

Digital is dead, long live analog

Prof. Dr. Bernd Ulmann, anabrid GmbH

14. März 2024

anabrid



After many incredibly successful decades of stored-program digital computing different computational paradigms are shifting into the limelight:

- Analog Computing and
- Quantum Computing.

Why is that and what, exactly, is Analog Computing?¹

But first: Why should we even care?

¹Quantum Computing will not be discussed in this talk.

Problems of classic digital computers

Why analog/hybrid computing?

Although stored program digital computers are ubiquitous by now, they have a number of drawbacks which get more and more of a real problem for future applications. Some of these challenges are:

- High power consumption
- Clock frequencies are limited (energy consumption, pipeline length, etc.)
- On-chip structures can not be shrunk much further – atoms are quite big after all. . .
- Parallelism is hard to exploit with digital computers (AMDAHL's law)
- Only a tiny fraction of the silicon used in a digital computer actually performs computations, the vast majority deals with data storage, data management, control, etc.

The following slides depict some of these problems.

Size: Frontier



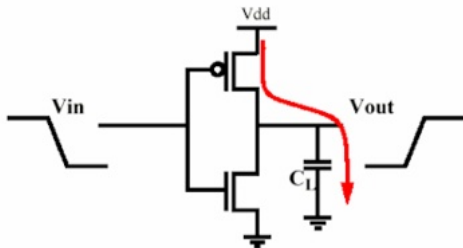
https://upload.wikimedia.org/wikipedia/commons/e/e0/Frontier_Supercomputer_%282022%29.jpg, 11.03.2024

Cooling



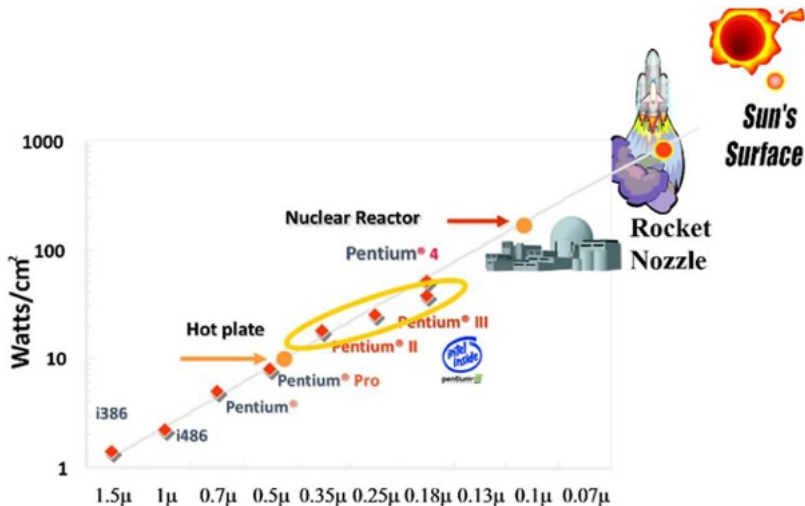
<http://www.lanl.gov/newsroom/picture-of-the-week/pic-week-1.php>, 23.01.2017.

A typical output stage looks like this:



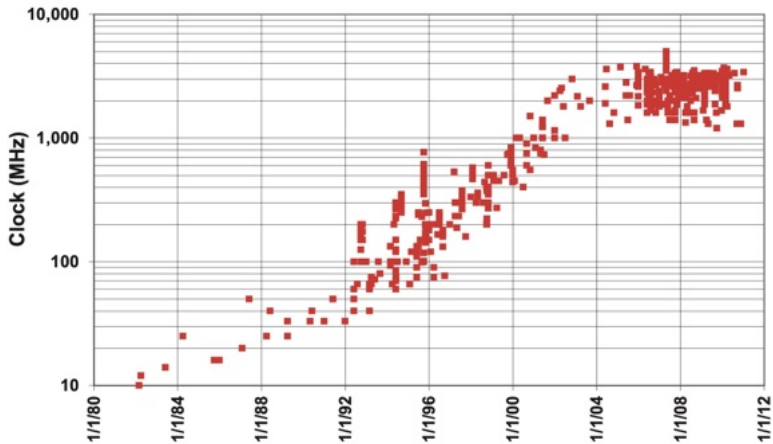
Problems:

- C_L gets charged/discharged at every change of the output.
- Both output transistors are switched on (very briefly) during a state transition.
- The transistors are not perfect and exhibit leakage currents.
- $P_{\text{cpu}} = P_{\text{dyn}} + P_{\text{sc}} + P_{\text{leak}}$ is super-linear with respect to the clock frequency.



See [ETIEMBLE(2018), Fig. 8].

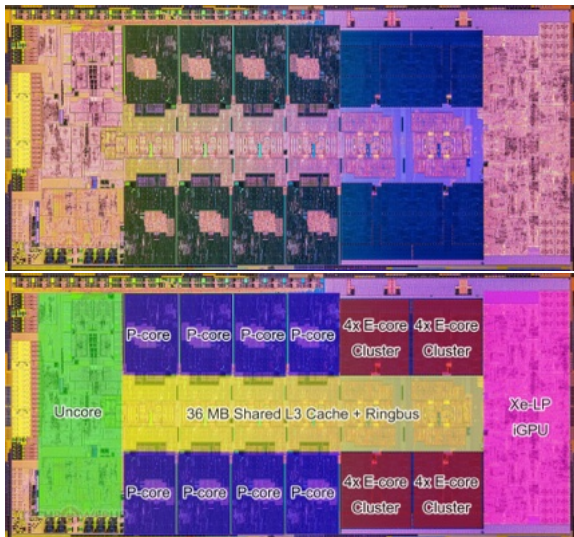
Clock frequencies



<https://wgropp.cs.illinois.edu/courses/cs598-s15/lectures/lecture15.pdf>, p. 11, 11.03.2024

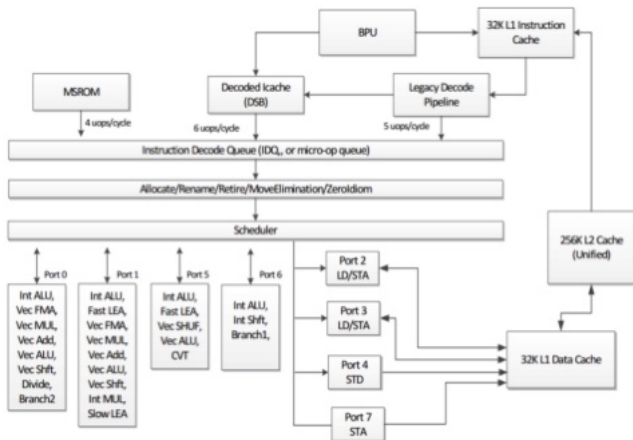
Typical chip structure

Intel Core i9-13900K:



<https://www.techpowerup.com/review/intel-core-i9-13900k/2.html>, retrieved: June 8th, 2023

Typical chip structure



<https://www.anandtech.com/show/11550/>

the-intel-skylakex-review-core-i9-7900x-i7-7820x-and-i7-7800x-tested/3, retrieved: June 8th, 2023

Fastest digital supercomputer (top500.org): Frontier, Oak Ridge National Laboratory, USA:

- 1 194 PFLOPS
- 8 335 360 cores
- Ca. 22.703 MW of electrical power
- About 52 GFLOPS/W
- Cost: \approx 600 000 000 USD

Human brain:

- Ca. 38 PFLOPS
- 25 W
- Ca. 1.52 PFLOPS/W
- Not digital but analog!

- The main problem is that algorithms are basically sequences of instructions to be executed serially to solve a given problem.
- In contrast to this, reality is inherently parallel!
- A biological brain is an ideal example for this – it consists of an incredible amount of basically rather simple cells, neurons, which work in full parallelism without any need for a central memory, central control circuitry etc.
- This is what distinguishes digital computers from analog and quantum computers:

Our current digital computers are stored program computers, while analog and quantum computers are not programmed in an algorithmic way but rather by setting up a circuit consisting of computing elements working in full parallelism to solve a problem.

Analog Computing

- Analog computers are programmed by setting up a model, an *analogue*. There is no algorithm at all (this is a major advantage! :-))
- They are highly energy-efficient.
- Values are typically (but not necessarily so) represented as continuous voltages or currents.²
- They operate in an inherently parallel fashion and do not suffer from AMDAHL's law, etc.
- Analog computers adapt seamlessly to our analog world – no need for data conversions.
- Analog computers excel at problems that can be described by (systems of) differential equations (most problems relevant in science and engineering fall into this category).
- Analog computers can even do some things typically attributed to quantum computers (oscillator based Ising machines).

²There are digital analog computers, *DDAs*, but we won't focus on these.

An analog computer

- contains many computing elements such as integrators (sic!), summers, multipliers, etc. (Very small systems feature about a dozens computing elements, while large systems contain hundreds and thousands and potentially many more.)
- These computing elements are connected to each other in order to form an *analogue* for the problem to be solved (that's where the name stems from). So there is **no algorithm** and **no memory** at all. The program is a *directed graph* describing these connections. (In this respect, analog and quantum computers are quite similar.)
- All computing elements operate in a fully parallel fashion.

The following (historic) pictures show the main differences between a digital computer and an analog computer:³

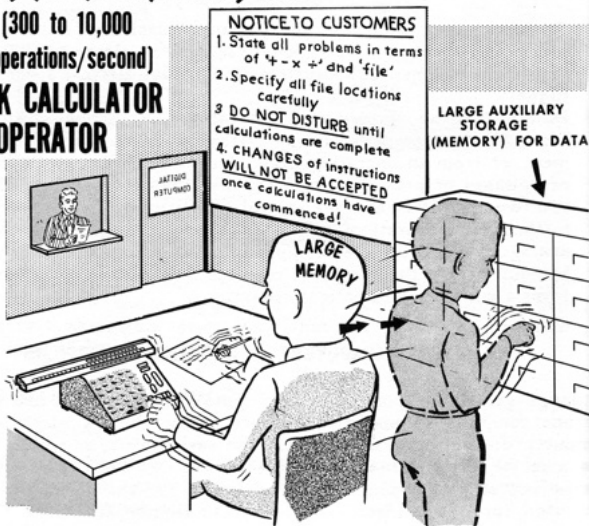
³Cf. [TRUITT et al.(1960), pp. 1-40/41].

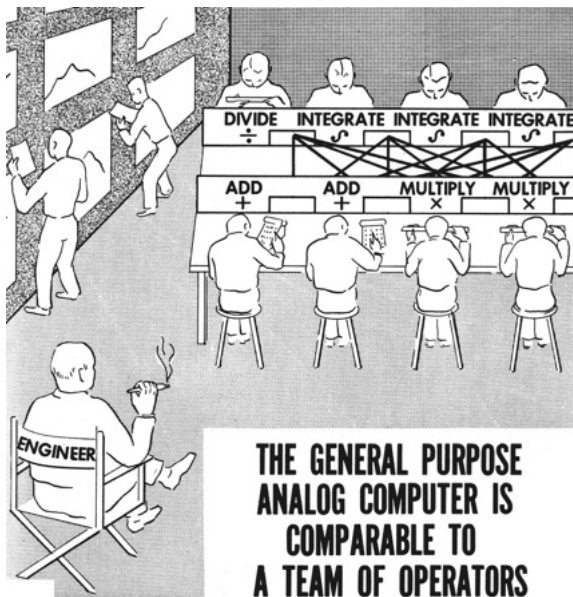
**A DIGITAL COMPUTER is equivalent to a very reliable,
highly paid, exceptionally fast**

(300 to 10,000

operations/second)

**DESK CALCULATOR
OPERATOR**





Of course, analog computers have some disadvantages, too (there is no such thing like a free lunch):

- Limited precision – typically about 3 to 4 decimal places (it turns out that this is often not a problem at all).
- Only time is directly available as free variable of integration, so solving partial differential equations requires some transformations/tricks.
- Generating arbitrary functions, especially those having more than one argument, requires special circuitry.
- Values are limited to the interval $[-1, 1]$, so problems must typically be *scaled* accordingly.

In modern applications, analog computers will be coupled with traditional digital computers forming *hybrid computers*:

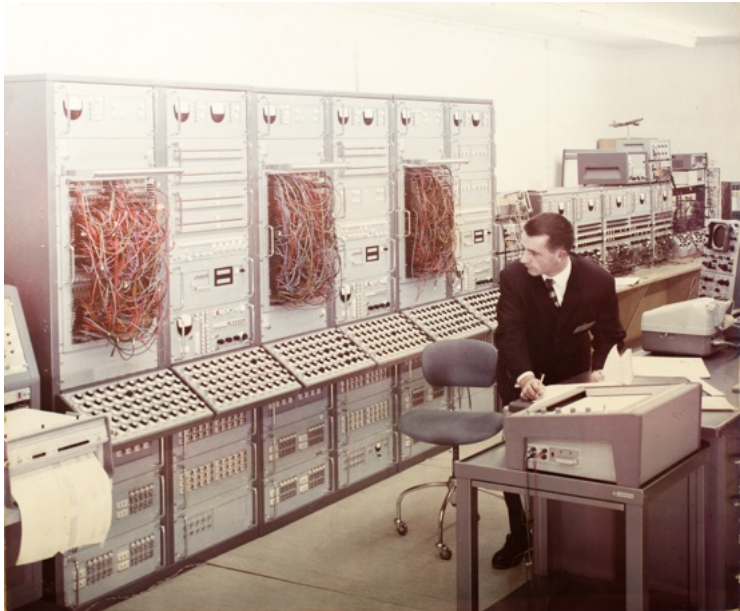
- A hybrid computer is a combination of a traditional stored program digital computer and an analog computer.
- The analog computer offloads the digital computer from compute intensive tasks and thus acts as a co-processor.
- The digital computer controls the configuration and parametrization of the analog computer.
- The analog co-processor must be integrated in a seamless fashion which requires an extensive and complex software stack including an abstract programming language for the analog computer, associated compilers, libraries, etc.

- High performance computing
- Artificial intelligence (implementation/training of ANNs, etc.) and machine learning
- Medical applications (cardiac pacemakers, brain pacemakers, insulin sensing/pump control) – the power consumption could be sufficiently low to power some implants with energy harvesting thus getting rid of clumsy batteries which limit life span and have to be recharged.
- Industrial control systems (basically stateless, not hackable in a traditional sense)
- Signal pre-/postprocessing for mobile devices and the like
- Trigger word detection for smart devices
- Monte-Carlo simulations, financial mathematics
- Optimization problems

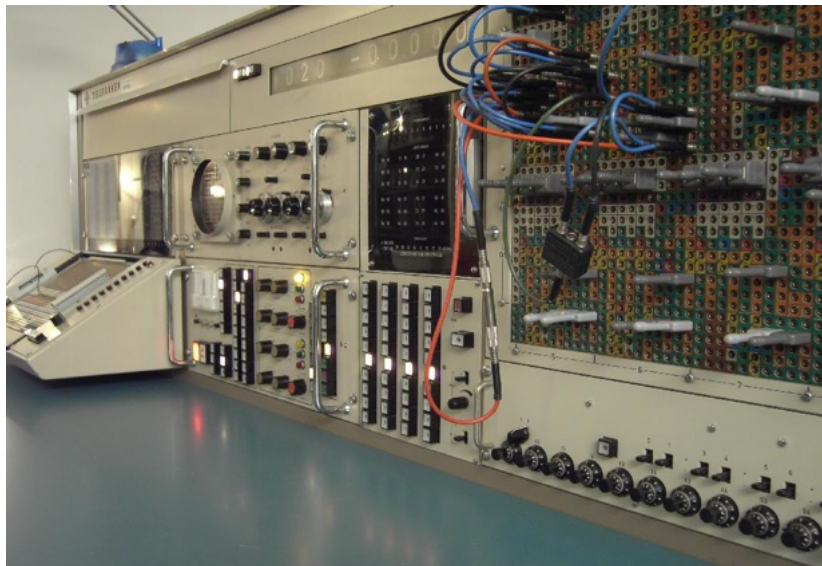
Historic analog computers

- Analog computers were dominant from the late 1940s to about the mid-1970s.
- They were (and still are) faster than stored-program digital computers in certain application areas and were used extensively in aerospace technology, vehicle design, chemistry, medicine etc.

Telefunken RA800, 1960



Telefunken RA770, 1966



VW car simulator, 1970s



- Digital computers became increasingly cheaper while classic analog computers remained quite expensive in comparison.
- Analog computers could not be used in a time sharing fashion, while digital computers typically offered this mode of operation.⁴
- Programming classic analog computers was quite time consuming; switching from one program to another also took a long time.

All of these problems can be easily overcome by modern integrated circuit technology making the inherent advantages of analog computers usable for the 21st century.

⁴There was some research in that direction but this did not lead to commercial products.

Modern developments

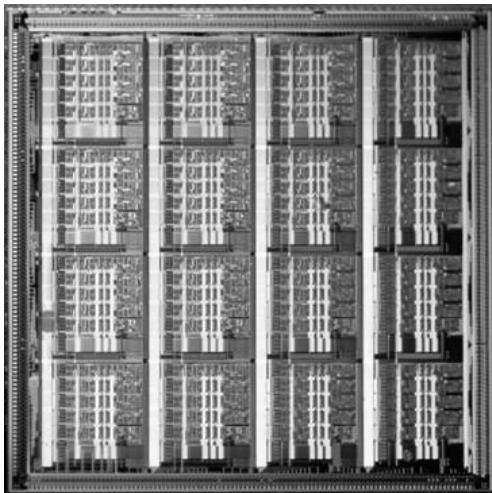
Modern approaches must address the following issues of classic analog computers:

- The patch panel belongs into a museum – modern analog computers must be automatically reconfigurable in a few milliseconds at most.
- This requires a *Domain Specific Language (DSL)* with associated compiler and libraries to specify an analog computer program.
- Tight integration of the analog computer in a digital environment is necessary which requires
 - low-level software (device drivers etc.)
 - libraries, numerical algorithms suitable for hybrid computer approaches. . .
- The computing elements should have very high bandwidth (ideally in the range of several 10 Mhz to 100 MHz) to achieve short solution times.

There are already some recent developments such as these:

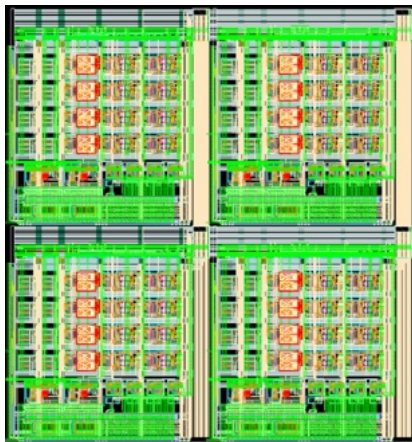
Academic projects

First fully reconfigurable analog computer on chip – mainly used for stochastic differential equations:⁵



⁵See [COWAN(2005)].

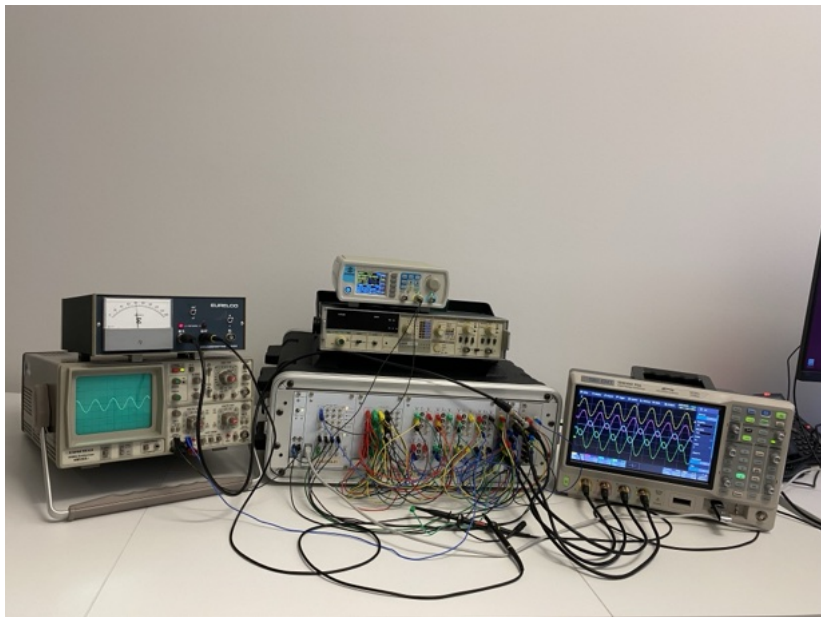
More advanced analog computer on chip:⁶



(Picture source: http://yipenghuang.com/wp-content/uploads/2016/07/v2_chip.png, retrieved 2019-09-19).

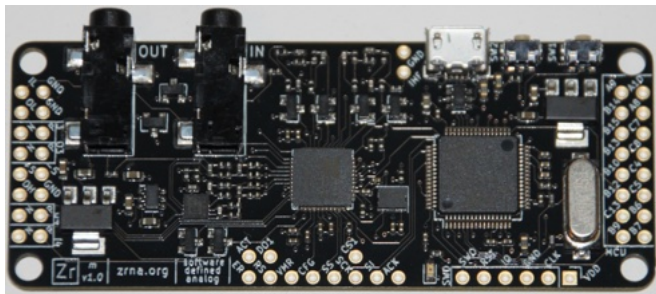
⁶See [GUO et al.(2016)].

- None of these chips made it into a viable commercial product.
- Software support is quite minimal.
- The computing elements often exhibit rather low bandwidth (about 20 kHz).
- Static and dynamic errors including phase shifts caused by the interconnect structure etc. are often pretty large.

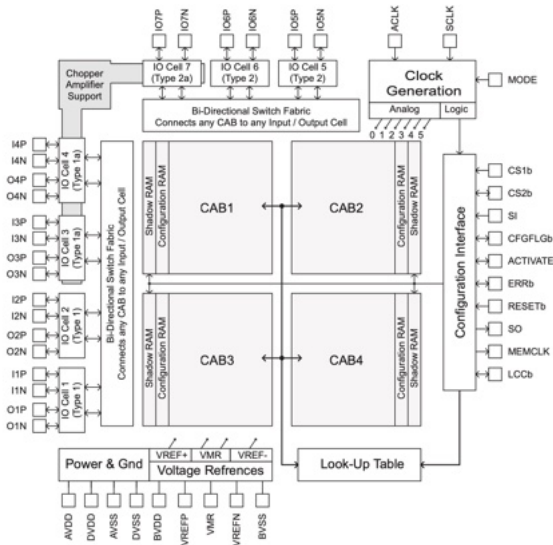


Application area: Edge processing

Anadigm manufactures FPAAs (using switched capacitors for the central interconnect structure) mainly aimed at the signal processing market. A typical application example is shown here (zrna.org):



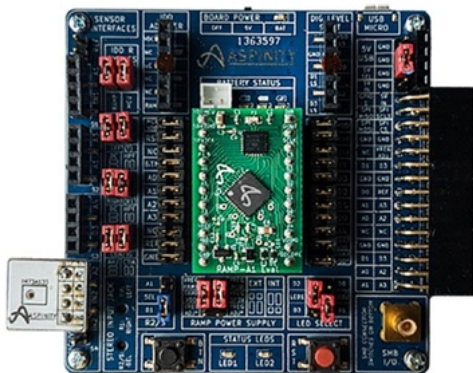
At its heart are *configurable analog blocks* (CABs) which are internally connected by switched capacitors.



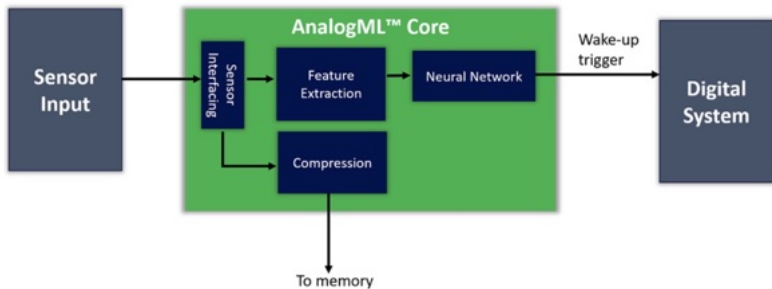
(Cf. [Anadigm(2006)].)

Application area: AI

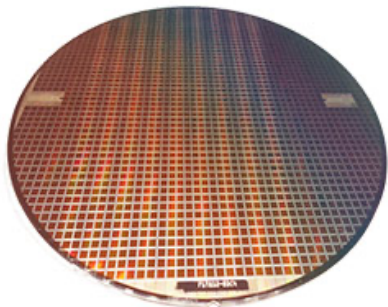
Aspinity⁷ develops special analog processor chips for voice recognition, acoustic event and vibration classification:



⁷<https://www.aspinity.com>, retrieved 25.09.2022

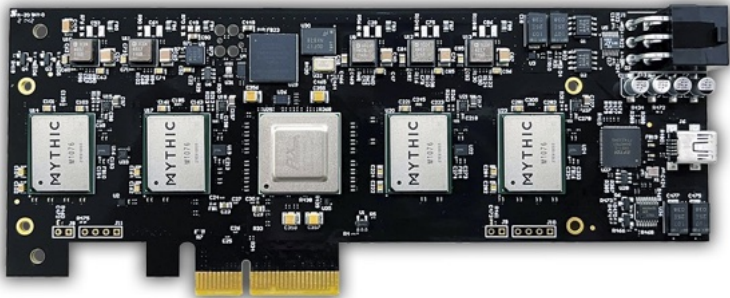


Source: <https://www.aspinity.com/AnalogML-Core>, retrieved 26.06.2023

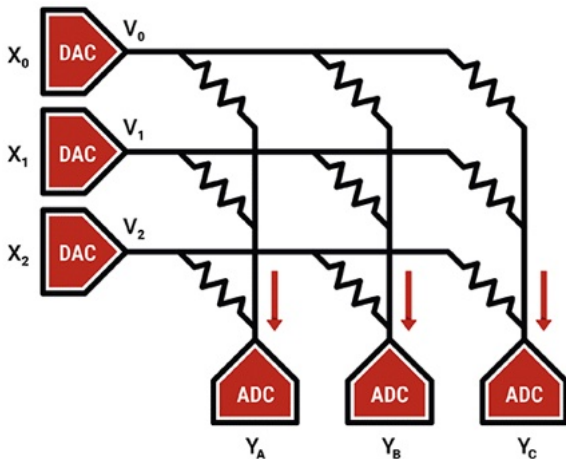


Aspinity
has also developed the
*Reconfigurable Analog
Modular Processor*
(*RAMP*) which is
aimed at neuromorphic
computing
(see <https://www.aspinity.com/RAMP-Technology>,
retrieved 26.06.2023).

MYTHIC⁸ develops *analog matrix processors (compute in memory)* aimed at AI applications:



⁸<https://mythic.ai>, retrieved 25.09.2022



At the heart of these matrices are Flash memory cells for the synaptic weights.

Source: <https://mythic.ai/technology/analog-computing/>, retrieved 26.06.2023

IBM has also developed an analog AI accelerator which was unveiled in 2022.⁹

- *Resistive RAM (ReRAM)* in the form of *phase change memory* is used for the synaptic weights. The PCM state is switched between amorphous and crystalline.
- This experimental chip implements $35 \cdot 10^6$ PCM cells.

⁹<https://research.ibm.com/blog/the-hardware-behind-analog-ai>,
retrieved 25.09.2022



When mimicking biological neural networks it is very desirable to implement artificial neural networks in a true 3D topology.

Using analog computing techniques, this becomes feasible due to the high energy efficiency.

IBM actively pursues this path with (see <https://research.ibm.com/blog/vlsi-hardware-roadmap>, retrieved 26.06.2023).

Memristors are often mentioned as “ideal” devices to implement synaptic weights in analog neural networks. A few questions and remarks might be of interest in this context:

- What type of memristor should/could be used?
- Memristors based on filament conduction have severe drawbacks such as a limited number of state changes as well as the necessity for a *device forming* period before their use in a circuit.
- There are approaches too memristors which are not based on filament conductivity but these developments are relatively new.
- How can memristors be integrated with a standard CMOS process?
- Will such devices allow it to implement self-learning artificial neural networks (“fire together, wire together”)?

Application area:

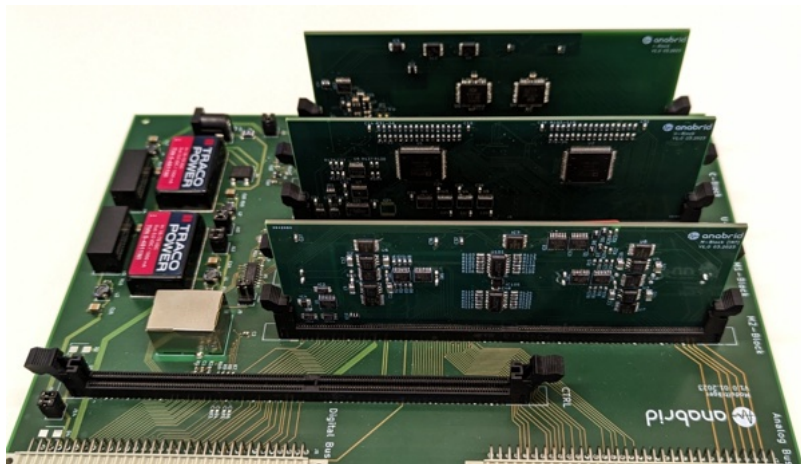
General purpose analog computing

anabrid GmbH (Germany) was founded in 2020 with the goal of developing, producing and marketing a high-performance reconfigurable general purpose analog computer on chip.

As of now we are working on

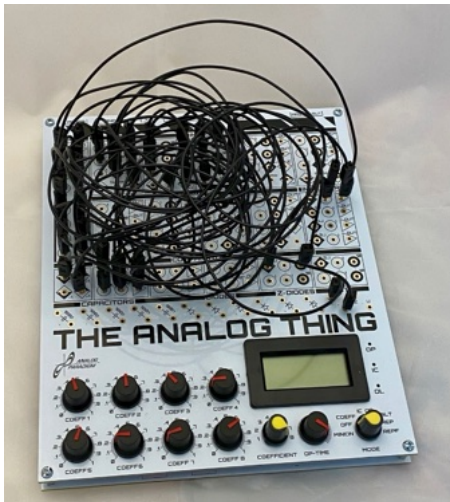
- interconnect structures for large analog computers,
- a DSL with associated compiler,
- hard- and software-support for high-performance hybrid computers,
- numerical algorithms for hybrid computers, and
- quantum inspired oscillator based Ising machines.

Current developments

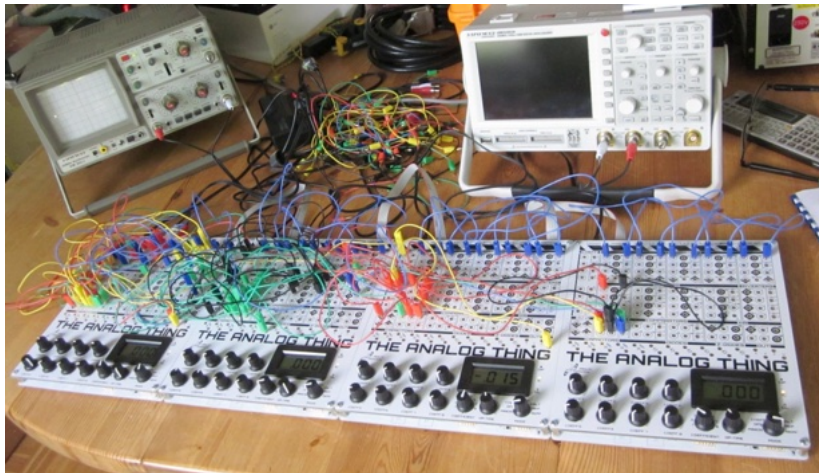


THE ANALOG THING

To bring analog computing to schools, universities and into the hands of hobbyists, we have created an open-hardware project:

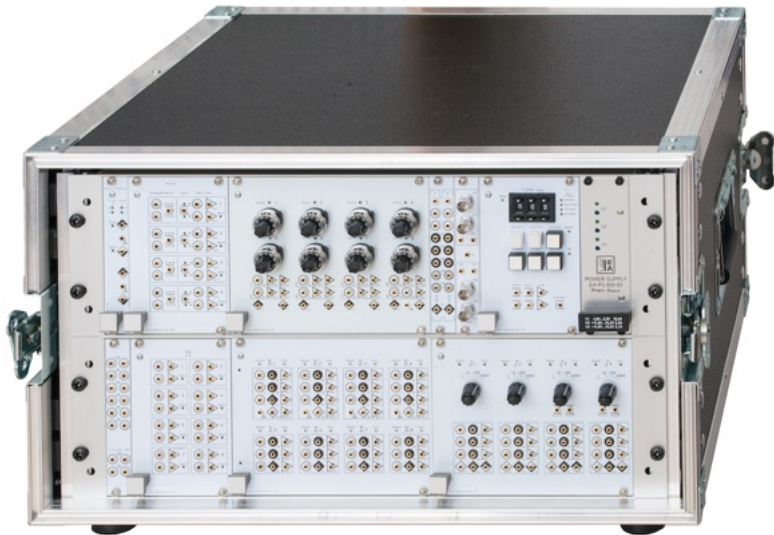


THE ANALOG THING



Model-1

For applications where THE ANALOG THING is not sufficient, there is also a modern modular analog computers, the *Model-1*:







**Thank you for your interest and
happy analog computing!**

The author can be reached at
`ulmann@anabrid.com`.

Bibliography

Bibliography

-  Anadigm, *AN13x/AN23x series, AnadigmApex dpASP Family User Manual*
-  GLENN EDWARD RUSSELL COWAN, *A VLSI Analog Computer / Math Co-processor for a Digital Computer*, Columbia University, 2005
-  DANIEL ETIEMBLE, *45-year CPU evolution: one law and two equations*,
https://www.researchgate.net/publication/323510528_45-year_CPU_evolution_one_law_and_two_equations,
retrieved: June 8th, 2023
-  N. GUO, Y. HUANG, T. MAI, S. PATIL, C. CAO, M. SEOK, S. SETHUMADHAVAN, and Y. TSIVIDIS, “Energy-Efficient Hybrid Analog/Digital Approximate Computation in Continuous Time”, in *IEEE Journal of Solid-State Circuits*, vol. 51, no. 7, pp. 1514-1524, July 2016



Thos. D. Truitt, A. E. Rogers, *Basics of Analog Computers*, John F. Rider Publisher, Inc., New York, December 1960



BERND ULMANN, *Analog and hybrid computer programming*, DeGruyter, 2nd edition, 2023

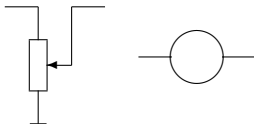
Appendix

Programming analog computers

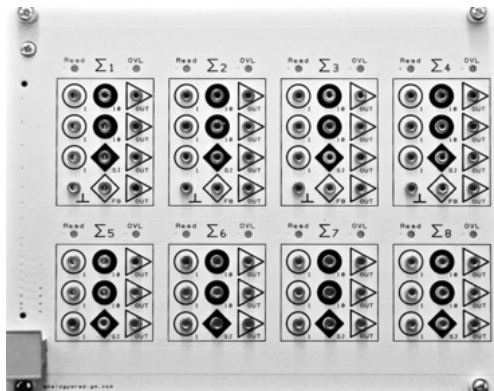
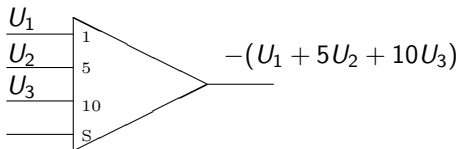
- Programming an analog computer (as well as a quantum computer) differs completely from the traditional algorithmic approach.
- Therefore, a few introductory programming examples are shown in the following slides.
- This is far from a comprehensive introduction to analog computer programming. Cf. [ULMANN(2023)] for more information.

Typical computing elements

- Coefficient units (e. g. a voltage divider/multiplying DAC etc.)

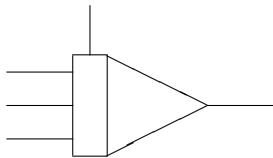


- Summers, computing $-U_{\text{out}} = \sum_{i=1}^n a_i U_i$:



- Integrators, computing¹⁰

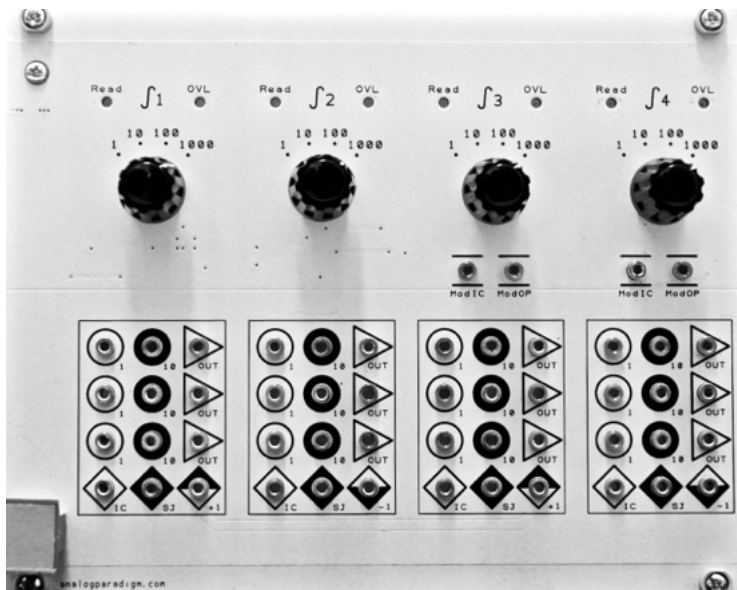
$$-U_{\text{out}} = \int_0^t \sum_{i=1}^n a_i U_i(t) dt + U(0)$$



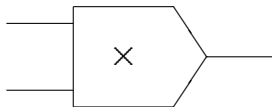
An integrator has three modes of operation: *initial condition*, *operate*, *halt*.

¹⁰The “magic component” – (near) perfect integration with a resistor, a capacitor and an operational amplifier.

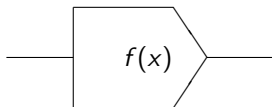
Typical computing elements



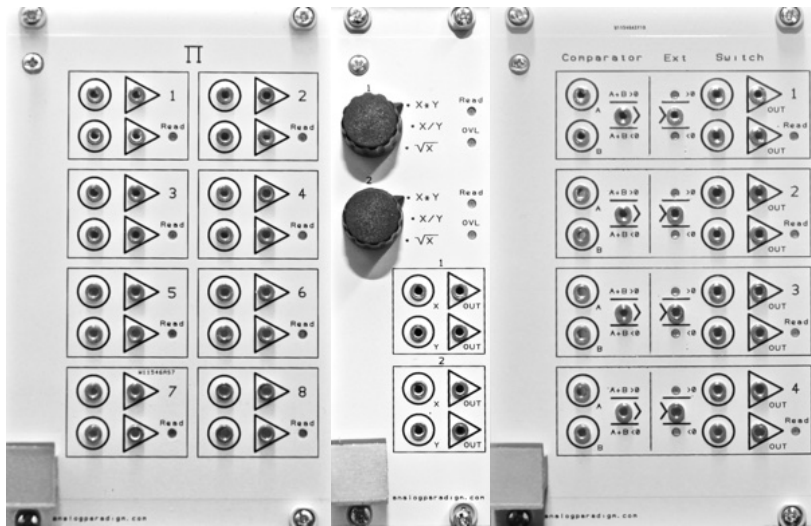
- Multipliers



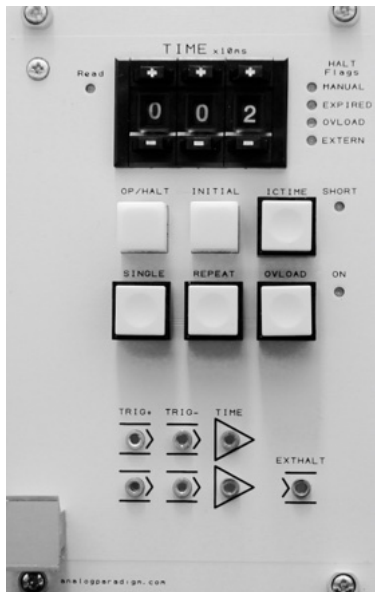
- *Open amplifiers* which can be used to implement inverse functions such as square root, division as well as special functions like limiters etc.
- Comparators with electronic switches
- Function generators (traditionally using polygonal interpolation, nowadays table-lookups using ADCs and DACs)



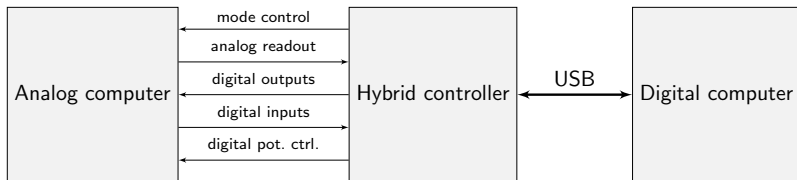
Typical computing elements



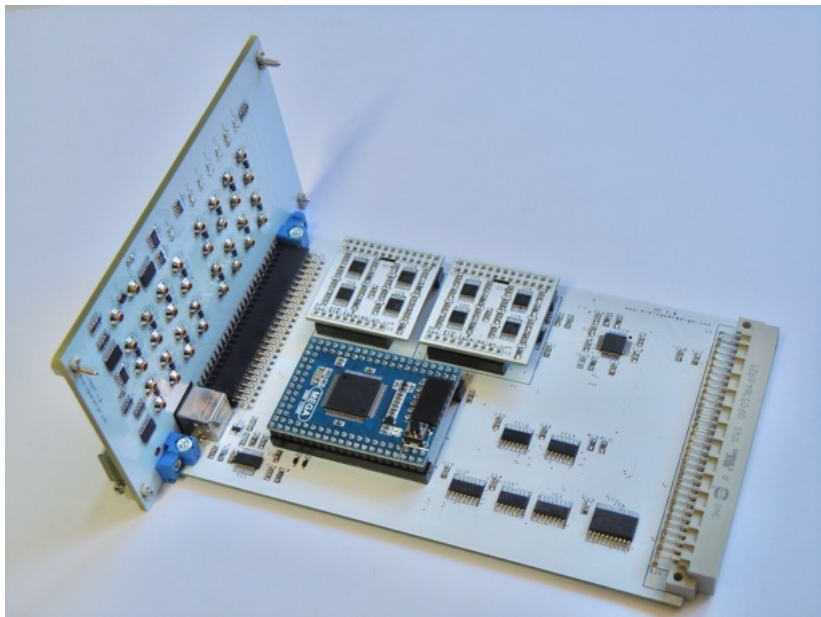
Control unit



Shown on the left is a typical control unit. It allows full manual control of the analog computer (initial condition, operate, halt) and also supports single run operation as well as repetitive operation.



Hybrid controller



Basic programming – ODEs

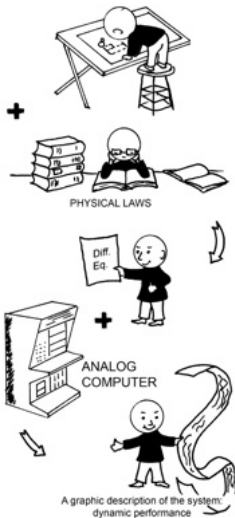
Programming an analog computer is much simpler than programming a stored program digital computer as there is nothing like an intricate algorithm controlling the system.

An analog computer program, i. e. the interconnection scheme for its computing elements, is derived from the problem equations.

A classic way to develop an analog computer setup is due to Lord KELVIN:

- 1** Solve the equation for the highest derivative.
- 2** Generate all lower derivatives by repeated integrations.
- 3** Derive all terms on the right hand side of the equation based on these lower derivatives.
- 4** Tie all these terms together. According to the equation this process started with, this must be equal to the highest derivative, so this value can be fed back into the circuit as this highest derivative.

A PHYSICAL SYSTEM
can be SIMULATED BY
AN ANALOG COMPUTER



A simple example

Often, a sine/cosine signal pair is required in a simulation. A typical way to generate this on an analog computer is by solving the DEQ

$$\ddot{y} + \omega^2 y = 0 \quad (1)$$

with the initial conditions

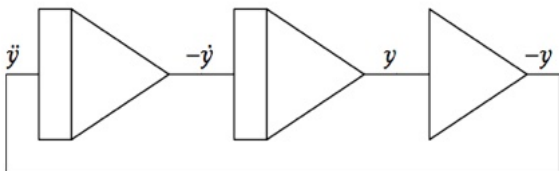
$$y_0 = a \sin(\varphi) \text{ und}$$

$$\dot{y}_0 = a\omega \cos(\varphi)$$

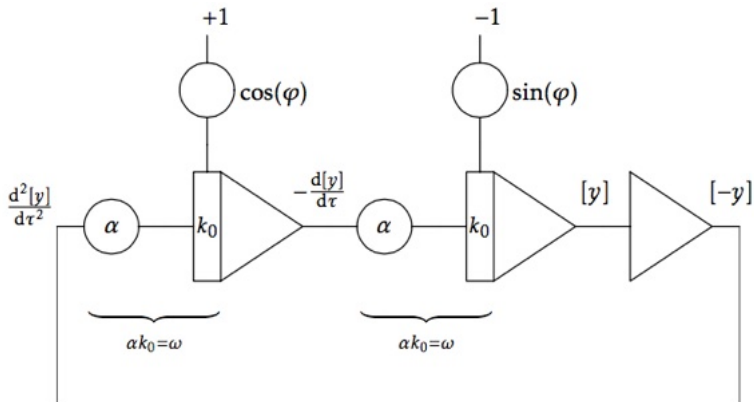
Solving (1) for \ddot{y} and assuming $\omega = 1$ yields

$$\ddot{y} = -y$$

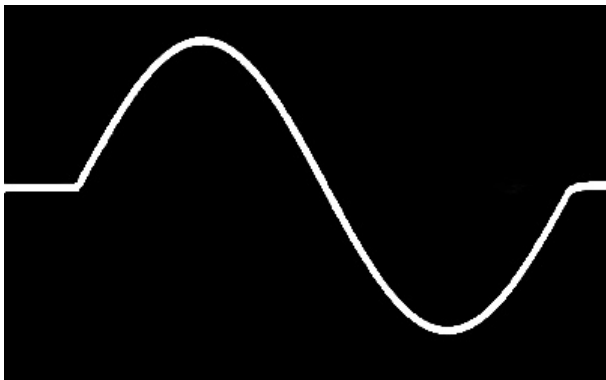
which leads to the following program sketch:



Taking ω into account as well as the initial conditions yields the following complete program:



This yields the following output at the second integrator (actual screen shot – the analog computer was running for the time of one period as determined by ω):



Two more complex examples:

- VAN DER POL-equation
- the heat equation

van der Pol equation

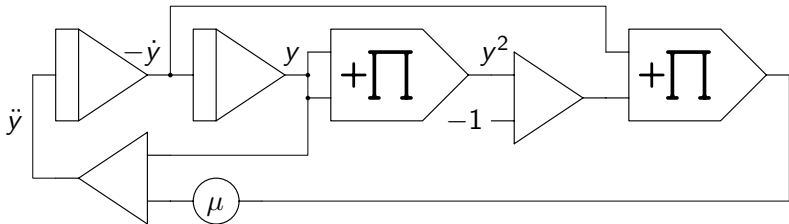
The VAN DER POL equation looks like

$$\ddot{y} + \mu (y^2 - 1) \dot{y} + y = 0,$$

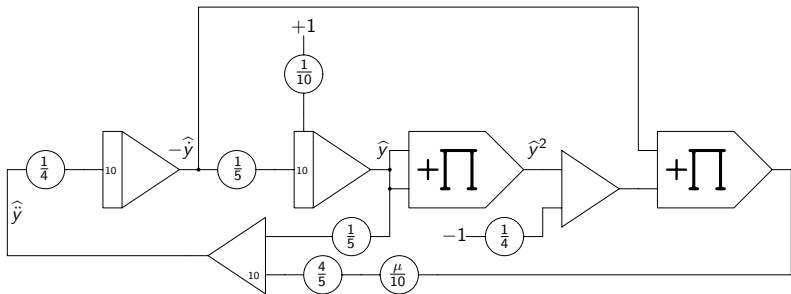
which can be rearranged, yielding

$$\ddot{y} = -y - \mu (y^2 - 1) \dot{y}.$$

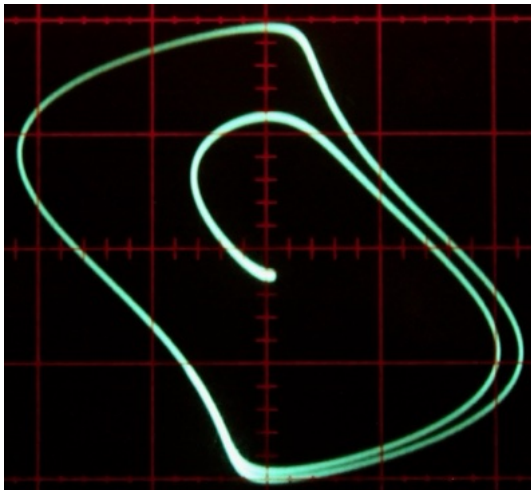
The unscaled analog computer program looks like this:



This problem must be scaled so that no variable exceeds the interval $[-1, 1]$ yielding the following program:



A typical phase space plot generated with this setup looks like this:



An analog computer setup for the two-dimensional heat-equation starts with

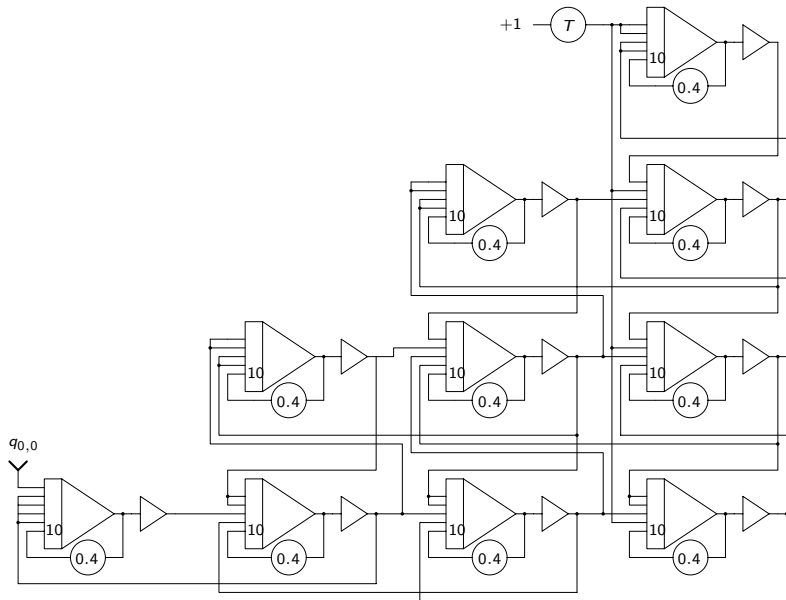
$$\dot{u} = \alpha \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right).$$

Since an analog computer can only integrate with respect to time, an obvious approach would be to replace the right-hand side of the above equation by a difference term:

$$\dot{u}_{i,j} = \alpha (u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1} - 4u_{i,j}) + q_{i,j}.$$

This can now be treated as shown before. In the following one octant of a quadratic sheet is modelled (taking a symmetry argument into account):

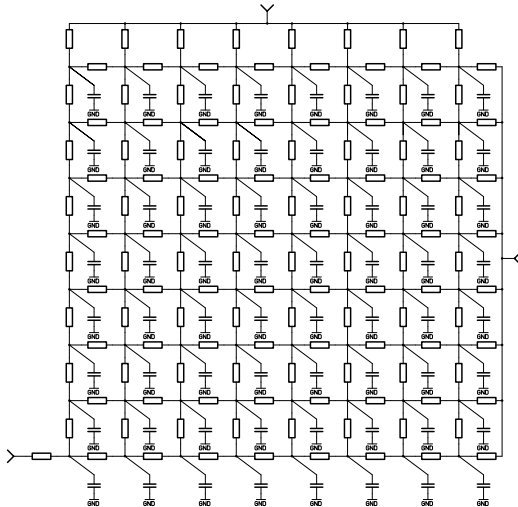
The heat equation



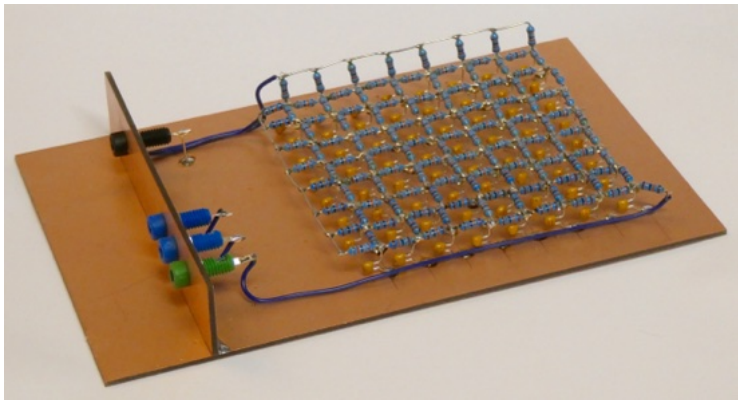
Passive indirect Analogies – the heat equation

A passive analogue

The following example shows a simple passive model for the two-dimensional heat-equation:

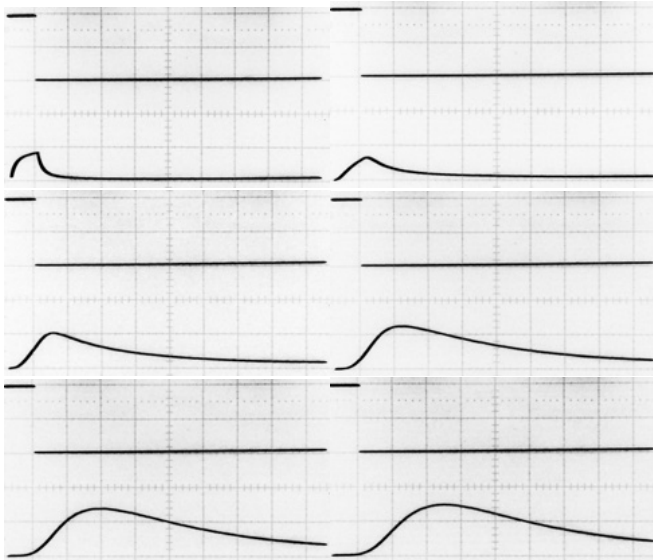


A passive analogue



A passive analogue

Response along the diagonal elements to an input pulse at $u_{0,0}$:



Steady state solution:

