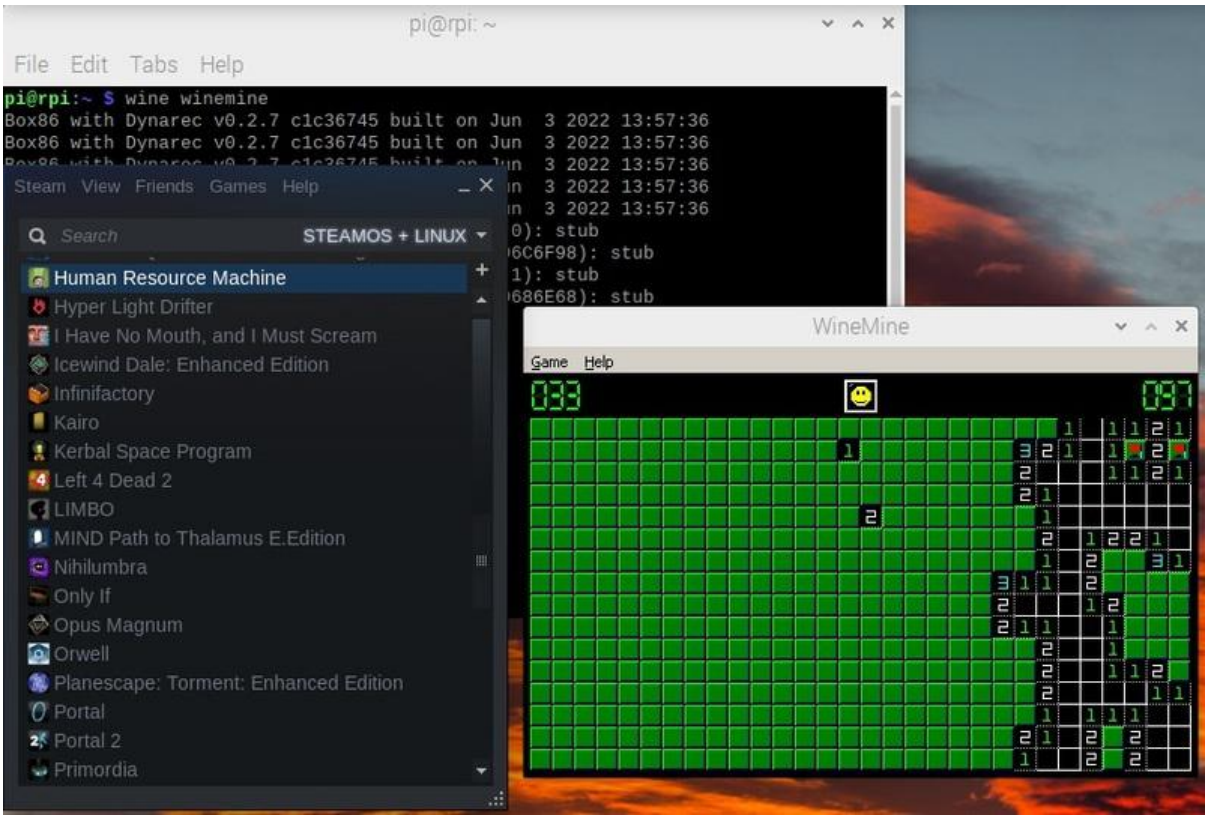
What's in the Box86? More gaming possibilities on Pi!

Created by Matt G

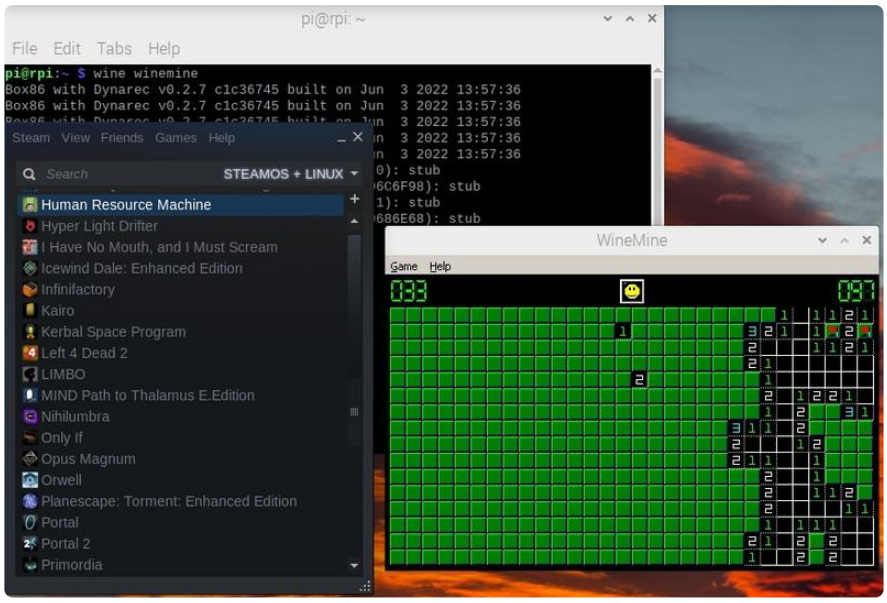


https://learn.adafruit.com/box86-gaming-on-arm-raspberry-pi

Last updated on 2022-12-01 04:12:10 PM EST

Table of Contents

Overview



Have you ever wanted to run something on your Raspberry Pi but it was only made for x86 processors? Box86 () solves that and lets you run x86 Linux binaries on your ARM system - Pi, ODroid, OpenPandora, etc. It can even do dynamic recompilation rather than just using an interpreter, giving you a 5-10x speed boost.

What does that actually mean? You can run all kinds of games on a Pi! If it has an x86 Linux binary, you can probably run it.

What's that? All of your games are in Steam? Well you're in luck, because Steam runs through box86 too!

What's another fun x86 binary that can be run? WINE! Using box86 with WINE gives you the ability to run Windows x86 binaries on an ARM system like a Raspberry Pi.

This guide has you build it and get a bunch of things running on a system where they normally wouldn't work.

Build It

You'll need your favorite ARM-based computer - for this I'll be using a Raspberry Pi 400 since they're still readily available. If you have a Pi 4, that'll work just as well. Older Pis can work but will need a custom kernel if you want to run Wine.

If you're into the whole brevity thing you can skip all of this and just run TwisterOS () which includes everything.

This guide will assume you're running Raspberry Pi OS 32-bit - we need the 32-bit libraries to run everything. There's box64 () to run 64-bit binaries but it needs some additional work if you also want to run 32-bit binaries.

You'll need cmake to build it:

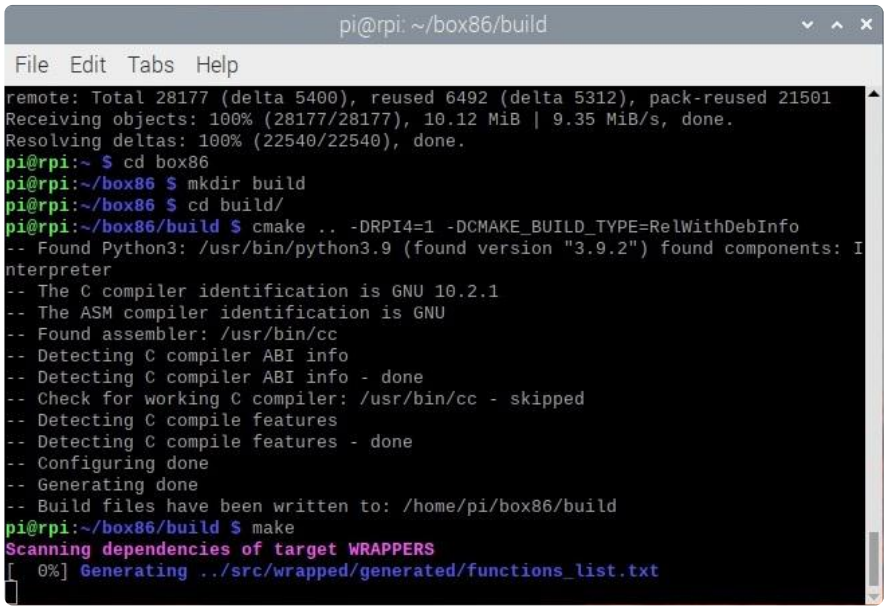
```
sudo apt install cmake
```

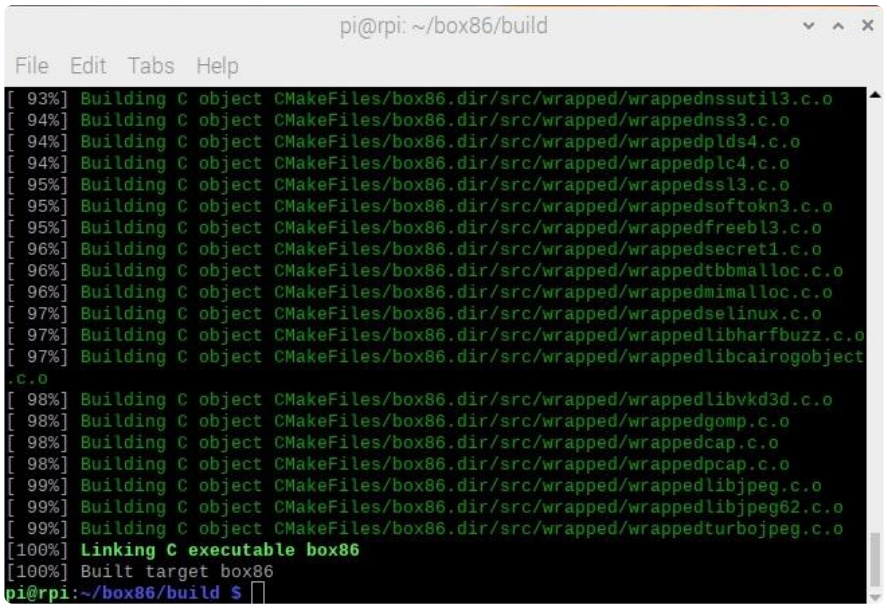
Now get the box86 repo:

```
git clone https://github.com/ptitSeb/box86
```

And build it! This might take a while:

```
cd box86
mkdir build
cd build
cmake .. -DRPI4=1 -DCMAKE_BUILD_TYPE=RelWithDebInfo
make
```





Now install it!

```
sudo make install
sudo systemctl restart systemd-binfmt
```

That last line will restart the service that handles different binary formats - now it knows to pass x86 binaries to box86.

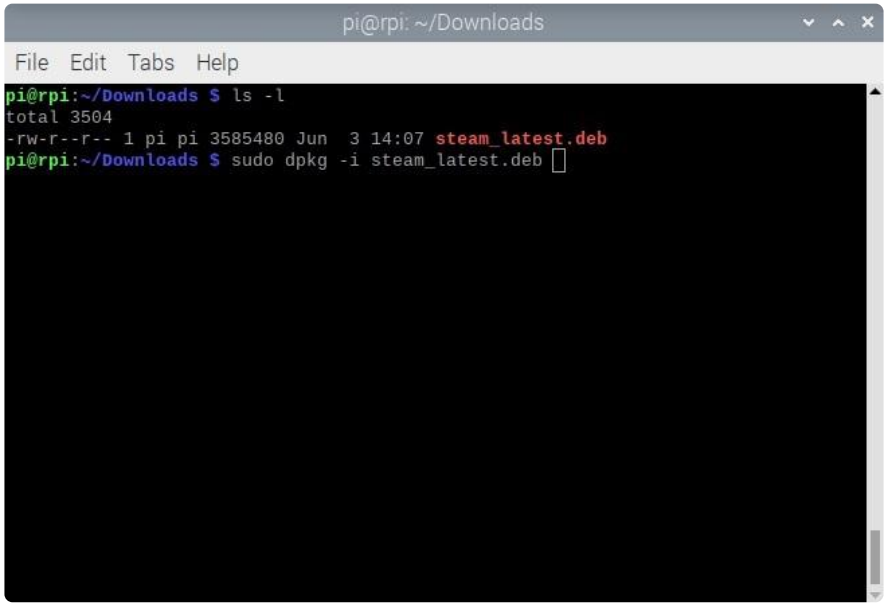
Running Linux Binaries

Now that x86 binaries can be run, it's time to install Steam and play some games!

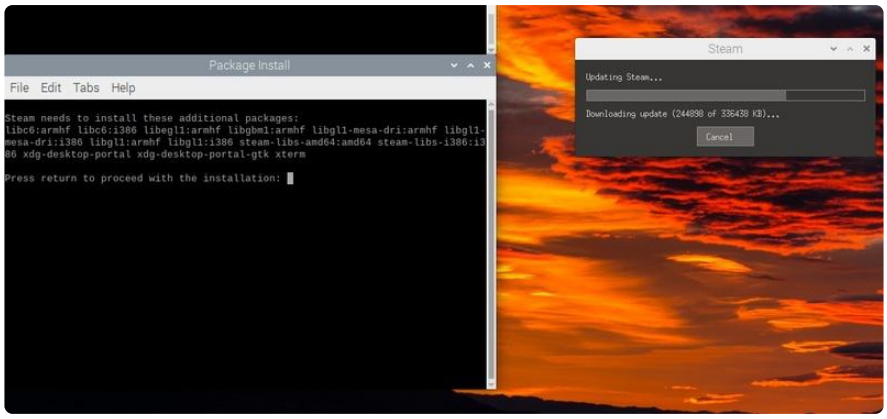
First download the SteamOS debian package from https://store.steampowered.com/about/ () (it's the little Steam icon under the blue INSTALL STEAM button).

Then install the package:

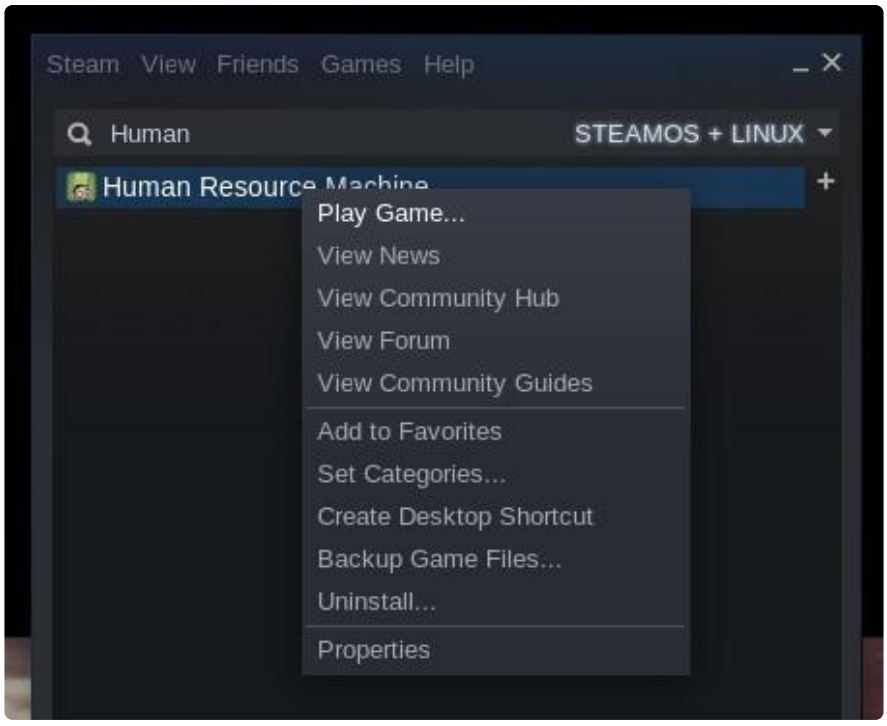
```
sudo dpkg -i steam_latest.deb
```



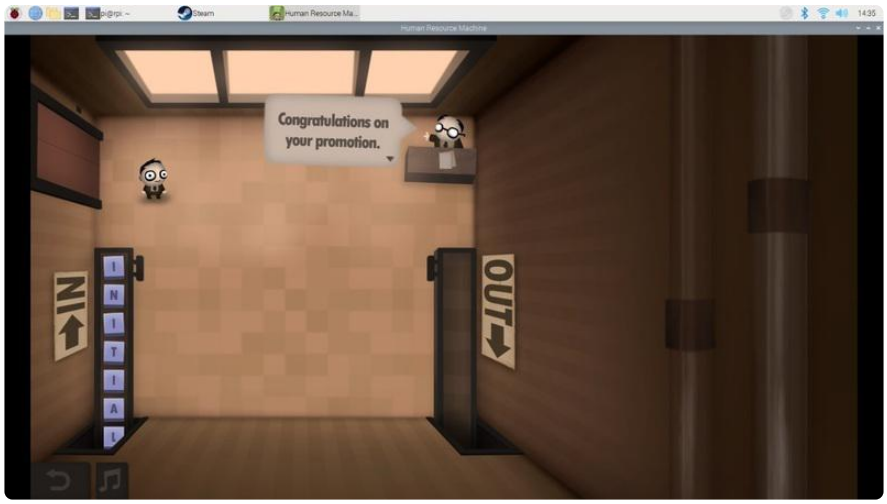
Now you can run Steam from the main menu under Games. It will complain about some missing dependencies and a missing library. Don't worry, box86 is going to handle all of that, Steam just doesn't know it. In the terminal window you can push Ctrl+C to cancel the additional package installation. Let Steam update and eventually it will launch.



Sign in and take a look - things aren't quite right. That's because Steam is a hybrid of 32-bit and 64-bit pieces and only the 32-bit parts will work for us. That's fine, we just can't see some parts like the browser. Go to View - Small Mode to just change to a list of games. In the upper right, change to SteamOS + Linux to see the games you have that will work in Linux. This doesn't differentiate between 32- and 64-bit so not all of them will work. Check the compatibility list () to see if anyone has had luck with a specific game.



Install a game and try running it - some will work right away, some need a few tweaks. Human Resource Machine is a great one that has no problems.



What else can be done?

Using WINE

What's another x86 binary which can be run? WINE!

First get a version of i386 WINE from WineHQ () - you can pick from wine-stable, wine-staging, or wine-devel. Don't get distracted by the ARM versions - those will only run binaries for ARM Windows which is not what is wanted.

If you're using Raspberry Pi OS, you'll need a Pi 4 - earlier models need a 3G/1G memory split custom kernel. Check the Resources page for more info.

```
wget https://dl.winehq.org/wine-builds/debian/dists/bullseye/main/binary-i386/wine-stable-i386_7.0.0.0~bullseye-1_i386.deb
wget https://dl.winehq.org/wine-builds/debian/dists/bullseye/main/binary-i386/wine-stable_7.0.0.0~bullseye-1_i386.deb
```

The i386 file and the other shared file are needed. This method will install wine in your user directory so that you can have multiple versions if you want.

```
dpkg-deb -xv wine-stable-i386_7.0.0.0~bullseye-1_i386.deb wine-installer
dpkg-deb -xv wine-stable_7.0.0.0~bullseye-1_i386.deb wine-installer
mv wine-installer/opt/wine-stable ~/wine
```

Now to get needed symlinks to make everything work. The first line will create a script to ensure WINE is called properly as a 32-bit binary.

```
echo -e '#!/bin/bash\nsetarch linux32 -L '"$HOME/wine/bin/wine "'"$@"' | sudo tee -a /usr/local/bin/wine >/dev/null
```

```
sudo ln -s ~/wine/bin/wineboot /usr/local/bin/wineboot
sudo ln -s ~/wine/bin/winecfg /usr/local/bin/winecfg
sudo ln -s ~/wine/bin/wineserver /usr/local/bin/wineserver
```

```
sudo chmod +x /usr/local/bin/wine /usr/local/bin/wineboot /usr/local/bin/winecfg /usr/local/bin/wineserver
```

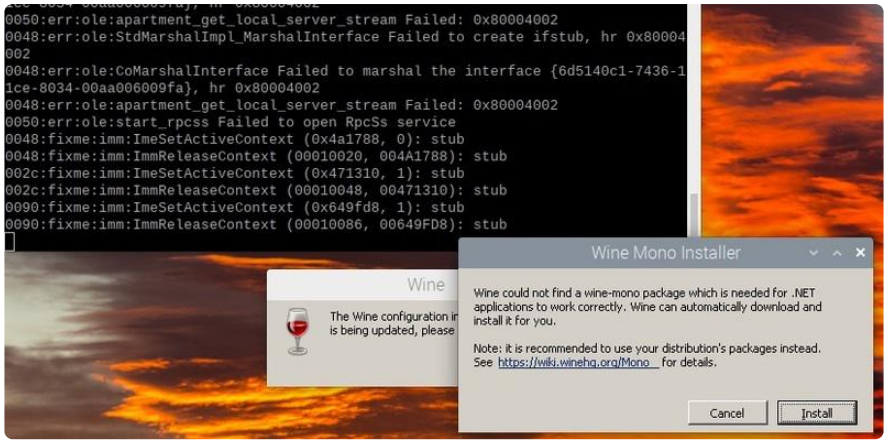
You may need libfaudio - this will get the latest version:

```
wget -r -l1 -np -nd -A "libfaudio0_*~bpo10+1_i386.deb" http://ftp.us.debian.org/debian/pool/main/f/faudio/
```

```
dpkg-deb -xv libfaudio0*.deb libfaudio
sudo cp -TRv libfaudio/usr/ /usr/
```

Now to boot wine in order to create the new wineprefix:

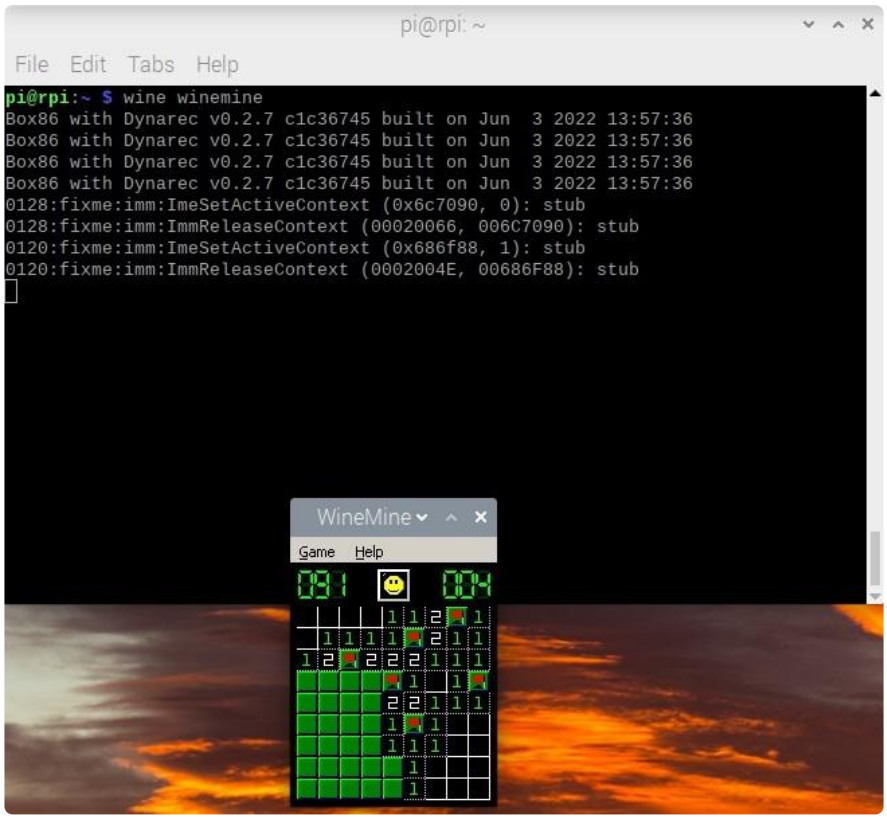
```
wine wineboot
```



Here it will ask you to install Wine Mono - do that so that you can run .NET programs.

Once that's finished, test things out by running winemine!

```
wine winemine
```



Translating a Windows executable into an x86 Linux binary into ARM - nice! Now to get even more ambitious.

Running Windows Programs

You may want to get winetricks in order to adjust some settings:

```
sudo apt install cabextract
wget https://raw.githubusercontent.com/Winetricks/winetricks/master/src/winetricks
sudo chmod +x winetricks && sudo mv winetricks /usr/local/bin/
```

With this you can change the renderer, sound driver, etc. When running it, you need to suppress the box86 banner with `BOX86_NOBANNER=1`. Then you can install some fonts:

```
BOX86_NOBANNER=1 winetricks corefonts
```

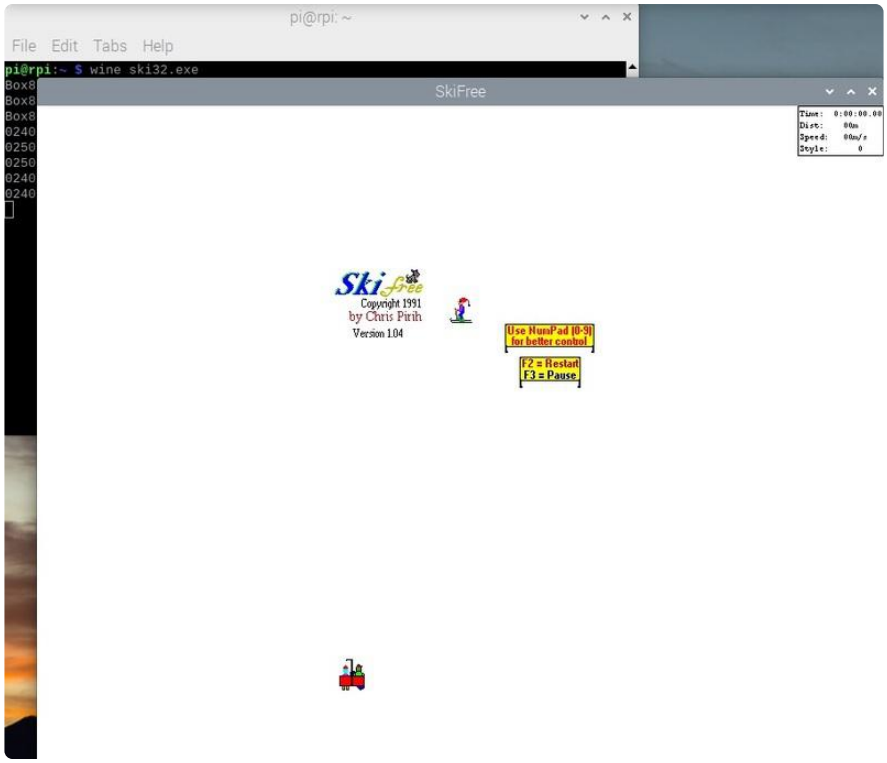
You can also run winetricks without any arguments and it will present a menu full of possibilities. Explore and see what's there, or just click Cancel until it exits.

You can change the DirectDraw renderer if something isn't working well:

```
BOX86_NOBANNER=1 winetricks renderer=gdi
```

That will set it to GDI and might work better with some programs. Setting `renderer=gl` will usually perform better. Check this list of arguments () for everything you can do.

Now what? Why run Windows programs? Well...



How about another classic?

Starcraft running on a Raspberry Pi!

You could do practical things as well, like replace an ancient system running old business software with a modern, tiny, cheap system that can run the same thing.

Some things will segfault when starting but after trying another time or two they run fine. You may need to adjust settings in WINE or use different command line arguments for the program you're trying to run.

Check the compatibility list () for things tagged with Wine to see if someone else has found a way to run it.

More Bits and Resources

There's a 64-bit version of box86 called box64 (), but there are a lot more caveats to deal with. If you also need to run 32-bit binaries, you'll need a multiarch setup and that's a bit beyond this guide.

If you want to run 3D binaries using OpenGL, you'll probably want gl4es ().

More info about WINE and box86 can be found in the box86 docs (). If you need to build a 3G/1G memory split kernel for a Pi 2 or 3, check out this kernel guide () and configure it with `CONFIG_VMSPLIT_3G=y` (Memory split -> 3G/1G user/kernel split in menuconfig).

TwisterOS () is a distribution that includes Box86, WINE, RetroPie, and much more!

You can also contribute! Box86 is released under the MIT license. Check out the Github page () for more.