SDS PODCAST EPISODE 1028: THE CHIP BUILT FOR AGENTIC AI INFERENCE, WITH SAMBANOVA'S ANTON MCGONNELL

Jon Krohn:	00:01	GPUs are today, of course, the workhorse of AI inference in production, but they aren't actually optimized for that. Today's guest built a new chip that is pushing the frontier of real-time AI speed and bandwidth. Welcome to another episode of the Super Data Science Podcast. I'm your host, Jon Krohn. My guest today is Anton McGonnell, VP of Product at SambaNova, a Bay Area company that has raised over $2 billion to develop a chip optimized for AI inference. In this episode, Anton explains why agentic AI has changed the shape of inference workloads and how SambaNova's chip sidesteps the memory bottleneck that slows GPUs when generating output tokens, making SambaNova the most profitable way to run AI hardware. Enjoy this informative episode. Anton, welcome to the Super Data Science podcast. Thank you for joining us today. Where are you calling in from?
Anton McG.:	00:59	Thank you, Jon. I'm calling in from our office in San Jose.
Jon Krohn:	01:03	And our is SambaNova, right? Do you want to tell us a bit about what SambaNova does?
Anton McG.:	01:07	Yeah. So SambaNova is an AI chip and systems company. We've been around for nine years now. It was founded out of Stanford University, many years of research there, and the exploration of better approaches to solving data flow-based computational problems of which AI has become the premier one of today.
Jon Krohn:	01:31	What else even is there?
Anton McG.:	01:33	Yeah, there used to be other things, believe it or not.
Jon Krohn:	01:36	All right. Well, so one of the key things that SambaNova seems to be solving is dealing with inference time compute for AI workloads. So let's provide a bit of context to the audience on what that means. So I suspect our core audience is very familiar with the idea that when you are creating your own large language model, your own AI

model, your own foundation model, whatever it does, you do lots of training cycles. So you can have chips that are specialized to training, or there are also chips that are made to do both to handle training workloads as well as inference workloads. And those inference workloads are after you have your model trained up, you want to be using the model in real time. So some input goes in, some output comes out, the model weights don't change, you just get some kind of AI model result.

02:28 And so probably most of our audience already do that, but just to get that in their minds. What people might not realize is that the inference market is changing fast and particularly in the past 12 months. Why is that, Anton?

Anton McG.: 02:41 Yeah, I think the big catalyst has been agentic AI. Obviously for the past three or four years now, there's been a huge uptick in AI workloads. Slowly they've transitioned from not just training, but inferences, people start to use AI for useful stuff. What's really happened in the past, I would say nine months in particular is the usefulness of these agentic systems, which is really just using AI to automate work or automate tasks or automate discoveries and research. The usefulness of this has accelerated to such a degree that we cannot possibly have enough capacity because we've got so many problems to solve. And as a result of all of these problems to solve, we need more inference to be able to help us solve these problems. And the interesting thing about agentic AI relative to the workloads that came before it is that it is a different computational profile.

03:46 It's got larger inputs, much larger inputs, but a high degree of reusability of those inputs. So even though there's lots of input, you're only computing a small amount, but the amount of data that you're then feeding to generate output tokens for the decoding phase of inference, you're dealing now with much larger heavy caches. So for a lot of reasons, the systems that were

really good at the first iteration of AI, generative AI inference are not necessarily the ones that are very good for agentic AI inference.

Jon Krohn: 04:22 Right. That makes a lot of sense. And I guess SambaNova's chips specialize in the inference phase, which if you probably know this step better than me, but my understanding is that chips actually rarely get used for training. Chips are mostly used in the AI workload sense for inference. 99% of the time or more chips are being used for inference. So yeah, let us know about whether I'm right about SambaNova being specialized for that big chunk of the AI workload market.

Anton McG.: 04:49 Yeah. I mean, traditionally we did both training and inference. We started to specialize in inference to frankly lower the aperture of work and focus our resources on what is a market that is becoming much larger relative to training, as you mentioned, but also where the barriers to entry are lower, it's less concentrated market, whereas in training there's just few buyers, they buy large amounts of compute, but Nvidia frankly is that market quite locked in. So the inference market strategically is better for non-incumbents, but then also just the profile of our chip and what it's good at lends itself really well to be able to really differentiate on inference, which over time that has become more true. And just on your data point, even in the training phase, there is more and more inference. There's more need for inference compute for training now than there is for actual back propagation for the actual training part

Jon Krohn: 05:55 Of training with reinforcement learning. I had never

Anton McG.: 05:57 Thought of that. So much of the post-training compute is actually just inference compute.

Jon Krohn: 06:04 Right. Yeah, that's a really good point. And I somehow had never thought of that myself. So what makes

SambaNova chips different? You claim that they're the fastest inference chips, but I'm sure a lot of companies claim that. How do you compare yourself with other chips available out there and how do you achieve those differences?

Anton McG.: 06:27 Yeah. Speed is relative. For end users, speed really matters in terms of how quickly they get their output tokens. For service providers that are buying systems, our customers, the speed that matters is their payback. So they buy a big system or they buy many systems. They want to know with this big capital investment that they're making, how quickly are they going to make a return on their investment? And that's where, from our perspective, our chips are incredibly differentiated because we provide the speed, the token generation speed, but we balance it with really, really good throughput. And this really is the juxtaposition that AI inference finds itself in where if you want to give users more speed, which they really need because agentic AI is all about automating work. The more work you can automate, the faster you can automate it, the more you can differentiate from your competitors and move faster than your competitors.

07:29 But the more you do that, the fewer concurrent users you can actually run on each chip. So this is sort of why we see these Pareto curves where on the Y axis we see throughput per chip and on the X axis we see speed per user and those things are at odds with each other. So what we do incredibly well is balance both of these things give you really, really fast speed with really, really great throughput such that as a service provider, you're able to charge more for your tokens because you're serving them to your customers faster, but also serve more and more and more users such that every second you're making more money. You're making more money and you're making more money per token.

Jon Krohn:	08:15	Right. So you're getting speed and throughput on a single rack. I'm starting to understand this and the differentiator there for SambaNova being on that part of the Predo curve where you're maximizing both of those things as much as you can at the same time. But how do you do it? What is different about a SambaNova chip? Is it the memory, the compiler? Is it the chip itself?
Anton McG.:	08:39	Yeah. I mean, it's a combination of those things. It's hard to disentangle it all because the right compiler is only right for the right chip and the right workload is only right for the right compiler. Really what it comes down to is the prefill phase of inference, which is processing the input tokens is highly parallelizable. It runs well on GPUs, but the bottleneck is the decode phase where you're generating output tokens. And that's where the GPU's architecture is just ill suited. The reason being it's a kernel by kernel execution machine. It breaks the model's computational graphs into these chunks that they call kernels and they execute the computation of that kernel and then they feed it back to their external HPM memory. They load the next kernel and they do it again. And so it's very, very time consuming on their bottleneck by their memory bandwidth.
	09:39	Our architecture, the RDU's architecture is a data flow based architecture where we just -
Jon Krohn:	09:45	If you don't mind me quickly interrupting you, I think a key thing here, and I can't believe I didn't mention this earlier, is that instead of calling it a GPU, like an Nvidia GPU, you call your processor an RDU, or I believe it stands for reconfigurable data unit. And so our listeners would use that in lieu of a GPU right at inference time with their AI workload?
Anton McG.:	10:08	They could use it alongside it. They could use it in lieu of it or they could use it alongside it.

Jon Krohn:	10:14	I see.
Anton McG.:	10:15	The GPU being really good at prefill at the input token processing, but insufficient for the decode, the output token generation phase. So the reconfigurable data flow unit is our chip. And what makes it really different is you are not executing all of these kernels, these slices. You are loading the entire model, rolling it out spatially across the chip and letting data flow through such that you're never having to do all of this back and forth with your external memory, nor are you having to have all of this overhead of communication overhead between each chip. So really what the RDU, what SambaNova's architecture enables is the ability to perfectly overlap communication and computation. And this is very, very, very important for the decode stage of inference because time is so finite. All of these operations are happening at such a granularity that any overhead in computation or communications will just automatically become a big bottleneck and that'll slow down your ability to generate tokens.
Jon Krohn:	11:32	Right. Gotcha. So this RDU reconfigurable data unit, I guess it builds on what you were talking about nine years ago as this approach to having really refined data flow. And so it's kind of like data moving like an assembly line through the processor and it's kind of that specialized use that allows you to get such throughput at such high speed.
Anton McG.:	12:04	Yeah. And really hiding all of that overhead. And the other thing is it's not just on one chip, it's across many chips. So a lot of the overheads that the GPU will have to deal with for the decoding phase comes from the fact that they need to parallelize over many chips. Nvidia created the newer version of their system, the Blackwell system and their upcoming Rubin system has a single all to all communication between 72 chips in a rack. The reason they do that is because they need more chips to be able to

run bigger models, longer contacts lens with more throughput and more speed. The problem is as they use more chips to run these models, the overhead of the communication between the chips really exasperates their inefficiencies. So by virtue of our architecture, we are able to scale linearly. So whenever we can be twice as fast on two chips as we can be in one chip and four times as fast in four chips.

13:11 So that ability to scale linearly is really an incredibly important characteristic of the architecture.

Jon Krohn: 13:19 You should have led with that, Anton. That's brilliant. That's so easy to understand and how that differentiates you against your competitors. So yeah, so that's the whole idea of this assembly line, you can break up loads and have them sent down each of the assembly lines that you have. The more RDUs that you have, the faster the inference time compute, the more throughput that you have. That is pretty cool. Talk us through how this works. If we have listeners out there today who are like, "This sounds awesome. I'm doing lots of inference time workloads. I'd love something that's more efficient than what I can get from the well-known incumbents out there." How do they do it in terms of hardware? How do they get it set up? In terms of software, is there some kind of equivalent to a CUDA library or how do they run and process on RDUs?

Anton McG.: 14:15 So it starts with how easy is it to deploy the hardware? And another consequence of our architecture is that we don't need to densely pack our racks with 72 chips. We keep fewer chips in a rack such that the rack can be lightweight and air cooled. So by virtue of doing that, now we can go into a lot of these existing brownfield data centers or retrofitted data centers that were previously telephone exchanges or data centers that ran CPUs, for instance, that don't have the same need for power delivery, cooling, et cetera. So we are in a world right now

where perhaps the biggest bottleneck is these net new data center build outs. And that's going to take some number of years to play out and for all of that data center capacity to come online. So we can be deployed into existing brownfield data centers today where other providers can't do this because they're having to build a very dense racks.

15:21 So that's the first big unlock. Just the ability for customers, whether it be enterprises or neoclouds and service providers and sovereign clouds, if they're able to secure existing brownfield data center capacity, they can roll in our racks and get it up and running same day as it rolls in. In terms of then the software stack, we have specificities about our architecture that will see us mapping the model differently. We don't map the model the same way the GPU does because it's a different architecture, our benefits are different, but all of that is abstracted from the user. And there's really a couple of different entry points that a user can come in at. The first of which is if they just want models, they know which models they want and they want them to run fast, they can click a few buttons and deploy that model and start serving inference.

16:18 Then you have customers that want to bring their own model, maybe it's not a model that we support out of the box, providing them an ability to do that via some abstraction that they already know like PyTorch and VLLM. We provide that capability as well. Then for researchers that are really trying to figure out how to better utilize this architecture in ways that we haven't though about, for instance, in Samanova because we need to be focused on our customers and our immediate term problems, being able to democratize RDU programming so that anybody can do it, that's what we're now enabling, making it easier to interact with the lower levels of our stack. And the reason that's become so important is because now AI agents can do this. You can

just point AI agents at our software and let them iterate and try to bring these models up and map them and make them run really fast and performantly.

Jon Krohn: 17:17 That's all right. So thanks for giving us that tour of both how you get things set up from a hardware side and how you use Samanova RDUs from a software side. My understanding is that the latest generation of your SambaNova RDU is called the SN50. It's your fifth generation chip and I guess you're shipping them to customers this quarter. Tell us about the economics of that chip and how it compares to your competitors. How long would it take for somebody who buys an SN50 RDU to make their money back?

Anton McG.: 17:54 Yeah. So SN50, if a customer buys a cluster of SN50s and runs inference on these systems, even on the latest and greatest open source models, will get their money back within six months. And that is really unprecedented within this industry for something that as a substantial capital investment to then get that money back within six months and to be able to keep that system in production for five years, at least five years, and in some cases longer, minus your small amount of OPEX, everything after that six months is pure profit. So this is an incredibly profitable machine for any service provider.

Jon Krohn: 18:49 It's like a 10X ROI where it takes you six months to pay off the purchase and then the remaining, then you have nine times more time, assuming that they're keeping it online for five years to be bearing the fruits, basically as you say, a small amount of OPEX, but almost cost-free.

Anton McG.: 19:10 Exactly, exactly. So typically within this industry, a two to three year payback period is the expectation to be able to collapse that down into six months really is just a huge unlock for all of these service providers who, by the way, are already making money because they're running inference at scale. We're in a capacity constrained world

because inferences, we just have such a need for inference. To be able to get net new inference capacity, but also to be able to make such an incredibly attractive return on that investment is why our business has just really, really, really accelerated over the past

Jon Krohn: 19:51 Nine

Anton McG.: 19:52 Months.

Jon Krohn: 19:53 It's pretty remarkable. Can you tell us a bit about what your typical buyer is? Who's your typical customer?

Anton McG.: 19:57 Yeah. So we have a number of different segments who have similar needs, but their own specifics. The neoclouds, this emerging class of customers called the neo clouds are -

Jon Krohn: 20:13 Sure. Like Lightning AI?

Anton McG.: 20:15 Like Lightning AI. And there's a broad scope I would say of NeoClouds as a category. There are those that are focused on software and optimizing software and leasing hardware capacity from others. And there are those that are building capacity directly, building data centers, making capital purchases of systems, and then they're leasing that either directly to enterprises or consumers or leasing that capacity out to the software focused neoclouds. For better or worse, the industry has coalesced around calling all of these neo-clouds. But I would say more specifically, when I refer to neo-clouds as direct customers, they're those that are building data centers, building data centers, building racks, managing the data center operations because they're the ones that actually make the capital purchases. The other class of customers is sovereign clouds. They have differences, and in some cases are neo-clouds themselves, but some differences than the more general neo-clouds.

	21:25	And then we have enterprises. And enterprises have emerged as surprisingly late comer to this category because so much of AI inference offtake has come from these AI natives and startups. But any enterprise in the world that is not wholly embracing AI inference today is going to be left behind. So the risk for them is company defining and they are now wholly and readily embracing AI inference, not just consuming cloud APIs, but starting to think about their AI sovereignty, like owning their AI and not being reliant on third parties to control their destiny. So those really are the categories. Really, they're all service providers. Even the enterprises themselves are service providers. They're just service providers for internal users, but loads of similarities, but then lots of nuance differences too.
Jon Krohn:	22:29	Yeah, it makes perfect sense. My last technical question for you, and this is something that actually we're kind of getting right into perfectly before I started asking about the different kinds of customers you have, is that inference compute is scarce. The more that inference becomes available, the more usefulness that we find and the more demand there is for even more inference. And so because of that compute is scarce, what is the cost of running the wrong AI inference workload on general purpose capacity? So probably most of our listeners are using general purpose GPUs for their inference time. What's the cost to them of doing that?
Anton McG.:	23:12	I mean, ultimately the cost is time and money. We are inhibited by a couple of different things today. Number one is we have finite amount of energy that we can deliver to AI systems to run inference. You want to get as much out of every watt as possible. You want to be able to generate as many tokens for each watt of power that you deliver. Then you have space, which is a constraint and all of the infrastructure needed to support running these systems. And that's sort of the infrastructure and service provider layer. They're the big constraints they have.

Capital is another big constraint. More and more that's been solved through financing products, but still a real constraint. As you move up the stack, the constraints become what tokens am I consuming because not all tokens are created equal. You want your tokens to be high quality from a really, really good model, but you also want them to be very fast.

24:19 And we recently ran a survey with AI infrastructure leaders. Four out of five of them said they would be willing to pay a premium for faster inference. The reason they're willing to pay that premium is because they recognize that faster tokens are better tokens. It allows them to outcompete their competitors. If you are spending time and money on tokens that are low quality and low speed, then that's the opportunity cost. So the answer really depends on the layer of the stack, but those are the constraints across the market today. And frankly, we think we are very, very well positioned across all of those constraints.

Jon Krohn: 25:00 Nice. Well, thank you for the introduction to SambaNova Chips, to the SN50 RDU in particular and the advantages that that has over other ways that you can be doing inference compute, particularly in this agentic AI era that we're now in. Before I let you go, Anton, I think you've been prepared for this, although I forgot to tell you myself, which is that we usually ask for a book recommendation at the end of episodes. Do you have anything for us?

Anton McG.: 25:27 I unfortunately exclusively read classical fiction, so I don't know how relevant it is for

Jon Krohn: 25:36 This podcast. Our audience loves all the recommendations. Hit us up.

Anton McG.: 25:39 Good, good, good. I read a lot of books. The latest book that I finished was a book called Lonesome Dove, which is

about a very interesting time in American history after the Texas Rangers had spent a lot of time fighting the bandits and trying to find purpose. The interesting takeaway of that book actually is that sometimes maybe you've solved a big problem in your life and you're very proud of yourself, but we have this strange insatiable desire to just go and solve more problems sometimes without understanding what problem we're trying to solve. And I think it's an important lesson for all of us working in AI these days that it's incredible the amount of progress we've made, but we need to keep sight of what the end goal is and make sure that it's all for the right reasons.

Jon Krohn: 26:40 Nice. Love it, Anton. Thanks for that great recommendation. Nice takeaway from the book as well. Thank you. My final question for you is how people can keep up with your thoughts or on SambaNova after this episode. Yeah.

Anton McG.: 26:56 I'm lucky enough to be the only Anton McDonnell in the world. So if you search for my name on LinkedIn, high probability you'll find me as well as on Acts where I spend a lot of time reading and sometimes commentating.

Jon Krohn: 27:16 Nice.

Anton McG.: 27:17 And Samanova more broadly is very active on LinkedIn Acts and all of the social media platforms.

Jon Krohn: 27:24 Perfect. Well, yeah, we'll have links to all of those in the show notes for our listeners. Anton, thank you so much for taking the time with us today and can't wait to see what comes out of SambaNova next.

Anton McG.: 27:34 Thanks so much, Tom.

Jon Krohn: 27:36 Great episode today. In it, Anton McGonnell detailed why agentic AI has changed the computational profile of inference, the trade off between speed per user and throughput per chip, and how SambaNova's RDU aims to

deliver both by laying the whole model out spatially across the chip. He talked about how that architecture scales linearly. So two chips are twice as fast as one and four are four times as fast. While the SN50 rack stays lightweight and air cooled enough to be rolled into retrofitted data centers, the day it arrives. I hope you enjoyed this hardware, AI hardware conversation. To be sure not to miss any of our exciting upcoming episodes, subscribe to this podcast if you haven't already. But most importantly, I hope you'll just keep on listening. Until next time, keep on rocking it out there and I'm looking forward to enjoying another round of the Super Data Science podcast with you very soon.