NASDAQ:CEVA Q3 2025 Earnings Call Transcript

Generated on 6/10/2026

Roni Wattelmacher | Director of Product Marketing, Vision and AI Business Unit, SIVA:

Hello everyone and welcome to our webinar, what it really takes to build a future-proof AI architecture. I'm Roni Wattelmacher, Director of Product Marketing for the Vision and AI Business Unit at SIVA. Joining me today is Asaf Ganor, our Director of AI Architecture. We're excited to discuss, show, and teach you how one should build his AI chip architecture for it to be future-proof and why it's important. Before we start, a few logistics. First of all, please fill out the survey that you see at the top of your screen. It's important to us. Secondly, we'll be doing a live Q&A session at the end. If you have questions, please write them in the Q&A box to your right. You can enter them at any time, and we will answer them later. On the left, you can see a resource list with links that you can download, links to interesting web pages, and contact information for Asaf and me if you wish to send us a message. Let's start. Let me quickly walk you through what we'll cover in today's session. We'll start with a quick overview of the AI market landscape and how it's evolving. Then we'll dive into key technology trends driving AI workloads, both at the edge and in the Cloud, and how the industry is shifting. Next, we'll talk about why future-proofing is no longer optional in AI hardware and software design, especially as workloads grow more complex and distributed. Then Asaf will take over and go deeper into how NPU solutions are designed to address these challenges, ensuring scalability across multiple generations of AI workloads. And finally, we'll open it up for your questions. Feel free to drop them in the chat during the session. For those of you not familiar with SIVA, we license silicon and software IP to semiconductor and OEM companies. Our customers ship around 2 billion chips and devices annually using our technologies. Our IPs address three core use cases across multiple end markets, Connect, Sense, and Infilm. Connect will provide IPs for Bluetooth, Wi-Fi, UWB, cellular IoT, 5G modems, from base station to satellites. In the sense, we offer IPs and software for vision, audio, radar, motion, and microphone sensing powered by DSPs and sensing algorithms. In fair, today's webinar focuses on this domain, our AI solutions. We license a scalable family of NPUs and the unified AI SDK to enable efficient inference at the edge. This chart shows how different use cases require different performance, which correlates with the power consumption and area. Milliwatts to many watts, less than one square millimeter to few cores of few square millimeters. As we move right on the chart, power consumption increases, and as we move up, inference capability increases. AI inference happens across a wide range of devices, from tiny wearables consuming milliwatts of power to large multi-core systems in the cloud. Multi-core architectures are needed as we move towards complex AI workloads like automotive, co-pilots, and cloud inference. Computing has evolved in distinct eras, each marked by a dominant platform and a step change in the number of deployed units. We started with mainframes in the 1960s, which had about a million units deployed and moved through many computers, PCs, desktop internet, and the mobile internet. Each era saw 10x to 1,000x growth in the number of connected devices. Today, we're entering the AI era, which is expected to doff all previous computing cycles. Forecasts suggest there will be tens of billions of AI-enabled devices, making this the largest compute cycle ever in terms of unit volume. Just like CPUs powered the PC era and GPUs scaled cloud compute, NPUs are the compute engines of the AI era. NPUs are optimized for AI inference and are designed to handle computational needs for AI workloads efficiently. What's unique about the AI era is that compute is not just in the cloud anymore. It's moving to the edge, powering devices like wearables, autonomous vehicles, smart cameras, and IoT nodes. Cloud still plays a role for training and large-scale inference, but real-time AI happens at the edge, where NPUs deliver the required performance and energy efficiency. This growth isn't incremental, it's exponential. Let's talk briefly about how cloud AI infrastructure is evolving to support the demand of the AI era. Cloud providers are the home for training massive foundation models and running large-scale inference that serves billions of users worldwide. Developers today don't have to manage the complexity of this hardware. All of this compute power is abstracted into AI services, fully managed, scalable, and easy to deploy. We're seeing a shift where models and hardware are being designed together.

Transformers, for example, are being optimized to run efficiently on new silicon, driving better throughput and lower latency. Energy consumption has become a critical concern for AI farms. Cloud providers are focusing heavily on performance per watt, creating chips that do more while consuming less energy. It's not just NVIDIA anymore. Cloud providers are building purpose-built AI accelerators like Google's TPU, AWS, Inferentia, and Tranium, and Microsoft Maya chips, each optimized for the specific AI workloads. Finally, it's important to understand that the cloud and the edge complement each other. Heavy compute workloads like model training and large batch inference happen in the cloud, but real-time latency-sensitive tasks run directly on the edge where fast response is critical. Industry leaders are now expanding from cloud-first approaches to edge-native intelligence. recognizing the need for low latency on-device AI across robotics, IoT, industrial systems, and consumer products. These examples you see here on the slide show how the entire industry is preparing for an edge-first AI future, where devices themselves are intelligent, not just connected to a cloud that is. More optimized and efficient models are accelerating edge deployment, smaller size, and lower memory use. Generative AI in edge devices offers a wide range of benefits. Lower energy consumption, reduced latency, enhanced privacy, and cost efficiency. Use cases include chatbots using Edge LLMs and SLMs, for example, for appliances, medical analysis based on text, text generation by IoT devices. Emergent Edge devices will support complex GenAI tasks with on-device learning, enabling personalized private and cloud-independent AI. Dedicated edge NPU hardware is required for boosting real-time performance and energy efficiency using advancements like compression, distillation, transformer support, and low-bit networks. We've all heard of LLMs, Large Language Models, which brought human-like text generation to mainstream AI. Then came LVMs, large vision models, which applied transformer architectures to images and video. Now we're entering the era of LMMs, large multimodal models, which process and understand text, vision, voice, and sensor data together, creating much more capable AI systems. Originally, these multimodal models were designed for the cloud, but now we've seen models optimized to run efficiently on mobile devices, wearables, and industrial edge nodes. This is possible thanks to the breakthroughs like low-bit quantization in, for example, precision and optimized transformer architectures. As shown here, the multimodal AI model sits at the center, integrating and processing text, images, audio, and video, enabling devices to better understand the world and act as it should in the world around it. Now, let's talk about agentic AI, a key evolution in edge intelligence. These are AI systems that don't just recognize patterns. They autonomously plan and execute tasks without human intervention. Agentic AI is built on three core capabilities. Autonomy, the ability to operate independently and make its own decisions. Adaptability, adjust its action based on changing environments or goals. And decision-making, selects the best action to achieve a given objective. Let's see some use cases. Smart personal assistants. Think about wearables or home hubs that don't just respond to voice commands, but proactively assist based on your context, adjusting the lightning, suggesting action, or providing health insights in real time. Predictive maintenance. In industrial settings, machines use agentic AI to monitor their own performance, detect anomalies, and even schedule maintenance before a breakdown happens, reducing downtime and saving costs. These use cases run directly on edge devices where real-time decision-making is critical and cloud latency isn't acceptable. So far, we've talked a lot about AI models and compute platforms, but it's also important to recognize how AI is being integrated into the physical world. Physical AI refers to a system where AI doesn't just process data, it controls real-world devices. robots, drones, autonomous vehicles, and smart appliances. One of the most advanced examples today is Waymo's fully autonomous vehicles operating on public roads with no human driver in the front seat. On the right, you can see market data from San Francisco showing how Waymo's market share in ride-hailing has been growing steadily. In fact, Waymo's fully autonomous fleet has now suppressed lift in gross booking share and continues to close the gap with Uber. This is real-world proof that AI-powered physical systems are commercially viable and scaling up in urban environments. Physical AI is not a future concept. It's happening now. From autonomous delivery robots to self-driving cars, AI is making decisions and interacting with the real world in real time. Okay, let's talk about future-proofing. When we talk about future-proofing AI architectures, we're really talking about preparing for change. Most of what I just showed you did not exist just a while back. The AI and machine learning landscape evolves extremely fast. Models that are state of the art today could be obsolete in 12 to 18 months. New hardware accelerators, better algorithms, and entirely new use cases emerge constantly. AI workloads are evolving fast. Models today are bigger, more complex, and optimized differently than they were just a few years ago. If your NPU

only supports a narrow set of operations or model types, you'll be stuck when the next generation of models comes out. Future proofing is critical. Now let's see how a typical system flow looks like. This common block diagram represents how different hardware components work together to run vision and AI workloads efficiently. It starts with the sensors capturing raw data, sensors like cameras, lidars, and so on. Some systems will include an image signal processor to pre-process that data. Then the data flows into the NPU and vision processor, where the main AI inferencing happens. The NPU engine accelerates the AI workloads. The NPU controller manages the processing flow. L1 and L2 memories are tightly integrated to provide fast access to data and instructions. Outside the NPU, the CPU, usually ARM or RISC-V are used, runs the system software decision-making logic, but also can run simpler processing tasks. The GPU, if present, can be used for graphics or some parallel compute workloads. The SRAM provides additional fast memory where needed, The system also interacts with peripherals, and for larger data, accesses external DDR memory. This is the high-level view. Now I'll hand it over to Asaf, who will take you deeper into how the AI and vision flows work and how to prepare for the future.

Asaf Ganor | Director of AI Architecture, SIVA:

Thanks, Roni. Hello, everyone. Great to be with you today. Let's take a step further into what it really takes to design AI architectures that stand the test of time. Building a future-proof AI architecture comes down to three foundational pillars. Scalability, which is the ability to dynamically scale compute resources to match a wide range of workloads. This isn't just within a single product, but across your entire product portfolio, from edge devices to the cloud. Extendability, the ability to seamlessly add support for new workloads, operators, and system flows, while keeping both software and hardware overhead minimal. And finally, sustained efficiency, ensuring maximum computerization, even as workloads diversify and evolve over time. These three pillars, scalability, extendability, and efficiency, are what enable resilient, long-living AI systems. Let's take a deeper look into the first pillar. Scalability is the ability to efficiently scale computes, memory, and system resources across workloads from ultra-low power wearables to cloud-scale data centers without compromising performance, power, or area. But this is easier said than done. Here are some of the key scalability challenges architects face. Resource imbalance occurs when an architecture optimized for high compute workloads is first forced to run lightweight tasks. In these scenarios, large computer arrays remain underutilized, and the overhead from logic and memory leads to unnecessary power consumption, making the solution inefficient for these tasks. Efficient workload distribution is another core challenge. The workload is initially optimized for a specific set of system resources. But as new workloads are introduced, this must share resources with legacy tasks. Over time, this puts pressure on both hardware and software tool chains to maintain efficiency and performance across use cases. Memory systems add their own layer of complexity. A truly scalable AR architecture must be able to handle a wide range of memory configurations, from remote high latency memory to ultra bandwidth systems like HBM or 3D DDR. It must dynamically balance memory access and processing cycles, maximizing data reuse and minimizing memory latency exposure. Next, physical design complexity. As we scale up compute capabilities, physical challenges intensify. Efficient floor planning, architecture wear implementation, and advanced physical design tools are critical to meet frequency and power targets while maintaining scalability. And finally, customization. One size does not fit all. Different applications may prioritize different KPIs, performance, power, area, or a balance of the three. A scalable architecture allows you to tune the resolution of processing elements, enable efficient workload-specific optimization across your product portfolio. Let's walk through a high-level block diagram of SIVA's Nucor-M AI architecture as an example. This diagram shows the main units of the IP and how each one addresses key scalability challenges we discussed earlier. At the top, we have the NPM engine, which serves as the main computer. Let's break down its components. The tensor processing units. This is the core MAC array optimized for metrics-heavy operations like conclusions and metrics multiplication across various data types. The NPM core's job is to keep the MAC read as busy as possible to maintain high utilization throughout the inference process. Vector processing unit, or VPU, is a flexible DSP core tuned for programmable workloads and integrated into the hardware pipeline. It's ideal for supporting new parameter types and specialized control logic without stalling performance. Sparsity acceleration. This unit takes

advantage of the inherent sparsity in many neural networks. By skipping zeros, it reduces memory bandwidth demands, saves power, and boosts performance. Streaming nonlinear and quantization acceleration. Responsible of nonlinear operations like activations, such as railing, pooling, normalization, quantization, and even softmax. It processes data in a streaming pipeline, avoiding back pressure to the MAC grid and sustaining throughput. L1 memory fabric is a dedicated ultra-high bandwidth local memory that feeds data into each engine with minimum latency and maximum efficiency. Engine scheduler, a hardware level controller that orchestrates data flow between units within the engine. It ensures smooth data transition and prevents stalls across the pipeline. The number of NPM engines in the core is configurable. You can tailor the compute density to your application's needs, from low power to high performance use cases. Now let's look at the L2 subsystem, which manages data movement between external memory and the compute engines. The L2 fabric acts as a central buffer between the memory, like DDR, and the engines. The system DMAs handle the transfer of data from the remote memory to L2. These are optimized to offload complexity from the software stack, making 2-chain linear and more efficient. Weight decompression. Compressed weights are stored in memory and decompressed on the fly to reduce bandwidth and memory footprint. Data reshaping converts data layouts from DDR to L2, from L2 to L1 in each engine, and vice versa, to match the expected format and dimensions of each compute unit. This minimizes overhead and maximizes throughput. System scheduler manages coordination between engines. It enables dynamic workload sharing, task splitting, and buffer synchronization so engines can collaborate efficiently. Lastly, the controller core plays a crucial role. It interfaces with the host CPU, orchestrates application-level tasks, and handles pre- and post-processing when needed. Let's take a closer look at the configurable building blocks of the new Pro-M core. One of the key strengths of this architecture is its scalability through configuration, enabling precise optimization for performance, power, and area. First, the number of NPM engines is configurable to match your workload scale. Whether you're designing for edge AI or high-performance compute, the architecture scales to fit. but flexibility does not stop at the top level many of the compute resources within each engine are also highly customizable you can adjust the number of mugs per cycle the supported data types are selectable from a wide arrayed range both in the integer and floating points and realms in every product variant only the needed data types are implemented helping maintain maximum area and power efficiency and you can tailor the mac array topology itself adjusting its structure to match specific goals for example In an area-optimized configuration, int for max can be reused in a laddered setup to support int8. Meanwhile, a power-optimized implementation might dedicate separate mark hardware for each data type to minimize switching and improve efficiency. The same scalability applies to the nonlinear quantization accelerator itself, which may be provisioned to match your expected network workload. And finally, the software toolchain enables accurate PPA modeling, help you analyze trade-offs early and make informed design decisions based on your application KPIs. In uProM, flow control is managed through a dedicated hardware sequencing mechanism. This means that once tasks are set up, the hardware handles much of the execution autonomously, reducing the need for constant software oversight. The benefit of this approach is that it helps maintain consistent utilization across the compute engines, even when system configurations vary, whether you're working with a minimal setup or a highly parallel one. On the software side, the design supports VMA virtualization and abstraction. Rather than having to manually program buffer sizes or adapt drivers for every product variant, the architecture supports a consistent driver interface decoupled from hardware-specific parameters. This approach can significantly reduce integration efforts and code duplication when building multiple products on the same architecture, especially when working with different memory layouts, bandwidth constraints, or compute profiles. Scalability goes beyond just compute. It extends into memory architecture and system-level scaling. First, memory resources are tunable to match the system class. Whether you're targeting a microcontroller with PS RAM or a high-end platform with HBM or 3D DDR, the architecture supports the right bandwidth and size trade-offs to fit your performance and power envelope. Second, at the system level, NewProM is built to scale across multiple cores and even across multiple clusters of cores. That means you can grow the architecture to handle larger models or higher throughput without re-architecting your design from scratch. Together, these features ensure that scalability isn't just about adding compute. It's about growing all system components in harmony. Let's now look at how efficiencies are addressed across different layers of NewProM architecture. from hardware to software and from block level design to system behavior. First, at the hardware level, the design incorporates several well-established techniques for power efficiency.

Fine-grained clock gating helps eliminate unnecessary switching in inactive models. Power domain shutdowns allows different blocks to be completely powered down when not in use. And memory retention supports fast wake up without data loss during sleep transitions. These techniques together support dynamic power management, which is essential for workloads that fluctuate in activity level. At runtime, the system provides an efficient execution environment, enabling smart allocation of compute and memory resources depending on task needs. This is especially important when running multiple models or switching between workloads with very different compute profiles. From a physical design perspective, the macro level layout of NewPro-M is engineered to be scalable and portable. That means it can adapt across different silicon variants such as power optimized or area constraint design with minimal layout effort. And finally, the compiler plays a critical role in maintaining efficiency. It adapts the tiling strategy, how data and operations are split based on available compute and memory resources. This allows for better data reuse, lower transfer overhead, and ultimately higher throughput per watt. Together, this element supports a design that can scale effectively, both across SQL configurations and across real-world workloads. When evaluating AI architectures, it's easy to get caught up in the wrong metrics, numbers that look impressive but don't necessarily reflect real-world efficiency or scalability. Let's start with a few commonly used metrics that can be misleading. Power in watts. A low power number might seem desirable, but it doesn't mean the design is efficient. It could simply indicate underutilization. In other words, the system isn't doing much work to begin with. Area in millimeter square. Less area often seems desirable. It suggests lower costs and higher integration potential. But if that smaller area isn't used efficiently, you may end up needing more silicon overall to meet performance targets. In the end, this can actually increase cost and power, defeating the original intent. And inference per second, IPS. This seems like a solid performance metric, but it lacks normalization. IPS can artificially be increased by scaling up hardware, so it doesn't reflect architectural quality or efficiency on its own. Instead, we should focus on normalized efficiency metrics, ones that tell us how much useful work the system is doing relative to cost or resource usage. Two key metrics are IPS per millimeter square. This is your area efficiency. It tells you how much inference throughput you get per unit of silicon, a critical metric when optimizing for cost integration or DICON strain, and IPS per watt. This is your energy efficiency. It tells you how much useful work you're doing for every unit of power consumed, which is key in both battery power devices and thermal limited applications. On the right, we see how these two axes combine into a performance map. The top right corner, high RPS per watt and high RPS per millimeter square represent the scalable sweet spot. That's where you want to be. Other quadrants reveal trade-offs. For instance, dense but slow designs might be efficient in area but drain too much power. And energy hobs might perform well but consume more power than they're worth in dive area. Evaluating AR architecture based on these normalized metrics give a far more realistic picture of how well a system will scale, especially when moving from proof of concept to real deployment. So when someone claims high RPS or low power, the next question should be, for what? Efficiency only becomes meaningful when we anchor it to area and energy context. Let's now shift to the second pillar of future-proof AI architecture, extendability. In simple terms, extendability means the ability to add new operators and system flows to your existing AR hardware and software without having to redesign the architecture every time something changes. This has become increasingly important as the AI landscape evolves so quickly, and it brings a few challenges worth highlighting. First, we have rapid model evolution. New neural networks with new layer ops or patterns are emerging all the time. Architectures that can't adapt will fall behind quickly. Second, many accelerators rely on fixed operator sets. They work well for today's model, but struggle when something new appears. This limits flexibility and makes it harder to stay up to date. Third, the fallback strategy is offering to the CPU, but that's not sustainable. Offloading unsupported ops to the CPU consumes more power and slows down inference, especially at scale. Fourth, it's not just about operators anymore. We're seeing full system flow changes with new quantization strategies, memory layouts, and normalization schemes that affect the entire system pipeline. A rigid system can't handle this gracefully. And lastly, the toolchain becomes critical. Hardware extendability is only useful if your SDK and software tools let you actually use it. Supporting new flows in software is what makes real-world deployment flexible and maintainable. So, while performance and efficiency often steal the spotlight, extendability is what keeps an AI architecture relevant over time, especially in a fast-moving field like this. Let's come back to our new Pro-M example. One of the key enablers for architectural extendability is having a flexible non-linear pipeline. Instead of hard coding activations, NewPro-M uses software-controlled non-linear functions, making

it future-proof as new activation type emerges in models. This flexibility goes beyond just math. It enables custom system flows where operations like softmax, normalization, or dynamic quantization can run in tailored data streams, independent of the main path. In some use cases, this pipeline can operate in a standalone mode, bypassing the L1 memory completely, saving latency and energy for ops like pooling. Finally, dynamic tensor shaping lets the system adjust data on the fly, so even if the model changes layout or shape mid-inference, the hardware can handle it without stalling. A flexible pipeline needs tight orchestration, and that's where hardware-controlled sequencing steps in. Instead of relying on software to micromanage timing, NewProM Engine's scheduler enforces timing in hardware while still following software's high-level policies. This approach also allows for modular integration of external compute engines. Each new unit connects through a standard data control handshake, keeping the system clean and scalable. And with centralized flow control, you avoid data mismatches or deadlocks, since each stage in the data path is synchronized. This level of control keeps performance stable, even as workloads get more dynamic or complex. To support unknown or emerging operators, NewPro-M integrates the vector processing units, or VPU, directly into the pipeline. You can think of the VPU as a configurable DSP, designed to handle a wide range of operator types. It's not locked to specific logic. Instead, it supports everything from basic element-wise operations to complex model-specific custom kernels. Importantly, these kernel extensions are not isolated. They are supported end-to-end by the software toolchain, and they plug into the existing system flow without requiring structural changes or manual workarounds. This makes the VPU a powerful tool for efficient kernel mapping. Instead of falling back to a CPU or relying on its external cores, future operators can run directly on the VPU with minimal overhead and high throughput. And because the VPU is tightly coupled with the main pipeline scheduler, data can move fluently between the VPU and tensor data path, eliminating idle time and keeping the pipeline efficient even as new workloads are introduced. Let's now look at the third pillar of the future-proof architecture, efficiency. Efficiency is becoming more complex to maintain as models evolve, especially with a shift towards low-precision computing. The industry is clearly moving towards smaller data types, like INT4 and also FP4 and FP8, because they offer compelling benefits. They reduce power consumption, lower memory footprint, and enable smaller MAC units, leading to more compact circle designs. But with those gains come non-trivial trade-offs. As we shrink data precision, we also increase the risk of degraded accuracy. The math becomes more sensitive, and models can lose fidelity if precision is cut too aggressively. To counter that, systems must support advanced quantization techniques and sometimes even mixed precision execution, balancing small formats with higher precision fallbacks where needed. Another efficiency strategy is sparsity, skipping over zero values or redundant computations. While this can reduce bandwidth and improve performance, it's not always a drop-in solution. Many models need retraining to regain accuracy after a sparsity is introduced. And since sparsity techniques are often model or platform specific, they can reduce the portability and reuse of pre-trained models across different systems. In short, efficiency today isn't just about smaller formats. It's about managing the trade-offs they introduce and designing systems that can adapt intelligently to them. As precision shrinks in modern AI workloads, Maintaining accuracy becomes a key concern, especially with formats like INT4 or FP4. Traditional quantization typically assigns one scale factor per tensor or channel. That's simple, but it often results in degraded accuracy, especially for low precision formats where fine-grained variations get lost. Dynamic group quantization takes a more nuanced approach. Instead of one scale for the entire tensor, it applies different scale to smaller element groups, say a few rows or columns at a time. This boosts local precision without needing more bits. The challenge, of course, is to implement this without sacrificing performance on the weights and the dynamic data generated within the network itself, which is exactly what we've set out to solve. Let's illustrate the problem with traditional quantization. In this simplified metrics example, both weights and activations are quantized using a single scale parametrics. That means the entire range of values is compressed using a common scale, regardless of local variation. This course granularity may work fine for some data, but in real world scenarios, it often leads to poor value representation, especially at low bit width. Several values differences get lost and the model may suffer from accuracy degradation as a result. Dynamic group quantization solves this by applying multiple scales across smaller element groups. For example, rows or columns can be grouped and scaled individually. You can see that in the coloring. Each group has its own scaling and zero point. This makes the quantization more adaptive to local variation, preserving accuracy where it matters most. What's especially important is that this more detailed scaling doesn't compromise

efficiency. The compute pipelines remain streamlines, even when activations are quantized dynamically at runtime from one layer to the next. The equation at the bottom just expresses the composite scaling and zero-point logic needed to reconstruct accurate results from quantized values. To show the impact of dynamic quantization in practice, here's a quick benchmark using LAMA-27B. The baseline model is FP16, and we compare it to dynamic quantization configuration using 4x8 bit quantization, 4 bits for the weight, 8 for activations. The results, a 4x performance gain and 4x smaller memory footprint for weights, while keeping the accuracy degradation to just 2%. This is a great example of how smarter quantization strategies like dynamic quantization can unlock meaningful efficiency gain without paying a heavy price in model quality. One of the most effective ways to boost efficiency in AI workloads is by leveraging sparsity, specifically the ability to skip over zeros in data and weights during inference. New Pro-M enables this through support of unstructured sparsity, which means it can dynamically detect and skip randomly scattered zeros without requiring any special pruning patterns. This allows the architecture to take advantage of the native sparsity present in many neural networks, achieving real performance gains without needing to retrain the model. Of course, structured and semi-structured sparsity is also supported, and retraining can be used when there is a need to guarantee a specific acceleration factor, even if that comes with a small accuracy trade-off. For example, many transformer models can be proven to run 50% sparsity without degrading accuracy. That unlocks several key advantages to performance, DDR bandwidth reduction, and power consumption. Let's wrap things up by revisiting what it really takes to build a future-proof AI architecture. AI is evolving fast, so the system we build needs to evolve just as quickly. And that comes down to three fundamental principles. First, scalability. Your architecture needs to stretch across a full performance spectrum, from the smallest edge devices to the largest cloud deployment. That means modular building blocks, flexible configurations, and the ability to serve a wide range of applications without redesign. Second, extendability. AI workloads don't stand still. Operators change. Models evolve. New techniques emerge. A robust architecture must support that evolution, allowing new features and flows to plug in over time without ripping out what already works. And finally, sustained efficiency. It's not just about having high peak performance on paper. It's about keeping the hardware busy even when the workloads are messy. That includes handling low precision data, exporting sparsity, and minimizing memory bottlenecks. Together, these three pillars, scalability, extendability, and efficiency, are what makes an architecture resilient, adaptable, and ready for what's next.

Roni Wattelmacher | Director of Product Marketing, Vision and AI Business Unit, SIVA:

Okay, thanks Asaf for the deep dive into the NPU design challenges and solutions. I'd like to emphasize that SIVA's award-winning NewPRO architecture brings together the key pillars of a future-proof AI solution as was just explained. Scalability across edge-to-cloud deployments, extensibility to adapt to evolving AI models and workloads, sustained efficiency through advanced quantization, sparsity, and power management. New Pro-Am is available today, powering a wide range of future-ready AI products. We hope you've enjoyed this. Let's move on to the live Q&A section and take your questions.

Moderator | Webinar Moderator:

Thank you, Roni and Asaf.

Moderator | Webinar Moderator:

We'll now move to the live Q&A session. You will find an icon with a question mark at the bottom of your screen where you can type in questions. Also note the survey at the bottom of your screen. If you can answer that, that would be great. And we'll start to answer some of the questions which you have fed here.

Roni Wattelmacher | Director of Product Marketing, Vision and AI Business Unit, SIVA:

so the first question is for you ronnie sure how can i pick the right mpu a member which is right for the models i plan to run well thank you and good question so siva provides um sdk a few tools that will allow you also with that and we have the architecture planner It helps you to analyze the model's compute, memory, and bandwidth, and according to the requirements, gives you the results. It allows you to simulate different NPU configuration and find the optimal match to those requirements, the performance, power, and area. After you do that, and you select the suitable architecture, you can use the AET tool, the accuracy estimation tool, to verify and to evaluate the impact of quantization and other hardware-specific optimization on the model accuracy. Together these tools help you to confidently choose the right SIVA MPU for your application, but you don't do it alone. SIVA's support personnel are by your side helping you during the entire process, so they guide you and take you all the way until you find the optimal solution.

Moderator | Webinar Moderator:

Thanks, Roni. Another question here is about Probably for you, Asaf, what is the estimated power saving when using the dynamic sparsity and quantization you spoke about in the session today? Sure, I'll take this one.

Asaf Ganor | Director of AI Architecture, SIVA:

So the power saving is very model-specific and also data-specific. Features like the dynamic sparsity give you more the more sparseness you have in the network itself, or the more aggressive you want to be with the tuning versus the accuracy trade-offs. Regarding the data types and quantization, again, advanced techniques allow you to go lower in data type without sacrificing the accuracy. The more aggressive you are and the more the network is comfortable with quantization, you can get more benefits. You don't have to guess. The SIVAS toolchain allows you for each network to input the number, the types of data types and quantization strengths you want to apply. You have accuracy estimation tools that you see results of your pruning, so you can know exactly what you get before you get running. Maybe adding a few sentences, what are the dominant power factors from these features, right? So if we talk about sparsity and also quantization, and the things that you say for me are DDR power, for one. DDR bandwidth is one of the more significant factors in the power of the entire inference. When you get less data stored and transferred from the DDR to the IP itself, you pay less in power and you have a lower memory footprint and also power. And of course, when you get acceleration, when you get a cycle reduction, you can benefit from the dynamic power. Also allows you to go with aggressive power management techniques like race to halt, meaning you perform your inference and you shut down your IP until you need it again. Or you can go with a DVFS approach, meaning if you can go with a lower frequency, you can sometimes lower the voltage and get the benefit of the power reduction as a result of that.

Moderator | Webinar Moderator:

Maybe there is a continuation question here from the audience about how does dynamic grouping work? Maybe you can continue with that one. Sure.

Asaf Ganor | Director of AI Architecture, SIVA:

So I tried to touch a little bit about the dynamic group quantization. I can talk a little bit about the algorithm. So what you want to achieve, as I said, you don't want to use the same scale for the entire population of elements. Let's say that you have data and you multiply it by the weights. If you want to do quantization, you need to represent the entire population with a small number of bits, four bits, eight bits, and this is inaccurate.

You cannot fit all of the numbers to this small number of bits. So what you do, while you run during inference, you can group up the data while you generate it. Every set number of elements, you attach a scale, a more accurate scale to them. And then while you do the computation of the next layer, you do p-quantization, And you gain back the accuracy so that you get effective data type, which is higher than what you pay for in memory bandwidth and power consumption. Of course, there is a big challenge of doing so. How do you do it without compromising your performance? And how do you do it without compromising your accuracy? And this is exactly the expertise of SIVA, and this is what we know how to do well. So maybe it's a few more seconds. You can do it separate. The data and the weights don't need to be the same group size. If you want to keep, let's say, your weights in 4 bits and the data, you can entertain 8 bits. Then you can have a bigger group of them that go back to a scale or a channel of a tensor in the activations. And you can be more aggressive, let's say, 32, a group size of 32 elements in the weights. So these are separate decisions that you're And of course, you can see what accuracy you get as a result using SIVA tools.

Moderator | Webinar Moderator:

Thank you, Asaf. There was a general question about whether the slides will be available or the video itself. The slides and the video from this webinar will be available on demand right after this session. So you can view it or share it with colleagues at any time. Moving on to the next question, probably for you, Roni. How does SIVAD SDK ensure extensibility without requiring hardware redesigns as new operators emerge, as was discussed here about future Wolfram?

Roni Wattelmacher | Director of Product Marketing, Vision and AI Business Unit, SIVA:

So the SDK is built with the operator level abstraction and support software control, nonlinear pipelines. This enables new operators and flows to be integrated directly through the software updates. And we do it with what we call our tool SIVA Invite. And so this is how we do that.

Moderator | Webinar Moderator:

Thanks, Roloni. Another question for you, Asaf. How does Siva help customers balance performance, power, cost when optimizing for either edge or cloud deployment?

Asaf Ganor | Director of AI Architecture, SIVA:

Either edge or cloud. So I take this question, tell me if this is what you meant, guys, but I take this question as a question about how to select the right topology, hardware topology, versus what I talked before, which is how to choose the right configuration in the IP that I have in my product, right? So I will answer that. So when a customer comes to decide which configurations he wants, which KPIs he wants to get in his application, Then because the architecture is super scalable and very customizable, we start by getting to know the KPIs, the main networks, the main use cases the customer wants to get. And we can choose many different nodes. We can analyze it and show trade-offs to the customer, depending on what he cares about most, about the area, about power consumption, about the performance, raw throughput. And there are a lot of different scalability options and extendibility options I discussed before regarding the MAC accounts, even the MAC topology, the memory size, bandwidth, all according to the use case. So we can fit everything according to the specific use case. I'll give you an example. If the customer has a very aggressive power KPI, right, and he doesn't care about area, just a playground example to emphasize, we can even offer a higher MAC account, which costs more KPI area, But it allows you to go lower in frequency, and then you can run at a lower voltage and frequency point. So it all can be analyzed according to the use cases, and we can offer a very specific configuration to the customer.

Moderator | Webinar Moderator:

Thanks, Asaf. Another question. One is probably for you. What does the support lifecycle look like for Ziva NPU customers with AI workloads, models change every few years?

Roni Wattelmacher | Director of Product Marketing, Vision and AI Business Unit, SIVA:

So SIVA provides multi-year support with a roadmap that tracks AI model evolution, SDK updates, introduce new operator libraries, silicon partners receive long-term architectural enhancement, everything to ensure customers can evolve their products over time. So, yes.

Moderator | Webinar Moderator:

Okay. Maybe another question related a little bit for you, Roni. So the customers that already use legacy SIVA AI IPs, what is the migration path to the new program discussed here?

Roni Wattelmacher | Director of Product Marketing, Vision and AI Business Unit, SIVA:

Yes, so C20 ProM was designed with backward compatibility and software usability in mind. The SDK allows graph recompilation with minimal changes, and existing NPUs and also DSP integrations can be upgraded through standard IP replacement flows. So it is supported.

Moderator | Webinar Moderator:

Thank you. I see there are a few more questions here on the window. We will not answer all of them now, but we will make sure to get back to you directly, to anyone who is typing the questions here. We'll follow up. As said, the slides can be downloaded after this session on demand. The video can be viewed again. Thank you, everyone, for joining. and see you at one of our next webinars thank you