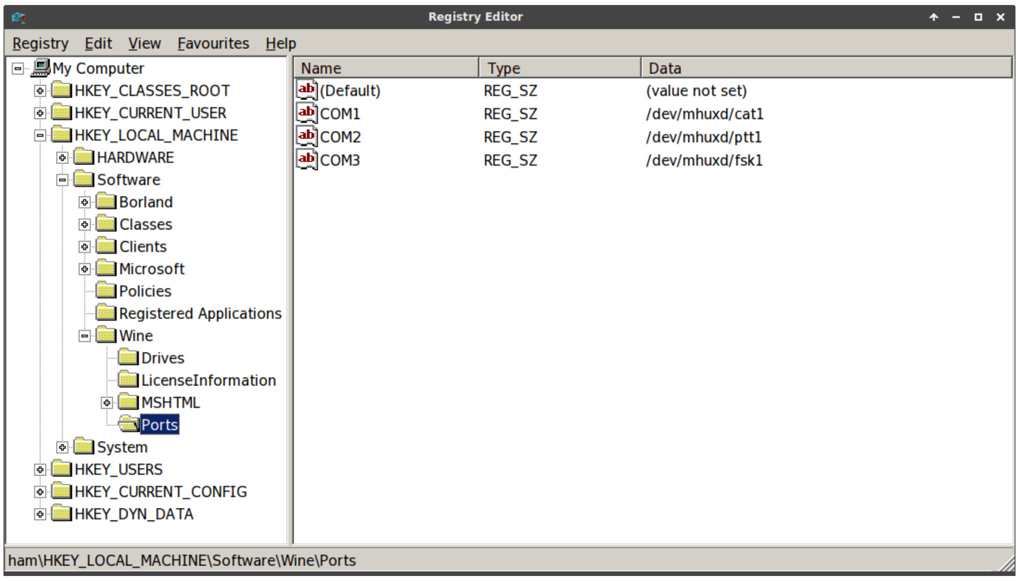


Figure 11: The registry editor can be opened through: `PlayOnLinux` → `Configure` → `win-test` virtual drive → `Wine` tab → `Registry Editor`.

Table 3: String values to be added

name	type	data
`COM1`	`REG_SZ`	`/dev/mhuxd/cat1`
`COM2`	`REG_SZ`	`/dev/mhuxd/ptt1`
`COM3`	`REG_SZ`	`/dev/mhuxd/fsk1`



Name	Type	Data
(Default)	REG_SZ	(value not set)
COM1	REG_SZ	/dev/mhuxd/cat1
COM2	REG_SZ	/dev/mhuxd/ptt1
COM3	REG_SZ	/dev/mhuxd/fsk1

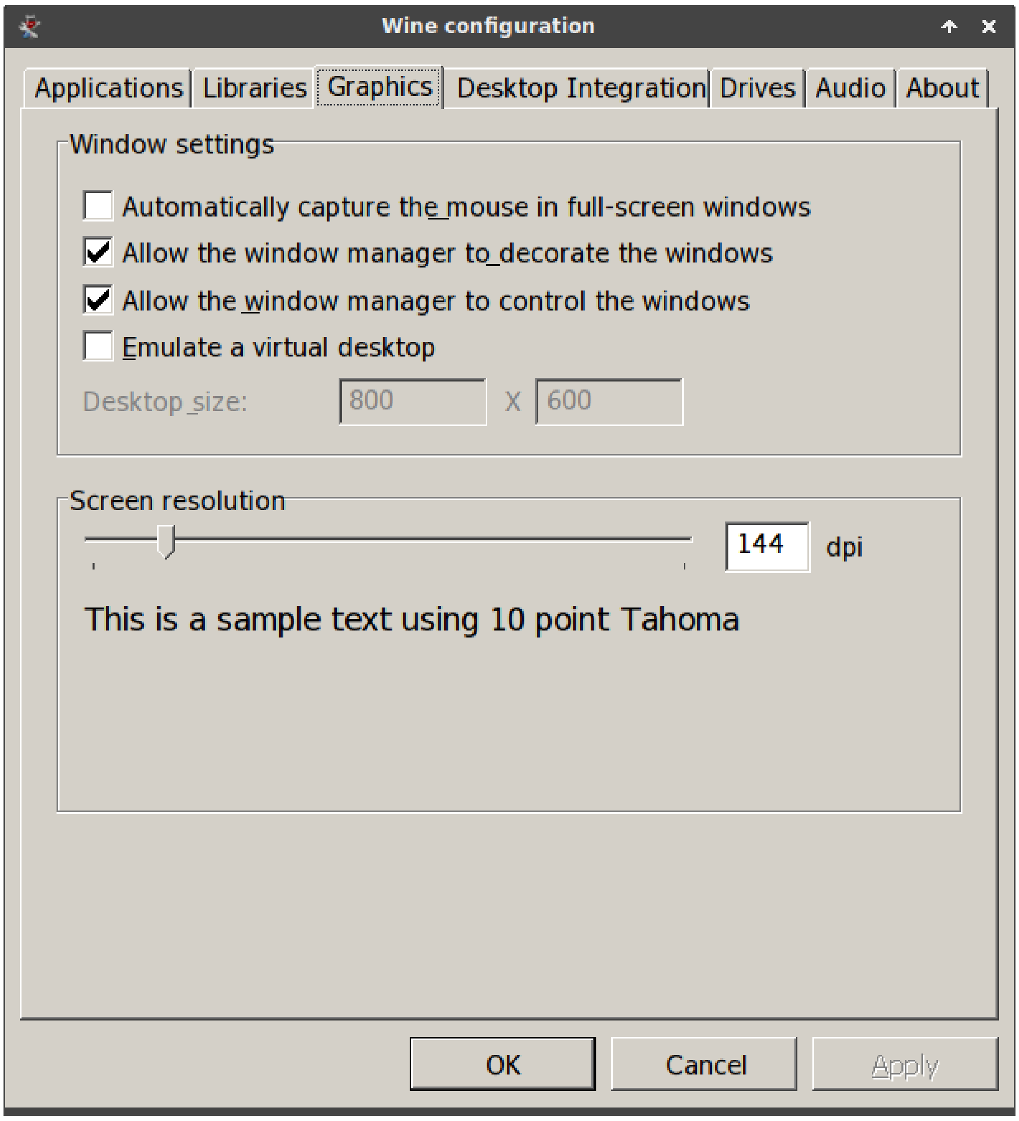
ham\HKEY_LOCAL_MACHINE\Software\Wine\Ports

Figure 12: The `HKEY_LOCAL_MACHINE\Software\Wine\Ports` key with three new string values added to remap `COM1`, `COM2` and `COM3` to the `mhuxd` device ports.

After editing the registry, shut down Wine with `wineserver -k`. Next time Wine runs a program, changes will take effect.

Wine DPI setting

When you are used to GNU/Linux or for people with poor eyesight, the fonts on the Windows™ screens may appear particularly small. Luckily, the dots per inch (DPI) screen resolution can easily be set for each `wine` virtual drive individually.



Window settings

- [ ] Automatically capture the mouse in full-screen windows
- [✓] Allow the window manager to decorate the windows
- [✓] Allow the window manager to control the windows
- [ ] Emulate a virtual desktop

Desktop size: 800 X 600

Screen resolution

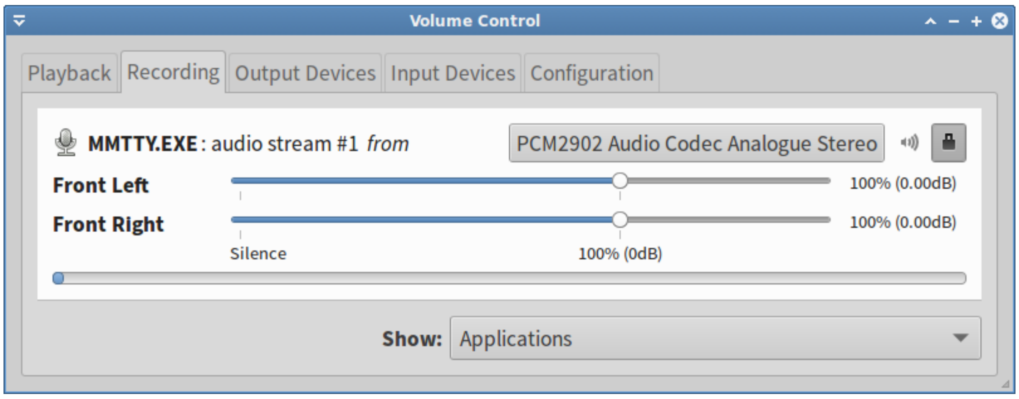
144 dpi

This is a sample text using 10 point Tahoma

Figure 13: The dots per inch (DPI) screen resolution can easily be increased through: `PlayOnLinux` → `Configure` → `win-test` virtual drive → `Wine` tab → `Configure Wine` → `Graphics`.

Pulseaudio settings

The microHAM Digi Keyer contains its very own low-noise PCM2902 stereo USB1.1 codec chip. To use it, open `pulseaudio` volume control while MMTTY is running and select the PCM2902 chip as recording source for MMTTY. Newer microHAM interfaces use a differently named chip, as do modern transceivers with a built-in sound device and USB port.



MMTTY.EXE : audio stream #1 *from* PCM2902 Audio Codec Analogue Stereo

Front Left 100% (0.00dB)

Front Right 100% (0.00dB)

Show: Applications

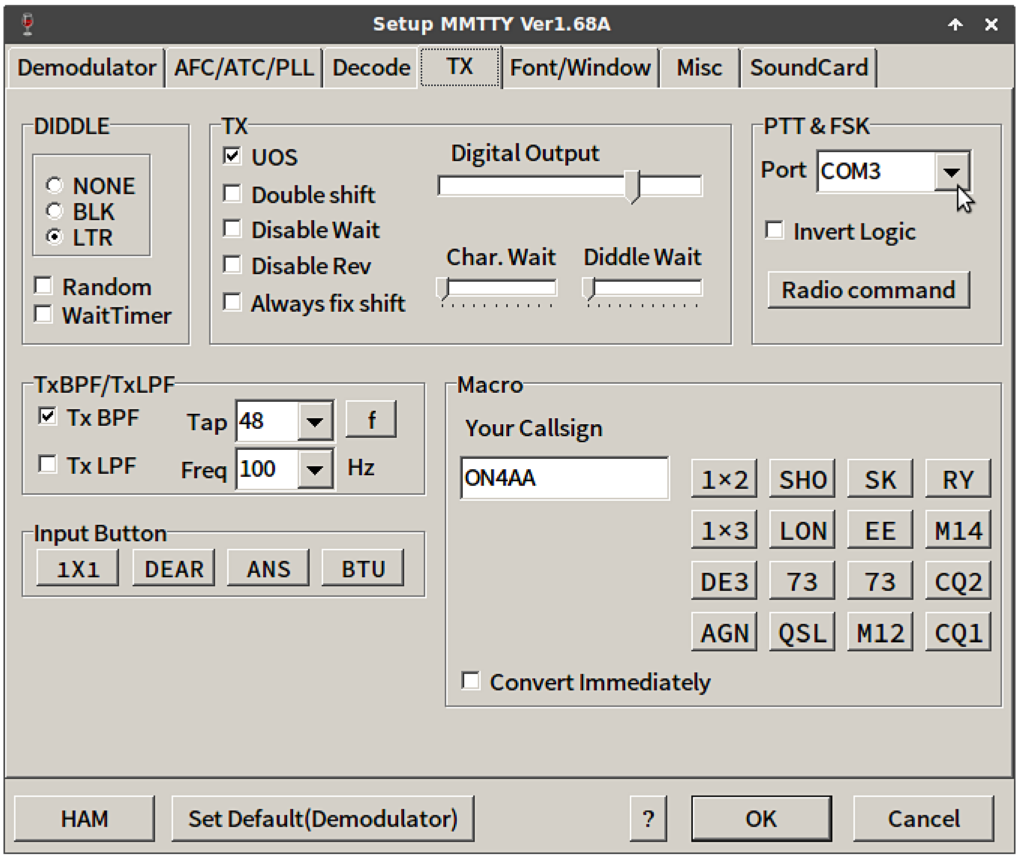
Figure 14: Binding MMTTY recording to the PCM2902 codec using `pulseaudio` volume control in Xubuntu LTS

MMTTY configuration

MMTTY is best configured and tested running standalone, i.e. independently of Win-Test. All MMTTY configuration settings will remain in place under Win-Test, except for the CAT settings and casual QSO message macros which apply only to MMTTY running standalone.

COM port for keying

I am a fervent advocate of hard FSK keying, which in a contest environment is less prone to errors than modulating audio AFSK. To do so, select `COM3` in MMTTY's `TX` menu for FSK (and PTT) keying.



DIDDLE: ○ NONE ○ BLK [✓ LTR]
☐ Random
☐ WaitTimer

TX:
☑ UOS
☐ Double shift
☐ Disable Wait
☐ Disable Rev
☐ Always fix shift

PTT & FSK: Port COM3
☐ Invert Logic

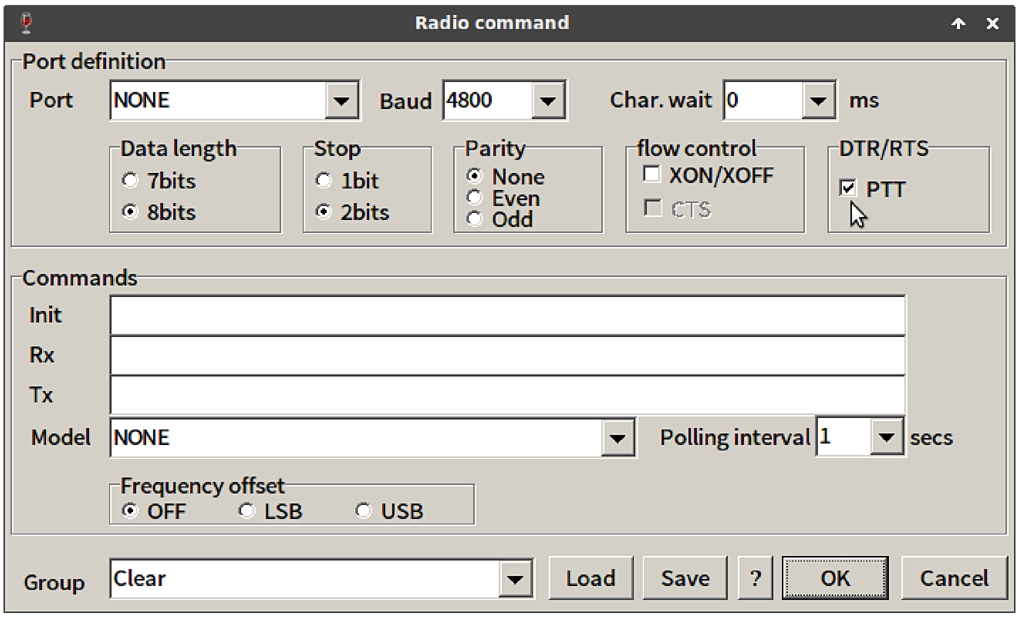
TxBPF/TxLPF:
☑ Tx BPF Tap 48 f
☐ Tx LPF Freq 100 Hz

Input Button: 1X1 DEAR ANS BTU

Macro: Your Callsign ON4AA
☐ Convert Immediately

Figure 15: Select `COM3` for FSK (and PTT) keying in the MMTTY `TX` menu

MMTTY is only able to use a single COM port for both FSK and PTT keying. Hence, PTT keying will be performed with a RTS (ready to send) signal on the very same `COM3` port. For this to happen, place the appropriate check mark in MMTTY's `Radio command` menu, underneath the `TX` menu. This setup will only work when RTS-PTT was previously selected on the microHAM port routing screen.


Radio command

Port definition

Port: NONE Baud: 4800 Char. wait: 0 ms

Data length: 7bits [✓ 8bits]

Stop: 1bit [✓ 2bits]

Parity: [✓ None] Even Odd

flow control: ☐ XON/XOFF ☐ CTS

DTR/RTS: ☑ PTT

Commands

Init:

Rx:

Tx:

Model: NONE Polling interval: 1 secs

Frequency offset: [✓ OFF] LSB USB

Group: Clear

Figure 16: Select `RTS` for PTT keying in the MMTTY `Radio command` menu

MMTTY's all important main settings are next up for discussion.

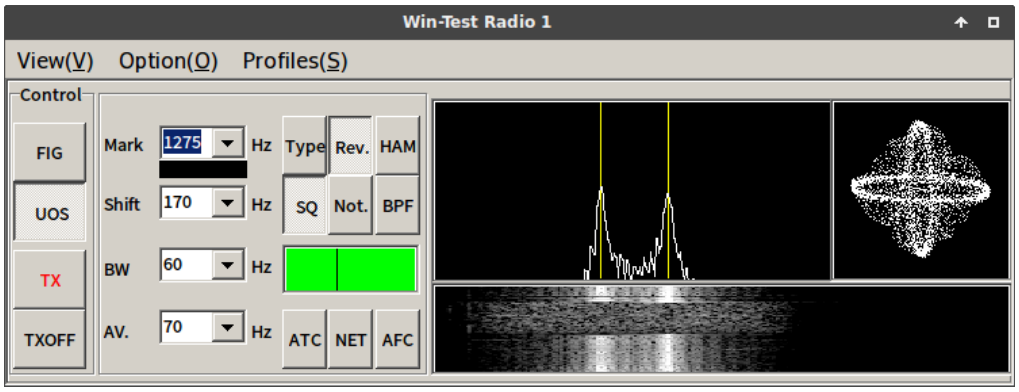


Figure 17: MMTTY on Linux!

Rev.

Because of confusing FT-990 CAT messaging, I need to operate my Yaesu FT-990 in RTTY-USB mode. Consequently, MMTTY's *Reverse* Rev. needs to be always `on` in my case. Test this setting by entering in an RTTY QSO before the contest! Very occasionally, when restarting under Win-Test, MMTTY might be a little confused showing Rev. as `on` whilst not decoding in reverse. Should this happen, simply click Rev. twice, and all should be again in good order.

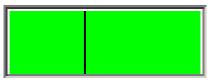
I have noticed that, **upon starting MMTTY V1.68A,** one needs to click the Rev. button **twice** in order to put MMTTY effectively in reverse decoding!

UOS `on`

It is recommended to *Unshift On Space* UOS, on both transmit and receive, whilst employing space delimiters (instead of using `599-001-001` hyphens). Doing so, increases the noise immunity for letters (which are more common), by always forcing the `LTRS` code page (i.e. unshifting) after a space.[1] By using unshifting spaces as delimiters between number sequences, the `FIGS` shift will also be repeated more often.

ATC

Automatic Threshold Control ATC adjusts the input level to the comparator in accordance with the strength of the input signal. If propagation conditions are good, it is better left `off`. It is only useful when there is an echo on the received signal.[2]



MMTTY's decoding threshold can also be set manually by moving the slider along the green bar. Set this according to your local noise level and propagation conditions.

NET

Activating NET makes your transmit frequency follow your receive frequency when using AFSK. This feature should never be used during a contest. It is of no effect when employing truly keyed FSK.

AFC

Automatic Frequency Control AFC lets MMTTY track the frequency of the received signal within the receive passband. Leave AFC `off` when operating in `S&P` mode; In `RUN` mode, having AFC switched `on` may be useful for working «lid operators». However, remember to reset your `MARK` tone using the HAM button, when switching it back `off`! Otherwise, you yourself become the «lid operator»!!!

Table 4: AFC

operating mode	AFC
search & pounce	always `off`
running	allowed `on` reset `MARK` tone using HAM after use!

HAM

HAM resets your `MARK` tone and `SHIFT` to your preset defaults.

There is an excellent MMTTY help which you should have read at least once. Additional MMTTY setup documentation is made available by Donald A. Hill, AA5AU.

Casual QSO messages for MMTTY

Contest messages are always sent by Win-Test; not MMTTY. Independently of these, you can define casual QSO messages for standalone MMTTY use. Here are some examples. Adding line feeds at the beginning and end, makes your messages stand out more at the reception end.

```
—

%c DE %m %m %m K

_\
```

```
—

%f %n
UR %r %r
OP SERGE SERGE
QTH HASSELT HASSELT
%c DE ON4AA K

_\
—

%f %n
UR %r %r
OP SERGE SERGE
QTH HASSELT HASSELT
LOC JO 20 QW  JO 20 QW
P 50 W 50 W
CENTER LOADED OCFD ANT AT 15 M
MMTTY ON XUBUNTU LTS GNU LINUX
WWW.HAMWAVES.COM
%c DE ON4AA K

_\
```

```
—

%c DE %m
TNX %n FOR QSO
LOTW BURO EQSL
73 PEACE AND GD HEALTH
%c DE %m SK

_\
```

```
—

AGN? AGN?

_\
```

```
—

QSL QSL DE %m K

_\
```

Caveat: Avoid frequent shifts to `FIGS`!

When defining RTTY messages, try to avoid frequent changes between letter and figure characters. Missing a `LTRS/FIGS` shift means that all subsequent characters will be printed wrong up until the next `LTRS/FIGS` shift. This situation can be improved upon by **setting *Unshift On Space* UOS, on both transmit and receive, whilst employing space delimiters (instead of using `599-001-001` hyphens).** Doing so, increases the noise immunity for letters (which are more common), by always forcing the `LTRS` code page (i.e. unshifting) after a space.[1] By using unshifting spaces as delimiters between number sequences, the `FIGS` shift will also be repeated more often.

Binary code 00000 is the null character, used for idling.

Table 5: 5-bit ITA2 USTTY code pages

binary	hex	LTRS	FIGS
00011	03	A	-
11001	19	B	?
01110	0E	C	:
01001	09	D	$
00001	01	E	3
01101	0D	F	!
11010	1A	G	&
10100	14	H	#
00110	06	I	8
01011	0B	J	BELL
01111	0F	K	(
10010	12	L	)
11100	1C	M	.
01100	0C	N	,
11000	18	O	9
10110	16	P	0
10111	17	Q	1
01010	0A	R	4
00101	05	S	′
10000	10	T	5
00111	07	U	7
11110	1E	V	;
10011	13	W	2
11101	1D	X	/
10101	15	Y	6
10001	11	Z	"
01000	08	CR	CR
00010	02	LF	LF
00100	04	SP	SP
11111	1F	LTRS	LTRS
11011	1B	FIGS	FIGS

Win-Test installation

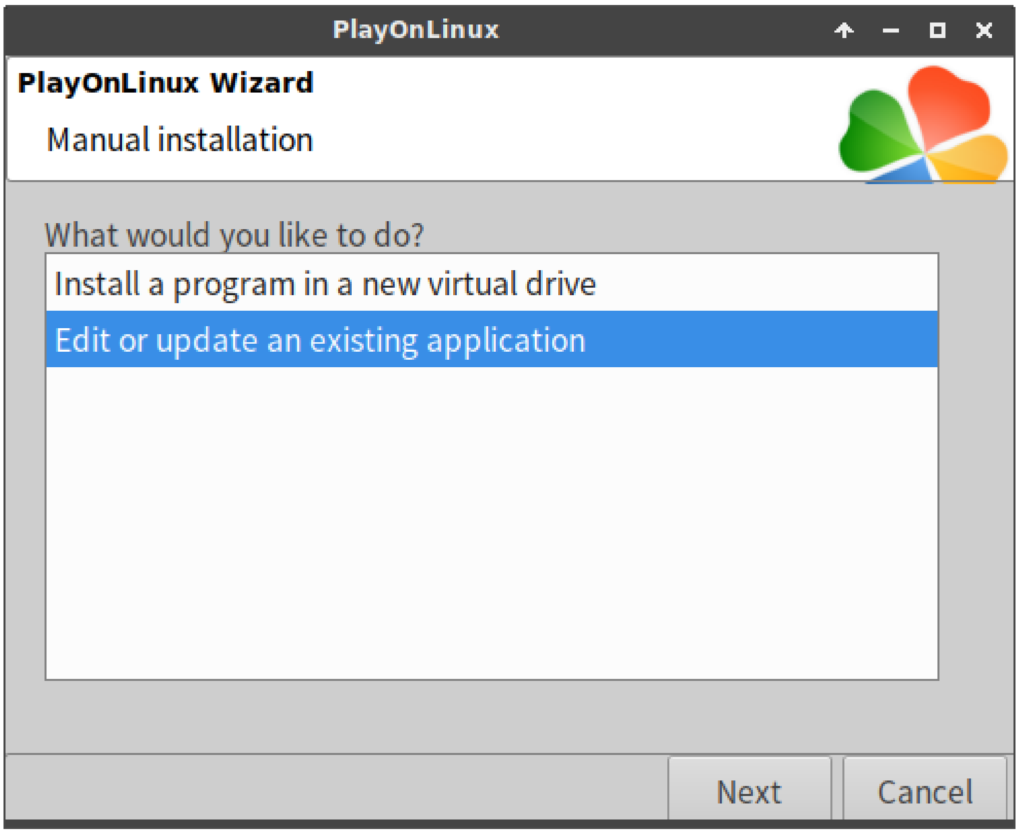
The story behind Win-Test is absolutely fascinating. Win-Test appears to be developed by a handful of French rocket scientists who, in their 35-hour work week, apparently had too much time on their hands waiting for the next rocket launch. Olivier Le Cam, F5MZN, spearheads this tremendous software project. The proceeds of Win-Test go in their integrity to FY5KE, the M/S contest station of the Radio Amateur Club of Kourou, better known as RACK and located at the Centre Spatial Guyanais, the space port of the European Space Agency (ESA) in French Guiana.

Figure 18: Testimony to Win-Test being a serious contest logger: the FY5KE M/S contest team hard at work.

The registration process is quite lengthy. Upon acquiring a license, an e-mail will be sent with credentials to download the registrable version of Win-Test.

Figure 19: FY5KE, the M/S contest station of the Radio Amateur Club of Kourou (RACK) located on the grounds of the European Space Agency (ESA) in French Guiana.

Install Win-Test on the same `win-test` virtual drive where MMTTY was installed previously.


PlayOnLinux

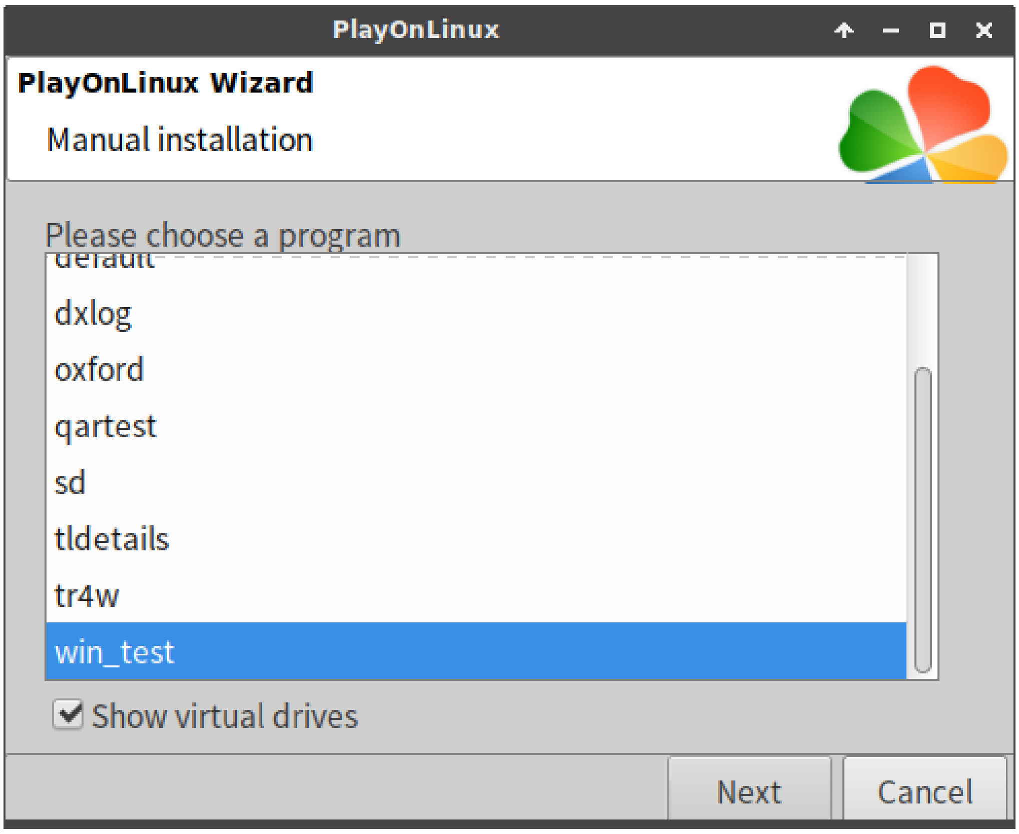
PlayOnLinux Wizard

Manual installation

What would you like to do?

- Install a program in a new virtual drive
- Edit or update an existing application

Figure 20: Choose «Edit or update an existing application» and click on Next.


PlayOnLinux

PlayOnLinux Wizard

Manual installation

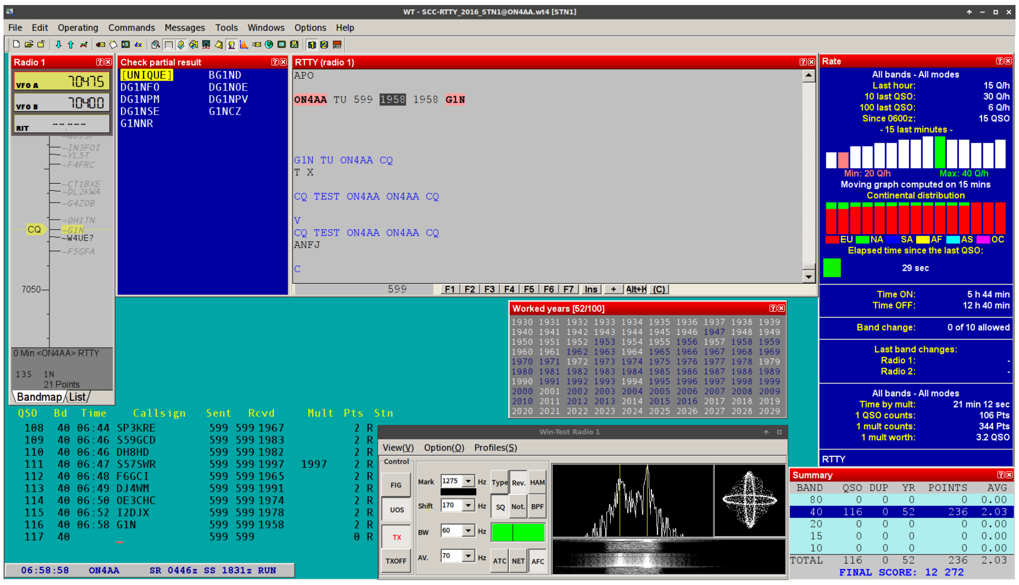
Please choose a program

- default
- dxlog
- oxford
- qartest
- sd
- tldetails
- tr4w
- win_test

[✓] Show virtual drives

Figure 21: Select «Show virtual drives» and choose the `win-test` virtual drive. Continue by clicking on Next.

Without further ado, here is what Win-Test looks like on my Xubuntu LTS GNU/Linux system:


```
WT - SCC-RTTY_2016_STN1@ON4AA.wt4 [STN1]
```

Check partial result

```
UNIQUE    BG1ND
DG1NFO    DG1NOE
DG1NPM    DG1NPV
DG1NSE    G1NCZ
G1NNR
```

RTTY (radio 1)

```
APO

ON4AA TU 599 1958 1958 G1N

G1N TU ON4AA CQ
T X

CQ TEST ON4AA ON4AA CQ

V
CQ TEST ON4AA ON4AA CQ
ANFJ

C
```

Worked years [52/100]

```
1930 1931 1932 1933 1934 1935 1936 1937 1938 1939
1940 1941 1942 1943 1944 1945 1946 1947 1948 1949
1950 1951 1952 1953 1954 1955 1956 1957 1958 1959
1960 1961 1962 1963 1964 1965 1966 1967 1968 1969
1970 1971 1972 1973 1974 1975 1976 1977 1978 1979
1980 1981 1982 1983 1984 1985 1986 1987 1988 1989
1990 1991 1992 1993 1994 1995 1996 1997 1998 1999
2000 2001 2002 2003 2004 2005 2006 2007 2008 2009
2010 2011 2012 2013 2014 2015 2016 2017 2018 2019
2020 2021 2022 2023 2024 2025 2026 2027 2028 2029
```

```
0 Min <ON4AA> RTTY
13S  1N
     21 Points

QSO   Bd  Time    Callsign     Sent  Rcvd        Mult  Pts  Stn
 108  40  06:44  SP3KRE        599  599 1967            2  R
 109  40  06:46  S59GCD        599  599 1983            2  R
 110  40  06:46  DH8HD         599  599 1982            2  R
 111  40  06:47  S57SWR        599  599 1997   1997     2  R
 112  40  06:48  F6GCI         599  599 1965            2  R
 113  40  06:49  DJ4WM         599  599 1991            2  R
 114  40  06:50  OE3CHC        599  599 1974            2  R
 115  40  06:52  I2DJX         599  599 1978            2  R
 116  40  06:58  G1N           599  599 1958            2  R
 117  40                       599  599                 0  R

06:58:58   ON4AA    SR 0446z SS 1831z RUN
```

Rate

```
All bands - All modes
Last hour:              15 Q/h
10 last QSO:            30 Q/h
100 last QSO:            6 Q/h
Since 0600z:            15 QSO
- 15 last minutes -
Min: 20 Q/h         Max: 40 Q/h
Moving graph computed on 15 mins
Continental distribution
EU NA SA AF AS OC
Elapsed time since the last QSO:
29 sec

Time ON:              5 h 44 min
Time OFF:            12 h 40 min

Band change:      0 of 10 allowed

Last band changes:
Radio 1:                       -
Radio 2:                       -

All bands - All modes
Time by mult:      21 min 12 sec
1 QSO counts:            106 Pts
1 mult counts:           344 Pts
1 mult worth:            3.2 QSO

RTTY
```

Summary

```
BAND    QSO  DUP   YR   POINTS    AVG
  80      0    0    0        0   0.00
  40    116    0   52      236   2.03
  20      0    0    0        0   0.00
  15      0    0    0        0   0.00
  10      0    0    0        0   0.00
TOTAL   116    0   52      236   2.03
      FINAL SCORE: 12 272
```

Figure 22: Win-Test with MMTTY on my Xubuntu LTS GNU/Linux system, both running under PlayOnLinux. While writing this article, I made a somewhat casual effort for the 2016 SCC RTTY contest.

Well in advance of any contest…

Check whether the contest is supported by Win-Test and if any Win-Test updates are available.

RTTY contest messages

Win-Test's default RTTY contest messages can be improved upon. Donald A. Hill, AA5AU, of RTTY contesting fame offers a great breakdown of optimal RTTY contest messages. Some of this great information has been condensed in the following set of rules[3] and the tables below. On his fabulous website, Don also offers a number of contest-specific examples.

1. Start all primary messages with a carriage return (CR/LF).
2. End *all* messages with a single space character.
3. Do not use outdated characters or character strings such as ~~DE, K, BK, SK or UR~~ in any of your messages.
4. Send RST always as `599` and send it only once if required. If not required, do not send RST at all.
5. Always employ space delimiters (instead of `599-001-001` hyphens).[1]
6. Always end your CQ message with `CQ`.
7. Always end your run confirmation message with either `CQ` or `QRZ`.

If you are completely new to RTTY, you will certainly find the RTTY DXing presentation[1] of Ed Muns, WØYK/P49X, very helpful in getting you started.

As indicated by the RTTY messaging documentation, RTTY messages are defined under the `Options → RTTY → Modify standard messages…` menu.

During a contest, you can work stations in two ways:

- either *running*: i.e. calling CQ TEST,
- or *search & pounce*: i.e. working stations that are calling CQ.

Contest mode switching is done by means of the Ctrl+TAB key combination. The currently selected mode is shown in the clock window.

Table 6: Switching between running and search & pounce

key combination	action
Ctrl+TAB	Switch the primary radio between RUN and S&P mode.
SHIFT+Ctrl+TAB	Switch the secondary radio between RUN and S&P mode.

Table 7: Win-Test run macros for a contest with serial exchange

run messages	key
F1	`$13 $RSTEXCHSENT CQ TEST $MYCALL $MYCALL CQ\`
F2	`$13 $SETEXCHSENT $LOGGEDCALL $RST $SERIAL $SERIAL $LOGGEDCALL\`
F3	`$CR $13 $LOGGEDCALL TU $MYCALL CQ\`
F4	`$13 AGN? AGN?\`
F5	`$13 $SERIAL $SERIAL $SERIAL $SERIAL\`

Table 8: Win-Test S&P macros for a contest with serial exchange

key	search & pounce messages
F1	`$13 $MYCALL $MYCALL $MYCALL\`
F2	`$13 $SETEXCHSENT $RST $SERIAL $SERIAL $MYCALL\`
F3	`$13 QSL $MYCALL\`
F4	`$13 AGN? AGN?\`
F5	`$13 $SERIAL $SERIAL $SERIAL $SERIAL\`

Table 9: Win-Test run macros for a contest with zone exchange

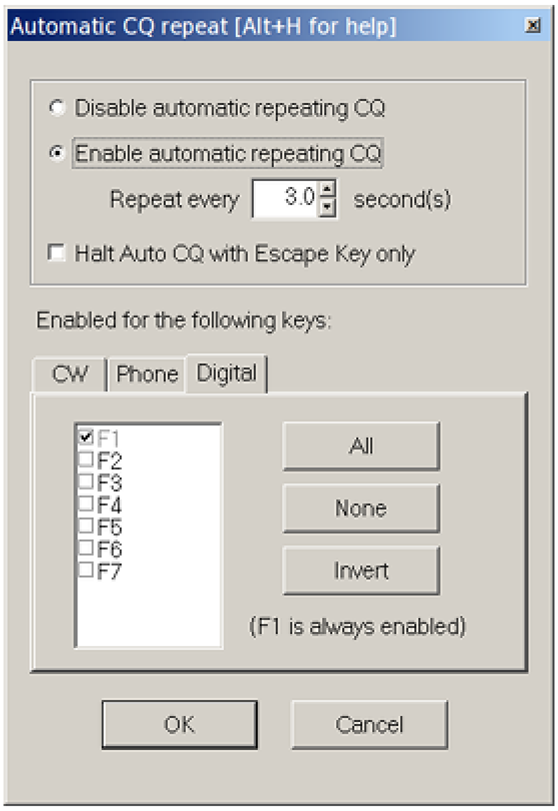
key	run messages
F1	`$13 $RSTEXCHSENT CQ TEST $MYCALL $MYCALL CQ\`
F2	`$13 $SETEXCHSENT $LOGGEDCALL $RST $ZONE $ZONE2 $LOGGEDCALL\`
F3	`$CR $13 $LOGGEDCALL TU $MYCALL CQ\`
F4	`$13 AGN? AGN?\`
F5	`$13 $ZONE $ZONE2 $ZONE $ZONE2\`

Table 10: Win-Test S&P macros for a contest with zone exchange

key	search & pounce messages
F1	`$13 $MYCALL $MYCALL $MYCALL\`
F2	`$13 $SETEXCHSENT $RST $ZONE $ZONE2 $MYCALL\`
F3	`$13 QSL $MYCALL\`
F4	`$13 AGN? AGN?\`
F5	`$13 $ZONE $ZONE2 $ZONE $ZONE2\`

The `$13` macro code sends a carriage return character to the other station. Are you missing some exchange variable? Here is a list of all available Win-Test message variables

In the wee hours of the contest, even hitting the F1 key will get draining. This is where `Tools → Automatic CQ repeat…` comes handy. Hit ESC to interrupt it at any time.


Automatic CQ repeat [Alt+H for help]

( ) Disable automatic repeating CQ
(●) Enable automatic repeating CQ
Repeat every 3.0 second(s)
☐ Halt Auto CQ with Escape Key only

Enabled for the following keys:

CW | Phone | Digital

☑ F1
☐ F2
☐ F3
☐ F4
☐ F5
☐ F6
☐ F7

(F1 is always enabled)

Figure 23: Automatic CQ repeat settings

QSO entry

Caveat: Select the Win-Test window to communicate!

MMTTY lives in a window which is actually separate from Win-Test's window. By consequence, **after tinkering with MMTTY, be sure to reselect Win-Test's window!** Failure to do so, will result in you hitting buttons without anything happening. You are warned.

Table 11: Win-Test callsign colours

callsign colours
`NEW CALLSIGNS`
`DUPES`
`NEW MULTIPLIERS`
`NEW DOUBLE MULTIPLIERS`

Table 12: Win-Test QSO entry keys

key combination	action
ESC	Abort the current transmission.
CLICK on call	Enter the RTTY call.
CLICK on exchange	Enter the RTTY exchange.
RETURN	**Log** the QSO with bandmap entry.
numeric ENTER	Log the QSO **without bandmap entry.**
SPACE	Move to **next** essential QSO field.
TAB	Move to the **adjacent** QSO field.
Alt + F	Edit date, time, frequency or operator of QSO.
Alt + W	**Wipe** the current QSO record completely.
Ctrl + W	Wipe the current call, **keeping the exchange.**
↑ / ↓	**Edit** a QSO or reuse a QSO record for messaging.
Ctrl + TAB	Switch the primary radio between `RUN` and `S&P` **mode.**

- RTTY window documentation
- QSO entry documentation

Band maps

Right click `Properties…` on the band map to configure the band map spot lifetime. The maximum value is 120 minutes. In this very same configuration window, also change the spot bandwidth to 400 Hz for RTTY. For more information, refer to the band map documentation.

Table 13: Band map keys

key combination	action
RETURN	**Log** the QSO with bandmap entry.
numeric ENTER	Log the QSO **without bandmap entry.**
Ctrl + RETURN	Place the current call **on the band map without logging.**
DOUBLE CLICK	**Use** the spot.
Ctrl + DOUBLE CLICK	**Delete** the spot.
MOUSE WHEEL	Zoom the band map.

More Win-Test information

- At any time, one can modify the contest configuration by reopening the log file.
- Log files are available under `/home/user/win-test`
- http://docs.win-test.com/wiki/Menu:Tools
- http://docs.win-test.com/wiki/Interfaces
- http://docs.win-test.com/wiki/RTTY
- http://www.country-files.com/
- http://www.supercheckpartial.com/

2Tone

According to Marc, DO4DXA, the RTTY-only decoder 2Tone can be used with Win-Test. This new decoder by G3YYD may offer improved decoding. However, MMTTY is still required for transmitting RTTY as 2Tone does not transmit. Marc's trick involves renaming `2tone.exe` to `mmtty.exe`. I have not tried this yet, but when I do, I will report here.

Cabrillo & ADIF

At the end of the contest, create `Cabrillo` and `ADIF` log files for log submission, respectively importing to the CQRLOG DX logging program . These files will be available in the `~/win-test/` directory, after going through `File → Create log file…`.

Conclusion

There you have it. Provided you take care of using the right software, RTTY contesting at a competitive level sure is possible with GNU/Linux! Once you have set this up in, for example, Xubuntu LTS, you will experience surprisingly less maintenance issues than with whatever Windows™ version.

TODO

- Installing a Windows™ program under PlayOnLinux
- band maps: segments
- country files
- super check partial

- Reference for "Truly keyed FSK on transmit is preferred over transmitting audio AFSK."
- spot click mode

References

1. Ed Muns, WØYK/P49X. RTTY DXing. Published online 2012. https://www.ncdxc.org/presentations/2012/W0YK-RTTY-DXing-06212012.pdf

2. Jan Ditzian, KX2A. MMTTY help. Published 2003. https://hamwaves.com/linux.rtty/doc/MMTTY_Help.v166G.pdf

3. Donald A. Hill, AA5AU. RTTY contest messages. https://www.rttycontesting.com/lagniappe/rtty-messages/