FIRST STEPS
THE ANALOG THING

THE ANALOG THING

FIRST STEPS

anabrid

THE ANALOG THING – FIRST STEPS
English edition, version 2.01

Authors: Thomas Fischer and Bernd Ulmann

The authors gratefully acknowledge the valuable comments and suggestions they received for this booklet from Karl-Heinz Dahlmann, Chris Giles, Lars Heimann, Dirk Killat, Michael Koch, Sven Köppel, Max Peschke, and Joost Rekveld.

anabrid GmbH
Am Stadtpark 3
12167 Berlin
Germany
Phone: +49 177 5633531
Email: `THAT@anabrid.com`
Web: `https://www.anabrid.com`

Never connect this device directly to the main power line. Do not apply voltages greater than ±12 V to this device. The application of voltages greater than ±12 V to this device may cause damage to property, personal injury, or death.

This device is designed for users aged 12 years and older. Users below the age of 12 require adult supervision.

TABLE OF CONTENTS

1. WELCOME

Computing today is a digital monoculture. At anabrid, we want to see a comeback of analog computing and, eventually, an analog-digital hybrid computing future, because:

- Analog computing is vastly more energy-efficient than digital computing [1], promising to lower our computational CO_2 footprint and energy costs significantly.
- As the exponential growth of digital computing power described by Moore's Law approaches its limits [2], analog-digital hybrid technology promises to sustain growth in computer performance for decades to come.
- Analog computing may protect critical infrastructure by reducing attack surfaces of online operational technology and by offering failover options. [3]
- Analog computing is a fascinating intellectual and artistic activity. As a computing paradigm it is radically different from Turing machine-type thinking, offering an excellent way to learn about mathematics, science, and engineering. [4] It also enables creative expression when used, for instance, in conjunction with voltage-controlled audio synthesizers.

You are invited to join the quest to bring analog computing back. THE ANALOG THING (or simply THAT) is here to get you started. THAT is a high-quality, low-cost, open-source, not-for-profit cutting-edge analog computer. It computes with continuous voltages rather than with zeroes and ones. Capable of solving (sets of) differential equations, it enables the modeling of a broad range of dynamic systems. This booklet offers guidance to users taking first steps with THAT, focusing on the qualitative rather than quantitative aspects of analog computing. We hope you find this booklet, along with THAT, its online resources, and its community of users enjoyable resources in your learning inquiries and in our shared quest to make computing more diverse!

[1] Jennifer Hasler (2016). Opportunities in physical computing driven by analog realization, *2016 IEEE Int'l Conference on Rebooting Computing*, San Diego, CA.

[2] John L. Hennessy and David A. Patterson (2019). *Computer Architecture. A Quantitative Approach*, 6th edition, Morgan Kaufmann, Cambridge, MA, p. 3.

[3] Daniel E. Geer Jr. (2018). *A Rubicon*, Hoover Working Group on National Security, Technology, and Law, Aegis Paper Series No. 1801.

[4] George F. Lang (2000). Analog was not a computer trademark! *Sound and Vibration*, August 2000, 16–24.

2. REQUIREMENTS

To get started with THAT, the following items are required:

- a system to display the output of THAT (time-varying voltages). See Section 6 of this guide for options.
- a USB power supply such as a phone charger with a USB-C plug (not included)
- suitable cables and adapters to connect THAT to the output display setup used (not included)
- a set of 2 mm banana plug patch cables (included)

3. WHAT IS IN THE BOX?

THAT ships in a box that contains:

- 1 power cord: USB-A to USB-C
- 1 stereo RCA-to-RCA cable
- 1 set (30 pcs.) 2 mm banana plug patch cables
- 1 master-to-minion ribbon cable
- The First Steps booklet

The box does not include:

- an oscilloscope or other visual display system
- a USB power supply (many people have spare ones)
- BNC adapters/cables to connect THAT to an oscilloscope

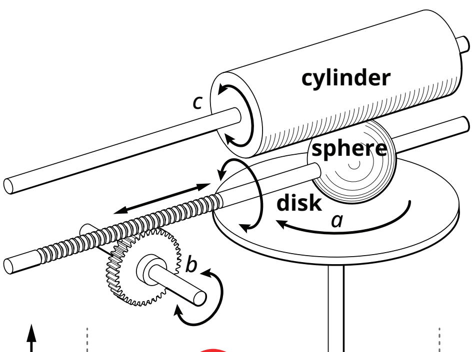
For a more comprehensive details and documentation of THAT, please check `https://the-analog-thing.org/docs`.
For accessories, please check the THAT shop at `https://shop.anabrid.com`.

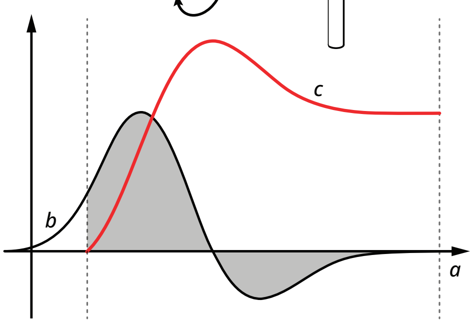
4. WHAT IS ANALOG COMPUTING?

Analog computing is one of the three computational paradigms: analog, digital, and quantum computing. Each of these paradigms has its own strengths and weaknesses, and they differ in both their development and prevalence. Digital computing is ubiquitous today, while quantum computing is in its infancy. Analog computing is tried and tested, yet has fallen out of fashion and is largely unknown today. THAT is here to change this because analog computing offers exceptional power efficiency and performance, both of which are much needed as the limits of Moore's Law are approaching.

Generally speaking, analog computing is about modeling dynamic systems, i.e., systems that change over time according to known relationships. Examples include market economies, the spread and control of diseases, population dynamics, nutrient absorption, nuclear chain reactions, and mechanical systems. Models of dynamic systems are useful similarly to how architectural models are useful in building design and crash test dummies are useful in car safety engineering. They offer insights into matters that would be too difficult, laborious, expensive, or harmful to study directly. Analog computing can serve a variety of purposes. It may help understand what is (models of), or it may help bring about what should be (models for). It may be used to explain in educational settings, to imitate in gaming, to predict in the natural sciences, and to control in engineering – or it may be pursued for the pure joy of it. Analog computing is also a great way to learn about calculus, science, and engineering.

Analog computers are modular and analog computer "programming" is a process of translating the behavior of a given system into patched connections between computing elements – the modules that make up an analog computer. As intermediate steps, this process requires that temporal behavior be described mathematically in the form of differential equations and, in turn, that these equations be converted into patch diagrams. While solutions of algebraic equations are single values, solutions of differential equations are functions – i.e., relationships that can be presented as graphs. Consequently, analog computers produce output in the form of (typically two-dimensional) graphs. All differential equations can be modeled with just a few kinds of computing elements: inverters, summers, multipliers, and, crucially, integrators.





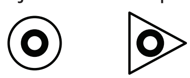
Analog computing has one of its major roots in the invention of the mechanical "disk-globe-and-cylinder" integrator. Comparable to how interlocked, different-sized gears multiply rotation rates and can thus be seen as performing mathematical multiplication, the disk-globe-and-cylinder integrator can be seen as performing mathematical integration by computing the area under a curve in x/y space. The disk-globe-and-cylinder, shown in the figure on the left, consists of a disk, a sphere (or globe), and a cylinder, each mounted to rotate.

Disk, sphere, and cylinder are configured with the circular surface of the disk touching the sphere and the sphere, in turn, touching the curved surface of the cylinder such that rotating either one of the three will, under most circumstances, turn the other two. In addition to its ability to rotate, the sphere can also be moved linearly back and forth along its rotational axis, which runs parallel to the disk's diameter, without ever losing contact with either disk or cylinder. With the disk rotating at a constant rate *a* to mark the passage of time, the input rotation *b* is made to follow a curve that changes over its horizontal time axis, moving the sphere across the diameter of the disk. The resulting output rotation *c* of the cylinder is then the integral of the area under curve *b*.

By feeding the output rotation of the cylinder *c* back to drive the input rotation *b*, the integrator can be used, in principle, to solve first-order differential equations and plot their solution graphs. Two integrators in series with the output of the second integrator fed back to drive the input of the first one allows a second-order differential equation to be solved, and so on. This working principle of mechanical analog computers can be implemented more efficiently using analog electronics, specifically by patching operational amplifiers such that they integrate their electrical inputs over time by charging and discharging capacitors.

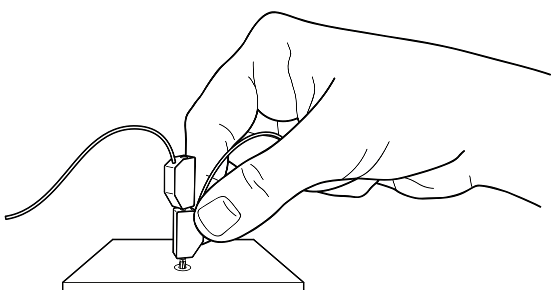
5. USING THAT

THAT is an electronic analog computer designed for desktop use to solve (sets of) differential equations. With its patch panel instead of keyboard, mouse, and monitor, its user interface differs noticeably from those of its digital stored-program cousins. The patch panel is divided into groups of analog computing elements such as integrators, summers, and multipliers. These are listed and explained in Section 7. Each computing element on THAT has one or more inputs and an output, accessible via jacks on the plug board. Input jacks are marked with circles, and output jacks are marked with triangles as shown below. See Section 8 for more details on the different types of jacks on the patch panel.



input output

THAT is patched by connecting computing elements into feedback circuits via patch cables – single-wire cables with banana plugs on both ends. Note that patch cable plugs do not insert completely into THAT’s patch panel as shown in the image below. This is explained in more detail on page 25. Multiple patch cable plugs may be stacked in order to connect multiple cables to the same jack.



The output of each computing element can be fed to one or more inputs of other elements. However, an input of a computing element may only take a single signal, such as an output of another computing element. To solve differential equations, chains of connected computing elements must contain at least one closed feedback loop. Feedback around one integrator allows first-order differential equations to be solved, feedback around two integrators allows second-order differential equations to be solved, and so on.

THAT represents quantities as time-varying voltages in a range with fixed boundaries. This range, called the *machine unit*, is thought of as –1 to +1. The actual voltage range THAT uses is –10 V to +10 V. Translating patterns of change in dynamic systems into differential equations and further into analog computer patches commonly involves the scaling of speed and quantities.

Scaling speed allows users to compress or stretch the independent variable time, typically by several orders of magnitude, to fit patch runtimes into convenient time frames for observation and measurement. In this way, the rapid decay of a volatile compound can be simulated slowly enough for observation and interactive manipulation, while population dynamics occurring over decades or centuries can be simulated in the blink of an eye. Scaling quantities is about multiplying all values occurring in a given patch uniformly, such that the greatest value occurring in the patch fits into and makes good use of the machine unit interval. A model of the global human population size, for example, representing millions of individuals, would not involve millions of volts but be scaled to fit within –1 to +1 (i.e., –10 V to +10V).

THAT outputs solutions of differential equations in the form of time-varying voltages. These can be captured by an attached digital computer for further processing, or they can be studied visually and interactively using an oscilloscope or a similar display system. Just as single-board digital computers allow “headless” operation without a monitor, THAT can operate without a visual display. Often, however, and especially when taking first steps with THAT, using a visual display system is advisable. Section 6 shows several options.

Once THAT is connected to a power supply and a display system, it is ready to be patched. This typically begins with an understanding of the quantitative relationships that are present in some dynamic system, expressed in the form of one or more differential equations. This then guides the development of a patching diagram that can then be implemented on THAT’s patch panel. Quantities enter the patch via coefficient potentiometers. These can be set at the beginning of the patching process by connecting the input of each potentiometer used to +1, setting the mode selector to COEFF, selecting the potentiometer using the coefficient selector knob, monitoring its output on the panel meter, and turning the potentiometer knob until the desired value is shown.

Setting THAT to OP, REP, or REPF mode will then start the patch to compute the unknown solution(s) of the differential equation(s) and output them as time-varying voltages that correspond to the temporal change in the modeled dynamic system. The most immediate way to read these solutions and manipulate them interactively is to visualize them live on an oscilloscope or a similar setup.

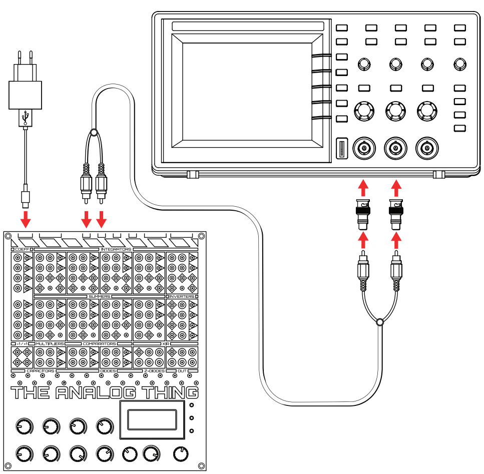
6. VISUAL DISPLAY SYSTEM SETUP

Patches running on THAT produce output in the form of time-varying voltages at various places on the patch panel. The most immediate way to read these solutions and manipulate them interactively is to visualize them on an oscilloscope or a similar visual display system. Up to four values can be selected and connected to a visual display system.

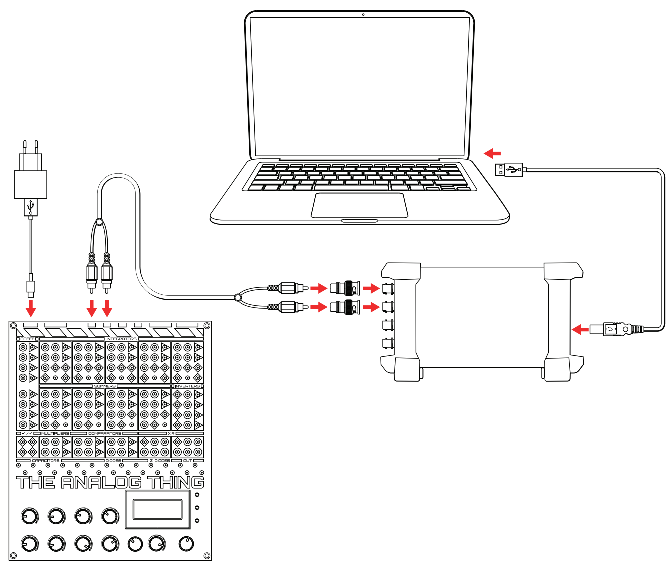
This section describes several ways to set up THAT with a visual display system. These vary considerably in terms of capabilities and affordability. Ideally, visual display systems used with THAT should be capable of visualizing multiple (two or ideally four) channels, frequencies of 200 kHz, and support x/y display mode. Less capable but more affordable systems such as software oscilloscopes with a single channel and a 20 kHz frequency limit offer less than ideal yet satisfactory performance for the applications covered in this guide.

6.1 OSCILLOSCOPE

THAT can be used with various kinds of oscilloscopes, such as conventional cathode ray tube oscilloscopes, digital oscilloscopes, and USB oscilloscopes in conjunction with PCs. Connect one or more of the outputs X, Y, Z, and U on the back of THAT directly to the inputs of the oscilloscope. Use suitable cables such as the included RCA-to-RCA cable in combination with RCA-to-BNC adapters (not included). Up to four channels can be connected if supported by the oscilloscope. Refer to your oscilloscope manual for setting the display mode, channel sensitivity, and time deflection. Connect the USB-C IN on the back of THAT to a USB power supply using a USB-C cable.



Using a desktop oscilloscope



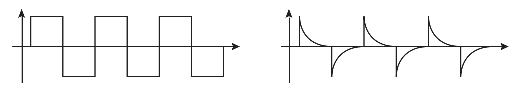
Using a PC with an external USB oscilloscope

6.2 PC WITH AUDIO INPUT INTERFACE

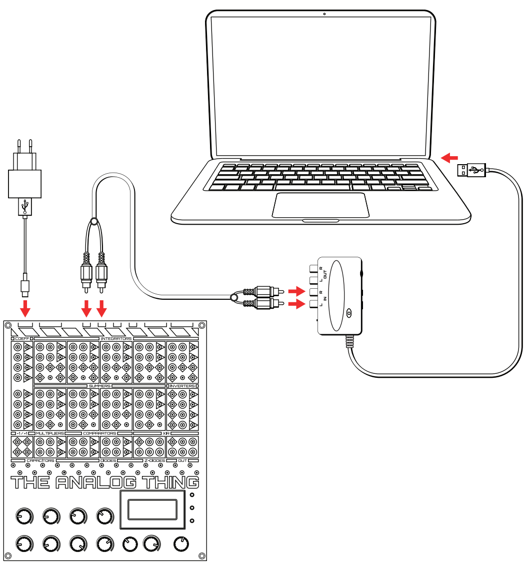
THAT can be used in conjunction with software oscilloscopes running on PCs and taking input via audio input interfaces, such as external USB audio interfaces and standard sound cards. Connect one or two of the outputs X, Y, Z, and U on the back of THAT to the RCA audio inputs of the audio interface using suitable cables and adapters. The RCA-to-RCA cable included with THAT is suitable for connections to USB audio interfaces with RCA jacks. Soundcards with a 3.5 mm jack require cables or adapters that are not included with THAT. The number of audio input channels available (typically two: one left and one right) determines the number of values that can be monitored. Install an oscilloscope software application on the digital computer and select the audio input interface as its input device. Connect the USB-C IN on the back of THAT to a USB power supply using a USB-C cable.

Audio input interfaces contain capacitors that alter signal waveforms by filtering out DC and low frequency components. Note the difference between the original waveform and the waveform measured using a sound card and software oscilloscope shown here:

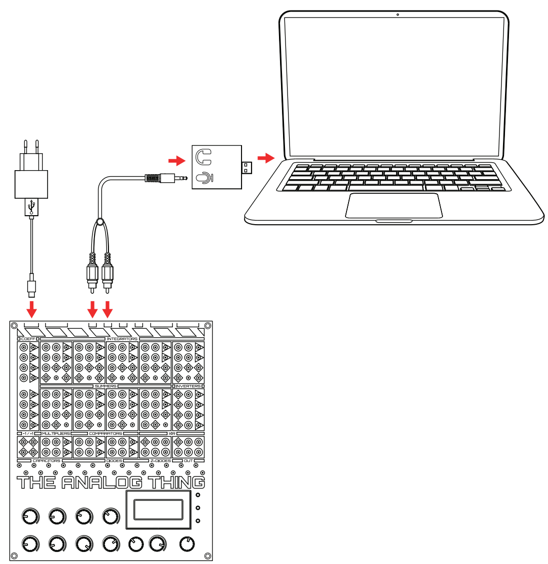
original: measured:



As qualitative demonstrations, the applications discussed in this guide are not seriously affected by this.

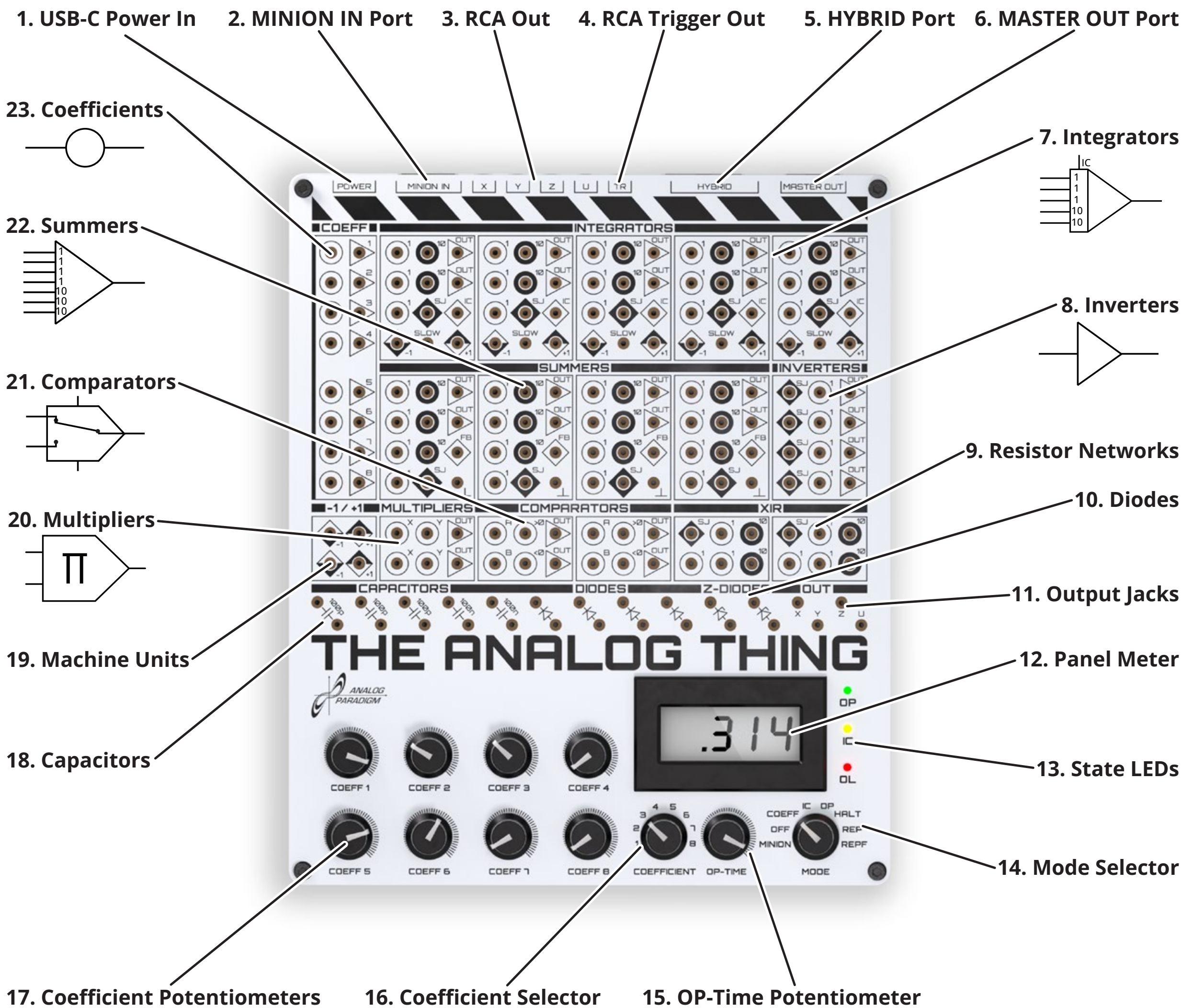


Using a PC with an external USB audio interface



Using a PC with a soundcard

7. FEATURES OF THAT



1. USB-C Power In. Power input jack (the USB data pins are not used). Use a USB power supply unit such as a phone charger or any other USB outlet with a USB-C cable.

2. MINION IN Port. When applications require more computing elements than are available on a single THAT, multiple THATs can be linked in a "minion chain" using their "MASTER OUT" and "MINION IN" ports. Connecting the MASTER OUT port of a THAT to the MINION IN port of another THAT with the included ribbon cable makes the first THAT the "master" and the second THAT its "minion." There is no limit to the number of THATs that can be linked in a minion chain.

3. RCA Out. The four RCA jacks labeled X, Y, Z, and U provide the signals plugged to the x, y, z, and u jacks on the patch field, attenuated to a ±1 V range, which is compatible with audio hardware such as sound cards, allowing software oscilloscopes to be used.

4. RCA Trigger Out provides a trigger signal for oscilloscopes when THAT is used in REP (repeat) or REPF (repeat fast) mode.

5. HYBRID Port allows THAT to be controlled from digital devices, enabling the development of analog-digital hybrid programs. Outputs on the HYBRID Port are attenuated and shifted to 0 V to 3.3 V to facilitate analog/digital conversion.

6. MASTER OUT Port. See 2. MINION IN Port.

7. Integrators integrate the sum of their (weighted) input values over time. Each integrator has five inputs (two of which are weighted by factor 10) and one output via two output jacks. The IC (initial condition) input allows the start value of an integration to be set. Integrators change the sign of their output values implicitly.

8. Inverters yield the input value with the opposite sign.

9. Resistor Networks can be used to add inputs to an integrator, summer, or inverter by connecting the SJ jack of a resistor network with the SJ jack of the computing element.

10. Diodes and **10 V Zener Diodes** support various applications.

11. Output Jacks. Patch cable connections from any output on the patch panel into these jacks make those outputs available at the RCA Out ports as well as via the HYBRID port at the back of THAT, shifted to a narrower voltage range (see 3. RCA Out above).

12. Panel Meter displays values or operation times, depending on the Mode Selector position. In COEFF mode, the output value of the coefficient potentiometer selected using the Coefficient Selector is shown in the unit range 0 to 1. In the MINION, IC, OP, or HALT modes, the value patched into the U OUTPUT jack is shown in the unit range –1 to 1. In REP mode, the OP TIME is shown in the 0 to 10 sec range. In REPF mode, the OP TIME is shown in the 0 to 100 ms range.

13. State LEDs display device states: OP=patch running, IC=initial condition mode, OL=overload.

14. Mode Selector controls the operational mode of THAT: In COEFF mode, the value of the coefficient selected using the COEFFICIENT selector is displayed on the Panel Meter. In IC mode, the outputs of the integrators are set to the values applied to their respective IC (initial condition) inputs (with the opposite sign) to set the stage for a patch run. In OP (operate) mode, the current patch is run. In HALT mode, integration is suspended and outputs are held at their last values. In REP (repeat) mode, THAT repeatedly follows a process of briefly entering IC mode, then OP mode. This allows steady graphs to be displayed on an oscilloscope. REPF: As REP mode, but 100 times faster. In MINION mode, THAT is controlled by another THAT that operates as the MASTER.

15. OP-Time Potentiometer sets time spent in OP mode between 0 and 10 seconds in REP (repeat) mode and between 0 to 100 milliseconds in REPF (repeat fast) mode.

16. Coefficient Selector selects the coefficient potentiometer value displayed on Panel Meter in COEFF mode.

17. Coefficient Potentiometers are used to change the values of coefficients (see 23. Coefficients).

18. Capacitors support various applications.

19. Machine Units provide –1 and +1 unit values.

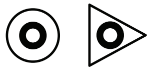
20. Multipliers multiply values supplied to their inputs.

21. Comparators allow conditional switching. If A+B>0, then the input to > is available at the two output jacks, otherwise the input to < is available at the two output jacks.

22. Summers add up values supplied to their inputs. Each summer has seven inputs (three of which are weighted by factor 10) and one output via two output jacks. Summers invert the sign of their output values implicitly.

23. Coefficients. Input and output jacks of the Coefficient Potentiometers.

8. TYPES OF PLUG BOARD JACKS AND THEIR USES



input output

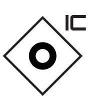
This section describes the different jacks on the plug board of THAT. As already mentioned in Section 5, input jacks are marked with circles, and output jacks are marked with triangles.

Jacks marked with diamond shapes split horizontally into a white and a black half provide machine unit values. The unit value −1 is provided via jacks marked with diamond shapes whose lower halves are black. The unit value 1 is provided via jacks marked with diamond shapes whose upper halves are black. These jacks are available in the machine unit section of the plug board (see Item 19 in Section 7 above) and grouped with the integrators.

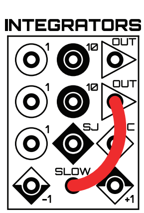
unweighted input

input weighted by factor 10

The integrators, summers, and resistor networks (XIR) have inputs marked with white circles and the number 1 and inputs marked with black circles and the number 10. The numbers indicate weightings, i.e., factors by which input signals are multiplied. Signals plugged to inputs marked with white circles and the number 1 remain unchanged. Signals plugged into inputs marked with black circles, and the number 10 are multiplied by factor ten. For example, a signal with the value −0.02 plugged into an input marked with a black circle, and the number 10 will be converted internally to the value −0.2.

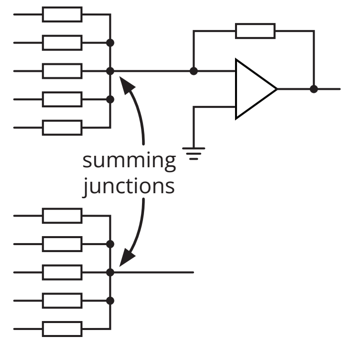


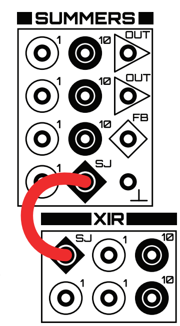
Each integrator has a jack marked with a white diamond shape and the label IC. This jack is used to set the initial condition (IC) of the respective integrator, i.e., that integrator's (inverted – see Item 7 in Section 7) output value at the beginning of a patch run.



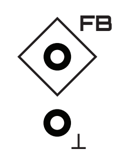
Each integrator has a jack labeled SLOW. Connecting an integrator's output to its SLOW jack, as shown in the patch diagram on the left here, reduces the integrator's operating speed to 0.01 of its regular speed. Without this connection, i.e., at the normal operating speed, an input of −1 and a run beginning at IC=0 leads to the (inverted) output of +1 in 1 ms. With this connection, i.e., in the slow operation mode, the input of −1 and a run beginning at IC=0 lead to the output of +1 in 100 ms.

Each integrator, summer, inverter, and resistor network (XIR) has a jack labeled SJ, which is the abbreviation for Summing Junction. The following two circuit diagrams show the summing junctions between a resistor network that sums the inputs to an operational amplifier (the core component of integrators, summers, and inverters) as well as in a stand-alone resistor network (XIR):

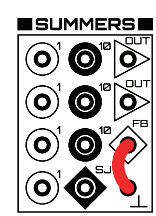




The SJ jacks on integrators, summers, and inverters allow adding further inputs to these elements by connecting the summing junction of a given element to the summing junction of a resistor network. The patch diagram on the right here shows the addition of inputs to a summer by way of connecting a resistor network (XIR) via their summing junction jacks.



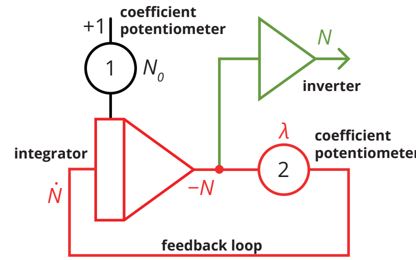
Each of the summers has a jack marked with a white diamond shape and the label FB (feedback), as well as a jack marked with the ground symbol (⊥), which gives access to THAT's ground. In combination, these two jacks can be used to disable a given summer's feedback resistor, turning it in effect to an open amplifier. For this purpose, the FB Jack needs to be connected to the ground (⊥) jack, as shown in the patch diagram here on the right. Open amplifiers are typically used to implement inverse functions. Combinationed with a multiplier, for example, an open amplifier can implement a division or a square root.



9. APPLICATIONS

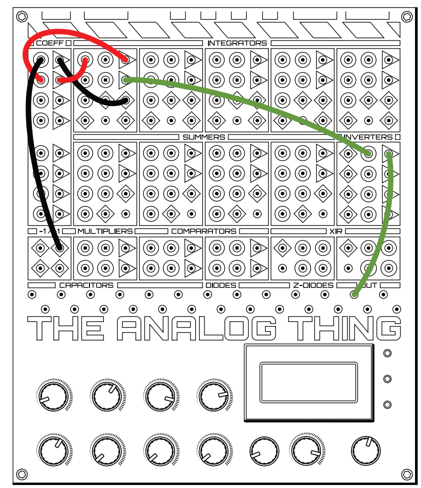
This section presents six analog computer applications from several fields, exemplifying typical modeling workflows from the translation of differential equations into patch diagrams to patch panel connections and parametric exploration.

9.1 RADIOACTIVE DECAY



A minimal analog computer patch consists of an integrator whose output is fed back as its input. Imagine rotation *c* in the image in Section 4 being fed back to drive rotation *b*. Depending on the rotational direction of this feedback, *c* plotted over *a* results in an either exponentially growing or exponentially decaying curve and thus offers a model for phenomena with exponential dynamics. Let's give this a try electronically by setting up an analog computer patch to model the decay in a radioactive sample. Exponential decay can be described with the first-order differential equation shown here.

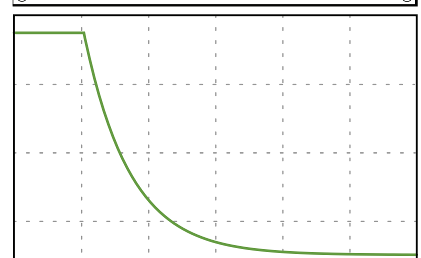
$\dot{N} = -N\lambda$ **differential equation**



In this equation, λ denotes the decay coefficient. Note that $\dot{N}$ in Newton's notation is the equivalent of *dN/dt* in Leibniz's notation. To translate this equation into a patching diagram, the Kelvin feedback technique can be used. As a first step, this technique typically requires arranging the equation(s) such that the highest derivative is isolated on the left side of the equal sign. Showing $\dot{N}$ isolated on is left, the equation above is already arranged in this way. Establishing the feedback loop required to solve the equation can now take advantage of the equality of both sides of the equation. For this purpose, the highest derivative on the left of the equal sign is, for the time being, assumed to be known. The term on the right of the equal sign $-N\lambda$ is then modeled with a suitable chain of computing elements. In the radioactive decay example, this is simply an integrator connected to a potentiometer that implements the decay coefficient. The output of this chain of computing elements, known to be equal to the highest derivative, is then connected to the input of the first computing element in the chain, closing the feedback loop. In the radioactive decay example, a second potentiometer is needed to set the size of the sample at the start of the patch run (i.e., the initial condition of the integrator).

As integrators change the signs of their output, feeding an integrator's output back to its input is sufficient to model decay (i.e., negative exponential growth). To display diminishing positive values, however, the initial value of the integrator's output must be negative, and the sign of the output must be changed using an inverter. Two coefficient potentiometers allow the initial number of molecules N_0 and the decay coefficient λ to be adjusted.

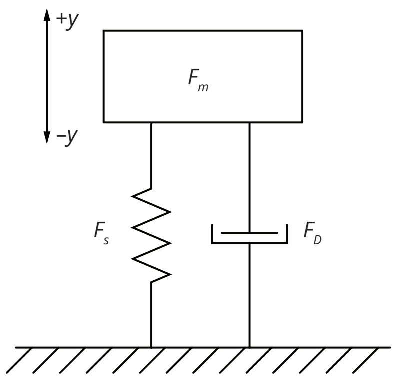
To run this patch, put THAT in COEFF mode. Plug cable connections on the patch panel as shown in the patch diagram. Use the coefficient selector to select coefficient potentiometers 1 and 2 in turn and set both to 0.5. Connect your display system to RCA Out X on the back of THAT. Run the patch in REP (repeat) mode to display a flicker-free image on the display system. As the patch runs, change coefficients 1 and 2 and observe how the graph changes. Also, adjust the OP-Time and run the patch in REPF mode to observe how these changes affect the output.

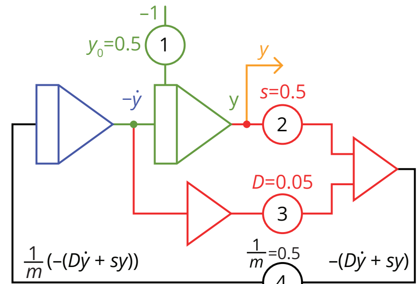


Output: decay curve

9.2 MASS-SPRING-DAMPER SYSTEM

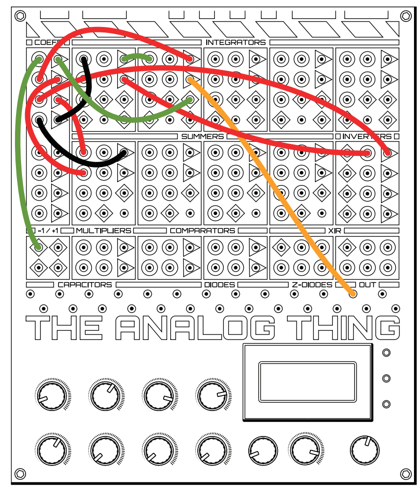
Vehicle suspensions absorb bumps in the road to provide comfortable and safe rides. A typical suspension system includes a spring and a damper, which support the mass of the vehicle, its passengers, and cargo. By selecting the ideal spring and damper settings for a given mass and impact force, suspensions systems are tuned to a “sweet spot” called critical damping. In this condition, the suspension absorbs as much impact energy as possible and returns to equilibrium without overshooting and oscillating. Testing suspension characteristics for varying masses and impact forces tends to be infeasible in the field, which makes analog computer modeling an excellent alternative. As a first step, this requires a description of the system of interest in the form of one or more differential equations arranged such that the highest derivative is isolated on the left of each equation. To describe a suspension system in this way, we start by noting that the sum of the forces exerted by mass, spring, and damper is zero at all times:





$$F_m + F_s + F_D = 0$$

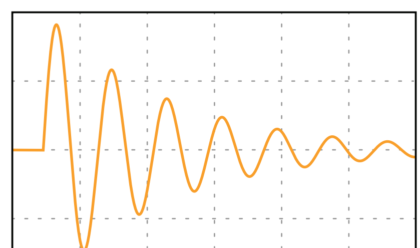
According to Newton’s second law of motion, F_m is mass m times acceleration a. The force with which the damper resists movement F_D is a damping coefficient D times the speed v of its vertical displacement. The force exerted by the spring F_s is a spring coefficient s times its vertical displacement y. The speed v is the first derivative of vertical displacement over time, which we denote by $\dot{y}$, and the acceleration a is the second derivative of vertical displacement over time, which we denote by $\ddot{y}$. This yields $m\ddot{y} + D\dot{y} + sy = 0$ or, resolved for the highest derivative $\ddot{y}$:



$$\ddot{y} = \frac{1}{m}(-(D\dot{y} + sy))$$

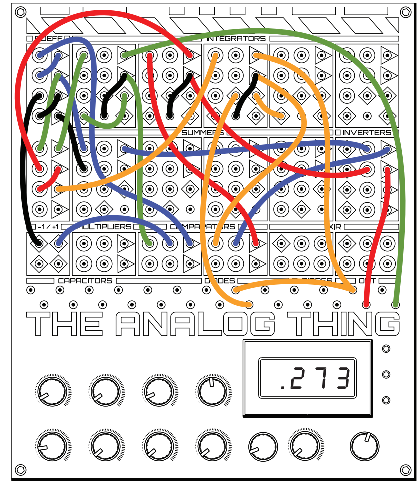
Developing a patching diagram from this second-order differential equation takes advantage of the equality of both sides of the equal sign. Assuming that $\ddot{y}$ is known, we model the term on the right of the equal sign using two integrators and feed the resulting lower derivatives, with coefficients applied and summed, back to the input of the first integrator, as shown in the diagram on the top right.

Run the patch in REPF (repeat fast) mode at 80 ms OP-Time to view a flicker-free image on the display system. As the patch runs, change the settings of coefficient potentiometers 1 through 4 and observe the suspension dynamics change. This patch also applies to damped oscillators in scenarios other than vehicle suspension tuning, for example in earthquake safety engineering and electronic circuit design.



Output: displacement of an underdamped suspension

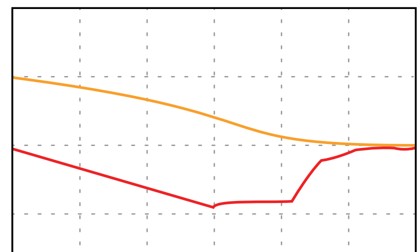
9.3 LUNAR LANDING



"Houston, Tranquility Base here. The Eagle has landed." Neil Armstrong radioed to Houston Mission Control after landing the Apollo 11 Lunar Module on the surface of the Moon on July 20, 1969. Spacecraft Communicator Charles Duke replied from Houston, "Roger, Tranquility. We copy you on the ground. You got a bunch of guys about to turn blue. We're breathing again. Thanks a lot!" Commander Neil Armstrong flew the lunar landing approach in tight interaction with pilot Buzz Aldrin and the Lunar Module Guidance Computer. Descending from a lunar orbit, the lander moved on an initially near-horizontal but soon increasingly steep downward trajectory. As lunar surface features came into view ever more clearly from lower and lower altitudes, Armstrong repeatedly used a joystick to redesignate the targeted landing site, aiming for terrain features suitable for a safe landing.

Meanwhile, the computer ensured a steady descent rate, which Armstrong could adjust using an (increase)-neutral-(decrease) toggle switch. Controlling the rate of descent in this way, Armstrong was able to trade fuel for time during which safer landing sites could be spotted and reached. In doing so, he had to use the fuel conservatively, to ensure a safe touchdown. After Armstrong decided to fly beyond a boulder field, the lander finally touched down with less than 5 percent fuel left in its descent stage. To make matters worse, sloshing motions of the fuel inside the tanks caused false, even lower fuel level readings, and the computer threw up a total of five false program alarms during the landing approach, turning the first manned lunar landing into one of the most hair-raising maneuvers of NASA's lunar program.

The analog computer patch shown here simulates the powered descent of a lunar lander during its final approach, putting you in charge of the descent engine throttle. A system of three differential equations models vertical velocity v under the influence of lunar gravity g, altitude h, and fuel level F remaining from a limited initial supply. Coefficient potentiometer 1 is your descent engine throttle. Set coefficient potentiometers 2 through 7 to the values shown in the diagram below. Connect h to OUT X, v to OUT Y. Connect F to OUT U to monitor the fuel level on the panel meter. Connect your oscilloscope or other display system to RCA Out X and Y on the back of THAT. Set the display system to "roll mode." Run the patch in THAT's OP mode and use coefficient potentiometer 1 to control your descent engine's burn rate and thrust. Watch your altitude and vertical velocity on the display system and monitor your fuel level on the panel meter. Touch down as gently as you can. Good luck and Godspeed!

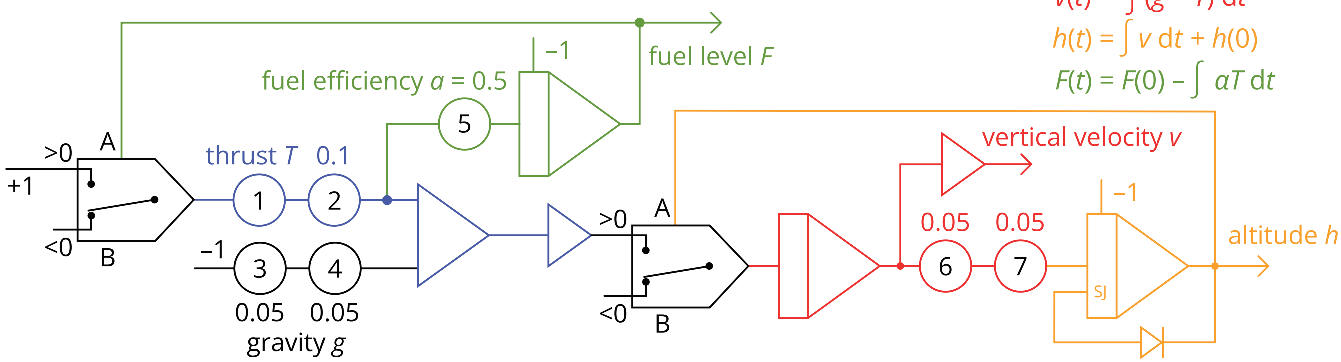


Output: spacecraft altitude and vertical velocity

$$v(t) = \int (g - T)\, dt$$

$$h(t) = \int v\, dt + h(0)$$

$$F(t) = F(0) - \int \alpha T\, dt$$

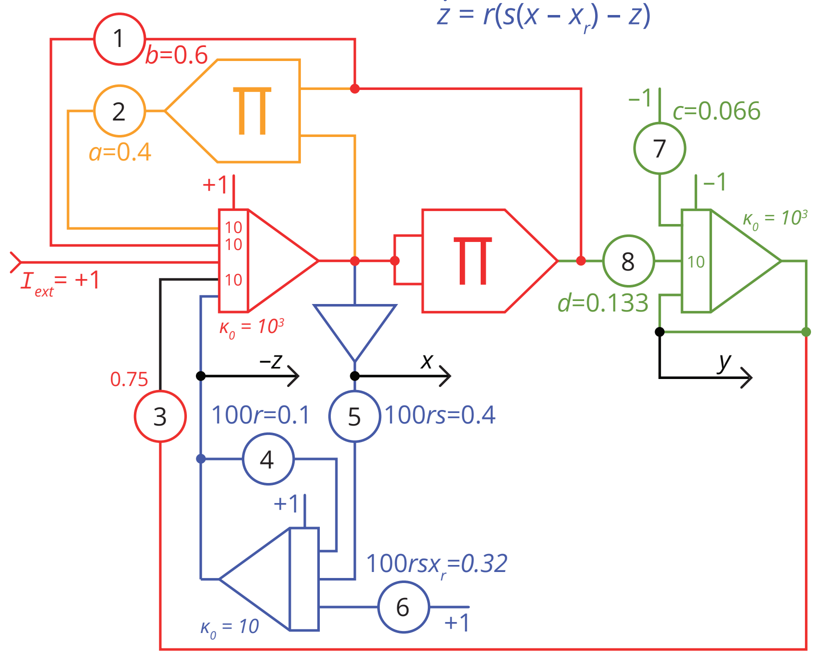


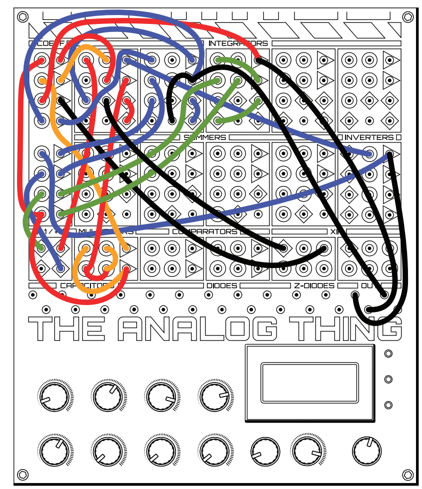
9.4 NEURONAL BURSTING

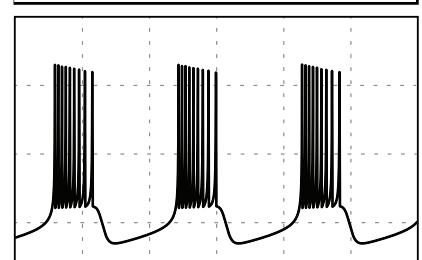
Neurons in the central nervous system receive nerve impulses from other “upstream” neurons and, if such inputs exceed their firing thresholds, give off nerve impulses to “downstream” neurons. Computationally speaking, the aggregation of incoming nerve impulses (also referred to as summation) is a process of integration, albeit within a limited time window, such that incoming impulses soon expire, and only the most recent ones carry weight in the integration as time passes. A few sporadic impulses are usually not enough to exceed the firing threshold. Instead, it takes either multiple upstream neurons to fire together at least near-simultaneously, or a single upstream neuron to send multiple impulses in short bursts. James L. Hindmarsh and R. Malcolm Rose proposed a model for this neuronal bursting in 1984. The model, consisting of the three first-order differential equations shown below, responds to inputs to I_{ext}. The variables x, y, and z correspond to the neuron’s (bursting) output potential, the transport of sodium and potassium through fast ion channels, and the transport of other ions through slow channels, respectively.

To add a 10-weighted input to the first (red) integrator, a resistor network is connected to its summing junction (SJ) jack. To operate the second (blue) integrator more slowly than the other two, its output is patched to its SLOW jack. Set the coefficient potentiometers to the values shown in the patching diagram. Connect x (the output of the inverter) to OUT X and connect your oscilloscope or other display system to RCA Out X on the back of THAT. If more channels are available on your visual display system, connect y and $-z$ accordingly. Run the patch in OP mode and observe the image on the display system while connecting and disconnecting I_{ext} to and from +1.

$$\dot{x} = -ax^3 + bx^2 + y - z + I_{ext}$$
$$\dot{y} = -dx^2 + c - y$$
$$\dot{z} = r(s(x - x_r) - z)$$






Output: neuronal burst pattern

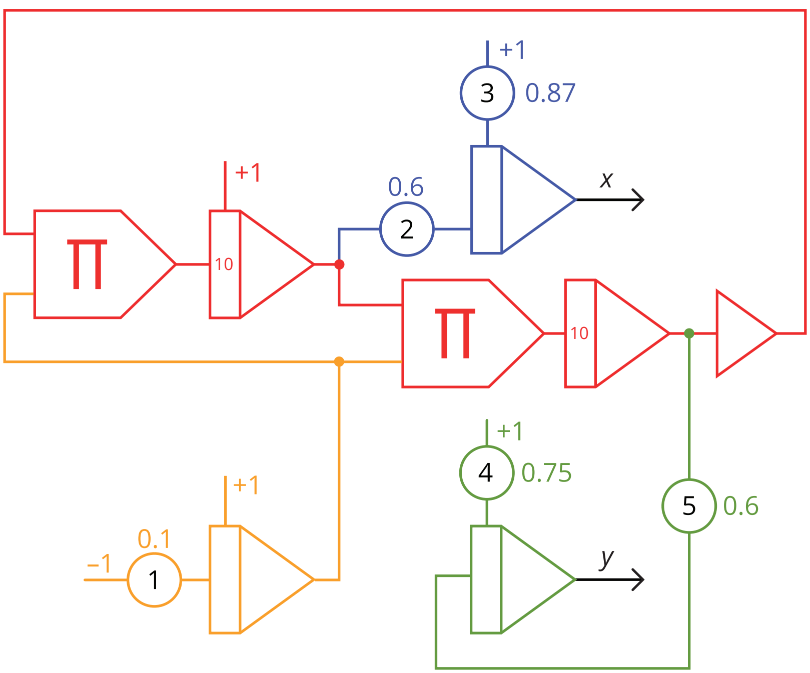
9.5 EULER SPIRAL

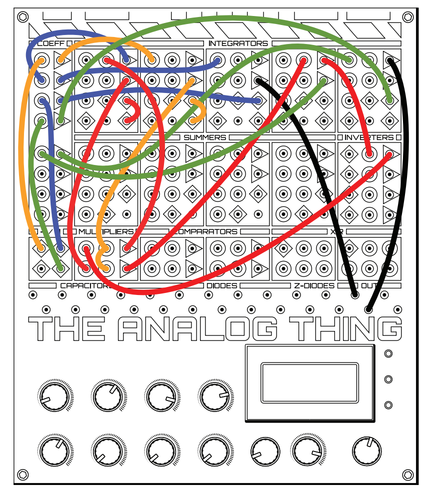
The Euler spiral is defined as a curve whose curvature increases linearly with its length. With the sign of its curvature following the sign of its length, the Euler spiral takes the shape of a double spiral with point-symmetry around the origin. Besides its visual appeal, the Euler spiral can be used to design smooth transitions between straight lines and circular arcs in two dimensions, which has practical value in several fields. An obvious and immediate application area is graphic design. The design of typography, in particular, requires numerous smooth transitions between straight and circular line segments. Another application area is road and rail traffic network design. To appreciate the benefits the Euler spiral offers in this area, imagine, as an alternative, the immediate tangential connection between straight and circular road segments. This would require motorists to change steering radii instantaneously, which is not only physically impossible but uncomfortable and dangerous to attempt. For similar reasons, the Euler spiral is used to generate movement paths in high-speed machinery and robotics. It has also been found to offer a good description of the curvature of rat whiskers.

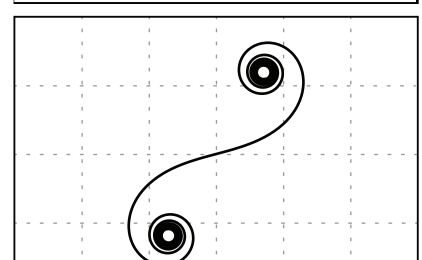
To display the Euler spiral, a display system capable of visualizing two channels in x/y mode is required. Set the coefficient potentiometers 1 through 5 to the values shown in the patching diagram. Connect x (the output of the inverter shown in blue in the patching diagram) to OUT X and connect y (the output of the inverter shown in green in the patch diagram) to OUT Y. Connect your oscilloscope or other display system to RCA Out X and Y on the back of THAT. Set the display system to x/y mode and run the patch in REPF (repeat fast) mode. Vary the settings of the coefficient potentiometers 1 through 5 as well as the OP-Time potentiometer and observe the changing image on the display system.

$$x(t) = \int_0^T \cos\left(\frac{t^2}{2}\right) dt$$

$$y(t) = \int_0^T \sin\left(\frac{t^2}{2}\right) dt$$





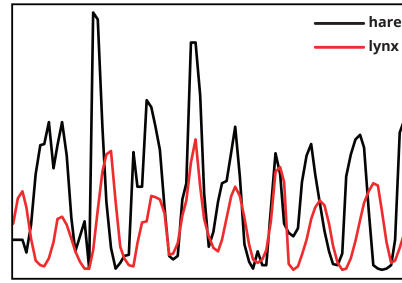


Output: Euler spiral

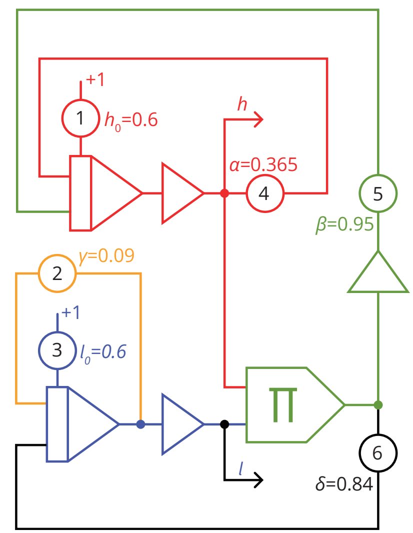
9.6 HUNTER PREY POPULATION DYNAMICS

How do the population sizes of hunter and prey species interact? An initial quantitative insight into this question was found in the early 20th century in the trading records of the Hudson's Bay Company, a Canadian-based fur trading business at the time. The company recorded the numbers of lynx and snowshoe hare pelts it bought from trappers, thereby creating an indicative historical documentation of the changing sizes of Canadian lynx and snowshoe hare populations. These records – the 1845 to 1935 data is shown in on the right – reveal a curious pattern of periodic and temporally related increases and decreases in both population sizes.

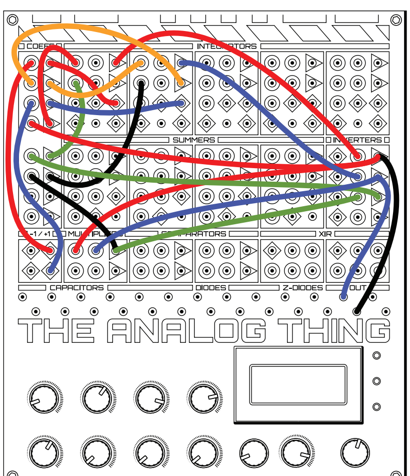
As the lynx prey almost exclusively on the snowshoe hare, it was unsurprising to see the size of the hare population affected by the size of the lynx population. It seemed puzzling, however, that the size of the lynx population also seemed to be affected by the size of the hare population, leading some to ask: "Do hare eat lynx?" An explanation for these predator-prey population dynamics was proposed independently by Alfred J. Lotka in 1925 and Vito Volterra in 1926 in the form of the following two first-order differential equations:



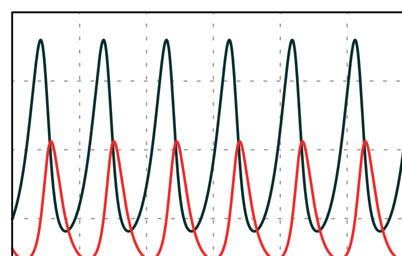
$$\dot{h} = (\alpha - \beta l)\, h$$
$$\dot{l} = (\delta h - \gamma)\, l$$



Variable h refers to the size of the hare population, while variable α refers to the growth rate of the hare population (based on their consumption of plant life, which is assumed to be abundantly available in the environment). Variable l refers to the size of the lynx population, and variable δ refers to the growth rate of the lynx population (based on their consumption of the changing hare population). The natural death rate of lynx is denoted by variable γ, and the rate at which hare are killed by lynx is denoted by variable β. As it turns out, hare do, of course, not eat lynx, but an excessively large lynx population will overhunt the hare population, thereby depleting the lynx's food source, which, in turn leads to a decimation of the lynx population.



To display the changing population sizes of both the predator and the prey species simultaneously, a visual display system capable of visualizing at least two channels is required. Set the coefficient potentiometers 1 through 6 to the values shown in the patch diagram on the right. Set THAT to OP mode and your visual display system to "roll mode." To explore the dynamics of both interacting populations further, change the settings of coefficient potentiometers 1 through 6 interactively as the patch runs. If possible, also try visualizing the output in x/y mode on your display system. The result is a so-called phase-space plot for the modeled predator prey relationship. This patch can be used to model competitive relationships beyond ecological systems, for example in economic systems and communicable diseases.

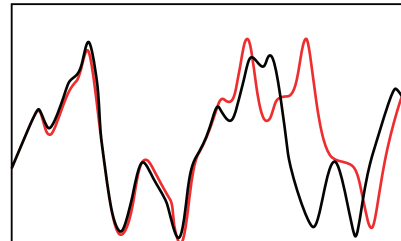

Output: changing population sizes

9.7 LORENZ ATTRACTOR

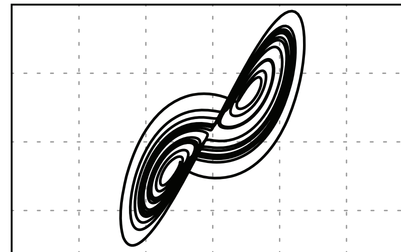
Until the mid-20th century, it was generally thought that the accuracy of model-based predictions depended entirely on the quality of the models used and the accuracy of their input data. Consequently, feeding roughly accurate input into a good model was assumed to yield roughly accurate output. This assumption was challenged in 1961 when Edward Norton Lorenz tested a weather model described by a set of coupled differential equations.

Having run the model (on a digital computer) and reviewed an output plot, Lorenz decided to produce a more extended plot by running it again with the same input data for a longer duration. He restarted the model and went for a cup of coffee. Upon his return, Lorenz found a surprise: Initially, the new plot resembled the first plot closely, but it soon deviated and took on a distinctly different shape, as shown on the top right of this page. The first program run read an input value of 0.506127 from the computer's memory. Lorenz entered this value manually for the second run, reading it from the first plot, which showed it rounded to three places: 0.506. This deviation of less than of one part in a thousand caused the two plots to soon take on rather different shapes. Instead of attributing the difference between the two plots to a glitch, and despite his weather model being rather simplistic, Lorenz realized that the dissimilar plots exemplified why it is so difficult to predict the weather and complex dynamic systems in general. In such systems, tiny differences in earlier states can lead to vast changes in later states: the butterfly effect!

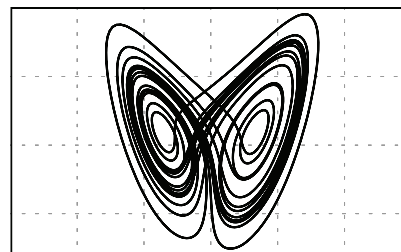
Lorenz's model is one of many chaotic (or "strange") attractors that have since been found. You can implement it on THAT, as shown in the diagrams on this page. To visualize the characteristic three-dimensional pattern of the Lorenz Attractor, switch your display system to x/y mode and visualize any two of the values x, y, and z marked below.



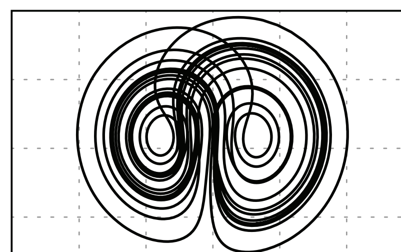
Data plots of Lorenz's two program runs with near-identical initial coefficients



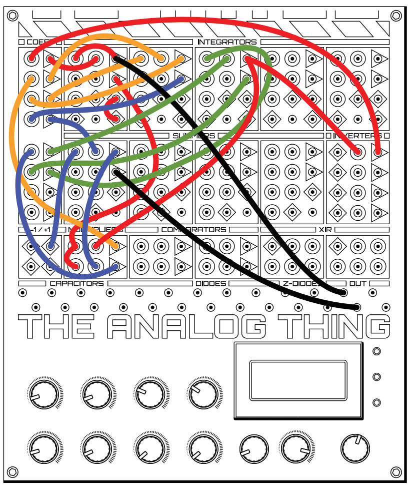
Output in the x/y plane: top view



Output in the z/x plane: front view



Output in the z/y plane: side view



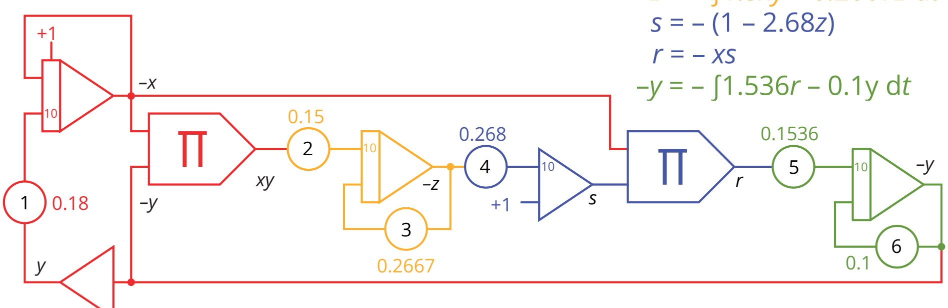
$$-x = -\int 1.8y - x \, dt + C$$

$$-z = -\int 1.5xy - 0.2667z \, dt$$

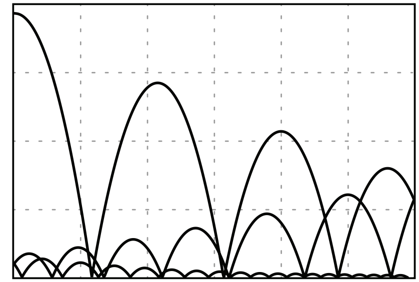
$$s = -(1 - 2.68z)$$

$$r = -xs$$

$$-y = -\int 1.536r - 0.1y \, dt$$

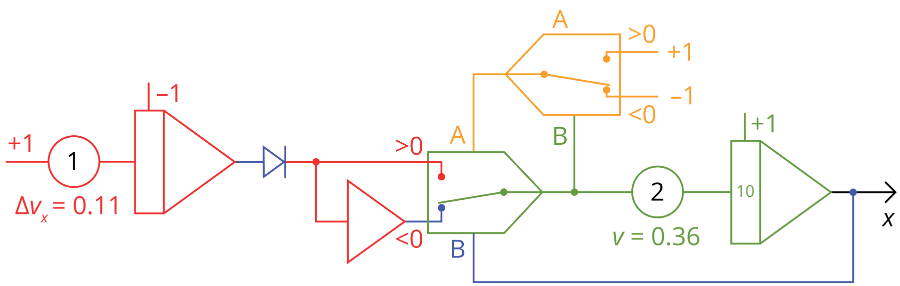


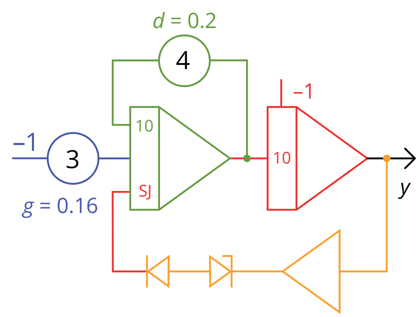
9.8 BOUNCING BALL

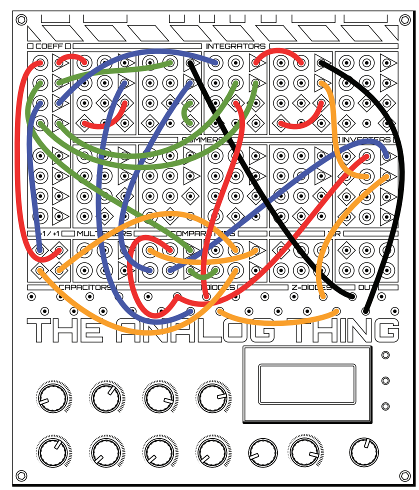


As an archetypal theme in computational physics simulation, game and demo development, and programming education, bouncing balls occupy a special place in the history and culture of computing. Here, we simulate the motion path of a bouncing ball dropping in a two-dimensional "box" with machine unit width and height, taking into account gravity and drag due to air resistance.

Assuming the horizontal *x* and the vertical *y* components of the ball's movig position to be independent from one another, these can be computed by two seperate circuits. The circuit computing the *x*-component of the ball's position (shown on the left below) changes the ball's horizontal direction each time the ball hits either the left or the right wall of the box. Its horizontal position is obtained by the second (green) integrator, which begins on the left of the screen (IC=+1) and integrates over the ball's velocity *v*, which, in turn, is decreased linearly over time by the first (red) integrator.







The *y*-component of the ball's position is that of a free-falling object bouncing back elastically when hitting the floor. It is computed by the circuit shown in the diagram on the right above and can be described with the following differential equation:

$$\ddot{y} = -g + d\dot{y} \begin{cases} -c(y-1) & \text{if } y > 1 \\ +c(-y-1) & \text{if } y < 1 \end{cases}$$

The term on the left of this equation describes the ball's free fall under the influence of gravity *g* and drag *d* due to air resistance. The two conditions shown on the right of this equation are satisfied when the ball hits either the floor or the ceiling of the box, respectively. Beginning its fall at the ceiling (IC=−1 at the fourth, red integrator) and then losing energy due to drag, the ball will not reach the ceiling again, such that the lower condition on the right of the formula will never be satisfied. The upper condition, however, will be satisfied repeatedly each time the ball hits the floor. The Zener diode becomes conductive at that instant and momentarily adds upwards velocity to the third (green) integrator. Connect the TRIG output of THAT with the external trigger input of your oscilloscope (if available) and run the analog computer in REPF (repeat fast) mode.

9.9 POLYNOMIAL GENERATOR

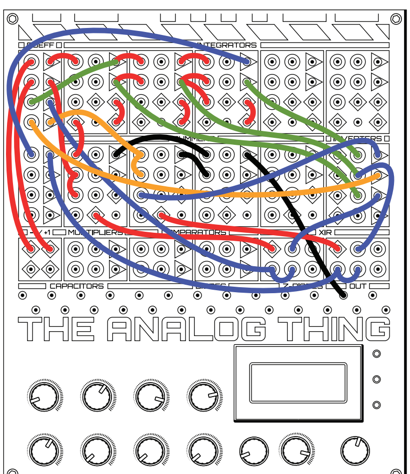
Polynomials are a family of functions that are sums of power functions. Their graphs are smooth, differentiable curves without discontinuities such as gaps, or sharp turns. A polynomial whose highest power is three is called a cubic polynomial. Its general form is:

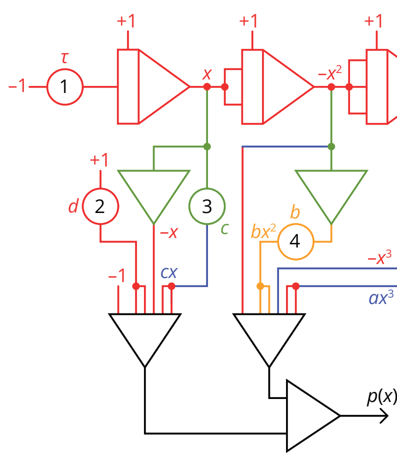
$$p(x) = ax^3 + bx^2 + cx + d$$

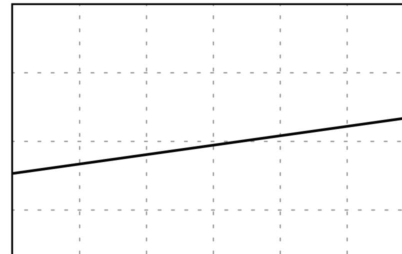
Choosing suitable degrees of polynomials and manipulating their coefficients (*a*, *b*, *c*, and *d* in the above example) allows approximating a wide variety of smooth graphs within certain value ranges. This process is useful in numerous scientific and engineering contexts and beyond. For example, it allows modeling smooth motion paths in robotics, interpolating sampled or otherwise incomplete data, reconstituting noisy signals, and synthesizing various timbres of sound.

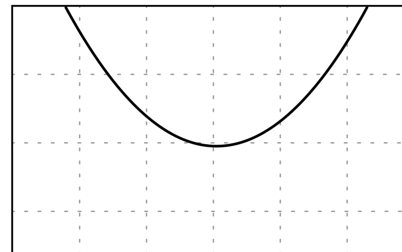
Polynomial functions can relate to analog computing in two basic ways. They can be used to feed varying inputs into analog computer programs, or, as solutions of differential equations solved using analog computers, they can be outputs of analog computer programs. Here, we focus on the second scenario and program a polynomial generator – a utility for the flexible modeling of arbitrary cubic polynomials, with the values of *a*, *b*, *c*, *d* as well as the polynomial being restricted to the machine unit interval −1 to +1.

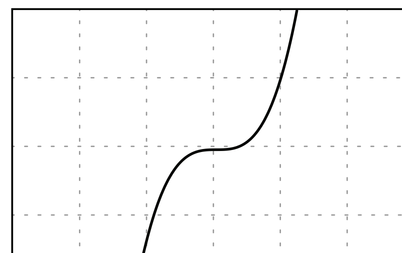
The wiring of the coefficient potentiometers 2 through 5 corresponds to the helper function described in Section 10.4, allowing to set the values of *a*, *b*, *c*, and *d* within the entire machine unit range of −1 to +1. Connect the TRIG output of THAT with the external trigger input of your oscilloscope (if available) and run the analog computer in REPF (repeat fast) mode.

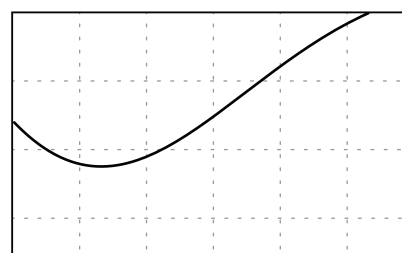





Output: basic term for x

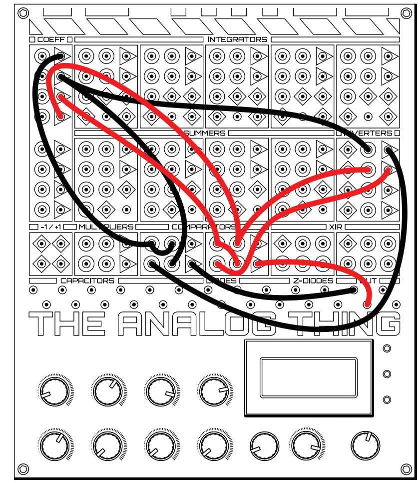

Output: basic term for x^2

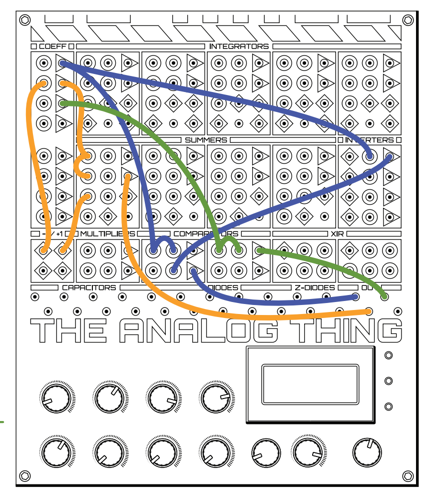

Output: basic term for x^3


$p(x) = -0.3x^3 + 0.4x^2 + 0.7x + 0.1$

10. HELPER FUNCTIONS

This section presents five helper functions that allow shaping values in various analog computer applications. Besides their utility in larger applications, these helper functions also make for good beginner's exercises − each standing alone or in various combinations. The five circuit diagrams shown here correspond to the setups in the two patch diagrams using the same cable colors.

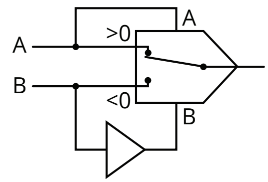




10.1 MAXIMUM OF TWO VALUES

This function takes two input values *A* and *B* and gives as its output whichever of the two input values is greater.

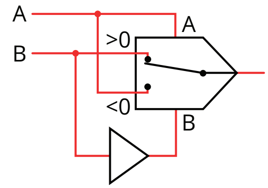
$$\text{out} = \max(A, B)$$



10.2 MINIMUM OF TWO VALUES

This function also takes two input values *A* and *B*. As its output, it gives whichever of the two input values is smaller.

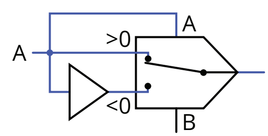
$$\text{out} = \min(A, B)$$



10.3 ABSOLUTE VALUE

This function takes one input value *A* and gives as its output the absolute value of *A*. In other words, it gives *A* when *A* is positive, the positive value of *A* when *A* is negative and 0 when *A* is 0.

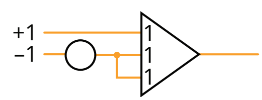
$$\text{out} = \text{abs}(A)$$



10.4 ADJUSTABLE VALUE -1 to +1

While the conection of a coefficient potentiometer to the machine units supply gives an adjustable value in a range of either −1 to 0 or 0 to +1, this function offers an adjustable value within the entire machine unit range of −1 to +1.

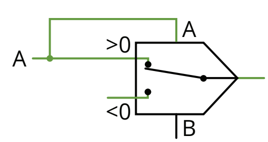
$$\text{out} = -1 \ldots +1$$



10.5 NON-NEGATIVE VALUES ONLY

This function takes one input value *A* and gives as its output *A* when *A* is greater than 0. Otherwise, it gives 0.

$$\text{out} = A \text{ if } A>0$$
$$\text{out} = 0 \text{ if } A\leq 0$$



11. FREQUENTLY ASKED QUESTIONS

Is THAT a general-purpose computer?
Yes and no. The term general-purpose computer is commonly used to describe digital stored-program computers that can execute arbitrary algorithms. While THAT does not belong in this category, it is a general-purpose analog computer in that it can solve any (set of) differential equation(s) within the means of its computing elements. By connecting multiple THATs in master/minion chains, it is possible to implement arbitrarily large analog computer patches involving any number of computing elements.

Why do the plugs not go fully into the patch panel?
The 2 mm plug cables were originally designed to plug entirely into a corresponding type of gold-plated socket. One of these sockets plus mounting costs about USD 1.00, which would add up significantly for the 186 plug positions on THAT's patch panel. We saved most of this cost by using an extra-thick top circuit board with appropriately-sized, gold-plated through-holes. Since the length of the plugs is greater than the thickness of the circuit board, we placed stop-limits below the patch field to ensure that the small, contact-assuring springs halfway along the length of each plug make reliable contact.

How precise is THAT compared to a digital computer?
THAT is precise to about two positions after the decimal point, relative to its machine unit (±1). Comparing the precision of analog and digital computers is a bit like comparing apples and oranges. Analog computers usually handle quantities based on measuring only ("What is your body height?"). Digital computers, however, also handle quantities based on counting ("How many siblings do you have?"), which requires strict numerical precision. Consider this: A bank clerk getting the third decimal place of an interest rate wrong commits a severe error, while a tailor being off by a few micrometers when taking a client's measurements has no such problem. Furthermore, numerical digital computing involves rounding, and hence rounding errors, which can add up quickly in iterative loops. Analog computers do not operate numerically and do not round. In this sense, the great precision of today's digital computers helps minimize a problem that is specific primarily to digital computing. Representing quantities as continuous voltages, THAT does not suffer from many issues that are inherent to binary value representations. While analog computer solutions can be affected by noise and instabilities, the precision of THAT is perfectly appropriate for most analog computer applications.

THAT is switched on, but the display is blank. Why?
This can happen when one or more patch connections create short circuits. Ensure that none of the outputs of computing elements are connected to ground or each other and that the +1 signal is not connected to ground (⊥).

The integrators run into overload too quickly. Why?
This can happen when there are conductive fingerprints or dirt on the patch panel. Gently cleaning the panel should resolve the issue.

If THAT is powered by USB, i.e., by 5 V⎓, then how is it possible that its machine unit is physically ±10 V?
Powered via a USB-C socket, THAT is indeed supplied with 5 V⎓ in the first instance. The relatively bulky, cuboid-shaped component near to the USB-C socket on the upper left of THAT's base PCB, is a DC/DC converter, which turns a 4.5 V⎓ to 5.5 V⎓ input into a ±12 V output. This output powers all the main functions of THAT, allowing its ±10 V machine unit.

With outputs varying between −10 V to 10 V, how can THAT model smaller or greater quantities?
Translating patterns of change in dynamic systems into mathematical representations and further into analog computer programs commonly involves the scaling of quantities. Quantities are represented on analog computers in a voltage or current interval with fixed boundaries called the machine unit. On THAT, this interval is −10 V to +10 V. For the sake of simplicity, the machine unit is generally thought of as ± 1, regardless of the actual voltage or current interval of a given analog computer. To model arbitrary quantities on THAT, they can be scaled to make efficient use of the machine unit. Output can then be converted back to the original scale.

How can I use THAT to create useful models of very fast or very slow phenomena?
Translating patterns of change in dynamic systems into mathematical representations and further into analog computer programs commonly involves the scaling of speed. THAT allows compressing or stretching the independent variable time by several orders of magnitude. In this way, the rapid decay of a volatile compound can be simulated slowly enough for observation and interactive manipulation, while population dynamics occurring over decades or centuries can be simulated in the blink of an eye.

12. USEFUL RESOURCES

Various resources are available to make your explorations and use of THAT enjoyable and rewarding. Here are several related web and social media references. These will help you stay up-to-date with all things THAT and get in touch with other members of the analog computing community.

THAT Online

- THE ANALOG THING:
  `https://the-analog-thing.org`
- Analog Paradigm:
  `https://analogparadigm.com`
- anabrid GmbH:
  `https://www.anabrid.com`
- THAT Shop:
  `https://shop.anabrid.com`

Social Media

- THAT on Facebook:
  `https://www.facebook.com/groups/theanalogthing`
- Analog Paradigm on Twitter:
  `https://twitter.com/analogparadigm`
- anabrid GmbH on LinkedIn:
  `https://www.linkedin.com/company/anabrid`
- THAT Open Hardware on GitHub:
  `https://github.com/anabrid/the-analog-thing`

Sometimes, when we are out and about, when there is no THAT at hand, or as a matter of personal preference, it can be useful to have a way to think through analog programming projects on paper before implementing them. For this purpose, you can download a printable paper template from the THAT website at this URL:
`https://the-analog-thing.org/THAT_template.pdf`

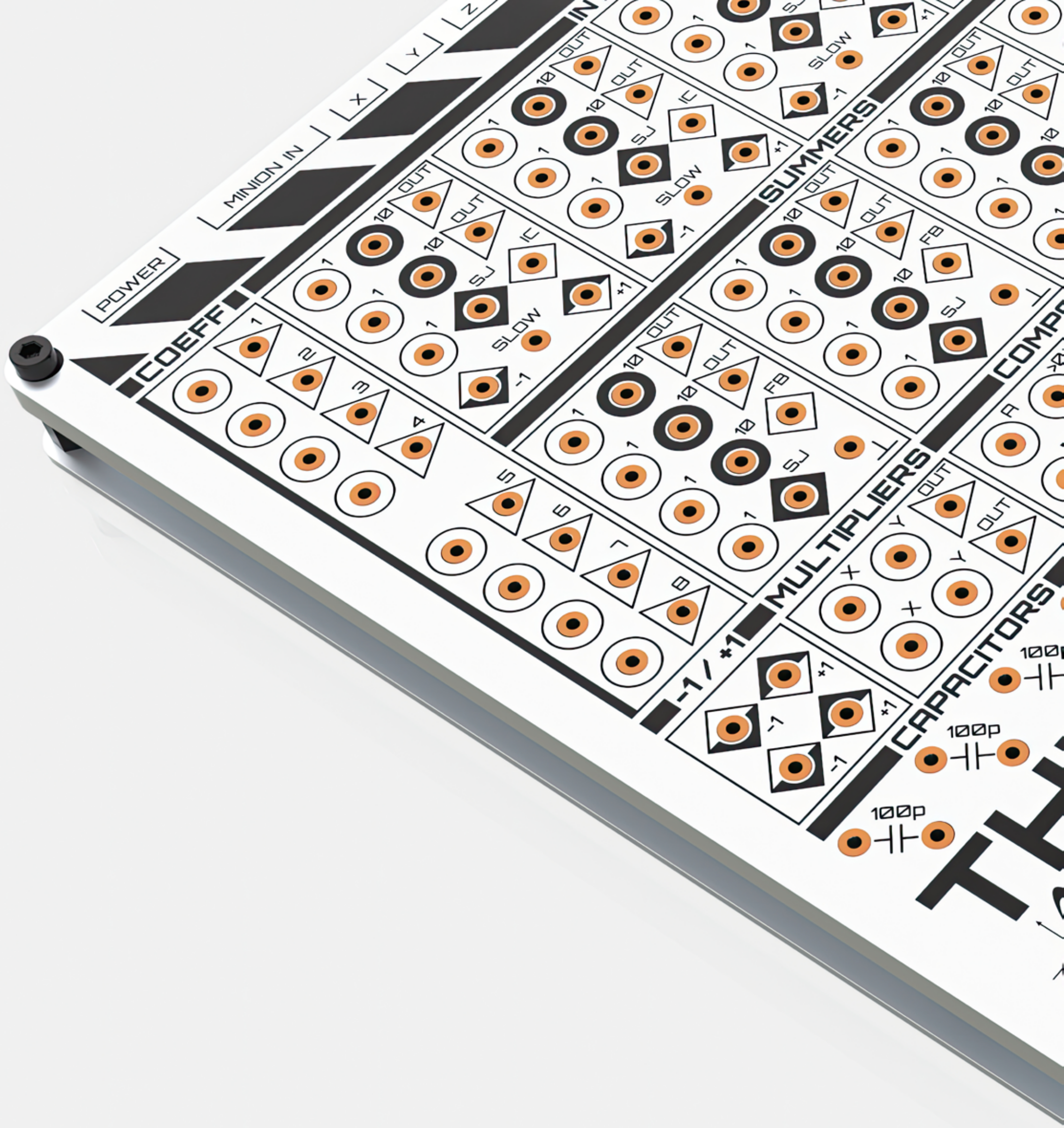
13. FURTHER READING

Books

- Bernd Ulmann (2023). *Analog and Hybrid Computer Programming*, DeGruyter, Berlin.
- Bernd Ulmann (2023). *Analog Computing*, Oldenbourg Wissenschaftsverlag, Munich.
- Bruce J. MacLennan (2012). Analog Computation. In: Meyers Robert A., ed., *Computational Complexity. Theory, Techniques and Applications*, Springer, New York, pp. 151–184.
- Charles Care (2010). Technology for Modelling. Electrical Analogies, Engineering Practice, and the Development of Analogue Computing, Springer, London.
- James S. Small (2001). *The Analogue Alternative. The Electronic Analogue Computer in Britain and the USA, 1930–1975*, Routledge, London and New York.

Online

- THE ANALOG THING online documentation:
  `https://the-analog-thing.org/docs/dirhtml`
- Thomas Fischer (2021). The Analog Way to Compute:
  `https://medium.com/@7f15ch3r/b8a2ca4a762d`
- Charles Platt (2023). The Unbelievable Zombie Comeback of Analog Computing, *WIRED*:
  `https://www.wired.com/story/unbelievable-zombie-comeback-analog-computing`
- Veritasium (2022). Future Computers Will Be Radically Different (Analog Computing):
  `https://youtu.be/GVsUOuSjvcg`


POWER
MINION IN
X
Y
Z
COEFF
SUMMERS
MULTIPLIERS
CAPACITORS
-1 / +1
100p

anabrid