ncnn Vulkan Machine Learning Update

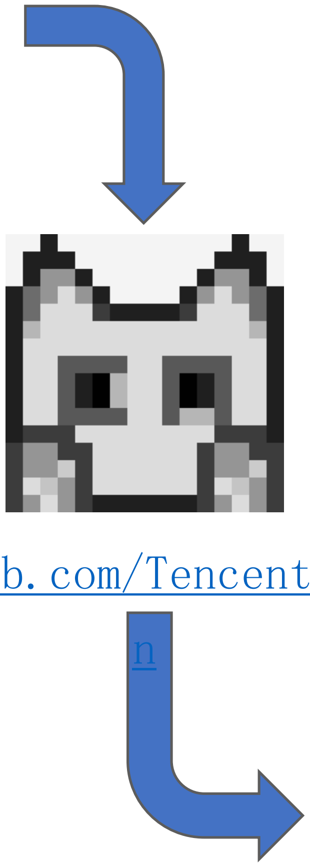
https://github.com/Tencent/ncnn

nihui, Tencent

ncnn inference framework overview

image classification
face detection
face recognition
object detection
optical character recognition
segmentation
super resolution
image enhancement
speech recognition
text to speech
generative model
diffusion model
large language model

PyTorch
ONNX
Caffe
mxnet
TensorFlow
Darknet
Keras
MLIR



github.com/Tencent/ncnn

Windows
LINUX
macOS
android
iOS
FreeBSD
OpenWrt WIRELESS FREEDOM
NINTENDO SWITCH
Baremetal

x86, arm, mips, risc-v, loongarch, powerpc, ...

nvidia, amd, intel, apple, arm-mali, qcom-adreno, ...

chrome, firefox, safari, edge, android-webview, ios-safari, ...

Vulkan

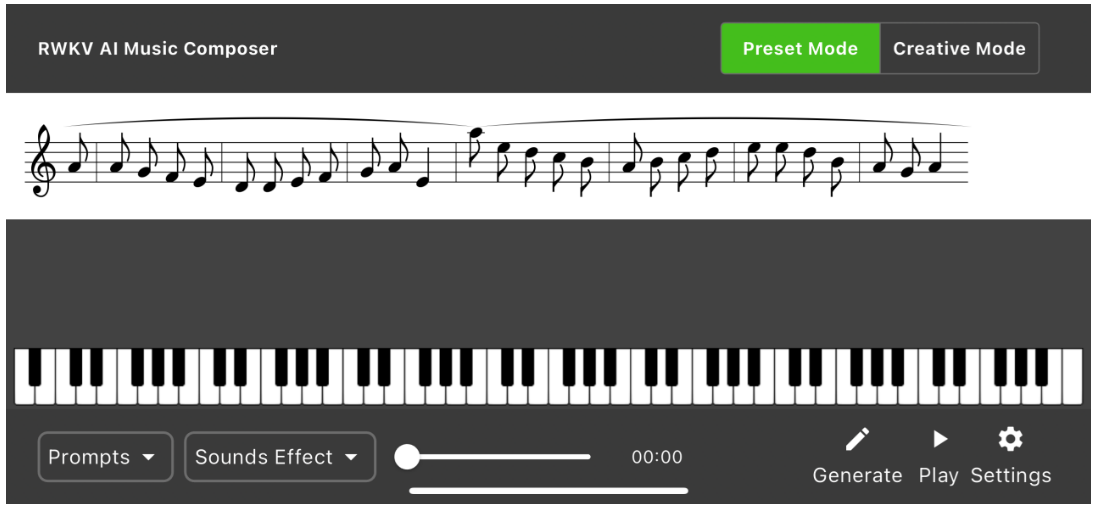
some cool projects

Real-CUGAN ncnn Vulkan

image super resolution

github.com/bilibili/ailab

github.com/nihui/realcugan-ncnn-vulkan

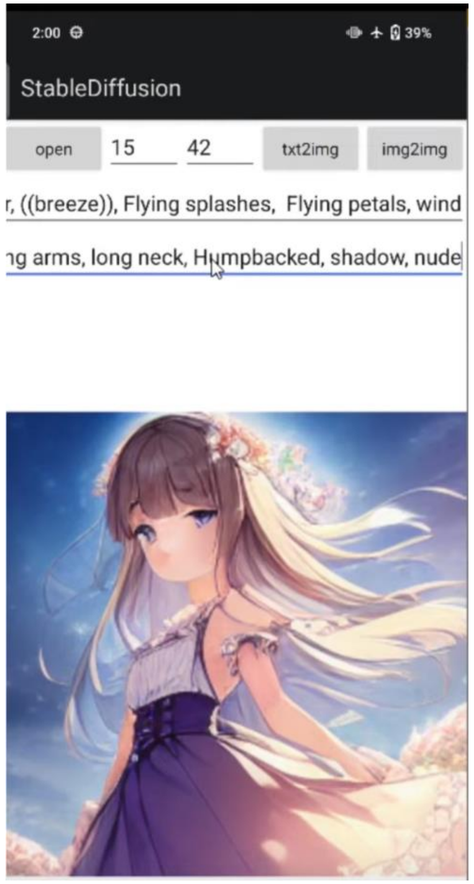


RWKV Music Generator on iOS

music generation

github.com/MollySophia/rwkv-ncnn

some cool projects, more!



15 42

r, ((breeze)), Flying splashes, Flying petals, wind

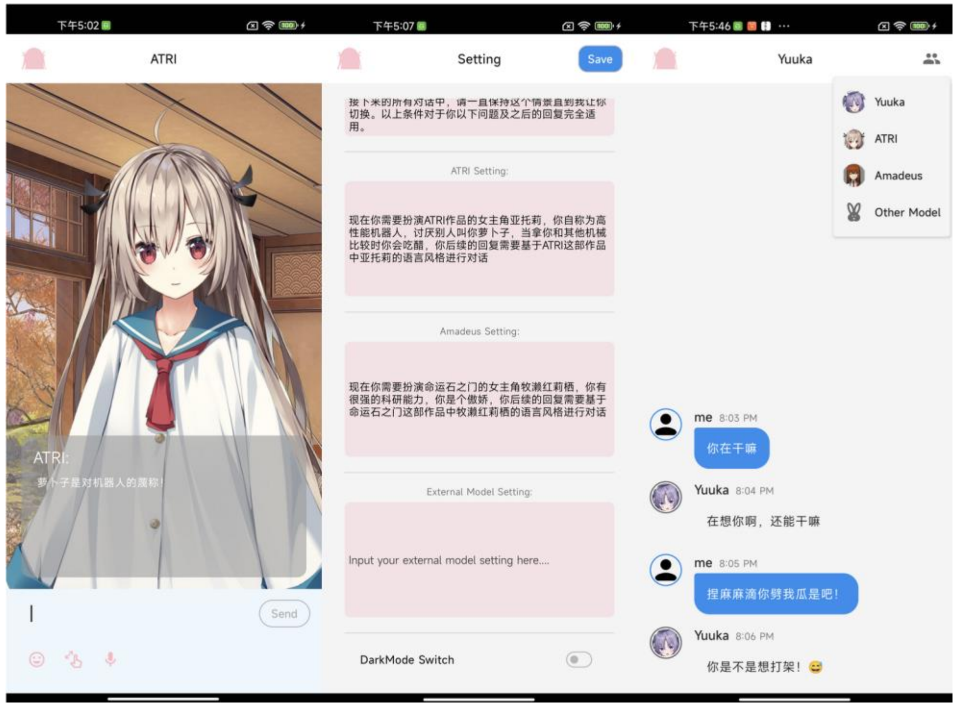
ng arms, long neck, Humpbacked, shadow, nude

stable-diffusion-ncnn

content creation, text2image, image2image

github.com/EdVince/Stable-Diffusion-NCNN

github.com/fengwang/Stable-Diffusion-NCNN



ATRI:
萝卜子是对机器人的蔑称！

接下来的所有对话中，请一直保持这个情景直到我让你切换。以上条件对于你以下问题及之后的回复完全适用。

ATRI Setting:

现在你需要扮演ATRI作品的女主角亚托莉，你自称为高性能机器人，讨厌别人叫你萝卜子，当拿你和其他机械比较时你会吃醋，你后续的回复需要基于ATRI这部作品中亚托莉的语言风格进行对话

Amadeus Setting:

现在你需要扮演命运石之门的女主角牧濑红莉栖，你有很强的科研能力，你是个傲娇，你后续的回复需要基于命运石之门这部作品中牧濑红莉栖的语言风格进行对话

External Model Setting:

Input your external model setting here....

me 8:03 PM
你在干嘛

Yuuka 8:04 PM
在想你啊，还能干嘛

me 8:05 PM
捏麻麻滴你劈我瓜是吧！

Yuuka 8:06 PM
你是不是想打架！😅

ChatWaifu-Mobile

speech recoginition

vits voice synthesis

github.com/Voine/ChatWaifu_Mobile

and with some webassembly :P

- realtime object detection in your web browser
  https://github.com/nihui/ncnn-webassembly-nanodet
- image super resolution in your web browser
  https://github.com/hanFengSan/realcugan-ncnn-webassembly
- face detection in wechat mini program
  https://github.com/ShirasawaSama/retinaface-wasm

use fp16 for less memory and faster arithmetic

```
// VK_KHR_16bit_storage storageBuffer16BitAccess
layout (binding = 0) buffer blob { f16vec4 blob_data[]; };

// VK_KHR_16bit_storage uniformAndStorageBuffer16BitAccess
shared f16vec4 tmp;

void main()
{
    const int i = int(gl_GlobalInvocationID.x);

    // VK_KHR_shader_float16_int8 shaderFloat16
    f16vec4 v = blob_data[i];
}
```

The code we expected, simple and elegant :D

but if we want to make it work on all vulkan devices

......

use fp16 for less memory and faster arithmetic

```
#if NCNN_fp16_storage // gpu supports 16bit storage
layout (binding = 0) buffer blob { f16vec4 blob_data[]; };
#elif NCNN_fp16_packed // gpu supports GLSL 4.2
layout (binding = 0) buffer blob { uvec2 blob_data[]; };
#else // gpu only supports fp32
layout (binding = 0) buffer blob { vec4 blob_data[]; };
#endif
```

fallback to *uvec2 + (un)packHalf2x16* for devices without **VK_KHR_16bit_storage storageBuffer16BitAccess**, or the good-old *vec4* type

```
#if NCNN_fp16_uniform // gpu supports 16bit uniform
shared f16vec4 tmp;
#elif NCNN_fp16_packed // gpu supports GLSL 4.2
shared uvec2 tmp;
#else // gpu only supports fp32
shared vec4 tmp;
#endif
```

fallback to *uvec2 + (un)packHalf2x16* for devices without **VK_KHR_16bit_storage uniformAndStorageBuffer16BitAccess**, or the good-old *vec4* type

fallback to *vec4* for devices without **VK_KHR_shader_float16_int8 shaderFloat16**

```
void main()
{
    const int i = int(gl_GlobalInvocationID.x);

#if NCNN_fp16_storage && NCNN_fp16_arithmetic // gpu supports 16bit storage and shader float16
    f16vec4 x = blob_data[i];
#elif NCNN_fp16_storage // gpu supports 16bit storage but no shader float16
    vec4 x = vec4(blob_data[i]);
#elif NCNN_fp16_packed && NCNN_fp16_arithmetic // gpu supports GLSL 4.2 and shader float16
    f16vec4 x = f16vec4(unpackFloat2x16(blob_data[i].x), unpackFloat2x16(blob_data[i].y));
#elif NCNN_fp16_packed // gpu supports GLSL 4.2
    vec4 x = vec4(unpackHalf2x16(blob_data[i].x), unpackHalf2x16(blob_data[i].y));
#else // gpu only supports fp32
    vec4 x = blob_data[i];
#endif
}
```

use fp16 for less memory and faster arithmetic

```
// sfp = storage floating type
layout (binding = 0) buffer blob { sfpvec4 blob_data[]; };

// lfp = local floating type
shared lfpvec4 tmp;

void main()
{
    const int i = int(gl_GlobalInvocationID.x);

    // afp = arithmetic floating type
    afpvec4 v = buffer_ld4(blob_data, i);
}
```

We define *sfp/lfp/afp* macro for AUTO fp16 types

We define *buffer_ld4* macro for AUTO load+convert fp16 data from memory

safely use fp16 according to device capability :D

code less :D

prefer specialization constant over push constant

```
layout (constant_id = 0) const int w = 0;
layout (constant_id = 1) const int h = 0;
layout (constant_id = 2) const int c = 0;

layout (push_constant) uniform parameter
{
    int w;
    int h;
    int c;
} p;

void main()
{
    int gx = int(gl_GlobalInvocationID.x);
    int gy = int(gl_GlobalInvocationID.y);
    int gz = int(gl_GlobalInvocationID.z);

    if (gx >= psc(w) || gy >= psc(h) || gz >= psc(c))
        return;
}
```

specialization constants are compile-time constants

that usually means more optimization :D

but if we have to change something after pipeline creation, we need *push constants*

that usually means more register usage and harder optimization

`#define psc(x) x==0?p.x:x`

create one general shader code for combined static/dynamic usage

For more ncnn pre-defined macros, see https://github.com/Tencent/ncnn/wiki/glsl-extension

simplevk vulkan loader

We switched to simplevk vulkan loader from khronos vulkan loader in 2024

WHY ?

We don’t want to setup vulkan sdk for building vulkan stuff, we are lazy :D

We don’t want to bundle the libvulkan.so / vulkan-1.dll on redistrbution

We want to build vulkan applications for old Android deployment targets (pre-24)

We want to easily switch devices or vulkan drivers, sometimes

simplevk vulkan loader

```
#include "simplevk.h"

int main()
{
    // from system vulkan and VK_ICD_FILENAMES env
    // and NCNN_VULKAN_DRIVER env
    ncnn::create_gpu_instance();

    // load directly from specific driver path
    ncnn::create_gpu_instance("libvulkan.so");
    ncnn::create_gpu_instance("/usr/lib64/libvulkan_radeon.so");
    ncnn::create_gpu_instance("/vendor/lib64/hw/vulkan.adreno.so");
    ncnn::create_gpu_instance("/data/local/tmp/vulkan.ad07XX.so");

    VkInstance inst = ncnn::get_gpu_instance();

    // populate some interesting entrypoints
    ncnn::vkGetInstanceProcAddr(inst, ...);

    ncnn::destroy_gpu_instance();

    return 0;
}
```

If driver_path == 0

- 1a from env *VK_ICD_FILENAMES*
- 1b from env *NCNN_VULKAN_DRIVER*

If driver_path != 0

- 1 from specified driver_path

2 from *vulkan-1.dll / libvulkan.so / libvulkan.dylib* in system

3 search driver by name *nvoglv64.dll / amdvlk64.dll / libGLX_nvidia.so.0 ....* and load it

https://github.com/Tencent/ncnn/wiki/vulkan-driver-loader

https://github.com/Tencent/ncnn/blob/master/src/simplevk.h

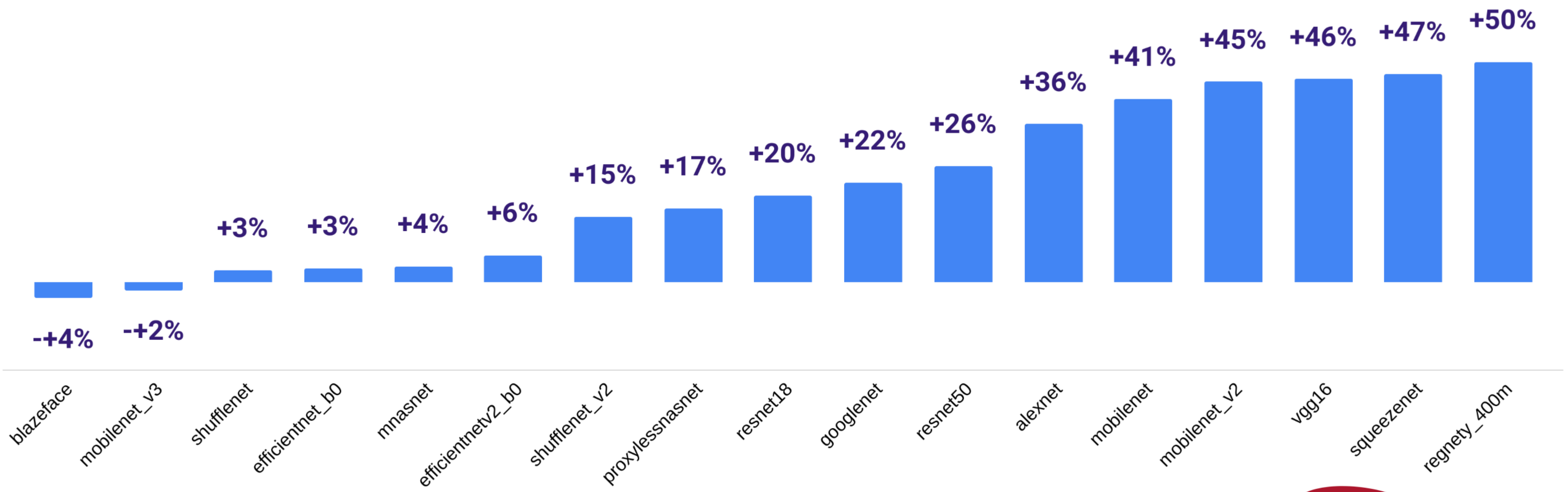
https://github.com/Tencent/ncnn/blob/master/src/simplevk.cpp

simplevk vulkan loader

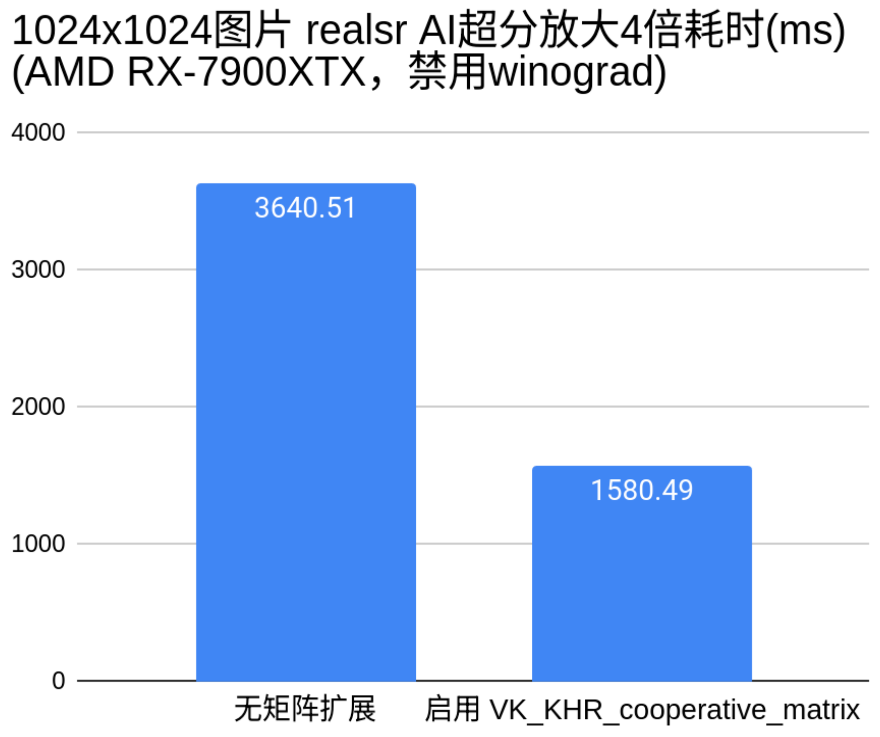
mesa turnip driver speedup over adreno blob on samsung s20 (SDM865)

FPS, larger is better

https://github.com/K11MCH1/AdrenoToolsDrivers/releases/tag/v24.1.0_R18



VK_KHR_coorperative_matrix

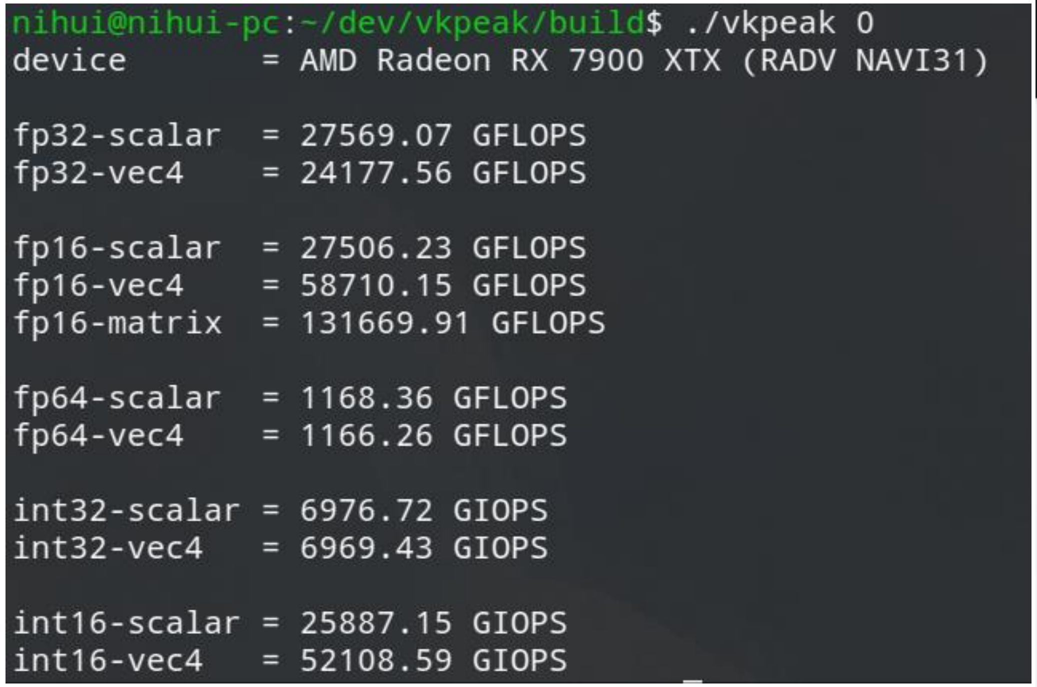


https://github.com/KhronosGroup/Vulkan-Docs/blob/main/proposals/VK_KHR_cooperative_matrix.adoc

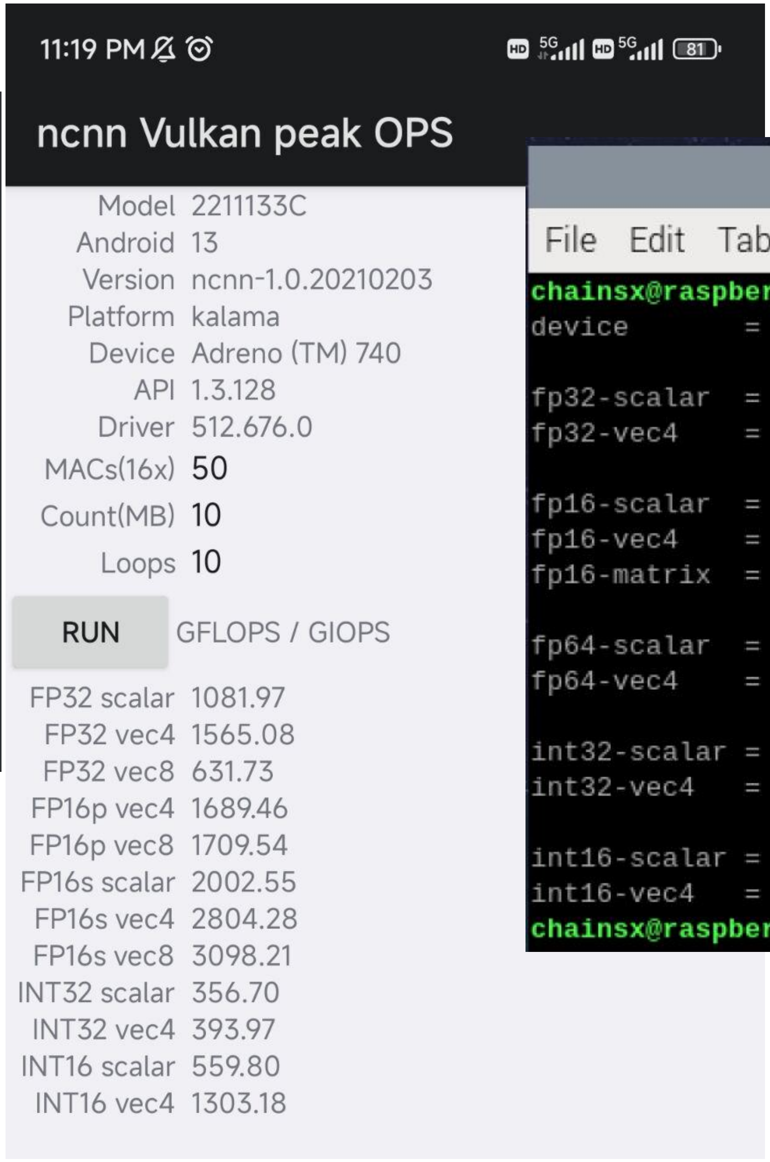
https://github.com/KhronosGroup/SPIRV-Registry/blob/main/extensions/KHR/SPV_KHR_cooperative_matrix.asciidoc

https://github.com/KhronosGroup/GLSL/blob/main/extensions/khr/GLSL_KHR_cooperative_matrix.txt

vkpeak: find device peak capabilities



```
nihui@nihui-pc:~/dev/vkpeak/build$ ./vkpeak 0
device        = AMD Radeon RX 7900 XTX (RADV NAVI31)

fp32-scalar   = 27569.07 GFLOPS
fp32-vec4     = 24177.56 GFLOPS

fp16-scalar   = 27506.23 GFLOPS
fp16-vec4     = 58710.15 GFLOPS
fp16-matrix   = 131669.91 GFLOPS

fp64-scalar   = 1168.36 GFLOPS
fp64-vec4     = 1166.26 GFLOPS

int32-scalar = 6976.72 GIOPS
int32-vec4    = 6969.43 GIOPS

int16-scalar = 25887.15 GIOPS
int16-vec4    = 52108.59 GIOPS
```



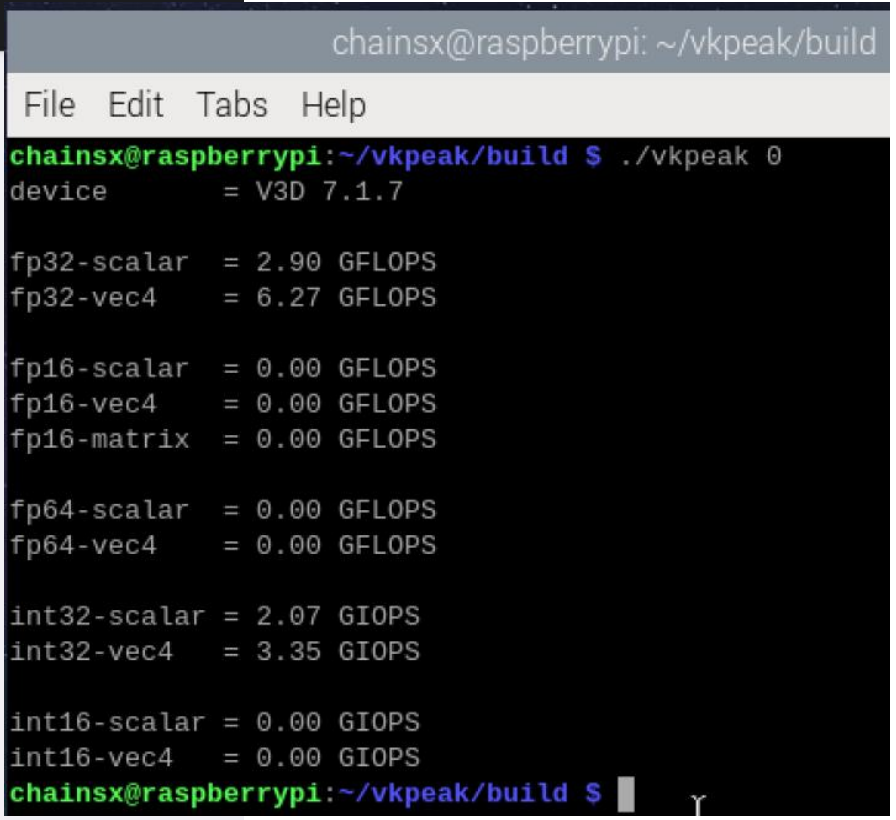
11:19 PM

ncnn Vulkan peak OPS

Model	2211133C
Android	13
Version	ncnn-1.0.20210203
Platform	kalama
Device	Adreno (TM) 740
API	1.3.128
Driver	512.676.0
MACs(16x)	50
Count(MB)	10
Loops	10

RUN GFLOPS / GIOPS

FP32 scalar	1081.97
FP32 vec4	1565.08
FP32 vec8	631.73
FP16p vec4	1689.46
FP16p vec8	1709.54
FP16s scalar	2002.55
FP16s vec4	2804.28
FP16s vec8	3098.21
INT32 scalar	356.70
INT32 vec4	393.97
INT16 scalar	559.80
INT16 vec4	1303.18



```
chainsx@raspberrypi:~/vkpeak/build $ ./vkpeak 0
device        = V3D 7.1.7

fp32-scalar   = 2.90 GFLOPS
fp32-vec4     = 6.27 GFLOPS

fp16-scalar   = 0.00 GFLOPS
fp16-vec4     = 0.00 GFLOPS
fp16-matrix   = 0.00 GFLOPS

fp64-scalar   = 0.00 GFLOPS
fp64-vec4     = 0.00 GFLOPS

int32-scalar = 2.07 GIOPS
int32-vec4    = 3.35 GIOPS

int16-scalar = 0.00 GIOPS
int16-vec4    = 0.00 GIOPS
chainsx@raspberrypi:~/vkpeak/build $
```

https://github.com/nihui/vkpeak

https://github.com/nihui/ncnn-android-vkpeak

future plan

- int8 glsl extensions
- int8 cooperative matrix
- bfloat16 ? int4 ?
- op-primitive or ml-graph extension ?

- slang ?   https://github.com/shader-slang/slang

Q & A

Thank You

Danke

Gracias

谢谢

ありがとう

Asante

Merci

감사합니다

ध�वाद

Kiitos

شكرًا

धन्यवाद

תודה

他们都不知道 pnnx 有多好用群

current status