Neural Processing Unit (NPU)

AI hype keeps going without losing any pace these days. Every morning, we wake up on a day when new records have been broken on Wall Street. Undoubtedly, semiconductor giants have a significant stake in this sharp trend. What makes blue-chip companies popular is their specific solutions to handle a high amount of AI load due to massive models. The first companies that come to mind are NVIDIA, Qualcomm, AMD, and Apple (Apple is not a pure semiconductor company, but its success is strongly correlated with its approach to bottlenecks of AI loads). How they manage to accelerate models without enormously increasing power consumption lies under the new concept: NPU. In this blog, we will try to answer some questions such as “What is an NPU?”, “ Why is it popular now?” “ For which applications can one use NPU?” and “What are the challenges facing NPU?”. Lastly, we will compare popular NPU architectures and companies’ problem approaches.

Figure1: AI hype

Let's start with the most fundamental question.

- **What is NPU?**

NPU, which stands for Neural Processing Unit, is a designed integrated circuit for accelerating AI-related tasks. One may wonder what is the difference between NPU and CPU. CPUs are also processing units but are responsible for a wide range of tasks, such as performing calculations, managing data flow, controlling peripheral devices, and running the operating system. Therefore, it is hard for them to be as fast as NPUs for some AI-related tasks. Generally, there is a trade-off between various instruction sets and throughput. On the other hand, speed is not the only problem. It depends on the application, but power consumption may be another bottleneck, especially for IoT devices.

AI workloads primarily calculate neural network layers comprised of scalar, vector, and tensor math followed by a non-linear activation function. Each of these layers expects different hardware configurations. So, we can think of NPU as a system that includes specialized computation engines, memory blocks, and Interfaces. NPUs excel in processing vast amounts of data in parallel.

One crucial part is sometimes confusing: Generally, NPU is an inference chip. Inference and training are two different sides of an AI application. Training means optimizing models by varying weights and so on (with backpropagation, which has different workloads). However, inference is the part in which we use predetermined weights for our model and try to exploit the model for actual applications.

NPUs can be integrated into SoC with other processing units such as CPU or GPU. This integration reduces the load on the different processors, leading to more efficient computer operation. This is expected because, due to the alleviation of AI-related tasks, the CPU can focus on its duties, or the GPU could realize graphic visualization. One of the blue chips of the semiconductor industry, Qualcomm, recommends heterogeneous use of processors. Choosing the right processor seems like selecting the right tool from the toolbox. Every processor excels in different tasks. If some system optimizes the usage of these various processors, it can reach maximum efficiency and throughput. For example, the CPU is used for sequential control and immediacy, the GPU is used for streaming parallel data, and the NPU is used for core AI workloads with scalar, vector, and tensor math.

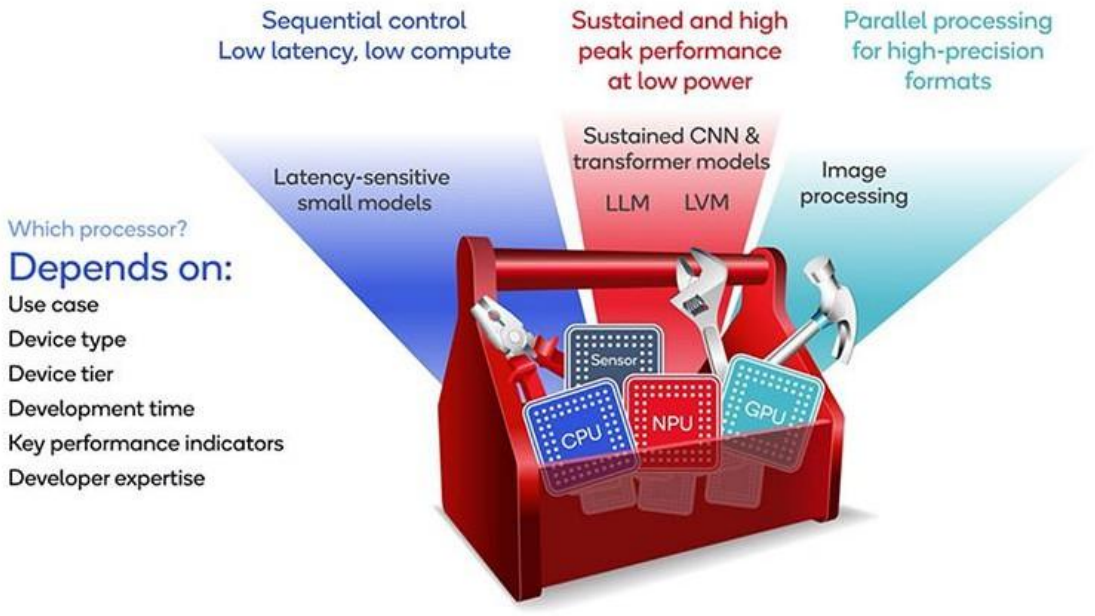


Figure 2 : Qualcomm's heterogenous use recommendation

- **Why has NPU gained popularity now?**

Most of the innovations stem from the needs of markets and consumers. From the day the first transistor was innovated to today, increasing throughput and efficiency of hardware to satisfy modern software's needs was always in the first place. At this stage, we assume that if you reserved some time for this blog, you probably have heard about Moore's Law before. Most basically, this rule is trying to tell us that the industry side of the sector will manage to increase the number of transistors on the same area of a chip by approximately 2 times every two years. If we further simplify the idea, physics is the reason behind this shrinking. Since the distance that electron travels inside the chip is reduced to half, this shrinking will reduce the power consumption. More than this, the operation per second will increase due to faster clock frequencies, which enables faster integrated circuits.

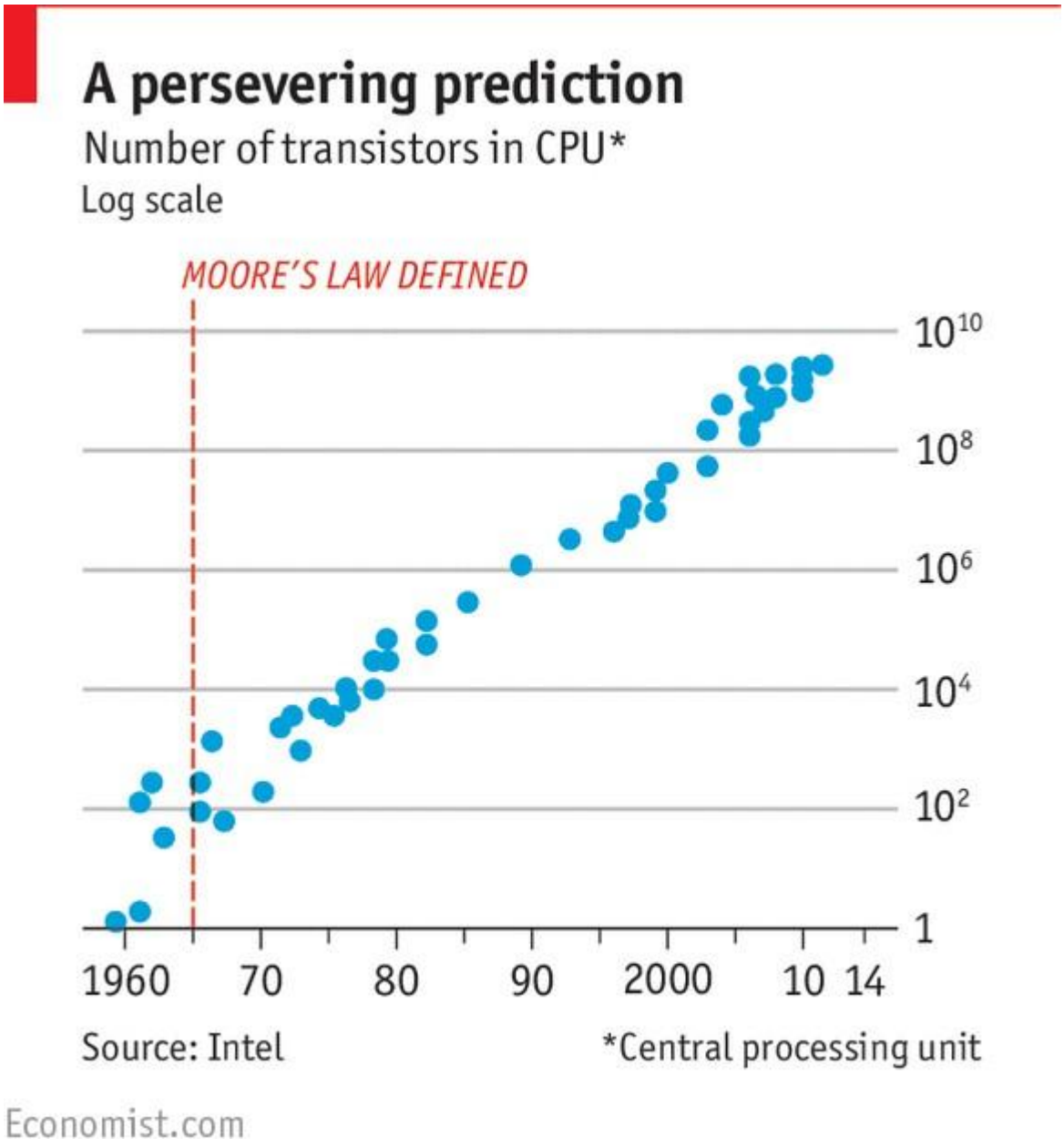


Figure 3 : Moore's trend

However, the law lost validity after some point because of its physical limitations. Supplied power from the source stopped to reduce further, even if you decrease the size of the transistors. Therefore, with the increasing number of transistors per area, power consumption per unit area also increased. Consequently, we faced with heating problems. These problems are serious on their own. Unfortunately, we have more issues that stem from the increasing demand for computing capabilities due to huge AI models. So, both sides of the equation drove traditional techniques to a bottleneck. Consequently, the only solution was to utilize existing resources effectively.

One may think that this is the part where NPU comes up to the stage. But this is not the case. The transition between traditional CPUs and NPUs is not that sharp. We need to go back to the 2012 ImageNet contest to see the whole picture. This contest was held annually between 2010 and 2017 and aimed to create the most effective object detection and image classification algorithm. Geoffrey Hinton, Ilya Sutskever, and Alex Krizhevsky submitted their model, AlexNet, in the ImageNet competition 2012. They used CNN, and their model showed significant improvement. But the weirdest thing is that this kind of CNN architecture was available back in the nineties and was not new. Their innovation lies in using GPUs to train their model to get good inference results. This fact sheds light on how important the hardware side of AI is. So, as a solution to our first fundamental problem, GPUs have shown revolutionary progress due to their ability to handle multiple operations concurrently with multiple cores. GPUs were a good match because they could handle linear algebraic operations in parallel and significant amounts.

- **Targeted use cases of NPU**

The increasing integration of Neural Processing Units (NPUs) into PCs, laptops, and other smart devices can be attributed to their capacity to perform AI-related tasks quickly. NPUs are vital in object identification, photo enhancement, video editing, and document drafting applications. In addition, they are employed in text summarization, language translation, camera settings adjustment, picture effects creation, and content identification.

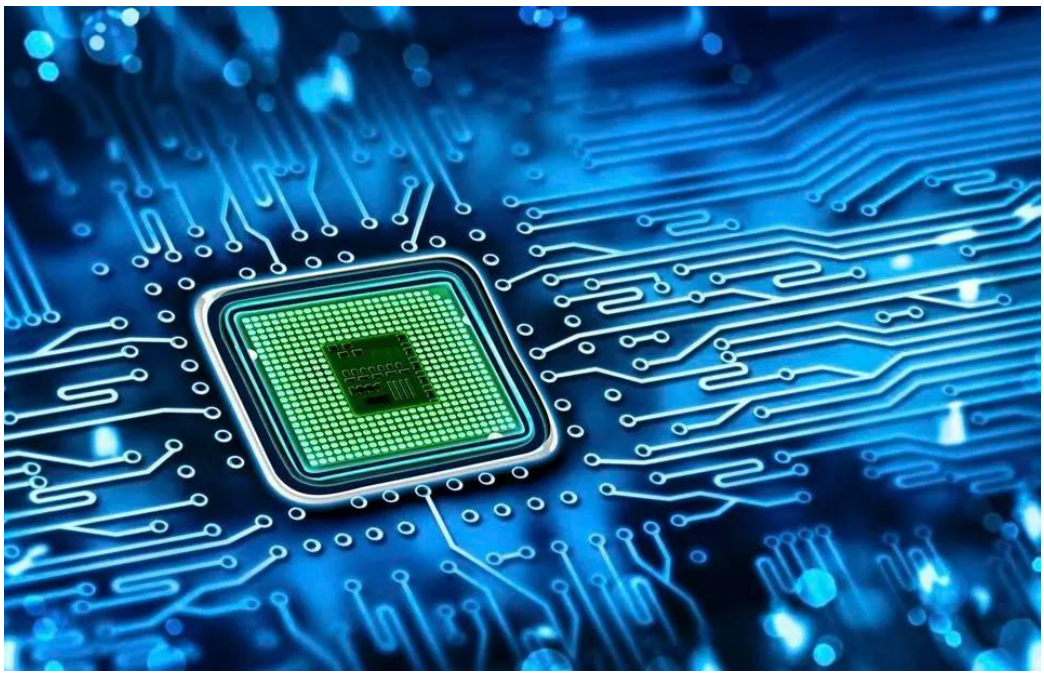


Figure 4 : NPU applications

NPUs are used in televisions to upgrade older movie scenes to current 4K quality. Television is not the only smart home device that uses NPUs. Also, NPUs are essential for devices that need data privacy. They do this by enabling edge devices to handle sensitive data locally and diminish the need to transport data to external hubs. The security and confidentiality of smart home devices depend on this edge processing capability.

Another example is natural language processing, which is used in virtual assistants like Apple's Siri and Samsung's Bixby Vision. In the first versions, Apple Neural Engine was used for FaceID.

Another example is a trending technology. NPUs are very critical for autonomous cars. They have to complete the inference in real-time. Otherwise, they may face an accident.

- **Challenges**

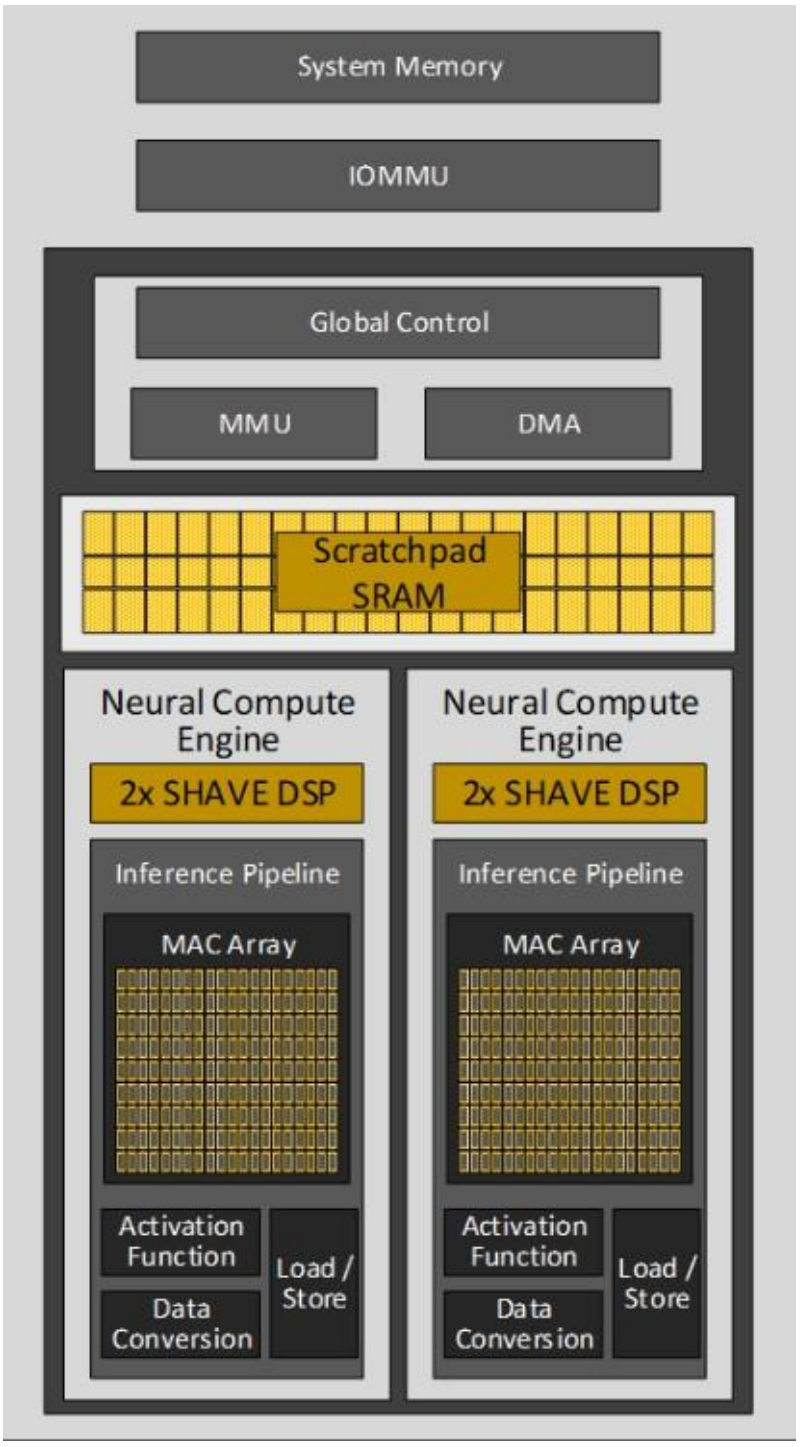
The most common challenge with NPU is the first step. Detecting the needs of models and market constraints is the hardest part. Most of the time, you don't have unlimited resources to design your circuit, such as power consumption or space. You have to guess approximately two years' future for the market since designing, verification, fabrication, and packaging processes take considerable amount of time. Lastly, your design must be as versatile as possible to run a wide range of models efficiently. One other problem is memory itself. If you think the most problematic part is computing, you have a huge mistake. Most of the energy and time is spent to reach memory during NPU runtime. Because of this reason, NPU architectures try to reduce external memory access. As we will see while examining different architectures, designers located memory blocks inside the chip to minimize external memory access. In contrast to that, some of them tried to locate computational blocks near memories.

Lastly, sometimes, even if you accept to pay the price as energy defects, you cannot increase your computational power further due to memory bandwidth problems. Imagine your computational resources as a monster. It does not matter how fast you can compute if you cannot feed it with enough data. It is hard to design a successful architecture without solving memory bandwidth problems.

NPU Architectures

This part will examine different NPU architectures from popular blue chips and start-ups. There will be terminological concepts, but no worries, they are not that complicated. You can get the idea with a small search on the internet. I will try to show different ideas.

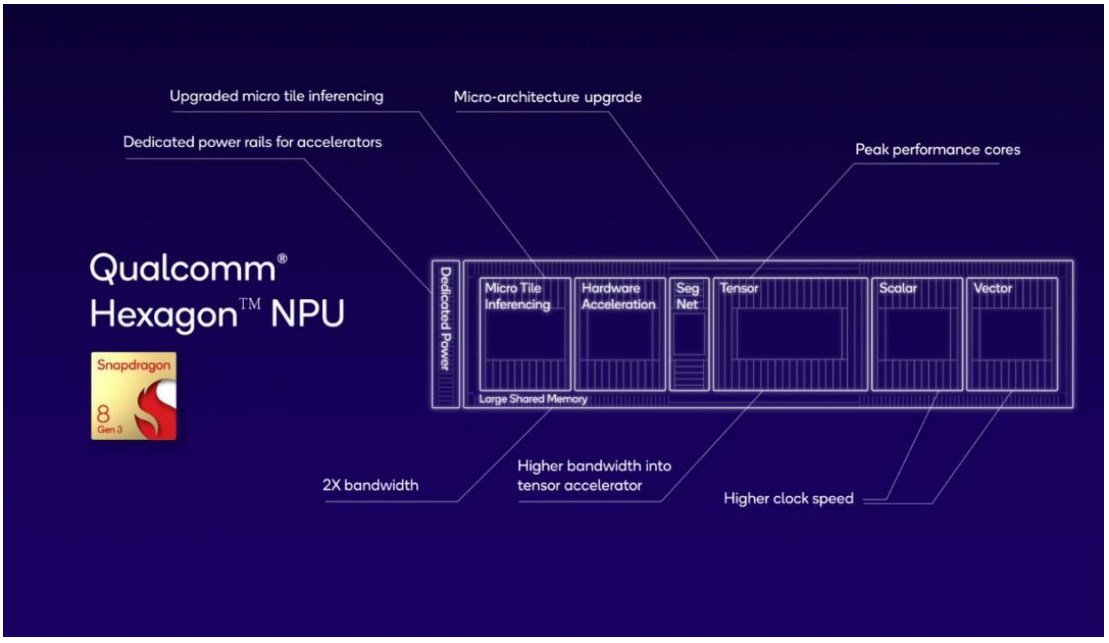
i. Intel:



This is Intel's solution. NPU is integrated alongside GPU and CPU at the same SoC. SoC integration reduces physical lengths and provides efficient and fast computation. When we look into Intel's solution, we see two Neural Engines inside NPU. Both have necessary computing elements. One control block is located to lead two engines efficiently. The control block is also responsible for memory management. Another component is scratchpad SRAM. As we mentioned before, the most problematic part is memory access. To reduce external memory access, Intel's NPU has built-in memory within. Furthermore, Intel used 2 DSP inside Neural Engines to handle vector operations.

ii. Qualcomm :

- **Qualcomm Hexagon:** Qualcomm's idea is different. Generally, NPUs have dedicated blocks for each layer. Unfortunately, this means there are too many intermediate memory accesses. Qualcomm's Micro tile inferencing is based on slicing the Neural Network layers into smaller tiles, executing multiple layers on one tile, and then writing out to memory. This eliminates the intermediate reading and writing to DDR memory.



iii. Apple Neural Engine: Apple Neural Engine is popular because of Apple product's long-lasting battery performance. Apple does not reveal any details. However, there is a block diagram from Apple's patent application in 2017. If we take this as a base, we can guess that ANE has multiple neural engines. But it has only one planar engine. Neural engine circuits perform convolution with one or more kernels to generate the first output. After that, the planar engine circuit processes this data and generates the second output. Planar engine has different modes. In pooling mode, the spatial size of the second input is reduced. In an elementwise mode, the planar engine circuit performs an elementwise operation. In reduction mode, it reduces the rank of a tensor. The rest of the design is straightforward: control units and memories.

FIG. 3

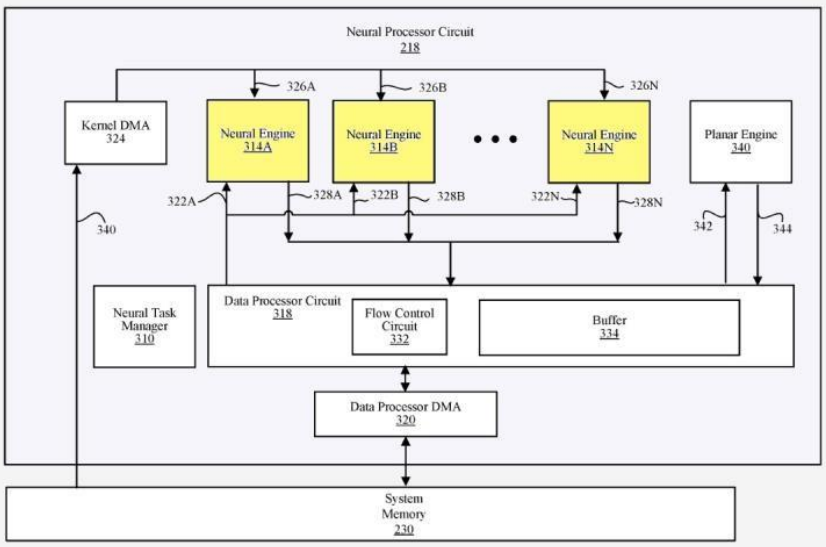


FIG. 6A

Pooling

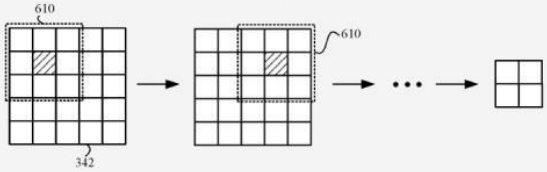


FIG 6B

Elementwise

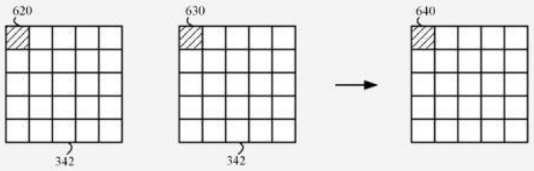
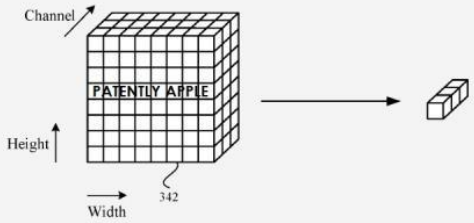


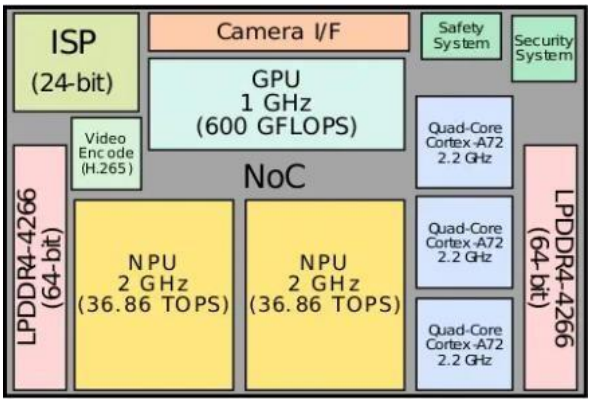
FIG. 6C

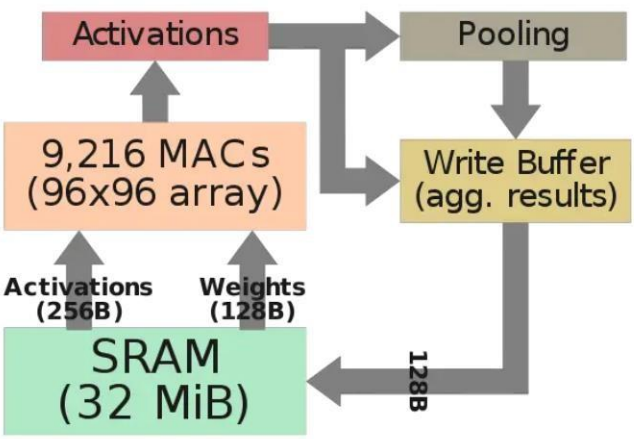
Reduction



iv. Tesla:

- **FSD:** Tesla's hardware to run complete self-driving algorithms for autonomous cars. There are 2 NPUs inside FSD.





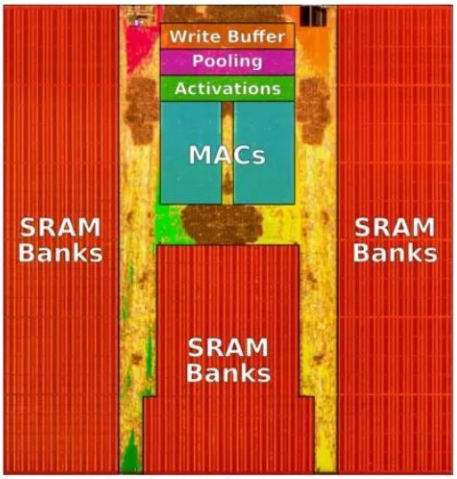


Figure: Internal architecture of NPU

Each custom-designed NPU includes 32 MiB SRAM. Per cycle, 256 activation data and 128 weight data are read from SRAM into a 96x96 MAC array. There are 9,216 MACs total, and 18,432 operations realized every cycle. In FSD chips, 8-bit by 8- bit integer multiplication and 32-bit integer addition are options for further optimizations. Operating at 2 GHz, each NPU achieves 36.86 TOPS, and with two NPUs, the FSD chip reaches 73.7 TOPS. The FSD supports activation functions like ReLU, SiLU, and TanH.

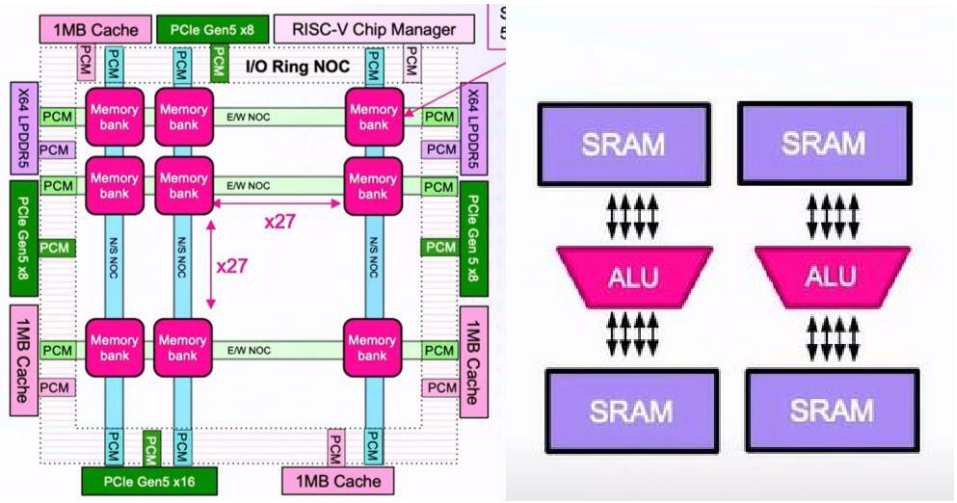
Luckily, we know FSD's instructions set.

Instruction Set:

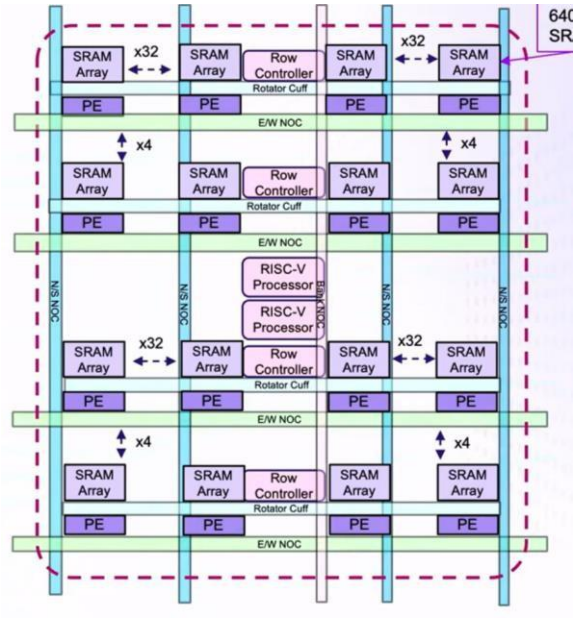
- 2 DMA operations for main memory read/write
- three dot-product instructions (convolution, deconvolution, inner-product)
- Scale (1-input, 1-output)
- Eltwise (2-input, 1-output)
- Stop half-processing

As we can see, there are a few instructions.

v. Untether AI: Their distinctive at-memory IC architecture puts computing elements directly adjacent to memory cells. They assert this leads to ICs with unrivaled compute density and minimal power consumption. The at-memory computation differs from the in-memory computation, where MAC operations are calculated analogously. The idea is to locate computational units between SRAM blocks.



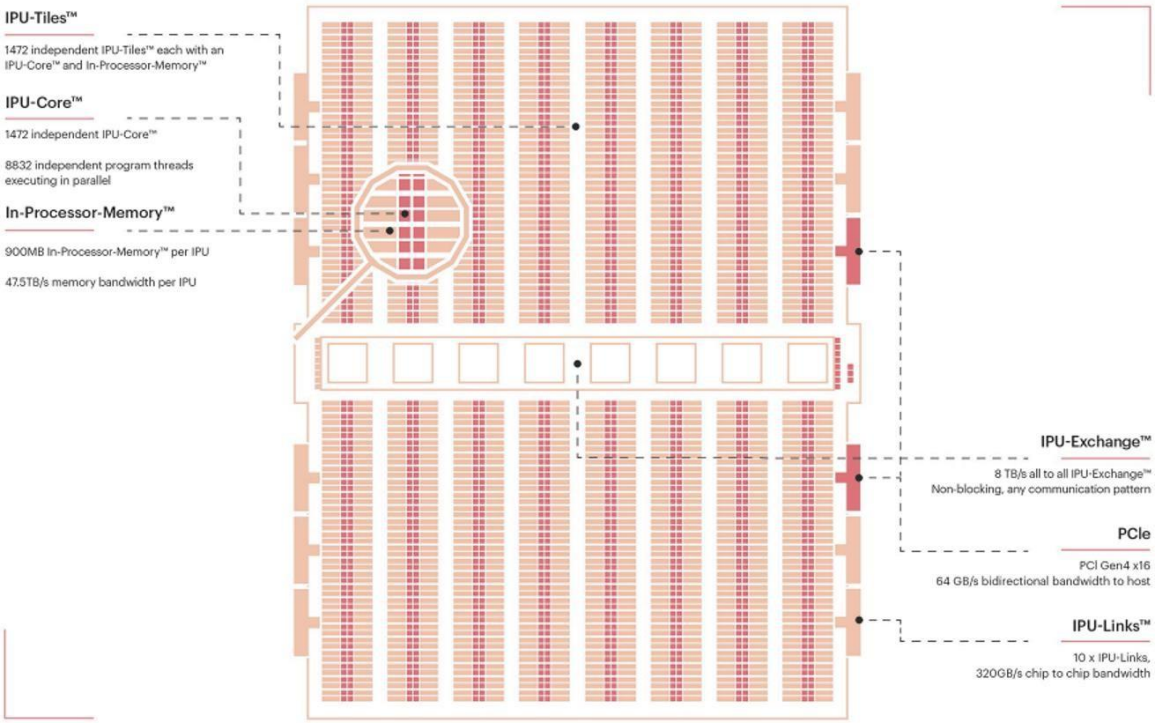
This is the chip itself. There are multiple memory banks and internal networks on the Chip for communication.



From the above figure, we can see the internal architecture of the memory banks. Each bank consists of 4 rows with 4 PE(computational units) and 4 SRAMs. A Row controller controls rows, and each bank includes two memory banks. Also, there is a component that I want to highlight, which is the PE mask. **PE Mask** (Processing Element Mask) is a control mechanism used in parallel computing architectures, particularly in systems with multiple processing elements (PEs), to selectively turn specific PEs on or off during

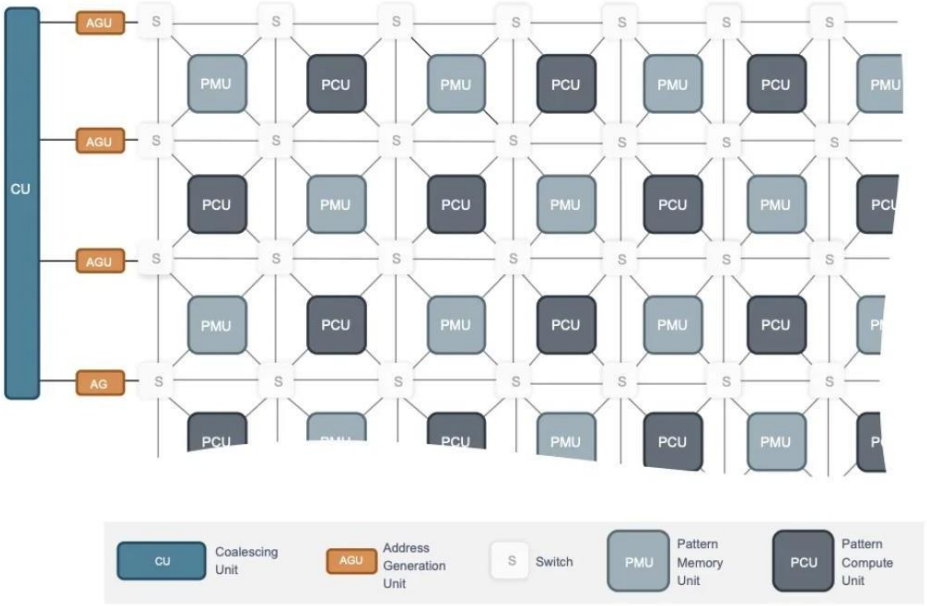
computation. This allows energy efficiency.

vi. Graphcore IPU:



Graphcore's architecture consists of 1472 tiles. Each tile includes a six-thread processor and memory blocks. Chip has the necessary interfaces for communication with external components and tiles. The principle of execution is straightforward. There are three general processing operations of tiles. They operate concurrently. (i) *Compute:* all tiles perform the mathematical computations specified by their assigned threads. (ii) *Sync:* a phase in which we wait for all tiles to finish their execution. (iii) *Exchange:* all the computed output data is written to the exchange memory, and if needed, it will be used as input by (potentially) other tiles in the next Compute-Sync-Exchange phase.

vii. Samba Nova: They use CGRA architecture for AI inference and training. CGRA consists of an array of word-level processing elements connected by on-chip interconnects. They can be reconfigured per cycle based on on-chip configuration memory content. CGRAs bridge the gap between efficient but inflexible domain- specific accelerators and flexible but inefficient general-purpose processors.



CU here is responsible for data management. It sends necessary signals to corresponding units. AGU helps it as an address manager. PMU and PCU are computing and memory units of this architecture. What the switch does is just communication between units. This architecture takes advantage of CGRA. Therefore, they can adapt to different tasks faster, which is advantageous for AI tasks. Also, in some cases, we expect them to be more efficient than FPGA implementations since they are word-level configurable instead of bit-level.

viii. Rain Neuromorphics: Rain Neuromorphics is taking a different approach to the challenge than traditional methods to produce a revolutionary analog trainable circuit resembling the human brain. Unlike digital strategies, which process data using binary (1s and 0s), this analog system will process data using voltages, currents, and resistances. This method is intended to provide data processing that is significantly more energy-efficient. Neural networks in the current digital era are essentially software simulations of the brain, which means that hardware-based synapses and actual neurons do not exist. However, data handling is done differently on this analog chip: weights are recorded as resistances (memristors) between synapses, and matrix multiplication—the foundation of neural networks—is accomplished by stimulating neurons as voltages. The output chip reads the final current at the end of the process. Thus, the internal physics of the semiconductor does the math for us. They want to train robots to learn just like people by completing the training totally on hardware. CEO asserts that their work is creative, but they also acknowledge that it will be tough.

- **Conclusion**

In this blog, I tried to explain NPU briefly. I hope at least you got the idea and understand why people are trying to implement such circuits. We also dived into the market's perspective on this cutting-edge technology. Lastly, we examined different architectures with different ideas. This blog was generated from my notes while searching NPU architectures during my Queen's University of Belfast (CSIT) undergraduate research. I hope it will be helpful for future undergraduate researchers or interns interested in AI's hardware side. I would be happy to hear if you have any recommendations or feedback. You can directly contact me via e-mail (ybaran@ucsd.edu).

- **References**

- https://semiconductor.samsung.com/support/tools-resources/dictionary/the- neural-processing-unit-npu-a-brainy-next-generation-semiconductor/#:~:text=There%20is%20a%20type%20of,just%20like%20the%20human%20brain
- https://en.wikichip.org/wiki
- https://www.qualcomm.com/news/onq/2024/02/what-is-an-npu-and-why-is-it- key-to-unlocking-on-device-generative-ai
- https://medium.com/@adi.fu7/ai-accelerators-part-i-intro-822c2cdb4ca4
- https://www.ibm.com/topics/artificial-intelligence
- https://developer.nvidia.com/discover/artificial-neural-network#:~:text=An%20artificial%20neural%20network%20transforms,input%20into%20the%20next%20layer
- https://www.forbes.com/sites/moorinsights/2024/04/29/at-the-heart-of-the-ai- pc-battle-lies-the-npu/
- https://www.comp.nus.edu.sg/~tulika/CGRA-Survey.pdf
- https://www.tesla.com/en_gb/AI
- https://www.graphcore.ai/bow-processors
- https://www.qualcomm.com/content/dam/qcomm-martech/dm-assets/documents/Unlocking-on-device-generative-AI-with-an-NPU-and-heterogeneous-computing.pdf
- https://www.youtube.com/watch?v=WEymRJb0dsoCt=301s
- https://www.youtube.com/watch?v=J8N9bG5YQ_g