My Linux Mint Setup

- My Setup
- GPU Passtrough on Linuxmint 21

My Setup

Install Packages

```
apt install vim curl wget tor torsocks torbrowser-launcher vokoscreen-ng wireshark filezilla thunderbird keepassxy ssh openvpn network-manager-openvpn-gnome kolourpaint zsh tmux screen rsync rclone bridge-utils virt-manager qbittorrent aria2 kleopatra rclone-browser remmina-common flatpak gnome-software gnome-software-plugin-flatpak qtox spotify-client nmapsi4 virtualbox virtualbox-guest-additions-iso virtualbox-ext-pack neofetch weechat hexchat newsboat ranger htop ncmpcpp mpd mpv mpc mdadm sshfs traceroute dhcpcd5 nm-tray blueman picom polybar python3-pip autorandr
```

Install Python Packages

```
sudo pip3 install pywal
```

switch shell to ZSH

```
chsh -s /usr/bin/zsh
```

switch screen Shell

```
screen
chsh -s /usr/bin/zsh
```

switch tmux shell

```
tmux
chsh -s /usr/bin/zsh
```

Install Brave Browser

```
sudo apt install apt-transport-https curl

sudo curl -fsSLo /usr/share/keyrings/brave-browser-archive-keyring.gpg https://brave-browser-apt-release.s3.brave.com/brave-browser-archive-keyring.gpg

echo "deb [signed-by=/usr/share/keyrings/brave-browser-archive-keyring.gpg arch=amd64] https://brave-browser-apt-release.s3.brave.com/ stable main"|sudo tee /etc/apt/sources.list.d/brave-browser-release.list

sudo apt update

sudo apt install brave-browser
```

Restore Backups

```
rsync -av --progress user@backupbox:/path-to-backup ~/
```

Create a Linux Bridge

add the following to `/etc/network/interfaces`

```
auto enp4s0

iface enp4s0 inet dhcp
```

add the following to `/etc/network/interfaces.d/br0`

```
# static ip config file for br0 ##
auto br0
iface br0 inet static
```

```
        address 192.168.178.10
        broadcast 192.168.178.255
        netmask 255.255.255.0
        gateway 192.168.178.1
        # If the resolvconf package is installed, you should not edit
        # the resolv.conf configuration file manually. Set name server here
        dns-nameservers 192.168.178.1
        # If you have muliple interfaces such as eth0 and eth1
        # bridge_ports eth0 eth1
        bridge_ports enp4s0
        bridge_stp off       # disable Spanning Tree Protocol
        bridge_waitport 0    # no delay before a port becomes available
        bridge_fd 0          # no forwarding delay
```

Install flatpak

```
sudo flatpak remote-add --if-not-exists flathub https://flathub.org/repo/flathub.flatpakrepo
```

Remember to log in an out - profit!

Install flatpak packages

```
flatpak install anydesk signal simplenote
```

Configure ncmpcpp + mpd

`sudo vim /etc/mpd.conf`

```
music_directory /home/user/Music/files
playlist_directory /home/user/Music/playlists
db_file /home/user/Music/tag_cache
state_file /home/anon/Music/state
sticker_file /home/anon/Music/sticker.sql
user user

audio_output {
```

```
        type            "alsa"
        name            "Alsa for audio sound card"
        mixer_type      "software"      # optional
}


audio_output {
    type                    "fifo"
    name                    "my_fifo"
    path                    "/tmp/mpd.fifo"
    format                  "44100:16:2"
}
```

`systemctl --user enable --now mpd`

GPU Passtrough on Linuxmint 21

0: Pretext

This Setup requires 2 Graphic Cards (1ntegrated and one internal PCI Graphicscard - or - two internal PCI Graphic Cards) Also I am using and AMD CPU, remember to use the equivalents if you have an intel CPU. Also the Sound setup is for Pulseaudio.

Install virt-manager

`sudo apt-get install virt-manager bridge-utils`

Create a Linux Bridge

add the following to `/etc/network/interfaces`

```
auto enp4s0


iface enp4s0 inet dhcp
```

add the following to `/etc/network/interfaces.d/br0`

```
# static ip config file for br0 ##
auto br0
iface br0 inet static
        address 192.168.178.10
        broadcast 192.168.178.255
        netmask 255.255.255.0
        gateway 192.168.178.1
        # If the resolvconf package is installed, you should not edit
        # the resolv.conf configuration file manually. Set name server here
        dns-nameservers 192.168.178.1
```

```
        # If you have muliple interfaces such as eth0 and eth1
        # bridge_ports eth0 eth1
        bridge_ports enp4s0
        bridge_stp off       # disable Spanning Tree Protocol
        bridge_waitport 0    # no delay before a port becomes available
        bridge_fd 0          # no forwarding delay
```

Step 1: Enable IOMMU

edit `/etc/default/grub` at the line `GRUB_CMDLINE_LINUX_DEFAULT` so that it reads like

```
GRUB_CMDLINE_LINUX_DEFAULT="quiet amd_iommu=on iommu=pt"
```

Then reconfigure Grub

```
sudo grub-mkconfig -o /boot/grub/grub.cfg
```

Step 2: Tell VFIO we want to pass through the NVIDIA card

```
lspci -nnk
```

In my case the id of the GPU is `10de:17c8` and the HDMI sound output is `10de:0fb0`. Note the part beneath the GPU where it says `Kernel driver in use: nouveau.`. If everything works correctly that should change by the time we're done. To flag the card for use by VFIO, create the file `sudo nano /etc/default/grub` at the line `GRUB_CMDLINE_LINUX_DEFAULT` with the contents:

```
GRUB_CMDLINE_LINUX_DEFAULT="quiet splash amd_iommu=on iommu=pt kvm.ignore_msrs=1 vfio-pci.ids=10de:17c8,10de:0fb0"
```

Then reconfigure Grub

```
sudo grub-mkconfig -o /boot/grub/grub.cfg
```

Step 3: Create the Windows VM without GPU passthrough

I recommend using `virt-manager` and setting up a regular Windows 10 VM using the default QXL video card before trying to do any passthrough stuff. When creating the VM, make sure to select "Customize before install" and set the Firmware option to "UEFI". Create the VM and go through the Windows installer until you have a working Windows 10 installation with no GPU passthrough, then shut down the VM.

Step 4: Fix the Windows Code 43 error

This error seems to happen because the NVIDIA driver realizes that it's running inside a VM and will disable itself. Since we don't want that we need to "hide" the fact that there's a VM from the driver. KVM has a mechanism for doing that but it's not exposed in virt-manager, so we'll need to edit the XML config for the virtual machine manually. To do that, run:

`sudo virsh edit win10`

where win10 is the name of the VM that you gave when you created it inside virt-manager. You'll need to edit the contents of the tag in the following way:

Inside the tag: add the line:

```
<vendor_id state='on' value='1234567890ab'/>
```

(the actual value of the vendor_id is arbitrary, but it should be a 12 digit hex number).

Inside the tag: add the line:

```
<hidden state='on'/>
```

Inside the tag: add the line:

```
<ioapic driver='kvm'/>
```

The end result should look something like:

```
  <features>
    <acpi/>
    <apic/>
    <hyperv>
      <relaxed state='on'/>
      <vapic state='on'/>
      <spinlocks state='on' retries='8191'/>
      <vendor_id state='on' value='1234567890ab'/>
    </hyperv>
    <kvm>
      <hidden state='on'/>
    </kvm>
    <vmport state='off'/>
    <ioapic driver='kvm'/>
  </features>
```

If you boot the machine up again, the NVIDIA driver should actually work! Windows will probably default to using the GPU as the primary card, which means that the Windows login prompt will likely appear on the display connected to the video card rather than the QXL display that you can see in virt-manager.

Step 5: evdev keyboard and mouse switching

Make sure the first line looks like this:

```
<domain type='kvm' id='1' xmlns:qemu='http://libvirt.org/schemas/domain/qemu/1.0'>
```

See usb devices that you want to pass through (usualy the ones with event in the name)

```
ls -l /dev/input/by-id/
```

edit `/etc/libvirt/qemu.conf`

```
user = "anon"
group = "root"

cgroup_device_acl = [
```

```
        "/dev/null", "/dev/full", "/dev/zero",
        "/dev/random", "/dev/urandom",
        "/dev/ptmx", "/dev/kvm", "/dev/kqemu",
        "/dev/rtc","/dev/hpet",
        "/dev/input/by-id/usb-Dell_Dell_USB_Keyboard-event-kbd",
        "/dev/input/by-id/usb-Logitech_USB-PS_2_Optical_Mouse-event-mouse"
]
```

paste this in the very first line in xml:

```
<domain type='kvm' id='1' xmlns:qemu='http://libvirt.org/schemas/domain/qemu/1.0'>
```

If it doesn’t, replace the first line with that. Next, add the following near the bottom, directly above “”:

```
 <qemu:commandline>
 <qemu:arg value='-object'/>
 <qemu:arg value='input-linux,id=mouse1,evdev=/dev/input/event4'/>
 <qemu:arg value='-object'/>
 <qemu:arg value='input-linux,id=kbd1,evdev=/dev/input/by-id//event3,grab_all=on,repeat=on'/>
 </qemu:commandline>
```

Step 6: switching from PS/2 to VirtIO

As with many other technologies, VirtIO has the capacity to improve evdev virtual input devices’ responsiveness. It is not necessary to remove the PS/2 input devices and replace them. Instead, you can simply add VirtIO input devices to the LibVirt XML file, install the guest drivers, and profit! Find the following section of your XML:

```
<input type='mouse' bus='ps2'/>
<input type='keyboard' bus='ps2'/>
```

Add VirtIO mouse and keyboard devices, so in the end it looks like so:

```
<input type='mouse' bus='virtio'>
        <address type='pci' domain='0x0000' bus='0x00' slot='0x0e' function='0x0'/>
</input>
```

```
<input type='keyboard' bus='virtio'>
        <address type='pci' domain='0x0000' bus='0x00' slot='0x0f' function='0x0'/>
</input>
<input type='mouse' bus='ps2'/>
<input type='keyboard' bus='ps2'/>
```

Step 7: Patch Apparmor

Edit `/etc/apparmor.d/abstractions/libvirt-qemu`

paste:

```
  # Input
  /dev/input/* rw,
```

Setup Audio

in this configuration, we will get the vm sound and pass it on to the host system open `/etc/pulse/daemon.conf` and find/add/uncomment the lines and set these values:

```
default-sample-rate = 44100
...
alternate-sample-rate = 48000
```

Step 8: use Pulse Audio

Make sure the very first line of the file does read

```
<domain type='kvm' xmlns:qemu='http://libvirt.org/schemas/domain/qemu/1.0'>
```

Instead of

```
<domain type='kvm'>
```

Check at the bottom of your config if a line `<qemu:commandline>` exists. If yes make sure to add these options:

```
  <qemu:arg value='-device'/>
  <qemu:arg value='ich9-intel-hda,bus=pcie.0,addr=0x1b'/>
```

```
  <qemu:arg value='-device'/>
  <qemu:arg value='hda-micro,audiodev=hda'/>
  <qemu:arg value='-audiodev'/>
  <qemu:arg value='pa,id=hda,server=unix:/run/user/1000/pulse/native'/>
  </qemu:commandline>
```

In case `<qemu:commandline>` is missing, find the line which ends with `</devices>` and add the following block afterwards:

```
<qemu:commandline>
  <qemu:arg value='-device'/>
  <qemu:arg value='ich9-intel-hda,bus=pcie.0,addr=0x1b'/>
  <qemu:arg value='-device'/>
  <qemu:arg value='hda-micro,audiodev=hda'/>
  <qemu:arg value='-audiodev'/>
  <qemu:arg value='pa,id=hda,server=unix:/run/user/1000/pulse/native'/>
</qemu:commandline>
```

The 1000 in

```
server=unix:/run/user/1000/pulse/native
```

represents your user-id, 1000 is the default (one user) Id.

Open the apparmor libvirt abstractions file `/etc/apparmor.d/abstractions/libvirt-qemu` and append the following:

```
/etc/pulse/client.conf.d/ r,
/etc/pulse/client.conf.d/* r,
/run/user/1000/pulse/native rw,
/home/your-username/.config/pulse/* r,
/home/your-username/.config/pulse/cookie k,
```

Also remember to replace "anon" with your own username

CPU Pinning

According to this Archwiki Article and Forum Post, I edited my XML

https://wiki.archlinux.org/title/PCI_passthrough_via_OVMF#CPU_pinning

https://bbs.archlinux.org/viewtopic.php?pid=1710966#p1710966

XML

```
<domain xmlns:qemu="http://libvirt.org/schemas/domain/qemu/1.0" type="kvm">
  <name>GamingVM</name>
  <uuid>821f1a1d-c836-4c31-a136-2ab789064f56</uuid>
  <metadata>
    <libosinfo:libosinfo xmlns:libosinfo="http://libosinfo.org/xmlns/libvirt/domain/1.0">
      <libosinfo:os id="http://microsoft.com/win/10"/>
    </libosinfo:libosinfo>
  </metadata>
  <memory unit="KiB">8388608</memory>
  <currentMemory unit="KiB">8388608</currentMemory>
  <memoryBacking>
    <source type="memfd"/>
    <access mode="shared"/>
  </memoryBacking>
  <vcpu placement="static">4</vcpu>
  <iothreads>1</iothreads>
  <cputune>
    <vcpupin vcpu="0" cpuset="8"/>
    <vcpupin vcpu="1" cpuset="10"/>
    <vcpupin vcpu="2" cpuset="12"/>
    <vcpupin vcpu="3" cpuset="14"/>
  </cputune>
  <os>
    <type arch="x86_64" machine="pc-q35-6.2">hvm</type>
    <loader readonly="yes" type="pflash">/usr/share/OVMF/OVMF_CODE_4M.ms.fd</loader>
    <nvram>/var/lib/libvirt/qemu/nvram/GamingVM_VARS.fd</nvram>
    <boot dev="hd"/>
  </os>
  <features>
    <acpi/>
    <apic/>
    <hyperv mode="custom">
      <relaxed state="on"/>
      <vapic state="on"/>
      <spinlocks state="on" retries="8191"/>
      <vendor_id state="on" value="1234567890ab"/>
    </hyperv>
```

```
    <kvm>
      <hidden state="on"/>
    </kvm>
    <vmport state="off"/>
    <ioapic driver="kvm"/>
  </features>
  <cpu mode="host-passthrough" check="none" migratable="on">
    <topology sockets="1" dies="1" cores="4" threads="1"/>
    <feature policy="disable" name="smep"/>
  </cpu>
  <clock offset="localtime">
    <timer name="rtc" tickpolicy="catchup"/>
    <timer name="pit" tickpolicy="delay"/>
    <timer name="hpet" present="no"/>
    <timer name="hypervclock" present="yes"/>
  </clock>
  <on_poweroff>destroy</on_poweroff>
  <on_reboot>restart</on_reboot>
  <on_crash>destroy</on_crash>
  <pm>
    <suspend-to-mem enabled="no"/>
    <suspend-to-disk enabled="no"/>
  </pm>
  <devices>
    <emulator>/usr/bin/qemu-system-x86_64</emulator>
    <disk type="file" device="disk">
      <driver name="qemu" type="qcow2"/>
      <source file="/home/anon/ssd/virt/win10.qcow2"/>
      <target dev="sda" bus="sata"/>
      <address type="drive" controller="0" bus="0" target="0" unit="0"/>
    </disk>
    <controller type="pci" index="0" model="pcie-root"/>
    <controller type="pci" index="1" model="pcie-root-port">
      <model name="pcie-root-port"/>
      <target chassis="1" port="0x8"/>
      <address type="pci" domain="0x0000" bus="0x00" slot="0x01" function="0x0"
multifunction="on"/>
    </controller>
    <controller type="pci" index="2" model="pcie-root-port">
      <model name="pcie-root-port"/>
      <target chassis="2" port="0x9"/>
```

```
      <address type="pci" domain="0x0000" bus="0x00" slot="0x01" function="0x1"/>
    </controller>
    <controller type="pci" index="3" model="pcie-root-port">
      <model name="pcie-root-port"/>
      <target chassis="3" port="0xa"/>
      <address type="pci" domain="0x0000" bus="0x00" slot="0x01" function="0x2"/>
    </controller>
    <controller type="pci" index="4" model="pcie-root-port">
      <model name="pcie-root-port"/>
      <target chassis="4" port="0xb"/>
      <address type="pci" domain="0x0000" bus="0x00" slot="0x01" function="0x3"/>
    </controller>
    <controller type="pci" index="5" model="pcie-root-port">
      <model name="pcie-root-port"/>
      <target chassis="5" port="0xc"/>
      <address type="pci" domain="0x0000" bus="0x00" slot="0x01" function="0x4"/>
    </controller>
    <controller type="pci" index="6" model="pcie-root-port">
      <model name="pcie-root-port"/>
      <target chassis="6" port="0xd"/>
      <address type="pci" domain="0x0000" bus="0x00" slot="0x01" function="0x5"/>
    </controller>
    <controller type="pci" index="7" model="pcie-root-port">
      <model name="pcie-root-port"/>
      <target chassis="7" port="0xe"/>
      <address type="pci" domain="0x0000" bus="0x00" slot="0x01" function="0x6"/>
    </controller>
    <controller type="pci" index="8" model="pcie-root-port">
      <model name="pcie-root-port"/>
      <target chassis="8" port="0xf"/>
      <address type="pci" domain="0x0000" bus="0x00" slot="0x01" function="0x7"/>
    </controller>
    <controller type="pci" index="9" model="pcie-root-port">
      <model name="pcie-root-port"/>
      <target chassis="9" port="0x10"/>
      <address type="pci" domain="0x0000" bus="0x00" slot="0x02" function="0x0"
multifunction="on"/>
    </controller>
    <controller type="pci" index="10" model="pcie-root-port">
      <model name="pcie-root-port"/>
      <target chassis="10" port="0x11"/>
```

```
      <address type="pci" domain="0x0000" bus="0x00" slot="0x02" function="0x1"/>
    </controller>
    <controller type="pci" index="11" model="pcie-root-port">
      <model name="pcie-root-port"/>
      <target chassis="11" port="0x12"/>
      <address type="pci" domain="0x0000" bus="0x00" slot="0x02" function="0x2"/>
    </controller>
    <controller type="pci" index="12" model="pcie-root-port">
      <model name="pcie-root-port"/>
      <target chassis="12" port="0x13"/>
      <address type="pci" domain="0x0000" bus="0x00" slot="0x02" function="0x3"/>
    </controller>
    <controller type="pci" index="13" model="pcie-root-port">
      <model name="pcie-root-port"/>
      <target chassis="13" port="0x14"/>
      <address type="pci" domain="0x0000" bus="0x00" slot="0x02" function="0x4"/>
    </controller>
    <controller type="pci" index="14" model="pcie-root-port">
      <model name="pcie-root-port"/>
      <target chassis="14" port="0x15"/>
      <address type="pci" domain="0x0000" bus="0x00" slot="0x02" function="0x5"/>
    </controller>
    <controller type="sata" index="0">
      <address type="pci" domain="0x0000" bus="0x00" slot="0x1f" function="0x2"/>
    </controller>
    <controller type="usb" index="0" model="ich9-ehci1">
      <address type="pci" domain="0x0000" bus="0x00" slot="0x1d" function="0x7"/>
    </controller>
    <controller type="usb" index="0" model="ich9-uhci1">
      <master startport="0"/>
      <address type="pci" domain="0x0000" bus="0x00" slot="0x1d" function="0x0"
multifunction="on"/>
    </controller>
    <controller type="usb" index="0" model="ich9-uhci2">
      <master startport="2"/>
      <address type="pci" domain="0x0000" bus="0x00" slot="0x1d" function="0x1"/>
    </controller>
    <controller type="usb" index="0" model="ich9-uhci3">
      <master startport="4"/>
      <address type="pci" domain="0x0000" bus="0x00" slot="0x1d" function="0x2"/>
    </controller>
```

```
<interface type="network">
  <mac address="52:54:00:29:79:98"/>
  <source network="default"/>
  <model type="e1000e"/>
  <address type="pci" domain="0x0000" bus="0x01" slot="0x00" function="0x0"/>
</interface>
<serial type="pty">
  <target type="isa-serial" port="0">
    <model name="isa-serial"/>
  </target>
</serial>
<console type="pty">
  <target type="serial" port="0"/>
</console>
<input type="mouse" bus="virtio">
  <address type="pci" domain="0x0000" bus="0x00" slot="0x0e" function="0x0"/>
</input>
<input type="keyboard" bus="virtio">
  <address type="pci" domain="0x0000" bus="0x00" slot="0x0f" function="0x0"/>
</input>
<input type="mouse" bus="ps2"/>
<input type="keyboard" bus="ps2"/>
<audio id="1" type="none"/>
<hostdev mode="subsystem" type="pci" managed="yes">
  <source>
    <address domain="0x0000" bus="0x01" slot="0x00" function="0x0"/>
  </source>
  <address type="pci" domain="0x0000" bus="0x03" slot="0x00" function="0x0"/>
</hostdev>
<hostdev mode="subsystem" type="pci" managed="yes">
  <source>
    <address domain="0x0000" bus="0x01" slot="0x00" function="0x1"/>
  </source>
  <address type="pci" domain="0x0000" bus="0x04" slot="0x00" function="0x0"/>
</hostdev>
<redirdev bus="usb" type="spicevmc">
  <address type="usb" bus="0" port="1"/>
</redirdev>
<redirdev bus="usb" type="spicevmc">
  <address type="usb" bus="0" port="2"/>
</redirdev>
```

```
    <memballoon model="virtio">
      <address type="pci" domain="0x0000" bus="0x05" slot="0x00" function="0x0"/>
    </memballoon>
  </devices>
  <qemu:commandline>
    <qemu:arg value="-object"/>
    <qemu:arg value="input-linux,id=kbd1,evdev=/dev/input/event3,grab_all=on,repeat=on"/>
    <qemu:arg value="-object"/>
    <qemu:arg value="input-linux,id=mouse1,evdev=/dev/input/event4"/>
    <qemu:arg value="-device"/>
    <qemu:arg value="ich9-intel-hda,bus=pcie.0,addr=0x1b"/>
    <qemu:arg value="-device"/>
    <qemu:arg value="hda-micro,audiodev=hda"/>
    <qemu:arg value="-audiodev"/>
    <qemu:arg value="pa,id=hda,server=unix:/run/user/1000/pulse/native"/>
  </qemu:commandline>
</domain>
```