SADRAM Empowering AI

Online presentation at the **Göttingen AI Developers Forum**

GWDG - **University of Göttingen** and the **Max Planck Society, Germany**

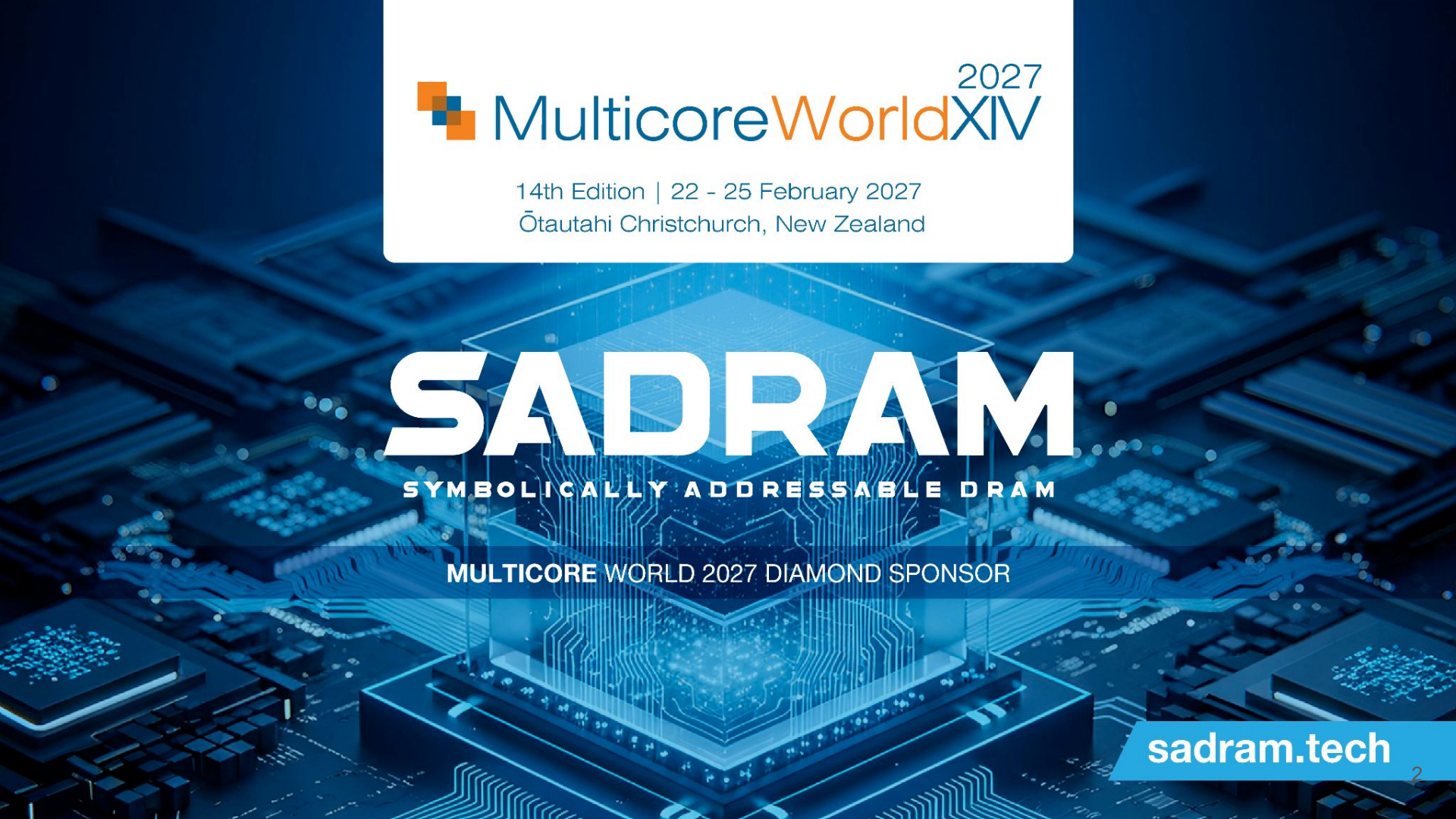
Nicolás Erdödy (Speaker)

Director, Open Parallel Ltd - Oamaru, New Zealand

Director, CCO, SADRAM Ltd.

Dr. Richard O'Keefe

Consultant, Open Parallel Ltd - Dunedin, New Zealand


2027
MulticoreWorldXIV
14th Edition | 22 - 25 February 2027
Ōtautahi Christchurch, New Zealand
SADRAM
SYMBOLICALLY ADDRESSABLE DRAM
MULTICORE WORLD 2027 DIAMOND SPONSOR
sadram.tech

Outline: **Three questions**

1. **What is SADRAM? (4 - 12)**
   - What is Processing Near Memory about?
   - What special tasks does SADRAM accelerate?

2. **What is AI? (13 - 38)**
   - Classic (symbolic) AI
   - Modern (neural) AI

3. **“Empowering” what, exactly? (39 - 45)**
   - Training issues
   - Output issues

What is SADRAM

- SADRAM applies the **Processing Near Memory** (PNM) concept
- To improve the speed and reduce the cost of **in-memory database processing**
- Including relational tables, spatial data, semi-structured data, graph data, and free-text data.

Processing Near Memory: Time

Distance from CPU to memory in cycles

- 1 cycle - register
- 2 cycles - L1 cache
- 5-8 cycles - L2 cache
- 30-50 cycles - L3 cache
- 150-200 cycles - main DRAM

You can do a lot of processing in 200 cycles.

Processing Near Memory: Energy

- Accessing DRAM takes energy as well as time
- It has been reported that 62% of the energy used by consumer applications is spent moving data between DRAM and CPU (*Source: Boroumand et al., "Google Workloads for Consumer Devices," ASPLOS 2018. Data movement accounts for 62.7% of energy in consumer apps.)*
- Idea: ***don't*** move the data, move the ***processing***.
- This is an old idea, but with today's large chips there's ***room*** in the memory chips for very simple regular processing.
- "Neural" chips mix memory and simple arithmetic, for example.

Processing Near Memory: Directness

- General purpose CPUs spend a lot of energy *being* general purpose.
- General purpose vs specialised purpose is like interpreted vs compilers: spending effort “deciding” what to do at runtime where a specialised system can’t help but do what it’s meant for.
- System-on-a-Chip designs put specialised systems such as cryptography, compression, video and audio decoding on the same chip as the CPU.
- Take less time and energy, but can’t reprogram for new tasks.

Processing Near Memory: Choice of task

- PNM has standard memory cells, lots of very simple fixed processing units. One or more small CPUs (ARM Cortex M0, RISC-V, SAMPU -SADRAM's own CPU) can go in the same package but for technical reasons, best not on the same die.
- The extra stuff has to be *useful* to be worth adding.
- We need a task that is often needed, uses a lot of memory, and depends on a small number of simple operations.
- Linear algebra is great for scientific processing and modern AI
- Sorting and searching is great for databases…AHA!!

So what then is SADRAM?

- “Symbolically Addressable DRAM”
- SADRAM is an application of PNM
- That adds lots of comparators and shifters and a simple control unit to otherwise unmodified DRAM
- In order to build and use sorted indices for in-memory data
- Think “in-memory B-Tree indices for in-memory data” but updating and using indices without the CPU getting involved.
- SADRAM augments the DRAM.

Reality check: What exists (and doesn’t)

- The hardware design exists
- A prototype using and FPGA exists
- This includes the programmable SAMPU
- A toolchain for the SAMPU exists
- Sample code is running
- What doesn’t exist: a C++ binding for application programmers. The architecture of that binding is being vigorously debated.
- Also nascent: complete sample applications.

What kind of data can be used as keys

- Numbers
- Strings (including URLs)
- 2-D and 3-D data (8 bytes is enough to represent any point on the Earth with 1cm accuracy)
- Graph edges (8 bytes handles up to 232 nodes)
- Anything you can extract a hash from (like Microsoft SQL Server indexing JSON)
- Addresses of tuples, allowing graphs and annotations

Priority queues

- A data structure supporting add(in data, in priority), is_empty?, and remove_best(out data, out priority) operations and perhaps decrease_priority(in node reference, in new_priority).
- Used in graph search (Dijkstra's and Prim's algorithms) and AI search problems (A* and iDA*), information retrieval (reporting best matches first), and statistical natural language processing (process most likely analyses first). "Maps"-style route planning is an example.
- SADRAM is good at this.

What is AI?

- There are two different things called Artificial Intelligence.
- Classic AI (20th century)
- Modern AI (21st century)
- They do things very differently and have different strengths and weaknesses.
- Classic AI led to Lisp Machines and Connection Machines.
- Modern AI has neural compute units/tensor processing units
- Both can benefit from SADRAM but in different ways

Classic AI

- Knowledge is expressed in *knowledge* bases
- Which are collections of *facts* and *rules*
- Expressed in a *formal language*
- Which has a *semantics*
- And a clear idea of when some piece of knowledge is more or less general than another and when something is or is not a valid consequence of or support for another.
- There are notions of *soundness* and *completeness*

Classic AI: aims

- Find ways to solve problems that would require intelligence if we solved them
- Knowledge should be *intelligible* to a human being. You should be able to ask “why do you think that?” and get a meaningful answer.
- The recommendations or actions of an AI program should be *justified*: they should be sound consequences of beliefs the AI holds and we also accept.

Classic AI: problems

- The knowledge acquisition bottleneck
- Brittleness
- Scaling
- Success

The knowledge acquisition bottleneck

- Training people to write knowledge in formal languages is like training them to program, only harder.
- Writing knowledge down in a formal language is like programming, only harder, especially because much of the knowledge we need to express is implicit, we know it but we don’t know we know it. (If you have an autistic colleague or family member you know how frustrating it is for you both when you think something is so obvious it “goes without saying”, but actually doesn’t.)
- This led to machine learning as a major hope in classic AI.

Machine Learning

- Machine Learning is a form of *abduction*, not *deduction*.
- That is, you can demand that what you learn should explain the data you are learning from. But you cannot expect that the data you are learning from imply a single model. And that means that you can only *hope* that the model works for the next observation; you can never be sure.
- And that goes double for Modern AI, where the model might not even explain all the data seen so far.

Knowledge Discovery in Databases

- “KDD is the process of extracting valuable insights and patterns from large datasets through a series of steps, including data selection, preprocessing, transformation, mining, and interpretation”.
- - Geeks-for-geeks web site.

- Hello SADRAM!

Brittleness

- Classic AI programs would sometimes work brilliantly within their domain, but fail just outside them.
- For example, a natural language processing program might be perfect with “The fool has said ‘I am bright’” but fail completely with “The fool hath said ‘I am bright’” or “The fool has said ‘bright me’”. Performance errors are common in written text and exceedingly common in speech.
- This led to the development of statistical methods, like statistical machine translation.

Scaling

- Classic AI is like bioinformatics.
- When you can describe a problem computationally, it’s usually
- Unsolvable or
- Exponential time or worse
- Response 1: heuristics. Use methods that might work fast or might fail. (Like randomised algorithms.) Abandon optimality.
- Response 2: search for problem spaces that are *just* tractable. This led to Description Logics – enrichments of propositional calculus that are solvable but wouldn’t be if any richer.

Success

- Whenever any subfield of AI succeeded, it immediately got called something else.
- The Graph Traverser was AI in the 1960s. Now Neo4J and GraphQL count as “database”, not “AI”.
- Logic programming was AI in the 1970s. Now Datalog counts as “database”, not “AI”.
- Planning was AI in the 1970s and 1980s. Now it counts as “Operations Research”.

Success 2

- Symbolic integration was “AI”. Now it is “Computer Algebra”.
- Theorem proving was “AI”. Now it is “Automated Reasoning”.
- Program verification was “AI”. Now tools like SPARK/Ada and Frama-C are just “program verification tools”.
- Game playing algorithms were “AI”. Now they’re just what games do.
- “Machine vision” was “AI”. Now it’s it’s own field.

Success 3

- Psi-terms were invented in an “AI” context for representing knowledge in the lexicon of a natural language and constraints on the grammar. Nowadays we call (a restricted form) JSON.
- Concept hierarchies were “AI” in the 1980s. Now they are called “ontologies” and they’re part of business-to-business data and medical information exchange.
- Description logics were “AI”. Now they are called RDF and OWL and key parts of the Semantic Web.

Success 4

- Heuristic search used to be thought of as “AI”.
- Now we have constraint satisfaction and combinatorial optimisation, with methods like genetic algorithms (the key data structure is a collection of (genotype, score) data resembling a priority queue) and simulated annealing and tabu search (with a growing data base SADRAM can manage) and a wide range of “nature-inspired” algorithms.
- Combine GPU-accelerated candidate evaluation with SADRAM-accelerated candidate pool management.

Pereira’s Prediction

In about 1982, Fernando Pereira, then a PhD student at the University of Edinburgh, predicted **“natural language processing will be solved by brute force.”**

ChatGPT and its relatives proved him correct.

Pereira spent 18 years at Google, recently as VP of Google Mind till he left in March 2026

Now is hiring PhDs in AI/ML for a company called Inceptive at $250k (https://job-boards.greenhouse.io/inceptive/jobs/4961579007)

(you’re welcome)(just checking if you are still there ;-)

Modern AI

- Knowledge is expressed as numerical *weights* on
- Connection between *nodes*, the weights are obtained
- By *training* from large data sets, which are *preprocessed* (transformed) in various ways.
- A node or weight has no meaning in itself, it is the pattern of weights which determines what a model “knows” or does.
- Different items of knowledge are not distinct structures but simply propensities encoded as patterns of weights for the same nodes

Modern AI: Aims

- Avoid brittleness. Learned patterns do not sharply match or fail to match observations, they match more or less well. “has” and “hath”, if suitably encoded, may be similar enough for one to be almost as good a match as the other.
- Handle very large training sets. Don’t do anything that doesn’t scale.
- Hope that “intelligence” can be mimicked as well or better by lots of experience rather than explicit rules. (Doug Lenat’s Cyc project was not to be imitated.)

Modern AI: Aims (cont.)

- Avoid the knowledge acquisition bottleneck. Have humans say what data is to be learned from, but leave the program to learn.
- *DON'T* worry about intelligibility. No human being is ever going to comprehend what 10^9 numbers mean.
- *DON'T* worry about justification.

Modern AI: Problems

- Alien responses
- Confabulation
- Long distance dependencies
- Plausible-but-wrong answers
- Intellectual property leakage
- Energy consumption

Alien responses

- Machine vision: people saw pictures as collections of features, programs saw them as *textures*. Classic example: a program confidently classifying random noise as a picture of a kitten because it was similar in texture to a kitten's fur. Another example: self-driving cars mistaking a picture of a stop sign for a real stop sign.
- The problem is inherent in machine learning unless you constrain (bias) the language of what's learned, which modern AI makes hard.

Confabulation

- Asked to write a technical paper with references, a modern AI may do so convincingly, but with references to papers that do not exist and data that are entirely made up. (Well, some human-written papers are like that...)
- There are detailed responses from AI convincingly arguing that the COVID “vaccines” killed many people and saved practically none. There are also detailed responses from AI convincingly arguing that the COVID “vaccines” were safe and effective. Something is wrong here.
- AIs will, in short, pass on human confabulation

Long distance dependencies

- Easily seen in AI-written fiction.
- Characters change name.
- Relationships between characters change.
- In one story, the narrator introduced herself as the CEO of a company she founded, but later in the story reported a problem to the CEO.
- The connection between a parameter and a calculated result may be lost.

Plausible-but-wrong answers

- If you use AI for coding, you have found that the AI's initial answer to your prompt looks really good, but doesn't work. It takes a long sequence of reminders and corrections before you get something that works, and then it is still likely to be inefficient.
- Actual query: "write a Smalltalk method to permute an Array of Integers in-place". The answer was a method to *randomly* permute an array. This is part confabulation (where did the query say anything about randomness?) and part plausible-but-wrong. This is the opposite of brittleness: when you don't have all the information you need, *make something up*.

Intellectual property leakage

- If the training data includes intellectual property belonging to others (e.g., if it includes patent records), then answers constructed for you may include IP you have not paid for.
- If the training data includes intellectual property belonging to yourself, then answers constructed for clients may contain IP they have not paid you for.

Energy consumption

- Modern AI systems use specialised devices to perform common numerical operations on massive amounts of data faster and using less energy. Even so,
- “By 2028, more than half of the electricity going to data centres will be used for AI. At that point, AI alone could consume as much electricity as 22% of US households.” (Shehabi et al., "2024 United States Data Center Energy Usage Report," Lawrence Berkeley National Laboratory (LBNL), Dec. 2024.
- “80-90% of the energy is used for inference.” Bigger models use more power. "The Carbon Footprint of AI," MIT Technology Review (citing NVIDIA data)
- Testing responses will increase energy costs even more.

Modern AI in the public eye

- “ChatGPT can hallucinate – meaning it can make stuff up that didn’t really happen – and it has more than a few biases it gleaned from researching human behavior on the internet. So be careful what you believe, and **fact check its answers always.”**
- “ChatGPT repurposes data to generate its responses. In so doing, it can **illegally copy the work of other people that’s protected by law.”**

Source: Pam Baker, ChatGPT for Dummies, 2nd edition.

Opposite problems

- You can *trust* classical AI but not *rely* on it.
- You can *rely* on modern AI but not *trust* it.
- Modern AI handles big data much better than classical AI, but it costs a lot more in energy because it handles big data.

“Empowering” what, exactly?

SADRAM empowering classical AI

- SADRAM is an excellent basis for graph data models.
- SADRAM is an excellent basis for the “triple store” used in Resource Description Framework (RDF) processing.
- SADRAM is an excellent basis for storing data about a program for program verification tools.
- SADRAM is an excellent basis for Knowledge Discovery in Databases.

SADRAM empowering modern AI

- Modern AI needs to be more affordable and more trustworthy
- SADRAM does not help directly with training.
- It can help with the process of storing, selecting, and transforming what to train on, addressing bias.
- SADRAM does not help directly with inference.
- It can help with the process of ***testing*** the results of inference, addressing hallucination. Let's see two examples.

(I) Checking for IP leakage

- Build and/or acquire a collection of IP that you do not want to be included in output to users. The MAREC collection of patents is a good example.
- For each response you generate,
- Use it to query the IP document collection. SADRAM accelerates free-text queries using a term index and combining postings lists to compute relevance scores and ranking similar documents.
- If any IP matches the response too well, reject it and generate another. This is an iterative process.

(II) Checking generated code

- Use identified problems to expand the prompt and try again. This is an iterative process.
- If the generated code involves database or graph processing, SADRAM may be useful in running test cases.
- Use white box testing algorithms to generate test data from the generated code (graph processing) and include test cases in revised prompts.

Three questions, answered

1. **What is SADRAM?** It is **Active Memory**. By embedding logic directly into the DRAM, SADRAM eliminates the energy-heavy "transportation tax" of moving data to the CPU.
2. **What is AI?** A dual-natured field. **Neural AI** provides intuitive scale, while **Symbolic AI** provides logical rules. SADRAM is the hardware that allows them to work together.
3. **How does SADRAM empower AI?** It provides the speed needed for **real-time verification**. It allows AI to check its own work against massive IP databases (like MAREC) to ensure outputs are safe, legal, and logical.

Summary

Conclusion

We have spent decades making processors faster, but we ignored the "straw" connecting them to our data. As AI scales toward a global energy crisis, we can no longer afford the cost of moving every byte. **SADRAM turns the memory itself into a partner in the thinking process.**

Summary

- **Stops the Waste:** Eliminates the massive energy overhead of moving data for AI inference.
- **Hardware-Accelerated Logic:** Makes complex graph and search operations a native hardware feature.
- **The Trust Layer:** Enables real-time IP verification and code testing to ground "black box" AI in reality.

The Punch Line "If 90% of AI energy is spent on inference, we don't have a computation shortage—we have a **distance problem. SADRAM is the end of the commute.**

The content of this slide was generated by Gemini ;-)

Contacts

Nicolás Erdödy nicolas.erdody@sadram.tech

Dr. Richard O'Keefe raoknz@gmail.com

SADRAM and its implementation (Jason Trout)
https://multicore.world/speakers-2026/jason-e-trout/

SADRAM benchmarks

At Multicore World 2027

Christchurch, New Zealand

22-25 February

www.multicore.world