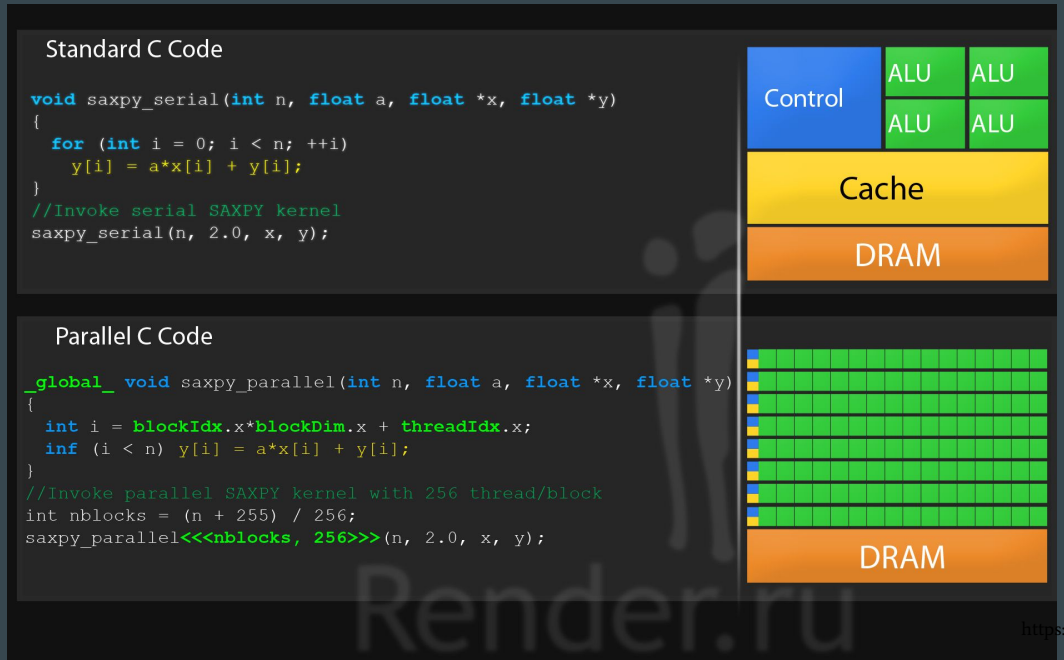
Dynamic Voltage and Frequency Scaling on Embedded Systems

By Emerson Jacobson

What is a GPU

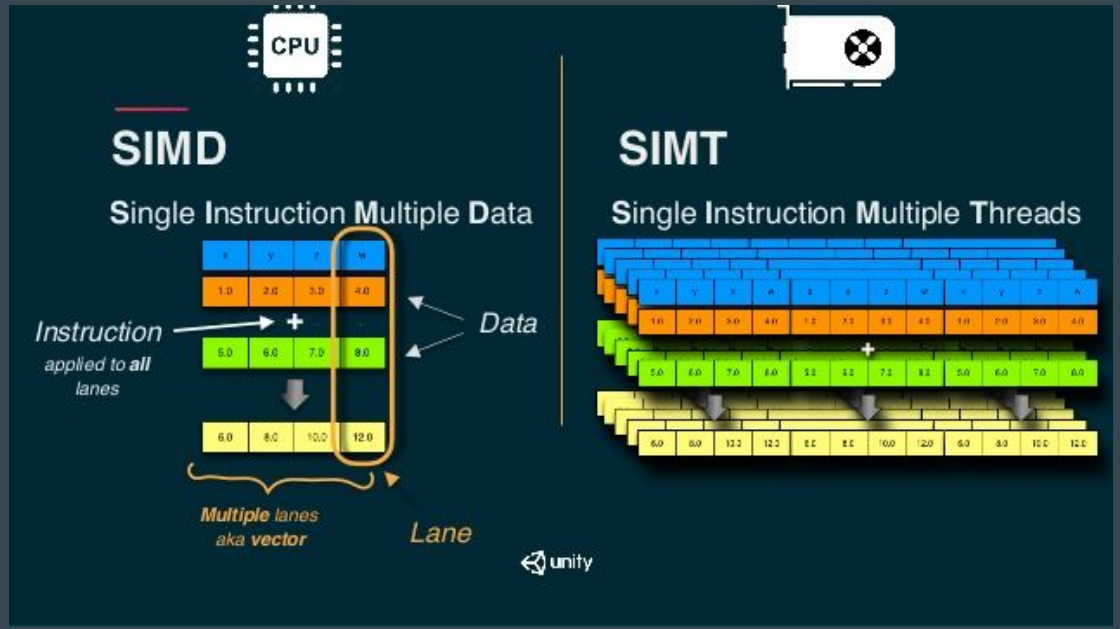
- Graphics Processing Unit
- GPGPU - General Purpose GPU
- Libraries CUDA (Nvidia) and ROCm (AMD)



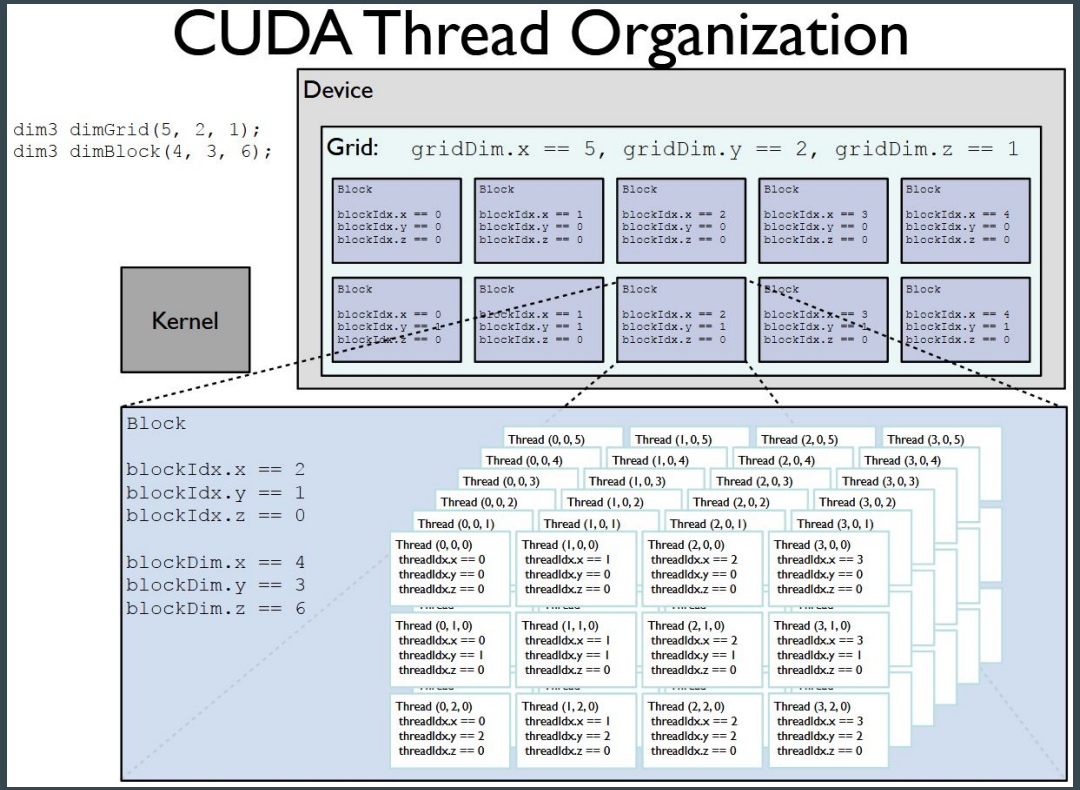
https://normbest.letua.info/nvidia-cuda-75-for-mac.html

SIMT

- Single Instruction Multiple Thread
- *Can* become inefficient with branching

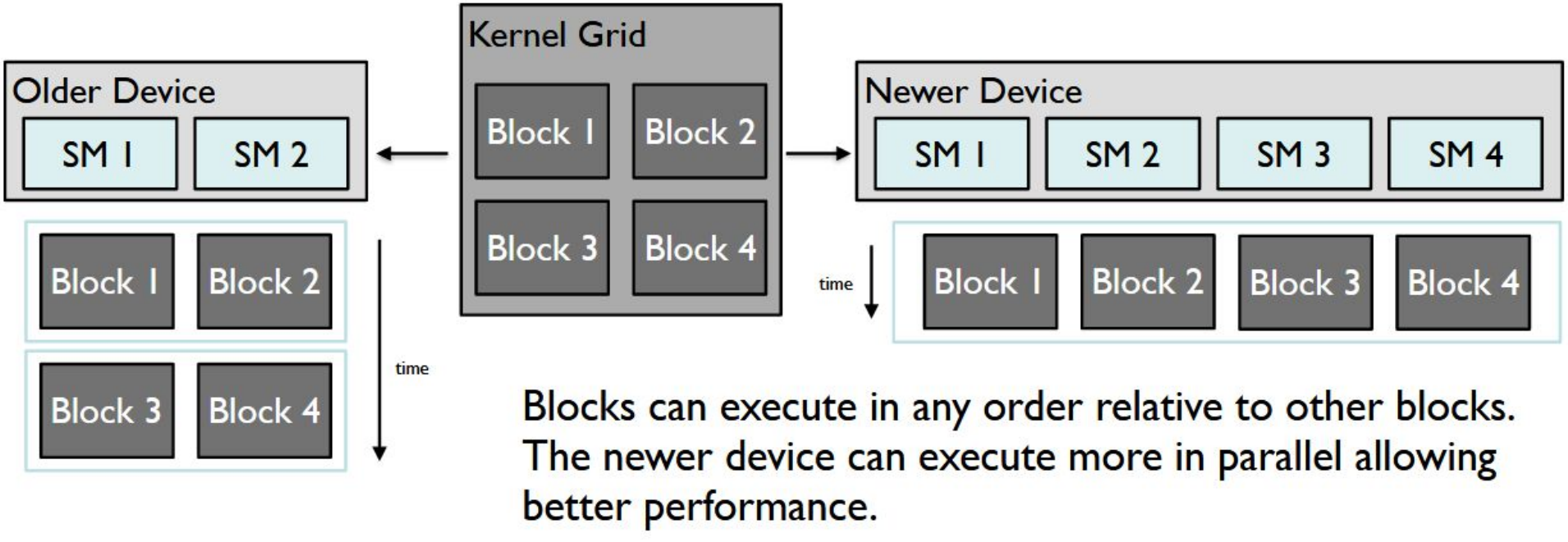


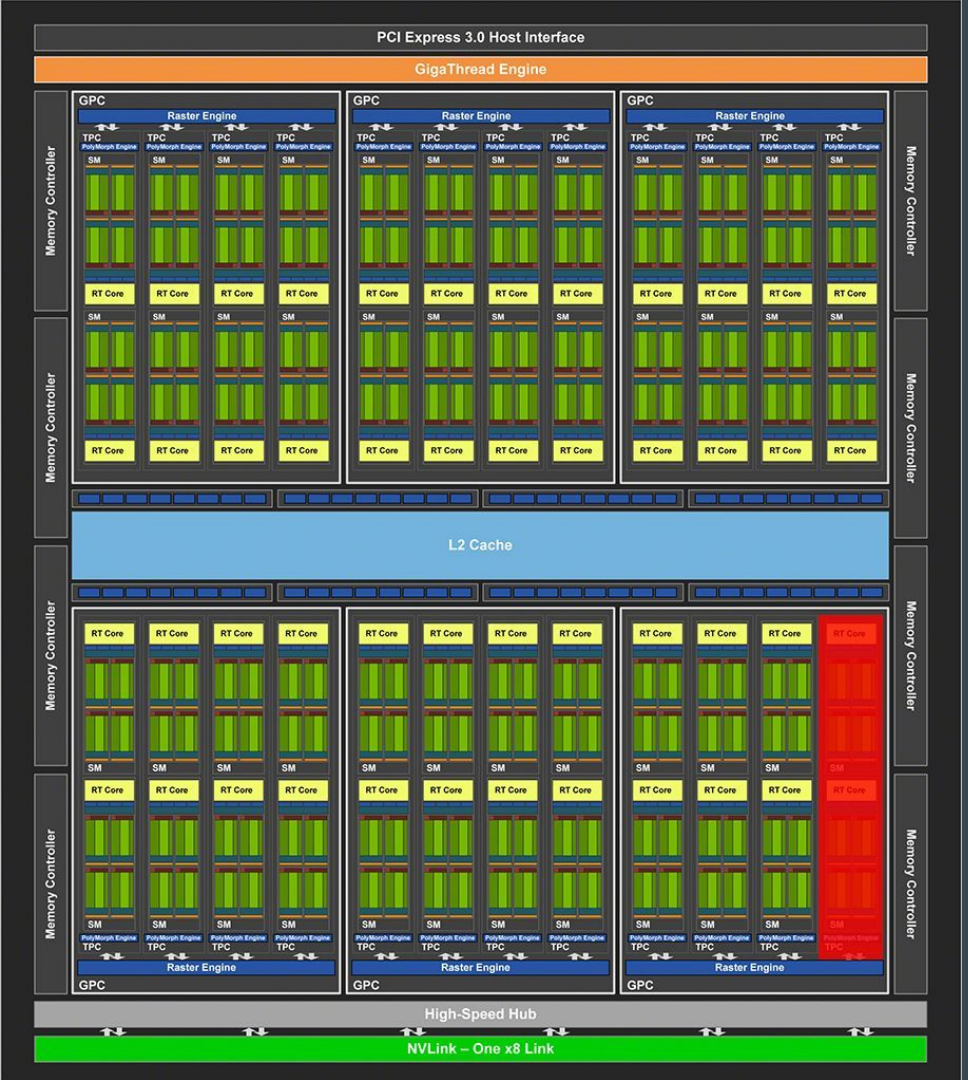
Grid/Blocks/Threads



http://tdesell.cs.und.edu/lectures/cuda_2.pdf

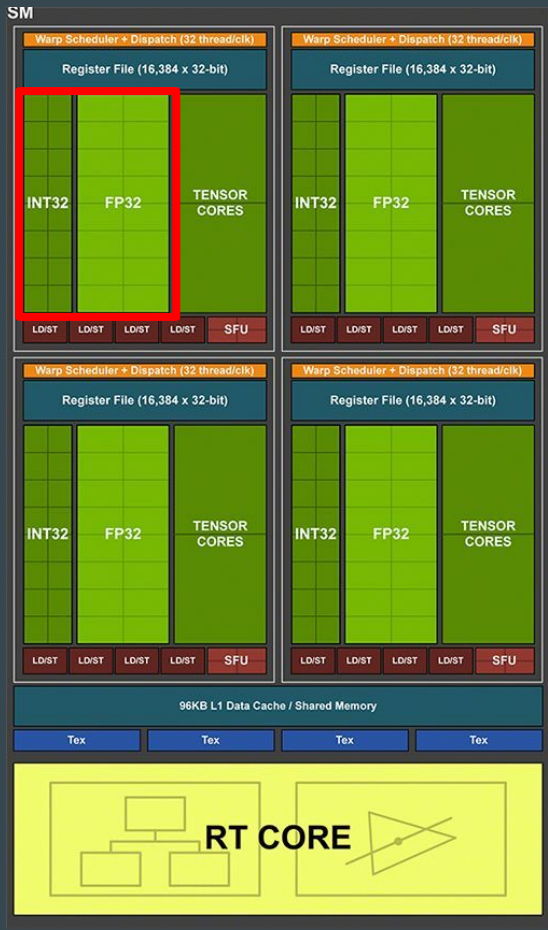
Block Scheduling





RTX 2080

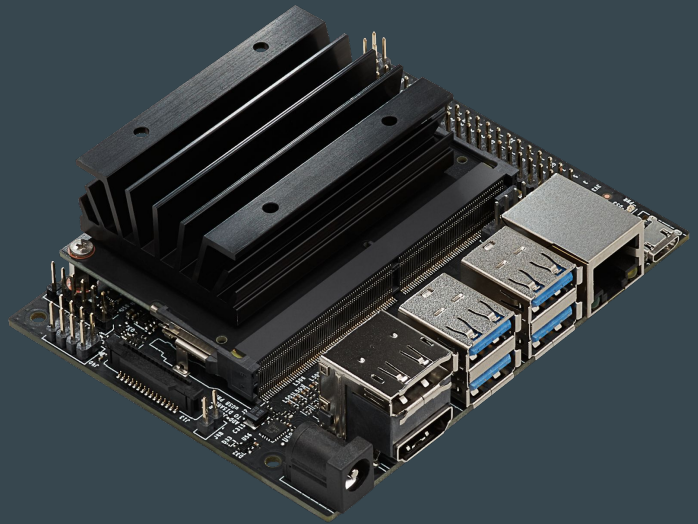
- 46 SM (Streaming Multiprocessor)
- 2944 CUDA Cores



Embedded Systems

Jetson Nano (10W)

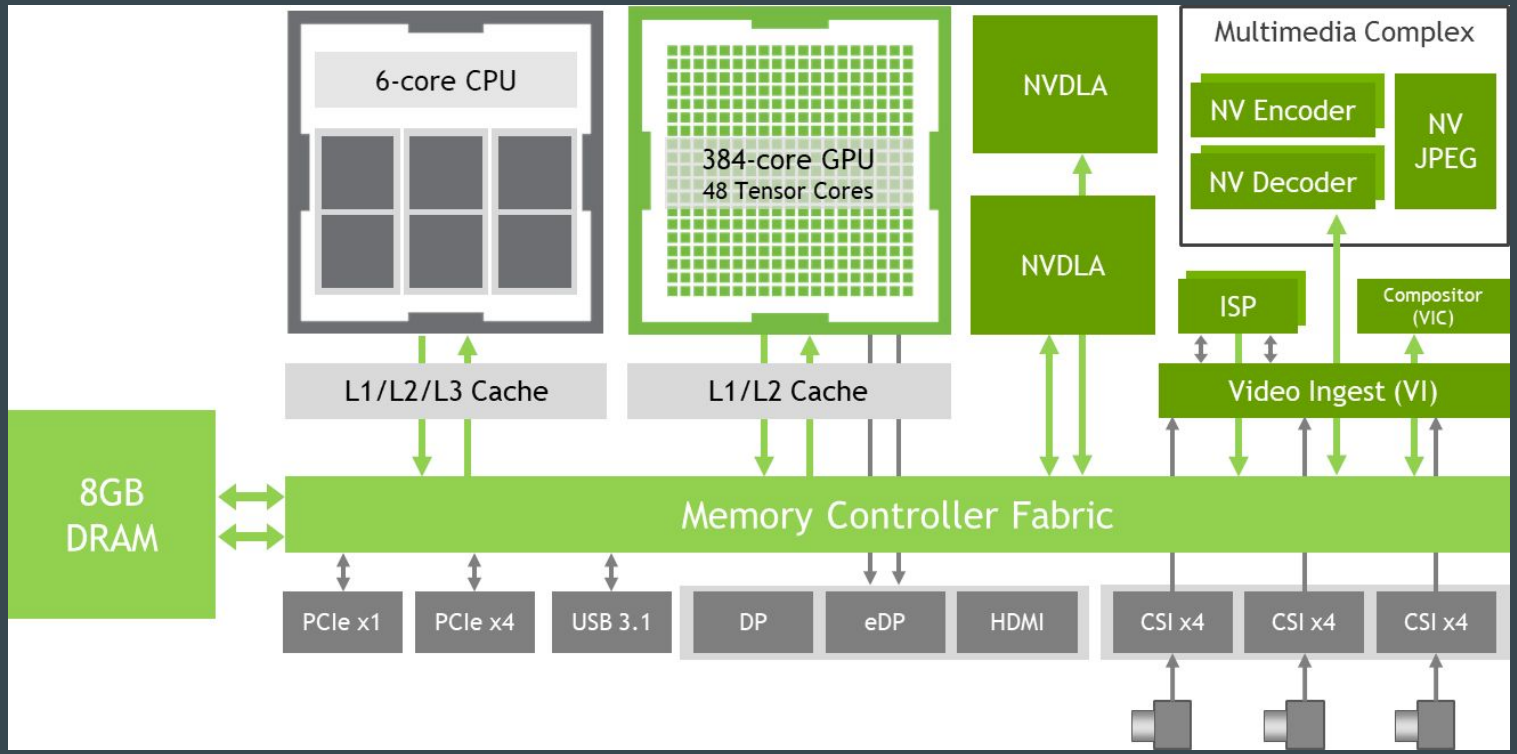
- 128 GPU Cores
- 4 CPU Cores



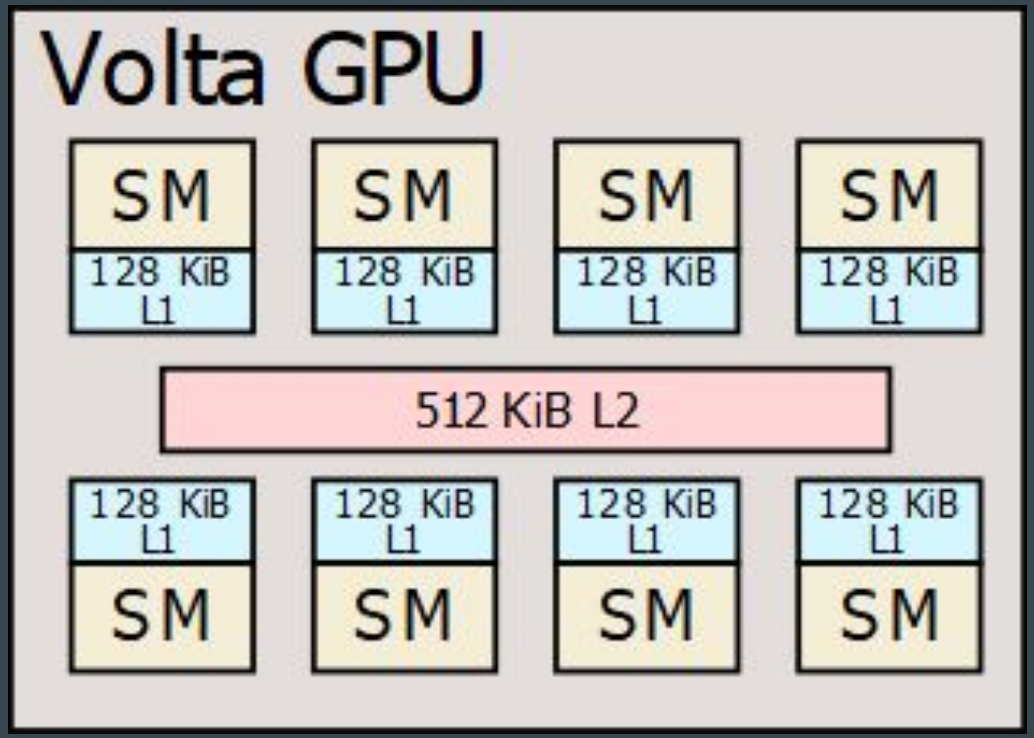
Jetson AGX Xavier (55W)

- 512 GPU Cores
- 8 CPU Cores

Jetson Architecture

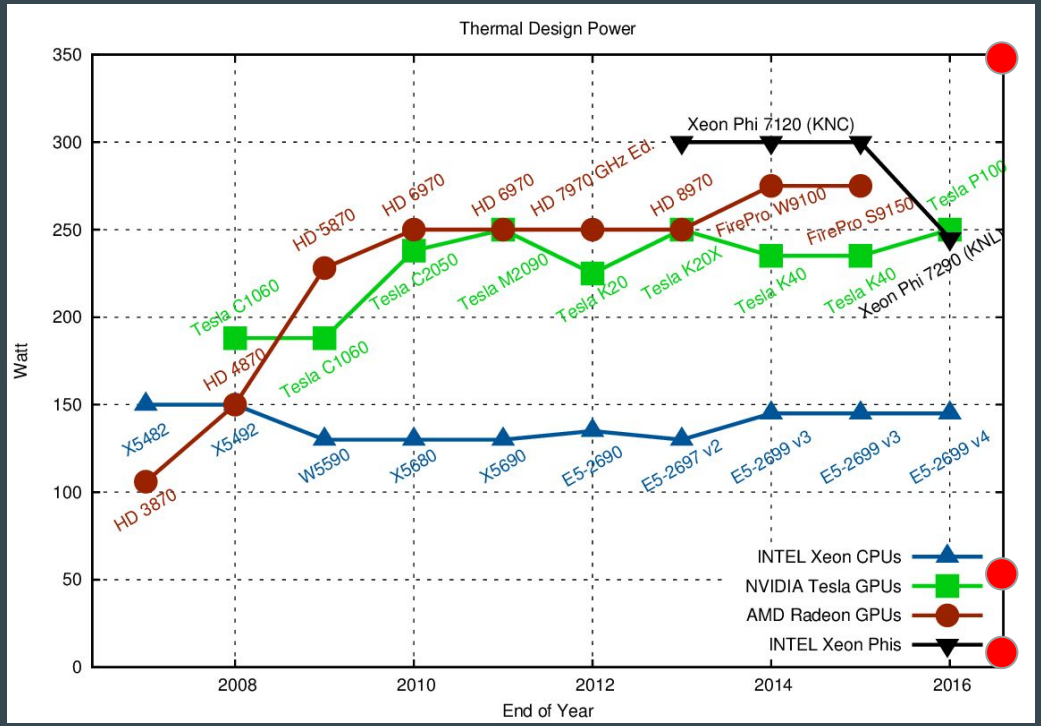


The Jetson AGX Xavier



- 8 SM
- 64 Cores per SM

Power Consumption



RTX 3090

AGX Max Power (55W)

Nano Max Power (10W)

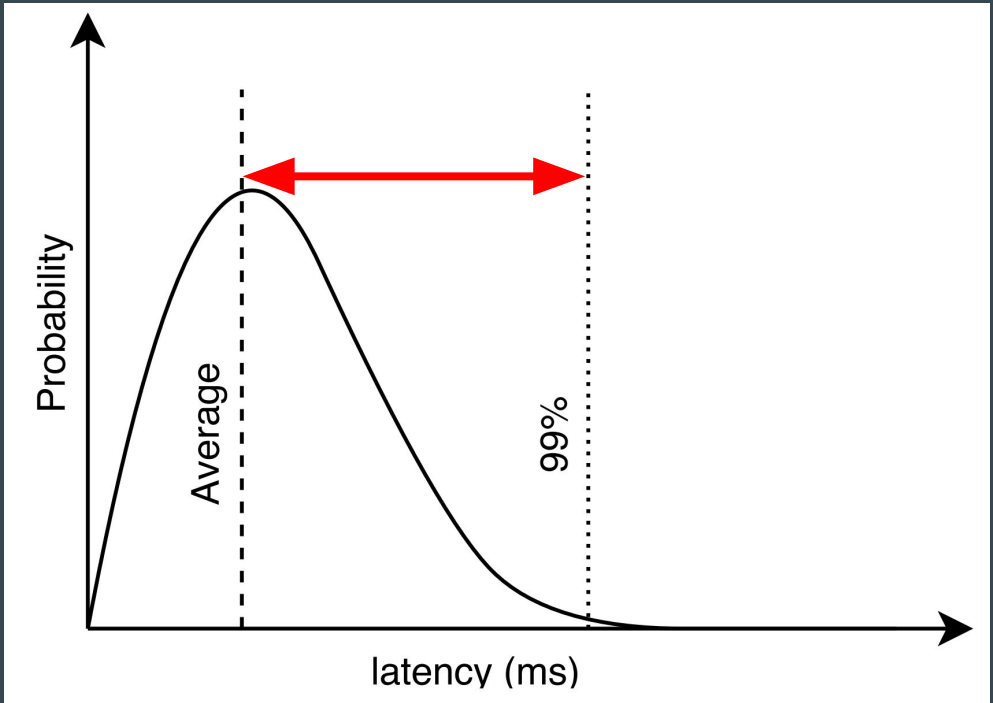
Power Consumption

- “In field” applications
- Battery powered

DVFS

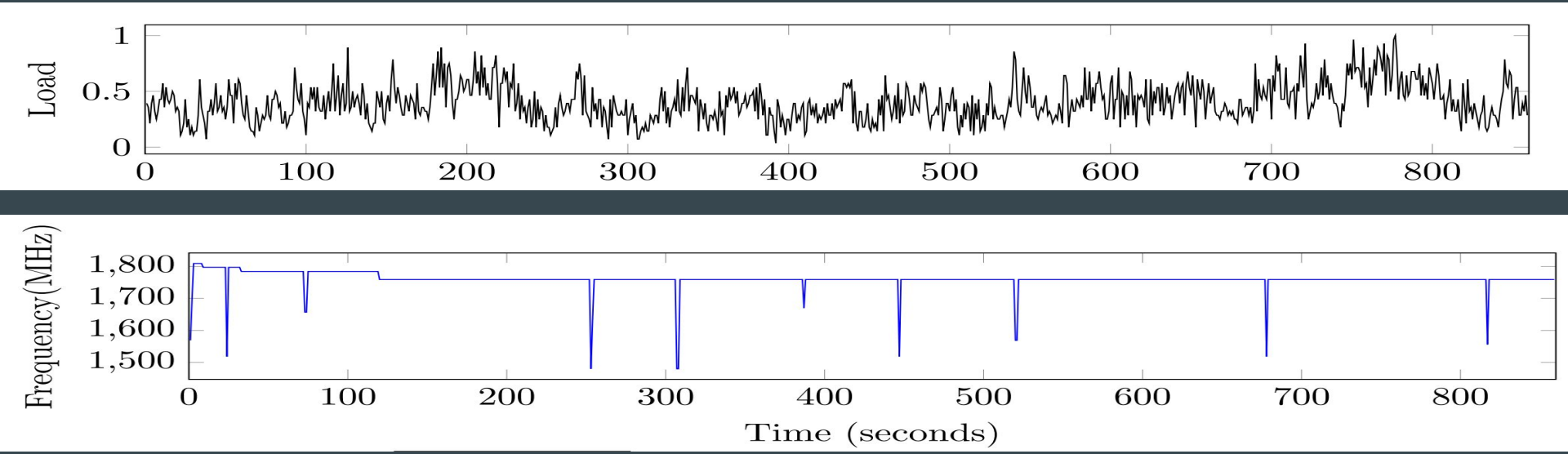
- **D**ynamic **V**oltage and **F**requency **S**caling
- No real support for modifying voltage
  - Reduce frequency -> reduce power consumption

DVFS - Latency Deadlines



- Need to meet QoS
- Want latency within 99%

DVFS - Power Usage

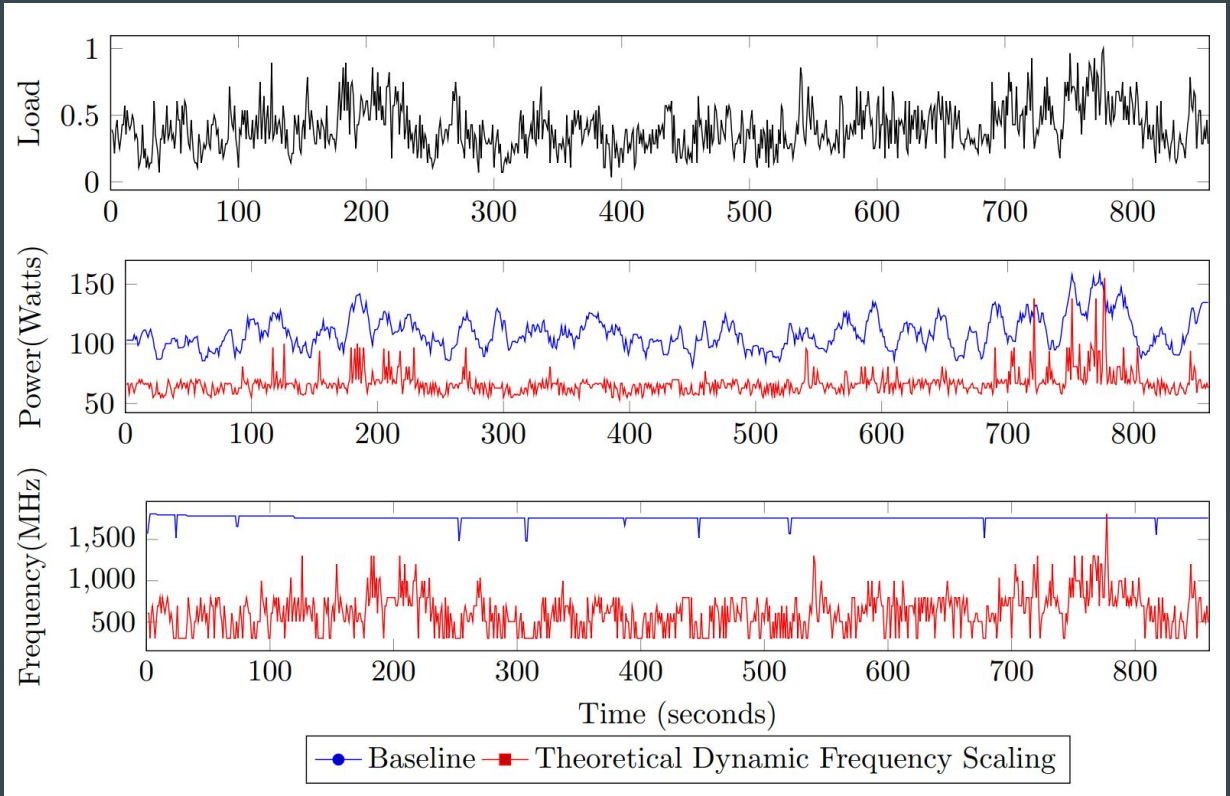


DVFS - Frequency Scaling

Load	Frequency	Power	Load	Frequency	Power
5	303	53	55	898	70
10	303	53	60	898	75
15	303	55	65	1202	81
20	303	57	70	1202	80
25	303	61	75	1404	96
30	506	59	80	1404	96
35	607	67	85	1404	101
40	797	66	90	1809	155
45	797	67	95	1809	150
50	898	70	100	1809	155

- Find the optimal frequency for some load

DVFS - Power Savings



DVFS - Heterogeneous Systems

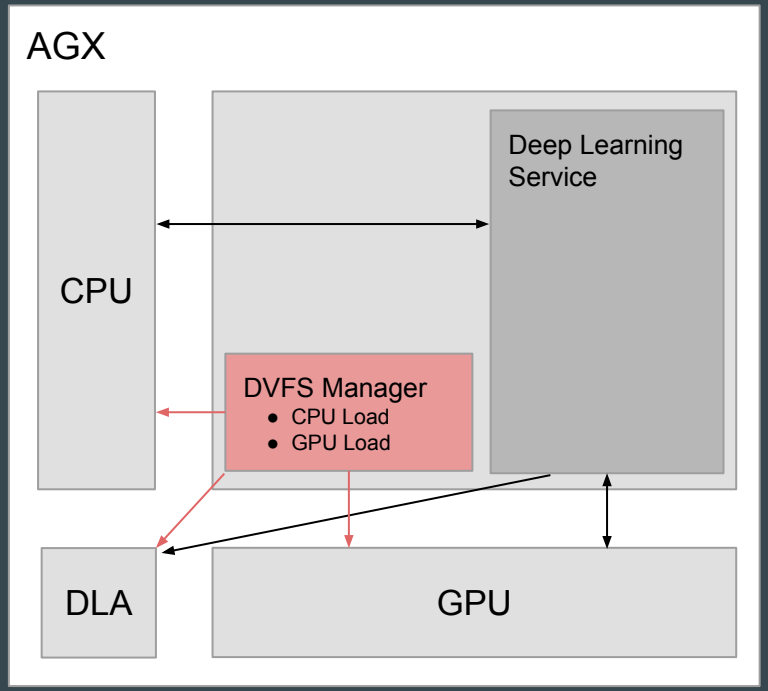
- CPU and GPU share the power cap
  - Balance power to CPU or GPU

DVFS - AGX Power Modes

Property	Mode				
	MAXN	10W	15W	30W	30W
Power budget	n/a	10W	15W	30W	30W
Mode ID	0	1	2	3	4
Online CPU	8	2	4	8	6
CPU maximal frequency (MHz)	2265.6	1200	1200	1200	1450
GPU TPC	4	2	4	4	4
GPU maximal frequency (MHz)	1377	520	670	900	900

DVFS - Heterogeneous Power Scaling

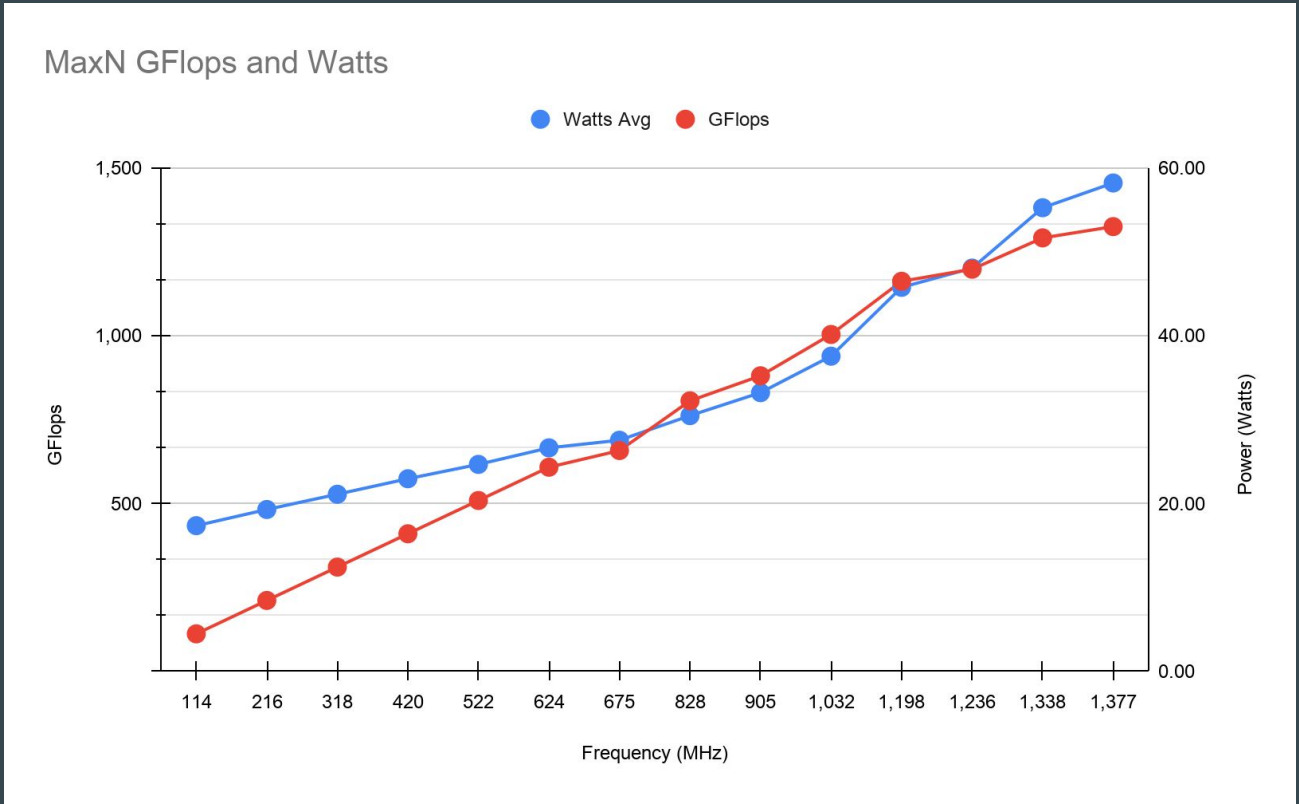
- Power split based off current CPU/GPU load
- DVFS manager use current CPU and GPU load
- jetson_clocks



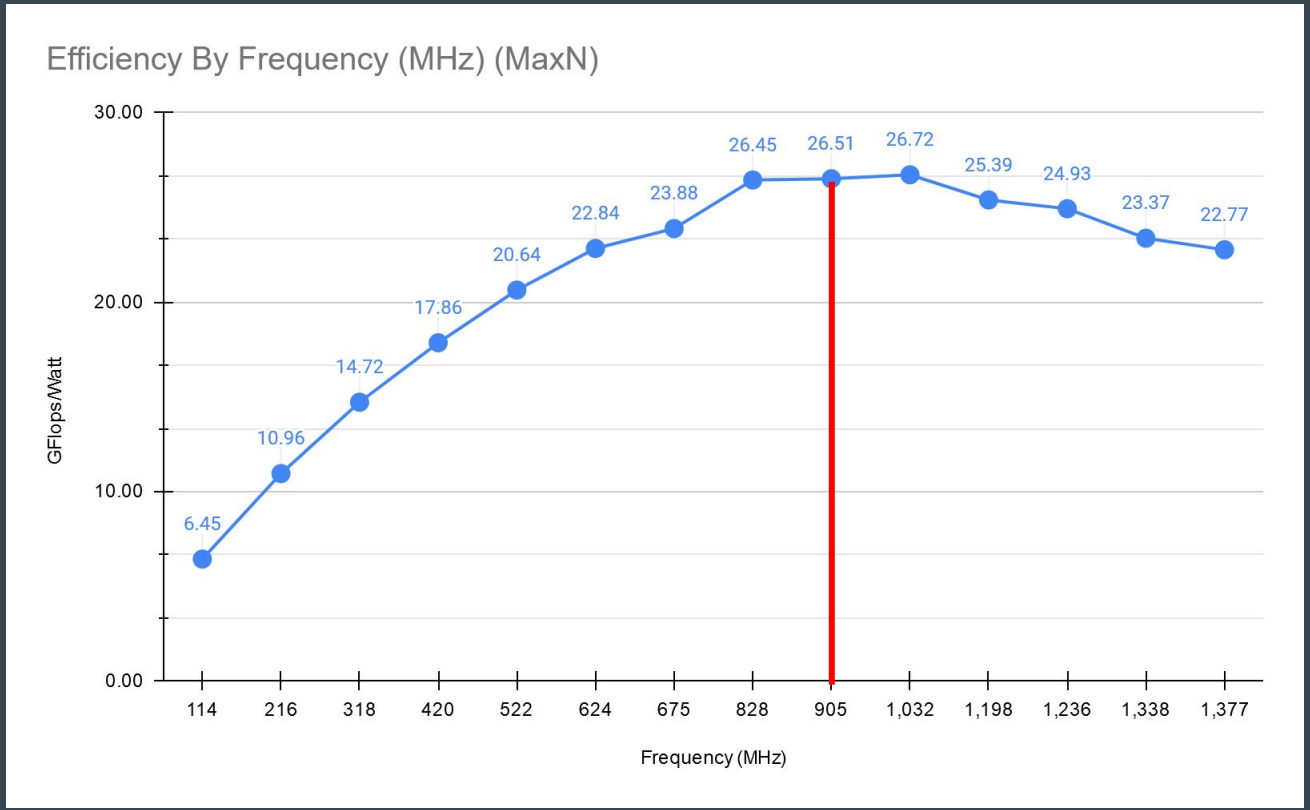
FLOPs Benchmark

- Floating Point Operations Per Second
- Loop ran doing matrix multiplication on the GPU
- Number of operations are known, time it takes can be measured
  - FLOPs = operations/time
- Frequencies are static for benchmarking purposes

FLOPS by Frequency



FLOPS by Frequency



Deep Learning Benchmarks

- Inference performance
- Uses Nvidia's TensorRT
- Throughput measured based on latency
- Batch sizes, streams

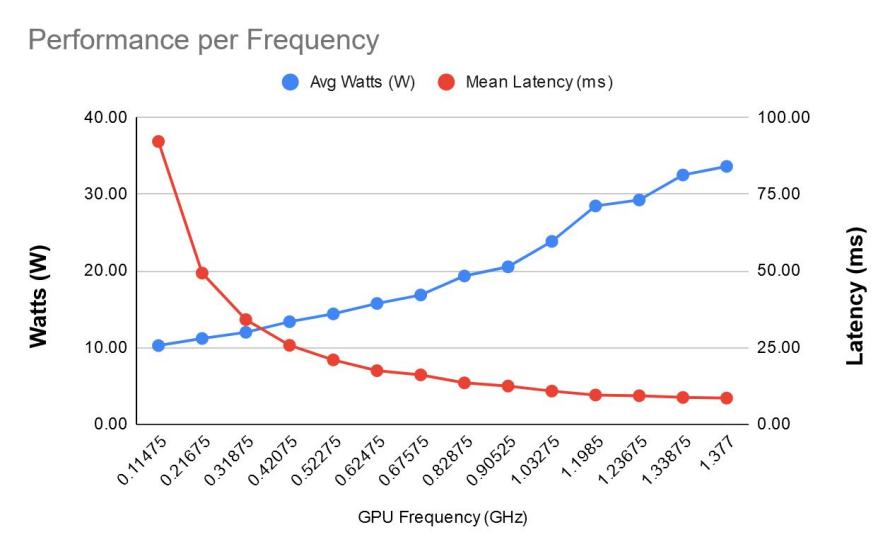
https://github.com/NVIDIA-AI-IOT/jetson_benchmarks

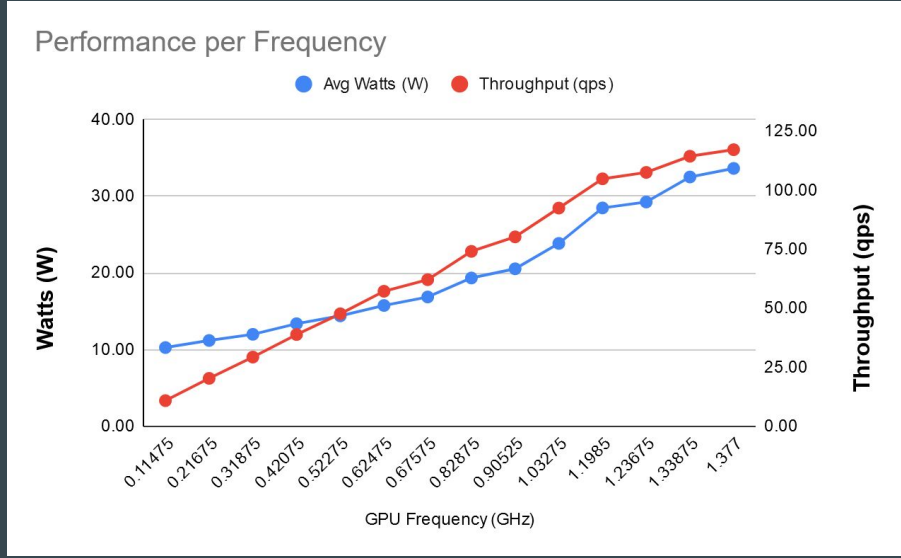
TensorRT

- Deep learning inference optimizer
  - Reduced precision -> reduced latency
- inception_v4 -> TensorRT engine
- Use ‘trtexec’ for benchmark

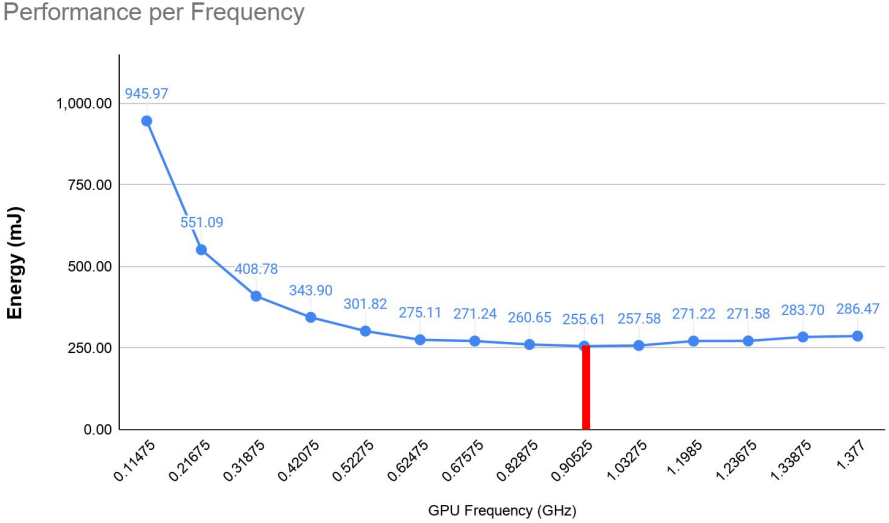
https://developer.nvidia.com/tensorrt

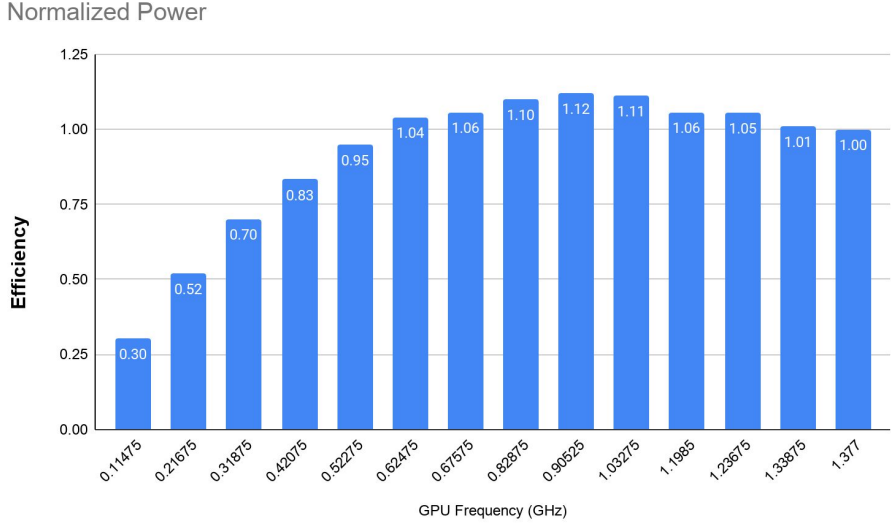
Deep Learning Performance





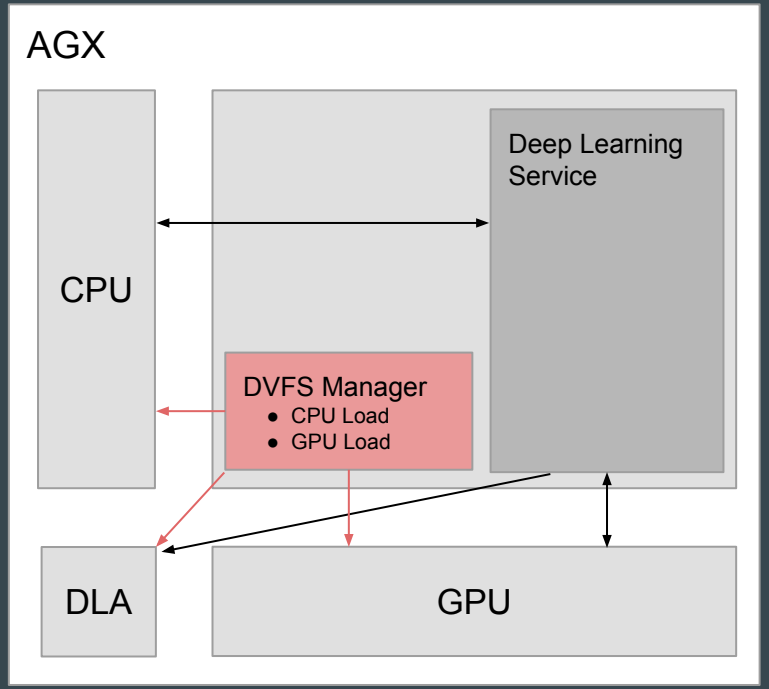
Deep Learning Energy





Future Work

- Scaling CPU & GPU with current workload
- Run benchmarks on Jetson Nano
- Usage of DLAs on Jetson AGX
- Non-TensorRT benchmarking
  - Caffe



Questions?