Internship Daily Journal

65011277 Chanasorn Howattanakulphong

B.Eng. in Software Engineering

School of Engineering, KMITL

Internship company: Frontware International

Internship period: 22 April 2024 - 21 June 2024

Mon 22 April 2024

- P'Alex explained the concept of the ongoing project that I'll be working on and the assigned work. The rest of the time is spent on setting up the environment and reading related documentations that I'll be using within the project.

Tue 23 April 2024

- My part is to retrieve specs and performance data from AMD's gpu. There's an open source library available from AMD called ADLX
- https://gpuopen.com/adlx/

Wed 24 April 2024

- Wrote functions to retrieve the driver version of the gpu by comparing the guid of each device(get from window's cfgmgr32.h) to the guid returned from ADLX library

---

Thu 25 April 2024

- configured zig for cross compiling in linux (compiling windows executable and linux via makefile)
- Was compiling in visual studio before, the company lets me use a linux computer and use any desk to connect to a shared window computer.(for compiling the code, since the code should be run on both platform)
- attempting to use load library (to dynamically load the library at runtime) for windows version issue

---

Fri 26 April 2024

- Got Loadlibrary to work
- Start making a command line program
- I didn't know there was already a library for that called <getopt.h>
  I spent a lot of time implementing the command line system.
- Show specs and performance for multiple gpus

Mon 29 April 2024

- Finish up last week's work by displaying the specs and performance as a JSON.
- Learn how to use Linux commands in terminal

Tue 30 April 2024

- I Had to create the same program but on linux, found one library able to get some of the data. Have to look for more libraries to add.
- Read documents / Search for linux / C compatible resources for the development

---

Wed 1 May 2024

- Holiday

Thu 2 May 2024

- Attempt to separate parts of the RadeonTop library that is needed for the project.

Fri 3 May 2024

- Change the output from RadeonTop to JSON
- Add parsed PCI id to json output
- Try to understand RadeonTop library more, so I can cut more unnecessary parts out.

Mon 6 May 2024

- Added bus id to the json spec output
- Made data extracted with radeon top compatible for multiple gpu devices.
- Read and tried several libraries, but still haven't found the right one to use.

Tue 7 May 2024

- extracting hwmon (hardware monitor) folder to get performance information for linux.
- fix small things on window version

- read a lot of documents on hwmon and related stuffs (most of them aren't compatible with either C or linux)

Wed 8 May 2024

- browse for pciid list since the one given by radeontop library is outdated and our company's gpu in unknown to the list. There are too many available pciid on the list so I can't convert them all to C. found a C file that contain these pciid as array.
- https://fossies.org/linux/wireshark/epan/pci-ids.c
- Will try to adapt current code to use it.(todo)
- Attempting to read multiple drm devices.
- Also making it work on the linux pc downstairs. Somehow it doesn't work on that device.

Thu 9 May 2024

- read radeontop header files again to find out why max core clock speed is always 0.
- read AMD SMI library to find out if it's useful to the project.
- radeontop modules can now read from multiple drm devices.
- writing a function to locate the hwmon folder (in progress)
- learn about wrapper to use with AMD SMI

Fri 10 May 2024

- Clean up dead code
- Learn deeper on how to write a makefile
- Got access to the main project's git
- Continue looping the hwmon directory

Mon 13 May 2024

- Found ways to get all data except driver version
- Done iterating directories to search for hwmon directory
- Integrating into main code
- Found pci id parts and construct it to identify each gpu from each other so when syncing data from radeon top and system files, the data will get assign to the correct gpu
- Head exploding from searching too many sources. It's a relief to find solutions that I was looking for though. Just implementing left.

Tue 14 May 2024

- Introduce 3 new struct types and 1 union type to keep track of the data before printing.
    - struct pciId : contains data needed to construct pci id for each gpu, fetched from pci_device instance from pciaccess.h libpciaccess library. Will have to get some from the system gpu directory later. (The one I tried to loop to get the path on yesterday's report : /sys/class/pci_bus/0000:03/device/0000:03:00.0/)
    - union RetData : contains either GPUSpecData or GPUPerfData. Determined with the third DataType value, which is set according to the command line inputs (--spec or --perf)
    - GPUSpecData and GPUPerfData contain Specs or Performance data. GPUSpecData contains an instance of pciId.

- Trying to print everything in linux version as json

Wed 15 May 2024

- I took the day off to renew my driving license.
- Already requested a leave from the company since sunday and they acknowledged it.

Thu 16 May 2024

- P'Alex explains a more in-depth version of what Fluxcore and Fluxedge is. Since I have to make a presentation for the Internship Progress Meeting, next monday.
- Finish up the performance part.
- Working on specs part

Fri 17 May 2024

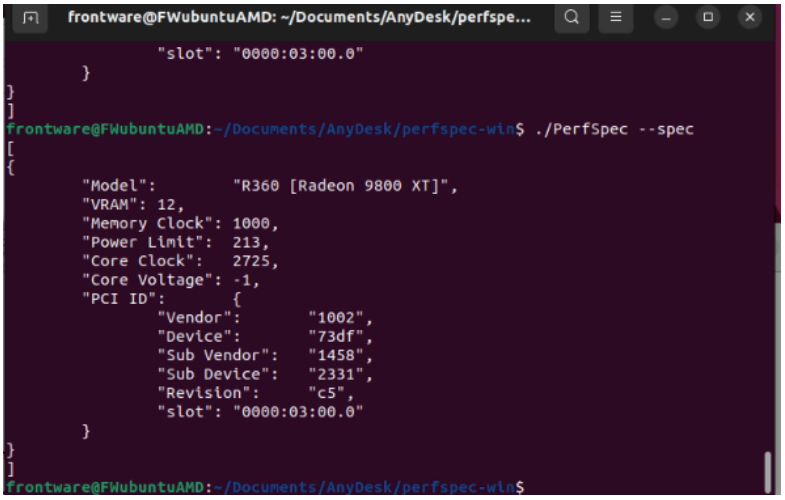
- Tried to get driver version similar to glxinfo command
- learned that there's AMD's driver and mesa driver
- gave up and will work on gpu model first.
- Using the vendor id, device id, subvendor and subdevice id from the pciid extracted last time to compare with a list of all pciids available. (I only took the AMD section since non AMD devices should all be filtered out by now).

- Spend a lot of time on an error involving enum value changing itself when I allocate memory on a totally unrelated variable.
- Decorating the model name since somehow they are not formatted the same way
- Eg.
- {0x1002, 0x4E6A, 0x1002, 0x4E6A, "R360 [Radeon 9800 XT] (Secondary)(0x1002-0x4E6A)"}
- {0x1002, 0x9851, 0x1179, 0xF928, "Beema [Radeon R5 Graphics](0x1179-0xF928)"},

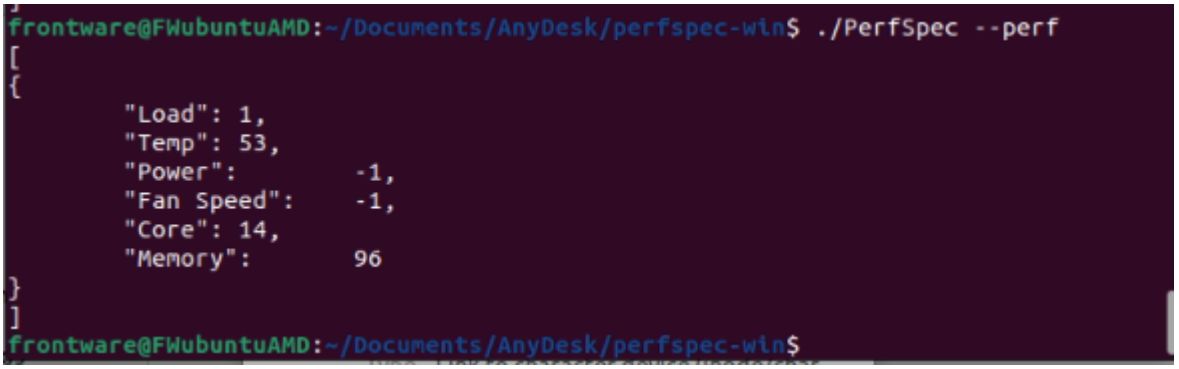
```
chanasorn@ubuntucho:~/Documents/perfspec-win/includes$ cat defineWin.h  | grep "
] (" | grep -v "0xFFFF"

{0x1002, 0x4E65, 0x1681, 0x0003, "Hercules 3D Prophet 9500 PRO [Radeon 9500 Pro]
 (Secondary)(0x1681-0x0003)"},
{0x1002, 0x4E6A, 0x1002, 0x4E6A, "R360 [Radeon 9800 XT] (Secondary)(0x1002-0x4E6
A)"},
chanasorn@ubuntucho:~/Documents/perfspec-win/includes$
```

```
{0x1002, 0x73DF, 0x1DA2, 0x445E, "Radeon RX 6700 XT GAMING OC 12G [Sapphire PULS
E](0x1DA2-0x445E)"},
{0x1002, 0x73EF, 0x1458, 0x2405, "Navi 23 [Radeon RX 6650 XT](0x1458-0x2405)"},
{0x1002, 0x744C, 0x1002, 0x0E3B, "RX 7900 GRE [XFX](0x1002-0x0E3B)"},
{0x1002, 0x744C, 0x1EAE, 0x7901, "RX-79XMERCB9 [SPEEDSTER MERC 310 RX 7900 XTX](
0x1EAE-0x7901)"},
{0x1002, 0x9480, 0x103C, 0x3628, "Mobility Radeon HD 4650 [dv6-1190en](0x103C-0x
3628)"},
{0x1002, 0x9552, 0x1028, 0x1103, "M92 [Mobility Radeon HD 4330](0x1028-0x1103)"}
```

- when I was trying to remove the last part in the curved brackets, I can't treat the two of them the same way since one has space in between the "] (" and the other one don't
- Also, the list doesn't support all the minor variance. Some subvendors and subdevices are colllected into one big category
- Eg. 0x73DF devices may have different subvendors and subdevices. But ones that are not listed are written using 0xFFFF as subvendors and subdevices.
- {0x1002, 0x73DF, 0xFFFF, 0xFFFF, "Navi 22 [Radeon RX 6700/6700 XT/6750 XT / 6800M/6850M XT](0x73DF)"},
- {0x1002, 0x73DF, 0x1043, 0x16C2, "Radeon RX 6800M(0x1043-0x16C2)"},
- {0x1002, 0x73DF, 0x1458, 0x2408, "Radeon RX 6750 XT GAMING OC 12G(0x1458-0x2408)"},
- {0x1002, 0x73DF, 0x1462, 0x3980, "Radeon RX 6700 XT Mech 2X 12G [MSI](0x1462-0x3980)"},
- So I had to remove that part if "/" is detected in "[Radeon RX 6700/6700 XT/6750 XT / 6800M/6850M XT]".



Linux version's spec info



Linux version's performance info

* The -1 are the monitoring that is unsupported on the testing device.
** Fan speed is supported but it occasionally disables itself. I had to use sudo nano to modify the read only file to enable it

- Only driver version left and I can move on to the next task.

Mon 20 May 2024

- Find a way to retrieve driver version
- Finished the driver version part
- Clean up some code
- Stop using radeontop because reading from hwmon myself takes less code and gets the same result
- Meet AJ.sun

Tue 21 May 2024

- Renew makefile to support all OS

- clean codes
- Use libGL to read driver version
- fix bugs (misuse of C unions, misuse on pointers, etc....)
- I arrived an hour late today because of a transportation problem (I have been calling a grab to work since my car is getting repaired for a few days. Today's grab was canceled after I waited for 30 mins. Had to find another cab.)

Wed 22 May 2024

- Fix dependency problems
- Fix makefile
- Fix output for linux version to match windows version
- Use dlopen to dynamically load the library on runtime. (Because we can't find libGL on p'Alex's computer, so maybe the customer's computer might not have it too)
- I broke a gpu library on the PC and spent 40 minutes fixing it

```
1886  sudo apt-get remove --auto-remove libgl1-mesa-glx
1887  sudo apt-get remove --auto-remove libgl1-mesa
1888  sudo apt update && sudo apt install -y libgl1-mesa-glx
1889  sudo apt install -y libgl1-mesa-glx
1890  ps -p 142082
1891  kill 142082
1892  sudo kill 142082
1893  sudo apt install -y libgl1-mesa-glx
1894  ps -p 142082
1895  sudo kill 142082
1896  make
1897  make rebuild
1898  sudo apt-get update
1899  sudo apt install -y libgl1-mesa-glx
1900  sudo killall apt apt-get
1901  ps aux | grep -i apt
1902  sudo kill -0 142082
1903  sudo kill -9 142082
1904  sudo apt install -y libgl1-mesa-glx
1905  sudo dpkg --configure -a
1906  locate libgl1
1907  locate libgl1.so
1908  locate libgl1-mesa
1909  sudo apt install -y libgl1-mesa-glx
1910  sudo apt update
1911  sudo apt install -y libgl1-mesa-glx
1912  apt-cache search libgl1-mesa-glx
1913  sudo apt install -y libgl1-mesa-glx
1914  sudo nano /etc/apt/sources.list
1915  sudo nano /etc/apt/sources.list.d/ubuntu.sources
1916  sudo apt install -y libgl1-mesa-glx
```

Thu 23 May 2024

- Spent more time in the morning to regain old gnome environment
- Turns out, p'Alex's computer has the libGL library and it's default for mesa drivers
- Found out libGL can also read model name, so I added it as another option if the old model name fails to work

- Fixed old no-brain solution I did because I don't know how to fix the error. The error is from misusage of union type. I tried to assign data into multiple union type.
- read on how to overclock the gpu

  What ADLX offers
  ADLX tuning list :
- fantuning
  > speed range
  > temperature range
  > fantuning state

- graphictuning
  > freq range
  > voltage range
  > gputuning state

- powertuning
  > power limit
  > TDC limit (?)

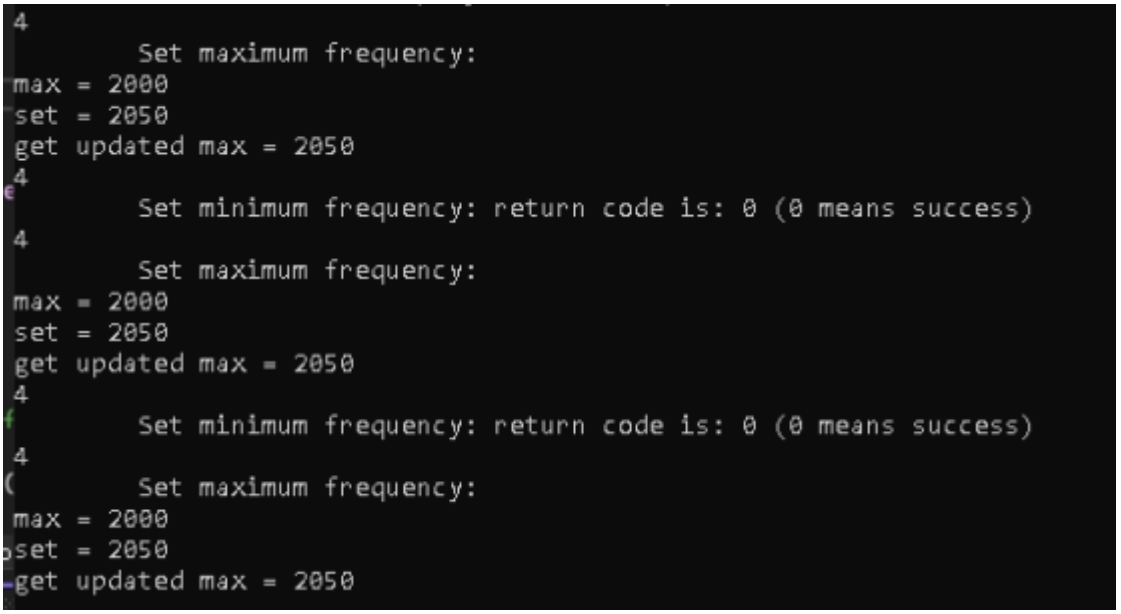
- VRAMtuning
  > memory timing description
  > max VRAM freq
  > freq (probably VRAM freq)
  > voltage
  > VRAM tuning state

- SmartAccessMemory
  > set enable/disable AMD SmartAccess Memory

Fri 24 May 2024

- Try using ADLX to change clock values on windows.
- Planning a system for the overclock program.
- Started making functions for setting core clock, core voltage and power limit. half way done
- Will paste a screenshot here on monday because I already turned off the PC downstairs.
-
```
        selected option not implemented
PS C:\Users\frontware\Documents> .\win-overclock-amd64.exe --cc 2700 --pl 0 --cv 890 --cs 3
        Set GPU max frequency succeeded
        Current GPU max frequency: 2700
        Set GPU voltage succeeded
        Current GPU clock voltage: 890
        Set power limit succeeded
        Set current power limit to: 0
        selected option not implemented
PS C:\Users\frontware\Documents>
```
-

Mon 27 May 2024

- Added fan tuning to the program
- Added gpu indexing for users
- I tried running some of ADLX's code on VRAM tuning and now the gpu stops supporting gpu tuning
- Restarting the computer
- It's back to normal (It's back to supporting gpu tuning and I can continue my job)
- 

- Now setting new max VRAM frequencies won't save when I start a new program.
- Fixed it and can now change VRAM values
- check when setting some value with ADLX if the value is updated because sometimes when you insert invalid value, it still says success while nothing changes.

Tue 28 May 2024

- Fix fan speed to using the correct function
- Read and try out / look for how to overclock on linux
- Most of the libraries out there are c++, python and rust
- Write README file

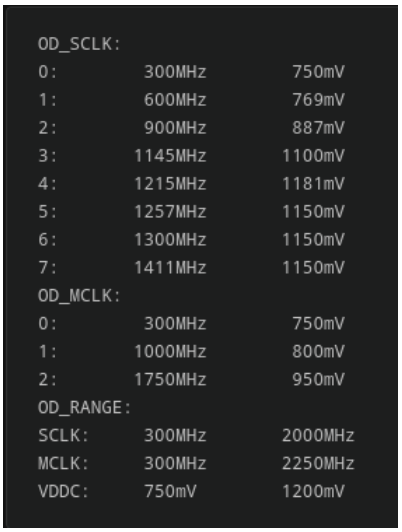
Wed 29 May 2024

- still finding ways to overclock in linux other than using command line and directly writing into the system file.
- most of the open source libraries out there uses C++, python or rust. I still take a look at their codes though, in case there's something I can rewrite in C and make use of.

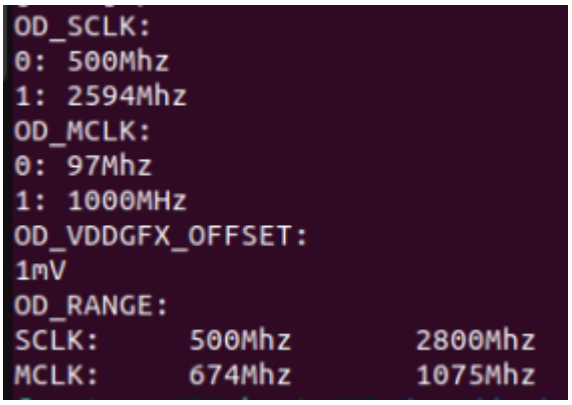
(rocm_smi_lib, amdcovc, WattmanGTK, ADL sdk, LACT, corectrl, tuxclocker, Radeon-profile)
- give up on finding libraries and back to writing it manually
- writing manually works, so I'm moving from test file to the big file
- did core clock, memory clock and power limit
- Was able to reset core clock and memory clock

Thu 30 May 2024

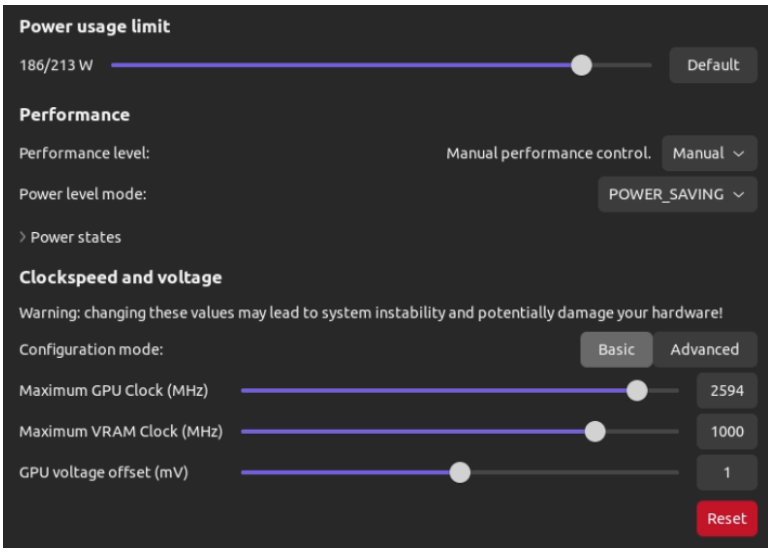
- Merged the overclock file and the main file
- read more and found out that pre navi gpus set their clocks in many states while the newer ones only allow setting min and max clock values
- Pre-navi ->

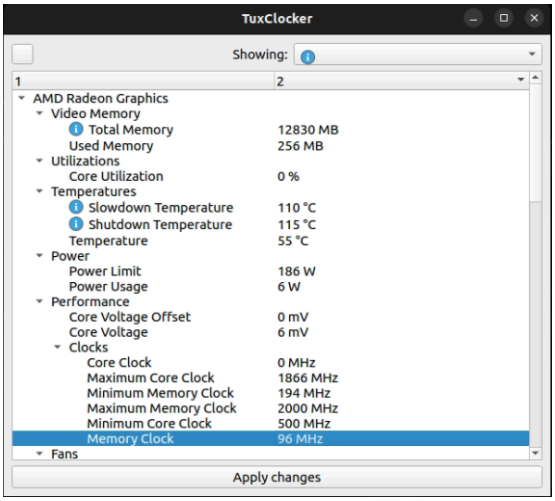


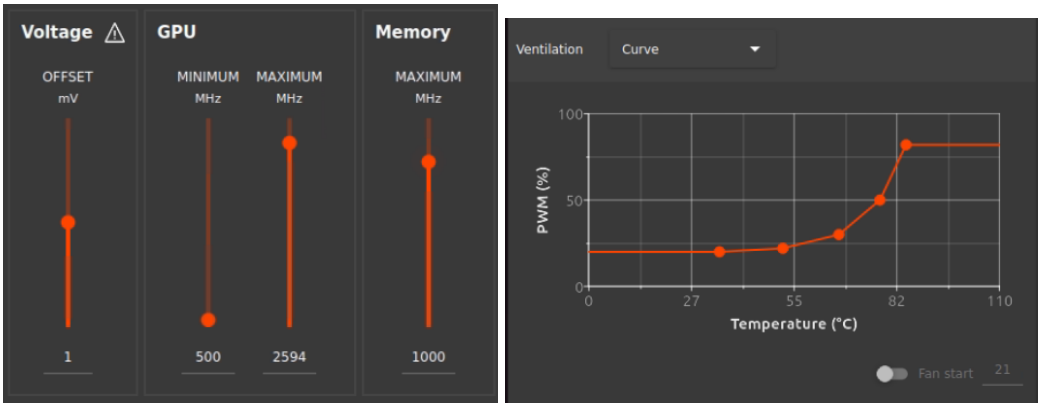
- Post-navi ->

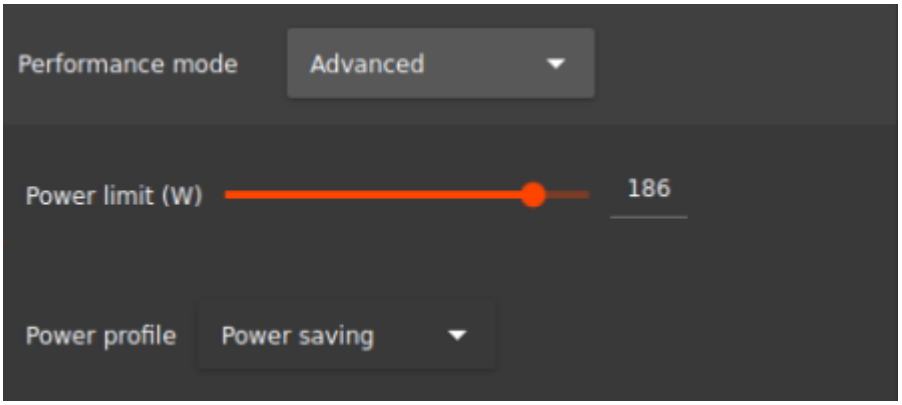


- I acidentally removed graphic libraries again. Had to install them again. I even removed graphic environments this time. The screen is pitch dark. I had to use my main pc to ssh into it and install mesa drivers, GNOME and anydesk before I can see things again. Spent almost 3 hours doing this
- There's not many things linux can do, based on kernel's documentation and archlinux's guide on overclocking. Also by these softwares I found (rocm_smi_lib, amdcovc, WattmanGTK, ADL sdk, LACT, corectrl, tuxclocker, Radeon-profile, amdgpu-clocks)

- https://wiki.archlinux.org/title/AMDGPU#Overclocking
  https://docs.kernel.org/gpu/amdgpu/thermal.html#pp-od-clk-voltage
- What each of these softwares can do
- 

- Lact : crashes when apply
- 

- Tuxclocker : value doesn't change when clicking apply changes
- 

- 

- Corectrl

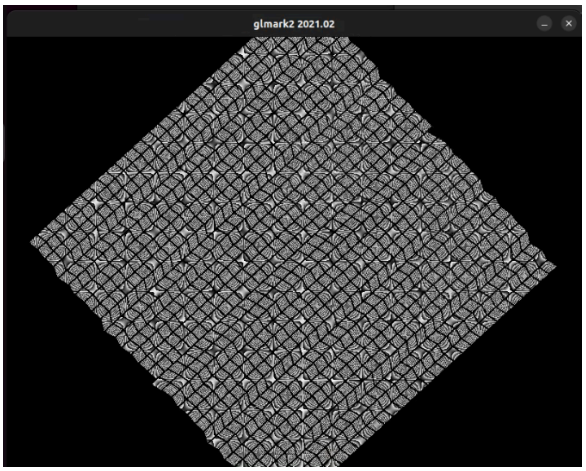
- The rest has similar values / use depreciated libraries

Fri 31 May 2024

- Figured out the fanspeed problem. Apparently, while pwm(fan's pulse width modulation) and fan speed goes linearly, these are two different things.
  NOTE: DO NOT set the fan speed via “pwm1” and “fan[1-*]_target” interfaces at the same time. That will get the former one overridden.
- If you set fan1_target, fan speed and pwm1 will change and vice versa.
- But when setting these values directly, it won't care about fan1_max, so I'll have to check with that file before assigning values
- Reset power limits
- set and reset fan speed
- yesterday I tried to free some memory in older part of the project. Just saw that it causes the program to be stuck in a while loop. Spent 10 mins finding out the problem.
- found out the command to set voltage in pp_od_clk_voltage
- (this file) ->

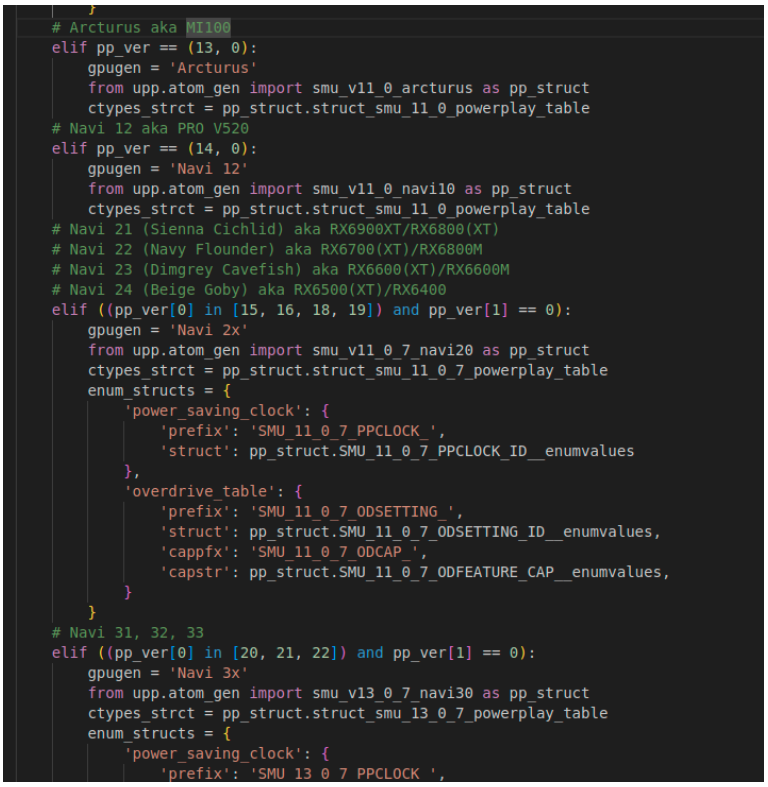
```
OD_SCLK:
0: 500Mhz
1: 2594Mhz
OD_MCLK:
0: 97Mhz
1: 1000MHz
OD_VDDGFX_OFFSET:
1mV
OD_RANGE:
SCLK:     500Mhz       2800Mhz
MCLK:     674Mhz       1075Mhz
```

Mon 3 June 2024

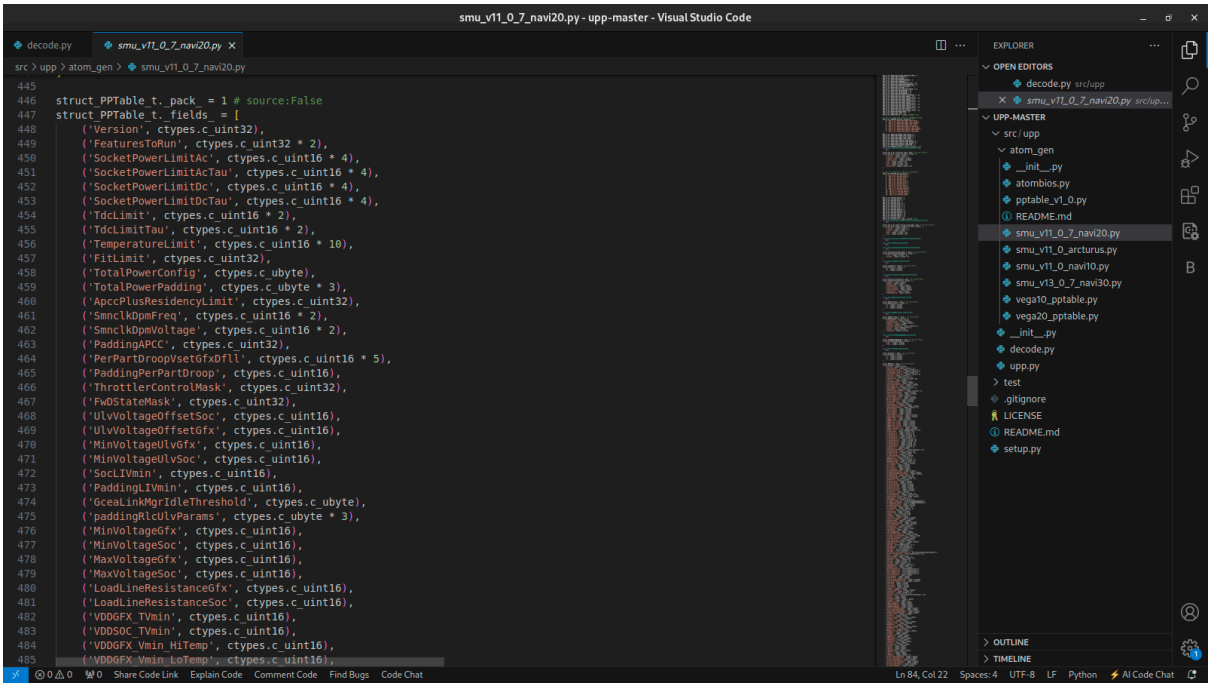
- I was looking for a way to distinguish between vega 20 or newer with vega 10 or older. Since AMD changed the structure of the config file pp_od_clk_voltage.
- The smallest program from the list is amdgpu-clocks, it uses bash script to read the file and make changes.
- Turns out amdgpu-clocks just straight out read how many lines are under each category since the newer asics (vega 20) has 2 states for sclk, while older asics (vega 10) has 8 states
- Created the function to check whether the gpu has the older or newer asics by checking the number of states sclk contains in the file pp_od_clk_voltage. The result is stored in an enum.
- set and reset core and memory state
- Current progress :
  - Core Clock, Mhz ✔️
  - Core State, mV ✔️
  - Core Voltage, mV ✔️ ❓
  - Power Limit, Watt ✔️
  - SoC VDD Max, mv
  - Fan Speed, % ✔️ don't know how to get the default value yet
  - Memory Clock, Mhz ✔️
  - Memory Controller Voltage, mV
  - Memory State ✔️
  - Memory Voltage, mV
  - SoC Frequency, Mhz
- I don't really know if the rest are available for overclocking or not.
- From what I searched, I will need to edit the pp_table (power play table) file in sysfs
- I watched this benchmark tester for a while
- 

- Try to read hiveo's source code but I don understand it.
- hive os uses upp(Uplift Power Play) tools to read the power play table and saves it temporarily somewhere, then make the changes in the temp file onto the real file. upp is in python and I don't really understand how it works.
- UPP: Uplift Power Play
  A tool for parsing, dumping and modifying data in Radeon
  PowerPlay tables

Tue 4 June 2024

- read Uplift Power Play tools.
- it's written in python, has a class as a structure to keep the data from pp_table. They import ctypes to use C data types. It's going to be very hard to replicate their structures, and I don't think I can modify only some of the values and not create the whole structure because I don't know which bits represent what kind of data
- The structure is also different for each architecture
- 

```
    }
# Arcturus aka MI100
elif pp_ver == (13, 0):
    gpugen = 'Arcturus'
    from upp.atom_gen import smu_v11_0_arcturus as pp_struct
    ctypes_strct = pp_struct.struct_smu_11_0_powerplay_table
# Navi 12 aka PRO V520
elif pp_ver == (14, 0):
    gpugen = 'Navi 12'
    from upp.atom_gen import smu_v11_0_navi10 as pp_struct
    ctypes_strct = pp_struct.struct_smu_11_0_powerplay_table
# Navi 21 (Sienna Cichlid) aka RX6900XT/RX6800(XT)
# Navi 22 (Navy Flounder) aka RX6700(XT)/RX6800M
# Navi 23 (Dimgrey Cavefish) aka RX6600(XT)/RX6600M
# Navi 24 (Beige Goby) aka RX6500(XT)/RX6400
elif ((pp_ver[0] in [15, 16, 18, 19]) and pp_ver[1] == 0):
    gpugen = 'Navi 2x'
    from upp.atom_gen import smu_v11_0_7_navi20 as pp_struct
    ctypes_strct = pp_struct.struct_smu_11_0_7_powerplay_table
    enum_structs = {
        'power_saving_clock': {
            'prefix': 'SMU_11_0_7_PPCLOCK_',
            'struct': pp_struct.SMU_11_0_7_PPCLOCK_ID__enumvalues
        },
        'overdrive_table': {
            'prefix': 'SMU_11_0_7_ODSETTING_',
            'struct': pp_struct.SMU_11_0_7_ODSETTING_ID__enumvalues,
            'cappfx': 'SMU_11_0_7_ODCAP_',
            'capstr': pp_struct.SMU_11_0_7_ODFEATURE_CAP__enumvalues,
        }
    }
# Navi 31, 32, 33
elif ((pp_ver[0] in [20, 21, 22]) and pp_ver[1] == 0):
    gpugen = 'Navi 3x'
    from upp.atom_gen import smu_v13_0_7_navi30 as pp_struct
    ctypes_strct = pp_struct.struct_smu_13_0_7_powerplay_table
    enum_structs = {
        'power_saving_clock': {
            'prefix': 'SMU_13_0_7_PPCLOCK_',
```

- 

- refactor code (added a new global variable for card path to avoid passing gpu index on every single function)
- tried to add 2>/dev/null to all the command casted in the program to discard stderr on the terminal. But learned that it's a bad idea because I won't be able to handle errors. So I revert the changes.
- trying to overclock with upp, if it has everything I need, I will try to execute the py files in C
- try running upp to see if it is working or not. It crashes the PC :(will talk about it with p'alex tomorrow

---

Wed 5 June 2024

- I spent the day on upp
- First I try to use upp to modify pp_table. (Yesterday the PC crashes upon trying)
- I made it work and the value in pp_table is modified.
- Then I was lost because I couldn't find the byte index they used to modify the pp_table. There's no document for it either.
- "The purpose of this project is to be able to analyse, extract and modify any PP table parameter, even for future cards. Interpreting or documenting each and every possible PP table parameter is practically impossible, because even Linux kernel developers are not sure what most of them represent, and AMD doesn't really bother to provide any meaningful documentation."

```
frontware@FWubuntuAMD:~$ upp --pp-file /sys/class/drm/card0/device/pp_table set
--write \overdrive_table/max/0=2799
Changing overdrive_table.max.0 from 2800 to 2799 at 0x0e2
Committing changes to '/sys/class/drm/card0/device/pp_table'.
```

- upp tells you which byte location is being changed.
- Since pp_table has a different structure for almost every ASICS and we only have testing PC with Navi 22 ASICS, my plan is to find out the byte location for each value I need and define it in my program.
-
- After that I try to get the hex bytes from pp_table and write into it.
- Then I tried actually replacing pp_table with my modified version.
- Right now I'm integrating it into the main program.
- Should be done with this part tomorrow

Thu 6 June 2024

- I lied. There's a lot of things to be done.
- I made functions for
    - Modifying the pp_table and write it back
    - Check the pp_table version because each card has a different structured pp_table. It will be dangerous if I set the wrong byte.
    - Function to return byte index based on the model of your card
    - Function that receives decimal value (input from users) then convert it into hexadecimal. Returns two separate bytes as integer.
    - Function that receives hexadecimal value (parsed from pp_table) then converts it into decimal. Returns an integer version of that hex value used tp output for users (core voltage modified from 2700 to 2800).

Fri 7 June 2024

- parse pp_table from raw data into kernel struct ❌ unused
- check if the changes made to pp_table is really applied.

- Find out two main problems
- Sometimes when writing pp_table back, it freezes the computer. Seems like it is a firmware problem and has not been fixed yet.
  https://github.com/azeam/powerupp/issues/21
  https://gitlab.freedesktop.org/drm/amd/-/issues/2060
- How UPP get the byte offset
  - they use clang2py to convert kernel structure into python code. (smu_v11_0_7_pptable.h)
  - The structures are structs with dictionaries inside.
  - I tried to parse raw data into the blank struct (using unconverted smu_v11_0_7_pptable.h)

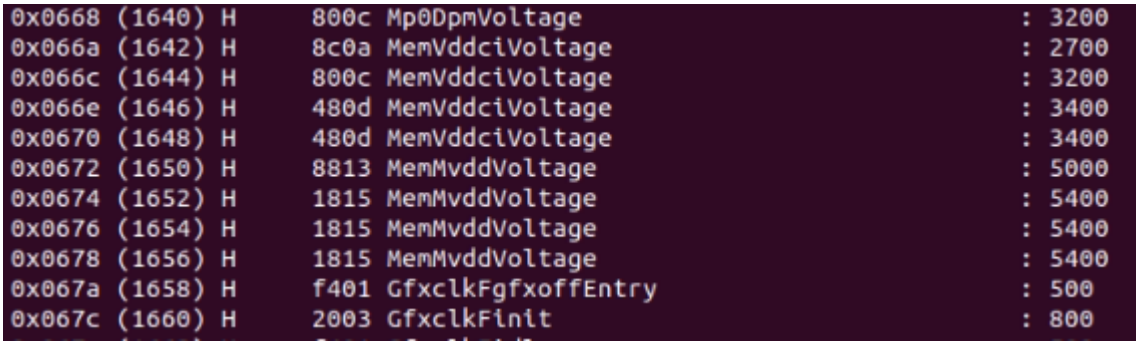
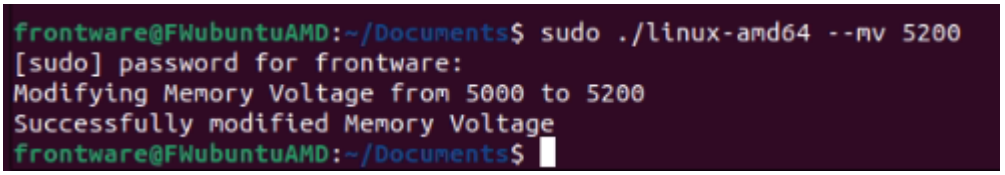
- The problem is I don't know how to identify the values.
- Will have to check how the kernel does it (or where they hardcoded it) tomorrow

I want to finish this program alreadyyyyyy. I want to hurry up and start the third project. I hate amd so much. They have half Nvidia's documentation.

Mon 10 June 2024

- Read the kernel git to find out the byte offsets to overclock.
- I found many parts that I need but I can't find how to initialize the handles. So I can't try the code out.
- Still can't run the functions
- 
- I have million of tabs running right now.
- Will try again tomorrow.

Tue 11 June 2024

- found a way to get byte offsets for pp_table
- work on a function that sets byte offsets for selected value
- currently works on navi 10 and 20 only
- other architecture has different pp_table and I was not sure and don't want to guess what the fields are.
- 
```
0x0668 (1640) H     800c Mp0DpmVoltage                          : 3200
0x066a (1642) H     8c0a MemVddciVoltage                        : 2700
0x066c (1644) H     800c MemVddciVoltage                        : 3200
0x066e (1646) H     480d MemVddciVoltage                        : 3400
0x0670 (1648) H     480d MemVddciVoltage                        : 3400
0x0672 (1650) H     8813 MemMvddVoltage                         : 5000
0x0674 (1652) H     1815 MemMvddVoltage                         : 5400
0x0676 (1654) H     1815 MemMvddVoltage                         : 5400
0x0678 (1656) H     1815 MemMvddVoltage                         : 5400
0x067a (1658) H     f401 GfxclkFgfxoffEntry                     : 500
0x067c (1660) H     2003 GfxclkFinit                            : 800
```
- 
```
frontware@FWubuntuAMD:~/Documents$ sudo ./linux-amd64 --mv 5200
[sudo] password for frontware:
Modifying Memory Voltage from 5000 to 5200
Successfully modified Memory Voltage
frontware@FWubuntuAMD:~/Documents$
```

```
0x0668 (1640) H     800c Mp0DpmVoltage                    : 3200
0x066a (1642) H     8c0a MemVddciVoltage                  : 2700
0x066c (1644) H     800c MemVddciVoltage                  : 3200
0x066e (1646) H     480d MemVddciVoltage                  : 3400
0x0670 (1648) H     480d MemVddciVoltage                  : 3400
0x0672 (1650) H     5014 MemMvddVoltage                   : 5200
0x0674 (1652) H     5014 MemMvddVoltage                   : 5200
0x0676 (1654) H     5014 MemMvddVoltage                   : 5200
0x0678 (1656) H     5014 MemMvddVoltage                   : 5200
0x067a (1658) H     f401 GfxclkFgfxoffEntry               : 500
0x067c (1660) H     2003 GfxclkFinit                      : 800
```

- Update readme files
- clean up main's include and function declarations. Moved it to main.h

Wed 12 June 2024

- Added navi 30 pp_table modification
- vega and polaris does not allow voltage modification
- fix pptable struct name problem

Imported kernel files for different gpu architecture have conflicted file names. So P'Alex suggested I should create new .c and .h files to handle those. In these files I include each version of the struct and create a function to handle them. Then call these functions from other files

```
137  struct smu_11_0_powerplay_table {
138          struct atom_common_table_header header;
139          uint8_t  table_revision;
140          uint16_t table_size;
141          uint32_t golden_pp_id;
142          uint32_t golden_revision;
143          uint16_t format_id;
144          uint32_t platform_caps;
145
146          uint8_t  thermal_controller_type;
147
148          uint16_t small_power_limit1;
149          uint16_t small_power_limit2;
150          uint16_t boost_power_limit;
151          uint16_t od_turbo_power_limit;
152          uint16_t od_power_save_power_limit;
153          uint16_t software_shutdown_temp;
154
155          uint16_t reserve[6];
156
157          struct smu_11_0_power_saving_clock_table
158          struct smu_11_0_overdrive_table
159
160  #ifndef SMU_11_0_PARTIAL_PPTABLE
161          PPTable_t smc_pptable;
162  #endif
163  };
```

-

```
519   #pragma pack(push, 1)
520 > typedef struct {…
818   } PPTable_t;
819   #pragma pack(pop)
```

- For SMU_11

```
1364  #pragma pack(push, 1)
1365  typedef struct {
1366    SkuTable_t SkuTable;
1367    BoardTable_t BoardTable;
1368  } PPTable_t;
1369  #pragma pack(pop)
```

- For SMU_13
- I was finishing the jobs, so I tried compiling everything and run them again. I don't know when did windows version broke

---

Thu 13 June 2024

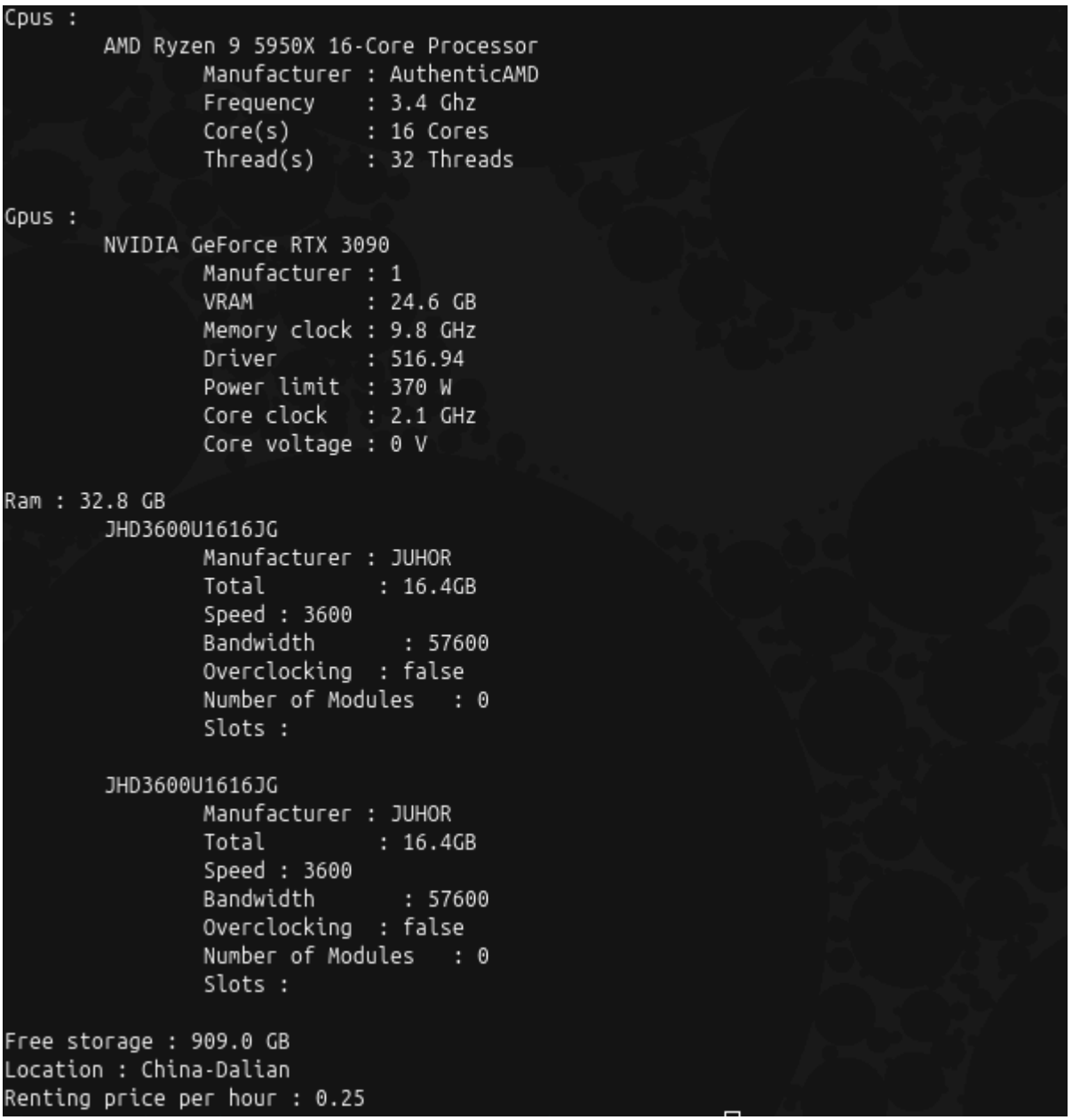
- Yesterday before going back home, amd adrenalin can't be open and my windows program wasn't working. So I tried reinstalling the adrenalin driver. There's a windows update so I just went home. I tested it this morning before coming upstairs. Everything is working like usual now.
- Refactored a lot of code
- Add more functions to get max and min for each value. This will be displayed in the help section. Which means it might not go through the normal main pipeline and might be executed at the command line argument handles. So I had to implement many things to support this.

Fri 14 June 2024

- add value ranges for help command windows and linux
- Had to do for all navi versions
- fix an old bug that I just found.
- read rust syntax from scratch. Because The next assignment I'm getting is in rust and I literally forgot everything.
- read what are tonic and grpc.

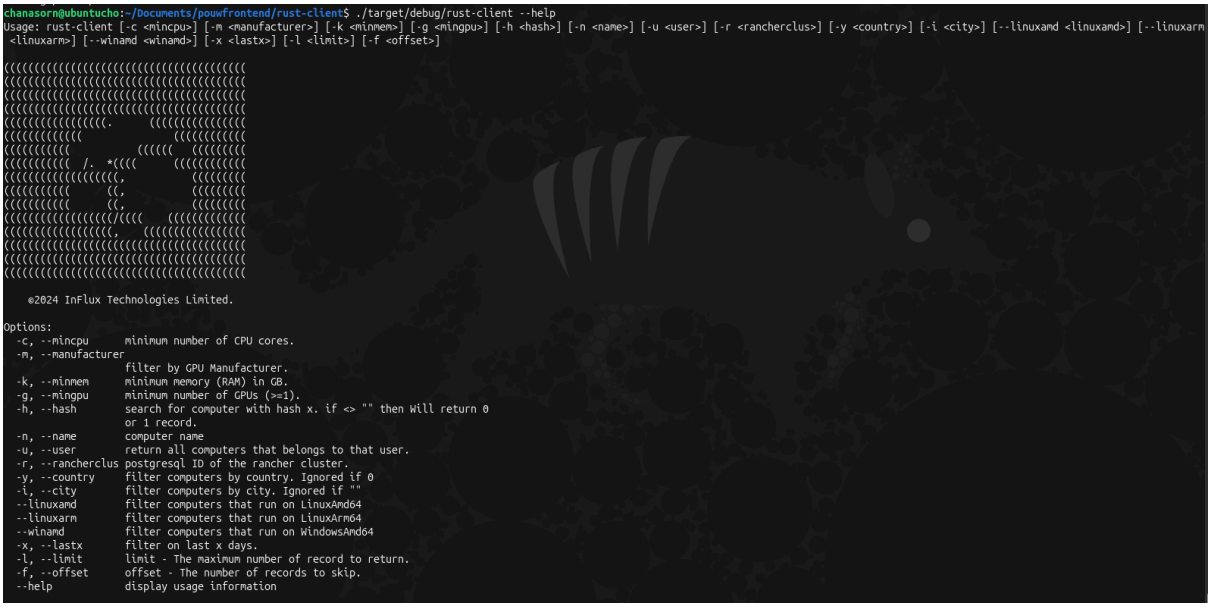
Mon 17 June 2024

- Last weeek of internship
- The new work is in rust and I already forgot all of it.
- I came early because I knew I'd spend a bit of time learning the company's code . I was so focused that I forgot to check in for work and I'm one hour late on the system.
- print out computer detail requested from the server
- fix bugs on the old program. the spec command on windows suddenly just doesn't print out anything. Test computer can't compile and build the program, while my computer and P'Alex's could. Which is pretty weird. Turns out the zig on that device is outdated. Simply updating zig solved everything.
- By the end of the day, I was a lot more familiar with Rust programming language since I learned some of them in the first year.
- I'm happy that they let me use rust a bit in the last week of my internship and look forward to learning more.



Tue 18 June 2024

- Finish up printing the details on each computer requested from the server.
- Add command line feature to let user filter out desired computers



- Start working on a new task. Using ratatui library to create ui in the terminal for this program

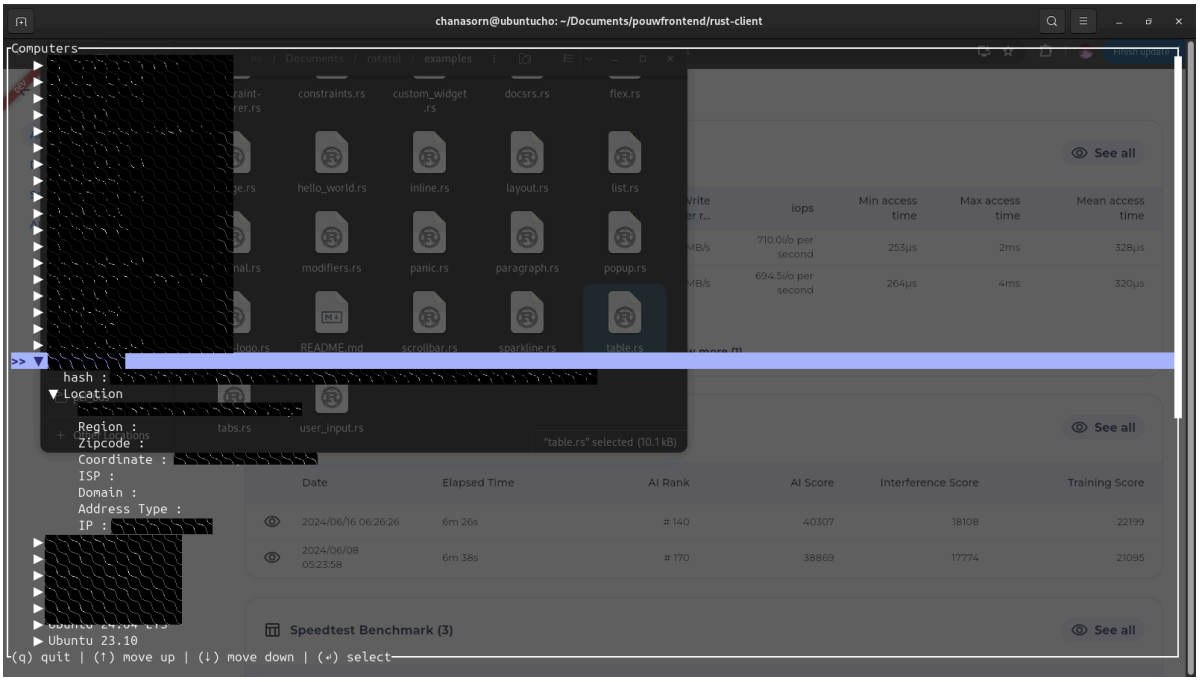
---

Wed 19 June 2024

- try to use ratatui to make ui for the previous program
- tried and deleted the whole thing several times.
- still at the same place as yesterday

Thu 20 June 2024

- Finally got the gui to work.
- I understand the concept of lifetime in rust since yesterday, but I don’t know how to use it.



done making a collapsible gui for all computer details.
- still have a bug on cpu. only display one cpu.
- and storages. Does not display any storage device.
Will add more pictures tomorrow

Fri 21 June 2024

- - Continue with the terminal program that shows detail of each computers in tree-like structures
- - Can get all details of each computers
- - Can search by hash
- - Can use --benchmark command to show benchmark of a specific computer
- - Added light theme
-
