paceval.®

Unique,
light integrable
mathematical engine that
enables product
innovation and
enhances productivity
of software
development
and maintenance.

Vision Paper

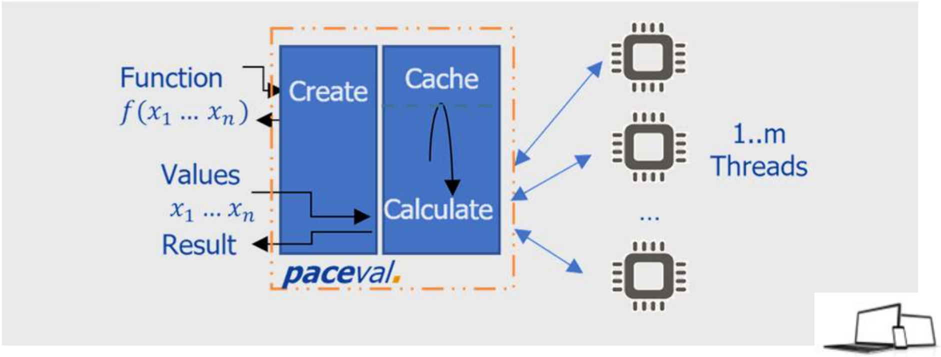
Faster mathematics with even less power consumption implemented on a mathematical engine in hardware

2022

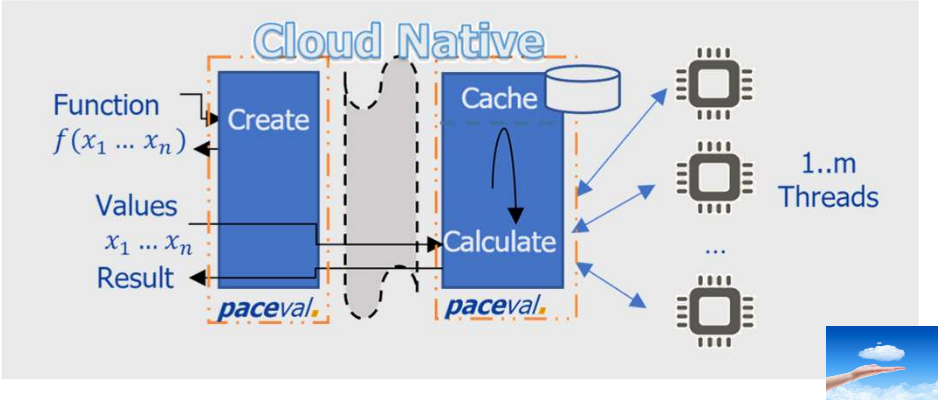
*Max Tegmark (physicist, cosmologist and machine learning researcher)

Vision Motivation

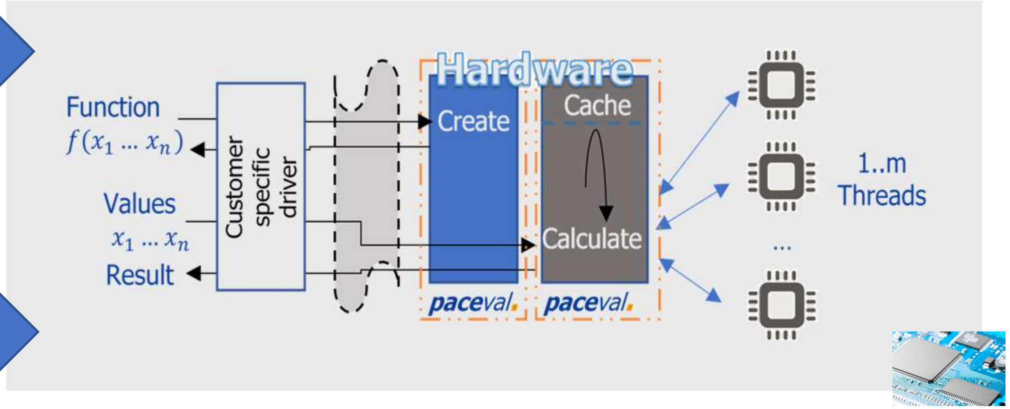
Why extend our mathematical engine in software …



… and cloud based mathematical engine …



… with a product in hardware?

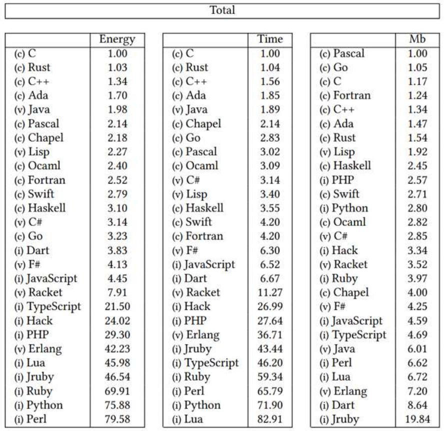


Status

Although our product already has a small binary footprint, impressive performance and at the same time low power consumption, **we would like to improve this status**.

***pace**val.* is written in C and C++ and this is already one of the **most sustainable combinations of programming languages**.

Of course, you have the option to add faster mathematical functions via our API. However, this also has the disadvantage that these mathematical functions are not as precise as those used by default in our product ***pace**val.**



Total					
	Energy		Time		Mb
(c) C	1.00	(c) C	1.00	(c) Pascal	1.00
(c) Rust	1.03	(c) Rust	1.04	(c) Go	1.05
(c) C++	1.34	(c) C++	1.56	(c) C	1.17
(c) Ada	1.70	(c) Ada	1.85	(c) Fortran	1.24
(v) Java	1.98	(v) Java	1.89	(c) C++	1.34
(c) Pascal	2.14	(c) Chapel	2.14	(c) Ada	1.47
(c) Chapel	2.18	(c) Go	2.83	(c) Rust	1.54
(v) Lisp	2.27	(c) Pascal	3.02	(v) Lisp	1.92
(c) Ocaml	2.40	(c) Ocaml	3.09	(c) Haskell	2.45
(c) Fortran	2.52	(v) C#	3.14	(i) PHP	2.57
(c) Swift	2.79	(v) Lisp	3.40	(c) Swift	2.71
(c) Haskell	3.10	(c) Haskell	3.55	(i) Python	2.80
(v) C#	3.14	(c) Swift	4.20	(c) Ocaml	2.82
(c) Go	3.23	(c) Fortran	4.20	(v) C#	2.85
(i) Dart	3.83	(v) F#	6.30	(i) Hack	3.34
(v) F#	4.13	(i) JavaScript	6.52	(v) Racket	3.52
(i) JavaScript	4.45	(i) Dart	6.67	(i) Ruby	3.97
(v) Racket	7.91	(v) Racket	11.27	(c) Chapel	4.00
(i) TypeScript	21.50	(i) Hack	26.99	(v) F#	4.25
(i) Hack	24.02	(i) PHP	27.64	(i) JavaScript	4.59
(i) PHP	29.30	(v) Erlang	36.71	(i) TypeScript	4.69
(v) Erlang	42.23	(i) Jruby	43.44	(v) Java	6.01
(i) Lua	45.98	(i) TypeScript	46.20	(i) Perl	6.62
(i) Jruby	46.54	(i) Ruby	59.34	(i) Lua	6.72
(i) Ruby	69.91	(i) Perl	65.79	(v) Erlang	7.20
(i) Python	75.88	(i) Python	71.90	(i) Dart	8.64
(i) Perl	79.58	(i) Lua	82.91	(i) Jruby	19.84

Source: https://greenlab.di.uminho.pt/wp-content/uploads/2017/09/paperSLE.pdf

*(the expected accuracy of these faster math functions is only between 5 and 9 digits)

MNIST benchmark comparison*

	Standard neural network processing GPU+CPU	*paceval.* Apple M1 (CPU only)
Power consumption	>500 Watt	39 Watt
Time per image	3-5 ms	12-15 ms
Purchase costs	>$7.000	$700
Running energy costs	>$850/year	$45/year

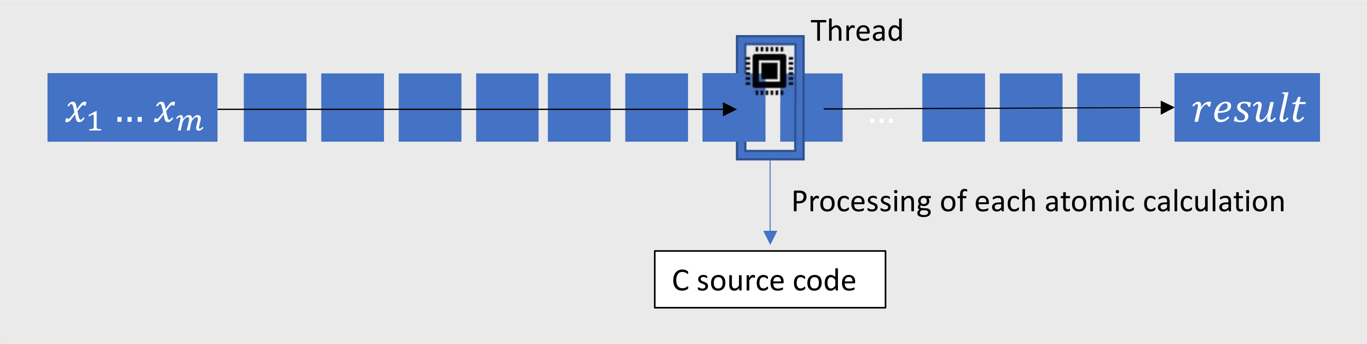
*(see document “paceval-Vision paper-'Mathematics is everywhere' Enabling sustainable distributed and decentralized mathematics.pdf”)

paceval®

What needs to be done to create a product in hardware based on our product ***pace**val.*?

paceval. intern

As described in our patent, ***paceval.* internally generates and processes a linked list of atomic calculations** that represent the user's mathematical function. This linked-list processing is done in a single C function “paceval_processDoComputationMath()” that is called by each thread.

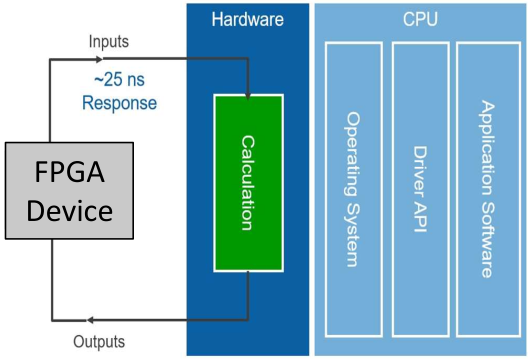


The C source code does this processing:
1. FETCH operator and operands (e.g. “addition of 2 and 3”)
2. DECODE and DECIDE use cached result or next step EXECUTE
3. EXECUTE operator and operands (e.g. calls C function to add 2 and 3 and get result 5)
4. WRITE BACK and cache results (this includes lower and upper interval limits or errors)

Obviously **this is the standard cycle used by all types of processors**.

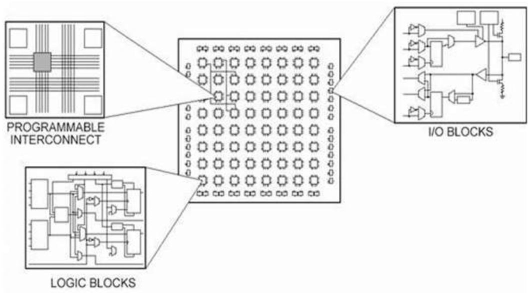
Hardware option FPGA

Systems based on FPGAs (Field Programmable Gate Arrays) offer many advantages compared to conventional implementations. The application logic in an FPGA is implemented in hardware circuitry instead of running on an operating system, drivers, and other application software. An **FPGA can function autonomously without interference** from other logic blocks.



Efficient systems, low power consumption

FPGAs offer the possibility of developing systems that are precisely tailored to the intended task and therefore work extremely efficiently. The power consumption can be significantly reduced by implementing an algorithm as an FPGA.



Accelerate software

Complex tasks are often solved by software implementations with fast processors. FPGAs offer an excellent alternative here, which offers a significant speed advantage over processor-based solutions through parallelization and adaptation to the application.

Since **processing is invoked for each atomic calculation** in the linked list, it makes sense to **convert this C function “paceval_processDoComputationMath()” to an FPGA**. But how?

To-do list: USB FPGA and *paceval.*

1. Get **USB FPGA form factor**
   e.g. see https://www.crowdsupply.com/sutajio-kosagi/fomu
2. **Convert C source code** of “paceval_processDoComputationMath()”
   to Hardware Description Language (HDL) for FPGA upload
   see https://en.wikipedia.org/wiki/C_to_HDL
3. Convert **additional C source code to HDL** for AI, e.g. +, -, *, /, exp()
   operators: multiply, add/subtract, accumulator, fused multiply-add, divide, square-root, comparison, reciprocal, reciprocal square-root, absolute value, natural logarithm, exponential*
   see https://www.xilinx.com/products/intellectual-property/floating_pt.html
4. **Add USB identification and communication**, i.e. when plugging in the USB-FPGA, the FPGA version of "paceval_processDoComputationMath()" is used automatically for AI**

*(this set of operators is sufficient for AI inference)
**(i.e. if only the operators in 3. are used)

Expectation MNIST benchmark comparison

	Standard neural network processing GPU+CPU	*paceval.* Apple M1 (CPU only)	*paceval.* Apple M1 (CPU only) + FPGA
Power consumption	>500 Watt	39 Watt	<26 Watt (assumption)
Time per image	3-5 ms	12-15 ms	<3 ms
Purchase costs	>$7.000	$700	$700 + price “product USB FPGA”
Running energy costs	>$850/year	$45/year	<$30/year

paceval.

Create value fast.

Contact: info@paceval.com