Setup for running VARA and RMS Express on Linux

Tested for Ubuntu 22.04 and Wine version 9.0-RC2 (Develop)

LA9RT - Bent

19. December 2023

Introduction

It seems that the development of Wine today has matured so far that it is possible without any big quirks to run both RMS Express and VARA in combination. Even packet modem TNC will work. However, ARDOP seems to be more difficult still. The following is my steps how I got this to work.

The computer I have used is rather standard, however self build. Main characteristics are:

- AMD Athlon(tm) 5350 APU with Radeon(tm) R3
- 16 Gbyte RAM
- Ubuntu 22.04 – 64 bit
- GeForce GT 1030 with passive cooling (no fan)

The CPU is rather old (2014), has four 64-bit cores and runs at 2 GHz with 25W TDP. You may call it a low end CPU.

Prerequisite

You will need an installation of Wine, either 32-bit or 64-bit. I suggest a 64-bit installation. However, you must also be certain that 32-bit libraries are included. Wine 64 bit seems to be able to run 32-bit software, at least for VARA and RMS Express, which are 32-bits as far as I know. I am running Ubuntu 22.04 64-bit, and this is used in my examples. You may download the Debian files, but it is probably more convenient to use PPA. Much easier to update.

Wine installation

This I very well described at WineHQ, https://www.winehq.org/ . I have used the Develop version 9.0-rc2, but have runned it stable from 8.20. I have not tested the stable version 8.0.2. You find packages at:

https://wiki.winehq.org/Download

For Ubuntu it is described how you use PPA, see https://wiki.winehq.org/Ubuntu. I recommend using this method. It makes updates very easy.

I think it also will be wise and necessary to enable 32 bit architecture. It will let 32 bits program run in a 64 bit Wine installation, and do not harm the 32 bit Wine version if you choose to also install that. Thus, do before installing Wine (on Ubuntu).

```
username@machinename:~$ sudo dpkg --add-architecture i386
```

Then just install the Wine version you like according to WineHQ advises. If you choose the stable one, and it doesn't work, try develop version which is tested. Don't forget to install the keys as described on WineHQ page. Release candidate 2 of the new version 9 is published, so soon a new stable version should be published.

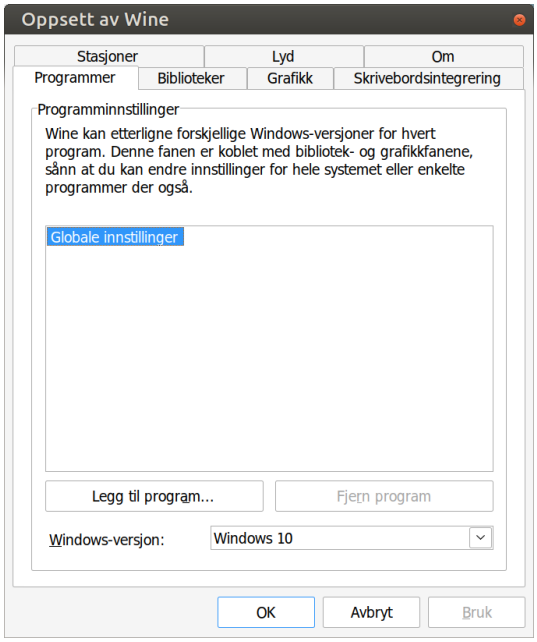
Do a standard apt update and install Wine with your preferred package manager. I suggest to do it without any special settings. It will install Wine 64 bit (On Linux 64 bit) and use the directory ~/.wine. This will be your default Wine version from which you very easy may create other installations. Update of this version will float to other installations automatically when used. We will follow this route. If you already has an installation in ~/.wine, rename it or choose another installation directory for this tutorial.

If the directory start with a . (dot, hidden directory), you may get problems with some pre-formatted messages using html forms. This is not an error in Wine or the setup. But it turns out that e.g. a snap installation of Linux programs will not be able to read files in a hidden directories due to security settings in snap. It is possible to override this, but that may very well give you unknown security risks. I strongly recommend not to do that.

After installing Wine, you run the configuration program from Linux command line:

```
username@machinename:~$ env WINEPREFIX=~/.wine winecfg
```

After an update sequence a window like this (image in Norwegian) is shown:



Select Windows 10 as your Windows version. You may later on play with this for different windows programs, if you like. If you have a UHD display, you may need to change the default screen setup from standard 96 DPI to 132 DPI or according to your preference. You do that on the Graphic tab. Press OK and you are finished with your base installation.

Now it is time to create an installation adopted for Winlink Express with accessories. Create the empty directory ~/wine64. Note the missing dot in front of wine64. Then:

```
username@machinename:~$ mkdir wine64
username@machinename:~$ WINEARCH=win64 WINEPREFIX=~/wine64 winecfg
```

The first line creates the directory, assuming it doesn't already exist. If so, be sure it is absolutely empty. The next line will create a 64-bit Wine installation in ~/wine64. It may run for a while. Just select Windows 10 when the window appears and if necessary change the DPI, just as you did for the primary install previously.

If you really want a 32 bit version, just do :

```
username@machinename:~$ mkdir wine32
username@machinename:~$ WINEARCH=win32 WINEPREFIX=~/wine32 winecfg
```

It is important that your win32 directory is empty when you start.

You should now have a Wine installation in "~/wine64". Wine operates with an environment variable WINEPREFIX. This is just an absolute path to that directory.

If you wanna run an application in 32 bit Wine, you gives the following command:

```
username@machinename:~$ env WINEPREFIX=~/wine32 wine \\PATH\\program.exe
```

and guess what. For 64 bit Wine it is:

```
username@machinename:~$ env WINEPREFIX=~/wine64 wine \\PATH\\program.exe
```

Note the that you use "\\" between directories and I use "env" in the beginning. It may work without this "env", but it is created automatically when you let installation program making shortcuts.

As an example, my shortcut for RMS is like:

```
 username@machinename:~$ env WINEPREFIX="/home/username/wine64" wine C:\\
users\\Public\\Desktop\\Winlink\ Express.lnk
```

All in one line. Note that a space is prefixed with «\».

Winetricks installation

Wintetricks is a shell script. So you just need to download and make it runnable. You will find winetricks at https://wiki.winehq.org/Winetricks . You will use this script later on.

Access to ports

Modern transceivers of today use may use USB ports to control operations of the rig and often has a build in sound card also using USB. To have access to those ports on your Linux computer, you must append the group those ports are using, to your username. On Ubuntu this group is «dialout». You may see the group by list the devices doing a long list «ls -l /dev» of the device directory. You add the «dialout» group doing this command:

```
username@machinename:~$ sudo usermod –a –G dialout username
```

Please remember the «-a» option. Otherwise your group membership will be replaced, not just appended «dialout».

Wine will create a mappings to ttyS and ttyUSB in the installation directory “~/wine64/dosdevices” or “~/wine32/dosdevices” as COM ports. Those COM ports are visible for Windows programs running under Wine.

Updated dvoa.dll

Download the updated dvoa.dll from https://groups.io/g/VARA-MODEM/files/Wine/dvoa.dll

MD5SUM: f4ba2760d0858ad10429d5a64dc383a6 dvoa.dll
SHA256: 1026c978b5801b865f7ee6fea4f6e691d3046596898190f9352177cf3048b4e3 dvoa.dll

You will need this file to have VARA and RMS Express to cooperate for Winlink Express 1.7.11.0 and earlier. This DLL is under testing on Windows platform by the winlink team. If it turns out ok, it is a good chance it will be included in newer Winlink Express versions.

In Linux, rename the dvoa.dll file that RMS Express installs to e.g. xdvoa.dll. Copy the downloaded dvoa.dll to the directory, effectively replacing the original dvoa.dll.

For this DLL the hole credit has to be given to VE3NEA, Alex.

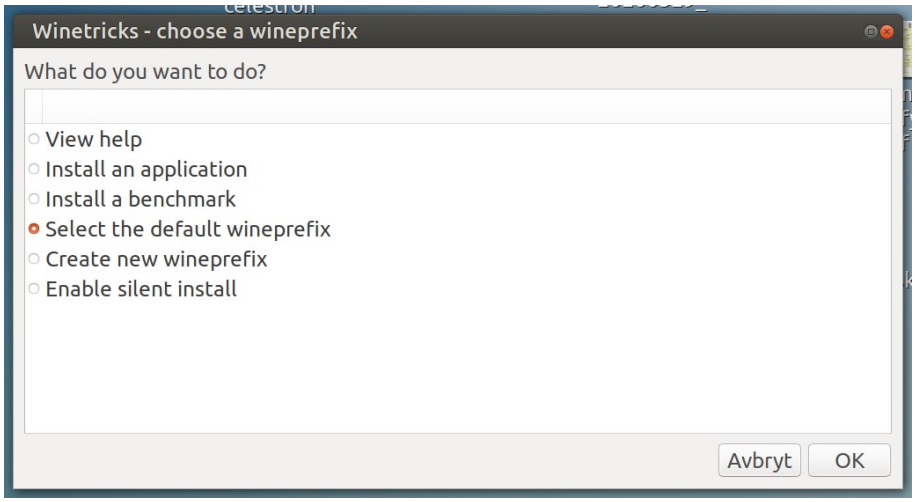
Installing Visual Basic 6 runtime

VARA do need VB6 runtime installed. The simplest way is to use winetricks for this as shown below. I have my Wine 64-bit installed in ~/wine64. I here do the install for the 64 bit version of Wine. I do all the work as a user. You start winetricks as showed:

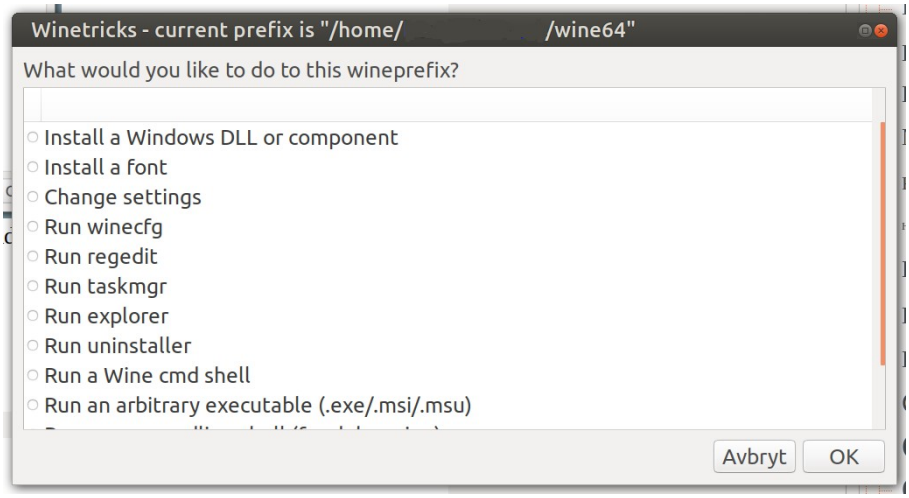
```
username@machinename:~$ env WINEPREFIX=~/wine64 winetricks

Using winetricks 20230212-next - sha256sum:
fe0550e0d843214f87dcb0f4aa591be0046fa93db7b8330217dd926258e628fc with wine-
9.0-rc2 and WINEARCH=win64

winetricks GUI enabled, using zenity 3.42.1
```

Prefixing the command with “env WINEPREFIX”=~/wine64” enables winetricks for that occurrence. If I have wanted to do this for my 32 bit Wine, I would have used “env WINEPREFIX”=~/wine32”. The GUI window should now be operative:

On the images “Avbryt” is the Norwegian word for “Cancel”.

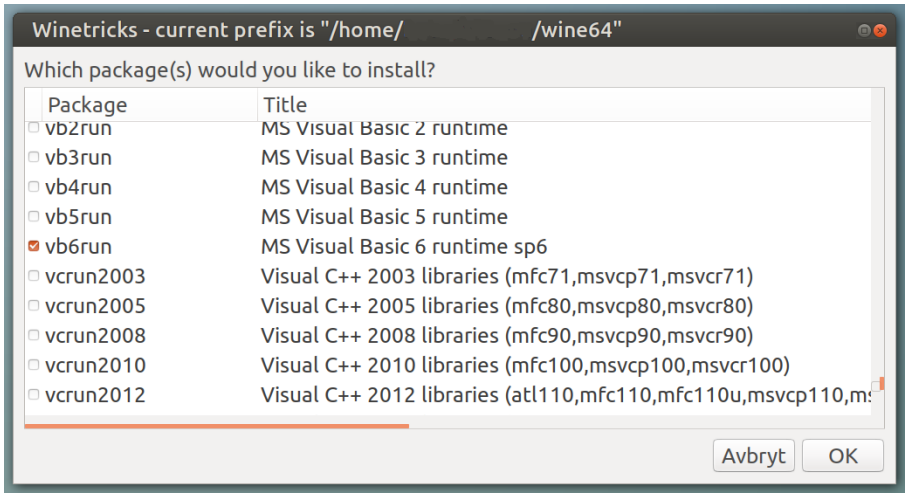


If not already selected “Select the default wineprefix, select it and press “OK”. Then you get this window:



Select “Install a Windows DLL or component” and press “OK”.

Now we almost are there :-) Select “Install a Windows DLL or component”. On the window that pop up, go down to you find vb6run and select it.



This you install by pressing “OK”. On 64 bits wine you may get a set of warnings. That is normal because this component is a 32 bit component.

Installing VARA

When the above is done, the rest is quiet easy. Download the VARA software from https://rosmodem.wordpress.com/. Then you install it with:

```
 username@machinename:~$ env WINEPREFIX=~/wine64 wine PATH/VARA\ setup\ \
(Run\ as\ Administrator\).exe
```

If you has only one version of Wine, you may drop WINEPREFIX part. You have to use «wine start» when you put in a path to the exe file. This set the working directory accordingly. Also note that you have to escape spaces and parentheses in the filename which is normal in a Linux terminal.

During installation, accepts all defaults and be sure you say Yes to create a shortcut on your desktop. Nice to have it defined by the installation.

Installing RMS Express

Then you install it with:

```
 username@machinename:~$ env WINEPREFIX=~/wine64 wine
PATH/Winlink_Express_install.exe
```

If you has only one version of Wine, you may try dropping drop the WINEPREFIX part..

During installation, accepts all defaults and be sure you say Yes to create a shortcut on your desktop. Nice to have it defined by the installation.

After installation you replace the dvoa.dll installed by the installation routine with the previous downloaded recompiled dvoa.dll. You find it in the directory “~/wine64/drive_c/RMS Express”.

Compiling the DLL

This part is more for reference and if someone else also want to do it for themselves. You may download the source code from https://github.com/VE3NEA/DVOACAP. It is written in Pascal by VE3NEA – Alex. I use Lazarus and FreePascal a lot (https://www.lazarus-ide.org/ and https://www.freepascal.org/), so this was a positive discovery.

Since I am running Linux, the first thing is to build a cross-compiler for FreePascal targeting 32-bit Windows. I have FreePascal version 3.2.2 installed, and this I used as given in the following description.

```
# Navigate to the fpc source folder.

cd /usr/share/fpcsrc/3.2.2

# Compile the cross-compiler.

sudo make clean all OS_TARGET=win32 CPU_TARGET=i386


# Install the cross-compiler.

sudo make crossinstall OS_TARGET=win32 CPU_TARGET=i386 INSTALL_PREFIX=/usr


# Using Lazarus, link the cross-compiler and place the link where Lazarus
# can see it.

sudo ln -sf /usr/lib/fpc/3.2.2/ppcross386 /usr/bin/ppcross386
```

As you can see, the cross-compiler is named “ppcross386”.

Presuming you have downloaded the source code, you move to that directory. In my installation that is:

```
 username@machinename:~$ cd ~/Pascal/Dvoa/DvoaDll

 # Cross-compile the DLL

 username@machinename:~/Pascal/Dvoa/DVoaDll$ ppcross386 -MDELPHI -B -Twin32 -
fPIC dvoa.dpr
```

These options tells FreePascal to use Delphi mode, Windows 32 bit and position independent code.