

Equipment required to duplicate the results

ASUS Crosshair VII HERO motherboard – mine does not have wifi attached.

AMD RYZEN 5700X

Corsair memory – 16 Gb total

SSD for the OS – I am using an old 35 Gb SSD that is more than sufficient. Blitz is using an unspecified SSD drive.

Use a good cooler – I am using a NOCTUA with a little fan that I run very slowly. I cannot hear it and DIETPI reports that the CPU is running very cool – even at this point before the magic is added.

I am using a WIFI to LAN adapter

Blitz is using a CORSAIR ATX supply while he works on his own linear supply. I am using an HDPLEX 400 watts linear supply which is no longer available. It is what I was using with WTF.

As Blitz stated before he found x86 to sound better. I assume the 8 cores available with this CPU is also a reason to use this.

This is a client-server setup so you need to use something to tell the computer what to do.

I am using the same computer that has REW and my DSP controller installed. This is plenty convenient for me. Blitz is using a Ipad and using SOUNDIROCK because he likes it. I figure most will have their own way of handling this.

If you use the ASUS Crosshair VII you will have to update the BIOS before it will boot. The board has a trick feature that allows you to update the BIOS before you install the CPU or anything else. You would be best looking at how to do this on the internet. Don't be dumb like me and have to find out even though I was told this had to be done.

The AMD CPU has no graphics stuff so you will have to use a video card initially. After we go to .ssh you can remove the card.

Now is a good time to go through the BIOS and turn everything off you think you can turn off. This is not settled as to what is best and where to stop.

The two most important things to change are in ADVANCED settings – make sure you choose ECO MODE to limit power to 45 watts and to disengage SMT which is what AMD calls Hyper-threading. If you do not turn this off you will see 16 cores instead of 8.

I disabled all of the USB ports other than the one I am using.

Download DIETPI for the platform you are going to use. Copy it to some kind of USB device.

Set your BIOS BOOT to use this USB device first.

Start your system and when the screen comes up ask for DIETPI to be installed.

It will turn off after the initial installation step and you should remove the USB device and then turn the board back on.

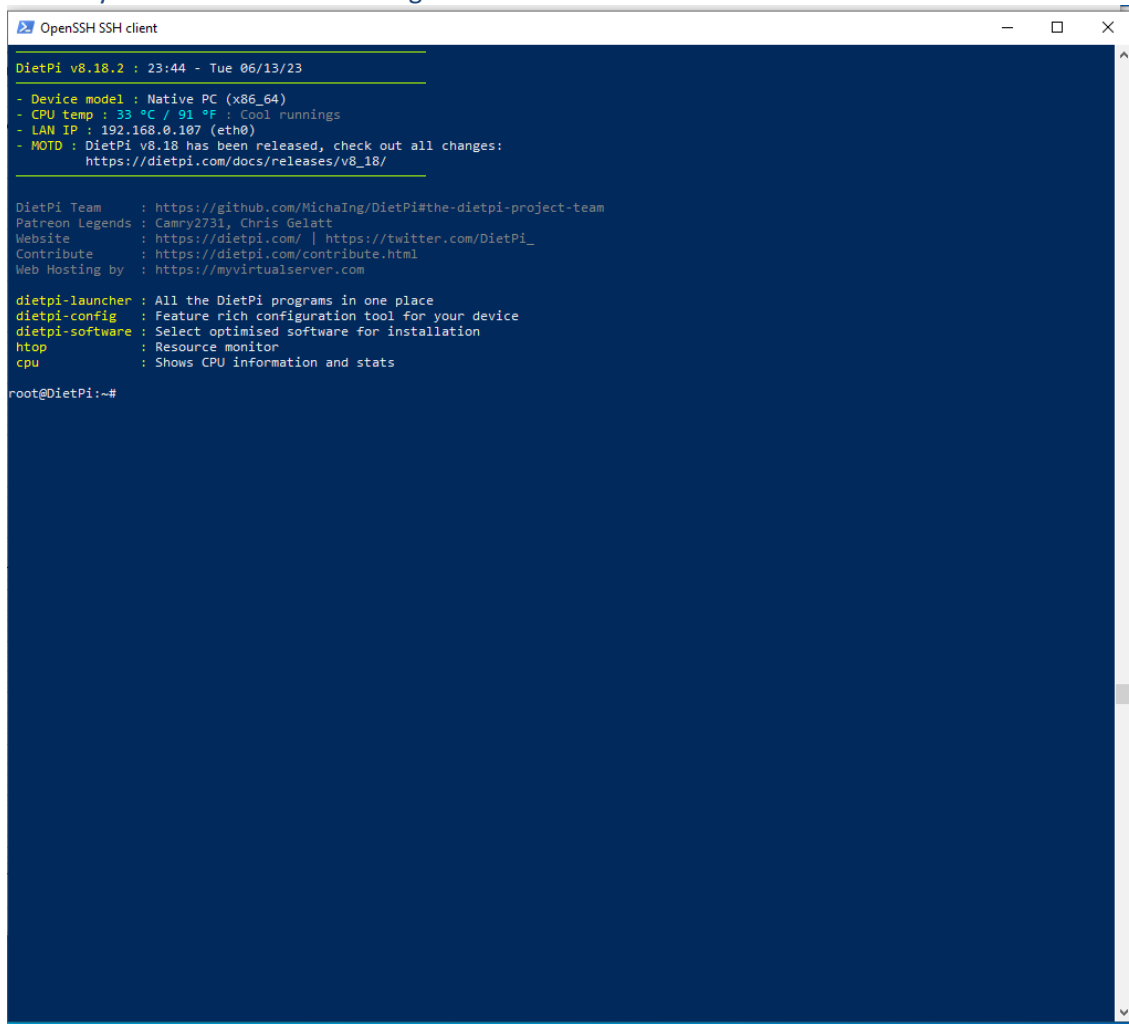
RETURN key to logon

type ROOT and then it will ask for the password and this is DIETPI.

You will then go through the remainder of the setup process.

Do not agree to the feedback option – choose your language and keyboard. Do not install the series port stuff. You can change your password during this if you think you need to.

This the first screen you will see after rebooting:



```
OpenSSH SSH client

DietPi v8.18.2 : 23:44 - Tue 06/13/23

- Device model : Native PC (x86_64)
- CPU temp : 33 °C / 91 °F : Cool runnings
- LAN IP : 192.168.0.107 (eth0)
- MOTD : DietPi v8.18 has been released, check out all changes:
        https://dietpi.com/docs/releases/v8_18/

DietPi Team : https://github.com/MichaIng/DietPi#the-dietpi-project-team
Patreon Legends : Camry2731, Chris Gelatt
Website : https://dietpi.com/ | https://twitter.com/DietPi_
Contribute : https://dietpi.com/contribute.html
Web Hosting by : https://myvirtualserver.com

dietpi-launcher : All the DietPi programs in one place
dietpi-config : Feature rich configuration tool for your device
dietpi-software : Select optimised software for installation
htop : Resource monitor
cpu : Shows CPU information and stats

root@DietPi:~#
```

Fill in the password and ENTER

Now you see the IP address of DIETPI – this is what you will use to make an .ssh connection to the client.

At this point you need to choose your client and that will depend on personal taste and what operating system is being used for the client. I chose MPDctrl – there are not too many for WINDOWS and this one does what I want.

Now is a good time to make sure it is working.

From this point on it is easier to work on DIETPI on another computer with .ssh.

In my case with WINDOWS 10 I type in WINDOWS PowerShell (admin) `ssh root@dietpi`

So give it a try.

(I only use WINDOWS since neither of the other programs I need did very well on a MAC mini. REW on WINDOWS is a completely different thing, so much better ad it doesn't crash.)

I am using a WIFI to LAN adapter to access my network.

Initially it will want to fingerprint you – Just say yes and then you can enter the password.

If you make a change to DIETPI, like re-installing the OS, you will have to go through this again but in that case you will need to erase the saved key in the .ssh folder first. It won't let you connect until you do.

Once connected – unless you changed it during installation – the default password is dietpi

Now we can begin making DIETPI work for us.

It is a good idea to familiarize yourself with what all of the modules do. Good way to get used to how you have to work with DIETPI – it took me a little while to get used to how you have to use the keyboard to do what you want – no mouse is used.

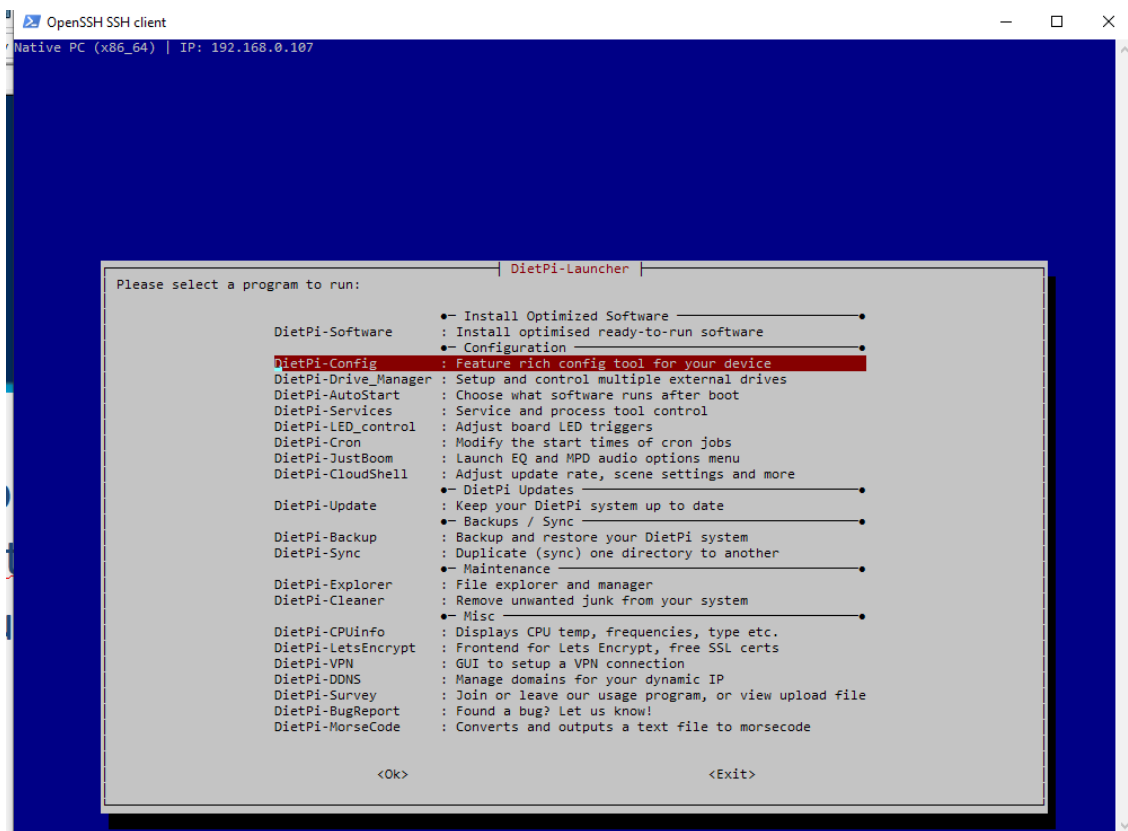
After you have your client working you can remove the video card. The only things connected to the motherboard will be the USB cable to your DAC, or if you use something like the MUTECH MC3+ USB which DIETPI will recognize as your “DAC”. The sata cable for the music drive, in my case, and the LAN cable for NAS. Blitz uses NAS and I am not sure if that is integrated with his controller. I guess it has to be, again, I know nothing about NAS.

You will need to turn the computer off to remove the video card. To turn it off use the command POWEROFF – to reboot is REBOOT.

Now we turn it back on again and go through the login process.

Type dietpi-launcher ENTER (you can get to all of the modules with this – you could also type dietpi-<whatever module you want> and ENTER)

You can get to everything with this one – it can take a little practice to get the hang of how you have to yes, no and esc to get around with the keyboard. It is not necessarily intuitive. You would have to be determined to screw things up so don't be afraid to see what happens. The best way to learn.

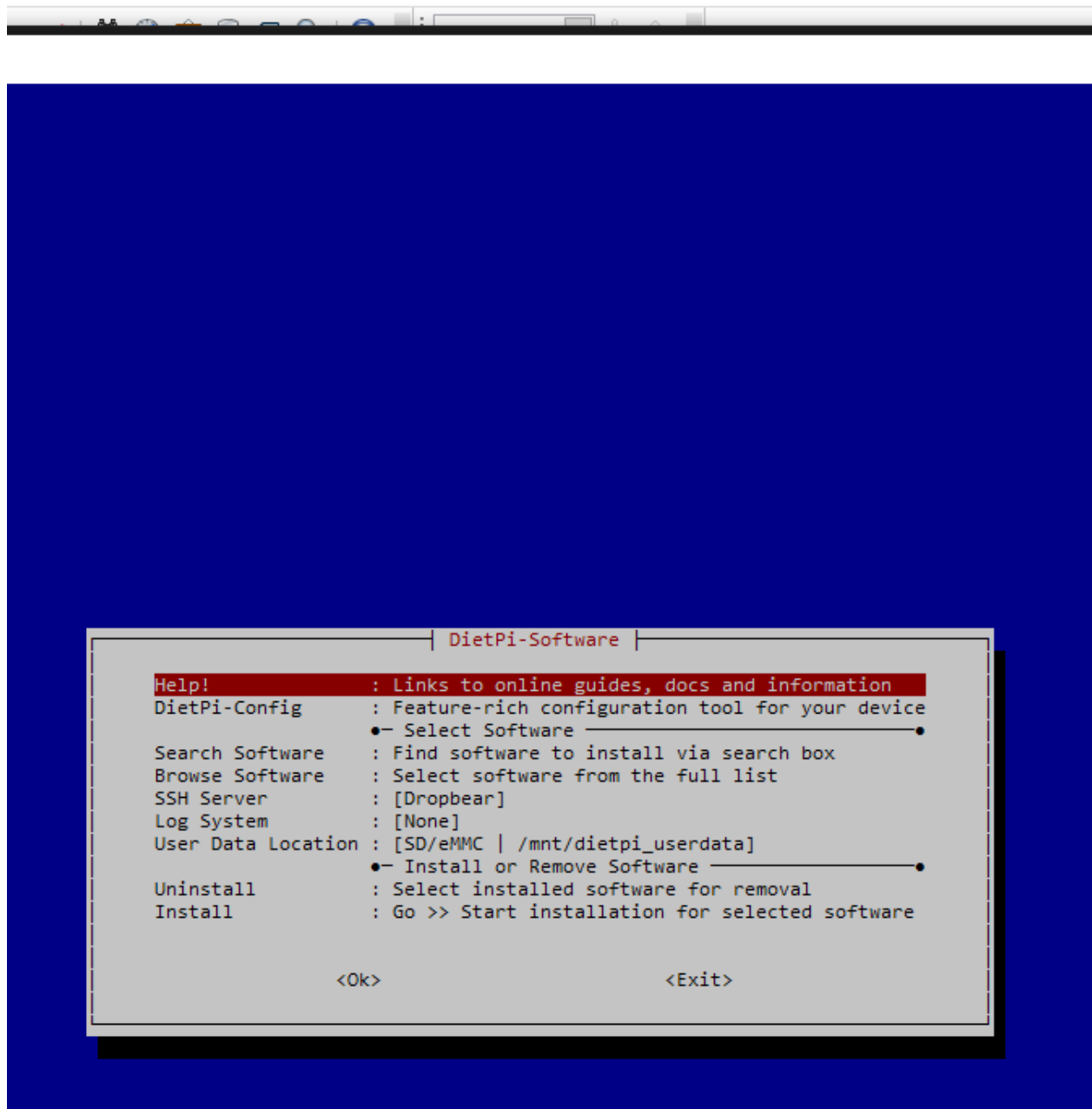


Start with software – scroll down to BROWSE SOFTWARE – ENTER

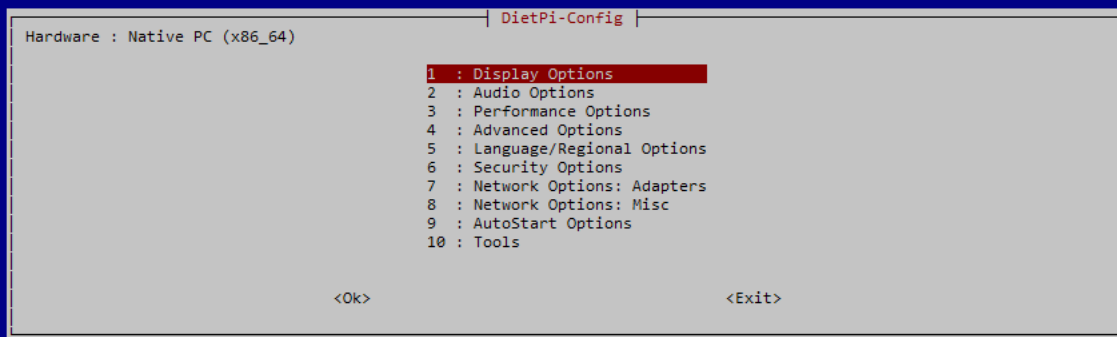
Scroll down to MPD – use the space bar to select – this all we will use

ESC out – SCROLL to INSTALL SOFTWARE – and follow the simple instructions

This will take awhile since it will also install ALSA



Now select DIETPI CONFIG



Select AUDIO OPTIONS

Select SOUND CARD - ENTER

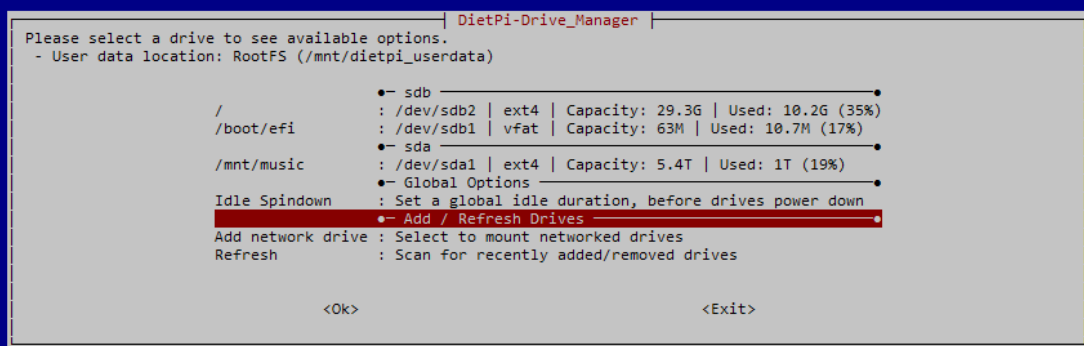
Scroll to your sound card - ENTER

ESC to go back to AUDIO OPTIONS

ESC to return to LAUNCHER

Now return to DIETPI-LAUNCHER select DIETPI-DRIVE_MANAGER – ENTER

I am assuming you will use an attached drive -if it is NAS Blitz will need to add instructions though I am assuming if you already have that you know to connect it.



You will see your drives – scroll down to your drive with your music files – ENTER You can here I have named by drive music

Yours will be the one that is unmounted. And most likely quite large in comparison to your OS drive.

You will then choose MOUNT – ENTER – and to make things easy remove all of the gibberish and name it something like music – ENTER – ESC to DIETPI-LAUNCHER

We need to tell MPD where your music is and which DAC to use. This is done in the MPD Config-File which we edit with the command: nano /etc/mpd.conf

Please study this file carefully and make the following changes (the # is important !):

Blitz's /etc/mpd.conf file looks like this (adjust to to your dac and music directory):

```
music_directory      "/mnt/Musik"
use the name of you NAS or, it like me and unlike Blitz, the name you have given your attached music drive. I name mine
music – this drive is mounted and named in dietpi-drive_manager
playlist_directory   "/var/lib/mpd/playlists"
db_file              "/var/lib/mpd/tag_cache"
log_file              "/var/log/mpd/mpd.log"
pid_file              "/run/mpd/pid"
I left the four above as default in my setup – when I would use these settings I could no longer reach MPD – I figure this
is because of Blitz's attached drive
state_file            "/var/lib/mpd/state"
sticker_file          "/var/lib/mpd/sticker.sql"
input_cache {
size "4 GB"
}
filesystem_charset    "UTF-8"
audio_buffer_size     "8192"
buffer_before_play    "100%"

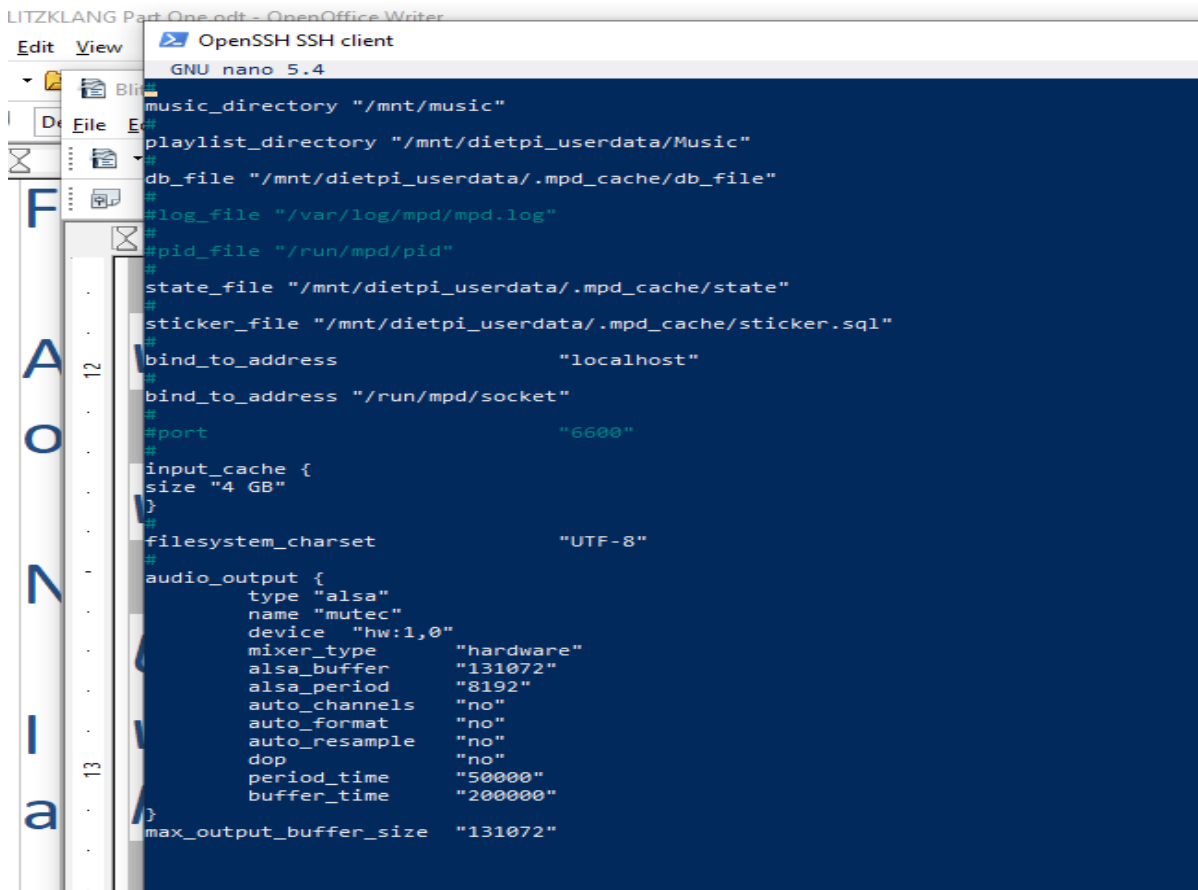
audio_output {
    type "alsa"
    name "Andrea" whatever name you choose
    device "hw:1,0" what you chose in dietpi-config/Audio Options/DAC

mixer_type "hardware" Use this setting for fixed volume/no control – use “software” if you want the system to be able
to adjust volume. Hardware is better.

    alsa_buffer "131072"
    alsa_period "8192"
    auto_channels "no"
    auto_format  "no"
    auto_resample "no"
    dop "no"
    period_time "50000"
    buffer_time "200000"
}

max_output_buffer_size "131072"
```

You can minimize the size of this folder along with making it easier to deal with by removing all of the text. Not that it makes a difference for sound quality. Mine is slightly larger than Blitz's – kept those areas he is using which I figure are for NAS in case I made change in the future- the only parts that are active are in white type.



```
music_directory "/mnt/music"
playlist_directory "/mnt/dietpi_userdata/Music"
db_file "/mnt/dietpi_userdata/.mpd_cache/db_file"
#log_file "/var/log/mpd/mpd.log"
#pid_file "/run/mpd/pid"
state_file "/mnt/dietpi_userdata/.mpd_cache/state"
sticker_file "/mnt/dietpi_userdata/.mpd_cache/sticker.sql"
bind_to_address "localhost"
bind_to_address "/run/mpd/socket"
#port "6600"
input_cache {
size "4 GB"
}
filesystem_charset "UTF-8"
audio_output {
type "alsa"
name "mute"
device "hw:1,0"
mixer_type "hardware"
alsa_buffer "131072"
alsa_period "8192"
auto_channels "no"
auto_format "no"
auto_resample "no"
dop "no"
period_time "50000"
buffer_time "200000"
}
max_output_buffer_size "131072"
```

You always need to reboot after making changes, any changes, to conf files or stop and restart a service, which is a bit more taping but is faster. (Did you mean taping, I am not familiar with the term)

Now you should be able to hear the thing play music.

I had no idea that MPD sounded like this – I have to say I think Blitz's choice of motherboard and CPU has much to do with what you will first hear.

After getting a handle on this sound we move on to the next step.

Please install a tool to understand precisely what is going on with your cpu frequency and your governor and cpu driver:

```
apt-get install -y cpufrequtils
```

After install please type

```
cpufreq-info
```

You will see a similar picture like this, but with different content. Please make a screenshot and post it to the thread. This is a screen from the finished version. Yours will report much higher speeds.

Blitzklang.odt - OpenOffice Writer

File Edit View Insert Format Table Tools Window Help

OpenSSH SSH client

```
analyzing CPU 2:
driver: amd-pstate
CPUs which run at the same hardware frequency: 2
CPUs which need to have their frequency coordinated by software: 2
maximum transition latency: 131 us.
hardware limits: 550 MHz - 4.66 GHz
available cpufreq governors: userspace, performance, schedutil
current policy: frequency should be within 550 MHz and 4.66 GHz.
    The governor "userspace" may decide which speed to use
    within this range.
current CPU frequency is 550 MHz.
analyzing CPU 3:
driver: amd-pstate
CPUs which run at the same hardware frequency: 3
CPUs which need to have their frequency coordinated by software: 3
maximum transition latency: 131 us.
hardware limits: 550 MHz - 4.66 GHz
available cpufreq governors: userspace, performance, schedutil
current policy: frequency should be within 550 MHz and 4.66 GHz.
    The governor "userspace" may decide which speed to use
    within this range.
current CPU frequency is 550 MHz.
analyzing CPU 4:
driver: amd-pstate
CPUs which run at the same hardware frequency: 4
CPUs which need to have their frequency coordinated by software: 4
maximum transition latency: 131 us.
hardware limits: 550 MHz - 4.66 GHz
available cpufreq governors: userspace, performance, schedutil
current policy: frequency should be within 550 MHz and 4.66 GHz.
    The governor "userspace" may decide which speed to use
    within this range.
current CPU frequency is 550 MHz.
analyzing CPU 5:
driver: amd-pstate
CPUs which run at the same hardware frequency: 5
CPUs which need to have their frequency coordinated by software: 5
maximum transition latency: 131 us.
hardware limits: 550 MHz - 4.66 GHz
available cpufreq governors: userspace, performance, schedutil
current policy: frequency should be within 550 MHz and 4.66 GHz.
    The governor "userspace" may decide which speed to use
    within this range.
current CPU frequency is 550 MHz.
analyzing CPU 6:
driver: amd-pstate
CPUs which run at the same hardware frequency: 6
CPUs which need to have their frequency coordinated by software: 6
maximum transition latency: 131 us.
hardware limits: 550 MHz - 4.66 GHz
available cpufreq governors: userspace, performance, schedutil
current policy: frequency should be within 550 MHz and 4.66 GHz.
    The governor "userspace" may decide which speed to use
    within this range.
current CPU frequency is 550 MHz.
analyzing CPU 7:
driver: amd-pstate
CPUs which run at the same hardware frequency: 7
CPUs which need to have their frequency coordinated by software: 7
maximum transition latency: 131 us.
hardware limits: 550 MHz - 4.66 GHz
available cpufreq governors: userspace, performance, schedutil
current policy: frequency should be within 550 MHz and 4.66 GHz.
    The governor "userspace" may decide which speed to use
    within this range.
current CPU frequency is 550 MHz.
root@DietPi:~#
```

Type here to search

Missing CPU0 & 1 – they say the same things

Before we start to make the tickless kernel let's check our current version

At this point Blitz assumes we are getting the idea of how to use LINUX

```
uname -r
```

I found I had to do uname first and then ask for uname -r

This will show you your current Kernel version and you should see something like 5.10

That is what both Blitz and I saw

We will be downloading the kernel from kernel.org (and yes, we want exactly this version of the kernel)

```
wget https://cdn.kernel.org/pub/linux/kernel/v5.x/linux-5.19.17.tar.gz
```

```
wget https://cdn.kernel.org/pub/linux/kernel/v5.x/linux-5.19.17.tar.sign
```

```
tar xvzf linux-5.19.17.tar.gz
```

This extracts the files we just downloaded

```
cd linux-5.19.17
```

We work on the kernel in this directory so the original kernel is not affected. If you mess up you will still have music if you are using it already and you should since it already sounds good

So, you downloaded, unpacked it and have gone into its folder

Now we need to install some tools to build it:

```
apt-get install build-essential linux-source bc kmod cpio flex libncurses5-dev  
libelf-dev libssl-dev dwarves bison
```

You will be asked to say yes to proceed with the install

Now we copy your old configuration from dietpi into the new kernel version:

```
make olddefconfig
```

Now we make the fun part, so you can select what you want to do and change the dietpi config file:

make menuconfig

Now for the configuration.

BY the way...please look over each setting carefully, not just those I explicitly named. Yours should look like mine (Blitz's).

DO NOT CHANGE ANY OTHER stuff !!! Or your Linux Kernel might be smoked. I smoked a dozen of kernels.

I was on my way to having that many

You find in the left upper corner the path of the menu where you do the settings, if you cant find stuff.

You will have to enlarge the screen and move over to the left side to see this – then you will have to move back to where the work is done.

It is a good idea to sure you are in the folder where our newly downloaded kernel is, which we have done previously by

cd linux-5.19.17

Type

make menuconfig

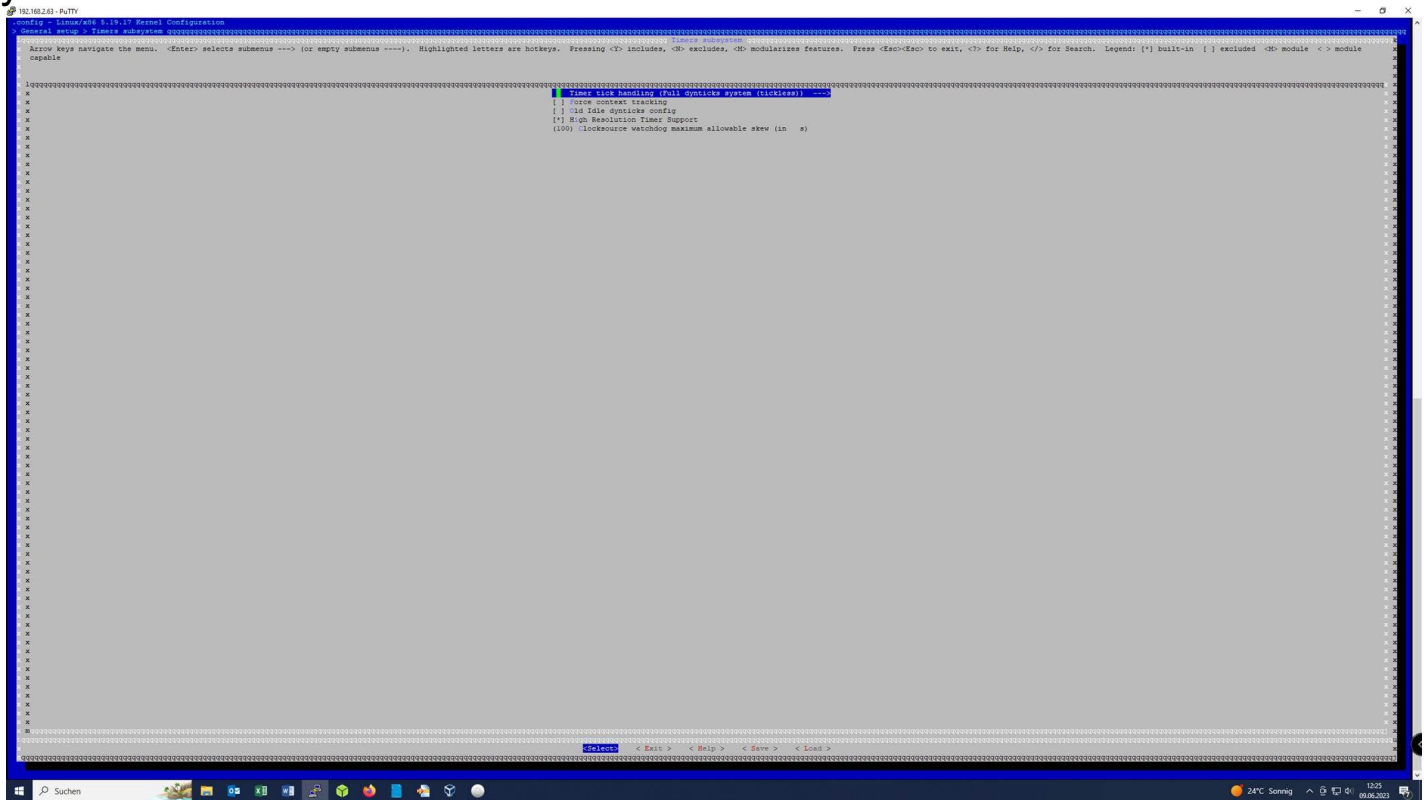
(please ensure that the terminal windows is large enough so the whole menu can be displayed)

This is now what we want: check that yours looks like mine, sometimes you will need to go into the menus. There will be a screenshot to show you when this happens.

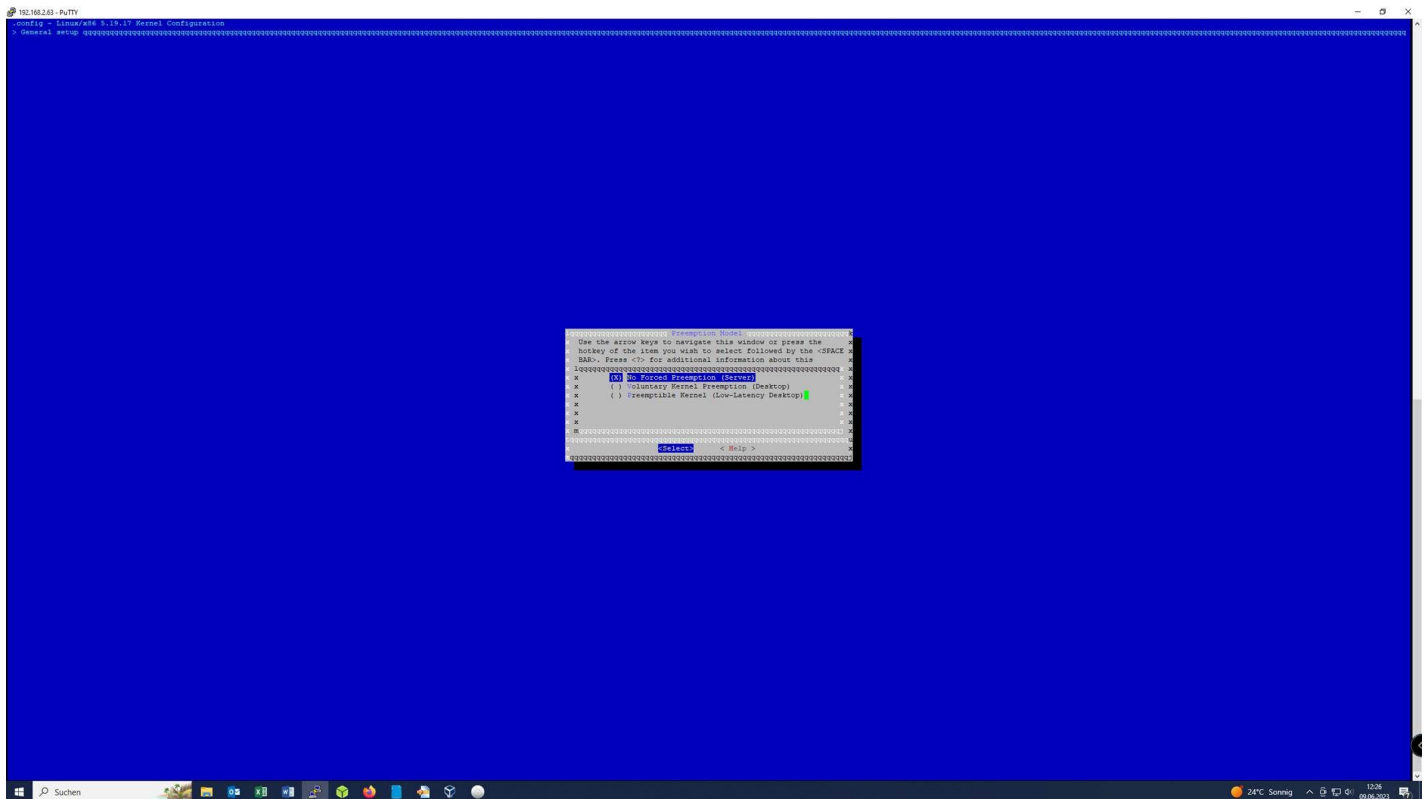
Press Y to get the X in the box – Press N to delete it – Press M to specify MODULE

We disable anything for spec. Vulnerabilities and Virtualization:
Third & fourth lines

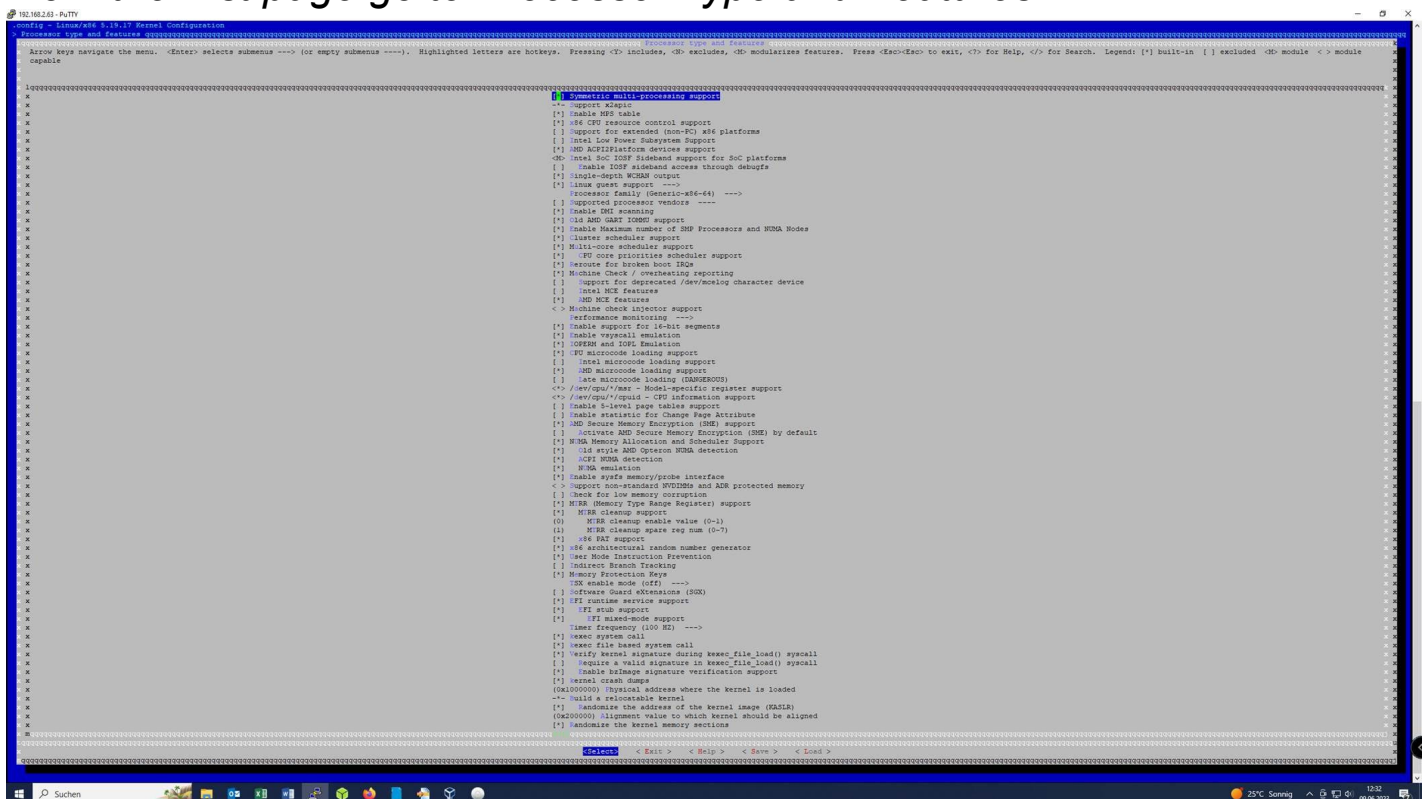
We want to setup tickless here and high resolution timer:
Where you see the green dot **TIMER TICK HANDLING** – ENTER – and
choose **FULL DYNTICKS SYSTEM (tickless)** – ENTER – when you return
your screen will look like this



We use the server mode here to minimize the thread overhead handling.
We want no overhead and max throughput instead of low latency.
*Path General/Preemption Model – ENTER and choose NO FORCED
PREEMPTION – ENTER and you will see your choices*

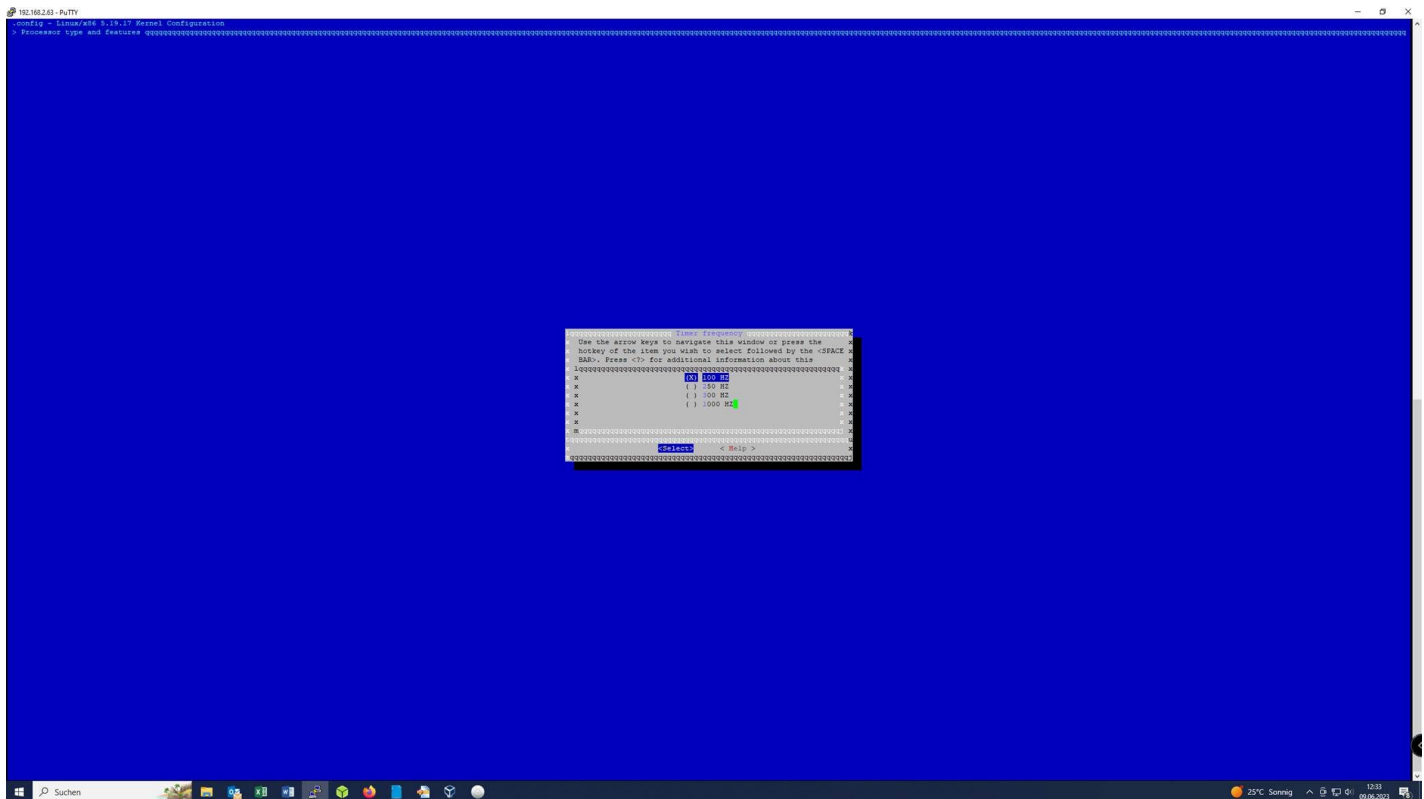


We get rid of the intel stuff and enable all the AMD stuff per below
From the first page go to Processor Type and Features



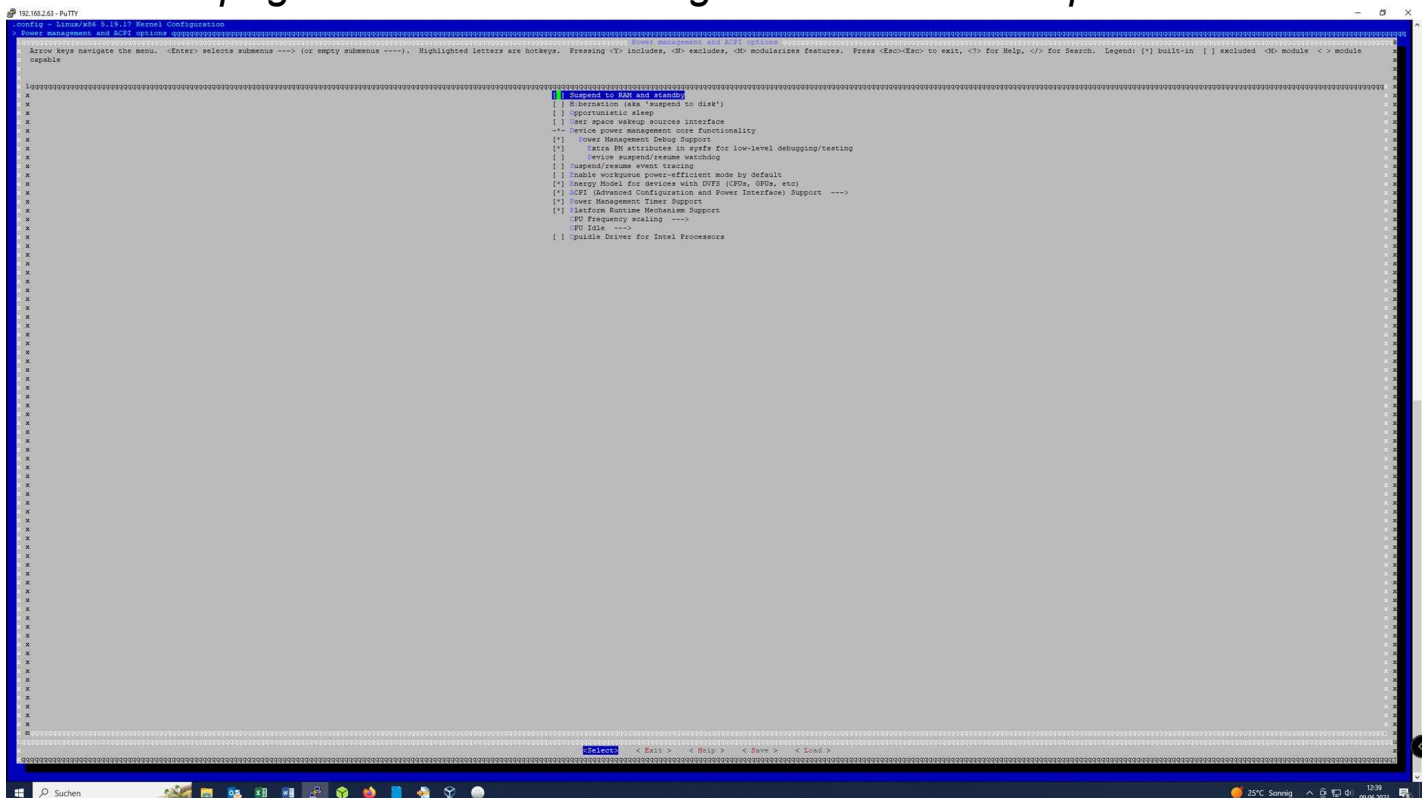
We set the timer frequency for highest throughput to 100HZ, less interrupts, less noise.

*This is on the Processor Type and Features page -TIMER FREQUENCY
ENTER and then select 100 Hz - ENTER*

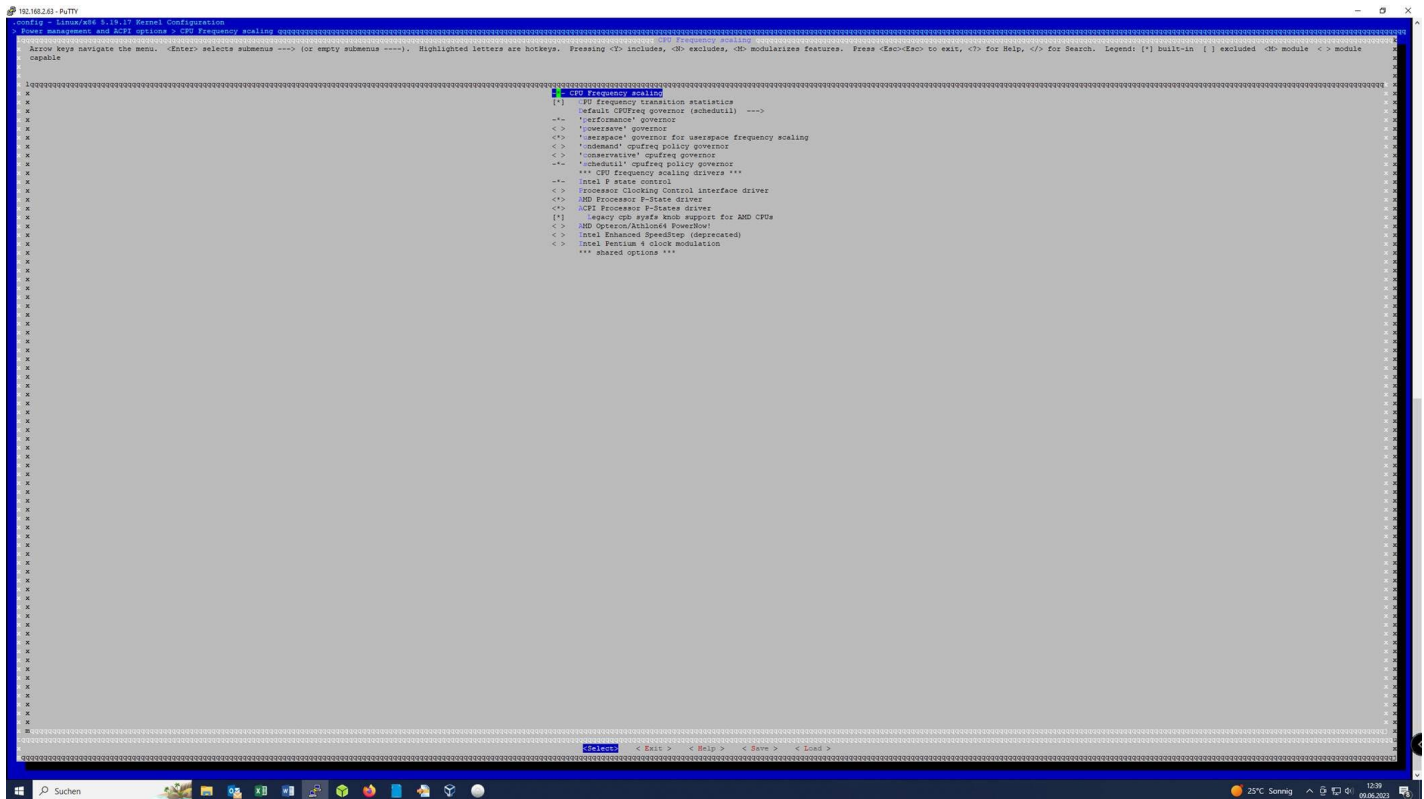


No Hibernation and stuff:

On the first page select Power Management and ACPI Options



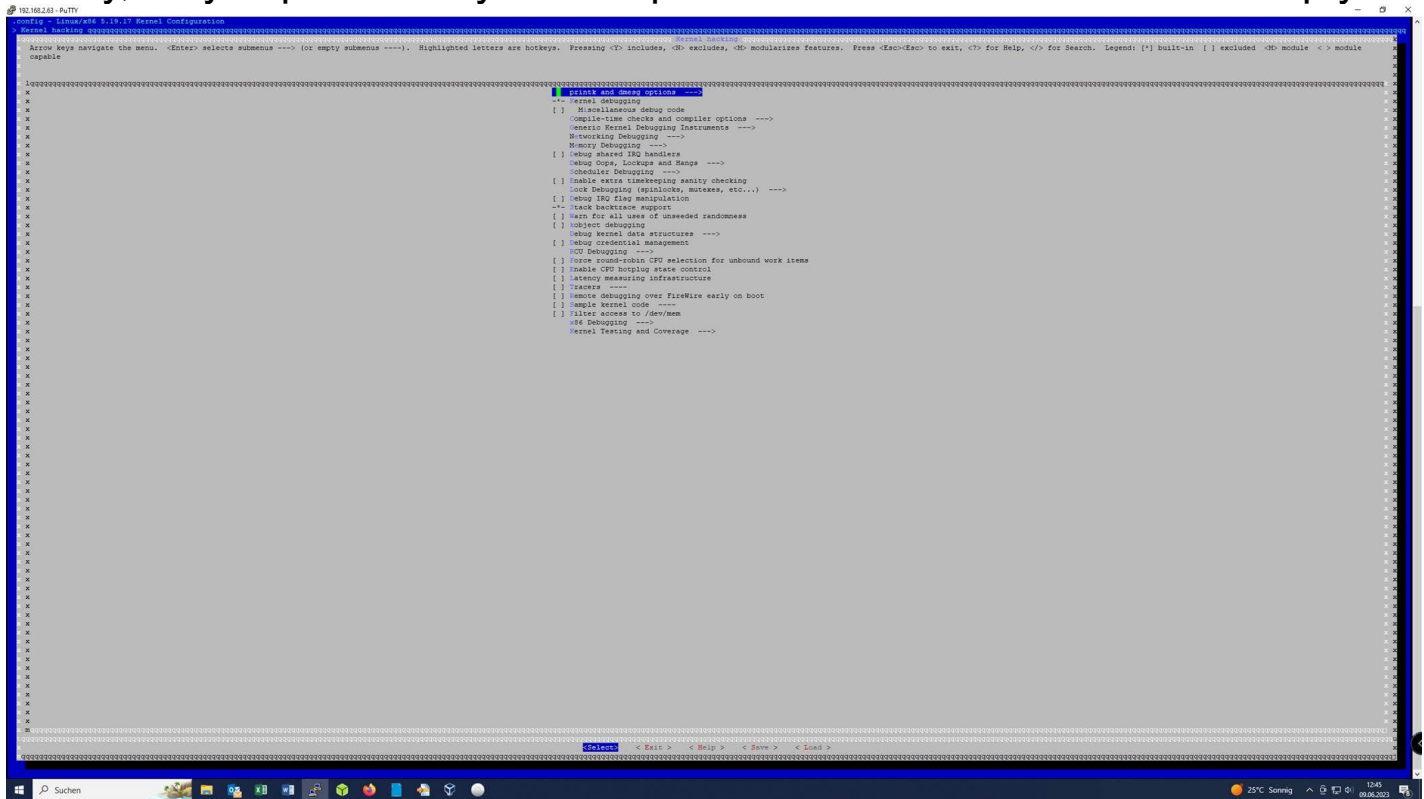
Governor Userspace enabled. AMD P-state & Driver enabled:
*Within Power Management and ACPI Options is the CPU Frequency
Scaling option - ENTER*



*Within Power Management and ACPI Options is the CPU Idle option -
ENTER*

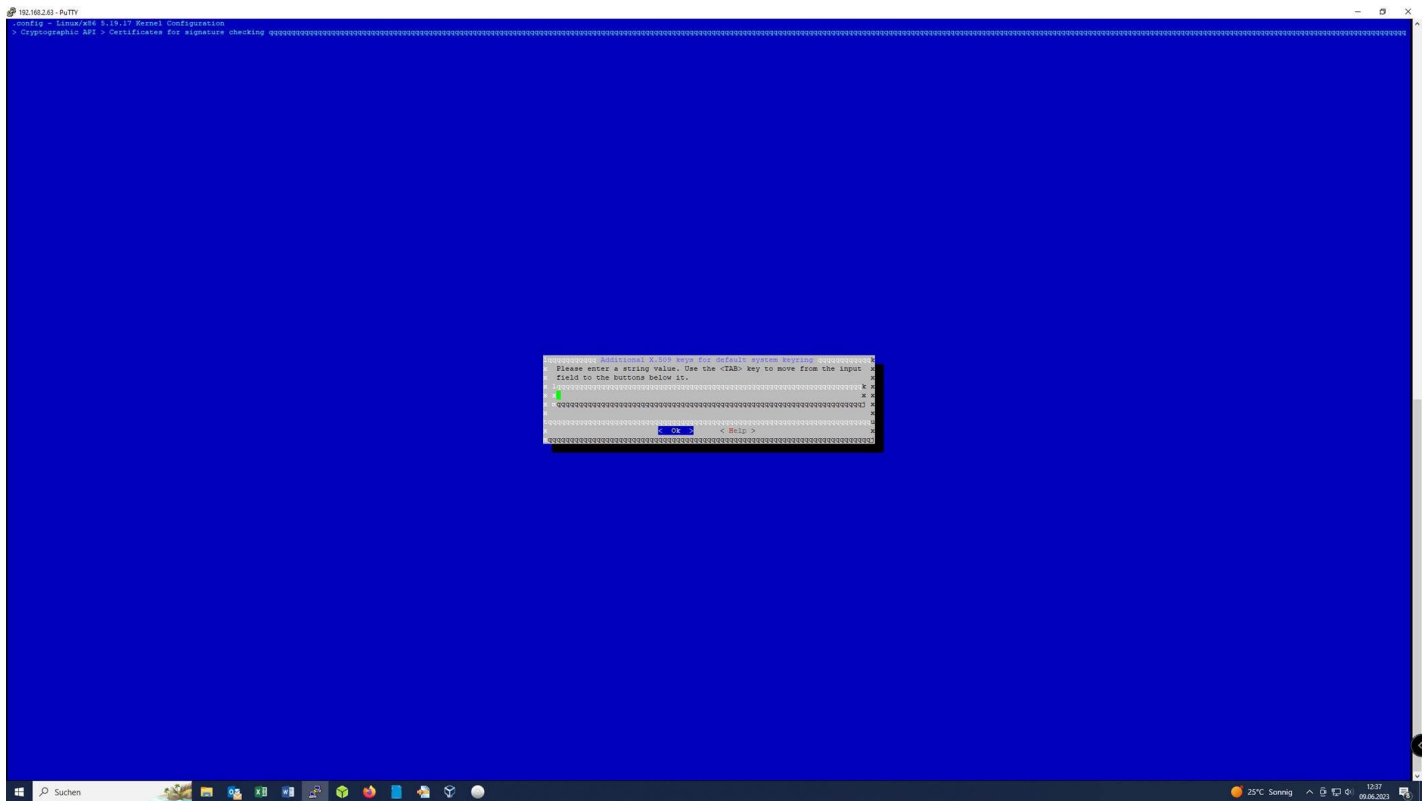
First page – look down for Kernel hacking – ENTER

Finally, very important or your compile will later die: Ensure this is empty:



```
Kernel Configuration
Kernel hacking
  Compile-time checks and compiler options
  Generic Kernel Debugging Instruments
  Memory Debugging
  Debug shared IRQ handlers
  Debug GDB, Lockups and Hangs
  Scheduler Debugging
  Enable extra linkchecking sanity checking
  Lock Debugging (spinlocks, mutexes, etc...)
  Debug IRQ flag manipulation
  Stack backtrace support
  Warn for all uses of unseeded randomness
  Object Debugging
  Debug kernel data structures
  Debug credential management
  RCU Debugging
  Force round-robin CPU selection for unbound work items
  Enable CPU hotplug state control
  Latency measuring infrastructure
  Tracers
  Enable debugging over FireWire early on boot
  Sample kernel code
  Filter access to /dev/mem
  KFD Debugging
  Kernel Testing and Coverage
```

This in within the first page menu – go to CRYPTOGRAPHIC APL – ENTER – then scroll down to ADDITIONAL X509 KEYS FOR DEFAULT SYSTEM KEYRING – ENTER – and clear the field - ENTER



Now, please save and continue with the following steps:
Do not think you should rename the file like I did

Once the configuration is done and saved:

make

make modules_install

make install

update-grub

reboot

Once rebooted, do

uname -r

It should show you your current Kernel version and now it should be 5.19.17.

*If you screw up your kernel like I did many times this will get you back to where you can begin again.
If you did not skip down to below the horizontal line*

Please google "Mr Proper Linux"

<https://unix.stackexchange.com/questions/387640/why-both-make-clean-and-make-mrproper-are-used>

Why both `make clean` and `make mrproper` are used?
unix.stackexchange.com

Use these commands

make clean
make mrproper
make distclean
will clean up the mess you created and then you should
be able to compile again (In the 5.19 folder).

Then start new from scratch...and this time just do EXACTLY what Blitz advised to do. There is absolutely no room for creativity. Did I say anything about renaming something ? No.

The above is when I had to sheepishly admit I thought I was supposed to name the config file something other than the choice given

More good advice and an explanation from Blitz

We have already a new name...it is the name of the folder...linux-5.19.17...the new config file is in there, the old config file is stored somewhere completely else and wont be lost at all. No need to backup anything.

You as well have to do things EXACTLY in the sequence I described. If you miss only one step, it will not work (like the certificates).

I missed the clearing out of the certificates field the first time

If this still does not work, delete the whole 5.19. folder and start from scratch.

In my experience the cleaning process works just fine

So we come to one of the most important step...we now give the Kernel the command to make use of what we prepared.

We do a lot of stuff here, but most importantly we isolate the cores, activate tickless mode and use the AMD energy driver instead of the generic.

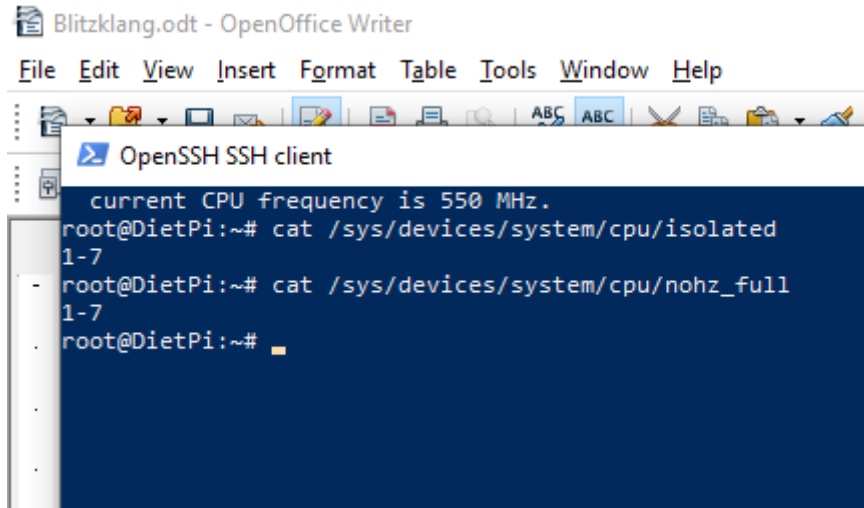
We need to change one line in your boot loader and you have to be very careful doing that or you wont have a system anymore and can start from scratch. So, PLEASE...be careful:

First let's check your current isolation status and tickless status:

cat /sys/devices/system/cpu/isolated

```
cat /sys/devices/system/cpu/nohz_full
```

It will be probably return nothing, while when done it looks like:



```
Blitzklang.odt - OpenOffice Writer
File Edit View Insert Format Table Tools Window Help
OpenSSH SSH client
current CPU frequency is 550 MHz.
root@DietPi:~# cat /sys/devices/system/cpu/isolated
1-7
root@DietPi:~# cat /sys/devices/system/cpu/nohz_full
1-7
root@DietPi:~#
```

So...core 1-7 are on my machine isolated and work in full tickless mode.

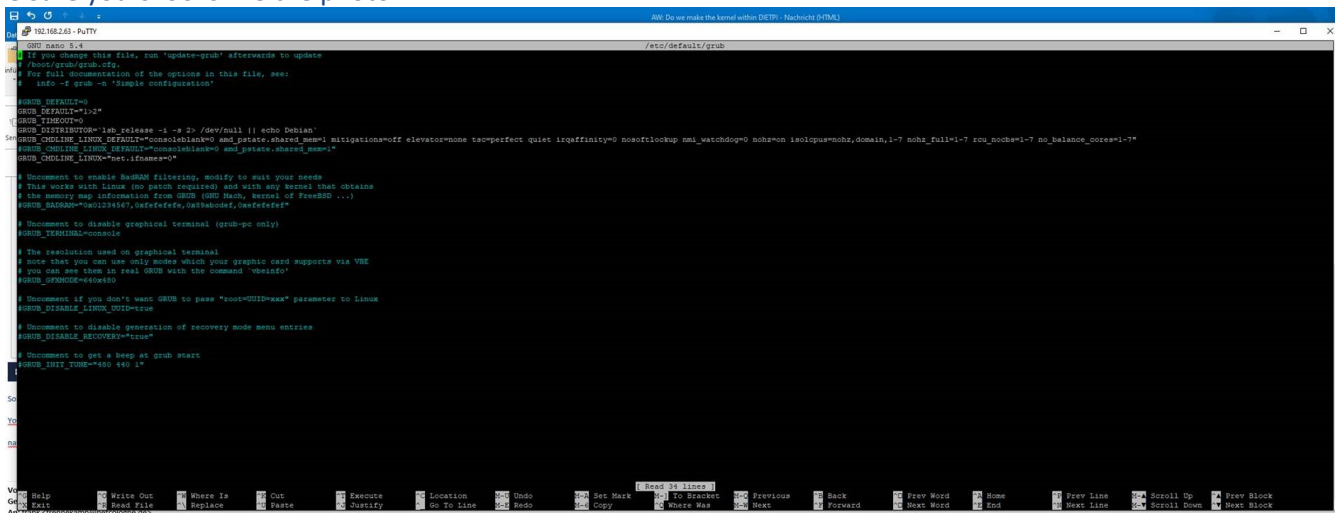
You get that now activated with

```
nano /etc/default/grub
```

Yours may look differently, that is fine... change **ONLY THE GRUB_CMDLINE_LINUX_DEFAULT LINE with the below – copy the below text and replace what is on this line with this completely**

```
GRUB_CMDLINE_LINUX_DEFAULT="consoleblank=0 amd_pstate.shared_mem=1
mitigations=off elevator=none tsc=perfect quiet irqaffinity=0 nosoftlockup
nmi_watchdog=0 nohz=on isolcpus=nohz,domain,1-7 nohz_full=1-7 rcu_nocbs=1-7
no_balance_cores=1-7"
```

Make sure yours looks like the photo.



```
AW: Do we make the kernel with 512M -> 1M (1M)
GNU nano 5.4 /etc/default/grub
# If you change this file, run 'update-grub' afterwards to update
# /boot/grub/grub.cfg.
# For full documentation of the options in this file, see:
# info -f grub -n 'Simple configuration'

GRUB_DEFAULT=saved
GRUB_TIMEOUT=10
GRUB_TIMEOUT_LIMIT=60
GRUB_CMDLINE_LINUX_DEFAULT="consoleblank=0 amd_pstate.shared_mem=1 mitigations=off elevator=none tsc=perfect quiet irqaffinity=0 nosoftlockup nmi_watchdog=0 nohz=on isolcpus=nohz,domain,1-7 nohz_full=1-7 rcu_nocbs=1-7 no_balance_cores=1-7"
GRUB_CMDLINE_LINUX=""
GRUB_TERMINAL=console

# The resolution used on graphical terminal
# Note that you can now only adjust when your graphical card supports via VBE
# you can see them in real grub with the command 'vbeinfo'
GRUB_GFXMODE=auto

# Uncomment to get a beep at grub start
GRUB_INIT_TUNE="480 440 1"
```

Save the conf file.

Now we will ask DIETPI to update the file by typing

update grub ENTER

and then

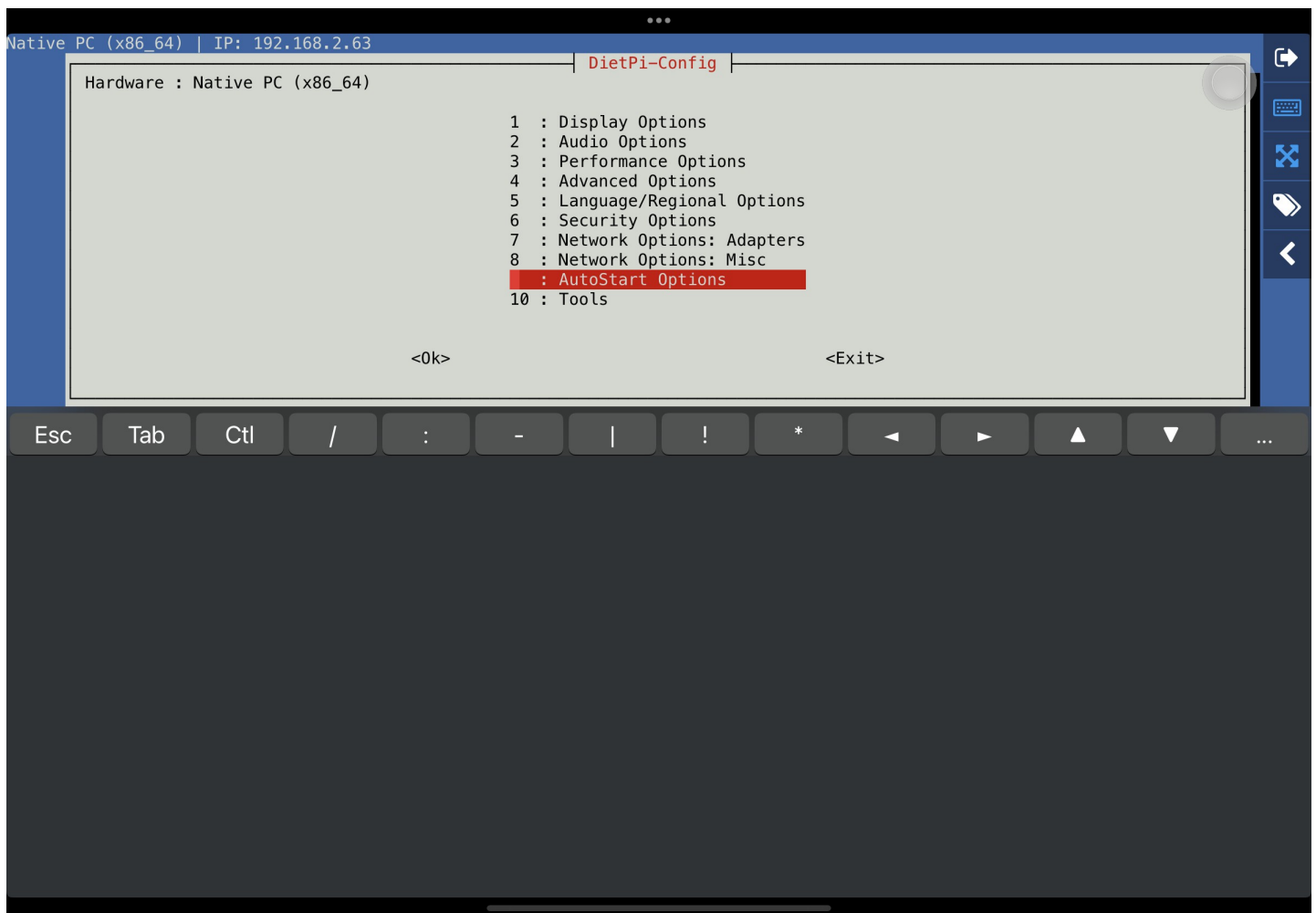
reboot ENTER

Check your isolation status and tickless status and run cpufreq-info as before.

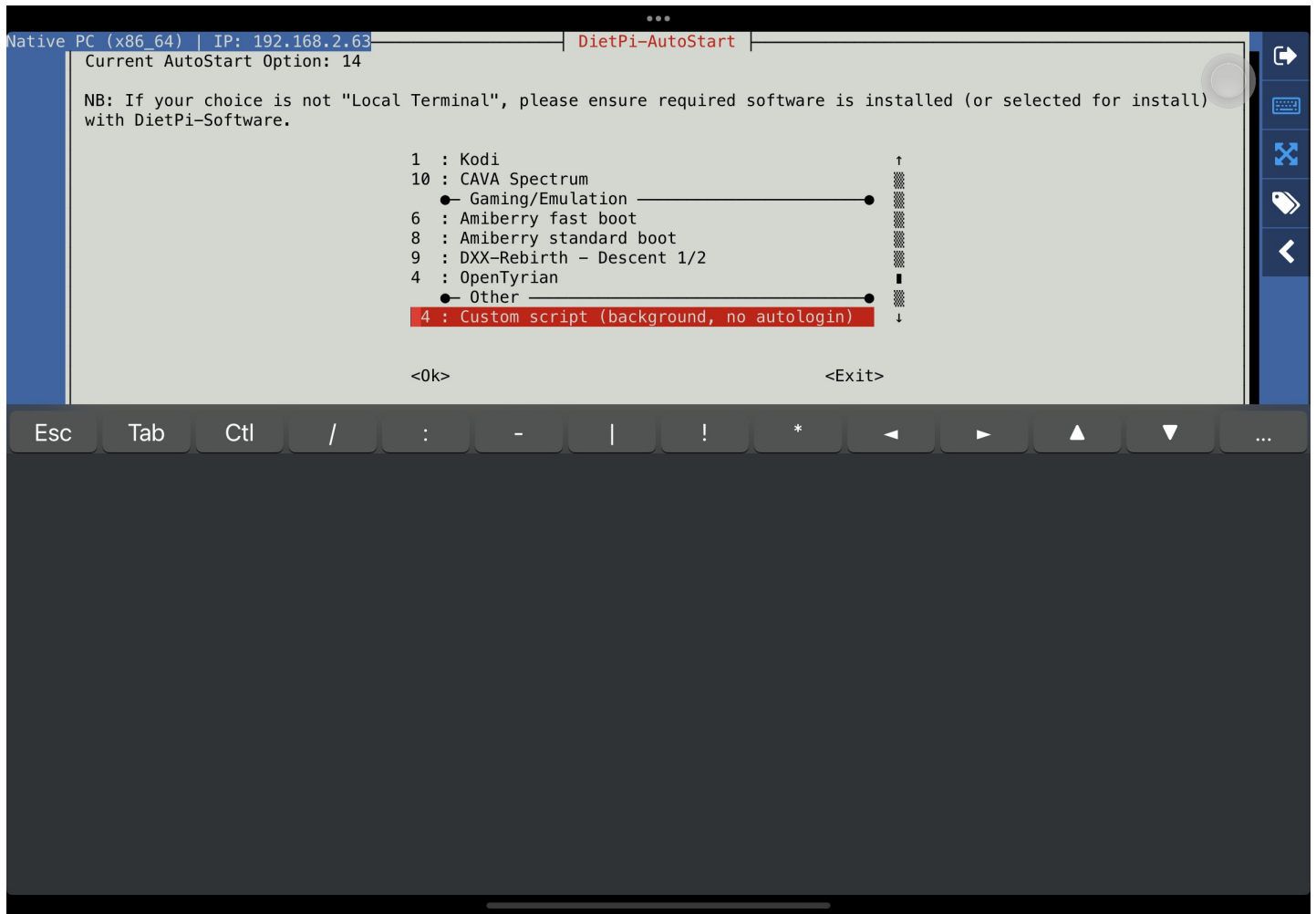
It should show you we are now in business.

Ok, lets set the frequency...

you go into dietpi-config:



And in autostart options you select 14 – you will see that is a 14 when you get there



you insert/add for the moment *only* the line with

cpupower -c all frequency-set -f 550Mhz

```
GNU nano 5.4 /var/lib/dietpi/dietpi-autostart/custom.sh
echo 0 > /sys/devices/system/machinecheck/machinecheck5/check_interval
echo 0 > /sys/devices/system/machinecheck/machinecheck6/check_interval
echo 0 > /proc/sys/kernel/nmi_watchdog
chrt -f -p 22 $(pgrep ksoftirqd/7)
chrt -f -p 22 $(pgrep ksoftirqd/6)
chrt -f -p 22 $(pgrep ksoftirqd/5)
chrt -f -p 22 $(pgrep ksoftirqd/4)
chrt -f -p 22 $(pgrep ksoftirqd/3)
chrt -f -p 22 $(pgrep ksoftirqd/2)
chrt -f -p 22 $(pgrep ksoftirqd/1)
cpupower -c all frequency-set -f 550Mhz

exit 0
```

Help Write Out Where Is Cut Execute Location Undo Set Mark

Esc Tab Ctl / : - | ! * < > ↑ ↓ ...

Save and reboot...

please run cpufreq-info again *if it is not as shown make sure you got it right*

So lets go into the final round.

Lets remember:

We want isolated, clean CPU cores, so our audio processing is not polluted. Therefore, the CPU cores are distributed:

Core 0 – Housekeeping Core for all OS-related tasks and NON-Audio stuff

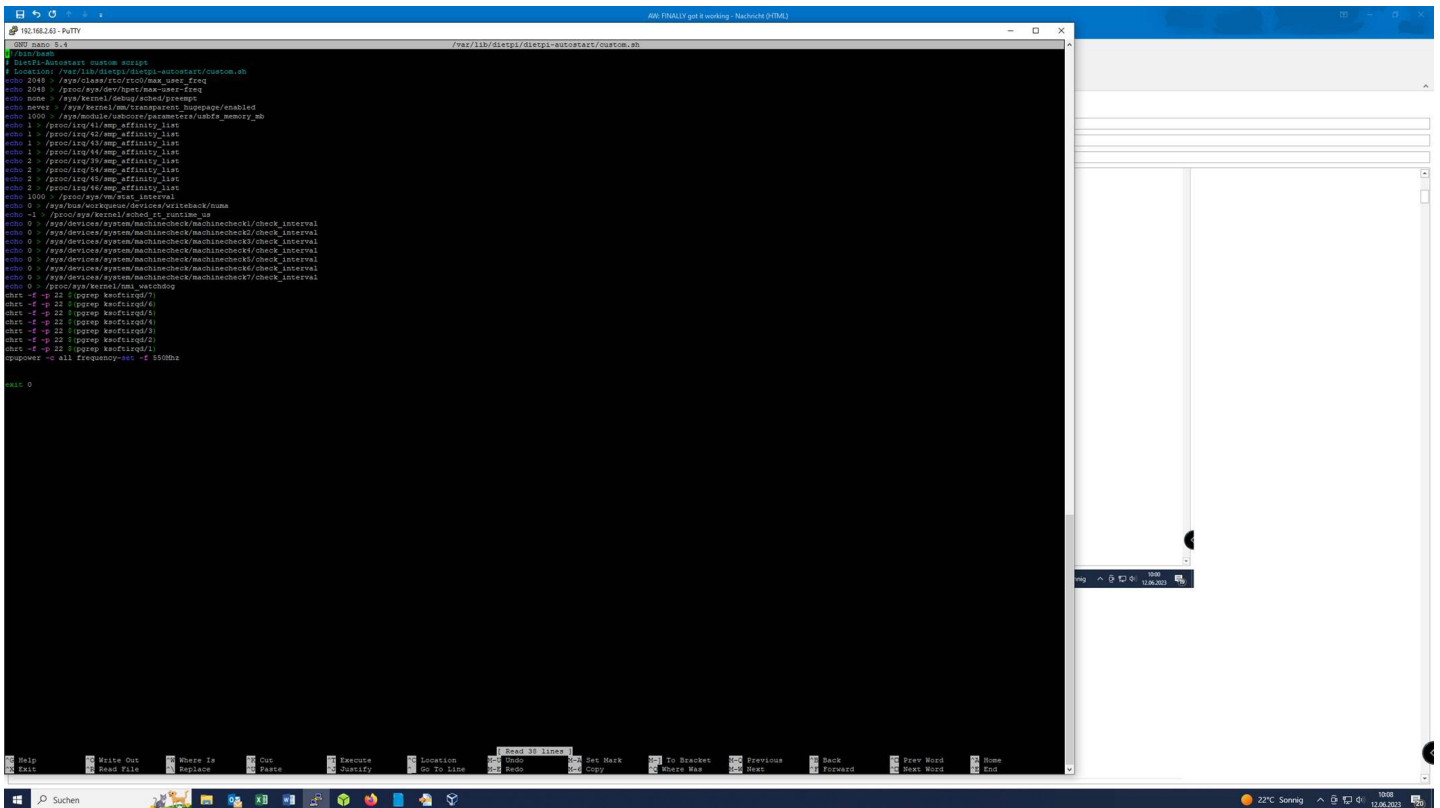
Core 1 – Reserved for LAN or SATA

Core 2 – Reserved for USB-Audio-Output

Core3-7 Reserved for MPD and its child processes

So, if you remember my HTOP screenshot in the beginning. Yours should look like that now. Please check and send me a screenshot.

It should look like this:



So, you see that I send interrupt 39 (and some other USB-Interrupts) to core 2 and now it runs on core 2.

Your numbers maybe different !!! And if you use a different USB-Port, the interrupt number may change, that is why I have spec'd more than only one interrupt for core 2...different USB-ports.

The other statements which have no irq in it are optimization statements for further audio improvements which came from different other Audio-PC-Projects. You can copy them into your autostartfile.

To make it more convenient for you, here is the ascii text of my file:

```
#!/bin/bash
# DietPi-Autostart custom script
# Location: /var/lib/dietpi/dietpi-autostart/custom.sh
echo 2048 > /sys/class/rtc/rtc0/max_user_freq
echo 2048 > /proc/sys/dev/hpet/max-user-freq
echo none > /sys/kernel/debug/sched/preempt
echo never > /sys/kernel/mm/transparent_hugepage/enabled
echo 1000 > /sys/module/usbcore/parameters/usbfs_memory_mb
echo 1 > /proc/irq/41/smp_affinity_list
echo 1 > /proc/irq/42/smp_affinity_list
echo 1 > /proc/irq/43/smp_affinity_list
echo 1 > /proc/irq/44/smp_affinity_list
echo 2 > /proc/irq/39/smp_affinity_list
echo 2 > /proc/irq/54/smp_affinity_list
echo 2 > /proc/irq/45/smp_affinity_list
echo 2 > /proc/irq/46/smp_affinity_list
echo 1000 > /proc/sys/vm/stat_interval
echo 0 > /sys/bus/workqueue/devices/writeback/numa
```

```
echo -1 > /proc/sys/kernel/sched_rt_runtime_us
echo 0 > /sys/devices/system/machinecheck/machinecheck1/check_interval
echo 0 > /sys/devices/system/machinecheck/machinecheck2/check_interval
echo 0 > /sys/devices/system/machinecheck/machinecheck3/check_interval
echo 0 > /sys/devices/system/machinecheck/machinecheck4/check_interval
echo 0 > /sys/devices/system/machinecheck/machinecheck5/check_interval
echo 0 > /sys/devices/system/machinecheck/machinecheck6/check_interval
echo 0 > /sys/devices/system/machinecheck/machinecheck7/check_interval
echo 0 > /proc/sys/kernel/nmi_watchdog
chrt -f -p 22 $(pgrep ksoftirqd/7)
chrt -f -p 22 $(pgrep ksoftirqd/6)
chrt -f -p 22 $(pgrep ksoftirqd/5)
chrt -f -p 22 $(pgrep ksoftirqd/4)
chrt -f -p 22 $(pgrep ksoftirqd/3)
chrt -f -p 22 $(pgrep ksoftirqd/2)
chrt -f -p 22 $(pgrep ksoftirqd/1)
cpupower -c all frequency-set -f 550Mhz
```

exit and save

Have fun..I think we are done...

let's check if everything works as expected...please send me the screenshots specified above (as well from HTOP).
...and let me know what you hear...

Below is what I am using in my installation – slightly different – each system will likely require slightly different instructions

```
#!/bin/bash
# DietPi-Autostart custom script
# Location: /var/lib/dietpi/dietpi-autostart/custom.sh
echo 2048 > /sys/class/rtc/rtc0/max_user_freq
echo 2048 > /proc/sys/dev/hpet/max-user-freq
echo none > /sys/kernel/debug/sched/preempt
echo never > /sys/kernel/mm/transparent_hugepage/enabled
echo 1000 > /sys/module/usbcore/parameters/usbfs_memory_mb
echo 1 > /proc/irq/41/smp_affinity_list
echo 1 > /proc/irq/42/smp_affinity_list
echo 1 > /proc/irq/43/smp_affinity_list
echo 1 > /proc/irq/44/smp_affinity_list
echo 2 > /proc/irq/40/smp_affinity_list
echo 2 > /proc/irq/54/smp_affinity_list
echo 2 > /proc/irq/45/smp_affinity_list
echo 2 > /proc/irq/46/smp_affinity_list
echo 1000 > /proc/sys/vm/stat_interval
echo 0 > /sys/bus/workqueue/devices/writeback/numa
echo -1 > /proc/sys/kernel/sched_rt_runtime_us
echo 0 > /sys/devices/system/machinecheck/machinecheck1/check_interval
echo 0 > /sys/devices/system/machinecheck/machinecheck2/check_interval
echo 0 > /sys/devices/system/machinecheck/machinecheck3/check_interval
echo 0 > /sys/devices/system/machinecheck/machinecheck4/check_interval
echo 0 > /sys/devices/system/machinecheck/machinecheck5/check_interval
```

```
echo 0 > /sys/devices/system/machinecheck/machinecheck6/check_interval
echo 0 > /sys/devices/system/machinecheck/machinecheck7/check_interval
echo 0 > /proc/sys/kernel/nmi_watchdog
chrt -f -p 22 $(pgrep ksoftirqd/7)
chrt -f -p 22 $(pgrep ksoftirqd/6)
chrt -f -p 22 $(pgrep ksoftirqd/5)
chrt -f -p 22 $(pgrep ksoftirqd/4)
chrt -f -p 22 $(pgrep ksoftirqd/3)
chrt -f -p 22 $(pgrep ksoftirqd/2)
chrt -f -p 22 $(pgrep ksoftirqd/1)
cpupower -c all frequency-set -f 550Mhz
```

exit and save

You should now experience a very different sound. One can listen to this all the while during the conversion. That can be interesting to hear the evolution.